

**PROJECT -2 REPORT**  
**DATABASE DESIGN**  
**SPRING 2019**

*Professor*

*Kong Li*

**MEAL MANAGER**

*Date of Submission*

05<sup>th</sup> May 2019

*Submitted by*

*Team #2*

Chaithra Lakshmi Sathyanarayana (Student ID: 012447565)

Thaijasa Badrinath Vijendranath (Student ID: 012470835)

Vineela Velicheti (Student ID: 012445186)

## **1. Choice of Database Project**

The mini world chosen as part of this project - Meal Manager, provides an interface between restaurants, dieticians and customers. The potential customers to this website will be people looking for meal subscriptions - breakfast, lunch and dinner, along with an option to enroll for a personalized diet plan.

With the order details known in advance, restaurant managers will be able to plan efficiently and thereby avoid food wastage. Consequently, they will offer lower prices as compared to the traditional approach. Customers, on the other hand, will get access to good food at a lower price and also benefit from skipping wait lines. Health conscious customers can choose from a group of affiliated dieticians to get personalized diet plans, and customers can choose meals based on their diet plan.

### **1.1. Business Rules**

1. Any restaurant/ Dietician will be registered to the application only by the admin.
2. Each customer will be provided with the following choices to select from
  - Meal type: Breakfast and/or Lunch and/or Dinner
  - Number of meals in a month: 15/30
  - Enroll with a dietitian: yes/no
3. Restaurant orders must be placed by customers on the previous day before 11:59 PM.
4. In a day, a customer will be able to place an order only for one item per meal type.
5. Dietician will need a maximum of 2 days to issue a new diet plan.
6. A customer enrolled with dietician should wait for a minimum of 3 days after the payment, to start the diet plan.
7. A customer enrolled with dietician will by default be registered under all meal types with 30 as count for each.
8. If a diet plan change request is initiated by a customer, he/she should wait for 3 days to start the updated diet plan.
9. A restaurant can offer an item under only one meal type.
10. As part of the update operation, a restaurant can change only the days on which the items are being offered. Any changes with respect to the other attributes of the item can be done by the combination of delete and add operations. For example, if a restaurant wants to change name of item from "A" to "B", restaurant should first delete item A and then add new entry with name as item B.

## **1.2. High Level Functioning of the Meal Manager Application**

Customers to the Meal Manager application will at first create an account by giving the basic registration details such as name, email ID, password. During the next registration step, customers will be displayed with options –either to get enrolled with a dietitian and a diet plan or not to get enrolled with a dietitian. The next and final page of the registration will be based on the option customer chooses in previous section. Customer who chooses to get registered with a dietitian will be prompted to enter a few of his lifestyle related details such as weight, height, any diseases that he/she is diagnosed with etc. After entering the respective data, he will make the payment to complete the registration process. On the other hand, if a customer chooses not to have dietitian, he will not be prompted to enter any lifestyle related data. He will make the payment to complete the registration process.

Any restaurant will be registered in to the application by admin of the application. The location to which restaurant belongs is entered along with the other basic details like email and password. When the restaurant adds items to its menu, nutritional information calories, carbohydrate content, fat content, protein content will be added for each item.

Similarly, any dietitian will be registered to the application by the admin of the application. A dietitian will use the customer given lifestyle information to formulate the diet plan and uploads the plan for that customer. A diet plan will include the following intake the customer can take for each day (15 days or 30 days based on the option selected) and for each meal type: calories, carbohydrate content, fat content, protein content.

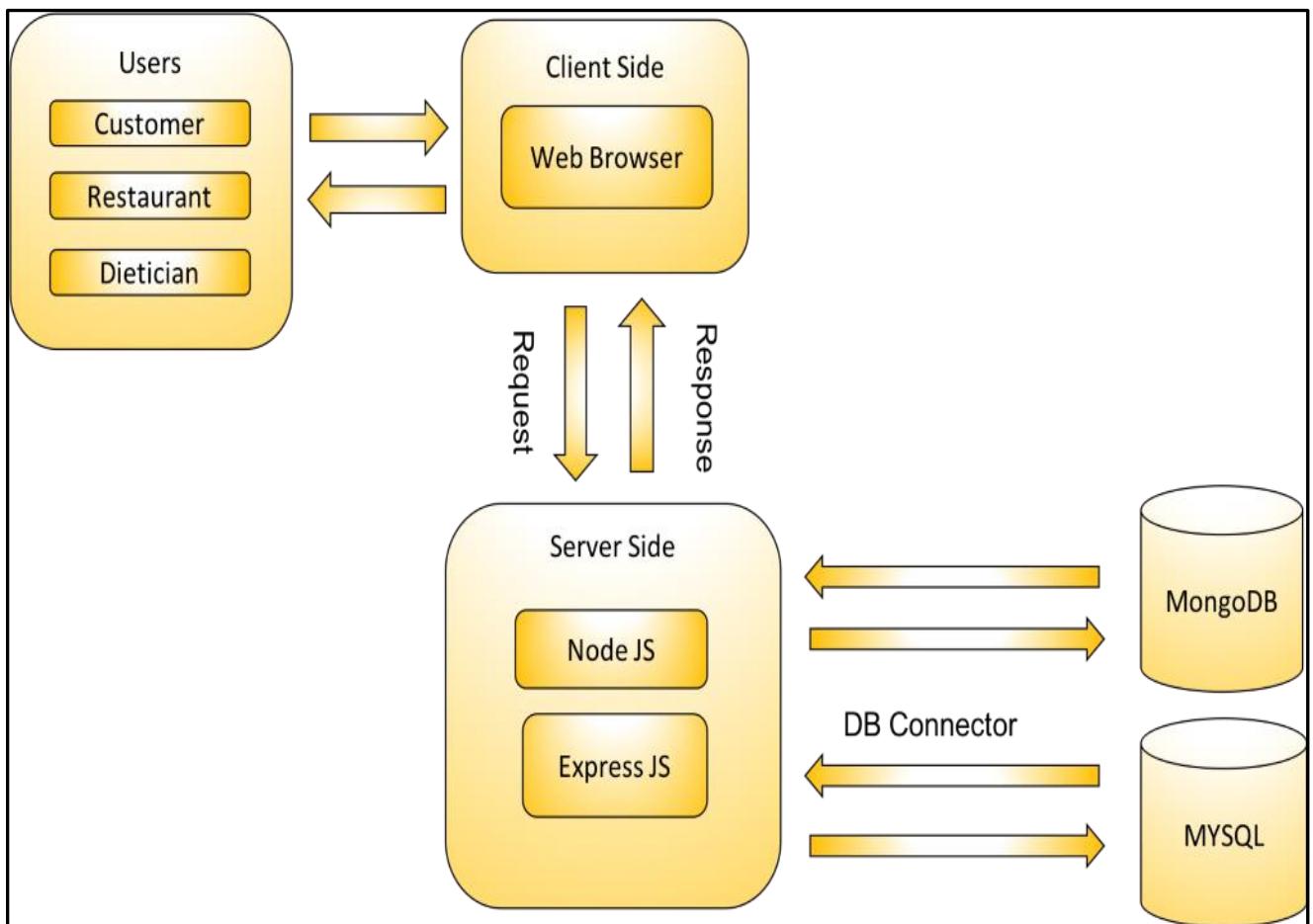
On any given day, a customer will be able to view menu items offered by the restaurant for the next day. Only those restaurants whose location match with the location of the customers will be displayed. Also, for each of the three meal types, breakfast/lunch/dinner, a customer will be able to view items from only those meal types to which he is enrolled for during the registration. Additionally, the menu items displayed will be further filtered for a customer enrolled with a dietitian. To do this, application considers the prescribed diet plan provided by the dietitian for that customer and matches with the item's nutritional information. Only those items which match the criteria will be displayed as item results.

As a last step, the customer chooses the items from the list displayed for him, selects the pick-up time slot and places the order.

## 2. Tech Stack

- Database Engine: MySQL, MongoDB
- Database Application Technologies
  - Front -end Technologies: HTML5, CSS3, JavaScript
  - Frameworks: Bootstrap, Express.js, ejs
  - Server-side Technologies: Node.js
- Operating System: Mac OS
- Languages: Javascript, SQL, HTML5, CSS3

## 3. Architecture



## **4. Final list of functionalities**

### **4.1. Modifications from ERD**

- Removed SELECTS\_AND\_PAYS\_FOR binary relation and replaced it with SELECTS\_AND\_GENERATES ternary relation. This is done to maintain history of all the previous plan details chosen by all customers.
- Created a new weak entity CUSTOMER\_BILLING to maintain all the previous billing information.
- Added new relation CHANGE\_REQUEST where a customer can request to change his diet plan.
- Added New\_Diet\_Plan\_Ind attribute to DIET\_PLAN entity. This acts as an indicator with values as Y or N where Y implies that diet\_plan\_id is still new and not addressed by the dietician and N implies that diet\_plan\_ID is not new and has been addressed by the dietician already.
- Removed Quantity attribute from ORDERS relation
- Removed Calendar\_Date from DIET\_PROGRESS and made "Day" attribute as a partial key.
- Added Offered\_Ind attribute in ITEM entity. This acts as an indicator with values as Y or N where Y implies item is offered currently by the restaurant and N implies item is not offered currently by restaurant for customers.
- Removed ORDERS binary relation and replaced with PLACES\_ORDER ternary relation. This helps to maintain the order history.
- Created a new weak entity ORDERS to maintain order history.

### **4.2. Functionalities and Status of Completion**

#### **4.2.1. Admin**

Functionality	Planned/New	Status
Admin must be able to log in	Planned	Complete
Admin must be able to register dieticians.	Planned	Complete
Admin must be able to register restaurants	Planned	Complete
Admin must be able to log out successfully	Planned	Complete

#### 4.2.2. Customer

Functionality	Planned/New	Status
Customer must be able to register	Planned	Complete
Customer who has registered must be able to log in	Planned	Complete
Customer may choose a meal plan	Planned	Complete
Customer must be able to view the next day's menu	Planned	Complete
Customer may select items from one or more restaurants	Planned	Complete
Customer may order an item from the menu of the selected restaurant.	Planned	Complete
Customer must be able to choose a pick-up time slot while placing an order	Planned	Complete
Customer must be able to rate items that he has ordered before	Planned	Complete
Customer should be able to see item ratings	Planned	Complete
Customer may enroll for a dietitian	Planned	Complete
Customer with a dietitian must have a diet plan	Planned	Complete
Customer must be able to see his current diet plan	Planned	Complete
Customer with a dietitian may input his progress for each day in the diet plan	Planned	Complete
Customer must be able to see his current diet progress	Planned	Complete
Customer may request for a change of diet plan from the enrolled dietitian	Planned	Complete
Customer must be able to view his diet history	New	Complete
Customer must be able to rate dietitians with whom he had enrolled before	Planned	Complete
Customer should be able to see dietitian ratings	Planned	Complete
Customer must be able to see his past item orders	New	Complete
Customer must be able to log out of the application	Planned	Complete

#### **4.2.3. Restaurant**

<b>Functionality</b>	<b>Planned/New</b>	<b>Status</b>
Restaurant must be registered via Admin	Planned	Complete
Restaurant must be able to log in	Planned	Complete
Restaurant must be able to view its item list as part of it's menu	Planned	Complete
Restaurant must be able to add an item to their menu	Planned	Complete
Restaurant must be able to update/delete an item from it's menu	Planned	Complete
Restaurant may access customer's orders for the current day	Planned	Complete
Restaurant may access customer's orders for the next day	Planned	Complete
Restaurant may request for sales report for a specific time period	Planned	Complete
Restaurant must be able to log out of the application	Planned	Complete

#### **4.2.4. Dietician**

<b>Functionality</b>	<b>Planned/New</b>	<b>Status</b>
Dietician must be registered via Admin	Planned	Complete
Dietician must be able to log in	Planned	Complete
Dietician must be able to view the list of customers currently enrolled under him/her	Planned	Complete
Dietician must be able to view the current diet plan of all enrolled customers	Planned	Complete
Dietician must be able to view current diet progress of all enrolled customers	Planned	Complete
Dietician must be able to view new diet plan requests from the customers	Planned	Complete

Dietician must be able to view new diet plan change requests from the customers	Planned	Complete
Dietician must provide diet plan for all new diet plan requests from customers	Planned	Complete
Dietician must be able to change customer's diet plan on request	Planned	Complete
Dietician must be able to log out of the application	Planned	Complete

## 5. Task Distribution

### 5.1. Timeline

Components	Duration
Proposal	February 20th
ERD	March 10th
ERD Changes	March 15th to March 20th
Normalization	March 22nd to March 29th; April 18th to April 22nd
Table Scripts	March 30th to April 7th
Insert Statements	March 30th to April 7th
Stored Procedures	April 8th to April 14th; April 18th to April 22nd
Views	April 8th to April 14th; April 18th to April 22nd
Trigger	April 8th to April 14th; April 18th to April 22nd
Index	March 30th to April 7th
Password Encryption	April 28th to April 30th
Multi statement transactions	April 28th to April 30th
UI Design Planning	March 20th - March 21st; April 18th to April 22nd
Node JS installation and learning	March 30th to April 7th
Session and Logs	March 30th to April 7th
EJS Pages Creation	April 8th to April 14th; April 18th to April 22nd
CSS Styling	April 8th to April 14th; April 18th to April 22nd
Mongo DB	April 22 <sup>nd</sup> to April 30th

Screenshots from DB and application	May 1st to May 5th
Normalization justification	May 1st to May 5th
Functionalities	May 1st to May 5th
Test cases	May 1st to May 5th
Report	May 1st to May 5th
Presentation slides	May 1st to May 5th

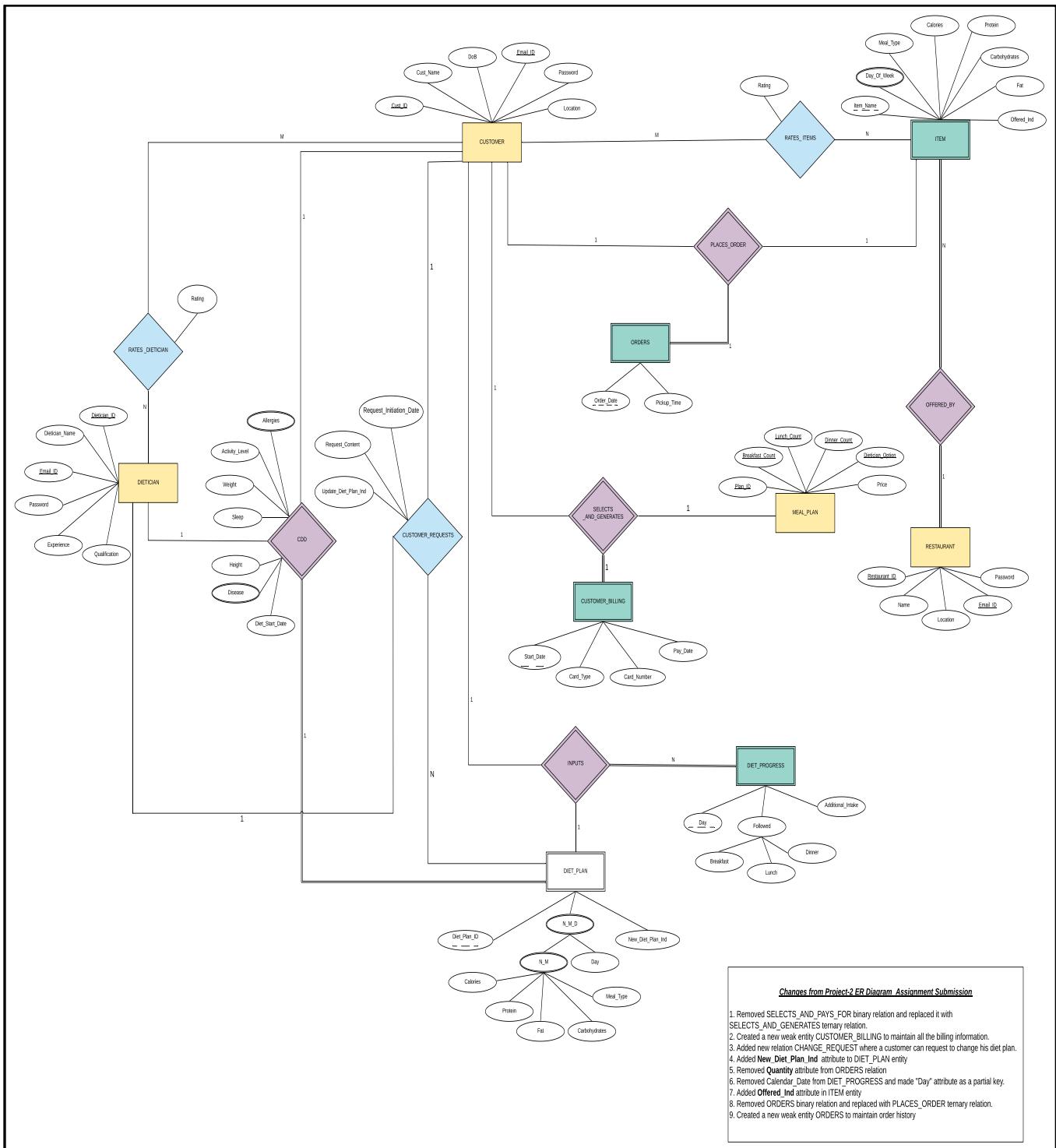
## 5.2. Individual Contributions

Student Name	Component	Task
<b>Chaithra Lakshmi Sathyanarayana (ID: 012447565)</b>	DB	Normalization, Table Scripts, Insert Statements, Stored Procedures, Views, Trigger, Index, Password Encryption, Multi statement transactions, MongoDB
	UI	Restaurant (view menu, add menu), Dietician - Full
	Documentation	Screenshots from DB and application, Normalization justification, Functionalities, Test cases, Report, Presentation Slides
<b>Thaijasa Badrinath Vijendranath (ID: 012470835)</b>	DB	Normalization, Table Scripts, Insert Statements, Stored Procedures, Views, Trigger, Index, Password Encryption, Multi statement transactions, MongoDB
	UI	Restaurant (update/delete menu, view report, view today's orders, view tomorrow's orders), Customer Registration, Admin - Full
	Documentation	Screenshots from DB and application, Normalization justification, Functionalities, Test cases, Report, Presentation Slides
<b>Vineela Velicheti (ID: 012445186)</b>	DB	Normalization, Table Scripts, Insert Statements, Stored Procedures, Views, Trigger, Index, Password Encryption, Multi statement transactions, MongoDB
	UI	Customer - Full

	Documentation	Screenshots from DB and application, Normalization justification, Functionalities, Test cases, Report, Presentation Slides
--	---------------	--

## 6. Final Design of Database Portion

### 6.1. ER Diagram



## 6.2. Specification of DB Objects

### 6.2.1. Tables

	<b>Table Name</b>	<b>Column Name</b>	<b>Description of the Column</b>
1	Customer: customer related basic details		
		cust_id	Unique identification number for customer
		cust_name	Name of the customer
		dob	Date of birth of the customer
		email_id	Email ID used by the customer for registration to the application
		password	Password used by the customer for registration to the application
		location	Location to which customer belongs to
2	Meal_plan: defined meal plans of the application		
		Plan_ID	Unique identification number for a unique combination of breakfast_count, lunch_count, dinner_count and dietician_option
		Breakfast_Count	Number of breakfast meals in a month
		Lunch_Count	Number of lunch meals in a month
		Dinner_Count	Number of dinner meals in a month
		Dietician_Option	"Yes or No options to signify dietician required in the plan and dietician not required in the plan"
		price	"total price based on breakfast_count, lunch_count, dinner_count and dietician_option"
3	customer_meal_plan: customers with selected with meal plan		
		cust_id	Unique identification number for customer
		start_date	Date on which selected meal plan to be started
		plan_id	Plan_ID selected by customer
4	Customer_Billing: payment details of customers for each enrollment		
		Cust_ID	Unique identification number for customer

		Start_Date	Date on which selected meal plan to be started
		Card_number	Card number of the card used for payment
		card_type	Category of the card type used for payment
		pay_date	Date on which payment is made for the mealplan
5	Restaurant: restaurant related basic details		
		restaurant_id	Unique identification number for restaurant
		name	Name of the restaurant
		email_id	Email ID used by the restaurant for registration to the application
		password	Password used by the restaurant for registration to the application
		location	Area in which restaurant is located
6	Item: food items names with nutritional value		
		Restaurant_ID	Unique identification number for restaurant
		Item_Name	Name of the food item
		Meal_Type	Category of meal in Breakfast/Lunch/Dinner
		Calories	Total calories in the food item
		Proteins	Total protein content in the food item
		Carbohydrates	Total carbohydrates content in the food item
		Fat	Total fat content in the food item
		offered_ind	Indicator with values as Y or N where Y implies Item_Name is offered currently by Restaurant_ID and N implies Item_Name is not offered currently by Restaurant_ID for customers
7	dietician: dietician related basic information		
		dietician_id	Unique identification number of dietitian
		dietician_name	Name of the dietitian
		email_id	Email ID used by the dietitian for registration to the application
		password	Password used by the dietitian for registration to the application

		experience	Number of years of professional experience that the dietitian has
		qualification	Academic qualification of the dietitian
8	restaurant_item_day: restaurant offered items on days		
		Restaurant_ID	Unique identification number of restaurant
		Item_Name	Name of the food item
		day_of_week	Day on which the Item_Name is offered by Restaurant_ID
9	diet_plan: customers with selected dietplans and basic lifetsyle details		
		cust_ID	Unique identification number of customer
		dietician_ID	Unique identification number of dietician
		diet_plan_id	Unique ID generated when each time a customer enrolls for a new plan_ID. The ID starts with 1 for new customer and increments there on
		diet_start_date	Date on which selected meal plan to be started
		new_diet_plan_ind	Indicator with values as Y or N where Y implies that diet_plan_id is still new and not addressed by the dietician_ID and N implies that diet_plan_ID is not new and has been addressed by the dietician_ID
		activity_level	Three levels of physical activity level indicators on an average daily basis which are low/medium/high
		weight	Measured weight in lbs of the customer
		sleep	"Three levels of sleep customer gets on an average daily basis which are low/medium/high
		height	Measured height in cms of the customer
10	diet_plan_details: details of each dietplan of customers		
		cust_ID	Unique identification number of customer
		dietician_ID	Unique identification number of dietician

		diet_plan_id	"Unique ID generated when each time a customer enrolls for a new plan_ID. The ID starts with 1 for new customer and increments there on"
		day	Day of the month. Starts from 1 and increments till 30
		meal_type	Category of meal in Breakfast/Lunch/Dinner
		calories	"Dietician suggested calories content for cust_ID with diet_plan_id for a day and for a meal_type"
		proteins	"Dietician suggested protein content for cust_ID with diet_plan_id for a day and for a meal_type"
		carbohydrates	"Dietician suggested carbohydrates content for cust_ID with diet_plan_id for a day and for a meal_type"
		fat	"Dietician suggested fat content for cust_ID with diet_plan_id for a day and for a meal_type"
11	allergy: food allergies if any associated with customers		
		cust_ID	Unique identification number of customer
		dietician_ID	Unique identification number of dietician
		diet_plan_id	Unique ID generated when each time a customer enrolls for a new plan_ID. The ID starts with 1 for new customer and increments there on"
		allergy	Type of allergy that cust_ID has when enrolled for diet_plan_ID with dietician_ID
12	disease: diseases if any associated with customers		
		cust_ID	Unique identification number of customer
		dietician_ID	Unique identification number of dietician
		diet_plan_id	Unique ID generated when each time a customer enrolls for a new plan_ID. The ID starts with 1 for new customer and increments there on

		disease	Type of disease that cust_ID has when enrolled for diet_plan_ID with dietician_ID
13	diet_progress: diet progress made by the customers of various dietplans		
		cust_ID	Unique identification number of customer
		dietician_ID	Unique identification number of dietician
		diet_plan_id	Unique ID generated when each time a customer enrolls for a new plan_ID. The ID starts with 1 for new customer and increments there on
		day	Day of the month. Starts from 1 and increments till 30
		breakfast	Indicator with values as Y or N which suggests whether or not cust_ID followed diet_plan_id on day for breakfast

### 6.2.2. Views

#### Customer

1. diet\_ratings\_view - This view is created to combine the average ratings received by each dietician from all the customers, with the other details of the dietician like name, email ID and qualification. If no ratings are provided to a particular dietician, then 'No Ratings' is displayed.
2. c\_customer\_remaining\_meal\_count: For each customer this view provides the number of Breakfast, Lunch and Dinner meals left in his current meal plan. This is calculated by taking the meal count from the meal plan and the Order details from Orders table. The information provided by this view is used as part of the c\_menu\_view.
3. c\_menu\_view: For each customer this view gives the list of items available from the restaurants in the same location as customer for the next day. For dietician enrolled customers, only those items which match the diet plan criteria are displayed. For other customers all the items available in the same location as the customer will be displayed
4. c\_order\_history\_view: The purpose of this view is to give the complete history of orders for each customer. For each order the view returns customer id, meal type, Item name, Restaurant name, Order date and pick up time. This information is also displayed as part

of “rate an item” page, as a customer should be able to rate only those items he had ordered.

5. c\_diet\_plan\_details: For all customers that are currently enrolled with a dietitian, this view returns the current diet plan details. It provides the nutritional intake the customer is supposed to consume each day in his diet plan. If a customer is enrolled with a dietitian in his past and currently has just a meal plan, there will be no records for that customer under this view.
6. c\_diet\_progress\_details: For all customers that are currently enrolled with a dietitian this view gives the current diet plan progress details. It will provide information if the customer has followed the suggested diet and if he had any additional food. For the view to provide this information, they must be submitted by customer first.
7. c\_diet\_plan\_list\_view: The purpose of this view is to return the list of past diet plans for each customer. It includes information like dietitian name, diet plan id, diet start date etc.
8. c\_diet\_history\_view: The purpose of this view is to return the complete information for the list of diet plans each customer has previously enrolled. It provides all the nutritional intake information the customer was supposed to take each day and also lists any additional calorie intake consumed by the customer. Along with this, the view also displays any diet progress updated by customer. If the customer did not update the progress for any completed day in the diet plan, this view doesn't return any details for that day.
9. c\_dietician\_list\_view: This view displays the list of dieticians each customer has enrolled, both in past and present. This information is displayed to the customer in the “rate a dietitian” page, as a customer should be able to rate only those dieticians he had enrolled.

### **Dietician**

1. d\_customers\_diet\_plan\_id\_view – This view used to find all diet plan ids for a customer, along with customer details.
2. d\_customer\_diet\_plan\_details\_view – This view gives the diet plan details for all 30 days for all 3 meal types – breakfast, lunch and dinner for all the diet plans ids of a customer.

3. d\_customer\_diet\_plans\_list\_view – This view is used to find all diet plans associated with a customer along with start dates, pay dates, dietitian of the diet plans.
4. d\_customer\_dietplan\_progress\_view – This view gives the diet plan details for all 30 days for all 3 meal types for all diet plans of a customer. It also shows the day-to-day diet progress details the customer has given.
5. d\_diet\_plans\_cust\_health\_details\_view – This view shows the customer health details like allergies and diseases associated with each diet plan of the customer.
6. diet\_plan\_day\_meal\_type\_view – This view is used to get the days and meal types in a typical 30-day diet plan.
7. d\_dietician\_customers\_view – This gives all the customers and customer details like diet start date, diet request date associated with a dietitian.
8. d\_update\_diet\_plan\_dates\_view – This gives the existing diet plan details for the diet days for which the diet plan update request is submitted.
9. d\_update\_diet\_plans\_list\_view – This gives the list of diet plan update requests associated with a dietitian.

### **Restaurant**

1. r\_restaurant\_item\_details\_day\_view – This gives the list of items and item details like calories, proteins, days of which the item is offered for a restaurant.
2. r\_update\_menu\_view – This view gives the items list along with whether or not it is offered on a particular day of week for each restaurant.

#### **6.2.3. Stored Procedures**

##### **Admin**

1. a\_check\_admin\_login\_info: The procedure is called to validate the login credentials of the admin user. It takes as input email ID and password from the UI and performs a

database check. It gives as output the count matching the email ID and password, the email ID of the admin.

2. proc\_d\_ins\_dietn\_regn\_details : The procedure is called to insert the details of the dietician from UI to database. This is called during the registration of the dietician by the admin. It takes as input name, email ID, password, experience in years and academic qualification. It gives as output as 1 if the insertion is successfully completed, If the email ID already exists, insertion will not happen and returns 0 as the output.

3. proc\_r\_ins\_rest\_details: The procedure is called to insert the details of the restaurant from UI to database. This is called during the registration of the restaurant by the admin. It takes as input name, email ID, password, and location. It gives as output as 1 if the insertion is successfully completed. If the email ID already exists, insertion will not happen and returns 0 as the output.

### **Customer**

1. c\_check\_customer\_login\_info: This stored procedure is called to validate the login credentials of the customer. It takes as input email ID and password from the UI and performs a database check. It gives as output the count matching the email ID and password, customer type. Customer type will be one of the following depending on their respective scenarios:

1. None: Not enrolled in any diet plan or meal plan
2. Diet-Regular: Enrolled in either a diet plan or a meal plan with startdate is less than or equal to tomorrow
3. Start\_Date\_Late: Enrolled in either a diet plan or meal plan but start date is greater than tomorrow

2. c\_place\_order: This stored procedure is used to place an order for the items selected by the customer. It takes as input item name, restaurant name and pick up time for each meal type along with the customer id. This procedure first checks if there is an existing record for the chosen meal type and if there is no record only then the order will be inserted. It returns as output three variables: bf\_chk, ln\_chk and din\_chk. If there are no existing records with the specified details they will have a value of 0 indicating success else they will have a value of 1.

3. c\_request\_diet\_plan\_change: This stored procedure is used to insert the new change diet plan request made by customer. It takes as input customer id, initiation date,

request description. If there is an existing request from same customer for same diet plan, it will be overwritten and the output variable v\_check value will be 1. If there are no existing records, v\_check will be 0.

4. c\_update\_diet\_progress: This stored procedure is used to insert the new diet progress information provided by customer. It takes as input customer id, log date, if customer has followed breakfast, lunch, dinner and if he had any additional intake. If there is an existing update from same customer for same date, it will be overwritten and the output variable v\_cnt value will be 1 indicating the record is overwritten. If there are no existing records, v\_cnt will be 0.
5. c\_update\_item\_rating: This stored procedure is used to insert a new item rating provided by customer. It takes as input customer id, restaurant id, item name and rating. If there is an existing rating from same customer for same item, it will be overwritten and the output variable v\_cnt value will be 1 indicating the record is overwritten. If there are no existing records, v\_cnt will be 0.
6. c\_update\_dietician\_rating: This stored procedure is used to insert a new dietitian rating provided by customer. It takes as input customer id, dietitian id and rating. If there is an existing rating from same customer for same dietitian, it will be overwritten and the output variable v\_cnt value will be 1 indicating the record is overwritten. If there are no existing records, v\_cnt will be 0.
7. proc\_c\_ins\_cust\_details: The procedure is called to insert the details of the customer from UI to database. This is called during the registration of the customer. It takes as input name, email ID, password, date of birth and location. It gives as output as 1 if the insertion is successfully completed. If the email ID already exists, insertion will not happen and returns 0 as the output.
8. proc\_dietician\_yes: The procedure is called to insert the details of customer who wishes to enroll with a diet plan. It takes as input the customer enters and inserts record into five tables based on the respective fields. It checks for the previous diet plans the customer has had and increments the diet plan ID accordingly. If the insertions happen successfully then 1 is returned else 0 is returned.
9. proc\_dietician\_no: The procedure is called to insert the details of customer who does not enroll with a diet plan. Because this customer will have no additional information related to lifestyle this involves inserting data only to two tables. If the insertions happen successfully then 1 is returned else 0 is returned.

## Dietician

1. `d_check_dietician_login_info`: The procedure is called to validate the login credentials of the dietitian user. It takes as input email ID and password from the UI and performs a database check. It gives as output the count matching the email ID and password, the email ID of the dietitian.
2. `d_add_diet_details`: The procedure is used to add and update diet plan details like calories, proteins for all meal types (Breakfast, Lunch and Dinner). If the diet plan details are present for a particular day, this procedure updates the details, otherwise inserts the diet plan. It gives as output as 1 if the operation is successfully completed, -1 if the insertion is not successful.
3. `d_finalize_new_diet_plan`: The procedure is used to finalize the new diet plan of a customer. If the dietitian is happy with the diet plan he has prepared for a customer and does not want to make any further changes, then he can use the "Finalize Diet Plan" option from UI, and this procedure is called. It updates the `new_diet_plan_ind` column in `diet_plan` table to 'N'. It gives as output as 1 if the operation is successfully completed, -1 if the insertion is not successful.
4. `d_finalize_update_diet_plan`: The procedure is used to finalize the updated diet plan of a customer. If the dietitian is happy with the diet plan he has prepared for a customer and does not want to make any further changes, then he can use the "Finalize Diet Plan" option from UI, and this procedure is called. It updates the `update_diet_plan_ind` column in `diet_customer_requests` table to 'N'. It gives as output as 1 if the operation is successfully completed, -1 if the insertion is not successful.

## **Restaurant**

1. `r_check_restaurant_login_info`: The procedure is called to validate the login credentials of the restaurant user. It takes as input email ID and password from the UI and performs a database check. It gives as output the count matching the email ID and password, the email ID of the restaurant.
2. `r_add_menu_item`: The procedure is called whenever a restauranr adds a new item to its menu. It takes the item name, nutritional information like calories,carbohydrates, meal type and the days of week on which the item is offered. The procedure checks if the item name is already present for that particular restauran and if it's a new item, then the new item is inserted in item table, and days of week on which the item is offered will be updated in `restaurant_item_day` table. It returns

output as 1 in case of successful previously mentioned operation. If the item is already present for the restaurant, then it returns 2 as output and does not insert the item. In case of DB errors, 0 is returned.

3. proc\_r\_generate\_report: The procedure is called when the restaurant requests for a sales report for a specific time period. It takes the input as dates to which the report is required and returns the items ordered during that time period, the number of times each item is ordered, and the rating received for each item. This will perform only select operations from combination of three tables.
4. proc\_update\_menu: The procedure is called when the restaurant chooses to update the menu. Item to be updated along with days of week variables are taken as the input. Based on the updated menu, for each item, the item name and the day of the week on which it is offered is updated in the respective table. Similarly, if an item is removed being offered on a particular day, the respective entry is deleted from the table. It performs the check and corresponding insertion or deletion for each day of the week.

### 6.3. Functional Dependencies and Normalization

#### 1. Customer

**Table:** Customer (Cust\_ID,Cust\_Name,DOB,Email\_ID,Password,Location)

**FDs:**

Cust\_ID → Cust\_Name  
Cust\_ID → DOB  
Cust\_ID → Email\_ID  
Cust\_ID → Password  
Cust\_ID → Location

**Normalization:**

- **1NF:** Yes, this relation is in 1NF because:
  - It has a primary key Cust\_ID
  - All the non-primary keys depend on primary key Cust\_ID
  - All the values are atomic.
  - There is no multivalued attribute
  - There is no nested relation

- **2NF:** Yes, this table is in 2NF because it is in 1NF and has no partial dependencies. Below are the details for each functional dependency.

<u>Functional Dependency</u>	<u>Partial Dependency</u>
Cust_ID → Cust_Name	All non-key attributes are fully functionally dependent on primary key
Cust_ID → DOB	All non-key attributes are fully functionally dependent on primary key
Cust_ID → Email_ID	All non-key attributes are fully functionally dependent on primary key
Cust_ID → Password	All non-key attributes are fully functionally dependent on primary key
Cust_ID → Location	All non-key attributes are fully functionally dependent on primary key

- **3NF:** Yes, this table is in 3NF because it is in 2NF and has no transitive dependencies. Below are the details for each functional dependency.

<u>Functional Dependency</u>	<u>Transitive Dependency</u>
Cust_ID → Cust_Name	All non-key attributes are directly dependent on primary key
Cust_ID → DOB	All non-key attributes are directly dependent on primary key
Cust_ID → Email_ID	All non-key attributes are directly dependent on primary key
Cust_ID → Password	All non-key attributes are directly dependent on primary key
Cust_ID → Location	All non-key attributes are directly dependent on primary key

- **BCNF:** Yes, this table is in BCNF because it is in 3NF and below are the details for each functional dependency.

