Software Engineering Project

License Plate Pass (LPP)

CS308 Software Engineering

Prof. Kanita Karađuzović - Hadžiabdić

Students:

Ahmed Krdžalić

Armin Šarić

Lejla Smajić

Vedad Jahjaefendić

Sarajevo, 18th May 2021

Table of content

1. Software requirements specification document 2
1.1. Introduction 3
1.2. System Features and Use Cases 3
1.2.1. System Features 3
1.2.2. Use Cases 5
2.2.1 Detailed use case diagram 5
1.2.3. Non-Functional Requirements 7
1.2.4 Release plan 7
1.2.5 System evolution 8
1.2.6 Technology 8
2. Design document 10
2.1 Objectives 10
2.2 System decomposition/design 10
2.3 Use case diagram 11
2.4 Dynamic model: sequence diagram 12
2.5 User interface 14
3. Implementation 17
4. Glossary 18
5. References 19

1. SOFTWARE REQUIREMENTS

SPECIFICATION DOCUMENT

This chapter includes Software requirements specification where you can find the expected behaviour of the software system, in this case License Plate Pass. Also, basic information about the system, system features, use cases and use case scenarios as well as non functional requirements, release plan and system evolution.

Revision history:

Rev 1.0 28th December,2020 – initial version, finished business objectives and customer needs;
Rev 1.1 29th December,2020 – added user classes;
Rev 1.2 30th December, 2020 – added UR and FR, created feature tree and table;
Rev 1.3 1st January, 2021 - created use case diagram and table;
Rev 1.4 2nd January, 2021 - created prototype;
Rev 1.5 3rd January, 2021 - added introduction, nonfunctional requirements;
Rev 1.6 4th January, 2021 - final version (for last semester);

Rev 2.0 20th March, 2021 - deleted section 2 from last version, revised and double checked all system features along with user requirements;
Rev 2.1 21st March, 2021 - made release plan and added use cases;
Rev 2.2 24th March, 2021 - added non functional requirements, finished design;

Rev 3.0 20th April, 2021 - made changes to use cases;
Rev 3.1 30th April, 2021 - made changes for system features: deleted SF1 Manage database entirely, divided SF2 Manage opening and closing ramp into two system features: SF1 Manage entrance and SF2 Manage exit. Added SF3 Manage payment and SF4 manage manual override;
Rev 3.2 14th May, 2021 - made changes to system evolution; made changes to use cases and use case scenarios: added use case for paying the parking (UR1.3) and use case for manual override (UR1.4); final version;

1.1. Introduction

There are big traffic jams in front of IUS parking in period from Monday to Friday and there are big crowds and that's why there is need for special system that makes easier entering and exiting the parking. The system is called License Plate Pass and it works with Python image processing to scan the license plates within seconds. With this system a lot of time is saved and traffic jams are avoided.

1.2. System Features and Use Cases

Section 1.2 includes all relevant information about product features, derived user requirements and corresponding functional requirements. Major features are written with short description and feature priority including corresponding user and functional requirements. Use cases for License Plate Pass are also included in this chapter as well as non-functional requirements, release plan and system evolution.

1.2.1. System Features

This section of software design document includes system features that are providing related system capabilities that provide value to a user, and are described by set of functional requirements. Features are prioritized by MoSCoW style prioritization: first state are "Must haves", second state are "Should haves", third one are "Could haves" and last are "Would haves". Also, functional requirements are written in terms of use cases in the table (*Table 1.1*).

SF1: Manage entering IUS parking: After scanning and storing license plate number the ramp opens automatically
Priority: Must have

UR1.1 User shall be able to enter IUS parking
FR1.1.1 System scans and stores license plate number
FR1.1.2 System opens the ramp automatically
UR1.2 User shall be able to pass the ramp
FR1.2.1 System recognizes that user passed the ramp
FR1.2.2 System closes the ramp

SF2: Manage exiting IUS parking: System opens the ramp only if the license plates are in paid bracket
Priority: Must have

UR2.1 User shall be able to exit IUS parking
FR 2.1.1 System scans license plate number
FR 2.1.2 System deletes license plate number from paid bracket

FR 2.1.3 System opens the ramp

UR2.2 User shall be able to pass the ramp

FR2.2.1 System recognizes that user passed the ramp

FR2.2.2 System closes the ramp

SF3: Manage payment for IUS parking: Financial manager puts license plate number in paid bracket

Priority: Must have

UR 3.1 User shall be able to pay bill for parking

FR 3.1.1 Security guard takes money and puts license plate number into paid bracket

UR 3.2 IUS user shall be able to pay monthly for parking

FR 3.2.1 Financial manager puts license plate number into paid bracket for a month

SF4: Manage manual override: In special cases, when diplomats and IUS special guests arrive, the exit ramp opens without paying.

