Table of Content

Ethiopian Traditional Food Recipe Mobile App

Chapter 1: Introduction

1.1. Introduction

This document presents the complete project plan for the Ethiopian Traditional Food Recipe mobile application, outlining both its technical architecture and development roadmap. Designed as a cross-platform solution using Flutter framework, the application seeks to transform how users engage with Ethiopian cuisine by creating an all-in-one digital platform for discovering authentic recipes, planning meals, and mastering traditional cooking techniques. More than just a recipe repository, this project aims to build an intuitive culinary companion that preserves cultural heritage while incorporating modern features like smart meal planning and personalized recommendations. The following documentation comprehensively covers all aspects of the initiative - from system design and technical specifications to development methodologies, resource allocation, testing protocols, and deployment strategies - providing stakeholders with a complete blueprint for bringing this vision to life.

1.2 Background of the Project

1.2.1 Context and Historical Significance

Ethiopian cuisine stands as one of Africa's most ancient and culturally significant food traditions, featuring distinctive elements that set it apart globally. At its heart lies the UNESCO-recognized gursha tradition of communal dining, along with unique ingredients like teff, an ancient gluten-free grain that forms the basis of injera, the national flatbread. These culinary practices represent not just nourishment but a centuries-old way of life that binds communities together. However, this rich heritage faces modern challenges. Recent data from the Ethiopian Cultural Survey (2023) reveals that 68% of diaspora youth lack the skills to prepare traditional dishes, while the Culinary Accuracy Index indicates 42% of online Ethiopian recipes contain significant inaccuracies. This growing knowledge gap threatens the preservation of authentic cooking methods and cultural traditions, particularly among younger generations and global communities seeking to connect with Ethiopian culture through its cuisine.

1.2.2 Current Landscape and System Limitations

The current ecosystem for Ethiopian recipe discovery presents several limitations:
Existing Solutions Analysis:

- Recipe Blogs offer detailed narratives but suffer from inconsistent standards
- Video Platforms provide visual learning but lack organizational structure
- General Recipe Apps contain large databases but miss cultural context
- Traditional Cookbooks maintain authenticity but offer no interactive features

These fragmented resources create significant barriers for users:

1. Accessibility Challenges: Traditional knowledge remains locked in physical cookbooks or family memories
2. Quality Concerns: Online recipes frequently compromise authenticity for convenience
3. Learning Curve: Beginners struggle with unstandardized instructions and measurements
4. Cultural Disconnect: Most platforms fail to convey the cultural significance behind dishes

Our solution directly addresses these gaps through:

- Rigorous authenticity verification by culinary experts
- Integrated meal planning and shopping list functionality
- Community-driven recipe validation system
- Cultural storytelling woven throughout the user experience

This project emerges at a critical juncture where technology can serve as both preserver and innovator - safeguarding traditional knowledge while making it accessible to global audiences through modern digital convenience. By creating this dedicated platform, we aim to establish a

new standard for authenticity in ethnic cuisine applications while fostering cross-cultural culinary appreciation.

1.3 Statement of the Problem

The digital landscape for Ethiopian culinary resources currently suffers from three fundamental deficiencies that this project seeks to address:

1. Cultural Preservation Crisis

The Ethiopian diaspora and younger generations face growing detachment from traditional cooking practices. Without accessible, authentic resources, valuable culinary knowledge risks being lost as it remains primarily transmitted through oral tradition and personal experience. This cultural erosion manifests in:

- Generational knowledge gaps in traditional techniques
- Adaptation of recipes that compromise authenticity
- Loss of cultural context surrounding dishes

2. Inefficient User Experience

Current solutions impose significant time burdens on users through:

- Fragmented resources requiring cross-platform searching
- Inconsistent formatting and measurement standards
- Lack of integrated meal planning tools
- Absence of visual learning aids for complex techniques

3. Quality Assurance Challenges

The unregulated nature of online recipe sharing leads to:

- Variance in ingredient authenticity (e.g., substitutions for traditional spices)
- Inaccurate cooking methods and times
- Missing nutritional information
- Unverified cultural claims

Research Framework

To systematically address these challenges, we focus on two key research questions:

1. Interface Design:
   "How can user experience principles optimize discovery and engagement with traditional recipes while maintaining cultural authenticity?"
2. Content Validation:
   "What verification mechanisms most effectively ensure recipe accuracy while accommodating regional variations?"

Validation Hypotheses

The project will test two core assumptions through measurable outcomes:

H_1: Standardization Effect

"Implementing standardized recipe templates with verified measurements and step-by-step visuals will increase first-time cooking success rates by ≥25% compared to existing online resources."

H_2: Integration Impact
"Seamless meal planning integration with automated shopping list generation will drive a 3× increase in weekly active usage compared to basic recipe repository apps."
This problem statement establishes both the urgent need for and measurable objectives of developing a dedicated Ethiopian culinary platform that bridges cultural preservation with modern convenience. The solution aims to transform how global audiences engage with one of Africa's most significant food traditions while setting new standards for ethnic cuisine applications.

1.4 Objective of the Project

1.4.1 General Objective

The primary objective of this project is to design, develop, and deploy a user-friendly, cross-platform mobile application that serves as a comprehensive digital resource for authentic Ethiopian traditional food recipes. The application aims to preserve culinary heritage while providing modern features that enhance meal preparation, cultural learning, and kitchen efficiency for users of all skill levels.

1.4.2 Specific Objectives

The project seeks to achieve the following specific objectives across five key areas:
A. Recipe Management System

- Develop a categorized and searchable database of over 300 verified Ethiopian traditional recipes.
- Implement advanced filtering options based on dietary preferences, cooking time, and difficulty level.
- Provide step-by-step recipe instructions with visual aids to support clear understanding.

B. Meal Planning Tools

- Design a smart weekly meal planner that promotes balanced nutrition.
- Generate automated shopping lists based on selected meal plans.
- Enable optional cooking reminders and meal preparation tracking features.

C. User Experience and Interface

- Develop an intuitive, responsive mobile UI using Flutter, supporting both English and Amharic languages.

- Implement a personalized recipe recommendation system based on user preferences and cooking history.

- Create community features such as user reviews, recipe ratings, and secure recipe submission.

D. Cultural Preservation

- Document traditional Ethiopian cooking techniques through video demonstrations and written guides.

- Include historical and cultural context for each dish to promote heritage awareness.

- Establish a recipe verification process with culinary experts to ensure authenticity and accuracy.

E. Technical Implementation

- Build the application using the Flutter framework to ensure cross-platform compatibility.

- Use Firebase for real-time data management, user authentication, and cloud storage.

- Provide offline access to favorited or saved recipes to ensure usability without internet connectivity.

- Conduct extensive testing to ensure performance, functionality, and usability.

- Deploy the application on major platforms, starting with the Android Play Store.

- Maintain clear documentation of the design process, system architecture, and testing results.

1.5 Scope and Limitations

1.5.1 Scope of the Project

The Ethiopian Food Recipe App is designed to serve as a culturally rich and user-friendly mobile application that promotes traditional Ethiopian cuisine while leveraging modern technology for meal planning and user engagement. The scope of this project includes the following dimensions:

A. Target Users

- Ethiopian individuals living locally and in the diaspora.

- International food enthusiasts interested in exploring Ethiopian culture.

- Culinary students and professionals seeking authentic recipe references.

B. Functional Scope

- Recipe Management: Browse recipes by category (e.g., stews, breads, vegetables, drinks) and search by name, ingredients, or dietary preferences (e.g., vegan, gluten-free).

- Recipe Detail View: Access comprehensive recipe pages with ingredients, preparation steps, serving size, cooking time, and high-quality images.

- User Features:

    - Optional account creation for saving favorite recipes.

    - (Planned) User-contributed recipes with moderation.

    - (Planned) Ratings, comments, and community interaction.

- Meal Planning:

    - Weekly meal calendar with customizable plans.

    - (Planned) Automated shopping list generation.

- Cultural Content:

    - Contextual and historical background of dishes.

    - (Planned) Traditional cooking methods via video demonstrations.

C. Technical Scope

- Platforms: Primary focus on Android; iOS support planned.

- Framework: Flutter for cross-platform development.

- Backend Services: Firebase Firestore for data storage, real-time sync, and user authentication (Google/Email sign-in).

