# Online Restaurant Booking Platform

## Table of Contents

# A. Requirements Document

## Project Description

The online restaurant booking platform will allow customers to search, view information, and make reservations at restaurants. Customers can filter restaurants by location, cuisine, budget, etc. and view restaurant details including menu, reviews, location. Customers can create accounts to make and manage reservations online.

The system will also allow restaurant owners/staff to manage their business information and availability. They can add menu items, update hours of operation, view and respond to customer reviews, and monitor upcoming reservations.

## Business Rules

- Customers must create an account to make reservations
- Reservations require date, time, party size, and contact information
- Customers can write one review per restaurant
- Reviews require rating (1-5 stars) and optional comment
- Restaurants can add/update menu items with name, description, price
- Menu items can be categorized by cuisine type
- Restaurants set available reservation times/dates

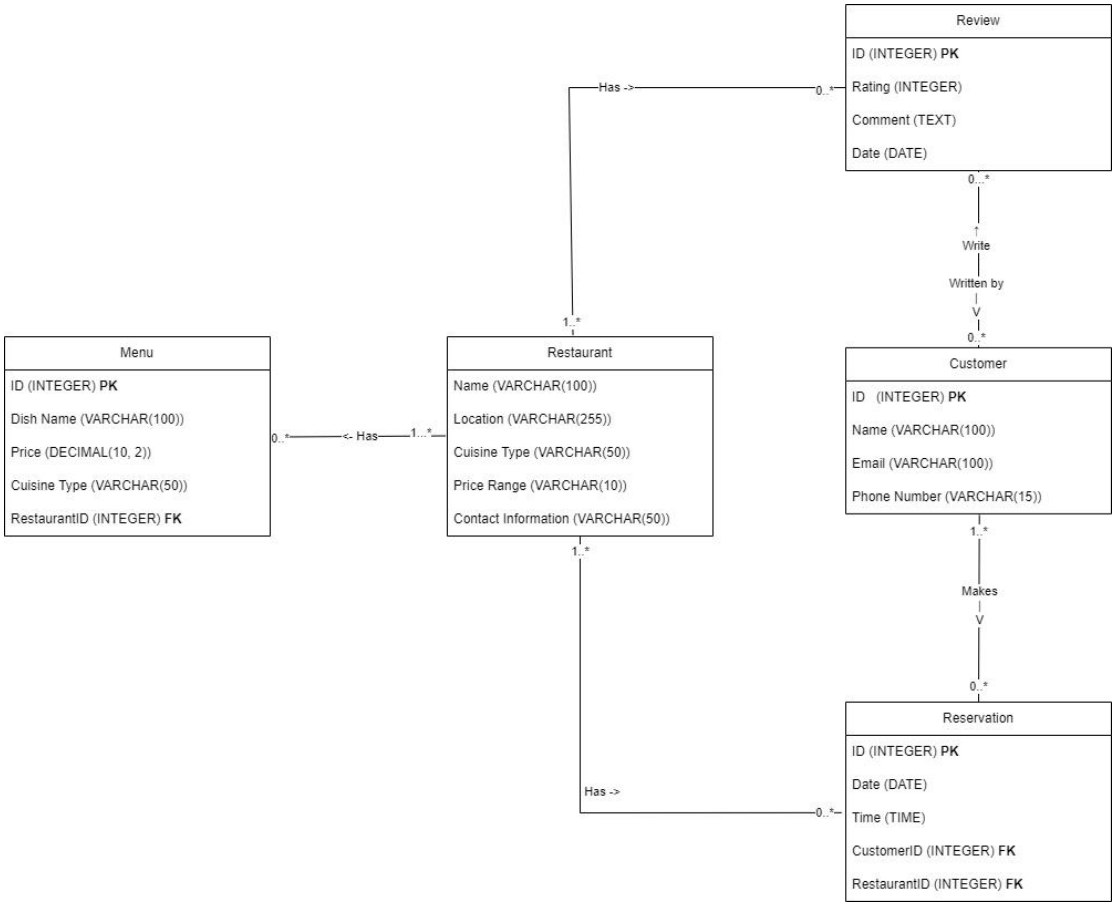
**Nouns:**

Customer, Restaurant, Reservation, Review, Menu, Menu Item, Account

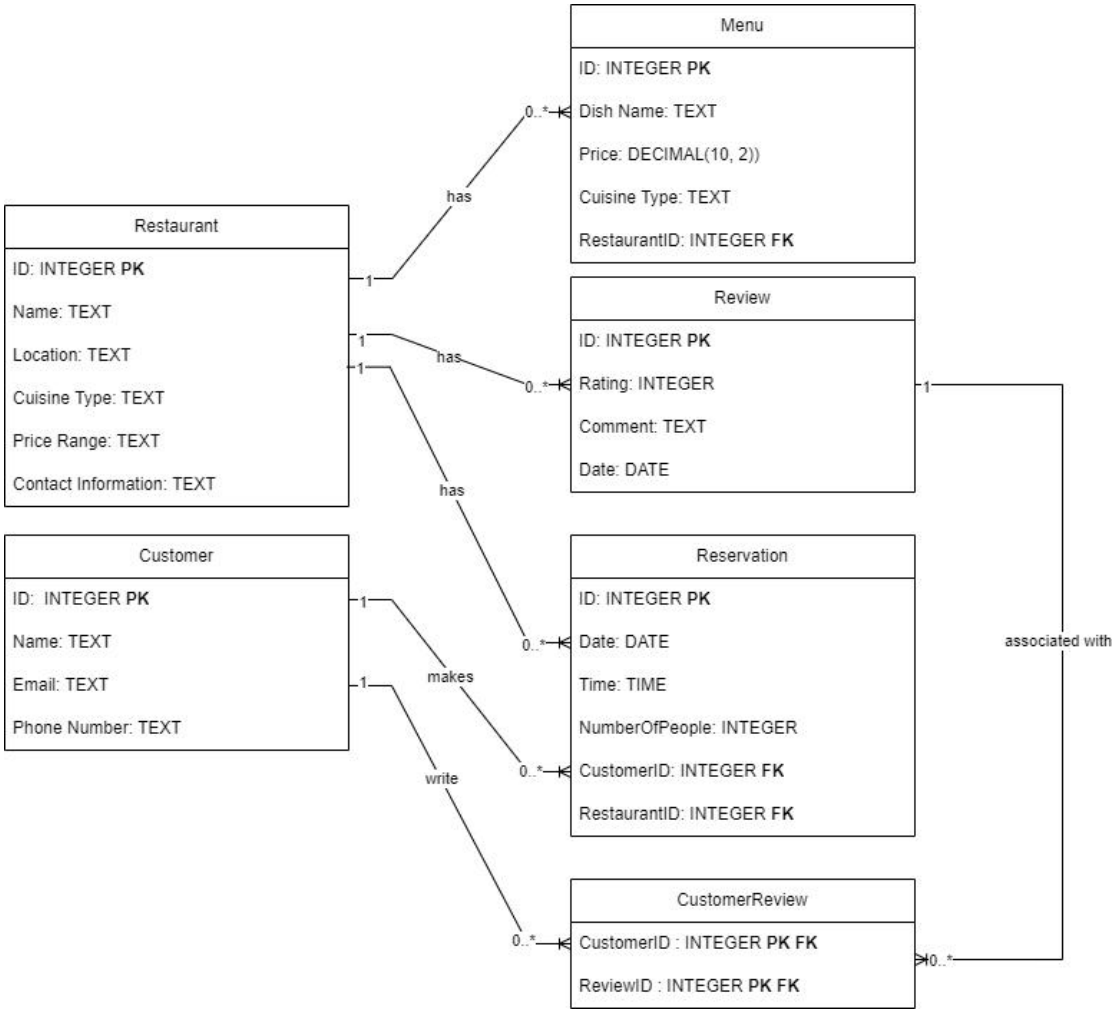
**Verbs:**
Search, View, Filter, Create, Make, Manage, Add, Update, Monitor, Set

This covers a high-level description of the online restaurant booking platform, key business rules, and identifies some initial nouns and verbs that will inform the data model. Let me know if you would like me to expand or modify this draft task 1 document in any way.

