FIT-ME APPLICATION

CL-219: Database Systems Lab

End of Semester Project

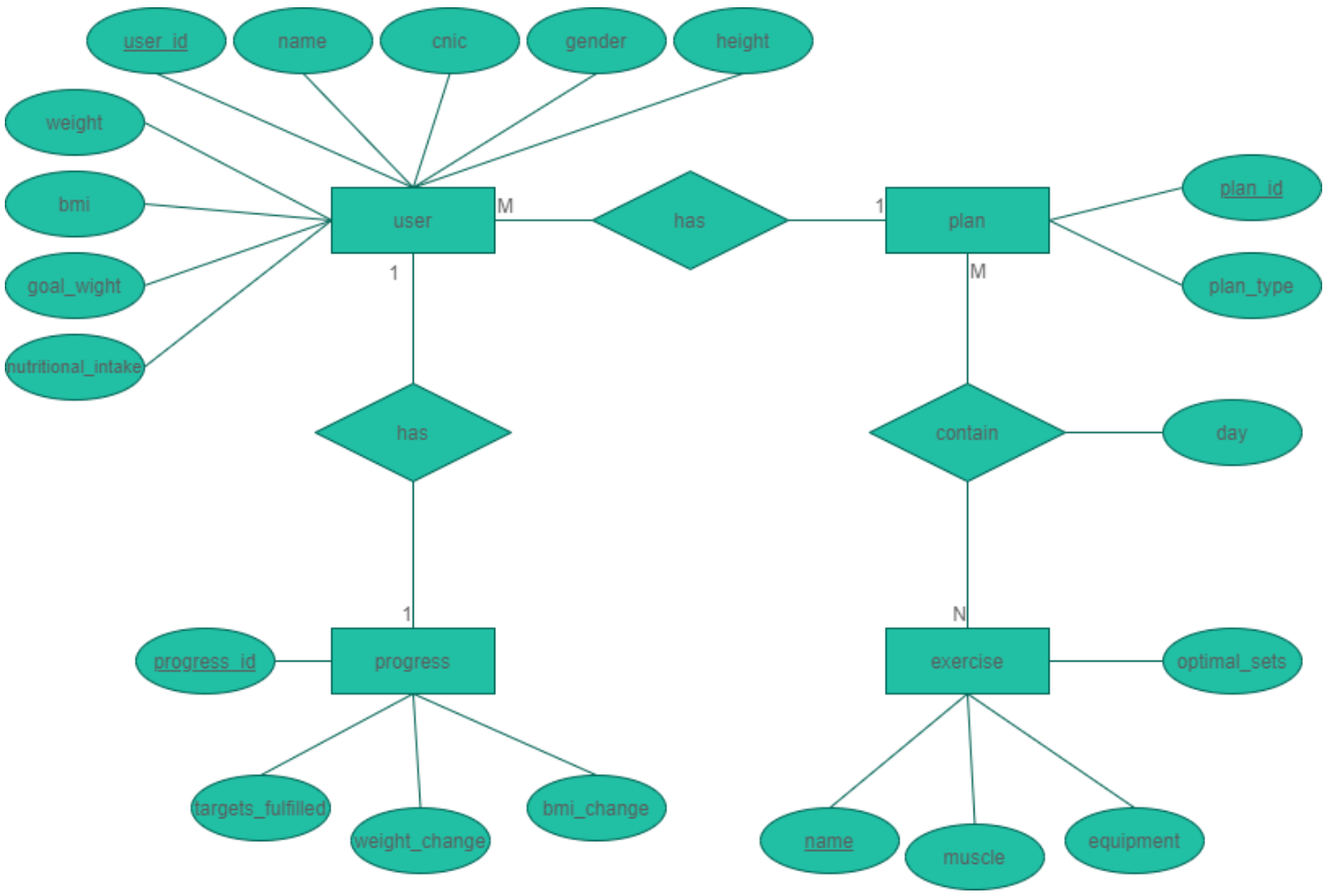
Hassan Shahzad 18i-0441

Azka Khurram 18i-0461

Zeeshan Ikram 18i-0660

TABLE OF CONTENTS

1: DATABASE ERD DIAGRAM:



Our schema consists of 5 **tables**. Each table is linked to each other via foreign keys or composite keys as shown in the above ERD Diagram. The logic behind the creation of tables is as follows:

- **User:** This table will contain all the details about a user. User table will have information about user's current data and goal weight etc. Furthermore, this table will also contain foreign keys of table Plan and Progress. Many users can have a single plan; hence the relation will be many-to-one between the user table and the plan table. Similarly, one user can have one progress only, hence the relation will be one-to-one between user and progress.

