**Cairo University  
Faculty of Computers and Artificial Intelligence** 

**CS251 - Software Engineering I**

Parking Garage Application

Software Requirements Specifications (SRS)

Aly Eyad Aly SayedAhmed

John Gamil Samir Hakim

Silvana Yacoub GabAllah

Yara Mohamed Saad

Abdelrahman Mohamed Ramadan

May 2022

**Contents**

[Team 3](#_heading=h.30j0zll)

[Document Purpose and Audience 3](#_heading=h.1fob9te)

[Introduction 3](#_heading=h.3znysh7)

[Software Purpose 3](#_heading=h.2et92p0)

[Software Scope 4](#_heading=h.tyjcwt)

[Definitions, acronyms, and abbreviations 4](#_heading=h.3dy6vkm)

[Requirements 4](#_heading=h.1t3h5sf)

[Functional Requirements 4](#_heading=h.4d34og8)

[Non Functional Requirements 5](#_heading=h.2s8eyo1)

[System Models 6](#_heading=h.17dp8vu)

[Use Case Model 6](#_heading=h.3rdcrjn)

[Use Case Tables 6](#_heading=h.26in1rg)

[Ownership Report 12](#_heading=h.lnxbz9)

[Policy Regarding Plagiarism: 12](#_heading=h.35nkun2)

# 

# Team

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Name** | **Email** | **Mobile** |
| 2020**0331** | Aly Eyad Aly SayedAhmed | alyeyad305@gmail.com | 01092817282 |
| 2020**0132** | John Gamil Samir Hakim | johnngamil@gmail.com | 01229510148 |
| 2020**1091** | Silvana Yacoub GabAllah | silvanajackoub54@gmail.com | 01276179212 |
| 2020**0629** | Yara Muhammad Saad | yaramuhammad762@gmail.com | 01144311996 |

# Document Purpose and Audience

**This document provides a basic breakdown of the Software to be implemented, a basic textual and graphical explanation of the different aspects of the Software, and provides a reference for developers who will Implement it by defining the Software from the perspective of its Users, to ensure high quality and efficiency of time and resources during Implementation.**

**For the most part, individuals who read this report are generally not required to know specific Technical Details of this Software or how to implement it. Technical Jargon is not expected to be understood and is only included when necessary. Due to the complexity of the subject, all complex terms are almost always accompanied by their definition.**

**Although this document can also be used by technical audiences such as Developers, it is also well-curated for General and Managerial audiences, including CEOs and Managers.**

# Introduction

## Software Purpose

**The purpose of this software is to solve the given problem statement regarding Automated Parking. That is to simulate a Parking Management System that manages Parking Spaces for a configurable maximum number of Vehicles, providing a satisfactory Customer Experience when interacting with the System, and ensuring a robust Administration Experience to retrieve required Information with Speed and Ease.**

## Software Scope

* **Storing and Processing relevant Vehicle Information only.**
* **Sorting and Searching Algorithms to determine the best-fit spot for a vehicle.**
* **Payment and Transaction Services to process Credit Card and Cash Payments.**
* **Time Capture Services to get a Vehicle’s Time of Arrival and Departure from the System.**

## Definitions, acronyms, and abbreviations

|  |  |
| --- | --- |
| **Term** | **Meaning** |
| **First Come, First Serve** | **Starts from the beginning of the Parking Garage, and checks Parking Spots sequentially till the First Fitting Spot is found.** |
| **Best-Fit Approach** | **Sorts all Available spots according to their Dimensions, then searches for the optimal Spot for a Vehicle.** |