# B. UML Class Diagram


Review
ID (INTEGER) PK
Rating (INTEGER)
Comment (TEXT)
Date (DATE)
Has ->
0..*
1..*
0..*
Write
Written by
V
0..*
Menu
ID (INTEGER) PK
Dish Name (VARCHAR(100))
Price (DECIMAL(10, 2))
Cuisine Type (VARCHAR(50))
RestaurantID (INTEGER) FK
0..*
<- Has
1...*
Restaurant
Name (VARCHAR(100))
Location (VARCHAR(255))
Cuisine Type (VARCHAR(50))
Price Range (VARCHAR(10))
Contact Information (VARCHAR(50))
Customer
ID (INTEGER) PK
Name (VARCHAR(100))
Email (VARCHAR(100))
Phone Number (VARCHAR(15))
1..*
Makes
V
0..*
1..*
Has ->
0..*
Reservation
ID (INTEGER) PK
Date (DATE)
Time (TIME)
CustomerID (INTEGER) FK
RestaurantID (INTEGER) FK

# C. ERD


Menu
ID: INTEGER PK
Dish Name: TEXT
Price: DECIMAL(10, 2))
Cuisine Type: TEXT
RestaurantID: INTEGER FK
Restaurant
ID: INTEGER PK
Name: TEXT
Location: TEXT
Cuisine Type: TEXT
Price Range: TEXT
Contact Information: TEXT
Review
ID: INTEGER PK
Rating: INTEGER
Comment: TEXT
Date: DATE
Customer
ID: INTEGER PK
Name: TEXT
Email: TEXT
Phone Number: TEXT
Reservation
ID: INTEGER PK
Date: DATE
Time: TIME
NumberOfPeople: INTEGER
CustomerID: INTEGER FK
RestaurantID: INTEGER FK
CustomerReview
CustomerID : INTEGER PK FK
ReviewID : INTEGER PK FK
has
has
has
makes
write
associated with
1
0..*

## D. Functionalities that you selected to be used as an in-memory key-value storage

1. Most viewed products

Arrange the reviewed products in order of the number of times they have been reviewed, with the products with the most reviews listed first.