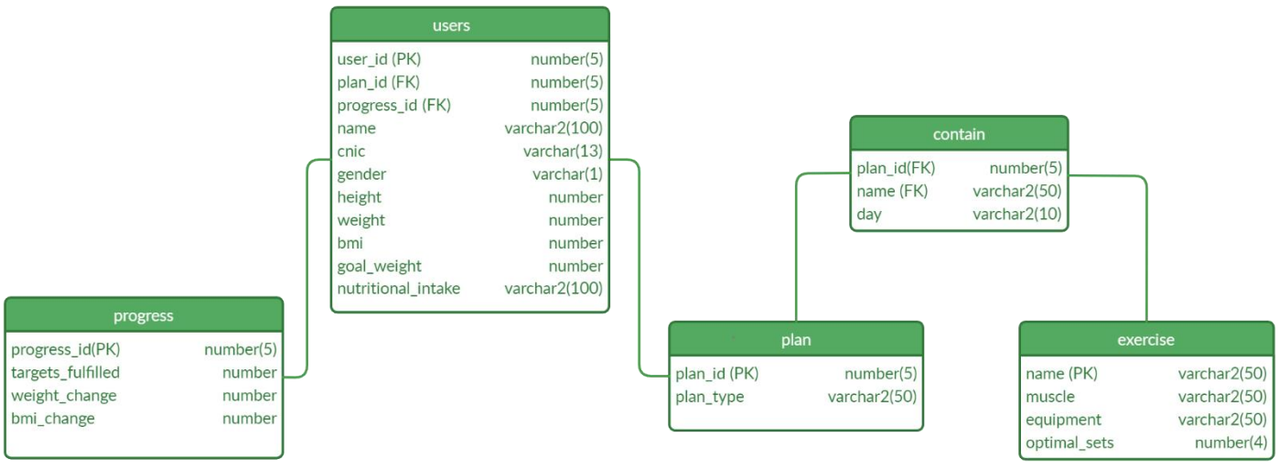
- **Plan:** Plan table will simply contain the type of plan (i.e., Gain or Loss) and the primary key.

- **Contains:** Contains table connects the Plan and Exercise table. One plan can have many exercises and many exercises can be in many plans. Hence the relation will be many-to-many.

