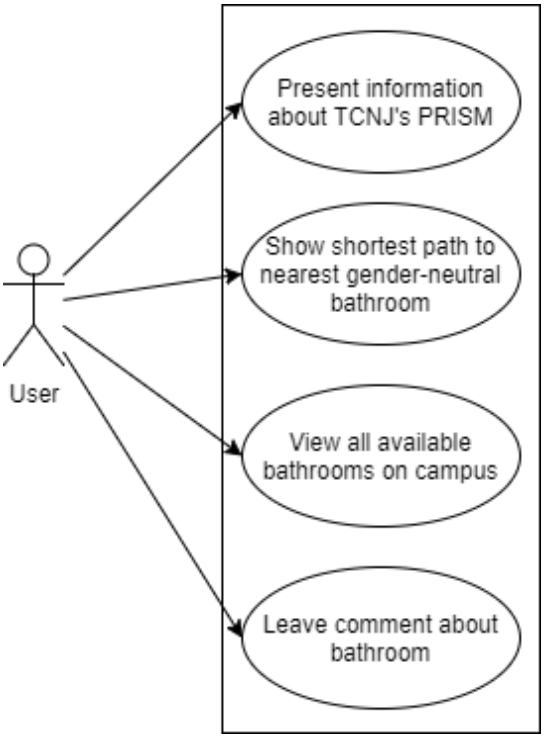
Jeffrey Sabo

11/10/17

Assignment 5: OSS: Analysis and Design

Github Link: https://github.com/saboj2/Assignment3_Sabo

Use-Case Descriptions:



Use Case 1: Find the shortest path to the nearest bathroom.

- Iteration: 1 Initial Prototype
- Primary Actor: User
- Goal in Context: To locate and present a recommended path to a gender-neutral bathroom on TCNJ's campus nearest to the current location of the user.
- Preconditions: User must have started the app, and must be looking at the home screen.
- Trigger: The User needs to use the bathroom and needs help finding a gender-neutral bathroom that they feel comfortable using.
- Scenario:
    1. The user opens up the app on the android phone.
    2. The system shows the home screen.

3. The user selects the building closest to their current location from a spinner populated by a precomputed list predetermined list.
4. The User clicks the search button.
5. The system uses Dijkstra's Algorithm to find the closest residential and non-residential bathroom vertex to the starting location.
6. The system displays the location of the found bathrooms on a map of the college campus, and highlights a path from the starting location to the found vertex locations.
7. The system presents an option for the user to view more information about the bathrooms, such as the floor the bathroom is located and reviews about the specific bathroom.
8. The system allows the user to go back to the home screen and perform a new search.

- Exceptions:
    1. A search is attempted without specifying a starting location. Proper error message is displayed.
    2. Two equidistant bathrooms are found. One bathroom is picked based on priority based on the floor it is located on, or the bathrooms are kept in a list and the user can view all bathrooms found.
    3. The bathroom found is not actually the nearest bathroom. This issue is difficult to fix because it is based on the success of Dijkstra's Algorithm. Similar to the previous exception, multiple bathrooms can be listed so that the user can view other possibilities.
    4. The switch page functionality is selected by the user. See Switch to About Page use case instead.
    5. The view all bathrooms functionality is selected by the user. See Leaving a Review for a Bathroom use case instead.
- Priority: This is the primary function of the app, so it is of very high priority.
- When Available: Third prototype
- Frequency of Use: Very Frequent
- Channel to Actor: Android-based Phone

- Secondary Actors: N/A
- Channels to secondary actors: N/A
- Open Issues:
    1. How will we be able to retrieve the path from the algorithm and use it to highlight a part of the image?
    2. Will the user be able to accurately select their current location from a list of buildings on TCNJ's campus?
    3. How will new bathrooms or buildings on campus be added to the already existing list?

Use Case 2: Leaving a review for a specific bathroom

- Iteration: 1 Initial Prototype
- Primary actor: User
- Goal in context: To let the user leave a review about a bathroom on campus.
- Preconditions: User must have started the app, and must be looking at the home screen. The phone must also be connected to the internet.
- Trigger: The user has an opinion about a bathroom located on campus and wants to let other users know about how good or bad the bathroom is.
- Scenario:
    1. The user opens up the app on the android phone.
    2. The system shows the home screen.
    3. The user chooses the option to search for a specific known bathroom on campus.
    4. The system allows the user to choose a bathroom from a list of known bathrooms on campus, and switches to a page displaying information about the bathroom including current reviews about this bathroom.
    5. The user selects the option to leave a review about the bathroom.
    6. The system presents the user with a field to write comments about the bathroom, and options to create a rating out of 5.
    7. The user enters their desired information and hits submit.
    8. The review is saved and added to the list of reviews for that bathroom.
    9. The user can go back to the home screen and exit the app.
- Exceptions:

1. The user attempts to submit a review with no rating or comments. A proper error should be displayed.
2. The user selects a starting location to search for the closest bathrooms. See the finding the shortest path use case instead.
3. The user tries to switch to the about page. See the switch to the about page use case instead.

