**🔹 Project Title:**

**Vehicle Verified App**

**🔹 Tech Stack:**

* **Frontend: Flutter (Dart)**
* **Backend: Firebase (Authentication, Cloud Firestore, Firebase Storage)**
* **APIs: Google Gemini API (for AI Chatbot)**
* **Other Tools: Git, Android Studio / VS Code**

**✅ Phase-wise Roadmap**

**📌 Phase 1: Planning & Requirement Gathering**

* **Duration: 3 Days**
* **Tasks:**
* **Define App Objective: The core objective was defined: to reduce traffic congestion and save time during document checks by creating a QR-code-based verification system.**
* **Prepare SRS: A detailed Software Requirement Specification (SRS) document was created to outline all functional and non-functional requirements.**
* **Define Core Features: The main modules were identified: Owner Dashboard, Police Portal, Service Management, and AI Help.**
* **System Design: An Entity-Relationship (ER) Diagram and a Data Flow Diagram (DFD) were prepared to finalize the database structure and data flow.**
* **Tech Stack Finalized: Flutter for the frontend and Firebase for the backend were chosen.**

**📌 Phase 2: UI/UX Design & Project Setup**

* **Duration: 4 Days**
* **Tasks:**
* **UI Design: All screens were designed directly during the Flutter development process, focusing on a clean and user-friendly interface.**
* **UI Assets: A color scheme (using color.dart) and app icons/images (assets/image/) were prepared.**
* **Project Initialization: A new Flutter project was initialized and a Git repository was set up for version control.**
* **Firebase Configuration: A new Firebase project was created, and Authentication, Cloud Firestore, and Storage were configured. The firebase\_options.dart file was integrated into the project.**
* **Dependencies Added: All necessary packages (firebase\_core, cloud\_firestore, image\_picker, qr\_flutter, mobile\_scanner, http, etc.) were added to the pubspec.yaml file.**

**📌 Phase 3: Authentication & Core Structure**

* **Duration: 3 Days**
* **Tasks:**
* **File Structure: A clear project directory was established, separating screens by roles (owner, police) and utility files.**
* **Core App Flow:**
* **main.dart: The entry point of the app, initializing Firebase.**
* **splash\_screen.dart: The initial welcome screen.**
* **auth\_wrapper.dart: A gatekeeper to check the user's login status and role, redirecting them to the correct dashboard.**
* **auth\_selector\_screen.dart: A screen for users to choose their role (Owner or Police).**
* **Authentication Implementation:**
* **login\_screen.dart & register\_screen.dart: Implemented Sign Up and Login functionality using Firebase Authentication for both roles.**
* **forgot\_password.dart: Implemented the password reset feature.**
* **User data (name, email, role, etc.) is saved to the users collection in Firestore upon registration.**

**📌 Phase 4: Owner Module - Feature Development**

* **Duration: 10 Days**
* **Tasks:**
* **Owner Dashboard (owner\_dashboard\_screen.dart):**
* **Fetches and displays the owner's vehicles in real-time using StreamBuilder.**
* **Implemented a dynamic status system that checks document validity (Valid, Expired, Missing) in real-time.**
* **Integrated a notification system icon that becomes active when a service confirmation is pending.**
* **Vehicle Management (add\_vehicle\_screen.dart, vehicle\_details\_screen.dart):**
* **Implemented functionality to add new vehicles with all mandatory fields.**
* **Added "Edit" functionality to update existing vehicle details.**
* **Implemented a "Delete" feature for vehicles, which also removes all associated documents and history.**
* **Document Management (add\_edit\_document\_screen.dart, view\_doc\_screen.dart):**
* **Allowed users to upload document images using image\_picker and save them to Firebase Storage.**
* **Created a screen to view all documents for a specific vehicle, with options to view or delete each document.**
* **QR Code Generation (generate\_qr\_code\_screen.dart):**
* **Implemented QR code generation using the qr\_flutter package. The QR code securely contains only the vehicle's unique Firestore ID.**
* **Profile Management (owner\_profile\_screen.dart, edit\_profile\_screen.dart, etc.):**
* **Connected all profile screens to Firebase to fetch and display the user's real name, email, and profile picture.**
* **Implemented the "Edit Profile" feature to update user details and upload a profile picture.**
* **Implemented the "Change Password" feature for logged-in users.**
* **Service Management (general\_maintenance\_screen.dart, service\_history\_screen.dart):**
* **Created a dynamic service booking screen where users can select multiple services.**
* **The total cost is calculated in real-time based on the selected services and vehicle type (2-Wheeler/4-Wheeler).**
* **Service history is fetched from Firebase and displayed, with an option to filter by vehicle.**
* **Implemented the service confirmation flow, allowing users to mark a service as "Completed (Good/Bad)" after the service date has passed.**

