Tailored Campus Event Finder

A Major Qualifying Project Proposal

Team Members:

Zahr Lyttle	Management Information Systems
John Ross	Management Information Systems
Brandon Malofsky	Business - Operations
Matthew McCarthy	Computer Science
Tyler Bennett	Computer Science

Project Advisors:

Professor Diane Strong

Assistant Teaching Professor Wilson Wong

Date: March 23th, 2017

Abstract

The Tailored Event Finder is an application designed for both students looking for events and event organizers on the WPI campus. The application gives both students and event organizers a central place to focus their efforts in promoting and finding events. Students can filter events based on their interests and club memberships and organizers can target events at different demographics. In addition to event promotion, the application provides methods to indicate attendance at events giving organizers metrics to expect at events. This paper tracks the application's development as well as requirements gathering and testing.

Executive Summary

According to the 2009 National Survey of Student Engagement (NSSE Studies), "student success is directly linked to student involvement." This means that the level of a student's involvement affects the satisfaction of their overall college experience and success. Worcester Polytechnic Institute (WPI) embraces this philosophy as the university acts as a hub for over 200 student run organizations. Unfortunately, with so many clubs, organizations, and university departments vying for the attention of the student body, many struggle with broadcasting their events in an effective manner.

WPI's Student Government Association and Marketing Department attempted several technical solutions and workarounds in recent years to remedy the issue, but experienced significant shortcomings in their efforts. For example, the implementation of Campus Labs' OrgSync Portal was intended to solve the club management and event publication dilemma WPI organizations faced, but the technology failed to deliver a comprehensive system for advertising club events. Additionally, the licensing of OrgSync dwarfed initiatives of organizations and departments seeking better alternatives and services because WPI is under a three-year contract for the expensive OrgSync service until summer 2017. Whereas recent solutions to the club event publicizing issue incorporate OrgSync in the solution in some manner, our team decided to not incorporate the legacy software in our solution.

Our project addresses the lack of efficient and effective club event marketing by creating a completely new platform rooted in the feedback and suggestions of WPI undergraduates. The goals that drove our project include:

1. Identifying the basic features needed by both the student body and campus organizations;
2. Developing an Event Finder application that allows clubs and organizations to publicize their event and meets in one location, so that student can view what activities are occurring around campus;
3. Improving and testing the application using agile methodologies; and,
4. Evaluating the effectiveness of the application in terms of student satisfaction with the application, perceived usefulness, and perceived ease of use.

Using a combination of software development life cycle and agile scrum methodologies, our team sought to create a campus event finder application specially designed for WPI undergraduates and clubs.

During the software development life cycle phase of our project, otherwise called our architectural spike, our team discovered the initial business requirements for the application, determined our development environment, and designed/developed our minimal viable application. To discover our initial business requirements, we performed a competitor analysis on existing campus event applications and surveyed over 140 WPI undergraduates about their preferences for features and functionalities.

From our responses and research, we realized users wanted a dynamic application that allows them to view event details and set preferences to filter event/club content. We decided to utilize the AngularJS/Ruby on Rail framework as it was useful for creating web/mobile applications that enable user-generated content and user-specific feeds, and we developed the minimal application by the end of A-Term. Despite the application only allowing users to post and view events, our adoption of agile practices accelerated our development and significant functionalities were shortly added to the application.

The agile scrum phase of our development utilized incremental design practices rooted in adapting to the needs and wants of users. Using sprint planning meetings, biweekly/weekly sprints, sprint reviews, and sprint retrospectives, our team produced worthwhile increments to our applications and avoided developing features unwanted by our users. During our agile scrum development phase, we held a focus group with club presidents to gain a better understanding of event planner's needs and interviewed WPI undergraduates about additional improvements they wanted in the application. Features mentioned by the parties included: designing Event Feeds in an understandable way such as Instagram, tracking attendance and interest to an event, and quick menu access to clubs and events. Using their feedback, we created these functionalities along with others mentioned during the feedback sessions.

Additionally, we sought the consultation of WPI's Marketing Department who guided our development decisions and served as our liaison to WPI's Information Technology (IT) Department to integrate IT's Central Authentication System (CAS) into our application. Unfortunately, IT could not assist with our integration during our project timeline, so our team pivoted to beta-testing of the application, rather than full implementation.

By the end of C-Term, we delivered on its initial promises by fully developing an application specific to the needs of WPI undergraduate students and event planners. The functionalities for general users provide personalization with features like subscribing to clubs for event information and expressing interest in an event. For event planners, the application allows them to create and edit club/event content for their clubs. Additionally, the application remembers the club admin permissions of users through a user profile system which requires users to log in.

After development, we fully tested the application through a series of rigorous unit, integration, and system tests to assure the application's reliability. Finally, we performed acceptance testing utilizing the Technology Acceptance Model (TAM). Four (4) of our ten beta-testers partook in the survey which gauged the perceived usefulness and ease of the application. The application is more than satisfactory in its perceived usefulness but can still improve its ease of use. Based on these results and additional feedback from stakeholders, we recommend the following actions be taken in the future:

1. Integrate the WPI IT’s Central Authentication System into the Application;
2. Continue development based on the product backlog and user feedback;
3. Develop in iOS and Android languages to optimize the mobile application experience;
4. Work with Student Activities Office and SGA to maximize club outreach and marketing;
5. Conduct feasibility study of the use of the application at other universities, especially the Worcester Consortium.

Acknowledgements

Our team would like to express our gratitude to the following individuals and groups that support us throughout the project process.

We would like to first thank our faculty advisors, Diane Strong and Wilson Wong. Your insight, guidance, and advice have been paramount to our success. We will always be thankful for the trust and confidence you had in us to reach our project's goals. Your willingness to support us at every point throughout the project will always be appreciated.

Our team wanted to also thank the WPI Marketing Department for their support and insight through the project. Your assistance was greatly appreciated throughout the project.

Next, we wanted to thank the WPI Organization Presidents and students who were willing to help our team gather opinions, insight, and valuable information to ensure our project's success.

Lastly, we want to thank Worcester Polytechnic Institute and the Foisie School of Business for allowing us the opportunity to take a diverse team and create a lasting and valuable MQP project. We will take the lessons we learned this year to help us succeed into the future.

Table of Contents

Table of Figures

1. Introduction

1.1 Problem Being Addressed & Project Viability

For many college students, the experiences and the memories created during extracurricular activities, such as Student Government or a Greek organization, are just as valuable as what is learned in the classroom. Per the National Survey of Student Engagement in 2009 (NSSE Studies), "student success is directly linked to student involvement." This means that the level of a student's involvement affects the satisfaction of their overall college experience and success. An expert in the field of higher education, Alexander Astin, wrote that, "...the strongest single source of influence on cognitive and affective development is a student's peer group; the greater the interaction with peers, the more favorable the outcome." and that, "...the amount of personal development and learning that can occur [in a student] is directly proportional to the quality and quantity of student involvement." (Astin, 1999). This asserts that the activities college students partake in during their college careers are just as impactful and beneficial as experiences within group projects or inside the classroom.

Through preliminary research and surveys, our group has found that many college students feel that they either do not know what activities different organizations are hosting on their campus at any given time or that the process of finding such information is very difficult. This information is important for student retention on the part of the university because students need to be engaged in social activities to succeed and develop into well rounded graduates.

To produce the best graduates, universities need to maintain an interest in not only creating the best academic programs, but also encouraging students to develop and sustain an engaging environment on campus. Promoting student involvement provides students the opportunity to gain experience and skills that can boost a resume, and facilitates a large amount of personal development that is critical for a successful college experience. Astin defines student involvement as, "the amount of physical and psychological energy that the student devotes to the academic experience" (Astin, 1999). It is often this personal development gained from student involvement that distinguishes students and determines whether they are offered a job after college or not. There is a correlation that having an involved student body improves the quality of graduates but is often also, "...the catalyst for other institutional and community improvements" (OrgSync, n.d.). Therefore, universities benefit greatly in many ways by having and encouraging a high level of student involvement on campus.

Many universities have more than 100 different on campus organizations for students to learn and explore life beyond the classroom, but only a few institutions provide the resources or tools to facilitate and encourage their students to get involved with these organizations. Most of the resources that are available are not focused on regularly promoting student involvement and personal development, but rather have a minor set of resources to address the topic.

With over 89% of college students now using either a laptop, smartphone, or tablet on a regular basis, universities have adjusted their methods of communicating and providing services to their students (Pearson, 2015). Per Hollis Poll, in 2015 over 64% of college students used their smartphones for school related work and tasks (Pearson, 2015). That number is expected to continue to rise in 2016 which means that universities need to provide a way to facilitate and encourage student involvement on their campuses; with major considerations directed toward the use of mobile technology to do so.

Worcester Polytechnic Institute (WPI) is a university of nearly 4,300 undergraduate students located in Worcester, Massachusetts. With over 200 organizations on campus, many WPI students and organizations such as WPI's Student Government Association are raising the issue of over-programming and how it has become incredibly difficult to determine exactly what is happening on campus at any given time. Many clubs and departments tried to solve this issue by creating their own digital newsletters, sending out emails, and even handing out paper flyers. None of these methods seem to solve the issue; they only add to the confusion. WPI faces a problem in which its student body feels they are not aware or are unable to easily determine what activities clubs and organizations are hosting on campus at any given time. Additionally, clubs feel they are unable to effectively promote their organizations and events for the same reason. WPI realizes the significance of promoting and maintaining a high level of involvement within the student population and has begun to investigate a possible solution to this issue. The primary stakeholders in this issue are the university, students, and campus organizations.

After conducting research and evaluating the current solutions available at universities in the Worcester area, our group feels that the most viable way to solve this problem is by creating a mobile application that can centralize the information for all campus organizations. With this app, students can easily determine exactly what activities are happening on campus, and receive suggestions and details for activities of possible interest.

1.2 Goals

To find a solution to the lack of information and awareness about student activities on university campuses, our project goal is to create an event finder application that grants WPI students access to information regarding what activities are taking place around campus and suggest activities the individual will likely be interested in attending. To develop an effective solution to this issue we:

1. Identified the basic features needed by both the student body and campus organizations;
2. Developed an Event Finder application that allows students to view what activities are occurring around campus;
3. Improved and tested the application using agile methodologies; and,
4. Evaluated the effectiveness of the application in terms of student satisfaction with the application, perceived usefulness, and perceived ease of use.

2. Literature Review

2.1 Current Relationship Between Students and Technology

The current college student relationship with technology is constantly adapting and growing. Where decades ago technology was rarely used by youth or university students, now mobile devices and overall technology usage has become a constant in the daily lives of college students. According to a 2014 EDUCAUSE report, “86 percent of undergraduates owned a smartphone as of [2013], and nearly half (47%) owned a tablet.” Today, technology has become fully integrated into the daily lives of students; influencing how they communicate, learn, act, and receive information. A Pearson Review College study further confirmed the high technology usage of university students where 80% of students use a mobile device (phone) or laptop regularly for personal use and 90% for school related activities. Furthermore, 78% of the 1200 students surveyed by the Pearson Review preferred a smartphone over any other mobile technology available such as tablets or laptops. This demonstrates exactly how integral technology has become in universities (Pole, 2015b).

To ensure success of our proposed application we reviewed a multi-year study at the University of Central Florida (UCF). Reviewing this study validated that college students use mobile technology to obtain information, navigate, and manage their campus social lives. The survey and figures below display mobile device usage and ownership between 2012 and 2014 among students at UCF. The data reinforced Pearson’s survey results highlighting smartphone ownership at 95% of all university students. Smartphone ownership, while prevalent, was shown alongside 57% of students also owning tablets (Bauer, 2015a).

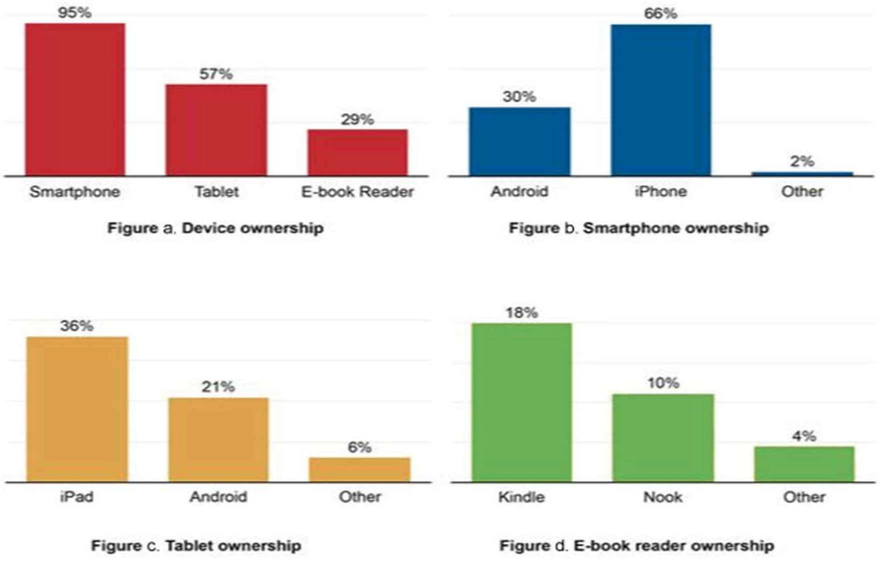


Figure 1 - Ownership of Electronic Devices in Universities. (Bauer, 2015b)

Beyond technology ownership, UCF's survey analyzed the most popular application categories and the changes among them over the two-year study. The data showed a usage growth of navigation, entertainment, and college specific applications within the two-year period. The usage growth in these three sectors further demonstrates and asserts the viability of a WPI campus event finder application as it would fall under both the entertainment and university specific category. The graphs below illustrate the usage of application categories measured in the UCF study.

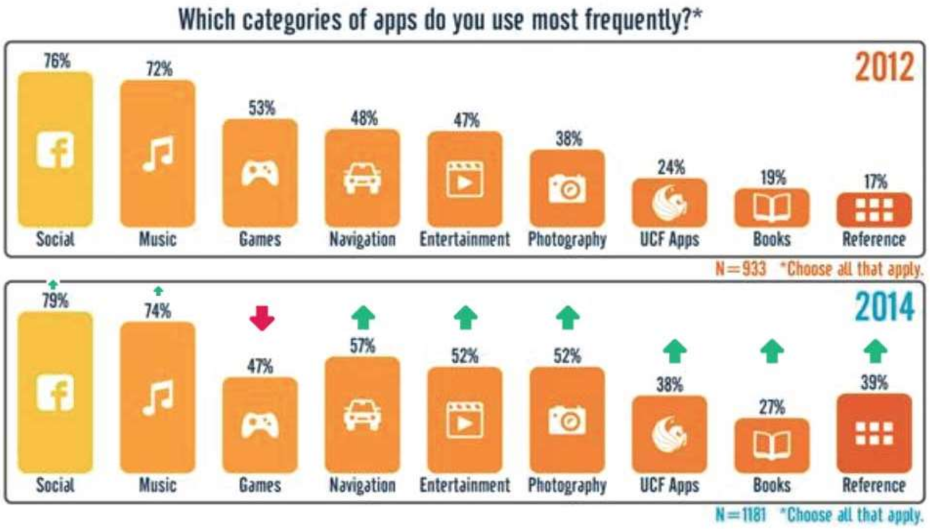


Figure 2 - Difference in App Category Usage in UCF Student (Bauer, 2015c)

Pearson's Review and Florida Central University's research display an increased usage of, and dependence on, mobile technology, particularly apps, among university students. From the information in both studies, we believe that the demand and need for location-based applications and general platform technology will rise in the general marketplace.

2.2 Campus Event Planning Technologies and Software

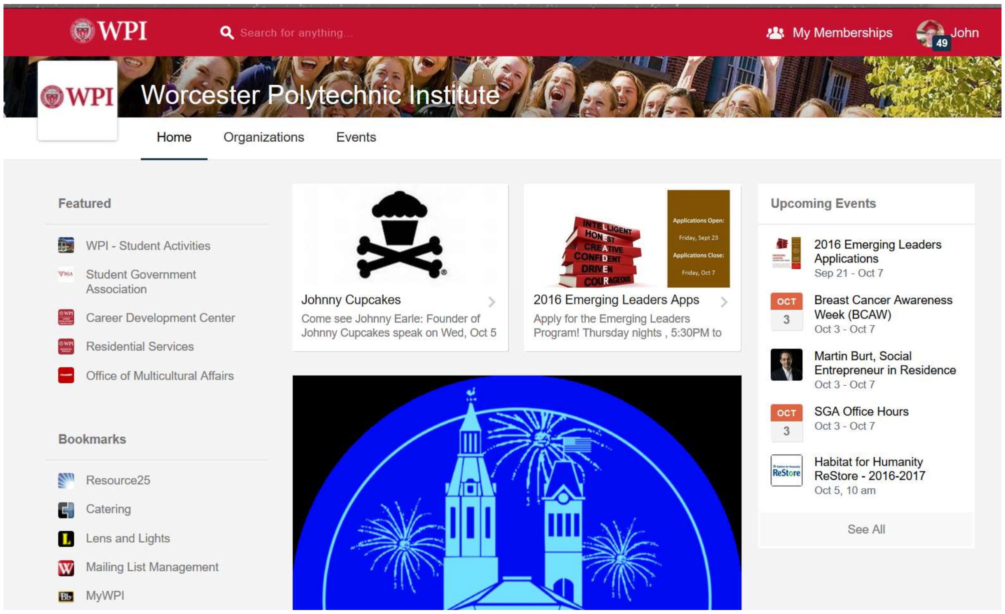
For universities aiming to increase and digitally manage student involvement within their campuses, there are a variety of pre-existing products and methods that support individualized event finding. As of 2016, there are a few common solutions among universities for providing event management capabilities. Currently, universities can use professionally developed programs from Campus Labs and Guidebook, university developed in-house software, or social media specific event planning programs such as Facebook Events as solutions for the universities event planning needs. Each of these methods provides its own benefits and restrictions to university organizations. Institutions determine the appropriate software based on the features available and of greatest value to their campus. For the following product analysis, the needs and culture of WPI were considered when analyzing potential solutions.

2.2.1 Campus Labs Solutions

Since 2001, Campus Labs has delivered products and services designed to alleviate the struggles of college administrators for organizing campus life (Campus Labs, n.d.). Currently, Campus Labs offers two flagship services, OrgSync and CollegiateLink, which focus on providing an organizational system and tools for college organizations and students.

2.2.1.1 Orgsync

OrgSync's value proposition is to solve the qualms of institution administrators and club members alike by providing a centralized online resource for campus organizations (OrgSync, n.d.). Utilizing a subscription-based system between students and organizations, OrgSync provides features to publicize important organization documents and oversee organization activities from the perspective of the institution and organization administrators. WPI as an institution is most familiar with OrgSync over other options. The Student Government Association has subscribed all campus organizations to the service for the academic period from 2013 to 2017. Below is an image of WPI's OrgSync main page.



Johnny Cupcakes
Come see Johnny Earle: Founder of Johnny Cupcakes speak on Wed, Oct 5

2016 Emerging Leaders Apps
Apply for the Emerging Leaders Program! Thursday nights , 5:30PM to

2016 Emerging Leaders Applications
Sep 21 - Oct 7

OCT 3 Breast Cancer Awareness Week (BCAW)
Oct 3 - Oct 7

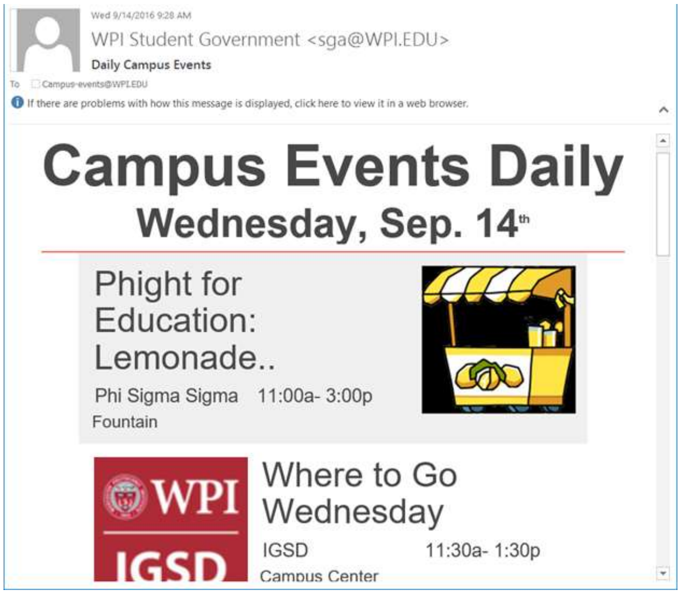
Martin Burt, Social Entrepreneur in Residence
Oct 3 - Oct 7

OCT 3 SGA Office Hours
Oct 3 - Oct 7

Habitat for Humanity ReStore - 2016-2017
Oct 5, 10 am

Figure 3 - OrgSync Web User Interface

Having been required to utilize OrgSync for the past three years, WPI campus organizations have not only become familiar with the system's positive attributes but also many of its shortcomings. One is the lack of effective event management features. OrgSync aims to provide a centralized, online resource for campuses that prioritizes document filing, anonymous polling, and campus-wide forms over event planning and advertising.



Wed 9/14/2016 9:28 AM

WPI Student Government <sga@WPI.EDU>

Daily Campus Events

To Campus-events@WPI.EDU

If there are problems with how this message is displayed, click here to view it in a web browser.

Campus Events Daily

Wednesday, Sep. 14th

Phight for Education: Lemonade..

Phi Sigma Sigma 11:00a- 3:00p

Fountain

Where to Go Wednesday

IGSD 11:30a- 1:30p

Campus Center

Figure 4 - WPI's Campus Events Daily Digest (Daily Digest, personal communication, September 14, 2016)

Consequently, OrgSync's tradeoff of providing an integrated filing system first and event finding capabilities second makes it more difficult for users to perform event related tasks such as searching for an event. In recent years, the Student Government Association along with many large campus organizations have developed work-arounds such as newsletters and other alternative methods to advertise events and activities to the student body more effectively rather than strictly using OrgSync.

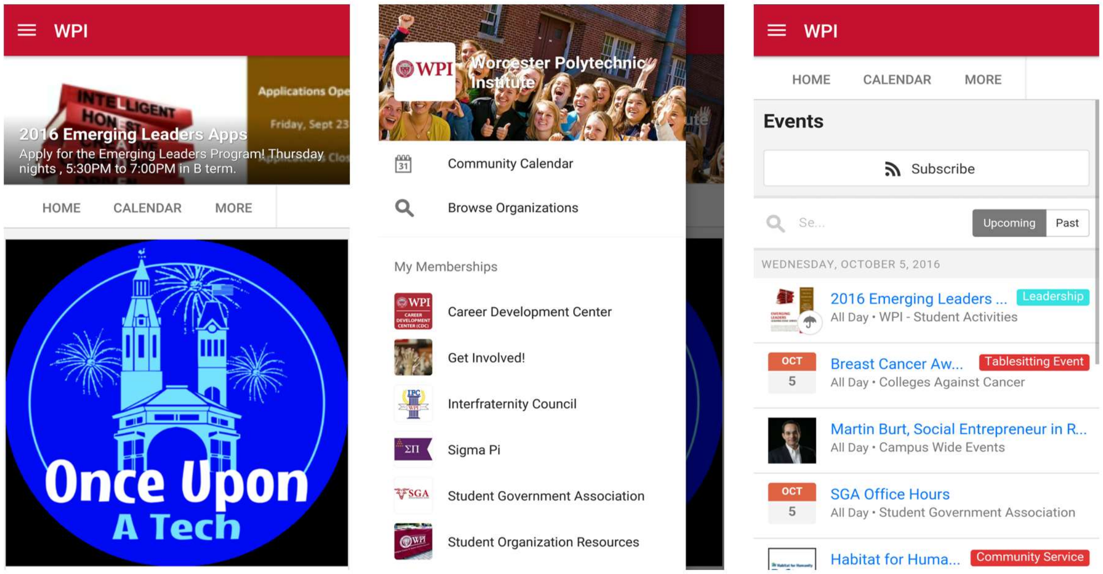
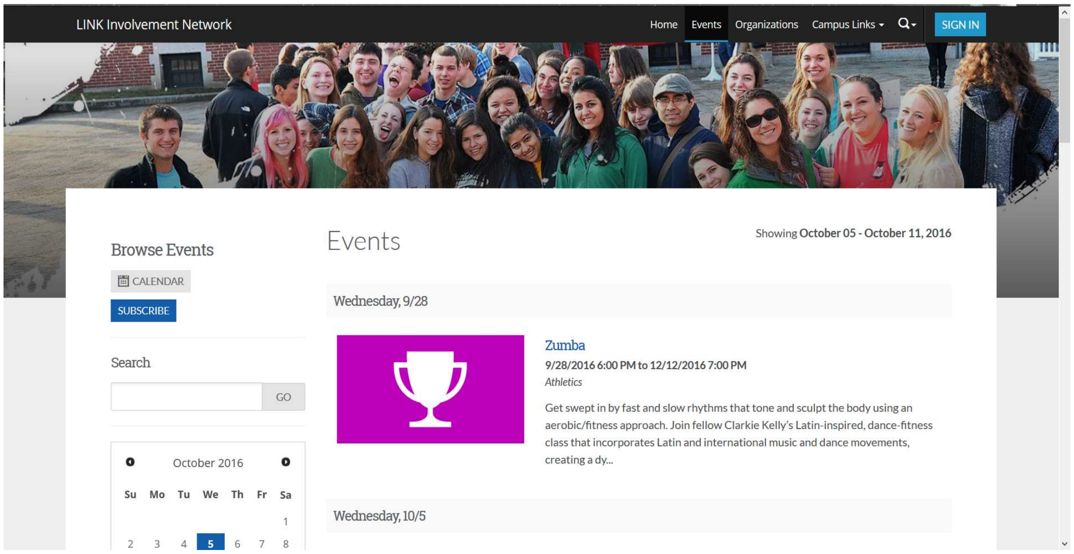

WPI
2016 Emerging Leaders Apps
Apply for the Emerging Leaders Program! Thursday nights , 5:30PM to 7:00PM in B term.
HOME
CALENDAR
MORE
Once Upon A Tech
WPI
Worcester Polytechnic Institute
Community Calendar
Browse Organizations
My Memberships
Career Development Center
Get Involved!
Interfraternity Council
Sigma Pi
Student Government Association
Student Organization Resources
WPI
HOME
CALENDAR
MORE
Events
Subscribe
Se...
Upcoming
Past
WEDNESDAY, OCTOBER 5, 2016
2016 Emerging Leaders ...
Leadership
All Day • WPI - Student Activities
OCT 5
Breast Cancer Aw...
Tablesitting Event
All Day • Colleges Against Cancer
Martin Burt, Social Entrepreneur in R...
All Day • Campus Wide Events
OCT 5
SGA Office Hours
All Day • Student Government Association
Habitat for Huma...
Community Service

Figure 5 - Examples of WPI OrgSync Mobile Application (OrgSync, n.d.)

Additionally, OrgSync focuses more on providing a web browsing tool for campuses rather than a mobile environment for users. Compared to other campus events mobile applications, OrgSync's application appears cluttered. This was a major shortcoming in OrgSync's product development as many of its competitors forecasted and adapted to the demand of college students for dedicated mobile experiences.

2.2.1.2 CollegiateLink

What OrgSync lacks in providing campuses with coherent event discovery capabilities, CollegiateLink delivers. Focusing first on event planning and advertising, CollegiateLink aims to notify its users of campuses events in the form of daily digests and location-based notifications.



Events

Showing **October 05 - October 11, 2016**

Wednesday, 9/28

Zumba

9/28/2016 6:00 PM to 12/12/2016 7:00 PM

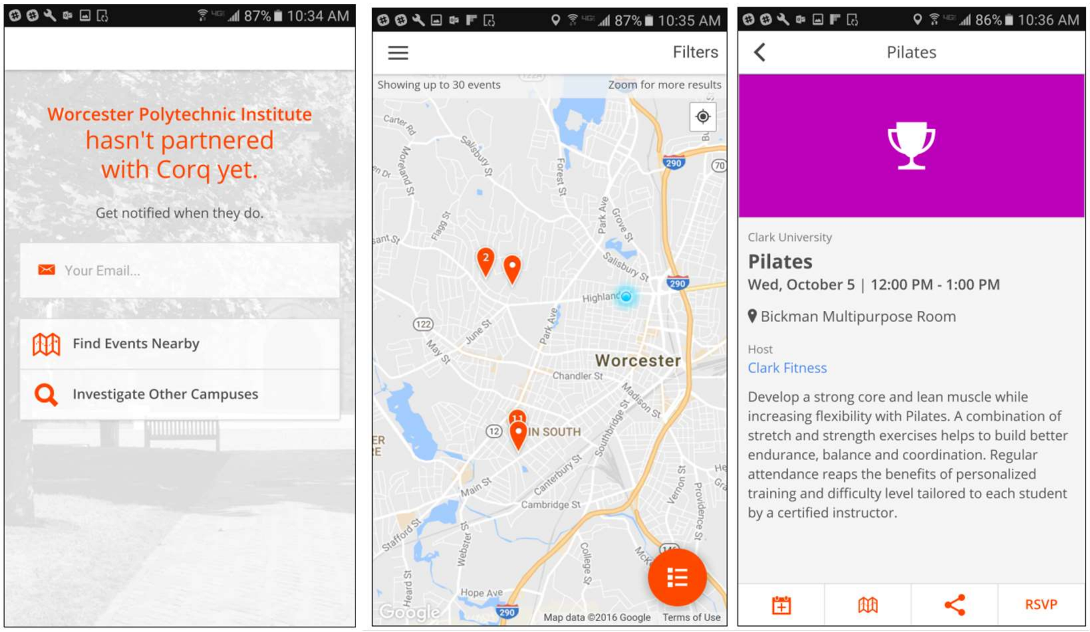
Athletics

Get swept in by fast and slow rhythms that tone and sculpt the body using an aerobic/fitness approach. Join fellow Clarkie Kelly's Latin-inspired, dance-fitness class that incorporates Latin and international music and dance movements, creating a dy...

Wednesday, 10/5

Figure 6 - Example of CollegiateLink's Web Events User Interface

They understand the importance of, and utilize the power of, consistent and short presentations of information to their users (Campus Labs, n.d.). By creating a minimalistic interface for event planners to upload their events and a mobile-friendly environment to access the service's event library, CollegiateLink provides a simple platform for universities to upload and advertise to their campuses. Concerning its user interface, CollegiateLink's mobile application, Corq, is also designed to be easily navigable and simplistic unlike options such as OrgSync.



Worcester Polytechnic Institute hasn't partnered with Corq yet.

Get notified when they do.

Clark University

Pilates

Wed, October 5 | 12:00 PM - 1:00 PM

Bickman Multipurpose Room

Host

Clark Fitness

Develop a strong core and lean muscle while increasing flexibility with Pilates. A combination of stretch and strength exercises helps to build better endurance, balance and coordination. Regular attendance reaps the benefits of personalized training and difficulty level tailored to each student by a certified instructor.

Figure 7 - Examples of Corq's Mobile Application (Campus Labs, n.d.)

Unlike OrgSync, CollegiateLink events can be viewed and accessed by anyone, as students do not have to use a secure login to access event information as logging in is not necessary to view campus events. The absence of such security precautions make uploading information to the application a risk for universities. Additionally, there are no features within the app to help club and institution administrators calculate metrics for events such as the number of reservations and attendance; which are important in planning an event's success, and later analysis.

2.2.2 Guidebook

Campus Labs is not the only major player in the college events application market. Guidebook aims to acquire market share through its designer-friendly, mobile creation templates (Guidebook, n.d.). Like Campus Labs, Guidebook's goal is to attract universities that want to create campus specific mobile applications using as little technical expertise as possible; the service provides developers with features such as drag-and-drop menus for customizing.

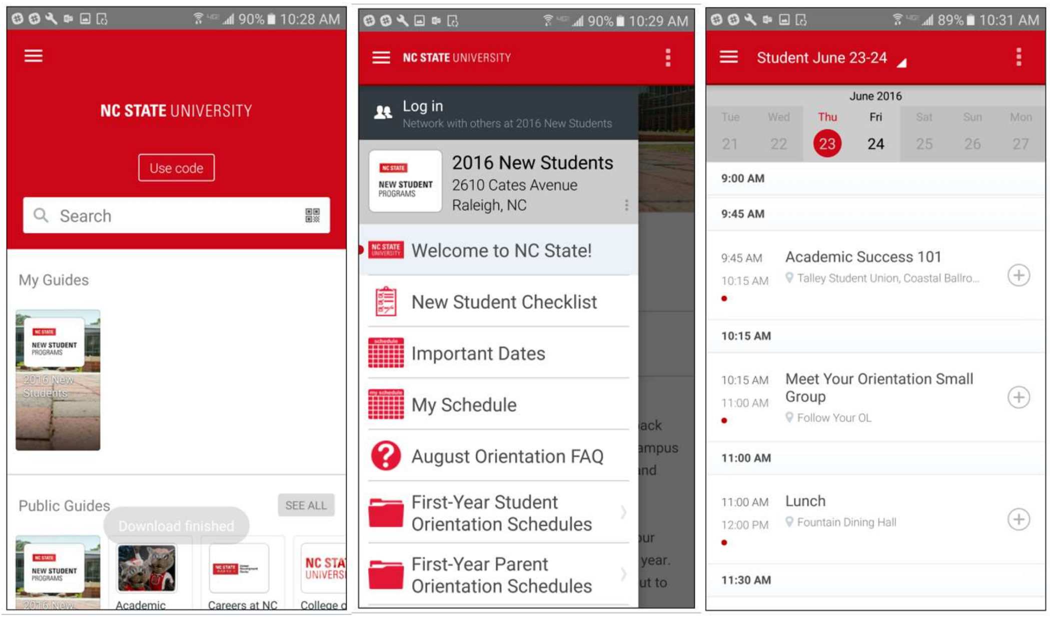
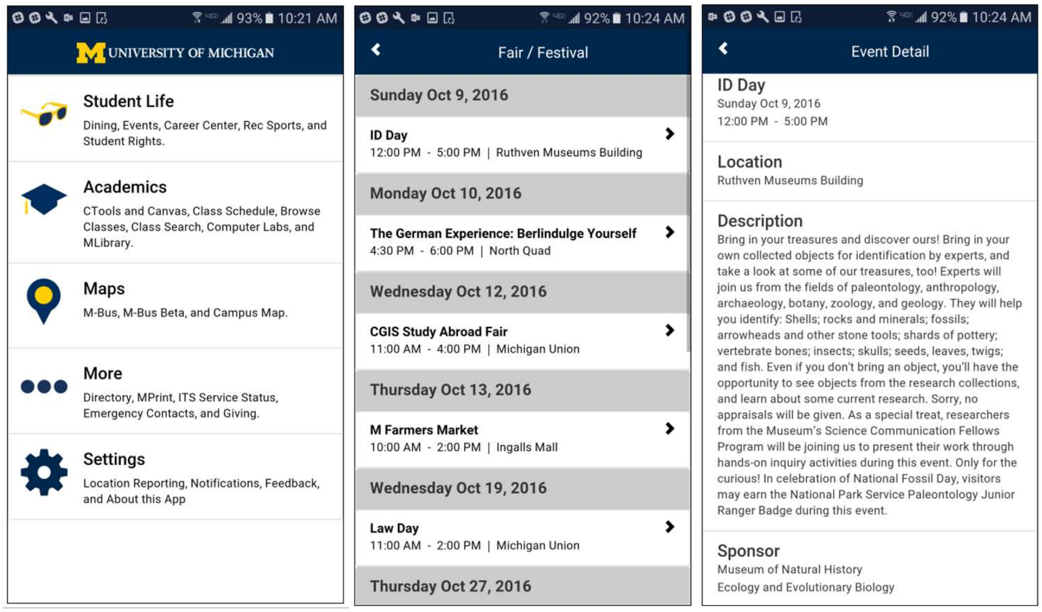


Figure 8 - Examples of Guidebook's Mobile Application (Guidebook Inc, n.d.)

Much like Campus Labs's CollegiateLink service, Guidebook's mobile templates focus primarily on the advertisement and data collection surrounding events. Guidebook is also like CollegiateLink concerning the lack of security as anyone is allowed to download and view campus activities. Beyond event planning, Guidebook applications offer additional features that allow users to manage their daily schedule, interact with others for academic or club purposes, and provide the university innovative tools to conduct virtual campus tours, orientation programming, and career fair management.

2.2.3 In-House App Development

At large state universities, in-house development is typically the preferred and most viable alternative method for providing university specific applications. Resources needed by universities to develop an application include the capital and personnel necessary to develop and maintain an application. Despite the extra costs involved with in-house development, the opportunities are numerous. With in-house development comes the possibility to make a highly unique experience for a university's student body; therefore, schools can develop software to the needs of students rather than providing cookie-cutter solutions offered by software companies.



Student Life
Dining, Events, Career Center, Rec Sports, and Student Rights.

Academics
CTools and Canvas, Class Schedule, Browse Classes, Class Search, Computer Labs, and MLibrary.

Maps
M-Bus, M-Bus Beta, and Campus Map.

More
Directory, MPrint, ITS Service Status, Emergency Contacts, and Giving.

Settings
Location Reporting, Notifications, Feedback, and About this App

Sunday Oct 9, 2016

ID Day
12:00 PM - 5:00 PM | Ruthven Museums Building

Monday Oct 10, 2016

The German Experience: Berlindulge Yourself
4:30 PM - 6:00 PM | North Quad

Wednesday Oct 12, 2016

CGIS Study Abroad Fair
11:00 AM - 4:00 PM | Michigan Union

Thursday Oct 13, 2016

M Farmers Market
10:00 AM - 2:00 PM | Ingalls Mall

Wednesday Oct 19, 2016

Law Day
11:00 AM - 2:00 PM | Michigan Union

Thursday Oct 27, 2016

ID Day
Sunday Oct 9, 2016
12:00 PM - 5:00 PM

Location
Ruthven Museums Building

Description
Bring in your treasures and discover ours! Bring in your own collected objects for identification by experts, and take a look at some of our treasures, too! Experts will join us from the fields of paleontology, anthropology, archaeology, botany, zoology, and geology. They will help you identify: Shells; rocks and minerals; fossils; arrowheads and other stone tools; shards of pottery; vertebrate bones; insects; skulls; seeds, leaves, twigs; and fish. Even if you don't bring an object, you'll have the opportunity to see objects from the research collections, and learn about some current research. Sorry, no appraisals will be given. As a special treat, researchers from the Museum's Science Communication Fellows Program will be joining us to present their work through hands-on inquiry activities during this event. Only for the curious! In celebration of National Fossil Day, visitors may earn the National Park Service Paleontology Junior Ranger Badge during this event.

Sponsor
Museum of Natural History
Ecology and Evolutionary Biology

Figure 9 - Examples of University of Michigan's Mobile Application (University of Michigan, n.d.)

The University of Michigan is one such school that utilizes in-house development. Having sufficient capital and an excellent Computer Science Department, the University of Michigan utilized their assets to create a mobile-app experience designed for their students (Mobile Apps Center, n.d.). With their resources, the University of Michigan focused on the needs of their community and developed an application suited to those needs.

2.2.4 Social Media Substitutes

To substitute or supplement campus event applications, students and university organizations typically take advantage of social media platforms to advertise and manage events. Facebook, specifically the Facebook Events feature, is a commonly used platform because of its immense popularity and how intertwined it is in the lives of millennials. More precisely, roughly 82 percent of Internet users between the ages of 18 and 29 have an active Facebook account and 70 percent of Facebook users check the site daily (Duggan, 2015).

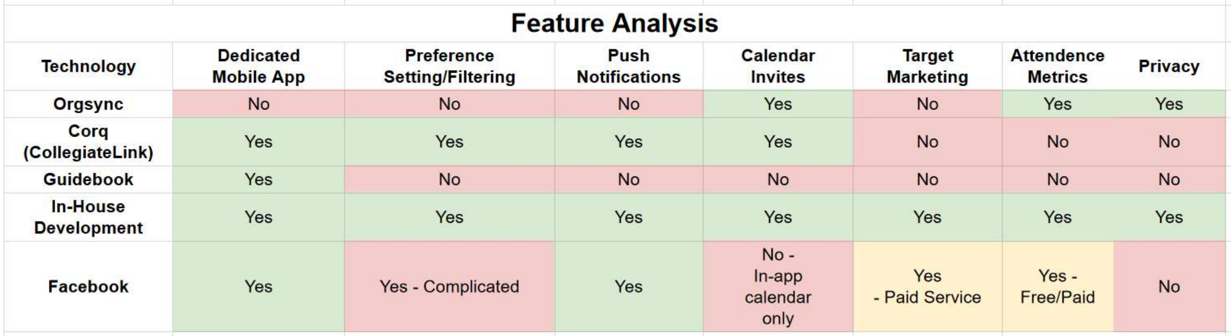
2.2.4.1 Facebook Events

Due to Facebook's immense popularity among US college students (ages 18-22), utilizing its event management features has become almost intuitive for its user base. Facebook Events provides students the capability to post, advertise, and collect metrics for free and is an effective way to inform the student community about new and interesting events.

2.3 Competitive Analysis

2.3.1 Feature Analysis

Based on the summary of the main competitors and substitutes for campus event finding technologies, Figure 2.10 was constructed to visualize some important features identified among the alternative solutions.



Feature Analysis

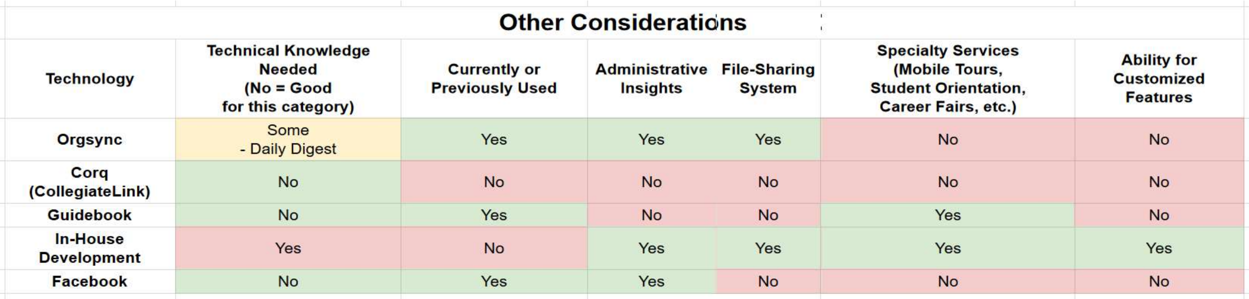
Technology	Dedicated Mobile App	Preference Setting/Filtering	Push Notifications	Calendar Invites	Target Marketing	Attendence Metrics	Privacy
Orgsync	No	No	No	Yes	No	Yes	Yes
Corq (CollegiateLink)	Yes	Yes	Yes	Yes	No	No	No
Guidebook	Yes	No	No	No	No	No	No
In-House Development	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Facebook	Yes	Yes - Complicated	Yes	No - In-app calendar only	Yes - Paid Service	Yes - Free/Paid	No

Figure 10 - Campus Event Finder Feature Analysis

In short, a dedicated mobile application developed in-house appears to be the most useful and viable approach to creating a campus event finder application that can provide all of the features required by the WPI student body. Simply, in-house development allows the freedom for WPI developers to make the best-overall application experience for their end-users. Software solutions created by outside organizations are not flexible enough and do not address the specific need of WPI students and stakeholders to have a centralized method of promoting and finding out what activities are happening on campus.

2.3.2 Other Considerations

Despite the initial advantages of in-house development, there are some disadvantages to the option. As denoted by Figure 2.11 below, the largest detractors pertaining to in-house development include the need for technological expertise in developing applications and WPI's unfamiliarity with the option. Time and resources are not to be easily allocated unless there is confidence in the success of a project, so these two detractors are large disadvantages because they instill uncertainty into the process for WPI stakeholders.



Other Considerations

Technology	Technical Knowledge Needed (No = Good for this category)	Currently or Previously Used	Administrative Insights	File-Sharing System	Specialty Services (Mobile Tours, Student Orientation, Career Fairs, etc.)	Ability for Customized Features
Orgsync	Some - Daily Digest	Yes	Yes	Yes	No	No
Corq (CollegiateLink)	No	No	No	No	No	No
Guidebook	No	Yes	No	No	Yes	No
In-House Development	Yes	No	Yes	Yes	Yes	Yes
Facebook	No	Yes	Yes	No	No	No

Figure 11 - Other Considerations for Development Options

As a team, we will overcome this uncertainty and prove to WPI that its best resources for solving the campus events advertising problem is in fact its concerned students. By understanding the target demographic and developing the application using a strict development process, our team will not only overcome the challenges associated with creating an application, but will also strive to provide the WPI campus a solution that addresses its needs in a way no large software company can.

2.4 User-Centered Design

Our objective of developing an event finding application for WPI students stems from the student body's frustration with the current methods of advertising events on campus. Students desire a mobile friendly and easy to use interface, however, the current software in use is not capable of meeting these needs. For years, WPI has tested and purchased multiple systems to try to address the complex needs concerning event planning and student involvement, but these efforts have not provided a sufficient solution for event management. Unsatisfied with the services available to us, we set out to develop an application based on feedback about the needs of students and university stakeholders. We practiced user-centered design in order to develop an effective solution for WPI, which is essentially a framework of processes in which the needs, wants, and limitations of our end users are prioritized.

User-centered design refers to a process that focuses on "usability throughout the entire development process" (Gulliksen et al., 2003). In terms of software development, user-centered design is rooted in several key principles aimed at assuring that a team is creating solutions that focus on the needs and goals of users. Such principles include: (1) Allowing the Practices of Users to Control Development; (2) Instilling Active User Participation Throughout the Project; (3) Adopting an Evolutionary Systems Development Method; (4) Understanding User Interface Needs.

The first principle instructs developers and designers to accomplish a few tasks. First, it forces them to identify their target users so they can then understand their goals, behaviors, and attitudes. Second, it encourages the team to design a solution based upon information about their users such as their preferences and what is intuitive for them. It also requires developers and designers to abide by a developmental process that ensures the team maintains communication with users throughout all the stages of development (Gulliksen and Goransson, 2001).

Abiding by the second principle emphasizes that a team will involve end-users in the development process to acquire feedback. By incorporating collaborative development with users early and continuously, there will be fewer opportunities for the team to produce solutions that are not aligned with users' goals (Nielsen, 1993).

The third principle emphasizes that developers should construct an iterative and incremental development process as it is, "impossible to know exactly what to build from the outset." (Gulliksen et al., 2003). Each iteration should consist of user needs analysis, design, and documentation of evaluation and recommendation phases. This helps teams stay organized with their priorities and how their efforts benefit their end-users.

The final principle emphasizes that teams should rigorously test and acquire feedback on ways to make a user's experience with the system's interface as intuitive and simple as possible. Too often, teams skim over user interface testing to save time and resources. This malpractice often contributes to designs that are unnatural to the end-user and as a result hinders the effectiveness of the application as a whole (Cooper, 1999).

2.5 Native vs. Hybrid Application

When developing software, one of the most important decisions is determining what type of technology will be used to create the product. It is imperative to weigh the advantages and disadvantages of each choice before the beginning of any technical project to create an effective product using the most efficient development process possible.

In the case of creating a mobile application, the most popular choices currently for developers are native development and hybrid development. A native application "is an application program that has been developed for use on a particular platform or device" (Rouse 2016). This allows for the direct use of device specific hardware and software, such as vibration and push notifications. Examples of native environments for mobile applications are Android and iOS. Hybrid applications, though, are built using multiple web technologies like HTML, CSS, and Javascript. What makes these apps "hybrid" is the fact that they are "hosted inside a native application that utilizes a mobile platform's WebView" (Bristowe, 2016). This housing for hybrid applications allows for the use of device specific hardware as well. To further analyze these two strategies, we have broken them down into different parts for comparison: development community resources, user interface design, and user experience.

Resources for native development are straightforward. With native development, a team would need to design separate applications for every target environment. For example, if an application was aimed at both iOS and Android environments, then both would need to be written separately. This would come at both an advantage and disadvantage to an author as that author would have access to all resources (tools and code developed by other authors) for each codebase. This could potentially be larger than any one codebase, but the issue arises that most tools and code would only be useful for one of multiple codebases. This may make translation between codebases difficult and overall more work for a team. The advantage to hybrid development is that there is one central pool of resources to utilize and explore, in this case those for web development which happens to be massive in the community.

Design for native development is similarly complex to its resources. With any environment, there are different variables and design choices to consider before beginning development. For example, the way in which code communicates to the database may be different between environments. With hybrid development, we take advantage of the smaller pool of choices and focus in on one way to approach each design decision. By making design simpler, the team can maximize the number of work hours and focus on building the application rather than getting lost in design.

User experience may vary machine to machine based on a user's hardware and software. The advantage of native development is that each environment can optimize the experience for users on the devices for which they are designed. For example, if the team were to write the application aimed at iOS users using the standard language Swift for iOS, Swift would be able to optimize the housed functions to run well on Apple hardware since iOS applications only run on Apple iPhones and other Apple devices. Limiting the audience can allow further optimization on each environment. Hybrid applications are not capable of doing this since they are designed to allow anyone to use them regardless of the device. Since the audience is more general, less optimization is available for an individual user.

Speed of development and cost are simple to analyze. With native development, a new application needs to be built for each environment. With each application built there are marginally fewer users who may utilize the application in comparison to the work required to create each app. Since hybrid applications are designed to be created once and transferred to as many users as possible, only one application needs to be built and this application can be transferred to as many types of devices as necessary. Hybrid applications have a huge advantage in this area. If cost is analyzed by the number of work hours on the development team, then cost is closely associated with speed of development. Since hybrid applications take less time to development, they must also take fewer work hours to complete and would have a cost advantage associated with them.

When creating an application, questions typically arise concerning how to correctly implement sections of code. When this occurs, it is essential to have resources available, solid documentation, and community of developers. For native developers, this is the case. Android and iOS already have extensive documentation on all aspects of the native development process. There are also many message boards, websites, and videos available to help walk developers through almost all of their potential issues. Since hybrid development can be achieved by using multiple different technological combinations, the available resources tend to be scattered across multiple locations and need to be pieced together. This can oftentimes be a daunting and time consuming process for developers.

