Insta-foodies

Instagram Recipe App

Software Design Document

Orel Zamler

Harel Giladi

Aviel Edri

Eilon Barashi

Israel - 13/1/2023

Table of Contents

INTRODUCTION

The Smarter Foodies app targets food enthusiasts who want to share and enjoy food-related content responsibly. Unlike typical social apps, it prioritizes privacy and skill showcasing. To combat food waste (108 billion pounds yearly in the US), the app promotes responsible grocery shopping via accurate lists and online store suggestions, encouraging sustainability among users.

1.1 Purpose

The purpose of this Software Design Document (SDD) is to outline the design and development of a recipe-sharing application, akin to Instagram, but with a focus on recipes. This document serves as a comprehensive guide for the development team, elucidating the project's scope, goals, and objectives.

1.2 Scope

The project's scope encompasses the creation of an application that allows users to create and share recipe posts, closely resembling the functionalities of Instagram. The application will include features such as post creation and editing, following other users, searching for friends, and receiving notifications. A user-friendly interface will facilitate effortless navigation and interaction. The application will be developed using either Android Studio or React, with Firebase or SQLite as the database.

1.3 Overview

This SDD provides an extensive overview of the project's organization, development process, and system design. It delves into the system architecture, data design, component design, human interface design, and requirements matrix in detail.

1.4 Reference Material

The SDD takes inspiration from the Instagram App for Android, utilizing either Firebase or SQLite as the database. The application will be developed using React or Android Studio.

SYSTEM OVERVIEW

2.1 Project Goal

The core objective of the Smarter Foodies app is to establish a new food sharing platform that emphasizes efficient resource utilization and intelligent meal planning. The application's primary purpose is to function as a social app where users can seamlessly share recipes with their friends and followers. Users will be able to create and share recipe posts, follow other users, search for recipes, plan weekly meals, and even export their shopping lists as WhatsApp messages for their convenience. The app boasts a user-friendly interface that enables effortless navigation and interaction.

Additionally, the app facilitates the efficient organization of users' weekly meals by generating precise shopping lists for the ingredients needed for their preferred dishes.

2.2 Selected Approach

The Scrum methodology serves as the foundation of our project, enabling agility, collaboration, iterative improvement, and predictable delivery. Embracing Scrum allows us to swiftly adapt to changes, cultivate teamwork, continually enhance our processes, and ensure the project's delivery within a predictable timeline. The Scrum Master oversees this methodology, ensuring daily standup meetings to manage the project's progress effectively.

2.3 Solution Description

Smarter Foodies is a cutting-edge social app for food enthusiasts, delivering a delightful and unique experience. Empowered by a Node.js server with load balancing and multi-threading capabilities, our app guarantees optimal performance and streamlined user traffic management. Employing the Model-View-View Model (MVVM) architecture, we maintain a well-structured codebase, ensuring a seamless user interface and efficient data management. Advanced AI tools enhance user interactions through features like food classification, OCR, and audio-to-text capabilities. Our scraping mechanism aggregates recipes and related data, enriching the app's content. Smarter Foodies epitomizes innovation, user-friendliness, and an expansive recipe database to connect food lovers worldwide.

SYSTEM ARCHITECTURE

3.1 Architectural Design

The architectural design of the Smarter Foodies app follows the Model-View-View Model (MVVM) pattern. This pattern facilitates the separation of concerns and ensures easy codebase maintenance.

Model: This component encompasses both communication logic for interacting with the server and the logic for handling data from the server.

View: The View is responsible for the Android UI, layouts, and user interface elements.

View Model: The View Model manages the preparation of data from the Model for the UI and contains UI-related logic...

3.2 Decomposition Description

The app will be decomposed into several modules, each addressing specific functionalities:

1. User Authentication and Registration: This module manages user registration and authentication processes, encompassing tasks like validating user input, securely storing user credentials, and implementing email verification for secure authentication.

2. Post Creation and Editing: Users can create, edit, and manage their recipe posts. This module facilitates image uploading, text input, and structured storage of recipe data.

3. Feed Display and Navigation: Responsible for displaying the user's feed, including posts from followed users, recommended content, and trending posts. It enables user interaction with posts through likes, comments, and shares.

4. User Profile and Settings: This module handles user profiles, allowing users to view their profiles, update profile pictures, edit bios, and manage profile-related data.

