Software Requirement Specification (ABC Restaurant)

Student reg. no. – st20315315

Table of Contents

Introduction

This report depicts the RESTful API developed for ABC Restaurant chain. It is designed to streamline the business activities of ABC restaurant with a seamless manner. The system is developed with the assistance of various languages and frameworks. For the front-end, Angular was used and for the back-end C# was used. As the database MySQL was chosen. This architecture is supporting various functionalities such as user management, restaurant management, service management, payment management, query management and much more.

This report wishes to cover the entire development process, initial steps that were taken to design and implement the system, the testing process etc.

TASK A

System Design and Development

UML Diagrams

Use Case Diagram

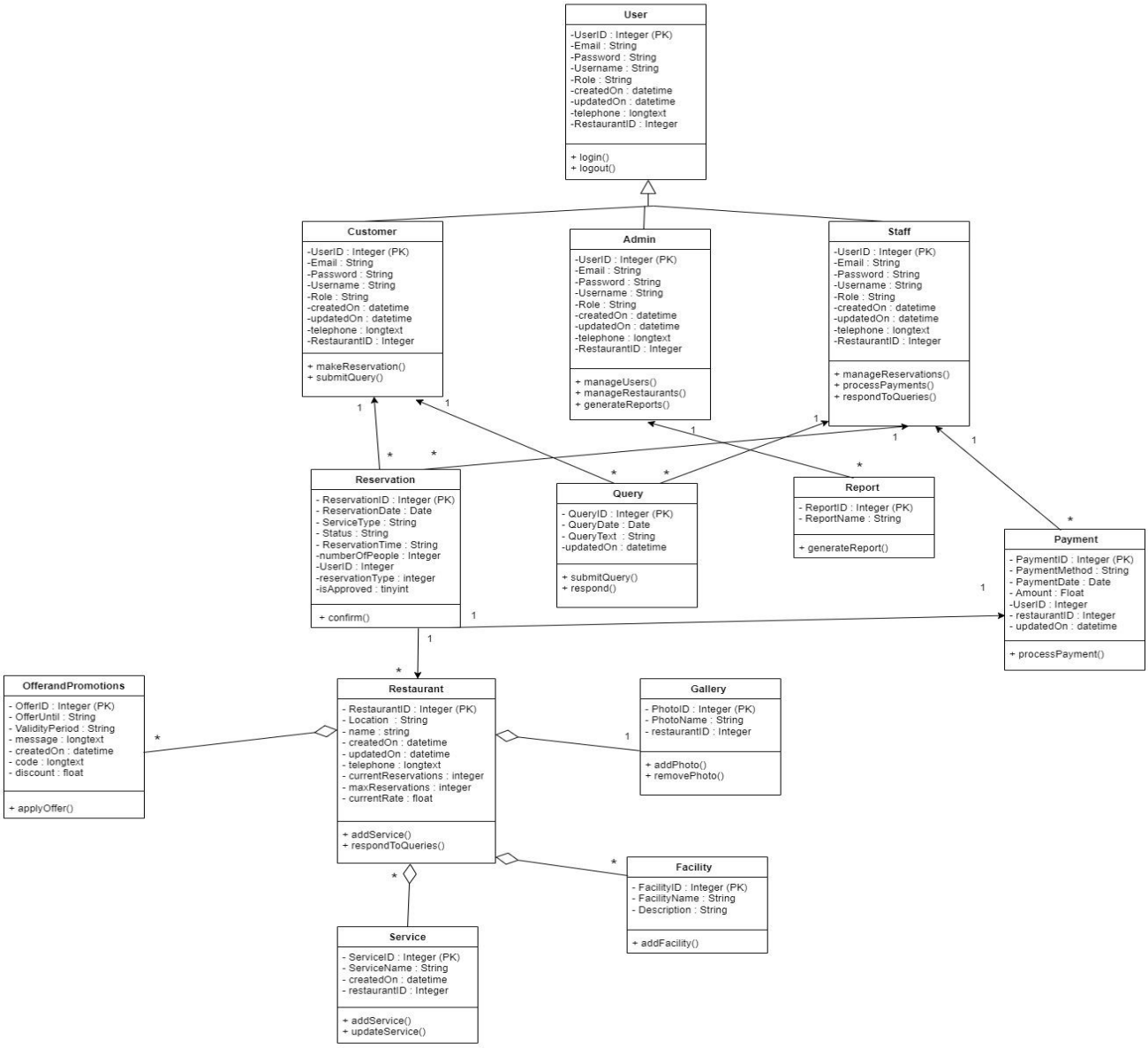
All interactions are functioning in the system according to each role. Admin possess the core management of the system having been able manage users, reservations, restaurants, offers, payments, queries and gallery. Admin is able to generate reports as well.

Staff can manage reservations, respond to customer queries, process payments and generate some of the reports as well. Staff has the responsibility to make sure that the emails are sent to the customers when they confirm a reservation and their queries are answered.


ABC Restaurant
admin
Manage Users
Manage Services
Generate Reports
Manage Offers
manage restaurants
manage payments
manage queries
manage gallery
manage reservations
Register
Make Reservation
view Services
Submit Query
View offers
customer
view locations
Login
staff
Manage Reservations
<<extend>>
Send Confirmation Email
Extension Point:
ReservationConfirmed.
Respond to Queries
<<extend>>
Send Confirmation Email
Extension Point:
ResponseReceived.
Process Payments
generate reports

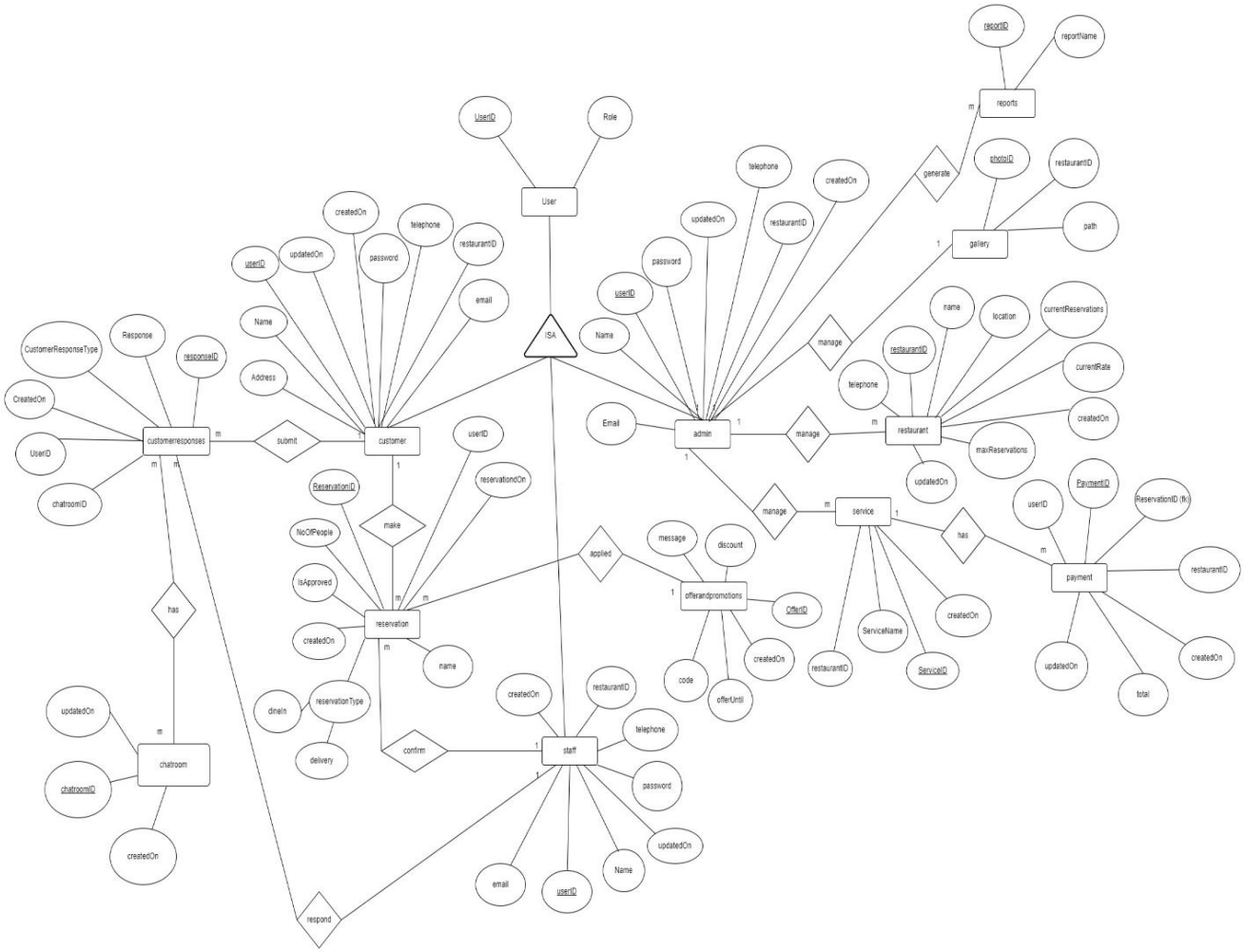
Class Diagram

User class acts as the parent class for the other three types of users. These user types are illustrated in the above diagram using inheritance. The relationships connected to restaurant through gallery, facility, service and offerandpromotions are depicted using aggregation and directed association has been used to depict all the other relationships in this diagram. Reservation class takes details about reservations such as noOfPeople, reservationDate etc. while query class is handling customer queries. Payments class manages payments and the report class is handling the report generation. Restaurant class has details about restaurants in the chain while gallery, offerandpromotion, service and facility classes are handling the additional features.



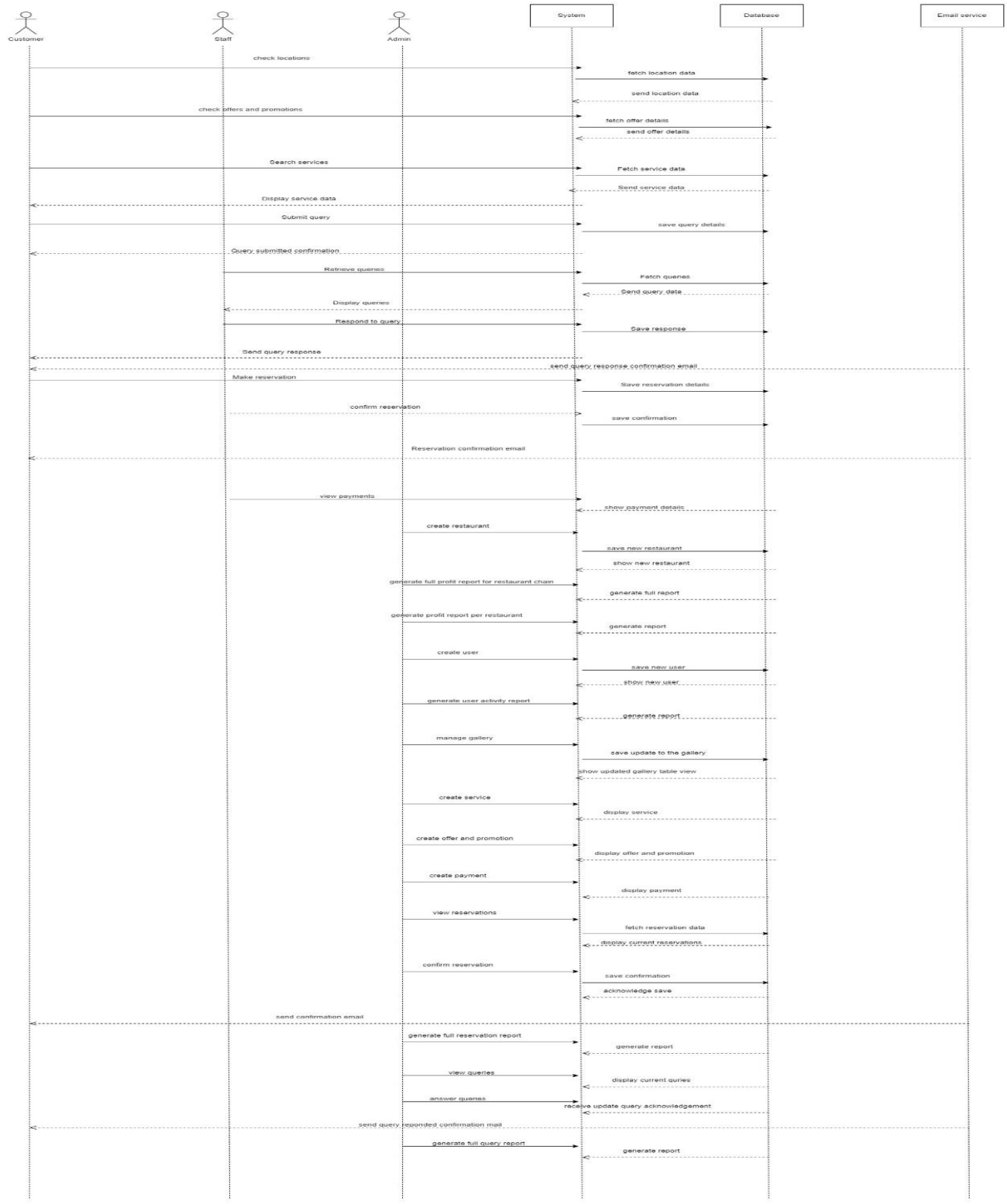
ER Diagram

All the three users are illustrated using a ISA relationship. User is a generalized entity which has inheritance as well where admin, staff and customer are specializations. The customer can make many reservations and the system includes the reservationtype entity that includes dine-in or delivery options. The reservation is connected with staff where staff can control payments and customerresponses. Admin manages the restaurant entity which is associated with offers, services and offerandpromotions etc. Admin manages the gallery. The system comes with chartroom facility for the customers to submit queries where staff respond to them.



Sequence Diagram

Sequence diagram depicts the flow of the system highlighting the interactions between three types of users with the system, database and the email system. Key interactions such as reservation management, query management, location management, user management, service management etc. are illustrated here. The sequence of requests and their responses are depicted with the information flow of the system.


Customer
Staff
Admin
System
Database
Email service
check locations
fetch location data
send location data
check offers and promotions
fetch offer details
send offer details
Search services
Fetch service data
Send service data
Display service data
Submit query
save query details
Query submitted confirmation
Retrieve queries
Fetch queries
Send query data
Display queries
Respond to query
Save response
Send query response
send query response confirmation email
Make reservation
Save reservation details
confirm reservation
save confirmation
Reservation confirmation email
view payments
show payment details
create restaurant
save new restaurant
show new restaurant
generate full profit report for restaurant chain
generate full raport
generate profit report per restaurant
generate report
create user
save new user
show new user
generate user activity report
generate report
manage gallery
save update to the gallery
show updated gallery table view
create service
display service
create offer and promotion
display offer and promotion
create payment
display payment
view reservations
fetch reservation data
display current reservations
confirm reservation
save confirmation
acknowledge save
send confirmation email
generate full reservation report
generate report
view queries
display current quries
answer queries
receive update query acknowledgement
send query reponded confirmation mail
generate full query report
generate report

Assumptions

- Admin and staff can create reservations for customers.
- Admin can enter rate per person for each restaurant
- Admin can enter a maximum number of reservations per restaurant. When the amount is exceeded, customers no longer can place reservations for the particular restaurant. Once the admin sets the max no. of reservations, the make reservation button gets enabled.
- Database has a separate table for reservations so that we can make a many-to-many relationship with restaurants.
- The chatrooms table was created so that we can take every query and make them a single question.
- Only the customers can register themselves. Staff and other admin users are created by admin.

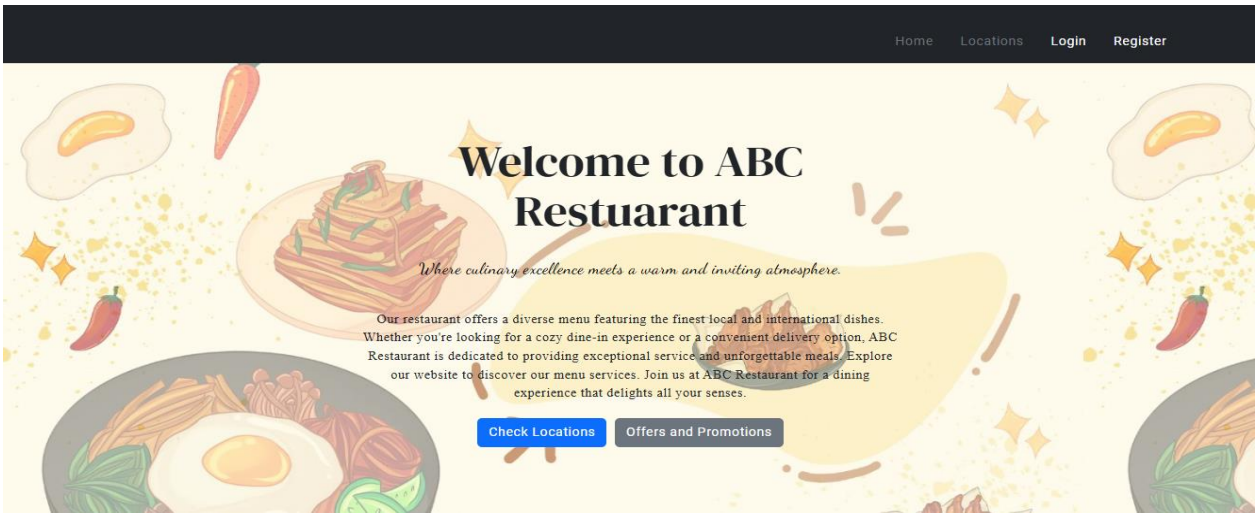
TASK B

Development of the System

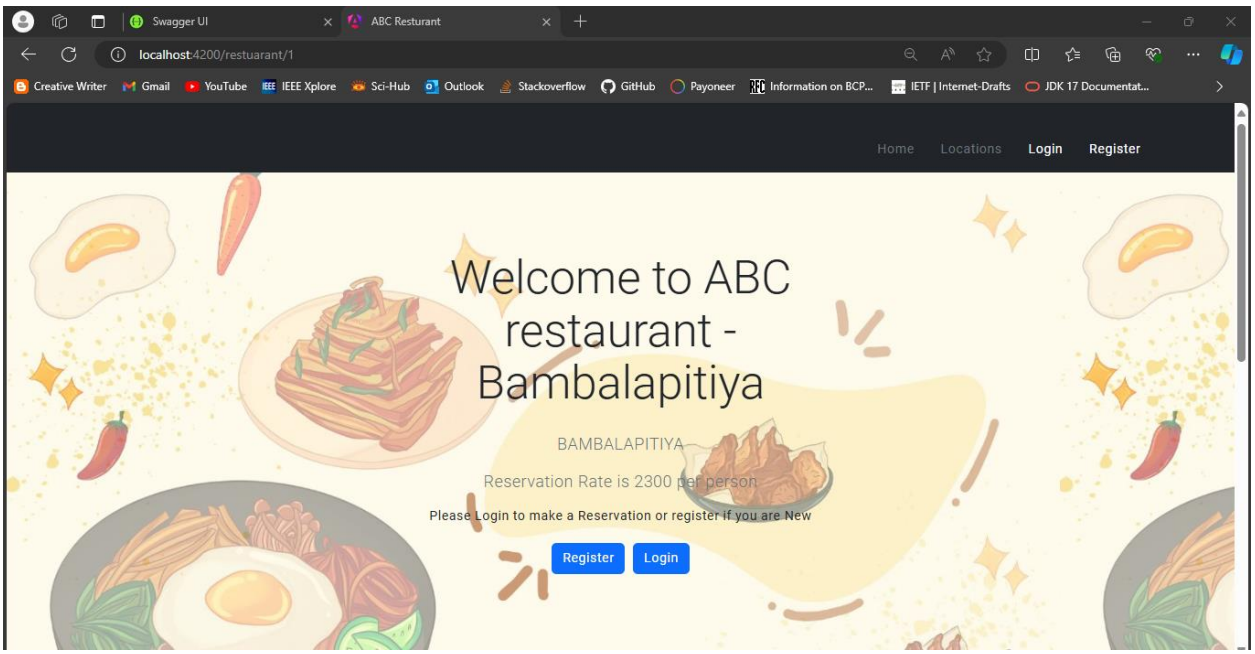
In the development process, C# and .NET 8 was used for the back-end while Angular 18 was used for the front-end. QuestPDF was the package that was used in the back-end to help with report generation and Bootstrap, Angular material and PrimeNG are the packages that were used in the front-end. Papercut SMTP desktop email server was used to send emails. Visual Studio code was used for the server-side development. Visual Studio Code was used for client-side development (UI). MySQL Workbench was used to implement the database and for query testing.

Set of Interfaces with Their Inputs and Outputs

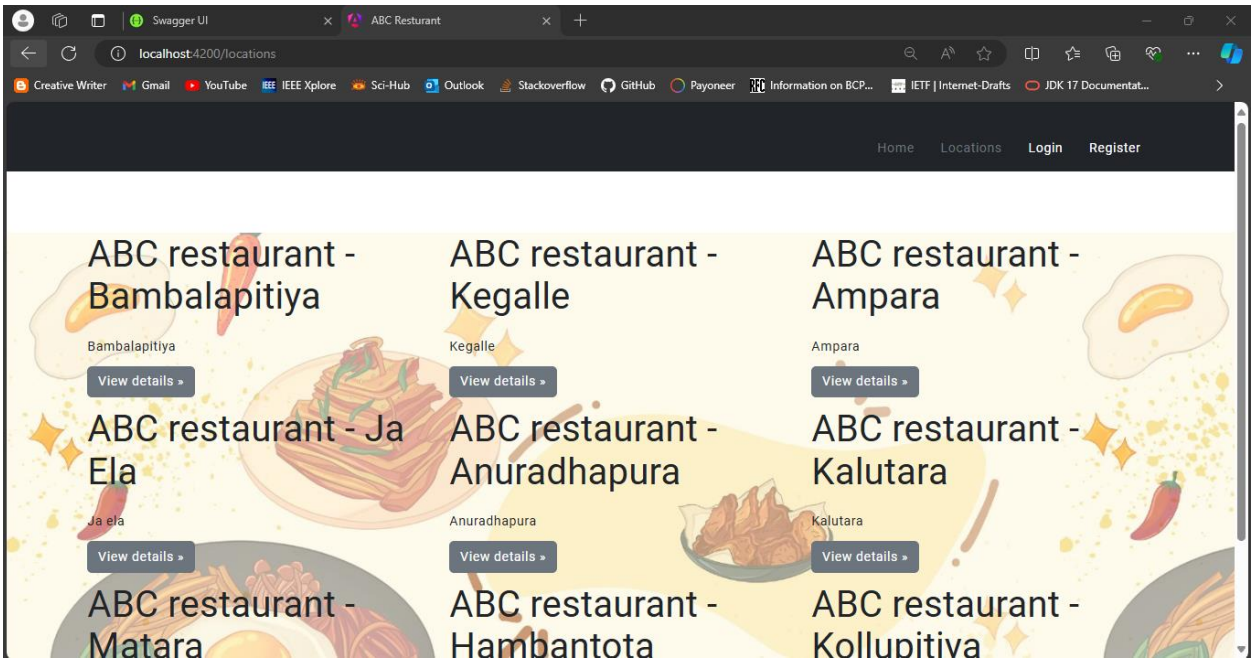
Home page



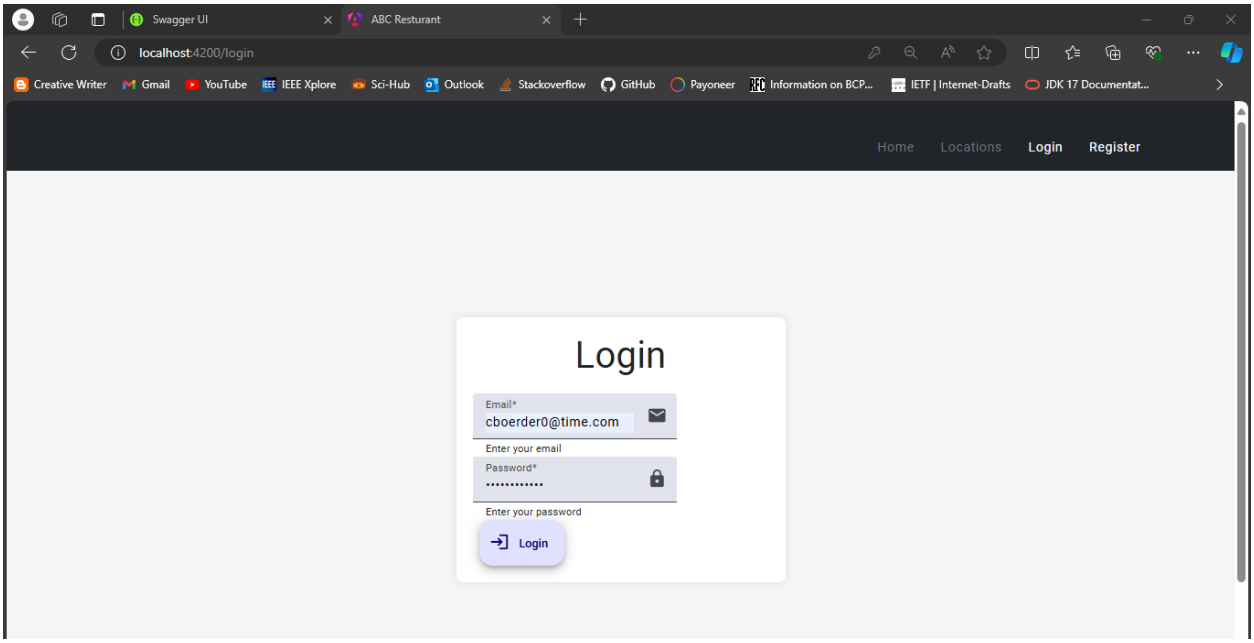
Any user can view Offers and promotions page



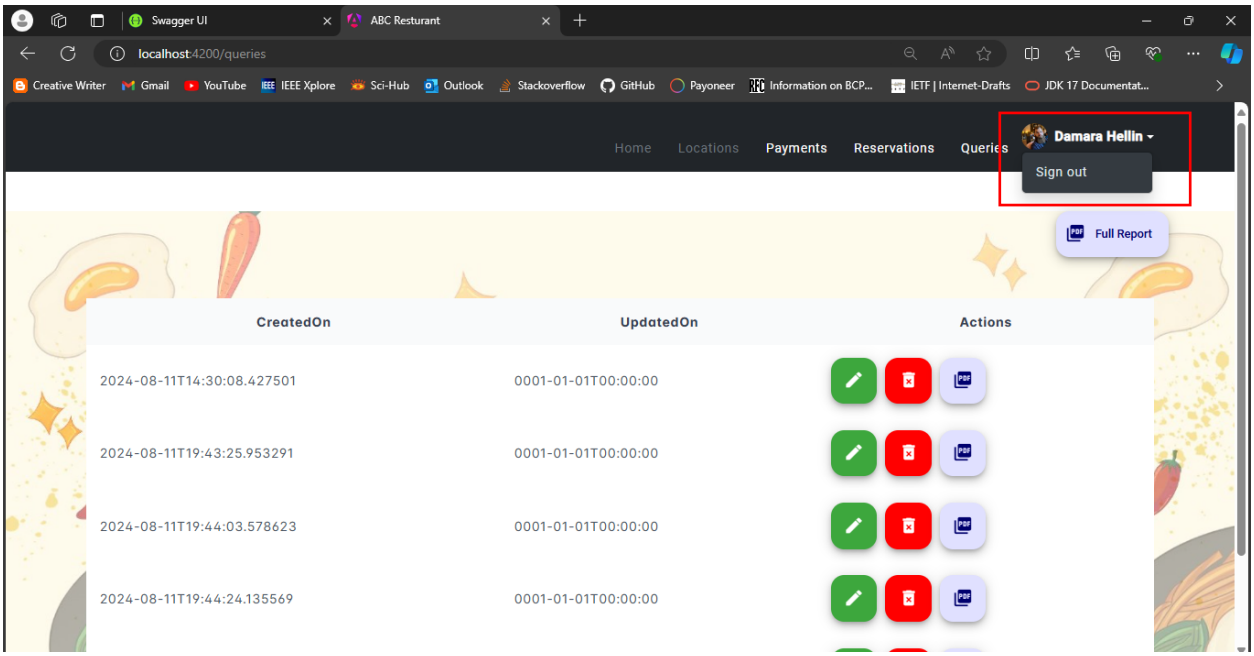
All users can view locations



Login interface for all users

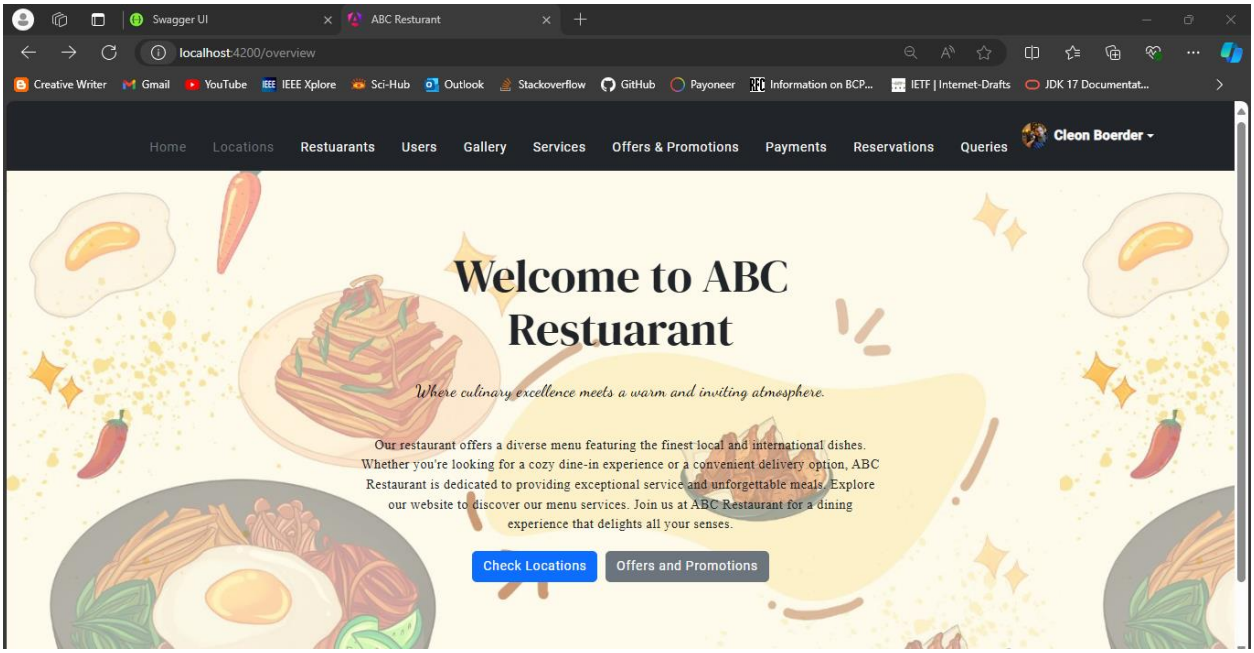


All users can logout by clicking on the profile picture

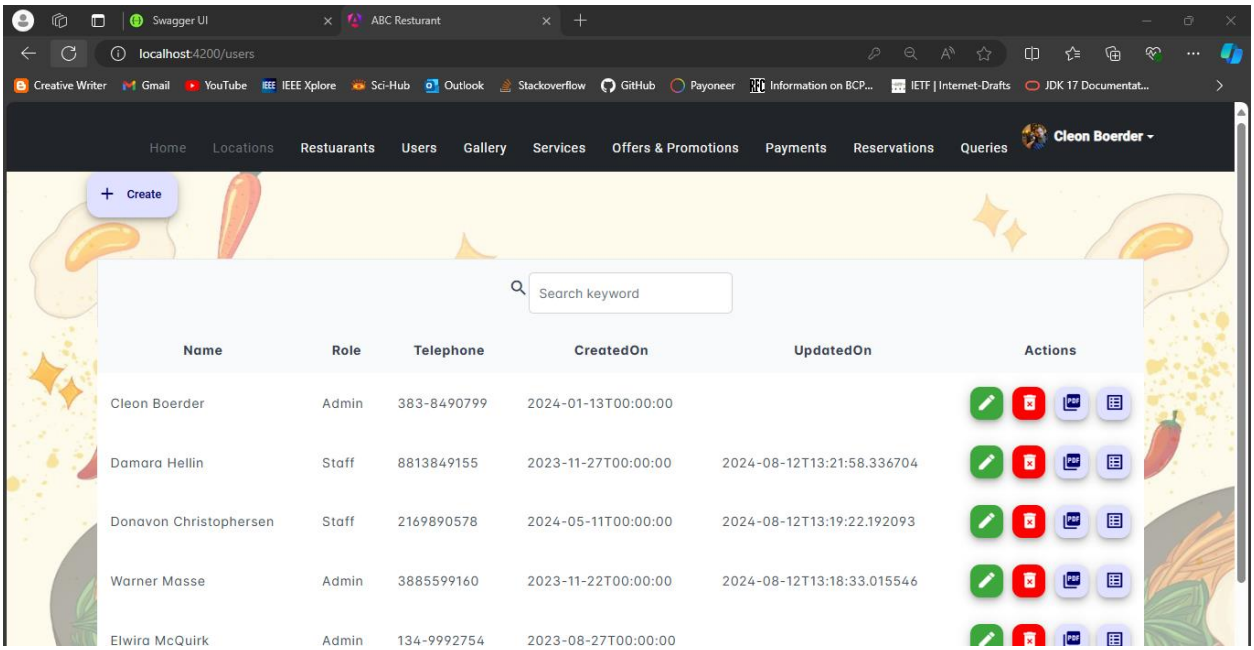


Interfaces for admin

Home page for admin

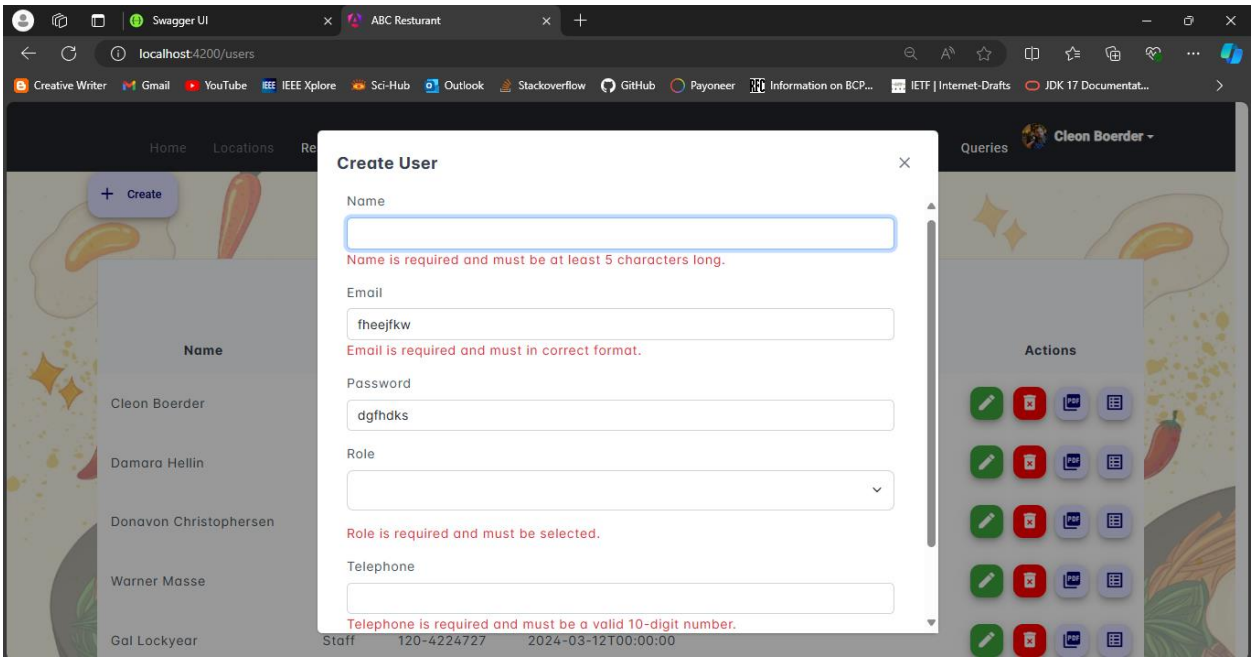


User management

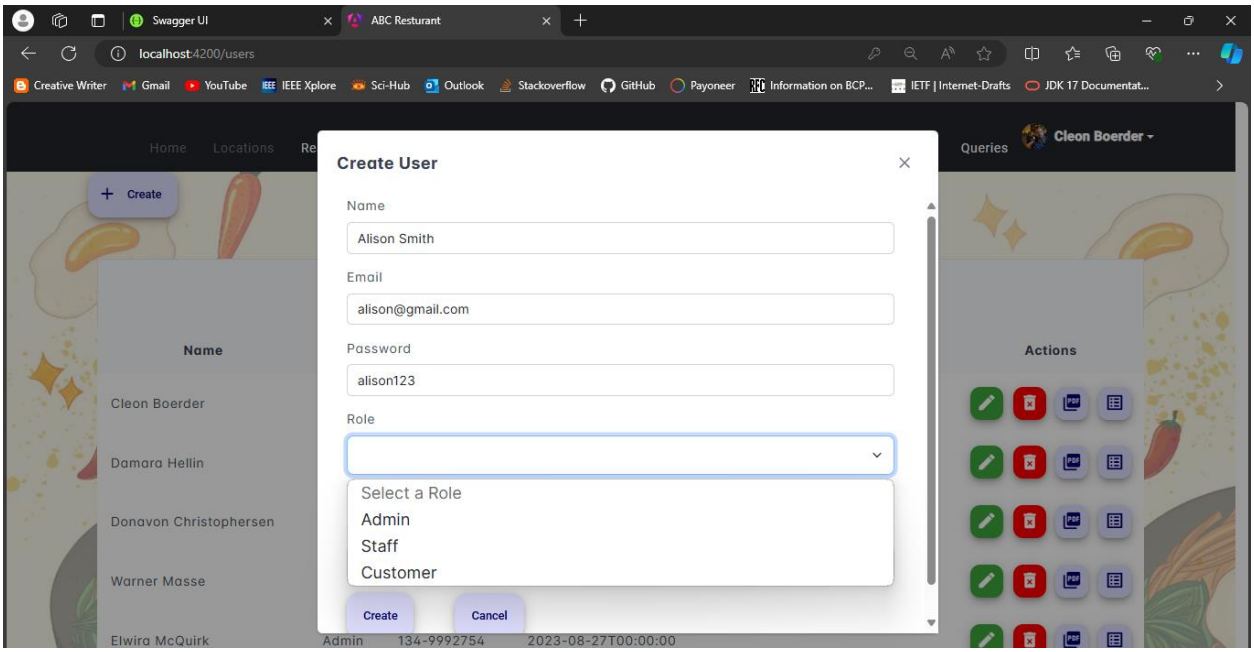


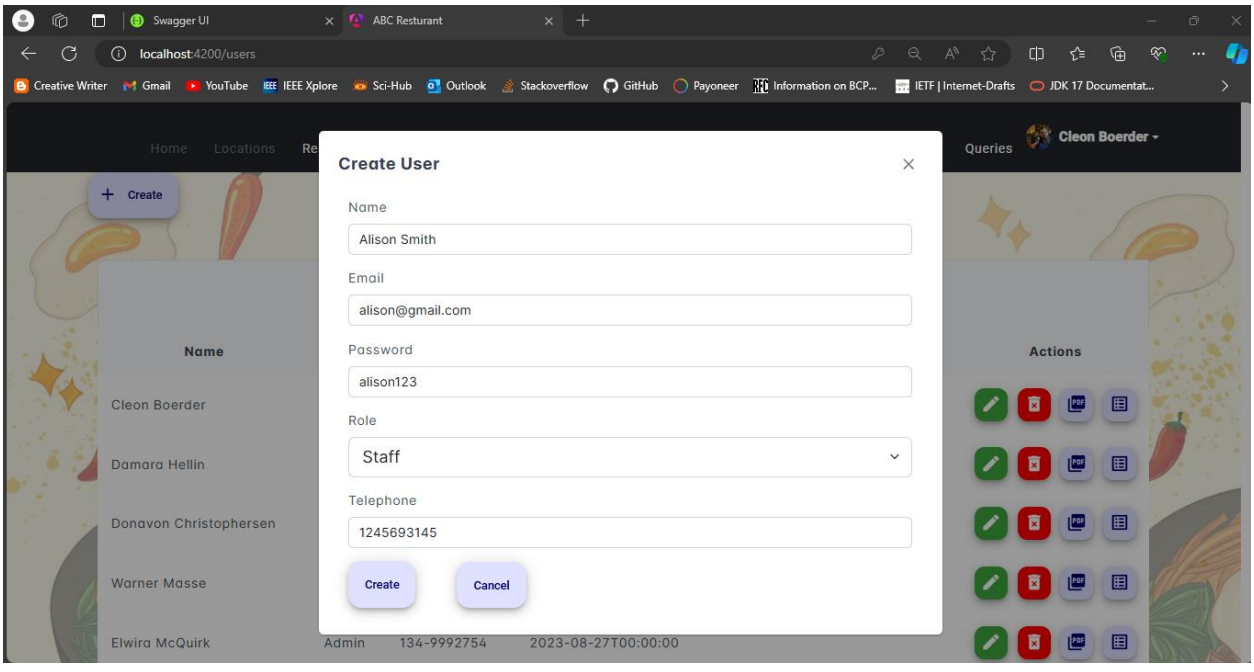
Create user

Form validations

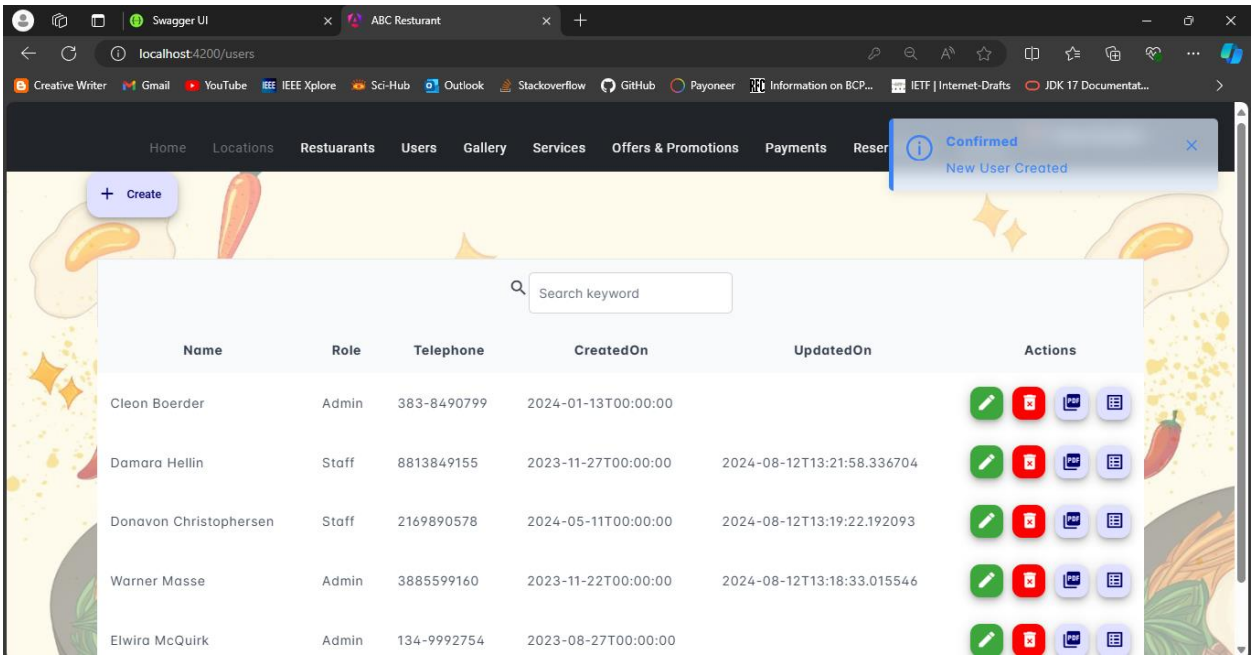


Select role from drop down

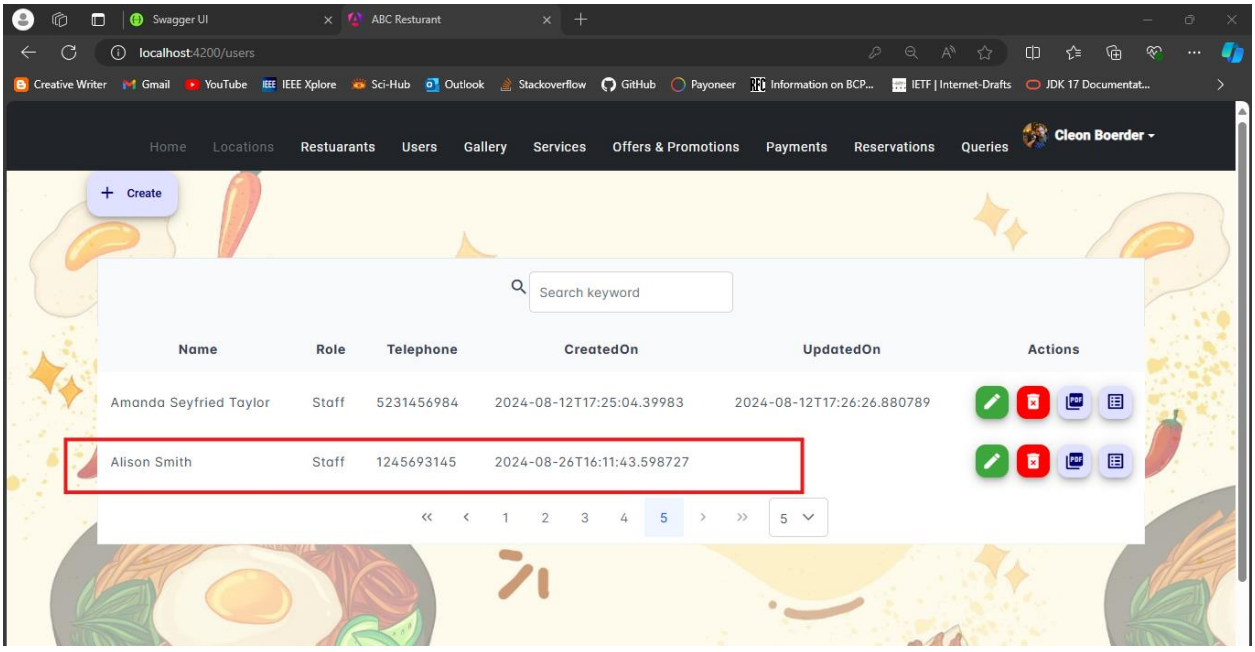




Upon clicking on create button the new user is created and a notification is received to confirm it.

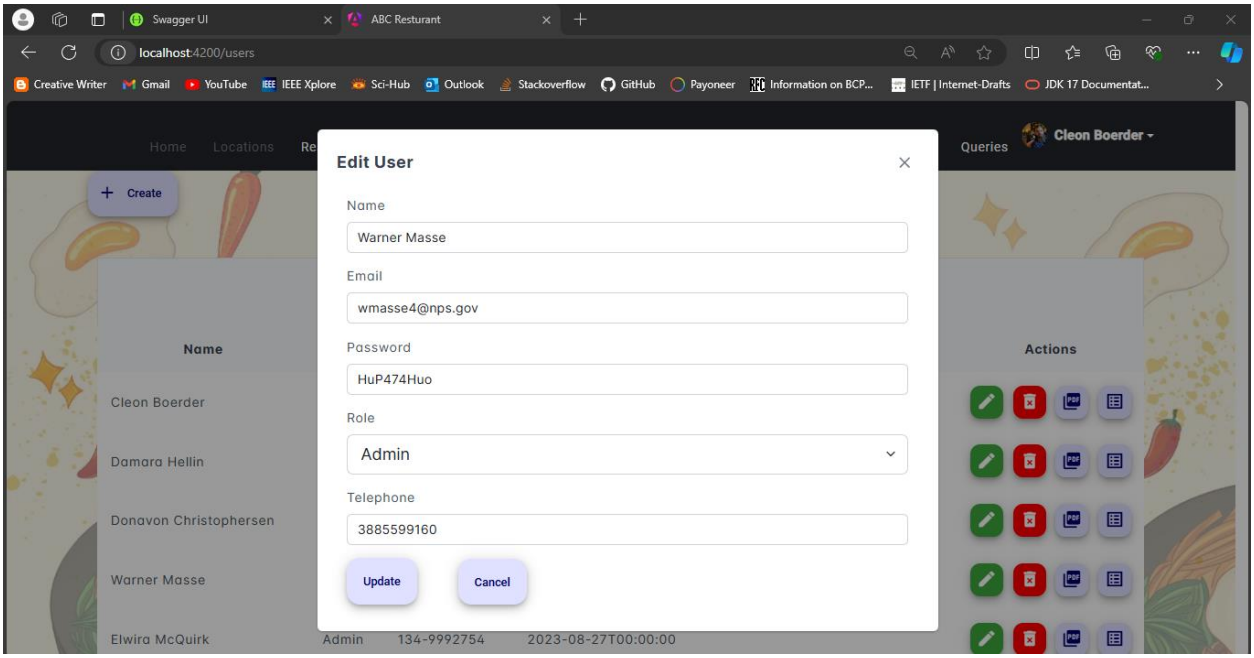


New user is created.

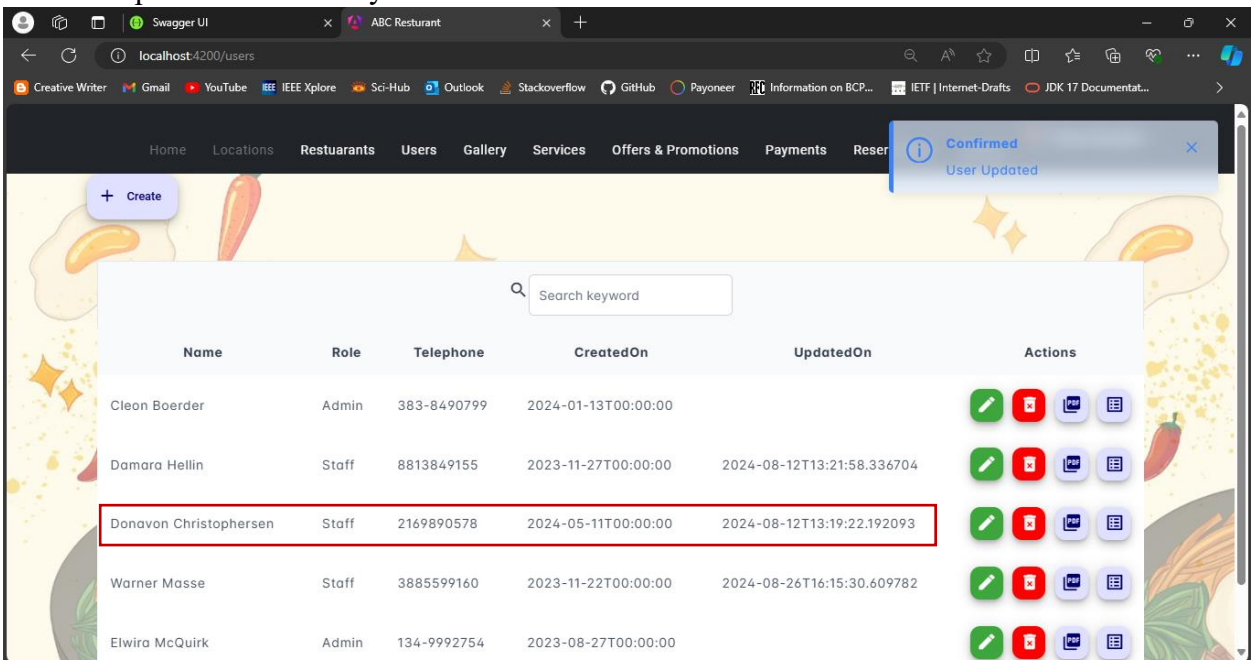


Edit user

Admin can edit all user accounts

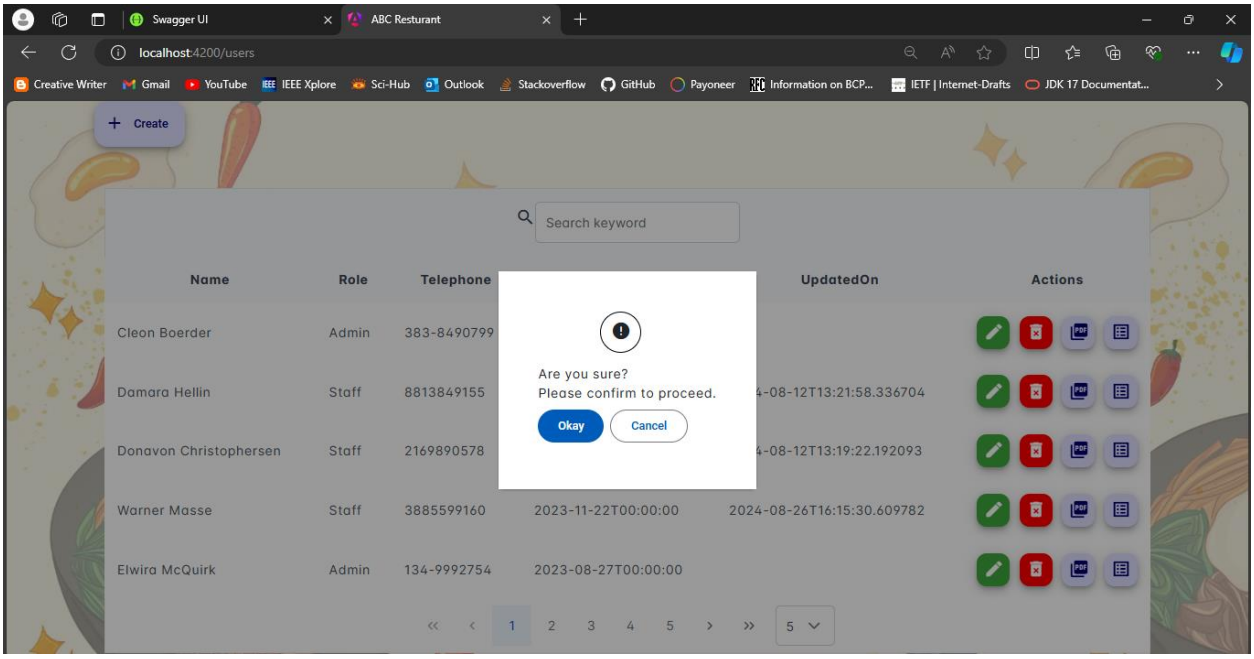


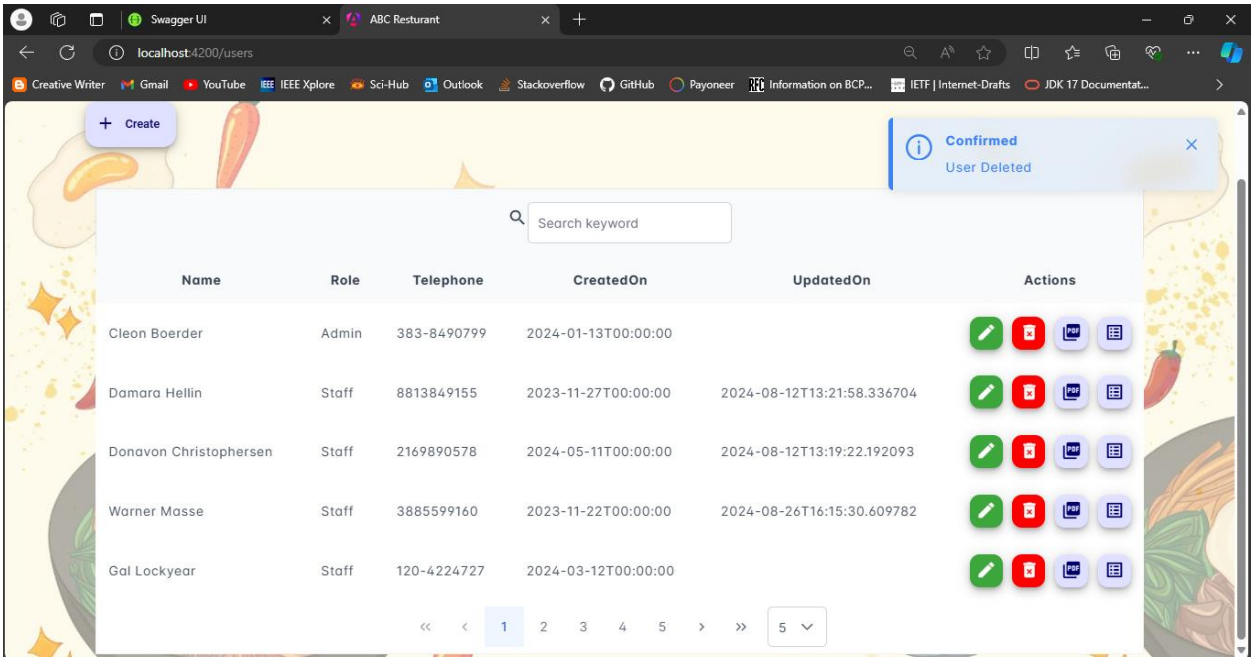
Account updated successfully



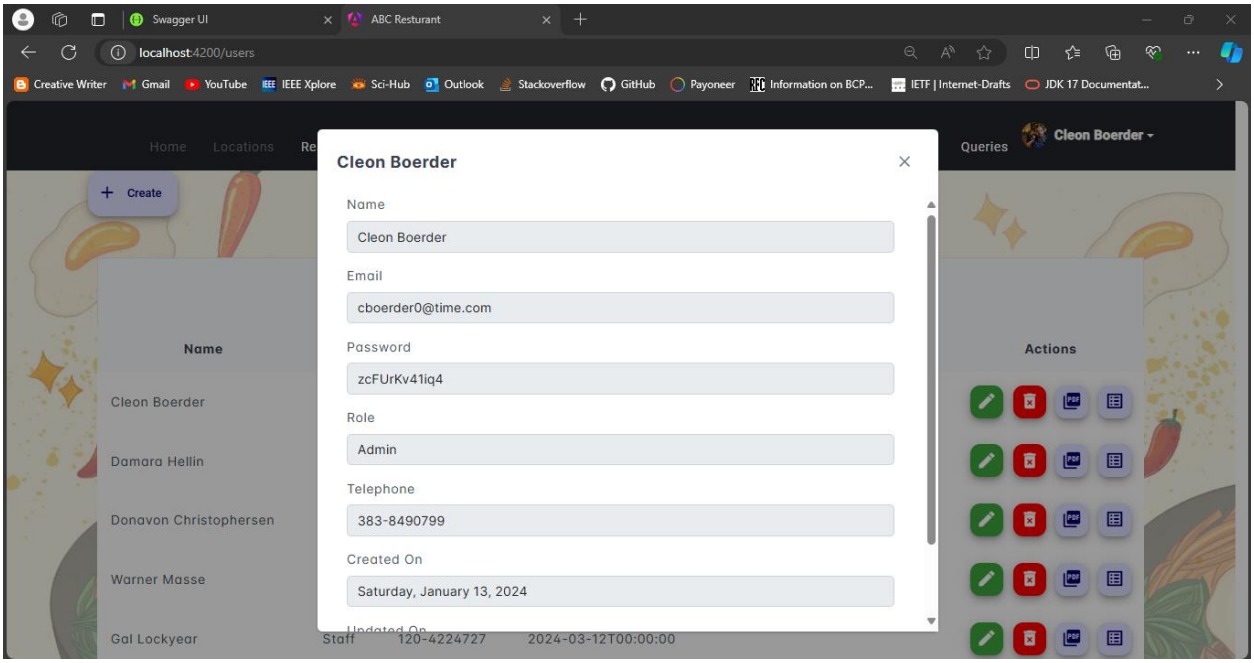
Delete user

To delete a user the delete button should be clicked under actions. And then the following pop up appears and the admin should select okay.

