Project Overworld

Interactive Qualifying Project Report
submitted in partial fulfillment
of the degree of Bachelor of Science

Dillon DeSimone
Tom Farro
Francesca Carletto-Leon

and advised by
Prof. Keith Zizza

Submitted March 6th, 2015

Table of Contents

Overview

Introduction:

Project Overworld is an interactive GPS enabled map of the Worcester Polytechnic Institute campus, which is accessible through mobile devices. The project aims to act as a guidance system for freshmen and to provide connectivity between WPI students. The application features a avatar creation and customization system, similar to fantasy role-playing type games. By exploring various locations of campus and answering trivia questions, players can level up and further customize their avatar.

Our team decided it was important to us to not only research and write about a project concept, but to actually create a working prototype. By doing so, we were able to explore mobile development much more effectively and receive feedback from peers.

The initial design of the mobile application was focused around creating a satisfactory single player experience through the inclusion of music, avatar personalization, and a mini-game system (trivia). As we began fleshing out our feature ideas, it became apparent that this project would benefit greatly from multi-player aspects, allowing for interconnectivity between users and allowing competitive and cooperative play. Though this idea was eventually scrapped for the final prototype due to time constraints and scope, we made the decision to focus on and polish our single player elements.

Design Concept Development

Target Audience:

The application aims to appeal to users unfamiliar with campus. When discussing our target audience, we theorized about whom the application would be most helpful to and made decisions in the application's design which would benefit this audience. Project Overworld's target audience is freshmen and other newcomers to campus.

The application is also aimed towards a younger audience who are receptive to the mechanics of casual games. We were looking to target players who would use the application in short bursts during their walks around campus and who would be motivated to customize their avatars with collectable items.

While deciding on our target audience the topic of the user experience goal came up. The experience goal, as we define it, is a list of feelings or emotions which the game elicits from users. Project Overworld was created as a means of education and assistance. However, we also intended for it to motivate users to explore parts of campus they don't usually travel to, or would otherwise miss out on. The user should also feel a sense of achievement through successful trivia gameplay. As such, we decided our experience goals were as follows:

- Exploration / Curiosity
- Accomplishment and pride
- A sense of connection and interconnectivity.

Research & Planning:

Going in to the project, our team consisted of three people: Francesca Carletto-Leon, our designer, Dillon DeSimone, our lead artist, and Tom Farro, our programmer. The team's research for the project extended to include the observation of popular mobile games and analyzing them to discern what common elements attracted players. Some of the more noticeable features included:

- Simple and repetitive gameplay which is easy to learn
- Competitive play or single-player experiences where players can compete by comparing high scores
- Abundant and frequent positive feedback for player achievements
- Tangible rewards for achievements (collectables/trophies/items)

The initial design of the application was not intended to replicate a casual game and therefore contained none of these elements. Though, as we began to restructure and revise the application, we found that it would not be successful and widely used unless it was, at its core, a fun experience. At this point we began incorporating more elements of casual games into the application and added features which demonstrated these elements.

Initial and Updated Design Features:

The design of the application was initially a simplistic GPS tracker with almost no clickable buttons. The project was meant to be much more music-based, the entirety of the design centered around the composition of musical tracks which would correspond to each "location" on the campus map. These "locations" were academic and dormitory buildings on campus and a number of outdoor areas of interest, such at the central campus fountain and the Quad. This design, and almost the entirety of the original application concept, were eventually scrapped in favor of a more focused design centered around our team's strengths. Our team was enthusiastic about the idea of delving into sound design for the application, but was inexperienced in music composition. While we did spend time educating ourselves in regard to music creation programs and techniques, it was ultimately decided that it would be a better use of our time to utilize our strengths and create an alternate design that was more interactive in nature. The result was an application with a menu allowing for increased interaction between the user and the game.

Taking our target audience into account, we redesigned the application with additional features to aid users who would not be familiar with campus. One of the major additions was a pathfinding system, with which users can specify a campus building they would like to go to and an arrow will appear to guide them in the correct direction. Since the application is designed for mobile phones, we also added a phone menu with which users can select a contact and the phone will automatically dial the number. The contacts include handy campus contacts (including Campus Police and the SNAP transportation shuttle service) and nearby food establishments which offer delivery. The contact listing dynamically adjusts depending on the time of day, as well as the day of the week, to account for when these selected services are open and available.

