Software Requirements Specification

Caitlyn, Powers

WSU Cpt S 322

Contents

[**1** **INTRODUCTION** 4](#_Toc129360660)

[**1.1** **Purpose** 4](#_Toc129360661)

[**1.2** **Reference Documents** 4](#_Toc129360662)

[**1.3** **Abbreviations and Acronyms** 4](#_Toc129360663)

[**GENERAL DESCRIPTION** 5](#_Toc129360664)

[**1.4** **Product** **Description** 5](#_Toc129360665)

[**1.5** **Users** 5](#_Toc129360666)

[**1.6** **Operating Environment** 5](#_Toc129360667)

[**1.7** **General Constraints, Assumptions, Dependencies, Guidelines** 5](#_Toc129360668)

[**1.8** **Design and Implementation Constraints** 6](#_Toc129360669)

[**2** **Use Cases** 7](#_Toc129360670)

[**2.1** **Use Case Diagram** 7](#_Toc129360671)

[**2.2** **Use Cases** 8](#_Toc129360672)

[2.2.1 **Use Case 1 – Login to the Application** 8](#_Toc129360673)

[2.2.2 **Use Case 2 – Load data to start application** 8](#_Toc129360674)

[2.2.3 **Use Case 3 – Exit the application** 9](#_Toc129360675)

[2.2.4 **Use Case 4 – Add User** 9](#_Toc129360676)

[2.2.5 **Use Case 5 – Check for Available Parking** 9](#_Toc129360677)

[2.2.6 **Use Case 5 – View Users** 10](#_Toc129360678)

[**3** **REQUIREMENTS** 11](#_Toc129360679)

[**3.1** **Functional Requirements** 11](#_Toc129360680)

[3.1.1 Login Interface 11](#_Toc129360681)

[3.1.2 User Settings Interface 11](#_Toc129360682)

[3.1.3 Admin Settings Interface 11](#_Toc129360683)

[3.1.4 Change password 11](#_Toc129360684)

[3.1.5 User add/ remove 11](#_Toc129360685)

[3.1.6 Car add/ remove 12](#_Toc129360686)

[3.1.7 Beacon add/ remove 12](#_Toc129360687)

[3.1.8 Connect beacons and cars 13](#_Toc129360688)

[3.1.9 Swap car/beacon 13](#_Toc129360689)

[3.1.10 Parking Statistics 13](#_Toc129360690)

[3.1.11 Parking Interface Representation 13](#_Toc129360691)

[3.1.12 Update parking data in real time 13](#_Toc129360692)

[3.1.13 Loading data for parking representation 14](#_Toc129360693)

[**3.2** **Non-Functional Requirements** 14](#_Toc129360694)

[3.2.1 **Performance Requirements** 14](#_Toc129360695)

[3.2.1.1 Parking accuracy Representation (free/occupied) 14](#_Toc129360696)

[3.2.1.2 Parking accuracy Representation (multiple beacons/cars) 15](#_Toc129360697)

[3.2.1.3 Accurate Load of parking lot information 15](#_Toc129360698)

[3.2.2 **Real time delay** 15](#_Toc129360699)

[3.2.2.1 Parking Lot Representation Update Information 15](#_Toc129360700)

[3.2.2.2 Parking Lot Representation Update Calculation 15](#_Toc129360701)

[3.2.2.3 Parking Lot Representation Update Display 15](#_Toc129360702)

[3.2.3 **Usability** 15](#_Toc129360703)

[3.2.3.1 Ease of button use 15](#_Toc129360704)

[3.2.3.2 Ease of reading the parking lot display 15](#_Toc129360705)

[3.2.3.3 Accurate user and car/beacon databases 15](#_Toc129360706)

[**3.3** **Other** **Requirements** 16](#_Toc129360707)

[3.3.1.1 Use data from firebase 16](#_Toc129360708)

[3.3.1.2 Keep track of the kind of user (admin or regular user) 16](#_Toc129360709)

[3.3.1.3 User credentials validation 16](#_Toc129360710)

# Document Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| Revision Number | Revision Date | Description | Rationale |
| 1.0 | 3/10/23 | Initial document | Initial document |
| 2.0 | 4/30/23 | Revised with more edge cases and removal of unimplemented items | Matching requirements and current software |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# **INTRODUCTION**

## **Purpose**

The purpose of this document is to outline the requirements of the Desktop Smart Parking Simulator System. This includes a description of the product, what are the users of the product, what operating environments it can run on, the assumptions and constraints that will be used to make the product, the use cases of the product, the functional requirements of the product and the non-functional requirements of the product. The features talked about in this document include the ability for the user to login, view the parking display, add and remove users, beacons, and cars if they are the admin, and change their password. Other requirements in the document include that the parking display must show information in real-time.