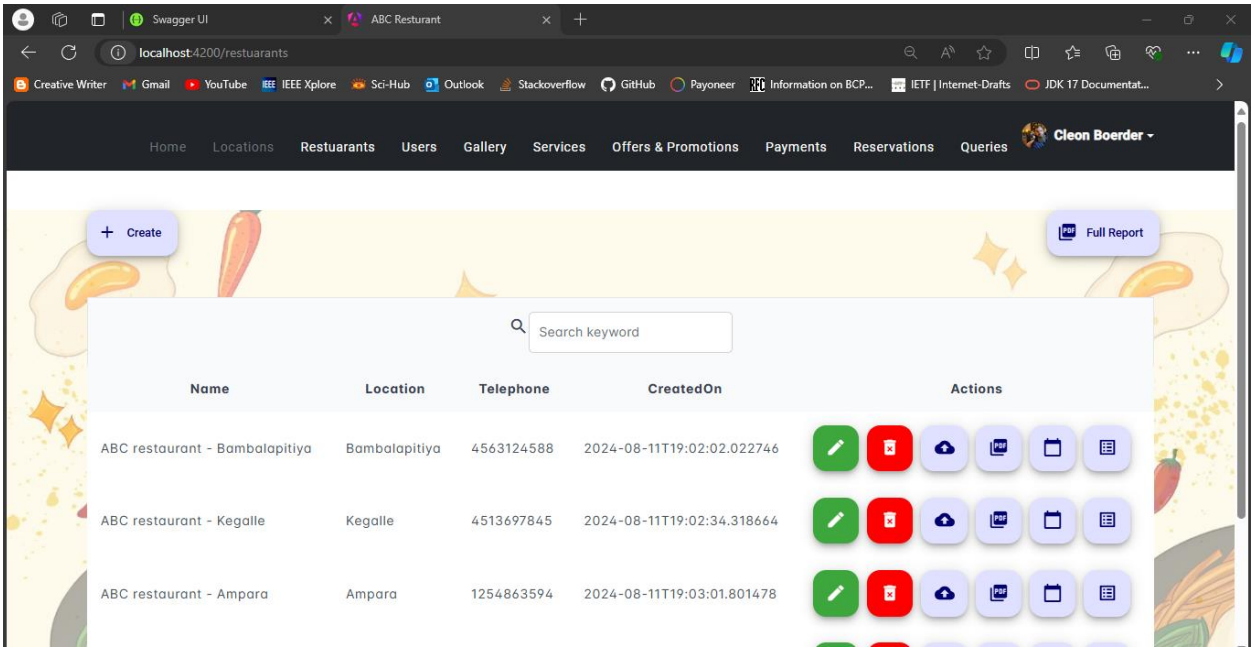




View user

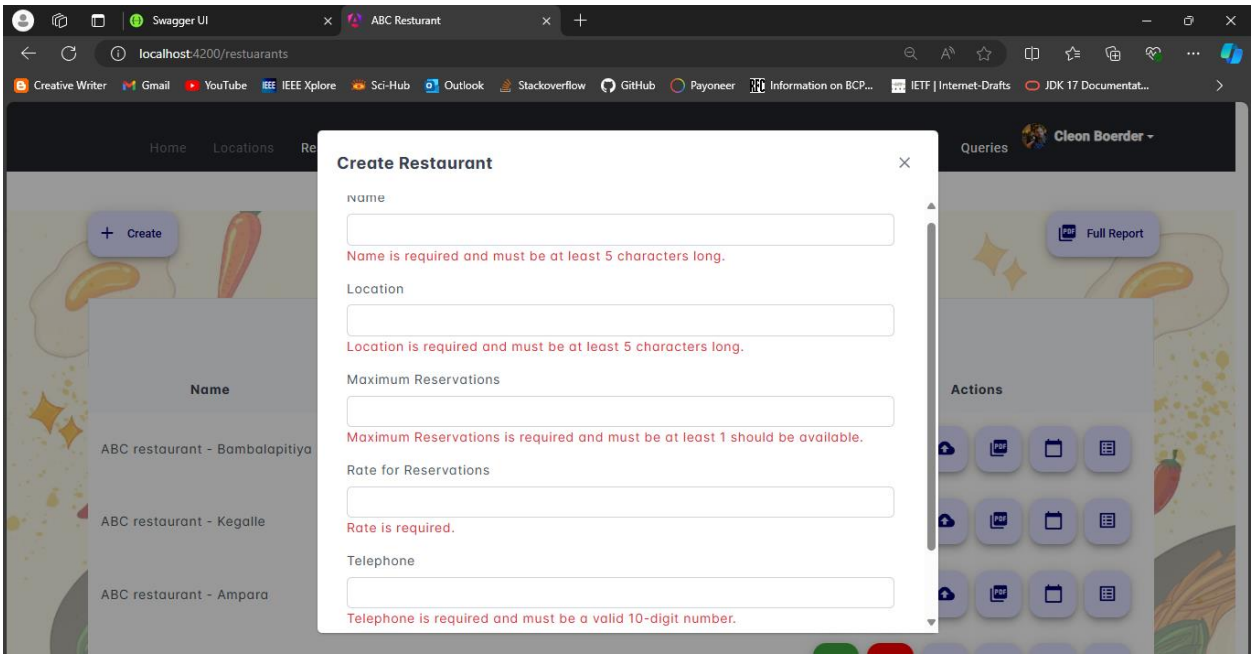


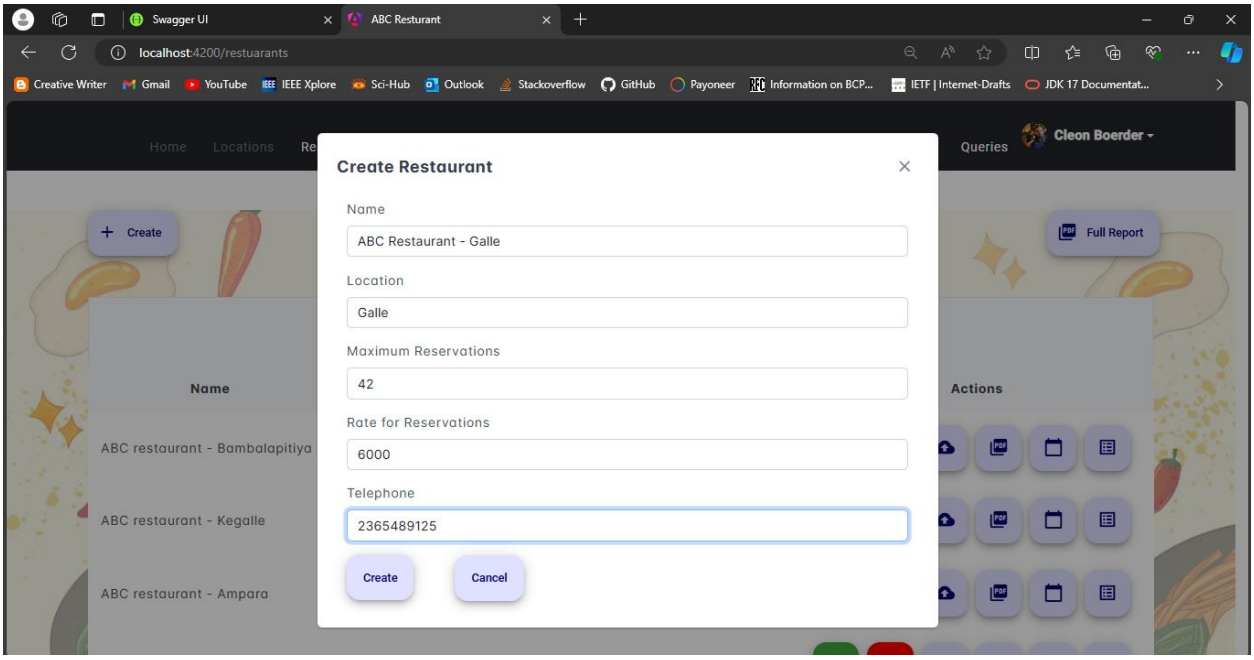
Restaurant management

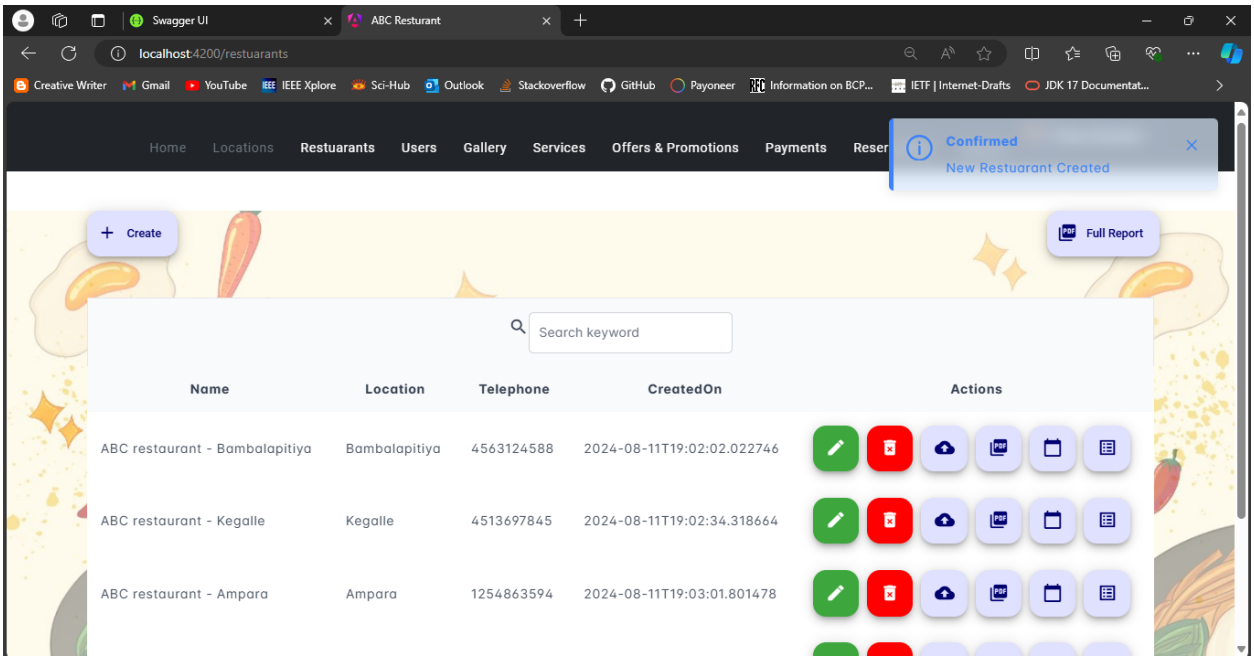


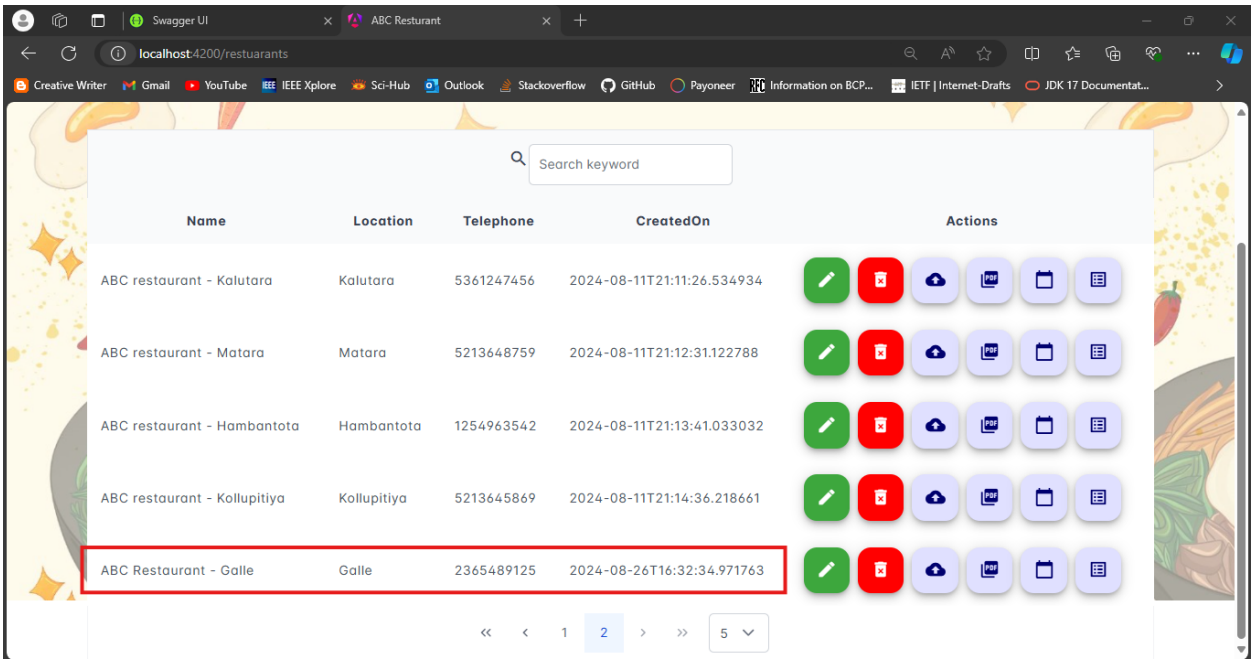
Create restaurant

Form validations

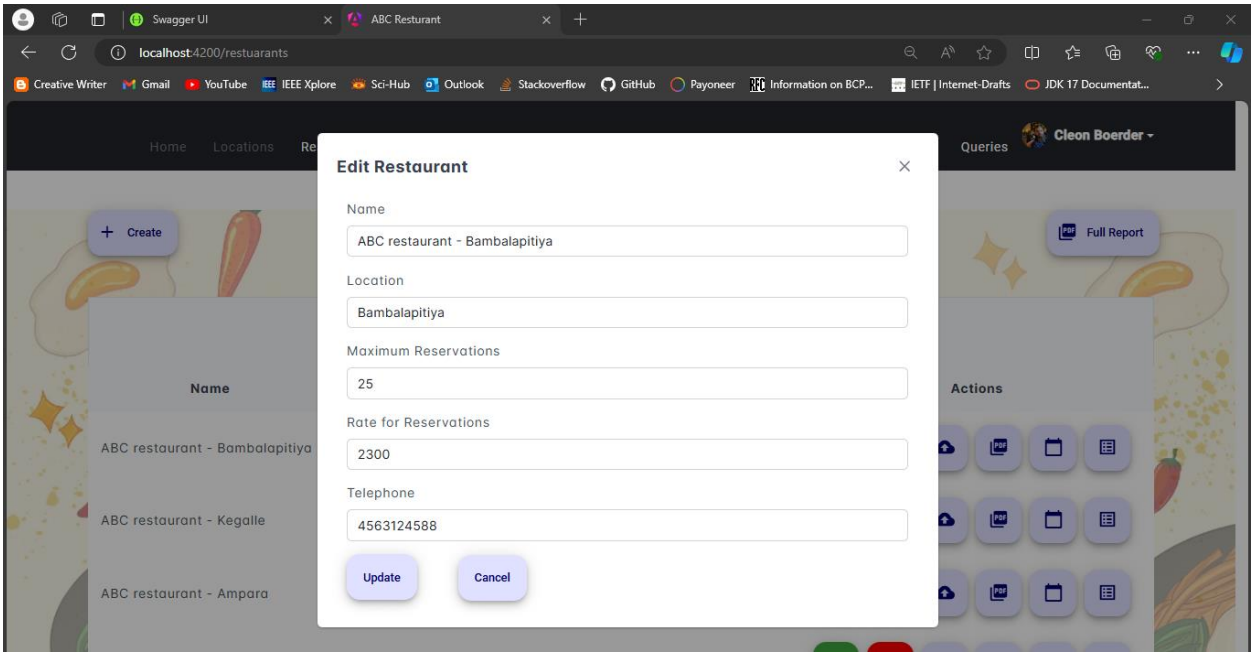


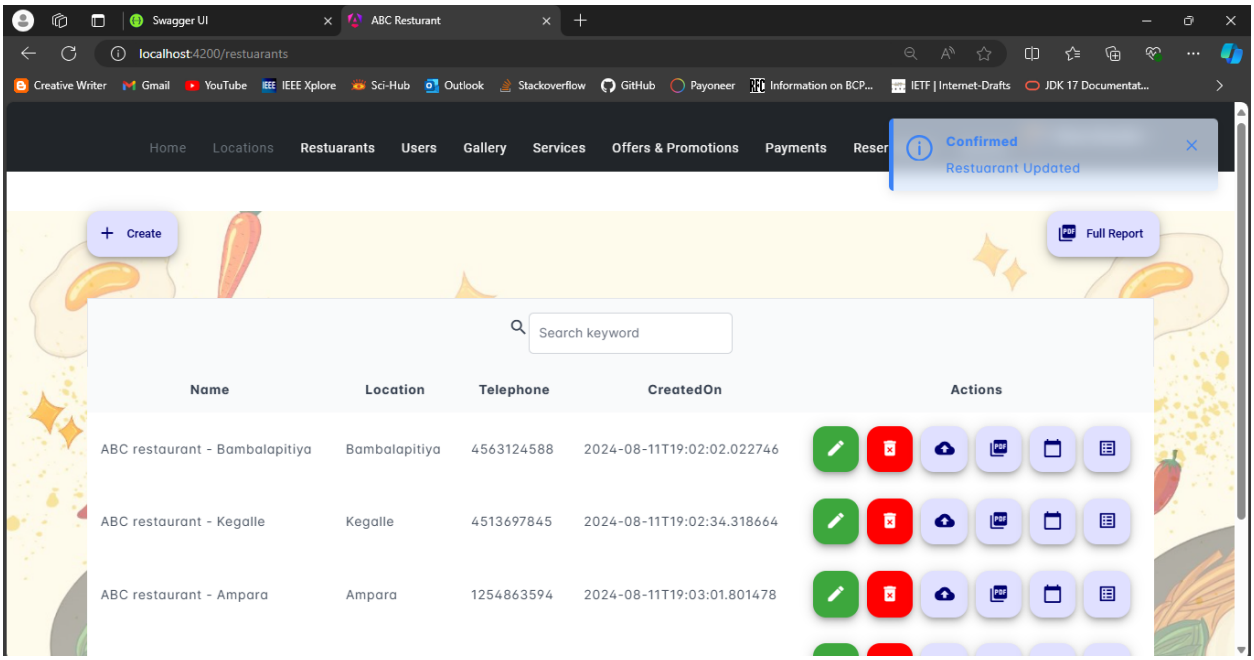



Swagger UI
ABC Resturant
localhost:4200/restuarants
Creative Writer
Gmail
YouTube
IEEE Xplore
Sci-Hub
Outlook
Stackoverflow
GitHub
Payoneer
Information on BCP...
IETF | Internet-Drafts
JDK 17 Documentat...
Home
Locations
Restuarants
Users
Gallery
Services
Offers & Promotions
Payments
Confirmed
New Restuarant Created
Create
Full Report
Search keyword
Name
Location
Telephone
CreatedOn
Actions
ABC restaurant - Bambalapitiya
Bambalapitiya
4563124588
2024-08-11T19:02:02.022746
ABC restaurant - Kegalle
Kegalle
4513697845
2024-08-11T19:02:34.318664
ABC restaurant - Ampara
Ampara
1254863594
2024-08-11T19:03:01.801478

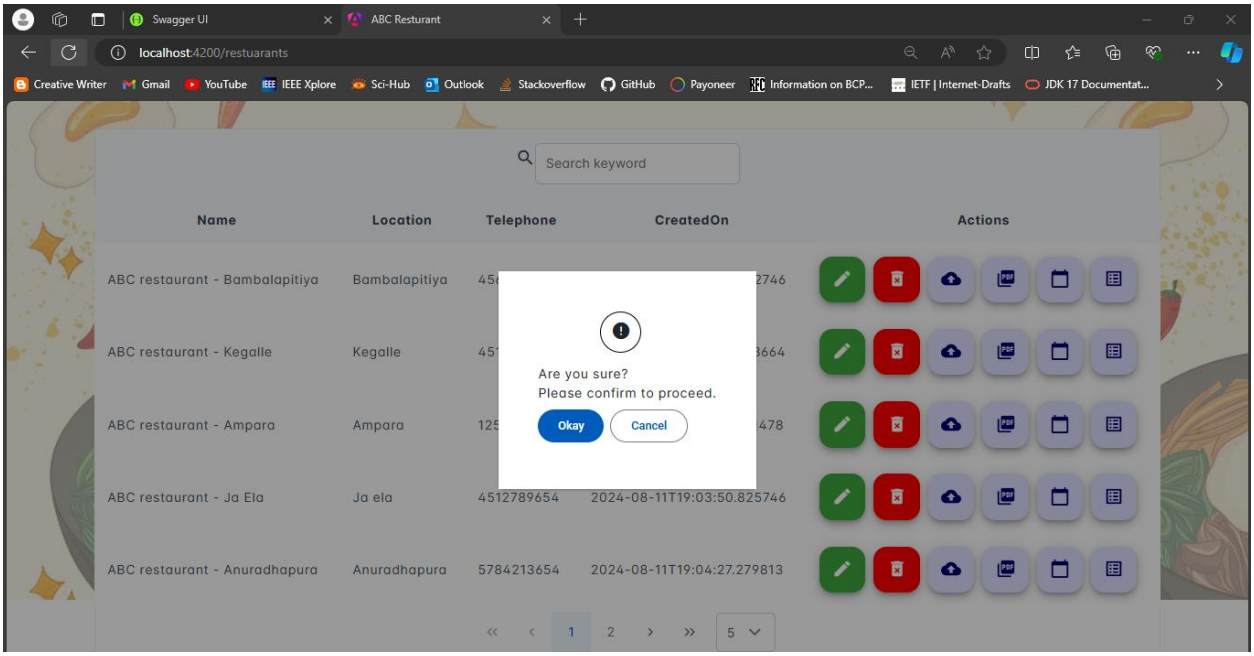

Swagger UI
ABC Resturant
localhost:4200/restuarants
Creative Writer
Gmail
YouTube
IEEE Xplore
Sci-Hub
Outlook
Stackoverflow
GitHub
Payoneer
Information on BCP...
IETF | Internet-Drafts
JDK 17 Documentat...
Search keyword
Name
Location
Telephone
CreatedOn
Actions
ABC restaurant - Kalutara
Kalutara
5361247456
2024-08-11T21:11:26.534934
ABC restaurant - Matara
Matara
5213648759
2024-08-11T21:12:31.122788
ABC restaurant - Hambantota
Hambantota
1254963542
2024-08-11T21:13:41.033032
ABC restaurant - Kollupitiya
Kollupitiya
5213645869
2024-08-11T21:14:36.218661
ABC Restaurant - Galle
Galle
2365489125
2024-08-26T16:32:34.971763
1
2
5

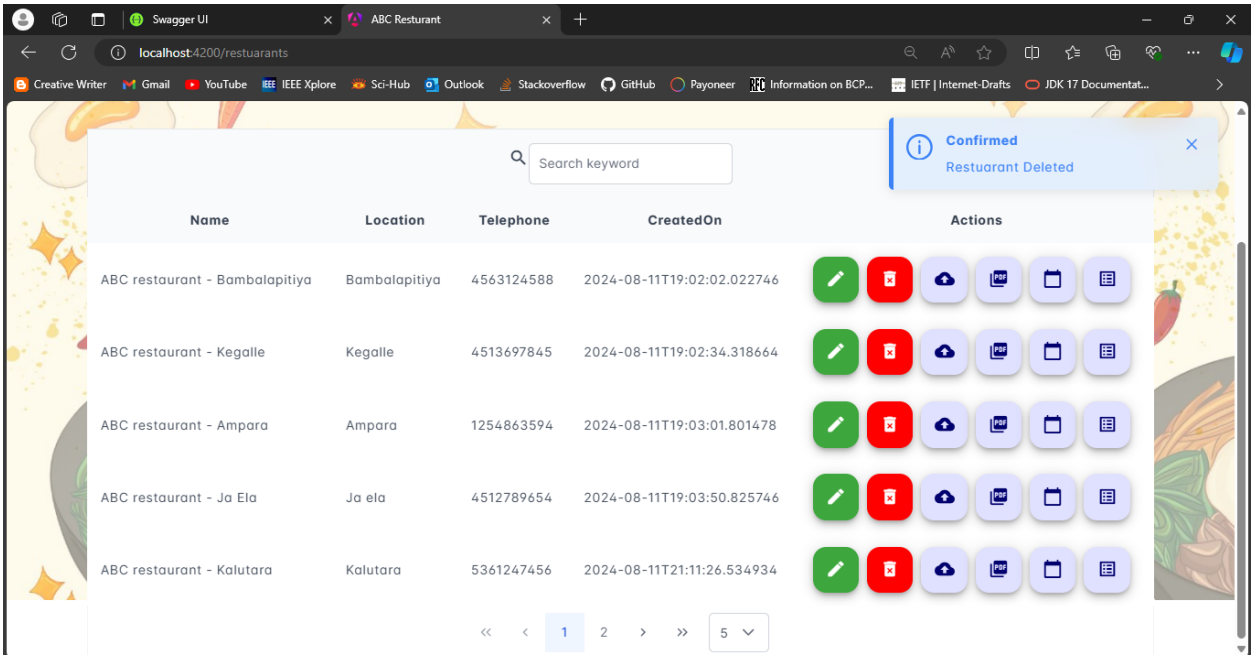
Update restaurant





Delete restaurant

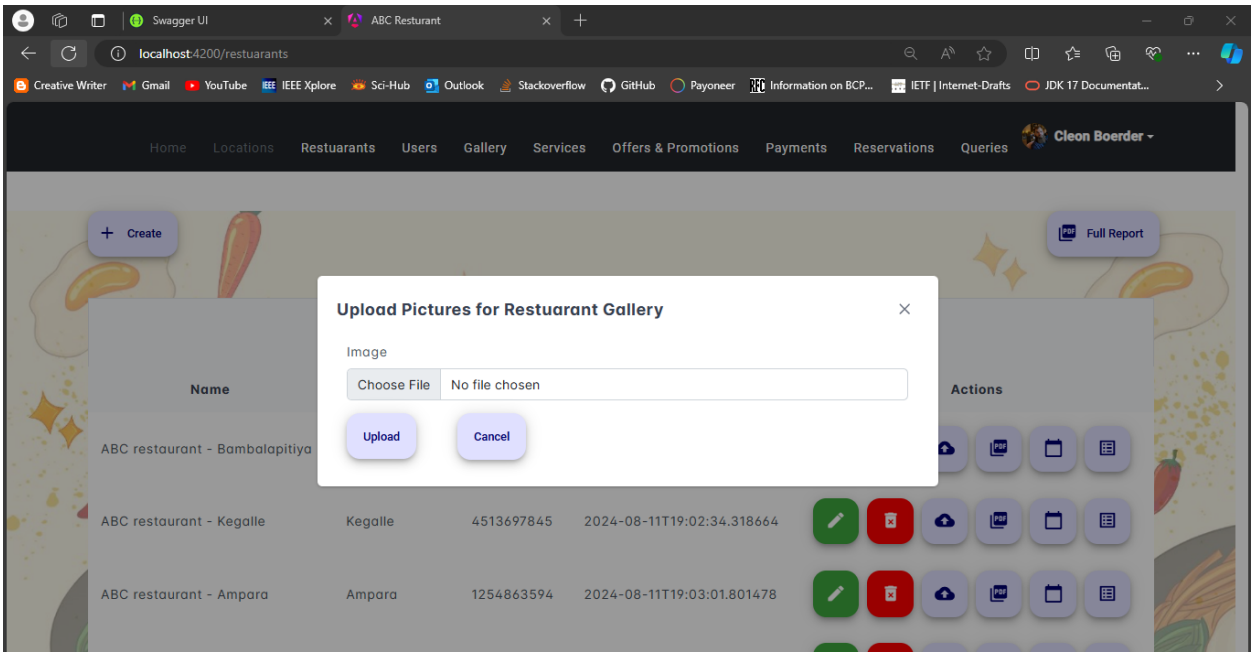




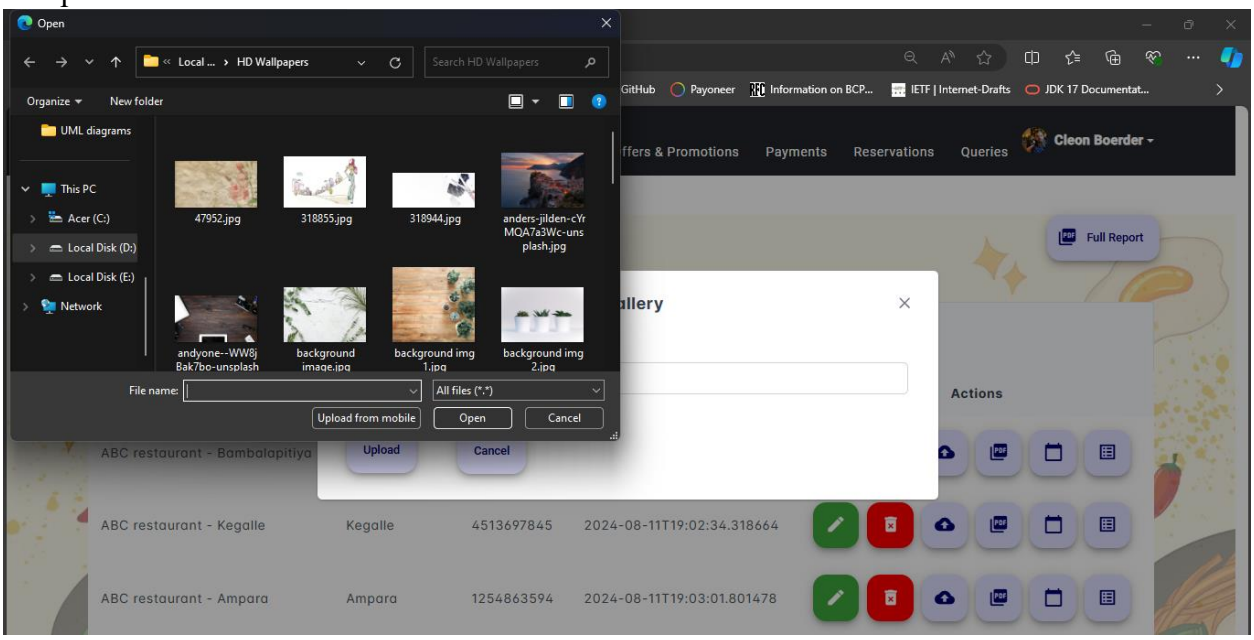
ABC restaurant – Anuradhapura is deleted.

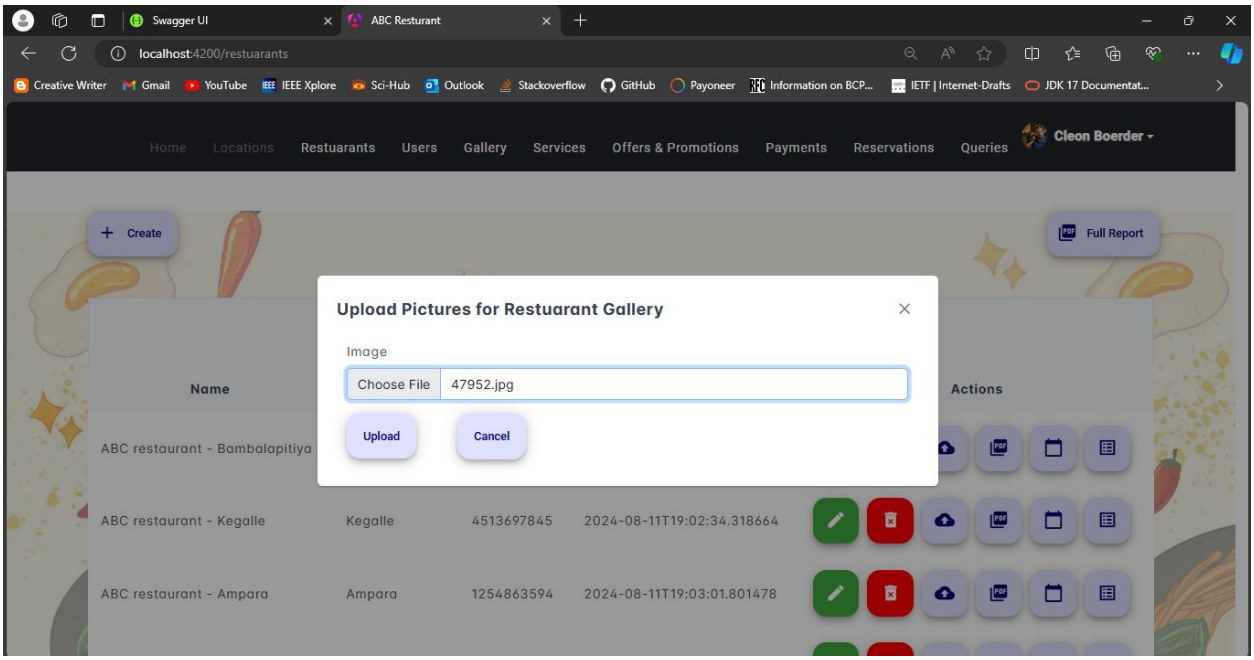
Gallery Management

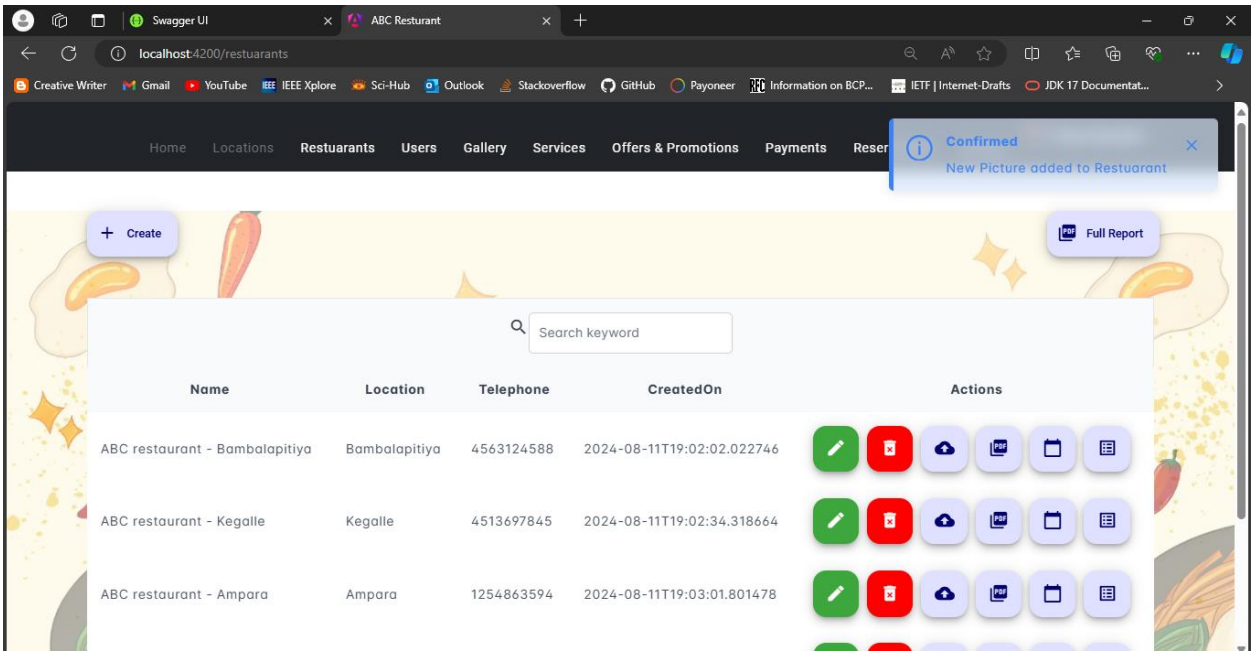
Upload images to restaurant gallery



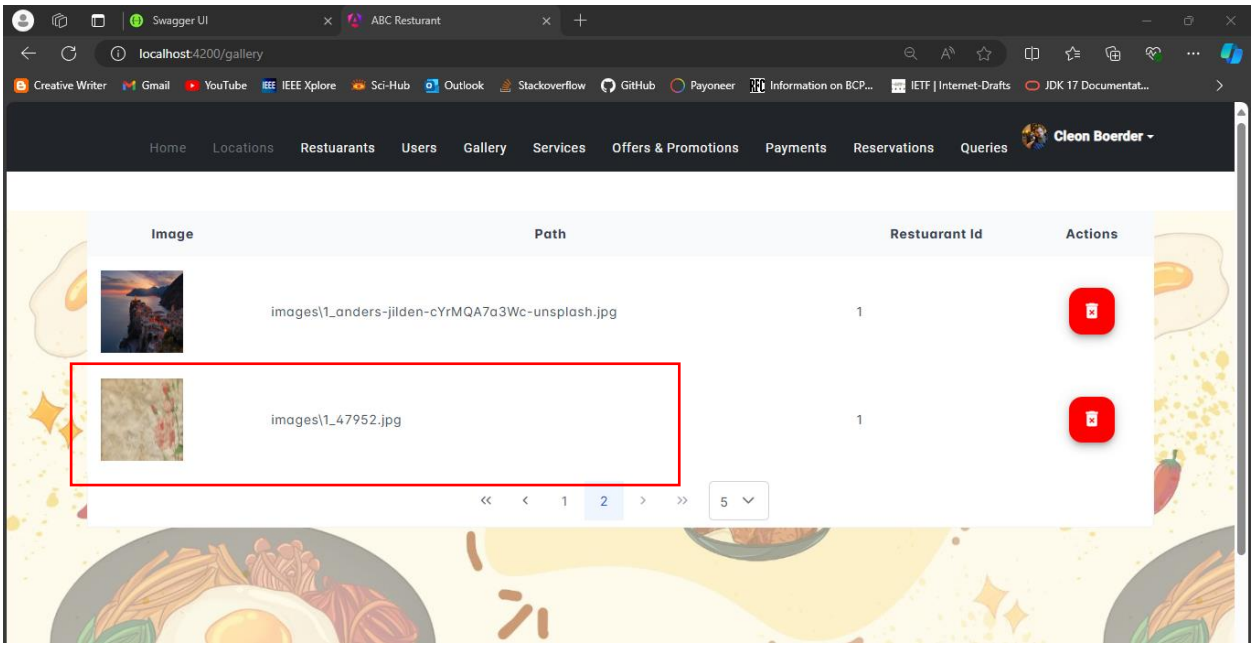
File picker



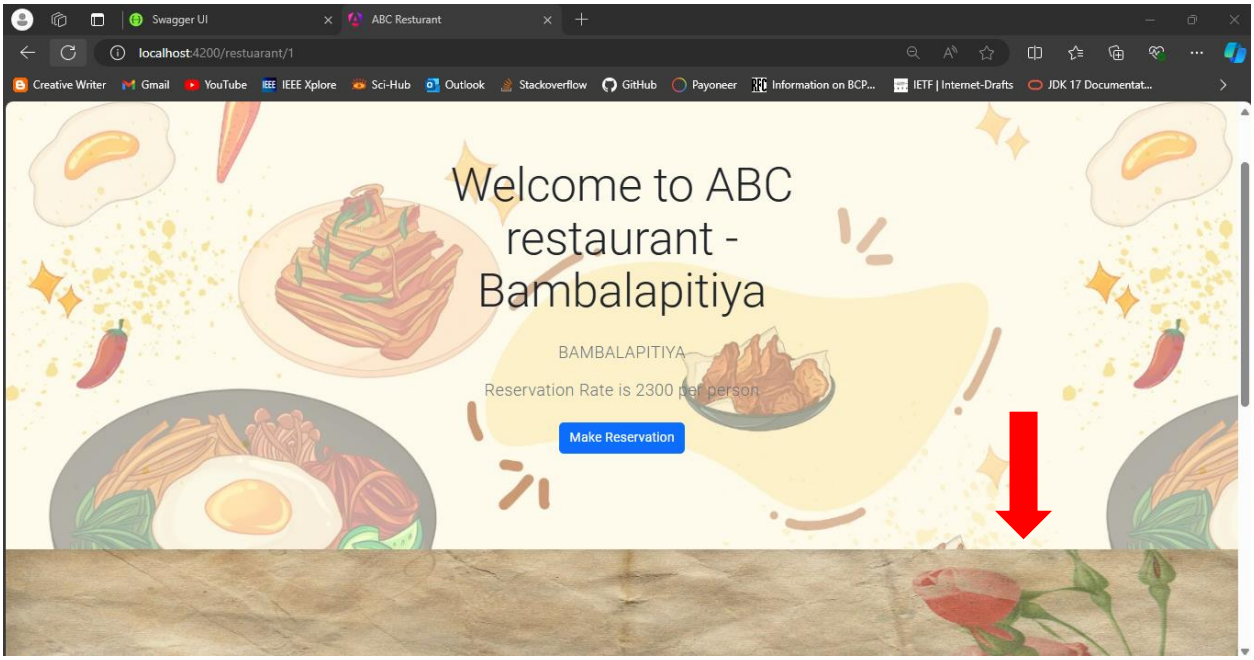

Swagger UI
ABC Resturant
localhost:4200/restuarants
Home
Locations
Restuarants
Users
Gallery
Services
Offers & Promotions
Payments
Reservations
Queries
Cleon Boerder
Create
Full Report
Upload Pictures for Restuarant Gallery
Image
Choose File
47952.jpg
Upload
Cancel
Name
Actions
ABC restaurant - Bambalapitiya
ABC restaurant - Kegalle
Kegalle
4513697845
2024-08-11T19:02:34.318664
ABC restaurant - Ampara
Ampara
1254863594
2024-08-11T19:03:01.801478


Swagger UI
ABC Resturant
localhost:4200/restuarants
Home
Locations
Restuarants
Users
Gallery
Services
Offers & Promotions
Payments
Confirmed
New Picture added to Restuarant
Create
Full Report
Search keyword
Name
Location
Telephone
CreatedOn
Actions
ABC restaurant - Bambalapitiya
Bambalapitiya
4563124588
2024-08-11T19:02:02.022746
ABC restaurant - Kegalle
Kegalle
4513697845
2024-08-11T19:02:34.318664
ABC restaurant - Ampara
Ampara
1254863594
2024-08-11T19:03:01.801478

Uploaded picture in the gallery.

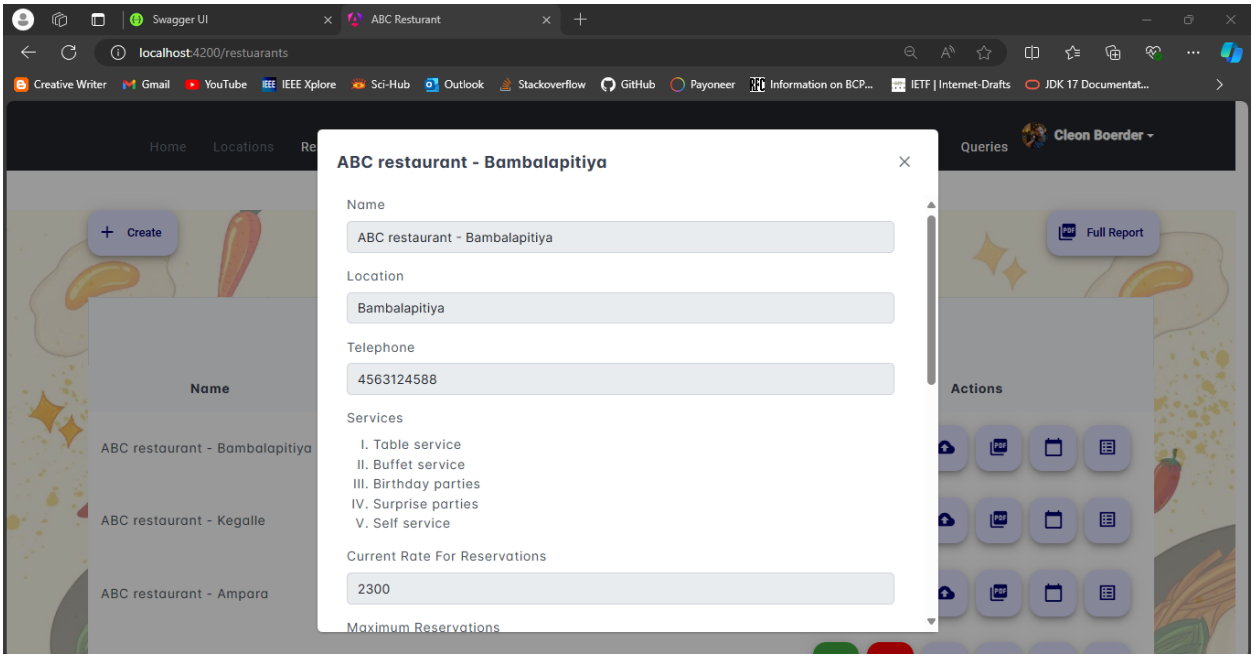


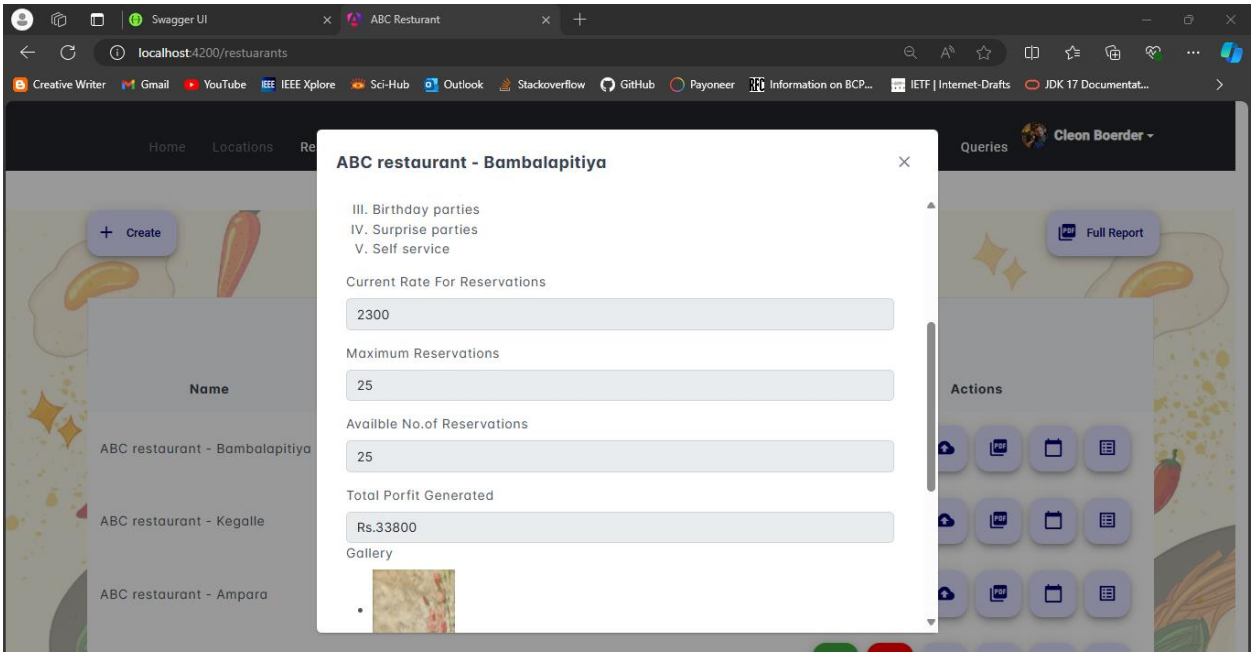
And in the relevant location, it is shown also.



View restaurant

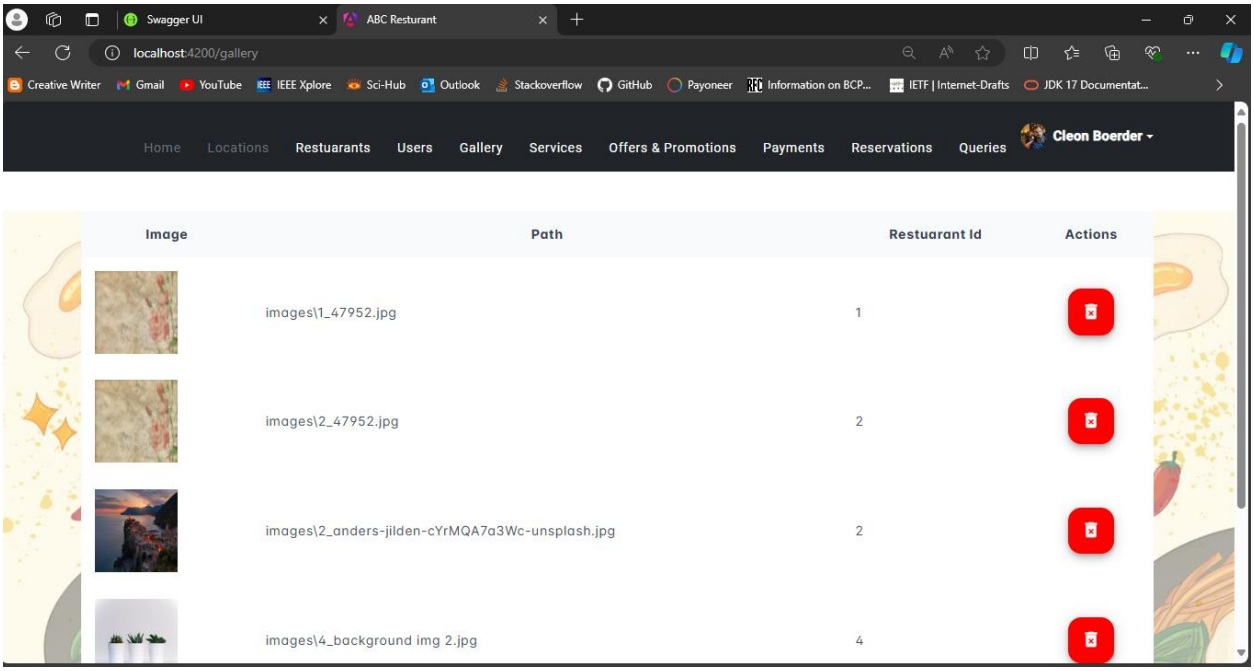
View button to view details



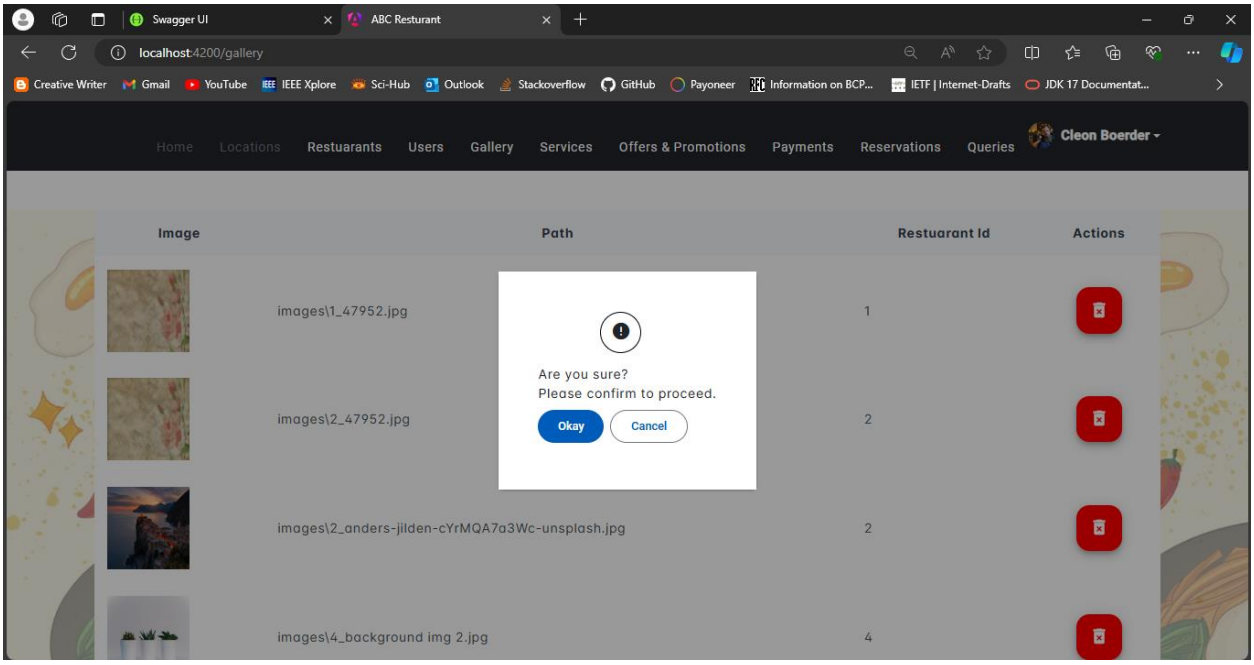


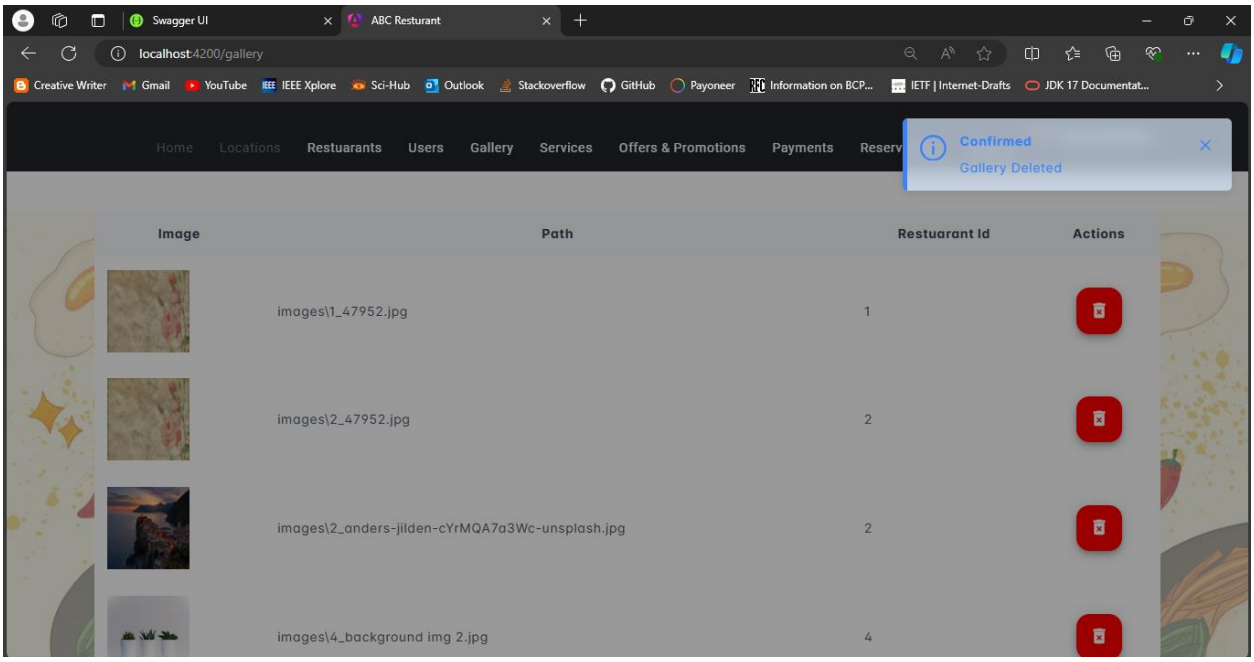
View gallery

Under gallery tab all the uploaded pictures are shown with the restaurant ID.

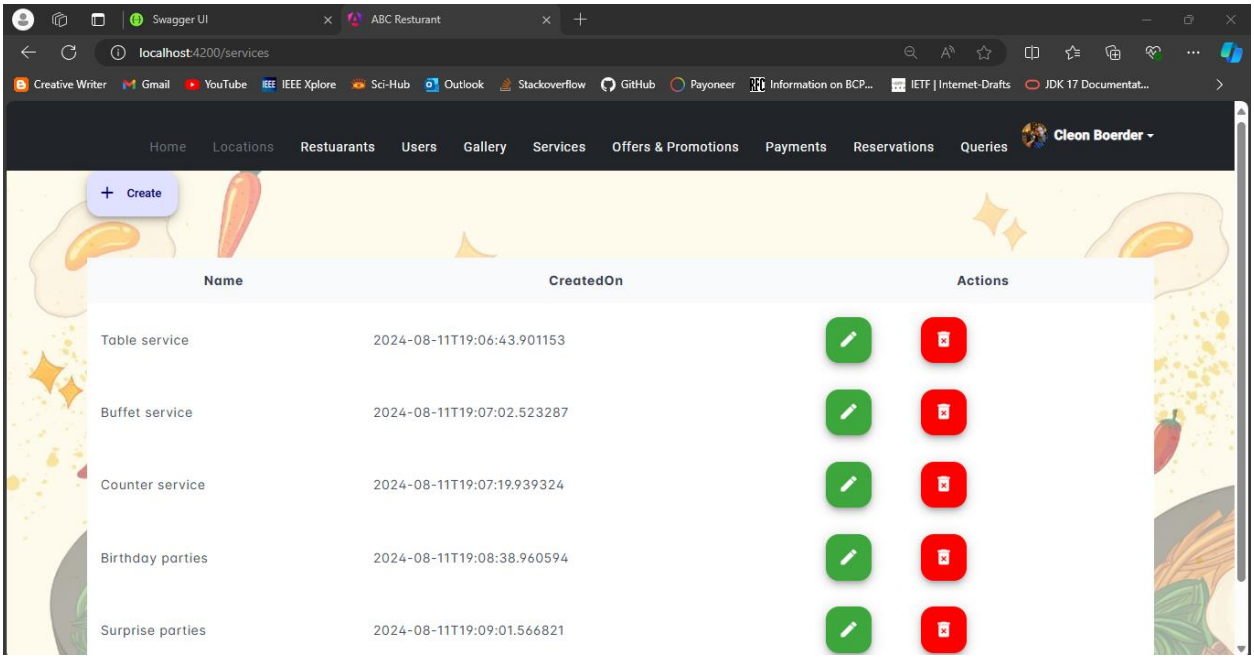


Delete gallery items



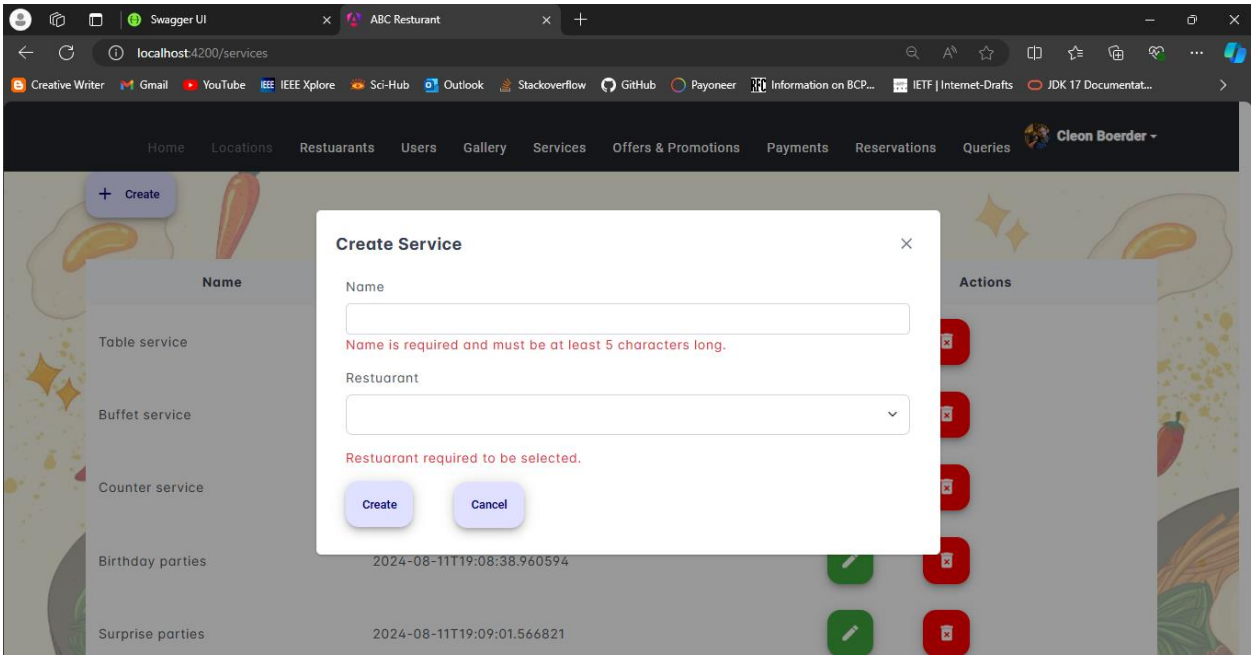


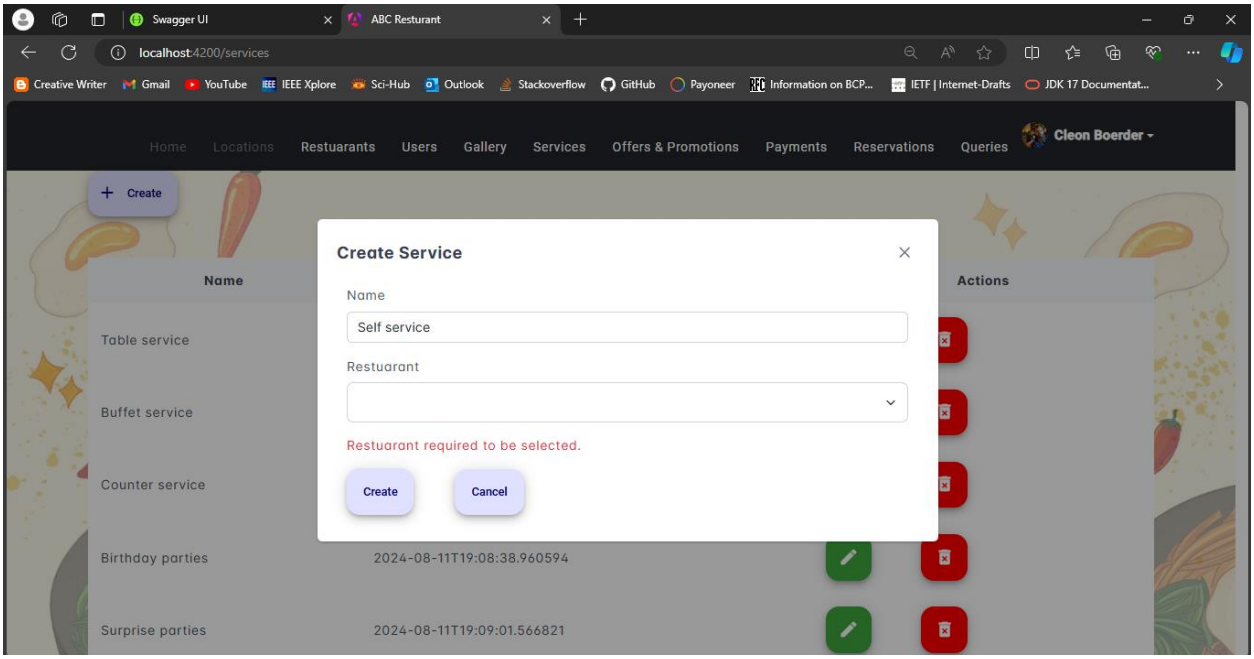
Service management

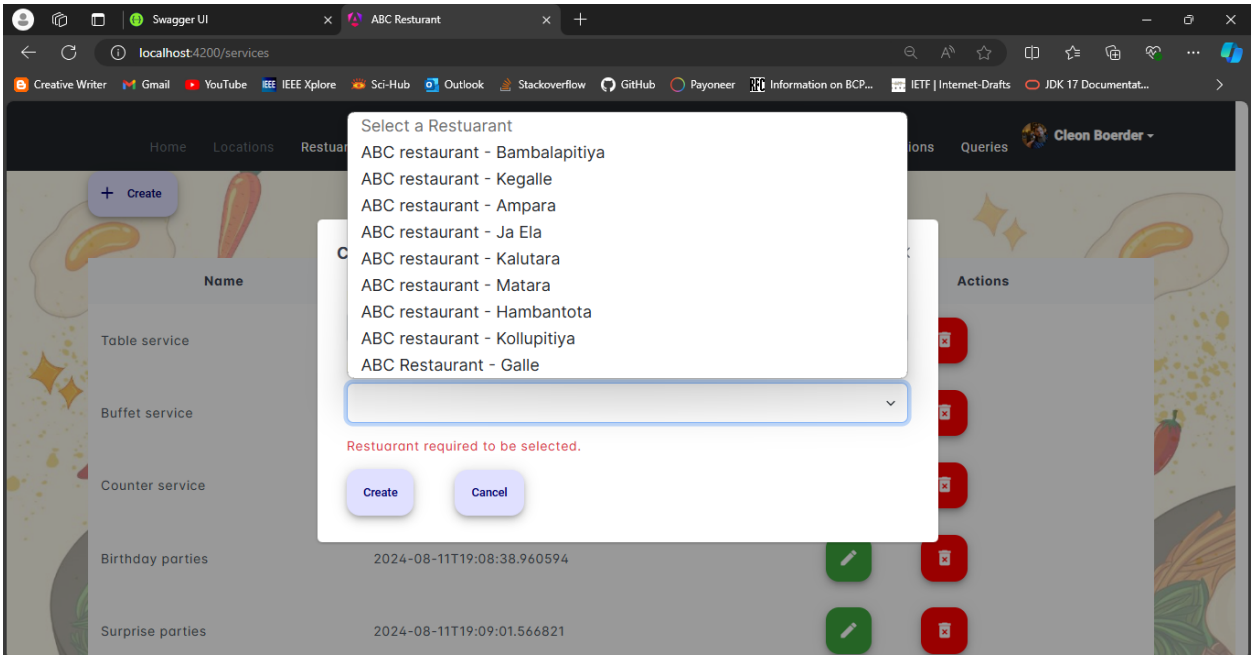


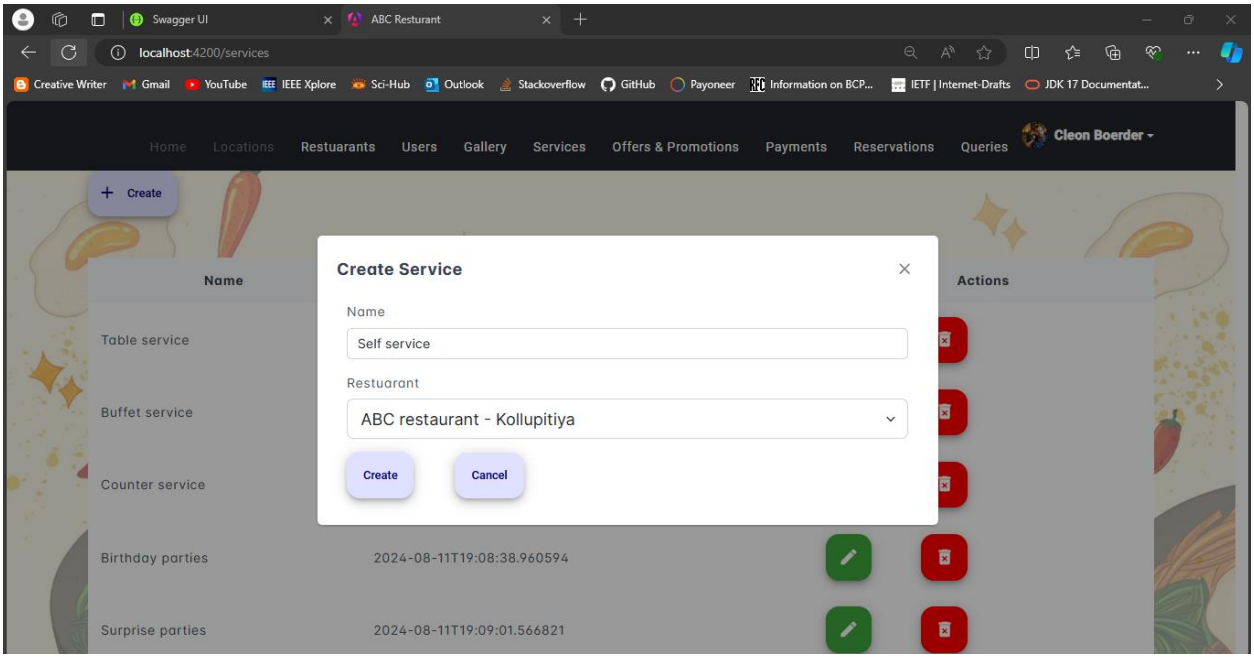
Create service

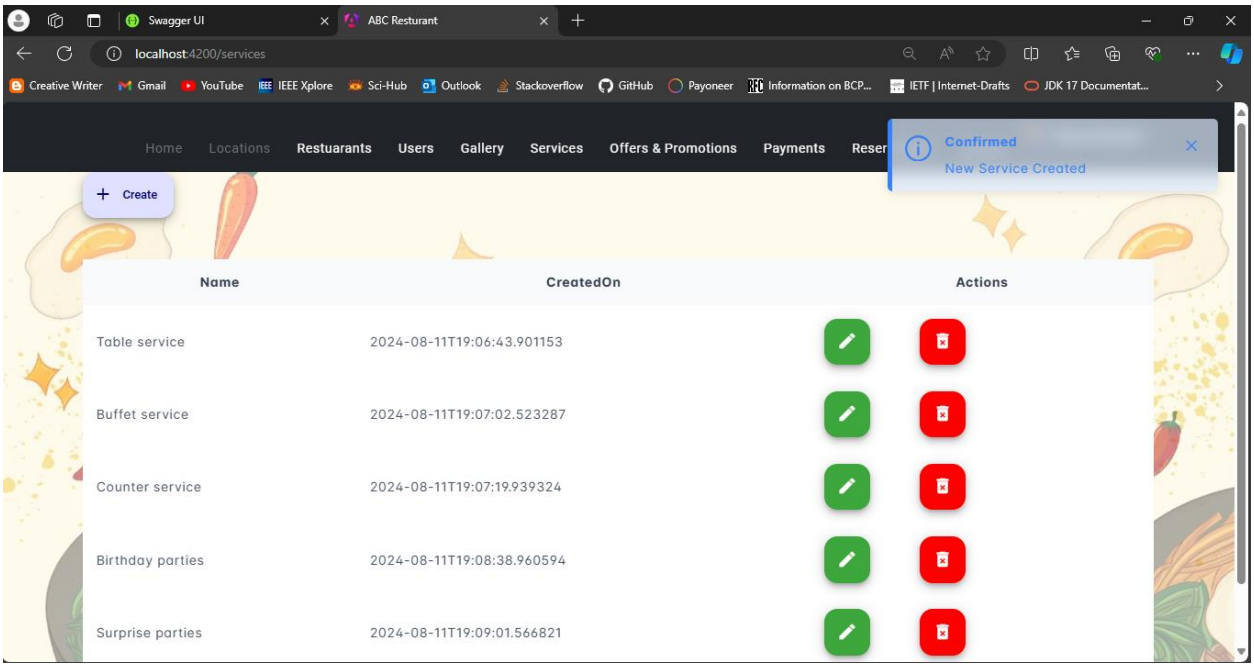
Form validations

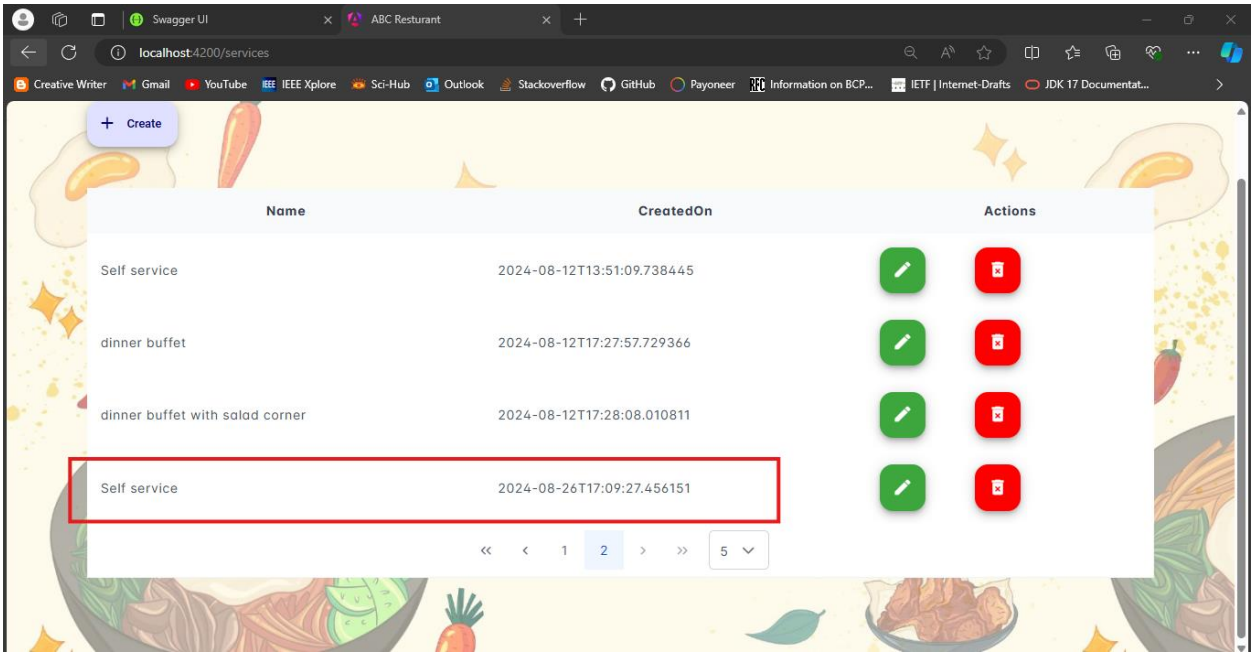





Swagger UI
ABC Resturant
localhost:4200/services
Home
Locations
Restuar
Queries
Cleon Boerder
Create
Select a Restuarant
ABC restaurant - Bambalapitiya
ABC restaurant - Kegalle
ABC restaurant - Ampara
ABC restaurant - Ja Ela
ABC restaurant - Kalutara
ABC restaurant - Matara
ABC restaurant - Hambantota
ABC restaurant - Kollupitiya
ABC Restaurant - Galle
Restuarant required to be selected.
Create
Cancel
Name
Actions
Table service
Buffet service
Counter service
Birthday parties
2024-08-11T19:08:38.960594
Surprise parties
2024-08-11T19:09:01.566821


Swagger UI
ABC Resturant
localhost:4200/services
Home
Locations
Restuarants
Users
Gallery
Services
Offers & Promotions
Payments
Reservations
Queries
Cleon Boerder
Create
Create Service
Name
Self service
Restuarant
ABC restaurant - Kollupitiya
Create
Cancel
Name
Actions
Table service
Buffet service
Counter service
Birthday parties
2024-08-11T19:08:38.960594
Surprise parties
2024-08-11T19:09:01.566821