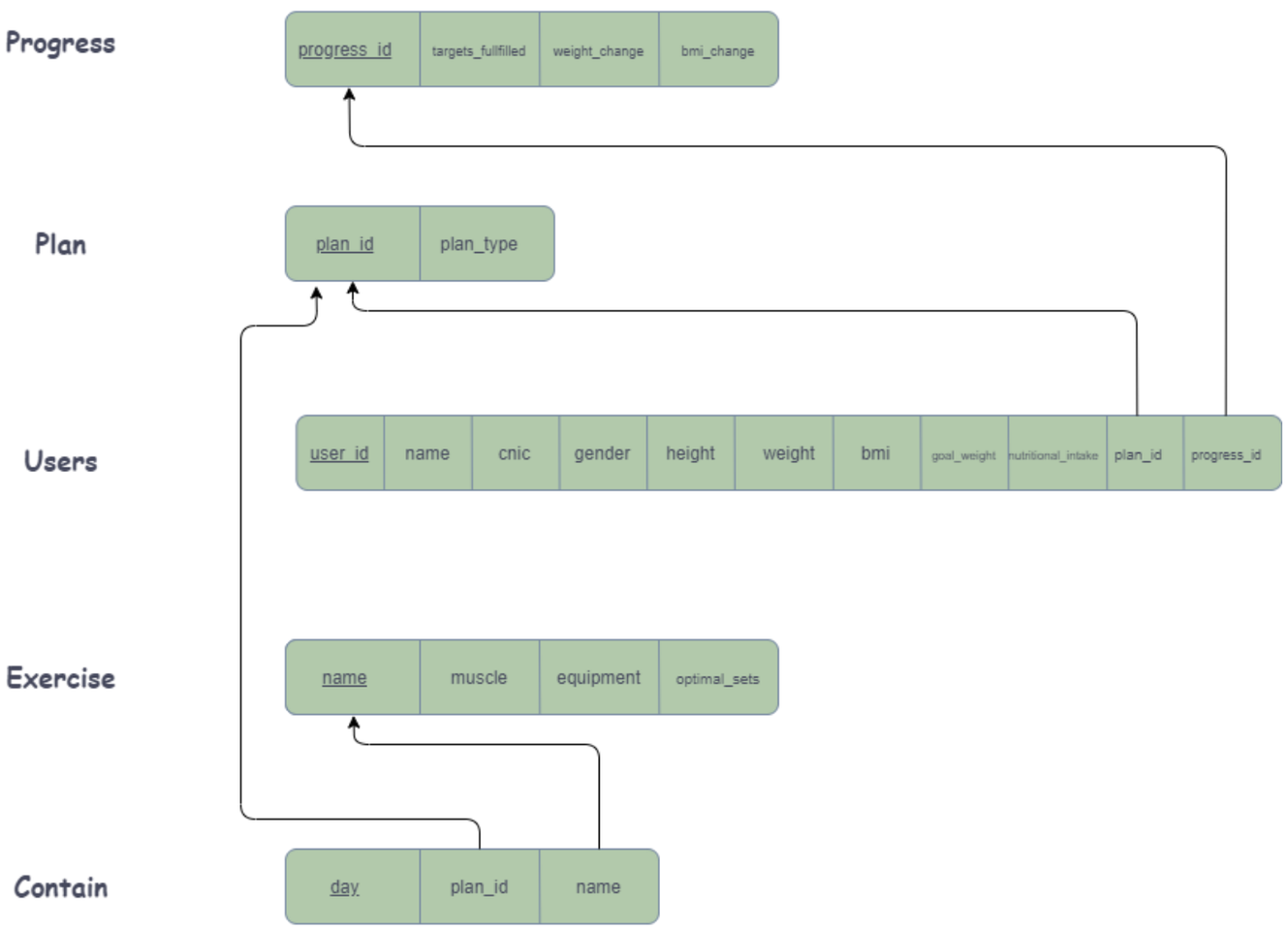
- **Progress:** This table will contain progress of a user. The information in this table can be updated when user updates his/her progress.

- **Exercise:** This table will contain the list of all of the exercises with their names and descriptions.

2: DATABASE SCHEMA DIAGRAM



3: RELATIONAL DATABASE SCHEMA:



4: TABLE DESCRIPTION:

Users			
Name	Description	Type	Constraints
User_id	Unique id assigned to each user	Number (5)	PK
Name	Name of the user	VARCHAR (50)	NotNull
Cnic	Cnic of the user	NUMBER (13)	NotNull
Gender	Gender of the user (M/F)	VARCHAR (1)	NotNull
Height	Height of the user	NUMBER	NotNull
Weight	Weight of the usr	NUMBER	NotNull
Bmi	Body Mass Index calculated automatically	NUMBER	
Goal_weight	The goal weight of the user	NUMBER	NotNull
Nutritional_intake	Nutritional Intake of the User	VARCHAR (100)	
Plan_id	Plan's unique id	NUMBER (5)	FK
Progress_id	Progress's unique id	NUMBER (5)	FK

Plan			
Name	Description	Type	Constraints
Plan_id	Unique id assigned to each plan	NUMBER (5)	PK
Plan_type	The type of plan (Gain/Loss)	VARCHAR (50)	NotNull

Exercise			
Name	Description	Type	Constraints
Name	The name of the exercise (E.g., Squats etc.)	VARCHAR (50)	PK
Muscle	The targeted muscle	VARCHAR (50)	NotNull
Equipment	Equipment Used for this exercise	VARCHAR (50)	
Optimal_sets	The optimal number of sets for an exercise	NUMBER (4)	NotNull

Contain			
Name	Description	Type	Constraints
Day	The day on which that particular exercise is to be scheduled	VARCHAR (10)	PK
Plan_id	Plan's unique id	NUMBER (5)	FK
Name	Exercise's unique name	NUMBER (5)	FK