Priority: Could have

UR4.1 Special quest shall be able to enter the IUS

FR4.1.1 Checking by guards if the car plates are from diplomats or IUS special guests.

FR4.1.2 Guard manually opens the ramp on entrance

UR 4.2 Special quest shall be able to exit IUS parking without payment

FR 4.2.1 Checking by guards if the car plates are from diplomats or IUS special guests.

FR4.2.2 Guard manually opens the ramp when diplomats or IUS special guests are exiting from IUS parking lot, without payment

In the table below (*Table 1.1*) primary actors as well as their use cases are described.

Primary actor	**Use cases**
Guest	1. Entering IUS parking 2. Exiting IUS parking
IUS member	1. Pay daily bill 2. Pay monthly bill 3. Entering IUS parking 4. Exiting IUS parking
Special guest	1. Manual override for entering 2. Manual override for exiting
Security guard	1. Manage payment of daily bill 2. Manual override
Financial manager	1. Manage payment of monthly bill

Table 1.1: Primary actors use cases for the software

1.2.2. Use Cases

Use case diagram describes the behaviour of the system from external point of view. Actors in use case diagram are people and other systems. This use case diagram (*Diagram 1.1*) shows relationship between actors and systems. Here are 5 actors which are Guest, IUS member, Security Guard, Special Guest and Financial Manager and IUS Parking as a system.

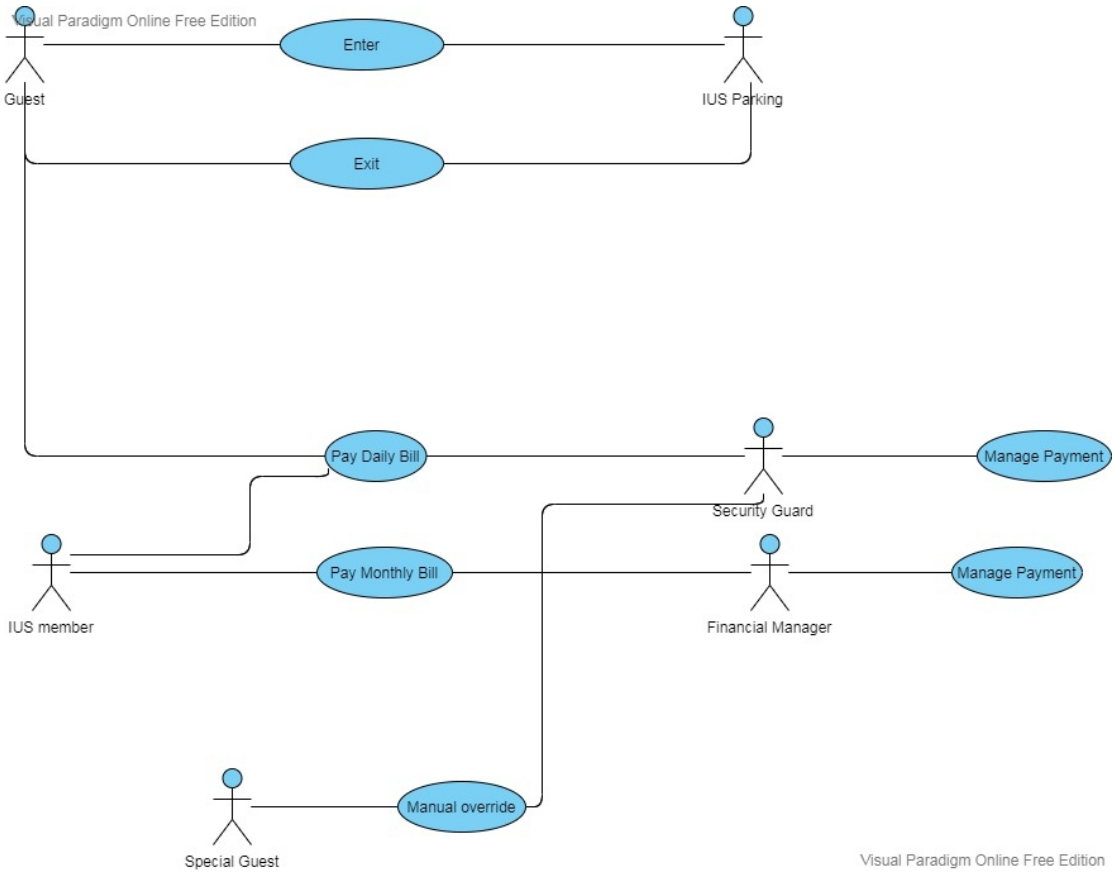


Diagram 1.1: Use cases for the software[1]

2.2.1 Detailed use case diagram

For each use case in section 1.2.2, use cases are explored in more depth in the template below.

