|  |
| --- |
| **A**  **PROJECT REPORT ON** |
|  |
|  |
| Restaurant Management System |
|  |
|  |
| SUBMITTED IN  PARTIAL FULFILLMENT OF  **DIPLOMA IN ADVANCED COMPUTING (PG-DAC)** |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
| **BY**  **Tejas Gopal Badgujar**  **Suruchi Sudhir Sonone**  **Shivam Gaur**  **Saurabh Anand Ladi**  **Ghanshyam Sahebrao Mali**  **Latika Kashinath Mitkari** |
|  |
|  |
| **UNDER THE GUIDENCE OF**  **Malkeet Singh** |
|  |
|  |
|  |
| **AT**  **Center For Development of Advance Computing,**  **Mumbai** |

|  |  |
| --- | --- |
|  | |
| **CERTIFICATE**  This is to certify that the project work under the title ‘Restaurant Management System’ is done by Team-9 in partial fulfillment of the requirement for award of Diploma in Advanced Computing Course.     |  |  | | --- | --- | | **Mr. Malkeet Singh** | **Mrs. Kiran Waghmare** | | **Project Guide** | **Course Co-ordinator** |     Date: 31 August 2023 | |
|  | |
|  | |
|  | |
|  | |
|  | |
|  | |
|  | |
|  | |
|  | |
|  | |
|  | |
|  | |
|  | |
|  | |
|  | |
|  | |
|  | |
|  | |
|  |  |
|  | |
|  | |
|  | |
|  | |
|  | |
|  |  |
|  |  |

**ACKNOWLEDGEMENT**

A project usually falls short of its expectation unless aided and guided by the right persons at the right time. We avail this opportunity to express our deep sense of gratitude towards **Mrs. Kiran Waghmare** (Course Coordinator, CDAC, Mumbai) .

We are deeply indebted and grateful to them for their guidance, encouragement and deep concern for our project. Without their critical evaluation and suggestions at every stage of the project, this project could never have reached its present form.

We express our sincere thanks to **Mr. Malkeet Singh** Sir for her valuable guidance and continuous support. Without her inspiration and help it would not have been possible for us to complete this project.

Last but not the least we thank the entire faculty and the staff members of CDAC, Mumbai for their support.

**Tejas Gopal Badgujar**

**Suruchi Sudhir Sonone**

**Shivam Gaur**

**Saurabh Anand Ladi**

**Ghanshyam Sahebrao Mali**

**Latika Kashinath Mitkari**

**ABSTRACT**

Managing a restaurant in a traditional way can be very time consuming and a complex process. To avoid this a restaurant management system has been created which is used to automate day to day activity of a restaurant. This system to provide service facility to the restaurant as well as to the customer. This system can be used by employees to handle clients and their orders. The customer can reserve table and also place their orders using the system and also keep track of their previous orders. The main point of the system is to help restaurant administrator manage the restaurant business and help customer according to their needs. The user can search for a menu according to their choice based on price and category.

**INDEX**

|  |  |  |
| --- | --- | --- |
|  | **INTRODUCTION** | 1 |
|  | 1.1 Introduction | 2 |
|  | **Product Overview and Summary** |  |
|  | 2.1 Purpose | 8 |
|  | 2.2 Scope | 8 |
|  | 2.3 User Classes and Characteristics | 8 |
|  | 2.4 Design and Implementation Constraints | 9 |
|  | **REQUIREMENTS** |  |
|  | 3.1 Functional Requirements |  |
|  | 3.1.1 Use case for Administrator. | 10 |
|  | 3.1.2 Use case for Customer. | 11 |
|  | 3.1.3 Use case for Manager. | 12 |
|  | 3.1.4 Use case for Chef | 13 |
|  | 3.1.5 Use case for Waiter | 14 |
|  | 3.1.6 Use case for Supplier. | 15 |
|  | 3.2 Non - Functional Requirements |  |
|  | 3.2.1 Entity Relationship Diagram | 16 |
|  | 3.2.2 Performance Requirement | 16 |
|  | 3.2.3 Constraint | 16 |
|  | 3.2.4 Other Requirement | 16 |
|  | 3.2.5 Interface | 25 |
|  | **PROJECT DESIGN** |  |
|  | 4.1 Data Model |  |
|  | 4.1.1 Database Design | 17 |
|  | 4.2 Coding Standards Implemented | 21 |
|  | 4.2.1 Class diagram | 22 |
|  | **TEST REPORT** | 23 |
|  | **PROJECT RELATED STATISTICS** | 24 |
|  | **REFERENCES** | 31 |
|  | **CONCLUSIONS** |  |

**LIST OF TABLES**

|  |  |  |
| --- | --- | --- |
| **Sr. No** | **Table Title** | **Page** |
| **1** | Administration | 17 |
| **2** | Customers | 17 |
| **3** | Ingredients | 18 |
| **4** | Menu\_details | 18 |
| **5** | Menu\_Ingredient | 18 |
| **6** | Order\_details | 19 |
| **7** | orders | 19 |
| **8** | Stock | 19 |
| **9** | Supplier\_Ingredient | 20 |
| **10** | Coding Standards | 21 |
| **11** | Test Report | 23 |
| **12** | Project Related Statistics | 24 |

**LIST OF FIGURES**

|  |  |  |
| --- | --- | --- |
| **Sr. No** | **Figure Title** | **Page** |
| **1** | Use case for owner | 10 |
| **2** | Use case for customer | 11 |
| **3** | Use care for manager | 12 |
| **4** | Use case for chef | 13 |
| **5** | Use case for waiter | 14 |
| **6** | Use case for supplier | 15 |
| **7** | E-R Diagram | 16 |
| **8** | Class diagram | 22 |
| **9** | Interface | 25 |

**1: INTRODUCTION**

* 1. **Introduction**

The main objective of this project is to develop a client/server model which deals with “Restaurant Management System”. This system has two parts, one part is for management side and other part is for customers.

After successful login the customer side allows customer to reserve table, view menu according to category and price and add food items to their cart and also check previous orders.

At the management side the staff is allowed to edit information regarding menu list, food price, assign chef, take ratings and reviews and also maintain information regarding placed orders.

**2: PRODUCT OVERVIEW AND SUMMARY**

**2.1 Purpose**

The main purpose of this system is to manage restaurant in an easy and less time consuming way. It is created to avoid rush by letting customer reserve table in advance and also place orders.

**2.2 Scope**

The primary scope of the restaurant management system project is well depicted on the user case diagrams that are well showcased in this report. However, the central system functionalities of the restaurant management system comprise generating reports, inventories, employee records, and managing orders. The order management involves the creation and deleting of orders, removing and adding food staff from a request, and closing the orders. All the orders made should be stored in the system's database and connected to the significant project tasks, with testing and implementation that consume marginally more time than design.

**2.3 User Classes and Characteristics**

The Restaurant Management System has 6 actors and 1 co-operating system. The actors involved are: Owner, Manager, Customer, Chef, Waiter & Supplier.

1. The Owner has access to full system and can manage employees, salaries, inventory and revenue management.
2. The Manager can manage orders, table reservations, orders and total analysis.
3. The customer can reserve table, view and order food items from menu, update orders, pay bill and give rating.
4. The chef can manage orders placed by customer and keep track of inventory.
5. The Waiter can access table and order management.
6. The Supplier can access inventory and supply ingredients as per restaurant’s needs.

**2.4 Design and Implementation Constraints**

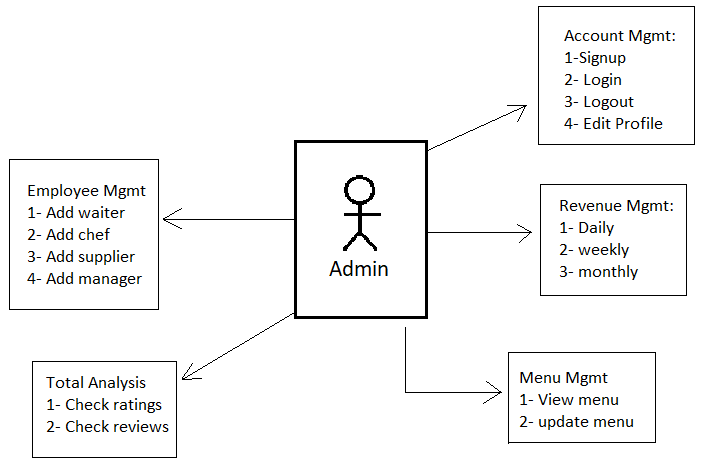
There are some constraints which costs more for the system. If those constraints are overcome then this whole system will perform best. They are:

1. iOS, Windows and Android apps
2. Faster server system such as LINUX Server.
3. Add Hindi, Marathi Langauage.
4. Ways to make system more secure.