- Architecture: Client-server model with potential cloud-based backend for scalability.
- Offline Access: Cached access to recently viewed or saved recipes.

D. Non-Functional Requirements

- Usability: Intuitive, accessible interface supporting Amharic and English.
- Performance: Fast loading times (<2s for core features).
- Reliability: Stable app behavior with error-handling.
- Security: Safe storage of user data and secure authentication.
- Accessibility: Designed to be inclusive of diverse users and devices.

E. Evaluation Criteria

- Verified accuracy for at least 90% of recipes.
- <2-second load time for primary features.
- At least 85% positive feedback in user testing phase.

1.5.2 Limitations of the Project

Despite the app's broad scope, the initial version will face several limitations due to time, resources, and technological constraints:

A. Content and Feature Constraints

- Initial Database: Limited to five major Ethiopian dish categories with gradual expansion.
- Nutritional Information: May have ±15% variation due to ingredient and preparation differences.
- Regional Variations: Not all regional versions of dishes will be included initially.
- Advanced Features:
    - Ingredient substitutions and allergy warnings are not available in version 1.

    - Step-by-step video tutorials and health app integrations are planned for future versions.

B. Technical Constraints

- Offline Access: Limited to approximately 50 most recent or favorited recipes.
- Connectivity Requirements: Video content requires a stable internet connection (minimum 4 Mbps).
- Platform Support: No desktop (Windows/macOS) version in the initial release.
- Flutter Limitations: Accessing advanced native features may require platform-specific development.

C. Operational and Moderation Challenges

- User Submissions: Community recipes will undergo a 24–48-hour moderation process to ensure quality and cultural relevance.
- Language Support: Initial release will primarily support English; Amharic support will follow.
- Dietary Categories: Support limited to 10 primary dietary filters in the first version.

1.6 Deliverables

The successful completion of the Ethiopian Food Recipe App project will result in the delivery of a range of technical artifacts, documentation, and support materials. These deliverables are organized into four categories: Technical Components, Documentation, Training Materials, and Quality Assurance Artifacts.

1. Technical Components

- Mobile Application Packages
    - Fully functional Flutter-based mobile application.
    - Deliverables include:
        - Android APK file.
        - (If within scope) iOS IPA file.

- Completion Criteria: App meets all functional and non-functional requirements and runs smoothly on target devices.

- Backend Configuration
  - Firebase setup for:
    - Firestore database integration.
    - Authentication and cloud storage.
  - Completion Criteria: Backend supports real-time synchronization and secure data handling.

- API Documentation
  - Technical documentation of APIs used or exposed for future integration (e.g., health tracking apps, external recipe platforms).
  - Completion Criteria: Clear, versioned API references with usage examples and authentication flow.

2. Documentation

- System Architecture Documentation
  - Diagrams and descriptions of the app's architecture, data flow, backend structure, and component interactions.
  - Completion Criteria: Provides a high-level overview and supports maintainability.

- User Manual (Digital/PDF)
  - A user-friendly guide for navigating the app, including:
    - Installation instructions.
    - Step-by-step feature usage.
    - Troubleshooting tips.

    - Completion Criteria: Designed for both novice and experienced users.

- Administrator Guide
    - A guide for content managers or admins to:
        - Add/edit/delete recipes.
        - Moderate user-submitted content.
    - Completion Criteria: Enables independent recipe management without developer intervention.

3. Training Materials

- Video Tutorials
    - Short instructional videos highlighting:
        - How to use key features.
        - How to submit a recipe.
        - Navigation and personalization tips.
    - Completion Criteria: At least 3 well-edited video tutorials covering major user interactions.

- Interactive Onboarding
    - In-app onboarding experience for new users.
    - Completion Criteria: Demonstrates core features in a guided, interactive format.

- Recipe Submission Guidelines
    - Step-by-step written and visual documentation for users contributing recipes.
    - Completion Criteria: Clear guidelines ensuring submission quality and cultural authenticity.

4. Quality Assurance Artifacts

- Test Case Documentation
    - Detailed unit, integration, and system test cases.
    - Completion Criteria: Covers critical paths and edge cases.
- Beta Testing Report
    - Summary of feedback from test users, including:
        - Bugs encountered.
        - Usability observations.
        - Suggestions for improvement.
    - Completion Criteria: Used to validate app readiness before public release.
- Performance Benchmark Metrics
    - Evaluation of key performance indicators such as:
        - Load times.
        - Crash rates.
        - Battery usage.
    - Completion Criteria: Meets or exceeds set performance standards.

1.7 Feasibility Study

The feasibility study evaluates the viability of developing and deploying the Ethiopian Food Recipe mobile application based on technical, operational, and economic considerations. Each aspect has been analyzed to identify enabling factors, potential risks, and mitigation strategies.

1.7.1 Technical Feasibility

The project is technically feasible, supported by proven development tools, scalable technologies, and available expertise.
Enabling Factors:

- Cross-Platform Development: Flutter enables simultaneous development for Android and iOS, significantly reducing time and effort compared to building separate native apps.

- Scalable Backend: Google Firebase offers a reliable, scalable backend solution for authentication, cloud storage, and real-time database operations with minimal infrastructure overhead.

- Content Availability: Existing culinary datasets and recipes can be utilized to populate the initial version of the app with curated content.

Identified Technical Risks:

- Cloud Storage Requirements: Video tutorials and rich media content may demand substantial storage and bandwidth, requiring cloud storage upgrades.

- Network Dependency: Real-time synchronization features could be challenged by users in low-bandwidth or rural areas.

- Hardware Compatibility: Device-specific issues (e.g., camera or media processing inconsistencies) may arise, particularly on older or less common Android devices.

Mitigation Strategies:

- Implement media compression and adaptive streaming for videos.

- Enable caching mechanisms for offline access to recently viewed content.

- Perform extensive testing across a range of device specifications.

1.7.2 Operational Feasibility

The project demonstrates high operational viability due to clear user interest, a sustainable maintenance approach, and the potential for community engagement.

User Adoption Potential:

- Target Market Readiness: Surveys and informal feedback indicate high interest in Ethiopian food from diaspora communities, culinary enthusiasts, and international users.

- Community Engagement: Social media groups and forums provide a ready-made platform for user acquisition, promotion, and feedback.

- Stakeholder Support: Preliminary discussions suggest that partner restaurants and chefs are open to contributing authentic content.

Maintenance and Sustainability:

- Modular App Design: A modular architecture allows for future updates, feature expansions, and localized enhancements with minimal disruption.

- Backup and Recovery: Automated backup systems will ensure data integrity and prevent loss of content.

- Crowdsourced Moderation: A user-driven moderation framework will support scalable, culturally sensitive recipe review processes.

1.7.3 Economic Feasibility

From a financial standpoint, the project is viable with manageable costs and promising long-term returns.

Cost Overview:

- Estimated Development Budget: $25,000 for a 6-month development cycle including design, coding, and testing.

- Recurring Cloud Costs: Approximately $300/month for Firebase services (database, storage, hosting, and analytics).

- Content Development: $8,000 allocated to recruit culinary experts, acquire media assets, and curate initial recipe content.

Return on Investment (ROI):

- Break-Even Target: The application is projected to break even with approximately 50,000 active users utilizing premium features or in-app purchases.

- Monetization Opportunities:

    - Premium subscription plans.

    - Sponsored recipe listings.

    - Strategic partnerships with Ethiopian restaurants or culinary brands.

    - Potential for cultural or tech grant funding.

1.8. Significance of the Project

The Ethiopian Food Recipe App carries profound importance in promoting cultural heritage, enhancing culinary education, and leveraging technology to bridge traditional practices with modern lifestyles. Its significance can be viewed from multiple impactful perspectives:

1. Cultural Preservation and Heritage Promotion

Ethiopian cuisine, renowned for its diversity and deep cultural roots, is at risk of being diluted or lost, especially among younger generations and the diaspora. This app serves as a digital repository that documents, preserves, and promotes traditional recipes—ensuring they remain accessible, relevant, and appreciated by future generations both locally and globally.