## **Reference Documents**

* Microsoft Learn WinForms Overview
  + [https://learn.microsoft.com/enus/dotnet/desktop/winforms/overview/?view=netdesktop-7.0](https://learn.microsoft.com/en-us/dotnet/desktop/winforms/overview/?view=netdesktop-7.0)
* Information on real time firebase use with C#
  + <https://medium.com/step-up-labs/firebase-c-library-5c342989ad18>

## **Abbreviations and Acronyms**

* SPSS – Desktop Parking Simulator System

# **GENERAL DESCRIPTION**

## **Product** **Description**

The purpose of SPSS is to track and monitor what parking spots are occupied by cars, in order to use the space more efficiently. This product includes 4 main pages, which are the login screen, the parking lot display screen, the settings screen and the administrator settings screen. The parking lot display screen shows a real time simulation of the parking spaces contained within the area made by the four sensors. The information for this is retrieved from the clients firebase database which gathers data from sensors. In order for this to work their must be beacons in the cars, that the sensors can sense.

The settings screen is for regular users and contains the change password feature. The administrator settings screen contains the change password feature, as well as the ability to manipulate and view users, cars, and beacons. The administrator settings screen is only available to the administrator.

## **Users**

There are two kinds of users, the first is called user and the second is called admin. The admin has complete control of the software, they should be a higher level administrative worker. This means they are able to add and remove users, cars and beacons. They can also connect and modify the connection between cars and beacons. They are also allowed to see the parking lot display, login, and change their password.

Regular users may only see the parking lot display, login, and change their password. They should be regular employees of the business, and the lower level administrative workers.

## **Operating Environment**

The target operating system is the Windows 10 computer. Internet access is required to be fully functional.

## **General Constraints, Assumptions, Dependencies, Guidelines**

* Should run on windows computers with internet.
* Should run as a desktop application.
* There is one admin and potentially many users.
* Properly formatted Firebase database will be available.
* There will be 4 sensors in the parking lot creating a rectangular area, with 6 parking spaces with a driving area dividing them into two rows of three spaces.
* There is a beacon in each of the cars.

## **Design and Implementation Constraints**

* To be implemented in C#
* Uses Firebase
* Uses Firebase C# library
* Should be implemented using object oriented programming
* Uses WinForms C# graphic library

# **Use Cases**

## **Use Case Diagram**

Use Case Diagram

Description automatically generated

## **Use Cases**

## **Use Case 1 – Login to the Application**

|  |  |
| --- | --- |
| Overview | |
| Title | Login |
| Description | Login to the application |
| Actors and Interfaces | User, SPSS desktop application login interface |
| Initial Status / Preconditions | Have valid user credentials and currently on the login interface |
| Steps | |
| 1. User: Type in username in username text input box 2. User: Type in password in username text input box 3. User: Press login button 4. SPSS: Validate user credentials 5. SPSS: If valid login and move to parking lot display page | |
| Post Condition | |
| The parking lot display page should be on the screen. | |
| Alternative Flow | |
| 1a. Order of typing username and password does not matter, so step 2 can be done before step one.  5.a. if invalid credentials are entered ask user to try again and steps start again from step one. | |

## **Use Case 2 – Load data to start application**

|  |  |
| --- | --- |
| Overview | |
| Title | Load data to start the application |
| Description | Load parking lot data, and beacon position data. |
| Actors and Interfaces | SPSS, firebase database |
| Initial Status / Preconditions | The SPSS desktop application must be open, user logged in to the application, and the computer must be connected to the internet |
| Steps | |
| 1. SPSS: Request information on the dimensions of the parking lot from firebase. 2. Firebase: Give dimensions to SPSS 3. SPSS: Request information on position of beacons. 4. Firebase: Give information on positions of beacons 5. SPSS: Do calculations to see if any of the beacons are in a parking space. 6. SPSS: Change the color of the parking spaces that have beacons in them. | |
| Post Condition | |
| The parking lot display indicates which spaces are filled. | |
| Alternative Flow | |
| 3a. The SPSS system does not have access to firebase error message is shown and no more steps are done.  4a. Firebase does not have information either error message shown and/or the parking spaces never change color. | |

## **Use Case 3 – Exit the application**

|  |  |
| --- | --- |
| Overview | |
| Title | Exit the application |
| Description | The user exits the SPSS desktop application. |
| Actors and Interfaces | User, SPSS |
| Initial Status / Preconditions | SPSS application is open |
| Steps | |
| 1. User: Press exit button at top right of screen. 2. Application closes | |
| Post Condition | |
| The SPSS program has terminated and is no longer open. | |
| Alternative Flow | |
|  | |

