# Chapter 3 Methodology

This chapter provides a detailed account of the design process and the decisions that were made during the project. The aim of this chapter is to provide a comprehensive understanding of the design process, the methods used, and the rationale behind the decisions made. In this chapter, the design methodology, the design requirements, the design process, and the final design solution are presented and discussed. The is made up of mainly 3 parts, the web application, vehicle hardware and a database. Figure 1 shows the breakdown of the system. The website consists of authentication functionality which is responsible for user registration and login, vehicle management system where users can register their vehicles and authorise drivers to us the vehicles. The owners of the vehicles can add and remove authorised at will. The system also has a vehicle tracking system where its location can be tracked and its state whether it’s being driven or its stationary. The last component of the website a functionality to view images taken by the camera system of the vehicle.

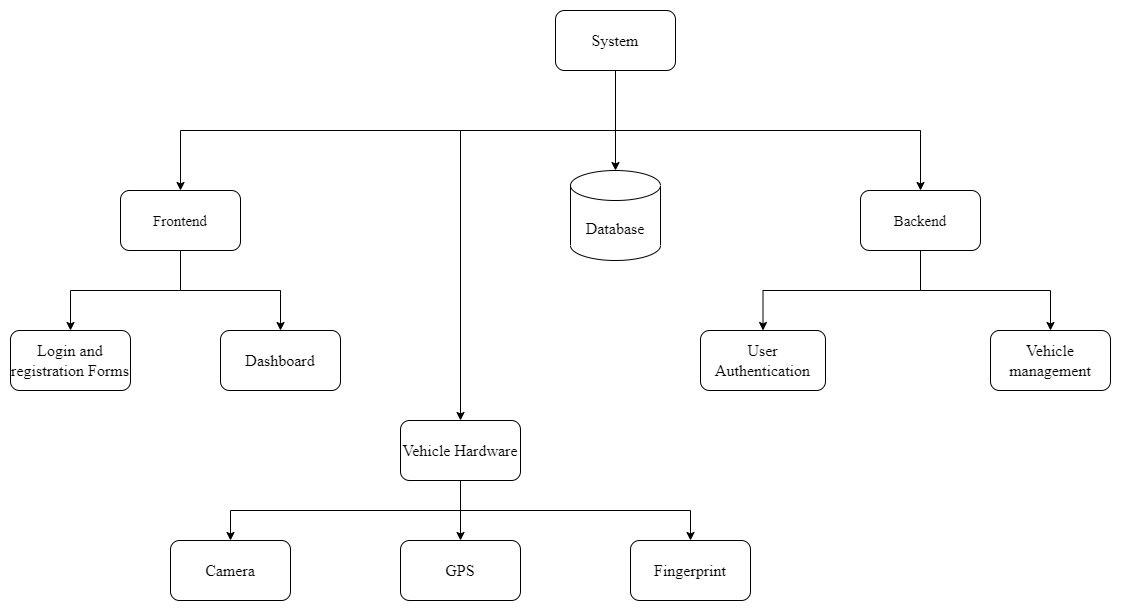


Figure 1.1 System Block Diagram

The vehicle hardware is a small microcomputer system which tracks the car’s location, sending alerts to the authorised drivers when in emergency. The microcomputer has a fingerprint scanner and a camera which is used for biometric authentication. Figure 1.2 shows the breakdown of the system.

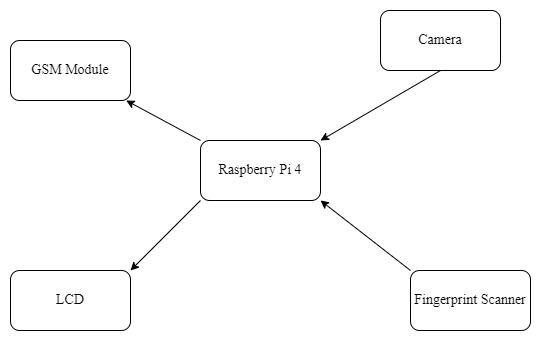


Figure 1.2

The database is used to store the system data for persistence. It stores data like user information which is username, email, phone number and encrypted password. It also stores vehicle information which is vehicle registration number and a list authorised driver in form of system user ids.

