1. **Introduction**



FlavourFeed is like a cozy kitchen on the internet where food lovers meet. It is super easy – just like sharing recipes with friends. The website looks great and works smoothly, thanks to simple tools like HTML, CSS, and JavaScript. Behind the scenes, we have got Spring Boot, Hibernate, and MySQL making everything run smoothly. FlavourFeed is not just a place for recipes; it is a friendly community where you can discover new flavors, make friends who love food as much as you do, and maybe even learn a thing or two about what is good for you. Join us and let us create a delicious experience together!

This web application offers a user-friendly platform for exploring, sharing, and contributing to a diverse collection of food recipes, while also empowering users to create personalized diet plans, and access valuable nutritional information.

1. **Problem Definition & Scope**
2. **Problem Definition**

Imagine a world where people who love cooking find it a bit tricky to share recipes and connect with others who enjoy food. Regular cooking websites might be a bit too complicated or not as fun. That is where FlavourFeed comes in! We noticed this problem and want to create a simple, easy-to-use place where everyone can share their favorite recipes, find yummy dishes, and make food-loving friends. So, our mission is to make cooking and sharing recipes online a lot more enjoyable and friendly for everyone!

**2. Goals**

* 1. Create a Friendly Food Community
  2. Seamless Recipe Sharing
  3. Recipe Discovery
  4. Enhance Nutritional Awareness
  5. Interactive User Engagement

**3. Objectives**

* 1. User Registration and Onboarding
  2. Recipe Submission
  3. Efficient Recipe Search and Discovery
  4. Nutritional Information Integration
  5. Community Interaction Features
  6. Responsive Design Across Devices
  7. Security and Privacy Measures
  8. Regular User Feedback and Improvement

1. **Major Constraints**
   1. The web application should support common web browsers such as Chrome, Safari, Firefox, etc.
   2. User Device Compatibility
   3. Compliance with relevant data protection regulations.
   4. Content Moderation
   5. Budget Constraints
2. **Outcomes**
   1. Establishment of a Vibrant Food Community
   2. Simplified Recipe Sharing
   3. Diverse Recipe Discovery
   4. Enhanced Nutritional Awareness
3. **Software Requirements Specification (SRS)**
4. **Team Members (Stake Holders):**
   1. Admin
   2. User
5. **Scope:**

FlavourFeed aims to create a user-friendly culinary platform, enabling users to share, discover, and engage in a vibrant community. The scope includes intuitive recipe sharing, robust search functionalities, nutritional information integration, cross-device accessibility, security measures, continuous improvement based on user feedback, and fostering an interactive and supportive user environment. The platform's goal is to inspire culinary exploration, connection, and a positive experience for food enthusiasts.

1. **Functional Requirements:**
   * **Admin**
   1. The admin should be able to login himself.
   2. The admin should be able to create and manage Users.
   3. The admin should be able to create, modify, and deactivate user accounts.
   4. This includes the ability to reset passwords and manage user roles.
   5. The admin should be able to review, approve, or reject recipe submissions from users after checking for inappropriate content and ensuring compliance with guidelines.
   6. The admin should have tools to monitor and moderate user-generated content, including comments and user interactions.
   7. The admin should have access to user activity logs, including login/logout times, recipe submissions, and other significant actions.
   8. The admin should be able to configure and manage automated email notifications, such as account verification, password reset, and system updates.
   9. The admin should be able to adjust system parameters, such as maximum recipe size, image upload limits, and other application settings.
   10. The admin should be able to respond to user queries and issues, providing assistance and resolving problems.
   11. The admin should be able to communicate important information or updates to users through the platform, such as changes in policies or upcoming maintenance.
   * **User**
2. User should be able to create accounts.
3. User should get secure login and authentication mechanisms.
4. User should be able to submit, edit, and delete recipes.
5. Recipes added by user should include details like ingredients, instructions, and serving sizes.
6. User should be able to search functionality to discover recipes based on ingredients, cuisine, and dietary preferences.
7. User should be able to plan meals for the day or for the week.
8. User can generate a shopping list based on selected recipes.
9. User should be able to customize the shopping list.
10. User should get nutritional content such as calaories per serving for each recipe.
11. User should get aggregate nutritional information for the entire meal plan.
12. User should be able to like, comment, and share recipes.
13. User can follow other users and receive updates on their activity.
14. **Non-Functional Requirements:**
15. **Performance**
    1. The web-application should handle a large user base and increasing data traffic.
    2. The web-application should work fast and availability should be 24x7.
16. **Security** 
    1. Implement secure data storage and transmission.
    2. Protection against web malfunctions such as SQL injections, cross-site scripting.
17. **Usability** 
    1. Provide user-friendly interface
    2. Web-application accessibility on various devices.
18. **Scalability**
    1. The system should be scalable to accommodate future growth.
19. **Reliability**
    1. Minimum downtime for maintenance and updates.
    2. Regular data backups to prevent data loss.