## **Use Case 4 – Add User**

|  |  |
| --- | --- |
| Overview | |
| Title | Add User |
| Description | The administrator adds a new user. |
| Actors and Interfaces | Administrator, SPSS admin settings page |
| Initial Status / Preconditions | SPSS is open and on the admin settings page |
| Steps | |
| 1. Administrator: Type in username of new user 2. Administrator: Type in password of new user 3. Administrator: Press add user button 4. SPSS: Adds user to file of users | |
| Post Condition | |
| The new user can now be seen on the list of users. | |
| Alternative Flow | |
| 1a. Administrator can type in password then type in username  4a. SPSS system doesn’t add user, instead an error message appears and Administrator must start steps from the beginning. | |

## **Use Case 5 – Check for Available Parking**

|  |  |
| --- | --- |
| Overview | |
| Title | Check for Available Parking. |
| Description | Available parking is shown on the Parking lot display. |
| Actors and Interfaces | User,SPSS |
| Initial Status / Preconditions | SPSS application is open and user has valid login credentials, and computer has access to the internet |
| Steps | |
| 1. User: Log in to SPSS application 2. SPSS: Loads data to start application 3. SPSS: calculates position of beacons and repaints parking lot display | |
| Post Condition | |
| The SPSS is on the parking lot display page and shows how many parking spaces are available by showing them as white. | |
| Alternative Flow | |
| 1a. User is already logged in so they should already be on parking lot display page  1b. If user is on settings page press parking lot display button to go back to the display. | |

## **Use Case 5 – View Users**

|  |  |
| --- | --- |
| Overview | |
| Title | View Users |
| Description | See a list of all the usernames of the users on the system |
| Actors and Interfaces | Administrator, SPSS system, including the login page, parking lost display page and admin settings page. |
| Initial Status / Preconditions | Open SPSS desktop application, and have administrator credentials |
| Steps | |
| 1. Administrator: Type in administrator login credentials and press the login button on the login page 2. Administrator: Press the setting button on the Parking lot display page 3. Administrator: Scroll down on Admin Settings page, until list of users is seen | |
| Post Condition | |
| List of users shown on screen | |
| Alternative Flow | |
| 3.a.Scrolling down may be unnecessary | |

# **REQUIREMENTS**

## **Functional Requirements**

## Login Interface

* The software should check to see if the credentials are for admin or for a regular user, to assign appropriate permissions.
* If user credential validation fails the software should show the user a message saying that the credentials are incorrect.
* The software must take username and password input and when the login button is pressed verify that a user exists with those cedentials.
* If it is correct the software must log the user in and switch to the Parking lot display page.

## User Settings Interface

* The software should go to this page if the user does not have administrator access, and they have pressed the settings button on the parking lot display page.
* The software must have the change password feature on this interface

## Admin Settings Interface

* The software should go to this page if the user has administrator access, and they have pressed the settings button on the parking lot display page.
* The software must have the change password feature, add and remove user feature, view users feature, add and remove car feature, add and remove beacon feature, connect car and beacon feature, view cars and beacons feature, swap cars feature and swap beacons feature on it.

## Change password

* If the old password entered is incorrect the software should show a message saying that the old password is incorrect.
* The software must allow for the input of the current password and a new password and when the change password button is pressed verify that the current password is correct, and if it is change the password to what was input as the new password.
* The software must have this on the admin settings interface and the user settings interface.

## User add/ remove

* The software should send an error message if there is no such user and remove user was pressed.
* The software should also send an error message if add user is pressed, and both the username and password boxes are not filled in, if there is already a user with that username, or ‘\*’ is used in the username.
* The software must put this feature only on the admin settings interface.
* The software must also allow for the input of a username and password for the creation of a new user and create the new user if entered username is unused and if add user button is pressed.
* The software must allow the input of just a username to remove a user, and remove the user if the user with that username exists and the remove user button is pressed.
* The software must also update the viewing list of users and firebase to reflect the new users or the removed users.

## Car add/ remove

* The software should send an error message if there is a car with that plate number, or all fields are not filled in, and add car was pressed.
* The software should also send an error message if remove car is pressed and the plate number is not given or if their does not exist a car with that plate number.
* The software should not remove the connected beacon if remove car is pressed.
* The software must put this feature only on the admin settings interface.
* The software must also allow for the input of a plate number, car owner and colour for the creation of a new car and create the new car if entered plate number is unused and if add car button is pressed.
* The software must allow the input of just a plate number to remove a car, and remove the car if the car with that plate exists and the remove car button is pressed.
* The software must also update the viewing list of cars and beacons and firebase to reflect the new cars or the removed cars.

