Versions:

|  |  |  |  |
| --- | --- | --- | --- |
| Date | Version | Autor | Comments |
| 04.11.2024 | 1 | Borzan Călina-Annemary  Gozman-Pop Maria-Eliza |  |

Analysis Design for <YumCycle >

Contents:

1. **General Presentation**
2. **Theoretical Fundamentals**
3. **IT Technology (**which will be implemented**)**
4. **Functionalities**
5. **Actors and related access rights**
6. **Use Case Diagrams**
7. **System Architecture**
8. **Design (UML Diagrams)**

1. **Operation Mode (Operation Guide, including screen-shots)**
2. **Portability**
3. **General Presentation**

YumCycle is a mobile application designed to help users efficiently manage the contents of their fridge and pantry, reduce food waste, and promote healthier eating habits by suggesting recipes based on ingredients users already have. Developed by a team inspired by personal experiences with food waste, YumCycle combines inventory management with recipe discovery, simplifying food tracking and planning while reducing unnecessary purchases. The primary features of YumCycle include a barcode-scanning feature that allows users to quickly add items to their virtual fridge by scanning product barcodes, and a manual entry option for items without barcodes. Users can also receive personalized recipe suggestions based on their current inventory. Additionally, YumCycle has a recipe-sharing component where users can submit their own recipes, which are then validated by an administrator before being added to a shared recipe database. With these tools, YumCycle empowers users to make the most of their food inventory, save money, and reduce environmental impact.

1. **Theoretical Fundamentals**

The development of YumCycle incorporates several fundamental principles in software design and user experience:

* **User-Centric Design**: YumCycle’s features are centred around the user’s needs, enabling them to manage inventory with minimal input and maximize the utility of their existing food items. Features like recipe suggestions create a seamless experience that directly benefits users’ daily lives.
* **Inventory Management**: The concept of digital inventory management is central to YumCycle, allowing users to efficiently track food items. This management feature is designed to be user-friendly and leverages a database backend to store details about each item, including name and quantity with barcode scanning to make inventory updates easier and faster.
* **Recipe Discovery and Sharing**: The app’s recipe component utilizes an algorithm that matches users’ ingredients with available recipes, supporting customized and relevant recipe suggestions. This feature encourages engagement by allowing users to submit and share their favourite recipes, contributing to a broader user-driven database of recipe options.
* **Moderation and Feedback Mechanisms**: An essential part of the YumCycle ecosystem is moderation. The administrator role involves validating recipes to ensure quality and prevent inappropriate content. This system promotes reliability in the recipe database while offering feedback and improvement opportunities through user ratings and reviews.

1. **IT Technology**

Given that YumCycle is built using Java, Spring Boot, and a MySQL database, the following components of the application architecture can be outlined:

* **Backend Development with Java and Spring Boot**:
* Java: A robust and scalable programming language, Java is widely used for building mobile and backend applications due to its security, cross-platform capabilities, and compatibility with Android, which is the target platform for YumCycle.
* Spring Boot: This Java-based framework simplifies application development by providing out-of-the-box configurations for commonly used features like RESTful APIs, security, and database management. For YumCycle, Spring Boot is ideal for handling backend operations such as user authentication, recipe validation, and inventory management, allowing for a modular, scalable application structure.
* RESTful API Development: Spring Boot’s REST API support facilitates seamless data exchange between the frontend and backend. For example, when a user scans a barcode or submits a recipe, these interactions are handled through RESTful API calls, making the backend highly responsive and efficient.
* **Database Management with MySQL**:
* MySQL Database: The relational database stores critical information such as user accounts, inventory items, recipes, and reviews. MySQL is known for its reliability, scalability, and ease of integration with Java-based applications, making it a suitable choice for managing YumCycle’s data-intensive operations.
* Entity-Relationship Modeling: The data model for YumCycle would likely consist of several tables representing entities like User, InventoryItem, Recipe, Feedback, and Favorites, among others. Relationships between these tables allow for complex querying, such as finding recipes based on ingredients, tracking user preferences, and managing administrative actions.
* Data Access Layer (DAO): In Spring Boot, the Data Access Object (DAO) layer manages database interactions. Through the Spring Data JPA module, developers can use Java classes to interact with database tables, making data management straightforward and minimizing the need for manual SQL queries.
* **Frontend and User Interface**:
* Mobile-First Design: YumCycle’s interface is optimized for mobile devices, focusing on intuitive navigation, large touch targets, and streamlined flows for tasks like adding items, viewing recipes, and managing inventory.
* Barcode Scanning Integration: Barcode scanning in the app can be implemented using libraries compatible with Android, such as ZXing or ML Kit, which work with the device’s camera to identify and translate barcode data into actionable information.

