Software Requirements Specification

for

Smart Car Parking System

Version 0.1 candidate

Prepared by

Name	ID
MD SAKIB HOSSAIN	20-42640-1
MD. RIAZ KHAN	19-40183-1
MAHEZABIN SHEELA	20-42810-1
SADIA LUBNA PROVA	19-39949-1
MARZIA MEHZABEEN	19-39594-1

AIUB

2022-12-12

Table of Contents

Revision History

Name	Date	Reason For Changes	Version
Version1	10/08/22	Update	1.1
Version2	10/12/22	Update	1.2

1. Introduction

1.1 Purpose

The software described in this document is currently called Smart Car Parking System. This name might change later. In this software we want to establish a system that can allocate free parking space all over Dhaka and the customers parks their cars in that space. Our project will provide much-needed relief for this city and its people from the lack of parking spaces.
The version of this document is 0.1

1.2 Document Conventions

This template is fetched at https://web.cs.dal.ca/~hawkey/3130/srs_template-ieee.doc.
Defined terms are highlighted with **bolding.**
Requirements will come with priority to indicate in which order they will be implemented. Versions
might be released with only some of the total list of requirements implemented.

1.3 Intended Audience and Reading Suggestions

This document is written to demonstrate the functionality and appearance of the software for aerial survey professionals (users and testers of the system). Additionally, those designing surveys or using the results could have functional wishes. The SW's administrators will have access to unique
and uncommon features with less clear user interfaces.
The people who contributed to this document are listed below. This group will receive the draft for comments.

MD SAKIB HOSSAIN	20-42640-1
MD. RIAZ KHAN	19-40183-1
MAHEZABIN SHEELA	20-42810-1
SADIA LUBNA PROVA	19-39949-1
MARZIA MEHZABEEN	19-39594-1

1.4 Product Scope

We aim to provide a Parking solution. We will have two types of users one is Driver and another one is Renter. The parking spaces will be offered by the renters. The drivers have

to pay charges for their car parking for the allocated time. Parking charges will be estimated by our system and admins. Our system will use google Maps for showing available parking places. The payments will be done by Mobile banking. Ex: Bkash, Rocket, Nagad, and others.

1.5 References

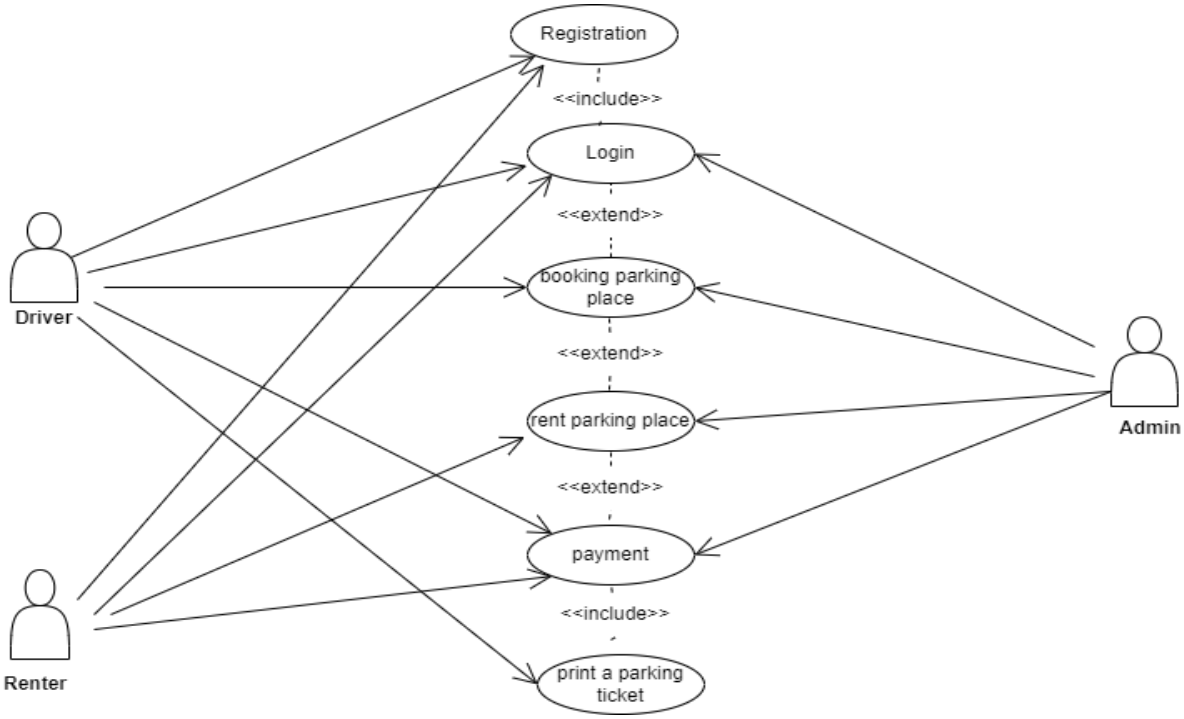
https://web.cs.dal.ca/~hawkey/3130/srs_template-ieee.doc

