Project Design Document

Restaurant Management System

Team -
Dhakne Ajay Sopan - 190050033
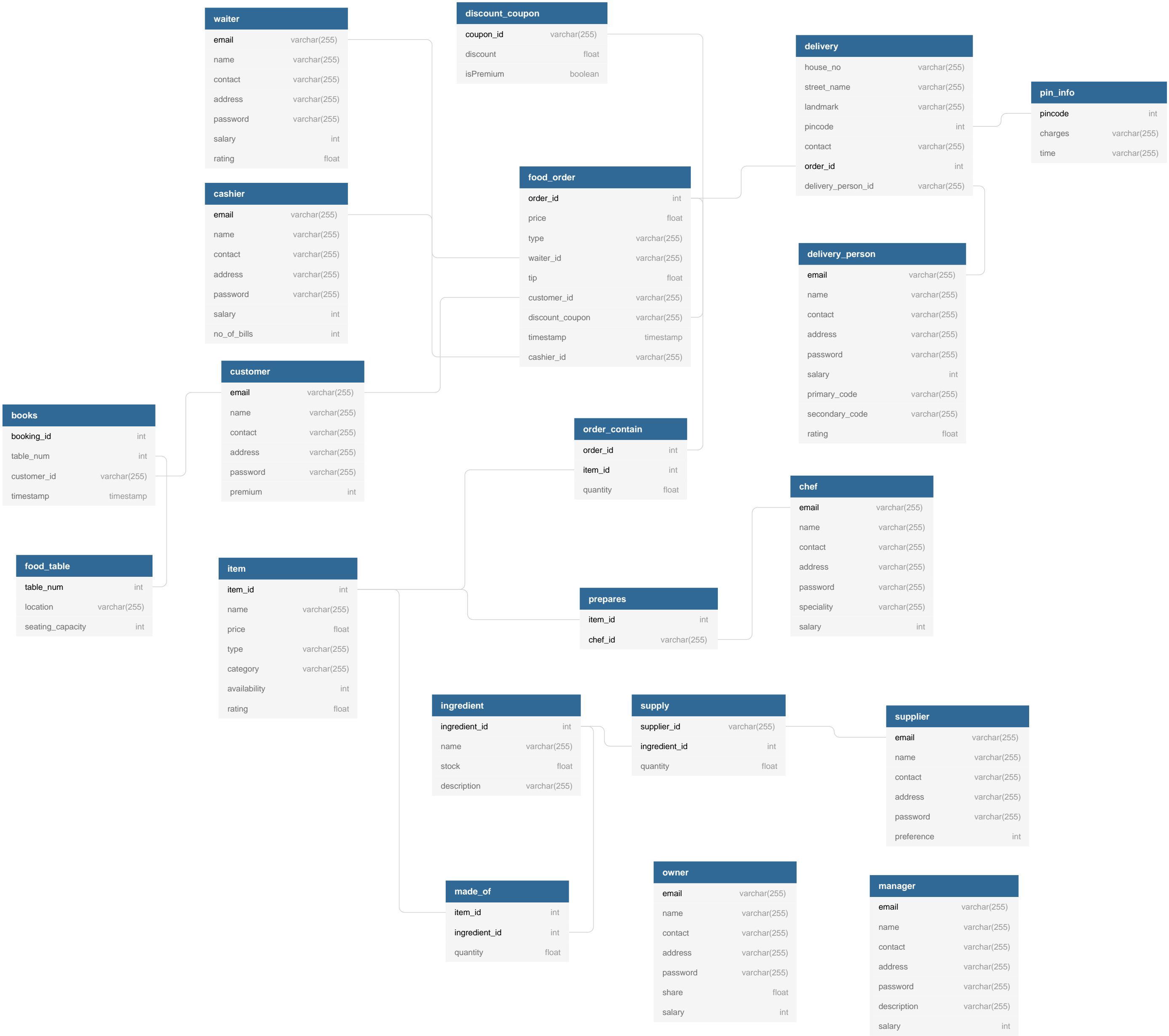Megha Baboria - 190050067
Vikas Panwar - 190050129

Integrity Constraints:

