Ministry of Higher Education and Scientific Research
Manouba University

National School for Computer Sciences

Immersion in Company Internship Report

Subject : Vehicle Recognition System In Tunisia

Carried Out By :
Ghassene Tanabene

Host company : Chambi Eagle Technology
Company Supervisor : Mr. Mohamed Ali Ben Amor
Adresse : Tazmour Street, 6 - Menzah 1, Tunis
TEL : +216 29 988 828
eMail : adnene.zayene@chambieagle.com

Academic Year :
2019-2020

Contents

List of Figures

List of Tables

List of acronyms

ALPR *Automatic Licence Plate Recognition*

LPR *Licence Plate Recognition*

LP *Licence Plate*

AI *Artificial Intelligence*

CV *Computer Vision*

ML *Machine Learning*

DL *Deep Learning*

ANNs *Artificial Neural Networks*

NN *Neural Networks*

CNN *Convolutional Neural Network*

YOLO *You Only Look Once*

FPS *Frames Per Second*

R-CNN Region with Convolutional Neural Network

SSD Single Shot MultiBox Detector

SVM Support Vector Machine

SIFT Scale-Invariant Feature Transform

HOG Histogram of Oriented Gradients

OCR *Optical Character Recognition*

MVP *Model View Presenter*

MVC *Model View Controler*

API *Application Programming Interface*

UML *Unified Modeling Language*

UI *User Interface*

GUI *Graphical User Interface*

General introduction

These days, everything is tending to shift towards automation. Consequently, the use of automated systems based on artificial intelligence to solve diverse issues is becoming a necessity in the twenty-first century in view of their ability to make very powerful decisions in real time and to solve new problems by learning through the previous knowledge. Added to that, they could ensure unbelievable accuracy and low error rate compared to humans.

For these reasons, since an autonomous information system does not have meaning without data, the problems dealing with massive information are efficiently resolved with the artificial intelligence solutions. Among them, vehicle recognition through licence plate is considered as a very interesting and challenging area of research because of its importance for a broad range of commercial applications. Actually, LPR systems have several applications such as traffic flow analysis, digital security surveillance, parking violation enforcement, and finding stolen cars...

Considering that licence plates are the identifiers of the means of transport, their detection and recognition is a complex challenge due to a variety of factors which can be split into two different categories. In the first hand, we find technical factors such as as various LP numbering systems, sizes, forms, types, and languages. In the second hand, we have also other factors related to the quality of the image and the environmental conditions, especially bad lighting conditions, blurred pictures, orientation, many camera angles, noisy background etc...

In this context, we are interested in our project to the Tunisian licence plate case in which we take up the challenge to improve the autonomous information systems in our country with the deep learning approach, where the LPR process will be divided into three distinguished stages: LP detection and extraction, LP segmentation, and LP recognition.

In this report, we give a presentation of the work done to develop our application « **Hawk Eye Tunisia** » according to the following plan.

In the first chapter entitled «Overview», we will give a brief presentation of the host company **Chambi Eagle Technology** and the general context of the project.
Afterwards, we will describe the most important concepts and fundamental notions, accompanied by a brief analysis of existing solutions, that lead us to accomplish our work.
Moving to the third chapter, we reveal the requirements and specifications of our application, whether functional or non-functional needs, and we present the global design of the project.
Finally, the fourth and last chapter provides the different technologies used to build our solution then details the results of the implementation and test stages.

In the end, we will conclude our work by giving prospects and improvements for our solution.

Chapter 1

Overview

In this first chapter, we provide an overview of the work accomplished throughout the internship.

1.1 General context

1.1.1 Context of the internship

This project is about developping a vehicle recognition system in Tunisia. It demonstrates the outcome of the immersion in company internship for second year students of the National School for Computer Sciences.

1.1.2 Presentation of the host company

Chambi Eagle Technology [1] is a Tunisian company specialized in delivering complete, tailored and innovative solutions to clients who would like to have an outstanding digital transformation experience. Because the digital transformation is more than just buying technology, it is deploying a full change management project that involves people, assets, and business processes.

Thanks to its versatile team of experts, Chambi Eagle guarantees to its local and international clients high added value services such as digital transformation consultancy, web and mobile development and 24/7 IT managed services.

1.2 Context of the project

1.2.1 Problem setting

The number of vehicles in the world is increasing rapidly every year, particularly in Tunisia. Therefore, several difficulties will be found to discover the vehicles identities when there is some problems such as parking violation enforcement, highway toll payment and also finding stolen cars. Added to that, it is impossible sometimes to identify the vehicle's owner who escaped from authorities after causing an accident. This fact leads to develop an autonomous licence plate recognition system to identify vehicles and facilitate traffic management which is a popular and active research topic in the field of computer vision, image processing and intelligent transport systems.

1.2.2 Main objective

This project aims to solve the problem of vehicle identification in Tunisia using Deep Learning models. To do that, our mission is to generate methods based on computer vision and image processing that can accomplish the following tasks :

- Detecting the presence of the vehicle's licence plate.
- Extracting the text written on the licence plate.
- Recognition of licence plate.

Afterwards, an application will be developed to include these services.

1.2.3 Project management methodology

In order to increase product quality and speed up the software development process, we adopted the popular agile approach as a process model and exactly, the scrum methodology [2].

