Smart Gate System

Odelia Hochman | odeliamos0@gmail.com

Efrat Cohen | cohenefratoni747@gmail.com

Instructor : Dr. Amos Azaria

Software Design Document

Version: 1.0 **Date: 29/12/2020**

TABLE OF CONTENTS

1. INTRODUCTION

1.1 Purpose

This software design document describes the architecture and system design of the "Smart Gate System". It will illustrate the appearance, the detailed structure of the components, and a complete description of the using the system. It is assumed that the reader has read the SRS document since this document also defines the implementation details of the desired behavior given the requirements within it.

1.2 Scope

The system is designed for parking lots where management wants to allow access only to authorized persons. The system is suitable for all types of parking lots: private parking lots, employee parking lots, etc. Our system will be made up of two parts: the first part is a system that recognizes the driver's face and license plate of his vehicle using algorithms, and the second part is an application designed for the parking lot guard who will connect to the system of face recognition and license plate so that the guard can know in real-time who enters the parking lot.
If the system fails to identify the face and license plate, the guard will have to verify details with the driver (ID number, Phone number, etc) and open the parking lot gate for him through the app.
The system will also save security costs by having the guard supervise several parking lots at the same time.

1.3 Overview

This SDD document is divided into six sections to provide a complete and understandable perception of the system to the target readers. The first section is mostly about the scope and purpose of the document. In the second part, system overview, a general description of the software system including its functionality and matters related to the overall system and its design is provided. The third section states the design considerations and consists of two parts. In the first part, design assumptions, dependencies, and constraints of the system are defined. In the second part, design goals and guidelines are given in terms of reliability, usability, portability, and extensibility of the system. The organization of the data structures of the system is explained in section four. Subsequently, a data dictionary is provided to provide a detailed description of the system's major data, including data objects, their attributes, and methods. The sixth section is all about the user interface design. In this section, the functionality and expected features of the user interface is given. Also, some possible screenshots showing the interface from the user's perspective are provided and the purpose of the screen objects are explained.

1.4 Definitions and Acronyms

- ID - Identifier
- SRS - Software Requirements Specification
- DB - DataBase
- XML- Extensible Markup Language
- JSON - JavaScript Object Notation. An open standard file format, and data interchange format, that uses human-readable text to store and transmit data objects consisting of attribute–value pairs and array data types (or any other serializable value).
- Raspberry Pi - Raspberry Pi is a series of small single-board computers.
- LCD - Liquid Crystal Display. electronically modulated optical device that uses the light-modulating properties of liquid crystals combined with polarizers.
- Owner - Owns the system.
- Client - The people who enter the parking lot.
- User - The guard who uses the app to authenticate the people entering the parking lot.

2. SYSTEM OVERVIEW

- An application consists of several screens and each screen has its own unique purpose.
- All the logic and algorithms of the program are not provided to the user, he can only see the results and behavior of the program.
- The system contains three types of users:
  1. Owner- this user has access to the entire system without permission restriction. He can update the details of the authorized persons in the system and also he has the option to see the license plates that have entered the parking lot in real-time.
  2. Parking lot guard - this user has access to the app only. It has the option to verify the driver information in the system manually and in addition to open the parking lot gate through the app.
  3. Driver - this user has no access to the system other than pressing the button in the parking lot entry position to open the cameras.

- To use the system, the users must be familiar with the graphical interface described in section 6.2.

