CS387: Database and Information Systems Lab

Project : Restaurant Management System
Project Design

Ayush Jangir (190050025)
D Chandrasekhara S S Hethahavya (190050031)
Koustav Sen (190050062)
Vishwanth K (190050131)

Department of Computer Science and Engineering
Indian Institute of Technology Bombay
2021-2022

Contents

Abstract

The domain we are dealing with is Restaurant management. Our goal is to make a database-backed front-end that makes managing the restaurant easy. Any person in the restaurant management stack can login to his account, and can easily interact with the database via the front-end forms provided.
There are mainly five kinds of users for this system.

Customers Book tables on premise, make online orders on the website, or via phone. Check their loyalty points, cash-back points on the website which they can redeem to get discounts.

Hotel Manager and Owner Manage items on the menu and their prices, Visualize the income and expenses of the company. Make changes to the menu according to popular dishes and the net income obtained from them .etc.. Also register employees to the database based on roles.

Waiter Check the website if any table is assigned to them. Check the comments for their service.

Delivery person Check the website if any delivery is assigned to them. Check the status of deliveries, comments on their service etc.

Chef They are responsible for maintaining the stock of ingredients as per the database point of view.

A database is a good idea for this task mainly because the volume of data is quite large. So querying the database for our necessary requirements is fairly easy as opposed to other search mechanisms in sets of files. Also data representation becomes less redundant using database due to concepts like normalization.
The database we are using is a relational database. This helps to capture certain features which only applies when a pair of entities interact. For example, a restaurant bill forms only when a customer orders a list of items. So to form a bill it is actually an interaction between customer and the menu.

1. Roles and use cases

1.1 Roles in the system

- Customer
- Manager
- Owner
- Chef
- Deliverer(Waiters and online delivery boys)
- Cashier

1.2 Use-Cases

1.2.1 Register customer

- **Primary Actor**: Customer
- **Description**: Customers must register themselves for access to any feature
-
  - **Inputs**: His name, email ID, phone number, address and password.
  - **Constraints**: As usual email ID should be valid and password should satisfy some minimum requirements like length ≥ 10, at least one character of each kind Capital, small, symbol, digit. email ID must be unique.
  - **Trigger**: Auto trigger add user to the customer table with the given details.
  - **Success Path**: Message of successful authentication or failed authentication as output and the User would taken to his role-specific home page.
  - **Exception**: Occurs if the password or email ID does not satisfy the constraints, or if the trigger finds that the password/ email ID are invalid. Appropriate error message is shown and the form would be cleared.
  - **Post-Conditions**:Upon success a new customer is added to the customer table

1.2.2 Logging in to the system

- **Primary Actor**: All the roles
- **Description**: Any roles logs in to the system to see his homepage and the views available for his role. Any forms that are intended for his role would only be accessible after logging in to the system.
-
  - **Inputs**: His email ID and password.
  - **Constraints**: As usual email ID should be valid and password should satisfy some minimum requirements like length ≥ 10, at least one character of each kind Capital, small, symbol, digit.
  - **Trigger**: email ID should be present in the role-appropriate database table. The password is checked if the password matches to the one stored in the database for that particular user name and role
  - **Success Path**: Message of successful authentication or failed authentication as output and the User would taken to his role-specific home page.
  - **Exception**: Occurs if the password or email ID does not satisfy the constraints, or if the trigger finds that the password/ email ID are invalid. Appropriate error message is shown and the form would be cleared.
  - **Post-Conditions**:No change in state of system

1.2.3 Table-Booking

- **Primary Actor**: Customer
- **Description**: Customer books the table at the restaurant.
-
  - **View**: In this page the customer will be able to see the time slots, unoccupied tables in that time-slots, and their locations like window side etc.
  - **Inputs**: Table id that the customer wishes to book, and the time-slot for which the table is to be booked. Slots will be in the length of 45 mins.
  - **Constraints**: table-id, time-slot should be from the view of free tables shown.
  - **Trigger**: The table is marked as assigned in the database for the respective time-slot against the respective customer-id.
  - **Success Path**: Message of successful booking. No further interactions with the system.
  - **Exception**: None
  - **Post-Conditions**: The table is booked against this user for this time-slot. So it won't be assigned to any other user in that time-slot.

