**Software Requirements Specification (SRS) for VehicleVerified**

**Version: 3.0**

**Date: July 14, 2025**

**Author: Suraj Kumar (Project Developer)**

**1. Introduction**

**1.1 Purpose**

**The primary purpose of this document is to provide a comprehensive and detailed description of all requirements for the "VehicleVerified" mobile application.**

**The main motive behind this project is to solve a significant real-world problem: traffic congestion and time wastage caused by manual, one-by-one vehicle document checks. For drivers who are short on time and for police officers managing traffic flow, the current process is inefficient. This application aims to digitize and streamline this entire process by converting all necessary vehicle documents into a single, scannable QR code, enabling verification in seconds.**

**This SRS will serve as the foundational agreement between all stakeholders, guiding the development, testing, and final evaluation of the product.**

**1.2 Scope**

**The scope of this project is to develop a complete mobile application that:**

1. **Allows Vehicle Owners to securely store and manage their vehicle documents (RC, Insurance, PUC, etc.) digitally.**
2. **Empowers Traffic Police Officials to instantly verify these documents by scanning a secure QR code or by manually entering the vehicle's registration number.**
3. **Provides a smart, user-friendly interface for service booking, history tracking, and receiving important notifications.**

**The application will be a standalone product, covering the entire lifecycle from user registration to real-time document verification.**

**1.3 Intended Audience**

**This document is intended for:**

* **Project Developers (Suraj Kumar): As a guide for building the application.**
* **Testers: To create test cases and verify that the software meets all requirements.**
* **Project Stakeholders: To understand the vision, features, and limitations of the product.**

**1.4 Definitions and Acronyms**

* **SRS: Software Requirements Specification**
* **UI/UX: User Interface / User Experience**
* **RC: Registration Certificate**
* **PUC: Pollution Under Control Certificate**
* **Firebase: A Backend-as-a-Service (BaaS) platform by Google used for the database, authentication, and file storage.**
* **API: Application Programming Interface**
* **Gemini API: An AI model API by Google used for the chatbot feature.**

**2. Overall Description**

**2.1 Product Perspective**

**VehicleVerified is a standalone, self-contained mobile application. It does not depend on any pre-existing system but utilizes the Google Firebase suite as its backend infrastructure.**

**2.2 Product Functions**

**The high-level functions of the app are:**

* **User Management: Secure registration, login, and profile management for both Vehicle Owners and Traffic Police.**
* **Vehicle & Document Management: Adding vehicles, uploading documents, and generating secure QR codes.**
* **Verification System: QR code scanning and manual search functionality for police.**
* **Service Management: Service booking, history viewing, and post-service confirmation.**
* **Notification System: Alerts for document expiry and service confirmations.**
* **AI-Powered Support: An integrated chatbot for user assistance.**

**2.3 User Classes and Characteristics**

* **Vehicle Owner: Any individual with a smartphone who owns one or more vehicles and wishes to digitize their documents for convenience and quick verification.**
* **Traffic Police: A trained and authorized official who needs a reliable and fast tool to verify vehicle documents on the road. They are expected to have a smartphone with a camera.**

**2.4 Operating Environment**

* **Platform: Android (6.0 Marshmallow and above) and iOS (12.0 and above).**
* **Framework: Flutter SDK (latest stable version).**
* **Backend: Google Firebase (Authentication, Cloud Firestore, Firebase Storage).**
* **External APIs: Google Gemini API for the AI chatbot.**

**2.5 Design and Implementation Constraints**

* **The application must be developed using the Dart programming language and the Flutter framework.**
* **The entire backend infrastructure will be hosted on Google Firebase.**
* **All user data must be encrypted both in transit (using HTTPS) and at rest (using Firebase's default encryption).**
* **The app must be designed with a mobile-first, user-friendly interface.**

**2.6 Assumptions and Dependencies**

* **An active internet connection is required for most features, including login, data synchronization, and verification.**
* **Users must have a valid email address for registration.**
* **The device must have a functional camera for QR code scanning and document uploading.**
* **The AI chatbot's performance is dependent on the availability and response time of the Google Gemini API.**

**3. Specific Requirements**

**3.1 Functional Requirements**

**FR-1: User Registration & Login**

* **FR-1.1: The system shall allow two types of users to register: owner and police.**
* **FR-1.2: Owners will register using their name, email, phone number, and password.**
* **FR-1.3: Police officials will register with the same details plus a mandatory "Official ID" field.**
* **FR-1.4: All users will log in using their registered email and password. The system will direct them to their respective dashboards based on their role.**
* **FR-1.5: A "Forgot Password" feature shall be available, which sends a reset link to the user's registered email.**

