Global Distributed Software Development

Master Team Project WS 2022/23

“ReservEat”

Milestone 2 - Project Team 01 - Hochschule Fulda

December 07

Utsav Shrestha
Team Lead, Frontend Lead
utsav.shrestha@informatik.hs-fulda.de

Sanjay George
Backend lead, DevOps
sanjay.varkey-george@informatik.hs-fulda.de

Omer Zogubi
GitHub master, Backend Member
omer.zogubi@informatik.hs-fulda.de

Muhammad Bilal
Frontend member
muhammad.bilal@informatik.hs-fulda.de

Parthiv Jani
Frontend Member, Document master
parthiv-yogesh-kumar.jani@informatik.hs-fulda.de

Kristijan Lazeski
Database admin, Backend member
kristijan.lazeski@informatik.hs-fulda.de

History Table (Revisions)

Date Submitted	Date Revised	Revision Summary

1 Functional Requirements

Henceforth the Table of all of the Functional Requirements listed with a Priority:

1.1 Priority 1

ID	Functional Requirement	Priority
1	The Application is a Vendor for Multiple Restaurants	1
2	User can Search for Restaurants by name, cuisine, food, and rating	1
2.1	Search results can be filtered additionally and sorted	1
3	User can reserve a table and specify the number of seats in a given Restaurant	1
3.1	User reserves one or Multiple Tables at the Restaurant	1
3.2	User specifies the number of seats	1
4	Restaurants can be added, removed and updated by Restaurant owners	1
4.1	Restaurants can be Added by the Restaurant Owners	1
4.2	Restaurants can be Removed by the Owners of given Restaurant	1
4.3	Restaurants can be Updated by the owners of said Restaurant	1
5	There is a Grace Period to Cancel or update the booking of the table	1
5.1	Every User can Cancel the reservation up to 2h before said reservation	1
6	The number of booked seats are specified per table	1
7	Users can view Menus of the Restaurants	1
8	Users search for restaurants, make/change/cancel reservations and also post reviews.	1
8.1	User can cancel Reservation	1
8.2	User can change Reservation	1
8.3	User can Post Reviews	1
9	User can View the Restaurant Reviews	1
10	Every booking is associated with an account	1
11	One Account can not be associated with multiple users	1
12	User can chat with the Restaurant Host	1
13	User accumulates Reward Points for every reservations with the application	1

1.2 Priority 2

ID	Functional Requirement	Priority
1	Restaurants can add Extra Services with themes through Application	2
1.1	Restaurants can add Themes as a Service	2
1.2	Restaurants can add Special Foods or Beverages as a Service to be clicked/booked beforehand	2
2	Restaurants can change the availability status of any table at any time.	2
2.1	Restaurant owner updates the number of free seats for given table	2
3	Restaurant Owners can deny or approve of a booking	2
3.1	Restaurants owner can disapprove a booking made for given number of Seats at a given Time	2
3.2	Restaurants owner can approve a booking made for given number of Seats at a given Time	2
4	User can choose Event Themes	2
5	User can only book two tables daily (to avoid spam reservations)	2
6	Site administrator approves restaurant info for posting. Also deals with typical admin duties like managing user registrations etc.	2
6.1	Site Admin approves Restaurant info for Posting	2
6.2	Site Admin Identifies and addresses usability Problems	2
6.3	Site Admin manages user Registration	2
6.4	Site Admin tracks and analyses the Traffic that is being made on our Website	2
6.5	Site Admin Develops Training Manual for use for Restaurants	2
6.6	Site Admin Flags user if there is a valid reasoning	2
7	User can Check his Reservations	2

1.3 Priority 3

ID	Functional Requirement	Priority
1	Users can Provide Reviews for the Restaurant which they have visited through our Application	3
1.1	User provides Text of the Review	3

1.2	User gives a Number Rating between 1 and 5	3
2	Restaurant owners can Flag users for their lack of attendance on booking	3
2.1	Restaurant owners submit a report for time and place and reasoning of why said user should be flagged	3
3	Admin can Ban Restaurants	3
4	Host/hostess reviews daily calendar and checks and greets incoming guests This List might be Edited as the Project goes on.	3

2 List of main data items and entities

The main data items and entities used in this project are:

2.1 Registration

A user needs to be registered and logged in to be able to reserve a restaurant. Similarly a restaurant admin should register themselves before they are able to list their restaurants in the system. The restaurant admin then can register their host/hostess in the system.

2.2 Restaurant

Restaurants are the main entity of our system. All the data about the restaurants are added and shown in the website so that a user can choose the restaurant according to their liking and reserve them.

2.3 Roles

The system has 4 roles that an actor can be assigned to. Each role has its own responsibility. The roles are listed below:

- **Users:** Searches for a restaurant and books it.
- **Restaurant Administrator:** Upload the information about the restaurant and manage the data.
- **Site Administrator:** Approves or rejects the changes made by the restaurant admins and Restaurant hosts.
- **Restaurant Host:** Reviews the reservations made by the users and manages the user activities.

2.4 Reviews

A user can post a review about a restaurant in the restaurant details page. They can rate by giving a number of stars and adding a review of how they felt about the restaurant. The review needs to be approved by system admin. The restaurant can flag the user reviews if they deem it inappropriate.

2.5 Chat

A user can chat about the more enquiries they have regarding the restaurants with the restaurant host. The chat will be available in each restaurant's details page.

2.6 Flag

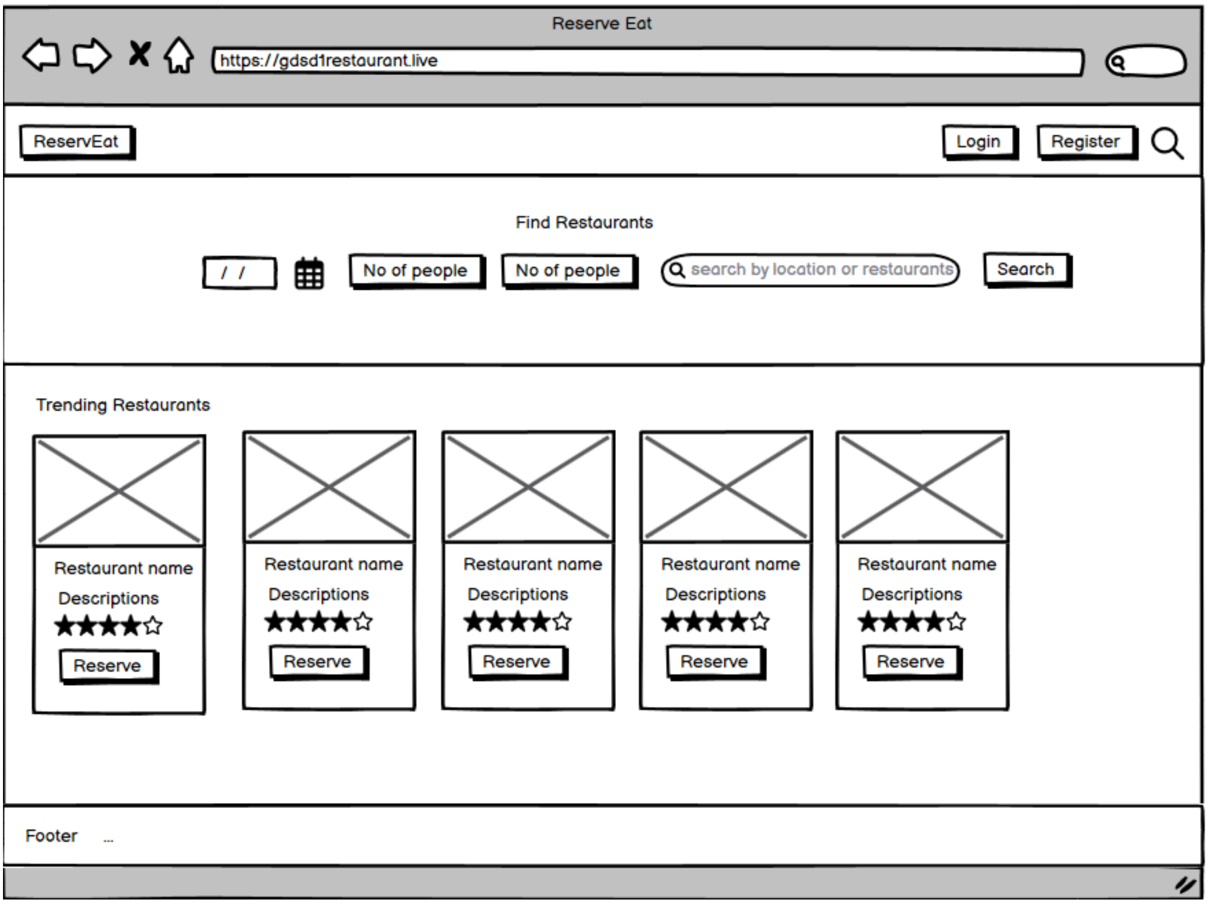
A restaurant admin or host/hostess can flag a user or their reviews. The flagged user's reservation needs to be then approved by the restaurant management to finalise it.