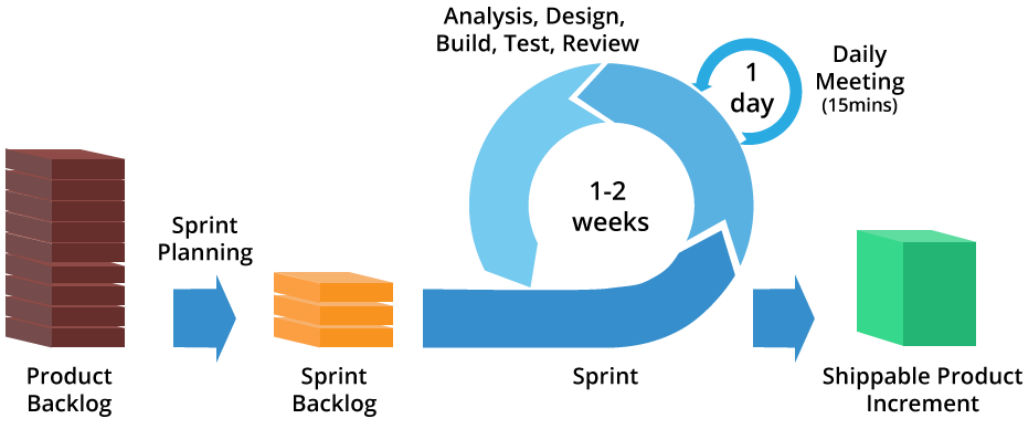

Figure 1.1: Scrum Process

As illustrated in the above figure, Scrum [3] refers to a framework that focus on meetings, roles and tools that work together supervised by the **Scrum Master** [4] to help development teams to better structure and manage their workload. In short, the product owner will draw up a list of work needed, ordered by priority called **Product Backlog**. After that, a **Sprint Planning** building on the product backlog will be fixed where teams begin with the highest priority items and determine how to accomplish this goal. Actually, the **Sprint Backlog** is a specific list of specific tasks pulled from the product backlog It will be divided into many sprints. A **Sprint** is a short period of time, usually from two to four weeks aims to complete objectives, with regular daily meetings to ensure that everything is progressing as required.
The biggest advantage of the agile scrum approach is its flexibility which increase the productivity.

Conclusion

After presenting the problem setting and the main objective of our project, we will explain in the next chapter the fundamental concepts that we will use in our work.

Chapter 2

Preliminary Study

In order to understand our system's functionalities, we will discuss in this second chapter the fundamental keywords and knowledge required with a view to approaching such a problem. Then, we will present our proposed solution.

2.1 General Concepts

2.1.1 Artificial Intelligence

Artificial intelligence (AI) [5] is a large branch of computer science dedicated to creating intelligent machinery capable of carrying out tasks usually requiring human understanding. It concerns the intelligent machines' ability to perform tasks normally requiring human intelligence, including decision-making, speech recognition, computer vision, etc.

2.1.2 Computer Vision

The computer vision [6] is a sub-field of AI that uses machines to understand digital images and videos. It aims to automate activities that can be performed by human vision. This includes strategies and techniques to collect, process, interpret, and understand digital images, and to extract information from the real world in order to generate information.

2.1.3 Object Recognition

The object recognition [7] involves finding and identifying objects present in an image or video.

- ✓ **Image Classification :** Algorithms seek to assign an image to one of a number of different categories (type or classes) such as car, cat, human, etc.
- ✓ **Single-Object Localization :**

  Algorithms generate a list of object categories present in the image. In fact, they produce an axis-aligned bounding box that indicate the scale and also the position of one instance of one object category.
- ✓ **Object Detection :** Algorithms that combine these two previous tasks. They locate the presence of objects in an image with a bounding box and their types or classes.
- ✓ **Object Segmentation :** The segmentation of objects is another extension of the CV, often called the "object instance segmentation", where the instances of known objects are indicated by highlighting the object's pixels in place of the coarse bounding box.

2.1.4 Machine Learning

The word ML refers to the ability of IT systems to independently find a way to solve problems. “Machine learning is the science (and art) of programming computers so they can learn from data." - Aurélien Géron.

2.1.5 Deep Learning

The Deep Learning [8] is a sub-field of Machine Learning covering the development of algorithms inspired by human brain neuron system and known as artificial neural networks.

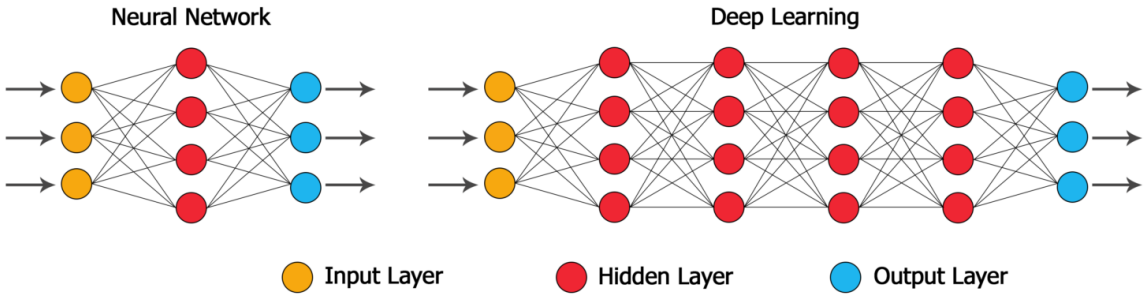

Figure 2.1: Deep Learning Layers

2.1.6 Neural Network

Artificial Neural Networks (ANNs), or simply called Neural Networks (NNs), [9] are a number of algorithms that replicate the functioning of a human brain to identify relationships between vast amounts of data. The NN itself consists of many core processing units called **Neurons** and grouped into several layers. As shown in the Figure 2.2, **the input, hidden and output layers** are interconnected with "Weighted Connections" [10].

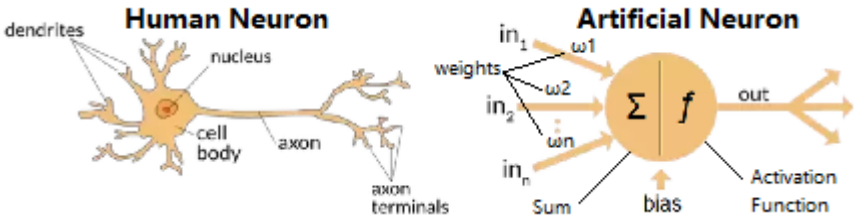

Figure 2.2: Similarity between human and artificial neurons

2.1.7 Convolutional Neural Network (CNN or ConvNet)

CNN [11] is a Deep Learning algorithm that takes an input image, attribute learnable weights and biases to different image’s patterns (aspects/objects) and have the ablility to differentiate one from the other. With its less pre-processing requirement, a ConvNet is favored over other NN architectures where it can replace the primitive methods filters called hand-engineered by learning these filters/characteristics. The CNN architecture [12], as figure 2.3 shows, contains an input layer, output layer and hidden layers presenting three-layer types : Convolutional layer, Pooling layer, and Fully connected layer. The **convolutional layer** includes a number of filters which have parameters to be learned. **The pooling layer** decreases the size of a large

picture so that the average presence of a feature is summarised. After dividing the image into various features and separately analyzing them, this process's outcome feeds into a **fully connected layer** that aims for the final classification decision.