Trivia:

Trivia was incorporated as the core 'play' interaction in Project Overworld. The concept of trivia is assumedly well-known by the majority of players and needs little explanation. Through correctly answering trivia about the WPI campus, players are able to gain experience and level-up their avatar. Upon leveling, the player receives a new item with which to customize their avatar. The final prototype allows the player to reach a maximum of three levels, with the requirement of answering three trivia questions correctly to gain each level. Trivia questions are 'collected' by utilizing the application and are held in the 'Trivia' tab stemming off of the main menu dropdown list.

When designing our trivia system, the team had to decide how players would collect trivia questions. Initially, it was decided that having trivia questions which were location specific would increase player motivation to explore new parts of campus. We asked ourselves what the best method for trivia 'drops' would be. We came up with the following:

- Time-based: every 5 minutes or so utilizing the app the player receives a 'drop'
- Chance-based: on entering a location, the player has a 50% chance of receiving a 'drop'
- Combination: every 5 minutes or so utilizing the app, the player has a chance of receiving a 'drop'

Due to the prototype's relatively small trivia system and the resulting inability to implement location-based trivia, the team decided to utilize the combination style for 'drops'. By doing so, the chance of receiving trivia is frequent but is not guaranteed. We hope this motivates users to use the application more frequently, but will make our prototype (which currently only contains the capability for three levels) require a significant amount of time to complete.

Character Customization:

The ability to customize the player's avatar had been on the to-do list for Project Overworld since the project's creation. We wanted the player to be able to swap out their character's clothes, colors, and facial features so that they may feel properly represented within the gamespace.

The character customization slots were: shirt, head, mouth, nose, eyes, and hair. Each of these had a number of alternative styles and sprites that could be overlaid onto one another, so that together they formed a complete character. Each of these styles had a range of colors that could be chosen, ultimately allowing a wide degree of customization for each character.

The customization options were to be handed out as unlockables for answering trivia correctly and exploring the map. This would provide an incentive to actually use the app- every few questions answered meant a new wearable item, usually a hat or shirt. Facial features and skin tones would all be unlocked from the start, so that at the very least players could make a character that looks something like themselves.

Streetpass Multiplayer:

One feature which the team wanted to push was some form of always-active multiplayer, drawing particular inspiration from the "StreetPass" system featured by Nintendo's hugely successful 3DS handheld. The primary motivating factor behind trying to implement this feature is that such an addition would add a 'competitive' element and encourage students to interact with fellow app-users.

The general idea was that players' apps could connect to each other. There would be no 'live' multiplayer, but instead, each player's unique avatar would show up in other apps. These avatars would have unique looks and could be used in small minigames. Once used, the avatars would vanish until that particular player was passed again in real life.

The idea was simple enough in design. Simple online interaction makes games like Nintendo's StreetPass immensely popular. The pickup and play gameplay and real-life social interaction make them highly addicting. The social aspect provides an incentive to complete in-game activities and show off to people they can actually interact with.

However, the idea was severely flawed in terms of scope. Not only was the asset list vastly oversized with respect to the time we had, but the coding necessary to get short-range communication between any two phones was extremely difficult. Within the time constraints, we could not find a way to program a method for this sort of 'impromptu' communication between devices. The idea, in the end, had to be cut.

The multiplayer system was supposed to be another place for the character customization to come into play. The intention was for players to be able to passively receive other players' avatars in their own world, along with a little information about that player.

Music:

Project Overworld's original concept was actually a simple stylized GPS map of WPI campus, and music to accompany each unique piece of the map. Similar to the overworld maps of games such as those in the Pokemon series, each area of campus would have a tune that would play through the phone. Depending on the area the user was actually walking through, a different tune would play.

Originally, we had planned to have around 10 total pieces of music for Project Overworld. Each one would be uniquely composed by an acquaintance who had experience making background music for video games. Using free soundfonts, he composed a demo track for us to test with. We were able to create a system where certain instruments in the song would fade and grow depending on the location.