2. Educational Resource and Culinary Learning Tool

The app functions as an educational platform for users at all skill levels. It offers insights into Ethiopian ingredients, cooking methods, and meal customs. By providing detailed instructions, visual guides, and cultural context, it empowers users to deepen their knowledge and confidently prepare authentic dishes.

3. Convenience and Accessibility in Food Preparation

Designed with user-centric features like smart filters (e.g., meal type, fasting status, ingredient availability), customizable recipes, shopping list generators, and meal planners, the app enhances day-to-day meal preparation. It overcomes the challenges of scattered or inconsistent online resources by centralizing everything in one intuitive platform.

4. Community Engagement and Cultural Crowdsourcing

Incorporating features for recipe submissions, reviews, and community ratings, the app fosters an interactive and inclusive environment. This encourages users to share family traditions, learn from others, and collectively shape a growing archive of Ethiopian culinary knowledge—preserving regional variations and promoting shared identity.

5. Cross-Cultural Exchange and Global Appreciation

By presenting Ethiopian recipes in both Amharic and English, the app invites international users to explore and appreciate Ethiopia’s unique cuisine. This opens doors for cultural exchange, food diplomacy, and greater global recognition of Ethiopian culinary art.

6. Support for the Ethiopian Diaspora

For Ethiopians living abroad, the app acts as a cultural bridge—helping them reconnect with their roots, recreate familiar meals, and share their heritage with friends, family, or new communities. It can play a powerful role in maintaining identity and passing traditions across generations.

7. Potential for Economic and Educational Expansion

Looking ahead, the app could evolve into a broader digital ecosystem including:

- Virtual cooking classes
- Collaborations with chefs and local restaurants

- Tourism tie-ins promoting Ethiopian food culture
- Culinary certifications or educational modules

These expansions position the app as not just a recipe tool, but a platform for entrepreneurship, learning, and cultural influence.

8. Technological Innovation in Cultural Applications

By utilizing Flutter and Firebase, the app exemplifies how accessible, modern technology can be harnessed to solve real-world problems—preserving heritage, promoting healthy living, and enhancing user experience through scalable, cost-effective mobile development.

1.9. Beneficiaries of the Project

The Ethiopian Food Recipe App is designed to provide value to a diverse range of users, with direct and indirect benefits extending across individual, cultural, and entrepreneurial levels:

Everyday Users and Home Cooks

Home cooks and everyday users benefit from organized, step-by-step guidance on how to prepare traditional Ethiopian meals. Features such as recipe filters (by meal type, difficulty, and fasting status), meal planners, and shopping lists promote convenience and enhance everyday meal planning and preparation.

Students and Culinary Learners

Aspiring chefs, culinary students, and anyone eager to learn Ethiopian cooking techniques can utilize the app as a digital classroom. The educational content, such as ingredient explanations and preparation steps, supports practical learning in a structured format.

Ethiopian Diaspora and Global Audience

Ethiopians living abroad, as well as international food enthusiasts, gain reliable access to authentic recipes. The app serves as a cultural connection tool, allowing users to explore or maintain ties with Ethiopian culinary traditions.

Health-Conscious Individuals

Users with dietary restrictions or those observing religious fasts can benefit from personalized recipe filtering. This promotes healthy eating habits by enabling informed choices based on nutritional needs or fasting guidelines.

Restaurants and Food Entrepreneurs

Small food businesses, caterers, and restaurant owners can use the app for culinary inspiration, adapting or showcasing their own recipes. The platform may also support collaborations for promoting local ingredients or regional specialties.

Cultural and Educational Institutions

By preserving traditional recipes in a digital format, the app serves as a resource for cultural institutions, schools, and heritage programs focused on documenting and sharing Ethiopian culture through food.

1.10. Methodology

To effectively develop and deliver the Ethiopian Food Recipe App within the project timeline, an Agile development methodology was employed. This approach supports flexibility, continuous feedback, and iterative progress, making it ideal for fast-paced and evolving software projects.

Overall Strategy

- Agile Development: The app was developed through four iterative sprints over a two-week period. Each sprint involved planning, implementation, testing, and review.

- Team Collaboration: Daily stand-ups and progress reviews ensured synchronization, timely issue resolution, and shared ownership of outcomes.

- User Feedback Integration: Regular testing and peer reviews allowed real-time adjustments to improve functionality and user experience.

Key Development Phases

1. Planning
    - Conducted a project kickoff meeting to define scope, roles, and deliverables.
    - Set up Git for version control and Google Docs for collaborative documentation.
    - Established a two-week development schedule with daily milestones.
2. Design
    - Defined system architecture using a mobile-first design approach.
    - Created wireframes and interactive UI/UX mockups (using Figma) focusing on recipe presentation, filters, and planning tools.
    - Structured the database around categories, user profiles, recipes, and meal planning data.
3. Development
    - Frontend: Developed user interface components using Flutter with a focus on responsiveness and usability.
    - Backend: Integrated Firebase Firestore for real-time data storage and retrieval.

    - Implemented custom features including ingredient swapping, recipe saving, and dietary filtering.

4. Testing
    - Performed unit testing using Dart's testing tools for Flutter widgets and logic.
    - Conducted integration testing to ensure seamless interaction between UI and backend.
    - Gathered user feedback through peer testing and incorporated improvements.
5. Evaluation
    - Final project review and performance evaluation.
    - Documented learnings, design challenges, and recommendations for future development.

1.11. Development Tools and Technologies

The project leveraged a modern and scalable technology stack to ensure efficient development and future extensibility:

Programming Languages

- Dart: Used with Flutter for cross-platform mobile development
- JavaScript, HTML, CSS: Supporting technologies for possible backend extensions or web integration

Frameworks and Libraries

- Flutter SDK: Core development framework for UI design and functionality
- Firebase Firestore: Real-time, NoSQL cloud database for scalable data handling
- Express.js (Optional): For potential backend API development

Development Environment

- Visual Studio Code: Primary code editor
- Node.js & npm: Used for dependency management and optional backend utilities
- Windows OS: Development platform

Design and Collaboration Tools

- Figma: For wireframing, prototyping, and UI/UX mockup creation
- Git & GitHub: Version control and codebase collaboration
- Google Docs: Documentation and planning

Testing Tools

- Flutter Testing Framework: For unit and integration testing
- Chrome Browser & Emulators: For real-time interface testing

Project Management Tools

- Trello/Jira: (Optional) For task management, sprint planning, and workflow tracking

1.12. Required Resources and Cost Estimation

This project leverages primarily voluntary contributions and open-source tools, making it cost-effective. The breakdown of required resources is as follows:

Material and Software Resources

Resource	Description	Estimated Cost (ETB)
Development Machines	Personal laptops used for development	Already available
Software Tools	VS Code, Git, Firebase (Free Tier), Figma	0
Internet Access	Used for collaboration, deployment, and remote testing	~wifi available in the campus

1.13. Task Breakdown and Project Schedule

The project is structured into a 2-week sprint-based schedule, divided into key milestones and development phases:

Week 1: Planning, Design, and Initial Setup

Days	Activities
Day 1–2	Project kickoff, requirement gathering, team role assignment
Day 3–4	UI/UX wireframing, system architecture design, and Firebase schema setup
Day 5–6	Flutter and Firebase environment setup; development of initial recipe screens

Week 2: Development, Testing, and Finalization

Days	Activities
Day 7–8	Complete implementation: recipe features, filters, meal planner, shopping list
Day 9–10	Backend integration, unit and widget testing
Day 11–12	User testing and feedback incorporation, UI polishing
Day 13–14	Final evaluation, project documentation compilation, and submission

:

Chapter 2: Description of Existing System and Literature Review

2.1. Major Functions of the Existing System

The Ethiopian Food Recipe App aims to provide users with a digital solution for discovering, organizing, and preparing Ethiopian meals. The existing systems or comparable solutions may provide similar functionalities; however, for this app, we are focusing on key features that differentiate it in terms of accessibility, cultural relevance, and personalization.

Input:

- User Preferences: Users can input preferences such as dietary needs (e.g., vegetarian, gluten-free), cuisine type (Ethiopian, traditional), or specific ingredients they have.

- Recipe Data: Detailed information about each recipe, including ingredients, preparation steps, cooking time, and nutrition facts.

Process:

- Recipe Search and Filtering: Users can search for recipes based on filters such as difficulty level, time, meal type (breakfast, lunch, dinner), or fasting status (e.g., Lent).

