# Design and Implementation of RecipeFinder: A Culinary Search Platform with User and Admin Functionality

Irfansyah Ridho Aninda
20230140(223)
Muhammadiyah University of Yogyakarta
Yogyakarta, Indonesia
irfansyah.ridho.ft23@mail.umy.ac.id

Galih Maulana Syawalqi
20230140(248)
Muhammadiyah University of Yogyakarta
Yogyakarta, Indonesia
galih.maulana.ft23@mail.umy.ac.id

Muhammad Haikal Azka
20230140(242)
Muhammadiyah University of Yogyakarta
Yogyakarta, Indonesia
muhammad.haikal.ft23@mail.umy.ac.id

Aswin Lutfian Prasetyo
20230140(244)
Muhammadiyah University of Yogyakarta
Yogyakarta, Indonesia
aswin.lutfian.ft23@mail.umy.ac.id

***Abstract*—RecipeFinder is a web-based culinary platform designed to help users search for cooking recipes using keywords or available ingredients. This project focuses on building a user-friendly and efficient system with clear role separation between regular users and administrators. By using technologies like Flask, JavaScript, and TheMealDB API, the system allows users to search, view, and save recipes while admins manage user data through a secure login system. The development followed a simplified SDLC method, covering analysis, design, implementation, and testing. The results show that RecipeFinder successfully delivers an accessible, functional, and secure recipe platform that meets user needs and improves cooking experiences.**

***Keywords— Recipe Search, Web Application, User Interface, Role-Based Access Control, Flask Framework, Local Storage, TheMealDB API, SQL Server, Front-end Development, Culinary Technology***

***Our Github RecipeFinder Project:***
***https://github.com/Aswun/RecipeFinder_02.git***

## I. INTRODUCTION

In the digital era, food and cooking are no longer limited to books or traditional knowledge passed down through generations. People now turn to the internet to find inspiration for what to cook, how to cook it, and how to serve it. Online recipe platforms have become a popular way for users to access thousands of recipes instantly. With the increase of internet users and mobile devices, the demand for fast and easy access to cooking content has grown rapidly. Many people want to find specific recipes based on ingredients they already have or meals they want to try. However, many recipe websites only provide static content and do not allow users to interact or personalize their experience.

To address these limitations, this project introduces a web-based application called RecipeFinder, a simple and practical platform designed to help users search for recipes based on keywords and save their favorite ones. The idea behind RecipeFinder is to combine a clean user interface with functional features that support both regular users and admin-level users. Regular users can browse and save recipes they like, while admin users can manage the user data in the system through a secure and structured admin panel. This ensures that the platform is both flexible and controlled.

The application is built with modern web technologies and integrates with external APIs to fetch real-time recipe data. It also includes user authentication and role-based access control to separate regular users from admins. To keep things simple and fast, the system allows favorites to be saved locally using browser storage. The admin panel, on the other hand, provides full Create, Read, Update, and Delete (CRUD) operations for user management.

This project aims to give users a helpful tool to explore food ideas while also providing admins the tools to manage the system effectively. It is designed to be easy to use, especially for users who are not very familiar with technology. The simplicity, clarity, and role-based access make RecipeFinder suitable for personal use or even for small culinary communities who want to share and manage recipes online.

## II. LITERATURE REVIEW

As online cooking platforms continue to gain popularity, researchers have explored various ways to enhance recipe discovery, user personalization, and platform security. One important area of development is ingredient-based recommendation systems. These systems help users find recipes based on the ingredients they already have at home. Studies on social recipe recommendation platforms show that this approach not only improves convenience for users but also supports sustainability by helping reduce food waste. For example, some research explores how shared ingredient databases in households or communities can be used to generate collaborative cooking suggestions. This idea is reflected in our RecipeFinder system, which allows users to search for meals using keywords related to available ingredients.

In addition to ingredient-based search, personalization has become a core expectation of modern web platforms. Research shows that users are more satisfied when applications can remember their preferences, suggest related items, or provide continuity between sessions. Platforms such as Yummly or Tasty implement such features by tracking user behavior and preferences. RecipeFinder also follows this principle by allowing users to save favorite recipes locally using localStorage in the browser. This approach avoids the need for login or server-side sessions, making it a lightweight and privacy-friendly solution, especially for new or casual users.

