MAKERERE UNIVERSITY

COLLEGE OF COMPUTING AND INFORMATION SCIENCES

SCHOOL OF COMPUTING AND INFORMATICS TECHNOLOGY

TRAFFIC LIGHTS COMPLIANCE AND PENALTY SYSTEM.

By

BSE 23-14

EMBEDDED SYSTEM COMBINED WITH A WEB SYSTEM.

DEPARTMENT OF NETWORKS.

SCHOOL OF COMPUTING AND INFORMATICS TECHNOLOGY

A Project Report Submitted to the School of Computing and Informatics Technology for the Study Leading to a Project in Partial Fulfillment of the Requirements for the Award of the Degree of Bachelor of Science in Software Engineering of Makerere University.

Supervisor

DR. Rashidah Kasauli

Department of Networks

School of Computing and Informatics Technology, Makerere University

rashidahk@gmail.com

June 11, 2023.

Declaration

We, group **BSSE 23-14,** hereby declare that the work presented is original and has never been submitted for an award to any university or institution of higher learning. We can confirm that where we have done consultations either from published material or the works of others, it has been attributed in this report.

#	**Names**	**Registration Number**	**Signature**
1	NAKIBINGE BONNY	1900712552	
2	KATEMBEKO CHRISTOPHER	1900724692	
3	OTIM DERRICK MURUNGI	1900712613	
4	NSUBUGA ALPHA FRANCIS	1900708652	

Approval

This project report titled TRAFFIC LIGHTS COMPLIANCE AND PENALTY SYSTEM has been submitted for examination with my approval as the supervisor of group BSE 23- 14

Dr./Mr/Mrs.....

Department of Networks

School of Computing and Informatics Technology;

College of Computing and Information Sciences,

Makerere University

Signature: ..... Date: .....

Supervisor

Dedication

We would like to dedicate this report to all the individuals who have inspired and motivated us throughout this project and for assessment.

To our families, whose unwavering love and support have been our foundation towards aid in the financial support, thank you for always believing in us and encouraging us to pursue our goals.

To our friends, who have been there to lend a helping hand, offer valuable insights, and provide a much-needed dose of laughter during challenging times, thank you for being by our side.

To our project supervisors, DR Mary Nsabagwa and Dr Rashida Kasauli, their guidance, expertise, and continuous encouragement have been invaluable. Your mentorship has shaped our understanding and skills, and we are grateful for the opportunity to work under your guidance.

To all the individuals who have contributed their time, expertise, and feedback to this project, thank you for your valuable contributions. Your input has played a significant role in shaping the outcomes of this work.

Lastly, I would like to dedicate this report to all the individuals who will benefit from the Traffic Light Compliance and Penalty System. May it contribute to enhancing road safety and promoting adherence to traffic regulations.

Acknowledgements

We would like to express our sincere gratitude to all individuals who have contributed to the successful completion of this report on the Traffic Light Compliance and Penalty System.

First and foremost, we would like to extend our deepest appreciation to our project supervisors, Dr. Rashida Kasauli and Dr Mary Nsabagwa, for their guidance, support, and valuable insights throughout the development and documentation process. Their expertise and encouragement have been instrumental in shaping this project.

Furthermore, we would like to acknowledge the invaluable assistance provided by Makerere University towards giving us the opportunity to gain advance knowledge in the field of software development and engineering. Their provision of resources, technical support, and access to necessary infrastructure have been crucial in the successful execution of this project.

We would also like to extend our appreciation to our friends and family for their unwavering support and encouragement throughout this endeavor.

Lastly, we are grateful to all the individuals who have contributed their time, feedback, and expertise during the testing and evaluation phases of the project.

Without the collective efforts and support of the aforementioned individuals and organizations, this project would not have been possible. Thank you all for your contributions and commitment to this project.

Abstract

The Traffic Light Compliance and Penalty System (TLCPS) is a technological solution designed to detect and capture vehicles violating traffic lights, issue penalties and notify the drivers thorough SMS notifications. This report presents an overview of the system, including its objectives, design, implementation, testing, performance, maintenance, and support requirements.

The system utilizes a combination of traffic lights through led light simulation, pi cameras, ultra-sonic sensor, and a Raspberry Pi control system to monitor and capture traffic violations. Violations are detected through analyzing the motion vehicles at red lights. Once the vehicle is analyzed, the license plate is identified and analyzed to obtain the text license plate, penalty notifications are generated and sent to the registered vehicle owners via SMS using the Africa's Talking API.

During the development phase, comprehensive testing was conducted to ensure the system's functionality, accuracy, and reliability. Test scenarios were defined to simulate traffic violations, and the system's ability to detect and capture violating vehicles was validated. Evidence capturing, penalty issuance, and SMS delivery were also tested to ensure their proper functioning.

To maintain the system's performance, regular service and maintenance are essential. Documentation of service and support, including future updates, problem solutions, requested modifications, and support to clients, is necessary for efficient maintenance and customer satisfaction. Any anomalous operating conditions, both in the computer system and used instruments, should be addressed through precautionary steps and proper testing.

Performance requirements, such as the maximum time taken before providing output, should be met to ensure efficient operation. Support for clients should include prompt response to inquiries, troubleshooting assistance, and guidance on system usage.

In terms of future improvements, incorporating changes may be necessary to enhance the system's functionality or address evolving traffic regulations. Upgrades should be carefully planned and executed to ensure a seamless transition from the old system to the new one. Data transfer from the old system's SQL database to the new system should be carried out systematically on the cloud to prevent data loss or inconsistencies.

In conclusion, the Traffic Light Compliance and Penalty System provides an effective solution for promoting traffic safety and enforcing compliance with traffic lights regulations. With proper maintenance, support, and continuous improvement, the system can contribute to reducing traffic violations and enhancing road safety in our communities.

Table of Contents.

Table of Contents

List of Figures

List of Tables

Abbreviations/Acronyms

HTML: Hyper Text Markup Language

LED: Light-Emitting Diode

OCR: Optical Character Recognition

OS: Operating System

PHP: Hypertext Preprocessor

SSD: Solid-State Drive

SMS: Short Message Service

SSH: Secure Shell

TLCPS: Traffic Light Compliance and Penalty System

USB: Universal Serial Bus

Chapter 1: Introduction

1.1 Background and scope of the project

The traffic lights compliance and penalty system is an implementation that aims to address the issuing of penalties to drivers that have violated the traffic lights on the red light for stop. The identification of these specific drivers is through detection of their license plates using a deep learning model that is trained using tensor flow to capture license plates. The system utilizes different technologies such as OpenCV, Python, PHP, HTML, Tesseract-OCR, deep learning using Tensor Flow to achieve its objectives.

The system's main purpose is to automate the process of license plate capturing, detection, recognition, and analysis to extract the extract and analysis to obtain the corresponding captured driver. The detection is aided with the use of an ultra-sensor that helps to measure the distance within which the vehicle is, and detect it. The distance measuring is to ensure that the vehicle detected on fall in the particular lane of the road. It's the constraints that help the system not to detect vehicles out of its lane of the road for the detection.