2.7 Rewards

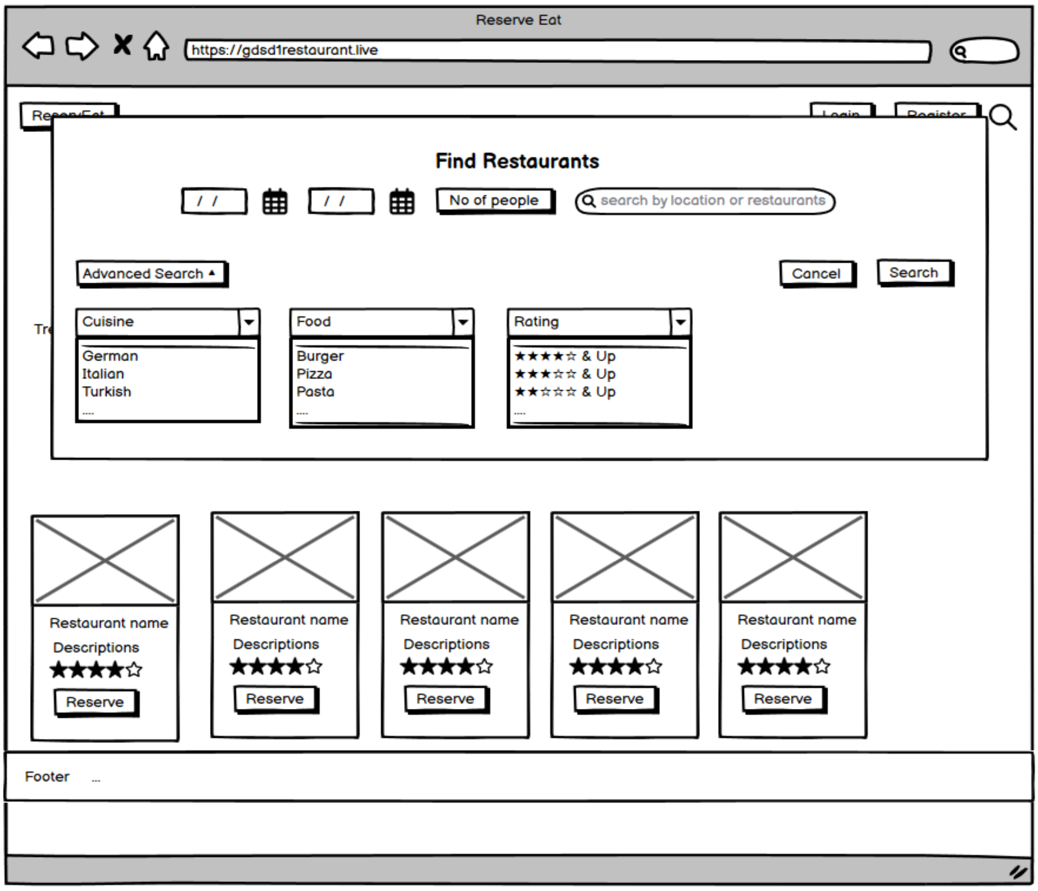
A user will be awarded with reward points every time he reserves a restaurant. The user can collect the reward points and get some free gifts with certain points. If the user does not dine in the restaurant after booking, the restaurant owner/host can flag the user and his reward point will be deducted. The reward points can be gained by reserving the restaurant and giving reviews for the restaurants that he has attended. The reward points will be matched with the money in a certain ratio (e.g. 1:1000 - 1 being money and 1000 being the reward points) and can be used to dine in any restaurant equivalent to that money.