5. Search Functionality: Users can search for other users using auto-complete suggestions. This module involves implementing user-friendly search queries and presenting search results effectively.

6. Notifications: This module manages notifications related to post interactions, such as likes, comments, follows, and other activities. Real-time updates and a notification center enhance user engagement.

7. Chat Messaging: Enables users to send and receive direct messages. It encompasses features like individual and group chats, message synchronization, multimedia handling, and real-time updates.

***8. Content Recognition and Reporting*:** Utilizes Firebase ML Kit's object detection to recognize food content in images. If non-food content is detected, the post is reported and managed accordingly.

9. Recipe Web Scraping and Integration: Fetches recipes from external websites and integrates them into the app's database, enhancing the variety of available recipes.

10. Multi-Threaded Server and Load Balancing: Manages application logic and functionality through a multi-threaded server architecture and load balancing using NGINX. This ensures efficient request handling, scalability, and high availability.

3.3 Design Rationale

The MVVM architectural pattern was chosen for its separation of concerns, promoting maintainability. The modular decomposition approach aligns with software design best practices, ensuring clear functionality and focused responsibility for each module.

DATA DESIGN

4.1 Data Description

The Smarter Foodies app will utilize either Firebase or SQLite as the database to store essential data related to users, posts, and more. The data components will encompass:

1. User Information: A table or collection storing user relationships, including details like usernames, emails, and online/offline status.

2. User Account Information: A table or collection containing account-related data, such as follower and following counts, cart contents, user type (business/regular), profile photo, and website link.

3. Post Information: A table or collection storing details of recipe posts, including captions, ingredients, cooking instructions, images, publication dates, comments, and other relevant data.

4. User Feed: A table or collection storing references to posts from followed users. This allows users to see their own customized feed based on followed users' posts.

5. User Liked Posts: A table or collection storing references to posts that a user has liked, facilitating easy access to their favorite content.

6. Reported Posts: A table or collection storing references to posts reported by AI models. This supports moderation and content quality control.

7. Utils: A table or collection for storing frequently used ingredient options, enhancing user convenience.

4.2 Data Dictionary

User Information:

User ID (Primary Key)

Username

Email

Online Status

Last Seen

Profile Photo URL

User Account Information:

User ID (Primary Key)

Follower Count

Following Count

Cart Contents

User Type (Business/Regular)

Profile Photo URL

Website URL

Post Information:

Post ID (Primary Key)

User ID (Foreign Key)

Caption

Ingredients

Cooking Instructions

Images URLs

Publication Date

Comment IDs (References to Comments)

User Feed:

User ID (Primary Key)

Post IDs (References to Posts)

User Liked Posts:

User ID (Primary Key)

Post IDs (References to Liked Posts)

Reported Posts:

Report ID (Primary Key)

Post ID (Foreign Key)

Reporting User ID (Foreign Key)

Report Date

Report Type (Food Detection/Spam Detection)

Utils:

Ingredient ID (Primary Key)

Ingredient Name

This data structure ensures organized storage and retrieval of app-related information, promoting efficient app functionality.

COMPONENT DESIGN

5.1 User Authentication and Registration

Component: Login and Registration

Responsibilities: this component manages the user registration and authentication processes.

Subcomponents:

Login Activity

Layout Login Activity

Register Activity

Layout Register Activity

Functionalities:

Users can register using their university email addresses.

Registered users can securely log in using their credentials.

5.2 Main Feed and Navigation

Component: Home

Responsibilities: This component manages the main feed display, posts, and navigation.

Subcomponents:

Home Activity

Layout Home Activity

Post Adapter

Navigation Menu

Functionalities:

The dynamic main feed displays food-related posts from followed users.

Users can interact with posts, like and view post details.

Navigation facilitates movement between the Home and Profile sections.

5.3 User Profile and Settings

Component: Profile

Responsibilities: This component manages user profiles, settings, and customization.

Subcomponents:

Profile Activity

Layout Profile Activity

Edit Profile

Layout Edit Profile

Functionalities:

Each user has a personalized profile with user-provided details.

Users can update profile information, profile pictures, and interests.

Users can follow/unfollow other users within the app.

5.4 Sharing with Friends, Post Creation, and Editing

Component: Share

Responsibilities: This component handles post uploading and sharing with followers.

Subcomponents:

Share Activity

Image Adapter

Create Post Activity

Layout Share Activity

Layout Create Post Activity

Functionalities:

Users can create new posts containing food-related content.

