ISSN 2355-0082

Personalized Restaurant Recommendations: A Hybrid Filtering Approach for Mobile Applications

Christopher Matthew Marvelio[1], Alexander Waworuntu[2]
[1,2] Department of Informatics, Universitas Multimedia Nusantara, Tangerang, Indonesia
[1]christopher.marvelio@stutent.umn.ac.id, [2]alex.wawo@umn.ac.id

Accepted 16 June 2025
Approved 15 July 2025

***Abstract*— Selecting a restaurant that suits your taste can be a major challenge for consumers, especially given the vast array of online dining options. Traditional recommendation systems or simple filtering methods often fail to handle this complexity well. To address these limitations, we developed a mobile app-based restaurant recommendation platform that combines content-based filtering and collaborative filtering methods in a hybrid approach. The application was built using Expo, React Native, Express, and Flask technologies. The evaluation was conducted using the End-User Computing Satisfaction (EUCS) framework, and the results showed a very high user satisfaction rate of 93.9%. This result demonstrates that the recommendation system we developed is effective in providing relevant suggestions and is well-received by users.**

***Index Terms*— Collaborative Filtering; Content-Based Filtering; Hybrid Filtering; Mobile Application; Restaurant Recommendation**

I. INTRODUCTION

The rapid development of digital technology has brought about major changes in various aspects of human life. People now communicate, search for information, and organize daily activities more easily and efficiently [1] [2]. This progress not only makes life more practical but also opens up new opportunities in the fields of education, entertainment, and daily lifestyles. One real form of this progress is the existence of mobile applications that have now become an important part of modern society [3]. In the past, applications only functioned as a simple tool. However, now, applications have developed into smart solutions that support work, maintain health, and facilitate access to consumption services, such as online food delivery. These services are very helpful, especially for people who are busy, have no time to cook, or have difficulty leaving the house [4]. However, these conveniences also raise a new challenge. The many menu choices often make it difficult for users to determine foods that suit their taste. The results of a survey from 1,000 users of online food delivery services in Indonesia (age 18–45 years) show many users felt frustrated when they had to choose food because too many choices didn't match their personal preference [5].

To overcome the problem of confusion in choosing food, a restaurant recommendation system can be a very helpful solution. This system is designed to provide suggestions for places to eat or menus that suit the user's taste [6]. One method that is quite effective in this system is Collaborative Filtering (CF). The CF method works by analyzing the assessment patterns and habits of users in choosing food. Based on this data, the system looks for similarities between one user and another. Thus, the system can recommend food that is liked by people who have similar tastes [7]. However, although quite effective, the CF method also has several weaknesses. One of them is the cold start problem, which is when the amount of user data is still small, so the system has difficulty providing accurate recommendations. In addition, CF is also less effective in handling menus that are rarely rated, and it requires a large computational process if the data is very large. To overcome these weaknesses, some previous studies have tried to use the user-based CF approach, which relies on user assessment and demographic data like age or gender to find users who have similarities. In fact, there is also research that combines CF with the matrix factorization method so the food recommendation system can work more optimally.

Besides the CF method, the Content-Based Filtering (CBF) method is also often used in recommendation systems. CBF works by analyzing the characteristics of items, such as food types, and matching them with the preferences of each user [8]. Unlike CF, which relies on the opinions of many users, CBF focuses more on the details of the item itself, so it can provide more personal and specific suggestions. However, CBF also has several weaknesses. This method requires a complete description of the item and tends to only suggest items that are similar to those previously liked, making the recommendations given less varied. In the context of food recommendations, this can cause users to receive suggestions that are too limited. A number of studies on CBF have tried to develop food recommendation systems not only based on user tastes but also adjusted

to healthy eating patterns based on the food's nutritional value. In addition, various recent studies have developed more sophisticated mobile application-based recommendation systems by combining techniques such as matrix factorization, feature extraction, and user location information. With the addition of these techniques, the system is expected to provide more accurate and contextual recommendations, according to user conditions in real-time.

With the various limitations of each method, it's important to continue developing how recommendation systems work so they can provide more accurate suggestions tailored to user needs. One promising solution is a hybrid recommendation system, which combines Collaborative Filtering (CF) and Content-Based Filtering (CBF) methods [9]. Research shows that hybrid recommendation systems generally perform better than using only one method [10]. By combining the advantages of CF—which analyzes patterns from many users—and CBF—which focuses on item characteristics—this system is able to provide more relevant, diverse, and personalized suggestions. There are various ways to combine these two methods in a hybrid recommendation system. One is weighted combination, where the results of CF and CBF are combined by assigning specific weights to each result, allowing the system to adjust how much influence each method has. Additionally, a switching strategy approach dynamically selects the most appropriate method based on user data conditions—for example, using CBF when user data is still limited and switching to CF when user data is sufficient. Another method is feature combination, which integrates information from user and item characteristics into a more comprehensive recommendation model. Finally, there's a multilevel structure approach, where the recommendation process is carried out in stages; for instance, the initial stage filters items using one method, and the next stage refines the results with another. These four approaches enable hybrid recommendation systems to work more flexibly and effectively in generating relevant and diverse suggestions for users. One study applied a hybrid model to a music playlist recommendation system, combining collaborative data and song characteristic information, thus overcoming the problem of limited data. Another study developed a hybrid system specifically for food and restaurant recommendations, and the results show this approach has great potential to improve the quality and accuracy of recommendations [11].

II. Theoretical Framework

A. Machine Learning

Machine Learning is a field of science that focuses on how to make computer systems learn by themselves from data, without having to be specifically programmed for every situation. So, instead of writing code for every possibility, developers simply provide data, and the machine will learn from there. In this way, Machine Learning algorithms can carry out tasks automatically after learning and processing the available information. This ability to keep learning and adapting is the main strength of Machine Learning and is the reason why this technology is widely used in various aspects of life. In recommendation systems, Machine Learning plays an important role. This technology helps the system recognize user tastes and understand the characteristics of existing items. As a result, the system can provide more appropriate suggestions that are in accordance with user needs [12].