By implementing computer vision techniques using OpenCV and Python, the system can detect and extract license plate regions from images or video streams from the pi camera frame relating to the trained deep learning model. It then utilizes Tesseract-OCR for accurate text recognition of the extracted license plate text. Additionally, the system integrates with the Africa's Talking API to send SMS notifications to the driver's contact issuing a penalty corresponding the captured license plate.

1.2 Overview of the document

The implementation, testing, and validation results of the Traffic Lights compliance and penalty system are included in this document. The document is organized into numerous sections to make it easy to grasp the system's implementation, testing, installation, and maintenance aspects.

Introduction

This chapter provides an introduction to the document presenting the background and scope of the project. It also offers an overview of the entire document and sets the context for the subsequent chapters as well the system's implementation details.

System Specifications

In this chapter, the complete system specifications are described. It includes the versioning of requirements, input details, output formats, system functionalities, limitations, safety precautions,

default settings, special requirements, errors, and possible alarms. The chapter also covers aspects such as version control and tools used for code management like git.

Design Output

This chapter entails the implementation details, coding, and compilation aspects. It includes information about the development tools used, any anomalies encountered during the implementation, device interfaces and equipment supported, hardware and software operating environment, and documentation details.

Inspection and Testing

This chapter focuses on the inspection plan and performance of the system. It covers the inspection of design outputs, documentation, software development environment, and the results of the inspection. The chapter also discusses the test plan and performance, including test objectives, scope, types of tests, test cases, and expected results.

Installation and System Acceptance Test

In this chapter, the installation process and system acceptance testing are addressed. It includes details about input files, supplementary files, installation qualification, and a checklist for the installation procedure.

Performance, Servicing, Maintenance, and Phase Out

Here, the requirements for service, maintenance, performance, and support are discussed. The chapter covers topics such as problem detection and solutions, functional maintenance, performance improvement suggestions, and the phase-out process.

Conclusion and Recommendations

The final chapter provides a conclusion based on the findings of the report. It summarizes the key points discussed throughout the document and offers recommendations for further improvements or enhancements.

Appendix A: User Manual

This appendix provides a user manual for the traffic lights compliance and penalty system. It includes detailed instructions on how to use the system, where to seek help or support, and any accompanying visual aids such as screenshots or diagrams.

Finally, the document concludes with a section for final approval for use, where the identification of the responsible personnel, validation status, and any remarks can be recorded.

2. System Specifications

2.1 Version of requirements and Version Control

The version of the requirements document for the Traffic lights compliance and penalty system is 1.0. Throughout the development process, changes have been made to the requirements document to accommodate evolving project needs and stakeholder feedback. The changes made between versions Python 3.7.0 and raspberry pi 3B+. The change in the python version 3.9.0 to 3.7.0 inside the project folder to create a safe virtual environment to run the project dependencies. The change in the version of the raspberry pi 3B+ from raspberry pi 4 was also made to the system.

The reason for the change in the python version was to accommodate the python dependencies that couldn't be installed on our raspberry pi 3B+. In order to make the system more efficient to our software components, the device provides a way to install the python 3.7.0 inside the project folder in order not to affect other components that are installed on python 3.9.0. Therefore, the virtual environment inside the folder takes into account of python 3.7.0.

The reason for the change in the raspberry pi version to 3B+ was due to the cost funds of the whole project. And also, both version of the raspberry pi (4 and 3) performs the same functions. However, version 3 is weaker but we created software components that are lighter to take into account of the efficiency and performance of the system.

To manage versioning of the code and ensure effective collaboration, version control tools such as Git were utilized. These tools allow for tracking changes, branching, merging, and maintaining a history of code revisions. By leveraging version control, we managed different versions of the software and identified specific versions (such as 3.9, 3.7) based on the commit history and tagging.

2.2 Input

The Traffic lights compliance and penalty system receives various inputs, which are crucial for its functionality and performance as well. They act as the basis to which the system operates.

Input 1: Image or video data

Before we capture the images, the system detects using the ultra-sonic sensor a violating vehicle and the pi camera video is obtained at that moment. The system then accepts image or video data as input for license plate detection and recognition and the input is obtained from the raspberry pi camera.

Input 2: User configuration settings

Users can provide configuration settings, such as detection thresholds, or text analysis options. The region of interest and detection thresholds are inputs obtained from the deep learning

model that detects the license plate region and therefore, they influence the system's behavior and customization.

You can state the traffic officer login details as follows:

Input 3: Traffic Officer Login Details

The traffic officer login details are required to access the system. The traffic officer needs to enter their user id and password in the respective fields to authenticate themselves and gain access to the system's functionalities. This helps to ensure confidentiality and security of the system.

2.3 Output

Output 1: Detected license plate information

The system outputs the detected license plate regions within the input images or video frames using the labels from the trained model. It provides bounding box coordinates which indicate the location of the license plates.

Output 2: Extracted license plate text

The system performs text analysis using Tesseract-OCR on the detected license plate regions and it outputs the recognized text which represents the alphanumeric characters present on the license plates. The plate is stored in a relational database in a table containing a trigger that matched the plates to the corresponding plate in the driver table and the driver details are obtained to be stored in the other table such that the text can be constructed.

Output 3: SMS notifications

After analyzing the license plate text and obtaining the driver contact using the triggers, the system can send SMS notifications to the violating driver for the penalty issuing. The notification includes information such as the license plate number, timestamp, and location where the vehicle was captured from.

2.4 Functionality

Functionality 1: License plate detection

The system utilizes computer vision techniques like Open Cv to detect and localize license plates within images or video frames. It employs object detection algorithms such as a deep learning-based model that uses the single shot multi-box detector (SSD) that predicts the object bounding boxes and class labels in a single pass, to identify license plate regions accurately.

Functionality 2: License plate recognition

The system performs optical character recognition (OCR) on the detected license plate regions to extract the alphanumeric characters. By leveraging Tesseract-OCR, the system can recognize and interpret the license plate text with high accuracy to obtain the text from the image boundaries.

Functionality 3: SMS sending

The system integrates with the Africa's Talking API to send SMS notifications to the driver's contacts who are captured violating the traffic lights. After analyzing the license plate information, the system composes and sends SMS messages to predefined contacts issuing a penalty.

Functionality 4: Traffic Officer Performance Metrics

To assess the performance and effectiveness of the system, a performance monitoring feature was implemented for monitoring the system by the authorized IT traffic officer. This functionality allows the tracking and analysis of various metrics related to the system performance such as capturing and identifying the violating drivers and accuracy of the captured number plates.

2.5 Limitations and safety

Internet connectivity

The system requires an active internet connection to interact with the Africa's Talking API for sending SMS notifications. Without internet access, the SMS sending functionality will be unavailable.

Image quality and variations

The accuracy of license plate detection and recognition heavily depends on the quality of input images or video frames. Challenges such as low lighting, blur, or frame movement or video display affect the system during it performance. This is most likely due to low power system component CPU i.e. the raspberry pi 3 B+.