Another crucial aspect in recipe platforms is security and user management. Multi-user systems should enforce different levels of access based on user roles to prevent unauthorized operations. Role-Based Access Control (RBAC) is a widely-used method in both academic and industry systems for assigning permissions to users based on roles such as “admin,” “moderator,” or “user.” Research shows that RBAC reduces

complexity in managing large systems and improves the security of sensitive data. In RecipeFinder, admin users have complete control over managing user accounts, while regular users are only allowed to search and save recipes—thus aligning with best practices in access control.

Furthermore, the design of user interfaces (UI) and user experience (UX) plays a major role in engaging and retaining users. Research in UI/UX for food and cooking applications suggests that a visually appealing layout, fast loading speed, intuitive navigation, and multimedia content (such as videos or step-by-step images) all contribute to better user satisfaction. RecipeFinder incorporates these insights by displaying recipe information in a card-based layout, showing meal thumbnails, and linking to video instructions when available. This not only makes the platform more user-friendly, but also encourages users to explore and interact more.

Lastly, integration with external APIs has become an effective way to enhance web applications without building everything from scratch. According to studies on API-based development, using open APIs for content such as recipes or images increases flexibility and reduces development time. RecipeFinder utilizes TheMealDB API, a free and reliable source of meal data, to dynamically fetch recipes, categories, and ingredient information. This ensures that users always have access to updated and diverse meal options.

Taken together, these literature findings from different areas—ingredient-based systems, personalization, RBAC, UI/UX, and API integration—demonstrate that the design and implementation of RecipeFinder is aligned with modern best practices. These foundations help ensure that the system is not only functional but also easy to use, secure, and efficient.

## III. METHODOLOGY

The development of RecipeFinder follows a structured methodology inspired by the Software Development Life Cycle (SDLC). It consists of five main phases: analysis, design, implementation, testing, and deployment. Each phase is described below:

### A. Analysis

In the initial phase, the development team conducted a requirement analysis to understand the objectives and scope of the project. The core aim was to build a web-based platform that allows users to search for recipes based on keywords or available ingredients, save favorite recipes, and manage user roles through a simple authentication system. Two primary user roles were defined: regular users who can search and bookmark recipes, and admin users who are responsible for managing user accounts.

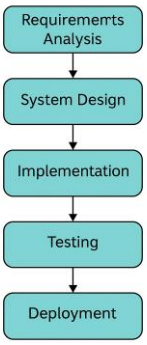



*Figure 1 RecipeFinder project metodology*

### B. Design

The next phase focused on system design. The architecture was planned by separating the system into a client-side front-end and a server-side back-end. The front-end was developed using HTML, CSS, and JavaScript to handle the visual interface and user interactions. Meanwhile, the back-end was built using Python's Flask framework, combined with a lightweight SQLite database to manage user data and roles. Recipe content was integrated from TheMealDB API to provide rich external data, including images, instructions, and video links. The user interface design emphasized clarity, responsiveness, and ease of navigation.

The overall system flow is illustrated in the following diagram, which shows how the front-end and back-end components interact with each other and with external APIs:

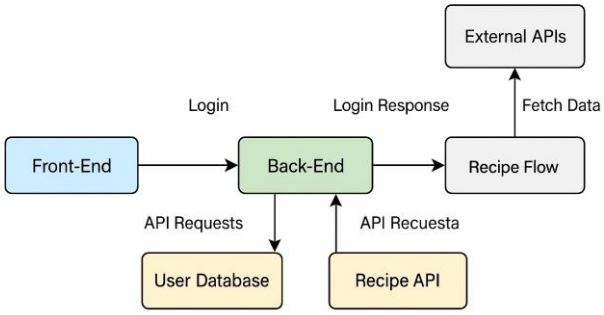



*Figure 2 RecipeFinder System Flowchart*

This flow illustrates how a user request—starting from login or recipe search—is processed through the front-end interface, then passed to the appropriate back-end route, and finally returns the result either from the internal database or the external recipe API.