Progress			
Name	Description	Type	Constraints
Progress_id	The unique id of the progress made by user	NUMBER (5)	PK
Targets_fullfilled	The number of targets fulfilled	NUMBER	
Weight_change	The change in weight	NUMBER	
Bmi_change	The BMI change	NUMBER	

5: QUERIES FOR TABLES CREATION AND CONSTRAINTS:

USERS:

```
create Table Users
(
        User_id number (5),
        Name varchar2(50) NOT NULL,
        CNIC number(13) NOT NULL UNIQUE,
        Gender varchar(1),
        Height number NOT NULL CHECK (Height>0),
        Weight number NOT NULL CHECK (Weight>0),
        BMI number,
        Goal_weight number NOT NULL,
        nutritional_intake varchar2(100),
        Plan_id number(5),
        Progress_id number(5),
        Primary Key(User_id),
        FOREIGN KEY (Plan_id) REFERENCES Plan(Plan_id),
        FOREIGN KEY (Progress_id) REFERENCES Progress(Progress_id)
);
```

PLAN:

```
create Table Plan
(
        Plan_id number(5),
        Plan_type varchar2 (50) NOT NULL,
        Primary Key(Plan_id)
);
```

EXERCISE:

```
create Table Exercise
(
        Name varchar2 (50) ,
        Muscle varchar2 (50) NOT NULL,
        Equipment varchar2 (50),
        Optimal_sets number (4) NOT NULL,
        Primary Key(Name)
);
```

CONTAIN:

```
create Table Contain
(
        Plan_id number(5),
        Name varchar2(50),
        Day varchar2(10),
        FOREIGN KEY (Plan_id) REFERENCES Plan(Plan_id),
        FOREIGN KEY (Name) REFERENCES Exercise(Name),
        constraint C1_pk primary key(Plan_id, Name)
);
```

PROGRESS:

```
create Table Progress
(
        Progress_id Number(5),
        Targets_fulfilled Number ,
        Weight_change Number,
        BMI_change Number,
        Primary Key(Progress_id)
);
```

6: QUERIES FOR DATA INSERTION:

USERS:

```
INSERT INTO Users VALUES (10001, 'Hassan Shahzad', '12345', 'M', 65, 65, 19, 62, '2000 Calories', 10001, 10001);

INSERT INTO Users VALUES (10002, 'Sana Ali', '12346', 'F', 48, 48, 19, 55, '2400 Calories', 10002, 10002);

INSERT INTO Users VALUES (10003, 'Azka Khurram', '12347', 'F', 60, 60, 55, 62, '3200 Calories', 10003, 10003);

INSERT INTO Users VALUES (10004, 'Zeeshan Ikram', '12348', 'M', 75, 67, 19, 78, '2600 Calories', 10004, 10004);
```

PLAN:

```
INSERT INTO Plan VALUES (10001, 'Loss');

INSERT INTO Plan VALUES (10002, 'Gain');

INSERT INTO Plan VALUES (10003, 'Loss');

INSERT INTO Plan VALUES (10004, 'Gain');
```

EXERCISE:

```
INSERT INTO Exercise VALUES ('Squats', 'Thigh', 'Gym Mat', '3' );

INSERT INTO Exercise VALUES ('Biceps', 'Arm', 'Dumbells', '3' );

INSERT INTO Exercise VALUES ('Pushups', 'Arm, Chest', 'Gym Mat', '3' );

INSERT INTO Exercise VALUES ('Pullups', 'Arm, Back', 'Metal Rod', '3' );

INSERT INTO Exercise VALUES ('Plank', 'Abdomen', 'Gym Mat', '1' );

INSERT INTO Exercise VALUES ('Crunches', 'Abdomen', 'Gym Mat', '3' );
```

CONTAIN:

```
INSERT INTO Contain VALUES (10001, 'Squats', 'Monday');

INSERT INTO Contain VALUES (10001, 'Crunches', 'Tuesday');

INSERT INTO Contain VALUES (10001, 'Plank', 'Wednesday');

INSERT INTO Contain VALUES (10002, 'Bicpes', 'Monday');

INSERT INTO Contain VALUES (10002, 'Pushups', 'Wednesday');

INSERT INTO Contain VALUES (10003, 'Crunches', 'Saturday');

INSERT INTO Contain VALUES (10004, 'Squats', 'Saturday');

INSERT INTO Contain VALUES (10004, 'Pushups', 'Sunday');
```

