

PULSE PARKING AUTOMATION

**UCS 503
Software Engineering**

Submitted by

**101603074 – Bhavay Pahuja
101603078 – Chirag Mahawar
101603199 – Navjot Singh Sandhu
101610022 – Deepak Lather**

B.E Third Year, Computer Engineering

Submitted to

Ms. Deepali Bhagat



**Thapar Institute of Engineering and Technology
Nov 2018**

UCS 503

Software Engineering, BE 3rd Year

Contents

Page No

1. Introduction

- 1.1 Scope Definition
- 1.2 Block Diagram of the Project
- 1.3 Ishikawa diagram of problem formulated

2. Analysis Phase

- 2.1 Use Case Diagrams based on Use Case Scenarios
- 2.2 Activity diagram to depict various procedural and algorithmic flows
- 2.3 Data flow diagram (DFD: Level 0,1,2) [It may become the part of design phase]
- 2.4 Software Requirement Specifications (SRS) in IEEE format

3. Design Phase

- 3.1 Class diagram *showcasing Logical structure*
- 3.2 Collaboration Diagram
- 3.3 Sequence Diagrams
- 3.4 State Chart Diagrams

4. Testing Phase

- 4.1 Test Plan
- 4.2 Test Cases for various test strategies (Scenario, Boundary value analysis and Equivalence class testing)
- 4.3 Test Report Generation

5. Deployment Phase

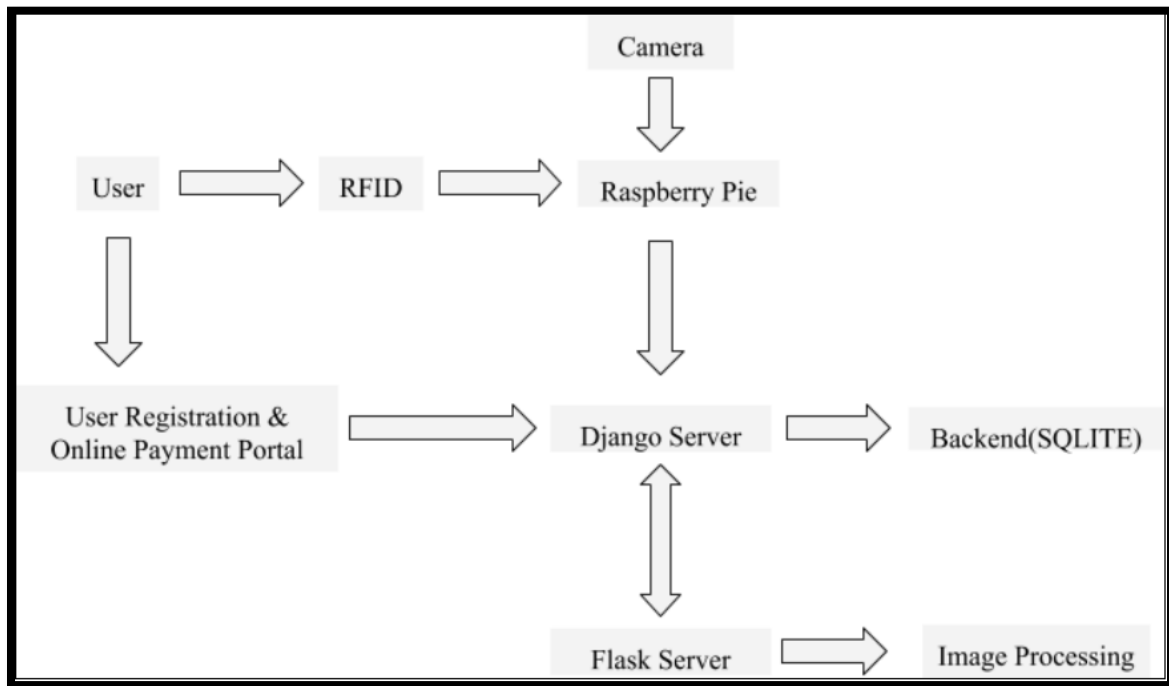
- 5.1 Component Diagram
- 5.2 Deployment Diagram

1. INTRODUCTION

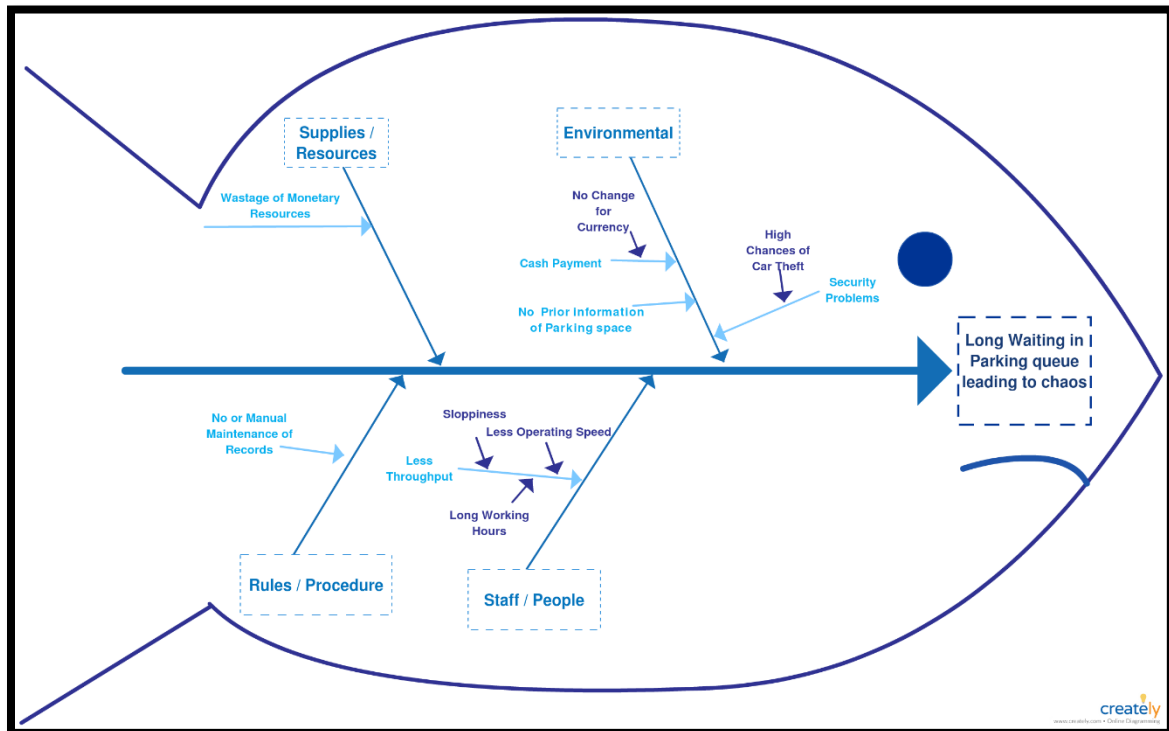
1.1 SCOPE DEFINITION

Parking a car in many cities is a frustrating adventure due to the scarcity of public parking and the high cost of garages. Although public parking (e.g. meters, residential parking and illegal parking spots) is inexpensive, it is inconvenient because the supply of parking spots is too low. Finding a parking spot typically requires time, luck, spare change, and/or the risk of an expensive parking ticket or towing. Parking garages may be more convenient in that available parking spots may be easier to find, but the cost of such a parking spot is significantly higher. The ever-increasing use of automobiles and limited space for parking, particularly in urban areas, has led to serious problems in parking management. It is often expensive and logistically cumbersome for police or municipality officials to keep track of offenders who park their vehicles without paying for the parking space. Parking meters do not fully solve the problem because they require a large investment on the part of the municipality and comprehensive monitoring by enforcement personnel. Most parking systems in developing countries are manually operated. Our product aims to decrease manual intervention and decreasing the waiting time for customers which is limited by human processing power and use of conventional methods to record and verify the data, which often leads to long waiting time especially during rush hours. Therefore, to address these problems and to provide a much convenient, better accessible and cost effective parking solution, there is a long felt need to develop a system and method which will be able to put into use a wider range of existing parking spots which till now were restricted, in particularly managing parking spaces which will provide ability to owners of private parking spaces to rent their own parking spots to other vehicle drivers. Moreover, there is a long felt need to develop a system which will have improved means for informing drivers when an available parking space is located on their way.