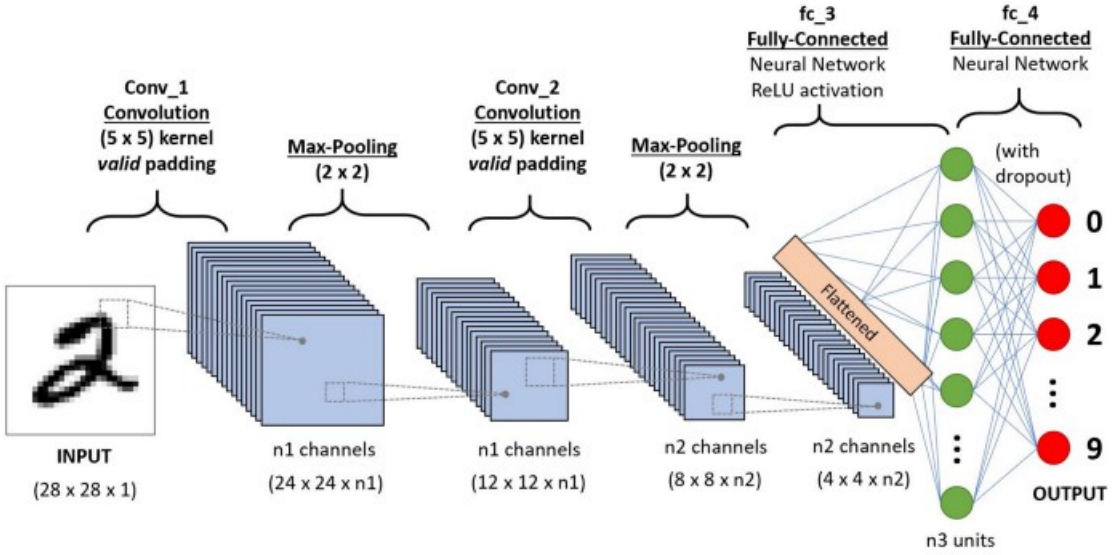

Figure 2.3: Convolutional Neural Network Architecture

2.1.8 Transfer learning

Transfer learning consists of reusing a pretrained model that has been learned to solve a problem and apply it to another problem. This technique, based on saving knowledge for reuse and optimization, speeds up training and increases deep learning models performance.

2.1.9 YOLO - You Only Look Once

You Only Look Once or **YOLO** [13] is an object detection algorithm using Deep Learning and Convolutional Neural Network (CNN). It differs from other algorithms since, as its name indicates, it only needs to «see» each frame once. This allows YOLO to be one of the fastest detection algorithms. Thanks to this speed, YOLO can detect objects in real time (up to 45 FPS). To perform the detection, the image is divided into a grid of SxS. Each of the cells will predict N possible bounding boxes and the level of certainty (or probability) of each of them. The majority of these boxes will have a very low probability, which is why the algorithm proceeds to remove boxes that are below a certain minimum probability threshold. The figure 2.4 shows an illustration of YOLO dividing a source figure into grids and assigning bounding and class boxes to each grid.

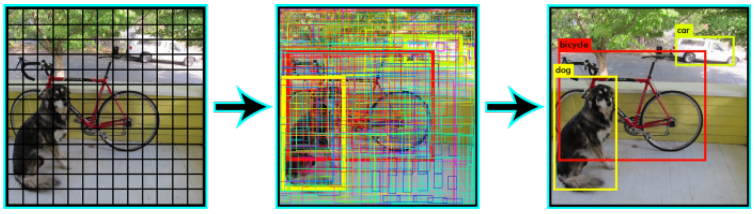

Figure 2.4: YOLO object detection

2.1.10 OCR - Optical Character Recognition

The **OCR** is a technology to differentiate handwritten or printed text characters from physical documents in digital photos, for example the scanned paper document. The OCR fundamental method consists of review of the document text and conversion of characters into code.

2.2 Study of the solution

2.2.1 An overview of previous work

After doing a research to discover the existing solutions, we find that for computer vision, there are two approaches based either on ML such as SVM, SIFT, HOG .. or DL such as Region Proposals (R-CNN,Fast-RCNN,Faster-RCNN, Mask-RCNN..), SSD, YOLO, Retina-Net, etc.. to detect the LP and identify its characters. We find that some solutions are implemented with tensorflow and keras and others with pytorch. However, there many solutions based only on OCR and image processing where the character recognition is done by comparing pixel by pixel each character to many text fonts and styles. The table [14] in figure 2.5 shows the detection rate, recognition rate and computational time of three different algorithms for LP detection and segmentation.

Sr. No.	Methods	Detection Rate	Recognition Rate	Computational Time (s)
1	Morphological	95.5%	90.75%	1.1578
2	Histogram Based	97.75%	-	2.7819
3	Mathematical Morphology	96.50%	85.50%	3.5182

Figure 2.5: Comparison of 3 algorithms

2.2.2 Proposed solution

The main goal is to develop an application that allows users to identify vehicles automatically. Our solution will be based on computer vision with deep learning approach where we have three essential parts : LP detection, LP segmentation and characters recognition. The image pre-processing stage includes resizing, noise removal and edge enhancement. The figure 2.6 shows the vehicle recognition process where we used two different DL models : the first is for LP detection and the second is for characters classification.