3. SYSTEM ARCHITECTURE

3.1 Architectural Design

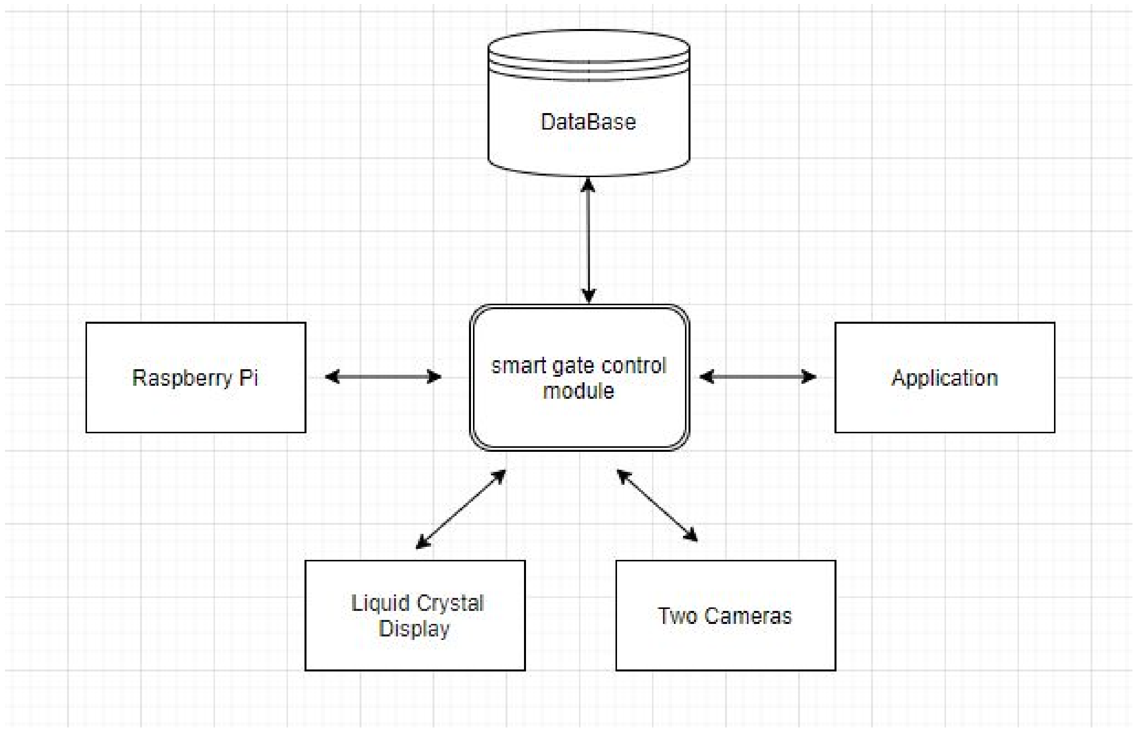


Fig.1

- Database - the system retrieves data about the driver's information in order to check the correspondence between them and images obtained in real-time using image processing algorithms of face recognition and license plate identification.

- Application - receives the images taken in real-time so that the guard can know who entered the parking lot, and can see if the details have been verified in the system. If there is a problem with identification, the guard must verify the driver's details manually in order for the parking lot gate to open.

- Two cameras - a camera that captures the driver's face, and a second camera that captures the vehicle's license plate.

- Liquid Crystal Display - a monitor through which the driver is given the opportunity to watch the data identification process until the parking lot gate opens.

- Raspberry Pi - a computer through which the cameras are linked, in which the system itself for face recognition and license plate recognition is located.

3.2 Decomposition Description

The following diagram "Data Flow Diagram" is a graphical means of describing activities (processes, functions) and the flow of information between them. Its purpose is to display the processes performed in the Smart Gate System and the information transmitted between the processes, i.e. the input required to perform an activity and the output generated as a result of its execution.