INTEGRITY CONSTRAINTS		
Relation	**Attribute**	**Reason**
manager	description	It is not important for manager to have a description associated if the there is a single manager
chef	speciality	Chef needs not have a speciality
delivery_person	rating	New delivery person would have rating as NULL
waiter	rating	New waiter would have rating as NULL
item	rating	New item would have rating as NULL
customer	password	Customer who has only ordered offline would have password as NULL
ingredient	description	extra information like storage intructions etc. Can be null
food_order	discount_cupon	If no discount cupon is used with the order , this attribute is null
food_order	waiter_id	If order is taken by the online or telephone mode then waiter has no role to play so its null
food_order	tip	similar idea as waiter_id
food_order	cashier_id	cashier we assume has no role to play in automatic orders so it can be null in that case
Others Integrity Constraints are :		
Relation	**Attribute**	**Reason**
owner	share	greater than 0 but less than 100
owner	sum of shares	Sum of shares of all owners must be less than or equal to 100
all employee tables	salary	salary must be greater than0
cashier	Total number of bills	Cashier makes a bill in case the order is made by offline or telephone mode. We also need to ensure that a bill is made for every such order. Thus number of bills made by all cashiers = total number of orders of type offline or telephone
cashier	no_of_bills	greater than or equal to 0
Item	Item Availability	Item availability should be 0 in case any of its ingredient is out of stock
customer	premium	greater than or equal to 0
item/ingredient/supply/made_of	stock/quantity	greater than or equal to 0
discount_coupon	discount	greater than 0 but less than 100
table	seating_capacity	greater than 0
food_order	type	ONLINE / OFFLINE / TELEPHONE
item	type	BEVERAGE / FOOD
category	category	VEG/NON-VEG/ALCOHOLIC/NON-ALCOHOLIC

Logical Schema is shown on the next page.

waiter

email	varchar(255)
name	varchar(255)
contact	varchar(255)
address	varchar(255)
password	varchar(255)
salary	int
rating	float

discount_coupon

coupon_id	varchar(255)
discount	float
isPremium	boolean

delivery

house_no	varchar(255)
street_name	varchar(255)
landmark	varchar(255)
pincode	int
contact	varchar(255)
order_id	int
delivery_person_id	varchar(255)

pin_info

pincode	int
charges	varchar(255)
time	varchar(255)

cashier

email	varchar(255)
name	varchar(255)
contact	varchar(255)
address	varchar(255)
password	varchar(255)
salary	int
no_of_bills	int

food_order

order_id	int
price	float
type	varchar(255)
waiter_id	varchar(255)
tip	float
customer_id	varchar(255)
discount_coupon	varchar(255)
timestamp	timestamp
cashier_id	varchar(255)

delivery_person

email	varchar(255)
name	varchar(255)
contact	varchar(255)
address	varchar(255)
password	varchar(255)
salary	int
primary_code	varchar(255)
secondary_code	varchar(255)
rating	float

customer

email	varchar(255)
name	varchar(255)
contact	varchar(255)
address	varchar(255)
password	varchar(255)
premium	int

books

booking_id	int
table_num	int
customer_id	varchar(255)
timestamp	timestamp

order_contain

order_id	int
item_id	int
quantity	float

chef

email	varchar(255)
name	varchar(255)
contact	varchar(255)
address	varchar(255)
password	varchar(255)
speciality	varchar(255)
salary	int

food_table

table_num	int
location	varchar(255)
seating_capacity	int

item

item_id	int
name	varchar(255)
price	float
type	varchar(255)
category	varchar(255)
availability	int
rating	float

prepares

item_id	int
chef_id	varchar(255)

ingredient

ingredient_id	int
name	varchar(255)
stock	float
description	varchar(255)

supply

supplier_id	varchar(255)
ingredient_id	int
quantity	float

supplier

email	varchar(255)
name	varchar(255)
contact	varchar(255)
address	varchar(255)
password	varchar(255)
preference	int

made_of

item_id	int
ingredient_id	int
quantity	float

owner

email	varchar(255)
name	varchar(255)
contact	varchar(255)
address	varchar(255)
password	varchar(255)
share	float
salary	int

manager

email	varchar(255)
name	varchar(255)
contact	varchar(255)
address	varchar(255)
password	varchar(255)
description	varchar(255)
salary	int