2. Overall Description

2.1 Product Perspective

Our project will provide much-needed relief for this city and its people from the lack of parking spaces. We will have two types of users one is Driver and another one is Renter. The smart parking system using parking space recognition, mobile Internet, and other technologies applied to the parking lot. The drivers who want to park their car for 30min, 1hr, and their allocated time they want. The parking spaces will be offered by the renters. The drivers have to pay charges for their car parking for the allocated time. Drivers will find parking places through our system and our system will show available parking places near the driver's location. Renters post their parking places with pictures and specific locations of their parking spaces. Parking charges will be estimated by our system and admins. Our system will use goggle Maps for showing available parking places.

Diagram:



2.2 Product Functions

The following are the product functions of the smart parking system:
1. The login box should on the intranet.
2. The password field should be secured.
3. By clicking on the dropdown box of the options the user should be able to view reports and available slots.
4. User should be able to change the passwords

2.3 User Classes and Characteristics

There are 3 classes in developing system: Driver, Renter and admin. Our 2 main class- Driver and Renter, they give full information about his/her such as name, surname, Id, address, phone. So different users will never be mixed in tables and they can be initialized by all these characteristics. Another class is admin. They controlled the full management system.
1. Users have to verify their faces. Users have to give access of location.
2. Users can find available parking places near their location.
3. Users can post their parking places for rent.
4. Users can payment their charges through mobile banking.
5. Users can Reserve parking places for park their car.
6. Users have to be verified by our system.
7. Users can withdraw their money after confirmation by our system or admin.

2.4 Operating Environment

There are many OS environment where our system is compatible with differ version of OS as describe in hardware and software requirements , so it does not need to rely on any other Operating System . Though, one dependency is hardware, and the performance of will depend on the degree of hardware performance. The PMS will provide the best features of both Android and IOS combined, so it will have to provide complete support to old and new applications designed for both Android OS and IOS. To achieve this PMS will have to provide app development environment (and support) for both Java and Swift / C ++ applications.
This software will run on android and iOS platforms.

2.5 Design and Implementation Constraints

Server limit is the number of clients can get to or can be online immediately. More is the quantity of clients more will be the organization traffic and henceforth the server arrives in a down state. Individual firewall and refreshing are an intense undertaking, it ought to be with the end goal that it ought not hinder the organization traffic, making the framework more slow. As an operating system, the major constraint for PMS will be hardware and its performance. There are several trivial and

non - trivial constraints, which are from both development perspective and context perspective. Some of them are listed as follows:

- Memory of the device.
- Cope with boot up procedures of different manufacturers
- Precise exception handling for both hardware and software
- Prevent deadlock during applications execution (Prioritize core tasks).

2.6 User Documentation

User document is the document where we describe how user interact with the system. In below those are our user story.