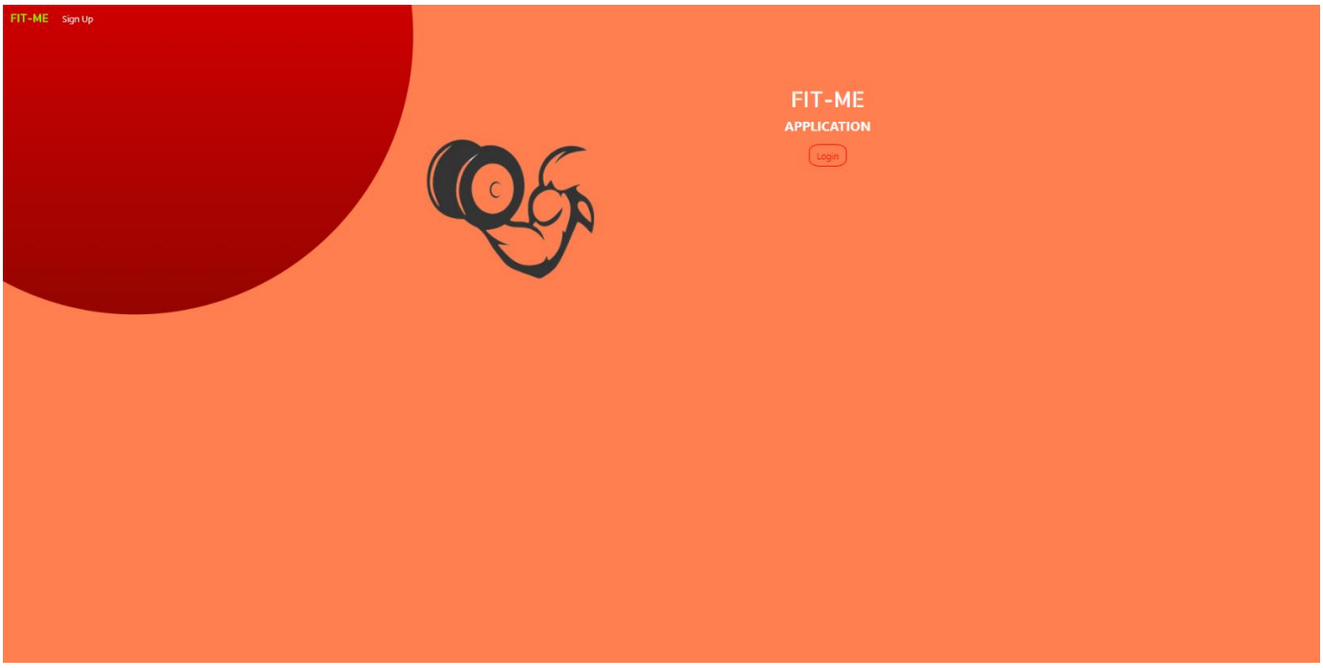
PROGRESS:

```
INSERT INTO Progress VALUES (10001, 60, 65, 18.5);

INSERT INTO Progress VALUES (10002, 85, 53, 18);

INSERT INTO Progress VALUES (10003, 40, 58, 18.7);

INSERT INTO Progress VALUES (10004, 32, 74, 19);
```

7: INTERFACE SCREENSHOTS AND EXPLANATION:

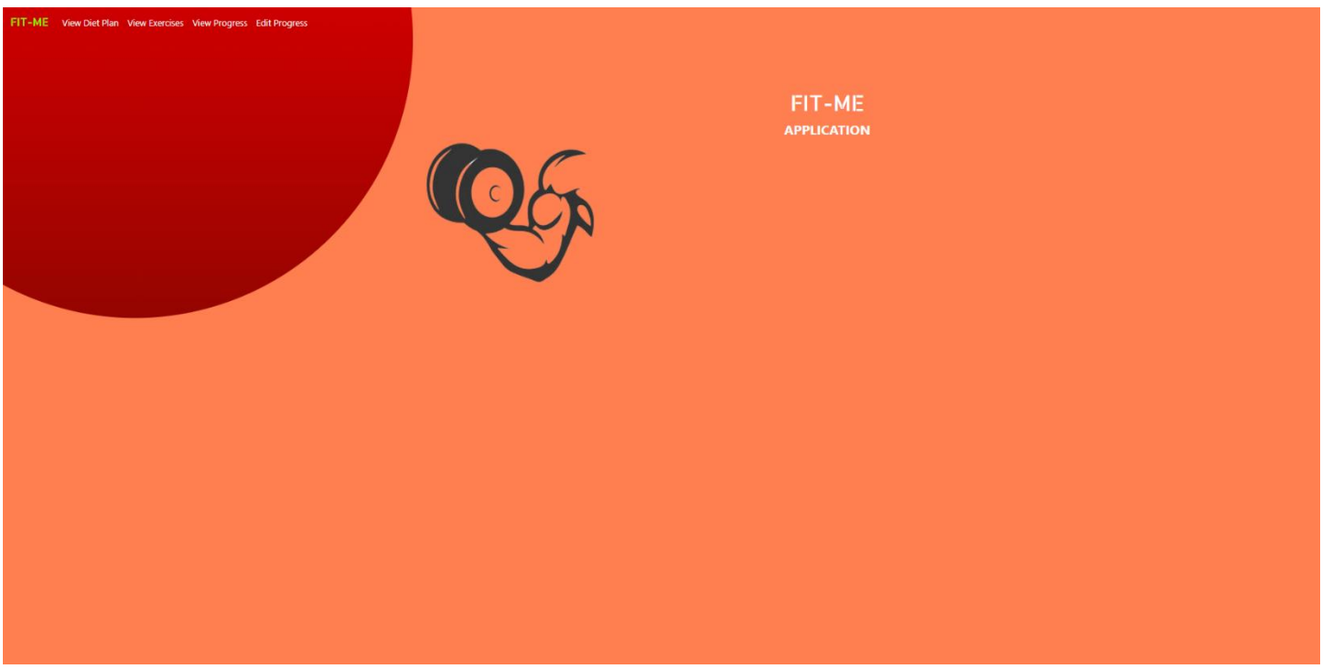
BOOTSTRAP:

We created 2 screens in bootstrap. One is the homepage where the user can either login or sign up (register). Upon logging in, this page directly leads to a second bootstrap page where the user can access all the reports and forms from. Attached are the screenshots of both the screens in the order described above:



In the above screen, we have also added a hover effect on the login button such that it changes its color whenever the mouse is placed on it.

Note: The font and button (in fact the overall interface) are looking small because of my PC's screen resolution. These were designed on a laptop.

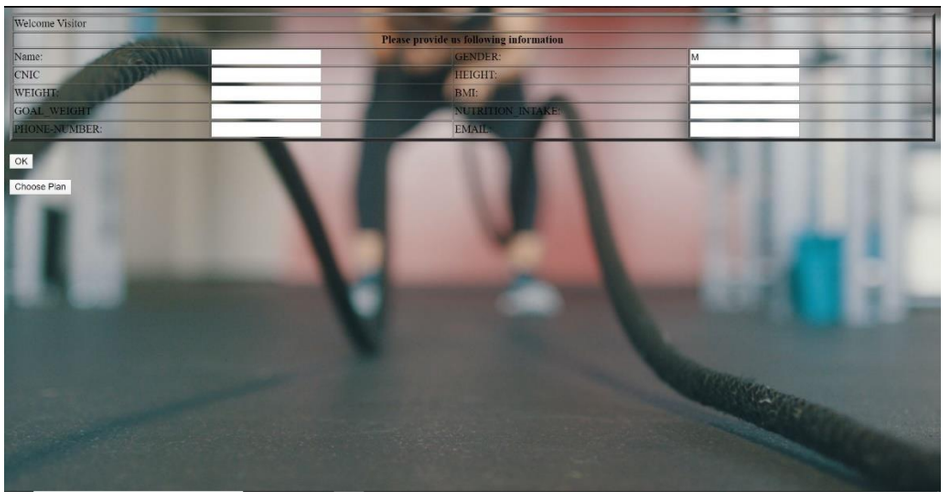


In the above screen, buttons can be seen that lead to those specific reports and forms. The remaining forms are linked to other forms, hence are not displayed directly. In order to access any report, the user must be logged in.