Safety Precaution

Data privacy

The traffic lights compliance and penalty system handles sensitive information, such as license plate numbers and personal data for the drivers. Appropriate measures are taken in place to ensure the privacy and security of this information, adhering to relevant data protection regulations through allowing only the authorized administrator to access the system to monitor its performance.

2.6 Default settings

By default, the system should be able to perform License plate detection based on a default object detection model with predefined detection thresholds that was trained to perform the detection through the Open Cv.

For License plate recognition, by default Tesseract-OCR settings for license plate text extraction should be installed to run the extraction program.

For SMS sending, Africa's Talking API configuration with predefined message content and driver contact identification should be deployed to perform issuing of penalties.

2.7 Special requirements

Hardware specifications

The system requires a more powerful raspberry pi with sufficient processing power and memory to perform real-time or near real-time license plate detection and recognition. GPU acceleration and Coral platform to optimize for running machine learning models effectively allowing for faster and real time influence of the raspberry pi performance, but it is not mandatory.

Software dependencies

The system relies on several software dependencies including OpenCV, Python, Tesseract-OCR, Africa's Talking API, and any other libraries or frameworks specified in the documentation. These dependencies must be installed and configured correctly for the system to function as intended.

2.8 Errors and alarms

2.8.1 License plate not detected

One of the most frequent errors encountered in the system was the failure to detect any license plate regions in the input images or video frames based on the license plate detection model. The error is identified when no label region if encountered around the detected license plate.

This could be due to factors such as poor image quality, incorrect region of interest selection, or limitations of the object detection algorithm therefore to enhance the system to avoid the error we can use the coral platform to improve functionality or deploy the system with better hardware components.

2.8.2 Text recognition failure

Another common error was when the system cannot accurately recognize the license plate text. if the system fails it produces an error message due to failure to detect the text.

Some of the factors that may contribute to recognition failures include poor image quality, complex background patterns, or variations in license plate design.

To address this error, training the model with more data and through more steps can improve the detection of the images captured. Deploying other OCR platform such as Google cloud Vision, Amazon Textract, OCRopus could be more accurate than tesseract.

2.8.3 Alarm 1: Failure to Issue Penalty

In the event of an error or issue with issuing penalties due to failure to send SMS, the system generates an alarm to address the failure. The error could occur due to network connectivity problems, incorrect Africa's Talking API configuration, or limitations imposed by the SMS service provider.

To go around the error, always verify and check the internet connections of the system and ensure maintenance activities and always confirm the API used.

Chapter 3: Design output

3.1 Implementation (coding and compilation)

The system was developed using different programming languages that communicate together to accomplish the system functionalities. Python, a widely and versatile programming language was used to run the sensor and led lights that simulate the traffic lights control system to perform object detection using a trained model. The entire process of object detection and license plate capture was facilitated by python.

The system has a web interface thats developed using the web-based programming languages such as HTML, PHP, JAVA SCRIPT.

Finally, the raspberry pi system runs on the bullseye operating system 32-bit software.

For coding and development, the system was developed using an integrated development environment that utilizes Visual Studio Code that connects to the raspberry pi through a remote connection using SSH. It provides a streamlined development experience with code editing, debugging, and version control integration.

The frameworks and libraries used to enhance functionality include open Cv that was used for image processing, computer vision tasks, and object detection. They provide functions and algorithms for manipulating and analyzing images and videos. Additionally, Tensor flow library was employed in the system for machine learning tasks specifically for training and deploying object training models. Tensor flow offers a high-level API that simplifies the process of building and training models within platforms such as Jupyter note book and Colab an online platform.

Version control such as Git were employed to manage the source code and track changes throughout the development process and as the system relied on python, build tools like pip is utilized to install and manage the required dependencies. Pip simplifies the installation process and ensures that the necessary libraries and packages on the raspberry pi control system.

During the development process, various anomalies and issues were encountered. These anomalies include software bugs, performance bottle necks or compatibility problems in libraries and software platforms.

3.2 Device interfaces

3.2.1 Ultra sonic sensor

These are used to detect the vehicle that is violating the traffic lights. The reason behind the use of the ultra-sonic sensor was to both detect the vehicles and create a difference between the lanes of the road. The roads consist of two lanes the right and the left, therefore the ultra-sonic sensor will only detect the vehicles within a given distance for which the traffic lights are ordering and it can also be used to detect distance.

3.2.2 A breadboard

These were used for prototyping circuits without the need for soldering. It consists of a grid of interconnected metal strips and holes that allows to insert and connect electronic components easily. The breadboard provided a platform to connect the ultra-sonic sensor and LED lights to the Raspberry Pi.

3.2.3 LED lights (Light-Emitting Diodes)

These were used to visually simulate the traffic light control system.

3.2.4 Computing devices such as computer, flat screen

These were used to access the system and to visualize the web interface and also help in operating the and programming the raspberry pi.

3.3 Hardware environment

Consists of components such as raspberry pi, power supply, peripherals, bread board.

The raspberry pi 3 model B+ is a micro control processing system unit for managing and controlling the system function.

The power supply unit such as a USB power adapter is used to supply the power to the raspberry pi to power it up.

Peripherals highlight the additional devices that are connected to the raspberry pi to add on its functionality. These include flat screen, keyboard, mouse, and the pi camera module for the raspberry pi.

3.3.1 Software Environment consist of different operating software the is installed on the raspberry pi such as Raspbian, Raspberry pi OS. Additionally, other software used on the system include, python 3.7.0, tensor flow 2.5, OpenCV 4.5, Tesseract 4.1, HTML, PHP, and JAVA Script.

The system runs on a network and we deployed a network router such as a phone that provides interconnection between the devices that interacts the system to a local area network and internet access.

3.4 Documentation

3.4.1 User Manuals

User manuals provide instructions and guidelines on how to use the system effectively. They explain the system's features, functionalities, and operation, helping users understand how to interact with the system and achieve their desired tasks.

3.4.2 Technical Specifications

Technical specifications provide detailed information about the system's architecture, components, interfaces, and requirements. They serve as a technical reference for developers, system administrators, and other technical personnel involved in the implementation, maintenance, and troubleshooting of the system.

3.4.3 Design Documents

Design documents outline the system's design approach, methodology, and overall structure. They describe the system's modules, their interactions, and the rationale behind design decisions. Design documents are important for developers, as they provide insights into the system's design principles and assist in understanding its internal workings.

3.4.4 Installation Guides

Installation guides provide step-by-step instructions for installing and configuring the system. They ensure that the system is properly set up and ready for use, guiding system administrators or end-users through the installation process.

