IT632 : Software Engineering

Group 10 – Diet Tracker and Analyser Documentation

Project Report

2nd Semester

Guided By:

Saurabh Tiwari

Mentor :

Sameeksha Jain

TEAM MEMBERS :

Prachi Gole [**Group Leader**] - 202112077

Faisal Ajmeri - 202112031

Karina Bavishi - 202112052

Sharvi Gabani - 202112066

Nirmal Panchal - 202112085

Uttamkumar Dobariya - 202112094

Arohi Singhal - 202112100

Dhrumil Manojkumar Gohil - 202112107

Nisarg Shah - 202112111

Ayush Garg - 202112119

Mansi Jani - 202112120

Acknowedgement

We have an opportunity to express our gratitude today. We take this occasion with much pleasure to thank all the people who have helped us throughout the course of our journey towards producing this project. We sincerely thank our project guide ***Prof. Saurabh Tiwari*** for their guidance, help and motivation.

Apart from the subject of our project, we learnt a lot from them, which we are sure, will be useful to us in different stages of our life. We would like to express our gratitude to them for much help with the project design and methodology, and for review and many helpful comments.

We would like to thank our mentor ***Sameeksha Jain*** for his valuable guidance and suggestions which helped us in various phases of the completion of this project. We will always be thankful to you in this regard. We take this opportunity to thank all our lecturers who have directly or indirectly helped our project.

INDEX

1. Introduction

Diet Advisor/Analyser is a **wellness platform** that provides services such as calorie tracking, fitness coaching, and diet plan.

It will allow users to track calories, and monitor progress toward weight-management goals.

Diet Advisor takes a holistic lifestyle tracking approach to keep users engaged, motivated, and adhere to exercise and diet goals.

2. Overall Description

2.1 Project Scope Description

Currently a lot of people are becoming aware of the importance of fitness in their lives. Our website helps them keep their weight in check by keeping track of the food they consume throughout the day and the exercise they do.

We have designed a **Diet Advisor Website**. This web-app helps user reach a specific goal of weight loss/gain/maintenance. Here first user will enter his/her weight, normal activity level, goal to reach and time to reach the goal. Accordingly, he is given the calories to be consumed daily that he has to abide by to reach his goal. Website contains list of food items along with calories which user can add to his daily diet. If certain food is not present, he/she can add a food item by requesting it to the admin. Weekly/monthly/end of goal reports are given as per the goal-end date. Further, option to change goal provided.

Here we want to expand our project further by including features like an automatic diet generation, suggestion of food that would be better suited than certain opted choice, generation of workout routine etc. which is current out of the scope of our project. However, since we have chosen the evolutionary prototyping model. It will be easier to implement additional changes to the existing project to further improve and build on these aspects.

2.2 USERS AND STAKEHOLDERS

1. User: Common People
2. Stakeholder: Admin

2.3 POSSIBLE FEATURES

1. **USER-FRIENDLY:** Our project provides a clean user interface that has been designed taking into consideration all the requirements of our system. It is straightforward and easy to understand making the user navigate it with no confusion.

2. **PORTABILITY:** This project has the ability to adapt with different software, that is, it can run across different computers. The project can be executed on maximum software environments. It can be easily moved from one computer to another.

3. **INTEGRITY:** Another feature is Integrity that means the information provided by our system is real, verified and safe. In the system any unauthorized user cannot modify the data and hence the system is reliable.

4. **RELIABILITY:** Any and all data of our project has been verified by certified by dieticians (admins).

2.4 SOFTWARE INTERFACE

The following will be the technology stack used in this application:

Tools	**Technologies**
UI/UX Design	Figma
Backend	NodeJS , Express,Js, JavaScript
Frontend	HTML, CSS, JavaScript
Database	MySQL
Version Control System	GitHub
IDE	Visual Studio Code
Category	Web Application

2.5 REQUIREMENT ELICITATION TECHNIQUE

The following questions were raised upon requirement elicitation:

1. **How are you maintaining the data in the project?**

   Data is being maintained on the AWS Amazon Server.

2. **What are the risk factors that affect the proposed system development and what to do to avoid that?**

   The only current risk factor in our system is when a user does not accurately represent his diet details as consumed by him.

3. **What are the main features of the project?**

   The main feature of the project is to track the caloric content by meal basis of a user such that he/she can work towards a final goal that he/she has set.

4. **How will you verify if a user is registered or not?**

   A Registered user will have his credentials stored in the database already hence if a new user tries to login it will automatically recognise him/her as a new user and consequently append their credentials to the database.

5. **Will a non-registered user story be different from that of a registered user? If so, how?**

   Yes, they are different. A non-registered user essentially will only be able to use the application upon login and registration after which he is converted into a registered user who has access to all the functionalities of the application.

6. **How/when will a user see his progress?**

   Upon goal completion the user will be able to see his/her progress in the application. Furthermore, individually he can check his goal stats for previous days too.

7. **Will he be able to change his goals once he starts a goal?**

   There 2 ways in which a user would be able to change his goal.

   1. When he/she wants to change it midway from a goal he has already set. He can go to his user profile page and edit the goal which will reset the particular goal entirely and delete all previous data from the database to accommodate for the new goal that is being set.
   2. When a new goal is set after a goal is completed. He is instantly redirected to edit new goals page on completion where he can enter his new goal.

8. **Is the user able to logout mid process? Will the user data be stored if he logs out in the middle of the day?**

   Yes, all the user data is stored on the database and can be retrieved upon re-login of the user. So, even if a user logs out mid process, all his data from the session would be stored.

9. **Will the user be able to see his/her previous goals? If not, will it be in the future scope?**

   Currently the user cannot see his previous goals. However, we are taking that into consideration in our future scope of the project where each user will be able to see all the goals he has worked towards.

10. **What action does the project take when a user is not able to reach his/her goal?**

    If a user is not able to reach his/her goal, we additionally suggest / advise the user with some exercises that he/she can do daily so as to perform better and reach his goal next time.

11. **When is the goal process terminated?**

    The goal process is either terminated when a user cancels his goal to re-enter a new one or when the number of days since he started a goal are over.

12. **What happens when the user exceeds his daily caloric goal?**

    When a user enters the number of calories exceeding his suggested amount then the number calculating the total calories per meal will transform to a warning RED colour indicating that he has crossed his limit of calories for that particular meal.

13. **How do you analyse the progress of the user in the goal section?**

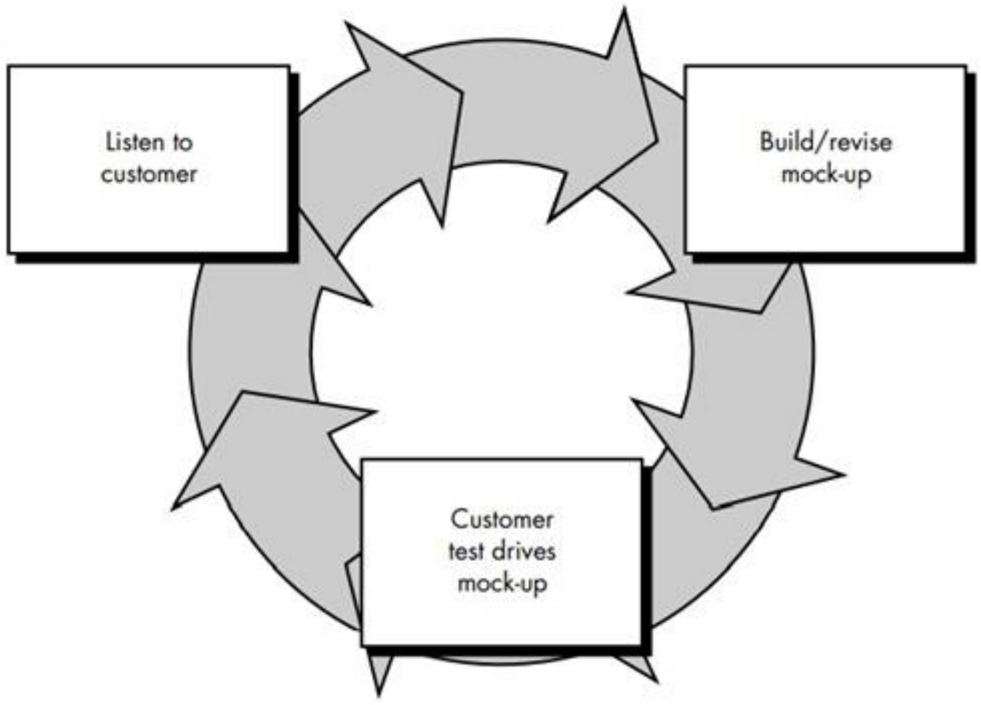
    User progress is analysed based on the total number of days that he/she has stuck to his diet regimen as provided by the app. In case his success rate is 75% plus - he has successfully completed the goal; if his success rate is 50% plus - he has partially completed the goal; if his success rate is 25% plus - he has not attempted to completed the goal; finally if his success rate is 0% plus and 25% minus - he has failed to completed the goal. According to the user stats we then suggest the exercises or steps we need to take to result in a better progress for the user.

2.6 PROCESS MODEL

The **prototyping model** is a systems development method in which a prototype is built, tested and then reworked as necessary until an acceptable outcome is achieved from which the complete system or product can be developed.

This model works best in scenarios where not all of the project requirements are known in detail ahead of time. It is an iterative, trial-and-error process that takes place between the developers and the users.

Here we want to expand our project further by including features like an automatic diet generation which is currently out of the scope of our project. However, since we have chosen the **evolutionary prototyping model**. It will be easier to implement additional changes to the existing project by building a prototype of the entire application. A prototype helps us visualize the entire app making it easy to point out the requirements and even the faults of the system.



3. Functional Requirement

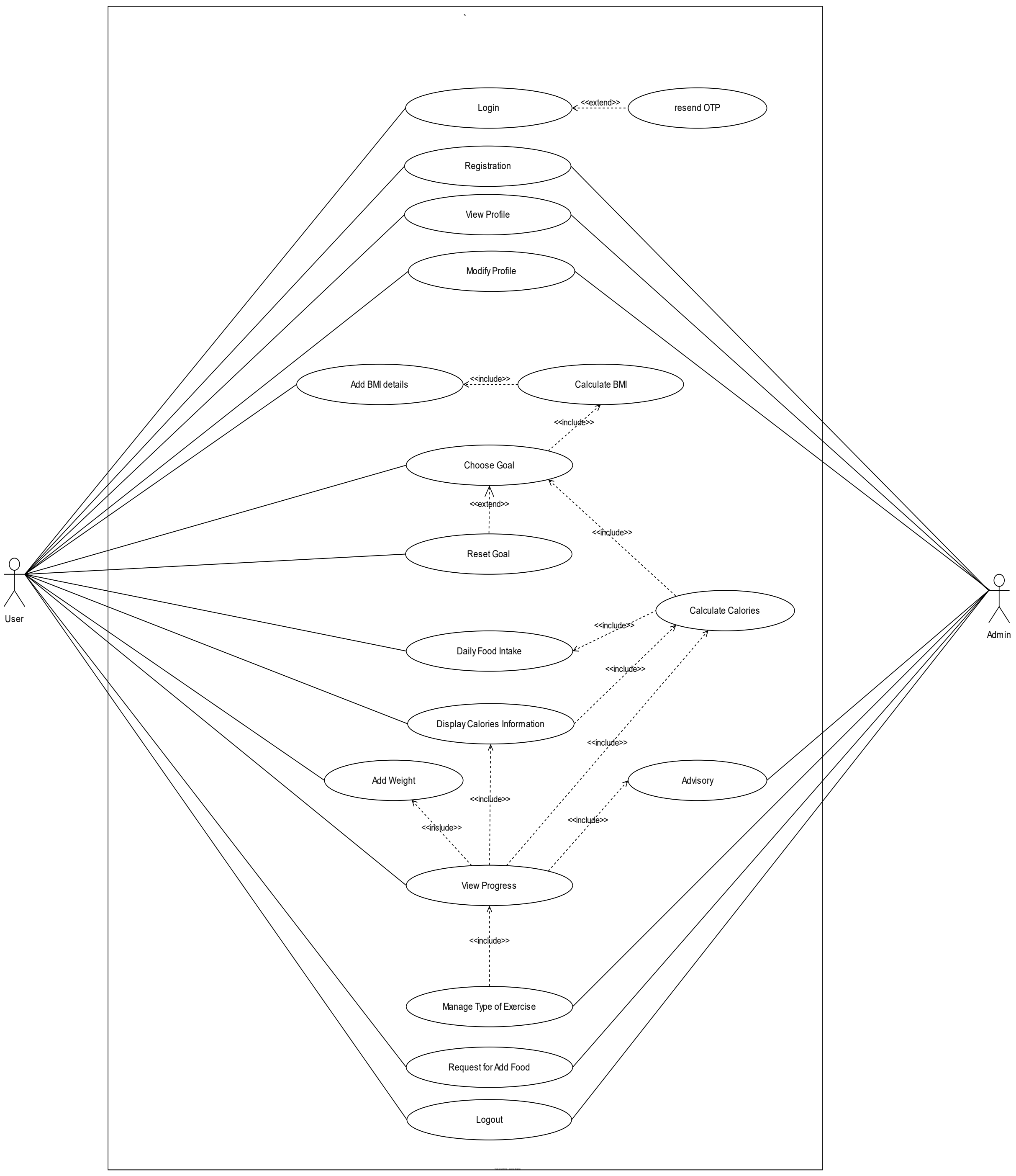
3.1 FUNCTIONAL REQUIREMENTS AND USER STORIES