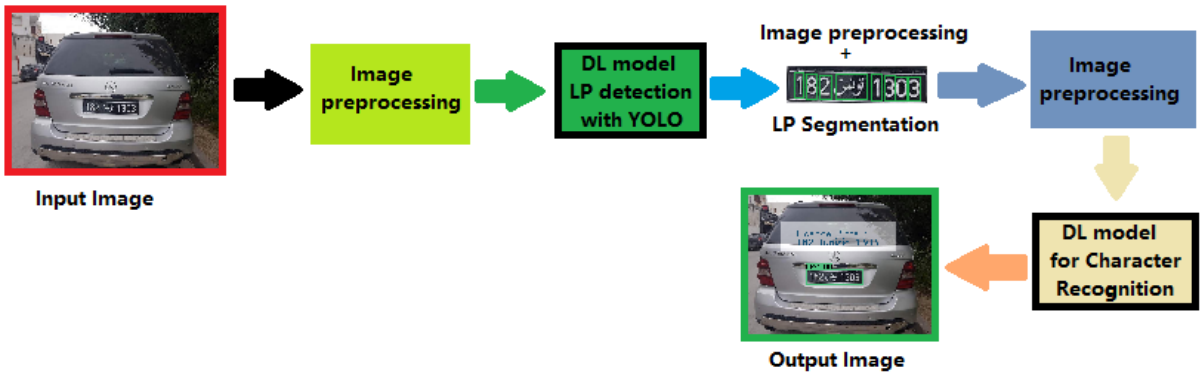

Figure 2.6: The vehicle recognition process of Hawk EYE Tunisia

Chapter 3

Requirements Analysis and Design

In the following chapter, we will describe the system requirements and specification. Then, we will briefly touch on the design of our application : the «Hawk Eye Tunisia», through a global and detailed architecture overview described with UML diagrams.

3.1 Requirements Analysis and Specification

To understand our system's specifications, we will firstly highlight in this section the main requirements in the system's use case diagram. Additionally, we will explain the vehicle recognition scenario in the sequence diagram.

3.1.1 Requirements Analysis

3.1.1.1 Actors

In our system, we have one actor : the user of «Hawk Eye Tunisia».

3.1.1.2 Functional Requirements

- ✓ Vehicle's licence plate detection
- ✓ Vehicle recognition through camera
- ✓ Vehicle recognition with an image from the user's machine
- ✓ Verification of the legal status of the vehicle and getting details about it
- ✓ User authentication

3.1.1.3 Non-Functional Requirements

- ✓ Security
- ✓ Maintainability
- ✓ Reliability
- ✓ Enrich the database with new vehicles photos to improve the deep learning models

3.1.2 Requirements Specification

3.1.3 Use Case Diagram

The figure 3.1 below illustrates the use case diagram of our system.

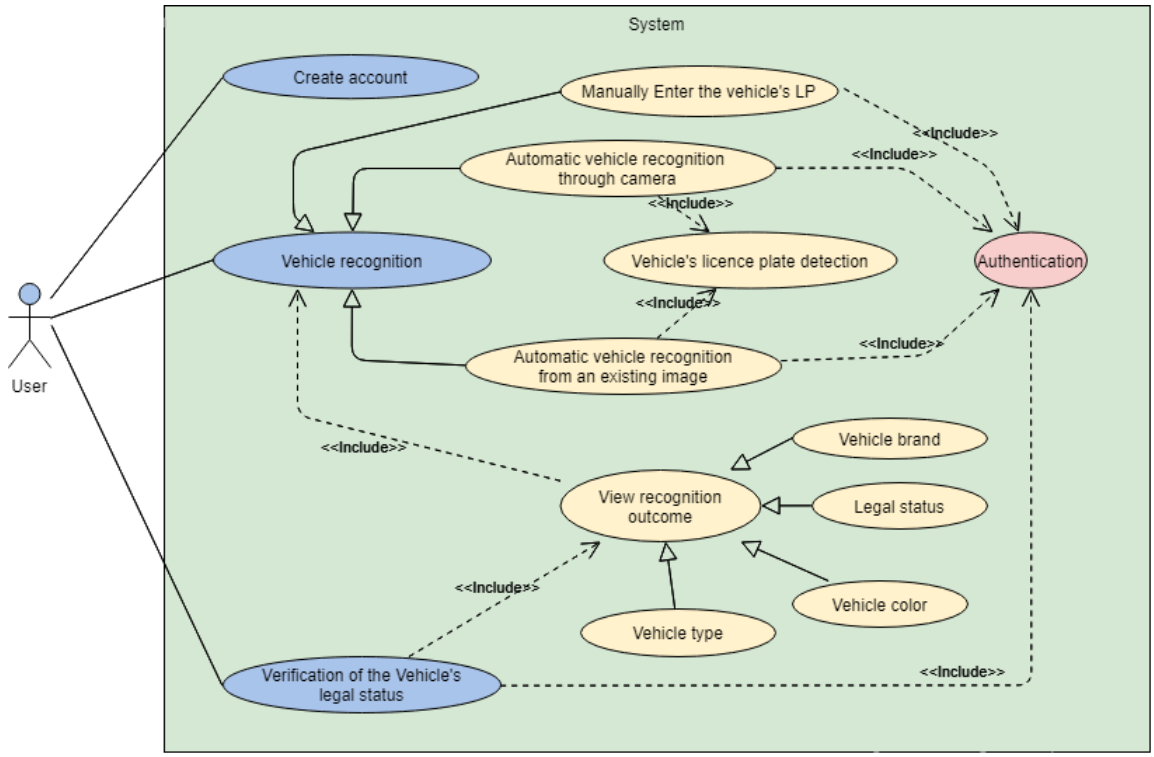

Figure 3.1: Use Case Diagram

3.1.4 System Sequence Diagram

The figure 3.2 shows the User's interactions with the system.

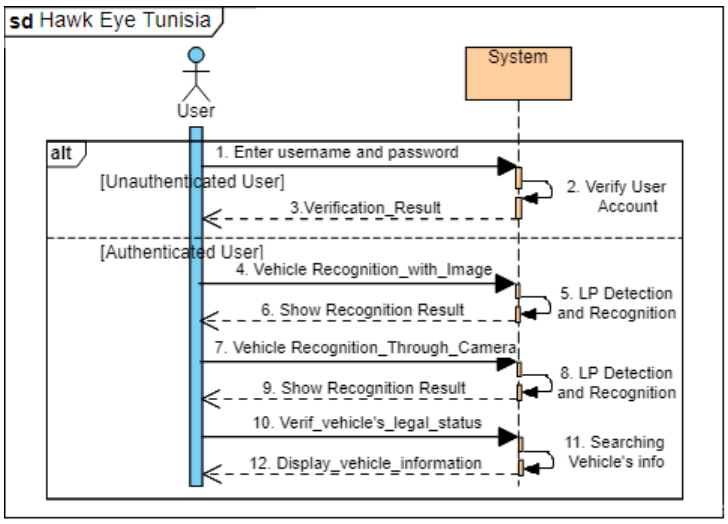

Figure 3.2: System Sequence Diagram

3.2 Design

3.2.1 Global Design

3.2.1.1 Physical Architecture

To implement our application, we opt for the 3-tier architecture [15]. It is used mostly by a large number of applications that have data processing, requiring a powerful machine. The figure 3.3 shows that the application is divided into three layers.