Title	Description	Trigger	Primary Actor	Input	Constraints	Exceptions	Post Conditions	Functional testing(tests)	SQL Query
Signup/Create Customer	User signup on the website or is added by the employee, the customer will be added by the cashier in offline order	Human	All	Form 1		If user is created by a employee, then the customer will have an issue while signing up. In this case forgot password option will instead be used	User is signed up. Details specific to user are visible now	(1)User already registered(	INSERT INTO customer (email_id, name, contact, address, password, premium_validity) VALUES ('$email_id', '$name', '$contact', '$address', '$password', '$premium_validity');
Logging In	Login using Credentials	Human	All	Form 3	User must be already registered		Orders/Details updated	(1)SQL Query returns empty result	SELECT password from customer where email_id = '$email_id';
Adding a new Customer	Add new Customer	Human	Manager/Owner/Cashier	Form 1	User must not be already registered	The cashier before adding looks for the customer in the database. If not found only then it is added	New user in the system	(1)New record added to customer relation (2)Customer may already exists(3)role adding customer has not privileges	INSERT INTO customer (email_id, name, contact, address, password, premium_validity) VALUES ('$email_id', '$name', '$contact', '$address', NULL, '$premium_validity');
Adding a new Employee (Except Manager)	Add any employee except manager	Human	Owner/Manager	Form10,12,13	User must be already registered	If user is not already registered, it is first registered and then assigned the role of the employee	New manger in the system	(1)Employee already exists(2)New	INSERT INTO cashier (email_id, name, contact, address, password, salary, no_of_bills) VALUES ('$email_id', '$name', '$contact', '$address', '$password', '$salary', 0);INSERT INTO cashier (email_id, name, contact, address, password, salary, no_of_bills) VALUES ('$email_id', '$name', '$contact
Adding a new Manager	Creating a New Manager	Human	Owner	Form 11	User must be already registered	If user is not already registered, it is first registered and then assigned the role of a manager		(1)Manager already exists(2)New	INSERT INTO manager (email_id, name, contact, address, password, salary, description) VALUES ('$email_id', '$name', '$contact', '$address', '$password', '$salary', '$description');
Remove a user	Remove user/employee	Human	Admin/Owner/Manager		User must exist		Removed user in the system	(1)Given user/email_id does not exists in the system(2)On success ,one record removed from user relation	DELETE FROM customer where email_id = '$email_id';
Rating	Give rating to item/delivery	Human	Customer		User must be already registered		Item no longer exits in the system	(1)Rating must be zero initially can not add to a NULL value(2)Rating updated for given employee(3)Given item/employee to be rated no longer exist/in the system(Query returns empty result	UPDATE item set rating = ('$rating' + (SELECT count(order_id) from order where order_id in (SELECT order_id from order_contain where item_id='$item_id'))*(SELECT rating from item where item_id = '$item_id'))/ (SELECT count(order_id) from order where order_id in (SELECT order_id from order_contain where item_id='$item_id')+1 where item_id = '$item_id';
Order confirmed	Change the amount of ingredients present	automatic	None		Amount of ingredients>= required for given order		Ingredients updated	(1)new record added to the order_contain table(2)stock in ingredient table gets updated	--update order_contain table INSERT INTO order_contain(order_id, item_id,quantity) VALUES ('$order_id','$item_id','$quantity'); --rest handled by trigger update_stock();
Order placed	Waiter adds the order(Offline or phone) / Customer adds the order online	Human	Waiter/Customer	Form 6	All items/ingredients in order must be available		Order table updated	(1)Query adds new row to the food_order table	
Book Table	Customer books a table	Human	Customer	Form 4	table is available		Availabe tables updated	(1)Table not available to book(Query returns false)(2)Table available (3) Books table gets updated(table_num) and availability set to false	select table_num ,location,capacity, availability from food_table; UPDATE books SET table_num ='$table_num'; UPDATE food_table SET availability ='$availability';
Order successful/delivered	Update the order table	automatic	Customer/Delivery Person		Payment must be recorded		Order table updated	(1)New record added in food_order table on success	INSERT INTO food_order(order_id, price,type,waiter_id,tip,customer_id, discount_coupon,timestamp, cashier_id) VALUES ('$order_id','$price', '$type', '$waiter_id','$tip','$customer_id','$disco unt_coupon','$timestamp','$cashier_id') ;
Order Confirmed(Online/Offline)	Chef prepares the order	Human	Chef		All items/ingredients in order must be available		Ingredients updated		