**3: REQUIREMENTS**

**3.1 Functional Requirements**

I- Use case for Owner



**A- User Management**

1. Customer
2. Manager
3. Staff

**C- Menu Management**

1. View menu
2. Update menu

**B- Payment Management**

1. Manage salary(Waiter, chef, manager)
2. Inventory Payment

**C- Revenue Management**

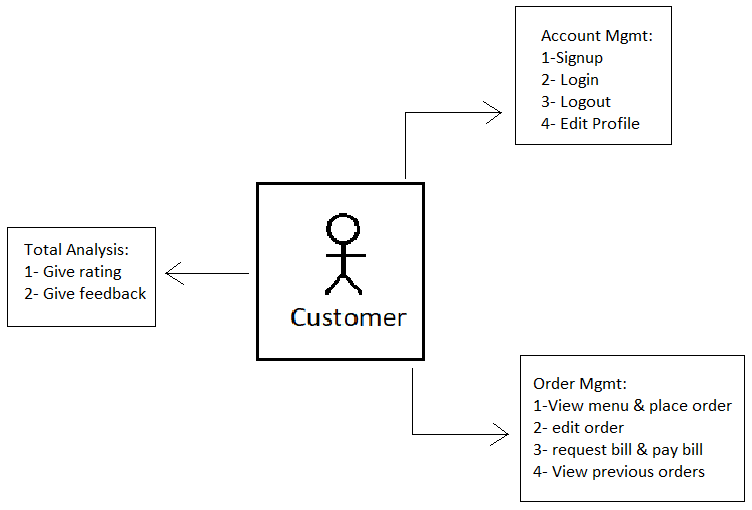
1. Monthly, Weekly, Daily revenue.

**D-** Edit Profiles

**E- Total Analysis**

1. Check ratings
2. Check reviews

II- Use case for Customer



**A- Account Management**

1. Customer can create account by signing up.
2. Customer can Login and also edit information as needed.
3. Customer can logout as per their need.

**B- Order Management**

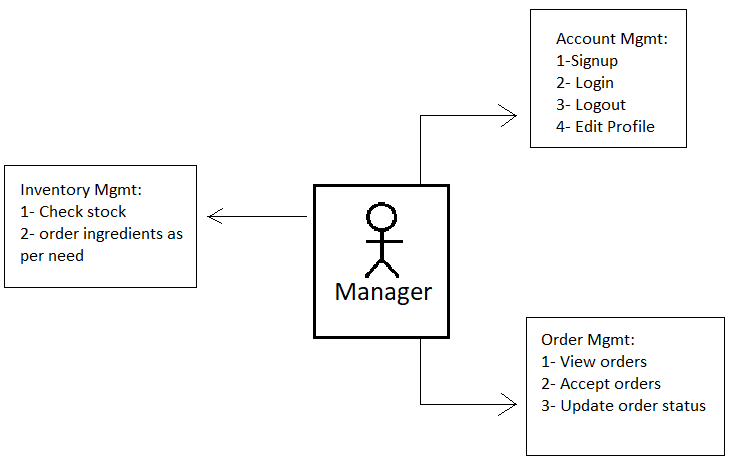
1. Customer can view menu.
2. Customer can place order according to category and price.
3. View previous orders.

**C-** Customer should pay the bill

**D- Analysis**

1. Customer can give rating
2. Customer can give feedback.

III- Use case for Manager



**A- Account Management**

1. Manager can create account by signing up.
2. Manager can Login and also edit information as needed.
3. Manager can logout as per their need.

**B- Order Management**

1. Check Table reservation
2. Check Orders
3. Assign chef and waiter for an occupied table.
4. Cancel order incase needed.

**C- Inventory Management**

1. Check Inventory and order ingredients accordingly.

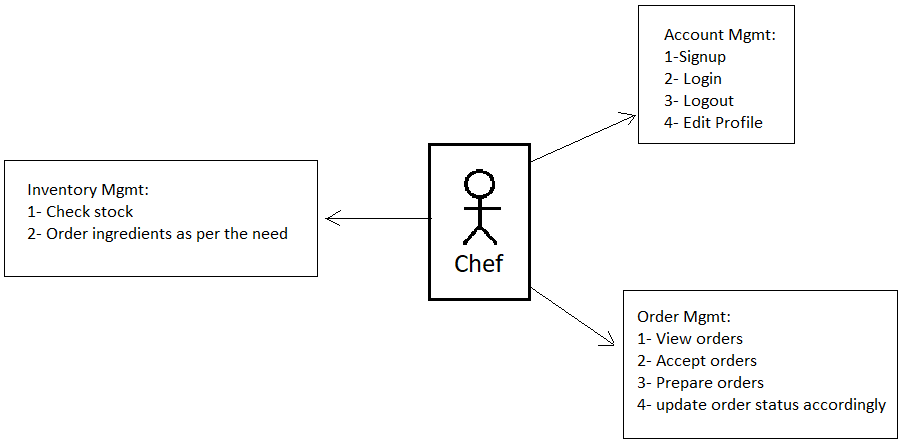
**D- Bill Payments**

1. Check bill and collect payment from customer

**E- Analysis**

1. Ask for rating
2. Take feedback.

IV- Use case for Chef



**A- Account Management**

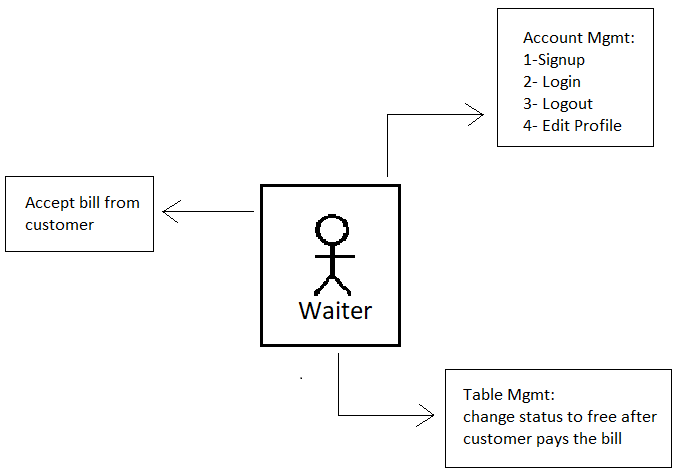
1. Chef can create account by signing up.
2. Chef can Login and also edit information as needed.
3. Chef can logout as per their need.

**B- Order Management**

1. Chef should be able to check orders that are placed
2. Chef should prepare orders.
3. Chef should check for any special requirements
4. Chef should update status of orders

**C-** Chef should be able to keep track of inventory.

V- Use case for Waiter



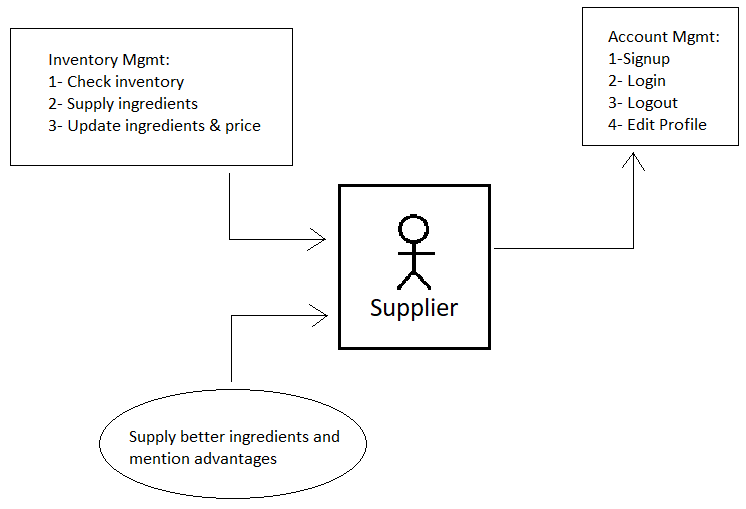
**A- Account Management**

1. Waiter can create account by signing up.
2. Waiter can Login and also edit information as needed.
3. Waiter can logout as per their need.

**B- Table Management**

1. Waiter should clear previous table.
2. Waiter should take customer’s order.
3. Change table status to free after bill paid by customer.

VI- Use case for Supplier



**A- Account Management**

1. Supplier can create account by signing up.
2. Supplier can Login and also edit information as needed.
3. Supplier can logout as per their need.

