



Valley Automated Vehicle  
Registration  
and  
Recognition System  
VARRS

## **Project Team & Stakeholders**

**Project Sponsor: Mr. Mupaso**

**Project Manager: Genius Mukichi**

**Programmers: Nyasha Gandah and Takudzwa Chikomo**

**Database Administrator: William Gray**

**User Interface Designers: Joseph Chisoni and Ruvarashe Chinaka**

**Quality Assurance: Rejoice Sedze**

## **Background**

At Africa University the gating system currently being used manages vehicles by manually writing down details of vehicles and drivers entering and exiting the premises and the time. The manual aspect of this process has led to some challenges.

## **Business Problem**

The vehicle registration system is inefficient and tedious leading to human error-prone data, data redundancy, poor data security, and increased vulnerability of premise assets to theft by unaccounted-for vehicles.

## **Business Solution**

Implement an automated system for car registration and recognition allowing easy and secure gate access by utilizing a database to link vehicles to staff/student-specific information. Our system recognizes vehicles by automatically reading vehicle plates using cameras and running the data against already stored user data.

## **Project Scope**

**In Scope:**

- License plate recognition system implementation

- Database creation and management for vehicle registration details
- Secure linkage of vehicles to staff/student information
- Interface development for security guards
- Automated login and logout time tracking
- Report generation capabilities.

**Out of Scope:**

- Registration and logging in/out of Africa University vehicles e.g. school bus.
- Integration with other university systems beyond the car registration system.

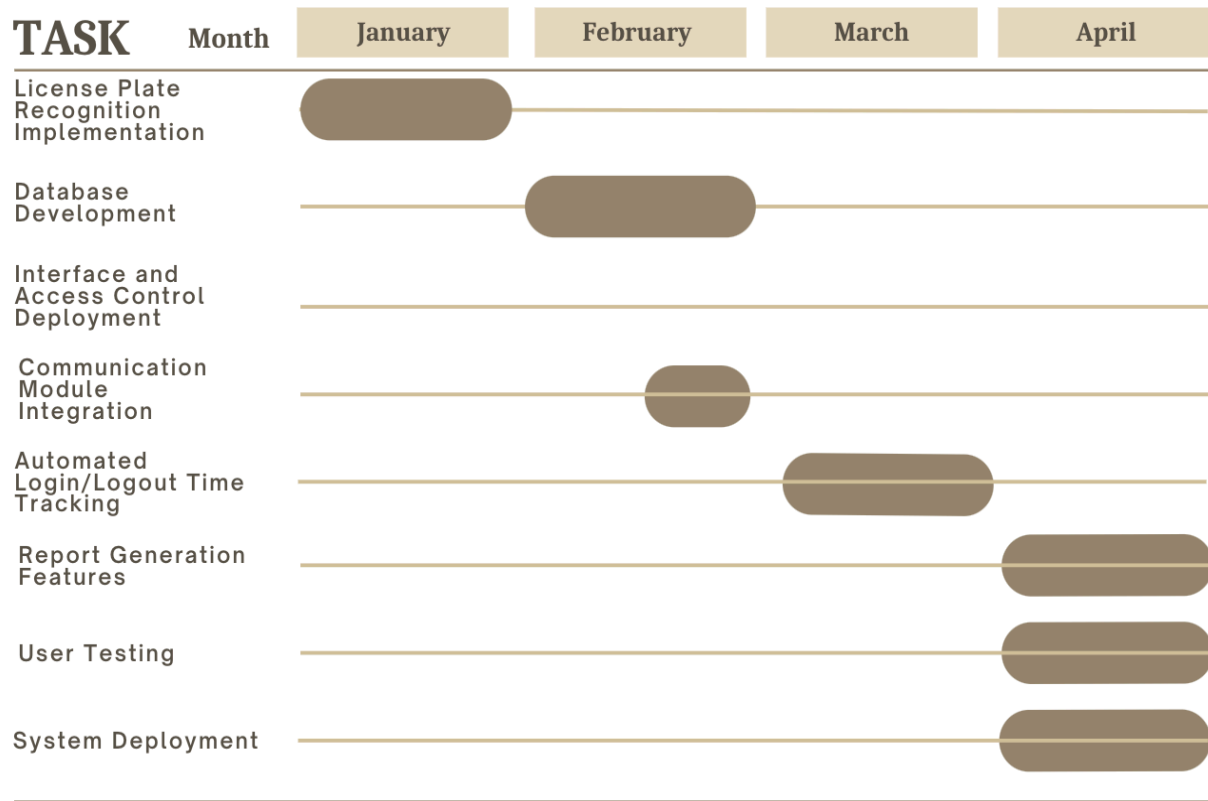
## **Success Metrics**

- Reduction in manual registration errors by at least 50% in first 2 months prior implementation.
- Enhanced security through automated recognition.
- 60% reduction in average time spent at the gate by vehicles.
- Timely communication and reporting capabilities.

## **Millstones & Deliverables**

## Vally Automated Vehicle Registration & Recognition System

## GANTT CHART



### Assumptions

Carry on this project we are assuming that:

- The project team is established and committed.
- Necessary hardware and software for license plate recognition are available.
- User training will be conducted before system deployment.
- The university approves the integration of the new system at the gate.