B. Recommendation System

Recommendation systems are intelligent platforms designed to provide suggestions tailored to each user's taste and needs. They are used in many fields and help improve the user experience of the services or products they use. Their main goal is to suggest items or content that users are more likely to find interesting or useful [13]. To do this, recommendation systems use special models and algorithms that analyze data. These algorithms study data from previous user activity, such as search history or ratings, to predict what users might like in the future. In this way, the system helps users make more informed choices that are in line with their interests [14].

C. Collaborative Filtering (CF)

Collaboration Filtering (CF) is a common and proven approach in recommendation systems. CF works by looking for patterns of similarity between users or items to generate suggestions [15]. Its main goal is to predict how a user will rate or respond to an item they haven't seen or used before.

CF has two main methods: user-based CF and item-based CF. In user-based CF, the system searches for other users who have similar preferences to the target user. After that, the system recommends items that are liked by similar users. In contrast, item-based CF focuses on finding items that are similar to items that have been liked by the target user. This similarity is calculated based on rating patterns from many users. As a result, the system suggests similar items to the users [16].

One of the main advantages of CF is that it does not require detailed item descriptions or complete user profiles. However, CF also has several challenges. One of them is the data scarcity problem, which happens when the initial data is still small, so the system has difficulty providing accurate recommendations. There is also the cold-start problem, which is the difficulty of giving suggestions for new users or new items that don't have any history of usage. In addition, if the number of users and items increases, CF systems can require large computing power to process all this information [17].

To measure similarity in CF, various techniques are employed. For item-based collaborative filtering, a set of items rated by a user is utilized to identify the most similar items based on their ratings. The Mean Squared Differences (MSD) method is used to quantify the similarity between items by evaluating the accuracy of predicting one item as a sole recommender for another [18].

The prediction of a rating for an item a by user u (Pu,a) using only one neighboring item b is calculated as shown in Equation (1):

$$P_{u,a} = \bar{r}_a + \left(r_{u,a} - \bar{r}_b\right) \quad (1)$$

In Equation (1), $r_{u,a}$ represents the rating of item a by user u. $\bar{r}_a$ and $\bar{r}_b$ are the average ratings for item a and item b, respectively.

The MSD similarity ($\text{iSim}_{a,b}^{MSD}$) is defined as shown in Equation (2):

$$i\text{Sim}_{a,b}^{MSD} = 1 - \left(\frac{\sum_{u=1}^{|U_{a\sim b}|}\left(P_{u,a} - r_{u,a}\right)^2}{|U_{a\cap b}|}\right) \quad (2)$$

This formula measures similarity based on the differences between predicted and actual ratings of common users. The scores are normalized using max-min normalization to ensure values fall between 0 and 1.

Furthermore, the Ochiai similarity measure is used to consider the percentage of common users who have rated both items ($U_{a\cap b}$) relative to the total number of users who have rated each item individually (U_a and U_b), as shown in Equation (3):

$$\text{iSim}_{a,b}^{\text{Ochiai}} = \frac{|U_{a\cap b}|}{\sqrt{|U_a| \times |U_b|}} \quad (3)$$

The overall CF similarity ($\text{iSim}_{a,b}^{CF}$) is then computed by combining the MSD and Ochiai similarities, as shown in Equation (4):

$$i\text{Sim}_{a,b}^{GF} = i\text{Sim}_{a,b}^{MSD} \times i\text{Sim}_{a,b}^{Och} \quad (4)$$

D. Content-Based Filtering (CBF)

Content-Based Filtering (CBF) is an algorithm used in recommendation systems that operates by recommending content similar to a user's past interactions. This algorithm analyzes the characteristics of the content a user has interacted with, such as keywords, topics, or genres, to predict their preference for similar content in the future. This approach helps users to discover content that aligns with their historical taste and preferences [19].

A key advantage of CBF is its ability to provides recommendation for new user or new items without needing historical data from other user. This makes it particularly useful in "cold start" scenario where collaborative filtering method might struggles. However, a significant drawback from CBF is overspecialization. This happens because recommendation are limited to items very similar with the user's past preferences, which can reduce diversity and prevent discovering new and various item. This limitation often arise when item description or category is incomplete [8].

In item-based content similarity, each item are associated with its respective category. If two item share the same categories, they considered related or similar to each other. All items are represents as vectors with values of [0, 1].

The representation of an item a as a vector (v_a) is given by:

$$v_a = \left(v_{a,1}, v_{a,2}, \ldots, v_{a,c}\right) \quad (5)$$

$$v_{a,c} = \begin{cases} 1 & \text{if item a belongs to category C} \\ 0 & \text{if item a does not belong to category C} \end{cases}$$

Subsequently, vector-based cosine similarity is employed to compute the content-based similarity between pairs of items. This measurement quantifies the cosine from the angle between two vectors, indicating their similarity.

$$\text{Sim}_{a,b}^{\text{Content}} = \frac{\sum_{u=1}^{n} V_a \times V_b}{\sqrt{\sum_{u=1}^{n} (V_a)^2} \times \sqrt{\sum_{u=1}^{n} (V_b)^2}} \quad (6)$$

E. Hybrid Content-Based and Collaborative Filtering

Hybrid recommendation systems represent an advanced approach that combines Collaborative Filtering (CF) and Content-Based Filtering (CBF) to overcome the individual limitations of each method and achieve higher recommendation accuracy. By integrating the strengths of both, these systems can leverage general knowledge derived from user interactions (CF) and the detailed item characteristics (CBF) to broaden the scope of recommendations. This comprehensive approach allows the system to consider a wider range of factors, such as user preferences and item attributes, thereby enhancing the overall quality and relevance of recommendations.