1.2 BLOCK DIAGRAM OF PROJECT

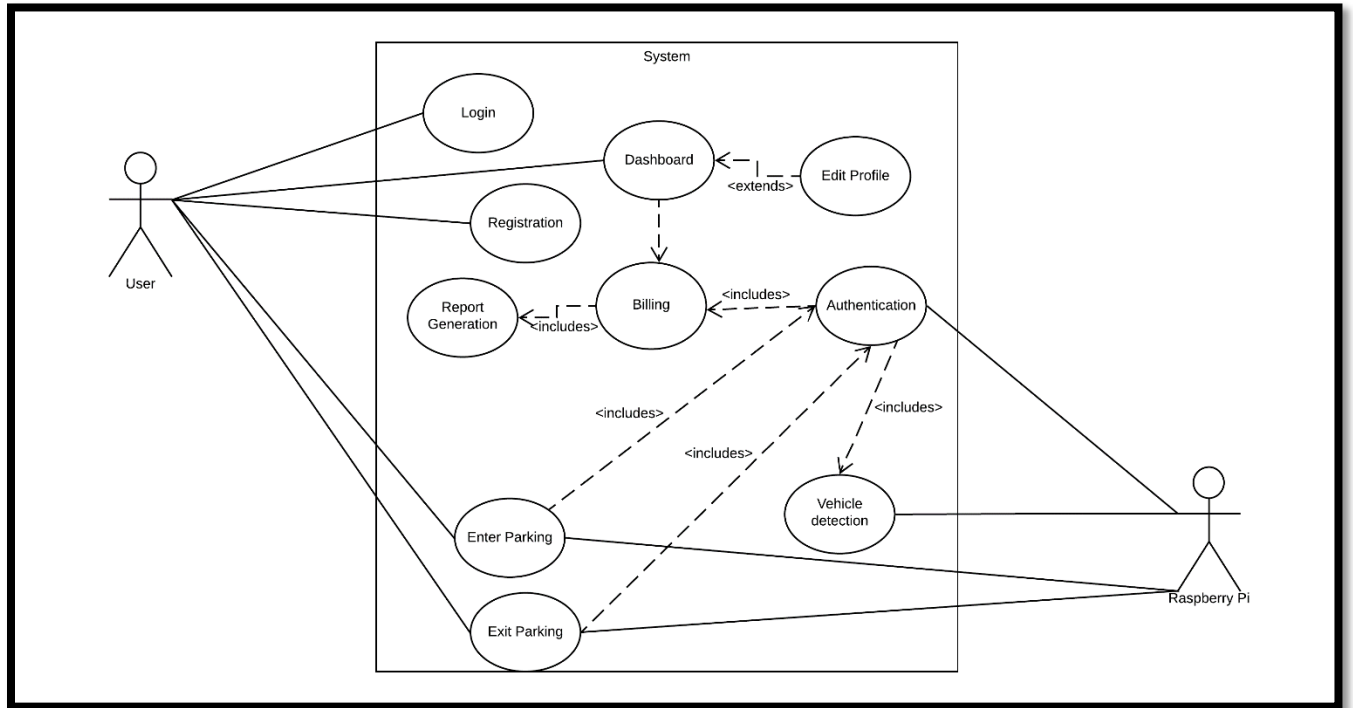


1.2 ISHIKAWA DIAGRAM OF PROBLEM FORMULATED

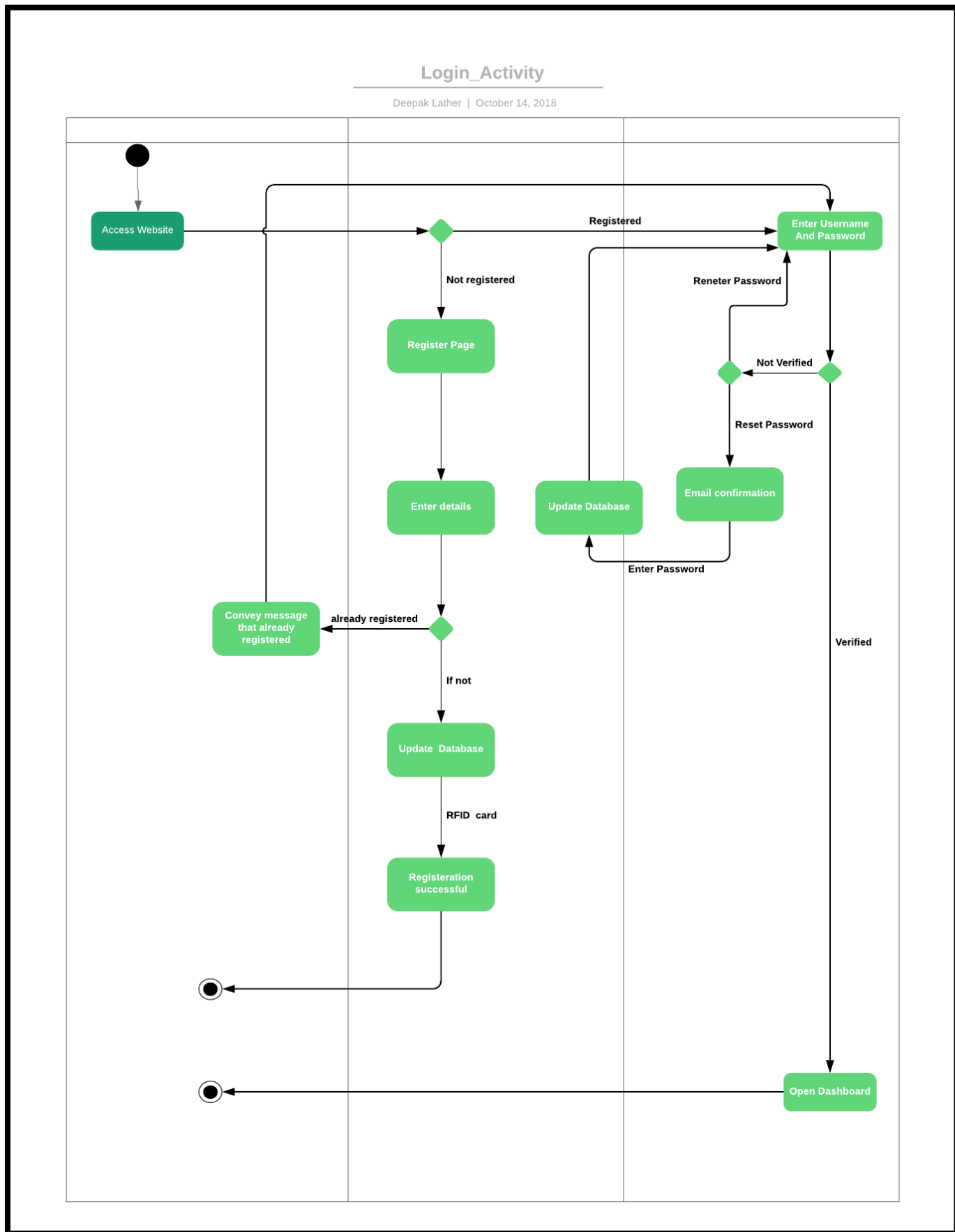


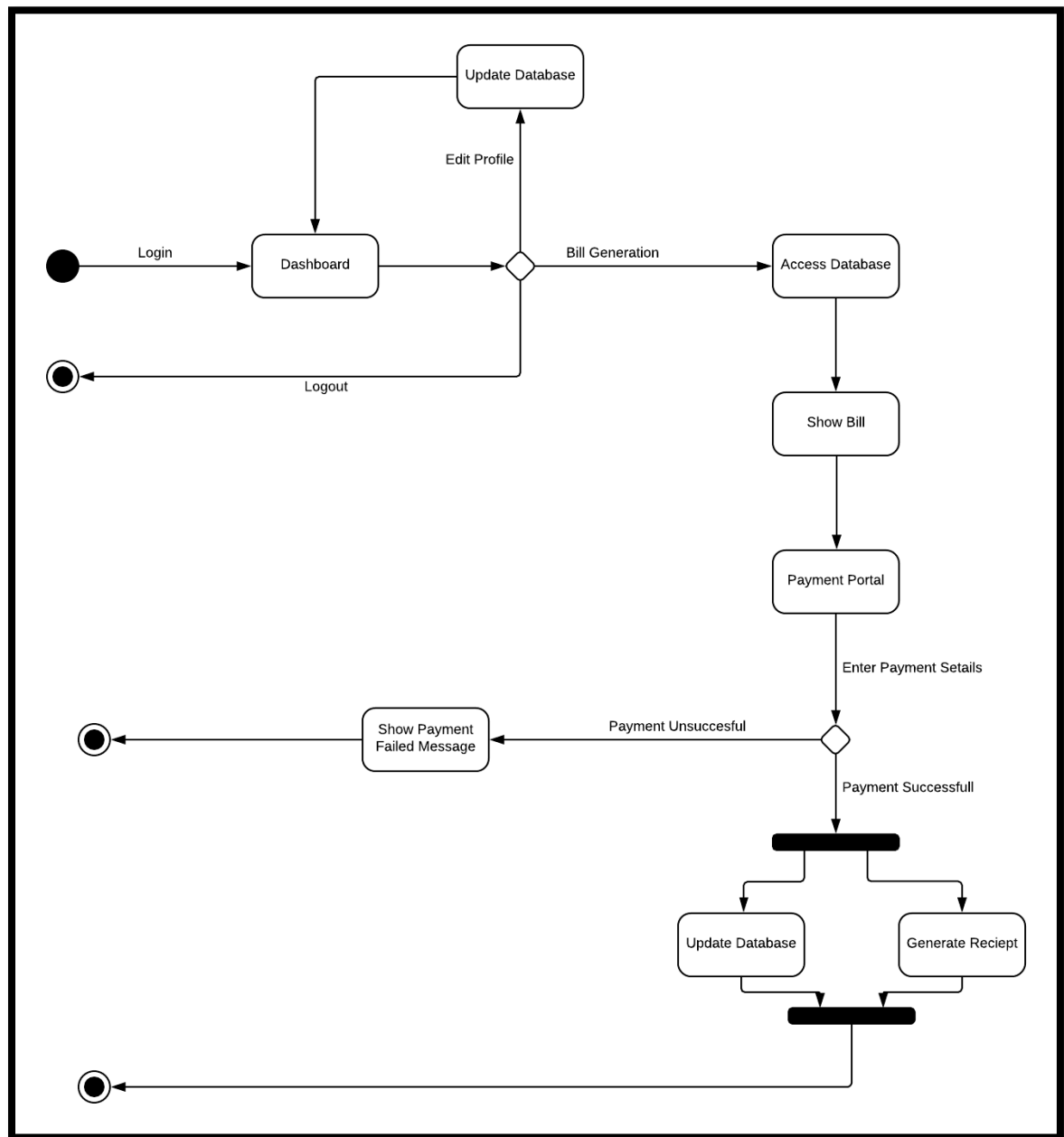
2. ANALYSIS PHASE

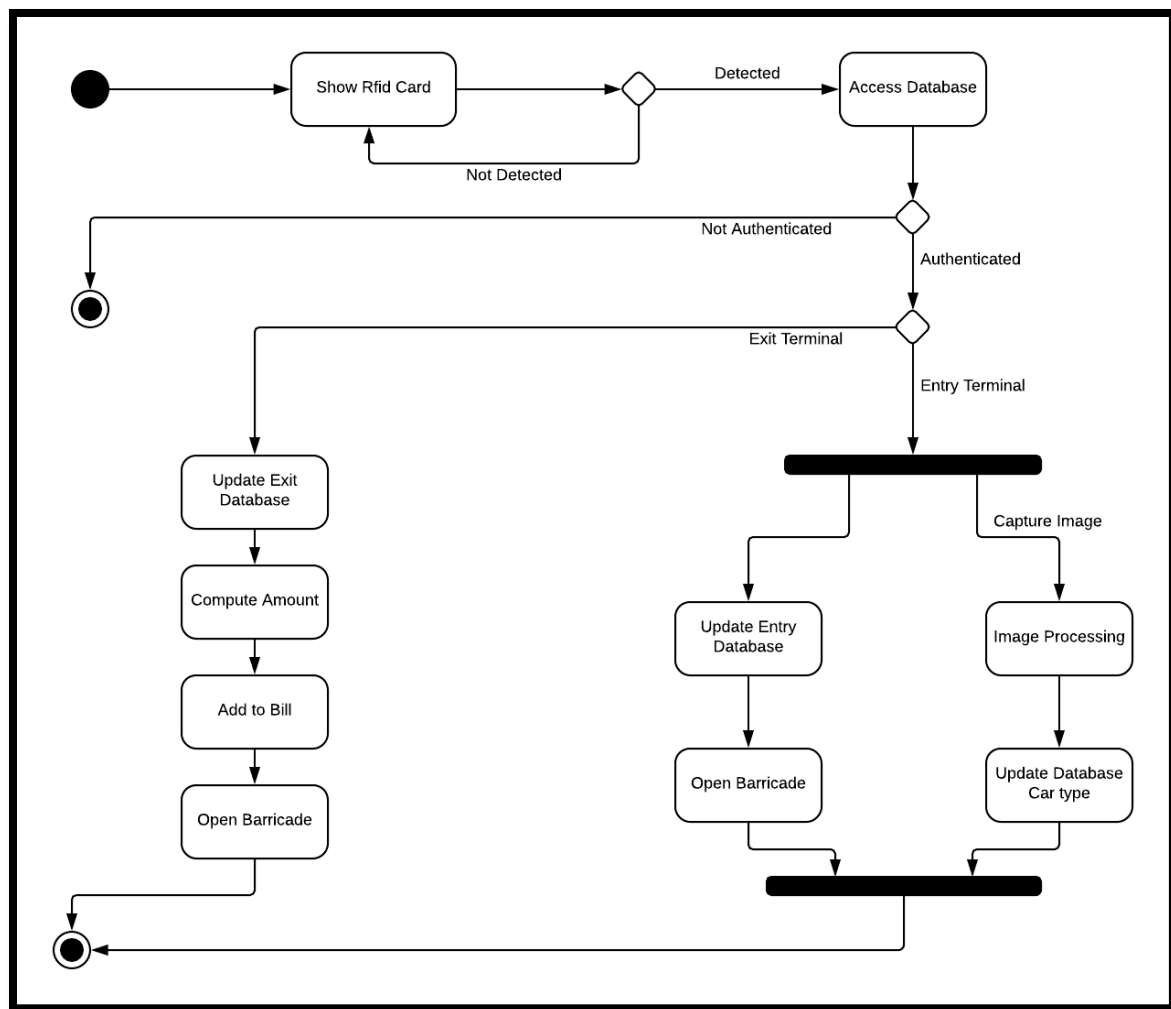
2.1 USE CASE DIAGRAM BASED ON USE CASE SCENARIOS