While the demo song worked well and fit the app, we were unable to produce more than the single track. The acquaintance who was contracted to produce music fell through due to circumstances out of our control. Initially, we thought we might be able to produce our own music. Professor Zizza lent us a copy of Reaper music production software, and we still had our acquaintance's Ableton license to work with. However, more complications arose. Producing nice music, as we learned, is substantially difficult. Even with access to professional production programs,

creating music was a massive time sink in both learning and composing. After some deliberation, and after the creation of some of the app's utility, we decided to move on from the original concept and focus on the features we already had expertise in. We left the demo song intact, but moved to make the app more navigation based and less recreational.

Developing for Mobile:

Developing for mobile platforms was a new endeavor for our team, and one which came with a number of unforeseen challenges.The relatively small screens of smartphones heavily influenced the visual design of the application, including the size of user interface elements and text. The team also had to take touch controls into account, and worked to place buttons on the screen in places that were comfortably reachable when holding the phone. Where possible, we implemented the use of easily understood symbols instead of utilizing text and instructions. Doing so left the screen relatively clear and kept the underlying map as visible as possible at all times.

Developing for mobile also enabled the use of incorporating the device's orientation into the design of the application. The majority of Project Overworld is presented in a landscape view, meaning the user holds the phone on its side. Doing so allowed for certain user interface elements (such as the menu) to function while not interfering with the core of the game (the map). By prioritizing keeping the center of the screen clear and the map visible, users are able to interact with the menu and observe the clock in the lower right hand corner without interrupting the core experience. Character customization, which features a standing avatar, is instead presented in portrait view (upright phone position), in order to allow the image to be as large as possible and the focus on the screen. Customization is treated as a different mode where an entirely new screen is opened and covers the map. Upon finishing and exiting customization, the screen reverts back to landscape mode.

Visual Decisions:

The visual design of the map and menus was heavily influenced by our decision to develop for mobile platforms. The resolution we chose to develop for, when scaled up, provided a crisp, pixelated look that was easily readable on mobile devices. Since it's a mobile game, we kept text to a minimum. Excessive text would place a strain on the user's eyes and clutter the UI on such a small screen.

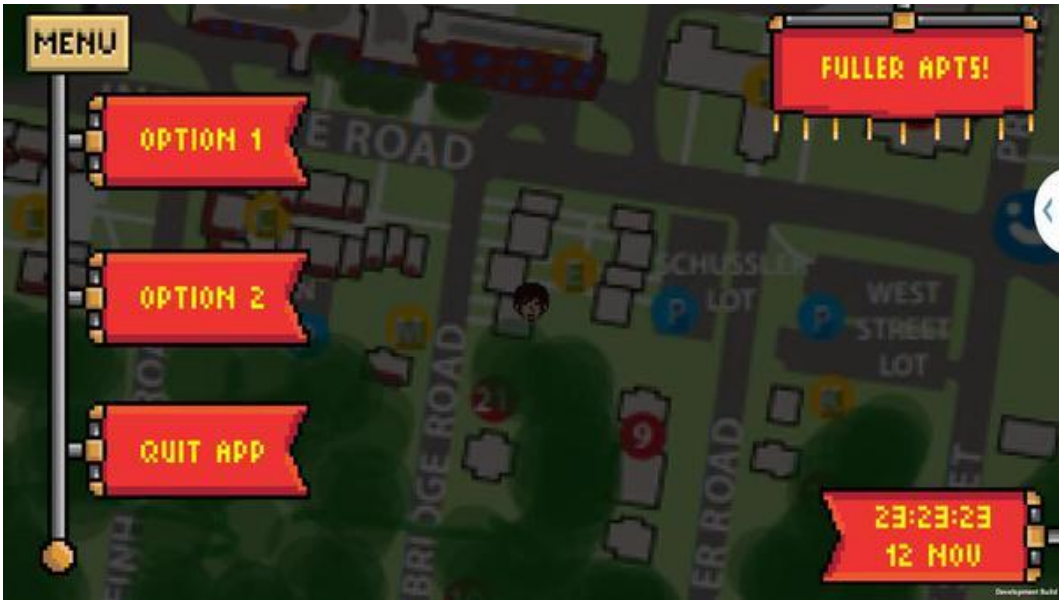
One challenge we faced during development was how to indicate to users that certain elements on the screen were clickable while others were static visuals. For buttons, we used colors that stood out, such as yellow and white. These colors attracted the eye against the dull red of the buttons, and made them easily readable. On the contrary, dull greys were used for buttons and text that was not made to be interacted with. The contrast between bright and dull made it easy to tell which buttons were made to be pressed.