1. **Functionalities**

The YumCycle application includes a variety of functionalities to help users manage their fridge and pantry contents, reduce food waste, and find relevant recipes. Here’s a breakdown of its main functionalities:

* **User Registration and Authentication**:
  + Sign Up / Registration: New users can create accounts by entering personal details (e.g., name, email, password).
  + Log In / Log Out: Existing users can securely log in to access their saved inventory and recipes, clearing session data to prevent unauthorized access.
* **Inventory Management**:
  + Barcode Scanning: Users can add items to their inventory by scanning product barcodes, which automatically retrieves details like name from a connected database.
  + Manual Entry: For items without barcodes, users can manually enter details, including product name, and quantity.
  + Viewing and Organizing Inventory: Users can view a list of items in their inventory, sorted by categories like product type, or custom tags.
* **Recipe Management**:
  + Recipe Discovery: Users can search for recipes based on ingredients in their inventory, dietary preferences, or keywords. The app suggests recipes that use ingredients the user already has to minimize additional shopping.
  + Recipe Submission: Users can submit their own recipes to the app’s recipe database, which are reviewed by the administrator for approval before becoming available to the community.
  + Favourites: Users can save favourite recipes for easy access in the future, as well as manage their preferred recipes list.
* **Feedback and Ratings**:
  + Rating and Reviewing Recipes: After trying a recipe, users can rate and review it to help other users determine recipe quality and reliability. Ratings and reviews contribute to the recipe’s overall ranking in the app.
  + User Feedback for App Improvement: Users can provide feedback on their experience with the app itself, which administrators can use to enhance functionality and user experience.
* **Administrator Functionalities**:
  + Recipe Validation: The administrator reviews user-submitted recipes to ensure they meet the app’s quality and content guidelines. Approved recipes are published in the database, while unapproved recipes are returned with feedback.
  + User Management: Administrators can delete user accounts if necessary due to misuse, spam, or violations of app guidelines.
  + App Improvement Management: The administrator reviews user feedback and app analytics to plan and implement improvements that enhance the overall user experience.

1. **Actors and related access rights**

* **Administrator**

Access Rights:

* + Full Access: Administrators have complete access to all app features, including those restricted from general users, like user and recipe management.
  + Recipe Moderation: Administrators can review, approve, or reject recipes submitted by users. They can provide feedback on rejected recipes for potential improvement.
  + User Management: Administrators have access to delete user accounts if necessary. This includes full control over the user's data and inventory management privileges.
  + App Improvement: Administrators can access and analyse user feedback and app usage data to make decisions about updates and improvements.
* **User**

Access Rights:

* + Inventory Management: Users have full access to manage their personal inventory, including adding, viewing, and deleting items via barcode scanning or manual entry.
  + Recipe Management: Users can search for, view, rate, and review recipes in the database. They can also submit their own recipes, though these submissions require administrator approval before publication.
  + Personal Preferences: Users can customize settings such as notification preferences for expiring items, dietary restrictions, and recipe discovery preferences.
  + Feedback and Review: Users can rate and review recipes they have tried, contribute feedback about the app, and manage their list of favourite recipes and saved items.

These access rights are designed to ensure that users can personalize and engage with the app while the administrator maintains oversight and quality control.

1. **Use Case Diagrams**

A diagram of a person's diagram

Description automatically generated

A diagram of a user

Description automatically generated

1. **System Architecture**

The YumCycle application follows a multi-layered architecture, typically structured as a three-tier architecture to separate concerns and ensure modularity, scalability, and maintainability. The system architecture is organized into the following layers:

* Presentation Layer (Frontend)
* Application Layer (Backend)
* Data Layer (Database)

**7.1. Presentation Layer (Frontend)**

The presentation layer serves as the user interface for the YumCycle mobile application. It is responsible for displaying the application’s functionality, receiving user inputs, and sending requests to the backend. It also handles the barcode scanning feature and provides a responsive and intuitive user experience.

Components:

* Mobile Application Interface: The app is designed primarily for mobile devices, with a user-friendly interface that enables users to easily navigate features like inventory management, recipe discovery.
* Barcode Scanning: The app uses a barcode scanning library (e.g., ZXing or Google ML Kit for Android) to enable users to scan products. This component interacts with the backend to retrieve product details based on barcode data.

Technologies:

* Android for mobile app development (Java/Kotlin).
* Barcode Scanning SDK such as ZXing or ML Kit for barcode scanning capabilities.

**7.2. Application Layer (Backend)**

The application layer is the core of the YumCycle system, developed using Java and Spring Boot to manage the application’s business logic, RESTful APIs, and data processing. This layer acts as a bridge between the frontend and the database, handling data retrieval, validation, and processing before returning results to the frontend.

Components:

* RESTful API: Spring Boot serves as a RESTful service layer, allowing the frontend to communicate with the backend through HTTP requests. This API manages tasks like user authentication, inventory updates, and recipe searches.
* User Authentication and Authorization: Handles secure logins, session management, and access control for both user and administrator roles.
* Inventory Management Services: Manages the addition and retrieval of items in the user's inventory.
* Recipe Management and Approval Workflow: Includes services for users to submit recipes and for administrators to review, approve, or reject these submissions.
* Data Validation and Error Handling: Ensures that all inputs (e.g., barcode scans, manual entries, recipe submissions) are validated to prevent errors or inconsistencies in the application.

Technologies:

* Java for the backend logic.
* Spring Boot for building the REST API and managing dependency injection, data processing, and business logic.
* Spring Security for authentication and access control.
* JSON for data format in API requests and responses.

1. **Data Layer (Database)**

The data layer is responsible for persistent storage, housing user data, inventory records, recipes, and user feedback. MySQL is used as the primary relational database, chosen for its reliability, ease of integration with Java-based applications, and efficient handling of structured data.

Components:

* User Data Management: Stores information such as user profiles, login credentials, preferences, and inventory settings.
* Inventory Records: Tracks items added by each user, including product names, quantities, and associated barcodes.
* Recipe Database: Houses all recipes, including user-submitted and approved recipes, along with ratings, reviews, and any categorization tags.
* Feedback and Reviews: Manages user reviews, recipe ratings, and general app feedback, enabling administrators to monitor user satisfaction.
* Relationships and Constraints: Enforces referential integrity and relationship mappings to maintain data consistency, such as linking inventory items to users and recipes to reviews.