Another important part of the development process is the design and creation of an appropriate user interface and experience. To meet the demands of their users, developers need to work with technologies that give them complete control of the design. While both native and hybrid development can account for most interfaces, they both have key differences when it comes to designing user interfaces. Native development provides developers the ability to use many standardized "design widgets for creating certain standard interfaces," an option that is not currently supported by hybrid application development tools (Comentum 2016). These available design widgets can cut down development time for each specific mobile platform. Hybrid development, however, is much more flexible when it comes to designing web pages with dynamic content and sizing. Ultimately, native development is less time consuming but also less flexible in this aspect, whereas hybrid development is extremely flexible but takes more time. Both options present solid benefits, but it is the job of the developer to decide which benefits are most important for the project on hand.

Another main difference between hybrid and native development is the user experience of the application. It is critical to understand what kinds of interactions will be present in the mobile application when determining whether to go with native or hybrid development. Native development creates a fluid, responsive experience in which users cannot detect load times (Comentum, 2016). Hybrid applications can run into problems when processing clicks and scrolls, as they can sometimes produce a slow, laggy experience for users. While these issues can be corrected through clever coding techniques, this can be time consuming and prone to errors. Additionally, hybrid apps can be susceptible to poor performance when displaying large animations (Comentum, 2016). Developers must have a general idea of what interactions will be present in the app before they decide which technology would best support the application.

In certain situations, the speed of development can be crucial. Hybrid development shines in this area. One of the main strengths of hybrid is its ability to allow developers to write one code base and port it to multiple platforms. If a team wants to create native applications they need to understand the nuances of multiple programming languages as they relate to a platform, compared to only needing to know HTML5, JavaScript, and CSS for Hybrid. Quickly creating multiple applications for different platforms can not only reach a larger customer base faster, but will also cut down on costs as well.

When working on a development project that has money allocated towards it, it is critical to analyze the costs associated with the project. Hybrid development allows the team to spend less money on the development process because it is typically shorter than the Native development alternative. Unlike native applications, where the amount of time spent is relative to the number of platforms being targeted, hybrid allows the development team to focus their time, effort, and money into one code base instead of multiple.

The differences between native and hybrid development are apparent, and create the need for development teams to seriously analyze and understand the nuances of their own project before they can decide on which path to take. Choosing between hybrid or native technologies should be a decision tailored to the project at hand. The more suitable choice creates the most desirable result for both the team and end-users.

2.6 Summary

Our goal for this project is to develop a campus event finder application for WPI undergraduates. To do this, we must justify the need for a campus event finder application and analyze the potential routes that can be taken to complete such a project: our goal. Now that we have analyzed the needs of WPI and outlined the potential paths for development, we will further explain the options we chose that would set our team up for success in terms of delivering an application that campus wants and needs. We aim to be an efficient and effective team, and with our detailed and rigorous development process we are confident that we will meet our objectives and overall goal.

3. Methodology

We met with our advisors during the early stages of our project to formulate our project's goals and objectives. The meeting prompted improvements to our original idea alongside solidifying our project goals, objectives, and long term expectations. This initial meeting allowed us to establish our project goal: *create an application that would allow WPI students to have access to information about exactly what activities are happening around campus and can suggest activities that interest them.* To accomplish this, we developed the following objectives:

1. Identify the basic features needed by both the student body and campus organizations.
2. Develop a basic application that allows students to view what activities are occurring around campus.
3. Improve and test the application using Agile methodologies.
4. Evaluate the effectiveness of the application in terms of student satisfaction with the application, perceived usefulness, and perceived ease of use.

The project's 3 terms (each of 7 weeks) resulted in determining to use each term to accomplish a specific yet different objective.

The first term's work was to complete our requirements needs analysis, create a basic user interface (UI) mockup, and develop a basic working application. The basic working application is the minimum viable product, or MVP. Our team completed these tasks through interviews, research, becoming familiar with material design, and multiple programming sessions using Agile development methodologies.

The second term focused on controlled application testing with end-users to determine application changes and feature requirements. We tested our application with end-users by administering focus groups and interviews. Additional necessary features and other specific data was acquired through each test. After each test was conducted, we used the information we received to add tasks that improved the application to our backlog for the term. By the end of the term we had created a fully functioning application that our team felt comfortable and confident sharing with all WPI students.

During our final term, we intended to release our application to the public. The team's efforts focused on continuous product testing, application promotion, and long term expansion viability to other Universities. We surveyed students from WPI and at other universities in the Worcester area to test determine our best course of actions for our application's success. Additionally, we continued to improve our application via our agile sprints, but our primary objective was to assess the feasibility of our application expanding and becoming an actual business.

3.1 Methodology Selection

Software development methodologies are defined as the splitting of software development work into distinct phases containing activities with the intent of better planning and management (Radigan, 2016b). While numerous development methodologies exist, our group compared the Software Development Life Cycle (SDLC) and Agile Development as two options that were most applicable to our project.

The SDLC consists of a detailed plan that describes how to develop, maintain, replace, and alter or enhance specific software. The life cycle focuses quality improvement of the overall software development process by breaking it into six phases. Each phase has a different purpose and are as follows: Planning, Defining, Designing, Building, Testing, and Deployment (Radigan, 2016c). All phases are linear and dependent upon one another, so each phase come after each other. The SDLC has different models, or ways of operating the phases, but SDLCs' goal remains the same regardless of model type. (Radack 2009).

Typically, the SDLC is used when a development team has all the time and resources needed, usability of the product is limited so that only a select or controlled group can utilize the software, and when the requirements are fixed and not too complex. This can be problematic because it requires a very controlled environment unlike most situations. Should complexity or variability increase SDLC becomes very inefficient.

Agile Development fundamentally incorporates continuous feedback and iteration to enable development teams to successively refine and deliver a software system. Like the SDLC, Agile Development is an umbrella term that contains multiple methodologies that a team can choose. Although somewhat different, each Agile methodology encourages continuous planning, feedback, integration, and refinement in the software and product (Beedle, 2016). The key for Agile Development's success comes from 'sprints', which allows teams to routinely review, check, and understand each piece of the project to ensure the greatest likelihood of success (Radigan 2016). A sprint "is a short (ideally two to four week) period in which the development team implements and delivers a discrete product increment" (Atlassian, 2016). Unlike the SDLC though, Agile requires constant communication among team members to enforce collaboration and understanding for the most effective production environment possible.

When deciding between the SDLC and Agile, our team took into account our specific project restrictions and requirements. Our project is on a specific timetable of three terms, each consisting of seven weeks. Considering that we had minimal time to develop and then test the application, our team agreed that we needed a development methodology that had high adaptability and constant user feedback. Agile Development presented our greatest chance for a successful project completion and product launch leading to its adoption as our development methodology.

3.2 Agile Methods

As previously mentioned, Agile Development is a broad term that contains many different methodologies. The methodologies include Scrum, Lean Development, Feature-Driven Development, Crystal, and Dynamic Systems Development (Radigan, 2016b). After assessing each methodology, we decided that using the Scrum methodology would be the most beneficial.

It should be noted that another option our team considered was extreme programming, or "XP" agile software development. Using techniques like pair programming, the XP methodology leverages teamwork to deliver value immediately to customers (Wells, 2013). Although our team favors using pair programming, we found that scrum still allows for pair programming through its emphasis on co-location and communication.

Scrum is a simple standard set of roles, responsibilities, and meetings that never change. By creating a standard, it removes unnecessary unpredictability, so that teams are better able to handle the unpredictability that is part of every project. Essentially, the Scrum methodology allows teams to continuously improve their product while always having it in a condition that is ready for deployment and use.

3.3 Agile Scrum Development

The Agile Scrum framework was created to be a modern replacement to SDLC and other document-driven development methods. Scum quickly became a business strategy team solution to deliver cost effective results (Saddington, 2013). Through Scrum a team can effectively complete their goals with maximized efficiency.

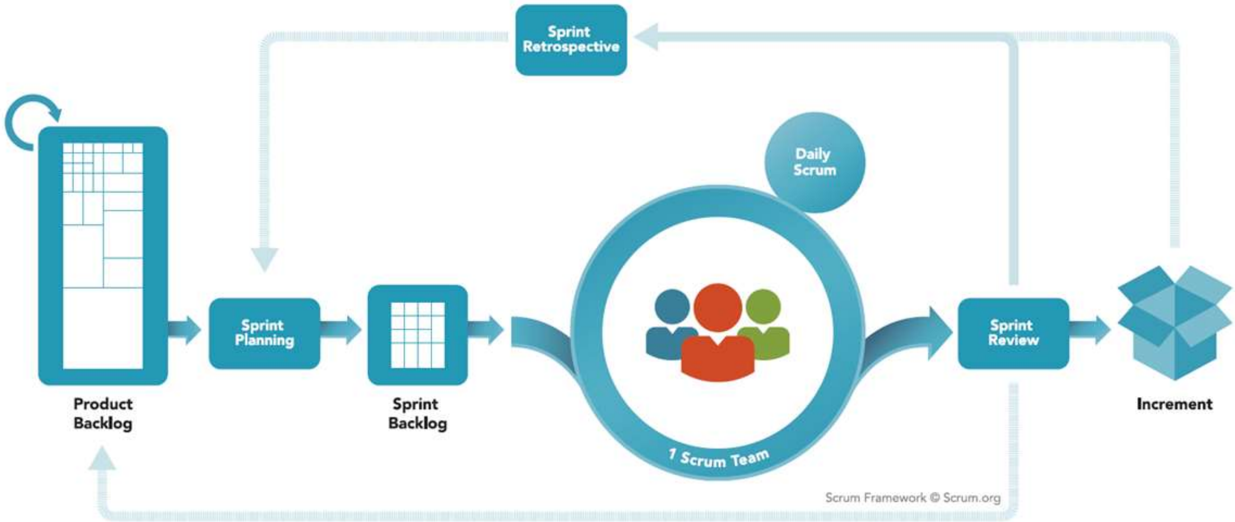


Figure 12 - Scrum Development Process (What is Scrum?, n.d.)

Creating a software product is a complex process. As previously mentioned, Agile Development breaks down the work into smaller yet different parts to make it more manageable. These parts are called epics. Epics are composed of many user stories that describe what the user wants to accomplish when using the product (*Atlassian*, 2016b). All User stories must be broken down into tasks to begin the software development. These tasks indicate what steps are needed for each user stories completion (*Software and I*, 2011). When any team member receives a task they now know all requirements to accomplish the task and how the task supports the project's completion. When tasks are created, they are encapsulated in what are called sprints, which teams use to forecast what set of user stories will be completed during a fixed time period usually ranging from one to four weeks.

Task work estimation can be used to create sprint burndown charts (*Atlassian,* 2016a). The charts give all project members an idea of all work currently completed and remaining. Sprint usage makes the Agile Scrum framework unique in process development and product delivery. Team members pull tasks from the backlog that they would like to complete instead of having tasks pushed to them by their manager. They then help to create the estimation time for the tasks on which they will be working. This cooperative method allows a team to hold greater productivity, morale, and constant accountability.

The structure of the Agile team is important. There are multiple key members involved in delivering a product: the stakeholders, the product owner, the scrum master, and the scrum team. Stakeholders are the people to whom the product owner reports the progress of the team (Saddington, 2013). The product owner is responsible for making sure the team delivers the most value to the business. They do this through multiple activities, such as managing the current team tasks, keeping both the business team and development team on the same page about the current sprint, and directing the team what to do next (Radigan, 2016b). Another important member of the agile scrum framework is the scrum master. The scrum master informs the agile team members about the scrum process, resolves issues arising within the team, and most importantly defends against sprint scope change (Saddington, 2013). Finally, the last main role of the scrum framework is the scrum team, which is the engine of development. Through communication and teamwork, the scrum team members forecast how long it will take to complete their work and attempt to carry out these tasks during the allotted sprint timeframe.

To encourage communication and keep all members of the team updated on the progress of the project, agile scrum teams use meetings or "ceremonies" (Radigan, 2016a). There are different types of ceremonies, each with a distinct role. Before each sprint, the team will conduct a sprint planning meeting. During the meeting, the product owner goes through all the current user stories that need to be completed with the team, and together they decide what can be done during the upcoming sprint. These user stories are then broken down into tasks and placed into the sprint backlog. Another ceremony that is important to scrum success are daily stand up meetings. These meetings are typically held in the morning with all team members present to discuss the progress, next steps, and conflicts each team member faces. They are intended to be a compact and effective way for the team to keep each other in the loop while allowing the entire team to shed light on a problem a member might be having.

The last ceremony used in scrum framework are sprint retrospective meetings. The meeting's goal is to look at the previous sprints successes and failures. From these observations, team members determine what actions should be taken in the next sprint to improve performance. Continual improvement is paramount to an Agile team's success (Radigan, 2016a).

Our project will follow an iterative process that is tailored to our three-term situation. At the beginning of every sprint, we will conduct a sprint planning meeting to decide what user stories we will develop. This meeting will include breaking down user stories into specific tasks for the development team. Sprint completion will be done via daily stand up meetings, agile tools, and co-location. Every week, we will report our progress through sprint burndowns to our advisors, who will be our stakeholder. During the end of each sprint, our team will carry out a sprint retrospective meeting. All notable information will be accounted to ensure a successful next iteration sprint planning meeting. Finally, we will plan and a conduct development release process at the beginning of C term.

3.4 Development Environment

3.4.1 Trello

To stay on task, the team used specific project management tools. Without these tools, it would be easy for the team to lose track of our progress and lose sight of what we need to accomplish each week. This is where software tools like Trello and JIRA come into play.

Trello is a web-based project management tool developed by Fog Creek Software in 2011 and became its own company in 2014 (Trello, 2016). The software allows users to create simple lists and cards to help keep track of tasks and assignments by team member. Doing this during our weekly meetings allows us to remain accountable to the tasks we commit to each week and helps the project manager maintain a pulse on how much we have accomplished and where we need help moving forward. What makes Trello great is its versatility. Not only are lists and cards used for software development, but they can be used for the tasks targeted at the business team as well as the entire team in general. Everyone can keep track of everyone else's progress, but the central themes are kept separate within different boards or groupings of lists. This way, the business and development teams can each keep track of their progress without the added clutter of other assignments. Tasks are in one place and checking on anyone's progress is only three clicks away from the home screen.

The other project management tool that is widely used in industry is JIRA by Atlassian. JIRA is specifically built to manage Agile processes. It contains many features that Trello does not have including ready to use workflows, agile planning and reporting, real-time release tracking, a wide variety of add-ons designed specifically for Agile Development and data center deployment. This extensive list of features makes JIRA superior to Trello in most ways, but these advantages come at a cost. Pricing scales with the number of users on the board. If JIRA would be used exclusively for the development team, the pricing for one to ten members starts at

$10 per month (Atlassian, 2016). As the number of users increases the pricing jumps from $10 a month to $75 with 11-15 users, $150 with 16-25, and $300 with 26-50. These features may be extremely beneficial to large scale projects in industry, but for our team the pricing was not worth the potential benefits. Additionally, to avoid a jump in the charges we will most likely keep as few people on JIRA as possible, our project management would become decentralized and more difficult to control. If the project were to ever scale to a point where the team would take full advantage of these benefits it may be worth investing in JIRA in the future.

3.4.2 IntelliJ IDEA & SublimeText2

Our next decision was to determine which coding environment we should utilize. There are two main forms of coding environments from which we could choose, an Integrated Development Environment (IDE) or a Text Editor environment. For Integrated Development Environments, RubyMine by JetBrains is the dominant tool on the market. Having an IDE such as RubyMine would make development a lot faster. IDE's come with dozens of key features that allow programmers to avoid tedious tasks and get productive faster. Such key features include "GoTo Source" which allows the user to quickly examine the source code behind functions and gems called within the code. This is helpful when trying to understand code that was written by teammates or other programmers within the Ruby community or trying to understand how your code links together. The drawback to Integrated Development Environments, in particular RubyMine, is the expense of purchasing the software. Software for ruby development is $70 for each developer on the team (JetBrains, 2016).

The second option, Text Editor environments, had more options. The two main competitors for graphical text editors were Atom and Sublime Text. Atom has many great community packages which help enhance the environment, such as atom-beautify which cleans up the code as your write, or autoclose-html which generates the close tags automatically for HTML code. Most of these packages, however, were also offered in some way by default with Sublime Text. Additionally, Sublime Text operates well on all operating systems and has excellent community support (Gaebel, 2015). Finally, most of the development team had already worked with Sublime Text in some shape or manner and it was not worth the time investment on the team's behalf to switch their environment to Atom. Here, familiarity won.

The choice between Integrated Development Environment and Text Editor environment was clear cut. It was not worth the team's time and monetary investment to switch over to an IDE when most of the functionality supplied by the IDE was available through Sublime Text with community packages. If the code we write were to become more complex, it might be worth the investment to switch over to RubyMine, but in the meantime, we are satisfied with the tools we have currently at our disposal.

3.4.3 Heroku

To transfer the code to an environment where we can share it publicly, we needed a way to host our code so that our initial users could use it. This is essential in the early stages of development so that we can get our iterations in front of real users in our focus sessions. Heroku is a cloud Platform-as-a-Service where users can upload their code, test, and deploy all in the cloud (Heroku, 2016). Heroku also implements a feature where developers can link their Github repositories to the service so publishing new edits to the code is as simple as committing to a branch and re-deploying the code on Heroku.

Additionally, the cost of hosting our code on Heroku is only seven dollars per month which is less expensive than other more professional options such as Amazon Web Services. While Amazon Web Services may have much more power and configurability, it was beyond what we needed to complete our project (Smashingboxes, 2012). While investigating the possibility of using Amazon Web Services, the team determined that it was too complicated to utilize at the beginning of the project, but we may move our platform to Amazon Web Services once the project is up and running and we need additional support/features.

3.4.4 PhoneGap

Traditionally, developing for cross platform environments has been a time-consuming process. Not only does a team need to learn an entirely new programming language for each platform, but they need to reimplement their entire application and ensure its quality matches up with all other platforms. Finally, teams must also provide test suites for each of the platforms along with documentation. Supplying your application for cross-platform would double, sometimes triple, the workload on the team while only supplying marginally more consumers.

Adobe PhoneGap attempts to resolve this issue by supplying a method for developers to create cross-platform applications for iOS, Android, and Windows phone, using HTML, CSS, and Javascript (Adobe, 2016). This allows the team to focus on technologies they know (web development) and create mobile applications that can reach the entire audience of the mobile community. By allowing developers to use web technologies, the team can take advantage of advancements in these areas, such as AngularJS developed by Google or Ruby on Rails for backend support.

PhoneGap makes developing the application easier and saves time and resources which we can devote back into new features and reaching new users. With PhoneGap we can hit the entire market simultaneously.

3.5 Architectural Spike

Our team decided to operate using agile methodologies for this project, but we recognized that before we could fully become agile we first needed to go through an “architectural spike” during our first term of work. This architectural spike was a period where our methodologies for development and requirements gathering were not completely agile, and were a mix between the SDLC and agile due to the initial time constraint in our first term. Although our team used agile methodologies during the entirety of our project, our first term consisted of requirements gathering, design, and the coordination of both of those. This period was planned to first do what a systems development life cycle (SDLC) project would typically focus on doing first, so even though most of our project was strictly agile, we considered this period an “architectural spike.”

3.6 Requirements Gathering & Analysis

To gather the functional and nonfunctional requirements of our proposed application, we conducted surveys, focus groups, and interviews that were targeted towards specific populations of the WPI community. This process spanned throughout the entirety of our project.

3.6.1 Requirements Gathering

Throughout the course of developing the application, tactics such as surveys, focus groups, and interviews were used to understand the behaviors, issues, and needs of our stakeholders.

3.6.1.1 Surveys

Prior to development, it was important for us to first identify exactly what the application needed to do and what the potential end-users wanted and needed it to do. We needed to ensure that our potential users were interested in using our application in the future. To gauge this, the team decided to send out an initial survey in our first term of work, A-term, to assess the validity of our project. In this survey, we also gathered information on what our potential users would like in an application, in order to help create the initial set of requirements.

A survey allows the team to acquire qualitative and quantitative data on WPI students’ usage of event planning and notification tools and determine the viability of an Event Planner Application. The survey extended beyond viability to learn demographics, preferences, and other information to know how to best produce a marketable application to WPI students.

With the survey, we aimed to obtain responses from at least 100 undergraduate students from a diverse age and campus involvement background. We aimed for demographic splits that were near evenly distributed between WPI freshman, sophomore, juniors, and senior classmen. Secondarily, we made sure to capture the demographics of survey participants of those who are members of organizations, are not members, and who live in varying living situations. Our group predicted that the survey would hold the greatest value in terms of data analysis for the products creation, if we could obtain all this information.

3.6.1.2 Survey Execution Plan Timeline

The timeline forms the basis of the survey's viability. We orchestrated the plan over the course of a week and a half and the process is described below.

Before we could run the survey, we needed to first test if our survey would collect the data we needed and were looking for. We first conducted a mock-survey which was a first draft of our actual survey. We ran this mock-survey on 20 WPI students before editing and finalizing the new survey. Once the survey was finalized it was disseminated using the following tactics: personal references, spontaneous conversation, and mass emailing.

Personal referencing refers to the process of team members targeting individuals that would not only participate in the survey, but would refer the survey to their friend groups. Individuals such as club presidents and friends involved in a variety of clubs were targeted as their network outreach potential made it easier to get the survey to a broad group of stakeholders.

Spontaneous conversation refers to approaching students to take the survey in a variety of places on campus, such as the library, campus center, and dining areas. This method was chosen so that we could ensure a good split of responses between the 4 undergraduate classes.

Mass emailing consisted of emailing the survey to club or group aliases with the intent to distribute as many students as possible. This method was geared to allow us to collect responses by members of types of demographics.

We chose to start the survey process by simultaneously using all the above-mentioned methods. Given that we had only a week and a half to collect responses we decided to reach our goal of at least 100 responses that we needed to collect responses as quickly and effectively as possible.

3.6.2 Focus Groups

Focus groups are used to develop understanding of how and why stakeholders hold certain beliefs or opinions about points of interest. Traditionally, focus groups are interactive discussions between fewer than ten stakeholders that is facilitated by a moderator. Moderators are responsible for creating an inclusive environment that elicits stakeholder's perspectives to discover trends, mindsets, and insights on the topic. An observer will accompany the moderator to record the discussion and analyze the behaviors and nonverbal indicators of the participants (Section, 2016).

It is important to note that focus groups have several limitations. Moderators should remain aware of the validity of information presented and derived from focus groups as participants could misconstrue facts or inaccurately represent the beliefs of the stakeholder category they represent.

3.6.2.1 Focus Groups Audience

For the Campus Event Finder Application, one focus group with club presidents was facilitated to gain insight about the issues and needs of the current WPI event planning process and determine the desires and must have features of the application to help them pursue their turnout, productivity, and longevity of membership goals. In this focus group, we had a variety of clubs represented so that we could fully understand the desires of different clubs for the application. Additionally, this audience was foreseen to have tremendous influence over the success of the implementation of the application at WPI, so we sought to gain their feedback in particular about improvements and features necessary for them to adopt it.

3.6.2.2 Focus Group Sessions

Our focus group session was made up of 2 members of our team along with five (5) club presidents. In the session, we discussed a prepared set of questions that covered concerns, needs, and issues our audience found with anything related to finding events on a college campus. The question prompts used for the focus group can be found in Appendix A.

We made sure to hold this session in a conference room without distractions and where voices can easily be heard across the given location. The moderator, a member of our team, checked a session room prior to the focus group occurring to ensure windows, open doors, and other noises were not paramount that could reduce productivity of the session. We also prepared a backup location in case issues occurred, alongside recording and backup recording equipment, or note taker to support and avoid all possible issues.

During the session, the moderator ensured that there was no dominant group member influencing people's actions or dissuading a conversation from its overarching goal or objective. All attending members were encouraged to speak freely, safely, and openly on any topic to promote any shy and quiet group members to participate in the given focus group.

3.6.3 Interviews

Interviews provide data by recording an individual's perspective and experience on issues relating to our product’s current development stage. Two primary techniques that were used during all interviews were open-ended questions and a conversational dialogue tone. This qualitative approach was geared to allow our team to obtain in-depth and varied data that could not be acquired using simple "yes/no" or "multiple choice" questions. Furthermore, this approach allows the interviewer to be able to approach a non-traditional line of questions to further understand and analyze the interviewee and this/her thoughts.

Lastly, a qualitative interview process was chosen because it would allow us to gather emotional data. The ability to understand how a person feels about our product’s design and interface was crucial for the development and adoption of the Event Finder application into campus life at WPI.

3.6.3.1 Target Audience

The target audience of our interviews consisted of 18-23-year-old average WPI undergraduate students. We define ‘average’ undergraduate students as those who do not hold an executive position in any campus clubs or organizations. We made sure that in addition to this that we interviewed an even spread of students among the four (4) undergraduate classes. Each class has varying preferences and desires, so it was critical that we captured them to understand how to best ensure the application's usefulness and success.

3.6.3.2 Interview Sessions

Our interviews were done between a single member of our team and a single interviewee at a time. We held eight (8) interviews so it was important that we conducted them in a calm and private setting so that each interviewee felt confident in sharing exactly how they felt based on our questions. Each interviewer made sure to document any additional clarifying questions and answers as we decided this information would be useful although it was not originally planned for. In these sessions, we discussed a specific prepared set of questions that covered WPI student's current perception of event advertising at WPI and determined how a new technology could benefit students finding and knowing about campus events. The question prompts used for each interview can be found in Appendix B.

We made sure to hold these interviews in private locations where both the interviewer and interviewee could clearly hear one another, and where the interviewee felt comfortable sharing their thoughts and opinions openly. During the session, the interviewer avoided influencing the interviewee’s responses or opinions regardless of their content. Our team felt it was extremely valuable not to reject any ideas or assert any of our biases onto the interviewees.

3.7 Design

When creating our original design, or mockups, we researched and implemented Material Design within our application. Material Design is a design language developed in 2014 by Google which expands upon and uses grid-based layouts, responsive animations and transitions, and depth effects such as lighting and shadows. Although only two features drove our original designs, we wanted to create our MVP knowing how we wanted to design the user interface to help long term feature and functionality alterations. Furthermore, an early design style allowed for consistency and avoided developing an inconsistent or poorly received user interface.

3.7.1 Diagramming

3.7.1.1 Entity-Relationship Diagramming

An Entity Relationship Diagram (ERD) is a visual representation of the relationships between data entities stored in an application’s backend (Technopedia, n.d.). In our agile methodology, we create and iterate on our ERD as we design and develop during each of our sprints. Therefore, our ERD changes on a near weekly basis.

3.7.1.2 Data Flow Diagramming

The team developed data flow diagrams (DFD) to keep track of the flow of information within our application. These diagrams allow the reader to understand how data is handled at a high level within an application. Another data flow diagram perk is the ability to show how data flows externally as well as internally, making the diagrams useful for both internal and external entities interacting with the application. Additionally, these diagrams are beneficial to a team with multiple disciplines because it helps teammates convey concepts and interactions within the application in ways that would be difficult to explain using words (Lucidchart, 2016).

3.7.2 Programming Language Selection

3.7.2.1 AngularJS

For the application's front end, the team developed using Google Material Design via AngularJS. The Google Material Design library provides aesthetically pleasing UI features that are intuitive to use and follow standard industry guidelines. Building the application using AngularJS allows for a quick way to begin developing the application and get an iteration built for testing and focus groups to gain additional feedback for future iterations. The faster the team can finish the first iteration, the quicker we can begin to develop additional features that will be crucial during the launch of the application.

3.7.2.2 Ruby on Rails

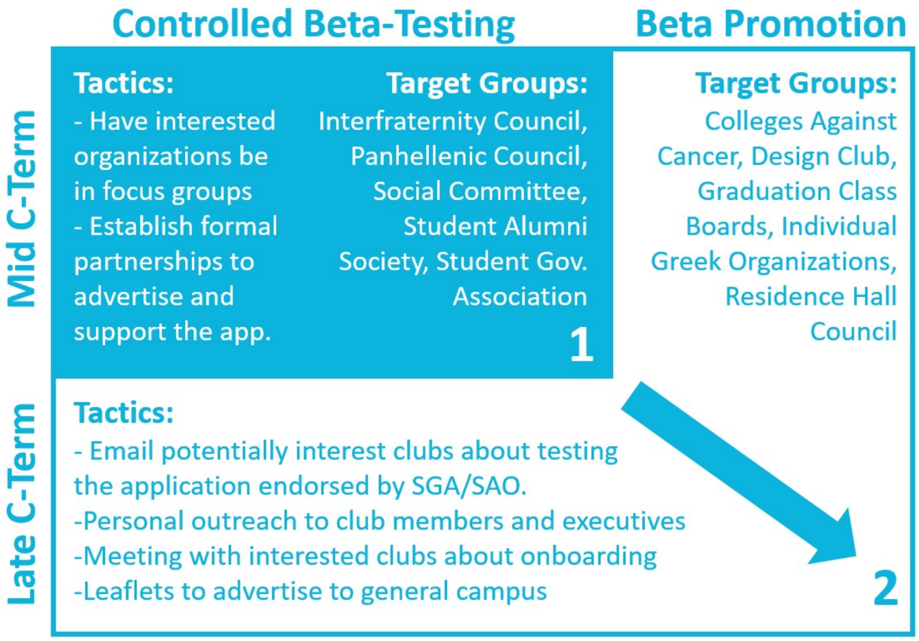
For the application's backend, the team developed using Ruby on Rails. The main benefits for developing with Ruby is the immense community supporting development behind the scenes. Many features such as authentication, REST APIs, and database migration are simple in Ruby. There are hundreds of gems (imports from other developers) that allow for an application to be developed quickly and begin gaining traction. Like AngularJS, the quicker the team can get a working iteration of the backend code, the faster development will continue in the future.

3.7.2.3 PostgreSQL

For the application database, the team used PostgreSQL. The main benefits are shared with Ruby in that there is a large community that supports development using PostgreSQL. Therefore, problems and minor bugs that come up during development require little effort to quickly resolve. Additionally, the integration of PostgreSQL and Ruby on Rails is seamless and requires only minor configuration outside of database migrations, which are taken care of with command line calls in Rails.

3.8 Implementation

For our campus event finder application, our team decided to utilize a “rolling launch” approach over the traditional “hard” or “full launch” methodology. According to Oxford Dictionaries, a rolling launch is when a service or product is gradually introduced to the general market (Oxford). We decided the rolling launch approach best aligned with our pre-existing Agile Development methodology. This would have allowed us to test our market assumptions concerning our application and control the application’s growth and scope by parsing the launch into three distinct phases: controlled-testing, club promotion, and full implementation.



	Controlled Beta-Testing		Beta Promotion
Mid C-Term	**Tactics:** - Have interested organizations be in focus groups - Establish formal partnerships to advertise and support the app.	**Target Groups:** Interfraternity Council, Panhellenic Council, Social Committee, Student Alumni Society, Student Gov. Association 1	**Target Groups:** Colleges Against Cancer, Design Club, Graduation Class Boards, Individual Greek Organizations, Residence Hall Council
Late C-Term	**Tactics:** - Email potentially interest clubs about testing the application endorsed by SGA/SAO. -Personal outreach to club members and executives -Meeting with interested clubs about onboarding -Leaflets to advertise to general campus		2

Figure 13 - Launch Plan Summary

Unfortunately, we were not able to execute this original plan. Although we had done work to ensure we could execute this plan, we were unable to launch the fully release the application at WPI. To be able to release an application to WPI, it needs to abide by certain security protocols such as integrating with the university's Central Authentication Service (CAS). Being able to connect to the CAS was all we needed for our application to be compliant to WPI’s standards, but due to time constraints and low priority on WPI’s IT team we were unable to accomplish this.

In place of this rolling launch our team used focus groups to verify and test whether our application would be successful at WPI. This testing is discussed further in Chapter 7.

3.8.1 Controlled-Testing & Introduction

During controlled-testing, our team presented and made the application available to only a few of the largest and most prominent clubs on campus to first assist us in performance testing the application, to increase popularity of the application, and to fill the application with content. Participants consisted of clubs and organizations who had large amounts of active members and whose members were involved with many other campus organizations. The clubs and organizations on campus that we worked with for this controlled-testing phase were: SGA, IFC, Panhel, and SocComm. This initial phase spanned a week during our final term, C-term. This timespan was more than sufficient because it allowed us to conduct the testing that we needed to complete along with introducing the application to campus through the most involved and influential students.

3.8.2 Promotion & Full Implementation

After controlled-testing and introducing the application to WPI, our team then began to heavily promote the launch of the application with the help of some WPI clubs and departments. This phase consisted of working in collaboration with prominent clubs and departments on campus to promote the use of the application to the larger student body. To allow for this, we made key partnerships with the clubs and organizations we worked with in the previous phase with the addition of WPI's Student Activities Office (SAO) as they are the overseeing body of all campus organizations and our best channel in reaching WPI organizations. This was incredibly important because now that the application had content from many clubs and people using it, which provided the best opportunity to attract and have students become users. Through our partnerships, our team could utilize resources such as club aliases, the budgeting process, and club presidents' meetings to effectively advertise and onboard clubs and students onto our application. With these tools, the tactics that our team used to promote the application were email blasts, personal outreach, meetings with different clubs/organizations, and paper media. This phase began immediately after our controlled-testing and introduction phase which helped us gain momentum for this phase of full implementation near the end of our final term of the project.

3.9 Testing Methodology

Near the end of Sprint 9 and until after full implementation of the application, our team conducted testing of the application in a variety of ways that would be valuable to the application's longevity and success. We documented these tests in a formal 'test plan.'

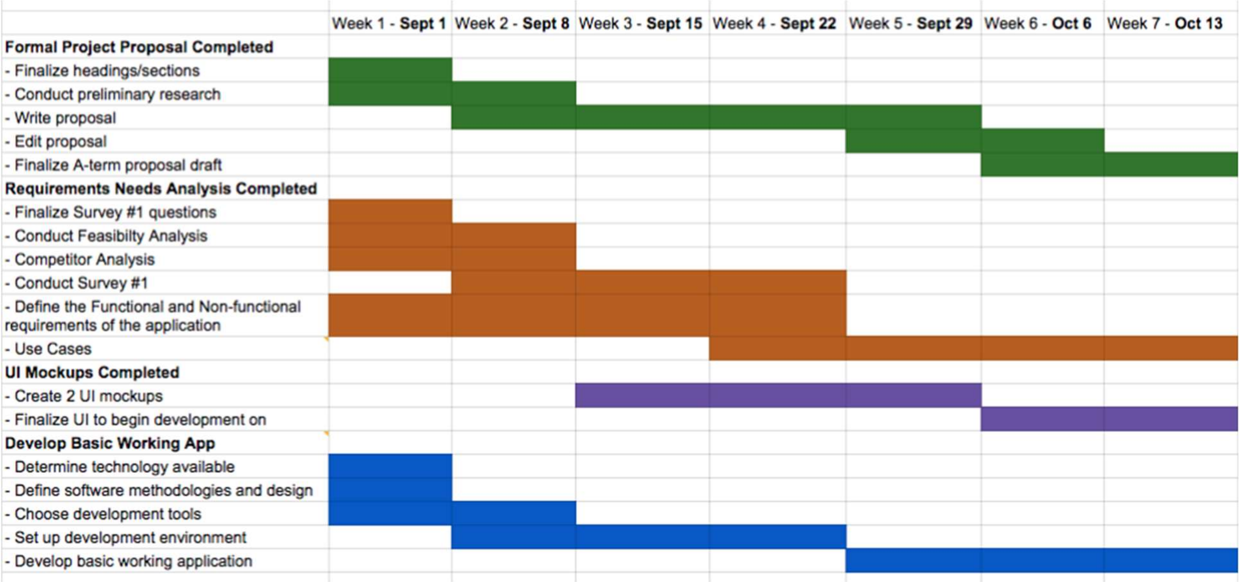
3.9.1 Test Plan

Our test plan for the application consisted of 4 distinct testing phases to assure the application's reliability; those phases include: (1) Unit Testing, (2) Integration Testing, (3) System Testing, and (4) Acceptance Testing. Specific information about how we conducted these tests and their results are presented in Chapter 7.

3.10 Timeline

Utilizing the term system to our advantage, we separated our project into three distinct phases; initial requirements gathering and development, iteration and minimal application release, and implementation and long-term feasibility analysis.

3.10.1 A-Term Objectives



	Week 1 - **Sept 1**	Week 2 - **Sept 8**	Week 3 - **Sept 15**	Week 4 - **Sept 22**	Week 5 - **Sept 29**	Week 6 - **Oct 6**	Week 7 - **Oct 13**
Formal Project Proposal Completed							
- Finalize headings/sections	🟩						
- Conduct preliminary research	🟩	🟩					
- Write proposal		🟩	🟩	🟩	🟩		
- Edit proposal					🟩	🟩	
- Finalize A-term proposal draft						🟩	🟩
Requirements Needs Analysis Completed							
- Finalize Survey #1 questions	🟧						
- Conduct Feasibilty Analysis	🟧	🟧					
- Competitor Analysis	🟧	🟧					
- Conduct Survey #1		🟧	🟧	🟧			
- Define the Functional and Non-functional requirements of the application	🟧	🟧	🟧	🟧			
- Use Cases				🟧	🟧	🟧	🟧
UI Mockups Completed							
- Create 2 UI mockups			🟪	🟪	🟪		
- Finalize UI to begin development on						🟪	🟪
Develop Basic Working App							
- Determine technology available	🟦						
- Define software methodologies and design	🟦						
- Choose development tools	🟦	🟦					
- Set up development environment		🟦	🟦	🟦			
- Develop basic working application					🟦	🟦	🟦

Figure 14 - A-Term Task Timeline

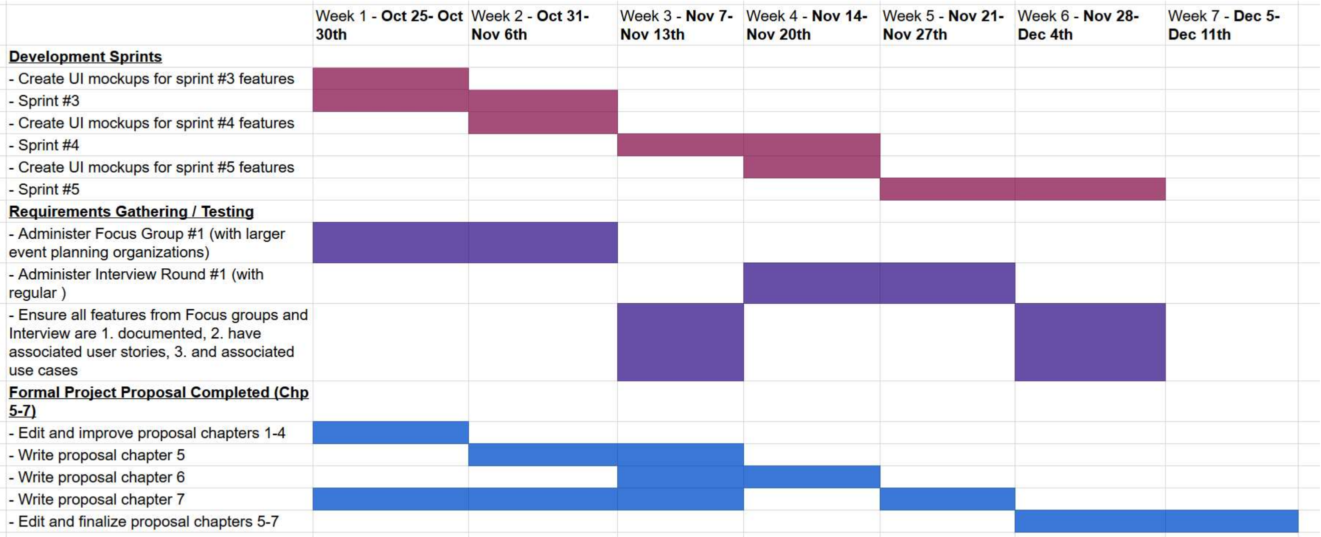
During A-Term our team goals were to validate our project concept, gather initial minimal application requirements, and develop a minimal application. Our team set ambitious task deadlines to accomplish our goals.

Validating our project's concept and initial requirements gathering occurred concurrently during the first half of A-Term. By conducting preliminary research and a competitor analysis of on-campus event finding applications, we validated the existence and value of event finder applications on college campuses. Afterwards, we surveyed WPI students to identify their specific needs for a campus event finding application.

To develop a minimum application, the team understood the need to determine a developer environment suitable for creating web and mobile applications. After doing so, we began the 'architectural spike' segment of our development which included the requirements gathering as mentioned before. From our preliminary analysis, we determined several user stories and use cases that made up the anatomy of our application. From there, we began performing bi-weekly sprints to develop the two features needed to create our minimum application. These two features were 1. Being able to create events, and 2. Being able to view all of the created events. By the end of A-term we had created an application with these two key features completely functional.

With the minimum application completed at the end of A-term, we were prepared to begin gathering feedback from intended end-users, and iterating the application based on that feedback in B-Term.

3.10.2 B-Term Objectives



	Week 1 - **Oct 25- Oct 30th**	Week 2 - **Oct 31- Nov 6th**	Week 3 - **Nov 7- Nov 13th**	Week 4 - **Nov 14- Nov 20th**	Week 5 - **Nov 21- Nov 27th**	Week 6 - **Nov 28- Dec 4th**	Week 7 - **Dec 5- Dec 11th**
Development Sprints							
- Create UI mockups for sprint #3 features	🟥						
- Sprint #3	🟥	🟥					
- Create UI mockups for sprint #4 features		🟥					
- Sprint #4			🟥	🟥			
- Create UI mockups for sprint #5 features				🟥			
- Sprint #5					🟥	🟥	
Requirements Gathering / Testing							
- Administer Focus Group #1 (with larger event planning organizations)	🟪	🟪					
- Administer Interview Round #1 (with regular)				🟪	🟪		
- Ensure all features from Focus groups and Interview are 1. documented, 2. have associated user stories, 3. and associated use cases			🟪			🟪	
Formal Project Proposal Completed (Chp 5-7)							
- Edit and improve proposal chapters 1-4	🟦						
- Write proposal chapter 5		🟦	🟦				
- Write proposal chapter 6			🟦	🟦			
- Write proposal chapter 7	🟦	🟦	🟦		🟦		
- Edit and finalize proposal chapters 5-7						🟦	🟦

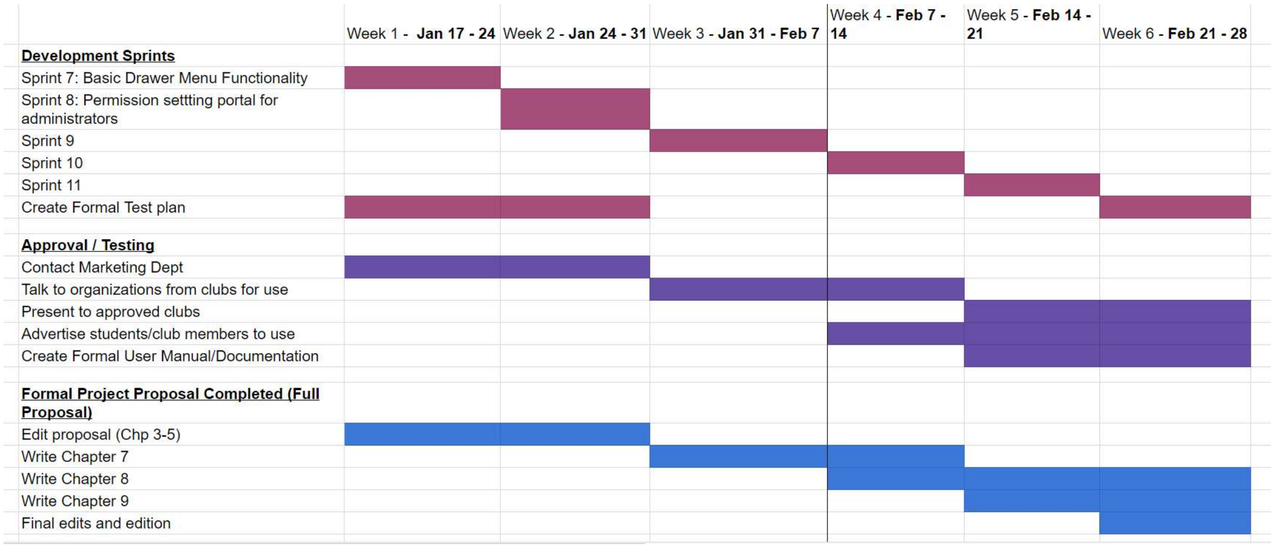
Figure 15 - B-Term Task Timeline

For B-Term, our team goal was to complete our applications' minimal viable product development which included adding a signup feature, logging in, subscribing to clubs, club representatives creating events, and presenting users with information about events that they would likely be interested in. We continued to use an agile methodology to gather feedback through focus groups and interviews with end-users, and developed new user-stories and features for the application during this time. We successfully developed new features and functionality for the application every two weeks, and to do this we had to be diligent and effective at quickly recording and analyzing user feedback, and communicating internally to identify, define, and document these new features.

By the end of B-term, our team had created an application that met the basic needs of students searching for events and organizations seeking to gain more attendance at their events. Next, we planned to aesthetically improve the application during our winter break and address minor bugs so that we would be able to have a rolling launch of the product near the start of C-term as we gradually publicized the app from a controlled base to the greater student body.

Throughout the launch, we planned to gather information from users on whether the application adequately met their needs and how we could improve it further.

3.10.3 C-Term Objectives



	Week 1 - **Jan 17 - 24**	Week 2 - **Jan 24 - 31**	Week 3 - **Jan 31 - Feb 7**	Week 4 - **Feb 7 - 14**	Week 5 - **Feb 14 - 21**	Week 6 - **Feb 21 - 28**
Development Sprints						
Sprint 7: Basic Drawer Menu Functionality	🟥					
Sprint 8: Permission settting portal for administrators		🟥				
Sprint 9			🟥			
Sprint 10				🟥		
Sprint 11					🟥	
Create Formal Test plan	🟥	🟥				🟥
Approval / Testing						
Contact Marketing Dept	🟪	🟪				
Talk to organizations from clubs for use			🟪	🟪		
Present to approved clubs					🟪	🟪
Advertise students/club members to use				🟪	🟪	🟪
Create Formal User Manual/Documentation					🟪	🟪
Formal Project Proposal Completed (Full Proposal)						
Edit proposal (Chp 3-5)	🟦	🟦				
Write Chapter 7			🟦	🟦		
Write Chapter 8				🟦	🟦	🟦
Write Chapter 9					🟦	🟦
Final edits and edition						🟦

Figure 16 - C-Term Task Timeline

In C-term, we continued developing features for the application that would help legitimize the possibility of its use at WPI, fully tested the application, and began marketing the application to influential organizations through our beta-testing efforts.

The features developed during C-Term that helped legitimize our application included finalizing the club admin permission functionalities and auto-refreshing the application's content. The club admin permission was important because it addressed user concerns about others posting invalid event and club information. Auto-refreshing helped make the application more dynamic and gave user instant feedback in regards to event information.

Testing was another crucial step toward a potential launch of the application. Without proper and thorough testing, no self-respecting company or information technology department will adopt a technology, and WPI is no different. Therefore, developing a testing plan and utilizing test cases and focus group testing were necessary to accomplish this term.

Finally, we continued to market the application to influential student clubs and WPI departments to make launching the application at WPI more possible. Student organizations such as the Student Government Association and Interfraternity Council were asked to beta-test the application as they have been heavily involved in assisting us with creating the application thus far and they can influential to marketing the application to the wider student body and other organizations. WPI's Marketing and IT Departments were also followed up with this term to generate professional support of the application at the university. The support of both are crucial to have for an application endeavor since either have the power to cease the production or use of the application. We managed to get Marketing's full support for the project, however, we were not able to completely collaborate with IT to integrate their profile protection services.

3.11 Summary

Our team thoroughly analyzed our projects situation to accomplish our goal of creating an application that grants WPI students access to information regarding what activities are taking place around campus and suggest activities the individual will likely be interested in attending. After gathering the necessary requirements for our project, we concluded that based upon our unique, shortened timeframe and the amount of feedback we wanted from users, our team needed to iterate quickly. This meant adhering to the Agile Development process, which gave us a flexible option that would allow our team to make changes on the fly while always giving the development team a definitive set of tasks to complete. Tools such as Trello and Heroku enabled the development team to manage and quickly launch the application to users, which is essential given our timeframe.

Our project spanned over 3 terms making the objective and focus splitting necessary to ensure efficient and optimized productivity across each term. In A-Term, we had conducted consumer research analysis to determine product viability and developed a minimum application using feedback from potential end-users. In B-Term we had continued the consumer research analysis through focus groups and interviews alongside constant development while also preparing for the application's launch. Lastly, in C-Term we continued development as we analyzed the application's effectiveness and long term sustainability.

4. Requirements Gathering and Analysis

After determining the methodology ideal to developing this application, the team commenced a rigorous and continual requirements gathering/development cycle. By utilizing the requirements gathering techniques of surveying, focus grouping, and interviewing, the team discovered development requirements early and often.

4.1 Survey Results

A preliminary survey of over 130 WPI undergraduates was conducted to gather early initial insights of our user population. The survey was designed to understand undergraduate's:

1. Rate and difficulty of attending campus events;
2. Rate of utilizing the campus daily events advertisements;
3. Preferences of campus events marketing channels;
4. Receptiveness to utilizing a new, WPI-centric application;
5. Preferences of application functionalities.