Users can edit or delete their own posts.

5.5 AI Models for Food & Spam Detection

Component: MLKit

Responsibilities: This component detects recognizable food content and spam ***text.***

Subcomponents:

Object Detection

Spam Text Detection

Functionalities:

AI models analyze content for food elements.

AI models detect basic spam text.

5.6 Notify User by Text

Component: Notifications

Responsibilities: This component simulates notifications for user interactions.

Subcomponents:

Reminder

Reminder Broadcast

Functionalities:

Users can trigger simulated notifications to understand their functionality.

5.7 Search Functionality

Component: Search

Responsibilities: This component enables users to search for user profiles.

Subcomponents:

Search Activity

Layout Search Activity

Functionalities:

Auto-complete users' search bar assists users in finding profiles.

Users can search for specific user profiles.

5.8 Weekly Meal Planning and Grocery List Generation

Component: Meal Planning

Responsibilities: This component helps users plan meals and generate shopping lists.

Subcomponents:

Meal Planning Activity

Grocery List Generator

Online Grocery Store Referral

Functionalities:

Users can plan weekly meals and generate ingredient lists.

The app refers users to online grocery stores for optimized resource usage.

5.9 Chef-Contributed Recipes

Component: Recipes

Responsibilities: This component allows chefs to upload public recipes.

Subcomponents:

Recipe Upload

Recipe Database

Functionalities:

Certified chefs can upload recipes visible to all users.

Regular users can add private recipes for approved friends and followers.

5.10 Business Account and Payment

Component: Business Account and Payment

Responsibilities: This component facilitates the establishment of business accounts and handles payment transactions within the app, enabling users to access exclusive content from business users.

Subcomponents:

Payment Activity

Layout Payment Activity

Business Profile Activity

Layout Business Profile Activity

Functionalities:

Users can subscribe to business accounts to access premium content.

Payment processing ensures secure and efficient transactions for users subscribing to business accounts.

Business profiles showcase exclusive content and offer a personalized experience for subscribers.

5.11 <u>Chat and Messaging (WhatsApp-like)</u>

Component: Chat and Messaging

Responsibilities: Manage real-time chat and messaging interactions, allowing users to communicate seamlessly within the app.

Subcomponents:

Chat Activity

Chat Profile Activity

Chat Find Friends Activity

Messages Adapter

Request to Chat Activity

Functionalities:

Users can send and receive chat messages to and from other users.

Real-time chat functionality enables instant messaging between users.

Users can send emojis, images, files, and more within chat conversations.

Notifications inform users about new messages even when the app is in the background.

5.12 <u>Recipe Web Scraping and Integration</u>

Component: Recipe Web Scraping and Integration

Responsibilities: Fetch and seamlessly integrate external recipes into the app's database, enhancing the variety of available content.

Subcomponents:

Scraping File

Functionalities:

Implement a web scraping mechanism to extract relevant data from recipe websites.

Parse and normalize scraped data for integration into the app's data model.

Store scraped recipe data in the app's database for display and access.

5.13 Multi-Threaded Server and Load Balancing

Component: Multi-Threaded Server and Load Balancing

Responsibilities: Manage a multi-threaded server architecture with load balancing to ensure efficient request handling and scalability.

Subcomponents:

Server

Workers Handler

Worker

Entry Controller

Functionalities:

Utilize multiple threads to handle incoming requests concurrently, improving response times.

Implement mechanisms for thread management and workload distribution.

Employ NGINX as a load balancer to evenly distribute traffic across server instances.

Scale server instances based on increasing user traffic to ensure optimal performance.

HUMAN INTERFACE DESIGN

6.1 Overview of User Interface

The user interface (UI) of the application is designed to be user-friendly and intuitive. The UI encompasses several screens that allow for seamless navigation and interaction. The key screens include:

Feed Screen: This screen dynamically displays recipe posts from followed users, offering a central hub for engaging with culinary content.

Profile Screen: Users can access their personalized profile, showcasing their uploaded recipe posts, friends count, profile picture, and additional details.

Search Screen: The search functionality empowers users to find specific user profiles conveniently, enhancing the social experience.

Post Upload Screen: This screen facilitates the easy creation and uploading of recipe posts, allowing users to share their culinary creations.

Chat and Messaging Screens: Users can initiate chat conversations with friends and send messages, emojis, images, and more.