- Priority: Medium-Low. This feature is helpful, but is a luxury and is non-essential to the app if it were to be published to the app store.
- When Available: Prototype 4
- Frequency of Use: Somewhat Frequent.
- Channel to Actor: Android-Based Phone
- Secondary Actor: N/A
- Channel to secondary actor: N/A
- Open Issues:
    1. How do we prevent reviews from being submitted if they contain inappropriate language?
    2. How do we connect the phone to the internet to load reviews submitted by other users?
    3. Do we keep the reviews anonymous, or give users credentials with which to reference their reviews?

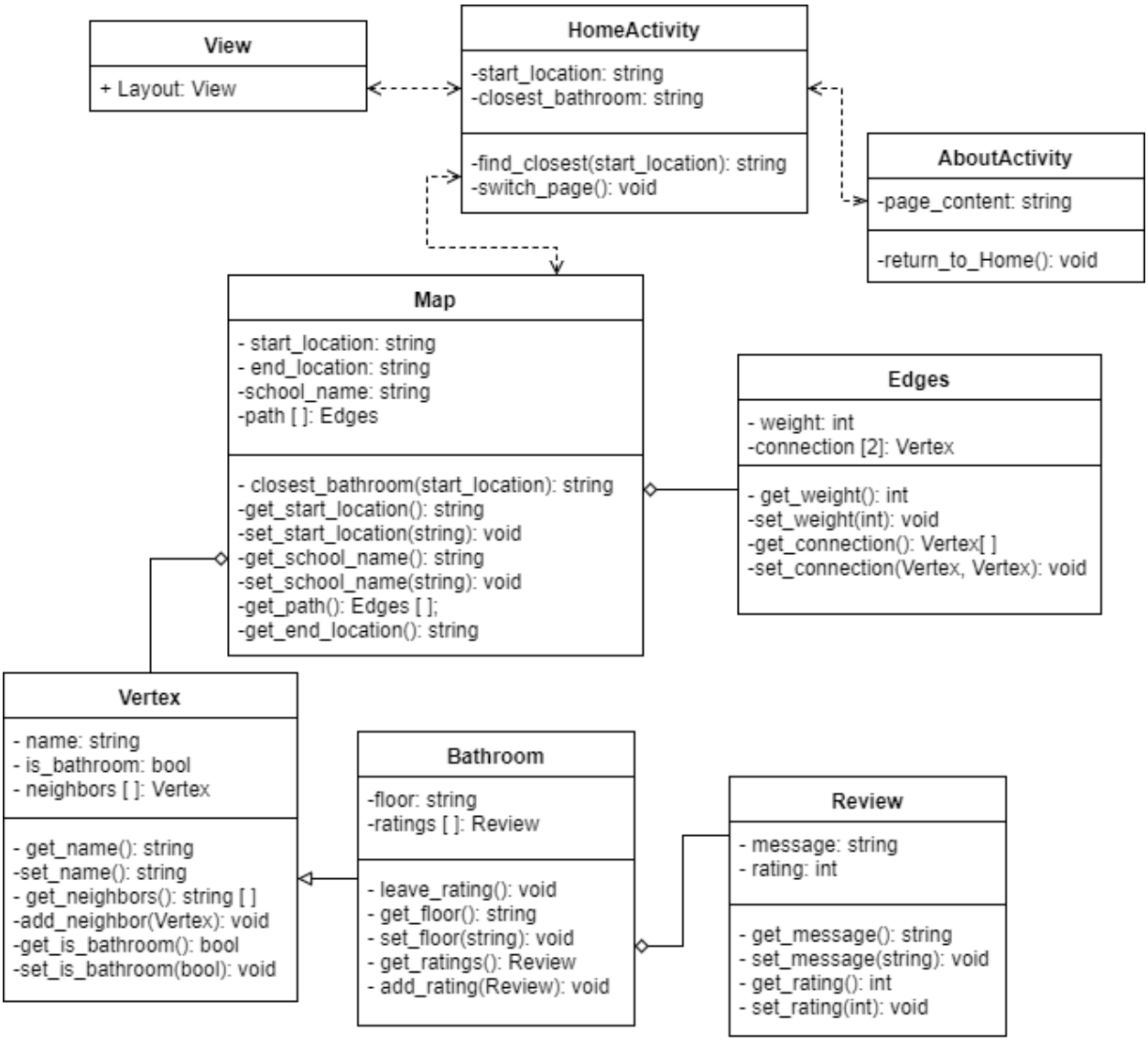
Use Case 3: Switching to the About Page

- Iteration: 1 Initial Prototype
- Primary actor: User