2.2 ACTIVITY DIAGRAMS TO DEPICT VARIOUS PROCEDURAL & ALGORITHMIC FLOWS

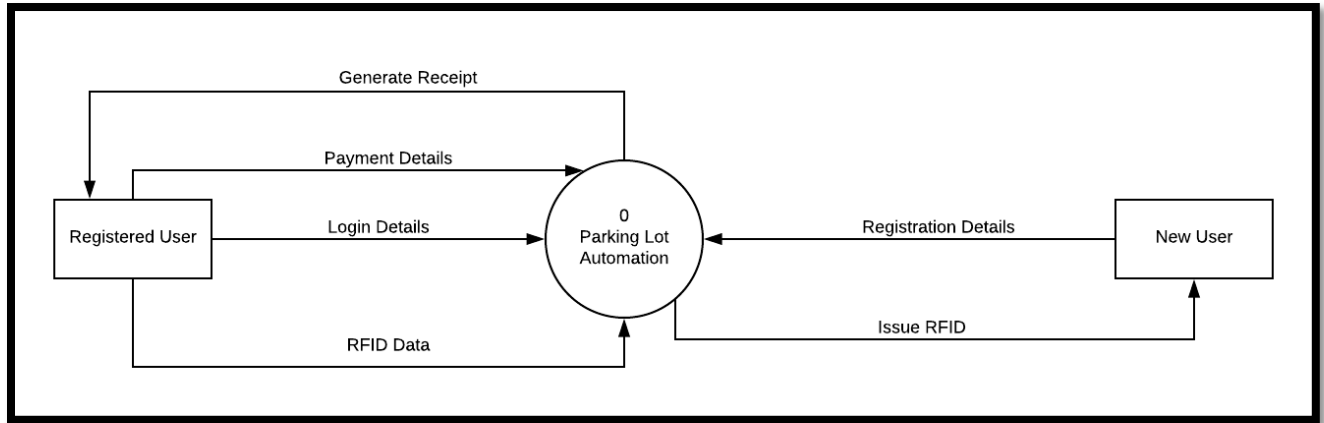




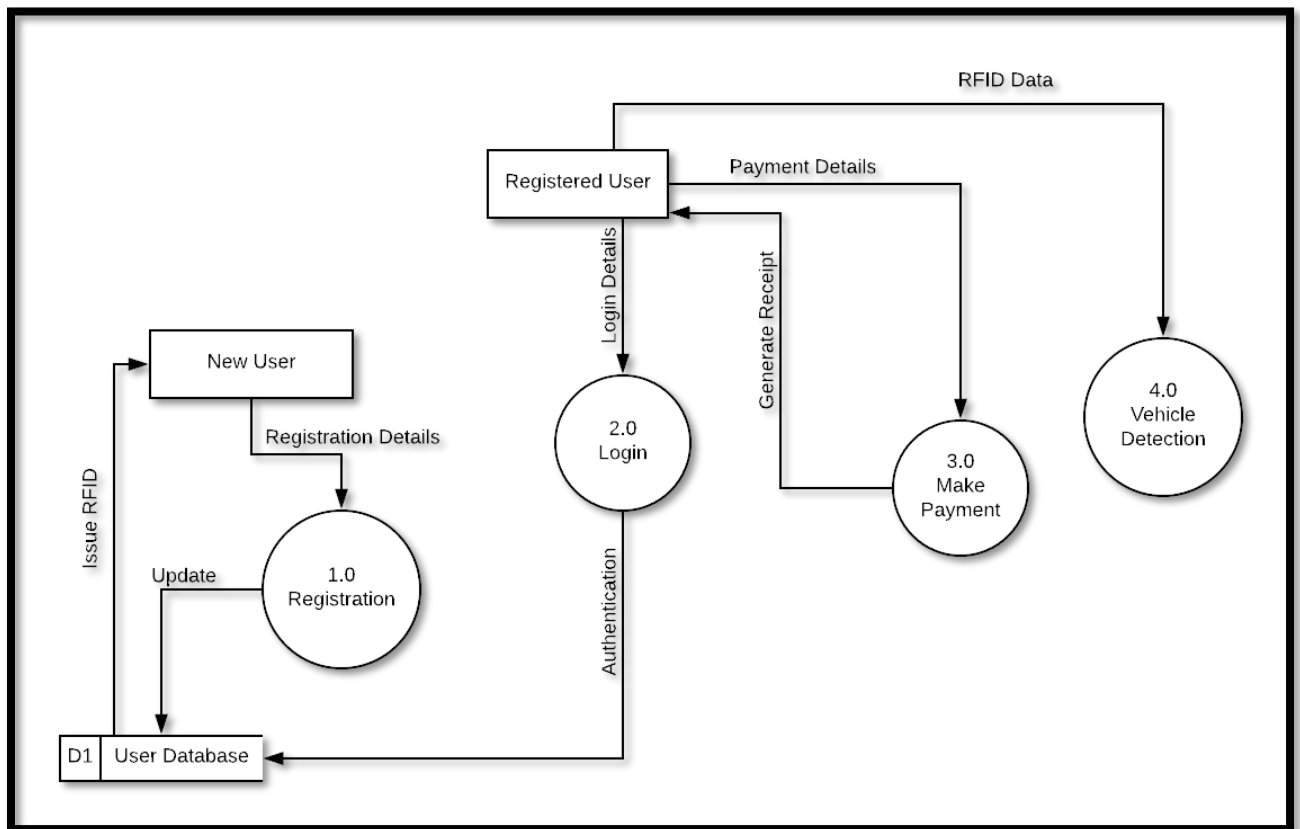


2.3 DATA FLOW DIAGRAM

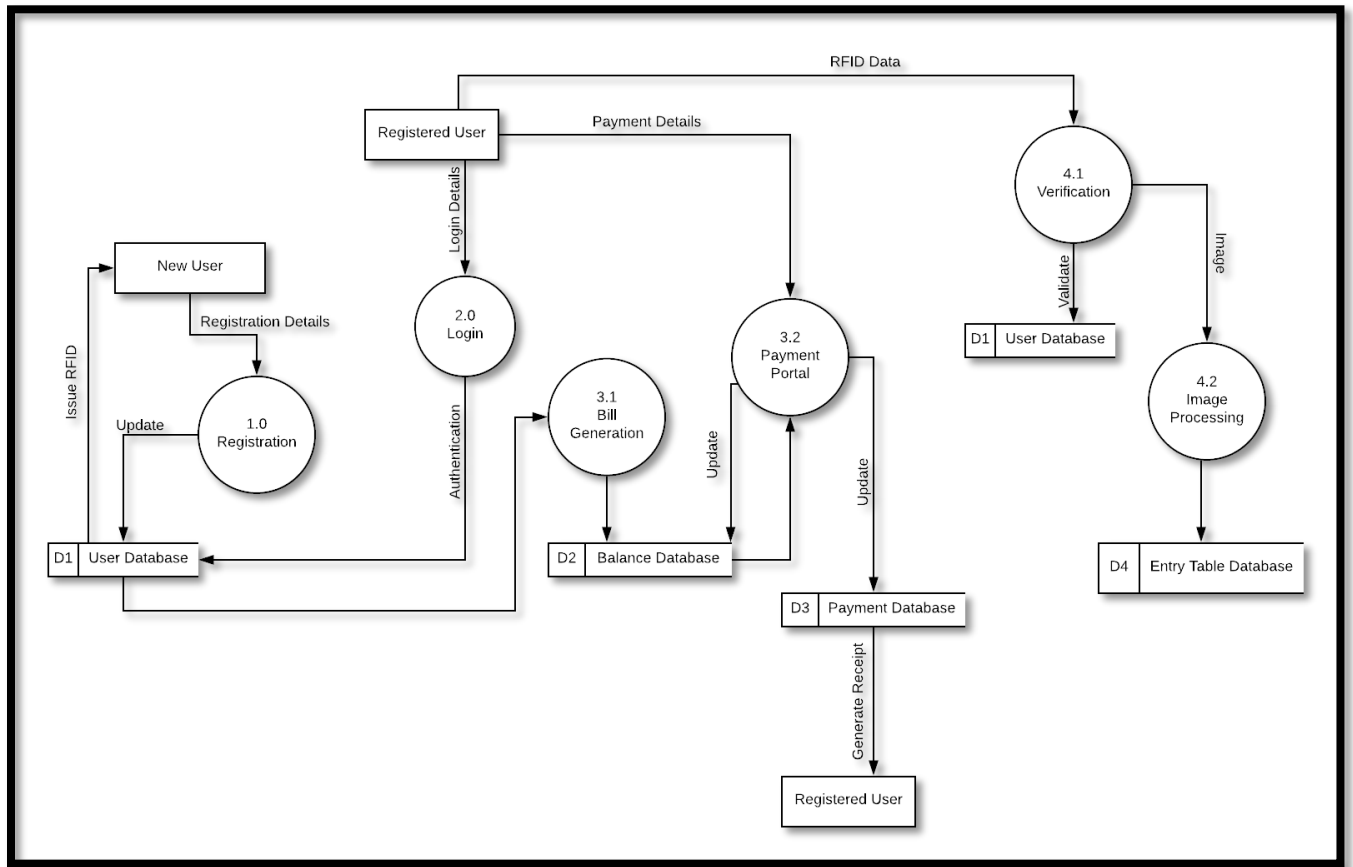
LEVEL 0



LEVEL1



LEVEL 2



2.4 SOFTWARE REQUIREMENT SPECIFICATION

Software Requirements Specification

For

Parking Automation using IoT

Version 1.0

Prepared by

Group Name: Pulse

Deepak Lather	101610022	dlather_be16@thapar.edu
Bhavay Pahuja	101603074	bpahuja_be16@thapar.edu
Navjot Singh Sandhu	101603199	nsandhu_be16@thapar.edu
Chirag Mahawar	101603078	cmahawar_be16@thapar.edu

