Contents

[Project Requirements 2](#_Toc153645223)

[Purpose of the Project: 2](#_Toc153645224)

[Expected Outcomes: 2](#_Toc153645225)

[Specific Constraints or Limitations: 2](#_Toc153645226)

[Project Plan 2](#_Toc153645227)

[Scope of the Project: 2](#_Toc153645228)

[Source Code: 3](#_Toc153645229)

[Database Schema: 3](#_Toc153645230)

[Table Definitions: 3](#_Toc153645231)

[User Documentation: 4](#_Toc153645232)

[Application Access: 4](#_Toc153645233)

[Login Credentials: 5](#_Toc153645234)

[Navigation: 5](#_Toc153645235)

[Technical Documentation: 6](#_Toc153645236)

[Architecture: 6](#_Toc153645237)

[Implementation: 6](#_Toc153645238)

[Relevant Technical Details: 6](#_Toc153645239)

[Gallery 7](#_Toc153645240)

# Project Requirements

## Purpose of the Project:

This is an internal firmware application meant to register all clients that come to the carwash to have their cars washed, display them, and keep records of them. The clients receive emails upon booking an appointment informing them that they have registered successfully.

### Expected Outcomes:

1. **Record Keeping**: Employees should be able to keep a clear record of all the clients that have passed through the car wash.
2. **Client verification**: Accurate recording of clients and email verification to verify that the information is accurate.
3. **Enhanced Security**: Implementation of a robust authentication system to ensure data integrity.
4. **User-Friendly Interface**: An intuitive interface for easy navigation and use by police personnel.

### Specific Constraints or Limitations:

* **Time Limitations**: Completion is expected within 3 weeks.
* **Technology Constraints**: The application will be developed using Java, Spring Boot, Thymeleaf, and MySQL.

# Project Plan

## Scope of the Project:

1. **Client Management:**

- Addition of new client with details.

- Handling returns of clients.

- Keeping a log of all clients.

# Source Code:

The source code for the project is hosted on GitHub: [Insert Link].

Key Files:

- `ClientController.java`: Handles HTTP requests related to clients.

- `LoginController.java`: Manages applications-related operations.

# Database Schema:

## Table Definitions:

1. **Gun Table:**

-id (Primary Key)

-appointment\_date

-car\_model

-client\_email

-client\_name

\_plate\_number

-price

2. **User Table:**

- id (Primary Key)

- email

-name

-password

# User Documentation:

## Application Access:

The application is accessible at [Insert URL]. Use the provided login credentials to access the system.

## Login Credentials:

- **Email:** [sample username]

- **Password:** [sample password]

## Navigation:

1. **Add New Client**

- Navigate to dashboard.

- Fill in client Name, client Email, client Name,plate Number, price, car model.

- Click "submit" to add a new client.

Note: Email will be sent to the Client with the login info as the body of the email.

Password will be provided if he/she is allowed to access the system.

1. **Add New Weapons:**

- Navigate to "Add New Weapons."

- Fill in manufacturer, model, type, serial number, purchase date, and registration number.

- Click "Save" to add a new weapon.

Note: Added a Combo box of information mostly common to all guns to make the task easy for the users.

1. **Assign Weapons:**

- Navigate to "Assign Weapons."

- Select Gun and Police Officer then select the date.

- Click "Save" to add a new request.

Note: Now the police can search and see that he/she has been assigned a weapon.

2. **Return Clients:**

- Immediately after registering the client, you’ll see the full list of clients.

2. **Search:**

- Go to "Search" on the index or on Navbar on the View

- Enter client Name the search the information and will display all information related to that clients.

# Technical Documentation:

## Architecture:

The application follows a three-tier architecture:

1. **Front-end:** Thymeleaf for server-side rendering.

2. **Back-end:** Java and Spring Boot for RESTful APIs.

3. **Database:** MySQL for data storage.

## Implementation:

1. **Back-end:**

- Spring Security for user authentication.

2. **Front-end:**

- Bootstrap for responsive and user-friendly design.

3. **Database:**

- MySQL for data storage.

- Foreign key relationships ensure data integrity.

## Relevant Technical Details:

- **Security**: Spring Security ensures secure user authentication.

- **Communication**: RESTful APIs enable seamless communication between front-end and back-end.

# Gallery