<u>Functional Dependency</u>	<u>Transitive Dependency</u>
Cust_ID → Cust_Name	Every determinant in the FD is a candidate key
Cust_ID → DOB	Every determinant in the FD is a candidate key
Cust_ID → Email_ID	Every determinant in the FD is a candidate key

Cust_ID → Password	Every determinant in the FD is a candidate key
Cust_ID → Location	Every determinant in the FD is a candidate key

## 2. Customer Billing

**Table:**

Customer\_Billing(Cust\_ID, Plan\_ID, Start\_Date, Card\_Number, Card\_Type, Pay\_Date)

**FDs:**

Cust\_ID, Start\_Date → Card\_Number

Cust\_ID, Start\_Date → Card\_Type

Cust\_ID, Start\_Date → Pay\_Date

**Normalization:**

- **1NF:** Yes
- **2NF:** This table is not in 2NF because some of the functional dependencies have partial dependency. Below are the details.

<u>Functional Dependency</u>	<u>Partial Dependency</u>
Cust_ID, Start_Date → Card_Num	Non-key attribute Card_Num is not fully functionally dependent on primary key. It depends only on part of the primary key Cust_ID, Start_Date
Cust_ID, Start_Date → Card_Type	Non-key attribute Card_Type is not fully functionally dependent on primary key. It depends only on part of the primary key Cust_ID, Start_Date
Cust_ID, Start_Date → Pay_Date	Non-key attribute Pay_Date is not fully functionally dependent on primary key. It depends only on part of the primary key Cust_ID, Start_Date

To normalize the table to 2NF, we have identified the problematic attributes: Card\_Num, Card\_Type, Pay\_Date. These attributes along with the part of original primary key on which they are fully functionally dependent is formed as a new table: Customer\_Billing.

The original primary key is preserved as part of Customer\_Meal\_Plan table.

Below are the new tables generated after doing 2NF:

- 2a. Customer\_Billing ( Cust\_ID, Start\_Date, Card\_Number, Card\_Type, Pay\_Date )**
- 2b. Customer\_Meal\_Plan ( Cust\_ID, Plan\_ID, Start\_Date )**

- **3NF: Customer\_Billing:** Yes, this table is in 3NF because it is in 2NF and has no transitive dependencies. Below are the details for each functional dependency.

<u>Functional Dependency</u>	<u>Partial Dependency</u>
Cust_ID, Start_Date → Card_Num	All non-key attributes are directly dependent on primary key
Cust_ID, Start_Date → Card_Type	All non-key attributes are directly dependent on primary key
Cust_ID, Start_Date → Pay_Date	All non-key attributes are directly dependent on primary key

**Customer\_Meal\_Plan:** As all the columns are part of primary key, there are no dependencies in this table.

- **BCNF:**

**Customer\_Meal\_Plan :** As all the columns are part of primary key, there are no dependencies in this table.

**Customer\_Billing :** Yes, this table is in BCNF because it is in 3NF and below are the details for each functional dependency.

<u>Functional Dependency</u>	<u>Partial Dependency</u>
Cust_ID, Start_Date → Card_Num	Every determinant in the FD is a candidate key
Cust_ID, Start_Date → Card_Type	Every determinant in the FD is a candidate key
Cust_ID, Start_Date → Pay_Date	Every determinant in the FD is a candidate key

### 3. Meal\_Plan

**Table:** Meal\_Plan(Plan\_ID,Breakfast\_Count,Lunch\_Count,Dinner\_Count,dietician\_Option,Price)

**FDs:**

Plan ID → Breakfast\_Count  
Plan ID → Lunch\_Count  
Plan ID → Dinner\_Count  
Plan ID → dietician\_Option  
Plan ID → Price

**Normalization:**

- **1NF:** Yes, this relation is in 1NF because:
  - It has a primary key Plan\_ID
  - All the non-primary keys depend on primary key Plan\_ID
  - All the values are atomic.
  - There is no multivalued attribute
  - There is no nested relation
- **2NF:** Yes, this table is in 2NF because it is in 1NF and has no partial dependencies.  
Below are the details for each functional dependency.

<i><u>Functional Dependency</u></i>	<i><u>Partial Dependency</u></i>
Plan ID → Breakfast_Count	All non-key attributes are fully functionally dependent on primary key
Plan ID → Lunch_Count	All non-key attributes are fully functionally dependent on primary key
Plan ID → Dinner_Count	All non-key attributes are fully functionally dependent on primary key
Plan ID → dietician_Option	All non-key attributes are fully functionally dependent on primary key
Plan ID → Price	All non-key attributes are fully functionally dependent on primary key

- **3NF:** Yes, this table is in 3NF because it is in 2NF and has no transitive dependencies. Below are the details for each functional dependency.

<i><u>Functional Dependency</u></i>	<i><u>Transitive Dependency</u></i>
Plan ID → Breakfast_Count	All non-key attributes are directly dependent on primary key
Plan ID → Lunch_Count	All non-key attributes are directly dependent on primary key
Plan ID → Dinner_Count	All non-key attributes are directly dependent on primary key
Plan ID → dietician_Option	All non-key attributes are directly dependent on primary key
Plan ID → Price	All non-key attributes are directly dependent on primary key

- **BCNF:** Yes, this table is in BCNF because it is in 3NF and below are the details for each functional dependency.

<i><u>Functional Dependency</u></i>	<i><u>Transitive Dependency</u></i>
Plan ID → Breakfast_Count	Every determinant in the FD is a candidate key
Plan ID → Lunch_Count	Every determinant in the FD is a candidate key
Plan ID → Dinner_Count	Every determinant in the FD is a candidate key
Plan ID → dietician_Option	Every determinant in the FD is a candidate key
Plan ID → Price	Every determinant in the FD is a candidate key

#### 4. *Restaurant*

**Table:** Restaurant(Restaurant\_ID,Name,Email\_ID,Password,Location)

**FDs:**

Restaurant\_ID → Name

Restaurant\_ID → Email\_ID

Restaurant\_ID → Password

$\text{Restaurant\_ID} \rightarrow \text{Location}$

**Normalization:**

- **1NF:** Yes, this relation is in 1NF because:
  - It has a primary key Restaurant\_ID
  - All the non-primary keys depend on primary key Restaurant\_ID
  - All the values are atomic.
  - There is no multivalued attribute
  - There is no nested relation
- **2NF:** Yes, this table is in 2NF because it is in 1NF and has no partial dependencies.  
Below are the details for each functional dependency.

<i><b>Functional Dependency</b></i>	<i><b>Partial Dependency</b></i>
$\text{Restaurant\_ID} \rightarrow \text{Name}$	All non-key attributes are fully functionally dependent on primary key
$\text{Restaurant\_ID} \rightarrow \text{Email\_ID}$	All non-key attributes are fully functionally dependent on primary key
$\text{Restaurant\_ID} \rightarrow \text{Password}$	All non-key attributes are fully functionally dependent on primary key
$\text{Restaurant\_ID} \rightarrow \text{Location}$	All non-key attributes are fully functionally dependent on primary key

- **3NF:** Yes, this table is in 3NF because it is in 2NF and has no transitive dependencies. Below are the details for each functional dependency.

<i><b>Functional Dependency</b></i>	<i><b>Transitive Dependency</b></i>
$\text{Restaurant\_ID} \rightarrow \text{Name}$	All non-key attributes are directly dependent on primary key
$\text{Restaurant\_ID} \rightarrow \text{Email\_ID}$	All non-key attributes are directly dependent on primary key
$\text{Restaurant\_ID} \rightarrow \text{Password}$	All non-key attributes are directly dependent on primary key
$\text{Restaurant\_ID} \rightarrow \text{Location}$	All non-key attributes are directly dependent on primary key

- **BCNF:** Yes, this table is in BCNF because it is in 3NF and below are the details for each functional dependency.

<u>Functional Dependency</u>	<u>Transitive Dependency</u>
Restaurant_ID → Name	Every determinant in the FD is a candidate key
Restaurant_ID → Email_ID	Every determinant in the FD is a candidate key
Restaurant_ID → Password	Every determinant in the FD is a candidate key
Restaurant_ID → Location	Every determinant in the FD is a candidate key

## 5. Dietician

### Table:

Dietician(Dietician\_ID,Dietician\_Name,Email\_ID,Password,Experience,Qualification)

### FDs:

Dietician\_ID → Dietician\_Name  
 Dietician\_ID → Email\_ID  
 Dietician\_ID → Password  
 Dietician\_ID → Experience  
 Dietician\_ID → Qualification

### Normalization:

- **1NF:** Yes, this relation is in 1NF because:
  - It has a primary key Dietician\_ID
  - All the non-primary keys depend on primary key Dietician\_ID
  - All the values are atomic.
  - There is no multivalued attribute
  - There is no nested relation
- **2NF:** Yes, this table is in 2NF because it is in 1NF and has no partial dependencies.  
 Below are the details for each functional dependency.

<u>Functional Dependency</u>	<u>Partial Dependency</u>
Dietician_ID → Dietician_Name	All non-key attributes are fully functionally dependent on primary key
Dietician_ID → Email_ID	All non-key attributes are fully functionally dependent on primary key
Dietician_ID → Password	All non-key attributes are fully functionally dependent on primary key
Dietician_ID → Experience	All non-key attributes are fully functionally dependent on primary key
Dietician_ID → Qualification	All non-key attributes are fully functionally dependent on primary key

- **3NF:** Yes, this table is in 3NF because it is in 2NF and has no transitive dependencies. Below are the details for each functional dependency.

<u>Functional Dependency</u>	<u>Transitive Dependency</u>
Dietician_ID → Dietician_Name	All non-key attributes are directly dependent on primary key
Dietician_ID → Email_ID	All non-key attributes are directly dependent on primary key
Dietician_ID → Password	All non-key attributes are directly dependent on primary key
Dietician_ID → Experience	All non-key attributes are directly dependent on primary key
Dietician_ID → Qualification	All non-key attributes are directly dependent on primary key

- **BCNF:** Yes, this table is in BCNF because it is in 3NF and below are the details for each functional dependency.

<u>Functional Dependency</u>	<u>Transitive Dependency</u>
Dietician_ID → Dietician_Name	Every determinant in the FD is a candidate key
Dietician_ID → Email_ID	Every determinant in the FD is a candidate key
Dietician_ID → Password	Every determinant in the FD is a candidate key

Dietician_ID → Experience	Every determinant in the FD is a candidate key
Dietician_ID → Qualification	Every determinant in the FD is a candidate key

## 6. Item

**Table:** Item(Restaurant\_ID, Item\_Name, Meal\_type, Calories, Carbohydrates, Protein, Fat, Offered\_Ind)

**FDs:**

Restaurant\_ID, Item\_Name → Meal\_type  
 Restaurant\_ID, Item\_Name → Calories  
 Restaurant\_ID, Item\_Name → Carbohydrates  
 Restaurant\_ID, Item\_Name → Protein  
 Restaurant\_ID, Item\_Name → Fat  
 Restaurant\_ID, Item\_Name → Offered\_Ind

**Normalization:**

- **1NF:** Yes, this relation is in 1NF because:
  - It has a primary key Restaurant\_ID, Item\_Name.
  - All the non-primary keys depend on primary key Restaurant\_ID, Item\_Name
  - All the values are atomic.
  - There is no multivalued attribute
  - There is no nested relation
- **2NF:** Yes, this table is in 2NF because it is in 1NF and has no partial dependencies.  
 Below are the details for each functional dependency.

<u>Functional Dependency</u>	<u>Partial Dependency</u>
Restaurant_ID, Item_Name → Meal_type	All non-key attributes are fully functionally dependent on primary key
Restaurant_ID, Item_Name → Calories	All non-key attributes are fully functionally dependent on primary key
Restaurant_ID, Item_Name → Carbohydrates	All non-key attributes are fully functionally dependent on primary key

Restaurant_ID, Item_Name → Protein	All non-key attributes are fully functionally dependent on primary key
Restaurant_ID, Item_Name → Fat	All non-key attributes are fully functionally dependent on primary key
Restaurant_ID, Item_Name → Offered_Ind	All non-key attributes are fully functionally dependent on primary key

- **3NF:** Yes, this table is in 3NF because it is in 2NF and has no transitive dependencies. Below are the details for each functional dependency.

<i><b>Functional Dependency</b></i>	<i><b>Transitive Dependency</b></i>
Restaurant_ID, Item_Name → Meal_type	All non-key attributes are directly dependent on primary key
Restaurant_ID, Item_Name → Calories	All non-key attributes are directly dependent on primary key
Restaurant_ID, Item_Name → Carbohydrates	All non-key attributes are directly dependent on primary key
Restaurant_ID, Item_Name → Protein	All non-key attributes are directly dependent on primary key
Restaurant_ID, Item_Name → Fat	All non-key attributes are directly dependent on primary key
Restaurant_ID, Item_Name → Offered_Ind	All non-key attributes are directly dependent on primary key

- **BCNF:** Yes, this table is in BCNF because it is in 3NF and below are the details for each functional dependency.

<i><b>Functional Dependency</b></i>	<i><b>Transitive Dependency</b></i>
Restaurant_ID, Item_Name → Meal_type	Every determinant in the FD is a candidate key
Restaurant_ID, Item_Name → Calories	Every determinant in the FD is a candidate key
Restaurant_ID, Item_Name → Carbohydrates	Every determinant in the FD is a candidate key

Restaurant_ID, Item_Name → Protein	Every determinant in the FD is a candidate key
Restaurant_ID, Item_Name → Fat	Every determinant in the FD is a candidate key
Restaurant_ID, Item_Name → Offered_Ind	Every determinant in the FD is a candidate key

## 7. (Day Of Week) Restaurant Item Day

**Table:** Day\_Of\_Week(Restaurant\_ID,Item\_Name,Day\_Of\_Week)

**FDs:** As all the columns are part of primary key, there are no dependencies in this table.

**Normalization:**

**1NF:** Yes

**2NF:** Yes

**3NF:** Yes

**BCNF:** Yes

## 8. Diet Plan

**Table:** Diet\_Plan(Cust\_ID, Dietician\_Id, Diet\_Plan\_Id, New\_Diet\_Plan\_Ind, Activity\_level, Weight, Height, Sleep, Measurement\_Date)

**FDs:**

Cust\_ID, Dietician\_Id, Diet\_Plan\_Id → New\_Diet\_Plan\_Ind  
 Cust\_ID, Dietician\_Id, Diet\_Plan\_Id → Activity\_level  
 Cust\_ID, Dietician\_Id, Diet\_Plan\_Id → Weight  
 Cust\_ID, Dietician\_Id, Diet\_Plan\_Id → Height  
 Cust\_ID, Dietician\_Id, Diet\_Plan\_Id → Sleep  
 Cust\_ID, Dietician\_Id, Diet\_Plan\_Id → Measurement\_Date

**Normalization:**

- **1NF:** Yes, this relation is in 1NF because:
  - It has a primary key Cust\_ID, Dietician\_Id, Diet\_Plan\_Id
  - All the non-primary keys depend on primary key Cust\_ID, Dietician\_Id, Diet\_Plan\_Id
  - All the values are atomic.

- There is no multivalued attribute
- There is no nested relation
- **2NF:** Yes, this table is in 2NF because it is in 1NF and has no partial dependencies.  
Below are the details for each functional dependency.

<u>Functional Dependency</u>	<u>Partial Dependency</u>
Cust_ID, Dietician_Id, Diet_Plan_Id → New_Diet_Plan_Ind	All non-key attributes are fully functionally dependent on primary key
Cust_ID, Dietician_Id, Diet_Plan_Id → Activity_level	All non-key attributes are fully functionally dependent on primary key
Cust_ID, Dietician_Id, Diet_Plan_Id → Weight	All non-key attributes are fully functionally dependent on primary key
Cust_ID, Dietician_Id, Diet_Plan_Id → Height	All non-key attributes are fully functionally dependent on primary key
Cust_ID, Dietician_Id, Diet_Plan_Id → Sleep	All non-key attributes are fully functionally dependent on primary key
Cust_ID, Dietician_Id, Diet_Plan_Id → Measurement_Date	All non-key attributes are fully functionally dependent on primary key

- **3NF:** Yes, this table is in 3NF because it is in 2NF and has no transitive dependencies. Below are the details for each functional dependency.

<u>Functional Dependency</u>	<u>Transitive Dependency</u>
Cust_ID, Dietician_Id, Diet_Plan_Id → New_Diet_Plan_Ind	All non-key attributes are directly dependent on primary key
Cust_ID, Dietician_Id, Diet_Plan_Id → Activity_level	All non-key attributes are directly dependent on primary key
Cust_ID, Dietician_Id, Diet_Plan_Id → Weight	All non-key attributes are directly dependent on primary key
Cust_ID, Dietician_Id, Diet_Plan_Id → Height	All non-key attributes are directly dependent on primary key

Cust_ID, Dietician_Id, Diet_Plan_Id → Sleep	All non-key attributes are directly dependent on primary key
Cust_ID, Dietician_Id, Diet_Plan_Id → Measurement_Date	All non-key attributes are directly dependent on primary key

- **BCNF:** Yes, this table is in BCNF because it is in 3NF and below are the details for each functional dependency.

<u>Functional Dependency</u>	<u>Transitive Dependency</u>
Cust_ID, Dietician_Id, Diet_Plan_Id → New_Diet_Plan_Ind	Every determinant in the FD is a candidate key
Cust_ID, Dietician_Id, Diet_Plan_Id → Activity_level	Every determinant in the FD is a candidate key
Cust_ID, Dietician_Id, Diet_Plan_Id → Weight	Every determinant in the FD is a candidate key
Cust_ID, Dietician_Id, Diet_Plan_Id → Height	Every determinant in the FD is a candidate key
Cust_ID, Dietician_Id, Diet_Plan_Id → Sleep	Every determinant in the FD is a candidate key
Cust_ID, Dietician_Id, Diet_Plan_Id → Measurement_Date	Every determinant in the FD is a candidate key

## 9. NMD (Diet Plan Details)

**Table:**

Diet\_Plan\_Details(Cust\_ID,Dietician\_Id,Diet\_Plan\_Id,Day,Meal\_Type,Calories,Carbohydrates,Protein,Fat)

**FDs:**

Cust\_ID, Dietician\_ID, Diet\_Plan\_ID, Day, Meal\_Type → Calories

Cust\_ID, Dietician\_ID, Diet\_Plan\_ID, Day, Meal\_Type → Carbohydrates

Cust\_ID, Dietician\_ID, Diet\_Plan\_ID, Day, Meal\_Type → Protein

Cust\_ID, Dietician\_ID, Diet\_Plan\_ID, Day, Meal\_Type → Fat

**Normalization:**

- **1NF:** Yes, this relation is in 1NF because:
  - It has a primary key Cust\_ID,Dietician\_Id,Diet\_Plan\_Id,Day,Meal\_Type
  - All the non-primary keys depend on primary key  
Cust\_ID,Dietician\_Id,Diet\_Plan\_Id,Day,Meal\_Type
  - All the values are atomic.
  - There is no multivalued attribute
  - There is no nested relation
- **2NF:** Yes, this table is in 2NF because it is in 1NF and has no partial dependencies.  
Below are the details for each functional dependency.

<i><u>Functional Dependency</u></i>	<i><u>Partial Dependency</u></i>
Cust_ID, Dietician_ID, Diet_Plan_ID, Day, Meal_Type → Calories	All non-key attributes are fully functionally dependent on primary key
Cust_ID, Dietician_ID, Diet_Plan_ID, Day, Meal_Type → Carbohydrates	All non-key attributes are fully functionally dependent on primary key
Cust_ID, Dietician_ID, Diet_Plan_ID, Day, Meal_Type → Protein	All non-key attributes are fully functionally dependent on primary key
Cust_ID, Dietician_ID, Diet_Plan_ID, Day, Meal_Type → Fat	All non-key attributes are fully functionally dependent on primary key

- **3NF:** Yes, this table is in 3NF because it is in 2NF and has no transitive dependencies. Below are the details for each functional dependency.

<i><u>Functional Dependency</u></i>	<i><u>Transitive Dependency</u></i>
Cust_ID, Dietician_ID, Diet_Plan_ID, Day, Meal_Type → Calories	All non-key attributes are directly dependent on primary key
Cust_ID, Dietician_ID, Diet_Plan_ID, Day, Meal_Type → Carbohydrates	All non-key attributes are directly dependent on primary key
Cust_ID, Dietician_ID, Diet_Plan_ID, Day, Meal_Type → Protein	All non-key attributes are directly dependent on primary key

Cust_ID, Dietician_ID, Diet_Plan_ID, Day, Meal_Type → Fat	All non-key attributes are directly dependent on primary key
--	--

- **BCNF:** Yes, this table is in BCNF because it is in 3NF and below are the details for each functional dependency.

<i>Functional Dependency</i>	<i>Transitive Dependency</i>
Cust_ID, Dietician_ID, Diet_Plan_ID, Day, Meal_Type → Calories	Every determinant in the FD is a candidate key
Cust_ID, Dietician_ID, Diet_Plan_ID, Day, Meal_Type → Carbohydrates	Every determinant in the FD is a candidate key
Cust_ID, Dietician_ID, Diet_Plan_ID, Day, Meal_Type → Protein	Every determinant in the FD is a candidate key
Cust_ID, Dietician_ID, Diet_Plan_ID, Day, Meal_Type → Fat	Every determinant in the FD is a candidate key

## **10. Allergies**

**Table:** Allergies(Cust\_ID,Dietician\_Id,Diet\_Plan\_Id,Allergy)

**FDs:** As all the columns are part of primary key, there are no dependencies in this table.

**Normalization:**

**1NF:** Yes

**2NF:** Yes

**3NF:** Yes

**BCNF:** Yes

## **11. Disease**

**Table:**Disease(Cust\_ID,Dietician\_Id,Diet\_Plan\_Id,Disease)

**FDs:** As all the columns are part of primary key, there are no dependencies in this table.

**Normalization:****1NF:** Yes**2NF:** Yes**3NF:** Yes**BCNF:** Yes**12. Diet Progress****Table:**

Diet\_Progress(Cust\_ID, Dietician\_Id, Diet\_Plan\_Id, Day, Breakfast, Lunch, Dinner, Additional\_Intake)

**FDs:**

- Cust\_ID, Dietician\_ID, Diet\_Plan\_ID, Day → Breakfast  
Cust\_ID, Dietician\_ID, Diet\_Plan\_ID, Day → Lunch  
Cust\_ID, Dietician\_ID, Diet\_Plan\_ID, Day → Dinner  
Cust\_ID, Dietician\_ID, Diet\_Plan\_ID, Day → Additional\_Intake

**Normalization:**

- **1NF:** Yes, this relation is in 1NF because:
  - It has a primary key Cust\_ID, Dietician\_Id, Diet\_Plan\_Id, Day
  - All the non-primary keys depend on primary key Cust\_ID, Dietician\_Id, Diet\_Plan\_Id, Day
  - All the values are atomic.
  - There is no multivalued attribute
  - There is no nested relation
- **2NF:** Yes, this table is in 2NF because it is in 1NF and has no partial dependencies.  
Below are the details for each functional dependency.

<u>Functional Dependency</u>	<u>Partial Dependency</u>
Cust_ID, Dietician_ID, Diet_Plan_ID, Day → Breakfast	All non-key attributes are fully functionally dependent on primary key
Cust_ID, Dietician_ID, Diet_Plan_ID, Day → Lunch	All non-key attributes are fully functionally dependent on primary key

Cust_ID, Dietician_ID, Diet_Plan_ID, Day → Dinner	All non-key attributes are fully functionally dependent on primary key
Cust_ID, Dietician_ID, Diet_Plan_ID, Day → Additional_Intake	All non-key attributes are fully functionally dependent on primary key

- **3NF:** Yes, this table is in 3NF because it is in 2NF and has no transitive dependencies. Below are the details for each functional dependency.

<u>Functional Dependency</u>	<u>Transitive Dependency</u>
Cust_ID, Dietician_ID, Diet_Plan_ID, Day → Breakfast	All non-key attributes are directly dependent on primary key
Cust_ID, Dietician_ID, Diet_Plan_ID, Day → Lunch	All non-key attributes are directly dependent on primary key
Cust_ID, Dietician_ID, Diet_Plan_ID, Day → Dinner	All non-key attributes are directly dependent on primary key
Cust_ID, Dietician_ID, Diet_Plan_ID, Day → Additional_Intake	All non-key attributes are directly dependent on primary key

- **BCNF:** Yes, this table is in BCNF because it is in 3NF and below are the details for each functional dependency.

<u>Functional Dependency</u>	<u>Transitive Dependency</u>
Cust_ID, Dietician_ID, Diet_Plan_ID, Day → Breakfast	Every determinant in the FD is a candidate key
Cust_ID, Dietician_ID, Diet_Plan_ID, Day → Lunch	Every determinant in the FD is a candidate key
Cust_ID, Dietician_ID, Diet_Plan_ID, Day → Dinner	Every determinant in the FD is a candidate key
Cust_ID, Dietician_ID, Diet_Plan_ID, Day → Additional_Intake	Every determinant in the FD is a candidate key

### **13. Customer Requests**

#### **Table:**

Customer\_Requests(Cust\_ID,Dietician\_Id,Diet\_Plan\_Id,Request\_Initiation\_Date,Request\_Content,Update\_Diet\_Plan\_Ind)

#### **FDs:**

$\text{Cust\_ID, Dietician\_ID, Diet\_Plan\_ID} \rightarrow \text{Request\_Initiation\_Date}$   
 $\text{Cust\_ID, Dietician\_ID, Diet\_Plan\_ID} \rightarrow \text{Request\_Content}$   
 $\text{Cust\_ID, Dietician\_ID, Diet\_Plan\_ID} \rightarrow \text{Update\_Diet\_Plan\_Ind}$

#### **Normalization:**

- **1NF:** Yes, this relation is in 1NF because:
  - It has a primary key Cust\_ID,Dietician\_Id,Diet\_Plan\_Id
  - All the non-primary keys depend on primary key  
Cust\_ID,Dietician\_Id,Diet\_Plan\_Id
  - All the values are atomic.
  - There is no multivalued attribute
  - There is no nested relation
- **2NF:** Yes, this table is in 2NF because it is in 1NF and has no partial dependencies.  
Below are the details for each functional dependency.

<u>Functional Dependency</u>	<u>Partial Dependency</u>
$\text{Cust\_ID, Dietician\_ID, Diet\_Plan\_ID} \rightarrow \text{Request\_Initiation\_Date}$	All non-key attributes are fully functionally dependent on primary key
$\text{Cust\_ID, Dietician\_ID, Diet\_Plan\_ID} \rightarrow \text{Request\_Content}$	All non-key attributes are fully functionally dependent on primary key
$\text{Cust\_ID, Dietician\_ID, Diet\_Plan\_ID} \rightarrow \text{Update\_Diet\_Plan\_Ind}$	All non-key attributes are fully functionally dependent on primary key

- **3NF:** Yes, this table is in 3NF because it is in 2NF and has no transitive dependencies. Below are the details for each functional dependency.

<b><i>Functional Dependency</i></b>	<b><i>Transitive Dependency</i></b>
Cust_ID, Dietician_ID, Diet_Plan_ID → Request_Initiation_Date	All non-key attributes are directly dependent on primary key
Cust_ID, Dietician_ID, Diet_Plan_ID → Request_Content	All non-key attributes are directly dependent on primary key
Cust_ID, Dietician_ID, Diet_Plan_ID → Update_Diet_Plan_Ind	All non-key attributes are directly dependent on primary key

- **BCNF:** Yes, this table is in BCNF because it is in 3NF and below are the details for each functional dependency.

<b><i>Functional Dependency</i></b>	<b><i>Transitive Dependency</i></b>
Cust_ID, Dietician_ID, Diet_Plan_ID → Request_Initiation_Date	Every determinant in the FD is a candidate key
Cust_ID, Dietician_ID, Diet_Plan_ID → Request_Content	Every determinant in the FD is a candidate key
Cust_ID, Dietician_ID, Diet_Plan_ID → Update_Diet_Plan_Ind	Every determinant in the FD is a candidate key

#### **14. Orders**

**Table:** Orders(Cust\_ID, Restaurant\_ID, Item\_Name, Order\_Date, Pickup\_Time)

**FDs:**

Cust\_ID, Restaurant\_ID, Item\_Name, Order\_Date → Pickup\_Time

**Normalization:**

- **1NF:** Yes, this relation is in 1NF because:

- It has a primary key `Cust_ID,Restaurant_ID,Item_Name,Order_Date`
  - All the non-primary keys depend on primary key  
`Cust_ID,Restaurant_ID,Item_Name,Order_Date`
  - All the values are atomic.
  - There is no multivalued attribute
  - There is no nested relation
- **2NF:** Yes, this table is in 2NF because it is in 1NF and has no partial dependencies.  
Below are the details for each functional dependency.

<u>Functional Dependency</u>	<u>Partial Dependency</u>
<code>Cust_ID, Restaurant_ID, Item_Name, Order_Date → Pickup_Time</code>	All non-key attributes are fully functionally dependent on primary key

- **3NF:** Yes, this table is in 3NF because it is in 2NF and has no transitive dependencies. Below are the details for each functional dependency.

<u>Functional Dependency</u>	<u>Transitive Dependency</u>
<code>Cust_ID, Restaurant_ID, Item_Name, Order_Date → Pickup_Time</code>	All non-key attributes are directly dependent on primary key

- **BCNF:** Yes, this table is in BCNF because it is in 3NF and below are the details for each functional dependency.

<u>Functional Dependency</u>	<u>Transitive Dependency</u>
<code>Cust_ID, Restaurant_ID, Item_Name, Order_Date → Pickup_Time</code>	Every determinant in the FD is a candidate key

## 15. Item Ratings

**Table:** Item\_Ratings(`Cust_ID,Restaurant_ID,Item_Name,Rating`)

**FDs:**

Cust\_ID, Restaurant\_ID, Item\_Name → Rating

**Normalization:**

- **1NF:** Yes, this relation is in 1NF because:
  - It has a primary key Cust\_ID, Restaurant\_ID, Item\_Name
  - All the non-primary keys depend on primary key Cust\_ID, Restaurant\_ID, Item\_Name.
  - All the values are atomic.
  - There is no multivalued attribute
  - There is no nested relation
- **2NF:** Yes, this table is in 2NF because it is in 1NF and has no partial dependencies.  
Below are the details for each functional dependency.

<u>Functional Dependency</u>	<u>Partial Dependency</u>
Cust_ID, Restaurant_ID, Item_Name → Rating	All non-key attributes are fully functionally dependent on primary key

- **3NF:** Yes, this table is in 3NF because it is in 2NF and has no transitive dependencies. Below are the details for each functional dependency.

<u>Functional Dependency</u>	<u>Transitive Dependency</u>
Cust_ID, Restaurant_ID, Item_Name → Rating	All non-key attributes are directly dependent on primary key

- **BCNF:** Yes, this table is in BCNF because it is in 3NF and below are the details for each functional dependency.

<u>Functional Dependency</u>	<u>Transitive Dependency</u>
Cust_ID, Restaurant_ID, Item_Name → Rating	Every determinant in the FD is a candidate key

## 16. Rates Dietician

**Table:** Dietician\_Ratings(Cust\_ID,Dietician\_ID,Rating)

**FDs:**

$\text{Cust\_ID}, \text{Dietician\_ID} \rightarrow \text{Rating}$

**Normalization:**

- **1NF:** Yes, this relation is in 1NF because:
  - It has a primary key  $\text{Cust\_ID}, \text{Dietician\_ID}$
  - All the non-primary keys depend on primary key  $\text{Cust\_ID}, \text{Dietician\_ID}$
  - All the values are atomic.
  - There is no multivalued attribute
  - There is no nested relation
- **2NF:** Yes, this table is in 2NF because it is in 1NF and has no partial dependencies.  
Below are the details for each functional dependency.

<u>Functional Dependency</u>	<u>Partial Dependency</u>
$\text{Cust\_ID}, \text{Dietician\_ID} \rightarrow \text{Rating}$	All non-key attributes are fully functionally dependent on primary key

- **3NF:** Yes, this table is in 3NF because it is in 2NF and has no transitive dependencies.  
Below are the details for each functional dependency.

<u>Functional Dependency</u>	<u>Transitive Dependency</u>
$\text{Cust\_ID}, \text{Dietician\_ID} \rightarrow \text{Rating}$	All non-key attributes are directly dependent on primary key

- **BCNF:** Yes, this table is in BCNF because it is in 3NF and below are the details for each functional dependency.