Technologies:

* MySQL for relational data storage and management.
* Spring Data JPA as an ORM (Object-Relational Mapping) layer, allowing Java classes to interact directly with database tables.
* ER Diagrams to model relationships between tables, such as users, inventory items, recipes, and reviews.

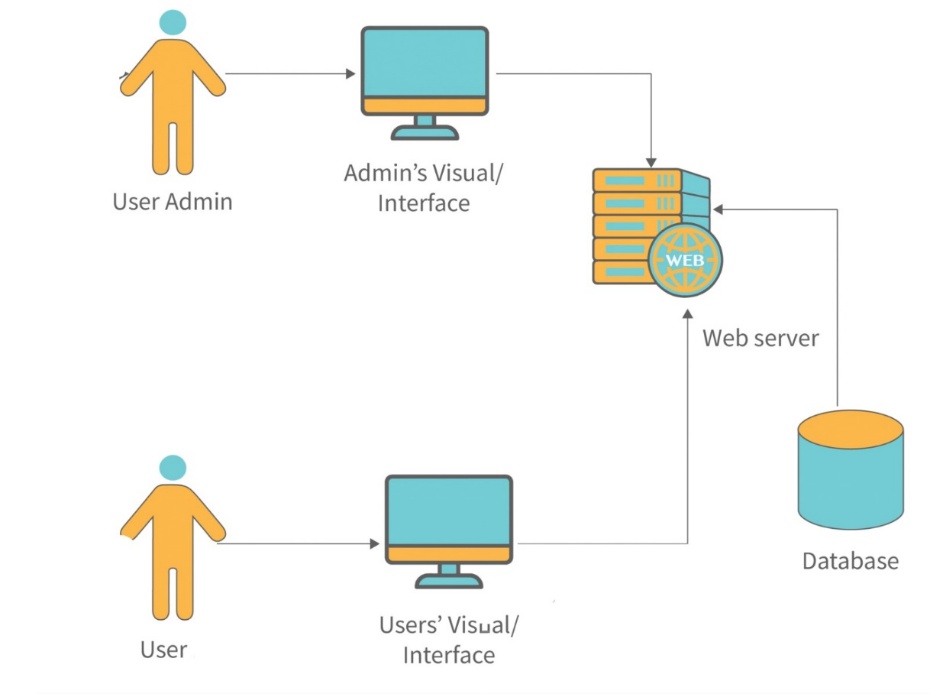
**Inter-Layer Communication and Workflow**

* User Action Initiation: A user initiates an action (e.g., scanning a barcode, adding an item) on the frontend.
* API Call to Backend: The frontend sends a REST API request to the backend to perform the desired operation. For example, if the user scans a barcode, the backend queries the barcode database for product information.
* Data Processing in Backend: The backend processes the request, interacts with the MySQL database if necessary, and applies business logic (e.g., validating recipes).
* Response to Frontend: The backend sends a response to the frontend with the results, such as the item details retrieved or a confirmation message.