Analytics	Owner views the sales analytics	Human	Owner/Admin		userid,password:User must exists		None	(1)Query returns restaurant sale for day to day basis(2)Error when timestamp is NULL	select order_id,price,timestamp from food_order; select (sum(manager.salary)+sum (chef.salary)+sum(delivery_person. salary)+sum(cashier.salary)+sum (waiter.salary)) as total_salary from manager , chef,delivery_person, cashier,waiter;
Update Premium		Human	Premium Customer		Must be a premium customer		Premium validity is updated for the customer	(1)On success premium_validity column of customer is updated (set to new positive value)(2)returns empty result if customer is not a premium customer	select premium_validity from customer where WHERE email_id = '$customer_id' and premium_validity >0 ; ;UPDATE customer SET premium_validity='$validity' WHERE email_id = '$customer_id';
Create bill		Human	Cashier		Order must be successful		Order status is updated	(1)No of bills of cashier updated and new record added in food_order table	UPDATE cashier SET $no_of_bills='$no_of_bills' WHERE email_id = '$email_id'; -- update order table INSERT INTO food_order(order_id, price,type,waiter_id,tip,customer_id, discount_coupon,timestamp, cashier_id) VALUES ('$order_id','$price', '$type', '$waiter_id','$tip','$customer_id','$disco unt_coupon','$timestamp','$cashier_id') ;
Add/Remove/Update Item		Human	Chef	Form 8	Item must exist		Item is updated	(1)Item availability/stock gets updated (2)New record inserted in item table (3)Query may return item_id is not valid or item does not exists	UPDATE item SET $attribute='$attribute' WHERE item_id = '$item_id'; INSERT INTO item(item_id,name, price,type,category,availability,rating) VALUES ('$item_id', '$name', '$price','$type','$category','$availability', '$rating'); DELETE FROM item WHERE item_id = '$item_id';
Upgrade to Premium		Human	Normal Customer		Must be a Normal customer		Normal customer is updated to premium customer	(1)On success premium_validity column of customer is updated (set to new positive value)(2)returns empty result if customer is already a premium customer	UPDATE customer SET premium_validity='$validity' WHERE email_id = '$customer_id';
Add Coupon/Discount	Customer adds coupon (Premium/Non Premium) in order	Human	Customer		Coupon must be valid		Order details updated	(1)Query returns false if coupon does not exists in discount_coupon table(2) discount attribute of food_order table gets updated(3)Price of food_order relation gets updated	--check if coupon is valid select coupon_id ,discount from discount_coupon where coupon_id ='$discount_coupon'; --add coupon UPDATE food_order SET discount_coupon= '$discount_coupon' WHERE customer_id = '$customer_id' and order_id = '$order_id'; --update price UPDATE food_order SET price = price -price*discount WHERE customer_id = '$customer_id' and order_id = '$order_id';
Assign Delivery Person		Human	Manager		At least one Delivery Person Must be available		Delivery details updated	(1)New record added in delivery table (2)Delivery person not available - returns false	INSERT INTO delivery (house_no, street_name, landmark,pincode, contact,order_id,delivery_person_id) VALUES ('$house_no', '$street_name', '$landmark','$pincode','$contact','$orde r_id','$delivery_person_id');
Delivery	Deliver the order	Human	Deleevery person		Customer Address/pincode must be valid		Order details updated	(1)Delivery person made available	INSERT INTO delivery (house_no, street_name, landmark,pincode, contact,order_id,delivery_person_id) VALUES ('$house_no', '$street_name', '$landmark','$pincode','$contact','$orde r_id','$delivery_person_id');
Querying for a dish	Customer Queries for a item	Human	Customer		None		None	(1)On success item returned(2)item does not exists(3)item name wrong (4) item not available	select name,availability from item where name = '$name';
View/Edit Employee details(salary,etc)		Human	Manager/Owner	Form 10 - 13			Employee salary/details updated	(1)Employee columns updated(2) Employee does not exists (email invalid)(3)role must be manage or owner	UPDATE $employee SET salary ='$salary' WHERE email_id = '$email_id';
Supplier	Give/update preference to suplier	Human	Manager/Owner		None		Supplier preference updated	(1)Supplier does not exists(2)On success supplier preference is updated	UPDATE supplier SET preference = '$preference' WHERE email_id = '$email_id';
Querying for a best waiter/delivery person	Return waiter/delivery person with highest rating	Human	ALL				None	(1)Two or more persons with same best rating exists(2)returns best employee on success(3)Given employee no longer exists in the restuarant	select name from waiter order by rating DESC LIMIT 1; select name from delivery_person order by rating DESC LIMIT 1;
Most frequent customers	Customers with highest no. of orders placed	Human	Manager/Owner		Number Orders placed must be greater than some postive constant		None	(1)On success returns top 10 frequent customers	WITH X(c,customer_id) AS (select count(order_id) as c, customer_id FROM food_order group BY customer_id order BY c DESC LIMIT 10) select name from customer,X where customer.email_id = X. customer_id ;

