Your Thesis Title

Your Name

March 28, 2024

Abstract

Contents

Chapter 1

Introduction

In recent years, the restaurant industry has experienced a significant shift towards digitization. The move towards web-based restaurant management applications signifies a critical step in this direction, offering a bridge between the traditional dining experience and modern technological conveniences. This final year project aims to design, develop, and evaluate a sophisticated web-based restaurant management application, leveraging contemporary web technologies to enrich the dining experience for customers and streamline restaurant operations.

The application at the core of this project is built upon a React.js frontend and a Django backend, handled by a SQLite database. The decision to use React.js stems from its efficiency in rendering dynamic user interfaces and facilitating an interactive web experience. Django offers comprehensive support for web development needs, including ORM (Object-Relational Mapping) for database interactions, REST Framework for API development, and built-in mechanisms for user authentication and authorization.

One of the application's standout features is its use of QR codes, enabling customers to access the menu and place orders directly from their smartphones. This not only enhances the dining experience by offering convenience and speed but also aligns with contemporary expectations for contactless services. The application further personalizes the customer experience through algorithms that analyze user preferences and order history, delivering tailored menu recommendations that cater to individual tastes and dietary needs. This implementation also solves the problem of splitting bills by introducing a session system to tables which allows customers to order under their own bill or share the bill with others easily all while sitting at the same table.

The backend architecture of the restaurant management system is designed to be efficient. The Django ORM facilitates interactions with the database, abstracting SQL queries into Python. Django REST Framework (DRF) plays a crucial role in this architecture, providing tools for building a RESTful API that the React frontend consumes. This API layer enables seamless communication between the frontend and backend, ensuring that data flows securely and efficiently across the application.

On the frontend, React.js's component-based architecture allows for the development of a modular and interactive user interface. The use of React components enables the reuse of UI elements across the application, ensuring consistency and reducing development time. State management is handled adeptly through React's context API, facilitating the tracking and updating of application state across components without prop drilling. This is crucial for managing user sessions, order details, and authentication states throughout the application. Additionally, the application employs React Router for client-side routing, enhancing the user experience with smooth transitions between views without the need for page reloads.

The database architecture is designed to cater to the needs of restaurant management. It includes models for the restaurant settings, menu items, orders, and customer profiles, among others. Each model is designed to capture data and relationships, enabling the application to offer features such as menu customization, order tracking, and personalized recommendations.

An integral part of the project is the implementation of a recommender algorithm within the UserProfile model. This algorithm is a cornerstone of the application's ability to offer personalized menu recommendations, to enhance the customer experience. By analyzing user preferences and order history, the algorithm identifies preferences, suggesting menu items that align with the user's taste profile. The decision to use a hybrid filtering approach, combining the strengths of collaborative and content-based filtering, underscores the project's commitment to leveraging advanced technologies for personalization.

For restaurant staff, the project includes features for order tracking, menu management, and sales analysis. These tools are designed to improve operational efficiency, enabling staff to manage orders more effectively, update the menu as needed, and analyze sales data to make informed decisions. The inclusion of a kitchen display system further streamlines kitchen operations, displaying orders in real-time and helping staff prioritize based on urgency.

In conclusion, this final year project aims to further improve current implementations and cover some gaps that have not been done before such as the recommender system and the session system.

Chapter 2

Literature Review

2.1 Evolution of Restaurant Management Systems

The evolution of restaurant management systems has seen a shift from traditional paper-based methods to digital systems, driven by the need for improved efficiency and customer experience. Traditional systems, such as paper-based menu cards and manual order taking, have limitations in terms of time consumption, manual errors, and customer dissatisfaction [1]. The adoption of digital systems, such as tablet food ordering and digital based ordering, has aimed to address these limitations by providing cost and time efficiency benefits for both management and customers [1]. Factors driving the adoption of digital restaurant management systems include the need for faster services, reduced dependency on waiters, which increases productivity [3]. Additionally, the use of technology to replace some of the jobs done by human beings and make machines do the work has been a driving factor in the adoption of online food ordering management systems [4].

The adoption of digital systems has also been influenced by the increasing trend of consumers adopting a more tech-savvy approach to conducting business transactions and leisure activities, leading to the need for restaurant owners to keep up with technological advancements to attract and retain a broader customer base [5]. Overall, the evolution of restaurant management systems from traditional to digital has been driven by the need for improved efficiency, reduced manual errors, and enhanced customer satisfaction, while also aligning with the changing technological landscape and consumer preferences.

2.2 Digital Ordering Systems

The implementation of digital ordering systems in restaurants has been shown to have a significant impact on customer experience and business outcomes. The use of QR codes for ordering has been designed to provide more advantages, including electronic payment of bills and entertainment facilities, and to reduce the time between ordering and delivering goods to customers [6]. The system allows customers to place food orders by scanning the QR code on the restaurant table, which then directs them to a digital version of the restaurant's menu, enabling them to place orders directly from their phones [2]. This automation of the ordering process has been found to reduce the time of order registry to delivery, improving customer satisfaction and business efficiency [6]. The use of digital menus in restaurants has been shown to provide several benefits, including the elimination of traditional ordering stages, more favourable choices for customers, and the ability to pay bills digitally, which prevents the pollution of money exchange and has a significant effect on protecting the environment due to the reduced use of paper [6]. The system also allows restaurant owners to have an insightful view of their business data, such as sales data, which can improve decision-making and forecasting demand using data analysis techniques [2].

Research has shown that the implementation of digital ordering systems has led to increased customer satisfaction, improved table turnover, and reduced labour and menu costs [5]. Customers have reported that the QR menu ordering systems provide convenience, value, and enjoyment, leading to increased customer attraction and satisfaction [5]. Furthermore, the system has been found to improve table turnover and reduce labour and menu costs, leading to improved business efficiency [5]. In conclusion, the implementation of QR code ordering systems in restaurants has had a positive impact on customer experience and business outcomes, leading to improved customer satisfaction, business efficiency, and insightful views of business data for restaurant owners.

2.3 Customer Behavior and Technology Acceptance Towards Digitalisation

The Unified Theory of Acceptance and Use of Technology (UTAUT) model has been applied to understand customer willingness to use QR code ordering systems in luxury restaurants in Xi'an, China [5]. The UTAUT model contains four independent variables: performance expectancy, effort expectancy, social influence, and facilitating conditions 1.2. Additionally, the UTAUT 2 model, an extended version of UTAUT, includes hedonic motivation, price value, trust, experience, and habit as independent variables, focusing more on the individual use of technology [5]. The study found that performance expectancy, effort expectancy, social influence, facilitating conditions, hedonic motivation, price value, and trust positively affect customer intention to use QR menu ordering [5]. However, effort expectancy and facilitating conditions negatively affect customer behavioural intention [5].

Factors influencing customer behaviour and acceptance include performance expectancy (the degree to which technology will benefit consumers performing certain activities), effort expectancy (the degree of ease associated with consumers' use of technology), social influence, facilitating conditions, hedonic motivation, price value, and trust [5]. These factors have been found to significantly influence customers' behavioural intention to use QR menu ordering systems [5].