Sl	As A/An	I want to	So that	Acceptance criteria
1.	Driver	Create an account	I can use this service	Show create account form
2.	Driver	See the available parking nearby	I can park my car	Show available location on map
3.	Driver	Send text to renter	I can assure the place	Chatting system
4.	Driver	Send text to customer care	I can report about things	Chatting system with customer care
5.	Driver	Book a parking place for later	I can reserve a space	Show available reserve parking
6.	Renter	Create an account	I can use this service	Show create account form
7.	Renter	I want see the active users	I make decision about my space	Showing active user
8.	Renter	Send text to customer care	I can report about things	Chatting system with customer care
9.	Renter	Send text to the drivers	I can confirm their condition	Chatting system
10.	Renter	Post multiple slot at a time	I can use my all available spaces	Showing post option
11.	Driver	Book the parking place	I can pre-book a place	Notify driver with booking info.
12.	Renter	Give rent one or more area with slot	I can earn money	Notify when approve by admin
13.	Customer	Open an account	I can use the app to park my car	Accept all the terms and condition,
14.	Customer	I want to report a fake account	The admin can check that	Notify when the report is reviewed by admin
15.	Customer	Give review about the renter	Other can see this before they use	Review section with comment box.
16.	Renter	Give a rating about this app	People know that the app is good or bad while download	Review submitted message

Sl	As A/An	I want to	So that	Acceptance criteria
17.	Online payment	Give my service to the people	They can use for online payment.	Online payment method option.
18.	Manager	Check active users	I can manage the system	Show active users
19.	Manager	Check the reviews	I can notify the admins	Sent to the admin
20.	Admin	Check the reviews	I can disable fake account	Show customer id with message
21.	Driver	Cancel my pre-booking.	I can get back my money.	Show notification of successfully cancel
22.	Renter	Cancel or Delete my parking slot area.	I can get back my parking slot.	Show notification of successfully cancel
23.	Driver	Share this app with others	Other people can use this	Show available sharing option
24.	Google map	Show the precise location live	People can see whats nearby	Show location precisely
25.	Customer	Delete my account	I can assure my safety	Show confirmation when deleted
26.	Renter	Close my account	I can assure my safety	Show confirmation when deleted
27.	Customer	Pay my payment with bkash	I can easily use this app.	Show available payment option with Bkash.
28.	Driver	Add money from bkash	I can payment automatically	Confirmation notification with transaction id
29.	Renter	Withdraw my money with Nagad	I can use my nagad account for payment.	Confirmation notification with transaction id
30.	Renter	Withdraw my money with Bkash	I can use my nagad account for payment.	Confirmation notification with transaction id
31.	Admin	I want to see customers profile info	I can verify customers	Show info of customer
32.	Manager	I want to see customers profile info	I can verify customers	Show info of customer
33.	Manager	Monitor maintain and configure the app	The app can run smoothly	Show dashboard with access control
34.	Admin	View and manage the user permission	The app can protect the customers data	In app notification
35.	Customer	Contact with admin	I can verify my account.	Send notification after verification

Sl	As A/An	I want to	So that	Acceptance criteria
36.	Driver	Contact with admin	I can report any fake places.	Send notification after verification
37.	Renter	Post my available parking place	It show to the nearby drivers	Notify with available parking list.
38.	Renter	Edit my post	I can edit my mistakes.	Notify with update message.
39.	Renter	Delete My post	I can remove my parking place	Notify activity with post deleted
40.	Renter	Select timing slot for my parking place	I can manage my parking space	Show timing slot with date.
41.	Driver	Book parking slot	I can park my car.	Notify Customer with location.
42.	Driver	Park the car with allocated time.	I can park the car.	Show notification.
43.	Driver	Choose location	I can choose my parking place.	Give location access.
44.	Driver	Contact the admin	Inform my issue.	Send message by chat system.
45.	Admin	Contact with driver	I can give their desire places.	Send message to the driver.
46.	Admin	Contact with driver	I can solve their issue.	Send message to the driver.
47.	Admin	Divide the timing slot	I can maintain the parking timing	Send notification of empty slot.
48.	Renter	Call customer care	I can tell my issues	Calling system to the customer care
49.	Renter	Call the driver	I can assure their want	Calling system to driver
50.	Admin	Remove posts	Fake post will not be available	Show notification with confirmation of deleted post

2.7 Assumptions and Dependencies

We identified the following assumptions about the car parking system-to-be:

1. The license-plate recognition system works correctly 100 % of time, with no errors because of dirty or scratched license plates. If the registration number cannot be recognized, the software-to-be will assume that this vehicle does not belong to a registered customer.

