Cairo University  
Faculty of Computers and Artificial Intelligent

**CS251 - Software Engineering I**

Project Name

Software Requirements Specifications (SRS)

Geeks

Month & Year

Contents

[Instructions [To be removed] 2](#_Toc101814799)

[Team 2](#_Toc101814800)

[Document Purpose and Audience 2](#_Toc101814801)

[Introduction 3](#_Toc101814802)

[Software Purpose **Error! Bookmark not defined.**](#_Toc101814803)

[Software Scope 3](#_Toc101814804)

[Definitions, acronyms, and abbreviations 3](#_Toc101814805)

[Requirements 3](#_Toc101814806)

[Functional Requirements 4](#_Toc101814807)

[Non Functional Requirements 4](#_Toc101814808)

[System Models 5](#_Toc101814809)

[Use Case Model 5](#_Toc101814810)

[Use Case Tables 6](#_Toc101814811)

[Ownership Report 17](#_Toc101814812)

[Policy Regarding Plagiarism: 17](#_Toc101814813)

# 

# Document Purpose and Audience

# Introduction

**Software purpose:**

* In this software we design an application for parking garage system and handling some things:

1. Owner input number of slots in garage that he wants and enter dimension of each slot.
2. This system can calculate parking fees by using time of park in and park out
3. This system can calculate total income and number of vehicles in any point of time
4. Owner of garage handles the way that vehicle use to parking (first come or best fit).
5. This system also displays all available slots.

* Also, this software make customer enter information about car like model year, model name, license of car and its dimensions.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

## Software Scope

* **Any software could have too many components / Major features .. but we should implement specific things...this is the scope**
* **In simple points, what is the software scope (focus on components / Major features, not tiny things)**

## Definitions, acronyms, and abbreviations

* **Sensors**: a device that responds to a physical stimulus (such as heat, light, sound, pressure, magnetism, or a particular motion) and transmits a resulting impulse (as for measurement or operating a control)

# 1.0Requirements

## 1.1 Functional Requirements

**1. Login**

**2. Check logged in**

**3. Park in**

**4. Park out**

**5. Display Free Slots**

**6. Show Total Income**

**7. Display Total Cars**

**8. Get Fees**

**9. Choose Suitable Slot**

**10. Way Of Parking**

**11. Free Slot**

**12. Add Slot**

**13. Validate Ticket**

**14. get Time**

**15. Is Available**

**1.1.1**

**Login**

Enter username and password for owner of administrator and check it with local storge.

**1.1.2**

**Check logged in**

Every time user tries to use any sensitive feature in the program check first it logged in and if not make him login.

**1.1.3**

**Park in**

Make user park car by giving him ticket to specific slot.

**1.1.4**

**Park out**

Make user park car go out of parking with ticket and system calculate the charge automatically.

**1.1.5**

**Display Free Slots**

All slots with green sensor printed

**1.1.6**

**Show Total Income**

It displays all money claimed from first of the until the second of call of the function

**1.1.7**

**Display Total Cars**

It displays all cars entered the garage from the first of the day

**1.1.8**

**Get Fees**

It gives the charge of car by ticket number.

**1.1.9**

**Choose Suitable Slot**

Based on given algorithm by owner it chooses suitable slot for the coming car.

**1.1.10**

**Way Of Parking**

It detrained by owner on first time program works.

**1.1.11**

**Free Slot**

It makes specific slot sensor from red to green to be available to park again.

**1.1.12**

**Add Slot**

It specifies the dimensions of one slot or more.

**1.1.13**

**Validate Ticket**

Check if ticket number found in local storage.

**1.1.14**

**get Time**

it gets the time of enter of the car in the slot.

**1.1.15**

**Is Available**

Check if there are free slots in parking or no by checking the color of sensor in each slot.

## 2.1Non Functional Requirements:

**2.2.1 Usability:**

Ease of use by users, as the number of steps to perform a specific task takes no more than 6 clicks.

**2.2.2 Security**

Username and password are required to access the owner or admin functions and data as a kind of protection of information and using resources.

**2.2.3 Privacy**

No sensitive information will be public (e.g., Total Income) but anyone can access park in.