3 New UI Mockups

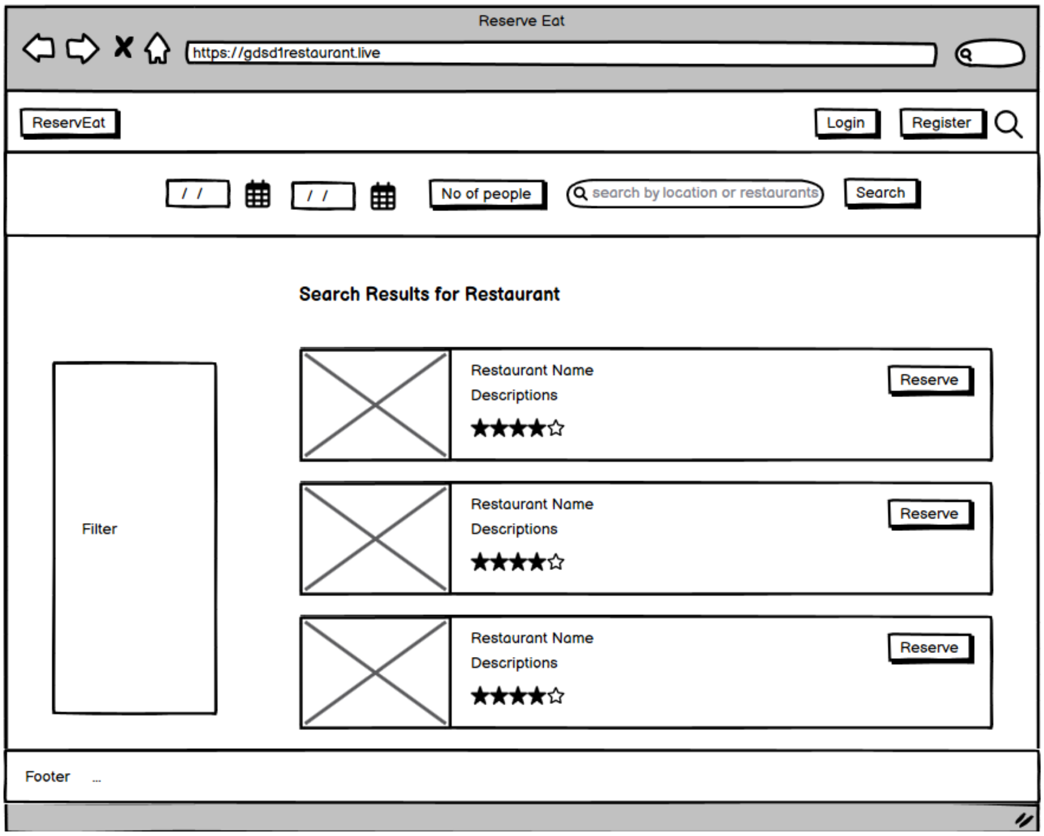
1- Home Page



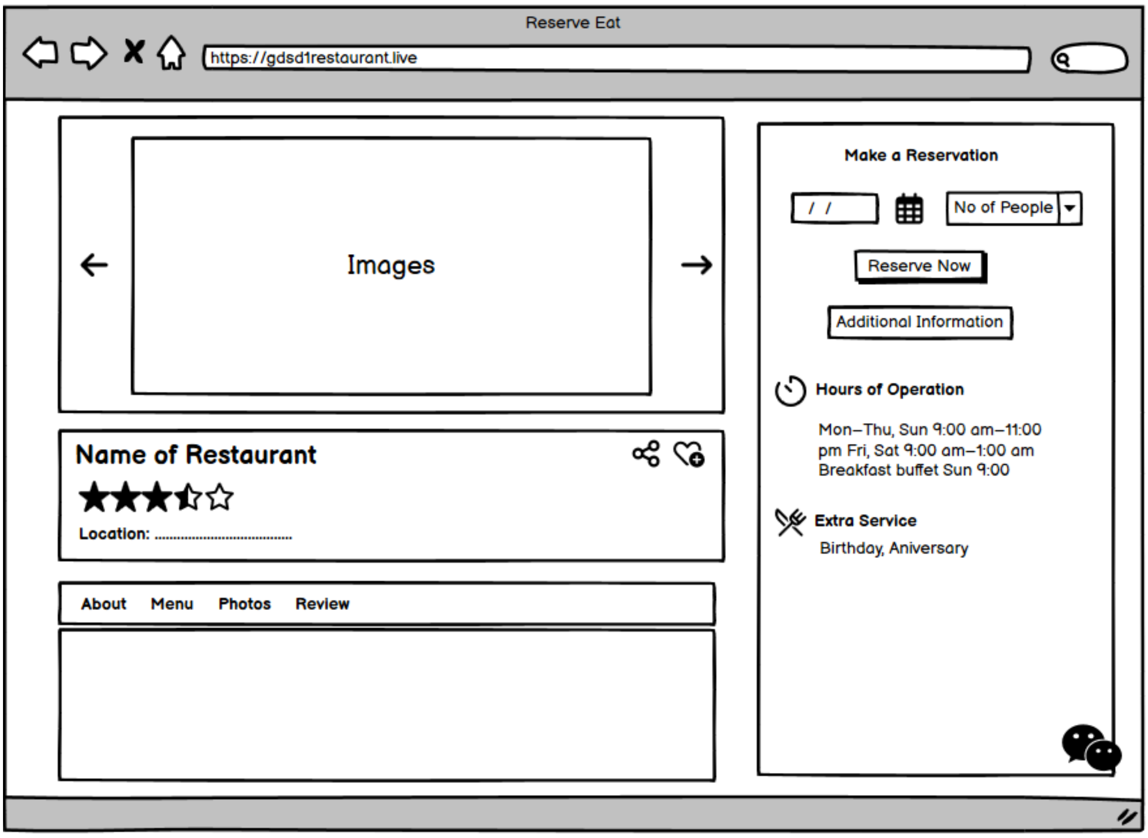
2-Search Restaurant Page



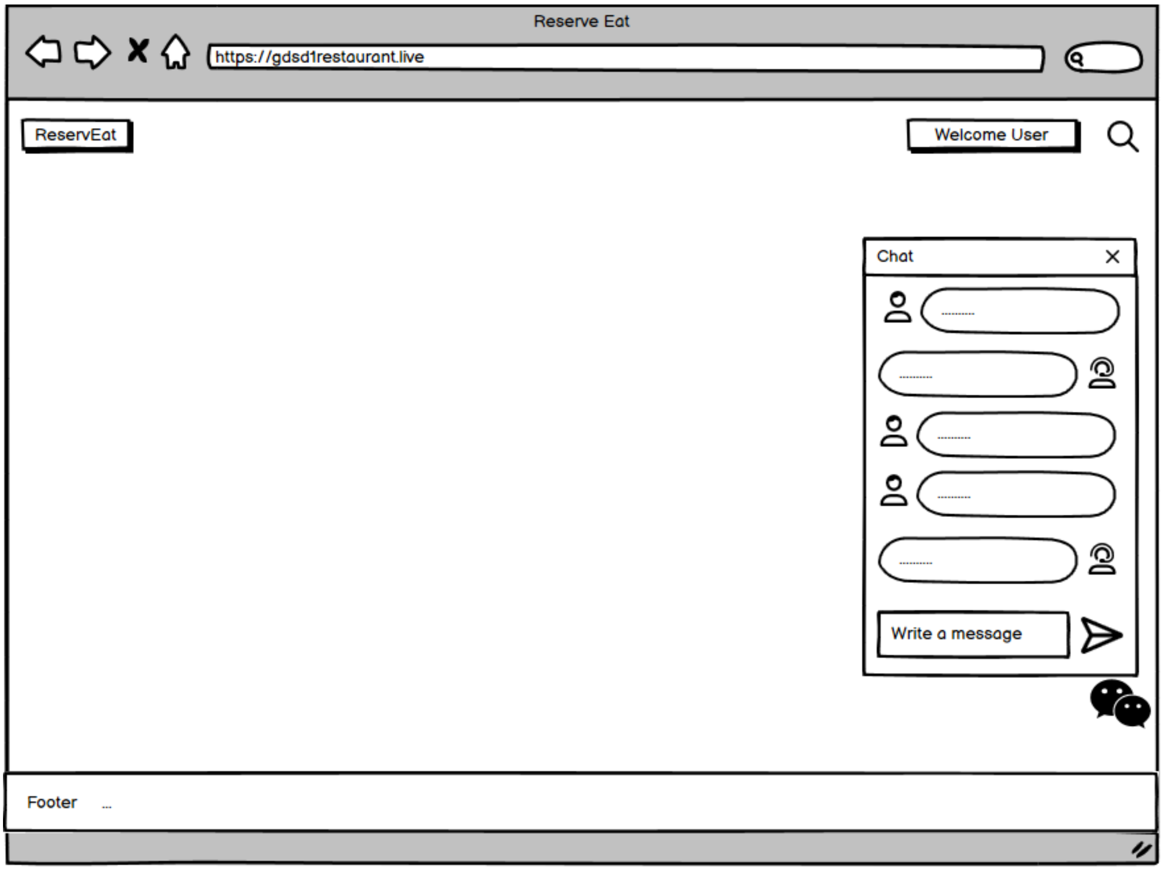
3-Restaurants List Page



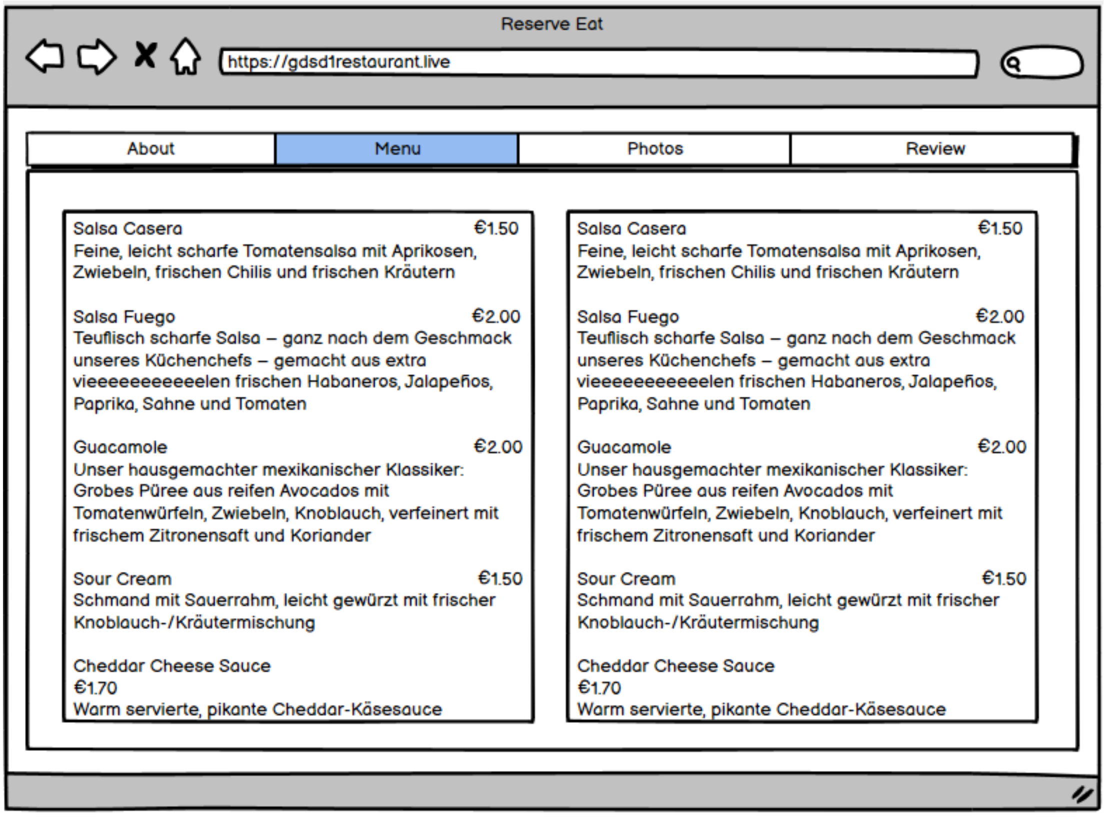
4-Restaurant Details Page



5- Chatting Page



6-Menu Tab



Salsa Casera €1.50
Feine, leicht scharfe Tomatensalsa mit Aprikosen, Zwiebeln, frischen Chilis und frischen Kräutern

Salsa Fuego €2.00
Teuflisch scharfe Salsa – ganz nach dem Geschmack unseres Küchenchefs – gemacht aus extra vieeeeeeeeeeelen frischen Habaneros, Jalapeños, Paprika, Sahne und Tomaten

Guacamole €2.00
Unser hausgemachter mexikanischer Klassiker: Grobes Püree aus reifen Avocados mit Tomatenwürfeln, Zwiebeln, Knoblauch, verfeinert mit frischem Zitronensaft und Koriander

Sour Cream €1.50
Schmand mit Sauerrahm, leicht gewürzt mit frischer Knoblauch-/Kräutermischung

Cheddar Cheese Sauce
€1.70
Warm servierte, pikante Cheddar-Käsesauce

Salsa Casera €1.50
Feine, leicht scharfe Tomatensalsa mit Aprikosen, Zwiebeln, frischen Chilis und frischen Kräutern