4.1.1 Rate and Difficulty of Attending Campus Events by Year

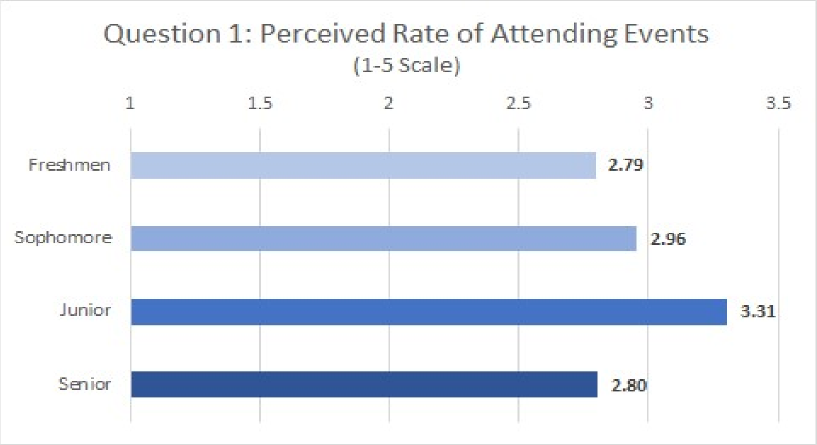


Figure 17 - Perceived Rate of Attending Events by WPI Students

This chart, Figure 17, when analyzed with respect to class year at WPI, gives the following information. First, there is a growth rate in event attendance as a student progresses at WPI. From freshman year to junior year, a student becomes more involved in campus life and therefore will have an event attendance growth. The growth rate, however, takes an immediate drop off senior year that is nearly equivalent to a freshman student's event attendance level. We believe that the average senior is now developing preparations for leaving college with focus elsewhere, not campus events. The similarity of freshman and senior club attendance rates were what we thought to be because of freshman having limited knowledge of events and seniors simply having too much to accomplish and focus on to attend many campus events.

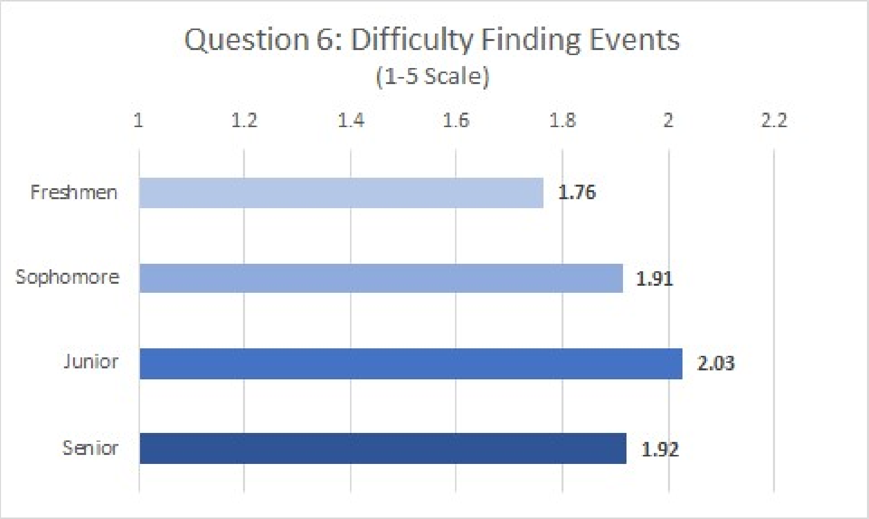


Figure 18 - Perceived Difficulty of Finding Events by WPI Students

The Difficulty Finding Events Graph displays a student's rating of how difficult it is to find events on campus. The figure follows a similar growth rate to Question 1, but for different causes. A freshman student is continuously prompted to attend pre-scheduled events and has yet to join clubs and organizations until later in the school year. The lack of knowledge and student connections makes finding and attending events difficult, but the least difficult in comparison to all three other grades. Sophomores and Juniors excluding the clubs they currently attend have a greater difficulty finding events due to a lack of easy discovery of events beyond physical and online media outlets. The only outlier, like Question 1 (Figure 18), is the senior class. The senior class has an easier way of finding events on campus presumably due to them holding the most overall knowledge of events and student life than the previous three class years. We believe that their knowledge and familiarity with WPI allows them to have an ease in event discovery similar to the freshman class.

4.1.2 WPI students rate of Campus Event Mail Reading

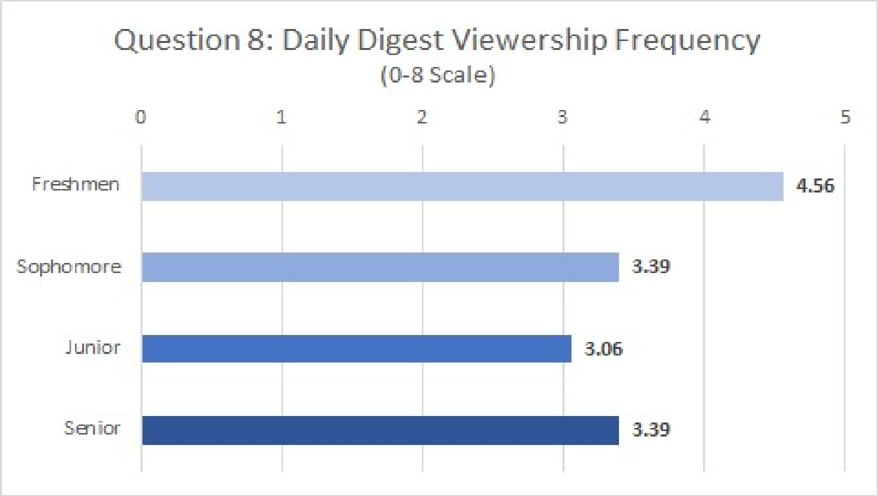


Figure 19 - Daily Digest Viewership Frequency

When looking at the Daily Digest Viewership Frequency graph we can see that the freshman class holds a much greater average of viewership, we believe this to be for two primary reasons. The first reason stands that they were brought onto campus alongside the emails first introduction making it something that they are accustomed to and must rely on and adopt, unlike upperclassmen. The older classes hold a lower yet nearly all equal viewership frequency rate. We believe this to be due to their own current involvement in many activities and the Daily Digest email being a new introduction to the WPI campus. Traditionally, newly adopted technology in any environment will take time for routine usage by older members of an organization unlike new members who have never used any other process.

4.1.3 WPI students Preferred Advertising Methods

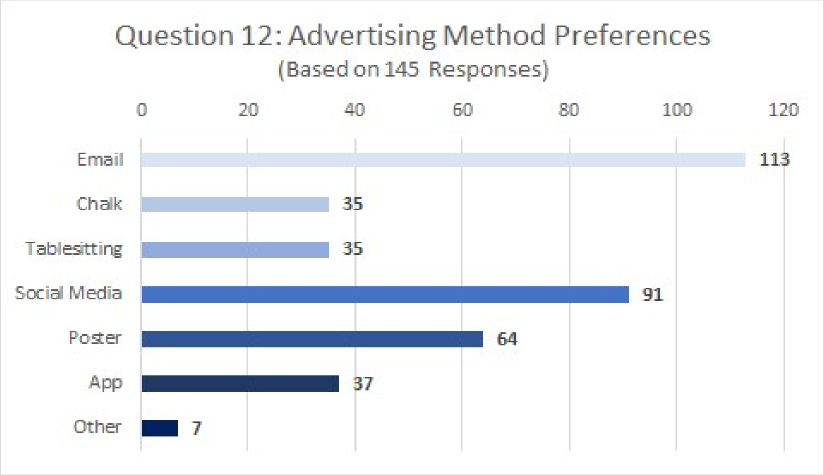


Figure 20 - Count of Advertising Method Preferences

The WPI campus primarily requires email usage by all students and faculty to maintain communication. When the email usage requirement is coupled with the newly introduced daily event email it is clear why students prefer email advertising than anything else as shown in Figure 20. It must also be accounted for that other forms of advertising are limited and used by niche groups at WPI including chalk, table sitting, and posters. However, the second most preferred advertising method, Social Media, is a direct result of the technological age where college students using mobile devices can most easily find information through a social network if an application does not already exist for what they require. The most important information, a preferred use of an application for advertising shows promise for our prototype and eventual product. A favorable number of people would use an application and, coupled with no product existing at WPI currently, it is possible many more would prefer an App once one is created successfully.

4.1.4 WPI students Event Finder Application Adoption

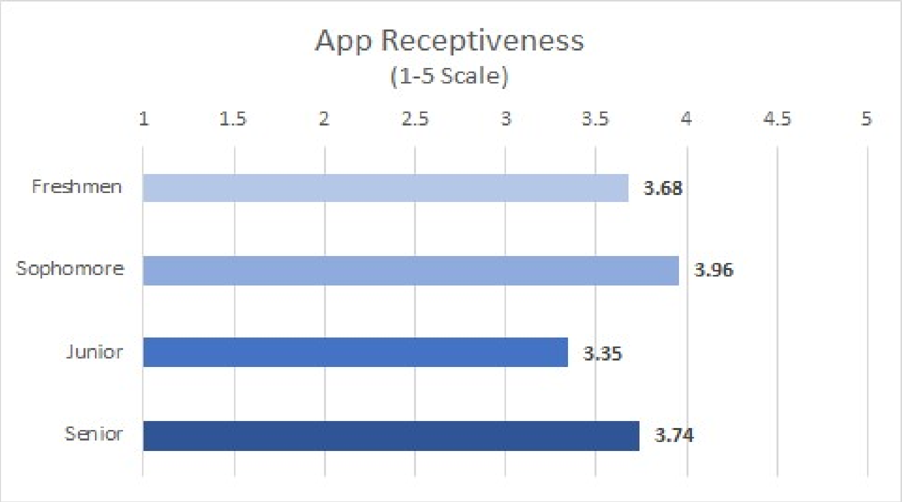
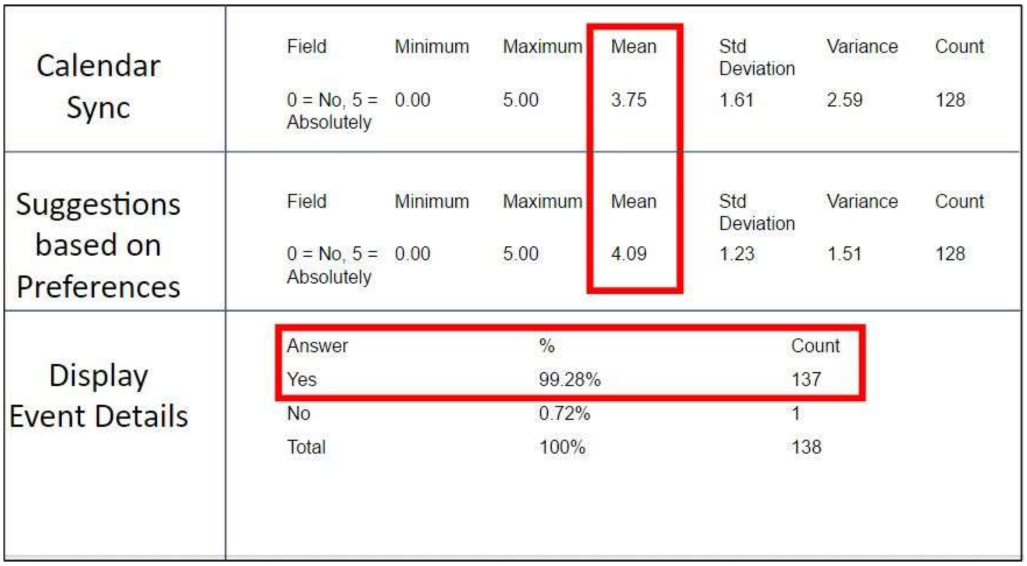


Figure 21 - App Receptiveness

All WPI classes hold an above average favor to using an application to find events on campus. Comparing the positive results to the consumer event finder preferences we see that people have a desire for a centralized application for event notification and discovery. This information further substantiates the possibility that our application does solve a current need at WPI and potentially other colleges.

4.1.5 Desired Features Adoption



	Field	Minimum	Maximum	Mean	Std Deviation	Variance	Count
Calendar Sync	0 = No, 5 = Absolutely	0.00	5.00	3.75	1.61	2.59	128

	Field	Minimum	Maximum	Mean	Std Deviation	Variance	Count
Suggestions based on Preferences	0 = No, 5 = Absolutely	0.00	5.00	4.09	1.23	1.51	128

	Answer	%	Count
Display Event Details	Yes	99.28%	137
	No	0.72%	1
	Total	100%	138

Figure 22 - Preferred Features

All survey participants were polled for their desire for Calendar Sync, Suggestions based on Preferences, and Event Detail Displays. All three features received overly favorable responses highlighting students have first a desire for interconnection between the application and their calendar for notifications. Second, WPI students desire an application that can help them make decisions faster and better to have more enjoyable experiences. Third, 99% of all those who filled out the survey had a desire to be able to know more information about an event. The desire for detailed information means the final product must allow consumers to have greater knowledge of an event to make better decisions for themselves.

However, decision making must be simple and streamlined for all consumer levels. Event creation must be simplistic, easy to use, and maintain high functionality. Event viewing, synchronizing, and other consumer capabilities must maintain a heavy visual focus similar to Instagram or scrolling online retail applications. Our results tell us that a picture holds more weight than any writing and will catch a viewer's eye beyond any other possibility as seen from Focus Groups, Interviews, and Informal Discussions.

4.1.6 Survey's Effect on Deliverables

Based on preliminary survey results, WPI undergraduates expressed an overall receptiveness to utilize an application for WPI campus events. Specifically, the survey results called for the following functionalities:

1. The ability to view campus event details (US-01);
2. The ability to set preferences to influence the campus events that are shown (US-04, US-06, US-07, US-08); and,

3. The ability to synchronize interesting events to their calendars (PB-03).

4.2 Focus Groups Results

Focus groups maintain an important value to acquiring individual level information on our MQP prototype and eventual MVP. All information regarding focus group results are visible below. The data is from Student Executives who run organizations on the WPI campus. Each set of results grants the team important and required information to ensure a successful product from development, testing, to product completion.

The student executive focus group yielded important data from a top-level perspective. These focus groups included leaders in both small and large campus organizations. It was determined from the executive focus group that our development priorities should be:

1. Utilizing an intuitive user interface like Instagram (US-05);
2. Creating event attendance and interest tracking capabilities (U-14); and,
3. Event timing and location collision detection (PB-04).

Current executives were appreciative and excited about the application initiative as they are faced with numerous issues scheduling events and obtaining knowledge of how to improve events beyond spontaneous biased information. Furthermore, executives currently must go over and beyond to advertise events across numerous sources to attempt to gain new attendees beyond those within current organization social circles. Should a streamlined service be available like current social media applications, they would readily use it and advocate for the application. Overall, should a simple to use, calendar-linked, and visual heavy application be developed all executives have a vowed to push for product adoption.

4.3 Interviews Results

Multiple interviews were completed between all four class levels of WPI undergraduates. Interviews resulted in the discovery of several potential requirements which influenced the development of the application. Themes included:

1. Utilizing an intuitive interface to ensure users could retrieve information for an event efficiently (US-05, US-09);
2. Displaying basic event information to users on the first level of interaction and more detailed information after further interaction such as click-throughs (US-01, US-03, US-16, PB-08);
3. Making event cards distinguishable from one another (US-05); and,
4. Designing event card templates that are visually appealing to users (US-05).

The four requirement themes listed above were the most prevalent commentary received during all completed interviews. The first theme became prevalent in every conversation in both interview and informal discussion results. Student's do not have time to learn new technology that requires lengthy time to respond. An application must respond rapidly and all decision-making tools must be intuitive and like current products.

The second theme was promoted from executive interviews and then became prevalent in student interviews over quick decision making. Nearly all interviewed students especially upperclassmen with busy schedules must be able to easily see basic decision making information on an event with minimal effort. If the application required greater effort to simply see an event, they would not use it.

The final two themes we saw came hand in hand with each other. All Events must be easily distinguished so users do not confuse them together and they must be visually appealing with minimal text to catch attention. We were reminded countlessly that an image will attract someone to use the application and look at an event over any lengthy text. Numerous students stated that with social media drowning in text information they avoid it and simply look for well-done graphics to prompt them to an event or decision.

4.4 Functional and Nonfunctional Requirements

As discussed in the previous sections, to determine our application's functional and nonfunctional requirements, we used surveys, focus groups, and interviews. The requirements we identified are as follows:

Functional Requirements:

1. **Club Content Management**
   F1.1 The system will provide functionality for the creation of club profiles: name, description, picture.
   F1.2 The system will provide functionality for club admins to edit club profiles.
2. **Event Content Management**
   F2.1 The system will provide functionality for the creation of events by clubs: name, description, location, time, picture, audience.
   F2.2 The system will provide functionality for only club admins to post events.
   F2.3 The system will provide functionality for club admins to edit events sponsored by their clubs.
3. **User-Club Interactions**
   F3.1 The system will provide functionality for users the ability to view club pages and information.
   F3.2 The system will provide functionality for users the ability to search for club pages and information.
   F3.3 The system will provide functionality for users to subscribe to clubs.
   F3.4 The system will provide functionality for users to identify events a club is organizing from the club's information page.

F3.5 The system will provide functionality for users to quick access clubs they are subscribed or admined to via the drawer menu.

4. **User-Event Interactions**

   F4.1 The system will provide functionality for users to view event pages and information that are uploaded.

   F4.2 The system will provide functionality for users to indicate that they are interested in attending an event.

   F4.1 The system will provide functionality for users to filter events based on the event's category.

   F4.4 The system will provide functionality for users to view their upcoming events (events the user is attending) via the drawer menu.

5. **User Identification**

   F5.1 The system will provide functionality to identify the user based on their login information.

   F5.2 The system will provide functionality for user to change their password when requested.

6. **Permission Granting**

   F6.1 The system will provide functionality for users to grant other users creation and editing permissions.

Non-Functional Requirements:

1. **Operational**

   N1.1 The system should be operational on iOS, Android, and web browsers.

   N1.2 The system should be integrated with the WPI's Central Authentication System (CAS).

   N1.3 The should only allow for the uploading of specific file types when creating an event or club.

   N1.4 The system should only allow for the correct type of data to be entered specific text boxes and menus.

   N1.5 During a system restart, the system will be able to return to a functioning state.

   N1.6 Club admins must be able to create, edit, and delete any content related to their affiliated club.

2. **Performance**

   N2.1 The system should handle up to six (6) events created per minute.

   N2.2 The system can handle at least 50 users simultaneously.

   N2.3 The system should query search and filter results <5 seconds.

   N2.4 The system should take no more than 10 seconds to execute an interaction.

3. **Security**

   N3.1 The system will use CAS to perform and protect user's login information.

N3.2 The system should only permit club event planners to create events for their respective clubs.

N3.3 Two-factor authentication should be used to initially sign up a new user.

4. **Cultural/Political**

N4.1 The systems information and data should be protected per WPI's Information Technology Department policies.

N4.2 The system should allow WPI's Student Activities Office (SAO) to have school-wide admin permissions to monitor interactions within the application.

N4.3 The system should allow WPI's Student Government Association (SGA) to have school-wide admin permissions to monitor interactions within the application.

N4.4 The system should allow students to be better informed of events, and centralize where they look for information about events.

N4.5 The system should make the process of finding campus events easier which should increase attendance at WPI club events for those who use the application.

N4.6 The system should give WPI organizations more information/metrics about their events.

4.4 User Types

Within the application there are a variety of user types to model the behaviors of the diverse user base. Users are categorized into the following categories: general users (users), club administrators (club admin), and school administrators (school admin).

Users

General users are the baseline permission level for any WPI undergraduate user in the application and are limited to the level of interaction they can have in the application. Users are confined to accessing event and club information, receiving notifications, expressing interest, and setting preferences via the application.

Club Admin

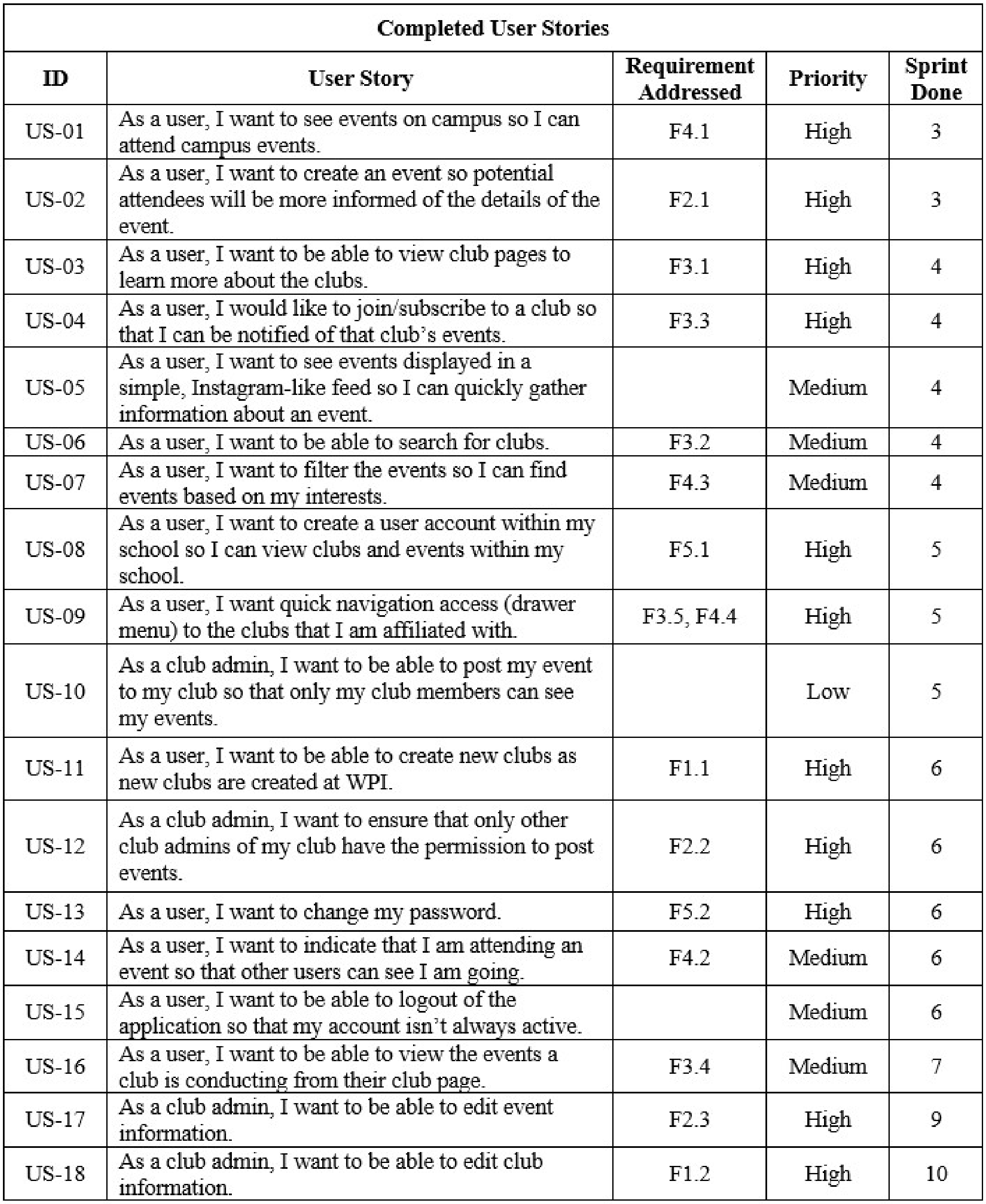
Whereas users can only interact with event/club content at a surface level, club admins have content creation/editing permissions within the application. In addition to being content drivers, club admins would ideally be able to extract club pertinent information and analytics.

School Admin

School admins are unlike club admins and general users as they are not expected to interact with content in the same way that student users will. Instead, school admins are anticipated to have functionalities such as school-wide monitoring, quality assurance, and analytics to improve and monitor the event discovery experience.

4.5 User Stories

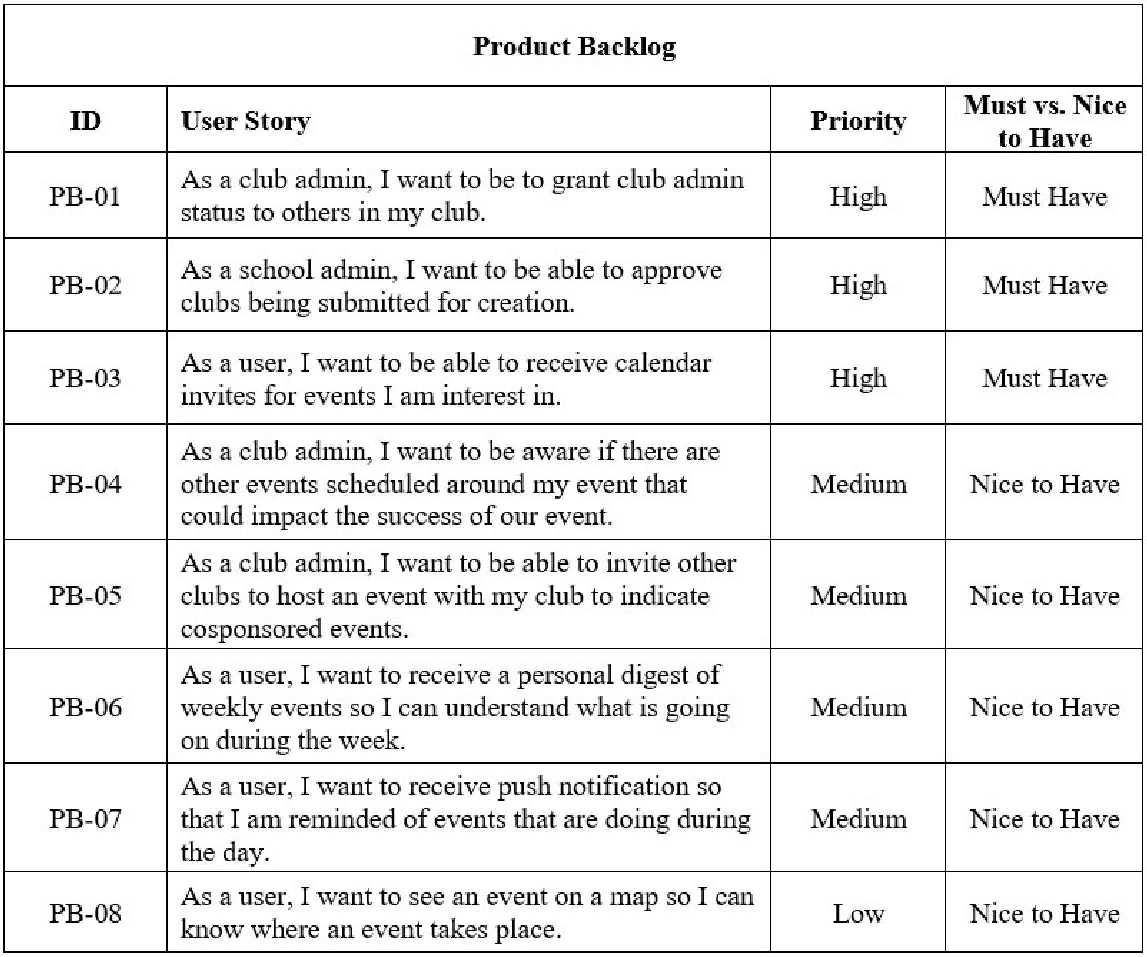
When developing user stories, requirements are broken down into user specific scenarios and prioritized based primarily on user feedback. For the campus event finder application, the user stories below were executed:



Completed User Stories

ID	User Story	Requirement Addressed	Priority	Sprint Done
US-01	As a user, I want to see events on campus so I can attend campus events.	F4.1	High	3
US-02	As a user, I want to create an event so potential attendees will be more informed of the details of the event.	F2.1	High	3
US-03	As a user, I want to be able to view club pages to learn more about the clubs.	F3.1	High	4
US-04	As a user, I would like to join/subscribe to a club so that I can be notified of that club's events.	F3.3	High	4
US-05	As a user, I want to see events displayed in a simple, Instagram-like feed so I can quickly gather information about an event.		Medium	4
US-06	As a user, I want to be able to search for clubs.	F3.2	Medium	4
US-07	As a user, I want to filter the events so I can find events based on my interests.	F4.3	Medium	4
US-08	As a user, I want to create a user account within my school so I can view clubs and events within my school.	F5.1	High	5
US-09	As a user, I want quick navigation access (drawer menu) to the clubs that I am affiliated with.	F3.5, F4.4	High	5
US-10	As a club admin, I want to be able to post my event to my club so that only my club members can see my events.		Low	5
US-11	As a user, I want to be able to create new clubs as new clubs are created at WPI.	F1.1	High	6
US-12	As a club admin, I want to ensure that only other club admins of my club have the permission to post events.	F2.2	High	6
US-13	As a user, I want to change my password.	F5.2	High	6
US-14	As a user, I want to indicate that I am attending an event so that other users can see I am going.	F4.2	Medium	6
US-15	As a user, I want to be able to logout of the application so that my account isn't always active.		Medium	6
US-16	As a user, I want to be able to view the events a club is conducting from their club page.	F3.4	Medium	7
US-17	As a club admin, I want to be able to edit event information.	F2.3	High	9
US-18	As a club admin, I want to be able to edit club information.	F1.2	High	10

Figure 23 - Completed User Stories



Product Backlog			
ID	User Story	Priority	Must vs. Nice to Have
PB-01	As a club admin, I want to be to grant club admin status to others in my club.	High	Must Have
PB-02	As a school admin, I want to be able to approve clubs being submitted for creation.	High	Must Have
PB-03	As a user, I want to be able to receive calendar invites for events I am interest in.	High	Must Have
PB-04	As a club admin, I want to be aware if there are other events scheduled around my event that could impact the success of our event.	Medium	Nice to Have
PB-05	As a club admin, I want to be able to invite other clubs to host an event with my club to indicate cosponsored events.	Medium	Nice to Have
PB-06	As a user, I want to receive a personal digest of weekly events so I can understand what is going on during the week.	Medium	Nice to Have
PB-07	As a user, I want to receive push notification so that I am reminded of events that are doing during the day.	Medium	Nice to Have
PB-08	As a user, I want to see an event on a map so I can know where an event takes place.	Low	Nice to Have

Figure 24 - Product Backlog

During A term, the development team focused on creating a barebones application that met the basic needs of both students and event planners in a university setting. During one of our agile sprint meetings, we discussed what user stories we would be focusing on during our first two sprints in A term after our initial architectural spike. Our first sprint focused on the following user story. “As an event planner, I want to create an event so that I can get the most people possible at my event.” This focuses on the audience of event planners and ensures that they will be able to create events for students to examine and determine whether they are worth attending.

Our second sprint focused on the following user story. “As a student, I want to see events on campus so I can better spend my free time.” The audience focus change to students attending a university ensures that they will be able to see events that are happening around them so that they can determine whether to attend them. Each of these user stories focused on the central idea of getting a prototype into the hands of potential users to determine whether this app would be of interest to them and help them save time searching for things they want to do.

B-Term saw heavy development with the Third, Fourth, Fifth, and Sixth Sprint. The Third Sprint saw all attributes from the ERD completed, tests continuously developed, and the UI for the create user, event page, and club pages. The Sprint ended with club relationship database developed along with the club model. The Fourth Sprint saw code being tested and documented, with authentication, Instagram-like viewing, formatting, and implementation in club and student tools being developed. The Sprint concluded with filtering for events and filter button for the entire campus event feed. The Fifth Sprint of B-Term saw the club page population developed, menu drawer creation, sign in and log out styling and development, Campus Feed angular chips, password changes, and concluding with code that allowed clubs to post events exclusively to their clubs and allowance for users to "Joynup" events.

Lastly, in C-term we continued developing features for the application, fully tested the application, and began marketing the application to influential organizations through our beta-testing efforts.

The features developed during C-Term included finalizing the club admin permission functionalities and auto-refreshing the application's content. The club admin permission was important because it addressed user concerns about others posting invalid or incorrect event and club information. Auto-refreshing helped make the application more dynamic and gave user instant feedback in regards to event information. We also made numerous UI updates. We spaced content more evenly throughout the application, added a smoother drawer menu, and altered colors to a specific palette so that the application looked cleaner.

Testing was another crucial step toward a potential launch of the application. Without proper and thorough testing, no company or university information technology department would adopt our application. Therefore, we developed a test plan to verify all of our use cases and test cases work properly.

4.6 Use Case Diagrams

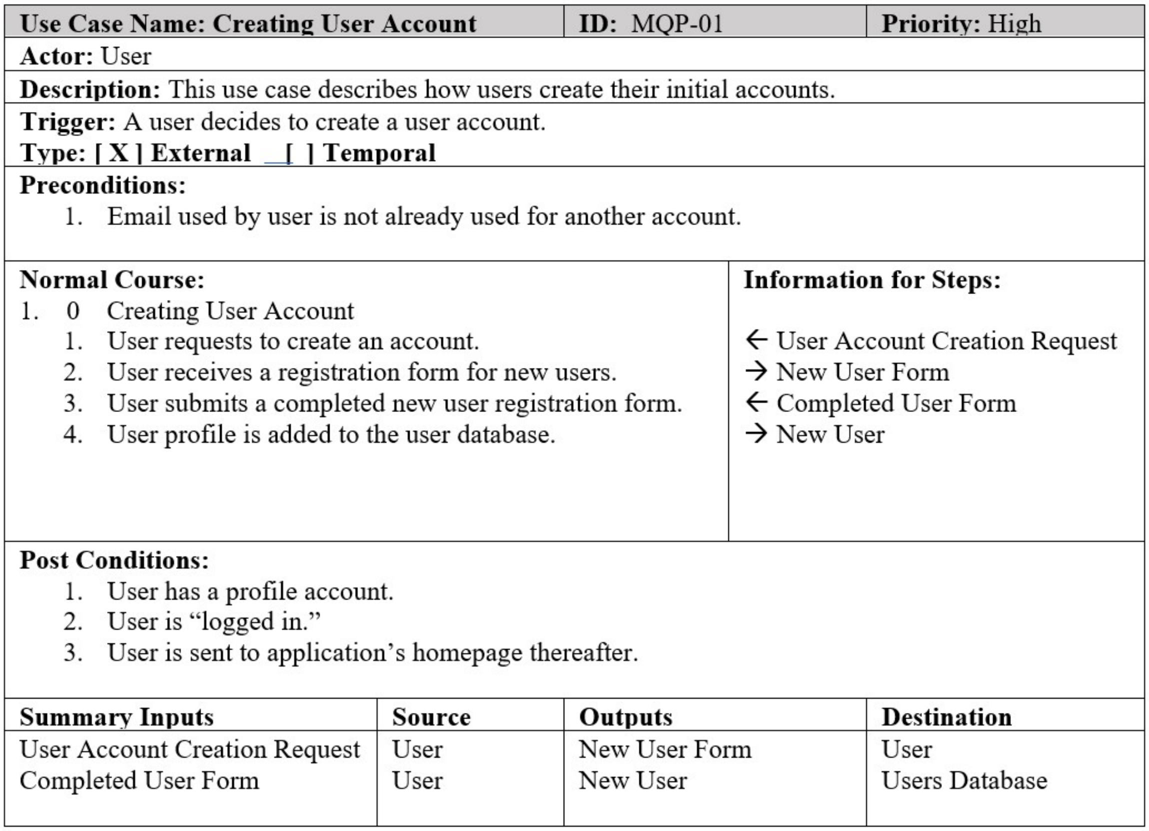
The use cases for this application are based off user stories detailed in Section 4.5. Since the application was incrementally developed, the application and its subsequent use cases were changed periodically from their original form. Below are the use cases and their respective descriptions in their current state. For the sake of encapsulating alike information, our use cases are broken down into five application operations: (1) Managing User Accounts; (2) Interacting with Clubs; (3) Interacting with Events; (4) Display Application Info; and (5) Searching/Filtering for Information.

4.6.1 Managing User Account

The use cases found in this section pertain to the creation, editing, and account status of the users account. They include: Creating a User Account, Logging into Account, Logging Out of Account, and Changing an Account Password.

4.6.1.1 Creating a User Account

This use case details the process of a user creating an account for the first time. The process starts with the user intending to sign up for the application by pressing the “Signup” tab. Next, the user inputs the necessary fields on the “Signup” form to create a user profile, such as @wpi.edu email account, password, and name. Finally, the user submits the form to officially create their user account.



Use Case Name: Creating User Account	ID: MQP-01	Priority: High

Actor: User

Description: This use case describes how users create their initial accounts.

Trigger: A user decides to create a user account.
Type: [X] External [ ] Temporal

Preconditions:

1. Email used by user is not already used for another account.

Normal Course:	Information for Steps:
1. 0 Creating User Account	
1. User requests to create an account.	← User Account Creation Request
2. User receives a registration form for new users.	→ New User Form
3. User submits a completed new user registration form.	← Completed User Form
4. User profile is added to the user database.	→ New User

Post Conditions:

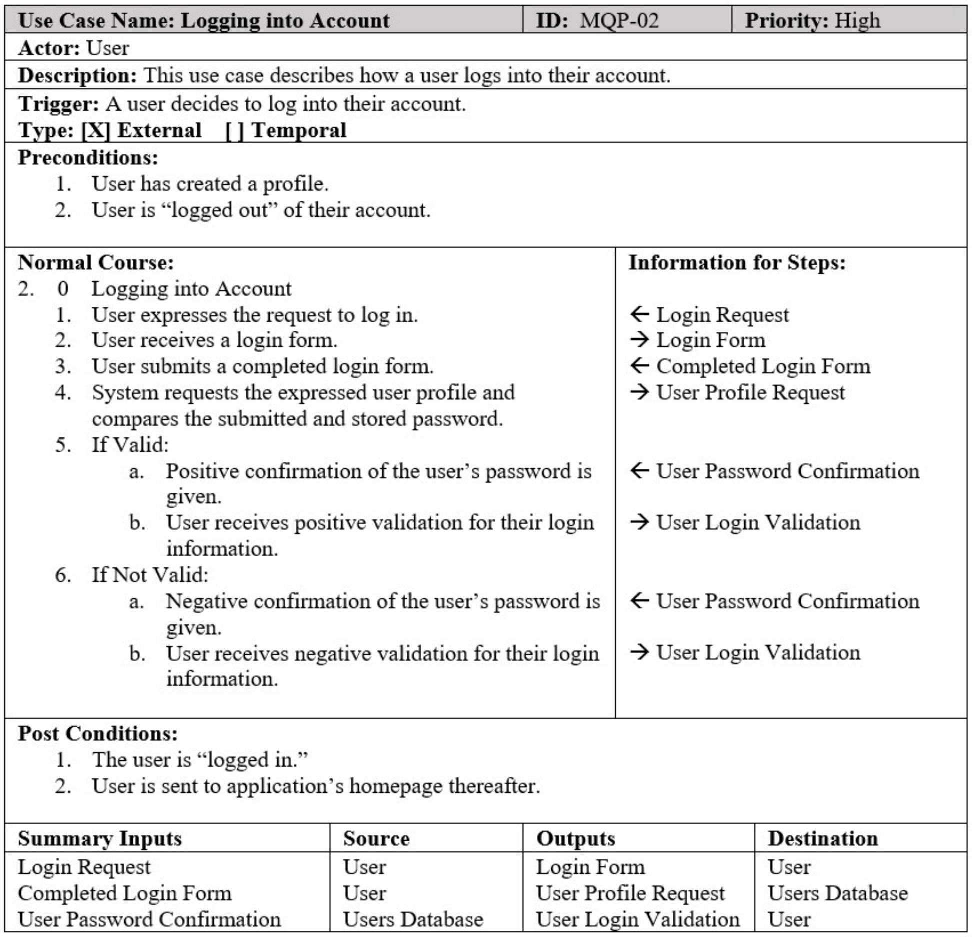
1. User has a profile account.
2. User is “logged in.”
3. User is sent to application’s homepage thereafter.

Summary Inputs	Source	Outputs	Destination
User Account Creation Request	User	New User Form	User
Completed User Form	User	New User	Users Database

Figure 25 - Use Case #1: Creating a User Account

4.6.1.2 Logging into Account

This use case describes the process of how a user logs into their account. Initially the user indicates they want to log in by pressing the “Login” tab. Next the user submits their user email and password until they are logged in or blocked out; in either case the user will be notified.



Use Case Name: Logging into Account	**ID:** MQP-02	**Priority:** High

Actor: User

Description: This use case describes how a user logs into their account.

Trigger: A user decides to log into their account.
Type: [X] External [ ] Temporal

Preconditions:
1. User has created a profile.
2. User is “logged out” of their account.

Normal Course:	**Information for Steps:**
2. 0 Logging into Account	
1. User expresses the request to log in.	← Login Request
2. User receives a login form.	→ Login Form
3. User submits a completed login form.	← Completed Login Form
4. System requests the expressed user profile and compares the submitted and stored password.	→ User Profile Request
5. If Valid:	
a. Positive confirmation of the user’s password is given.	← User Password Confirmation
b. User receives positive validation for their login information.	→ User Login Validation
6. If Not Valid:	
a. Negative confirmation of the user’s password is given.	← User Password Confirmation
b. User receives negative validation for their login information.	→ User Login Validation

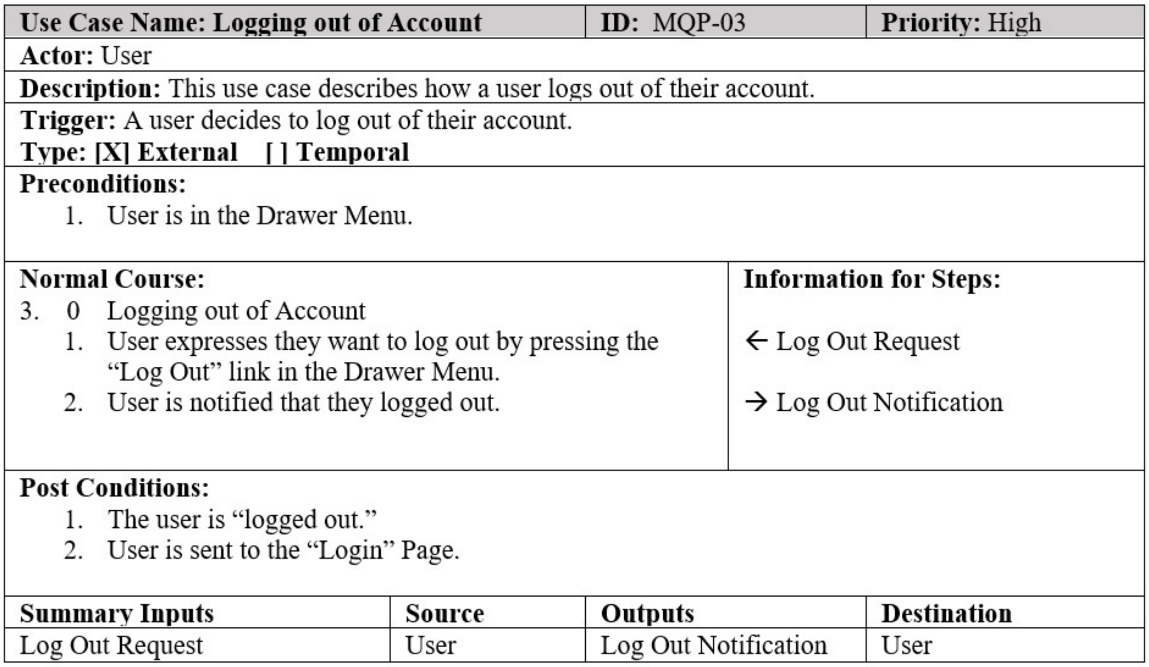
Post Conditions:
1. The user is “logged in.”
2. User is sent to application’s homepage thereafter.

Summary Inputs	**Source**	**Outputs**	**Destination**
Login Request	User	Login Form	User
Completed Login Form	User	User Profile Request	Users Database
User Password Confirmation	Users Database	User Login Validation	User

Figure 26 - Use Case #2: Logging into Account

4.6.1.3 Logging Out of Account

This use case describes the process in which a user logs out of the application. Given that the user is in the drawer menu, the user will simply press the “Log Out” link. The system should log the user out of their account and the user will be brought back to the “Login” page.

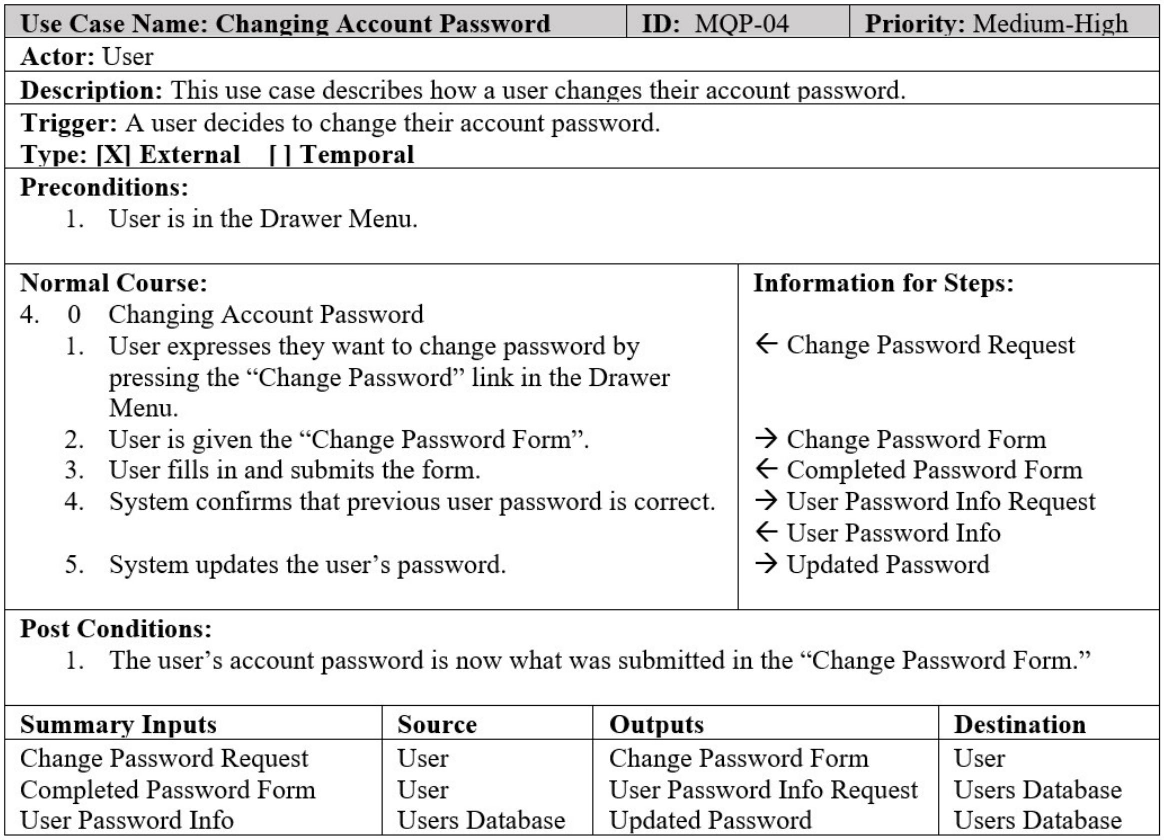


Use Case Name: Logging out of Account		**ID:** MQP-03	**Priority:** High
Actor: User			
Description: This use case describes how a user logs out of their account.			
Trigger: A user decides to log out of their account. **Type: [X] External [ ] Temporal**			
Preconditions: 1. User is in the Drawer Menu.			
Normal Course: 3. 0 Logging out of Account 1. User expresses they want to log out by pressing the “Log Out” link in the Drawer Menu. 2. User is notified that they logged out.		**Information for Steps:** ← Log Out Request → Log Out Notification	
Post Conditions: 1. The user is “logged out.” 2. User is sent to the “Login” Page.			
Summary Inputs	**Source**	**Outputs**	**Destination**
Log Out Request	User	Log Out Notification	User

Figure 27 - Use Case #3: Logging Out of Account

4.6.1.4 Changing an Account Password

This case describes the process in which a user changes their account password. It walks through the steps of having to enter the drawer menu, requesting to change their password, confirming their existing password, and then inputting their new password.



Use Case Name: Changing Account Password	**ID:** MQP-04	**Priority:** Medium-High

Actor: User

Description: This use case describes how a user changes their account password.

Trigger: A user decides to change their account password.
Type: [X] External [ ] Temporal

Preconditions:
1. User is in the Drawer Menu.

Normal Course:	**Information for Steps:**
4. 0 Changing Account Password	
1. User expresses they want to change password by pressing the “Change Password” link in the Drawer Menu.	← Change Password Request
2. User is given the “Change Password Form”.	→ Change Password Form
3. User fills in and submits the form.	← Completed Password Form
4. System confirms that previous user password is correct.	→ User Password Info Request ← User Password Info
5. System updates the user’s password.	→ Updated Password

Post Conditions:
1. The user’s account password is now what was submitted in the “Change Password Form.”