Table 1: Design details (check all that apply to your project. Make sure you can defend what you tick)

Topics	**Design output**	
Good programming practice Efforts made to meet the recommendations for good programming practice...	Source code is... ☑ Modulized ☑ Encapsulated ☑ Functionally divided ☐ Strictly compiled ☑ Fail-safe (handling errors)	Source code contains... ☐ Revision notes ☑ Comments ☑ Meaningfull names ☑ Readable source code ☐ Printable source code
Dynamic testing Step-by-step testing made dynamically during the implementation...	☑ All statements have been executed at least once ☑ All functions have been executed at least once ☑ All case segments have been executed at least once ☐ All loops have been executed to their boundaries ☑ Some parts were not subject to dynamic test Comments:	

Chapter 4: Inspection and testing

4.1 Introduction

The inspection and testing phase of the Traffic lights compliance and penalty system was crucial to ensure its overall quality, functionality, and adherence to the specified requirements. This phase involves a systematic and thorough evaluation of the system's components, features, and performance to identify any discrepancies, errors, or areas for improvement.

The primarily objectives of the inspection and testing include, Quality Assurance to verify that the system meets the desired quality standards such as accuracy of the license plate text from the captured images, identification of the license plate and the driver that’s violating the traffic light.

Functionality Validation was essential to ensure the system’s responsiveness to perform its specified functions.

Risk mitigation to identify and address any potential risks, issues, or vulnerabilities that could affect the system's performance or security. Such as dark environment that restricts the system from detecting and capturing the license plates.

Table 2: Inspection plan and performance

Topics	**3.3.1 Inspection plan and performance**	Date / Initials
Design output Results from the Design Output section inspected...	☑ Program coding structure and source code ☑ Evidence of good programming practice ☐ Design verification and documented reviews ☑ Change-control reviews and reports Comments: The code best operates on the raspberry pi.	10/06/2023 – NB 11/06/2023 – KC
Documentation Documentation inspected...	☑ System documentation, flow charts, etc. ☑ Test results ☑ User manuals, On-line help, Notes, etc. ☑ Contents of user manuals approved Comments: Images were included to aid easy understandability.	10/06/2023 – NB 11/06/2023 – NA 11/06/2023 – OD
Software development environment Environment elements inspected...	☑ Data integrity ☑ File storage ☑ Access rights ☐ Code protection ☑ Installation kit, replication and distribution Comments: Data is stored and easily managed in a relational database ie Maria DB	10/06/2023 – NB 11/06/2023 – KC 10/06/2023 – NB 11/06/2023 – NA 11/06/2023 – OD

Topics	**3.3.1 Inspection plan and performance**	Date / Initials
Result of inspection Approval of inspection.	☑ Inspection approved Comments: The system inspection is approved by all the group members to ensure full functionality.	09/06/2023 – NB 09/06/2023 – NA 11/06/2023 – OD 11/06/2023 – KC

4.2 Test plan and performance

Test Objectives of the development of the system

The test objectives for the Traffic lights compliance and penalty system outlined the specific goals of the testing phase. This included verifying the functionality of the traffic lights, assessing the accuracy of the object detection algorithm, evaluating the performance of the system under different traffic conditions, ensuring compliance with regulatory requirements and ensuring the issuing of penalties to violating drivers.

Test Coverage

The test plan defined the scope of the testing activities for the Traffic lights compliance and penalty system. It identified the components and functionalities of the system that need to be tested, such as the hardware components (Raspberry Pi, sensors, LEDs), software modules (object detection model, web interface), and their interactions.

Test Methods and Techniques

The test plan specified the testing methods and techniques to be employed for the Traffic lights compliance and penalty system. This included functional testing to ensure proper functioning of each component, performance testing to assess response times and system scalability, security testing to identify vulnerabilities, and usability testing to evaluate user interaction with the web interface.

Test Procedures

The test plan outlined the step-by-step procedures for conducting the tests specific to the Traffic lights compliance and penalty system. It included instructions for setting up the hardware

components, configuring the software environment, simulating traffic scenarios, and collecting test data. Additionally, it defined the expected results and criteria for passing each test.

Test Data and Environments

The test plan identified the test data required for evaluating the Traffic lights compliance and penalty system. This involved generating sample traffic patterns, capturing real-world traffic data, or using simulated data for testing purposes. The test plan specified the necessary test environments including the hardware and software configurations required to run the system.

Test Execution and Reporting

The test plan detailed how the tests will be executed and how the results will be recorded and reported for the Traffic lights compliance and penalty system. It would include information on the testing team responsible for carrying out the tests, the schedule for conducting the tests, and the format for documenting the test outcomes. This ensures that the testing process is systematic, well-documented, and allows for tracking any issues or deviations encountered.

4.2.1 Test objectives

The main objective of the system to detect, capture license plate and issue a penalty, a Traffic Violation Detection and Penalty Issuance Test was performed.

Description

The Traffic Violation Detection and Penalty Issuance Test aims to verify the system's ability to detect and capture vehicles that violate traffic lights, as well as issue penalties by sending SMS notifications. This test ensures that the system accurately identifies traffic violations, captures the license plate clearly, and effectively delivers penalty notifications to the responsible vehicle owners.

Why

The test was conducted to ensure the system's effectiveness in enforcing traffic regulations, deterring violations and promoting road safety. By validating the detection and penalty issuance functionalities, we can ensure that the system operates reliably and contributes to the overall traffic management objectives.

Steps Taken and What Was Tested.

1. Set up the Test Environment:

Prepare the physical setup with the traffic lights, cameras, ultra- sensors, and Raspberry Pi control system and configure the system software and ensure proper connectivity using the bread board and jumper cables.

2. Define the Test Scenario:

Simulate traffic scenarios where vehicles may potentially violate traffic lights. such as red-light running. Then determine the expected behavior of the system when a violation occurs.

3. Perform Test Violations:

Trigger traffic violations intentionally by simulating vehicles violating traffic lights and validate that the system detects and captures the vehicle license plate accurately.

4. Capture Evidence:

Verify that the system captures relevant images or videos of the violating vehicles and ensure that the captured images are clear, identifiable, and timestamped through observing the system’s extracted license plate from the image.

5. Penalty Issuance:

Confirm that the system generates penalty notifications for the identified violations and validate that the penalty notifications the reason for the penalty and where it was captured.

6. SMS Delivery:

Verify that the system sends SMS notifications through the Africa’s Talking API to the registered vehicle owners associated with the violations and ensure that the SMS notifications

4.2.2 Scope and Relevancy of tests

1. Coverage:

The tests covered all critical aspects of the system related to detecting and capturing license plates, extracting the license plate text and issuing penalties through delivering SMS notifications. This includes testing the functionality of the traffic lights, cameras, ultra-sensors, Raspberry Pi control system, image processing algorithms, license plate recognition and penalty generation.

2. Volumes:

The tests considered varying volumes of traffic lights violations to ensure that the system can handle different scenarios effectively. This includes testing the system's performance and scalability in detecting and capturing license plates accurately.

3. Complexity:

The tests addressed the complexity of the system by evaluating its ability to handle different license plates extracting from the images and conditions in which the system finds hard time to perform its objective. This includes testing the system's robustness and reliability in detecting license plates in real-time and capturing the actual plates in challenging environments.

Therefore, the tests align with the project objectives which include the detection, capture, and penalty issuing for traffic lights violations. They focus on verifying that the system can accurately detect violations, capture the necessary license plates and penalties associated registered vehicle owners.

