Software Design Document

for

RUVacant

Version 0.1

Prepared by Jonnelin Marzielli Leonardo

July 2020

Table of Contents

[1.0 Introduction 4](#_Toc46244234)

[1.1 Purpose 4](#_Toc46244235)

[1.2 Intended Audience 4](#_Toc46244236)

[1.3 Intended Use 4](#_Toc46244237)

[1.4 Scope 4](#_Toc46244238)

[1.5 Product Perspective 4](#_Toc46244239)

[1.5.1 System Interfaces 4](#_Toc46244240)

[1.5.2 External Services 4](#_Toc46244241)

[2.0 Use Cases 4](#_Toc46244242)

[2.1 Actors 4](#_Toc46244243)

[2.1.1 User 4](#_Toc46244244)

[2.2 Use Cases 5](#_Toc46244245)

[3.0 Design Overview 11](#_Toc46244246)

[3.1 Introduction 11](#_Toc46244247)

[3.2 User Interface 11](#_Toc46244248)

[3.2.1 Splash Activity 11](#_Toc46244249)

[3.2.2 Options Activity 12](#_Toc46244250)

[3.2.3 Building Listing Activity 13](#_Toc46244251)

[3.2.4 Building Activity 14](#_Toc46244252)

[3.2.5 Room Activity 15](#_Toc46244253)

[3.2.6 Info Activity 16](#_Toc46244254)

[3.2.7 Navigation Drawer 16](#_Toc46244255)

[3.3 Class Design 17](#_Toc46244256)

Revisions

|  |  |  |
| --- | --- | --- |
| **Date** | **Name** | **Revision Description** |
|  |  |  |

# 1.0 Introduction

## 1.1 Purpose

The purpose of RUVacant is to inform users which classrooms within Rutgers University is empty. It will display information such as how many classrooms in a building is empty, what percentage of the selected times will the classroom be empty for, and more information about specific rooms in the building.

## 1.2 Intended Audience

RUVacant is intended to be used by traditional college students in Rutgers University. This includes both Undergraduate and Graduate students.

## 1.3 Intended Use

RUVacant is intended to be used during an active semester, when classrooms are in use. It is to be used when a student needs a currently unused classroom. However, this application only takes into case course information and not event information. Hence, the application will not be aware of event details; so if an event is placed in a classroom, the application will not know.

## 1.4 Scope

The goal of this application is to ease the process of finding an empty classroom. Students will no longer need to look at every classroom to determine whether it is being used. The student shall open the application, search for the building, and immediately discover which rooms are empty and for how long.

## 1.5 Product Perspective

### 1.5.1 System Interfaces

This application runs in Android phones and tablets

### 1.5.2 External Services

This application uses the Rutgers Course and Places API to fetch necessary data

# 2.0 Use Cases

## 2.1 Actors

### 2.1.1 User

There is only one actor for this application as there are no different types of users for this application. There is a user for the application whom will use all parts of the application. Typically, the user will be a student but it is not limited to be used only by students. Any user can use this application and can access all parts of the application.

## 2.2 Use Cases

|  |  |
| --- | --- |
| **Use case name:**  Select Options | **ID:**  UC01 |
| **Brief description:**  User selects three options | |
| **Goal:**  The user will select three options and clicks save, in which the application then proceeds with this information | |
| **Success Measurement:**  All options have one selection and the user clicks save, where the application successfully downloads and saves data | |
| **Precondition:**  All three options have exactly one selection  **Trigger:**   * The user opens the application and the application currently has no data   + The user deletes all data through the device settings and opens the application   + The user opens the application for the first time * The user deletes all data within the app | |
| **Typical flow of events:**   1. The user opens the application for the first time or the user deletes all data either within the app or outside of it and continues to use the app 2. The user selects a Semester option 3. The user selects a Campus option 4. The user selects a Level option 5. The user clicks Save 6. The application proceeds to download data and save it | |

|  |  |
| --- | --- |
| **Use case name:**  Search for Building | **ID:**  UC02 |
| **Brief description:**  The user will type in text in the search bar to search for a building | |
| **Goal:**  As the user types, the application filters the list depending on the query text | |
| **Success Measurement:**  The user finds the building they search for | |
| **Precondition:**   * Data exists in database   **Trigger:**   * User types text into search bar or clicks the search submit button | |
| **Typical flow of events:**   1. User clicks on the search bar 2. User types text into search bar 3. As user types or submits search, the list is filtered 4. User finds searched for building | |

