

Cloud-Based Restaurant Management System

A Project Report
Presented to
The Faculty of the College of
Engineering
San Jose State University
In Partial Fulfillment
Of the Requirements for the Degree
Master of Science in Software Engineering

By
Netra Amrale, Martin Duong, Mayuri Gupta, Ruchitkumar Patel
May 2023

APPROVED

Andrew Bond, Project Advisor

[Program Director's Name], Director, MS Software Engineering

[Department Chair's Name], Department Chair

ABSTRACT

Cloud-Based Restaurant Management System

By Netra Amrale, Martin Duong, Mayuri Gupta, Ruchitkumar Patel

The restaurant industry is constantly evolving with innovative technologies to help better cater towards consumers' preferences. The COVID-19 pandemic has played a significant role in shaping these consumer preferences. According to a survey by Bluedot, the pandemic has reduced the time a consumer is willing to wait for food from 10 minutes to 6 minutes. Currently there are many restaurant management solutions such as the Presto tablet, DoorDash, Uber Eats, OpenTable, and Toast, each with similar and varying functions. Consumers often must shuffle through multiple applications for food whether it be for dine-in, takeout, or delivery. 52% of consumers surveyed by the National Restaurant Association would like to see restaurants incorporate technology to make ordering and paying easier. Restaurants are often deterred from investing in these solutions for reasons such as cost of and ease of use, but to stay competitive, restaurants have no choice but to invest. According to research done by Toast, the average restaurant today offers five different ordering paths and seven different service models, with 74% of restaurants expected to increase their technology budget in 2023.

This project's purpose is to develop the foundations for an all-in-one restaurant management Software as a Service (SaaS) backed by artificial intelligence to improve consumer experience and restaurant service quality. The proposed SaaS application will utilize a microservice architectural style to ensure scalability and availability of the application. While the concept of the application is not novel, it seeks to address some of the shortcomings of current solutions such as the need for multiple applications each with a different purpose. Additionally, a new challenge for the restaurant industry partly due to the COVID-19 pandemic is addressed: labor shortages. This application will hopefully serve as a good foundation for future expansion to further cater to the needs of not only restaurants, but any business, and most importantly, consumers.

Acknowledgments

The authors are deeply indebted to Professor Andrew Bond for his invaluable comments and assistance in preparing this study.

Table of Contents

| | | |
|-------------------|---|-----------|
| Chapter 1. | Project Overview..... | 1 |
| | Introduction | 1 |
| | Proposed Areas of Study and Academic Contribution | 4 |
| | Current State of the Art | 5 |
| | In-Store Food Ordering Solutions | 6 |
| | Remote Food Ordering Solutions..... | 7 |
| Chapter 2. | Project Architecture | 8 |
| | Introduction | 8 |
| | Cloud Computing | 8 |
| | Microservice Architecture | 9 |
| | React / Redux Architecture | 10 |
| | Architecture Subsystems | 11 |
| Chapter 3. | Technology Descriptions | 15 |
| | Client Technologies..... | 15 |
| | Programming Languages..... | 15 |
| | Frameworks / Libraries | 15 |
| | Middle-Tier Technologies | 15 |
| | Programming Languages..... | 15 |
| | Frameworks / Libraries | 15 |
| | Data-Tier Technologies | 15 |
| | Databases..... | 15 |
| | Cloud Technologies..... | 16 |
| | Third-Party APIs | 16 |
| | Miscellaneous Tools..... | 16 |
| Chapter 4. | Project Design..... | 18 |
| | Client Design | 18 |
| | Middle-Tier Design | 26 |
| | Data-Tier Design | 37 |
| | Database Schemas | 37 |
| Chapter 5. | Project Implementation..... | 41 |
| | Client Implementation | 41 |
| | Middle-Tier Implementation | 42 |

| | |
|---|-----------|
| API Endpoints | 42 |
| Data-Tier Implementation | 45 |
| Chapter 6. Testing and Verification | 48 |
| Testing Tools | 48 |
| Environment | 48 |
| Test Cases | 48 |
| Chapter 7. Performance and Benchmarks | 52 |
| Entry Criteria | 52 |
| Exit Criteria | 52 |
| Environmental Requirements | 52 |
| System Environment Requirements | 52 |
| Mandatory Test Cases | 53 |
| Key Performance Indicators and Benchmarking Criteria | 53 |
| Chapter 8. Deployment, Operations, and Maintenance | 61 |
| Deployment Plan | 61 |
| Operational Plan | 61 |
| Maintenance Plan | 61 |
| Chapter 9. Summary, Conclusions, and Recommendations | 63 |
| Summary | 63 |
| Conclusions | 63 |
| Recommendations for Further Research | 63 |

List of Figures

| | |
|--|----|
| FIGURE 1 - MICROSERVICES CLOUD DEPLOYMENT..... | 10 |
| FIGURE 2 - CONTAINER BASED DEPLOYMENT FOR ORDER, PAYMENT, AND TABLE RESERVATION SERVICES | 13 |
| FIGURE 3 - CLOUD DEPLOYMENT FOR USER AND MENU MANAGEMENT, AND REVIEWS SERVICES | 14 |
| FIGURE 4 - HOME PAGE | 18 |
| FIGURE 5 - TABLE RESERVATION PAGE..... | 18 |
| FIGURE 6 - MENU WITH CHAT BOT AND CART..... | 19 |
| FIGURE 7 - CHAT BOT WITH EXAMPLE RESPONSE | 19 |
| FIGURE 8 - MENU ITEM WITH REVIEWS | 20 |
| FIGURE 9 - REVIEW DIALOG | 20 |
| FIGURE 10 - EXAMPLE ORDER CONFIRMATION WITH REVIEW BUTTON | 21 |
| FIGURE 11 - ADMIN DASHBOARD..... | 21 |
| FIGURE 12 - MANAGER DASHBOARD | 22 |
| FIGURE 13 - MANAGER MENU VIEW | 22 |
| FIGURE 14 - ADD NEW MENU ITEM DIALOG..... | 23 |
| FIGURE 15 - EXAMPLE REDUX DATA FLOW | 23 |
| FIGURE 16 - TABLE RESERVATION FLOW | 24 |
| FIGURE 17 - ORDER FLOW..... | 24 |
| FIGURE 18 - PAYMENT FLOW..... | 25 |
| FIGURE 19 - TABLE RESERVATION CLASS DIAGRAM..... | 26 |

| | |
|---|----|
| FIGURE 20 - ORDER MANAGEMENT SEQUENCE DIAGRAM..... | 27 |
| FIGURE 21 - PAYMENT SERVICE SEQUENCE DIAGRAM..... | 28 |
| FIGURE 22 - REVIEWS SERVICE SEQUENCE DIAGRAM | 29 |
| FIGURE 23 - ORDER MANAGEMENT USE CASE | 30 |
| FIGURE 24 - ADMIN USER MANAGEMENT USE CASE..... | 31 |
| FIGURE 25 - MANAGER MENU MANAGEMENT USE CASE | 32 |
| FIGURE 26 - REVIEWS SERVICE USE CASE..... | 33 |
| FIGURE 27 - JENKINS PIPELINE TO BUILD THE DOCKER IMAGE AND DEPLOY TO EKS CLUSTER..... | 34 |
| FIGURE 28 - ORDER SERVICE CI/CD..... | 34 |
| FIGURE 29 - RESERVATION SERVICE CI/CD | 35 |
| FIGURE 30 - PAYMENT SERVICE CI/CD..... | 35 |
| FIGURE 31 - VAULT FOR SECRETS MANAGEMENT..... | 35 |
| FIGURE 32 - SERVERLESS FRAMEWORK CLOUD DEPLOYMENT (USER, MENU MANAGEMENT) | 36 |
| FIGURE 33 - SERVERLESS FRAMEWORK CLOUD DEPLOYMENT (REVIEWS) | 36 |
| FIGURE 34 - USER MANAGEMENT | 37 |
| FIGURE 35 - MENU MANAGEMENT | 38 |
| FIGURE 36 - WAITLIST | 38 |
| FIGURE 37 - TABLE RESERVATION..... | 39 |
| FIGURE 38 – ORDER AND PAYMENT | 39 |
| FIGURE 39 - REVIEWS SERVICE ERD..... | 40 |
| FIGURE 40 - MENU MANAGEMENT STORED PROCEDURES | 46 |

| | |
|--|----|
| FIGURE 41 - USER MANAGEMENT STORED PROCEDURES..... | 46 |
| FIGURE 42 - PAGESPEED, HOME PAGE MOBILE RESULTS | 55 |
| FIGURE 43 - PAGESPEED, HOME PAGE DESKTOP RESULTS..... | 56 |
| FIGURE 44 - PAGESPEED, TABLE RESERVATION MOBILE RESULTS..... | 57 |
| FIGURE 45 - PAGESPEED, TABLE RESERVATION DESKTOP RESULTS | 58 |
| FIGURE 46 - PAGESPEED, MENU PAGE MOBILE RESULTS | 59 |
| FIGURE 47 - PAGESPEED, MENU PAGE DESKTOP RESULTS..... | 60 |

Chapter 1. Project Overview

Introduction

The COVID-19 pandemic and ever-changing consumer dining preferences have caused a technological evolution in the restaurant industry. The pandemic negatively affected the restaurant industry as it lost approximately 3.1 million jobs, and at least 110,000 restaurants closed either temporarily or permanently [1]. As a result, consumers' dining preferences and the way restaurants provided service changed. Restaurants had to shift from people sitting down and dining in, to people ordering delivery or takeout. In 2021, 53% of all adults, 54% of Gen Z, 64% of Millennials surveyed said that food delivery and takeout are "essential to the way they live" [2]. This shift has caused consumers to rely on online food ordering and delivery applications such as DoorDash, Uber Eats, and Grubhub. As of 2022, DoorDash has a US food delivery market share of 65% [3]. Restaurants across the board saw double-digit percentage increases in off-premises sales as compared to pre-pandemic [2].

According to a survey by Deloitte, the largest professional services firm in the world, 70% of respondents now prefer to order food online and have it delivered [4]. Consumers can choose from many food ordering applications. A variety of food ordering applications means that restaurants must manage separate user interfaces for each individual application, thereby increasing workload during a time of labor shortages. On average, a restaurant in 2023 offers five different ordering paths and seven different service models [5]. The National Restaurant Association stated that 84% of consumers prefer to go to a restaurant than cook and clean in their leisure time [6], exacerbating the labor shortage problem. In addition to convenience, consumers want fast service from restaurants. Bluedot, a San Francisco-based company that provides location technology for mobile applications, surveyed more than 1,500 consumers and 77% of respondents would consider not dining at a restaurant if there is a long wait [7]. The pandemic has also reduced the time a consumer is willing to wait for food from 10 minutes to 6 minutes [7]. A lot of

time is spent on dining out when considering the commute to the restaurant, the wait time to order food or sit down at a table, the wait time for food preparation, and the time to find parking. With consumers constantly demanding higher levels of service, restaurants have no choice but to invest in various technologies such as touchless payments and online food ordering systems to stay competitive, especially for smaller restaurants. Online reviews are another important attribute for restaurants to stay competitive. In 2021, 30% of consumers read online reviews [8]. Consumers are more likely to dine at a restaurant with positive reviews and even more likely if the restaurant is crowded [9]. Unfortunately, descriptive, and helpful reviews can be overshadowed by negative reviews which play a larger role in a consumer's decision making. With the way reviews are currently handled on many websites, such as Yelp, reviews can be susceptible to monopolization, manipulation or mislabeling as to whether a review is helpful or not [10, 11].

Gen Z and Millennials, or those currently aged between 4 and 40, account for 40% of global consumers and account for approximately \$350 billion (about \$1,100 per person in the US) in spending power in the United States alone [12]. Restaurants must cater to this technological advance generation or risk losing valuable business to their more tech savvy competitors. These generations are 99% more likely to consult social media in determining where to dine [13] and as a result, restaurants get free marketing by word-of-mouth. This puts pressure on restaurants to provide the best dining experience possible. Online review sites such as Yelp and TripAdvisor put further pressure on restaurants to always provide exceptional service. Restaurants that fail to attract Gen Z and Millennials may see retention rates decline by at least 43% [14]. When it comes to contactless experiences, consumers, regardless of generation, have placed a restaurant's cleanliness and sanitation level as deciding factors in choosing where to dine at [15]. In 2023, 47% of restaurants expect competition to be more intense than last year [6].

As the restaurant industry continues to evolve with innovative technologies, many of these technologies each come with their advantages and disadvantages. Since 2007, the

use of technology in the industry has led to a 70% reduction in the number of required employees a restaurant needs for efficient operation [16]. Despite this, labor shortages are still an ongoing struggle. Consumers are given many options when it comes to deciding how they will order food and restaurants have many options on how to handle those orders. These options range from in-store food ordering kiosks to remote food ordering mobile applications. The pandemic has accelerated the speed of which consumer interactions are being digitized by double digit percentage increases despite remaining constant years prior [17]. The pandemic has caused consumers to prefer contactless payments [18] with a survey conducted by Mastercard, a global payment technology solutions provider, stating that 79% of consumers have transitioned to contactless payments [19]. In an industry previously dominated by pen and paper pre-pandemic, many older restaurants struggle with the sudden change. On the other hand, 84% of new restaurants chose a cloud-based Point of Sale (POS) system in 2022 [5].

This project's purpose is to develop the foundations for an all-in-one restaurant management Software as a Service (SaaS) backed by artificial intelligence to improve consumer experience and restaurant service quality. The restaurant management SaaS application is Application Programming Interface (API) driven. The set of APIs are deployed as microservices. The set of microservices include menu management, user management, order, notification, table reservation, reviews, and payment service. These microservices will make up the complete solution to drive a restaurant's management User Interface (UI). A microservice architectural style has many advantages when compared with a monolithic architectural style. The microservices are containerized using Docker. Kubernetes is an open-source platform that manages deployment of these Docker containers. Therefore, the dependency on proprietary hardware is not always necessary due to the application being accessible on any internet capable device. Additionally, QR codes will allow consumers to, for example, quickly open menus and order food all from their smartphone. Although these features are available in existing solutions, they are not an all-in-one solution.

By applying various aspects of cloud computing such as Infrastructure as a Service (IaaS) and container orchestration for the deployment of the SaaS application, which is all done in the background, restaurants can provide worry-free convenient and fast service to consumers with minimal downtime and investment. As stated, this project's purpose is to propose an all-in-one restaurant management Software as a Service (SaaS) application developed to improve the overall restaurant service quality and consumer experience. In this case, the term, “all-in-one,” is defined as having all the features a restaurant and consumer require in one central application. This system will serve as a cost-effective and highly secure solution for restaurants with both consumers and restaurant owners reaping the benefits. Restaurants will be able to manage various aspects of their restaurant and view valuable analytics. Consumers will be able to order food, reserve tables, put themselves on a waitlist, chat with an AI bot, write, and view restaurant reviews all in a singular SaaS application.

Proposed Areas of Study and Academic Contribution

This project's boundaries will consist of a collection of microservices that will all form a cloud-based restaurant management system. This system will be usable by both restaurants and consumers. The following microservices will be included:

- Menu Management Service – Manages every restaurant’s menu
- Notification / Email Service – Handles all notifications such as order and payment confirmation
- Order Management Service – Handles users’ orders and restaurants can manage and fulfill these orders
- Payment Service – Handles order payments

- Reviews Service – Allows users to upload photos, and write and view reviews, as well as chat with a bot regarding food, both driven by artificial intelligence
- Table Reservation Service – Allows users to reserve tables at restaurants ahead of time and be notified by SMS
- User Management Service – Contains all user operations such as registration, management, and role-based access to different resources

While this project is not a completely novel idea, it is important because while there are current solutions that do what this project is aiming to do, most current solutions do not provide a fully seamless and consolidated experience for both restaurants and consumers. Consumers must shuffle through dozens of different applications to order food, reserve tables, or check reviews. In the case of review websites such as Yelp and TripAdvisor, reviews are very general and often it is difficult for consumers to find what they are looking for such as how a certain item is. Restaurants must deal with the overhead costs of extra software and hardware associated with each solution. Additionally, restaurants often must maintain multiple solutions to stay competitive, leading to costs adding up due to subscription and / or commission fees for each solution. Dine-in, waiter-less devices such as the Presto tablet are often unfamiliar to consumers and require restaurants to invest in proprietary hardware that may become obsolete in the future requiring future investment. A solution called Toast comes the closest to providing an all-in-one restaurant management solution but is still lacking in some areas such as some hardware requirements and reviews integration. Therefore, this project does not aim to develop a novel solution but rather aims to improve upon current solutions.

Current State of the Art

There are current solutions such as the Presto tablet, DoorDash, OpenTable, and Toast that provide features such as ordering food online, reserving tables, and providing

point-of-sale devices. Toast comes the closest to providing an all-in-one cloud-based restaurant management system. They provide all-in-one point of sale and restaurant management systems built on the Android operating system, online ordering, and most recently mobile order and pay. Essentially providing both in-store and online food ordering solutions and more [20].

Online review sites such as Yelp and TripAdvisor both allow consumers to write and view reviews and photos for restaurants. They both also allow consumers to order food but not directly; food ordered through these sites is fulfilled by third parties such as DoorDash and Grubhub. Additionally, Yelp allows consumers to put themselves on a waitlist remotely or in-person with an estimated seating time.

In-Store Food Ordering Solutions

Many in-store food ordering solutions currently exist such as self-ordering kiosks at fast food restaurants such the Presto tablet. Self-ordering kiosks give consumers a consistent ordering experience and allow them to have more direct control over orders to avoid mistakes. The Presto tablet is a table-top self-ordering kiosk for dine-in restaurants that includes entertainment such as games and movies for a fee. Consumers can place and pay for orders by themselves therefore if restaurants are understaffed, staff can be allocated where they are needed most.

Since self-ordering kiosk solutions are in-store only, there are some disadvantages. Consumers are not able to order ahead of time as well as checking how busy a certain restaurant is before arriving. The last point is especially important as consumers are less likely to dine at a restaurant if they must wait an extended period, especially since the pandemic has decreased consumers' patience [21].

Regarding consumers' concerns about high-touch surfaces such as self-ordering kiosks, Toshiba is currently developing a touchless point-of-sale system, "Digi POS," that allows for consumers to checkout at 7-11 convenience stores in Japan without contacting any touchscreens [22, 23].