2. If the license-plate recognition system cannot recognize the vehicle registration number and the customer do not provide a valid confirmation number or membership number, the customer will be asked (display message) to back away from the vehicle elevator (business policies (P1) and (P2)). We assume that the customer will always obey and depart. If we suspect otherwise, we may manage this risk by having the system to notify the garage security personnel. We also need an additional sensor to sense a vehicle presence in the elevator and assumptions about its correct operation. (Perhaps the license-plate recognition camera can serve this purpose)

3. The spot-occupancy recognition system works correctly 100 % of time, with no errors because of sensor malfunctioning or incorrect sensing. If the sensor positively detects occupancy, we assume that this result is only due to a vehicle, and no other object occupying the spot. For example, if the sensor is based on visible light, we assume that “dark” is sensed is only because a vehicle is present above the sensor, rather than because of lighting outage or some other condition; similarly, “light” is sensed is only because of a vacant spot and not because of another accidental light source.

4. The vehicle elevator will lift the vehicle to the appropriate deck, and will never stop on a wrong deck by mistake.

5. In the case of the business policy (P10), we assume that the customer will always accept the offered parking spot and will never wish to opt out and leave the parking garage without parking his or her vehicle.

6. We assume that the customer will always park at their assigned spot, and will never park at an arbitrary vacant spot. We assume that the garage currently does not have installed sensors for continuous tracking of customers from the vehicle elevator to their assigned spot. This is a strong assumption (and we know that people are unpredictable!), but the accuracy of the parking reservations table depends on this assumption. If a customer parks on a wrong spot (e.g., spot B), and the system thinks that the customer is parked correctly on spot A, then the system will direct future customers to an already

occupied spot B (or accept reservations for B), and meanwhile the system will consider spot A as occupied, while it is actually available.

7. If the license-plate reader successfully recognizes the vehicle registration number, the system assumes that the driver is a registered customer. The system-to-be will not consider separately scenarios where the driver is a non-registered customer who borrowed the vehicle from a friend who is a registered customer, or if the vehicle was stolen from a registered customer. The customer to be billed for the parking garage use will be decided as per the business policy.

8. We assume that the customer has access to email and a mobile phone with SMS texting capabilities. We do not assume that the customer will regularly check his or her email or will always be able to receive instantaneously SMS messages. For example, the customer may be in a meeting with the phone turned off. Also, we do not assume that the customer has access to a computer or smartphone while driving.

The above description of the system is only preliminary, provided as a reference example. The student developers' team may add, drop, or modify any of the statements as deemed appropriate. Also, the team should consider how will the system functioning be affected by scenarios in which the above assumptions are not satisfied.

3. External Interface Requirements

External interface requirements are a subset of functional requirements. They are essential for integrated devices. They also define how the solution will interact with other aspects. The External Requirement for Online Car Parking System are-

3.1 User Interfaces

In the Smart Car parking system app, the user will have access to many switches for a variety of purposes through the app's user interface, which features a digital screen layout with all of the programs or options presented. For instance, the user must select the "Book Now" option to reserve a parking space, or the "Ask Us" option to submit a question. Slide up and down menus will be present on the app. The users will be exposed to digital visuals and pop culture videos.