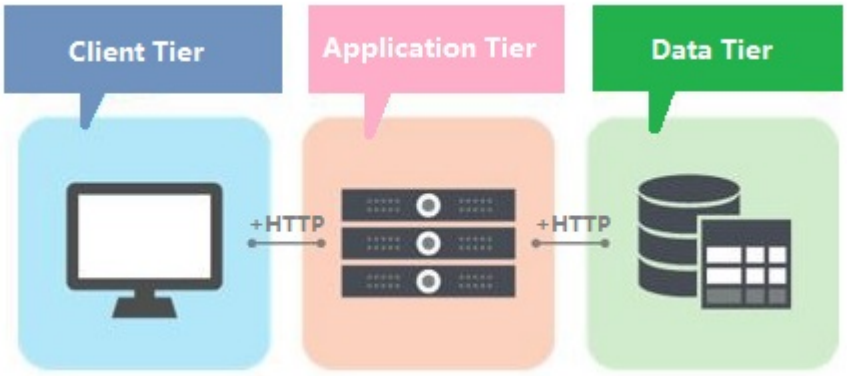

Figure 3.3: Three-tier architecture

- **The Client Tier:** The Client, also called the presentation layer, represents the web browser in most cases. In other instances including our project, the client tier represents the UI (User Interface) for the Windows Forms application «Hawk Eye Tunisia».
- **The Application Tier:** Also called logic tier or business logic layer, the application tier is the middle layer that guarantees coordination and communication between the two other layers. On this server, all work and processing are performed.
- **The Data Tier:** Typically, the database tier is a server used to store and manage databases such as Oracle SQL sever or MySQL, etc. It is used to store and provide authorized users with data access.

3.2.1.2 Logical Architecture

In order to build our application, we adopted the MVP model shown in the figure below.

Model – View – Presenter (**MVP**) [16] is a derivation of the popular architectural pattern Model – View – Controller (**MVC**) which is mostly used for the development of user interfaces. In the MVP pattern, the Controller is replaced by the Presenter that assumes the the "middle-man" functionality between Model and View.

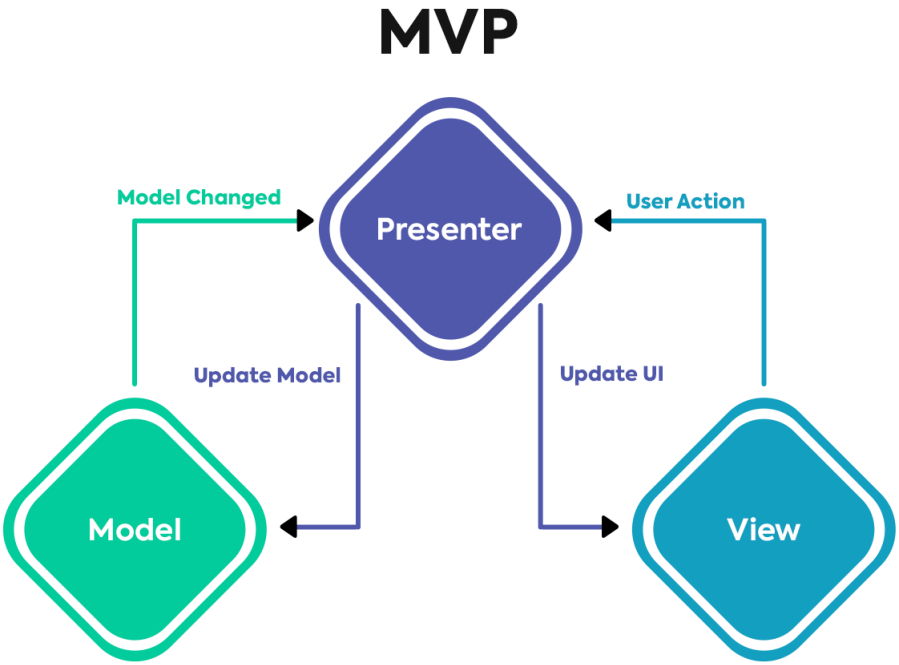

Figure 3.4: Model View Presenter Architecture pattern

- **Model :** It manages our application's data part. The roles of the model [17] include the use of APIs, data caching, database management and so on.
- **View :** It is responsible for the presentation of the application's views using the appropriate data as instructed by the Presenter.
- **Presenter :** It seems like a bridge connecting the view and the model. After formatting them, the Presenter retrieves and returns data from the Model to the View. But unlike the regular MVC, it also defines what happens when you interact with the View.

3.2.2 Detailed Design

3.2.2.1 Sequence Diagram

In the figure 3.5, we illustrate the scenario of vehicle recognition through camera.