1. **Constraints**

1. The web application should support common web browsers such as Chrome, Safari, Firefox, etc.
2. User Device Compatibility
3. Compliance with relevant data protection regulations.
4. Content Moderation
5. Budget Constraints
6. **Assumptions and Dependencies**
7. Users will have a stable internet connection.
8. Users will provide accurate personal information.
9. Users will provide accurate recipes for food items.
10. Users will provide accurate nutritional information for recipes.
11. **System Modules**
12. **User Management:**

Includes user registration, profile creation, and account management.

1. **Recipe Hub:**

Central module for sharing, submitting, and exploring diverse recipes.

1. **Search and Discovery:**

Functionality allowing users to search for recipes based on criteria like cuisine, ingredients, and dietary preferences.

1. **Nutritional Information:**

Module for integrating and displaying nutritional details for each recipe.

1. **Community Engagement:**

Features for user interaction, including likes, comments, and sharing within the FlavourFeed community.

1. **User Interface:**

Development of a user-friendly interface using HTML, CSS, and JavaScript.

1. **Cross-Device Compatibility:**

Ensuring the platform is accessible and responsive across various devices.

1. **Security and Privacy:**

Implementation of security measures to protect user data and ensure privacy.

1. **Feedback and Improvement:**

Mechanisms for users to provide feedback and continuous improvement based on user suggestions.

1. **Community Guidelines and Moderation:**

Establishment of guidelines and content moderation to maintain a positive and safe environment.

1. **Performance Optimization:**

Continuous efforts to optimize platform performance for a smooth user experience.

1. **External API Integration:**

Potential integration with external APIs for real-time updates or additional features.

1. **User Support and Communication:**

Provision of user support and communication channels for addressing queries and providing updates.

These system modules collectively form the backbone of FlavourFeed, ensuring a seamless, interactive, and enriching experience for users passionate about food and culinary exploration.

1. **Performance Requirements**

Performance requirements for a recipe website would include quick loading times for smooth navigation, dependable and secure data storage to protect user data, and capacity to manage many concurrent users looking for recipes. A pleasant user experience depends on mobile compatibility for access while on the go, as well as user-friendly navigation and an intuitive recipe interface. To improve the recipe-sharing platform, it is imperative to track user interaction, recipe popularity, and other relevant indicators using comprehensive reporting and analytics tools.

1. **Software Requirements**

|  |  |  |
| --- | --- | --- |
| **FRONT END** | **BACKEND** | **DATABASE** |
|  |  |  |
| **REACT JS** | **J2EE** | **MySQL** |

1. **Frontend Technologies:**

**React:**

Version: 18.2.0 (Latest stable release)

Description: JavaScript library for building user interfaces.

**HTML, CSS, JavaScript:**

Utilized for structuring web pages, styling, and client-side scripting.

**Bootstrap:**

Version: 5.3.2 (Latest stable release)

Description: Powerful, extensible, and feature-packed frontend toolkit. Build and customize with Sass, utilize prebuilt grid system and components, and bring projects to life with powerful JavaScript plugins.

1. **Backend Technologies:**

**Spring Boot:**

Version: 2.7.14 (Latest stable release)

Description: Java-based framework for building robust and scalable backend applications.