- Goal in context: To display information about the app and TCNJ's gay-straight alliance PRISM to the user.
- Preconditions: The user must want to view information provided by the app about resources for students in the LGBT community.
- Trigger: The user wants to find out more information about the app and gender-neutral bathrooms.
- Scenario:
    1. The user opens up the app on the android phone.

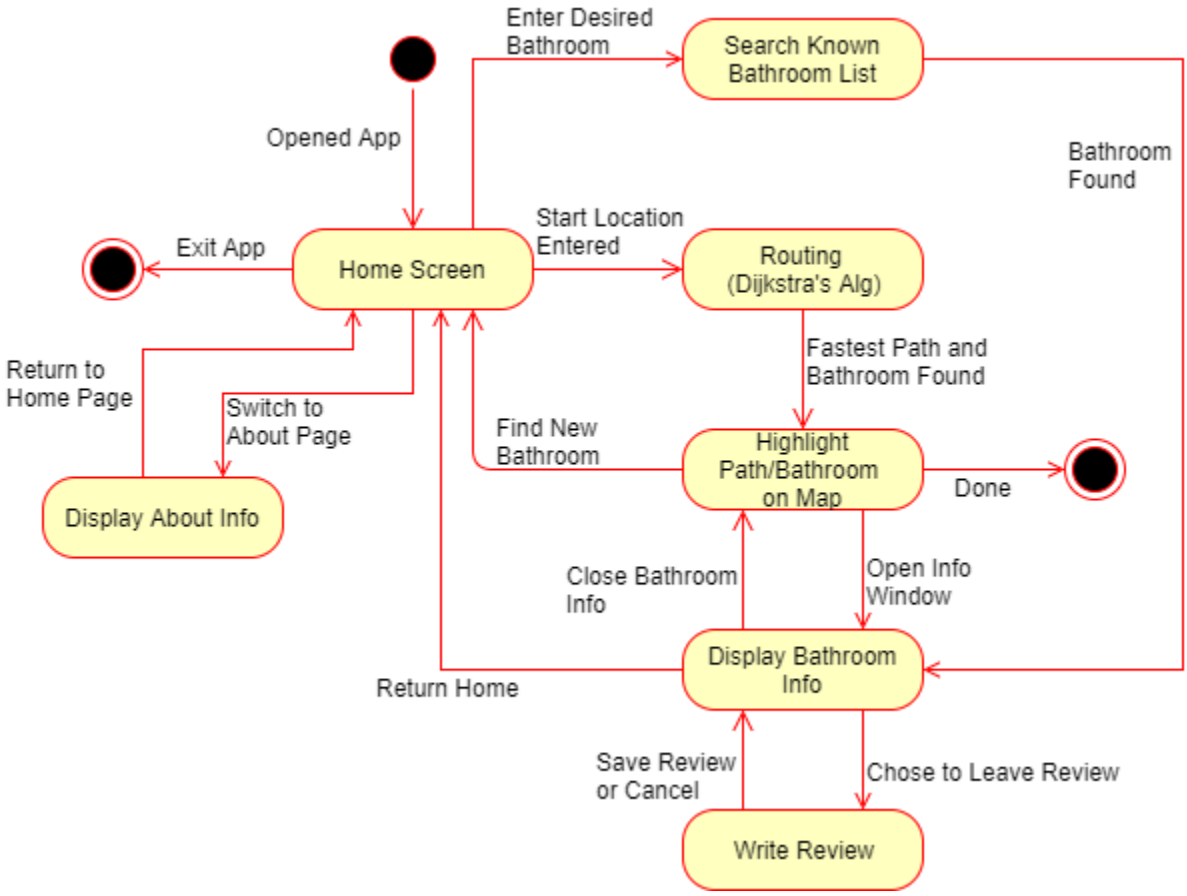
2. The system shows the home screen.
3. The user chooses the option to navigate the About page.
4. The system pauses the home screen and displays the information about the app and TCNJ's resources for transgender and non-binary identifying students.
5. The user views the information they are looking for, and chooses to return back to the home screen.
6. The user exits the app.