Various techniques can be employed to implement hybrid recommendation systems:

- Switching: Selects the best recommendation method based on the specific situation or context.
- Mixed: Integrates recommendations from various methods into a single list.
- Feature Combination: Merges features from different methods. This method is often chosen for its ability to optimally combine the advantages of both CF and CBF.
- Feature Augmentation: Adds new features to improve recommendations.

- Cascade: Helps resolve conflicts when recommendations from different methods contradict each other.
- Meta-level: Uses the output of one method as input for another.

This research specifically utilizes the Feature Combination method for the food recommendation system. This choice was motivated by its proven effectiveness in optimally merging the strengths of both CF and CBF. CF relies on past user behaviors and preferences for recommendations, while CBF uses item or food information and their characteristics. By combining the features from both methods, the recommendation system can generate more accurate and relevant suggestions for each user. The effectiveness of the Feature Combination method in improving recommendation accuracy has been shown in previous studies. For example, one study applied feature combination in a recommendation system by integrating a user-based CF approach with demographic information. Another research focused on a music playlist recommendation system using feature combination, which merged collaborative information from music playlists with song feature vectors from different sources. This integration helped solve data sparsity and improved song representations, leading to better recommendations, especially in cold-start cases and for songs that were not popular.

The hybrid recommendation process, particularly using the feature combination method, typically involves three stages:

1. Item-based Collaborative Filtering Similarity: Calculates the similarity between items using CF principles.
2. Item-based Content Similarity: Calculates the similarity between items using CBF principles.
3. Hybrid Prediction: Combines the similarities calculated in the previous two stages to generate final predictions.

In the hybrid prediction step, the predicted rating for an unknown item a by user u is typically derived through a two-step process. First, a Weighted Sum approach calculates the total score for an item by combining the predicted scores from both Collaborative Filtering (CF) and Content-Based Filtering (CBF). This weighted sum is based on the ratings provided by user u for items b that are most similar to item a, as used in both item-based CF and item-based content methods for prediction. The predicted score from Content-Based Filtering ($P_{u,a}^{\text{Content}}$) is given by Equation (7):

$$P_{u,a}^{\text{Content}} = \frac{\sum_{b \in I} \left(r_{u,b} \times \text{Sim}_{a,b}^{\text{Content}} \right)}{\sum_{b \in I} \left| \text{Sim}_{a,b}^{\text{Content}} \right|} \quad (7)$$

Meanwhile, the predicted score from Collaborative Filtering ($P_{u,a}^{CF}$) is given by Equation (8):

$$P_{u,a}^{CF} = \frac{\sum_{b \in I} \left(r_{u,b} \times \text{Sim}_{a,b}^{CF} \right)}{\sum_{b \in I} \left| \text{Sim}_{a,b}^{CF} \right|} \quad (8)$$

Subsequently, linear weighted hybridization is applied to combine these predicted ratings from both item-based CF and item-based content methods to produce a final hybrid prediction. This is represented by Equation (9):

$$P_{u,a}^{\text{Hybrid}} = \lambda \cdot P_{u,a}^{CF} + (1 - \lambda) \cdot P_{u,a}^{\text{Gontent}} \quad (9)$$

In Equation (9), λ and 1−λ (where λ∈[0,1]) represent the relative significance of the item-based CF and item-based content predictions in the final hybrid prediction.

F. Vector Space Model (VSM)

The Vector Space Model (VSM) is a conceptual framework frequently used in information retrieval to represent documents in a way that makes comparison and searching easier. In VSM, each document is conceptualized as a point inside a multi-dimensional space. Every dimension in this space corresponds to a unique word or term that is present across the document collection [20].

The presence and significance of a word in a document are quantified and represented by a numerical weight along its corresponding dimension. When a search query is initiated, it is also transformed into a vector within the same multi-dimensional space. The similarity or relevance between documents, or between a document and a query, is then determined by the "closeness" of their points (vectors) in this space. For example, a smaller angle between two vectors indicates a higher similarity [21].

The main goal of VSM is to place documents with similar topics or content close to each other in the vector space. This arrangement makes it easier to find relevant documents during a search. VSM proves very effective because it allows the calculation of similarity using mathematical formulas, making the search process both efficient and more effective.

G. Singular Value Decomposition (SVD)

Singular Value Decomposition (SVD) is a powerful mathematical technique used to break down a rectangular matrix into three simpler and more interpretable matrices. The three matrices consist of two orthogonal matrices and one diagonal matrix [22].

The main utility of SVD lies in its ability to reveal hidden patterns and structures in data by transforming it into a form that is easier to analyze. The two orthogonal matrices resulting from this decomposition represent the rows and columns of the original matrix, but in a way that highlights the relationships between them. Meanwhile, the diagonal matrix contains singular values, which show the "strength" or importance of those identified relationships. This decomposition

makes SVD an a useful tool for dimensionality reduction, noise reduction, and identifying latent semantic factors in data, which can be really beneficial in various data analysis or machine learning applications [23].

H. End-User Computing Satisfaction (EUCS)

End-User Computing Satisfaction (EUCS) is a recognized method designed to measure the level of satisfaction users have with an information system. The main purpose for using EUCS is to determine how effectively a system fulfills user requirements by comparing their expectations with the actual performance and features of the system [24].

The EUCS model is structured around five key aspects that are considered influential in shaping user satisfaction [25]:

- Content: This aspect pertains to the quality and completeness of the information provided by the system.
- Accuracy: This refers to the correctness and truthfulness of the information generated and presented by the system.
- Format: This dimension evaluates how easily information displayed by the system can be read and understood.
- Ease of Use: This focuses on how simple and straightforward the system is for users to learn and interact with.
- Timeliness: This relates to the promptness and efficiency with which the system delivers necessary information to the user.

By evaluating these five criteria, EUCS provides a comprehensive assessment of user satisfaction, making it a valuable tool for system developers and researchers to understand user perception and identify areas for improvement.