Menu Design:

The menu of the application, which is the core method of navigation, was the visual element the team spent the most time designing. It was important to us that the menu be simplistic, easy to use, and would not needlessly occupy a large portion of the screen at any given time. Because the most important feature of Project Overworld is the map itself, we prioritized keeping the map visible at all times unless it was absolutely necessary to cover it with elements of the user interface. We decided on the final menu design because it was the least intrusive option.

The menu is accessed through a single button on the top left of the screen. By pressing the button, a menu drops down and presents the use with a clickable list of topics. After selecting a topic, the menu branches from the selection to the right and shows either a "page" or a set of sub-topics. The "pages" indicate that the user has reached the end of the specified branching of topics and sub-topics. Examples of these "pages" include the setting menu (which is reached by clicking the menu button > settings), the auto-dial contact menu (Covered later in Additional Features), and the pathfinding system.

The pathfinding system is a simple addition that can be reached from the menu. It brings up a list of all the locations implemented on campus, sorted by alphabetical tabs, all of which can be selected. Selecting a building brings up an arrow around your player avatar on the map, pointing towards the selected building. It functions as a sort of compass, directing the player towards the building so that those unfamiliar with campus might find their way around a bit easier.



[Figure 1: an example menu]

Because of the branching nature of the menu, earlier branches remain visible and therefore enable the user to re-select from a previous branch at any time. For example, after changing the settings in the options, a player could immediately switch to the quick-dial menu without closing the first one. This was designed as such so that the design did not need to include a "back" button and would remain visually intuitive at all times, minimizing how many touches are needed to navigate the menu.

Due to the size of the typical smartphone device, the number of options that the menu could display was limited. Deciding on what the initial branches should be was challenging, but forced the team to focus our efforts and decide what features of the application were the most critical to creating the user experience we were aiming for.

Technical Development

The First Technical Milestone: GPS Tracking

After deciding on a core concept for the project, we were tasked with deciding which features were most fundamental to the project and should therefore be implemented first, as well as which of these features might prove technically difficult to produce, and should therefore have as much time invested into as possible. After some deliberation, it was clear that these two driving focuses intersected at a single point: GPS functionality.

Fortunately, and much to the delight of our programmer, Unity and the Android IDE are both well attuned to the needs of developers, and provided easy hookups to poll the raw GPS coordinates of the device on which the app was running. Using this functionality, a quick application was compiled in Unity which was little more than a print-out of the user's current GPS coordinates; while this may seem like it was a trivial step along this project's path, this first build actually represented our first milestone towards completion.

Using this prototype build, our programmer set out to gather reference data for creating a scale map of campus, in the form of the coordinate extrema around the campus perimeter. This task was accomplished exactly as one would think: by physically patrolling the desired polygon around campus of which our app was to encompass, and by writing down the raw GPS coordinates at each vertex as they appeared on screen in a notebook. These field tests also served as an important indicator as to the degree of accuracy which the GPS hardware found inside a typical mobile device was capable of.

With this data in hand, the next step was to turn it into something useful to display to the user. Based upon the goals and desired usage of this application, 'useful' for us entailed providing real-time positional feedback across a digital area while the user traversed the physical area. For this task, two possible paths of progression might be apparent; we could produce an actual-size map of campus to place the player avatar in, and simply update the coordinates of the player avatar with the coordinates polled by GPS, or, with a bit of data manipulation, we could produce a much smaller map campus and scale the GPS coordinates polled throughout the physical bounding polygon to their equivalents throughout the digital one.