### Feasibility Study

### Technical Capacity

**Hardware requirements – The University possesses computing equipment such as desktops and robust internet infrastructure capable of supporting the required programming.**

**Software requirements – The hardware available can support the use of compilers and code editors such as visual studio editors which are needed to code, design, and implement the system.**

**Human requirements – The project team has the technical know how to implement the system.**

## **Economic Feasibility Study**

**Cost-Benefit Analysis: The initial investment for implementing the system including software development, hosting, deployment, hardware procurement, installation and training is estimated at \$820.00. The potential revenue that can be expected to be generated by the system is \$2400, giving a ROI of approximately 192%.**

**Funding Sources: Possible funding sources for the project include personal funds from the project team and internal university funds allocated for service automation and system upgrades.**

## **Legality**

**Legal Requirements: The project aligns with the university's overarching student-cantered policies emphasizing safety and security for all campus members, including staff and visitors. While student-centeredness is a major policy, it seamlessly integrates with various security policies to ensure a holistic approach to campus safety. Data collected and stored by the new system will be securely managed and accessed only by the authorized personnel.**

**Compliance: Regular consultations with the university's legal team will ensure alignment with these policies and other relevant internal standards fostering a comprehensive security strategy for the entire Africa University community.**

## **Schedule Feasibility Study**

**Timeline Assessment:** The project timeline spans 4 months with key milestones.

**Resource Availability:** Adequate human resources including project manager, programmers, database administrators and designers have been allocated to meet the project timeline.

**Risk Mitigation:** Potential risks such as delays in software development or unexpected technical issues have been identified and contingency plans have been developed to mitigate their impact on the project schedule.

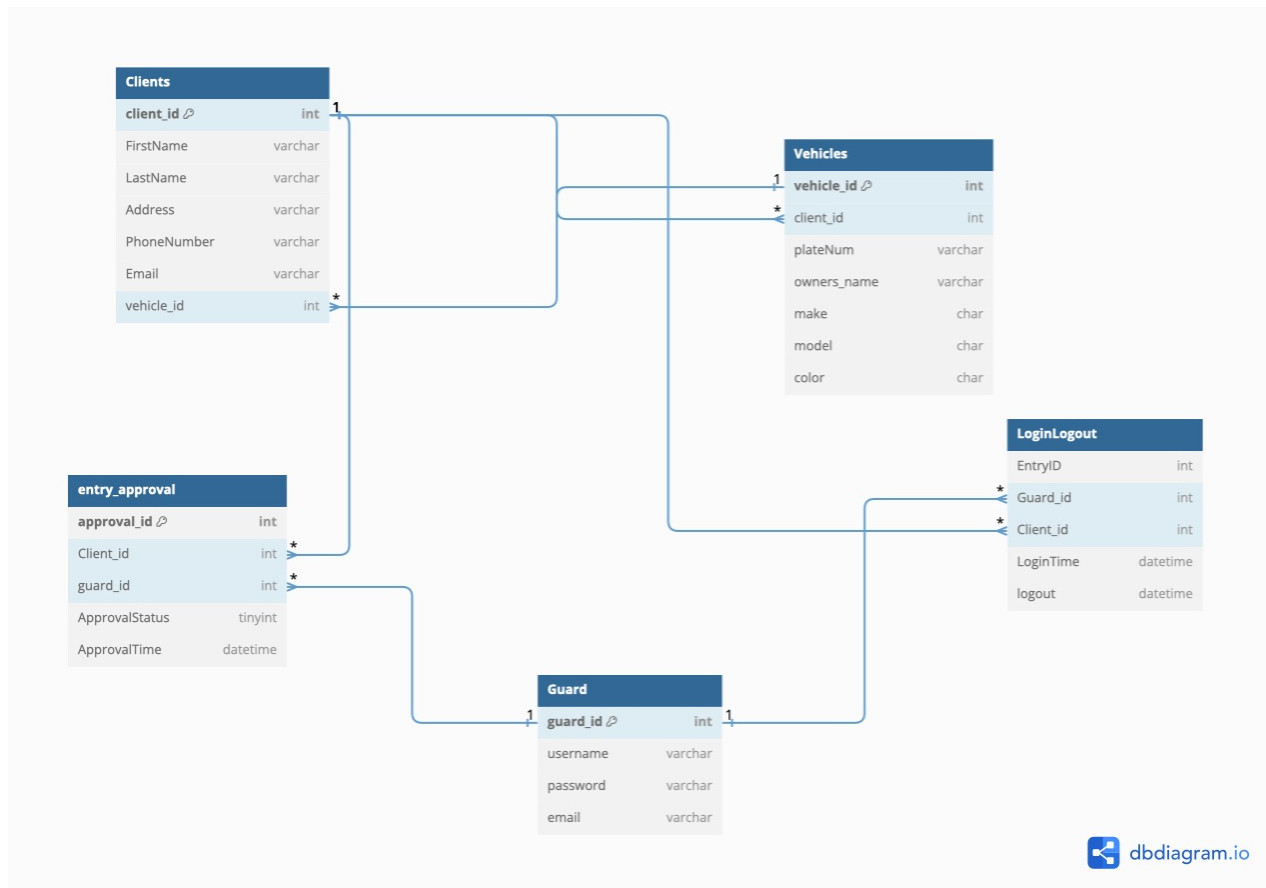
## **Operational Feasibility Study**

**Problem Solving:** Our project addresses operational challenges associated with the existing manual system by enhancing efficiency, improving data security, and reducing errors.

**User Acceptance:** Initial feedback from security personal indicated a positive attitude towards adopting the automated vehicle registration system as it is expected to streamline their workflow and enhance campus security measures.

**Training Needs:** A comprehensive training program will be developed to ensure that security personal is proficient in using the new system. Training sessions will cover system operation, data management procedures and troubleshooting techniques.

## **Database Schema/Erd**



## Implementations

Our system was implemented using the flask framework. Major functionalities of the system include the following.

