Project Number: MXC MQP 0571

SNAP Application

A Major Qualifying Project
submitted to the Faculty of
WORCESTER POLYTECHNIC INSTITUTE
in partial fulfilment of the requirements for the
degree of Bachelor of Science

by
Jack Palmstrom
Yosuke Nakamura

Date:
01 March 2019

Report Submitted to:

Professor Ciaraldi
Worcester Polytechnic Institute

ABSTRACT

The goal of this project was to develop a system for the SNAP service offered to the WPI community and maintained by the WPI Police Department. The project resulted in a web application hosted by WPI’s web servers and MySQL servers that can be used by the WPI community. The resulting application allows dispatchers to manage ride requests for SNAP as well as provides students the ability to submit request rides.

ACKNOWLEDGEMENTS

We would like to express our appreciation to Worcester Polytechnic Institute (WPI) and the IT Department as well as the Police Department for their support throughout this project. Given the utilities and support from the IT Department, we were able to complete this project. Without the cooperation from the Police Department, we would not be able to complete this project as a service to them. We would also like to thank Professor Ciaraldi for advising this project and guiding us through the project each week. Finally, we would like to thank the SNAP IQP team for their efforts in laying the groundwork for this project as without their efforts the inception for this project would have never been founded.

Table of Contents

CHAPTER 1: Introduction

1.1 Need for SNAP Application

SNAP is one of WPI's ride service programs, which allows WPI students to request rides at night within a one-mile radius of campus. The inception of the SNAP Application was based on the need set forth by the SNAP IQP team. [1] This team found that students frequently abandoned their SNAP requests after waiting for lengthy periods of time. After completing surveys, the IQP team found that students would be less likely to abandon a request if they were to submit them via a web form. Another need for the SNAP web application stems from students abandoning ride requests without informing a dispatcher. This situation requires the SNAP van to still drive out to a location even though the requester is no longer in need of a ride, wasting resources and time for SNAP. The integration of a SNAP web application would help relieve this. Lastly, by shifting the SNAP service to a web application, record keeping becomes easier. The current pen and paper system makes it difficult to find trends in the day-to-day data recorded about SNAP requests as well as maintain this data.

1.2 Existing SNAP System

The SNAP web application aims to replace the current workflow for SNAP dispatchers without losing any of the functionality from their current system. The SNAP dispatchers currently take ride request calls from WPI community members and log them using pen and paper. These requests are then assigned to a given van and the request is completed accordingly. To complete the request, a dispatcher calls a van driver to rely the new ride requests. There are a few drawbacks with pen and paper, including human error, time consumption, and difficulty running data analytics. With a web application, many of these drawbacks can be minimized. The web application would aim to collect more data about these ride requests that would otherwise be lost.

1.3 Target Audience

The target audience for this web application is the WPI community looking to use the SNAP service. This application was built in order to assist students requesting SNAP rides and the SNAP dispatchers handling those requests. This project aims to improve the user experience of SNAP requesters, as the SNAP IQP team outlined that students felt the current system was not fitting their needs. This system results in more students opting to not use the service or not rely on the service after making a request. The IQP team pointed out that this issue is compounded by the high crime rate in Worcester. Also, this application not only seeks to assist students, it aims to aid the job of the SNAP dispatchers. By providing SNAP dispatchers a tool for better workflow, this will help improve the efficiencies of the SNAP service.

1.4 Potential Features

The potential features for the SNAP application were highlighted in the proposal for this project. The features for this application were broken into three different portions of the system including the dispatcher user interface, the rider user interface, and the database for the application. These features focused on maintaining the same functionalities that the current system has while providing a better workflow for the dispatchers and a better user experience for ride requesters. This system also needs a way to persistently store the requests in the form of an archive viewable to the dispatchers.

CHAPTER 2: Background

WPI created the SNAP (Student Night Assistance Patrol) service for safer transportation at night for students, faculty, and staff of WPI. SNAP service operates between 6 P.M. to 4 A.M. during A and D term, 4 P.M. to 4 A.M. during B and C term, and 6 P.M. to 2 A.M. during recess/break. Transportation is limited under a one-mile radius from WPI campus, with Union Station being an exception. SNAP service was made for safer transportation, but the safety of the SNAP service has been questioned by users due to performance issues. Common issues include long wait times, being unable to fit in the van because of the luggage and/or other passengers, lack of notification if student have missed their ride, and the lack of information about the ETA. These problems are mainly caused by the current system of how the SNAP service is operated. Ride requests can be only made by phone calls, and dispatchers keep track of the requests with pen and paper. There are no communications or follow ups between the dispatcher and the rider once the ride is requested. Experiencing the performance issues as the users, three students, Marysol Zamaniego, Shannon McCormack, and Juan Caraballo conducted research about the need of student and dispatcher applications to improve SNAP's performance as part of their Interactive Qualifying Project (IQP) titled "SNAPP WPI SNAP services Assistive Application" in 2017-2018 academic year. By giving a survey to the students and dispatchers, they confirmed the desire for an application to improve the SNAP. [1] This project was put into practice based on their IQP.

2.1 Taking Over from the IQP

In addition to the research and the survey mentioned above, the IQP team has done a lot of preparatory work before the development phase of the application. First, they worked with the WPI Police Department, the co-sponsor of the SNAP service, and Lieutenant Karen Bueno, the SNAP Service Liaison. Cooperating with the WPI Police Department was essential to get the permission to create the application for SNAP, to make decisions on the product requirements of the applications, and to understand the SNAP service from the dispatcher side. In addition, they worked with the WebTech Group, the group of WPI Computer Science students that take on application/web development projects to improve student life, to ensure the maintainability of the application. WebTech Group helped the IQP team by giving advices to the user interface and developing the beta version of the student side application. Lastly, they worked on the development of a Terms of Use and Privacy Policy contract to provide with the SNAP application.

This year's WebTech Group did not show any interest to the SNAP application, thus we had to find another way to maintain the application after our graduation. Fortunately, we were able to contact the WPI IT department, and came to an agreement that they would maintain the application after our graduation.

2.2 Functional Requirements

The following sets of functional requirements were created by the SNAP IQP team in cooperation with the WPI Police Department.

2.2.1 Dispatcher Application Requirements [1]

- Ability for dispatcher to accept or reject a request.
    - Requests may be rejected if they did not meet the requirements.
- Separately organize new requests and requests in process.
- Application must be able to keep track of these statistics per day/week/month/year:
    - Number of cancellations
    - Number of no-shows
    - Total number of completed rides
- Make an archive to store historical service data.
- Edit the van assignments until the request has been completed.
- Be able to add requests manually to handle requests received through a phone call.

2.2.2 Rider Application Requirements [1]

- Student login/registration.
- Submit the ride request from the app.
    - Required Information:
        - Destination
        - Departure
        - Number of Passengers
    - Optional:
        - Accommodations

For both requirements, please read the SNAP IQP report for more details.

2.3 Web Technologies Used

There are many options to make the application nowadays. In the subsections below we will describe the web technologies we mainly used in this project.

2.3.1 HTML, CSS, and JavaScript

HTML, CSS, and JavaScript are the most popular languages used to construct web pages. HTML, which stands for Hypertext Markup Language, is the standard markup language used for creating web pages and applications. This language is used to describe the structure of elements on the web page/application, such as where to place the header of the page [2]. CSS, which stands for Cascading Style Sheets, is the stylesheet language used to describe how the HTML elements are to be displayed on the screen, such as font, size, and color of the text. [3] JavaScript is the programming language that is commonly used to create interactive effects within web browsers, such as actions caused by button clicks. [4] We used these three languages to develop our application since the combination of these three languages is the most common method for web development.