<u>Functional Dependency</u>	<u>Transitive Dependency</u>
$\text{Cust\_ID}, \text{Dietician\_ID} \rightarrow \text{Rating}$	Every determinant in the FD is a candidate key

## 6.4. Multi statement DB Transactions (Initiated from DB Server Side)

### 6.4.1. Admin

#### 1. proc\_d\_ins\_dietn\_regn\_details

```
7 • -- PROCEDURE TO INSERT DIETICIAN REGISTRATION DETAILS
8
9 drop procedure if exists proc_d_ins_dietn_regn_details;
10 DELIMITER $$

11 • create procedure proc_d_ins_dietn_regn_details
12   (in dietn_name varchar(50),
13    in dietn_email varchar(50),
14    in dietn_pwd varchar(50),
15    in dietn_exp decimal(5,2),
16    in dietn_qua varchar(50),
17    out vc int,
18    out eid varchar(50))
19 begin
20   DECLARE `_rollback` BOOL DEFAULT 0;
21   DECLARE CONTINUE HANDLER FOR SQLEXCEPTION SET `_rollback` = 1;
22
23   START TRANSACTION;
24   set eid = NULL;
25   set vc = 0;
26   select email_id into eid from dietician where email_id = dietn_email;
27   if (eid is NULL) then
28     insert into dietician (dietician_name, email_id, password, experience, qualification) values (dietn_name, dietn_email, SHA1(dietn_pwd), dietn_exp, dietn_qua);
29     set vc = 1;
30   end if;
31
32   IF `_rollback` THEN ROLLBACK;
33   ELSE COMMIT;
34   END IF;
35 end
36 $$
```

## 2. proc\_r\_ins\_rest\_details

```
50 • -- PROCEDURE TO INSERT RESTAURANT REGISTRATION DETAILS
51
52 drop procedure if exists proc_r_ins_rest_details;
53 DELIMITER $$

54 • create procedure proc_r_ins_rest_details
55   (in rest_name varchar(50),
56   in rest_email varchar(50),
57   in rest_pwd varchar(50),
58   in rest_location varchar(30),
59   out vc int,
60   out eid varchar(50))
61 begin
62   DECLARE `_rollback` BOOL DEFAULT 0;
63   DECLARE CONTINUE HANDLER FOR SQLEXCEPTION SET `_rollback` = 1;
64   START TRANSACTION;

65
66   set eid = NULL;
67   set vc = 0;
68   select email_id into eid from restaurant where email_id = rest_email;
69   if (eid is NULL) then
70     insert into restaurant (name, email_id, password, location) values (rest_name, rest_email, SHA1(rest_pwd), rest_location);
71     set vc = 1;
72   end if;
73
74   IF `_rollback` THEN ROLLBACK;
75   ELSE COMMIT;
76 END IF;
77 end
78 $$

79 DELIMITER ;
80
```

### 3. proc\_c\_ins\_cust\_details

```
1 -- PROCEDURE TO INSERT CUSTOMER REGISTRATION DETAILS
2 • drop procedure if exists proc_c_ins_cust_details;
3 DELIMITER $$ 
4 • create procedure proc_c_ins_cust_details
5 (in cust_name varchar(50),
6 in cust_dob date,
7 in cust_email_id varchar(50),
8 in cust_password varchar(50),
9 in cust_location varchar(50),
10 out vc int,
11 out eid varchar(50))
12 begin
13     DECLARE `_rollback` BOOL DEFAULT 0;
14     DECLARE CONTINUE HANDLER FOR SQLEXCEPTION SET `_rollback` = 1;
15     START TRANSACTION;
16
17     set eid = NULL;
18     set vc = 0;
19     select email_id into eid from customer where email_id = cust_email_id;
20     if (eid is null) then
21         insert into customer (cust_name, dob, email_id, password, location) values (cust_name, cust_dob, cust_email_id, SHA1(cust_password),
22         cust_location);
23         set vc = 1;
24     end if;
25
26     IF `_rollback` THEN ROLLBACK;
27     ELSE COMMIT;
28 END IF;
29 end
30 $$ 
31 DELIMITER ;
```

#### 6.4.2. Customer

##### 1. proc\_dietician\_yes

```
33 |
34 • -- Dietician_YES
35
36 drop procedure if exists proc_dietician_yes;
37 DELIMITER $$;
38 • ⊞ create procedure proc_dietician_yes(
39     in v_customer_id int,
40     in stdt date,
41     in cardtype varchar(10),
42     in cardnumb bigint(20),
43     in dieticianid int(11),
44     in measr_date date,
45     in activitylevel varchar(10),
46     in weight decimal(6,2),
47     in height decimal(5,2),
48     in sleep int(11),
49     in disease1 varchar(25),
50     in disease2 varchar(25),
51     in disease3 varchar(25),
52     in disease4 varchar(25),
53     in disease5 varchar(25),
54     in allergy1 varchar(30),
55     in allergy2 varchar(30),
56     in allergy3 varchar(30),
57     in allergy4 varchar(30),
58     in allergy5 varchar(30),
59     out res int,
60     out cnt int)
61 ⊞ begin
62     DECLARE `_rollback` BOOL DEFAULT 0;
63     DECLARE CONTINUE HANDLER FOR SQLEXCEPTION SET `_rollback` = 1;
64     START TRANSACTION;
65
66     set res = 0;
67     set cnt = 0;
```

```

63  DECLARE CONTINUE HANDLER FOR SQLEXCEPTION SET `__rollback` = 1;
64  START TRANSACTION;
65
66  set res = 0;
67  set cnt= 0;
68
69  insert into customer_meal_plan(cust_id, start_date, plan_id)
70  values (v_customer_id, stdt, '1');
71
72  insert into customer_billing (cust_ID, start_date, card_number, card_type, pay_date)
73  values(v_customer_id, stdt, cardnumb, cardtype, curdate());
74
75  select count(cust_ID) into cnt from diet_plan
76  where cust_ID = v_customer_id;
77  set cnt = cnt + 1;
78
79  insert into diet_plan
80  values(v_customer_id, dieticianid, cnt, stdt, 'Y', activitylevel, weight, sleep, height);
81
82  if (disease1 != 'undefined') then
83    insert into disease values(v_customer_id, dieticianid, cnt, disease1);
84  end if;
85  if (disease2 != 'undefined') then
86    insert into disease values(v_customer_id, dieticianid, cnt, disease2);
87  end if;
88  if (disease3 != 'undefined') then
89    insert into disease values(v_customer_id, dieticianid, cnt, disease3);
90  end if;
91  if (disease4 != 'undefined') then
92    insert into disease values(v_customer_id, dieticianid, cnt, disease4);
93  end if;
94  if (disease5 != 'undefined') then
95    insert into disease values(v_customer_id, dieticianid, cnt, disease5);
96  end if;
97  if (allergy1 != 'undefined') then
98    insert into allergy values(v_customer_id, dieticianid, cnt, allergy1);
99  end if;
100 if (allergy2 != 'undefined') then

```

```

09      end if;
10      if (allergy2 != 'undefined') then
11          insert into allergy values(v_customer_id, dieticianid, cnt, allergy2);
12      end if;
13      if (allergy3 != 'undefined') then
14          insert into allergy values(v_customer_id, dieticianid, cnt, allergy3);
15      end if;
16      if (allergy4 != 'undefined') then
17          insert into allergy values(v_customer_id, dieticianid, cnt, allergy4);
18      end if;
19      if (allergy5 != 'undefined') then
20          insert into allergy values(v_customer_id, dieticianid, cnt, allergy5);
21      end if;
22      set res =1;
23
24      IF `_rollback` THEN ROLLBACK;
25      ELSE COMMIT;
26      END IF;
27  end
28  $$;
29  DELIMITER ;

```

## 2. proc\_dietician\_no

```
22
23 • -- Dietician_NO
24 drop procedure if exists proc_dietician_no;
25 DELIMITER $$

26 • - create procedure proc_dietician_no(
27     in v_customer_id int,
28     in stdt DATE,
29     in mealytypeid int(11),
30     in cardtype varchar(10),
31     in cardnumb bigint(20),
32     out res int
33 )
34 begin
35     DECLARE `_rollback` BOOL DEFAULT 0;
36     DECLARE CONTINUE HANDLER FOR SQLEXCEPTION SET `_rollback` = 1;
37     START TRANSACTION;

38
39     set res = 0;
40     insert into customer_meal_plan
41         values (v_customer_id, stdt, mealytypeid);
42     insert into customer_billing
43         values(v_customer_id, stdt, cardnumb, cardtype, curdate());
44     set res = 1;

45
46     IF `_rollback` THEN ROLLBACK;
47     ELSE COMMIT;
48     END IF;
49 end
50 $$

51 DELIMITER ;
52
```

### 3. c\_place\_order

MySQL Workbench

SQL File 11\*

```
1 DELIMITER $$  
2 • CREATE PROCEDURE c_place_order  
3   (IN p_cust_id int, IN restaurant_name_bf varchar(30), IN item_name_bf varchar(30), IN restaurant_name_ln varchar(30),  
4    IN item_name_ln varchar(30), IN restaurant_name_din varchar(30), IN item_name_din varchar(30),  
5    IN p_bf_time time, IN p_ln_time time, IN p_din_time time, OUT bf_chk int, OUT ln_chk int, OUT din_chk int )  
6 BEGIN  
7   DECLARE restaurant_id_bf int DEFAULT 0;  
8   DECLARE restaurant_id_ln int DEFAULT 0;  
9   DECLARE restaurant_id_din int DEFAULT 0;  
10  DECLARE `rollback` BOOL DEFAULT 0;  
11  DECLARE CONTINUE HANDLER FOR SQLEXCEPTION SET `rollback` = 1;  
12  START TRANSACTION;  
13  
14  SELECT COUNT(o.cust_id) into bf_chk  
15  FROM orders as o  
16  JOIN item as i  
17  ON i.restaurant_id = o.restaurant_id AND i.item_name = o.item_name  
18  WHERE cust_id = p_cust_id AND order_date = DATE(NOW()) AND meal_type = 'breakfast' AND restaurant_name_bf != '';  
19  SELECT COUNT(o.cust_id) into ln_chk  
20  FROM orders as o  
21  JOIN item as i  
22  ON i.restaurant_id = o.restaurant_id AND i.item_name = o.item_name  
23  WHERE cust_id = p_cust_id AND order_date = DATE(NOW()) AND meal_type = 'lunch' AND restaurant_name_ln != '';  
24  SELECT COUNT(o.cust_id) into din_chk  
25  FROM orders as o  
26  JOIN item as i  
27  ON i.restaurant_id = o.restaurant_id AND i.item_name = o.item_name  
28  WHERE cust_id = p_cust_id AND order_date = DATE(NOW()) AND meal_type = 'dinner' AND restaurant_name_din != '';  
29  SELECT restaurant_id into restaurant_id_bf  
30  FROM restaurant  
31  WHERE name = restaurant_name_bf;  
32  SELECT restaurant_id into restaurant_id_ln  
33  FROM restaurant  
34  WHERE name = restaurant_name_ln;  
35  SELECT restaurant_id into restaurant_id_din  
36  FROM restaurant  
37  WHERE name = restaurant_name_din;  
38  
39  IF restaurant_id_bf != 0 AND bf_chk = 0 THEN  
40    INSERT INTO orders values(p_cust_id,restaurant_id_bf,item_name_bf,NOW(),p_bf_time);  
41  END IF;  
42  IF restaurant_id_ln != 0 AND ln_chk = 0 THEN  
43    INSERT INTO orders values(p_cust_id,restaurant_id_ln,item_name_ln,NOW(),p_ln_time);  
44  END IF;  
45  IF restaurant_id_din != 0 AND din_chk = 0 THEN  
46    INSERT INTO orders values(p_cust_id,restaurant_id_din,item_name_din,NOW(),p_din_time);  
47  END IF;  
48  IF `rollback` THEN ROLLBACK;  
49  ELSE COMMIT;  
50  END IF;  
51  END;  
52  $$  
53  DELIMITER ;
```

#### 4. c\_request\_diet\_plan\_change

```
297  DELIMITER $$  
298  • CREATE PROCEDURE c_request_diet_plan_change(IN p_cust_id int, IN input_date date, IN p_request varchar(50),OUT v_check int)  
299  BEGIN  
300  DECLARE v_cust_id int;  
301  DECLARE v_dietician_id int;  
302  DECLARE v_diet_plan_id int;  
303  DECLARE `_rollback` BOOL DEFAULT 0;  
304  DECLARE CONTINUE HANDLER FOR SQLEXCEPTION SET `_rollback` = 1;  
305  START TRANSACTION;  
306  SELECT DISTINCT cust_id, dietician_id, diet_plan_id into v_cust_id,v_dietician_id,v_diet_plan_id  
307  FROM c_diet_plan_details  
308  WHERE cust_id = p_cust_id;  
309  -- To check if there is an existing request change  
310  SELECT COUNT(*) into v_check  
311  FROM customer_requests  
312  WHERE cust_id= v_cust_id AND dietician_id = v_dietician_id AND diet_plan_id = v_diet_plan_id;  
313  IF v_check != 1 THEN  
314      INSERT INTO customer_requests values(v_cust_id,v_dietician_id,v_diet_plan_id,input_date,p_request,'Y');  
315  ELSE  
316      UPDATE customer_requests  
317      SET request_initiation_date = input_date,request_content = p_request,update_diet_plan_ind='Y'  
318      WHERE cust_id= v_cust_id AND dietician_id = v_dietician_id AND diet_plan_id = v_diet_plan_id;  
319  END IF;  
320  IF `_rollback` THEN ROLLBACK;  
321  ELSE COMMIT;  
322  END IF;  
323  END;  
324  $$  
325  DELIMITER ;  
326
```

## 5. c\_update\_diet\_progress

```
331
332  DELIMITER $$ 
333 • ⊕ CREATE PROCEDURE c_update_diet_progress (IN p_cust_id int, IN log_date date, IN breakfast varchar(10), IN lunch varchar(10),
334   IN dinner varchar(10), IN additional_intake varchar(50), OUT v_cnt int)
335 ⊖ BEGIN
336   DECLARE v_day int;
337   DECLARE v_cust_id int;
338   DECLARE v_dietician_id int;
339   DECLARE v_diet_plan_id int;
340   DECLARE v_start_date date;
341   DECLARE `_rollback` BOOL DEFAULT 0;
342   DECLARE CONTINUE HANDLER FOR SQLEXCEPTION SET `_rollback` = 1;
343   START TRANSACTION;
344   SELECT DISTINCT cust_id, dietician_id, diet_plan_id, diet_start_date INTO v_cust_id, v_dietician_id, v_diet_plan_id, v_start_date
345   FROM c_diet_plan_details
346   WHERE cust_id = p_cust_id;
347   SELECT DATEDIFF(log_date, v_start_date)+1 INTO v_day;
348   -- To check if the progress is already logged for tht day
349   SELECT COUNT(*) INTO v_cnt
350   FROM diet_progress
351   WHERE cust_id= v_cust_id AND dietician_id = v_dietician_id AND diet_plan_id = v_diet_plan_id AND day = v_day;
352   ⊖ If v_cnt != 1 THEN
353     INSERT INTO diet_progress values(v_cust_id, v_dietician_id, v_diet_plan_id, v_day, breakfast, lunch, dinner, additional_intake);
354   ELSE
355     UPDATE diet_progress
356     SET breakfast = breakfast, lunch = lunch, dinner = dinner, additional_intake = additional_intake
357     WHERE cust_id= v_cust_id AND dietician_id = v_dietician_id AND diet_plan_id = v_diet_plan_id AND day = v_day;
358   END IF;
359   ⊖ IF `_rollback` THEN ROLLBACK;
360   ELSE COMMIT;
361   END IF;
362   END;
363   $$ 
364  DELIMITER ;
```

## 6. c\_update\_item\_rating

```
375
376      DELIMITER $$ 
377  •  CREATE PROCEDURE c_update_item_rating
378      (IN p_cust_id int, IN p_restaurant_id int, IN p_item_name varchar(30), IN p_rating int, OUT v_chck int )
379  • BEGIN
380      DECLARE `_rollback` BOOL DEFAULT 0;
381      DECLARE CONTINUE HANDLER FOR SQLEXCEPTION SET `_rollback` = 1;
382      START TRANSACTION;
383      SELECT COUNT(*) into v_chck
384      FROM item_ratings
385      WHERE cust_id = p_cust_id AND restaurant_id = p_restaurant_id AND item_name= p_item_name;
386  • If v_chck != 1 THEN
387          INSERT INTO item_ratings values(p_cust_id,p_restaurant_id,p_item_name,p_rating);
388      ELSE
389          UPDATE item_ratings
390          SET rating = p_rating
391          WHERE cust_id = p_cust_id AND restaurant_id = p_restaurant_id AND item_name= p_item_name;
392      END IF;
393  • IF `_rollback` THEN ROLLBACK;
394      ELSE COMMIT;
395  • END IF;
396      END;
397      $$ 
398      DELIMITER ;
```

## 7. c\_update\_dietician\_rating

```
404
405  DELIMITER $$ 
406 • CREATE PROCEDURE c_update_dietician_rating (IN p_cust_id int, IN p_dietician_id int, IN p_rating int, OUT v_chck int)
407  BEGIN
408      DECLARE `_rollback` BOOL DEFAULT 0;
409      DECLARE CONTINUE HANDLER FOR SQLEXCEPTION SET `_rollback` = 1;
410      START TRANSACTION;
411      SELECT COUNT(*) into v_chck
412      FROM dietician_ratings
413      WHERE cust_id = p_cust_id AND dietician_id = p_dietician_id;
414      If v_chck != 1 THEN
415          INSERT INTO dietician_ratings values(p_cust_id,p_dietician_id,p_rating);
416      ELSE
417          UPDATE dietician_ratings
418          SET rating = p_rating
419          WHERE cust_id = p_cust_id AND dietician_id = p_dietician_id;
420      END IF;
421      IF `_rollback` THEN ROLLBACK;
422      ELSE COMMIT;
423      END IF;
424  END;
425  $$ 
426  DELIMITER ;
```

### 6.4.3. Dietician

#### 1. d\_add\_diet\_details

```
1  DELIMITER $$  
2  • □ CREATE PROCEDURE d_add_diet_details(  
3      IN  v_custid_dietstartdate VARCHAR(50),  
4      IN  v_day                  VARCHAR(10),  
5      IN  v_breakfast_calories   VARCHAR(10),  
6      IN  v_breakfast_proteins  VARCHAR(10),  
7      IN  v_breakfast_carbohydrates  VARCHAR(10),  
8      IN  v_breakfast_fat        VARCHAR(10),  
9      IN  v_lunch_calories      VARCHAR(10),  
10     IN  v_lunch_proteins      VARCHAR(10),  
11     IN  v_lunch_carbohydrates  VARCHAR(10),  
12     IN  v_lunch_fat           VARCHAR(10),  
13     IN  v_dinner_calories     VARCHAR(10),  
14     IN  v_dinner_proteins     VARCHAR(10),  
15     IN  v_dinner_carbohydrates  VARCHAR(10),  
16     IN  v_dinner_fat          VARCHAR(10),  
17     OUT v_result             INT)  
18  □ BEGIN  
19    DECLARE v_count_breakfast INT DEFAULT 0;  
20    DECLARE v_count_lunch     INT DEFAULT 0;  
21    DECLARE v_count_dinner    INT DEFAULT 0;  
22    declare v_cust_id INT default 0;  
23    declare v_diet_start_date date default now();  
24    declare v_dietician_id int default 0;  
25    declare v_diet_Plan_id int default 0;  
26    DECLARE `_rollback` BOOL DEFAULT 0;  
27    DECLARE CONTINUE HANDLER FOR SQLEXCEPTION SET `_rollback` = 1;  
28    START TRANSACTION;  
29  
30    select cust_id,diet_start_date,dietician_id,diet_Plan_id into v_cust_id,v_diet_start_date,v_dietician_id,v_diet_Plan_id  
31    from diet_plan where concat(cust_id,'-',diet_start_date)=v_custid_dietstartdate;  
32  
33    SELECT count(1)  
34    INTO v_count_breakfast  
35    FROM diet_plan_details  
36    WHERE  cust_id = v_cust_id  
37    and dietician_id = v_dietician_id  
38    and diet_Plan_id = v_diet_Plan_id  
39    AND day = v_day  
40    AND meal_type = 'Breakfast';  
41  
42    IF v_count_breakfast = 0  
43    THEN  
44    □    INSERT INTO diet_plan_details(cust_id,dietician_id,diet_plan_id,  
45        day,  
46        meal_type,  
47  
0% ▲ 12:155 |  
t vines skinned
```

```

44      INSERT INTO diet_plan_details(cust_id,dietician_id,diet_plan_id,
45                                         day,
46                                         meal_type,
47                                         calories,
48                                         proteins,
49                                         carbohydrates,
50                                         fat)
51      VALUES (v_cust_id,v_dietician_id,v_diet_plan_id,
52                v_day,
53                'Breakfast',
54                v_breakfast_calories,
55                v_breakfast_proteins,
56                v_breakfast_carbohydrates,
57                v_breakfast_fat);
58      ELSEIF v_count_breakfast > 0
59      THEN
60          UPDATE diet_plan_details
61              SET calories = v_breakfast_calories,
62                  proteins = v_breakfast_proteins,
63                  carbohydrates = v_breakfast_carbohydrates,
64                  fat = v_breakfast_fat
65              WHERE cust_id = v_cust_id
66              and dietician_id = v_dietician_id
67              and diet_Plan_id = v_diet_Plan_id
68              AND day = v_day
69              AND meal_type = 'Breakfast';
70      END IF;
71
72      SELECT count(1)
73          INTO v_count_lunch
74          FROM diet_plan_details
75          WHERE cust_id = v_cust_id
76          and dietician_id = v_dietician_id
77          and diet_Plan_id = v_diet_Plan_id
78          AND day = v_day
79          AND meal_type = 'Lunch';
80
81      IF v_count_lunch = 0
82      THEN
83          INSERT INTO diet_plan_details(cust_id,dietician_id,diet_plan_id,
84                                         day,
85                                         meal_type,
86                                         calories,
87                                         proteins,
88                                         carbohydrates,
89                                         fat)

```

```

88                               carbohydrates,
89                               fat)
90      VALUES (v_cust_id,v_dietician_id,v_diet_plan_id,
91                 v_day,
92                 'Lunch',
93                 v_lunch_calories,
94                 v_lunch_proteins,
95                 v_lunch_carbohydrates,
96                 v_lunch_fat);
97 ELSEIF v_count_lunch > 0
98 THEN
99   UPDATE diet_plan_details
100    SET calories = v_lunch_calories,
101        proteins = v_lunch_proteins,
102        carbohydrates = v_lunch_carbohydrates,
103        fat = v_lunch_fat
104 WHERE cust_id = v_cust_id
105 and dietician_id = v_dietician_id
106 and diet_Plan_id = v_diet_Plan_id
107 AND day = v_day
108 AND meal_type = 'Lunch';
109 END IF;

110
111
112 SELECT count(1)
113 INTO v_count_dinner
114 FROM diet_plan_details
115 WHERE cust_id = v_cust_id
116 and dietician_id = v_dietician_id
117 and diet_Plan_id = v_diet_Plan_id
118 AND day = v_day
119 AND meal_type = 'Dinner';
120
121 IF v_count_dinner = 0
122 THEN
123   INSERT INTO diet_plan_details(cust_id,dietician_id,diet_plan_id,
124                                 day,
125                                 meal_type,
126                                 calories,
127                                 proteins,
128                                 carbohydrates,
129                                 fat)
130   VALUES (v_cust_id,v_dietician_id,v_diet_plan_id,
131             v_day,
132             'Dinner'

```

```

128
129           carbohydrates,
130           fat)
130     VALUES (v_cust_id,v_dietician_id,v_diet_plan_id,
131           v_day,
132           'Dinner',
133           v_dinner_calories,
134           v_dinner_proteins,
135           v_dinner_carbohydrates,
136           v_dinner_fat);
137   ELSEIF v_count_dinner > 0
138   THEN
139     UPDATE diet_plan_details
140     SET calories = v_dinner_calories,
141         proteins = v_dinner_proteins,
142         carbohydrates = v_dinner_carbohydrates,
143         fat = v_dinner_fat
144     WHERE   cust_id = v_cust_id
145     and dietician_id = v_dietician_id
146     and diet_Plan_id = v_diet_Plan_id
147     AND day = v_day
148     AND meal_type = 'Dinner';
149   END IF;
150   IF `_rollback` THEN ROLLBACK;  SET v_result = -1;
151   ELSE COMMIT;    SET v_result = 1;
152   END IF;
153 END;
154 $$;
155 DELIMITER ;

```

## 2. d\_finalize\_new\_diet\_plan

```
1
2     DELIMITER $$;
3     CREATE PROCEDURE d_finalize_new_diet_plan(
4         IN v_custid_dietstartdate VARCHAR(50),
5             OUT v_result           INT)
6     BEGIN
7
8         DECLARE `_rollback` BOOL DEFAULT 0;
9         DECLARE CONTINUE HANDLER FOR SQLEXCEPTION SET `_rollback` = 1;
10        START TRANSACTION;
11        update diet_plan set new_diet_plan_ind='N' where
12            concat(cust_id,'.',diet_start_date)=v_custid_dietstartdate;
13        IF `_rollback` THEN ROLLBACK;   SET v_result = -1;
14        ELSE COMMIT;    SET v_result = 1;
15        END IF;
16    END;
17    $$;
18    DELIMITER ;
```

## 3. d\_finalize\_update\_diet\_plan

```
22
23     DELIMITER $$;
24     • CREATE PROCEDURE d_finalize_update_diet_plan(
25         IN v_custid_dietstartdate VARCHAR(50),
26             OUT v_result           INT)
27     BEGIN
28
29         DECLARE `_rollback` BOOL DEFAULT 0;
30         DECLARE CONTINUE HANDLER FOR SQLEXCEPTION SET `_rollback` = 1;
31         START TRANSACTION;
32         update customer_requests set update_diet_plan_ind='N'
33         where exists (select * from diet_plan where diet_plan.cust_id=customer_requests.cust_id
34             and customer_requests.dietician_id=diet_plan.dietician_id and
35             customer_requests.diet_plan_id=diet_plan.diet_plan_id
36             and concat(cust_id,'.',diet_start_date)=v_custid_dietstartdate);
37         IF `_rollback` THEN
38             ROLLBACK;
39             SET v_result = -1;
40         ELSE
41             COMMIT;
42             SET v_result = 1;
43         END IF;
44     END;
45     $$;
```

#### 6.4.4. Restaurant

##### 1. proc\_update\_menu

```
31 • -- Procedure to update Menu
32 drop procedure if exists proc_update_menu;
33 DELIMITER $$

34 • ⊕ create procedure proc_update_menu(
35     IN v_restaurant_id INT,
36             IN v_item_name VARCHAR(10),
37             IN v_monday    VARCHAR(1),
38             IN v_tuesday   VARCHAR(1),
39             IN v_wednesday VARCHAR(1),
40             IN v_thursday  VARCHAR(1),
41             IN v_friday    VARCHAR(1),
42             IN v_saturday  VARCHAR(1),
43             IN v_sunday    VARCHAR(1),
44             OUT v_result   INT)
45 ⊕ BEGIN
46
47
48
49     DECLARE v_count INT DEFAULT 0;
50     DECLARE v_item  INT DEFAULT 0;
51     DECLARE `rollback` BOOL DEFAULT 0;
52     DECLARE CONTINUE HANDLER FOR SQLEXCEPTION SET `rollback` = 1;
53     SET v_result = 0;
54     START TRANSACTION;

55
56     SELECT count(*)
57         INTO v_item
58         FROM restaurant_item_day
59         WHERE restaurant_id = v_restaurant_id AND item_name = v_item_name and day_of_week='Monday';
60     if v_monday='Y' and v_item=0 then
61         ⊕ INSERT INTO restaurant_item_day(restaurant_id,
62                                         item_name,
63                                         day_of_week)
64             VALUES (v_restaurant_id, v_item_name, 'Monday');
65     elseif v_monday='N' and v_item=1 then
66         delete from restaurant_item_day where restaurant_id = v_restaurant_id AND item_name = v_item_name and day_of_week='Monday';
67     end if;
68
```

```

19  SELECT count(*)
20  INTO v_item
21  FROM restaurant_item_day
22  WHERE restaurant_id = v_restaurant_id AND item_name = v_item_name and day_of_week='Tuesday';
23  if v_tuesday='Y' and v_item=0 then
24      INSERT INTO restaurant_item_day(restaurant_id,
25          item_name,
26          day_of_week)
27          VALUES (v_restaurant_id, v_item_name, 'Tuesday');
28  elseif v_tuesday='N' and v_item=1 then
29      delete from restaurant_item_day where restaurant_id = v_restaurant_id AND item_name = v_item_name and day_of_week='Tuesday';
30  end if;
31
32  SELECT count(*)
33  INTO v_item
34  FROM restaurant_item_day
35  WHERE restaurant_id = v_restaurant_id AND item_name = v_item_name and day_of_week='Wednesday';
36  if v_wednesday='Y' and v_item=0 then
37      INSERT INTO restaurant_item_day(restaurant_id,
38          item_name,
39          day_of_week)
40          VALUES (v_restaurant_id, v_item_name, 'Wednesday');
41  elseif v_wednesday='N' and v_item=1 then
42      delete from restaurant_item_day where restaurant_id = v_restaurant_id AND item_name = v_item_name and day_of_week='Wednesday';
43  end if;
44
45
46  SELECT count(*)
47  INTO v_item
48  FROM restaurant_item_day
49  WHERE restaurant_id = v_restaurant_id AND item_name = v_item_name and day_of_week='Thursday';
50  if v_thursday='Y' and v_item=0 then
51      INSERT INTO restaurant_item_day(restaurant_id,
52          item_name,
53          day_of_week)
54          VALUES (v_restaurant_id, v_item_name, 'Thursday');
55  elseif v_thursday='N' and v_item=1 then
56      delete from restaurant_item_day where restaurant_id = v_restaurant_id AND item_name = v_item_name and day_of_week='Thursday';

```

```

56      delete from restaurant_item_day where restaurant_id = v_restaurant_id AND item_name = v_item_name and day_of_week='Thursday';
57  end if;
58
59  SELECT count(*)
60  INTO v_item
61  FROM restaurant_item_day
62  WHERE restaurant_id = v_restaurant_id AND item_name = v_item_name and day_of_week='Friday';
63  if v_friday='Y' and v_item=0 then
64    INSERT INTO restaurant_item_day(restaurant_id,
65          item_name,
66          day_of_week)
67        VALUES (v_restaurant_id, v_item_name, 'Friday');
68  elseif v_friday='N' and v_item=1 then
69    delete from restaurant_item_day where restaurant_id = v_restaurant_id AND item_name = v_item_name and day_of_week='Friday';
70  end if;
71
72  SELECT count(*)
73  INTO v_item
74  FROM restaurant_item_day
75  WHERE restaurant_id = v_restaurant_id AND item_name = v_item_name and day_of_week='Saturday';
76  if v_saturday='Y' and v_item=0 then
77    INSERT INTO restaurant_item_day(restaurant_id,
78          item_name,
79          day_of_week)
80        VALUES (v_restaurant_id, v_item_name, 'Saturday');
81  elseif v_saturday='N' and v_item=1 then
82    delete from restaurant_item_day where restaurant_id = v_restaurant_id AND item_name = v_item_name and day_of_week='Saturday';
83  end if;
84
85  SELECT count(*)
86  INTO v_item
87  FROM restaurant_item_day
88  WHERE restaurant_id = v_restaurant_id AND item_name = v_item_name and day_of_week='Sunday';
89  if v_sunday='Y' and v_item=0 then
90    INSERT INTO restaurant_item_day(restaurant_id,
91          item_name,
92          day_of_week)
93        VALUES (v_restaurant_id, v_item_name, 'Sunday');

```

```

95      delete from restaurant_item_day where restaurant_id = v_restaurant_id AND item_name = v_item_name and day_of_week='Sunday';
96  end if;
97  commit;
98
99  IF `__rollback` THEN ROLLBACK;
10 ELSE COMMIT;
11 END IF;
12
13
14 END;
15 $$

16 DELIMITER ;
17
18

```

## 2. r\_add\_menu\_item

```
45      THEN
46      INSERT INTO restaurant_item_day(restaurant_id,
47                      item_name,
48                      day_of_week)
49          VALUES (v_restaurant_id, v_item_name,'Wednesday');
50      END IF;
51      IF v_thursday = 'Y'
52          THEN
53              INSERT INTO restaurant_item_day(restaurant_id,
54                      item_name,
55                      day_of_week)
56              VALUES (v_restaurant_id, v_item_name, 'Thursday');
57          END IF;
58          IF v_friday = 'Y'
59              THEN
60                  INSERT INTO restaurant_item_day(restaurant_id,
61                      item_name,
62                      day_of_week)
63                      VALUES (v_restaurant_id, v_item_name, 'Friday');
64                  END IF;
65                  IF v_saturday = 'Y'
66                      THEN
67                          INSERT INTO restaurant_item_day(restaurant_id,
68                              item_name,
69                              day_of_week)
70                              VALUES (v_restaurant_id, v_item_name, 'Saturday');
71                          END IF;
72                          IF v_sunday = 'Y'
73                              THEN
74                                  INSERT INTO restaurant_item_day(restaurant_id,
75                                      item_name,
76                                      day_of_week)
77                                      VALUES (v_restaurant_id, v_item_name, 'Sunday');
78                                  END IF;
79                                  COMMIT;
80                                  SET v_result = 1;
81 elseif v_item > 0
82     THEN
83         SET v_result = 2;
84     END IF;
85     IF `_rollback` THEN ROLLBACK;
86     ELSE COMMIT;
87     END IF;
88     END;
89     $$;
90     DELIMITER
```

```

1 • DROP PROCEDURE IF EXISTS r_add_menu_item;
2 DELIMITER $$;
3 • CREATE PROCEDURE r_add_menu_item(IN v_restaurant_id INT,
4 IN v_item_name VARCHAR(10), IN v_meal_type VARCHAR(10),
5 IN v_calories VARCHAR(10), IN v_proteins VARCHAR(10),
6 IN v_carbohydrates VARCHAR(10), IN v_fat VARCHAR(10),
7 IN v_monday VARCHAR(1), IN v_tuesday VARCHAR(1),
8 IN v_wednesday VARCHAR(1), IN v_thursday VARCHAR(1),
9 IN v_friday VARCHAR(1), IN v_saturday VARCHAR(1),
10 IN v_sunday VARCHAR(1), OUT v_result INT)
11 BEGIN
12     DECLARE v_count INT DEFAULT 0;
13     DECLARE v_item INT DEFAULT 0;
14     DECLARE `rollback` BOOL DEFAULT 0;
15     DECLARE CONTINUE HANDLER FOR SQLEXCEPTION SET `rollback` = 1;
16     SET v_result = 0;
17     START TRANSACTION;
18     SELECT count(*)
19         INTO v_item
20         FROM item
21         WHERE restaurant_id = v_restaurant_id AND item_name = v_item_name;
22     IF v_item = 0
23     THEN
24         INSERT INTO item(restaurant_id, item_name, meal_type,
25                         calories, proteins, carbohydrates,
26                         fat, offered_ind)
27             VALUES (v_restaurant_id, v_item_name, v_meal_type,
28                     v_calories, v_proteins, v_carbohydrates,
29                     v_fat, 'N');
30     IF v_monday = 'Y'
31     THEN
32         INSERT INTO restaurant_item_day(restaurant_id,
33                                         item_name,
34                                         day_of_week)
35             VALUES (v_restaurant_id, v_item_name, 'Monday');
36     END IF;
37     IF v_tuesday = 'Y'
38     THEN
39         INSERT INTO restaurant_item_day(restaurant_id,
40                                         item_name,
41                                         day_of_week)
42             VALUES (v_restaurant_id, v_item_name, 'Tuesday');
43     END IF;
44     IF v_wednesday = 'Y'
45     THEN
46         INSERT INTO restaurant_item_day(restaurant_id,

```

## 6.5. Additional DB Objects

### 6.5.1. Triggers

Triggers `update_offered_ind_for_insert` and `update_offered_ind_for_delete` exist on `restaurant_item_day` table and are used manipulate `offered_ind` in `Item` table. The item table has an `offered_ind` column to indicate if the restaurant is offering the particular item on any day. If the item is being offered on any day, then the day on which the item is offered will be in `restaurant_item_day` table and in item table `offered_ind` should be Y, else it will be N. If the restaurant wants to stop offering the item, we cannot delete the item from item table due to the necessity to maintain order history. The `offered_ind` helps in determining if the item is being offered on any day or not.

1. **`update_offered_ind_for_insert`:** If the restaurant is not offering the item currently on any day, and decides that it wants to offer the item, then records will be inserted in `restaurant_item_day` table. This trigger updates `offered_ind` to Y from N in item table.

```
1
2 • DROP TRIGGER IF EXISTS update_offered_ind_for_insert;
3
4
5 • DELIMITER $$;
6
7 CREATE TRIGGER update_offered_ind_for_insert
8 BEFORE INSERT
9 ON restaurant_item_day
10 FOR EACH ROW
11 BEGIN
12     DECLARE v_count INT;
13     SELECT COUNT(*) INTO v_count FROM restaurant_item_day
14     WHERE restaurant_id = NEW.restaurant_id AND item_name = NEW.item_name;
15     IF v_count>0 THEN
16         UPDATE Item
17         SET Offered_Ind = 'Y'
18         WHERE restaurant_id = NEW.restaurant_id AND item_name = NEW.item_name;
19     ELSEIF v_count=0 THEN
20         UPDATE Item
21         SET Offered_Ind = 'N'
22         WHERE restaurant_id = NEW.restaurant_id AND item_name = NEW.item_name;
23     END IF;
24     END;
25
26 DELIMITER :
```

2. **`update_offered_ind_for_delete`:** If the restaurant is currently offering an item, and wants to stop offering the item, then all records for the item in `restaurant_item_day` table will be deleted. In this case, the trigger updates `offered_ind` to N in item table.

```

25
26 •   DROP TRIGGER IF EXISTS update_offered_ind_for_delete;
27
28 •   DELIMITER $$ 
29
30     CREATE TRIGGER update_offered_ind_for_delete
31       AFTER DELETE
32         ON restaurant_item_day
33           FOR EACH ROW
34             BEGIN
35               DECLARE v_count INT;
36               SELECT
37                 COUNT(*)
38               INTO v_count FROM
39                 restaurant_item_day
40               WHERE
41                 restaurant_id = OLD.restaurant_id
42                   AND item_name = OLD.item_name;
43               IF v_count>0 THEN
44                 UPDATE Item
45                   SET Offered_Ind = 'Y'
46                   WHERE restaurant_id = OLD.restaurant_id AND item_name = OLD.item_name;
47               ELSEIF v_count=0 THEN
48                 UPDATE Item
49                   SET Offered_Ind = 'N'
50                   WHERE restaurant_id = OLD.restaurant_id AND item_name = OLD.item_name;
51               END IF;
52             END;
53             $$ 
54             DELIMITER ;

```

### 6.5.2. Index

Index customer\_meal\_plan\_idx1 is created on customer\_meal\_plan table on cust\_ID, start\_date columns as these columns are parent referential keys and are part of composite primary keys of the table.

```

1
2 •   CREATE INDEX customer_meal_plan_idx1 ON
3     customer_meal_plan (cust_ID, start_date);
4

```

## 6.6. DB Execution Screenshots

Below are DB execution screenshots for some important operations for the application:

### 6.6.1. Admin

1. proc\_d\_ins\_dietn\_regn\_details: The procedure is called to insert the details of the dietitian from UI to database. This is called during the registration of the dietitian by the admin. The below screenshot shows the output from dietician table with the newly inserted dietitian 'dietician102'

The screenshot shows a MySQL Workbench interface. At the top, there is a SQL editor window containing the following code:

```
30
31 • CALL proc_d_ins_dietn_regn_details ('dietician102', 'dietician102@mm.com', '602c57ffb51af99d6f3b54c0ee9587bb110fb990',
32      10, 'M.S.', @vc, @eid);
33
34
35 •  SELECT * FROM dietician;
```

Below the code, the status bar shows "250%" and "12:79". The main area displays a "Result Grid" with the following data:

dietician_id	dietician_name	email_id	password	experience	qualification
201	dietician1	dietician1@gmail.com	88ea39439e74fa27c09a4fc0bc8eb6d00978392	3.50	M.S.
202	dietician2	dietician2@gmail.com	88ea39439e74fa27c09a4fc0bc8eb6d00978392	0.00	M.S.
203	dietician3	dietician3@gmail.com	88ea39439e74fa27c09a4fc0bc8eb6d00978392	3.00	M.S.
10000	dietician100	dietician100@mm.com	2bdba55ec2c947b1c5d0f68385f2cc5a78488c24	10.00	M.S.
10001	dietician101	dietician101@mm.com	2bdba55ec2c947b1c5d0f68385f2cc5a78488c24	10.00	M.S.
10002	dietician102	dietician102@mm.com	2bdba55ec2c947b1c5d0f68385f2cc5a78488c24	10.00	M.S.
HULL	HULL	HULL	HULL	HULL	HULL

### 6.6.2. Customer

1. proc\_dietician\_yes: The procedure is called to insert the details of customer who wishes to enroll with a diet plan. The below screenshot shows the output from one of the updated tables which is customer\_meal\_plan with the new cust\_ID 50000.

