Software Requirements Specification (SRS) Document for EcoHabit

Project Title: EcoHabit - Habit Tracker
Team Members: *Josue Pavon, Yared Bustillo, Jonathan Concha.exe, Liana Mikhailova*
GitHub Repository: *https://github.com/YaredBM/Habit-Tracker-App*

1. Introduction

1.1 Purpose

This document details the specifications for "EcoHabit - Habit Tracker," a mobile application designed to assist students in developing healthy routines through an engaging and user-friendly habit-tracking system.

1.2 Scope

The scope of this product is a mobile application developed for the Android platform. The app will allow users to:

- Sign up and log in to an account.
- Create, track, and manage personal habits.
- View habit progress and streaks.
- Browse popular habit templates.
- Manage account and app settings.

1.3 Definitions, Acronyms, and Abbreviations

- **Android:** A mobile operating system developed by Google.
- **APK (Android Package):** The package file format used by the Android operating system for the distribution and installation of mobile apps.
- **Auth (Authentication):** The process of verifying the identity of a user.
- **Backend:** The server-side components of an application, which handle logic, the database, and authentication.
- **Dart:** The programming language used to build Flutter applications.
- **DB (Database):** An organized collection of structured data.
- **Figma:** A collaborative web-based design tool used for creating user interfaces.
- **Firebase:** A platform developed by Google for creating mobile and web applications. It provides Backend services.
- **Firebase Auth:** A Firebase service that provides user authentication.
- **Firestore:** A cloud-hosted, NoSQL database provided by Firebase.
- **Flutter:** An open-source UI software development kit created by Google, used to develop cross-platform applications.
- **Frontend:** The client-side components of an application
- **FR (Functional Requirement):** Defines a specific function or behavior of the system.
- **NFR (Non-functional Requirement):** Defines quality attributes or constraints of the system.
- **SRS (Software Requirements Specification):** This document.
- **UI (User Interface):** The visual and interactive elements of an application.

1.4 References

- EcoHabit - Habit Tracker, Final Project Proposal (SE2 - EcoHabit.pdf).
- EcoHabit Figma Screen Designs (Screens 1-12).

1.5 Overview

This document is organized into three main sections.

- **Section 1 (Introduction):** Provides the purpose, scope, and overview of the application and this document.
- **Section 2 (Overall Description):** Describes the product perspective, user characteristics, operating environment, and high-level constraints.
- **Section 3 (Specific Requirements):** Details the explicit functional and non-functional requirements the software must meet.

2. Overall Description

2.1 Product Perspective

EcoHabit is a new, self-contained mobile application. It is not an extension of any existing system. It will operate using a Flutter-based frontend and will rely on Google's Firebase services (specifically Firestore and Auth) for backend database and authentication functionalities.

2.2 Product Functions

The primary functions of the EcoHabit app are:

- **User Authentication:** Allowing users to sign up and log in to the application.
- **Habit Management:** Enabling users to create, complete, and monitor habits.
- **Data Analytics:** Providing users with dashboards, streaks, and charts to visualize their habit consistency.
- **Push-Notifications:** Sending reminders to users to help them maintain habit streaks.

2.3 User Classes and Characteristics

The primary user class for EcoHabit is Students.

- **Characteristics:** These users often have busy and irregular schedules, juggling classes, part-time work, and a social life.
- **Needs:** They require a simple, fast, and easy-to-use mobile tool that helps them stay organized and make more sustainable choices without requiring significant effort. They are motivated by seeing tangible outcomes and progress.

2.4 Operating Environment

- The application will be a mobile app developed for the Android operating system.
- It will be built using the Flutter framework with the Dart language.
- The backend database and authentication services will be provided by Firebase (Firestore).
- The app is expected to have offline functionality.

2.5 Design and Implementation Constraints

- **Technology Stack:** The project uses Flutter (Dart) for the frontend and Firebase (Firestore) for the backend and database.
- **Platform:** The initial deliverable must be an installable Android app (APK).
- **Design:** The app must follow the dark-mode, high-contrast visual design specified in the Figma files.
- **Privacy:** Location tracking features must be optional and require explicit user opt-in.
- **Security:** The system must use Firebase Auth for security and ensure all user data is user-owned.