Almost needless to say, we opted for the latter. To make this option possible, a simple remap formula was needed, as reproduced below as *Figure 2*. Here, 'input' represents the value given to rescale, the range 'a' to 'b' represents the current range between which the input currently sits, and the range 'x' to 'y' represents the desired range between which the output is to sit.

```
output = x + (input - a) * (y - x) / (b - a)
```

[Figure 2: the 'remap' function]

The purpose of this function is produce an output value that is offset the same relative amount along a new range as the input value was along its initial range. In practical terms, this simple mathematical function allowed us to take a raw GPS coordinate that was between the previously-recorded extrema and rescale it to a relative position on a much smaller map, which is exactly what we did.

[Figure 3: providing useful positional feedback to the player]

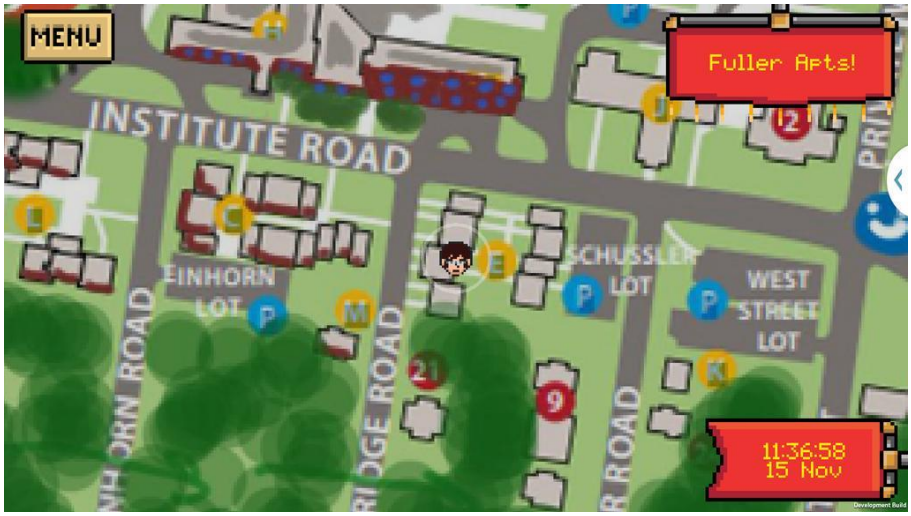
Using a quick paintover of a campus overhead map as the backdrop, another build of the application was produced and tested in the field. As *Figure 3* illustrates, this build of the application polled raw GPS coordinates, denoted as Longitude/Latitude X/Y, and rescaled these coordinates into playable in-engine space, denoted as Player X/Y. With this sturdy foundation laid out, we were ready to start fleshing out our vision of the application.

The Second Technical Milestone: Modular Development

It was during this stage in development at which one of our artists made a remark which, at the time, seemed anecdotal, but in fact proved pivotal in the way the app would be built; he stated that his parents had a friend working in the real estate industry, and that an application like ours, localized to the area in which they worked, could help this friend's business considerably. Taking this newly expanded scope of possible uses into account, our programmer set forth to ensure that the inner workings of the application were designed to be as modular as possible, to allow for potential repurposing in the future, should there be a market for it.

Because the primary focus of the application was the map itself, the primary focus of designing a modular backend was allowing for our map to be able to be easily supplanted with another map, and still retain functionality. To that end, the proprietor of the application would need to know the extrema of GPS coordinates for their new map, as well as the extrema of in-engine coordinates for their digital map. To make the latter half easier, we made use of Unity's already-existing two-dimensional polygon collider constructs, as well as code which accessed the in-engine coordinates of each vertex in the collider, and stored the vertical and horizontal extrema. With this system in place a proprietor could use open up the source files in Unity, replace the map image with their own, and visually drag the vertices of the bounding collider to match the boundaries of their newly-defined area; the only information which the system could not handle for them would be the

real-world GPS coordinate extrema of their new map. Assuming that the proprietor could provide that much information on their own, the system was set to allow for a high degree of flexibility, which proved not only useful for potential proprietors, but for simplistic implementation of our own features.