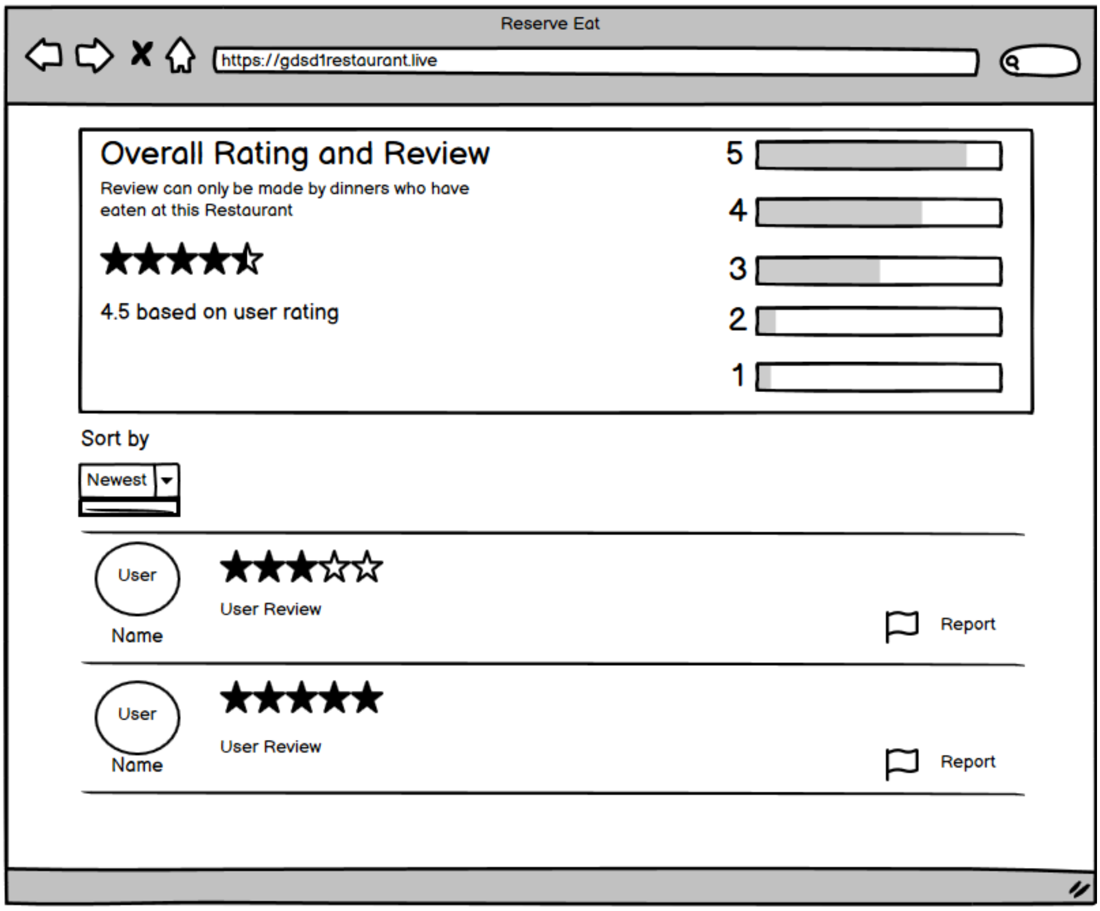
Salsa Fuego €2.00
Teuflisch scharfe Salsa – ganz nach dem Geschmack unseres Küchenchefs – gemacht aus extra vieeeeeeeeeeelen frischen Habaneros, Jalapeños, Paprika, Sahne und Tomaten

Guacamole €2.00
Unser hausgemachter mexikanischer Klassiker: Grobes Püree aus reifen Avocados mit Tomatenwürfeln, Zwiebeln, Knoblauch, verfeinert mit frischem Zitronensaft und Koriander

Sour Cream €1.50
Schmand mit Sauerrahm, leicht gewürzt mit frischer Knoblauch-/Kräutermischung

Cheddar Cheese Sauce
€1.70
Warm servierte, pikante Cheddar-Käsesauce

7-Review Tab



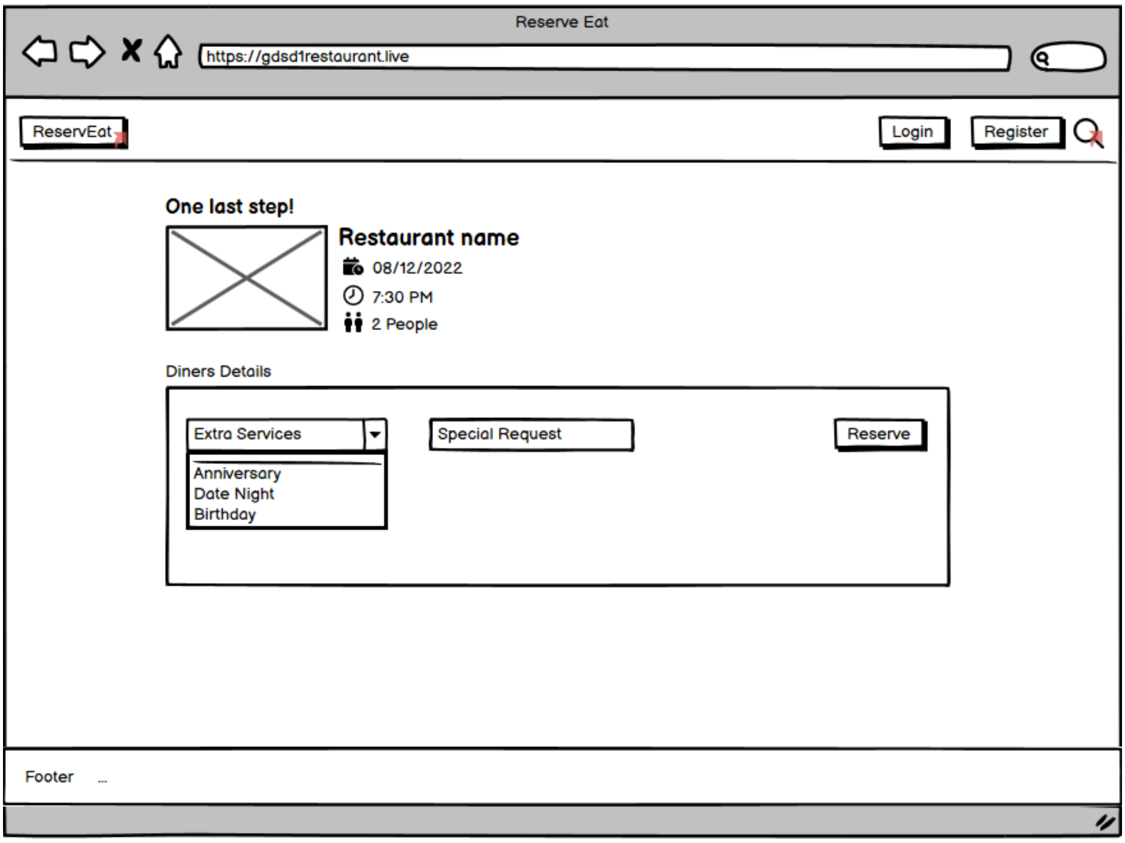
Overall Rating and Review

Review can only be made by dinners who have eaten at this Restaurant

★★★★☆

4.5 based on user rating

8-Restaurant Reservation Page



One last step!