The screenshot shows the MySQL Workbench interface. In the SQL tab, the following code is executed:

```
'2 • CALL proc_dietician_yes (50000, '2019-10-10', 'visa', '123456789', '201', '2019-01-01', 'low', '150', '175', 0,
'3   'Diabetes','undefined', 'undefined', 'undefined', 'undefined', 'undefined', 'Peanuts', 'undefined', 'undefined', 'undefined',
'4   @res, @cnt);
'5
'6 •  select @res,@cnt;
'7
'8 •  select * from customer_meal_plan;
```

The status bar at the bottom indicates "300" rows, "12:470" time, and "1 error found". Below the SQL tab is a Result Grid tab displaying the data from the customer\_meal\_plan table:

cust_id	start_date	plan_id
1001	2019-02-07	1
1001	2019-03-30	1
1008	2019-05-03	1
1010	2019-05-06	1
50000	2019-10-10	1
1002	2019-03-22	2
1002	2019-04-22	2
1003	2019-04-04	5

2. diet\_ratings\_view: This view is created to combine the average ratings received by each dietitian from all the customers, with the other details of the dietitian like name, email ID and qualification. If no ratings are provided to a particular dietitian, then 'No Ratings' is displayed. The below screenshot shows the sample data executed from the view. It is seen that only dieticain1 has the ratings.

The screenshot shows the MySQL Workbench interface. In the SQL tab, the following code is executed:

```
33
34 •  SELECT * FROM diet_ratings_view;
35
36
```

The status bar at the bottom indicates "1:174" time. Below the SQL tab is a Result Grid tab displaying the data from the diet\_ratings\_view:

dietician_id	dietician_name	email_id	experience	qualification	ratings
201	dietician1	dietician1@gmail.com	3.50	M.S.	5.0000
202	dietician2	dietician2@gmail.com	0.00	M.S.	No Ratings
203	dietician3	dietician3@gmail.com	3.00	M.S.	No Ratings
10000	dietician100	dietician100@mm.com	10.00	M.S.	No Ratings
10001	dietician101	dietician101@mm.com	10.00	M.S.	No Ratings

3. c\_customer\_remaining\_meal\_count: For each customer this view provides the number of Breakfast, Lunch and Dinner meals left in his current meal plan. Below screenshot shows the remaining meal count for the current users from sample data.

The screenshot shows a database query results grid. At the top, there are three lines of code: 234 • SELECT \* FROM c\_customer\_remaining\_meal\_count;, 235, and 236. Below the code is a progress bar indicating 100% completion and the time 47:234. The main area is a 'Result Grid' with the following columns: cust\_id, start\_date, rem\_breakfast\_count, rem\_lunch\_count, and rem\_dinner\_count. The data is as follows:

cust_id	start_date	rem_breakfast_count	rem_lunch_count	rem_dinner_count
1008	2019-05-03	30	30	30
1010	2019-05-06	30	30	30
1002	2019-04-22	0	0	15
1003	2019-05-04	0	15	15
1004	2019-05-03	0	30	30
1005	2019-05-01	0	30	15
1009	2019-05-06	30	15	0
1001	2019-04-30	30	30	30
1007	2019-04-08	30	30	30

4. c\_diet\_history\_view: The purpose of this view is to return the complete information for the list of diet plans each customer has previously enrolled. Below screenshot shows the complete diet plan details of first 9 days along with the diet progress with respect to cust id 1001 and dietician id 201.

406  
 407 • `SELECT * FROM c_diet_history_view;`  
 408  
 409  
 410

100% 1:408

**Result Grid** Filter Rows: Search Export:

cust_id	dietician_id	diet_plan_id	day	meal_type	calories	proteins	carbohydrates	fat	followed	additional_intake
1001	201	1	1	Breakfast	100.00	20.00	20.00	30.00	Y	NULL
1001	201	1	1	Dinner	100.00	40.00	100.00	30.00	Y	NULL
1001	201	1	1	Lunch	180.00	20.00	80.00	30.00	Y	NULL
1001	201	1	2	Breakfast	120.00	20.00	20.00	30.00	Y	Biryani
1001	201	1	2	Dinner	130.00	45.00	100.00	30.00	N	Biryani
1001	201	1	2	Lunch	150.00	30.00	70.00	30.00	N	Biryani
1001	201	1	3	Breakfast	120.00	30.00	30.00	30.00	NULL	NULL
1001	201	1	3	Dinner	130.00	45.00	10.00	30.00	NULL	NULL
1001	201	1	3	Lunch	150.00	30.00	75.00	30.00	NULL	NULL
1001	201	1	4	Breakfast	100.00	20.00	20.00	10.00	Y	NULL
1001	201	1	4	Dinner	130.00	45.00	80.00	30.00	Y	NULL
1001	201	1	4	Lunch	140.00	30.00	70.00	20.00	N	NULL
1001	201	1	5	Breakfast	170.00	20.00	20.00	30.00	Y	Apple Pie
1001	201	1	5	Dinner	130.00	45.00	70.00	30.00	N	Apple Pie
1001	201	1	5	Lunch	150.00	35.00	70.00	30.00	Y	Apple Pie
1001	201	1	6	Breakfast	120.00	20.00	20.00	10.00	Y	NULL
1001	201	1	6	Dinner	130.00	45.00	60.00	30.00	Y	NULL
1001	201	1	6	Lunch	150.00	75.00	70.00	20.00	Y	NULL
1001	201	1	7	Breakfast	100.00	20.00	20.00	30.00	Y	NULL
1001	201	1	7	Dinner	110.00	45.00	60.00	30.00	Y	NULL
1001	201	1	7	Lunch	150.00	85.00	70.00	30.00	Y	NULL
1001	201	1	8	Breakfast	170.00	35.00	20.00	10.00	NULL	NULL
1001	201	1	8	Dinner	130.00	45.00	85.00	30.00	NULL	NULL
1001	201	1	8	Lunch	150.00	35.00	65.00	30.00	NULL	NULL
1001	201	1	9	Breakfast	120.00	35.00	50.00	10.00	Y	Pastry
1001	201	1	9	Dinner	130.00	25.00	85.00	30.00	Y	Pastry
1001	201	1	9	Lunch	150.00	35.00	75.00	30.00	Y	Pastry
1001	201	1	10	Breakfast	130.00	35.00	50.00	20.00	Y	NULL

5. c\_place\_order: This stored procedure is used to place an order for the items selected by the customer. Below screenshot shows the execution while customer 1008 places an order for breakfast, lunch and dinner meal types.

```

292
293 • CALL c_place_order( 1007, 'Indian', 'Dosa', 'Malaysian', 'Burmese Rice','Indian','Naan', '06:30','12:30', '20:30',
294   @bf_chk, @ln_chk, @din_chk);
295 • SELECT * FROM orders WHERE cust_ID = 1007;
296

130% 1:294

Result Grid Filter Rows: Q Search Edit: Export/Import: 

```

cust_ID	Restaurant_ID	Item_Name	order_date	pickup_time
1007	102	Dosa	2019-05-05	06:30:00
1007	102	Naan	2019-05-05	20:30:00
1007	103	Burmese Rice	2019-05-05	12:30:00
NULL	NULL	NULL	NULL	NULL

6. c\_update\_diet\_progress: This stored procedure is used to insert the new diet progress information provided by customer. Below screenshot shows execution when customer 1008 tries to update his progress for 1<sup>st</sup> May.

```

396
397
398
399 • CALL c_update_diet_progress(1008,'2019-05-01', 'Y', 'Y', 'Y', 'Nothing', @v_cnt);
400 • SELECT @v_cnt;
401 • SELECT * FROM diet_progress WHERE cust_id = 1008;
402
403

100% 1:396

Result Grid Filter Rows: Q Search Edit: Export/Import: 

```

cust_ID	dietician_ID	diet_plan_id	day	breakfast	lunch	dinner	additional_intake
1008	202	1	1	Y	Y	Y	NULL
1008	202	1	2	Y	Y	N	1 Slice Cake
1008	202	1	7	Y	Y	Y	Nothing
1008	202	1	8	Y	Y	N	1 cake
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

### 6.6.3. Dietician

7. d\_add\_diet\_details: This procedure is used by dietician to add diet plan details for a customer for a particular day for all 3 meal types.

The screenshot shows the MySQL Workbench interface with the following details:

- SQL Editor:** Contains the following code:

```
20
21 • CALL d_add_diet_details('1021;2019-05-08',1,100,20,20,30,180,20,80,30,100,40,100,30 , @v_result);
22 • SELECT @v_result;
23 • select * from diet_plan_details where cust_id=1021 and day=1 and dietician_id=205 and diet_plan_id=2;
24 |
```
- Result Grid:** Displays the results of the query in the SQL editor. The table has columns: cust\_ID, dietician\_ID, diet\_plan\_id, day, meal\_type, calories, proteins, carbohydrates, fat. The data is:

cust_ID	dietician_ID	diet_plan_id	day	meal_type	calories	proteins	carbohydrates	fat
1021	205	2	1	Breakfast	100.00	20.00	20.00	30.00
1021	205	2	1	Dinner	100.00	40.00	100.00	30.00
1021	205	2	1	Lunch	180.00	20.00	80.00	30.00
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL
- Action Output:** Shows the history of actions taken during the session:

Action	Time	Response
select * from diet_plan_details where cust_id=1021 and day=1 and dietician_id=205 a...	16:30:12	3 row(s) returned
SELECT @v_result LIMIT 0, 1000	16:30:24	1 row(s) returned
SELECT @v_result LIMIT 0, 1000	16:30:27	1 row(s) returned
select * from diet_plan_details where cust_id=1021 and day=1 and dietician_id=205 a...	16:30:33	3 row(s) returned

2. `d_customer_diet_plan_details_view` - This view gives the diet plan details for all 30 days for all 3 meal types – breakfast, lunch and dinner for all the diet plans ids of a customer.

1 ● select \* from `d_customer_diet_plan_details_view`;

100% 49:1

Result Grid | Filter Rows: Search Export:

cust_id	dietician_id	diet_start_date	day	calendar_date	new_diet_plan_request_date	meal_type	calories	proteins	carbohydrates	fat
1001	201	2019-02-07	1	2019-02-08	Y	Breakfast	100.00	20.00	20.00	30.00
1001	201	2019-02-07	1	2019-02-08	Y	Lunch	180.00	20.00	80.00	30.00
1001	201	2019-02-07	1	2019-02-08	Y	Dinner	100.00	40.00	100.00	30.00
1001	201	2019-02-07	2	2019-02-09	Y	Breakfast	120.00	20.00	20.00	30.00
1001	201	2019-02-07	2	2019-02-09	Y	Lunch	150.00	30.00	70.00	30.00
1001	201	2019-02-07	2	2019-02-09	Y	Dinner	130.00	45.00	100.00	30.00
1001	201	2019-02-07	3	2019-02-10	Y	Breakfast	120.00	30.00	30.00	30.00
1001	201	2019-02-07	3	2019-02-10	Y	Lunch	150.00	30.00	75.00	30.00
1001	201	2019-02-07	3	2019-02-10	Y	Dinner	130.00	45.00	10.00	30.00
1001	201	2019-02-07	4	2019-02-11	Y	Breakfast	100.00	20.00	20.00	10.00
1001	201	2019-02-07	4	2019-02-11	Y	Lunch	140.00	30.00	70.00	20.00
1001	201	2019-02-07	4	2019-02-11	Y	Dinner	130.00	45.00	80.00	30.00
1001	201	2019-02-07	5	2019-02-12	Y	Breakfast	170.00	20.00	20.00	30.00
1001	201	2019-02-07	5	2019-02-12	Y	Lunch	150.00	35.00	70.00	30.00
1001	201	2019-02-07	5	2019-02-12	Y	Dinner	130.00	45.00	70.00	30.00
1001	201	2019-02-07	6	2019-02-13	Y	Breakfast	120.00	20.00	20.00	10.00
1001	201	2019-02-07	6	2019-02-13	Y	Lunch	150.00	75.00	70.00	20.00
1001	201	2019-02-07	6	2019-02-13	Y	Dinner	130.00	45.00	60.00	30.00
1001	201	2019-02-07	7	2019-02-14	Y	Breakfast	100.00	20.00	20.00	20.00

3. `d_customer_diet_plans_list_view` – This view is used to find all diet plans associated with a customer along with start dates, pay dates, dietician of the diet plans.

1 ● select \* from `d_customer_diet_plans_list_view`;

100% 15:1

Result Grid | Filter Rows: Search Export:

cust_id	cust_name	cust_email_id	cu...	dieti...	height	weight	acti...	sleep	di...	diet_start_d...	new_diet_plan_request_date	new_diet_plan_ind
1001	Sara	Sara@gmail.com	26	201	150.00	200.00	low	0	1	2019-02-07	2019-02-04	Y
1001	Sara	Sara@gmail.com	26	202	153.00	100.00	me...	2	1	2019-03-30	2019-03-27	Y
1021	James	James@gmail.com	25	204	150.00	200.00	low	7	1	2019-03-09	2019-03-06	Y
1021	James	James@gmail.com	25	205	150.00	195.00	low	7	1	2019-04-08	2019-04-05	Y
1021	James	James@gmail.com	25	205	150.00	180.00	low	8	2	2019-05-08	2019-05-05	Y
1022	Joseph	Joseph@gmail.com	33	205	150.00	200.00	low	6	1	2019-04-25	2019-04-22	Y

4. d\_diet\_plans\_cust\_health\_details\_view – This view shows the customer health details like allergies and diseases associated with each diet plan of the customer.

The screenshot shows a MySQL Workbench interface. At the top, a query is being run: "select \* from d\_diet\_plans\_cust\_health\_details\_view;". The results are displayed in a grid titled "Result Grid". The columns are: cust\_id, cust\_name, cust\_email\_id, cust\_age, dietician\_id, height, weight, activity..., sleep, diet\_pl..., diet\_start\_date, new\_diet\_plan..., new..., a\_d\_type, and allergy\_disease. The data includes multiple rows for customers like Sara and James, detailing their health metrics, diet plans, and associated allergies or diseases.

cust_id	cust_name	cust_email_id	cust_age	dietician_id	height	weight	activity...	sleep	diet_pl...	diet_start_date	new_diet_plan...	new...	a_d_type	allergy_disease
1001	Sara	Sara@gmail.com	26	201	150.00	200.00	low	0	1	2019-02-07	2019-02-04	Y	Allergy	Lactose
1001	Sara	Sara@gmail.com	26	201	150.00	200.00	low	0	1	2019-02-07	2019-02-04	Y	Allergy	Peanuts
1001	Sara	Sara@gmail.com	26	202	153.00	100.00	medium	2	1	2019-03-30	2019-03-27	Y	Allergy	No Allergy
1021	James	James@gmail.com	25	204	150.00	200.00	low	7	1	2019-03-09	2019-03-06	Y	Allergy	Lactose
1021	James	James@gmail.com	25	204	150.00	200.00	low	7	1	2019-03-09	2019-03-06	Y	Allergy	Peanuts
1021	James	James@gmail.com	25	205	150.00	195.00	low	7	1	2019-04-08	2019-04-05	Y	Allergy	Lactose
1021	James	James@gmail.com	25	205	150.00	195.00	low	7	1	2019-04-08	2019-04-05	Y	Allergy	Peanuts
1021	James	James@gmail.com	25	205	150.00	180.00	low	8	2	2019-05-08	2019-05-05	Y	Allergy	Lactose
1021	James	James@gmail.com	25	205	150.00	180.00	low	8	2	2019-05-08	2019-05-05	Y	Allergy	Peanuts
1022	Joseph	Joseph@gmail.com	33	205	150.00	200.00	low	6	1	2019-04-25	2019-04-22	Y	Allergy	No Allergy
1001	Sara	Sara@gmail.com	26	201	150.00	200.00	low	0	1	2019-02-07	2019-02-04	Y	Disease	HBP

#### 6.6.4. Restaurant

1. r\_add\_menu\_item - This procedure is used to add menu item for a particular restaurant, and input the item name, nutritional information and days of week on which the item will offered.

The screenshot shows a MySQL Workbench interface. A stored procedure is being executed with the following code:

```

12
13
14 SET @v_result = 0;
15 CALL r_add_menu_item(102,'Apple Pie','Breakfast',468,23,43,12,'Y','N','Y','N','N','N','N','N',@v_result);
16 SELECT @v_result;
17 select * from item i join restaurant_item_day r on i.item_name=r.item_name
18 and i.restaurant_id=r.restaurant_id where r.restaurant_id=102 and r.item_name='Apple Pie';

```

The results are displayed in a grid titled "Result Grid". The columns are: Restaurant\_ID, Item\_Name, Meal\_Type, Calories, Proteins, Carbohydrates, Fat, offered\_ind, Restaurant\_ID, Item\_Name, and day\_of\_week. The data shows three rows for the item "Apple Pie" offered at restaurant ID 102 on Monday, Thursday, and Wednesday.

Restaurant_ID	Item_Name	Meal_Type	Calories	Proteins	Carbohydrates	Fat	offered_ind	Restaurant_ID	Item_Name	day_of_week
102	Apple Pie	Breakfast	468.00	23.00	43.00	12.00	Y	102	Apple Pie	Monday
102	Apple Pie	Breakfast	468.00	23.00	43.00	12.00	Y	102	Apple Pie	Thursday
102	Apple Pie	Breakfast	468.00	23.00	43.00	12.00	Y	102	Apple Pie	Wednesday

2. r\_restaurant\_item\_details\_day\_view – This gives the list of items and item details like calories, proteins, days of which the item is offered for a restaurant.

The screenshot shows a MySQL Workbench interface with multiple tabs at the top: Query 1, SQL File 3\*, SQL File 4\*, and SQL File 5\*. The SQL File 5\* tab is active, displaying the following SQL code:

```
1 select * from r_restaurant_item_details_day_view
2 where restaurant_id=102 order by offered_ind desc,item_name, day_of_week;
```

The results are displayed in a grid titled "Result Grid". The columns are: restaurant\_id, item\_name, meal\_type, calories, proteins, carbohydrates, fat, offered\_ind, and day\_of\_week. The data shows various items offered by restaurant ID 102 across different days of the week, ordered by offering status (desc) and item name, and then by day of the week.

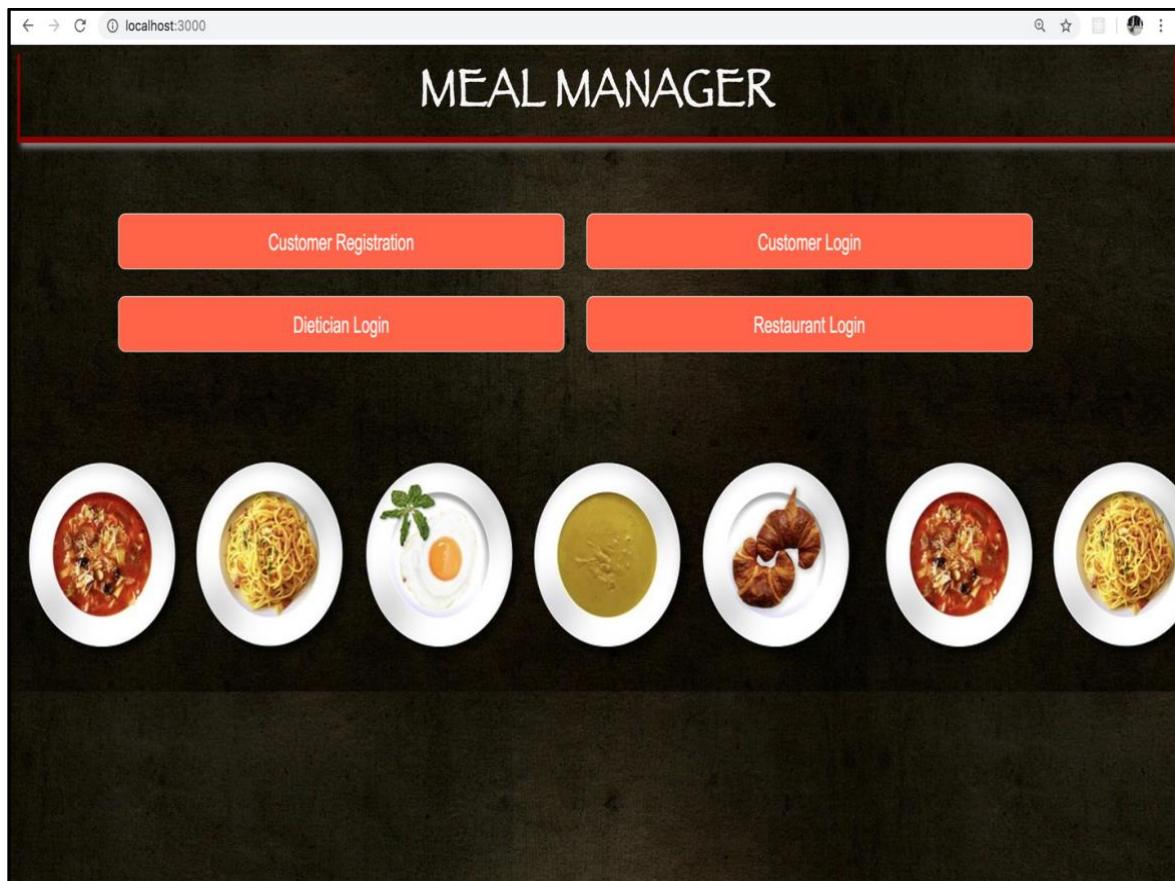
restaurant_id	item_name	meal_type	calories	proteins	carbohydrates	fat	offered_ind	day_of_week
102	Dosa	Breakfast	110.00	30.00	30.00	30.00	Y	Friday
102	Dosa	Breakfast	110.00	30.00	30.00	30.00	Y	Monday
102	Dosa	Breakfast	110.00	30.00	30.00	30.00	Y	Saturday
102	Dosa	Breakfast	110.00	30.00	30.00	30.00	Y	Sunday
102	Dosa	Breakfast	110.00	30.00	30.00	30.00	Y	Thursday
102	Dosa	Breakfast	110.00	30.00	30.00	30.00	Y	Tuesday
102	Dosa	Breakfast	110.00	30.00	30.00	30.00	Y	Wednesday
102	Sandwich	Breakfast	300.00	23.00	43.00	12.00	Y	Friday
102	Sandwich	Breakfast	300.00	23.00	43.00	12.00	Y	Tuesday
102	Chicken Curry	Dinner	150.00	80.00	20.00	30.00	N	NULL
102	Pizza	Lunch	785.00	43.00	264.00	12.00	N	NULL
102	Upma	Breakfast	110.00	30.00	30.00	30.00	N	NULL

## 7. Final Design of DB Application

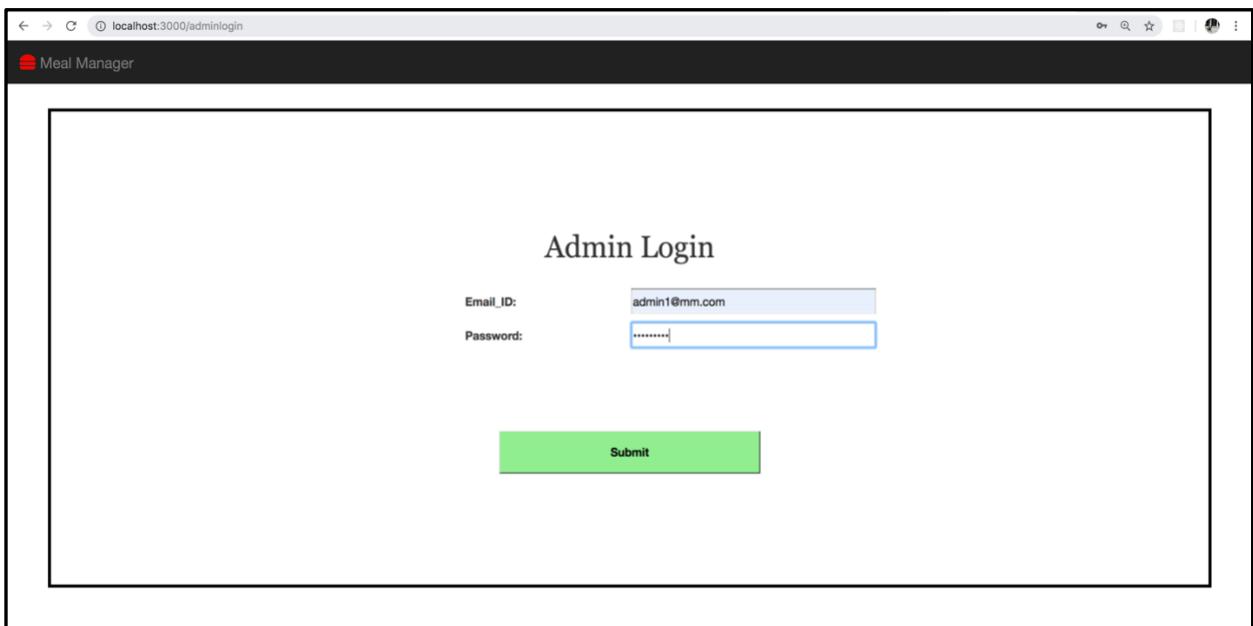
### 7.1. Application Screenshots

#### 7.1.1. Admin Operations

1. Application home page: onload of application homepage

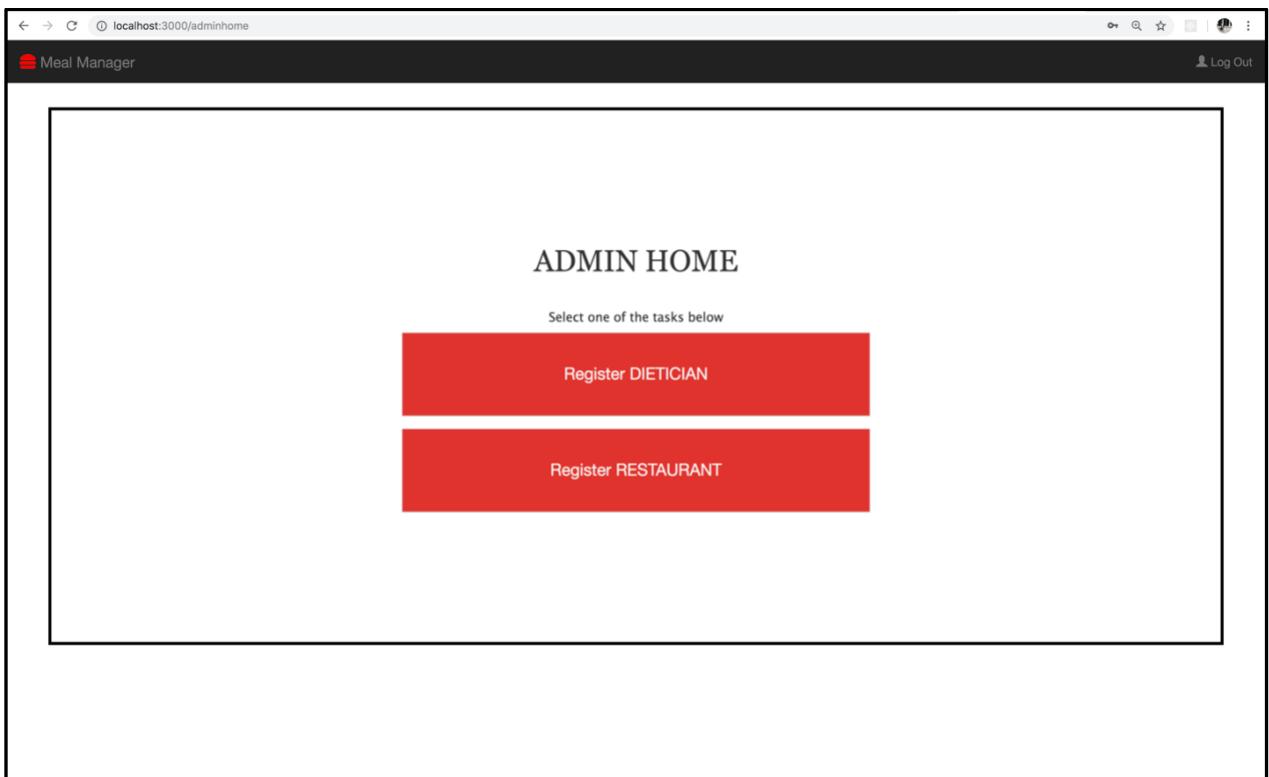


1. Admin login page: page displayed when localhost:3000/adminlogin is selected as url.  
Details entered after page load.



A screenshot of a web browser window showing the 'Admin Login' page. The URL in the address bar is 'localhost:3000/adminlogin'. The page has a dark header with the 'Meal Manager' logo. The main content area is titled 'Admin Login'. It contains two input fields: 'Email\_ID:' with the value 'admin1@mmr.com' and 'Password:' with a masked value. Below the inputs is a green 'Submit' button. The entire form is enclosed in a large black rectangular box.

2. Admin home page: page displayed when admin successfully logs in



A screenshot of a web browser window showing the 'ADMIN HOME' page. The URL in the address bar is 'localhost:3000/adminhome'. The page has a dark header with the 'Meal Manager' logo and a 'Log Out' link. The main content area is titled 'ADMIN HOME' and contains the text 'Select one of the tasks below'. There are two red rectangular buttons with white text: 'Register DIETICIAN' and 'Register RESTAURANT'.

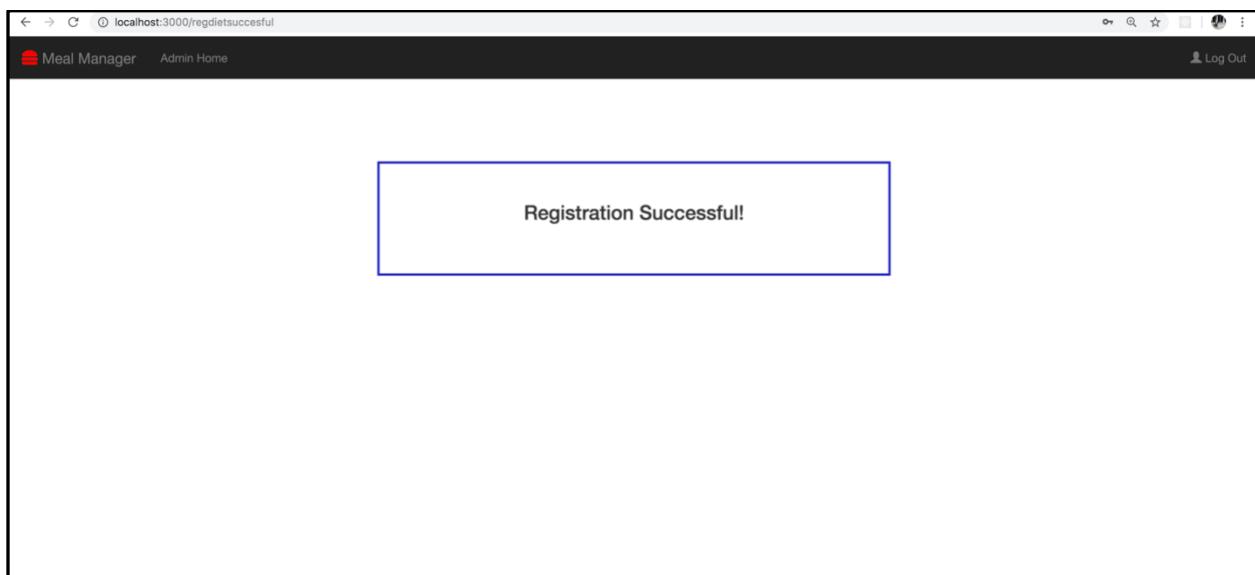
3. Register dietician page: page displayed when clicked on 'Register Dietician' button.  
Details entered after page load.

The screenshot shows a web browser window with the URL `localhost:3000/registerdietician?` in the address bar. The title bar says "Meal Manager". On the right, there is a "Log Out" link. The main content area has a large black rectangular redaction box. Inside this box, centered, is the heading "Register Dietician". Below the heading is a form with five input fields:

Name:	dietician1111
Email_ID:	dietician1111@gmail.com
Password:	.....
Experience(Years)	10
Qualification:	M.S.

Below the form is a green rectangular button with the word "SUBMIT" in white capital letters.

4. Register dietician successful page: page displayed on successful registration of dietician



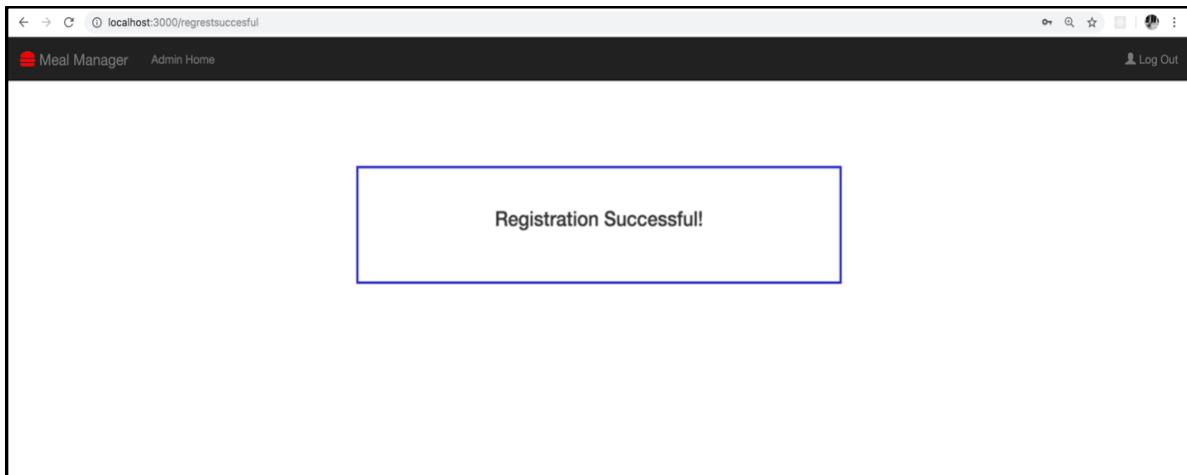
5. Admin logout page: page displayed when admin clicks on logout button

The screenshot shows a web browser window with the URL `localhost:3000/adminlogout` in the address bar. The title bar says "Meal Manager". The main content area is titled "Admin Login". It contains two input fields: "Email\_ID:" and "Password:", both currently empty. Below these is a green "Submit" button.

6. Register restaurant page: page displayed when Register RESTAURANT button is clicked (admin relogs with admin1). Details entered after page load.