III. METHODOLOGY

The research employs a structured approach encompassing several distinct phases to achieve its objective. This systematic progression begins with a comprehensive literature review, followed by meticulous data collection, a well-defined system design phase, subsequent implementation, and concludes with a thorough evaluation of the developed system. This method is well-suited for the development and assessment of a mobile-based recommendation system, allowing both the creation of the system and the measurement of its effectiveness from a user satisfaction perspective.

Data collection and preprocessing were critical steps in preparing the necessary information for the restaurant recommendation system. The primary data source of this research was the Grab Food API. Through API calls, a comprehensive dataset was acquired, including merchant info, detailed menu descriptions, restaurant locations, pricing, categories, and user ratings. This raw data was then subjected to a rigorous preprocessing pipeline to ensure quality, consistency, and suitability for filtering algorithms. Key preprocessing steps involved cleaning and transforming the raw data to fix any inconsistencies or errors. Specifically, the 'tags' column, which was initially stored in JSON format in the restaurant table, was converted into a string. Furthermore, for the purpose of Content-Based Filtering (CBF), the 'chain' column and the now-string 'tags' column were combined to create a new 'combined' feature, enriching the dataset for similarity calculation. Throughout this phase, missing values and inconsistencies were handled carefully to maintain data integrity and avoid potential bias in the recommendation process. Finally, the acquired and preprocessed data, including restaurant info, user ratings, and user preferences, was transformed into Pandas DataFrames for efficient manipulation and analysis in the system.

The system design outlines the architectural blueprint and operational flow of the mobile-based restaurant recommendation system, specifying its core components, user interactions, and the underlying data structures. The system architecture consists of a frontend, a backend, and a database, working together to deliver personalized restaurant recommendations. The mobile application, serving as the frontend, is developed using Expo and React Native, ensuring cross-platform compatibility and a responsive user interface. The backend services, which are responsible for data processing and recommendation logic, are built using Express and Flask.

The application's processes and user interactions are meticulously illustrated through various flowcharts. The key operational flows are described as follows:

1. Landing Page and Sign In/Sign Up Process: This flow (Fig. 1) begins by presenting users with the option to either sign up for a new account or sign in if they are existing users. For new sign-ups, user data is securely stored in the database, and the user is then directed to the home page. Existing users input their email and password, which are validated against the database; successful authentication leads them to the home page.
2. Home Page Flow: This flow (Fig. 2) dynamically checks for the presence of user ratings data in the database. If ratings data exists, the system displays hybrid-filtered restaurant recommendations; otherwise, an empty recommendation message is presented.
3. Hybrid Filtering Recommendation Process: This core process (Fig. 3) orchestrates the generation of personalized recommendations. It starts with the initial acquisition of comprehensive user data,

ratings, restaurant details, and menu information. Subsequently, data preprocessing and feature extraction are performed to prepare the data. In the Collaborative Filtering (CF) phase, the system creates a matrix between users and all restaurants/menus based on their ratings. User similarity is then calculated based on their given ratings. Concurrently, the Content-Based Filtering (CBF) phase calculates restaurant and menu similarity based on their content or features, such as name and category. The similarities derived from both CF and CBF are then combined, and the recommendations are further refined by applying user-defined preferences. Finally, the system sorts the combined items and retrieves the top 10 items as recommendations to be displayed to the user.

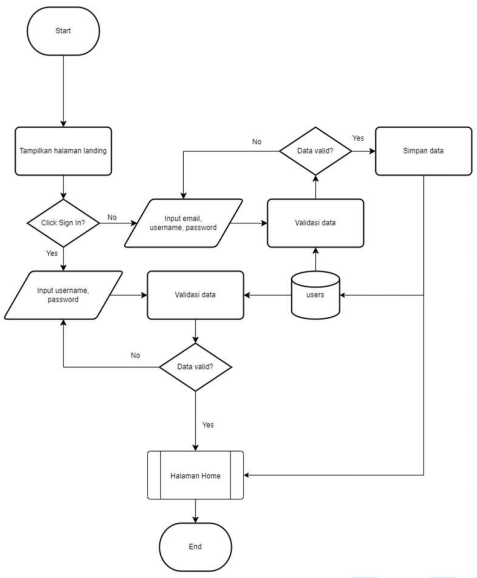

Fig. 1. Flowchart of Landing Page and Sign In/Sign Up Process

4. This flow (Fig. 4) facilitates user interaction with unrated restaurants and menus. It involves fetching data from the API, and any new restaurant or menu data not already in the database is added. Unrated restaurants and menus are presented to the user in a stacked card format. Users can interact with these cards by swiping left to dislike or right to like an item; these interactions update their preferences and ratings in the database.
5. Profile Page Flow: This flow (Fig. 5) allows users to manage their account and recommendation preferences. Users can modify their recommendation preferences, such as maximum price, minimum rating, and maximum distance, through interactive sliders. The system also provides options to reset all existing ratings, which deletes previously performed ratings , and to log out, returning the user to the landing page.