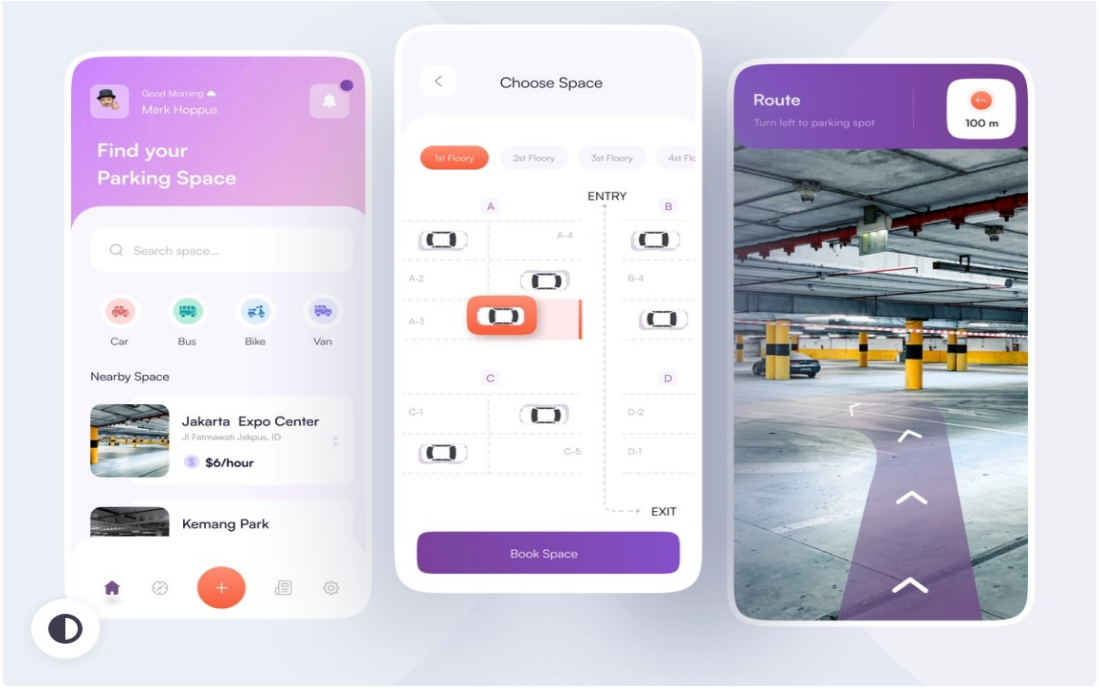
- Drivers and clients can create accounts on the system.
- The system allows the system administrator to book parking spaces, produce receipts, and manage client and parking lot information, such as deleting, updating, adding viewing the client information and viewing different type of parking lot status.

- The technology enables drivers to discover and reserve a parking space online by using a web portal.
- The system allows the client and the system administrator to see whether parking is available or reserved.
- Clients can access their account information (booking status) and print from the system.

GUI BASED UI:

The GUI, or graphical user interface, is a sort of user interface that allows users to interact with electronic devices using graphical symbols and auditory indicators like main notation rather than text-based UIs, written command labels, or text navigation.

With The help pf GUI, this app will offer various features for this online auto parking project, such as open id, the types of vehicles the user wants, nearby spaces, notification, helpline, user may manually pick their location, choose space, and locate routes in the apps map. This allows them to easily go to their desired parking location. Users can also pay using a variety of methods.



3.2 Hardware Interfaces

- In Smart Car parking system app, will have certain requirements for the hardware interface, such as mobile device models and app versions, etc. Also, servers and mainframes This app will work on both Apple and Android devices. However, in order to utilize the app smoothly, both the user and the mobile device need be updated. The app can be used with

both mobile data and WIFI. The mobile device needs memory space in order for the software and hardware to work together. As enormous volumes of data searches must be handled, a fast processor is essential.

- Also, because server runs multiple processes concurrently, 8 GB of memory is required for quicker performance, while the memory in the computer should be sufficiently fast to run the tasks required.
- A server's hard disk must have 200 GB of storage capacity in order to store massive amounts of data, whereas a consumer computer's disk space should be adequate.
- Since the program is mostly internet-based, a good network connection is essential for speedier communication and information retrieval.

Hardware	**Minimum Requirement**
Processor speed	1.6GHZ
Memory of user PC	512MB RAM
Disk Space of user PC	5GB
Memory of server Pc	8GB
Bandwidth(network connection)	15Mbps
Disk space of server	200GB

3.3 Software Interfaces

This app's software interface will make use of Google Maps to locate a parking space, GPS to get there, mobile banking to process payments, trackers to monitor cars to determine whether or not users were successful in using the location, a software management system to oversee the app, among other things. The app will help new users by offering training, and it will also include user guidelines. Additionally, users can contact the help line to learn more about the software. The application is compatible with the following operating systems: Windows 7, Windows 8, Windows 8.1, and Linux. These operating systems were chosen since they are inexpensive and widely available. MySQL, which is relatively inexpensive, simple to use, and scalable, was employed in the building of the databases. Browsers include opera, Google Chrome, Microsoft Internet Explorer, and Mozilla Firefox.