2.3.2 MySQL

MySQL is a relational database management system. [5] Relational database means that the data is stored in separate tables containing a column or columns that other tables can relate to. [6] Management system lets the user add, access, and process data stored in the computer database to maximize the utility of relational database.

2.3.3 Node.js

Node.js is the platform built on Chrome's JavaScript runtime that is designed to build scalable network applications. By using Node.js, the application will be lighter and more efficient. There are also libraries to solve most of the generic problems by using npm (Node package manager). [7]

2.3.4 Passport-SAML

Passport is the authentication library provided by Node.js. Passport-SAML uses Security Assertion Markup Language (SAML) for authentication. SAML works by exchanging the digitally signed XML documents to transfer the user's identity from one place to another. [8]

CHAPTER 3: Methodology

3.1 Overview of Application

The SNAP Application was built based on the need the WPI SNAP IQP team set forth in their report. The web application would need to be able to handle ride requests from users, while still allowing the SNAP dispatchers to have all the same functionality for handling these ride requests. The web app would need to be able to handle ride request call ins as well as requests made from the rider UI of the application.

This application would need to support three different kinds of users for the system: riders, dispatchers, and administrators. Riders would need to have both a call-in functionality and to request a ride via a request form. The dispatchers must have functionalities to handle these requests, assign these requests, and complete them. System administrators would need to be able to assign other administrators, assign other dispatchers, and signal that SNAP is unavailable at this time.

3.2 Use Cases

Our team created a series of use cases for the SNAP application based upon the preliminary use cases outlined by the SNAP IQP report. These use cases were developed for each of the three kinds of users for the application. The use cases are extensive and allowed our team to develop an application to suit all of our users' needs. The user cases are listed below and shown in Appendix C.

General

- Login to application
- Logout of application

Dispatcher

- View all new ride requests
- View all in process ride requests

- View the archive of ride requests
- Add a new ride request
- Assign a ride request to a van
- Cancel a ride request
- Mark a ride request as completed
- Change a van assignment
- View van maintenance page

Rider

- Request a ride
- View the SNAP policy page
- View the Gateway Shuttle hours
- Click to call the SNAP phone number

Administrator

- Assign other administrators
- Assign other dispatchers
- Delete dispatchers
- Discontinue SNAP service

3.3 Dispatcher Side

The dispatcher side of the application was the first user interface to be created within our application. The user interface for the dispatcher component of this application was a topic previously researched by the IQP team that preceded this project. Due to the IQP team already creating mock-ups for the dispatcher UI, we modeled the UI to replicate their mock-ups as these mock-ups had already been approved by the WPI Police Department. The IQP report also goes through all the necessary features that the WPI dispatchers would need to replace their current system. These features were derived from the use cases described in Section 3.2.

3.4 Rider Side

The rider side of the application was designed for use by WPI students. Based on the SNAP IQP team's finding that most students would request SNAP rides via smartphone, we designed this portion of the application to be mobile friendly. This portion of the application utilizes Bootstrap 4, which is a front-end component library for mobile-first web design. [12] This library allows many of the components for the SNAP rider interface to be scalable from mobile to desktop versions. Similarly to the dispatcher interface for the application, the rider interface was designed based on mock-ups completed by the IQP team that were approved by the WPI Police Department. The UI contains features such as making a ride request, seeing WPI Gateway shuttle hours, and click-to-call SNAP. These features were derived from the use cases described in Section 3.2.

3.5 Database Implementation

The SNAP Application utilizes a MySQL database to persistently store information used throughout the application. Due to MySQL being a relational database, proper table structure is important to ensure that queries are done as efficiently as possible. The table structure for the application can be found in the entity relationship diagram found in Appendix B. We planned the database to have appropriate Foreign Key to Primary Key relationships, as well as appropriate column modifiers. When a ride request is inputted into the system, it is assigned a unique ride ID that can be used to track the request depending on whether the request is still in the new request queue, the request is in process, or the request has been archived. Besides storing ride request information, we also allow dispatchers to store information on van maintenance.

Security is a concern when deploying a database, and when accessing that database through the application. To access the database within our program we are using the MySQL npm package. This package assists with connections to the database as well as makes preventing SQL injections easier with prepared statements that escape query values. When creating queries, the only values that are variable are ones marked with a '?' within the prepared statement. This prevents a malicious user from querying the database for information he/she shouldn’t have access to.

3.6 WPI Server

The goal of this project was to have our web application hosted on a WPI server and accessible through a WPI endpoint. Once we received approval from the WPI IT department, we began migrating our project away from AWS. We received a new Ubuntu 18.04 instance with a snap.wpi.edu endpoint from the WPI IT department. Next, we set up the environment to host this project including setting up Node.js, NPM, MySQL, Nginx, and PM2 on this instance. We then migrated the AWS database over to the MySQL database hosted locally on the server. A reverse proxy would need to be set up to properly show our application on the endpoint that was given to us. The reverse proxy was set up based on an example found on Medium [9]. Finally, we want our application to run on startup of the server. PM2, a production process manager package, allows us to have built-in load balancing, keep programs ‘alive’ forever, and reload the application without downtime. The server was set up with PM2 based on the quick start guide of the PM2 site [10].

3.7 Authentication

The SNAP Application must use WPI’s authentication system for users. This system allows WPI to verify the credentials for all students and staff members. Since our application should only be accessible to users in the WPI community, it is important to implement WPI’s authentication system into the application. To meet this requirement, the application needed to support SAML 2.0 for authenticating users. The application would then need the passport and passport-saml packages to utilize SAML. Using passport, we set up a ‘SAML strategy’ that would redirect users to the WPI authentication screen, and upon a successful login redirect back to our application. We set up our SAML strategy based upon the README found on the NPM page for passport-saml [11].

CHAPTER 4: Results and Analysis

While we take over from the IQP, the most important part of our project was to create an application that fulfills all the product requirements. In short, we successfully made the dispatcher application and student application with all required functionalities except for fully-working authentication. In the next section, results of the dispatcher application, the student application, and the authentication system are described. Please check Appendix A for details about how to use the application.

4.1 Dispatcher Application

4.1.1 Main Page

The main page of the dispatcher application consists of two tables placed side by side. The table on the left side is for new requests and the table on the right side is for in-process requests. The new requests table has columns for ride ID, pick-up location, drop-off location, number of passengers, accommodations, and time received. The in-process table is almost the same as the new requests table, but it has a column for assigned van number to show which van is assigned to that request. In addition to the tables, there is a button in each top corner. The button on the left-top corner is linked to the page with a form to manually add the request. Top-right corner has a hamburger menu containing the page for archive of requests, page to check the van in maintenance, and an admin page. For the main page, we followed the well-thought-out idea of the IQP team for the basic user interface.

4.1.2 Other pages

We successfully implemented other required functionalities as well. Dispatchers can access the “Ride Assignment” page by double clicking the request in the “New Request” table to select a van to assign the request or to reject the request. The “Edit Request” page can be accessed by double clicking the request in the “In Process Requests” table. On this page, the dispatcher can edit the request, complete the request, or report the request as a no-show. After the request is rejected, completed, or reported as a no-show, that request will be saved in the archive with the chosen status.

4.1.3 Hamburger Menu

As we mentioned above, the hamburger menu contains the page for archive of requests, page to check the van in maintenance, and an admin page. The archive page has a table with sort, search, and export functionalities. Statistics per day/week/month/year of number of cancellations, no-shows, and total number of completed rides can be taken by utilizing these functionalities. Data in the archive can be exported into CSV or Excel to take the back-up. The van in maintenance page simply shows a table of vans under maintenance. The admin page has "Check/ Edit Van Status" page, and only admins can edit the van status for now.