Payment Screen: For business users' exclusive content, this screen handles payments to access premium features.

Notification Screen: Simulated notifications inform users about interactions and engagements related to their posts.

6.2 Screen Images

The application's user interface comprises visually appealing and cohesive designs. Key screens include:

Profile Feed: Displays the user's recipe uploads, friends count, profile picture, and more.

Upload and Update Post: Provides a user-friendly way to add and edit recipe posts.

Recipe Post in the Main Feed: Showcases a recipe post within the main feed for users to engage with.

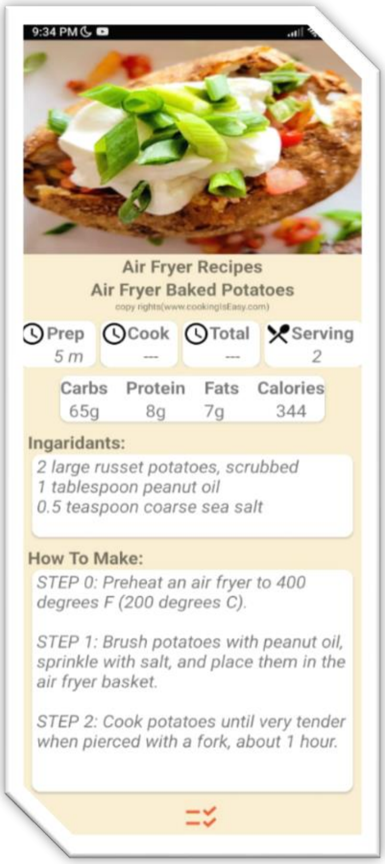
Main Feed: Features the dynamic main feed, displaying followed users' recipe posts.

Recipe Post in Full Detail: Offers a detailed view of a recipe post, including ingredients and instructions.

Login Page: The entry point of the application, allowing users to log in.

Please note that the design is still under consideration, and the final UI may vary.

For reference:


9:34 PM

Air Fryer Recipes
Air Fryer Baked Potatoes
copy rights(www.cookingIsEasy.com)

Prep	Cook	Total	Serving
5 m	---	---	2

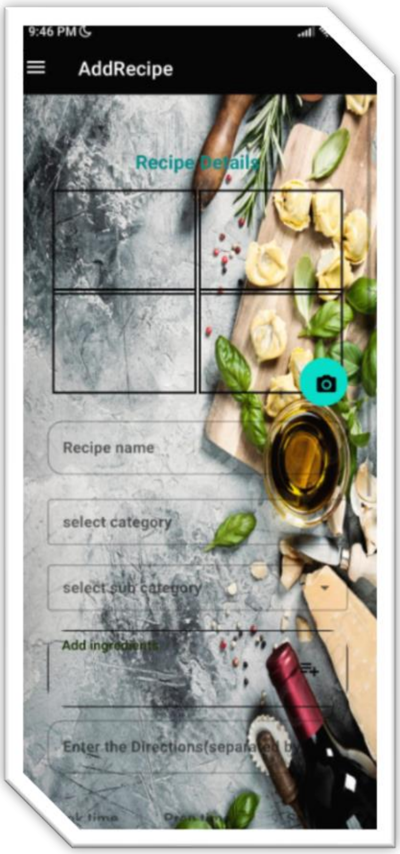
Carbs	Protein	Fats	Calories
65g	8g	7g	344

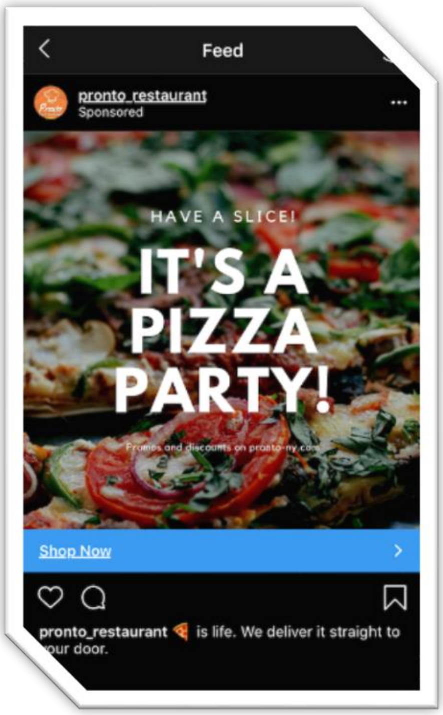
Ingaridants:
2 large russet potatoes, scrubbed
1 tablespoon peanut oil
0.5 teaspoon coarse sea salt