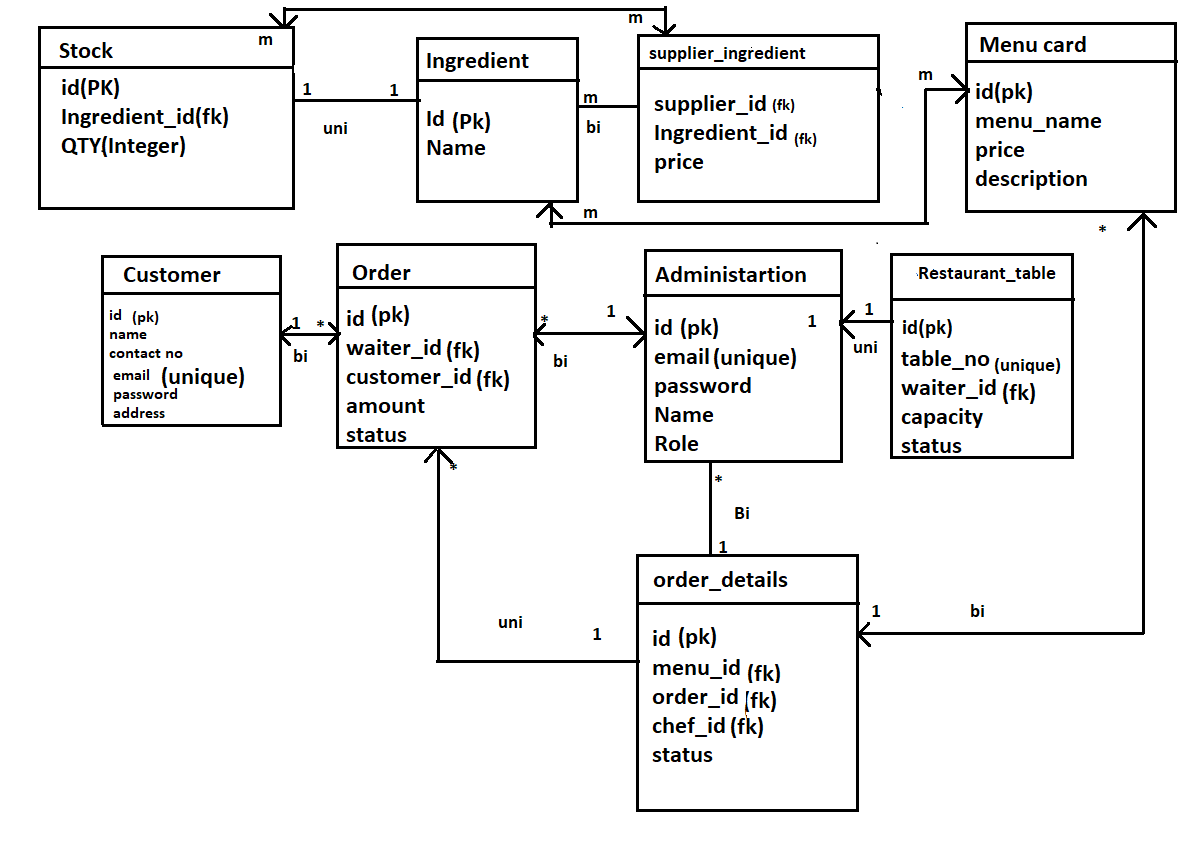
**B- Inventory Management**

1. Supplier should supply ingredients that are ordered.
2. Supplier can update prices of ingredients.

**C-** Supplier can supply better alternatives.

**3.2: Non-functional Requirements**

I- ER Diagram



II- Performance Requirement

III- Constraint

IV- Other Requirement

**3.2.5: Interface**

Refer to Appendix A

**4: PROJECT DESIGN**

* 1. **Database Design**

Table1: Administrator

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Key Type/ Constraint** | **Column Name** | **Data Type** | **Length** | **Allow Null (1=Yes;0=No)** |
| 3 | Id | Int | 4 | 0 |
| 0 | Email | Varchar | 6 | 1 |
| 0 | Name | Varchar | 20 | 1 |
| 0 | Password | Varchar | 20 | 1 |
| 0 | Role | Varchar | 20 | 1 |

Table2: Customers

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Key Type/ Constraint** | **Column Name** | **Data Type** | **Length** | **Allow Null (1=Yes;0=No)** |
| 3 | Id | Int | 4 | 0 |
| 0 | City | Varchar | 6 | 1 |
| 0 | Country | Varchar | 20 | 1 |
| 0 | State | Varchar | 20 | 1 |
| 0 | Contact | Varchar | 20 | 1 |
| 0 | email | Varchar | 20 | 1 |
| 0 | name | Varchar | 20 | 1 |
| 0 | password | varchar | 20 | 0 |

Table3: Ingredients

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Key Type/ Constraint** | **Column Name** | **Data Type** | **Length** | **Allow Null (1=Yes;0=No)** |
| 3 | Id | Int | 4 | 0 |
| 0 | email | Varchar | 20 | 1 |

Table4: Menu\_details

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Key Type/ Constraint** | **Column Name** | **Data Type** | **Length** | **Allow Null (1=Yes;0=No)** |
| 3 | Id | Int | 4 | 0 |
| 0 | Category | Varchar | 20 | 1 |
| 0 | Menu\_name | Varchar | 20 | 1 |
| 0 | Price | Varchar | 20 | 1 |

Table5: Menu\_Ingredient

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Key Type/ Constraint** | **Column Name** | **Data Type** | **Length** | **Allow Null (1=Yes;0=No)** |
| 3 | Menu\_Id | Int | 4 | 1 |
| 0 | Ingredient\_id | Int | 4 | 1 |

Table6: Order\_details

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Key Type/ Constraint** | **Column Name** | **Data Type** | **Length** | **Allow Null (1=Yes;0=No)** |
| 3 | Id | Int | 4 | 0 |
| 0 | Status | Varchar | 20 | 1 |
| 0 | Chef\_id | Int | 20 | 1 |
| 0 | Menu\_id | int | 20 | 1 |
| 0 | Order\_id | int | 20 | 1 |

Table7: Orders

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Key Type/ Constraint** | **Column Name** | **Data Type** | **Length** | **Allow Null (1=Yes;0=No)** |
| 3 | Id | Int | 4 | 0 |
| 0 | Amount | Double | 20 | 1 |
| 0 | Order\_category | Varchar | 20 | 1 |
| 0 | Order\_status | Varchar | 20 | 1 |
| 0 | Customer\_id | Int | 4 | 1 |
| 0 | Waiter\_id | int | 4 | 1 |

Table8: Stock

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Key Type/ Constraint** | **Column Name** | **Data Type** | **Length** | **Allow Null (1=Yes;0=No)** |
| 3 | Ingredient\_Id | Int | 4 | 0 |
| 0 | quantity | int | 4 | 0 |

Table9: Supplier\_Ingredient

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Key Type/ Constraint** | **Column Name** | **Data Type** | **Length** | **Allow Null (1=Yes;0=No)** |
| 3 | Ingredient\_Id | Int | 4 | 0 |
| 0 | Supplier\_id | int | 20 | 0 |
| 0 | price | double | 20 | 1 |

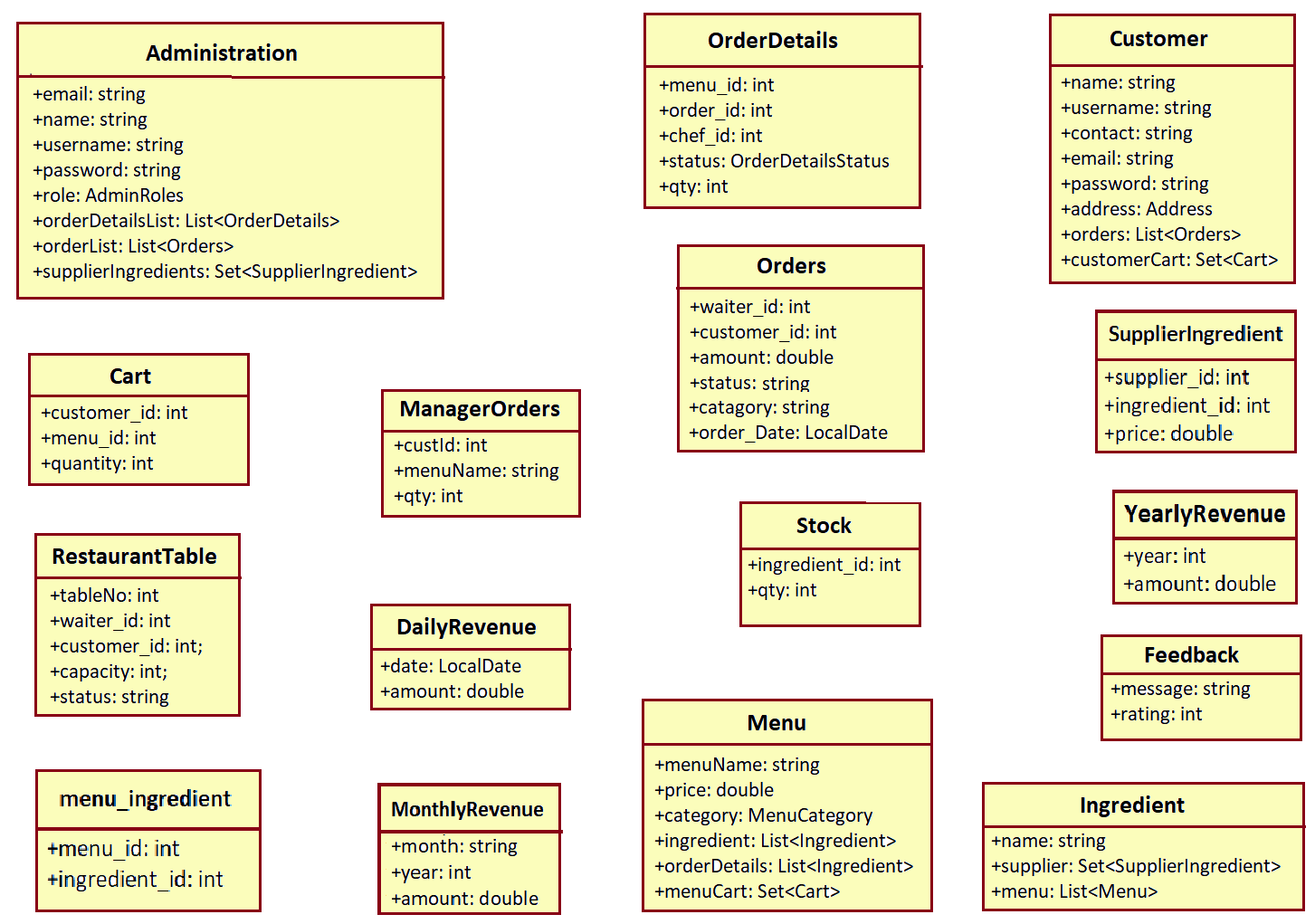
* 1. **Coding Standards implemented**

### Naming and Capitalization

Below summarizes the naming recommendations for identifiers in Pascal casing is used mainly (i.e. capitalize first letter of each word) with camel casing (capitalize each word except for the first one) being used in certain circumstances.