Restaurant name

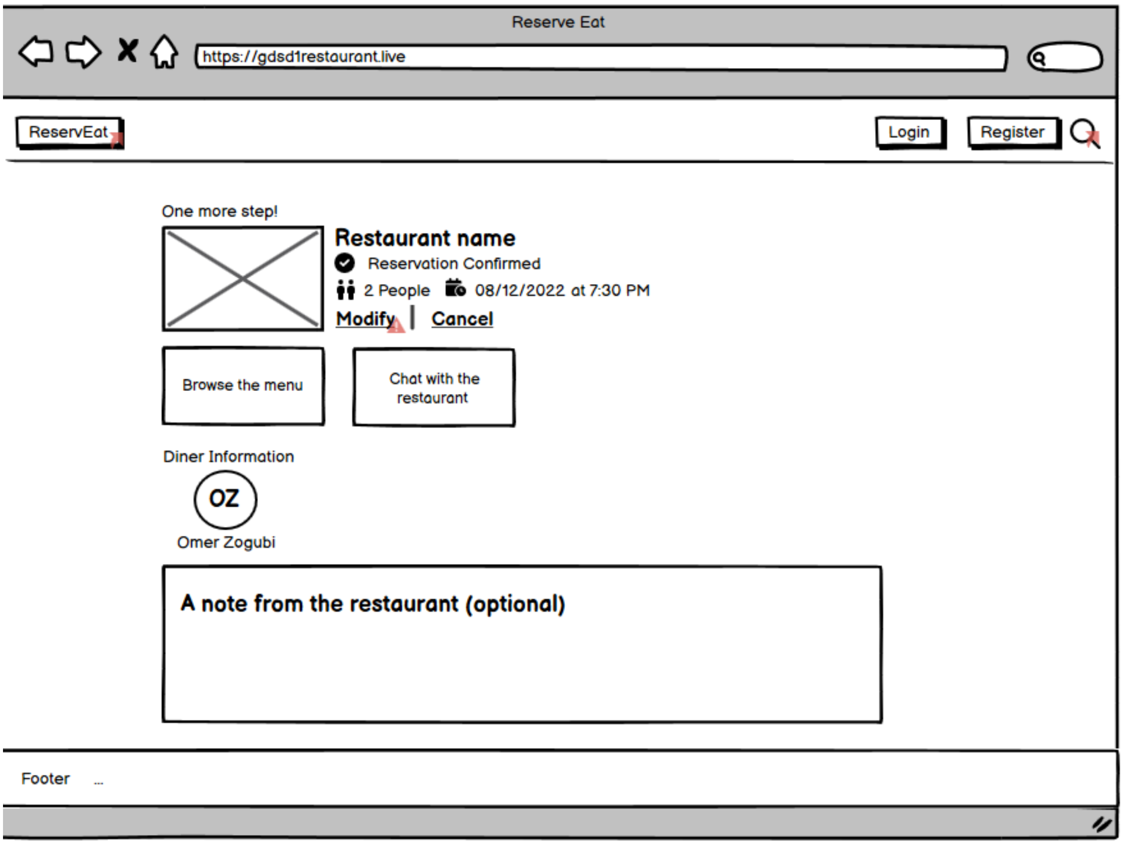
08/12/2022

7:30 PM

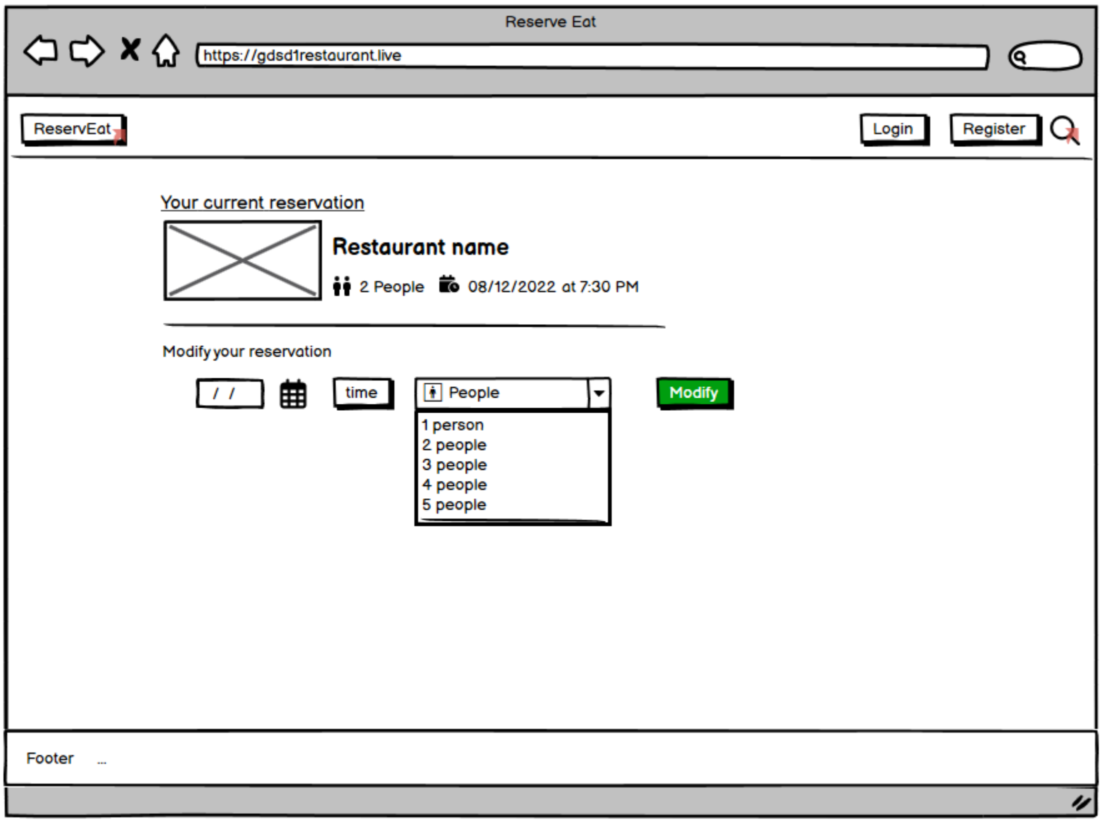
2 People

Diners Details

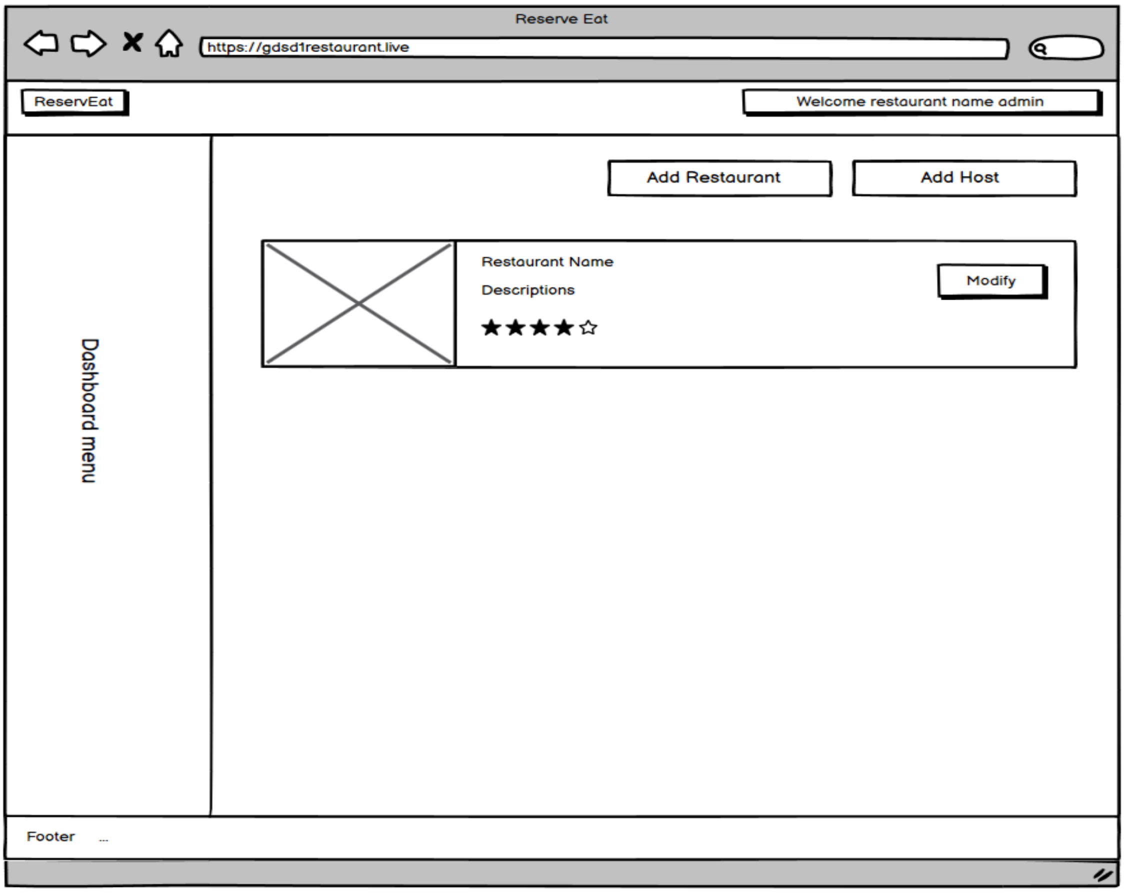
9- Reservation Confirmed/Details Page



