**Software Requirements Specification**

**for**

<OccupyUC**>**

**Version 1.0 approved**

**Prepared by <Carmen Fortino, Danielle Kritz, and Jake Ghaul>**

**<Arcana>**

**<November 2nd, 2016>**

**Table of Contents**

**Table of Contents**

**Revision History**

**1. Introduction**

1.1 Purpose

1.2 Document Conventions

1.3 Intended Audience and Reading Suggestions

1.4 Product Scope

1.5 References

**2. Overall Description**

2.1 Product Perspective

2.2 Product Functions

2.3 User Classes and Characteristics

2.4 Operating Environment

2.5 Design and Implementation Constraints

2.6 User Documentation

2.7 Assumptions and Dependencies

**3. External Interface Requirements**

3.1 User Interfaces

3.2 Hardware Interfaces

3.3 Software Interfaces

3.4 Communications Interfaces

**4. System Features**

4.1 System Feature 1

4.2 System Feature 2 (and so on)

**5. Other Nonfunctional Requirements**

5.1 Performance Requirements

5.2 Safety Requirements

5.3 Security Requirements

5.4 Software Quality Attributes

5.5 Business Rules

**6. Other Requirements**

**Appendix A: Glossary**

**Appendix B: Analysis Models**

**Appendix C: To Be Determined List**

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason For Changes** | **Version** |
|  |  |  |  |
|  |  |  |  |

# **Introduction**

## **Purpose**

The purpose of this document is to present a detailed description of the Occupy UC real-time reservation system version 1.0. It will explain the purpose and features of the system, the system interfaces, what the system will do, the constraints under which it must operate, and how the system will interact with the user. This document is intended for both the stakeholders and the developers of the system and will be proposed to Professor Schilling for its approval.

## **Document Conventions**

There are few, if any, typographical standards currently notable within this document.

## **Intended Audience and Reading Suggestions**

This document is intended for the developers and stakeholders, primarily referring to Professor Schilling who will be approving the project and advising the developers throughout the software development process. The remainder of this document contains descriptions of the interface of the software, especially the user interface, as well as a breakdown of the system requirements and features to be addressed within the development process. The structure of this document is intended so that the reader follows the order of sections as they are presented beginning with this introduction and following through the overall description to the specific features and requirements of the system.

## **Product Scope**

The OccupyUC system will be a real-time reservation system displaying the room occupancy status of all the study and classrooms accessible to Ursinus College students. The design of the system will provide a visually relevant and interactive user interface to improve usability over currently available systems. Overall, the system will be designed to facilitate the process of finding open study and work spaces for students by eliminating unnecessary physical walk-throughs of campus buildings in the attempt of locating such spaces. Making this process more efficient for students offers the benefits of increasing student productivity and lessening possible frustrations. The software will limit access to only current members of the Ursinus College community with valid identification and aims to develop an object-oriented database for storing, organizing, and retrieving relevant room status information. The goal is to provide an easy-to-use system for the campus community for obtaining studying and meeting areas.

## **References**

IEEE*. IEEE Std 830-1998 IEEE Recommended Practice for Software Requirements Specifications. IEEE Computer Society,* 1998*.*

OccupyUC Use Case Document. Version 1. November 2, 2016. Located on local machine of Danielle Kritz, member of Arcana.

# **Overall Description**

## **Product Perspective**

This software is meant to enhance existing scheduling software and provide a more user friendly interface. As the existing software is used almost exclusively by faculty, this system is intended to bridge the gap of functionality to include student users as well. The real-time component is an new element intended to make the functionality of the program more relevant and informative for the intended users. It is anticipated that this system will work on top of the current faculty reservation system to incorporate existing reservations to prevent confusion of room use and allow for a separate functionality with greater student applicability.