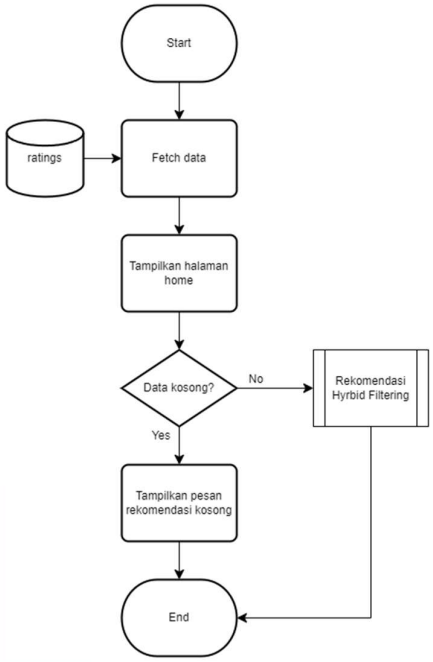

Fig. 2. Flowchart of Home Page

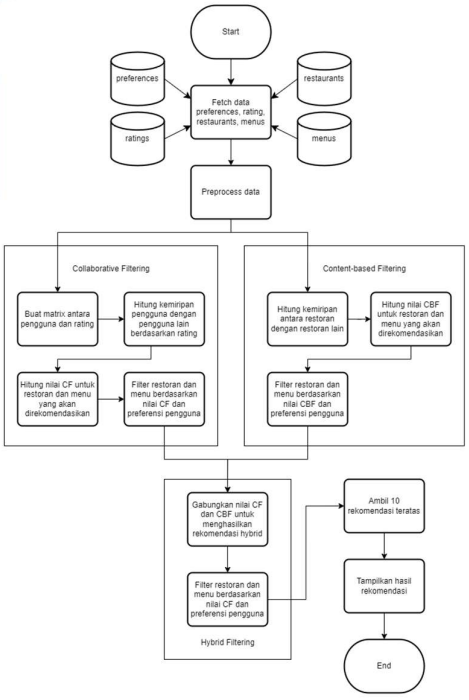

Fig. 3. Flowchart of Hybrid Filtering Recommendation

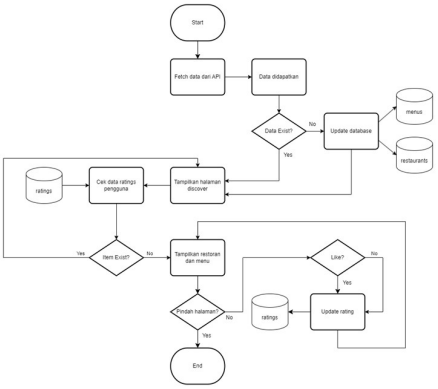

Fig. 4. Flowchart of Discover Page

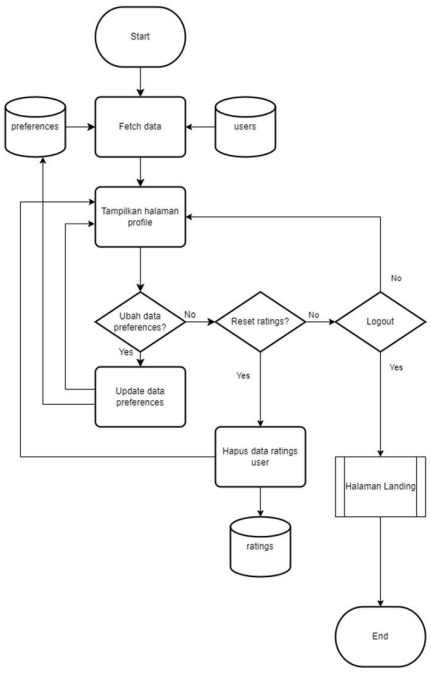

Fig. 5. Flowchart of Profile Page

The underlying Database Structure (Table 1) is meticulously designed to support the restaurant recommendation system's functionality. It comprises several key tables:

1. The users table stores user account information (id: UUID, created_at: timestamp, email: varchar, password: varchar).
2. The preferences table manages user-specific recommendation settings (id: INT, user_id: UUID FK, price_high: INT, min_rating: FLOAT, max_distance: INT, halal_only: BOOL).
3. The restaurants table holds detailed information about each restaurant (resto_id: varchar PK, name: varchar, latitude: float, longitude: float, rating: float [0-5], image_url: varchar, chain: varchar, halal: bool, tags: json).
4. The menus table stores specific menu item details (id: varchar PK, name: varchar, resto_id: varchar FK, image_url: varchar, price: int, description: text).
5. The ratings table records user feedback on restaurants and menus (id: INT PK, created_at: timestamp, user_id: UUID FK, resto_id: varchar FK, menu_id: varchar FK, is_liked: bool).

TABLE I. DATABASE STRUCTURE

Table	Column	Data Type
users	id	uuid
	created_at	timestamp
	email	varchar
	password	varchar
preferences	id	int
	user_id	uuid
	price_high	int
	min_rating	float
	max_distance	int
	halal_only	bool
restaurants	resto_id	varchar
	name	varchar
	latitude	float
	longitude	float
	rating	float
	image_url	varchar
	halal	bool
	chain	varchar
	tags	json
menus	id	Varchar
	name	varchar
	resto_id	varchar
	image_url	varchar
	price	int
	description	text
ratings	id	int
	created_at	timestamp
	user_id	uuid
	resto_id	varchar
	is_liked	bool

IV. RESULT AND IMPLEMENTATION

The User Interface Implementation provides the user-facing components of the mobile application, ensuring intuitive and engaging user experience.

1. The Landing Page (Fig. 6) serves as the initial screen upon application launch, prominently displaying the application's name, "FoodieMatch," alongside clear "Sign In" and "Sign Up" buttons.
2. The Sign In Page (Fig. 7) facilitates user login, prompting for email and password input. Upon successful authentication, users are directed to the home page.

3. For new users, the Sign Up Page (Fig. 7) allows for account registration by requiring email, password, and password confirmation. Valid new user data is then registered, and the user is automatically authenticated and navigated to the home page.