Use case ID and title: UR1.1 Entering the IUS parking
Description: User enters IUS parking
Priority: Must have
Pre-condition: User has the valid car plate number
Post-condition: The license plate number are stored. The ramp opens.
Basic Flow:
1. User drives near the ramp
2. System reads and stores license plate number

3. User passes
4. The sensor recognizes if the user passed
5. Ramp closes

What can go wrong

If user drives near the ramp with license plates that are different than format anticipated, security guard on his PC gets an alert with the specific exception handler.

Use case ID and title: UR1.2 Exiting the IUS parking

Description: Users exits IUS parking

Priority: Must have

Pre-condition: User paid for parking

Post-condition: Delete license plate from paid bracket (if user has not monthly subscription). The ramp opens.

Basic Flow:

1. User drives near the ramp
2. System reads license plate number
3. System deletes license plate from paid bracket
4. The sensor recognizes if the user passed
5. Ramp closes

What can go wrong

If user drives near the ramp with car plates that are different than format anticipated, security guard on his PC gets an alert with the specific exception handler.

Use case ID and title: UR1.3 Paying the IUS parking

Description: Users pays for IUS parking

Priority: Must have

Pre-condition: User entered parking and did not pay for parking

Post-condition: Stores license plate into paid bracket (if user has not monthly subscription).

Basic Flow:

1. User enters parking
2. User pays for parking to financial manager or guard
3. Financial manager or guard stores license plate to paid bracket

What can go wrong

Financial manager has not put users license plate into paid bracket although the user has paid.

Use case ID and title: UR1.4 Manual override

Description: Guards manually open and close the ramp

Priority: Could have

Pre-condition: Special guest enters or exits IUS parking

Post-condition: Guard opens and closes the ramp

Basic Flow:

1. Special guest drives near the ramp
2. Guard opens the ramp to for special guest to enter/exit
3. Guard closes the ramp after special guest passes

What can go wrong

Guard can open the ramp manually for someone who is not special guest or diplomat. Guard may not be on his workplace when special guest wants to enter/exit the IUS parking.

1.2.3. Non-Functional Requirements

This section provides information written in table (*Table 1.2*) about Hardware and software interface, software simulation and performance with their definition and more detailed explanation about them.

Requirement	**Definition**	**More details**
NFR1.: Hardware interface	Phone Camera (We will use phone camera as a simulation)	Resolution: HD
NFR2.: Software Interface	Use the database	Database:Relational (MySQL or Postman)
NFR3.: Software Simulation	Mobile application development	Software: Flutter (Dart)
NFR4.: Performance	Internet connection	Speed: 10 Mbps

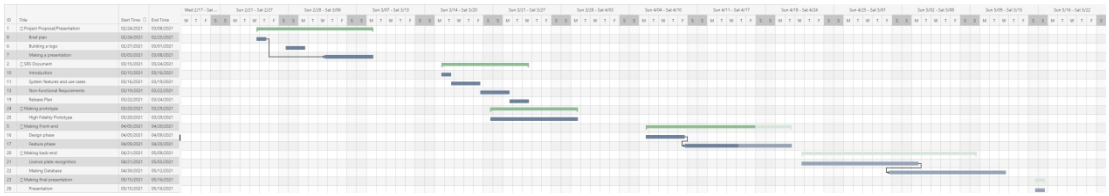
Table 1.2: A list of non functional requirements for the software

1.2.4 Release plan

This section includes a table design (*Table 1.3*) that captures and tracks the features planed for an upcoming release. Timeline graph (Gantt chart)(*Graph 1.1*) where each increment start and end date is clearly set is also included in this section.

Requirement	Duration	Increment	Priority	Dependencies	Release
FR1.1.1	10 days	1	High, Must have	-	1
FR1.1.2	3 days	2	High, Must have	FR1.1.1	1
FR1.2.1	2 days	2	Low, Must have	FR1.1.2	1
FR1.2.2	1 day	2	Low, Must have	FR1.2.1	1
FR2.1.1	2 days	2	High, Must have	FR1.1.1	1
FR2.1.2	2 days	2	High, Must have	FR2.1.1	1
FR2.1.3	1 day	3	Low, Must have	FR2.1.2	1
FR2.2.1	2 days	3	High, Must have	FR1.1.2	1
FR2.2.2	1 day	3	High, Must have	FR1.2.1	1
FR3.1.1	1 day	3	Medium, Must have	FR1.1.1	1
FR3.2.1	2 days	2	Medium, Must have	FR1.1.1	1
FR4.1.1	2 days	2	Low, Could have	-	2
FR4.1.2	1 day	1	Low, Could have	FR4.1.1	2
FR4.2.1	2 days	1	Low, Could have	FR4.1.1	2
FR4.2.2	1 day	1	Low, Could have	FR4.2.1	2