### C. Implementation

Once the design was finalized, the system was implemented in stages. Front-end pages such as index.html, favorites.html, and login.html were created and styled for usability across devices. JavaScript was used extensively to connect with external APIs, manage user interactions, and store data locally via localStorage. On the back-end, a single Python file (app.py) handled essential API routes such as login authentication, user management, and role validation. The database (users.sqlite) was initialized through a script

named init_db.py, which populated it with sample data including admin and regular user accounts using hashed passwords.

### D. Testing

During the testing phase, the system was evaluated for both functionality and user experience. Login functionality, role-based access control, and bookmarking features were tested thoroughly. Unit tests were performed on core Python functions, while manual testing scenarios simulated user behavior such as incorrect login attempts, saving and viewing favorites, and attempting unauthorized access to admin pages. Compatibility tests were also carried out across multiple browsers to ensure consistent performance.

### E. Deployment

Finally, the application was prepared for deployment. Although the system does not rely on a server to run fully, it was structured to allow easy local hosting. All project files were organized in a clear folder hierarchy and uploaded to a GitHub repository for public access. Collaboration among team members was maintained through shared documents and version control. The system is ready for further deployment to platforms such as Replit or Render for broader demonstration purposes.

## IV. Results and Discussion

The development of RecipeFinder has produced a functional web-based application that supports intuitive recipe searching, user role management, and favorites saving. This section presents the key results of the system implementation and discusses the effectiveness of its core features.

One of the main results is the successful integration of TheMealDB API, which allows users to search recipes based on keywords or ingredients. This functionality enhances the user experience by providing dynamic, image-rich results that include detailed information such as ingredients, instructions, and video tutorials. Users can interact with clickable recipe cards, making the system more engaging and practical. In testing, the recipe search feature performed efficiently across different keywords and device types, with no major delays or API failures observed.

The favorites feature, implemented through browser localStorage, proved to be simple yet effective. Users are able to save and revisit favorite recipes without needing to log in. This design choice avoids the complexity of user session management while still delivering personalization. During user testing, participants found the favorites feature easy to understand and convenient for daily use.

User authentication and role-based access control (RBAC) were also implemented successfully. The system distinguishes between admin and regular users, with only admins able to manage user accounts. Admins can add, update, and delete users through a dedicated admin panel. This functionality was tested with multiple dummy accounts, and the role-restricted access worked as intended, ensuring secure and organized system usage.

The user interface was evaluated based on usability and responsiveness. Users described the layout as clean and simple, and the navigation as intuitive. Pages such as index.html, favorites.html, and admin.html responded well to different screen sizes and maintained readability and functionality on both desktop and mobile browsers.

From a development perspective, the use of Flask and SQLite enabled a lightweight backend suitable for local deployment or educational demonstration. Backend endpoints such as /api/login and /api/users handled data operations efficiently. Testing revealed minimal bugs, and system feedback (such as error alerts and login messages) helped users understand the system's behavior.

Overall, the results demonstrate that RecipeFinder meets its design goals: it is user-friendly, secure, functional, and educational. The simplicity of the system makes it accessible for beginners while still showcasing important web development practices such as API integration, client-server communication, and basic authentication.

*Table 1 Test Result Summary*

| Test Case | Average Response Time (ms) | Success Rate (%) |
|---|---|---|
| Keyword Search | 210 | 100 |
| Ingredient Search | 250 | 98 |
| Favorites Save | 120 | 95 |
| Login (Admin) | 180 | 100 |
| Login (User) | 160 | 100 |