Moreover, the study revealed that trust is an important variable affecting customers' behavioural intention to use the QR menu, especially when they are asked to provide confidential information such as transaction codes or personal details [5]. This highlights the significance of trust in influencing customer acceptance of QR code ordering systems.

2.4 Challenges and Solutions

The operational challenges faced by restaurants in implementing and managing web-based systems include difficulties in managing customer orders during peak hours, the need for efficient labour scheduling, and the management of customer reservations and waitlists [7].

Potential solutions to these challenges involve the implementation of a self-ordering System, which can automate the ordering process, provide real-time order tracking, and manage customer reservations efficiently [7].

Common technological barriers in implementing web-based systems include compatibility issues with existing systems, scalability concerns, and the need for secure data exchange between different systems [2].

These barriers can be overcome by using open-source technologies to maintain low costs, ensuring that the system is scalable to accommodate many users, and implementing secure data exchange protocols to protect customer and business data [2].

2.5 Current Implementations

There are some QR code ordering solutions reviewed, including solutions such as Toast [8], Square [9], Zuppler [10], TouchBistro [11], Bbot [12], Menufy [13], and Future Ordering [14]. These solutions provide a a lot of functionalities, such as online ordering, and loyalty programs, to specialized tools for creating and managing QR code-based menus. To summarise it, each solution presents features such as real-time menu updates, and data analytics. These solutions highlights the importance of selecting a system that aligns with a restaurant's specific operational needs, budget, and customer engagement goals, emphasizing the role of digital technology in transforming the traditional dining model into a more interactive, convenient, and streamlined process.

Future Ordering [14] is one of the most notable solutions in the UK as several big restaurant names such as KFC, Nandos and Burger King have implemented them. Future Ordering provides a digital ordering platform for food and beverage businesses, focusing on app, web, and kiosk channels. Their system supports various user journeys, including curbside, click'n'collect, and table ordering, and offers comprehensive management tools for digital channels. However, Future Ordering requires you to have their proprietary hardware which can be a barrier in terms of cost for smaller restaurants.

Solution	Online Ordering	Customer Analysis	Data Analytics	Personalised Experience
Toast [8]	Yes	No	Yes	No
Square [9]	Yes	Yes	Yes	No
Zuppler [10]	Yes/No	No	Yes	No
TouchBistro [11]	No	No	Yes	No
Bbot [12]	Yes	No	Yes	No
Menufy [13]	Yes	No	Yes	No
Future Ordering [14]	Yes	No	Yes	No

Table 2.1: Comparison of QR code ordering solutions

2.6 Research Gaps

While the adoption of digital ordering systems and the implementation of technologies like QR codes have greatly improved the efficiency of order-taking and the overall customer experience, there is a significant gap regarding personalized customer experiences through these digital systems. Most existing systems, as discussed, focus on streamlining operations, reducing errors, improving table turnover, and providing insights into business data. However, there's a scarcity of discussion around how these systems can leverage customer data to offer personalized dining experiences.

Personalization in restaurant management systems can significantly enhance customer satisfaction and loyalty by making customers feel valued and understood. By tracking previous orders and preferences, restaurants can offer tailored recommendations, which not only improves the customer experience but can save time during ordering as the recommendations are more likely to be accepted.

Another gap is the lack of solution for bill splitting. Bill splitting is a common issue in restaurants, especially when dining in groups. It can be a time-consuming and error-prone process for both customers and staff. A digital ordering system that can automate the bill splitting process can significantly improve the customer experience and reduce the workload on staff.

2.7 Conclusion

To conclude this literature review, the evolution of restaurant management systems has seen a shift from traditional paper-based methods to digital systems, driven by the need for improved efficiency and customer experience. The adoption of digital systems has been influenced by the increasing trend of consumers adopting a more tech-savvy approach to conducting business transactions and leisure activities, leading to the need for restaurant owners to keep up with technological advancements to attract and retain a broader customer base.

The implementation of digital ordering systems in restaurants has been shown to have a significant impact on customer experience and business outcomes, leading to improved customer satisfaction, business efficiency, and insightful views of business data for restaurant owners. The Unified Theory of Acceptance and Use of Technology (UTAUT) model has been applied to understand customer willingness to use QR code ordering systems in luxury restaurants in Xi'an, China [5]. Factors influencing customer behaviour and acceptance include performance expectancy, effort expectancy, social influence, facilitating conditions, hedonic motivation, price value, and trust.

These factors have been found to significantly influence customers' behavioural intention to use QR menu ordering systems.

The operational challenges faced by restaurants in implementing and managing web-based systems include difficulties in managing customer orders during peak hours, the need for efficient labour scheduling, and the management of customer reservations and waitlists. Common technological barriers in implementing web-based systems include compatibility issues with existing systems, scalability concerns, and the need for secure data exchange between different systems.

There are some QR code ordering solutions reviewed, including solutions such as Toast, Square, Zuppler, TouchBistro, Bbot, Menufy, and Future Ordering. These solutions provide a lot of functionalities, such as online ordering, and loyalty programs, to specialized tools for creating and managing QR code-based menus.

However, there is a significant gap regarding personalized customer experiences through these digital systems and the lack of solution for bill splitting. Personalization in restaurant management systems can significantly enhance customer satisfaction and loyalty by making customers feel valued and understood. By tracking previous orders and preferences, restaurants can offer tailored recommendations, which not only improves the customer experience but can save time during ordering as the recommendations are more likely to be accepted. A digital ordering system that can automate the bill splitting process can significantly improve the customer experience and reduce the workload on staff. This research aims to address these gaps by proposing a digital ordering system that leverages customer data to offer personalized dining experiences and automate the bill splitting process. The next chapter will discuss the research methodology used to achieve this goal.

Chapter 3

Methodology

3.1 Empirical Exploration

This section describes the preparatory empirical work performed for this project. It aims to define the scope of the final solution by presenting a preliminary discussion with restaurant owners and staff about their current practices and needs. The goal is to understand the current state of the art in the restaurant industry and to identify the main challenges and opportunities for the development of a new system. This section also presents the results of a preliminary survey conducted with potential users of the system, which aims to confirm the research gaps and the interest of new technology.

Interviews with Restaurant Owners and Staff

The first step in the empirical exploration was to conduct interviews with restaurant owners and staff. The goal was to understand the current state of the art in the restaurant industry and to identify the main challenges and opportunities for the development of a new system. The interviews were conducted with the help of a semi-structured questionnaire, which was designed to guide the conversation and ensure that all relevant topics were covered. The questionnaire was divided into three main sections: the current practices and challenges of the restaurant, the potential benefits and opportunities of a new system, and the main requirements and expectations for the system. The interviews were conducted with a total of 2 restaurant owners, one is the owner of a small restaurant in Portugal and the other is the owner of a multi-branch restaurant in Malaysia, and 6 restaurant staffs.