- Exceptions:
    1. The user selects to search for the nearest bathroom instead. See the find the nearest bathroom use case instead.
    2. The user selects to search for a bathroom and leave a review. See the leaving a review use case instead.
- Priority: Medium. The information to display is important, but this is an easy feature to implement.
- When Available: Prototype 2
- Frequency of Use: Not that frequent.
- Channel to Actor: Android-Based Phone
- Secondary Actor: N/A
- Channel to secondary actor: N/A
- Open Issues:
    1. If the user switches to the about page, will it save what was happening on the home screen?
    2. Will this page be navigated to via a menu or toolbar, or will things like screen swipes enact this page switch?

Detail Design Class Diagram:

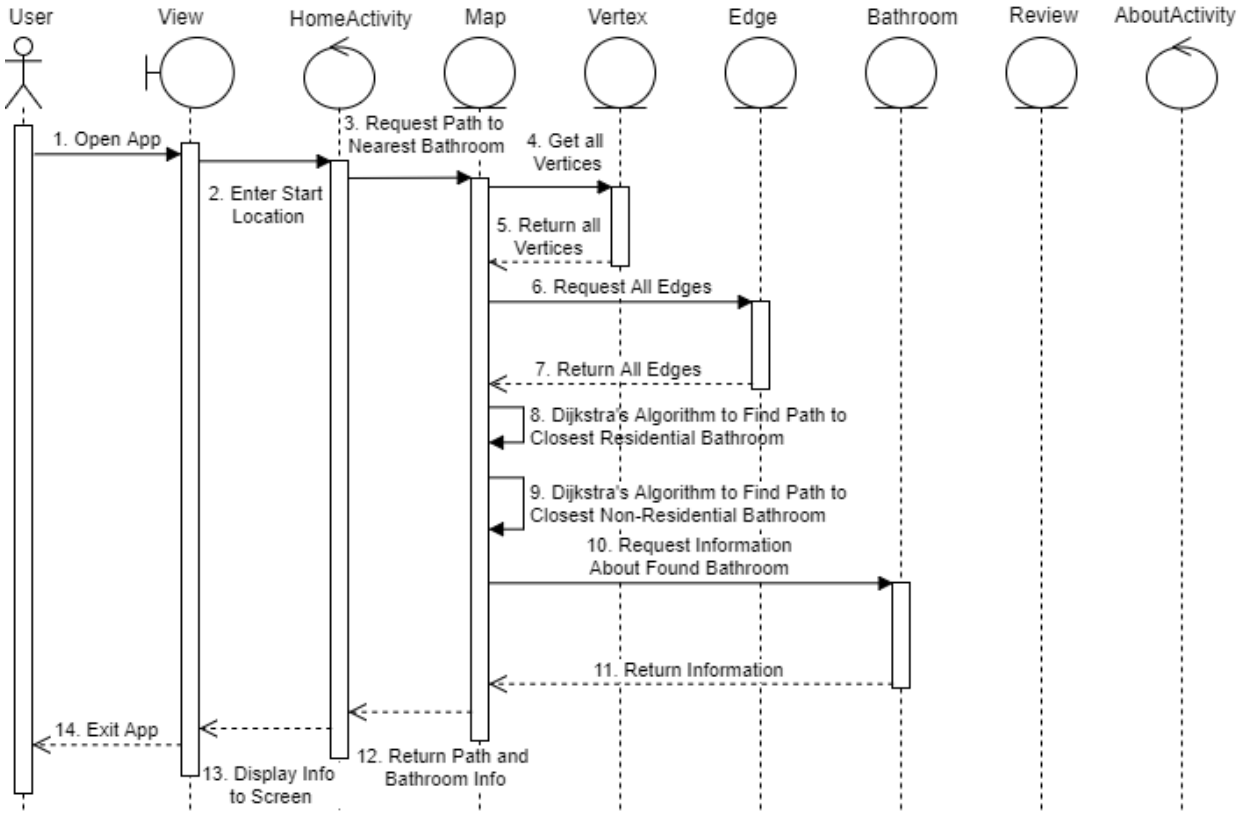


State Chart:

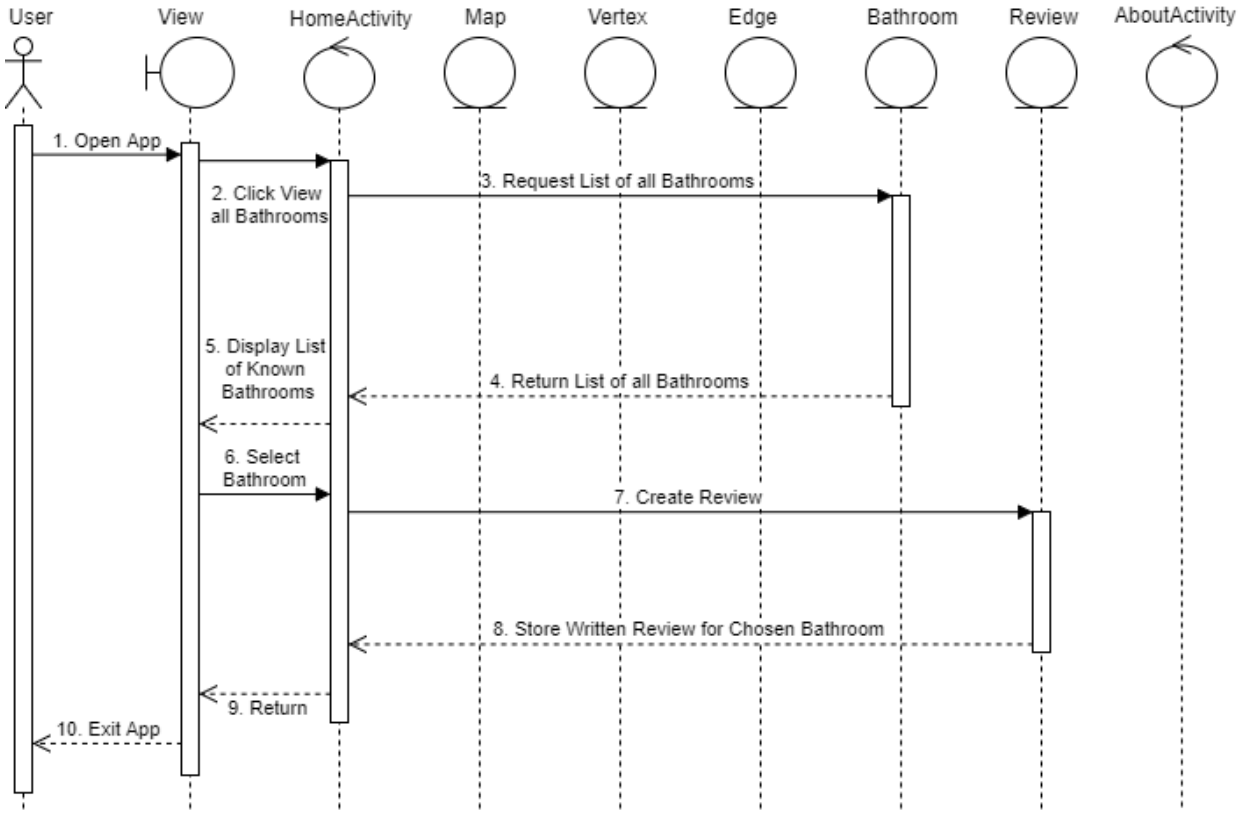

Enter Desired Bathroom
Search Known Bathroom List
Opened App
Bathroom Found
Exit App
Home Screen
Start Location Entered
Routing (Dijkstra's Alg)
Return to Home Page
Switch to About Page
Fastest Path and Bathroom Found
Find New Bathroom
Highlight Path/Bathroom on Map
Done
Display About Info
Close Bathroom Info
Open Info Window
Display Bathroom Info
Return Home
Save Review or Cancel
Chose to Leave Review
Write Review

System Sequence Diagrams:

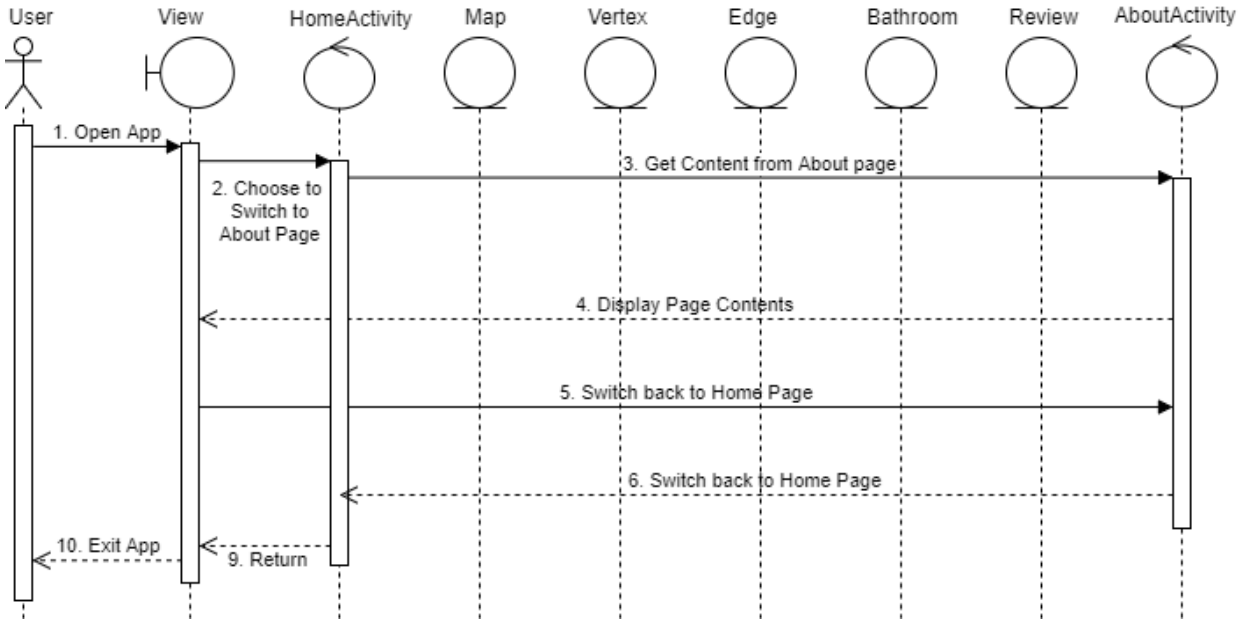
SSD 1: Find and present a path to the nearest gender-neutral bathroom.



SSD 2: Leave a review about a bathroom by searching for a specific bathroom



SSD 3: Switch to the about page to view information about the app



User Interfaces:

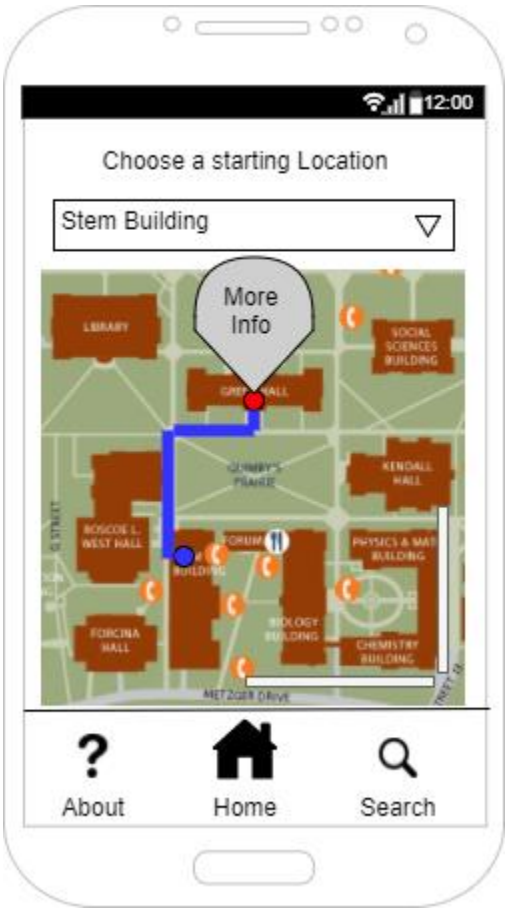
8 Golden Rules:

1. Strive for Consistency
    - The UI will use many standard conventions for inputting and displaying data. Spinners, text boxes will be used for inputting data. Displayed data will incorporate icons that make it obvious what clicking something might do. For example, the bottom navigation menu will have a magnifying glass for the search feature, and a house icon for the home page. Scroll bars will also be implemented where appropriate to indicate when there is more data to view.
2. Enable Frequent Users to Use Shortcuts
    - The most frequently used function of the app, which is the finding of the nearest gender-neutral bathroom, will be located on the home screen to allow easy access. Someone who needs to use the bathroom shouldn't have to navigate through endless menus to get to the functionality they need. The about page and search functionalities also have their own navigation icons on the bottom navigation bar.
3. Offer Informative Feedback
    - Toasts are great for giving users feedback. For example, once the algorithm has found an appropriate bathroom, a toast stating that a bathroom was found can appear on the screen. The icons on the bottom navigation screen can grow and shrink depending on what page the user is viewing to let the user know what page they are looking at. Button presses can play noises to let the user know that the button received a click input.
4. Design Dialogs to Yield Closure
    - After writing and submitting a review, a message thanking the user for leaving a review will appear, giving them the assurance that their review was received. Licensing details or the app name can be placed at the bottom of loaded text to alert the user that what they are looking at is done loading.
5. Offer Simple Error Handling
    - Methods and checks will be used to catch different errors, such as searching for a bathroom without specifying a starting location. Toasts can be used to alert the

user of the error they've encountered, and the current opened activity will remained intact.