1. **Register**:
    - If a user wants to use our system services then first, he/she must register him/herself.
    - The registration process allows linking your application to each specific user and providing him or her with personalized content.
2. **Login**:
    - Existing users have to login into our system via their credentials (email and OTP) to use our services.
3. **Calculate BMI:**
    - System intakes the user's weight, height, age for calculating Body Mass Index.
4. **User goal:**
    - Our System is defined on the goal that the user wants to achieve i.e., Gain/Lose/Maintain Weight.
5. **User Preferences:**
    - User can add his food preferences [veg, non-veg, eggetarian] so that he is shown a list of food items that is conforming to his choice.
6. **Calculate calories:**
    - Our system calculates the calories on the user's goal on a per **meal** basis where Breakfast is the largest meal of the day and Dinner being the smallest excluding Snacks and BMI.
    - Our system also calculates the food's calories that the user has consumed.
7. **Display calories**:
    - It will display the calories that the user has to intake to achieve the goal per meal.
    - In case a user skips a meal, he will be shown a warning at the end of the day to enter his meal. In case a meal is not entered that day, it will be counted as a day in the calculation.

- In case he has added an item or 2 to the list, but has calories to spare, these similarly will be added to the next meal.
- Dinner calories will not be added to the next meal as that will be the end of the day and will mean that the user has stayed under his goal.

8. **Daily Food Intake, Display Type of Exercise**:
    - Users can add meals, exercise details (low, medium or high).
    - These details are added on a daily basis.
9. **User progress**:
    - Our system is designed so that user can check whether he/she is exceeding the goal or lagging behind the goal.
    - Progress is checked on the data inserted by the user on a daily basis and its calorie count.
10. **Progress at end of day/week/month**:
    - generating the progress report on the user's demand.
11. **Change goal**:
    - the user can change the goal at any time.
12. **Logout**:
    - Users can log out from the system if they want.
13. **View Profile**:
    - User can view his/her name, contact details, gender, age, weight and height.
    - Admin can view the user's profile.
14. **Update Profile**:
    - User can view his/her name, contact details, gender, age, weight and height.

3.2 USE CASE DIAGRAM



3.3 USE CASE DESCRIPTION

1. Use Case: Login

Use Case Name	Login
Description	This use case describes how a user and admin logs into the Diet Advisor/Analysis System.
Actors	Admin, User
Pre – Condition	None
Post - Condition	If the use case was successful, the actor is now logged into the system. If not the system state is unchanged.
Basic Path	This use case starts when an actor wishes to log into Diet Advisor/Analysis System. 1. The system requests that the actor enter his/her Email id and correct OTP. 2. The actor enters his/her Email ID and OTP. 3. The system validates the entered email ID and OTP and logs the actor into the system.
Alternative Path	Invalid Name / OTP If in the *Basic Flow* the actor enters an invalid email Id and/or OTP, the system displays an error message. The actor can choose to either return to the beginning of the *Basic Flow* or cancel the login, at which point the use case ends.

2. Use Case: Registration

Use Case Name	Registration
Description	This use case describes how a user Registers into the Diet Advisor/Analysis System.
Actors	User
Pre – Condition	Login should be done with the valid OTP.
Post - Condition	If the registration is successful, the actor now registers into the system. Otherwise, user will not able to track their goal.
Basic Path	This use case starts when an actor wishes to Register into Diet Advisor/Analysis System. 1. The system requests that the actor enter his/her name, dob, activity level, weight, goal weight, timeline to achieve goal, height. 2. The actor enters his/her name, dob, activity level, weight, goal weight, timeline to achieve goal, height.

	3. The system validates the entered name, dob, activity level, weight, goal weight, timeline to achieve goal, height and registers the actor into the system.
Alternative Path	Invalid Data field entry If in the *Basic Flow* the actor enters an invalid input for given fields, the system displays an error message. The actor can choose to either return to the beginning of the *Basic Flow* or cancel the registration, at which point the use case ends.

3. Use Case: View Profile

Use Case Name	View Profile
Description	This use case describes how a user and admin view their profile in the Diet Advisor/Analysis System.
Actors	User, Admin
Pre – Condition	User and admin should be logged in.
Post - Condition	User and admin are able to view their profile.
Basic Path	4. User selects "My Account" and Admin can select "Profile" to view their profile. 5. System displays categories of profile for User (email, weight, height, activity level, gender, goal weight, time period). System displays categories of profile for Admin (username, email, DOB, gender). 6. System displays details.
Alternative Path	none

4. Use Case: Modify Profile

Use Case Name	Modify Profile
Description	This use case describes how a user and admin modify their profile in the Diet Advisor/Analysis System.
Actors	User, Admin
Pre – Condition	User and admin should be logged in.
Post - Condition	User and admin profile has been updated.
Basic Path	1. User and Admin selects "Profile" 2. System displays categories of profile for User (email, weight, height, activity level, gender, goal weight, time period).

	System displays categories of profile for Admin (username, email, DOB, gender). 3. System displays details. 4. User/Admin updates detail, presses "Update Profile" 5. System validates data as required, updates User and Admin profile.
Alternative Path	After edit all details if user/admin not select update profile button then it remains unchanged and user can go back and here use case ends.

5. Use Goal: Choose Goal

Use Case Name	Choose Goal
Description	This use case describes how a user choose goal in the Diet Advisor/Analysis System.
Actors	User
Pre – Condition	User should be logged in. They need to add BMI details like height and weight.
Post - Condition	User is able to insert the detail of food on their daily basis.
Basic Path	1. User selects "Register" to choose their goal. 2. System displays BMI information for User. 3. After that they able to finalize their goal.
Alternative Path	none

6. Use Case: Calculate BMI

Use Case Name	Calculate BMI
Description	This use case describes how a user will get the information about their BMI.
Actors	User
Pre – Condition	User should be logged in. They need to give information of their height and weight for calculation of BMI.
Post - Condition	User is able to view the detail of their BMI with user understandable graph.
Basic Path	1. User enter detail of height and weight. 2. System displays BMI information of User.
Alternative Path	none

7. Use Case: Reset Goal

Use Case Name	Reset Goal
Description	This use case describes how a user will reset their goal and set new goal.
Actors	User
Pre – Condition	User should be logged in. They have already chosen any goal.
Post - Condition	User is able to set new goal or change their recent goal.
Basic Path	1. This use case starts when a user wants to change their recent goal and again want to set new goal. 2. They will again enter BMI details. 3. User click on Reset Goal and after that they can add food details.
Alternative Path	After successfully completed goal user can logout or set new goal.

8. Use Case: Daily Food intake

Use Case Name	Daily Food Intake
Description	This use case describes how a user will insert their daily meal information (breakfast, lunch, snacks, dinner)
Actors	User
Pre – Condition	User should be logged in. They have already chosen any goal.
Post - Condition	User is able to calculate their daily meal calories according to their given information of food calories.
Basic Path	1. User will select breakfast food item list which will be given by advisor(admin) with food calories and its quantity. 2. User will select lunch food item list which will be given by advisor(admin) with food calories and its quantity. 3. User will select snacks food item list which will be given by advisor(admin) with food calories and its quantity. 4. User will select dinner food item list which will be given by advisor(admin) with food calories and its quantity.
Alternative Path	User can choose one of the meals from breakfast, lunch, snacks and dinner. User has also liberty of not choosing any option of meal.

9. Use case: Calculate Calories

Use Case Name	Calculate Calories
Description	This use case describes thar user can view their progress according to their goal.
Actors	User
Pre – Condition	User should be logged in. They have already chosen any goal. user daily meal consumption information should be required.
Post - Condition	User progress report will be generated.
Basic Path	Daily meal calories will be calculated by using the information provided by the user.
Alternative Path	none.

10. Use case: View Progress

Use Case Name	View Progress
Description	This use case describes that user can view their progress according to their goal.
Actors	User
Pre – Condition	User should be logged in. They have already chosen any goal. User current weight should be inserted. Daily meal consumption should be calculated for particular day wise and month wise should be inserted.
Post - Condition	User can decide that he/she wants to continue with that recent goal by following the given review from the advisor(admin) or choose a new goal.
Basic Path	1. User will insert his/her current weight for generating progress report. 2. User will see current progress of their choosing goal. 3. User can also see their meal detail from day 1 to last day. 4. User will also be advised for some exercise according to their goal timeline reached.
Alternative Path	User can set new goal after view the progress report. User can logout from the system.

11. Use case: display type of exercise

Use Case Name	Display type of exercise
Description	This use case describes that user can view the recommendation exercise list to maintain their goal.
Actors	Admin
Pre – Condition	Admin should be logged in. Exercise name and exercise level field should not be empty.
Post - Condition	All the record of inserted exercise should be display on Admin dashboard with its detail.
Basic Path	1. Admin will select Add option to insert the exercise information. 2. After click on save button record will be inserted into table and it will reflect on admin dashboard and this exercise will be suggested to user during the view progress report.
Alternative Path	Admin can update or delete the exercise from the exercise information list.

12. Use case: Request for add food

Use Case Name	Request for add food
Description	This use case describes how a user can request to admin for adding new food item in the meal list.
Actors	User, admin
Pre – Condition	User should be logged in. They have already chosen any goal. Food name field should not be empty.
Post - Condition	Admin will insert the food item in food table.
Basic Path	1. User selects "Add Food" for requesting to add new food item. 2. User will enter food name and submit the request.
Alternative Path	User have requested food but admin has not inserted the food item in table.

13. Use case: Logout

Use Case Name	Logout
Description	This use case describes that user or admin can logout from the Diet Advisor/Analysis System.

Actors	User, Admin
Pre – Condition	Admin or user must be logged in.
Post - Condition	Successfully logout message will be displayed.
Basic Path	This use case starts when an actor wishes to logout from Diet Advisor/Analysis System.
Alternative Path	none.

4. Non-Functional Requirement

1. **Platform independent**:

   Our system will be built on a web-based framework so it can be accessible by any browser easily so there is no platform dependence, therefore, a website can work on Linux, windows and mac operating systems.

2. **Usability**:

   The system will allow the users to access the system from the Internet using HTML or its derivative technologies like CSS. System will require the latest chrome version. The system uses a web browser as an interface. Since all users are familiar with the general usage of browsers, no special training is required. The system is user friendly.

3. **Availability**:

   Our system will be available 100% for the user and it can be used 24 hours a day and 365 days a year.

4. **Security:**

   Our system will have user authentication and validation of members using their email with OTP. Captcha code will be used for user login. Our system will not allow a member to see another member's account.