With the completion of the hamburger menu, we fulfilled all the requirements for the dispatcher application. Every required product functionality is implemented. However, we were not able to implement any additional functionalities mentioned in the IQP report. The van maintenance page still needs improvement to be utilized well. We also think that some improvements on the existing functionalities will make the application more user friendly. For example, double clicking on the table entry can be replaced with drag and drop between tables. Other improvements can be also made based on users' feedback.

4.2 Student Application

The student application is a simple application to send the request. The main page of the student application is a request form. To submit the request, the user must provide all the necessary information: departure, destination, and number of passengers. There is an optional text box for accommodations. Other than submitting a request, the user can check the Gateway Shuttle hours, check the SNAP policy, and make a call to SNAP from the menu bar located in the upper left corner.

This straightforward application is easy to use and fulfils the final requirements given by the IQP team. However, there are still many things that can be improved, such as adding the how-to tutorial and a cancel request button allowing students to cancel their requests without giving a call to SNAP.

4.3 Authentication

As mentioned in the beginning of this chapter, we were not able to fully implement the authentication system in our application. We have foundational code written in our application to use passport and passport-saml packages, however, we are waiting for the XML file to be issued from the IT department to utilize the packages. Since we have already made the admin page with local login system, we are certainly sure that we can complete our application fully as soon as we get the XML file.

CHAPTER 5: Conclusion and Recommendations

5.1 Measure of Success

Our team considers the development of the SNAP web application to be successful as we developed an application to meet the need the WPI SNAP IQP team. The application is currently is a state for improvement upon further testing. This application has been tested by shadowing a SNAP dispatcher, the notes from this testing can be found in Appendix D. This application can serve as a model for other schools to implement a web application for their transportation service.

5.1.1 Utilization of WPI Resources

This web application was created in conjunction with the WPI IT Department and Police Department representatives. Since this will be a project that is inherited by both departments, we needed to create an application tailored to their needs. Our team feels this has been a success after the testing we have performed with the WPI dispatchers along with the utilization of WPI IT resources.

5.1.2 Application Capabilities

The current state of the application successfully met the initial project goal. The application can be used for ride requesting and ride request management. The current state of the application supports riders making requests along with dispatchers manually entering requests. Once a request has been entered a dispatcher can assign a van to the request, mark the request as completed, or delete the request for a cancellation. The application stores all ride request entries in an archive. This archive allows the dispatchers and administrators to potentially run data analytics on the past requests in hopes to find trends.

5.2 Future Advancements

5.2.1 Possible additional Features

There are many possible areas for improvement within the current state of the SNAP application. Some of these areas for improvement vary from increasing workflow efficiency to administrative features for the application. While working on the project, we kept a running list of features whether these features be critical to the system or simply aesthetically pleasing. These features include but are not limited to: SNAP is down notifications do not work, a drag and drop ride request features for the dispatcher UI, the dispatcher home page should update automatically with new ride requests, caching user data, user is informed when their request is in process, a personalized UI based on the signed in user, and resetting the archive of requests on a given interval.

5.2.2 Expanding User Testing

Due to time constraints with this project, we were only able to conduct limited user testing on the application. User testing for this project was an issue due to the development process taking the entirety of the project's lifetime. With having the WPI IT Department sign on to the project, this came with a whole extra set of features that needed to be implemented before testing could be done. Future work on this project should also include more thorough user testing. This would allow for more necessary features for the application to emerge and be developed for the application.

5.2.3 Code Revision

Lastly, any further development of this project should attempt to make revisions to the current code base. This code base is commented well but could be made more modular. The overall file structure of the application could use some slight revision to make the code base more intuitive. This code base also has room for increasing efficiency as we did not make this our main focus for the project.

REFERENCES

[1] J.S. Caraballo, M. Zamaniego, S.K. McCormak, “SNAPP WPI SNAP Services Assistive Application “ May 2018. [Online]. Available: https://web.wpi.edu/Pubs/E-project/Available/E-project-050818-163611/unrestricted/Final_Report.pdf [Accessed 25 Feb. 2019]

[2] w3school, “HTML Introduction” [Online].Available: https://www.w3schools.com/html/html_intro.asp [Accessed 24 Feb. 2019]

[3] w3school, “CSS Introduction” [Online].Available: https://www.w3schools.com/css/css_intro.asp [Accessed 24 Feb. 2019]

[4]MDN web docs, “What is JavaScript?” [Online].Available: https://developer.mozilla.org/en-US/docs/Learn/JavaScript/First_steps/What_is_JavaScript

[Accessed 24 Feb. 2019]

[5] MySQL, “What is MySQL?” 2019. [Online]. Available:

https://dev.mysql.com/doc/refman/5.7/en/what-is-mysql.html [Accessed 25 Feb. 2019]

[6] HowStuffWorks.com, “What are relational databases?” 23 March 2001. [Online]. Available:

https://computer.howstuffworks.com/question599.htm [Accessed 26 Feb. 2019]

[7] Priyesh Patel. freeCodeCamp, “What exactly is Node.js?” 18 Apr. 2018. [Online]. Available:

https://medium.freecodecamp.org/what-exactly-is-node-js-ae36e97449f5 [Accessed 25 Feb.

2019]

[8] OneLogin DEVELOPERS, “Overview of SAML”. [Online]. Available:

https://developers.onelogin.com/saml [Accessed 25 Feb. 2019]

[9] Utkarsh Verma, “Configure Nginx as a web server and reverse proxy for Nodejs application on AWS Ubuntu 16.04 server” 01 Feb. 2017. [Online].Available: https://medium.com/@utkarsh_verma/configure-nginx-as-a-web-server-and-reverse-proxy-for-nodejs-application-on-aws-ubuntu-16-04-server-872922e21d38 [Accessed 25 Feb. 2019].

[10] Keymetrics, “PM2: Advanced, production process manager for Node.js” 19 Feb. 2019. [Online].Available: http://pm2.keymetrics.io/ [Accessed 20 Feb. 2019].

[11] npm, “Passport-SAML” 20 Dec. 2018. [Online].Available:
https://www.npmjs.com/package/passport-saml [Accessed 14 Feb. 2019].

[12] Bootstrap, “Bootstrap” [Online].Available: https://getbootstrap.com/ [Accessed 14 Feb. 2019]