Table 1.3 Table that shows release plan for features planned



Graph 1.1: Graph shows timeline of implementing the system[2]

1.2.5 System evolution

This section describes the fundamental assumptions on which the system is based and any anticipated changes.
Initial version does not include license plate that are different from bosnian license plates (LNNLNNN). System could evolve and recognize more than one format of license plates. For example, in image processing new license plates (from some other country other than Bosnia) should be recognized and stored easily.
In new release of the software system can be planned to add usual special guests license plates into storage so that ramp can be opened automatically when special guest drives near the ramp.

1.2.6 Technology

For developing License Plate Pass, Python programming language is used. Because the software is a simulation of real parking scanner and gates, for image processing Pytesseract library is used and for testing, images from gallery have been processed. Also, MYSQL database for managing payment and storing license plates is used. For the prototype frontend phone camera is used and connected with Flutter for mobile application development. It was decided that prototype will not be used and all of the work is done in Python programming language along with the GUI.

Contribution table:

Project Name: License Plate Pass
MILESTONE: SRS document
Date: 18th May, 2021

Team member name	**Task assigned** (a short description which should not be ambiguous!)	**Status of the tasks** (either completed/uncompleted. If uncompleted by the designated team member, write the member or the team who completed the task instead.)

Ahmed Krdzalic	25% of the work	Completed
Armin Saric	25% of the work	Completed
Lejla smajic	25% of the work	Completed
Vedad Jahjaefendic	25% of the work	Completed

We had 3 calls: on 20th and 30th April and 14th of May that lasted for two hours each. All of the work was done together and all members were present in calls.

2. DESIGN DOCUMENT

This chapter includes design of software product called License Plate Pass which is described in details with system decomposition, use case diagrams, sequence diagrams and user interface.

Revision history:

Rev 1.0 - 20th April, 2021 - Initial version: objectives of the document were written along with modules of the system. In initial version, system had 3 classes and user interface was divided into two sections;
Rev 1.1 - 22nd April 22, 2021 - Modules were deleted because of misunderstanding the task. Objectives rewritten. Sequence diagram made for the whole project;
Rev 1.2 - 23rd April 22, 2021 - Sequence diagram divided into two parts: for entering and for exiting the parking;

Rev 2.0 - 14th May, 2021 - made changes to use cases;
Rev 2.1 - 15th May, 2021 - made changes to class diagram: removed aggregation between IUS parking and user; added sequence diagram for manage payment use case;

2.1 Objectives

This is design system that will include class diagram, use case diagram, sequence diagrams and pictures of user interface with appropriate description of every diagram and picture. The system is simulation made as an application for mobile phone so the gate will be shown only as message.

2.2 System decomposition/design

In this section is the specification of objects to be used in a system. In the diagram below (*Diagram 2.1*) there are twelve separate classes with all necessary attributes, methods and associations.