## 1. Data capture by the camera.

```
website > routes > system.py > ...
193 system.route('/video_feed')
194 def video_feed():
195     # Provide a streaming video feed using the generator function
196     return Response(generate_frames(), mimetype='multipart/x-mixed-replace; boundary=frame')
197
198 system.route('/start_recognition')
199 def start_recognition():
200     global plate_recognition_active
201
202     # Enable the plate recognition process
203     plate_recognition_active = True
204     return "Plate Recognition Started"
205
206 system.route('/stop_recognition')
207 def stop_recognition():
208     global plate_recognition_active
209
210     # Disable the plate recognition process
211     plate_recognition_active = False
212     return "Plate Recognition Stopped"
213
214 system.route('/get_latest_plate')
215 def get_latest_plate():
216     global latest_plate, plate_recognition_active
217     # Check if recognition is active and a plate has been detected
218     if plate_recognition_active and latest_plate:
219         vehicle = Vehicle.query.filter_by(plate_num=latest_plate).first()
220         if vehicle:
221             new_approval = EntryApproval(client_id=vehicle.client_id, guard_id=current_user.guard_id, approval_status=True, approval_time=datetime.now())
222             db.session.add(new_approval)
223
224             # Logout Logic
225             existing_entry = LoginLogout.query.filter_by(
226                 client_id=vehicle.client_id
227             ).order_by(LoginLogout.entry_id.desc()).first() # Get the latest entry
228
229             if existing_entry:
230                 if existing_entry.logout_time is None: # Logout not set
```

The above is a snippet of the code we used to provide a streaming video feed using the generator function as well as enabling and disabling the plate recognition process.

## 2. Look for captured plate in database.

```
def check_plate_in_database(plate_text):
    """Checks if a given license plate number exists in the database."""
    vehicle = Vehicle.query.filter_by(plate_num=plate_text).first()
    # We query the 'Vehicle' model to find a vehicle record where the 'plate_num' column matches the provided 'plate_text'.

    if vehicle:
        # We check if the query returned a matching 'Vehicle' object.
        return True # If a matching vehicle exists, we return True.

    return False # If no matching vehicle is found, we return False.
```

The above is a snippet of the code used to check if a captured licence plate exists in the database. We query the vehicle model to find a vehicle record where the 'plate num' column matched the provided 'plate text'.

## 3. Creating the app

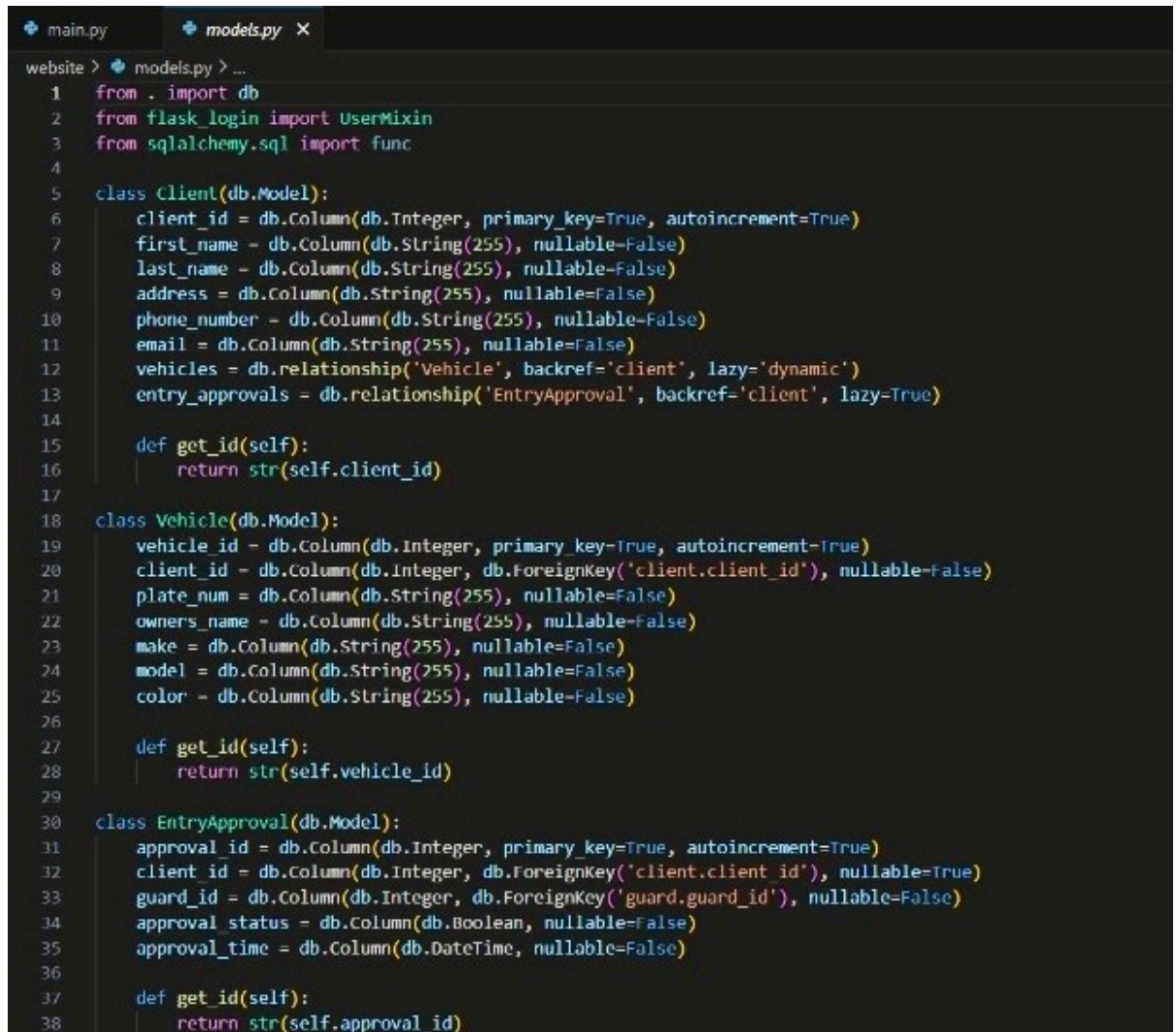
```
main.py > ...
1 from website import create_app, db
2
3 app = create_app()
4
5 if __name__ == "__main__":
6     with app.app_context():
7         db.create_all()
8     app.run(debug=True)
```

The above show the snapshot of the code used to create the flask app.