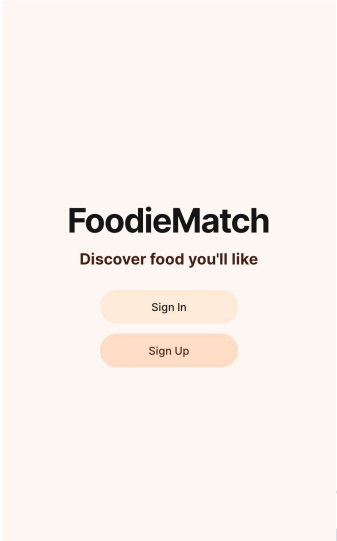

Fig. 6. Implementation Result of Landing Page

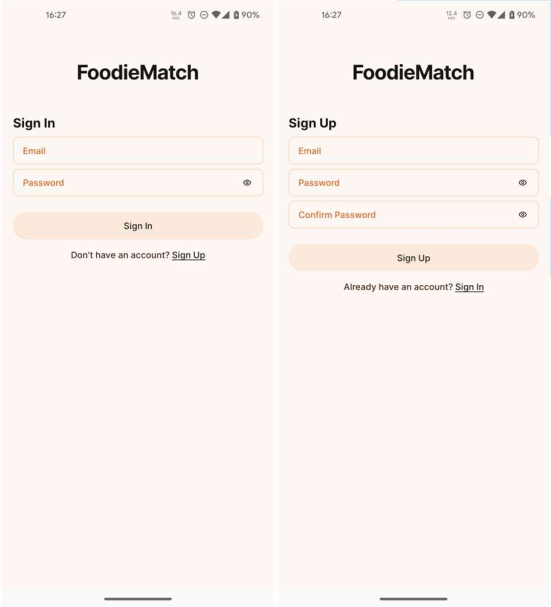

Fig. 7. Implementasion Result of Sign In and Sign Up Pages

4. The Home Page (Fig. 8) is designed to dynamically display restaurant recommendations or an empty message if no ratings data exists. It incorporates a navigation bar with tabs for "Home," "Discover," and "Profile" to facilitate seamless navigation. Users can also view a list of recently liked restaurants (Fig. 9) and access detailed information about specific restaurants and their menus by tapping on a recommended item.

5. The Discover Page (Fig. 10) presents unrated restaurants and menus in a stacked card format, allowing users to interact by swiping left to dislike or right to like an item. These interactions dynamically update their preferences and ratings in the database.

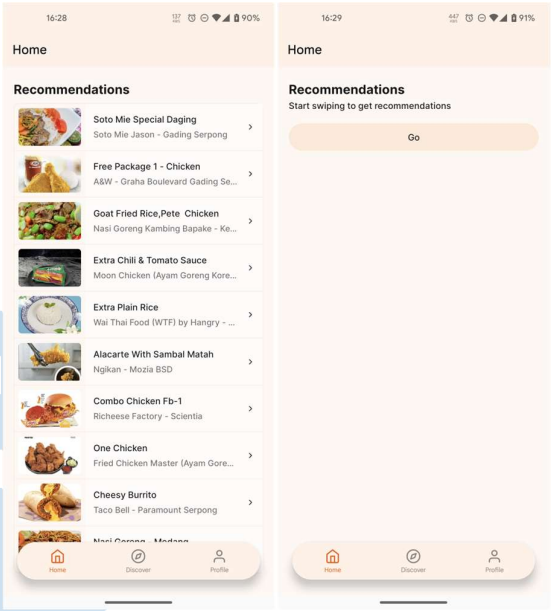

Fig. 8. Implementation Result of Home Page (Recommendations and Empty State)

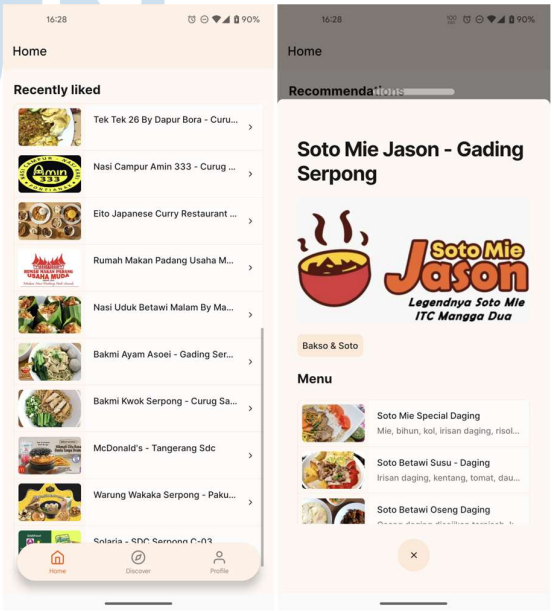

Fig. 9. Implementation Result of Home Page (Recently Liked and Restaurant Details)

Fig. 10. Implementation Result of Discover Page

6. The Profile Page (Fig. 11) provides users with access to their account and recommendation preferences. Here, users can modify parameters such as maximum price, minimum rating, and maximum distance through interactive sliders. It also includes options to reset all previous ratings, reverting preferences to default values, and a logout feature to return to the landing page.

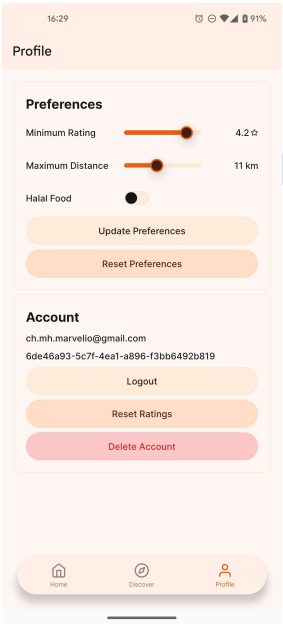

Fig. 11. Implementation Result of Profile Page

The Hybrid Recommendation System Implementation is a core component, demonstrating how the theoretical algorithms are translated into functional code that drives the application's recommendation engine. The Haversine Function (Fig. 12) is implemented to accurately calculate the great-circle distance between two geographical points using their longitude and latitude coordinates, which is essential for location-based filtering.

```
def haversine(lon1, lat1, lon2, lat2):
    lon1, lat1, lon2, lat2 = map(radians, [lon1, lat1, lon2, lat2])
    dlon = lon2 - lon1
    dlat = lat2 - lat1
    a = sin(dlat / 2) ** 2 + cos(lat1) * cos(lat2) * sin(dlon / 2) ** 2
    c = 2 * asin(sqrt(a))
    r = 6371
    return c * r
```