The screenshot shows a web browser window with the URL `localhost:3000/adminlogout` in the address bar. The title bar says "Meal Manager" and has a "Log Out" link. The main content area is titled "Register Restaurant". It contains four input fields: "Name" (value: "restaurant1122"), "Email\_ID" (value: "resturant1122@gmail.com"), "Password" (value: "\*\*\*\*\*"), and "Location" (value: "San Jose"). Below these is a green "SUBMIT" button.

7. Register restaurant successful page: page displayed on successful registration of restaurant



### 7.1.2. Customer

1. Customer registration page: page loaded when 'Customer Registration'. Button is clicked in application homepage. Details entered after page load.

A screenshot of a web browser window titled "Meal Manager". The URL in the address bar is "localhost:3000/customerregistration?". The main content area features a red header bar with the text "Welcome to Meal Manager!". Below it, a message says "Please enter the below details:". There are five input fields: Name (customer1111), Email (customer1111@gmail.com), Password (\*\*\*\*\*), Date\_of\_Birth (01/01/2000), and Location (Montain View). A green "Register Me!" button is at the bottom.

- Customer login page: page displayed after successful registration of customer.  
Details entered after page load.

The screenshot shows a web browser window with the URL `localhost:3000/customerlogin`. The title bar says "Meal Manager". The main content area has a heading "Customer Login". It contains two input fields: "EMAIL\_ID\*" with the value "customer1111" and "PASSWORD\*" with the value "\*\*\*\*\*". Below the inputs is a green "Submit" button.

- Customer home page: page displayed when customer credentials are verified and logged in successfully

The screenshot shows a web browser window with the URL `localhost:3000/customerhome`. The title bar says "Meal Manager" and there is a "Log Out" link. The main content area has a red banner with the text "Make your choice!". Below the banner are two red buttons: "YES, I will get enrolled with a Dietician" and "NO, I will not get enrolled with a Dietician".

4. Dieticianyes page: page displayed when 'Yes, I will get enrolled with a dietitian' button is clicked. Details entered after page load.

The screenshot shows a web application interface for 'Meal Manager'. At the top, there are browser navigation icons, a search bar, and a 'Log Out' link. The main content area has a header 'Enter below lifestyle related details' with a red border. Below this, there are several input fields and dropdown menus:

- Weight: 100 lbs
- Height: 150 cms
- Measurement date of height and weight: 04/30/2019
- Activity Level: I will have HIGH level of physical exercise everyday
- Sleep: 6 - 8 hrs
- Select the diseases that you are diagnosed with:
  - Diabetes
  - High Blood Pressure
  - Low Blood Pressure
  - High Cholesterol
  - Low Cholesterol
- Select the ones that you are allergic to:
  - Peanuts
  - Pollen
  - Lactose
  - Soy
  - Mold
- Select start date for meal plan (date > than 2 days): 05/30/2019

Please select a dietican from the below list:

	Name of the Dietician	Email_ID	Experience	Qualification	Rating
<input type="radio"/>	dietician1	dietician1@gmail.com	3.5	M.S.	5.0000
<input type="radio"/>	dietician2	dietician2@gmail.com	0	M.S.	No Ratings
<input checked="" type="radio"/>	dietician3	dietician3@gmail.com	3	M.S.	No Ratings
<input type="radio"/>	dietician100	dietician100@mm.com	10	M.S.	No Ratings
<input type="radio"/>	dietician101	dietician101@mm.com	10	M.S.	No Ratings
<input type="radio"/>	dietician102	dietician102@mm.com	10	M.S.	No Ratings
<input type="radio"/>	dietician1111	dietician1111@gmail.com	10	M.S.	No Ratings

**Payment Details**

Enter your card number:

Select type of card:

CVV:

Expiry Date:

**SUBMIT**

5. Yes payment successful page: page displayed when details entered by customer is successfully stored in database

localhost:3000/yespaymentsuccessful

Meal Manager

Log Out

**Payment Successful ! Thank You !**

Your meal plan will be ready in the next two days.

6. After registration of new customer customer2222:

Dieticianano page: page displayed when ‘NO, I will not get enrolled with a dietitian’ button is clicked. Details entered after page load.

The screenshot shows a web application titled "Meal Manager" with a "Log Out" link in the top right corner. The main title "Meal Plan Selection" is centered above a table. A sub-instruction "Select your meal plan from the below table:" is displayed above the table. The table has columns: "Select Meal Plan", "Breakfast Count", "Lunch Count", "Dinner Count", and "Price (\$)". The rows show various meal plan configurations and their prices. The last row is highlighted with a red border.

Select Meal Plan	Breakfast Count	Lunch Count	Dinner Count	Price (\$)
0	0	0	15	104.85
0	0	15	0	104.85
0	15	0	0	59.85
0	0	15	15	209.7
0	15	15	0	164.7
0	15	0	15	164.7
0	0	0	30	180
0	0	30	0	180
0	30	0	0	105
0	0	30	30	360
0	30	30	0	285
0	30	0	30	285
0	15	15	15	269.55
0	30	15	15	314.7
0	15	30	15	344.7
0	15	15	30	344.7
0	30	30	15	389.85
0	30	15	30	389.85

localhost:3000/dieticianno?

Meal Manager

Log Out

	30	0	30	285
	15	15	15	269.55
	30	15	15	314.7
	15	30	15	344.7
	15	15	30	344.7
	30	30	15	389.85
	30	15	30	389.85
	15	30	30	419.85
	15	0	30	239.85
	0	15	30	284.85
	15	30	0	239.85
	30	0	15	209.85
	0	30	15	284.85
	30	15	0	209.85
	30	30	30	465

localhost:3000/dieticianno?

Meal Manager

Log Out

Start Date Selection

Select start date for your meal plan: 05/31/2019

Payment Details

Enter your card number: 98765432

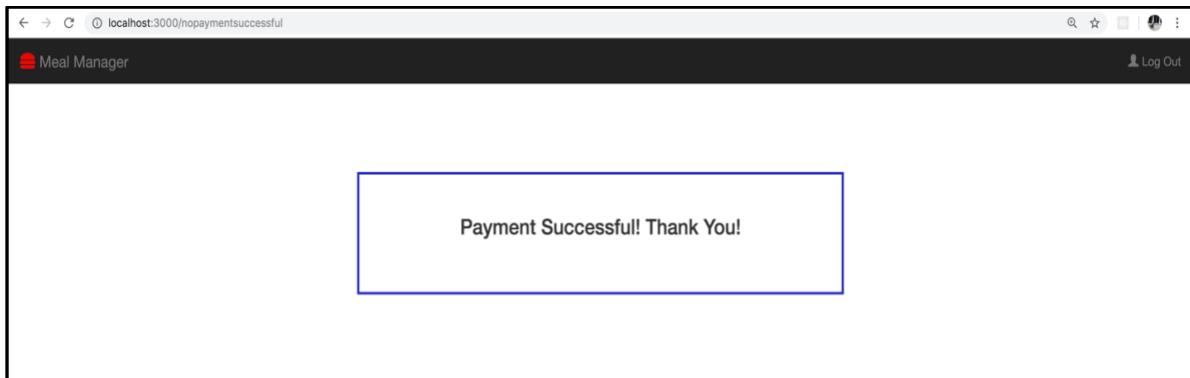
Select type of card: Credit Card

CVV: 111

Expiry Date: 06/29/2019

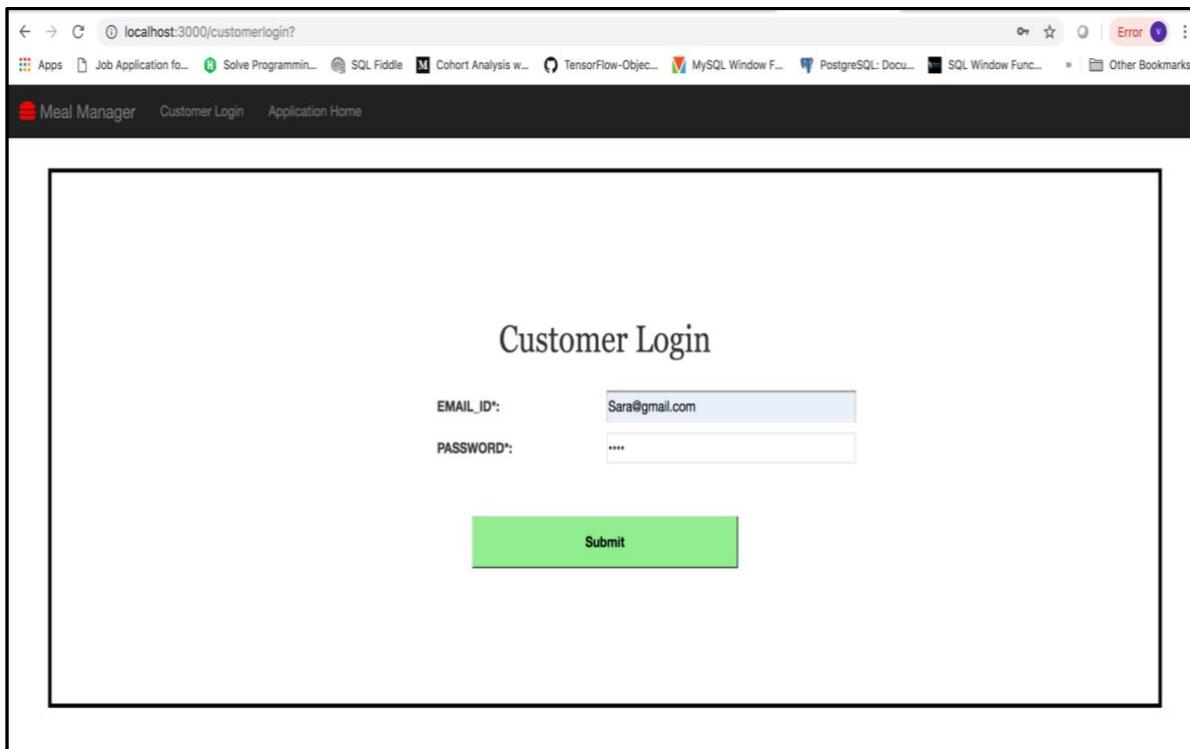
SUBMIT

7. Nopaymentsuccessful page: page displayed when details entered by customer is successfully stored in database

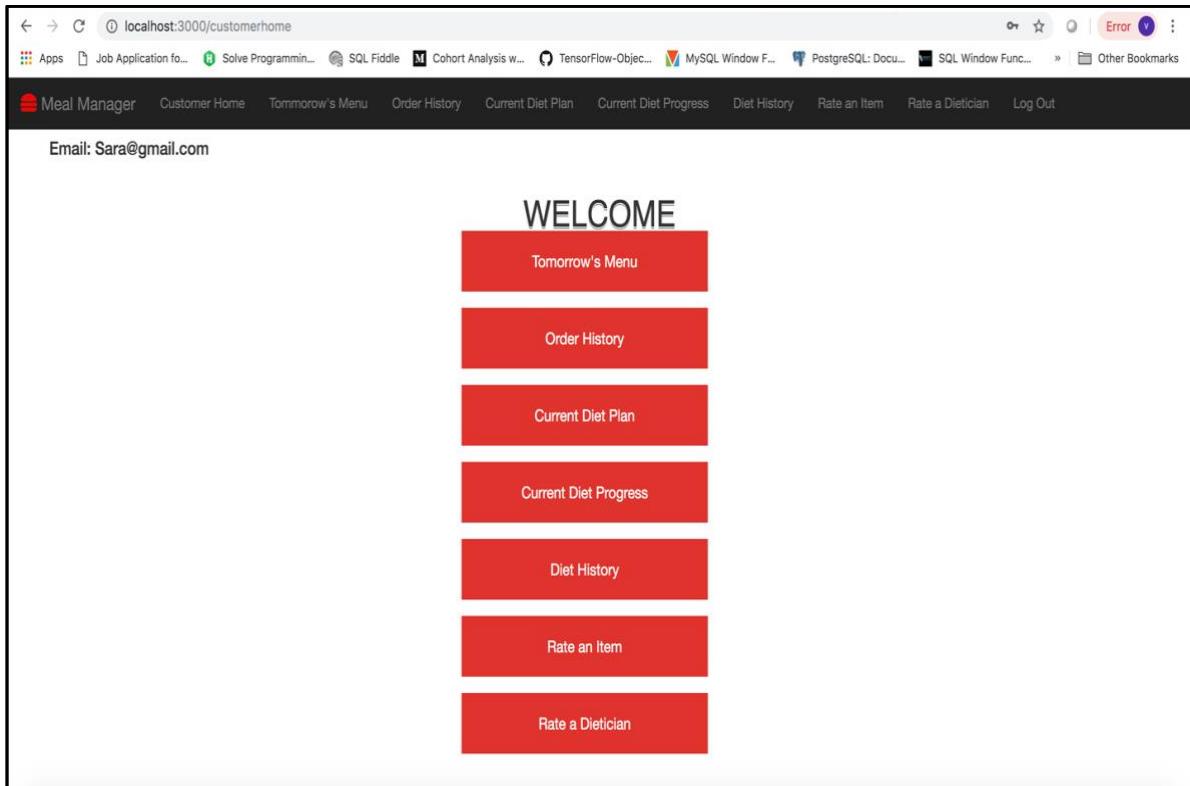


8. Customer Login Page: Customer logs in with email-id [Sara@gmail.com](mailto:Sara@gmail.com) and password "Sara".

Customer Sara is currently enrolled in a regular meal plan but has two diet plans in her past.



9. Customer Home Page after Sara logs in successfully.



10. Click on Tomorrow's Menu button and you will get the below page with the personalized menu for next day. You can select any item you want to order in each table along with the pick-up time slot from drop down and click on place order button. (similar to below three screenshots)

BREAKFAST MENU

Select Option	Restaurant Name	Item Name	Meal Type	Calories	Proteins	Carbohydrates	Fat	Average Rating
<input checked="" type="radio"/>	Indian	Dosa	Breakfast	110	30	30	30	No Ratings
<input type="radio"/>	Malaysian	Roti	Breakfast	110	30	30	30	No Ratings
<input type="radio"/>	Indian	Upma	Breakfast	110	30	30	30	No Ratings

Please choose a pick up time for your breakfast:

- 06.00 AM
- 06.30 AM
- 07.00 AM
- 07.30 AM**
- 08.00 AM
- 08.30 AM
- 09.00 AM
- 09.30 AM
- 10.00 AM
- 10.30 AM

LUNCH MENU

Select Option	Restaurant Name	Item Name	Meal Type	Calories	Proteins	Carbohydrates	Fat	Average Ratings
---------------	-----------------	-----------	-----------	----------	----------	---------------	-----	-----------------

LUNCH MENU

Select Option	Restaurant Name	Item Name	Meal Type	Calories	Proteins	Carbohydrates	Fat	Average Ratings
<input type="radio"/>	Indian	Biryani	Lunch	200	50	100	30	5.0000
<input type="radio"/>	Malaysian	Burmese Rice	Lunch	200	50	100	30	No Ratings
<input type="radio"/>	Malaysian	Pad Thai Noodle	Lunch	150	60	20	30	No Ratings
<input checked="" type="radio"/>	Indian	Paneer Butter Masala	Lunch	150	60	20	30	No Ratings
<input type="radio"/>	Malaysian	Pineapple Biryani	Lunch	200	50	100	30	No Ratings

Please choose a pick up time for your Lunch:

- 11.30 AM

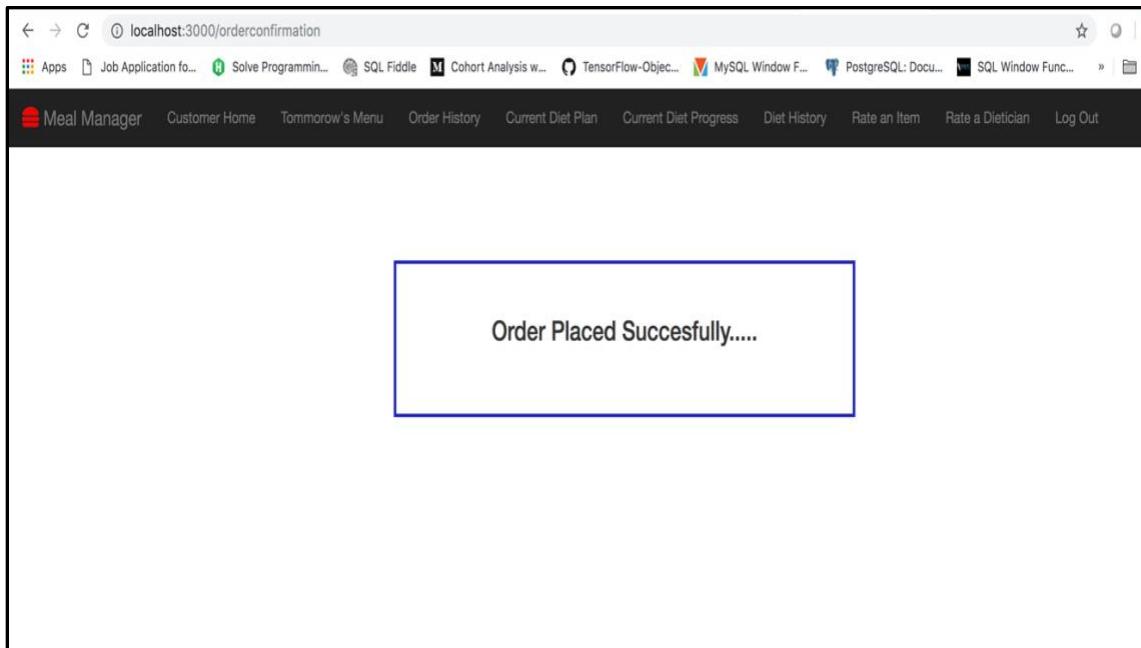
**DINNER MENU**

Select Option	Restaurant Name	Item Name	Meal Type	Calories	Proteins	Carbohydrates	Fat	Average Ratings
<input type="radio"/>	Indian	Chicken Curry	Dinner	150	80	20	30	No Ratings
<input checked="" type="radio"/>	Indian	Naan	Dinner	100	30	50	30	No Ratings
<input type="radio"/>	Malaysian	Tom Yum soup	Dinner	120	60	20	30	5.0000
<input type="radio"/>	Malaysian	Veg Spring Rolls	Dinner	120	30	60	30	4.0000

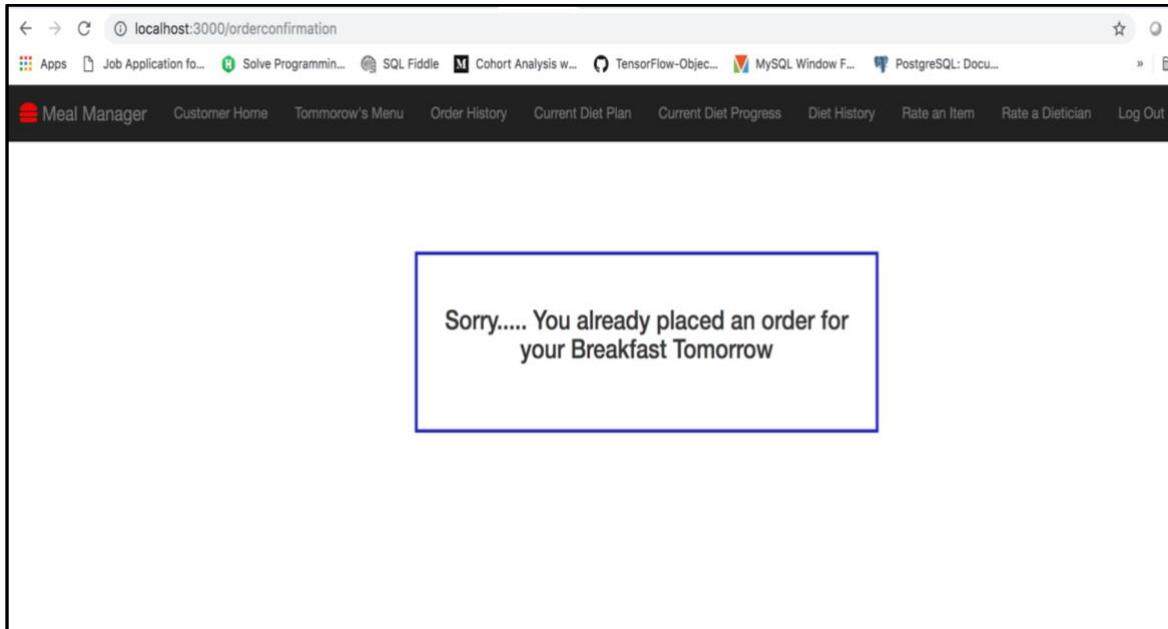
Please choose a pick up time for your Dinner:

**PLACE ORDER**

11. Once clicked on place order button, you get the message that the order is successfully placed.



12. If you go to the Menu page again and try placing an order again for breakfast, the below message will be displayed because one customer can order only one item per meal type each day according to the business rule.

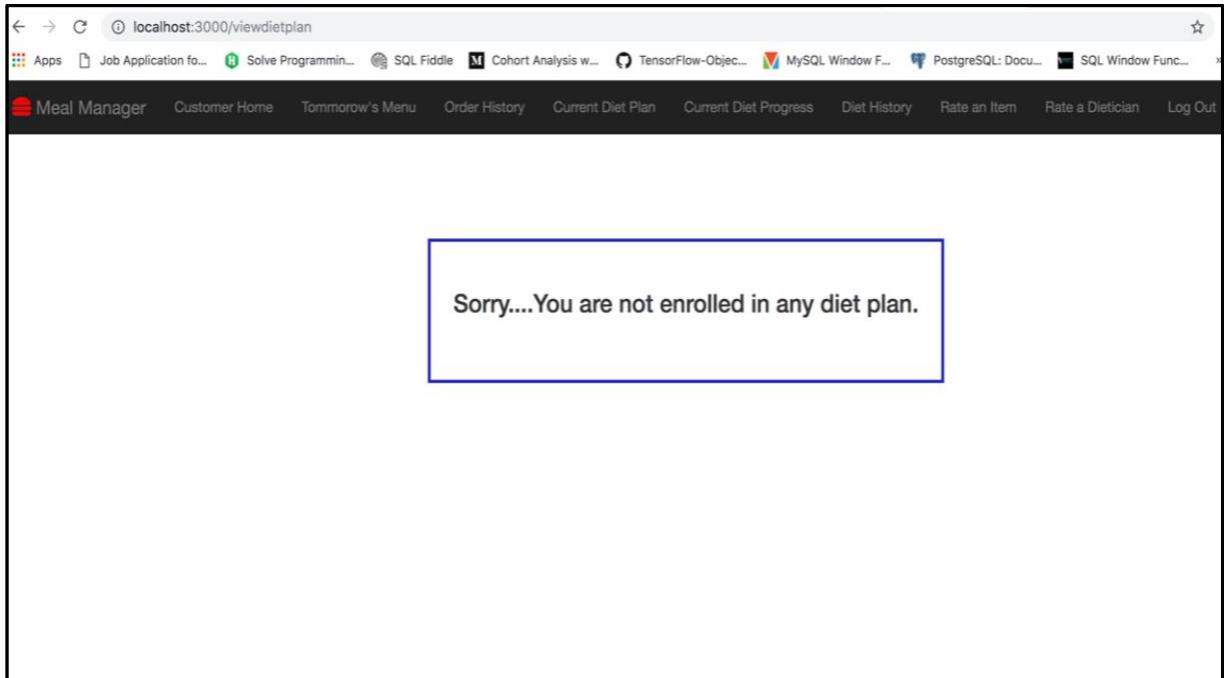


13. Go to home page and click on order history button. You will be able to get all the orders previously placed by Sara.

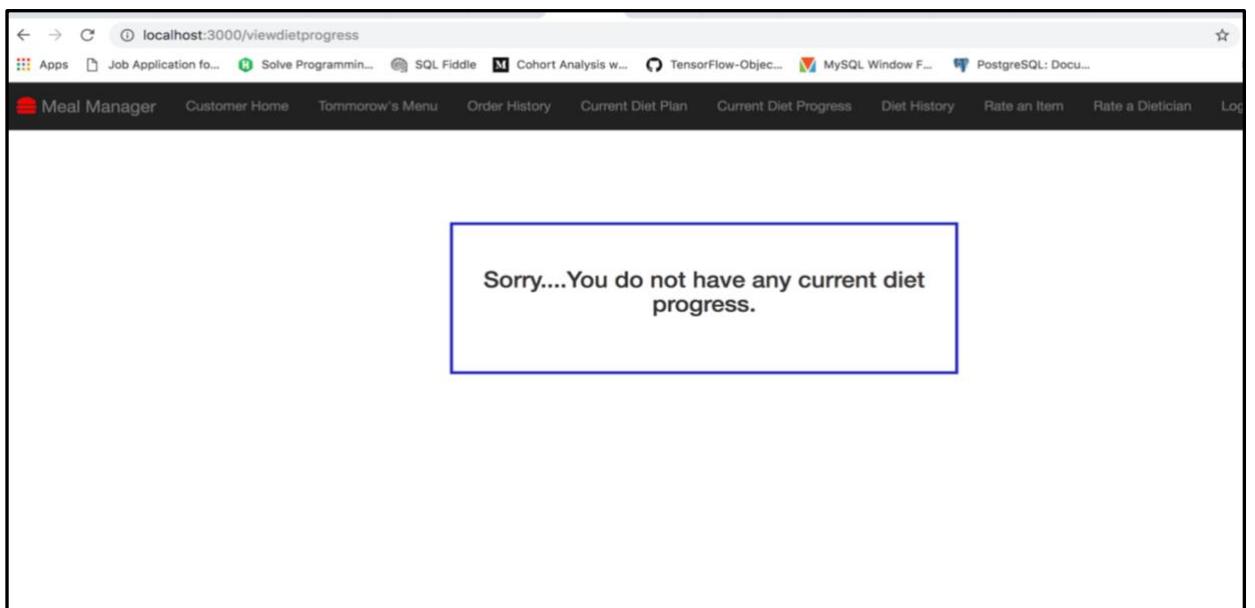
A screenshot of a web browser window titled 'localhost:3000/vieworderhistory'. The browser's address bar shows the same URL. The page title is 'ORDER HISTORY'. Below it is a table with the following data:

Ordered Date	Meal Type	Restaurant Name	Item Name	Day of Week	Pick Up Time
Sun May 05 2019	Dinner	Indian	Naan	Monday	21:00:00
Sun May 05 2019	Lunch	Indian	Paneer Butter Masala	Monday	11:30:00
Sun May 05 2019	Breakfast	Indian	Dosa	Monday	07:30:00

14. If you go to customer home page and click on current diet plan button, you get the below message as Sara is not enrolled with any dietician currently.



15. If you go to customer home page and click on current diet progress button, you get the below message as Sara is not enrolled with any dietician currently.



16. If you go to customer home page and click on diet history button, you get the list of previous diet plans which Sara has.

The screenshot shows a web browser window with the URL `localhost:3000/viewdiethistory`. The page title is "PREVIOUS DIET PLANS". There is a table with four columns: "SelectCustomer", "DieticianName", "Qualification", and "Start Date". The first row has a radio button next to "SelectCustomer", the value "dietician1" in "DieticianName", "M.S." in "Qualification", and "Thu Feb 07 2019 00:00:00 GMT-0800 (Pacific Standard Time)" in "Start Date". The second row has a radio button next to "SelectCustomer", the value "dietician2" in "DieticianName", "M.S." in "Qualification", and "Sat Mar 30 2019 00:00:00 GMT-0700 (Pacific Daylight Time)" in "Start Date". A green "View Details" button is located at the bottom right of the table.

SelectCustomer	DieticianName	Qualification	Start Date
<input type="radio"/>	dietician1	M.S.	Thu Feb 07 2019 00:00:00 GMT-0800 (Pacific Standard Time)
<input checked="" type="radio"/>	dietician2	M.S.	Sat Mar 30 2019 00:00:00 GMT-0700 (Pacific Daylight Time)

17. You can select one diet plan and click on view details button to get the complete information. (Shown in screenshot below)

The screenshot shows a web browser window with the URL `localhost:3000/viewdiethistorydetails`. The page title is "DIET PLAN COMPLETE DETAILS". There is a table with eight columns: "Day", "Meal Type", "Calories", "Proteins", "Carbohydrates", "Fat", "Followed", and "Additional Intake". The table contains 15 rows representing meals for five days. The last row shows an additional item, "Apple Pie".

Day	Meal Type	Calories	Proteins	Carbohydrates	Fat	Followed	Additional Intake
1	Breakfast	100	20	10	20	Y	
1	Dinner	100	35	100	40	Y	
1	Lunch	180	20	75	10	Y	
2	Breakfast	100	25	20	30	Y	
2	Dinner	130	45	100	30	N	
2	Lunch	120	30	70	30	N	
3	Breakfast	170	30	30	40		
3	Dinner	160	45	10	35		
3	Lunch	150	30	75	30		
4	Breakfast	150	20	20	10	Y	
4	Dinner	180	45	80	10	Y	
4	Lunch	150	30	70	20	N	
5	Breakfast	170	20	20	30	Y	Apple Pie

18. If you go to customer home page and click on Rate an item button, you get the list of items which Sara has previously ordered. You can select one of the items from this list, then a rating from the dropdown and click on submit rating button.

The screenshot shows a web browser window with the URL `localhost:3000/itemrating`. The page title is "ITEM RATING". Above the table, there is a message: "Please provide the Rating: ". Below the table is a green "Submit Rating" button.

Select Item	Ordered Date	Meal Type	Restaurant Name	Item Name	Day of Week	Pick Up Time
<input type="radio"/>	Sun May 05 2019	Dinner	Indian	Naan	Monday	21:00:00
<input checked="" type="radio"/>	Sun May 05 2019	Lunch	Indian	Paneer Butter Masala	Monday	11:30:00
<input type="radio"/>	Sun May 05 2019	Breakfast	Indian	Dosa	Monday	07:30:00

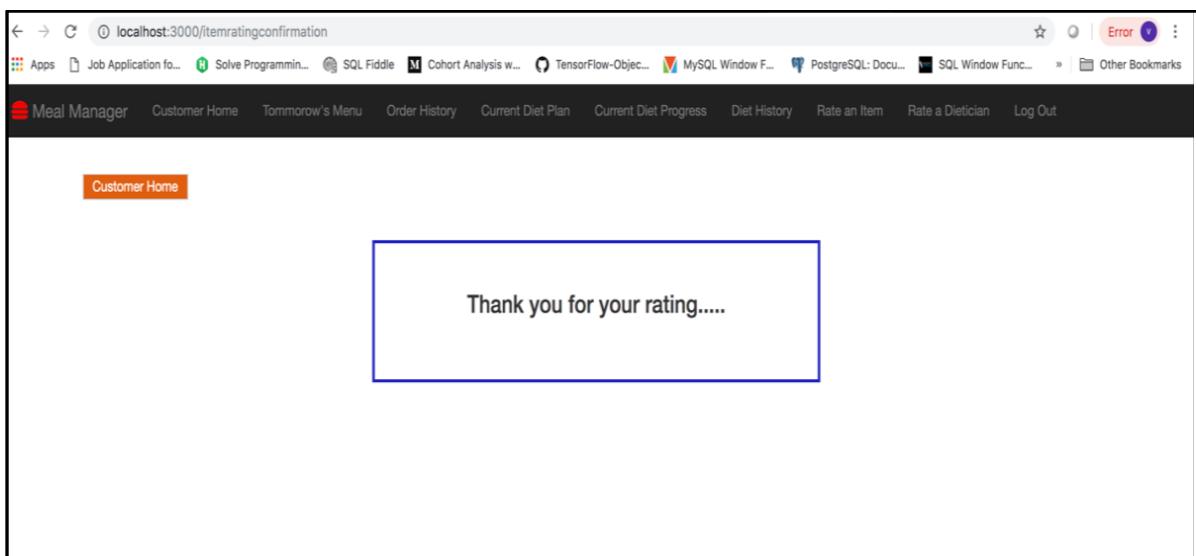
19. You will get the below message indicating success for the operation.

The screenshot shows a web browser window with the URL `localhost:3000/itemratingconfirmation`. The page title is "Customer Home". In the center, there is a blue-bordered box containing the text "Thank you for your rating.....".

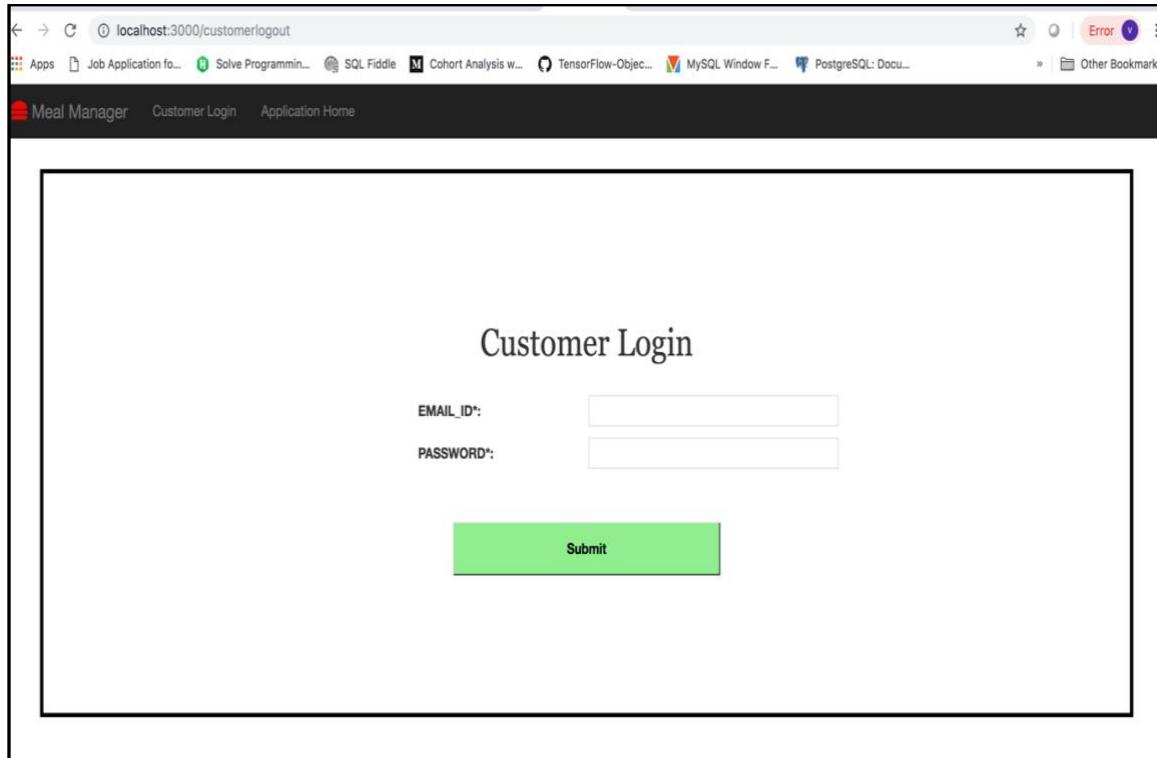
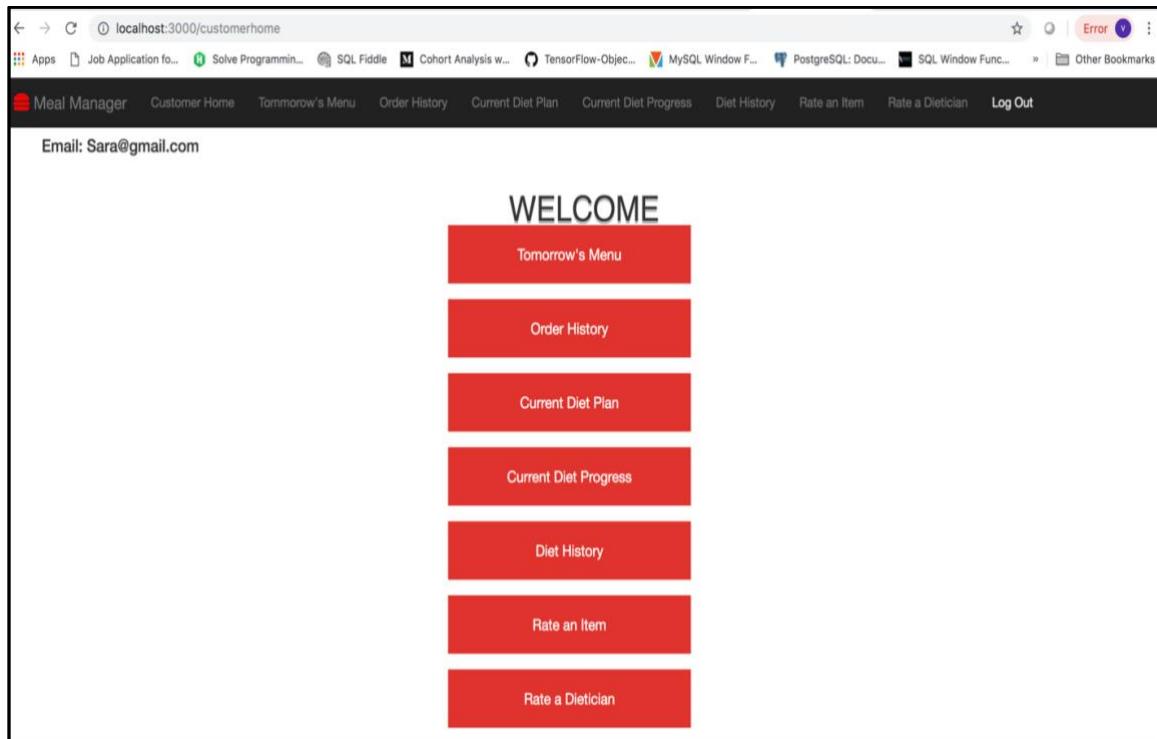
20. If you go to customer home page and click on Rate a Dietician button, you get the list of dieticians to whom Sara has previously enrolled. You can select one of the dieticians from this list, then a rating from the dropdown and click on submit rating button.