## 4. Database

Our database was created in the Flask app using SQL Alchemy



```
main.py  models.py X
website > models.py > ...
1  from . import db
2  from flask_login import UserMixin
3  from sqlalchemy.sql import func
4
5  class Client(db.Model):
6      client_id = db.Column(db.Integer, primary_key=True, autoincrement=True)
7      first_name = db.Column(db.String(255), nullable=False)
8      last_name = db.Column(db.String(255), nullable=False)
9      address = db.Column(db.String(255), nullable=False)
10     phone_number = db.Column(db.String(255), nullable=False)
11     email = db.Column(db.String(255), nullable=False)
12     vehicles = db.relationship('Vehicle', backref='client', lazy='dynamic')
13     entry_approvals = db.relationship('EntryApproval', backref='client', lazy=True)
14
15     def get_id(self):
16         return str(self.client_id)
17
18     class Vehicle(db.Model):
19         vehicle_id = db.Column(db.Integer, primary_key=True, autoincrement=True)
20         client_id = db.Column(db.Integer, db.ForeignKey('client.client_id'), nullable=False)
21         plate_num = db.Column(db.String(255), nullable=False)
22         owners_name = db.Column(db.String(255), nullable=False)
23         make = db.Column(db.String(255), nullable=False)
24         model = db.Column(db.String(255), nullable=False)
25         color = db.Column(db.String(255), nullable=False)
26
27         def get_id(self):
28             return str(self.vehicle_id)
29
30     class EntryApproval(db.Model):
31         approval_id = db.Column(db.Integer, primary_key=True, autoincrement=True)
32         client_id = db.Column(db.Integer, db.ForeignKey('client.client_id'), nullable=True)
33         guard_id = db.Column(db.Integer, db.ForeignKey('guard.guard_id'), nullable=False)
34         approval_status = db.Column(db.Boolean, nullable=False)
35         approval_time = db.Column(db.DateTime, nullable=False)
36
37         def get_id(self):
38             return str(self.approval_id)
```

Our App was designed was various features including:

### 1. Report generation.

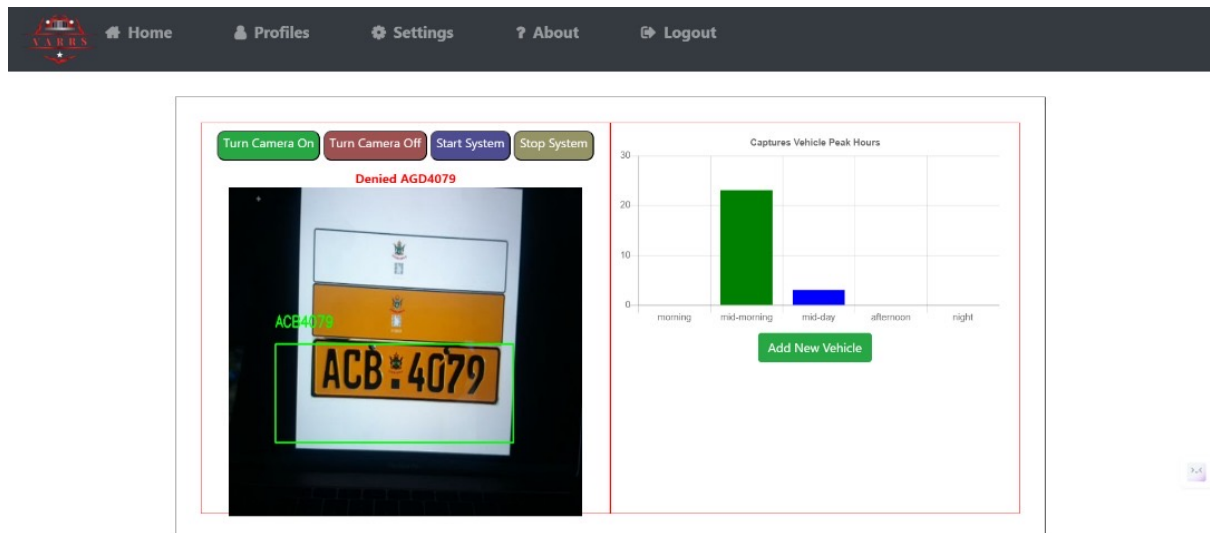
```
main.py reports.html X
website > templates > reports.html
39 {% block content %}
55 <table class="table">
56 <thead>
57 <tr>
58 <th>Entry ID</th>
59 <th>Client ID</th>
60 <th>Guard ID</th>
61 <th>Login Time</th>
62 <th>Logout Time</th>
63 <th>Duration</th>
64 </tr>
65 </thead>
66 <tbody>
67 {% for entry in entries %}
68 <tr>
69 <td>{{ entry.entry_id }}</td>
70 <td>{{ entry.client_id }}</td>
71 <td>{{ entry.guard_id }}</td>
72 <td>{{ entry.login_time }}</td>
73 <td>{{ entry.logout_time or '-' }}</td>
74 <td>
75     {% if entry.logout_time %}
76     {% set duration_seconds = (entry.logout_time - entry.login_time).total_seconds() %}
77     {% set hours = duration_seconds // 3600 %}
78     {% set minutes = (duration_seconds % 3600) // 60 %}
79     {{ hours }}h {{ minutes }}m
80     {% else %}
81     -
82     {% endif %}
83 </td>
84 </tr>
85 {% endfor %}
86 </tbody>
87 </table>
88 <div class="report-actions">
89 <button type="button" onclick="printReport()">Print</button>
90
91 </div>
92
93 {% endblock %}
```

Using html and python we were able to add this functionality to the app.

## Post Implementation

Expected - License plate recognition system implementation.

Actual – We were able to create a license plate recognition system that is fully operational.



The image above shows the camera scanning a license plate, but it does not recognize as it is not in the system.

**Expected –** System should be able to add new clients, update their existing profiles and delete old clients.

**Actual –** The system can create profiles for new clients in the database and update those profiles if need be as well as deleting the profiles both for users i.e. Guards and clients i.e. students, staff etc.