Remote Food Ordering Solutions

Just as there are in-store solutions, there are many remote food ordering solutions. Mobile applications such as DoorDash and Uber Eats allow consumers to order food ahead of time for delivery or takeout. While other applications such as OpenTable and TabbedOut allow consumers to reserve tables ahead of time and pay whenever they are done dining, respectively. Consumers can place orders ahead of time and check restaurant table availability from anywhere. If consumers wish to dine without leaving their homes that is also possible with delivery.

Convenience comes at a fee; often consumers will have to pay more for delivery than if they were to pick up or dine-in. Restaurants may not even break even on delivery orders. While these services were optional pre-pandemic, for restaurants to stay afloat during the pandemic, they had no choice but to invest in these services such as DoorDash and Uber Eats to compete [24]. For example, DoorDash charges a minimum of 21% commission if a restaurant were to use them for delivery [25] and other services are similar. These costs begin to add up quickly for a restaurant but to compete, restaurants have no choice but to utilize these services.

Chapter 2. Project Architecture

Introduction

Cloud Computing

Cloud computing is the pay-as-you-go provision of on-demand computing services over the internet. Software as a Service (SaaS), Platform as a Service (PaaS), and Infrastructure as a Service (IaaS) are the three main forms of cloud architectures [26]. Using a remote server on the Internet and cloud computing, anyone may access and manage data without the need for specialized computer hardware or software. Amazon Web Services (AWS) provides shared resources such as hardware, software, and network via the Internet. SaaS is an entire software that runs natively over the Internet. Some examples of SaaS are Dropbox, Google Drive, and DocuSign. Furthermore, IaaS provides the virtual infrastructure for SaaS applications to use compute servers on-demand. Applications can be isolated from the environment in which they function, enabling the straightforward and reliable deployment of container-based applications regardless of whether the intended environment is a personal laptop for the developer, a public cloud, or a private data center. AWS was the first public cloud provider to offer infrastructures on demand. Below are some benefits of IaaS:

- Elastic Architecture - The traditional scale-up or scale-out approaches for physical hardware can be outperformed by the AWS cloud providers.
- Design for Failure / Disaster - Multiple backup strategies are in place to ensure business continuity.
- High Availability - AWS cloud providers support multiple availability zones and regions to make your system more robust and always running.
- High Performance - Multiple caching techniques are incorporated to support high performance low latency requests.

- Secure - Enforced well-established security practices such as firewalls, DDOS attacks, key-based authentication and so on are covered.
- Continuous Monitoring - Cloud based SaaS applications require continuous monitoring and AWS cloud providers provision out-of-box monitoring services.

Microservice Architecture

The microservice architectural style has a lot of advantages over monolithic architectural style. The monolithic application is a single consolidated unit, whereas a microservice architectural style application breaks an application into small services. The service has its own business layer, data layer and database [27]. Each microservice communicates with another service using HTTP or HTTPS protocol [28]. Containers are lightweight as compared to VM, as they follow operating system level virtualization. Docker is used to containerize microservices [29]. The container orchestration is managed using Kubernetes to deploy and maintain containers. The Kubernetes allows users to easily scale up and scale down a microservice [30]. In addition, in a study [31], the authors verified effectiveness of availability with default Kubernetes configuration against multiple test scenarios such as terminating container and terminating machine. Thus, the microservice architecture is used to ensure scalability and availability.

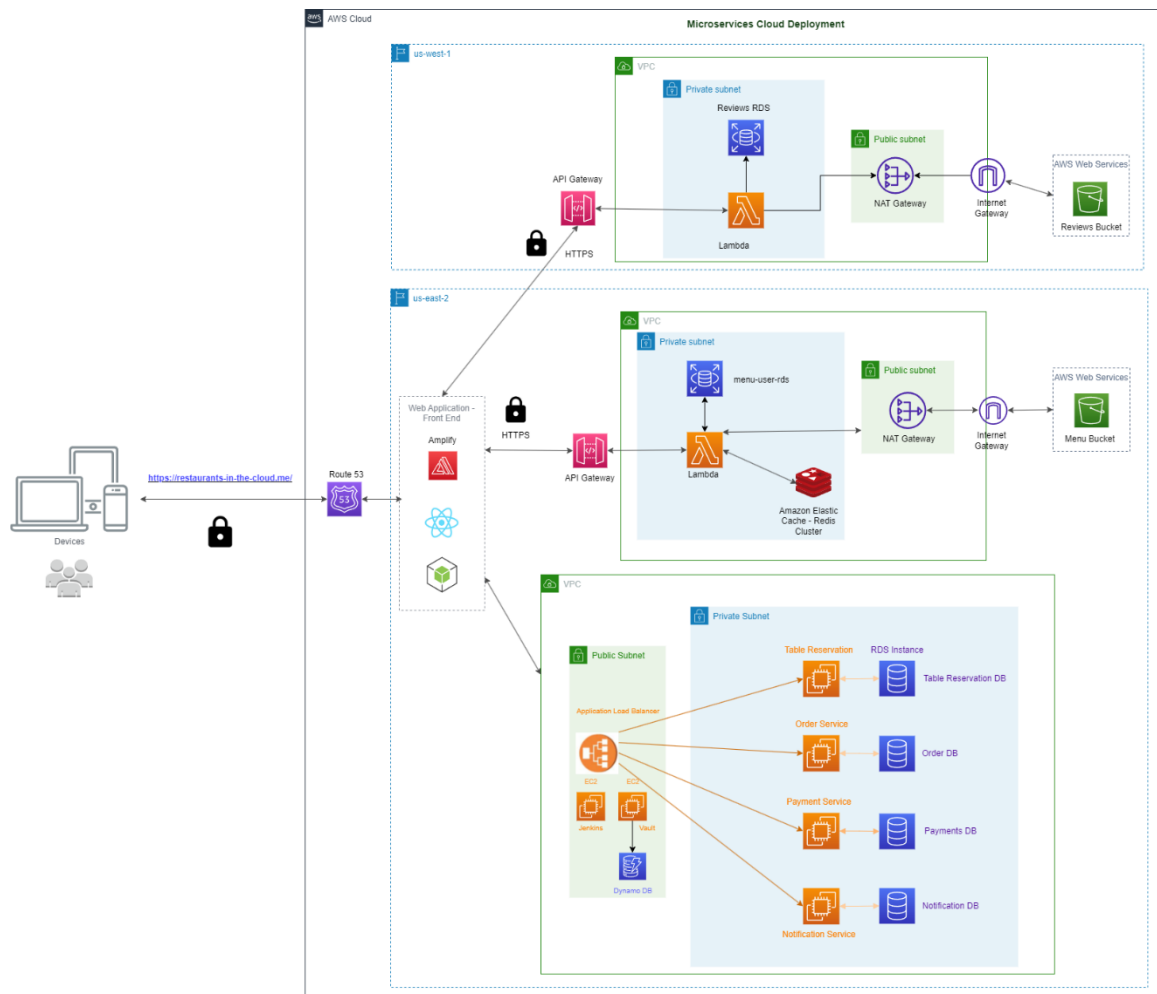


Figure 1 - Microservices Cloud Deployment

React / Redux Architecture

React architecture is an important aspect of building scalable and maintainable applications. By following good architectural principles, applications can be created that are easier to understand, test, and modify. React provides a flexible architecture that allows developers to create applications that can handle complex state management and data flow [32].

One of the key benefits of using React architecture is it helps to keep code organized and easy to maintain. By separating the concerns of different components and breaking

down the application into smaller parts, it becomes easier to debug and test the code. Additionally, it allows for better code reuse, since components can easily be reused across different parts of your application.

Another benefit of using React architecture is that it provides a declarative programming model. This means that given a description of what a application should look like and React takes care of updating the DOM to match that description. This makes it easier to code since the low-level details of updating the DOM do not need to be worried about [33].

In addition to the React architecture, Redux architecture is important because it provides a predictable and centralized state management solution for larger and more complex React applications. In Redux, the application state is stored in a single store, which acts as a single source of truth for the entire application. This makes it easier to manage and maintain the state of the application, as it eliminates the need for multiple state management solutions and reduces the possibility of bugs and inconsistencies. Redux also makes it easier to manage application complexity by enforcing a strict unidirectional data flow, which makes it easier to debug and test the application. It also provides a clean separation of concerns between the state and the user interface components, which helps to make the application more modular and easier to refactor [34].

Architecture Subsystems

The overall microservices cloud deployment diagram of the Cloud-Based Restaurant Management System is shown in the previous section in Figure 1. Consumers can access the application via QR code from their mobile phones or through a web page on kiosk devices such as a tablet placed on a restaurant table. The application is internally resolved in AWS Route 53, and all requests are forwarded to an Internet Application Load

Balancer. Webpages are served through a web server EKS pod with a NGINX container running on them.

Each microservice will run in its own EKS pod along with Sidecar Containers such as Vault Agent (secret management), monitoring and logging (Data Dog). Application code runs in a container inside the pod. One microservice per pod will be implemented as depicted in the architecture diagram. Images are stored in Elastic Container registries which will be deployed to pods via Jenkins Job. One Master Jenkins server (cater CI/CD) will be installed on AWS EC2 instances to create docker images and store them on Artifactory and deploy images on EKS pods.

The Vault server will be provisioned on an EC2 instance with AWS DynamoDB as storage. Vault will store the secrets for the application, database, and infrastructure code. AWS S3 buckets will be used to store the images required to be displayed on the order web page delivered securely via AWS CloudFront. Since this is microservices architecture, each microservice will have its own database created on one AWS RDS instance. Below is the EC2 deployment diagram for four services: notification, order, payment, and table reservation.

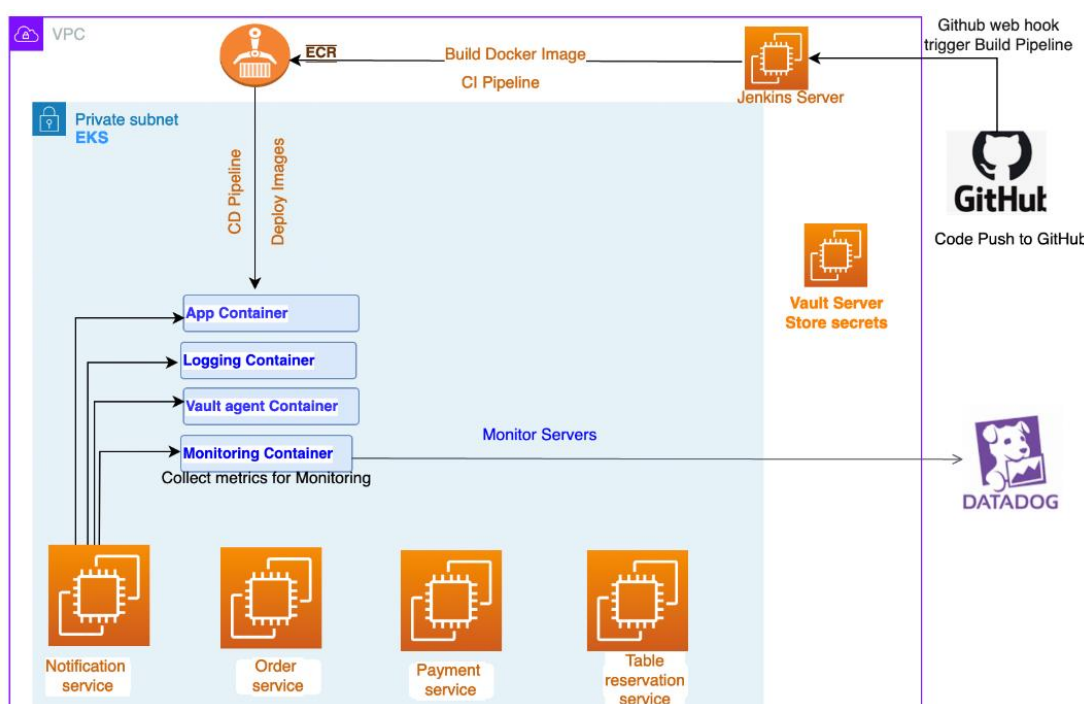
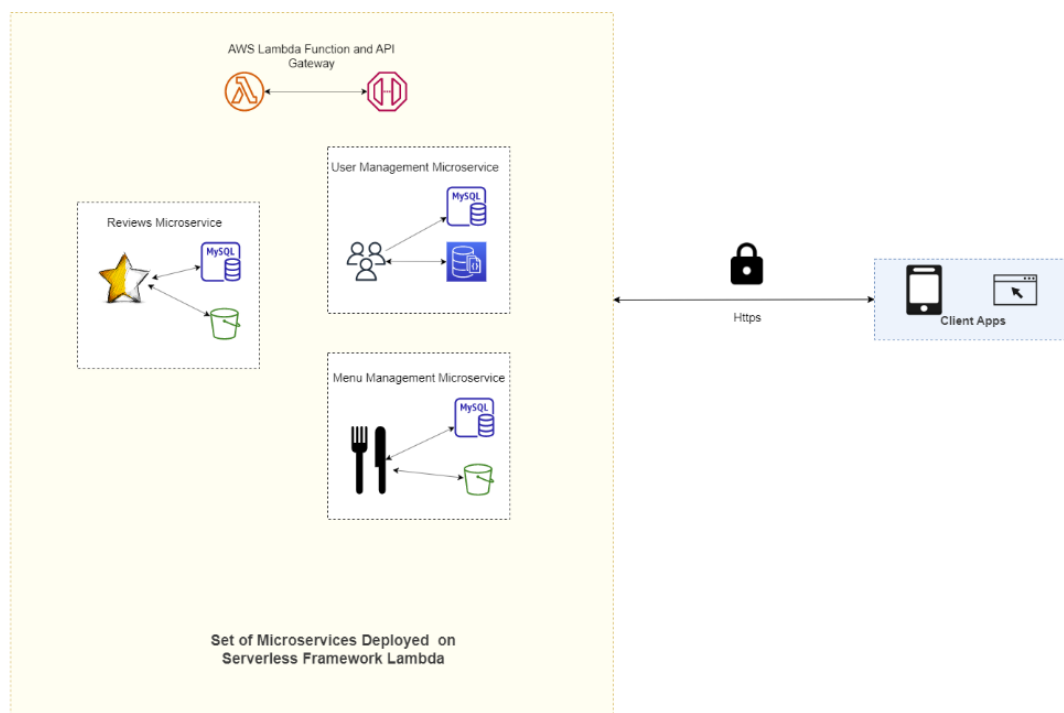


Figure 2 - Container Based Deployment for Order, Payment, and Table Reservation Services

In addition to the deployment of the four previously stated services, three other services: user management, menu management, and reviews will be deployed using AWS Lambda. The conceptual microservice architecture diagram below illustrates the set of microservices driving the cloud-based restaurant management application using Lambda. The microservices will be exposed using the HTTP / HTTPS protocol. Lambda, Elastic Cache Redis, and RDS MySQL are a part of one VPC (Virtual Private Cloud) for the user and menu management services. While for the reviews service, Lambda and RDS PostgreSQL are a part of another VPC that consists of several private subnets and a public subnet for high availability. Lambda enables serverless deployment for the backend APIs for user management, menu management, and reviews services. Since the Lambda functions are bounded to a VPC, internet access is not available, therefore, by using a NAT gateway in a public subnet we allow the Lambda functions to access S3 and

third-party APIs. Two AWS S3 buckets are connected to each VPC using an Internet Gateway endpoint. AWS API Gateway sits in front of the Lambda functions as a proxy.

The deployment of the front-end is done using AWS Amplify. The domain name purchased from Namecheap for this application is: “restaurants-in-the-cloud.me” The hosted zone for “restaurants-in-the-cloud.me” is configured in Route 53. The custom DNS servers are set up in the Namecheap domain name configuration. The domain is added in domain management (Amplify). Amplify generates two domains: <https://www.restaurants-in-the-cloud.me/> and <https://restaurants-in-the-cloud.me/>



**Conceptual Microservice Architecture for
Cloud-Based Restaurant Management System**

Figure 3 - Cloud Deployment for User and Menu Management, and Reviews Services

Chapter 3. Technology Descriptions

Client Technologies

Programming Languages

- HTML
- CSS
- JavaScript / JSX

Frameworks / Libraries

- React
- Redux Toolkit
- MUI / Material-UI

Middle-Tier Technologies

Programming Languages

- Python

Frameworks / Libraries

- Flask
- Flask-JWT
 - Adds basic JWT features to the user management service
 - Returns encoded JWT token for conditional rendering in the user interface
- Redis
 - In-memory data structures stores used as database and message broker

Data-Tier Technologies

Databases

- MySQL RDS
- PostgreSQL RDS

Cloud Technologies

- Amazon Web Services (AWS)
 - Amplify
 - API Gateway
 - CloudWatch
 - CodeWhisper
 - Elastic Compute Cloud (EC2)
 - Elastic Kubernetes Service (EKS)
 - Identity and Access Management (IAM)
 - Lambda
 - Relational Database Service (RDS)
 - Route 53 (R53)
 - S3
 - Virtual Private Cloud (VPC)
- Datadog (Cloud Monitoring as a Service)
- Docker
- GitHub Copilot
- GitHub (Git)
- HashiCorp
 - Vault
 - Terraform
- Jenkins

Third-Party APIs

- OpenAI API (Large Language Model, LLM)
 - GPT-3.5-Turbo, GPT-4, ChatGPT
 - Used for review summarization and chat bot functionalities
 - Assisted in development by curating menu items, writing boilerplate code, and debugging
- Stripe API (Payment)

Miscellaneous Tools

- Canva / draw.io (Diagram creation)
- Chrome DevTools
- Google PageSpeed Insights (Web application performance testing)

- Jira (Project management)
- Midjourney (AI image generation)
 - Used to generate all menu item images
- Postman (Testing API endpoints)
- Redux DevTools
- Zappa (Python package to deploy serverless microservices with Flask)