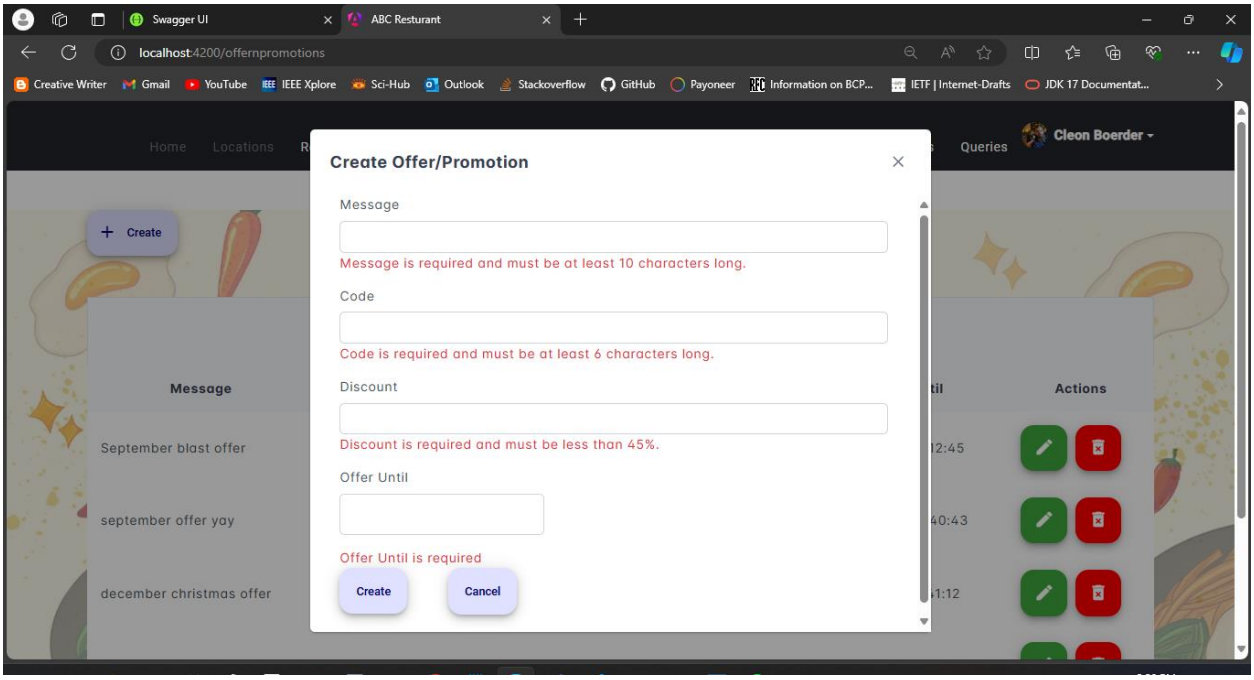

Swagger UI
ABC Resturant
localhost:4200/services
Home
Locations
Restuarants
Users
Gallery
Services
Offers & Promotions
Payments
Confirmed
New Service Created
Create
Name
CreatedOn
Actions
Table service
2024-08-11T19:06:43.901153
Buffet service
2024-08-11T19:07:02.523287
Counter service
2024-08-11T19:07:19.939324
Birthday parties
2024-08-11T19:08:38.960594
Surprise parties
2024-08-11T19:09:01.566821

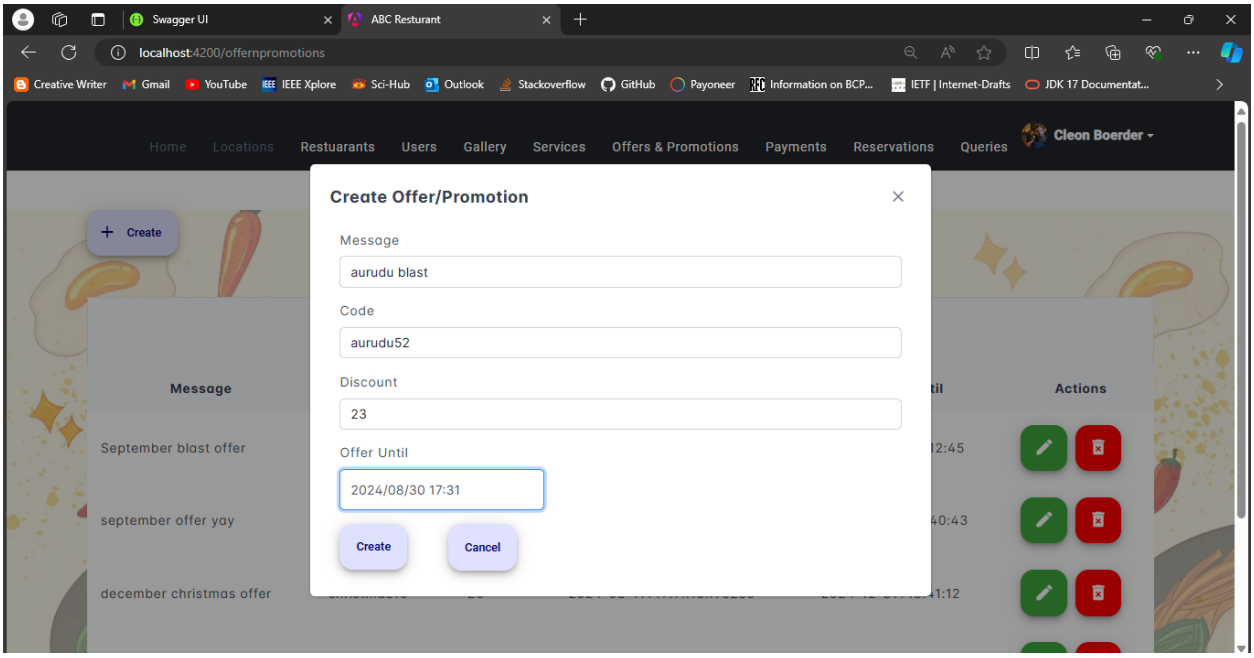

Swagger UI
ABC Resturant
localhost:4200/services
Create
Name
CreatedOn
Actions
Self service
2024-08-12T13:51:09.738445
dinner buffet
2024-08-12T17:27:57.729366
dinner buffet with salad corner
2024-08-12T17:28:08.010811
Self service
2024-08-26T17:09:27.456151
1
2
5

Offers and promotions management

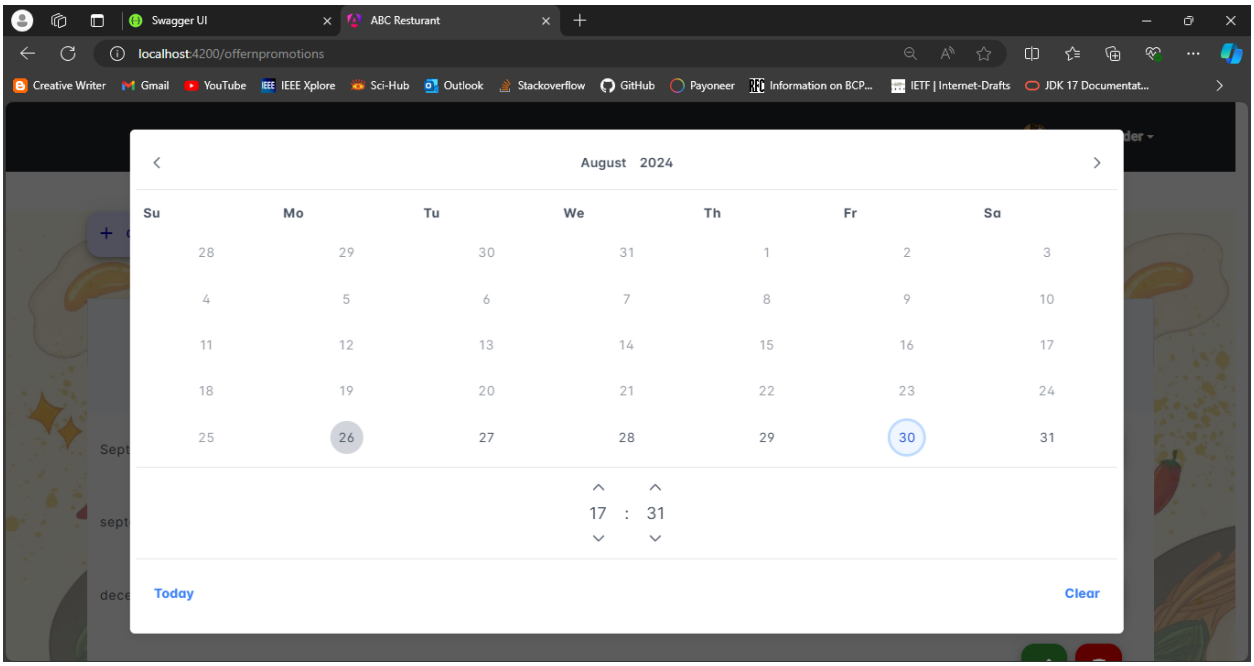
Create offer and promotion

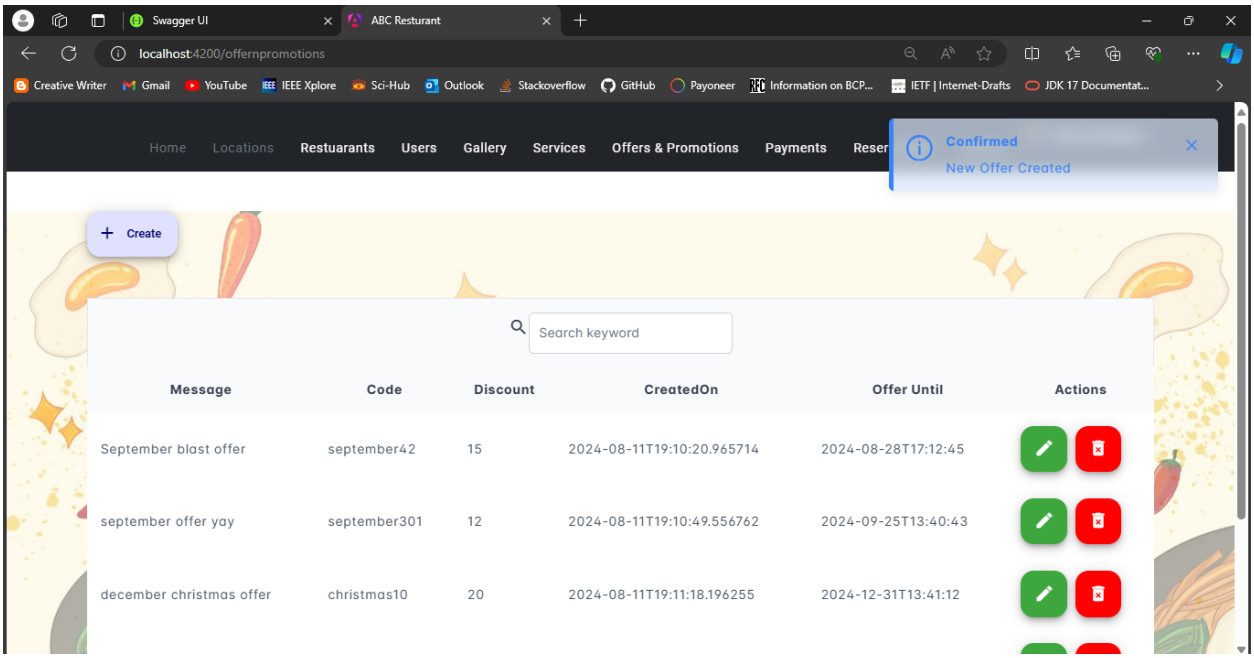
Form validations

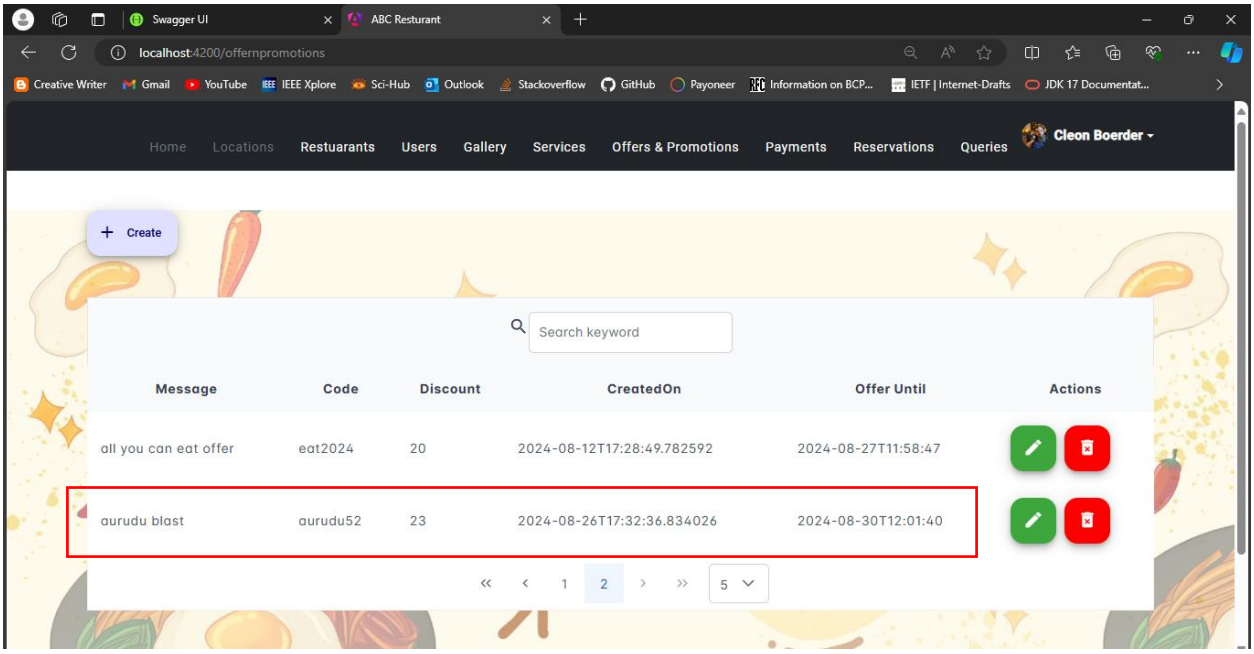




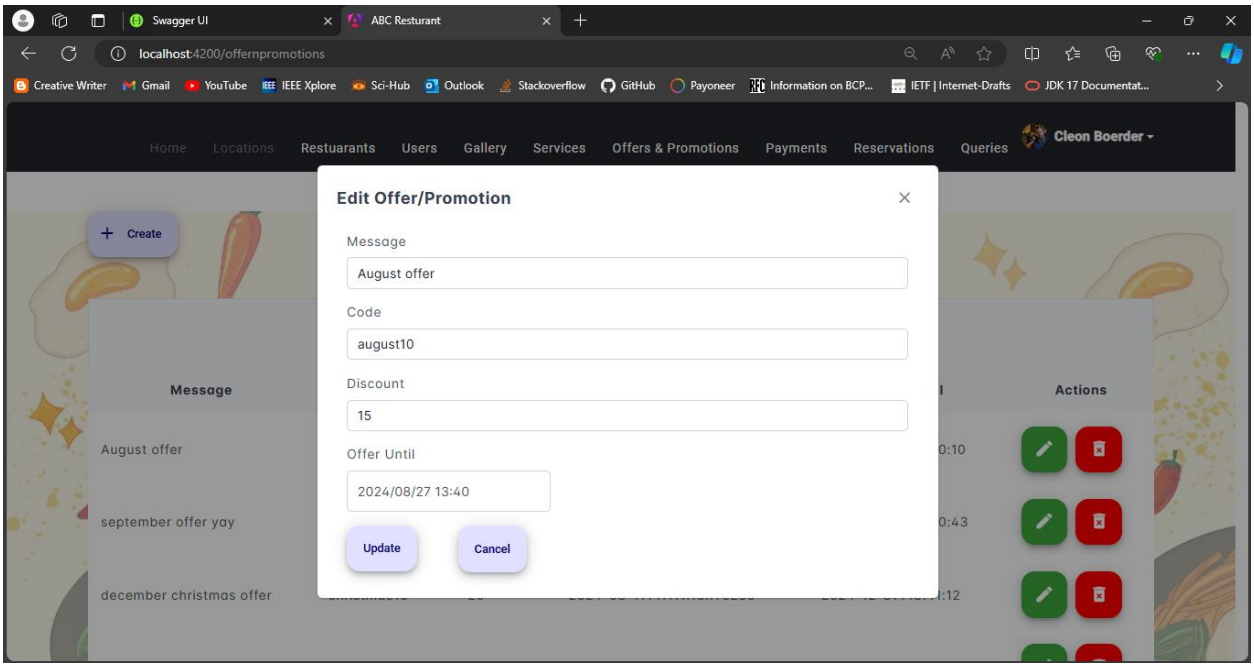
Date picker


Swagger UI
ABC Resturant
localhost:4200/offernpromotions
August 2024
Su Mo Tu We Th Fr Sa
17 : 31
Today
Clear

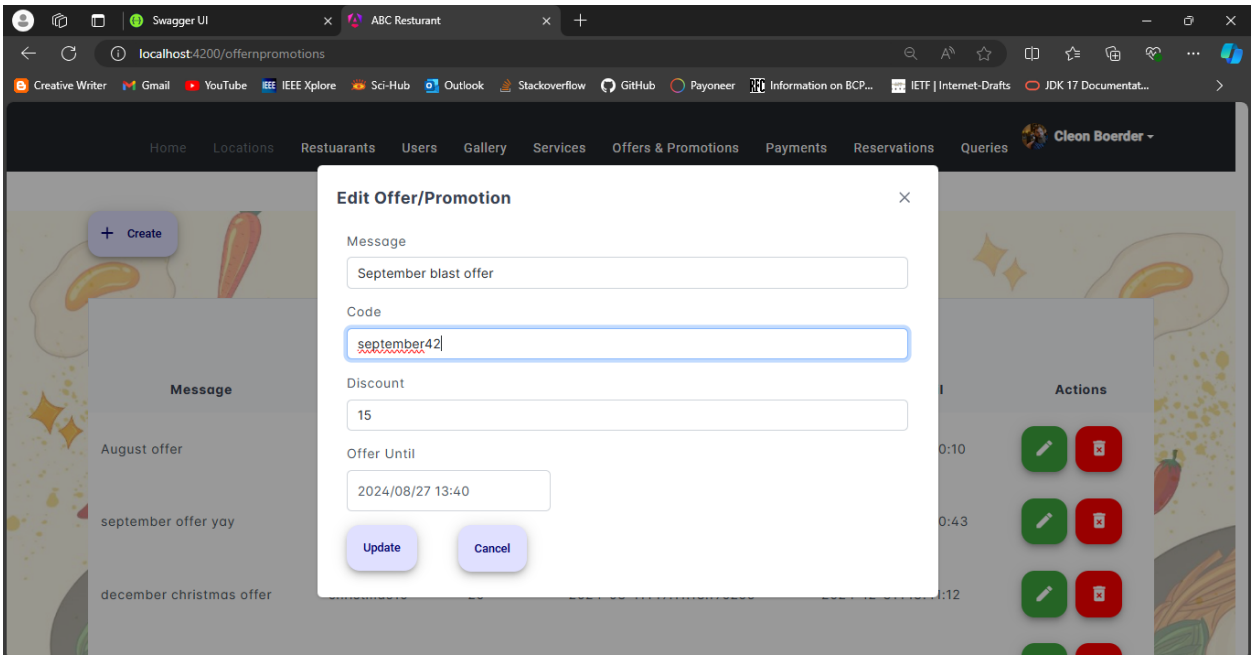

Swagger UI
ABC Resturant
localhost:4200/offernpromotions
Home Locations Restuarants Users Gallery Services Offers & Promotions Payments
Confirmed
New Offer Created
Create
Search keyword
Message Code Discount CreatedOn Offer Until Actions
September blast offer september42 15 2024-08-11T19:10:20.965714 2024-08-28T17:12:45
september offer yay september301 12 2024-08-11T19:10:49.556762 2024-09-25T13:40:43
december christmas offer christmas10 20 2024-08-11T19:11:18.196255 2024-12-31T13:41:12

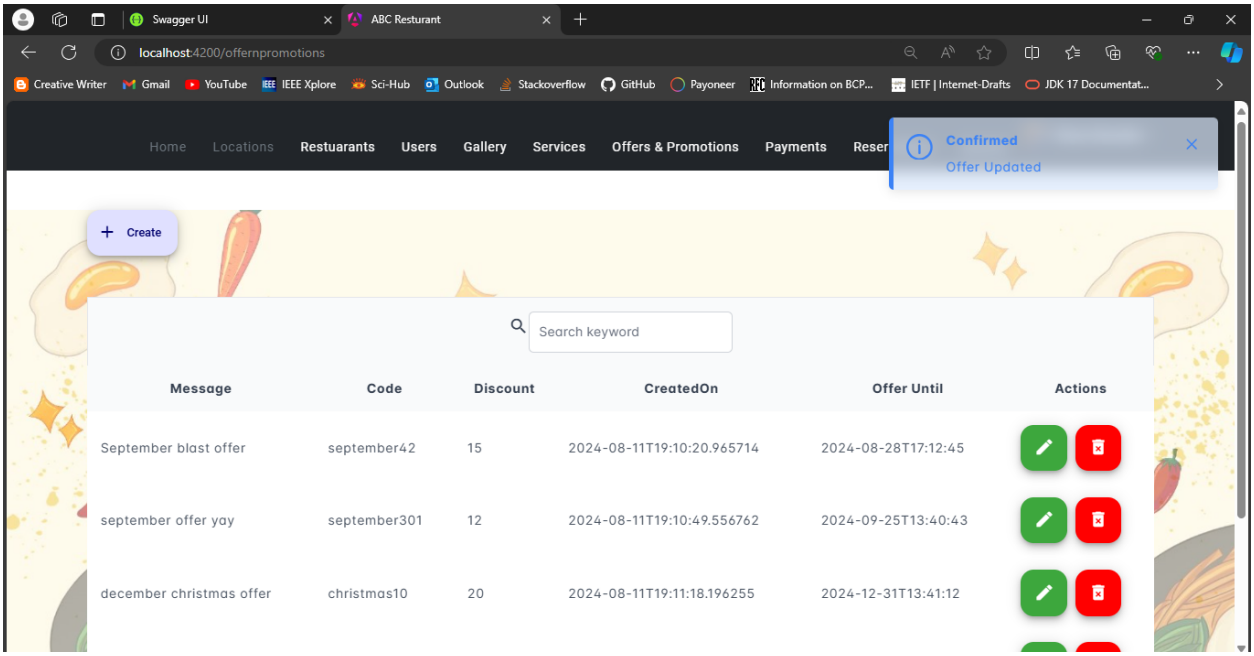


Edit offer and promotion

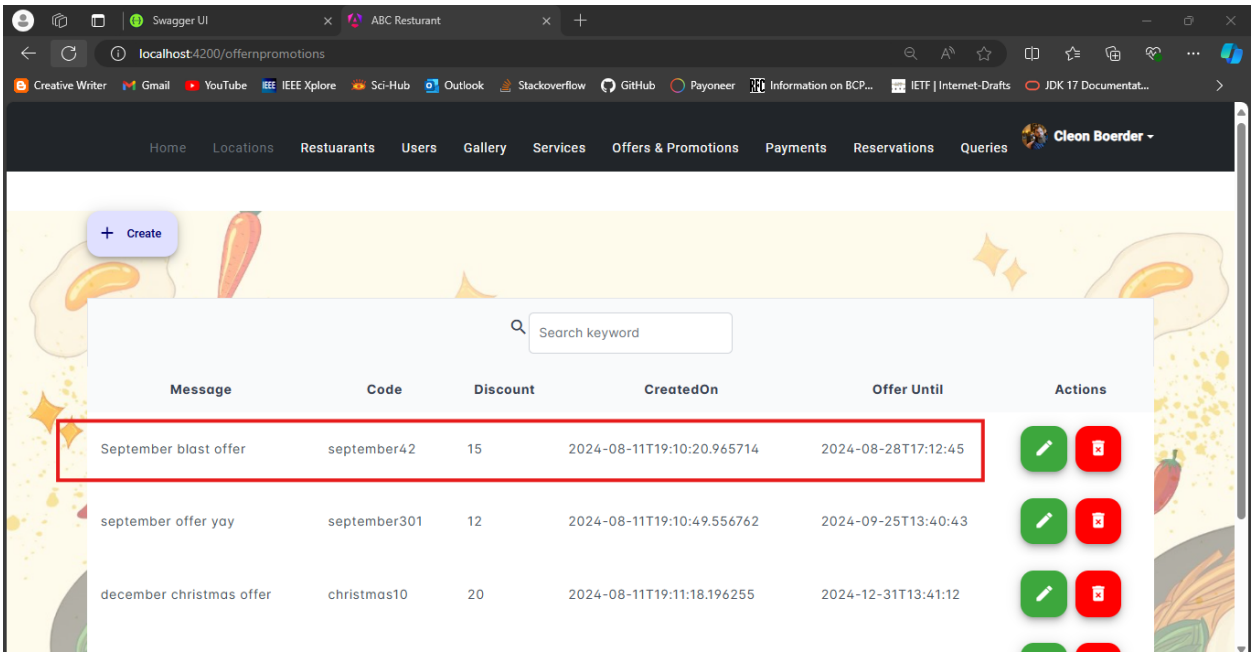


Above offer was edited as below

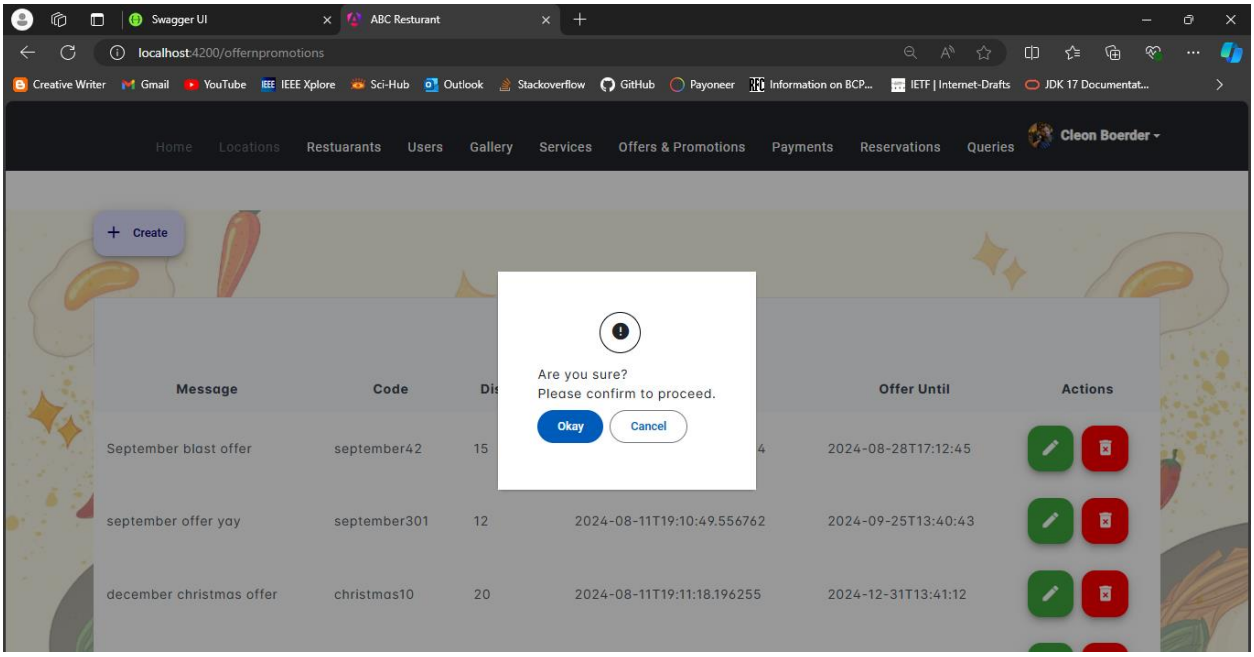




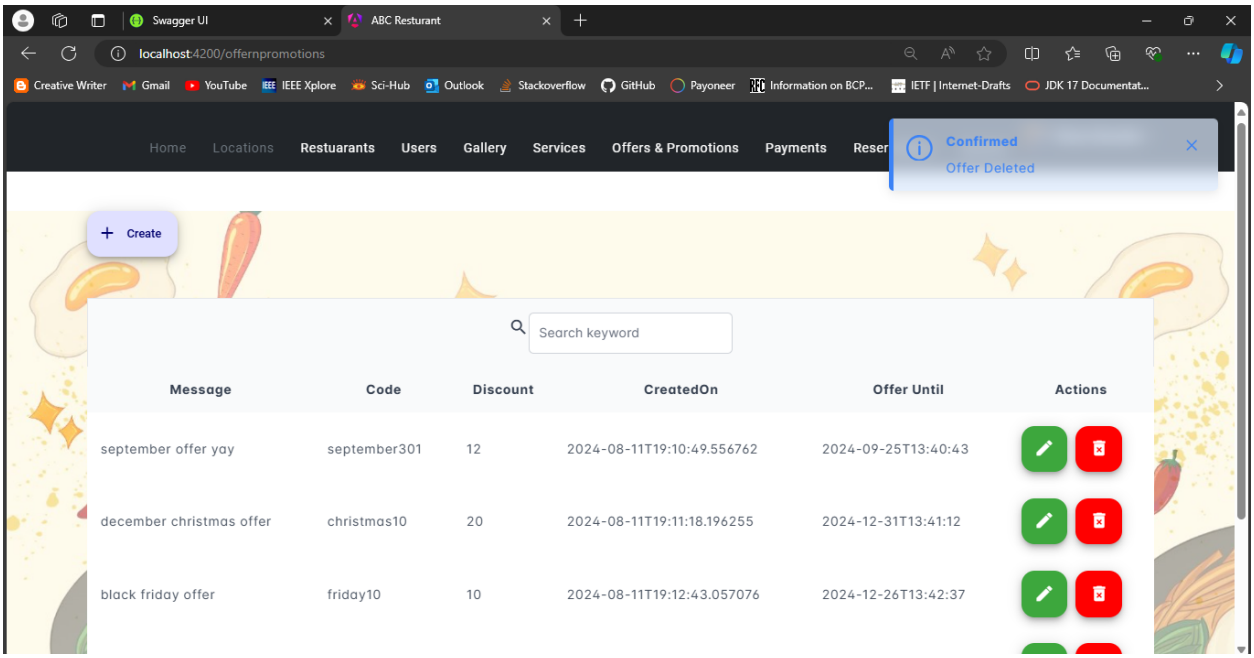
Successfully updated



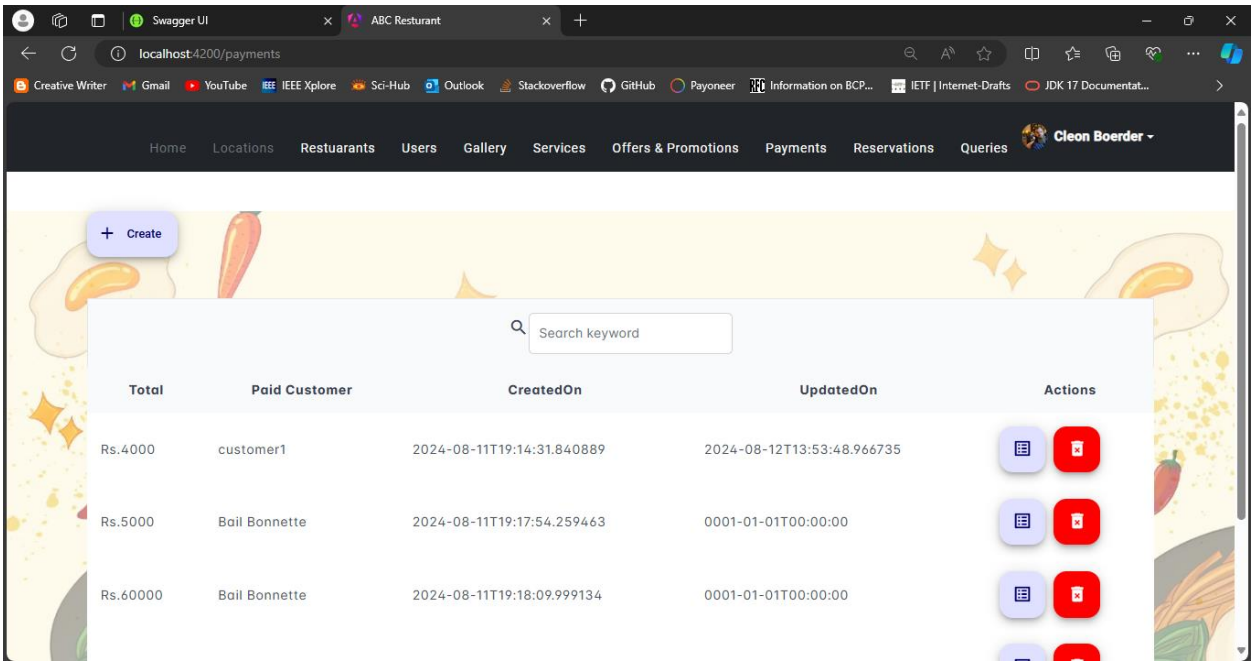
Delete offer and promotion



September blast offer deleted

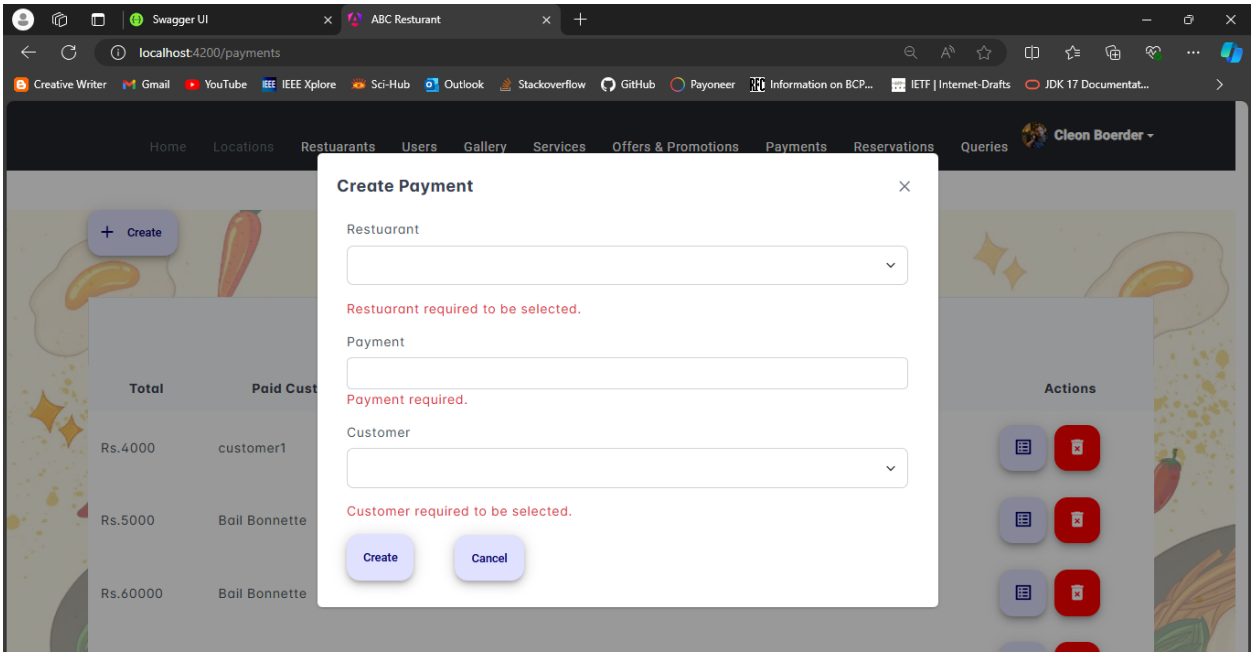


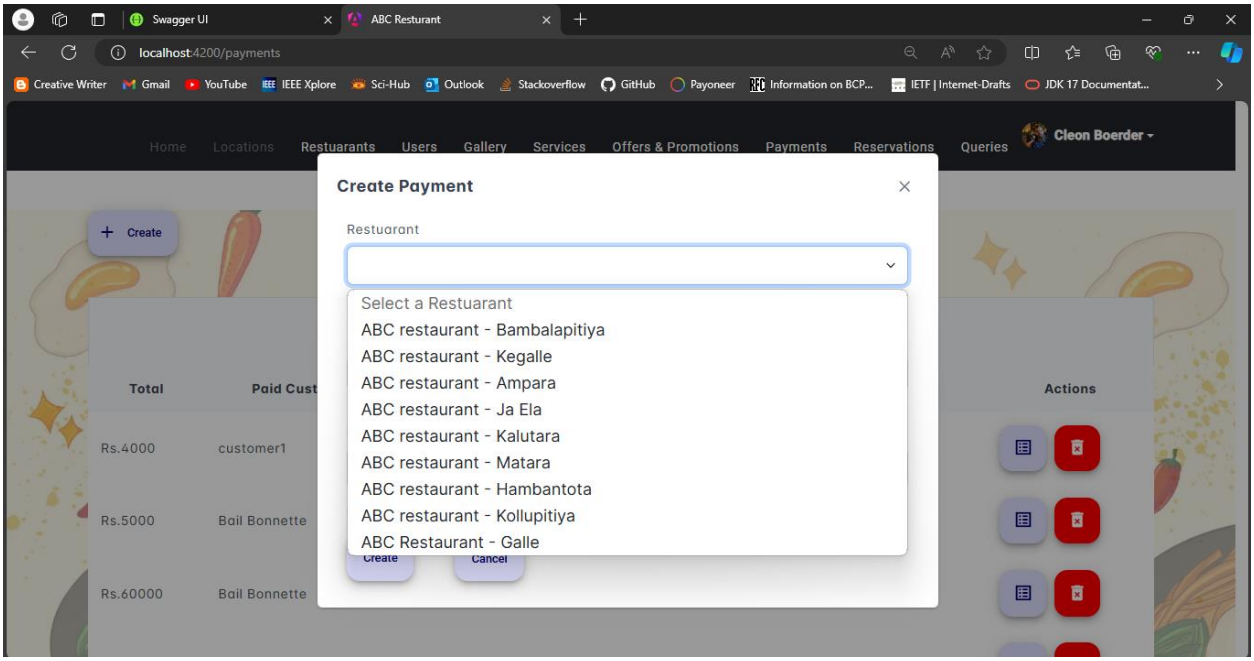
Manage payments

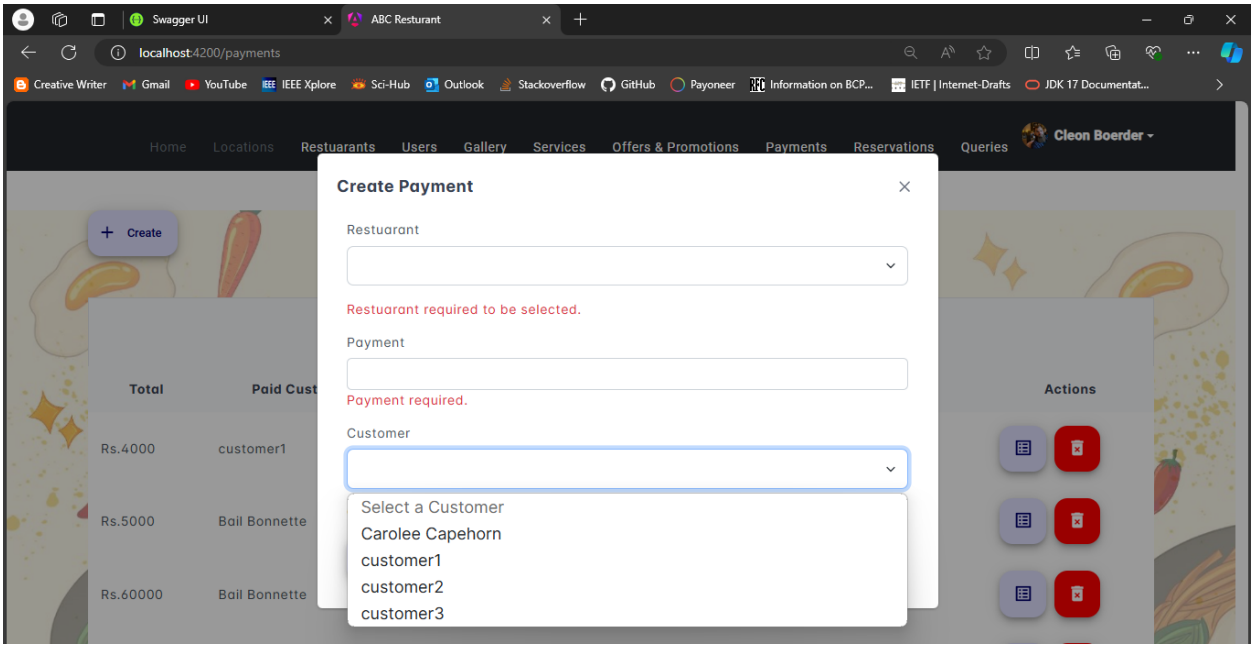


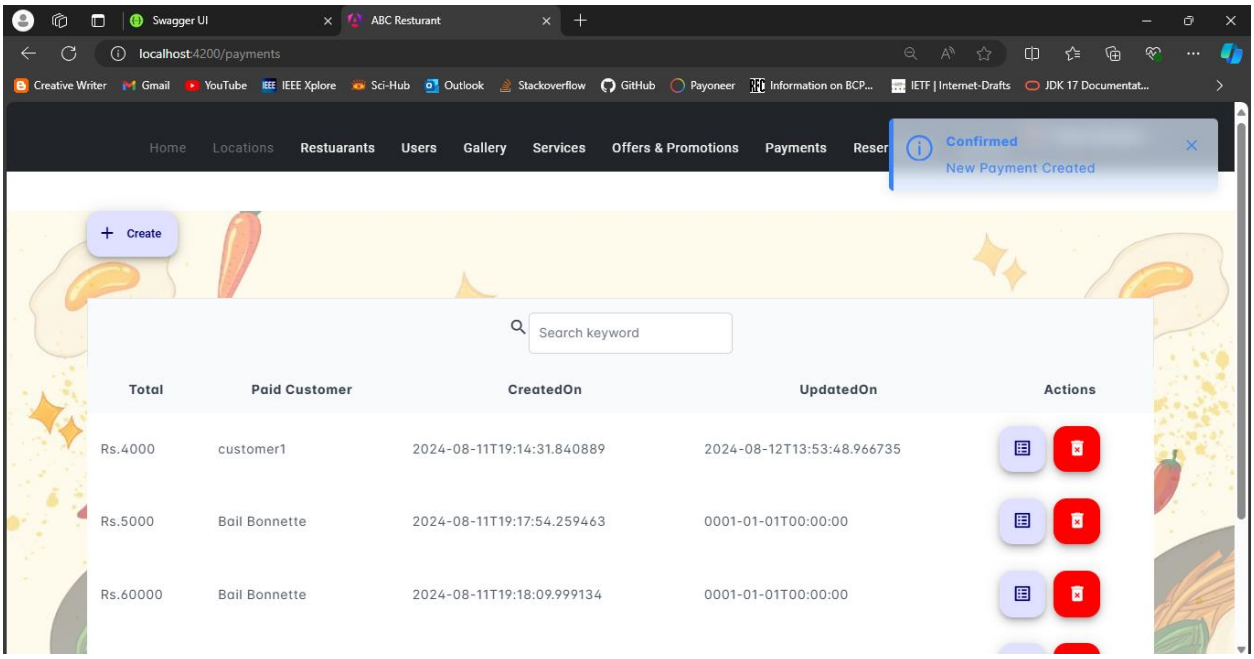
Create payment

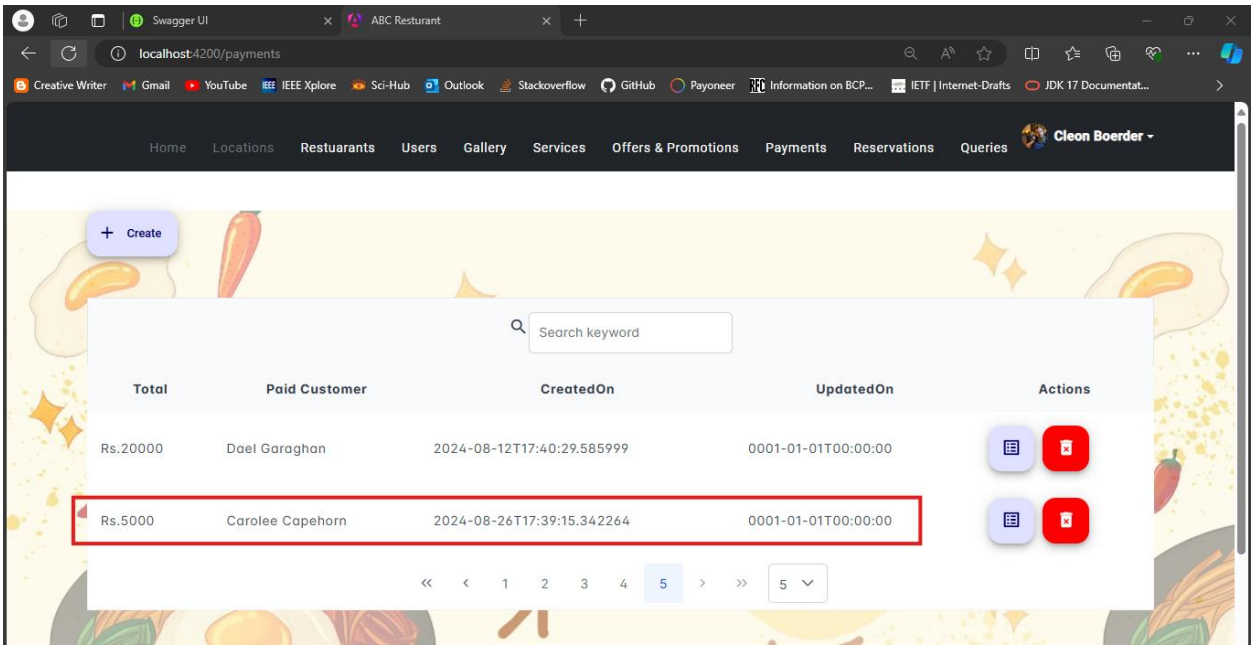
Form validations



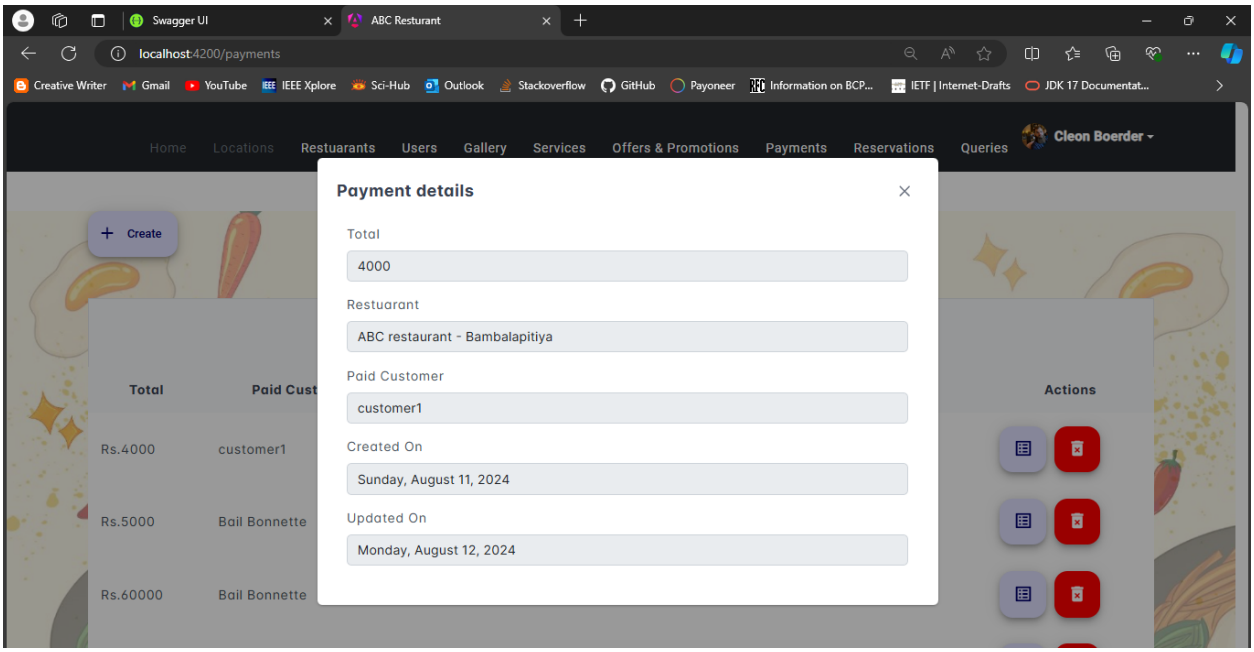



Swagger UI
ABC Resturant
localhost:4200/payments
Home
Locations
Restuarants
Users
Gallery
Services
Offers & Promotions
Payments
Reservations
Queries
Cleon Boerder
Create Payment
Restuarant
Restuarant required to be selected.
Payment
Payment required.
Customer
Select a Customer
Carolee Capehorn
customer1
customer2
customer3
Create
Total
Paid Cust
Actions
Rs.4000
customer1
Rs.5000
Bail Bonnette
Rs.60000
Bail Bonnette

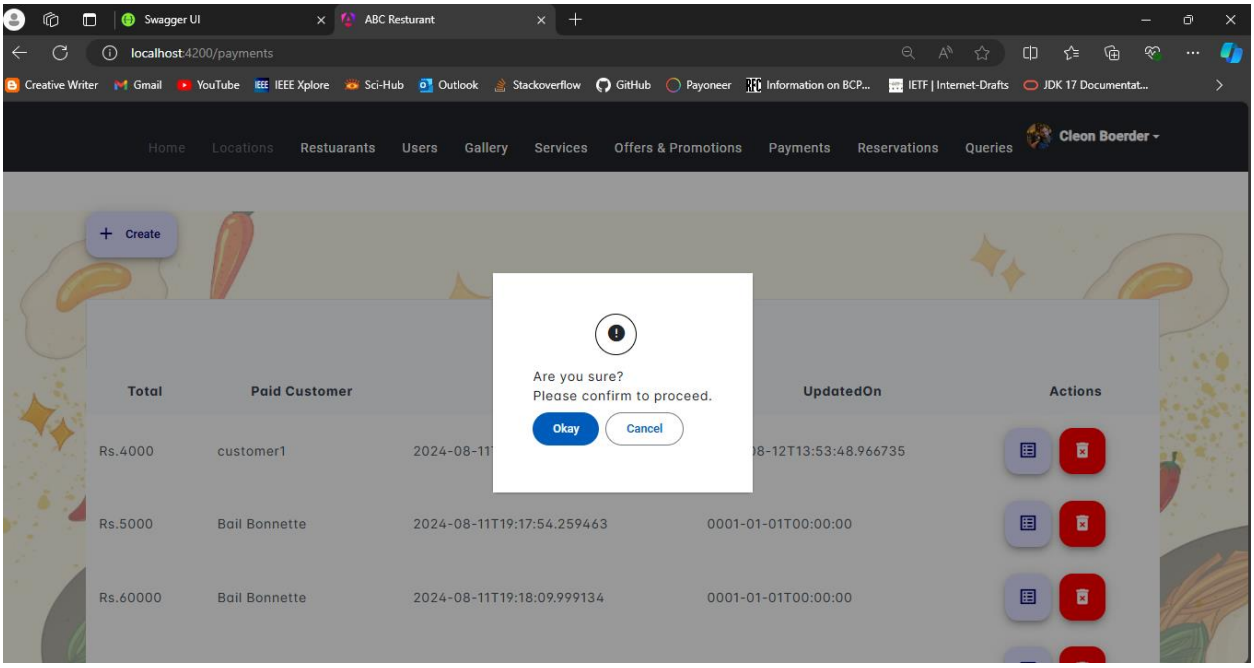

Swagger UI
ABC Resturant
localhost:4200/payments
Home
Locations
Restuarants
Users
Gallery
Services
Offers & Promotions
Payments
Confirmed
New Payment Created
Create
Search keyword
Total
Paid Customer
CreatedOn
UpdatedOn
Actions
Rs.4000
customer1
2024-08-11T19:14:31.840889
2024-08-12T13:53:48.966735
Rs.5000
Bail Bonnette
2024-08-11T19:17:54.259463
0001-01-01T00:00:00
Rs.60000
Bail Bonnette
2024-08-11T19:18:09.999134
0001-01-01T00:00:00



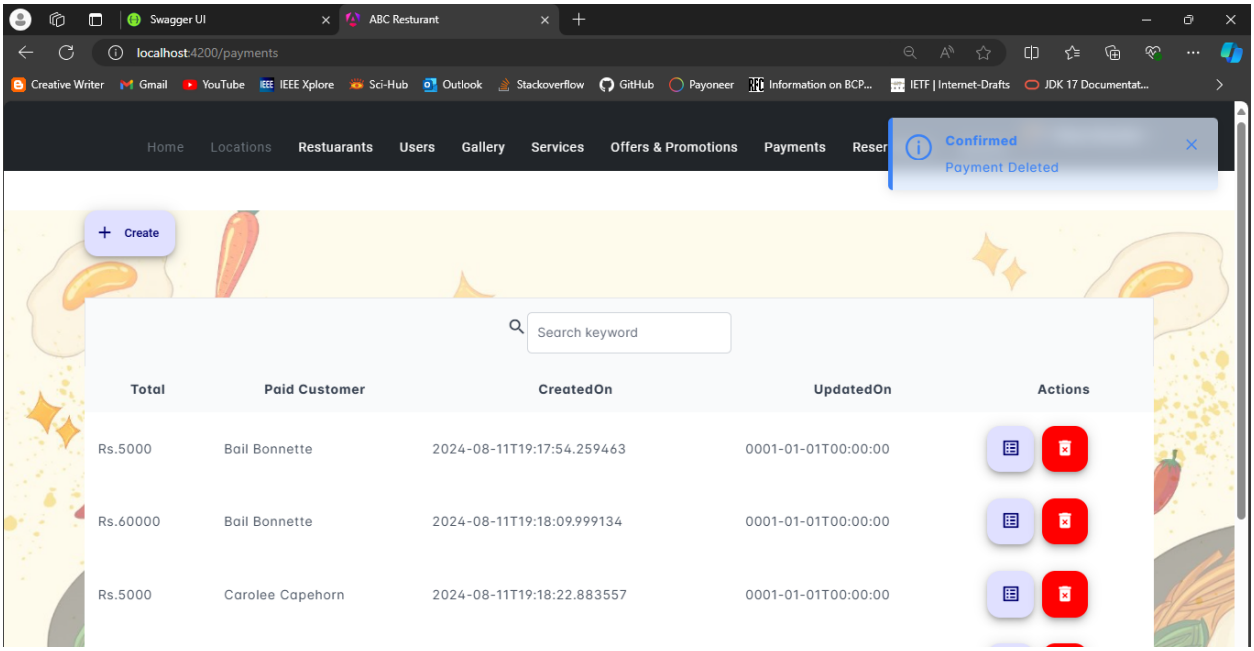
View payment details



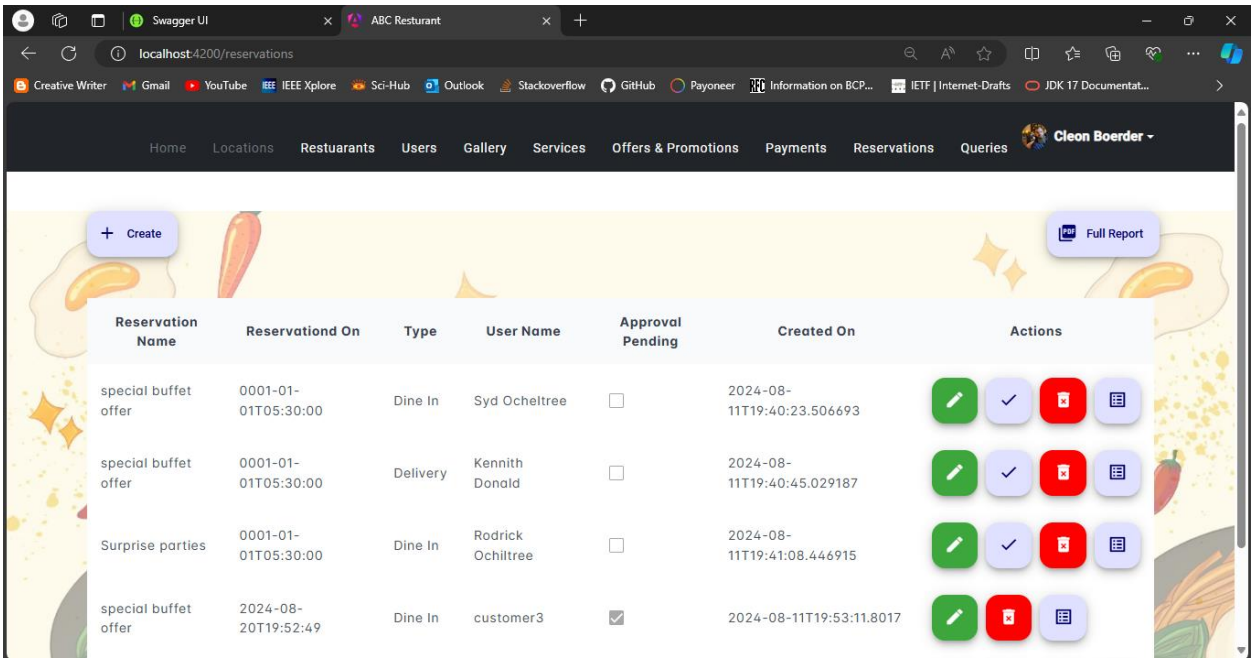
Delete payment



Customer1's payment is deleted

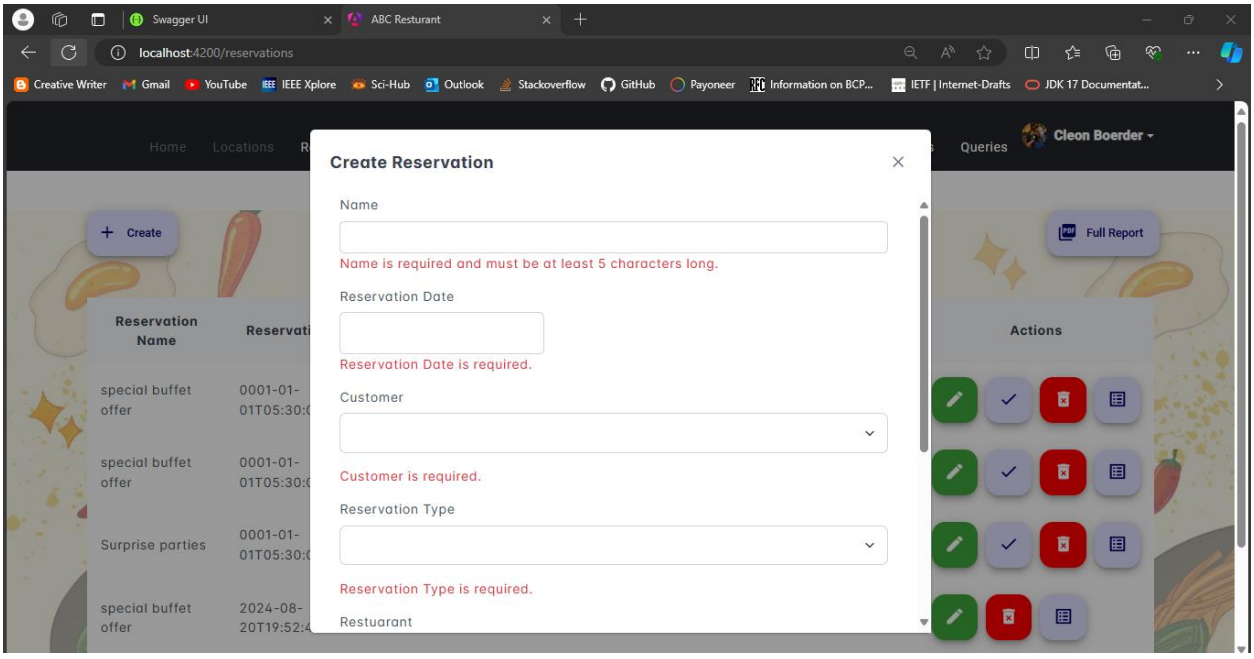


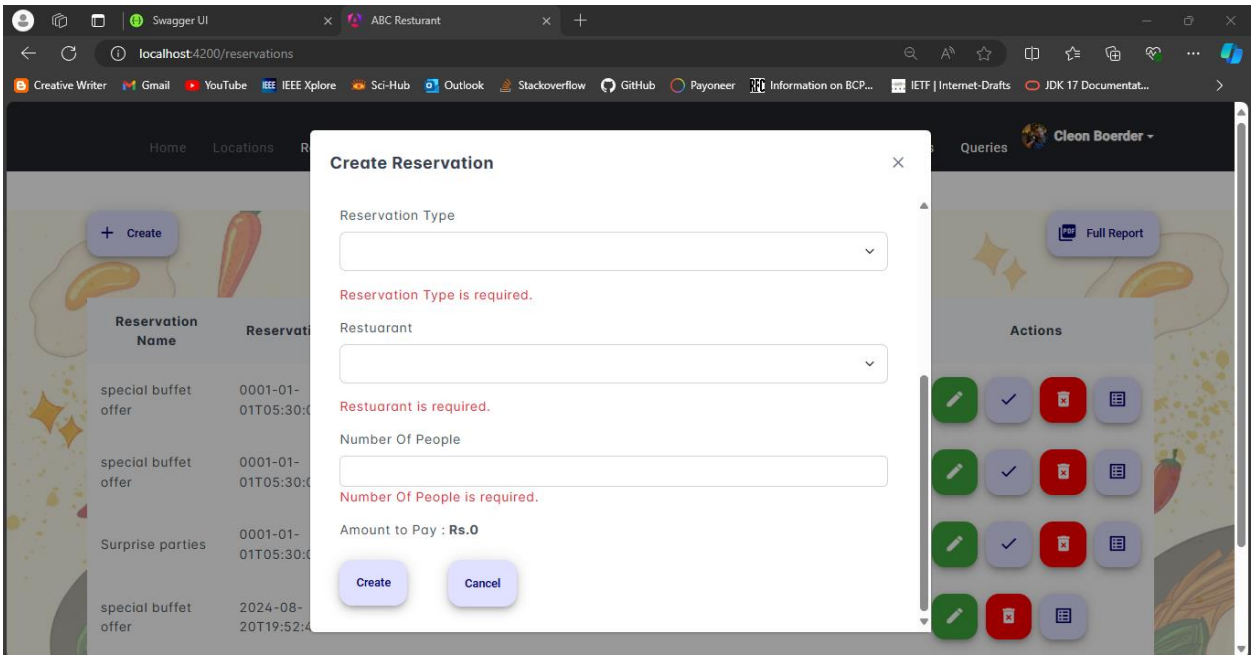
Reservation management

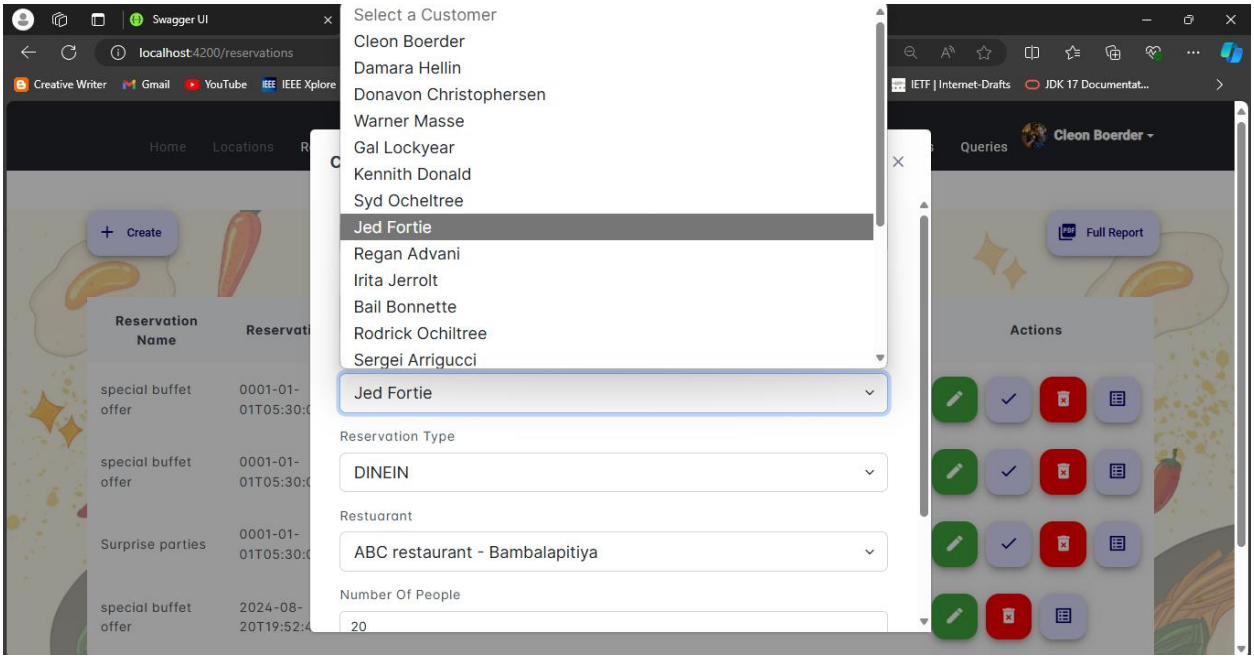


Create reservation

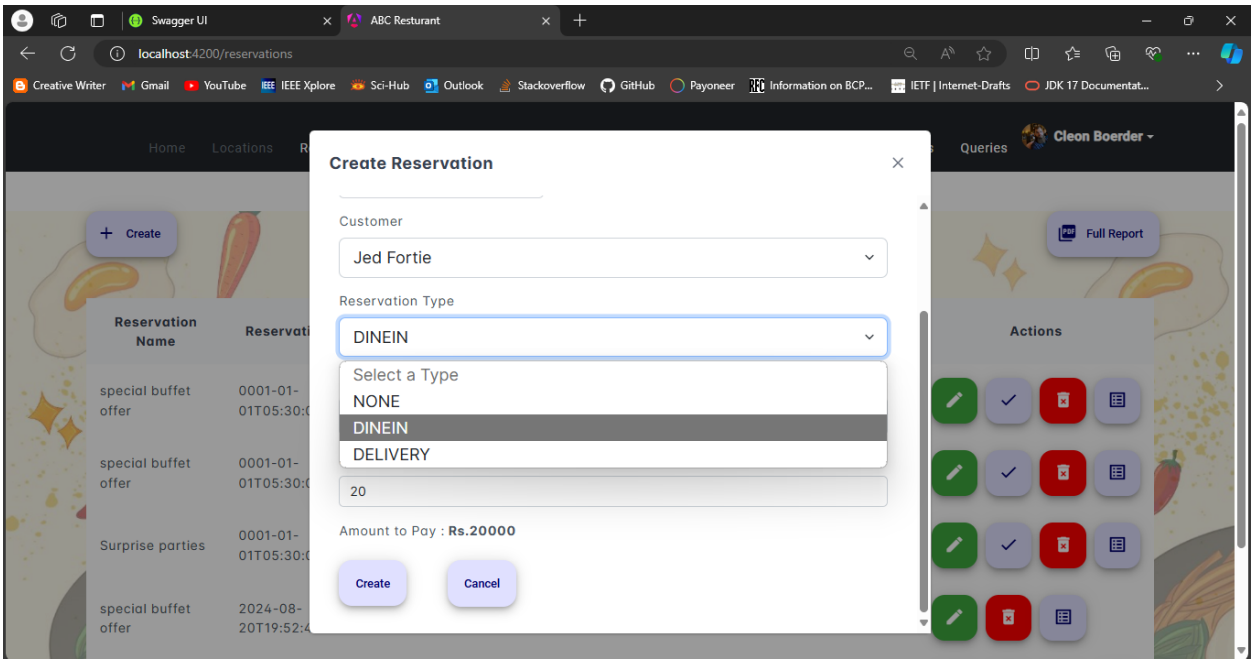
Form validations

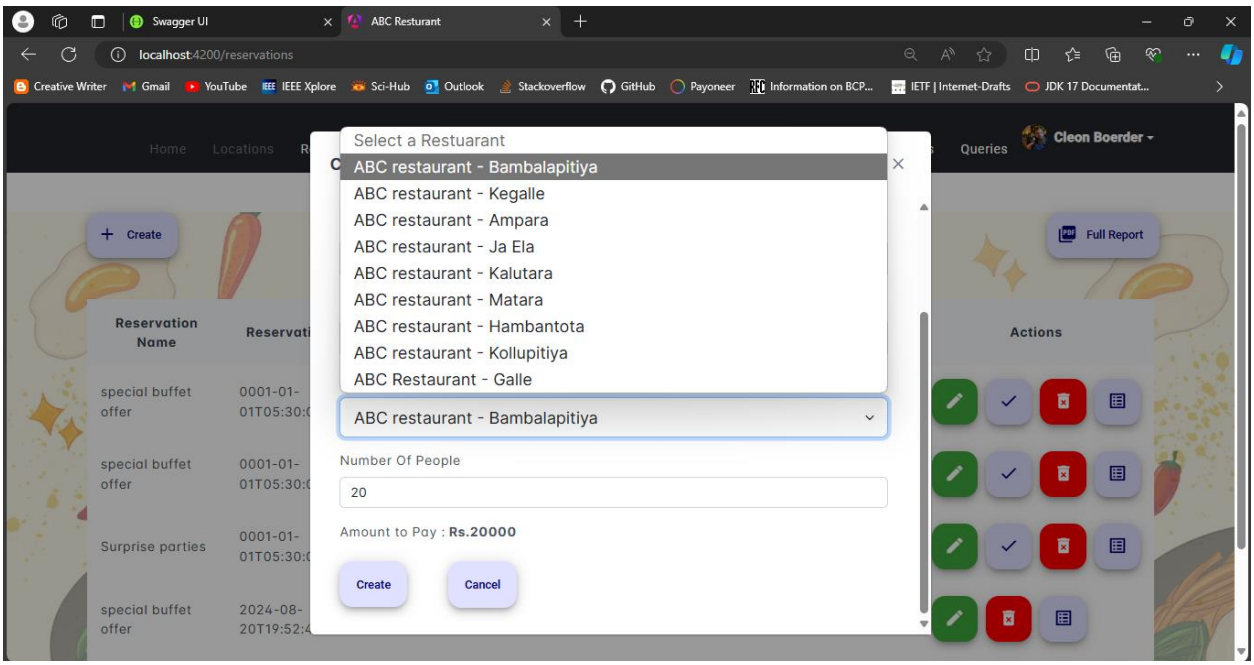



Swagger UI
ABC Resturant
localhost:4200/reservations
Create Reservation
Reservation Type
Reservation Type is required.
Restuarant
Restuarant is required.
Number Of People
Number Of People is required.
Amount to Pay : Rs.0
Create
Cancel
Cleon Boerder
Queries
Full Report
Actions
Reservation Name