First Name  
[Redacted]

Last Name  
[Redacted]

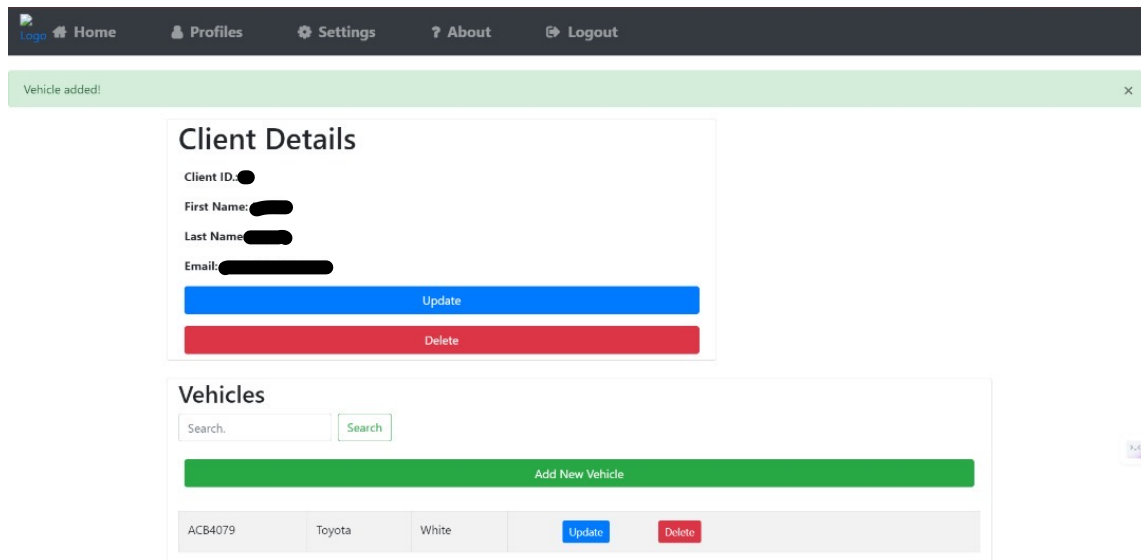
Address  
[Redacted]

Phone Number  
[Redacted]

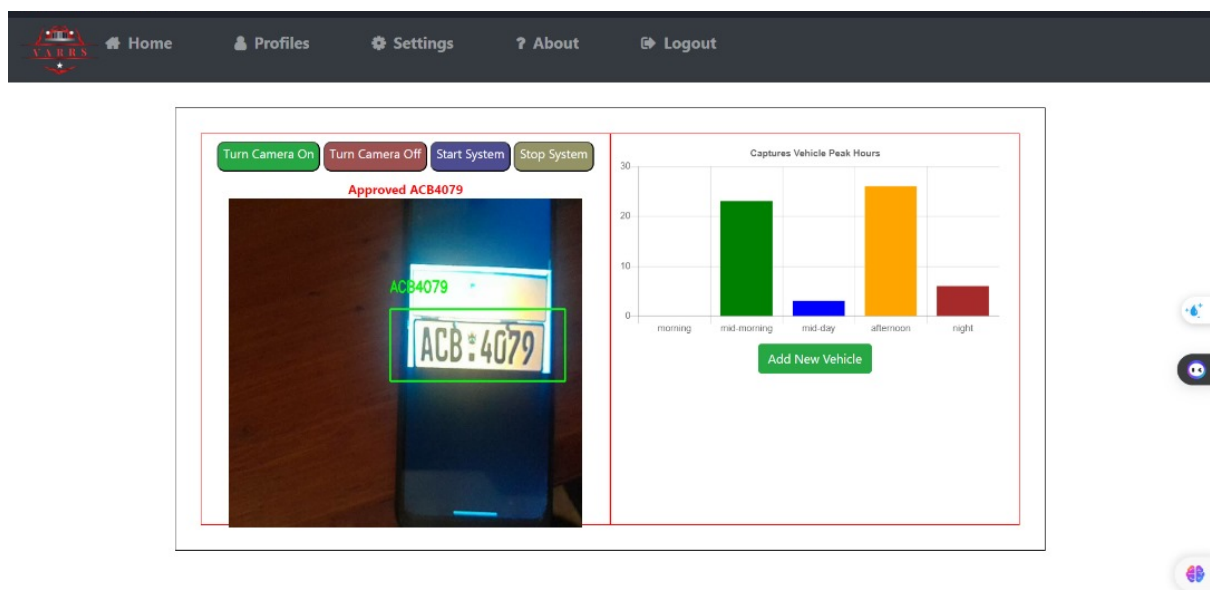
Email  
[Redacted]

[Submit](#)

The image above shows a new client being added into the database.




The above image shows a client being successfully added with a notification being used to communicate this.



The above image now shows the license plate being recognized as it is now in the database.

**Expectation** – Guard on duty or supervisor to be able to login and out into the system as well as sign into it securely.

**Actual** – Our system does allow the user to log into it or sign up if it's there first time using, and user is able to select role either guard or supervisor.

 Login Sign Up

Incorrect password, try again. ×

Login

Select Role: Guard ▼

Email Address




Enter email

Password

Enter password

Login

Forgot Password?



The guard is also able to request for a password change is the forget.




Login Sign Up

Reset Your Password

Enter your registered email address to receive a password reset link.

Email:

Send Reset Link



Inside our app we have include some functionalities including:

- Log in /out Report.