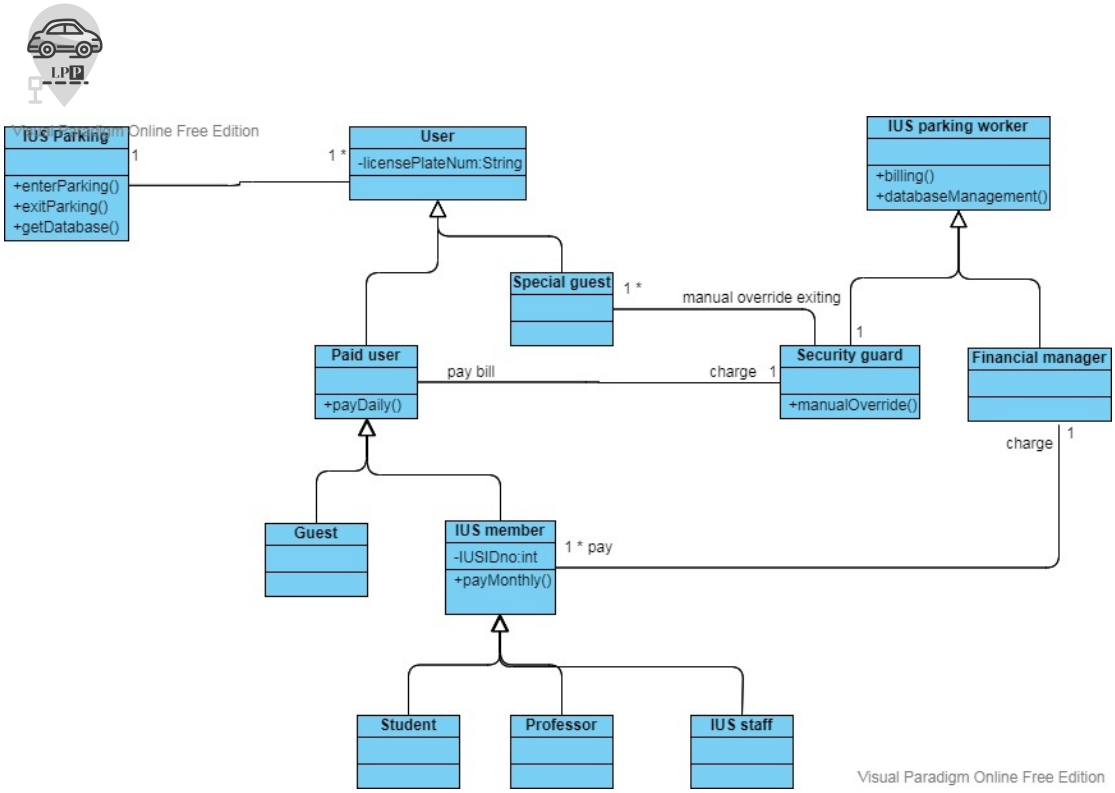


Diagram 2.1: Class diagram for the system[3]

2.3 Use case diagram

Use case diagram describes the behaviour of the system from external point of view. Actors in use case diagram may be people or other systems. This use case diagram (*Diagram 2.2*) shows relationship between actors and systems. Here are 6 actors which are IUS Parking as a system, Guest, IUS member, Security Guard, Special Guest and Financial Manager which is illustrated below.

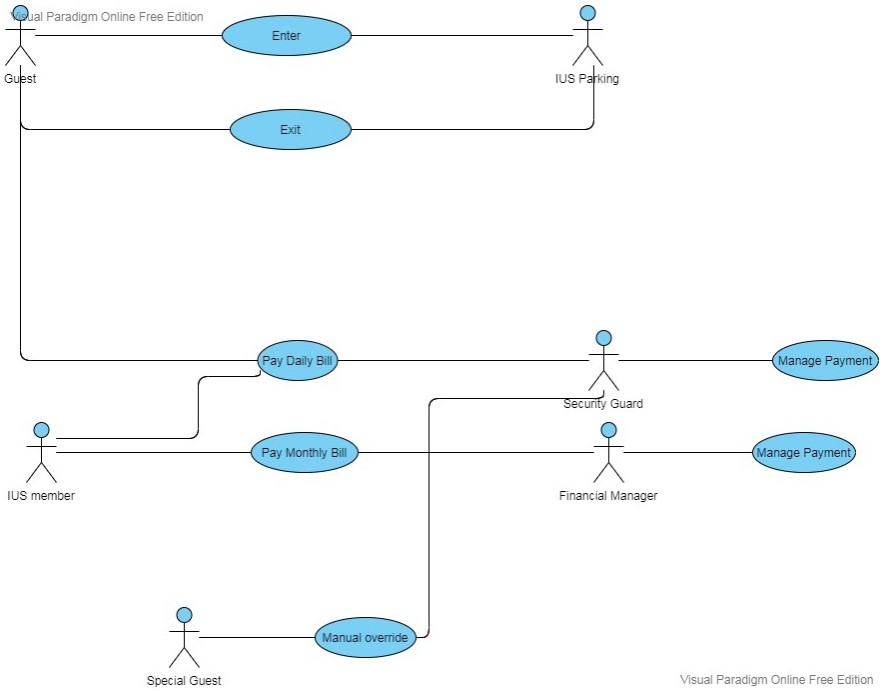


Diagram 2.2: Use case diagram for the system[4]

2.4 Dynamic model: sequence diagram

First Sequence diagram (*Diagram2.3*) shows how the entering to the IUS Parking should look like. The diagram consists of two objects, which are User and IUS Parking System. When the User wants to enter the IUS Parking he must come close to the gate (approachEntranceGate()), in order for the IUS Parking System to scan and store the license plate numbers in the database. When done the IUS Parking System should open the entrance gate to the user.