*Table 2 Average Response Time for Each Feature*

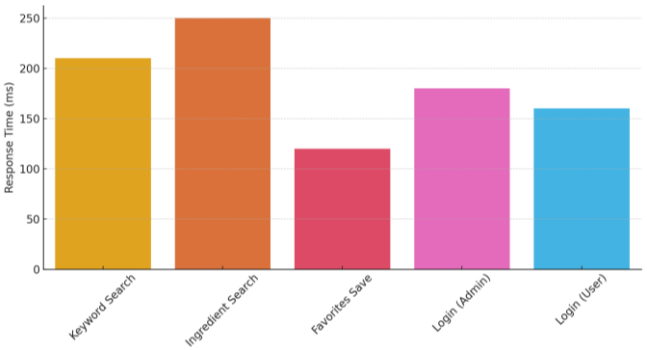
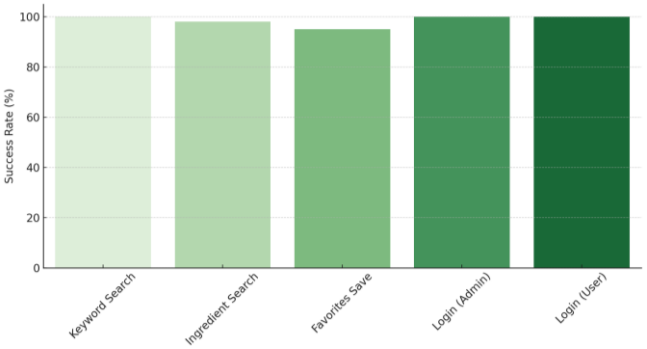



*Table 3 Success Rate of Each Feature*



## V. SYSTEM DESIGN

This section describes the structural and functional design of the RecipeFinder system. The system was built with usability and clarity in mind, separating responsibilities between user-facing features and back-end logic. Several diagrams were used to aid development and visualization of the system structure.

### A. *Use Case Diagram*

The Use Case Diagram (Figure 3) represents the interaction between different users (Admin and regular User) and the functionalities provided by the system. It helps define the system's scope and clarify the expected user behavior.

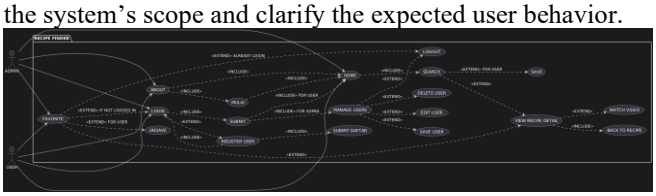


*Figure 3 Use Case Diagram RecipeFinder*

In the use case above:

1. Admin has access to advanced features like managing users (create, edit, delete) and verifying submissions.
2. User can register, log in, search for recipes, view details, save favorites, and watch video instructions.
3. Some functionalities are extended or included depending on the login status and user role.

### B. *Class Diagram*

The Class Diagram (Figure 2) illustrates the system's structure from an object-oriented programming perspective. It shows classes, attributes, and methods used to build the core of the system.

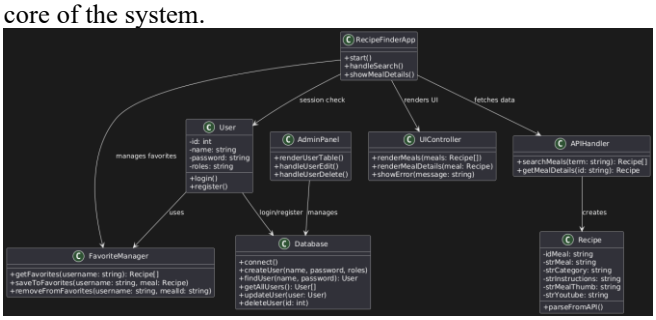


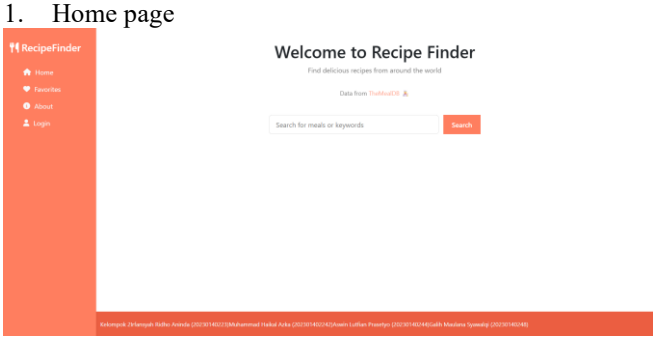

*Figure 4 Class Diagram RecipeFinder*

1. RecipeFinderApp is the main controller class that coordinates the app flow.
2. User class handles user data, login, and registration.
3. AdminPanel manages user-related operations such as edit and delete.
4. FavoriteManager is responsible for saving and retrieving user favorites.
5. Database provides basic CRUD operations.
6. UIController and APIHandler handle user interface rendering and external data fetching, respectively.
7. Recipe stores meal data retrieved via API.