5. **Performance**:

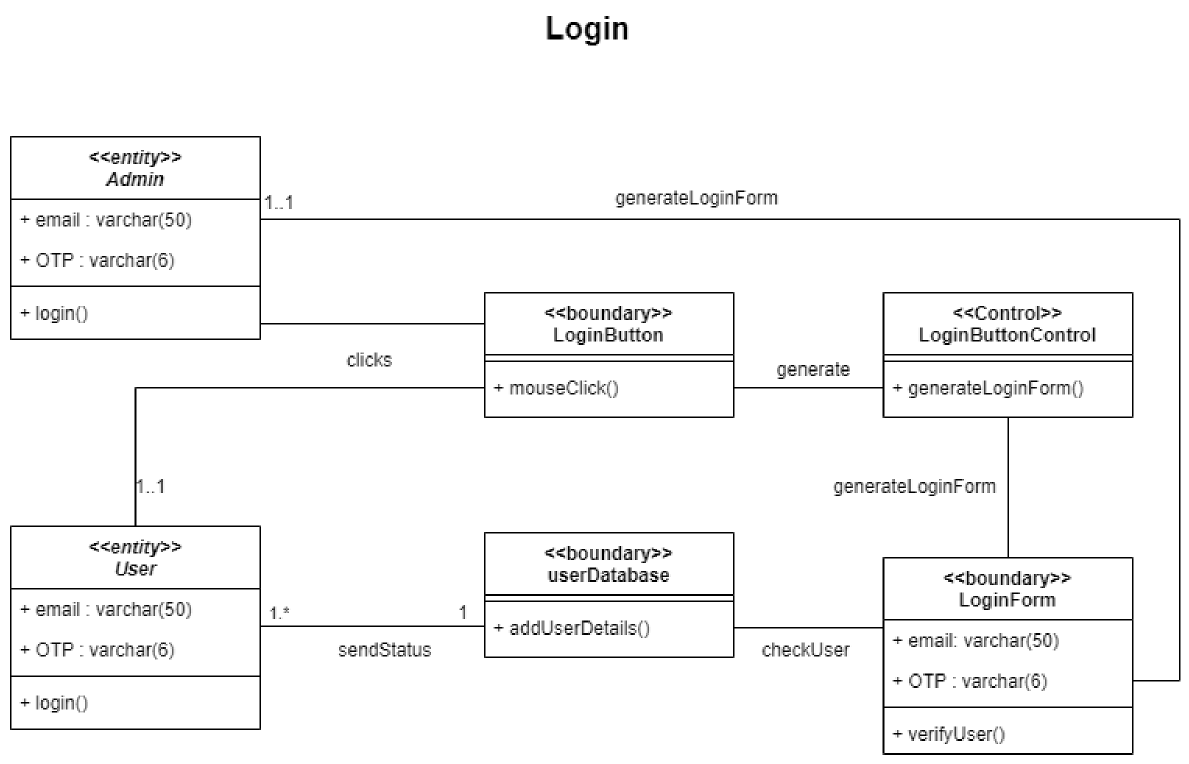
   Performance of our system will be fast and accurate. It will be able to handle large amounts of data. Our system performance will not affect if there will be traffic load of multiple users.

6. **Reliability**:

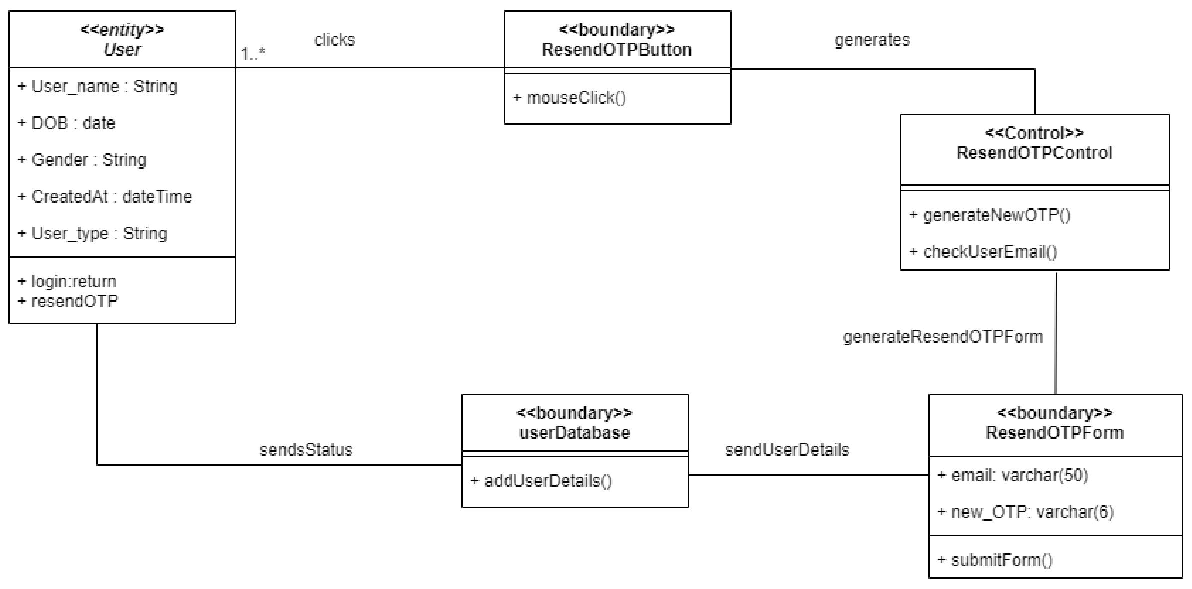
   Our system will be reliable in every system. Due to security and privacy the data of customer gets confidential so it will be reliable.