## Login/Logout Report

Start Date: dd/mm/yyyy

End Date: dd/mm/yyyy

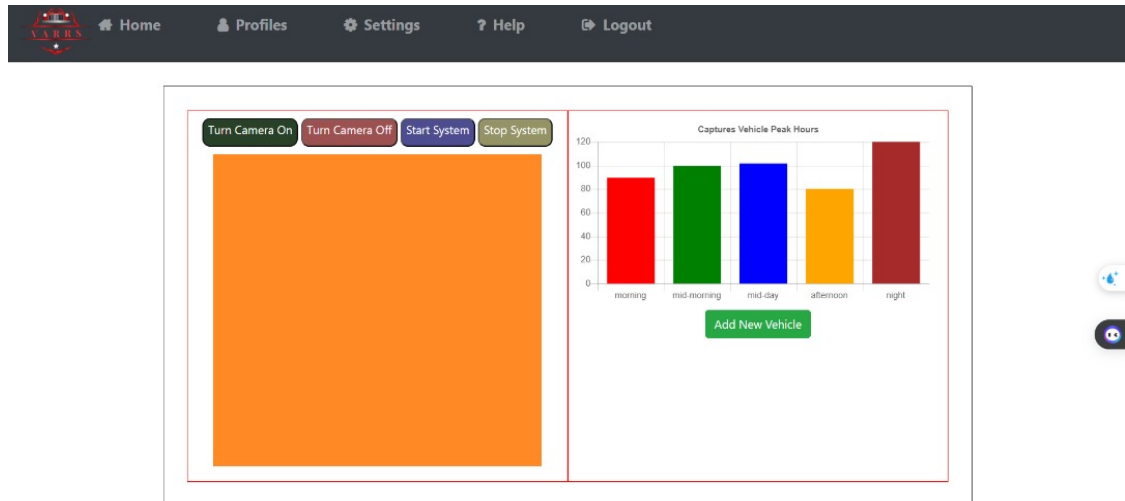
Filter

Entry ID	Client ID	Guard ID	Login Time	Logout Time	Duration
1	1	3	2024-04-02 09:46:32	2024-04-02 09:46:33	0.0h 0.0m
2	1	3	2024-04-02 09:46:35	2024-04-02 09:46:36	0.0h 0.0m
3	1	3	2024-04-02 09:46:38	2024-04-02 09:46:39	0.0h 0.0m
4	1	3	2024-04-02 09:46:41	2024-04-02 11:05:38	1.0h 18.0m
5	2	3	2024-04-02 09:53:34	2024-04-02 09:53:34	0.0h 0.0m
6	2	3	2024-04-02 09:53:34	2024-04-02 09:53:34	0.0h 0.0m
7	2	3	2024-04-02 09:53:36	2024-04-02 09:53:48	0.0h 0.0m
8	2	3	2024-04-02 09:53:51	2024-04-02 10:25:44	0.0h 31.0m
9	2	3	2024-04-02 10:27:05	2024-04-02 10:27:08	0.0h 0.0m
10	2	3	2024-04-02 10:29:44	2024-04-02 10:43:46	0.0h 14.0m
11	2	3	2024-04-02 10:43:48	2024-04-02 10:43:49	0.0h 0.0m
12	1	3	2024-04-02 11:08:09	2024-04-02 14:49:47	3.0h 41.0m

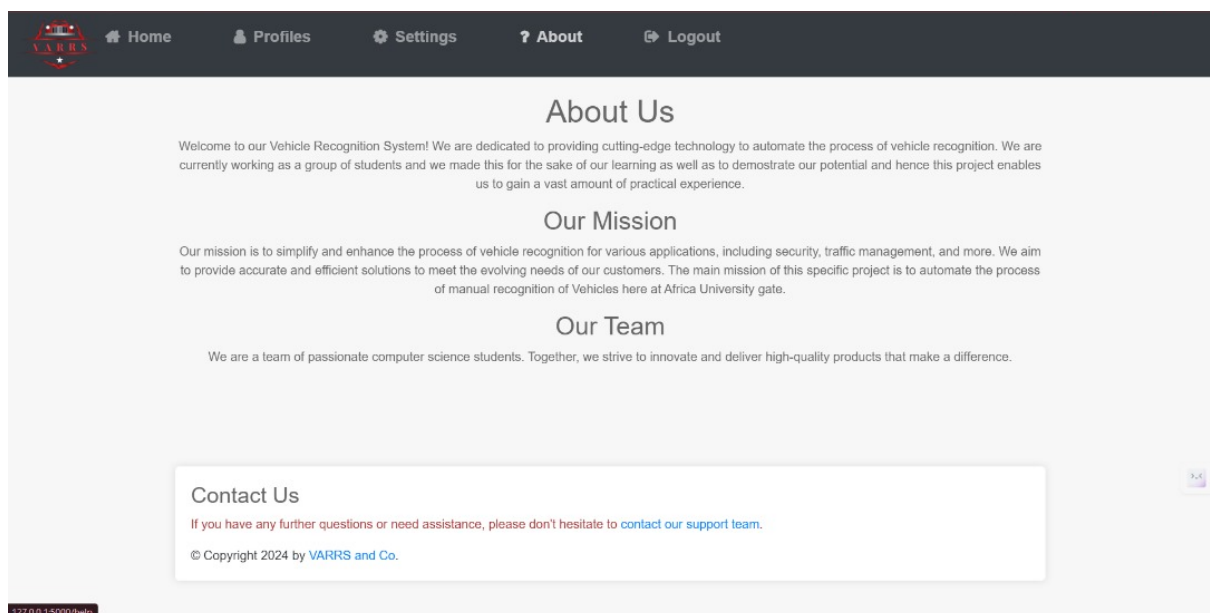
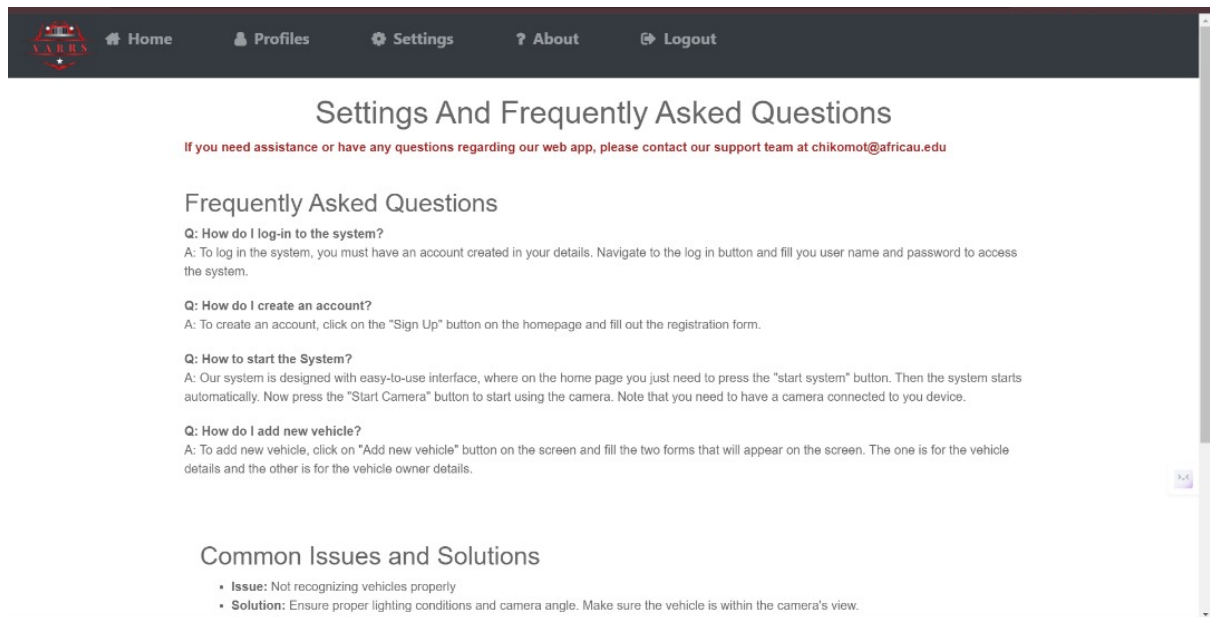
This functionality allows users to generate reports on the log in and out of vehicles thus can determine how long vehicles have stayed inside the campus for security and accountability.