|  |  |  |  |
| --- | --- | --- | --- |
| **Identifier** | **Case** | **Examples** | **Additional Notes** |
| Class | Pascal | Person, BankVault, SMSMessage, Dept | Class names should be based on "objects" or "real things" and should generally be **nouns**. No ‘\_’ signs allowed. Do not use type prefixes like ‘C’ for class. |
| Method | Camel | getDetails, updateStore | Methods should use **verbs** or verb phrases. |
| Parameter | Camel | personName, bankCode | Use descriptive parameter names. Parameter names should be descriptive enough that the name of the parameter and its type can be used to determine its meaning in most scenarios. |
| Interface | Pascal with "I" prefix | Disposable | Do not use the ‘\_’ sign |
| Property | Pascal | ForeColor, BackColor | Use a noun or noun phrase to name properties. |
| Associated private member variable | \_camelCase | \_foreColor, \_backColor | Use underscore camel casing for the private member variables |
| Exception Class | Pascal with "Exception" suffix | WebException, |  |

* + 1. **Class Diagram**

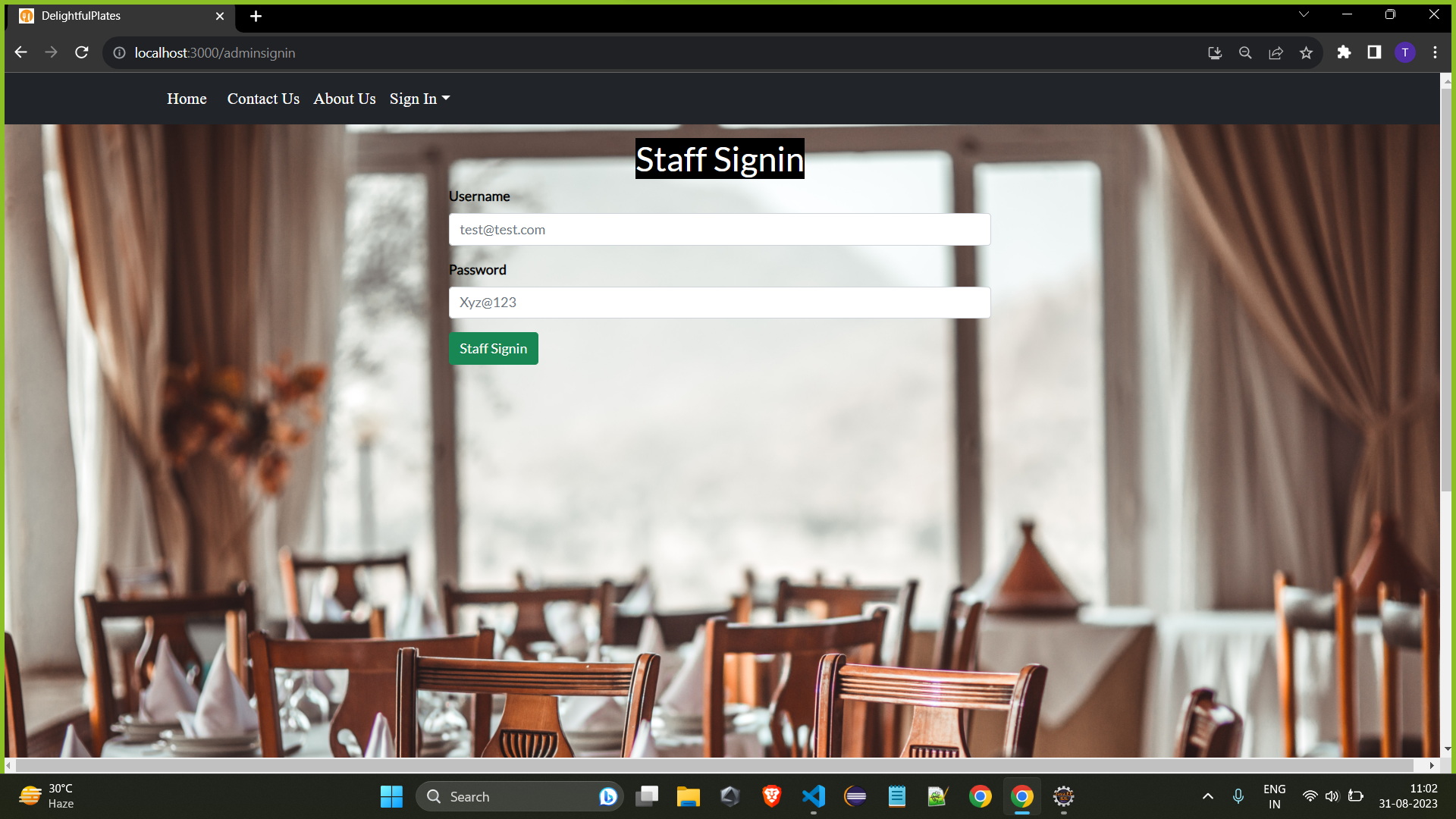


**5: TEST REPORT**

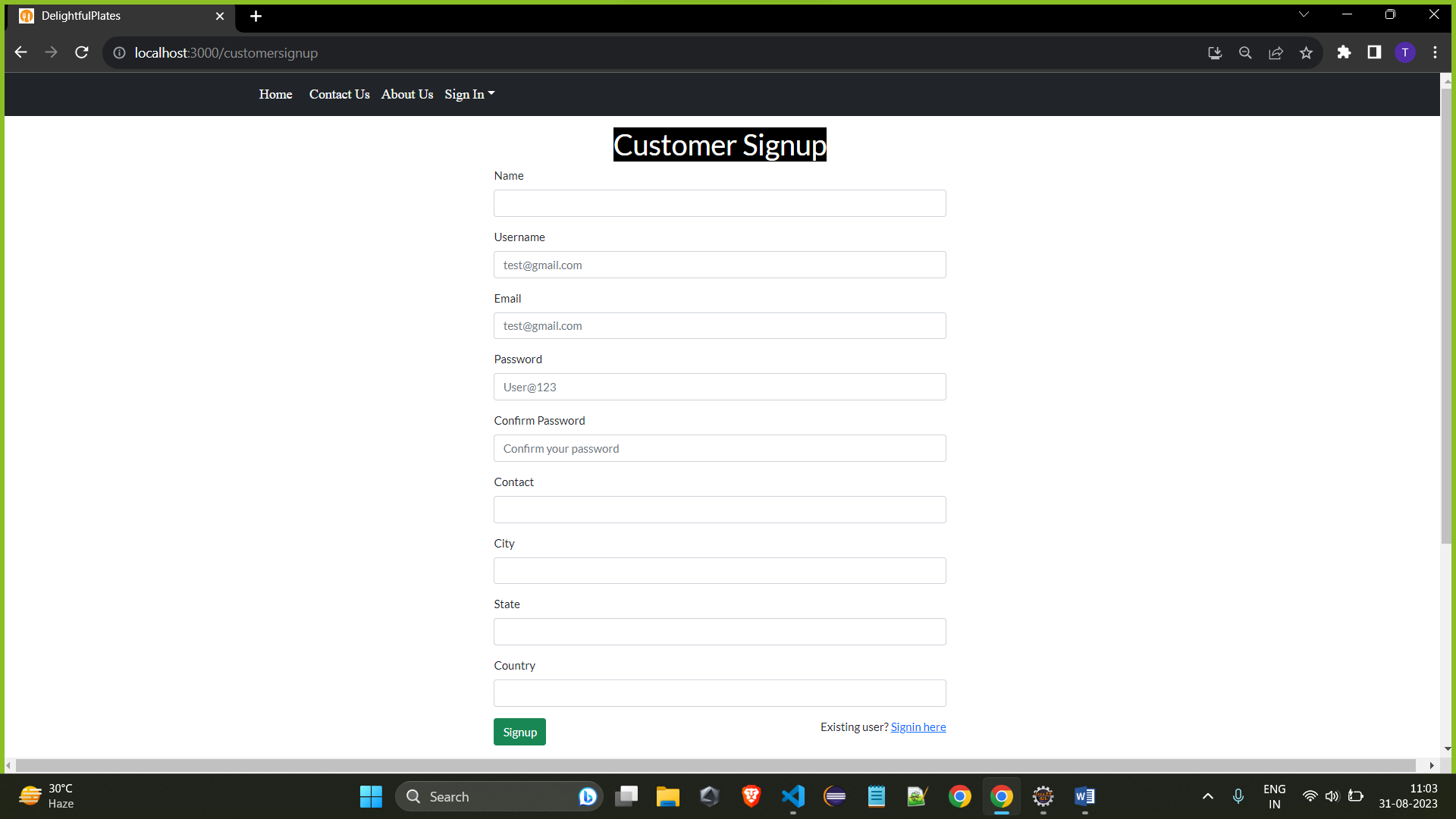
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **SR.**  **NO** | **TEST CASE** | **EXPECTED RESULT** | **ACTUAL RESULT** | **ERROR MESSAGE** |
| 1 | Sign Up Page | Used to sign up | Ok | No error |
| 2 | Login page | Used to Log in | Ok | No error |
| 3 | Logout page | Used to log out | Ok | No error |
| 4 | Menu | Shows restaurant menu | Ok | No error |
| 5 | Add food items | Adds food items to cart | Ok | No error |
| 6 | Table reservation | User can reserve table | Ok | No error |
| 7 | Revenue mgmt. | Owner can see daily, weekly, monthly revenue | Ok | No error |