2.6 User Documentation

End-user documentation will be minimal, as the app is designed to be intuitive. For development, the project will include clear documentation on architecture, data models, and emission factors.

2.7 Assumptions and Dependencies

- **Assumptions:**
  - Users will have access to a smartphone running the Android operating system.
- **Dependencies:**
  - The application is dependent on Google's Firebase platform for backend services.

3. Specific Requirements

3.1 Functional Requirements

FR-1: User Authentication

- **FR-1.1 (Login Screen):** The system shall provide a Login screen (Screen 1) with:
  - A "Username" input field.
  - A "Password" input field with a show/hide toggle.
  - A "Sign in" button.
  - A "Register" button.
  - A "Forgot Password?" link.
- **FR-1.2 (Registration Screen):** The system shall provide a Register screen (Screen 2) with:
  - A "Create Username" input field.
  - A "Create Password" input field with a show/hide toggle.
  - A "Sign in" button.
- **FR-1.3 (Social Login):** The system shall allow users to continue with Google, Apple, or Facebook for both login and registration.
- **FR-1.4 (Logout):** The system shall provide a Log out button in the Settings screen (Screen 5).

FR-2: Habit Management

- **FR-2.1 (Create Habit Screen):** The Create Habit screen (Screen 4) shall allow a user to define a new habit by providing:
  - A "Name" for the habit.
  - A "Description" for the habit.
  - A "Color" for the habit.
  - A "Repeat" toggle.
  - If "Repeat" is on, the user should be able to select frequency ("Today", "Weekly", "Overall") and specific days (M, T, W, T, F, S, S).
  - A "Reminder" toggle.
  - An "Add my friend" toggle.
  - A "Create" button to save the new habit.
- **FR-2.2 (Complete Habit):** From the Habits screen (Screens 9, 10), users shall be able to mark a habit as complete for the day by tapping a checkmark circle for that day.
- **FR-2.3 (Popular Routines):** The system shall provide a Popular Routines screen (Screen 8) that displays predefined habit templates (e.g., Don't drink alcohol) that the user can select from.

FR-3: Visualization & Insights

- **FR-3.1 (Habits Dashboard):** The main Habits screen (Screens 3, 9, 10, 12) shall serve as the user's dashboard.
- **FR-3.2 (Habit Views):** The Habits screen shall have three tabs for viewing habits: Today, Weekly, and Overall.
- **FR-3.3 (Empty State):** If no habits are scheduled for the "Today" view, the system shall display an "Oops, no habits for today" message and a "Create" button (Screen 3).
- **FR-3.4 (Habit Cards):** The "Overall" view (Screen 10) shall display a list of all habits as cards. Each card shall show the habit name, description (e.g., "100 days of..."), and daily completion checkmarks.
- **FR-3.5 (Streak Visualization):** The "Today" view (Screens 9, 12) shall display a large circular graphic showing the user's current streak in days (e.g., "35 DAYS").
- **FR-3.6 (Motivational Message):** The dashboard shall display motivational messages, such as "Keep Moving Forward!" (Screen 12).
- **FR-3.7 (Analytics Screen):** The system will feature an analytics screen, accessible through the bottom navigation bar. This screen will display weekly and monthly charts detailing habit completion rates and earned badges.

FR-4: General System

- **FR-4.1 (Navigation):** The system shall have a main bottom navigation bar with icons for "Home" (Habits), "Popular Routines" (Heart icon), and "Analytics" (Stats icon).
- **FR-4.2 (Notifications Icon):** The system shall have a notification bell icon in the header.
- **FR-4.3 (Settings Screen):** The "Settings" screen (Screen 5) shall provide:
  - "Language" selection (EN, ES, RU).
  - A "Notifications" toggle.
  - A "Time" selector for notifications, active if "Notifications" is toggled on.
  - A "Motivational Quotes" toggle.
- **FR-4.4 (Account Screen):** The "Your account" screen (Screen 6) shall display:
  - Read-only fields for "Username", "Password", and "Email".
  - A navigation link to "My habits".
  - A navigation link to "Clean everything".