APPENDIX A: User's Manual

SNAP Application

User's manual

Written by
Jack Palmstrom
and
Yosuke Nakamura
March 10, 2019

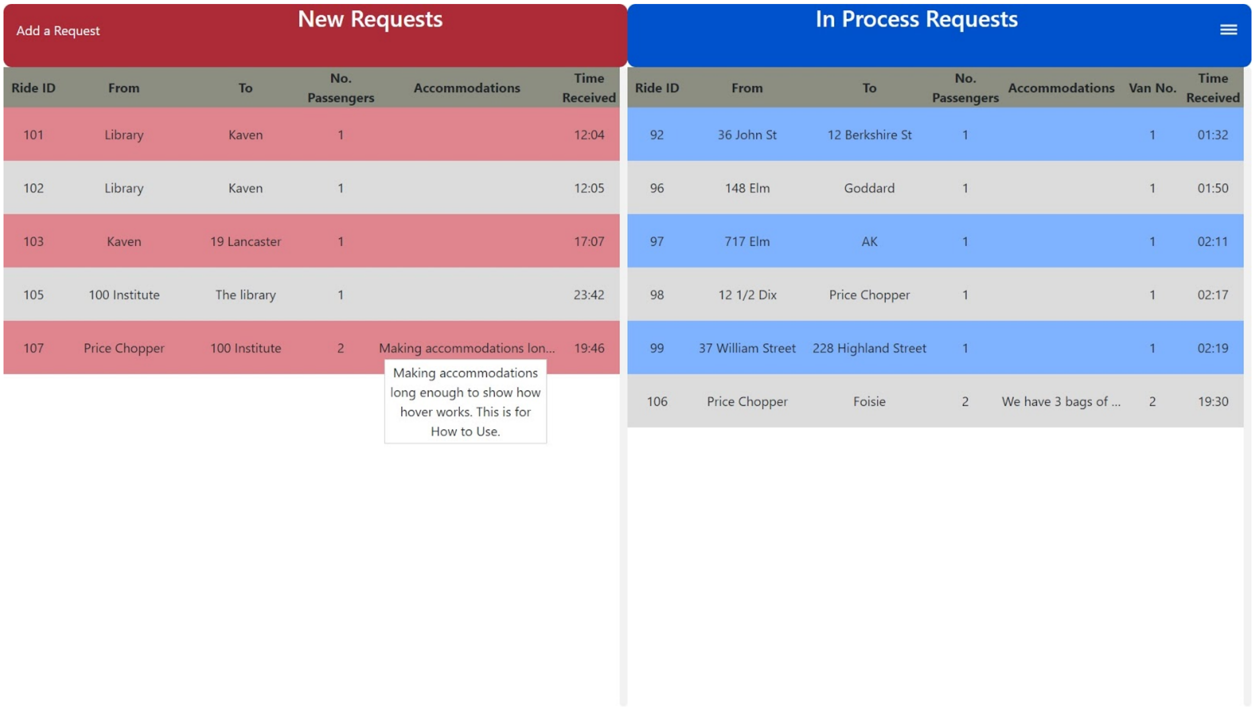
Table of Contents

List of Figures

Dispatcher Application

***You must be hired as the SNAP dispatcher to gain the access to this web application.**

Home



New Requests

Ride ID	From	To	No. Passengers	Accommodations	Time Received
101	Library	Kaven	1		12:04
102	Library	Kaven	1		12:05
103	Kaven	19 Lancaster	1		17:07
105	100 Institute	The library	1		23:42
107	Price Chopper	100 Institute	2	Making accommodations lon...	19:46

Making accommodations long enough to show how hover works. This is for How to Use.

In Process Requests

Ride ID	From	To	No. Passengers	Accommodations	Van No.	Time Received
92	36 John St	12 Berkshire St	1		1	01:32
96	148 Elm	Goddard	1		1	01:50
97	717 Elm	AK	1		1	02:11
98	12 1/2 Dix	Price Chopper	1		1	02:17
99	37 William Street	228 Highland Street	1		1	02:19
106	Price Chopper	Foisie	2	We have 3 bags of ...	2	19:30

Figure 1: Dispatcher Application Home Page

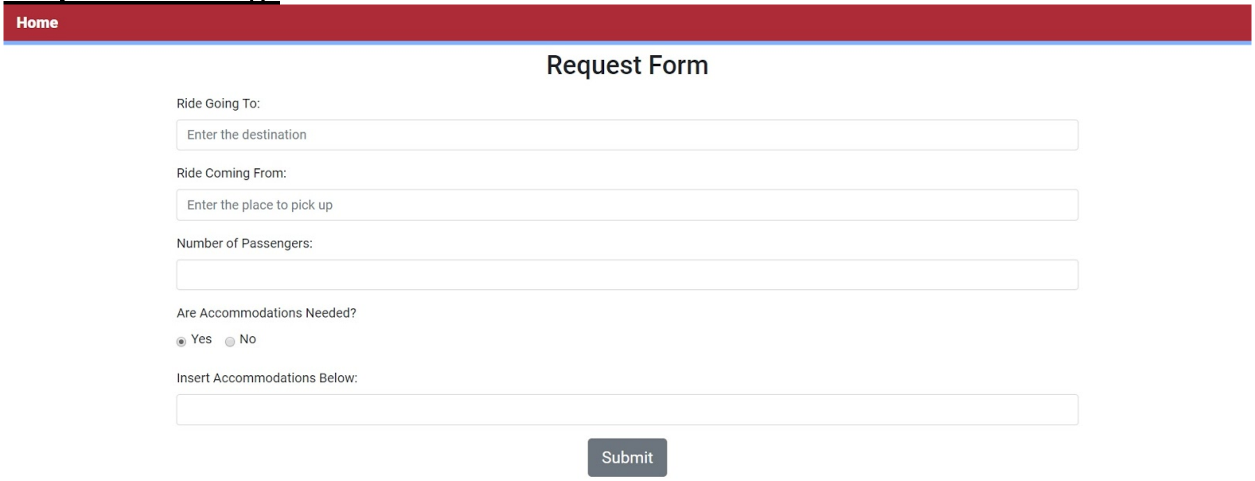
Type in "snap.wpi.edu/dispatcher" to the address bar of your browser and hit the Enter key to access the Home page. Home page consists of two tables, "Add a Request" button, and a hamburger menu. The table on the left shows the incoming requests, and the table on the right shows the in-process requests. "New Requests" table has columns for ride ID, departure, destination, number of passengers, accommodations, and time received. "In Process Requests" table has "van No." column in addition. This column shows the van number to which the request was assigned to. When the text did not fit in the accommodations column, you can hover on that cell to see the full text as it is shown in the figure above.

How to Access Other Pages:

1. To open the "Request Form", click the "Add a Request" button on the upper left.
2. To open the "Ride Assignment" page, double click on the request in the "New Requests" table.
3. To open the "Edit Request" page, double click on the request in the "In Process Requests" table.

4. To view the archive, select “Archive” located under the hamburger menu.

Request Form Page



Request Form

Ride Going To:

Enter the destination

Ride Coming From:

Enter the place to pick up

Number of Passengers:

Are Accommodations Needed?

● Yes ○ No

Insert Accommodations Below:

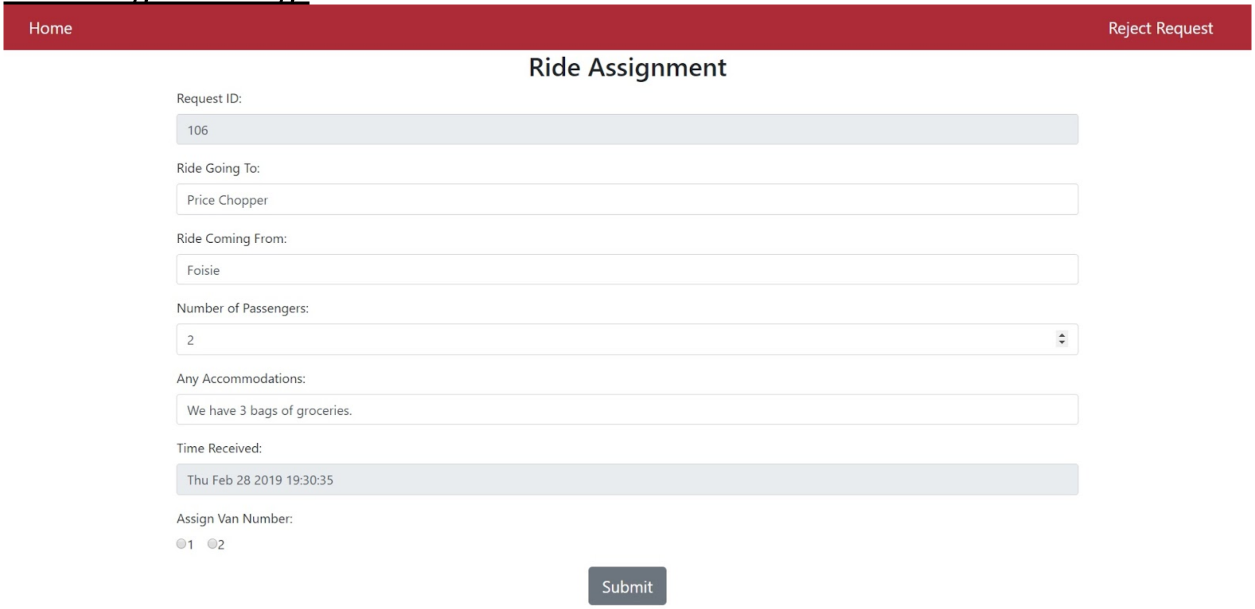
Figure 2: Dispatcher Request Form

To open this page, click on the “Add a Request” button located in the upper left corner of home page. You can manually add a request on this page.