## Beacon add/ remove

* The software should send an error message if there is already a beacon with that id, the beacon field was not filled in, or a positive number was not entered and add beacon was pressed.
* The software should also send an error message if remove beacon is pressed and the beacon id is not given or if their does not exist a beacon with that id.
* The software should not remove the connected car if remove beacon is pressed.
* The software must put this feature only on the admin settings interface.
* The software must also allow for the input of beacon id for the creation of a new beacon and create the new beacon if entered beacon id is unused and if add beacon button is pressed.
* The software must allow the input of a beacon id to remove a beacon, and remove the beacon if the beacon with that id exists and the remove beacon button is pressed.
* The software must also update the viewing list of cars and beacons and firebase to reflect the new beacons or the removed beacons.

## Connect beacons and cars

* The software should send an error message if there is already a beacon assigned to that car, or the beacon is already assigned to a car, and the connect button is pressed.
* The software should also send an error message if there is no car with that plate or any beacon with the given beacon id.
* The software must put this feature only on the admin settings interface.
* The software must also allow for the input of beacon id and a plate number and connect the car with that plate number to the beacon with that id if both of them exist and if the connect button is pressed.

## Swap car/beacon

* The software should send an error message if one or both of the beacon ids do not represent a current beacon and swap beacons is pressed.
* The software should also send an error message if there is one or both of the plate numbers are not used by a car and swap cars is pressed.
* The software must put this feature only on the admin settings interface.
* The software must also allow for the input of two beacon ids and swap the cars they belong to if the swap beacons button is pressed and the beacons with those ids exist.
* The software must allow the input of two plate numbers and swap the beacons they belong to if the swap cars button is pressed and the cars with those plate numbers exist.

## Parking Statistics

* The software should show how many spaces have cars that are parked delinquently accurately, through the colour of the parking spots.
* The software must show how many spots are available and unavailable accurately, through the colour of the parking spots.

## Parking Interface Representation

* The software should show the parking spots changing colour if the spot is occupied, or is delinquent.
* The software must show 6 parking spaces with a driving area in-between such that there are 2 rows of three with the driving area in-between.
* It is not necessary for the orientation of the map to match the orientation of the parking display at this time.

## Update parking data in real time

* The software should update the position data automatically regularly.
* The software must take data from the provided firebase database.

## Loading data for parking representation

* The software should accurately represent what spaces are occupied, available and delinquent.
* The software must take data from firebase, and use trilateration from the beacons to get the information for what spaces the cars are in.

## **Non-Functional Requirements**

## **Performance Requirements**

## Parking accuracy Representation (free/occupied)

* The software must show the parking space as free if the inside 70% of the space does not have beacons in it.
* The software must show the parking space as occupied if there is a beacon in that inside 70% of the space.
* Their should be an exception to the above requirements if there is a beacon in the parking space to the left or right of that inner space the software should show the parking space and the parking space next to it on that side should indicate that there may be a car parked delinquently, as shown in figure 1.
* Chart

  Description automatically generatedThe software should use trilateration with the data from firebase in order to see if there is a beacon in the inside 70% of the space.

Figure .Visual model of how space is divided in a parking space.

## Parking accuracy Representation (multiple beacons/cars)

* The software should be able to check for their being multiple beacons in the parking lot and should ensure that it is calculating the location of each beacon, and is representing whether that beacon is in a parking space or not.

## Accurate Load of parking lot information

* The software must accurately retrieve information from firebase about the dimensions of the parking lot and the size of the parking spaces.
* It must also accurately calculate the inside 70% of that parking space.

## **Real time delay**

## Parking Lot Representation Update Information

* The parking lot simulation must update its information on the position of beacons and cars at least every 200m/s.
* When first setting up the parking lot display however it may take a few seconds.

## Parking Lot Representation Update Calculation

* The parking lot simulation must recalculate the position of the beacons every time there is new information.

## Parking Lot Representation Update Display

* The parking lot simulation must change the display every time there is new information and calculations.

## **Usability**

## Ease of button use

* The software must make it clear what input boxes are required for the buttons to work.
* The software should give messages telling the user why a button did not work.

## Accurate user and car/beacon databases

* The software must maintain the car/beacon and user databases across uses of the system.
* The software should keep this information in firebase to be read from and write to it before the program exits.

## **Other** **Requirements**

## Use data from firebase

* The software must get information about the location of the beacons and the dimensions of the parking lot from the clients firebase.

## Keep track of the kind of user (admin or regular user)

* The software must know if a administrator or a regular user is logged in.
* The software should move to the admin settings page if the logged in user is the administrator and the settings button on the parking lot display page is pressed.
* The software should move to the regular settings page if the logged in user is a regular user and the settings button on the parking lot display page is pressed.

## User credentials validation

* The software must validate user credentials before allowing for a user to change their password or log in.
* The software should do this by making sure the typed in password matches the password for the user.