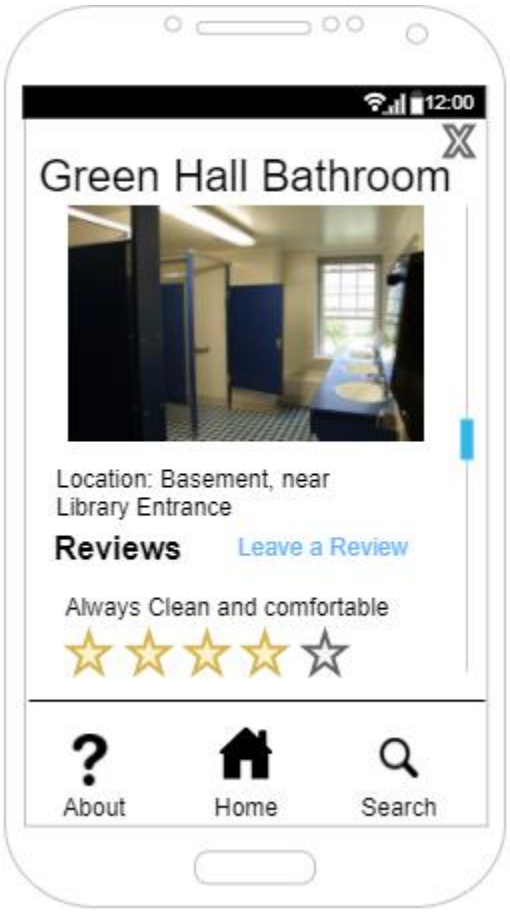
6. Permit Easy Reversal of Actions
    - Android phones include a back button that can be used to navigate through activities, but extra control should be added to allow the users to control which page they need to return to. Also, cancel or restart buttons should be included to allow the user to stop an action or dialogue that they have started by accident.
7. Support Internal Locus of Control
    - The UI will provide lots of buttons and options to allow the user to feel like they are in control. Providing a lot of buttons makes the user feel an action cannot occur without their permission, and adding options like close-out or cancel allows the user to feel like if they make a mistake they can easily fix it.
8. Reduce Short-Term Memory Load
    - The app will limit the number of places the user can input data to prevent them from needing to remember a lot of information. The main places users need to input data is a starting location for the search algorithm, and comments and ratings when leaving a review. The app will also limit the number of menus and screens so that users do not need to remember a complex navigation path to get to the function they are looking for.

UI Mockup 1: Home screen



UI Mockup 2: About Page

UI Mockup 3: Bathroom Info and Reviews page



Design Requirements:

The architecture of my app is going to follow the Model-Viewer-Controller (MVC) model. The View layer will consist of XML files that can describe the GUI. Different XML files will be used to contain the layout of different screens encountered throughout the use of the app. The Controller layer will contain the activity classes that create the connection between the XML UI actions and the data held in the Model layer of the app. These activity classes will hold method calls that instantiate different XML layout files, wait for different UI elements to be clicked, and make changes to the data held within the app based on user inputs. The Model layer will consist of a series of classes that hold data that the app will use. The most important class will be a map class that holds information and the school campus that is being navigated. The map class will hold the graph data structure that is filled with vertices and edges representing different locations around the schools campus and the distances between them. Some vertices will be bathroom objects that hold detailed information about the location and status of a gender-

neutral bathroom. Regular vertices will also exist, that will act as sidewalk junctions or buildings that don't contain a gender-neutral bathroom within them to provide more accurate routing. The use of the MVC model provides increased modularity by breaking up the major functions provided by the different files and classes. The setup of the Model layer demonstrates proper use of encapsulation by using classes to group variables and methods related to classes that will have many objects created from them. Information hiding is shown by making all the variables and methods private to prevent them from being accessed by unauthorized methods or even users.