**FR-2: Profile Management**

* **FR-2.1: Users shall be able to view and edit their profile information, including their name and profile picture.**
* **FR-2.2: Users shall be able to change their account password after re-authenticating with their current password.**

**FR-3: Vehicle & Document Management (Owner)**

* **FR-3.1: Owners shall be able to add new vehicles by providing mandatory details like make, model, registration number, engine number, chassis number, and all relevant dates.**
* **FR-3.2: Upon adding a vehicle, the system will automatically create placeholder records for mandatory documents (e.g., Insurance, PUC) using the expiry dates provided.**
* **FR-3.3: Owners shall be able to upload image files for each document.**
* **FR-3.4: Owners can generate a unique, secure QR code for each vehicle. This QR code will only contain the vehicle's unique database ID.**

**FR-4: Verification (Police)**

* **FR-4.1: Police officials shall be able to scan a vehicle's QR code using the in-app camera.**
* **FR-4.2: Upon a successful scan, the app will fetch and display the vehicle's details, owner's name, and the real-time status (VALID, EXPIRED, MISSING) of all documents.**
* **FR-4.3: Police officials shall have the option to manually enter a vehicle's registration number to perform a search if QR scanning is not possible.**

**FR-5: Service & Notification Management**

* **FR-5.1: Owners can book services by selecting from a list of predefined tasks (e.g., oil change). The total cost will be calculated dynamically based on the selected services and vehicle type.**
* **FR-5.2: The system shall send a notification to the owner 30 days before any document is set to expire.**
* **FR-5.3: After a booked service's date has passed, the system will send a notification to the owner to confirm the service completion and provide a rating.**

**3.2 Non-Functional Requirements**

* **NFR-1 (Performance): The application must load within 3 seconds. UI transitions should be smooth. Data fetch operations from Firebase should not exceed 3 seconds on a stable 3G connection.**
* **NFR-2 (Security): All communication between the app and Firebase servers will be over HTTPS. Firebase Security Rules will be strictly configured to prevent unauthorized data access.**
* **NFR-3 (Reliability): The application should handle network errors gracefully and inform the user. It should have a crash-free session rate of over 99.5%.**
* **NFR-4 (Usability): The UI must be intuitive and easy to use for both tech-savvy and non-tech-savvy users. Key actions should be easily discoverable.**

**3.3 External Interfaces**

* **3.3.1 User Interface: The app will adhere to Material Design principles for a consistent and familiar user experience on Android and iOS.**
* **3.3.2 APIs: The app will interface with the Google Gemini REST API for the chatbot functionality.**
* **3.3.3 Database: The app will use Cloud Firestore as its primary database. The data structure is defined in the project's ER Diagram.**

**4. System Features (Use Case Breakdown)**

**Use Case 1: Verify Vehicle via QR Code**

* **Use Case Name: Instant Document Verification**
* **Actor: Traffic Police Official**
* **Preconditions: The police official is logged into the app and has an active internet connection.**
* **Main Flow:**

1. **The official taps the "Start Scanning" button on their dashboard.**
2. **The app opens the camera scanner.**
3. **The official points the camera at the vehicle's QR code.**
4. **The app instantly captures the vehicle's unique ID from the QR code.**
5. **The app sends this ID to Firebase to fetch the latest vehicle and document data.**
6. **The app displays the Scanned Result Screen with the owner's name, vehicle details, and a clear list of all documents and their status (VALID/EXPIRED).**

* **Alternate Flow: If the QR code is invalid or not found in the database, the app displays an error message.**
* **Postconditions: The verification process is complete. The official can choose to scan another vehicle.**

**5. Wireframes / UI Sketches**

***This section is intended for visual representations of the app's screens. Detailed UI mockups and prototypes, created using tools like Figma, will be maintained in a separate design document linked here for reference.***

**6. Appendix**

**6.1 Tools Used**

* **Framework: Flutter**
* **Language: Dart**
* **Backend: Firebase (Authentication, Firestore, Storage)**
* **IDE: Android Studio & VS Code**
* **AI Service: Google Gemini API**

**6.2 Future Scope**

* **Integration with government databases (like Parivahan) for automatic data fetching.**
* **Digital payment integration for challans and service costs.**
* **A dedicated web portal for administrative tasks, such as approving police accounts.**