# Requirements

## Functional Requirements

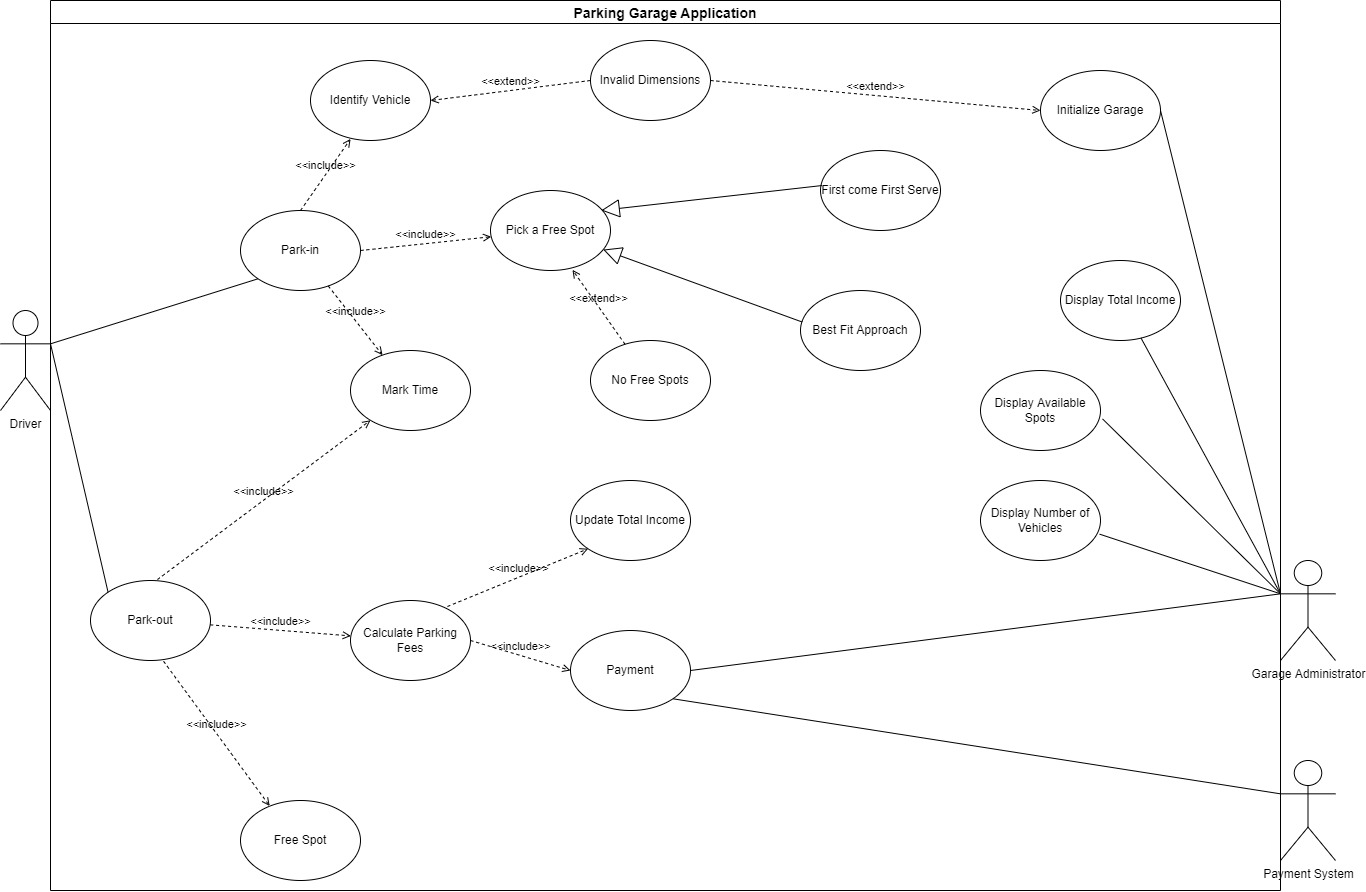
* **The Garage Administrator must be able to Initially Setup their Garage Prior to use, by setting its maximum capacity, as well as the specifications of every parking spot**
* **The Vehicle Driver must be able to register their vehicle’s information when parking.**
* **The Garage Controller must search for a valid parking spot for the driver, based on the search criteria specified by the Garage Administrator**
* **The driver should be notified clearly if there isn’t any suitable slot.**
* **The Garage Controller must precisely capture system time when parking a vehicle in or out and use this information to calculate the vehicle’s duration of stay and parking fees.**
* **The Vehicle Driver must be able to request the release of their vehicle, by passing its information to the System.**
* **The Garage Controller should mark the driver’s slot as a free spot when he parks out.**
* **The Vehicle Driver must know the fees when he parks out according to the time marked.**
* **When a Driver parks out the system should add his fees to the total income.**
* **The Garage Administrator must be able to display all currently available parking spots in the Garage.**
* **The Garage Administrator should be able to know the number of vehicles using the garage at any moment of time, as well as the total income.**
* **The Payment System must be able to successfully process Payment Transactions.**

## Non Functional Requirements

|  |  |
| --- | --- |
|  | **Details** |
| **Usability** | * **Users must be able to finish their choice of action in no more than 5 clicks.** |
| **Performance** | * **All Searching Operations must be finished in no more than 15 Seconds.** * **The System must support up to 10 Vehicle Parking Operations at once.** |
| **Robustness** | * **The System must display a clear error message to the User in case of any invalid input and should never terminate due to invalid input.** * **The System must allow up to 3 trials with every invalid input entered by the user.** |
| **Safety** | * **The System must add and remove Vehicles without tampering with existing Vehicles.** * **The System should mark the free and taken slots accurately in order to avoid problems during parking.** |
| **Supportability** | * **The System should support changes to the Method of Finding Free Spots** |