Instructor: Dr Vinay Arora

Course: Software Engineering

Lab Section: COE-6

Teaching Assistant: Ms. Deepali Bhagat

Date: 16-09-2018

Contents

CONTENTS	2
REVISIONS	2
1 INTRODUCTION	3
1.1 DOCUMENT PURPOSE	3
1.2 PRODUCT SCOPE	3
1.3 INTENDED AUDIENCE AND DOCUMENT OVERVIEW	4
1.4 DOCUMENT CONVENTIONS	5
2 OVERALL DESCRIPTION	6
2.1 PRODUCT OVERVIEW	6
2.2 PRODUCT FUNCTIONALITY	8
2.3 DESIGN AND IMPLEMENTATION CONSTRAINTS	8
2.4 ASSUMPTIONS AND DEPENDENCIES	8
3 SPECIFIC REQUIREMENTS	9
3.1 EXTERNAL INTERFACE REQUIREMENTS	9
3.1.1 USER INTERFACES	9
3.1.2 HARDWARE INTERFACES	9
3.2 FUNCTIONAL REQUIREMENTS	11
3.3 USE CASE MODEL	13
4 OTHER NON-FUNCTIONAL REQUIREMENTS	14
4.1 PERFORMANCE REQUIREMENTS	14
4.2 SAFETY AND SECURITY REQUIREMENTS	14
4.3 SOFTWARE QUALITY ATTRIBUTES	15
5 REFERENCES	17
APPENDIX A – DATA DICTIONARY & ABBREVIATIONS	18
APPENDIX B - GROUP LOG	19

Revisions

Version	Primary Author(s)	Description of Version	Date Completed
Version 1.0	Deepak Lather Bhavay Pahuja Navjot Singh Sandhu Chirag Mahawar	Version 1.0 - Initial Release Parking Automation using IoT	16/09/18

1.Introduction

1. Document Purpose

The purpose of this SRS document is to provide a detailed overview of our software product, its parameters and goals. This document describes the project's target audience and its user interface, hardware and software requirements. It defines how our client, team and audience see the product and its functionality. Our product “Parking Automation using IoT” is computer-implemented system and method for managing motor vehicle , version 1.0. This product is a part of parking system.

2. Product Scope

Parking a car in many cities is a frustrating adventure due to the scarcity of public parking and the high cost of garages. Although public parking (e.g. meters, residential parking and illegal parking spots) is inexpensive, it is inconvenient because the supply of parking spots is too low. Finding a parking spot typically requires time, luck, spare change, and/or the risk of an expensive parking ticket or towing. Parking garages may be more convenient in that available parking spots may be easier to find, but the cost of such a parking spot is significantly higher. The ever-increasing use of automobiles and limited space for parking, particularly in urban areas, has led to serious problems in parking management. It is often expensive and logistically cumbersome for police or municipality officials to keep track of offenders who park their vehicles without paying for the parking space. Parking meters do not fully solve the problem because they require a large investment on the part of the municipality and comprehensive monitoring by enforcement personnel. Most parking systems in developing countries are manually operated. Our product aims to decrease manual intervention and decreasing the waiting time for customers which is limited by human processing power and use of conventional methods to record and verify the data, which often leads to long waiting time especially during rush hours. Therefore, to address these problems and to provide a much convenient, better accessible and cost effective parking solution, there is a long felt need to develop a system and method which will be able to put into use a wider range of existing parking spots which till now were restricted, in particularly managing parking spaces which will provide ability to owners of private parking spaces to rent their own parking spots to other vehicle drivers. Moreover, there is a long felt need to develop a system which will have improved means for informing drivers when an available parking space is located on their way.

3. Intended Audience and Document Overview

Intended Audience :

- **Clients:** The users of the system will get a clear idea of the software and hardware requirements to be engaged.
- **Developers:** Project developers have an advantage of quickly understanding the methodology enabled and personalizing the product.

The authors would suggest clients to go through the requirement section thoroughly before installing the software. This Document starts with giving an detailed overview of our project along with its architecture, its design and implementation constraints.

Moving on to the Functional Requirements Section which describes each interface which user interacts with in detail and other hardware requirements. The instructor should read the document in the current sequence and refer the Appendix for any unknown terms. Developers can utilize the documentation as a resource in developing the project to a new product.

4. Document Conventions

This document follows IEEE formatting requirements. Bold-faced text has been used to emphasize section and subsection headings. Highlighting is to point out words in the data dictionary in appendix section and italicized text is used for comments (if any) and figures with respective description is used for further clarification on data.

2. Overall Description

1. Product Overview

Product comprises of hardware & software units. The present innovation relates to the management of a parking lot and, more particularly, to setting up and using a parking lot managing system that relies on Internet Of Things. Product will run using RFID card & readers.

Main motive of the product is to reduce long waiting time for customers in queues in a parking lot that generally leads to chaos. Other objectives of the product is to provide guidance to customers to efficiently find available parking in a parking lot, to improve parking lot security by maintaining proper records of the vehicles entering & exiting the parking lot, to provide an interface for user registration, an online payment portal for recharge of RFID cards, to reduce manpower associated with parking lot management & to determine the type of object or vehicle that is currently parked in the parking space, to determine if it is a car, motorcycle, person, parking cart, or other object.

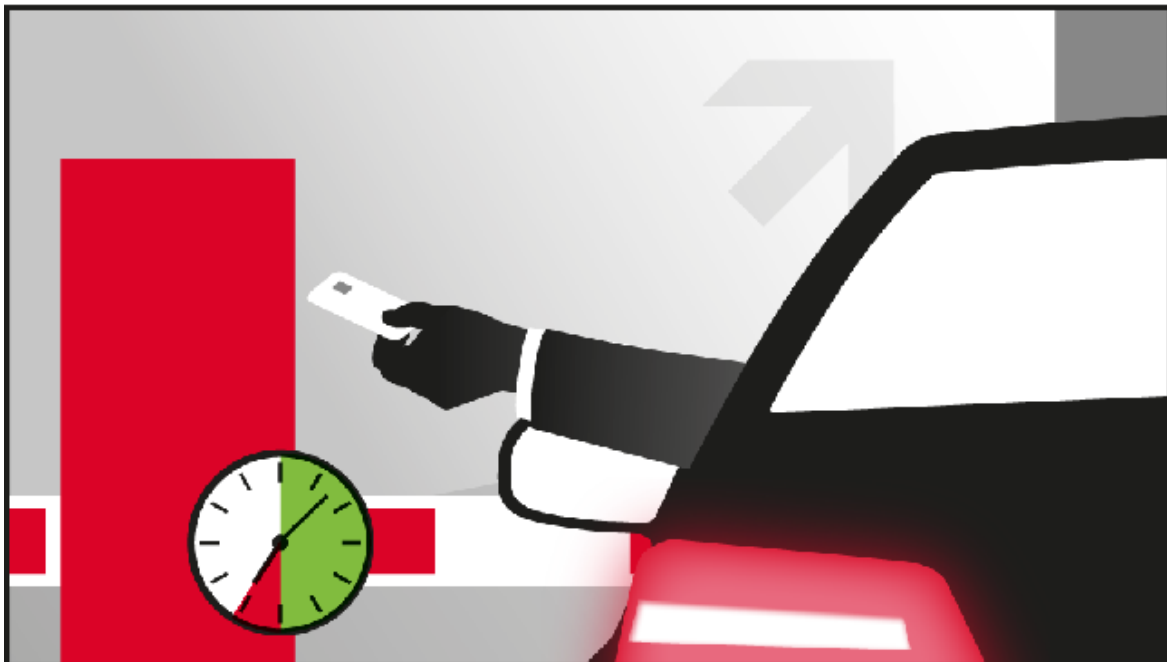


Fig 2.1.2 : Depiction of Entry Terminal

In a parking lot, at entry terminal, RFID reader will read the RFID card & send the data to Django Server using Raspberry pie & a camera will take an image & send it to sever . Django server maintains the data using SQLite.



Fig 2.1.3 : Image Processing for Vehicle Type Detection

Two microservices Django & Flask server interacts with each other in order to process an image clicked using the camera for vehicle type detection. In order to provide prior information about the available parking spaces, a display will be put up at entry terminal. Web interface will also contain online payment portal where user can recharge their cards.

2. Product Functionality

Product should be able to perform following operations -

- It must be able to authenticate the user using the RFID.
- Must provide platform to users to register for their RFID.
- Must maintain database for payment & registered user information.
- Must provide portal for online payment.
- Must maintain count of vacant parking space.
- Must identify vehicle type by capturing image at entry terminal

3. Design and Implementation Constraints

Following is the list of constraints that are imposed upon implementation of the product -

- Product requires synchronization with cloud data. That makes network connection a necessary element in the product.
- Since payment information is not stored in RFID, response time for loading the data from cloud may increase.
- Product requires hardware maintenance.
- Client must be familiar with online payment portal.
- Product uses Django server which in turn uses SQLite in backend, run-time queries will include interaction with database, thus increasing response time.

4. Assumptions and Dependencies

Following is the list of assumptions that are imposed upon implementation of the product

-

- Product on small scale assumes no duplication of RFID.
- Product on small scale assumes RFID delivery by any means possible.
- Product is prepared for allotting parking space to registered users only.

3. Specific Requirements

1. External Interface Requirements

1. User Interfaces

- Registration Interface
- Payment / Recharge
- Login
- User Dashboard
- RFID Reader Gate (For Registered Users)
- LED screen to display no. of free parking spaces available

2. Hardware Interfaces

Our project consists of Raspberry pi that collects data from the camera and the RFID reader. This data is sent to the website server that is powered by Django Framework.