**APPENDIX A**

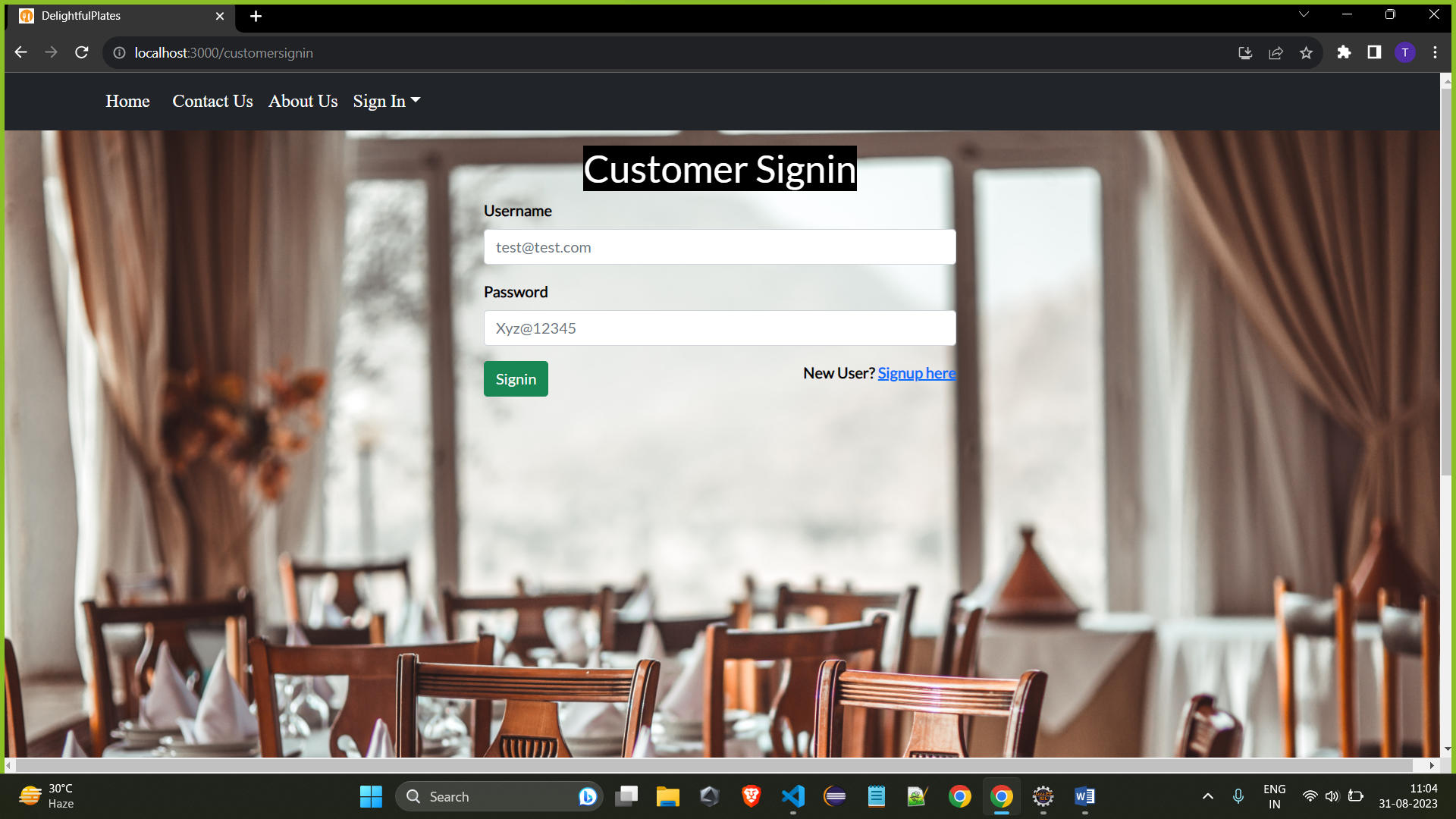
1: Admin SignIn



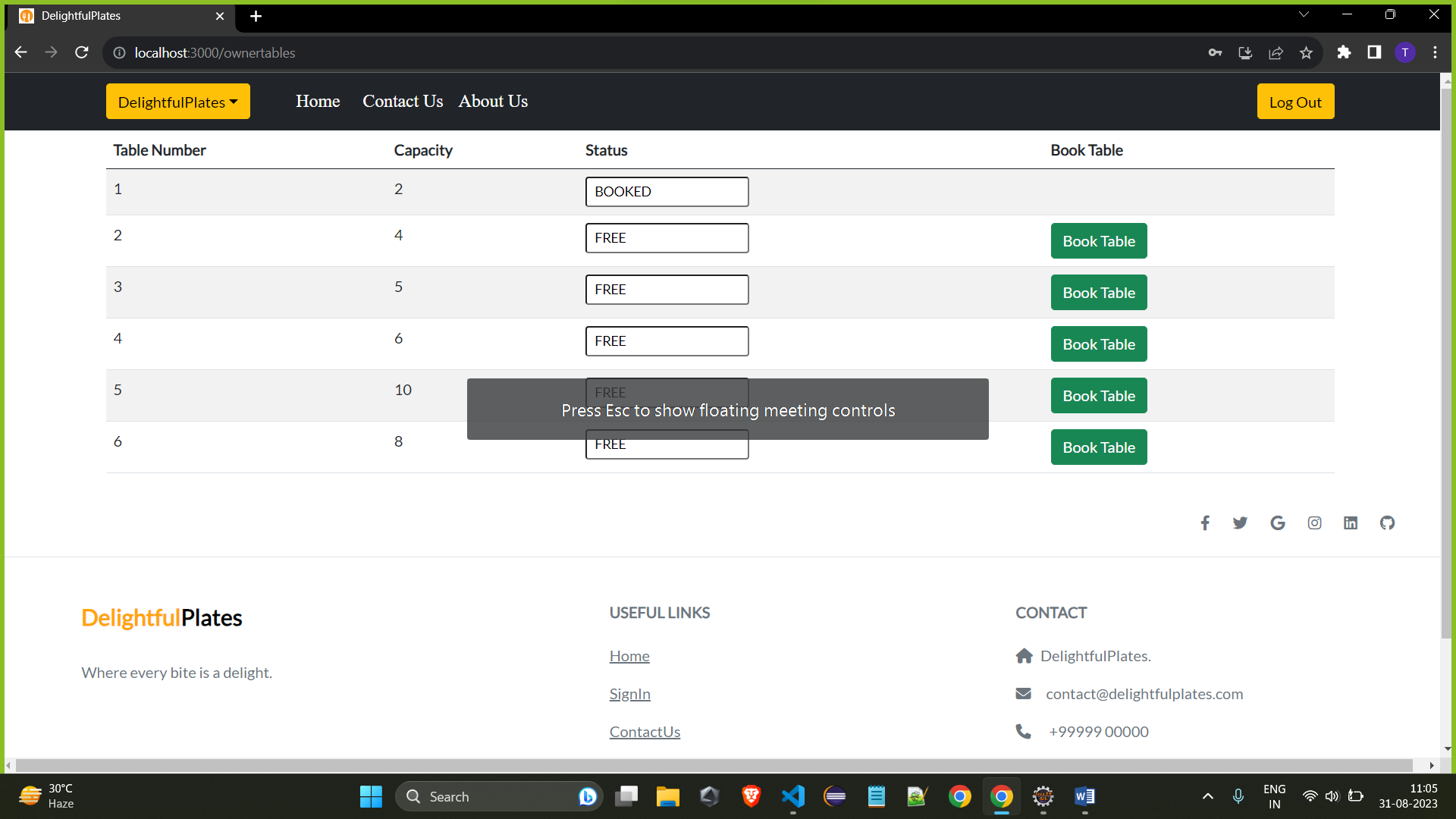
2: Customer SignUp



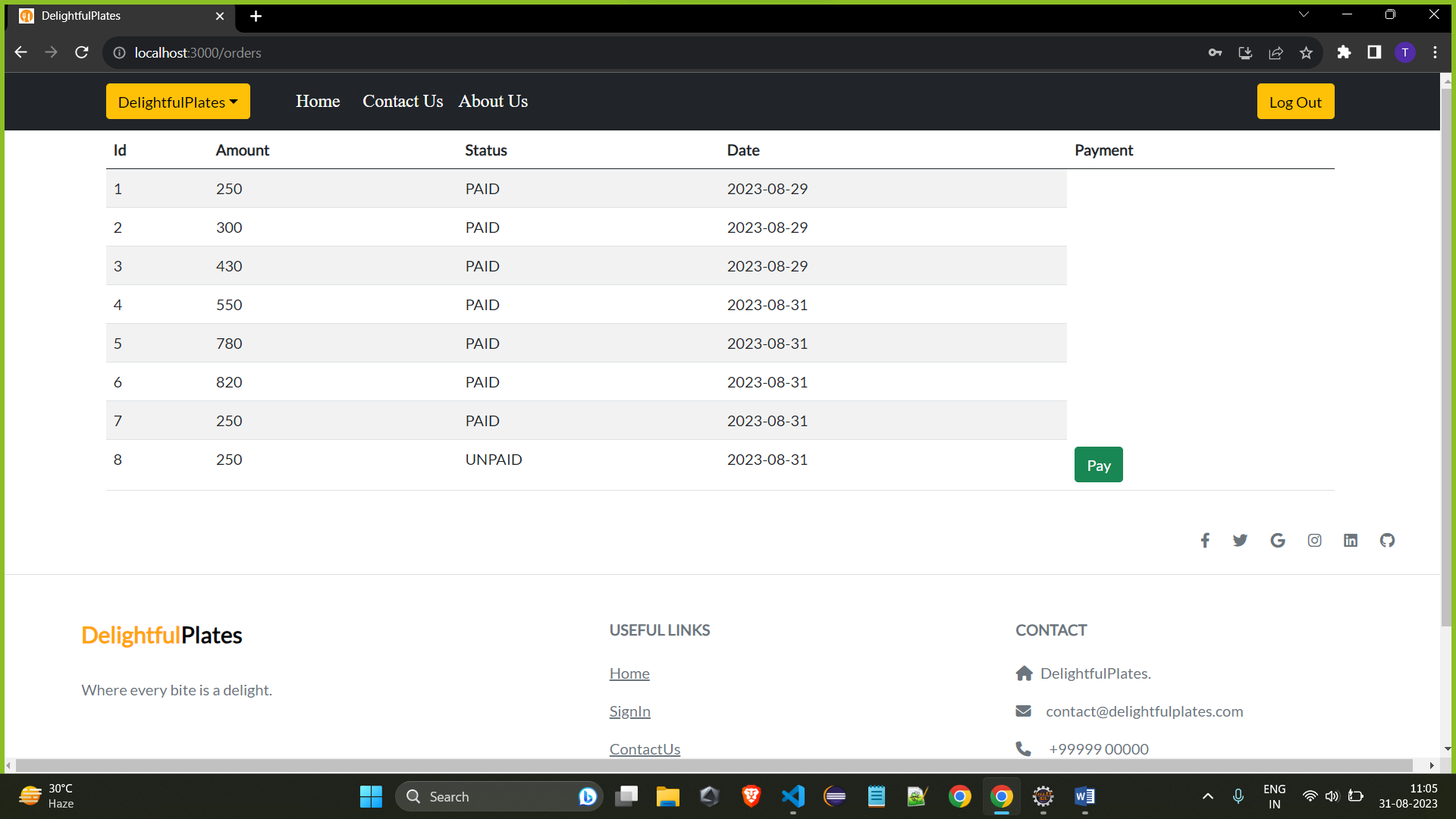
3: Customer Sign In