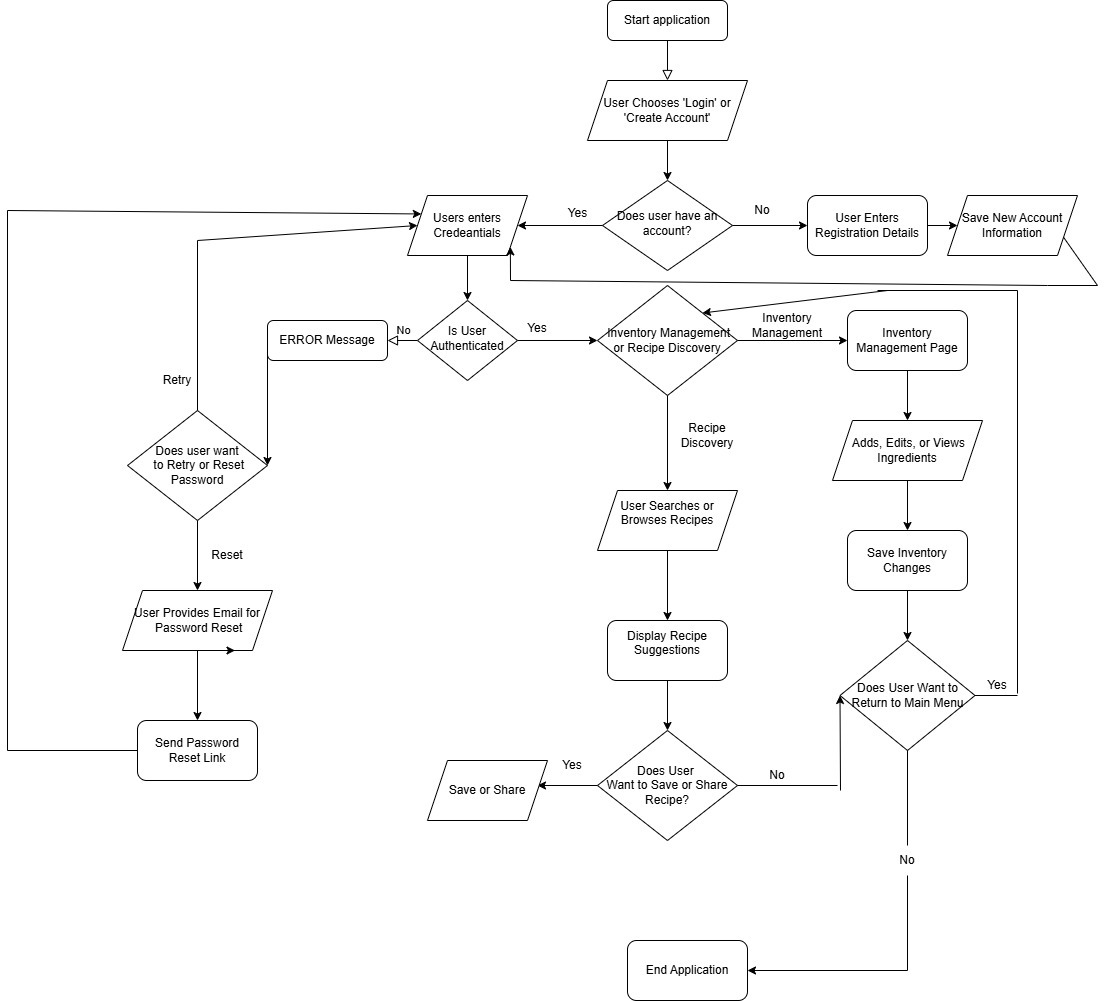
**Additional Considerations**

* Scalability: The architecture can be scaled by deploying the application on cloud platforms like AWS or Google Cloud, using load balancers and microservices to handle high user loads.
* Security: Spring Security and JWT (JSON Web Tokens) for secure user authentication, role-based access control, and data encryption ensure that user data is protected.
* Future Expansion: The architecture is designed to allow the integration of additional services, such as machine learning for personalized recipe recommendations or cloud storage for images in recipe submissions.

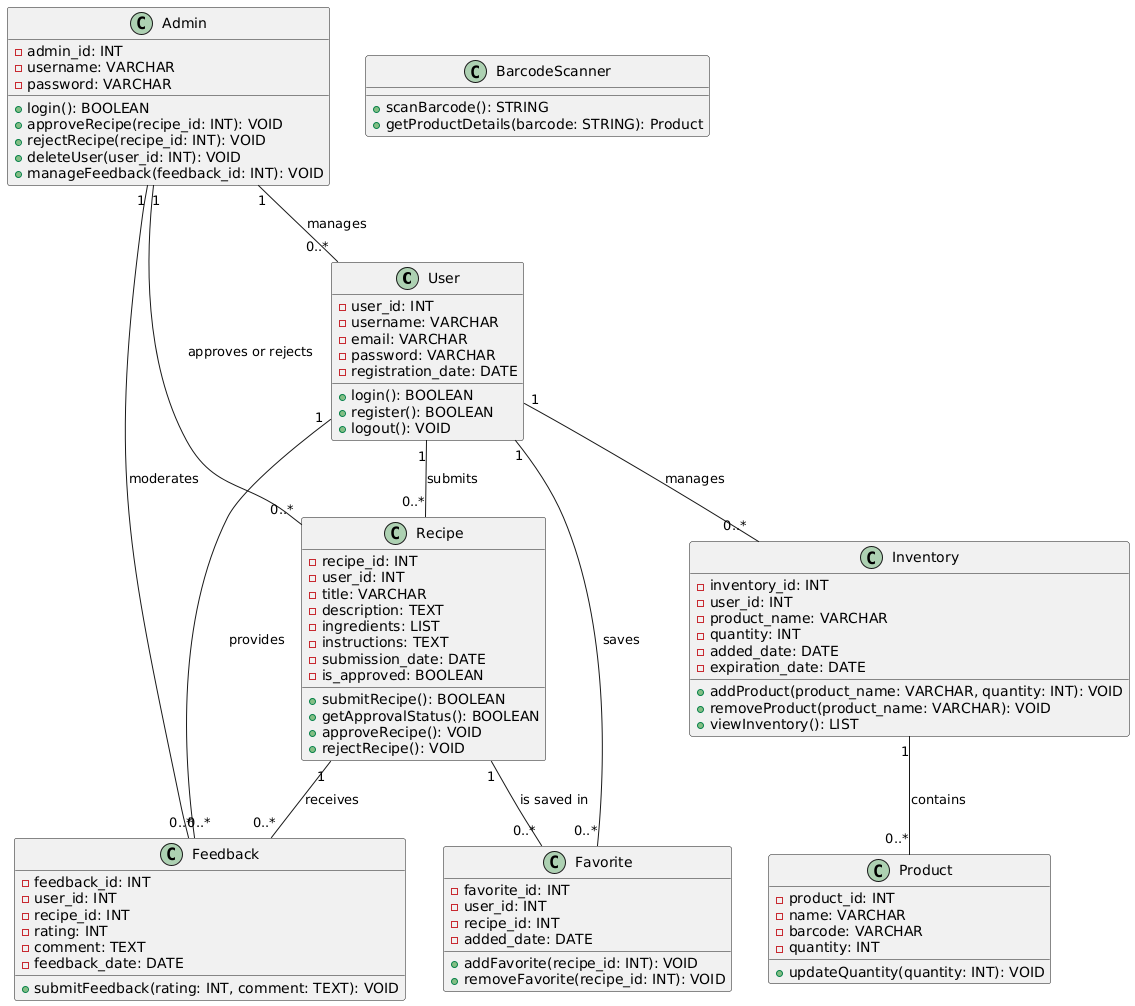
1. **Design (approx. 12 UML Diagrams!!!)**
   1. **General Architecture Diagram**