- Customization: Recipes can be tailored by adjusting portion sizes or substituting ingredients based on availability.

- Meal Planning: The system allows users to generate a meal plan based on their weekly preferences and automatically suggests recipes.

Output:

- Customized Recipes: Tailored recipe suggestions based on user input, preferences, and available ingredients.

- Shopping List: Generates a shopping list for the recipes users select, ensuring efficient grocery shopping.

- Meal Planner: A daily or weekly meal planner to assist in organized meal preparation, saving time for users.

Service:

- User-Friendly Interface: The app offers an intuitive interface for both beginners and experienced cooks.

- Recipe Database: A constantly updated database of Ethiopian recipes, including new and traditional dishes.

- Reminders & Notifications: Users can set reminders for meal prep or ingredients that need to be purchased.

2.2. Users of the Current System

The Ethiopian Food Recipe App is designed for a wide range of users, each with specific needs, expectations, and preferences. These users are categorized as follows:

1. Home Cooks (Beginner & Experienced)
    - Needs:
        - Easy access to a variety of traditional Ethiopian recipes.
        - Ability to modify recipes based on personal preferences or dietary restrictions.
    - Expectations:
        - Simple navigation and clear recipe steps.

        - Personalized recipe suggestions.
    - Preferences:
        - Ability to save favorite recipes.
        - Option to set reminders for meal prep and ingredient shopping.
2. Families & Meal Planners
    - Needs:
        - Meal planning features to help organize family meals.
        - Shopping lists that align with weekly meal planning.
    - Expectations:
        - Time-saving features such as auto-generated meal plans and shopping lists.
    - Preferences:
        - Ability to filter recipes based on meal type (e.g., vegetarian, non-vegetarian).
3. Health-Conscious Users
    - Needs:
        - Nutritional information for each recipe.
        - Fasting-status filters for users observing religious fasting periods.
    - Expectations:
        - Transparent nutritional data.
        - Recipes tailored to specific dietary needs.

- Preferences:
    - Nutritional analysis based on ingredients used in the recipe.

4. Cultural Enthusiasts and Tourists
    - Needs:
        - Access to authentic Ethiopian recipes.
    - Expectations:
        - A deep dive into the cultural significance of each dish.
    - Preferences:
        - Integration of cultural notes and tips about traditional Ethiopian cooking.

2.3. Drawbacks of the Current System

While existing recipe apps provide basic functionalities, there are several limitations in the current systems that this Ethiopian Food Recipe App aims to address:

1. Limited Customization: Many existing recipe apps do not allow enough customization in recipes to accommodate diverse dietary preferences or ingredient substitutions.

2. Lack of Cultural Relevance: Current apps often lack authentic Ethiopian dishes, leaving a gap in culturally specific cooking resources.

3. Insufficient Filtering Options: Some recipe apps do not provide filters that account for specific needs, such as fasting status or traditional meal types.

4. Poor Meal Planning Integration: Many apps lack a comprehensive meal planner, which can help users organize their meals ahead of time.

5. Limited Nutritional Insights: Existing systems may not provide adequate nutritional breakdowns or tailored recommendations for users with health-conscious goals.

The new Ethiopian Food Recipe App seeks to overcome these limitations by offering features such as full recipe customization, a culturally rich recipe database, and meal planning tools that cater to different dietary and cultural needs.

2.4. Literature Review

The literature review for the Ethiopian Food Recipe App focuses on the existing body of work concerning recipe management systems, mobile food apps, and the integration of cultural food knowledge in digital platforms. A few key points are highlighted below:

1. Recipe Management Systems:
    - Various studies have shown the importance of recipe management systems in enhancing cooking experiences by offering efficient tools for searching, saving, and sharing recipes (Smith et al., 2019). These systems are typically designed to help users organize their culinary practices and simplify the cooking process.
    - Customization is a growing trend in these systems, allowing users to adjust ingredients and meal plans to their dietary preferences (Jones & Harris, 2020).
2. Mobile Food Apps:
    - Mobile recipe apps have surged in popularity due to their convenience and accessibility (Gosling & Baird, 2021). Features such as recipe search, shopping lists, and step-by-step instructions have been implemented in various food apps to improve user experience.
    - However, there is a gap in the market for apps that focus on specific cuisines, such as Ethiopian food, which often lack comprehensive and accurate representations of traditional recipes (Anderson, 2022).
3. Cultural Food Apps:
    - Cultural food apps that offer users an immersive culinary experience are becoming more popular, particularly in regions with diverse food traditions (Khan & Meyer, 2020). Such apps integrate cultural history, food customs, and authentic cooking methods, providing not just recipes but also a cultural learning experience.
    - In the case of Ethiopian cuisine, there is a need for digital platforms that accurately portray the richness of Ethiopian dishes, along with their historical and cultural context.

The review indicates a clear opportunity for a specialized app that addresses these gaps by focusing on Ethiopian traditional food, offering customization options, and integrating meal planning and nutritional insights.

Chapter 3: Proposed System

3.1. Overview

The Ethiopian Food Recipe App is a mobile application designed to enhance the cooking experience by providing users with access to a diverse collection of traditional Ethiopian recipes. The app focuses on personalization, allowing users to customize recipes, plan meals, and browse an extensive recipe database that caters to various dietary preferences and ingredient availability. By incorporating features such as advanced recipe search, meal planning, and shopping list generation, the app provides an all-in-one solution for users who want to simplify meal preparation. Whether users are beginners exploring new dishes or experienced cooks looking for easy-to-follow recipes, the app serves as an invaluable kitchen companion. It enables users to search, save, and share recipes, while also suggesting weekly meal plans that align with individual preferences and dietary needs, including fasting requirements.
Additionally, the app's user-friendly interface ensures a seamless and intuitive experience for all users, whether on iOS or Android platforms. Overall, the app is designed to help users discover, organize, and prepare traditional Ethiopian meals effortlessly, while offering flexibility to meet diverse tastes and nutritional goals.

3.2. Functional Requirements

1. User Registration and Authentication:
    - Users should be able to register and create profiles using Firebase Authentication.
    - Differentiate user roles (e.g., general users, admin users) for personalized access.
    - Secure login with features for password recovery and profile management.
2. Recipe Search and Customization:
    - Users can search for recipes based on various filters such as ingredients, cuisine type, difficulty, and cooking time.
    - Customization options for dietary preferences (e.g., vegetarian, gluten-free) will be available.

- Users can save their favorite recipes for quick access.

3. Meal Planning and Recommendations:

    - The app will suggest weekly meal plans based on user preferences, including fasting status or meal type.

    - Users can create and manage meal plans for specific days or weeks, with automated recipe suggestions.

    - Meal planners will assist in organizing meals in advance to align with dietary or nutritional goals.

4. Shopping List Generation:

    - The app will automatically generate shopping lists based on the recipes selected by users.

    - The shopping list will include all necessary ingredients, and users can manually add additional items.

    - Items in the shopping list can be checked off during shopping for easy tracking.

5. Fasting Status and Dietary Preferences:

    - The app will allow users to filter recipes according to fasting requirements (e.g., Lent, Orthodox fasting days).

    - Additional dietary preferences can be specified, such as low-fat, high-protein, or gluten-free.

6. Recipe Sharing and Community Features:

    - Users can share their favorite recipes and meal ideas with others within the app or on social media.

    - A rating and review system will allow users to provide feedback on recipes, fostering a community of sharing and learning.

7. Push Notifications and Reminders:
    - Users will receive timely notifications for meal reminders, shopping list updates, and meal preparation alerts.
    - These reminders will help ensure users stay on track with meal planning and preparation.
8. Profile Management:
    - Users can manage their profiles, update dietary restrictions, preferences, and saved recipes.
    - Admin users will have access to manage user data and maintain app content.
9. Admin User Management:
    - Admin users can oversee the app's content, manage user profiles, and handle moderation of submitted recipes.
    - Admins will also monitor user activity and feedback to continuously improve the app's offerings.
10. Recipe History:
    - Users will have access to a history of the recipes they've saved or tried.
    - This feature enables easy revisiting of previously enjoyed recipes.