Most Ordered/Item		Human	All		Number Orders placed must be greater than some postive constant		None	(1)Item name invalid or item not exists in item table(2)on success returns the item	WITH X(item_id,c) as (select item_id,count(order_id) as c FROM order_contain group by item_id order by c DESC) select name from item,X where order_contain.item_id = X.item_id LIMIT 1;
Highest rated Dish/Item		Human	All		Number Orders placed must be greater than some postive constant		None	(1)Item name invalid or item not exists in item table(2)on success returns the item(3)Two or more items with same rating exists (4)Item only ordered once but given max(rating)	select name , rating FROM item order BY rating DESC LIMIT 1;

Business Logic Controller

If user is customer:

    /signup- Signup page -

        check_if_user_exists(email) - `SELECT * FROM customer WHERE email = '$email'`

        add_user(email, name, contact, address, password, premium)

    / - redirects to login page (if not already logged in)

    /login - Login page

        user_authentication(email, password) - `SELECT * FROM customer where email = $email AND password = $password

    /add - Redirects to /

    /orders - list of orders for this customer - display_orders(customer_id)

        `SELECT * FROM food_order where customer_id = $customer_id`

    /order/$order_id - list of items and ratings given to them

        display_order (customer_id, order_id)

            `SELECT * FROM food_order where customer_id = $1 and order_id = $2`

    /book - book table page

        show_tables() - `SELECT * FROM food_table where table_num not in (SELECT table_num FROM books where now() - timestamp < interval 2 hour'

        book_table(customer_id, table_num, timestamp)

    /premium - shows premium validity

        show_validity(customer_id) - `SELECT premium FROM customer WHERE customer_id = $customer_id`

    /premium/update - takes to premium update page

        update_validity(customer_id, interval)- `UPDATE customer SET premium = premium+interval

    /menu - takes to the list of items page

        display_menu() - `SELECT name, price, rating FROM item`

    /menu/$item_id - takes to item detail page

        Display_item_details (item_id) - `SELECT a.name, price, type, category, availability, rating, b.name FROM (SELECT * FROM item) as a JOIN (SELECT * FROM made_of NATURAL JOIN ingredient) as b ON a.item_id = b.item_id WHERE a.item_id = $1`

For all employee -

    / - redirects to login page (if not already logged in), else employee detail page

For manager -

    /users - list of all users, filtered by type

    /users/$email_id - user detail page, can be deleted

    /add - form to add new customer/employee

    /supplier - list of suppliers

    /supplier/$email_id - view/edit supplier details

/stock - view/edit stock of ingredients
/menu - view/edit items from menu

For chef -
/menu - add/remove/update items

Link to project_ddl.sql and insert_data.sql -
https://drive.google.com/drive/folders/1E2RGFEfBPZhMKod5f
oI9nQnjSqzfu3O-?usp=sharing