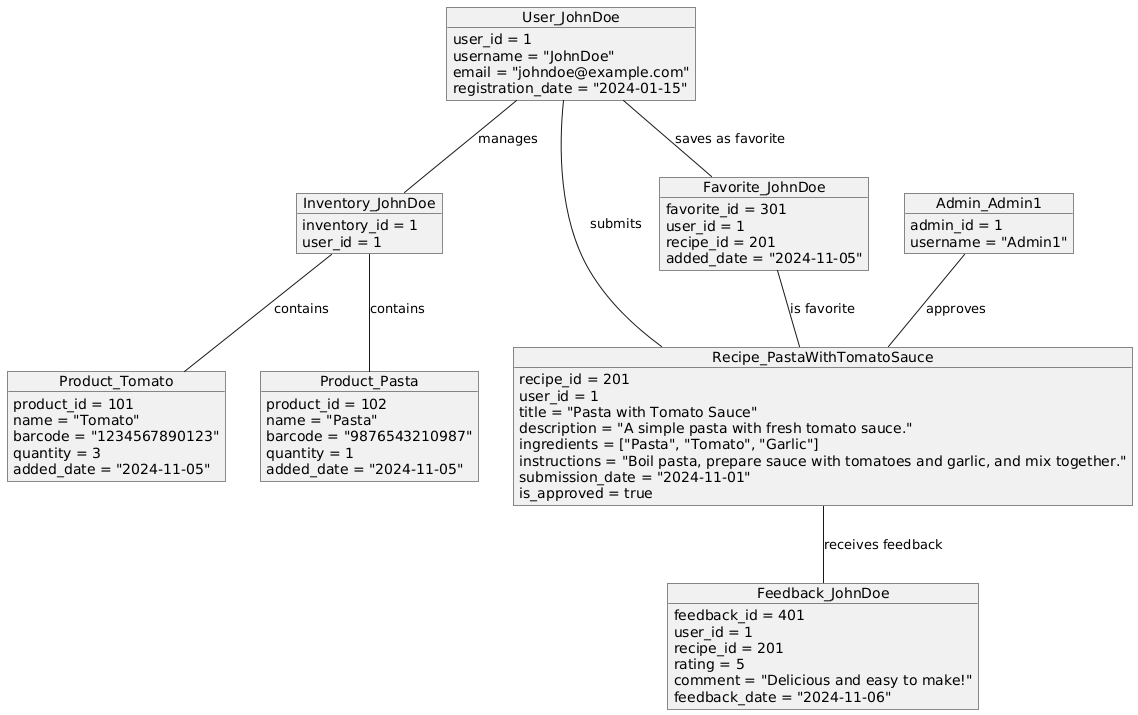
**8.2 Flowchart**

****

**8.3 Class Diagram**



**8.4 Object Diagram**



**8.5 Use Cases**

A diagram of a person's diagram

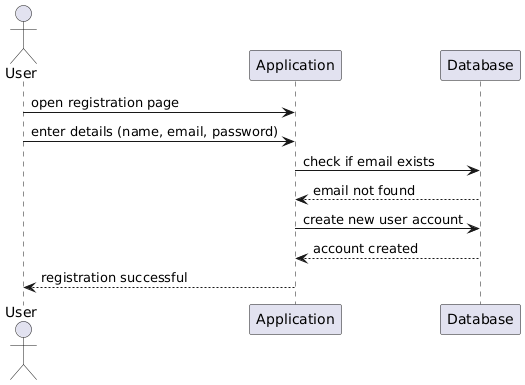
Description automatically generated A diagram of a user

Description automatically generated

**8.6 Sequence Diagram**

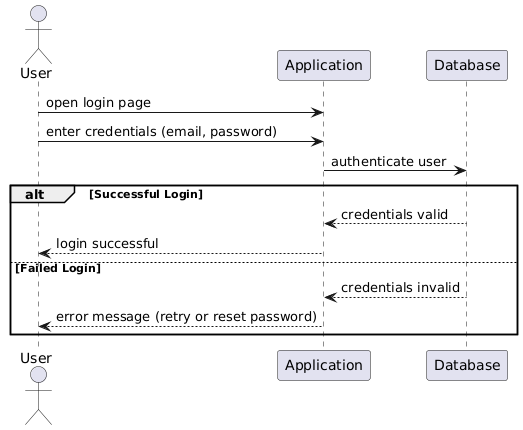
**8.6.1. User Registration Sequence Diagram**

This diagram shows the flow when a new user registers an account.



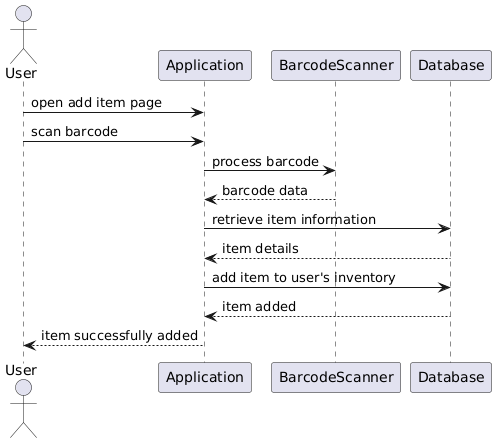
**8.6.2 User Login Sequence Diagram**

This diagram demonstrates the process of a user logging into the app.