The screenshot shows a web browser window with the URL `localhost:3000/dieticianrating`. The page title is "DIETICIAN RATING". A table lists two dieticians: "dietician2" and "dietician1". The "Select Dietician" column contains radio buttons, with "dietician1" selected. The "DieticianName" column shows "dietician2" and "dietician1". The "Qualification" column shows "M.S." for both. The "Start Date" column shows "Sat Mar 30 2019 00:00:00 GMT-0700 (Pacific Daylight Time)" for dietician2 and "Thu Feb 07 2019 00:00:00 GMT-0800 (Pacific Standard Time)" for dietician1. Below the table is a dropdown menu labeled "Please provide the Rating: 3 ▾". A green "Submit Rating" button is at the bottom right.

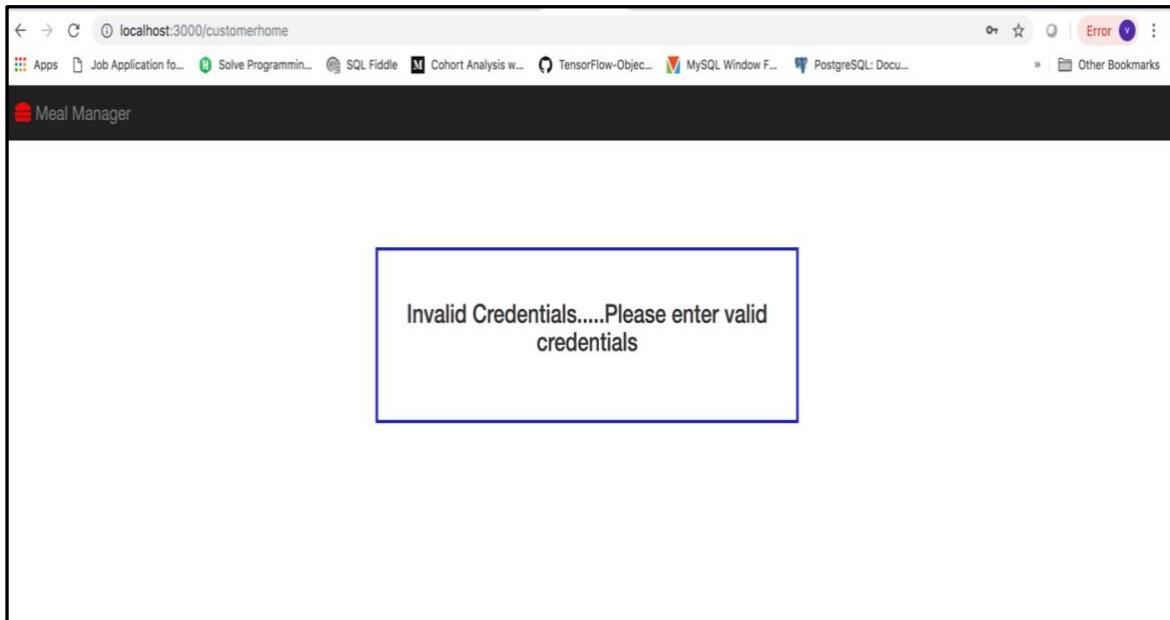
21. You will get the below message indicating success for the operation.



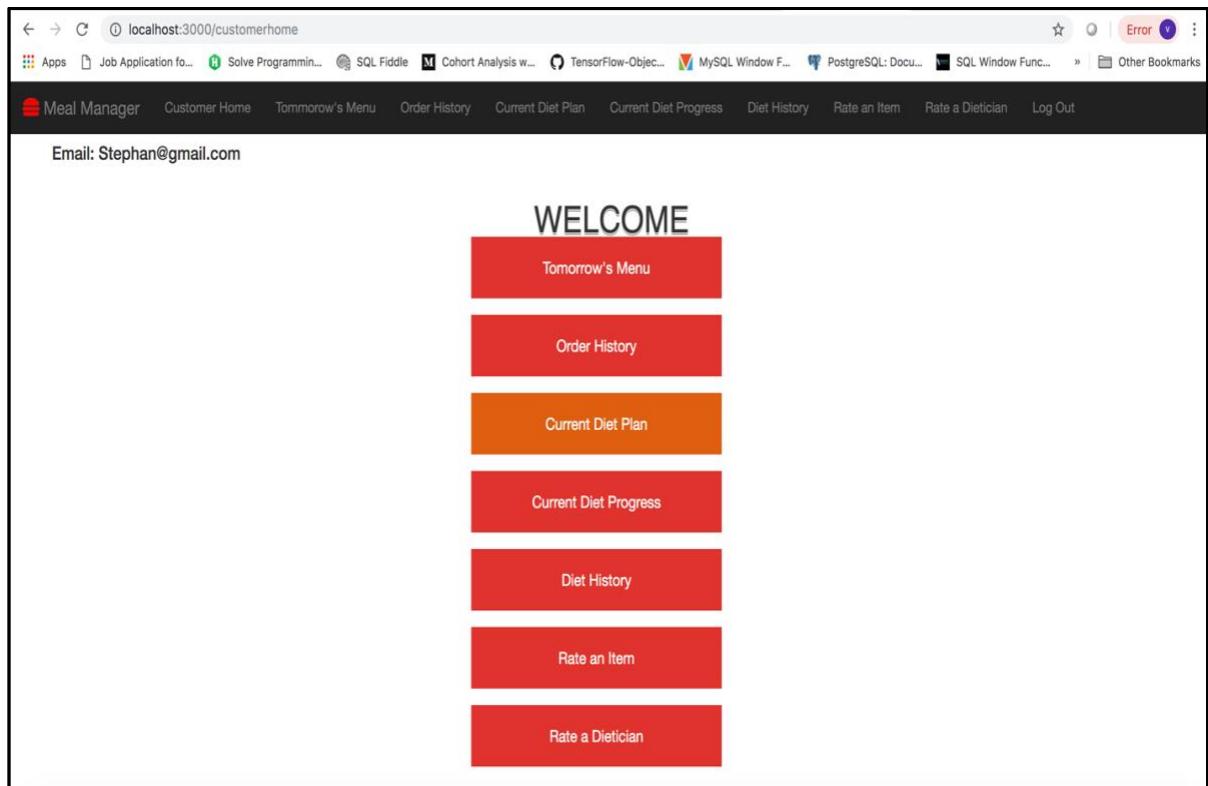
22. If you go to customer home page and click on Log out button you will be logged out and redirected to customer login page again.



23. Now, if you enter email-id as [Sara@gmail.com](mailto:Sara@gmail.com) and password as “sara” (instead of “Sara”), you will get the below message as you entered wrong password.



24. If you go to customer login page and enter email-id as [Stephan@gmail.com](mailto:Stephan@gmail.com) and password as "Stephan" you will enter the customer home page. Customer Stephan is a dietician enrolled person.



25. If you click on Current Diet Plan button, you will get the below page with the current diet plan details for Stephan. Now, if you want to request dietician to make some change in your current diet plan, you can click on Request Change of Diet Plan button in the top left corner of the page.

The screenshot shows a web browser window with the URL `localhost:3000/viewdietplan?` in the address bar. The browser interface includes standard navigation buttons and a toolbar with various bookmarks. The main content area is titled "CURRENT DIET PLAN DETAILS". On the left, there is a red button labeled "Request Change Of Diet Plan". The central part of the page contains a table with the following data:

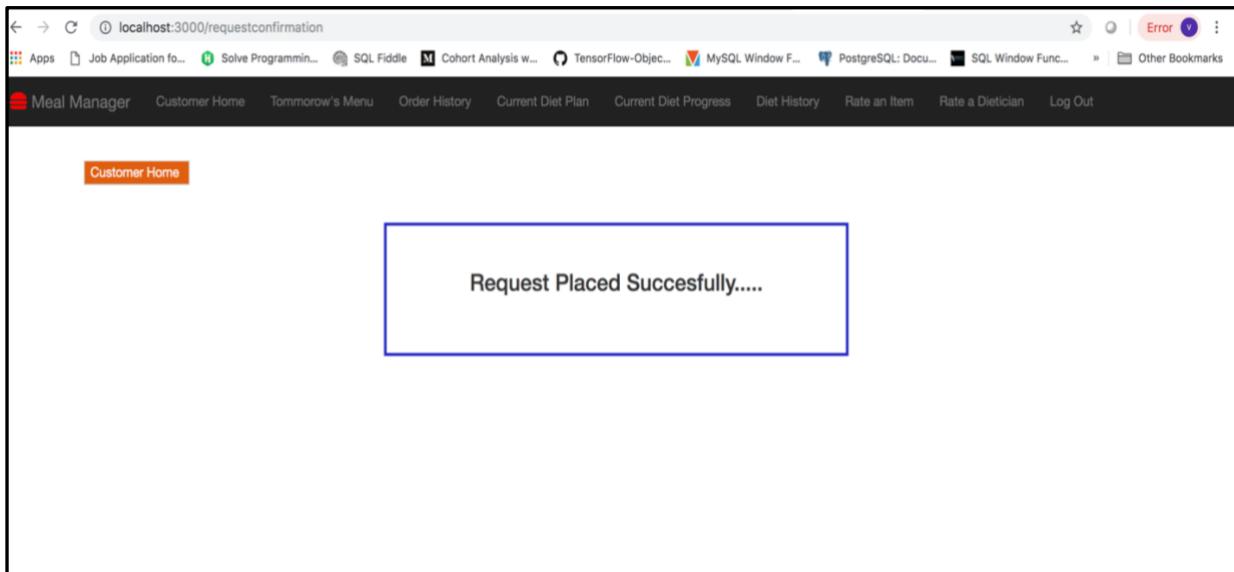
Day	Meal Type	Calories	Proteins	Carbohydrates	Fat
1	Breakfast	100	20	10	20
1	Dinner	100	35	100	40
1	Lunch	180	20	75	10
2	Breakfast	100	25	20	30
2	Dinner	130	45	100	30
2	Lunch	120	30	70	30
3	Breakfast	170	30	30	40
3	Dinner	160	45	10	35
3	Lunch	150	30	75	30
4	Breakfast	150	20	20	10
4	Dinner	180	45	80	10
4	Lunch	150	30	70	20
5	Breakfast	170	20	20	30

26. You will get the below form in which you have to enter the date from which you want to start your new diet plan and also you should provide some description on the change you want to have. Once you enter the date and description, you can click on submit button.

The screenshot shows a web browser window with the URL `localhost:3000/reqchangedp?`. The page title is "Meal Manager". The main content area has a heading "PLEASE INPUT YOUR REQUEST IN DETAIL BELOW". Inside this area, there is a form with a blue border. The form contains the following fields:

- A text input field labeled "From Date: 05/09/2019".
- A note "(From Date should be atleast 3 days from now)".
- A text input field labeled "Description: Please increase carbs in".
- A red "Submit" button.

27. You will get the below message indicating that the request is placed successfully.



28. If you go to customer home page and click on current diet progress button, you will get a new page with the data Stephan has logged previously. If you want to add new diet progress, you can click on Update Diet Progress button on top left corner of the page.

Day	Breakfast	Lunch	Dinner	Additional Intake
1	Y	Y	Y	
2	Y	Y	N	1 Slice Cake

29. You can enter the date for which you want to update your progress, then update if you have followed the diet for each meal type. Then click on submit button.

Please Enter the date(YYYY-MM-DD) for which you want to log in your progress: 2019-05-03

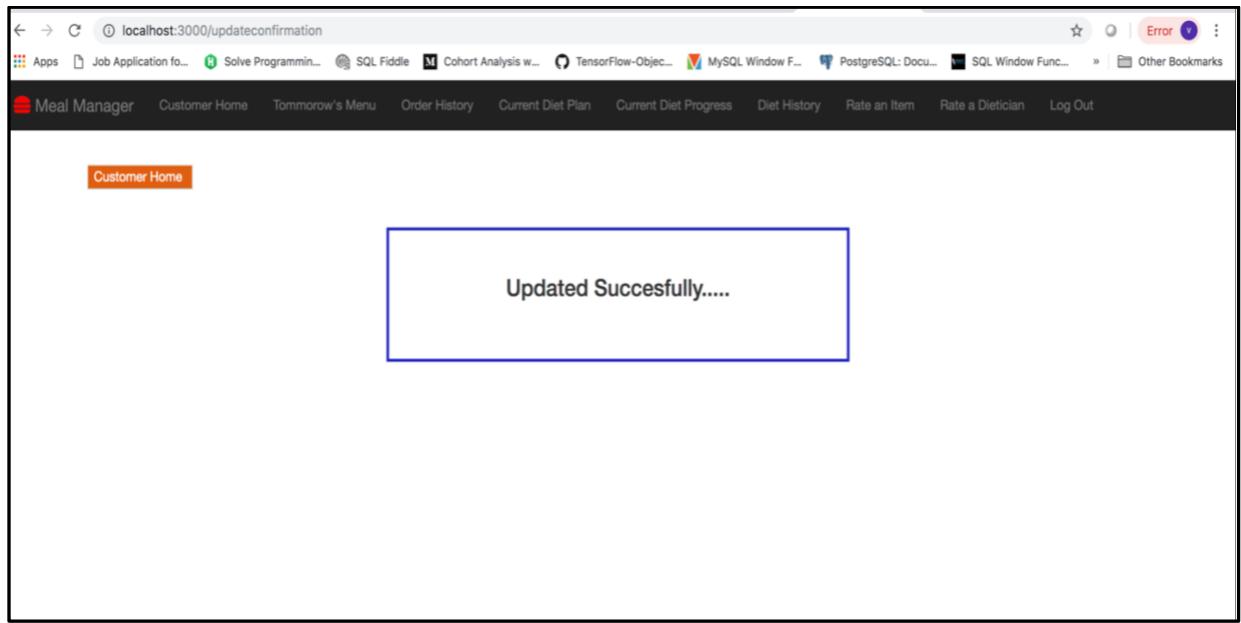
Did You Follow Your Diet For Breakfast: Yes

Did You Follow Your Diet For Lunch: No

Did You Follow Your Diet For Dinner: Yes

If You Have Taken Any Additional Intake, Please Mention: 1 Candy

30. Once you click on submit button, you get the below message indicating that your update is successfully done.



### 7.1.3. Dietician

1. Dietician Login Page:

Login Page: <http://localhost:3000/dieticianlogin>

2. Dietician Login Page: Dietician logs in with email-id [kate@gmail.com](mailto:kate@gmail.com) and password “Kate”.

Please Enter Your Userid and Password!

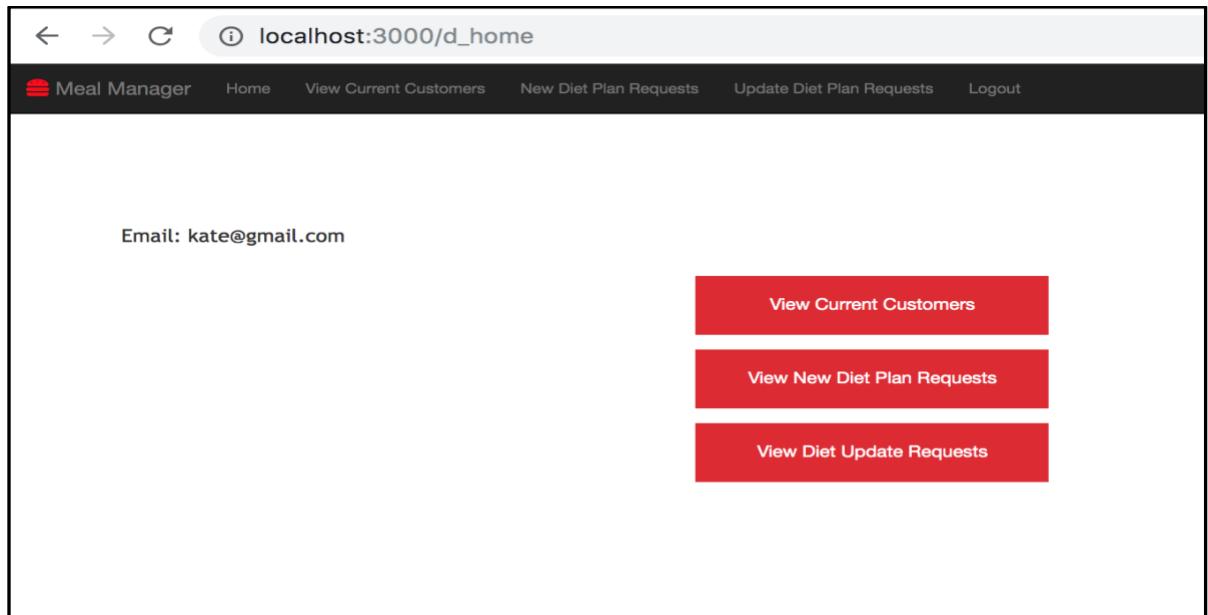
Dietician Login

EMAILID\*: kate@gmail.com

PASSWORD\*: ....

Submit

3. Dietician home page when dietician Kate logs in successfully.



4. List of current customers when clicked on “View Current Customers” button on home page or navigation bar.

A screenshot of a web browser showing the 'All Customers' page at localhost:3000/d\_view\_current\_customers. The page has a dark header bar with the 'Meal Manager' logo and navigation links: Home, View Current Customers, New Diet Plan Requests, Update Diet Plan Requests, and Logout. The main content area displays a table titled 'All Customers' with five columns: SelectCustomer, CustomerId, CustomerName, DietStartDate, and DietRequestDate. The table contains three rows of data. A large orange button labeled 'View Diet Plan' is positioned below the table.

SelectCustomer	CustomerId	CustomerName	DietStartDate	DietRequestDate
<input checked="" type="radio"/>	1021	James	Mon Apr 08 2019	Fri Apr 05 2019
<input type="radio"/>	1021	James	Wed May 08 2019	Sun May 05 2019
<input type="radio"/>	1022	Joseph	Thu Apr 25 2019	Mon Apr 22 2019

**View Diet Plan**

5. When “View Diet Plan” button is clicked for James, list of diet plan for James is displayed.

The screenshot shows a web browser window with the URL `localhost:3000/d_customer_diet_plan_list`. The page title is "Meal Manager". The navigation menu includes "Home", "View Current Customers", "New Diet Plan Requests", "Update Diet Plan Requests", and "Logout". Below the menu, the text "CustID:1021 CustName:James CustEmailID:James@gmail.com" is displayed. The main content area is titled "Customer Diet Plans List". It contains a table with three rows:

Select_Diet_Plan	DietRequestDate
<input type="radio"/>	Sat Mar 09 2019
<input checked="" type="radio"/>	Mon Apr 08 2019
<input type="radio"/>	Wed May 08 2019

Below the table is a large orange button labeled "View Diet Plan Progress".

6. When Diet Plan for Apr 8 2019 is selected for James and “view Diet Plan Progress” is clicked, diet plan details and progress can be seen for that particular diet plan. The diet plan given by dietician can be viewed. If James has updated diet plan progress, we can see Y or N in diet\_followed column, otherwise ‘No Data’ is seen. When James has added any additional\_intake we can see the additional\_intake data. This helps dietician monitor the diet plan.

localhost:3000/d\_customer\_diet\_plan\_progress

Meal Manager Home View Current Customers New Diet Plan Requests Update Diet Plan Requests Logout

CustID:1021 CustName:James CustEmail:D.James@gmail.com

**Customer Diet Plan & Progress Details**

CalendarDate	meal_type	calories	proteins	carbohydrates	fat	diet_followed	additional_intake
Tue Apr 09 2019	Breakfast	100	20	20	30	Y	
	Lunch	180	20	80	30	Y	
	Dinner	100	40	100	30	Y	
Wed Apr 10 2019	Breakfast	120	20	20	30	Y	
	Lunch	150	30	70	30	N	
	Dinner	130	45	100	30	N	
Thu Apr 11 2019	Breakfast	120	30	30	30	No Data	
	Lunch	150	30	75	30	No Data	
	Dinner	130	45	10	30	No Data	
Fri Apr 12 2019	Breakfast	100	20	20	10	Y	
	Lunch	140	30	70	20	N	
	Dinner	130	45	80	30	Y	
Sat Apr 13 2019	Breakfast	170	20	20	30	Y	
	Lunch	150	35	70	30	Y	
	Dinner	130	45	70	30	N	
Sun Apr 14 2019	Breakfast	120	20	20	10	Y	
	Lunch	150	75	70	20	N	
	Dinner	130	45	60	30	Y	
Mon Apr 15 2019	Breakfast	100	20	20	30	Y	
	Lunch	150	85	70	30	N	
	Dinner	110	45	60	30	Y	
Tue Apr 16 2019	Breakfast	170	35	20	10	No Data	
	Lunch	150	35	65	30	No Data	
	Dinner	130	45	85	30	No Data	

7. When “New Diet Plan Requests” button from home page or link for navigation bar, can see new diet plan requests for dietician Kate.

The screenshot shows a web browser window with the URL `localhost:3000/d_new_diet_plan_requests`. The page title is "New Diet Plan Requests". At the top, there is a navigation bar with links: "Meal Manager", "Home", "View Current Customers", "New Diet Plan Requests" (which is the active link, highlighted in orange), "Update Diet Plan Requests", and "Logout". Below the navigation bar, the main content area displays a table with one row of data. The table has columns: "SelectCustomer", "CustomerId", "CustomerName", "DietPlanStartDate", and "DietPlanRequestDate". The data in the table is:

SelectCustomer	CustomerId	CustomerName	DietPlanStartDate	DietPlanRequestDate
<input checked="" type="radio"/>	1021	James	Wed May 08 2019	Sun May 05 2019

Below the table is a large orange button labeled "Create New Diet Plan".

8. To create or to continue creating the new diet plan for James, “Create New Diet Plan” is clicked. Customer details like age, height, weight, sleep, activity level, allergies and diseases can be seen at the top. The days for which Kate has already entered diet plan details, we can see the diet plan details. If the diet plan details is not entered, then diet day is displayed, diet plan details will not be displayed, for example Day -7.

← → ⌂ ⓘ localhost:3000/d\_new\_diet\_plan\_details

Meal Manager Home View Current Customers New Diet Plan Requests Update Diet Plan Requests Logout

### Diet Plan Details

CustID:1021 CustName:James CustEmailID:James@gmail.com  
 CustAge:25 Height:150 Weight: Sleep:8 hrs ActivityLevel:low  
 Allergy:Lactose Allergy:Peanuts Disease:BP Disease:Diabetes

#### Diet Plan Details

SelectDay	Day	CalendarDate	MealType	Calories	Proteins	Carbohydrates	Fat
			Breakfast	100	20	20	30
	1	Thu May 09 2019	Lunch	180	20	80	30
			Dinner	100	40	100	30
	2	Fri May 10 2019	Breakfast	120	20	20	30
			Lunch	150	30	70	30
			Dinner	130	45	100	30
	3	Sat May 11 2019	Breakfast	120	30	30	30
			Lunch	150	30	75	30
			Dinner	130	45	10	30
	4	Sun May 12 2019	Breakfast	100	20	20	10
			Lunch	140	30	70	20
			Dinner	130	45	80	30
	5	Mon May 13 2019	Breakfast	170	20	20	30
			Lunch	150	35	70	30
			Dinner	130	45	70	30
	6	Tue May 14 2019	Breakfast	120	20	20	10
			Lunch	150	75	70	20
			Dinner	130	45	60	30
	7	Wed May 15 2019	Breakfast				
			Lunch				
			Dinner				
	8	Thu May 16 2019	Breakfast	170	35	20	10
			Lunch	150	35	65	30
			Dinner	130	45	85	30
	9	Fri May 17 2019	Breakfast	120	35	50	10
			Lunch	150	35	75	30
			Dinner	130	25	85	30
	10	Sat May 18 2019	Breakfast	130	35	50	20
			Lunch	140	75	75	20

9. To create diet plan for day 7, select the radio button, and click on “Create New Diet Plan” button shown below.

				Dinner	190	25	80	10		
①	24	Sat Jun 01 2019	Breakfast	120	35	20	10			
			Lunch	100	55	75	20			
			Dinner	100	25	90	10			
			Breakfast	120	65	20	10			
②	25	Sun Jun 02 2019	Lunch	100	55	75	20			
			Dinner	100	25	90	10			
			Breakfast	120	35	70	10			
			Lunch	200	55	75	20			
③	26	Mon Jun 03 2019	Dinner	200	25	80	10			
			Breakfast	120	45	80	10			
			Lunch	200	55	80	20			
			Dinner	160	25	90	10			
④	27	Tue Jun 04 2019	Breakfast	150	35	20	10			
			Lunch	100	55	55	20			
			Dinner	150	25	80	10			
			Breakfast	120	35	20	10			
⑤	28	Wed Jun 05 2019	Lunch	200	45	75	20			
			Dinner	100	55	90	20			
			Breakfast	120	35	20	10			
			Lunch	200	45	75	20			
⑥	29	Thu Jun 06 2019	Dinner	100	35	90	20			
			Breakfast	120	35	20	10			
			Lunch	200	85	75	10			
			Dinner	100	35	90	10			
			Create New Diet Plan							
			Finalize Diet Plan							

10. Add Diet Plan details is displayed for day-7. Fill the details and click on “Add Diet Plan Details” button.

The screenshot shows a web browser window with the URL `localhost:3000/d_add_diet_plan_details`. The page is titled "Meal Manager". The main content area displays customer information: CustID:1021, CustName:James, CustEmailID:James@gmail.com, CustAge:25, Height:150, Weight:, Sleep:8 hrs, ActivityLevel:low, Allergy:Lactose, Allergy:Peanuts, Disease:BP, Disease:Diabetes. Below this, the title "Add Diet Plan Details" is centered. A yellow box contains three sections: "Breakfast", "Lunch", and "Dinner", each with four input fields for Calories, Proteins, Carbohydrates, and Fat. The "Fat\*" field for Dinner is currently set to 33 and has a dropdown arrow. At the bottom of the yellow box is a button labeled "Add Diet Plan Details".

CustID:1021 CustName:James CustEmailID:James@gmail.com  
CustAge:25 Height:150 Weight: Sleep:8 hrs ActivityLevel:low  
Allergy:Lactose Allergy:Peanuts Disease:BP Disease:Diabetes

## Add Diet Plan Details

**Breakfast**

Calories*:	201
Proteins*:	11
Carbohydrates*:	12
Fat*:	13

**Lunch**

Calories*:	301
Proteins*:	21
Carbohydrates*:	22
Fat*:	23

**Dinner**

Calories*:	401
Proteins*:	31
Carbohydrates*:	32
Fat*:	33

Add Diet Plan Details

11. Diet plan details for Day-7 is added, we can see the success message at the top. Also, for Day-7, the diet plan details can be seen.

The screenshot shows a web application interface for managing meal plans. At the top, there is a navigation bar with links for 'Meal Manager', 'Home', 'View Current Customers', 'New Diet Plan Requests', 'Update Diet Plan Requests', and 'Logout'. Below the navigation bar, a success message 'Diet Plan Details Added Successfully!' is displayed in red. Following this, there is a block of text providing customer details: 'CustID:1021 CustName:James CustEmailID:James@gmail.com', 'CustAge:25 Height:150 Weight: Sleep:8 hrs ActivityLevel:low', and 'Allergy:Lactose Allergy:Peanuts Disease:BP Disease:Diabetes'. A section titled 'Diet Plan Details' contains a table with data for 17 rows (9 days). The table has columns for 'SelectDay', 'Day', 'CalendarDate', 'MealType', 'Calories', 'Proteins', 'Carbohydrates', and 'Fat'. The data is summarized below:

SelectDay	Day	CalendarDate	MealType	Calories	Proteins	Carbohydrates	Fat
1	1	Thu May 09 2019	Breakfast	100	20	20	30
			Lunch	180	20	80	30
			Dinner	100	40	100	30
2	2	Fri May 10 2019	Breakfast	120	20	20	30
			Lunch	150	30	70	30
			Dinner	130	45	100	30
3	3	Sat May 11 2019	Breakfast	120	30	30	30
			Lunch	150	30	75	30
			Dinner	130	45	10	30
4	4	Sun May 12 2019	Breakfast	100	20	20	10
			Lunch	140	30	70	20
			Dinner	130	45	80	30
5	5	Mon May 13 2019	Breakfast	170	20	20	30
			Lunch	150	35	70	30
			Dinner	130	45	70	30
6	6	Tue May 14 2019	Breakfast	120	20	20	10
			Lunch	150	75	70	20
			Dinner	130	45	60	30
7	7	Wed May 15 2019	Breakfast	201	11	12	13
			Lunch	301	21	22	23
			Dinner	401	31	32	33
8	8	Thu May 16 2019	Breakfast	170	35	20	10
			Lunch	150	35	65	30
			Dinner	130	45	85	30
9	9	Fri May 17 2019	Breakfast	120	35	50	10
			Lunch	150	35	75	30
			Dinner	130	25	85	30
			Breakfast	130	35	50	20

12. If not happy with the diet plan for any day, select the day and click on “Create new diet plan” button at the bottom of the page. Let’s update for Day-7. Diet plan details for day-7 is prefilled. Modify the details as required, and click on “Add Diet Plan Details” button.

The screenshot shows a web browser window with the URL `localhost:3000/d_add_diet_plan_details`. The page is titled "Add Diet Plan Details". At the top, there is a header bar with the "Meal Manager" logo and links for "Home", "View Current Customers", "New Diet Plan Requests", "Update Diet Plan Requests", and "Logout". Below the header, there is some user information: CustID:1021, CustName:James, CustEmailID:James@gmail.com, CustAge:25, Height:150, Weight:, Sleep:8 hrs, ActivityLevel:low, Allergy:Lactose, Allergy:Peanuts, Disease:BP, and Disease:Diabetes.

The main content area is a yellow box containing three sections: Breakfast, Lunch, and Dinner. Each section has four input fields for Calories, Proteins, Carbohydrates, and Fat, each with a value of 101, 11, 12, and 13 respectively. At the bottom of the yellow box is a button labeled "Add Diet Plan Details".

Calories*	201
Proteins*	11
Carbohydrates*	12
Fat*	13

Calories*	301
Proteins*	21
Carbohydrates*	22
Fat*	23

Calories*	401
Proteins*	31
Carbohydrates*	32
Fat*	33

[localhost:3000/d\\_add\\_diet\\_plan\\_details](http://localhost:3000/d_add_diet_plan_details)

Meal Manager [Home](#) [View Current Customers](#) [New Diet Plan Requests](#) [Update Diet Plan Requests](#) [Logout](#)

CustID:1021 CustName:James CustEmailID:James@gmail.com  
CustAge:25 Height:150 Weight: Sleep:8 hrs ActivityLevel:low  
Allergy:Lactose Allergy:Peanuts Disease:BP Disease:Diabetes

## Add Diet Plan Details

<b>Breakfast</b>	
Calories*:	202
Proteins*:	14
Carbohydrates*:	15
Fat*:	16
<b>Lunch</b>	
Calories*:	302
Proteins*:	24
Carbohydrates*:	25
Fat*:	26
<b>Dinner</b>	
Calories*:	402
Proteins*:	34
Carbohydrates*:	35
Fat*:	36

[Add Diet Plan Details](#)

13. Diet plan details are successfully added. Success message with modified diet plan day 7 can be seen.

Diet Plan Details Added Successfully!

CustID:1021 CustName:James CustEmailID:James@gmail.com  
 CustAge:25 Height:150 Weight: Sleep:8 hrs ActivityLevel:low  
 Allergy:Lactose Allergy:Peanuts Disease:BP Disease:Diabetes

SelectDay	Day	CalendarDate	MealType	Calories	Proteins	Carbohydrates	Fat
	1	Thu May 09 2019	Breakfast	100	20	20	30
			Lunch	180	20	80	30
			Dinner	100	40	100	30
	2	Fri May 10 2019	Breakfast	120	20	20	30
			Lunch	150	30	70	30
			Dinner	130	45	100	30
	3	Sat May 11 2019	Breakfast	120	30	30	30
			Lunch	150	30	75	30
			Dinner	130	45	10	30
	4	Sun May 12 2019	Breakfast	100	20	20	10
			Lunch	140	30	70	20
			Dinner	130	45	80	30
	5	Mon May 13 2019	Breakfast	170	20	20	30
			Lunch	150	35	70	30
			Dinner	130	45	70	30
	6	Tue May 14 2019	Breakfast	120	20	20	10
			Lunch	150	75	70	20
			Dinner	130	45	60	30
	7	Wed May 15 2019	Breakfast	202	14	15	16
			Lunch	302	24	25	26
			Dinner	402	34	35	36
	8	Thu May 16 2019	Breakfast	170	35	20	10
			Lunch	150	35	65	30
			Dinner	130	45	85	30
	9	Fri May 17 2019	Breakfast	120	35	50	10
			Lunch	150	35	75	30
			Dinner	130	25	85	30

14. If the dietician is happy with diet plan for all 30 days, click on “Finalize Diet Plan” at the bottom of the page. This will finalize the diet plan, and the request will be removed from new diet plans list.

				Dinner	190	25	80	10
	24	Sat Jun 01 2019	Breakfast	120	35	20		10
			Lunch	100	55	75		20
			Dinner	100	25	90		10
	25	Sun Jun 02 2019	Breakfast	120	65	20		10
			Lunch	100	55	75		20
			Dinner	100	25	90		10
	26	Mon Jun 03 2019	Breakfast	120	35	70		10
			Lunch	200	55	75		20
			Dinner	200	25	80		10
	27	Tue Jun 04 2019	Breakfast	120	45	80		10
			Lunch	200	55	80		20
			Dinner	160	25	90		10
	28	Wed Jun 05 2019	Breakfast	150	35	20		10
			Lunch	100	55	55		20
			Dinner	150	25	80		10
	29	Thu Jun 06 2019	Breakfast	120	35	20		10
			Lunch	200	45	75		20
			Dinner	100	55	90		20
	30	Fri Jun 07 2019	Breakfast	120	35	20		10
			Lunch	200	85	75		10
			Dinner	100	35	90		10

[Create New Diet Plan](#)
|
[Finalize Diet Plan](#)

15. We get the following success message.

The screenshot shows a web browser window with the URL `localhost:3000/d_finalize_new_diet_plan`. The page title is "Meal Manager". The main content area displays a red success message: "Diet plan for the following customer has been finalized!". Below this message, there is some technical information: "CustID:1021 CustName:James CustEmailID:James@gmail.com" and "CustAge:25 Height:150 Weight: Sleep:8 hrs ActivityLevel:low".

16. When “New Diet Plan Requests” is clicked again, the new diet plan request from James will be removed. Also, since, Kate does not have any other new diet plan requests, “No New Diet Plan Requests” message will be displayed.