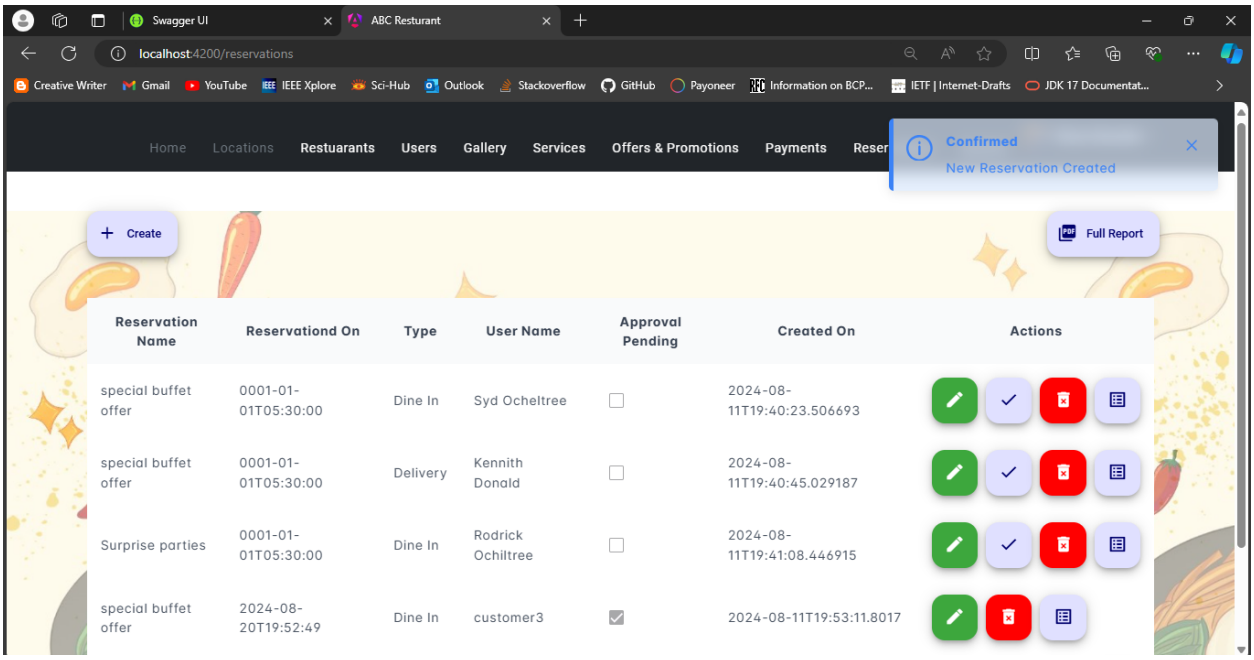

Select a Customer
Cleon Boerder
Damara Hellin
Donavon Christophersen
Warner Masse
Gal Lockyear
Kennith Donald
Syd Ocheltree
Jed Fortie
Regan Advani
Irita Jerrolt
Bail Bonnette
Rodrick Ochiltree
Sergei Arrigucci
Jed Fortie
Reservation Type
DINEIN
Restuarant
ABC restaurant - Bambalapitiya
Number Of People
20

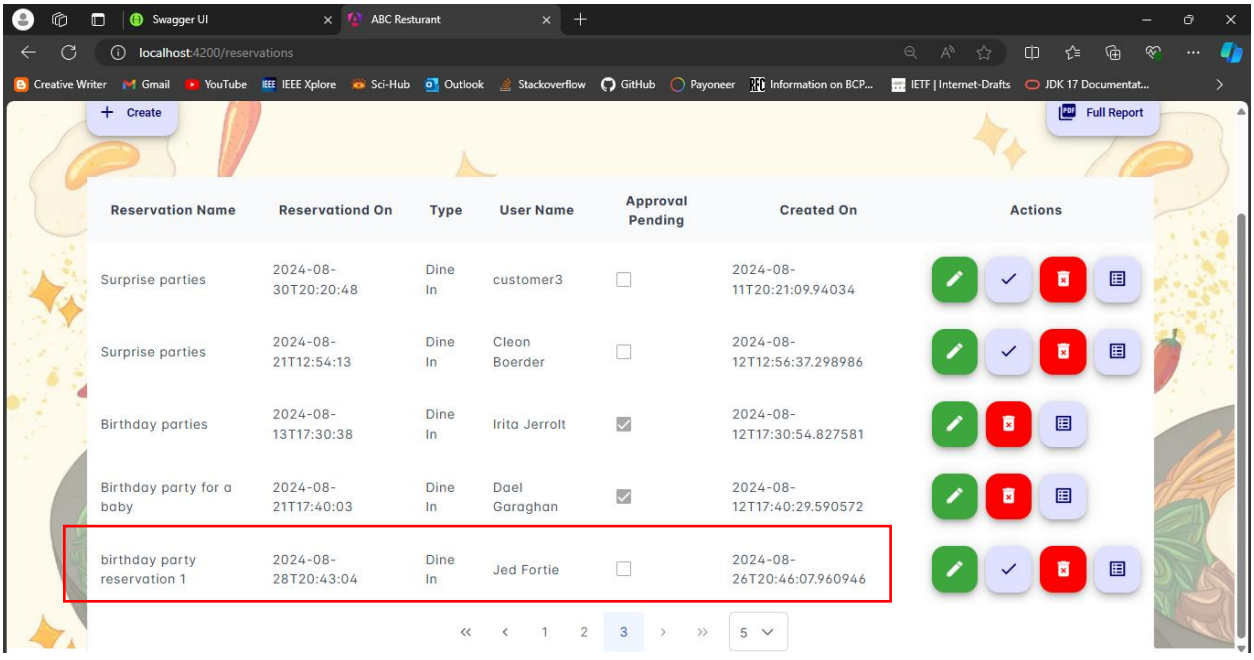
Reservation types are dine-in, delivery



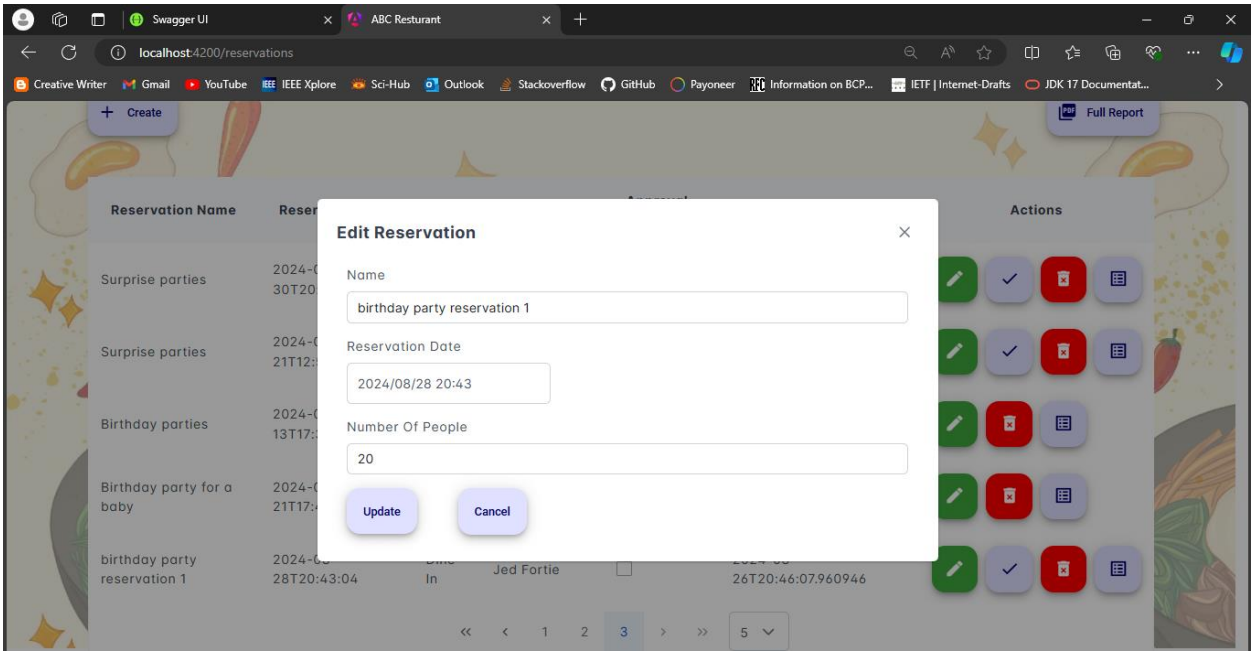


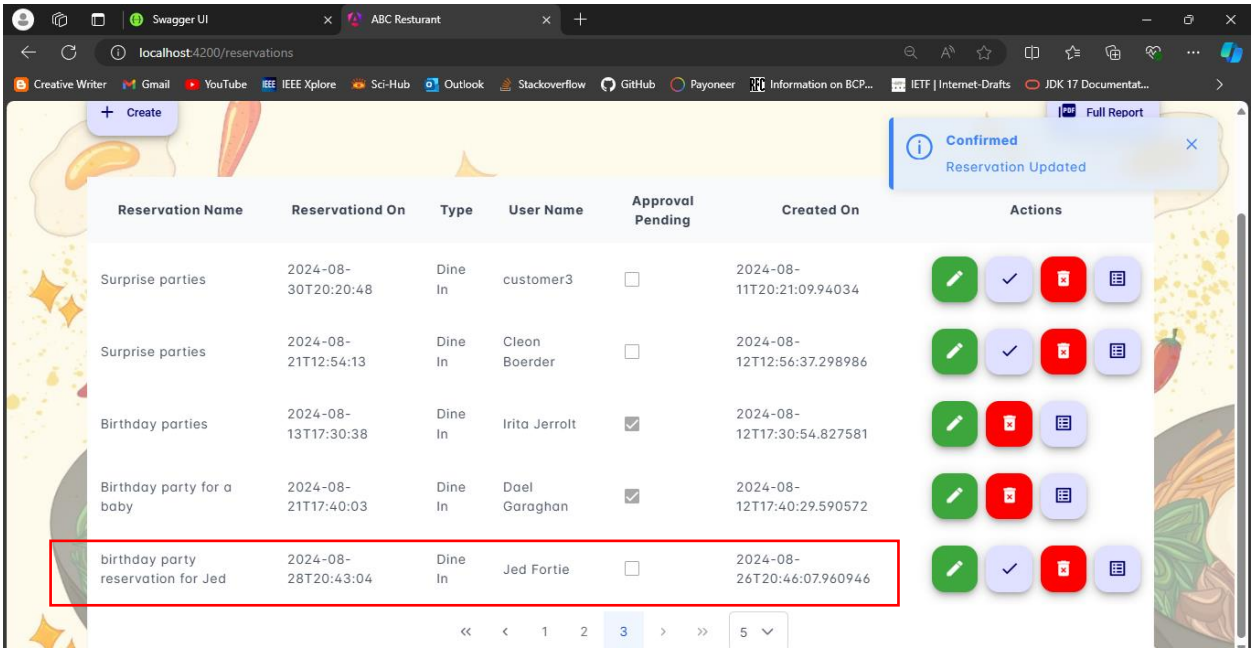
Notification is received – new reservation created.



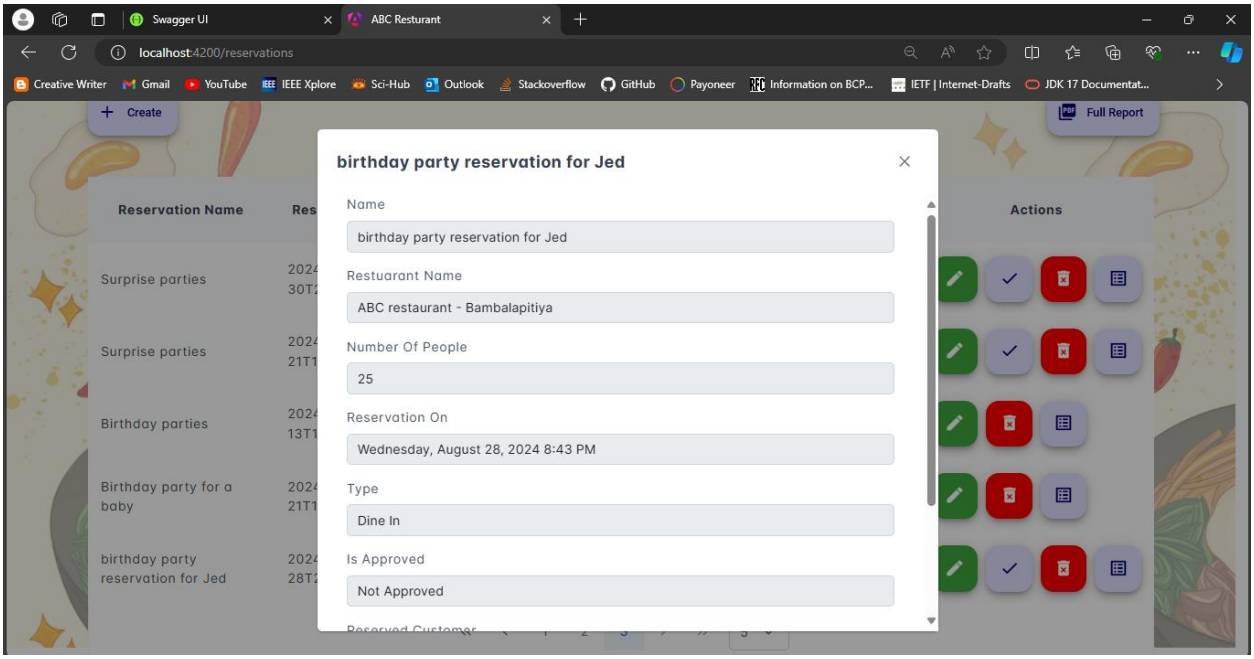


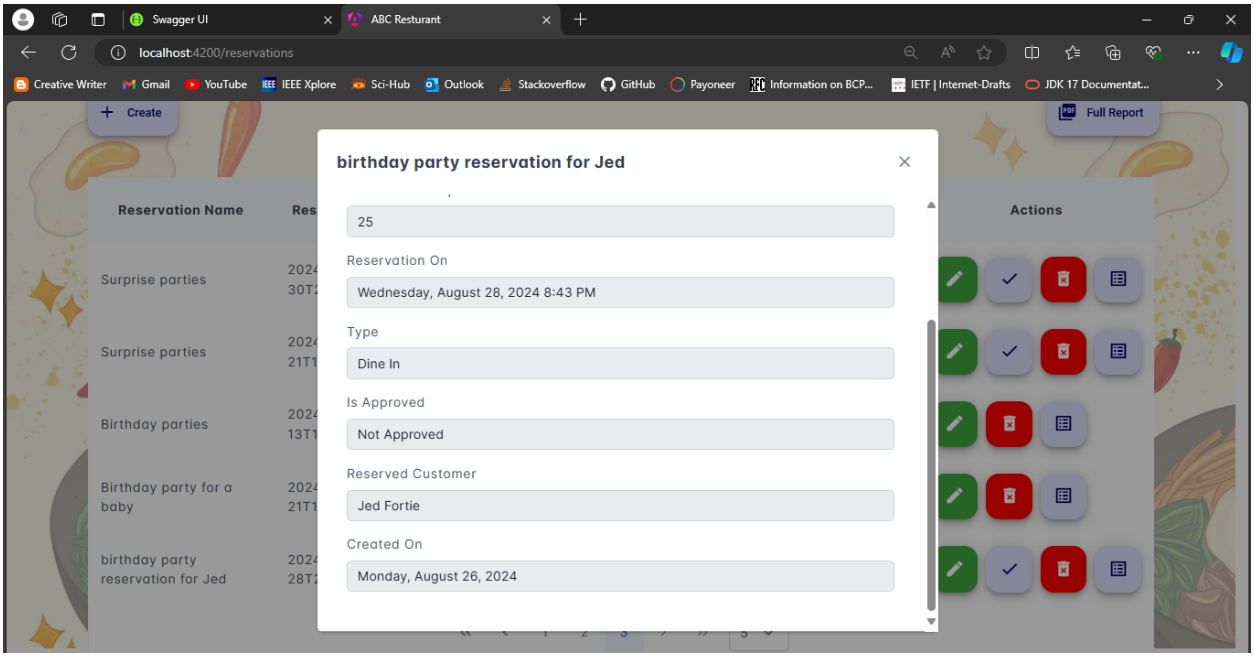
Edit reservation





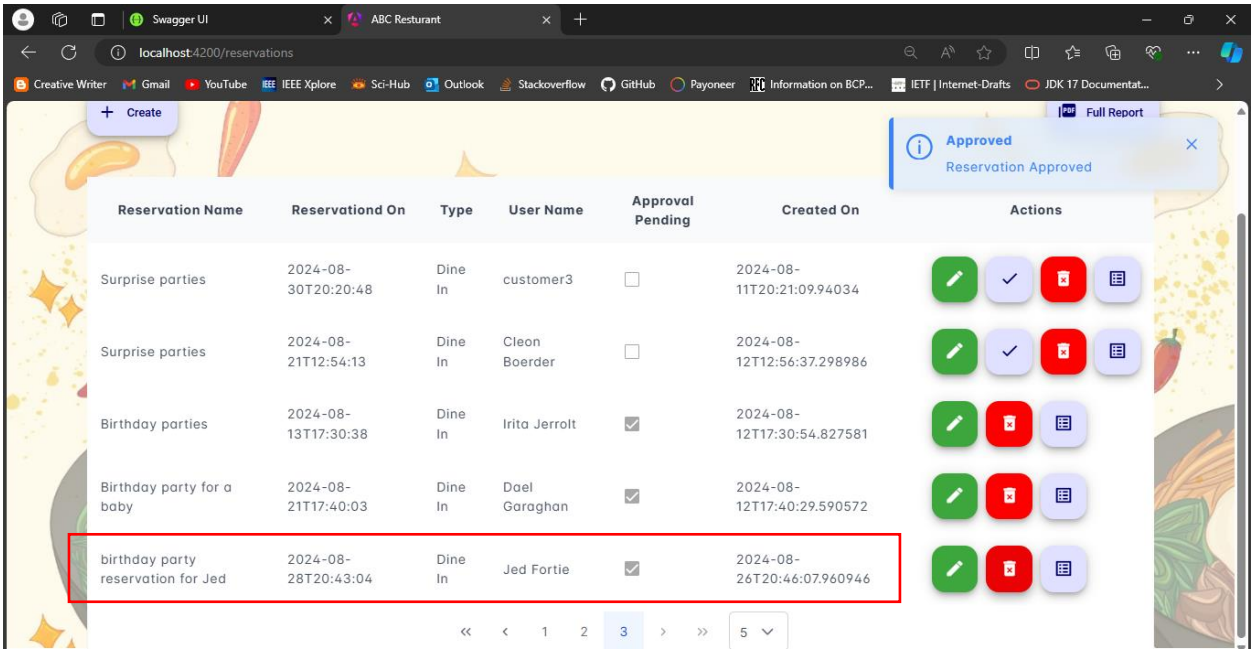
View reservation



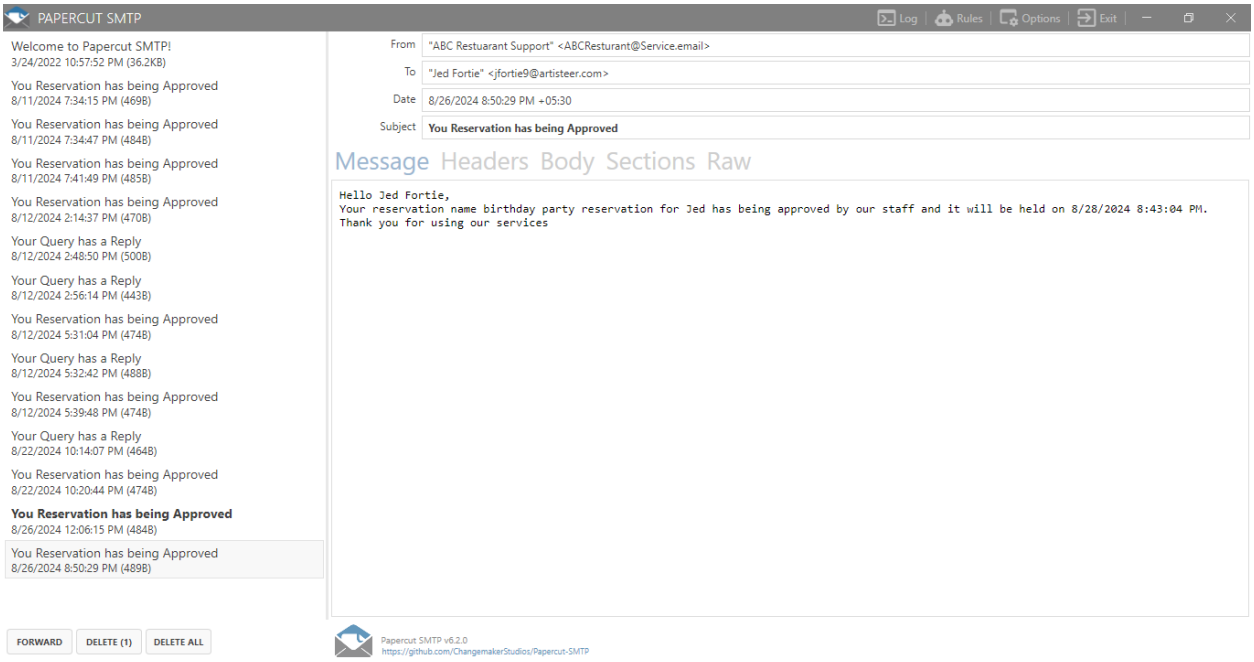


Confirm reservation

Once the confirm button is clicked under actions, the confirm button disappears and approval pending is ticked.

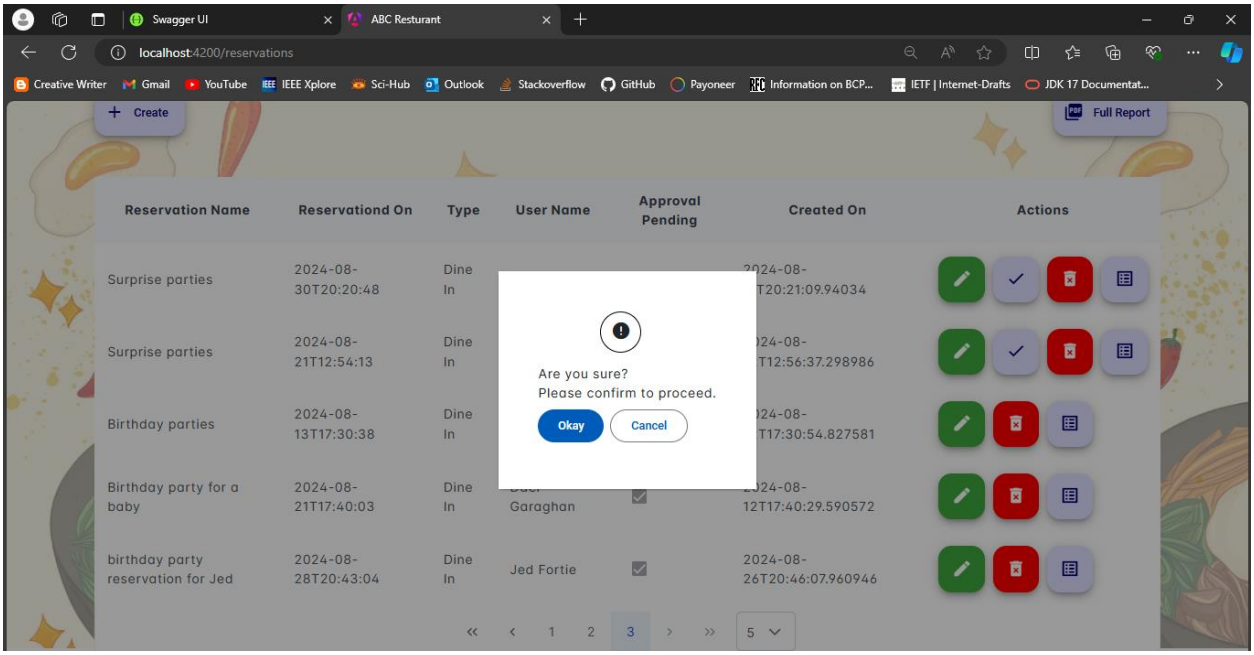


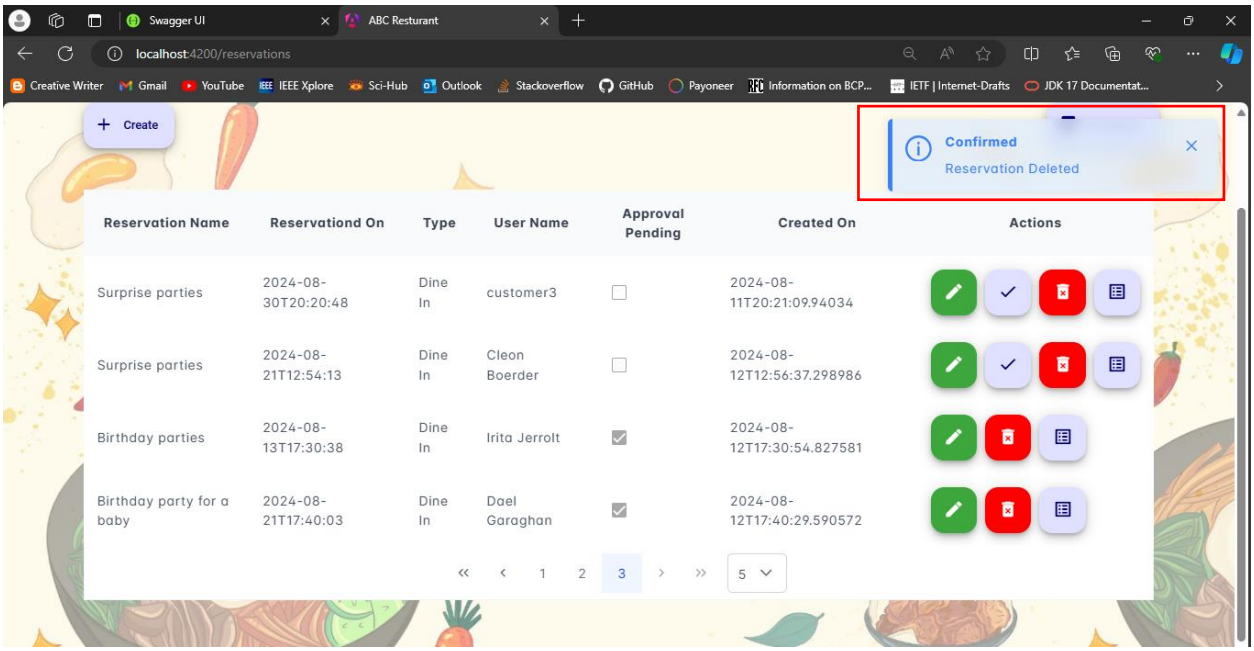
The confirmation email is sent



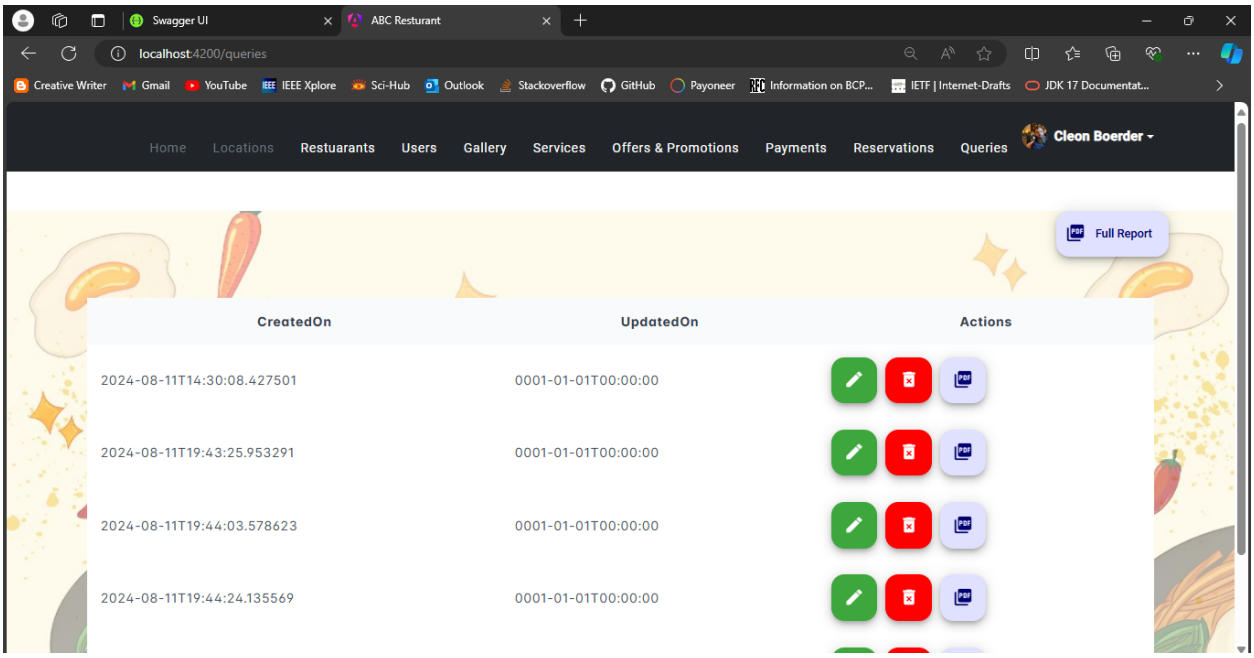
Delete reservation

Deleting reservation “birthday party reservation for Jed”



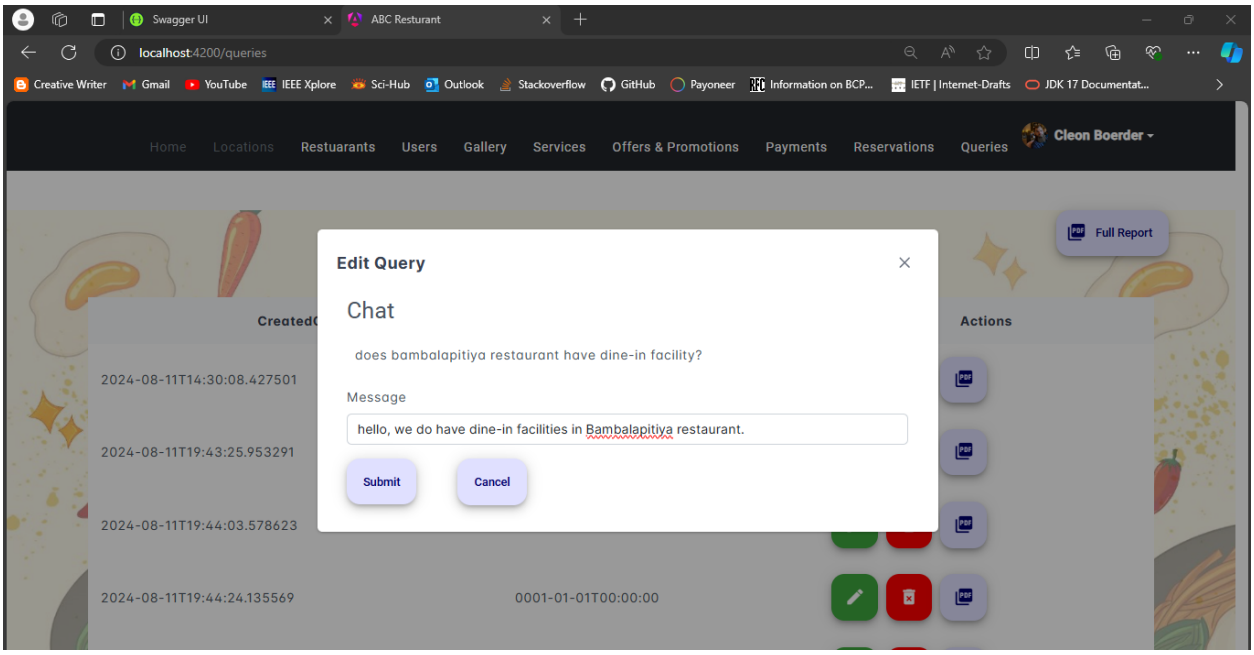


Query management

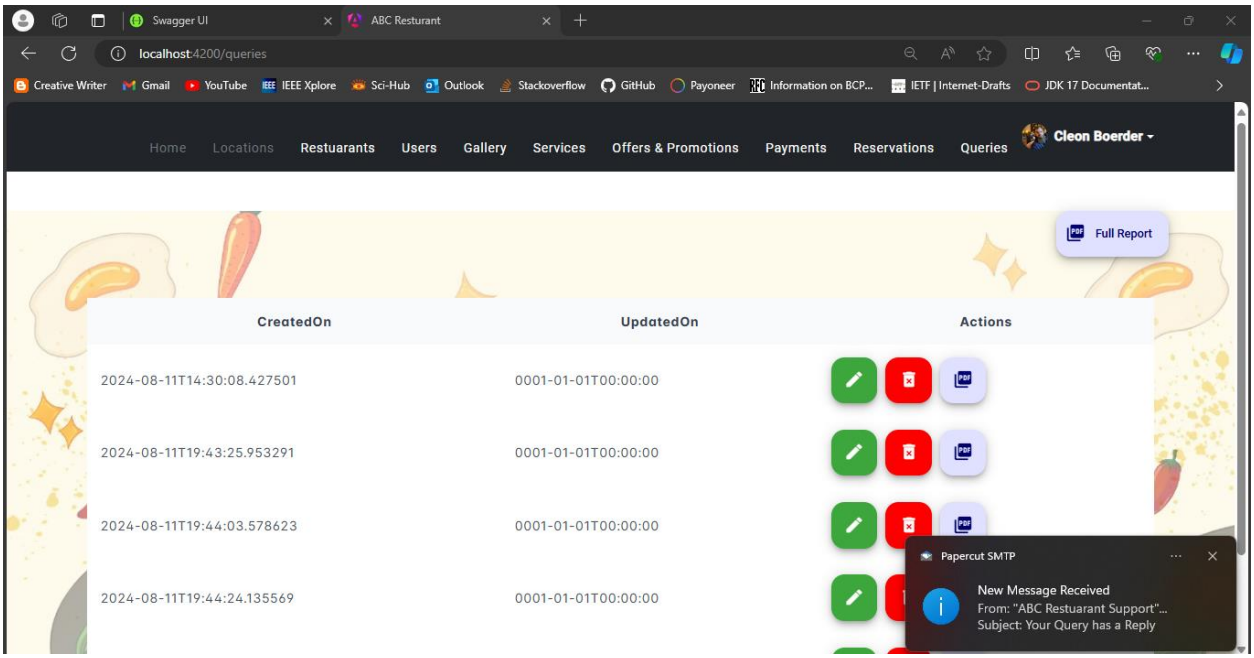


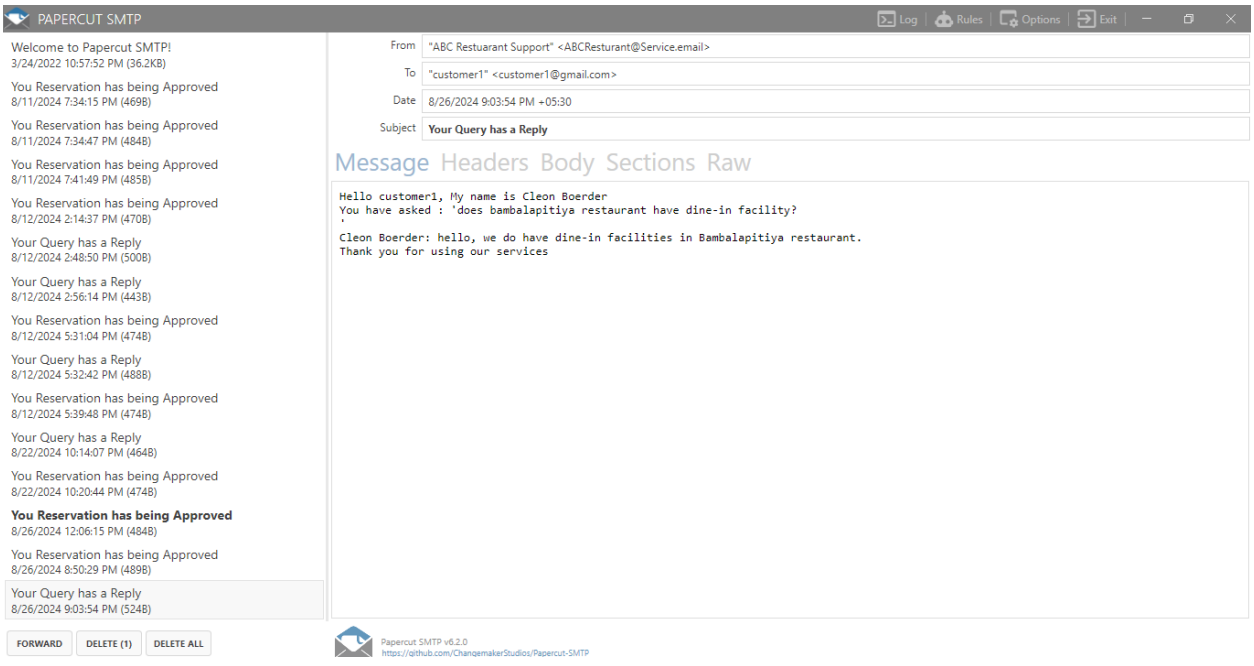
Submit answer to query

Once the edit button under actions is clicked, this pop up appears. Admin/staff can answer this query by typing a message on the message text field



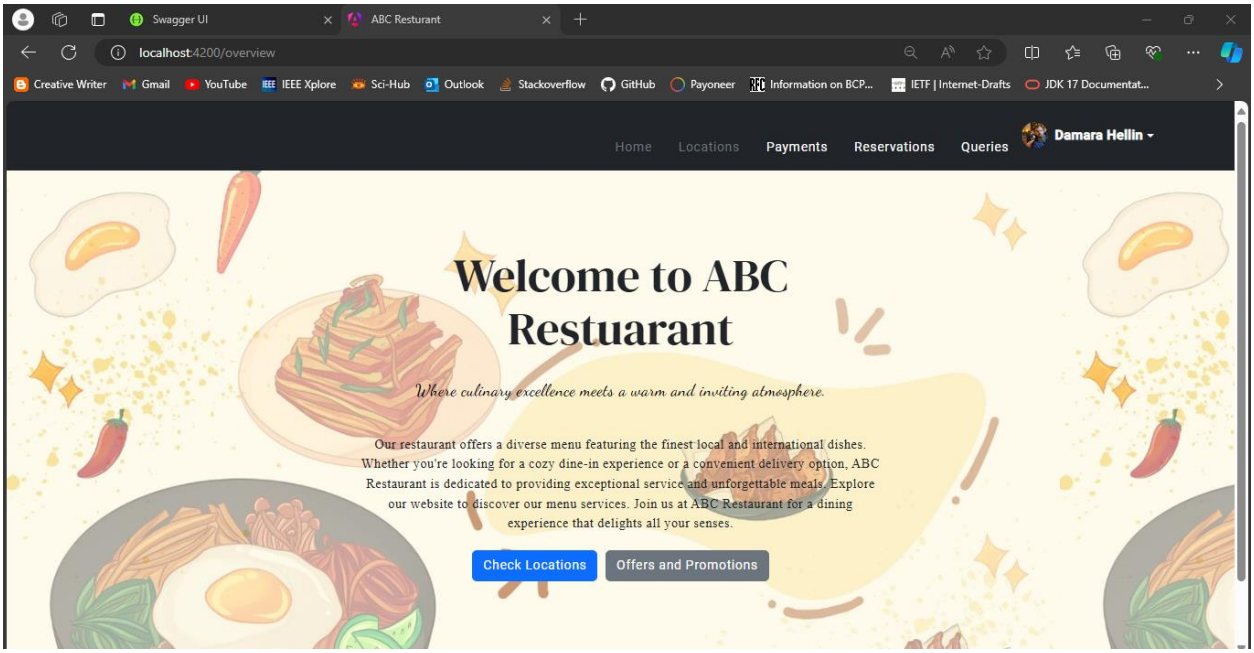
When the query is answered, an email is received.





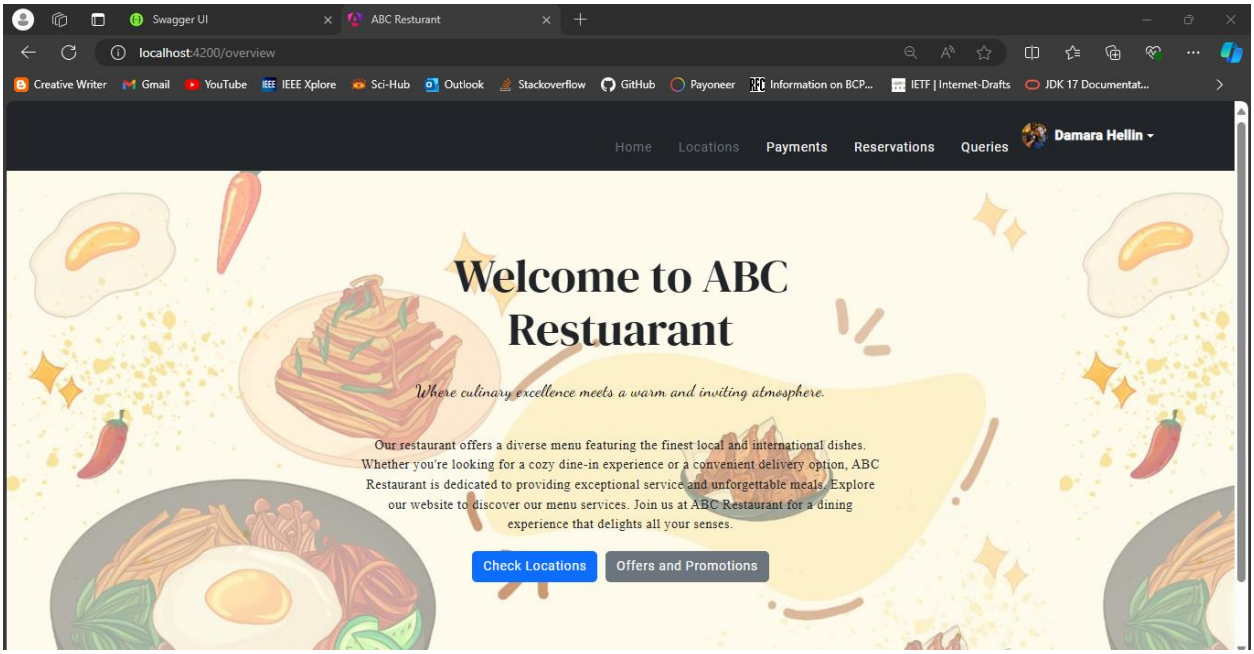
Interfaces for staff

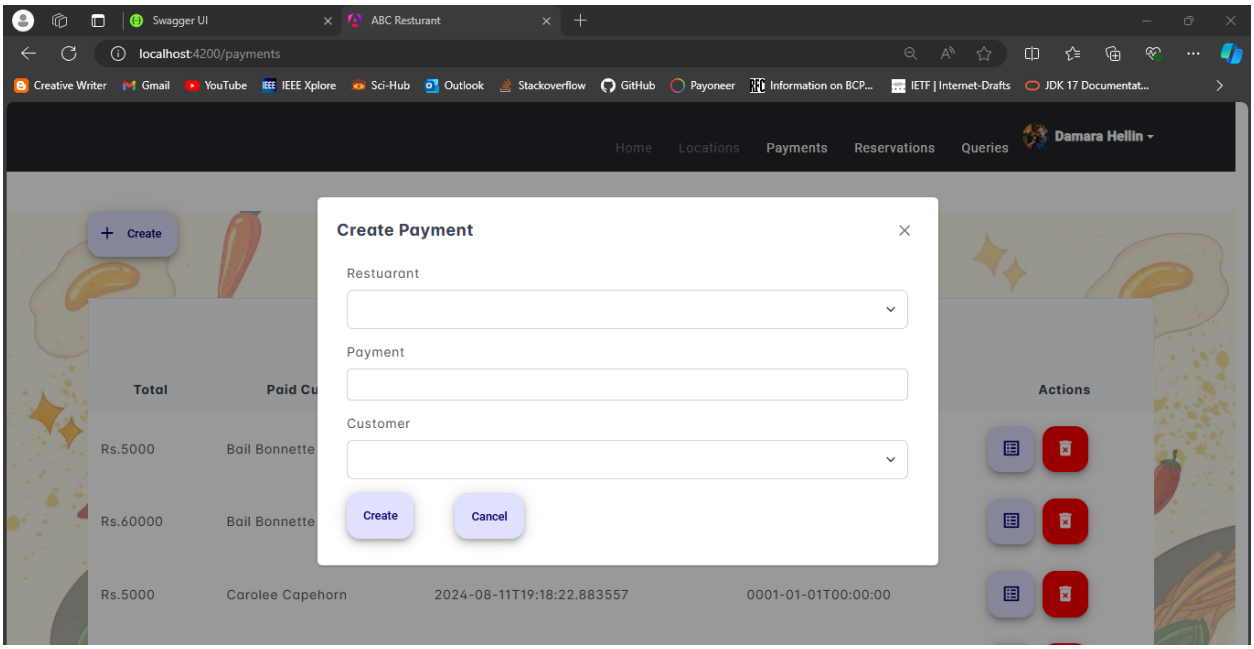
This is the homepage for staff after logging in

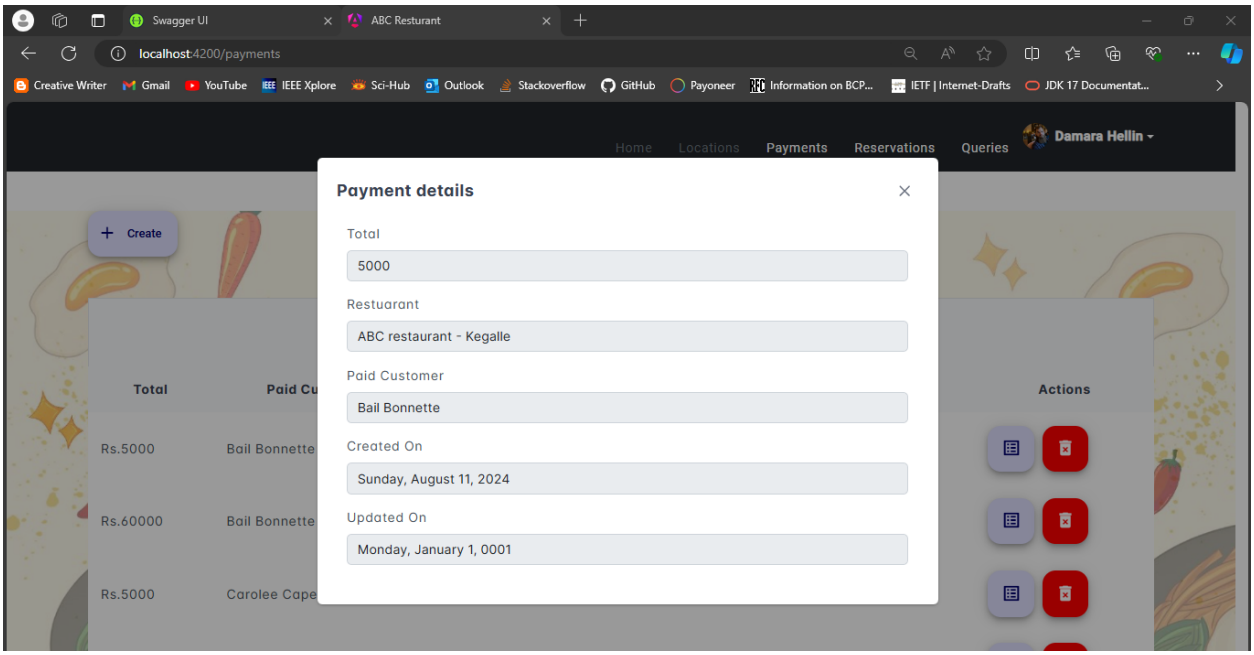


<u>Payment management</u>

Staff can create payments, view them and delete them.

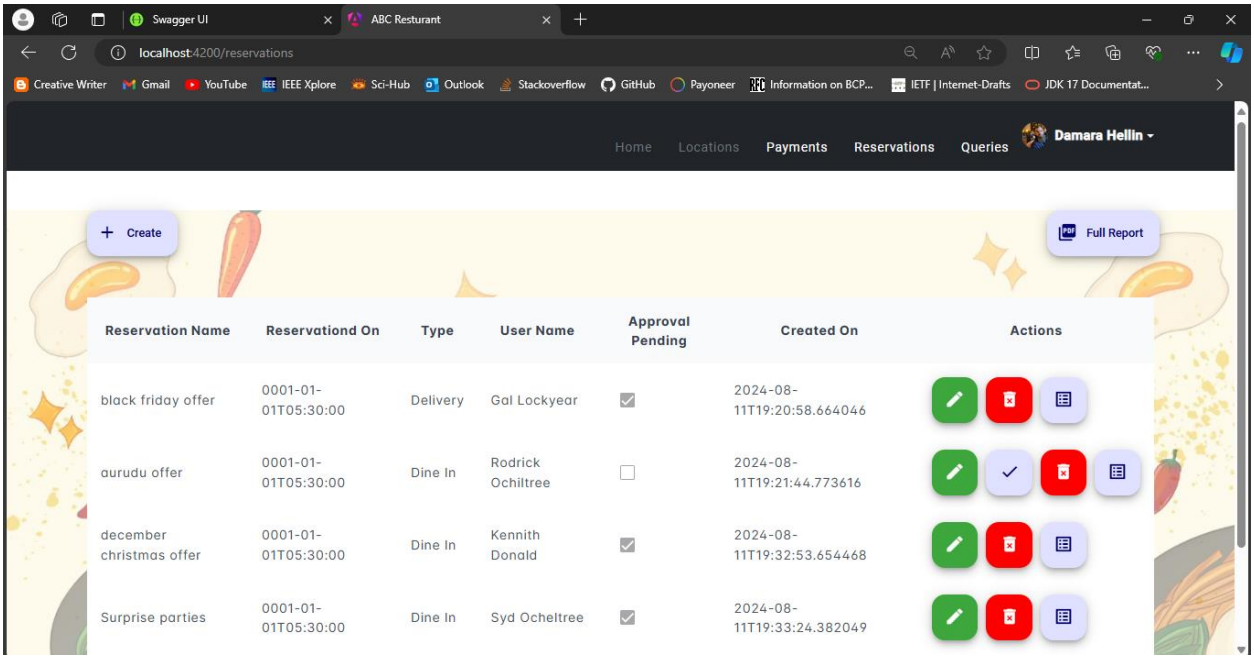


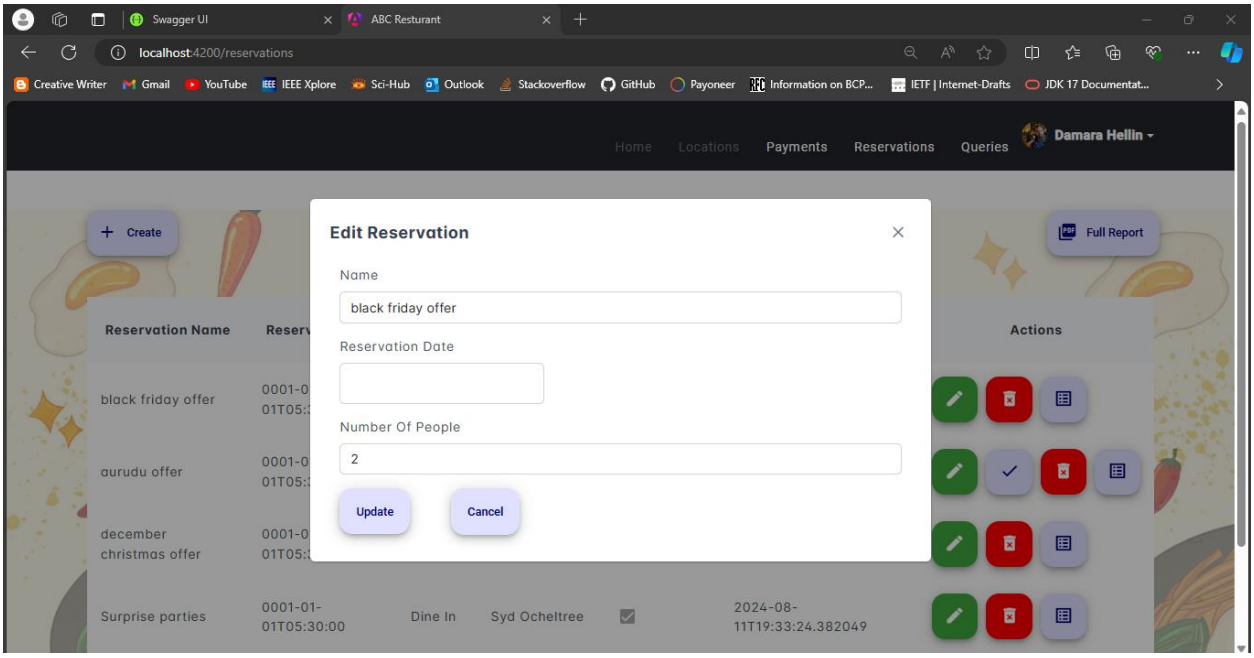


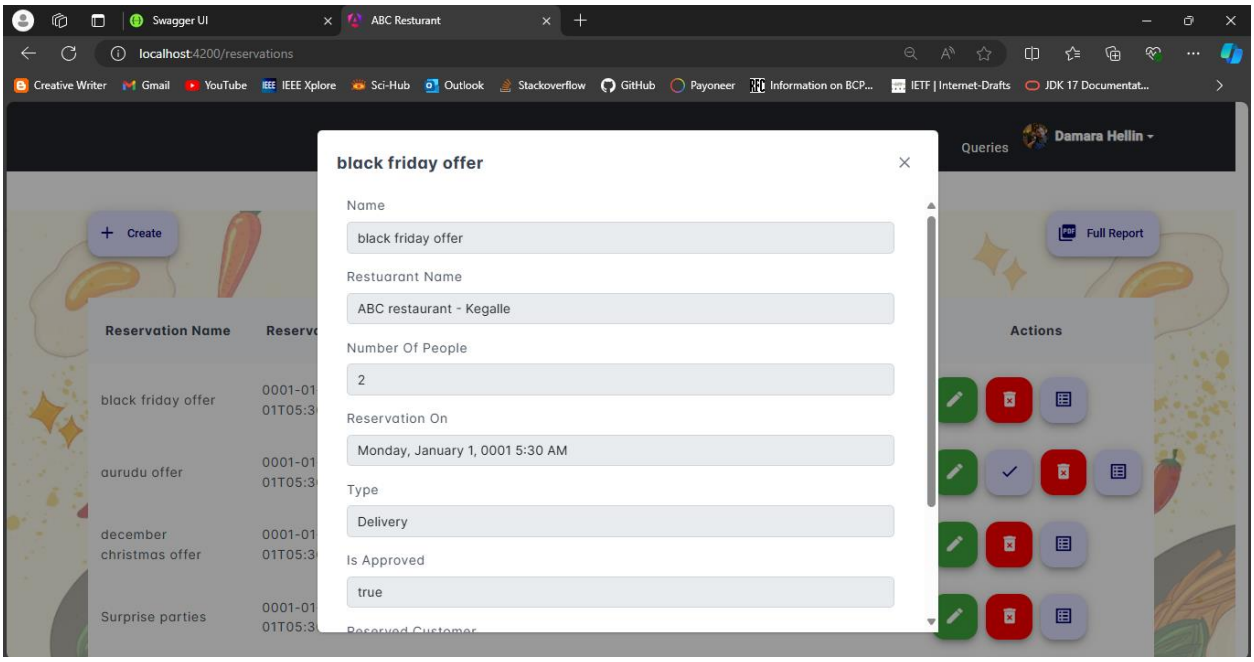


Reservation management

Staff can create reservations for customers, edit, approve, view and delete them like admin does. But staff can generate only the full reservation report.

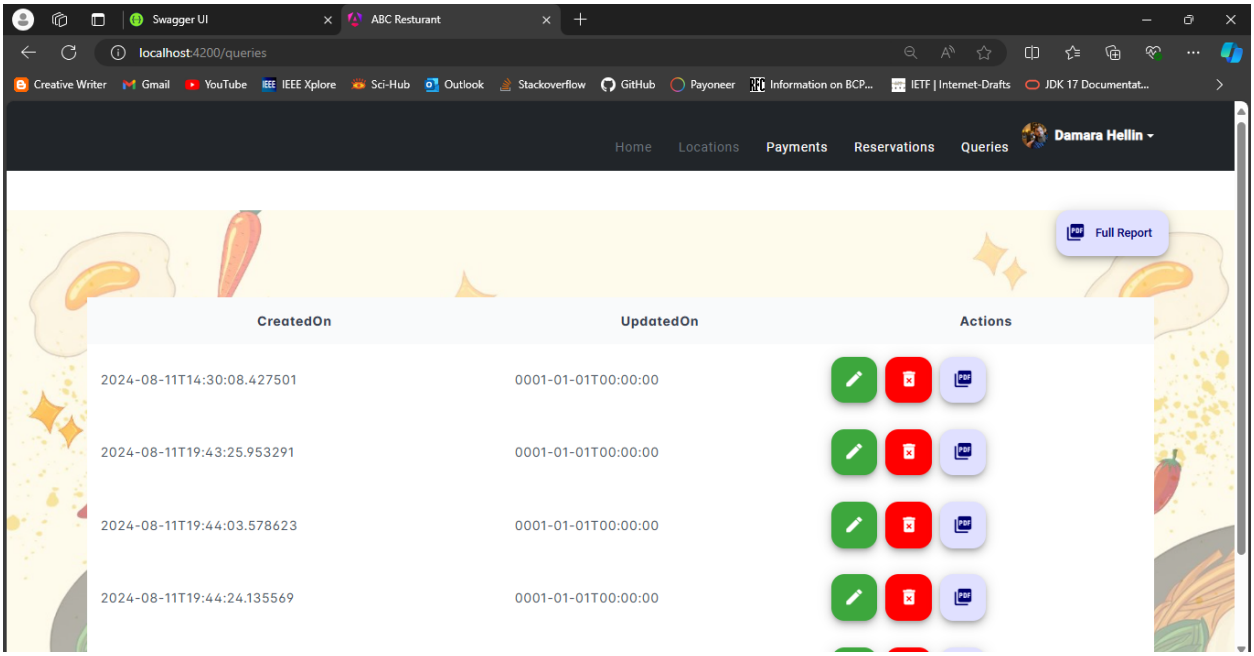

Swagger UI
ABC Resturant
localhost:4200/reservations
Creative Writer
Gmail
YouTube
IEEE Xplore
Sci-Hub
Outlook
Stackoverflow
GitHub
Payoneer
Information on BCP...
IETF | Internet-Drafts
JDK 17 Documentat...
Home
Locations
Payments
Reservations
Queries
Damara Hellin
Create
Full Report
Reservation Name
Reservationd On
Type
User Name
Approval Pending
Created On
Actions
black friday offer
0001-01-01T05:30:00
Delivery
Gal Lockyear
2024-08-11T19:20:58.664046
aurudu offer
0001-01-01T05:30:00
Dine In
Rodrick Ochiltree
2024-08-11T19:21:44.773616
december christmas offer
0001-01-01T05:30:00
Dine In
Kennith Donald
2024-08-11T19:32:53.654468
Surprise parties
0001-01-01T05:30:00
Dine In
Syd Ocheltree
2024-08-11T19:33:24.382049


Swagger UI
ABC Resturant
localhost:4200/reservations
Home
Locations
Payments
Reservations
Queries
Damara Hellin
Create
Full Report
Edit Reservation
Name
black friday offer
Reservation Date
Number Of People
2
Update
Cancel
Reservation Name
Actions
black friday offer
aurudu offer
december christmas offer
Surprise parties
0001-01-01T05:30:00
Dine In
Syd Ocheltree
2024-08-11T19:33:24.382049



Query management

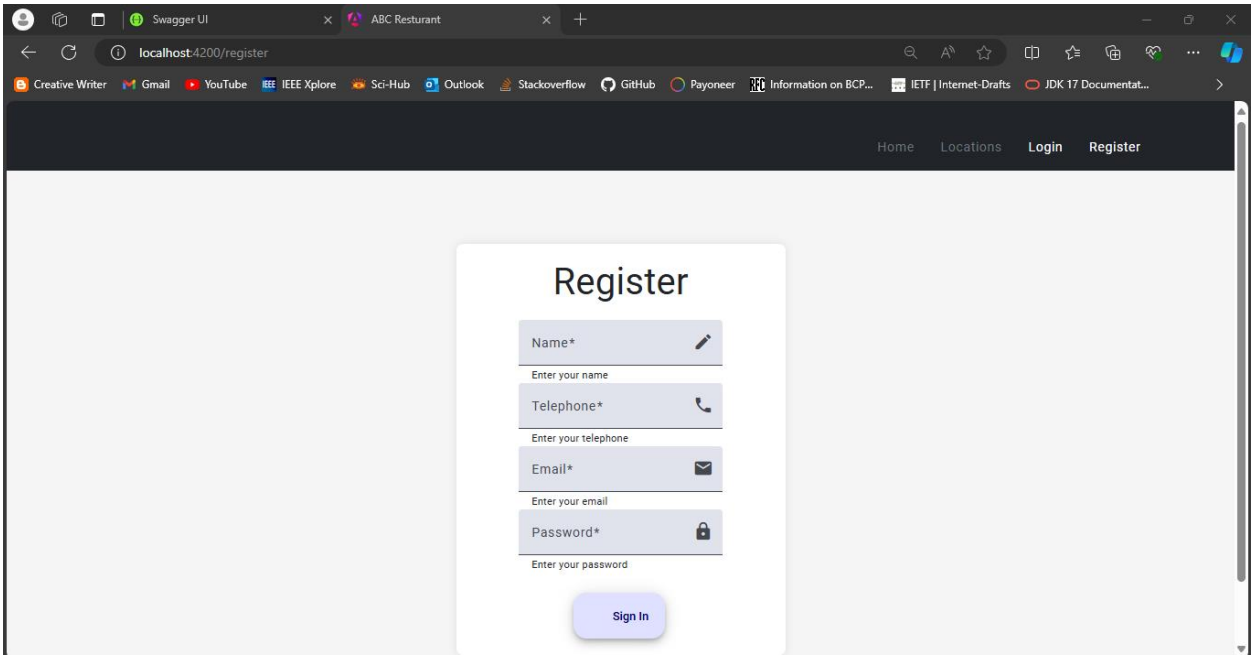
Staff can respond to queries, delete them and generate full report and query report per customer.