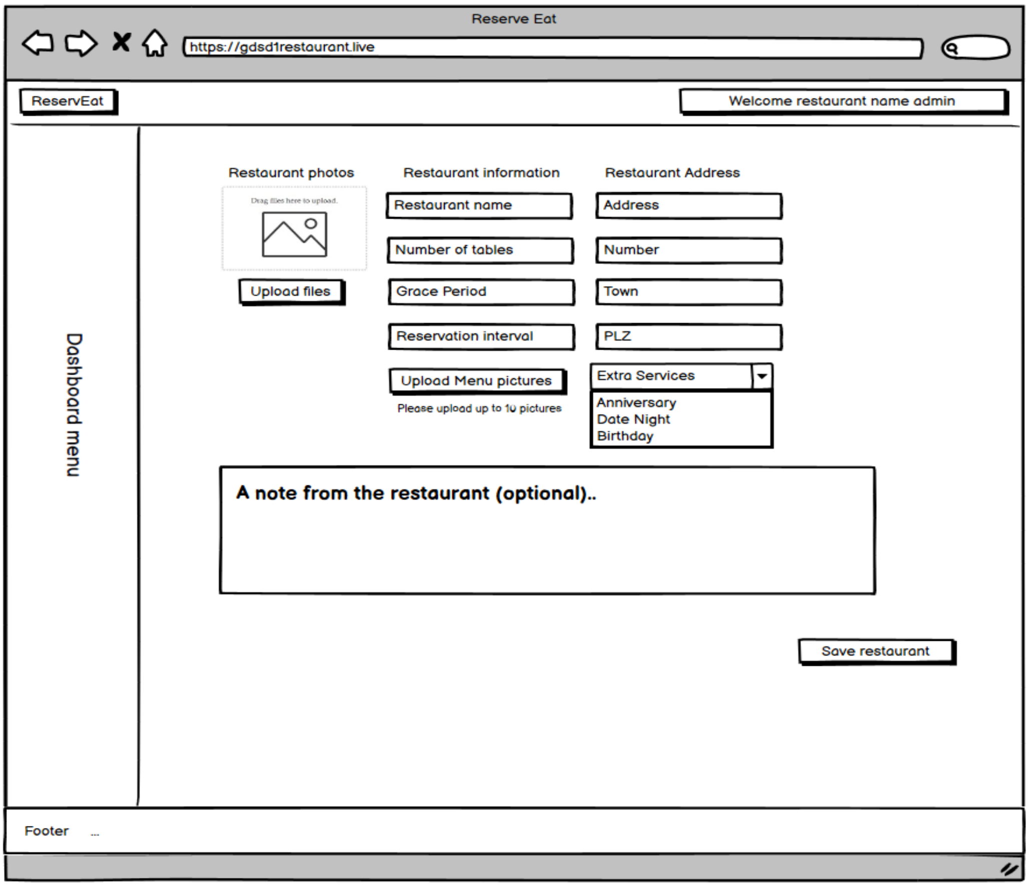
10- Modify Reservation Page



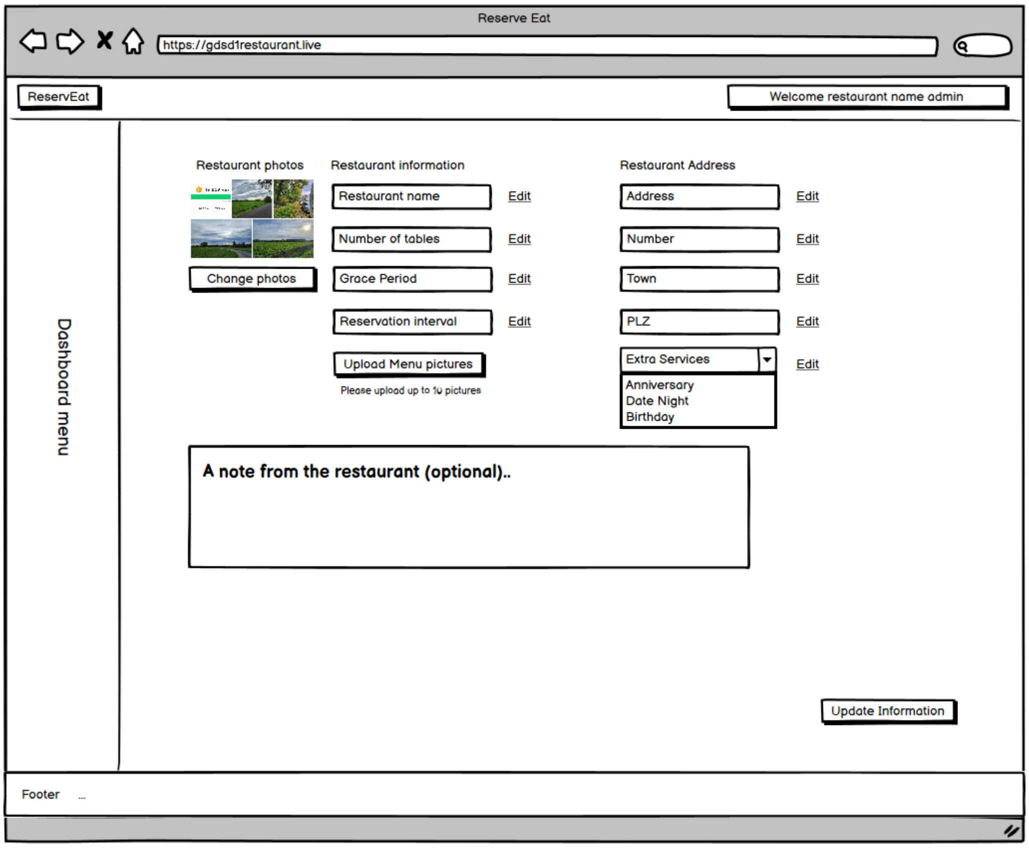
11- Restaurant Host Dashboard Page



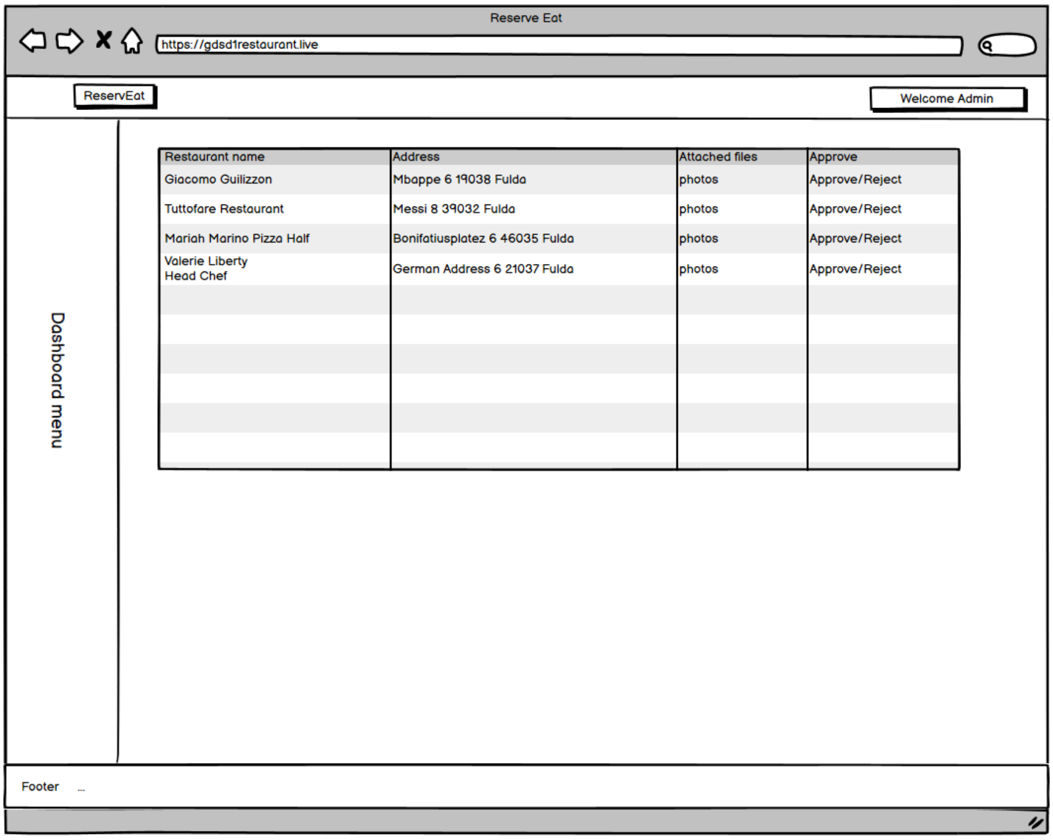
12- Restaurant host add new restaurant page