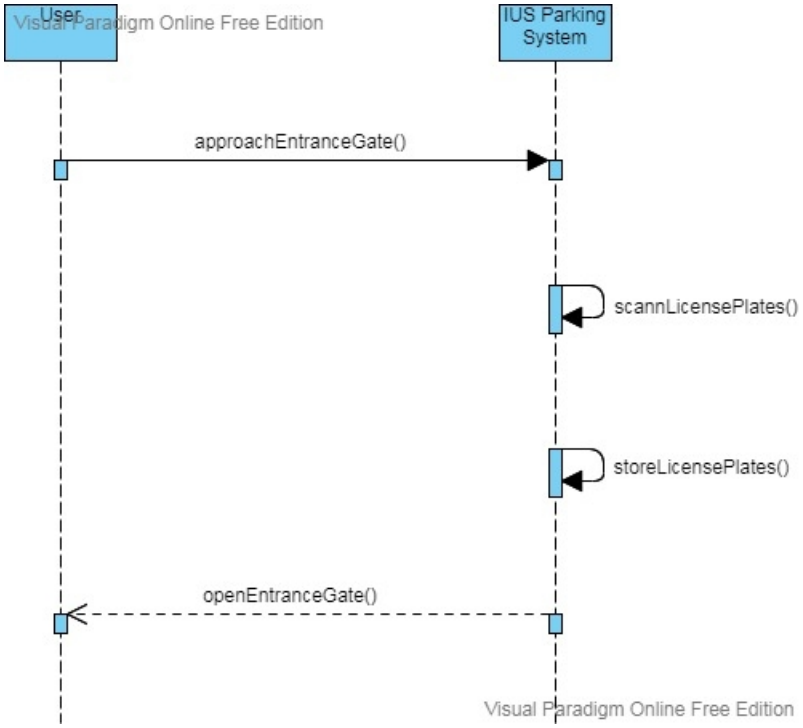


Diagram 2.3: Sequence diagram for Entering IUS parking[5]

Second Sequence diagram (*Diagram 2.4*) shows how the exiting from the IUS Parking should look like. The diagram consists of three objects, which are User, IUS Parking System and IUS Worker, and two alt cases. When User wants to exit the IUS Parking, the IUS Parking System should scan users license plate numbers. Then there are two cases. First one is if user is not special guest of IUS, then the IUS Parking System should check paid bracket of the User. If the User has paid a bill for parking, the exiting gate should be opened, else User needs to pay a bill for parking to the IUS Worker which will put him into paid bracket what should allow him to exit from IUS Parking. But if user is Special Guest of IUS then the IUS Worker should manually open the exiting gate.