Interview with Restaurant Owners

The interviews with the restaurant owners aimed to identify their perspective of a restaurant owner to understand the current practices and challenges of the restaurant, the potential benefits and opportunities of a new system, and the main requirements and expectations for the system. Some of the questions that were asked during the interviews include:

- What are the main challenges that you face in managing your restaurant?
- What are the main benefits that you expect from a new system?
- What are the main requirements and expectations that you have for the system?
- What are their current practices in managing the restaurant?
- What are the other opportunities that you think a new system can bring to your restaurant?
- How do they deal with the bill splitting problem?
- What do they think about having a system that would be able to help them with bill splitting?
- What do they think about having personalised digital menus for their customers?
- What do they think about my idea of the system?

Interview with Restaurant Staff

Similar questions were asked to the restaurant staffs, but the questions were tailored to their roles and responsibilities in the restaurant. The interviews aimed to identify the main challenges and opportunities that the staff face in their daily work, and to understand their main requirements and expectations for the system. Some of the questions that were asked during the interviews include:

- What are the main challenges that you face in your daily work?
- What are the main benefits that you expect from a new system?
- What are the main requirements and expectations that you have for the system?

- What are their current practices in managing the restaurant?
- What are the other opportunities that you think a new system can bring to your restaurant?
- How do they deal with group orders and bill splitting?
- What do they think about having a system that would be able to help them with bill splitting?
- What do they think about having personalised digital menus for their customers?
- What do they think about my idea of the system?

Surveys With Customers

This survey aimed to understand dining habits, preferences, and challenges faced by people when eating out, particularly focusing on the use of QR code ordering systems, group ordering dynamics, bill splitting difficulties, and interest in technological solutions to these issues. The respondents varied in their frequency of dining out, ranging from daily to a few times a month. The survey was conducted with a total of 43 respondents. The survey questions are as follows:

- How often do you dine out at restaurants?
- Have you ever ordered food in a restaurant using a QR code?
- If yes, how would you rate the experience? (Only answer if you selected ”Yes” above.)
- When dining with others, how do you prefer to place your order?
- What challenges have you faced when ordering food in restaurants?
- How do you currently split bills when dining out with friends or family?
- How challenging is it to split bills when ordering as a group?
- How interested would you be in an app that allows you to order individually or as a group and easily split the bill directly through the app?
- Would you be interested in a feature that recommends menu items based on your past orders?
- Do wish that restaurants have a personalized menu just for your preferences?

A large portion of respondents has utilized QR code ordering systems, with experiences rating from ”Very Satisfactory” to ”Unsatisfactory”, mostly leaning towards Satisfactory. Most prefer to place orders as a group, either opting for one bill for everyone or paying individually, yet the common practice is to calculate everyone’s share after a single person pays the bill. Challenges identified include difficulty in getting the server’s attention, limited time to decide on the menu, specifying dietary restrictions, and the inconvenience of splitting bills.

Regarding bill splitting, a large number of respondents found it somewhat to very challenging, indicating a potential area for improvement in dining experiences. On top of that, high interest was expressed in an solution that covers individual or group ordering and simplifies bill splitting.

Furthermore, the idea of personalised menu recommendations based on past orders received strong support, all respondents were interested for personalized menus tailored to their preferences.

To conclude this survey, the results suggest that there is a demand for a system that can help with group ordering and bill splitting, and that can provide personalized digital menus to customers. The survey also aligns with the findings from the literature review that there is a growing trend in the use of digital ordering systems in restaurants, and that there is still room for improvement.

3.2 Requirements

This section presents the requirements for the system, which were identified based on the empirical exploration and the literature review. The requirements are divided into functional and non-functional requirements.

Functional Requirements

The functional requirements describe the main features and capabilities of the system. They were identified based on the interviews with restaurant owners and staff, and the survey with customers. The main functional requirements for the system are as follows:

- **QR Code Ordering:** The system should allow customers to place orders by scanning a QR code on their table, which will take them to a digital menu.
- **Group Ordering:** Use sessions to allow customers to place orders as a group, and to split the bill easily. Additional customers can join the existing sessions or create a new session.
- **Bill Splitting:** The adminstrator would be able to see all session orders on a table and would be able to split the bill automatically.
- **Personalised Digital Menus:** The system should provide customers with personalised digital menus, which recommend menu items based on their past orders and preferences.
- **Shopping Cart:** The system should allow customers to add items to a shopping cart, and to review and modify their orders before submitting them. The order is submitted to the server containing the table number and the user's session.
- **Table Management:** The adminstrator would be able to add and remove tables, and print QR codes for each table.
- **Order Management:** The adminstrator would be able to view all orders from all the tables and track status of the orders. The system would show the time since the order was placed.
- **Kitchen Display System:** The system should provide a kitchen display system to help kitchen staff manage orders and track their status. Have separate views for different sections of the kitchen. Example, Bar and Kitchen.
- **Floor Staff Management:** The system should provide a floor staff orders that needs to be served and the table numbers of orders.
- **Menu Management:** The system should allow restaurant staff to manage the digital menu, add new items, update prices, and remove items as needed.
- **Customer Management:** The system should allow restaurant staff to manage customer accounts, view customer preferences and past orders, and update customer information as needed.
- **Settings:** The system should allow restaurant staff to configure settings such as tax rates, name of restaurant, restaurant logo, menu theme.
- **Analytics:** The system should provide restaurant owners with analytics and reports on customer orders, preferences, and trends, to help them make data-driven decisions.

Non-Functional Requirements

The non-functional requirements describe the quality attributes of the system, such as performance, security, and usability.

- **Performance:** The system should be able to handle a large number of users and orders, and should be able to process orders quickly and efficiently.
- **Security:** The system should be secure, and should protect customer data and payment information. It should also be able to prevent unauthorized access to the system. There should be a clear separation between the customer and the restaurant staff.
- **Usability:** The system should be easy to use, and should provide a user-friendly interface for both customers and restaurant staff.

- **Reliability:** The system should be reliable, and should be able to handle errors and failures gracefully. It should also be able to recover from failures quickly and without data loss.
- **Scalability:** The system should be able to scale to accommodate a growing number of users and orders, and should be able to handle increased demand without a significant decrease in performance.
- **Accessibility:** The system should be accessible to people with disabilities, and should provide support for assistive technologies such as screen readers and voice recognition software.
- **Compatibility:** The system should be compatible with a wide range of devices and operating systems, and should be able to run on both desktop and mobile platforms.
- **Maintainability:** The system should be easy to maintain, and should be able to accommodate changes and updates without a significant impact on the system's performance or reliability.
- **Cost:** The system should be cost-effective, and should provide good value for money. It should also be able to reduce costs for restaurant owners and staff, and should be able to increase revenue and customer satisfaction.
- **Ease of Deployment:** The system should be easy to deploy, and should be able to run on a wide range of hardware and software platforms. It should also be able to integrate with existing systems and technologies.
- **Environmental:** The system should be environmentally friendly, and should be able to reduce waste and energy consumption. It should also be able to promote sustainable practices and support environmental conservation efforts.

3.3 Prototyping

This section presents the prototyping process for the system, which was performed to validate the requirements and to explore the main features and capabilities of the system.