Chapter 4. Project Design

Client Design

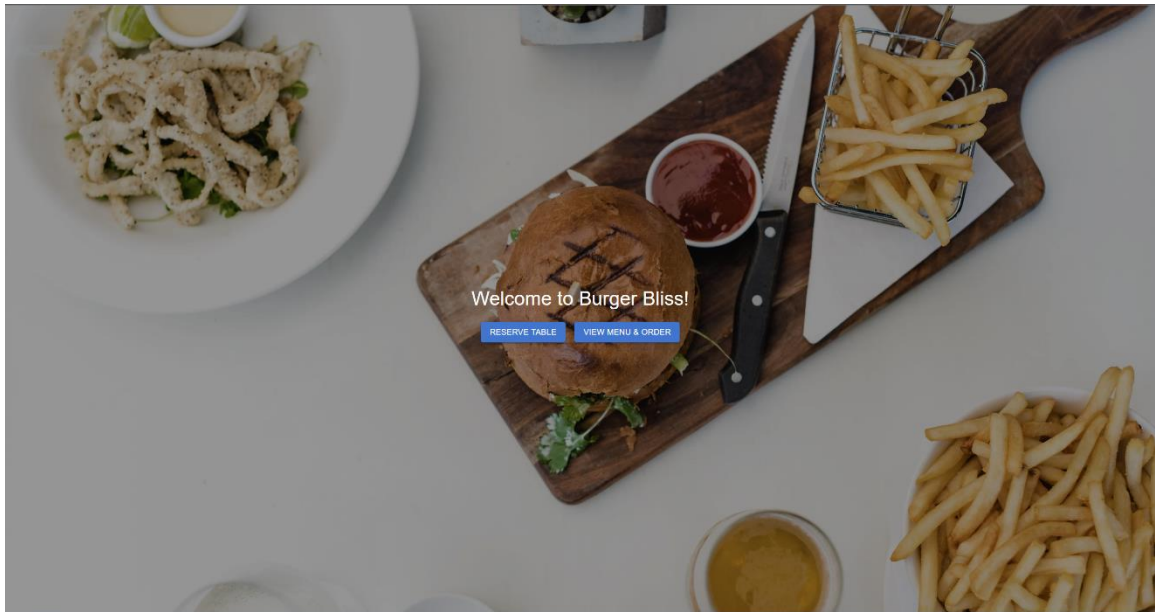


Figure 4 - Home Page

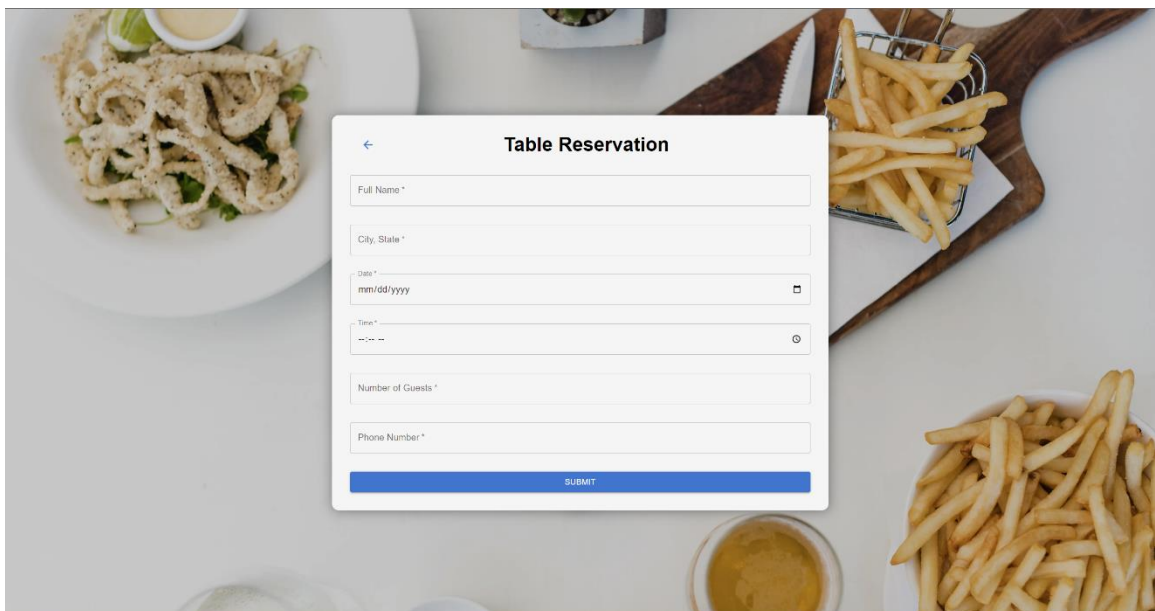


Figure 5 - Table Reservation Page

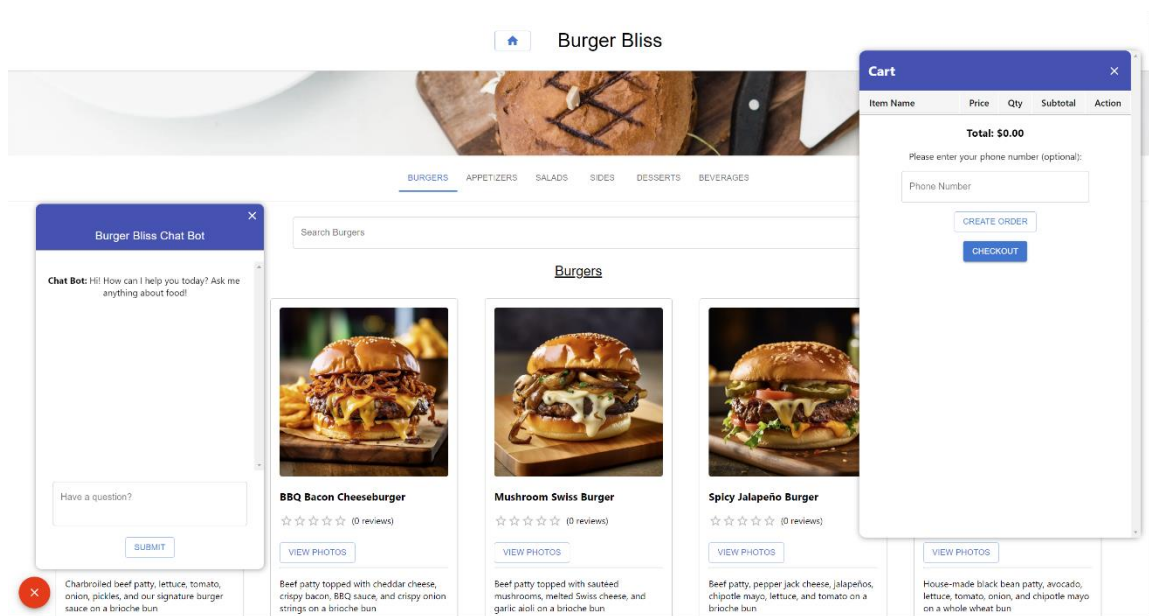


Figure 6 - Menu with Chat Bot and Cart

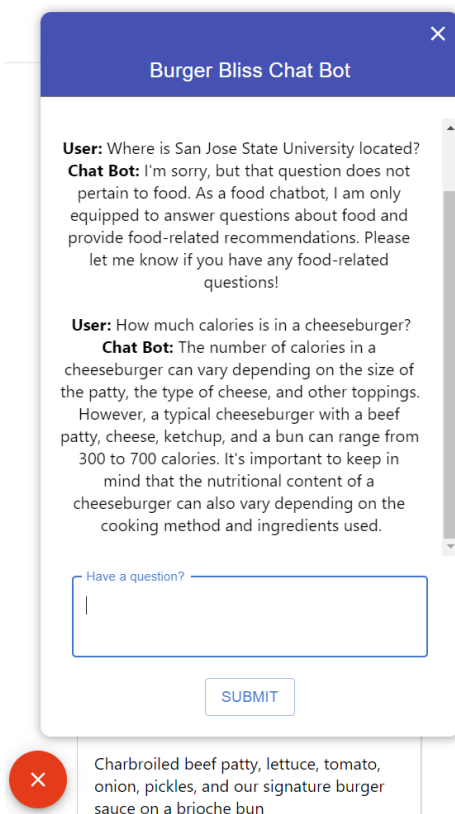


Figure 7 - Chat Bot with Example Response

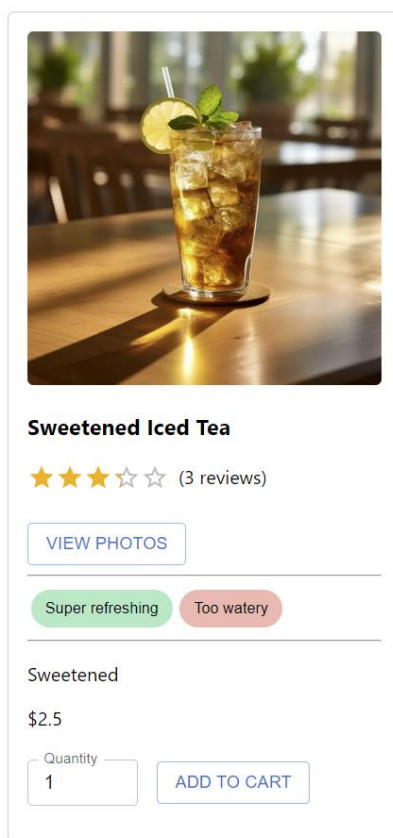


Figure 8 - Menu Item with Reviews

Submit a Review for Classic Beef Burger

×

★ ★ ★ ★ ★

Comment (optional)

Max Characters: 0/50

Upload a Photo (optional)

CHOOSE PHOTO

CANCEL

SUBMIT REVIEW

Figure 9 - Review Dialog

Order Confirmation - Burger Bliss

Order #1

| Item Name | Price | Qty | Subtotal | Leave a Review! |
|------------------------|---------|-----|----------|----------------------------------|
| Classic Beef Burger | \$10.95 | 2 | \$21.90 | REVIEW THIS ITEM |
| BBQ Bacon Cheeseburger | \$12.95 | 3 | \$38.85 | REVIEW THIS ITEM |

Total: \$60.75

[HOME](#)

Figure 10 - Example Order Confirmation with Review Button

Admin Dashboard

[LOGOUT](#)

[VIEW ALL RESTAURANTS](#)

[CREATE NEW RESTAURANT](#)

[REGISTER NEW ADMIN](#)

[VIEW ALL ADMINS](#)

| Search Restaurants | |
|---------------------|--|
| Restaurant ID | Restaurant Name |
| 8963039430973587531 | P.J. Fresh (224 Daniel Payne Drive) |
| 8963039430973587532 | J'li'z Smoothie-N-Coffee Bar |
| 8963039430973587533 | Philly Fresh Cheesesteaks (541-B Graymont Ave) |
| 8963039430973587534 | Papa Murphy's (1580 Montgomery Highway) |
| 8963039430973587535 | Nelson Brothers Cafe (17th St N) |
| 8963039430973587536 | Ocean Restaurant |
| 8963039430973587537 | Jinsei Sushi |
| 8963039430973587538 | Little India |
| 8963039430973587539 | Captain D's (1284 Decatur Hwy) |
| 8963039430973587540 | Cajun Bistro Express |

Rows per page: 10 ▾ 1-10 of 243 < >

Figure 11 - Admin Dashboard

Manager Dashboard (Burger Bliss)

[!\[\]\(c638f83b9613146d28e16d6691150831_img.jpg\)](#) [LOGOUT](#)

[REGISTER NEW MANAGER](#) [VIEW ALL MANAGERS](#)

User Registration

First Name *

Last Name *

Email *

Confirm Email *

Password *

Confirm Password *

[REGISTER](#)


Figure 12 - Manager Dashboard

Manage Burger Bliss Menu

[!\[\]\(066fd00a512afefb0c0bec9ad5a7bf2c_img.jpg\)](#) [ADD NEW MENU ITEM](#)

[BURGERS](#) [APPETIZERS](#) [SALADS](#) [SIDES](#) [DESSERTS](#) [BEVERAGES](#)

Burgers



Classic Beef Burger

Charbroiled beef patty, lettuce, tomato, onion, pickles, and our signature burger sauce on a brioche bun


Category: Burgers
(Category ID: 8963039430973588658)

Item ID: 8963346759992475884

\$10.95

[UPDATE MENU ITEM](#)

[DELETE ITEM](#)



BBQ Bacon Cheeseburger

Beef patty topped with cheddar cheese, crispy bacon, BBQ sauce, and crispy onion strings on a brioche bun


Category: Burgers
(Category ID: 8963039430973588658)

Item ID: 8963346759992475885

\$12.95

[UPDATE MENU ITEM](#)

[DELETE ITEM](#)



Mushroom Swiss Burger

Beef patty topped with sautéed mushrooms, melted Swiss cheese, and garlic aioli on a brioche bun


Category: Burgers
(Category ID: 8963039430973588658)

Item ID: 8963346759992475886

\$11.95

[UPDATE MENU ITEM](#)

[DELETE ITEM](#)



Spicy Jalapeño Burger

Beef patty, pepper jack cheese, jalapeños, chipotle mayo, lettuce, and tomato on a brioche bun


Category: Burgers
(Category ID: 8963039430973588658)

Item ID: 8963346759992475887

\$11.95

[UPDATE MENU ITEM](#)

[DELETE ITEM](#)



Black Bean Veggie Burger

House made black bean patty, avocado, lettuce, tomato, onion, and chipotle mayo on a whole wheat bun

Category: Burgers
(Category ID: 8963039430973588658)

Item ID: 8963346759992475888

\$10.95

[UPDATE MENU ITEM](#)

[DELETE ITEM](#)