4.2.3 Levels of tests

1. **Module Test**

The module tests focused on testing the individual components or modules that contribute to achieving the system's objective. This included testing the functionality and accuracy of modules such as the traffic light detection module, image processing module, license plate recognition module, penalty generation module, and SMS delivery module. The module tests ensure that each component performs its specific tasks correctly and contributes to the overall objective of detecting and capturing license plates of the violating drivers.

2**. Integration Test.**

The integration tests aimed at verifying the seamless integration and communication between different modules of the system. This involved testing the interactions and data exchange between the modules to ensure that information flows correctly throughout the system. For example, it included testing the integration of the traffic light detection module with the image processing module, the integration of the license plate recognition module with the penalty issuing module, and the integration of the penalty issuing module with the SMS delivery module. The integration tests ensured that the modules work together harmoniously to achieve the objective of detecting violations and issuing penalties.

3. **System Acceptance Test.**

The system acceptance test focused on evaluating the system as a whole to ensure it meets the defined acceptance criteria and fulfills the intended objective. This involved conducting end-to-end tests that simulate real-world traffic scenarios to verify the system's capability to detect and capture vehicles violating traffic lights accurately. Additionally, it ensured the proper generation and delivery of penalty notifications through SMS. The system acceptance test validates that the system performs as expected, meets the specified requirements, and successfully achieves the objective of detecting violations and issuing penalties.

4.2.4 Types of tests

1**. Input Tests**:

Input tests focused on validating the system's ability to handle different types of inputs effectively. In the context of the system, input test involved providing various traffic scenarios as inputs to test how the system responds to different license plate detection. These tests ensure that the system can accurately detect and process different types of input data related to license plates of the violating vehicles.

2. **Functionality Tests**:

Functionality tests assess whether the system functions correctly and performs the intended operations. In this case, functionality tests verified that the system accurately detects traffic violations, captures images or videos of the violating vehicles, extracts license plate information, issues penalty notifications, and sends SMS notifications to the respective vehicle owners. The tests ensured that the core functionalities of the system align with the objective of detecting and capturing traffic violations while issuing penalties through SMS.

3**. Boundary Tests.**

Boundary tests evaluated the system's behavior at the boundaries of its capabilities. For example, it involved testing the system's response to extreme or edge cases, such as vehicles violating traffic lights at high speeds or in challenging lighting conditions. The tests ensured that the system can or can’t handle and process data at the limits of its expected operational scenarios, identifying any potential issues or vulnerabilities in the system's performance.

4. **Performance Tests.**

Performance tests assed the system's performance in terms of speed, accuracy, and resource utilization. In the context of the traffic lights violations and penalty system, performance tests measured the system's response time in detecting and capturing violations, the accuracy of license plate recognition, and the efficiency of penalty generation and SMS delivery. These tests ensured that the system meets performance expectations and can handle the anticipated volume of traffic violations effectively.

4.2.5 Sequence of tests

1**. Test Case: Red-Light Violation Detection**

Test Procedure: Simulate a scenario where a vehicle runs a red light.

Test Data: a vehicle violating the red lights traffic and it’s captured by the system.

Expected Result: The system accurately detects the red-light violation, captures clear images or videos of the violating vehicle, extracts the license plate using tesseract and generates a penalty with the appropriate details of the owner if the exists.

2. **Test Case: Penalty Notification Delivery**

Test Procedure: Verify the delivery of penalty notifications through SMS.

Test Data: Simulate sending penalty notifications to registered vehicle owners.

Expected Result: The system sends SMS notifications to the respective vehicle owners associated with the violations, delivering the penalty information accurately and in a timely manner.

3. **Test Case: Penalty Management**

Test Procedure: Validate the system's penalty management functionality.

Test Data: Track penalties issued to violators and their corresponding records through the system’s dashboard.

Expected Result: The system correctly records and manages penalty information allowing easy retrieval of penalty records and tracking of violators who have been issued penalties.

4.2.6 Configuration and calculation tests

1. **Configuration Test.**

Platform Compatibility: Verify that the system is compatible with the chosen hardware platform, such as Raspberry Pi and that all necessary components (traffic lights, pi-cameras, ultra-sensors) are properly connected and configured.

Network Integration: Test the system's ability to integrate with the network infrastructure, ensuring that it can send and receive data, including SMS notifications through the Africa's Talking API.

Integration with Other Systems: Validate the integration of the system with any external systems or APIs required for functionality, such as license plate recognition or penalty management databases.

2. **Calculation Test.**

Known Input Validation: Provide specific input scenarios with predetermined outputs to validate the accuracy of calculations and algorithms used in the system.

Algorithm Verification: Ensure that the system's algorithms for detecting traffic violations, extracting license plate information, and generating penalty notifications produce the expected outputs for known inputs. This involves validating the accuracy and reliability of the system's calculations and algorithms.

4.3 Precautions

4.3.1 Anomalous conditions

Anomalies are unexpected or abnormal behaviors, conditions or events that deviate the expected or intended operation of the system. Therefore, we encountered some anomalies in the interaction between the system and the instruments used such as the traffic led light could sometimes fail to follow the pattern of the lights and also incase an image is identified and the text is not recognizing the system could fail however much the image contains the plate number.

4.3.2 Precautionary steps taken

1. Raspbian Configuration.

Ensuring that the Raspbian operating system is properly configured for the system's requirements. This involves setting up the appropriate software packages, libraries, and dependencies, as well as configuring system settings for optimal performance and compatibility.

2. Device Compatibility.

Verifying the compatibility of the hardware devices used in the system, such as cameras, sensors, and Raspberry Pi, with Raspbian. This may involve installing device drivers, adjusting configurations, and conducting compatibility tests to ensure smooth integration and operation.

3. System Stability:

Monitoring the system's stability and addressing any potential anomalies or undesired operating conditions specific to the Raspbian environment. This includes managing system resources effectively, monitoring for system errors or crashes, and implementing appropriate error handling mechanisms.

Chapter 5: Installation and system acceptance test

5.1 Input files

1. Raspbian Operating System.

This is the primary file that needs to be installed on the Raspberry Pi's SD card. It contains the necessary components and configurations to run the Raspbian operating system.

2. System Configuration Files.

These files include various configuration settings specific to the Raspberry Pi, such as network settings, user accounts, system preferences, and hardware configurations. Examples of such files are `/etc/hostname`, and `/boot/config.txt`.

3.Python Scripts.

The system may consist of multiple Python scripts responsible for different functionalities, such as traffic light detection, license plate recognition and penalty issuing. These Python files contain the code for implementing these features.

4.Libraries and Dependencies

Depending on the specific requirements of the system, certain libraries and dependencies may need to be included in the installation media. These files ensure that the system has access to the necessary resources and functions to run properly. Examples may include the

TensorFlow library for object detection, the OpenCV library for image processing, and MySQL for database connections.

5.2 Supplementary files

Readme Files

These files provide instructions, guidelines, or additional information about the system. They typically include details on how to set up the system, install dependencies, configure settings, and troubleshoot common issues. Readme files are useful for users and developers to understand the system's functionality and usage.