The screenshot shows a web browser window with the URL `localhost:3000/d_new_diet_plan_requests`. The page title is "Meal Manager". The main content area displays a message: "New Diet Plan Requests" followed by "No new diet plan requests!" in red text.

17. Update diet plan requests can be viewed from “Update Diet Plan Requests” button on home page or from link in navigation bar. This can be used to either update or continue updating a diet plan.

localhost:3000/d\_update\_diet\_plan\_requests

Meal Manager Home View Current Customers New Diet Plan Requests Update Diet Plan Requests Logout

### Update Diet Plan Requests

SelectCustomer	CustomerId	CustomerName	DietPlanStartDate	DietPlanUpdateStartDate	RequestContent
<input checked="" type="radio"/>	1022	Joseph	Thu Apr 25 2019	Sat May 04 2019	Could not follow diet

Update Diet Plan

18. Kate has 1 update diet plan request from Joseph. The diet start date was Apr 25<sup>th</sup> 2019. He needs the plan to be updated from May 4<sup>th</sup> 2019. When the radio button is selected and “Update Diet Plan” button is clicked, the diet plan details from May 4<sup>th</sup> 2019 till the end of diet plan will be displayed.

← → ⌂ ⓘ localhost:3000/d\_update\_diet\_plan\_details

 Meal Manager   [Home](#)   [View Current Customers](#)   [New Diet Plan Requests](#)   [Update Diet Plan Requests](#)   [Logout](#)

## Diet Plan Details

CustID:1022 CustName:Joseph CustEmailID:Joseph@gmail.com  
 CustAge:33 Height:150 Weight: Sleep:6 hrs ActivityLevel:low  
 Allergy:Lactose Allergy:Peanuts Disease:BP Disease:Diabetes

### Diet Plan Details

SelectDay	Day	CalendarDate	MealType	Calories	Proteins	Carbohydrates	Fat
①	10	Sat May 04 2019	Breakfast	130	35	50	20
			Lunch	140	75	75	20
			Dinner	120	25	95	30
②	11	Sun May 05 2019	Breakfast	120	35	20	10
			Lunch	150	35	75	20
			Dinner	130	25	90	30
③	12	Mon May 06 2019	Breakfast	100	25	20	20
			Lunch	200	35	55	20
			Dinner	200	85	90	30

19. To update the diet plan for May 4<sup>th</sup> 2019, select the corresponding button and click on the “Update Diet Plan” button at the bottom of the page.

The screenshot shows a web-based application titled "Meal Manager". The URL in the browser is "localhost:3000/d\_update\_diet\_plan\_details". The interface includes a navigation bar with links for "Home", "View Current Customers", "New Diet Plan Requests", "Update Diet Plan Requests", and "Logout". Below the navigation bar is a table representing a diet plan for the month of May 2019. The table has columns for Date (Day, Month Year), Breakfast, Lunch, Dinner, and nutritional values (190, 25, 80, 10). The rows are grouped by date: May 18, 19, 20, 21, 22, 23, and 24. The row for May 4 is not visible in the current view. At the bottom of the table are two buttons: "Update Diet Plan" (orange) and "Finalize Diet Plan".

				Dinner	190	25	80	10	
●	24	Sat May 18 2019	Breakfast	120	35	20		10	
			Lunch	100	55	75		20	
			Dinner	100	25	90		10	
●	25	Sun May 19 2019	Breakfast	120	65	20		10	
			Lunch	100	55	75		20	
			Dinner	100	25	90		10	
●	26	Mon May 20 2019	Breakfast	120	35	70		10	
			Lunch	200	55	75		20	
			Dinner	200	25	80		10	
●	27	Tue May 21 2019	Breakfast	120	45	80		10	
			Lunch	200	55	80		20	
			Dinner	160	25	90		10	
●	28	Wed May 22 2019	Breakfast	150	35	20		10	
			Lunch	100	55	55		20	
			Dinner	150	25	80		10	
●	29	Thu May 23 2019	Breakfast	120	35	20		10	
			Lunch	200	45	75		20	
			Dinner	100	55	90		20	
●	30	Fri May 24 2019	Breakfast	120	35	20		10	
			Lunch	200	85	75		10	
			Dinner	100	35	90		10	

[Update Diet Plan](#) | [Finalize Diet Plan](#)

20. Update diet plan form for May 4<sup>th</sup> along with existing diet plan details is fetched.

The screenshot shows a web application interface for 'Meal Manager'. At the top, there is a header bar with navigation icons (back, forward, search), the URL 'localhost:3000/d\_update\_diet\_plan\_details\_form', and a logo labeled 'Meal Manager'. Below the header, a dark navigation bar contains links for 'Home', 'View Current Customers', 'New Diet Plan Requests', 'Update Diet Plan Requests', and 'Logout'. The main content area displays customer information: CustID:1022, CustName:Joseph, CustEmailID:Joseph@gmail.com, CustAge:33, Height:150, Weight: Sleep:6 hrs, ActivityLevel:low, Allergy:Lactose, Allergy:Peanuts, Disease:BP, and Disease:Diabetes. Below this information, a large yellow box titled 'Update Diet Plan Details' contains three sections for meal planning: 'Breakfast', 'Lunch', and 'Dinner'. Each section includes four input fields for 'Calories\*', 'Proteins\*', 'Carbohydrates\*', and 'Fat\*'. The values entered are: Breakfast (Calories: 130, Proteins: 35, Carbohydrates: 50, Fat: 20); Lunch (Calories: 140, Proteins: 75, Carbohydrates: 75, Fat: 20); and Dinner (Calories: 120, Proteins: 25, Carbohydrates: 95, Fat: 30). A blue 'Update Diet Plan Details' button is located at the bottom of the yellow box.

CustID:1022 CustName:Joseph CustEmailID:Joseph@gmail.com  
CustAge:33 Height:150 Weight: Sleep:6 hrs ActivityLevel:low  
Allergy:Lactose Allergy:Peanuts Disease:BP Disease:Diabetes

## Update Diet Plan Details

**Breakfast**

Calories*:	130
Proteins*:	35
Carbohydrates*:	50
Fat*:	20

**Lunch**

Calories*:	140
Proteins*:	75
Carbohydrates*:	75
Fat*:	20

**Dinner**

Calories*:	120
Proteins*:	25
Carbohydrates*:	95
Fat*:	30

**Update Diet Plan Details**

21. Update the diet plan details and click on “Update Diet Plan Details” button.

The screenshot shows a web browser window with the URL `localhost:3000/d_update_diet_plan_details_form`. The page title is "Meal Manager". The main content area displays customer information: CustID:1022, CustName:Joseph, CustEmailID:Joseph@gmail.com, CustAge:33, Height:150, Weight: Sleep:6 hrs, ActivityLevel:low, Allergy:Lactose, Allergy:Peanuts, Disease:BP, and Disease:Diabetes. Below this, the heading "Update Diet Plan Details" is centered. A yellow box contains the meal breakdown and their nutritional values:

Meal	Calories*	Proteins*	Carbohydrates*	Fat*
Breakfast	131	31	51	21
Lunch	141	71	72	22
Dinner	121	23	91	33

An "Update Diet Plan Details" button is located at the bottom of the yellow box.

22. Diet plan details is updated for May 4<sup>th</sup> and a success message is displayed at the top.  
The page also shows the updated plan details.

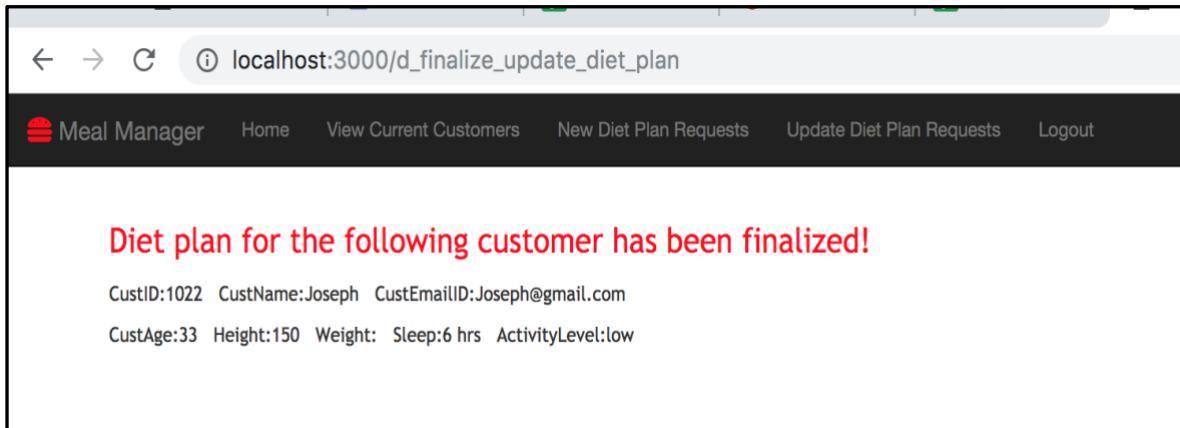
The screenshot shows a web browser window with the URL `localhost:3000/d_update_diet_plan_details_submit`. The page title is "Meal Manager". A red success message "Diet Plan Details Updated Successfully!" is displayed prominently. Below it, user information is listed: CustID:1022, CustName:Joseph, CustEmailID:Joseph@gmail.com, CustAge:33, Height:150, Weight: Sleep:6 hrs, ActivityLevel:low, Allergy:Lactose, Allergy:Peanuts, Disease:BP, Disease:Diabetes. The main content is a table titled "Diet Plan Details" showing meal data for three days: May 4, 2019, May 5, 2019, and May 6, 2019. The table has columns for SelectDay, Day, CalendarDate, MealType, Calories, Proteins, Carbohydrates, and Fat.

SelectDay	Day	CalendarDate	MealType	Calories	Proteins	Carbohydrates	Fat
●	10	Sat May 04 2019	Breakfast	131	31	51	21
			Lunch	141	71	72	22
			Dinner	121	23	91	33
●	11	Sun May 05 2019	Breakfast	120	35	20	10
			Lunch	150	35	75	20
			Dinner	130	25	90	30
●	12	Mon May 06 2019	Breakfast	100	25	20	20
			Lunch	200	35	55	20
			Dinner	200	85	90	30
			Breakfast	100	35	20	10

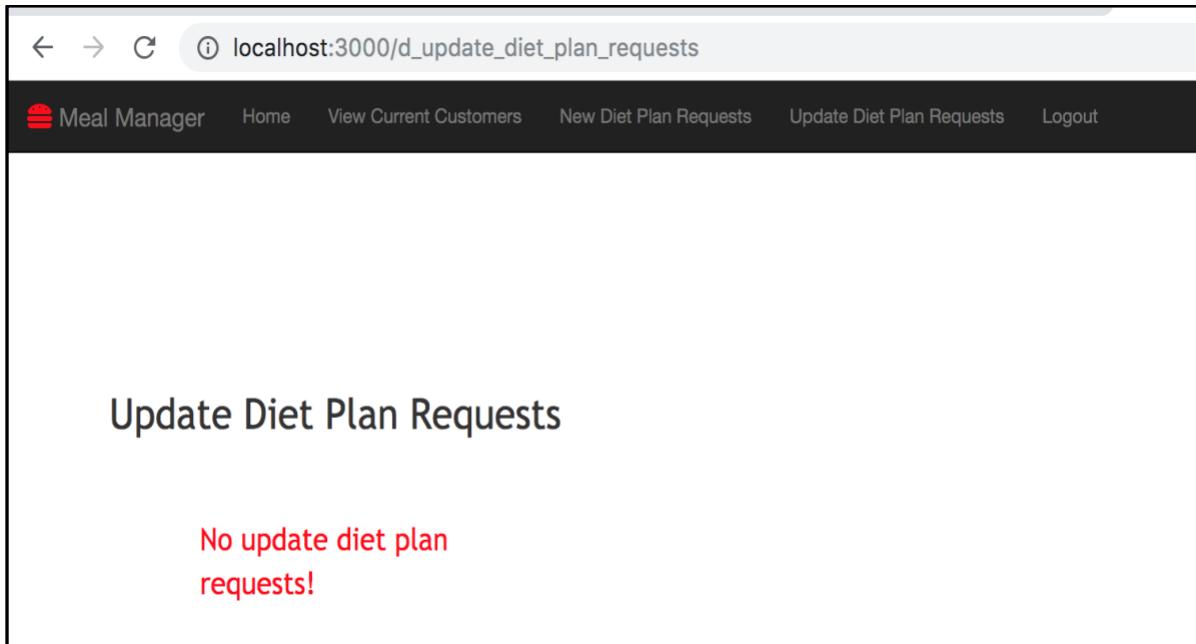
23. This process of updating can be continued till the dietitian is happy with the diet plan. Once the diet plan is updated completely, “Finalize Diet Plan” button at the bottom of the page can be clicked. This will remove the request from the update diet plan request list.

localhost:3000/d_update_diet_plan_details_submit								
Meal Manager		Home	View Current Customers	New Diet Plan Requests	Update Diet Plan Requests	Logout		
				Dinner	190	25	80	10
		24	Sat May 18 2019	Breakfast	120	35	20	10
				Lunch	100	55	75	20
				Dinner	100	25	90	10
		25	Sun May 19 2019	Breakfast	120	65	20	10
				Lunch	100	55	75	20
				Dinner	100	25	90	10
		26	Mon May 20 2019	Breakfast	120	35	70	10
				Lunch	200	55	75	20
				Dinner	200	25	80	10
		27	Tue May 21 2019	Breakfast	120	45	80	10
				Lunch	200	55	80	20
				Dinner	160	25	90	10
		28	Wed May 22 2019	Breakfast	150	35	20	10
				Lunch	100	55	55	20
				Dinner	150	25	80	10
		29	Thu May 23 2019	Breakfast	120	35	20	10
				Lunch	200	45	75	20
				Dinner	100	55	90	20
		30	Fri May 24 2019	Breakfast	120	35	20	10
				Lunch	200	85	75	10
				Dinner	100	35	90	10
				Update Diet Plan	Finalize Diet Plan			

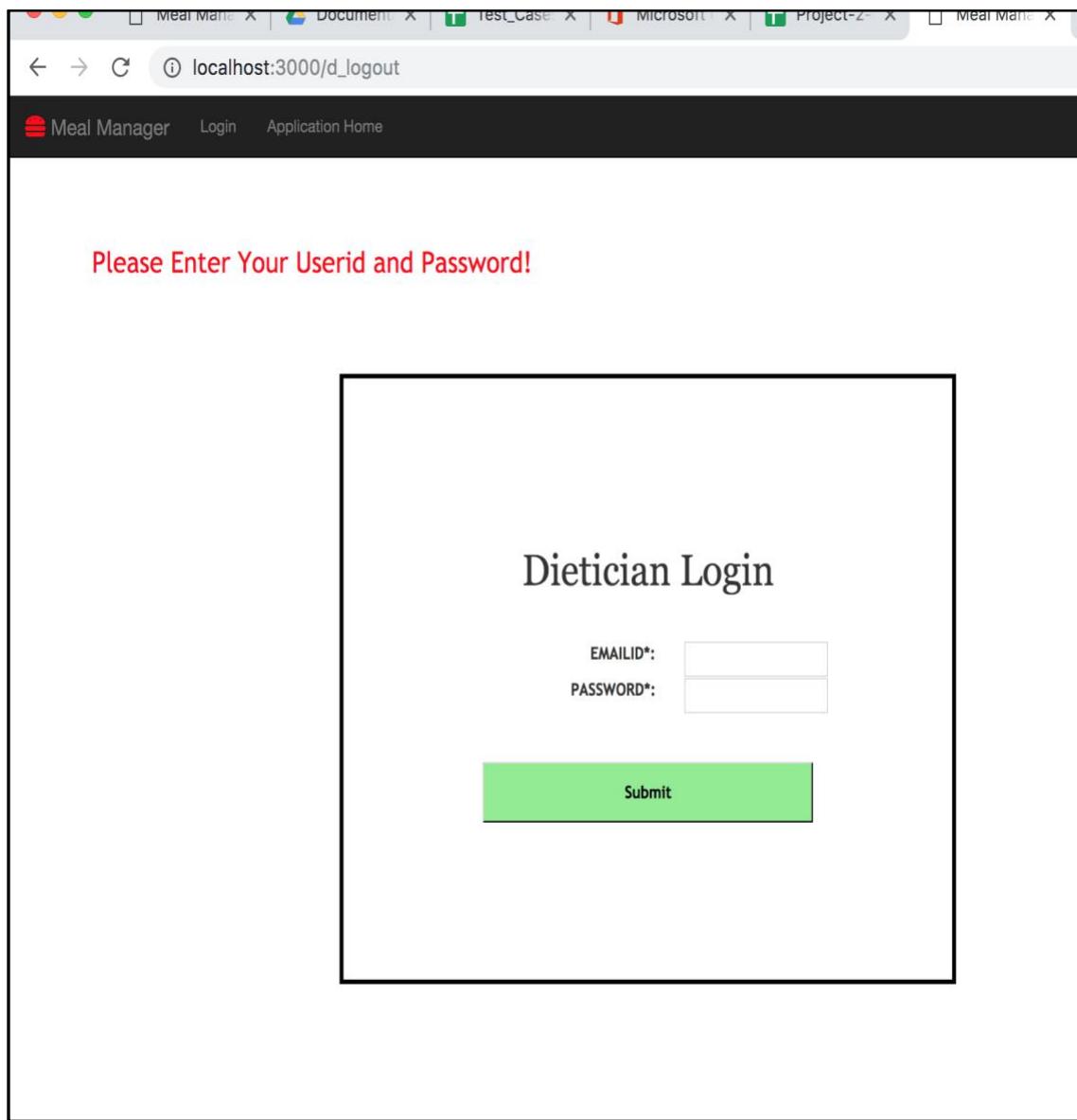
24. On finalizing the diet plan, the following success message is seen, and Joseph's request is removed from the list.



25. Joseph's request is removed from the "Update Diet Plan Requests Page". And since Kate has no more update requests, "No update diet plan requests" message is displayed.



26. On clicking on logout, Kate is logged out and login form is displayed.



27. If invalid credentials are entered in login form, error message is displayed along with login form.

← → ⌛ ⓘ localhost:3000/d\_home

Meal Manager    Login    Application Home

Invalid Credentials! Please Enter Correct Userid and Password!

### Dietician Login

EMAILID\*:

PASSWORD\*:

**Submit**

#### 7.1.4. Restaurant

##### 1. Restaurant Login Page:

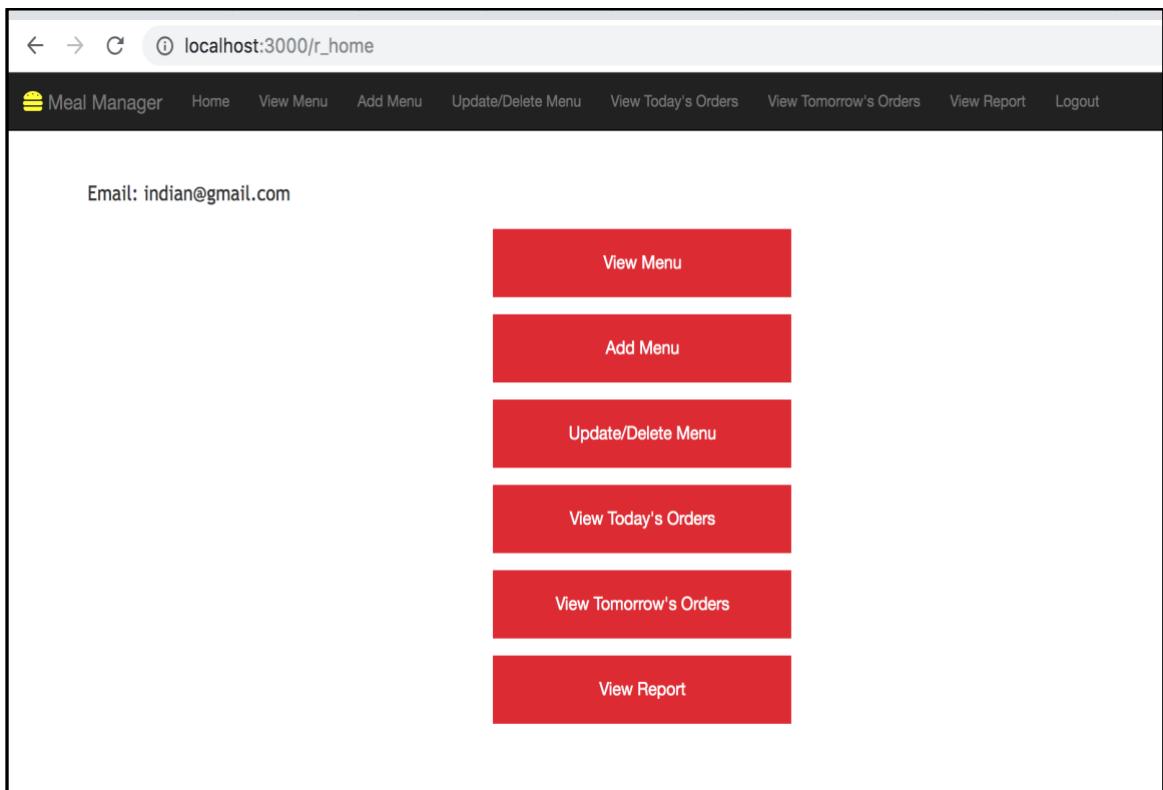
Login Page: <http://localhost:3000/restaurantlogin>

Login in to restaurant module with email id [indian@gmail.com](mailto:indian@gmail.com) and password “Indian”

Click on submit.

The screenshot shows a web browser window with the URL `localhost:3000/restaurantlogin` in the address bar. The page has a dark header bar with the text "Meal Manager", "Login", and "Application Home". Below the header, there is a red error message: "Please Enter Your Userid and Password!". A large rectangular form area contains the title "Restaurant Login". Inside this form, there are two input fields: one for "EMAILID\*" containing "indian@gmail.com" and another for "PASSWORD\*" containing "\*\*\*\*\*". Below the inputs is a green "Submit" button. The entire form is enclosed in a black border.

2. If the login credentials are valid, restaurant home page is displayed.



3. Click on “View Menu” button from home page or from navigation bar. The items the restaurant is offering is displayed on Menu Page, along with nutritional information. An item can be added into the menu without offering on any days. This can be because the restaurant wants to offer the item in future or because it stopped offering an item. For those records offered is N and DayOfWeek is empty.

The screenshot shows a web-based meal management system. The URL in the address bar is `localhost:3000/r_view_menu`. The page has a header with the title "Meal Manager" and a navigation menu with links: Home, View Menu, Add Menu, Update/Delete Menu, View Today's Orders, View Tomorrow's Orders, View Report, and Logout. The main content area is titled "Menu" and contains a table with the following data:

Offered	DayOfWeek	MealType	ItemName	Calories	Proteins	Carbohydrates	Fat
Y	Monday	Breakfast	Dosa	110	30	30	30
Y	Monday	Lunch	Biryani	200	50	100	30
Y	Tuesday	Breakfast	Dosa	110	30	30	30
Y	Wednesday	Breakfast	Dosa	110	30	30	30
Y	Wednesday	Lunch	Biryani	200	50	100	30
Y	Thursday	Breakfast	Dosa	110	30	30	30
Y	Friday	Breakfast	Dosa	110	30	30	30
Y	Friday	Lunch	Biryani	200	50	100	30
Y	Saturday	Breakfast	Dosa	110	30	30	30
Y	Saturday	Lunch	Biryani	200	50	100	30
Y	Sunday	Breakfast	Dosa	110	30	30	30
N		Breakfast	Upma	110	30	30	30
N		Dinner	Chicken Curry	150	80	20	30

4. If the restaurant does not have any items added, “Please add item to menu!” message is displayed.

A screenshot of a web browser window displaying the 'Meal Manager' application. The URL in the address bar is 'localhost:3000/r\_view\_menu'. The page title is 'Menu'. A red error message 'Please add items to menu!' is centered on the page. The navigation bar at the top includes links for Home, View Menu, Add Menu, Update/Delete Menu, View Today's Orders, View Tomorrow's Orders, View Report, and Logout.

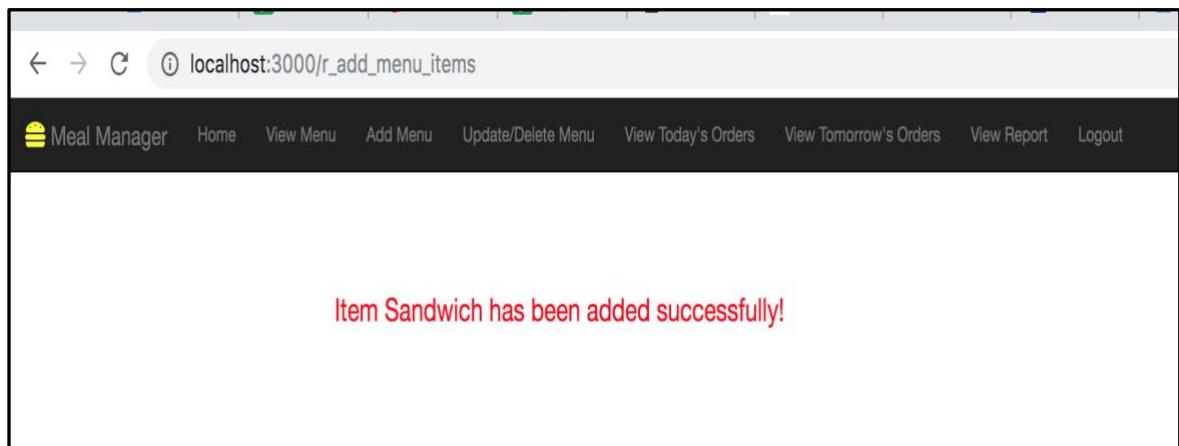
5. To add a new item to the menu, click on “Add Menu” button from home page or link from navigation bar. Add Menu form is displayed. Add all the details required and click on “Add Item” button.

A screenshot of a web browser window displaying the 'Meal Manager' application. The URL in the address bar is 'localhost:3000/r\_add\_menu\_items\_form'. The page title is 'Add Items To Menu'. The form contains the following fields:

ItemName*:	Sandwich
MealType*:	<input checked="" type="radio"/> Breakfast
	<input type="radio"/> Lunch
	<input type="radio"/> Dinner
Calories*:	300
Proteins*:	23
Carbohydrates*:	43
Fat*:	12
DaysOfWeek*:	<input type="checkbox"/> Monday
	<input checked="" type="checkbox"/> Tuesday
	<input type="checkbox"/> Wednesday
	<input type="checkbox"/> Thursday
	<input checked="" type="checkbox"/> Friday
	<input type="checkbox"/> Saturday
	<input type="checkbox"/> Sunday

At the bottom of the form is a 'Add Item' button.

6. Success message is displayed that the item Sandwich is added successfully.



7. We can see the item added on Menu page with offered as Y since it is being offered.

The screenshot shows a web application titled "Meal Manager" with a menu bar including Home, View Menu, Add Menu, Update/Delete Menu, View Today's Orders, View Tomorrow's Orders, View Report, and Logout. The main content area is titled "Menu" and displays a table of meal offerings:

Offered	DayOfWeek	MealType	ItemName	Calories	Proteins	Carbohydrates	Fat
Y	Monday	Breakfast	Dosa	110	30	30	30
Y	Monday	Lunch	Biryani	200	50	100	30
Y	Tuesday	Breakfast	Dosa	110	30	30	30
Y	Tuesday	Breakfast	Sandwich	300	23	43	12
Y	Wednesday	Breakfast	Dosa	110	30	30	30
Y	Wednesday	Lunch	Biryani	200	50	100	30
Y	Thursday	Breakfast	Dosa	110	30	30	30
Y	Friday	Breakfast	Sandwich	300	23	43	12
Y	Friday	Breakfast	Dosa	110	30	30	30
Y	Friday	Lunch	Biryani	200	50	100	30
Y	Saturday	Breakfast	Dosa	110	30	30	30
Y	Saturday	Lunch	Biryani	200	50	100	30
Y	Sunday	Breakfast	Dosa	110	30	30	30
N		Breakfast	Upma	110	30	30	30
N		Dinner	Chicken Curry	150	80	20	30

8. Let's add an item Pizza and not offer on any day.

localhost:3000/r\_add\_menu\_items\_form

Meal Manager Home View Menu Add Menu Update/Delete Menu View Today's Orders View Tomorrow's Orders View Report Logout

### Add Items To Menu

ItemName\*: Pizza

MealType\*: Breakfast  
Lunch  
Dinner

Calories\*: 785

Proteins\*: 43

Carbohydrates\*: 264

Fat\*: 12

DaysOfWeek\*: Monday  
Tuesday  
Wednesday  
Thursday  
Friday  
Saturday  
Sunday

Add Item

9. We get the success message that the item is added.

localhost:3000/r\_add\_menu\_items

Meal Manager Home View Menu Add Menu Update/Delete Menu View Today's Orders View Tomorrow's Orders View Report Logout

Item Pizza has been added successfully!

10. In Menu, we can find the item Pizza with offered as N and DayOfWeek empty.

The screenshot shows a web application titled "Meal Manager" at the URL "localhost:3000/r\_view\_menu". The menu page displays a table of food items with the following data:

Offered	DayOfWeek	MealType	ItemName	Calories	Proteins	Carbohydrates	Fat
Y	Monday	Breakfast	Dosa	110	30	30	30
Y	Monday	Lunch	Biryani	200	50	100	30
Y	Tuesday	Breakfast	Dosa	110	30	30	30
Y	Tuesday	Breakfast	Sandwich	300	23	43	12
Y	Wednesday	Breakfast	Dosa	110	30	30	30
Y	Wednesday	Lunch	Biryani	200	50	100	30
Y	Thursday	Breakfast	Dosa	110	30	30	30
Y	Friday	Breakfast	Dosa	110	30	30	30
Y	Friday	Breakfast	Sandwich	300	23	43	12
Y	Friday	Lunch	Biryani	200	50	100	30
Y	Saturday	Breakfast	Dosa	110	30	30	30
Y	Saturday	Lunch	Biryani	200	50	100	30
Y	Sunday	Breakfast	Dosa	110	30	30	30
N		Breakfast	Upma	110	30	30	30
N		Lunch	Pizza	785	43	264	12
N		Dinner	Chicken Curry	150	80	20	30

11. Restaurant update menu page: page displayed when 'UPDATE MENU' is selected.  
Data entered after page load

A screenshot of a web browser showing a table titled 'List of Items offered'. The table has six rows, each containing an icon, an item name, and a radio button. The items listed are Biryani, Chicken Curry, Dosa, Naan, Paneer Butter Masala, and Upma. Below the table is a button labeled 'Update selected item'.

	List of Items offered
<input checked="" type="radio"/>	Biryani
<input type="radio"/>	Chicken Curry
<input type="radio"/>	Dosa
<input type="radio"/>	Naan
<input type="radio"/>	Paneer Butter Masala
<input type="radio"/>	Upma

Update selected item

12. Item update menu page: page displayed when 'Update selected item is selected.'

A screenshot of a web browser showing a table with four columns: Item, Offered On Days, and Item Offered. The table lists seven days of the week, each with a checkbox next to it. The 'Offered On Days' column shows the day of the week, and the 'Item Offered' column shows 'Y' or 'N'. The 'Item' column lists 'Biryani' for all days. Below the table is a single 'Update' button.

	Item	Offered On Days	Item Offered
<input checked="" type="checkbox"/>	Biryani	Monday	Y
<input type="checkbox"/>	Biryani	Tuesday	N
<input checked="" type="checkbox"/>	Biryani	Wednesday	Y
<input type="checkbox"/>	Biryani	Thursday	N
<input checked="" type="checkbox"/>	Biryani	Friday	Y
<input checked="" type="checkbox"/>	Biryani	Saturday	Y
<input type="checkbox"/>	Biryani	Sunday	N

Update

13. Check operations for Thursday and Sunday:

The screenshot shows a web browser window for the "Meal Manager" application at the URL `localhost:3000/updatemenupage`. The page title is "Meal Manager". The navigation menu includes links for Home, View Menu, Add Menu, Update/Delete Menu, View Today's Orders, View Tomorrow's Orders, View Report, and Logout.

The main content area displays a table with the following data:

	Item	Offered On Days	Item Offered
<input checked="" type="checkbox"/>	Biryani	Monday	Y
<input type="checkbox"/>	Biryani	Tuesday	N
<input checked="" type="checkbox"/>	Biryani	Wednesday	Y
<input checked="" type="checkbox"/>	Biryani	Thursday	N
<input checked="" type="checkbox"/>	Biryani	Friday	Y
<input checked="" type="checkbox"/>	Biryani	Saturday	Y
<input checked="" type="checkbox"/>	Biryani	Sunday	N

A single "Update" button is located below the table.

14. Click on update

The screenshot shows a web browser window for the "Meal Manager" application at the URL `localhost:3000/updatemenupagedays`. The page title is "Meal Manager". The navigation menu includes links for Home, View Menu, Add Menu, Update/Delete Menu, View Today's Orders, View Tomorrow's Orders, View Report, and Logout.

The main content area displays a message in a blue-bordered box: "Updated Succesfully.....".

15. View todays orders page: page displayed when view todays order is clicked on header

The screenshot shows a web browser window with the URL `localhost:3000/r_view_todays_orders`. The page title is "Meal Manager". The main content area displays a table titled "Today's Orders" with three rows of data:

Restaurant_ID	Item Name	Ordered Date
102	Chicken Curry	Sat May 04 2019
102	Biryani	Sat May 04 2019

16. View tomorrows orders page: page displayed when view tomorrows order is clicked on header

The screenshot shows a web browser window with the URL `localhost:3000/c_view_tomos_orders`. The page title is "Meal Manager". The main content area displays a table titled "Tomorrow's Orders" with three rows of data:

Restaurant_ID	Item Name	Order Date
102	Chicken Curry	Sun May 05 2019
102	Biryani	Sun May 05 2019
102	Biryani	Sun May 05 2019

17. View report page: page displayed when view report is clicked on header. Data entered after page load

Select From Date

Select To Date

**Generate Report**

18. Page displayed when clicked on Generate Report

Item_Name	Number of Times_Ordered (For Selected Dates)	Average Rating(From Beginning Of Time)
Biryani	4	5.0000
Chicken Curry	3	No Rating
Dosa	0	No Rating
Naan	0	No Rating
Paneer Butter Masala	1	No Rating
Upma	0	No Rating

20. When logout is clicked in navigation bar, the restaurant is logged out and login form appears.

The screenshot shows a web browser window with the URL `localhost:3000/r_logout` in the address bar. The page has a dark header bar with a yellow icon, the text "Meal Manager", "Login", and "Application Home". Below this, a red error message "Please Enter Your Userid and Password!" is displayed. A large rectangular form box contains the heading "Restaurant Login" and two input fields labeled "EMAILID\*" and "PASSWORD\*". A green "Submit" button is at the bottom of the form. The entire form is enclosed in a black border.

21. If the credentials provided are invalid, then login form is displayed with error message.

← → ⌛ i localhost:3000/r\_home

Meal Manager    Login    Application Home

Invalid Credentials! Please Enter Correct Userid and Password!

Restaurant Login

EMAILID\*:

PASSWORD\*:

Submit

## 8. Major Design Decisions

For the purpose of maintaining history, we have changed some binary relations in the originally submitted ER diagram to ternary relations.

- To maintain order history for each customer, we converted orders binary relation to a ternary relation and added a new weak entity orders with order date as partial key.
- To maintain customer meal plan history i.e. to keep track of different meal plans, each customer had from the month they first registered to the application, we have created a weak entity customer billing with start date as partial key.