Chapter 4

Implementation

4.1 Overview

This chapter outlines the systematic methodology employed to design, develop, and evaluate a web-based restaurant management application. The application leverages modern web technologies, including a React.js frontend [15], to enhance the dining experience by enabling customers to place orders using QR codes, receive personalized menu recommendations, and choose flexible bill-sharing options. The development process is guided by user-centered design principles, aiming to improve customer satisfaction and operational efficiency for restaurants.

4.2 System Design

The web application is built using React.js [15]frontend with Django backend [16], with a SQLite database [17]. The frontend is responsible for rendering the user interface and handling user interactions, while the backend manages the business logic and data storage. The frontend and backend communicate via Djano RESTful APIs [15]using Axios [18] from React. The application can hosted on a cloud platform, such as AWS [23] or Heroku [24] or Railway [25].

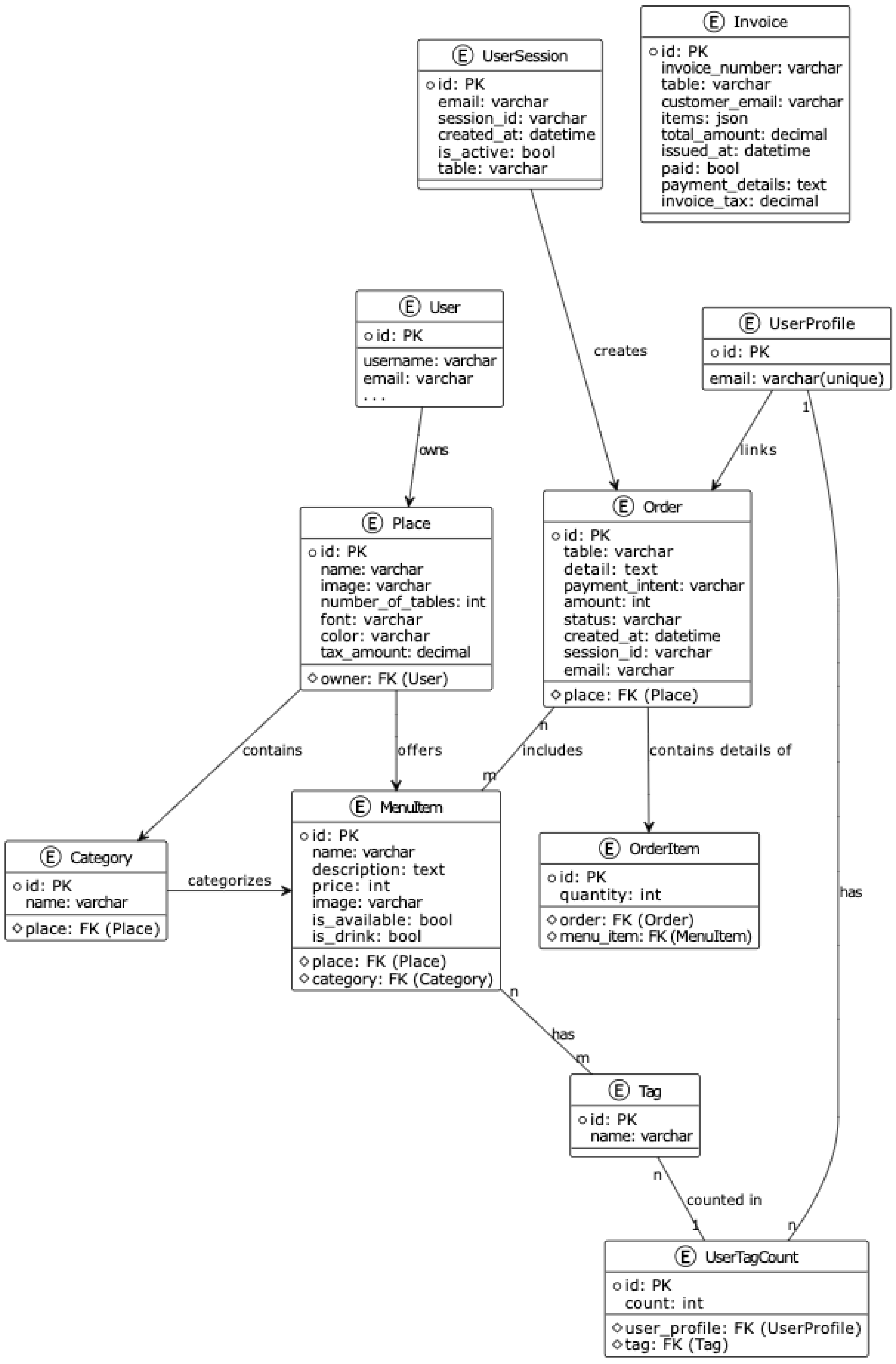
4.2.1 Backend Architecture

The backend architecture of the restaurant management system leverages Django's core components. Django was chosen as the backend framework due to its built in features, including the Django ORM for database management, Django REST Framework for API development allows for further scalability, and built-in user authentication and authorization mechanisms. Django models define the database schema, representing core entities like Place, Category, MenuItem, and Order. Views handle the application's business logic, receiving and responding to web requests, interacting with models, and returning JSON responses for use by a React frontend [18]. Django's URL routing directs incoming requests to the appropriate view. The Django REST Framework (DRF) is used to construct a RESTful API for the React interface, providing serializers for complex data to JSON conversion and viewsets for common API operations [20]. Authentication and authorization are handled through Django's built-in system and DRF's mechanisms, such as token authentication. Finally, a database like SQLite is used to persist the system's data [17].

4.2.2 Frontend Architecture

The frontend architecture of the application leverages React.js to build its user interface, utilizing React components for crafting reusable UI elements like buttons, forms, and menus [15]. State management is handled through React's state and context API, which manage crucial application states, including user authentication, order details, and session information [19]. For navigating between different views without refreshing the page, React Router is employed for client-side routing [22]. The integration with the backend is facilitated through Axios, which is used for making HTTP requests to the Django backend, thereby enabling data exchange between the frontend and the server. The user interface is designed with responsiveness and user-friendliness in mind, aiming to provide an intuitive and visually appealing experience for users. Additionally, Bulma SCSS components are utilized alongside React Components to ensure responsive and uniform styling across the application [22].

4.3 Database Architecture



4.3.1 Place

Represents a restaurant or place that uses the app. It includes details like the owner (linked to the Django `User` model), name, image, number of tables, and aesthetic customizations (font and color). A new field, `tax_amount`, has been added to manage tax calculations.

4.3.2 Category

Organizes menu items into categories (e.g., appetizers, entrees, desserts) for each place. This helps in structuring the menu for easier navigation by the customers.

4.3.3 MenuItem

Details about each menu item, including the place it belongs to, its category, name, description, price, and availability. It also specifies whether the item is a drink and allows for tagging (e.g., vegan, spicy) through a many-to-many relationship with the `Tag` model.

4.3.4 Tag

Used to label menu items with specific attributes, enhancing the ability to offer personalized recommendations based on customer preferences. Tags are used to describe attributes of food and based on (research) it has been shown that using attributes of food is a good way to recommend food items to customers.