Following are the hardware interfaces used in the product-

- Raspberry Pi 3
- RFID Tag
- RFID Reader
- Camera

RFID reader uses electromagnetic fields to automatically identify and track RFID card. The card contain electronically-stored information. Passive RFID cards is used which collect energy from a nearby RFID reader's interrogating radio waves. RFID is one method for Automatic Identification and Data Capture (AIDC). Signaling between the reader and the tag is done in several different incompatible ways, depending on the frequency band used by the tag. Tags operating on LF and HF bands are, in terms of radio wavelength, very close to the reader antenna because they are only a small percentage of a wavelength away. In this near field region, the tag is closely coupled electrically with the transmitter in the reader. The tag can modulate the field produced by the reader by changing the electrical loading the tag represents. By switching between lower and higher relative loads, the tag produces a change that the reader can detect. RFID reader is directly connected with raspberry pie which as act as a transporter of data from RFID reader to server.



Fig 3.1.2.1 : RFID Reader & Tag



Fig 3.1.2.2 : Raspberry Pi 3



Fig 3.1.2.3 : Supervision Camera

RFID Reader triggers the camera to capture the image through raspberry pie. Image captured at entry terminal is sent to server by pi for image processing.

2. Functional Requirements

Table 1: Function requirement for the authentication on user entry

Purpose	To authenticate the user on entry
Input	When RFID is scanned, its unique id is extracted
Processing	that unique id is queried in the database and the new entry is generated, its timestamp is stored along with its vehicle type which is queried from the vehicle detection api
Output	the barricade opens and lets user's vehicle enter the parking lot

Table 2: Function requirement for detecting vehicle type

Purpose	to detect the vehicle type
Input	image which is captured when the RFID is scanned by the RFID reader
Processing	this image is sent to the server where tensorflow framework is used to classify the image
Output	vehicle type

Table 3: Function requirement for the exit of the user

Purpose	to allow the user to exit
Input	When RFID is scanned, its unique id is extracted
Processing	that unique id is queried in the database and its exit timestamp is stored
Output	the barricade open and let the user's vehicle exit the parking lot

Table 4: Function requirement for billing

Purpose	monthly billing
Input	the corresponding entries from the database is extracted which includes the durations of different instances in the whole month that the user has used the parking lot
Processing	the corresponding amount is calculated based on the durations of the different instances that the user has parked the his/her vehicle in the parking lot and the type of vehicle
Output	bill is generated

Table 5: Function requirement for payment

Purpose	make payments
Input	bill generated in the billing function
Processing	the bill is sent to the external payment server
Output	payment is validated and the receipt is generated

3. Use Case Model

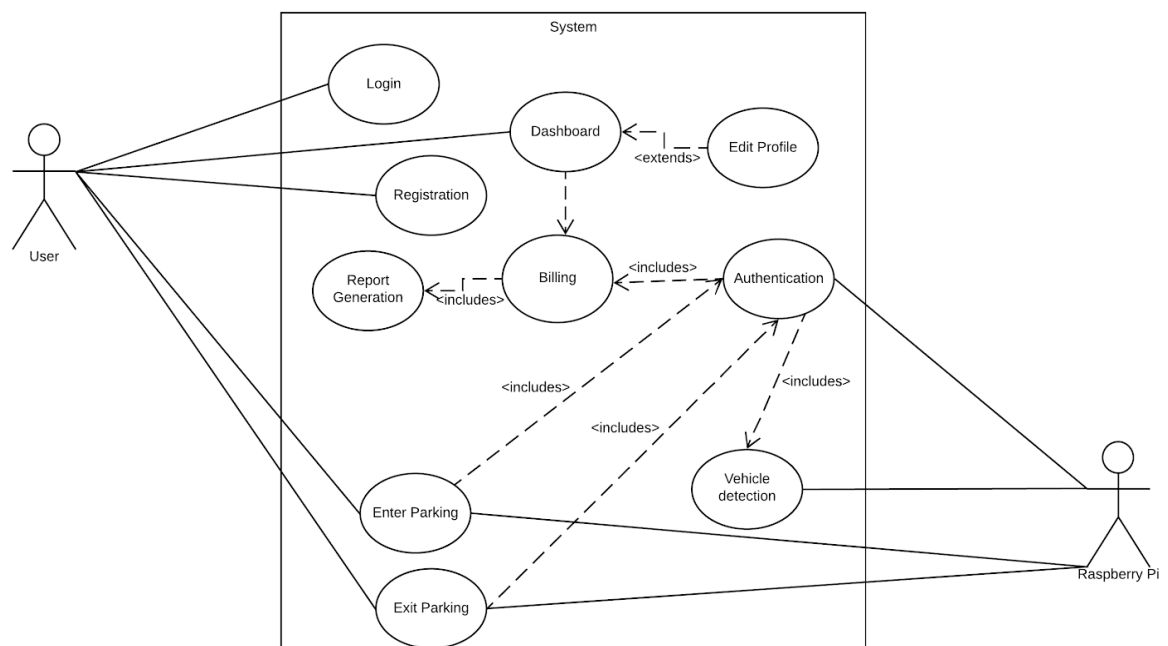


Fig. 3.3.1 Use Case Model

Author – Bhavay Pahuja, Navjot Singh Sandhu, Deepak Lather, Chirag Mahawar

Purpose - Use Case Diagram shows the overall System and its various processing units and relationship among them.

Priority - High Priority

Actors – User and Raspberry Pi

4. Other Non-functional Requirements

1. Performance Requirements

- If the customer's RFID card gets damaged then the client will have to follow the procedure of the unregistered user to enter the parking lot.
- If RFID card is lost/damaged while the vehicle in parking, the user has to issue a QR ticket along with his details (so that entry time may be used to generate bill according to parking time) and follow the procedure of the unregistered user to exit the parking. And he may block his RFID card online and have to register again to get a new RFID card issued to him.
- The image of the vehicle must be at the time when RFID is authenticated.
- A secondary RFID detection module will be activated in case of failure and management team will be informed about the fault.

2. Safety and Security Requirements

- Provide safety/security requirements based on your interview with the client - again you may need to be somewhat creative here. At the least, you should have some security for the mobile connection.
- If no vehicle number is found on the vehicle, then parking permission won't be granted.
- User must keep their RFID card in secure place as RFID cards can be duplicated easily, hence if the user finds any discrepancy, he should block his card immediately so as to minimize losses.
- All the data communication should be encrypted so that security of the network may be enhanced.



Fig 4.2.1 : Secure Communication

Secure communication is when two entities are communicating and do not want a third party to listen in. For that they need to communicate in a way not susceptible to eavesdropping. Secure communication includes means by which people can share information with varying degrees of certainty that third parties cannot intercept what was said.

3. Software Quality Attributes

4.3.1 Stability

Stability is realized by balancing the cost of hardware and labor of manual calibration. You can use the most stable hardware, least shift after long time work and the impedance is ideally matched. All of them will cost unaffordable price. Besides, there are other drawbacks such as high complexity, low reliability and big size. Within the budget, you can choose hardware with suitable reliability, size and weight. Meanwhile, with the help of surveillance with high precision instruments automatically instead of manual calibration to avoid mass labor and manual measurement error, higher accuracy would be realized. At last, suitable hardware and solution for calibration should be chosen

4.3.2 Consistency of Calibration and Measurement

Most of the time, when the instrument is calibrating, the continuous signal is transmitted and measured. Otherwise, in RFID test, the signal is mostly modulated. The difference of the calibration factor in these two modes is the key element of calibration accuracy, The maintenance of different hardware and calibration parameters is a tricky problems too. Software of different calibration parameters files has the same interpretation. With the larger amount of software versions and hardware models, the working load would rise exponentially.

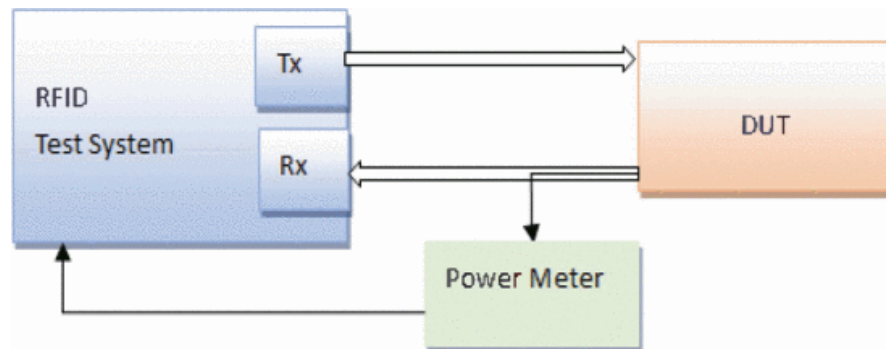


Fig4.3.2.1 : Dynamic Calibration

4.3.3 Adaptation to Most Protocols

The instrument is required to not only have the capability of covering frequency, bandwidth for most protocols in hardware level but also have the adaptation in software development level. This requirement is also for calibration steps which will guarantee the accuracy of measurement results at the proper cost when customizing the new instrument for new protocols.

5. References

- Ishikawa Diagram, Available at: https://en.wikipedia.org/wiki/Ishikawa_diagram
- Fishbone Diagram, Available at: <http://asq.org/learn-about-quality/cause-analysis-tools/overview/fishbone.html>
- RFID, Available at: https://en.wikipedia.org/wiki/Radio-frequency_identification
- Raspberry Pi, Available at: https://en.wikipedia.org/wiki/Raspberry_Pi
- Automation of Parking System by S.C. Hanche, Pooja Munot, Pranali Bagal, Kirti Sonawale & Pooja Pise, Available at: <https://pdfs.semanticscholar.org/a715/f651c45ce40b2544ba5a4ac78a3bc44115d1.pdf>
- Internet of Things, Available at: https://en.wikipedia.org/wiki/Internet_of_things
- Django Documentation, Available at: <https://docs.djangoproject.com/en/2.1/>
- Tensorflow, Available at: <https://www.tensorflow.org/>

Appendix A – Data Dictionary & Abbreviations

DATA DICTIONARY

- **Ishikawa Diagram** - Ishikawa diagrams (also called fishbone diagrams, herringbone diagrams, cause-and-effect diagrams, or Fishikawa) are causal diagrams created by Kaoru Ishikawa that show the causes of a specific event.
- **Internet of Things** - The Internet of Things (IoT) is the network of physical devices, vehicles, home appliances, and other items embedded with electronics, software, sensors, actuators, and connectivity which enables these things to connect and exchange data, creating opportunities for more direct integration of the physical world into computer-based systems, resulting in efficiency improvements, economic benefits, and reduced human exertions.
- **RFID** - Radio-frequency identification (RFID) uses electromagnetic fields to automatically identify and track tags attached to objects.
- **Raspberry Pi** - The Raspberry Pi is a series of small single-board computers
- **Django** - Django is a free and open-source web framework, written in Python, which follows the model-view-template (MVT) architectural pattern
- **SQLite** - SQLite is a relational database management system
- **Flask** - Flask is a micro web framework written in Python.
- **Image Processing** is a method to perform some operations on an image, in order to get an enhanced image or to extract some useful information from it.
- **TensorFlow** - TensorFlow is an open-source software library for dataflow programming across a range of tasks.