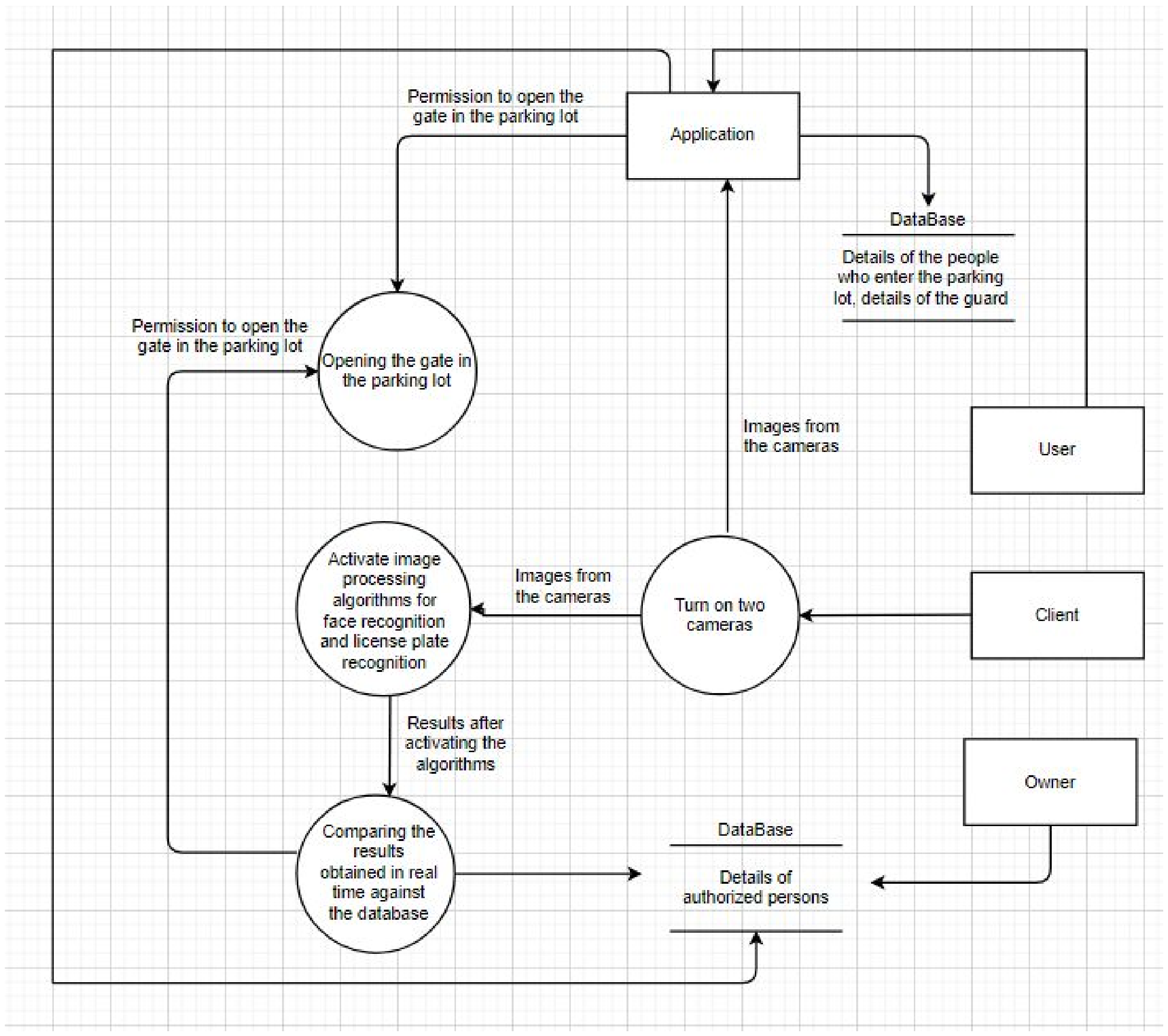

Permission to open the gate in the parking lot
Application
DataBase
Details of the people who enter the parking lot, details of the guard
Permission to open the gate in the parking lot
Opening the gate in the parking lot
Images from the cameras
User
Activate image processing algorithms for face recognition and license plate recognition
Images from the cameras
Turn on two cameras
Client
Results after activating the algorithms
Owner
Comparing the results obtained in real time against the database
DataBase
Details of authorized persons

Fig.2 - Data Flow Diagram

The following diagram shows the "Sequence Diagram" which shows the process of the system from the moment the cameras are turned on in the parking lot by the driver until the opening of the parking lot gate without the intervention of the guard.

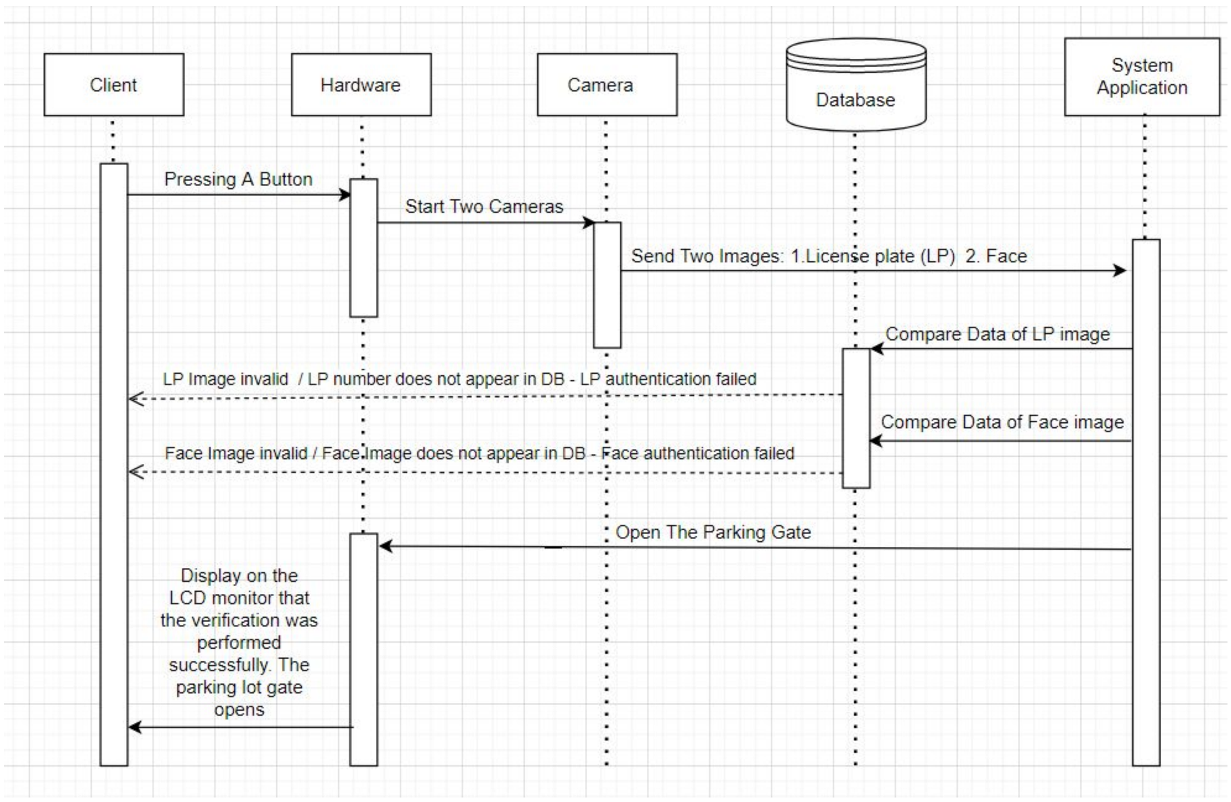

Client
Hardware
Camera
Database
System Application
Pressing A Button
Start Two Cameras
Send Two Images: 1.License plate (LP) 2. Face
Compare Data of LP image
LP Image invalid / LP number does not appear in DB - LP authentication failed
Compare Data of Face image
Face Image invalid / Face Image does not appear in DB - Face authentication failed
Open The Parking Gate
Display on the LCD monitor that the verification was performed successfully. The parking lot gate opens

Fig.3 - Sequence Diagram

4. DATA DESIGN

4.1 Data Description

The following diagram "Entity Relationship Diagram" consists of entities and the relationships between them. Each entity represents an object that you want to represent in the database. Entities have attributes depending on the object they represent, and relationships between entities themselves or other entities.

According to the diagram:

- The owner has permission to add and update the details of the people who are authorized to enter the parking lot.
- Entry and departure times of the guard will be reserved in the DB.
- The client's personal details including the license plate of his vehicle and his face image will be kept in DB.