1.2.4 Take-out

- **Primary Actor**: Customer
- **Description**: Customer orders online
-
  - **View**: In this page the customer will be able to see the menu
  - **Inputs**: Dishes along with their quantity
  - **Constraints**: dishes should be in the dishes table
  - **Trigger**: The system auto searches for an available delivery person and accepts the order
  - **Success Path**: Message of successful order if delivery person available
  - **Exception**: All delivery person may be unavailable. In that case ask user to wait or cancel order.
  - **Post-Conditions**: Upon successful order mark delivery person as unavailable and create bill in bills relation

1.2.5 Reviews

- **Primary Actor**: Customer
- **Description**: Customer gives reviews on orders
-
  - **View**: In this page the customer will be able to see paid bills
  - **Inputs**: rating for unrated bill
  - **Constraints**: rating should be 0-5
  - **Trigger**: The system adds a new entry to bills review table
  - **Success Path**: Message of successful review to customer
  - **Exception**: None
  - **Post-Conditions**: Upon successful review a new bill has been rated

1.2.6 Register Employee

- **Primary Actor**: Manager
- **Description**: Manager registers new employee
-
  - **Inputs**: Employee name, email ID, phone number, address, role, salary and password.
  - **Constraints**: As usual email ID should be valid and password should satisfy some minimum requirements like length ≥ 10, at least one character of each kind Capital, small, symbol, digit. email ID must be unique.
  - **Trigger**: The system adds a new employee in appropriate role table
  - **Success Path**: Message of successful entry to manager

- **Exception**: Occurs if the password or email ID does not satisfy the constraints, or if the trigger finds that the password/ email ID are invalid. Appropriate error message is shown and the form would be cleared.
- **Post-Conditions**: Upon successful addition a new employ is created

1.2.7 Fire Employee

- **Primary Actor**: Manager
- **Description**: Manager fires an employee
- - **Inputs**: Employee email ID, role
  - **Constraints**: As usual email ID should be valid and password should satisfy some minimum requirements like length ≥ 10, at least one character of each kind Capital, small, symbol, digit. email ID must be unique.
  - **Trigger**: The system deletes an employee in appropriate role table
  - **Success Path**: Message of successful deletion to manager
  - **Exception**: Occurs if the email ID not present in appropriate role table
  - **Post-Conditions**: Upon successful deletion an employ is removed

1.2.8 Assign Waiter to table

- **Primary Actor**: Manager
- **Description**: Manager assigns waiter to a table
- - **Inputs**: Table ID, Employee ID
  - **Constraints**: Employee ID must belong to a waiter
  - **Trigger**: Search for available waiters
  - **Success Path**: Successful assignment of waiter to table
  - **Exception**: None since no of waiters greater than no of tables and each waiter serves exactly on table
  - **Post-Conditions**: Mark employee unavailable

1.2.9 View Assigned Orders

- **Primary Actor**: Deliverer
- **Description**: Deliverer checks if he is assigned any order in this page and if yes, then he'll deliver that order.
- - **Inputs**: None.

- **Constraints**: The deliverer must be logged-in (which should be true anyway because this page is only-shown to logged in deliverers).
- **Trigger**: Once he lands in this page, a back-end query is run, and the order-id, address of delivery are fetched.
- **Success Path**: None.
- **Exception**: None
- **Post-Conditions**: None

1.2.10 Mark Order Delivered

- **Primary Actor**: Customer
- **Description**: Customer marks the order as delivered.
- - **View**: Customer views pending orders to be delivered
  - **Inputs**: Order-id. Rating/ Review.
  - **Constraints**: The customer marking this should have made the order corresponding to input order-id
  - **Trigger**: Mark the deliverer available and the order delivered. The rating field in deliverers view will get updated.
  - **Success Path**: None.
  - **Exception**: Arises if the user has not made the corresponding order. Respective error-message is shown.
  - **Post-Conditions**: User cannot mark this order as delivered anymore and can't change the feedback anymore.

1.2.11 Update stock

- **Primary Actor**: Chef
- **Description**: Chef can update stocks as and when required
- - **Inputs**: Ingredient ID, change amount, price per unit
  - **Constraints**: Ingredient ID must exist in stock table
  - **Trigger**: Update column regarding quantity and price in stock table corresponding to Ingredient ID
  - **Success Path**: Successful update message
  - **Exception**: ingredient id may be absent
  - **Post-Conditions**: Stock is updated in database

1.2.12 View Stock

- **Primary Actor**: Manager, Chef
- **Description**: Manager and Chefs can view current stock of ingredients available
-
  - **Inputs**: None.
  - **Constraints**: The primary actor must be logged-in
  - **Trigger**: Once he lands in this page, a back-end query is run, and the stock of ingredients is displayed
  - **Success Path**: None.
  - **Exception**: None
  - **Post-Conditions**: None

