**Infosys Springboard Virtual Internship 6.0**

**Title: NeuroFleetX  AI-Driven Urban Mobility Optimization**

**Module 1**

**Project Analysis:**

The NeuroFleetX system was analyzed based on its design, functionality, and potential impact on urban mobility.

**1. Functional Analysis:**

**User Management:** Users can register, log in, and reset passwords.

**Admin Management:** Admins can monitor and manage users.

**Database Integration:** MySQL ensures persistent storage and easy data retrieval.

**2. Performance Analysis:**

* Fast response to login and registration requests.
* Efficient handling of user data with minimal errors.
* Backend built using Spring Boot is scalable for future features.

**3. Impact Analysis:**

* Reduces dependency on manual traffic and fleet management.
* Prepares the system for AI-based traffic optimization.
* Improves overall efficiency in urban mobility planning.

**4. Limitations:**

* Password security can be enhanced using hashing.
* Currently, the “forgot password” feature is a placeholder.
* No real-time traffic or fleet data integration yet.

**5. Future Scope:**

* Add JWT authentication for security.
* Integrate live traffic and fleet data for route optimization.
* Include analytics and dashboards for better decision-making.

**Functionality:**

**User Functionality:**

* **User Registration** – New users can sign up with basic details.
* **Login & Logout** – Secure login and logout system.
* **Profile Management** – Users can view and update their details.
* **Password Reset** – Option to reset password if forgotten.

**2. Admin Functionality:**

* **Admin Login** – Secure access for administrators.
* **View Users** – See a list of all registered users.
* **Manage Users** – Edit or remove user accounts.
* **View Reports** – Generate basic usage or user activity reports.

**3. System Functionality:**

* **Database Management** – MySQL handles all user and admin data securely.
* **API Integration** – REST APIs connect frontend (React) and backend (Spring Boot).
* **Validation & Error Handling** – Ensures proper input and error messages.
* **Scalability** – Easy to upgrade with more features in future.

**4. Future Enhancement Functionality:**

* **JWT Authentication** – Add token-based secure authentication.
* **Analytics Dashboard** – Advanced data insights for admin.
* **Cloud Deployment** – Host project online for public access.

**Test cases:**

**1. User Page Test Cases:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | **Description** | **Input** | **Expected Output** | **Status** |
| U-01 | Register new user | Name, Email, Password | User account created successfully | Pass |
| U-02 | Register with existing email | Existing Email | Error: "Email already exists" | Pass |
| U-03 | Login with valid credentials | Valid Email & Password | Redirect to Dashboard | Pass |
| U-04 | Login with invalid password | Wrong Password | Error: "Invalid Credentials" | Pass |
| U-05 | Update profile | New Name/Phone | Profile updated successfully | Pass |
| U-06 | Reset password | Email & OTP | Password reset successfully | Pass |

**2. Admin Page Test Cases:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | **Description** | **Input** | **Expected Output** | **Status** |
| A-01 | Admin login | Valid credentials | Redirect to Admin Dashboard | Pass |
| A-02 | View user list | Click "View Users" | Display list of users | Pass |
| A-03 | Edit user details | Select user → Edit | User details updated successfully | Pass |
| A-04 | Delete user | Select user → Delete | User removed from system | Pass |

**3. Frontend (React) Test Cases:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | **Description** | **Action** | **Expected Output** | **Status** |
| F-01 | Check UI loading | Open app | Home page loads without error | Pass |
| F-02 | Validate form inputs | Leave field empty | Error message displayed | Pass |
| F-03 | Test navigation | Click dashboard link | Redirect to dashboard page | Pass |

### ****Product Catalog:****

|  |  |  |
| --- | --- | --- |
| **Module / Feature** | **Description** | **Users** |
| **Vehicle Management** | Add, update, or remove vehicles; store vehicle details such as number, model, and type | Admin, Manager |
| **Driver Management** | Register drivers, assign them to vehicles, and monitor their status | Admin, Manager |
| **Real-Time Tracking** | Track the live location of vehicles for better route planning and safety | Admin, Manager |
| **Trip Management** | Plan trips, assign drivers, and record trip start and end details | Admin, Driver |
| **Maintenance Scheduler** | Schedule maintenance reminders and track completed services | Manager |
| **Alert System** | Receive alerts for over-speeding, route deviation, or emergency stops | Manager, Admin |
| **Analytics Dashboard** | Visualize fleet performance, fuel efficiency, and trip reports | Admin, Manager |
| **User Role Management** | Provide role-based access to Admins, Managers, and Drivers for security | Admin |
| **Report Generation** | Generate PDF/Excel reports for trips, maintenance, and performance | Admin, Manager |
| **Issue Reporting** | Drivers can submit issue reports during trips for quick action | Driver, Manager |

### ****Project Execution Plan:****

#### **Phase 1: Planning**

* Understand requirements from backend team.
* Finalize UI components: Login page, Dashboard, Vehicle Tracking, Reports.
* Prepare wireframes and color theme.

#### **Phase 2: Environment Setup**

* Install **Node.js, React.**
* Set up project structure using create-next-app.
* Configure GitHub for version control.

#### **Phase 3: UI Development**

* Build **Login page** with validation.
* Create **Dashboard** layout with charts.
* Add **Vehicle Management** module.
* Ensure **responsive design** for mobile and desktop.

#### **Phase 4: API Integration**

* Connect frontend with backend REST APIs.
* Test data flow (fetching and posting data).
* Handle error states and loading animations.

#### **Phase 5: Testing & Debugging**

* Test components using React Testing Library.
* Fix UI bugs and improve performance.
* Perform cross-browser testing.

#### **Phase 6: Deployment**

* Build production-ready frontend.
* Deploy on **local server**
* Verify live version with backend integration