The system stores data in the following tables:

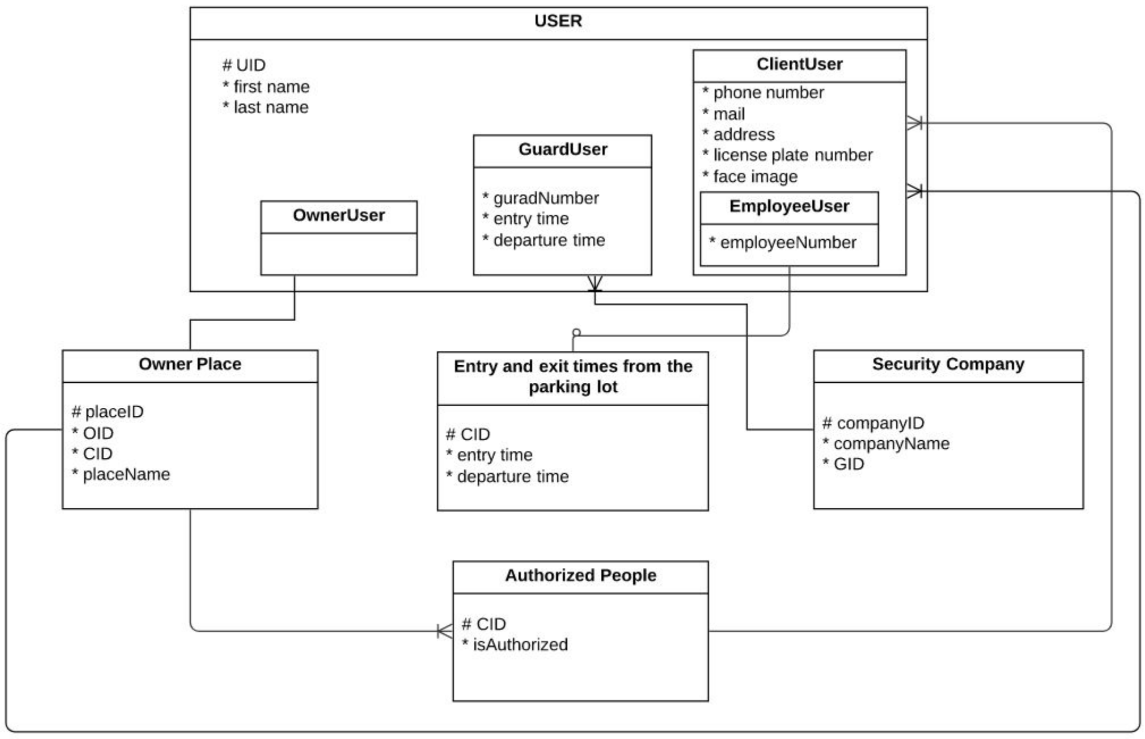


Fig.4 - Entity Relationship Diagram

4.2 Data Dictionary

Alphabetically list the system entities or major data along with their types and descriptions. If you provided a functional description in Section 3.2, list all the functions and function parameters. If you provided an OO description, list the objects and its attributes, methods and method parameters.

Functions:

- openGate() – the function opens the gate after the details have been successfully verified.
- getImages() – the function activates two cameras and returns two images: a license plate image and a face image.
- faceRecognition(Image1) – the function activates the face recognition algorithm. Gets a face image and returns a result from activating the algorithm.
- licensePlateRecognition(Image2) - the function activates the license plate recognition algorithm. Gets an image of a license plate and returns a string with the license plate number in numbers and letters.
- checkLicensePlateAgainstDB() - the function returns true if details verification of license plate number was performed successfully, otherwise return false.
- checkFaceAgainstDB() -the function returns true if details verification of face recognition was performed successfully, otherwise return false.
- sendImagesToApp() - the function sends the images that were taken in real-time to the app.
- sendIdentificationResultsToApp() - the function sends True/False value to the app according to the identification results from two other functions.

APP:

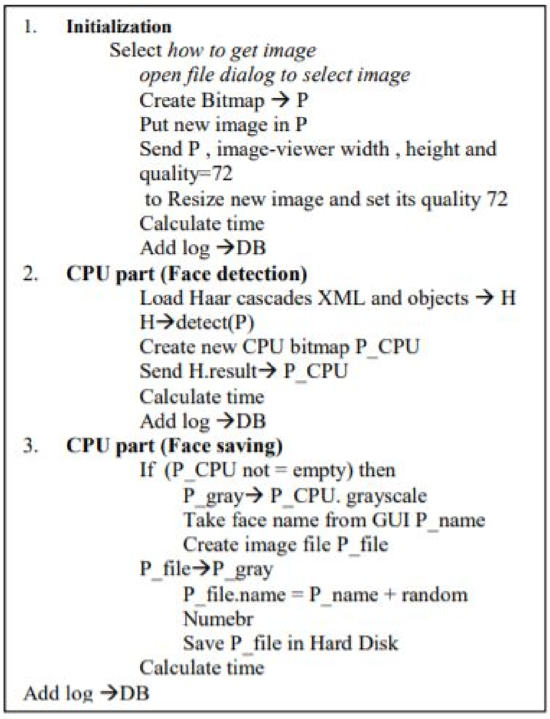
- verificationDetailsAgainstDB() - the function retrieves from the DB the details of the authorized person and compares them with the details it received. Returns true if the details have been successfully verified, otherwise returns false.
- openGate() – the function opens the gate after the details have been successfully verified.
- connectAppToDB() - the function connects the application to the DB.