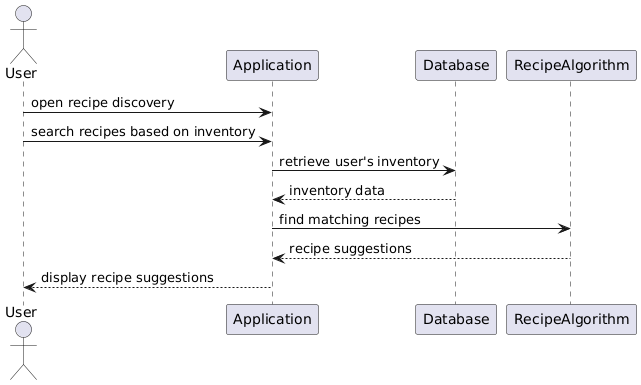
**8.6.3 Inventory Management - Adding an Item with Barcode**

This sequence shows a user adding an item to their inventory by scanning a barcode.



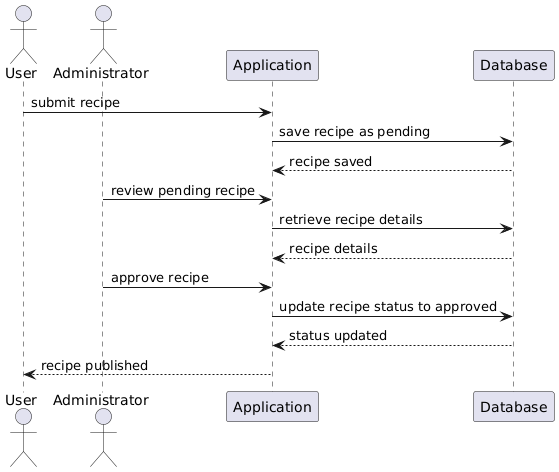
**8.6.4 Recipe Discovery Sequence Diagram**

This diagram shows the interaction flow for a user searching for recipes based on their current inventory.



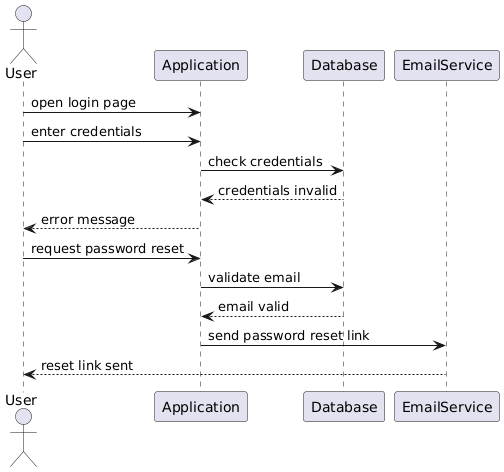
8.6.5 **Recipe Submission and Validation Sequence Diagram**

This sequence demonstrates a user submitting a recipe, which is then reviewed and approved by an administrator.



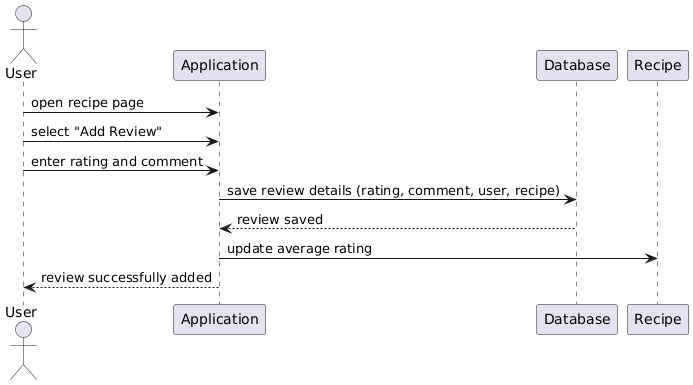
8.6.6 **Password Reset Sequence Diagram**

This diagram shows the process of resetting a password after a failed login attempt.



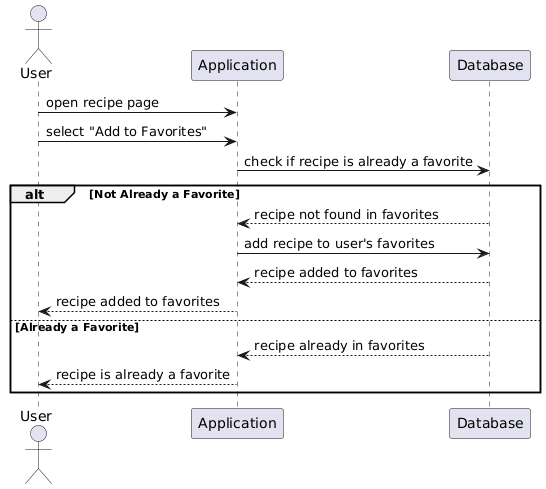
**8.6.7 Adding a Review Sequence Diagram**

This sequence diagram shows the process of a user adding a review for a recipe they have tried.



**8.6.8. Adding a Recipe as Favorite Sequence Diagram**

This sequence diagram illustrates a user marking a recipe as a favorite for easy access later.