License Agreements.

These files contain the licensing terms and conditions for the software used in the system. They specify the rights and limitations associated with using, modifying, or distributing the software. License agreements ensure compliance with open-source or proprietary software licenses and protect the intellectual property rights of the software.

Documentation.

This may include user manuals, technical guides, or API documentation that provide comprehensive information about the system's features, functionalities, and usage. The documentation assists users and developers in understanding how to interact with the system, utilize its capabilities, and integrate it into other applications if needed.

Table 3: Checklist of the Installation and system acceptance test

Topics	**Installation summary**
Installation method Automatic or manual installation...	☐ Automatic - installation kit located on the installation media ☑ Manual - Copy & Paste from the installation media Comments: Since the raspberry pi operates on a linux OS, we used copy and paste to install the files and libraries.

Topics	**Installation summary**
Installation media Media containing the installation files...	☑ Diskette(s) ☐ CD-ROM ☑ Source disk folder (PC or network) ☑ Download from the Internet Comments: The Raspbian OS is downloaded from the internet therefore an internet connection is required. Together with downloading software like SD card formatter the helps to prepare the sd card for the OS.
Installed files List of (relevant) installed files, e.g. EXE- and DLL-files, spreadsheet Add-ins and Templates, On-line Help, etc.	• DBK files • Plg files • Opt files

Table 4: Installation Procedure Check

Topics	**Installation procedure**	Date / Initials
Authorization Approval of installation in actual environment.	Person responsible: NAKIBING BONNY	28TH MAY 27023
Installation test The following installations have been performed and approved...	☑ Tested and approved in a test environment ☑ Tested and approved in actual environment ☐ Completely tested according to test plan ☑ Partly tested (known extent of update) Comments:	09/06/2023 – NB 09/06/2023 – NA 11/06/2023 – OD 11/06/2023 – KC

Chapter 6: Performance, servicing, maintenance, and phase out

6.1 Service and maintenance

Due to the need for change in software equipment and in the environment, software has to continuously change in order for it to remain functional and be able to perform its tasks to accomplish a certain set of requirements.

To ensure proper functionality and reliability of the traffic lights compliance and penalty system, as a team we performed several tasks to make sure our system can be fully functional and efficient through capturing, license extraction and issuing of penalty to the violating driver.

For maintenance procedures, we performed regular inspection of the hardware components such as the sensors, resistors in the system, lights and the raspberry pi to ensure that they are functioning at full capacity. Software inspections of the raspberry pi such that it is always up to date. And before installing any other libraries, frameworks and other dependencies, we always first update the system through running “sudo apt-get update” command such that we are able to detect and fix bugs in our software and also recover from security patches and finally to enhance the performance of the system.

We also performed maintenance and training through the user manual. IT traffic officers were trained of how the system works and operates by providing step by step procedures for routine maintenance procedures, how to trouble shoot and problem and how to resolve the problem. This was to elevate the system performance and avoid issues of misuse of the system.

6.2 Performance and Maintenance

Service and maintenance activities are crucial to keep the system functioning optimally. Regular inspections of hardware components (raspberry pi, sensor, pi camera), software updates (Raspbian, python, tensor flow), database maintenance, and data backups are necessary to prevent unexpected failures and minimize downtime. By performing these activities, the system will operate smoothly and ensure reliable performance.

Performance requirements are also significant in this phase. The system should meet specific performance criteria, which may include the maximum time taken to deliver an SMS after identification of the violating driver. Monitoring and optimizing the system's performance are essential to meet user expectations and maintain a satisfactory user experience. Performance testing and monitoring tools can be used to assess and improve system performance.

Effective user support is another critical aspect during this phase. We will help the IT traffic officers by providing technical assistance, troubleshooting guidance, user training, and prompt response to inquiries or issues raised by IT traffic officers towards the system hardware components. User support ensures that users can effectively utilize the system, resolves problems in a timely manner, and enhances overall user satisfaction.

The causes for incorporating changes into the system raised due to system performance to visualize the violating license plates and modification in the system components such as the pi camera and the sensors to maintain full functionality. And upgrades should be incorporated into the system through updating the software remotely using the SSH to access the raspberry pi control system.

Since the data is on the MySQL database on the cloud, in terms of moving from the old system to a new system, we shall provide an analysis on the schema and design of our existing SQL database to understand the tables and relationships and datatypes. Then, during the transfer from the old system to the new system, we shall create a strategy to migrate our data to the new system. Then extract the data that needs to be moved to the new system through executing SQL queries. And finally, transfer and map the data to the new system using the SQL query.

Table 5: Performance and maintenance details

Performance and maintenance details

Topics	**Performance and maintenance**	Date / Initials
Problem / solution	When the number of license plates identified are note known in the database with driver details, there will be need to always carryout hardware checks especially on the raspberry pi camera to check the efficiency of vision and also trained and update the model further more with more data.	09/06/2023 – NB 09/06/2023 – NA
Functional maintenance	If there is upgrade in the raspberry pi version, hardware needs to be changed to reflect fast performance of the system. Therefore, regular software and hardware checks need to be performed onto the system through making new installations and upgrades.	11/06/2023 – NB 11/06/2023 – KC

Topics	**Performance and maintenance**	Date / Initials
Functional expansion and performance improvement	List of suggestions and requests, which can improve the performance of the computer system. eg • Use of more advanced cameras that have a wide coverage view and clear. • Use of coral to increase the speed and functioning of the raspberry pi camera.	11/06/2023

Chapter 7: Conclusion and Recommendations

In conclusion, the Traffic Light Compliance and Penalty System has been successfully developed to detect and capture vehicles violating traffic lights and issue penalties through SMS notifications. Throughout the development process, various inspections and tests were conducted to ensure the system's quality, functionality, and compliance with requirements.

The system has demonstrated its effectiveness in accurately detecting violations, capturing license plate information, and generating penalty notifications. It has proven to be reliable in delivering SMS notifications to registered vehicle owners associated with the violations. The integration with the Raspberry Pi control system and other components has been seamless, providing a robust and efficient solution. However, there are some challenges in spend in capturing that need to be addressed using better hardware platform such as raspberry pi 4 and coral to speed up functionality.

Recommendations.

Based on the successful implementation of the Traffic Light Compliance and Penalty System, the following recommendations are made:

Deployment and Operational Considerations.

Prior to deploying the system in a live traffic environment, it is recommended to thoroughly evaluate and test its performance and scalability in the real world. Conducting pilot tests in a controlled setting can help identify any potential challenges and fine-tune the system for optimal operation.

Continuous Monitoring and Maintenance.

Implementing a comprehensive monitoring system to continuously monitor the performance and health of the system. This will help identify any issues or anomalies promptly and allow for

proactive maintenance and troubleshooting. Regular software updates and patches should be applied to ensure the system remains secure and up to date.

User Support and Training.

Providing a comprehensive user support and training to the IT traffic officer. This will enable them to effectively utilize and manage the system, handle penalty management, and address any technical issues that may arise. Clear documentation and user guides should be made available for easy reference.