5. COMPONENT DESIGN

In this project, we use two main algorithms, a face recognition algorithm and a license plate recognition algorithm.

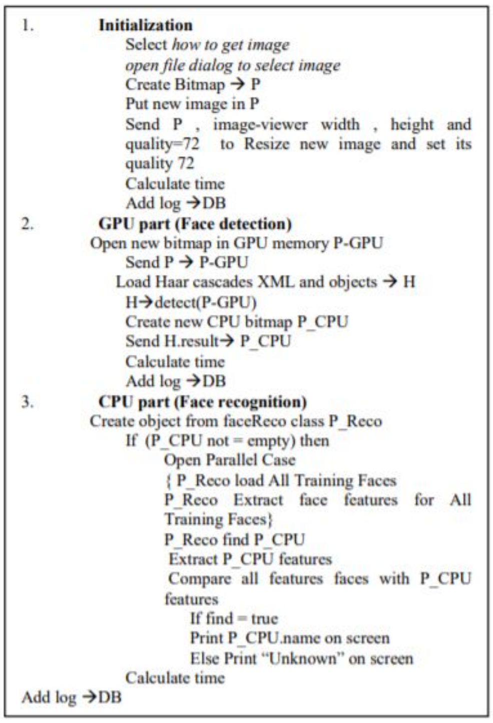
- Face Recognition Algorithm:

  The CPU Mono Training Form enables the user to select the source of the image (i.e., the image comes from a DB). After the system detects a face, the user can write the person's name in the text filed, click the save button to write result log in the database, save the face image and the name in the image's DB, this form is running in a single-core CPU. Unlike the training phase, which is implemented in the CPU Mono model, the recognition phase has four separate models. An image from the DB can be selected, this image will display in the image box, then the system will detect the face and search if this face is predefined in the database to get face name in the text filed and then click save button to write the result log in the database.
  Time will be measured in each step in training and recognition models and also the overall time (for total steps) will be measured and saved in the DB to measure the performance factor in later procedures.
  The following algorithm are written in pseudocode. [1]



```
1.  Initialization
        Select how to get image
            open file dialog to select image
            Create Bitmap → P
            Put new image in P
            Send P , image-viewer width , height and
            quality=72
             to Resize new image and set its quality 72
            Calculate time
            Add log →DB
2.  CPU part (Face detection)
            Load Haar cascades XML and objects → H
            H→detect(P)
            Create new CPU bitmap P_CPU
            Send H.result→ P_CPU
            Calculate time
            Add log →DB
3.  CPU part (Face saving)
            If  (P_CPU not = empty) then
                P_gray→ P_CPU. grayscale
                Take face name from GUI P_name
                Create image file P_file
            P_file→P_gray
                P_file.name = P_name + random
                Numebr
                Save P_file in Hard Disk
            Calculate time
Add log →DB
```

Fig.5
The Pseudo Code for Training Phase



```
1.      Initialization
            Select how to get image
            open file dialog to select image
            Create Bitmap → P
            Put new image in P
            Send P , image-viewer width , height and
            quality=72  to Resize new image and set its
            quality 72
            Calculate time
            Add log →DB
2.      GPU part (Face detection)
        Open new bitmap in GPU memory P-GPU
            Send P → P-GPU
          Load Haar cascades XML and objects → H
            H→detect(P-GPU)
            Create new CPU bitmap P_CPU
            Send H.result→ P_CPU
            Calculate time
            Add log →DB
3.      CPU part (Face recognition)
        Create object from faceReco class P_Reco
            If  (P_CPU not = empty) then
                Open Parallel Case
                { P_Reco load All Training Faces
                P_Reco Extract face features for All
                Training Faces}
                P_Reco find P_CPU
                 Extract P_CPU features
                 Compare all features faces with P_CPU
                features
                    If find = true
                    Print P_CPU.name on screen
                    Else Print "Unknown" on screen
            Calculate time
Add log →DB
```

Fig.6
The Pseudo Code for the Recognition Phase

- License Plate Recognition Algorithm:
  Our system contains two main modules. Given an image sequence as input, the first module, the detection module, applies algorithms for detecting and tracking of license plates. In the second module, segmentation and, subsequently, character recognition is performed to generate the final result.[2]

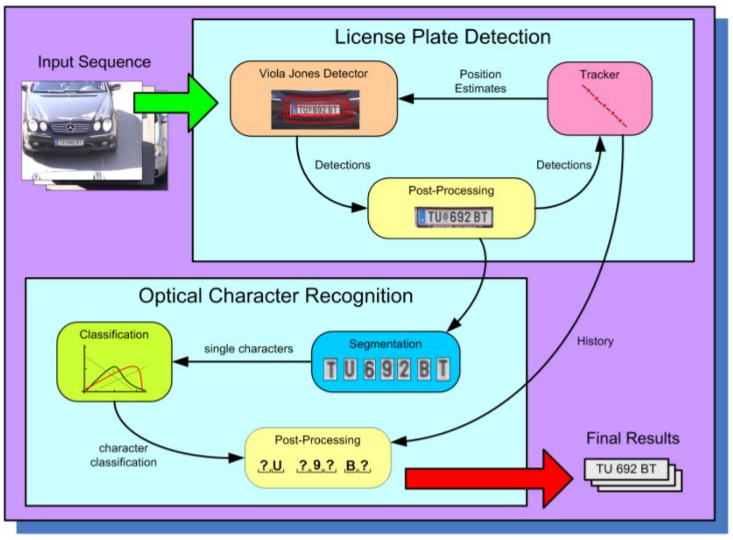


Fig.7

License Plate Character Recognition overview. After segmentation, a geometric check is performed to early discard regions unlikely being a character. The regions are subsequently preclassified to further eliminate false positives. The Support-Vector classification is subsequently used to determine the true number. All results for a single plate track are involved in a majority-voting step to finally determine the true plate number.[2]