Figure 13 - Manager Menu View

Add New Menu Item

Item Restaurant ID
8963346759992475873

Item Name

Item Description

Item Price

Item Image Link
imagelink

CHOOSE IMAGE UPLOAD

Item Category

CANCEL ADD NEW MENU ITEM

Figure 14 - Add New Menu Item Dialog

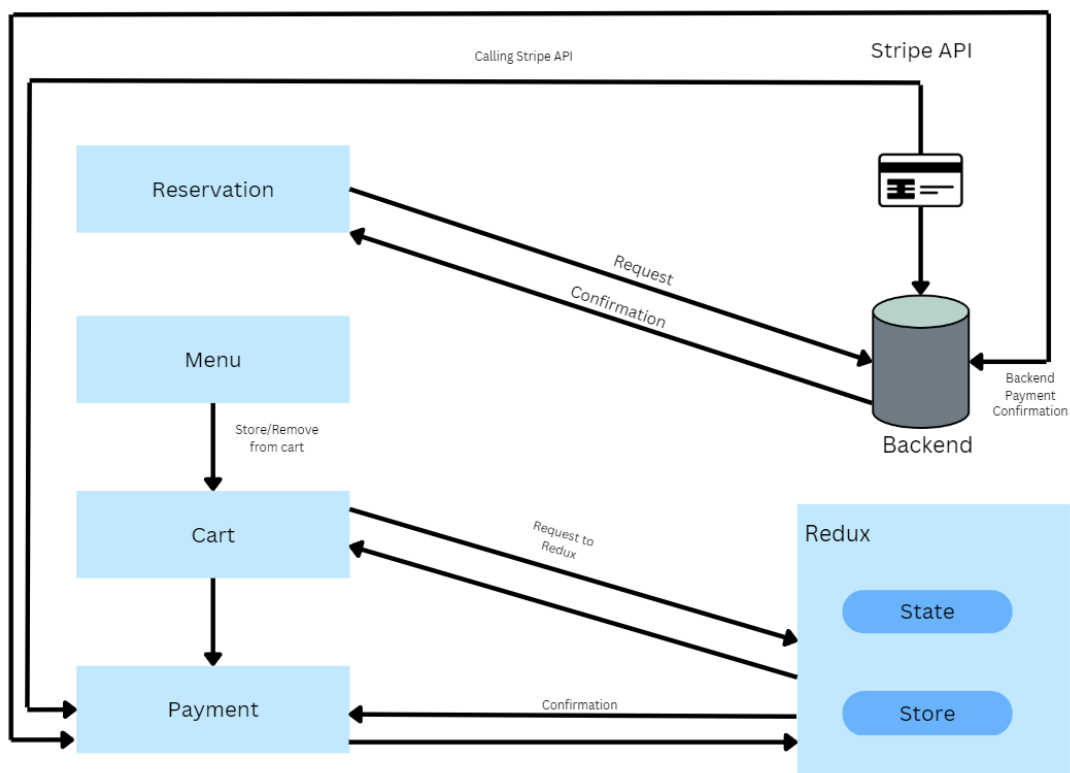


Figure 15 - Example Redux Data Flow

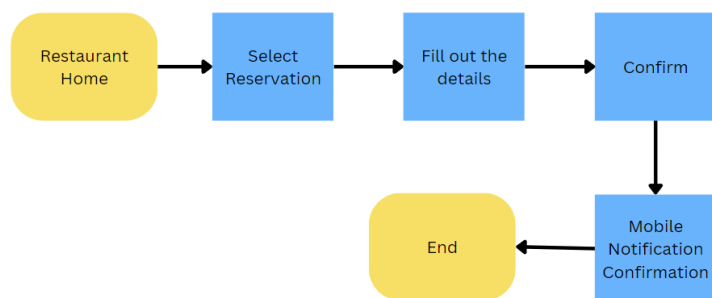


Figure 16 - Table Reservation Flow

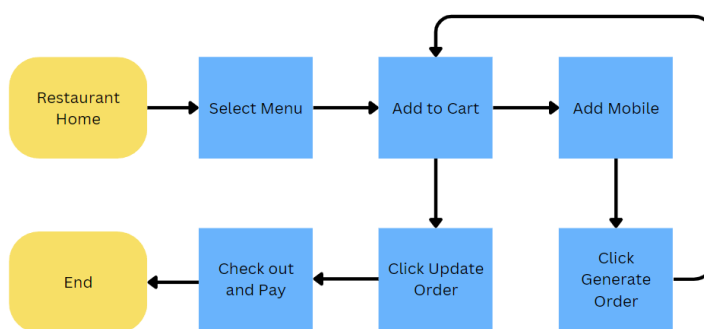


Figure 17 - Order Flow

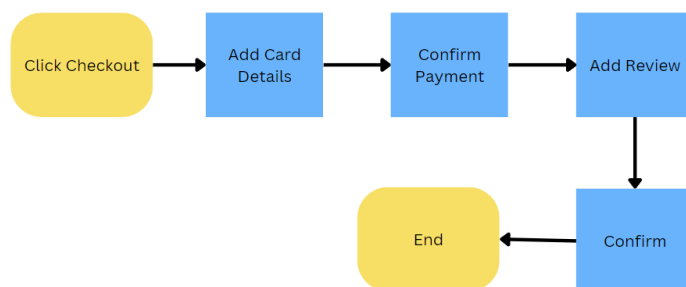


Figure 18 - Payment Flow

Middle-Tier Design

Class Diagram

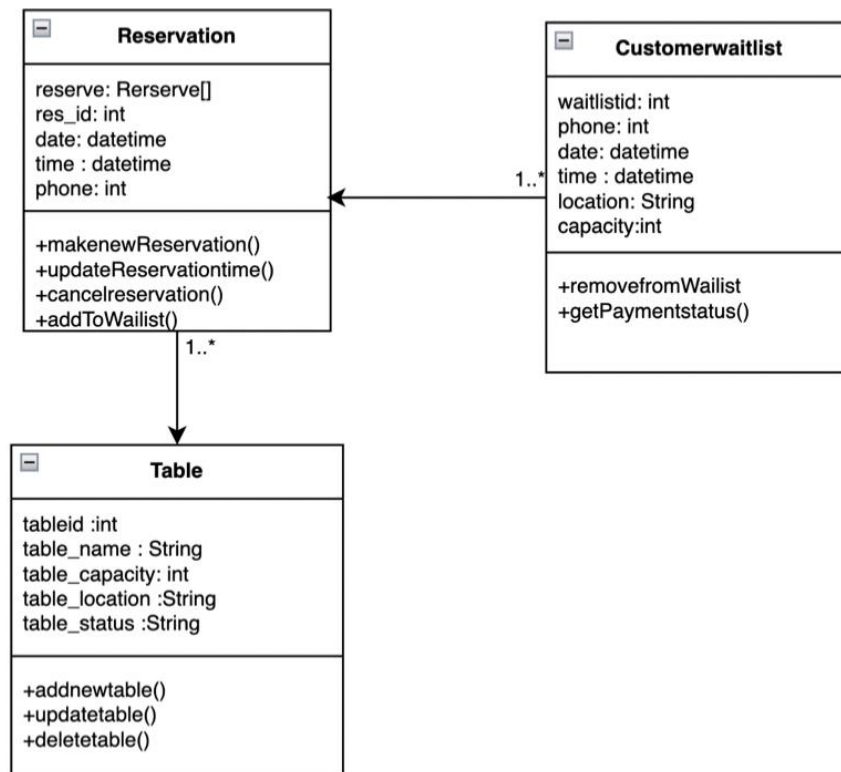


Figure 19 - Table Reservation Class Diagram

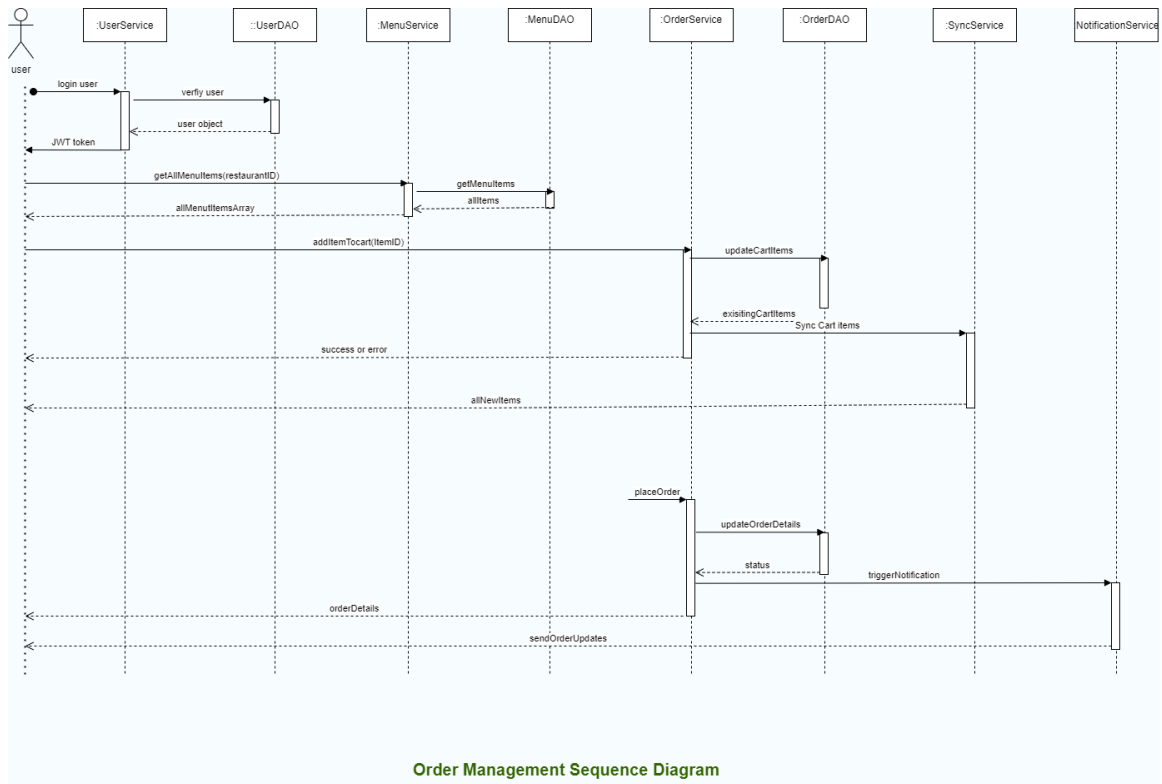


Figure 20 - Order Management Sequence Diagram

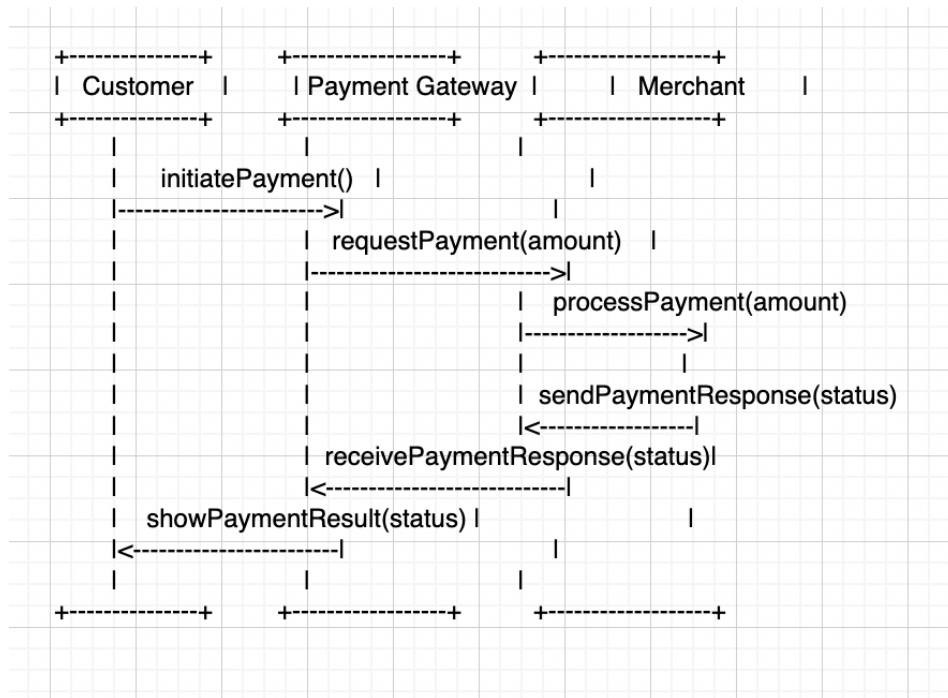


Figure 21 - Payment Service Sequence Diagram

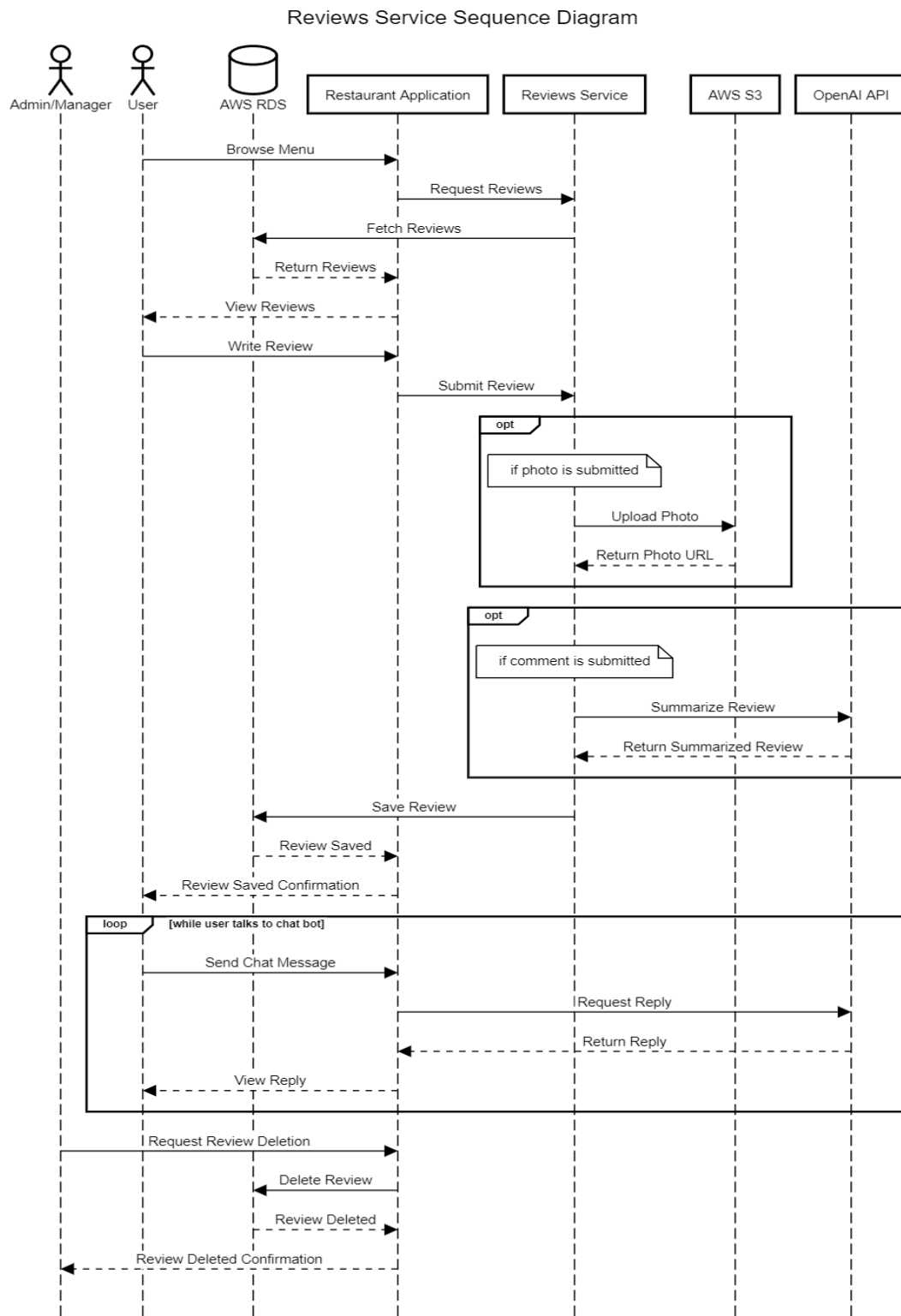


Figure 22 - Reviews Service Sequence Diagram

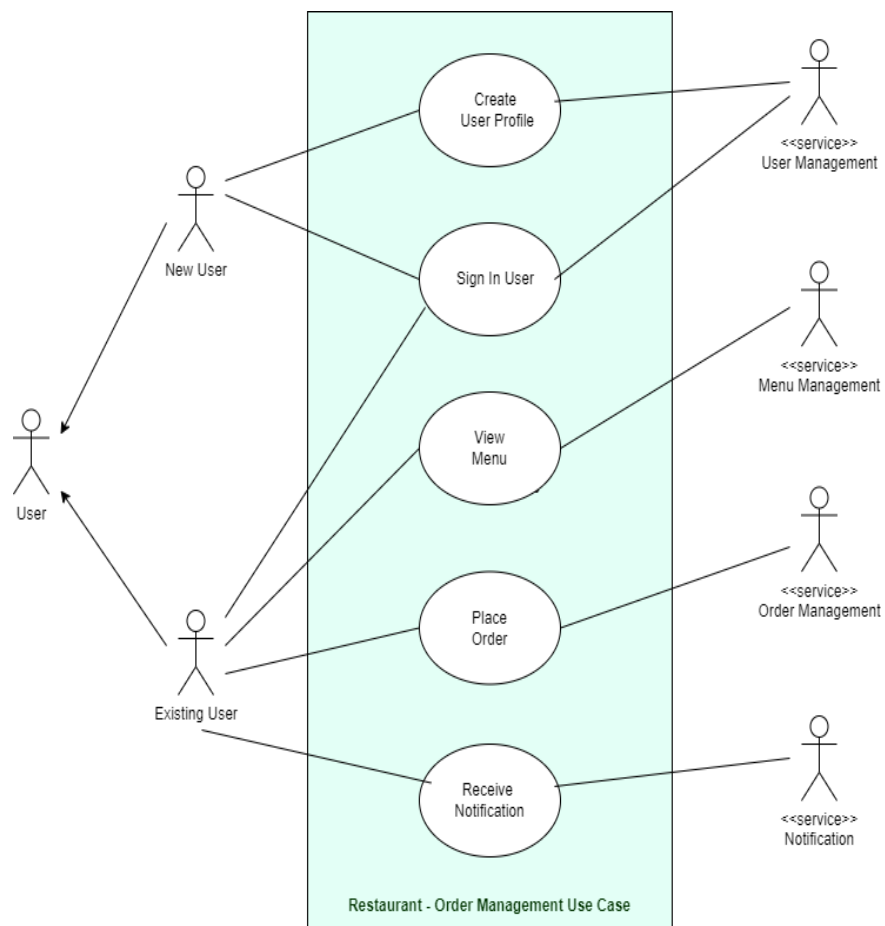


Figure 23 - Order Management Use Case

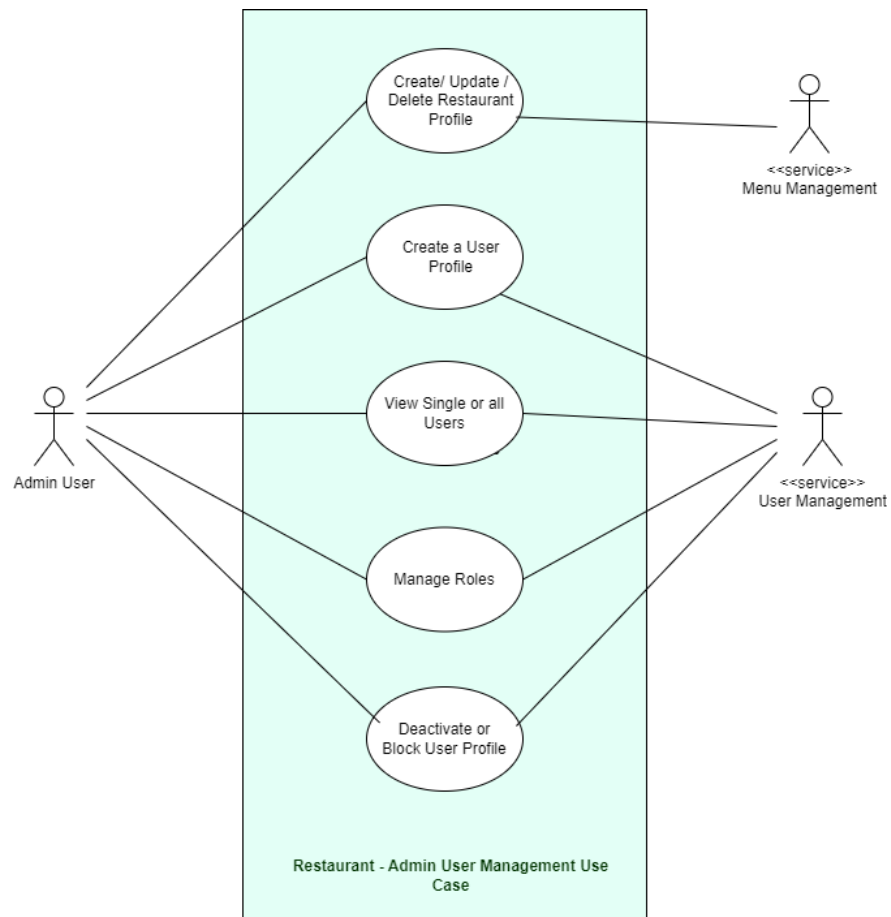


Figure 24 - Admin User Management Use Case

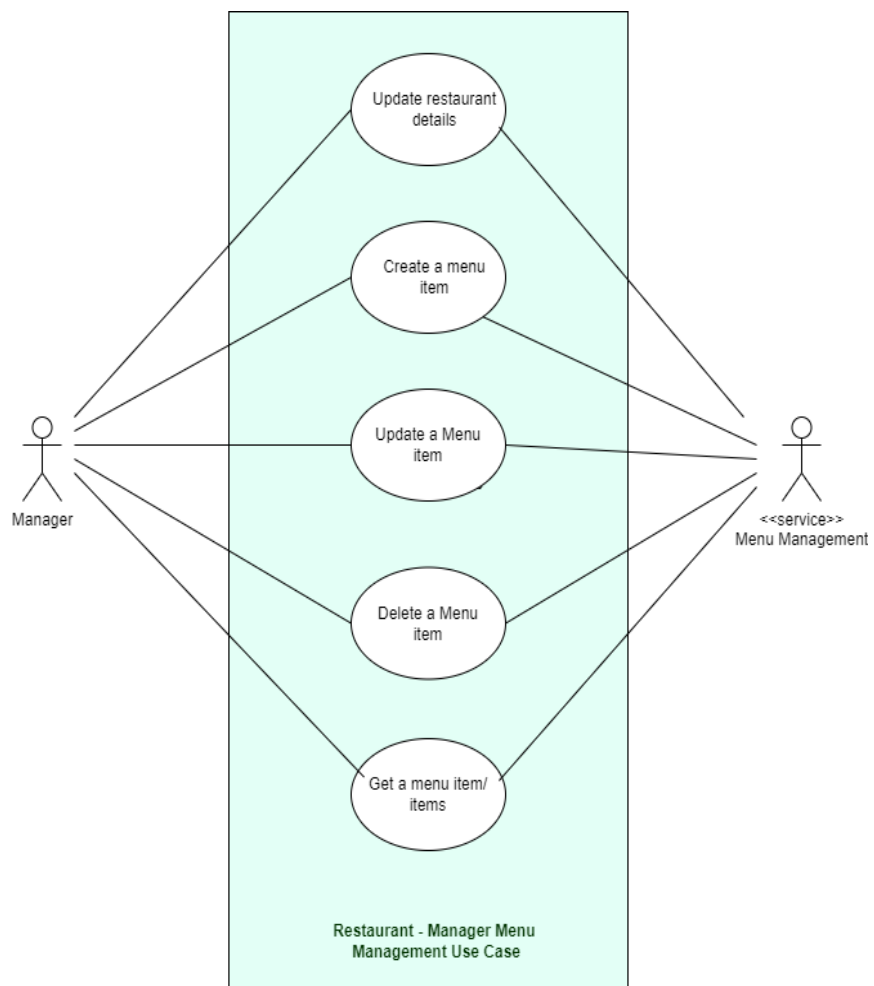


Figure 25 - Manager Menu Management Use Case

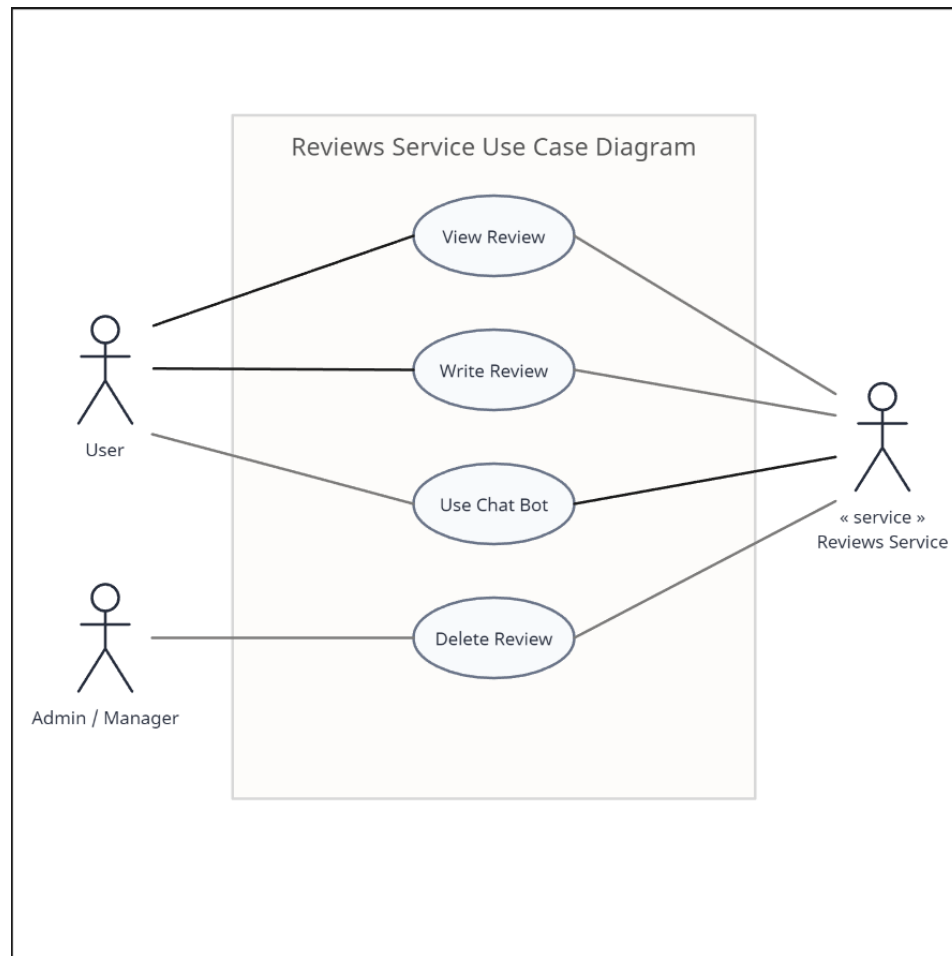


Figure 26 - Reviews Service Use Case

WEBHOOKS / Manage Webhook

Settings Recent Deliveries

✓ 6b98d0d8-c41d-11ec-86cd-eea5dad0c75c 2022-04-24 15:25:18 ...

Request Response **200** Redeliver ⌚ Completed in 0.35 seconds.

Headers

```
Request URL: http://jenkinsserver.buildyourownmind.com:8080/github-webhook/