Manually Add the Request

1. Fill out the “Request Form” shown below. All of the fields are necessary except for the accommodations. Textbox for accommodations is hidden when “No” is selected.
2. Hit the “Submit” button to add the request to “New Request” table.

Ride Assignment Page



Ride Assignment

Request ID: 106

Ride Going To: Price Chopper

Ride Coming From: Foisie

Number of Passengers: 2

Any Accommodations: We have 3 bags of groceries.

Time Received: Thu Feb 28 2019 19:30:35

Assign Van Number:

○1 ○2

Figure 3: Ride Assignment Page

To open this page, double click on the request in the “New Requests” table. There are two things you can do on this page. You can assign the request to the van, or you can choose to reject the request if the request was not following the SNAP policy.

Assign the Request

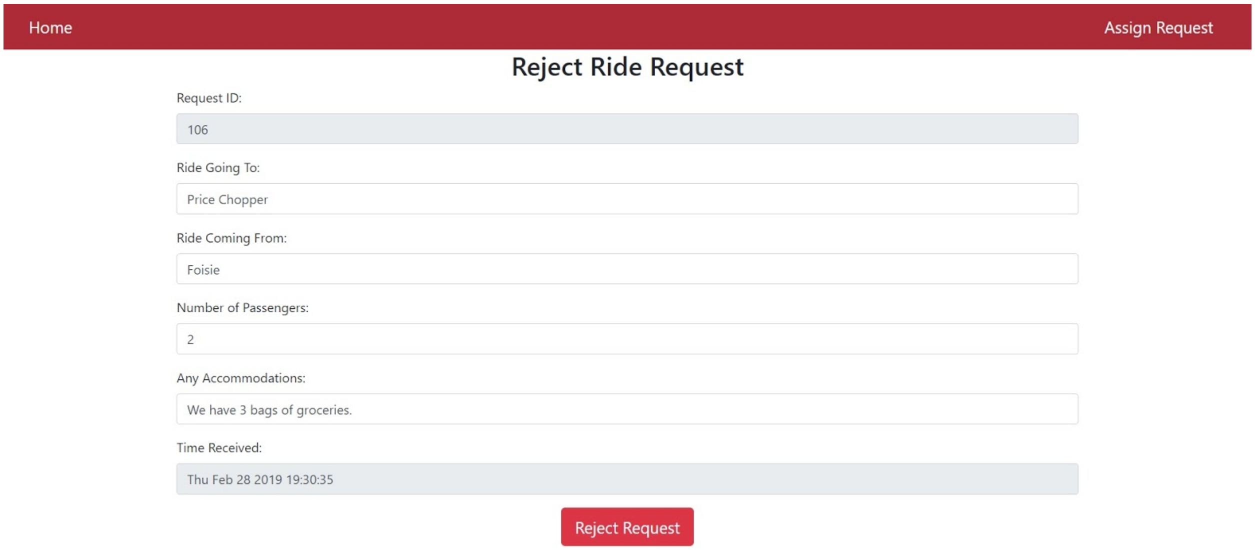
1. Select a van to assign the request. Change other information if needed.

2. Hit the “Submit” button to move the request to the “In Process Requests” table.

Reject the Request

1. Click on the “Reject Request” button located at the upper right corner. This will bring you to the “Reject Ride Request” page.

2. Carefully check the details of the request to make sure you are rejecting the request you want to. After checking the request, click on the “Reject Request” button. This will send the request into archive with “Rejected” status.



Reject Ride Request

Request ID: 106

Ride Going To: Price Chopper

Ride Coming From: Foisie

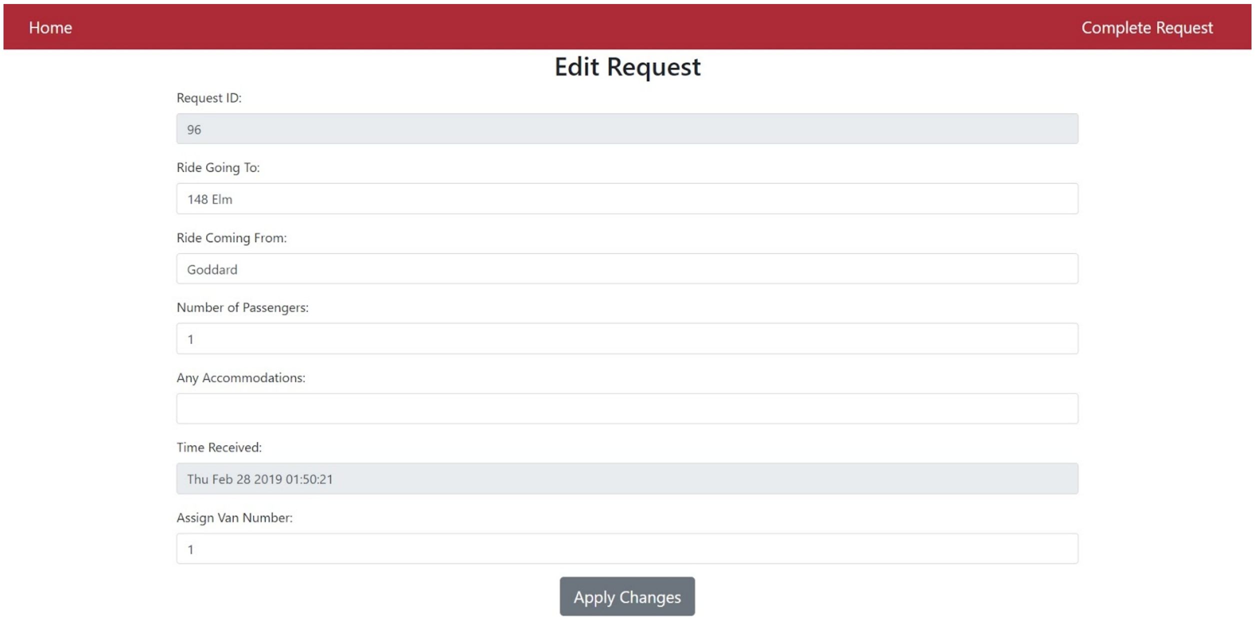
Number of Passengers: 2

Any Accommodations: We have 3 bags of groceries.

Time Received: Thu Feb 28 2019 19:30:35

Figure 4: Reject Ride Request Page

Edit Request Page



Edit Request

Request ID: 96

Ride Going To: 148 Elm

Ride Coming From: Goddard

Number of Passengers: 1

Any Accommodations:

Time Received: Thu Feb 28 2019 01:50:21

Assign Van Number: 1

Figure 5: Edit Request Page

To open this page, double click on the request in the “In Process Request” table. There are three things you can do on this page. You can edit the request, complete the request, or report the request as no-show.