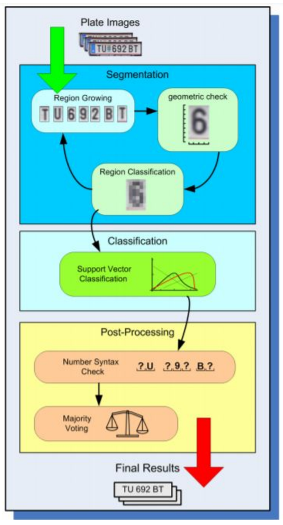


Fig.8

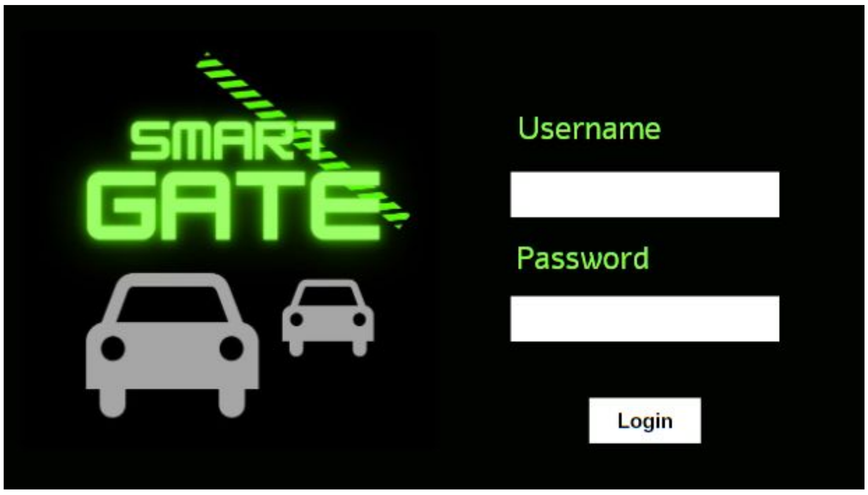
6. HUMAN INTERFACE DESIGN

6.1 Overview of User Interface

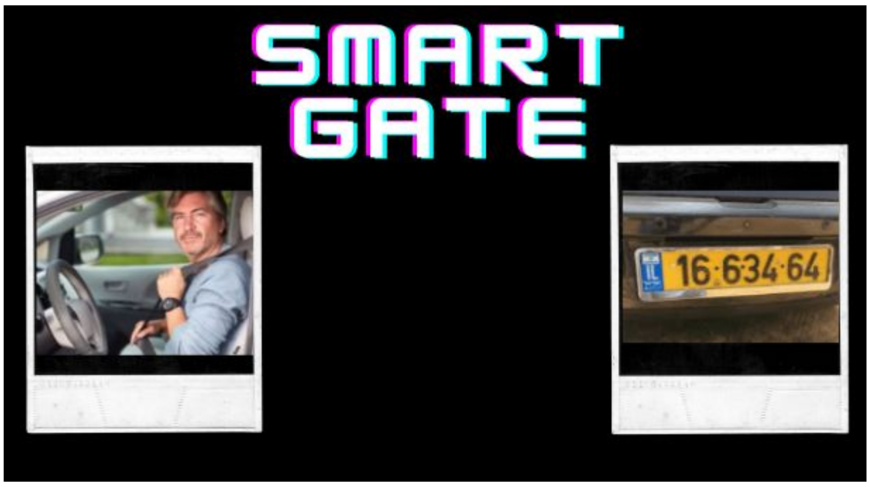
The guard (User) connects to the application using a username and password, so the application verifies the guard's information and connects it to the system. After the guard connected to the system, each time a driver enters the parking lot, the guard will receive images of the driver's face and license plate of his car and through markings in the app (red or green) the guard will know whether the system was able to successfully identify the driver details or not. If the verification fails, the guard will have to verify the driver's information himself and after this, the parking gate will open automatically.

6.2 Screen Images

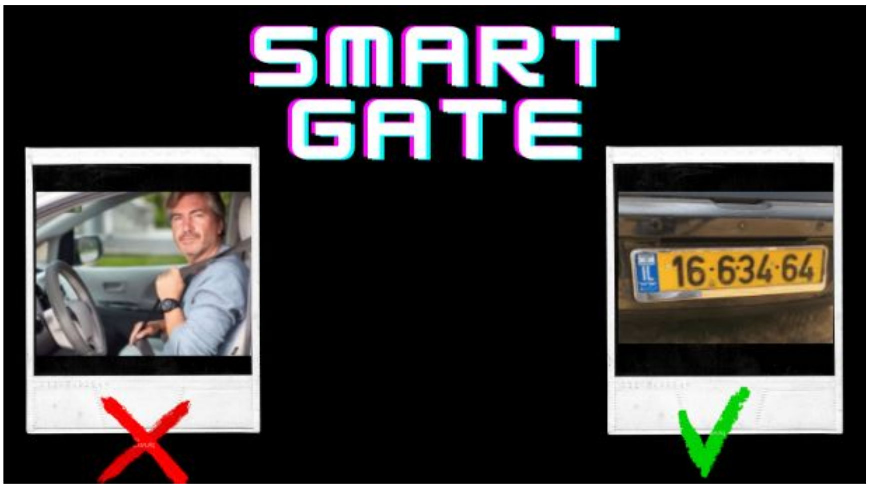
Login screen:
User - receives the username and password from the owner.