5. Analysis Design Documents

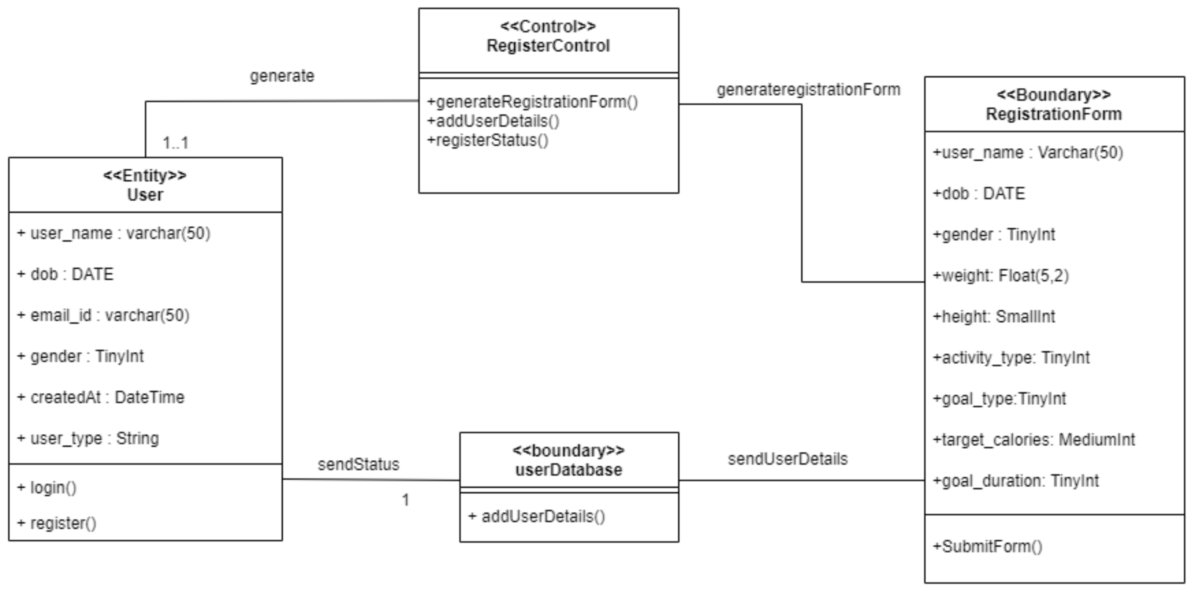
5.1 Analysis Class Diagrams



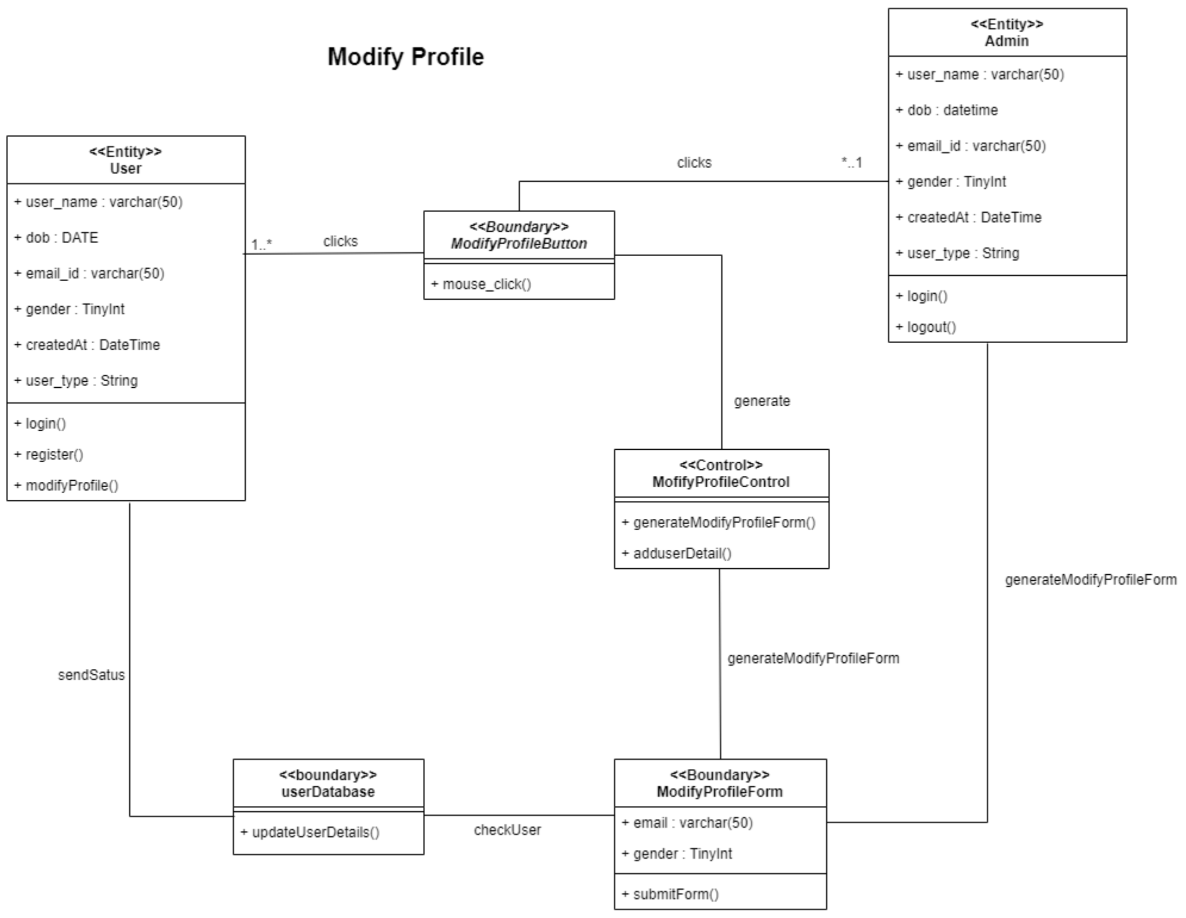
Resend OTP



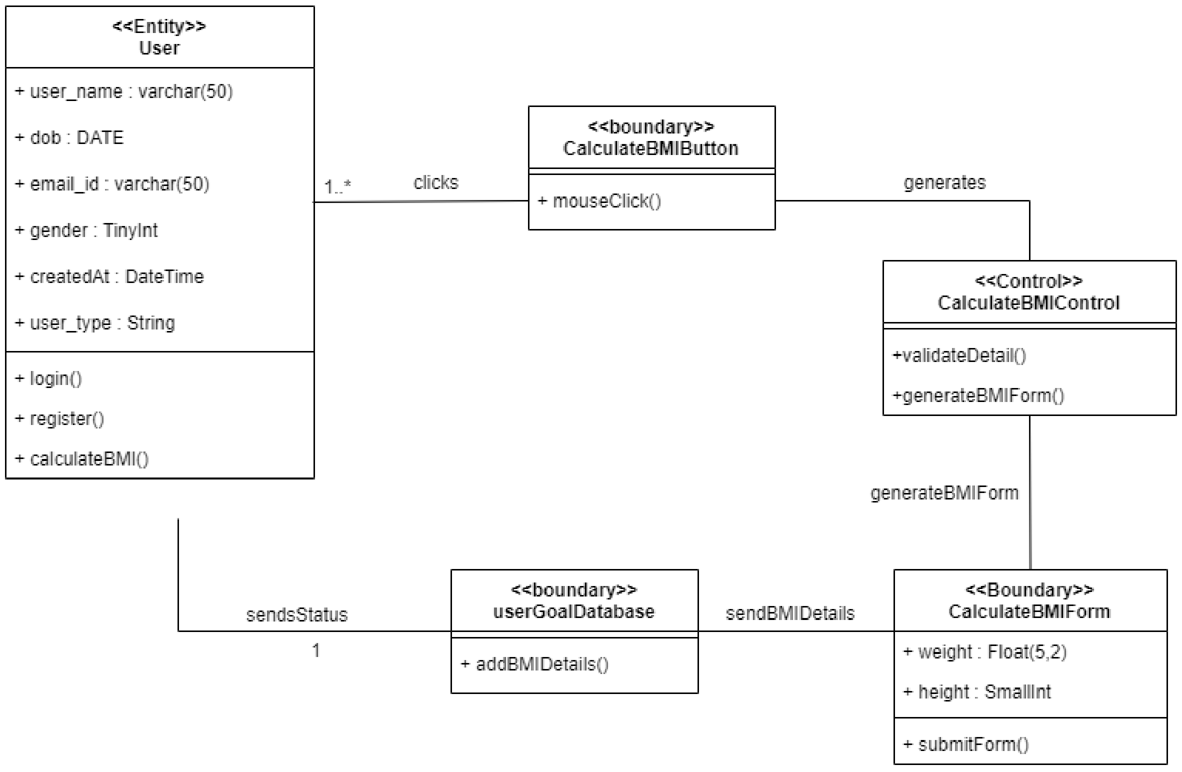
Register



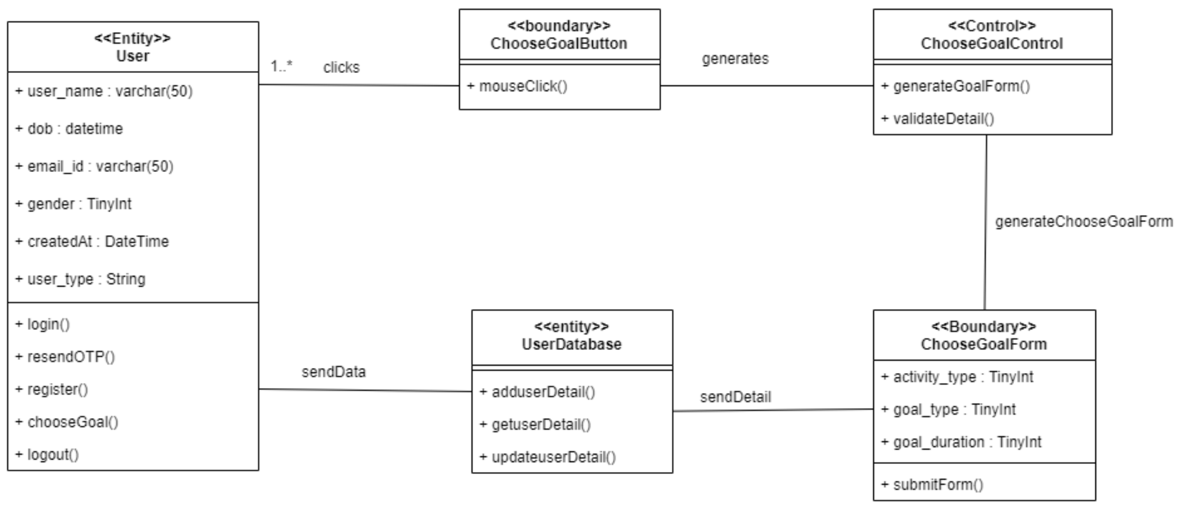
Modify Profile



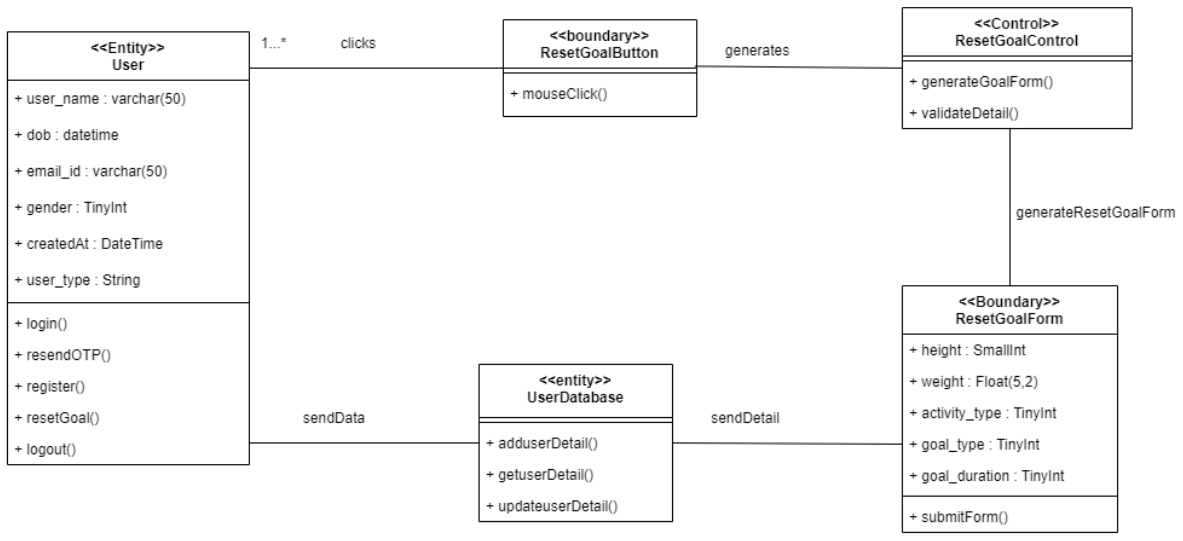
calculate BMI



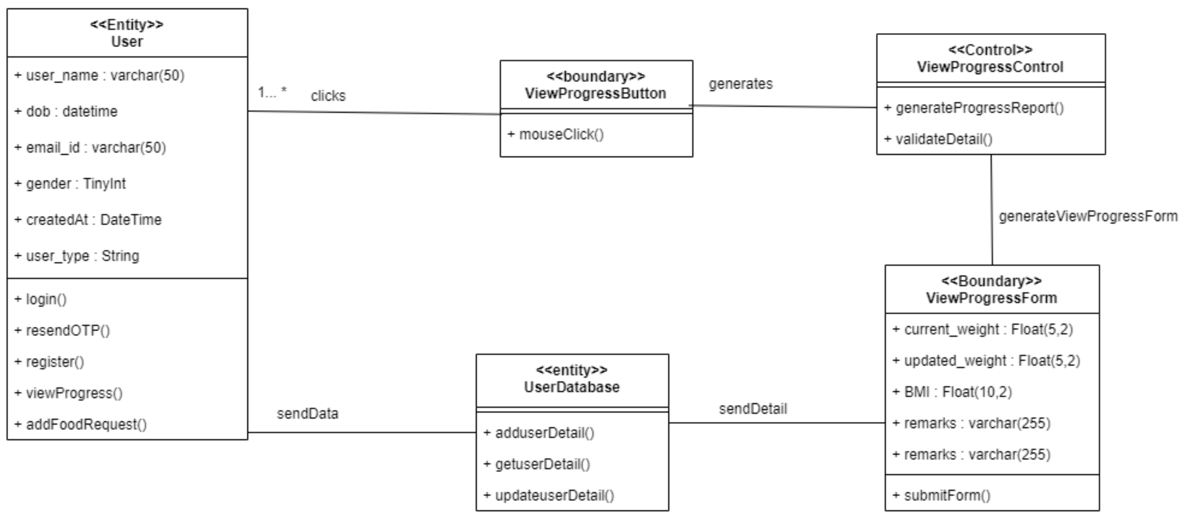
Choose Goal



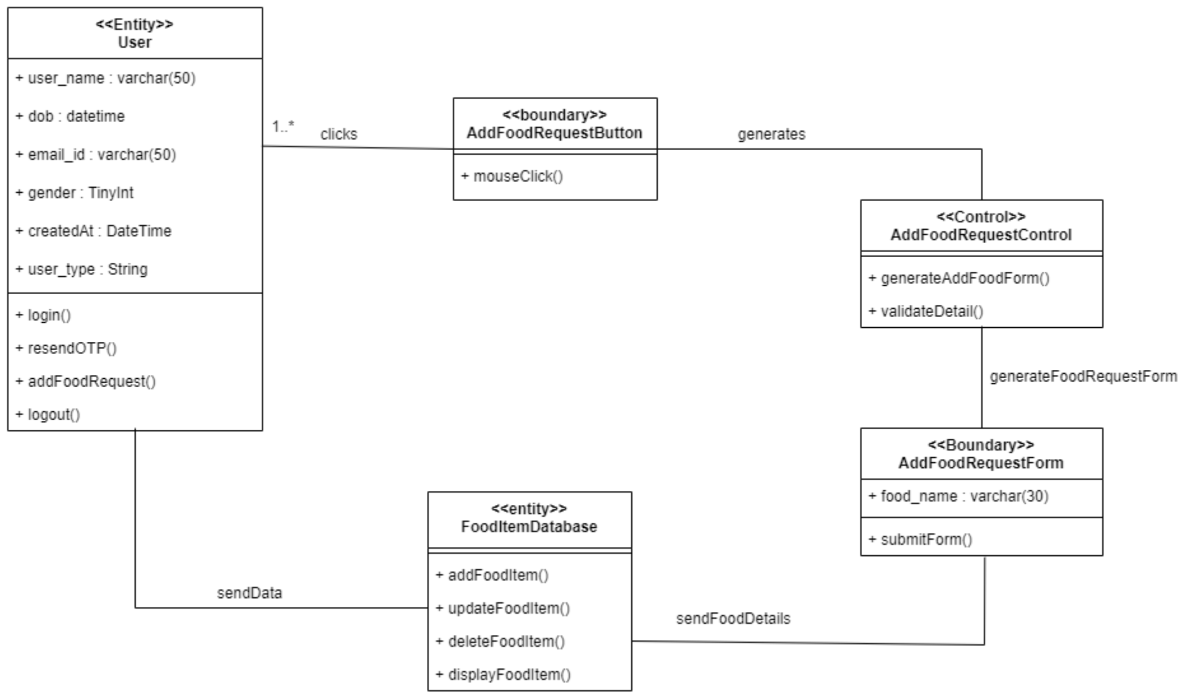
Reset Goal



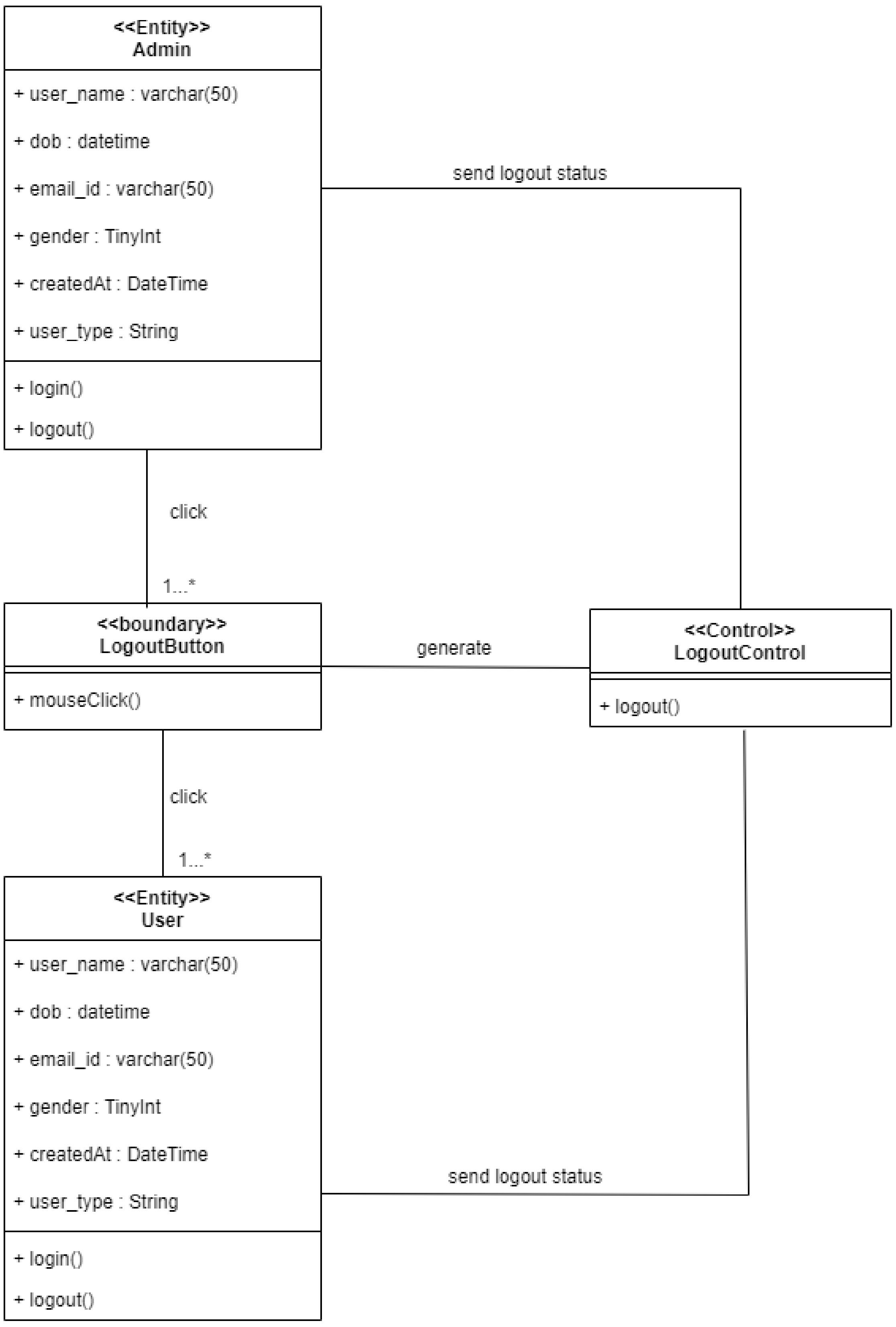
View Progress



Add Food Request

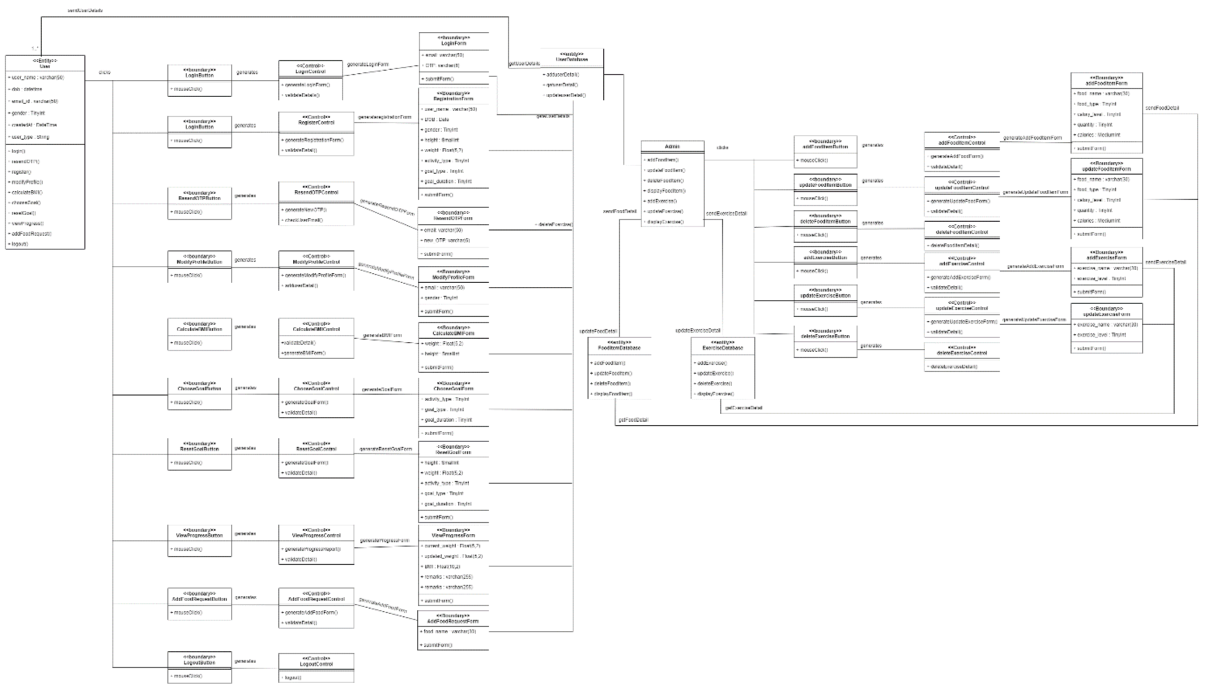


Logout



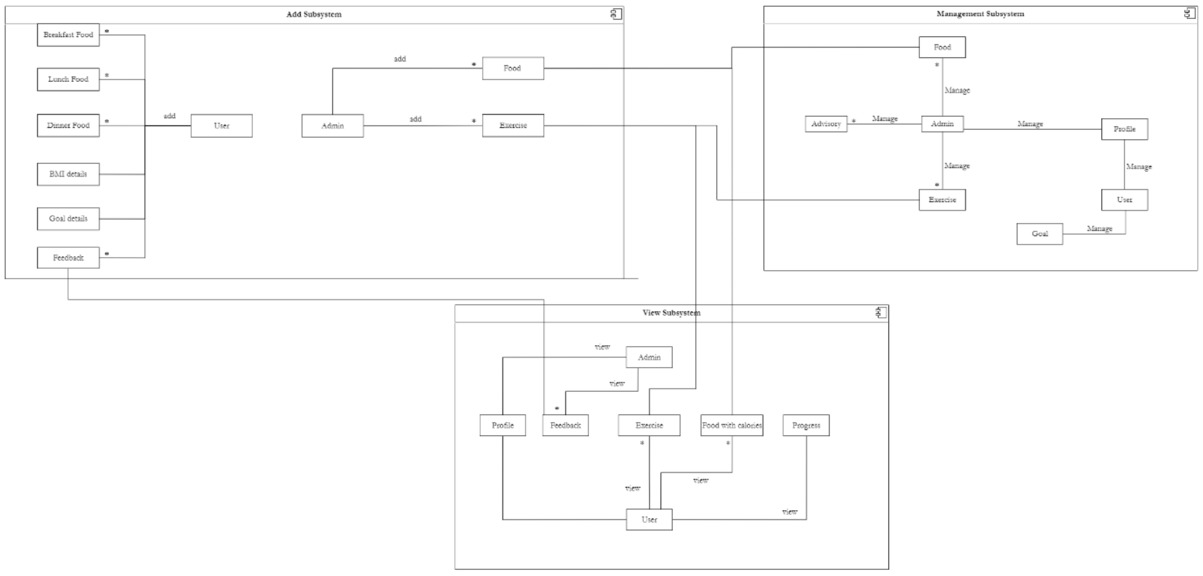
<<Entity>>
Admin
+ user_name : varchar(50)
+ dob : datetime
+ email_id : varchar(50)
+ gender : TinyInt
+ createdAt : DateTime
+ user_type : String
+ login()
+ logout()
send logout status