4.3.5 UserSession

Tracks active sessions by customers, including their email, session ID, and table number. It supports functionalities like order tracking and session management.

4.3.6 Order

Records details of customer orders, including the items ordered (linked through `OrderItem`), the table and place, payment information, and status. It also tracks the session and customer email.

4.3.7 OrderItem

A through model for the many-to-many relationship between `Order` and `MenuItem`, capturing the quantity of each menu item ordered.

4.3.8 Invoice

Manages billing information, including the items ordered, total amount, and payment status. It introduces the `items` field as a JSON structure to store ordered items, allowing for flexible representation of order details.

4.3.9 UserProfile

Represents a user's profile with a unique email. It is linked to `Tag` through `UserTagCount` to track preference metrics.

4.3.10 UserTagCount

Tracks the count of tags associated with a user's orders, facilitating the personalization engine to recommend items based on preferred tags.

4.4 Digital Menu Implementation

QR codes placed on each table serve as the entry point for customers to access the menu and place orders. Upon scanning the QR code with a smartphone, customers are directed to a web interface where they can browse the menu, make selections, and specify their order preferences.

4.4.1 Menu Display

The menu is organized into categories, with each item displaying its name, description, price, and availability. The user interface is designed to be visually appealing and easy to navigate, with a responsive layout that adapts to different screen sizes.

4.4.2 Order Placement

Customers can add items to their order by clicking on the menu items and specifying the quantity. The system tracks the items ordered and the customer's session, allowing for order tracking and billing.

4.4.3 Personalized Recommendations

The application uses the customer's order history and preferences to recommend menu items that match their taste. This is achieved through methods in the `UserProfile` model, which calculate preferences based on past orders and tag popularity. The algorithm takes into account the most ordered tags and preferred price range to recommend items that align with the customer's preferences.

4.5 Recommender Algorithm

The recommendation algorithm embedded within the `UserProfile` model of our restaurant management application is designed to offer personalized menu suggestions to users. It analyzes users' past orders, preferences, and price sensitivity to recommend items that align with their tastes.

4.5.1 Possible Implementations

The recommendation algorithm can be implemented in several ways. Here are some of the considered directions.

Collaborative Filtering

Collaborative filtering recommends items based on the preferences of similar users. It identifies users with similar tastes and suggests items that they have liked. The initial step involves gathering data on customer interactions with the menu. For example, this includes which items customers order, ratings if available, and possibly the context of their orders (time of day, group size, etc). This data is used to identify similarities among users. Then, calculate the similarity between users based on their ordering patterns. Similarity metrics such as Pearson correlation or cosine similarity are used to identify users with similar tastes. The algorithm uses techniques like Singular Value Decomposition (SVD) or Alternating Least Squares (ALS) to analyse datasets. These methods help in predicting a user's interest in an item based on the latent factors extracted from the user-item interaction matrix. Based on the similarity metrics or matrix factorization, generate a list of recommended menu items for each user. This method is effective for discovering new items based on the preferences of a user's peers [26] [27]. However, it requires a large amount of data to generate accurate recommendations.

Content-Based Filtering

Content-based filtering uses item features to recommend other items similar to what the user likes, based on their previous actions or explicit feedback. In the context of a restaurant application, CBF can personalize menu recommendations by considering the characteristics of menu items (e.g., ingredients, cuisine type, dietary tags) and matching them with a user's past preferences. The algorithm identifies relevant features of menu items that can influence user preferences, such as type of cuisine, ingredients, dietary restrictions (vegan, gluten-free), spiciness level, and more. These attributes will have to be defined when creating the menu. Use these features to create a profile for each menu item in the database. The algorithm can use a similarity metric (e.g., cosine similarity, Euclidean distance) to compare the feature vectors of the user's profile with those of each menu item. Calculate a similarity score that indicates how closely an item matches the user's preferences. Rank the menu items based on the similarity score and recommend the top items to the user. This method is effective for providing personalized recommendations based on the attributes of menu items and user preferences [29].

Hybrid Filtering

Hybrid Filtering combines collaborative and content-based filtering to leverage the strengths of both approaches. By combining the two methods, the system can provide more accurate and diverse recommendations, addressing the limitations of individual techniques [30]. This method is used by web services such as Netflix and Amazon to provide personalized recommendations to users [33].

Deep Learning Models

Neural networks can be used to learn complex patterns in user-item interactions, enabling more accurate and personalized recommendations. These models can capture non-linear relationships between users and items, improving the quality of recommendations. Deep Learning models would need to preprocess the data to convert categorical attributes into numerical or vector representations suitable for neural network input. There are several deep learning architectures that can be used as the recommender algorithm such as Convolutional Neural Networks (CNNs) for analyzing item features or Recurrent Neural Networks (RNNs) and Long Short-Term Memory (LSTM) networks for sequential order data. These models can be trained using backpropagation and optimization techniques like stochastic gradient descent. The model can be evaluated using metrics like Mean Squared Error (MSE) or Mean Absolute Error (MAE) to measure the accuracy of the predictions. This method is effective for capturing complex patterns in user-item interactions and providing accurate recommendations. However, they require large amounts of data and computational resources [31].

Stable Matching Algorithms

Stable matching algorithms can be used to match users with items based on their preferences and constraints. By formulating the recommendation problem as a stable matching problem, the system can generate optimal matches between users and items, ensuring fairness and efficiency [32]. Applied to a restaurant usage, this can optimize the recommendation of menu items to customers by aligning diner preferences with menu offerings. The algorithm would need to collect data on user preferences through previous orders, ratings, and explicit feedback. Preferences can include favorite cuisine types, dietary restrictions, and flavor profiles. Then, generate preference lists for both users and menu items. For users, this list ranks menu items based on their historical preferences and feedback. For menu items, create a hypothetical preference list ranking users based on the likelihood of enjoying that item, inferred from user profile similarities and item attributes. Use a stable matching algorithm, like Gale-Shapley, to pair users with menu items. The algorithm iteratively proposes menu items to users based on the preference lists until stable matches are made, where no unmatched pair would both prefer each other over their current matches. After users are recommended items and make selections, collect feedback to refine their preference profiles. This feedback is essential to adjust future recommendations and improve the matching process. However, the need for ongoing user feedback to refine recommendations might increase user interaction requirements, potentially affecting the user experience.

Conclusion

The approach that has been chosen for this implementation is Hybrid Filtering. The system combines collaborative and content-based filtering to provide more accurate and diverse recommendations. While not using the traditional user-to-user collaborative filtering, the system uses the user's past orders and preferences to recommend items that align with their tastes. This way the customer does not need to provide a preference when they first make their order and the recommender system can instead learn from their past orders. This method leverages the strengths of both collaborative and content-based filtering. The algorithm can use the attributes of menu items and user preferences based of their past orders to recommend items that align with the user's taste profile.

4.5.2 User Preferences and Order History Analysis

The algorithm analyses the user's previous orders to deduce their preferences. The algorithm calculates the average price of the items ordered to establish a preferred price range and identifies the most ordered tags to understand the user's food preferences. This information is then used to score and recommend menu items that align with the user's taste.