Request method: POST
Accept: */*
content-type: application/json
User-Agent: GitHub-Hookshot/85c02f5
X-GitHub-Delivery: 6b98d0d8-c41d-11ec-86cd-eea5dad0c75c
X-GitHub-Event: push
X-GitHub-Hook-ID: 354933419
X-GitHub-Hook-Installation-Target-ID: 473721808
X-GitHub-Hook-Installation-Target-Type: repository
X-Hub-Signature: sha1=4c98406eecd04767b8be0f7df4e227e117bc4eb
X-Hub-Signature-256: sha256=f49a16a1490a4d60aa105f66d4f531b694f5ee18da5df218a72a0bb9affe87f7
```

Payload

```
{
  "ref": "refs/heads/master",
  "before": "77578d983f57306984e6c741b5b0ad5dd9435de6",
  "after": "d0ed030caba81719b78b9cabd128fd2555bf9228",
  "repository": {
```

Figure 27 - Jenkins pipeline to build the Docker image and deploy to EKS Cluster

Webhooks to auto-trigger Jenkins pipeline

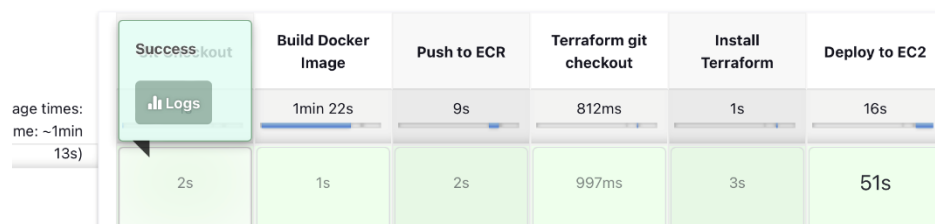


Figure 28 - Order Service CI/CD

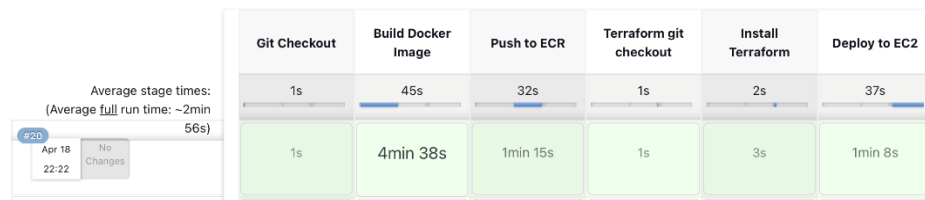


Figure 29 - Reservation Service CI/CD

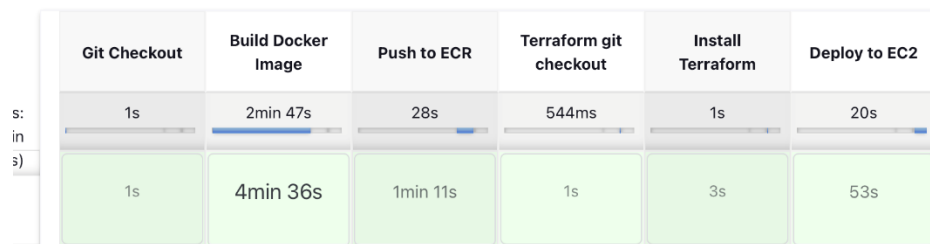
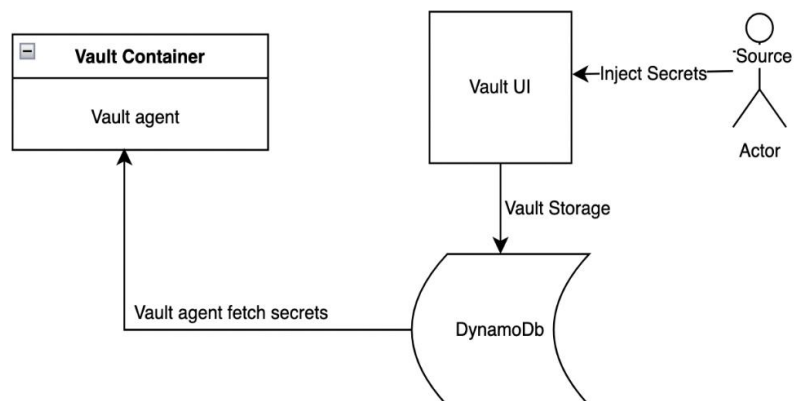


Figure 30 - Payment Service CI/CD

Vault Storage- Dynamo db .

All the Secrets will be injected into the Vault server using Vault UI .
Application service will pull the secrets from Vault using the Vault agent running on the container in K8s pods .



Vault conceptual Diagram

Figure 31 - Vault for Secrets Management

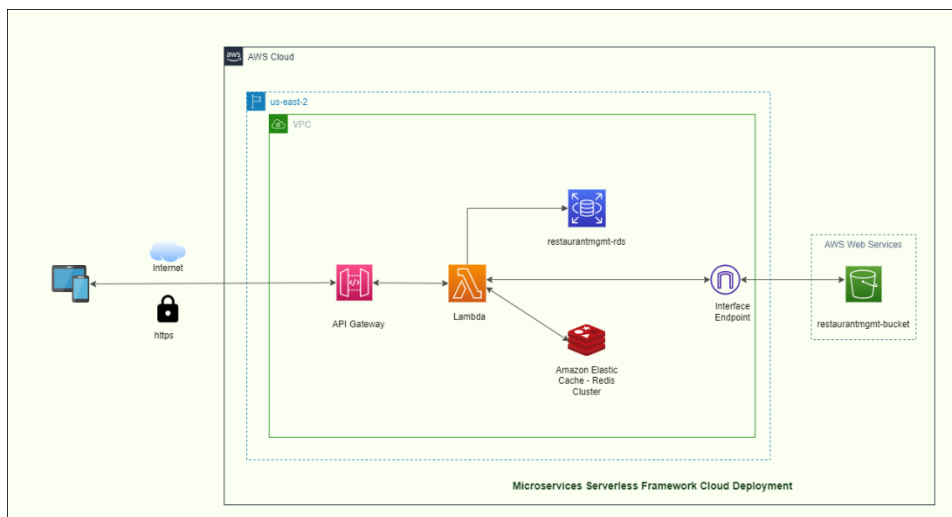


Figure 32 - Serverless Framework Cloud Deployment (User, Menu Management)

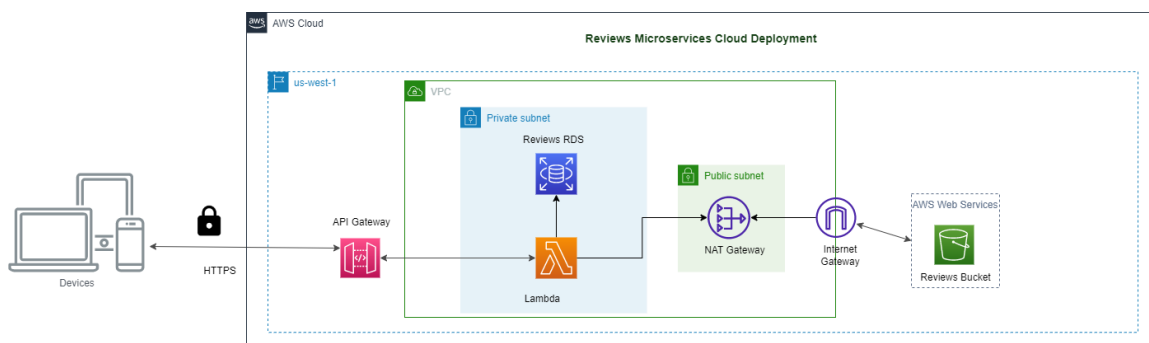


Figure 33 - Serverless Framework Cloud Deployment (Reviews)