Data Security and Privacy.

Ensuring strict adherence to data security and privacy regulations, including secure storage and transmission of captured images and personal information. Regular audits and risk assessments should be conducted to identify and address any potential vulnerabilities or risks.

Appendix A: User Manual

Installation and Setup:

Ensure that the Raspberry Pi control system, traffic lights, cameras, and ultra-sensors are properly set up and configured according to the system's installation instructions.

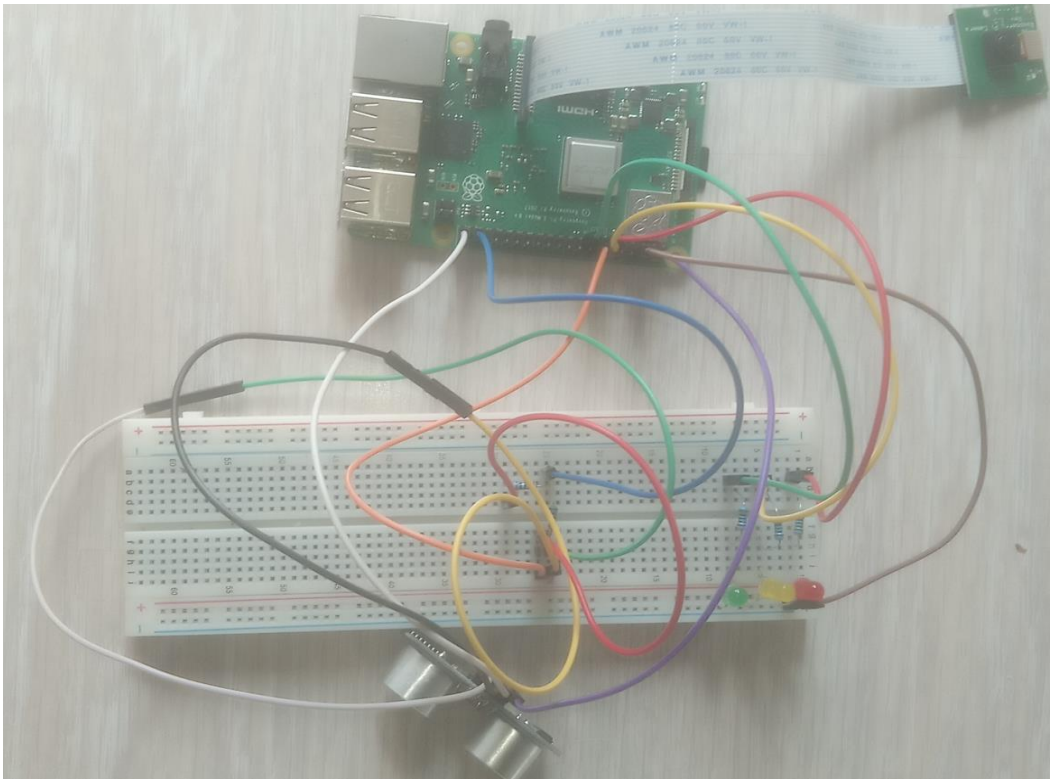

Figure 1: Components inter-connected to Each other

System Activation:

Power on the Raspberry Pi and ensure that the system software is running. Verify that the system components are properly connected and communicating.

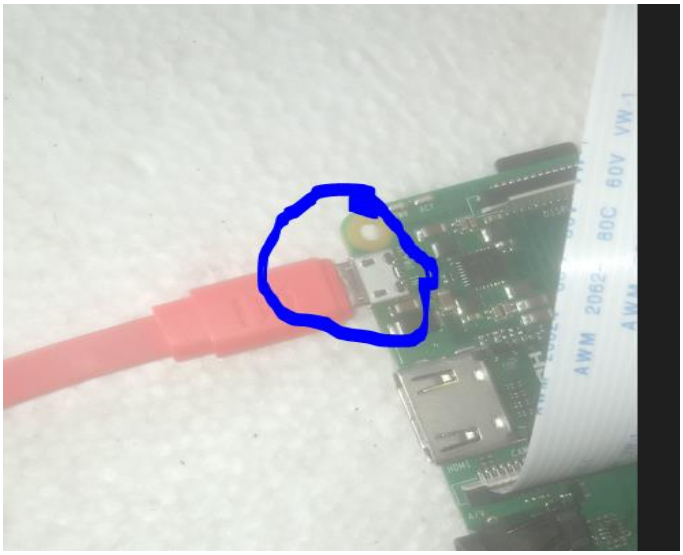

Figure 2: Connection To Power Supply

Traffic Monitoring:

The system will continuously monitor the traffic lights and capture images or videos when a violation occurs. It uses image processing algorithms to detect license plates and record relevant information.

Figure 3: insert USB Mouse and Keyboard

Violation Detection and Capture:

When a vehicle violates a traffic light, the system will detect the violation based on predefined criteria. It will capture clear and identifiable images or videos of the violating vehicle, including the license plate using the pi camera.

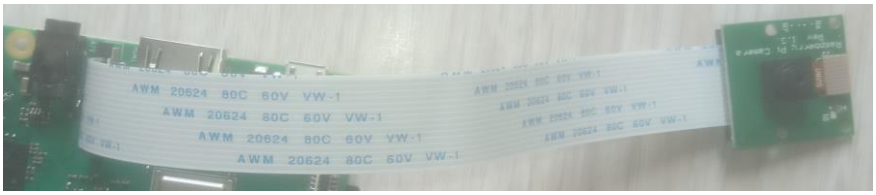

Figure 4: PI camera

Monitoring Performance.

To monitor the performance and accomplishing of tasks, connect to the same network with system and access the user interface of the system for only authorized traffic personnel will be allowed to login.