[Figure 4: location detection in real time]

Using the same polygon collider technology that was designed for the map at large, we were able to implement individual sub-locations with ease. Instead of needing to pre-program in the coordinates of each distinct on-campus location, we instead created smaller polygons that enclosed these locations, and have the player avatar send a flag when it entered one such polygon. As demonstrated in *Figure 4*, this design strategy worked wonders; because our system already scaled GPS coordinates into engine coordinates, we could safely rely on the system also moving the player to the correct locations around the map as they wandered around campus. With the basic location detection system working as desired, we added supplemental code to display the name of the current location of the player on our recently-redesigned User Interface, which marked the second major milestone of development, as our app began to more closely resemble our original vision for it.

The Third Technical Milestone: Additional Features

Now that our app was capable of both maneuvering the player avatar around a digital map in relation to the physical location of the user, and displaying the name of these locations to the user in a convenient and easily accessible manner, we were ready to begin adding additional features which would enhance the user experience. In line with our overarching goal to make this application a digital guide for new freshmen, we decided that our app should incorporate two distinct feature branches: the ability to provide users with information about the campus and its history, as well as the ability to provide ease of access to utilities and services that freshmen would find particularly useful.

To address the first of these feature branches, providing information, we developed the trivia system, which users could participate in to learn facts pulled from the "Tech Bible". By correctly answering these questions, users would be able to earn experience points with which to level up their player avatar, thereby adding the first inklings of a game to our app. While it helped us meet one of our development goals, the trivia system initially posed a threat to the previously established desire

to keep all internal systems modular, as all of the text for trivia questions would in theory need to be preprogrammed in to our app. In order to combat this threat to modularity, we decided that it would be best to redesign how all the text in the application was stored. Rather than having objects with pre-written text, all text was now pulled from an external text file using a Regular Expression upon booting the application, and was stored in a library. Objects were given text string reference handles to supplant their previous lines of text, and the system was able to look up these handles in the library and provide the corresponding text.

```
#loc_fuller_apt   {Fuller Apartments}
#loc_fuller_apt_info {Fuller Apartments are conveniently located behind
Daniels Hall and the Quad. Featuring full kitchens, these apartments are
designed to accommodate two, three, five, or seven students.}
```

[Figure 5-1: an example of a text string reference handle]

```
#trivia_q1 {Dear Worcester Tech, Our Worcester Tech, Our praises ____ to
thee!}
#trivia_q1_a1 {toll}
#trivia_q1_a2 {rise}
#trivia_q1_a3 {*ring}
#trivia_q1_a4 {sing}
```

[Figure 5-2: an example of a string reference handle for trivia]

While this new system may initially seem far more obtuse and less developer-friendly than our previous method of presenting text, it actually provides far more flexibility. *Figure 5-1* provides an example of the '`#loc_fuller_apt`' string reference handle, which was used for storing text relevant to the Fuller Apartments location. Instead of needing to make unique code to present the text found within the brackets for our location polygons around the map, we simply needed to provide them with the proper reference handle, and our system could pull the all the related text automatically. Returning to the idea of theoretically licensing out this application to proprietors for other uses, now all one would need to do is provide their own map and GPS coordinates, draw polygons around their desired locations, give these polygons a string reference handle, and edit a text file accordingly.

For the purposes of trivia, this system allowed us to continually add more questions by adding iterations to the '`#trivia_q`' string reference handle, as shown in *Figure 5-2*. Our trivia system bundled the appropriately numbered answer handles alongside the corresponding question, and identified the answer marked by an asterisk as the correct answer, while removing the asterisk from the displayed text in the process.

To address the second feature branch, providing easy access to useful utilities and services, we decided to make full use of the mobile platform by incorporating features that no other platform could provide. Namely, we added a menu which provided buttons that could automatically type in the phone number for nearby take-out restaurants and campus services, simply awaiting the user to press the "call" button.

Artistic Development

The Asset Production Pipeline:

Deciding on a unified art style for the game was the first step in the long process of art design. The goal of the app was to be an accessible map that WPI students, most likely freshmen, could use to help acquaint themselves with campus. The game had to be easy to pick up and immediately start using, so communication was important. A simplistic and charming pixel style was chosen for the map, UI, and other assets. Being colorful and vibrant would serve to distance our app from other, more straightforward map apps. The purpose of ours was to create a fun way to introduce students to WPI Campus, not provide a hardcore navigation service.