Data-Tier Design

Database Schemas

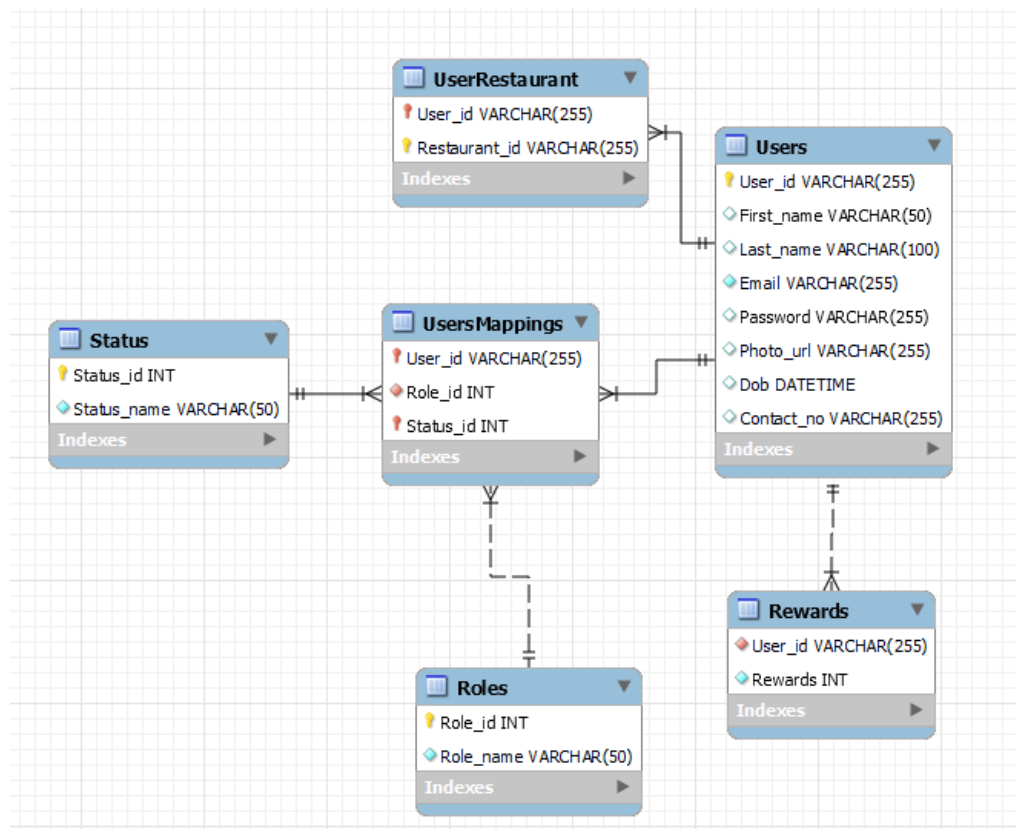


Figure 34 - User Management

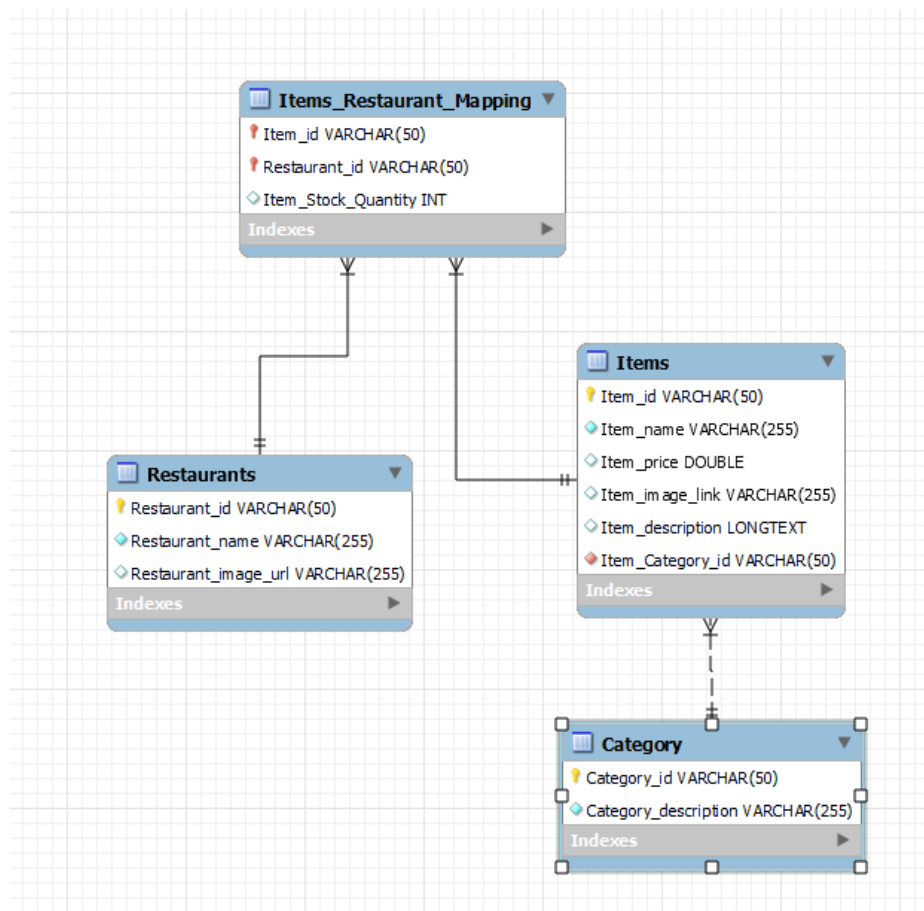


Figure 35 - Menu Management

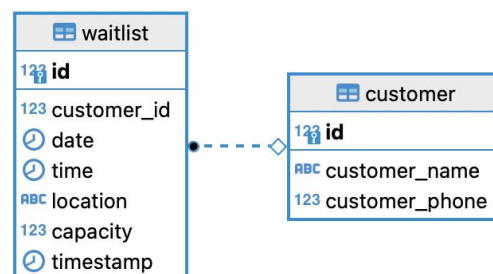


Figure 36 - Waitlist

Database/Entity Relationship

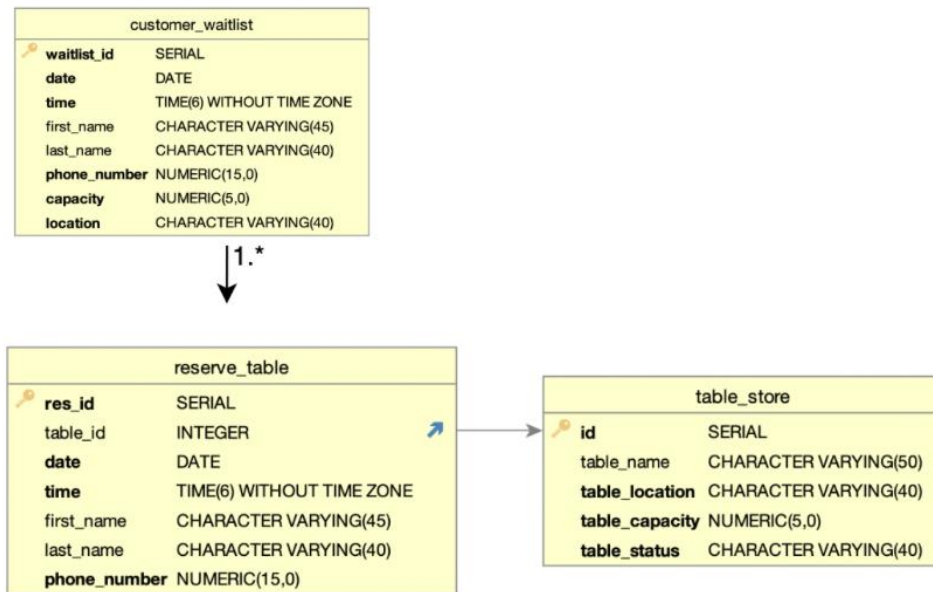


Figure 37 - Table Reservation

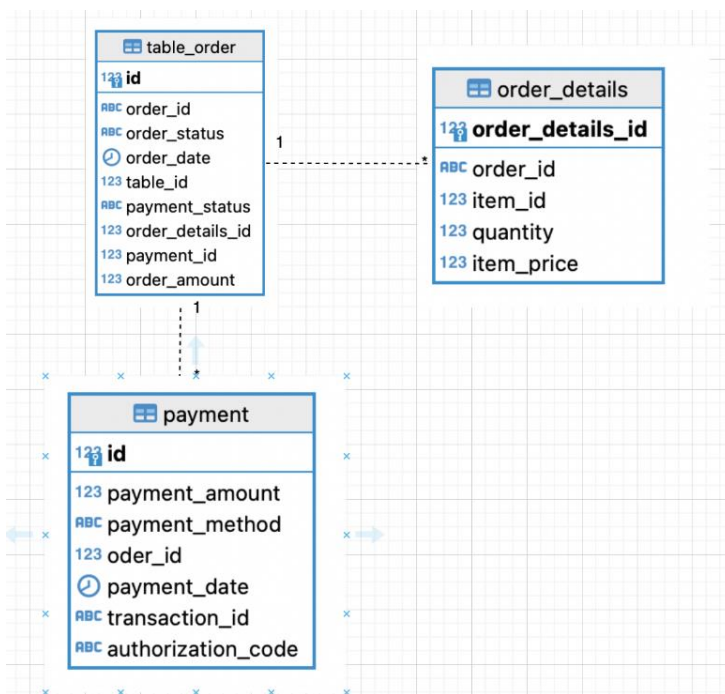


Figure 38 – Order and Payment

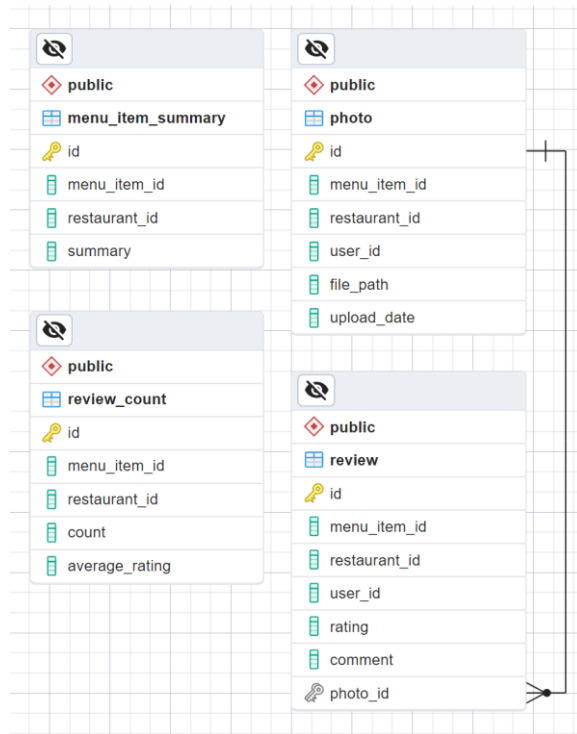


Figure 39 - Reviews Service ERD

Chapter 5. Project Implementation

Client Implementation

- Customer Features
 - Order food
 - Waitlist / Reserve table
 - Write / View reviews after order confirmation (“verified” reviews)
 - Upload / View photos for menu items
 - Real-time menus add and edit updates using Redis streams
 - Adding items to cart and updating order without refreshing
 - Talk with a chat bot regarding food
 - Secure Payment using Stripe API
- Manager Features
 - Manage restaurant menu inventory
 - Manager users can add, edit and delete menu items and reviews only from their assigned restaurant
 - Create other manager users for the restaurant they are assigned to
 - Change user status to active or inactive
 - Create new menu item categories
 - Manage orders
 - Adding extra tables for reservations
 - Managers will be automatically redirected to their restaurant upon login
- Admin Features
 - Manage and view all restaurants
 - Create new restaurants
 - Admin can create other admin and/or manager users
 - Change user status to active or inactive
 - Has all permissions that managers have
- User Interface conditional rendering for admin, managers, and users

- Cart, restaurant, reviews, and menu item data is stored in the application state managed by Redux. This allows for changes to be made dynamically without page refresh
- Chat bot message history is also stored in Redux so that if the user leaves the page, the message history is preserved

Middle-Tier Implementation

- Created cloud environment for menu and user management services using Lambda, S3, and RDS MySQL
- Created cloud environment for reviews service using Lambda, S3, and RDS PostgreSQL
- Deployed and configured Elastic Cache – Redis Cluster on AWS
- Configured CI/CD pipeline and deployed reviews, menu and user management services to AWS Lambda using Zappa
- Integrated Redis Streams in menu management service
- Utilized OpenAI API for review summarization and chat bot
 - Each individual menu item has a review summary of up to 7 (can be changed) positive and negative points regarding the item
 - The chat bot can only take in food related inquiries and does not answer unrelated questions
- Utilized Stripe API for secure payments

API Endpoints

Menu and User Management Services

- /restaurants (GET)
 - Returns an array of all restaurants
- /restaurant (POST)
 - Creates a new restaurant and returns a GUID

- /item?itemId=<MenuItemID> (GET)
 - Returns details about a specified menu item by menu item ID
- /item?itemId=<MenuItemID> (DELETE)
 - Deletes a specified menu item by menu item ID
- /categories (GET)
 - Returns an array of all menu item categories
- /category (POST)
 - Creates a new menu item category and returns category name and GUID
- /item (POST)
 - Creates a new menu item and returns all the details related to the menu item and GUID
- /items?restaurantID=<restaurantID> (GET)
 - Returns all the menu items of a specified restaurant ID
- /auth (POST)
 - Logins in users and returns a JWT token
- /users (GET)
 - Returns all users, admin only access
- /user/status (PUT)
 - Updates a user's status (active or inactive)
- /user (POST)
 - Creates a new user and returns all the details related to the user

Order, Payment, and Table Reservation Services

- /addTable
 - Adds new table to a restaurant
- /newreservation
 - Creates a new table reservation

- /addToWaitlist
 - Adds the customer to the waitlist
- /getCustomerReservation
 - Get the customer reservation info
- /getTableInfo
 - Get all the table information including availability
- /assignTableWaitlistCustomer
 - Assign the table to the customers in the waitlist
- /order/createOrder
 - Create new order
- /order/updateOrder
 - Update order
- /order/getOrderInfo/<order_id>
 - Get order info
- /order/getOrderItemsInfo/ <order_id>
 - Get the order details info that includes all the items in the order
- /create-payment-intent
 - Create new payment for the open order

Reviews Service

- /reviews/submit (POST)
 - Submit review
 - If comment is provided, calls OpenAI API to summarize review comments for specified menu item
 - If photo is provided, uploads to AWS S3 and associates the photo ID with the review ID
- /reviews/photos/<restaurantID>/<menuItemID> (GET)
 - Returns all the photos of a specific menu item ID of a specified restaurant ID

- /reviews/menu_item_details/<restaurantID> (GET)
 - Returns the average rating, total review count, and review summary of every existing menu item ID for the specified restaurant ID
- /reviews/delete/<restaurantID>/<menuItemID> (DELETE)
 - Deletes all reviews associated with a specified menu item ID of a specified restaurant ID
- /reviews/upload-photo (POST)
 - Upload photo (6 MB limit due to AWS Lambda payload size restriction)
- /chat (POST)
 - Chat bot, provides back and forth conversation using OpenAI API using the GPT-3.5-Turbo LLM model. While the GPT-4 LLM model is much better, the cost per token is significantly more.

Data-Tier Implementation

- Configure MySQL and PostgreSQL on AWS Cloud using AWS RDS MySQL and AWS RDS PostgreSQL, respectively
- Wrote initial database scripts for menu and user management services
- Wrote and executed stored procedures for develop endpoints for menu and user management services

A. Menu Management

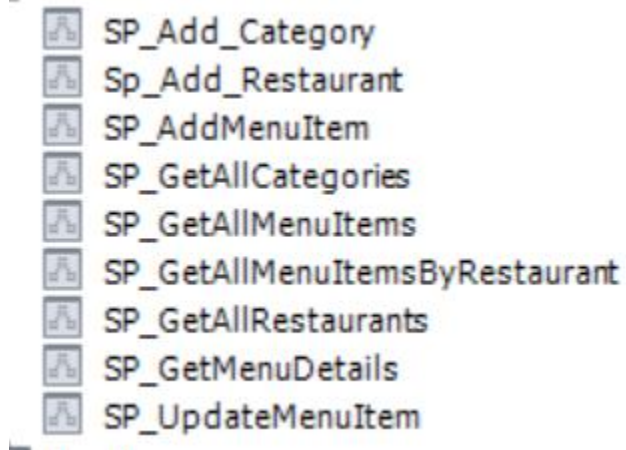


Figure 40 - Menu Management Stored Procedures

B. User Management

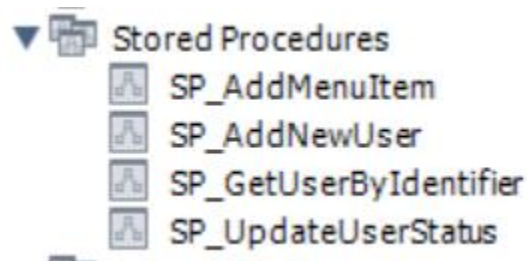


Figure 41 - User Management Stored Procedures

- Reviews Service

- Consists of four tables:

- menu_item_summary
 - Stores the review summary of each menu item
 - photo
 - Stores all the information in regard to a photo uploaded with a specific review

- review
 - Stores all the information in regards to a review, contains a foreign key which is the primary key of the 'photo' table
- review_count
 - Stores the number of reviews and the average rating of each menu item. Design decision to calculate average ratings and storing them instead of calculating them at runtime

Chapter 6. Testing and Verification

Testing Tools

- Postman (main testing tool)
- Redux DevTools
- Chrome DevTools

Environment

- Local: The services will run on localhost
- Development: Some services such as user management, menu management, and reviews will be deployed and configured using AWS Lambda and Zappa

Test Cases

Manual testing was done on all of tests listed successfully, API endpoints were tested on Postman successfully

- User Management Service
 - User can register and login
 - User with 'admin' role can deactivate any account
 - User cannot deactivate their own account
 - Users who are designated as 'inactive' cannot login until reactivated
 - User with a 'admin' role can view all user accounts and create new user accounts of admin or manager role
 - User with a 'manager' role can only create other user accounts of role 'manager'
 - User with a 'manager' role can only access resources within their assigned restaurant
 - User with a 'admin' role can access all resources
 - Users can search for users in the search bar
 - Only admin roles can log into the admin portal, while admins can login from any portal
 - Managers are automatically redirected to their restaurant upon login

- Admins can view all restaurants
- Menu Management Service
 - User with 'admin' role can create, edit, and delete menu items at any restaurant
 - User with 'manager' role can create, edit, and delete menu items only at their assigned restaurant
 - Deleting a menu item has a confirmation message to confirm deletion
 - Users with a 'admin' role can create restaurants, restaurant name and image are required
 - Adding and editing menu items are updated in real-time (e.g., if a manager marks a menu item as out of stock, all users viewing the menu item will see it become out of stock without refresh)
 - User can create a menu item category or use an existing category
 - Adding an item requires all fields to be filled out with the exception of the image field
 - Editing an item does not require all fields to be filled
 - Users can search for items in the search bar
 - Users with a 'admin' or 'manager' role
 - All actions have a success or failure message
- Reviews Service
 - User can write a review each individual item
 - User can only review individual items after their order has been paid for and confirmed
 - User can submit a review for any number of items in their confirmed order
 - Only the star rating is required for a review, comment and photo is optional
 - Review comments cannot exceed 50 characters
 - Submit review button is replaced by a circular loading icon while it is being submitted and user cannot click out of the dialog box
 - Upon successfully submission of review, confirmation message is shown

- User can hover over the star rating to view the average rating
 - User can view the reviews and ratings for individual items while ordering
 - User submitted text reviews are summarized with up to 7 positive points and 7 negative points that appear underneath each menu item
 - Positive reviews are highlighted in green while negative reviews are highlighted in red
 - User can view photos for each individual menu item
 - User can chat about any food related inquiries with a chat bot, chat bot will not respond to non-food related inquiries
 - When a menu item is deleted, its associated review data is also deleted
- Email Notification Service
 - User should get an invoice in PDF format after successful payment
- Table Reservation Service
 - Unit testing
 - Table store CRUD
 - User can make new reservations
 - User can add themselves into waitlist
 - Confirmation and waitlist notification
 - Twilio SMS API integration
 - Database is updated successfully
 - All endpoints are secured with JWT token
- Order Service
 - Users with 'customer roles' any roles should be able to add and update items to the cart
 - Users with 'customer roles' should be able to add more items to already ordered items without refreshing
 - Users with 'customer roles' should be able to manipulate carts that have not placed orders

- Users with 'customer roles' should not be able to manipulate already ordered times in cart
- Users with 'customer roles' should be able to add review after payment
- Users with 'customer roles' should be able to do payment with card details seamlessly
- Users with 'customer roles' should be able to reserve a table or add themselves to waitlist
- Users with 'customer roles' should get notification if they are bumped up with a table from waitlist
- Reservation should give alerts when form is left empty
- Reservation should not work without all the details.
- Cart should show up a little when hovered over the right side of the menu page
- Cart should not be visible on other pages
- Cart should open/close fully when clicked
- Items should be added to cart from menu without refreshing
- Items should be moved to the ordered items list from the cart
- When the order is generated, the cart should flush
- The checkout button should work before generating order
- Phone number should be optional when ordering

Chapter 7. Performance and Benchmarks

These tests are designed to evaluate the performance results in the presence of a high user load when accessing the cloud-based restaurant management system in production. The end goal is to ensure a highly scalable, available, and resilient system. Performance testing can help to remediate and fine-tune the overall system.

Entry Criteria

The enterprise restaurant system fulfills all the functional requirements of the system and is stable. The entire application must be deployed on the cloud.

Exit Criteria

The essential test cases are executed, and results are documented. The performance bottlenecks are properly documented. No critical defects are outstanding.

Environmental Requirements

The test environment backend database significantly replicates the production level database (version, cloud provider and database type). The few microservices like user, menu and reviews are deployed using AWS Lambda and API Gateway. The Redis cluster is deployed using AWS Elastic Cache.

System Environment Requirements

The system undergoing tests consists of three parts: front-end, set of back-end microservices and back-end database. The front end is deployed on AWS Amplify. The set of microservices are deployed on the Kubernetes cluster and serverless framework. The preliminary tests for a few microservices (user and menu management) are done in a development environment with the same database server. The Linux operating system and Linux container will be used. All front-end user interface and set of microservices are exposed via HTTP/HTTPS. The database is exposed using the TCP/IP protocol.

Mandatory Test Cases

| | Test Case | Category |
|----|---|----------------------------|
| 1. | Page-load, within 3 seconds | UI |
| 2. | Verify response time of an API for all CRUD operations on user and menu is less 500 milliseconds | Cloud and Backend REST API |
| 3. | Verify JWT token expires after 2 hours | Backend-REST API |
| 4. | Determine parallel registration of multiple users using API (Deployed on API Gateway and Lambda function) | Cloud and Backend REST API |
| 5. | Images are rendered less than 500 milliseconds | UI |
| 6. | JSON List of all menu items is retrieved from API having response time less than 1 second | Cloud and Backend REST API |
| 7. | User cart-items are coordinated across all its session within 1 second | Cloud and Backend REST API |

Key Performance Indicators and Benchmarking Criteria

- Load Time: time it takes for the web application to fully load and become interactive
 - Review images for each menu item are not loaded until requested
 - Reviews submitted with comments may sometimes take a few seconds due to the limitations of the OpenAI API
 - The first load of the menu sometimes takes a few seconds but after loaded, performance increases significantly

- Responsiveness: how quickly the web application responds to user input and actions
 - Once loaded, all user input and actions are nearly instantaneous
- Scalability: how well the web application can handle increasing number of users or requests without performance degradation
 - With cloud deployment, resources are scaled according to demand
- Browser Compatibility: evaluate web application's performance across different web browsers and platforms
 - Pages and components are scaled according to viewport, whether if the web application is accessed on a mobile device, tablet, or desktop.
- Accessibility: verify that the web application is accessible to users with disabilities
- SEO performance: analyze web application's search engine optimization performance

Performance testing was done using Google PageSpeed. For the main three consumer-facing pages in the application, it can be observed that the mobile version of the application is not as optimized as the desktop version especially for the menu page. The menu page has an excessive DOM size with approximately 1,900 elements. While the mobile version is responsive, it is not as optimized, therefore, a future improvement would be to optimize the mobile version of the application. For mobile testing an Emulated Motorola Moto G Power is used which is an older and slower device therefore it may affect results, but regardless the web application must perform well for all devices. For the other key performance indicators such as accessibility, best practices, and SEO performance all met and exceeded expectations.