|  |  |
| --- | --- |
| **Use case name:**  Favorite a Building | **ID:**  UC03 |
| **Brief description:**  The user favorites a building, either in the building list or in the building activity | |
| **Goal:**  The application adds a building as a favorite in the database | |
| **Success Measurement:**  The building will be added to the favorites list, and the UI star will be filled | |
| **Precondition:**   * Data exists in database * A building is not a favorite   **Trigger:**   * User clicks on the star button | |
| **Typical flow of events:**   1. User clicks on the star button | |

|  |  |
| --- | --- |
| **Use case name:**  Unfavorite a Building | **ID:**  UC04 |
| **Brief description:**  The user unfavorites a building, either in the building list or in the building activity | |
| **Goal:**  The application removes a building as a favorite in the database | |
| **Success Measurement:**  The building will be removed from the favorites list, and the UI star will be unfilled | |
| **Precondition:**   * Data exists in database * A building is a favorite   **Trigger:**   * User clicks on the star button | |
| **Typical flow of events:**   1. User clicks on the star button | |

|  |  |
| --- | --- |
| **Use case name:**  Click a Building | **ID:**  UC05 |
| **Brief description:**  User clicks (chooses) a building from the building list | |
| **Goal:**  Open a new activity and pass in the selected building’s information | |
| **Success Measurement:**  The building activity is opened and properly populated with the correct information | |
| **Precondition:**   * Data exists in database   **Trigger:**   * User clicks on a building | |
| **Typical flow of events:**   1. User opens the application 2. The building listing is presented to the user 3. The user clicks on a building 4. The user is taken to the building screen | |

|  |  |
| --- | --- |
| **Use case name:**  Fast Scroll | **ID:**  UC06 |
| **Brief description:**  The user holds and drags on an alphabetical fast scroll, which scrolls them to the appropriate location | |
| **Goal:**  The list is scrolled to the appropriate location | |
| **Success Measurement:**  The list is scrolled to the appropriate location | |
| **Precondition:**   * Data exists in database   **Trigger:**   * The user scrolls up or down on the list, which triggers the display of the fast scroll * The user holds and drags over the fast scroll | |
| **Typical flow of events:**   1. User opens app 2. User taken to building listing screen 3. User scrolls up or down the listing 4. Fast scroll appears 5. User holds and drags finger over the fast scroll 6. List is scrolled to appropriate location | |

|  |  |
| --- | --- |
| **Use case name:**  Pick a Day | **ID:**  UC07 |
| **Brief description:**  User picks a day of the week | |
| **Goal:**  User is shown a list of days of the week, who then picks a day of the week | |
| **Success Measurement:**  Correct day shows up on selection after user picks it and correct data shows up | |
| **Precondition:**   * Data exists in database * User is in either the Building Activity or Room Activity   **Trigger:**   * User clicks on the day selector | |
| **Typical flow of events:**   1. User opens app 2. User goes to either building or room activity 3. User clicks on the day selector 4. User picks a day 5. Application displays chosen day on selector 6. Application processes data and displays correct information | |

|  |  |
| --- | --- |
| **Use case name:**  Pick a Time | **ID:**  UC08 |
| **Brief description:**  User picks a time from a time picker | |
| **Goal:**  Allow user to pick a time then process and display correct data | |
| **Success Measurement:**  Correct time shows up after picking and correct data shows up | |
| **Precondition:**   * Data exists in database * User is in the building or room activity   **Trigger:**   * User clicks on the time picker | |
| **Typical flow of events:**   1. User launches app 2. User navigates to either building or room activity 3. User clicks on time picker 4. User picks a time 5. Application extracts data after picking and displays it 6. Application processes data 7. Application displays data | |

|  |  |
| --- | --- |
| **Use case name:**  Click a Room | **ID:**  UC09 |
| **Brief description:**  User clicks on a room, which then navigates to the room activity | |
| **Goal:**  After user clicks a room, a new activity is opened and the correct data shows up | |
| **Success Measurement:**  The correct room data shows up | |
| **Precondition:**   * Data exists in database * The user is in the building activity   **Trigger:**   * User clicks on a room | |
| **Typical flow of events:**   1. User launches app 2. User navigates to the building activity 3. User clicks one room 4. User is taken to the next screen, the room activity 5. Application displays correct data 6. Application scrolls to proper location depending on chosen data from previous screen | |

|  |  |
| --- | --- |
| **Use case name:**  Open Navigation Drawer | **ID:**  UC10 |
| **Brief description:**  User opens the navigation drawer | |
| **Goal:**  Open the navigation drawer when user either swiper left to right or clicks on a menu button | |
| **Success Measurement:**  Navigation drawer opens | |
| **Precondition:**   * Data exists in database * User is either in the building listing activity, building activity, or room activity   **Trigger:**   * User swipers left to right or clicks a menu button | |
| **Typical flow of events:**   1. User launches app 2. User navigates to one of activities that allow a navigation drawer 3. User swipes left to right or clicks a menu button if displayed 4. Navigation drawer opens | |