click
1...*
<<boundary>>
LogoutButton
+ mouseClick()
generate
<<Control>>
LogoutControl
+ logout()
click
1...*
<<Entity>>
User
+ user_name : varchar(50)
+ dob : datetime
+ email_id : varchar(50)
+ gender : TinyInt
+ createdAt : DateTime
+ user_type : String
+ login()
+ logout()
send logout status

5.2 Complete Analysis Diagram

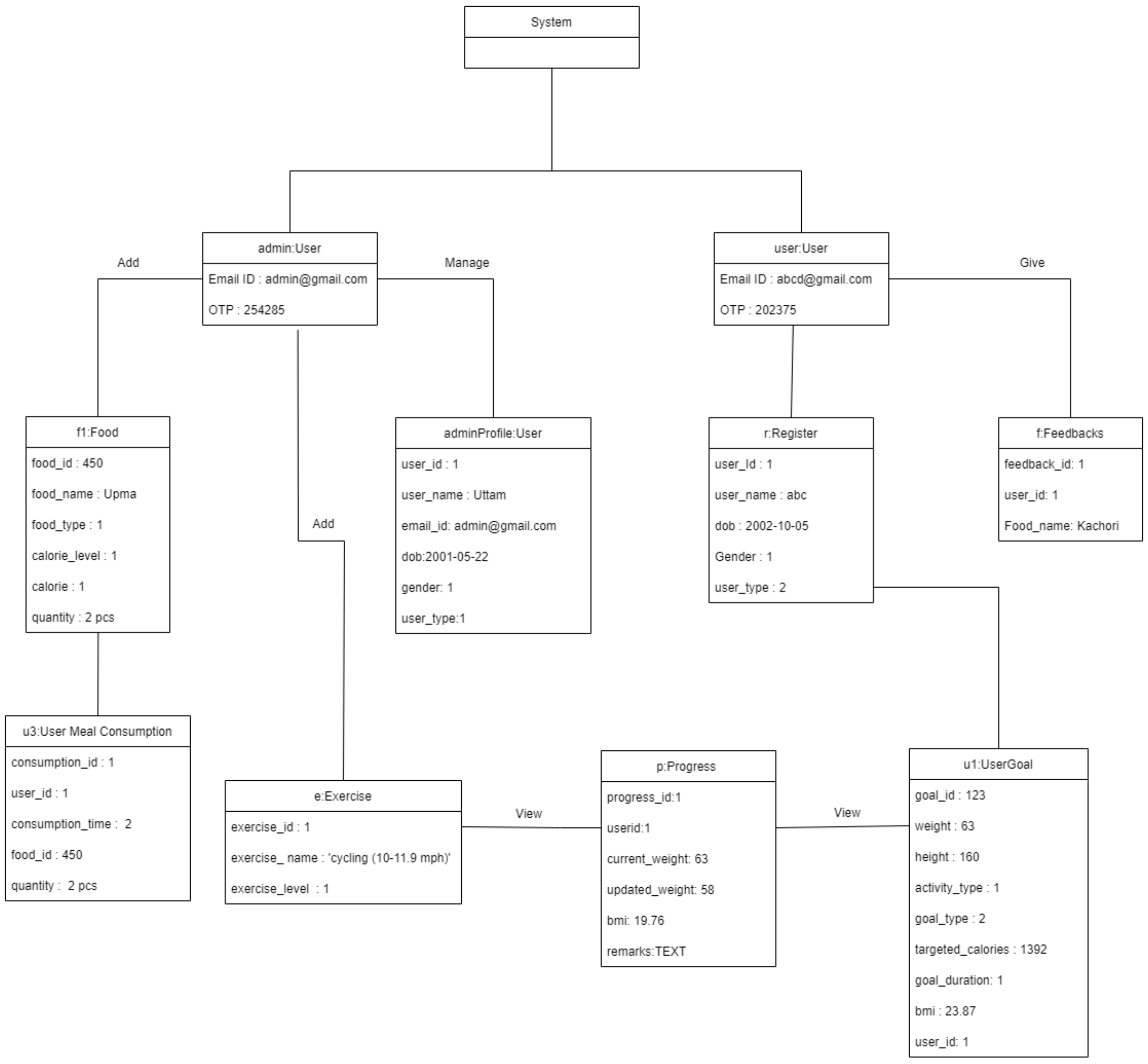


6. System Design

6.1 Sub System Design



6.2 Object Design



7. Testing Plan

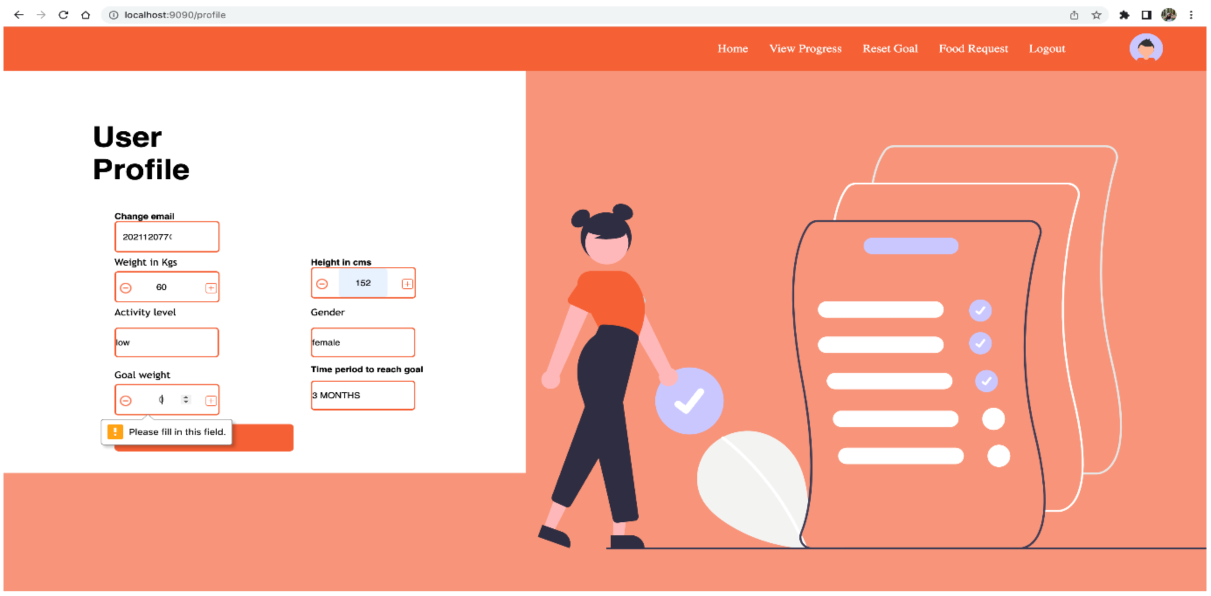
A **Test Plan** is a detailed document that describes the test strategy, objectives, schedule, estimation, deliverables, and resources required to perform testing for a software product. Test Plan helps us determine the effort needed to validate the quality of the application under test.