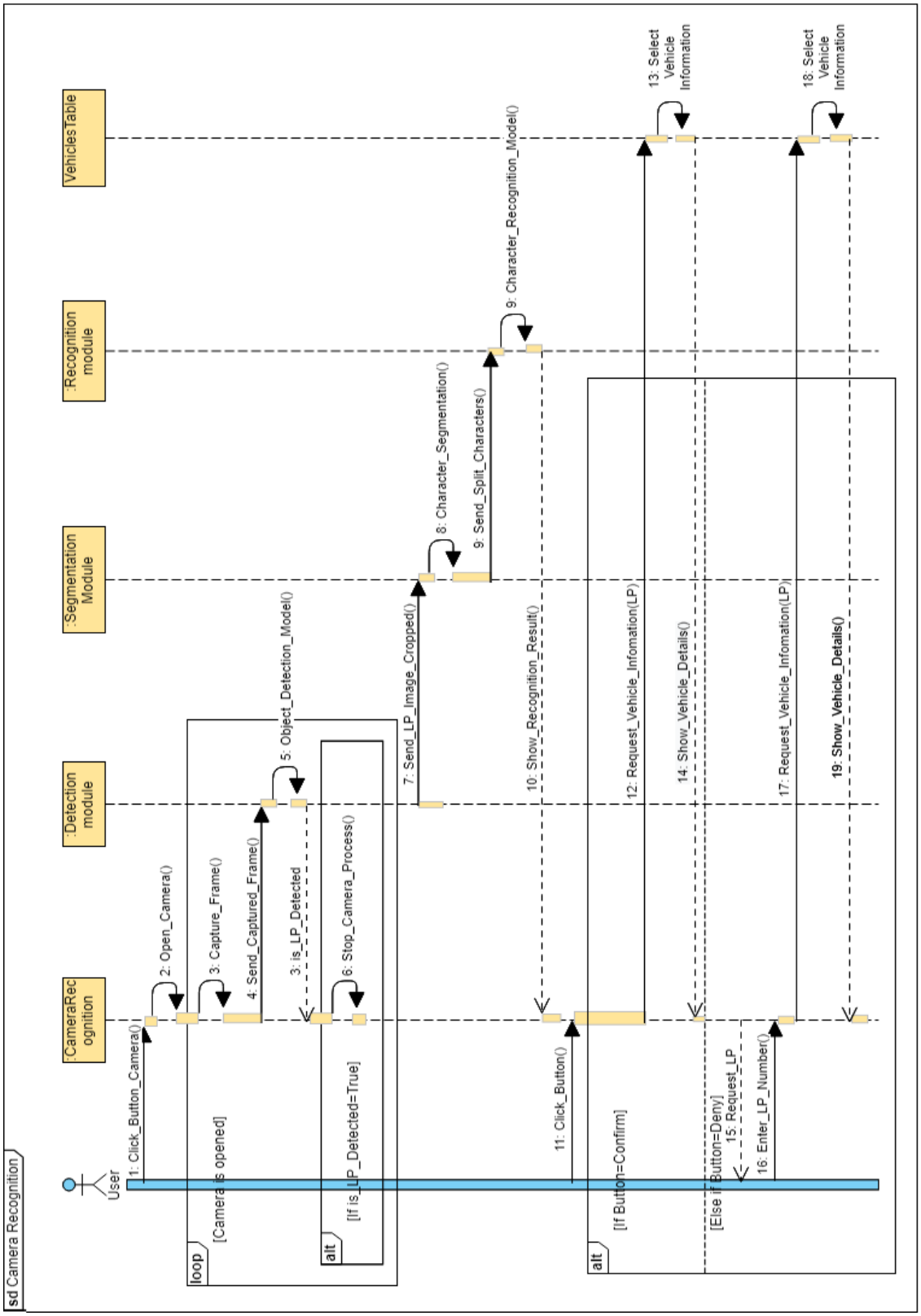

Figure 3.5: Sequence Diagram of Vehicle Recognition Through Camera

Chapter 4

Achievements

As for this last chapter, we will highlight the work environment, alongside the technologies used throughout the development. The chapter also includes several screenshots that describe the progress and the results of the project.

4.1 Work Environment

4.1.1 Hardware Environment

To achieve our goal, we used a personal computer with the characteristics shown in Table 4.1.

	Computer
Brand	Asus
Processor	Intel Core i5-7200U CPU
Memory	8 GB of RAM
Hard Disk	1 TB
Operating Systems	Window 10 and Ubuntu 20.04

Table 4.1: Hardware Environment

4.1.2 Software Environment

The following table 4.2 resumes the software environment used in our project.

Software	**Category**	**Usage**
Visual Studio 2019	Integrated development environment	Developing .NET application Deploying Azure Functions App
Jupyter Notebook	Web-based interactive computational environment	Data Cleaning, Model Training, Visualization
Python IDLE	Integrated DeveLopment Environment	
Microsoft Azure DevOps	Cloud services provider	Project management, collaboration & deployment
Overleaf	Online Latex editor	Report redaction
visual-paradigm.com	Online diagram tool	Drawing diagrams

Table 4.2: Software Environment

4.1.3 Technical Choices

The implementation of our ALPR system requires several technologies as table 4.3 shows.

Technology	Usage
Python	Programming language for the vehicle recognition
C#	Programming language for the desktop app
Windows Forms for .NET Core	The GUI included in the framework .Net

Main Libraries	Usage
Tensoflow	ML and DL Library
OpenCV	Computer Vision Library
Matplotlib	Visualizations in Python
Numpy	Multi-dimensional arrays & matrices, Calculations
Pickle	Saving in binary format file

Table 4.3: Technical Choices

4.2 Achieved Work

4.2.1 Computer Vision with Deep Learning Approach

In our project, we used two Deep Learning models based on CNNs : one for LP detection and another for character recognition.

4.2.1.1 Preparing data for the character recognition model

In the first step, we prepared data for the deep learning model used to recognize the licence plate's characters. This step consists of gathering images of Tunisian vehicles, then extracting their licence plates and finally cutting the numbers and words from the plates manually by resizing them in squares of 28x28 pixels after applying the image processing steps. We prepared 11 classes for the Tunisian license plate use case where the first 10 classes contains numbers from 0 to 9 and the last class refer to the word "Tunisia" in arabic. We mention that in the following report, we focused on the Tunisian use case only. The case of foreign licence plates is also treated by adding 26 classes containing the English alphabet images, but it is not included in the requirement chapter.

Figure 4.1: Preparing data for the character recognition model

4.2.1.2 Licence plate detection with YOLO

In order to detect licence we will use **YOLOv3** (You Only Look Once) deep learning object detection architecture based on CNN [18]. First of all, we prepared a dataset composed of Tunisian vehicles images, and for each one, we make an xml file that will be changed after that to text file containing LP coordinates compatible with Darknet config file input. We prepare this data using a desktop application called LabelImg. To have more accuracy, we apply transfer

learning to a pretrained model for licence plate detection that itself uses **Darknet** to retrain YOLO pretrained models.