**Hibernate:**

Version: Latest stable release

Description: Object-relational mapping (ORM) tool for data management.

1. **Database technologies:**

**MySQL:**

Version: 8.0 (Latest stable release)

Description: Relational database management system for storing application data.

1. **Development Environment:**

Integrated Development Environment (IDE): Eclipse, VS Code

Description: An IDE for coding, testing, and debugging application components.

1. **Git:**

Version: 2.42.0.windows.2 (Latest stable release)

Description: Distributed version control system for tracking changes in source code.

1. **Web Server:**

**Apache Tomcat:**

Version: 9.0 (Latest stable release)

Description: Web server and servlet container for hosting the Java-based application.

1. **Security Tools:**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. **Deployment Platform:**

Cloud Service (Optional)

Examples: AWS, Google Cloud, Azure

1. **Testing Frameworks:**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. **Hardware Technologies (Minimum)**

|  |  |  |  |
| --- | --- | --- | --- |
| **Server Side** | | | |
| **Monitor** | **Processor** | **RAM** | **Disk Space** |
| Resolution  1024x768 | Intel or AMD 2GHZ OR HIGHER | 4GB | 10GB |
| **Client Side** | | | |
| **Monitor** | **Processor** | **RAM** | **Disk Space** |
| Resolution: 1024x768 | Intel or AMD 1GHZ DUAL CORE MIN | 2 GB | 128 GB |

1. **Server**

A robust server with sufficient processing power and memory to handle concurrent user requests and ensure smooth application performance.