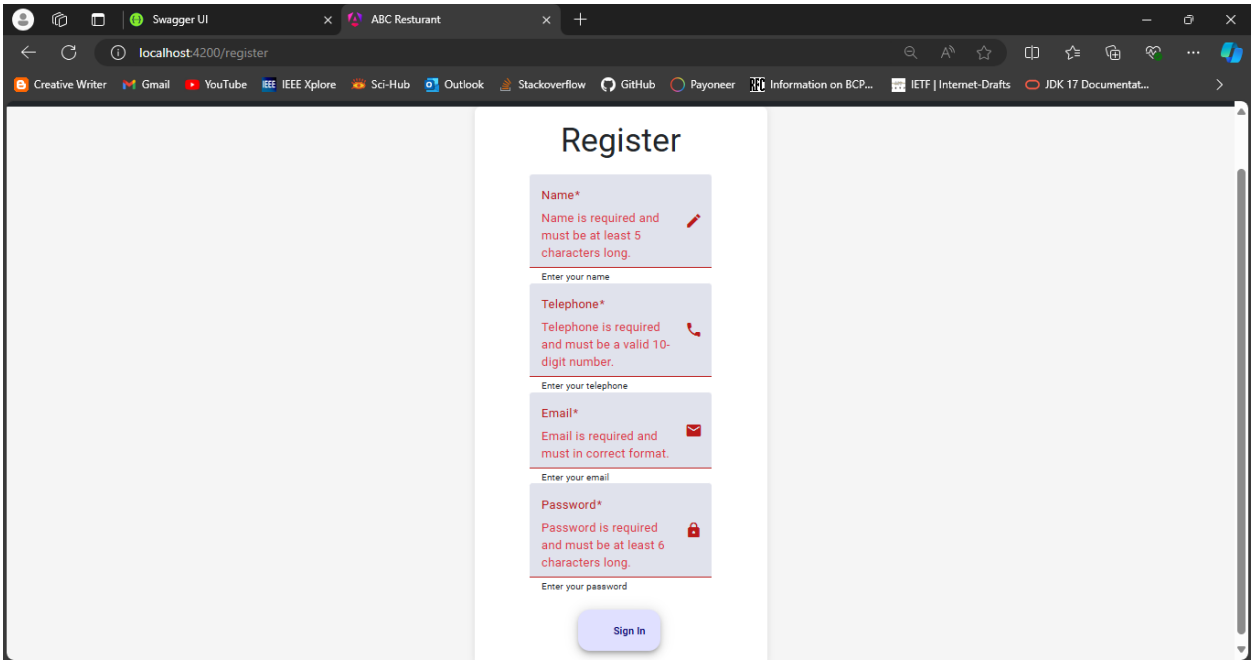
Interfaces for customer

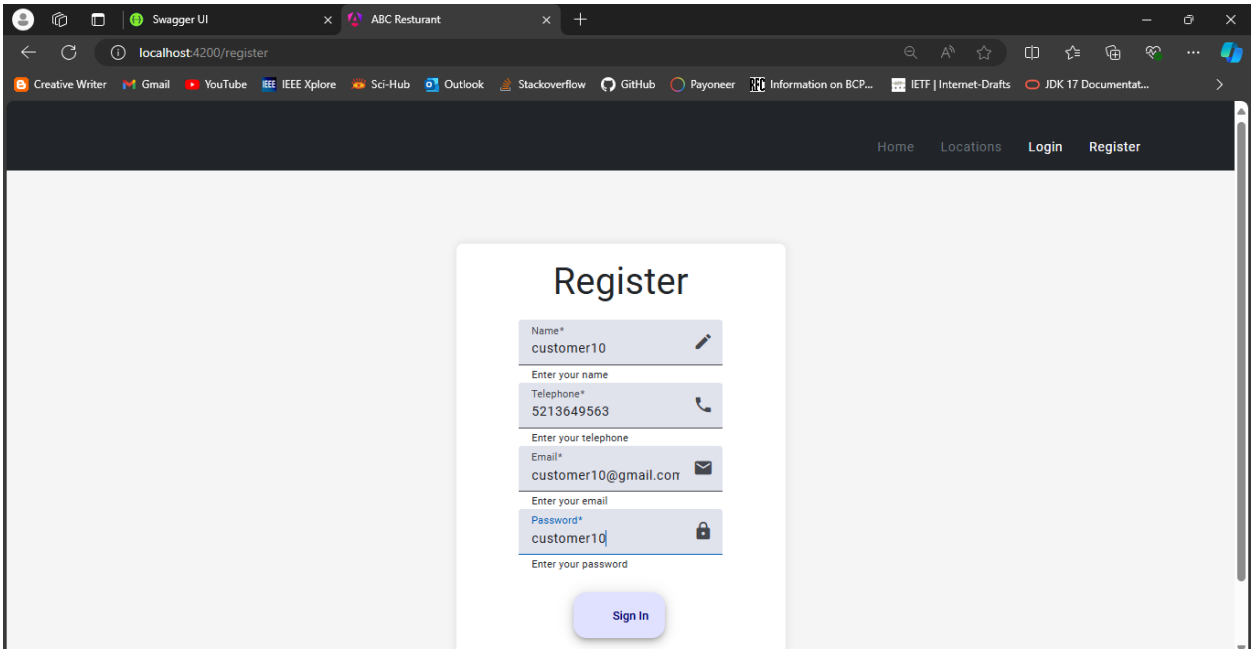
Register

Only customers can register themselves into the system

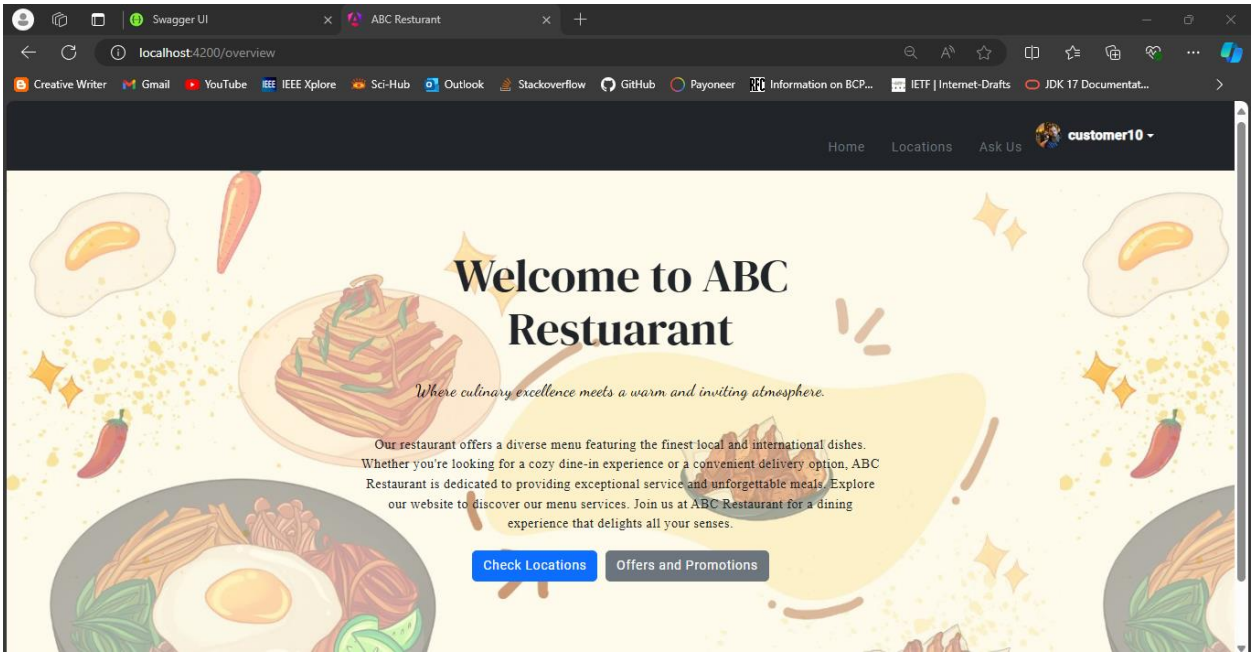


Form validations

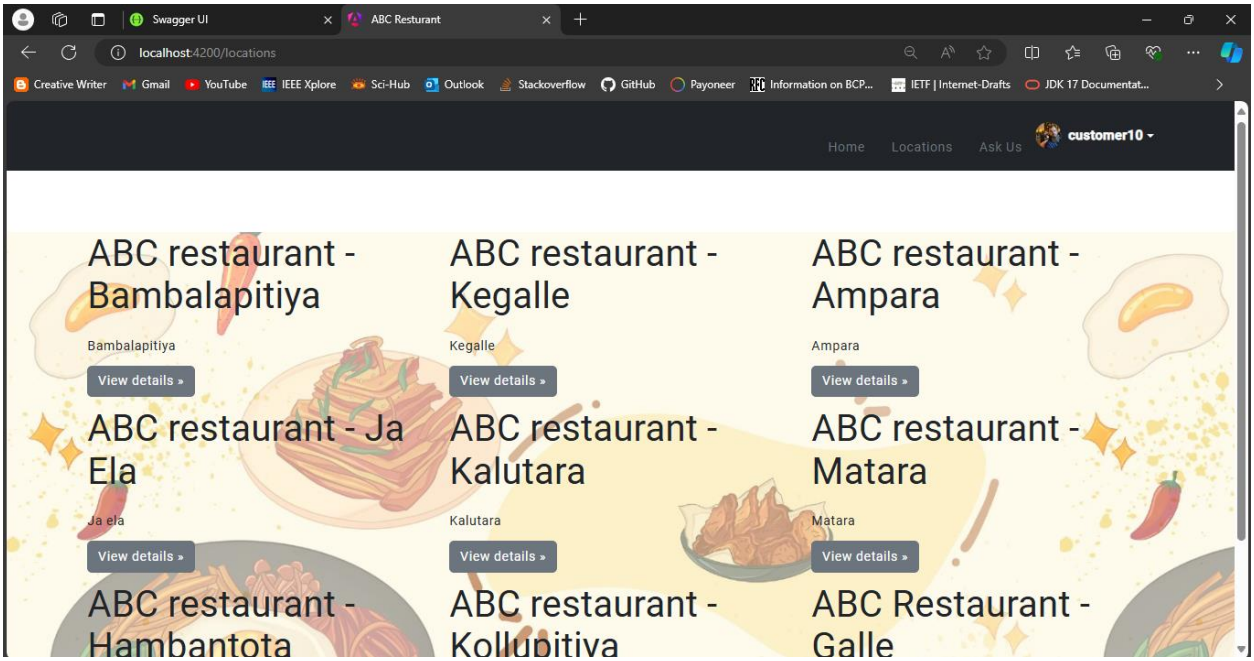


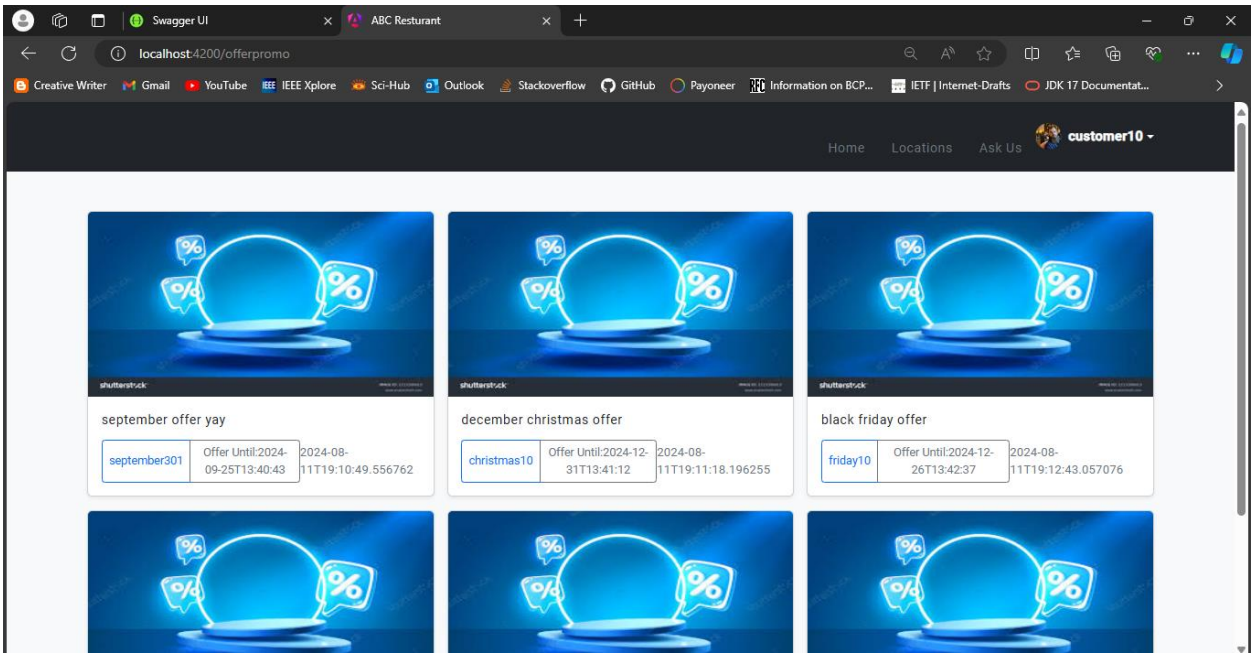


Upon registering, customer will be directed to home page



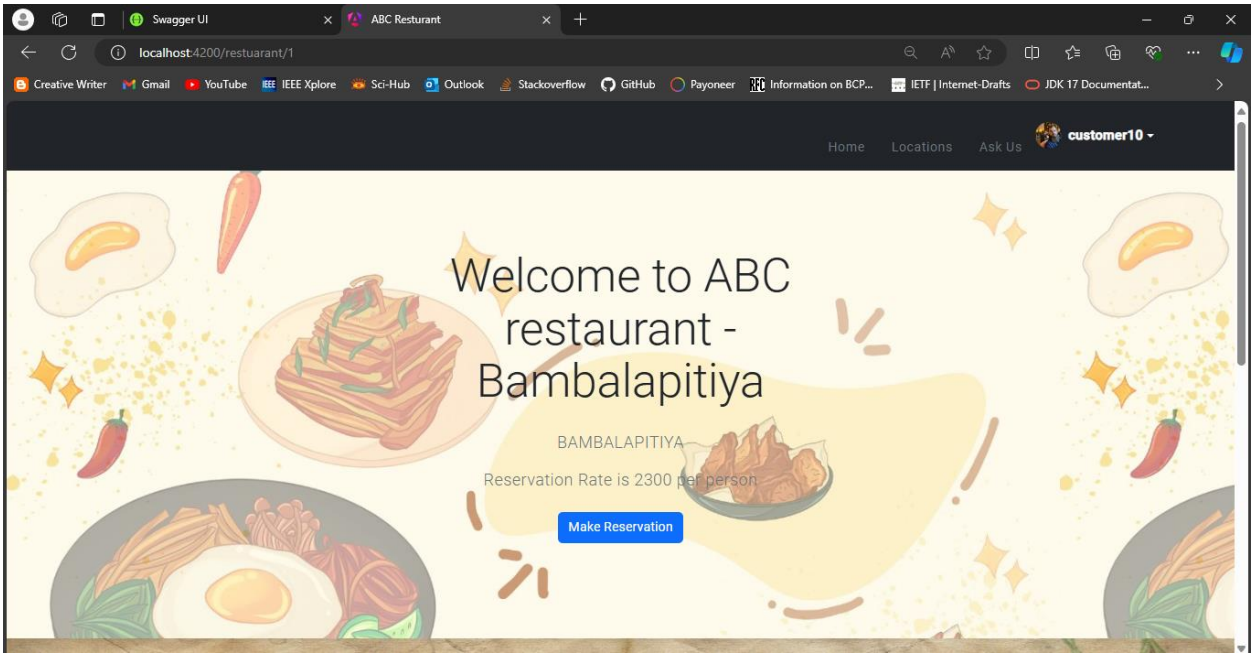
They can view locations and check offers and promotions



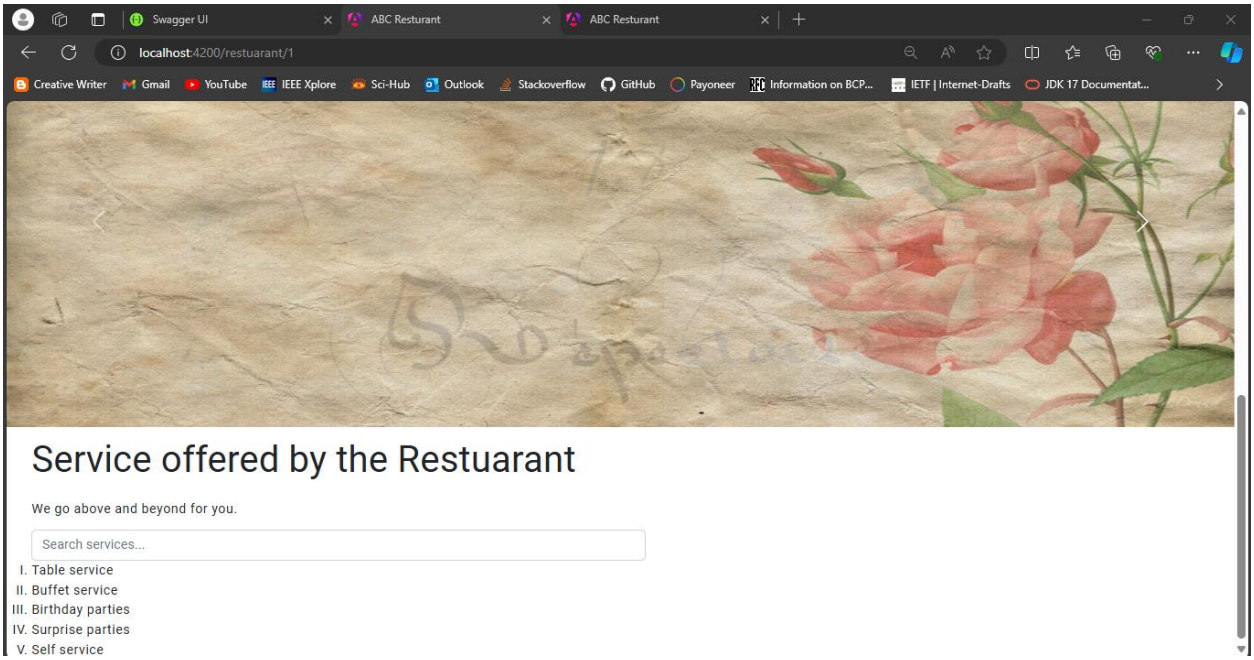


Make reservation

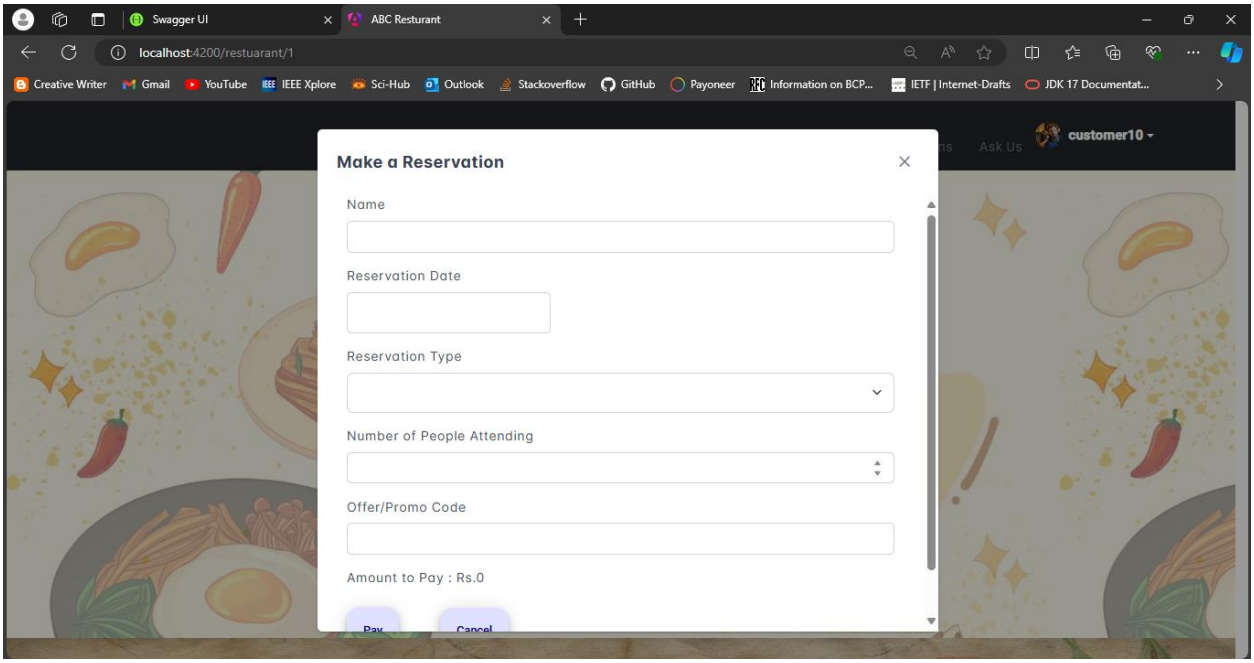
Customers can select a restaurant from locations and click on view details. Then they will be directed to the welcome page of the particular restaurant



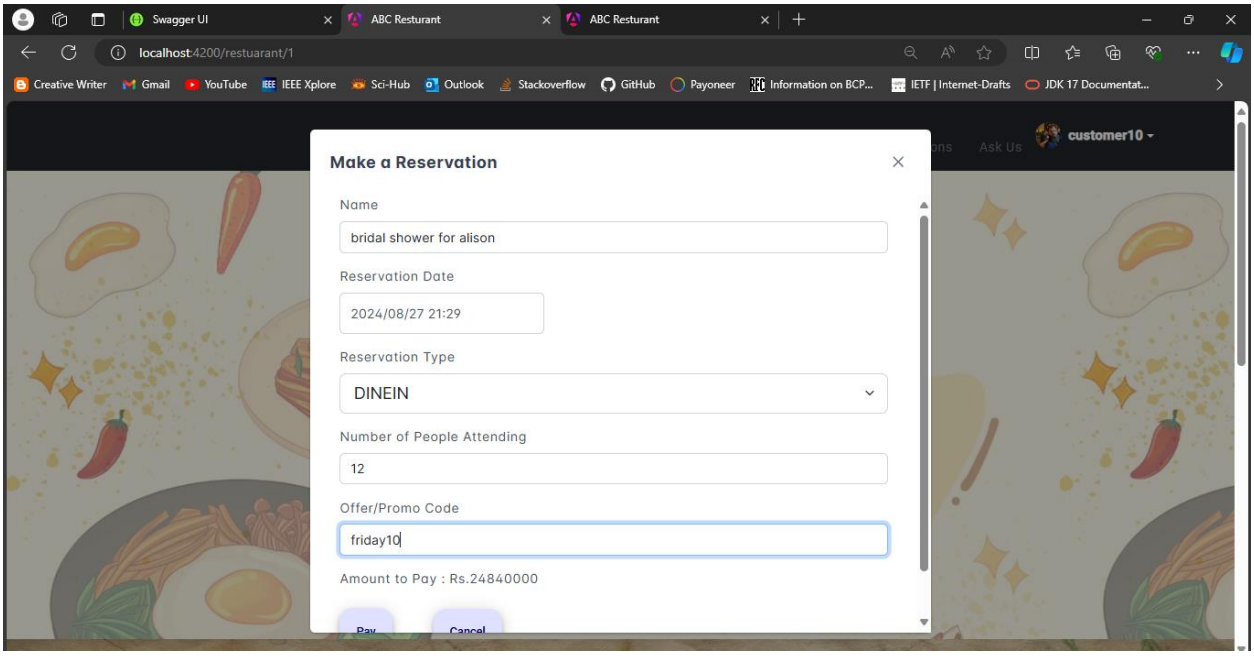
Customers can browse through services offered by restaurant.



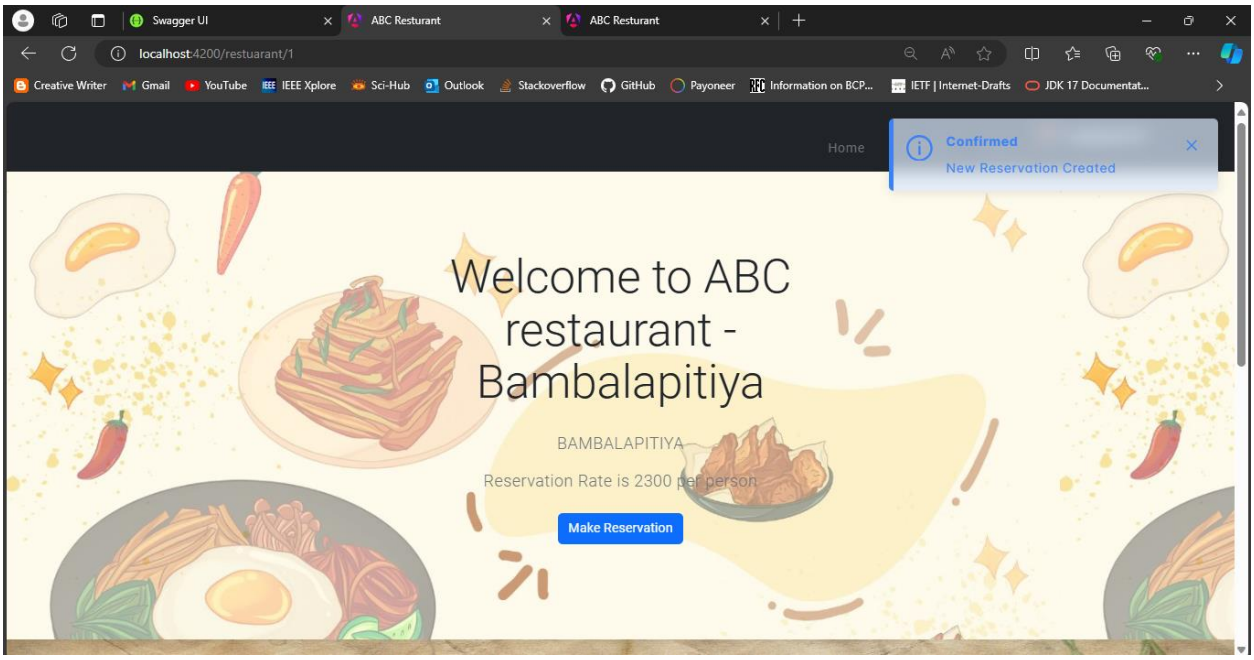
Once clicked on make reservation, this pop up appears.



When filling the form, if the customer has a promo code, they can enter it to receive a discount.



Upon making the reservation, the confirmation notification appears

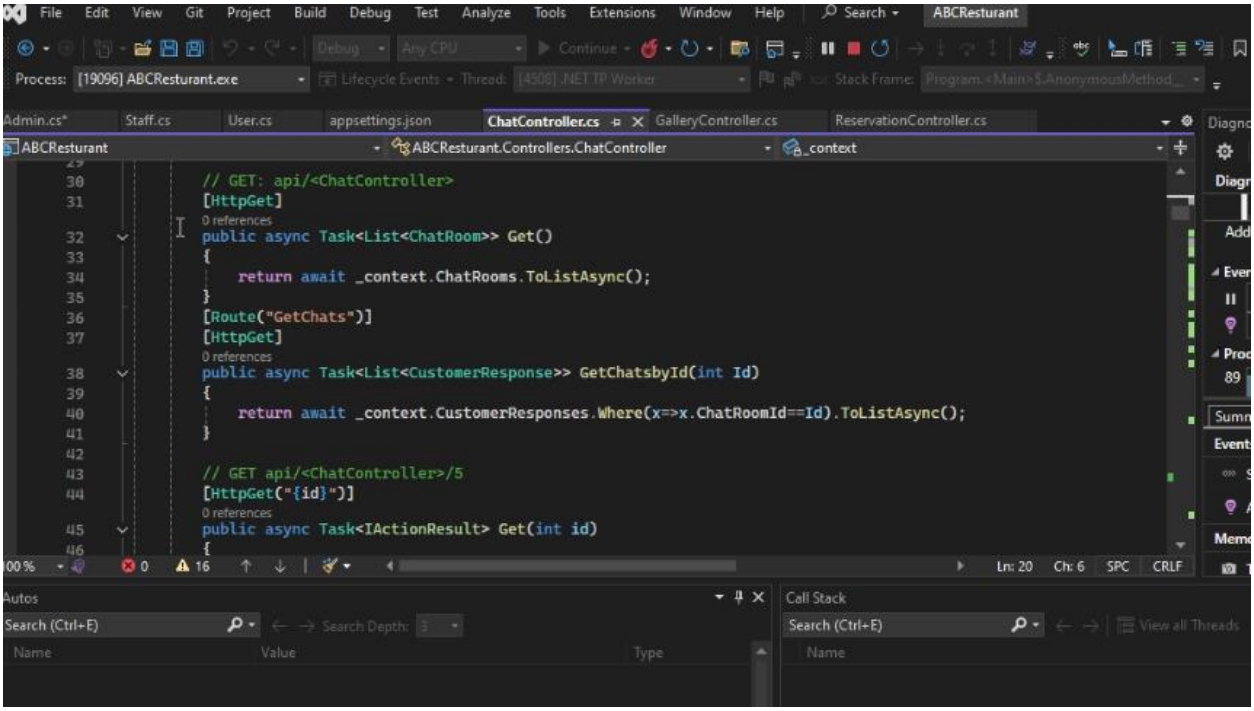


Use of Design Patterns

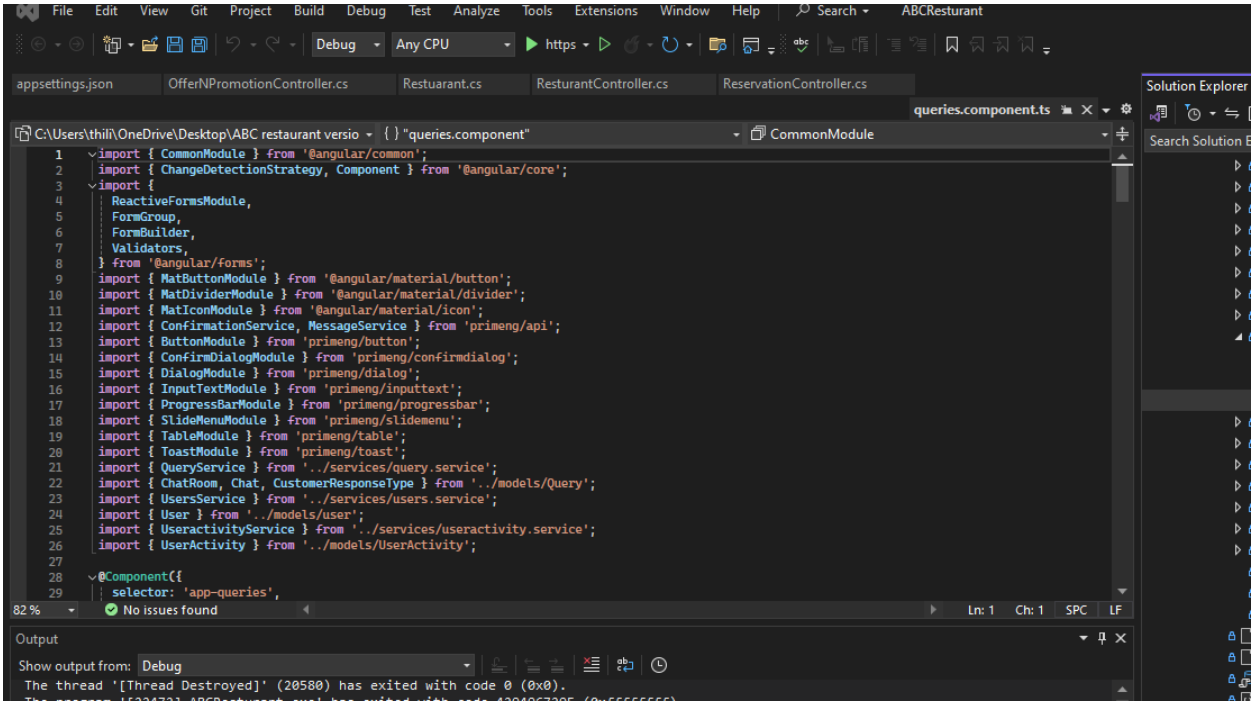
<u>MVC design pattern</u>

In here, we send data into every view through a controller, and those data are bound into a model and shown in view. One controller can provide multiple views and in one view more than one model is sometimes used.

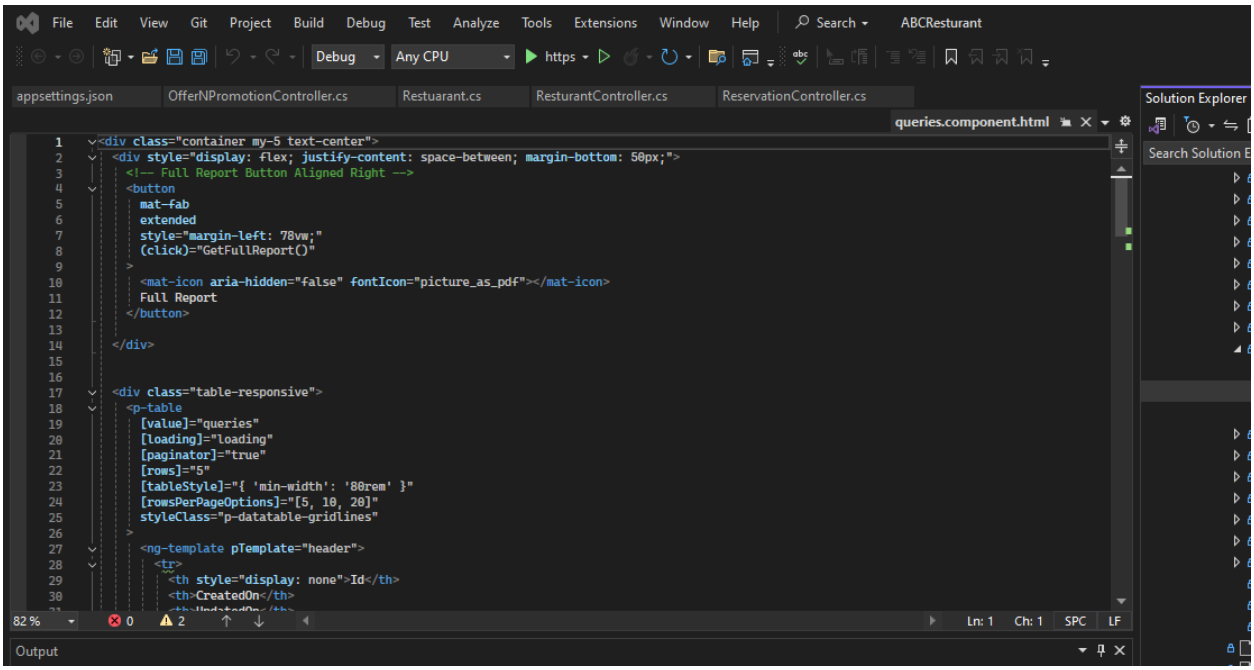
The model here shown is ChatRoom



Then we bind the model ChatRoom to queries.component.ts component in the front-end

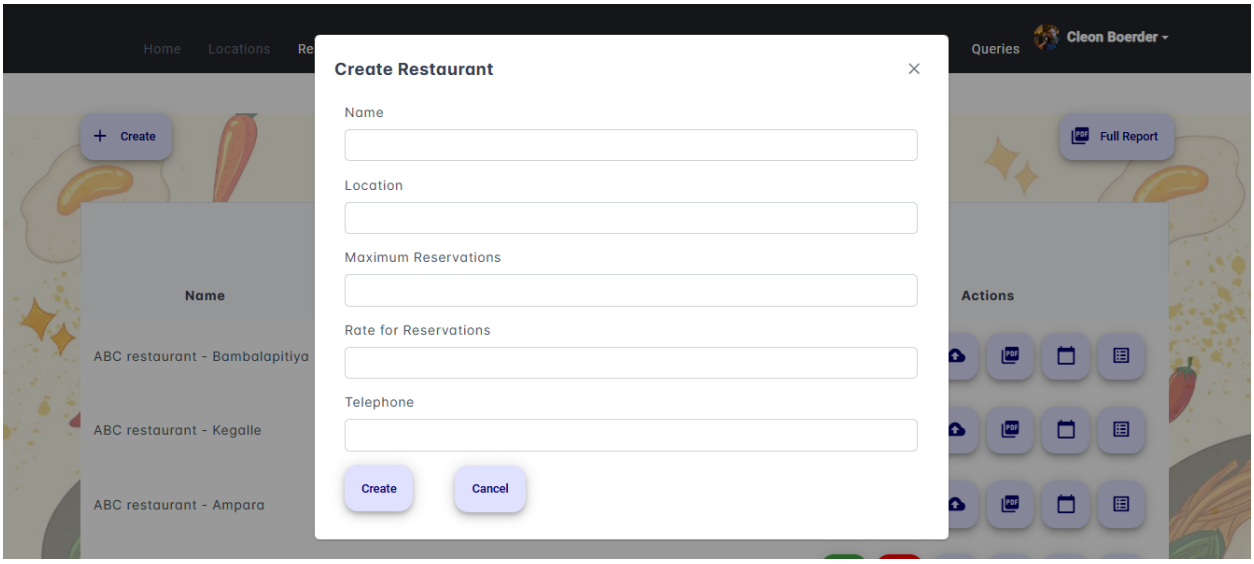


And then data that is bound are shown through the queries.component.html which acts as view.

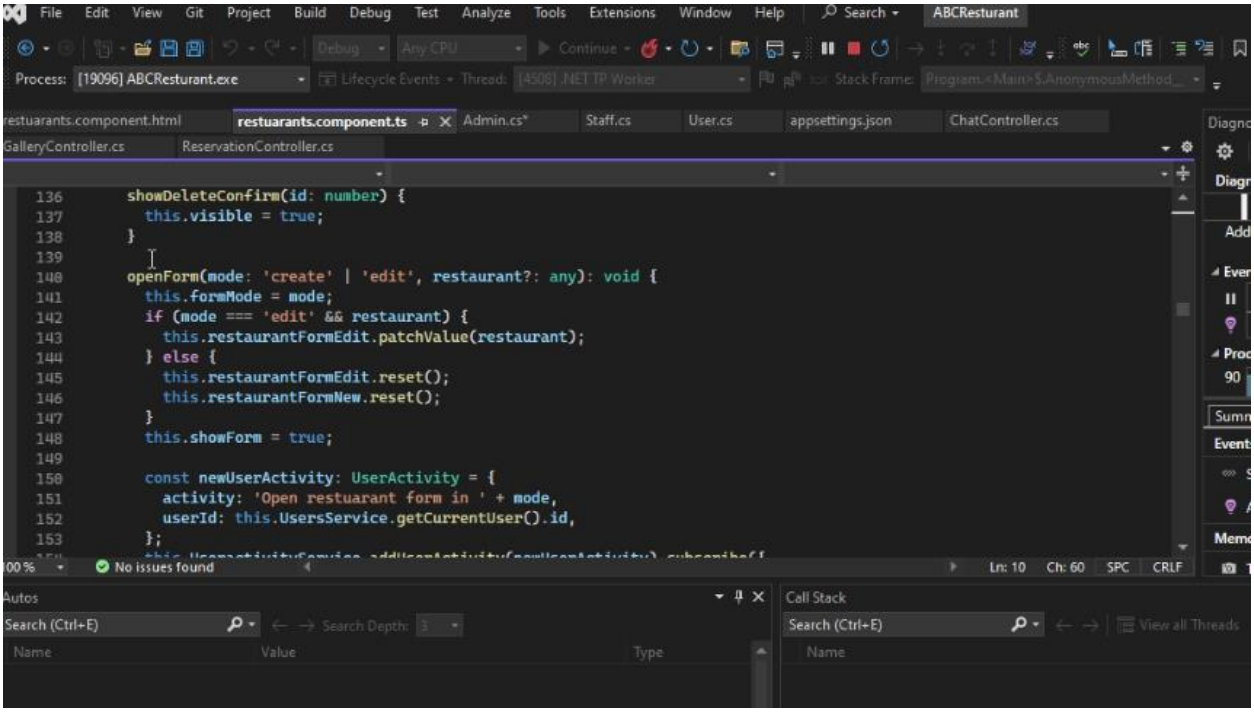


Observer design pattern

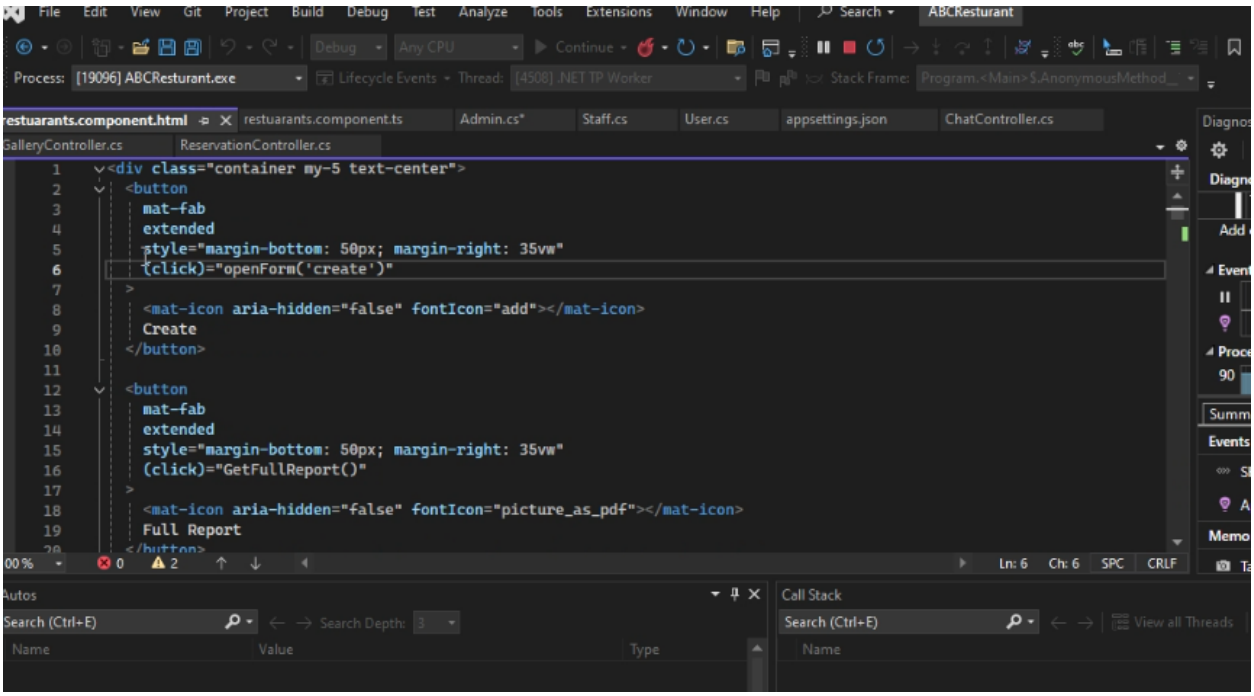
As an example, in this project for every button you click, an event is bound to it. As an example for that when you click the create restaurant button, a pop up appears.



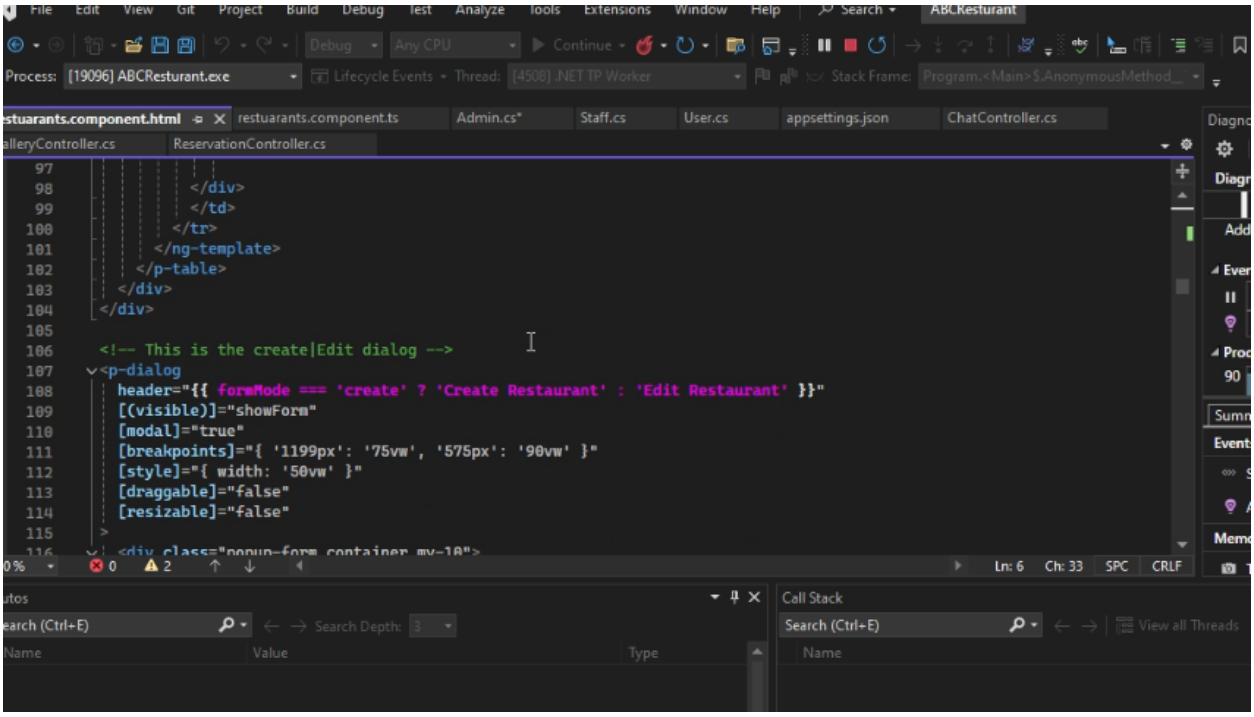
This works with the openForm function which works with both create and edit options in this case.



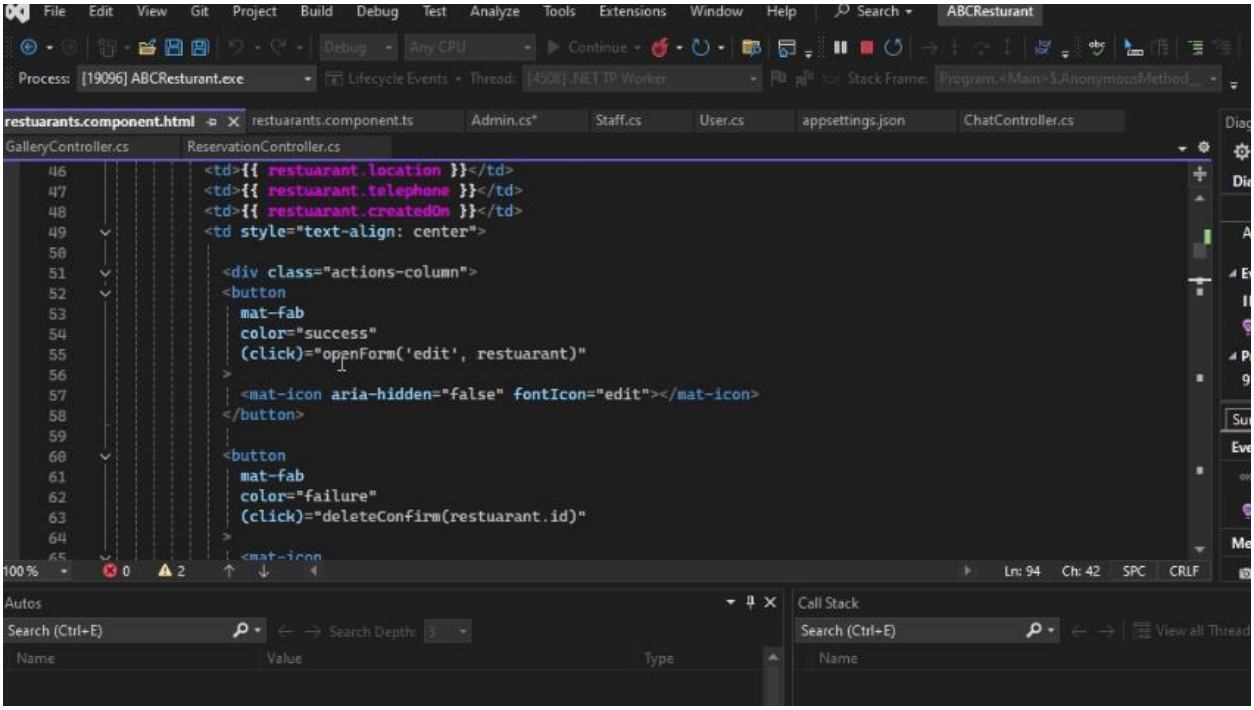
Following is the create button. With the openForm function, create parameter is passed here.



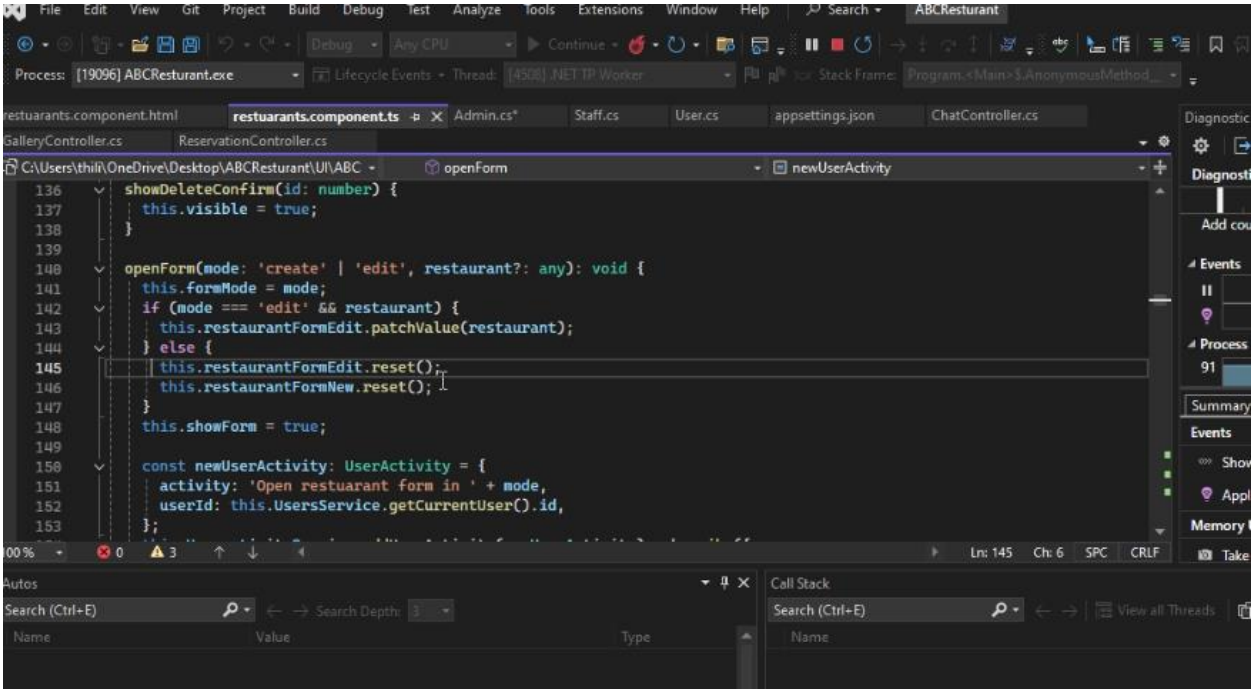
Once the button is clicked, it creates an event listener to respond with a pop up. When it gets true the pop up appears.



Then it determines whether it's an edit or a create. The following image shows the same function for edit with the entity restaurant

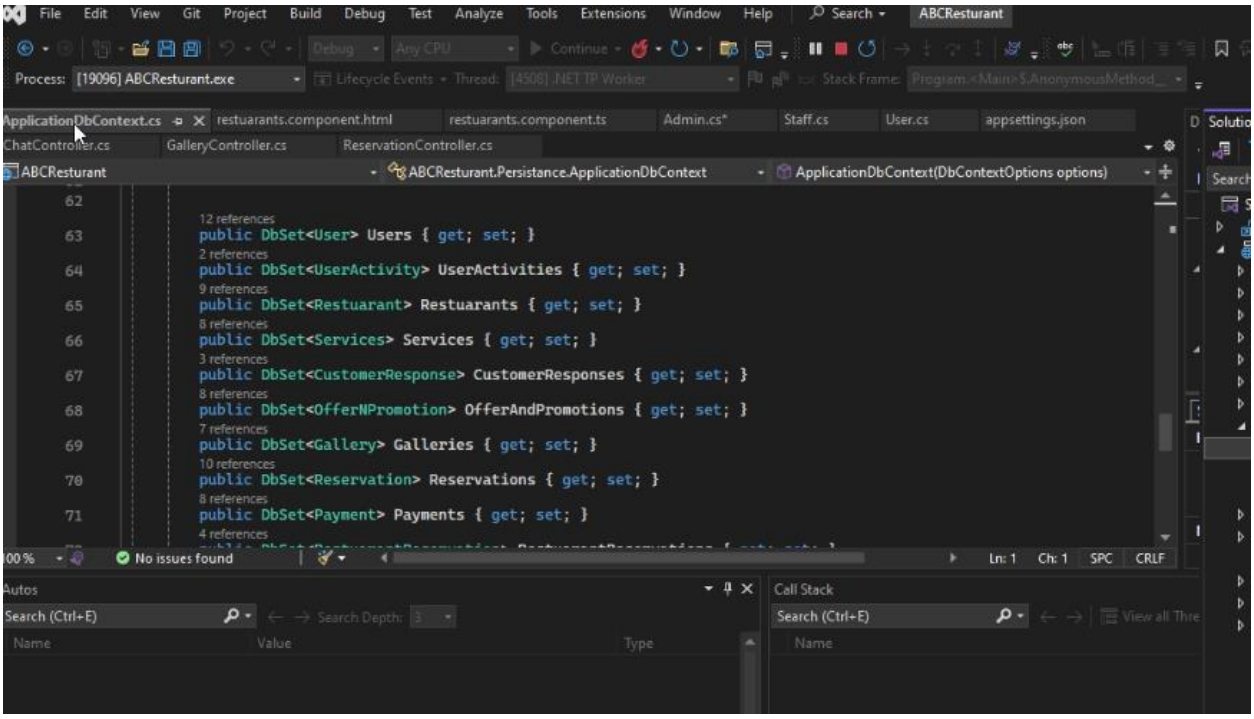


After that it determines whether it's an edit, create etc or which entity it is.


showDeleteConfirm(id: number) {
this.visible = true;
}
openForm(mode: 'create' | 'edit', restaurant?: any): void {
this.formMode = mode;
if (mode === 'edit' && restaurant) {
this.restaurantFormEdit.patchValue(restaurant);
} else {
this.restaurantFormEdit.reset();
this.restaurantFormNew.reset();
}
this.showForm = true;
const newUserActivity: UserActivity = {
activity: 'Open restuarant form in ' + mode,
userId: this.UsersService.getCurrentUser().id,
};

Repository pattern

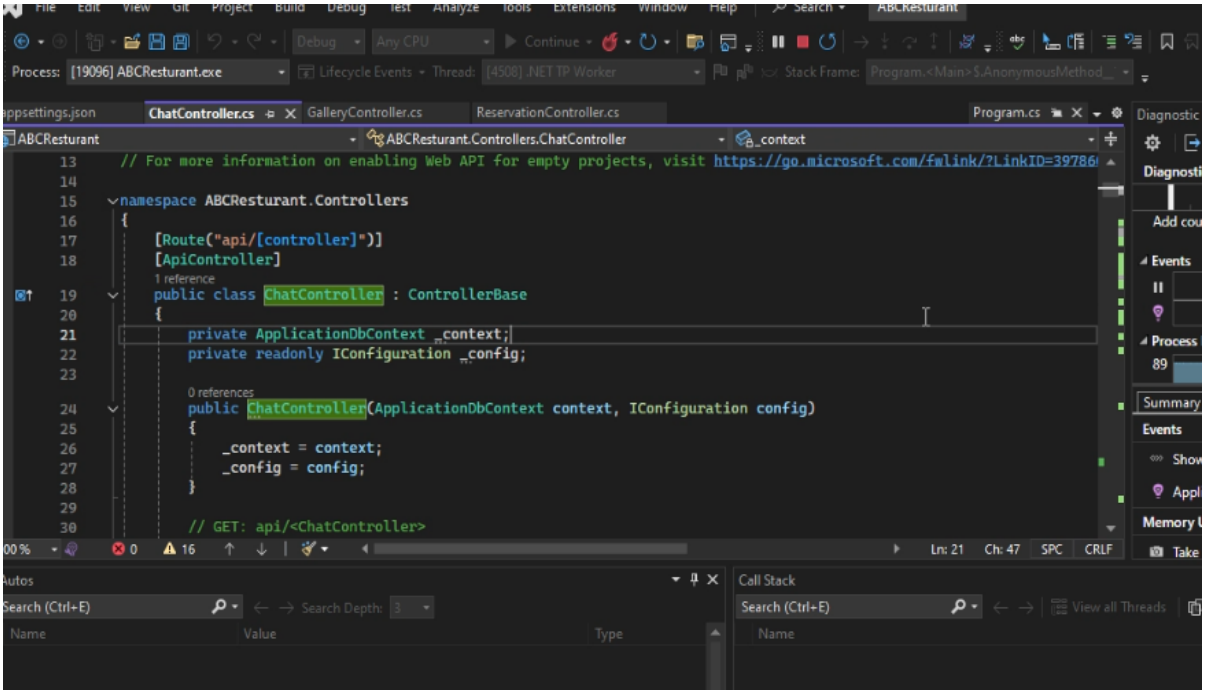
ApplicationDbContext.cs is used for the repository pattern here. Through the repository pattern we do the changes in the database such as update, drop etc. through this class. It handles the functionality of the repository pattern.



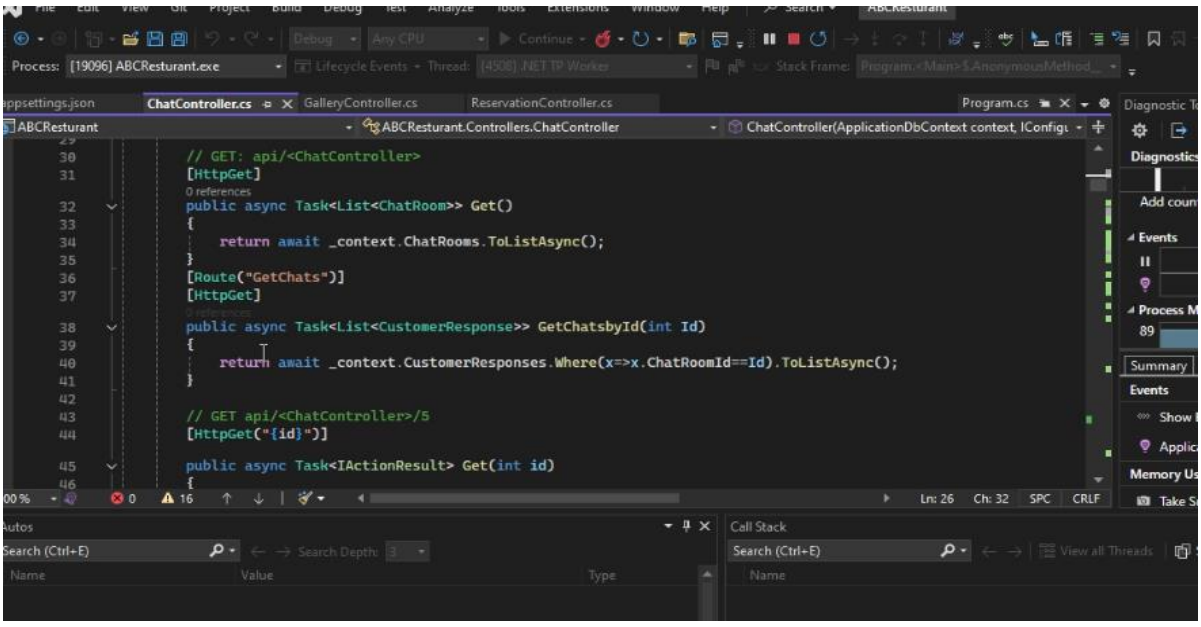
Use of OOP Concepts

Inheritance

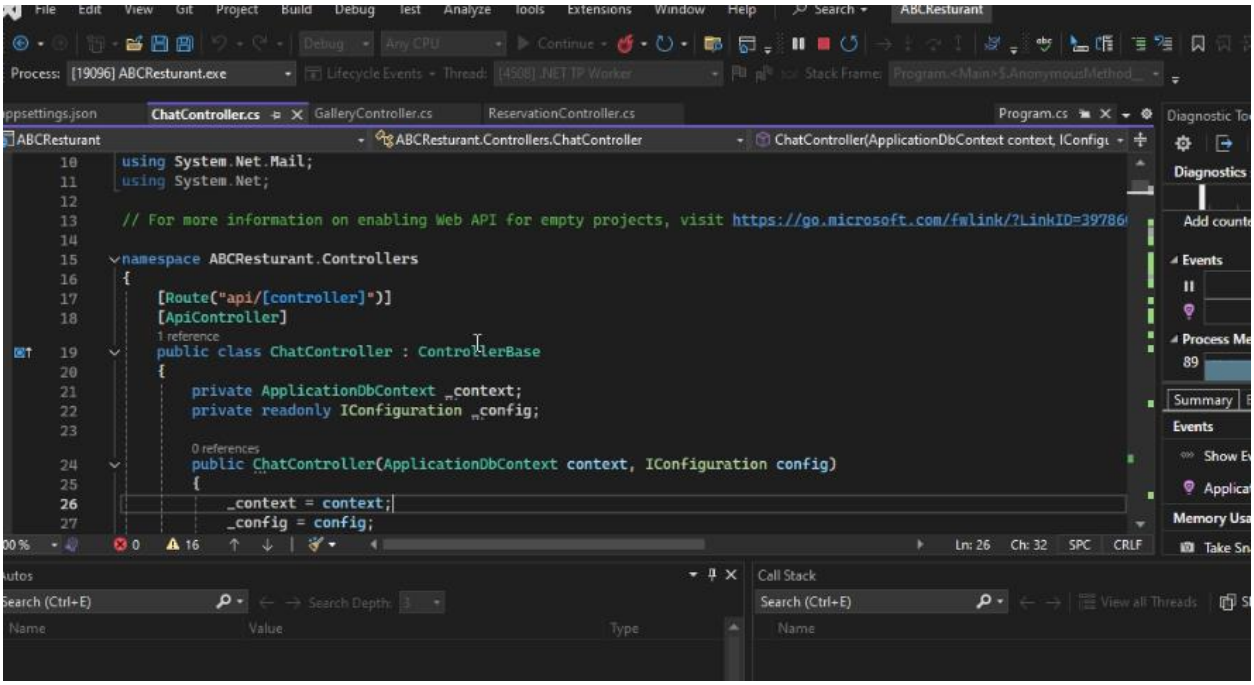
Inheritance OOP concept is used where ControllerBase is the parent and the ChatController has what was inherited from the parent class.



The ChatController can access HTTP get methods, route etc.

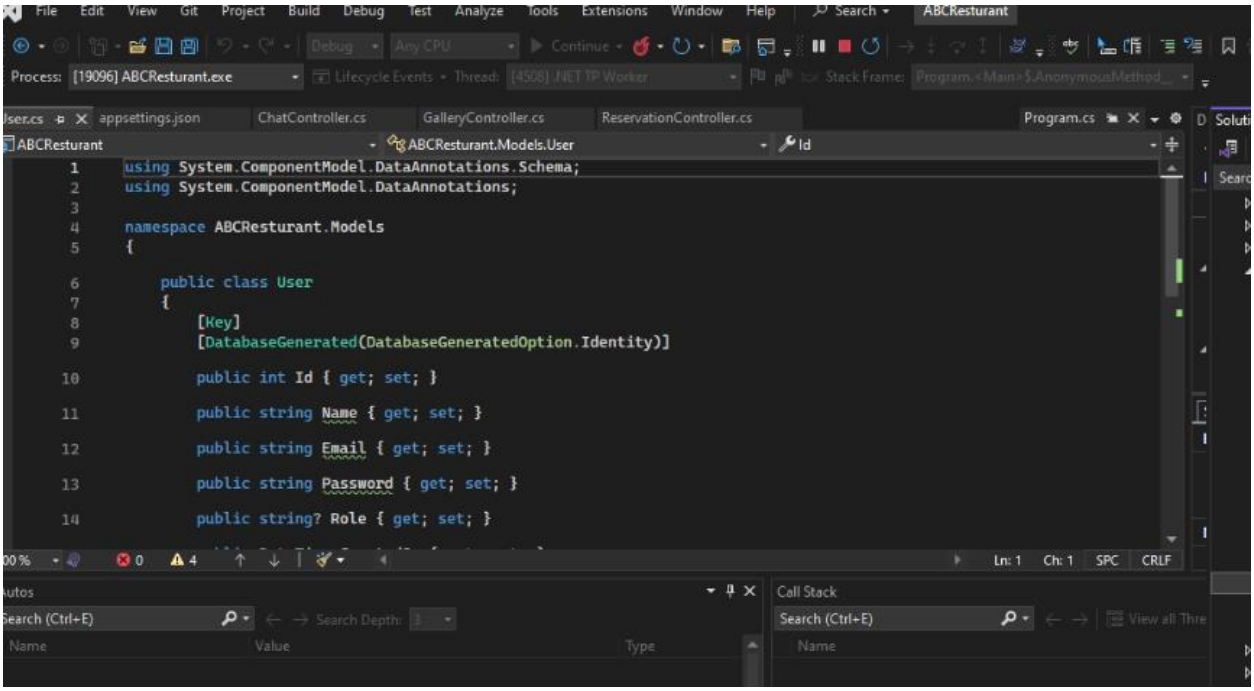


The ChatController can create APIs because of the ControllerBase

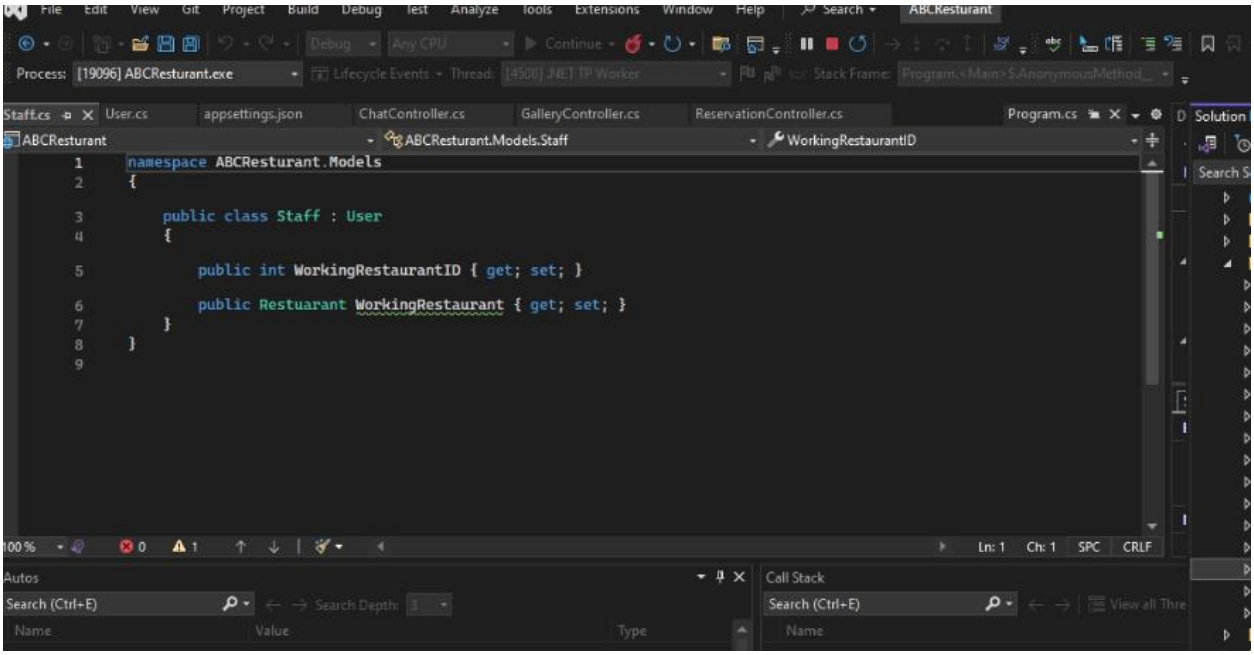


This works for the user roles as well. User is the parent where admin, staff and customer inherit data from user. So admin, staff and customer become the children of the parent user.