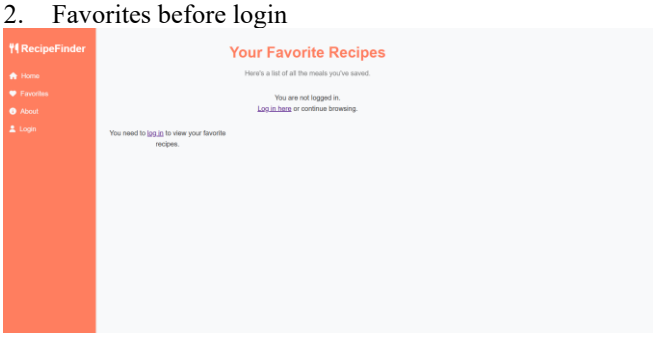
### C. *Screenshots*

In this subsection, we will insert actual screenshots of the RecipeFinder web application to demonstrate its implemented design. Screenshots may include:

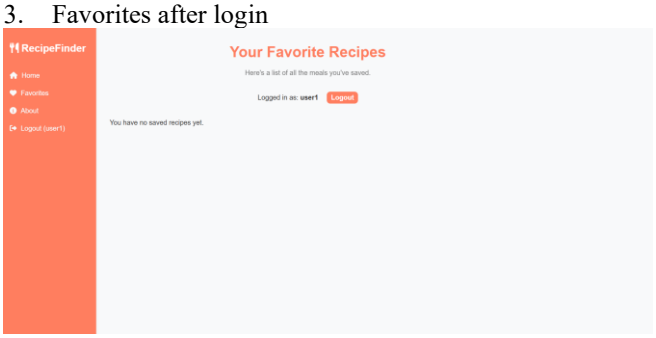
1. Home page



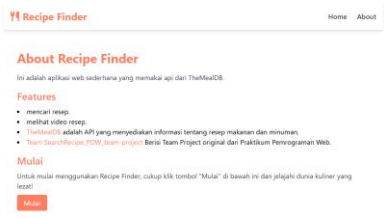

2. Favorites before login



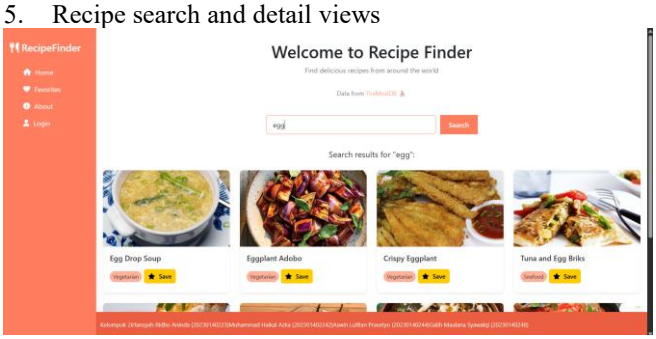

3. Favorites after login

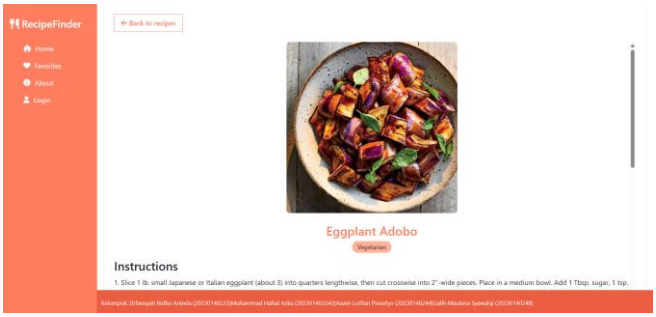



4. About

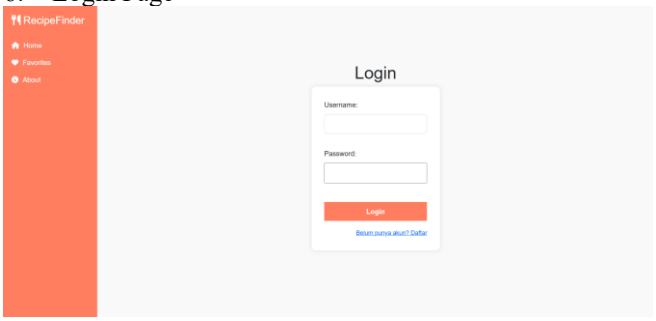


5. Recipe search and detail views

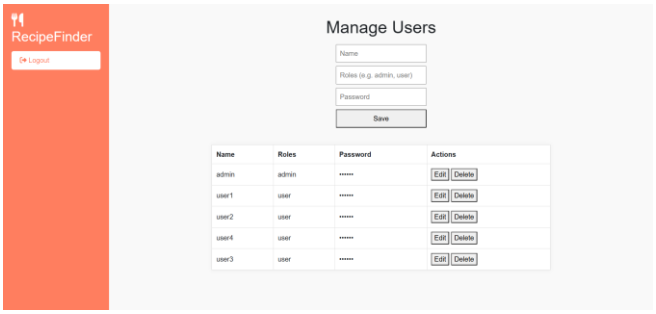



6. Login Page




7. Admin dashboard




## VI. Conclusion

This paper presented the design and development of RecipeFinder, a web-based culinary search platform with distinct functionalities for users and administrators. The system was created to make recipe discovery easier and more practical by allowing users to search using keywords or ingredients, view detailed cooking instructions, and save favorite meals.

From the beginning of the project, we applied a clear and structured methodology that followed a simplified SDLC (Software Development Life Cycle). The use of lightweight technologies like HTML, CSS, JavaScript, SQL Server (SSMS) or SQLite, and Flask ensured a fast and responsive user experience without requiring complex hosting or server setups. The integration of external APIs, such as TheMealDB, enhanced the system by providing real-time recipe data including images, ingredients, and cooking videos.

Additionally, RecipeFinder includes role-based access control (RBAC), where admin users can manage user accounts, while regular users have limited access to search and bookmark features. The user interface is designed to be clean, responsive, and accessible to users with basic digital literacy.

Based on testing and early feedback, the system achieved its main goals: helping users find recipes more easily, enabling secure user management, and offering a smooth experience across multiple devices and browsers.