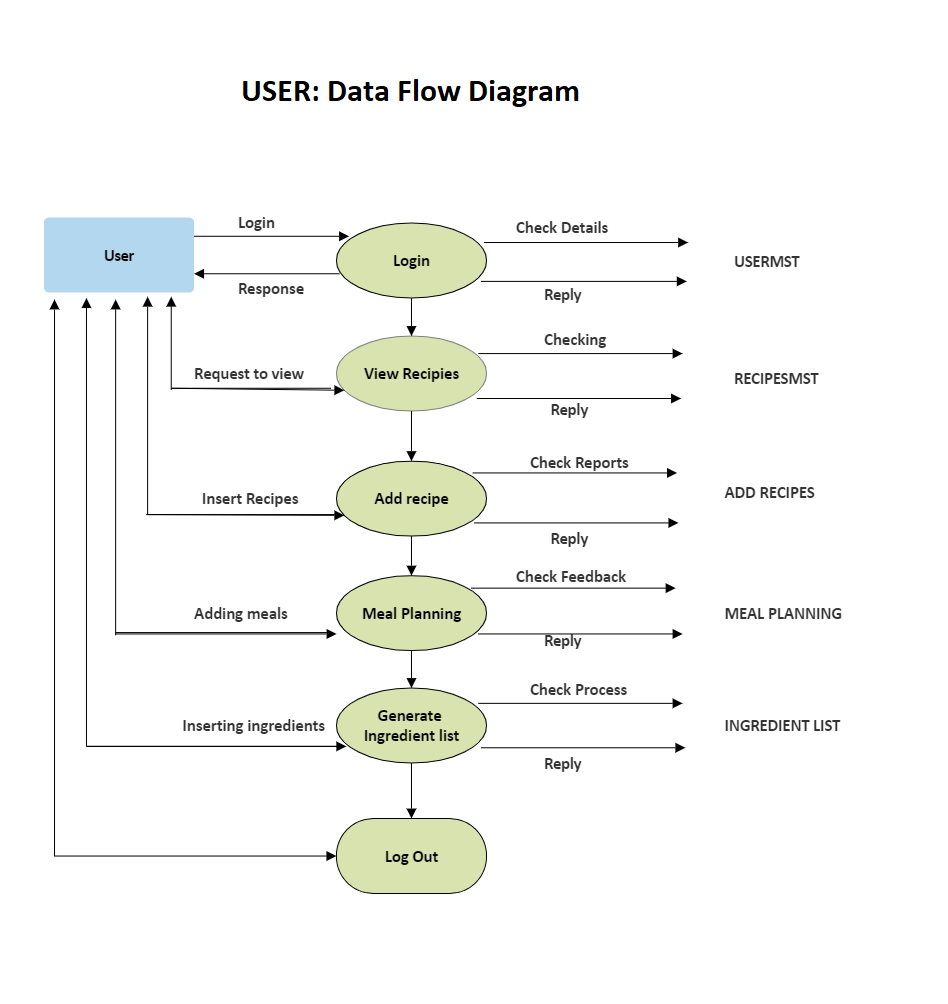
1. **Storage**

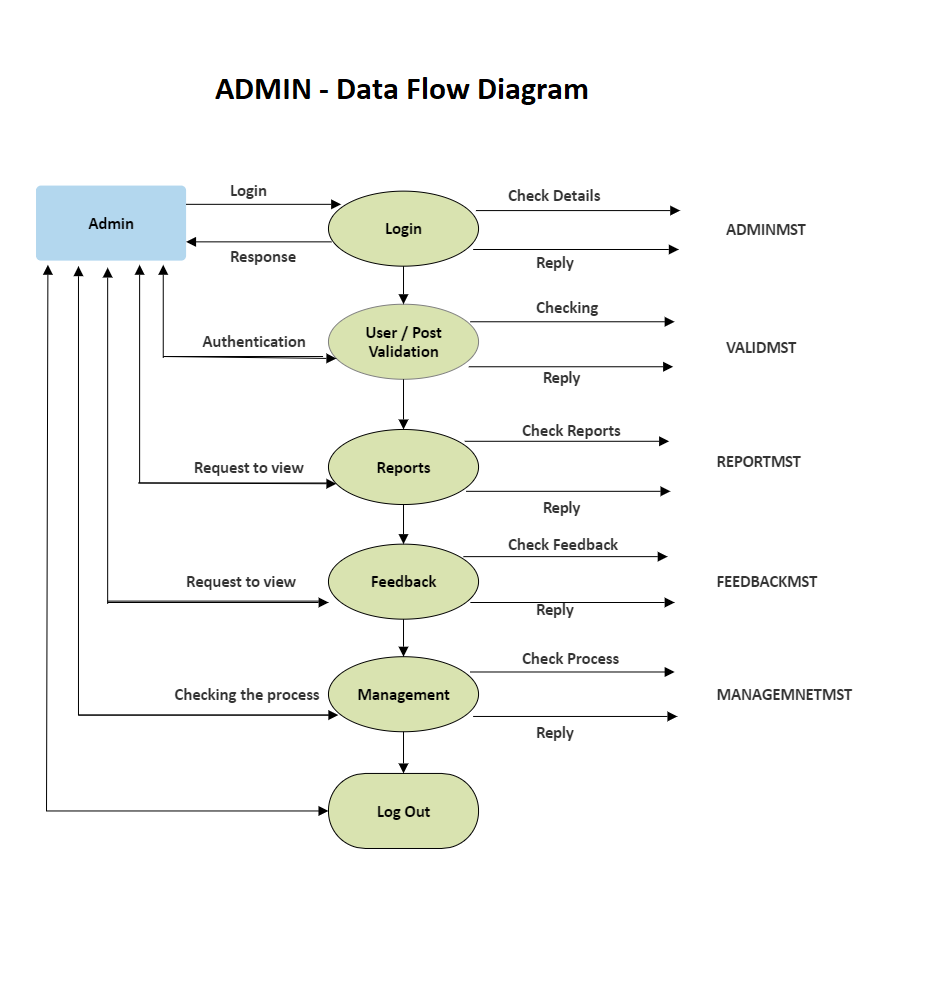
Adequate storage capacity for the database and multimedia content (images, videos) associated with recipes.