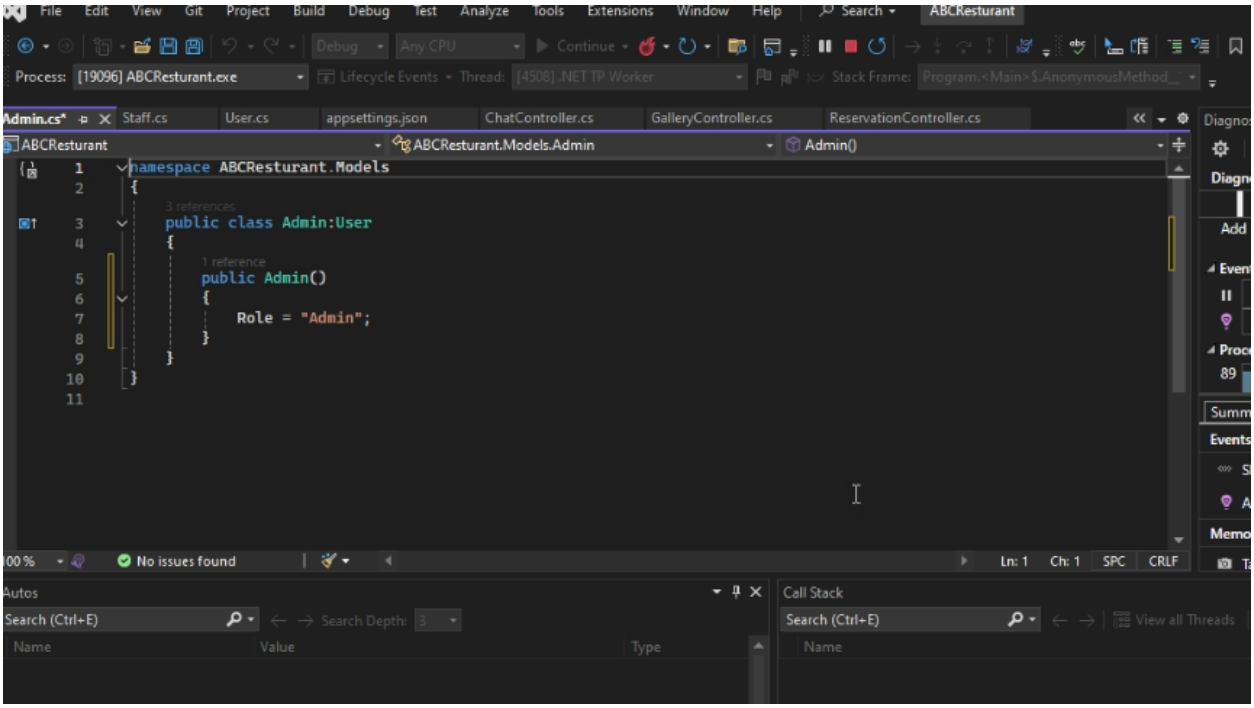
Following is the user



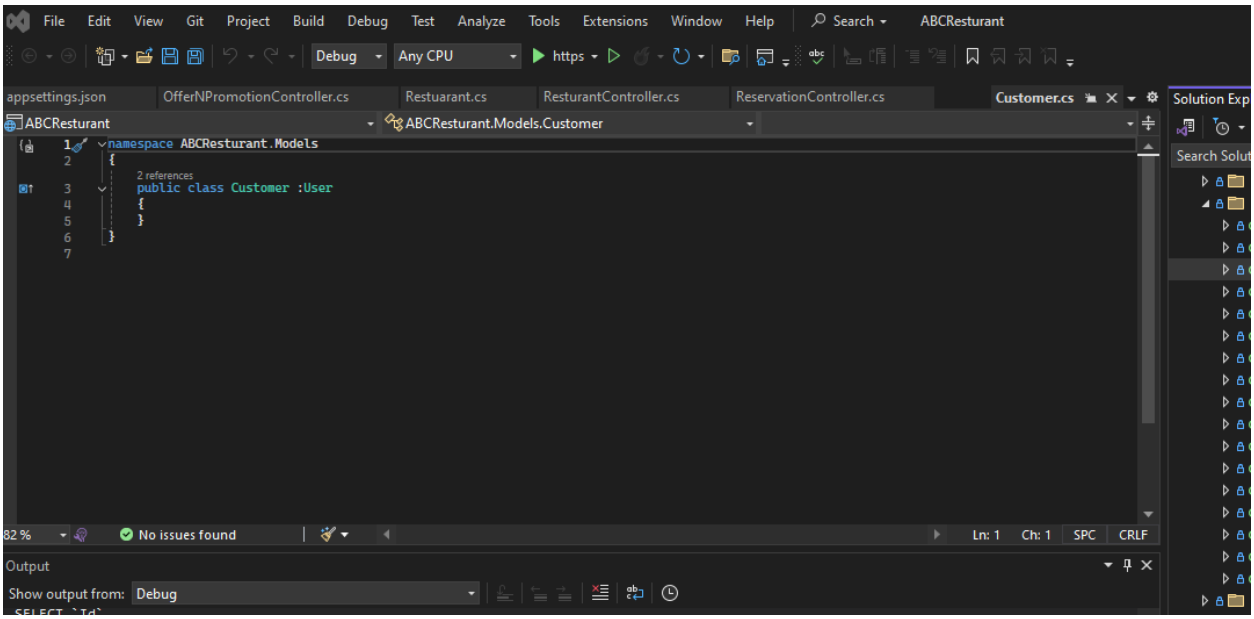
Staff


namespace ABCResturant.Models
{
public class Staff : User
{
public int WorkingRestaurantID { get; set; }
public Restuarant WorkingRestaurant { get; set; }
}
}

Admin


namespace ABCResturant.Models
{
public class Admin:User
{
public Admin()
{
Role = "Admin";
}
}
}

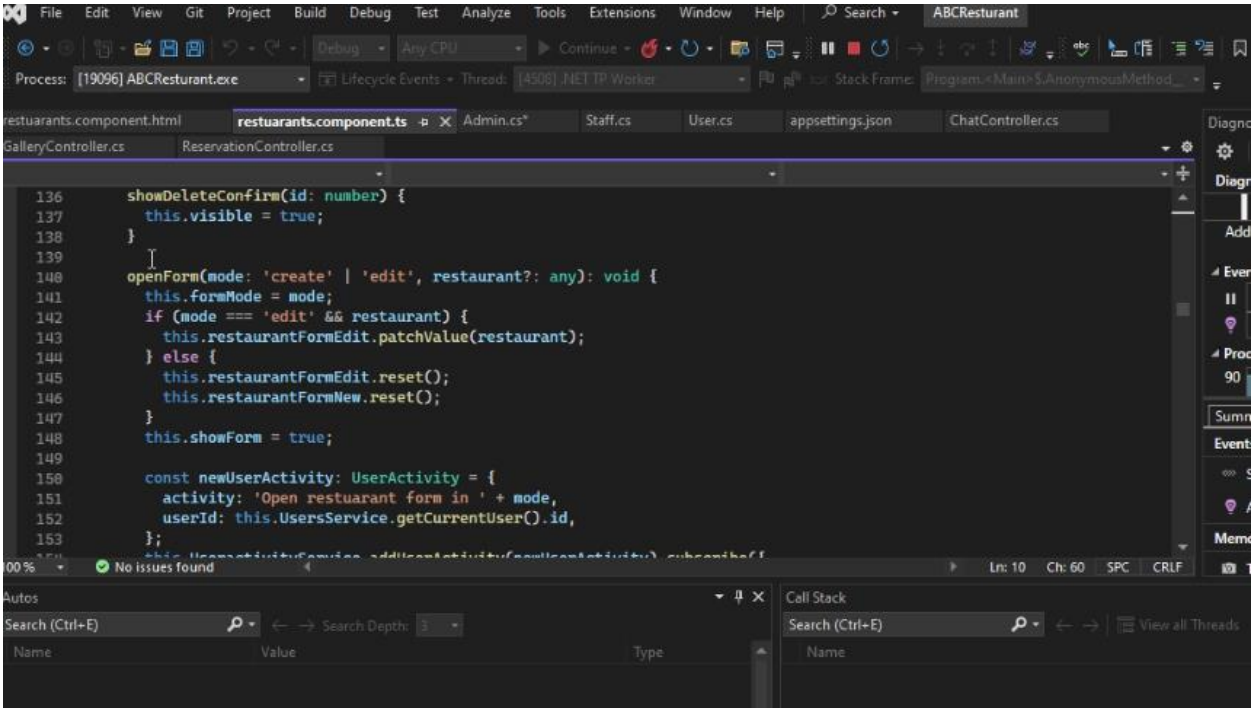
Customer



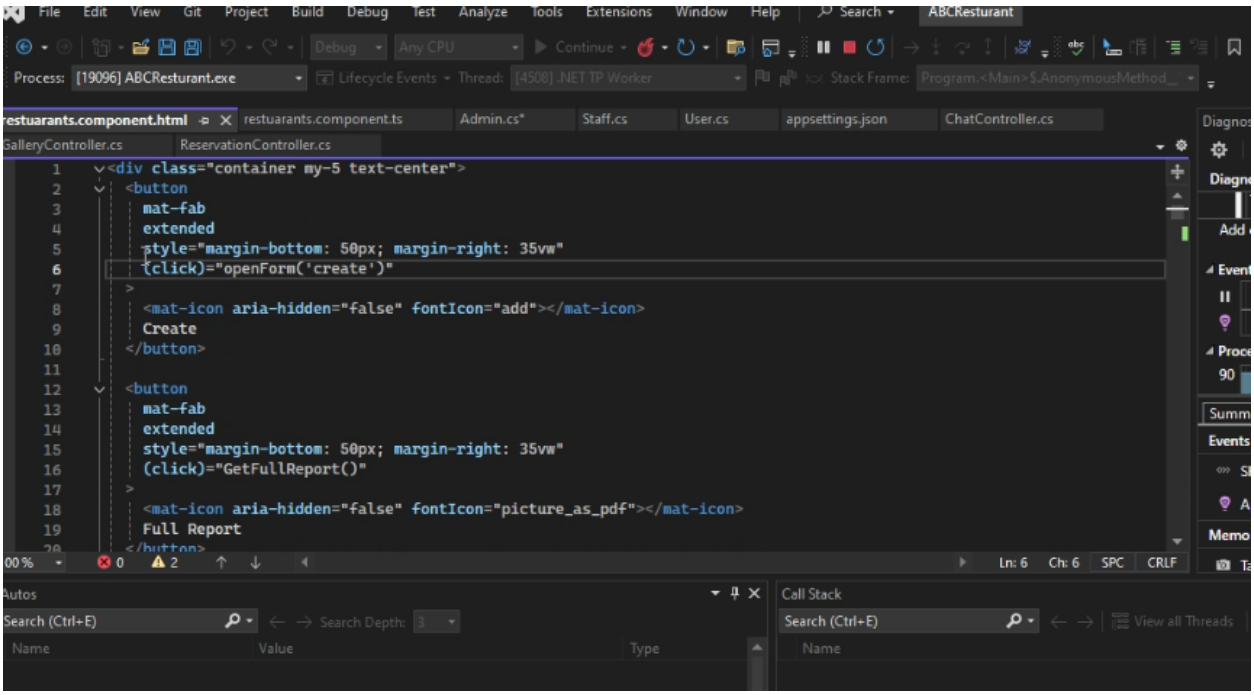
Polymorphism

For the same command many events are bound. As an example, when a certain button is clicked a pop up appears. The example is discussed further as follows.

When user clicks on create restaurant button a pop up appears. The openForm function works with create button which creates a listener for it and opens a pop up. The same function works with edit, delete etc. as well. Therefore, for the same function many events are bound.

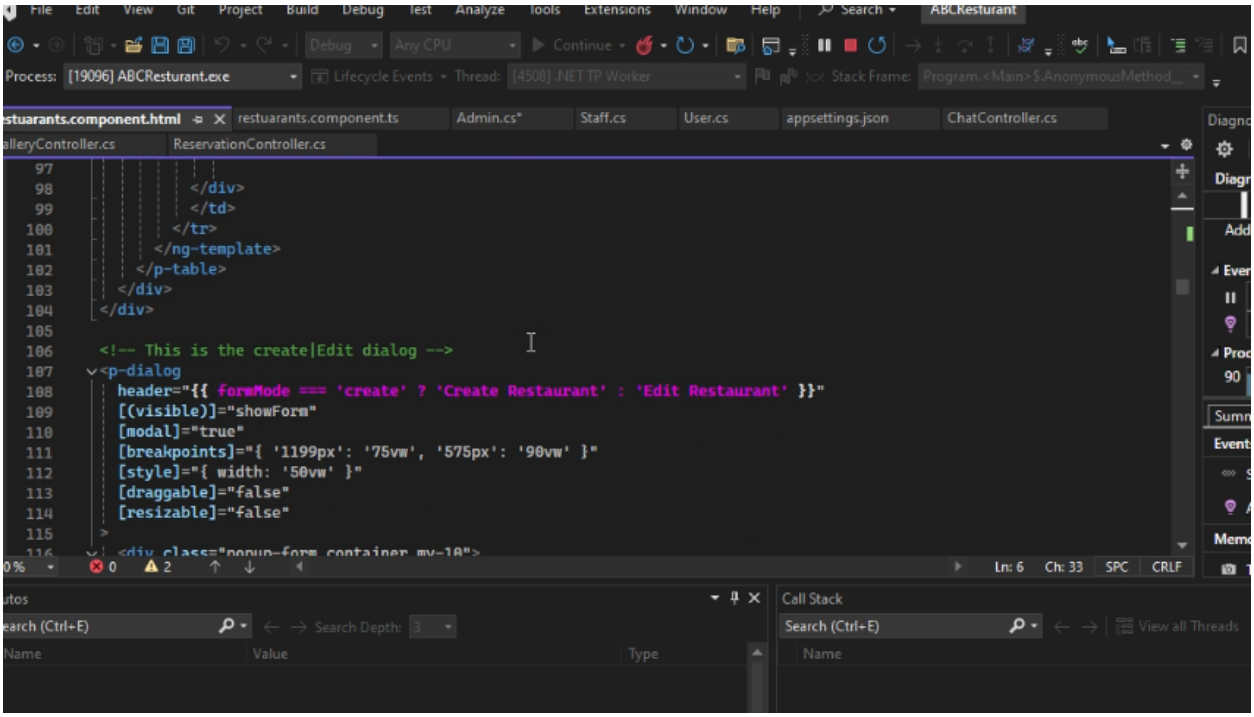


```
showDeleteConfirm(id: number) {
  this.visible = true;
}

openForm(mode: 'create' | 'edit', restaurant?: any): void {
  this.formMode = mode;
  if (mode === 'edit' && restaurant) {
    this.restaurantFormEdit.patchValue(restaurant);
  } else {
    this.restaurantFormEdit.reset();
    this.restaurantFormNew.reset();
  }
  this.showForm = true;

  const newUserActivity: UserActivity = {
    activity: 'Open restuarant form in ' + mode,
    userId: this.UsersService.getCurrentUser().id,
  };
```



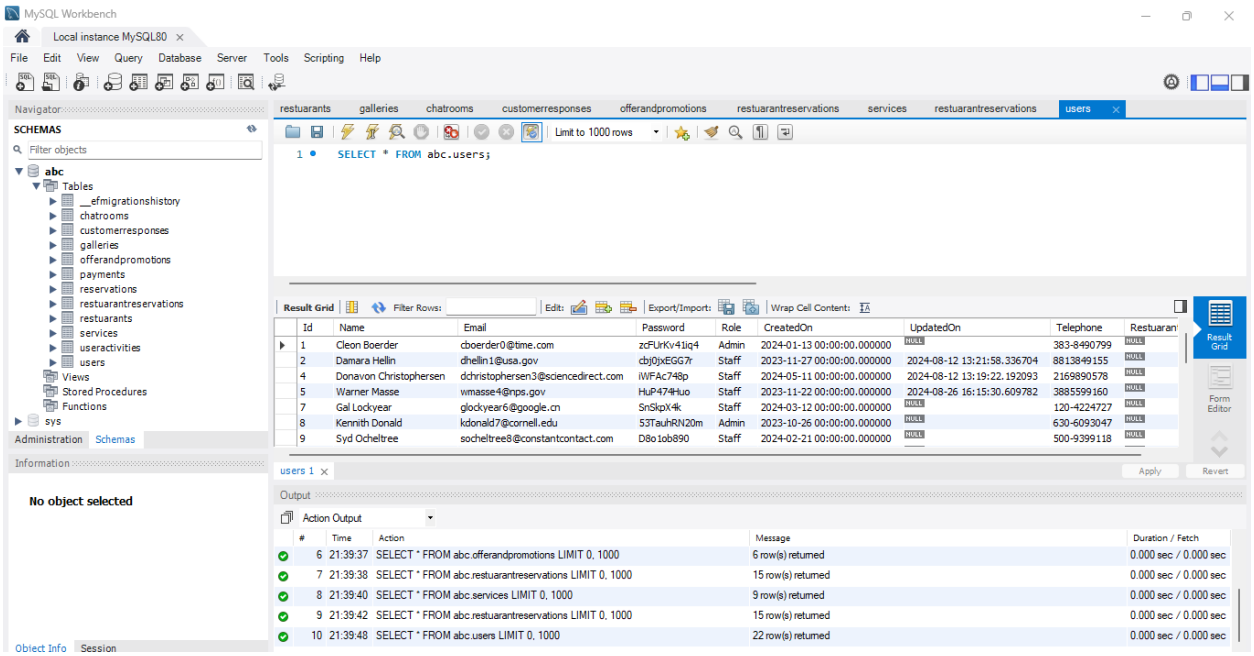
```
<div class="container my-5 text-center">
  <button
    mat-fab
    extended
    style="margin-bottom: 50px; margin-right: 35vw"
    (click)="openForm('create')"
  >
    <mat-icon aria-hidden="false" fontIcon="add"></mat-icon>
    Create
  </button>

  <button
    mat-fab
    extended
    style="margin-bottom: 50px; margin-right: 35vw"
    (click)="GetFullReport()"
  >
    <mat-icon aria-hidden="false" fontIcon="picture_as_pdf"></mat-icon>
    Full Report
  </button>
```

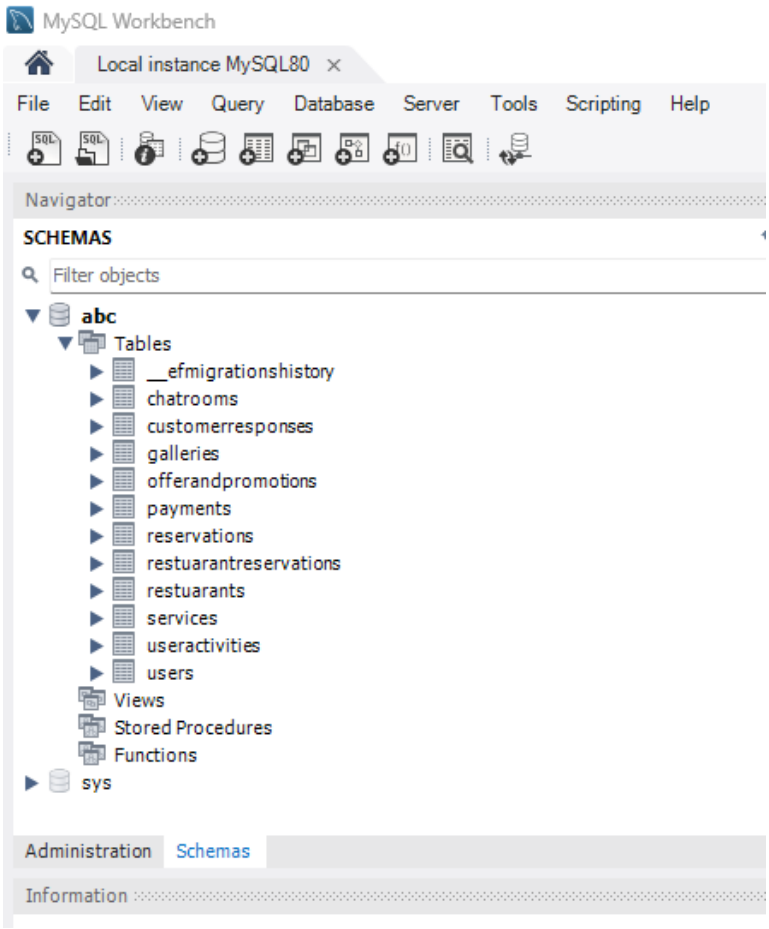
It determines whether the button click was create, edit or delete etc. and opens the pop up when model becomes true here.



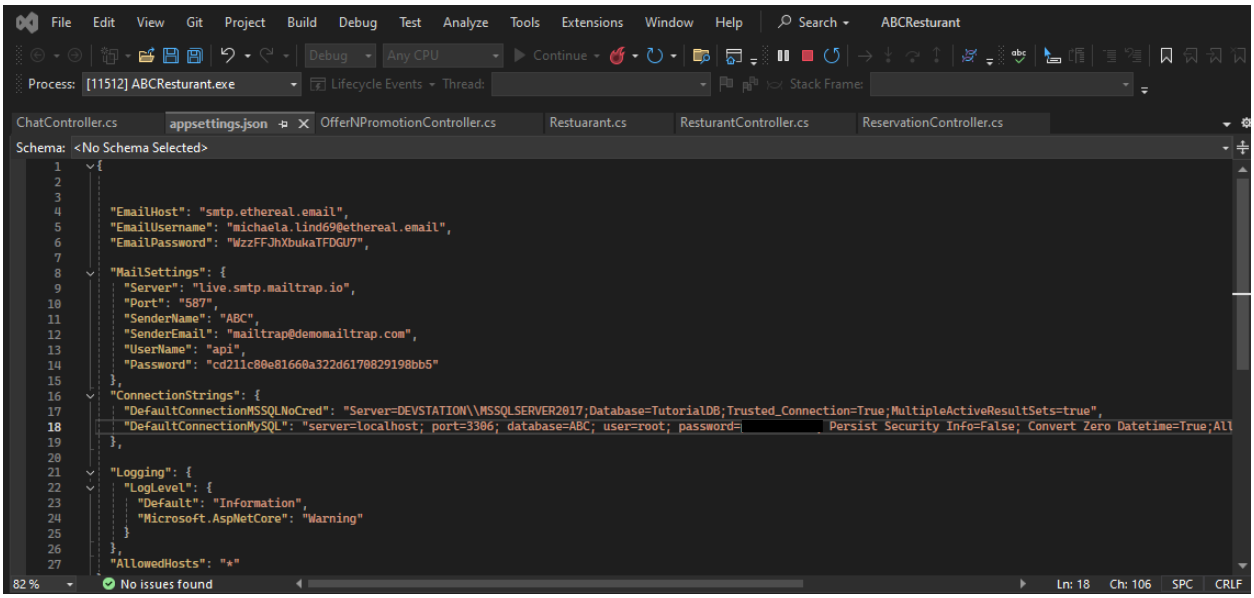
Database

MySQL was chosen as the database and MySQL workbench was used as the visual database design tool.





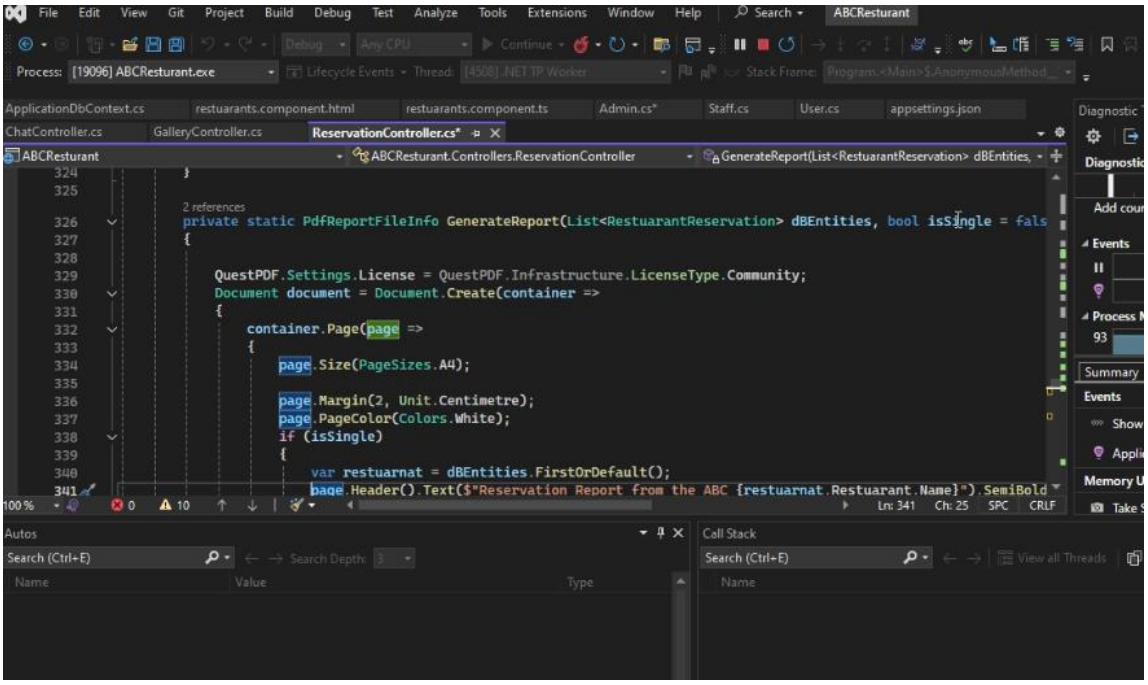
In appsettings.json ConnectionStrings are included.



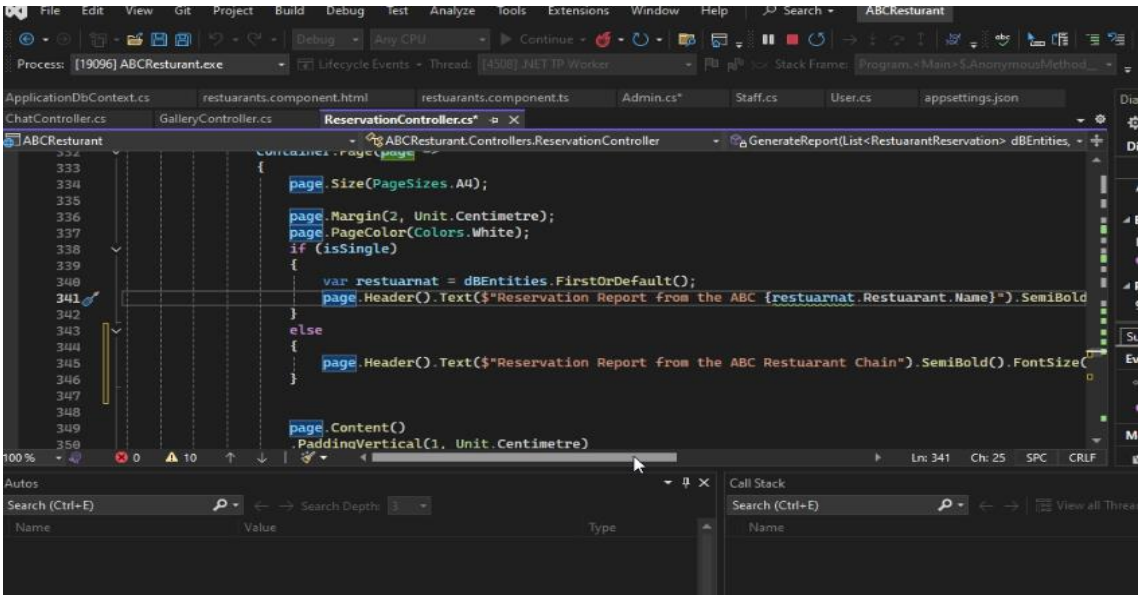
Report Generation

QuestPDF library was used for report generation.

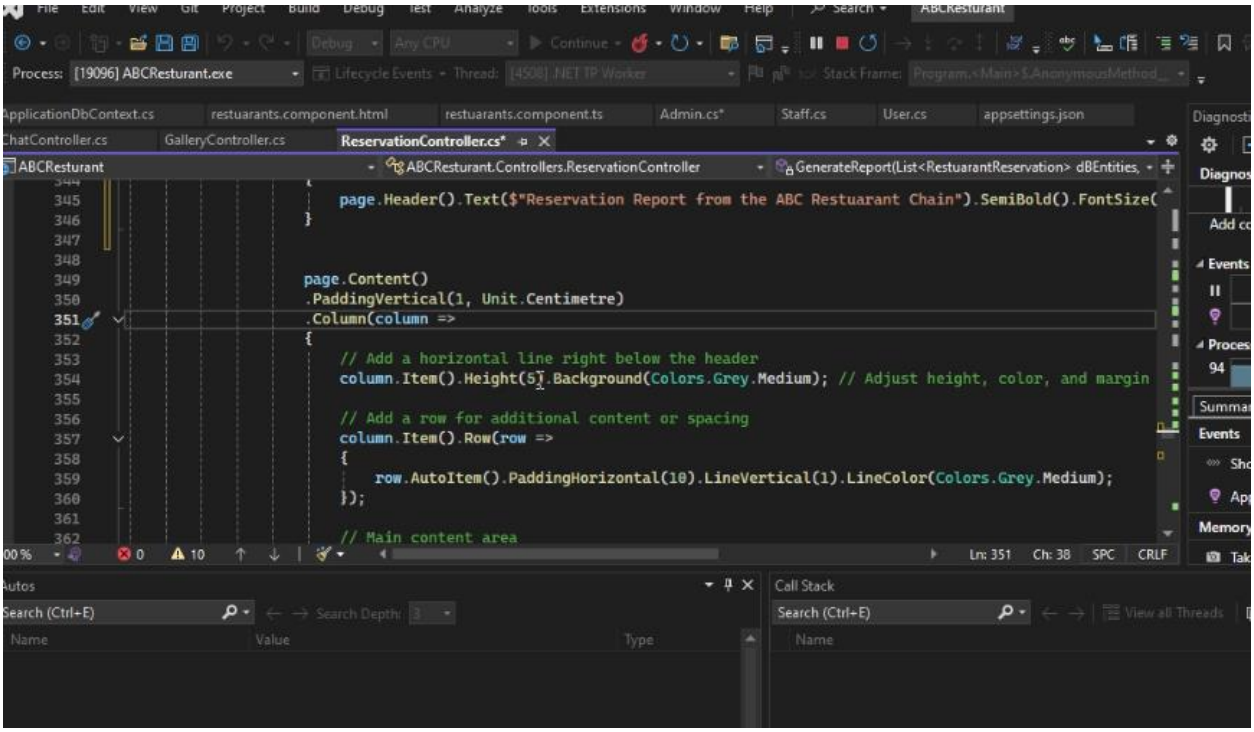
Following shows the function for it. First, it creates a document. Through the container in the following image, we define the page properties such as page size, margins, PageColor etc.

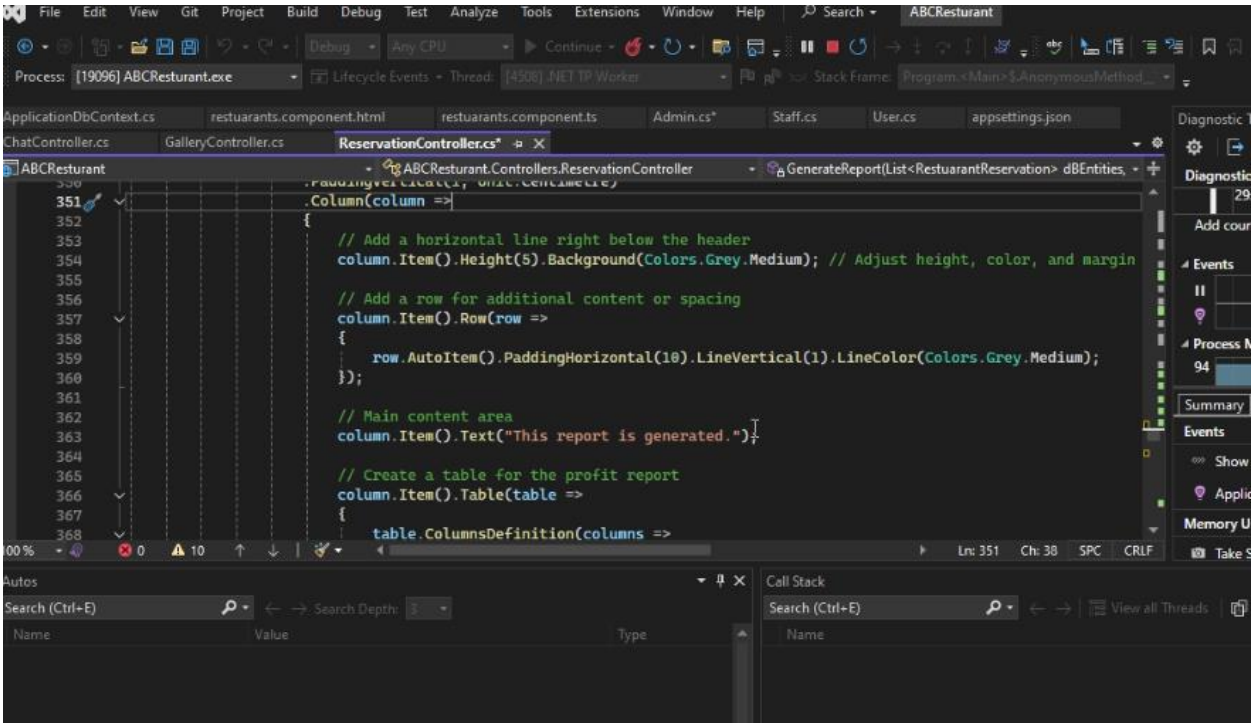


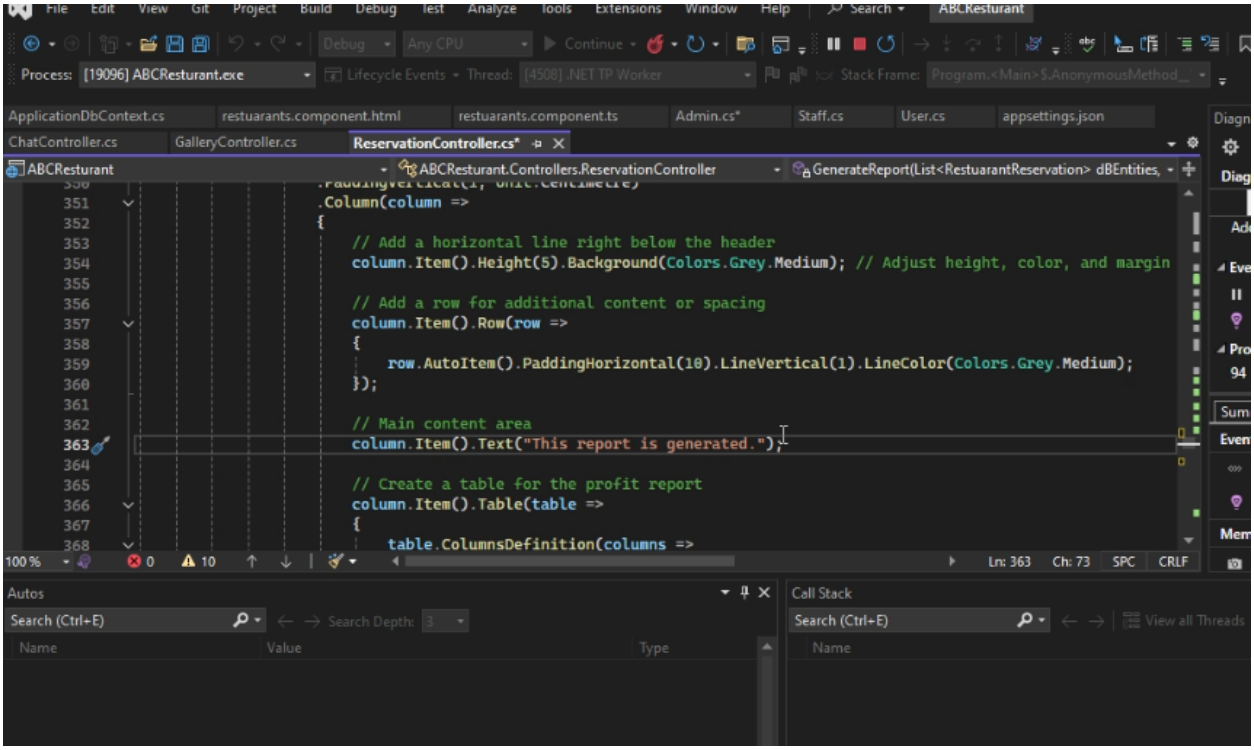
In the following image we can define the document header.



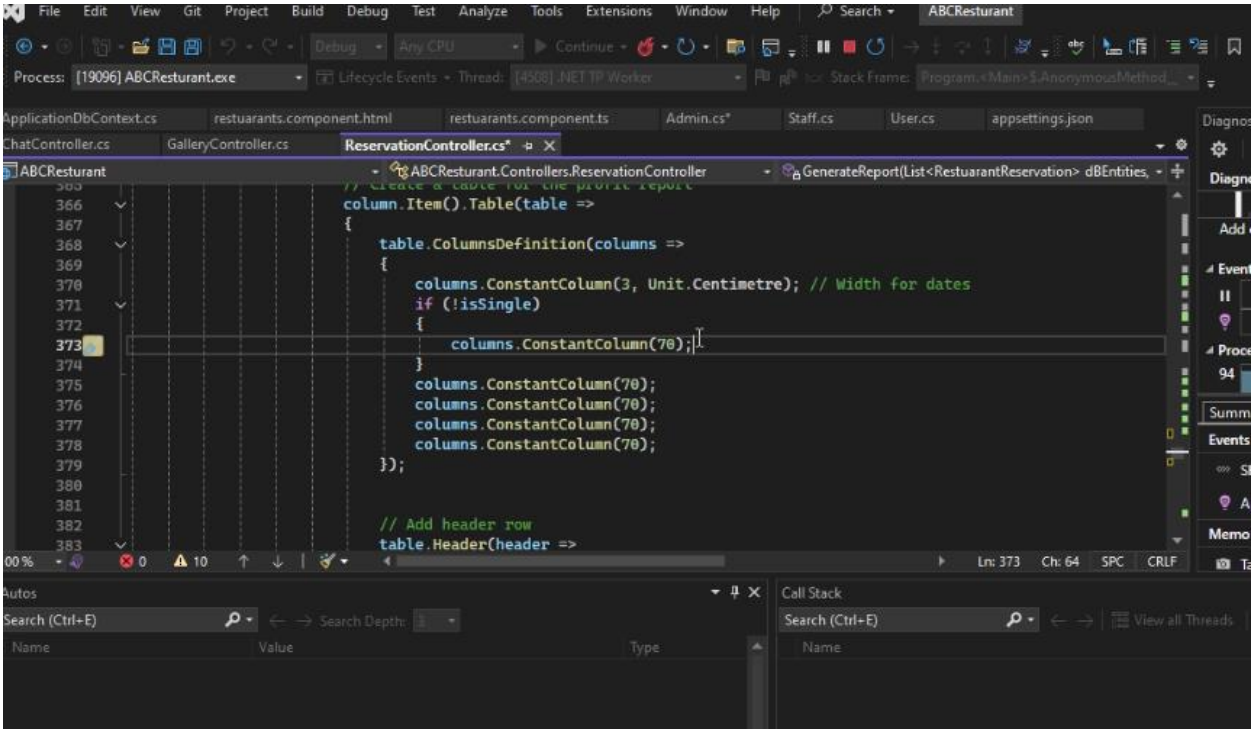
page.Content() defines the appearance of the document.



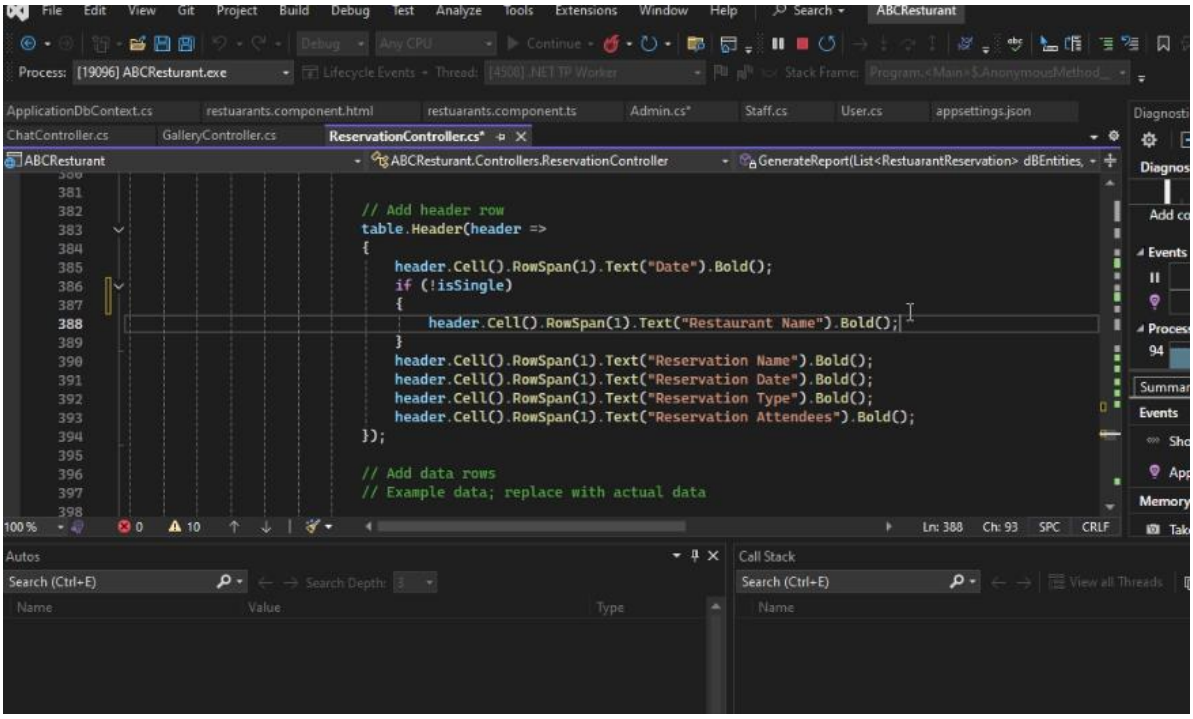




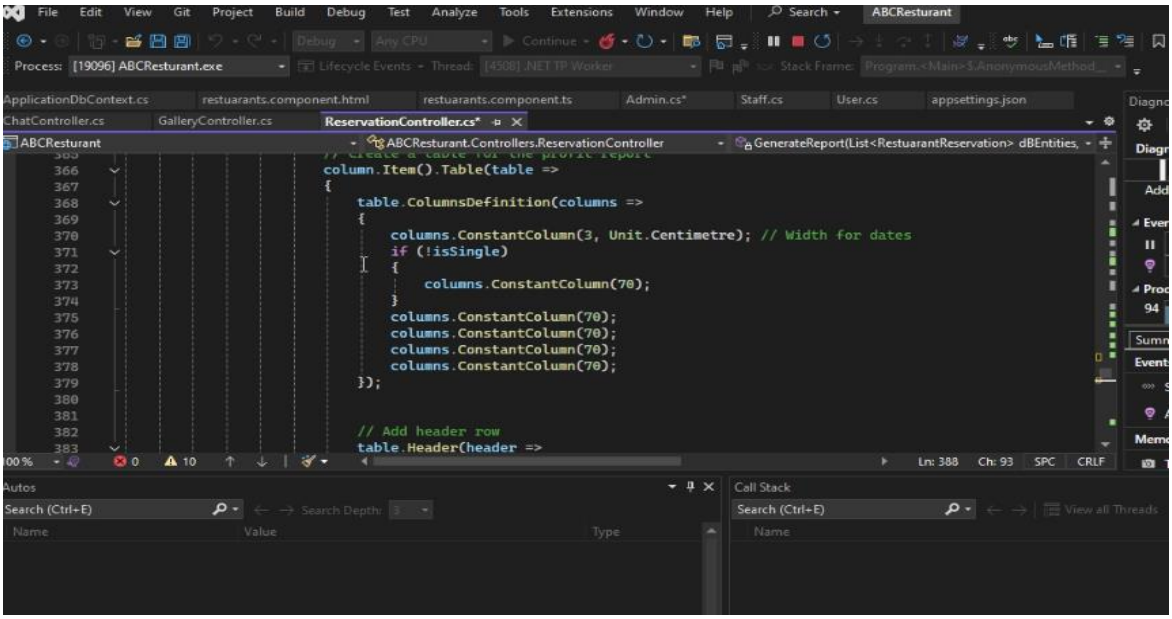
After that a table is created for the report



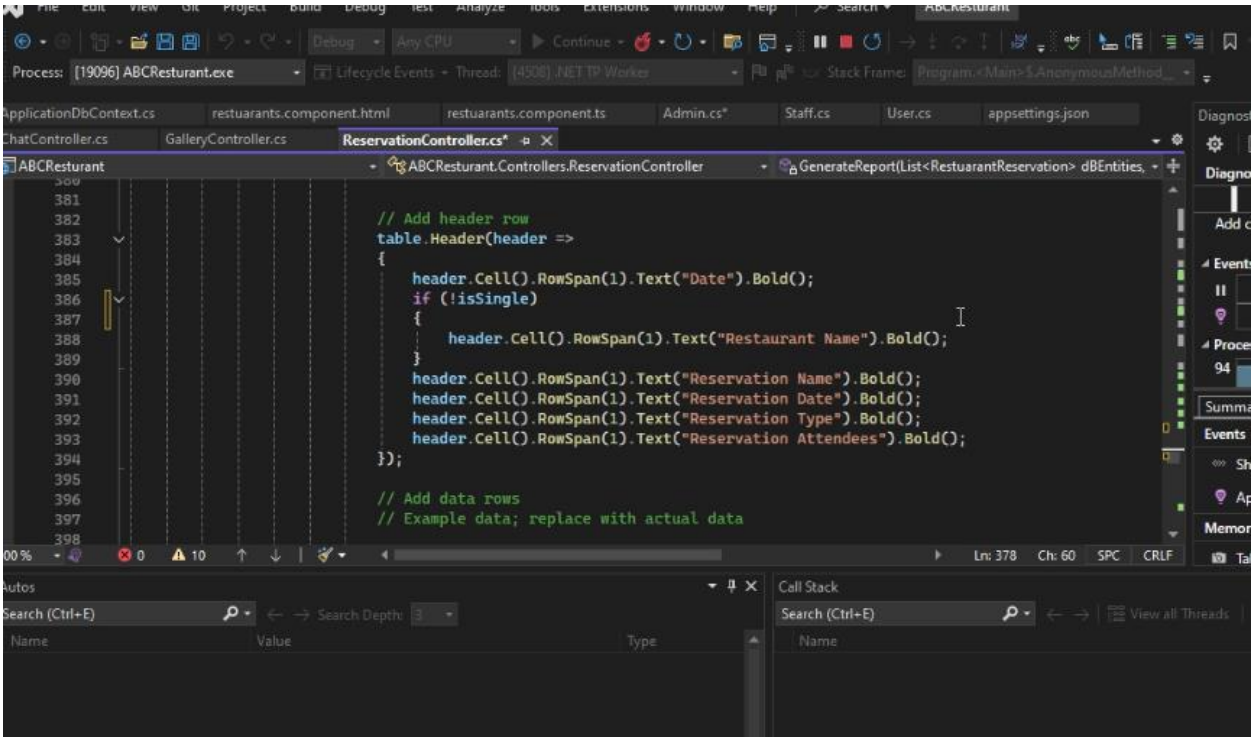
This takes restaurant name, date, attendees



In case this report doesn't depend on one restaurant but the entire restaurant chain, (eg: reservation list of the entire restaurant chain) as below it creates an additional column in the table to get the restaurant names.

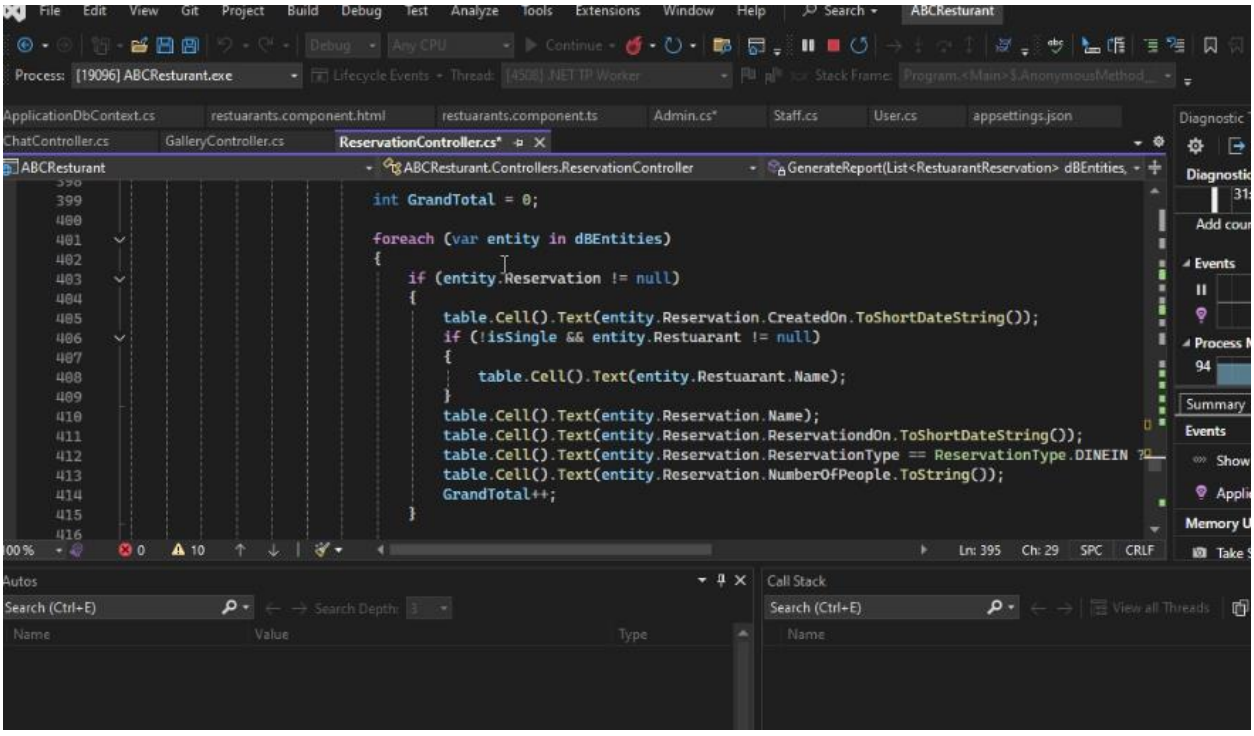


This is how we get the column headers of the report.

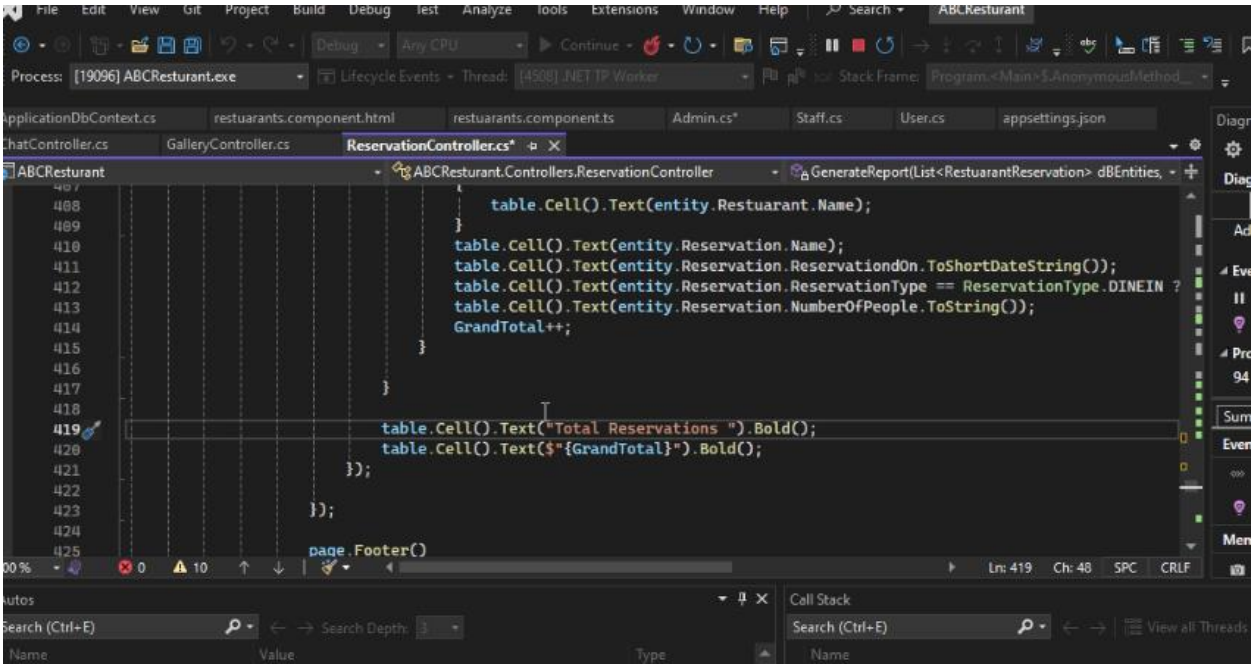


```
// Add header row
table.Header(header =>
{
    header.Cell().RowSpan(1).Text("Date").Bold();
    if (!isSingle)
    {
        header.Cell().RowSpan(1).Text("Restaurant Name").Bold();
    }
    header.Cell().RowSpan(1).Text("Reservation Name").Bold();
    header.Cell().RowSpan(1).Text("Reservation Date").Bold();
    header.Cell().RowSpan(1).Text("Reservation Type").Bold();
    header.Cell().RowSpan(1).Text("Reservation Attendees").Bold();
});

// Add data rows
// Example data; replace with actual data
```

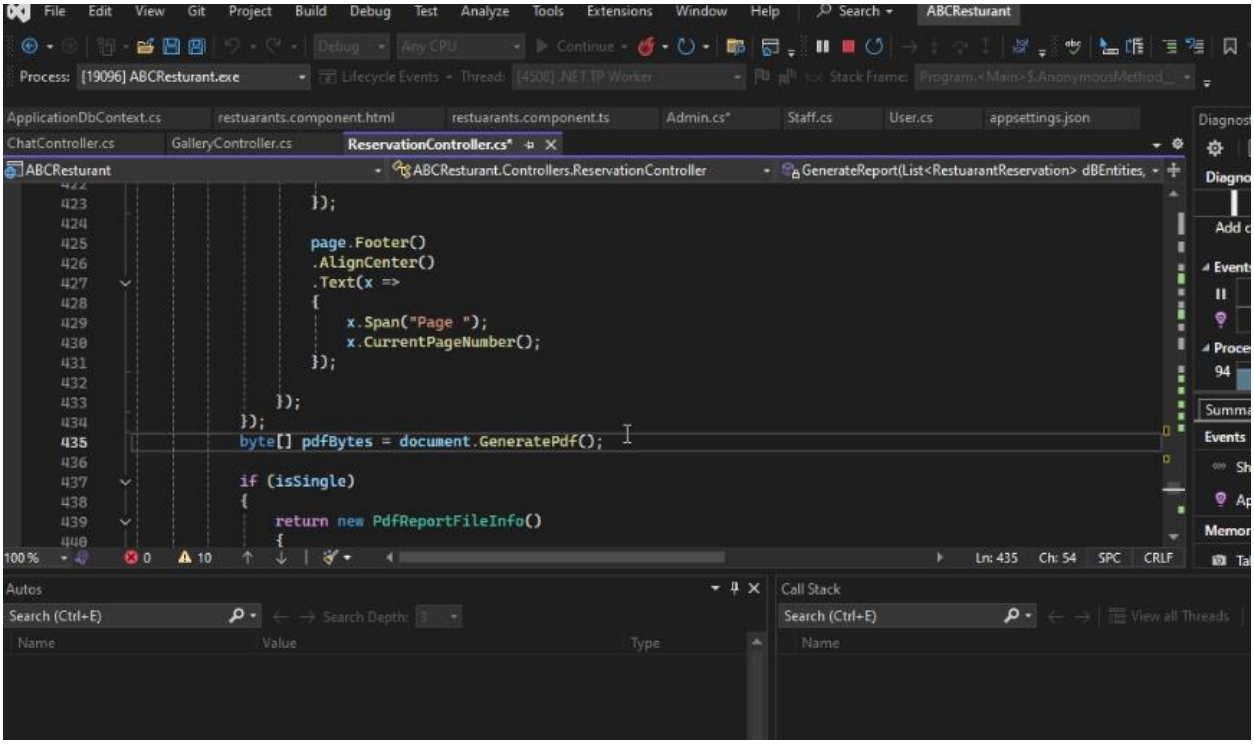
With the help of a foreach loop, it fills the data on the document. GrandTotal has the total number of reservations.



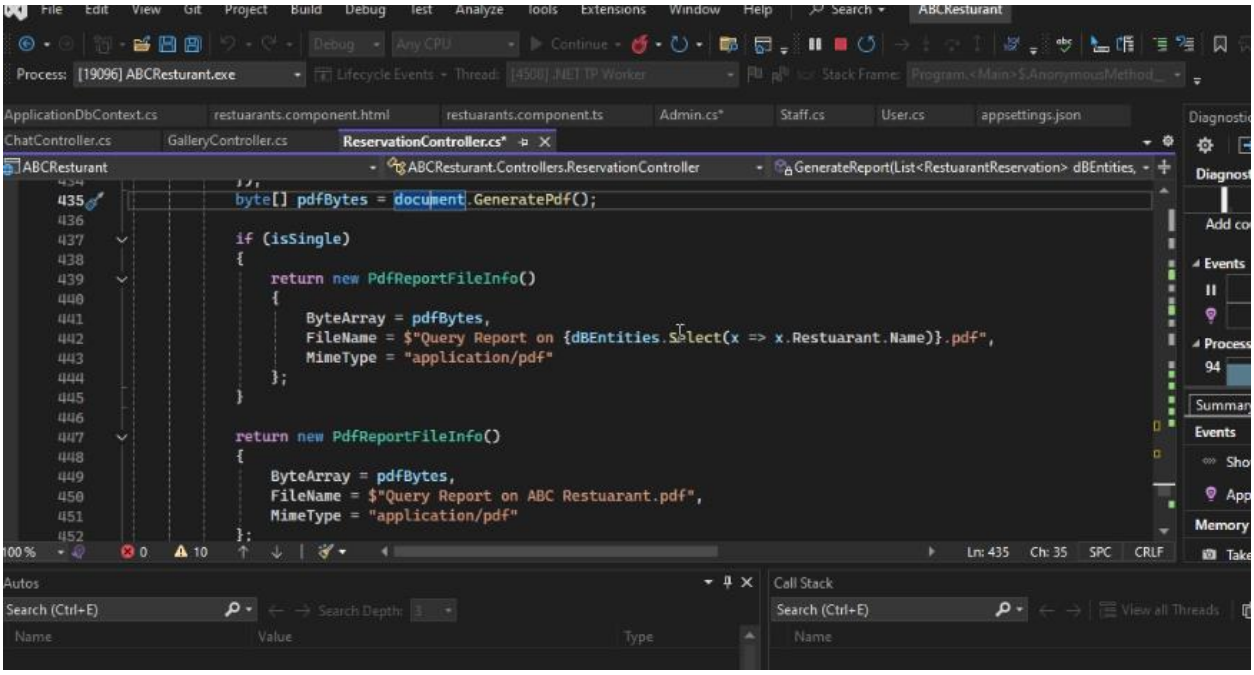
```
int GrandTotal = 0;

foreach (var entity in dBEntities)
{
    if (entity.Reservation != null)
    {
        table.Cell().Text(entity.Reservation.CreatedOn.ToShortDateString());
        if (!isSingle && entity.Restuarant != null)
        {
            table.Cell().Text(entity.Restuarant.Name);
        }
        table.Cell().Text(entity.Reservation.Name);
        table.Cell().Text(entity.Reservation.ReservationdOn.ToShortDateString());
        table.Cell().Text(entity.Reservation.ReservationType == ReservationType.DINEIN ?
        table.Cell().Text(entity.Reservation.NumberOfPeople.ToString());
        GrandTotal++;
    }
```

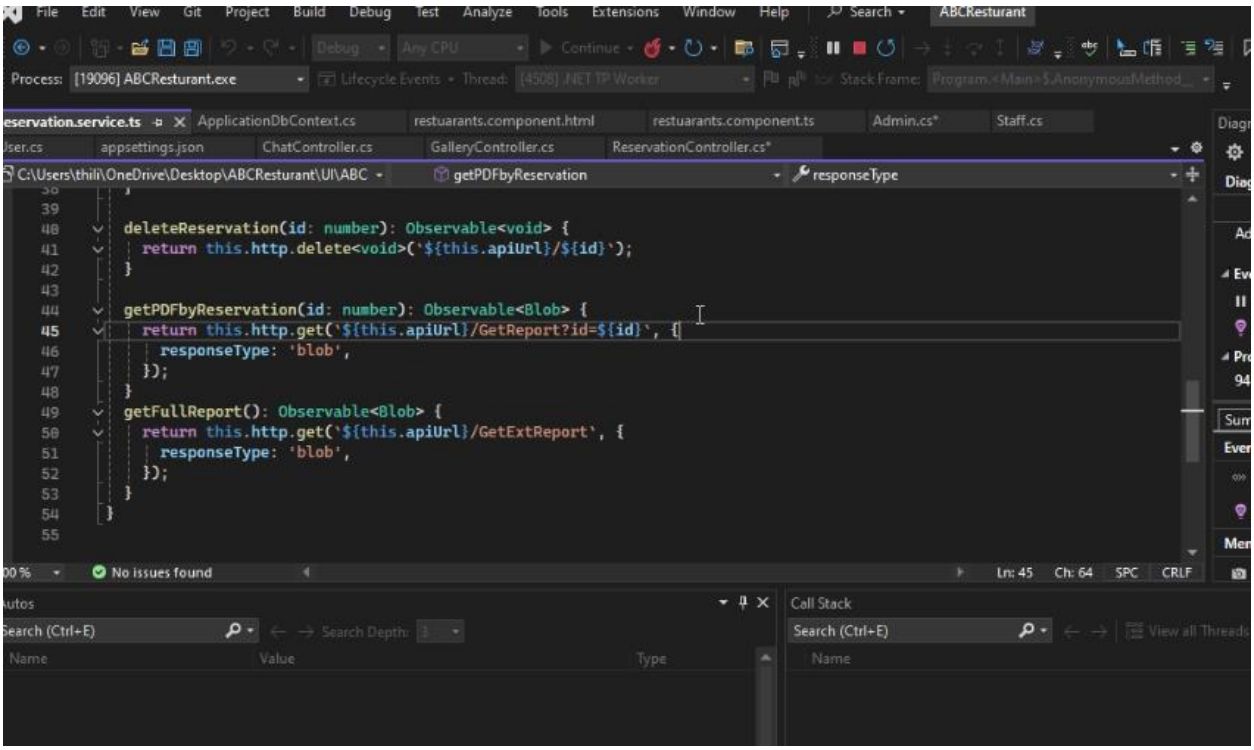


It further describes the content of the document. This is the code for page footer



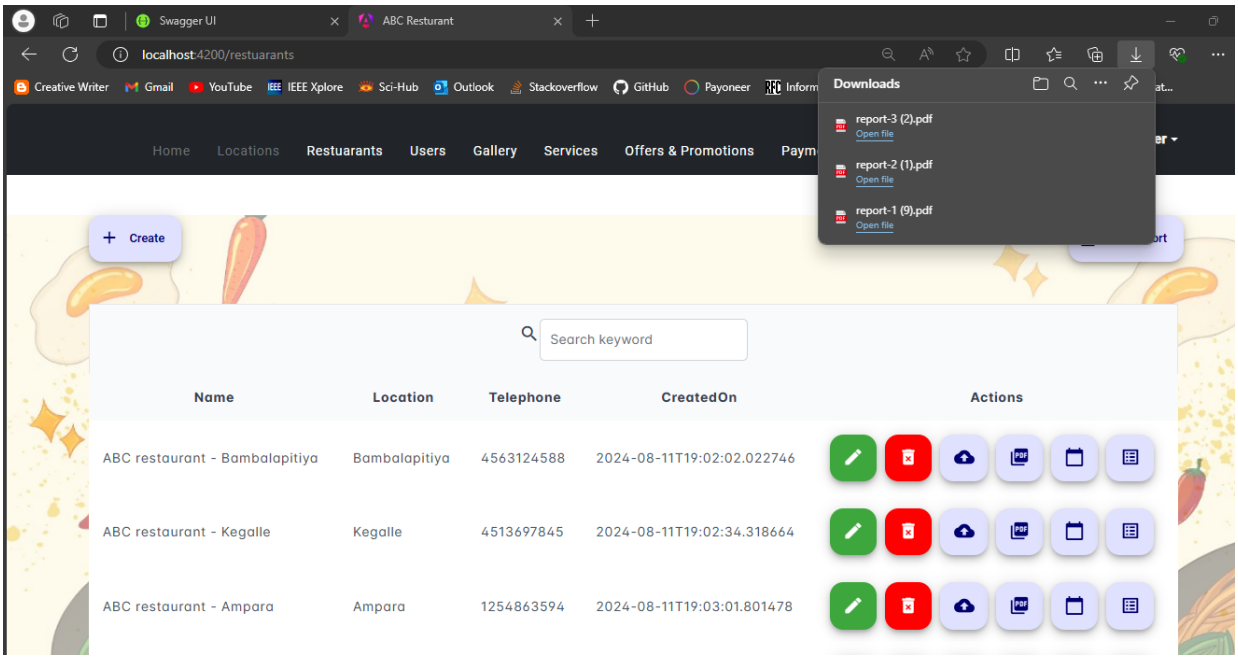
After that it generates the document





The same logic is used for all the types of reports that gets generated in this API.

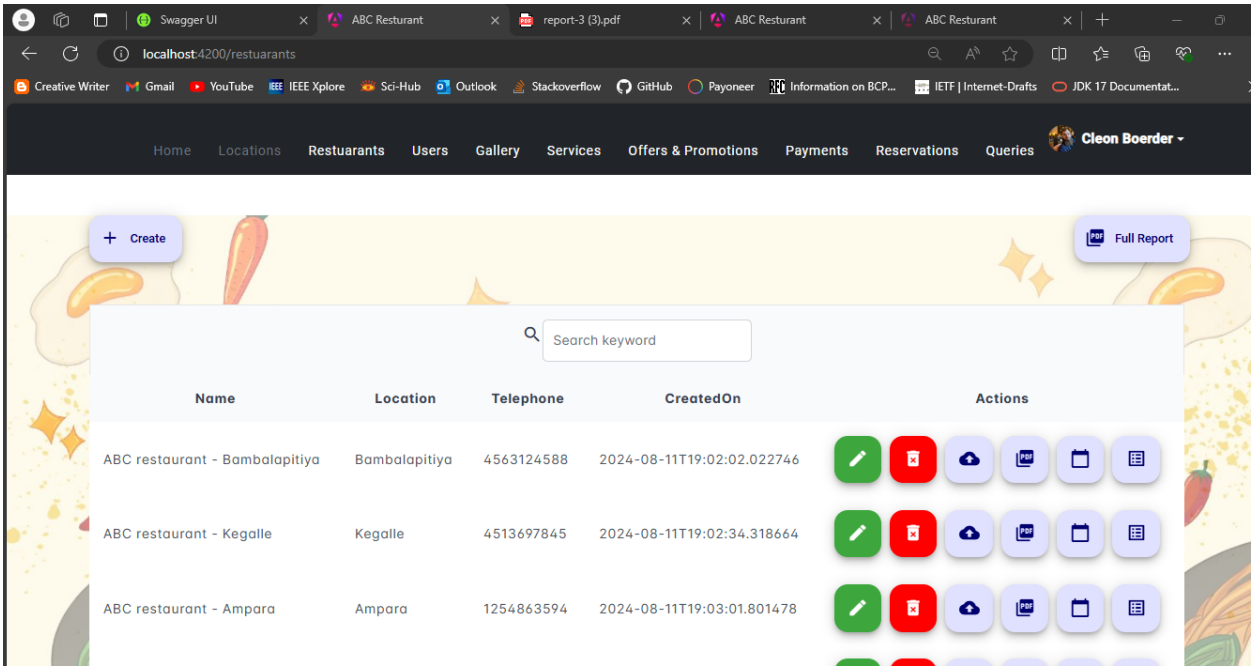
Once you click on the PDF or generate report buttons it downloads the document.