2. Current logged in users

Record all currently logged in users.

3. Most booked restaurants

Let the restaurants be sorted according to the number of times they have been ordered, with the ones with the most orders at the front.

## E. Redis data structures going to use

1.Most viewed products

To implement the most viewed products I will use a Redis sorted set with key "mostViewed:customerId", customerId uniquely identifies each user. This sorted set stores the products reviewed by this user and is sorted by the number of reviews.

2.Current logged in users

Since I only need to record all currently logged in users without sorting them, I can choose List as the structure to implement this function.

3.Most booked restaurants

Similar to Most viewed products, here I also choose to use sorted set as the structure to implement the Most booked restaurants function.

If Most Viewed Restaurants is to collect statistics for each customer, name this set as "mostBooked:customerId", customerId uniquely identifies each customer, this set saves all the restaurants booked by this customer, the key is the ID of each restaurant, and the value This is the number of reservations made at this restaurant.

If Most viewed products is for all customers, then the name of this set becomes "mostBooked", and the saved data is the number of restaurant orders accumulated by all customers. In the set, the key is the restaurant's ID and the value is the number of reservations.

## F. Application Demo

How to run the app

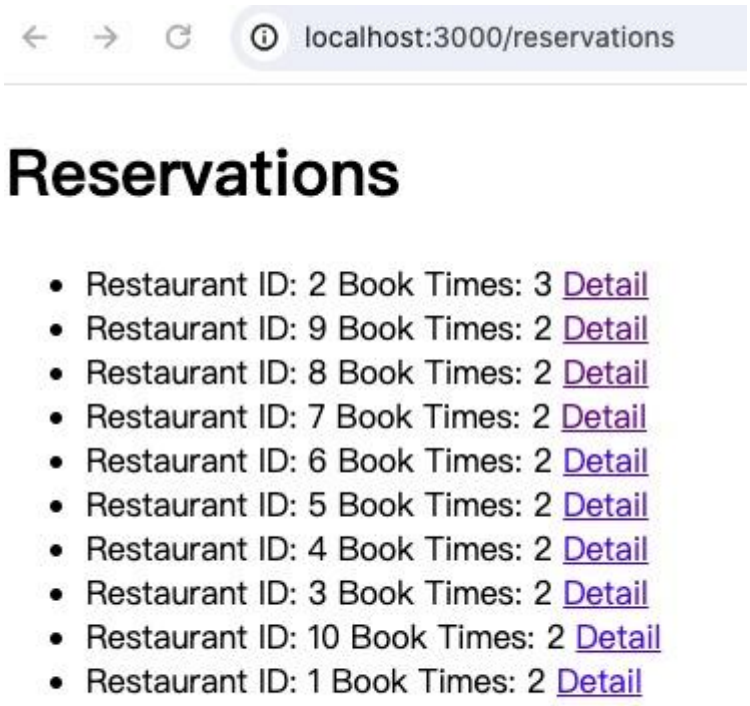
1. Enter the following command in the terminal:

```
cd Application
node ./app.js
```

2. Open http://localhost:3000 in your browser.

Default Page:


Most Booked Restaurants Page:



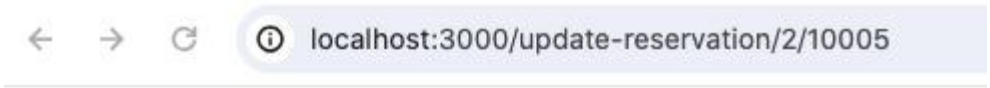

Reservation Detail Page:


# Restaurants Reservation Detail

- 10005 – 2022–12–05 – 09:10:25 – Update Delete
- 1111 – 111 – 11 – Update Delete
- 1112 – 1 – 2 – Update Delete

Back to Home

Update Reservation Page:

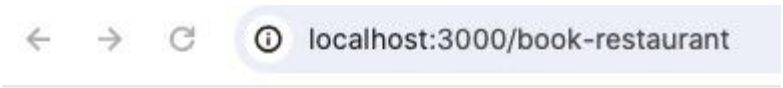



# Update Reservation

RestaurantID: 2
ReservationID: 10005
Date: 2022-12-05
Time: 09:10:25
CustomerID: 1005
Update

Back to Reservations

Book restaurant Page:




# Book Restaurant

RestaurantID:
ReservationID:
Date:
Time:
CustomerID:
Add

Back to Home