## **Product Functions**

The major functions of OccupyUC are included as follows:

-Clickable map of UC buildings on user interface

-Rooms Visible on Screen

-Reservation Form

## **User Classes and Characteristics**

The user is primarily intended to be a student at the college. As such, the User is expected to be internet literate, a current student of the college, and a frequent user of the system. Similarly, the User is anticipated to be able to master the system quickly and use it easily and conscientiously. He or she should be considerate of other students when using this system’s functionalities. Additional use by faculty members may be anticipated for future use, yet the faculty user is expected to be infrequent as the functionality is not as applicable as it is for students.

## **Operating Environment**

The web application will run through any internet browser i.e. Google Chrome, FireFox, Internet Explorer. The web application will be runnable on Unix, Windows, and iOS operating systems. The web application will function on any computer or tablet manufactured after the year 2010.

## **Design and Implementation Constraints (Optional)**

Our project for this project we are limited in that we can only use one data dump and try to simulate or show that a dumping of data regarding class reservations via the old system. This can be problematic, but I think we are capable of seeing how the system can handle edits to the previous data and how the User Interface reacts to those changes.

## **User Documentation**

In the course of the project, a web page with directions on how the system is used with be created.

## **Assumptions and Dependencies**

This project is dependent upon being able to retrieve information from the Ursinus’s current room reservation system called EMS. For this project we will be able to simulate the retrieval of information, with a generated spreadsheet, but will not be able to retrieve live data from the program. We can update the data on the spreadsheet thus simulating the fetching process.

# **External Interface Requirements**

## **User Interfaces**

The screen will show only the buildings that could possibly have rooms that a student could use for a study space. It will present open rooms in whatever building the student chooses on the map. Once they choose a room they will fill a reservation form that will pop up and store in that building. That room number will be removed from the open room list until it is emptied.

While this is all going on, back buttons will be on screen for the student to backtrack as need. The only software needed is a web browser to run the program and a user id log in and password to access room info.

## **Hardware Interfaces (Optional)**

The screen will have click listeners from java script which will pull up the list of open rooms provided and stored in a database. The program already set for the professors will be incorporated by adding the rooms that are taken and removing them from the list to occupy. So any device that has internet access with an ursinus login id will be able the use this on campus.

## **Software Interfaces**

The program already in place allows teachers to reserve classrooms for class and other activities. Since this is going to affect whether a room is open or not, this will be incorporated to the live room reservation system for students by dumping the rooms that are taken by the teachers, and not allowing the students to occupy them by tying in the database and displaying them using HTML and CSS to create the visual part of the program.

## **Communications Interfaces**

The system must use an internet network to access the app. The network will also access the login Id’s to verify the ursinus student and keep it safe and secure for the ursinus community

# **System Features**

## Clickable Map of Ursinus Campus Buildings

4.1.1 **Description and Priority**

HIGH PRIORITY,

There will be a map visible to users that will allow them to click on which building they want to study or look for spaces in. This allows users a fun easy way of choosing a building and also gives a rough estimate as to where in location this building is from them.

4.1.2 **Stimulus/Response Sequences**

The student will view the map and decide which building they desire and then simply use their mouse to click on it which will initiate the rest of the reservation process

4.1.3 **Functional Requirements**

REQ-1: The system shall present a map of the campus

REQ-2: The system shall have clickable buildings

## Rooms Visible on Screen

* + 1. **Description and Priority**

HIGH PRIORITY

The program will show a map of the campus with buildings that have rooms or study

spaces available to students.

This will make finding a room fast and easy. Having a map will allow students to see which building with a rough estimate (according to where the building on the map is) of how far that building might be/ choose a building close to them in the case that maybe that don’t know which building is which, they can just choose a building that has a location in which they like.

* + 1. **Stimulus/Response Sequences**

When the student opens the program, they will be asked to log in and then will be brought right to the map of the campus for them to click on the desired building. Once a building is chosen, the open rooms in that room will show in a list on screen. The student can then choose one of those rooms to occupy. When they choose a room they will be presented with a file that they have to fill out in order to reserve the room at that time

* + 1. **Functional Requirements**

REQ-3: The system shall provide a list of open rooms

REQ-4: The system shall have clickable open room numbers

REQ-5: The system shall present the reservation form

* 1. **Reservation Form**

4.3.1 **Description and Priority**

Once the student chooses a room the system will show a reservation form that they will need to fill out. This form will consist of a name text box, description text box, visibility choice, and a cancel and occupy button that will finish the form and reserve the room. High priority

4.3.2 **Stimulus/Response Sequences**

The user will choose a room which will then call to pull up the reservation form, some of the inputs on the form will be deemed optional so once the mandatory areas are filled and the user clicks occupy, the system will no longer allow that room to be occupied by another person.