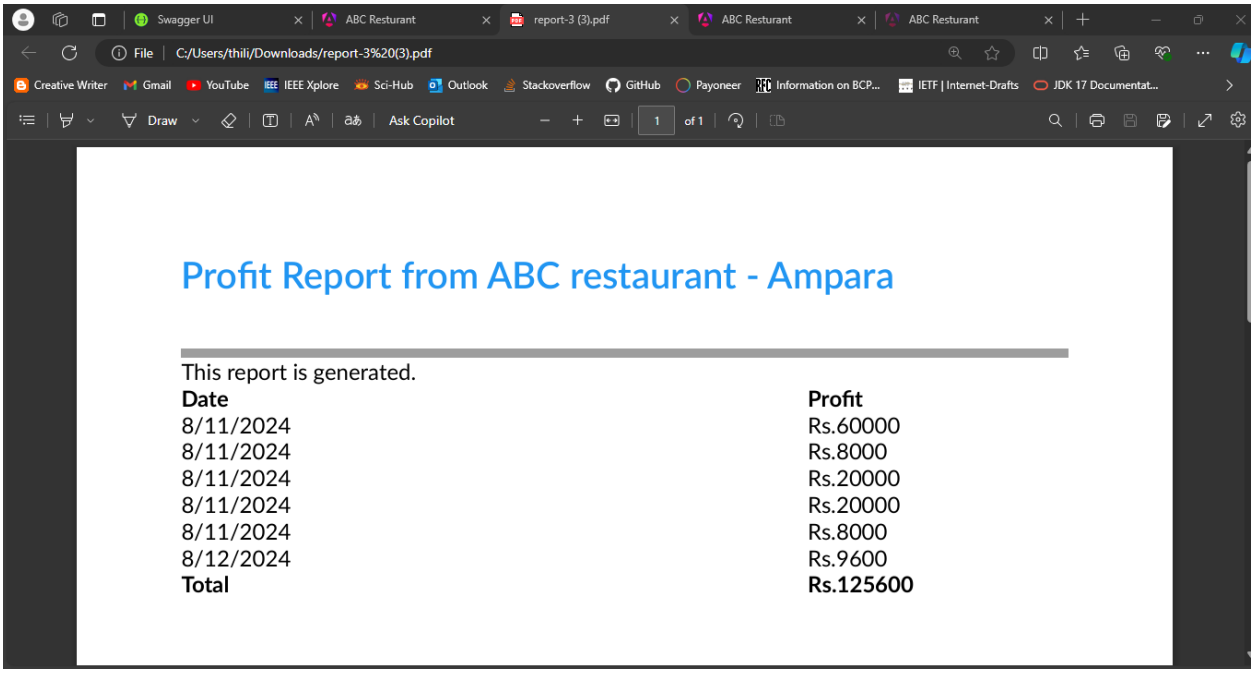


Profit report per restaurant –

This report includes the date and the profit received for the particular day and the total. Admin only can generate this report.

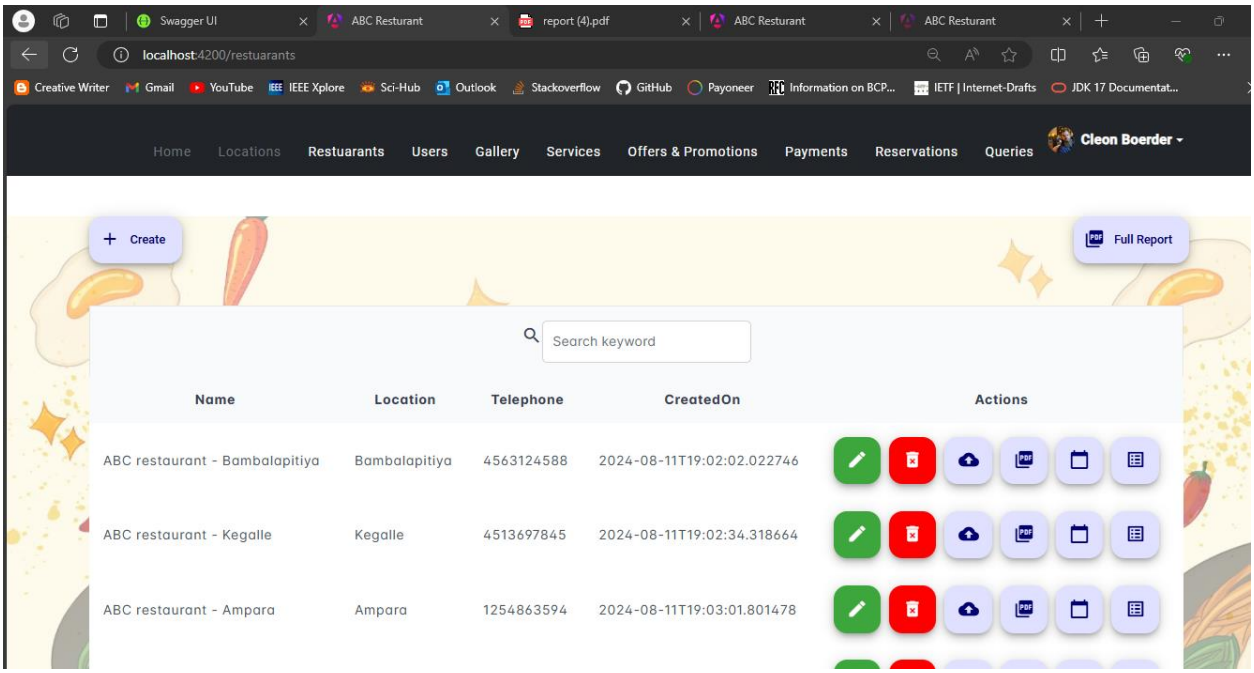
To generate this, the admin has to click on the PDF icon under actions for the particular restaurant under restaurants tab.

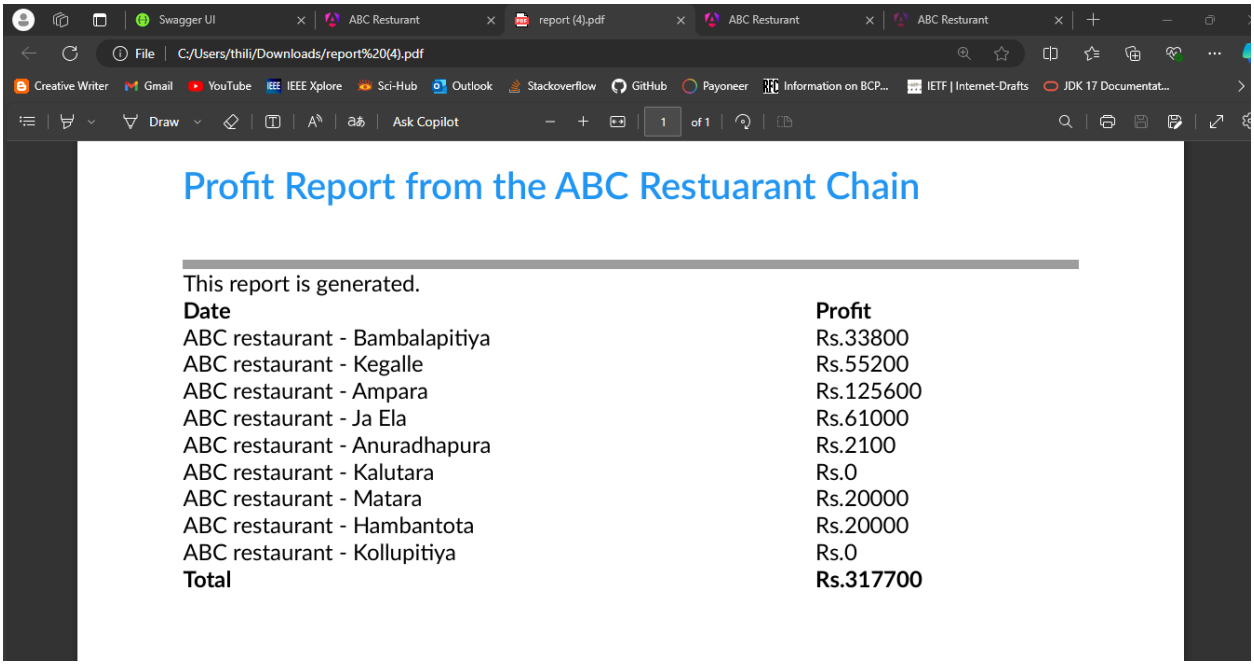




Profit report from ABC restaurant chain –

This report includes the full profit report of the whole restaurant chain. To generate it the admin has to click on the full report button.





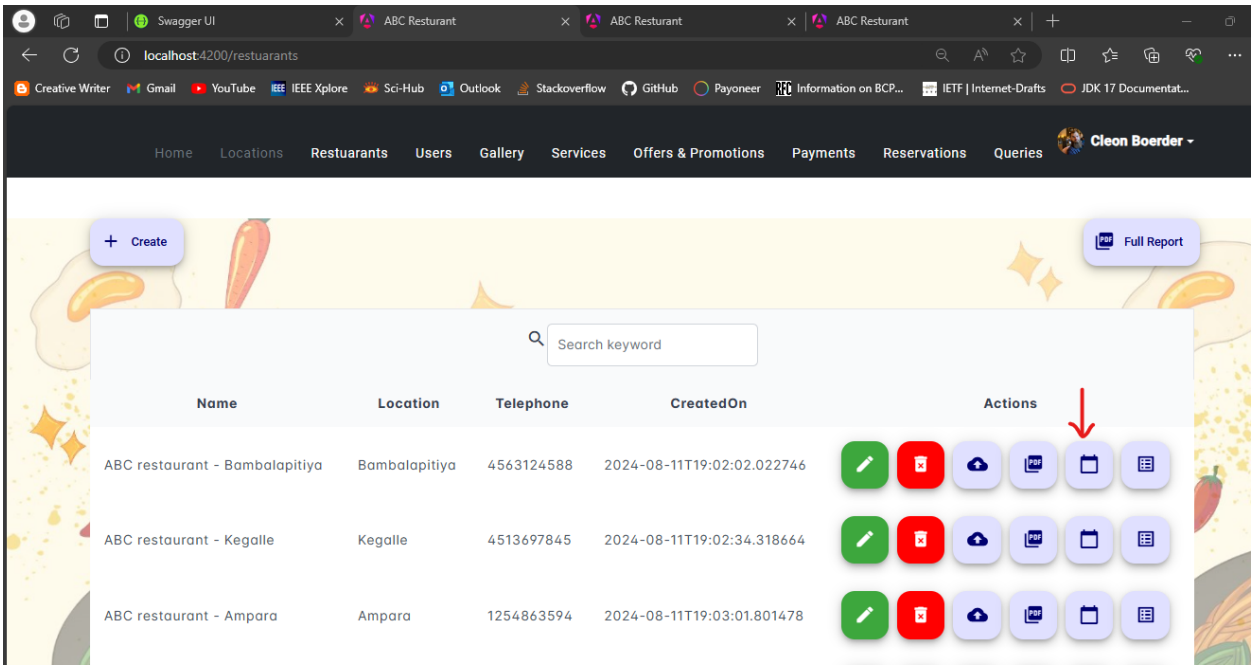
Profit Report from the ABC Restuarant Chain

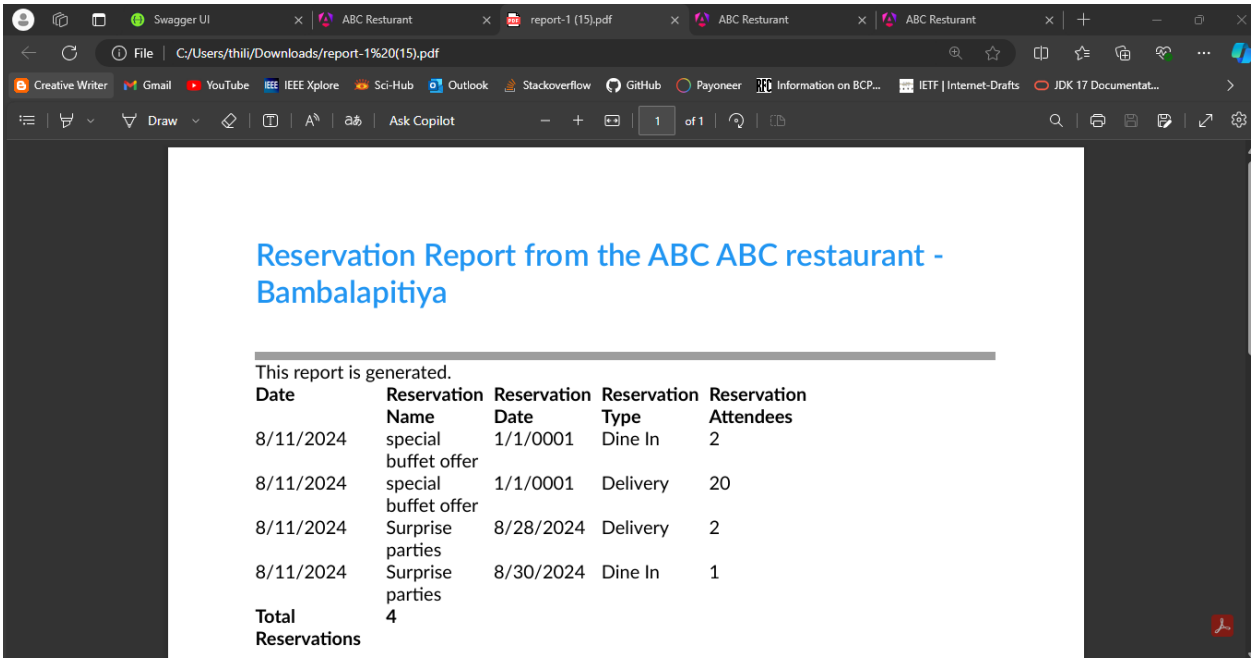
This report is generated.

Date	Profit
ABC restaurant - Bambalapitiya	Rs.33800
ABC restaurant - Kegalle	Rs.55200
ABC restaurant - Ampara	Rs.125600
ABC restaurant - Ja Ela	Rs.61000
ABC restaurant - Anuradhapura	Rs.2100
ABC restaurant - Kalutara	Rs.0
ABC restaurant - Matara	Rs.20000
ABC restaurant - Hambantota	Rs.20000
ABC restaurant - Kollupitiya	Rs.0
Total	**Rs.317700**

Reservation report per restaurant –

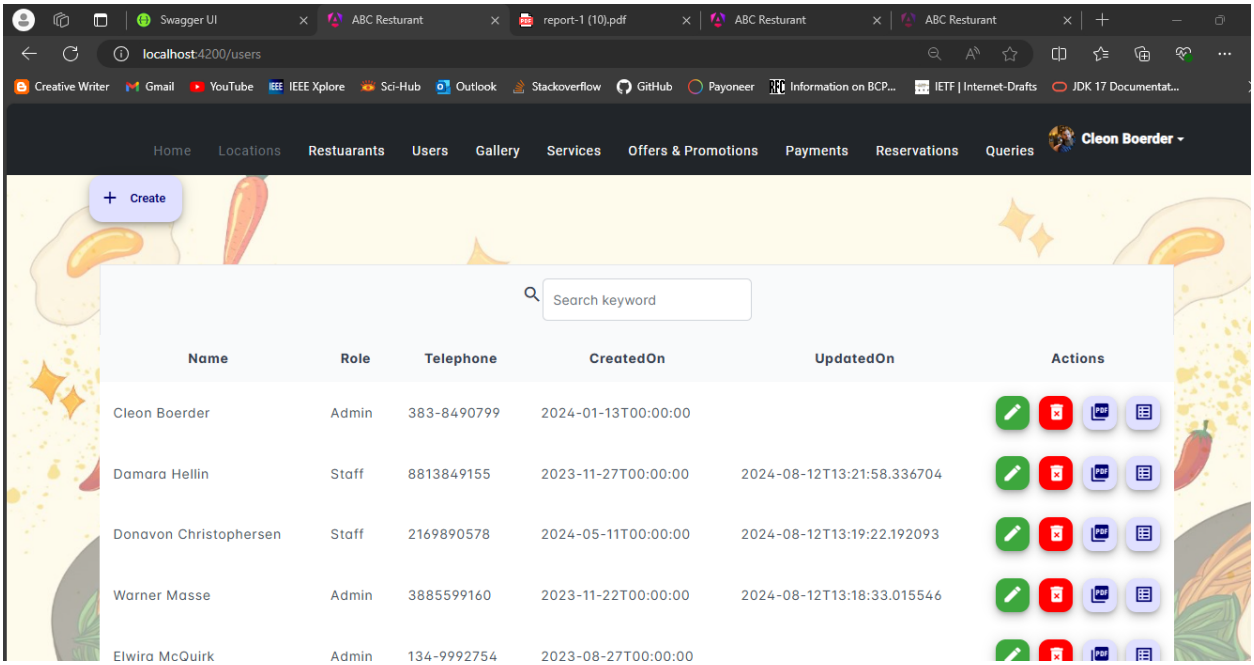
Admin can generate reservation report per restaurant using the report icon in reservations tab.



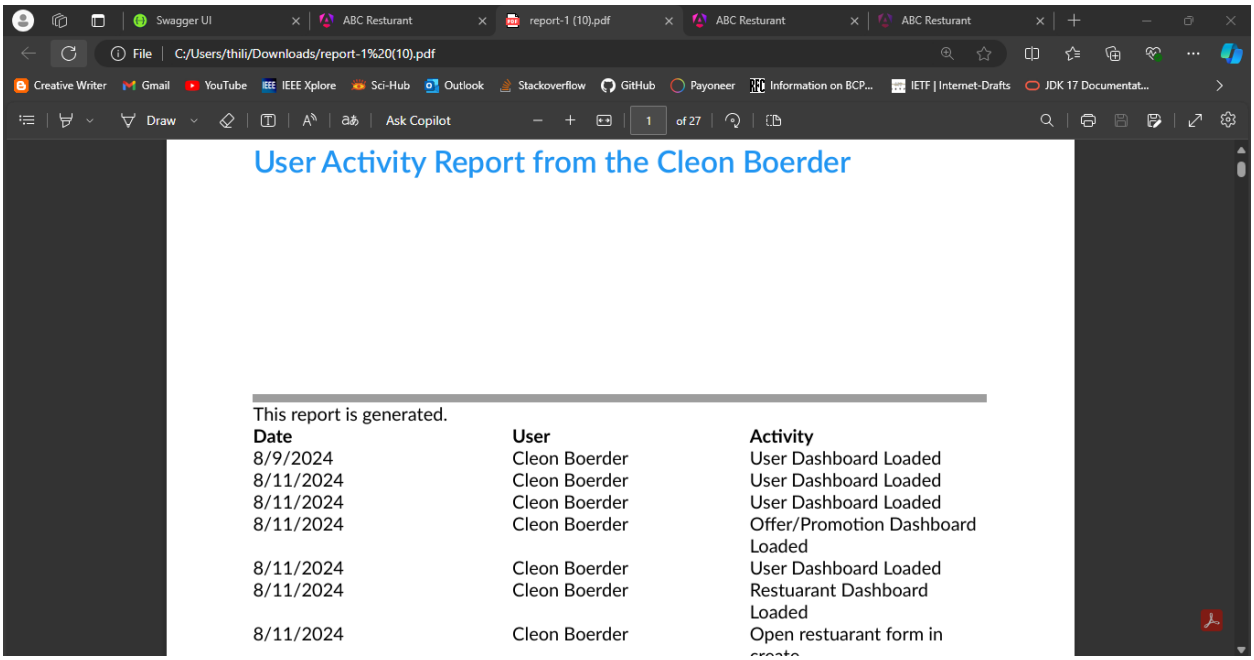


User activity report –

This report includes the date, who the user is and their activity. This can be generated per user by clicking on PDF icon.



This is the report for the admin user



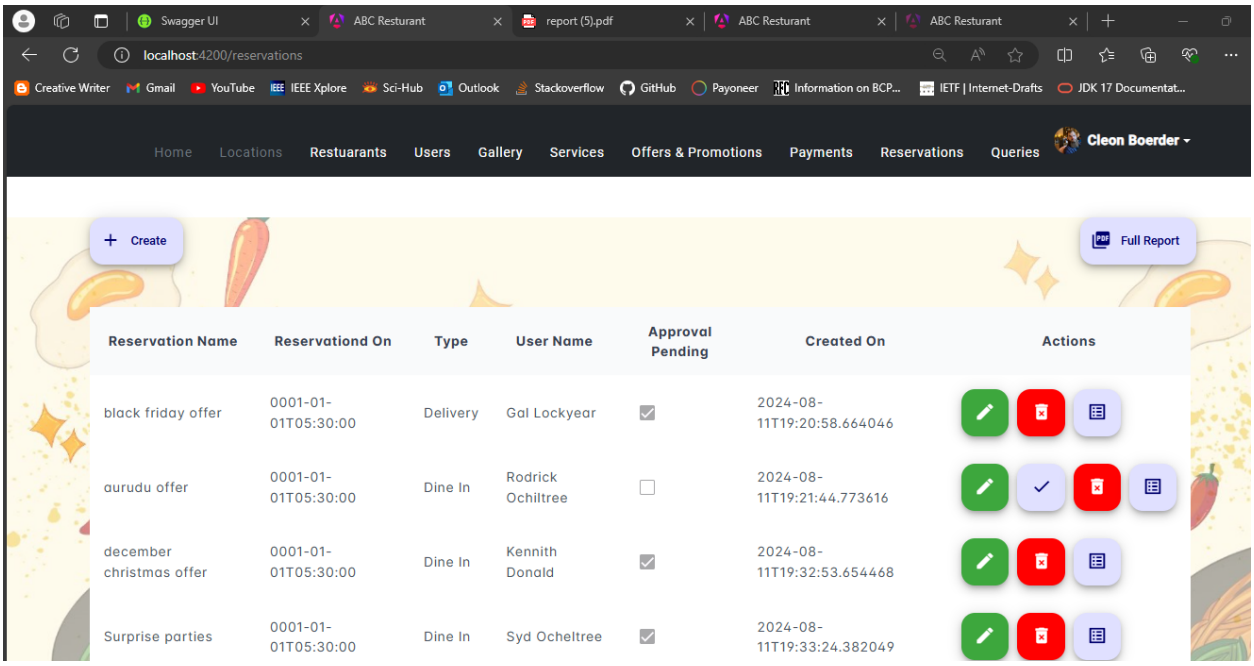
User Activity Report from the Cleon Boerder

This report is generated.

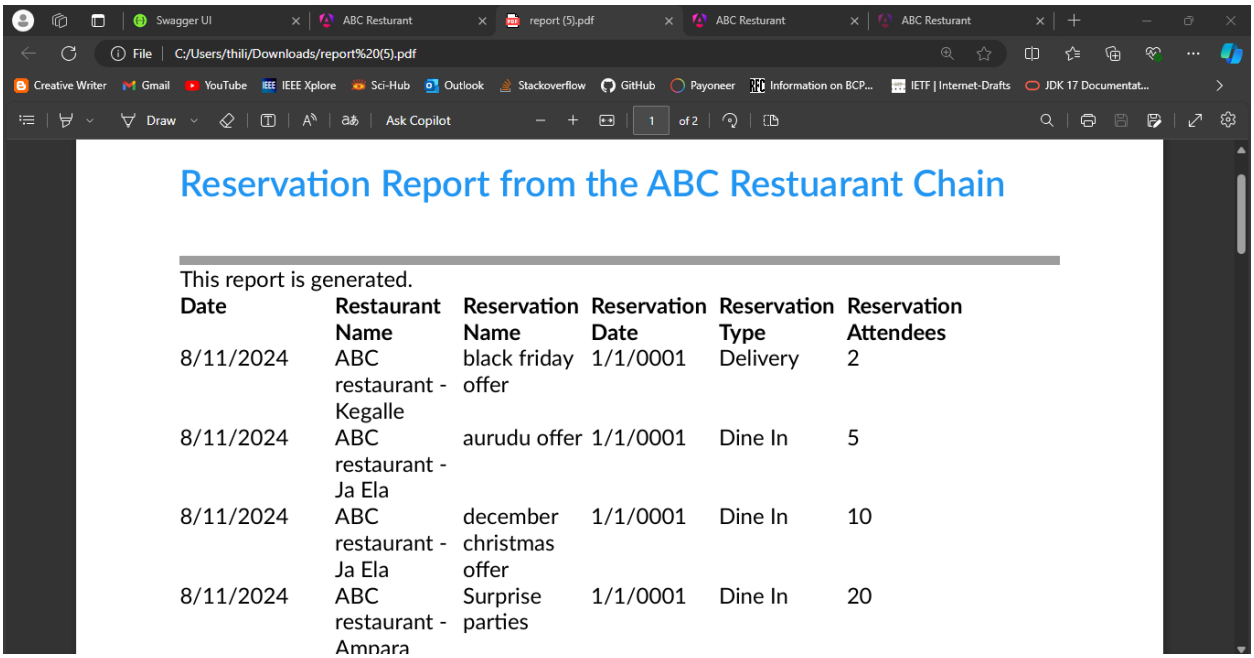
Date	User	Activity
8/9/2024	Cleon Boerder	User Dashboard Loaded
8/11/2024	Cleon Boerder	User Dashboard Loaded
8/11/2024	Cleon Boerder	User Dashboard Loaded
8/11/2024	Cleon Boerder	Offer/Promotion Dashboard Loaded
8/11/2024	Cleon Boerder	User Dashboard Loaded
8/11/2024	Cleon Boerder	Restuarant Dashboard Loaded
8/11/2024	Cleon Boerder	Open restuarant form in

Reservation report from ABC restaurant chain –

Admin and staff have permissions to generate this report by clicking on full report button in reservations.



This includes date, restaurant name, reservation name, reservation date, type and no.of attendees.



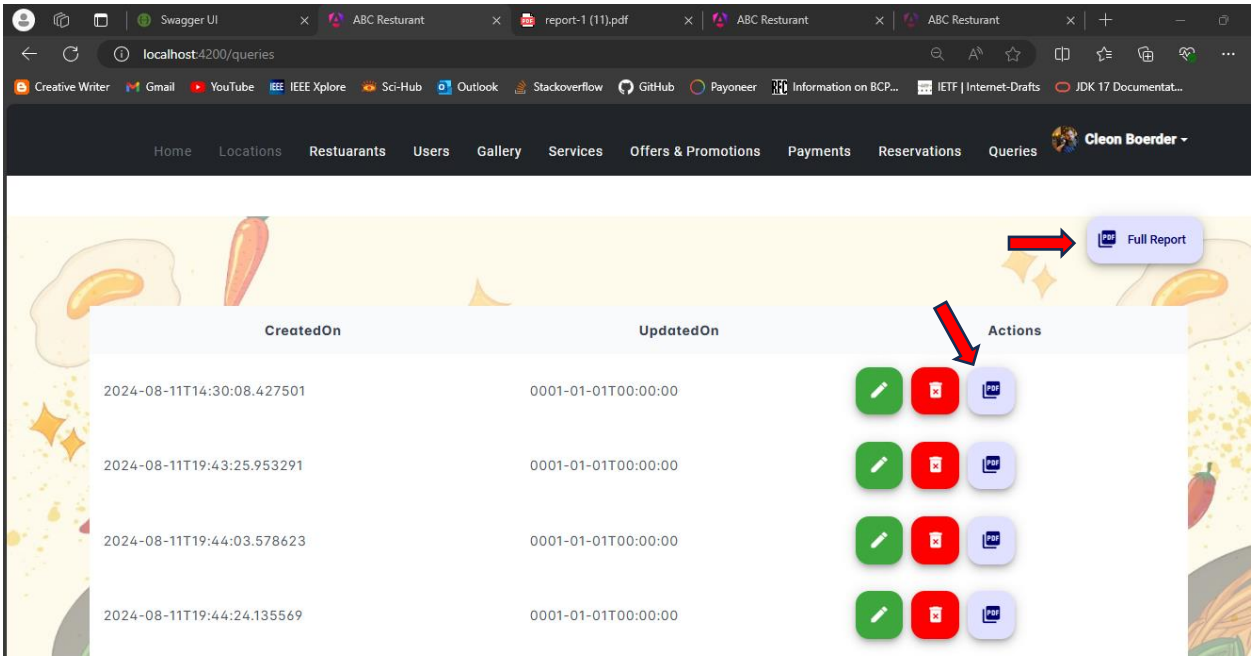
Reservation Report from the ABC Restuarant Chain

This report is generated.

Date	Restaurant Name	Reservation Name	Reservation Date	Reservation Type	Reservation Attendees
8/11/2024	ABC restaurant - Kegalle	black friday offer	1/1/0001	Delivery	2
8/11/2024	ABC restaurant - Ja Ela	aurudu offer	1/1/0001	Dine In	5
8/11/2024	ABC restaurant - Ja Ela	december christmas offer	1/1/0001	Dine In	10
8/11/2024	ABC restaurant - Ampara	Surprise parties	1/1/0001	Dine In	20

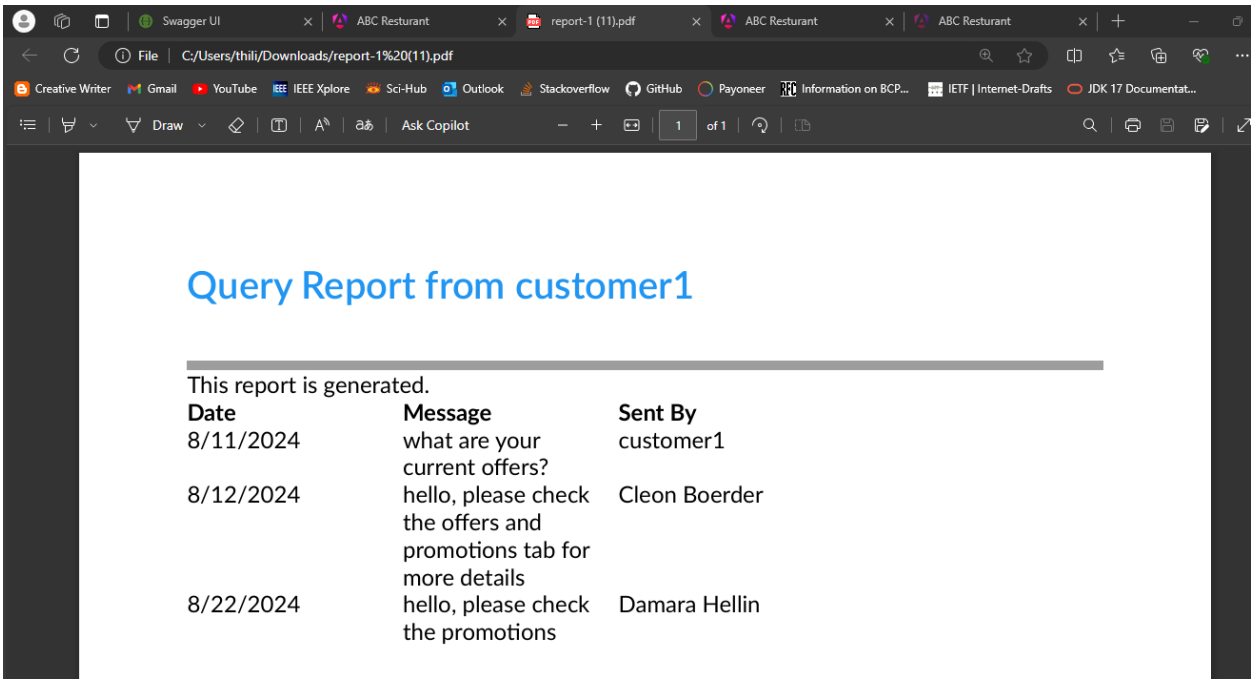
Query reports –

Query reports can be generated per customer and the full report as well. To generate the query report per customer user has to click on the PDF icon under actions in queries tab. Both admin and staff can generate these.



This includes the date, message and replies from the staff and the name of the person who responded.

Query report per customer -



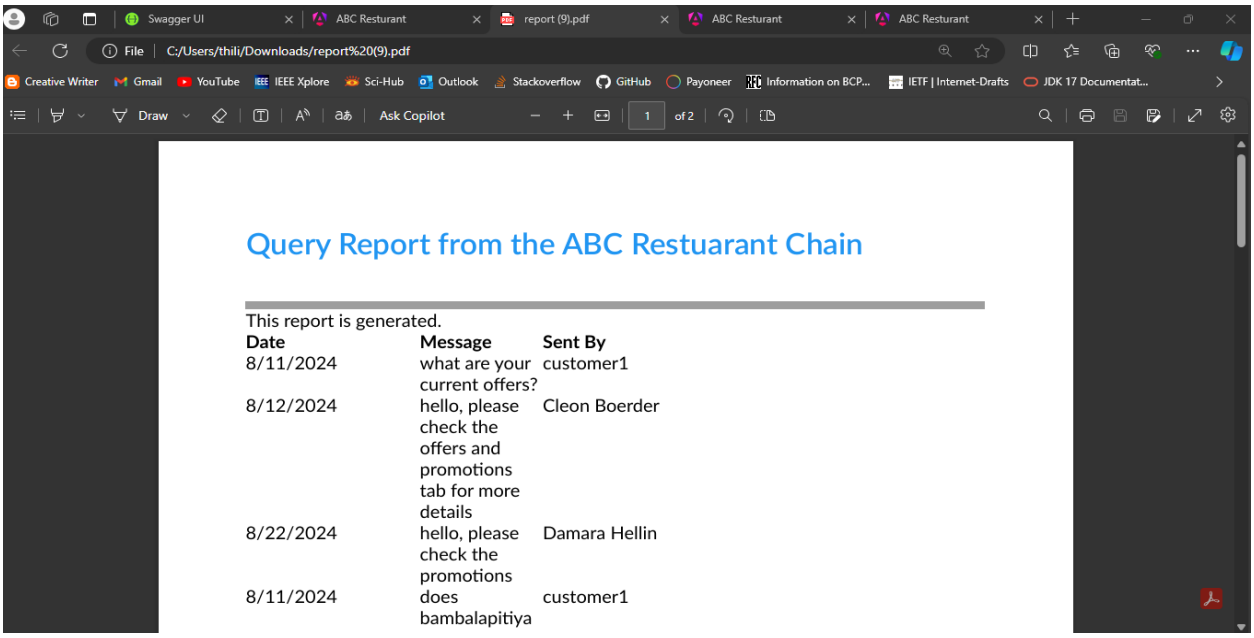
Query Report from customer1

This report is generated.

Date	Message	Sent By
8/11/2024	what are your current offers?	customer1
8/12/2024	hello, please check the offers and promotions tab for more details	Cleon Boerder
8/22/2024	hello, please check the promotions	Damara Hellin

To generate the full query report, user needs to click on full report button. This includes all the queries and their responses with dates. Admin and staff can generate this.

Full query report -



Query Report from the ABC Restuarant Chain

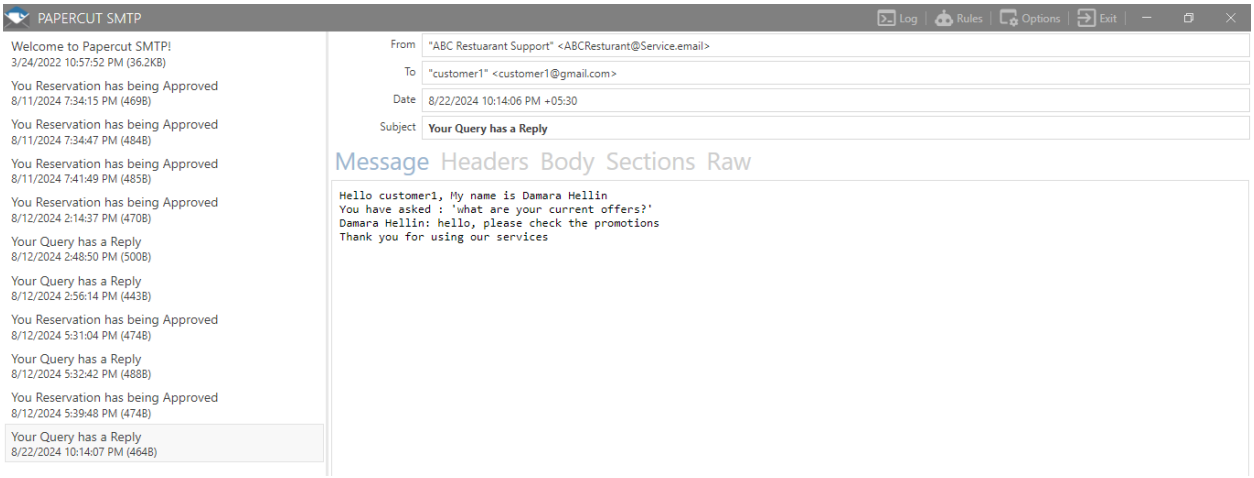
This report is generated.

Date	Message	Sent By
8/11/2024	what are your current offers?	customer1
8/12/2024	hello, please check the offers and promotions tab for more details	Cleon Boerder
8/22/2024	hello, please check the promotions	Damara Hellin
8/11/2024	does bambalapitiya	customer1

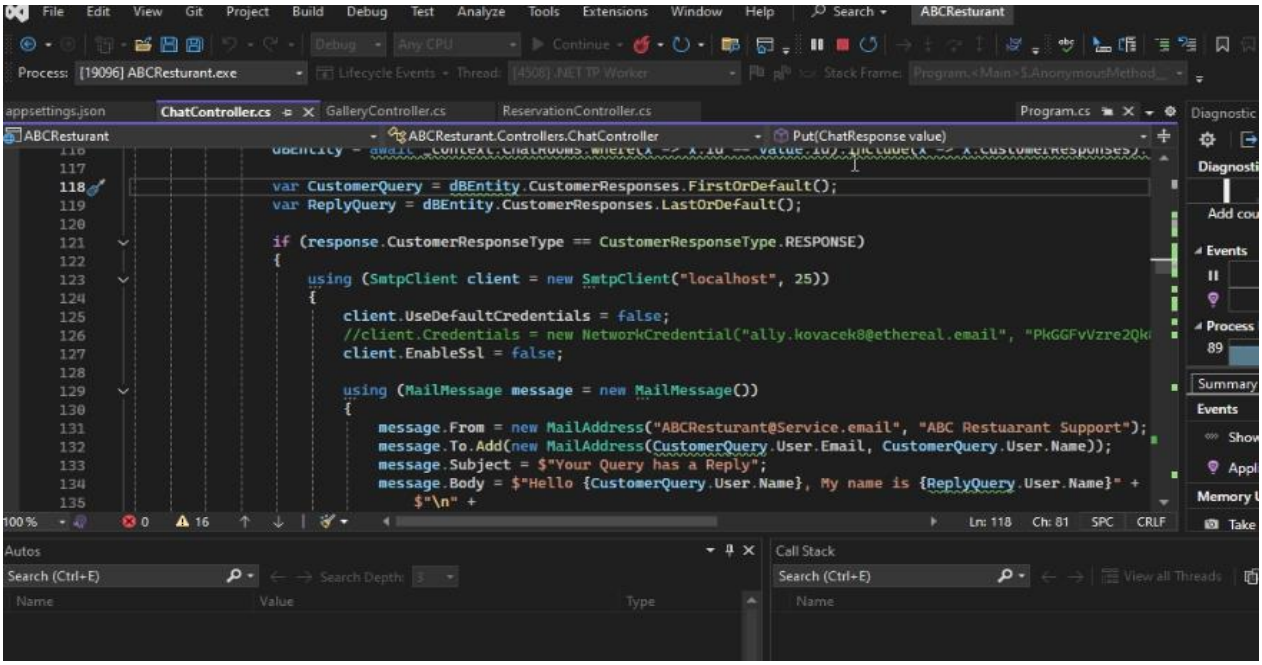
Email Confirmation

Papercut SMTP mail server was used for emailing. This application works as a quick email viewer and a built in SMTP server. The customer receives emails when their reservations are confirmed and once the staff responds to their queries.

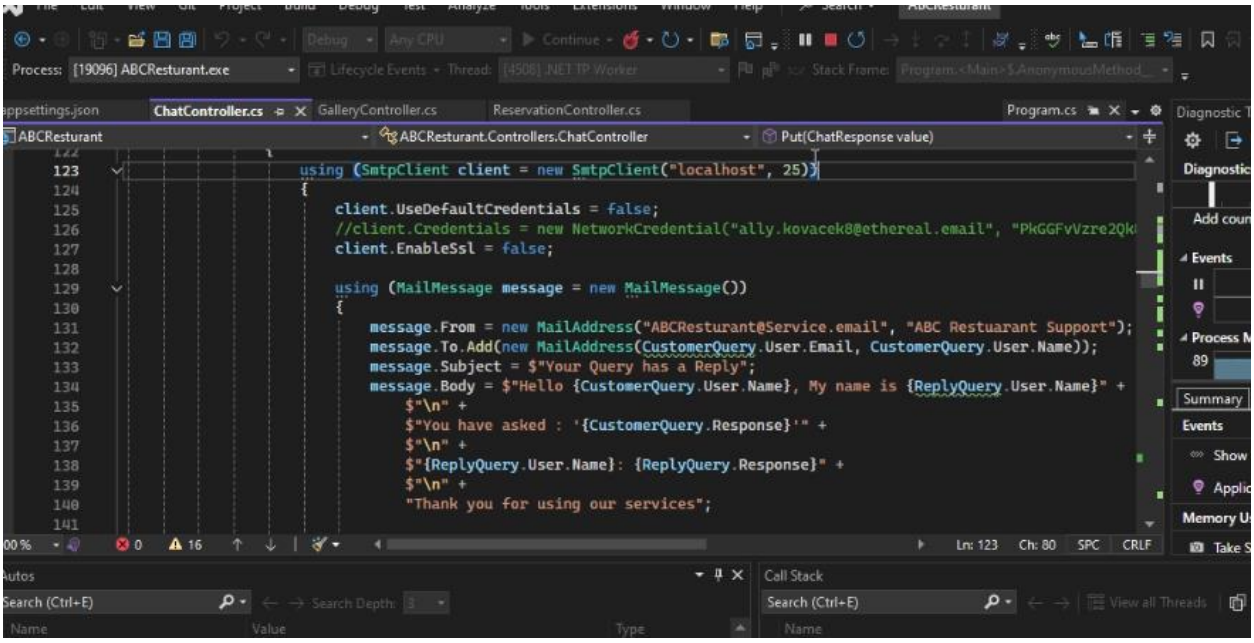
Overview of Papercut SMTP.



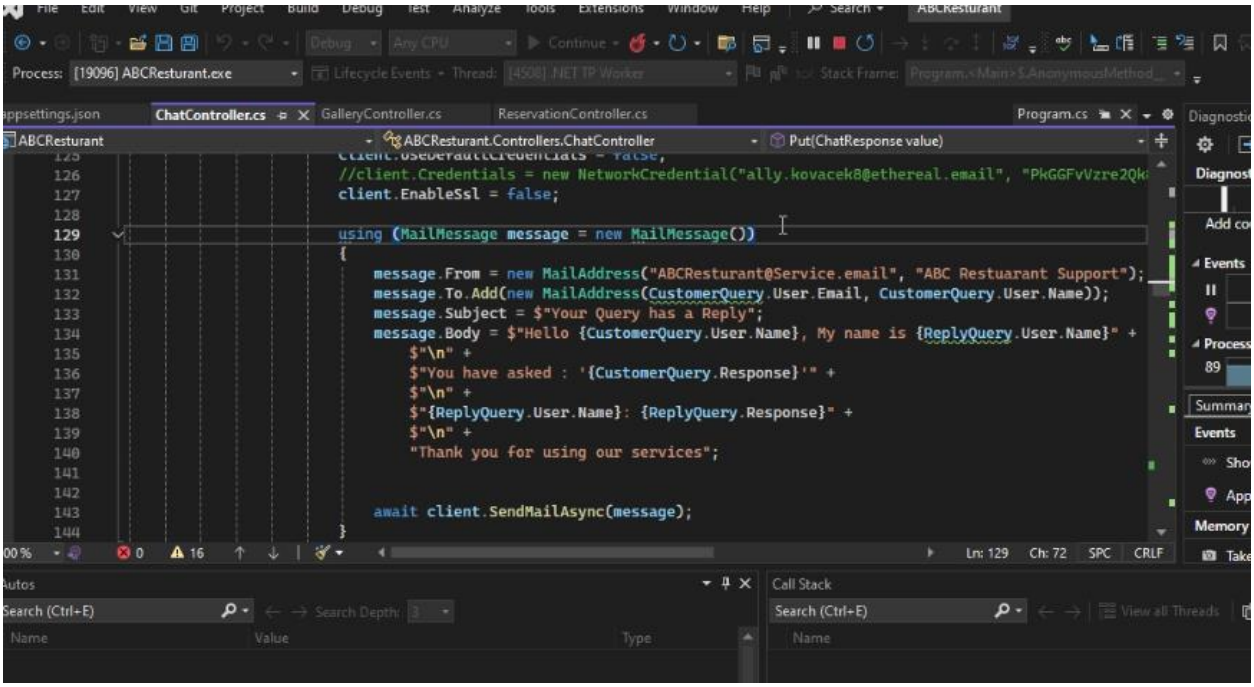
In the ChatController.cs it catches the customer query and the staff response.



Next it opens the SMTP mail application. The host is localhost and the port is 25. It asks not to use the default credentials and asks to disable the SSL as well since we're using the localhost for this.

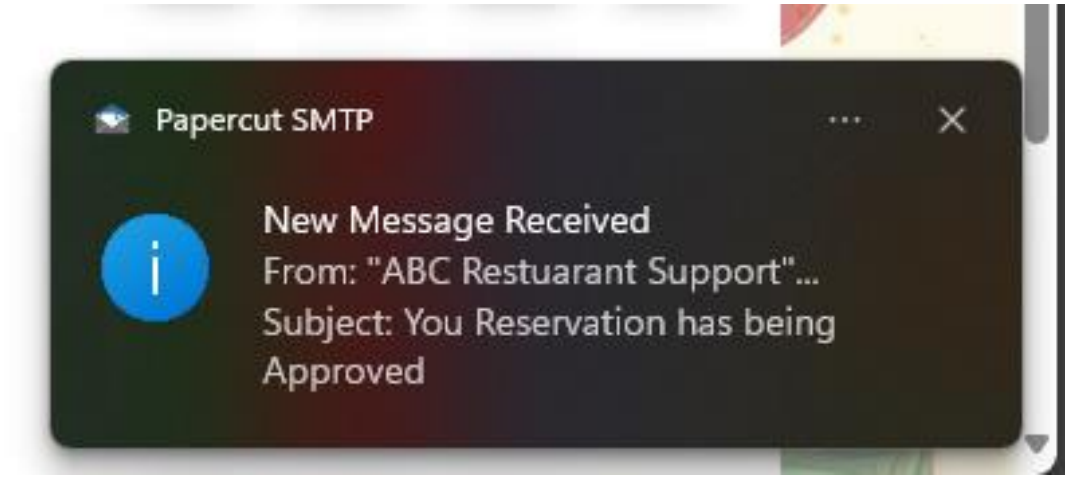


Next it describes what includes in the mail in MailMessage(). It includes From, To, Subject and Body fields. Finally it calls for client.MailAsync(message) and wait for the email to be received.

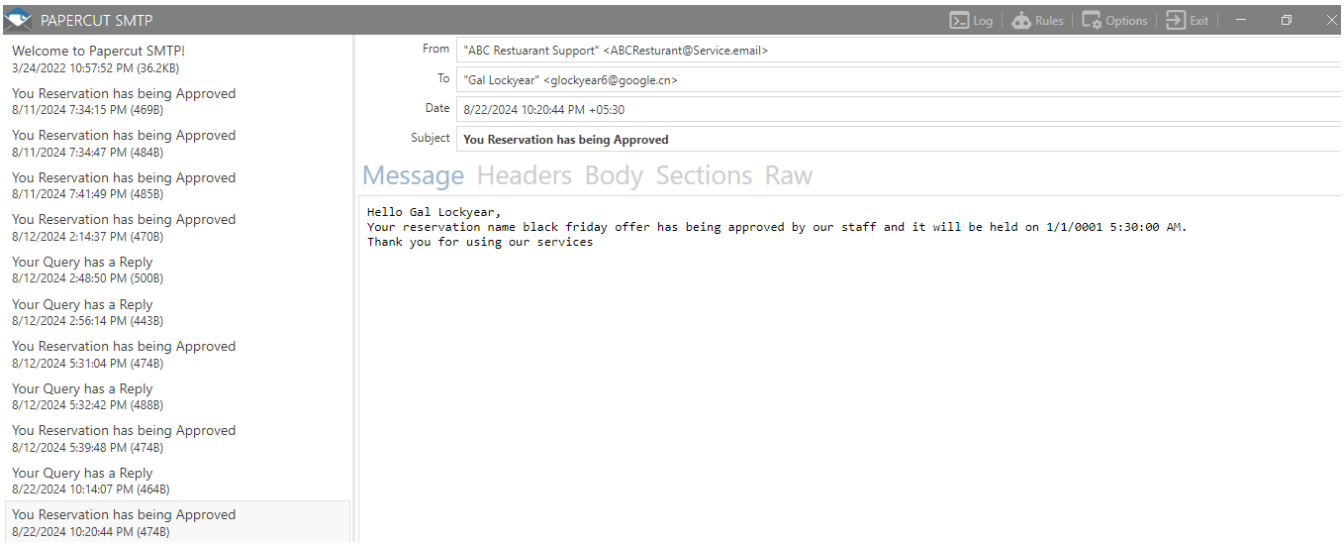


Once a reservation is approved by staff, the particular user gets an email confirmation saying their reservation is confirmed.

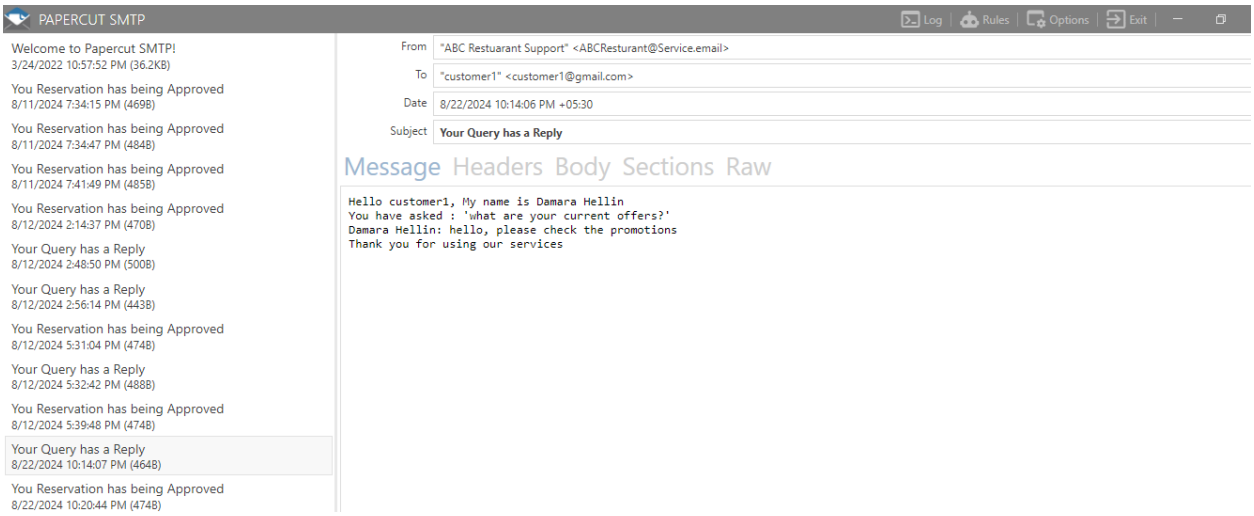
The notification -



The message appears like in the below image.

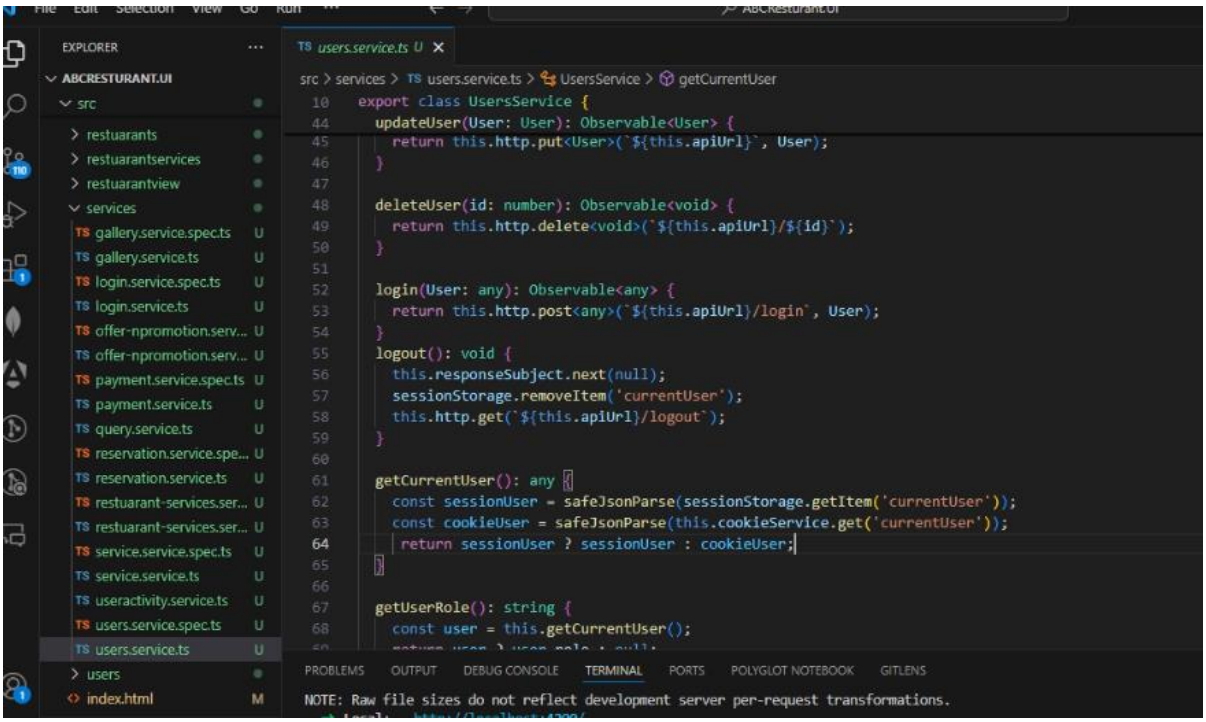


Once a response is received for a query, customer gets an email like below.

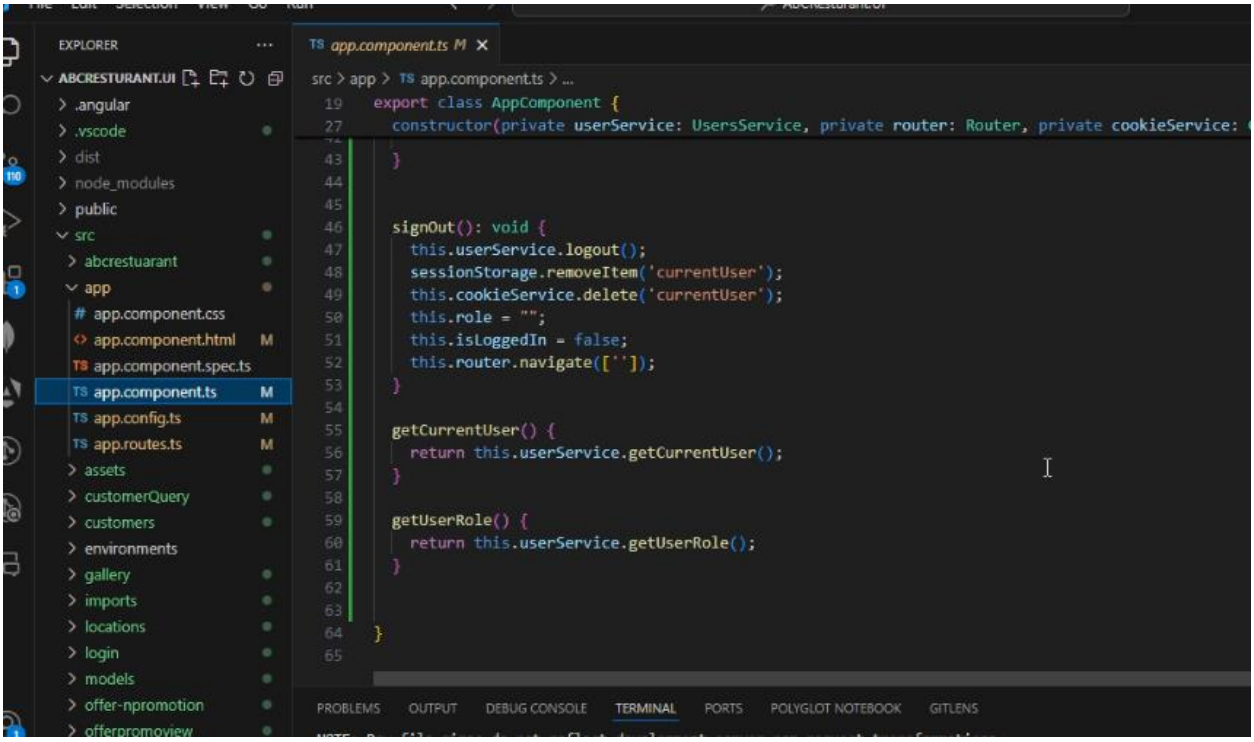


Use of sessions and cookies

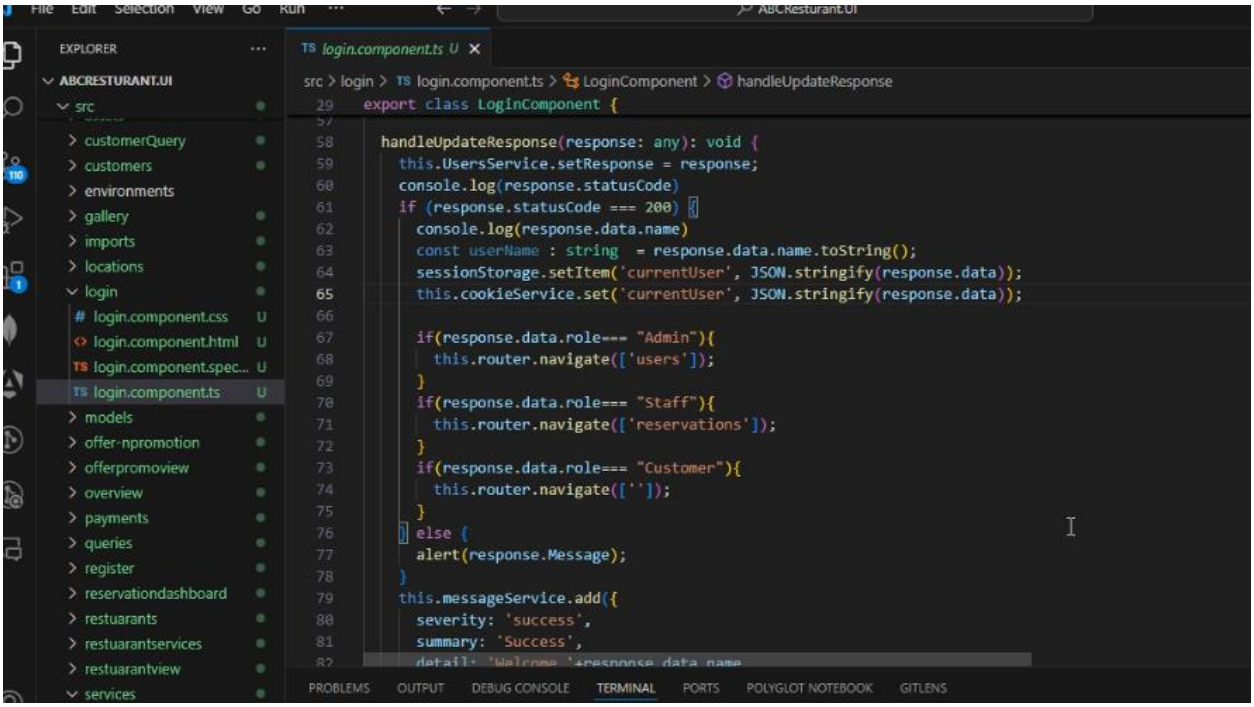
In the users.service.ts, through getCurrentUser(), currentUser session is accessed from the session storage. Same happens with cookies also since the same currentUser is accessing from the cookieService.



When the getCurrentUser is returned, it checks whether the user has a session or cookies. If the user has a cookie, the user is not logged out and they can log back in without entering username and password. If the user has logged out, the cookie should be destroyed. When the sign out is pressed, both the cookie and the session is destroyed.



In the login once the response is set sessionStorage is asked to create a new session called currentUser. If the login is successful (status 200) user's data is stored in cookieService and sessionStorage.



Functional Requirements

Admin operations – admin must be able to do the following.

- Restaurants – view, create, delete, edit, generate reports on restaurants
- Users - view, create, delete, edit, generate reports on users
- Gallery – view, upload, delete gallery
- Services - view, create, delete, edit services
- Offers and promotions – view, create, delete, edit promotions
- Payments – create, view, delete payments
- Reservations – view, create, delete, edit, generate reports, confirm reservations
- Queries – view, respond, delete, generate reports on queries

Staff operations – staff must be able to do the following.

- Payments – create, view, delete payments
- Reservation - view, create, delete, edit, generate reports, confirm reservations
- Queries – view, respond, delete, generate reports on queries

Customer operations – customers must be able to do the following

- Check locations
- Check rates and availability for reservations of each restaurant
- Make reservations
- Search for services
- Check offers and promotions
- Submit queries
- Receive email confirmations for query responses and reservations

All the forms must contain form validations.

Search facility for each table view. (example: search bar in reservations to search reservation records)

Non-functional Requirements

- The system tracks user activities of all the users and generate reports on it.
- The system is scalable in the future and can handle large sets of datasets.
- The system consists of user-friendly interfaces to handle user interactions seamlessly.
- Error messages and confirmation notifications are displayed to quickly resolve issues.
- System is integrated with third-party services (Papercut SMTP for emails)
- System is able to adapt to new business features

TASK C

Quality Assurance and testing

Test driven development was followed. First the test cases were written, then development was done. Once the development is done, the test cases were executed to check whether the system is working as intended.