FORMS:

REGISTRATION

This form takes input of User information and stores it into the user table.



After pressing **OK,** the information will be stored.

CHOOSE PLAN TYPE

After pressing this button, the user will be redirected to the page where the user can choose the plan.

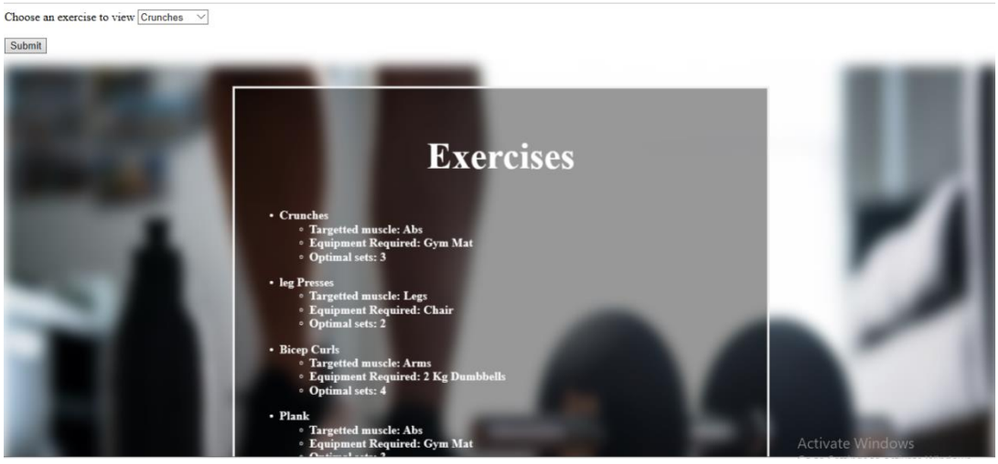
The user will press **Choose Plan**

CHOOSE PLAN

The plans with their ID will be displayed. The user will choose the plan from the above by pressing OK.

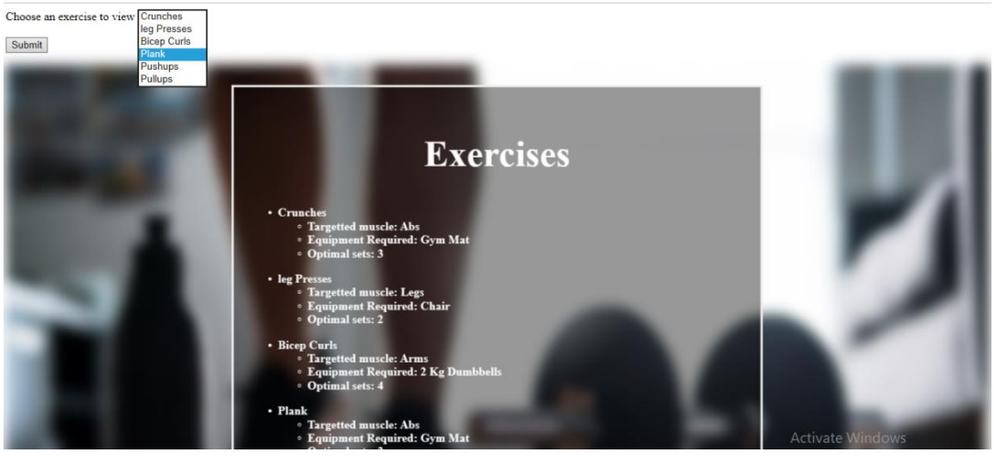
REPORTS:

EXERCISE_REPORT:

