**Software Requirements Specification (SRS)**

**Project:** SkillSwap – A Skill Exchange Platform  
**Prepared by:** [Fiza Bashir]  
**Date:** [26-09-2025]

**1. Introduction**

**1.1 Purpose**

The purpose of this document is to define the requirements for the SkillSwap application. SkillSwap allows users to sign in, browse computing-related skills, view details, and request skill swaps. The app also manages user sessions using persistent storage.

**1.2 Scope**

SkillSwap is a cross-platform mobile application built with React Native and Expo. It enables students/professionals to exchange skills in computing domains (e.g., programming, databases, networking). The app provides:

* User authentication (sign in, demo login, sign out).
* Persistent session management (via AsyncStorage).
* Display of skills with details (title, description, images).
* Option to request a skill swap.

**1.3 Definitions, Acronyms, and Abbreviations**

* **AsyncStorage** – External system for storing session tokens locally.
* **SkillSwap App** – Main application system.
* **Skill** – A computing-related ability shared between users.

**1.4 References**

* Project Codebase (React Native with Expo).
* React Native Documentation.
* AsyncStorage API Documentation.

**2. Overall Description**

**2.1 Product Perspective**

SkillSwap is a standalone mobile application. It interacts with **AsyncStorage** for session persistence but does not depend on external servers (in current version).

**2.2 Product Functions**

* User can log in with credentials.
* User can log in with a demo account.
* User can log out (session cleared).
* User can view a list of computing skills.
* User can view detailed information about a skill.
* User can request a skill swap.
* System saves/retrieves/removes login token using AsyncStorage.

**2.3 User Characteristics**

* Users are students, graduates, or professionals.
* Basic familiarity with mobile applications is assumed.

**2.4 Constraints**

* Internet connection required for full functionality (future versions).
* Application built with React Native + Expo (version compatibility constraints).
* Persistent data limited to local device storage.

**2.5 Assumptions and Dependencies**

* AsyncStorage is available on the device for session persistence.
* User provides valid login credentials.

**3. System Features**

**3.1 Sign In**

**Description:** User enters email and password to log in.  
**Actors:** User, SkillSwap App.  
**Preconditions:** User must have an account.  
**Postconditions:** Token is generated and stored in AsyncStorage.

**3.2 Demo Login**

**Description:** User logs in with predefined demo credentials.  
**Actors:** User, SkillSwap App.  
**Preconditions:** Demo credentials are predefined.  
**Postconditions:** Demo session is started.

**3.3 Sign Out**

**Description:** User logs out, and token is removed.  
**Actors:** User, AsyncStorage.  
**Preconditions:** User must be logged in.  
**Postconditions:** Token is deleted from AsyncStorage.

**3.4 View Skills List**

**Description:** User views list of available computing skills.  
**Actors:** User.  
**Preconditions:** App is running.  
**Postconditions:** Skills are displayed.

**3.5 View Skill Details**

**Description:** User selects a skill to view detailed info.  
**Actors:** User.  
**Preconditions:** Skills list must be displayed.  
**Postconditions:** Modal with skill info is shown.

**3.6 Request Skill Swap**

**Description:** User requests a swap for a skill.  
**Actors:** User.  
**Preconditions:** Skill details must be open.  
**Postconditions:** Alert confirms request.

**4. External Interface Requirements**

**4.1 User Interfaces**

* **Login Screen**: Email, password, demo login option.
* **Skills Screen**: List of skills with titles and images.
* **Skill Details Modal**: Description, image, and request button.

**4.2 Hardware Interfaces**

* Runs on mobile devices (Android, iOS).

**4.3 Software Interfaces**

* **AsyncStorage API** for token management.

**4.4 Communications Interfaces**

* Local storage interaction only (no remote API in current version).

**5. Non-Functional Requirements**

**5.1 Performance Requirements**

* App should load skills within 2 seconds.
* Session retrieval must be seamless on app restart.

**5.2 Security Requirements**

* User credentials must be securely handled.
* Token storage must follow secure practices.

**5.3 Usability Requirements**

* Intuitive interface with minimal steps for login and skill browsing.

**5.4 Reliability & Availability**

* The app must function offline for skills browsing (cached data).
* AsyncStorage must be accessible for session persistence.

**6. UML Diagrams**

* **Use Case Diagram** → Shows User + AsyncStorage interactions.
* **Class Diagram** → Defines User, Skill, SkillSwapApp, and AsyncStorage classes.

**7. Appendices**

* **Future Enhancements:**
  + Backend integration for real user accounts.
  + Real-time chat for skill exchange.
  + Rating and feedback system.