3.3. Non-Functional Requirements

1. Performance:
    - The app should handle large numbers of concurrent users and recipe queries with minimal delays.
    - Recipe loading and app responsiveness must be optimized for fast navigation and user interaction.

2. Reliability:
    - The system should ensure continuous uptime with minimal downtime, allowing users to access recipes, shopping lists, and meal plans at all times.
    - Data integrity will be essential to prevent loss of user profiles, saved recipes, or shopping lists.
3. Usability:
    - The app should feature an intuitive user interface that is easy to navigate for both novice and experienced cooks.
    - Clear instructions and intuitive design will guide users through recipe customization, meal planning, and shopping list generation.
4. Scalability:
    - The system should scale seamlessly as the user base and recipe database grow over time.
    - Future versions of the app should support new features like voice-assisted cooking instructions or integrations with grocery delivery services.
5. Security:
    - User data, including profiles and preferences, must be encrypted and securely stored.
    - Payment information, if applicable, and shopping list data must be securely processed.
    - The app will undergo regular updates to maintain security and data protection.
6. Regulatory Compliance:
    - The app must comply with relevant data protection laws, ensuring the privacy and security of user data.

- It will also ensure that all dietary information provided is accurate and legally compliant, particularly when considering local food laws.

7. Compatibility:
    - The app should be fully compatible with various devices and operating systems (iOS, Android), ensuring a consistent experience across platforms.
    - It will be optimized for various screen sizes, from smartphones to tablets.
8. Accessibility:
    - The app will adhere to accessibility guidelines, ensuring that it is usable by individuals with disabilities.
    - Features like voice assistance and screen reader compatibility will be integrated to improve accessibility for all users.

3.4. System Model

The Ethiopian Food Recipe App will leverage Firebase as its backend-as-a-service (BaaS) platform, providing scalability, security, and enhanced performance. The app will integrate several Firebase services to ensure smooth operation and user experience.

Key Components:

1. Mobile Application (Client):
    - Developed using Flutter for cross-platform support.
    - Provides the user interface (UI) and interaction layer.
2. Firebase Firestore:
    - A NoSQL cloud database that stores essential data such as recipes, user profiles, reviews, ratings, and nutritional information.
3. Firebase Authentication:
    - Manages user authentication, registration, and account security.

4. Firebase Storage:
    - Stores high-quality images of recipes, ensuring fast and seamless loading for users.
5. Firebase Cloud Functions:
    - Handles backend logic such as data validation, recipe moderation, average rating calculation, and push notifications.

The system will support a variety of functionalities, including browsing, searching, saving, submitting, rating, and reviewing recipes, all integrated seamlessly with Firebase services.

3.4.1. Scenario

Scenario 1: A Logged-in User Submits a New Recipe

1. The logged-in user navigates to the "Submit Recipe" screen.
2. The user fills in all necessary recipe details (name, description, category, ingredients, instructions, time, serving size, dietary restrictions) and optionally uploads images.
3. The user taps the "Submit" button.
4. The mobile application sends the recipe data (including image URLs from Firebase Storage) to Firebase Firestore, marking the recipe as "pending moderation."
5. (Optional) A Firebase Cloud Function is triggered, notifying administrators or performing initial data validation.
6. An administrator reviews the submitted recipe and, if appropriate, updates its status to "published."
7. Once "published," the recipe becomes visible to all users in the browsing and search functionalities.

Scenario 2: A Logged-in User Rates and Reviews a Recipe

1. The user views a recipe detail screen.
2. The user submits a rating by interacting with the rating component (e.g., tapping stars) and enters a review.

3. The mobile application sends the rating and review data (including user ID and recipe ID) to Firebase Firestore.

4. (Optional) A Firebase Cloud Function is triggered to recalculate the average rating and update the recipe's rating field in Firestore.

5. The updated average rating and the new review appear on the recipe detail screen.

3.4.2. Use Case Model

Actors:

- User: Any individual interacting with the mobile application.
- Registered User: A user authenticated via Firebase Authentication.
- Administrator: Manages recipe data, user submissions, and system functionality.

Use Cases:

- Browse Recipes (Firestore): Users view categorized lists of recipes fetched from Firestore.
- Search Recipes (Firestore): Users search for recipes using keywords and filters.
- View Recipe Details (Firestore, Storage): Users view detailed recipe information, including images from Firebase Storage.
- Register Account (Firebase Authentication): Users create new accounts.
- Login (Firebase Authentication): Users log into their accounts.
- Save Favorite Recipe (Firestore): Registered users save recipes to their profiles in Firestore.
- View Favorite Recipes (Firestore): Users view saved recipes from their profiles.
- Submit Recipe (Firestore, Storage, Cloud Functions): Logged-in users submit new recipes, including image uploads and data storage, triggering Cloud Functions for moderation.
- Moderate Recipes (Firestore, Cloud Functions): Administrators review and approve user-submitted recipes, possibly using Firebase Cloud Functions for checks.

- Rate Recipe (Firestore, Cloud Functions): Users rate recipes, triggering Cloud Functions to update average ratings.

- Review Recipe (Firestore): Users write and submit reviews for recipes.

- View Nutritional Information (Firestore): Users view nutritional information for recipes.

3.4.3. Conceptual Model

Entities:

- User: Represents an individual who interacts with the app, searches, saves, customizes, and rates recipes.

- Admin: Manages recipes, user accounts, and ensures smooth app functionality.

- Recipe: A collection of ingredients, instructions, and related information, available for search, customization, and saving by users.

- Meal Plan: A personalized schedule of meals based on the user's dietary preferences and needs.

- Shopping List: A list generated from selected recipes, detailing the ingredients needed for cooking.

Interactions:

- User-Recipe: Users can search, save, and rate recipes.

- User-Meal Plan: Users can create personalized meal plans.

- User-Shopping List: Users generate shopping lists based on selected recipes.

- Admin-Recipe: Admins manage the recipe database, ensuring accuracy and relevance.

- Admin-User: Admins monitor and manage user accounts.

Functional Flow:

- User Registration and Login: Users create accounts, log in, and access app features.

- Recipe Search and Customization: Users search and customize recipes based on dietary needs.
- Meal Plan Creation: Users create personalized meal plans.
- Shopping List Generation: Users generate shopping lists based on meal plans.
- Admin Management: Admins manage recipe content and user accounts.

3.4.4. Logical Model

Entities and Attributes:

- User: UserID, Name, Email, Password, Preferences (dietary restrictions, fasting status).
- Admin: AdminID, Name, Role (Admin, Moderator), Access Level (Full, Partial).
- Recipe: RecipeID, Name, Ingredients, Instructions, MealType, FastingStatus.
- Meal Plan: PlanID, UserID, Recipes (list of RecipeIDs), WeekStartDate, DietaryNeeds.
- Shopping List: ListID, UserID, RecipeIDs, IngredientsList.

Relationships:

- User-Recipe: One-to-Many (A user can save multiple recipes).
- User-Meal Plan: One-to-Many (A user can create multiple meal plans).
- Recipe-Ingredient: One-to-Many (A recipe can have multiple ingredients).
- Admin-Recipe: One-to-Many (Admins manage multiple recipes).
- User-Shopping List: One-to-Many (A user can generate multiple shopping lists).

Data Flow:

- User Registration and Login: Input: User details → Process: Validate and create user record → Output: Confirmation and login credentials.
- Recipe Search and Customization: Input: Search criteria → Process: Retrieve matching recipes → Output: List of customized recipes.

- Meal Plan Creation: Input: User-selected recipes → Process: Store in a personalized meal plan → Output: Customized meal plan.

- Shopping List Generation: Input: Selected recipes → Process: Extract ingredients → Output: Shopping list.

- Admin Management: Input: Admin actions → Process: Update recipe data and user accounts → Output: Updated content.

3.4.5. Physical Model

System Components:

- Frontend: Built using React.js, providing UI for browsing, meal planning, and shopping list generation.

- Backend: Powered by Node.js, handling API interactions and logic processing.

- APIs: Manage user data, recipes, meal plans, and shopping lists.

- Database: A NoSQL database (e.g., MongoDB) stores user, recipe, meal plan, and shopping list data.

Deployment Architecture:

- Web Server: Hosts the React.js frontend and handles user requests.

- Application Server: Runs the Node.js backend, processing API requests.

