**Parking Garage Management System**

Software Requirements Specification

Revision History

| **Date** | **Revision** | **Description** | **Author** |
| --- | --- | --- | --- |
| 9/14/2025 | 1.0 | Initial Version |  |
| 9/18/2025 | 1.1 | 50 Minutes looking over our SRS document and filling out parts of all sections | everyone |
| 9/22/2025 | 1.2 | meeting for 40 minutes, discussed roles implementation for completion of several components of the srs, brainstormed system design of the system, modified srs and discussed classes. Assign tasks. | everyone |
| 9/24/25 | 1.3 | 1 hr Meeting looking over work on the class diagrams, created rough draft of the use case diagrams, modified use case specification sheet, modified srs functional&non-functional definitions | everyone |
| 9/25/2025 | 1.4 | 1.5hr Meeting discussion on SRS, diagrams and the over parking garage system | everyone |
| 9/26/2025 | 1.5 | 2 hr Meeting finished sequence diagram, use case diagram, use case specification and SRS | everyone |
| 9/27/2025 | 1.6 | 1.5hr Meeting finishing up everything. Started slide for presentation | everyone |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Table of Contents

**1.** **Purpose 4**

1.1. Scope 4

1.2. Definitions, Acronyms, Abbreviations 4

1.3. References 4

1.4. Overview 4

**2.** **Overall Description 5**

2.1. Product Perspective 5

2.2. Product Architecture 5

2.3. Product Functionality/Features 5

2.4. Constraints 5

2.5. Assumptions and Dependencies 5

**3.** **Specific Requirements 6**

3.1. Functional Requirements 6

3.2. External Interface Requirements 6

3.3. Internal Interface Requirements 7

**4.** **Non-Functional Requirements 8**

4.1. Security and Privacy Requirements 8

4.2. Environmental Requirements 8

4.3. Performance Requirements 8

# Purpose

This document outlines the requirements for the Parking Garage Management (PGM) System.

## Scope

This document will catalog the user, system, and hardware requirements for the PGM system. It will not, however, document how these requirements will be implemented.

## 1.2. Definitions, Acronyms, Abbreviations

PGM: Parking garage management

API: License plate reader API

## 1.3. References

Use Case Specification Document – Step 2 in assignment description - Tuan Kegang

UML Use Case Diagrams Document – Step 3 in assignment description - Chris

Class Diagrams – Step 5 in assignment description - Diego

Sequence Diagrams – Step 6 in assignment description - Matt

## Overview

The Parking Garage Management (PGM) System is designed to manage and monitor parking garages. It automatically records each vehicle’s license plate, entry time, exit time, and calculates the appropriate parking fees. Operators can view records and generate reports. By automating tracking and billing, the system reduces errors and increases efficiency for parking garages

# Overall Description

## Product Perspective

The PGM system is meant to automatically facilitate vehicle parking in parking garages. It uses a license plate reader to capture a vehicle’s license plate, then charges a fee based on the amount of time spent in the parking garage. Upon exit, the payment module will calculate the fee and prompt the user to pay. Upon successful payment, the gate will open.

## Product Architecture

The system will be organized into three major modules: the user interface module, the database module, and the billing module.

## Product Functionality/Features

The high-level features of the system are as follows (see section 3 of this document for more detailed requirements that address these features):

## Constraints

The amount of vehicles that can be actively in use in the system is constrained by the physical space of the parking garage.

Height restrictions that limit the vehicle types that can be accommodated.

Retain a privacy policy to ensure user privacy when paying through a credit card.

All vehicles will be under 3200 kg

The maximum clearance height is 8’2”.

Traffic spikes restrict one way movement into the parking garage.

## Assumptions and Dependencies

Banking/Payment Processor

Payment processing fees take a cut of the revenue.

System downtime results in lost revenue.