4.5.3 Tag-Based Recommendations

The algorithm scores menu items based on the user's preferred tags and price sensitivity. The tags represent the attributes of the items, such as vegetarian, healthy, or spicy. By matching the tags of the menu items with the user's preferred tags, the algorithm can recommend items that are likely to appeal to the user. The reason behind the choice to use tags

4.5.4 Price Sensitivity

The algorithm calculates the user's preferred price range based on their past orders. By analyzing the average price of the items ordered, the system establishes a price range that the user is comfortable with. This information is used to filter out menu items that fall outside the user's price range, ensuring that the recommendations are tailored to the user's budget. The relationship between price and food choice has been highlighted. Research papers [34] and [35] has indicated that a large percentage of the population prioritizes price as the most determining factor in food choice, often opting for cheaper food regardless of its health or environmental impact.

4.5.5 Personalized Recommendations

The recommendation algorithm scores menu items based on the user's preferred tags and price sensitivity. By matching the tags of the menu items with the user's preferred tags and boosting the score if the price of the food item fits within the user's preferred price range, the algorithm can recommend items that are likely to appeal to the user. The algorithm ranks the menu items based on the score, providing personalized recommendations that align with the user's taste profile. It does the same for both food and drink items

4.5.6 Combining Recommendations for Combo Meals

The recommendation algorithm creates all possible combinations of recommended food and drink items to suggest combo meals to the user. By combining the top recommended food and drink items. This would save the customer time when ordering and would also increase the chances of the customer ordering more items. This is a common strategy used by restaurants to increase sales.

4.5.7 Example Usage

Step 1: Determining the Preferred Price Range

Assume the user has ordered the following items:

- Pizza: $12 (Quantity: 2)
- Burger: $9 (Quantity: 1)
- Salad: $7 (Quantity: 3)

Calculation

```
Total spent on Pizza = $12 * 2 = $24
Total spent on Burger = $9 * 1 = $9
Total spent on Salad = $7 * 3 = $21

Total spent = $24 + $9 + $21 = $54
Total quantity of items ordered = 2 + 1 + 3 = 6

Average price per item = Total spent / Total quantity = $54 / 6 = $9
```

Preferred price range is calculated as ±20 percent of the average price:

```
Lower bound = $9 * 0.8 = $7.20
Upper bound = $9 * 1.2 = $10.80
```

Step 2: Identifying Most Ordered Tags

Given tags for each item:

- Pizza: ["Italian", "Cheese"]
- Burger: ["American", "Beef"]
- Salad: ["Vegetarian", "Healthy"]

The most ordered tags based on quantity are "Vegetarian" and "Healthy".

Step 3: Scoring and Recommending Menu Items

New menu items to recommend:

- Spaghetti: $8 ["Italian", "Vegetarian"]
- Chicken Wrap: $10 ["Healthy", "Chicken"]
- Fish Tacos: $11 ["Seafood"]
- Veggie Burger: $9 ["Vegetarian", "American"]

Scoring

Spaghetti scores high for matching "Vegetarian" and being within the price range. Chicken Wrap scores for "Healthy" and being within the price range. Fish Tacos score lower due to being outside the price range and no matching tags. Veggie Burger scores for "Vegetarian" and being within the price range.

Results

Based on the scoring, the recommended items in order are:

1. Spaghetti
2. Chicken Wrap
3. Veggie Burger

4.6 Flexible Bill Sharing

This section outlines the systematic steps taken to design, implement, and evaluate this bill splitting implementation.

4.6.1 Design Considerations

A primary focus was to ensure an intuitive and hassle-free interaction with the bill-sharing feature, allowing customers to easily choose between individual payments or sharing the bill with others. The feature needed to accurately calculate and split the bill according to the specific orders and preferences of the diners, providing flexibility in how the bill could be shared. The system has to manage individual and group ordering, allowing customers to order individually or as a group, while ensuring accurate tracking of orders and payments.

Possible Implementations

There are several ways to implement the bill-sharing feature, each with its own advantages and considerations. Here are some of the possible implementations. One method is to use a session system that allows customers to join a session created by another diner at the same table. This feature is particularly useful for groups who want to combine their orders into a single bill, allowing easy payment splitting without the need for manual calculations or multiple transactions. The system can also provide an option for customers to create individual ordering sessions by entering their email address. This process ensures that each diner's selections are tracked separately, even though they are physically sitting at the same table. At the end of the meal, the group can decide whether to pay separately or together. If they opt for individual payments, each diner pays only for their ordered items. For a shared bill, the total cost is evenly divided among the participants in the joined session, or the payment can be split according to the specific items each person ordered, depending on the application's capabilities and settings. Another possible implementation it to generate new QR codes for each session that is created. This way the system can track the orders and payments. However, this method would require printing a lot of QR codes where as the session system would only require one QR code per table. Hence, the session system is the most sustainable solution over time and it would allow for customer profiling.

4.6.2 Implementation

- **Order Placement via individual Sessions**: When customers scan the QR code at their table, they have the option to create an individual ordering session by entering their email address. This process ensures that each diner's selections are tracked separately, even though they are physically sitting at the same table.
- **Joining Sessions for Shared Billing**: The application provides an option for customers to join a session created by another diner at the same table. This feature is particularly useful for groups who want to combine their orders into a single bill, allowing easy payment splitting without the need for manual calculations or multiple transactions.
- **Separate Payments or Single Payment**: At the end of the meal, the group can decide whether to pay separately or together. If they opt for individual payments, each diner pays only for their ordered items. For a shared bill, the total cost is evenly divided among the

participants in the joined session, or the payment can be split according to the specific items each person ordered, depending on the application's capabilities and settings.

4.7 Admin Panel

4.7.1 Features

- **Order Tracking**: Provides real-time tracking of customer orders, allowing staff to monitor and manage orders efficiently. This feature allows the adminstrator to keep track of which tables are occupied, the time since the last order, and the status of each order.
- **Billing and Invoice Management**: The adminstrator can manage billing and invoices, for each table and would be able to see all the separate sessions that are on that table. This feature also allows the adminstrator to view and manage the billing and invoices for each table, including the ability to split bills and track payments. All paid and unpaid invoices are stored, allowing the adminstrator to view and manage them as needed. The invoices are designed to work with carbon printers as well.
- **Sales Analysis**: The adminstrator can view sales data such as daily, weekly and monthly sales. The adminstrator can also view the most popular items and the least popular items. This feature also displays the peak hours of the day which along side with the popular items can be used to make decisions on inventory.
- **Global Settings**: The adminstrator can manage global settings such as tax rates, menu themes and fonts. They can also change the name of the restaurant and the logo.

4.8 Kitchen Display System

4.8.1 Features

- **Real-time Order Display**: The kitchen display system provides a real-time display of incoming orders, allowing kitchen staff to prepare and manage orders. This feature allows the kitchen staff to view the orders as they come in, and mark them as in progress or completed.
- **Separate Displays**: The system can display orders for different stations on separate screens, allowing staff to focus on their specific tasks. This feature allows the kitchen staff to view the orders that are relevant to their station, and manage them accordingly. Such as the bar which would only display drink orders.
- **Time Warnings:** When the order time exceeds 15 minutes, the order would turn yellow and when it exceeds 25 minutes, the order would turn red. This feature allows the kitchen staff to prioritize orders based on their urgency.