## 9. Test Cases and Test Plan Execution

Step #	Step Details	Inputs	Expected Results	Actual Results	Status
<b>Admin</b>					
1	Load admin page		Admin page should be loaded	Admin Page is loaded	Pass
2	Enter Email and Password fields and click Submit button	Email: <a href="mailto:admin1@mm.com">admin1@mm.com</a> Password: iamadmin1	Admin home page should be loaded	Admin home page is loaded	Pass
3	Click on Register Dietician button		Register dietician page should be loaded	Register dietician page is loaded	Pass
4	Enter Name, Email ID, Password, Experience and Qualification fields and click on Submit button	Name: dietician10 Email ID: <a href="mailto:dietician10@mm.com">dietician10@mm.com</a> Password: 10101010 Experience: 10 Qualification: M.S.	Registration successful page should be displayed	Registration successful page is displayed	Pass
5	Click on Admin Home in the header		Admin home page should be loaded	Admin home page is loaded	Pass
6	Click on Register Restaurant button		Register restaurant page should be loaded	Register restaurant page is loaded	Pass
7	Enter Name, Email ID, Password, and Location fields and click on Submit button	Name: restaurant10 Email ID: <a href="mailto:restaurant10@mail.com.com">restaurant10@mail.com.com</a> Password: 20202020	Registration successful page should be displayed	Registration successful page is displayed	Pass

		Location: San Jose			
8	Click on Logout button		admin1 should be logged out and admin login page should be displayed	admin1 is logged out and admin login page is displayed	Pass
<b>Customer Registration</b>					
1	Load home page of the application		Application home page should be loaded	Application home page is loaded	Pass
2	Click on Customer Registration button		Customer registration page should be displayed	Customer registration page is displayed	Pass
3	Enter Name, Email ID, Password, Date of Birth and Location fields and click on Register Me button	Name: customer10 Email ID: <a href="mailto:customer10@gmail.com">customer10@gmail.com</a> Password: 30303030 Date of birth: 01/01/1995 Location: San Jose	Customer login page should be displayed	Customer login page is displayed	Pass
4	Enter Email ID and Password and click on Submit button	EmailID: <a href="mailto:customer10@gmail.com">customer10@gmail.com</a> Password: 30303030	Dieticianoption should be displayed	Dieticianoption is displayed	Pass
5	Click on <i>Yes, I will get enrolled with a dietitian</i> button		Dieticianyes page should be displayed	Dieticianyes page is displayed	Pass
6	Enter the details given in the input filed		Yespaymentsuccessful page should be displayed	Yespayments uccessful	Pass

	<p>and click on Submit button</p> <p><i>Enter below lifestyle related details:</i></p> <p>Weight: 150</p> <p>Height: 175</p> <p>Measurement date of height and weight: 01/01/2019</p> <p>Activity level: I will have high level of physical exercise everyday</p> <p>Sleep: 6 -8 hrs</p> <p>Select the diseases that you are diagnosed with:</p> <p>High Cholestrol</p> <p>Select the ones that you are allergic to:</p> <p>Soy</p> <p>Select start date for meal plan (date &gt; than 2 days) : 05/10/2019</p> <p>Select A Dietician:</p> <p>dietician1</p> <p><i>Payment Details:</i></p> <p>Enter your card number:</p> <p>111122223333444</p>		page is displayed	
--	---	--	-------------------	--

		Select type of card: Visa CVV: 9999 Expiry Date: 10/10/2020			
7	Click on Logout button		customer10 should be logged out and customer login page should be displayed	customer10 is logged out and customer login page is displayed	Pass
8	Repeat Step 2 and Step 3 with data as given in the input fields	Name: customer20 EmailID: <a href="mailto:customer20@mail.com">customer20@mail.com</a> Password: 40404040 Date of birth: 02/02/1992 Location: Sunnyvale			Pass
9	Click on <i>No I will get enrolled with a dietitian</i> button		Dieticianno page should be displayed	Dieticianno page is displayed	Pass
10	Enter the details given in the input fields and click on Submit button	Meal plan selection: PlanID: 27 Start date selection: Select start date for your meal plan: (Date of execution + 2 days) Payment details: Enter your card number:	Nopaymentsuccessful page should be displayed	Nopaymentsuccessful page is displayed	Pass

		444433322221 111 Select type of card: Credit Card CVV: 8888 Expiry Date: 10/10/2022			
11	Click on Logout button		customer20 should be logged out and customerlogin page should be displayed	customer20 is be logged out and customerlogin page is be displayed	Pass
12	Enter Email ID and Password and click on Submit button	EmailID: <a href="mailto:customer20@gmail.com">customer20@gmail.com</a> Password: 40404040	"Sorry....Your Meal Plan Start Date is not tomorrow. Please Login the day before your start date." message should be displayed.	"Sorry....Your Meal Plan Start Date is not tomorrow. Please Login the day before your start date." message is be displayed.	Pass
<b>Customer Logged In Functionalities</b>					
Sara	<b>Given:</b> This customer is not enrolled with a dietitian.  Does not have a diet plan currently but has two diet plans in the past.  Location of customer - "San Jose"  <b>Assumptions Made:</b> Day of execution - Sunday				

	Date of Execution - May 5th 2019				
1	Customer Login: Enter UserID and Password	<u>UserId:Sara@gm ail.com</u> Password: Sara	Input credentials will be validated and customer home page should be displayed if they are correct	Input credentials are validated and customer home page is displayed if they are correct	Pass
2	Click on Tommorow's Menu button		Personalized menu for each customer depending on his meal plan should be displayed	Personalized menu for each customer depending on his meal plan is displayed	Pass
3	Select one item in each meal type (breakfast, lunch, dinner) and respective pick up time slots and click on Place Order button	Select the following options: 1. Breakfast: Restaurant Name: Indian Item Name: Dosa Pick up time slot: 8:30A.M  2. Lunch: Restaurant Name: Malaysian Item Name: Pad Thai Noodle Pick up time slot: 1:00P.M  3. Dinner:	"Order Placed Successfully" Message should be displayed	"Order Placed Successfully" Message is displayed	

		Restaurant Name: Indian Item Name: Naan Pick up time slot: 8:30P.M			
4	Click on Order History button		Customer should be able to see their previous orders	Customer is shown his/her previous orders	Pass
5	Click on Current Diet Plan button		"Sorry you are not enrolled in any diet plan currently" Message should be displayed	"Sorry you are not enrolled in any diet plan currently" Message is displayed	Pass
6	Click on Diet Progress button		"Sorry you do not have any current diet progress" Message should be displayed	"Sorry you do not have any current diet progress" Message is displayed	Pass
7	Click on Diet History button		Customer should be displayed the list of previous diet plans	Customer is displayed the list of previous diet plans	Pass
8	Select one of the diet plans and click on View Details button		Customer should be displayed complete details of that diet plan along with his progress.	Customer is displayed complete details of that diet plan along with his progress.	Pass
9	Click on Rate an Item button		Customer should be displayed the list of	Customer is displayed the list of	Pass

			list of previously ordered items.	previously ordered items.	
10	Select one of the items and provide a rating for that item	Select the following options:  Ordered date: Sun May 05 2019 Meal Type: Lunch Restaurant Name :Malaysian Item Name : Pad Thai Noodle Day of Week : Monday Pick up time slot : 1:00P.M  Rating : 3	"Thank you for your rating" Message should be displayed	"Thank you for your rating" Message is displayed	Pass
11	Click on Rate a Dietician button		Customer should be displayed the list of previously enrolled dieticians.	Customer is displayed the list of previously enrolled dieticians.	Pass
12	Select one of the dieticians and provide a rating for that dietician	Select the following options:  Dietician Name : dietician2 Qualification: M.S Start Date: March 30 2019	"Thank you for your rating" Message should be displayed	"Thank you for your rating" Message is displayed	Pass

		Rating : 3			
13	Click on Log Out button		Customer should be logged out and redirected to login page	Customer is logged out and redirected to login page	Pass
Stephan	<p><b>Given:</b> This customer is enrolled with a dietician.</p> <p>Has a current diet plan</p> <p>No diet plans in past</p> <p>Location of customer - "San Jose"</p> <p><b>Assumptions Made:</b></p> <p>Day of execution - Sunday</p> <p>Date of Execution - May 5th 2019</p>				
14	Customer Login: Enter User ID and Password	<u>UserId:Stephan@gmail.com</u> Password: Stephan	Input credentials will be validated and customer home page should be displayed if they are correct		Pass
15	Click on Current Diet Plan button		Customer should be displayed complete details of current diet plan	Customer is displayed complete details of current diet plan	Pass
16	Click on Request change of Diet Plan button		Customer should be displayed new diet plan request page	Customer is displayed new diet plan request page	Pass
17	Enter From Date and Description of the	Enter the following details:	"Request Placed Succesfully"	"Request Placed Succesfully"	Pass

	request and click on Submit button	From Date: May 9th 2019 Description: Please increase protein in my diet plan	Message should be displayed	Message is displayed	
18	Click on Current Diet Progress button		Customer should be displayed complete details of current diet progress	Customer is displayed complete details of current diet progress	Pass
19	Click on Update Diet Progress button		Customer should be displayed update diet progress page	Customer is displayed update diet progress page	Pass
20	Enter the details given in input fields and click on submit button	Enter the following details:  Log Date: May 1st 2019 Did you follow your diet for breakfast: Yes Did you follow your diet for lunch: Yes Did you follow your diet for dinner: Yes Any Additional Intake: Nothing	"Update Succesfully" Message should be displayed	"Update Succesfully" Message is displayed	Pass
21	Click on Log Out button		Customer should be logged out and redirected to login page	Customer is logged out and redirected to login page	Pass

**Restaurant Logged in Functionalities**

	Given: Restaurant considered in this section is already registered and has items added in the menu				
1	Load home page of the application		Application home page should be loaded	Application home page is loaded	Pass
2	Click on Restaurant Login button		Restaurant login page should be displayed	Restaurant login page is displayed	Pass
3	Enter Email ID and Password and click on Submit button	EmailID: <a href="mailto:Indian@gmail.com">Indian@gmail.com</a> Password: Indian	r_home page should be displayed.	r_home page is be displayed.	Pass
4	Click on View Menu button		Item names, day in which it is offered, nutritional information, offered currently or not should be displayed	Item names, day in which it is offered, nutritional information, offered currently or not is displayed	Pass
5	Click on Add Menu button from the header and enter the details given in the input field and click add item	Item Name: Item10 Breakfast Calories: 100 Carbohydrates:100 Protein: 80 Fat: 50 Monday	"Item Item10 has been added successfully!" message should be displayed	"Item Item10 has been added successfully!" message is displayed	Pass

6	Click on Update/Delete Menu button from the header and enter the details given in the input field and click on Update Selected Item button	List of items offered : select Item10	Item10 should be displayed with Item offered as Y for Monday and N for other six days	Item10 is displayed with Item offered as Y for Monday and N for other six days	Pass
7	Select the checkbox as given in the input field and click on Update button	Day: Thursday	Updated successfully' message should be displayed	Updated successfully' message is displayed	Pass
8	Click on View Todays Orders button from the header		Any order placed with data as [Date of execution - 1] should be displayed	Any order placed with data as [Date of execution - 1] is displayed	Pass
9	Click on View Tomorrows Orders button from the header		Any order placed with data as [Date of execution] should be displayed	Any order placed with data as [Date of execution] is displayed	Pass
10	Click on View Report button from the header		r_view_report page should be displayed	r_view_report page is displayed	Pass
11	Select the dates as given in the input and click on Generate report	From Date: 01/01/2019 To Date: Date of execution	All the items offered by the restauruant should be displyed along with the number of times each item is ordered and ratings if any	All the items offered by the restauruant is displyed along with the number of times each item is ordered and ratings if any	Pass

12	Click on Logout button		Restaurant should be logged out of the application and restaurant login page should be displayed	Restaurant is logged out of the application and restaurant login page is displayed	Pass
----	------------------------	--	--	--	------

#### Dietician Logged IN Functionalities

1	Load Dietician Login Page at <a href="http://localhost:3000/dieticianlogin">http://localhost:3000/dieticianlogin</a>		Dietician login page should be loaded	Dietician login page is loaded	Pass
2	Enter Emailid and Password fields and click Submit button	Email: <a href="mailto:kate@gmail.com">kate@gmail.com</a> Password: Kate	Dietician home page should be loaded	Dietician home page is loaded	Pass
3	Click on "View Current Customers" button on homepage or in navigation-bar		It should show the list of customers for who the current date is less than or equal to (diet_start_date+30) days. If the customer already has a diet plan for this month, and has renewed his membership for a new diet plan next month, then 2 records should come for this customer. 2 records for James - 1 for existing diet plan, and another for new diet plan where diet start	2 records for James - 1 for existing diet plan, and another for new diet plan where diet start date is in future and 1 record for Joseph is seen.	Pass

			date will be in future and 1 record for Joseph will be seen.		
4	Select the radio-button of the customer whose diet plan you want to view and click on "View Diet Plan" button	Select radio-button for customerId=102 1 and dietStartDate<to day and click on "View Diet Plan" button	It should show the list of all diet plans the customer=1021 has.	3 diet plan list for customer 1021 is seen	Pass
5	Select 1 of the diet plans and click on "View Diet Plan Progress" button	Select radio-button for customerId=102 1 and select the middle radio button and click on "View Diet Plan Progress" button	Now the diet plan details to be followed by CustomerId=1021 will be shown for all days in that particular diet plan. If the customer has entered any diet progress, that information will also be seen. If the customer has not entered any progress for a particular day, "No Data" will be seen for DietFollowed column. If the customer has added any additional intake, it will be displayed.	Diet plan details to be followed by CustomerId=1021 is be shown for all 30 days in the selected diet plan	Pass
6	Go to "New Diet Plan Requests" in the		Diet plan requests page will all new	1 record for customerid=1	Pass

	navigation bar or "New Diet Plan Requests" button from home-page		diet plan requests will be seen. 1 record for customerid=1021 with diet start date in future will be seen	021 with diet start date in future is seen	
7	Select the radio-button of the customerid=1021 and click on "Create New Diet Plan" button.		Diet Plan Details for the 30 days must be seen for customerid=1021. The customer's health details like age, height, weight, activity level, sleep, allergies, diseases information for the new diet plan must be seen. Diet plan details like calories, proteins are seen for entire month, except for day-7	Customer health details is seen. Diet plan details is seen for all 30 days. Day 7 has details empty.	Pass
8	To continue creating new diet plan select day 7 for customerid=1021 and click on "Create New Diet Plan" button at the bottom of the page		The customer's health details like age, height, weight, activity level, sleep, allergies, diseases information for the new diet plan must be seen. The form to add diet details will be displayed.	Customer health details is seen. Add diet plan details form is displayed	Pass
9	In "Add Diet Plan Details" page, add all	Breakfast: Calories:201,	After entering the details, when the	Diet Plan Details added	Pass

	the information and submit to add the diet details for the selected day. After entering the details, when click on "Add Diet Plan Details" button.	Proteins: 11, Carbohydrates: 12, fat: 13 Lunch: Calories:301, Proteins: 21, Carbohydrates: 22, fat: 23 Dinner: Calories:401, Proteins: 31, Carbohydrates: 32, fat: 33	"Add Diet Plan Details" button is clicked, the diet plan details will be added in the database. The "Diet Plan Details" page with the message "Diet Plan Details added successfully will be displayed". The added diet plan details can be seen for Day 7 for customerid=1021	successfully is displayed and diet plan details added can be seen for day 7	
10	To edit the created diet plan for Day 7, select day 7 for customerid=1021 and click on "Create New Diet Plan" button at the button of the page	Breakfast: Calories:101, Proteins: 11, Carbohydrates: 15, fat: 16 Lunch: Calories:201, Proteins: 21, Carbohydrates: 25, fat: 26 Dinner: Calories:301, Proteins: 31, Carbohydrates: 35, fat: 36	The page will be displayed with existing diet plan details information. After updating the diet plan details, when the "Add Diet Plan Details" button is clicked, the diet plan details will be updated in the database. The "Diet Plan Details" page with the message "Diet Plan Details added successfully will be displayed". The added diet plan details can be seen	Customer health details is displayed. The add diet plan details form with prefilled information is displayed. After submitting the form, "Diet Plan Details" page with the message "Diet Plan Details added successfully is displayed with updated	Pass

			for Day 7 for customerid=1021	details for day 7.	
11	If the dietician is satisfied with the diet plan for all 30 days, click on "Finalize Diet Plan" button		"Diet plan for the following customer has been finalized! " with customer name will be displayed.	"Diet plan for the following customer has been finalized! " with customer name is displayed.	Pass
12	Go to "New Diet Plan Requests" in the navigation bar or "New Diet Plan Requests" button from home-page		The new diet plan request for customerid=1021 will not be seen	New diet plan request for customerid=1021 is removed	Pass
13	Go to "New Diet Plan Requests" in the navigation bar or "New Diet Plan Requests" button from home-page		If there are no new diet plan requests, "No new diet plan requests!" message will be seen.	"No new diet plan requests!" message is displayed.	Pass
14	Go to "Update Diet Plan Requests" in the navigation bar or "Update Diet Plan Requests" button from home-page		1 record for customerid=1022 with dietupdatestartdate in future will be seen.	1 record for customerid=1022 with dietupdatestartdate in future is seen.	Pass
15	Select the radio button for customerid=1022 and click "Update Diet Plan" button.		The diet plan details for the selected customer and diet plan for all calendardates >= dietplanupdatestartdate will be displayed. The customer's health	The diet plan details for the selected customer and diet plan for all calendardates >= dietplanupdate	Pass

			details like age, height, weight, activity level, sleep, allergies, diseases information for the diet plan must be seen.	estartdate along with customer health details is displayed.	
16	select the radio button for day 10 to edit the diet plan details for day 10 for customerid=1022		The update diet plan details with prefilled values of the existing diet plan will be displayed for customerid=1022. The customer's health details like age, height, weight, activity level, sleep, allergies, diseases information for the diet plan must be seen.	The previous diet diet plan details is present in the form and customer health information is also displayed.	Pass
17	Enter the diet plan details for day 10 for customerid=1022 and click on "Update Diet Plan Details" button.	Breakfast: Calories:101, Proteins: 11, Carbohydrates: 15, fat: 16  Lunch: Calories:201, Proteins: 21, Carbohydrates: 25, fat: 26  Dinner: Calories:301, Proteins: 31,	The diet plan details will be updated in the database for customerid=1022 for day 10. The "Diet Plan Details" page with the message "Diet Plan Details updated successfully" will be displayed. The updated diet plan details can be seen.	The "Diet Plan Details" page with the message "Diet Plan Details updated successfully" is displayed and diet plan details for day 10 can be seen.	Pass

		Carbohydrates: 35, fat: 36	for Day 10 for customerid=1022		
18	If the dietician is satisfied with the diet plan for all days, click on "Finalize Diet Plan" button		"Diet plan for the following customer has been finalized! " with customer name will be displayed.	"Diet plan for the following customer has been finalized! " with customer name is displayed.	Pass
19	Go to "Update Diet Plan Requests" in the navigation bar or "Update Diet Plan Requests" button from home-page		The update diet plan request for customerid=1022 will not be seen	update diet plan request for customerid=1022 is removed.	Pass
20	Go to "Update Diet Plan Requests" in the navigation bar or "Update Diet Plan Requests" button from home-page		If there are no new diet plan requests, "No new update diet plan requests!" message will be seen.	"No new update diet plan requests!" message is seen.	Pass
21	Click on Logout button		Dietician Kate should be logged out and dietician login page should be displayed	Dietician Kate is logged out and dietician login page is displayed	Pass

## **10. Project Post Mortem**

### **10.1. Issues Uncovered**

1. We realized we had to maintain customer meal plan history and order history, to provide some functionalities we listed. So we converted some binary relations to ternary relations in ER diagram.
2. Creating an additional unique column in case of composite keys helps in easy maintenance of foreign keys. However, the new key affects the normalization process and resulted in further decomposition of the tables. Consequently, we removed the unique identifier and used the composite keys together.

### **10.2. Potential Improvements**

1. Add images of items in menu page
2. Add recipes for items, so that customers will have an option to either order the item from restaurant or even cook by themselves if they have time.
3. Based on the ratings received to restaurant-items best restaurants can be announced on a monthly basis.

## **11. Mongo DB (Extra Credit)**

We have used the nosql database MongoDB in our application to store the information from contact forms. Any user can interact with the admin through contact forms. They can send their grievances, concerns, business opportunities. The user must give a contact id and message. Whenever the user submits multiple messages over a period of time, the messages and the message timestamp will be stored for the corresponding contact id in mongo db. There is no limit on the message length.

The admin can view the messages he has got from users using “Contact Messages” button. He will get all contact ids that have sent him a message along with the timestamp of the last message from that particular user. He can further view all messages from the user, and decide how he wants to address the message.

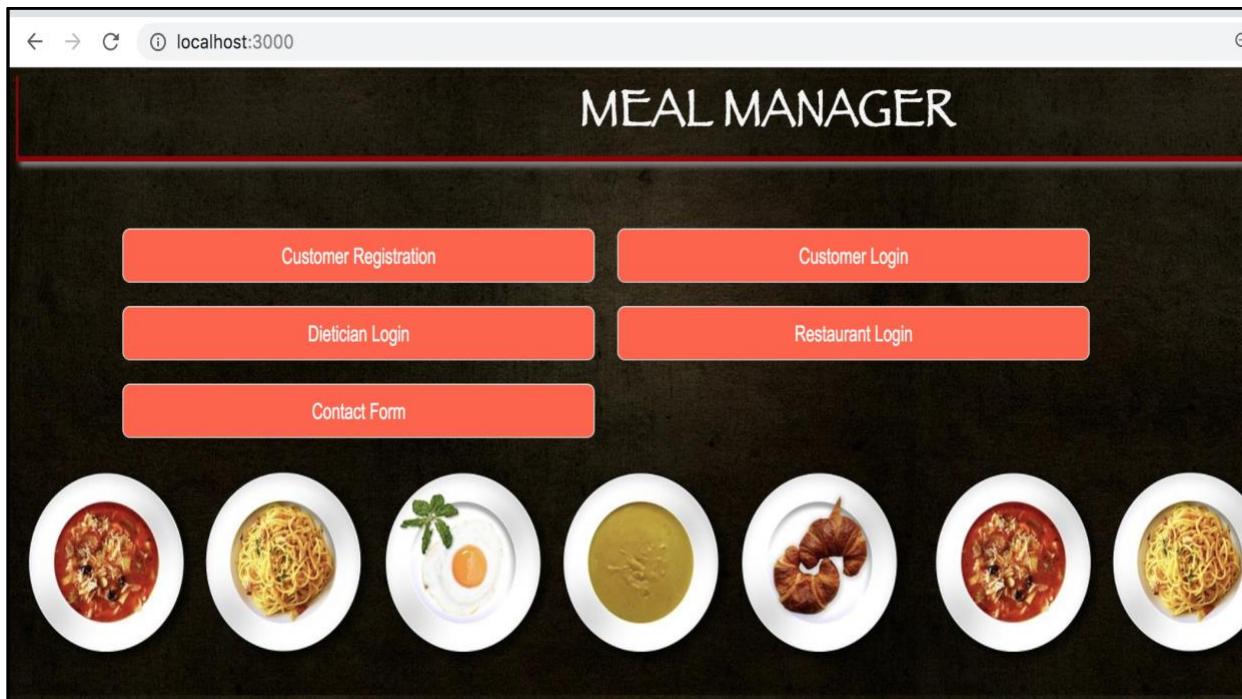
## 11.1. MongoDB application code screenshots

Screenshot of mongo db code used to update contact form.

```
3335
3336
3337 function func_updatecontactform(request,result)
3338 {
3339     var contact_email_id = request.body.email_id;
3340     var contact_message = request.body.contact_message;
3341     var MongoClient = require('mongodb').MongoClient;
3342     var url = "mongodb://localhost:27017/";
3343     MongoClient.connect(url, { useNewUrlParser: true }, function(err, db) {
3344         if (err) throw err;
3345         var dbo = db.db("mealmanager");
3346         var contact_message_time=new Date().toLocaleString();
3347         console.log(contact_message_time);
3348         dbo.collection("contactform").updateOne({name:contact_email_id},
3349             { $addToSet: {message:[ contact_message,contact_message_time]} , $set: { update_timestamp: contact_message_time} },
3350             { upsert: true }, function(err, res) {
3351                 if (err) {
3352                     throw err;
3353                 } else{
3354                     log.info('Sending success message for contact form');
3355
3356                     result.render('contactform',{cust_message:"Your message has been conveyed. You can add more messages with the same contact id."});
3357
3358                 }
3359                 //console.log("Number of documents inserted: " + res.insertedCount);
3360             });
3361
3362     });
3363
3364
3365 }
3366 module.exports.update_contactform = function(request, result)
3367 {
3368     log.info('Received request to update contact form');
3369     func_updatecontactform(request,result);
3370     console.log(request.body);
3371
3372 };
3373
```

## 11.2. MongoDB application screenshots

1. Go to home page <http://localhost:3000/> and click on “Contact Form” button. Any user can access the contact form to contact the meal manager and does not have to login.



2. The user can enter the following details and submit.

A screenshot of a web browser window displaying a contact form at the URL `localhost:3000/contactform?`. The page header includes navigation icons for back, forward, and refresh, along with the URL. Below the header, a message reads: "Please enter your message and contact details! We will get back to you soon!". The main section is titled "Contact Form". It contains two fields: "ContactID\*" with the value `mongo1@gmail.com` and "Message\*" with the value `Hi, This is message1!`. A "Submit Details" button is located below the message field. A green circular icon with a white letter "G" is positioned in the bottom right corner of the message input area.

Please enter your message and contact details! We will get back to you soon!

## Contact Form

**ContactID\*:** mongo1@gmail.com

**Message\*:** Hi, This is message1!

Submit Details

3. The user can send as many messages as he wants. All messages received from a particular contacted will be stored together in mongo db, along with the timestamp of the last message for that particular user.

The screenshot shows a web browser window with the URL `localhost:3000/update_contactform` in the address bar. The page content includes a success message: "Your message has been conveyed. You can add more messages with the same contact id." Below this is a heading "Contact Form". There are two input fields: "ContactID\*" with the value "mongo1@gmail.com" and "Message\*" with the value "This is message2". At the bottom left is a "Submit Details" button.

Your message has been conveyed. You can add more messages with the same contact id.

## Contact Form

ContactID\*: mongo1@gmail.com

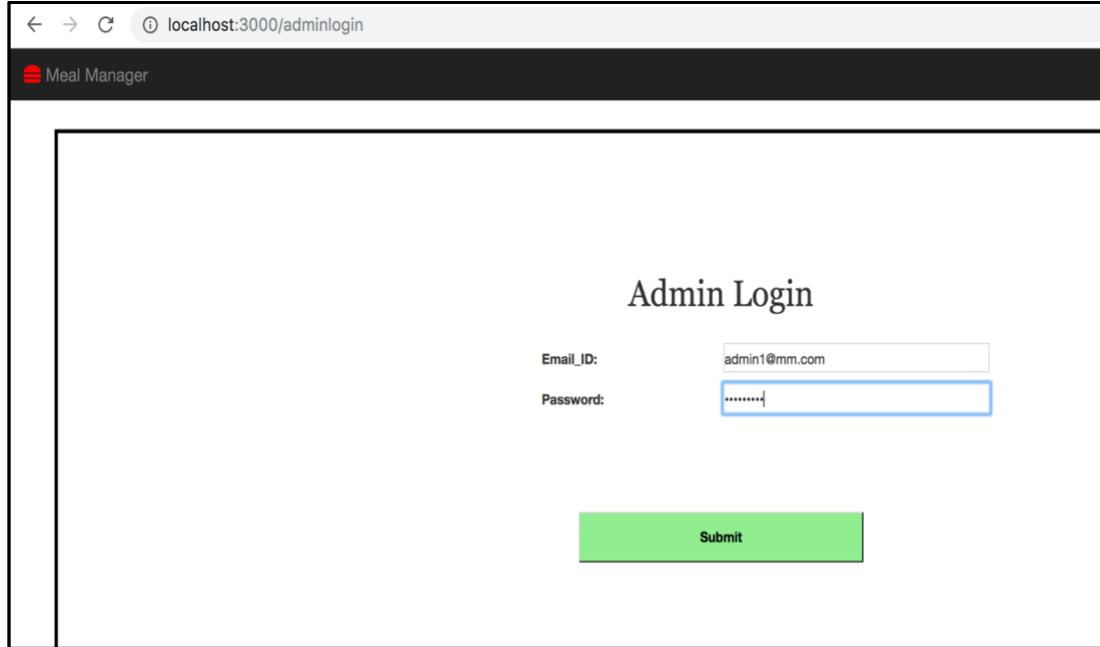
Message\*: This is message2

Submit Details

4. Screenshot of mongodb data:

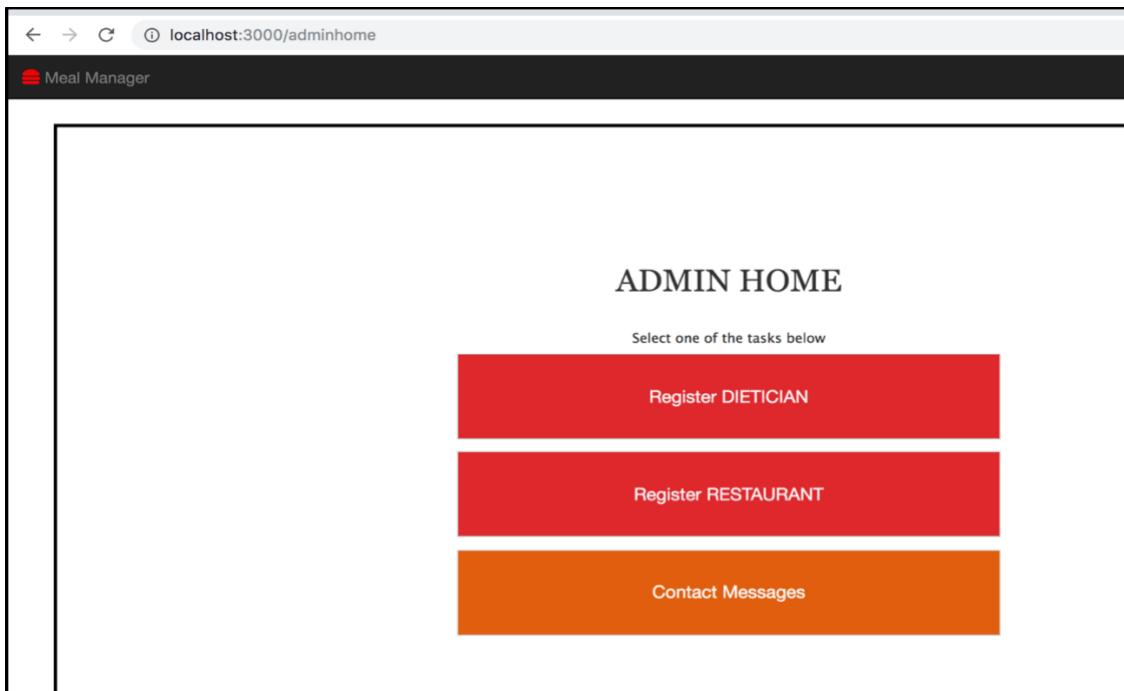
```
> db.contactform.find().pretty();
{
    "_id" : ObjectId("5ccfb7dda87c6fcdaaccf2283"),
    "name" : "mongo1@gmail.com",
    "message" : [
        [
            "This is message1!",
            "5/5/2019, 9:28:13 PM"
        ],
        [
            "This is message2",
            "5/5/2019, 9:28:31 PM"
        ],
        [
            "Please contact me on 98789373 on sundays between 9am and 10am.",
            "5/5/2019, 9:28:49 PM"
        ]
    ],
    "update_timestamp" : "5/5/2019, 9:28:49 PM"
}
{
    "_id" : ObjectId("5ccfb839a87c6fcdaaccf2324"),
    "name" : "mongo2@gmail.com",
    "message" : [
        [
            "Hello, \r\nI would like to discuss business opportunities. Please call me on monday.",
            "5/5/2019, 9:29:45 PM"
        ]
    ],
    "update_timestamp" : "5/5/2019, 9:29:45 PM"
}
> █
```

5. Admin will be able to view the contact form data. For that, he must login as an admin from <http://localhost:3000/adminlogin>. Give the email\_id as admin1@mm.com and password as "iamadmin1"



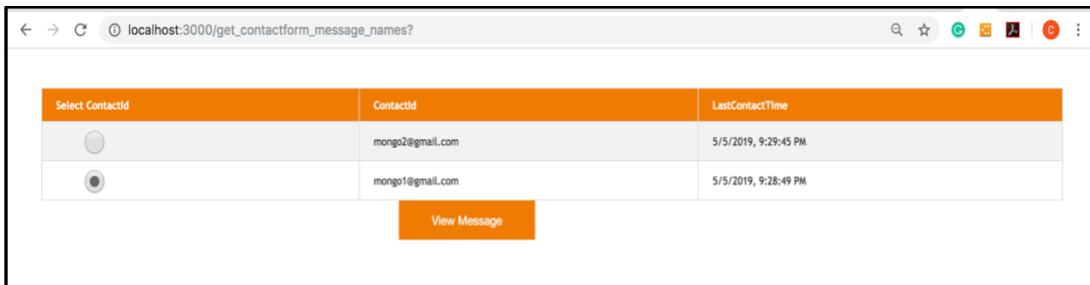
A screenshot of a web browser showing the 'Admin Login' page. The URL in the address bar is 'localhost:3000/adminlogin'. The page has a dark header with the text 'Meal Manager'. The main content area is titled 'Admin Login'. It contains two input fields: 'Email\_ID' with the value 'admin1@mm.com' and 'Password' with the value '.....'. Below the inputs is a green 'Submit' button.

6. When the admin logs in, on the home page he has to click on "Contact Messages" button to view contact form messages.



A screenshot of a web browser showing the 'ADMIN HOME' page. The URL in the address bar is 'localhost:3000/adminhome'. The page has a dark header with the text 'Meal Manager'. The main content area is titled 'ADMIN HOME' and contains the instruction 'Select one of the tasks below'. There are three buttons: a red button labeled 'Register DIETICIAN', a red button labeled 'Register RESTAURANT', and an orange button labeled 'Contact Messages'.

7. He can view the contact ids from whom he has received messages, and the data is ordered by timestamp in descending order.

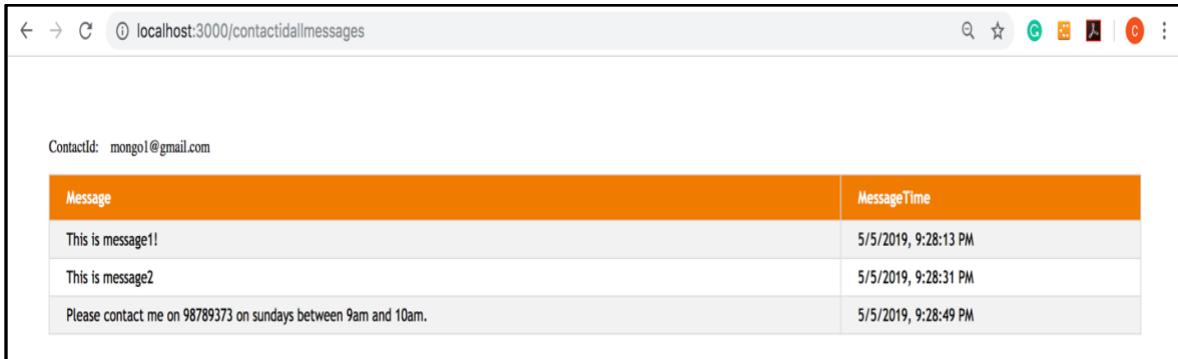


A screenshot of a web browser window titled "localhost:3000/get\_contactform\_message\_names?". The page displays a table with three columns: "Select ContactId", "ContactId", and "LastContactTime". There are two rows of data:

Select ContactId	ContactId	LastContactTime
<input type="radio"/>	mongo2@gmail.com	5/5/2019, 9:29:45 PM
<input checked="" type="radio"/>	mongo1@gmail.com	5/5/2019, 9:28:49 PM

At the bottom center of the table is a blue "View Message" button.

8. Select [mongo1@gmail.com](#) user and click on “View Message”. He can see all messages received from user [mongo1@gmail.com](#) from the contact form.



A screenshot of a web browser window titled "localhost:3000/contactidallmessages". The page shows a message header "ContactId: mongo1@gmail.com". Below it is a table with two columns: "Message" and "MessageTime". There are three rows of data:

Message	MessageTime
This is message1!	5/5/2019, 9:28:13 PM
This is message2	5/5/2019, 9:28:31 PM
Please contact me on 98789373 on sundays between 9am and 10am.	5/5/2019, 9:28:49 PM