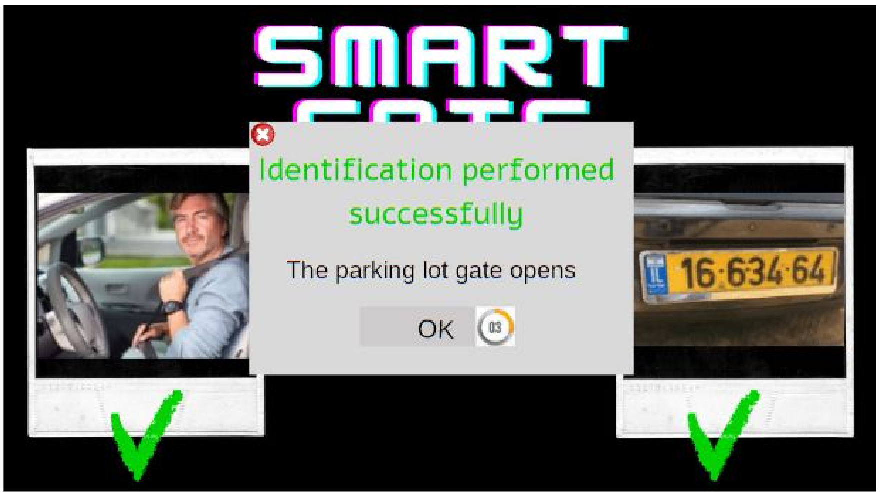
<u>Main screen:</u>
The user sees the driver's images taken in real-time (face and license plate).



<u>Main screen after an identification attempt has been made:</u>
The user sees if the identification was performed successfully on each of the images.

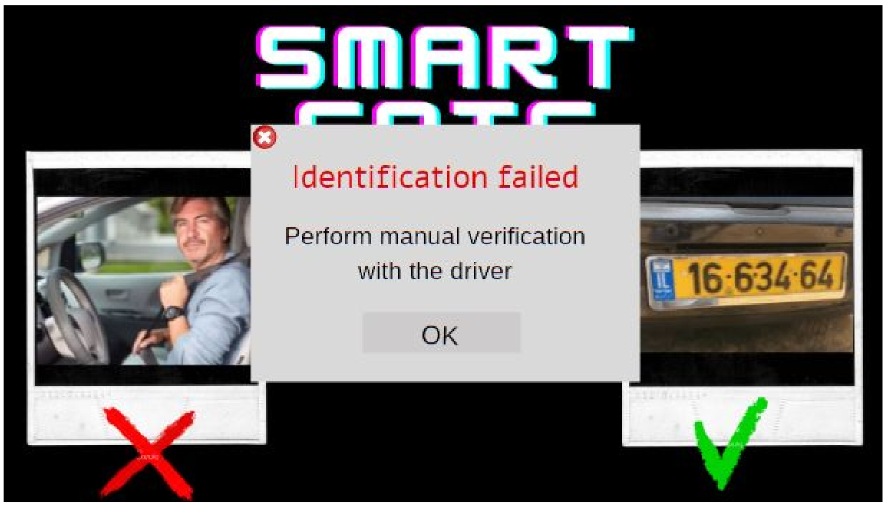


<u>Main screen - detection was performed successfully</u>:
A message appears to the user that the identification was performed successfully and the gate in the parking lot opens. The user does not have to do anything.


Identification performed successfully

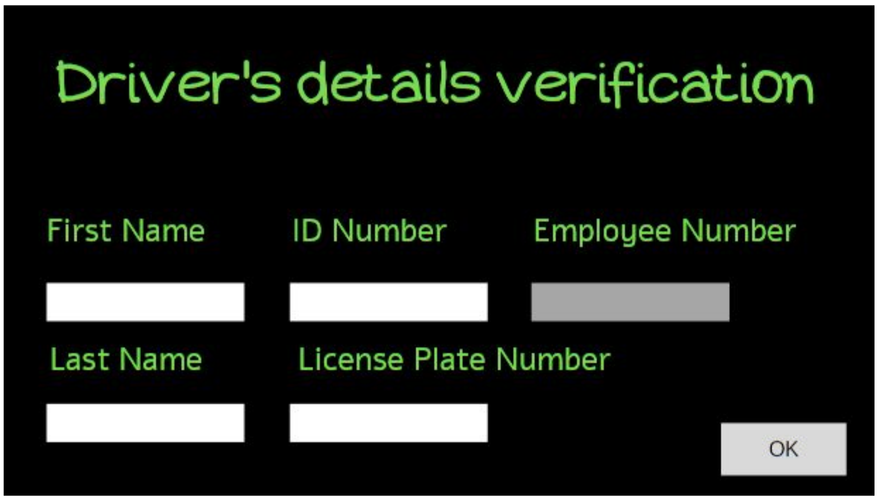
The parking lot gate opens

<u>Main screen - detection failed:</u>
The user receives a message that the detection failed.
The user must verify details manually via the following screen.