ABBREVIATIONS

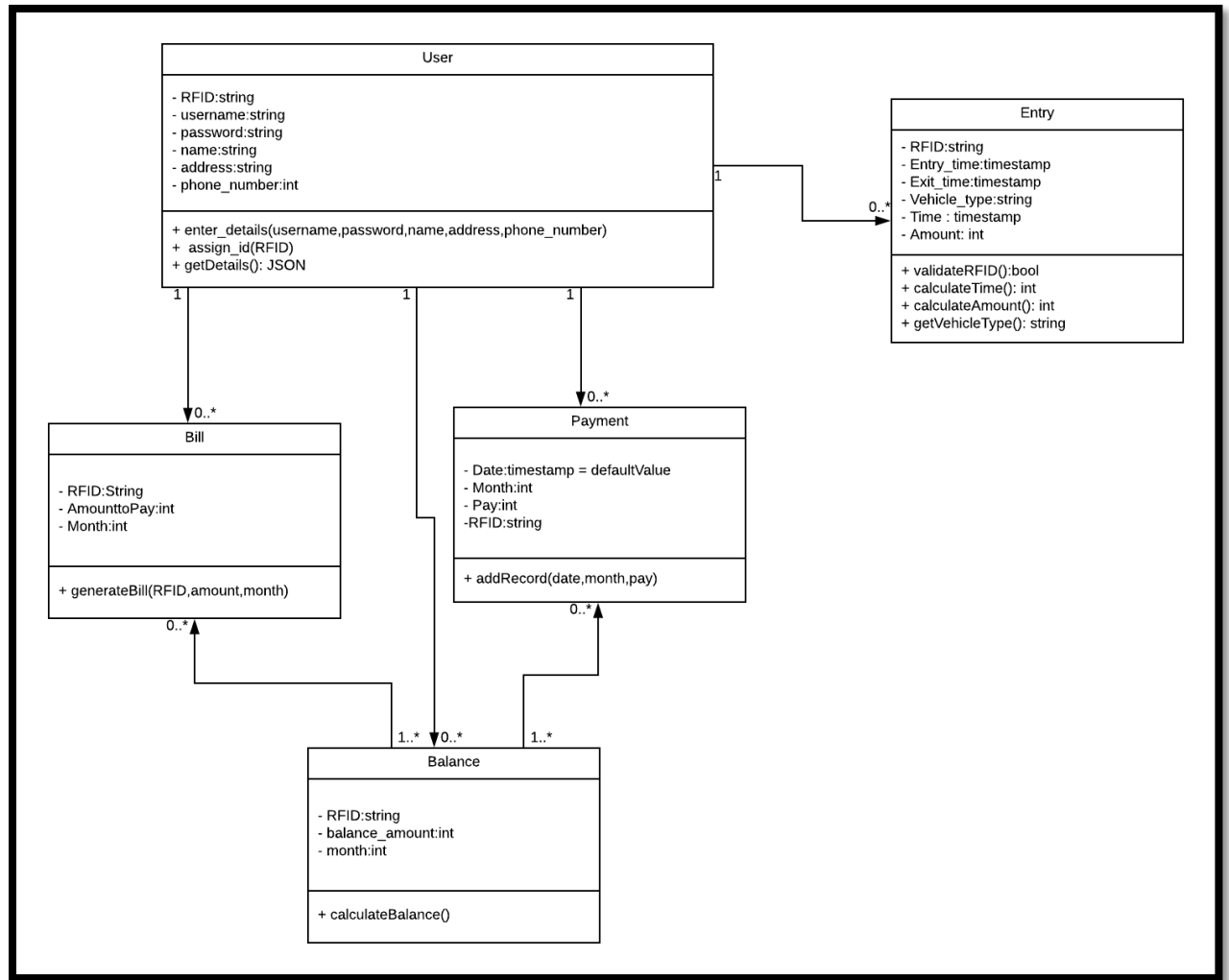
- **RFID** - Radio Frequency Identification
- **IoT** - Internet Of Things
- **AIDC** - Automatic Identification and Data Capture
- **LF** - Low Frequency
- **HF** - High Frequency
- **QR** - Quick Response

Appendix B - Group Log

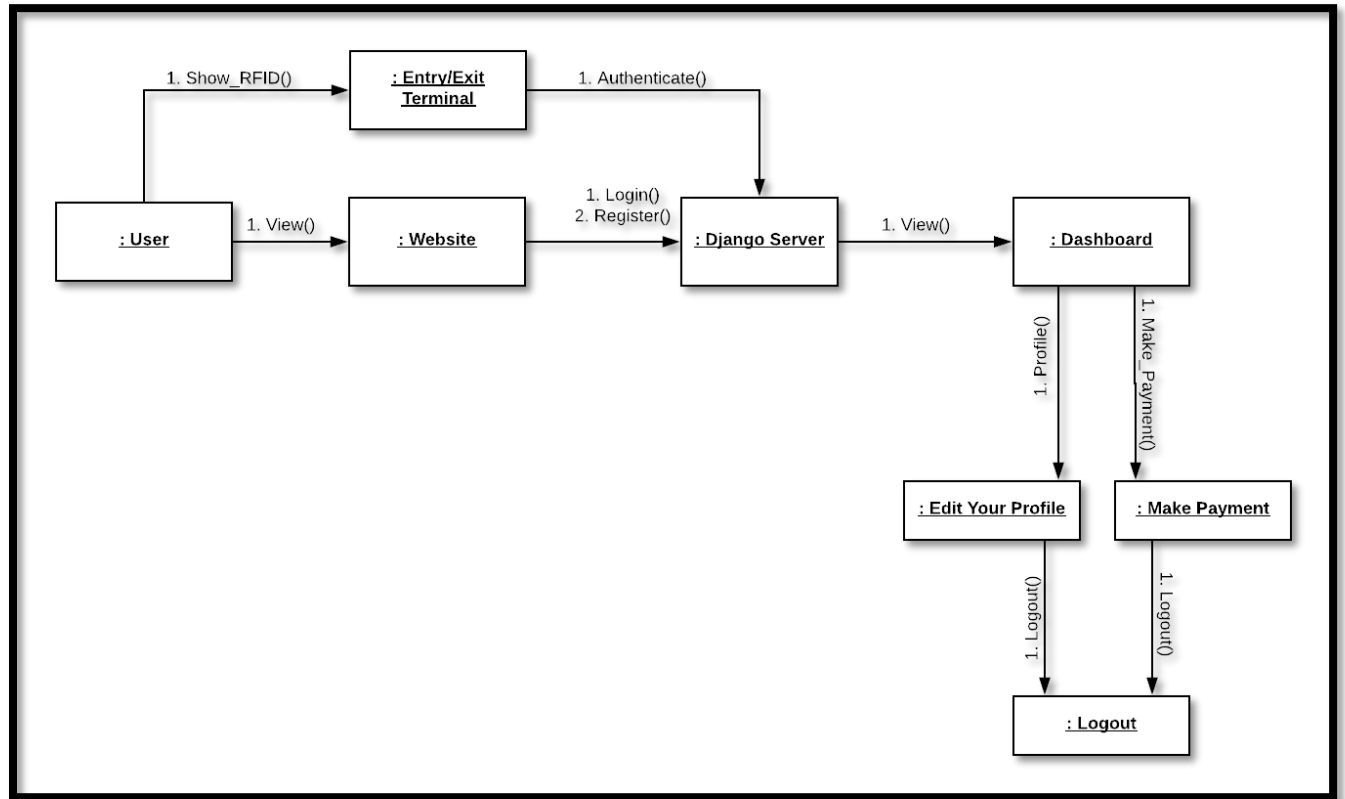
- We started discussion on project on 16th August,2018. Our first motto was to design this system so that user can use it easily.
- We thought about different ways to authenticate including Qr code, but at last,on 23rd August, our team settled on using RFID tags and they can be used with relative ease. We restricted our scope only for registered users as the process used for unregistered user could not provide ease and security.
- In our next meeting, on 26th August, we finalised the payment method. On the next day, we started discussing the frameworks to be used.
- After a week, on 3rd September, we finalised the ways of connecting different parts of software together.
- On 10th September, Use case Diagram was made.
- On 11th September, Ishikawa diagram was made.
- On 16th September, We completed our Srs document .