1. **Network Infrastructure:**

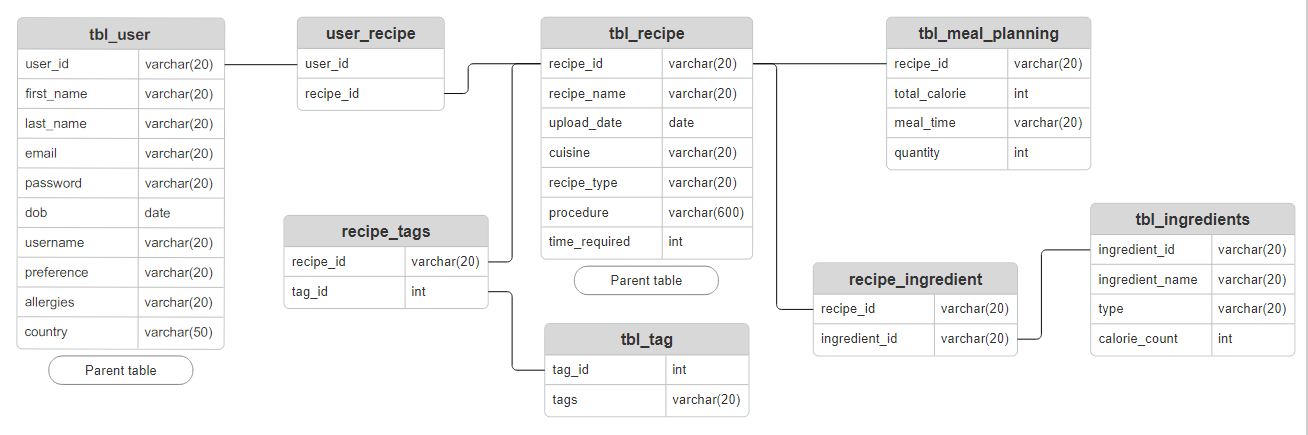
Reliable internet connectivity and network infrastructure to support data transmission between the frontend and backend components.

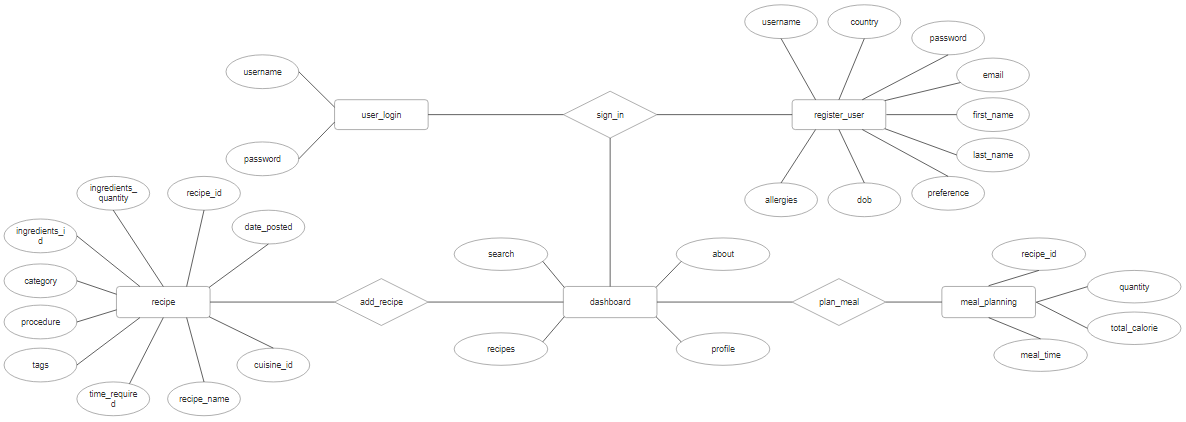
1. **UML Diagrams**
2. **Data Flow Diagram**