Manual test case report

Test environment details –
Operating system - Windows 11
Web browser – Microsoft Edge version 128.0.2739.42

Total number of test cases executed – 38
Passed – 38
Failed – 0

Test cases

Test case ID	Module	Test scenario	Test steps	Test data	Expected results	Actual result	Pass/fail
1	Restaurants	Verify create restaurant-admin	1. click restaurants 2. click create restaurant 3. fill form	Name – ABC restaurant - Negombo Location – Negombo	A new restaurant should be created	A new restaurant is created	Pass

			4. click on create	Maximum reservations - 250 Telephone - 1203254693			
2		Verify update restaurant – admin	1. click restaurants 2. click edit restaurant icon 3. edit form fields 4. click on update	Name – ABC restaurant - Hikkaduwa Location – Hikkaduwa Maximum reservations - 100 Telephone - 4521369845	The record should be updated	The record is updated	Pass
3		Verify delete restaurant – admin	1 click restaurants 2 click delete icon 3 confirm deletion by clicking on okay		The restaurant should be deleted	The restaurant is deleted	Pass
4		Verify upload image to restaurant record – admin	1 click restaurant table view 2 click image upload icon 3 select image from computer 4 click upload		The image should be uploaded. Image should be visible in gallery table view and in location.	Image uploaded and visible in gallery table view and location	Pass
5		Verify generate profit report for a restaurant – admin	1 click restaurants 2 click PDF icon for the restaurant you need to generate the report for		The report should be generated and downloaded	The report is generated and downloaded	Pass
6		Verify generate reservation	1 click restaurants 2 click PDF icon for the restaurant		The report should be generated	The report is generated and downloaded	Pass

		report for a restaurant – admin	you need to generate the report for		and downloaded		
7		Verify view restaurant – admin	1 click restaurants 2 click view icon		The view pop up should be visible and all the test fields should be disabled	View pop up appears and the text fields are disabled	Pass
8		Verify generate Profit Report from the ABC Restuarant Chain- admin	1 click restaurants 2 click full report button		The report should be generated and downloaded	The report is generated and downloaded	Pass
9		Verify search option – admin	1 click restaurants 2 enter search term on the search bar	Search term – ABC restaurant - Bambalapitiya	Correct search results should be visible	Correct search results are shown	Pass
10	Users	Verify create customer – admin	1 click users 2 click create 3 fill in details 4 click create	Name – Caroline Perera Email – Caroline@gmail.com Password – caroline@45 Role – customer Telephone - 1234561245	A new customer should be created and only the customer permissions should be available when logged in	A new customer is created and only the customer permissions are available when logged in	Pass
11		Verify create staff – admin	1 click users table view 2 click create 3 fill in details	Name – Freddie Perera	A new staff user should be created	A new staff user is created and only the staff	Pass

			4 click create	Email – fred@gmail.com Password – fred@4521 Role – staff Telephone - 5213601245	and only the staff permissions should be available when logged in	permissions are available when logged in	
12		Verify create admin – admin	1 click users table view 2 click create button 3 fill in details 4 click create	Name – Nelly Nelson Email – nel@gmail.com Password – nelly$nelson Role – staff Telephone - 5236489412	A new admin should be created	The admin is created	Pass
13		Verify edit user – admin	1 click users 2 click edit button 3 edit details 4 click update	Name – Nelly Manson Email – nelmanson@gmail.com Password – nelly$manson Role – staff Telephone - 5236489412	The role should be updated	The role is updated	Pass
14		Verify delete user – admin	1 click users 2 click delete 3 confirm deletion by clicking okay		The user record should be deleted	The user record is deleted	Pass
15		Verify generating user activity report - admin	1 click users 2 click PDF icon next to the user		User activity report should be downloaded	User activity report is downloaded	Pass
16		Verify view users – admin	1 click users 2 click view button		View pop up should be visible. All the	View pop up is visible. All the text fields are disabled	Pass

					text fields should be disabled		
17	Gallery	Verify delete image - admin	1 click gallery 2 click delete button next to image 3 confirm deletion		Image should be deleted	Image is deleted	Pass
18	Services	Verify create service – admin	1 click services 2 click create 3 fill details 4 click create	Name – self service Restaurant – ABC Restaurant-Bambalapitiya	New service should be created. It should be added to the particular restaurant	New service is created. It is added to restaurant	Pass
19		Verify edit service - admin	1 click services 2 click edit service 3 edit details 4 click update	Name – all-you-can-eat-service Restaurant – ABC Restaurant-Galle	Service should be updated	Service is updated	Pass
20		Verify delete service – admin	1 click services 2 click delete button 3 confirm deletion		Service should be deleted	Service is deleted	Pass
21	Offers and promotions	Verify create offer/promotion – admin	1 click offers and promotions 2 click create 3 fill details 4 click create	Message – lunch buffet offer Code – lunch20 Discount – 20 Offer until – 2024.09.30	Offer should be created	Offer is created	Pass
22		Verify edit offer/promotion – admin	1 click offers and promotions 2 click edit 3 edit details 4 click update	Message - buffet offer Code – lunch25 Discount – 25	Offer should be updated	Offer is updated	Pass

				Offer until – 2024.10.30			
23		Verify delete offer/promotion – admin	1 click offers and promotions 2 click delete 3 confirm deletion		Offer should be deleted	Offer is deleted	Pass
24	Payments	Verify create payment – admin, staff	1 click payments 2 click create	Restaurant – ABC Restaurant-Bambalapitiya Payment – 3200 Customer – customer1	Payment should be created	Payment is created	Pass
25		Verify delete payment – admin, staff	1 click payments 2 click delete		Payment should be deleted	Payment is deleted	Pass
26	Reservations	Verify create reservation – admin, staff	1 click reservations 2 click create 3 fill details 4 click create	Name – birthday party Date – 2024.09.01 Customer – customer1 Reservation type – dine-in Restaurant No.of people - 25	Reservation should be created. Make reservation button should be enabled for the particular restaurant	Reservation is created. Make reservation button is available	Pass
27		Verify edit reservation – admin, staff	1 click reservations 2 click edit 3 edit fields 4 click update	Name – bridal shower Date – 2024.09.02 Customer – customer2 Reservation type – dine-in Restaurant No.of people - 30	Reservation should be updated	Reservation is updated	Pass

28		Verify delete reservation – admin, staff	1 click reservations 2 click delete 3 confirm deletion		Reservation should be deleted	Reservation is deleted	Pass
29		Verify confirm reservation – admin, staff	1 click reservations 2 click confirm button		Reservation should be confirmed, confirmation email should be sent to customer	Reservation is confirmed, confirmation email is sent	Pass
30		Verify delete reservation – admin, staff	1 click reservations 2 click delete 3 confirm deletion		Reservation should be deleted	Reservation is deleted	Pass
31		Verify generate Reservation Report from the ABC Restuarant Chain – admin, staff	1 click reservations 2 click full report		Report should be downloaded	Report is downloaded	Pass
32	Queries	Verify respond to query – admin, staff	1 click queries 2 click respond icon 3 write response 4 click submit	Response – please check our website	Response should be submitted; email should be sent to customer	Response is submitted, email is sent	Pass
33		Verify delete query – admin, staff	1 click queries 2 click delete 3 confirm deletion		Query should be deleted	Query is deleted	Pass

34		Verify generate query report per customer – admin, staff	1 click queries 2 click PDF icon		Report should be downloaded	Report is downloaded	Pass
35		Verify generate Query Report from the ABC Restaurant Chain – admin, staff	1 click queries 2 click full report icon		Report should be downloaded	Report is downloaded	Pass
36	Locations	Verify check locations – admin, staff, customer	1 click locations 2 choose location and click view details		Users should be able to view details of location	Details can be viewed	Pass
37	Ask us	Verify submit query – customer	1 click ask us 2 type query 3 click submit	Query – do you deliver food?	Query should be submitted and should be viewed by staff	Query is submitted and can be viewed by admin and staff	Pass
38	Make reservation	Verify make reservation – customer	1 click locations 2 choose location, click view details 3 click make reservation 4 fill form 5 click pay	Name – engagement party Date – 2024.10.12 Reservation type – dine-in No.of people attending – 50	Reservation should be made successfully. Amount to pay should be calculated correctly based on	Reservation is made and amount is calculated correctly	Pass

				Offer/promo code – september20	no.of people, current rate, promo code		

Test Automation

Operating system - Windows 11

Web browser – Microsoft Edge version 128.0.2739.42

Test automation tool – Katalon studio 9.5.0

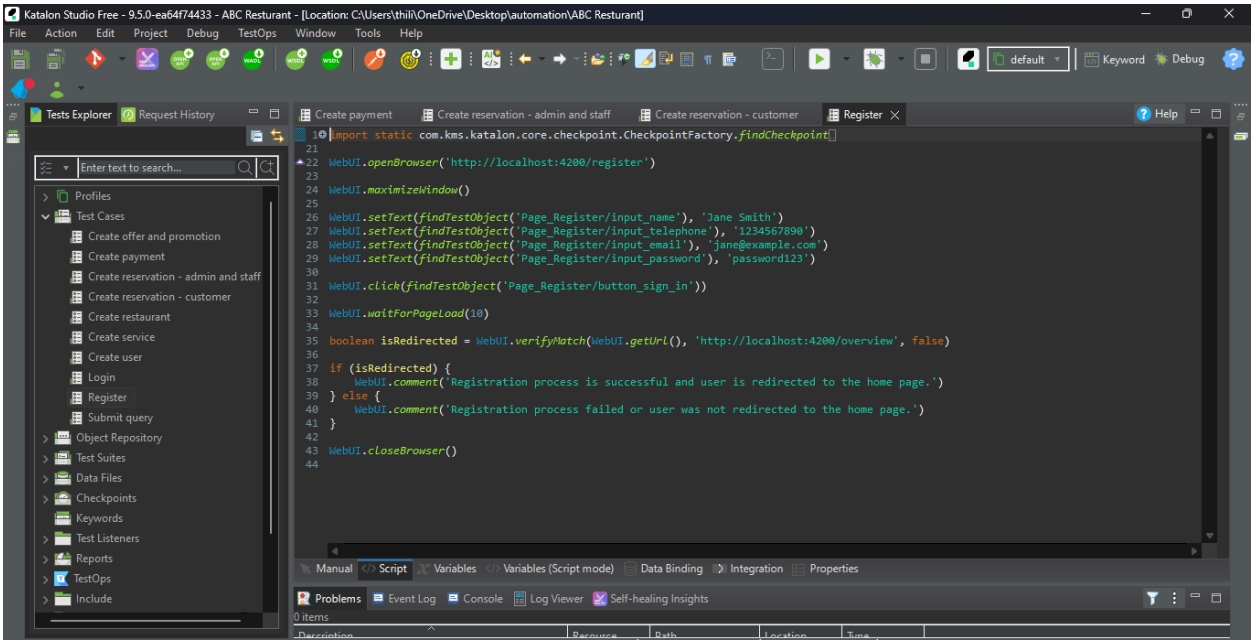
Total number of test cases executed – 10

Passed – 10

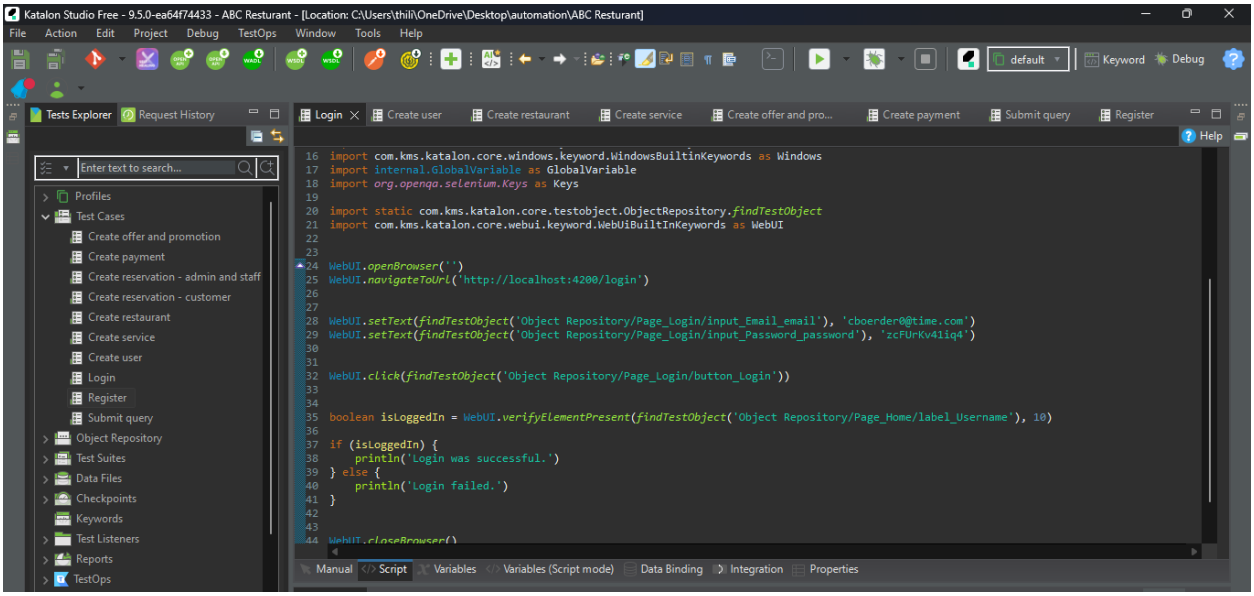
Failed – 0

Following are the test scripts for every test case coded in Groovy in Katalon studio.

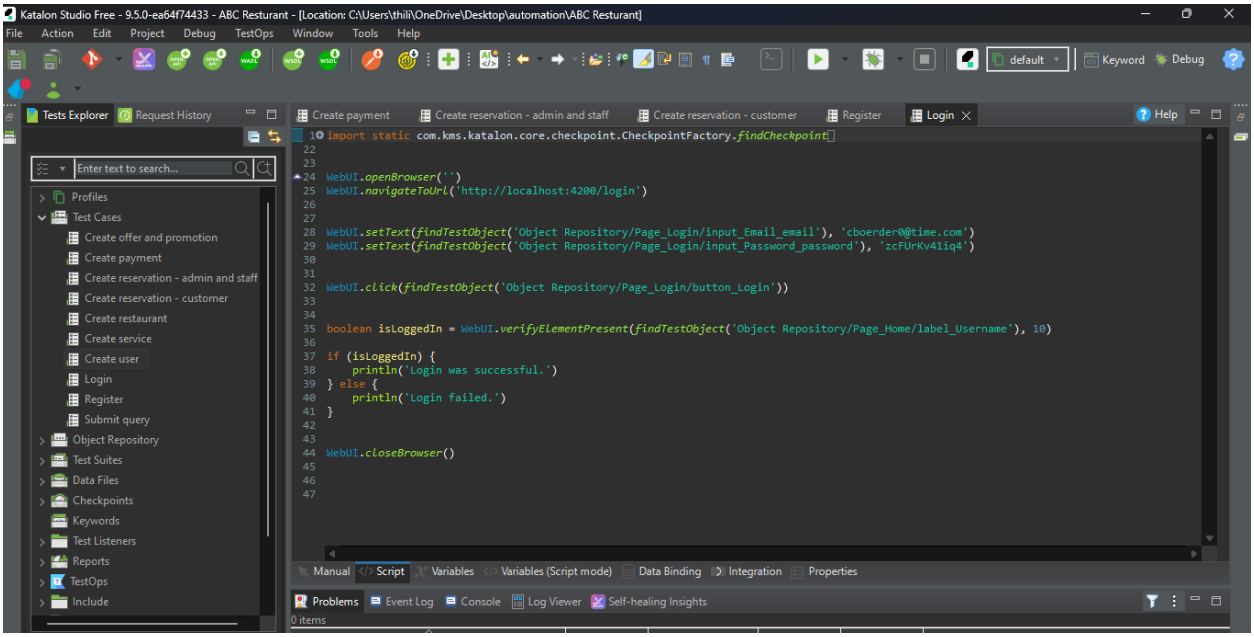
Register



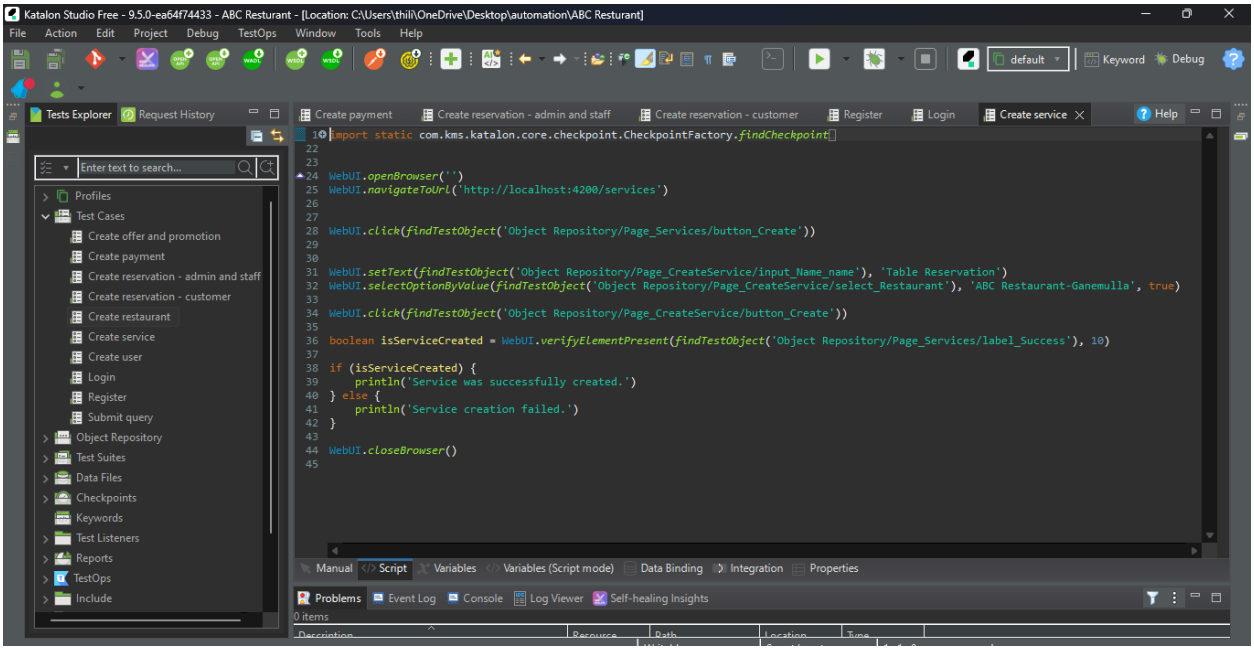
Login



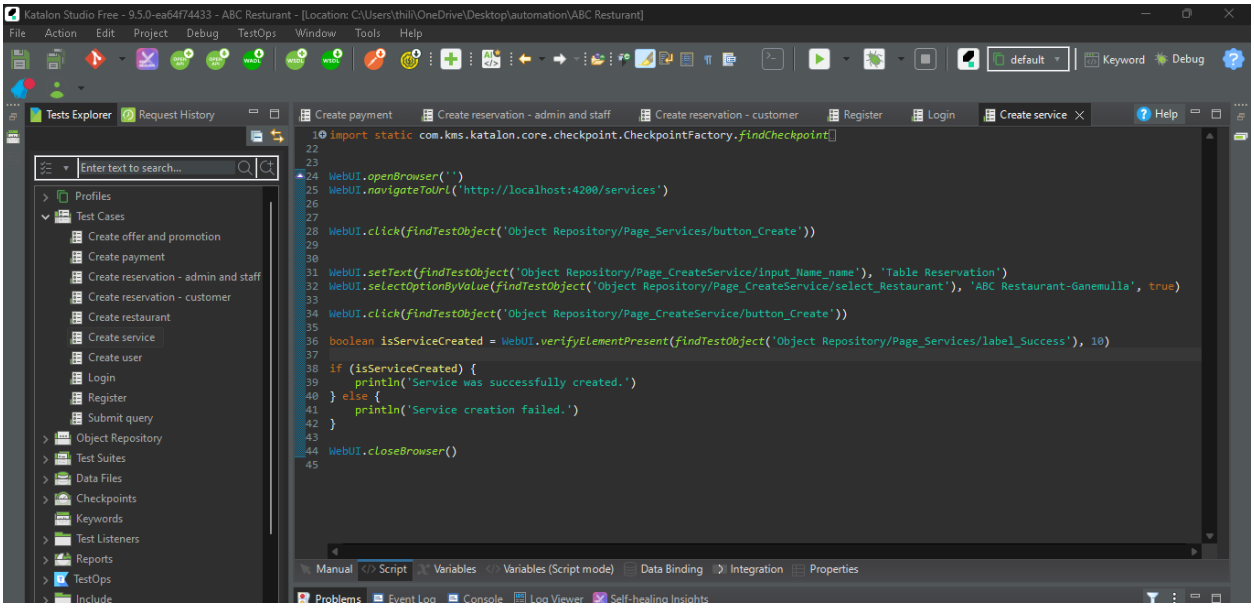
Create user



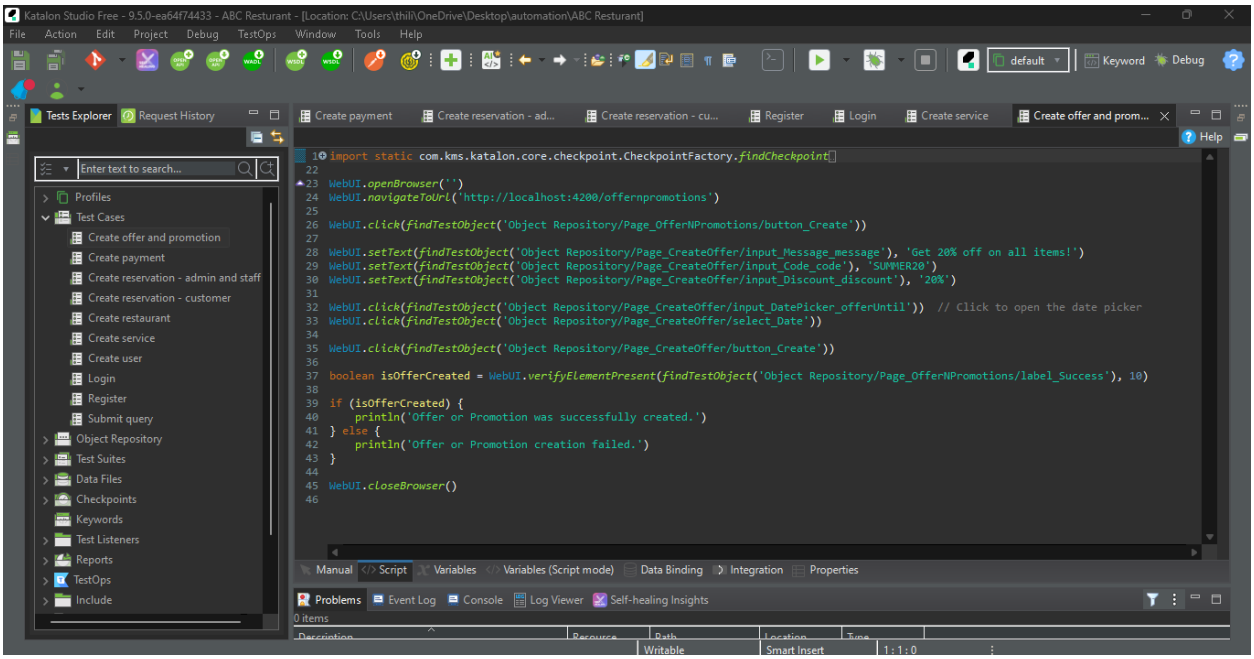
Create restaurant



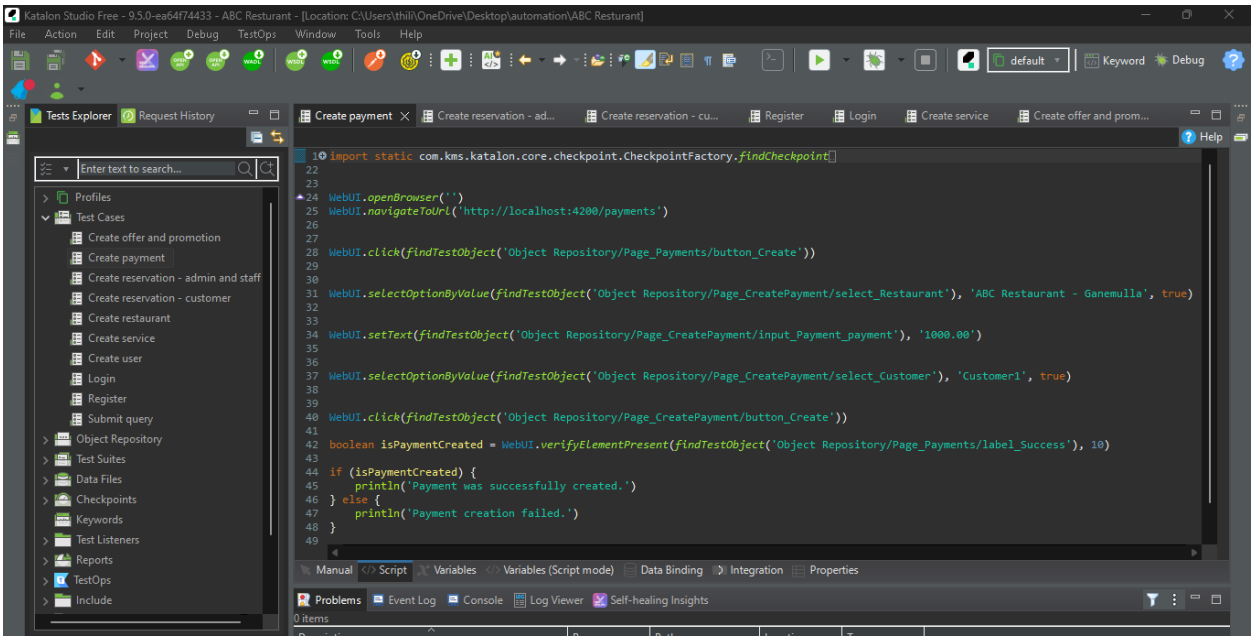
Create service



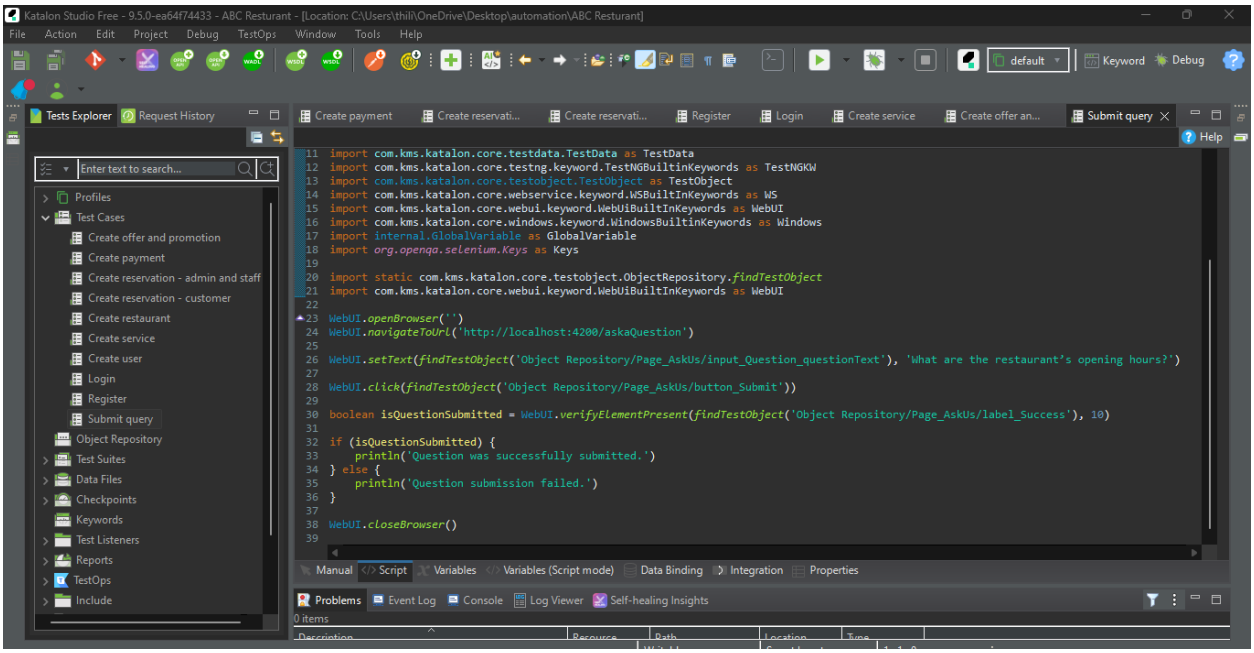
Create offer and promotion



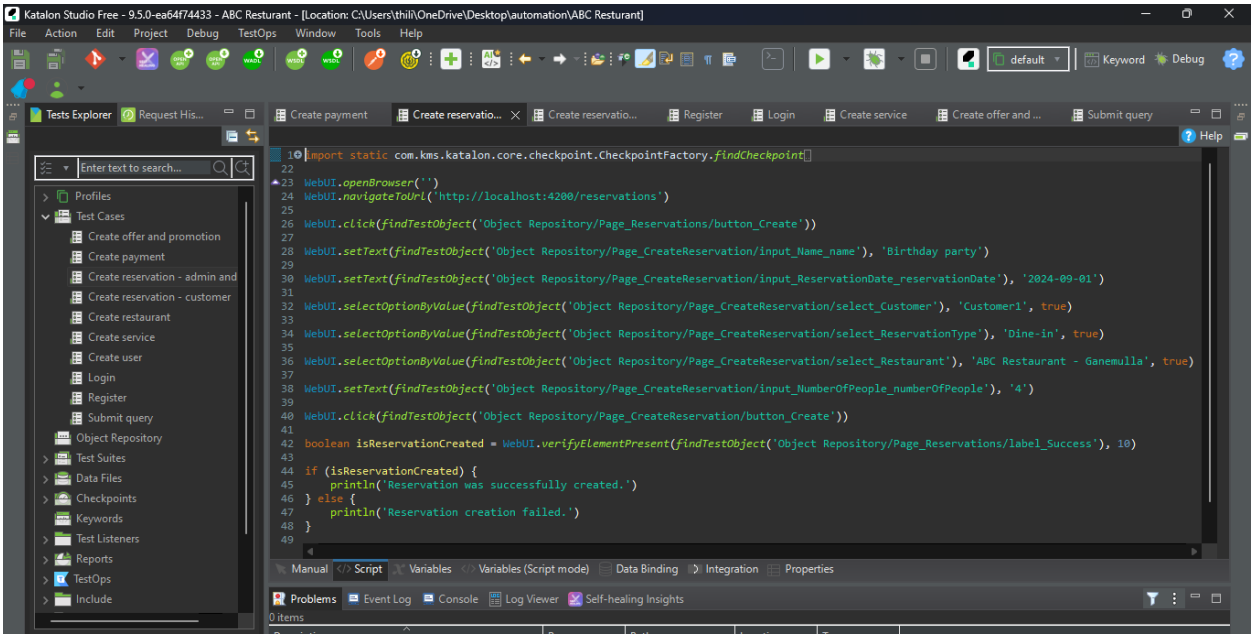
Create payment



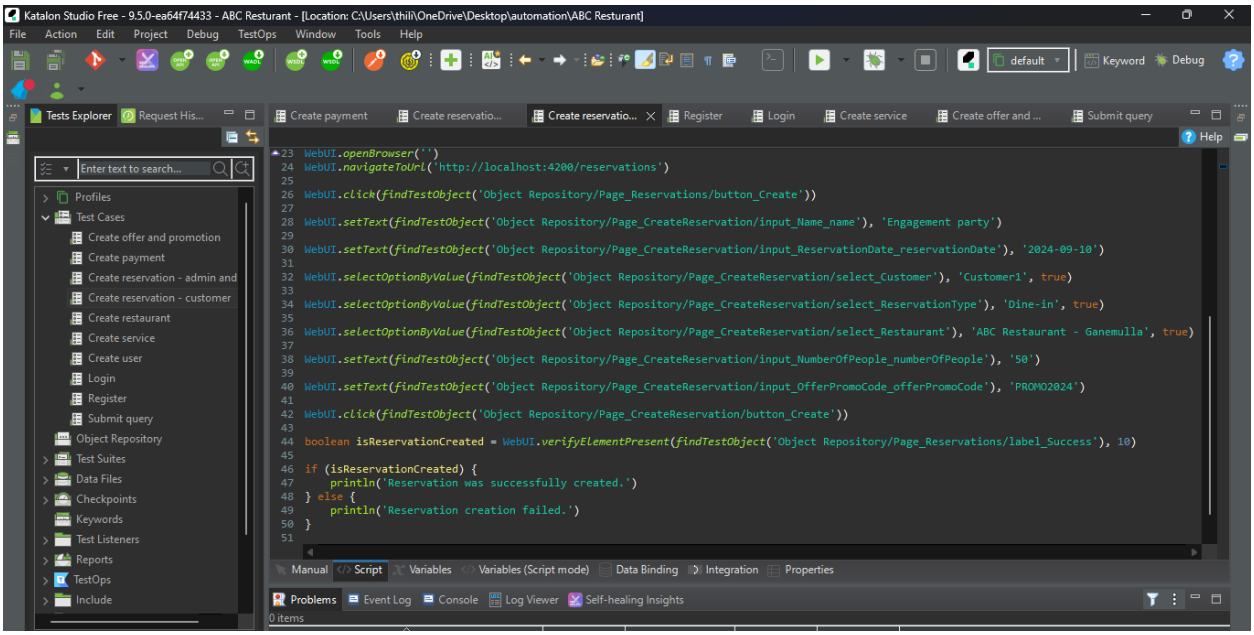
Submit query



Create reservation – admin, staff



Create reservation - customer



TASK D

Repository Management

Git and Github were used for version control.

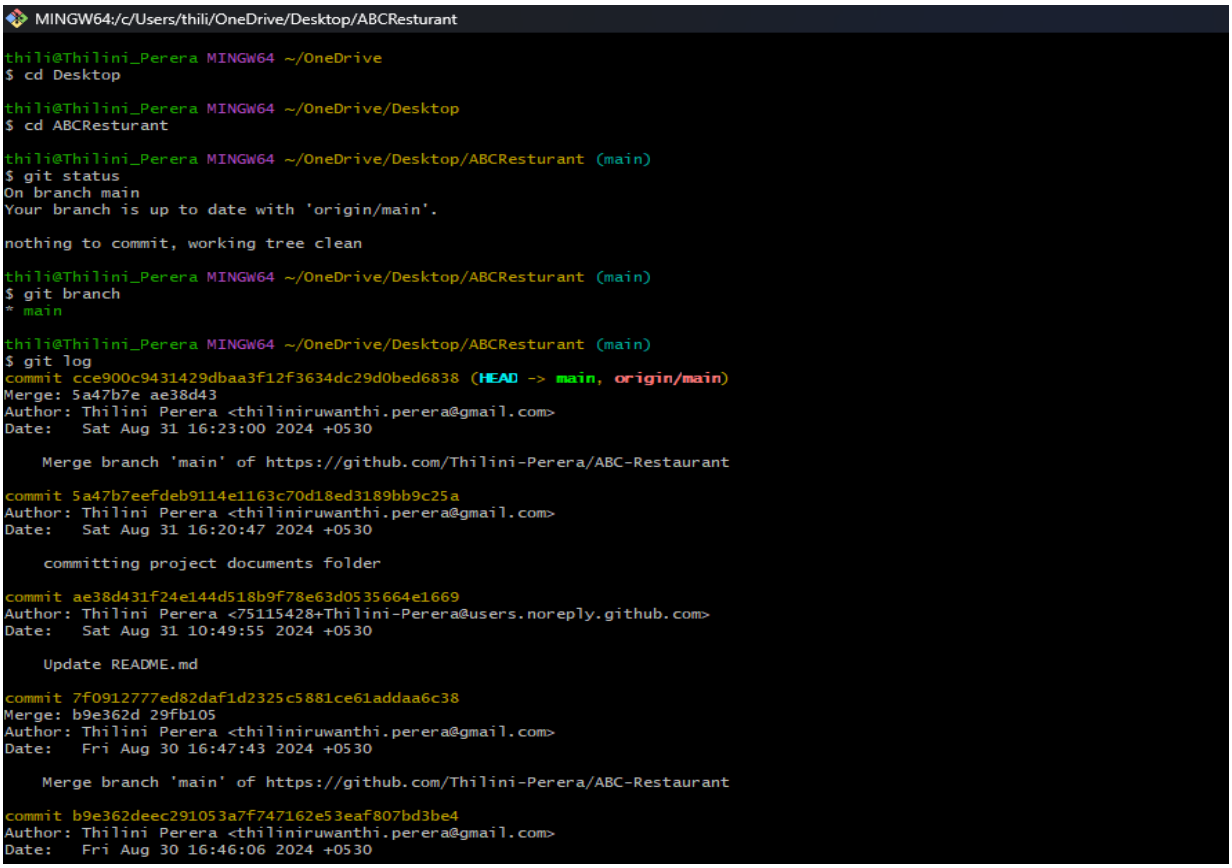
GitHub repository (including report, UML diagrams) –

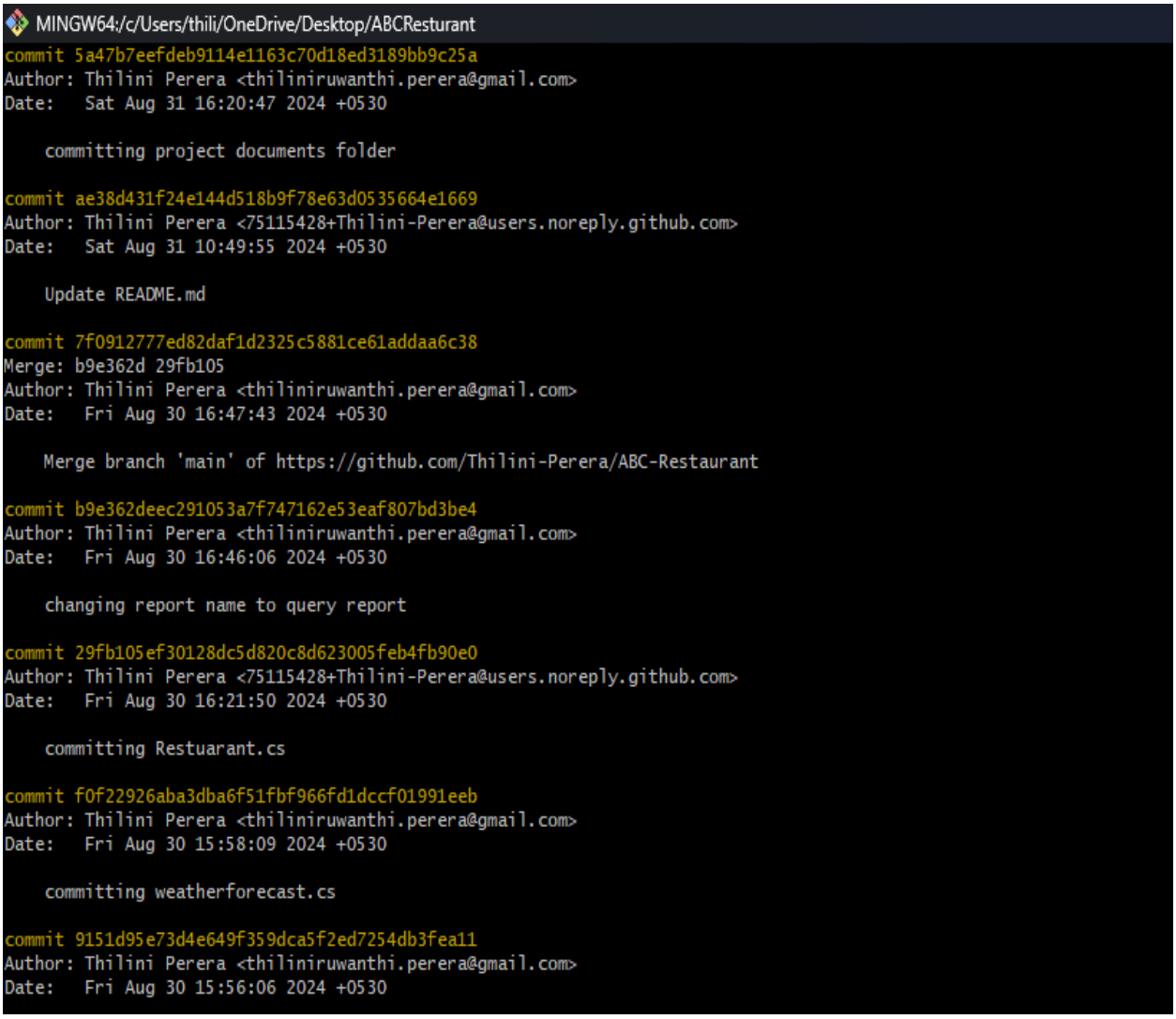
https://github.com/Thilini-Perera/ABC-Restaurant

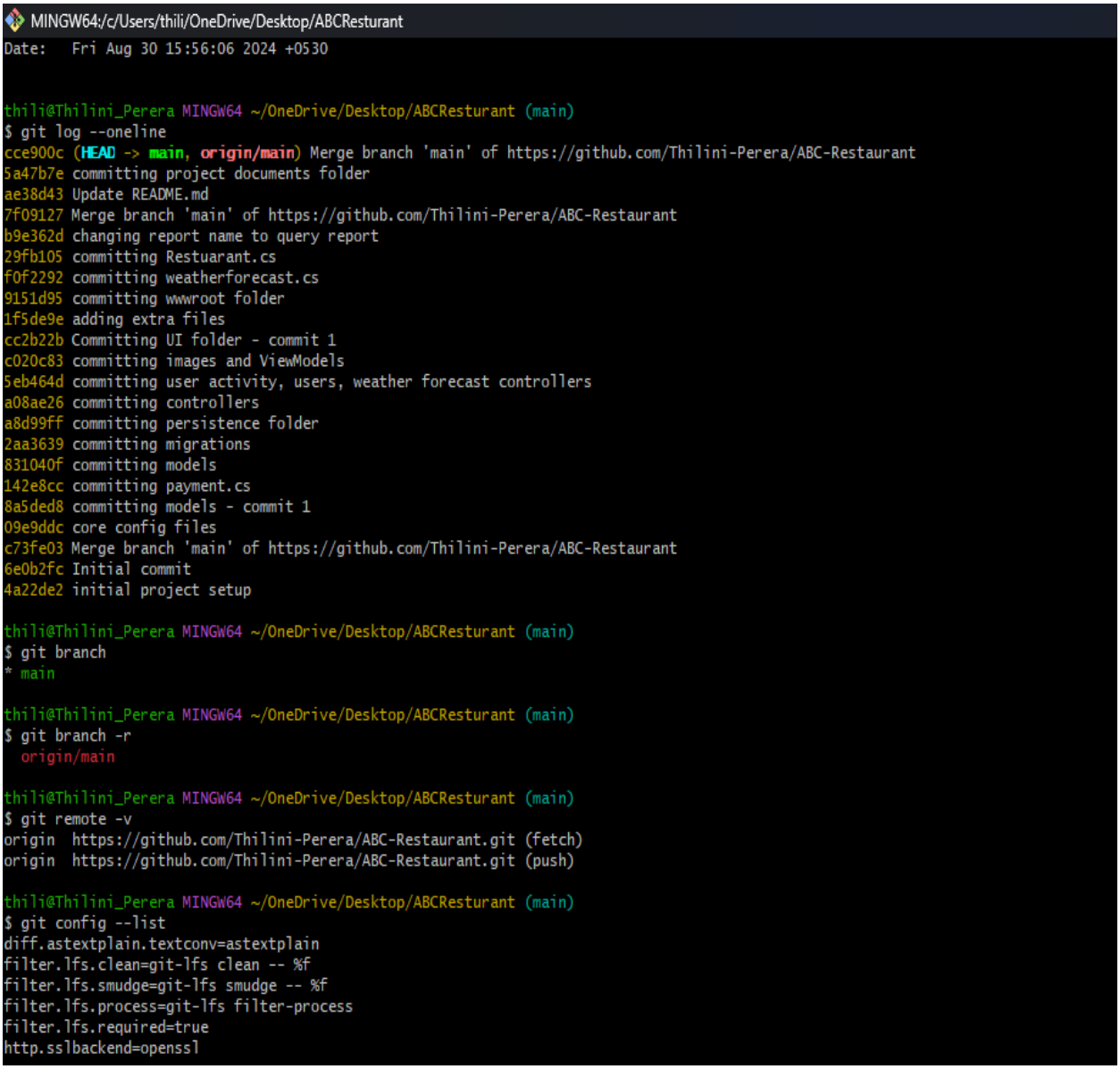
GitHub repository for automation –

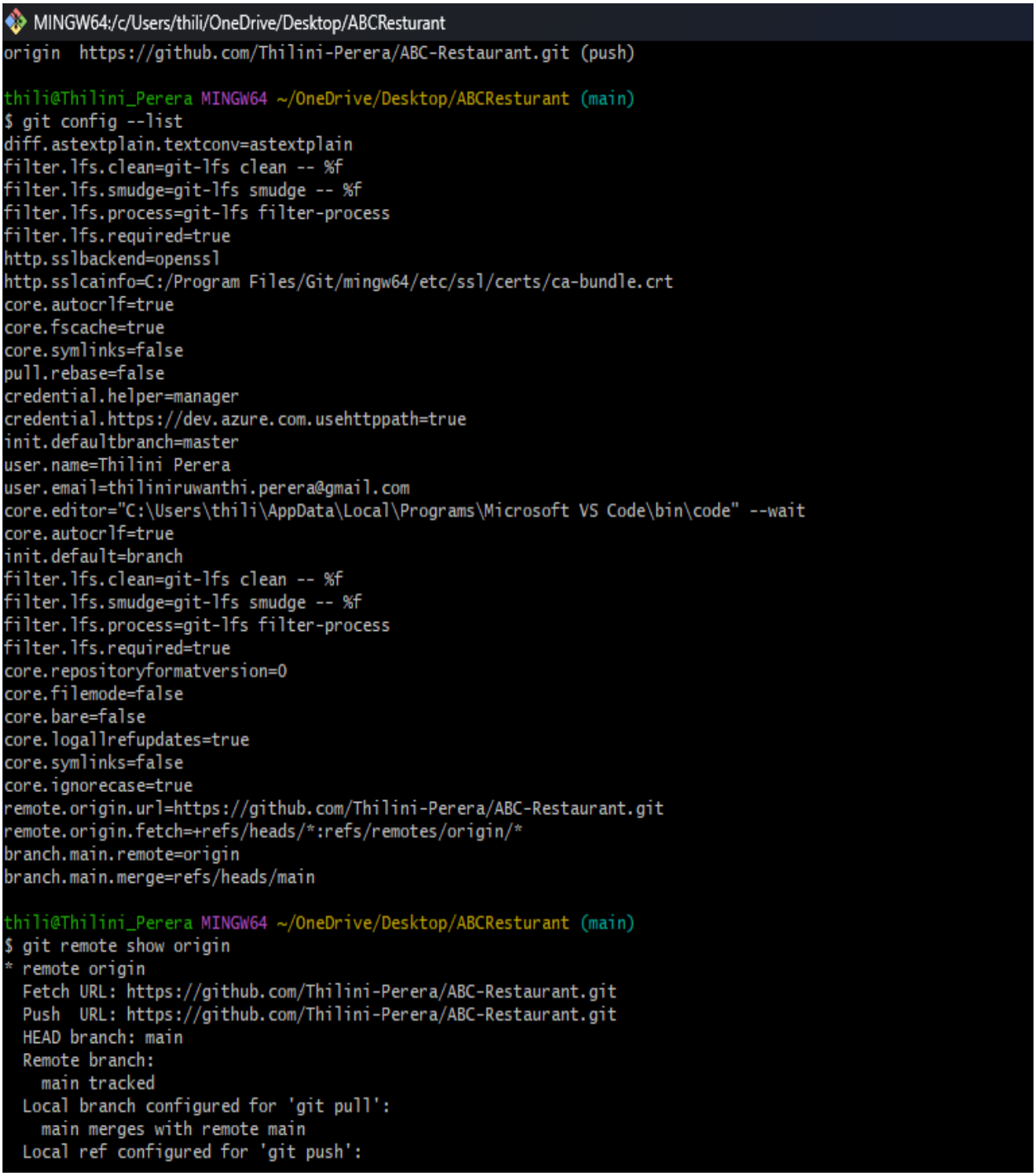
https://github.com/Thilini-Perera/ABC_Restaurant---automation

Local Git repository

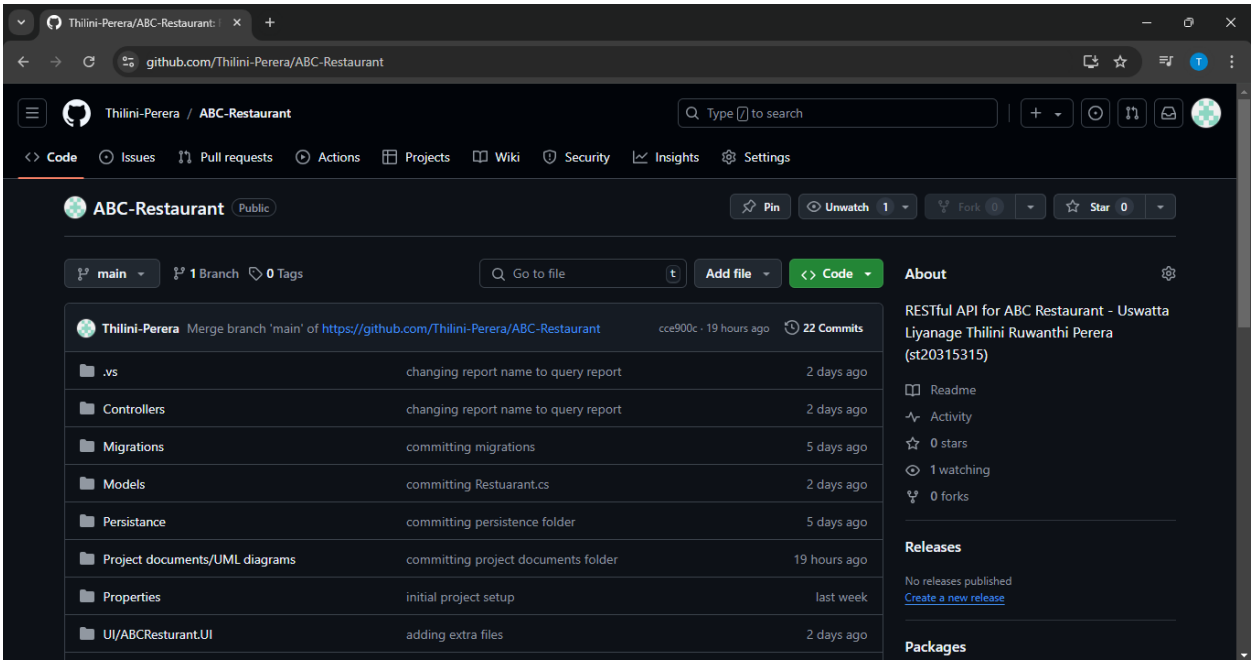


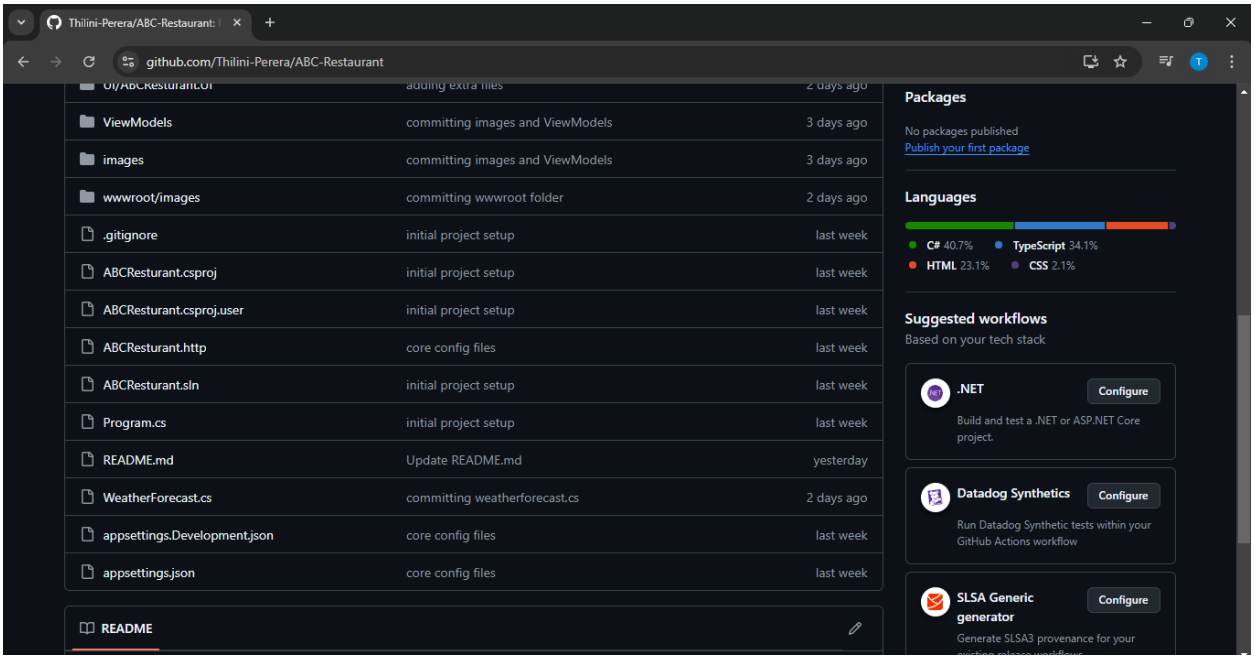

MINGW64:/c/Users/thili/OneDrive/Desktop/ABCResturant
commit 5a47b7eefdeb9114e1163c70d18ed3189bb9c25a
Author: Thilini Perera <thiliniruwanthi.perera@gmail.com>
Date: Sat Aug 31 16:20:47 2024 +0530
committing project documents folder
commit ae38d431f24e144d518b9f78e63d0535664e1669
Author: Thilini Perera <75115428+Thilini-Perera@users.noreply.github.com>
Date: Sat Aug 31 10:49:55 2024 +0530
Update README.md
commit 7f0912777ed82daf1d2325c5881ce61addaa6c38
Merge: b9e362d 29fb105
Author: Thilini Perera <thiliniruwanthi.perera@gmail.com>
Date: Fri Aug 30 16:47:43 2024 +0530
Merge branch 'main' of https://github.com/Thilini-Perera/ABC-Restaurant
commit b9e362deec291053a7f747162e53eaf807bd3be4
Author: Thilini Perera <thiliniruwanthi.perera@gmail.com>
Date: Fri Aug 30 16:46:06 2024 +0530
changing report name to query report
commit 29fb105ef30128dc5d820c8d623005feb4fb90e0
Author: Thilini Perera <75115428+Thilini-Perera@users.noreply.github.com>
Date: Fri Aug 30 16:21:50 2024 +0530
committing Restuarant.cs
commit f0f22926aba3dba6f51fbf966fd1dccf01991eeb
Author: Thilini Perera <thiliniruwanthi.perera@gmail.com>
Date: Fri Aug 30 15:58:09 2024 +0530
committing weatherforecast.cs
commit 9151d95e73d4e649f359dca5f2ed7254db3fea11
Author: Thilini Perera <thiliniruwanthi.perera@gmail.com>
Date: Fri Aug 30 15:56:06 2024 +0530

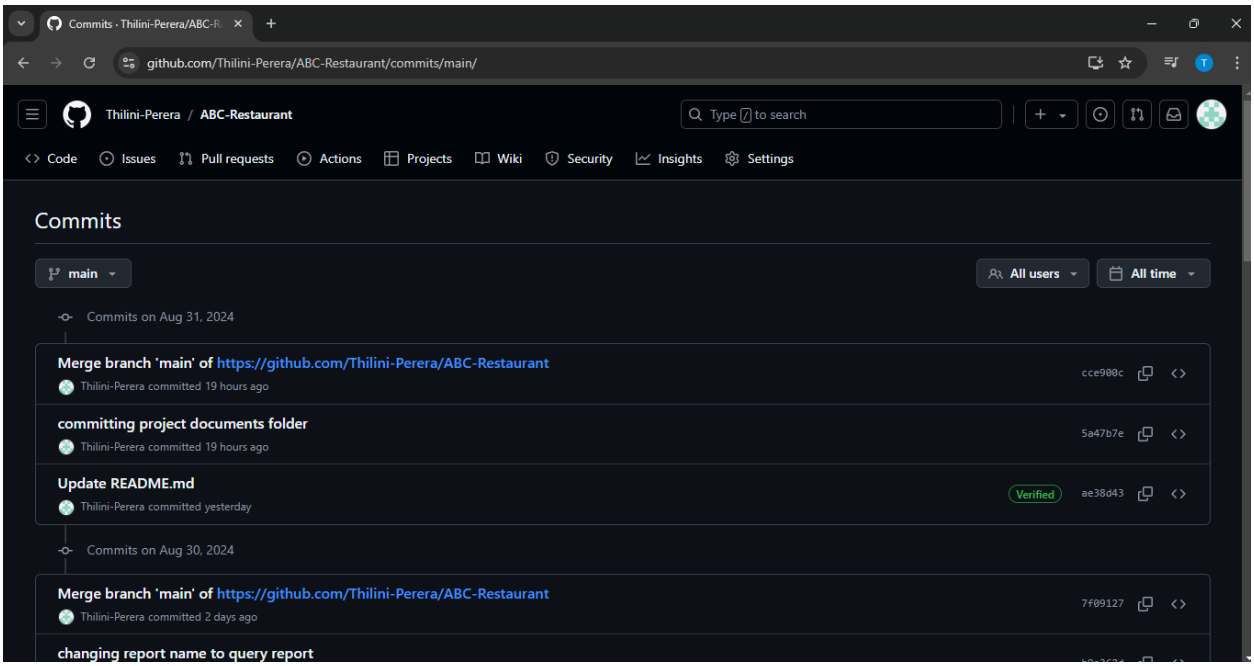

MINGW64:/c/Users/thili/OneDrive/Desktop/ABCResturant
Date: Fri Aug 30 15:56:06 2024 +0530
thili@Thilini_Perera MINGW64 ~/OneDrive/Desktop/ABCResturant (main)
$ git log --oneline
cce900c (HEAD -> main, origin/main) Merge branch 'main' of https://github.com/Thilini-Perera/ABC-Restaurant
5a47b7e committing project documents folder
ae38d43 Update README.md
7f09127 Merge branch 'main' of https://github.com/Thilini-Perera/ABC-Restaurant
b9e362d changing report name to query report
29fb105 committing Restuarant.cs
f0f2292 committing weatherforecast.cs
9151d95 committing wwwroot folder
1f5de9e adding extra files
cc2b22b Committing UI folder - commit 1
c020c83 committing images and ViewModels
5eb464d committing user activity, users, weather forecast controllers
a08ae26 committing controllers
a8d99ff committing persistence folder
2aa3639 committing migrations
831040f committing models
142e8cc committing payment.cs
8a5ded8 committing models - commit 1
09e9ddc core config files
c73fe03 Merge branch 'main' of https://github.com/Thilini-Perera/ABC-Restaurant
6e0b2fc Initial commit
4a22de2 initial project setup
thili@Thilini_Perera MINGW64 ~/OneDrive/Desktop/ABCResturant (main)
$ git branch
* main
thili@Thilini_Perera MINGW64 ~/OneDrive/Desktop/ABCResturant (main)
$ git branch -r
origin/main
thili@Thilini_Perera MINGW64 ~/OneDrive/Desktop/ABCResturant (main)
$ git remote -v
origin https://github.com/Thilini-Perera/ABC-Restaurant.git (fetch)
origin https://github.com/Thilini-Perera/ABC-Restaurant.git (push)
thili@Thilini_Perera MINGW64 ~/OneDrive/Desktop/ABCResturant (main)
$ git config --list
diff.astextplain.textconv=astextplain
filter.lfs.clean=git-lfs clean -- %f
filter.lfs.smudge=git-lfs smudge -- %f
filter.lfs.process=git-lfs filter-process
filter.lfs.required=true
http.sslbackend=openssl

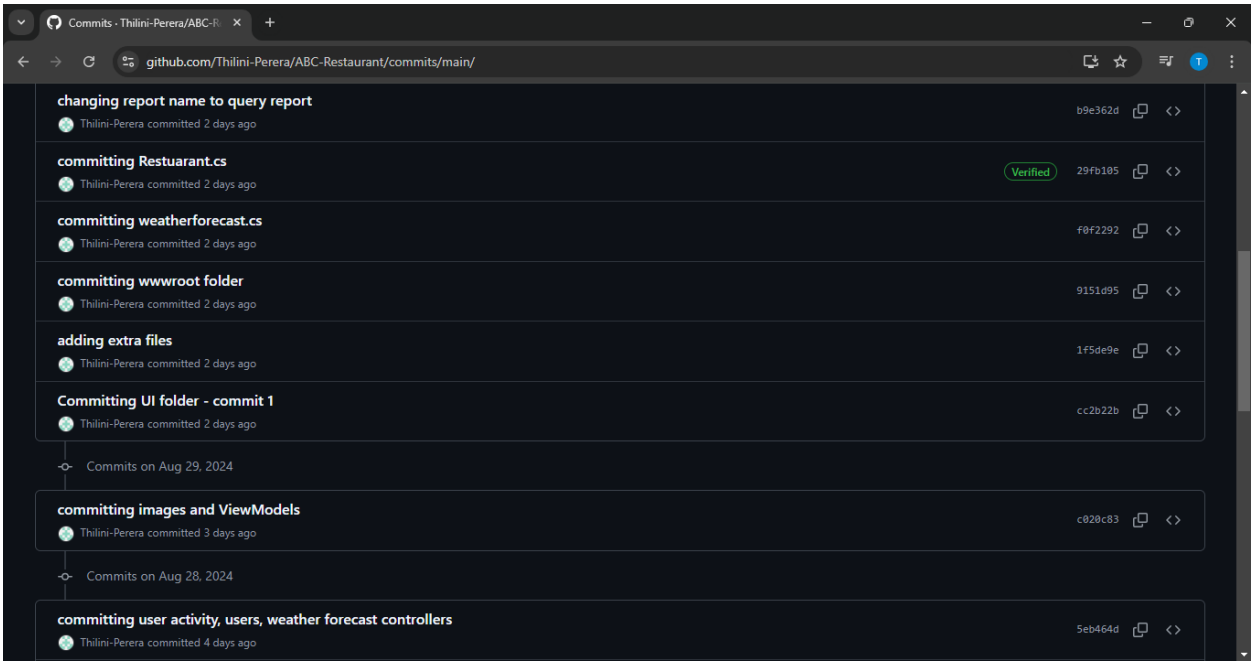

MINGW64:/c/Users/thili/OneDrive/Desktop/ABCResturant
origin https://github.com/Thilini-Perera/ABC-Restaurant.git (push)
thili@Thilini_Perera MINGW64 ~/OneDrive/Desktop/ABCResturant (main)
$ git config --list
diff.astextplain.textconv=astextplain
filter.lfs.clean=git-lfs clean -- %f
filter.lfs.smudge=git-lfs smudge -- %f
filter.lfs.process=git-lfs filter-process
filter.lfs.required=true
http.sslbackend=openssl
http.sslcainfo=C:/Program Files/Git/mingw64/etc/ssl/certs/ca-bundle.crt
core.autocrlf=true
core.fscache=true
core.symlinks=false
pull.rebase=false
credential.helper=manager
credential.https://dev.azure.com.usehttppath=true
init.defaultbranch=master
user.name=Thilini Perera
user.email=thiliniruwanthi.perera@gmail.com
core.editor="C:\Users\thili\AppData\Local\Programs\Microsoft VS Code\bin\code" --wait
core.autocrlf=true
init.default=branch
filter.lfs.clean=git-lfs clean -- %f
filter.lfs.smudge=git-lfs smudge -- %f
filter.lfs.process=git-lfs filter-process
filter.lfs.required=true
core.repositoryformatversion=0
core.filemode=false
core.bare=false
core.logallrefupdates=true
core.symlinks=false
core.ignorecase=true
remote.origin.url=https://github.com/Thilini-Perera/ABC-Restaurant.git
remote.origin.fetch=+refs/heads/*:refs/remotes/origin/*
branch.main.remote=origin
branch.main.merge=refs/heads/main
thili@Thilini_Perera MINGW64 ~/OneDrive/Desktop/ABCResturant (main)
$ git remote show origin
* remote origin
  Fetch URL: https://github.com/Thilini-Perera/ABC-Restaurant.git
  Push URL: https://github.com/Thilini-Perera/ABC-Restaurant.git
  HEAD branch: main
  Remote branch:
    main tracked
  Local branch configured for 'git pull':
    main merges with remote main
  Local ref configured for 'git push':

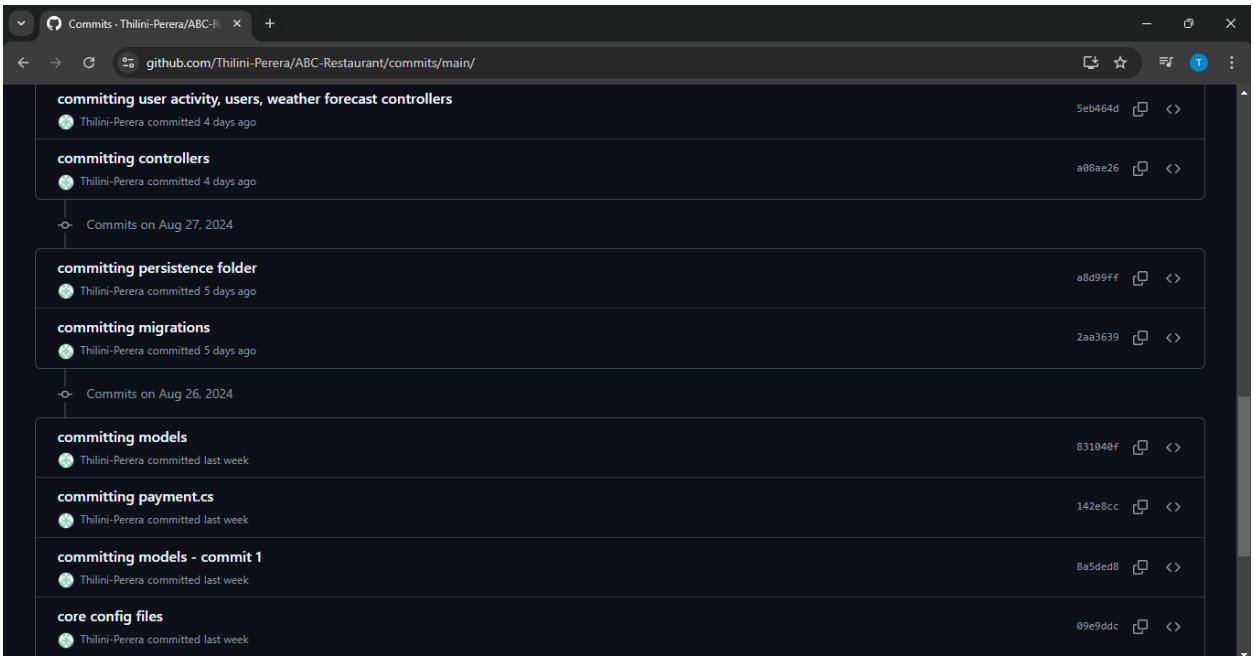
GitHub repository

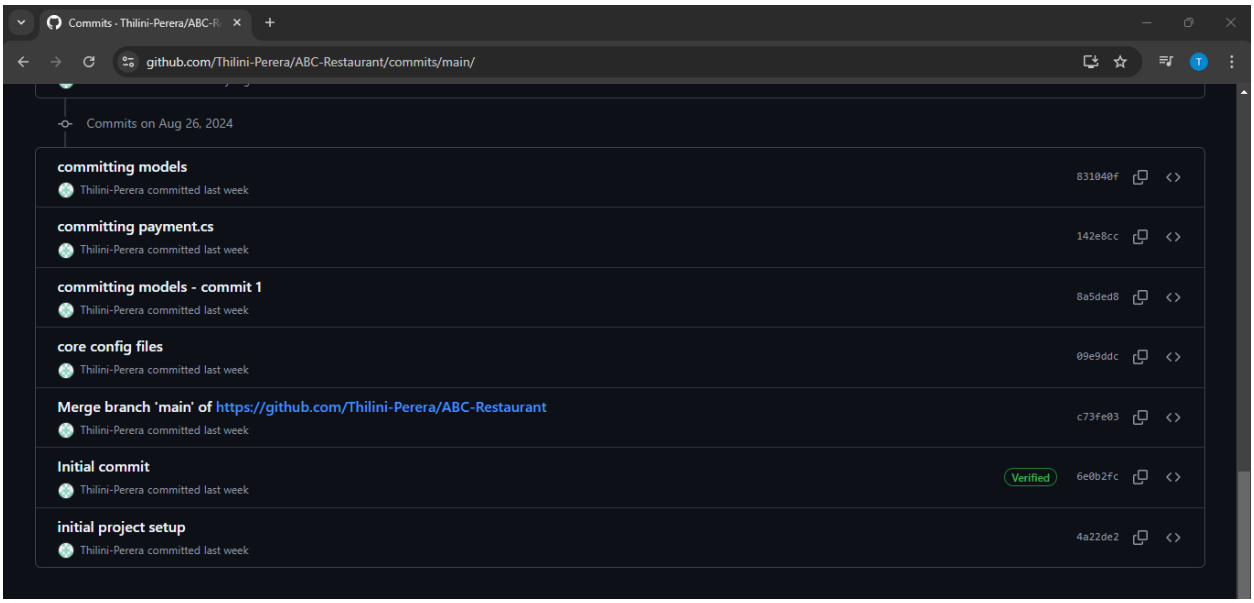





Commits · Thilini-Perera/ABC-R
github.com/Thilini-Perera/ABC-Restaurant/commits/main/
Thilini-Perera / ABC-Restaurant
Type / to search
Code
Issues
Pull requests
Actions
Projects
Wiki
Security
Insights
Settings
Commits
main
All users
All time
Commits on Aug 31, 2024
Merge branch 'main' of https://github.com/Thilini-Perera/ABC-Restaurant
Thilini-Perera committed 19 hours ago
cce900c
committing project documents folder
Thilini-Perera committed 19 hours ago
5a47b7e
Update README.md
Thilini-Perera committed yesterday
Verified
ae38d43
Commits on Aug 30, 2024
Merge branch 'main' of https://github.com/Thilini-Perera/ABC-Restaurant
Thilini-Perera committed 2 days ago
7f09127
changing report name to query report


Commits · Thilini-Perera/ABC-R
github.com/Thilini-Perera/ABC-Restaurant/commits/main/
changing report name to query report
Thilini-Perera committed 2 days ago
b9e362d
committing Restuarant.cs
Thilini-Perera committed 2 days ago
Verified
29fb105
committing weatherforecast.cs
Thilini-Perera committed 2 days ago
f0f2292
committing wwwroot folder
Thilini-Perera committed 2 days ago
9151d95
adding extra files
Thilini-Perera committed 2 days ago
1f5de9e
Committing UI folder - commit 1
Thilini-Perera committed 2 days ago
cc2b22b
Commits on Aug 29, 2024
committing images and ViewModels
Thilini-Perera committed 3 days ago
c020c83
Commits on Aug 28, 2024
committing user activity, users, weather forecast controllers
Thilini-Perera committed 4 days ago
5eb464d


Commits · Thilini-Perera/ABC-R
github.com/Thilini-Perera/ABC-Restaurant/commits/main/
committing user activity, users, weather forecast controllers
Thilini-Perera committed 4 days ago
5eb464d
committing controllers
Thilini-Perera committed 4 days ago
a08ae26
Commits on Aug 27, 2024
committing persistence folder
Thilini-Perera committed 5 days ago
a8d99ff
committing migrations
Thilini-Perera committed 5 days ago
2aa3639
Commits on Aug 26, 2024
committing models
Thilini-Perera committed last week
831040f
committing payment.cs
Thilini-Perera committed last week
142e8cc
committing models - commit 1
Thilini-Perera committed last week
8a5ded8
core config files
Thilini-Perera committed last week
09e9ddc


Commits · Thilini-Perera/ABC-R
github.com/Thilini-Perera/ABC-Restaurant/commits/main/
Commits on Aug 26, 2024
committing models
Thilini-Perera committed last week
831040f
committing payment.cs
Thilini-Perera committed last week
142e8cc
committing models - commit 1
Thilini-Perera committed last week
8a5ded8
core config files
Thilini-Perera committed last week
09e9ddc
Merge branch 'main' of https://github.com/Thilini-Perera/ABC-Restaurant
Thilini-Perera committed last week
c73fe03
Initial commit
Thilini-Perera committed last week
Verified
6e0b2fc
initial project setup
Thilini-Perera committed last week
4a22de2

GitHub repository for test automation