4.9 Floor Staff Display

4.9.1 Features

- **Order Tracking**: The orders that have been prepared by the kitchen would be displayed.
- **Table Numbers**: The table numbers of the orders would be displayed.
- **Update Status**: Once the order has been served, the floor staff would be able to update the status of the order as served.

4.10 Menu Management

4.10.1 Features

- **Add New Items**: The system should allow restaurant staff to add new items to the menu, including specifying the name, description, price, category, tags which will be used for the recommendation algorithm, and availability of the item. This feature allows the restaurant staff to add new items to the menu as needed, and specify the details of each item.
- **Update Items**: The system should allow restaurant staff to update existing items on the menu, including changing the name, description, price, category, tags, and availability of the item. This feature allows the restaurant staff to update the details of existing items on the menu as needed.
- **Remove Items**: The system should allow restaurant staff to remove items from the menu that are no longer available. This feature allows the restaurant staff to remove items from the menu that are no longer available or have been discontinued.
- **Create Categories**: The system should allow restaurant staff to create new categories for the menu, and assign items to specific categories. This feature allows the restaurant staff to organize the menu into different categories, making it easier for customers to navigate and find items.
- **Manage Tags**: The system should allow restaurant staff to manage tags for each item, specifying the attributes of the item such as vegan, spicy, or gluten-free. This feature allows the restaurant staff to tag items with specific attributes, which will be used for the recommendation algorithm.

Chapter 5

Project Management

Chapter 6

Evaluation

6.1 Technical Evaluation

API testing were carried out to the Django backend to measure the performance of the system. The API testing was done using Postman. The custom API requests were made through Postman and the requests have been designed to mimic real world use.

6.1.1 API Testing

Both the server and the database are needed to be tested with multiple requests per second to simulate real life usage scenario. Specifically the Ordering API was tested in this experiment. The reason to this is to test how the server would handle multipe order requests at the same time. Postman was used to send multiple POST requests to the server because the platform allows tests to be written in JavaScript and run in the Postman environment. Provided with the correct body and headers in the API call, every request had to return a 200 response code. The treshold for any response time is 200ms to maintain a responsive system. The tests are separated between POST requests and GET requests. The POST requests are used to send data and the GET requests are used to retrieve from the database.

6.1.2 GET and POST Request Testing

Some of the most demanding GET and POST requests were selected in this test. The selected requests are the most demanding API requests in the system and those that are most likely to affect the experience of the customers and the admin of the system. The following are the API requests that were tested:

- **Get all orders**: `localhost:8000/api/orders/`
- **Get all menu items**: `localhost:8000/api/places/1`
- **Get Food Recommendations**: `localhost:8000/api/food-recommendations/`
- **Get Drink Recommendations**: `localhost:8000/api/drink-recommendations/`
- **Get Combo Recommendations**: `localhost:8000/api/combo-recommendations/`
- **Get all table sessions**: `localhost:8000/api/check_session/`
- **Customer Analysis**: `localhost:8000/api/customer-analysis/`
- **Total Sales Today**: `localhost:8000/api/total-sales-today`
- **Total Sales This Week**: `localhost:8000/api/total-sales-week`
- **Total Sales This Month**: `localhost:8000/api/total-sales-month`

Results

The results of the GET request testing are shown in the table below. The table shows the response time of each request and the response code that was returned. The response time is measured in milliseconds and the response code is the HTTP status code that was returned by the server. The response time is the time taken by the server to process the request and send a response back to the client. The response code is a three-digit code that indicates the status of the request. A response code of 200 indicates that the request was successful, while a response code of 400 or 500 indicates that there was an error.

The Recommendations API calls were simulated with a load of 20 virtual users over a 2-minute duration. A total of 4201 requests were sent which translates to a throughput of 33.13 requests per second and the average response time was 76ms across all requests from the recommender algorithm. The total number of requests that were successful was 4201 which translates to 100% success rate. The total number of failed requests was 0 which translates to 0% failure rate. Wherelse the Get all menu items API call was simulated with a load of 20 virtual users over a 2-minute duration. A total of 2059 requests were sent and maintained a throughput of 16.23 requests per second and the average response time was 167ms. This indicates that the customer experience when loading into the menu is still acceptable as the response time is still below 200ms. The Get all table sessions API call was simulated with the same environment. A total of 2219 requests were sent which translates to a throughput of 17.50 requests per second with an average response time of 28ms across the test.

On the admin side, the Total Sales Today, Total Sales This Week, Total Sales This Month, Customer Analysis API calls was simulated with a load of 20 virtual users over a 2-minute duration. A total of 8628 requests were sent which translates to a throughput of 68.00 requests per second and the average response time was 17ms across all requests. There was no errors throughout the testing which translates to a 100% success rate. However the Get all orders API call was simulated with the same environment. The were only a total of 495 requests that were made in the 2-minute duration which translates to a throughput of 3.91 requests per second and the average response time was 3787ms. This indicates that the Get all orders API call needs more optimization as the response time is well above 200ms.

In conclusion, API calls on both the customer and the admin side are performing well and the system is able to handle multiple requests at the same time. The response time is still well below 200ms which is acceptable to maintain a responsive system. This meets the non-functional requirements of the performance, reliability, and scalability of the system as the system is able to handle multiple requests at the same time and still maintain a responsive system.

6.2 Requirements Evaluation

6.3 Comparative Evaluation

6.4 User Evaluation

Chapter 7

Discussion

Chapter 8

Conclusion

Chapter 9

Appendix

Bibliography

[1] R. Paul, R. R. Bhandopia, D. R. Patil, J. D. Raut, and A. D. Gotmare, 'An Efficient Digital Ordering System for Restaurant', Ijresm.com. [Online]. Available: https://www.ijresm.com/Vol.3_2020/Vol3_Iss3_March20/IJRESM_V3_I3_103.pdf. [Accessed: 17-Mar-2024].

[2] '(PDF) Smart QR-based Restaurant Dine-in System with Sales Analysis', ResearchGate. [Online]. Available: https://www.researchgate.net/publication/360414606_Smart_QR-based_Restaurant_Dine-in_System_with_Sales_Analysis/fulltext/627a7d2b2f9ccf58eb3d5475/Smart-QR-based-Restaurant-Dine-in-System-with-Sales-Analysis.pdf. [Accessed: 17-Mar-2024].

[3] D. Carcache and R. E, "Automated Services Order System," prcrepository.org, 2013, Accessed: Mar. 17, 2024. [Online]. Available: https://prcrepository.org/xmlui/handle/20.500.12475/836?show=full

[4] K. Dhiman, "Online Food Ordering Management System," International Journal for Research in Applied Science and Engineering Technology, vol. 9, no. VII, pp. 2096–2107, Jul. 2021, doi: https://doi.org/10.22214/ijraset.2021.36835.