Software	Minimum requirement
Operating System for computer	Window 7 windows 8.1,Linux,Mac Os ,Windows 8,windows xp
Database Management System	MySql
Browser	Opera,Google Chrome,IE,Mozilla Firefox

3.4 Communications Interfaces

The app's communication interface will offer a variety of communication channels, including hotline calls, emails, messages, and a reporting system. To get assistance, a user can dial the hotline number. Users can fill out a form in the report section to register a complaint about any issue. Users can choose the "Ask us" option for more assistance in formatting the app chat. The HTTP and TCP protocols will be used by the app because they meet all the requirements. The Parking Space app will rigorously uphold data privacy, saving all financial and personal information without ever sharing it with any third parties.

4. System Features

In system feature we include the feature that we have in our system.

4.1 System Feature 1 (Request Parking)

4.1.1 Description and Priority

In request parking we can request for parking. This have a priority of 1.

4.1.2 Stimulus/Response Sequences

List the sequences of user actions and system responses that stimulate the behavior defined for this feature. These will correspond to the dialog elements associated with use cases.

4.1.3 Functional Requirements

Itemize the detailed functional requirements associated with this feature. These are the software capabilities that must be present in order for the user to carry out the services provided by the feature, or to execute the use case. Include how the product should respond to anticipated error conditions or invalid inputs. Requirements should

be concise, complete, unambiguous, verifiable, and necessary. Use "TBD" as a placeholder to indicate when necessary information is not yet available.

Each requirement should be uniquely identified with a sequence number or a meaningful tag of some kind.

REQ-1:
REQ-2:

4.2 System Feature 1 (Post parking)

4.2.1 Description and Priority

In post parking we can post parking place. This have a priority of 1.

4.2.2 Stimulus/Response Sequences

List the sequences of user actions and system responses that stimulate the behavior defined for this feature. These will correspond to the dialog elements associated with use cases.

4.2.3 Functional Requirements

Itemize the detailed functional requirements associated with this feature. These are the software capabilities that must be present in order for the user to carry out the services provided by the feature, or to execute the use case. Include how the product should respond to anticipated error conditions or invalid inputs. Requirements should be concise, complete, unambiguous, verifiable, and necessary. Use "TBD" as a placeholder to indicate when necessary information is not yet available.

Each requirement should be uniquely identified with a sequence number or a meaningful tag of some kind.

REQ-1:
REQ-2:

4.3 System Feature 1 (Payment)

4.1.1 Description and Priority

By this feature we can do payment and withdraw income. This have a priority of 1.

4.1.2 Stimulus/Response Sequences

List the sequences of user actions and system responses that stimulate the behavior defined for this feature. These will correspond to the dialog elements associated with use cases.

4.1.3 Functional Requirements

Itemize the detailed functional requirements associated with this feature. These are the software capabilities that must be present in order for the user to carry out the services provided by the feature, or to execute the use case. Include how the product should respond to anticipated error conditions or invalid inputs. Requirements should be concise, complete, unambiguous, verifiable, and necessary. Use "TBD" as a placeholder to indicate when necessary information is not yet available.

Each requirement should be uniquely identified with a sequence number or a meaningful tag of some kind.

REQ-1:
REQ-2:

5. Other Nonfunctional Requirements

5.1 Performance Requirements

It is quantifiable and define at minimum, the context and expected throughput, response time, max error rate, and sustained amount of time.

Performance Requirements measure

Response Time:
Response time refers to the amount of time it takes for a server to respond to a client's request. Measured in milliseconds, the timer starts from the moment a client sends out a request and stops when the server sends back its first response.