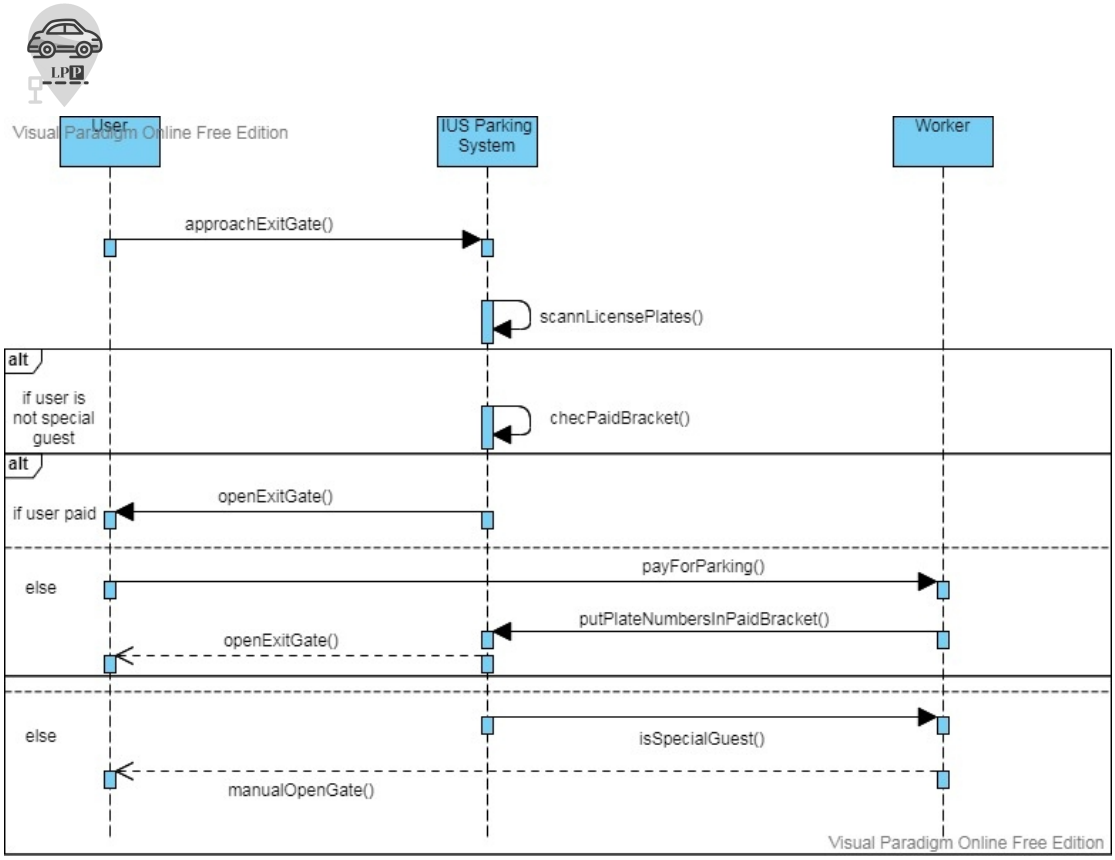


Diagram 2.4: Sequence diagram for exiting IUS parking[6]

Third Sequence diagram (*Diagram 2.5)* shows how the billing should look like. The diagram consists of four objects, which are Guest, IUS Member, Security Guard and Financial Manager. Guest has only one option which is to pay daily bill to Security Guard which puts his license plates in a paid bracket and allow him to get out from the IUS Parking. Unlike a guest, IUS members (students, professors, other IUS staff) in addition to daily payment option, they also have an option to pay a monthly bill. Like guests, they pay daily bills to Security Guard, while monthly bills they pay to Financial Manager. After they pay, Security Guard and Financial Manager put their licence plates in paid bracket and they can go out from IUS Parking.

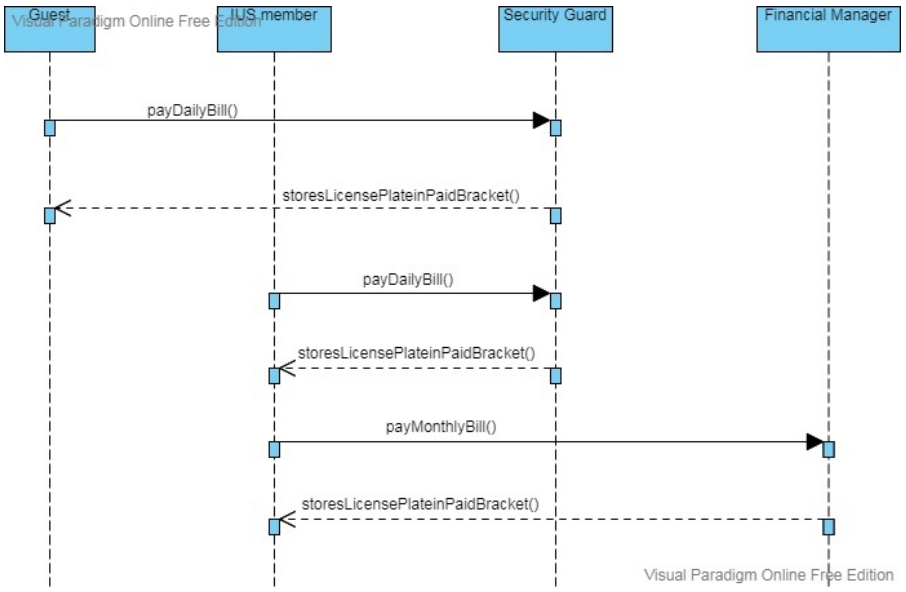


Diagram 2.5: Sequence diagram for Managing payment[7]

2.5 User interface

The software is a simulation of the real parking scanner and it is not made for usage in the market since the system needs to be implemented in a computer with real camera as scanner.

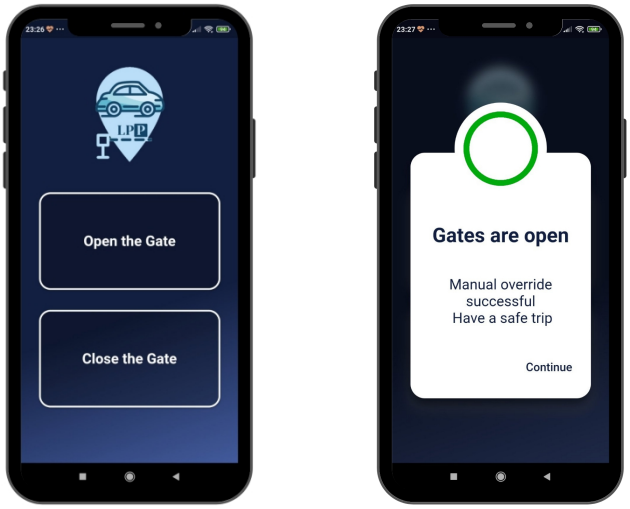


Image 2.1: Manual overriding in the application[7]

System allows manual overriding (*Image 2.1*) which is simple and easy to use. Two buttons: Open the Gate and Close the Gate will open/close the gate without scanning license plates, storing in database or anyhow checking the vehicle entering or exiting the gate.