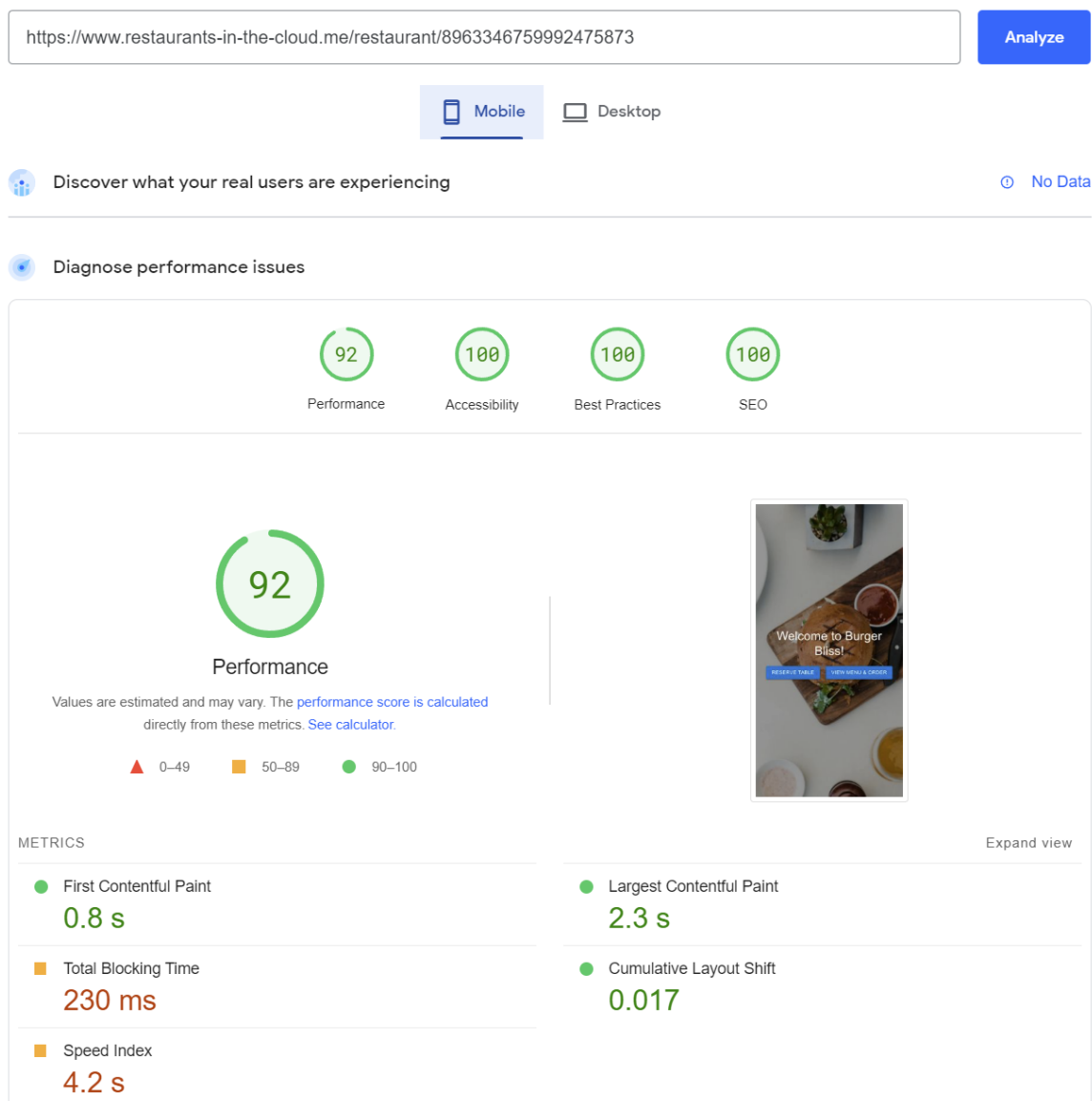


Figure 42 - PageSpeed, Home Page Mobile Results

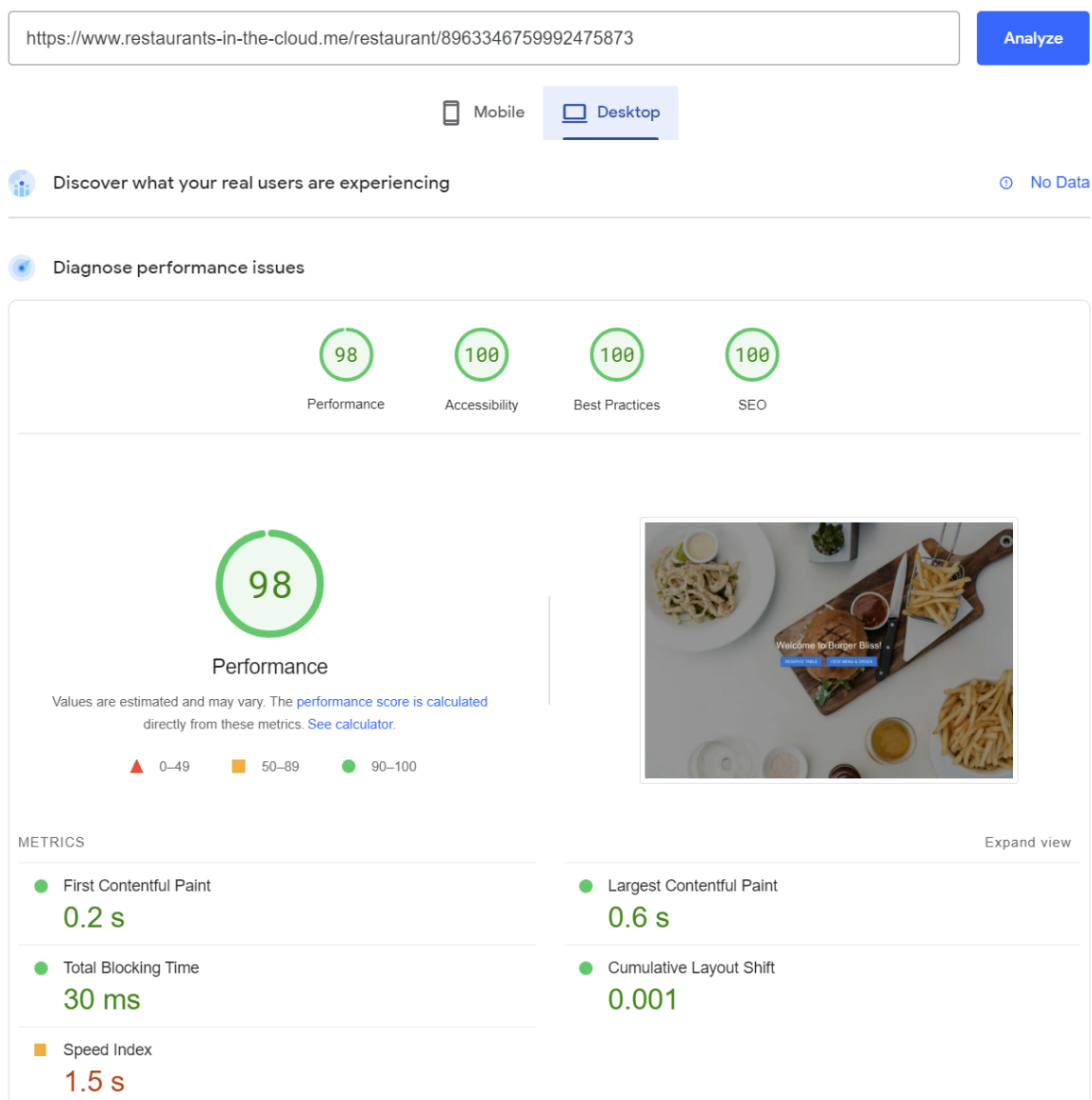


Figure 43 - PageSpeed, Home Page Desktop Results

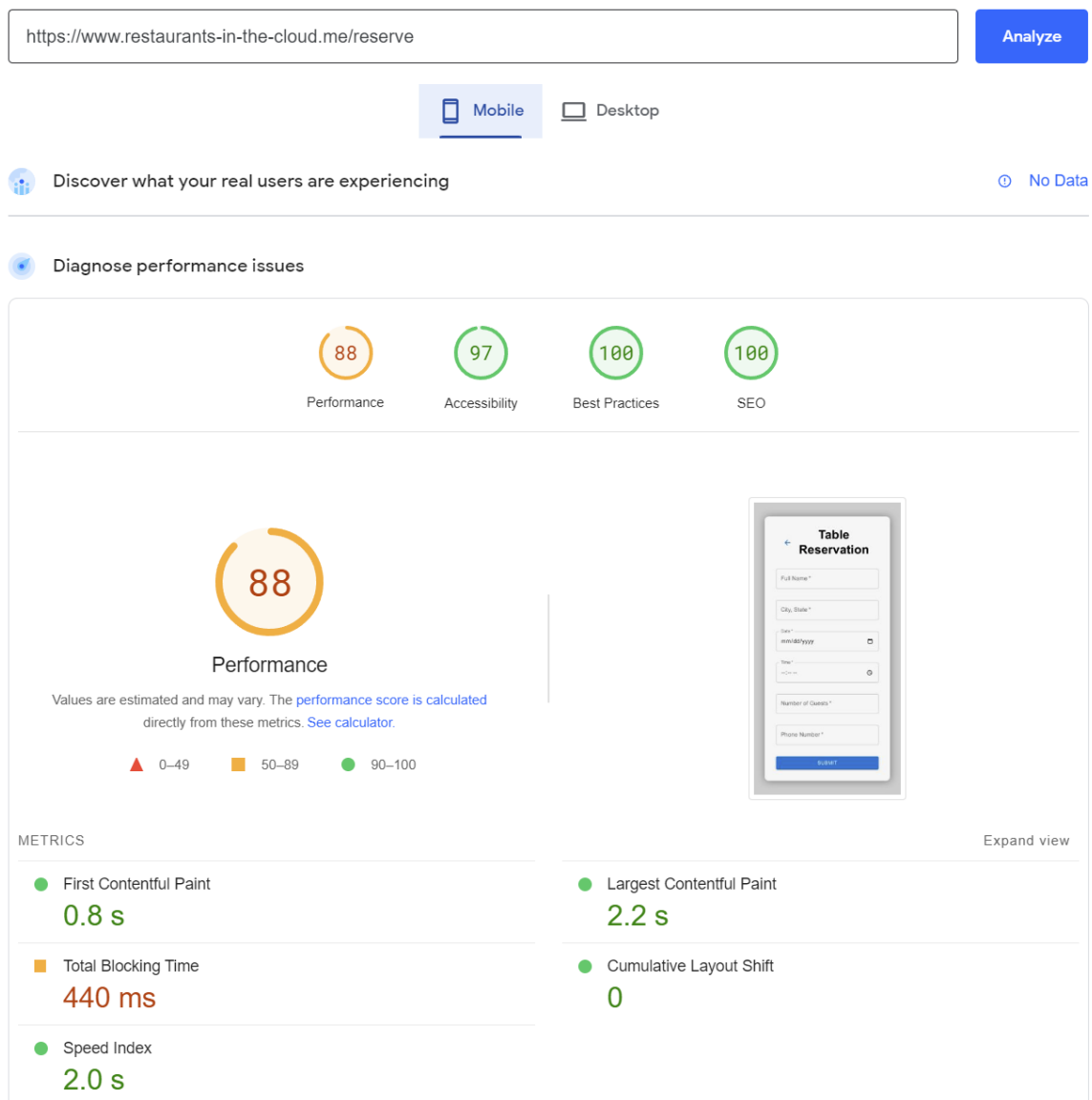


Figure 44 - PageSpeed, Table Reservation Mobile Results

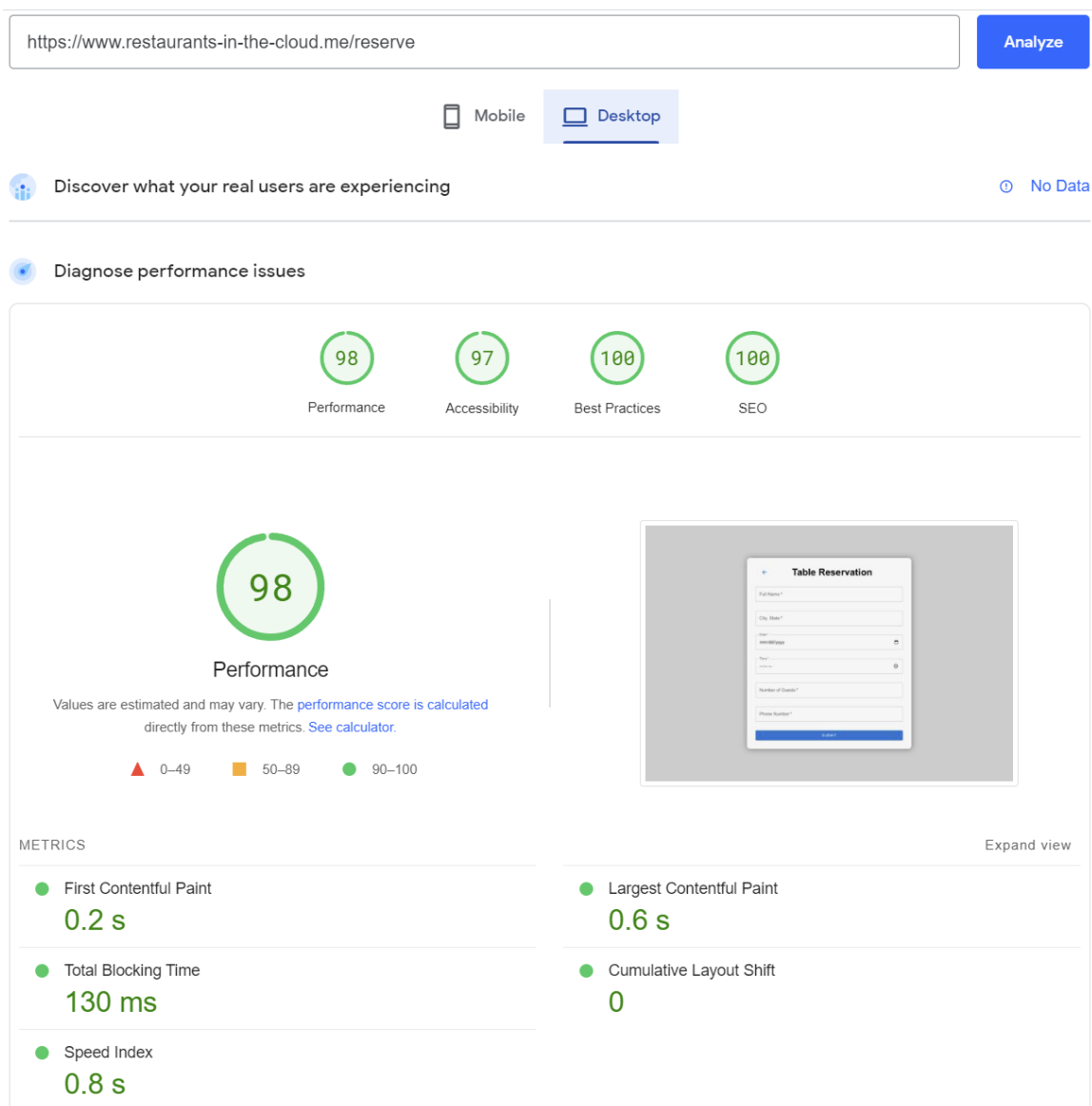


Figure 45 - PageSpeed, Table Reservation Desktop Results

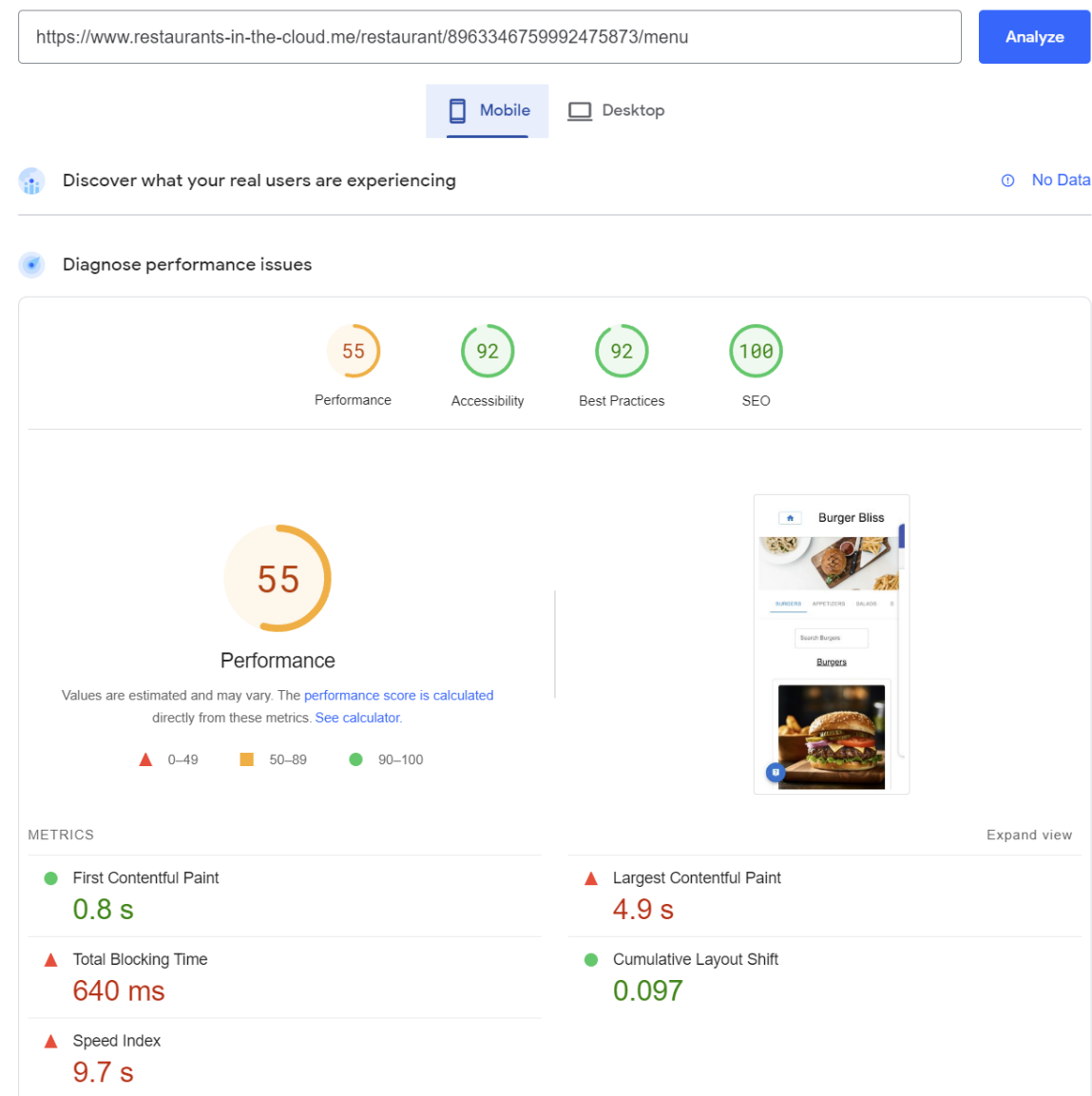


Figure 46 - PageSpeed, Menu Page Mobile Results

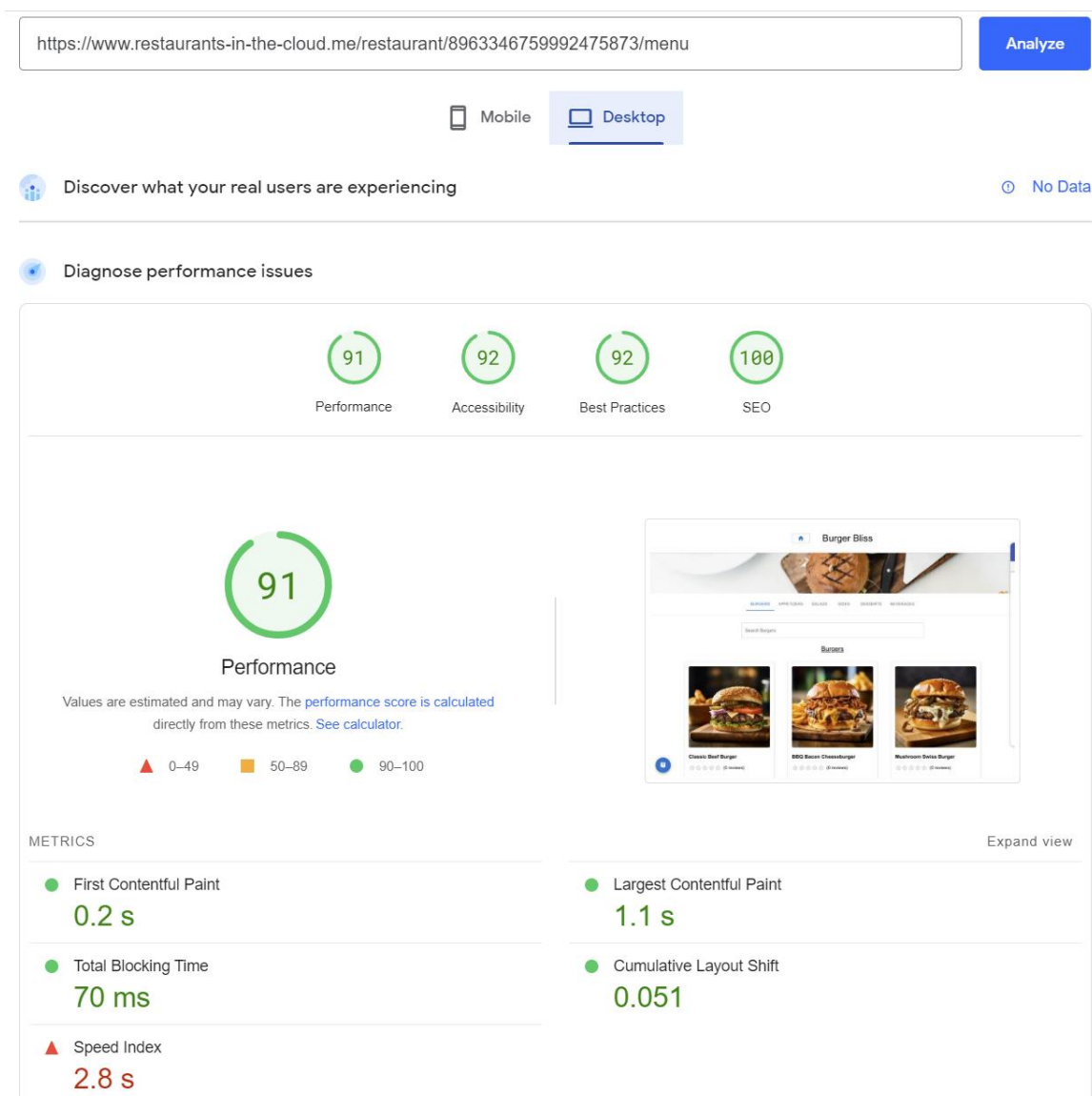


Figure 47 - PageSpeed, Menu Page Desktop Results

Chapter 8. Deployment, Operations, and Maintenance

Deployment Plan

- Continuous Integration and Continuous Deployment (CI / CD): This strategy involves automating the building, testing, and deploying processes to ensure that changes are quickly and safely deployed to production
- User and Menu Management, and Reviews services will be deployment in development and production environments on AWS Serverless Lambda
- Zappa is used to build and deploy user and menu management, and reviews on AWS Lambda and API Gateway
- AWS RDS, and Lambda are configured in the same cloud VPC
- S3 buckets are connected to VPC using Internet Gateway endpoint
- Lambda functions are granted internet access through a NAT gateway in a public subnet within a VPC

Operational Plan

- Monitoring: AWS CloudWatch is used to collect and visualize real-time logs, metrics, resource utilization and data in dashboard for service monitoring. It can also be useful to debug errors with API calls
- Database snapshots and backups are periodically taken to ensure all data is safe and secure
- Scalability: AWS Lambda is auto scalable and can handle multiple workloads asynchronously
- Backup and Disaster Recovery: Lambda functions and API Gateway can be quickly restarted in the event of unexpected downtime or system failure

Maintenance Plan

- Regular Updates: Python libraries like Flask, Flask-CORS, etc., and MySQL and PostgreSQL databases should be updated regularly to address bugs, security vulnerabilities, and other issues

- Performance Tuning: The system should be periodically tuned for optimal performance. Ensure performance tests are written and updated regularly
- Data Management: Data should be regularly backed up, archived, and optimized for efficient storage and retrieval
- Bug Fixes: Bugs and other issue should be reported in a timely manner using a project management tool such as Jira

Chapter 9. Summary, Conclusions, and Recommendations

Summary

Due to ever-changing consumer preferences, this project proposed the development of an all-in-one restaurant management Software as a Service (SaaS) to improve the overall consumer experience and restaurant service quality. By exploring and discussing the advantages and disadvantages of the many current solutions in the market, this project seeks to improve on those solutions. The application that was developed allows consumers to order food, reserve tables, waitlist, talk to a chat bot about food, write and view reviews, and upload photos. Reviews are summarized using artificial intelligence and verified meaning reviews can only be written after order confirmation. Admins can create new restaurants with ease and oversee all operations. Managers can manage various aspects of their restaurant such as managing the menu, orders, and employees.

Conclusions

In conclusion, while this project did not provide a completely novel solution, it aimed to improve on various aspects of current solutions. It establishes a good foundation for further extension and research. Due to the restaurant management system being cloud-based, the hardware requirements for the application are very minimal for restaurants. The application is accessible and responsive for different platforms and devices allowing for both consumers and employees to utilize it anytime and anywhere.

Recommendations for Further Research

- JWT authentication at granular level for each endpoint
- Centralized token authentication service
- Configure Lambda to return stream output

- Research best queue service (RabbitMQ or Kafka)
- Integrate restaurant analytics with the chat bot to provide personalized suggestions based on the restaurant. For example, a user could ask what the best menu item is at this specific restaurant and the chat bot would be able to answer. Additionally, further integrate artificial intelligence into reviews by analyzing text reviews, automatically assigning numeric ratings, and look for any correlations between what a user chooses for a numeric rating versus what their text review indicates and adjust reviews accordingly.
- Cost analysis of running this web application for an extended period then compare it to other current solutions and devise a pricing structure
- Explore possible expansions into other business sectors aside from restaurants such as grocery shopping. Turning this web application into a more generalized solution
- Investigate image compression and caching in order to reduce network payload sizes and increase page load times, especially for mobile

References

- [1] J. F. Lippert, M. B. Furnari, and C. W. Kriebel, "The impact of the COVID-19 pandemic on occupational stress in restaurant work: A qualitative study," *International Journal of Environmental Research and Public Health*, vol. 18, no. 19, Oct. 2021.
- [2] [5] D. Klein, "Restaurants in 2021: Takeout and delivery is now essential to customers," QSR magazine, <https://www.qsrmagazine.com/consumer-trends/restaurants-2021-takeout-and-delivery-now-essential-customers>.
- [3] D. Curry, "Food Delivery App Revenue and Usage Statistics (2023)," Business of Apps, <https://www.businessofapps.com/data/food-delivery-app-market/>.
- [4] J. Chick, K. Duffy, and S. Seligsohn, "The restaurant of the future arrives ahead of schedule - Deloitte," *Deloitte US*, 2020. [Online]. Available: <https://www2.deloitte.com/content/dam/Deloitte/us/Documents/consumer-business/us-cb-the-restaurant-of-the-future-arrives-ahead-of-schedule.pdf>.
- [5] A. Narang, "The Restaurant Industry Outlook Report 2023 Edition," Toast, <https://d2c9w5yn32a2ju.cloudfront.net/assets/Toast-Restaurant-Industry-Outlook-Report-2023-Edition.pdf>.
- [6] "2023 State of the Restaurant Industry," National Restaurant Association, <https://restaurant.org/research-and-media/research/research-reports/state-of-the-industry/>.
- [7] R. Ruggless, "Pandemic intensifies restaurant wait-time concerns, Bluedot survey finds," *Nation's Restaurant News*, 10-Feb-2021. [Online]. Available: <https://www.nrn.com/family-dining/pandemic-intensifies-restaurant-wait-time-concerns-bluedot-survey-finds>.