12- Restaurant host modify existing restaurant page

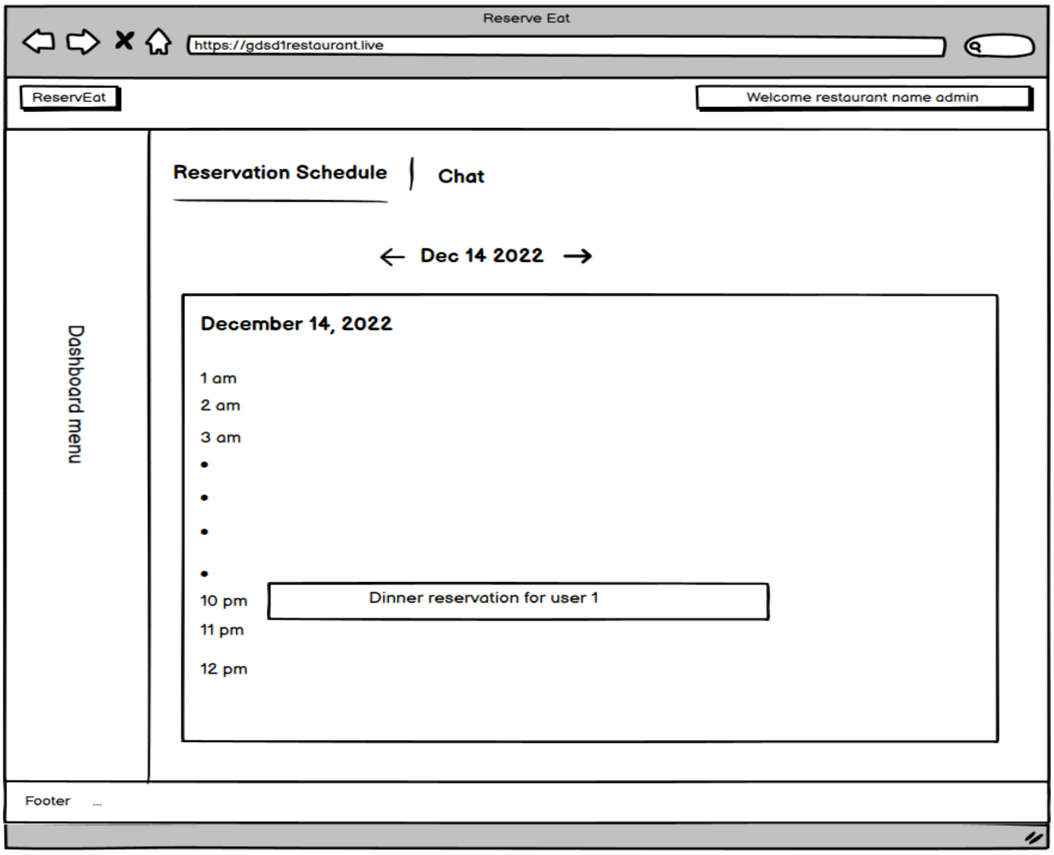


13- Site Administrator Restaurants approval page

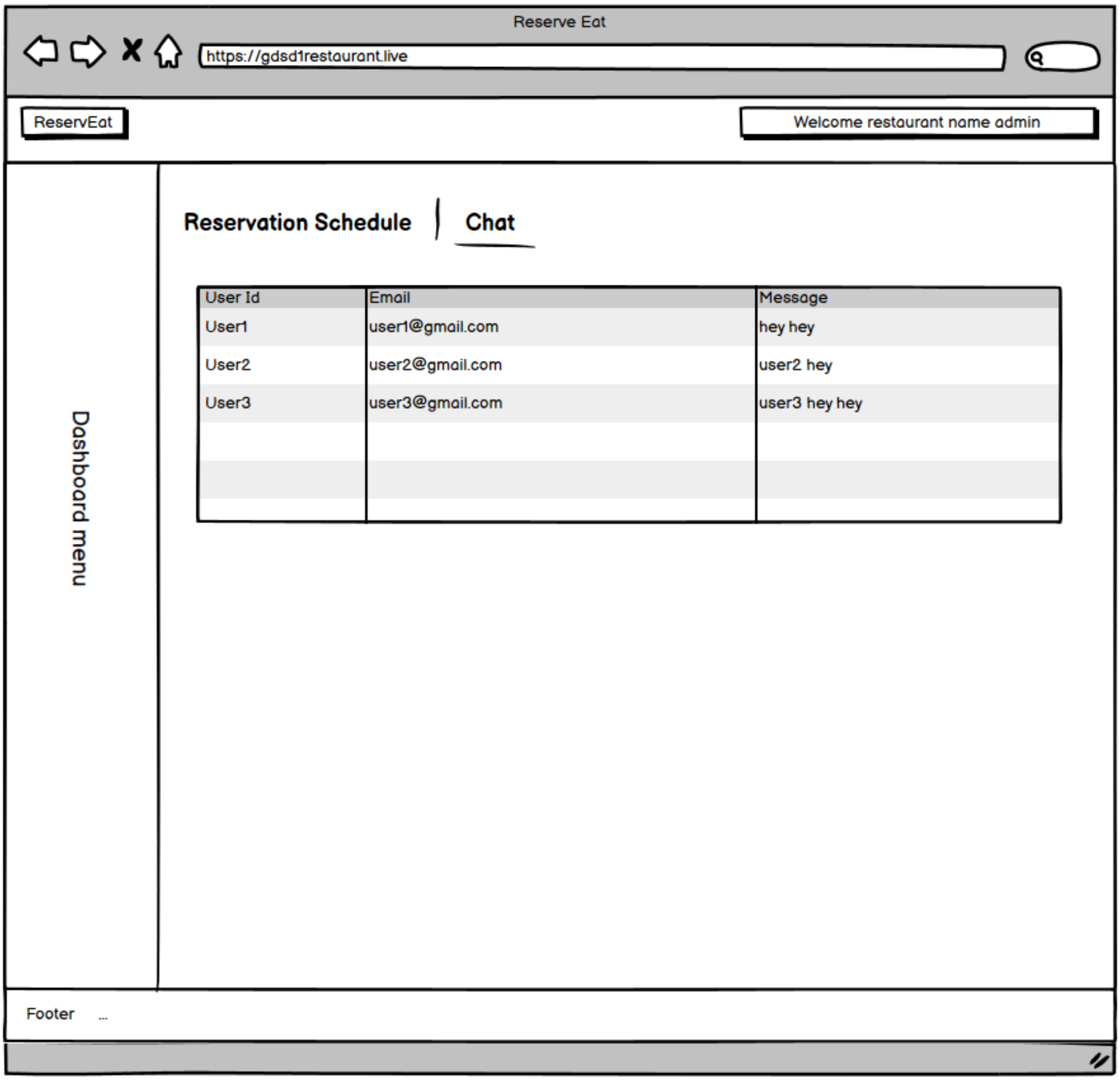


Restaurant name	Address	Attached files	Approve
Giacomo Guilizzon	Mbappe 6 19038 Fulda	photos	Approve/Reject
Tuttofare Restaurant	Messi 8 39032 Fulda	photos	Approve/Reject
Mariah Marino Pizza Half	Bonifatiusplatez 6 46035 Fulda	photos	Approve/Reject
Valerie Liberty Head Chef	German Address 6 21037 Fulda	photos	Approve/Reject

14-Restaurant Host Page



15- Restaurant host messages/chat page



User Id	Email	Message
User1	user1@gmail.com	hey hey
User2	user2@gmail.com	user2 hey
User3	user3@gmail.com	user3 hey hey

4 High Level Architecture, Database

This section discusses the architecture and database design with appropriate diagrams.

4.1 High Level Architecture

For the Project it was decided upon thorough discussion between our team members, that the Project will be hosted in the Microsoft Azure cloud. The Back-end Framework we decided upon was Express JS and using JavaScript (Node JS) as the complementary Back-end Language. The Front-end Framework was set to be React JS, with JavaScript as a Language accordingly.

4.2 Database

The choice of the Database is MySQL. It is a widely used Database that gives opportunities for others solving common problems that we might come into contact with. Therefore is this a perfect fit for this given project.

In addition here are the Tables that are going to be used for the Project and the website:

Table: User	
Column	**Data Type**
Id	int
FirstName	varchar
LastName	varchar
Email	varchar
Role	varchar
Password	varchar
PhoneNumber	varchar

Table: Restaurants	
Column	**Data Type**
Id	int
Name	varchar
PostalCode	tinyint
Address	varchar

City	varchar
MaxCapacity	int
MaxTables	int
RestaurantNote	varchar
GracePeriod	timestamp
TimeInterval	timestamp

Table: Reservation	
Column	**Data Type**
Id	int
UserId	varchar
RestaurantId	int
NumberOfSeats	int
Status	enum(“Reserved”, “Modified”, “Cancelled”)
Date	date
Time	timestamp
ExtraServiceId	int
LastUpdated	timestamp

Table: ExtraServices	
Column	**Data Type**
Id	int
Name	varchar
Description	varchar

Table: Chats	
Column	**Data Type**

Id	int
User1	int
User2	int

Table: Messages	
Column	**Data Type**
ChatId	int
Message	varchar
Timestamp	datetime

Table: Flags	
Column	**Data Type**
Id	int
UserId	int
ReservationId	int
Text	varchar

Table: Rewards	
Column	**Data Type**
Id	int
UserId	int
Type	enum(“ForReservation”, “ForUserReview”)
Points	int

Table: Reviews	
Column	**Data Type**

Id	int
UserId	int
RestaurantId	int
Rating	int
Text	varchar

Table: Images	
Column	**Data Type**
Id	int
RestaurantId	int
Status	enum(“Uploaded”, “Approved”, “Rejected”)
Path	varchar
IsMenuImage	boolean

4.3 Media Storage

Image uploaded by the restaurant admin constitutes the media of our application. These images will be stored on the server file system and the path will be stored in MySQL.