Summary Inputs	**Source**	**Outputs**	**Destination**
Change Password Request	User	Change Password Form	User
Completed Password Form	User	User Password Info Request	Users Database
User Password Info	Users Database	Updated Password	Users Database

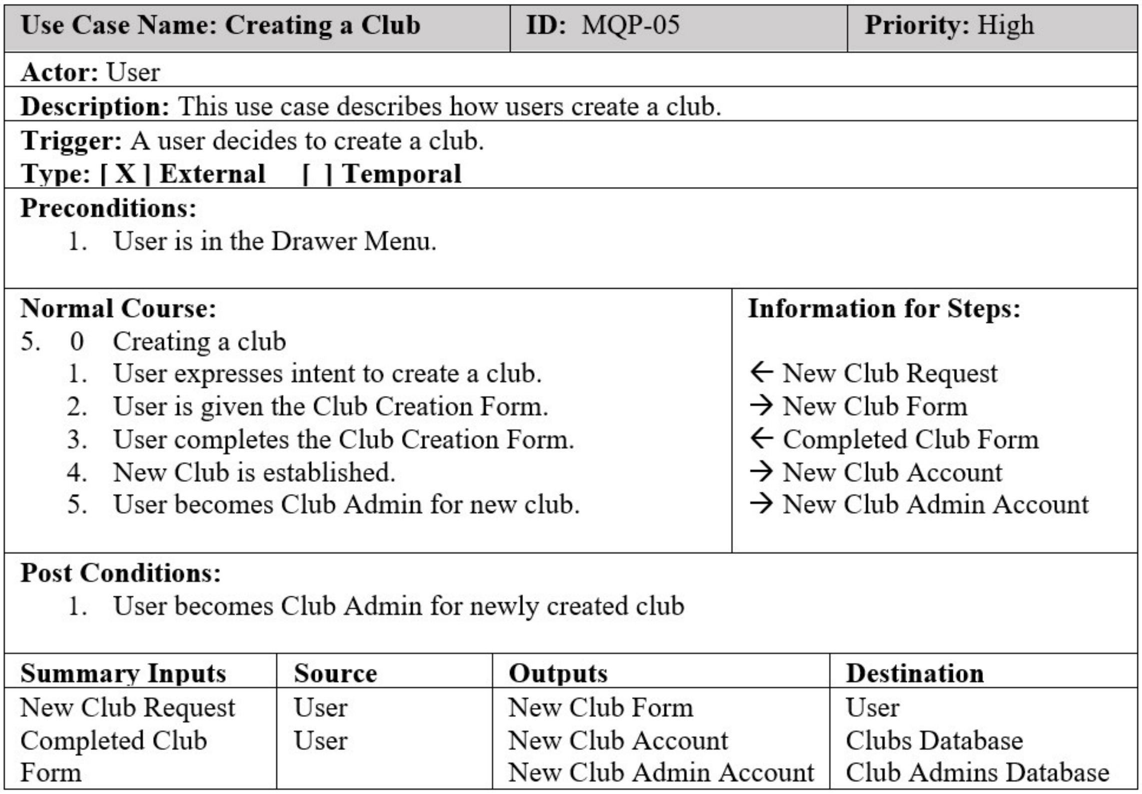
Figure 28 - Use Case #4: Changing Account Password

4.6.2 Interacting with Clubs

The use cases in this section focus on the interactions both Users and Club Admins have with clubs. The use cases include: Creating a Club, Becoming a Club Member, Unsubscribing from a Club, and Editing Club Information.

4.6.2.1 Creating a Club

This use case describes the process in which a user would create a new club account for the application. First, the user needs to be in the drawer menu and select the "Create a Club" link. Next, the user is given a Club Creation Form which they are to complete and submit. The club will then be added to the application, the submitting user will become a club admin, and the user will be notified that the club now exists.



Use Case Name: Creating a Club	**ID:** MQP-05	**Priority:** High

Actor: User

Description: This use case describes how users create a club.

Trigger: A user decides to create a club.
Type: [X] External [ ] Temporal

Preconditions:
1. User is in the Drawer Menu.

Normal Course:	**Information for Steps:**
5. 0 Creating a club	
1. User expresses intent to create a club.	← New Club Request
2. User is given the Club Creation Form.	→ New Club Form
3. User completes the Club Creation Form.	← Completed Club Form
4. New Club is established.	→ New Club Account
5. User becomes Club Admin for new club.	→ New Club Admin Account

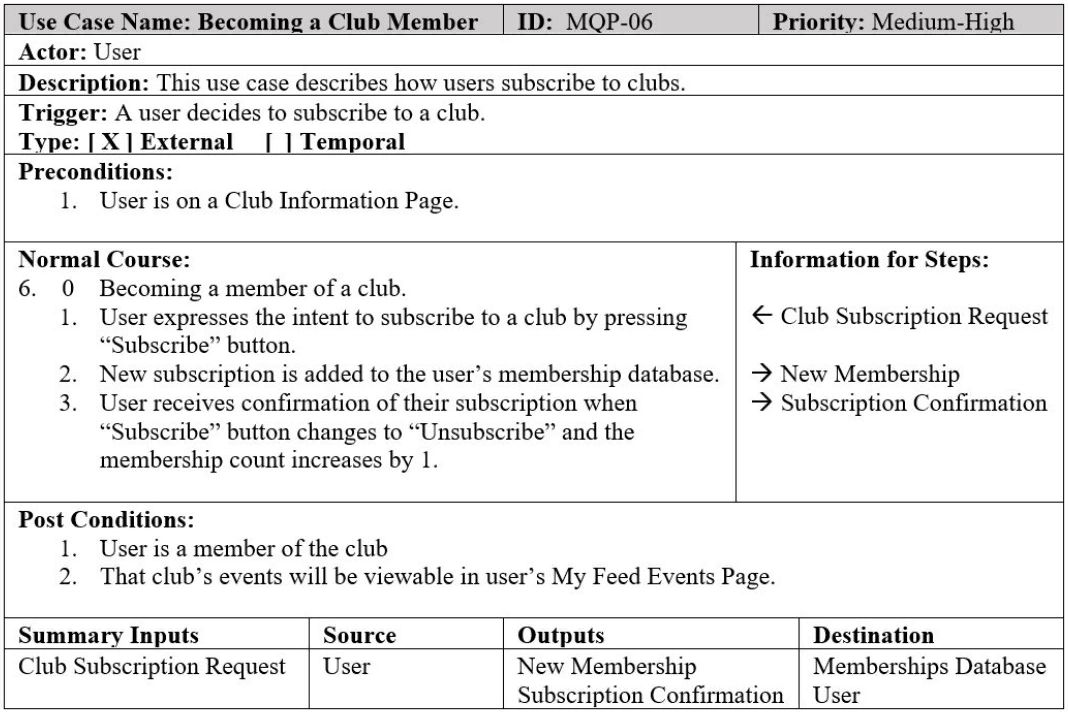
Post Conditions:
1. User becomes Club Admin for newly created club

Summary Inputs	**Source**	**Outputs**	**Destination**
New Club Request	User	New Club Form	User
Completed Club Form	User	New Club Account	Clubs Database
		New Club Admin Account	Club Admins Database

Figure 29 - Use Case #5: Creating a Club

4.6.2.1 Becoming a Club Member

This use case describes the process a user takes to become a member of a club. When a user is on a club's page, the user may click the "Subscribe" button to become a member of that club. Afterwards, the system will record that the user is now a member of that club.



Use Case Name: Becoming a Club Member	**ID:** MQP-06	**Priority:** Medium-High

Actor: User

Description: This use case describes how users subscribe to clubs.

Trigger: A user decides to subscribe to a club.
Type: [X] External [ ] Temporal

Preconditions:
1. User is on a Club Information Page.

Normal Course:	**Information for Steps:**
6. 0 Becoming a member of a club.	
1. User expresses the intent to subscribe to a club by pressing "Subscribe" button.	← Club Subscription Request
2. New subscription is added to the user's membership database.	→ New Membership
3. User receives confirmation of their subscription when "Subscribe" button changes to "Unsubscribe" and the membership count increases by 1.	→ Subscription Confirmation

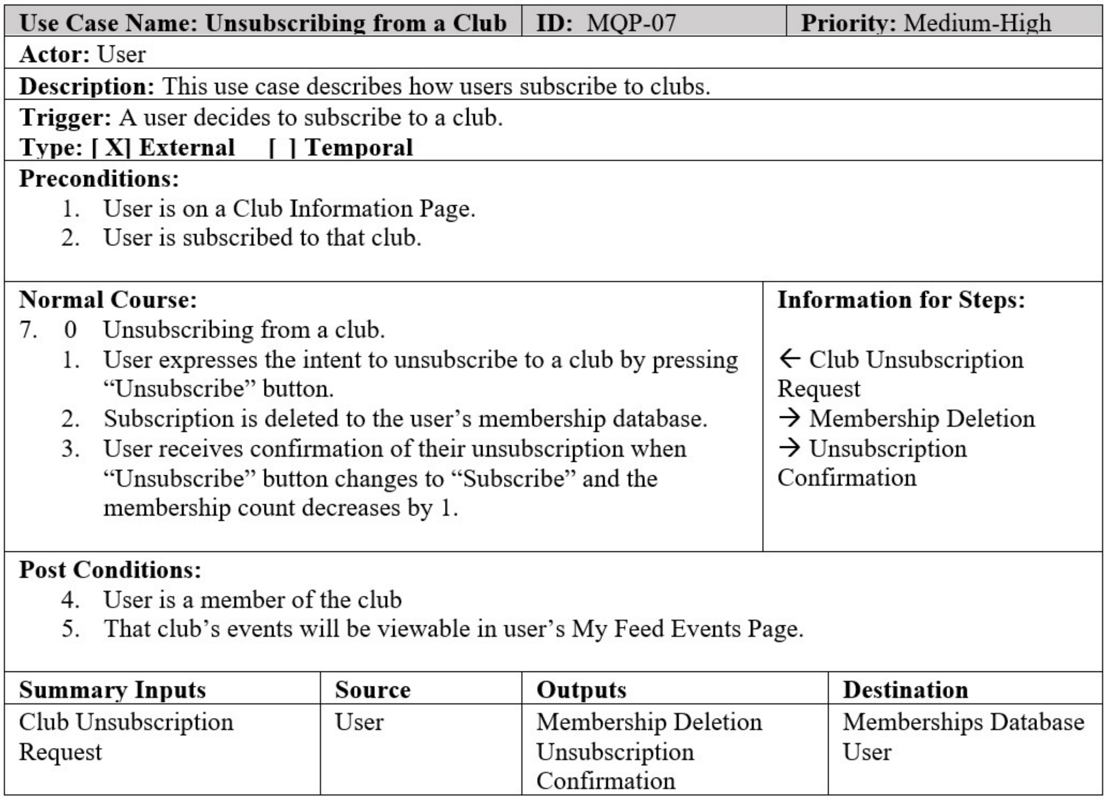
Post Conditions:
1. User is a member of the club
2. That club's events will be viewable in user's My Feed Events Page.

Summary Inputs	**Source**	**Outputs**	**Destination**
Club Subscription Request	User	New Membership Subscription Confirmation	Memberships Database User

Figure 30 - Use Case #6: Becoming a Club Member

4.6.2.3 Unsubscribing for a Club

This use case describes the process in which a user would unsubscribe from a club. Essentially, it is the same process as subscribing to a club, but the user must already be a club member to unsubscribe.



Use Case Name: Unsubscribing from a Club	**ID:** MQP-07	**Priority:** Medium-High

Actor: User

Description: This use case describes how users subscribe to clubs.

Trigger: A user decides to subscribe to a club.
Type: [X] External [ ] Temporal

Preconditions:

1. User is on a Club Information Page.
2. User is subscribed to that club.

Normal Course:	**Information for Steps:**
7. 0 Unsubscribing from a club. 1. User expresses the intent to unsubscribe to a club by pressing "Unsubscribe" button. 2. Subscription is deleted to the user's membership database. 3. User receives confirmation of their unsubscription when "Unsubscribe" button changes to "Subscribe" and the membership count decreases by 1.	← Club Unsubscription Request → Membership Deletion → Unsubscription Confirmation

Post Conditions:

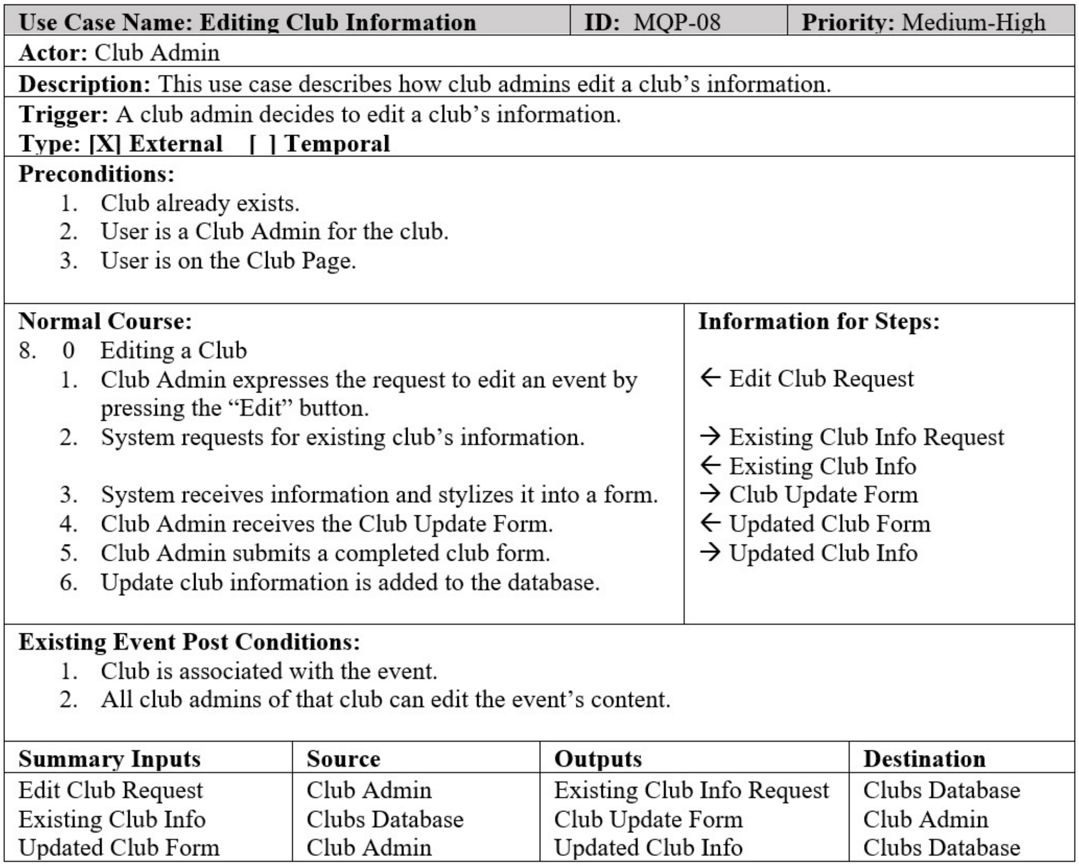
4. User is a member of the club
5. That club's events will be viewable in user's My Feed Events Page.

Summary Inputs	**Source**	**Outputs**	**Destination**
Club Unsubscription Request	User	Membership Deletion Unsubscription Confirmation	Memberships Database User

Figure 31 - Use Case #7: Unsubscribing from a Club

4.6.2.4 Editing Club Information

This use case describes the process a club admin would take when editing a club's information. Once on the club page, a club admin will request to edit the club's info then submit a form with the updated information. Afterwards, the club's info should be updated for all users.



Use Case Name: Editing Club Information	**ID:** MQP-08	**Priority:** Medium-High

Actor: Club Admin

Description: This use case describes how club admins edit a club's information.

Trigger: A club admin decides to edit a club's information.
Type: [X] External [ ] Temporal

Preconditions:

1. Club already exists.
2. User is a Club Admin for the club.
3. User is on the Club Page.

Normal Course:	**Information for Steps:**
8. 0 Editing a Club	
1. Club Admin expresses the request to edit an event by pressing the "Edit" button.	← Edit Club Request
2. System requests for existing club's information.	→ Existing Club Info Request ← Existing Club Info
3. System receives information and stylizes it into a form.	→ Club Update Form
4. Club Admin receives the Club Update Form.	← Updated Club Form
5. Club Admin submits a completed club form.	→ Updated Club Info
6. Update club information is added to the database.	

Existing Event Post Conditions:

1. Club is associated with the event.
2. All club admins of that club can edit the event's content.

Summary Inputs	**Source**	**Outputs**	**Destination**
Edit Club Request	Club Admin	Existing Club Info Request	Clubs Database
Existing Club Info	Clubs Database	Club Update Form	Club Admin
Updated Club Form	Club Admin	Updated Club Info	Clubs Database

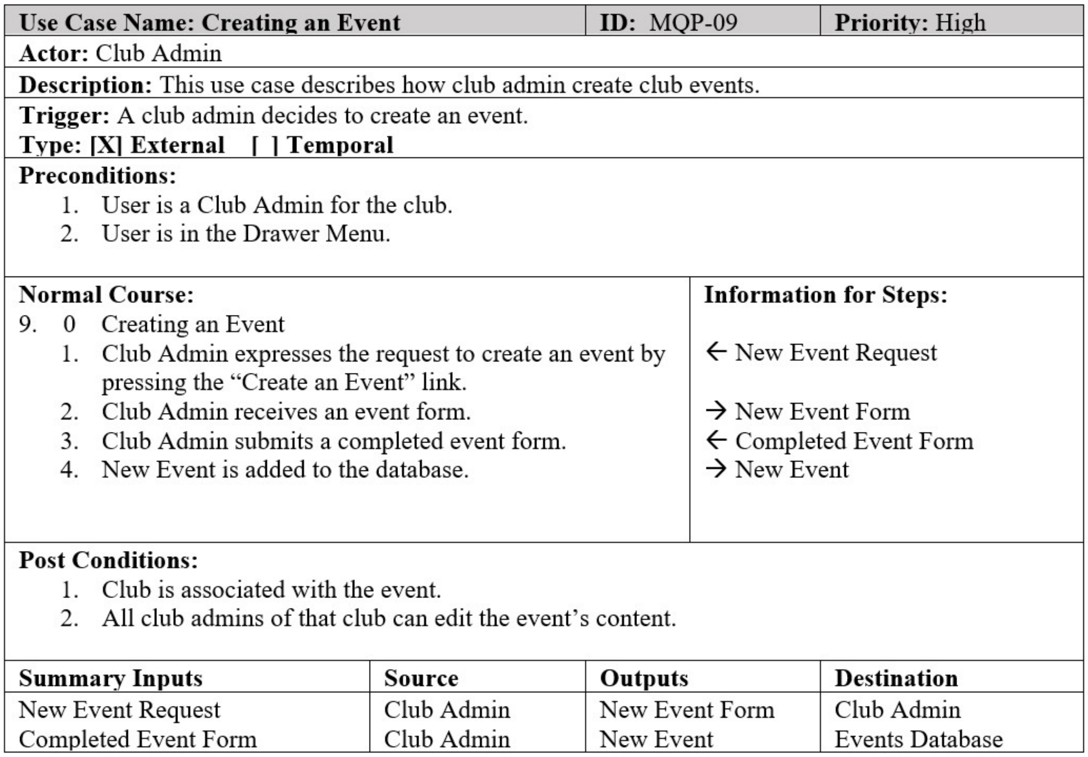
Figure 32 - Use Case #8: Editing Club Information

4.6.3 Interacting with Events

The use cases in this section focus on the interactions both users and club admins have with events. The use cases include: Creating an Event, Attending an Event, Unattending an Event, and Editing Event Information.

4.6.3.1 Creating an Event

This use case describes the process a club admin takes to create a new event for their club. First the club admin will press the "+" button to indicate they want to create an event. Next, the club admin will be given a form that they complete and submit to the system. Once the form is submitted, the event is created and can be viewed in the application.

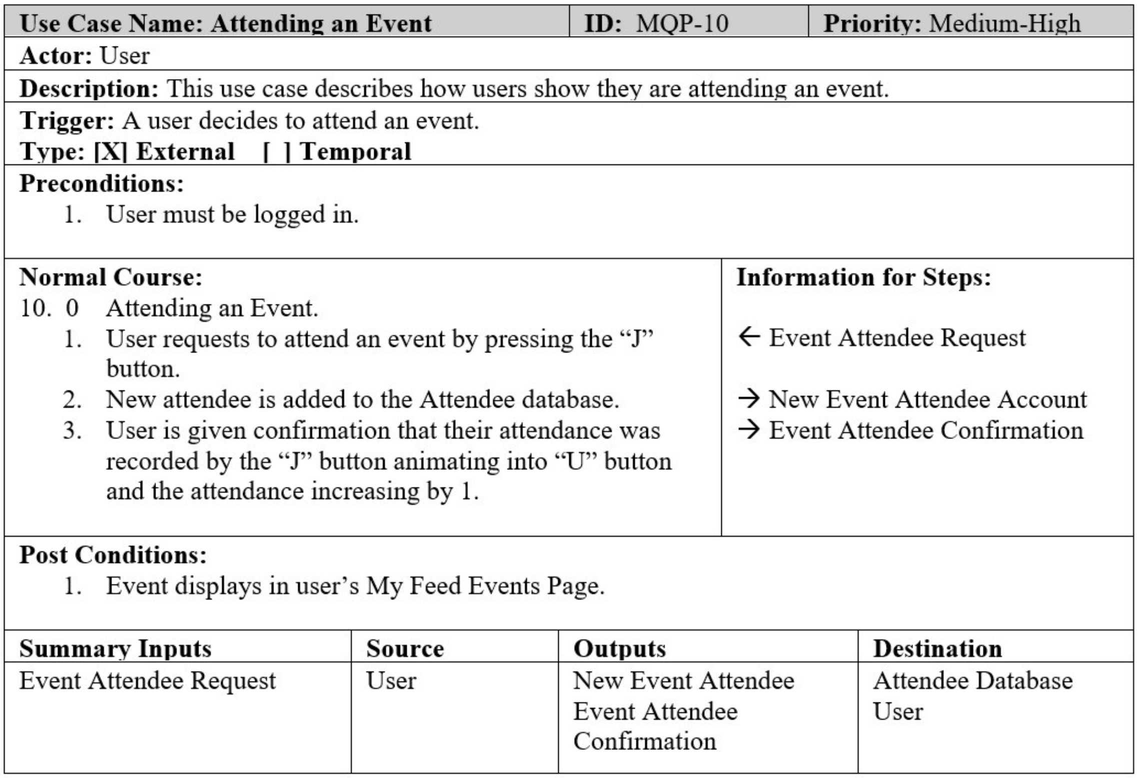


Use Case Name: Creating an Event		**ID:** MQP-09	**Priority:** High
Actor: Club Admin			
Description: This use case describes how club admin create club events.			
Trigger: A club admin decides to create an event. **Type: [X] External [ ] Temporal**			
Preconditions: 1. User is a Club Admin for the club. 2. User is in the Drawer Menu.			
Normal Course: 9. 0 Creating an Event 1. Club Admin expresses the request to create an event by pressing the "Create an Event" link. 2. Club Admin receives an event form. 3. Club Admin submits a completed event form. 4. New Event is added to the database.		**Information for Steps:** ← New Event Request → New Event Form ← Completed Event Form → New Event	
Post Conditions: 1. Club is associated with the event. 2. All club admins of that club can edit the event's content.			
Summary Inputs	**Source**	**Outputs**	**Destination**
New Event Request Completed Event Form	Club Admin Club Admin	New Event Form New Event	Club Admin Events Database

Figure 33 - Use Case #9: Creating an Event

4.6.3.2 Attending an Event

This use case describes the process a user takes to indicate they are interested in attending an event. When on an event, a user may press the “J” button below the event picture to display that they are interested in attending the event. The system confirms that the user’s action was recorded by updating the attendance count and turning the “J” into a “U”.



Use Case Name: Attending an Event	**ID:** MQP-10	**Priority:** Medium-High

Actor: User

Description: This use case describes how users show they are attending an event.

Trigger: A user decides to attend an event.
Type: [X] External [ ] Temporal

Preconditions:
1. User must be logged in.

Normal Course:	**Information for Steps:**
10. 0 Attending an Event.	
1. User requests to attend an event by pressing the “J” button.	← Event Attendee Request
2. New attendee is added to the Attendee database.	→ New Event Attendee Account
3. User is given confirmation that their attendance was recorded by the “J” button animating into “U” button and the attendance increasing by 1.	→ Event Attendee Confirmation

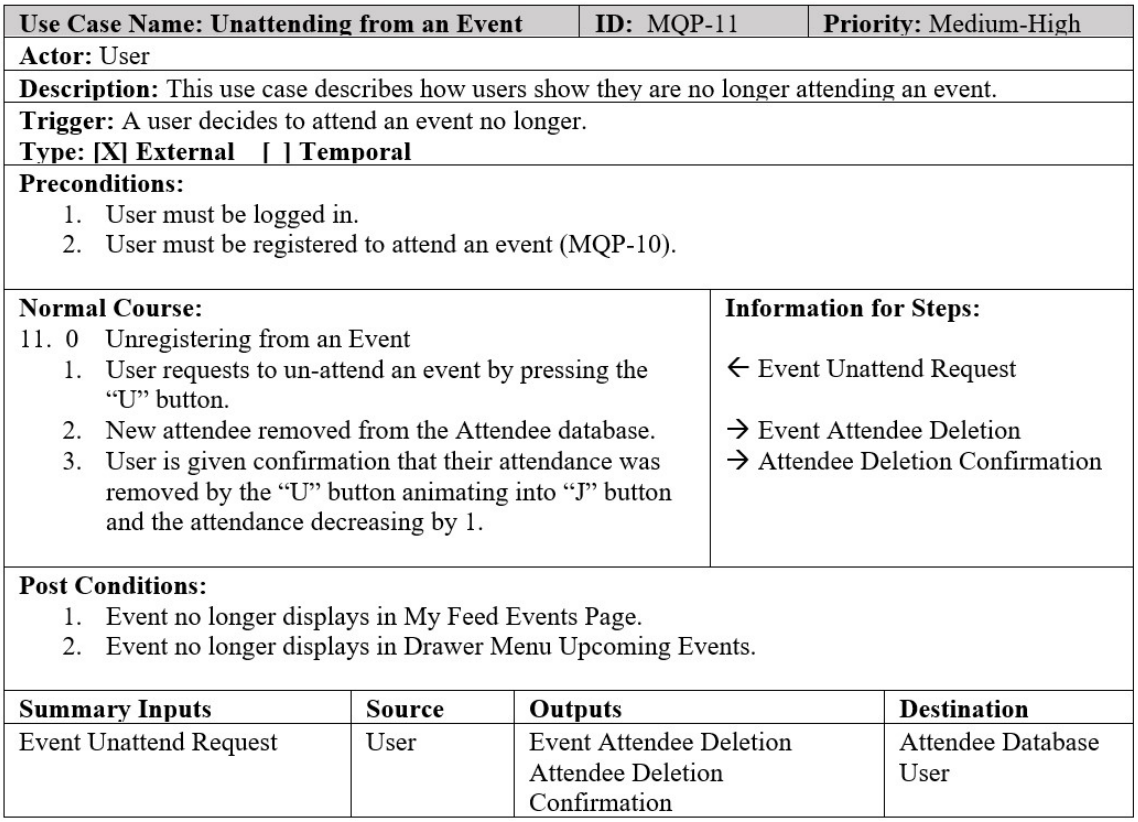
Post Conditions:
1. Event displays in user’s My Feed Events Page.

Summary Inputs	**Source**	**Outputs**	**Destination**
Event Attendee Request	User	New Event Attendee Event Attendee Confirmation	Attendee Database User

Figure 34- Use Case #10: Attending an Event

4.6.3.3 Unattending an Event

This use case describes the process a user takes to indicate they are interested in unattending an event. Essentially, it is the same process as attending an event, however, the user has already expressed interest in attending the event. By repeating the process, the user will no longer express that they are attending the event.



<table>
<tr><td colspan="2"><b>Use Case Name: Unattending from an Event</b></td><td><b>ID:</b> MQP-11</td><td><b>Priority:</b> Medium-High</td></tr>
<tr><td colspan="4"><b>Actor:</b> User</td></tr>
<tr><td colspan="4"><b>Description:</b> This use case describes how users show they are no longer attending an event.</td></tr>
<tr><td colspan="4"><b>Trigger:</b> A user decides to attend an event no longer.
<b>Type: [X] External [ ] Temporal</b></td></tr>
<tr><td colspan="4"><b>Preconditions:</b>
1. User must be logged in.
2. User must be registered to attend an event (MQP-10).</td></tr>
<tr><td colspan="3"><b>Normal Course:</b>
11. 0 Unregistering from an Event
1. User requests to un-attend an event by pressing the "U" button.
2. New attendee removed from the Attendee database.
3. User is given confirmation that their attendance was removed by the "U" button animating into "J" button and the attendance decreasing by 1.</td><td><b>Information for Steps:</b>

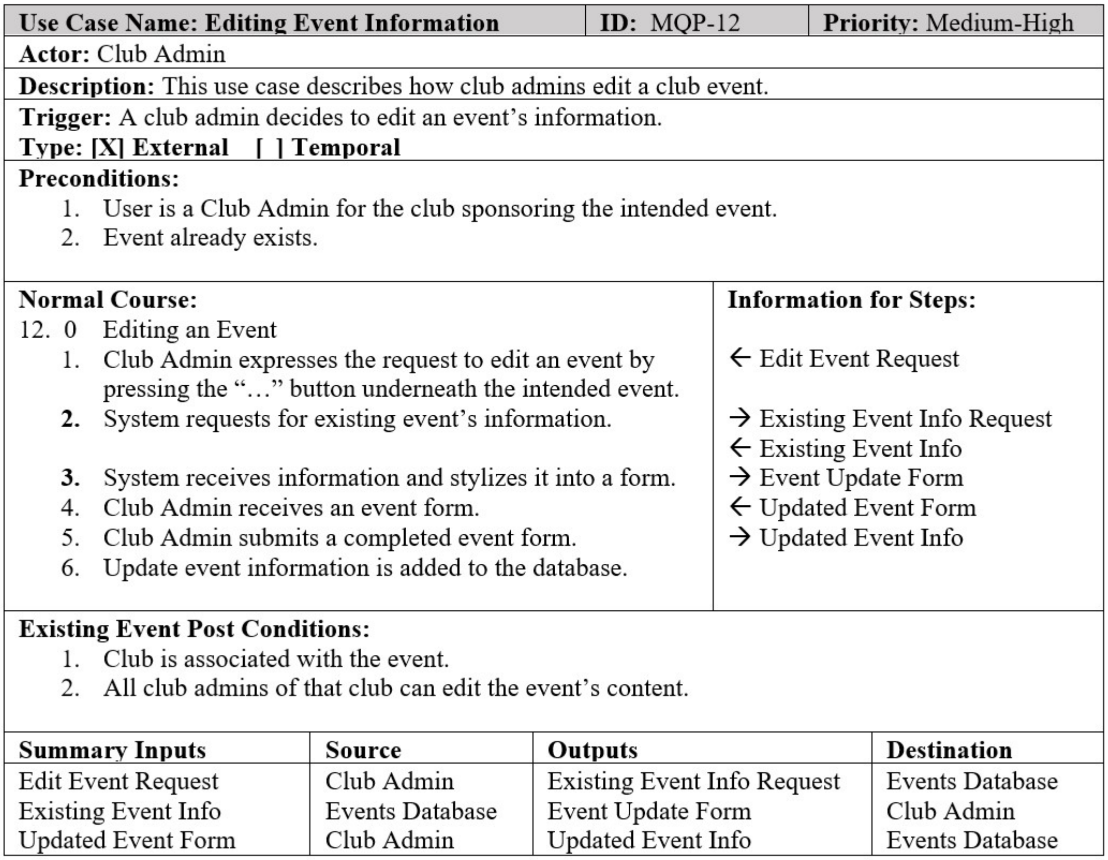
← Event Unattend Request

→ Event Attendee Deletion
→ Attendee Deletion Confirmation</td></tr>
<tr><td colspan="4"><b>Post Conditions:</b>
1. Event no longer displays in My Feed Events Page.
2. Event no longer displays in Drawer Menu Upcoming Events.</td></tr>
<tr><td><b>Summary Inputs</b></td><td><b>Source</b></td><td><b>Outputs</b></td><td><b>Destination</b></td></tr>
<tr><td>Event Unattend Request</td><td>User</td><td>Event Attendee Deletion
Attendee Deletion Confirmation</td><td>Attendee Database
User</td></tr>
</table>

Figure 35 - Use Case #11: Unattending an Event

4.6.3.4 Editing Event Information

This use case describes the process a club admin would take when editing an event's information. Once on an event, a club admin will request to edit the event's info then submit a form with the updated information. Afterwards, the event's info should be updated for all users.



Use Case Name: Editing Event Information	**ID:** MQP-12	**Priority:** Medium-High

Actor: Club Admin

Description: This use case describes how club admins edit a club event.

Trigger: A club admin decides to edit an event's information.
Type: [X] External [ ] Temporal

Preconditions:

1. User is a Club Admin for the club sponsoring the intended event.
2. Event already exists.

Normal Course:	**Information for Steps:**
12. 0 Editing an Event	
1. Club Admin expresses the request to edit an event by pressing the "…" button underneath the intended event.	← Edit Event Request
2. System requests for existing event's information.	→ Existing Event Info Request ← Existing Event Info
3. System receives information and stylizes it into a form.	→ Event Update Form
4. Club Admin receives an event form.	← Updated Event Form
5. Club Admin submits a completed event form.	→ Updated Event Info
6. Update event information is added to the database.	

Existing Event Post Conditions:

1. Club is associated with the event.
2. All club admins of that club can edit the event's content.

Summary Inputs	**Source**	**Outputs**	**Destination**
Edit Event Request	Club Admin	Existing Event Info Request	Events Database
Existing Event Info	Events Database	Event Update Form	Club Admin
Updated Event Form	Club Admin	Updated Event Info	Events Database

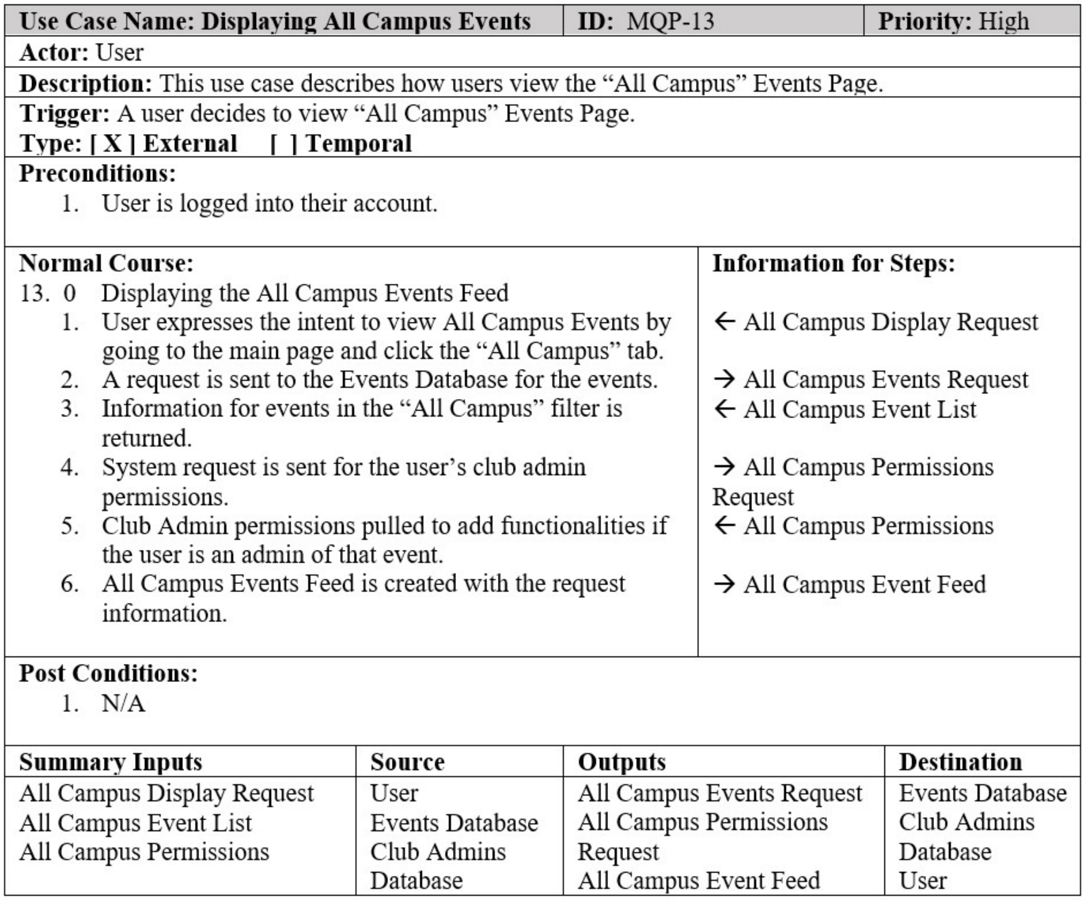
Figure 36 - Use Case #12: Editing Event Information

4.6.4 Displaying App Information

The use cases in this section focus on how information in the application is displayed to the user through various feeds and pages. The use cases covered in this section include: Displaying the All Campus Events Page, Displaying the Event Page, Displaying the My Feed Page, Displaying the Club Page, and Displaying the Drawer Menu.

4.6.4.1 Displaying All Campus Events

This use case describes the process a user takes to view the "All Campus" Events page. First the user will press the "All Campus" tab on the main JoynU page. Once pressed, the system will graphically display all events the user is invited to participate in.



Use Case Name: Displaying All Campus Events	**ID:** MQP-13	**Priority:** High

Actor: User

Description: This use case describes how users view the "All Campus" Events Page.

Trigger: A user decides to view "All Campus" Events Page.
Type: [X] External [ ] Temporal

Preconditions:
1. User is logged into their account.

Normal Course:	**Information for Steps:**
13. 0 Displaying the All Campus Events Feed	
1. User expresses the intent to view All Campus Events by going to the main page and click the "All Campus" tab.	← All Campus Display Request
2. A request is sent to the Events Database for the events.	→ All Campus Events Request
3. Information for events in the "All Campus" filter is returned.	← All Campus Event List
4. System request is sent for the user's club admin permissions.	→ All Campus Permissions Request
5. Club Admin permissions pulled to add functionalities if the user is an admin of that event.	← All Campus Permissions
6. All Campus Events Feed is created with the request information.	→ All Campus Event Feed

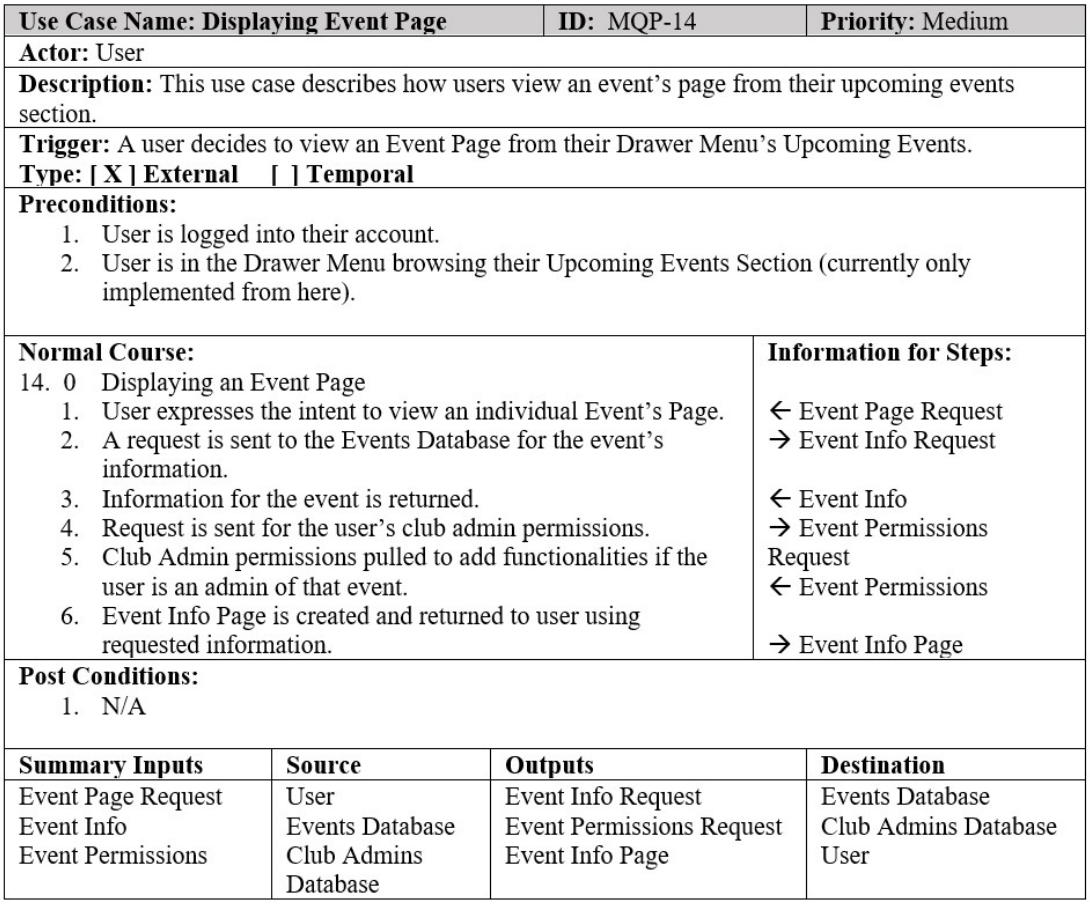
Post Conditions:
1. N/A

Summary Inputs	**Source**	**Outputs**	**Destination**
All Campus Display Request All Campus Event List All Campus Permissions	User Events Database Club Admins Database	All Campus Events Request All Campus Permissions Request All Campus Event Feed	Events Database Club Admins Database User

Figure 37 - Use Case #13: Displaying all Campus Events

4.6.4.2 Displaying Event Information Page

This use case describes the process a user takes to view an Event's Information page. The user will first indicate through pressing the Event Name in the drawer menu they want to view the event's information page. The user will be taken to a graphical display of the event's more detailed information was requested.



Use Case Name: Displaying Event Page	**ID:** MQP-14	**Priority:** Medium

Actor: User

Description: This use case describes how users view an event's page from their upcoming events section.

Trigger: A user decides to view an Event Page from their Drawer Menu's Upcoming Events.
Type: [X] External [ ] Temporal

Preconditions:

1. User is logged into their account.
2. User is in the Drawer Menu browsing their Upcoming Events Section (currently only implemented from here).

Normal Course:	**Information for Steps:**
14. 0 Displaying an Event Page	
1. User expresses the intent to view an individual Event's Page.	← Event Page Request
2. A request is sent to the Events Database for the event's information.	→ Event Info Request
3. Information for the event is returned.	← Event Info
4. Request is sent for the user's club admin permissions.	→ Event Permissions Request
5. Club Admin permissions pulled to add functionalities if the user is an admin of that event.	← Event Permissions
6. Event Info Page is created and returned to user using requested information.	→ Event Info Page

Post Conditions:

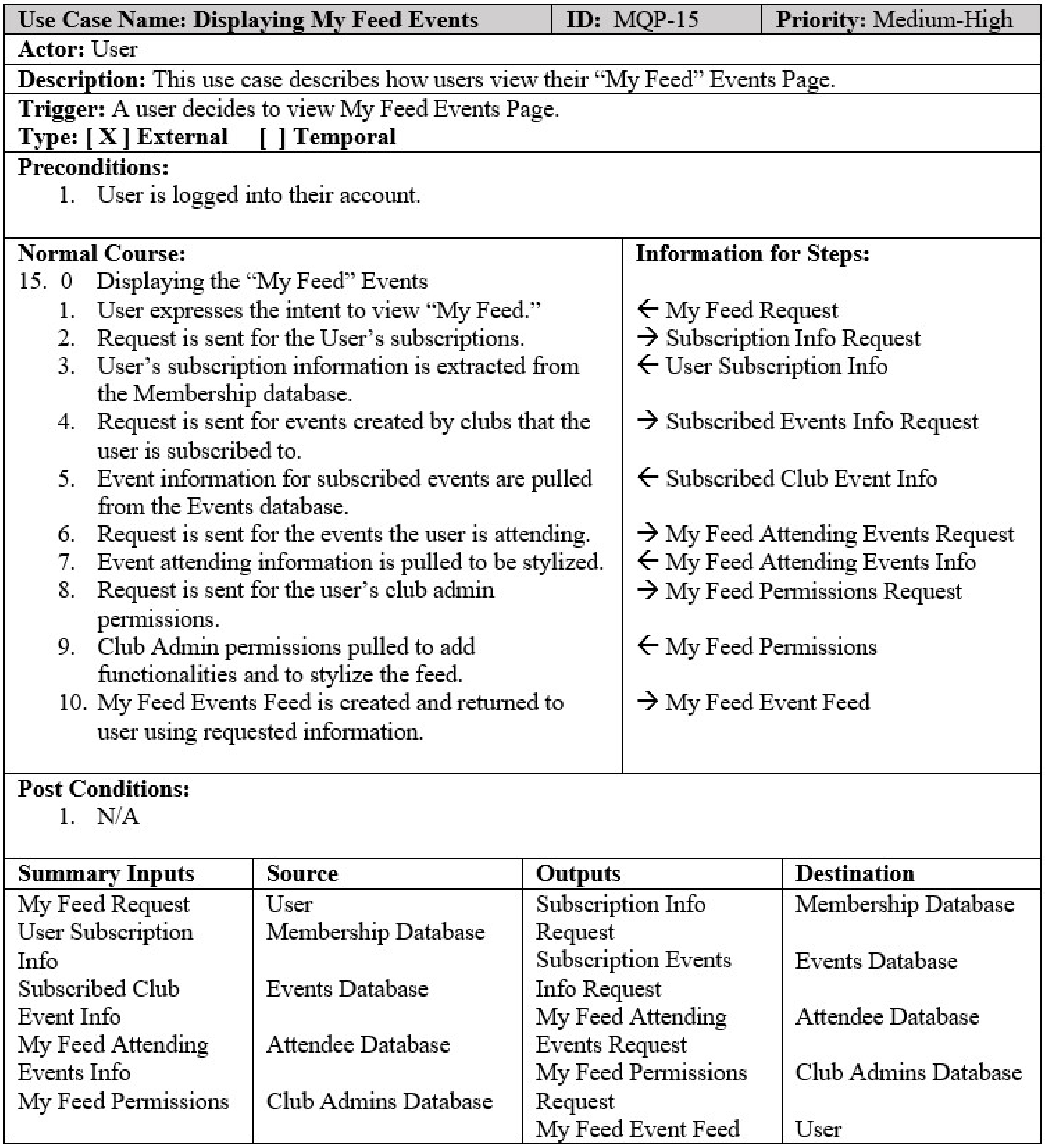
1. N/A

Summary Inputs	**Source**	**Outputs**	**Destination**
Event Page Request	User	Event Info Request	Events Database
Event Info	Events Database	Event Permissions Request	Club Admins Database
Event Permissions	Club Admins Database	Event Info Page	User

Figure 38 - Use Case #14: Displaying Club Page

4.6.4.3 Displaying My Feed

This use case describes the process a user takes to view their My Feed which consists of the user's subscribed club events, and the events they intend to attend. The user selects the "My Feed" tab on the "JoynU" page which then displays the user-specific event information.



Use Case Name: Displaying My Feed Events	**ID:** MQP-15	**Priority:** Medium-High

Actor: User

Description: This use case describes how users view their "My Feed" Events Page.

Trigger: A user decides to view My Feed Events Page.
Type: [X] External [ ] Temporal

Preconditions:
1. User is logged into their account.

Normal Course:	**Information for Steps:**
15. 0 Displaying the "My Feed" Events	
1. User expresses the intent to view "My Feed."	← My Feed Request
2. Request is sent for the User's subscriptions.	→ Subscription Info Request
3. User's subscription information is extracted from the Membership database.	← User Subscription Info
4. Request is sent for events created by clubs that the user is subscribed to.	→ Subscribed Events Info Request
5. Event information for subscribed events are pulled from the Events database.	← Subscribed Club Event Info
6. Request is sent for the events the user is attending.	→ My Feed Attending Events Request
7. Event attending information is pulled to be stylized.	← My Feed Attending Events Info
8. Request is sent for the user's club admin permissions.	→ My Feed Permissions Request
9. Club Admin permissions pulled to add functionalities and to stylize the feed.	← My Feed Permissions
10. My Feed Events Feed is created and returned to user using requested information.	→ My Feed Event Feed

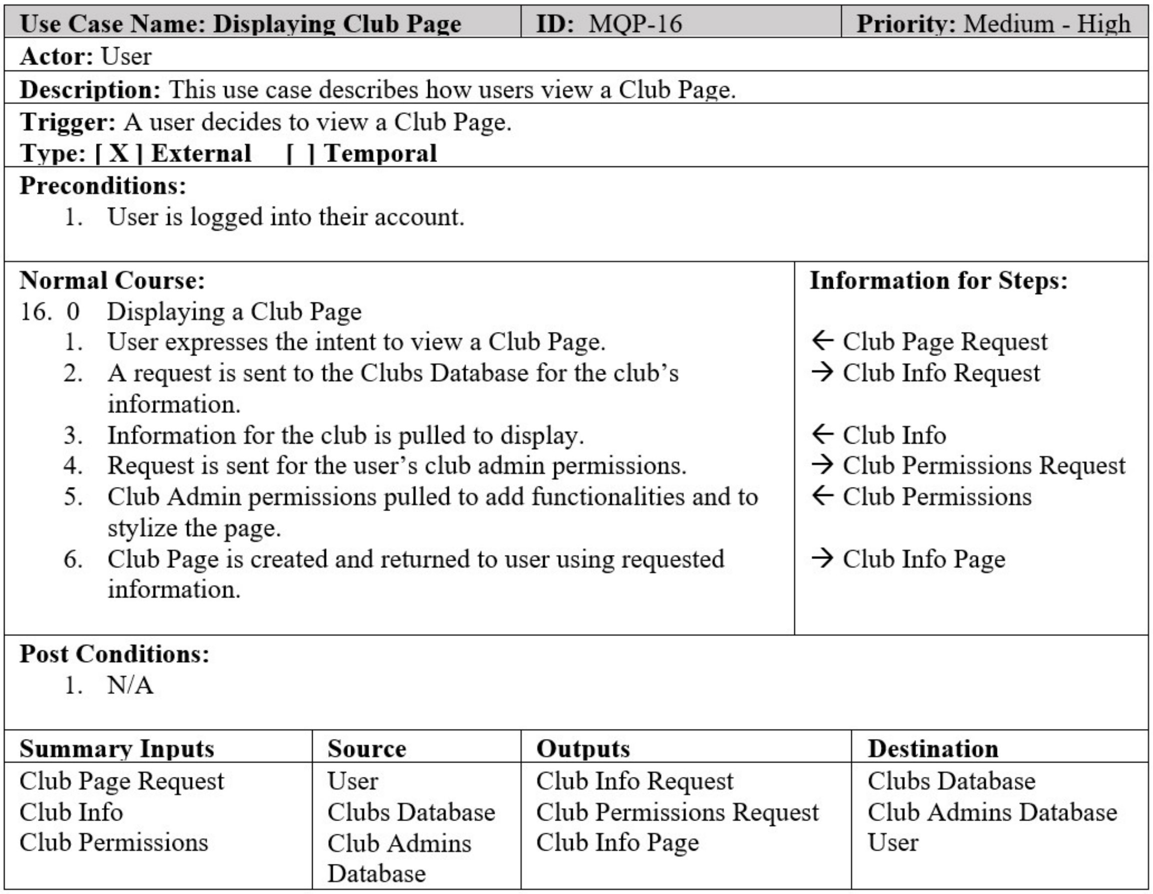
Post Conditions:
1. N/A

Summary Inputs	**Source**	**Outputs**	**Destination**
My Feed Request	User	Subscription Info Request	Membership Database
User Subscription Info	Membership Database	Subscription Events Info Request	Events Database
Subscribed Club Event Info	Events Database	My Feed Attending Events Request	Attendee Database
My Feed Attending Events Info	Attendee Database	My Feed Permissions Request	Club Admins Database
My Feed Permissions	Club Admins Database	My Feed Event Feed	User

Figure 39 - Use Case #15: Displaying My Feed Events

4.6.4.4 Displaying Club Information Page

This use case describes the process a user takes to view a club information page. Once a user expresses that they want to view a club's page, whether by pressing the club name link or search for the club, they should be taken to the club specific page. When there, the user should be able to see any events or meetings based on their membership status.