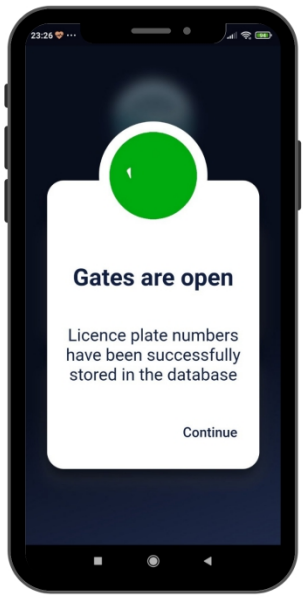


Image 2.2: Message after entering the parking[8]

After scanning license plates and storing them, the user will get the message (*Image 2.2*) that the gates are open and license plates are successfully stored in the database. This way, gate simulation is made and the user can know that he could pass the gate.

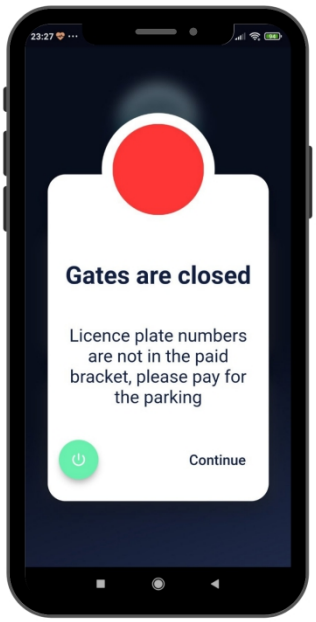


Image 2.3: Message when bill is not paid[9]

The second message (shown in image 3) will occur if the user wanted to pass the gate but still haven't paid the bill so the gates will remain closed. This way user will know that he cannot pass the get until he pays.

Contribution table:

Project Name: License Plate Pass
MILESTONE: Design document
Date: 18th May, 2021

Team member name	**Task assigned (**a short description which should not be ambiguous!**)**	**Status of the tasks** (either completed/uncompleted. If uncompleted by the designated team member, write the member or the team who completed the task instead.)
Ahmed Krdzalic	25% of the work	Completed
Armin Saric	25% of the work	Completed
Lejla smajic	25% of the work	Completed
Vedad Jahjaefendic	25% of the work	Completed

We had two calls: on 14th and 15th of May that lasted two hours each. All of the work was done together and all member were present in calls.

3. IMPLEMENTATION

This chapter includes relevant information for the implementation of the software.

Project title: License Plate Pass

Team members:
Ahmed Krdžalić, 180302008
Armin Šarić, 180302016
Lejla Smajić, 180302051
Vedad Jahjaefendić, 180302020

Link to github repistory:
https://github.com/ahmedkrdzalic/LPP_GUI
*All of the work was done during calls on one computer and that's why only one person uploaded the code.

Revision history:

Rev 1.0: 18th May, 2021 - final version of the code;

Contribution table:

Project Name: License Plate Pass
MILESTONE: Implementation
Date: 18th May, 2021

Team member name	**Task assigned** (a short description which should not be ambiguous!)	**Status of the tasks** (either completed/uncompleted. If uncompleted by the designated team member, write the member or the team who completed the task instead.)
Ahmed Krdzalic	25% of the work	Completed
Armin Saric	25% of the work	Completed
Lejla smajic	25% of the work	Completed
Vedad Jahjaefendic	25% of the work	Completed

All of the work was done in several calls where all team members were present.

4. GLOSSARY

This section explains terms used in the document which may not be familiar to reader
[1] Paid bracket: It is a term for place database where the system stores license plates from users that paid for parking
[2] Financial manager: the person who manages payment for IUS parking and stores licence plates into paid bracket.
[3] L - letter in license plate
[4] N - number in license plate

5. REFERENCES

This section includes references used for writing/drawing this document.
[0] - Sommerville, Software engineering (10th edition). 2016.
[1], [3], [4], [5], [6] - Visual studio for diagrams design: https://online.visual-paradigm.com/
[2] - Full size of *Gantt chart*: https://drive.google.com/file/d/1WfuWHhAOrdgt5kkW-cX0Vqz6y2l6oCja/view?usp=sharing
[7], [8], [9] - Canva for pictures of interface design: https://www.canva.com/

Link to Trello:

https://trello.com/b/kdipAqDp/software-engineering-project