**8.7 Activity Diagram**

A screenshot of a computer

Description automatically generated

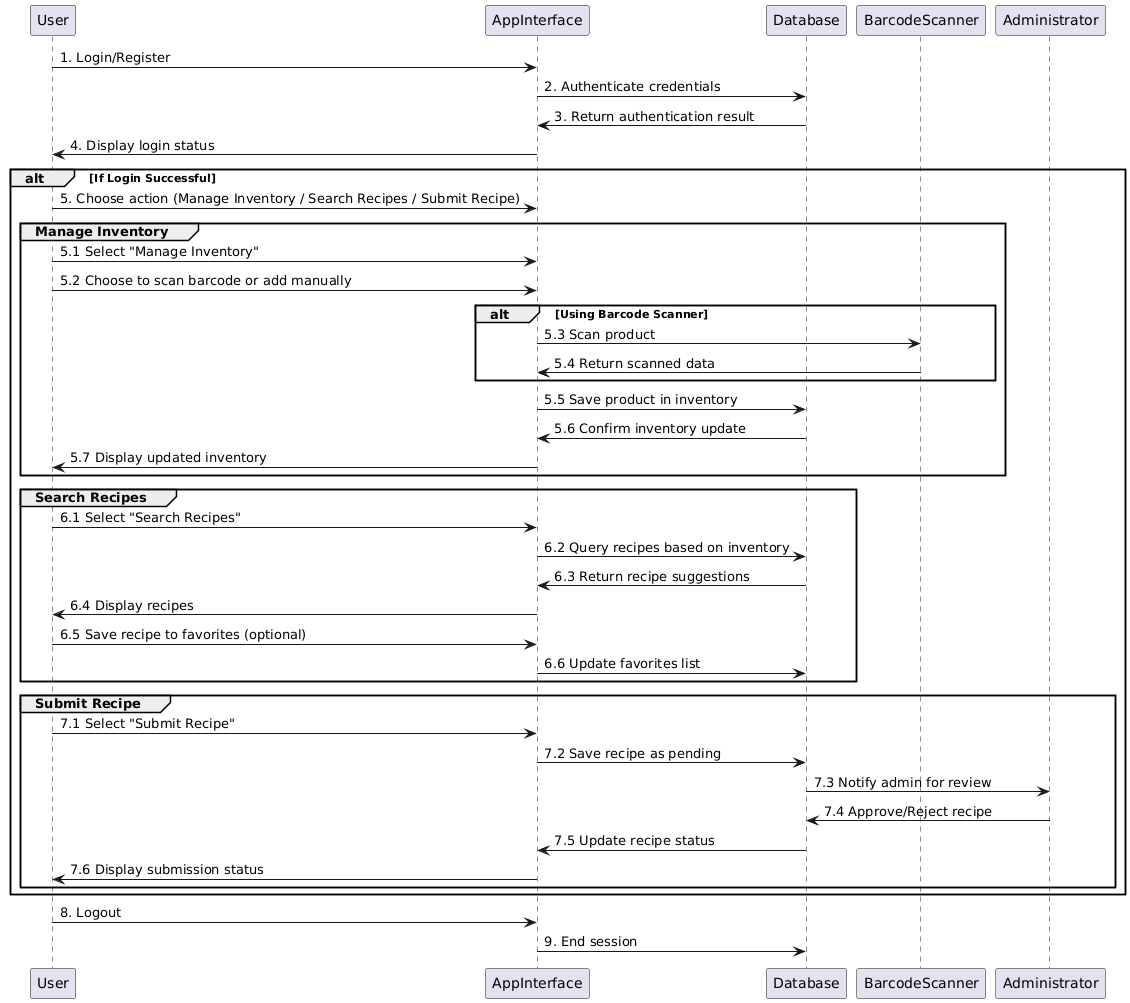
**8.8 State Transition Diagram**

In the case of my application, this type of diagram helps visualize the user's flow through various parts of the app and how their actions affect their state within the system.

A diagram of a diagram

Description automatically generated

**8.9 Communication diagram**



**8.10 Package Diagram**

This package diagram organizes the different components of the YumCycle application into logical groups, making it easier to understand the overall structure. This is useful, as it helps break down the system into manageable sections, showing dependencies between them, including packages for major parts of the system, such as User Management, Inventory Management, Recipe Management, and Administration.

A screenshot of a computer

Description automatically generated

**8.11 Deployment diagram**

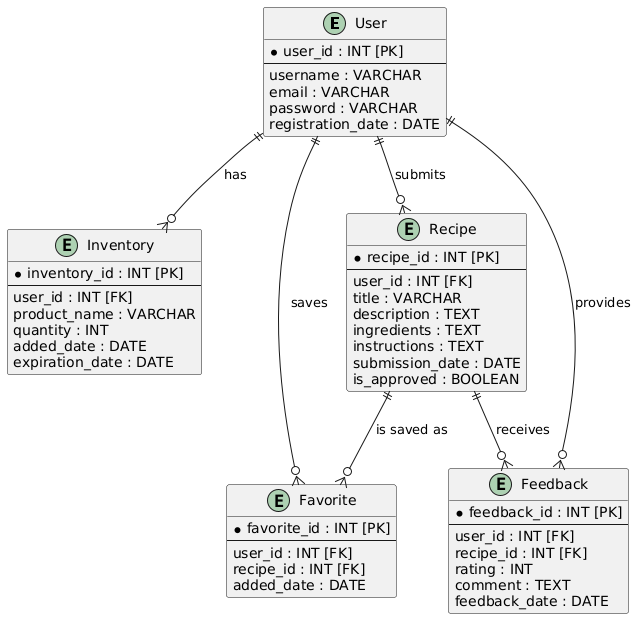
A deployment diagram represents the physical architecture of the system, showing how software components are deployed on hardware nodes and how different parts of the system communicate in the actual environment. For the *YumCycle* app, a deployment diagram would show how the mobile application, backend server, database, and other components are deployed in a real-world setup.

A close-up of a diagram

Description automatically generated

**8.12 Data Base Diagram**

This is the database schema diagram for the YumCycle app, which includes tables for Users, Inventory, Recipes, Favorites, and Feedback. This schema reflects the main entities and their relationships in the app's data model.