Workload:
A workload is the amount of computing resources and time it takes to complete a task or generate an outcome. Any application or program running on a computer can be considered a workload.

Scalability:
Scalability is the measure of a system's ability to increase or decrease in performance and cost in response to changes in application and system processing demands.
an application program would be scalable if it could be moved from a smaller to a larger operating system and take full advantage of the larger operating system in terms of performance (user response time and so forth) and the larger number of users that could be handled.

Platform:
A platform is any hardware or software used to host an application or service. An application platform, for example, consists of hardware, an OS and coordinating programs that use the instruction set for a particular processor or microprocessor.
An example of a computing platform is a modern laptop running Windows as an operating system. Another example would be an Apple computer running the Mac OS X operating system.

5.2 Safety Requirements

Timing Relationship for Real Time System:
Functional requirements are unique and are the heart of what your system is expected to do and capabilities it needs to have. Performance requirements tell us how the function will be done – how well, how often, how much, how many, etc.

Safety Requirements

Safety requirements specifications (SRS) are specifications that describe every required safety function that must be performed by a safety instrumented system (SIS). SRSs specify both what safety functions must be performed by a system and how well those functions must be performed.
for example, a software-based system along with management procedures, checklists, and validation procedures for using it.

Safety policies

- Incident reporting policy
- Drug and alcohol policy
- Safe driving policies
- Personal protective equipment (PPE) policy
- Lockout policy and procedures
- Transitional duty policy

Safety Certifications

Degree-Based Safety Certifications

- Associate Safety Professional (ASP)
- Graduate Safety Practitioner (GSP)
- Certified Safety Professional (CSP)
- Safety Management Specialist (SMS)
- Occupational Hygiene and Safety Technician (OHST)
- Safety Trained Supervisor (STS)
- Certified Safety Manager (CSM)

CSP - Certified Safety Professional

The Certified Safety Professional (CSP) Certification is the "Gold Standard" for the Safety Professional. Covering a broad range of EHS concepts, this certification carries some prestige within the world of occupational safety.

Security Requirements

The security requirements must cover areas such as:

- Authentication and password management.
- Authorization and role management.
- Audit logging and analysis.
- Network and data security.
- Code integrity and validation testing.
- Cryptography and key management.
- Data validation and sanitization.

User Identity Authentication

Identification is the act of identifying a particular user, often through a username. Authentication is the proof of this user's identity, which is commonly managed by entering a password. Only after a user has been properly identified and authenticated can they then be authorized access to systems or privileges.

Security Policies

- Acceptable use policy (AUP)
- Data breach response policy
- Disaster recovery plan
- Business continuity plan
- Remote access policy
- Access control policy

Security Certifications

- CompTIA Security+ ...
- Certified Ethical Hacker (CEH) ...
- GIAC Security Essentials Certification (GSEC) ...
- Systems Security Certified Practitioner (SSCP) ...
- CompTIA Advanced Security Practitioner (CASP+) ...
- GIAC Certified Incident Handler (GCIH) ...
- Offensive Security Certified Professional (OSCP)

5.3 Security Requirements

<Specify any requirements regarding security or privacy issues surrounding use of the product or protection of the data used or created by the product. Define any user identity authentication requirements. Refer to any external policies or regulations containing security issues that affect the product. Define any security or privacy certifications that must be satisfied.>

5.4 Software Quality Attributes

The following are the attributes of the product PMS:

- *It should be equipped with current and archive database.*
- *All records can easily be updated.*
- *It should have its personal firewall.*

5.5 Business Rules

There are several rules set before development system is to ensure that users and developers can access the only part which they are allowed to. These roles are broadly categorized as follows:

•Project managers will be able to lead the development team apart from accessing the code. Project Managers will further be divided so that only specific PMs can access their field within the development stage.

•Developers can access both backend (code) and frontend of the operating system. The developers are further ranked so that only specific expertise perform their duty and do not take control of all the development.
•QA team can only review bug reports and propose proper fixes for the bug. For this purpose, they may be given access to some part of the system's code.

•End - users can only use the front - end of the system and use its features. They are not allowed to, and will not be, given access of the code of the system.

6. 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.>