Figure 4.2: Test of the licence plate detection model

4.2.1.3 Licence plate segmentation

The segmentation is an important stage of every OCR system and the key processes for ALPR systems, as all other measures depend on it. This phase is the most complicated as several factors should be taken into account such as image angle, brightness , contrast, background, blur, etc.

Figure 4.3: Characters Segmentation

To achieve this segmentation, we used the method of **histogram of pixel projection**. It consists of finding the upper and lower boundary, left and right of each character. We perform a horizontal projection to find the top and bottom positions of each character, and we do the same vertically to find its left and right limits as shown in the figure.

Figure 4.4: Histogram of pixel projection method for segmentation

4.2.1.4 Characters Recognition

- **Importing libraries**

```
Entrée [1]: # libraries used for the model building

import tensorflow as tf
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Flatten, MaxPooling2D, Dropout, Conv2D
from tensorflow.keras import optimizers
import datetime
import pickle
```

Figure 4.5: Importing libraries for the recognition model

- **Data generator**

  Since each image is made up of a set of pixels with values between 0 and 255, the goal of this part of the code is to render those values between 0 and 1 because neural networks prefer to process small values. This task can be performed with the $ImageDataGenerator$ method provided by tensorflow.keras. After defining the generators for the training and validation images, the "flow_from_directory" method loads the images from disk and resizes the images to the required dimensions (IMG_HEIGHT and IMG_WIDTH).

```
Entrée [3]: train_datagen = ImageDataGenerator(rescale=1./255, width_shift_range=0.1, height_shift_range=0.1)
train_generator = train_datagen.flow_from_directory(
        'data/train',  # this is the target directory. It contains the different classes of the characters.
        target_size=(28,28),  # all images will be resized to 28x28
        batch_size=1,
        class_mode='categorical')

validation_generator = train_datagen.flow_from_directory(
        'data/val',  # this is the target directory
        target_size=(28,28),  # all images will be resized to 28x28       batch_size=1,
        class_mode='categorical')

Found 623 images belonging to 11 classes.
Found 66 images belonging to 11 classes.
```

Figure 4.6: Data generator

- **Model defining and training**

  In the figure 4.7, we create and compile our CNN model.

```
Entrée [4]: model = Sequential()
model.add(Conv2D(32, (24,24), input_shape=(28, 28, 3), activation='relu', padding='same'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.4))
model.add(Flatten())
model.add(Dense(128, activation='relu'))
model.add(Dense(11, activation='softmax'))

model.compile(loss='categorical_crossentropy', optimizer=optimizers.Adam(lr=0.00001), metrics=['accuracy'])
```

Figure 4.7: Model defining

After preparing our data and the structure of our model, we started the training using the "fitgenerator" function which takes the training data, the validation data and the number of epochs as a parameter.

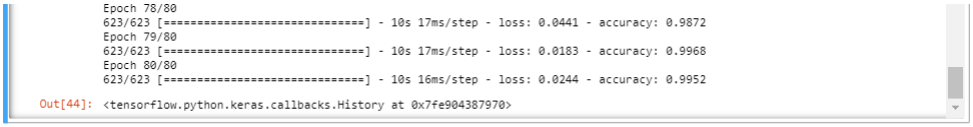

Figure 4.8: Model Training

- **Model Testing**

  In the figure 4.9, we did segmentation where characters are cropped from the LP. Then, after doing some image preprocessing, we used a function called "show_results" that load our saved model and recognize the class of each character separately.

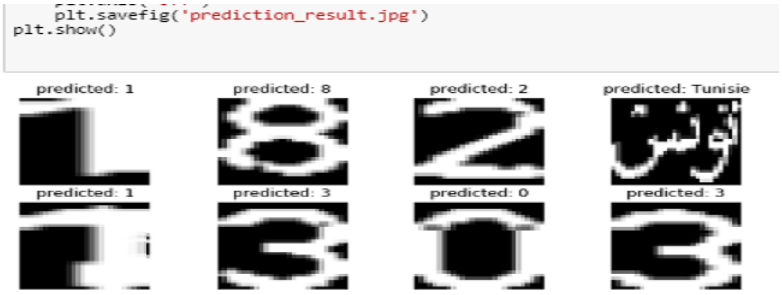

Figure 4.9: Testing the model prediction

4.2.2 Desktop Application - Hawk Eye Tunisia

As we mentioned before, Hawk Eye Tunisia is a Windows Forms application coded with the programming language C#. It will be an interface to communicate the recognition system with its users. The vehicle image will be sent to the DL saved models where the three modules of LP detection, LP segmentation and LP recognition will be run to identify the vehicle.

4.2.2.1 Authentication

To ensure the security of the vehicles data, only authenticated users can benefit from our system. The login interface, figure 4.10, allows user to enter his username and password.

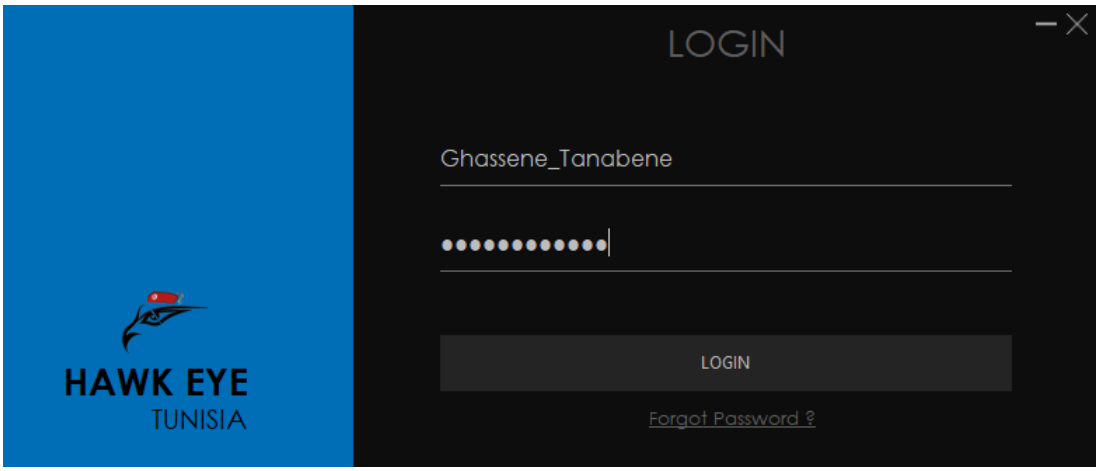

Figure 4.10: login

4.2.2.2 Vehicle recognition by loading an image from the user's machine

In the figure 4.11, the user have a "Browse" button that allows him to select an image from his machine. After selecting one, the process of recognition system will be run. Finally the outcome will be returned showing the input image with with a green box that illustrates the LP detection and the result of LP recognition.