### b. Representing car movement on a bar chart

The system can show car movement in and out of campus in form of a bar chart for statistical purposes.



The App also has help and information pages.



These pages allow users to get help and to know more about the system and its uses.

## Security Implementations

To enhance security in our system we implemented the following security features.

### 1. Email Confirmations

As a user is signing into the system, they are required to submit their email address and upon completing the sign-up process and email is sent to their mail to communicate that they have signed up.



## Sign Up

Email Address

Enter email

Username

Enter Username

Password


Enter password

Password (Confirm)

Confirm password

Submit

## 2. Log Files and Reports



Home

Profiles

Accounts


Settings

About

Logout

### Application Log

10-Apr-24 14:27:33 - This is a test  
10-Apr-24 14:27:36 - Neither CUDA nor MPS are available - defaulting to CPU. Note: This module is much faster with a GPU.  
10-Apr-24 14:27:37 - [31m][WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server inst  
\* Running on http://127.0.0.1:5000  
10-Apr-24 14:27:37 - [33mPress CTRL+C to quit]0m  
10-Apr-24 14:27:37 - \* Restarting with stat  
10-Apr-24 14:27:38 - This is a test  
10-Apr-24 14:27:40 - Neither CUDA nor MPS are available - defaulting to CPU. Note: This module is much faster with a GPU.  
10-Apr-24 14:27:42 - \* Debugger is active!  
10-Apr-24 14:27:42 - \* Debugger PIN: 768-865-340  
10-Apr-24 14:27:42 - 127.0.0.1 - - [10/Apr/2024 14:27:42] "GET /logs HTTP/1.1" 200 -  
10-Apr-24 14:27:42 - 127.0.0.1 - - [10/Apr/2024 14:27:42] "[36mGET /static/style.css HTTP/1.1]0m" 304 -  
10-Apr-24 14:27:42 - 127.0.0.1 - - [10/Apr/2024 14:27:42] "[36mGET /static/logo.png HTTP/1.1]0m" 304 -  
10-Apr-24 14:27:44 - 127.0.0.1 - - [10/Apr/2024 14:27:44] "GET /log\_stream HTTP/1.1" 200 -  
10-Apr-24 14:27:57 - 127.0.0.1 - - [10/Apr/2024 14:27:57] "[33mGET /highlightTitle.png HTTP/1.1]0m" 404 -  
10-Apr-24 14:27:59 - 127.0.0.1 - - [10/Apr/2024 14:27:59] "GET / HTTP/1.1" 200 -  
10-Apr-24 14:27:59 - 127.0.0.1 - - [10/Apr/2024 14:27:59] "[36mGET /static/style.css HTTP/1.1]0m" 304 -  
10-Apr-24 14:27:59 - 127.0.0.1 - - [10/Apr/2024 14:27:59] "[36mGET /static/logo.png HTTP/1.1]0m" 304 -  
10-Apr-24 14:27:59 - 127.0.0.1 - - [10/Apr/2024 14:27:59] "[36mGET /static/styles.css HTTP/1.1]0m" 304 -  
10-Apr-24 14:27:59 - ('morning': 0, 'midmorning': 0, 'midday': 0, 'afternoon': 0, 'night': 0)  
10-Apr-24 14:27:59 - 127.0.0.1 - - [10/Apr/2024 14:27:59] "GET /get chart data HTTP/1.1" 200 -  
10-Apr-24 14:28:03 - 127.0.0.1 - - [10/Apr/2024 14:28:03] "[33mGET /highlightTitle.png HTTP/1.1]0m" 404 -  
10-Apr-24 14:28:05 - 127.0.0.1 - - [10/Apr/2024 14:28:05] "GET /clients HTTP/1.1" 200 -  
10-Apr-24 14:28:06 - 127.0.0.1 - - [10/Apr/2024 14:28:06] "[36mGET /static/style.css HTTP/1.1]0m" 304 -  
10-Apr-24 14:28:06 - 127.0.0.1 - - [10/Apr/2024 14:28:06] "[36mGET /static/logo.png HTTP/1.1]0m" 304 -  
10-Apr-24 14:28:07 - 127.0.0.1 - - [10/Apr/2024 14:28:07] "[33mGET /highlightTitle.png HTTP/1.1]0m" 404 -  
10-Apr-24 14:28:16 - 127.0.0.1 - - [10/Apr/2024 14:28:16] "GET / HTTP/1.1" 200 -  
10-Apr-24 14:28:17 - 127.0.0.1 - - [10/Apr/2024 14:28:17] "[36mGET /static/style.css HTTP/1.1]0m" 304 -  
10-Apr-24 14:28:17 - 127.0.0.1 - - [10/Apr/2024 14:28:17] "[36mGET /static/logo.png HTTP/1.1]0m" 304 -