## Vehicle hardware design

The hardware of the system is centred around a raspberry pi single board computer. It has enough processing power to run a camera for facial recognition, a fingerprint scanner, GPS module, GSM module and a 16x2 character LCD. Figure shows the block diagram of the hardware. The raspberry pi has a connector for its special camera which is accelerated by the GPU for a better performance compered to other computers that uses USB to connect a camera.

The fingerprint scanner, GPS module and the GSM module connect to the raspberry pi using UART communication. Only the raspberry pi 4 has enough UART ports (6) to support these modules. Other version of raspberry pi single board computer needs a USB to UART convertor module to support this system. The python programming language was chosen as the programming language of choice to develop this hardware system as python language is easy to use and is widely supported in the raspberry pi community.

The python programme running on the raspberry pi uses threads to execute the designed program smoothly. The program has two threads one to update the vehicle’s location every 30 seconds and one to listen for events such as when the start button is pressed or for a fingerprint registration request.

### MQTT Client

On the main script which is the entry file of the program an MQTT client is initialised and is passed to classed that need it such as GPS class to update the vehicle’s location every 30 seconds. When the client connects to a broker a function is ran which subscribe to a fingerprint registration topic as shown in figure 1.2. When a message is received to register a fingerprint, a function is called to deal with the request, which is basically enrolling a fingerprint. A client thread is then started to allow the remaining of the code to run.



Figure 1.3 On connect and on message functions

When a fingerprint registration function is received the driver id and their phone numbers are stored in variables and a fingerprint enrolment function is ran which returns a number which is the location on which the fingerprint template is stored on the scanner or a Boolean value of false. If the enrolment is successful the driver id, fingerprint id and the driver’s phone number are stored in a local database and the backend server is updated by publishing an MQTT message that contains the driver id, licence number of that vehicle and the fingerprint id.

### Fingerprint

The fingerprint module has two main functionalities which are enrolling of fingerprints and searching for a fingerprint match. When enrolling the fingerprint, the user has to scan the same finger twice. The fingerprint scanner firsts capture a fingerprint image, if it can identify the features it creates a template and stores it in a variable. On the second scan it does the same, then it compares the two templates if they are the same it creates a model and stores it in the scanner’s internal memory location specified by the user. Figure shows the enrolling process.

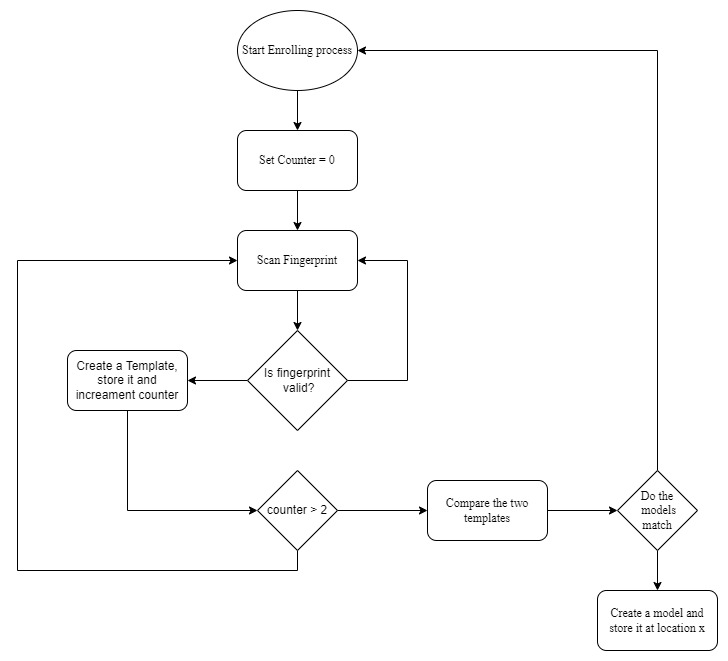


Figure 1.4

To search for a fingerprint the user scans their fingerprint then the scanner captures a photograph of the fingerprint. It then checks if it can extract fingerprint features from the photograph and goes on to search for a matching fingerprint in its memory. If a matching fingerprint is found its location and a confidence level are returned.