[5] '(PDF) Customer Acceptance of QR Menu Ordering System in Luxury Restaurants. A study of Xi'an, China', ResearchGate. [Online]. Available: https://www.researchgate.net/profile/Anshul-Garg-4/publication377586294_Customer_Acceptance_of_QR_Menu_Ordering_System_in_Luxury_Restaurants_A_study_of_Xi'an_China/links/65ae68e29ce29c458b939be6/Customer-Acceptance-of-QR-Menu-Ordering-System-in-Luxury-Restaurants-A-study-of-Xian-China.pdf. [Accessed: 17-Mar-2024].

[6] B. Shadaksharappa, Kotrachaithanya, and D. Kumar, "FOOD HUB A Model for Ordering In Restaurant Based On Qr Code Without Presence Of A Waiter At The Table," International Journal of Engineering Research in Computer Science and Engineering (IJERCSE), vol. 5, no. 5, pp. 2394–2320, 2018, Available: https://www.technoarete.org/common_abstract/pdf/IJERCSE/v5/i5/Ext_34756.pdf.

[7] E. E. Abel and E. Obeten, 'Restaurant customer self-ordering system: A solution to reduce customer/guest waiting time at the Point of sale', Ijcaonline.org. [Online]. Available: https://research.ijcaonline.org/volume111/number11/pxc3901332.pdf. [Accessed: 17-Mar-2024].

[8] Toast, Inc., "Toast: Restaurant Point of Sale and Management System," [Online]. Available: https://pos.toasttab.com/. [Accessed: Mar. 17, 2024].

[9] Square, Inc., "Square: Solutions For Your Small, Medium and Large Business," [Online]. Available: https://squareup.com/. [Accessed: Mar. 17, 2024].

[10] Zuppler, "Zuppler: Online Ordering System for Restaurants," [Online]. Available: https://www.zuppler.com/. [Accessed: Mar. 17, 2024].

[11] TouchBistro Inc., "TouchBistro: iPad POS System for Restaurants, Bars, and More," [Online]. Available: https://www.touchbistro.com/. [Accessed: Mar. 17, 2024].

[12] Bbot, "Bbot: Contactless Order and Pay," [Online]. Available: https://www.bbot.menu/. [Accessed: Mar. 17, 2024].

[13] Menufy, "Menufy: Online Ordering for Restaurants," [Online]. Available: https://www.menufy.com/. [Accessed: Mar. 17, 2024].

[14] Future Ordering, "Future Ordering: Digital Ordering Solutions," [Online]. Available: https://www.futureordering.com/. [Accessed: Mar. 17, 2024].

[15] React Team, "React – A JavaScript library for building user interfaces," React. [Online]. Available: https://reactjs.org/. [Accessed: Mar. 17, 2024].

[16] Django Software Foundation, "Django: The web framework for perfectionists with deadlines," Django. [Online]. Available: https://www.djangoproject.com/. [Accessed: Mar. 17, 2024].

[17] SQLite, "SQLite Home Page," SQLite. [Online]. Available: https://sqlite.org/index.html. [Accessed: Mar. 17, 2024].

[18] Axios, "Axios GitHub repository," GitHub. [Online]. Available: https://github.com/axios/axios. [Accessed: Mar. 17, 2024].

[19] D. Abramov and A. Clark, "Introducing Hooks," React, Oct. 25, 2018. [Online]. Available: https://reactjs.org/docs/hooks-intro.html. [Accessed: Mar. 17, 2024].

[20] T. Otwell, "Laravel: The PHP Framework For Web Artisans," Laravel. [Online]. Available: https://laravel.com. [Note: Replace with Django REST Framework if specific documentation is available]. [Accessed: Mar. 17, 2024].

[21] React Router, "React Router: Declarative Routing for React.js," ReactRouter. [Online]. Available: https://reactrouter.com/. [Accessed: Mar. 17, 2024].

[22] Bulma, "Bulma: a modern CSS framework based on Flexbox," Bulma. [Online]. Available: https://bulma.io/. [Accessed: Mar. 17, 2024].

[23] AWS, "Amazon Web Services (AWS) - Cloud Computing Services," Amazon. [Online]. Available: https://aws.amazon.com/. [Accessed: Mar. 17, 2024].

[24] Heroku, "Heroku: Cloud Application Platform," Heroku. [Online]. Available: https://www.heroku.com/. [Accessed: Mar. 17, 2024].

[25] Railway, "Railway: Infrastructure for modern applications," Railway. [Online]. Available: https://railway.app/. [Accessed: Mar. 17, 2024].

[26] Real Python, "Build a Recommendation Engine With Collaborative Filtering," Realpython.com, Jul. 10, 2019. https://realpython.com/build-recommendation-engine-collaborative-filtering/ (accessed Mar. 22, 2024)

[27] "What Is Collaborative Filtering: A Simple Introduction," Built In, 2019. https://builtin.com/data-science/collaborative-filtering-recommender-system (accessed Mar. 22, 2024).

[28] Shivam Baldha, "Introduction to Collaborative Filtering," Analytics Vidhya, Feb. 25, 2022. https://www.analyticsvidhya.com/blog/2022/02/introduction-to-collaborative-filtering/ (accessed Mar. 22, 2024).

[29] A. Roy, "Introduction To Recommender Systems- 1: Content-Based Filtering And Collaborative Filtering," Medium, Jul. 29, 2020. https://towardsdatascience.com/introduction-to-recommender-systems-1-971bd274f421 (accessed Mar. 22, 2024).

[30] Hybrid Recommender Systems, "Hybrid Recommender Systems — Saturn Cloud," Saturncloud.io, Feb. 23, 2023. https://saturncloud.io/glossary/hybrid-recommendation-system/ (accessed Mar. 22, 2024).

[31] Sciforce, “Deep Learning Based Recommender Systems - Sciforce - Medium,” Medium, Apr. 30, 2021. https://medium.com/sciforce/deep-learning-based-recommender-systems-b61a5ddd5456 (accessed Mar. 22, 2024).

[32] Krishnakanth Naik Jarapala, “Understanding Gale-Shapley (Stable Matching) Algorithm and its Time Complexity,” Medium, Jan. 31, 2023. https://medium.com/aiskunks/understanding-gale-shapley-stable-matching-algorithm-and-its-time-complexity-4b814ee2642 (accessed Mar. 22, 2024).

[33] “How Netflix’s Recommendations System Works,” Help Center, 2024. https://help.netflix.com/en/node/100639 (accessed Mar. 23, 2024).

[34] M. Geuens, “Research on Influencing Factors of Food Choice and Food Consumption,” Foods, vol. 12, no. 6, pp. 1306–1306, Mar. 2023, doi: https://doi.org/10.3390/foods12061306.

[35] Ingrid H.M. Steenhuis, W. E. Waterlander, and Anika de Mul, “Consumer food choices: the role of price and pricing strategies,” Public Health Nutrition, vol. 14, no. 12, pp. 2220–2226, Jul. 2011, doi: https://doi.org/10.1017/s1368980011001637.