3. DESIGN PHASE

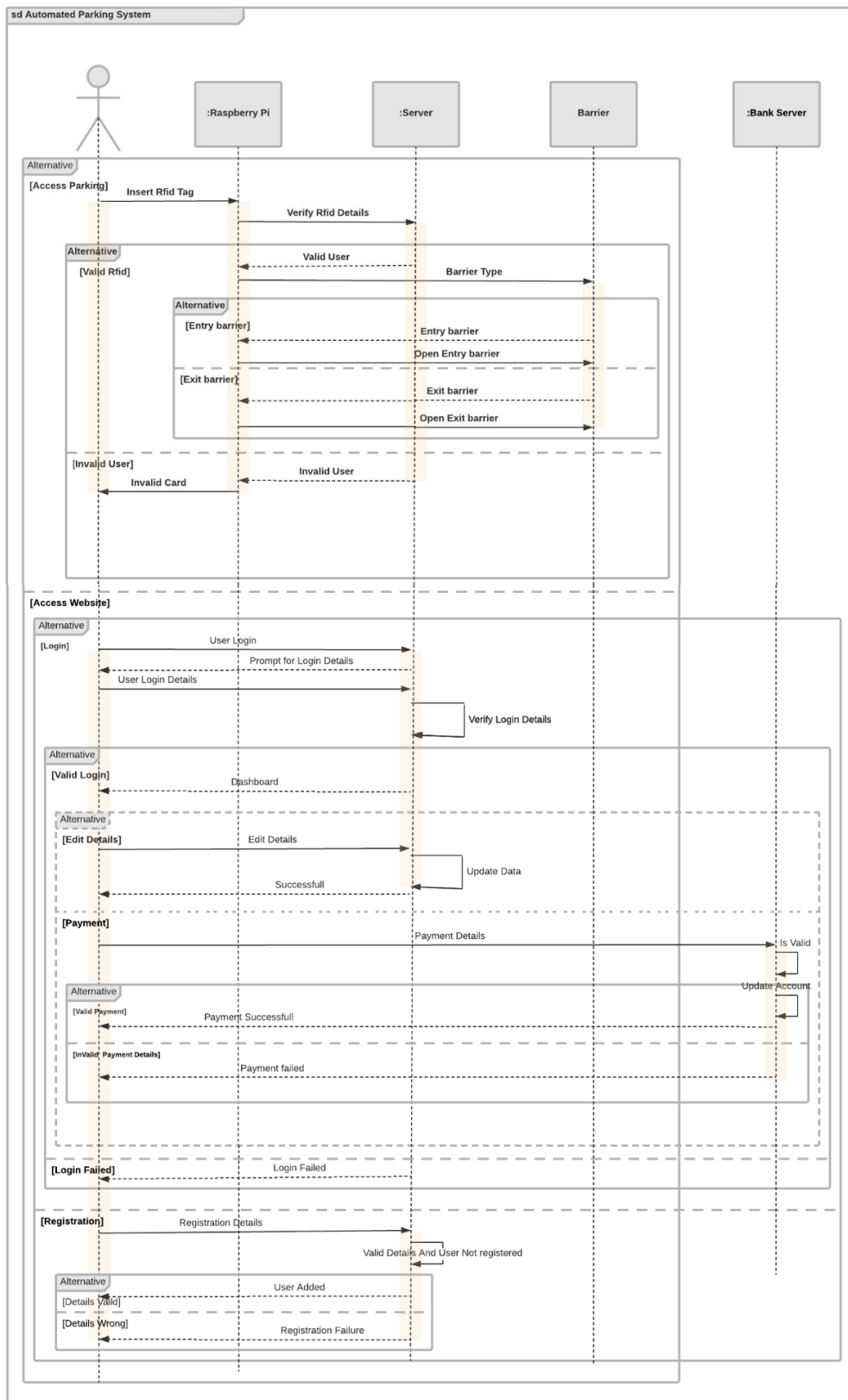
3.1 CLASS DIAGRAM



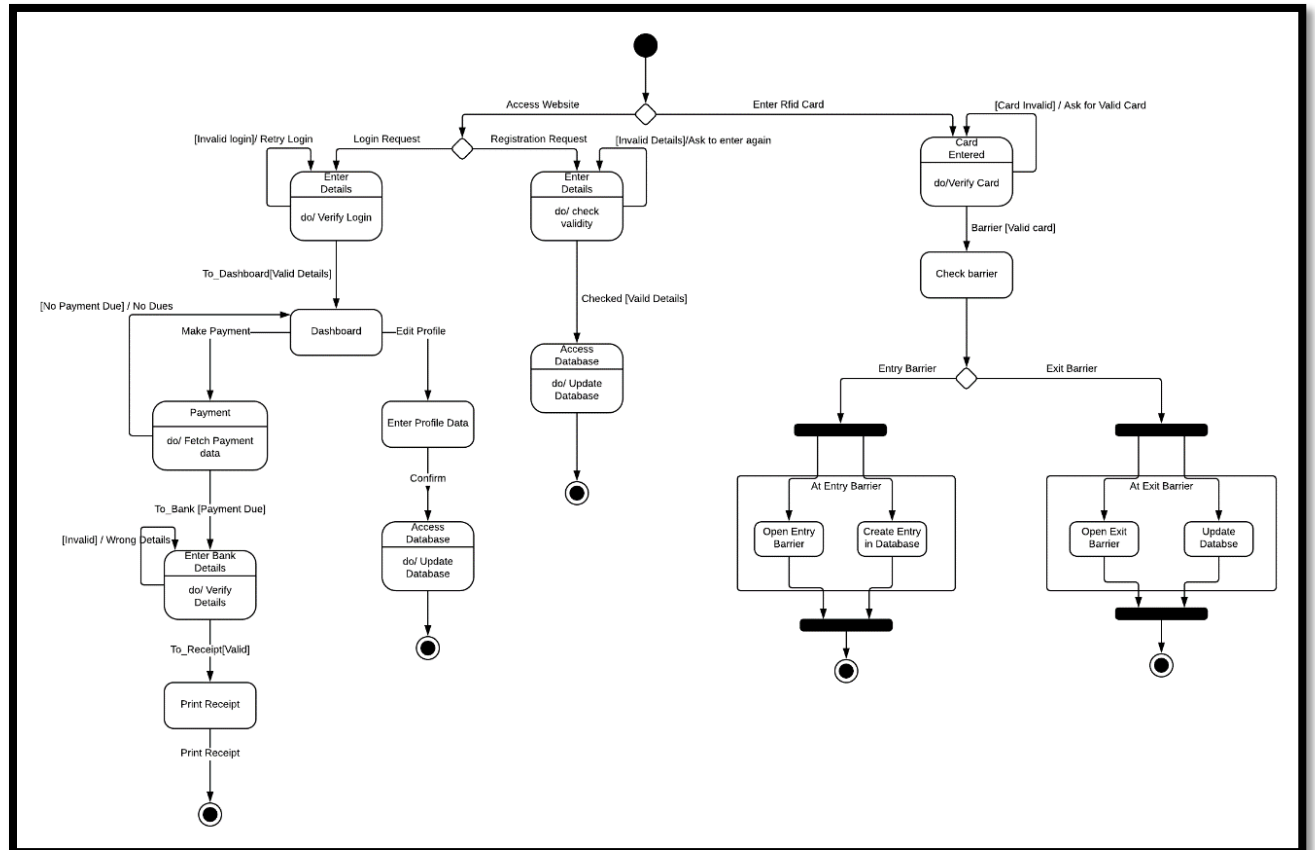
3.2 COLLABORATION DIAGRAM



3.3 SEQUENCE DIAGRAM



3.4 STATE CHART DIAGRAM



4. TESTING PHASE

PRODUCT: Pulse Parking Lot Automation
PRODUCT RELEASE VERSION: 1.0

TEST PLAN
Document Version 1.0

UCS 503
Software Engineering

Submitted by

101603074 – Bhavay Pahuja
101603078 – Chirag Mahawar
101603199 – Navjot Singh Sandhu
101610022 – Deepak Lather

B.E Third Year, Computer Engineering

Submitted to

Ms. Deepali Bhagat

INTRODUCTION

This document is made to check whether the product is working as expected or not.

REFERENCES

Software Requirements Specification v1.0

TEST ITEMS

- Hardware interface
- PULSE Parking Automation Website

FEATURES TO BE TESTED

- RFID Reader at entry & exit terminal
- Login Details
- Registration Details

APPROACH

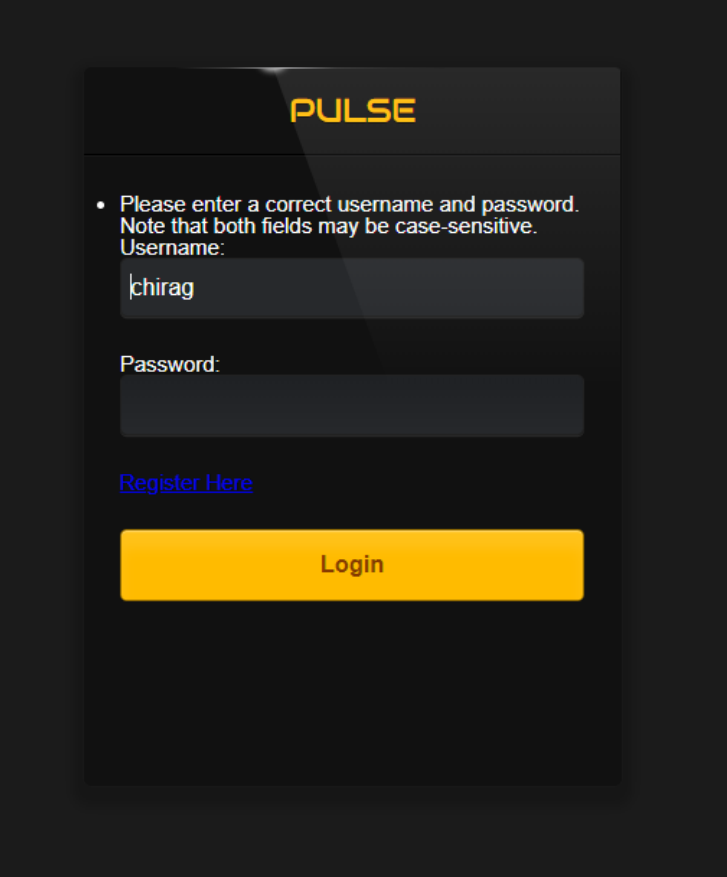
Testing is done by entering various possible options for the various fields in the forms.
Testing method – Manual

TEST ENVIRONMENT

Software & Hardware

TEST CASES & REPORT

Test 1: Incorrect Login Detail

A screenshot of a login interface for a system named 'PULSE'. The interface has a dark background. At the top, the word 'PULSE' is displayed in yellow. Below it, there is a message in white text: 'Please enter a correct username and password. Note that both fields may be case-sensitive.' Under this message, there are two input fields. The first field is labeled 'Username:' and contains the text 'chirag'. The second field is labeled 'Password:' and is empty. Below the password field, there is a blue link that says 'Register Here'. At the bottom of the form, there is a yellow button with the text 'Login' in black.

Expected Output: Error depicting the problem.

Actual Output: Error as shown in the image.