### GPS

The GPS module communicates with the raspberry pi using UART communication. The module once connected to GPS satellites it sends NMEA messages every second which contains various information such as the module’s location coordinates, time and altitude. The program captures the NMEA string and extract the location coordinates every second and store them in class variables and every 30 seconds the coordinates are published to an MQTT broker under a “location” topic with the licence plate number being the sub topic. Figure 1.3 shows the code how the program updates the location of the vehicle.



Figure 1.5

### Camera

The vehicle’s camera system is activated after the driver has been authenticated with their fingerprint. It captures images of the driver every 30 seconds and sends them to the backend server using MQTT. The images are then saved in a folder whose name is the same as the vehicle’s licence plate number.

### WhatsApp messages

The raspberry pi system sends alert messages using WhatsApp massaging. The system sends messages when the driver presses a panic button or when an unrecognised fingerprint is scanned more than two times.

## Database design

The database is one of the main components. This is where all the information about the system is stored. The database consists of 3 collections, users collection, vehicles collection and face descriptors collection. The users collection contains documents that have information that deals with users of the system such as name, username, email, phone number and encrypted password. Figure shows the user’s location schema.



Figure 1.7 User Schema

The vehicles schema contains information about the vehicle. The information contained includes the vehicle make, model and licence plate number. The vehicle schema also stores a list of authorised drivers. The list of authorised drivers has its own schema, it has the driver’s id and the fingerprint id as shown in figure 1.8.



Figure 1.8



Figure 1.9

## Web application design

A web application is a software application that is accessed through a web browser over the internet. It consists of two main components - the backend and the frontend. The backend is the part of the application that runs on the server and is responsible for processing requests from clients, such as web browsers or mobile devices. It typically uses a server-side programming language, such as Python, Java, PHP Node JS, and interacts with a database to store and retrieve data.

On the other hand, the frontend is the part of the application that users interact with directly. It is the user interface that is presented in the web browser and is responsible for rendering the visual elements, such as text, images, and buttons. The frontend communicates with the backend through APIs (Application Programming Interfaces) to retrieve data and perform actions on behalf of the user.

In order to create a successful web application, both the backend and frontend must be designed and developed in a way that is efficient, secure, and user-friendly. The design and implementation of the backend should be carefully considered, taking into account factors such as scalability, security, and performance. Similarly, the frontend should be designed with the user in mind, with a focus on creating an intuitive and visually appealing user interface that is accessible across different devices and browsers.

Overall, a well-designed and implemented web application should provide a seamless user experience, with a responsive and intuitive user interface, and a robust and secure backend that can handle a large volume of requests and data processing.

Node JS was chosen to develop the backend because of its asynchronous nature since most of the requests are I/O bound and Node JS by default is non-blocking. Node JS connects well with MongoDB which is NoSQL database and stores information in form of Binary JSON (JavaScript Object Notation) documents. The frontend was developed using React JS which is JavaScript frontend development framework. It uses a JSX notation where JavaScript code is mixed with HTML tags. It uses reusable components which makes the design and development of the user interface simple and understandable by grouping related code together.

### User Registration

The system needs users in order to function. To get users’ information the system uses user registration functionality. User registration is a process in which new users can create an account or profile on a website, application, or system. The main purpose of user registration is to provide a way for the system to identify and authenticate users, and keep track of which user is authorised to use which vehicle. A user clicks user registration link on the landing page and is redirected to a registration page. The user has to provide their name, username of choice, email, phone number and a secure password of their choice. After completion of the form the user sends the information to a backend server by clicking the Register button at the bottom of the page. At the backend server the information is validated. The server checks if there is a user already using the provided username and email, if they do an error message is sent back to the user notifying them that those two pieces of information are already in use in the system. Other wise the user is registered in the system by encrypting their password first using bcrypt algorithm and saving the information in the database.