There will always be more than one parking spot in a parking garage

There will be an assumption of a usable credit card payment system (won’t be implemented in this project)

The PGM system will use an assumed license plate reader API to read the license plate of vehicles (won’t be implemented in this project)

The garage will be open 24/7

# Specific Requirements

## Functional Requirements

### 3.1.1 Common Requirements:

3.1.1.1 SR10 Maintain a live count of parked cars

3.1.1.2 SR11 Sufficient, involatile memory to record cars entry/exit details.

### 3.1.2 User Interface Module Requirements:

3.1.2.1 SR20 Authenticate Operator log in.

3.1.2.2 SR21 A button to generate reports.

3.1.2.3 SR22 Display payment options (credit/debit).

3.1.2.4 SR23 Display payment confirmation or failure.

3.1.2.5 SR24 Display clear, legible text at a font size reasonable for those with sight difficulties to read.

3.1.2.6 SR25 Automatic logout from the operator after 30 seconds of no input.

3.1.2.7 SR26 Manual override for gate control (in case of failure).

### 3.1.3 Database Module Requirements:

3.1.3.1 SR30 Record car license plate into the database upon entry.

3.1.3.2 SR31 Record entry/exit time into the database.

3.1.3.3 SR32 Record Fee.

3.1.3.4 SR33 Keep track of revenue.

3.1.3.5 SR34 Keep track generated reports

3.1.3.6 SR35 Search for specific car by license plate.

### 3.1.4 Billing Module Requirements:

3.1.4.1 SR40 Calculate parking fee; $2/hour or $10 daily maximum.

3.1.4.2 SR41 Provide billing options (credit/debit card)

3.1.4.3 SR42 Handle payment processing

## 

## 3.2 External Interface Requirements

3.2.1 SR50 The system must provide a driver user interface for customers

3.2.2 SR51 User interface shows their duration of stay, total fee, and payment options.

3.2.3 SR52 The system must provide a user interface for operators to log in. Operators can search for a car by license plate, view the details/histories of the car and print reports.

3.2.4 SR53 Require a display in entrance to display # of spaces.

3.2.5 SR54 Require a display at the entrance to display hourly rate and day rate.

## 3.3 Internal Interface Requirements

3.3.1 SR60 The system must keep a record of reports generated by operators.

3.3.2 SR61 Upon entry/exit, must update occupancy count.

3.3.3 SR62 The billing module must communicate payment status (success/failure) to the user interface.

# Non-Functional Requirements

## 4.1. Security and Privacy Requirements

4.1.1 SR70 System must encrypt operator’s username and password.

4.1.2 SR71 System must process credit/debit card payments securely.

4.1.3 SR72 Gate opens upon successful payment.

4.1.4 SR73 Gate closes when the sensor does not detect a car under the gate.

## 4.2. Environmental Requirements

4.2.1 SR80 The system requires uninterrupted access to internet services in order to function.

4.2.2 SR81 Requires a license plate reader

4.2.3 SR82 Require one-way tire spike in entrance

4.2.4 SR83 Physical sign in front displaying the rates.

## 4.3. Performance Requirements

4.3.1 SR90 Processes user credit card under 30 seconds.

4.3.2 SR91 Display “Thank You” after fee paid

4.3.3 SR92 Time calculation should be accurate to the minutes to ensure accurate fee calculation, fee should be rounded to 2 decimals.

4.3.4 SR93 Being able to store the information of 17,000 cars in less than 20 minutes.

4.3.5 SR94 Being able to store records (# of cars, average stay, etc.) up to 30 days and generate reports from such records

<https://docs.google.com/presentation/d/1RLb8P_rrBseDSzUFbNVdHD_3QSceWJ8TnijF_eJY4no/edit?usp=sharing>

Classes:

<https://lucid.app/lucidchart/fd651a86-c4c3-4987-93ac-201cfa4bfb5b/edit?invitationId=inv_59cbd022-ab12-4927-bc55-1268e598f142>

**Garage**:

* Attributes: spaces available
  + fee rates
  + amount of cars inside of garage
  + closing and opening time
* Methods:
  + print Report

**Car**:

* Attributes:
  + license plate
  + **Ticket**(entry/exit time, fee )
* Methods:

OperatorInterface

Attributes: User C

Methods:

User Interface

Payment

method of payment

when you paid

Report

How many cars

How much was made

Average rate of stay