White Box Testing:- Unit Testing Each unit test is tested by the thunder client. This process is done by testing each module of our software project. Through unit testing, we have confirmed the performance of each unit component. Integration Testing:- In integration testing, the testing is done on combined unit test cases. Through integration testing, we have obtained the results about whether the modules work in combination or not.

Black box testing:- 24 In black-box testing, the input is taken from the frontend part and tested whether it fulfills the required requirements. This is done by taking the input from the tester and checking the conditions and based on that some output is generated. If the input satisfies the required requirements then the data flow will not be affected. If the input does not satisfy the required requirements then due to validations the data flow is affected and an alert message is displayed stating the corresponding message.

8. Testing Stratagies and Framework – White Box Testing and Black Box Testing

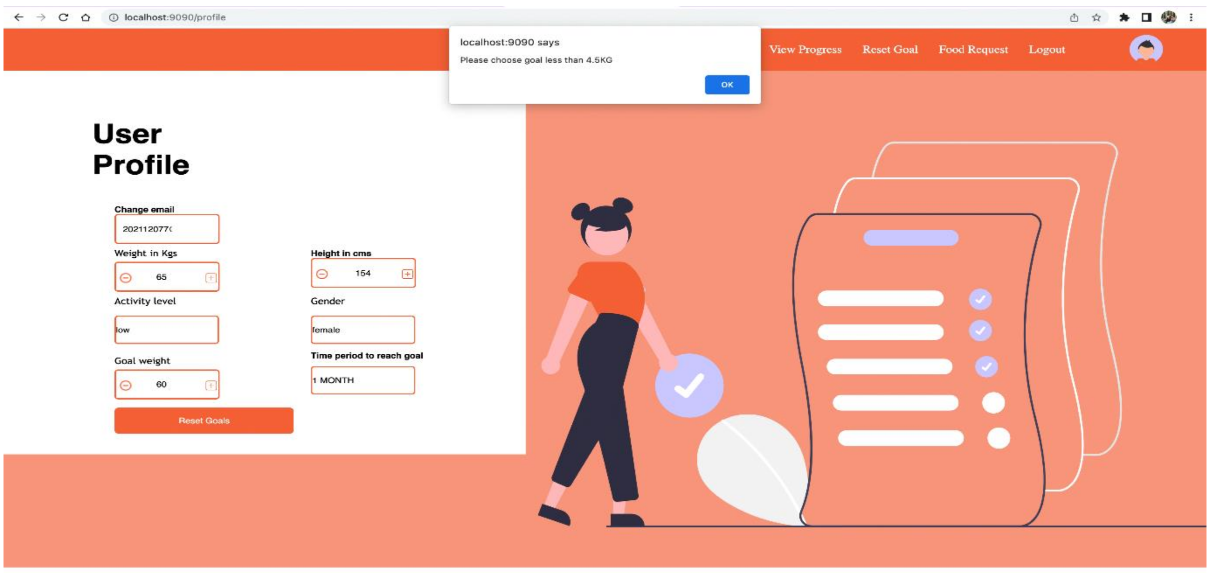
Black Box Testing



User Profile

Change email	202112077(
Weight in Kgs	60
Height in cms	152
Activity level	low
Gender	female
Goal weight	
Time period to reach goal	3 MONTHS

Please fill in this field.



localhost:9090 says

Please choose goal less than 4.5KG

User Profile

Change email: 202112077(

Weight in Kgs: 65

Height in cms: 154

Activity level: low

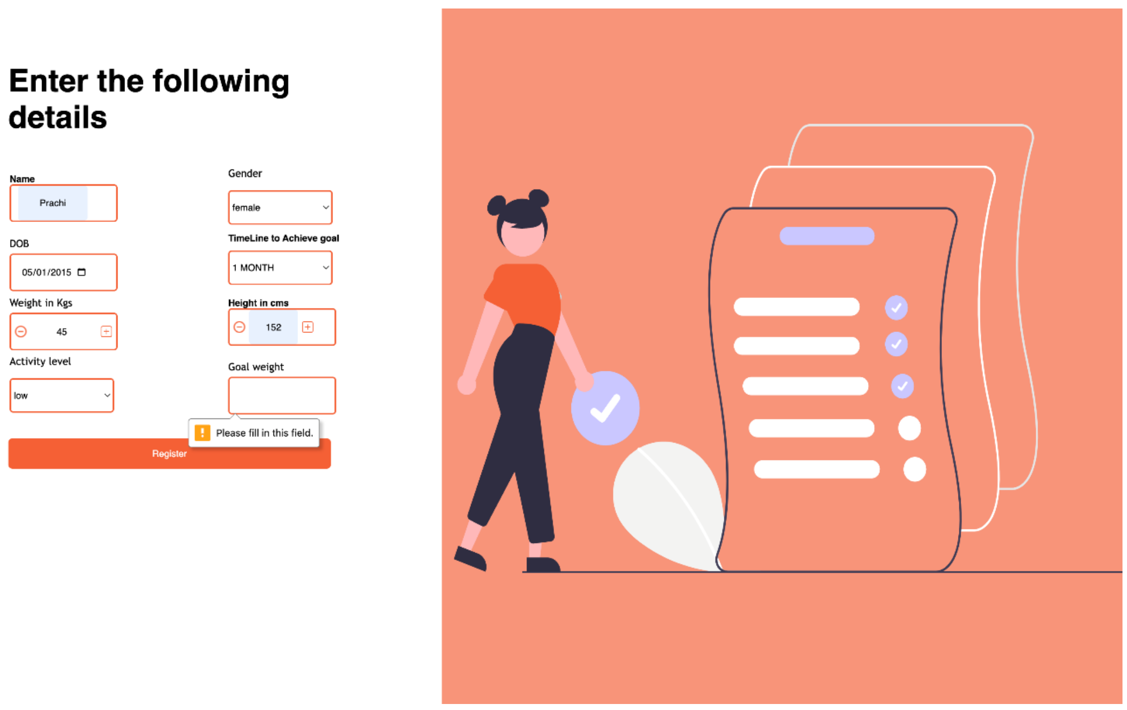
Gender: female

Goal weight: 60

Time period to reach goal: 1 MONTH

User Profile

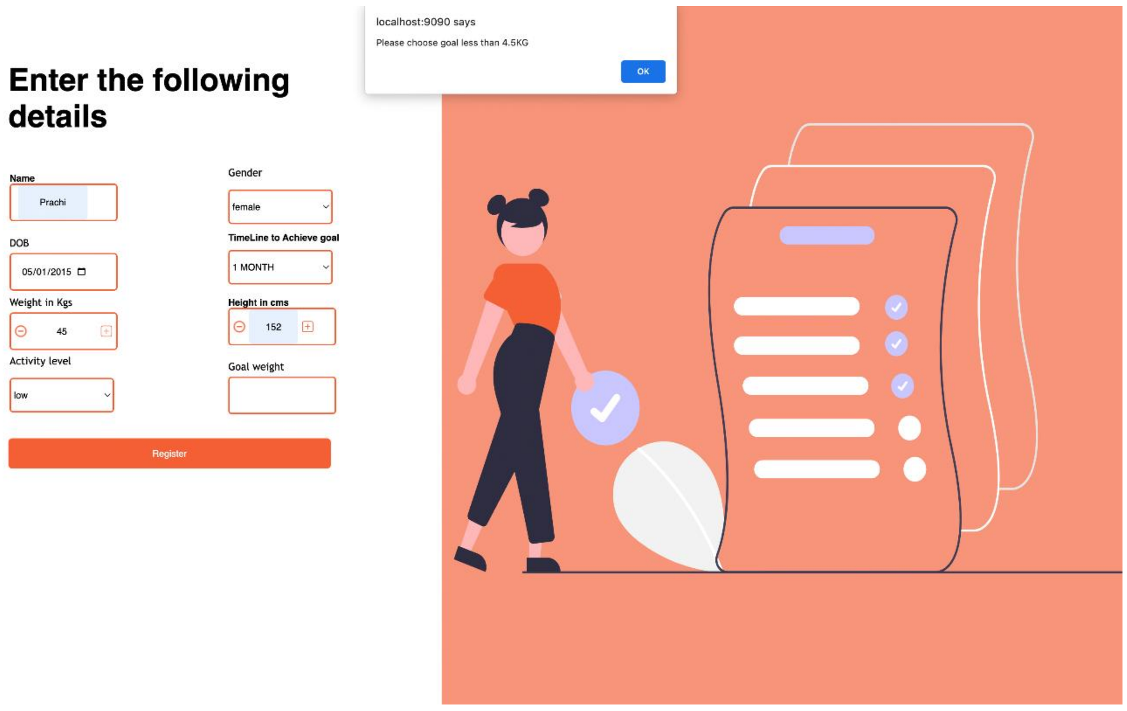
Test Action and Input Data	Expected outcome
Goal Weight: 60; Current Weight in kgs: 65; timeline to achieve goal: 1M	Alert Message: Max weight loss can be 4.5kgs in a month.
Goal Weight: 50; Current Weight in kgs: 60; timeline to achieve goal: 2M	Alert Message: Max weight loss can be 4.5kgs in a month.
Goal Weight: 50; Current Weight in kgs: 52; timeline to achieve goal: 1M	Registration Successful
Goal Weight: 50; Current Weight in kgs: 60; timeline to achieve goal: 3M	Registration Successful
Goal Weight: “empty”; Current Weight in kgs: 60; timeline to achieve goal: 3M	Alert box: Please fill this field



Enter the following details

Name	Prachi
Gender	female
DOB	05/01/2015
TimeLine to Achieve goal	1 MONTH
Weight in Kgs	45
Height in cms	152
Activity level	low
Goal weight	

Please fill in this field.



localhost:9090 says

Please choose goal less than 4.5KG

Enter the following details

Name	Prachi
Gender	female
DOB	05/01/2015
TimeLine to Achieve goal	1 MONTH
Weight in Kgs	45
Height in cms	152
Activity level	low
Goal weight	

Registration Page Black Box

Test Action and Input Data	Expected outcome
Goal Weight: 50; Current Weight in kgs: 55; timeline to achieve goal: 1M	Error Message: Max weight loss can be 4.5kgs in a month.

Goal Weight: 50; Current Weight in kgs: 52; timeline to achieve goal: 1M	Registration Successful
Goal Weight: 50; Current Weight in kgs: 60; timeline to achieve goal: 3M	Registration Successful

Request Food

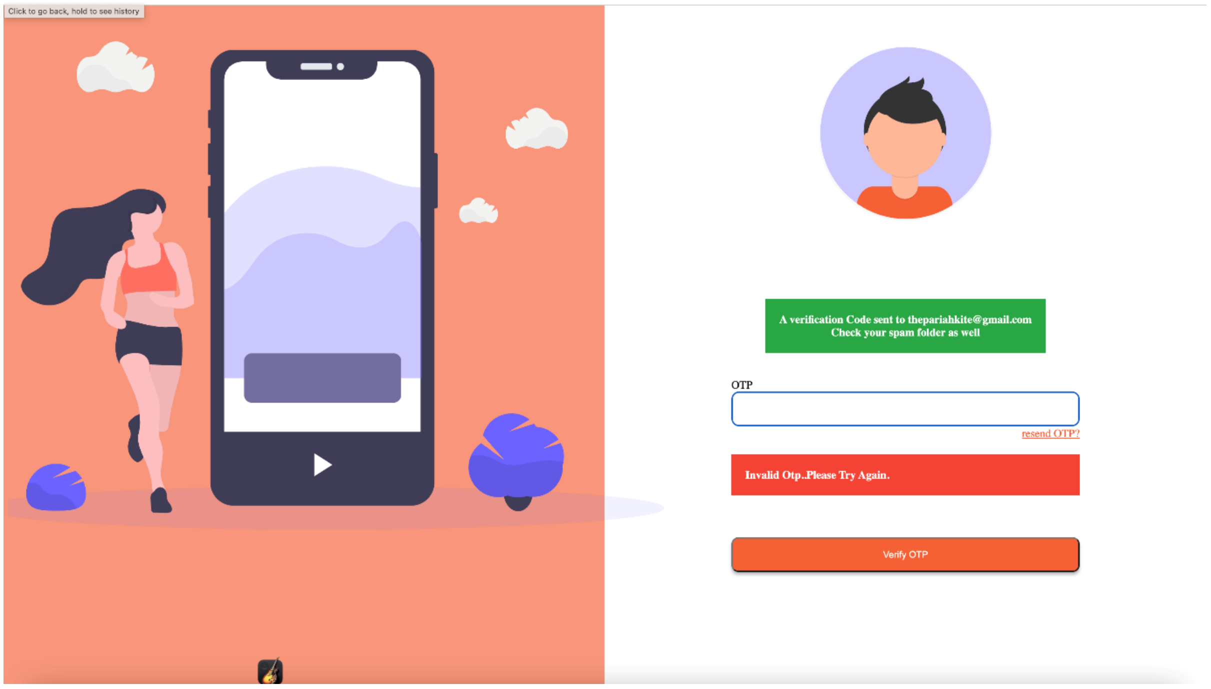
Test Action and Input Data	Expected outcome
blank textbox submits	Successful
food item: banana	Successful