Use Case Name: Displaying Club Page	**ID:** MQP-16	**Priority:** Medium - High

Actor: User

Description: This use case describes how users view a Club Page.

Trigger: A user decides to view a Club Page.
Type: [X] External [ ] Temporal

Preconditions:

1. User is logged into their account.

Normal Course:	**Information for Steps:**
16. 0 Displaying a Club Page	
1. User expresses the intent to view a Club Page.	← Club Page Request
2. A request is sent to the Clubs Database for the club's information.	→ Club Info Request
3. Information for the club is pulled to display.	← Club Info
4. Request is sent for the user's club admin permissions.	→ Club Permissions Request
5. Club Admin permissions pulled to add functionalities and to stylize the page.	← Club Permissions
6. Club Page is created and returned to user using requested information.	→ Club Info Page

Post Conditions:

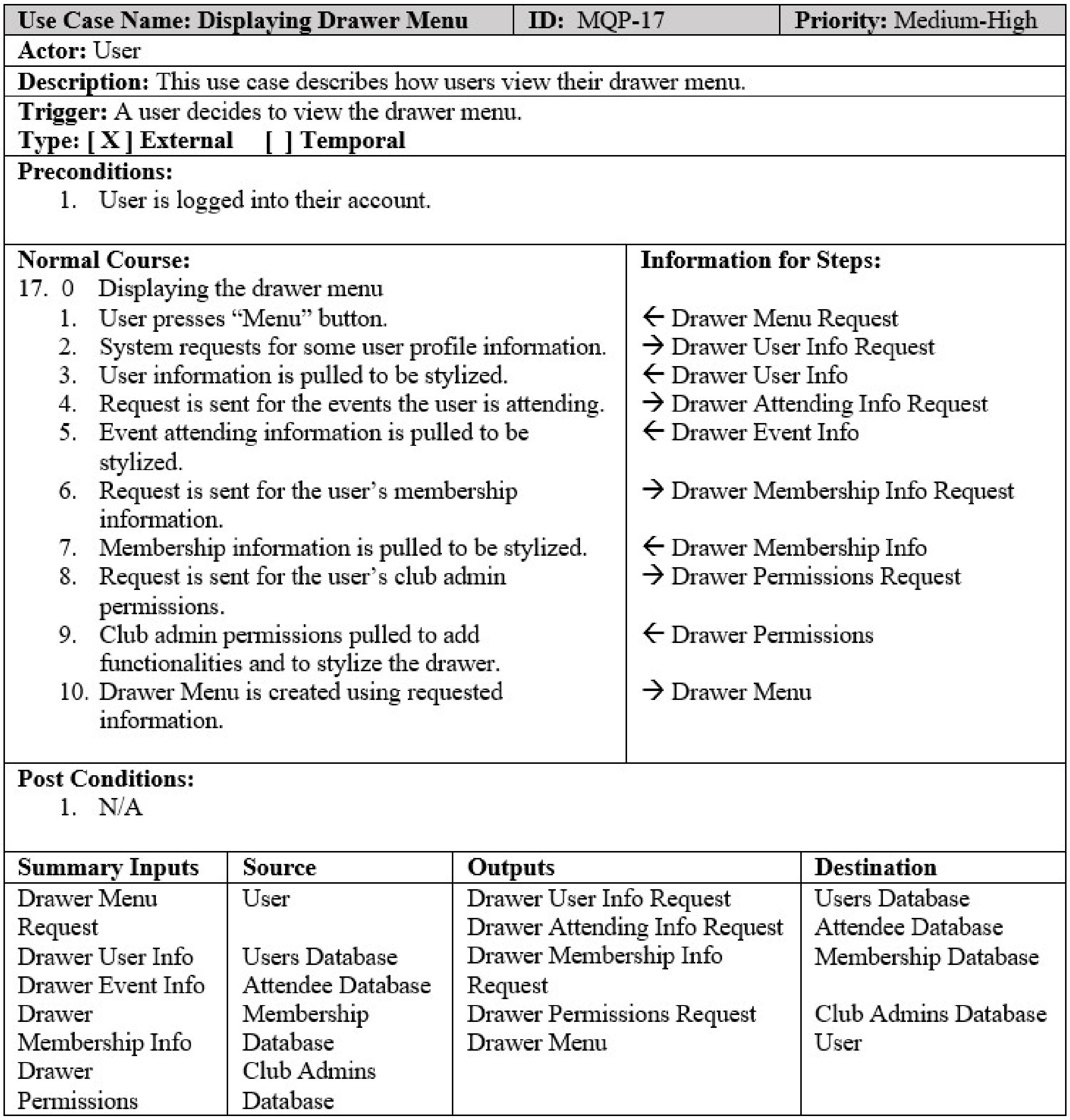
1. N/A

Summary Inputs	**Source**	**Outputs**	**Destination**
Club Page Request	User	Club Info Request	Clubs Database
Club Info	Clubs Database	Club Permissions Request	Club Admins Database
Club Permissions	Club Admins Database	Club Info Page	User

Figure 40 - Use Case #16: Displaying Club Page

4.6.4.5 Displaying Drawer Menu

The use case describes the process and information displayed to the user via the drawer menu. To access the drawer menu, users press the "Menu" button in the "JoynU" header. The drawer menu consists of information about the user's profile, memberships, attended events, club admin permissions, and additional links such as "Log Out" and "Create a Club."



<table>
<tr><td colspan="2"><b>Use Case Name: Displaying Drawer Menu</b></td><td><b>ID:</b> MQP-17</td><td><b>Priority:</b> Medium-High</td></tr>
<tr><td colspan="4"><b>Actor:</b> User</td></tr>
<tr><td colspan="4"><b>Description:</b> This use case describes how users view their drawer menu.</td></tr>
<tr><td colspan="4"><b>Trigger:</b> A user decides to view the drawer menu.
<b>Type: [X] External [ ] Temporal</b></td></tr>
<tr><td colspan="4"><b>Preconditions:</b>
1. User is logged into their account.</td></tr>
<tr><td colspan="2"><b>Normal Course:</b>
17. 0 Displaying the drawer menu
1. User presses "Menu" button.
2. System requests for some user profile information.
3. User information is pulled to be stylized.
4. Request is sent for the events the user is attending.
5. Event attending information is pulled to be stylized.
6. Request is sent for the user's membership information.
7. Membership information is pulled to be stylized.
8. Request is sent for the user's club admin permissions.
9. Club admin permissions pulled to add functionalities and to stylize the drawer.
10. Drawer Menu is created using requested information.</td><td colspan="2"><b>Information for Steps:</b>
← Drawer Menu Request
→ Drawer User Info Request
← Drawer User Info
→ Drawer Attending Info Request
← Drawer Event Info
→ Drawer Membership Info Request
← Drawer Membership Info
→ Drawer Permissions Request
← Drawer Permissions
→ Drawer Menu</td></tr>
<tr><td colspan="4"><b>Post Conditions:</b>
1. N/A</td></tr>
<tr><td><b>Summary Inputs</b></td><td><b>Source</b></td><td><b>Outputs</b></td><td><b>Destination</b></td></tr>
<tr><td>Drawer Menu Request</td><td>User</td><td>Drawer User Info Request</td><td>Users Database</td></tr>
<tr><td>Drawer User Info</td><td>Users Database</td><td>Drawer Attending Info Request</td><td>Attendee Database</td></tr>
<tr><td>Drawer Event Info</td><td>Attendee Database</td><td>Drawer Membership Info Request</td><td>Membership Database</td></tr>
<tr><td>Drawer Membership Info</td><td>Membership Database</td><td>Drawer Permissions Request</td><td>Club Admins Database</td></tr>
<tr><td>Drawer Permissions</td><td>Club Admins Database</td><td>Drawer Menu</td><td>User</td></tr>
</table>

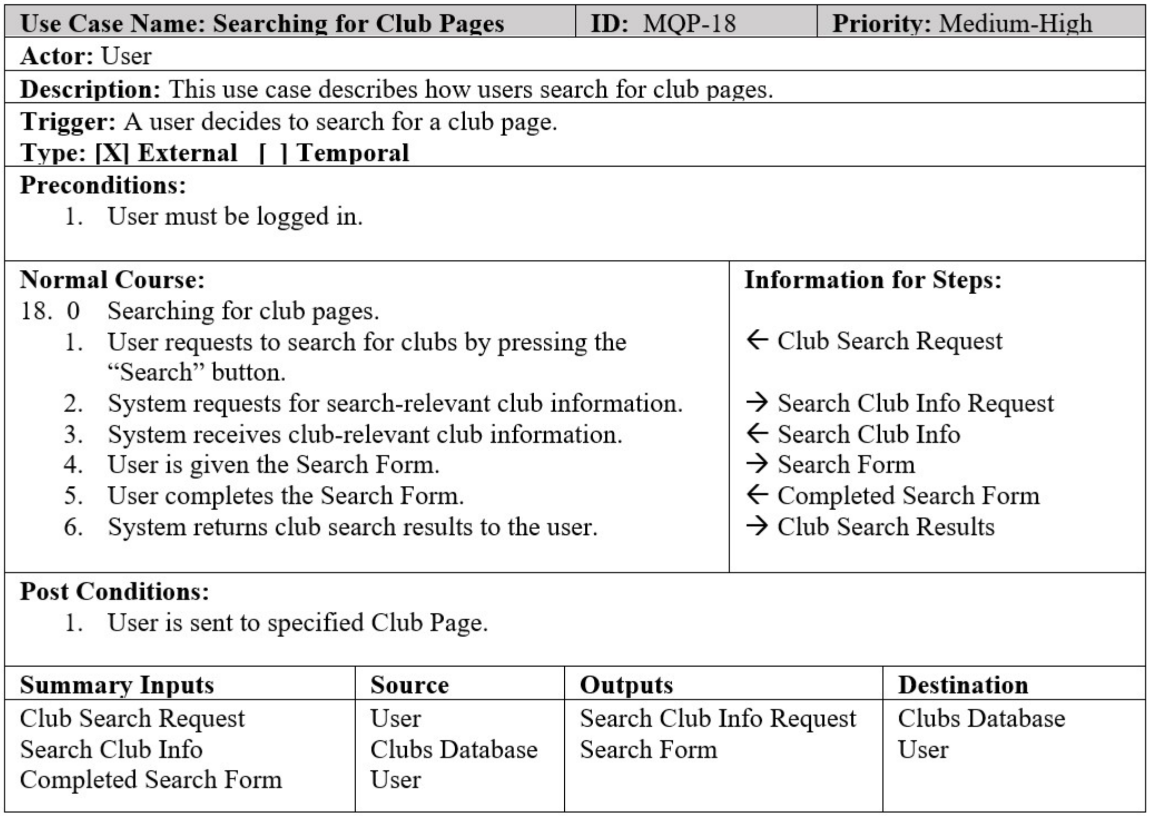
Figure 41 - Use Case #17: Displaying the Drawer Menu

4.6.5 Searching/Filtering Information

The use cases in this section focus on how the user can filter and search the existing information in the application to better find what they are looking for. The use cases covered include: Searching for Club Pages and Filtering Events by Type.

4.6.5.1 Searching for Club Pages

This use case describes the process the user would take to search for a Club Page. The user would press the “Search” button in the top right corner of the application where they would be able to fill out a search request for the club of their choosing. Afterwards, they will be brought to that club’s respective club page.



Use Case Name: Searching for Club Pages	**ID:** MQP-18	**Priority:** Medium-High

Actor: User

Description: This use case describes how users search for club pages.

Trigger: A user decides to search for a club page.
Type: [X] External [ ] Temporal

Preconditions:
1. User must be logged in.

Normal Course:	**Information for Steps:**
18. 0 Searching for club pages.	
1. User requests to search for clubs by pressing the “Search” button.	← Club Search Request
2. System requests for search-relevant club information.	→ Search Club Info Request
3. System receives club-relevant club information.	← Search Club Info
4. User is given the Search Form.	→ Search Form
5. User completes the Search Form.	← Completed Search Form
6. System returns club search results to the user.	→ Club Search Results

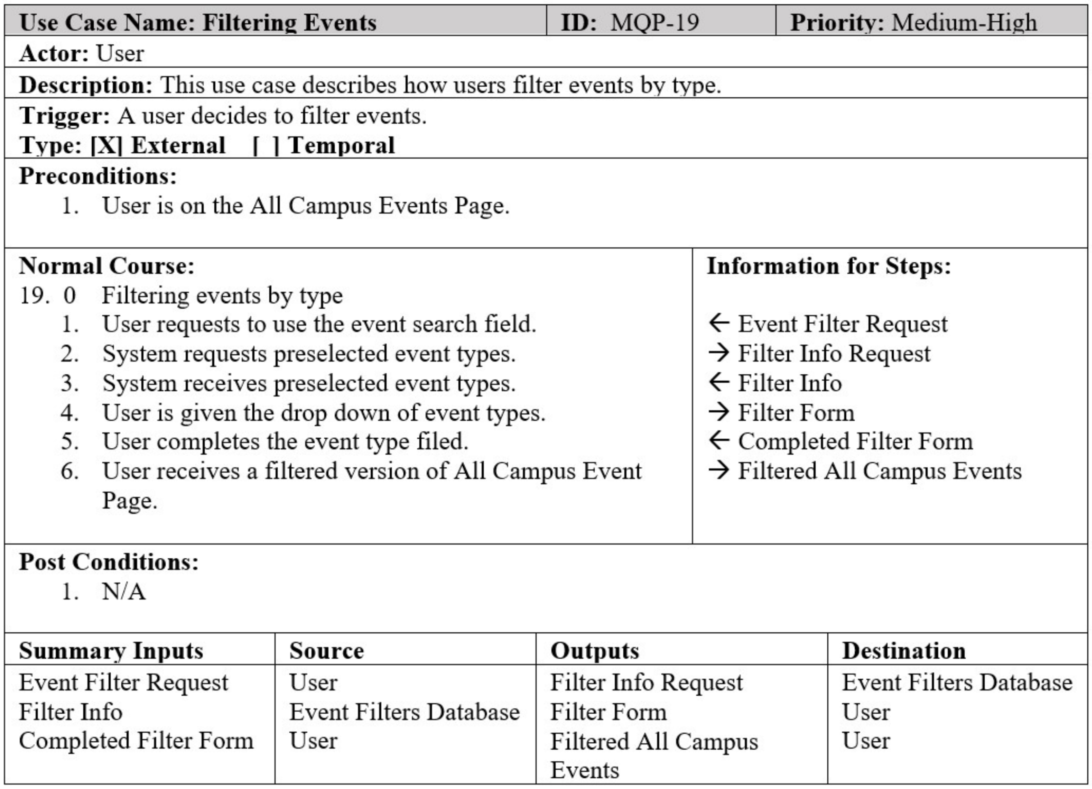
Post Conditions:
1. User is sent to specified Club Page.

Summary Inputs	**Source**	**Outputs**	**Destination**
Club Search Request	User	Search Club Info Request	Clubs Database
Search Club Info	Clubs Database	Search Form	User
Completed Search Form	User		

Figure 42 - Use Case #18: Searching for Club Pages

4.6.5.2 Filtering by Event Type

This use case describes the process the user would take to filter events on the All Campus Events Feed. The user would press the “Filter” field on the top of the All Campus Event Feed section where they would be able to determine which types of events they would like to display. Afterwards, they will be given an updated feed of campus events that are of that specified type.



Use Case Name: Filtering Events	ID: MQP-19	Priority: Medium-High

Actor: User

Description: This use case describes how users filter events by type.

Trigger: A user decides to filter events.
Type: [X] External [ ] Temporal

Preconditions:
1. User is on the All Campus Events Page.

Normal Course:	Information for Steps:
19. 0 Filtering events by type	
1. User requests to use the event search field.	← Event Filter Request
2. System requests preselected event types.	→ Filter Info Request
3. System receives preselected event types.	← Filter Info
4. User is given the drop down of event types.	→ Filter Form
5. User completes the event type filed.	← Completed Filter Form
6. User receives a filtered version of All Campus Event Page.	→ Filtered All Campus Events

Post Conditions:
1. N/A

Summary Inputs	Source	Outputs	Destination
Event Filter Request	User	Filter Info Request	Event Filters Database
Filter Info	Event Filters Database	Filter Form	User
Completed Filter Form	User	Filtered All Campus Events	User

Figure 43 - Use Case #19: Filtering Events

5. Design

5.1 Database Design

The first iteration of the application (developed in A-Term) was based on the Entity Relationship Diagram shown below. To simplify our development, we focused on only two entities: Users and Events. Each entity contained several attributes, and two notable relationships existed; those relationships consisted of user creating events, and users attending events.

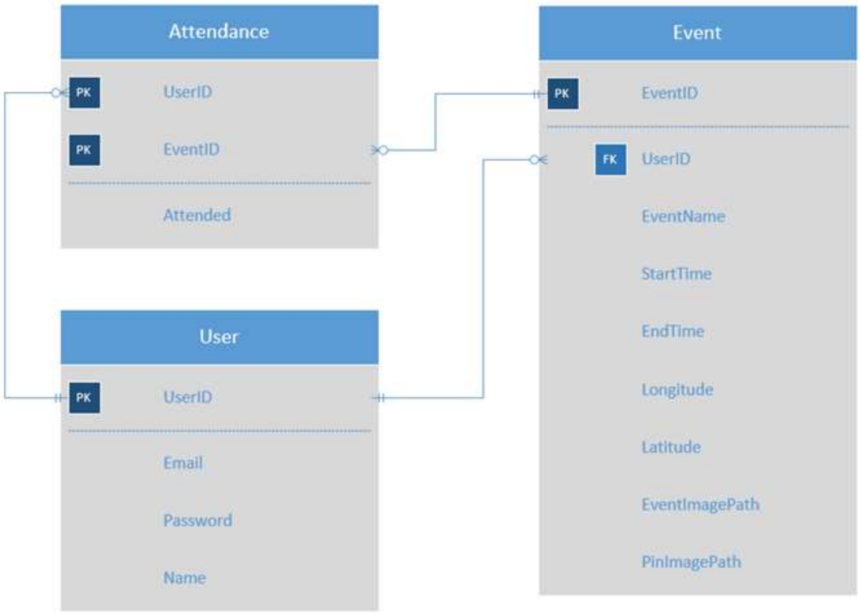


Figure 44 - Application Entity-Relationship Diagram (A-Term)

In our second application's iteration (completed in C-Term), we added Club and School entities as well as Location and EventType attribute tables to properly depict the relationships and specificity of details present in the application. The new relationships that exist include: Users being Members of Clubs, Clubs creating Events, Clubs belonging to Schools, Events taking place at Locations, and Events being of a type.

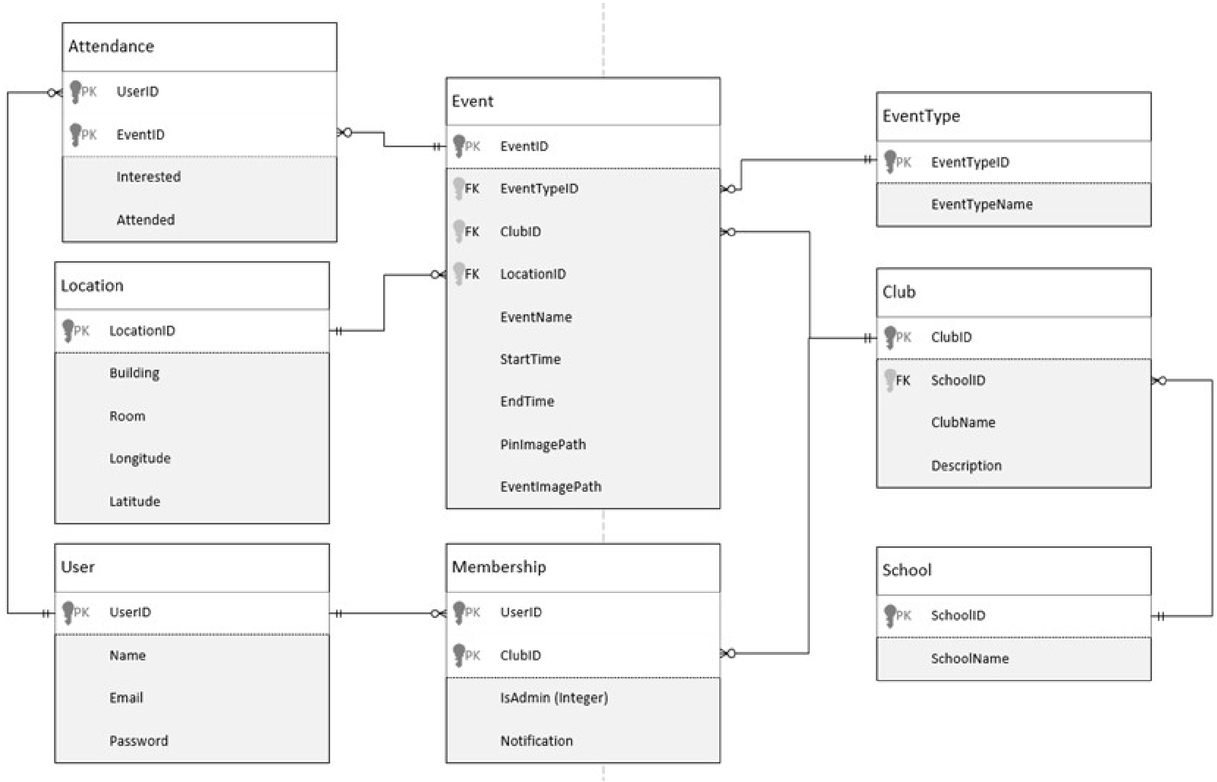


Figure 45 - Application Entity-Relationship Diagram (C-Term)

5.2 Data Flow Design

Figure 46 is our final context data flow diagram of application completed thus far. The diagram displays context for the overall Application, where we see the breakdown of where information is received and forwarded among General Users and Club Admins. Furthermore, we provide additional levels of how information flows for each specific functionality within the application.

5.2.1 Context DFD:

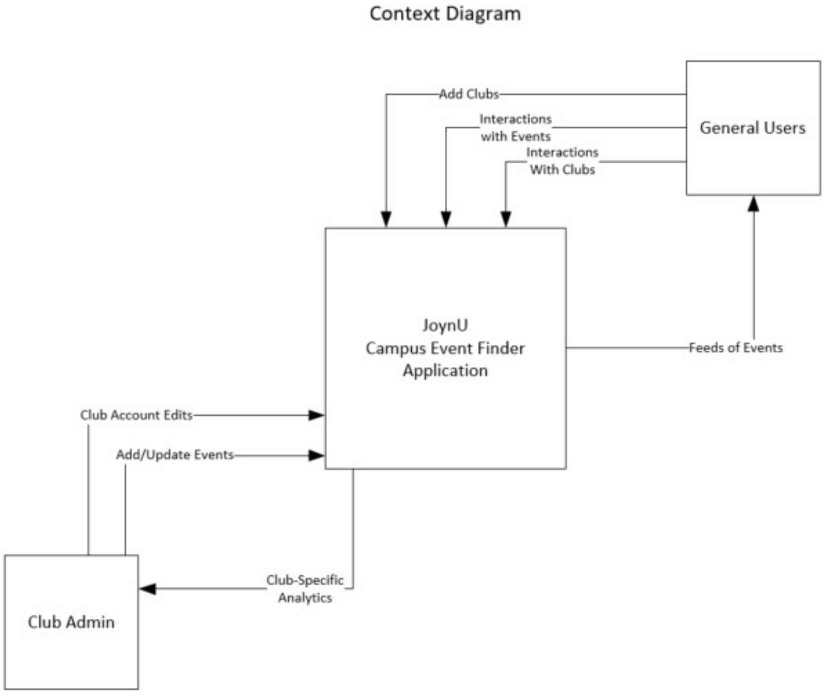


Figure 46 - Application Context Diagram

5.2.2 Level 0 DFD

To make it easier to comprehend the processes and how they are related, we first broke up our data flow into 5 overarching process, similarly to how we broke up our use cases. The processes include: Managing User Account, Interacting with Clubs, Interacting with Events, Displaying Information, Searching/Filtering Information.

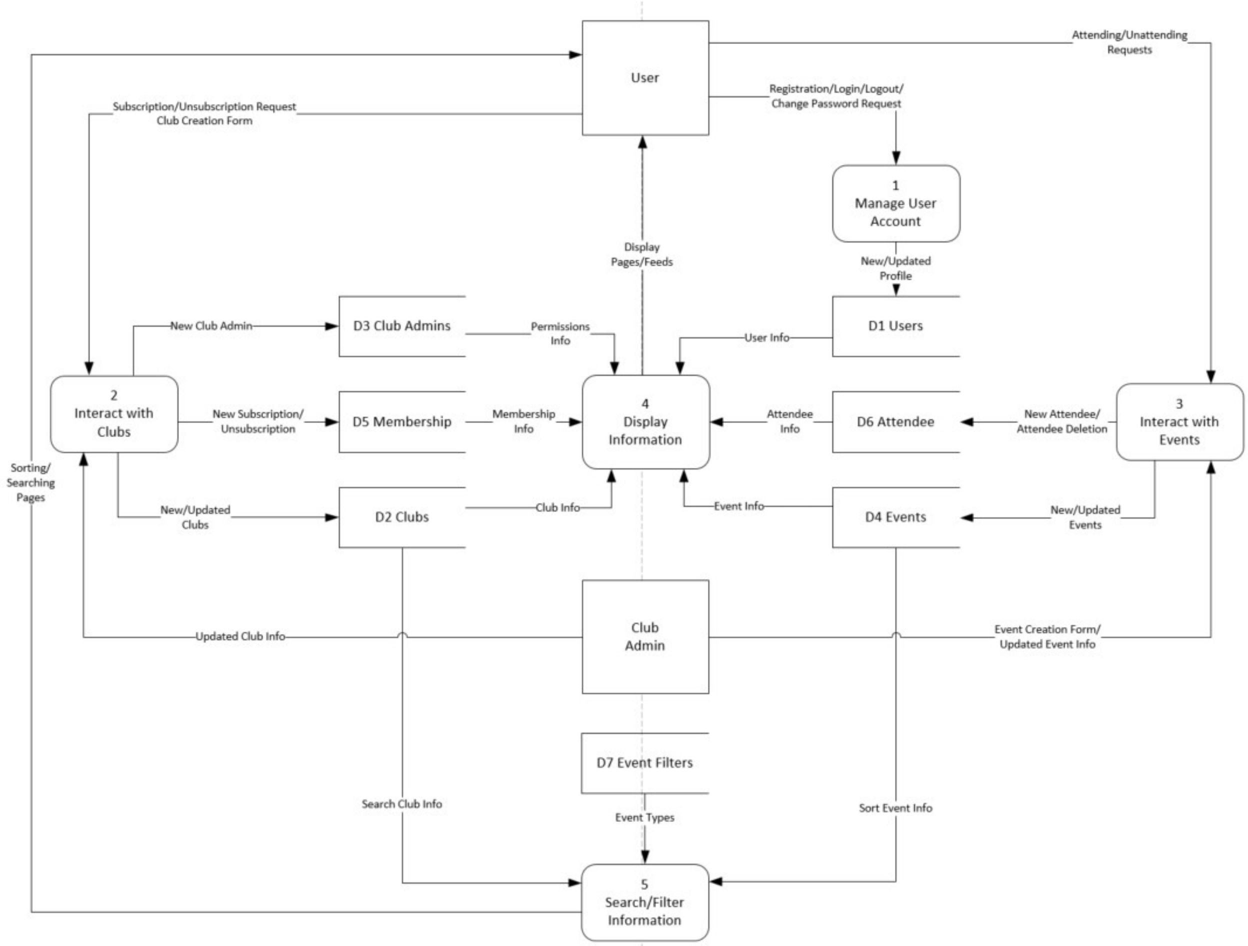


Figure 47 - Level 0 Diagram (in Landscape)

5.2.3 Level 1 Process 1: Managing User Account

This process outlines how data flows among the Managing User Account use cases which include: Creating a User Account, Logging into Account, Changing Account Password, and Logging Out.

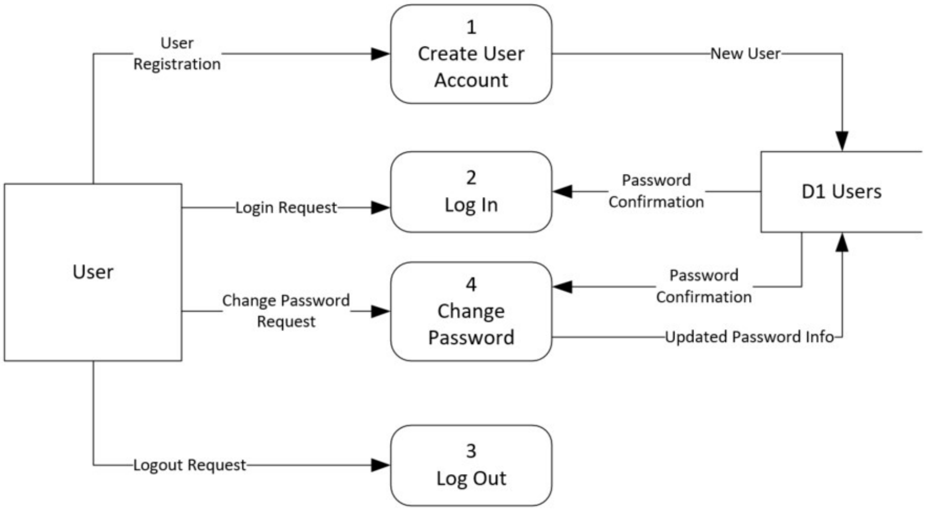


Figure 48 - Level 1: Managing User Account

5.2.3.1 Level 2 Process 1 - Creating User Account

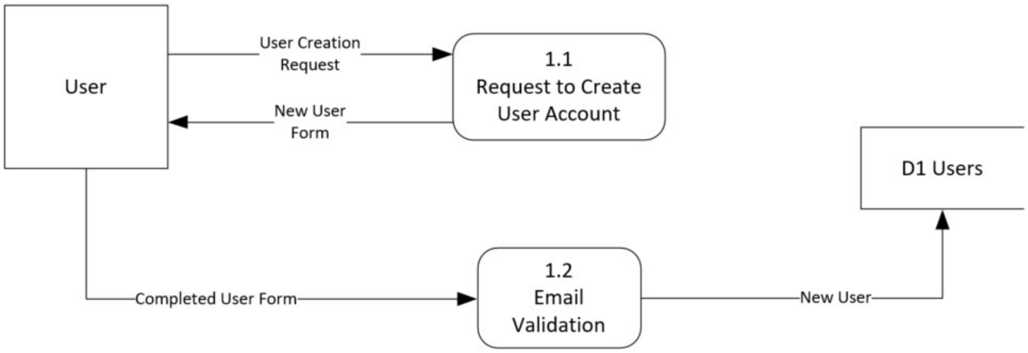


Figure 49 - Level 2 Process 1: Creating User Account

- **User Account Creation Request** – Pressing “Sign Up” on initial “Login” screen.
- **New User Form** – Consists of input fields for: *Profile Name, Profile Email, Profile Password, Profile Password Confirmation, Profile Picture*.
- **Completed User Form** – New User Form submitted with the “Sign Up!” button.
- **New User Account** – New User Account created using given user information: *Profile Name, Profile Email, Profile Password, Profile Password Confirmation, Profile Picture*.

5.2.3.2 Level 2 Process 2 - User Logging In

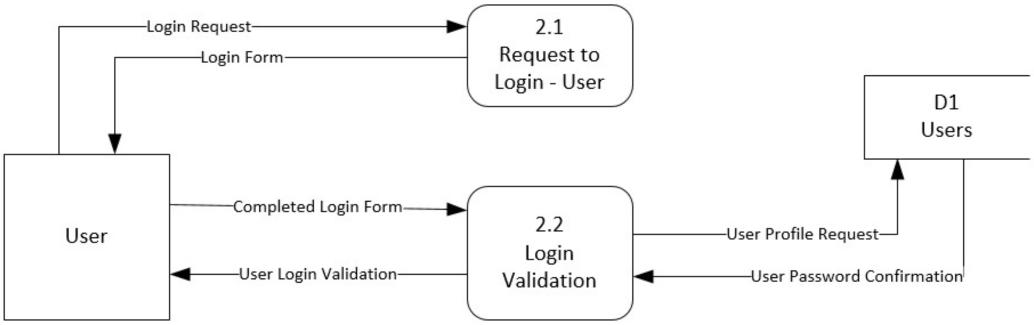


Figure 50 - Level 2 Process 2: Logging into Account

- **Login Request** – Pressing "Login" on the initial "Login" screen.
- **Login Form** – Consists of input fields for: *Login Email, Login Password.*
- **Completed Login Form** – Login Form submitted with the "Login!" button.
- **User Profile Request** – Accesses user information for the given *Login Email*.
- **User Password Confirmation** – Notification on whether the *Login Password* is a match to the intended *Profile Password*.
- **User Login Validation** – Notification to user about the success of logging in.

5.2.3.3 Level 2 Process 3 - Logging Out of Account

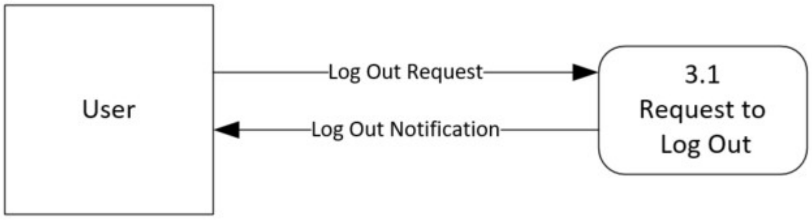


Figure 51 - Level 2 Process 3: Logging Out of Account

- **Log Out Request –** Pressing "Log Out" field in the Drawer Menu.
- **Log Out Notification –** Notification to the user about the success of them logging out.

5.2.3.4 Level 2 Process 4 – Changing Account Password

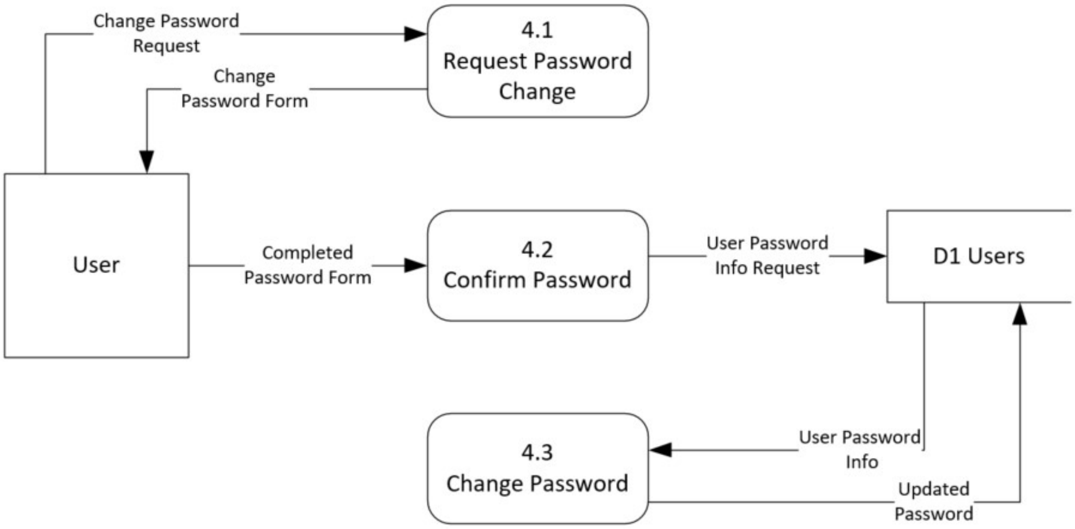


Figure 52 - Level 2 Process 4: Changing Account Password

- **Change Password Request** – User clicks the "Change Password" link
- **Change Password Form** – form with two fields given to the user: *Current Password, New Password.*
- **Completed Password Form –** User submission of the Change Password Form with completed fields.
- **User Password Info Request –** System request for the user's current password.
- **User Password Info –** The user's current password so that it can be compared to the password submitted in the field.
- **Updated Password –** *New Password* submitted by the user to become the user's new password.

5.2.4 Level 1 Process 2: Interacting with Clubs

This process outlines how data flows among the Interacting with Clubs use cases which include: Subscribing to Club, Unsubscribing to Club, Creating a club, and Editing a Club.

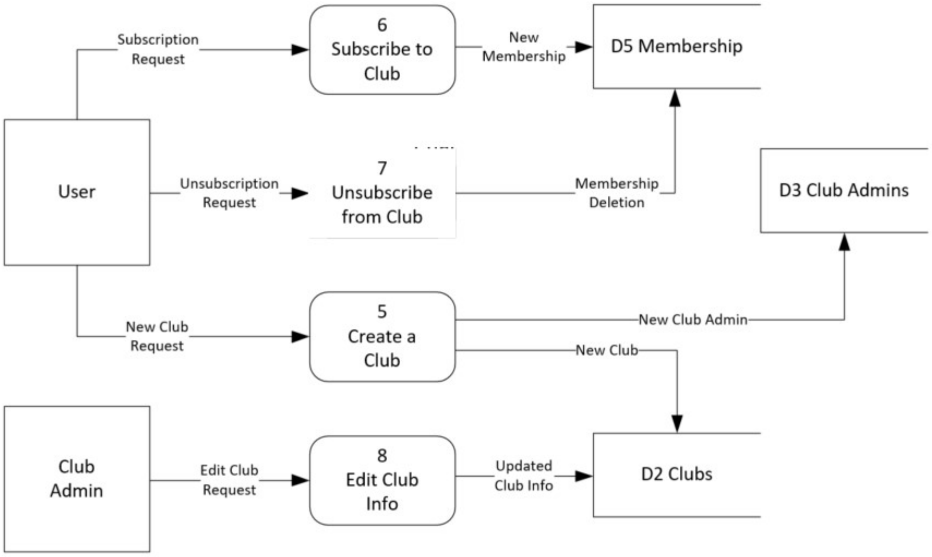


Figure 53 - Level 1 Process 2: Interacting with Clubs

5.2.4.1 Level 2 Process 5 - Creating Club Account

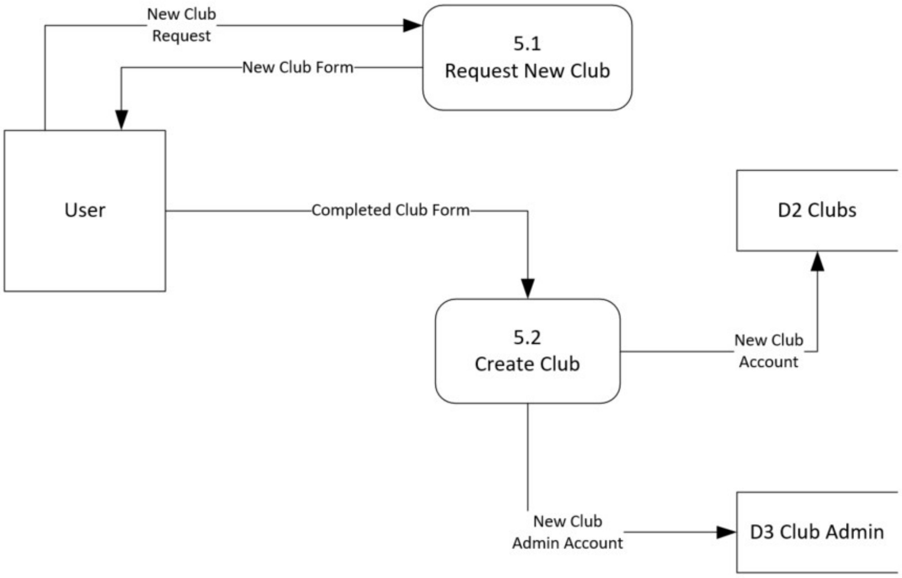


Figure 54 - Level 2 Process 5: Creating Club Account

- **New Club Request** – Pressing “Create A Club” link in Drawer Menu Settings.
- **New Club Form** – Consists of fields for: *Club Name, Club Description, Club Picture*.
- **Completed Club Form** – New Club Form submitted with the “Create Club” button.
- **New Club Admin Account** – Gives submitter of the New Club Form *Club Admin* permissions.
- **New Club Account** – New Club Account created using given club information: *Club Name, Club Description, Club Picture*.

5.2.4.2 Level 2 Process 6 - Subscribing to Club

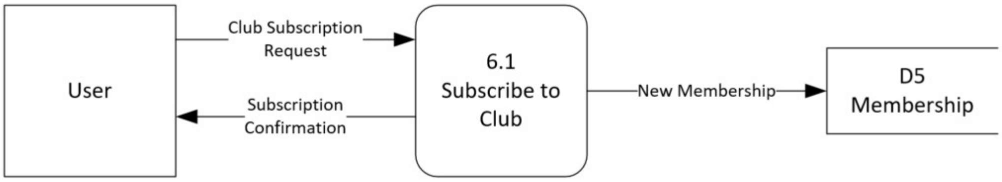


Figure 55- Level 2 Process 6: Subscribing to Club

- **Club Subscription Request** – Pressing "Subscribe" button on a Club Page**.**
- **New Membership Account –** Records that the user is now a subscriber to the account.
- **Subscription Confirmation** – Notification to user that their subscription was recorded.

5.2.4.3 Level 2 Process 7 - Unsubscribing from Club

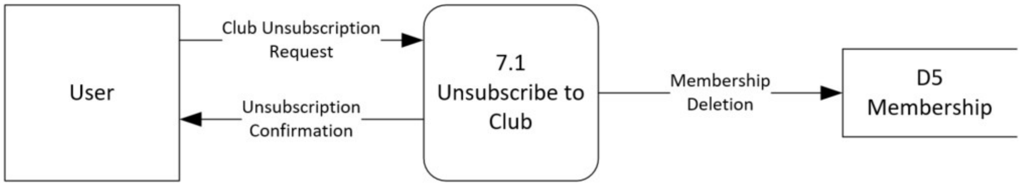


Figure 56 - Level 2 Process 7: Unsubscribing to Club

- **Club Unsubscription Request** – Pressing "Unsubscribe" button on a Club Page**.**
- **Membership Deletion –** Records that the user is no longer a subscriber to the account.
- **Unsubscription Confirmation** – Notification to user that their unsubscription was recorded.

5.2.4.4 Level 2 Process 8 – Editing Club Information

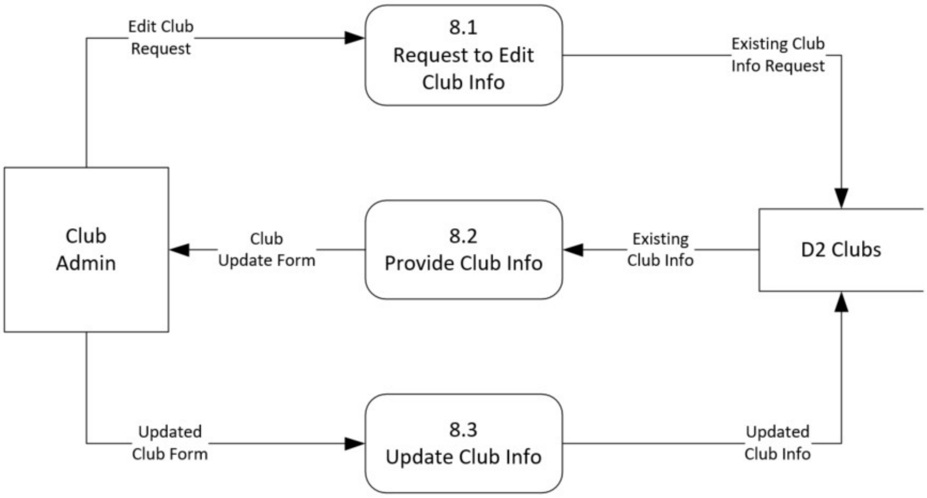


Figure 57 - Level 2 Process 8: Editing Club Information

- **Edit Club Request** – Pressing "Edit" button on the club page.
- **Existing Club Info Request** – System request to pull information about the club.
- **Club Update Form** – Form containing fields to update the club's information. Fields include: *Club Name, Club Description, Club Picture*.
- **Updated Club Form** – A completed and submitted Club Update Form.
- **Updated Club Info** – Club Account is updated using the information provided from the Updated Club Form.

5.2.5 Level 1 Process 3: Interacting with Events

This process outlines how data flows among the Interacting with Events use cases which include: Creating an Event, Editing an Event, Attending an Event, and Unattending an Event.

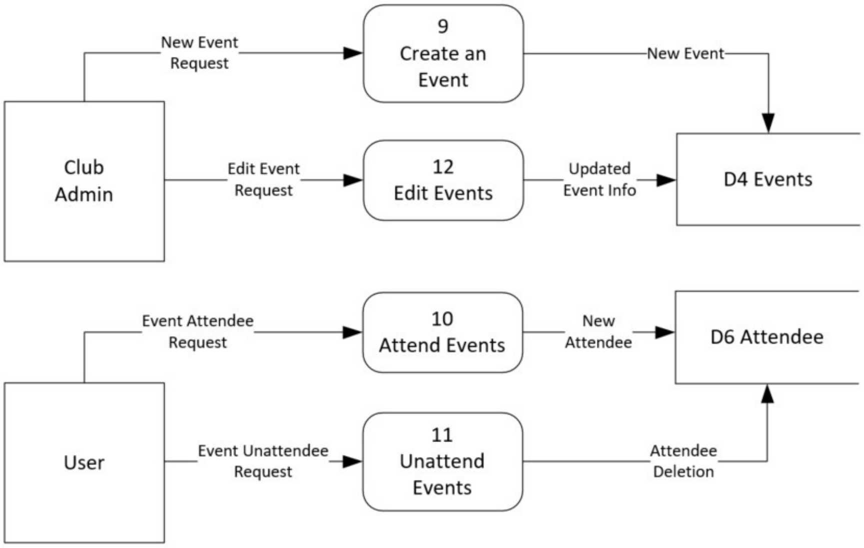


Figure 58 - Level 1 Process 3: Interacting with Events:

5.2.5.1 Level 2 Process 9 - Creating an Event

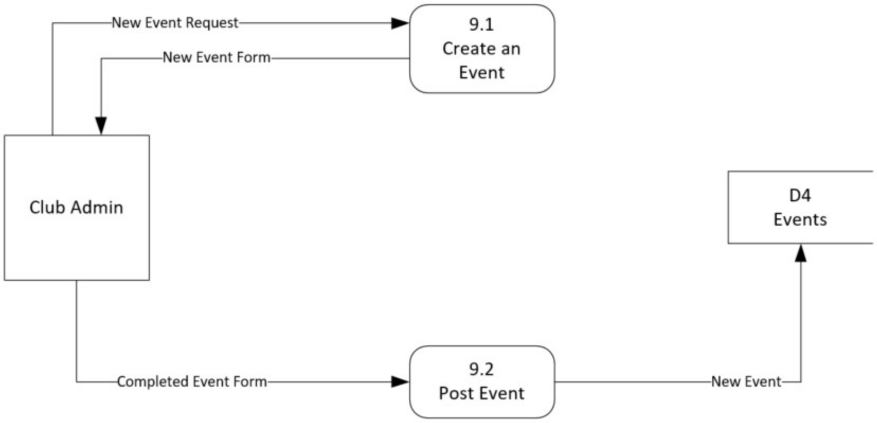


Figure 59 - Level 2 Process 9: Creating an Event

- **New Event Request** – Pressing “Create an Event” link in the Drawer Menu to add an event.
- **New Event Form** – Consists of fields for: *Event Name, Sponsoring Club, Event Location, Event Audience, Event Date, Event Start Time, Event End Time, Event Description, Event Picture.*
- **Completed Event Form –** New Event Form submitted with the “Create Event” button.
- **New Event –** New Event is created using given information: *Event Name, Sponsoring Club, Event Location, Event Audience, Event Date, Event Start Time, Event End Time, Event Description, Event Picture.*

5.2.5.2 Level 2 Process 10 - Attending an Event

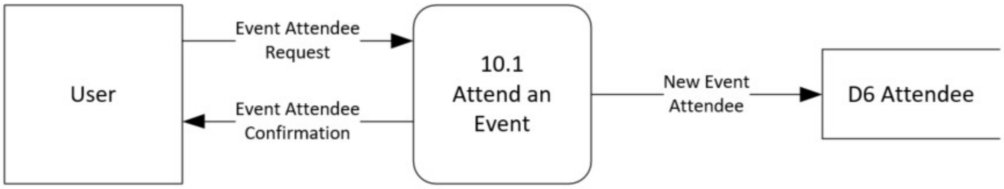


Figure 60 - Level 2 Process 10: Attending an Event

- **Event Attendee Request –** Pressing “J” button on an event card.
- **New Event Attendee Account –** New Event Interest Account is made consisting of: *Event ID, User ID.*
- **Event Attendee Confirmation** – Notification to user that their attendance was recorded.