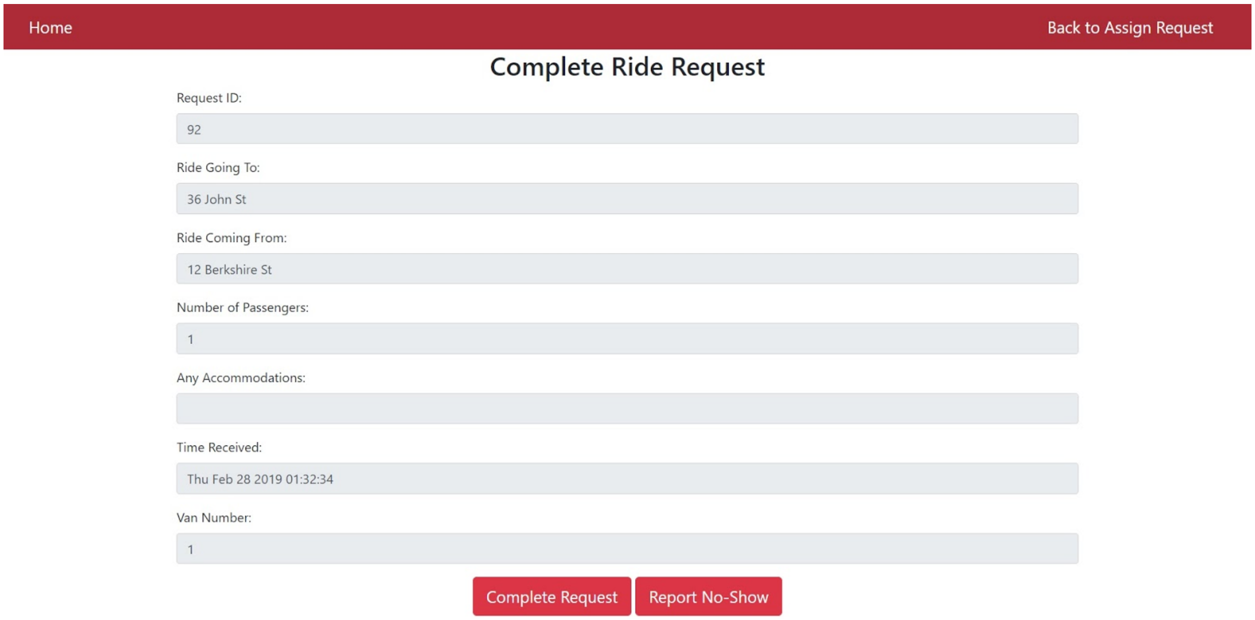
Edit the Request

1. Edit the information. “Ride ID” and “Time Received” cannot be changed.
2. Click on “Apply Changes” button to complete the edit.

Complete the Request or Report No-Show

1. Click on the “Complete Request” located in the upper right corner.
2. Check the information to make sure.
3. Click on the “Complete Request” to complete the request.

Click on the “Report No-Show” to report the request as no-show



Complete Ride Request

Request ID:
92

Ride Going To:
36 John St

Ride Coming From:
12 Berkshire St

Number of Passengers:
1

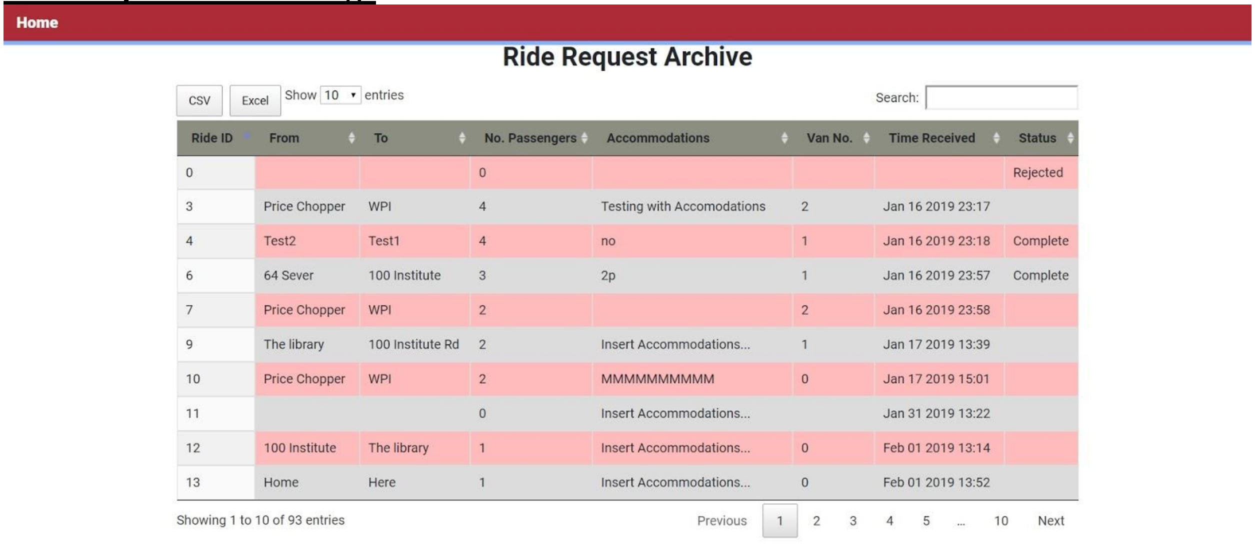
Any Accommodations:

Time Received:
Thu Feb 28 2019 01:32:34

Van Number:
1

Figure 6: Complete Ride Request Page

Ride Request Archive Page



Ride Request Archive

Ride ID	From	To	No. Passengers	Accommodations	Van No.	Time Received	Status
0			0				Rejected
3	Price Chopper	WPI	4	Testing with Accomodations	2	Jan 16 2019 23:17	
4	Test2	Test1	4	no	1	Jan 16 2019 23:18	Complete
6	64 Sever	100 Institute	3	2p	1	Jan 16 2019 23:57	Complete
7	Price Chopper	WPI	2		2	Jan 16 2019 23:58	
9	The library	100 Institute Rd	2	Insert Accommodations...	1	Jan 17 2019 13:39	
10	Price Chopper	WPI	2	MMMMMMMMMM	0	Jan 17 2019 15:01	
11			0	Insert Accommodations...		Jan 31 2019 13:22	
12	100 Institute	The library	1	Insert Accommodations...	0	Feb 01 2019 13:14	
13	Home	Here	1	Insert Accommodations...	0	Feb 01 2019 13:52	

Showing 1 to 10 of 93 entries

Figure 7: Ride Request Archive Page

To open this page, click on the hamburger menu located in the upper right corner of the home page and select “View Archive.” You can sort and search archived requests on this page. You can export the archive into CSV or Excel file as well. Please check DataTables package for more details.

Sort

Click on the table header to sort the table. Data in numbers will be sorted in ascending order, and data in alphabets will be sorted in alphabetical order. You can switch the order to descending or reverse alphabetical by clicking on the same header again.