- [8] "2021 State of the Restaurant Industry Mid-Year Update," National Restaurant Association, <https://restaurant.org/research-and-media/research/research-reports/2021-state-of-the-restaurant-industry-mid-year-update/>.
- [9] M. A. Ali, D. H. Ting, M. Ahmad-ur-Rahman, S. Ali, F. Shear, and M. Mazhar, "Effect of online reviews and crowd cues on restaurant choice of customer: Moderating role of gender and perceived crowding," *Frontiers in Psychology*, vol. 12, Dec. 2021.
- [10] Y. Liu, X. Huang, A. An and X. Yu, "Modeling and Predicting the Helpfulness of Online Reviews," *2008 Eighth IEEE International Conference on Data Mining*, 2008, pp. 443-452, doi: 10.1109/ICDM.2008.94.

- [11] A. Sylte, "Why negative reviews could have more of an impact on some of the most important customers," *Colorado State University | College of Business*, 21-Mar-2022. [Online]. Available: <https://biz.source.colostate.edu/negative-online-reviews-impact-study/>.
- [12] G. Moran, "Gen Z and millennials 2020 report," *Drapers*, 09-Sep-2020. [Online]. Available: <https://www.drapersonline.com/insight/drapers-bespoke/gen-z-and-millennials-2020-report>.
- [13] D. Lemin, "3 statistics on how Millennials Are Choosing Restaurants," *Convince & Convert*, 31-May-2022. [Online]. Available: <https://www.convinceandconvert.com/digital-marketing/millennial-restaurant-statistics/>.
- [14] "5 consumer insights for restauranteurs in 2021," *OrderUp*, 05-Mar-2021. [Online]. Available: <https://orderup.ai/blog/5-consumer-stats-restaurant-owners-2021/>.
- [15] M. Brandau, "COVID-19 Report 43: Optimistically Cautious," *Datassential*, 08-Jan-2021. [Online]. Available: <https://info.datassential.com/resources/covid-19-report-43-optimistically-curious>.
- [16] Z. Ma and L. Wang, "Identifying the Impacts of Digital Technologies on Labor Market: A Case Study in the Food Service Industry," *2021 IEEE Integrated STEM Education Conference (ISEC)*, 2021, pp. 214-214, doi: 10.1109/ISEC52395.2021.9764118.
- [17] L. LaBerge, C. O'Toole, J. Schneider, and K. Smaje, "How COVID-19 has pushed companies over the technology tipping point--and transformed business forever," *McKinsey & Company*, 05-Oct-2020. [Online]. Available: <https://www.mckinsey.com/business-functions/strategy-and-corporate-finance/our-insights/how-covid-19-has-pushed-companies-over-the-technology-tipping-point-and-transformed-business-forever>.
- [18] N. Jonker, C. van der Crujisen, M. Bijlsma, and W. Bolt, "Pandemic payment patterns," *Journal of Banking & Finance*, vol. 143, Jun. 2022, doi: 10.1016/j.jbankfin.2022.106593.
- [19] W. Tsang and K. Priebe, "Mastercard study shows consumers globally make the move to contactless payments for everyday purchases, seeking touch-free payment experiences," *Mastercard Newsroom*, 29-Apr-2020. [Online]. Available: <https://www.mastercard.com/news/press/press-releases/2020/april/mastercard-study-shows-consumers-globally-make-the-move-to-contactless-payments-for-everyday-purchases-seeking-touch-free-payment-experiences/>.

[20] “Toast | restaurant point of sale & management system.” [Online]. Available: <https://pos.toasttab.com/>.

[21] R. Ruggless, “Pandemic intensifies restaurant wait-time concerns, Bluedot survey finds,” Nation's Restaurant News, 10-Feb-2021. [Online]. Available: <https://www.nrn.com/family-dining/pandemic-intensifies-restaurant-wait-time-concerns-bluedot-survey-finds>.

[22] A. Liszewski, “7-eleven stores in Japan are getting touch-free floating holographic self-checkouts,” *Gizmodo*, 31-Jan-2022. [Online]. Available: <https://gizmodo.com/7-eleven-shows-off-touch-free-holographic-self-checkout-1848452133>.

[23] “World's first * ! Non-contact/aerial display technology adopted for POS cash register Demonstration test of ‘Digi POS’ started at Seven-Eleven stores,” 東芝テック株式会社. [Online]. Available: https://www.toshibatec.co.jp/release/20220128_01.html.

[24] A. Belarmino, C. Raab, J. Tang, and W. Han, “Exploring the motivations to use online meal delivery platforms: Before and during quarantine,” *International Journal of Hospitality Management*, vol. 96, Jul. 2021.

[25] “Products & pricing,” Products and Partnership Plans | DoorDash for Merchants, 2022. [Online]. Available: <https://get.doordash.com/en-us/products>.

[26] A. M, A. Dinkar, S. C. Mouli, S. B and A. A. Deshpande, "Comparison of Containerization and Virtualization in Cloud Architectures," 2021 IEEE International Conference on Electronics, Computing and Communication Technologies (CONECCT), 2021, pp. 1-5, doi: 10.1109/CONECCT52877.2021.9622668.

[27] O. Al-Debagy and P. Martinek, "A Comparative Review of Microservices and Monolithic Architectures," 2018 IEEE 18th International Symposium on Computational Intelligence and Informatics (CINTI), 2018, pp. 000149-000154, doi: 10.1109/CINTI.2018.8928192.

[28] “Microservices,” *martinfowler.com*, 2014. [Online]. Available: <https://martinfowler.com/articles/microservices.html>.

[29] “Docker - Build, Ship, and Run Any App, Anywhere,” Docker, 2022. [Online]. Available: <https://www.docker.com/>.

[30] “Kubernetes Documentation,” Kubernetes, 2022. [Online]. Available: <https://kubernetes.io/docs/home/>.

- [31] L. Abdollahi Vayghan, M. A. Saied, M. Toeroe and F. Khendek, "Deploying Microservice Based Applications with Kubernetes: Experiments and Lessons Learned," 2018 IEEE 11th International Conference on Cloud Computing (CLOUD), 2018, pp. 970-973, doi: 10.1109/CLOUD.2018.00148.
- [32] P. Kasundra, "React Architecture Best Practices and Tips from Community Experts," Simform, <https://www.simform.com/blog/react-architecture-best-practices/>.
- [33] J.-M. Möckel, "React Best Practices – Tips for Writing Better React Code in 2022," freeCodeCamp.org, <https://www.freecodecamp.org/news/best-practices-for-react/>.
- [34] "Technical benefits of using Redux," Medium, <https://fintelics.medium.com/technical-benefits-of-using-redux-f7d345e7cc9c>.
- [35] "Uber Eats USA Restaurants and Menus", Kaggle, 2023 [Online]. Available: <https://www.kaggle.com/datasets/ahmedshahriarsakib/uber-eats-usa-restaurants-menus>
- [36] OpenAI API, <https://platform.openai.com/docs/introduction>.
- [37] Stripe API reference, <https://stripe.com/docs/api>.
- [38] "About PageSpeed Insights | Google Developers," Google, <https://developers.google.com/speed/docs/insights/v5/about>.
- [39] "Midjourney Documentation and User Guide," Midjourney, <https://docs.midjourney.com/>.
- [40] Zappa, "Zappa/Zappa: Serverless Python," GitHub, <https://github.com/zappa/Zappa>.
- [41] AWS Documentation, <https://docs.aws.amazon.com/>.