# System Models

## Use Case Model

****

## Use Case Tables

|  |  |  |
| --- | --- | --- |
| Use Case ID: | UC.1 | |
| Use Case Name: | Park-in | |
| Actors: | Driver, ParkIn Controller | |
| Pre-conditions: | The Driver wants to park his car, so he presses the park-in button. | |
| Post-conditions: | The Vehicle is transported to its spot. | |
| Flow of events: | **User Action** | **System Action** |
| 1- The Driver chooses to Park-in |  |
|  | 2- System shows the user a form to identify the Vehicle |
| 3- The Driver fills the Form by entering the model name, unique identification number, Model year and vehicle dimensions. |  |
|  | 4- The System confirms the form.  5- The System directed the Driver to pick a spot.  6- If a spot is found, the system activates setArrival functionality to set the Vehicle’s Time of Arrival.  7- The system increments number of vehicles functionality to increase the number of vehicles currently in the garage.  8- Driver is shown a message indicating the success of the operation. |
| Exceptions: | **User Action** | **System Action** |
| 1- The Driver fills the Form |  |
|  | 2- The Data in Form was Invalid Data.  3- System rejects to direct the Driver to pick a spot. |
| Includes: | 1. Pick a Free Spot 2. Identify Vehicle 3. Mark Time 4. Increment Number of Vehicles | |
| Notes and Issues: | * The System Should be able to handle Wrong Forms, allowing up to 3 tries, and displaying an error message to the Driver. * It takes 30 seconds to process the Form validation. | |

|  |  |  |
| --- | --- | --- |
| Use Case ID: | UC.2 | |
| Use Case Name: | Pick a Free Spot | |
| Actors: | Vehicle Driver, ParkIn Controller | |
| Pre-conditions: | The Vehicle Driver requests the park-in function and successfully enters their Vehicle’s information | |
| Post-conditions: | The Controller shows a success message to the driver indicating a spot was found and proceeds in the park-in function to accept the Vehicle. | |
| Flow of events: | **User Action** | **System Action** |
| 1- The Driver enters their Vehicle Information |  |
|  | 2- The Controller activates the performFind functionality using the Finding Algorithm chosen during setup. |
|  | 3- If there is a free spot based on choosing one of two approaches, the system activates assignVehicle functionality |
|  | 4- The result of the Search is displayed to the User via a message |
|  | 5- The Spot Number is returned to the getSpots function, to proceed with accepting the Vehicle. |
| Exceptions: | **User Action** | **System Action** |
| 1- The Driver enters their Vehicle Information |  |
|  | 2- The Controller finds no free spots that match the configuration method.  3- The Controller rejects the Vehicle and displays a message to the Driver accordingly.  4- The Status of Failure is returned to the park-in Function. |
| Includes: | None | |
| Notes and Issues: | * The Spot Search must be finished in no more than 5 seconds. * Exceptions when No Free Spots are available should be handled without compromising Data of Existing Vehicles, by rejecting the Vehicle and displaying a clear Error Message to the User to avoid assigning the Vehicle to an already-occupied Spot. | |

|  |  |  |
| --- | --- | --- |
| Use Case ID: | UC.3 | |
| Use Case Name: | Park Out | |
| Actors: | Driver, ParkOutController | |
| Pre-conditions: | The Driver Has Already Parked in The Garage | |
| Post-conditions: | The Driver is redirected to the Payment function. | |
| Flow of events: | **User Action** | **System Action** |
| 1- The driver enters his car’s information. |  |
|  | 2-The system checks the validity of information.  3-If the information is correct.  4- The system performs the park out by:  5-The system finds the spot of the vehicle.  6- The system marks the spot as free spot.  7- The system sets the departure time of the vehicle.  8- The system gets the time difference using arrival and departure time.  9- The system calculates the fees the shows them to the user. |
| 10- The driver stops to pay the fees |  |
|  | **User Action** | **System Action** |
| Exceptions: | 1- The Driver identifies his vehicle |  |
|  | 2- The Data in Form was Incorrect Data.  3- System rejects to direct the Driver to Park-Out. |
| Includes: | 1. Free spot 2. Mark Time 3. Calculate Parking fees | |
| Notes and Issues: | * The system should check the correctness of the car information first. * Calculation of fees must be done within maximum 15 sec. * Setting the spot free and marking the departure time should be done accurately | |

# Ownership Report

| **Item** | **Owners** |
| --- | --- |
| Functional / Non Functional Requirements | All |
| Use Case description | All |
| Use Case Model | All |
| Document Purpose and Audience | All |
| Introduction | All |

# Policy Regarding Plagiarism:

**Students have collective ownership and responsibility of their project. Any violation of academic honesty will have severe consequences and punishment for ALL team members.**

1. تشجع الكلية على مناقشة الأفكار و تبادل المعلومات و مناقشات الطلاب حيث يعتبر هذا جوهريا لعملية تعليمية سليمة
2. ساعد زملاءك على قدر ما تستطيع و حل لهم مشاكلهم فى الكود و لكن تبادل الحلول غير مقبول و يعتبر غشا.
3. أى حل يتشابه مع أى حل آخر بدرجة تقطع بأنهما منقولان من نفس المصدر سيعتبر أن صاحبيهما قد قاما بالغش.
4. قد توجد على النت برامج مشابهة لما نكتبه هنا أى نسخ من على النت يعتبر غشا يحاسب عليه صاحبه.
5. إذا لم تكن متأكدا أن فعلا ما يعد غشا فلتسأل المعيد أو أستاذ المادة.

فى حالة ثبوت الغش سيأخذ الطالب سالب درجة المسألة ، و فى حالة تكرار الغش سيرسب الطالب فى المقرر.