Admin Enters food details

Test Action and Input Data	Expected outcome
Name: blank; Food type: losing; calorie level: high; qty: 10; calories: 100; meal time: Breakfast	Error: enter the food name
Name: dosa; Food type: losing; calorie level: high; qty: 10; calories: 100; meal time: Breakfast	Successful

Admin Enters Exercise Details

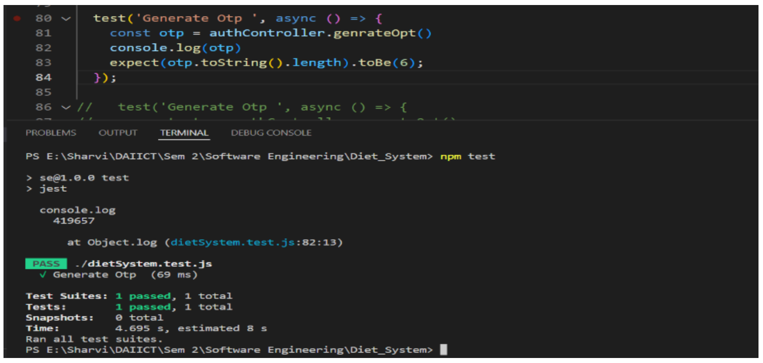
Test Action and Input Data	Expected outcome
Name: blank; Food type: losing; calorie level: high; qty: 10; calories: 100; meal time: Breakfast	Error: enter the food name
Name: dosa: Food type: losing; calorie level: high; qty: 10; calories: 100; meal time: Breakfast	Successful



A verification Code sent to thepariahkite@gmail.com
Check your spam folder as well

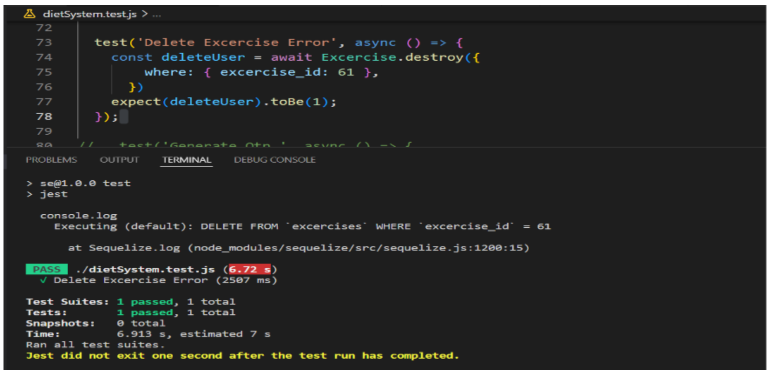
Invalid Otp..Please Try Again.

WhiteBox Testing



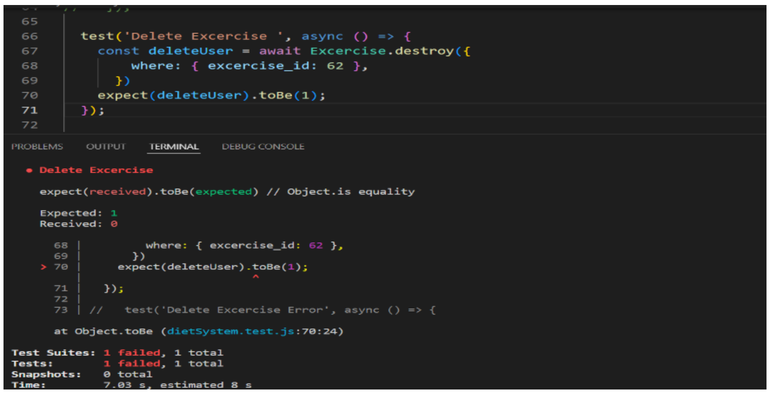
```
  test('Generate Otp ', async () => {
    const otp = authController.genrateOpt()
    console.log(otp)
    expect(otp.toString().length).toBe(6);
  });

// test('Generate Otp ', async () => {
```

```
PS E:\Sharvi\DAIICT\Sem 2\Software Engineering\Diet_System> npm test

> se@1.0.0 test
> jest

  console.log
    419657

      at Object.log (dietSystem.test.js:82:13)

 PASS  ./dietSystem.test.js
  √ Generate Otp  (69 ms)

Test Suites: 1 passed, 1 total
Tests:       1 passed, 1 total
Snapshots:   0 total
Time:        4.695 s, estimated 8 s
Ran all test suites.
PS E:\Sharvi\DAIICT\Sem 2\Software Engineering\Diet_System>
```



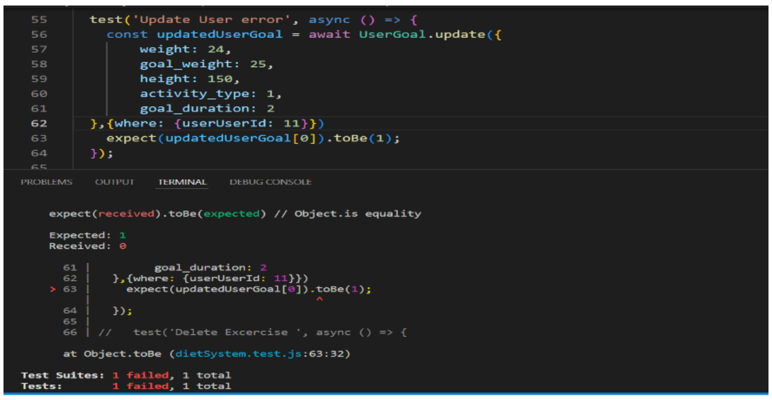
```

  test('Delete Excercise Error', async () => {
    const deleteUser = await Excercise.destroy({
        where: { excercise_id: 61 },
      })
    expect(deleteUser).toBe(1);
  });

```

```
> se@1.0.0 test
> jest

  console.log
    Executing (default): DELETE FROM `excercises` WHERE `excercise_id` = 61

      at Sequelize.log (node_modules/sequelize/src/sequelize.js:1200:15)

 PASS  ./dietSystem.test.js (6.72 s)
  √ Delete Excercise Error (2507 ms)

Test Suites: 1 passed, 1 total
Tests:       1 passed, 1 total
Snapshots:   0 total
Time:        6.913 s, estimated 7 s
Ran all test suites.
Jest did not exit one second after the test run has completed.
```



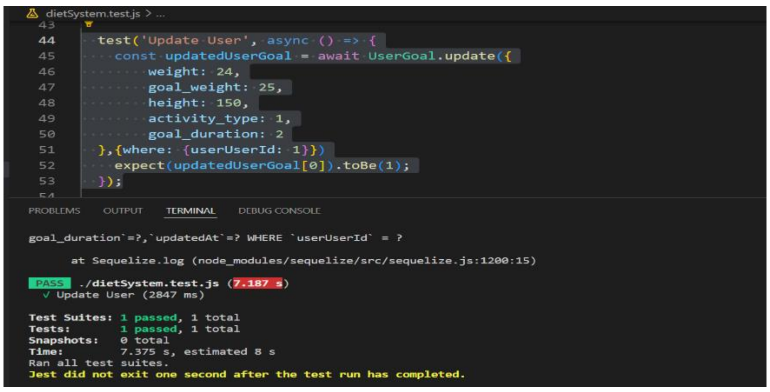
```
test('Delete Excercise ', async () => {
  const deleteUser = await Excercise.destroy({
      where: { excercise_id: 62 },
    })
  expect(deleteUser).toBe(1);
});
```

```
  ● Delete Excercise

    expect(received).toBe(expected) // Object.is equality

    Expected: 1
    Received: 0

      68 |         where: { excercise_id: 62 },
      69 |       })
    > 70 |     expect(deleteUser).toBe(1);
         |                        ^
      71 |   });
      72 |
      73 | //   test('Delete Excercise Error', async () => {

      at Object.toBe (dietSystem.test.js:70:24)

Test Suites: 1 failed, 1 total
Tests:       1 failed, 1 total
Snapshots:   0 total
Time:        7.03 s, estimated 8 s
```



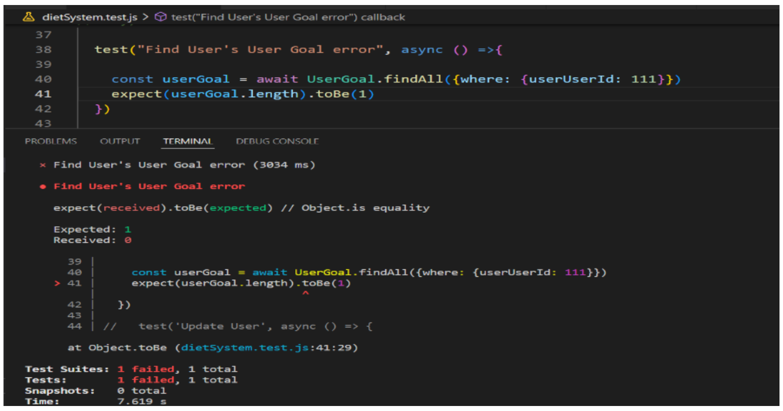
```
test('Update User error', async () => {
  const updatedUserGoal = await UserGoal.update({
      weight: 24,
      goal_weight: 25,
      height: 150,
      activity_type: 1,
      goal_duration: 2
  },{where: {userUserId: 11}})
  expect(updatedUserGoal[0]).toBe(1);
});
```

```
    expect(received).toBe(expected) // Object.is equality

    Expected: 1
    Received: 0

      61 |           goal_duration: 2
      62 |     },{where: {userUserId: 11}})
    > 63 |       expect(updatedUserGoal[0]).toBe(1);
         |                                  ^
      64 |     });
      65 |
      66 | //   test('Delete Excercise ', async () => {

      at Object.toBe (dietSystem.test.js:63:32)

Test Suites: 1 failed, 1 total
Tests:       1 failed, 1 total
```



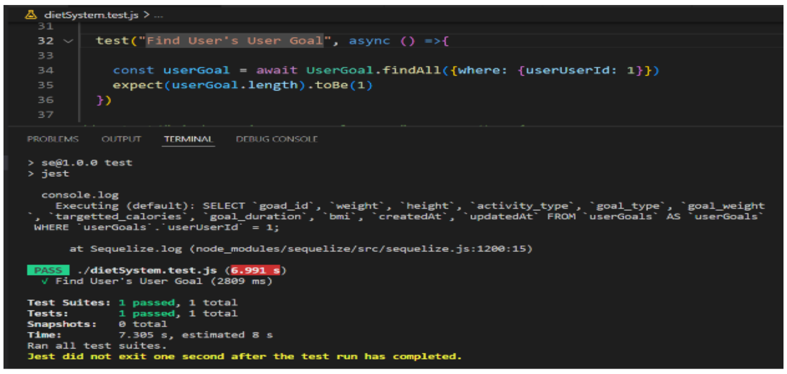
```
test('Update User', async () => {
  const updatedUserGoal = await UserGoal.update({
      weight: 24,
      goal_weight: 25,
      height: 150,
      activity_type: 1,
      goal_duration: 2
  },{where: {userUserId: 1}})
  expect(updatedUserGoal[0]).toBe(1);
});
```

```
goal_duration`=?,`updatedAt`=? WHERE `userUserId` = ?

      at Sequelize.log (node_modules/sequelize/src/sequelize.js:1200:15)

 PASS  ./dietSystem.test.js (7.187 s)
  √ Update User (2847 ms)

Test Suites: 1 passed, 1 total
Tests:       1 passed, 1 total
Snapshots:   0 total
Time:        7.375 s, estimated 8 s
Ran all test suites.
Jest did not exit one second after the test run has completed.
```


dietSystem.test.js > test("Find User's User Goal error") callback
PROBLEMS
OUTPUT
TERMINAL
DEBUG CONSOLE