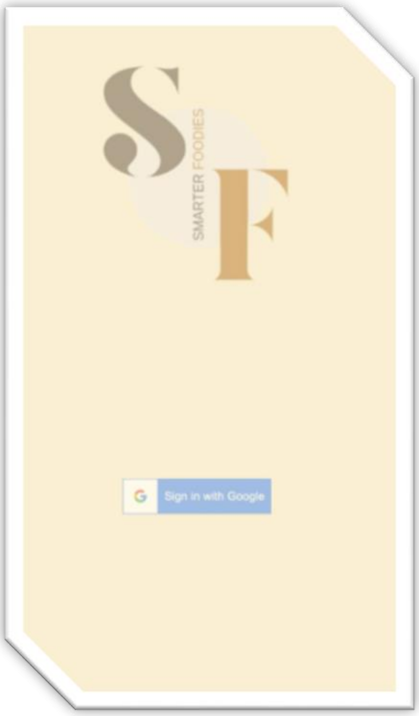
How To Make:
STEP 0: Preheat an air fryer to 400 degrees F (200 degrees C).

STEP 1: Brush potatoes with peanut oil, sprinkle with salt, and place them in the air fryer basket.

STEP 2: Cook potatoes until very tender when pierced with a fork, about 1 hour.




Feed
pronto_restaurant
Sponsored
HAVE A SLICE!
IT'S A
PIZZA
PARTY!
Promos and discounts on pronto-ny.com
Shop Now
pronto_restaurant 🍕 is life. We deliver it straight to your door.



6.3 Screen Objects and Actions

Login Page: Users can navigate to registration or log in using their credentials.

Main Feed: Interactions include liking posts, viewing post details, and seamless navigation between sections.

Full Detail Recipe Post: Users can view recipe details, add posts to likes, and export ingredient lists via WhatsApp.

Post Upload Page: Users can add ingredients and fill in recipe details, enhancing the content creation process.

Profile Page: Displays profile information, uploaded posts, and allows users to edit their profile.

Search Page: Auto-complete user search aids in finding friends' profiles easily.

Payment Page: Enables users to make payments to access premium content.

Chat and Messaging Pages: Users can initiate chat requests, send messages, emojis, images, and manage conversations.

REQUIREMENTS MATRIX

7.1 Functional Requirements

User Authentication and Registration:

Users must complete a registration process using their university email addresses.

Registered users can securely log in using their credentials.

Main Feed and Navigation:

A dynamic main feed displays sample food-related posts for users.

Users can interact with posts, including "liking" posts and viewing post details.

The navigation system facilitates seamless movement between the Home and Profile sections.

User Profile and Settings:

Each user has a personalized profile containing user-provided details.

Users can update profile information, including profile pictures and interests.

Users have the option to follow or unfollow other users within the app.

Sharing with Friends, Post Creation, and Editing:

Users can compose new posts containing food-related content.

Users possess the ability to edit or delete their posts as needed.

AI Models for Food & Spam Detection:

The application employs basic AI models to analyze content for food-related elements.

AI models will also detect basic spam text to enhance user experience.

Notify User by Text:

The app simulates notifications to demonstrate the notification feature.

Users can trigger simulated notifications to understand their functionality.

Search Functionality:

Users can initiate searches to find specific user profiles within the app.

A basic auto-complete feature assists users by suggesting profile names as they type.

Weekly Meal Planning and Grocery List Generation:

Users have the option to plan their weekly meals using the app.

The app generates precise ingredient lists for planned meals.

The app refers users to an online grocery store website to optimize resource usage.

Chef-Contributed Recipes:

Certified chefs are allowed to upload recipes publicly, visible to all app users.

Regular users can add private recipes, only viewable by their approved friends and followers.

7.2 Non-Functional Requirements:

Performance Requirements:

The application should provide prompt responses within 3 seconds for most interactions.

The app's performance should remain smooth and responsive even when managing up to fifty concurrent users.

Usability:

The user interface (UI) design should be intuitive, catering to students as primary users.

The UI must be adaptable to different screen sizes to ensure a seamless user experience.

APPENDICES

https://firebase.google.com/

https://developer.android.com/studio/