The main algorithm being used is a variant of Dijkstra's algorithm. This algorithm is used for finding the shortest path between two vertices in a weighted graph structure. The graph in this app will consist of vertices representing different locations on a college campus, and the weights of the edges between these vertices will be representative of the physical distances between the locations the vertices represent. Modifications will need to be made to Dijkstra's algorithm to mold it to the purpose of this app. Dijkstra's algorithm is normally given a start location and an end location, but the algorithm in this app will only be given a start location and will need to find both the end location and the shortest path to the end location. Some cases may appear where the starting location is the end location as well, so the algorithm will need to make necessary checks before doing any pointless computation. The algorithm will also need to tell the difference between vertices that are residential gender-neutral bathrooms and public gender-neutral bathrooms because not every student can access certain buildings by themselves.

The main data structure that is going to be implemented by this app is the graph data structure that will hold the locations of gender-neutral bathrooms around campus and will be navigated by the main algorithm in the app. Small list structures will be used by various classes to hold data such as the neighbors of a specific vertex, the collection of edges found that represent the path to the nearest bathroom, and the list of reviews left about a specific bathroom (these may need to be stored online for the reviews to update and be viewed by all users, so a remotely hosted database may be necessary instead). Lists will be used because it is much easier to add and remove elements from a List structure than a regular array.

Test Case Design:

Android apps are all written in Java and XML, making the apps object-oriented. Therefore, unit testing must be done to all methods as part of their classes and subclasses rather

than in isolation. Similarly, integration testing is slightly different due to the nature of object oriented programs. Use-based testing is an alternate method of integration testing that introduces classes to the testing environment based on their different dependencies on other classes. Classes that are relatively independent are introduced and tested first, and gradually classes that are very dependent on the initial classes are added and tested. System testing can be done at the end of the integration testing stage when the classes have all been implemented.

The Android Studio IDE provides multiple tools and debuggers for testing both functionality and UI. Most of the test tools provided in the IDE are based around JUnit 4. Some of the major tools provided include the Espresso tool and UI Automator, as well as the JUnitRunner. All of these tools are used for automating testing of the android project, however more research must be done on how to properly use these tools. I personally own an android device, so my device can be used for testing the app on how it feels to use and how responsive it is on hardware.

Test Case Design:

Functionality Tested	**Inputs**	**Expected Outputs**	**Actual Output**
Searching without specifying a starting location.	Spinner: Nothing Search Button: Click	Error message stating nothing was entered	
Searching from a non-residential building without a bathroom.	Spinner: non-residential building selected Search Button: Click	Paths to the closest residential and non-residential gender-neutral bathroom	
Searching from a non-residential building with a bathroom	Spinner: non-residential building selected Search Button: Click	Paths to the closest residential gender-neutral bathroom, message stating current building is the closest non-residential	

Searching from a residential building without a bathroom.	Spinner: residential building selected Search Button: Click	Paths to the closest residential and non-residential gender-neutral bathroom	
Searching from a residential building with a bathroom	Spinner: residential building selected Search Button: Click	Paths to the closest non-residential gender-neutral bathroom, and message saying the current building is the closest residential	
Switching to the about page from a different page using the bottom navigation bar	About navigation icon: click	Load the about page layout	
Switching to the about page from the same page using the bottom navigation bar	About navigation icon: click	Keep the current page loaded	
Switching to the home page from a different page using the bottom navigation bar	Home navigation icon: click	Load the home page layout, remembering the previous searches made	
Switching to the home page from the same page using the bottom navigation bar	Home navigation icon: click	Keep the current page loaded	
Switching to the search page from a	Search navigation icon: click	Load the search page layout	

different page using the bottom navigation bar			
Switching to the search page from the same page using the bottom navigation bar	Search navigation icon: click	Keep the current page loaded	
Leaving a review without writing anything in the comment box	Comments: nothing Rating: between 1 and 5 stars	Error stating the there was no comment	
Leaving a review without entering a rating	Comment: appropriate content Rating: no rating	Error message stating that there is no rating given	
Leaving a review with a valid input in all inputs	Comment: appropriate content Rating: between 1 and 5 stars	Message thanking user for submission, review added to list of reviews	
Leaving a review with inappropriate language in the comments	Comment: inappropriate content Rating: between 1 and 5 stars	Error stating that a blacklisted word was found in the comment	
Opening bathroom info tab from the home screen	Found bathroom pop-up: click	Load the page containing bathroom info	
Opening bathroom info tab from the search screen	Bathroom info link in search results: click	Load the page containing bathroom info	

Scrolling the screen	Gesture: swipe up or down	Move the layout of the screen to match the desired direction	
Opening up the review submission page	“Leave a Review” link: click	Load the input boxes to leave a review	