Search

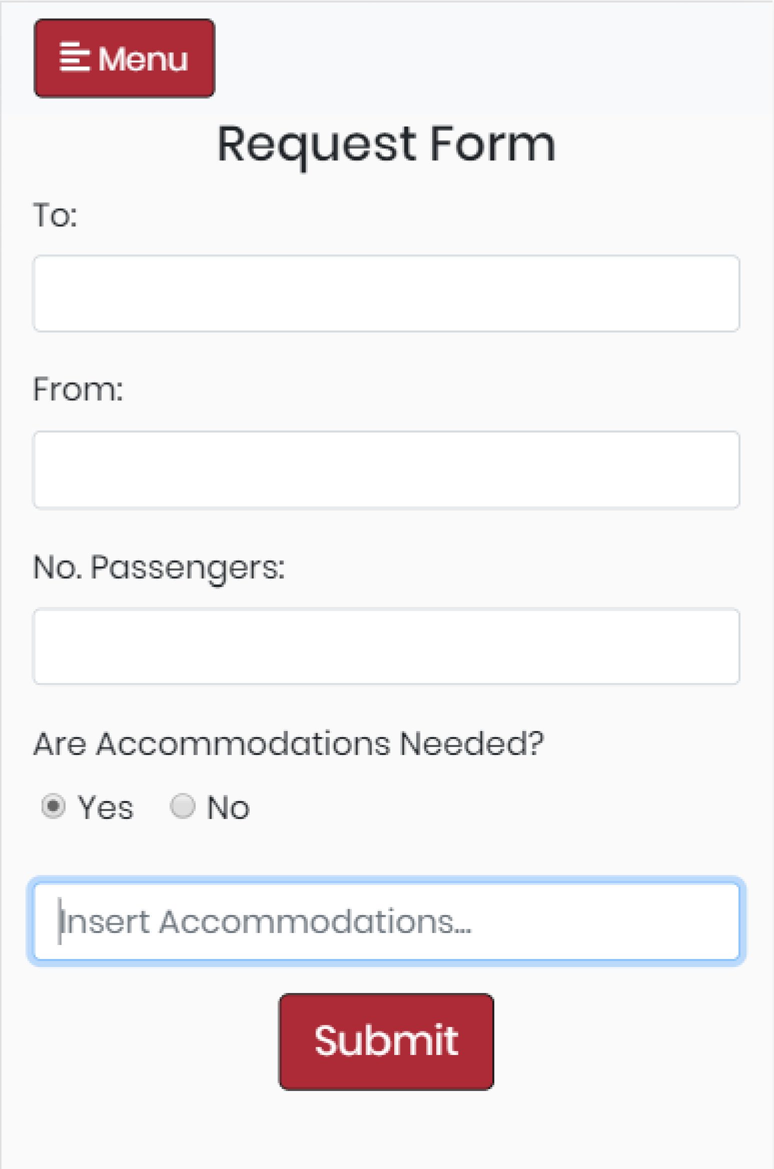
Simply, type in a word to search for. For example, search for “Complete” to find all the completed requests.

Export to CSV or Excel

Click on the button located at the upper left to export the archive into CSV or Excel file. If you click on the button after filtering, exported file will only contain the filtered archive. To easily collect the statistics for specific status, search with “Rejected”, “Completed”, or “No-Show” before exporting the archive.

Student Application

Home



Are Accommodations Needed?

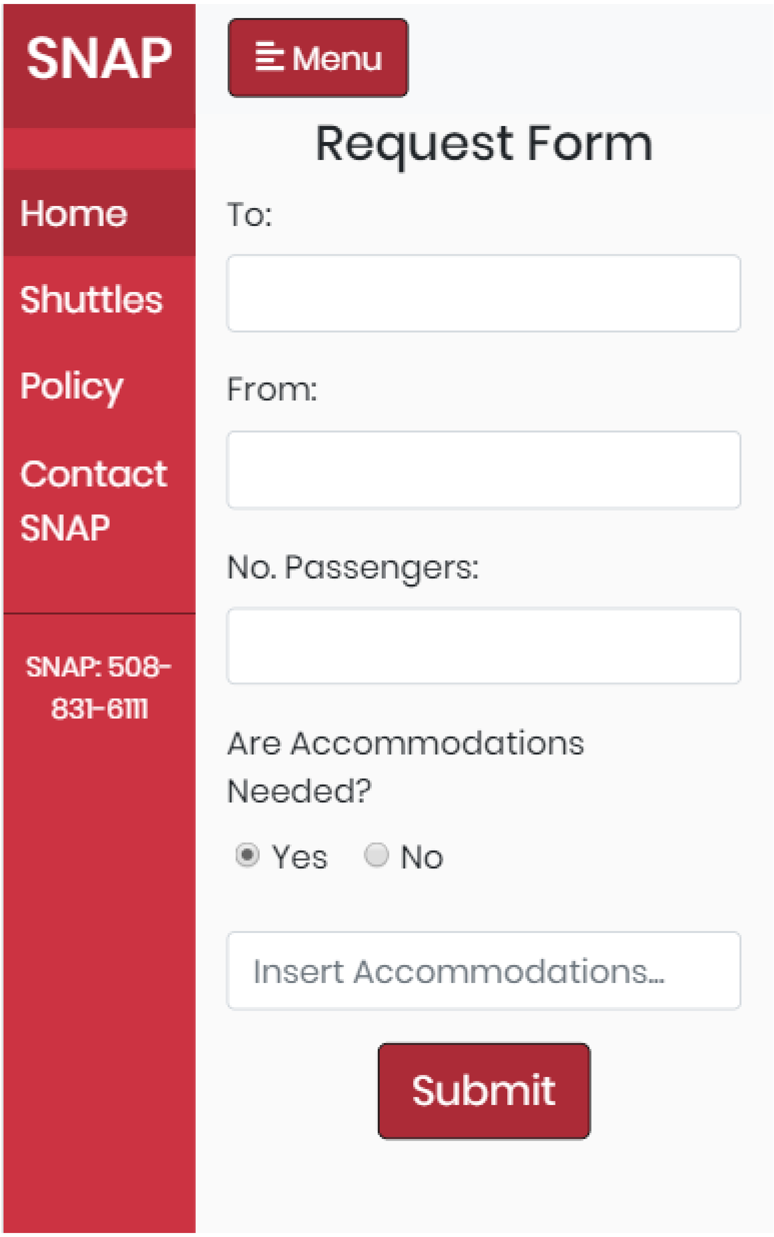
● Yes ○ No

Figure 8: Student Application Home Page

Send the Request

Type in “snap.wpi.edu” to the address bar of your browser and hit the Enter key to access this application. CAS login screen will pop-up if you are not logged in. Home page for student application serves as a request form. User can submit the request after filling out all the necessary fields: departure, destination, and number of passengers. User can add accommodations if needed. By selecting “Yes” under “Are Accommodations Needed?”, a field for accommodations will appear.

