**Blackrain: Software Requirements Specification (SRS)**

**Project Name:** Blackrain  
**Author:** Mr. Iraku  
**Version:** 1.0  
**Date:** June 2025  
**License:** GNU General Public License v3.0  
**Technology Stack:** React, Firebase, OpenWeatherMap API

**1. Introduction**

**1.1 Purpose**

The purpose of this document is to define the formal requirements for the **Blackrain** web application — a real-time, interactive, weather-forecasting application that provides users with current, hourly, and daily weather data, visualized through a clean, responsive, and customizable interface. It supports authentication, persistent preferences, and weather-related alerts.

**1.2 Scope**

Blackrain is a browser-based, Progressive Web App (PWA) built in React. It utilizes the OpenWeatherMap API for meteorological data and Firebase for user authentication and cloud storage of preferences and favorites. The system will provide features such as geolocation-based weather, AQI, UV Index, customizable settings, offline access, and accessibility support.

**1.3 Intended Audience**

* Software Developers (for implementation)
* Project Managers (for planning)
* QA Engineers (for test case generation)
* End Users (for usability expectations)

**1.4 Definitions**

|  |  |
| --- | --- |
| **Term** | **Definition** |
| PWA | Progressive Web App |
| AQI | Air Quality Index |
| UV Index | Ultraviolet Exposure Index |
| Firebase | Backend-as-a-Service used for auth and storage |
| OpenWeatherMap API | External weather data provider |

**2. Overall Description**

**2.1 Product Perspective**

Blackrain is a standalone system but integrates with:

* **OpenWeatherMap API**: for all weather-related data
* **Firebase Auth & Firestore**: for user login and storing bookmarks/settings

**2.2 User Classes and Characteristics**

|  |  |
| --- | --- |
| **User Role** | **Description** |
| Guest User | Can view current and forecasted weather for searched or geolocated locations |
| Authenticated User | Can save favorite locations, configure preferences, toggle dark mode, etc. |

**2.3 Operating Environment**

* Compatible with modern browsers: Chrome, Firefox, Edge, Safari
* Responsive design for desktops, tablets, and smartphones
* Deployed on Vercel/Firebase Hosting with HTTPS

**2.4 Design and Implementation Constraints**

* Must comply with OpenWeatherMap API rate limits
* Firebase Authentication and Firestore for backend services
* Licensed under GNU GPL v3

**3. Functional Requirements**

**3.1 Weather Data Features**

* **FR-01**: The system shall fetch and display current weather data for the selected location.
* **FR-02**: The system shall provide a 24-hour forecast for the selected location.
* **FR-03**: The system shall provide a 7-day forecast for the selected location.
* **FR-04**: The system shall fetch and display AQI data if available.
* **FR-05**: The system shall fetch and display the UV index for the selected location.
* **FR-06**: The system shall display sunrise and sunset times.
* **FR-07**: The system shall dynamically render weather icons based on condition codes.

**3.2 User Interaction Features**

* **FR-08**: The system shall allow users to search for locations by city name, coordinates, ZIP code, or country.
* **FR-09**: The system shall automatically detect the user's location (with permission).
* **FR-10**: The system shall allow users to toggle between Celsius and Fahrenheit.
* **FR-11**: The system shall provide a settings panel for unit and appearance configuration.
* **FR-12**: The system shall support dark and light mode themes.

**3.3 User Authentication & Data Storage**

* **FR-13**: The system shall allow users to sign up and log in using email/password or Google OAuth.
* **FR-14**: The system shall persist user preferences in Firebase Firestore.
* **FR-15**: The system shall allow users to bookmark favorite locations.
* **FR-16**: The system shall allow users to delete saved preferences.

**3.4 Offline & PWA Support**

* **FR-17**: The system shall cache the latest weather data for offline access.
* **FR-18**: The system shall function as a PWA, with installable capability.

**3.5 Notifications and Alerts**

* **FR-19**: The system shall display weather alerts when returned by the API.
* **FR-20**: The system shall display warning indicators for severe AQI or UV conditions.

**3.6 Accessibility & Responsiveness**

* **FR-21**: The system shall support keyboard navigation.
* **FR-22**: The system shall support high-contrast modes for visibility.
* **FR-23**: The system shall be fully responsive across all screen sizes.

**4. Non-Functional Requirements**

|  |  |
| --- | --- |
| **ID** | **Requirement** |
| NFR-01 | The app shall load on modern browsers within 3 seconds under standard network conditions. |
| NFR-02 | The app shall maintain a 99.9% uptime (cloud-hosted). |
| NFR-03 | The app shall comply with WCAG 2.1 accessibility standards. |
| NFR-04 | The app shall be built as a PWA with offline capabilities. |
| NFR-05 | All user preferences must be securely stored and retrieved from Firebase Firestore. |
| NFR-06 | The app shall be optimized for performance and avoid redundant API calls. |
| NFR-07 | The app shall securely handle API keys and Firebase credentials using environment variables. |

**5. External Interfaces**

**5.1 OpenWeatherMap API**

* **Data Retrieved**: Current, hourly, daily weather; AQI; UV index; alerts
* **Protocol**: HTTPS REST
* **Auth**: API key (stored in .env)

**5.2 Firebase (Auth & Firestore)**

* **Services**: Authentication (email/Google), NoSQL DB for bookmarks/settings
* **Security**: Firebase Rules + Auth tokens

**6. Use Cases**

**Use Case 1: View Current Weather**

**Actor**: Guest or Authenticated User  
**Precondition**: User has entered or allowed location  
**Flow**:

1. System retrieves weather from OpenWeatherMap.
2. System displays weather with icons, temperature, humidity, etc.

**Use Case 2: Sign In and Save Preferences**

**Actor**: Guest → Authenticated User  
**Flow**:

1. User signs up via Firebase Auth.
2. System creates a profile and Firestore doc.
3. User selects dark mode and bookmarks cities.
4. Preferences are stored in Firestore.

**Use Case 3: Offline Access**

**Actor**: Any User  
**Precondition**: User visited site with connection  
**Flow**:

1. User loses network connection
2. System displays cached weather data
3. Offline banner or indicator shown

**7. Assumptions & Dependencies**

* The OpenWeatherMap API and Firebase will remain available and within free-tier limits.
* Browser geolocation will be permitted by the user.
* Users have modern browsers that support ES6+ JavaScript and service workers.