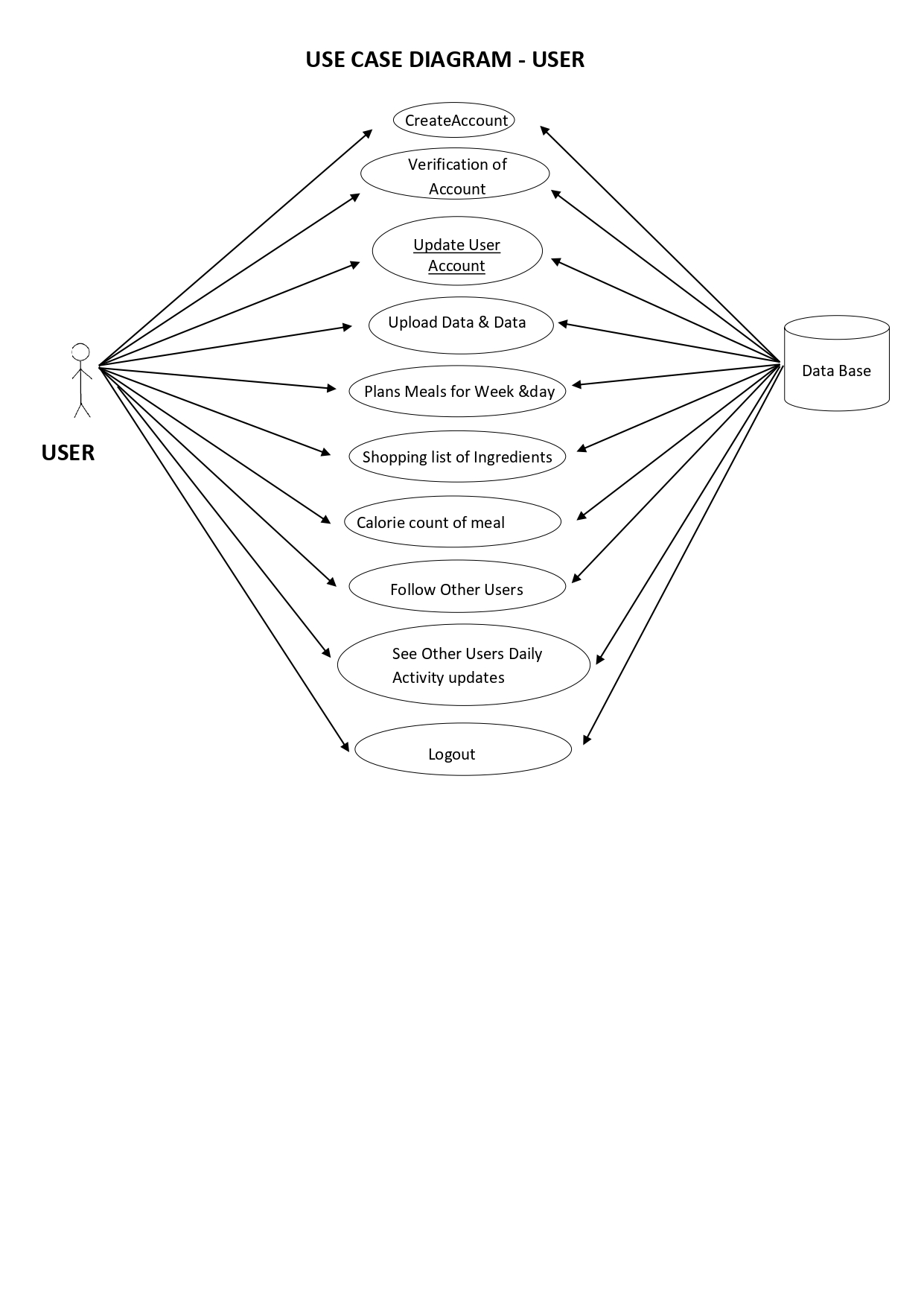


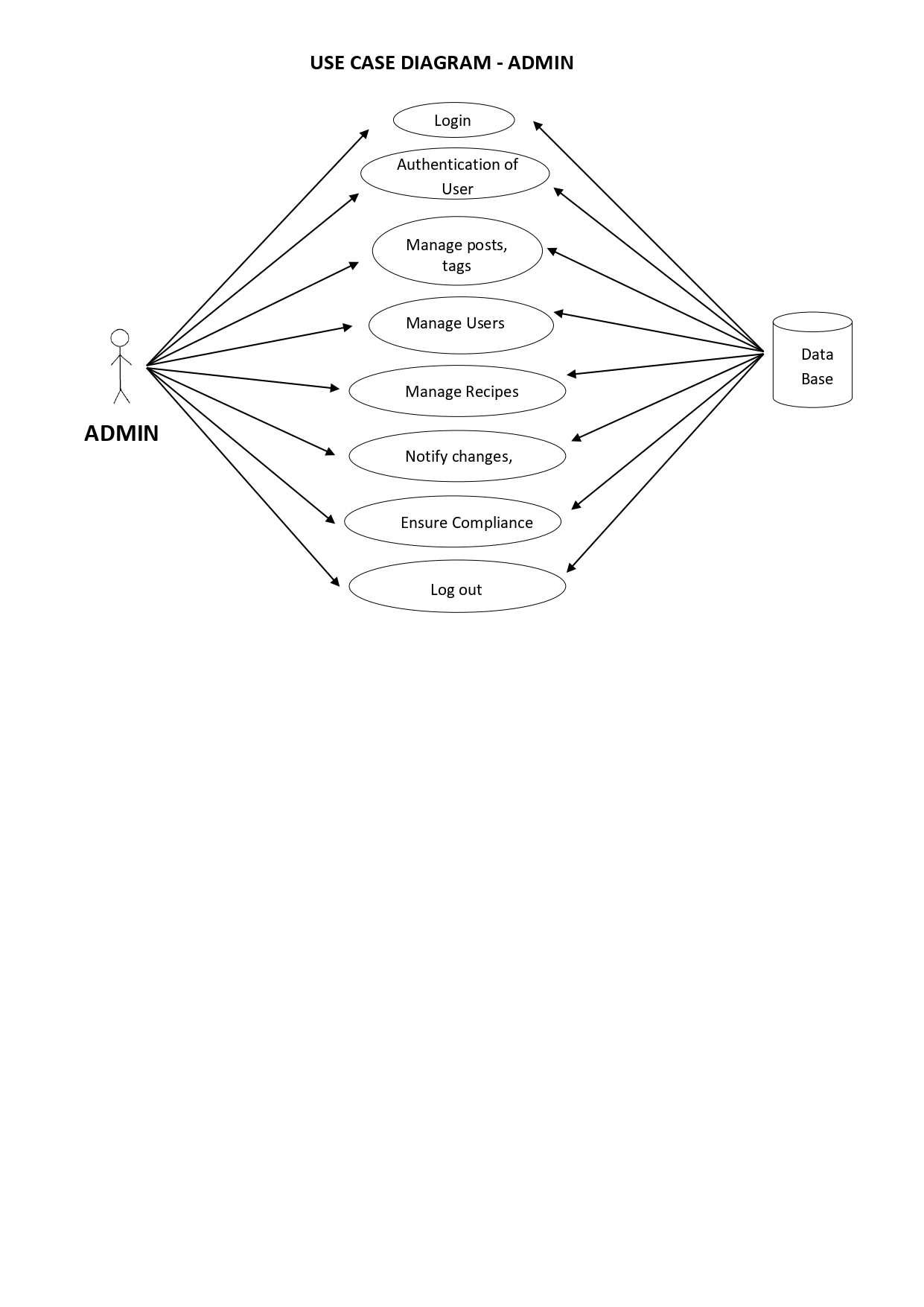
1. **Entity Relation Diagram (ERD)**

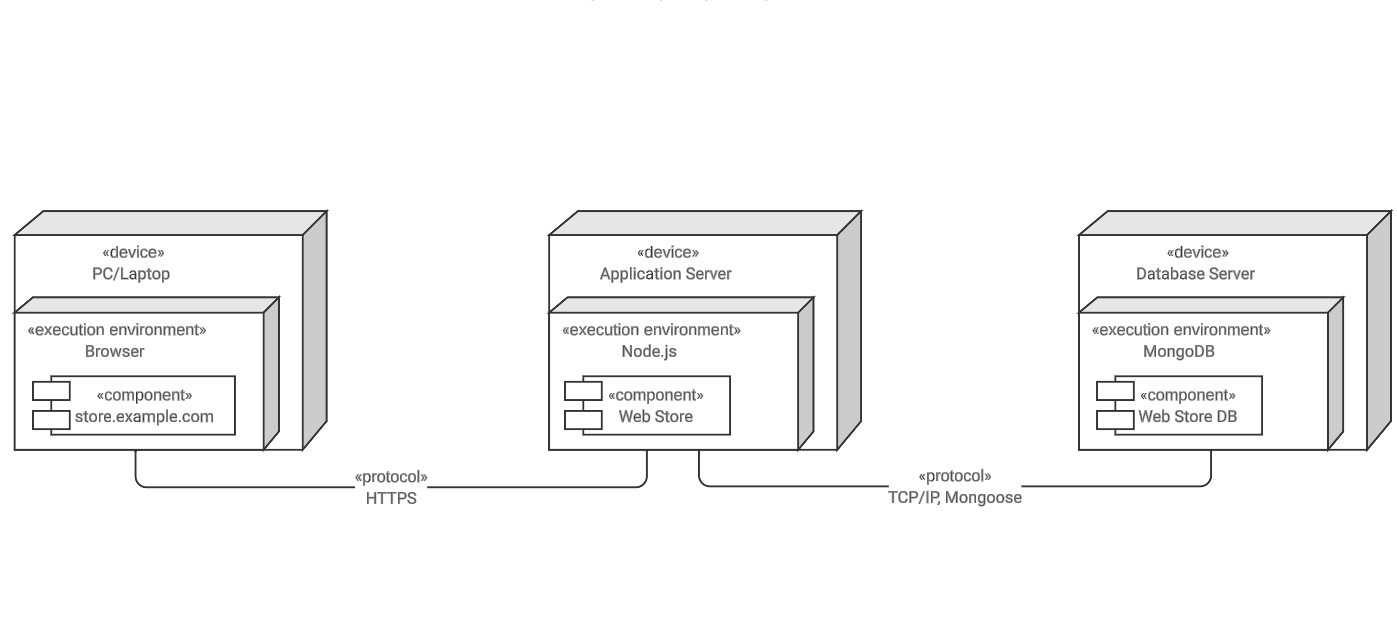




1. **Use Case Diagram**





1. **Class Diagram**
2. **Sequence Diagram**
3. **Activity Diagram**
4. **Deployment Diagram**
5. **System Architecture**



The system architecture for FlavourFeed can follow a typical three-tier architecture, separating the application into three main components: the presentation layer (frontend), the application layer (backend), and the data layer (database). Here is an overview of the system architecture:

1. **Presentation Layer (Frontend):**

Technology: React (HTML, CSS, JavaScript)

Description: The user interface where users interact with the application. React is used for building responsive and dynamic components, allowing for an interactive and engaging user experience.

1. **Application Layer (Backend):**

Technology: Spring Boot (Java)

Description: The backend logic and application processes are managed using the Spring Boot framework. It handles user requests, business logic, and communication with the database.

1. **Data Layer (Database):**

Technology: MySQL

Description: The MySQL database stores user profiles, recipes, and related data. It interacts with the backend to retrieve and update information.

1. **External APIs (Optional):**

Description: Additional external APIs, if integrated, provide services such as nutritional information retrieval. These APIs can be accessed by the backend for enhanced functionality.

1. **Cloud Service (Optional):**

Description: Cloud platforms like AWS, Google Cloud, or Azure can be used for hosting the application, providing scalability, flexibility, and reliable infrastructure.

1. **Development Environment:**

Technology: IntelliJ IDEA, Eclipse (Optional)

Description: Integrated Development Environments (IDEs) are used during the development phase for coding, testing, and debugging application components.

1. **Version Control:**

Technology: Git

Description: Git is employed for version control, allowing multiple developers to collaborate on the codebase, track changes, and manage the development workflow.

1. **Web Server (Optional):**

Technology: Apache Tomcat (or similar)

Description: A web server may be used to host the React frontend files and serve them to users' web browsers.

1. **Test Cases**
2. **Screenshots**

1. **References**