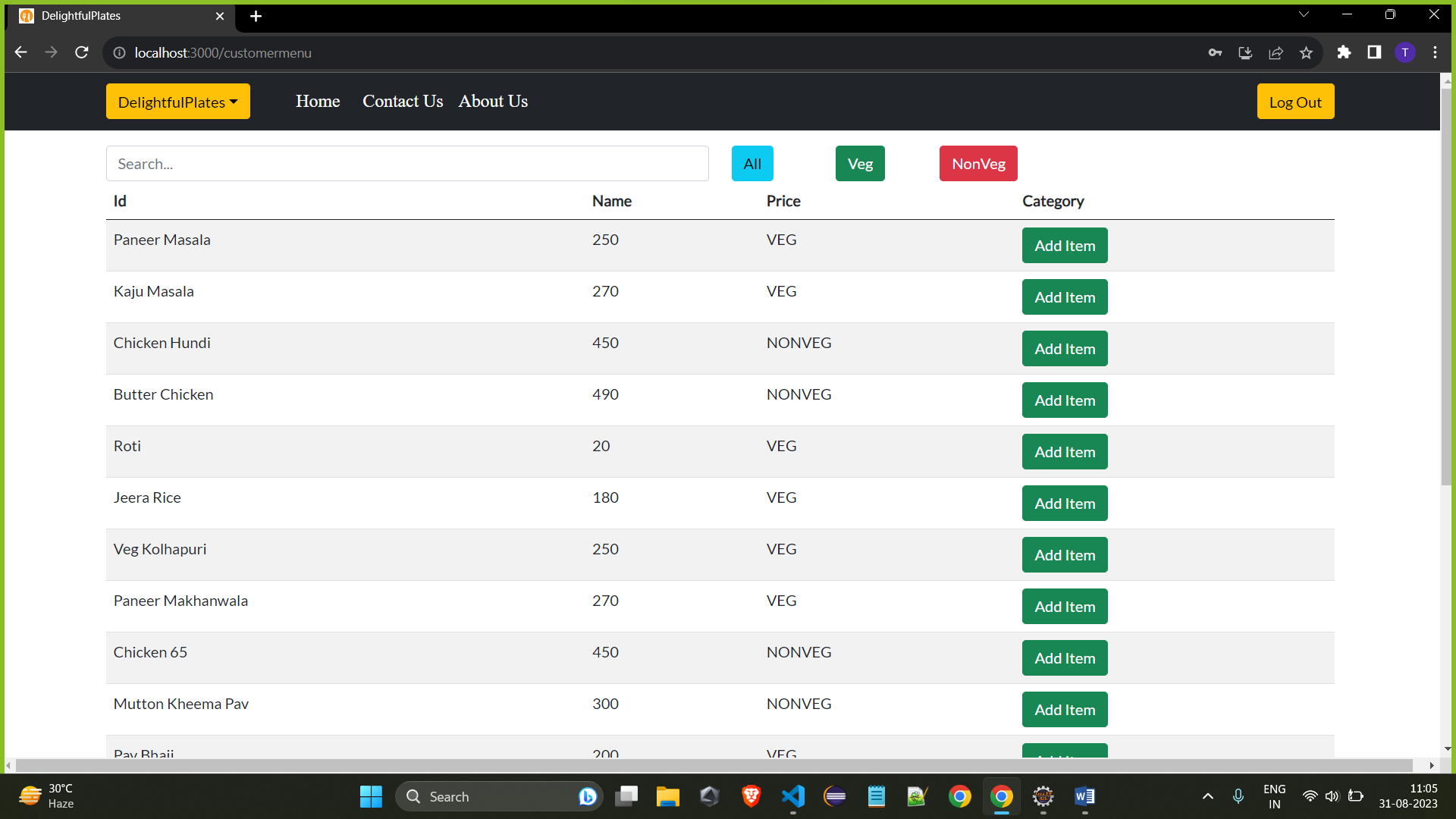
4: Table Booking



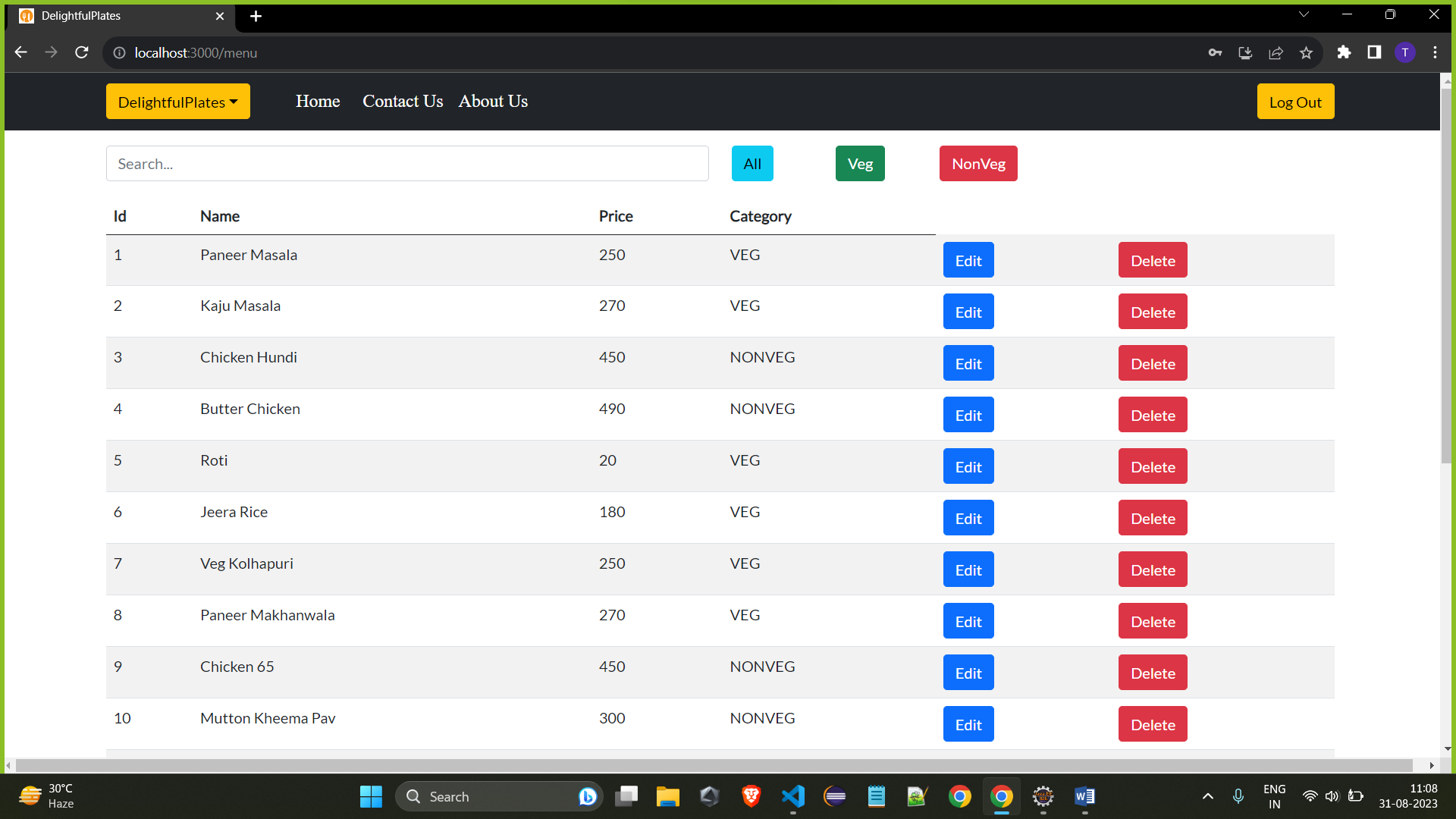
5: Previous Orders



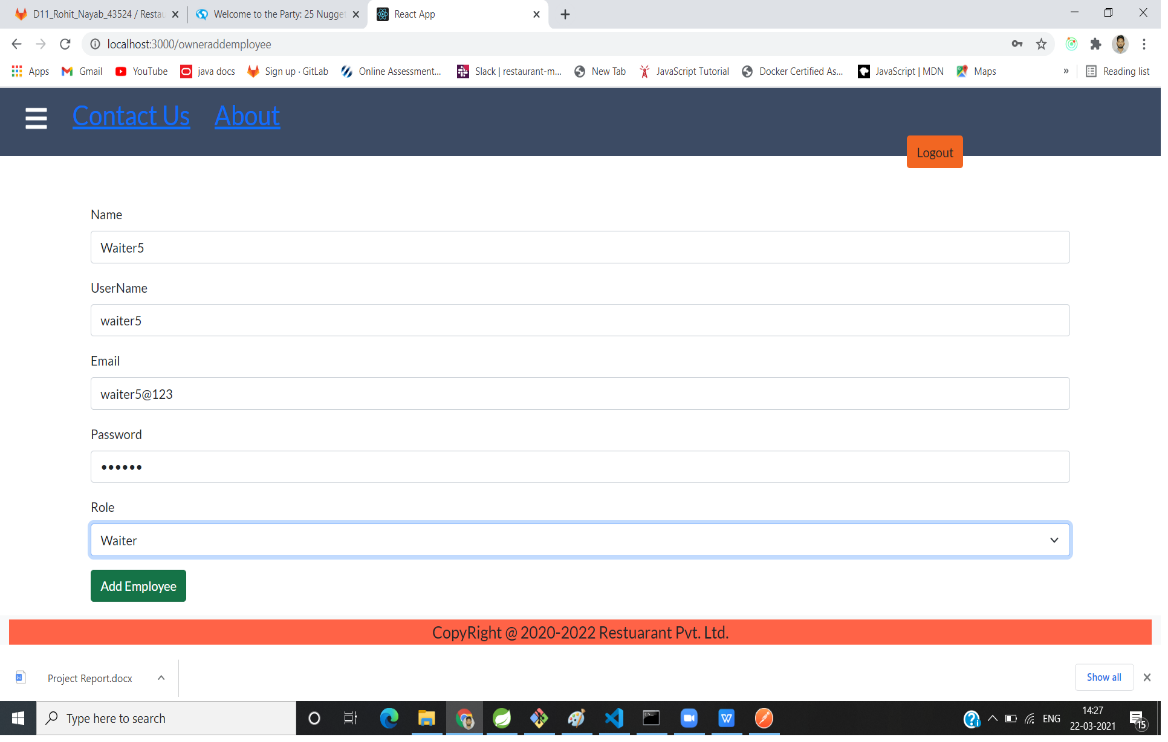
6: Customer side menu