4.4 Search Filter Algorithm

We will utilise the MySQL Full Text Search Functions to power user’s search in addition to the LIKE operator. Additional filters and sorting will be applied in advanced searching using WHERE and ORDER BY clauses, it can be filtered by (Cuisine, Food, Rating).

4.5 Important Algorithms

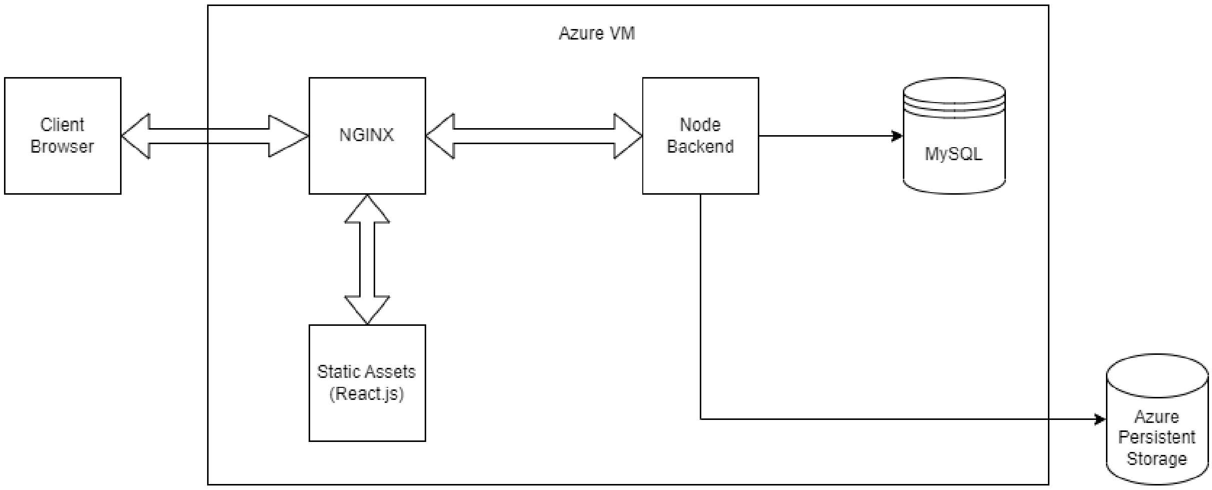
- **Ranking Trending Restaurants:** Restaurants will be ranked based on the relative percentage of bookings they get in a week. The reason for considering the relative percentage over the absolute count is because a restaurant with more tables could potentially have more bookings than a restaurant with fewer tables that is consistently booked out through the week.
- **Reward Points:** We will assign reward points for every “good” action taken by the user on our website, such as making a reservation for a restaurant without cancelling it, leaving a review on the restaurant, etc. We will deduct reward points for “bad” actions, such as not showing up for the reservation, multiple simultaneous cancellations of reservations, etc.

4.6 APIs

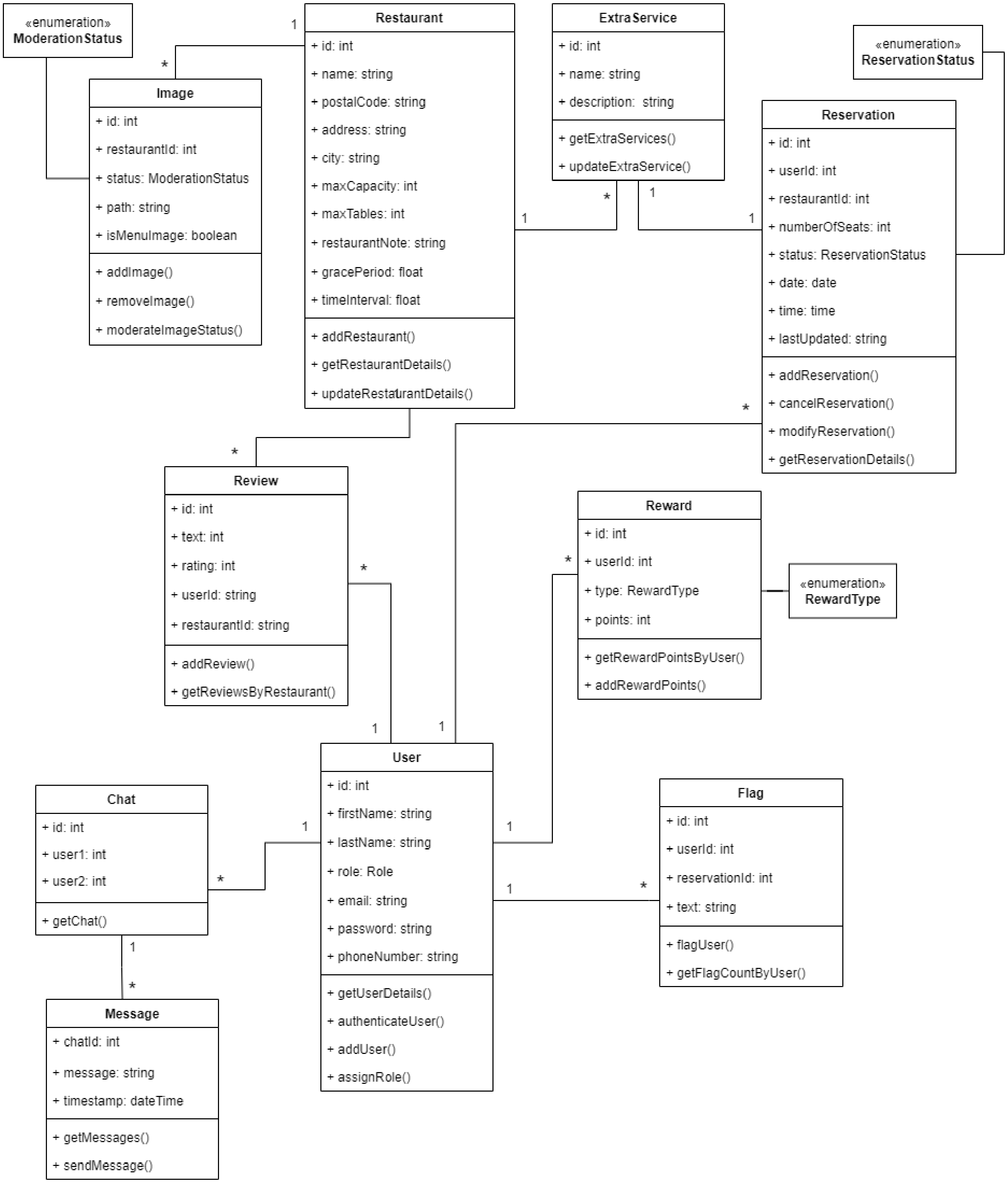
Other than the API endpoints for CRUD operations, there are no particular APIs required to be built. We will use the programming interfaces of the tools and frameworks we have chosen.

5. NEW High Level UML Diagrams

5.1 Deployment diagram



5.2 High Level UML Class Diagram



6. Identify actual key risks for your project at this time

- **Skills risks**
  Our team is composed of people with every skill set needed. But some only know one stack or need to learn both.
  Our plan of action:
  We have divided our team with one experienced member with other less experienced members in that stack.
- **Schedule risks**
  We have a tight schedule this time as we have much to cover and have less time for vacation.
  Our plan of action:
  We plan to increase our meeting times and give more hours on our other days.
- **Technical risks**
  Migration and Deployment
  Our plan of action:
  We have assigned two people with sound knowledge of both things but we will be testing it more than once.
- **Teamwork risks**
  Clash of ideas while working together
  Our plan of action:
  Healthy criticism

7. Project management

For our project management tool, we have decided to use **Trello** as it is easier and free to use with sufficient functionalities for us. We divided ourselves based on our skills and assigned everyone a role with Team Lead, Backend Lead, Frontend Lead, Github Master, Devops and Database Admin.

- **Team Lead:**
  The team lead organises the meeting every other day after the online meeting with the Class CEO. We discuss the milestones and divide our tasks according to the stack. The respective backend and frontend lead assign the tasks to their members and a day before our milestone submission team lead organises the meeting, compiles all the works, go through the whole document, print it and file it.

- **Frontend and Backend Lead:**
  The frontend and backend lead has a separate online meeting with their members and divides the tasks. The lead takes an update from the members and compiles before our next meeting.

- **Database Admin:**
  The database admin remains synchronised with the backend lead to create and complete the tasks. The admin overviews the tasks related to the database.

- **Github Master:**
  The github master maintains the team’s repo and decides branching strategy and ensures master branch is protected and that everyone adheres to the rules of merging into master.