|  |  |
| --- | --- |
| **Use case name:**  Close Navigation Drawer | **ID:**  UC11 |
| **Brief description:**  User closes the navigation drawer | |
| **Goal:**  Close the navigation drawer when user clicks outside the navigation drawer | |
| **Success Measurement:**  Navigation drawer closes | |
| **Precondition:**   * Data exists in database * User is either in the building listing activity, building activity, or room activity   **Trigger:**   * User clicks outside the navigation drawer | |
| **Typical flow of events:**   1. User launches app 2. User navigates to one of activities that allow a navigation drawer 3. User swipes left to right or clicks a menu button if displayed 4. Navigation drawer opens 5. User clicks outside the navigation drawer 6. Navigation drawer closes | |

|  |  |
| --- | --- |
| **Use case name:**  Open Info | **ID:**  UC12 |
| **Brief description:**  User opens the info screen | |
| **Goal:**  Open the info screen when user clicks on the info button on the navigation drawer | |
| **Success Measurement:**  The info screen opens with correct information | |
| **Precondition:**   * Data exists in database * The user is in either the building listing activity, building activity, or room activity   **Trigger:**   * User clicks on the info button on the navigation drawer | |
| **Typical flow of events:**   1. User launches app 2. User navigates to one of activities that allow a navigation drawer 3. User swipes left to right or clicks a menu button if displayed 4. Navigation drawer opens 5. User clicks on info button 6. Application navigates to info screen | |

|  |  |
| --- | --- |
| **Use case name:**  Reselect Options | **ID:**  UC13 |
| **Brief description:**  User chooses to delete data and reselect options | |
| **Goal:**  Data is deleted and repopulated with the correct options | |
| **Success Measurement:**  Data is deleted and repopulated with the correct options | |
| **Precondition:**   * Data already exists in database * User is in one of the activities that allow the navigation drawer   **Trigger:**   * User clicks on reselect options in navigation drawer | |
| **Typical flow of events:**   1. User launches app 2. User navigates to one of activities that allow a navigation drawer 3. User swipes left to right or clicks a menu button if displayed 4. Navigation drawer opens 5. User clicks on reselection options button 6. Application deletes all data 7. Application navigates to option screen | |

|  |  |
| --- | --- |
| **Use case name:**  Sync Data | **ID:**  UC14 |
| **Brief description:**  User clicks on the sync data button and all data is deleted and repopulated with the same options | |
| **Goal:**  Data is deleted and repopulated with the same options and correct data | |
| **Success Measurement:**  Data is deleted and repopulated with the same options and correct data | |
| **Precondition:**   * Data already exists in database * User is in one of the activities that allow the navigation drawer   **Trigger:**   * User clicks on the sync data button on the navigation drawer | |
| **Typical flow of events:**   1. User launches app 2. User navigates to one of activities that allow a navigation drawer 3. User swipes left to right or clicks a menu button if displayed 4. Navigation drawer opens 5. User clicks on sync data button 6. Application deletes all data 7. Application repopulates data with the same options 8. Application navigates back to the building listing activity | |

# 3.0 Design Overview

## 3.1 Introduction

This section will contain all designs for the application, including architecture, class, and user interface.

## 3.2 User Interface

### 3.2.1 Splash Activity

A close up of a sign

Description automatically generated

### 3.2.2 Options Activity

A close up of a sign

Description automatically generatedA picture containing clock

Description automatically generated

Screen of a cell phone

Description automatically generated

### 3.2.3 Building Listing Activity

A screenshot of a cell phone

Description automatically generatedA screenshot of a cell phone

Description automatically generatedA screenshot of a video game

Description automatically generatedA screenshot of a video game

Description automatically generated

### 3.2.4 Building Activity

A screenshot of a cell phone

Description automatically generatedA screenshot of a cell phone

Description automatically generatedA screenshot of a cell phone

Description automatically generatedA screenshot of a cell phone

Description automatically generated

### 3.2.5 Room Activity

A screenshot of a cell phone

Description automatically generatedA screenshot of a cell phone

Description automatically generatedA screenshot of a cell phone

Description automatically generatedA screenshot of a cell phone

Description automatically generated

### 3.2.6 Info Activity

A close up of a sign

Description automatically generated

### 3.2.7 Navigation Drawer

A screenshot of a cell phone

Description automatically generated

## 3.3 Class Design