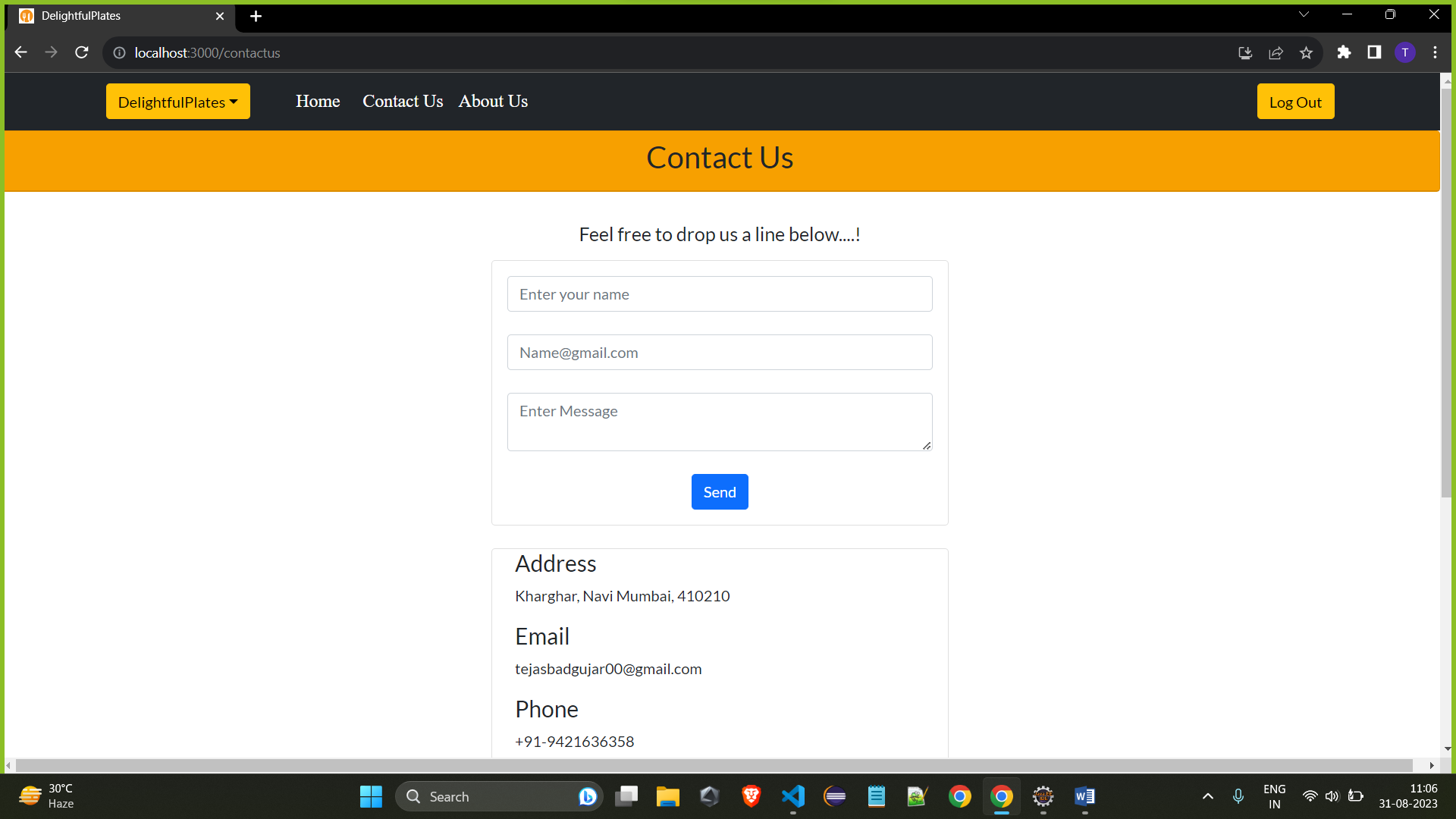
7: Restaurant side menu



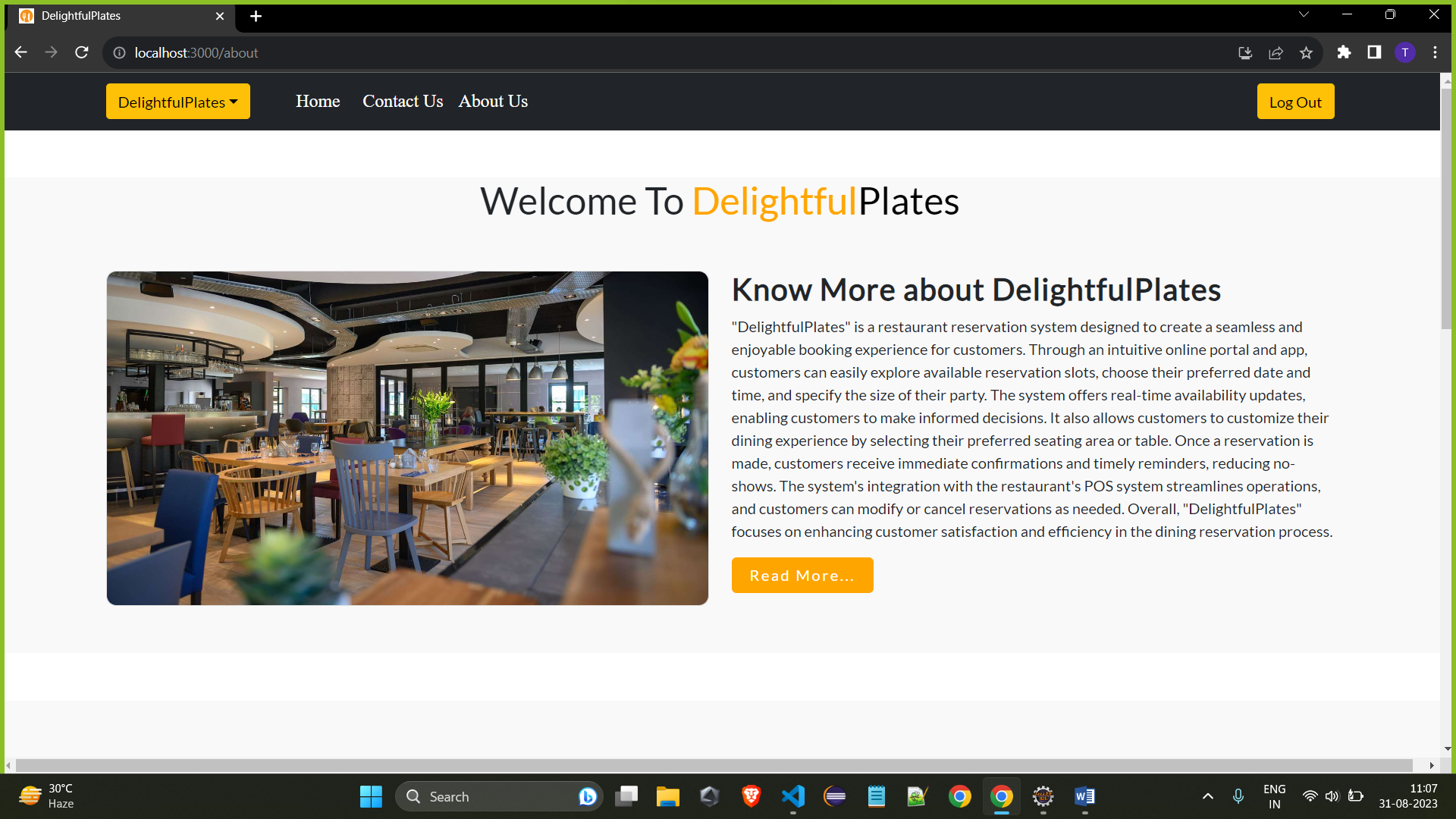
8: Add employee



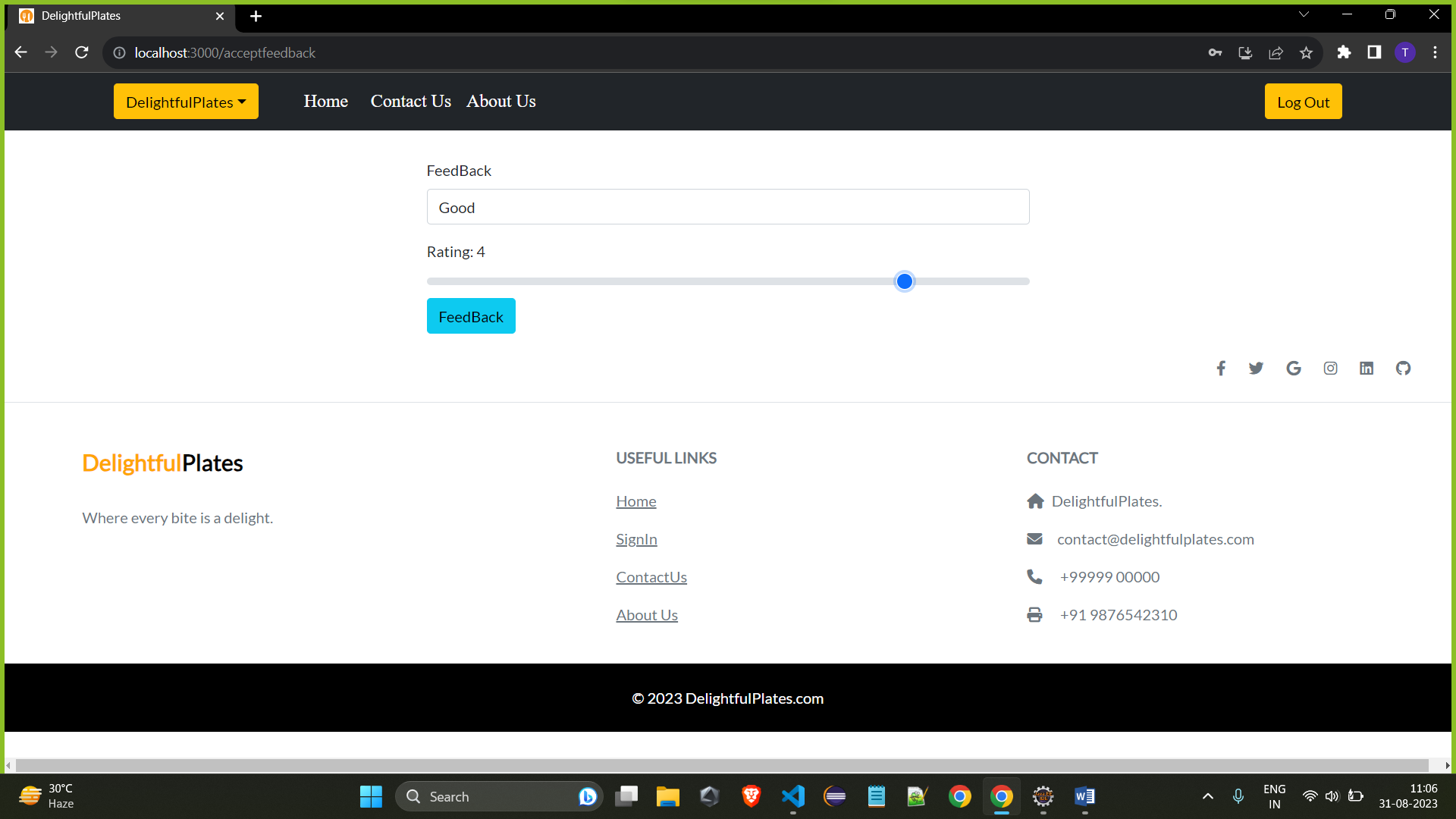
9: Contact Us



10: About Us



11: Feedback



**7: REFERENCES**

<http://www.google.com>

http://www.webdevelopersjournal.com/

http://www.w3.org

<http://www.wikipedia.org>

<http://reactjs.org>

<http://getbootstrap.com>