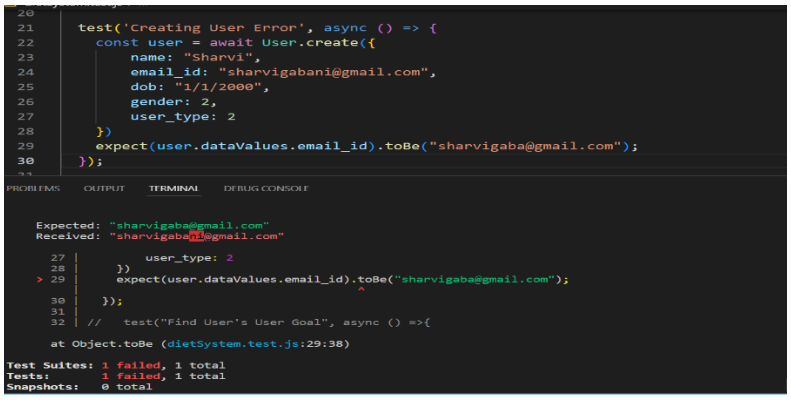
```

test("Find User's User Goal error", async () =>{

  const userGoal = await UserGoal.findAll({where: {userUserId: 111}})
  expect(userGoal.length).toBe(1)
})
```

```
  × Find User's User Goal error (3034 ms)

  ● Find User's User Goal error

    expect(received).toBe(expected) // Object.is equality

    Expected: 1
    Received: 0

      39 |
      40 |       const userGoal = await UserGoal.findAll({where: {userUserId: 111}})
    > 41 |       expect(userGoal.length).toBe(1)
         |                               ^
      42 |     })
      43 |
      44 | //   test('Update User', async () => {

      at Object.toBe (dietSystem.test.js:41:29)

Test Suites: 1 failed, 1 total
Tests:       1 failed, 1 total
Snapshots:   0 total
Time:        7.619 s
```



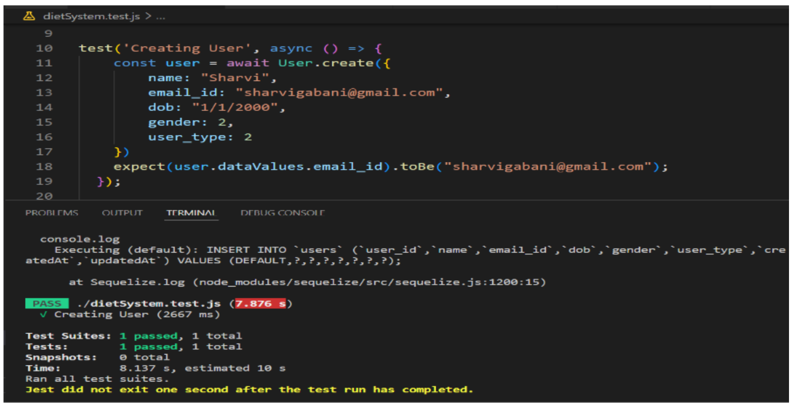
```javascript
test("Find User's User Goal", async () =>{

  const userGoal = await UserGoal.findAll({where: {userUserId: 1}})
  expect(userGoal.length).toBe(1)
})
```

```
> se@1.0.0 test
> jest

  console.log
    Executing (default): SELECT `goad_id`, `weight`, `height`, `activity_type`, `goal_type`, `goal_weight
`, `targetted_calories`, `goal_duration`, `bmi`, `createdAt`, `updatedAt` FROM `userGoals` AS `userGoals`
 WHERE `userGoals`.`userUserId` = 1;

      at Sequelize.log (node_modules/sequelize/src/sequelize.js:1200:15)

 PASS  ./dietSystem.test.js (6.991 s)
  √ Find User's User Goal (2809 ms)

Test Suites: 1 passed, 1 total
Tests:       1 passed, 1 total
Snapshots:   0 total
Time:        7.305 s, estimated 8 s
Ran all test suites.
Jest did not exit one second after the test run has completed.
```



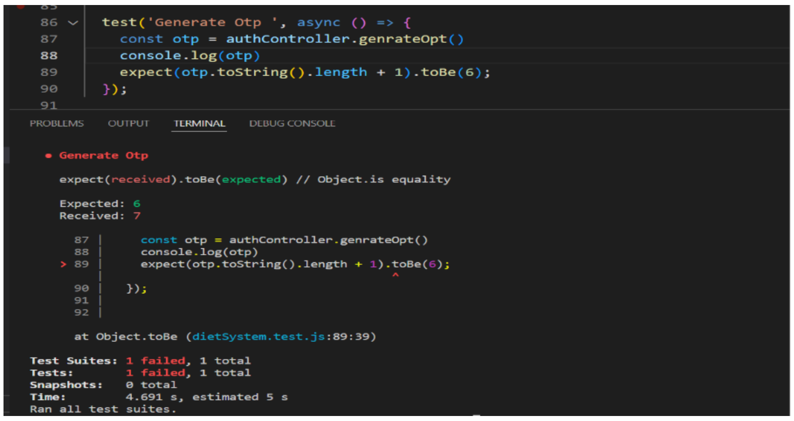
```javascript
test('Creating User Error', async () => {
  const user = await User.create({
      name: "Sharvi",
      email_id: "sharvigabani@gmail.com",
      dob: "1/1/2000",
      gender: 2,
      user_type: 2
  })
  expect(user.dataValues.email_id).toBe("sharvigaba@gmail.com");
});
```

```
    Expected: "sharvigaba@gmail.com"
    Received: "sharvigabani@gmail.com"

      27 |         user_type: 2
      28 |     })
    > 29 |     expect(user.dataValues.email_id).toBe("sharvigaba@gmail.com");
         |                                      ^
      30 |   });
      31 |
      32 | //   test("Find User's User Goal", async () =>{

      at Object.toBe (dietSystem.test.js:29:38)

Test Suites: 1 failed, 1 total
Tests:       1 failed, 1 total
Snapshots:   0 total
```



```javascript
test('Creating User', async () => {
    const user = await User.create({
        name: "Sharvi",
        email_id: "sharvigabani@gmail.com",
        dob: "1/1/2000",
        gender: 2,
        user_type: 2
    })
    expect(user.dataValues.email_id).toBe("sharvigabani@gmail.com");
  });
```

```
  console.log
    Executing (default): INSERT INTO `users` (`user_id`,`name`,`email_id`,`dob`,`gender`,`user_type`,`cre
atedAt`,`updatedAt`) VALUES (DEFAULT,?,?,?,?,?,?,?);

      at Sequelize.log (node_modules/sequelize/src/sequelize.js:1200:15)

 PASS  ./dietSystem.test.js (7.876 s)
  √ Creating User (2667 ms)

Test Suites: 1 passed, 1 total
Tests:       1 passed, 1 total
Snapshots:   0 total
Time:        8.137 s, estimated 10 s
Ran all test suites.
Jest did not exit one second after the test run has completed.
```



```javascript
test('Generate Otp ', async () => {
  const otp = authController.genrateOpt()
  console.log(otp)
  expect(otp.toString().length + 1).toBe(6);
});
```

```
  ● Generate Otp

    expect(received).toBe(expected) // Object.is equality

    Expected: 6
    Received: 7

      87 |     const otp = authController.genrateOpt()
      88 |     console.log(otp)
    > 89 |     expect(otp.toString().length + 1).toBe(6);
         |                                       ^
      90 |   });
      91 |
      92 |

      at Object.toBe (dietSystem.test.js:89:39)

Test Suites: 1 failed, 1 total
Tests:       1 failed, 1 total
Snapshots:   0 total
Time:        4.691 s, estimated 5 s
Ran all test suites.
```

9. Challenged We Faced

9.1 Open Issue

Pros:

- Gives you an idea (if used correctly) of how much you have eaten during the day, or how many calories are in a certain food or drink. People notoriously underestimate how much they eat.
- Gives you immediate feedback so you can adjust your food/drink intake accordingly for your individual circumstances and goals.
- For people who like routines and targets, having to log food and activity each day can help them stick to their plan and provide a framework and structure.
- Now you can see how many calories in food and drink are consumed as well as how much protein, carbs, and fats are. From here you can adjust your targets. Be wary though some of the nutrient data entered are incomplete or incorrect.
- You can use it to try and improve habits, see how many calories are in certain foods and drinks, then change your habits and routines accordingly.
- For more advanced dieters this can allow you to hit specific calorie and macro targets to help achieve the best body composition and nutrient targets for sporting goals.

Cons:

- Misreporting- This can be anything from weighing inaccuracies, the wrong portion sizes, or consciously/subconsciously not entering food and drink consumed.
- Some may find it painful to have to log every day or plan their intake. People can end up forgetting what they have.
- Users may end up getting so focused on the numbers, ending up a bit neurotic about the diet and numbers. Social situations may become daunting as you don't want to mess up the calorie and macro targets
- Intuitive eating: Intuitive eating is the idea you should eat when you are hungry and stop when you are full stop eating when you start feeling. Some may rely on the app and just the numbers instead of listening to your body's hunger signals.

10. Lesson Learned

Identify:

This is done by revising what went well, what didn't go well and what needs to be improved. This should be done during lesson learning participated by the key stakeholders of the project. In this we summarized the result and analysed them along with other key reports during the session to identify project failures and success.

Documentation (steps):

- We created a UI for the project
- Which gave us the good idea about what makes our project friendly to user
- After sending the UI to the internal and external project stakeholders, they started with implementation

In this we documented the results with a detailed report that included the participants feedback on the strength and weakness of the project and recommendations for the improvement and once the report got completed, it was shared with the team.

Analyse:

Analyse the document in order to determine how to apply them.

Archive:

Stores the documents in an easily accessible location for the project members. (i.e. Google Drive & GitHub).

Retrieve:

Refer to the documents for improvising the current project process and finalize the details. Organize the documents by creating folders for each type of project with their date and name.

11. Contribution

NAME	ID	WORK (Mid Evaluation)	WORK (After Mid Evaluation)
Faisal Ajmeri	202112031	User-Case: display progress,calculate calories Documentation:Define Users, Stakeholders and Timeline chart of the project.	Frontend Page: Feedback page BMI page
Karina Bavishi	202112052	Use-Case: Login,Register Documentation:Functional Requirements of project as per Final Use Case Diagram of the project	Progress Module Backend Documentation: Use case & description Analysis class diagram, exercise rearch
Sharvi Gabani	202112066	User-Case:Calculate BMI Documentation:Functional Requirements of project as per	Admin Module Backend Documentation:

		Final Use Case Diagram of the project	Use case & description Analysis class diagram, exercise rearch
Prachi Gole	202112077	User-Case: Feedback Documentation:Define Scope of the project, Describe which process model is used for your project and also explain why choose the specific process model for your project.	UI/UX figma prototype, Frontend admin Pages Documentation: FR,NFR,Process Model, black box testing
Nirmal Panchal	202112085	User-Case: Food Intake, calculate calories Documentation:Non-Functional Requirements of project	Frontend Pages : Final Goal Page Login page Documentation: PPT
Uttam Dobariya	202112094	User-Case: Reset goals ,User Logout Documentation:Functional Requirements of project as per Final Use Case Diagram of the project	Frontend Pages All progress pages, Daily Meal consumption page Documentation: System Design Object Design
Arohi Singhal	202112100	User-Case: display calories information,calculate calories Documentation:Define Scope of the project, Describe which process model is used for your project and also explain why choose the specific process model for your project.	Research about food analysis BMI and calories count Documentation: PPT,issues and challenges
Dhrumil Gohil	202112107	User-Case:Modify Profile, View Profile Documentation:Define Users, Stakeholders and Timeline chart of the project.	Login, Registration, BMI ,feedback Module Documentation: WhiteBox testing, BMI and BMR API
Nisarg Shah	202112111	User-Case: Type of Exercise, calculate calories Documentation:Define Scope of the project, Describe which process model is used for your project and also explain	Documentation: Scope introduction

		why choose the specific process model for your project.	
Ayush Garg	202112119	User-Case: user goal Documentation:Functional Requirements of project as per Final Use Case Diagram of the project	Progress Module Backend Documentation: Use case & stakeholder, Analysis class diagram, Google mail API
Mansi Jani	202112120	User-Case: display food details,calculate calories Documentation: Non-Functional Requirements of project	UserGoal Mudule Backend Documentation: System Design Object Design