Our app's visual identity was largely inspired by classic Role Playing Games, such as Earthbound, Final Fantasy, and Pokemon. Their simplistic, isometric world maps created a fun, creative environment to explore. Our team decided that trying to capture this charming atmosphere would be key to the experience goals of our app, so imitating the visual styles of classic games' maps was an obvious step. As a result, our app's map is drawn from a unique isometric style, eschewing real-life perspective for a view that would make each building immediately recognizable from the viewpoint of the player.

Getting the buildings to be identifiable and accurate to life was challenging. The original draft of the map was a painted-over version of the stock WPI campus map found in any brochure distributed by the school. This was an extremely helpful tool, as it accurately laid out all the footprints of the local roads and buildings. With those in place, the next step was accumulating reference photos. The app's isometric view meant that buildings would be viewed from the South-East. We went around campus, taking photos of every important building from the southern face and eastern face. This way we could more easily capture the likeness of the building, such as its color, window placement, window count, and architectural details. Preexisting satellite views of campus also helped with making sure its shape in our app was true to life.

[Figure 6: the official map of WPI Campus, overlaid with hand-made sprites]

These photos helped capture the visual identity of the buildings, but actually implementing them into the app's map world was difficult. The buildings were to be simplified into an isometric view that would emphasize the side of the building and bring out its recognizable features. However, doing this caused buildings to deviate from their footprint on the pre-existing GPS map; it simply wasn't possible to make an isometric view of a building fit into a perfectly top-down footprint.

To combat this, we played with the building's geometry and shape. We attempted to retain the recognizable features of the building, altering the layout and shape slightly so that the area covered by the simplified building was roughly the same size and shape as the footprint. The result is a not-entirely-accurate, but recognizable visualization of the buildings. The shape is slightly different from their real-life counterparts, but they retain the key features that allow them to be identified.

The UI was also tricky. As the player's only form of interaction with the app, it needed to be instantly readable. We decided on a themed approach to the buttons, portraying them as flags and banners. We thought this fit in well with the 'college campus' subject matter. Naturally, we used WPI's signature colors of grey and dark red for the menu. This had the added bonus of standing out against the primarily green map in the background. We used yellow as a highlight color on the banners, drawing attention to important buttons such as the user's current location. The button text is also yellow, thanks to its visibility against the red buttons.

Feedback and Future Development

Testing and Incorporating Player Feedback:

As with the development of any game or application, public testing was a crucial step needed to help guide the direction of development. However, in trying to take this step, we were met with a rather unique challenge; because our app was location-based, we could not simply demo it at many of the available venues open to student developers. Still, some on-campus opportunities did present themselves.

During GDC's Developer Showfest, a public build of Project Overworld was distributed to interested students using QR Codes that could be scanned off of business cards. Users responded very well to the graphical and stylistic choices. Many stated that the map read well from a visual standpoint and that the project concept was novel, with some upperclassmen noting that they wished such an application existed when they were new to the school. The contacts list was the second best-received feature, garnering acclaim for its ease of use and for indicating when the various listings were open for operation. However, a primary complaint of the application was that some users found the menu buttons slightly small for their fingers. As such, we will need to figure out the perfect balance between having buttons that are large enough for ease of use, while still being small enough so as not to obscure the map.

Future Plans and Development:

Going forward, we would like to continue to work on this app such that it reaches a point where we would feel it is polished enough for widespread distribution to incoming freshmen during the 2015-2016 academic year.

A large part of this polishing process is expanding upon and refining existing features. For example, we would like to expand the range of options for character customization to allow for a wider range of player avatars. We would also like to refine our location system such that each location not only features trivia and character item 'drops' related to it in some way, but also features a unique musical track, as per our original vision. Perhaps most ambitiously, we would like to continue researching inter-device connectivity methods, and find a feasible system which would allow for multiplayer aspects in some way. Lastly, once our app is fully fleshed out, we would also be interested in potentially licensing it out to interested businesses, such as the real estate example mentioned in previous sections.