

A
Project Review (MCA2099) Report
On
Online Food Ordering System



VIT[®]
BHOPAL
www.vitbhopal.ac.in

Submitted by

Shital Gupta 22MCA10022

Aastha Kumari 22MCA10098

Shrey Srivastava 22MCA10154

Priyanshu Saraswat 22MCA10155

Farheen Kouser 22MCA10198

Guided by
Dr Prabu Kanna



Submitted to
School of Computing Science and Engineering
VIT Bhopal University
Bhopal (MP) – 466 114

April 2023

ABSTRACT

[About the Project Online Food Ordering System]

The purpose of Online Food Ordering System is to automate the existing manual system by the help of computerized equipment's and full-fledged computer software, fulfilling their requirements, so that their valuable data/information can be stored for a longer period with easy accessing and manipulation of the same. The required software and hardware are easily available and easy to work with.

Online Food Ordering System, as described above, can lead to error free, secure, reliable and fast management system. It can assist the user to concentrate on their other activities rather to concentrate on the record keeping. Thus it will help organization in better utilization of resources. The organization can maintain computerized records without redundant entries. That means that one need not be distracted by information that is not relevant, while being able to reach the information.

The aim is to automate its existing manual system by the help of computerized equipment's and full-fledged computer software, fulfilling their requirements, so that their valuable data/information can be stored for a longer period with easy accessing and manipulation of the same. Basically the project describes how to manage for good performance and better services for the clients.

ACKNOWLEDGEMENT

I sincerely wish to thank our Program chair **Dr Kanchan Lata Kashyap** and respected guide **Dr Prabu Kanna** for their valuable guidance and supervision throughout the development of the project

We are obliged to the advice, encouragement tips and timely suggestion provided by Dr Prabu Kanna. His guidance and impassionate support all around is really thankful. The time spent with him for the development of the project was full of potential as well as knowledge. His prospects of motivation, discipline, timely discussions and tips along with the moral support as well as confidence extended us certainly to make a great deal of learning.

We want to express our special thanks to our entire batch mates for fruitful discussions. We are also grateful to all others who helped us directly or indirectly in our work.

We dedicate our project work to our parents who have influenced us in more ways than they know. Finally we bow down before the Divine Majesty who made everything possible.



VIT[®]
BHOPAL
www.vitbhopal.ac.in

VIT BHOPAL UNIVERSITY, M P - 466114

SCHOOL OF COMPUTING SCIENCE AND ENGINEERING

CANDIDATE'S DECLARATION

I hereby declare that the Dissertation entitled "Online Food Ordering System" is my own work conducted under the supervision of Dr Prabu Kanna, Asst. Professor, Master Of Computer Application, VIT University, Bhopal.

I further declare that to the best of my knowledge this report does not contain any part of work that has been submitted for the award of any degree either in this university or in other university / Deemed University without proper citation.

Shital Gupta 22MCA10022

Aastha Kumari 22MCA10098

Shrey Srivastava 22MCA10154

Priyanshu Saraswat 22MCA10155

Farheen Kouser 22MCA10198

This is to certify that the above statement made by the candidate is correct to the best of my knowledge.

Date: April 2023

Guide Name: Dr Prabu Kanna

Designation : Asst. Professor

TABLE OF CONTENTS

1. Introduction

- 1.1 About the Project
- 1.2 Features of Project
- 1.3 Scope
- 1.4 Limitation

2. Description of Project

- 2.1 Introduction
- 2.2 Overall Description

3. Project Requirements

- 3.1 Software Requirement
- 3.2 Hardware Requirement

4. System Design

- 4.1 ER Diagram
- 4.2 Flow Chart

5. Code and User Interface

6. Conclusion

7. Bibliography

Introduction

1.1 About the Project

The "Online Food Ordering System" has been developed to override the problems prevailing in the practicing manual system. This software is supported to eliminate and, in some cases, reduce the hardships faced by this existing system. Moreover, this system is designed for the particular need of the company to carry out operations in a smooth and effective manner.

The main objective of the Project on Online Food Ordering System is to manage the details of Food Item, Order, Confirm Order. It manages all the information about Food Item, Payment, Confirm Order. The purpose of the project is to build an application program to reduce the manual work for taking orders of the Food Item and Payment.

The application is reduced as much as possible to avoid errors while entering the data. It also provides error message while entering invalid data. No formal knowledge is needed for the user to use this system. Thus, by this all it proves it is user-friendly. Online Food Ordering System, as described above, can lead to error free, secure, reliable and fast management system. It can assist the user to concentrate on their other activities rather to concentrate on the record keeping. Thus, it will help organization in better utilization of resources.

Every organization, whether big or small, has challenges to overcome and managing the information of Category, Food Item, Order, Payment, and Confirm Order. Every Online Food Ordering System has different Food Item needs; therefore, we design exclusive employee management team that are adapted to your managerial requirements. This is designed to assist in strategic planning and will help you ensure that your organization is equipped with the right level of information and details for your future goals. Also, for those busy executives who are always on the go, our systems come with remote access features, which will allow you to manage your workforce anytime, at all times. These systems will ultimately allow you to better manage resources.

1.2 Features of Project

- Product and Component based
- Creating & Changing Issues at ease
- Query Issue List to any depth
- Reporting & Charting in more comprehensive way
- User Accounts to control the access and maintain security
- Simple Status & Resolutions
- Multi-level Priorities & Severities.
- Targets & Milestones for guiding the programmers
- Attachments & Additional Comments for more information
- Robust database back-end
- Various level of reports available with a lot of filter criteria's □ It contains better storage capacity.
- Accuracy in work.
- Easy & fast retrieval of information.
- Well-designed reports.
- Decrease the load of the person involve in existing manual system.
- Access of any information individually.
- Work becomes very speedy.
- Easy to update information

1.3 Scope

It may help providing perfect interface to users. In a very short time, the order will be made simple and sensible. It will help a person to easily order the food in his area according to liking. It also helps in current process of Online Food Ordering System. It will be also reduced the cost of collecting the management & collection procedure will go on smoothly.

- In computer system the person has to fill the registration details & login using credentials.
- In computer system, it is not necessary to login for just viewing the products
- To assist the staff in capturing the effort spent on their respective working areas.

- To utilize resources in an efficient manner by increasing their productivity through automation.
- It satisfy the user requirement
- Be easy to understand by the user
- Easy to operate
- Have a good user interface
- Delivered on schedule within the budget.

1.4 Limitations:

- High Competition
- Delivery Charges may vary
- Limited & Irregular Menu
- Food Can Get Cold
- Late & Incorrect Orders
- Effects on Health
- Excessive Production & Waste

DESCRIPTION OF THE PROJECT

2.1 INTRODUCTION

Order Food online system is a process in which one can order various foods and beverages from some local restaurant and hotels through the use of the internet, just by sitting at home or any place. And the order is delivered to the old location.

The user can place orders for the food items of their like from the list. The payment can be made online or pay-on-delivery system. The user's details are maintained confidentially because it maintains a separate account for each user. An id and password are provided for each user.

The traditional food ordering procedure is not efficient enough for hotels and restaurants, as they have to deal with the crowd, in their restaurant. The old methods can be classified into categories which are paper grounded and verbal grounded. For paper-based work, the waiter comes and pens down foods that customers order and pass the food list containing paper to the chefs or cooks in the kitchen for further process.

So, to cope up with these we have "Online Food Ordering System".

2.2 DETAILED DESCRIPTION

Nowadays everyone is having a busy schedule whether it is urban areas or rural. But talking specifically about the urban areas and deeply about the big cities, people out there are so busy in their life that they don't get enough time to have their meals properly. These days women are no less than men, in any field.

So, in big cities even wives are working women, therefore mostly the small families manage to have their food ordered from somewhere, as they lack time. Not only is this the case, if we talk about the children in the modern era, they like only fast food or something from the outside. But they ignore eating homemade meals.

So, the food ordering system these days has one of the fastest-growing markets, though being a new idea. In this project, we have developed something like the same to learn from and serve the nation in a much better way possible. Nowadays, people are more regular to dine-in at the restaurant for their meals.

The online food ordering system provides convenience for the customers that are nothing special but the general busy people of the society. It overcomes the demerits of the manual hotel or mess system and the old-fashioned queuing system. This system enhances the ready-made foods that people.

Therefore, this system enhances the speed of getting food on a person's plate and the quality and manner of taking the order from the customer. It provides a better communication platform. The user's details are stored using electronic media. The online food ordering system provides the

menu online and the customers can easily place the order by just clicking the mouse or by touching a button on their smartphones.

Also, with the food ordering system online, people can easily track their orders, and the admin can maintain the customer's database and advance

the food delivery system. This food ordering system allows the user to select the desired food items from a list of available menu items provided by the local hotel or restaurant.

The user can place orders for the food items of their like from the list. The payment can be made online or pay-on-delivery system. The user's details are maintained confidentially because it maintains a separate account for each user. An id and password are provided for each user.

Existing System of Online Food Ordering System

In the present scenario, people have to physically visit the hotels or restaurants for eating food and have to make payments through cash mode most of the time due to unawareness of advanced technologies at certain places. In this method time as well as physical work is required, among which time is something that no one has in ample amount.

The traditional food ordering procedure is not efficient enough for hotels and restaurants, as they have to deal with the crowd, in their restaurant. The old methods can be classified into categories which are paper grounded and verbal grounded. For paper-based work, the waiter comes and pens down foods that customers order and pass the food list containing paper to the chefs or cooks in the kitchen for further process.

Proposed System of Online Food Ordering System:

This system is a bunch of benefits from various points of view. This online application enables the end-users to register to the system online, select the food items of their choice from the menu list, and order food online. Also, the payment can be made through online mode or at the time of home delivery depending upon the customer's choice and convenience.

The aim of proposed system is to develop a system of improved facilities. The proposed system can overcome all the limitations of the existing system. The system provides proper security and reduces the manual work

- Security of data.
- Ensure data accuracy's.
- Proper control of the higher officials.
- Minimize manual data entry.
- Minimum time needed for the various processing.
- Greater efficiency.

PROJECT REQUIREMENTS-

3.1 SOFTWARE REQUIREMENTS

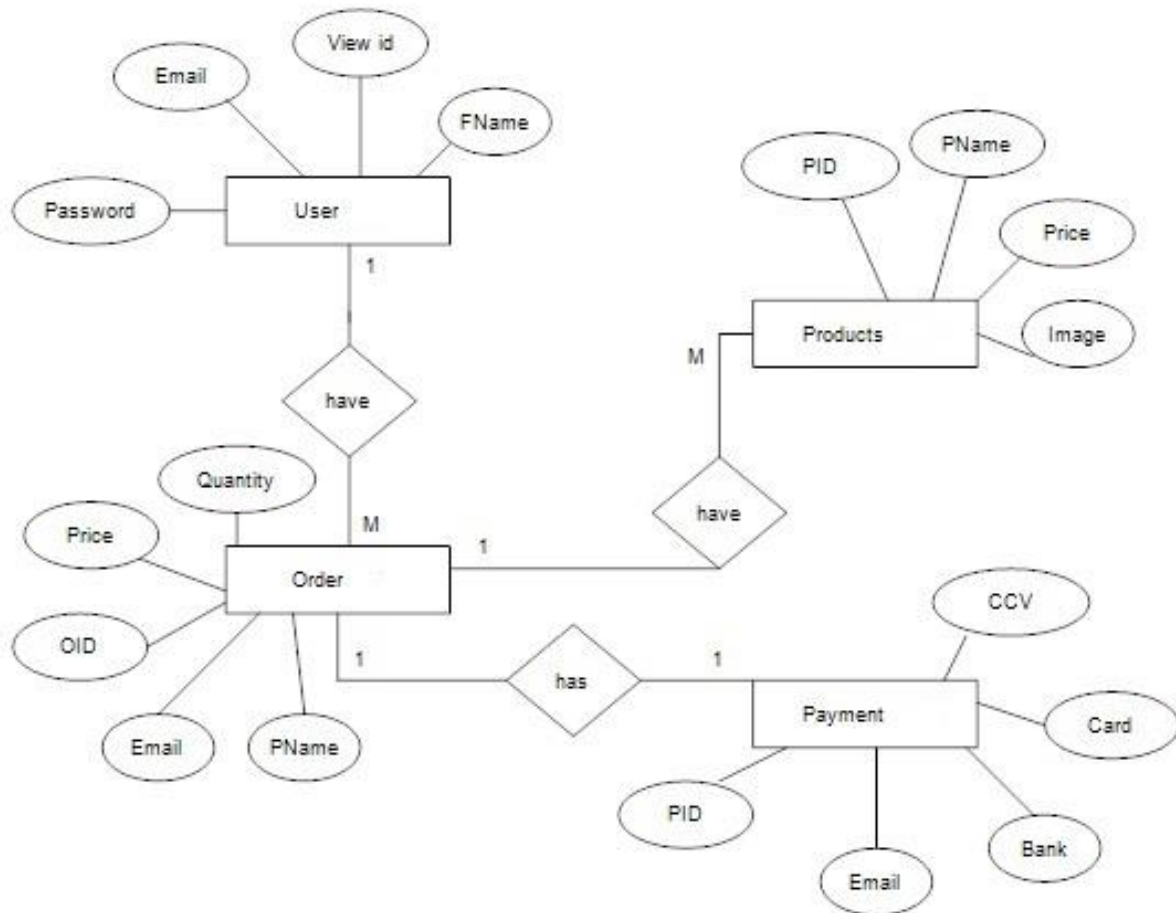
Name of Component	Specification
Operating System	Windows 8.1/10 , MAC
Language	React
Database	MongoDb
Browser	Any of Chrome, Mozilla, Opera etc.
Framework	MERN Stack
Web Technologies	ReactJs, NodeJs, ExpressJs

3.2 HARDWARE REQUIREMENTS

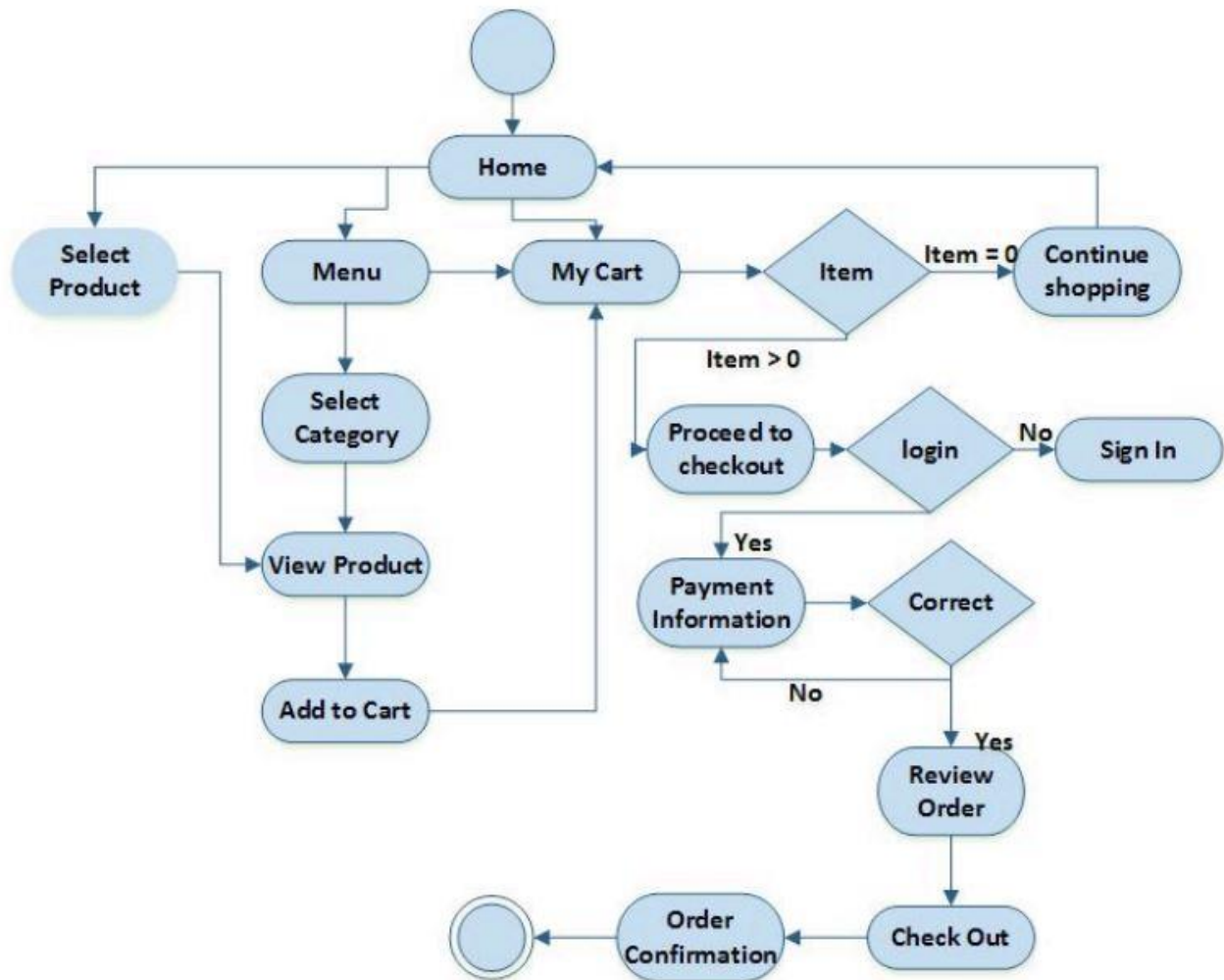
<u>Minimum RAM</u>	<u>4GB</u>
<u>Processor</u>	<u>Core i3</u>
<u>Free Space</u>	<u>1 GB</u>

DESIGN

4.1 ER DIAGRAM



4.2 FLOWCHART

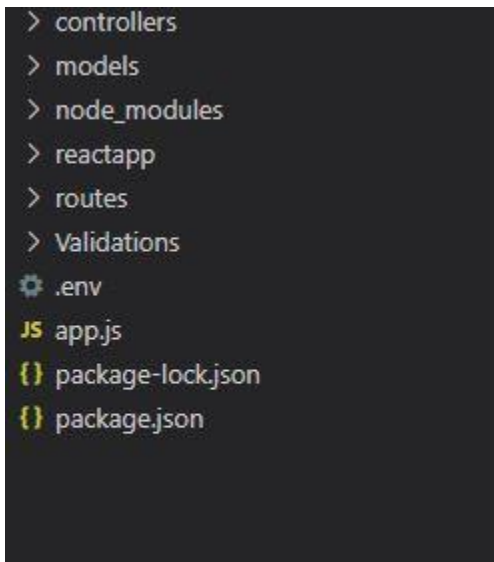


CODE AND USER INTERFACE

PROJECT MENUS

We have three parts of food ordering system:-

- Frontend
- Backend
- Database



```
> controllers
> models
> node_modules
> reactapp
> routes
> Validations
⚙ .env
📄 app.js
📄 package-lock.json
📄 package.json
```

Project Menus a

- ▼ controllers
 - JS city.js
 - JS FControllers.js
 - JS mealtype.js
 - JS Menu.js
 - JS Orders.js
 - JS paymentGatewayController.js
 - JS PaytmChecksum.js
 - JS Restaurant.js
 - JS User.js
- ▼ models
 - JS city.js
 - JS mealtype.js
 - JS Menu.js
 - JS Orders.js
 - JS Restaurant.js
 - JS User.js
- > node_modules
- > reactapp
- ▼ routes
 - JS router.js
- ▼ Validations
 - JS userSignup.js
- ⚙ .env
- JS app.js
- { } package-lock.json
- { } package.json

Backend Menu structure 1

- > controllers
- > models
- > node_modules
- ▼ reactapp
 - > node_modules
 - > public
 - ▼ src
 - > assets
 - ▼ components
 - JS Details.js
 - JS Filter.js
 - JS Header.js
 - JS Header1.js
 - JS Home.js
 - JS navigation_name.js
 - JS QuickSearches.js
 - JS QuickSearchItem.js
 - JS Restaurant.js
 - JS Transaction.js
 - JS Wallpaper.js
 - ▼ styles
 - # details.css
 - # Filter.css
 - # header.css
 - # Home.css
 - # Restaurant.css
 - # App.css
 - JS App.js
 - JS App.test.js
 - # index.css
 - JS index.js
 - 🖼 logo.svg
 - JS reportWebVitals.js
 - JS Router.js
 - JS setupTests.js
 - ≡ .eslintcache
 - 💎 .gitignore
 - ≡ backend
 - ≡ debug.log

Frontend menu Structure 1

SCREENSHOTS



Quick Searches

Discover restaurants by type of meal



Drinks

Have Fun With Drinks



Snacks

Party with numerous snacks



Dinner

End your day with exclusive dinner options



Lunch

Relax in your work time with exclusive lunch options



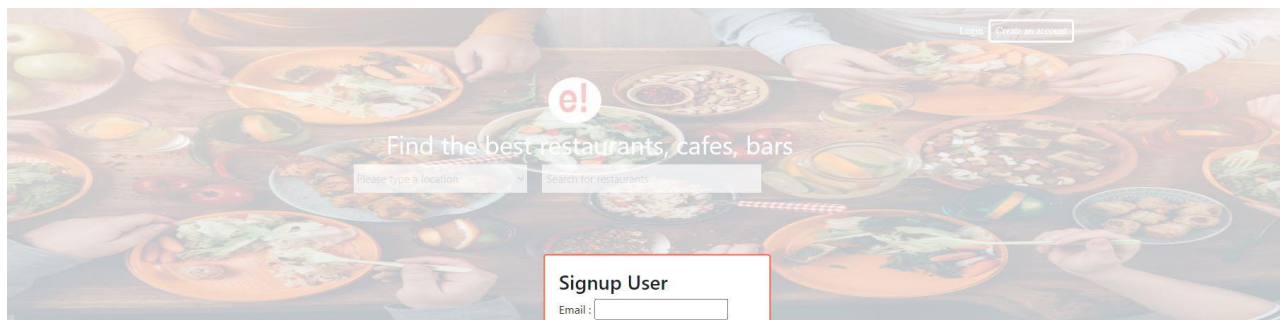
NightLife

Enjoy The Nightlife



Breakfast

Start your day with exclusive breakfast options



Quick Searches

Discover restaurants by type of meal



Drinks

Have Fun With Drinks



Snacks

Party with numerous snacks



Dinner

End your day with exclusive dinner options



Lunch

Relax in your work time with exclusive lunch options



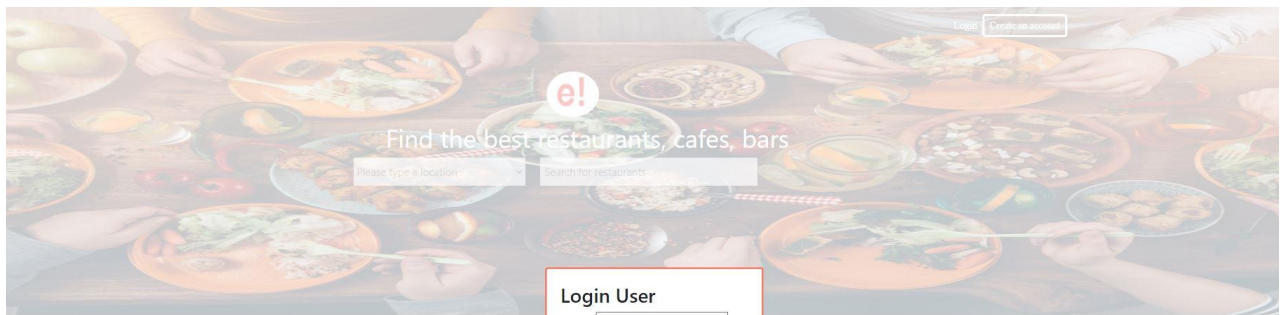
NightLife

Enjoy The Nightlife



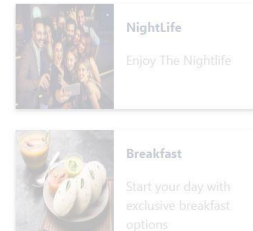
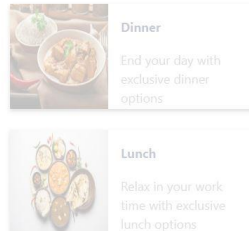
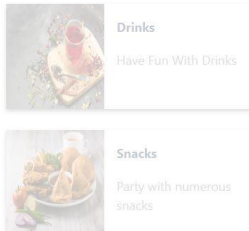
Breakfast

Start your day with exclusive breakfast options



Quick Searches

Discover restaurants by type of meal



Places in Delhi

Filters
Select Location

Delhi

Cuisine

☐ North Indian

☒ South Indian

☒ Chinese

☐ Fast Food

☐ Street Food

Cost For Two

☐ Less than ₹ 500

☐ ₹ 500 to ₹ 1000

☐ ₹ 1000 to ₹ 1500

☐ ₹ 1500 to ₹ 2000

☐ ₹ 2000 +

☐ All

Sort

☒ Price high to low

☐ Price low to high



AMA Cafe
Majnu ka Tila, New Delhi, Delhi
House 6, New Colony, Majnu ka Tilla, New Delhi

CUISINES : South Indian, Chinese,
COST FOR TWO : 450



Punjabi Angithi
Paschim Vihar, New Delhi, Delhi
32-22, A 4, DDA Market, Paschim Vihar, New Delhi

CUISINES : South Indian, Chinese,
COST FOR TWO : 350



Quick Searches

Discover restaurants by type of meal



Drinks

Have Fun With Drinks



Snacks

Party with numerous snacks



Dinner

End your day with exclusive dinner options



Lunch

Relax in your work time with exclusive lunch options



NightLife

Enjoy The Nightlife



Breakfast

Start your day with exclusive breakfast options



Punjabi Angithi

[Overview](#) [Contact](#)

About this place

Cuisine

South Indian, Chinese,

Average Cost

350

[Place online order](#)



Shrey

Logout



Punjabi Angithi

Overview Contact

Place online order

Phone Number

(+91) 14004566

Address

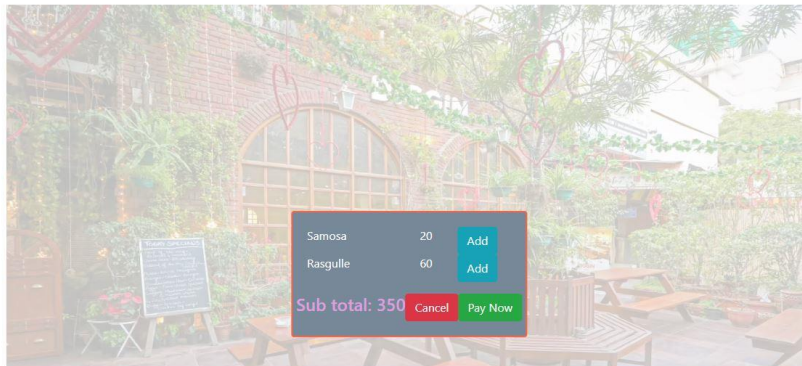
Punjabi Angithi

32-22, A 4, DDA Market, Paschim Vihar, New Delhi



Shrey

Logout



Samosa	20	Add
Rasgulle	60	Add

Sub total: 350 Cancel Pay Now

Punjabi Angithi

Overview Contact

Place online order

About this place

Cuisine

South Indian, Chinese,

Average Cost

CODE

I. BACKEND

A. CONTROLLERS

1. City.js

```
const city = require('../models/city');

const restaurantList = require('../models/Restaurant');

exports.getCityList = (req , res) => {
  city.find().then(result => {

    res.status(200).json({
      message: "City List Fetched",
      cities: result
    });
  }).catch(error => {
    res.status(500).json({
      message: error
    });
  });
}

exports.getRestaurantsByCity = (req, res) => {
  console.log(req.params.city.toString())
  restaurantList.find({
    city: req.params.city.toString()
  }).then(result => {
    res.status(200).json({
      restaurants: result,
      message: "Success"
    });
  }).catch(error => {
    res.status(500).json({
      message: error
    });
  });
}
```

2. Fcontroller.js

```
const restaurantDataModel = require('../models/Restaurant');

exports.filterRestaurants = async(req, res) => {
  let { city, cuisine, cost, sort, page } = req.body;
  const condition = {};
  if (city !== undefined) {
    condition["Cuisine:name"] = Cuisine;
  }
  switch (cost) {
    case '0-500':
      condition["cost"] = { $lt: 500, $gte: 0 };
      break;
    case '500-1000':
      condition["cost"] = { $lt: 1000, $gte: 500 };
      break;
    case '1000-1500':
      condition["cost"] = { $lt: 1500, $gte: 1000 };
      break;
    case '1500-2000':
      condition["cost"] = { $lt: 2000, $gte: 1500 };
      break;
    case '2000+':
      condition["cost"] = { $gte: 2000 };
      break;
    default:
      condition["cost"] = { $gte: 0 };
  }
  if (page == undefined) {
    page = 1;
  }
  const limit = 2;
  const skip = (page - 1) * limit || 0;
  const order = sort === 'dsc' ? -1 : 1;
  let restaurants;
  if (sort !== undefined) {
    restaurants = await restaurantDataModel.find(condition).sort({ cost: order
  }).limit(limit).skip(skip);
  } else {
    restaurants = await
restaurantDataModel.find(condition).limit(limit).skip(skip);
  }
  console.log(condition);
  res.json({
    success: true,
    docsInPresentPage: restaurants.length,
    result: restaurants
  });
}
```

```
});  
}
```

3. Mealtype.js

```
const meal = require('../models/mealtype');  
  
exports.getMeals = (req , res) => {  
  meal.find().then(result => {  
  
    res.status(200).json({  
      message: "Meal List Fetched",  
      meals: result  
    });  
  }).catch(error => {  
    res.status(500).json({  
      message: error  
    });  
  });  
}
```

4. Menu.js

```
• const Menu = require('../models/Menu');  
•  
• exports.getMenuForRestaurant = (req, res) => {  
•   Menu.find({  
•     restaurantId: req.params.id  
•   }).then(result => {  
•     res.status(200).json({  
•       menu: result,  
•       message: "Success"  
•     });  
•   }).catch(error => {  
•     res.status(500).json({  
•       message: error  
•     });  
•   });  
• }  
• }
```


5.Orders.js

```
const Orders = require('../models/Orders');

exports.getOrdersForUser = (req, res) => {
  Orders.find({
    username: req.params.username
  }).then(result => {
    res.status(200).json({
      orders: result,
      message: "Success"
    });
  }).catch(error => {
    res.status(500).json({
      message: error
    });
  });
}

exports.saveOrderForUser = (req, res) => {

  const username = req.body.username;
  const mobileNumber = req.body.mobileNumber;
  const address = req.body.address;
  const items = req.body.items;
  const total = req.body.total;

  const saveOrderForUsers = new User({
    username: username,
    mobileNumber: mobileNumber,
    address: address,
    items: items,
    total: total
  });

  saveOrderForUsers.save().then(
    result => {
      res.status(200).json({
        message: "User's order saved successfully !",
        user: result
      });
    }
  ).catch(
    error => {
      res.status(500).json({
        message: error
      });
    }
  );
}
```

```
    }  
  );  
}
```

6.PaymentGatewayController.js

```
require('dotenv').config();  
const formidable = require('formidable');  
const { v4: uuidv4 } = require('uuid');  
const https = require('https');  
const PaytmChecksum = require('./PaytmChecksum');  
  
exports.payment = (req, res) => {  
  const { amount, email, mobileNo } = req.body;  
  
  const totalAmount = JSON.stringify(amount);  
  var params = {};  
  params['MID'] = process.env.PAYTM_MID;  
  params['WEBSITE'] = process.env.PAYTM_WEBSITE;  
  params['CHANNEL_ID'] = process.env.PAYTM_CHANNEL_ID;  
  params['INDUSTRY_TYPE_ID'] = process.env.PAYTM_INDUSTRY_TYPE_ID;  
  params['ORDER_ID'] = uuidv4();  
  params['CUST_ID'] = process.env.PAYTM_CUST_ID;  
  params['TXN_AMOUNT'] = totalAmount;  
  params['CALLBACK_URL'] = 'http://localhost:3000/transaction';  
  params['EMAIL'] = email;  
  params['MOBILE_NO'] = mobileNo;  
  
  let paytmChecksum = PaytmChecksum.generateSignature(  
    params,  
    process.env.PAYTM_MERCHANT_KEY  
  );  
  
  paytmChecksum.then(response => {  
    let paytmChecksumResponse = {  
      ...params,  
      "CHECKSUMHASH": response  
    };  
    res.json({ checkSumResponse: paytmChecksumResponse });  
  }).catch(error => {  
    console.log(error);  
    res.status(404).json({  
      message: error  
    });  
  });  
});
```



```

}

exports.callback = (req, res) => {
  const form = new formidable.IncomingForm();
  form.parse(req, (error, fields, file) => {
    if (error) {
      console.log(error);
      res.status(500).json({ error });
    }
    let checksumHash = fields.CHECKSUMHASH;
    delete fields.CHECKSUMHASH;

    // verify the signature
    var isVerifySignature = PaytmChecksum.verifySignature(
      fields,
      process.env.PAYTM_MERCHANT_KEY,
      checksumHash
    )

    if (isVerifySignature) {
      // response is valid
      // get the transaction status
      // any communication with Paytm, server has to be secure
      var params = {};
      params["MID"] = fields.MID;
      params["ORDER_ID"] = fields.ORDERID;

      PaytmChecksum.generateSignature(
        params,
        process.env.PAYTM_MERCHANT_KEY
      ).then(checksum => {
        // go to paytm server and get the status
        params["CHECKSUMHASH"] = checksum;
        var data = JSON.stringify(params);
        var options = {
          hostname: "securegw-stage.paytm.in",
          port: 443,
          path: "/order/status",
          method: "POST",
          headers: {
            'Content-Type': "application/json",
            'Content-Length': data.length
          }
        }
      });

      var response = "";
      var request = https.request(options, (responseFromPaytm) => {
        responseFromPaytm.on('data', (chunk) => {
          response += chunk;
        });
      });
    }
  });
}

```

```

    });
    responseFromPaytm.on('end', () => {
        console.log(response);
        res.json(response);
    });
    });
    request.write(data);
    request.end();
    })
} else {
    console.log("Checksum Mismatch");
    res.status(500).json({ error: "Its a hacker !" });
}

});
}

```

7. PaytmChecksum.js

```

"use strict";

var crypto = require('crypto');

class PaytmChecksum {

    static encrypt(input, key) {
        var cipher = crypto.createCipheriv('AES-128-CBC', key, PaytmChecksum.iv);
        var encrypted = cipher.update(input, 'binary', 'base64');
        encrypted += cipher.final('base64');
        return encrypted;
    }

    static decrypt(encrypted, key) {
        var decipher = crypto.createDecipheriv('AES-128-CBC', key,
PaytmChecksum.iv);
        var decrypted = decipher.update(encrypted, 'base64', 'binary');
        try {
            decrypted += decipher.final('binary');
        }
        catch (e) {
            console.log(e);
        }
        return decrypted;
    }

    static generateSignature(params, key) {

```

```

        if (typeof params !== "object" && typeof params !== "string") {
            var error = "string or object expected, " + (typeof params) + "
given.";
            return Promise.reject(error);
        }
        if (typeof params !== "string"){
            params = PaytmChecksum.getStringByParams(params);
        }
        return PaytmChecksum.generateSignatureByString(params, key);
    }

    static verifySignature(params, key, checksum) {
        if (typeof params !== "object" && typeof params !== "string") {
            var error = "string or object expected, " + (typeof params) + "
given.";
            return Promise.reject(error);
        }
        if(params.hasOwnProperty("CHECKSUMHASH")){
            delete params.CHECKSUMHASH
        }
        if (typeof params !== "string"){
            params = PaytmChecksum.getStringByParams(params);
        }
        return PaytmChecksum.verifySignatureByString(params, key, checksum);
    }

    static async generateSignatureByString(params, key) {
        var salt = await PaytmChecksum.generateRandomString(4);
        return PaytmChecksum.calculateChecksum(params, key, salt);
    }

    static verifySignatureByString(params, key, checksum) {
        var paytm_hash = PaytmChecksum.decrypt(checksum, key);
        var salt = paytm_hash.substr(paytm_hash.length - 4);
        return (paytm_hash === PaytmChecksum.calculateHash(params, salt));
    }

    static generateRandomString(length) {
        return new Promise(function (resolve, reject) {
            crypto.randomBytes((length * 3.0) / 4.0, function (err, buf) {
                if (!err) {
                    var salt = buf.toString("base64");
                    resolve(salt);
                }
                else {
                    console.log("error occurred in generateRandomString: " + err);
                    reject(err);
                }
            })
        })
    }

```

```

    });
  });
}

static getStringByParams(params) {
  var data = {};
  Object.keys(params).sort().forEach(function(key,value) {
    data[key] = (params[key] !== null && params[key].toLowerCase() !==
"null") ? params[key] : "";
  });
  return Object.values(data).join('|');
}

static calculateHash(params, salt) {
  var finalString = params + "|" + salt;
  return crypto.createHash('sha256').update(finalString).digest('hex') +
salt;
}
static calculateChecksum(params, key, salt) {
  var hashString = PaytmChecksum.calculateHash(params, salt);
  return PaytmChecksum.encrypt(hashString,key);
}
}
PaytmChecksum.iv = '@@@@&&&####$$$$';
module.exports = PaytmChecksum;

```

8.Restaurant.js

```

const Restaurant = require('../models/Restaurant');

const restaurantList = [{}, {}];

exports.filter = (req , res) => {
  //const location_id = req.body.location_id;
  const mealtype_id = req.body.mealtype_id;
  const cuisine_ids = req.body.cuisine_ids;
  //const city_id = req.body.city_id;
  //const cost = req.body.cost;
  //const page = req.body.page;

  let payload={
    'type.0.mealtype' : mealtype_id
  }

  if (cuisine_ids && cuisine_ids.length > 0) {
    payload['Cuisine.cuisine'] = {

```

```

        $in: cuisine_ids
    };
}

Restaurant.find(payload)
    .then(result => {
console.log(result);

res.status(200).json({
    message: "filtered List Fetched",
    restaurants : result
});
}).catch(error => {
    console.log(error);
    res.status(500).json({
        message: error
    });
})
}

exports.getRestaurantById = (req,res) => {

    Restaurant.find({
        _id: req.params.id
    }).then(result => {
        res.status(200).json({
            restaurant: result[0],
            message: "Success"
        });
    }).catch(error => {
        res.status(500).json({
            message: error
        });
    });
}

exports.getRestaurantsByCity = (req,res) => {
    var result = restaurantList.filter(item => item.city_name ==
req.params.city_name);

    res.status(200).json({
        restaurants : result,
        message: "Success"

    });
}
}

```

9.User.js

```
const User = require('../Models/User');

exports.signUp = (req, res) => {

  const email = req.body.email;
  const password = req.body.password;
  const firstname = req.body.firstname;
  const lastname = req.body.lastname;

  const signUpUser = new User({
    email: email,
    password: password,
    firstname: firstname,
    lastname: lastname
  });

  signUpUser.save().then(
    result => {
      res.status(200).json({
        message: "User signed up successfully !",
        user: result
      });
    }
  ).catch(
    error => {
      res.status(500).json({
        message: "Please Try Again"
      });
    }
  );
}

exports.logIn = (req, res) => {
  const email = req.body.email;
  const password = req.body.password;

  User.find({
    email: email,
    password: password,
  }).then(
    result => {
      if (result.length >= 1) {
        res.status(200).json({
          message: "User logged in successfully !",
          isAuthenticated: true,

```

```

        user: result
    });
} else {
    res.status(200).json({
        message: "User NOT logged in successfully !",
        isAuthenticated: false,
        user: result
    });
}
}
).catch(
    error => {
        res.status(500).json({
            message: error
        });
    }
);
}

```

B. Models

1. City.js

```

const mongoose = require('mongoose');

const Schema = mongoose.Schema;

const citySchema = new Schema({
  _id: {
    type: String,
    required: true
  },
  name: {
    type: String,
    required: true
  },
  city_id: {
    type: Number,
    required: true
  },
  location_id: {
    type: Number,
    required: true
  },
  country_name: {

```

```

        type: String,
        required: true
    }
});

module.exports = mongoose.model('Location', citySchema , 'Location');
```

2. Mealtype.js

```

const mongoose = require('mongoose');

const Schema = mongoose.Schema;

const MealsSchema = new Schema({
  _id: {
    type: String,
    required: true
  },
  name: {
    type: String,
    required: true
  },
  content: {
    type: String,
    required: true
  },
  image: {
    type:String,
    required: true
  },
});

module.exports = mongoose.model('Mealtype', MealsSchema , 'Mealtype');
```

3.Menu.js

```

const mongoose = require('mongoose');

const Schema = mongoose.Schema;

const menuSchema = new Schema({
  item: {
```



```

        type: String,
        required: true
    },
    cost: {
        type: String,
        required: true
    },
    description: {
        type: String,
        required: true
    },
    restaurantId: {
        type: String,
        required: true
    }
  }
});

module.exports = mongoose.model('menu', menuSchema, 'menu');

```

4.Order.js

```

const mongoose = require('mongoose');

const Schema = mongoose.Schema;

const orderSchema = new Schema({
  username: {
    type: String,
    required: true
  },
  mobileNumber: {
    type: String,
    required: true
  },
  address: {
    type: String,
    required: true
  },
  items: {
    type: String,
    required: true
  },
  total: {
    type: String,
    required: true
  }
});

```

```
    }  
  });  
  
module.exports = mongoose.model('orders', orderSchema, 'orders');
```

5.) Restaurant.js

```
const mongoose = require('mongoose');  
  
const Schema = mongoose.Schema;  
  
const restaurantSchema = new Schema({  
  _id: {  
    type: String,  
    requiredjs: true  
  },  
  name: {  
    type: String,  
    required: true  
  },  
  city_name: {  
    type: String,  
    required: true  
  },  
  city: {  
    type: String,  
    required: true  
  },  
  area: {  
    type: String,  
    required: true  
  },  
  locality: {  
    type: String,  
    required: true  
  },  
  thumb: {  
    type: String,  
    required: true  
  },  
  cost: {  
    type: Number,  
    required: true  
  },  
  adress: {  
    type: String,
```

```

        required: true
      },
      type: [{
        mealtype: {
          type: String,
          required: true
        },
        name: {
          type: String,
          required: true
        }
      }
    ]],
    cuisine: [{
      cuisine: {
        type: String,
        required: true
      },
      name: {
        type: String,
        required: true
      }
    }
  ]
});

module.exports = mongoose.model('RestaurantData', restaurantSchema ,
'RestaurantData');
```

6. User.js

```

const mongoose = require('mongoose');

const Schema = mongoose.Schema;

const UserSchema = new Schema({
  email: {
    type: String,
    required: true
  },
  password: {
    type: String,
    required: true
  },
  firstname: {
```

```

        type: String,
        required: true
    },
    lastname: {
        type: String,
        required: true
    }
});

module.exports = mongoose.model('User', UserSchema);

```

c. Routes.js

```

const express = require('express');
const cityController = require('../controllers/city');
const RestaurantController = require('../controllers/Restaurant');
const MealController = require('../controllers/mealtype');
const userController = require('../controllers/User');
const menuController = require('../controllers/Menu');
const orderController = require('../controllers/Orders');
const paymentGatewayController =
require('../controllers/paymentGatewayController');

const router = express.Router();

router.get('/cityList',cityController.getCityList);
router.get('/getRestaurantsByCity/:city', cityController.getRestaurantsByCity);
router.get('/mealList',MealController.getMeals);
router.post('/restaurantFilter',RestaurantController.filter);
router.post('/signup',userController.signUp);
router.post('/login',userController.logIn);
router.get('/getRestaurantById/:id', RestaurantController.getRestaurantById);
router.get('/getMenuForRestaurant/:id', menuController.getMenuForRestaurant);
router.get('/getOrdersForUser',orderController.getOrdersForUser);
router.post('/saveOrdersForUser/:username',orderController.saveOrderForUser);
router.post('/payment', paymentGatewayController.payment);
router.post('/callback', paymentGatewayController.callback);

module.exports = router;

```

D. Validations

```

exports.validateUserSignup = (signupdata) => {
  if (!signupdata) {
    return false;
  }
  if (!signupdata.email) {
    return false;
  }
  if (!signupdata.password) {
    return false;
  }
}

```

E. App.js

```

const express = require('express');
const bodyparser = require('body-parser');
const mongoose = require('mongoose');
const path = require('path')

const apiRouter = require('./routes/router');

const port = 5000;
const app = express();

app.use(bodyparser.json());
app.use(express.json());

app.use((req, res, next) => {
  res.setHeader('Access-Control-Allow-Origin', '*');
  res.setHeader('Access-Control-Allow-Methods', 'GET, POST');
  res.setHeader('Access-Control-Allow-Headers', 'Content-Type, Authorization');
  next();
});

app.use('/api', apiRouter);

app.use(express.static(path.join(__dirname, '../build')));
app.get('*', (req, res) => {
  res.sendFile(path.join(__dirname, '../build'));
});

```

```

mongoose.connect(

  'mongodb+srv://Shrey_Srivastava:Edureka2611@cluster0.0d9at.mongodb.net/Zomato?retryWrites=true&w=majority',
  { useNewUrlParser: true, useUnifiedTopology: true }
).then(success => {

  console.log('connected to MongoDB');
  app.listen(port, () => {

    console.log(`server is running at: ${port}`)
  });
}).catch(error => {
  console.log(error);
});
•

```

II. Frontend

A. Assets

Contains all images

B. Components

a) Details.js

```

import React from 'react';
import axios from 'axios';
import queryString from 'query-string';
import { Tab, Tabs, TabList, TabPanel } from 'react-tabs';
import 'react-tabs/style/react-tabs.css';
import Modal from 'react-modal';
import Header from './Header';

const customStyles = {
  content: {
    top: '50%',
    left: '50%',
    right: 'auto',
    bottom: 'auto',
    marginRight: '-50%',
    transform: 'translate(-50%, -50%)',
    backgroundColor: 'white',
  }
}

```

```

        border: 'solid 2px tomato'
    }
};

class Details extends React.Component {

    constructor() {
        super();
        this.state = {
            restaurant: {},
            menu: [],
            isModalOpen: false,
            order: []
        };
    }

    componentDidMount() {
        const queryParams = queryString.parse(this.props.location.search);
        const restaurantId = queryParams.id;

        axios({
            method: 'GET',
            url: `http://localhost:5000/api/getRestaurantById/${restaurantId}`,
            headers: { 'Content-Type': 'application/json' }
        }).then(result => {
            this.setState({
                restaurant: result.data.restaurant
            });
        }).catch(error => {
            console.log(error);
        });

        axios({
            method: 'GET',
            url: `http://localhost:5000/api/getMenuForRestaurant/${restaurantId}`,
            headers: { 'Content-Type': 'application/json' }
        }).then(result => {
            this.setState({
                menu: result.data.menu
            });
        }).catch(error => {
            console.log(error);
        });
    }

    handlePlaceOrder = (e) => {
        // open a modal to show the menu
        this.setState({
            isModalOpen: true
        });
    }
}

```

```

    });
  }

  render() {
    const { restaurant, isModalOpen, menu } = this.state;
    return (
      <React.Fragment>
        <Header />
        <div className="container mb-5">
          <img src={restaurant.thumb} alt="restaurant" width="100%"
height="500px" className="mt-5"/>
          <h2 className="mt-3">{restaurant.name}</h2>
          <div style={{'position': 'absolute', 'right':
'160px'}}><button type="button" onClick={this.handlePlaceOrder}>Place online
order</button></div>
          <div className="mt-3">
            <Tabs>
              <TabList>
                <Tab>Overview</Tab>
                <Tab>Contact</Tab>
              </TabList>
              <TabPanel>
                <h3>About this place</h3>
                <h4>Cuisine</h4>
                <div>
                  {
                    restaurant.Cuisine &&
restaurant.Cuisine.length > 0
                    ?
                    restaurant.Cuisine.map(item => {
                      return <span>{ item.name }, </span>
                    })
                    :
                    null
                  }
                </div>
                <h4 className="mt-3">Average Cost</h4>
                <div>{restaurant.cost}</div>
              </TabPanel>
              <TabPanel>
                <h4>Phone Number</h4>
                <div>(+91) 9986851333</div>
                <h4 className="mt-3">Address</h4>
                <h5>{restaurant.name}</h5>
                <div>{restaurant.address}</div>
              </TabPanel>
            </Tabs>
            <Modal
              isOpen={isModalOpen}

```



```

        style={customStyles}>
        <div>
            {
                menu.map((item, index) => {
                    return (
                        <div className="row">
                            <div className="col-6" style={{'color':
'white'}}>{item.item}</div>
                                <div className="col-2"
style={{'color': 'white'}}>{item.cost}</div>
                                <div className="col-
4"><button>Add</button></div>
                            </div>
                        )
                    })
                }
            </div>
            <div className="row mt-3">
                <div className="float-left">
                    Sub total: 220
                </div>
                <div className="float-right">
                    <button>Pay Now</button>
                </div>
            </div>
        </Modal>
    </div>
</div>
</React.Fragment>
)
}
}
export default Details;

```

b) Filter.js

```

import React from 'react';
import '../styles/Filter.css';
import queryString from 'query-string';
import axios from 'axios';
import 'bootstrap/dist/css/bootstrap.min.css';
import Header from './Header';

class Filter extends React.Component {
    constructor() {

```

```

    super();
    this.state = {
      restaurantList: [],
      locationList: [],
      pageCount: [],
      location: undefined,
      cuisine: [],
      mealtype: undefined,
      hcost: undefined,
      lcost: undefined,
      page: 1,
      sort: 1
    }
  }

  componentDidMount() {
    const qs = queryString.parse(this.props.location.search);
    const { mealtype } = qs;
    const req = {
      "mealtype_id": mealtype
    }
    axios({
      method: 'POST',
      url: 'http://localhost:5000/api/restaurantFilter',
      headers: { 'Content-Type': 'application/json' },
      data: req
    }).then(result => {
      this.setState({
        restaurantList: result.data.restaurants,
        mealtype: mealtype,
      })
    }).catch(error => {
      console.log({error});
    })
  }

  handleCuisineChange(cuisineId) {
    const {
      cuisine = [],
      location,
      mealtype,
      hcost,
      lcost,
      sort,
      page
    } = this.state;

    if (cuisine.indexOf(cuisineId.toString()) == -1) {
      cuisine.push(cuisineId.toString());
    }
  }

```

```

    } else {
      var index = cuisine.indexOf(cuisineId.toString());
      cuisine.splice(index, 1);
    }

    let filterObj = {
      cuisine_ids: cuisine,
      mealtype_id: mealtype
    }

    axios({
      method: 'POST',
      url: 'http://localhost:5000/api/restaurantFilter',
      headers: { 'Content-Type': 'application/json' },
      data: filterObj
    }).then(result => {
      this.setState({
        restaurantList: result.data.restaurants,
      })
    }).catch(error => {
      console.log({error});
    });
  }
  onSortChange(priceR){
    const {
      restaurantList = [],
      cost,
      mealtype,
      hcost,
      lcost,
      sort,
      page
    } = this.state;

    if (cost.indexOf(priceR.toString()) == -1) {
      cost.push(priceR.toString());
    } else {
      var index = cost.indexOf(priceR.toString());
      cost.splice(index, 1);
    }

    let filterObj1 = {
      cuisine_ids: cost,
      mealtype_id: mealtype
    }

    axios({
      method: 'POST',
      url: 'http://localhost:5000/api/restaurantFilter',

```

```

        headers: { 'Content-Type': 'application/json' },
        data: filterObj1
    }).then(result => {
        this.setState({
            restaurantList: result.data.restaurants,
        })
    }).catch(error => {
        console.log({error});
    });
}

handleCostChange() {

}

render() {
    const { restaurantList, locationList, pageCount, sort } = this.state;
    return(
        <div>
            <Header />
            <div id="myId" className="heading">{item.name} Places in
Delhi</div>

            <div className="container-fluid">
                <div className="row">
                    <div className="col-sm-3 col-md-3 col-lg-3">
                        <div className="filter-options">
                            <span className="glyphicon glyphicon-th-list
toggle-span" data-toggle="collapse"
                                data-target="#demo"></span>
                            <div id="demo" className="collapse show">
                                <div className="filter-heading">Filters</div>
                                <div className="Select-Location">Select
Location</div>

                                <select className="Rectangle-2236"
onChange={this.handleLocationChange}>
                                    <option>Select</option>
                                    {locationList.map((item) => {
                                        return <option
value={` ${item.location_id}-${item.city_id}`}>{` ${item.name},
${item.city}`}</option>

                                    })}
                                </select>
                                <div className="Cuisine">Cuisine</div>
                                <div>
                                    <input type="checkbox" value="1"
onChange={() => this.handleCuisineChange(1)} />
                                    <span className="checkbox-items">North
Indian</span>

```

```

        </div>
        <div>
            <input type="checkbox" onChange={() =>
this.handleCuisineChange(2)} />
            <span className="checkbox-items">South
Indian</span>
        </div>
        <div>
            <input type="checkbox" onChange={() =>
this.handleCuisineChange(3)} />
            <span className="checkbox-items">Chineese</span>
        </div>
        <div>
            <input type="checkbox" onChange={() =>
this.handleCuisineChange(4)} />
            <span className="checkbox-items">Fast
Food</span>
        </div>
        <div>
            <input type="checkbox" onChange={() =>
this.handleCuisineChange(5)} />
            <span className="checkbox-items">Street
Food</span>
        </div>
        <div className="Cuisine">Cost For Two</div>
        <div>
            <input type="radio" name="cost"
onChange={() => this.handleCostChange(1, 500)} />
            <span className="checkbox-items">Less than
&#8377; 500</span>
        </div>
        <div>
            <input type="radio" name="cost"
onChange={() => this.handleCostChange(500, 1000)} />
            <span className="checkbox-items">&#8377;
500 to &#8377; 1000</span>
        </div>
        <div>
            <input type="radio" name="cost"
onChange={() => this.handleCostChange(1000, 1500)} />
            <span className="checkbox-items">&#8377;
1000 to &#8377; 1500</span>
        </div>
        <div>
            <input type="radio" name="cost"
onChange={() => this.handleCostChange(1500, 2000)} />
            <span className="checkbox-items">&#8377;
1500 to &#8377; 2000</span>

```

```

        </div>
        <div>
            <input type="radio" name="cost"
onChange={() => this.handleCostChange(2000, 10000)} />
            <span className="checkbox-items">&#8377;
2000 +</span>
        </div>
        <div>
            <input type="radio" name="cost"
onChange={() => this.handleCostChange(1, 10000)} />
            <span className="checkbox-items">All</span>
        </div>
        <div className="Cuisine">Sort</div>
        <div>
            <input type="radio" name="sort"
checked={sort == 1} onChange={() => this.onSortChange(1)} />
            <span className="checkbox-items">Price low
to high</span>
        </div>
        <div>
            <input type="radio" name="sort"
checked={sort == -1} onChange={() => this.onSortChange(-1)} />
            <span className="checkbox-items">Price
high to low</span>
        </div>
    </div>
    </div>
    <div className="res">
        {
            restaurantList.length > 0
            ?
            restaurantList.map((item, index) => {
                return (<div className="Item" key={index}
onClick={() => this.handleClick(item)}>
                    <div className="row pl-1">
                        <div className="col-sm-4 col-md-4 col-lg-
4">
                            <img className="img"
src={require('../assets/breakfast.png').default} alt=""/>
                        </div>
                        <div className="col-sm-8 col-md-8 col-lg-
8">
                            <div className="rest-
name">{item.name}</div>
                            <div className="res-
location">{item.locality}, {item.city_name}</div>

```

```

        <div className="rest-
address">{item.address}</div>
        </div>
    </div>
    <hr />
    <div className="row padding-left">
        <div className="col-sm-12 col-md-12 col-
lg-12">
            <div className="rest-address">CUISINES
: {item.Cuisine.map(itm => itm.name + ', ')}</div>
            <div className="rest-address">COST FOR
TWO : {item.cost}</div>
        </div>
    </div>
</div>)
})
: <div className="rect">
    <div className="no">Sorry, No Data Found
</div></div>
    }

    </div>
    </div>

    </div>
    </div>
    </div>
    );
}
}
export default Filter;

```

c) Header.js

```
import React from 'react';
import '../styles/header.css';
import 'bootstrap/dist/css/bootstrap.css';
import Modal from 'react-modal';
import axios from 'axios';
import logo from '../assets/logo.png';

const customStyles = {
  content: {
    top: '50%',
    left: '50%',
    right: 'auto',
```

```

        bottom: 'auto',
        marginRight: '-50%',
        transform: 'translate(-50%, -50%)',
        backgroundColor: 'white',
        border: 'solid 2px tomato'
    }
};

class Header extends React.Component {
    constructor() {
        super();
        this.state = {
            isModalOpen: false,
            isSigninModalOpen: false,
            email: '',
            password: '',
            firstname: '',
            lastname: '',
            isLoggedIn: false
        };
    }

    handleChange = (event, stateVariable) => {
        this.setState({
            [stateVariable]: event.target.value
        });
    }

    signUpHandler = (event) => {
        const {
            email,
            password,
            firstname,
            lastname
        } = this.state;

        const signUpRequestObj = {
            email: email,
            password: password,
            firstname: firstname,
            lastname: lastname
        }

        axios({
            method: 'POST',
            url: 'http://localhost:5000/api/signup',
            headers: {'Content-Type': 'application/json'},
            data: signUpRequestObj
        }).then(
            response => {

```



```

        if (response.data.message == 'User signed up successfully !') {
            this.setState({
                isModalOpen: false,
                email: '',
                password: '',
                firstname: '',
                lastname: ''
            });
            alert(response.data.message);
        }
    }
).catch(
    error => {
        console.log(error);
        alert(error);
    }
)
}

loginHandler = (event) => {
    const {
        email,
        password
    } = this.state;

    const loginRequestObj = {
        email: email,
        password: password
    }

    axios({
        method: 'POST',
        url: 'http://localhost:5000/api/login',
        headers: {'Content-Type': 'application/json'},
        data: loginRequestObj
    }).then(
        response => {
            if (response.data.user.length >= 1) {
                this.setState({
                    isSigninModalOpen: false,
                    email: '',
                    password: '',
                    isLoggedIn: response.data.isAuthenticated,
                    firstname: response.data.user[0].firstname
                });
                sessionStorage.setItem('isLoggedIn',
response.data.isAuthenticated);
            }
        }
    ).catch(

```

```

        error => {
            console.log(error);
            alert(error);
        }
    )
}

cancelHandler = (event) => {
    this.setState({
        isModalOpen: false,
    });
}

signUpOpenHandler = (event) => {
    this.setState({
        isModalOpen: true,
    });
}

signInOpenHandler = (event) => {
    this.setState({
        isSigninModalOpen: true,
    })
}

cancelLoginHandler = (event) => {
    this.setState({
        isSigninModalOpen: false,
    })
}

logoutHandler = (event) => {
    this.setState({
        firstname: '',
        isLoggedIn: false
    });
    sessionStorage.setItem('isLoggedIn', false);
}

render() {
    const {
        isModalOpen,
        isSigninModalOpen,
        email,
        password,
        firstname,
        lastname,
        isLoggedIn
    } = this.state;

```

```

return (
  <React.Fragment>
    <div className="header">
      <div>
        <a href="http://localhost:3000/">
          <img src={logo} alt="" className="ima" /></a>
        </div>
        <div className="btn-group login-block">
          {
            isLoggedIn
            ?
            <div>
              <span>{firstname}</span>
              <button className="btn btn-sm btn-primary ml-3"
onClick={this.logoutHandler}>Logout</button>
            </div>
            :
            <div>
              <button className="btn btn-sm text-white"
onClick={this.signInOpenHandler}>Login</button>
              <button className="btn default btn-sm text-white"
onClick={this.signUpOpenHandler}>Create an account</button>
            </div>
          }
        </div>
      <Modal
        isOpen={isModalOpen}
        style={customStyles}>
        <div>
          <h3>Signup User</h3>
          <div>
            <span>Email : </span>
            <input type="text" value={email} onChange={(event)
=> this.handleChange(event, 'email')}></input>
          </div>
          <div>
            <span>Password : </span>
            <input type="password" value={password}
onChange={(event) => this.handleChange(event, 'password')}></input>
          </div>
          <div>
            <span>First Name : </span>
            <input type="text" value={firstname}
onChange={(event) => this.handleChange(event, 'firstname')}></input>
          </div>
          <div>
            <span>Last Name : </span>
            <input type="text" value={lastname}
onChange={(event) => this.handleChange(event, 'lastname')}></input>

```

```

        </div>
        <button onClick={this.signUpHandler} class="btn btn-sm
btn-primary">Signup</button>
        <button onClick={this.cancelHandler} class="btn btn-sm
btn-primary">Cancel</button>
    </div>
</Modal>
<Modal
    isOpen={isSigninModalOpen}
    style={customStyles}>
    <div>
        <h3>Login User</h3>
        <div>
            <span>Email : </span>
            <input type="text" value={email} onChange={(event)
=> this.handleChange(event, 'email')}></input>
        </div>
        <div>
            <span>Password : </span>
            <input type="password" value={password}
onChange={(event) => this.handleChange(event, 'password')}></input>
        </div>
        <button onClick={this.loginHandler} class="btn btn-sm
btn-primary">Login</button>
        <button onClick={this.cancelLoginHandler} class="btn
btn-sm btn-primary">Cancel</button>
    </div>
</Modal>
</div>
</React.Fragment>
    )
}
}

export default Header;

```

d) Home.js

```

import React from 'react';
import '../styles/Home.css';
import Wallpaper from './Wallpaper';
import QuickSearches from './QuickSearches';
import axios from 'axios';
import 'bootstrap/dist/css/bootstrap.min.css';

class Home extends React.Component {

```

```

    constructor(){
        super();
        this.state = {
            cities: [],
            mealtypes: []
        };
    }
    componentDidMount(){
        axios.get('http://localhost:5000/api/CityList').then(result => {
            this.setState({
                cities: result.data.cities
            })
        }).catch(error => {
            console.log(error);
        });

        axios.get('http://localhost:5000/api/mealList').then(result => {
            this.setState({
                mealtypes: result.data.meals
            })
        }).catch(error => {
            console.log(error);
        });
    }

    render(){
        const {cities, mealtypes} = this.state;
        return(
            <React.Fragment>
                <Wallpaper cities={cities} />
                <QuickSearches mealtypes={mealtypes} />
            </React.Fragment>
        );
    }
}

export default Home;

```

e) Navigation_name.js

```
import React from 'react';
import { DocumentTitle } from 'react-document-title';

class navigation_name extends React.Component {
  render() {
    return (
      <DocumentTitle title={ this.props.title }>
        { this.props.title }
      </DocumentTitle>
    );
  }
}

export default navigation_name;
```

f) QuickSearches.js

```
import React from 'react';
import QuickSearchItem from '../components/QuickSearchItem';

class QuickSearches extends React.Component {
  constructor() {
    super();
  }
  render() {
    const { mealtypes } = this.props;
    return (
      <React.Fragment>
        <div className="quicksearch">
          <p className="quicksearchHeading">
            Quick Searches
          </p>
          <p className="quicksearchSubHeading">
            Discover restaurants by type of meal
          </p>
          <div className="container-fluid">

            <div className="row">
              {
                mealtypes && mealtypes.length > 0 ? (
                  mealtypes.map((mealtype, index) => {
```

```

                                return <QuickSearchItem key={index} id={index
+ 1} mealtype={mealtype} />
                                ))) : null
                                }
                                </div>
                                </div>
                                </div>
                                </React.Fragment>
                                );
                                }
                                }

export default QuickSearches;

```

g) QuickSearchitem.js

```

import React from 'react';
import { withRouter } from 'react-router-dom';

class QuickSearchItem extends React.Component {
  constructor() {
    super();
  }

  handleClick(id) {
    this.props.history.push(`/filter?mealtype=${id}`);
  }

  render() {
    const { id, mealtype } = this.props;
    const { name, content, image } = mealtype;
    const imagePath = require(`../${image}`).default;
    return (
      <div className="col-sm-12 col-md-4 col-lg-4 " onClick={() =>
this.handleClick(id)}>
        <div className="tileContainer clickable-pointer" >
          <div className="tileComponent1">
            <img src={imagePath} alt="" height="150" width="140" />
          </div>
          <div className="tileComponent2">
            <div className="componentHeading">
              { name }
            </div>
            <div className="componentSubHeading">
              { content }
            </div>
          </div>
        </div>
      </div>
    );
  }
}

```

```

        </div>
      </div>
    );
  }
}

export default withRouter(QuickSearchItem);

```

h) Restaurant.js

```

import React from 'react';
import '../styles/Restaurant.css';
import 'bootstrap/dist/css/bootstrap.css';
import axios from 'axios';
import queryString from 'query-string';
import { Tab, Tabs, TabList, TabPanel } from 'react-tabs';
import 'react-tabs/style/react-tabs.css';
import Modal from 'react-modal';
import Header from './Header';

const customStyles = {
  content: {
    top: '50%',
    left: '50%',
    right: 'auto',
    bottom: 'auto',
    marginRight: '-50%',
    transform: 'translate(-50%, -50%)',
    backgroundColor: 'lightslategrey',
    border: 'solid 2px tomato'
  }
};

class Restaurant extends React.Component {
  constructor() {
    super();
    this.state = {
      restaurant: {},
      menu: [],
      isModalOpen: false,
      order: []
    };
  }

  componentDidMount() {
    const queryParams = queryString.parse(this.props.location.search);
    const restaurantId = queryParams.id;

```



```

    axios({
      method: 'GET',
      url: `http://localhost:5000/api/getRestaurantById/${restaurantId}`,
      headers: { 'Content-Type': 'application/json' }
    }).then(result => {
      this.setState({
        restaurant: result.data.restaurant
      });
    }).catch(error => {
      console.log(error);
    });

    axios({
      method: 'GET',
      url: `http://localhost:5000/api/getMenuForRestaurant/${restaurantId}`,
      headers: { 'Content-Type': 'application/json' }
    }).then(result => {
      this.setState({
        menu: result.data.menu
      });
    }).catch(error => {
      console.log(error);
    });
  }

  handlePlaceOrder = (e) => {

    this.setState({
      isModalOpen: true
    });
  }

  cancelPayment = () => {
    this.setState({
      isModalOpen: false
    });
  }

  getData = (data) => {
    return fetch(`http://localhost:5000/api/payment`, {
      method: 'POST',
      headers: {
        Accept: "application/json",
        "Content-Type": "application/json"
      },
      body: JSON.stringify(data)
    }).then(response => {
      return response.json();
    });
  }

```

```

    }).catch(error => {
        console.log(error);
    });
}

obj = (val) => {
    return typeof val === 'object';
}

isDate = (val) => {
    return Object.prototype.toString.call(val) === '[object Date]';
}

stringifyMyParam = (paramValue) => {
    if (this.obj(paramValue) && !this.isDate(paramValue)) {
        return JSON.stringify(paramValue);
    } else {
        return paramValue;
    }
}

buildForm = (details) => {
    const { action, params } = details;

    const form = document.createElement('form');
    form.setAttribute('method', 'post');
    form.setAttribute('action', action);

    Object.keys(params).forEach(key => {
        const input = document.createElement('input');
        input.setAttribute('type', 'hidden');
        input.setAttribute('name', key);
        input.setAttribute('value', this.stringifyMyParam(params[key]));
        form.appendChild(input);
    });

    return form;
}

takeMeToPaymentGateway = (details) => {
    const form = this.buildForm(details);
    document.body.appendChild(form);
    form.submit();
    form.remove();
}

makePayment = (rc) => {

```

```

    this.getData({
      amount: rc,
      email: '123@gmail.com',
      mobileNo: '9986851333'
    }).then(response => {
      var information = {
        action: "https://securegw-stage.paytm.in/order/process",
        params: response.checkSumResponse
      };
      this.takeMeToPaymentGateway(information);
    }).catch(error => {
      console.log(error);
    });
  }

  render() {
    const { restaurant, isModalOpen, menu } = this.state;
    return (
      <React.Fragment>
        <Header />
        <div className="container mb-5">
          <img src={restaurant.thumb} alt="restaurant" width="100%"
height="500px" className="mt-5"/>
          <h2 className="mt-3">{restaurant.name}</h2>
          <div style={{ 'position': 'absolute', 'right':
'160px'}}><button type="button" class="btn btn-danger"
onClick={this.handlePlaceOrder}>Place online order</button></div>
          <div className="mt-3">
            <Tabs>
              <TabList>
                <Tab>Overview</Tab>
                <Tab>Contact</Tab>
              </TabList>
              <TabPanel>
                <h3>About this place</h3>
                <h4>Cuisine</h4>
                <div>
                  {
                    restaurant.Cuisine &&
restaurant.Cuisine.length > 0
                    ?
                    restaurant.Cuisine.map(item => {
                      return <span>{ item.name }, </span>
                    })
                    :
                    null
                  }
                </div>
              </TabPanel>
            </Tabs>
          </div>
        </div>
      </React.Fragment>
    );
  }
}

```

```

        <h4 className="mt-3">Average Cost</h4>
        <div>{restaurant.cost}</div>
      </TabPanel>
      <TabPanel>
        <h4>Phone Number</h4>
        <div>(+91) 14004566 </div>
        <h4 className="mt-3">Address</h4>
        <h5>{restaurant.name}</h5>
        <div>{restaurant.address}</div>
      </TabPanel>
    </Tabs>
    <Modal
      isOpen={isModalOpen}
      style={customStyles}>
      <div>
        {
          menu.map((item, index) => {
            return (
              <div className="row">
                <div className="col-6"
style={{'color': 'white'}}>{item.item}</div>
                <div className="col-2"
style={{'color': 'white'}}>{item.cost}</div>
                <div className="col-4"><button
class="btn btn-info" >Add</button></div>
              </div>
            )
          })
        }
      </div>
      <div className="row mt-3">
        <div className="float-left">
          <h4 className="total">Sub total:
{restaurant.cost}</h4>
        </div>
        <div className="float-right">
          <button onClick={this.cancelPayment}
class="btn btn-danger">Cancel</button>
          <button onClick={() =>
this.makePayment(restaurant.cost)}class="btn btn-success">Pay Now</button>
        </div>
      </div>
    </Modal>
  </div>
</div>
</React.Fragment>
)
}
}

```

```
export default Restaurant;
```

i) Transaction.js

```
import React from 'react';
import '../styles/header.css';
import axios from 'axios';

class Transaction extends React.Component {
  constructor() {
    this.status = {
      paymentStatus: ""
    };
  }

  componentDidMount() {
    this.state={paymentStatus}
  }

  render() {
    const { paymentStatus } = this.state;
    return (
      <React.Fragment>
        <h1>Trasaction Status</h1>
        <h2>{ paymentStatus }</h2>
      </React.Fragment>
    )
  }
}

export default Transaction;
```

j) Wallpaper.js

```
import React from 'react';
```

```

import homepageimg from '../assets/wallpaper.png';
import logo from '../assets/logo.png';
import breakfast from '../assets/breakfast.png'
import axios from 'axios';
import { withRouter } from 'react-router-dom';
import '../styles/header.css';
import 'bootstrap/dist/css/bootstrap.css';
import Modal from 'react-modal';

const customStyles = {
  content: {
    top: '50%',
    left: '50%',
    right: 'auto',
    bottom: 'auto',
    marginRight: '-50%',
    transform: 'translate(-50%, -50%)',
    backgroundColor: 'white',
    border: 'solid 2px tomato'
  }
};

class Wallpaper extends React.Component {
  constructor() {
    super();
    this.state = {
      suggestions: [],
      text: '',
      restaurants: [],
      isModalOpen: false,
      isSigninModalOpen: false,
      email: '',
      password: '',
      firstname: '',
      lastname: '',
      isLoggedIn: false
    };
  }

  componentDidMount() {

  }

  handleChange = (event) => {
    const cityId = event.target.selectedOptions[0].value;
    sessionStorage.setItem("city", cityId);
    axios({
      method: 'GET',
      url: `http://localhost:5000/api/getRestaurantsByCity/${cityId}`,
    })
  }
}

```

```

        headers: { 'Content-Type': 'application/json' }
    }).then(result => {
        this.setState({
            restaurants: result.data.restaurants
        });
    }).catch(error => {
        console.log(error)
    });
}

onTextChanged = (event) => {

    const searchInput = event.target.value;

    const { restaurants } = this.state;

    let suggestions = [];

    if (searchInput.length > 0) {
        suggestions = restaurants.filter(
            item =>
item.name.toLowerCase().includes(searchInput.toLowerCase())
        );
    }

    this.setState({
        suggestions,
        text: searchInput
    });
}

renderSuggestions = () => {
    const { suggestions } = this.state;
    if (suggestions.length == 0) {
        return null;
    }
    return (
        <ul className="suggestionsBox">
            {
                suggestions.map((item, index) => {
                    return (
                        <li key={index} onClick={() =>
this.selectRestaurant(item)} value={item}>{ ` ${item.name}, ${item.city}` }</li>
                    )
                })
            }
        </ul>
    )
}

```

```

selectRestaurant = (item) => {

    this.props.history.push(`/restaurant?id=${item._id}`)
}
handleChangeheader = (event, stateVariable) => {
    this.setState({
        [stateVariable]: event.target.value
    });
}

signUpHandler = (event) => {
    const {
        email,
        password,
        firstname,
        lastname
    } = this.state;

    const signUpRequestObj = {
        email: email,
        password: password,
        firstname: firstname,
        lastname: lastname
    }

    axios({
        method: 'POST',
        url: 'http://localhost:5000/api/signup',
        headers: {'Content-Type': 'application/json'},
        data: signUpRequestObj
    }).then(
        response => {
            if (response.data.message == 'User signed up successfully !') {
                this.setState({
                    isModalOpen: false,
                    email: '',
                    password: '',
                    firstname: '',
                    lastname: ''
                });
                alert(response.data.message);
            }
        }
    ).catch(
        error => {
            console.log(error);
            alert(error);
        }
    )
}

```



```

    }

    loginHandler = (event) => {
      const {
        email,
        password
      } = this.state;

      const loginRequestObj = {
        email: email,
        password: password
      }
      axios({
        method: 'POST',
        url: 'http://localhost:5000/api/login',
        headers: {'Content-Type': 'application/json'},
        data: loginRequestObj
      }).then(
        response => {
          if (response.data.user.length >= 1) {
            this.setState({
              isSigninModalOpen: false,
              email: '',
              password: '',
              isLoggedIn: response.data.isAuthenticated,
              firstname: response.data.user[0].firstname
            });
            sessionStorage.setItem('isLoggedIn',
response.data.isAuthenticated);
          }
        }
      ).catch(
        error => {
          console.log(error);
          alert(error);
        }
      )
    }

    cancelHandler = (event) => {
      this.setState({
        isModalOpen: false,
      });
    }

    signUpOpenHandler = (event) => {
      this.setState({
        isModalOpen: true,
      });
    }

```

```

}

signInOpenHandler = (event) => {
  this.setState({
    isSigninModalOpen: true,
  })
}

cancelLoginHandler = (event) => {
  this.setState({
    isSigninModalOpen: false,
  })
}

logoutHandler = (event) => {
  this.setState({
    firstname: '',
    isLoggedIn: false
  });
  sessionStorage.setItem('isLoggedIn', false);
}

render() {
  const { cities } = this.props;
  const { text } = this.state;
  const {
    isModalOpen,
    isSigninModalOpen,
    email,
    password,
    firstname,
    lastname,
    isLoggedIn
  } = this.state;
  return (
    <React.Fragment>
      <img src={homepageimg} alt="" style={{ width: '100%', height:
'450px', margin: 'auto' }} />
      <div>
        <div className="head">

          {
            isLoggedIn
            ?
            <div>
              <span>Welcome to Order Miss {firstname} !</span>
              <button className="btn btn-sm btn-primary ml-3"
onClick={this.logoutHandler}>Logout</button>
            </div>

```

```

        :
        <div>
            <button className="btn btn-sm text-white"
onClick={this.signInOpenHandler}>Login</button>
            <button className="btn default btn-sm text-white"
onClick={this.signUpOpenHandler}>Create an account</button>
        </div>
    }
</div>
<Modal
    isOpen={isModalOpen}
    style={customStyles}>
    <div>
        <h3>Signup User</h3>
        <div>
            <span>Email : </span>
            <input type="text" value={email} onChange={(event) =>
this.handleChangeheader(event, 'email')}></input>
        </div>
        <div>
            <span>Password : </span>
            <input type="password" value={password}
onChange={(event) => this.handleChangeheader(event, 'password')}></input>
        </div>
        <div>
            <span>First Name : </span>
            <input type="text" value={firstname} onChange={(event)
=> this.handleChangeheader(event, 'firstname')}></input>
        </div>
        <div>
            <span>Last Name : </span>
            <input type="text" value={lastname} onChange={(event)
=> this.handleChangeheader(event, 'lastname')}></input>
        </div>
        <button onClick={this.signUpHandler} class="btn btn-sm
btn-primary">Signup</button>
        <button onClick={this.cancelHandler} class="btn btn-sm
btn-primary">Cancel</button>
    </div>
</Modal>
<Modal
    isOpen={isSigninModalOpen}
    style={customStyles}>
    <div>
        <h3>Login User</h3>
        <div>
            <span>Email : </span>
            <input type="text" value={email} onChange={(event) =>
this.handleChangeheader(event, 'email')}></input>

```

```

        </div>
        <div>
            <span>Password : </span>
            <input type="password" value={password}
onChange={ (event) => this.handleChangeheader(event, 'password')} ></input>
        </div>
        <button onClick={this.loginHandler} class="btn btn-sm btn-
primary">Login</button>
        <button onClick={this.cancelLoginHandler} class="btn btn-
sm btn-primary">Cancel</button>
    </div>
</Modal>
</div>

    <div>
        <div className="logo">
            <img src={logo} alt="" className="im" />
        </div>
        <div className="headings">
            Find the best restaurants, cafes, bars
        </div>
        <div className="locationSelector">
            <select className="locationDropdown"
onChange={this.handleChange}>
                <option value="select" selected>Please type a
location</option>
                {
                    cities.map((city, index) => {
                        return <option key={index}
value={city.city_id}>{city.name}</option>
                    })
                }
            </select>
            <div className="suggestions-func">

                <input className="restaurantsinput" type="text"
value={text} placeholder=" Search for restaurants" onChange={this.onTextChanged}/>
                {
                    this.renderSuggestions()
                }
            </div>
        </div>
    </div>
</React.Fragment>
)
}
}

```

```
export default withRouter(Wallpaper);
```

C. Styles

a. Details.css

```
.tabs {  
  position: relative;  
  min-height: 400px;  
  clear: both;  
  margin: 50px 0 25px;  
  background: white;  
}  
  
.tab {  
  float: left;  
  height: 500px;  
}  
  
.tab label {  
  background: white;  
  padding: 10px;  
  border: 1px solid white;  
  margin-left: -1px;  
  position: relative;  
  left: 1px;  
  top: -29px;  
}  
  
.tab [type=radio] {  
  display: none;  
}  
  
.content {  
  position: absolute;  
  top: -1px;  
  left: 0;  
  background: white;  
  right: 0;  
  bottom: 0;  
  padding: 20px;  
  border-top: 1px solid gray;
```

```
    opacity: 0;
}

[type=radio]:checked~label {
    background: white;
    border-bottom: 1px solid white;
    z-index: 2;
    border-bottom: solid 4px red;
}

[type=radio]:checked~label~.content {
    z-index: 1;
    opacity: 1;
}

.button {
    position: absolute;
    left: 80%;
    top: 40%;
    width: 200px;
    height: 35px;
    opacity: 0.6;
    border-radius: 3px;
    background-color: #ffffff;
}

.heading {
    width: 498px;
    height: 43px;
    font-family: Poppins;
    font-size: 30px;
    font-weight: 600;
    font-stretch: normal;
    font-style: normal;
    line-height: 1.53;
    letter-spacing: normal;
    text-align: left;
    color: #192f60;
}

.about {
    width: 181px;
    height: 31px;
    font-family: Poppins;
    font-size: 22px;
    font-weight: 600;
    font-stretch: normal;
    font-style: normal;
    line-height: 1.5;
```

```
    letter-spacing: normal;
    text-align: left;
    color: #192f60;
}

.head {
    height: 25px;
    font-family: Poppins;
    font-size: 18px;
    font-weight: 600;
    font-stretch: normal;
    font-style: normal;
    line-height: 1.5;
    letter-spacing: normal;
    text-align: left;
    color: #192f60;
}

.value {
    height: 30px;
    font-family: Poppins;
    font-size: 16px;
    font-weight: normal;
    font-stretch: normal;
    font-style: normal;
    line-height: 1.88;
    letter-spacing: normal;
    text-align: left;
    color: #192f60;
}

@media only screen and (max-width: 600px) {
    .button {
        width: 242px;
        height: 48px;
        padding: 13px 13px 12px 16px;
        opacity: 0.8;
        border-radius: 3px;
        background-color: var(--white);
    }
}
```

b. Filter.css

```
.heading {
  height: 51px;
  font-family: Poppins;
  font-size: 36px;
  font-weight: bold;
  font-stretch: normal;
  font-style: normal;
  line-height: 1.53;
  letter-spacing: normal;
  text-align: left;
  color: #192f60;
  margin-bottom: 10px;
  margin-left: 15px;
}
.res{
  margin-top: 650px;
  width:1000px;
  height: 50px;
}
.rect{
  width: 794px;
  height: 352px;
  margin: -1100px 154px 330px 600px;
  padding: 155px 235px 154px;
  box-shadow: 0 3px 6px 0 rgba(0, 0, 0, 0.16);
  background-color: var(--white);
}
.no{
  width: 324px;
  height: 43px;
  font-family: Poppins;
  font-size: 30px;
  font-weight: 600;
  font-stretch: normal;
  font-style: normal;
  line-height: 1.53;
  letter-spacing: normal;
  text-align: left;
  color: var(--dark-slate-blue);
}
.filtercontainer {
  height: 100%;
  box-shadow: 0 3px 6px 0 rgba(0, 0, 0, 0.16);
```



```
background-color: #ffffff;
margin-left: 10px;
}

.filter-options {
  box-shadow: 0 3px 6px 0 rgba(0, 0, 0, 0.16);
  background-color: #ffffff;
  padding-left: 30px;
  margin-bottom: 10px;
  padding-bottom: 3%;
}
```

```
.filter-heading {
  height: 25px;
  font-family: Poppins;
  font-size: 18px;
  font-weight: 600;
  font-stretch: normal;
  font-style: normal;
  line-height: 1.5;
  letter-spacing: normal;
  text-align: left;
  color: #192f60;
  padding-top: 20px;
}
```

```
.Select-Location {
  height: 20px;
  font-family: Poppins;
  font-size: 14px;
  font-weight: normal;
  font-stretch: normal;
  font-style: normal;
  line-height: 1.5;
  letter-spacing: normal;
  text-align: left;
  color: #192f60;
  margin-top: 30px;
}
```

```
.Rectangle-2236 {
  width: 200px;
  height: 25px;
  border-radius: 5px;
  border: solid 1px #8c96ab;
  margin-top: 10px;
}
```

```
.Cuisine {
```

```
height: 20px;
font-family: Poppins;
font-size: 14px;
font-weight: normal;
font-stretch: normal;
font-style: normal;
line-height: 1.5;
letter-spacing: normal;
text-align: left;
color: #192f60;
margin-top: 12px;
}

.checkbox-items {
height: 20px;
font-family: Poppins;
font-size: 14px;
font-weight: normal;
font-stretch: normal;
font-style: normal;
line-height: 1.5;
letter-spacing: normal;
text-align: left;
color: #8c96ab;
}

.Item {
width: 100%;
height: 220px;
box-shadow: 0 3px 6px 0 rgba(0, 0, 0, 0.16);
background-color: #ffffff;
margin-bottom: 15px;
margin-left: 600px;
margin-top: -550px;
}

.Item1 {
width: 100%;
height: 220px;
box-shadow: 0 3px 6px 0 rgba(0, 0, 0, 0.16);
background-color: #ffffff;
margin-bottom: 15px;
margin-left: 600px;
margin-top: 100px;
}

.img {
width: 100px;
height: 100px;
border-radius: 35px;
margin-left: 25%;
```

```
}

.rest-name {
  height: 43px;
  font-family: Poppins;
  font-size: 30px;
  font-weight: 600;
  font-stretch: normal;
  font-style: normal;
  line-height: 1.53;
  letter-spacing: normal;
  text-align: left;
  color: #192f60;
}

.toggle-span {
  float: right;
  padding: 5px 0px 5px 0px;
  display: none;
}

.res-location {
  height: 23px;
  font-family: Poppins;
  font-size: 16px;
  font-weight: 500;
  font-stretch: normal;
  font-style: normal;
  line-height: 1.88;
  letter-spacing: normal;
  text-align: left;
  color: #192f60;
}

.rest-address {
  height: 23px;
  font-family: Poppins;
  font-size: 16px;
  font-weight: normal;
  font-stretch: normal;
  font-style: normal;
  line-height: 1.88;
  letter-spacing: normal;
  text-align: left;
  color: #636f88;
}

.Bakery-700 {
  height: 40px;
```

```
font-family: Poppins;
font-size: 16px;
font-weight: normal;
font-stretch: normal;
font-style: normal;
letter-spacing: normal;
text-align: left;
color: #192f60;
}

.padding-left {
padding-left: 60px;
}

.custom-button {
color: black;
font-weight: bold;
text-decoration: none;
box-shadow: 0 0 0 0 white;
}

.pagination {
display: inline-block;
margin-left: 500px;
margin-top: -3px;
margin-top: 100px;
}

.pagination a {
color: gray;
float: left;
padding: 8px 16px;
text-decoration: none;
border: solid 1px gray;
border-radius: 15px;
margin: 2px;
}

@media only screen and (max-width: 600px) {
.Item {
width: 100%;
min-width: 350px;
height: 290px;
box-shadow: 0 3px 6px 0 rgba(0, 0, 0, 0.16);
background-color: #ffffff;
margin-bottom: 15px;
}
.padding-left {
padding-left: 3px;
}
```

```
}  
.filter-options {  
  box-shadow: 0 3px 6px 0 rgba(0, 0, 0, 0.16);  
  background-color: #ffffff;  
  padding-left: 30px;  
  margin-bottom: 10px;  
  width: 455px;  
  margin-left: 15px;  
  min-height: 25px;  
}  
.rest-name {  
  height: 43px;  
  font-family: Poppins;  
  font-size: 30px;  
  font-weight: 600;  
  font-stretch: normal;  
  font-style: normal;  
  line-height: 1.53;  
  letter-spacing: normal;  
  text-align: left;  
  color: #192f60;  
  margin-left: 20px;  
}  
.res-location {  
  height: 23px;  
  font-family: Poppins;  
  font-size: 16px;  
  font-weight: 500;  
  font-stretch: normal;  
  font-style: normal;  
  line-height: 1.88;  
  letter-spacing: normal;  
  text-align: left;  
  color: #192f60;  
  margin-left: 20px;  
}  
.rest-address {  
  height: 23px;  
  font-family: Poppins;  
  font-size: 16px;  
  font-weight: normal;  
  font-stretch: normal;  
  font-style: normal;  
  line-height: 1.88;  
  letter-spacing: normal;  
  text-align: left;  
  color: #636f88;  
  margin-left: 20px;  
}
```

```
.toggle-span {
    float: right;
    padding: 5px 0px 5px 0px;
    display: initial;
}

.pagination {
    display: inline-block;
    margin-top: 25px;
    width: 380px;
    margin-left: 5%;
}

}

.suggestionsBox {
    margin: 0 auto;
    padding: 0;
    max-height: 390px;
    overflow-y: auto;
    border: 1px solid rgba(0, 0, 0, 0.1);
    padding: 5px 5px 0 5px;
    border-left: none;
    border-right: none;
    position: absolute;
    left: 260px;
    top: 36px;
    width: 320px;
}

.suggestionsBox > li {
    list-style: none;
    background-color: lightgray;
    background-image: linear-gradient(90deg, red 20px, lightgray 10px);
    padding: 10px 15px 10px 25px;
    border: 1px solid #CCC;
    box-shadow: inset 1px 1px 0 rgba(255, 255, 255, 0.5);
    margin-bottom: 5px;
    width: 100%;
    box-sizing: border-box;
    cursor: pointer;
    border-radius: 3px;
}

.suggestions-func {
    position: relative;
}
```

c. Header.css

```
.header {
  width: 100%;
  height: 50px;
  background-color: red;
  color: white;
}

.login-block {
  margin-top: -60px;
  margin-left: 1450px;
}

.head{
  color: white;
  text-align: left;
  font-size: 18px;
  font-weight: normal;
  position: relative;
  margin-top: -190px;
  margin-left: 1370px;
  font-size: 36px;
  top: -240px;
}

.ima{
  text-align: left;
  font-size: 60px;
  color: red;
  background: white;
  width: 40px;
  height: 35px;
  border-radius: 50%;
  margin-top: 7px;
  margin-left: 100px;
  margin-right: 80%;
}

.btn {
  border: 2px solid black;
  background-color: white;
  color: black;
  padding: 14px 28px;
  font-size: 16px;
  cursor: pointer;
}
```

```
.default {  
  border-color: white;  
  color:white;  
  outline-style: auto;  
  outline-color: white;  
  outline-width: -50px;  
}  
  
.default:hover {  
  background: #e7e7e7;  
}
```

d. Home.css

```
.logo {  
  text-align: center;  
  font-size: 80px;  
  color: red;  
  background: white;  
  width: 115px;  
  height: 123px;  
  border-radius: 50%;  
  top: 10%;  
  margin-left: 43%;  
  margin-right: 40%;  
}  
.clickable-pointer{  
  cursor: pointer;  
}  
.clickable-pointer:hover{  
  box-shadow: 0 6px 12px darkgrey;  
}  
.im{  
  text-align: center;  
  font-size: 80px;  
  color: red;  
  background: white;  
  width: 75px;  
  height: 65px;  
  border-radius: 50%;  
  margin-top: -450px;  
  margin-left: -30%;  
  margin-right: 40%;  
}  
.headings {  
  color: white;
```



```
text-align: center;
font-size: 18px;
font-weight: normal;
position: relative;
margin-left: -10%;
font-size: 36px;
top: -240px;
}
.head{
color: white;
text-align: left;
font-size: 18px;
font-weight: normal;
position: relative;
margin-top: -290px;
margin-left: 1970px;
font-size: 36px;
top: -240px;
}
.btn {
border: 2px solid black;
background-color: white;
color: black;
padding: 14px 28px;
font-size: 16px;
cursor: pointer;
}
.default {
border-color: white;
color:white;
outline-style: auto;
outline-color: white;
outline-width: -10px;
}

.default:hover {
background: #e7e7e7;
}
.locationSelector {
position: relative;
top: -238px;
margin-left: 27.5%;
margin-right: 20%;
text-align: center;
}

.locationDropdown {
height: 40px;
width: 245px;
```

```
    color: gray;
    border: none;
    border-radius: 0px;
    font-size: 15px;
    opacity: 0.7;
    margin-right: 21px;
    margin-bottom: 25px;
    float: left;
}

.restaurantsinput {
    height: 40px;
    width: 310px;
    color: gray;
    border: none;
    border-radius: 0px;
    font-size: 15px;
    opacity: 0.7;
    float: left;
}

.quicksearch {
    position: absolute;
    margin-left: 15px;
    margin-top: 35px;
}

.quicksearchHeading {
    text-align: left;
    color: #192f60;
    font-size: 30px;
    font-weight: bold;
    font-family: Georgia, "Times New Roman", Times, serif;
    padding-left: 35px;
}

.quicksearchSubHeading {
    text-align: left;
    color: gray;
    font-size: 18px;
    font-weight: normal;
    font-family: Georgia, "Times New Roman", Times, serif;
    padding-left: 35px;
}

.tileContainer {
    height: 150px;
    width: 350px;
    background-color: white;
```

```
margin-right: 30px;
margin-bottom: 30px;
margin-left: 20px;
box-shadow: 0 3px 6px gray;
float: left;
}

.tileComponent1 {
width: 45%;
float: left;
}

.tileComponent2 {
width: 50%;
float: left;
margin-top: 20px;
}

.componentHeading {
text-align: left;
color: #192f60;
font-size: 18px;
font-weight: bold;
margin-bottom: 20px;
}

.componentSubHeading {
text-align: left;
color: gray;
font-size: 18px;
}

.qs-block {
box-shadow: 0 3px 6px 0 rgba(0, 0, 0, 0.16);
background-color: #ffffff;
margin-bottom: 5px;
}

.img {
width: 100%;
height: 160px;
}

.marg-bot {
margin-bottom: 10px;
}

.heading {
width: 88px;
```

```
height: 25px;
font-family: Poppins;
font-size: 18px;
font-weight: 600;
font-stretch: normal;
font-style: normal;
line-height: 1.5;
letter-spacing: normal;
text-align: left;
color: #192f60;
}

@media only screen and (max-width: 600px) {
  .restaurantsinput {
    height: 40px;
    width: 245px;
    color: gray;
    border: none;
    border-radius: 0px;
    font-size: 15px;
    opacity: 0.7;
    float: left;
  }
  .logo {
    text-align: center;
    font-size: 80px;
    color: red;
    background-color: white;
    width: 115px;
    height: 123px;
    border-radius: 50%;
    position: absolute;
    top: 10%;
    margin-left: 38%;
    margin-right: 40%;
  }
  .locationSelector {
    position: relative;
    top: -238px;
    margin-left: 23.5%;
    margin-right: 20%;
    text-align: center;
  }
}
.suggestionsBox {
  margin: 0 auto;
  padding: 0;
  max-height: 390px;
  overflow-y: auto;
```

```

border: 1px solid rgba(0, 0, 0, 0.1);
padding: 5px 5px 0 5px;
border-left: none;
border-right: none;
position: absolute;
left: 260px;
top: 36px;
width: 320px;
}

.suggestionsBox > li {
list-style: none;
background-color: lightgray;
background-image: linear-gradient(90deg, red 20px, lightgray 10px);
padding: 10px 15px 10px 25px;
border: 1px solid #CCC;
box-shadow: inset 1px 1px 0 rgba(255, 255, 255, 0.5);
margin-bottom: 5px;
width: 100%;
box-sizing: border-box;
cursor: pointer;
border-radius: 3px;
}

.suggestions-func {
position: relative;
}

```

e. Restaurant.css

```

.tabs {
position: relative;
min-height: 400px;
clear: both;
margin: 50px 0 25px;
background: white;
}

.tab {
float: left;
height: 500px;
}

.total{
color: plum;
}

.tab label {

```

```
background: white;
padding: 10px;
border: 1px solid white;
margin-left: -1px;
position: relative;
left: 1px;
top: -29px;
}

.tab [type=radio] {
  display: none;
}

.content {
  position: absolute;
  top: -1px;
  left: 0;
  background: white;
  right: 0;
  bottom: 0;
  padding: 20px;
  border-top: 1px solid gray;
  opacity: 0;
}

[type=radio]:checked~label {
  background: white;
  border-bottom: 1px solid white;
  z-index: 2;
  border-bottom: solid 4px red;
}

[type=radio]:checked~label~.content {
  z-index: 1;
  opacity: 1;
}

.button {
  position: absolute;
  left: 80%;
  top: 40%;
  width: 200px;
  height: 35px;
  opacity: 0.6;
  border-radius: 3px;
  background-color: #ffffff;
}

.heading {
```

```
width: 498px;
height: 43px;
font-family: Poppins;
font-size: 30px;
font-weight: 600;
font-stretch: normal;
font-style: normal;
line-height: 1.53;
letter-spacing: normal;
text-align: left;
color: #192f60;
}
```

```
.about {
width: 181px;
height: 31px;
font-family: Poppins;
font-size: 22px;
font-weight: 600;
font-stretch: normal;
font-style: normal;
line-height: 1.5;
letter-spacing: normal;
text-align: left;
color: #192f60;
}
```

```
.head {
height: 25px;
font-family: Poppins;
font-size: 18px;
font-weight: 600;
font-stretch: normal;
font-style: normal;
line-height: 1.5;
letter-spacing: normal;
text-align: left;
color: #192f60;
}
```

```
.value {
height: 30px;
font-family: Poppins;
font-size: 16px;
font-weight: normal;
font-stretch: normal;
font-style: normal;
line-height: 1.88;
letter-spacing: normal;
```

```

text-align: left;
color: #192f60;
}

@media only screen and (max-width: 600px) {
  .button {
    width: 242px;
    height: 48px;
    padding: 13px 13px 12px 16px;
    opacity: 0.8;
    border-radius: 3px;
    background-color: var(--white);
  }
}

```

D. App.js

```

import './App.css';

function App() {
  return (
    <div className="App">

      </div>
    );
}

export default App;

```

E. App.css

```

.App {
  text-align: center;
}

.App-logo {
  height: 40vmin;
  pointer-events: none;
}

```



```
@media (prefers-reduced-motion: no-preference) {
  .App-logo {
    animation: App-logo-spin infinite 20s linear;
  }
}

.e {
  width: 32px;
  height: 47px;
  font-family: Poppins;
  font-size: 33px;
  font-weight: 600;
  font-stretch: normal;
  font-style: normal;
  line-height: 1.52;
  letter-spacing: normal;
  text-align: left;
  color: #eb2929;
}

.header{
  width: 1730px;
  height: 80px;
  margin: 0 0 37px;
  padding: 15px 101px 13px 102px;
  background-color: red;
}

.Ellipse{
  width: 52px;
  height: 52px;
  margin: 0 861px 0 0;
  padding: 1px 10px 4px;
  border-radius: 50px;
  background-color: var(--white);
}

.Login{
  width: 60px;
  height: 30px;
  margin: 0px 0px 1700px 0;
  text-align: right;
  color: var(--white);
  background-color: red;
}

.create{
  width: 181px;
  height: 46px;
  margin: 0 0 0 26px;
  padding: 12px 14px 11px 15px;
  border-radius: 3px;
  background-color: red;
}
```

```

    color: var(--white);
}
.App-header {
  background-color: #282c34;
  min-height: 100vh;
  display: flex;
  flex-direction: column;
  align-items: center;
  justify-content: center;
  font-size: calc(10px + 2vmin);
  color: white;
}

.App-link {
  color: #61dafb;
}

@keyframes App-logo-spin {
  from {
    transform: rotate(0deg);
  }
  to {
    transform: rotate(360deg);
  }
}

```

F. Index.js

```

import React from 'react';
import ReactDOM from 'react-dom';
import reportWebVitals from './reportWebVitals';
import Router from './Router';
import 'bootstrap/dist/css/bootstrap.css';

ReactDOM.render(
  <Router />,
  document.getElementById('root')
);

reportWebVitals();

```

G.Router.js

```
import {Route, BrowserRouter} from 'react-router-dom';
import Home from './components/Home';
import Restaurant from './components/Restaurant';
import Filter from './components/Filter';

import Transaction from './components/Transaction';

const Router = () => {
  return(
    <BrowserRouter>

    <Route exact path="/" component={Home} />
    <Route path="/home" component={Home} />
    <Route path="/filter" component={Filter} />
    <Route path="/restaurant" component={Restaurant} />
    <Route path="/transaction" component={Transaction} />

    </BrowserRouter>
  )
}

export default Router;
```

III. Database

a) Location

```
[
  {
    "_id": "1",
    "name": "ShalimarBhagh, Delhi",
    "city_id": "1",
    "location_id": "1",
    "country_name": "India"
  },
```

```
{
  "_id": "2",
  "name": "Janpat, Delhi",
  "city_id": "1",
  "location_id": "2",
  "country_name": "India"
},
{
  "_id": "3",
  "name": "MSP, Delhi",
  "city_id": "1",
  "location_id": "3",
  "country_name": "India"
},
{
  "_id": "4",
  "name": "MSP, Pune",
  "city_id": "2",
  "location_id": "4",
  "country_name": "India"
},
{
  "_id": "5",
  "name": "Anand Vihar, Delhi",
  "city_id": "1",
  "location_id": "5",
  "country_name": "India"
},
{
```

```
    "_id": "6",
    "name": "Mahadevapura, Bangalore",
    "city_id": "3",
    "location_id": "5",
    "country_name": "India"
  },
  {
    "_id": "7",
    "name": "Anna Nagar, Chennai",
    "city_id": "4",
    "location_id": "5",
    "country_name": "India"
  },
  {
    "_id": "8",
    "name": "Thane, Mumbai",
    "city_id": "5",
    "location_id": "5",
    "country_name": "India"
  }
]
```

b) Mealtype

```
[
  {
    "_id": "1",
    "name": "Breakfast",
    "content": "Start your day with exclusive breakfast options",
    "image": "assets/breakfast.png"
```

```
    },  
    {  
      "_id": "2",  
      "name": "Lunch",  
      "content": "Relax in your work time with exclusive lunch options",  
      "image": "assets/lunch.png"  
    },  
    {  
      "_id": "3",  
      "name": "Dinner",  
      "content": "End your day with exclusive dinner options",  
      "image": "assets/dinner.png"  
    },  
    {  
      "_id": "4",  
      "name": "Snacks",  
      "content": "Party with numerous snacks",  
      "image": "assets/snacks.png"  
    },  
    {  
      "_id": "5",  
      "name": "Drinks",  
      "content": "Have fun with drinks",  
      "image": "assets/drinks.png"  
    },  
    {  
      "_id": "6",  
      "name": "NightLife",  
      "content": "Enjoy the Nightlife",
```

```
        "image": "assets/nightlife.png"
    }
]
```

c) RestaurantData

```
[
{
    "_id" : "1",
    "name" : "Gulab",
    "city_name" : "Delhi",
    "city":'1',
    "area":'11',
    "locality" : "Pitampura, New Delhi",
    "thumb":
    "https://b.zmtcdn.com/data/pictures/chains/3/6303/640252389ddc3f264
    dd0e9f2741e73cd.jpg",
    "cost": 450,
    "address":"Shop 1, Plot D, Pitampura, New Delhi Complex,
    Chincholi, Delhi-110006, Delhi",
    "type":[
        {
            "mealtype":'1',
            "name":"breakfast"
        },
        {
            "mealtype":'3',
            "name": "dinner"
        }
    ] ,
    "Cuisine":[
```

```
        {
            "cuisine":'1',
            "name": "North Indian"
        },
        {
            "cuisine":'4',
            "name": "Fast Food"
        }
    ]
},
{
    "_id" : "2",
    "name" : "Pandit Ji Paratha Hut",
    "locality" : "Ashok Vihar Phase 2",
    "city_name" : "Delhi",
    "city":'1',
    "area":'12',
    "address": "Shop 44, Plot C, Ashok Vihar Phase 2, Chincholi, Delhi-110006, Delhi",
    "thumb" :
    "https://b.zmtcdn.com/data/pictures/chains/3/6303/640252389ddc3f264dd0e9f2741e73cd.jpg",
    "cost":230,
    "contact_number": "45352465",
    "type": [
        {
            "mealtype": '1',
            "name": "breakfast"
```



```
    },
    {
      "mealtype":'3',
      "name":"dinner"
    }
  ],
  "Cuisine":[
    {
      "cuisine":'1',
      "name":"North Indain"
    },
    {
      "cuisine":'3',
      "name":"Chinese"
    }
  ]
},
{
  "_id" : "3",
  "name" : "Food Adda",
  "locality" : "Borivali West",
  "city_name" : "Mumbai",
  "city":'2',
  "area":'21',
  "address":"Borivali West, Mumbai-210006, Mumbai",
  "thumb" :
  "https://b.zmtcdn.com/data/pictures/7/18690357/0df7f4ca0c645a68a1657b1e69b015fa.jpg",
```

```
"cost": 530,
"contact_number":"467564",
"type":[
  {
    "mealtype":'2',
    "name":"lunch"
  },
  {
    "mealtype":'3',
    "name":"dinner"
  }
],
"Cuisine":[
  {
    "cuisine":'3',
    "name":"Chinese"
  },
  {
    "cuisine":'4',
    "name":"FastFood"
  }
]
},
{
  "_id" : "4",
  "name" : "Apna Punjab",
  "locality" : "Magarpatta",
  "city_name" : "Pune",
```

```
"city":'3',
"area":'31',
"address": "Borivali West, Mumbai-210006, Mumbai",
"thumb" :
"https://b.zmtcdn.com/data/res_imagery/6508401_RESTAURANT_21a
925c42f2f93c9709e1945b9eae56f.jpg",
"cost":670,
"contact_number": "6508401",
"type":[
    {
        "mealtype":'4',
        "name": "snacks"
    },
    {
        "mealtype":'5',
        "name": "drinks"
    }
],
"Cuisine":[
    {
        "cuisine":'1',
        "name": "North Indain"
    },
    {
        "cuisine":'2',
        "name": "South Indian"
    }
]
]
```

```
    },
    {
      "_id" : "5",
      "name" : "Empire Restaurant",
      "locality" : "Rajajinagar",
      "city_name" : "Bangalore",
      "city": '4',
      "area":'41',
      "address":"Rajajinagar, Bangalore-430006, Bangalore",
      "thumb":
      "https://b.zmtcdn.com/data/pictures/1/50471/bcf68da39dcfb0fe5bcfb74
      2c337385e.jpg",
      "cost":230,
      "contact_number":"8731537",
      "type":[
        {
          "mealtype":'1',
          "name":"breakfast"
        },
        {
          "mealtype":'5',
          "name":"drinks"
        }
      ],
      "Cuisine":[
        {
          "cuisine":'2',
          "name":"South Indian"
        },
      ],
```

```
        {
            "cuisine":'4',
            "name":"FastFood"
        }

    ]
},
```

```
{
    "_id" : "6",
    "name" : "Captain Sams",
    "locality" : "Sector70, Chandigarh",
    "city_name" : "Chandigarh",
    "city": '5',
    "area":'51',
    "address":"Sector70, Chandigarh-515436",
    "thumb" :
    "https://b.zmtcdn.com/data/reviews_photos/c7a/634a2c0def8a8d044992
    aea9e7680c7a_1556257724.jpg",
    "cost":630,
    "contact_number":"123334",
    "type":[
        {
            "mealtype":'5',
            "name":"drinks"
        },
        {
            "mealtype":'6',
            "name":"nightlife"
        }
    ]
}
```

```
    }
  ],
  "Cuisine":[
    {
      "cuisine":'1',
      "name":"North Indain"
    },
    {
      "cuisine":'4',
      "name":"FastFood"
    }
  ]
},
{
  "_id" : "7",
  "name" : "AMA Cafe",
  "city_name" : "Delhi",
  "city":'1',
  "area":'11',
  "locality" : "Majnu ka Tila, New Delhi",
  "thumb":
  "https://b.zmtcdn.com/data/res_imagery/307374_RESTAURANT_6688d81a57b8da4bcf20d725de39a3d2.jpg",
  "cost": 450,
  "address":"House 6, New Colony, Majnu ka Tilla, New Delhi",
  "type":[
    {
      "mealtype":'2',
```

```
        "name": "lunch"
      },
      {
        "mealtype": '4',
        "name": "lunch"
      }
    ] ,
    "Cuisine": [
      {
        "cuisine": '2',
        "name": "South Indian"
      },
      {
        "cuisine": '3',
        "name": "Chinese"
      }
    ]
  },
  {
    "_id" : "8",
    "name" : "Punjabi Angithi",
    "city_name" : "Delhi",
    "city": '1',
    "area": '11',
    "locality" : "Paschim Vihar, New Delhi",
```

```
"thumb":
  "https://b.zmtcdn.com/data/pictures/3/307113/54e0e60a17086184f1e5a
  44d7f580b54.png",
  "cost": 350,
  "address":"32-22, A 4, DDA Market, Paschim Vihar, New Delhi",
  "type":[
    {
      "mealtype":'2',
      "name":"lunch"
    },
    {
      "mealtype":'4',
      "name": "lunch"
    }
  ] ,
  "Cuisine":[
    {
      "cuisine":'2',
      "name": "South Indian"
    },
    {
      "cuisine":'3',
      "name":"Chinese"
    }
  ]
},
{
```



```
"_id" : "9",
"name" : "Rajinder Da Dhaba",
"city_name" : "Delhi",
"city":'1',
"area":'11',
"locality" : "Safdarjung, New Delhi",
"thumb":
"https://b.zmtcdn.com/data/pictures/9/7319/e1b7673ed0aa2993b55b177
409d5596c.jpg",
"cost": 380,
"address":"AB 14, Safdarjung Enclave Market, Safdarjung, New
Delhi",
"type":[
  {
    "mealtype":'5',
    "name":"drinks"
  },
  {
    "mealtype":'6',
    "name": "nightlife"
  }
] ,
"Cuisine":[
  {
    "cuisine":'4',
    "name": "Fast Food"
  },
  {
```

```
        "cuisine": '5',
        "name": "Street Food"
    }

    ],
},
{
    "_id" : "10",
    "name" : "Diggin",
    "city_name" : "Delhi",
    "city": '1',
    "area": '11',
    "locality" : "Chanakyapuri, New Delhi",
    "thumb":
    "https://b.zmtcdn.com/data/pictures/3/307113/54e0e60a17086184f1e5a44d7f580b54.png",
    "cost": 650,
    "address": "10, Santushti Shopping Complex, Race Course Road, Chanakyapuri, New Delhi",
    "type": [
        {
            "mealtype": '5',
            "name": "drinks"
        },
        {
            "mealtype": '6',
            "name": "nightlife"
        }
    ]
}
```

```
    ] ,  
    "Cuisine":[  
        {  
            "cuisine":'4',  
            "name": "Fast Food"  
        },  
        {  
            "cuisine":'5',  
            "name": "Street Food"  
        }  
    ]  
}
```

```
]
```

d) Menu

```
[  
    {  
        item:"Kaju Katli",  
        cost:"450",  
        desription:"Special Kaju Sweet",  
        restaurantId:"5"  
    },  
    {  
        item:"Special Pav Bhaji",  
        cost:"30",  
        desription:"Awesome Taste",  
        restaurantId:"3"  
    },  
]
```

```
{
  item:"Samosa",
  cost:"20",
  desription:"very tasty",
  restaurantId:"8"
},
{item:"Shahi Paneer",
  cost:"150",
  desription:"delicious-dish",
  restaurantId:"4"
},
{item:"Rasgulle",
  cost:"60",
  desription:"Very Tasty",
  restaurantId:"8"
},
{
  item:"Bhelpuri",
  cost:"40",
  desription:"delicious",
  restaurantId:"7"
},
{
  item:"Matar paneer",
  cost:"100",
  desription:"delicious",
  restaurantId:"2"
},
{
```

```
item:"Honey Chilli Potato",
cost:"30",
desription:"Tasty Crispy Potatos",
restaurantId:"1"
},
{
item:"Pizza",
cost:"300",
desription:"delicious-bread",
restaurantId:"1"
},
{
item:"Special Bedai",
cost:"20",
desription:"Tasty Breakfast",
restaurantId:"7"
},
{
item:"Paneer Dosa",
cost:"50",
desription:"Famous And Tasty South Indian Dish",
restaurantId:"4"
},
{
item:"Chowmein",
cost:"150",
desription:"Chinese Dish",
restaurantId:"5"
```

```
},  
{  
  item:"Rajbhog",  
  cost:"20",  
  desription:"Very Sweet",  
  restaurantId:"2"  
}  
]
```

e) Orders

//will save the orders of users

f) Users

```
[  
  {  
    email:"abc@gmail.com",  
    password:"abc",  
    firstname:"Shrey",  
    lastname:"Project"  
  }  
]
```

//will save user information in the above format

CONCLUSION

This project is only a humble venture to satisfy the needs to manage their project work. Several user friendly coding have also adopted. This package shall prove to be a powerful package in satisfying all the requirements of the college. The objective of software planning is to provide a frame work that enables the manger to make reasonable estimates made within a limited time frame at the beginning of the software project and should be updated regularly as the project progresses.

At the end it is concluded that we have made effort on following points...

- A description of the background and context of the project and its relation to work already done in the area.
- Made statement of the aims and objectives of the project.
- The description of Purpose, Scope, and applicability
- We define the problem on which we are working in the project.
- We describe the requirement Specifications of the system and the actions that can be done on these things.
- We understand the problem domain and produce a model of the system, which describes operations that can be performed on the system.
- We included features and operations in detail, including screen layouts.
- We designed user interface and security issues related to system.
- Finally the system is implemented and tested according to test cases.

BIBLIOGRAPHY

[Online Food Ordering System]

- Search Engine Google.com
- MongoDB.com
- Joe Morgan “React.JS”
- Vasan Subramanian “MERN Stack”
- Roger s Pressman “Software Engineering”
- Wikipedia
- <http://getbootstrap.com/>
- <https://www.w3schools.com/>