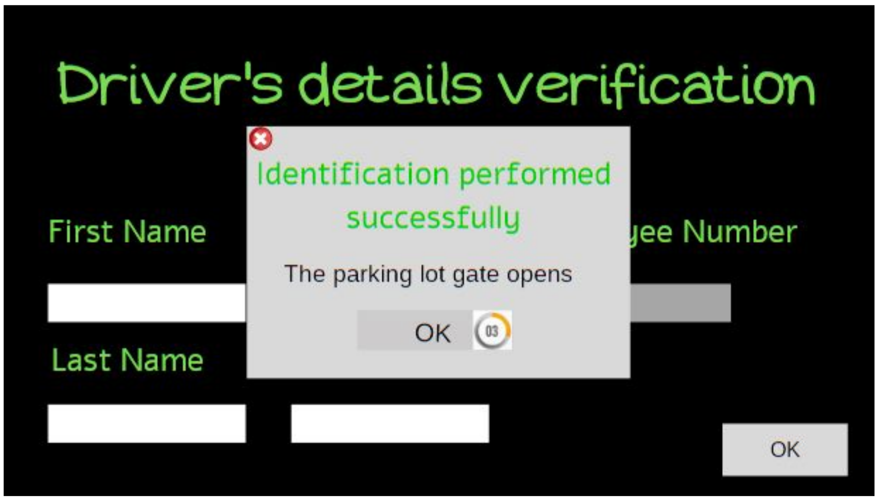

Identification failed

Perform manual verification with the driver

Driver's details verification screen:
If the identification fails on the main screen, the guard will manually verify the driver information on this screen.



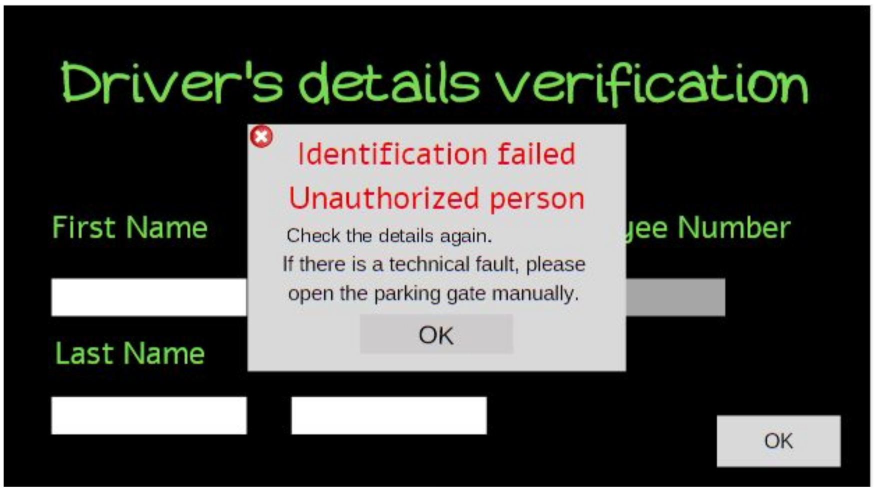
Driver's details verification screen - detection was performed successfully:
A message appears to the user that the identification was performed successfully and the gate in the parking lot opens. The user does not have to do anything.



Identification performed successfully

The parking lot gate opens

<u>Driver's details verification screen - detection failed:</u>
The user receives a message that the detection failed.
The user must re-check the information they typed.
In the event of a technical malfunction and the driver is indeed authorized to enter the parking lot, the user will open the gate manually.



Identification failed
Unauthorized person
Check the details again.
If there is a technical fault, please open the parking gate manually.
OK

<u>6.3 Screen Objects and Actions</u>

- <u>Login screen:</u>
    - Username - A unique username for each guard on duty. Consists of letters and numbers.
    - Password - Password can be ranged from 4 to 20 letters.

- <u>Main screen:</u>
    - Two squares appear on the screen - A square for license plate recognition and a square for face recognition. After taking the photos in the parking lot in real-time, the system will display the photos in squares. If both images are detected and displayed on the screen a message will be displayed to the guard that the identification has been successfully completed, the system will open the gate in the parking lot automatically. If there is a problem, a message will be displayed to the guard who must approach the driver to verify identification information.

- <u>Driver's details verification screen:</u>
    - Identification information of the driver that the guard types - Full Name, ID Number, License plate number (In some cases also employee number).
    - Opening of the parking gate: the gate opens automatically after the manual verification has been completed successfully.

7. References

[1] Zahraa Qasem Jaber, Mohammed Issam Younis College of Engineering, University of Baghdad Al-Jadriyah, Baghdad, Iraq - Design and Implementation of Real Time Face Recognition System (RTFRS), International Journal of Computer Applications (0975 – 8887) Volume 94 – No 12, May 2014.

[2] Clemens Arth, Florian Limberger, Horst Bischof - Real-Time License Plate Recognition on an Embedded DSP-Platform. Graz University of Technology Institute for Computer Graphics and Vision Inffeldgasse 16/2, 8010 Graz, Austria