**📌 Phase 5: Police Module - Feature Development**

* **Duration: 4 Days**
* **Tasks:**
* **Police Dashboard (police\_home\_screen.dart): Fetches and displays the logged-in officer's name from Firestore.**
* **QR Code Scanner (police\_scanner\_screen.dart):**
* **Integrated the mobile\_scanner package to open the camera.**
* **Added logic to verify the scanned vehicle ID with Firebase before proceeding.**
* **Manual Entry (manual\_entry\_screen.dart): Implemented a search feature to find vehicles by registration number in Firestore.**
* **Result Screen (scanned\_result\_screen.dart): Fetches and displays complete, real-time data (owner name, vehicle details, document status) from Firebase based on the scanned/entered vehicle ID.**
* **Police Profile (police\_profile\_screen.dart): Connected the screen to Firebase to display the officer's name and official ID.**

**📌 Phase 6: Integration, Testing & Security**

* **Duration: 5 Days**
* **Tasks:**
* **Firebase Security Rules: Wrote and refined complex security rules for both Firestore and Firebase Storage to ensure users can only access their own data, while allowing police officials to read necessary verification data. Fixed all permission-denied errors.**
* **Testing: Tested all features on an Android device, including user registration, document upload, QR scanning, and service booking.**
* **Bug Fixing: Resolved various bugs, including UI overflow issues, type-safety errors, and platform-specific problems (like camera permissions).**
* **AI Chatbot (help\_support\_screen.dart):**
* **Integrated the Google Gemini API using the http package.**
* **Implemented a secure way to handle the API Key using .env and flutter\_dotenv.**
* **Designed an interactive bottom sheet for the chat interface to solve keyboard visibility issues.**
* **Created a detailed prompt to provide context to the AI for accurate responses.**

**📌 Phase 7: Final Touches & Deployment**

* **Duration: 2 Days**
* **Tasks:**
* **UI Polishing: Refined the UI of several screens (about\_us\_screen.dart, terms\_privacy\_screen.dart) for a more professional look.**
* **Final Checks: Ensured all loading indicators, error messages, and user feedback mechanisms are in place.**
* **App Icon & Splash Screen: Finalized the app icon and splash screen.**
* **Release Build: Prepared the app for deployment by generating a release-ready APK.**

**📅 Timeline Summary**

|  |  |
| --- | --- |
| **Phase** | **Estimated Duration** |
| **Planning & Requirement** | **3 Days** |
| **UI/UX Design & Project Setup** | **4 Days** |
| **Authentication & Structure** | **3 Days** |
| **Owner Module Development** | **10 Days** |
| **Police Module Development** | **4 Days** |
| **Integration, Testing & Security** | **5 Days** |
| **Final Touches & Deployment** | **2 Days** |
| **Total Estimated Time** | **~31 Days** |

**🧾 Deliverables**

* **Working Flutter App (.apk)**
* **Complete Source Code**
* **Software Requirement Specification (SRS) Document**
* **This Project Roadmap Document**
* **Screenshots and Demo of the final app**