Project Proposal

Recipe Book App

— A Smart Kitchen Assistant —

By: Commit & Chill Team

Instructor Name: Yumna Al-Fara

Team Members:

- **Shahed Eyad Almobayed** – Product Owner
- **Shahed Haitham Alkhateeb** – Scrum Master
- **Rahaf Saifeldeen Elzebda** – Developer
- **Sara Basel Dwima** – Developer

1. Project Overview

RecipeBook is an interactive social platform for cooking enthusiasts, allowing users to discover, add, and manage their recipes in the cloud. The app aims to organize the cooking experience by dynamically categorizing recipes and providing quick access to ingredients and steps with multimedia support (photos and videos).

2. Problem Statement

Many cooking enthusiasts suffer from scattered recipes between traditional books and social media sites, making it difficult to access or modify a specific recipe. The app solves this problem by providing a personal cloud storage that allows smart searching and filtering by type, with the ability to modify and delete at any time.

3. Target Users

Primary User Type 1: Home Chefs / Foodies

- **Description:** Individuals who are passionate about cooking and are looking for a single place to save their experiences.
- **Goals:** Save their own recipes, browse other people's recipes, and quickly access ingredients.
- **Pain Points:** Losing paper recipes, difficulty organizing categories, and inability to save photos of dishes.

Primary User Type 2: Content Creators

- **Description:** Chefs who want to share their work with others.
- **Goals:** Present their recipes professionally with high-quality video and photo links.
- **Pain Points:** The need for a simple content management platform (CRUD) without technical complications.

Secondary User Type 3: Casual Browsers (Guests)

- **Description:** Users who are looking for quick inspiration for today's meal without wanting to create content.
- **Goals:**Browse the latest recipes, explore different categories, and watch preparation methods.

4. Core Features (Must-Have for MVP)

◆ **Cloud authentication and user profile**

Description: A complete system for registration, login, and password recovery with a personal profile (Firebase Auth).

User benefit: Protection of personal data and access to it from any device.

- **Dynamic Recipe Management (CRUD)**

  Description: Ability to add, edit, view, and delete recipes (Firestore).

  User benefit: Full control over personal content and continuous updates.

- **Cloud media hosting**

  Description: Upload recipe and profile photos to Cloudinary.

  User benefit: Save phone space and ensure image display quality and speed.

- **Smart filtering and category tabs**

  Description: Smart Filtering: An advanced filtering system based on category, time, and calories.

  User benefit: Access the desired recipe in seconds.

- **Detailed recipe display**

  Description: Formatted display of ingredients (points) and steps with video link support.

  User benefit: Easy to follow instructions during the cooking process.

5. Additional Features (Nice-to-Have)

1. **Offline Mode:** Browse preloaded recipes without an internet connection.
2. **Favorites List:** Save recipes in a special section for quick access later.
3. **Social Interaction:** Like or comment on other users' recipes.
4. **Shopping List:** Automatically convert ingredients into a shopping list.
5. **Share Recipe as Text/Image:** A button that allows the user to send the recipe text (ingredients and steps) or link via WhatsApp or other messaging apps.
6. **Recipe PDF Export**: Ability to convert the recipe page to a PDF file and save it on your phone.
7. **Search by ingredients**: A feature that allows users to enter the ingredients they currently have available, and the app will immediately display recipes that contain only those ingredients.
8. **Smart Filtering:** An advanced filtering system based on category, time, and calories.

6. Technical Stack

Frontend:

- **Framework:** Android SDK (Java)
- **Styling:** Material Design, XML Layouts
- **Additional Libraries:** Picasso (Image Loading), ViewPager2, View Binding, Shimmer Effect.

Backend:

- **Framework:** Firebase Services
- **API:** RESTful (via Cloudinary API for media)
- **Authentication:** Firebase Authentication

Database:

- **DBMS:** Firebase Firestore (NoSQL)
- **Storage:** Cloudinary (Cloud Media Storage)

Development Tools:

- **Version Control:** Git/GitHub
- **IDE:** Android Studio

Testing:

- JUnit & Mockito
- Espresso
- Firebase Test Lab
-

CI/CD

- GitHub Actions

7. System Architecture

User → Mobile App (Frontend) → Firebase Auth → Firestore (Metadata) → Cloudinary (Images)

8. Project Timeline

Sprint 1 (Week 3-4): Connectivity & Offline Access

- **Offline Mode:** Enable local data persistence using Firestore Persistence to allow users to browse recipes without an internet connection.
- **Favorites List:** Implement a system to save favorite recipes in a dedicated section for quick and easy access.

Sprint 2 (Week 5-6): Advanced Discovery Tools

- **Search by Ingredients:** Develop a smart search engine that filters and displays recipes based on the ingredients available to the user.

- **Advanced Smart Filtering:** Enhance the filtering system to include specific criteria such as preparation time, calorie count, and meal type.

Sprint 3 (Week 7-8): Social Engagement & Productivity

- **Social Interaction:** Add social engagement features, allowing users to "Like" and "Comment" on recipes shared by others.
- **Shopping List:** Implement a feature that automatically converts recipe ingredients into an interactive digital shopping checklist.

Sprint 4 (Week 9-10): Export & Sharing Features

- **Share Recipe as Text/Image:** Enable users to share recipe details or links via social media and messaging apps (e.g., WhatsApp).
- **Recipe PDF Export:** Develop a functionality to convert the recipe details page into a PDF file for local saving or printing.