**2.2.4 Flexibility**

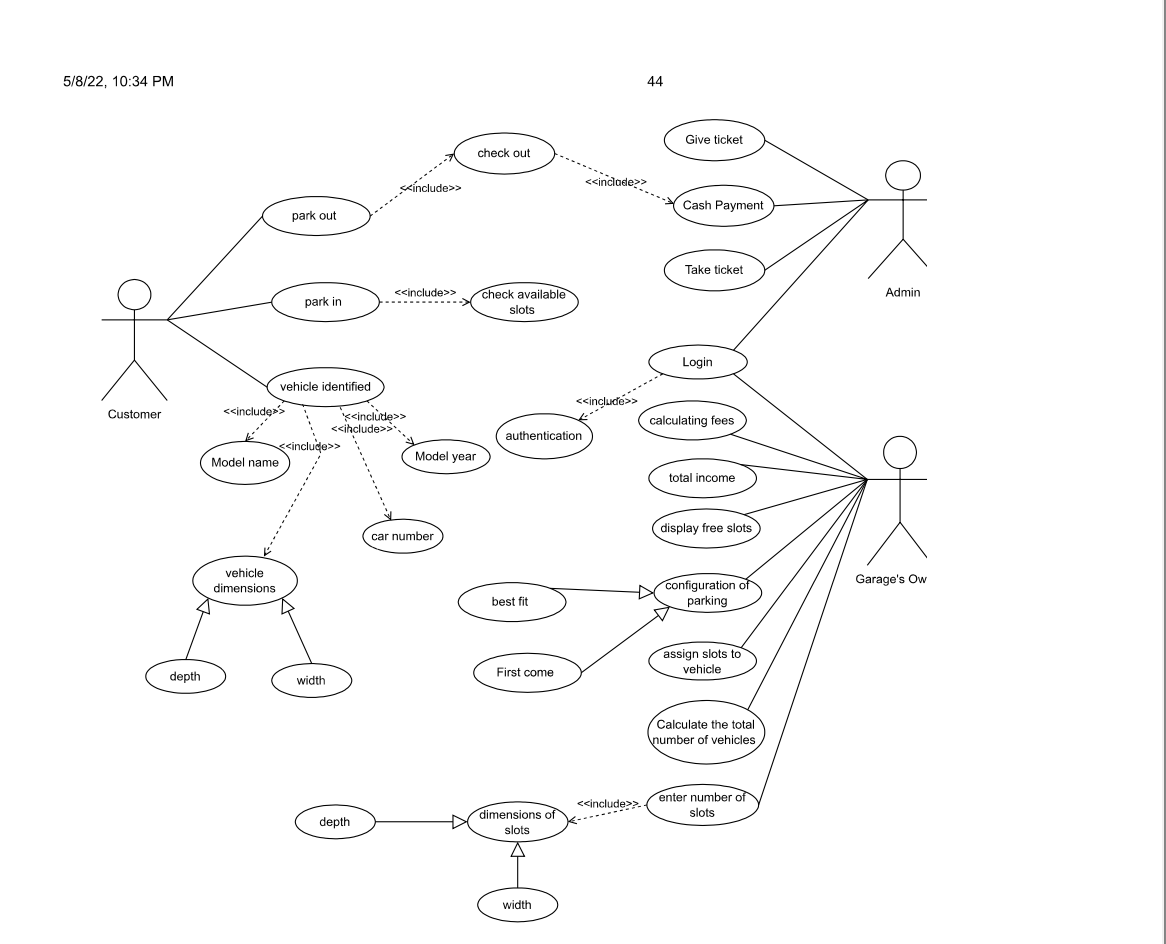
The system can interact with different users (owners and admins and customers) and easily responds to their requirements.

**2.2.5 Reliability**

The platform is estimated (on services used and algorithm working) to be running for 24 hours continuously with no failure and if system goes down, we are synchronizing changes so data loss is hard to happen.

# System Models

## Use Case Model

****

## 

## 

## Use Case Tables

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case ID: | **1** | | |
| Use Case Name: | Park out | | |
| Actors: | customer | | |
| Pre-conditions: | Exit from grage | | |
| Post-conditions: | Already exist from grage and check out. | | |
| Flow of events: | **User Action** | **Admin action** | **System Action** |
| 1- user give admin ticket. |  |  |
|  | 2-admin take ticket from user. |  |
|  |  | 3-system freeing slot to make any car can take this slot again.and make the sensor color being green. |
| 4-chechking out and give admin money. |  |  |
|  |  | 5-system calculate the total fees which based on the time-of-stay with an hourly rate of 5 EGP. |
| 6-give admin the money. |  |  |
|  | 7-take money. |  |
|  |  | 8-checked user paid all money add it to total income of the day .. |
| 9-user leave the parking after finishing all steps of payment. |  |  |
| Exceptions: | **User Action** | **Admin action** | **System Action** |
| 1-User did not find his/her ticket. |  |  |
|  | 1-admin don’t have the ticket. | 1-slot which id in ticket in system is free. |
|  |  | 2-the id in ticket is not matching with the ids number of slot in the system. |
|  |  | 3-system not found the id of car which is in the ticket. |
| Includes: | Check out. | |  |