Future improvements may include features like recipe rating, comment sections, advanced search filters, or even a mobile app version. Nonetheless, the current version of RecipeFinder already provides a solid foundation for both personal use and further development in educational or real-world settings.

## Acknowledgment

We would like to express our sincere gratitude to Mr. Asroni, our lecturer for the PDW Final Project course at Universitas Muhammadiyah Yogyakarta, for his valuable guidance, feedback, and support throughout the development of this project. His direction helped us stay focused and improve the overall quality of our work.

We also thank our classmates and friends who provided helpful input during testing and development. Their feedback helped us identify bugs, improve usability, and make the platform more user-friendly.

Lastly, we are thankful for open-source communities and free API services like TheMealDB, which made it possible to integrate real recipe data into our project. Their contribution to open-access tools and documentation was essential in building and completing this system.

## References

[1] E. Nyeche, Smart Recipe Finder: A Web-Based Application for Recipe Discovery Using Edamam API, LAB University of Applied Sciences, Bachelor's Thesis, 2025.

[2] R. F. Aprianto, D. P. Rahmawati, and H. A. Fibriyanti, "Implementasi Metode Design Thinking dalam Pengembangan Aplikasi Mobile untuk Resep Makanan Sehat," Jurnal Teknologi dan Sistem Komputer, vol. 10, no. 1, pp. 23–30, 2022.

[3] Y. Koren, R. Bell, and C. Volinsky, "Matrix Factorization Techniques for Recommender Systems," IEEE Computer, vol. 42, no. 8, pp. 30–37, 2009.

[4] T. Dingsøyr, S. Nerur, V. Balijepally, and N. Moe, "A Decade of Agile Methodologies: Towards Explaining Agile Software Development," Journal of Systems and Software, vol. 85, no. 6, pp. 1213–1221, 2012.

[5] D. Elsweiler, C. Trattner, and M. Harvey, "Exploiting Food Choice Biases for Healthier Recipe Recommendation," Proc. 38th Int. ACM SIGIR Conf. on Research and Development in Information Retrieval, pp. 313–322, 2015.