Sprint 5 (Week 11-12): Polish, Testing, & Bug Fixes

- **Comprehensive Testing:** Conduct thorough testing of all new features, ensuring the accuracy of the "Search by Ingredients" logic and PDF export functionality.
- **UI Polishing:** Refine the user interface and resolve any technical bugs or glitches discovered during the testing phase.

Sprint 6 (Week 13-14): Final Testing & Presentation Prep

- **Integrated System Testing:** Perform a final end-to-end test of the entire application to ensure system stability.
- **Presentation & Demo:** Prepare the final presentation and a demo video showcasing the user journey, from searching by ingredients to exporting a recipe

9. Success Criteria

The project will be considered successful if:

- **[ ] All core features are implemented and working:**
    - Full Cloud Authentication & Profile management (Firebase).
    - Advanced Recipe Management (CRUD) with Cloud Media hosting (Cloudinary).
    - Intelligence tools: Search by Ingredients and Smart Filtering (Time/Calories).
    - Productivity tools: Offline Mode and Auto-generated Shopping List.
    - Engagement tools: Social interactions (Likes/Comments) and PDF Export.
- **[ ] Users can complete primary workflows without errors:**
    - Users can register, create a recipe with images, and find it via smart filters.
    - Users can successfully convert ingredients to a checklist and use it in offline mode.

    - Users can export and share their recipes as formatted PDFs or via social apps.
- **[ ] Code is well-documented and maintainable:**
    - Using clean architecture (MVC/MVVM) with clear comments for Cloudinary and Firebase logic.
    - Following naming conventions for layouts, classes, and variables.
- **[ ] Test coverage is at least 70%:**
    - Unit tests for search algorithms and calorie calculations.
    - UI testing (Espresso) for critical flows like registration and recipe creation.
- **[ ] Application is responsive and user-friendly:**
    - The UI is optimized for different screen sizes and orientations.
    - Using Shimmer effects and Progress bars to ensure a smooth user experience during cloud uploads.
- **[ ] Security best practices are followed:**
    - Firestore Security Rules are implemented to ensure only owners can edit or delete their recipes.
    - Secure API key management for Cloudinary and Firebase services.

10. Risks and Mitigation

Risk 1: Cloud Storage Quota and Latency

- **Description:** Heavy reliance on Cloudinary for high-quality images might hit free tier limits or cause slow loading times on weak connections.
- **Mitigation:** Implement image compression before uploading and use the **Picasso** library for efficient image caching and resizing to reduce bandwidth consumption.

Risk 2: Complexity of "Search by Ingredients" Logic

- **Description:** As the database grows, searching through string-based ingredients can become slow or return inaccurate results.
- **Mitigation:** Structure ingredients as an **ArrayList** in Firestore to use optimized queries and implement a normalized search algorithm to handle typos or similar terms.

Risk 3: PDF Generation and Device Compatibility

- **Description:** Generating PDFs on different Android versions (API levels) can lead to crashes or inconsistent formatting.
- **Mitigation:** Use a robust, well-documented library like **PdfDocument** or **iText**, and conduct extensive testing across various Android versions and screen sizes to ensure layout stability.

Risk 4: Offline Data Synchronization

- **Description:** Conflicts may occur when a user modifies "Favorites" while offline and then reconnects to the internet.

- **Mitigation:** Enable **Firestore Persistence** and implement a "last-write-wins" strategy or timestamp-based synchronization to ensure data consistency between local and cloud storage.

Risk 5: Technical Complexity of Social Interactions

- **Description:** Managing real-time likes and comments can lead to high "Read/Write" costs in Firebase or data desynchronization.
- **Mitigation:** Use **Firebase Transactions** for incrementing likes to avoid race conditions and optimize Firestore rules to prevent unauthorized data manipulation.

11. Team Roles and Responsibilities

Shahed Eyad Almobayed - Product Owner:

- Manages the product backlog and refines user stories.
- Defines feature priorities (e.g., prioritizing Search by Ingredients over PDF Export).
- Accepts completed work and ensures it aligns with the project vision.

Shahed Haitham Alkhateeb - Scrum Master:

- Facilitates Scrum ceremonies (Daily Standups, Sprint Planning).
- Removes blockers and ensures the team has access to Firebase/Cloudinary resources.
- Tracks overall progress and manages the project timeline.

Rahaf Saifeldeen Elzebda - Lead Developer (Backend Focus):

- Architecting the Firebase Firestore structure and security rules.
- Implementing Cloudinary integration for cloud media hosting.
- Developing the logic for Offline Mode and data persistence.

Sara Basel Dwima - Frontend Developer (UI/UX Focus):

- Implementing the UI/UX design using XML and Material Design components.
- Developing responsive layouts for the Shopping List and Recipe Details.
- Ensuring frontend integration with Firebase and handling user interactions.

Note: All members contribute to all areas to ensure knowledge sharing, but these roles represent the primary focus of each member during the development lifecycle.

11. Deliverables

- Working web application

- Source code on GitHub
- Documentation (README, API docs, user guide)
- Test suite
- Presentation slides
- Demo video

Approved by Team:

- **Shahed Eyad Almobayed** - [Date: Feb 13, 2026]
- **Shahed Haitham Alkhateeb** - [Date: Feb 13, 2026]
- **Rahaf Saifeldeen Elzebda** - [Date: Feb 13, 2026]
- **Sara Basel Dwima** - [Date: Feb 13, 2026]

Instructor Approval: _____ Date: _____