- Database Server: Hosts the NoSQL database (MongoDB), ensuring data persistence.

3.4.6. Scenario Model

Scenario 1: User Registers and Creates an Account

- Entry Condition: User wishes to register.

- Flow of Events: User provides info → System validates and creates account → System sends verification → User completes verification → Account is created and user is logged in.

- Exit Condition: Successful account creation and login.

Scenario 2: User Searches and Customizes a Recipe

- Entry Condition: User wants to search and customize a recipe.
- Flow of Events: User searches → System displays recipes → User customizes → System updates recipe → User saves.
- Exit Condition: Customized recipe saved.

Scenario 3: User Creates a Meal Plan

- Entry Condition: User wishes to create a meal plan.
- Flow of Events: User selects recipes → System suggests meals → User customizes → Plan saved.
- Exit Condition: Meal plan saved.

Scenario 4: User Generates a Shopping List

- Entry Condition: User wants a shopping list.
- Flow of Events: User selects recipes → System generates list → User reviews and confirms → List ready for use.
- Exit Condition: Shopping list ready.

Scenario 5: Admin Manages Recipe Content

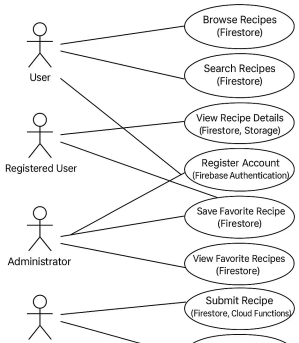
- Entry Condition: Admin manages recipes.
- Flow of Events: Admin logs in → Adds/edits/deletes recipes → System updates → Admin logs out.
- Exit Condition: Recipe database updated.

3.4.7. Use Case Model

Actors:

- User: Can register, log in, browse recipes, customize, create meal plans, and generate shopping lists.
- Admin: Manages recipe content and user accounts.

Use Cases:



- User Registration and Login
- Search Recipes
- Customize Recipes
- Create Meal Plans
- Generate Shopping Lists
- Manage Recipes (Admin)
- Manage User Accounts (Admin)
-

Fig 3.4.7 Use case model

3.5. Object Model

The Object Model represents the key objects, their attributes, and how they interact within the system. The Ethiopian Food Recipe App is built around several central objects such as User, Recipe, MealPlan, ShoppingList, and Admin. These objects interact with one another through various methods to provide a seamless experience for the users.
Key objects:

1. User: Represents the app's user. This object manages user details and preferences.

2. Recipe: Contains details about recipes, including ingredients, cooking instructions, and difficulty.

3. MealPlan: Allows users to organize their weekly meal schedule, storing recipes within a meal plan.

4. ShoppingList: A feature to generate shopping lists based on selected recipes.

5. Admin: Manages recipe content and user-generated content, including moderating and deleting inappropriate entries.

3.5.1. Data Dictionary

A Data Dictionary defines the data elements in the system.

Data Element	Type	Format	Description	Example Values
RecipeID	Integer	-	Unique identifier for a recipe.	101, 102

RecipeName	String	1-100 chars	Name of the recipe.	"Doro Wat", "Injera"
Ingredients	String	-	List of ingredients for the recipe.	"Chicken, Berbere, Garlic"
CuisineType	String	1-50 chars	Type of cuisine (e.g., Ethiopian, Mediterranean).	"Ethiopian", "Italian"
DifficultyLevel	String	-	Level of recipe difficulty.	"Easy", "Medium", "Hard"
CookingTime	Integer	Minutes	Time required to cook the recipe.	30, 45
MealPlanID	Integer	-	Identifier for a meal plan.	201, 202
ShoppingList	String	-	List of ingredients needed for shopping.	"Tomatoes, Garlic, Berbere"
RecipeInstructions	String	-	Step-by-step cooking instructions for the recipe.	"1. Cook chicken... 2. Add spices..."

3.5.2. Class Diagram

A Class Diagram shows the relationships between the main objects in the system. Below is a simplified overview:

- Recipe:
    - Attributes: RecipeID, RecipeName, Ingredients, CuisineType, DifficultyLevel, CookingTime, RecipeInstructions
    - Methods: viewRecipe(), customizeRecipe(), saveRecipe(), rateRecipe()
- User:
    - Attributes: UserID, UserName, Email, Preferences
    - Methods: login(), saveFavoriteRecipe(), createMealPlan(), rateRecipe()
- MealPlan:

- Attributes: MealPlanID, Recipes, MealDates
- Methods: addRecipe(), removeRecipe(), viewMealPlan()

- ShoppingList:
  - Attributes: ListID, UserID, RecipeID, Ingredients
  - Methods: addItem(), generateShoppingList(), removeItem()

- Admin:
  - Attributes: AdminID, AdminName
  - Methods: approveRecipe(), deleteRecipe(), moderateContent()

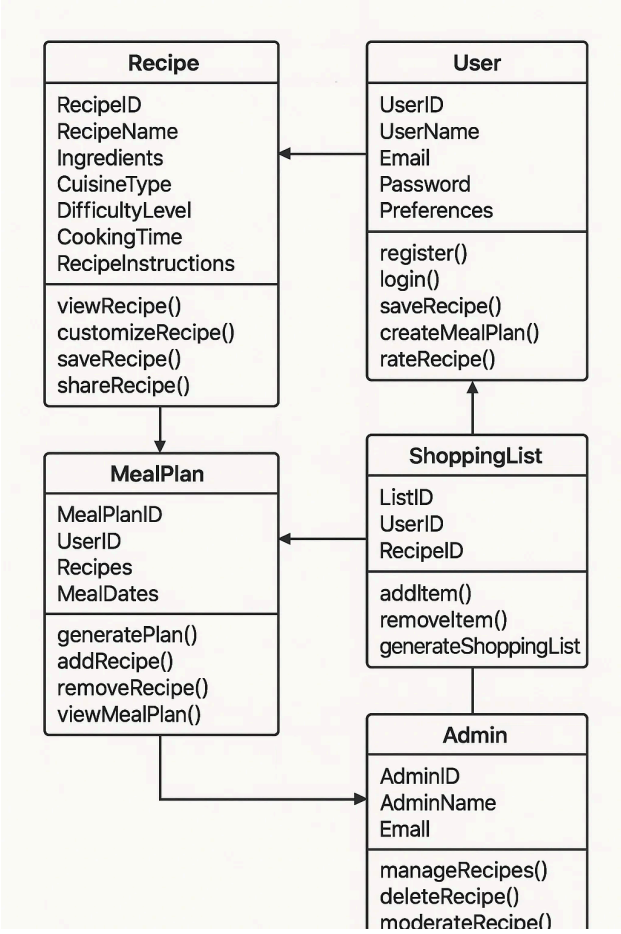


Fig 3.5.2 class diagram

3.5.3. Dynamic Model

The Dynamic Model illustrates how different system components interact dynamically.

- Login Process:
    - Event: The user initiates a login request.
    - Transition: The system verifies the credentials.
    - Action: If valid, the user is redirected to the dashboard.
    - State Change: User state changes from "Not Logged In" to "Logged In".

- Recipe Customization:
    - Event: User selects a recipe to modify.
    - Transition: The app offers customization options.
    - Action: User customizes ingredients or instructions.
    - State Change: Recipe state changes to "Customized".

3.5.4. Sequence Diagram

The Sequence Diagram shows the sequence of messages or communications between objects/components during specific actions. Let's take a look at the User Customizing a Recipe sequence:

1. User sends a request to view a recipe (viewRecipe()).
2. The App Backend fetches the recipe from the database.
3. The App responds to the User with the recipe details.
4. User selects customization options and sends a request to customize (customizeRecipe()).
5. The App updates the recipe and stores the customized version in the database.
6. The App confirms the changes to the User.