Home

Profiles

Accounts

Settings

About

Logout

### Login/Logout Report

Start Date:

dd/mm/yyyy

End Date:

dd/mm/yyyy

FilterClear


Entry ID	Client: Full Name	Guard : Username	Plate Number	Login Time	Logout Time	Duration
----------	-------------------	------------------	--------------	------------	-------------	----------

Print



Our system has the ability to create application log files and generate reports of logins and logouts which are accessible to the supervisor only.

### 3. Different Levels Of Accessibility



[Login](#) [Sign Up](#)

Login

Select Role: 

Supervisor

Guard


Supervisor

Email Address:

Password:

Login

[Forgot Password?](#)




[Home](#) [Profiles](#) [Accounts](#) [Settings](#) [About](#) [Logout](#)

Logged in successfully!

Manage Guard Accounts

Username	Email	Actions
		<a href="#">Delete</a>
		<a href="#">Delete</a>
		<a href="#">Delete</a>



[Home](#) [Profiles](#) [Accounts](#) [Settings](#) [About](#) [Logout](#)

Login/Logout Report

Start Date: 

dd/mm/yyyy

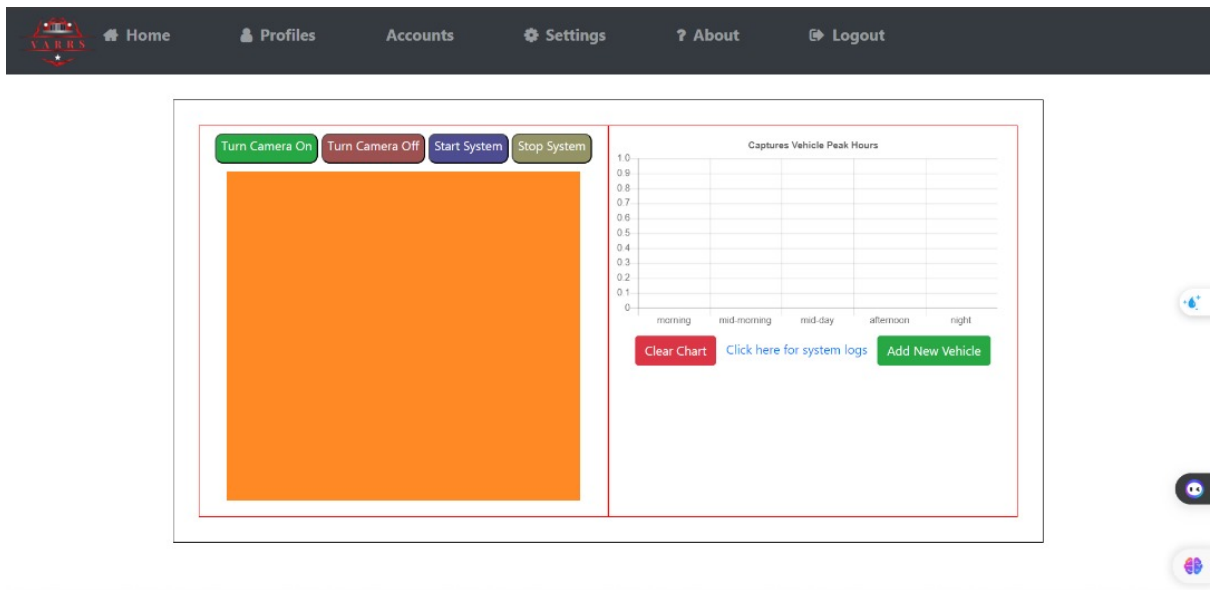
End Date: 

dd/mm/yyyy

[Filter](#) [Clear](#)

Entry ID	Client: Full Name	Guard : Username	Plate Number	Login Time	Logout Time	Duration
----------	-------------------	------------------	--------------	------------	-------------	----------

[Print](#)



To ensure secure handling of information our system has an interface for guards and an interface for a supervisor. The supervisor interface can view log files, delete guard account, and clear reports while guards do not. In the interface of the supervisor there is also a link that can take the supervisor to the system log files thus allowing them to see the activities of the guards in the system.

## **Recommendation**

**User Feedback:** We as VARRS prioritise user feedback and one of the recommendations to the IT team is to continue understanding the customer's needs, pain points, and suggestions for software improvement. This can be done through surveys, user interviews, or feedback forms within the software itself. Analyse the feedback and use it as a basis for identifying areas of improvement.

**Performance Optimization:** We recommend continued software's performance analysis to identify any bottlenecks or areas where it can be optimized. This may involve profiling the code, identifying resource-intensive operations, and finding ways to optimize algorithms or data structures. Improving performance can enhance the user experience and make the software more efficient.

**Bug Fixes:** Continuously address and fix bugs reported by users or discovered through testing. Prioritize critical or high-impact issues that affect the functionality or stability of the software. Regular bug fixes help maintain the quality and reliability of the software.

**Security Enhancements:** Assess the software's security posture and implement measures to enhance security. This may involve conducting security audits, addressing vulnerabilities or weaknesses, and implementing secure coding practices. Regularly update dependencies and libraries to ensure they are free from security vulnerabilities.

**Feature Enhancements:** Identify new features or improvements that can enhance the software's functionality and meet user requirements. Consider user feedback, market trends, and competitive analysis to prioritize and plan new feature development. Regularly release updates with new features to deliver value to users.