4.3.3 **Functional Requirements**

REQ-5: The system shall present the reservation form

REQ-6: The system shall have a name text box on the form

REQ-7: The system shall have an optional description text box on the form

REQ-8: The system shall only allow a room to be reserved for up to three hours

REQ-9: The system shall have an occupy button on the form

REQ-10: The system shall have a cancel button on the form

# **Other Nonfunctional Requirements**

## **Performance Requirements**

The system shall display the available rooms to the user in 5 seconds or less on the majority of modern day computers, .i.e. 2010 or later.

The system shall submit room reservation forms in 5 seconds or less on the majority of modern day computers, .i.e. 2010 or later.

The system shall update the new room list within in 5 seconds or less of the edit to the rooms that are currently available on the majority of modern day computers, i.e. 2010 or later.

## **Safety Requirements**

There are no safety concerns to be concerned with the use of the application, no more than the general use of internet or your computer in general.

## **Security Requirements**

The system’s server will scan for Ursinus Secure Network IP addresses and only allow access to those that have access to Ursinus Secure via that method. Ursinus Secure itself is password protected with Ursinus ID and password.

There will be a user agreement to not share the information with non-users.

## **Software Quality Attributes**

## 5.4.1 **Reliability**

TAG: System Reliability

SCALE: The reliability that the system gives correct information about study spaces.

MUST: More than 98% of data must be accurate

PLAN: More than 99% of data must be accurate

WISH: 100% of the data is accurate

***5.4.2 Availability***

TAG: System Availability

GIST: The availability of the system when it is used.

SCALE: average system availability

MUST: More than 98% of the time

PLAN: More than 99% of the time

WISH: More than 100% of the time

TAG: Internet Connection

DESC: The application will be able to connect to the internet.

Reason: The application needs the internet to access the database.

***5.4.3 Security***

TAG: Network Security

GIST: The system shall be protected through network verification via the server.

Reason: The application has at times confidential information.

***5.4.4 Maintainability***

TAG: Application extendibility

GIST: The application must have the ability to extend existing functionality.

Reason: In order to add future functionality to the system.

***5.4.5 Portability***

TAG:Range Restrictions

GIST: The application can only be accessed through the network on campus. Which can only be accessed when you are within range of the campus routers or if you are on the network via a VPN such as Dell SonicWALL Net Extender.

Reason: This is in compliance with security restrictions laid out in 3.4.3 and 5.3.

TAG: Laptop Requirement

GIST: The system will only be runnable on laptops.

Reason: The system is made for computers, not mobile devices.

***5.5* Business Rules**

## The System shall include an option for an admin to log-in and make adjustment to certain settings in the system such as clearing rooms or over-writing reservations. Users will only be able to reserve rooms which there are rooms open, and manage their reservation status.

# **Other Requirements**

*<Define any other requirements not covered elsewhere in the SRS. This might include database requirements, internationalization requirements, legal requirements, reuse objectives for the project, and so on. Add any new sections that are pertinent to the project.>*

**Appendix A: Glossary**

*<Define all the terms necessary to properly interpret the SRS, including acronyms and abbreviations. You may wish to build a separate glossary that spans multiple projects or the entire organization, and just include terms specific to a single project in each SRS.>*

**Appendix B: Analysis Models**

*<Optionally, include any pertinent analysis models, such as data flow diagrams, class diagrams, state-transition diagrams, or entity-relationship diagrams*.>

**Appendix C: To Be Determined List**

*<Collect a numbered list of the TBD (to be determined) references that remain in the SRS so they can be tracked to closure.>*