Figure 4.11: Vehicle Recognition With Image

4.2.2.3 Vehicle recognition with camera

In the figure 4.12, the user will use the camera to detect the presence of a vehicle in front of it and to recognize its LP automatically.

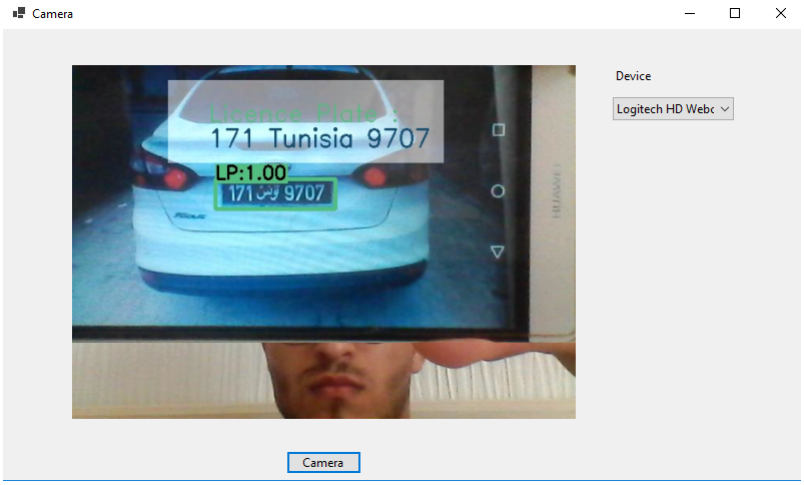

Figure 4.12: Vehicle Recognition Using Camera

Conclusion

In this final chapter, the software environment on which this project was developed was introduced and our main technical choices clarified. Finally, we have promoted pictures and user interfaces to display the significant accomplishments.

Conclusion and Perspectives

Most of the **ALPR systems** are based on image processing techniques and character recognition technologies to identify vehicles by automatically reading their licence plates. Each ALPR system consists of three different modules : **license plate detection and extraction, LP segmentation** and finally **character recognition**. The work presented in this paper aims to propose our solution, **Hawk Eye Tunisia**, based on **computer vision with the Deep Learning approach** to recognize vehicles in Tunisia.

In the first chapter, we started with presenting the project's overview, specifying the host company, giving the problem setting and the main objective and finally describing the project management methodology. Afterwards, we highlighted the most important keywords in the second chapter with a short overview of the previous work and our proposed solution. Then, we identified the functional and non-functional requirements of our system and we explained its interactions with users with UML diagrams in the design section. Finally, in the achievements chapter, we detailed the software development cycle and we presented the most important interfaces of our application.

Throughout this internship, we faced many challenges especially in the computer vision part since the ALPR is still a challenge for many AI researchers. The most difficult challenge is the segmentation module where many factors should be included to identify the characters and the word in arabic (Tunisia) and then split the licence plate. We tried many image processing ways to achieve this mission correctly, where some methods are more efficient with foreign LP due to the its forms and text styles. Actually, these methods require various advanced imagery techniques to apply them in the Tunisian use case. Despite all factors, we succeeded this segmentation challenge by applying the histogram of pixel projection method.

Nevertheless, improvements and optimization are always necessary in each project. For future researches and to improve our ALPR system, it would be interesting to retrain the model of recognition with a large number of characters' images cropped from real Tunisian LP images since we have a dataset of more than 30 000 vehicles images. In addition, it would be useful also to try the R-CNN family for the LP detection model and compare the run time with the results given by YOLO. To further push the limits, we can improve the LP segmentation by treating more difficult cases such as rotated plate, vehicle's dashboard, erased plate, etc. In addition, we can add the detection and segmentation of square Tunisian LP and other different not standard forms where the text is written in two lines or different styles.

Bibliography

[1] www.linkedin.com/company/chambi-eagle technology/about.

[2] https://vslsoft.com/competencies/project management.

[3] https://www.projectmanager.com/blog/scrum methodology.

[4] https://www.businessnewsdaily.com/4987-what-is-agile-scrum methodology.html.

[5] https://builtin.com/artificial intelligence.

[6] https://fullscale.io/blog/machine-learning-computer vision/.

[7] https://machinelearningmastery.com/object-recognition-with-deep learning/.

[8] https://alphabold.com/neural-networks-and-deep-learning-an overview/.

[9] https://www.investopedia.com/terms/n/neuralnetwork.asp.

[10] https://www.quora.com/What-is-the-difference-between-artificial-intelligence-and-neural networks.

[11] https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way 3bd2b1164a53.

[12] https://becominghuman.ai/neural-networks-and-deep-learning-cnn-vs-rnn 7710d69feebf.

[13] https://medium.com/@jonathan_hui/real-time-object-detection-with-yolo-yolov2 28b1b93e2088.

[14] https://www.ijert.org/research/comparative-study-of-license-plate-recognition IJERTV3IS031851.pdf.

[15] https://www.quora.com/What-is-the-role-of-a-web-service-in-a-three-tier architecture.

[16] https://medium.com/cr8resume/make-you-hand-dirty-with-mvp-model-view-presenter eab5b5c16e42.

[17] https://medium.com/flutterdevs/design-patterns-in-flutter-part-2-mvp e17b3be2e51b.

[18] https://arxiv.org/pdf/1804.02767.pdf.