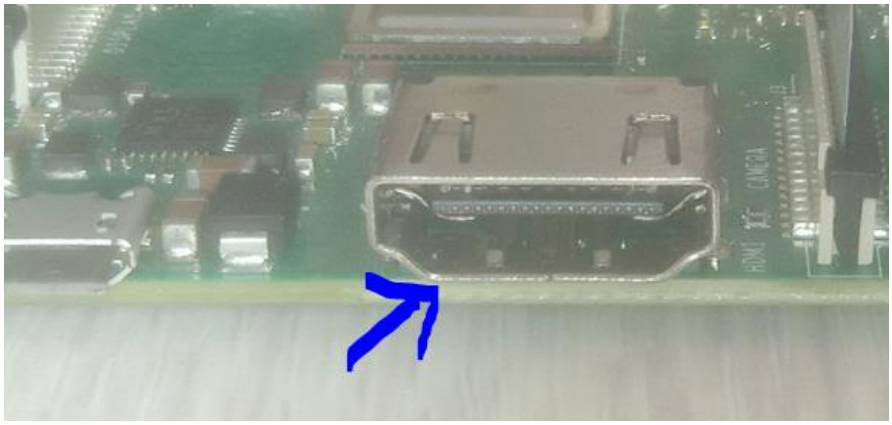

Figure 5: Connection of the HDMI to the Raspberry pi and Screen

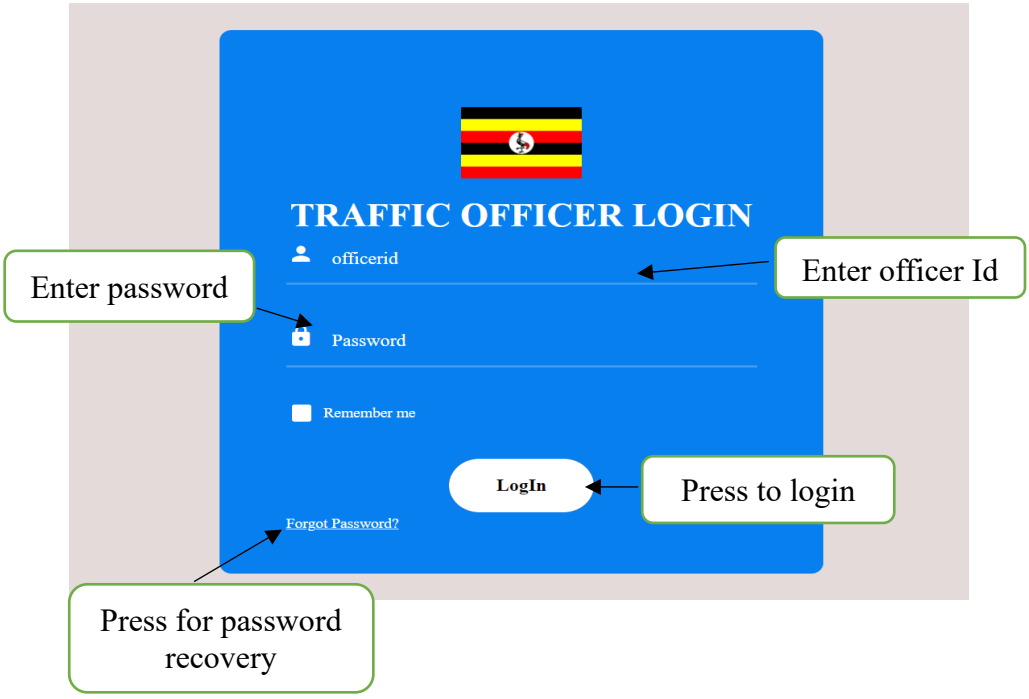

Figure.6: showing the login buttons

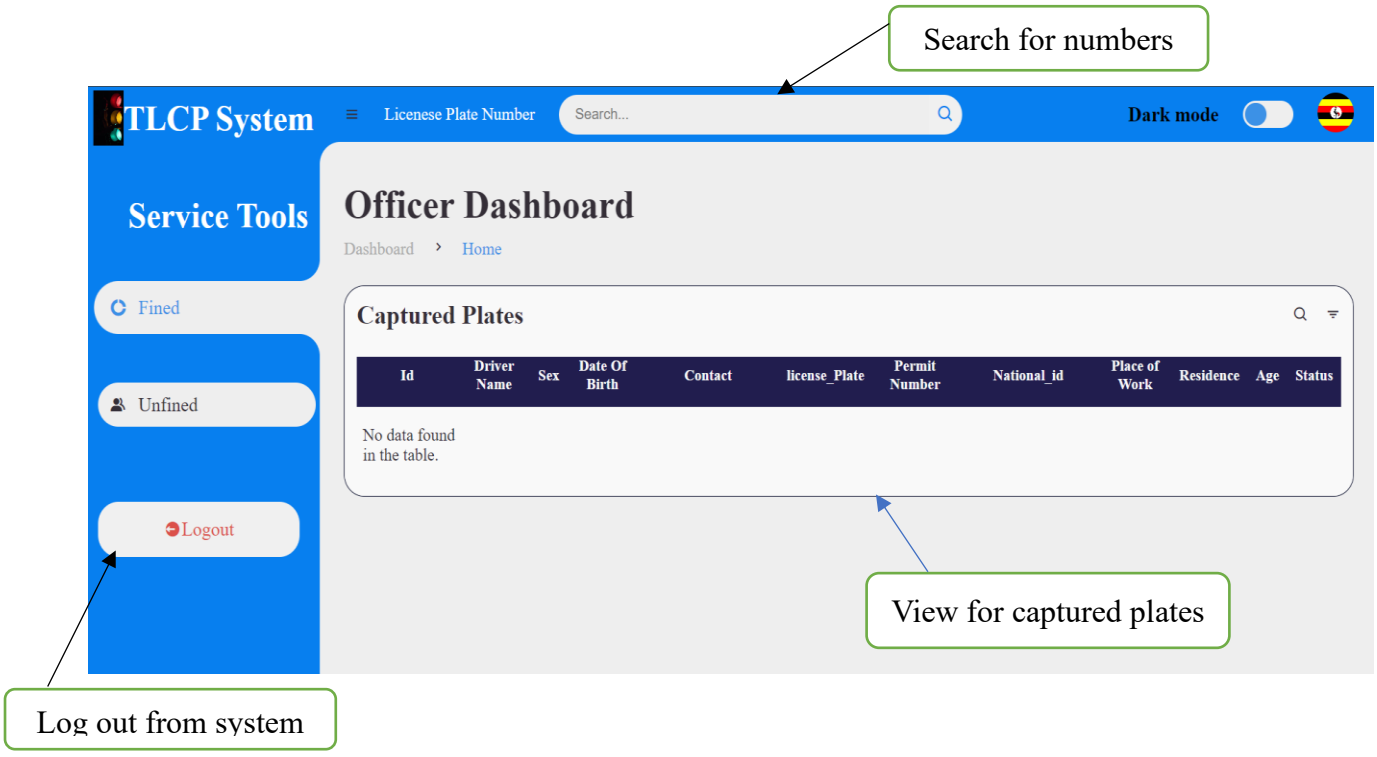

Figure 7: Screen showing Display of the system.

Seeking Help:

If you require assistance with the Traffic Light Compliance and Penalty System, you can reach out to the following resources:

User Manual: Refer to the system's user manual for detailed instructions on installation, setup, and usage. The manual should provide step-by-step guidance on system operation and troubleshooting.

Online Support: Visit the system's official website or online support portal, knowledge base articles, and troubleshooting guides. These resources can provide answers to common questions and help address technical issues.

Contact Support: If you encounter any technical issues or need personalized assistance, contact the system's support team. They can provide direct support, guidance, and further troubleshooting steps.

Name: Nakibinge Bonny
Contact: 0755665544
Role: Developer.

Name: Katembeko Christopher
Contact: 0770829724
Role: Project Co-Ordinator

Name: Nsubuga Alpha
Contact: 0705310594
Role: Project Manager

Name: Otim Derrick Mulungi
Contact: 0786604400
Role: Developer

Sources

Blog website: https://sites.google.com/view/bse23-14/home

Git hub: https://github.com/bo-nny/Traffic-Lights-Compliance-And-Penalty-System.git

Final approval for use	
Identification:	
Responsible for validation:	
Remarks:	
Date:	Signature: