Technical Interim Assessment

RESTAURANT MANAGEMENT SYSTEM

NAME : JEEVA SEKAR

ID : 12107

DOCUMENT PATTERN

- WH3 ANALYSIS
- ALGORITH & FLOWCHART
- SEQUENCE DIAGRAM
- USECASE DUAGRAM
- CLASS DIAGRAM
- DATABASE
- SPRING BOOT SOURCE CODE
- REACT JS SOURCE CODE
- CRUD OPERATIONS

W3H

WHAT?	HOW?

1. What are the tables needed for Restaurant Mangement System?
Ans:

1. Restaurant Table
2. Customer Table

2. What are the ways to sell the food?

1. Online
2. Offline

3. What are the modules I need to add?
Ans:

1. **Admin Module**

3.1 What Admin Can Do?

1. Approve the Reservation
2. Cancel the Customer Reservation
3. Approve The Order
4. Cancel The Order
5. View customer Details

2. **User Module**

3.2 What User Can Do?

1. Book the Reservation
2. Cancel the Reservation
3. Order The Food
4. Cancel The Food

1. Registration

Method 1:

1.Using Only Username & password

Method 2:

Using Only Email & password

Method 3:

Using Only Phone Number & password

Method 4:

Using all the details Name, Email, password and Phone Number

2.Login

Method 1:

Using only Username and password

Method 2:

Using Phone Number and password

Method 3:

Using Either Username & password or Phone Number & password

3. Admin Module
Change Reservation Times

Method 1:

The Reserved Day

Method 2:

Before The Reserved Day

4. What are the credentials needed for registration & Login?

1. Email
2. Username
3. Phone Number/Mobile Number
4. Password

5.What are the ways to provide Food

1. Online
2. Offline

6. What payment methods are supported?

1. UPI Transaction
2. Net Banking
3. Cash (Pay in Reception)

3. User Module

I. **Booking the Reservation or Food**

Method 1:

Booking the Reservation/Food using Phone.

Method 2:

Booking the Reservation/Food using Laptop/Computer

Method 3:

Applicable to book the ticket using both phone and laptop/computer.

ii. **Cancel Reservation/Food**

Method 1:

In the application directly

Method 2:

By reaching out to Restaurant

Method 3:

Either directly from the App or Reaching out to the Restaurant

WHY?	WHY NOT?
1. Registration **Method 4:** Using all the details Name, Email, password and Phone Number. To avoid complexity while registration. It is easier to retrieve data from a user. **2. Login** **Method 3:** Using Either Username & password or Phone Number & password. Giving users the flexibility to login to the application using either username or phone Number. **3.Admin** **Change Reservation Times** **Method 2:** Before The Reserved Day If The Admin cancel the reservation before one day It will be easy to search another restaurant for the customer.	**Registration** **Method 1:** Using Only Username & password **Method 2:** Using Only Email & password **Method 3:** Using Only Phone Number & password It might become difficult to get or view the details about the user. It might get difficult when retrieving the data of the customer or the user from the database. **Login** **Method 1:** Using only Username and password **Method 2:** Using Phone Number and password Few people might forget their Username, but they will remember their Phone Number.

4.User **I. Booking the Reservation or Food** **Method 3:** Applicable to book the Reservation /Food using both phone and laptop/computer. To allow Cross platform connection. It will be more helpful for the people who don't have laptops/computers **II. Cancel Reservation/Food** **Method 3:** Either directly from the App or Reaching out to the Restaurant. Allowing both the features for the people who do not know how to cancel directly from the app. They can directly come to the Restaurant to Cancel the Reservation/Food	**3. Admin Module** **Change Reservation Times** **Method 1:** The Reserved Day If The Admin cancel the Reservation on the reserved day the customer plans may affect **3. User Module** I. **Booking the Reservation or Food** **Method 1:** Booking the Reservation/Food using Phone. **Method 2:** Booking the Reservation/Food using Laptop/Computer Few might not have smartphones so they will be able to book Reservation/Food from computer centers **II. Cancel Reservation/Food** **Method 1:** In the application directly **Method 2:** By reaching out to Restaurant In this one by reaching out to restaurant to cancel the reservation may prevent cost of amount

ALOGORITHM & FLOWCHART

USER Registration & Login

Step 1: Start

Step 2: Open The Browser

Step 3: Enter URL For the Restaurant Application or open the app in the phone.

Step 4: Enter the Email Id, Phone Number, Username, and password.

Step 5: Click Register

Step 6: Verify yourself as valid user by clicking the verification mail to your mail id used for registration.

Step 7: Login With the Credentials.

Step 8: If The Login is Success Go to Step 9 Else Re-Enter the Credentials.

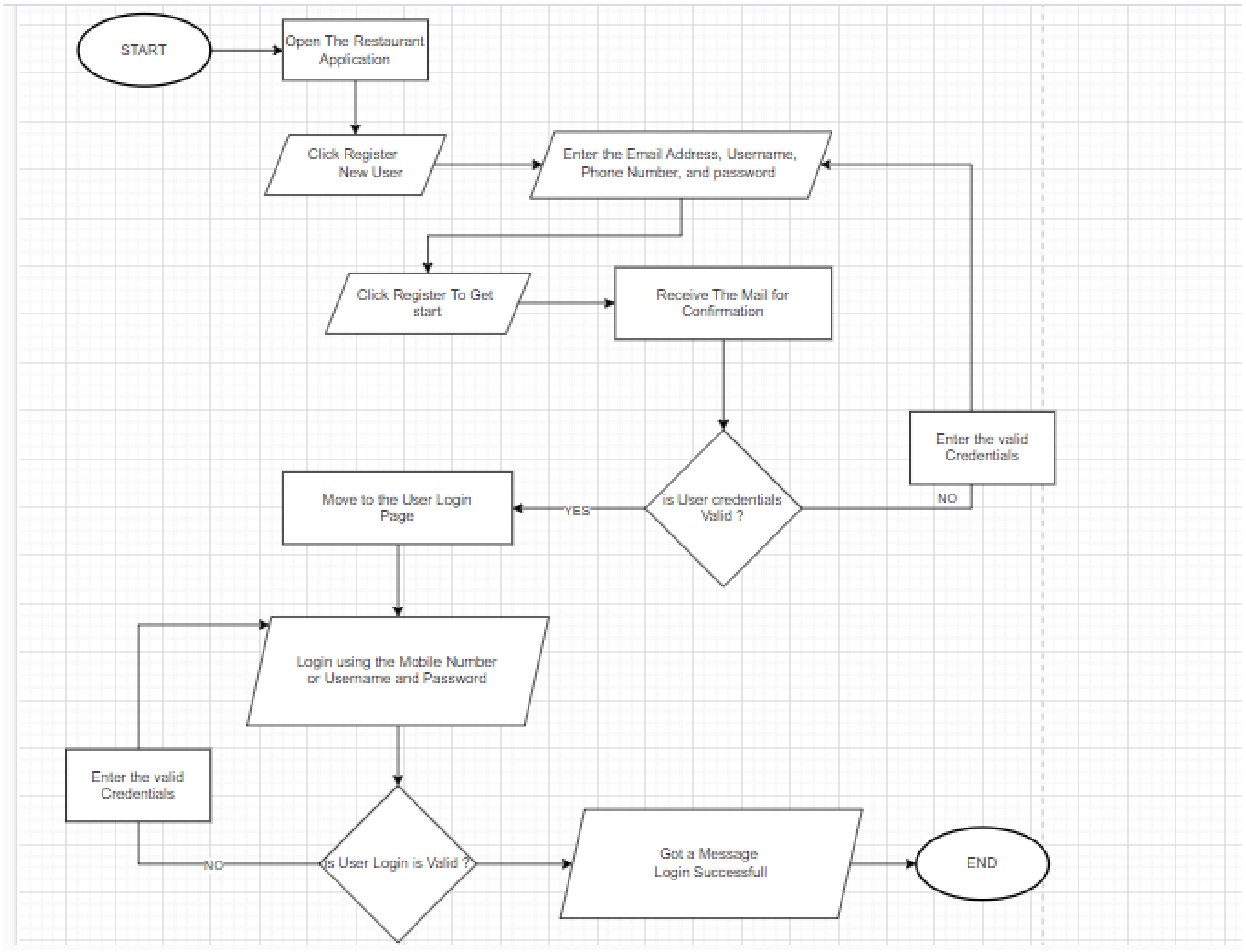
Step 9: Welcome To the Home Page

Step 10: View The Details About the Restaurant.

Step 11: Logout

Step 12: End

USER Registration & Login FLOWCHART


START
Open The Restaurant Application
Click Register New User
Enter the Email Address, Username, Phone Number, and password
Click Register To Get start
Receive The Mail for Confirmation
Enter the valid Credentials
NO
is User credentials Valid ?
YES
Move to the User Login Page
Login using the Mobile Number or Username and Password
Enter the valid Credentials
NO
is User Login is Valid ?
Got a Message Login Successfull
END

Algorithm and Flowchart for Ordering food in Restaurant

Step 1: Start

Step 2: Enter URL For the Restaurant Application to order food in the browser or open the app in the phone.

step 3: Login using the credentials. If credentials are not valid return to step 2.

Step 4: Welcome To the Home Page

Step 5: Choose The Food to Order

Step 6 : Check The Food is Available or Not

Step 7: If the Food is available go to step 8. Else go to step 6 to search for other food.

Step 8: Select The Count of the food.

Step 9: Click continue to payment.

Step 10: Choose the payment method

Step 11: Proceed To Pay

Step 12: The payment is successfully completed.

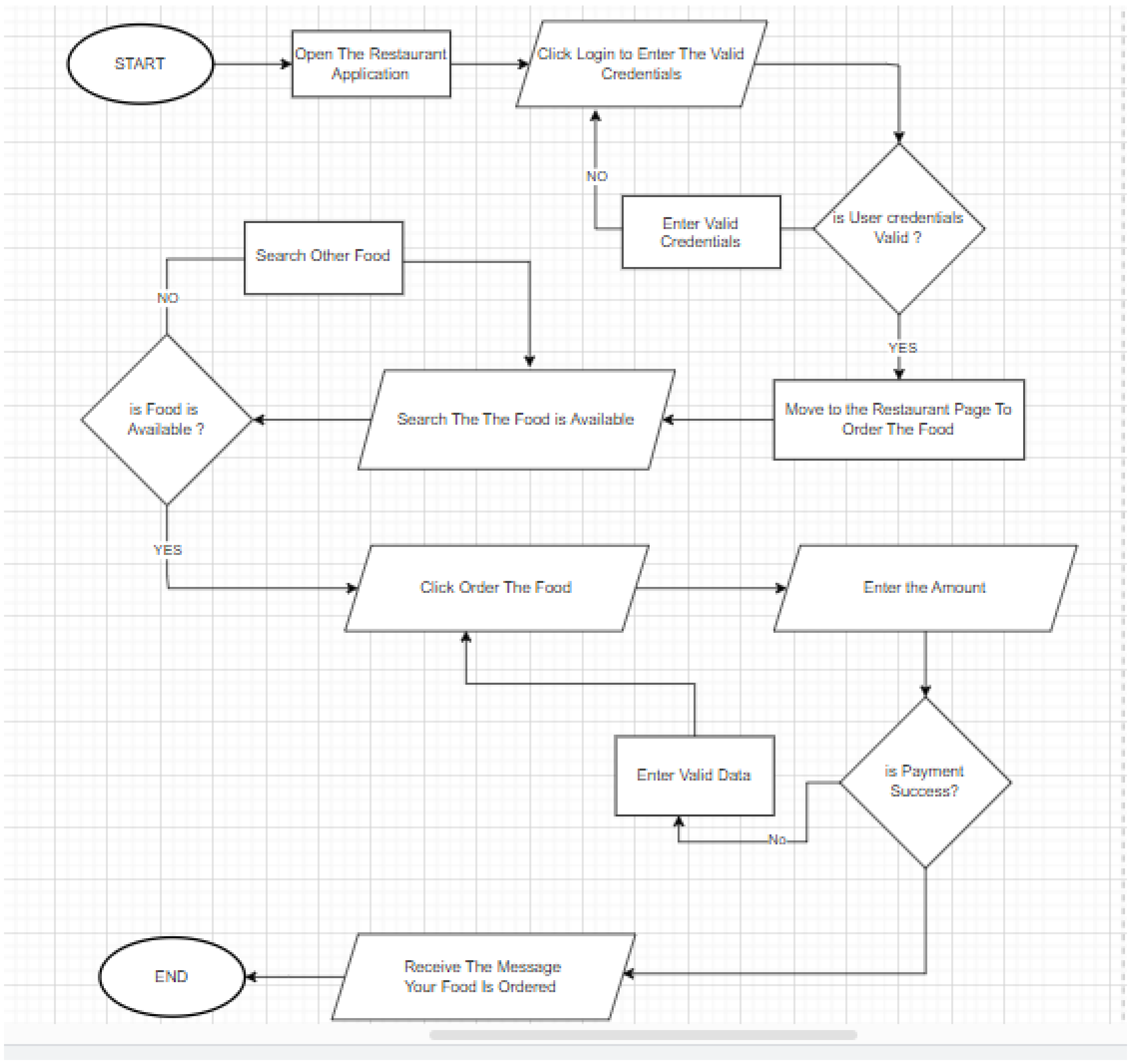
If the payment fails, go back to step 12.

Step 14: Got Message Successfully Ordered The Food.

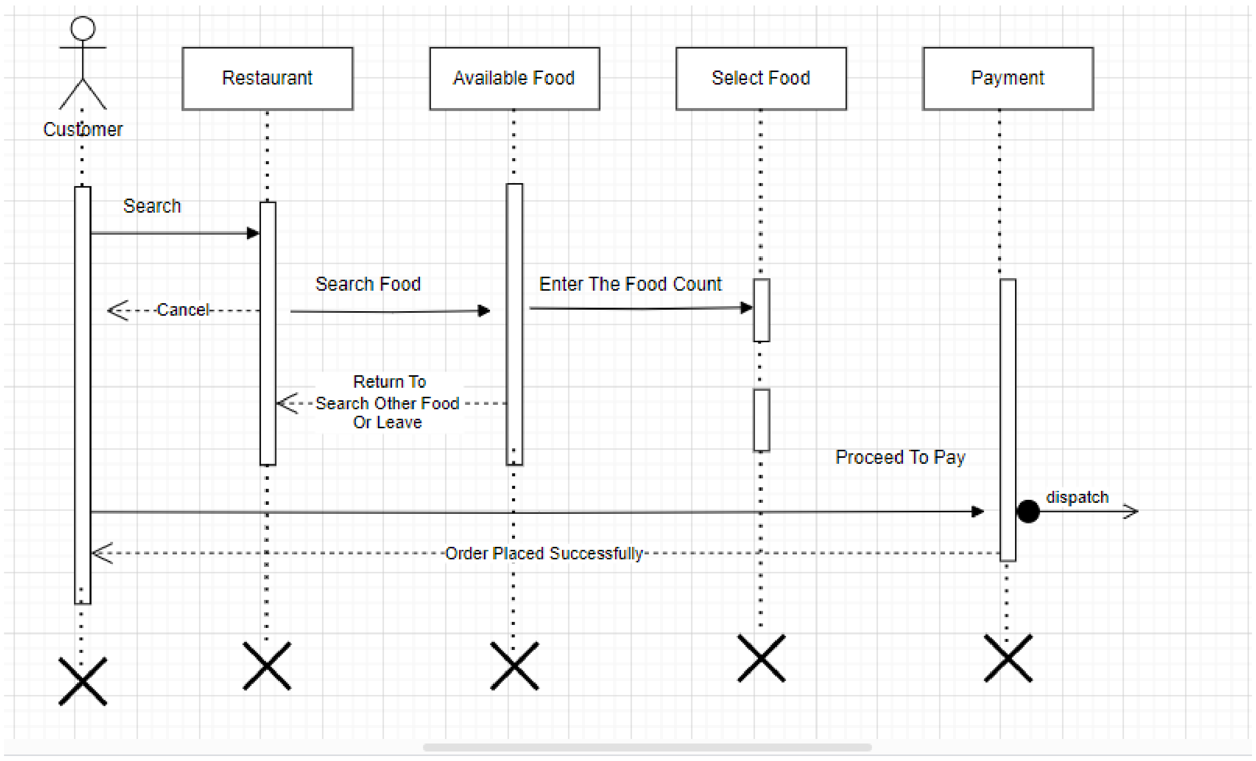
Step 15: Return To The Home Page

Step 16 : End.

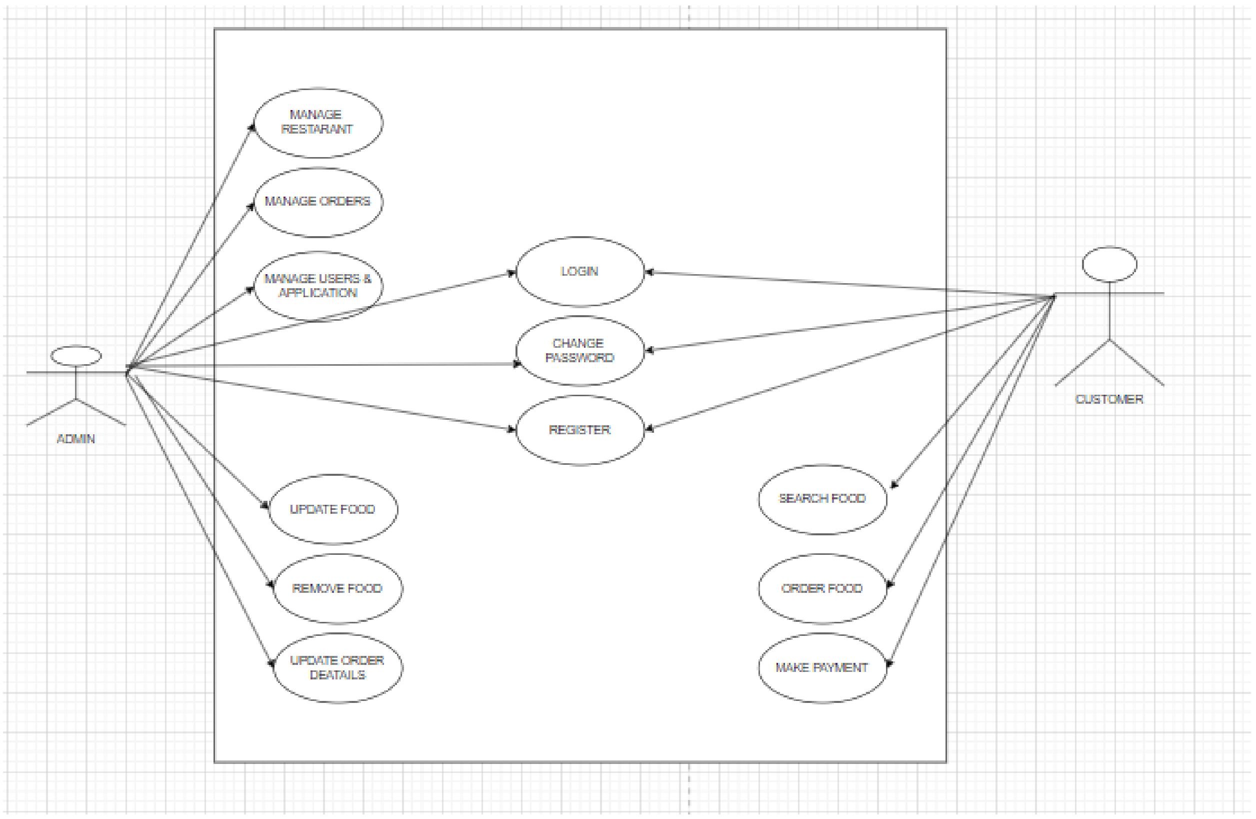
Food Order FLOWCHART


START
Open The Restaurant Application
Click Login to Enter The Valid Credentials
is User credentials Valid ?
NO
Enter Valid Credentials
YES
Move to the Restaurant Page To Order The Food
Search The The Food is Available
is Food is Available ?
NO
Search Other Food
YES
Click Order The Food
Enter the Amount
is Payment Success?
No
Enter Valid Data
Receive The Message Your Food Is Ordered
END

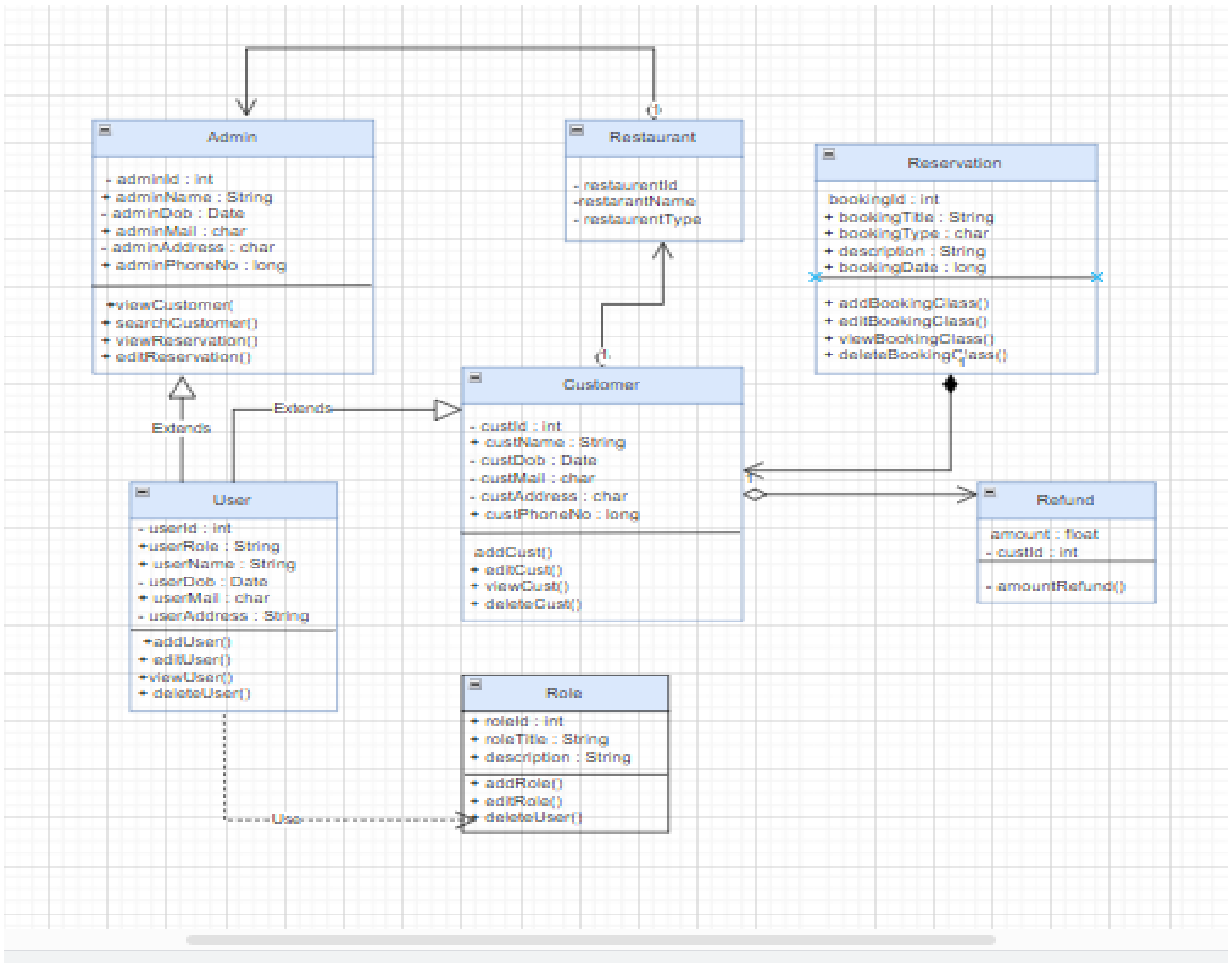
SEQUENCE DIAGRAM FOR ORDERING THE FOOD



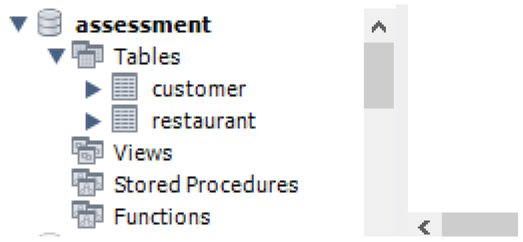
USECASE DIAGRAM



CLASS DIAGRAM



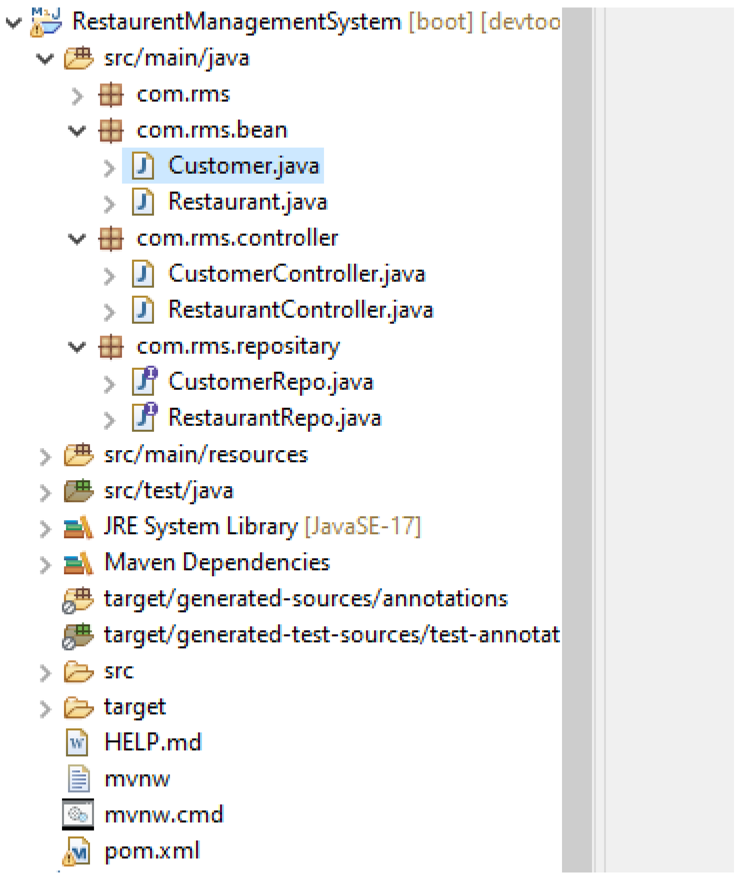
DATABASE



```
assessment
  Tables
    customer
    restaurant
  Views
  Stored Procedures
  Functions
```

SPRING BOOT SOURCE CODE

PACKAGES



```
RestaurentManagementSystem [boot] [devtoo
  src/main/java
    com.rms
    com.rms.bean
      Customer.java
      Restaurant.java
    com.rms.controller
      CustomerController.java
      RestaurantController.java
    com.rms.repositary
      CustomerRepo.java
      RestaurantRepo.java
  src/main/resources
  src/test/java
  JRE System Library [JavaSE-17]
  Maven Dependencies
  target/generated-sources/annotations
  target/generated-test-sources/test-annotat
  src
  target
  HELP.md
  mvnw
  mvnw.cmd
  pom.xml
```

Com.rms.bean

Customer Class

```
package com.rms.bean;




import jakarta.persistence.Entity;

import jakarta.persistence.Id;

import jakarta.persistence.JoinColumn;

import jakarta.persistence.ManyToOne;

import jakarta.persistence.Table;


@Entity

@Table

public class Customer {


@Id

private int cid;

private String cname;

private String caddress;

private int cage;

private String cdob;

private String cphone;
```

```
@ManyToOne

@JoinColumn(name ="rid")

private Restaurant res;




public Customer() {

super();

}


public Customer(int cid, String cname, String caddress, int cage, String cdob, String
cphone) {

super();

this.cid = cid;

this.cname = cname;

this.caddress = caddress;

this.cage = cage;

this.cdob = cdob;

this.cphone = cphone;

}


public int getCid() {

return cid;

}


public void setCid(int cid) {
```

```
this.cid = cid;

}


public String getCname() {

return cname;

}


public void setCname(String cname) {

this.cname = cname;

}


public String getCaddress() {

return caddress;

}


public void setCaddress(String caddress) {

this.caddress = caddress;

}


public int getCage() {

return cage;

}


public void setCage(int cage) {

this.cage = cage;

}
```

```
public String getCdob() {

return cdob;

}


public void setCdob(String cdob) {

this.cdob = cdob;

}


public String getCphone() {

return cphone;

}


public void setCphone(String cphone) {

this.cphone = cphone;

}


public Restaurant getRes() {

return res;

}


public void setRes(Restaurant res) {

this.res = res;

}
```

```
}
```

Com.rms.bean

Restaurant class

```
 package com.rms.bean;

import java.util.List;



import jakarta.persistence.CascadeType;
import jakarta.persistence.Entity;
import jakarta.persistence.Id;
import jakarta.persistence.OneToMany;
import jakarta.persistence.Table;

@Entity
@Table
```

```
public class Restaurant {

@Id
private int rid;
private String rname;
private String rcity;

@OneToMany(mappedBy = "res", cascade = CascadeType.ALL )
private List<Customer> clist;

public int getRid() {
return rid;
}

public void setRid(int rid) {
this.rid = rid;
}

public String getRname() {
return rname;
}
public void setRname(String rname) {
this.rname = rname;
}

public String getRcity() {
return rcity;
}

public void setRcity(String rcity) {
this.rcity = rcity;
}




}
```

Com.rms.controller

Customer Controller Class

```
package com.rms.controller;

import java.util.List;
```

```
import
org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.CrossOrigin;
import
org.springframework.web.bind.annotation.DeleteMapping;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.PutMapping;
import org.springframework.web.bind.annotation.RequestBody;
import
org.springframework.web.bind.annotation.RestController;


import com.rms.bean.Customer;
import com.rms.repositary.CustomerRepo;

@RestController
@CrossOrigin("http://localhost:3000")
public class CustomerController {

@Autowired
```

```
private CustomerRepo crepo;


@PostMapping("/insertCustomer")
public String insertCustomer(@RequestBody Customer cus) {
String Msg="";
try {
crepo.save(cus);
Msg= "Customer Record Inserted";
}catch(Exception e) {
System.out.println(e);
Msg= "Failed To Insert";
}
return Msg;
}



 @PutMapping("/updateCustomerDetails")
 public String updateEmployee(@RequestBody Customer eidd) {
 String Msg="";
try {
crepo.save(eidd);
```

```
Msg= "Customer Record Updated";
}catch(Exception e) {
System.out.println(e);
Msg= "Failed To Update";
}
return Msg;
}



@DeleteMapping("/deleteCusById/{cid}")
public String deleteCusById(@PathVariable int cid) {

try {

crepo.deleteById(cid);
return "Deletion Success";

}catch(Exception e) {
System.out.println(e);
return "Failed To Delete";
```

```
}
}

 @GetMapping("/getAllCustomers")
 public List<Customer> getAllCustomers(){
List<Customer> getAllCustomer=crepo.findAll();
return getAllCustomer;

 }

 @GetMapping("/getAllCusId")
 public List<Integer> getAllCusId(){
List<Integer> idList=crepo.findAllId();
return idList;

 }

 @GetMapping("/findCusById/{cid}")
 public Customer findCusById(@PathVariable int cid){
return crepo.findById(cid).get();
```

```
 }


}
```

RestaurantController Class

```
package com.rms.controller;

import java.util.List;

import
org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.CrossOrigin;
import
org.springframework.web.bind.annotation.DeleteMapping;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.PutMapping;
import org.springframework.web.bind.annotation.RequestBody;
import
org.springframework.web.bind.annotation.RestController;
```

```java
import com.rms.bean.Restaurant;

import com.rms.repositary.RestaurantRepo;

@RestController

@CrossOrigin("http://localhost:3000")

public class RestaurantController {



@Autowired

private RestaurantRepo resrepo;



@PostMapping("/insetRestaurant")

public String insetRestaurant(@RequestBody Restaurant res) {

try {

resrepo.save(res);

return "Inserted Successfully";

}catch(Exception e) {

return "Failed To  Insert";
}


}



@PutMapping("/updateResDetails")
 public String updateResDetails(@RequestBody Restaurant rid) {
 String Msg="";
try {
resrepo.save(rid);
Msg= "Restaurant Record Updated";
}catch(Exception e) {
System.out.println(e);
Msg= "Failed To Update";
}
return Msg;
}




@GetMapping("/getAllRestaurants")
```

```
public List<Restaurant> getAll() {

return resrepo.findAll();
}

@GetMapping("/findResById/{rid}")
public Restaurant findResById(@PathVariable int rid) {

return resrepo.findById(rid).get();
}


@DeleteMapping("/deleteResId/{rid}")
public String deleteResId(@PathVariable int rid) {

try {
resrepo.deleteById(rid);
return "Deletion Success";
}catch(Exception e) {
return "Deletion Failed";
}
}



@GetMapping("/getAllResId")
public List<Integer> getAllResId() {
List<Integer> listAllId=(List<Integer>) resrepo.getAllDepId();
return listAllId;


}




}
```

Com.rms.repositary

Customer Repositary

```
package com.rms.repositary;

import java.util.List;

import org.springframework.data.jpa.repository.JpaRepository;
import org.springframework.data.jpa.repository.Query;

import com.rms.bean.Customer;



public interface CustomerRepo extends JpaRepository<Customer, Integer> {

@Query("select cid from  Customer")
List<Integer> findAllId();

}
```

Restaurant Repositary

```
package com.rms.repositary;

import java.util.List;

import org.springframework.data.jpa.repository.JpaRepository;
import org.springframework.data.jpa.repository.Query;
```

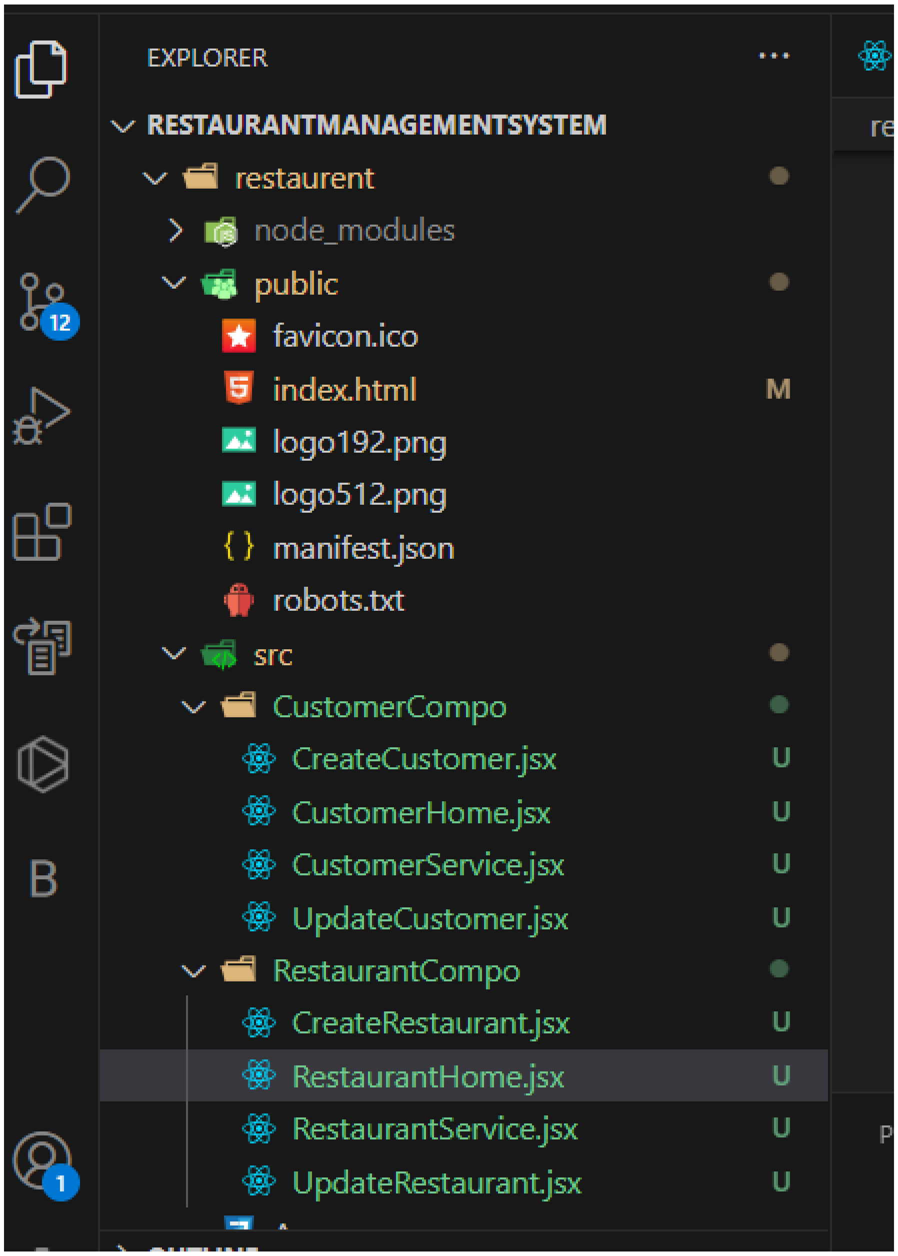
```
import
org.springframework.transaction.annotation.Transactional;

import com.rms.bean.Restaurant;



public interface RestaurantRepo extends
JpaRepository<Restaurant, Integer> {

@Transactional
public Restaurant deleteByRname(String rname);

@Query("select rid from Restaurant")
public List<Integer> getAllDepId();


}
```

Application.properties

```
spring.application.name=RestaurentManagementSystem
```

```
server.port=1234
spring.jpa.hibernate.ddl-auto=update
spring.datasource.url=jdbc:mysql://localhost:3306/assessment
spring.datasource.username=root
spring.datasource.password=Password@12345
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
spring.jpa.show-sql: true
```

REACT JS SOURCE CODE



```
EXPLORER                                   ...
∨ RESTAURANTMANAGEMENTSYSTEM
  ∨ restaurent                              ●
    > node_modules
    ∨ public                                ●
        favicon.ico
        index.html                          M
        logo192.png
        logo512.png
        manifest.json
        robots.txt
    ∨ src                                   ●
      ∨ CustomerCompo                       ●
          CreateCustomer.jsx                U
          CustomerHome.jsx                  U
          CustomerService.jsx               U
          UpdateCustomer.jsx                U
      ∨ RestaurantCompo                     ●
          CreateRestaurant.jsx              U
          RestaurantHome.jsx                U
          RestaurantService.jsx             U
          UpdateRestaurant.jsx              U
```

APP.JS

```
import './App.css';
import Container from 'react-bootstrap/Container';
import Nav from 'react-bootstrap/Nav';
import Navbar from 'react-bootstrap/Navbar';
import { BrowserRouter, Route, Routes } from 'react-router-dom';
import RestaurantHome from './RestaurantCompo/RestaurantHome';
import CreateRestaurant from './RestaurantCompo/CreateRestaurant';
import UpdateRestaurant from './RestaurantCompo/UpdateRestaurant';
import CustomerHome from './CustomerCompo/CustomerHome';
import CreateCustomer from './CustomerCompo/CreateCustomer';
import UpdateCustomer from './CustomerCompo/UpdateCustomer';
function App() {
  return (
    <div className="App">

        <Navbar expand="lg" className="bg-body-tertiary" style={{height:'80px'}}>
      <Container >
        <Navbar.Brand href="#home">R   M   S</Navbar.Brand>
        <Navbar.Toggle aria-controls="basic-navbar-nav" />
        <Navbar.Collapse id="basic-navbar-nav">
          <Nav className="me-auto">
            <Nav.Link                                                    href="/"
style={{paddingLeft:'70%',fontSize:'25px'}}>CUSTOMER</Nav.Link>

            <Nav.Link                                         href="/restaurantHome"
style={{paddingLeft:'70%',fontSize:'25px'}}>RESTAURANT</Nav.Link>


          </Nav>
        </Navbar.Collapse>
      </Container>
    </Navbar>
    <BrowserRouter>
<Routes>

   <Route path='/' element={<CustomerHome/>}></Route>
  <Route path='/createCustomer' element={<CreateCustomer/>}></Route>
  <Route path='/updateCustomer/:cid' element={<UpdateCustomer/>}></Route>


  <Route path='/restaurantHome' element={<RestaurantHome />}></Route>
  <Route path='/createRestaurant' element={<CreateRestaurant/>}></Route>
  <Route path='/updateRestaurant/:rid' element={<UpdateRestaurant/>}></Route>


</Routes>
</BrowserRouter>
      </div>

  );
}

export default App;
```

CreateRestaurant.jsx

```
import React, { useState } from 'react'
import { Link, useNavigate } from 'react-router-dom'
import RestaurantService from './RestaurantService'

const CreateRestaurant = () => {
    const [rid, setrid] = useState("")
    const [rname, setrname] = useState("")
    const [rcity, setrcity] = useState("")

    const navigate = useNavigate();
    const saveRestaurant=(e)=>{
        e.preventDefault();

        const departmentDetails={rid,rname,rcity}
        RestaurantService.createRestaurant(departmentDetails).then((response)=>{
            console.log(response.data);
            alert("Data: "+ response.data);
            navigate('/restaurantHome');
        }).catch((error)=>{
            console.log(error);
            alert("Error");
        })

    }
  return (
    <div>
        <div className='container ' style={{marginTop:"40px"}}>
        <center><h1>Create Restaurant</h1></center>
        <div
style={{height:'10px',width:'50%',marginLeft:'350px',marginTop:'50px'}}>

        <form onSubmit={(e)=>saveRestaurant(e)}>


        <div className="form-floating mb-3">
                                <input     type="text"     className="form-control"
id="floatingPassword"   placeholder="Department  Id"  required  name='EmployeeName'
value={rid}
                                 onChange={(e)=> setrid(e.target.value)}/>
                                <label      htmlFor="floatingPassword">Restaurant
Id</label>
                            </div>

                            <div className="form-floating mb-3">
                                <input     type="text"     className="form-control"
id="floatingPassword"  placeholder="Department Nmae" required name='EmployeeName'
value={rname}
                                onChange={(e)=> setrname(e.target.value)}/>
                                <label      htmlFor="floatingPassword">Restaurant
Name</label>
                            </div>

                            <div className="form-floating mb-3">
                                <input     type="text"     className="form-control"
id="floatingPassword"  placeholder="Department Nmae" required name='EmployeeName'
value={rcity}
                                onChange={(e)=> setrcity(e.target.value)}/>
                                <label      htmlFor="floatingPassword">Restaurant
City</label>
                            </div>
        <button className='btn btn-success'>Insert</button>
        <Link                                 to='/restaurantHome'className='btn'
style={{backgroundColor:'skyblue',fontWeight:'bold'}}
>BACK</Link><br></br><br></br>






        </form>
        </div>
      </div>
    </div>
  )
}

export default CreateRestaurant
```

Restaurant home .jsx

```
import React, { useState } from 'react'
import { Link, useNavigate } from 'react-router-dom'
import RestaurantService from './RestaurantService'

const CreateRestaurant = () => {
    const [rid, setrid] = useState("")
    const [rname, setrname] = useState("")
    const [rcity, setrcity] = useState("")

    const navigate = useNavigate();
    const saveRestaurant=(e)=>{
        e.preventDefault();

        const departmentDetails={rid,rname,rcity}
        RestaurantService.createRestaurant(departmentDetails).then((response)=>{
            console.log(response.data);
            alert("Data: "+ response.data);
            navigate('/restaurantHome');
        }).catch((error)=>{
            console.log(error);
            alert("Error");
        })

    }
  return (
    <div>
        <div className='container ' style={{marginTop:"40px"}}>
        <center><h1>Create Restaurant</h1></center>
        <div
style={{height:'10px',width:'50%',marginLeft:'350px',marginTop:'50px'}}>

        <form onSubmit={(e)=>saveRestaurant(e)}>


        <div className="form-floating mb-3">
                                <input    type="text"    className="form-control"
id="floatingPassword"   placeholder="Department  Id"  required  name='EmployeeName'
value={rid}
                                 onChange={(e)=> setrid(e.target.value)}/>
                                <label      htmlFor="floatingPassword">Restaurant
Id</label>
                            </div>

                            <div className="form-floating mb-3">
                                <input    type="text"    className="form-control"
id="floatingPassword"  placeholder="Department Nmae" required name='EmployeeName'
value={rname}
                                onChange={(e)=> setrname(e.target.value)}/>
                                <label      htmlFor="floatingPassword">Restaurant
Name</label>
                            </div>

                            <div className="form-floating mb-3">
                                <input    type="text"    className="form-control"
id="floatingPassword"  placeholder="Department Nmae" required name='EmployeeName'
value={rcity}
                                onChange={(e)=> setrcity(e.target.value)}/>
                                <label      htmlFor="floatingPassword">Restaurant
City</label>
                            </div>
        <button className='btn btn-success'>Insert</button>
        <Link                                 to='/restaurantHome'className='btn'
style={{backgroundColor:'skyblue',fontWeight:'bold'}}
>BACK</Link><br></br><br></br>






        </form>
        </div>
      </div>
    </div>
  )
}

export default CreateRestaurant
```

Restaurant Service .jsx

```
import { Component } from "react";
import axios from "axios";

 const createRes="http://localhost:1234/insetRestaurant";

 const getAllRes="http://localhost:1234/getAllRestaurants";

const updateRes="http://localhost:1234/updateResDetails";

 const deleteRes="http://localhost:1234/deleteResId";

 const findRestaurant="http://localhost:1234/findResById";

class RestaurantService extends Component {
    createRestaurant=(create)=>{
        return axios.post(createRes,create);
    }

    getAllRestaurant=()=>{
        return axios.get(getAllRes);
    }

    deleteRestaurant=(rid)=>{
            return axios.delete(deleteRes+'/'+rid);
    }

    findResDetails =(rid)=>{
        return axios.get(findRestaurant+'/'+rid);
    }

    updateRestaurant =(updaterid)=>{
        return axios.put(updateRes,updaterid);
    }

}
// eslint-disable-next-line import/no-anonymous-default-export
export default new RestaurantService();
```

UpdateRestaurant.jsx

```
import React, { useEffect, useState } from 'react'
import { useNavigate, useParams } from 'react-router';
import RestaurantService from './RestaurantService';

const UpdateRestaurant = () => {

    const navigate = useNavigate();
    const {rid}=useParams();
    const [res, setRes] = useState({
        rname: " ",
        rcity:" "
    })
    const { rname,rcity } = res;
    const passToUpdate=(e) => {
        e.preventDefault();

        const restaurantDetails={rid,rname,rcity}
        console.log(restaurantDetails);
        RestaurantService.updateRestaurant(restaurantDetails).then((response)  =>
{
            console.log(response.data);
          alert("updated Restaurant")
            navigate('/restaurantHome')
        }).catch((error) => {
                console.log("error")
        })
    }


    useEffect(()=>{
        RestaurantService.findResDetails(rid).then((response)=>{
            console.log(response.data);
            setRes(response.data);
        })
    },[])



  return (
    <div>
      <div>

        <div className='container ' style={{marginLeft:"40%",marginTop:"10%"}}>
          <h1>Edit Department</h1>
          <form onSubmit= {(e)=>passToUpdate(e)}>


          <input type='text' placeholder='Employee Name'
          value={rid} readOnly/>
          <br></br>
          <br></br>
          <input type='text' placeholder='Employee Name' required
          value={res.rname} name='rname'
          onChange={(e) => setRes({ ...res, [e.target.name]: e.target.value })} />
           <br></br>
          <br></br>
          <input type='text' placeholder='Employee Name' required
          value={res.rcity} name='rcity'
          onChange={(e) => setRes({ ...res, [e.target.name]: e.target.value })} />
          <br></br>
          <br></br>
          <input  type='submit'    class="btn  btn-primary"  value='Update'  />
</form>

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

export default UpdateRestaurant
```

Customer Home.jsx

```
import React, { useEffect, useState } from 'react'
import CustomerService from './CustomerService';
import { Link } from 'react-router-dom';
import CreateCustomer from './CreateCustomer';

const CustomerHome = () => {
  const[allDeatils,setAllDeatils]=useState([]);

    useEffect(()=>{
           CustomerService.findALLCusDetails().then((res)=>{
            console.log(res.data);
                setAllDeatils(res.data);
           }).catch((err)=>{
            console.log(err);
           })
    },[]);

    const  deleteCustomer =(cid)=>{


      CustomerService.deleteCustomer(cid).then((res)=>{

       console.log(cid);
           alert("Customer Record Deleted");
           window.location.reload();
      }).catch((err)=>{
       console.log(err);
      })


   }

  return (
    <div>
    <div className='container mt-5 ' >
     <div>
           <Link                              to='/createCustomer'className='btn'
style={{backgroundColor:'#5df542',fontWeight:'bold'}}    element={<CreateCustomer
/>}>CRETAE</Link><br></br><br></br>

 </div>
 <table className="table">
 <thead style={{fontStyle:'oblique'}}>
 <tr >
   <th        scope="col"style={{        fontSize:        '1.2rem',        padding:
'10px',backgroundColor:'#464f44',color:'white'}} >Customer ID</th>
   <th        scope="col"style={{        fontSize:        '1.2rem',        padding:
'10px',backgroundColor:'#464f44',color:'white'}}>Customer Name</th>
   <th        scope="col"style={{        fontSize:        '1.2rem',        padding:
'10px',backgroundColor:'#464f44',color:'white'}}>Customer Address</th>
   <th        scope="col"style={{        fontSize:        '1.2rem',        padding:
'10px',backgroundColor:'#464f44',color:'white'}}>Customer Age</th>
   <th        scope="col"style={{        fontSize:        '1.2rem',        padding:
'10px',backgroundColor:'#464f44',color:'white'}}>Customer DOB</th>
   <th        scope="col"style={{        fontSize:        '1.2rem',        padding:
'10px',backgroundColor:'#464f44',color:'white'}}>Customer Phone</th>
   <th        scope="col"style={{        fontSize:        '1.2rem',        padding:
'10px',backgroundColor:'#464f44',color:'white'}}>Actions</th>
 </tr>
 </thead>
```

```
  <tbody style={{ fontSize: '1.1rem' }}>
  {allDeatils.map((cus) => (
 <tr key={cus.cid}>
   <td style={{ fontSize: '1.1rem', padding: '10px' }}>{cus.cid}</td>
   <td>{cus.cname}</td>
   <td>{cus.caddress}</td>
   <td>{cus.cage}</td>
   <td>{cus.cdob}</td>
   <td>{cus.cphone}</td>


   <Link             to={`/updateCustomer/${cus.cid}`}             className='btn'
style={{backgroundColor:'blue',color:'black',fontWeight:'bold'}}>UPDATE</Link>
     <button                                                     className='btn'
style={{backgroundColor:'red',color:'black',fontWeight:'bold'}}onClick={()=>delet
eCustomer(cus.cid)} >DELETE</button>

     </tr>
))}
</tbody>
</table>

 </div>


 </div>
)
}



export default CustomerHome
```

CreateCustomer.jsx

```
import React, { useEffect, useState } from 'react'
import { useNavigate } from 'react-router';
import CustomerService from './CustomerService';

const CreateCustomer = () => {
    const [cid,setCid]=useState("");
    const [cname, setCname] = useState("");
    const [caddress,setCaddress]=useState("");
    const [cage,setCage] =useState("");
    const [cdob,setCdob]=useState("");
    const [cphone,setCphone]=useState("");

    const [rid,setRid]=useState("");


    const [rids,setRids]=useState([])
    const navigate=useNavigate();


   const sendCusDetails= (e)=>{
    e.preventDefault();

    const CusDeatails={
        cid,cname,caddress,cage,cdob,cphone,
        res:{
            rid
           }
    };

    console.log(CusDeatails);

    CustomerService.CustomerInsert(CusDeatails).then((response)=>{
        console.log(response.data);
        alert("Data Inserted");
        navigate('/');
    }).catch((error)=>{
        console.log(error);
    })
    }

    useEffect(()=>{
        CustomerService.getAllRestaurantId().then((response)=>{
            console.log(response.data);
            setRids(response.data);
        })
    },[])


  return (
    <div>
        <br></br>
      <div className='container ' >
      <center><h1>Create Customer</h1></center>
        <div
style={{height:'10px',width:'70%',marginLeft:'170px',marginTop:'50px'}}>
        <form onSubmit= {(e)=>sendCusDetails(e)}>

        <div className="form-floating mb-3">
                                <select className="form-control" required
                                    onChange={(e) => { setRid(e.target.value) }}>

                                    <option>Restaurant Ids</option>
                                    {rids.map((rids) => {
                                        return (<option key={rids} value={rids}
                                        >{rids}</option>)
                                    })
                                    }
                                </select>

                            </div>




        <br></br>
        <div className="form-floating mb-3">
                                <input    type="text"    className="form-control"
id="floatingPassword"     placeholder="Customer   Id"   required   name='ticketId'
value={cid}
                                onChange={(e)=> setCid(e.target.value)}/>
                                <label        htmlFor="floatingPassword">Customer
Id</label>
                            </div>

                            <div className="form-floating mb-3">
                                <input    type="text"    className="form-control"
id="floatingPassword"   placeholder="Customer  Name"  required  name='EmployeeName'
value={cname}
                                  onChange={(e)=> setCname(e.target.value)}/>
                                <label        htmlFor="floatingPassword">Customer
Name</label>
                            </div>

                            <div className="form-floating mb-3">
                                <input    type="text"    className="form-control"
id="floatingPassword"           placeholder="Customer      Address"      required
name='EmployeeSalary' value={caddress}
                                    onChange={(e)=>
setCaddress(e.target.value)}/>
                                <label        htmlFor="floatingPassword">Customer
Address</label>
                            </div>

                            <div className="form-floating mb-3">
                                <input    type="text"    className="form-control"
id="floatingPassword"  placeholder="Customer Age" required name='EmployeeSalary'
value={cage}
                                    onChange={(e)=> setCage(e.target.value)}/>
                                <label        htmlFor="floatingPassword">Customer
Age</label>
                            </div>
                            <div className="form-floating mb-3">
                                <input    type="text"    className="form-control"
id="floatingPassword"  placeholder="Customer DOB" required name='EmployeeSalary'
value={cdob}
                                    onChange={(e)=> setCdob(e.target.value)}/>
                                <label        htmlFor="floatingPassword">Customer
DOB</label>
                            </div>
                            <div className="form-floating mb-3">
                                <input    type="text"    className="form-control"
id="floatingPassword"  placeholder="Customer Phone" required name='EmployeeSalary'
value={cphone}
                                    onChange={(e)=> setCphone(e.target.value)}/>
                                <label        htmlFor="floatingPassword">Customer
Phone</label>
                            </div>




         <input type='submit'  class="btn btn-primary" value='Submit' />
         {/*                      <Link                          to='/'className='btn'
style={{backgroundColor:'skyblue',fontWeight:'bold'}}
>BACK</Link><br></br><br></br>  */}
              </form>
              </div>
      </div>

    </div>
  )
}
export default CreateCustomer
```

Customer Service .jsx

```
import { Component } from "react";
import axios from "axios";

 const insertCustomer="http://localhost:1234/insertCustomer";

 const getResIdList="http://localhost:1234/getAllResId";

const AllDetails="http://localhost:1234/getAllCustomers";

 const deleteCustomer="http://localhost:1234/deleteCusById";


 const findCusbyId="http://localhost:1234/findCusById";

 const customerUpdate="http://localhost:1234/updateCustomerDetails";

class CustomerService extends Component{

    CustomerInsert=(deatils)=>{
        return axios.post(insertCustomer,deatils);
      };

    getAllRestaurantId =()=>{
       return axios.get(getResIdList);
    }

    findALLCusDetails (){
      return axios.get(AllDetails);
    }

    deleteCustomer (cid){
      return axios.delete(deleteCustomer+'/'+cid);
    }



    findCustomer= (cid)=>{
      return axios.get(findCusbyId+'/'+cid);
    }

    updateCusDetails = (updateemp)=>{

      return axios.put(customerUpdate,updateemp)
    }



}
// eslint-disable-next-line import/no-anonymous-default-export
export default new CustomerService();
```

Customer Update

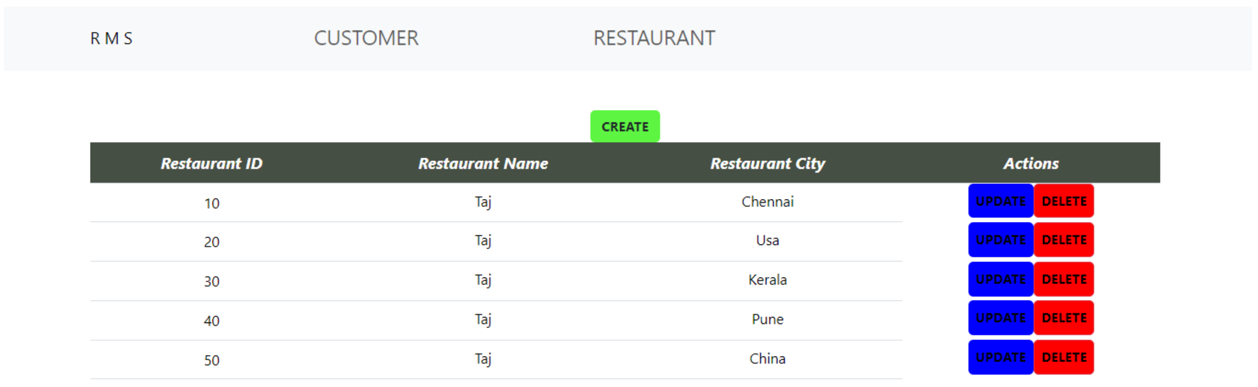
```
import React, { useEffect, useState } from 'react'
import { useParams } from 'react-router';
import CustomerService from './CustomerService';
import { Link } from 'react-router-dom';

const UpdateCustomer = () => {
    const [cname, setCname] = useState("");
    const [caddress,setCaddress]=useState("");
    const [cage,setCage] =useState("");
    const [cdob,setCdob]=useState("");
    const [cphone,setCphone]=useState("");

    const [rid,setRid]=useState("");


    const [customer, setCustomer] = useState({
        cname: " ",
        caddress:" ",cage:" ",cdob:" ",cphone:" "
    })

    const {cid}=useParams();
    const [rids,setRids]=useState([]);



   const passToUpdate=(e)=>{
    e.preventDefault();

    const CusDeatails={
        cid,cname,caddress,cage,cdob,cphone,
        res:{
            rid
           }
    };

    console.log(CusDeatails);
    alert("Data Inserted"+cid);
    CustomerService.updateCusDetails(CusDeatails).then((response)=>{
        console.log(response.data);

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

    useEffect(()=>{
        CustomerService.getAllRestaurantId().then((response)=>{
            console.log(response.data);
            setRids(response.data);
        })
    },[])

    useEffect(()=>{
        CustomerService.findCustomer(cid).then((response)=>{
            console.log(response.data);
            setCustomer(response.data);
        })
    },[])


  return (
    <div>
        <br></br>
      <div className='container ' >
      <center><h1>Update  Customer</h1></center>
        <div
style={{height:'10px',width:'70%',marginLeft:'170px',marginTop:'50px'}}>
        <form onSubmit= {(e)=>passToUpdate(e)}>


        <div className="form-floating mb-3">
                                <select className="form-control" required
                                    onChange={(e) => { setRid(e.target.value) }}>

                                    <option>Restaurant Ids</option>
                                    {rids.map((rids) => {
                                        return (<option key={rids} value={rids}
                                        >{rids}</option>)
                                    })
                                    }
                                </select>

                            </div>




        <br></br>
        <div className="form-floating mb-3">
                                <input     type="text"     className="form-control"
id="floatingPassword"      placeholder="Customer   Id"   required   name='ticketId'
value={cid}
                               />
                                <label        htmlFor="floatingPassword">Customer
Id</label>
                            </div>

                            <div className="form-floating mb-3">
                                <input     type="text"     className="form-control"
id="floatingPassword"      placeholder="Customer   Name"   required   name='cname'
value={customer.cname}
```

```
                                 onChange={(e)   =>   setCustomer({   ...customer,
[e.target.name]: e.target.value })}/>
                                <label        htmlFor="floatingPassword">Customer
Name</label>
                            </div>

                            <div className="form-floating mb-3">
                                <input    type="text"    className="form-control"
id="floatingPassword"   placeholder="Customer  Address"  required  name='caddress'
value={customer.caddress}
                                    onChange={(e)  =>  setCustomer({  ...customer,
[e.target.name]: e.target.value })}/>
                                <label        htmlFor="floatingPassword">Customer
Address</label>
                            </div>

                            <div className="form-floating mb-3">
                                <input    type="text"    className="form-control"
id="floatingPassword"      placeholder="Customer    Age"    required    name='cage'
value={customer.cage}
                                    onChange={(e)  =>  setCustomer({  ...customer,
[e.target.name]: e.target.value })}/>
                                <label        htmlFor="floatingPassword">Customer
Age</label>
                            </div>
                            <div className="form-floating mb-3">
                                <input    type="text"    className="form-control"
id="floatingPassword"      placeholder="Customer    DOB"    required    name='cdob'
value={customer.cdob}
                                    onChange={(e)  =>  setCustomer({  ...customer,
[e.target.name]: e.target.value })}/>
                                <label        htmlFor="floatingPassword">Customer
DOB</label>
                            </div>
                            <div className="form-floating mb-3">
                                <input    type="text"    className="form-control"
id="floatingPassword"     placeholder="Customer   Phone"   required   name='cphone'
value={customer.cphone}
                                    onChange={(e)  =>  setCustomer({  ...customer,
[e.target.name]: e.target.value })}/>
                                <label        htmlFor="floatingPassword">Customer
Phone</label>
                            </div>
```

```
         <input type='submit'  class="btn btn-primary" value='Submit' />
         <Link                                               to='/'className='btn'
style={{backgroundColor:'skyblue',fontWeight:'bold'}}
>BACK</Link><br></br><br></br>
              </form>
              </div>
      </div>

    </div>
  )
}
export default UpdateCustomer
```

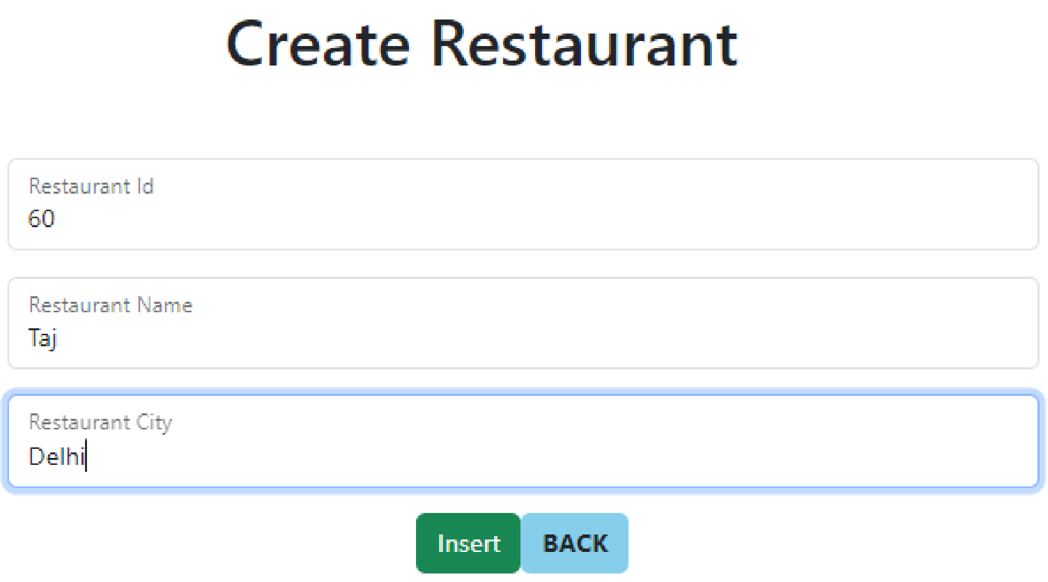
CRUD OPERATIONS

Restaurant Home Page



Restaurant ID	Restaurant Name	Restaurant City	Actions
10	Taj	Chennai	UPDATE DELETE
20	Taj	Usa	UPDATE DELETE
30	Taj	Kerala	UPDATE DELETE
40	Taj	Pune	UPDATE DELETE
50	Taj	China	UPDATE DELETE

CREATE PAGE



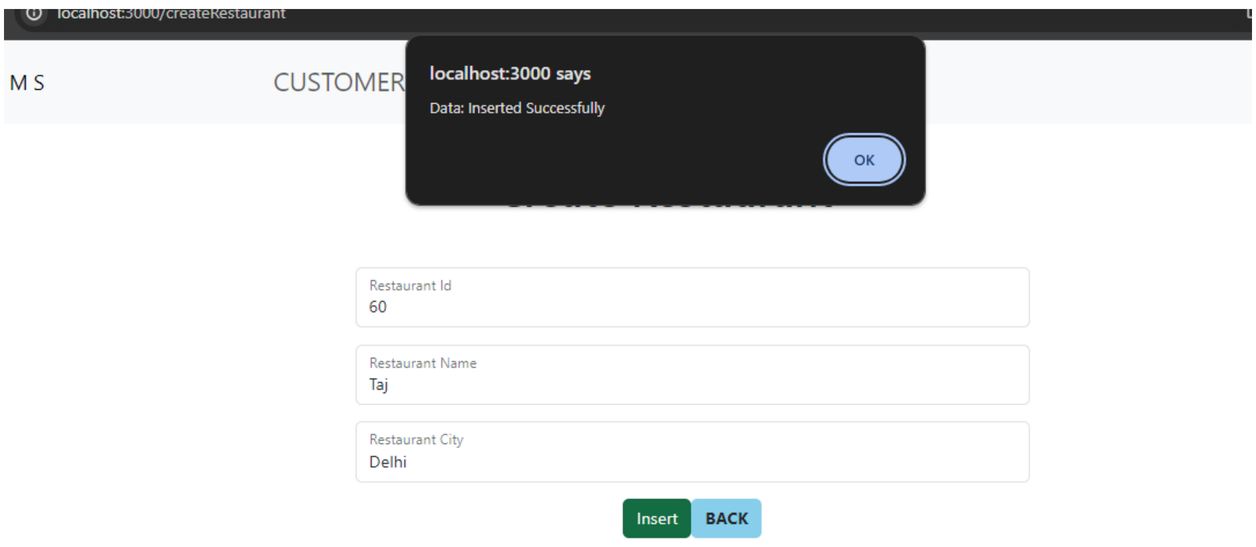
Create Restaurant

Restaurant Id
60

Restaurant Name
Taj

Restaurant City
Delhi

AFTER INSERTION



localhost:3000/createRestaurant

localhost:3000 says

Data: Inserted Successfully

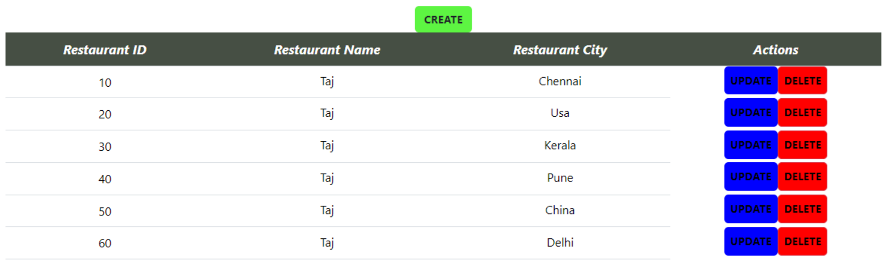
Restaurant Id
60

Restaurant Name
Taj

Restaurant City
Delhi

DELETE RECORD

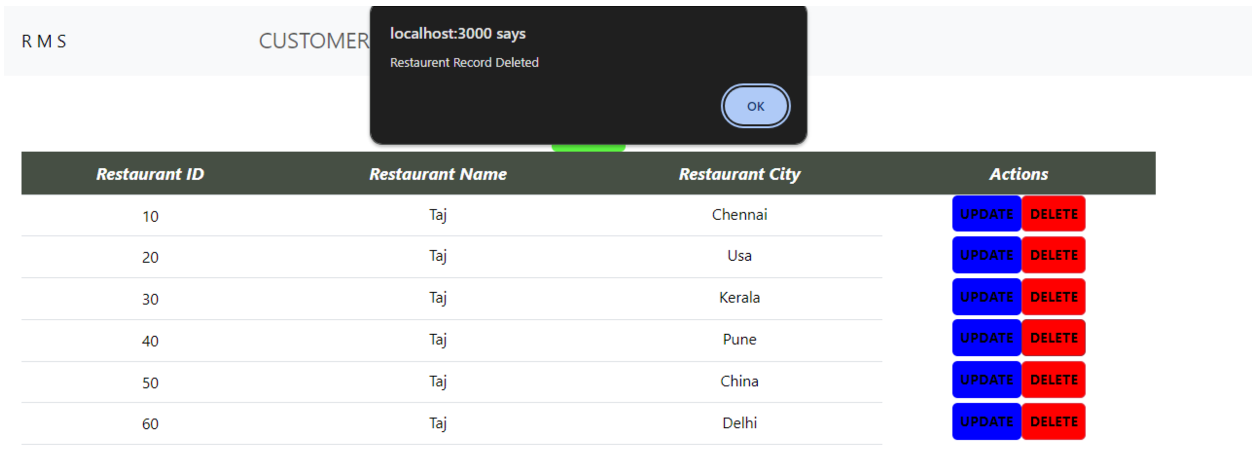
BEFORE DELETION



CREATE

Restaurant ID	Restaurant Name	Restaurant City	Actions
10	Taj	Chennai	UPDATE DELETE
20	Taj	Usa	UPDATE DELETE
30	Taj	Kerala	UPDATE DELETE
40	Taj	Pune	UPDATE DELETE
50	Taj	China	UPDATE DELETE
60	Taj	Delhi	UPDATE DELETE

WHILE DELETING



R M S   CUSTOMER

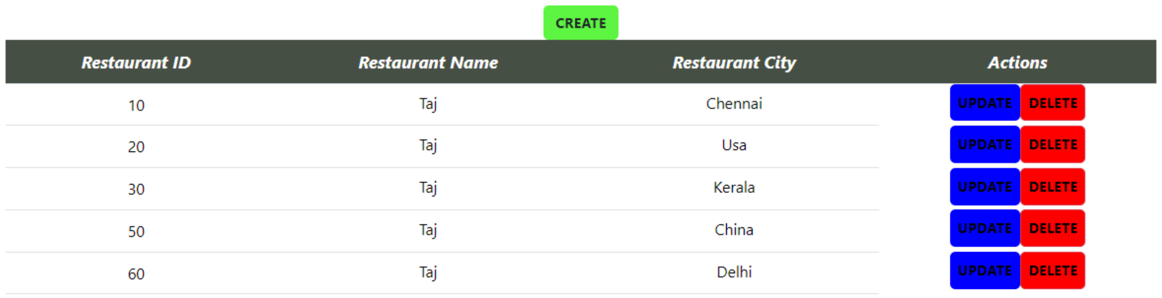
localhost:3000 says

Restaurent Record Deleted

OK

Restaurant ID	Restaurant Name	Restaurant City	Actions
10	Taj	Chennai	UPDATE DELETE
20	Taj	Usa	UPDATE DELETE
30	Taj	Kerala	UPDATE DELETE
40	Taj	Pune	UPDATE DELETE
50	Taj	China	UPDATE DELETE
60	Taj	Delhi	UPDATE DELETE

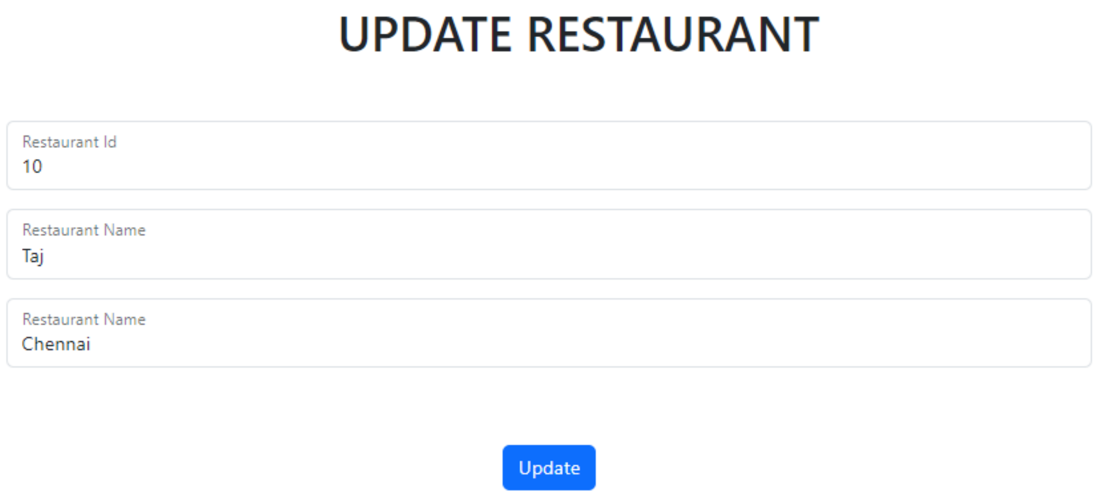
AFTER DELETION



CREATE

Restaurant ID	Restaurant Name	Restaurant City	Actions
10	Taj	Chennai	UPDATE DELETE
20	Taj	Usa	UPDATE DELETE
30	Taj	Kerala	UPDATE DELETE
50	Taj	China	UPDATE DELETE
60	Taj	Delhi	UPDATE DELETE

UPDATE RESTAURANT

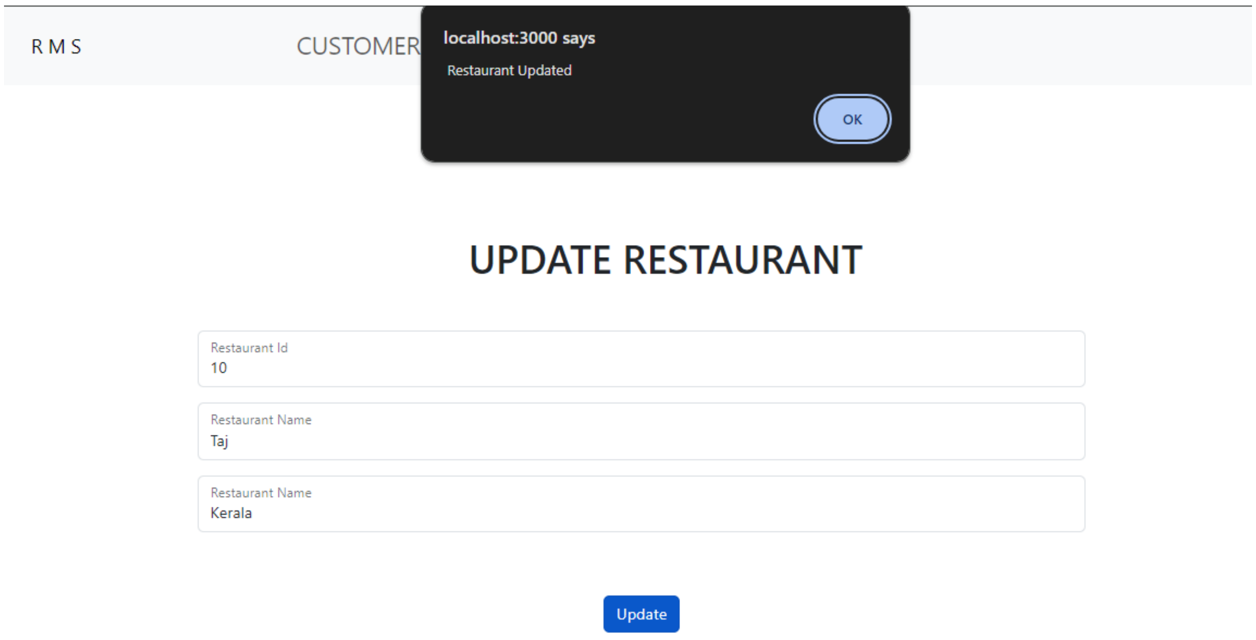


UPDATE RESTAURANT

Restaurant Id
10

Restaurant Name
Taj

Restaurant Name
Chennai



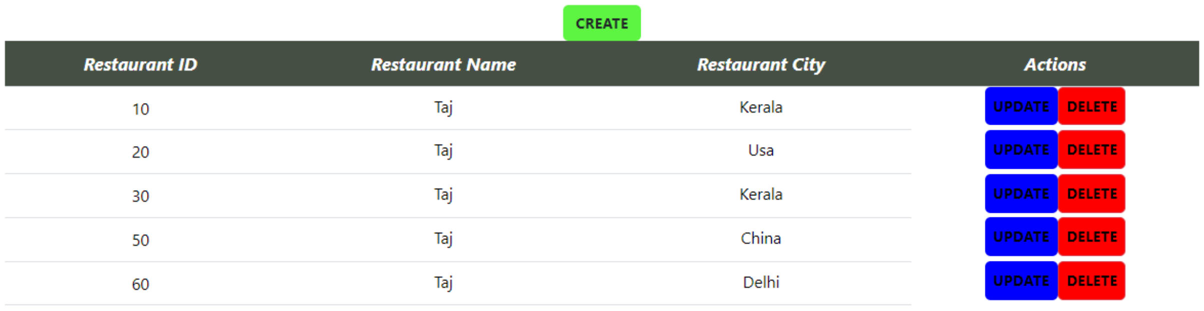
localhost:3000 says

Restaurant Updated

UPDATE RESTAURANT

Restaurant Id	10
Restaurant Name	Taj
Restaurant Name	Kerala

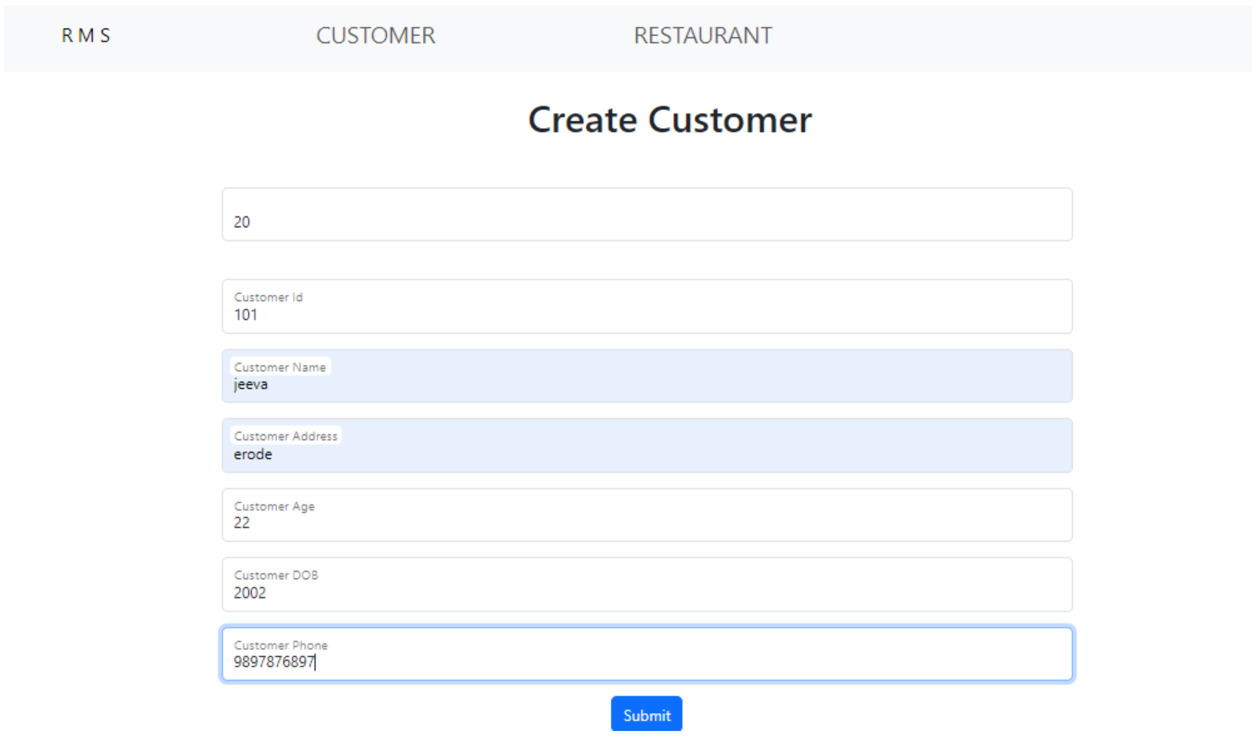
AFTER UPDATION



Restaurant ID	Restaurant Name	Restaurant City	Actions
10	Taj	Kerala	UPDATE DELETE
20	Taj	Usa	UPDATE DELETE
30	Taj	Kerala	UPDATE DELETE
50	Taj	China	UPDATE DELETE
60	Taj	Delhi	UPDATE DELETE

Customer Operations

Create customer



Create Customer

20

Customer Id
101

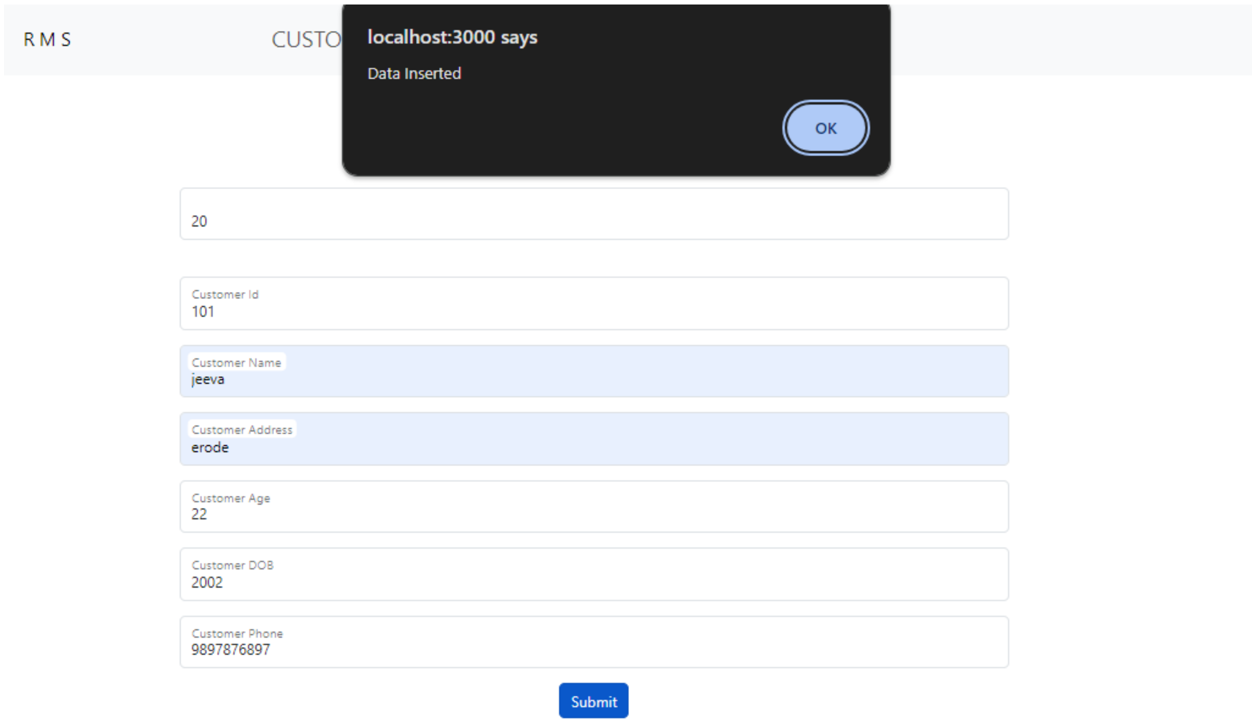
Customer Name
jeeva

Customer Address
erode

Customer Age
22

Customer DOB
2002

Customer Phone
9897876897



R M S CUSTO

localhost:3000 says

Data Inserted

OK

20

Customer Id
101

Customer Name
jeeva

Customer Address
erode

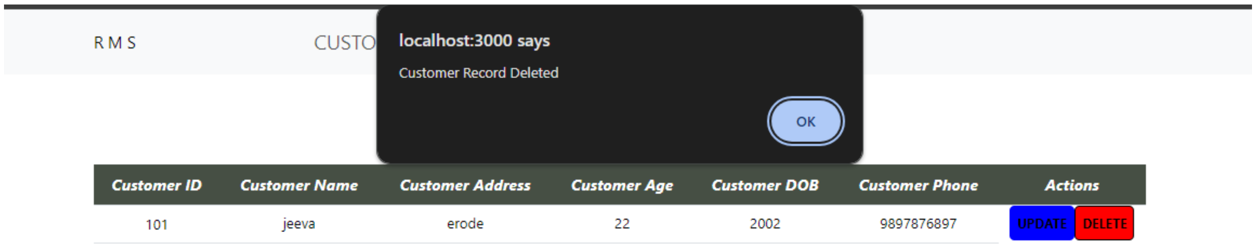
Customer Age
22

Customer DOB
2002

Customer Phone
9897876897

Submit

DELETE



R M S CUSTO

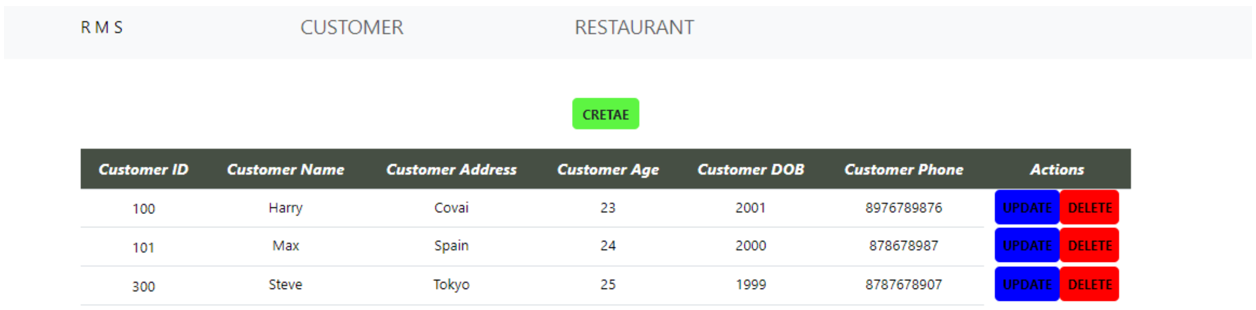
localhost:3000 says

Customer Record Deleted

OK

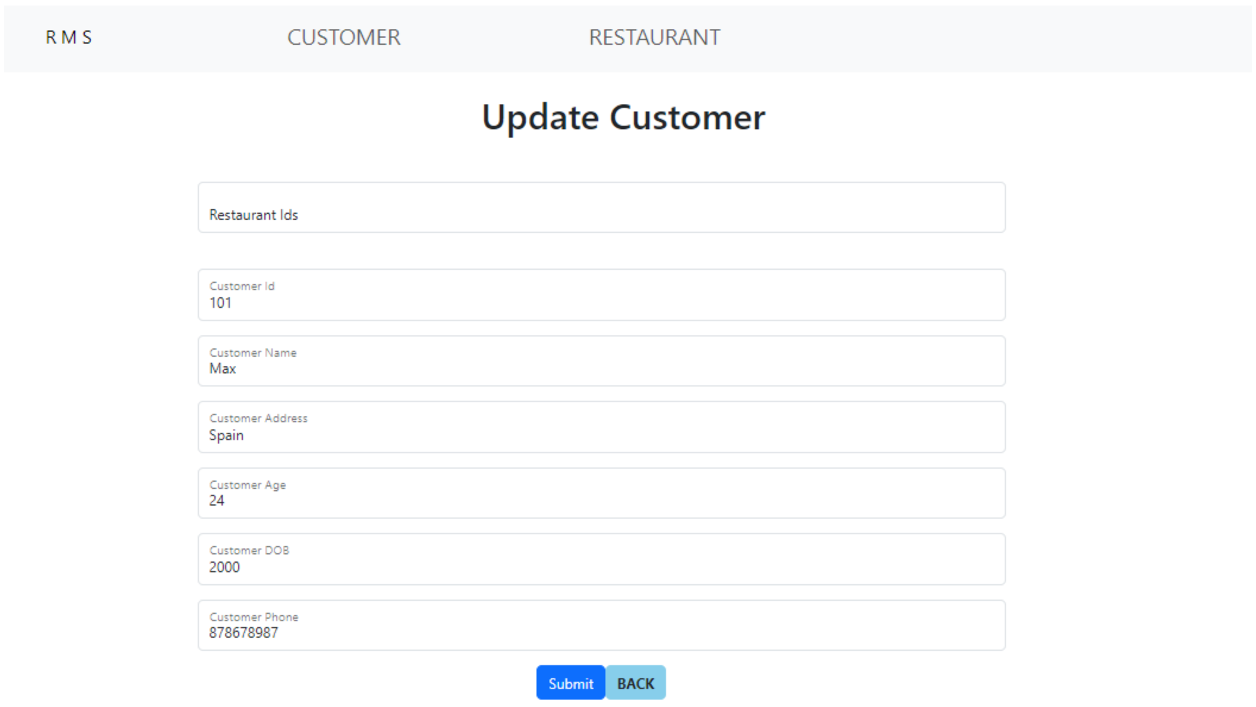
Customer ID	Customer Name	Customer Address	Customer Age	Customer DOB	Customer Phone	Actions
101	jeeva	erode	22	2002	9897876897	UPDATE DELETE

CUSTOMER HOME PAGE



Customer ID	Customer Name	Customer Address	Customer Age	Customer DOB	Customer Phone	Actions
100	Harry	Covai	23	2001	8976789876	UPDATE DELETE
101	Max	Spain	24	2000	878678987	UPDATE DELETE
300	Steve	Tokyo	25	1999	8787678907	UPDATE DELETE

UPDATE CUSTOMER



Update Customer

Restaurant Ids

Customer Id
101

Customer Name
Max

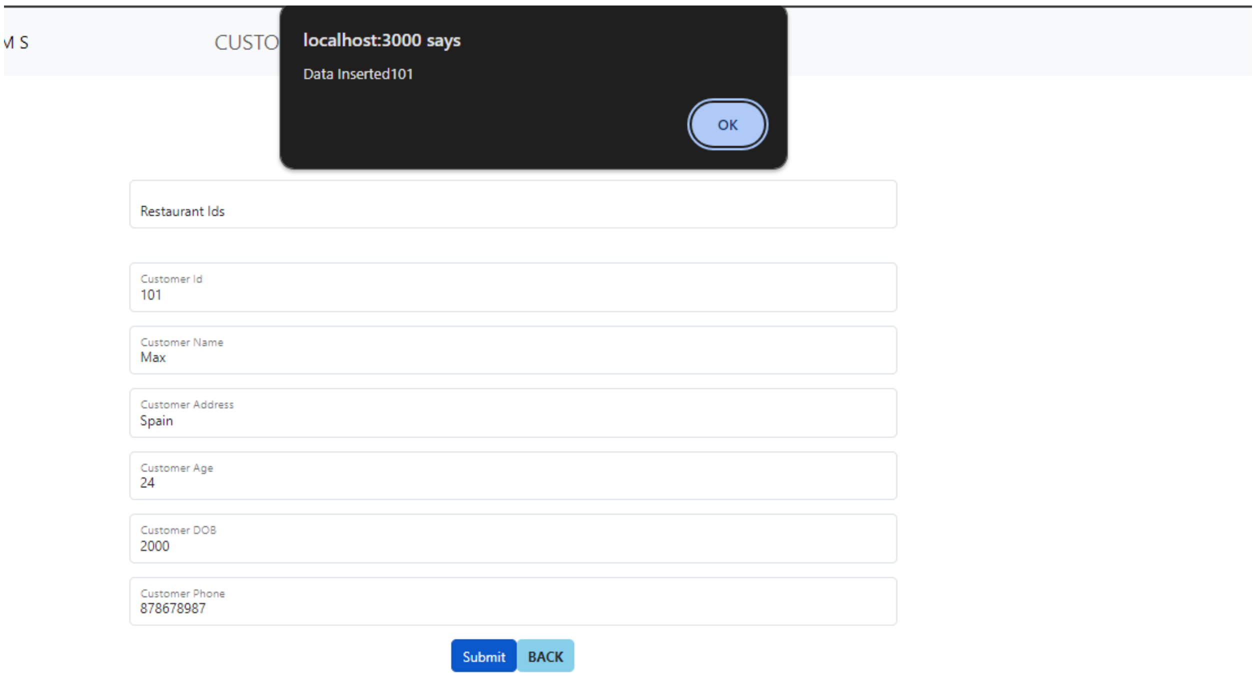
Customer Address
Spain

Customer Age
24

Customer DOB
2000

Customer Phone
878678987

UPDATION



localhost:3000 says

Data Inserted101

Restaurant Ids

Customer Id
101

Customer Name
Max

Customer Address
Spain

Customer Age
24

Customer DOB
2000

Customer Phone
878678987

THANKYOU