3.2 Non-functional Requirements (NFR)

NFR-1: Performance

- **1.1 App Launch:** The application should offer a rapid startup time. A warm start needs to be perceived as instantaneous by the user.
- **1.2 UI Responsiveness:** User Interface (UI) performance must be seamless; all transitions, animations, and screen loading times (such as opening or switching tabs) are required to be smooth and fluid, completely free of any stuttering or lag.
- **1.3 Data Interaction:** All user interactions require immediate visual confirmation. For instance, the UI should update instantly when a user taps the checkmark to complete a habit.
- **1.4 Data Operations:** The system will be highly responsive; specifically, saving a new habit (via the Create button) or updating user account settings will be quick when the user has a stable internet connection.

NFR-2: Usability

- **2.1 Visual Consistency:** The application's design should strictly conform to the aesthetic, including the dark-mode theme, typography, component styling, and color palette, as defined in the official Figma screen designs.
- **2.2 Intuitive Navigation:** The main bottom navigation bar consistently provides access to the primary sections of the application, as depicted in the design specifications.
- **2.3 Clarity and Readability:** All text, labels, and icons must be clearly legible against the dark background. Input fields (Name, Password, etc.) need to have clear placeholder text.
- **2.4 Efficiency of Use:** The core user flow of completing a habit has to be highly efficient. From the "Home" screen, a user should be able to mark a habit as complete in a single tap.
- **2.5 Learnability:** The app's functions need to be self-explanatory. A new user should be able to successfully sign up, create a new habit, and complete that habit without requiring external tutorials, based on the clear UI layout.

NFR-3: Reliability & Availability

- **3.1 Offline Functionality:** Users ought to be able to maintain basic functionality even without an active internet connection. This includes the ability to view their established habits and mark habits as completed for the current day.
- **3.2 Local Data Caching:** The app shall use local device storage to cache all user-generated data (habits, completion status, etc.) to enable offline functionality.
- **3.3 Data Synchronization:** Upon restoration of the internet connection, the application automatically and seamlessly synchronizes all locally stored data changes with the Firebase Firestore backend. This process is correct, requires no user interaction, and guarantees zero data loss.
- **3.4 Error Handling:** The application needs robust error handling. In cases of potential failures, such as a failed login or data synchronization, the app displays clear and helpful error messages to the user.

NFR-4: Security

- **4.1 Authentication:** User passwords must be securely managed by Firebase Auth and never stored in plain text in the database or on the device.
- **4.2 Data Access Control:** Security rules for Firebase Firestore ought to enforce a "user-owned" data model. Specifically, a user's access to read or write data should be strictly limited to the data associated with their unique User ID. Accessing or modifying data belonging to any other user is strictly prohibited.
- **4.3 Session Management:** The app securely manages user login sessions, persisting the login state after the app is closed. The "Log out" button should properly invalidate the user's session token and require re-authentication.
- **4.4 Data in Transit:** All communication between the app and the Firebase backend needs to be encrypted (e.g., using HTTPS).

NFR-5: Implementation

- **5.1 Frontend Technology:** The client-side application ought to be developed using the Flutter framework and Dart language.
- **5.2 Backend Technology:** All backend services, including the database and user authentication, must be implemented using Firebase.
- **5.3 Code Maintainability:** Code should be well-commented, organized, and follow standard Dart and Flutter conventions to ensure it is maintainable by the team.

NFR-6: Portability

- **6.1 Target Platform:** The application is required to be developed, compiled, and capable of deployment on the Android operating system.
- **6.2 Final Deliverable:** The final product deliverable shall be a signed, installable APK file.
- **6.3 Device Compatibility:** The application is required to be compatible with a standard array of Android devices and screen sizes, necessitating support for Android OS version 8.0 or later.