Menu Bar



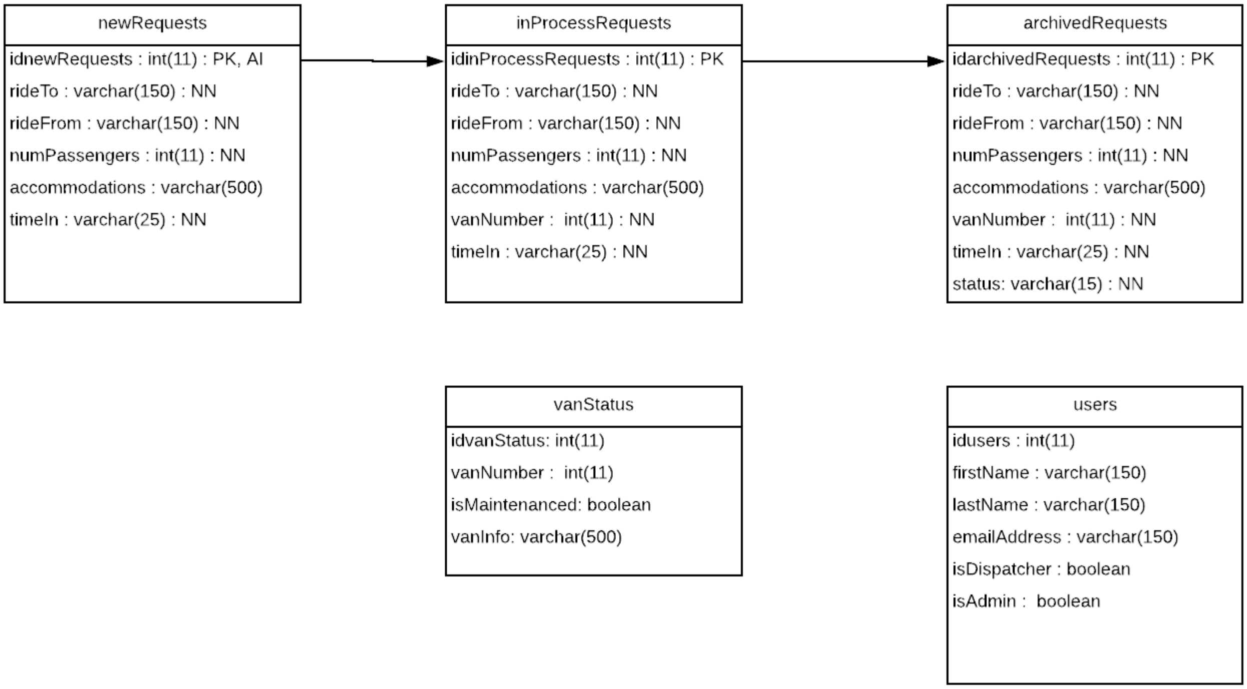
Are Accommodations Needed?

● Yes ○ No

Figure 9: Figure 9: Menu Bar

From the menu, user can access to pages to check The Shuttles hours and SNAP policy. User can call SNAP by clicking on “Contact SNAP, ” or check the SNAP phone number from the menu as well.

APPENDIX B: Entity Relationship Diagram



newRequests

idnewRequests : int(11) : PK, AI
rideTo : varchar(150) : NN
rideFrom : varchar(150) : NN
numPassengers : int(11) : NN
accommodations : varchar(500)
timeIn : varchar(25) : NN

inProcessRequests

idinProcessRequests : int(11) : PK
rideTo : varchar(150) : NN
rideFrom : varchar(150) : NN
numPassengers : int(11) : NN
accommodations : varchar(500)
vanNumber : int(11) : NN
timeIn : varchar(25) : NN

archivedRequests

idarchivedRequests : int(11) : PK
rideTo : varchar(150) : NN
rideFrom : varchar(150) : NN
numPassengers : int(11) : NN
accommodations : varchar(500)
vanNumber : int(11) : NN
timeIn : varchar(25) : NN
status: varchar(15) : NN

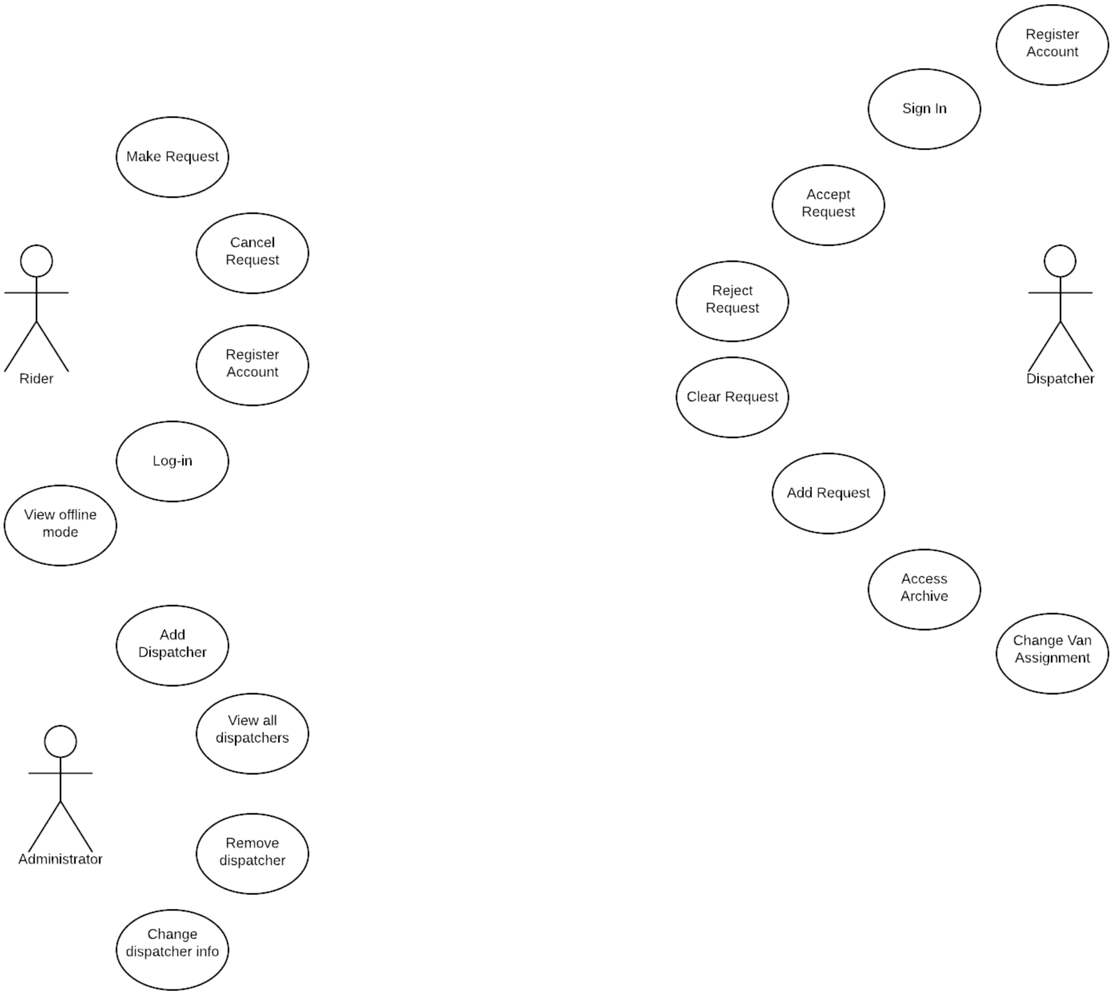
vanStatus

idvanStatus: int(11)
vanNumber : int(11)
isMaintenanced: boolean
vanInfo: varchar(500)

users

idusers : int(11)
firstName : varchar(150)
lastName : varchar(150)
emailAddress : varchar(150)
isDispatcher : boolean
isAdmin : boolean

APPENDIX C: Use Case Diagram



APPENDIX D: Dispatcher Shadowing Notes

Testing Notes

- The dispatcher realized that request was from a destination and to a destination already covered by the Gateway Shuttle. Instead of accepting the request, she told the student to take the shuttle.
- Another student called asking for information on the Gateway Shuttle service.
- Time received column for the request is wrong on dispatcher view running on WPI server. The time is off by five hours, so we believe this has to do with the time difference between GMT and EST.
- The dispatcher asks where the rider is coming from and then the destination they are going to.
- Rides requests that are called in immediately get assigned out. It is up to the van drivers' discretion on the order to pick up the riders.
- The dispatcher rarely relays the number of passengers to the van driver for a given request.
- The dispatcher did not ask the rider if they have any accommodations that need to be met.