DWIT COLLEGE

DEERWALK INSTITUTE OF TECHNOLOGY

QR BASED ONLINE FOOD ORDERING AND ORDER MANAGEMENT SYSTEM USING REACT AND DJANGO

A MICRO PROJECT REPORT

Submitted to

Department of Computer Science

DWIT college

Submitted by

Sulav Baskota

13th January, 2022

DWIT College

DEERWALK INSTITUTE OF TECHNOLOGY

SUPERVISOR' RECOMMENDATION

I hereby recommend that this project prepared under my supervision by SULAV BASKOTA entitled "**QR BASED ONLINE FOOD ORDERING AND ORDER MANAGEMENT SYSTEM USING REACT AND DJANGO**" in partial fulfillment of the requirements for the degree of B.Sc. in Computer Science and Information Technology be processed for the evaluation.

……

Hitesh Karki

Campus Chief

Deerwalk Institute of Technology

DWIT College

DWIT College

DEERWALK INSTITUTE OF TECHNOLOGY

STUDENT'S DECLARATION

I hereby declare that I am the only author of this work and that no sources other than that listed here have been used in this work.

……

Sulav Baskota

13th January, 2022

DWIT College

DEERWALK INSTITUTE OF TECHNOLOGY

LETTER OF APPROVAL

This is to certify that this project prepared by SULAV BASKOTA entitled "**QR BASED ONLINE FOOD ORDERING AND ORDER MANAGEMENT SYSTEM USING REACT AND DJANGO**" in partial fulfillment of the requirements for the degree of B.Sc. in Computer Science and Information Technology has been well studied. In our opinion it is satisfactory in the scope and quality as a project for the required degree.

……	……
Hitesh Karki	Bijay Babu Regmi
Campus Chief	Year In-Charge
DWIT College	DWIT College

ACKNOWLEDGEMENT

I would like to express my sincere thanks to Mr. Hitesh Karki, for his valuable guidance and support in completing my project. His vision, sincerity and motivation have deeply inspired me. He has taught me the methodology to carry out the research and to present the research works as clearly as possible. It was a great privilege and honor to work and study under his guidance. I am extremely grateful for what he has offered me.

Besides my advisor, I would like to thank the rest of my project supervisor committee: Prof. Ritu Raj Lamsal, Prof. Bijay Babu Regmi, Prof. Sanjesh Rimal and Prof. Niresh Dhakal, for their encouragement, insightful comments, and hard questions. Without their support and suggestions, this project would not have been completed.

I would also like to express my gratitude towards our college, Deerwalk Institute of Technology, for giving me this great opportunity to do a project on "QR based online food ordering and order management system using React and Django".

Sulav Baskota

Roll No.: 845

13th January, 2022

ABSTRACT

Restaurant industry is one of the most heavily impacted industry by Covid-19. Many restaurants were unable to operate during the pandemic and struggled to stay in business. Even as things are going back to normal, the pandemic is still not over and new variations continue to threaten our lives.

Eating out at restaurant is still a huge risk factor for Covid-19 transmission. Menus at a restaurant are difficult to sanitize, and constant interaction with the restaurant staffs is another concern for contacting Covid-19 for both staffs and diners. In such a scenario, this project will provide a convenient system to restaurants to operate their business by minimizing some risk factors associated with eating out at restaurants.

This paper analyses some of the current existing problem with dining out at restaurant and attempts to solve them using a QR based online food ordering system. At the same time system proposed in this paper also attempts to help restaurants keep proper track of orders and transactions. Finally, this paper tries to showcase the potential of digitizing a restaurant's workflow and the potential of QR code integration in such systems.

Keywords: *Digital-menu; Django; Online-food-ordering; Online-order-management; QR-code; React; Redux.*

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVATIONS

API	Application Programming Interface
DB	Database
ER	Entity Relationship
NPM	Node Package Manager
QR	Quick Response
UI	User Interface
UML	Unified Modeling Language
URL	Uniform Resource Locator

CHAPTER 1: INTRODUCTION

1.1. OVERVIEW

QR Code, also known as Quick Response Code, are a type of bar code that consists of a printed square pattern of small black and white squares that encode data which can be scanned into a computer system. They were developed in 1994 by the Japanese corporation Denso Wave to track automobile parts during the assembly process [1]. QR Codes are used in a number of sectors for sharing information. One of the functionalities of QR codes that is gaining popularity in recent years is the use of QR to perform web redirections. A number of business owners already use QR codes to redirect customers to their website. An industry that can hugely benefit from the use of QR codes for web redirection is the restaurant industry.

1.2. BACKGROUND AND MOTIVATION

The idea for this project came about with the novel goal of reducing the risk of Covid-19 transmission in restaurants. Restaurants are one the industry that were most affected by the pandemic and many restaurants had to close down after being unable to run their business during the pandemic. In the US alone, the food service industry lost nearly 3.1 million jobs, more than 110,000 restaurants permanently closed, and Restaurant industry sales in 2020 were down $240 billion from expected levels due to the economic fallout caused by the pandemic [2]. Observing this situation, a solution that came to mind was to use QR codes and digital menus to order food in a restaurant.

Such an approach for collecting and tracking orders in a restaurant are not new and have been in use in many other countries. In addition, many restaurants outside Nepal started buying similar services during the pandemic. Thus, the idea for this project was born observing this situation to provide similar services to restaurants in Nepal.

1.3. PROBLEM STATEMENT

In a restaurant, menus are frequently reused and are difficult to sanitize after every use. According to a study from the Centers for Disease Control and Prevention (CDC) adults with Covid-19 were twice as likely to have dined out in a restaurant in the two weeks prior to their infection [3]. Physical printed menus are also expensive to update in the case where the menu is updated frequently. It is also difficult to reduce physical contact between the staff and diners to ensure their safety while maintaining satisfactory dining experience. In addition, during rush hours it can be tough in a restaurant to correctly track orders. This project aims to build a web-app that allows users to order food at restaurants through a digital menu.

1.4. OBJECTIVE

The main aspiration of this project is to decrease the amount of interaction between the customers and restaurant staffs with the purpose of providing a safer dining experience, while ensuring that the food is delivered to the correct table. Another aim of this project is to make sure that orders are prepared and delivered accurately.

1.5. SCOPE

This system is target to work for small restaurants and cafes. This project can be used in these restaurants to conveniently collect orders from customers. It will be of assistance to the restaurant staff to track and manage orders and can be used to keep record of a restaurant’s transactions.

1.6. LIMITATION

This project is built with the assumption that customers dining at the restaurant do not change tables to make it easier to track orders. The project also does not include any feature that allows a customer to make online payment for their order. In addition, this project is based around the idea that only customers physically present in the restaurant order food use the online food ordering system i.e. the online food ordering system will be hosted by a local server, and will only be accessible from those devices that are connected to the

restaurant's intranet. Finally, the food ordering system suggested in this paper, at present does not offer the feature for cancelling one's order it has been confirmed by a customer.

1.7. OUTLINE

This project analyses the problem with existing food ordering system in restaurants and develops a system to solve some issues with the existing system. This paper includes the following sections:

Preliminary Section: This section consists of the title page, abstract, table of contents, list of figures, and list of tables.

Introduction Section: In this section, the overview of the project, the background and motivation of the project, problem statement, its objectives and scope are discussed.

Requirement and Feasibility Analysis Section: This section talks about the research done about online food ordering system, and the functional and non-functional requirements of this project.

System Design Section: This section includes diagrams that help to elaborate on the overall design of the system proposed in this project.

Implementation and Evaluation Section: This section explains in detail, how the project was created as well as presents some evaluation results.

Conclusion: This section includes a conclusion to this paper.

CHAPTER 2: BACKGROUND AND RESEARCH

2.1. LITERATURE REVIEW

In Mayurkumar Patel's paper on Online Food Order System for Restaurants [4], the main features needed by online food ordering system is the ability to provide a digital menu to the user that they can interact with and place their orders. Similarly, the system also needs to show the users a summary of the orders placed by the them before they confirm their order. The order management system needs a feature in place for the restaurant staff to view incoming orders and the menu management system needs a feature to allow manipulation of the digital menu. The system proposed in this paper supports all the aforementioned features. In addition, it adds the feature of using a QR code to conveniently redirect users to the digital menu, and at the same time extract information embedded in the QR code passed on in the URL to track an order's origin.

2.2. CURRENT SYSTEM

Currently, most restaurants in Nepal use paper-based menu, and do not have a digitalized system in place for placing taking orders and tracking them.

2.3. THE PROBLEM WITH CURRENT SYSTEM

It is difficult to sanitize paper base menu in restaurant after every use. In addition, paper-based menu can be costly to update frequently. Restaurant that depend on paper-based menu need waiter to take orders, which increases interaction between diners and customers that can be risky for both parties. In restaurants without digital system for tracking orders, orders are lost and sometimes incorrect orders are delivered.

CHAPTER 3: SPECIFICATION AND DESIGN

3.1. REQUIREMENT ELICITATION AND ANALYSIS

The following features were added to the system after analyzing all the papers and documentations listed in the references section:

- Only Restaurant owner can manage menu
- Data from backend is stored in a redux store for better state management in react
- Customer are shown a summary of their orders before orders are placed

3.1.1. Functional Requirement

- The QR code shall redirect the diners to the menu page
- Customers shall be able to add items from the menu to their current order
- Customers shall be able to edit and cancel their current order before confirmation
- The restaurant owner shall be able to create and update the digital menu
- The kitchen staff shall be able to view pending orders
- The kitchen staff shall be able to mark finished orders as completed

3.1.2. Non-Functional Requirement

- Single invoice must be generated for orders from one table
- Order management system must have elegant user interface
- Code must be optimized to reduce application size
- New orders must be sent to order management system as soon as possible

3.2. SYSTEM DESIGN

The system architecture is elaborated using UML and other diagrams listed in the table below:

Table 1: List of UML Diagram Used

UML Diagram	Description
Use Case Diagram	It shows the interactions between a system and its environment
Sequence Diagram	It shows interactions between actors and the system and between system components

Table 2: List of Other Forms Used for Design

Method	Description
ER Diagram	An entity relationship diagram (ERD) shows the relationships of entity sets stored in a database

3.2.1. Use Case Diagram

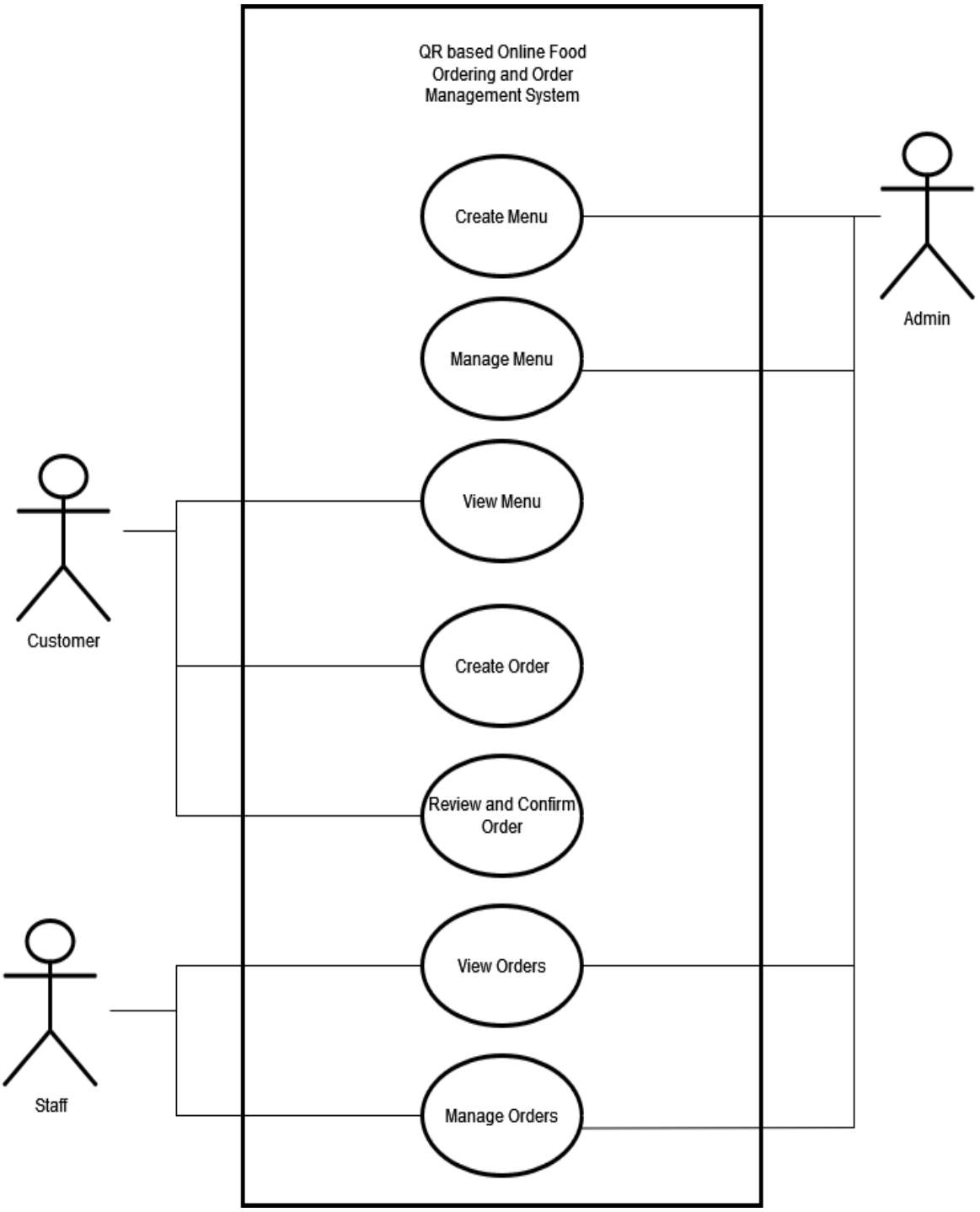

Figure 1: Use Case Diagram of the system

3.2.2. Sequence Diagram

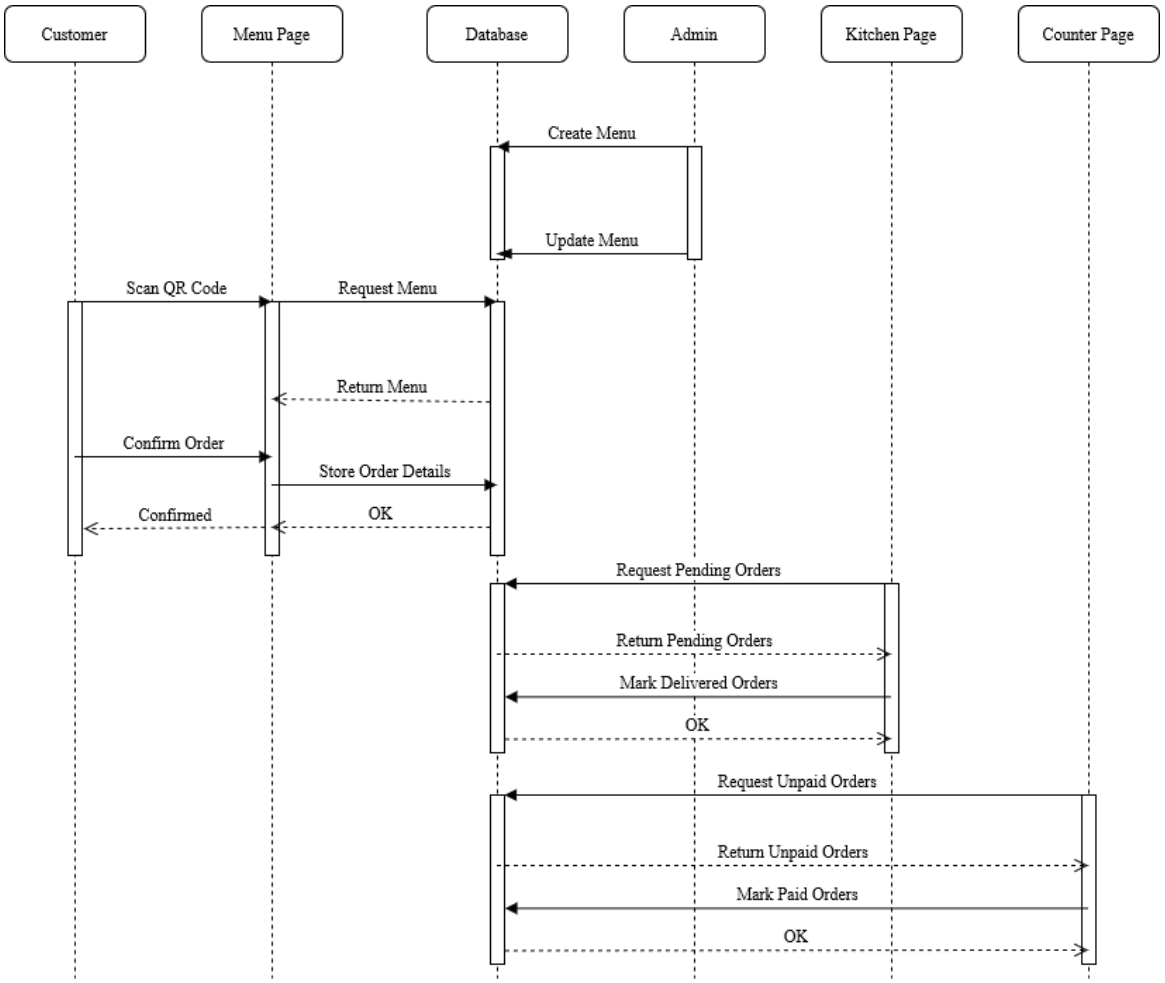

Figure 2: Sequence Diagram of the system

3.2.3. ER Diagram

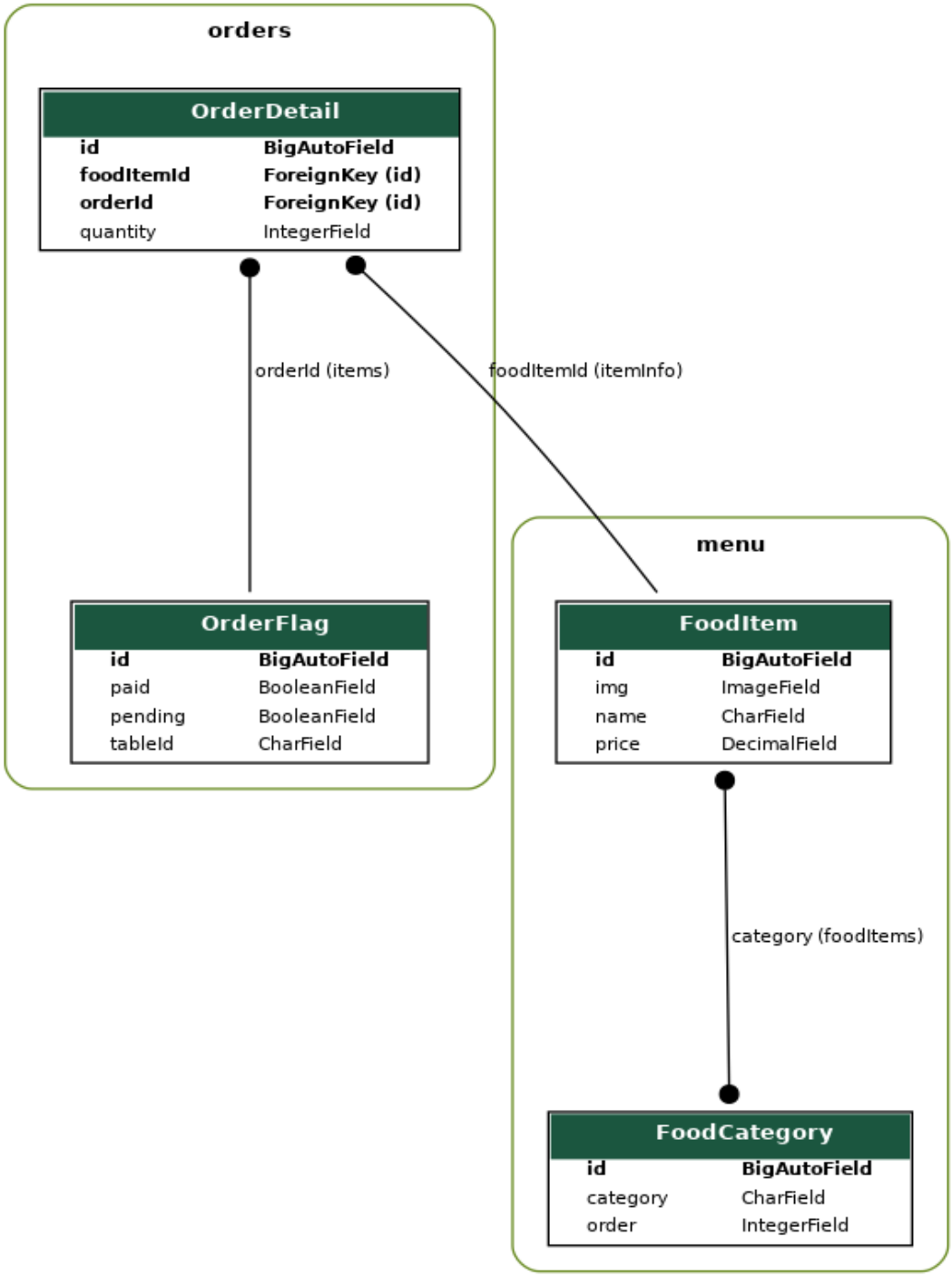

Figure 3: ER diagram of the system

CHAPTER 4: IMPLEMENTATION AND EVALUATION

4.1. TOOL AND TECHNOLOGY

The following tools and technologies were used to build the online food ordering and order management system:

- npm (v 6.14.15): A JavaScript package manager.
- Node.js (v 12.22.8): A JavaScript runtime environment.
- React (v 17.0.2): A JavaScript library for building user interfaces.
- qrcode.react (v 1.0.1): A React component to generate QR codes.
- @reduxjs/toolkit (v 1.6.2): It contains packages and functions that are essential for building a Redux app [5].
- react-router-dom (v 6.0.2): A package that contains bindings for using React Router in web applications.
- uuid (v 8.3.2): A package that is used to create RFC4122 UUIDs.
- @mui/material (v 5.2.2): A package to install Material UI, a popular React UI framework.
- Fetch API: A Web API that provides a JavaScript interface for accessing and manipulation parts of the HTTP pipeline, such as requests and responses. It also provides a global fetch() method that provides an easy, logical way to fetch resources asynchronously across the network [6].
- Python (v 3.9.5): An interpreted high-level programming language.
- Django (v 3.2.8): A python framework for building web applications.
- django-cleanup (v 5.2.0): A package to automatically delete old files for FileField and ImageField in a Django project.
- djangorestframework (v 3.12.4): A toolkit for building Web APIs in Django.

4.2. IMPLEMENTATION

The User Interface (frontend) is built using React, Redux and Material UI, whereas the data access layer (backend) is built using Django and Django REST framework.

The frontend is divided into five main components: Menu, Order, GenerateQR, Kitchen and Counter. The Menu and Order component are accessible to the customers. However, only restaurant staff have access to the other three components.

The “GenerateQR” component uses “qrcode.react” package and a table number value to generate a QR code which when scanned by a customer redirects them to the Menu component and passes the table number in the URL.

Menu component is built for fetching menu from the backend server and displays it to the customer. The Order component uses Redux to store order information given by the user. It also displays an order summary to the user once they are ready to confirm their order. And once an order is confirmed the Order component sends the order information along with the Table Id extracted from the URL to the backend to process this data.

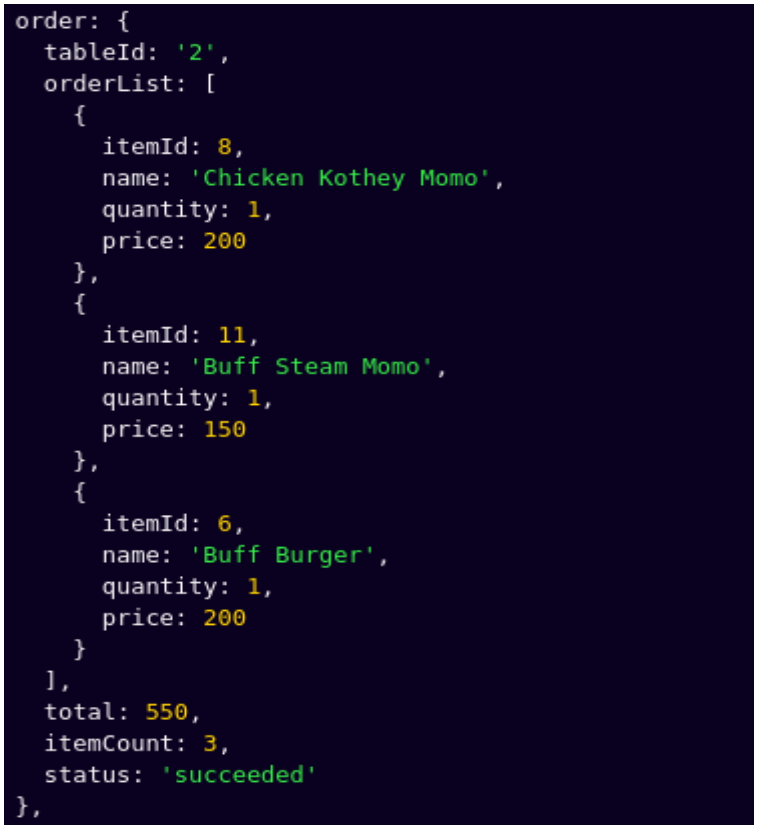

Figure 4: Example of an order stored in a state using Redux

The Kitchen component displays the list of pending orders to the restaurant staff. It uses FetchAPI to fetch the list of pending orders from the backend at a specified time interval, and then uses Redux to store this data into redux store. A feature to mark pending orders as “Delivered” is added to track orders that have been delivered to the customer. After an order has been marked as delivered, the corresponding Order Id is sent to the backend using FetchAPI to update its value. The Counter component operates in a similar fashion to the Kitchen component, however instead of displaying a list of pending orders, this component displays a list of unpaid orders. It allows the user to mark an unpaid order as “Paid”.

To update order information stored in the redux store whenever customer updates their order, five different reducer functions were defined: setTableId, increment, decrement, addOrder, and removeOrder. Reducers are functions that calculate a new state value based on previous state and an action [7].

```
setTableId: (state, action) => {
  state.tableId = action.payload;
},
addOrder: (state, action) => {
  return {
    ...state,
    orderList: [
      ...state.orderList,
      {
        itemId: action.payload.itemId,
        name: action.payload.itemName,
        quantity: 1,
        price: Number(action.payload.itemPrice),
      },
    ],
    itemCount: state.itemCount + 1,
    total: state.total + Number(action.payload.itemPrice),
  };
},
removeOrder: (state, action) => {
  return {
    ...state,
    orderList: state.orderList.filter(
      (order) => order.itemId !== action.payload.itemId
    ),
    itemCount: state.itemCount - 1,
    total: state.total - Number(action.payload.itemPrice),
  };
},
```

Figure 5: Code for setTableId, addOrder, and removerOrder reducer functions

```
increment: (state, action) => {
  return {
    ...state,
    orderList: state.orderList.map((order) => {
      if (order.itemId !== action.payload.itemId) {
        return order;
      }
      return {
        ...order,
        quantity: order.quantity + 1,
      };
    }),
    itemCount: state.itemCount + 1,
    total: state.total + Number(action.payload.itemPrice),
  };
},
decrement: (state, action) => {
  return {
    ...state,
    orderList: state.orderList.map((order) => {
      if (order.itemId !== action.payload.itemId) {
        return order;
      }
      return {
        ...order,
        quantity: order.quantity - 1,
      };
    }),
    itemCount: state.itemCount - 1,
    total: state.total - Number(action.payload.itemPrice),
  };
},
```

Figure 6: Code for increment and decrement reducer functions

Django is used to create two Django apps, menu and orders. The menu app defines two data models, FoodCategory and FoodItem. The FoodCategory model is used to create a new category of food items, and the FoodItem model is used to create a new food item.

The orders app also defines two data models, OrderDetail and OrderFlag. The OrderFlag model is used to store data about whether an Order is pending or not, and whether it is paid or not. The OrderDetail model stores information about different food items in an order along with their corresponding quantity.

```python
class FoodCategory(models.Model):
    category = models.CharField(max_length=30, unique=True)
    order = models.IntegerField(default=0, validators=[MinValueValidator(0)])

    class Meta:
        verbose_name = 'Food Category'
        verbose_name_plural = 'Food Categories'

    def __str__(self):
        return self.category


class FoodItem(models.Model):
    name = models.CharField(max_length=50, unique=True)
    category = models.ManyToManyField(FoodCategory, related_name='foodItems')
    price = models.DecimalField(max_digits=10, decimal_places=2,
                                verbose_name='Price in NPR', validators=[MinValueValidator(0.01)])
    img = models.ImageField(upload_to='images', verbose_name='Image')

    class Meta:
        verbose_name = 'Food Item'
        verbose_name_plural = 'Food Items'

    def __str__(self):
        return self.name
```

Figure 7: Code for defining FoodCategory and FoodItem models

```python
class OrderDetail(models.Model):
    orderId = models.ForeignKey('OrderFlag', on_delete=models.CASCADE, related_name='items')
    foodItemId = models.ForeignKey(
        FoodItem, on_delete=models.CASCADE, related_name='itemInfo')
    quantity = models.IntegerField()

    class Meta:
        verbose_name: 'Order Detail'
        verbose_name_plural: 'Order Details'


    def __str__(self):
        return str(self.orderId) + ', ' + str(self.foodItemId)


class OrderFlag(models.Model):
    tableId = models.CharField(max_length=2)
    pending = models.BooleanField(default=True)
    paid = models.BooleanField(default=False)

    class Meta:
        verbose_name: 'Order Flag'
        verbose_name_plural: 'Order Flags'


    def __str__(self):
        return "Order Number: " + str(self.id)
```

Figure 8: Code for defining OrderDetail and OrderFlag models

Django REST framework is used to create Web APIs that the frontend can interact with to access information about the food items and order details. Serializers are used to preprocess and format data that is to be send to the frontend.

```python
class ShowMenu(APIView):
    serializer_class = FoodCategorySerializer

    def get(self, request):
        queryset = FoodCategory.objects.all()
        response = {
            "menuList": [

            ]
        }
        if queryset.exists():
            for query_item in queryset:
                response['menuList'].append(self.serializer_class(query_item).data)
        return Response(response, status=status.HTTP_200_OK)
```

Figure 9: Code for defining API for fetching menu

```python
class FoodItemSerializer(serializers.ModelSerializer):
    class Meta:
        model = FoodItem
        fields = ('id', 'name', 'price', 'img')


class FoodCategorySerializer(serializers.ModelSerializer):
    foodItems = FoodItemSerializer(many=True)

    class Meta:
        model = FoodCategory
        fields = ('category', 'foodItems')
```

Figure 10: Code for defining Serializer class in menu app

4.3. EVALUATION AND RESULT

To test the system 5 different QR codes were generated. 8 users were asked to each select and scan a QR code and order food items. When the users were connected to the local hotspot where the local server was running, they were successfully able to access the digital menu and place orders. The orders placed by the users in this scenario were successfully displayed in the Kitchen and Counter component. Orders listed in Kitchen and Counter components were successfully marked as “Delivered” and “Paid” respectively.

In the scenario where, the users were not connected to the local hotspot, they were unable to access the menu after scanning a QR code.

CHAPTER 5: CONCLUSION

The pandemic that started at the end of 2019 forced many restaurants to close their business and eating out at a restaurant became a major health concern. Restaurants needed an effective solution that can ensure the safety of their customer while at the same time allowed their business to operate as normal. Thus, the project proposed in this paper is developed with the aim of replacing physical menus and reducing interactions in a restaurant to reduce risk of Covid-19 transmission. QR based online food ordering and order management system not only allows customers to safely and conveniently order food using a digital menu, but at the same time helps restaurants digitize part of their workflow.

This paper beseeches the reader to further explore other industries that can benefit from a system similar to the one proposed in this paper.

REFERENCES

[1] E. Gregersen, "Encyclopedia Britannica," [Online]. Available: https://www.britannica.com/technology/QR-Code. [Accessed 9 1 2022].

[2] "National Statistics," National Restaurant Association, [Online]. Available: https://restaurant.org/research-and-media/research/industry-statistics/national-statistics/. [Accessed 9 1 2022].

[3] Fisher KA, Tenforde MW, Feldstein LR, et al., "Community and Close Contact Exposures Associated with COVID-19 Among Symptomatic Adults ≥18 Years in 11 Outpatient Health Care Facilities," MMWR Morb Mortal Wkly Rep, 2020.

[4] M. Patel, "Online Food Order System for Restaurants," 2015.

[5] "Getting Started with Redux," [Online]. Available: https://redux.js.org/introduction/getting-started. [Accessed 10 1 2022].

[6] "Using Fetch," Mozilla, [Online]. Available: https://developer.mozilla.org/en-US/docs/Web/API/Fetch_API/Using_Fetch. [Accessed 10 1 2022].

[7] "Redux Fundamentals, Part 1: Redux Overview," [Online]. Available: https://redux.js.org/tutorials/fundamentals/part-1-overview. [Accessed 10 1 2022].