1.2.13 View Employee ratings

- **Primary Actor**: Manager
- **Description**: Manager can view ratings of his staff based on customer reviews
-
  - **Inputs**: None.
  - **Constraints**: The manager must be logged-in
  - **Trigger**: Once he lands in this page, a back-end query is run, and the ratings of his employees is displayed
  - **Success Path**: None.
  - **Exception**: None
  - **Post-Conditions**: None

1.2.14 View Monthly sales

- **Primary Actor**: Owner, Manager
- **Description**: Primary actor can view monthly sales/revenue
-
  - **Inputs**: Month
  - **Constraints**: The primary actor must be logged-in
  - **Trigger**: Once he lands in this page, a back-end query is run, and monthly sales and revenue is displayed
  - **Success Path**: None.
  - **Exception**: None
  - **Post-Conditions**: None

1.2.15 View Popular dish by course type

- **Primary Actor**: Owner, Manager
- **Description**: Primary actor can view top five most popular dishes grouped by course type
- – **Inputs**: Course type
  – **Constraints**: The primary actor must be logged-in
  – **Trigger**: Once he lands in this page, a back-end query is run, and top five most popular dishes grouped by course type
  – **Success Path**: None.
  – **Exception**: None
  – **Post-Conditions**: None

1.2.16 View most active days of week

- **Primary Actor**: Owner, Manager
- **Description**: Primary actor can view the most active days of week
- – **Inputs**: None
  – **Constraints**: The primary actor must be logged-in
  – **Trigger**: Once he lands in this page, a back-end query is run
  – **Success Path**: None.
  – **Exception**: None
  – **Post-Conditions**: None

1.2.17 View Menu

- **Primary Actor**: Customer
- **Description**: Get all available dishes along with their prices.
- – **Inputs**: None
  – **Constraints**: Even the un-logged person can view this page.
  – **Trigger**: System calculates the prices of all dishes taking into account the profit margin and base cost and sends the data back.
  – **Success Path**: User can click on any of these items to add them to checklist. But for this the user has to be logged in.
  – **Exception**: None
  – **Post-Conditions**: None

1.2.18 Create new Bill

- **Primary Actor**: Cashier
- **Description**: Prepare order bill for customer
-
  - **Inputs**: Customer ID, employee ID, Table ID, items list, quantity of each item, price of each item
  - **Constraints**: As required for each field
  - **Trigger**: System creates a new bill entry and remove table ID from booked table for that time slot and date
  - **Success Path**: User can see the bill invoice in his account
  - **Exception**: None
  - **Post-Conditions**: None

2. Entities and Relationships

2.1 Entities

- Stock
- Dishes
- Table
- Customer
- Employee
- Chef
- Manager
- Attendant

Customer
id
name
email_id
phone
address
password

Employee
id
name
email_id
phone
address
password
role
salary

Table
id
position
occupancy

Dish
id
name
is_veg
cuisine
item_type
profit

Stock
ingredient_id
name
quantity_left
min_required
price_per_unit

Bill
id
date
type

Chef
id
specialization

Manager
id
skill

Attendant
id
role(delivery-boy or waiter)

2.2 Relationships

2.2.1 Order

- **Arity**: 3-ary
- **Connects**: Customer, Bill and Employee
- **Attributes**:
  - **Rating**: Rating given by the customer for **service**.

2.2.2 Ordered_items

- **Arity**: Binary
- **Connects**: Bill and Dishes
- **Attributes**:
  - **Quantity**: Quantity of the dish ordered.
  - **Rating**: Rating given by the customer for **dish ordered**.

2.2.3 Ingredients

- **Arity**: Binary
- **Connects**: Dish and Stock
- **Attributes**:
  - **Quantity**: Quantity of stock item being used in dish.

2.2.4 Table_booking

- **Arity**: Binary
- **Connects**: Table and Customer
- **Attributes**:
  - **Time-slot**: The time-slot for which the table is booked.
  - **Day**: Day for which the booking has been done.

2.2.5 Assign Waiter

- **Arity**: Binary
- **Connects**: Table and Employee role as waiter
- **Attributes**: None

2.2.6 Pending Deliveries

- **Arity**: Binary
- **Connects**: Bill and Employee role as delivery person
- **Attributes**: None

3. ER Diagrams