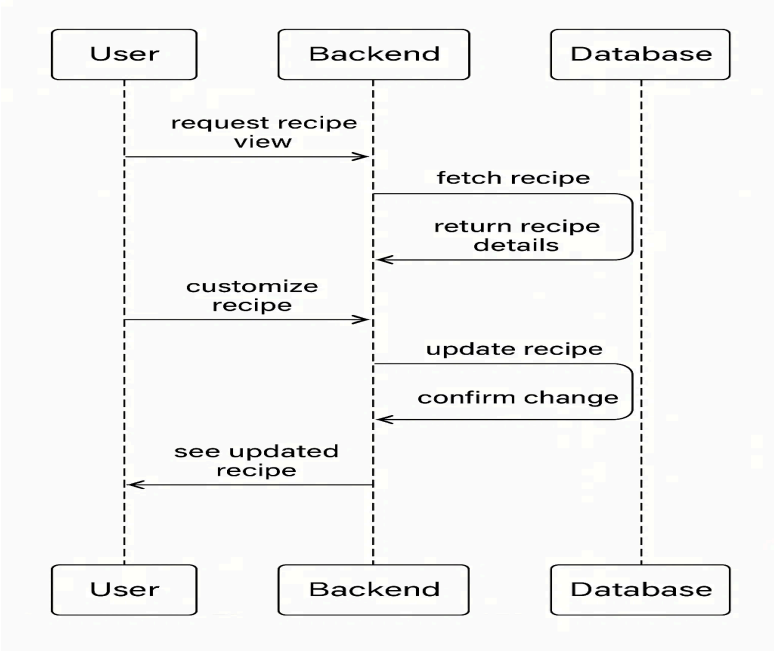
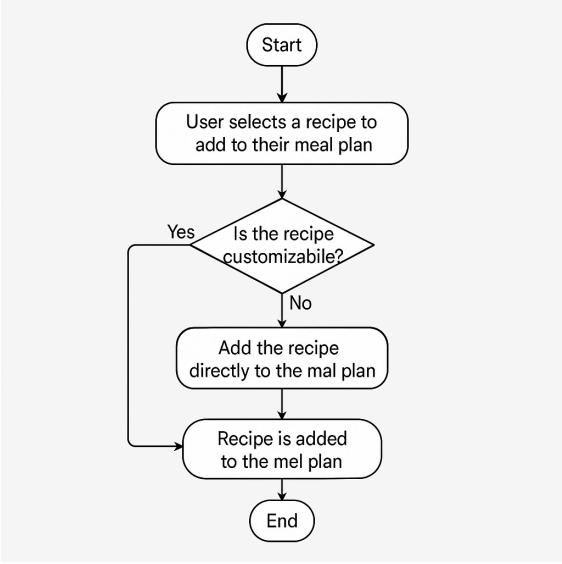


Fig 3.5.4 sequence diagram

3.5.5. Activity Diagram

An Activity Diagram shows the flow of activities during a specific task, such as Adding a Recipe to a Meal Plan:



1. Start: User selects a recipe.
2. Decision: Is the recipe customizable?
    - Yes: Present customization options.
    - No: Add the recipe directly.
3. Action: Recipe added to the meal plan.
4. End: Process complete.

Fig 3.5.5 Activity Diagram

3.5.6. State Chart Diagram

A State Chart Diagram shows the various states of a system component, such as the Recipe object:

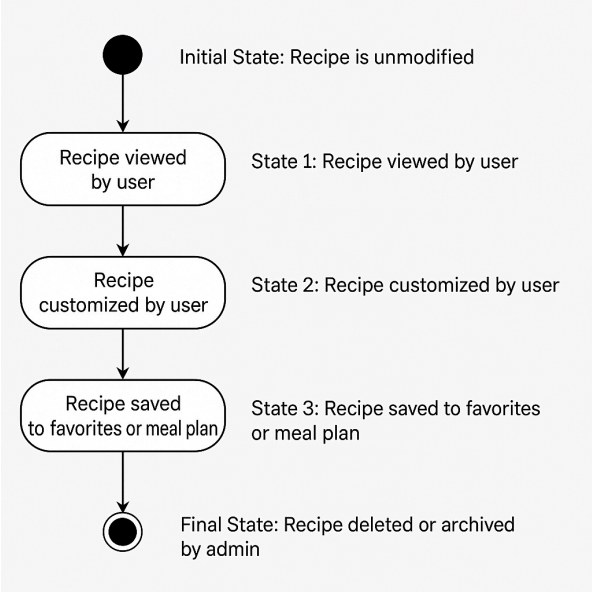


Fig 3.5.6 State chart diagram

Chapter 4: System Design

4.1 Overview

The Ethiopian Food Recipe App is designed with a modular, user-centered architecture that supports recipe discovery, customization, meal planning, and personal dietary management. It integrates a responsive mobile frontend with a scalable backend and cloud-based services. The main system components include:

- User Interface (UI): A mobile-friendly interface built using Flutter.
- Backend Services: Logic for user authentication, recipe management, filtering, and meal planning.
- Database Layer: Stores user data, recipes, preferences, and schedules using Firestore or MongoDB.
- External Services: Push notifications (Firebase Cloud Messaging), and authentication (Firebase Auth).

These components interact via RESTful APIs in a well-structured 3-tier (Layered) Architecture, ensuring scalability, maintainability, and performance.

4.2 Purpose of the System Design

The purpose of the system design is to serve as a clear technical blueprint that:

- Translates user requirements into system structure and logic.
- Supports smooth development, testing, and maintenance.
- Ensures modularity and clear separation of concerns.
- Aligns with both the object model and functional use cases.
- Helps deliver features like meal filtering, customization, shopping list generation, and dietary preferences.

4.3 Design Goals

The Ethiopian Food Recipe App is designed with the following software engineering principles:

Design Goal	Explanation
Modularity	Each feature (e.g., recipes, planner, notifications) is in its own module.
Cohesion	Each module has a focused, well-defined responsibility.
Low Coupling	Modules communicate via interfaces with minimal dependency.
Encapsulation	Data and methods are bundled within specific components or classes.
Abstraction	Complex logic (e.g., filtering) is hidden behind service layers.
Reusability	Common services like filtering and search are designed to be reused.
Inheritance & Polymorphism	Applied in user roles (e.g., Admin and Regular User inherit from User).

4.4 Proposed System Architecture

Architecture Style:

The app uses a Layered Architecture with clear responsibility for each layer:

1. Presentation Layer:
    - Built with Flutter for cross-platform UI.
    - Handles user inputs, displays data, and manages navigation.
2. Application Layer (Business Logic):
    - Processes inputs from the UI.
    - Handles filtering, recipe suggestions, planning, shopping list logic, and user settings.
3. Data Layer (Persistence):
    - Stores data using Firestore or MongoDB.
    - Includes repositories for users, recipes, and plans.
4. External Services:
    - Firebase Auth: User login/signup.
    - Firebase Cloud Messaging: Push notifications/reminders.
    - Cloud Storage: For images, videos of recipes.

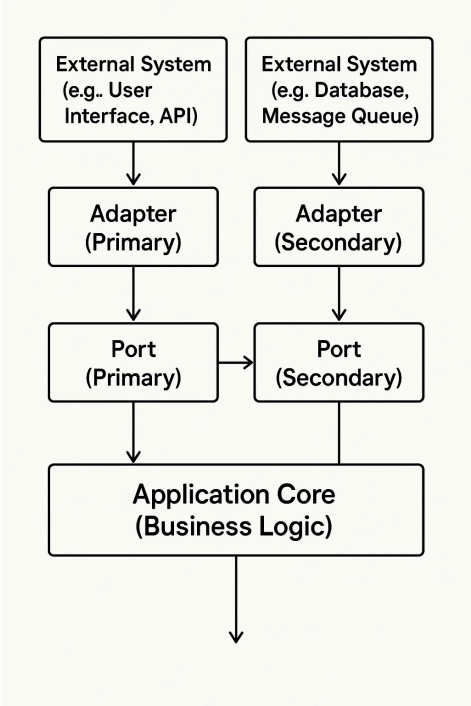


Fig 4.4 system architecture

4.5 Subsystem Decomposition

To enhance clarity and maintainability, the system is divided into the following logical subsystems:

Subsystem	Responsibilities
User Management	Manages registration, login, profile updates, and user preferences.
Recipe Management	Handles adding, editing, and retrieving recipes. Supports filtering & customization.
Meal Planner	Generates weekly meal plans based on user preferences and fasting status.
Shopping List Generator	Builds shopping lists from selected recipes.
Reminder & Notification	Sends reminders for meal preparation and planning.
Admin Dashboard	(Optional) For content management such as adding/editing recipes.