5.2.5.3 Level 2 Process 11 - Unattending an Event

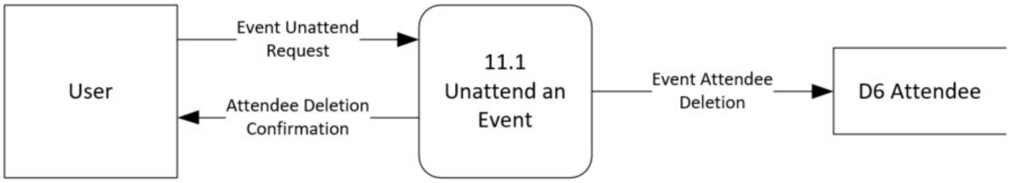


Figure 61 - Level 2 Process 11: Unattending an Event

- **Event Unattendee Request –** Pressing “U” button on an event card.
- **Event Attendee Deletion –** The user’s Event Attendee status is removed.
- **Attendee Deletion Confirmation** – Notification to user that their unsubscription was recorded.

5.2.5.4 Level 2 Process 12 – Editing Event Information

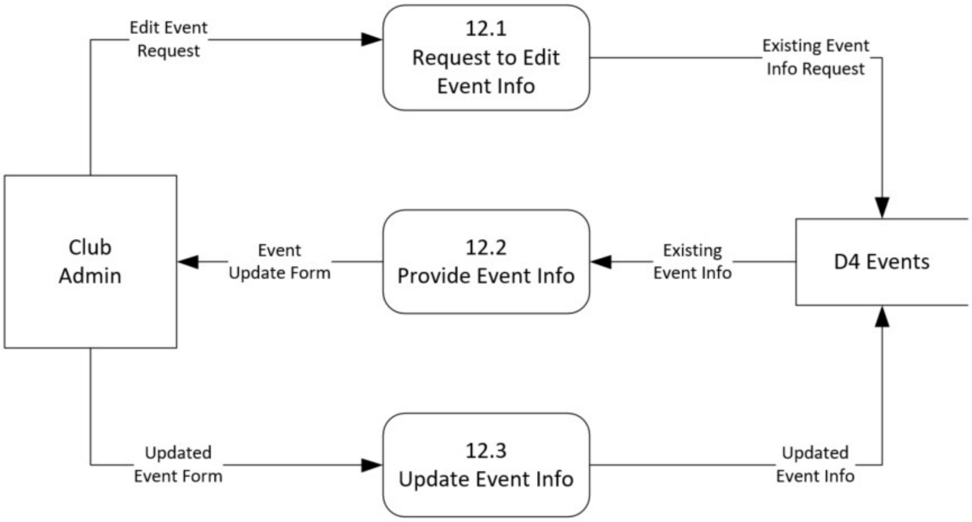


Figure 62 - Level 2 Process 12: Editing Event Information

- **Edit Event Request** – Pressing "…" button below an event.
- **Existing Event Info Request** – System request to pull information about the event.
- **Event Update Form** – Form containing fields to update the club's information. Fields include: *Event Name, Sponsoring Club, Event Location, Event Audience, Event Date, Event Start Time, Event End Time, Event Description, Event Picture.*
- **Updated Event Form** – A completed and submitted Event Update Form.
- **Updated Event Info** – Event Info is updated using the information provided from the Updated Event Form.

5.2.6 Level 1 Process 4: Displaying App Information

This process outlines how data flows among the Interacting with Events use cases which include: Displaying All Campus Events, Displaying Event Pages, Displaying My Feed Events, Displaying Club Pages, and Displaying the Drawer Menu.

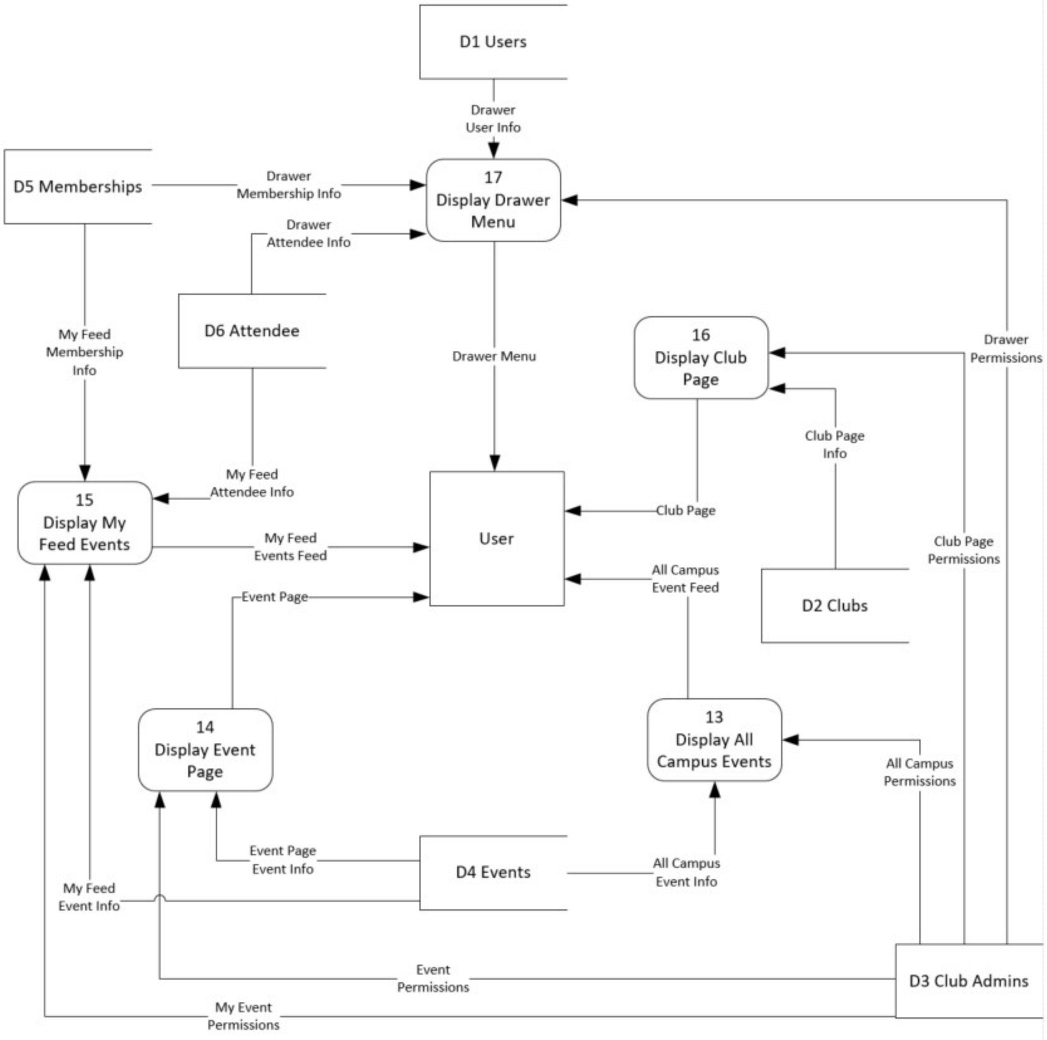


Figure 63 - Level 1 Process 4: Displaying App Information

5.2.6.1 Level 2 Process 13 - Displaying All Campus Events Page

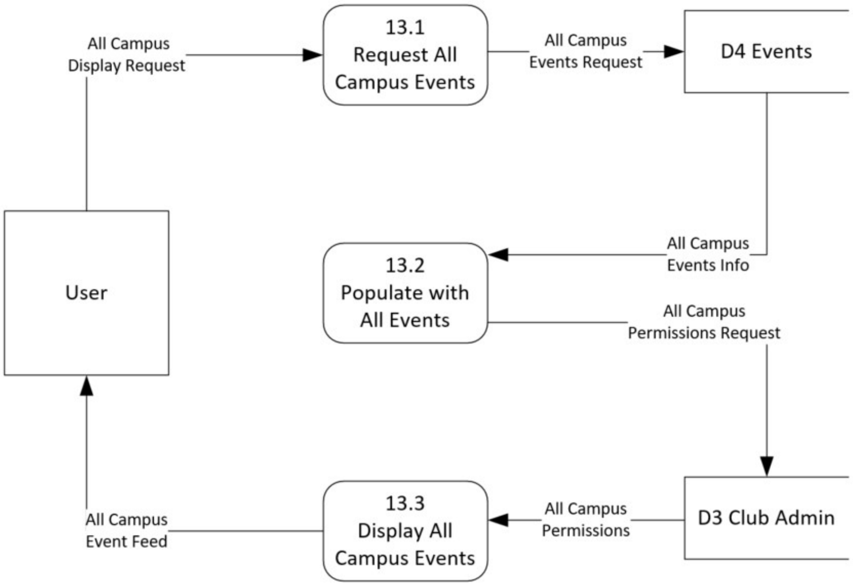

All Campus Display Request
13.1 Request All Campus Events
All Campus Events Request
D4 Events
User
All Campus Events Info
13.2 Populate with All Events
All Campus Permissions Request
All Campus Event Feed
13.3 Display All Campus Events
All Campus Permissions
D3 Club Admin

Figure 64 - Level 2 Process 13: Displaying All Campus Events Page

- **All Campus Display Request –** Pressing “All Campus” tab.
- **All Campus Events Request –** System request for Events to populate “All Campus” Page.
- **All Campus Events Info** – Data for events to be displayed in All Campus.
- **All Campus Permissions Request** – Request to determine for what events the user is a club admin of.
- **All Campus Permissions** – Information of which events the user is a club admin of.
- **All Campus Event Feed**– Graphical display of the All Campus Events List.

5.2.6.2 Level 2 Process 14 - Displaying Event Information Page

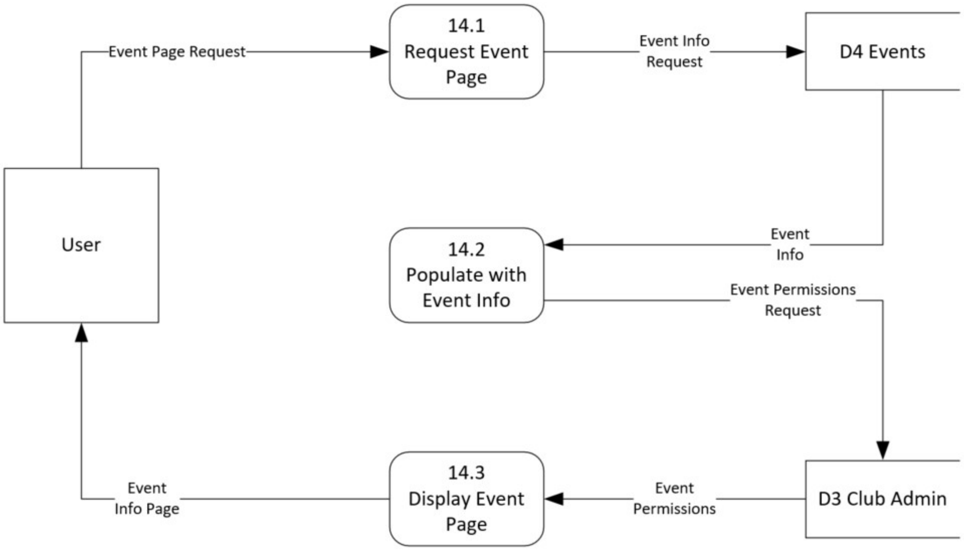


Figure 65 - Level 2 Process 14: Displaying Event Information Page

- **Event Page Request –** Pressing "Event Name" field in feeds or using search bar to discover more information about events.
- **Event Info Request** - System request for an event's information.
- **Event Info –** Data for the specified event.
- **Event Permissions Request** – Request to determine for what events the user is a club admin of.
- **Event Permissions** – Information of which events the user is a club admin of.
- **Event Info Page –** Graphical display of the event's information.

5.2.6.3 Level 2 Process 15 - Displaying My Feed Event Page

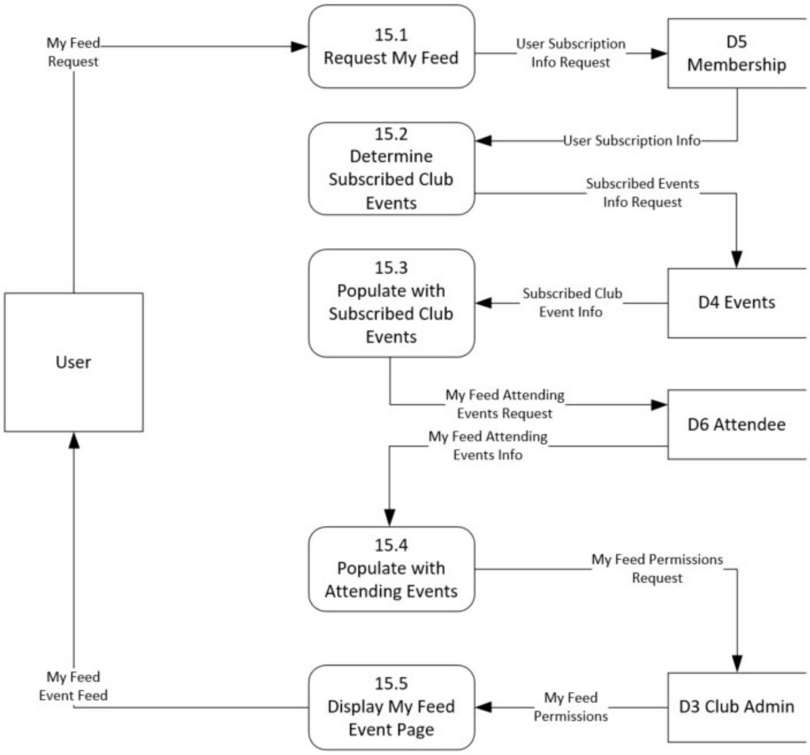

My Feed Request
15.1
Request My Feed
User Subscription Info Request
D5 Membership
User Subscription Info
15.2
Determine Subscribed Club Events
Subscribed Events Info Request
15.3
Populate with Subscribed Club Events
Subscribed Club Event Info
D4 Events
User
My Feed Attending Events Request
My Feed Attending Events Info
D6 Attendee
15.4
Populate with Attending Events
My Feed Permissions Request
My Feed Event Feed
15.5
Display My Feed Event Page
My Feed Permissions
D3 Club Admin

Figure 66- Level 2 Process 15: Displaying My Feed Event Page

- **My Feed Request –** Pressing “My Feed” tab on “JoynU” Page.
- **User Subscription Info Request –** System request for user’s membership information.
- **User Subscription Info –** Data of the user’s memberships.
- **Subscribed Events Info Request –** System request for event information the user is subscribed to.
- **Subscribed Club Event Info –** Data of the events sponsored by subscribed clubs.
- **My Feed Attending Events Request –** Request for data for events user is attending that will be displayed in My Feed.
- **My Feed Attending Events Info -** Data for events user is attending that will be displayed in My Feed.
- **My Feed Permissions Request** – Request to determine for what events the user is a club admin of.
- **My Feed Permissions** – Information of which events the user is a club admin of.
- **My Feed Event Page –** Graphical display of the My Feed Event Page.

5.2.6.4 Level 2 Process 16 - Displaying Club Information Page

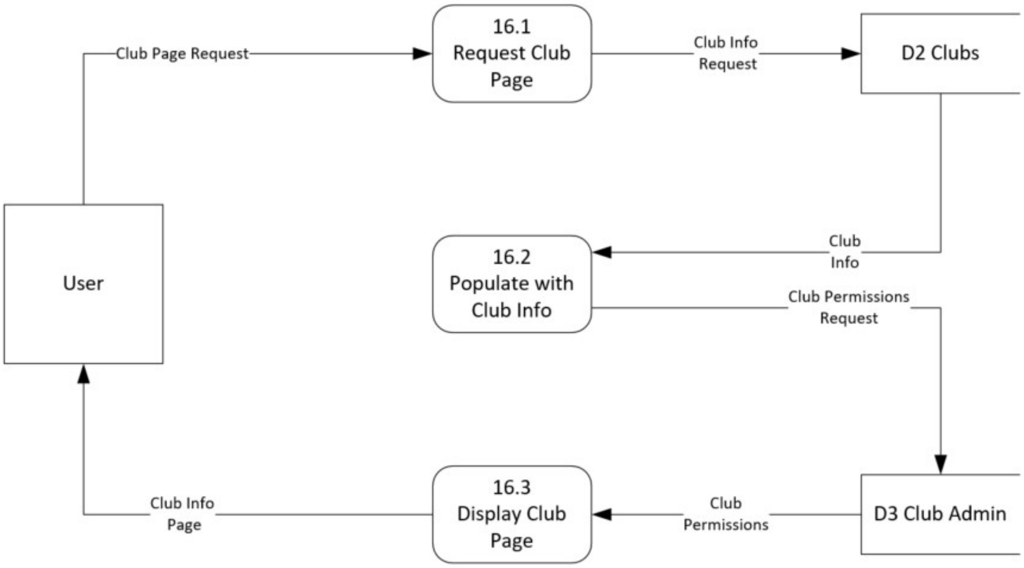


Figure 67- Level 2 Process 16: Displaying Club Information Page

- **Club Page Request -** Using the search bar to discover more information about clubs.
- **Club Info Request –** System request for a club's information.
- **Club Info –** Data for the specified club.
- **Club Permissions Request** – Request to determine if the user is the club admin of the selected club.
- **Club Permissions** – Information of which events the user is a club admin of.
- **Club Info Page –** Graphical display of the club's information.

5.2.6.5 Level 2 Process 17 - Displaying Drawer Menu

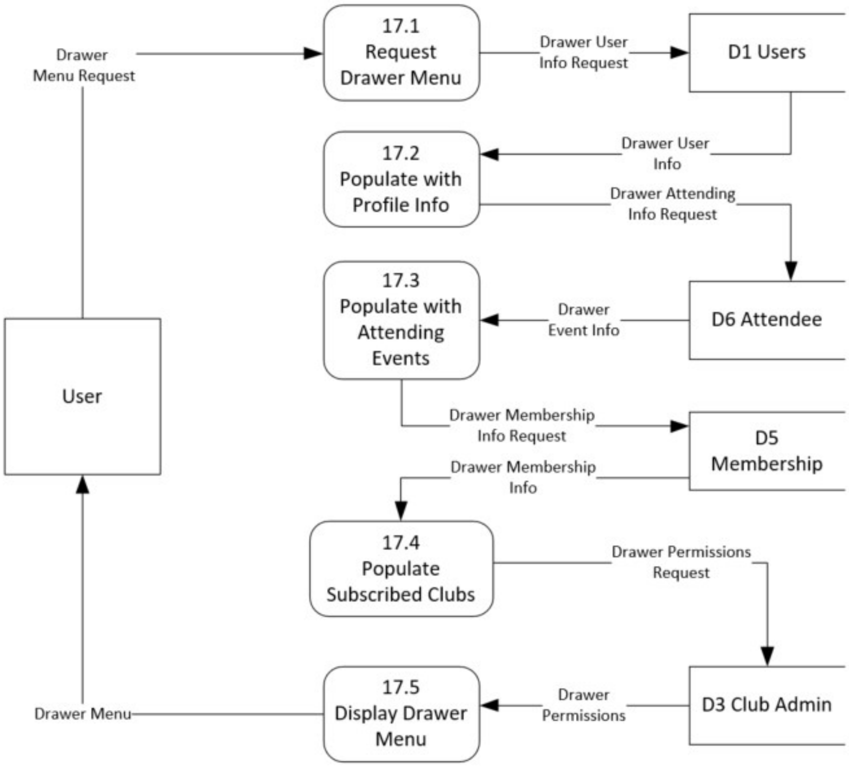


Figure 68 - Level 2 Process 17: Displaying Drawer Menu

- **Drawer Menu Request –** Pressing “Menu” button in “JoynU” header bar.
- **Drawer User Information Request –** System request for drawer-specific user information.
- **Drawer Event Information Request –** System request for drawer-specific event information.
- **Drawer Membership Information Request –** System request for drawer-specific membership information.
- **Drawer User Information –** Data of requested user information for drawer.
- **Drawer Event Information –** Data of requested event information for drawer.
- **Drawer Membership Information –** Data of requested membership information for drawer.
- **Drawer Permissions Request –** System request for user’s club admin permissions.
- **Drawer Permissions –** Data of requested club admin permissions.
- **Drawer Menu –** Graphical display of elements consisting of the Drawer Menu.

5.2.7 Level 1 Process 5: Displaying App Information

This process outlines how data flows among the Searching/Filtering Information use cases which include: Searching for Club Pages and Filtering Events by Type.

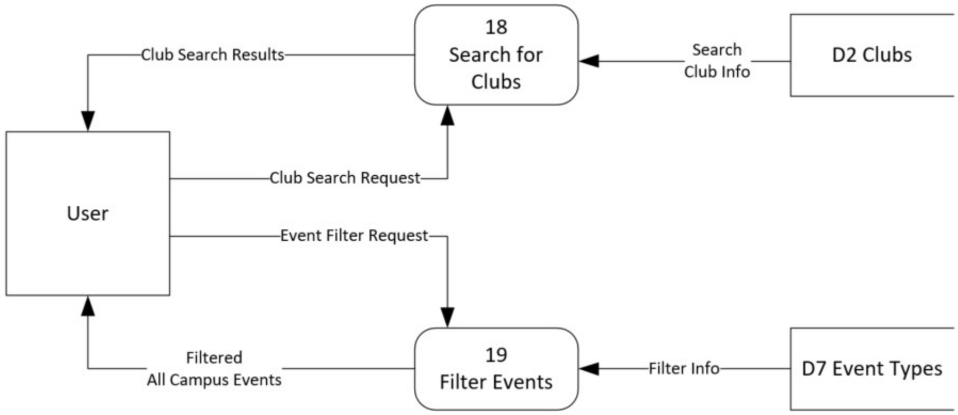


Figure 69 - Level 1 Process 5: Displaying App Information

5.2.7.1 Level 2 Process 18: Searching for Club Pages

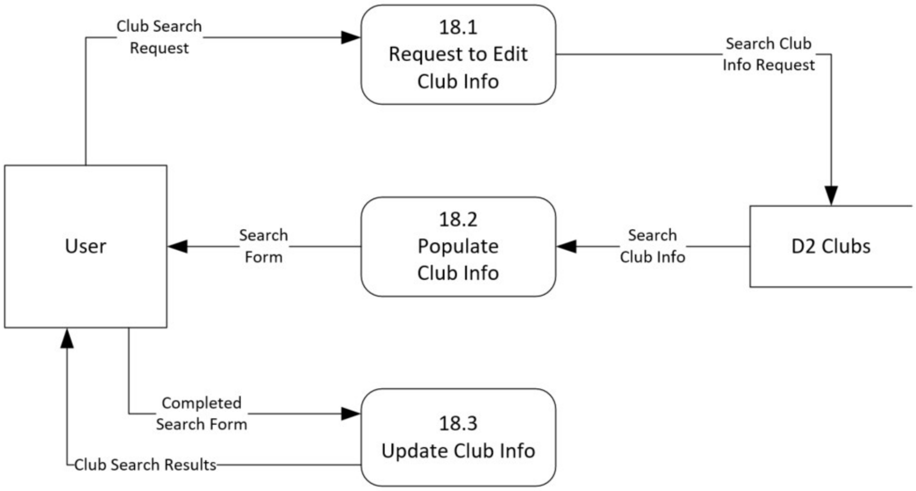


Figure 70- Level 2 Process 18: Searching for Club Pages

- **Club Search Request –** Pressing “Search” button in “JoynU” header bar.
- **Search Club Info Request –** System request for search-specific club information (club name).
- **Search Club Info –** Club names are returned to populate the search form.
- **Search Form –** Search Form including test field with a dropdown list of club names.
- **Completed Search Form –** User uses the search form to begin typing the desired club name.
- **Search Form Results –** Clubs returned to the user that include what was inputted into the text field.

5.2.7.2 Level 2 Process 19: Filtering Events by Type

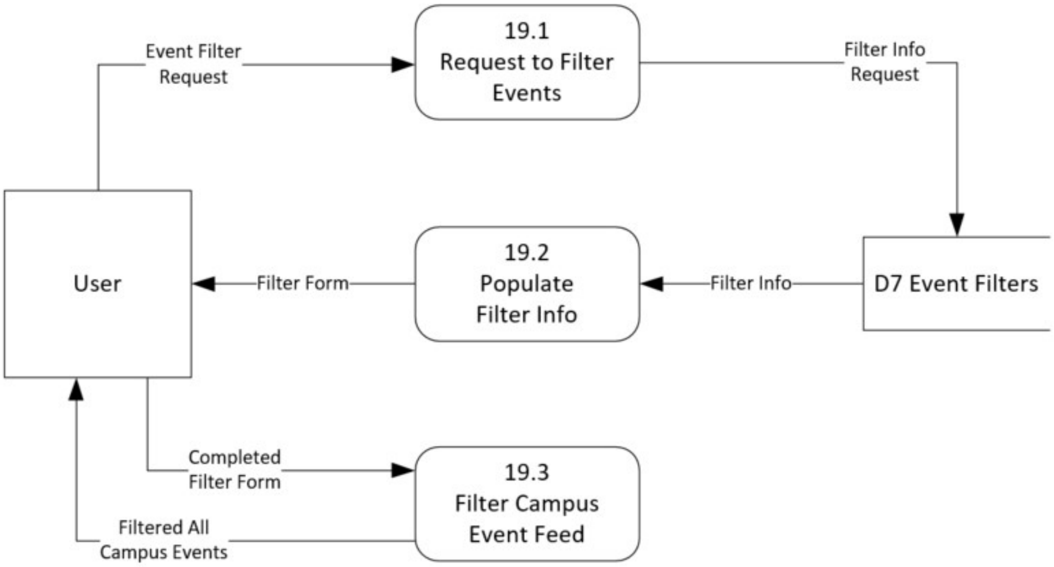

Event Filter Request
19.1 Request to Filter Events
Filter Info Request
User
Filter Form
19.2 Populate Filter Info
Filter Info
D7 Event Filters
Completed Filter Form
19.3 Filter Campus Event Feed
Filtered All Campus Events

Figure 71 - Level 2 Process 19: Filtering Events by Type

- **Event Filter Request –** Pressing "Event Type" on the All Campus Events Page.
- **Filter Info Request –** System request for list of event types.
- **Filter Info –** List of event types to choose from.
- **Filter Form –** Dropdown list for user to select event types from.
- **Completed Filter Form –** User selects an event types to filter the All Campus Events.
- **Filter Form Results –** Filtered list of All Campus Events returned to the user.

5.3 User Experience Design

User experience design is the process of enhancing user satisfaction with a product by improving the accessibility, usability, and pleasure provided in a customer's interaction with said product (Kujala et al., 2011). According to Donald Norman, who created the term in the early 1990's, the concept was meant to cover, "all aspects of the person's experience with the system including industrial design graphics, the interface, [and] the physical interaction" (Holtzblatt, n.d.). Therefore, user experience design is a holistic design process, that when applied to software development, means the consistent gathering of feedback by users to craft applications that address their needs, concerns, and preferences.

The general process of executing user experience design includes collecting information about a problem and who is affected, understanding the users and their environments, designing preliminary solutions based on user goals and needs, and testing and integrating solutions with actual users (UX Design Defined, n.d.). By focusing on user experience throughout the entirety of the software's development benefits both developers and users. General benefits include avoiding costly and unnecessary product features, the simplification of documentation, the improvement of usability and acceptance of the software, and the execution of business and marketing goals while preserving user freedom and empowerment.

6. System Implementation and Iterations

6.1 Iteration Stages

As mentioned earlier in our proposal, our project was separated into 3 separate phases which aligned with each of the 3 terms we would be working on this project. During each term, we significantly improved the application and considered the state of the application at the end of each term to be an ‘iteration.’ In total, we had 3 iterations which were large milestones in the timeline of our project.

6.2 Iteration 1

Our first iteration was a huge learning experience for our team. This iteration consisted of only 2 sprints, but we used this first period to familiarize ourselves with the Scrum Agile methodologies and develop a minimum application.

We decided that the best way to implement scrum for our iterations was to use five lists in Trello: a product backlog list, a sprint backlog list, a task list, an in-progress list, and a completed task list. The product backlog list contained the user stories we had to choose from going into our sprint. The user stories that we chose for this iteration were placed into the sprint backlog, indicating that these would be the user stories we would focus on in the coming two weeks. We then broke down the user stories into concrete tasks that could be carried out by the development team, and stored these in the task list for the sprint. Finally, we indicated our current progress by placing tasks in the “in progress” or “completed” lists.

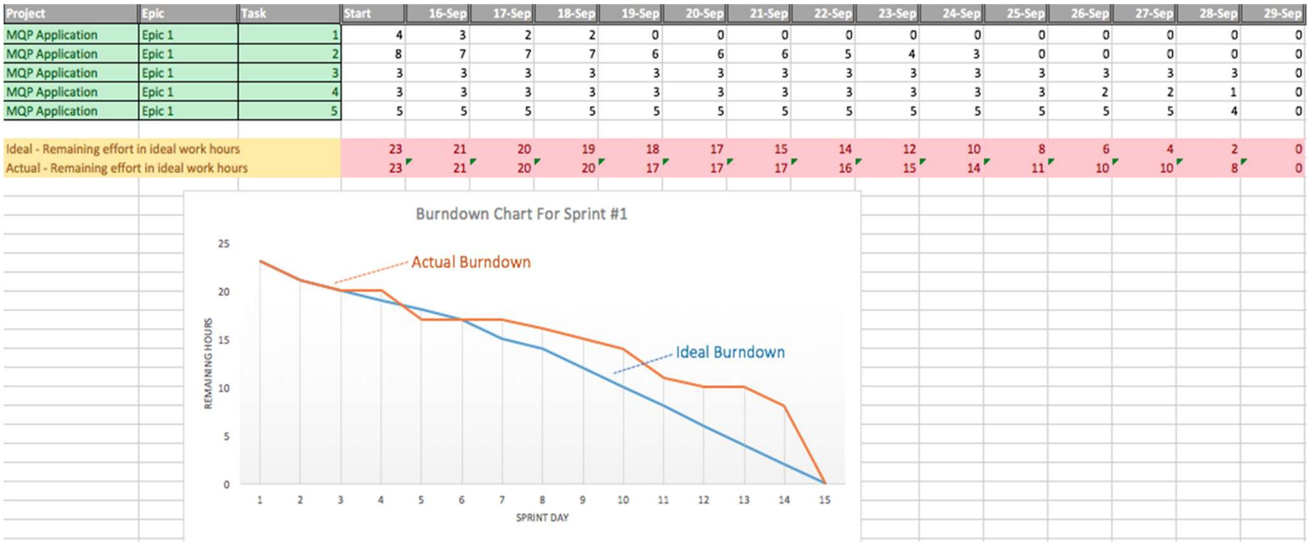
6.2.1 Sprint 1

During our first sprint, we planned to complete the following tasks to begin the development of our project:

1. Set up structural components of rails app with AngularJS and integrate Heroku and PostgreSQL with our application;
2. Create an event model in the database;
3. Add a create event page;
4. Update the database when an event is created.

Over the course of the two weeks we completed our tasks relating to events and setting up, but could not finish our Jasmine tests for our AngularJS code and Task 1. Our progress showed us that when we gave ourselves the time to code, we could get our work done. However, this was not always the case. To combat this, we decided to schedule more work and stand up meetings during the week to better communicate among ourselves what we were working on, and then execute our tasks.

Our burndown chart for sprint #1 is shown below.



Project	Epic	Task	Start	16-Sep	17-Sep	18-Sep	19-Sep	20-Sep	21-Sep	22-Sep	23-Sep	24-Sep	25-Sep	26-Sep	27-Sep	28-Sep	29-Sep
MQP Application	Epic 1	1	4	3	2	2	0	0	0	0	0	0	0	0	0	0	0
MQP Application	Epic 1	2	8	7	7	7	6	6	6	5	4	3	0	0	0	0	0
MQP Application	Epic 1	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	0
MQP Application	Epic 1	4	3	3	3	3	3	3	3	3	3	3	3	2	2	1	0
MQP Application	Epic 1	5	5	5	5	5	5	5	5	5	5	5	5	5	5	4	0
Ideal - Remaining effort in ideal work hours			23	21	20	19	18	17	15	14	12	10	8	6	4	2	0
Actual - Remaining effort in ideal work hours			23	21	20	20	17	17	17	16	15	14	11	10	10	8	0

Figure 72 - Sprint #1 Burndown Summary

6.2.2 Sprint 2

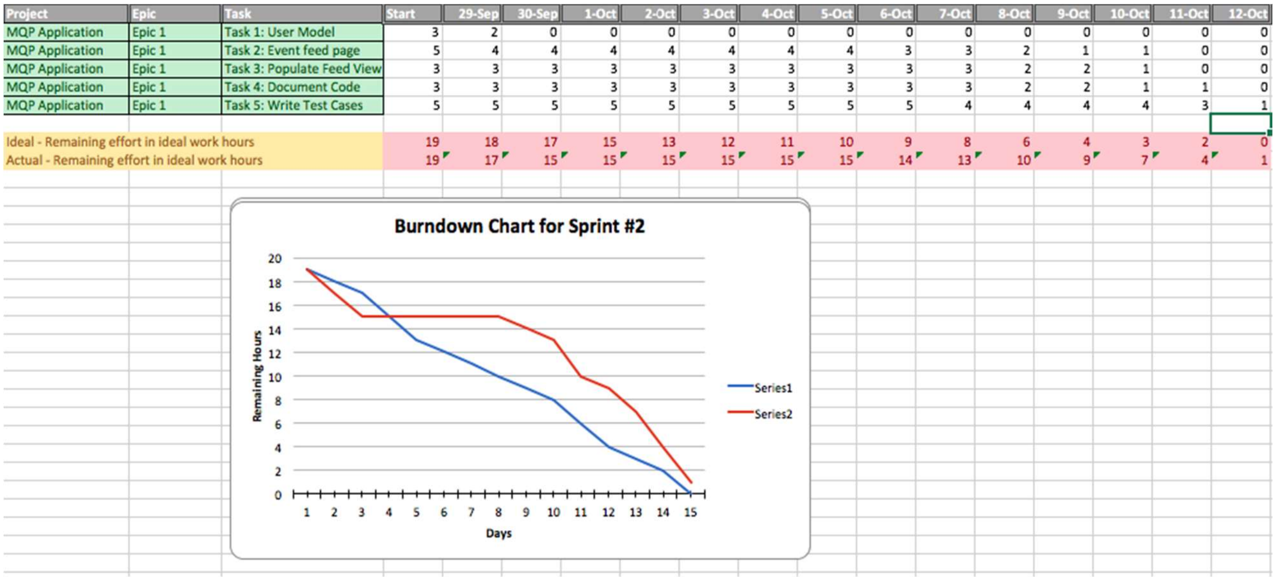
During our second sprint, we planned to complete the following tasks:

1. Create a User model in the database;
2. Add a new event feed page;
3. Read the events from the database and populate the view;
4. Document the code;
5. Write tests for code.

Over the course of the two weeks we completed most of our tasks except task 5, writing all our test cases.

In our sprint, retrospective meeting for sprint #2 we attributed our status at the end of the sprint to not having planned to write test cases in our first sprint. Thus, we had to write test cases for the work we completed in both sprint #1 and sprint #2 during our second sprint. In sprint #2's retrospective meeting, we discovered that we were becoming familiar with how to effectively work together and understand where each of our strengths lied in terms of programming. The things we identified as not going well during the sprint were not discussing the sprint thoroughly enough in our sprint planning meeting and not meeting enough as a group. To address these issues, we scheduled a larger discussion period in our weekly meetings and became more detailed in planning and discussing upcoming sprints.

Our burndown chart for sprint #2 is shown below.



Project	Epic	Task	Start	29-Sep	30-Sep	1-Oct	2-Oct	3-Oct	4-Oct	5-Oct	6-Oct	7-Oct	8-Oct	9-Oct	10-Oct	11-Oct	12-Oct
MQP Application	Epic 1	Task 1: User Model	3	2	0	0	0	0	0	0	0	0	0	0	0	0	0
MQP Application	Epic 1	Task 2: Event feed page	5	4	4	4	4	4	4	4	3	3	2	1	1	0	0
MQP Application	Epic 1	Task 3: Populate Feed View	3	3	3	3	3	3	3	3	3	3	2	2	1	0	0
MQP Application	Epic 1	Task 4: Document Code	3	3	3	3	3	3	3	3	3	3	2	2	1	1	0
MQP Application	Epic 1	Task 5: Write Test Cases	5	5	5	5	5	5	5	5	5	4	4	4	4	3	1
Ideal - Remaining effort in ideal work hours			19	18	17	15	13	12	11	10	9	8	6	4	3	2	0
Actual - Remaining effort in ideal work hours			19	17	15	15	15	15	15	15	14	13	10	9	7	4	1

Figure 73 - Sprint #2 Burndown Summary

6.3 Iteration 2

For our second iteration, we determined that our goal would be to improve upon our first iteration by adding additional features and functionality using information we gathered from our users during our first and second term. For this iteration, we held a total of 4 sprints.

We used a focus group and interviews to gather information from 2 groups of students, event planners and normal students. From these focus groups and interviews we aimed to, and did, receive more in depth information about the application in terms of usability & functionality.

6.3.1 Sprint 3

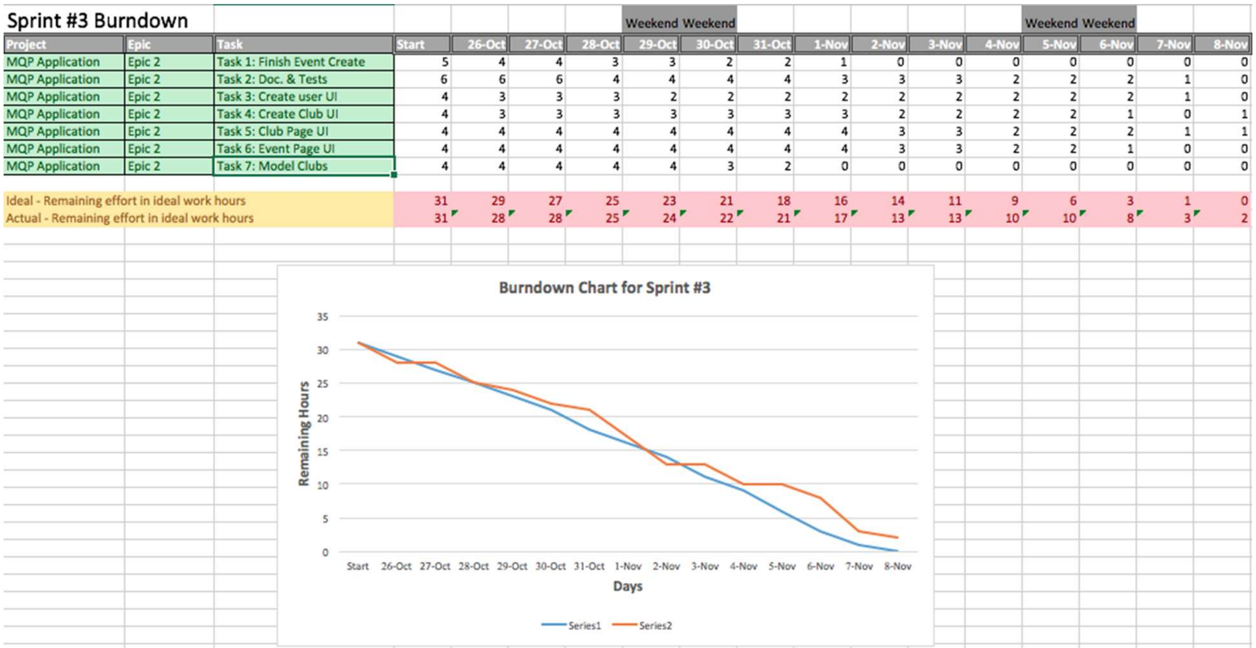
During our third sprint, we planned to complete the following tasks:

1. Finish adding all attributes from ERD to event creation page;
2. Document and write tests as we go along;
3. Develop create user UI;
4. Develop event page UI;
5. Develop club page UI;
6. Create club UI & model club member relationship in database;
7. Model club in database.

Over the course of the two weeks we increased our development velocity because we became more ambitious about the number of tasks we sought to and thought we could complete. After the first several days of this sprint we recognized that if we wanted to complete our objectives that we would have to change our work ethic. We began to meet more consistently and worked longer per meeting to ensure we were putting in enough time into development. Additionally, we began posting specific commit messages in our Github to help reduce confusion.

Despite our changes, we were still unable to finish creating the club UI, and club page UI as their development took longer than what we anticipated. However, we were confident that we would be able to complete these unfinished objectives over the next sprint.

Our burndown chart for sprint #3 is shown on the next page.



Sprint #3 Burndown

							Weekend	Weekend						Weekend	Weekend		
Project	Epic	Task	Start	26-Oct	27-Oct	28-Oct	29-Oct	30-Oct	31-Oct	1-Nov	2-Nov	3-Nov	4-Nov	5-Nov	6-Nov	7-Nov	8-Nov
MQP Application	Epic 2	Task 1: Finish Event Create	5	4	4	3	3	2	2	1	0	0	0	0	0	0	0
MQP Application	Epic 2	Task 2: Doc. & Tests	6	6	6	4	4	4	4	3	3	3	2	2	2	1	0
MQP Application	Epic 2	Task 3: Create user UI	4	3	3	3	2	2	2	2	2	2	2	2	2	1	0
MQP Application	Epic 2	Task 4: Create Club UI	4	3	3	3	3	3	3	3	2	2	2	2	1	0	1
MQP Application	Epic 2	Task 5: Club Page UI	4	4	4	4	4	4	4	4	3	3	2	2	2	1	1
MQP Application	Epic 2	Task 6: Event Page UI	4	4	4	4	4	4	4	4	3	3	2	2	1	0	0
MQP Application	Epic 2	Task 7: Model Clubs	4	4	4	4	4	3	2	0	0	0	0	0	0	0	0
Ideal - Remaining effort in ideal work hours			31	29	27	25	23	21	18	16	14	11	9	6	3	1	0
Actual - Remaining effort in ideal work hours			31	28	28	25	24	22	21	17	13	13	10	10	8	3	2

Figure 74 - Sprint #3 Burndown Summary

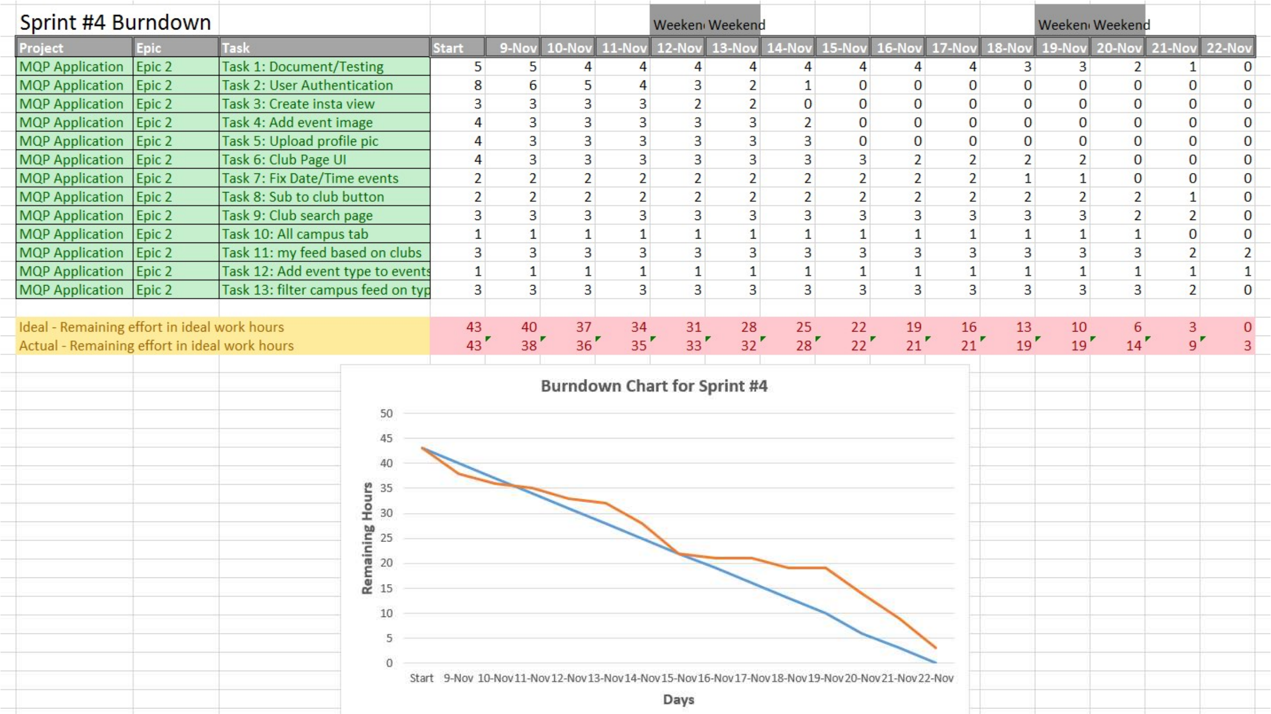
6.3.2 Sprint 4

During our fourth sprint, we planned to complete the following tasks:

1. Document and Test code;
2. User authentication (sign in or sign up);
3. Create an Instagram-like view for events;
4. Add event image to event creation;
5. Upload picture for profile;
6. Develop Club Page UI;
7. Fix Event Date and Time formatting;
8. Club subscription button developed;
9. Add a club search page;
10. Implement "All Campus" tab;
11. Implement “My Feed” based on club subscriptions;
12. Add an "event type" to the event model for filtering purposes;
13. Add a filter button the entire campus event feed.

Much like our last sprint, we increased our velocity in order to complete more tasks and see just how much we could accomplish when all working together. Utilizing the workflow changes from sprint #3, we could complete all but two of our thirteen tasks: implementing the “My Feed” events tab and adding event types to events. In our retrospective meeting, we realized that that our external commitments for the final week prevented us from putting in the full development hours we predicted we needed. However, we were confident we would complete these tasks quickly into the next sprint.

Our burndown chart for sprint #4 is shown on the next page.



Sprint #4 Burndown

							Weeken	Weekend						Weeken	Weekend		
Project	Epic	Task	Start	9-Nov	10-Nov	11-Nov	12-Nov	13-Nov	14-Nov	15-Nov	16-Nov	17-Nov	18-Nov	19-Nov	20-Nov	21-Nov	22-Nov
MQP Application	Epic 2	Task 1: Document/Testing	5	5	4	4	4	4	4	4	4	4	3	3	2	1	0
MQP Application	Epic 2	Task 2: User Authentication	8	6	5	4	3	2	1	0	0	0	0	0	0	0	0
MQP Application	Epic 2	Task 3: Create insta view	3	3	3	3	2	2	0	0	0	0	0	0	0	0	0
MQP Application	Epic 2	Task 4: Add event image	4	3	3	3	3	3	2	0	0	0	0	0	0	0	0
MQP Application	Epic 2	Task 5: Upload profile pic	4	3	3	3	3	3	3	0	0	0	0	0	0	0	0
MQP Application	Epic 2	Task 6: Club Page UI	4	3	3	3	3	3	3	3	2	2	2	2	0	0	0
MQP Application	Epic 2	Task 7: Fix Date/Time events	2	2	2	2	2	2	2	2	2	2	1	1	0	0	0
MQP Application	Epic 2	Task 8: Sub to club button	2	2	2	2	2	2	2	2	2	2	2	2	2	1	0
MQP Application	Epic 2	Task 9: Club search page	3	3	3	3	3	3	3	3	3	3	3	3	2	2	0
MQP Application	Epic 2	Task 10: All campus tab	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0
MQP Application	Epic 2	Task 11: my feed based on clubs	3	3	3	3	3	3	3	3	3	3	3	3	3	2	2
MQP Application	Epic 2	Task 12: Add event type to events	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
MQP Application	Epic 2	Task 13: filter campus feed on typ	3	3	3	3	3	3	3	3	3	3	3	3	3	2	0
Ideal - Remaining effort in ideal work hours			43	40	37	34	31	28	25	22	19	16	13	10	6	3	0
Actual - Remaining effort in ideal work hours			43	38	36	35	33	32	28	22	21	21	19	19	14	9	3

Figure 75 - Sprint #4 Burndown Summary

6.3.3 Sprint 5

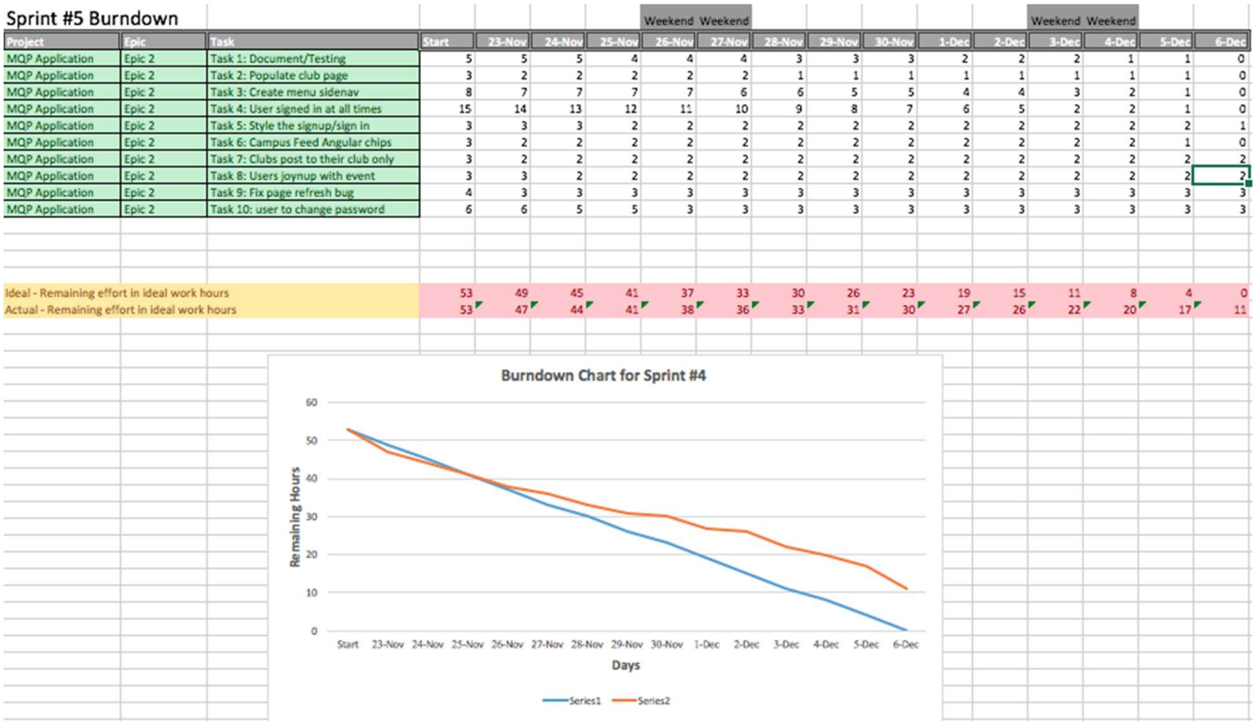
During our fifth sprint, we planned to complete the following tasks:

1. Document and Testing code;
2. Populate club page;
3. Create menu side navigation (drawer);
4. Make sure user can be signed in always after initial sign-in;
5. Style the sign-up/sign-in page;
6. Add Campus feed angular chips;
7. Edit code to make sure clubs can post events to their clubs only;
8. Edit code so users can 'joynup' to events;
9. Fix page refresh bug;
10. Allow user to change their password.