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case ID: | 2 | | |
| Use Case Name: | Park in | | |
| Actors: | Customer. | | |
| Pre-conditions: | He/she wants park her/his car. | | |
| Post-conditions: | Already found free slot and park the car. | | |
| Flow of events: | User Action | Admin action | System Action |
| 1-user ask for parking. |  |  |
|  |  | 2-system check if there are available slot or not.  3-if system found available slot.  4-take arrival time automatic,  ask user to enter the information. |
| 5- User Enter model name, unique identification number, Model year dimensions (vehicle width and depth). |  |  |
|  |  | 6-System saved this information. |
|  |  | 7-then checked these free slots there are  suitable or not from the  dimensions which saved. |
|  |  | 8-system choose the suitable slot, s  system print ticket that include the time of arrival, id\_of\_slot and id of car. |
|  | 9-admin give user the ticket which has info of slot like id of slot. |  |
| 10-user take the ticket from admin then park his/her car in the slot which system suggest .. |  |  |
|  |  | 11-sensor’scolor of slot being red from green and increment the number of cars in these day by one. |
| Exceptions: | User Action | Admin action | System Action |
| 1-user ask for parking. |  |  |
|  |  | 2-system show message on screen there are no free slot. |
| 2-user enter information (dimensions). |  |  |
| **.** |  | 3-system show message there is no suitable slot for this dimension. And sensor of slot still being green |
| Includes: | Check available slots | |  |



|  |  |  |
| --- | --- | --- |
| Use Case ID: | 3 | |
| Use Case Name: | Display free slots. | |
| Actors: | Garage owner | |
| Pre-conditions: | No pre-conditions are required | |
| Post-conditions: | Display free slots. | |
| Flow of even: | **Owner Action** | **System Action** |
|  | 1- User clicks on Garage owner option through the screen. |  |
|  | 2- System asks for the Garage owner information. |
| 3- User Enters Garage owner info through the screen (username and password). |  |
|  | 4-System save both Garage owner’s username and password.  5-System checks for Garage owner ‘s information validations then display an approval message to the Garage owner through the screen. |
| 6-owner ask the system to display free slots. |  |
|  | 7-system search for slots with green sensor and showed it to garage owner |
| 7-show this free slot. |  |
| Exceptions: | **Owner Action** | **System Action** |
| 1- User Enters Garage owner ‘s info through the screen (username and password). |  |
|  | 2-System checks for Garage owner information validations then displays. a rejection message to the Garage owner through the screen.  2-System returns to the main page. |
| 2-there are no free slots to show |  |
| Includes: | non |  |  |

|  |  |  |
| --- | --- | --- |
| Use Case ID: | 4 | |
| Use Case Name: | Display total income | |
| Actors: | Garage owner | |
| Pre-conditions: | No pre-conditions are required | |
| Post-conditions: | The total income of the garage for a given point of time is successfully displayed. | |
| Flow of even: | **User Action** | **System Action** |
| 1- User clicks on Garage owner option through the screen. |  |
|  | 2- System asks for the Garage owner information. |
| 3- User Enters Garage owner info through the screen (username and password). |  |
|  | 4-System save both Garage owner’s username and password.  5-System checks for Garage owner ‘s information validations then display an approval message to the Garage owner through the screen.  6-System calculates total income then displays the total income in a message through the screen. |
| 7-tell user about total income. |  |
| Exceptions: | **User Action** | **System Action** |
| 1- User Enters Garage owner ‘s info through the screen (username and password). |  |
|  | 2-System checks for Garage owner information validations then displays an rejection message to the Admin through the screen.  3-System returns to the main page. |

|  |  |  |
| --- | --- | --- |
| Includes: |  |  |

|  |  |  |
| --- | --- | --- |
| Use Case ID: | 5 | |
| Use Case Name: | Display total vehicles  . | |
| Actors: | Garage owner | |
| Pre-conditions: | No pre-conditions are required | |
| Post-conditions: | Display The total number of vehicles. | |
| Flow of even: | **User Action** | **System Action** |
| 1- User clicks on Garage owner option through the screen. |  |
|  | 2- System asks for the Garage owner information. |
| 3- User Enters Garage owner info through the screen (username and password). |  |
|  | 4-System save both Garage owner’s username and password.  5-System checks for Garage owner ‘s information validations then display an approval message to the Garage owner through the screen. |
| 6- Garage owner ask system to display the number of vehicles |  |
|  | 7-System calculates total number of all cars that entered to the parking today and display it to the owner |
| Exceptions: | **User Action** | **System Action** |
| 1- User Enters Garage owner ‘s info through the screen (username and password). |  |
|  | 2-System checks for Garage owner information validations then displays a rejection message to the Admin through the screen.  3-System returns to the main page. |
| 2-garage is closed at this day.  3-sensor has a problem and doesn’t work all day |  |
| Includes: | 1-non |  |