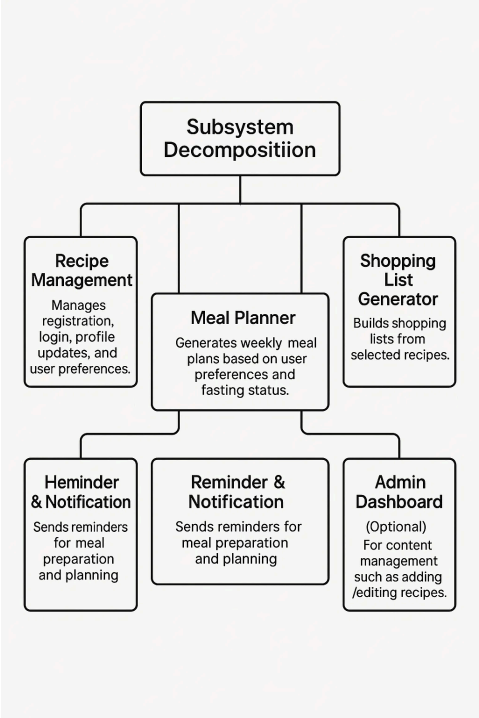


Fig4.5 subsystem decomposition

4.6. Subsystem Description

Each subsystem in the app is responsible for a specific set of functionalities and interacts with others through defined interfaces to ensure smooth user experiences and maintainability.

Subsystem	Function	Interfaces & Interactions
User Management	Handles user registration, authentication, profile editing, and preferences.	Interfaces with Firebase Auth for authentication and user profile database to retrieve/update preferences.
Recipe Management	Stores, filters, and customizes recipes based on ingredients, fasting type, and preferences.	Interacts with the recipe database, allows querying based on filters (cuisine, cooking time), and communicates with the shopping list and meal planner subsystems.
Meal Planner	Builds weekly plans based on saved preferences, time, and fasting schedules.	Uses data from the user profile and recipe subsystem, and sets meal preparation reminders using the reminder subsystem.
Shopping List	Compiles ingredients from selected recipes into a list for users to use when shopping.	Interacts with the recipe management subsystem to extract ingredient data and stores it in a user-specific shopping list.

Reminder System	Sends notifications for upcoming meal prep and planning tasks.	Interfaces with Firebase Cloud Messaging and uses scheduled times from the planner module.
Admin Panel	Manages recipe uploads, app content moderation, and feedback review.	Has elevated access to recipe management and user feedback records in the backend database.

4.7. Persistent Data Management

The app requires persistent storage to ensure continuity of user sessions, preferences, and recipe data.

- Storage Platform: Firestore (cloud-based NoSQL) or MongoDB.
- Data Types Stored:
    - User Data: Profile info, preferences, saved recipes, fasting status.
    - Recipe Data: Name, description, ingredients, steps, meal type, image URL, tags.
    - Meal Plans: Daily/weekly plans linked to user accounts.
    - Shopping Lists: Dynamic data generated from selected recipes.
    - Notification Schedules: Stored and synced with Firebase Cloud Messaging.
- Features:
    - Real-time data sync for updates.
    - Offline access support (optional with local caching).
    - Indexing for faster recipe filtering and search.

4.8. Component Diagram

The component diagram shows the high-level structure of the system and its main functional blocks, which are developed as independently deployable modules.
Key Components:

- User Interface (Flutter-based)
- User Auth (Firebase Auth)
- Recipe Service (Backend logic for filtering, CRUD operations)
- Meal Planner Engine
- Shopping List Generator
- Notification Handler (Firebase Cloud Messaging)
- Database Manager (handles read/write ops to Firestore/MongoDB)

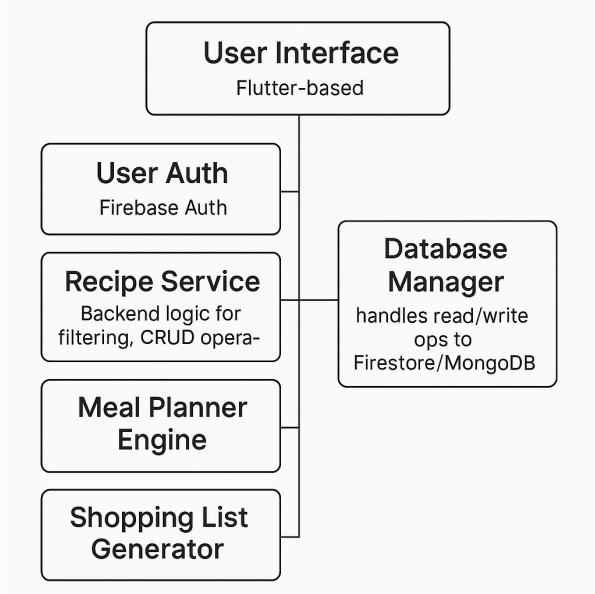


Fig 4.8 Component Diagram

4.9. Database Diagram

The app follows a document-oriented schema for storing data, optimized for flexibility and scalability.
Collections and Relationships (Example):

Collection	Fields
Users	uid, email, name, preferences, savedRecipes[], mealPlan[], shoppingList[]

Recipes	recipeId, title, ingredients[], instructions, mealType, fasting, time, difficulty, imageURL
MealPlans	userId, date, recipeId[], status (planned, completed)
Reminders	userId, recipeId, time, status

Relationships:

- Users ↔ Recipes: Many-to-many via savedRecipes.
- Users ↔ MealPlans: One-to-many.
- Recipes ↔ Ingredients: One-to-many (embedded array).

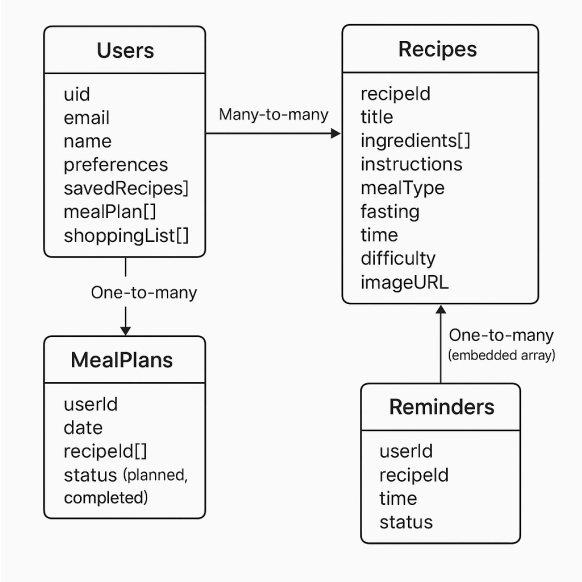


Fig 4.9 database diagram

4.10. Access Control

Security is enforced using role-based access control and encrypted communications.

Feature	Description
Authentication	Handled via Firebase Auth with email/password or third-party options.
Authorization	Admins have extended permissions (content management); users have restricted access to personal and public recipe data.
Data Encryption	All data in transit is encrypted using HTTPS.
Role-Based Access	Two roles: Admin and Regular User. Backend checks roles before allowing actions.
Token Management	JWTs (Firebase tokens) are used to manage sessions securely.

Conclusion

The Ethiopian Recipe app is a thoughtfully designed, all-in-one culinary platform that brings the rich and diverse flavors of Ethiopian cuisine to users' fingertips. By offering an extensive range of traditional and modern recipes—tailored to various tastes, dietary needs, and ingredient availability—the app makes it easy for users to discover, prepare, and enjoy authentic Ethiopian dishes. With powerful features such as recipe filtering, customization, meal planning, shopping list generation, and preparation reminders, the app ensures a seamless and enjoyable cooking experience.

Whether for beginners exploring Ethiopian cuisine for the first time or seasoned cooks looking to expand their repertoire, the Ethiopian Recipe app provides valuable tools that simplify meal preparation and encourage culinary exploration. Its intuitive interface and practical functionalities support not only daily cooking but also long-term meal planning and nutritional goals.

In summary, the Ethiopian Recipe app successfully combines cultural richness with modern convenience, making it an essential kitchen companion for anyone passionate about Ethiopian food. This documentation highlights the project's dedication to delivering a high-quality, user-centered mobile application that enhances and celebrates the experience of Ethiopian cooking.