Over the course of the two weeks we completed most of our tasks, however there were many tasks that were close to completion (tasks #5, #7, #8, #9, #10). Essentially, we were able to complete the main tasks for the sprint that we needed to accomplish but ended up not being able to complete a few of the 'extra' tasks we had assigned ourselves.

In our sprint, retrospective meeting for sprint #5 we attributed our performance to having underestimated the amount of time needed to complete a few of the tasks. Overall, we worked the appropriate number of hours that we had originally planned, but were not able to predict that our time estimates would be as off as much as they were. In this sprint, retrospective meeting, the things we discussed that went well were that we were all putting in a lot of time into development as a team and that we could accomplish the amount of hours/work we could physically manage each week. The things we identified as not going well during the sprint was miscalculating the amount of time required for a few tasks. To address these issues, we planned to reference previous sprint tasks when creating tasks for new sprints so that we would not commit to completing tasks and then being unable to do so.

Our burndown chart for sprint #5 is shown on the next page.



Sprint #5 Burndown

							Weekend	Weekend						Weekend	Weekend		
Project	Epic	Task	Start	23-Nov	24-Nov	25-Nov	26-Nov	27-Nov	28-Nov	29-Nov	30-Nov	1-Dec	2-Dec	3-Dec	4-Dec	5-Dec	6-Dec
MQP Application	Epic 2	Task 1: Document/Testing	5	5	5	4	4	4	3	3	3	2	2	2	1	1	0
MQP Application	Epic 2	Task 2: Populate club page	3	2	2	2	2	2	1	1	1	1	1	1	1	1	0
MQP Application	Epic 2	Task 3: Create menu sidenav	8	7	7	7	7	6	6	5	5	4	4	3	2	1	0
MQP Application	Epic 2	Task 4: User signed in at all times	15	14	13	12	11	10	9	8	7	6	5	2	2	1	0
MQP Application	Epic 2	Task 5: Style the signup/sign in	3	3	3	2	2	2	2	2	2	2	2	2	2	2	1
MQP Application	Epic 2	Task 6: Campus Feed Angular chips	3	2	2	2	2	2	2	2	2	2	2	2	2	1	0
MQP Application	Epic 2	Task 7: Clubs post to their club only	3	2	2	2	2	2	2	2	2	2	2	2	2	2	2
MQP Application	Epic 2	Task 8: Users joynup with event	3	3	2	2	2	2	2	2	2	2	2	2	2	2	2
MQP Application	Epic 2	Task 9: Fix page refresh bug	4	3	3	3	3	3	3	3	3	3	3	3	3	3	3
MQP Application	Epic 2	Task 10: user to change password	6	6	5	5	3	3	3	3	3	3	3	3	3	3	3
Ideal - Remaining effort in ideal work hours			53	49	45	41	37	33	30	26	23	19	15	11	8	4	0
Actual - Remaining effort in ideal work hours			53	47	44	41	38	36	33	31	30	27	26	22	20	17	11

Figure 76 - Sprint #5 Burndown Summary

6.3.4 Sprint 6

During our sixth sprint, we planned to complete the following tasks:

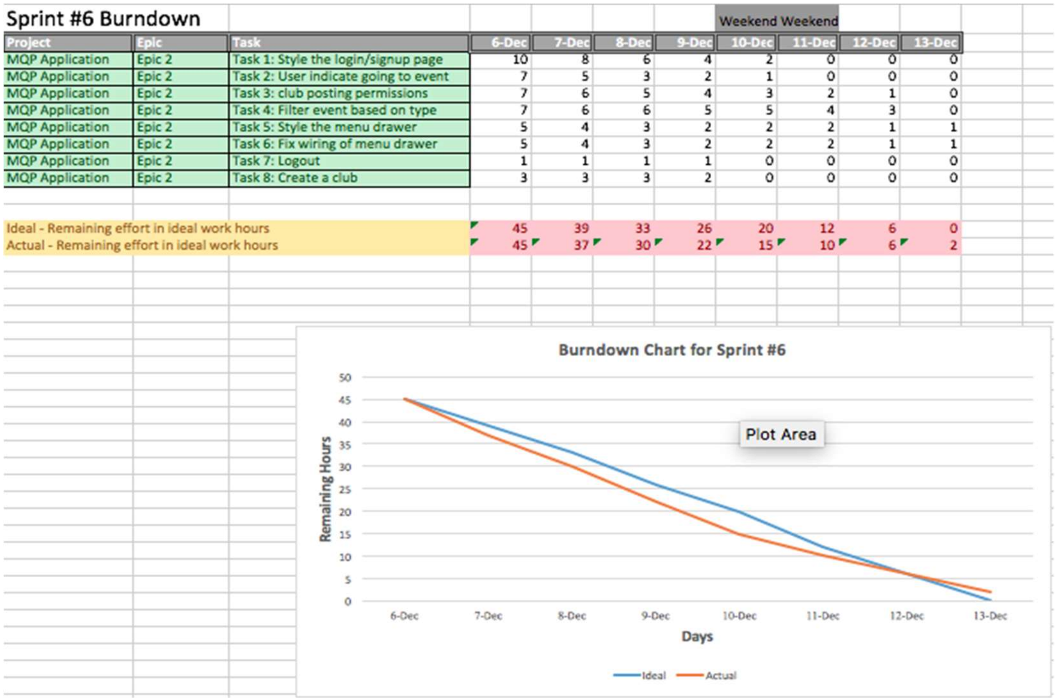
1. Style the login/signup page;
2. Implement button indicating user going to an event;
3. Implement club posting permissions;
4. Implement multiple filters for events based on type;
5. Style the menu drawer;
6. Implement Logout functionality;
7. Add club creation page.

Since we started sprint #6 with one week left in the term, we decided to only have the sprint go until the last day of the term instead of a normal 2-week period. Our team could complete all of our planned tasks and more this week. We had 1 hour of work left on the day our sprint ended, but could complete it very soon after that time before the end of the term.

During our retrospective meeting for this sprint, what we felt went well was that we completed all our tasks, we worked at double of our normal velocity, and complete more smaller tasks that came up that were not necessarily large enough to be documented in our backlog.

In terms of what we felt did not go so well during the term, our team did not identify anything. What was hard during this sprint was external obligations that each team member had such as course finals and interviews. Overall, we felt very good about this sprint and considered it a great success.

Our burndown chart for sprint #6 is shown below.



Sprint #6 Burndown

							Weekend	Weekend		
Project	Epic	Task	6-Dec	7-Dec	8-Dec	9-Dec	10-Dec	11-Dec	12-Dec	13-Dec
MQP Application	Epic 2	Task 1: Style the login/signup page	10	8	6	4	2	0	0	0
MQP Application	Epic 2	Task 2: User indicate going to event	7	5	3	2	1	0	0	0
MQP Application	Epic 2	Task 3: club posting permissions	7	6	5	4	3	2	1	0
MQP Application	Epic 2	Task 4: Filter event based on type	7	6	6	5	5	4	3	0
MQP Application	Epic 2	Task 5: Style the menu drawer	5	4	3	2	2	2	1	1
MQP Application	Epic 2	Task 6: Fix wiring of menu drawer	5	4	3	2	2	2	1	1
MQP Application	Epic 2	Task 7: Logout	1	1	1	1	0	0	0	0
MQP Application	Epic 2	Task 8: Create a club	3	3	3	2	0	0	0	0
Ideal - Remaining effort in ideal work hours			45	39	33	26	20	12	6	0
Actual - Remaining effort in ideal work hours			45	37	30	22	15	10	6	2

Figure 77 - Sprint #6 Burndown Summary

6.4 Iteration 3

For our third iteration, we determined that our goal would be to improve upon our application at a faster rate. Expecting to a least beta-test our application by the end of the term, we decided to execute one-week sprints for the entirety of C-Term.

We used a focus group and interviews to gather information from 2 groups of students, event planners and normal students. From these focus groups and interviews we aimed to, and did, receive more in depth information about the application in terms of usability & functionality.

6.4.1 Sprint 7

During our seventh sprint, we planned to complete the following tasks:

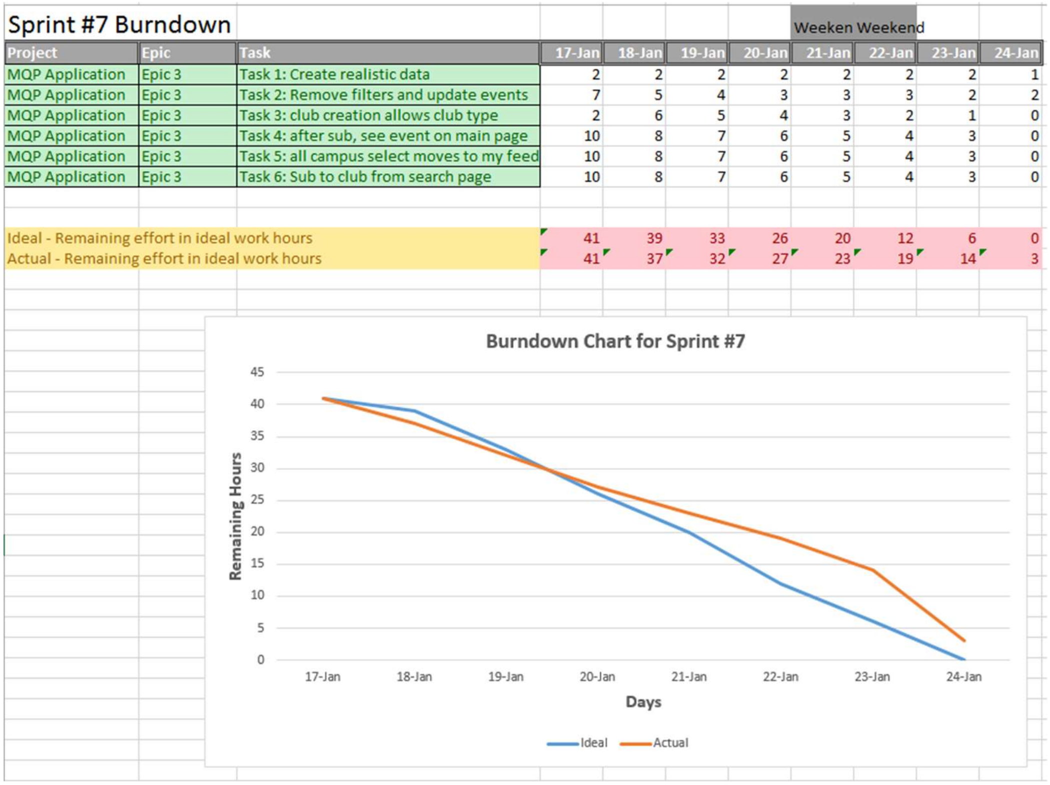
1. Create realistic data;
2. Remove filters and update events;
3. Club creation will allow for club type;
4. After subscription, see event on main page;
5. All campus select moves to my feed;
6. Subscribe to club from search page;

Sprint #7 began after our winter break. At this point we decided that each of our C term sprints would be condensed to one week in length, to quickly report our progress back to our stakeholders.

During our retrospective meeting for this sprint, what we felt went well was that we got back into the swing of things after a long break quite nicely. We finished many of the key tasks, that allowed our application to keep moving forward.

In terms of what we felt did not go so well during the sprint, we were disappointed that we were unable to create realistic data within the app. This could have been accomplished if we had allotted more time to development throughout the week. Overall, we felt good about this sprint, but we knew we had to be more efficient in the coming weeks.

Our burndown chart for sprint #7 is shown on the next page.



Sprint #7 Burndown

Project	Epic	Task	17-Jan	18-Jan	19-Jan	20-Jan	21-Jan	22-Jan	23-Jan	24-Jan
							Weeken	Weekend		
MQP Application	Epic 3	Task 1: Create realistic data	2	2	2	2	2	2	2	1
MQP Application	Epic 3	Task 2: Remove filters and update events	7	5	4	3	3	3	2	2
MQP Application	Epic 3	Task 3: club creation allows club type	2	6	5	4	3	2	1	0
MQP Application	Epic 3	Task 4: after sub, see event on main page	10	8	7	6	5	4	3	0
MQP Application	Epic 3	Task 5: all campus select moves to my feed	10	8	7	6	5	4	3	0
MQP Application	Epic 3	Task 6: Sub to club from search page	10	8	7	6	5	4	3	0
Ideal - Remaining effort in ideal work hours			41	39	33	26	20	12	6	0
Actual - Remaining effort in ideal work hours			41	37	32	27	23	19	14	3

Figure 78 - Sprint #7 Burndown Summary

6.4.2 Sprint 8

During our eighth sprint, we planned to complete the following tasks:

1. Create an event location description;
2. Unattend/Attend syncs in all places;
3. Menu bar updates correctly;
4. Refractor front end code;
5. Move code over to PhoneGap structure;

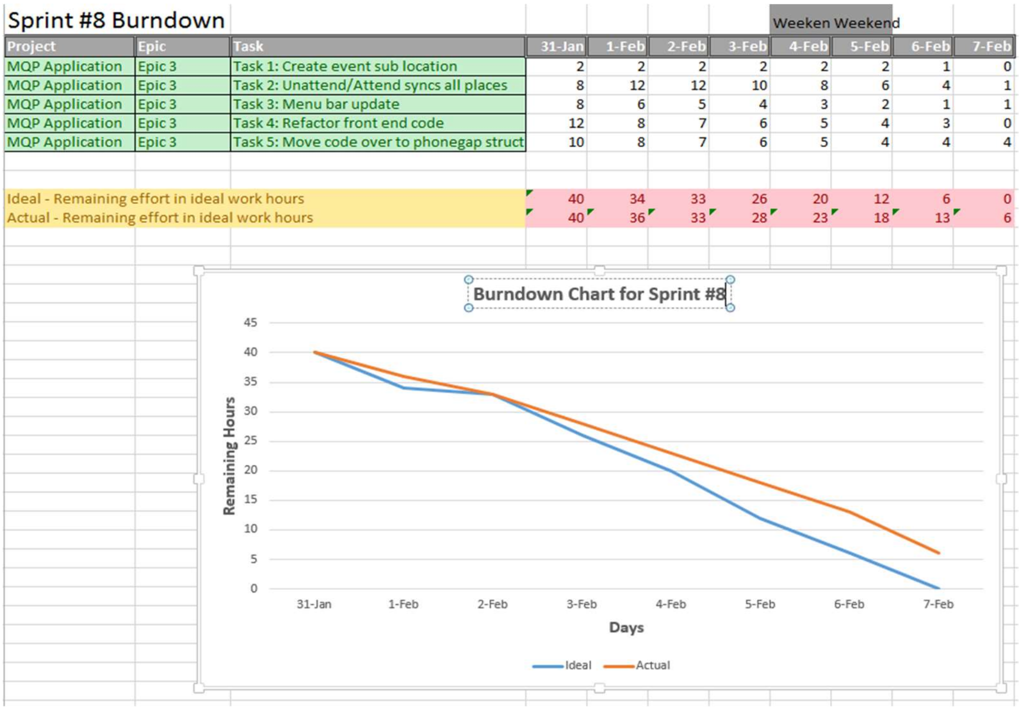
We began Sprint #8 with the intention to improve upon our effort from the last week. Although we did not fully meet our goals for this sprint, it was a very important sprint overall.

In our retrospective meeting, we were very glad that we could refactor the way that the front end handled the data. This sprint, we effectively condensed our calls to the back end and created a single location where data could be pulled from. This made the code more organized moving forward, which allowed us to tackle my feed and campus feed syncing problems we had been experiencing before.

The retrospective meeting also brought up the PhoneGap task. It was decided that since our efforts kept coming up short in this area, that we would place this task into our product backlog, with a lower priority.

Overall, this sprint was very successful, but it caused us to make some important design decisions that affected the rest of our term.

Our burndown chart for sprint #8 is shown below.



Sprint #8 Burndown

							Weeken	Weekend		
Project	Epic	Task	31-Jan	1-Feb	2-Feb	3-Feb	4-Feb	5-Feb	6-Feb	7-Feb
MQP Application	Epic 3	Task 1: Create event sub location	2	2	2	2	2	2	1	0
MQP Application	Epic 3	Task 2: Unattend/Attend syncs all places	8	12	12	10	8	6	4	1
MQP Application	Epic 3	Task 3: Menu bar update	8	6	5	4	3	2	1	1
MQP Application	Epic 3	Task 4: Refactor front end code	12	8	7	6	5	4	3	0
MQP Application	Epic 3	Task 5: Move code over to phonegap struct	10	8	7	6	5	4	4	4
Ideal - Remaining effort in ideal work hours			40	34	33	26	20	12	6	0
Actual - Remaining effort in ideal work hours			40	36	33	28	23	18	13	6

Figure 79 - Sprint #8 Burndown Summary

6.4.3 Sprint 9

During our ninth sprint, we planned to complete the following tasks:

1. Give users the capability to edit events;

Our Sprint #9 was created from testing our app with users. One of the main issues initial users found was that there was edit event capabilities on the app. So, during this sprint, we focused on implementing this capability. As a team, we identified the area in the UI where the user would be able to edit the event. We determined that below the event image on the right side would be a fitting place, so we inserted an ellipsis that would indicate to the user that there were more options available to the user on click. Once the user chooses to edit the event, they are brought to the edit event page, which auto fills many of the fields based on the current event information. Once the user changes what they want, they save the event and it is updated in the feeds.

During our sprint, retrospective meeting, we were pleased that we took feedback from testing and effectively implemented a feature into our app. We were thankful that the scrum methodology we followed allowed us to carry this task out in such a short timeframe.

6.4.4 Sprint 10

During our tenth sprint, we planned to complete the following tasks:

1. Correct Signup UI;
2. Created event will show up automatically on feed
3. User will can edit clubs

During sprint #10, we mirrored our sprint #9 process. Through user testing, we identified that there were three main issues keeping users from using the application effectively: Signing up would register a user in the database, but the UI was buggy, created events would not show up on the user feed, and clubs could not be edited. So, through a sprint planning meeting, we identified the areas of the code that needed work done, and came up with a plan for implementation.

After the sprint was over, we were once again pleased with how we took user feedback to create features within us during app over the course of one week. We also used much of the logic from editing events to implement edit clubs, which sped up our development.

6.4.5 Sprint 11

During our tenth sprint, we planned to complete the following tasks:

1. Implement permissions for administrators;
2. Prevent non-administrators from seeing tasks only available to administrators

During sprint #11, we wrapped up issues that were preventing us from starting our acceptance testing. These included restricting certain features of the application to those with proper authentication. Before this sprint, any user was able to create clubs, events, and edit those club and event pages without any restriction. Through this sprint planning meeting we determined that implementing this type of system where users can only access certain features based on their credentials was the primary and exclusive goal for this sprint.

After this sprint was over, the team was very satisfied with the progress made on the application and through the 11 agile sprints we stuck to throughout the terms. We determined that our use of code from previous sprints allowed us to implement these features much quicker than in previous sprints and we were satisfied with much more rapid progress.

7. Testing

7.1 Background

Testing any programming project is critical to ensuring working code and efficient project purpose completion. A project author cannot ensure code fluidity and durability for intended users without such testing. Any code errors may potentially cause terrible user experiences, making any product viability nearly impossible without rigorous testing.

7.2 Test Plan

Testing projects at a large scale can be difficult due to high code usage variability for any potential user. Testing should attempt to cover every case a user could use the code along with their acceptance of the project. Large-scale variable testing quickly becomes complex, requiring efficient test breakdown to check specific requirements.

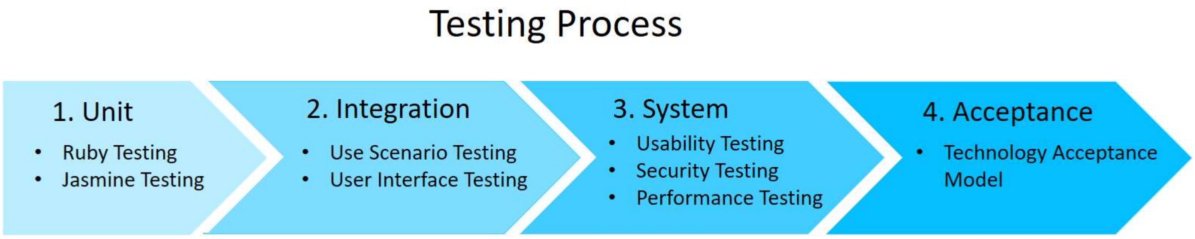


Figure 80 - Testing Process Diagram

Since our project was complex by nature our team decided to create a thorough test plan, which broke down and detailed exactly how we planned to test the application. Looking at all of the application's capabilities we identified four types of tests to ensure it worked properly: Unit Testing, Integration Testing, System Testing, and Acceptance Testing.

7.3 Unit Testing

Our team chose Unit Testing to test first. Unit testing is a software development process where the smallest testable parts of an application's code, units, are individually and independently scrutinized for proper operation. Unit testing can be done manually but often are automated. To conduct our unit tests our team used built-in functionality in Rails to test the applications' back end and with Jasmine for front-end testing.

Rails is the web application framework that our team used to develop the back-end portion of the application. We chose Rails partially due to built-in testing features to allow for easy long-term back-end unit testing. When generating models and controllers from the command line using the rails generator command, test files are automatically generated to test every feature that is initially modeled so developers have some of their code tested before they even being to develop. This helps keep developers honest and assisted our team in keeping up with the test-driven development methodology.

Using Rails our team tested the following:

1. Models could be created and saved properly on the backend with a correct representation in the database along with any foreign keys associated with that model.
2. The models could be retrieved, modified, and saved over.
3. That users were restricted to certain features of the application based on permissions stored in the database.
4. That users could save passwords in a secure manner with proper hashing.

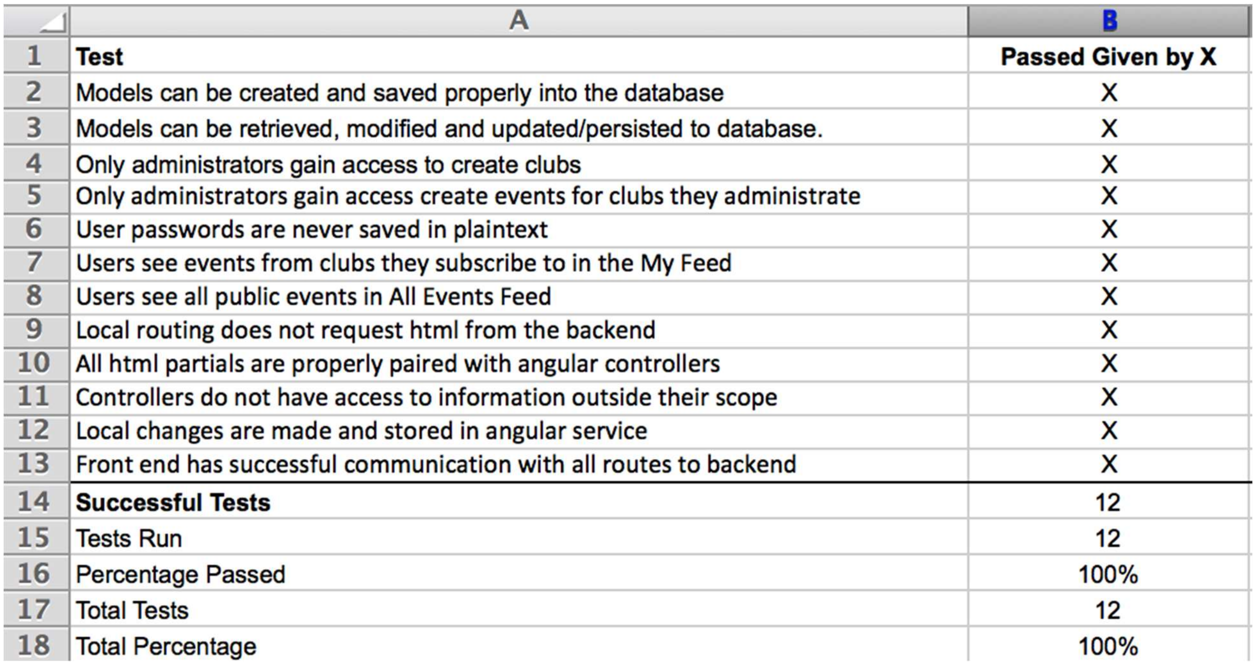
AngularJS along with other front-end code can often be very difficult to test due to its asynchronous nature. However, the issue was quickly resolved by the AngularJS user community where tools were developed to allow authors to generate unit tests for their JavaScript (as well as Angular). The framework our team used to create these tests was Jasmine, an open source software that is readily available for any developer to use. Jasmine is easy to implement into Rails, only requiring an additional gem dependency. Running a bundle install command easily installs the software. The team found Jasmine very difficult to use initially, but it was critical in maintaining the code and ensuring that code worked across browsers and machines outside of the development environment used.

Using Jasmine our team tested the following:

1. Controllers can match up with html partials
2. Ensure that local routing works so that the application feels native
3. Ensure users do not see buttons and other features that the backend would reject based on their permissions.

Because of our difficulty understanding Jasmine, much of the testing associated with the front-end Angular code was verified through vigorous manual integration testing. In reflection, it would have been beneficial to dedicate more time to this area of testing early on in A term as it would have saved countless hours of manual testing of our application once more precise automated tests were constructed.

7.3.1 Results



	A	B
1	**Test**	**Passed Given by X**
2	Models can be created and saved properly into the database	X
3	Models can be retrieved, modified and updated/persisted to database.	X
4	Only administrators gain access to create clubs	X
5	Only administrators gain access create events for clubs they administrate	X
6	User passwords are never saved in plaintext	X
7	Users see events from clubs they subscribe to in the My Feed	X
8	Users see all public events in All Events Feed	X
9	Local routing does not request html from the backend	X
10	All html partials are properly paired with angular controllers	X
11	Controllers do not have access to information outside their scope	X
12	Local changes are made and stored in angular service	X
13	Front end has successful communication with all routes to backend	X
14	**Successful Tests**	12
15	Tests Run	12
16	Percentage Passed	100%
17	Total Tests	12
18	Total Percentage	100%

Figure 81 - Unit Testing Results

The figure above shows a summary of groupings of unit tests we completed to ensure that our application works as expected. While further automated testing is always helpful for any development project, our tests are extensive enough that we are confident that together with all the integration and acceptance testing, our application works as intended.

7.4 Integration Testing

After the completion of unit testing, our team conducted Integration Testing on the application. Integration testing is the phase in software testing in which individual software modules are combined and tested in conjunction with one another or as a group. This is done to ensure that systems operate as intended under every possible situation. The method in which our team decided was best to conduct integration testing was to split it up into and run both Use Scenario tests and User Interface tests.

7.4.1 Use Scenario Testing

Use Scenario Testing is a software testing activity that uses scenarios, like use cases, to assist testers in verifying if a particular part of a system is functioning as intended. This type of testing helps testers understand and explore how the software will work in the hands of end users, and it is for that purpose that our team decided to conduct these tests.

To conduct these tests our team made the decision to take all of our previously created use cases and perform each one on the application. Use cases are intended to document what an end user should be able to do in a system, so we believed verifying that each use case was able to be performed would be sufficient in testing whether every scenario a user could be in with the application is able to be completed

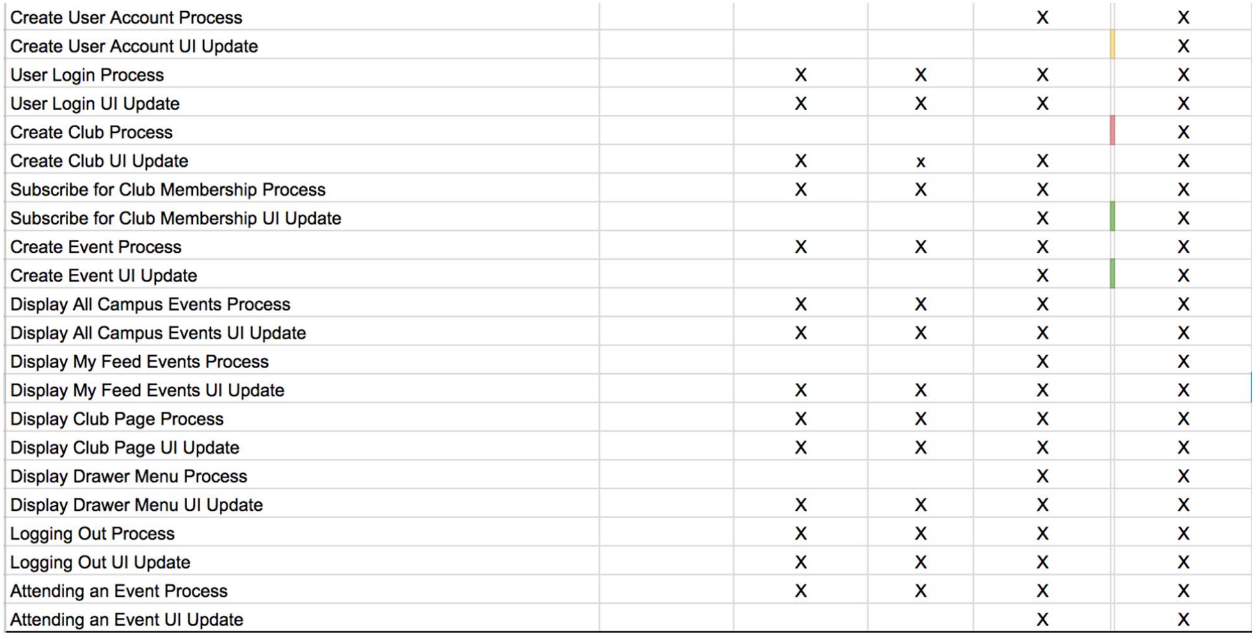
The following are each of the Use Cases that our team tested by performing them on the app:

- UC-01 Create User Account
- UC-02 User Login
- UC-03 Create Club
- UC-04 Subscribe to Club Membership
- UC-05 Create Event
- UC-06 Display All Campus Events
- UC-07 Display My Feed Events
- UC-08 Displaying Club Page
- UC-09 Displaying Drawer Menu
- UC-10 Logging Out
- UC-11 Attending an Event

7.4.2 Use Scenario Testing Results

Our group performed our first round of use scenario testing on 2/20/17. After attempting to execute each of our use cases, 7 out of our 11 tests for them passed. The use cases that were not able to be completed and did not pass our tests were UC-1, UC-3, UC-8, and UC-10. Specifically, we were unable to completely Create User Account, Create a Club, Display My Event Feed, and Display Drawer Menu. After documenting these results our team took another look at the application's code and progressively made corrections so that the failed use cases could be executed properly.

After making our code corrections we reran all of our tests multiple times. By our 4th round of Use Scenario testing, the results were that all of our tests passed. The results for all rounds of testing are shown below.



Create User Account Process				X		X
Create User Account UI Update					🟨	X
User Login Process		X	X	X		X
User Login UI Update		X	X	X		X
Create Club Process					🟥	X
Create Club UI Update		X	x	X		X
Subscribe for Club Membership Process		X	X	X		X
Subscribe for Club Membership UI Update				X	🟩	X
Create Event Process		X	X	X		X
Create Event UI Update				X	🟩	X
Display All Campus Events Process		X	X	X		X
Display All Campus Events UI Update		X	X	X		X
Display My Feed Events Process				X		X
Display My Feed Events UI Update		X	X	X		X
Display Club Page Process		X	X	X		X
Display Club Page UI Update		X	X	X		X
Display Drawer Menu Process				X		X
Display Drawer Menu UI Update		X	X	X		X
Logging Out Process		X	X	X		X
Logging Out UI Update		X	X	X		X
Attending an Event Process		X	X	X		X
Attending an Event UI Update				X		X

Figure 82 - Use Scenario & User Interface Testing Results (Just Make Use Scenarios)

7.4.3 User Interface Testing

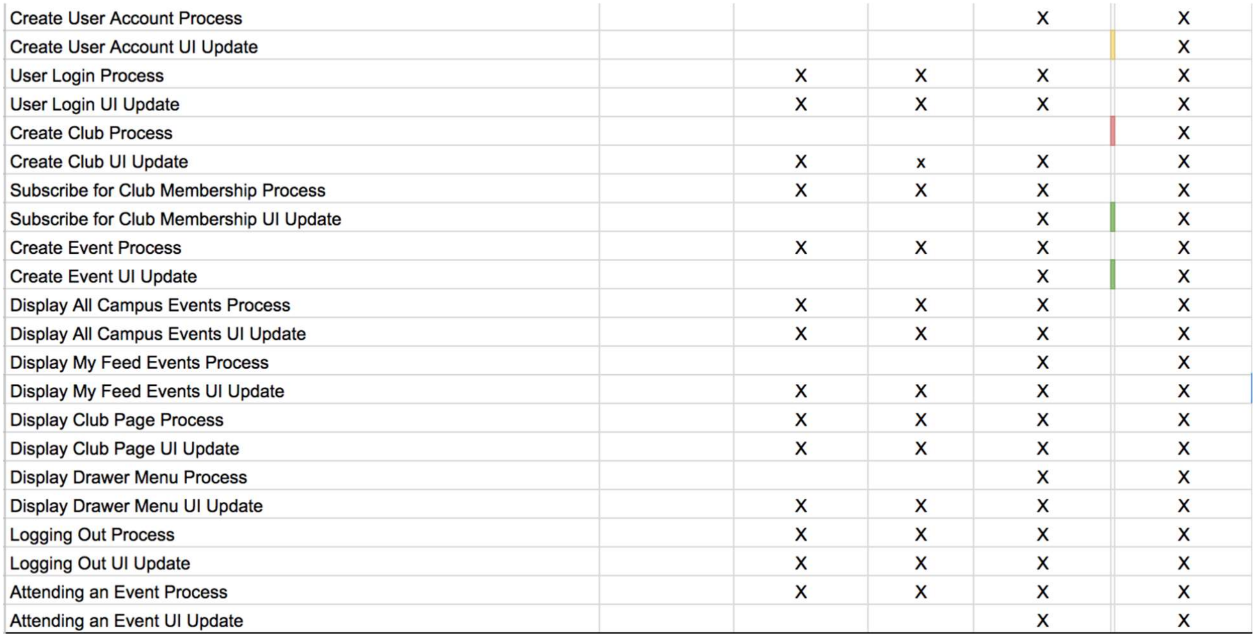
User Interface Testing is a software testing technique to identify the presence of any system's graphical user interface defects. In addition to Use Scenario testing, we felt it was critical that each use case was not only able to be performed correctly but that the user interface of the application accurately reflected what a user should see after completing each case. To perform this testing, after completing each use case our team checked the application's interface to verify that it displayed the proper changes and updates. We did this testing in conjunction with our use scenario testing and we checked the following user interfaces to make sure the appropriate changes were made:

- UI-01 Signup Page
- UI-02 Login Page
- UI-03 All Campus Events Page
- UI-04 My Feed Events Page
- UI-05 Club Search Page
- UI-06 Drawer Menu
- UI-07 Club Creation Page
- UI-08 Event Creation Page
- UI-09 Club Information Page

7.4.4 User Interface Testing Results

Our group performed our first round of user interface testing on 2/20/17 along with our use scenario testing. Since we decided it best to do these tests together, our first round of user interface testing did not fully pass. The results for the first round were 7 of 11 tests passing. The four (4) tests that failed did so because their associated use cases could not be performed. After documenting our results our team edited the application's code and made corrections so that the failed use cases could be executed properly.

After making our code corrections we reran all of our tests multiple times. By our 4th round of Use Interface testing the results were that all of our tests passed. The results for all rounds of testing are shown below.



Create User Account Process				X		X
Create User Account UI Update					🟨	X
User Login Process		X	X	X		X
User Login UI Update		X	X	X		X
Create Club Process					🟥	X
Create Club UI Update		X	x	X		X
Subscribe for Club Membership Process		X	X	X		X
Subscribe for Club Membership UI Update				X	🟩	X
Create Event Process		X	X	X		X
Create Event UI Update				X	🟩	X
Display All Campus Events Process		X	X	X		X
Display All Campus Events UI Update		X	X	X		X
Display My Feed Events Process				X		X
Display My Feed Events UI Update		X	X	X		X
Display Club Page Process		X	X	X		X
Display Club Page UI Update		X	X	X		X
Display Drawer Menu Process				X		X
Display Drawer Menu UI Update		X	X	X		X
Logging Out Process		X	X	X		X
Logging Out UI Update		X	X	X		X
Attending an Event Process		X	X	X		X
Attending an Event UI Update				X		X

Figure 83 - Use Scenario & User Interface Testing Results (Just Make User Interface)

7.5 System Testing

System testing is a level of software testing where complete and integrated software is tested. A system test's purpose is to evaluate the system's compliance with specific requirements. To test the application against our requirements, our group conducted system testing by holding a three person focus group to perform the four different system tests: Usability Testing, Security Testing, Performance, and Business Requirements Testing.

7.5.1 Usability Testing

Usability testing is a method used to evaluate how easy a system is to use. These tests typically take place with potential users to accurately measure how 'usable' or 'intuitive' a system is and how easy it is for users to reach their goals and conduct certain functionality.

We held a focus group and had users access the application to try and complete certain tasks. As users were or were not able to complete tasks, we took note of that as a test either passing or failing.

The list of tasks we asked our focus group participants to complete for usability testing were as follows:

- N1.1 The system should be operational on iOS, Android, and web browsers.
- N1.2 The system should be integrated with the WPI's Central Authentication System (CAS).
- N1.3 The system should only allow for the uploading of specific file types when creating an event or club.
- N1.4 The system should only allow for the correct type of data to be entered specific text boxes and menus.
- N1.5 During a system restart, the system will be able to return to a functioning state.
- N1.6 Club admins must be able to create, edit, and delete any content related to their affiliated club.

7.5.2 Usability Testing Results

Based on our initial focus group held on 2/23/17, only three of the six usability tests passed successfully. The reasons for the failures of N1.1, N1.2 and N1.6 were because of the lack of responsiveness of the application on the login screen of Android phones, the lack of CAS integration, and the lack of a club administrator feature being integrated. Over the next several days, the team managed to implement scrolling on Android devices and implemented club admin permissions within the application; therefore, tests N1.1 and N1.6 passed as of 2/28/17. The usability test, which still fails, test N1.2, will unfortunately not be implement by the end of this project.

7.5.3 Security Testing

Security testing is a method intended to reveal flaws in the security mechanisms of a system that protect data and maintain intended functionality. Due to the logical limitations of security testing, a system passing security testing is not an indication that no flaws. Although security testing varies for each system it is critically important that regardless of the system that the elements of confidentiality, integrity, authentication, availability, authorization, and non-repudiation are all verified and tested.

To conduct security testing our group utilized the focus group we had setup for System testing. Similar to how we conducted usability testing, our group had potential users use the app and try to complete certain tasks, which lined up with tests we had documented.

The list of tasks we asked our focus group participants to complete for security testing were as follows:

- N3.1 The system will use CAS to perform and protect user's login information.
- N3.2 The system should only permit club event planners to create events for their respective clubs.
- N3.3 Two-factor authentication should be used to initially sign up a new user.

7.5.4 Security Testing Results

Based on our initial focus group, none of our security requirements passed their respective tests. At the time of this first focus group, our team was acknowledged by WPI's Information Technology services as a prospect for CAS integration. This news encouraged the team to forgo the need for two-factor authentication and instead rely on the future enablement of CAS for login and user profile security. Therefore, following the 2/23/17 focus group, the team worked on requirement N3.2, to ensure that club administrator permissions were properly utilized in the application.

For our 2/28/17 focus group the club admin permission were properly utilized and test N3.2 passed. Unfortunately, the team heard news about that the CAS integration could fall well outside the intended timeline of our project; thus, requirements N3.1 and N3.3 will not be completed nor pass testing by the end of C-Term cutoff.

7.5.5 Performance Testing

Performance testing is the process of determining the speed or effectiveness of a system or device. This process can involve quantitative tests done in a lab, and even qualitative attributes such as reliability, scalability, and interoperability.

To conduct this testing our group utilized the focus group we had setup for System testing. Exactly how we conducted the previous tests, our group had potential users use the app and try to complete certain tasks, which lined up with tests we had created and documented.

The list of tasks we asked our focus group participants to complete for security testing were as follows:

- N2.1 The system should handle up to six (6) events made per minute.
- N2.2 The system can handle at least 50 users simultaneously.
- N2.3 The system should query search and filter results < 3 seconds, given 10 users.
- N2.4 The system should take no more than 10 seconds to execute an interaction.

7.5.6 Performance Testing Results

Based on the 2/19/17 focus group, three of our four performance tests passed their respective tests: N2.1, N2.3, and N2.4. All these tests intended to test the transaction speed of the application, and we tested these requirements by having focus group participants to perform a set of tasks in each time.

On 2/23/17, our team conducted a 75-person focus group to stress test the system. The goal of the stress test was to determine whether the application could handle a plethora of users and activity at once. The focus group was told to use the application at will and to report if any crashes occurred; luckily, no one report any crashes. Therefore, requirement N2.2 was successfully met.

7.6 Acceptance Testing

Acceptance Testing is a level of software testing where a system is tested for acceptability. The purpose of this test is to evaluate the system's compliance with the business requirements and assess whether it is acceptable for delivery.

Our team chose to conduct beta-testing as our method of acceptance testing as we hope it will inspire better feedback as testers can input real club and event information. For our beta-test, we chose to have ten (10) Interfraternity Council members act as our testers. We chose this organization because of their continuing backing of the application’s development and their ability to publicize the application to other organizations they work with.

To measure the results of our beta tester’s satisfaction, we designed a survey after the Technology Acceptance Model (TAM). The TAM provides a valid and reliable measure that predicts the acceptance or adoption of new technologies by end-users which is split into two sections: a measurement for Perceived Usefulness (PU) and Perceived Ease of Use (PEU). We incorporated the TAM by creating a survey of 10 questions that measured PU and PEU, which was distributed to our testers after having a chance to unrestrictedly use the application for a five (5) day span.

7.6.2 Acceptance Testing Results

We randomly selected four (4) beta testers to perform our TAM survey. Based on the averages of responses, we recognized that users experience a higher overall perceived usefulness of the application than perceived ease of use. The following sections comment further on the strengths and shortcomings of the application.

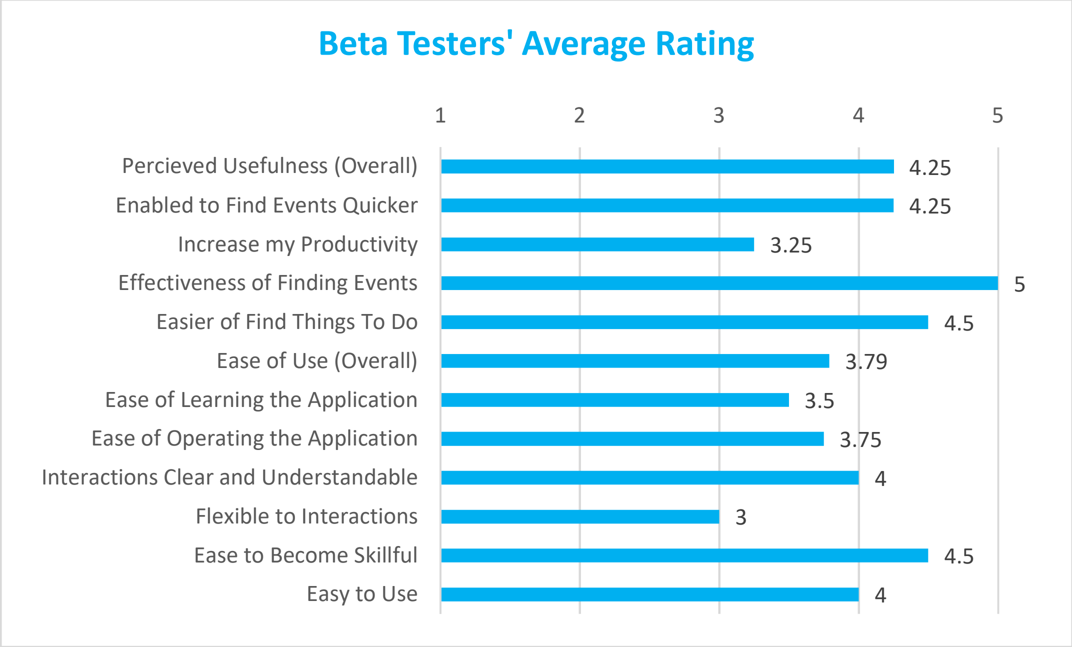


Figure 84 - TAM Beta Testers' Results

7.6.2.1 Perceived Usefulness Results

In general, the scores for our Perceived Usefulness category were above satisfactory (4 and above) to exceptional (rated a 5). Overall users found the application increased their effectiveness of finding WPI events to take part in, made it easier to find events to attend, and enabled them to find events quicker. The shortcomings of the application in this category was whether the application increased the tester's productivity. We realize that the term "productivity" may have confused beta testers as attending club events are not intrinsically a "productive" action in the perspective of WPI students.

7.2.6.2 Perceived Ease of Use Result

The Perceived Ease of Use of the application performed less desirably than the Perceived Usefulness as most average ratings fell between a satisfactory (between 3 and 4) or above satisfactory (4 and above). Beta testers rated that the application was rather easy to use, easy to understand, and easy to become skilled at using; however, testers noted that the application was not flexible to their interactions and that the initial learning curve associated with starting to use and operate the application was slightly difficult. Therefore, it may be wise in the future if tutorials were created to help new users understand the application, or that the user interface be simplified.

8. Conclusion and Recommendations

8.1 Project Conclusion

To find a solution to the lack of information and awareness about student activities on university campuses, we developed a functional web based mobile application that granted WPI students access to information regarding what activities were taking place around campus and suggest activities they would likely be interested in attending. Using a survey and interviews our team could identify the basic features needed by both the student body and campus organizations for the application we planned to develop to be valuable to them. From there, our team used scrum agile methodologies to develop a minimum viable application which allowed us to approach users to gather feedback that further improve the application. After implementing the discovered features, we gathered more user feedback so we could create a more complete, fully-functioning application. Lastly, through a series of focus groups and testing our team evaluated the application in terms of its functionality and student satisfaction. To build a more comprehensive and effective application, our team has developed several recommendations.

8.2 Recommendations

Concerning the continuation of this project, there are some specific milestones our team encourages for future success and continuation.

First, we highly recommend following up with WPI Information Technology Department to ensure the application can utilize the Central Authentication System (CAS). This is the most pertinent recommendation we have as failure to have WPI IT's consent to operate the application on campus will result in their closure of the application entirely. By working with IT and having them help with CAS integration will result in more potential success for the application and a more secure measure to protecting WPI student information.

Next, we strongly encourage utilizing the Student Activities Office (SAO) and student organizations, such as the Student Government Association, Interfraternity Council, Panhellenic Council, and SocComm to collaborate with the launch of the application. These organizations have tremendous oversight and influence with many WPI clubs and the student population. The process to onboard the student organizations to use and promote the application is already in progress; however, we believe it is crucial to formalize relations with the SAO to make the launch of the application more successful.

Finally, we recommend conducting a feasibility study about implementing the application at other universities, especially in the Worcester Consortium. The application was made to scale to multiple universities, and given the situation at WPI there are other smaller universities in need of a reliable university events application. Once more features have been implemented, either those that we listed in our Product Backlog or those discovered over user focus groups, we encourage the investigation on whether the application addresses the needs of other universities.

In terms of specific features to implement, our team recommends developing the features based on the user stories found in our Product Backlog, located in *Section 4.5 - User Stories*. Each feature was given as feedback by users in our focus groups and interviews.

8.3 Our Interdisciplinary Approach

Our MQP embodies an interdisciplinary approach as it is the culmination of computer science and business analyst methodologies and mentalities. Instead of solely focusing on writing well-designed code or executing system improvements to a given company's operations, we had to perform both; we merged CS and MIS practices which enabled us to discover an opportunity that technology could improve for our chosen stakeholders, develop application requirements rooted in user feedback, and produce a useable product that addresses our stakeholders' needs. To perform such a project, our team adopted agile scrum methodologies, based off current software company practices, and catalyzed our project with an initial requirements-gathering and project definition phase (architectural spike), as utilized in more traditional software development life cycle methodologies. The merging of these methodologies improved the execution of our deliverables and enhanced our group's cross-functionality. By creating a more holistic view of the software development process within our project, our team comprehended the full extent of how to accomplish our deliverables and their importance, enabling us to go beyond the necessary skills of our majors and understand the project from various roles.

8.4 Reflections

8.4.1 Tyler Bennett

MQP has been a key component to my learning experience at WPI. This three-term project taught me invaluable concepts moving forward in my Computer Science career. Through working with a team composed of multiple majors, I could gain both technical and team communication experience.

I am proud of what our team could accomplish. I think that one of the biggest takeaways from this project was our team's ability to stick to the agile scrum methodology to produce deliverables to our advisors. We used sprint planning meetings, daily standups, and sprint retrospective meetings to constantly inspect our progress while at the same time moving forward as a team. I was also happy with how our team managed to stay positive through the entirety of the three terms. Things did not always go as planned, and I learned it is key to stay positive and work with what cards are given to you. Finally, we could constantly deliver upon the features our stakeholders wanted to see implemented. This was due to the accountability and communication present within our team.