|  |  |  |
| --- | --- | --- |
| Use Case ID: | 6 | |
| Use Case Name: | Configuration of parking. | |
| Actors: | Garage owner. | |
| Pre-conditions: | 1- garage owner shall pick a free slot based on the active slot configuration. There are two configurations (i) first come first served slots (ii) best-fit approach where you need to find the slot with the minimum dimension to hold the vehicle.  . | |
| Post-conditions: | 2-Already give user the id of slot which user should park in it. | |
| Flow of even: | **owner Action** | **System Action** |
| 1- User clicks on Garage owner option through the screen. |  |
|  | 2- System asks for the Garage owner information. |
| 3- User Enters Garage owner info through the screen (username and password). |  |
|  | 4-System save both Garage owner’s username and password.  5-System checks for Garage owner ‘s information validations then display an approval message to the Garage owner through the screen. |
|  | 6-system ask to choose the way to park car in parking. after these actions There are two configurations (i) first come first served slots. (ii) best-fit approach where you need to find the slot with the minimum dimension to hold the vehicle. Garage owner choose from its. |
| 7-owner choose one way from two ways. |  |
|  | 8- save the chosen way to use it to park car. |
| Exceptions: | **Owner Action** | **System Action** |
| 1- User Enters Garage owner info through the screen (username and password). |  |
|  | 2-System checks for Garage owner information validations then displays a rejection message to the admin through the screen. |
| 2-if Garage owner enter number doesn’t equal one or two. |  |
|  | 3- System returns to the main page. And asked Garage owner to enter the number again. |
| Includes: | 1-Best fit.  2-first come. |  |
| Notes and Issues: |  |  |

|  |  |  |
| --- | --- | --- |
| Use Case ID: | 7 | |
| Use Case Name: | Enter the number of slots. | |
| Actors: | Garage owner | |
| Pre-conditions: | Garage owner wants to full each slot in the garage with suitable car for each slot, give each slot specific dimensions. | |
| Post-conditions: | Entered the number of slots which exist in garage with its dimensions for each one. | |
| Flow of even: | owner **Action** | **System Action** |
| 1- User clicks on Garage owner option through the screen. |  |
|  | 2- System asks for the Garage owner information. |
| 3- User Enters Garage owner info through the screen (username and password). |  |
|  | 4-System save both Garage owner’s username and password.  5-System checks for Garage owner ‘s information validations then display an approval message to the Garage owner through the screen. |
| 6- Garage owner divided the garage into number of slots each one of them has the specific dimensions and id. |  |
|  | 7-system take and save the data from Garage owner. |
| Exceptions: | owner **Action** | **System Action** |
| 1- User Enters Garage owner ‘s info through the screen (username and password). |  |
|  | 2-System checks for Garage owner information validations then displays an rejection message to the Admin through the screen.  3-System returns to the main page.  4-system showed error message  . |
| 2-enter negative dimensions. |  |
|  | 4-system showed error message (enter new dimensions again) |
| Includes: | Dimensions of slot. |  |

|  |  |  |
| --- | --- | --- |
| Use Case ID: | 8 | |
| Use Case Name: | Calculating fees. | |
| Actors: | Admin. | |
| Pre-conditions: | Admin wants to calculate fees. | |
| Post-conditions: | Admin calculate total fees. | |
| Flow of even: | Admin **Action** | **System Action** |
| 1- admin clicks on Garage owner option through the screen. |  |
|  | 2- System asks for the Admin information. |
| 3- Admin Enters Garage owner info through the screen (username and password). |  |
|  | 4-System save both Admin ‘s username and password.  5-System checks Admin ‘s information validations then display an approval message to the Garage owner through the screen. |
| 6- Admin asks system to calculate fees.  7-admin enter ticket id. |  |
|  | 8-system check if the ticket id is existed or not.  9-then display the total fees of chosen car.  10-system saved the calculated fees. |
| Exceptions: | **User Action** | **System Action** |
| 1- User Enters Admin ‘s info through the screen (username and password). |  |
|  | 2-System checks for Admin information validations then displays a rejection message to the Admin through the screen.  3-System returns to the main page.  4-system showed error message |
| 2- Admin enter ticket id. | 3-ticket id not found in the system. |
| Includes: |  |  |