Result: PASS

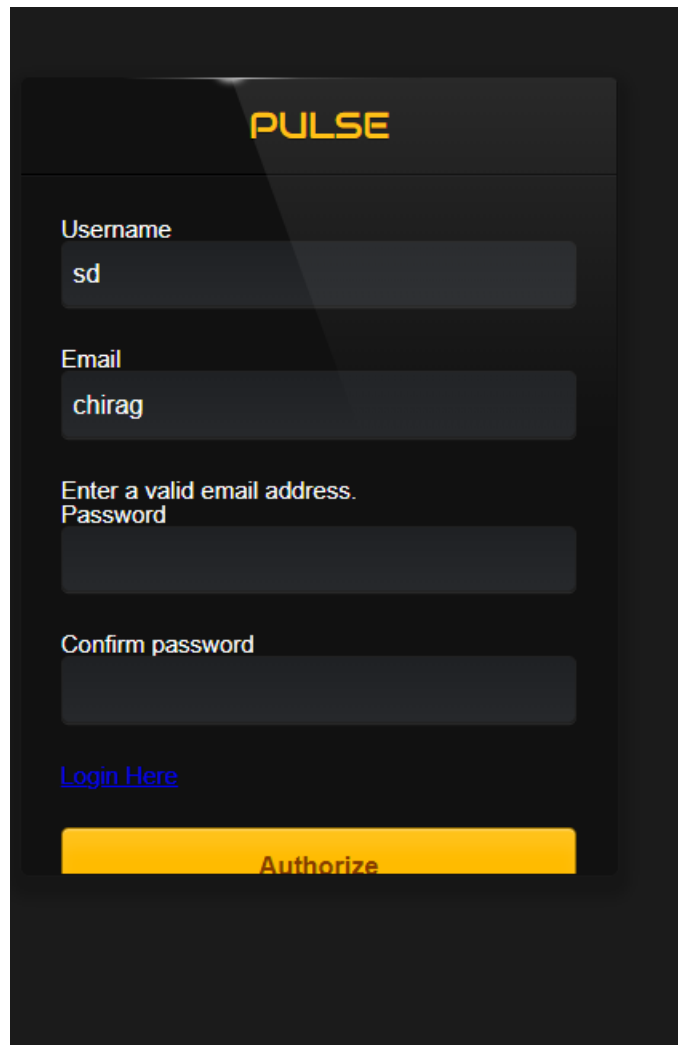
Test 2: Correct Login Detail

Expected Output: Redirection to Dashboard

Actual Output: Dashboard redirected

Result: PASS

Test 3: Incorrect Email Address in Registration form



The image shows a registration form titled "PULSE" in yellow text on a dark background. The form contains the following fields and elements:

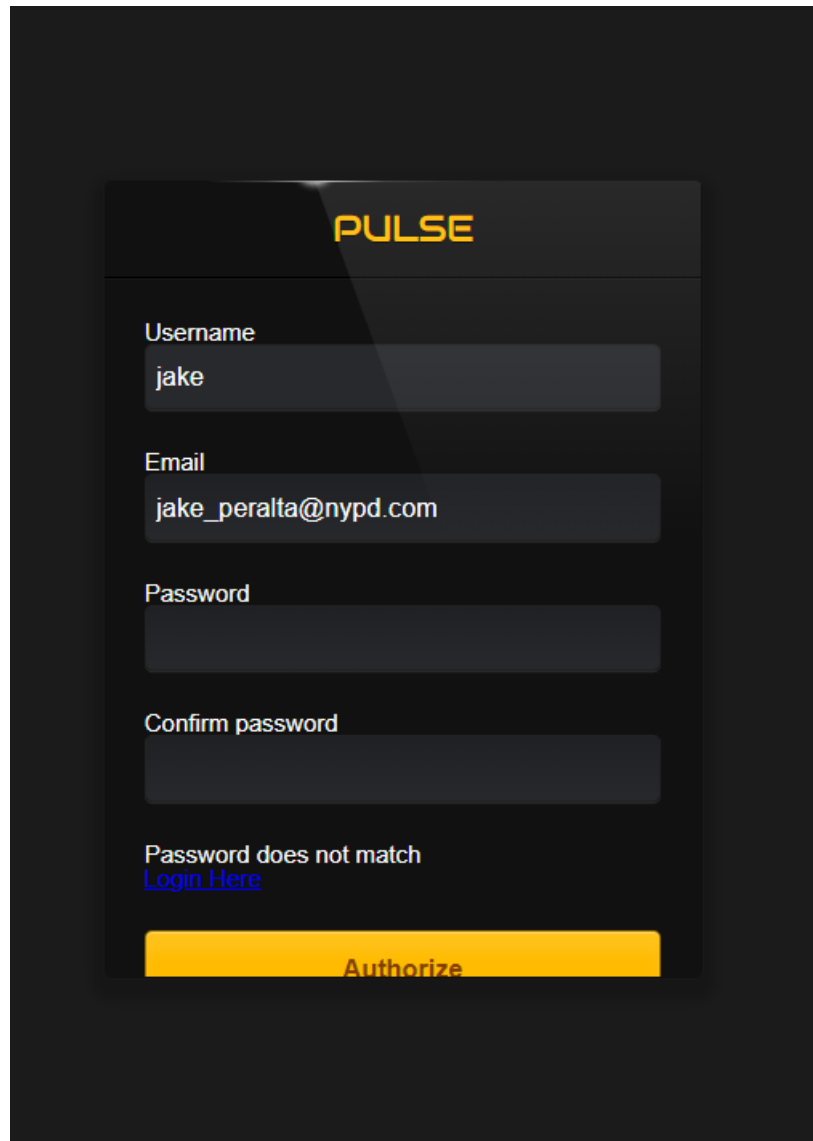
- Username:** A text input field containing the value "sd".
- Email:** A text input field containing the value "chirag".
- Error Message:** Below the email field, the text "Enter a valid email address." is displayed in yellow.
- Password:** A text input field.
- Confirm password:** A text input field.
- Login Link:** A blue hyperlink labeled "Login Here".
- Submit Button:** A yellow button labeled "Authorize".

Expected Output: Error

Actual Output: Error occurred as shown in the image.

Result: PASS

Test 4: Password & confirm password does not match in registration form



The image shows a registration form titled "PULSE" in yellow text on a dark background. The form contains four input fields: "Username" with the value "jake", "Email" with the value "jake_peralta@nypd.com", "Password", and "Confirm password". Below the "Confirm password" field, there is an error message "Password does not match" in red text, followed by a blue link "Login Here". At the bottom of the form is a yellow button labeled "Authorize".

Expected Output: Error

Actual Output: Error occurred as shown in the image.

Result: PASS

Test 5: Incorrect card shown at RFID terminal

```
pi@raspberrypi:~/MFRC522-python $ python newentry.py
Connecting.....
358399749701
login hua
Card not valid
Traceback (most recent call last):
  File "newentry.py", line 37, in <module>
    user_url = response[0]['user']
IndexError: list index out of range
```

Expected Output: Error

Actual Outcome: Error occurred: Card not valid

Result: PASS

Test 6: Correct card shown at entry terminal

```
pi@raspberrypi:~/MFRC522-python $ python newentry.py
Connecting.....
951546313148
login hua
{"id":5,"url":"http://192.168.43.140:8000/api/users/5/","username":"navjot"}
{"id":15,"url":"http://192.168.43.140:8000/api/exits/15/","user":"http://192.168.43.140:8000/api/users/5/","entry_time":"2018-11-25T17:55:20.499723Z","exit_time":null,"time":null,"amount":null}
15
written
14
```

Expected Output: Fetch data & update database

Actual Outcome: Data written in Database

Result: PASS

Test 7: Correct RFID card shown at exit terminal

```
('text' is '15
951546313148
token mila
login hua
get hogya
{"id":5,"url":"http://192.168.43.140:8000/api/users/5/","username":"navjot"}
{}
{"id":15,"url":"http://192.168.43.140:8000/api/exits/15/","user":"http://192.168.43.140:8000/api/users/5/","exit_time":"2018-11-25T17:56:45.471222Z"})
```

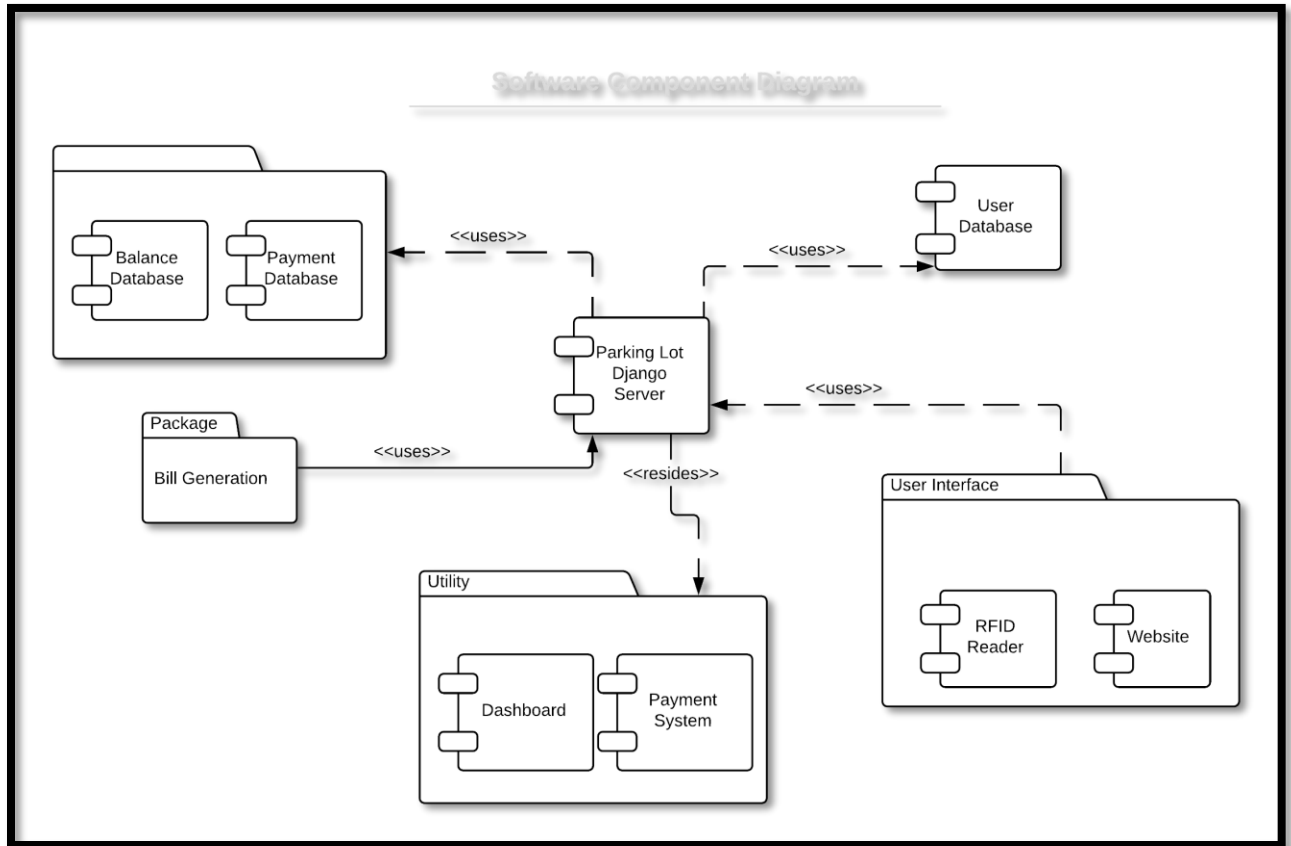
Expected Output: update database

Actual Database: Database updated

Result: PASS

5. DEPLOYMENT PHASE

5.1 COMPONENT DIAGRAM



5.2 DEPLOYMENT DIAGRAM