Personally, this project was very important for me because it allowed me to be immersed in major related activities almost every day for a three-term period. I feel much more confident and prepared to journey into the computer science field now that I have MQP under my belt. This project has influenced me to further pursue app development. I discovered that applications can be a great place for both art and development to coexist, two things I have a passion for.

I believe this project has potential to be very successful. There is still work to be done, but this project fulfills a need that is missing at Worcester Polytechnic Institute, and most likely other schools as well. I look forward to seeing the impact of our findings and work.

8.4.2 Brandon Malofsky

Our project brought numerous experiences and lessons to benefit myself and career moving forward after WPI. I obtained greater insight on the necessary skills and culture required for a successful team to operate effectively. Whether that was seen from simply person to person or how our team must operate to stakeholders to gain trust and support for our product. However, looking back onto the project I also saw the importance of long term planning beyond idealistic goals. Our team saw numerous times assumptions we made be broken or a new standard be set, or simply outlying blockers making us pivot our application's MVP. I learned that you must never forget to account for the unknown no matter if it is a personal objective to a large-scale project. An operation or a team can only move smoothly when you are prepared to deal with the unknown alongside the known. Furthermore, I saw myself gain a greater understanding of how to develop a mobile application, testing the product, defining stakeholders and customers, and seeing the requirements of bringing a product to a potential market. I found myself truly enjoy understanding the challenges that come from an applications development where previously my knowledge was limited from a technical standpoint. Perhaps the most important concept I was reminded numerous times of its value was communication within the team. When you are creating, an application using Agile Development communication is paramount. When someone is not communicating effectively all team members cannot work effectively and you can quickly fall behind. Any project or team I work alongside in the future I will always reinforce communication and honesty between members and stakeholders. If you stay honest and always remain in contact, you will earn trust and find your team will work many times more passionately and effectively then if you had not.

8.4.3 Matthew McCarthy

MQP was a turning point in my confidence as a computer scientist and programmer. Because of the design of the WPI Undergraduate curriculum the four seven week terms in place of semesters, depth in each subject is limited to the capability of professors to pack as much information as possible in such a short period of time. Although these seven week terms keep us focused and accelerate students through the curriculum, a key learning tool is sometimes lost on Computer Science students; students need experience working on projects for extensive periods of time, beyond the standard seven week term, especially projects that involve large scale software development, since most if not all students graduating from the Institution as Computer Science majors will be working for companies whose timelines extend beyond a seven week sprint.

This is where MQP falls into place as a critical aspect of the WPI Plan, for students majoring in Computer Science. For many, including myself, it is the first experience we have working towards a large project extending over three terms. I learned things through this project that I never would have been able to in a classroom.

First, I learned how to learn. Building this web application from scratch required an extraordinary amount of research, without the resources typically available for students such as T.A. office hours and resources provided from professors. No one was there to hold our hands as we developed this application. I learned how to discover what I don’t know and use my own resources to gain the knowledge I needed to proceed.

Second, I learned a greater appreciation for patience. There were many moments of seemingly perpetual frustration, but at the end of the project everything was resolved. No matter how big the problem, or how elusive the bug, we discovered a solution. Unlike a project for class, this wouldn’t go away in a few weeks. If there were issues, they needed to be resolved and they needed to be resolved quickly so that other members of the team could accomplish what they needed.

Finally, I discovered a confidence in myself as a developer and a reassurance that despite trials and tribulations, countless hours of frustration, and last minute panic when we broke the application thirty minutes before our meeting, I can get the job done. I value this the most as I prepare to enter the workforce. WPI, through its project based curriculum, prepares us for the next step. I’m grateful for the opportunity to work with my team members and complete a project that took us over six months from start to finish.

8.4.4 Zahr Lyttle

During the entirety of this project I learned a lot more than I thought I would. We had many experiences we had not originally planned for, but I believe it allowed each of us to grow and learn valuable skills for the future. Specifically, I learned about project management and how to be a part of a successful team, how to develop a mobile application using specific set requirements, and how to identify user/consumer needs and translating them into a product.

I learned that to be part of a successful team and to be an effective project manager it is critically important to be straight forward, consistent, and accountable. Working with my partners for so long, I feel we all realized that although we were all friends that we needed to adopt a certain standard to hold ourselves by while we were working on the project. Learning to display these things day in and day out for 7 months essentially ingrained in me the importance of being a good partner/co-worker to reach goals and accomplish things with a team.

After this I think the next thing I learned was exactly how to develop a mobile application in an environment where there are so many routes and options when doing so. I learned an immense about the technical aspects of an app, how to effectively test an application, and to use code to solve an actual human issue.

Another thing I learned was how to identify user needs and translating that to into a product that solves a problem and adds value to people's lives. There were difficulties but I found great satisfaction in running into problems and gathering with the team to decide how we felt we could best solve the issues so we can hit our goals. Before this project, I felt I had basic knowledge of app development and working in a professional team setting, but after completing the project I feel so much more comfortable and experienced not only discussing these topics but taking on work like this again. From here, for projects I undertake or teams I work on in the future I will always push for communication and accountability among team members and stakeholders. I find that working hard with a group unified by an idea or vision coupled with good group practices is key for the success of any team.

8.4.5 Johnny Ross

The MQP experience has been the most educational and rewarding experience I have had at WPI. This project not only helped me realize my potential as a business analyst, but also as programmer and professional.

In terms of growing as a business analyst, MQP helped me recognize and develop my skills of being able to identify stakeholder needs, conduct focus groups, interviews, and meetings, and develop functional and non-functional requirements based on user feedback. It was these areas which I discovered I most enjoyed spending my time; reaffirming that I made the right choice in career as a business analyst.

Concerning programming, I learned a great amount about software engineering methodologies and front-end development and became more confident in my abilities to learn more in the area. This MQP was my real experience with practicing scrum methodologies and using tools such as Git for group coding functionality. Additionally, it was my first time dealing with AngularJS and was by far the most extensive project I worked on in terms of web development. Despite not having to partake much in the coding of this application, I still found myself applying the programming knowledge I obtained from this project and expanded upon it in my other course work and independent projects.

Lastly, there are several key lessons I learned as a young professional. The first would have to be to communicate early and often, especially with your team. There were times where our team overcame uncertainty, navigated roadblocks, and assisted each other to complete tasks because we maintained honest communication with each other and set the precedent that we are all accountable for the tasks the group needs to accomplish. Next, I realized how easy it is to bite off more than you can chew or overpromise. Many times, our group discovered that certain tasks and objectives were just too large to feasibly accomplish in the given time period; it takes much honesty, communication, and empathy to accurately overcome the “we-can-do-it-all” mentality.

I am so fortunate to have been part of such a phenomenal team and to have learned as much as I have. This project certainly has reinvigorated me to pursue a career that blends business with technical solutions and I am excited to bring my acquired experience into my next endeavor.

9. Bibliography

Astin, A. (1999, September). Student Involvement: A Developmental Theory for Higher Education. *Student Involvement: A Developmental Theory for Higher Education, 40*(5), 1-529. Retrieved October 4, 2016, from http://www.ydae.purdue.edu/lct/hbcu/documents/Student_Involvement_A_Developmental_Theory_for_HE_Astin.pdf

"All About Data Flow Diagrams (DFDs)." *Lucidchart*. Lucidchart, 15 Nov. 2016. Web. 29 Nov. 2016.

Atlassian Documentation: Burndown Chart. (n.d.). Retrieved on October 07, 2016a. from

Bauer, S. (2015, June/July). Students' Mobile Learning Practices in Higher Education: A Multi-Year Study. Retrieved October 10, 2016, from http://er.educause.edu/articles/2015/6/students-mobile-learning-practices-in-higher-education-a-multiyear-study

Beedle, Mike. *Manifesto for Agile Software Development*. N.p., 2001. Web. 30 Nov. 2016.

Bristowe, John. (2016, July 6). *What Is a Hybrid Mobile App? | Telerik Developer Network.* Retrieved October 7, 2016, from

By releasing your mobile app early and often, and gathering metrics for quick iterations, your customers get what they want and need. (n.d.). Adobe PhoneGap. Retrieved October 10, 2016, from http://phonegap.com/products/

Campus Labs. (n.d.). About Us. Retrieved October 20, 2016, from http://www.campuslabs.com/about-us/

Campus Labs. (n.d.). Corq by CollegiateLink (0.9.9) [Mobile application software. Retrieved from https://play.google.com/store/apps/details?id=com.campuslabs.collegiatelink&hl=en

Cloud Application Platform | Heroku. (n.d.). Retrieved October 10, 2016, from http://heroku.com/ … Heroku Inc.

"Creating a Sprint." *Atlassian Documentation*. Atlassian, n.d. Web. 30 Nov. 2016.

Guidebook. (n.d.). College and University Mobile Apps. Retrieved October 20, 2016, from https://guidebook.com/schools/

Cooper, A. (1999). *The inmates are running the asylum: Why high-tech products drive us crazy and how to restore the sanity* (Indianapolis, Indiana: SAMS).

Daily Digest. (personal communication, September 14, 2016)

Duggan, M. (2015, August 19). *The Demographics of Social Media Users.* Retrieved October 04, 2016, from http://www.pewinternet.org/2015/08/19/the-demographics-of-social-media-users/

Epics, Stories, Versions, and Sprints. (n.d.). Retrieved on October 07, 2016b from

Foubert, J. D., & Urbanski, L. A. (2006). Effects of Involvement in Clubs and Organizations on the Psychosocial Development of First-Year and Senior College Students. *Journal of Student Affairs Research and Practice, 43*(1). doi:10.2202/1949-6605.1576

Gaebel, D. (2015, July 30). *The Battle Royale: Atom vs. Sublime - Web Design Weekly.* Retrieved October 10, 2016, from https://web-design-weekly.com/2015/07/30/atom-vs-sublime/

Guidebook Inc. (n.d.). Guidebook (4.22.0) [Mobile application software]. Retrieved from https://play.google.com/store/apps/details?id=com.guidebook.android&hl=en

Gulliksen J., Goransson, B., Boivie, I., Blomkvist, S., Persson, J. & Cajander, A. (2003). Key Principles for user-centred systems design, Behavior & Information Technology, 22:6, 397-409.

Gulliksen, J. and Goransson, B. (2001). Reengineering the systems development process for user centered design. In Michitaka Hirose (ed.) *Proceedings of INTERACT 2001*(Amsterdam: IOS Press).

Heroku vs. Amazon Web Services. (2012, June 4). Retrieved October 5, 2016, from http://smashingboxes.com/blog/heroku-vs-amazon-web-services ...was Smashingboxes

Holtzblatt, Karen (n.d.). Contextual Design. *Interactive Design Foundation.* Retrieved November 7, 2016 from https://www.interaction-design.org/literature/book/the-encyclopedia-of-human-computer-interaction-2nd-ed/contextual-design

How to Break a User Story into Tasks. Software and I. (25 Nov. 2011). Retrieved on October 7, 2016 from

JIRA Core Licensing and Pricing. (n.d.). Retrieved October 10, 2016, from https://es.atlassian.com/licensing/jira-core

JIRA Software vs. Trello Comparison | Atlassian. (n.d.). Retrieved October 10, 2016, from https://www.atlassian.com/software/jira/comparison/jira-vs-trello … was A?

Kujala, S., Vainio, K., Karapanos, E., & Sinnelaa, A. (2011). *UX Curve: A Method for Evaluating Long-Term User Experience* (pp. 473-483).

Mobile Apps Center. (n.d.). U-M Apps. Retrieved October 20, 2016, from http://mobileapps.its.umich.edu/um-apps

Native vs Hybrid / PhoneGap App Development Comparison | Comentum. (n.d.). Retrieved October 07, 2016, from

Nielsen, J. (1993). Usability Engineering (Cambridge, MA: AP Professional).

OrgSync. (n.d.). OrgSync (1.0.4) [Mobile application software]. Retrieved by https://play.google.com/store/apps/details?id=com.orgsync.orgsync&hl=en

OrgSync. (n.d.). Your Campus Engagement Network. Retrieved October 20, 2016, from http://www.orgsync.com/what_is_orgsync

Radigan, D. (n.d.). *Have We Met?* Retrieved on October 7, 2016a from

Radigan, D. (n.d.). *Scrum.* Retrieved on Oct 7, 2016b from

Radigan, D. (n.d.). *Three Ingredients for Great Software Releases.* Retrieved on October 7, 2016c from

Radack, S. (2009, April 01). THE SYSTEM DEVELOPMENT LIFE CYCLE (SDLC). Retrieved November 30, 2016.

Rouse, Margaret. (n.d.). *What Is Native App? - Definition from WhatIs.com.* Retrieved October 7, 2016, from

Rouse, M. (2007) *What Is Use Case? - Definition from WhatIs.com.* Retrieved on October 07, 2016, from

Ruby on Rails IDE :: JetBrains RubyMine. (n.d.). Retrieved October 10, 2016, from https://www.jetbrains.com/ruby/ … Jetbrains

Saddington, P. (2013). *The Agile Pocket Guide: A Quick Start to Making Your Business Agile Using Scrum and beyond*. Hoboken, NJ: John Wiley & Sons.

Schneider, G., & Winters, J. P. (2001). *Applying Use Cases: A Practical Guide, Second Edition*. N.p.: Addison-Wesley Professional.

Section 6. Conducting Focus Groups. (n.d.). Retrieved October 10, 2016, from http://ctb.ku.edu/en/table-of-contents/assessment/assessing-community-needs-and-resources/conduct-focus-groups/main

Student Involvement Means Success All Around. (2009, June 04). Retrieved September 26, 2016, from http://blog.orgsync.com/2009/student-involvement-means-success-all-around was H.

Techopedia. (n.d.). What is an Entity-Relationship Diagram (ERD)?. Retrieved December 09, 2016, from https://www.techopedia.com/definition/1200/entity-relationship-diagram-erd

Trello. (n.d.). Retrieved October 10, 2016, from http://trello.com/

UX Design Defined. (n.d.). *User Design: Humanizing Interaction*. Retrieved November 7, 2016 from http://uxdesign.com/ux-defined

Wells, Don. "Extreme Programming: A Gentle Introduction." *Extreme Programming: A Gentle Introduction.* N.p., 8 Oct. 2013. Web. 30 Nov. 2016.

What is Scrum? (n.d.). Retrieved January 20, 2017, from https://www.scrum.org/resources/what-is-scrum

University of Michigan. (n.d.). University of Michigan (3.7) [Mobile application software]. Retrieved from https://play.google.com/store/apps/details?id=edu.umich.michigan&hl=en

Appendix A: Club Presidents Focus Group- Question Prompt

Anticipated Questions:

- How effective are your current methods of advertising events on campus?
    - How effective is using OrgSync to advertise your events?
        - What features/characteristics do you like about OrgSync? Why?
        - What aspects of OrgSync do you dislike? Why?
    - How effective is Facebook at advertising your events?
        - What features do you like about Facebook? Why?
        - What aspects of Facebook do you dislike? Why?
    - In addition to OrgSync and Facebook, are there other platforms you use to advertise events?
- What would you want an Event Planner App to do for you?
    - What features must it have to be attractive to your club/organization? To students?
        - Why are these things "Must Haves?"
    - What do you envision are the potential roadblocks of getting campus to use this app?

Competitor Analysis:

- What do you like about OrgSync?
    - What is it you do not like about Orgsync?
- What could OrgSync do better for you?
    - What is it missing?
- How do you and your Club feel about OrgSync?
    - Would a streamlined mobile App be more appealing?
    - Do you and your officers enjoy using OrgSync?
- Do you use alternate methods to schedule and communicate?
    - If yes, what do you use and why?
    - If no, what makes you remain using OrgSync verse another product?

Feature Prioritization Questions:

- What would you want an Event Planner App to do for you?
    - What must it have to be attractive?
        - Why are these things "Must Haves?"

- What features does OrgSync have you like?
    - What features are they missing?
    - What is difficult about using it?
- What Features do you want in a scheduling application?
    - What are the most attractive?
        - Why are they appealing to you?

Appendix B: General Users Interview - Question Prompt

Anticipated Questions:

- Is finding events on campus difficult?
    - Do you find the current methods of advertising helpful to finding and knowing about events at WPI?
        - What methods work best for students like you?
            - Social Media?
            - Org Sync?
            - Physical Media?
        - What methods do you not like? Why? Are they more difficult?
    - Do you find yourself only attending club events you are already affiliated with?
        - Are finding other organization's events hard?
    - Are there other forms of advertising you find best?
- If you could change event advertising today what would you do?
    - What wouldn't you change?
    - What would you change?
        - Why are these things "Must Haves?"
    - What do you envision are potential roadblocks of getting campus to use this app?

Event Planner App:

- What would make you use an Event Finding/Planner App?
    - Why are these things "Must Haves?"
- How do you envision this App working?
    - How would you visualize / see looking for events?
    - How do you imagine finding about events?
        - Filtering Content?
        - Push Notifications?
        - Discovering New Events?
- What do you envision are the potential roadblocks of getting campus to heavily adopt or use this app?
- Would you prefer an Event Planner App to be visually more list or map oriented, or both?
    - We have several Mock-Ups for you, and would love your thoughts?
        - What do you not like?
        - What do you like?
        - What can we improve?

Feature Prioritization Questions:

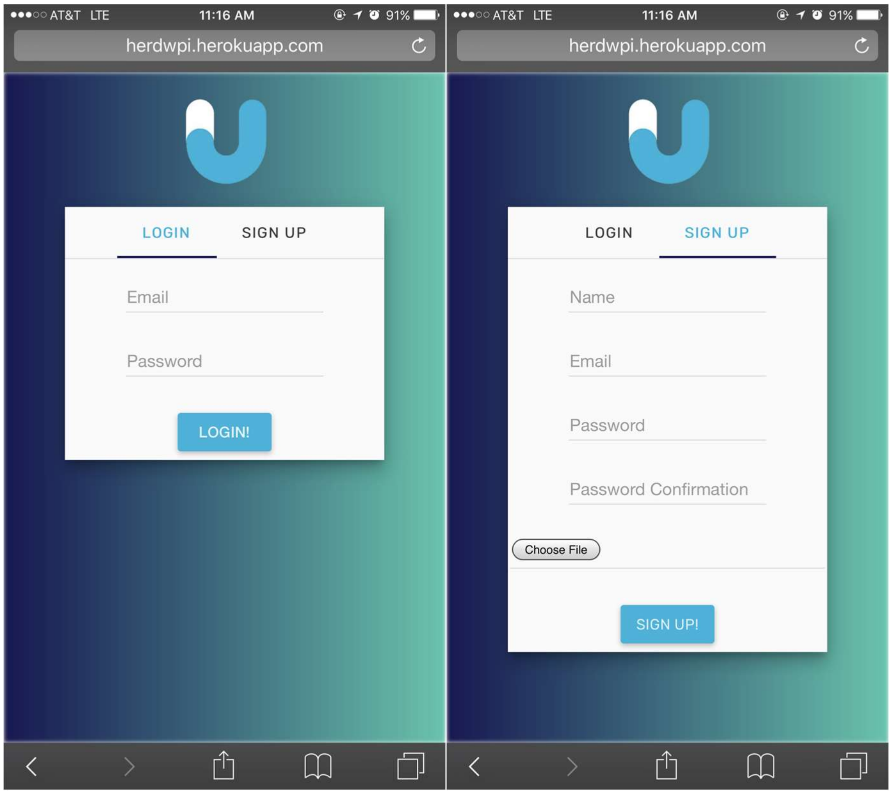
- What would you want an Event Planner App to do for you?
  - What must it have to be attractive?
    - Why are these things “Must Haves?”
- What Features do you want in a scheduling / event locater application?
  - What are the most attractive?
    - Why are they appealing to you?

Appendix C: Application User Manual

JoynU – Campus Event Finder Application

1. Setting up JoynU
    a. Creating an account
    b. Signing Up
2. The Main page
    a. Navigation
3. Events
    a. Components
4. The Drawer Menu
    a. View Club Page
    b. Creating an Event
    c. Creating a Club
    d. Change Password
    e. Log Out
5. The Club Page
    a. Subscribing to a Club
    b. Editing a Club Page
6. The Search Page
    a. Searching for a Club

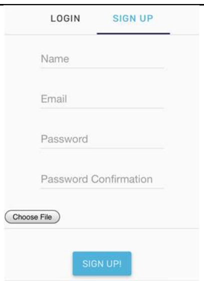
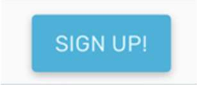
1. Setting up JoynU



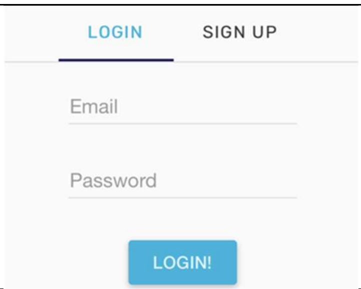
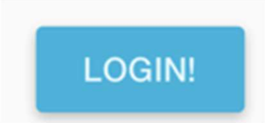
Overview:

The login/sign-up screen is the first thing users will each time they launch the application/go to the applications link. The application will prompt you with options to either login or sign-up via the use of two tabs near the top of the screen.

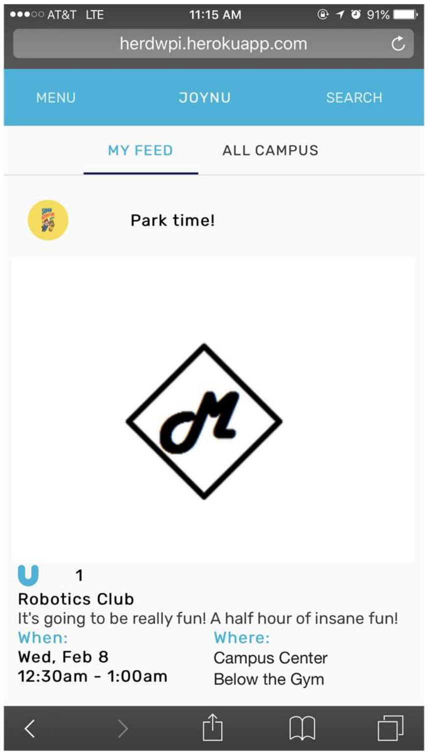
Creating an Account

1. Fill in all of the fields on the sign-up section of the login/sign-up screen. Users must use a '@wpi.edu' email address for this process to be successful.	 LOGIN SIGN UP Name Email Password Password Confirmation Choose File SIGN UP!
2. After all the fields are filled in, users simply hit the 'Sign Up!' button and the process of creating an account should be complete	

Signing Up

1. Fill in all of the fields on the login section of the login/sign-up screen using the info previously used to create an account.	
2. After all the fields are filled in, users simply hit the ‘Login!’ button and the process of signing in should be complete	

2. The Main Page



Park time!

1

Robotics Club

It's going to be really fun! A half hour of insane fun!

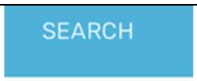
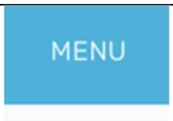
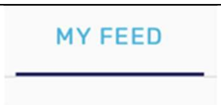
When:
Wed, Feb 8
12:30am - 1:00am

Where:
Campus Center
Below the Gym

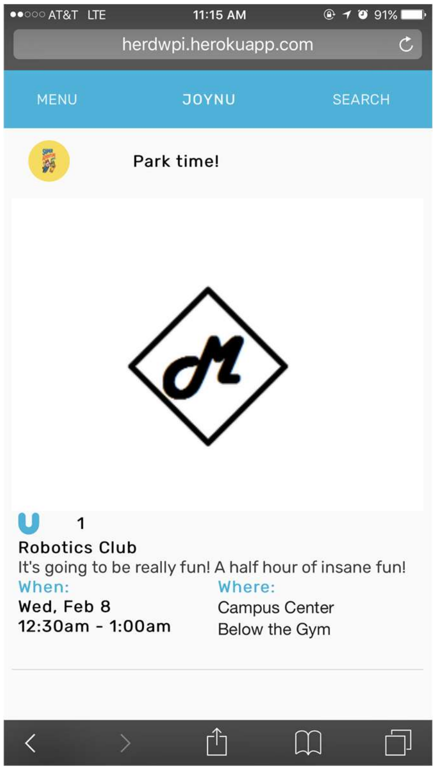
Overview:

The main page is the first thing users will see each time they login or create an account on the application. From this main page users can access the search page, the drawer menu, and the ‘My Feed’ and ‘All Campus’ tabs.

Navigation

1. To get to the ‘Search page’ users simply have to tap the ‘Search’ button at the top right of the screen. This screen is where users can search for clubs.	
2. To get to the ‘Drawer Menu’ users simply have to tap the ‘Menu’ button at the top left of the screen. This screen is where users can view upcoming events they are interested in, clubs they are subscribed to, buttons to create events and clubs (depending on the users privileges), and log out.	
3. To get to the ‘My Feed’ tab, users simply have to tap the ‘My Feed’ button at the top middle of the screen. This screen is where users can see events posted by clubs they are subscribed to.	
4. To get to the ‘All Campus’ tab, users simply have to tap the ‘All Campus’ button at the top middle of the screen. This screen is where users can see events posted by all clubs on campus.	

3. Events



Park time!

1

Robotics Club

It's going to be really fun! A half hour of insane fun!

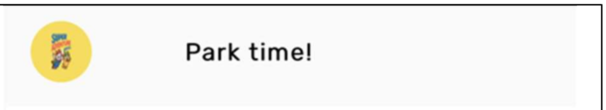
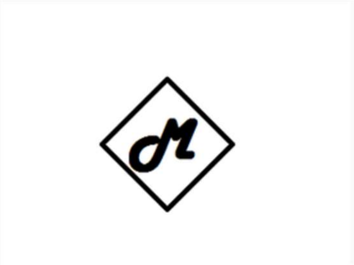
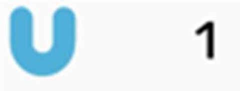
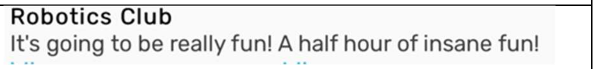
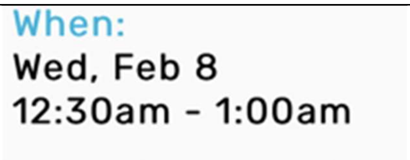
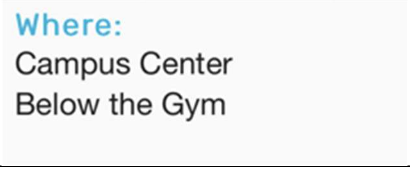
When:
Wed, Feb 8
12:30am - 1:00am

Where:
Campus Center
Below the Gym

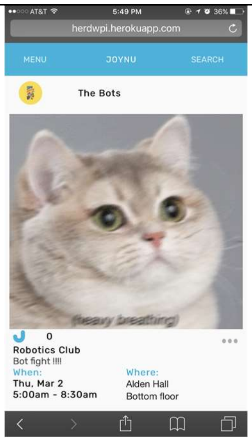
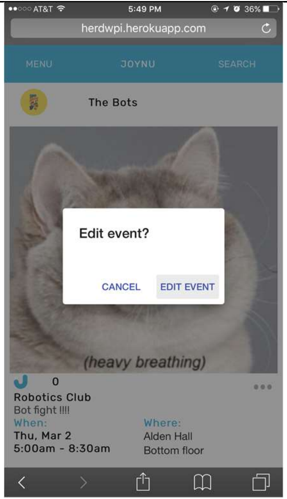
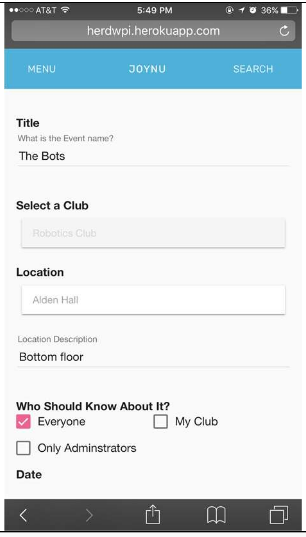
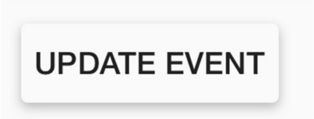
Overview:

The above image is a display of how events are shown on the application. A posted event contains a few key components: an event title, an image for the event, a button to indicate if a user is planning to attend the specific event, the club that posted the event & event description, info of when the event is taking place, info on where the event is taking place.

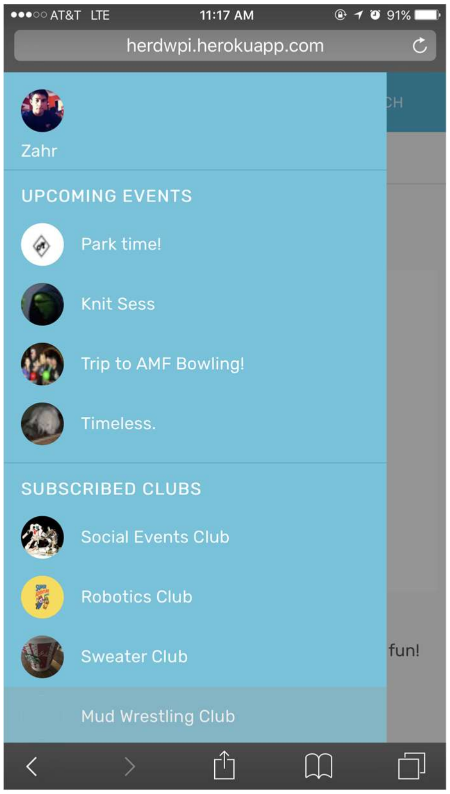
Components

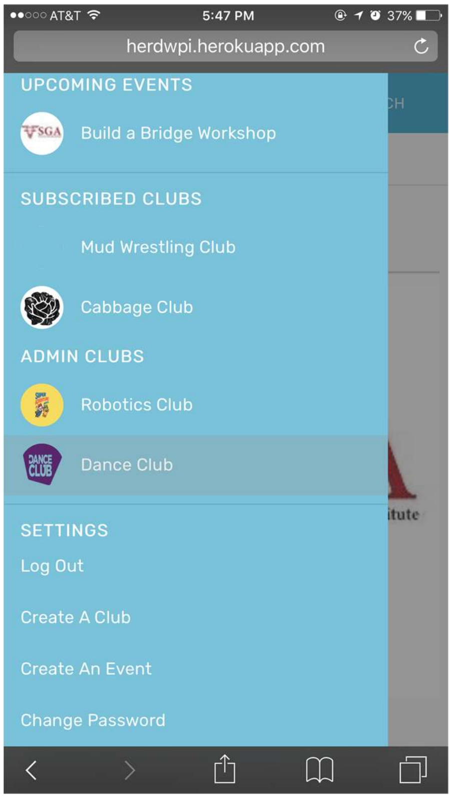
1. When looking at a posted event, the ‘Event Title’ is located at the top of the event post.	 Park time!
2. When looking at a posted event, the ‘Event Image’ is located in the middle of the event post and takes up much of the posts screen real estate.	
3. When looking at a posted event, the ‘JoynU Button’, which indicates if a user is planning to attend the specific event, is located directly the to the bottom left of the ‘Event Image’.	 1
4. When looking at a posted event, the ‘Club Name & Description’ of the club posting the event is located directly under the ‘JoynU Button’.	 Robotics Club It's going to be really fun! A half hour of insane fun!
5. When looking at a posted event, the information of when the event is taking place is located at the bottom left of the event post.	 When: Wed, Feb 8 12:30am - 1:00am
6. When looking at a posted event, the information of where the event is taking place is located at the bottom right of the event post.	 Where: Campus Center Below the Gym

Editing an Event

1. Once a user is on a specific event and has administrative privileges for the associated club, then they may edit the posted event information. To do this the user must tap the three grey dots on the posted event which will take them to a pop up screen.	 herdwpi.herokuapp.com MENU JOYNU SEARCH (heavy breathing) The Bots 0 Robotics Club Bot fight !!!! When: Thu, Mar 2 5:00am - 8:30am Where: Alden Hall Bottom floor
2. On this pop up screen a user needs to tap the 'Edit Event' button to proceed to a new screen where the user will have the ability to edit the event information.	 herdwpi.herokuapp.com MENU JOYNU SEARCH (heavy breathing) CANCEL EDIT EVENT Edit event? The Bots 0 Robotics Club Bot fight !!!! When: Thu, Mar 2 5:00am - 8:30am Where: Alden Hall Bottom floor
3. Once on this new screen the user has the ability to edit the information fields for the event.	 Title What is the Event name? The Bots Select a Club Robotics Club Location Alden Hall Location Description Bottom floor Who Should Know About It? ☑ Everyone ☐ My Club ☐ Only Adminstrators Date
4. After the user updates the information fields they must hit the 'Update Event' button, and then the posted events information will reflect the users changes.	

4. The Drawer Menu

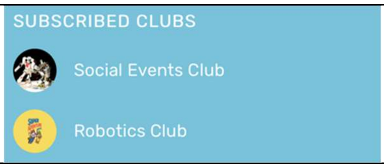




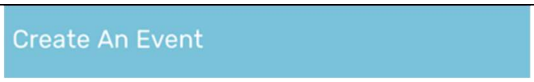
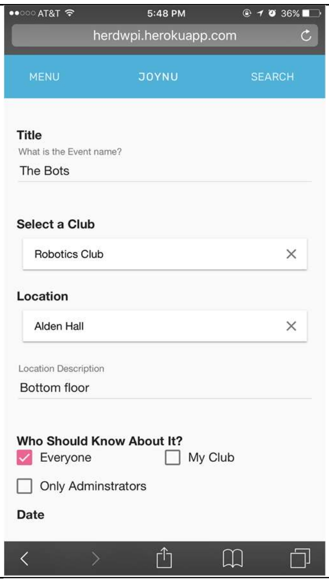
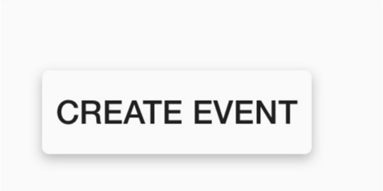
Overview:

The Drawer Menu is what users see after hitting the ‘Menu’ button from the ‘Main Page’ screen. From the Drawer Menu users can view upcoming events they’ve indicated they would be attending, view their subscribed clubs, have access to the ability to log out, have access to the ability to change their password, have access to create events (if the user has proper permissions), and have access to create clubs (if the user has proper permissions).

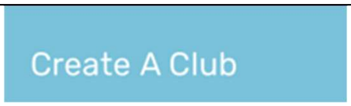
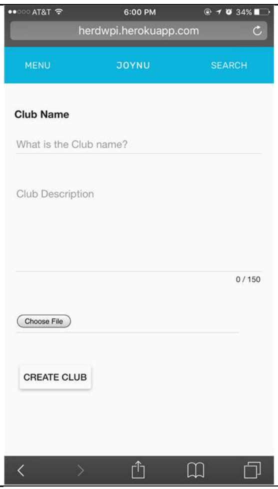
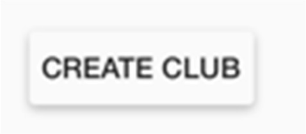
View Club Page

1. To view a club’s page from the Drawer Menu, a user simply needs to tap on a club’s name under the ‘Subscribed Clubs’ section.	

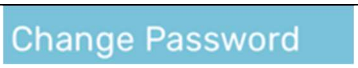
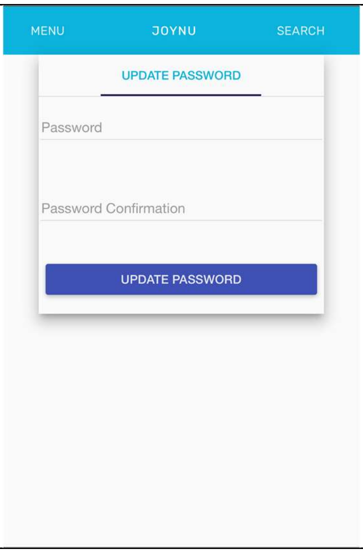
Creating an Event

2. To create an event first a user must tap the 'Create an Event' button in the Drawer Menu if they have administrative privileges for their club.	
3. The next step is to fill out the information fields that the application presents in order to create the event.	 ●●○○○ AT&T 5:48 PM 36% herdwpi.herokuapp.com MENU JOYNU SEARCHTitle What is the Event name? The Bots Select a Club Robotics Club × Location Alden Hall × Location Description Bottom floor Who Should Know About It? ☑ Everyone □ My Club □ Only Adminstrators Date
4. Once all fields are filled in, a user simply needs to hit the 'Create Event' button at the bottom of the page in order to have the application take the information from the page and create the event.	

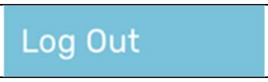
Creating a Club

1. To create a club first a user must tap the ‘Create a Club‘ button in the Drawer Menu if they have administrative privileges for their university.	
2. The next step is to fill out the information fields that the application presents in order to create the club.	 AT&T 6:00 PM 34% herdwpi.herokuapp.com MENU JOYNU SEARCH Club Name What is the Club name? Club Description 0 / 150 Choose File CREATE CLUB
3. Once all fields are filled in, a user simply needs to hit the ‘Create Club’ button in order to have the application take the information from the page and create the club.	

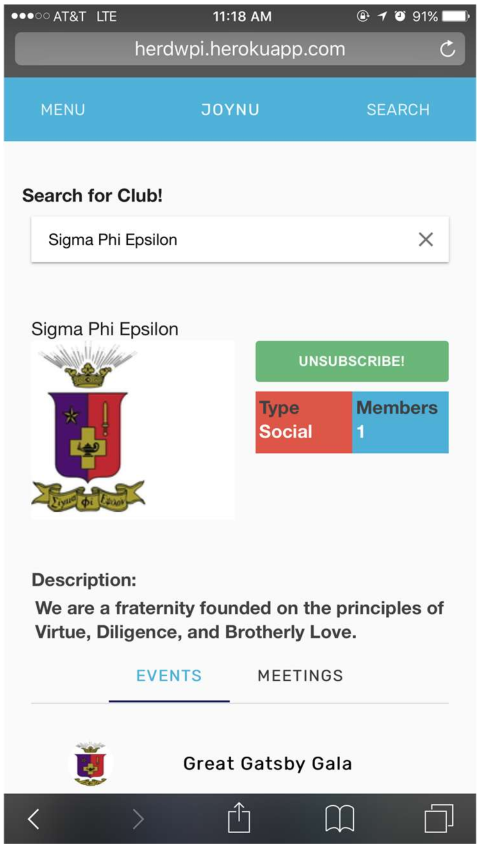
Change Password

1. To change a users password, they must first tap the 'Change Password' button in the Drawer Menu. Doing this will take the user to a page to change the password.	
2. Once on the screen to change the password for the account, all a user needs to do is type the new password twice to confirm, then tap the 'Update Password' button to officially change it.	 MENU JOYNU SEARCH UPDATE PASSWORD Password Password Confirmation UPDATE PASSWORD

Log out

1. To logout all a user needs to do is tap the 'Log out' button in the Drawer Menu.	

5. The Club Page



Search for Club!

Sigma Phi Epsilon

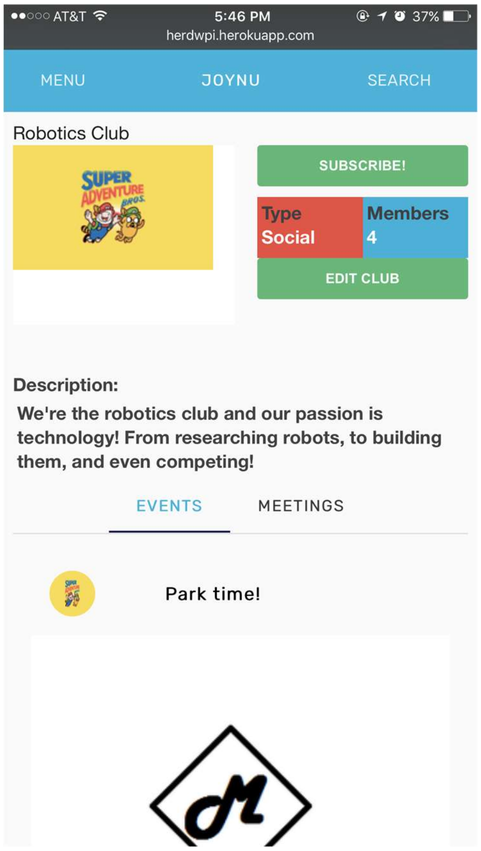
Sigma Phi Epsilon

Type	Members
Social	1

Description:

We are a fraternity founded on the principles of Virtue, Diligence, and Brotherly Love.

Great Gatsby Gala



Robotics Club

Type	Members
Social	4

Description:

We're the robotics club and our passion is technology! From researching robots, to building them, and even competing!

Park time!

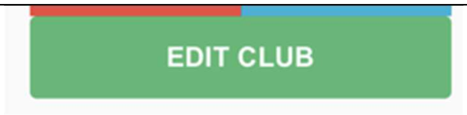
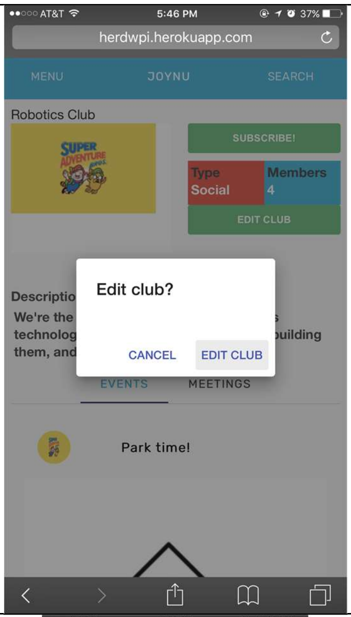
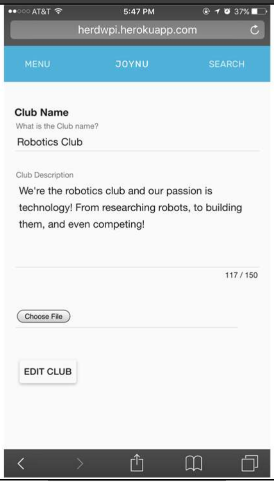
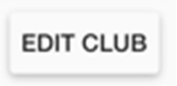
Overview:

A Club Page is the page for each individual club on the application which allows users to subscribe or unsubscribe to a club, view information about the club, and edit information about the club (if user has the appropriate privileges).

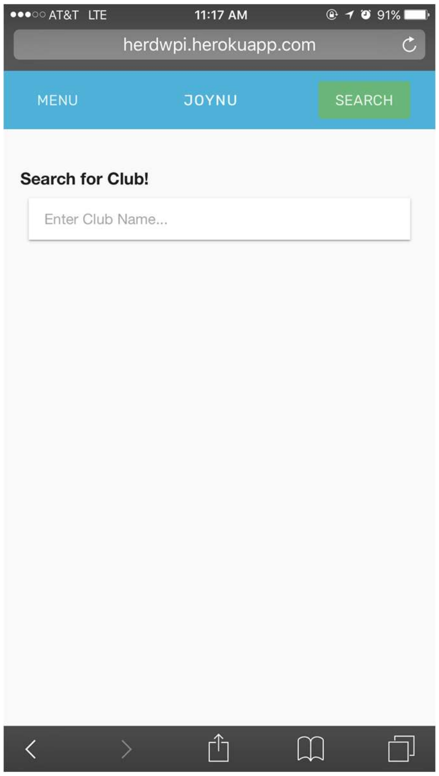
Subscribing/Unsubscribing to a Club

1. Once a user is on a club page, a user simply needs to tap the 'Subscribe' button on the respective club's page to subscribe to the club.	SUBSCRIBE!
2. Once a user is on a club page, a user simply needs to tap the 'Unsubscribe' button on the respective club's page to unsubscribe to the club.	UNSUBSCRIBE!

Editing a Club Page

1. Once a user is on a specific club page and has administrative privileges for the club, then they may edit the club pages information. To do this the user must tap the three grey dots on the club page which will lead them to a pop up screen.	
2. On this pop up screen a user needs to tap the 'Edit Club' button to proceed to a new screen where the user will have the ability to edit the club information.	 AT&T 5:46 PM 37% herdwpi.herokuapp.com MENU JOYNU SEARCH Robotics Club SUBSCRIBE! Type Social Members 4 EDIT CLUB Descriptio We're the technolog them, and uilding EVENTS MEETINGS Park time! Edit club?
3. Once on this new screen the user has the ability to edit the information fields for the club.	 AT&T 5:47 PM 37% herdwpi.herokuapp.com MENU JOYNU SEARCH Club Name What is the Club name? Robotics Club Club Description We're the robotics club and our passion is technology! From researching robots, to building them, and even competing! 117 / 150
4. After the user updates the information fields they must hit the 'Edit Club' button, and then the club pages information will reflect the users changes.	

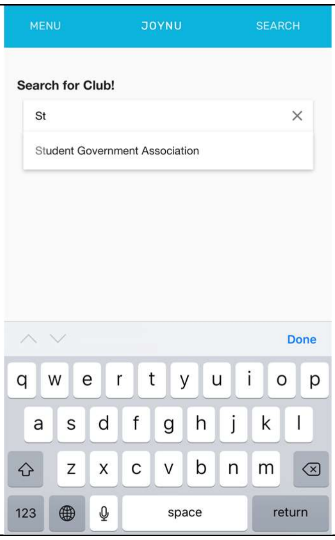
3. The Search Page



Overview:
The Search Page is the page in which a user can search for a specific club.

Searching for a Club

1. Once a user is on the Search Page, they have the ability to use the input field to type in the name of a club they wish to find. This field has autofill capabilities, once the desired club name appears a user simply needs to tap the name for the club page to be displayed.



Appendix D: Project Poster

Campus Event Finder Application

Tyler Bennett (CS), Zahr Lyttle (MIS), Matthew McCarthy (CS),
Brandon Malofsky (BUS), Johnny Ross (MIS)
Advisors: Professor Diane Strong (FBS) and Professor Wilson Wong (CS)

Abstract

The Tailored Event Finder is an application designed for both students looking for events and event organizers on the WPI campus. The application gives both students and event organizers a central place to focus their efforts in promoting and finding events. Students can filter events based on their interests and club memberships and organizers can target events at different demographics. In addition to event promotion, the application provides methods to indicate attendance at events giving organizers metrics to expect at events. This paper tracks the application's development as well as requirements gathering and testing.

Objectives

1. Identify the base requirements needed by both the student body and campus organizations.
2. Develop an Event Finder application that allows students to view what WPI-related activities are occurring.
3. Improve and test the application using agile methodologies.
4. Implement beta-testing of the application with WPI undergraduates and analyze their responses.
5. Determine recommendations for the application's continued development.

Background

With over 200+ clubs, organizations, and departments vying for the attention of WPI undergraduates to attend their events. WPI's current system, Campus Lab's OrgSync, has been under-serving the needs of club event planners and undergraduate students. Lacking intuitive event feeds, menu navigation, and content organization, OrgSync functions more as a bank-transferring and form-filing system than a club event application.

Despite efforts from WPI organizations and departments to overcome OrgSync's shortcomings, recent initiatives fall short of expectations as WPI stakeholders have little control over OrgSync's feature and user interface development. Furthermore, replacing OrgSync with a full-fledge alternative is not currently feasible as the licensing agreement is still active. Thus, utilizing student developed to create an application in cooperation with WPI students and departments is the most viable solution to the event marketing problem.

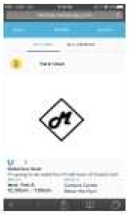
Methodology

To develop our application, we utilized a combination of a software development life cycle to develop our minimal application over A-Term, and an agile scrum methodology to incrementally improve the application over B/C-Terms. Our responsibilities are as follows:

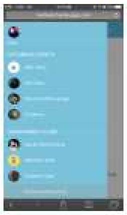
	Architectural Spike	Agile Scrum Development
Process Diagram	THE SOFTWARE DEVELOPMENT CYCLE	
Business Duties	• Initial Requirements Gathering via Discovery Survey and Literature Review • Initial Use Case and Data Flow Diagraming • Initial User Interface Designing	• Continuous Requirements Gathering: • Conduct Club President Focus Group • Interview WPI Undergraduates • Consult with WPI Marketing/IT Departments • Incremental Use Case and Data Flow Diagramming • Incremental User Interface Design
Computer Science Duties	• Establish Development Tools • Identify Development Environment and Frameworks for Application • Create Entity Relationship Diagram and Code Database • Develop Minimum Application	• Arrange Sprint Planning Meeting • Bi-weekly → Weekly Development Sprints to incrementally improve application • Partake in Team Daily Standups • Present to Advisors and stakeholders during Sprint Review Meetings. • Coordinate Team Sprint Retrospectives

Application Mockups

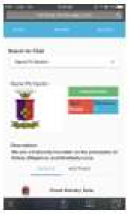
Event Feed



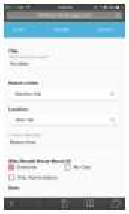
Personalized Drawer Menu



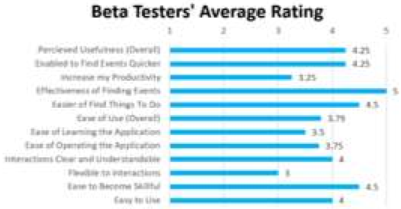
Club Page



Create Event Form



Beta Testers' Average Rating



Results

Based on the Technology Acceptance Model submissions by several randomly selected beta-testers, the application is more than satisfactory in its perceived usefulness but can still improve in said category and its ease of use. Based on these submissions and additional feedback from stakeholders, we recommend the following actions be taken:

1. Integrate the WPI IT's Central Authentication System into the Application
2. Continue development based on the product backlog and user feedback.
3. Develop in iOS and Android languages to optimize the mobile application experience.
4. Work with Student Activities Office and SGA to maximize club outreach and marketing.
5. Conduct feasibility study of other universities using the application, especially the Worcester Consortium

Acknowledgements

We would like to thank the following people and organizations for their generous support, time, and guidance throughout our project:

Project Advisors
Professor Diane Strong (FBS)
Professor Wilson Wong (CS)

WPI Departments:
WPI Marketing Department
WPI IT Department

WPI Students:
Those who completed surveys,
focus groups, and interviews