Fig. 12. Code Snippet for Haversine Function

For Collaborative Filtering (CF) Implementation, the system creates a user-restaurant matrix from the collected rating data. Each entry in this matrix indicates whether a user liked a particular restaurant. Subsequently, user similarity is computed using cosine similarity (Fig. 13), a crucial step in identifying users with similar tastes.

```
user_resto_matrix = ratings.pivot(index='user_id', columns='resto_id', values='is_liked').infer_objects(copy=False).fillna(0)
user_similarity = cosine_similarity(user_resto_matrix)
user_similarity = pd.DataFrame(user_similarity, index=user_resto_matrix.index, columns=user_resto_matrix.index)
```

Fig. 13. Code Snippet for Collaborative Filtering

In the Content-Based Filtering (CBF) Implementation, the TfidfVectorizer is employed to transform restaurant tags into a TF-IDF (Term Frequency-Inverse Document Frequency) feature matrix. Restaurant similarity is then calculated based on these processed tags using cosine similarity (Fig. 14), enabling the system to recommend items with similar content.

```
tfidf = TfidfVectorizer()
tfidf_matrix = tfidf.fit_transform(restaurants['combined'])
resto_similarity = cosine_similarity(tfidf_matrix, tfidf_matrix)
resto_similarity = pd.DataFrame(resto_similarity, index=restaurants['resto_id'], columns=restaurants['resto_id'])
```

Fig. 14. Code Snippet for Content-Based Filtering

User Preference Preparation involves defining and preparing user-specific preferences such as liked and disliked restaurants, minimum rating thresholds, maximum distance, and halal/non-halal food preferences (Fig. 15). This is crucial for tailoring recommendations. Filtering Based on Preferences then applies these user-defined criteria to filter the recommended items, ensuring that only relevant options are displayed. This step also ensures that restaurants previously liked or disliked by the user are excluded from the new recommendation list (Fig. 16).

Fig. 15. Code Snippet for User Preference Preparation

Fig. 16. Code Snippet for Preference Filtering

The Score Calculation (CF, CBF, Hybrid) phase (Fig. 17) involves a detailed process where CF scores are derived from user similarity and the ratings provided by other users. CBF scores are computed based on the content similarity between items and items previously liked by the user. Finally, these CF and CBF scores are combined to form a comprehensive hybrid score.

Fig. 17. Code Snippet for Collaborative and Content-Based Filtering Score Calculation

For Recommendation Generation, the system processes the items based on their calculated CF, CBF, and hybrid scores. These recommendations are then sorted by their highest score, and the number of recommendations displayed is limited as per the system's design. This process also includes handling the exclusion of duplicate restaurants or those from the same chain to ensure variety, returning the final recommendations in a structured dictionary format (Fig. 18).

Fig. 18. Code Snippet for Recommendation Results

The System Evaluation quantifies the effectiveness and user satisfaction of the developed restaurant recommendation system. The Evaluation Method employed was the End-User Computing Satisfaction (EUCS) [22], a widely accepted approach for measuring user satisfaction with information systems. EUCS is used to determine how well the system fulfills user requirements by comparing their expectations with the actual system performance. Evaluation Data Collection was conducted by distributing a questionnaire via Google Forms to 27 respondents who had used the application. The questionnaire comprised 10 questions divided into five key EUCS aspects, with responses collected using a 1-5 Likert scale, ranging from "Strongly Disagree" (1) to "Strongly Agree" (5). The specific questions were:

- Content: "Do you think this application provides content and information about restaurants and food that is appropriate?" (P1) and "Do you think the content in this application is clear and easy to understand?" (P2).
- Accuracy: "Do you think the recommendations given by this application are in accordance with your preferences?" (P3) and "Does the navigation in the application lead to the correct page?" (P4).
- Format: "Do you think the application display is attractive?" (P5) and "Do you think the application has an easy-to-understand structure and layout?" (P6).
- Ease of Use: "Do you think this application is easy to use?" (P7) and "Can you easily access all features in this application?" (P8).
- Timeliness: "Does the system display information with a fast response?" (P9) and "Does the application display the latest information?" (P10).

The Satisfaction Score Calculation for each question and overall criterion involved multiplying the frequency of responses by their respective scale weights, dividing by the total number of respondents, and then converting the result to a percentage. This calculation process is exemplified by the following equations:

- Content (P1):

$$\frac{(0\times1)+(0\times2)+(0\times3)+(9\times4)+(18\times5)}{27\times5}\times 100 \qquad (10)$$

- Content (P2):

$$\frac{(0\times1)+(0\times2)+(0\times3)+(6\times4)+(21\times5)}{27\times5}\times 100 \qquad (11)$$

- Percentage Content:

$$\frac{(93.3+95.5)}{2} = 94.4 \qquad (12)$$

Similar calculations were performed for Accuracy, Format, Ease of Use, and Timeliness. The detailed questionnaire results are presented in Table 2. The Results of this evaluation indicated an overall user satisfaction level of 93.9% for the recommendation system, derived from the average of all five criteria.

This high satisfaction rate across all aspects (Content: 94.4%, Accuracy: 93.6%, Format: 92.6%, Ease of Use: 97%, Timeliness: 92.2%) underscores the system's effectiveness in meeting user expectations.

TABLE II. EUCS QUESTIONAIRE RESULT

Question no.	Answer				
	1	2	3	4	5
1	0	0	0	9	18
2	0	0	0	6	21
3	0	0	0	16	11
4	0	0	0	1	26
5	0	0	2	11	14
6	0	0	0	5	22
7	0	0	0	4	23
8	0	0	0	4	23
9	0	0	1	11	15
10	0	0	0	8	19

V. CONCLUSION

This research successfully designed and developed a mobile-based restaurant recommendation system using a hybrid approach that combines Collaborative Filtering (CF) and Content-Based Filtering (CBF). The system was implemented as a mobile app utilizing Expo, React Native, Express, and Flask as its core frameworks and libraries. The evaluation of the developed system, which was conducted using the End-User Computing Satisfaction (EUCS) method, indicated that the system effectively met user expectations across all five key aspects: Content, Accuracy, Format, Ease of Use, and Timeliness. The overall user satisfaction for the system was remarkably high at 93.9%. This high satisfaction rate shows that the design and implementation of the restaurant recommendation system, which integrates hybrid collaborative and content-based filtering methods into a mobile app, was well received by its users.