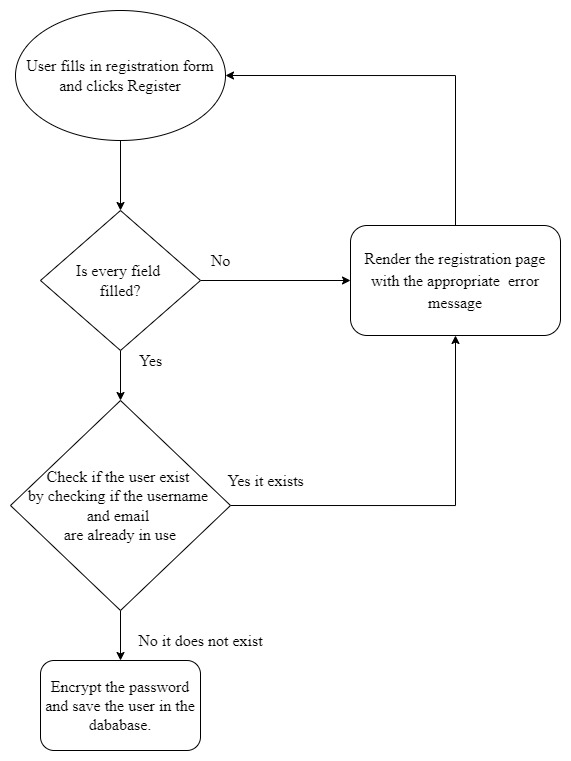


Figure 1.11 Registration Logic

Figure 1.1 shows the logic behind registration process.

### User Authentication

User authentication is the process of verifying the identity of a user, typically for the purpose of granting them access to the web application. The user provides a username and password by filling a login form in the login page. The user clicks the login button after entering the username and password and the information to a server for verification. The server first queries the database using the username to check if the user has been registered. If the user is found it is stored in a variable and the stored encrypted password is compared with the password entered by the user. If the passwords match, the server creates a JSON web token using the user’s id and protects the integrity of the token using a secret stored on the server. The token is set as an http only cookie and expires after seven days. The user is the redirected to the dashboard page.

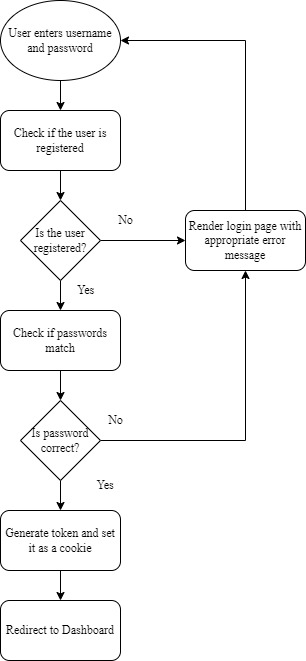


Figure 1.13

This token is then stored on the user’s browser and the user does not need to re-enter their username and password when they revisit the web application. The process of user authentication is shown in figure 1.3. The token is also used to identify the user whenever a request is made by the frontend to the backend component of the application.

### Vehicle Registration

A user can register their vehicle in the system. They have to enter the vehicle licence number, the make and the model, after vehicle registration the information is stored in the database and the user is set as the owner and is immediately authorised to drive that vehicle. To complete the process, the user has to register their biometrics on the vehicle.

### Driver Authorisation

The owner of a vehicle can authorise other users of the system to drive their vehicle by adding them to the list of authorised drivers. To authorise a driver the owner can enter the username of the user they intend to authorise. Once added the user gets a notification on their profile. In order for the new driver to drive the vehicle they have to register their biometrics on the vehicle’s system.

### Vehicle Monitoring

The system can monitor location of the vehicle and display its location on a map on the web application. The vehicle will keep publishing its location at regular intervals while the web application receives the location coordinates and update a marker on the map that represent the vehicle’s location.