1. **Operation Mode (Interactivity presentation) + screen shots (UI design)**
2. **Portability**

The YumCycle application is designed to be highly portable, allowing for broad compatibility and easy deployment across different environments.

1. Platform Compatibility:

* Android Operating System: The YumCycle app is developed for Android devices, using Kotlin or Java as the programming language. This ensures compatibility with a wide range of Android devices, from smartphones to tablets, provided they meet the minimum Android version requirements.
* Backend Server Compatibility: The backend, built with Spring Boot, is platform-agnostic and can run on any server environment that supports Java, making it compatible with various operating systems, such as Windows, Linux, and macOS.
* Database Portability: MySQL, the application’s database, is widely compatible and can be hosted on different database servers or cloud services (e.g., Amazon RDS, Google Cloud SQL), making the app database easily portable across different environments.

1. Cloud Deployment:

* Cloud-Ready Architecture: The backend and database can be hosted on cloud platforms like AWS, Google Cloud, or Microsoft Azure, allowing for scalability and redundancy. Cloud deployment enables YumCycle to serve a larger user base and maintain performance under varying loads.
* Containerization with Docker: The backend and database can be containerized using Docker, enhancing portability by enabling easy deployment across any Docker-supported platform, including cloud services and local environments.

1. Device Portability:

* Data Synchronization Across Devices: YumCycle’s data synchronization allows users to log in from multiple Android devices and access the same inventory and recipe data. User data is stored centrally, enabling portability across different devices with seamless access to stored information.
* API-First Approach: The app’s RESTful API design allows easy integration with other platforms or frontend applications, should YumCycle expand to other operating systems (e.g., iOS or web apps) in the future.

1. Future Expansion to Other Platforms:

* Cross-Platform Framework: If the app is to be expanded to other mobile operating systems (such as iOS), frameworks like Flutter or React Native could be used to rebuild the application for both Android and iOS.
* Web Version: Due to its API-driven backend, YumCycle could easily offer a web-based application, extending accessibility and making it available across different device types and operating systems.

The YumCycle app’s portability and adaptable design ensure it can be deployed, expanded, and maintained across various platforms and environments as user needs and technology evolve.

1. **Competing software**

* **Kitchen Pal: Food Pantry App**

“KitchenPal (iCuisto) is the only app that works as a pantry manager, grocery list, product comparison, meal planner, family organizer, and recipe ideas app - all rolled into one. The app learns and provides suggestions the more you use it, like a smart assistant.”

**A screenshot of a food pantry app

Description automatically generated**

While KitchenPal offers a broad range of features, YumCycle stands out by providing a more streamlined, user-friendly experience focused on essential functionalities without requiring a subscription. YumCycle’s barcode scanning feature connects directly with the inventory to suggest recipes based on ingredients users already have, making it easier to minimize food waste and grocery expenses. Unlike KitchenPal, YumCycle allows users to submit personal recipes for community sharing, fostering engagement and enriching the recipe database with unique, user-approved options.

Furthermore, YumCycle is designed to be less overwhelming, prioritizing core features like pantry tracking, recipe generation, and expiration reminders for a straightforward, accessible experience. Future plans for dietary filters, family syncing, and smart shopping lists will make YumCycle a highly personalized yet simple solution for managing household food effectively, offering an appealing alternative to KitchenPal’s extensive, premium-locked features.

* **Swing Fridge**

“Swing Fridge is a straightforward, user-friendly tool for tracking your refrigerator's contents. We provide various efficient methods for recording, catering to diverse food requirements. Enhanced with charming emojis and entertaining notifications, you'll discover an entirely novel way to engage with your food. Let's commence with minor changes, curbing food wastage, and elevating our sense of accomplishment.”

A screenshot of a mobile phone

Description automatically generated

YumCycle outshines Swing Fridge by offering more comprehensive functionality without sacrificing ease of use. While Swing Fridge emphasizes simplicity and a fun approach with emojis and notifications, YumCycle goes beyond basic fridge tracking by integrating recipe generation based on existing ingredients, barcode scanning, and expiration reminders to help users make the most of their pantry while reducing waste.

* **kitchin.app**

Kitchin helps you to optimize the running of the most important part of the home, the kitchen… it suggests what you can cook using the ingredients you have, compiles a shopping list for ingredients you need, creates and publishes your own favorite recipes, and reduces waste by helping you to make the most out of food.

**A screenshot of a cell phone

Description automatically generated**

YumCycle surpasses Kitchin in offering a more accessible and community-oriented approach to kitchen management. While Kitchin covers essential functions like ingredient-based recipe suggestions, shopping list creation, and waste reduction, YumCycle provides a more robust experience by allowing users to directly scan barcodes to add items to their inventory, which simplifies and speeds up pantry management.

* **My Fridge Food Tracker**

Harness the power of AI to generate recipe suggestions based on the contents of your kitchen - and never worry about what to eat again! Introducing MyFridge - the modern fridge management app that makes managing your fridge inventory a breeze! You can now generate recipe suggestions based on the contents of your kitchen using AI for free!

A screenshot of a phone

Description automatically generated

YumCycle offers a richer, more community-driven experience compared to MyFridge by integrating both pantry management and meal inspiration in a seamless, user-friendly app. While MyFridge focuses on tracking fridge inventory, expiration notifications, and AI-generated recipe suggestions, YumCycle goes further by providing a community space where users can share and access unique recipes, reviewed by administrators for quality. This feature allows YumCycle to deliver personalized meal ideas that go beyond simple ingredient matching.