REFERENCES

[1] A. Szymkowiak, B. Melović, M. Dabić, K. Jeganathan and G. S. Kundi, "Information technology and Gen Z: The role of teachers, the internet, and technology in the education of young people," *Technology in society,* vol. 65, p. 101565, 2021.

[2] R. Williams, "The technology and the society," in *Popular Fiction*, Routledge, 2023, p. 9–22.

[3] M. Ahmadi, S. N. Shahrokhi, M. Khavaninzadeh and J. Alipour, "Development of a mobile-based self-care application for patients with breast cancer-related lymphedema in Iran," *Applied Clinical Informatics,* vol. 13, p. 935–948, 2022.

[4] M. H. Azahari and F. A. H. Ali, "The development of an online food ordering system for JomMakan restaurant," *Applied Information Technology and Computer Science,* vol. 3, p. 369–376, 2022.

[5] Y. Bohang, "HUBUNGAN ANTARA MOTIVASI MEMBELI DAN PERILAKU PEMESANAN MAKANAN MELALUI APLIKASI ONLINE," 2020.

[6] S. G. Pillai, W. G. Kim, K. Haldorai and H.-S. Kim, "Online food delivery services and consumers' purchase intention: Integration of theory of planned behavior, theory of perceived risk, and the elaboration likelihood model," *International journal of hospitality management,* vol. 105, p. 103275, 2022.

[7] N. Thongsri, P. Warintarawej, S. Chotkaew and W. Saetang, "Implementation of a personalized food recommendation system based on collaborative filtering and knapsack method," *Int. J. Electr. Comput. Eng,* vol. 12, p. 630–638, 2022.

[8] U. Javed, K. Shaukat, I. A. Hameed, F. Iqbal, T. M. Alam and S. Luo, "A review of content-based and context-based recommendation systems," *International Journal of Emerging Technologies in Learning (iJET),* vol. 16, p. 274–306, 2021.

[9] R. Widayanti, M. H. R. Chakim, C. Lukita, U. Rahardja and N. Lutfiani, "Improving recommender systems using hybrid techniques of collaborative filtering and content-based filtering," *Journal of Applied Data Sciences,* vol. 4, p. 289–302, 2023.

[10] G. Parthasarathy and S. Sathiya Devi, "Hybrid recommendation system based on collaborative and content-based filtering," *Cybernetics and Systems,* vol. 54, p. 432–453, 2023.

[11] L. Li, Z. Zhang and S. Zhang, "Hybrid algorithm based on content and collaborative filtering in recommendation system optimization and simulation," *Scientific Programming,* vol. 2021, p. 7427409, 2021.

[12] C. Janiesch, P. Zschech and K. Heinrich, "Machine learning and deep learning," *Electronic markets,* vol. 31, p. 685–695, 2021.

[13] Q. Zhang, J. Lu and Y. Jin, "Artificial intelligence in recommender systems," *Complex & Intelligent Systems,* vol. 7, p. 439–457, 2021.

[14] S. M. Pande, P. K. Ramesh, A. Anmol, B. R. Aishwarya, K. Rohilla and K. Shaurya, "Crop recommender system using machine learning approach," in *2021 5th international conference on computing methodologies and communication (ICCMC)*, 2021.

[15] F. Ricci, L. Rokach and B. Shapira, "Recommender systems: Techniques, applications, and challenges," *Recommender systems handbook,* p. 1–35, 2021.

[16] H. Papadakis, A. Papagrigoriou, C. Panagiotakis, E. Kosmas and P. Fragopoulou, "Collaborative filtering recommender systems taxonomy," *Knowledge and Information Systems,* vol. 64, p. 35–74, 2022.

[17] L. Wu, X. He, X. Wang, K. Zhang and M. Wang, "A survey on accuracy-oriented neural recommendation: From collaborative filtering to information-rich recommendation," *IEEE Transactions on Knowledge and Data Engineering,* vol. 35, p. 4425–4445, 2022.

[18] K. Mao, J. Zhu, J. Wang, Q. Dai, Z. Dong, X. Xiao and X. He, "SimpleX: A simple and strong baseline for collaborative filtering," in *Proceedings of the 30th ACM international conference on information & knowledge management*, 2021.

[19] S. Mohmmad, R. Kanakam, R. Dadi, Shabana and S. N. Pasha, "Content and history based movie recommendation system," in *AIP Conference Proceedings*, 2022.

[20] K. Denistia, E. Shafaei-Bajestan and R. H. Baayen, "Exploring semantic differences between the Indonesian prefixes PE-and PEN-using a vector space model," *Corpus Linguistics and Linguistic Theory,* vol. 18, p. 573–598, 2022.

[21] A. Nazir, R. N. Mir and S. Qureshi, "Idea plagiarism detection with recurrent neural networks and vector space model," *International Journal of Intelligent Computing and Cybernetics,* vol. 14, p. 321–332, 2021.

[22] X. Cai, C. Huang, L. Xia and X. Ren, "LightGCL: Simple yet effective graph contrastive learning for recommendation," *arXiv preprint arXiv:2302.08191,* 2023.

[23] P. J. Schmid, "Dynamic mode decomposition and its variants," *Annual Review of Fluid Mechanics,* vol. 54, p. 225–254, 2022.

[24] R. M. Maisari, M. N. Alamsyah and L. Sunardi, "Analisis pengukuran tingkat kepuasan pengguna aplikasi OVO menggunakan metode End User Computing Satisfaction (EUCS)," *ESCAF,* p. 1405–1414, 2024.

[25] W. J. Doll and G. Torkzadeh, "The measurement of end-user computing satisfaction," *MIS quarterly,* p. 259–274, 1988.

UMN