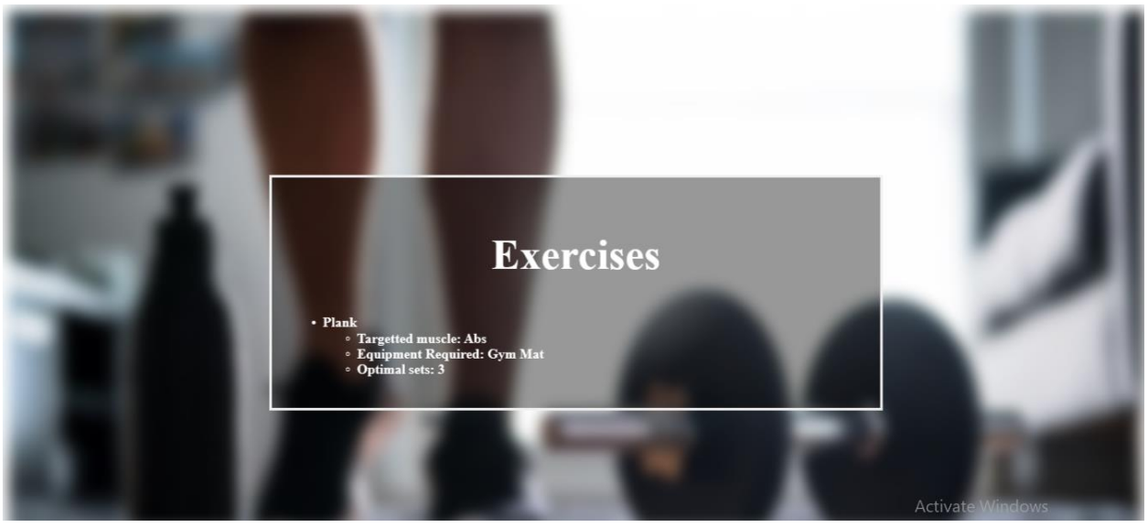


This page displays all the data regarding exercises stored in our data base. The information stored includes name of the exercises, targeted muscle, equipment required and optimal sets.

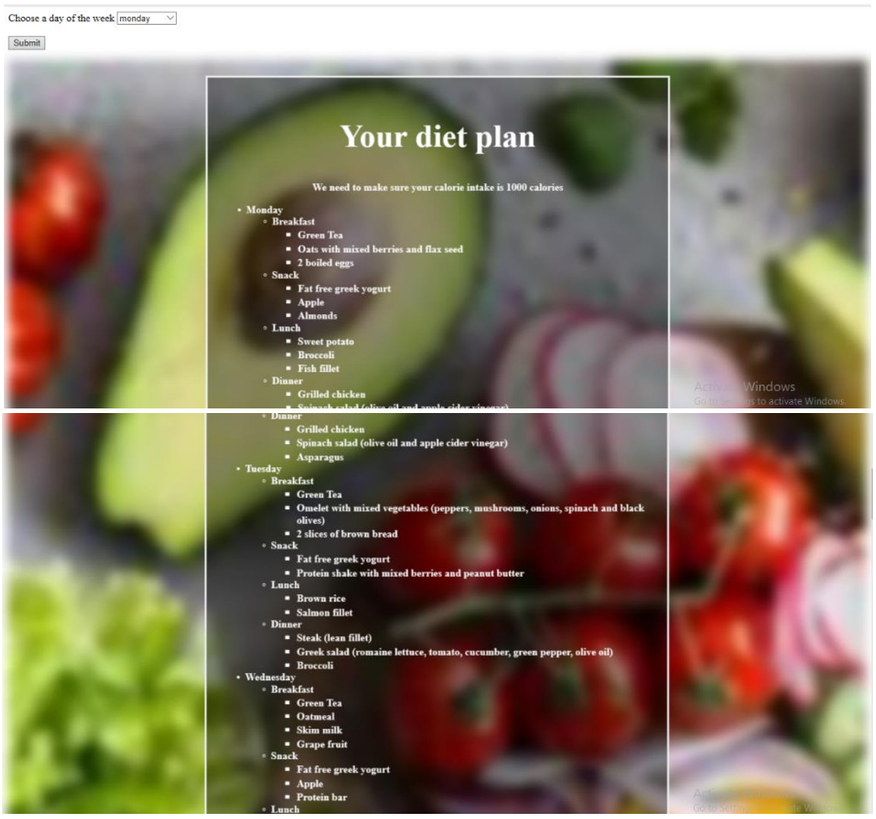
In the top left corner, we're giving user the option to select an exercise to view its information individually.



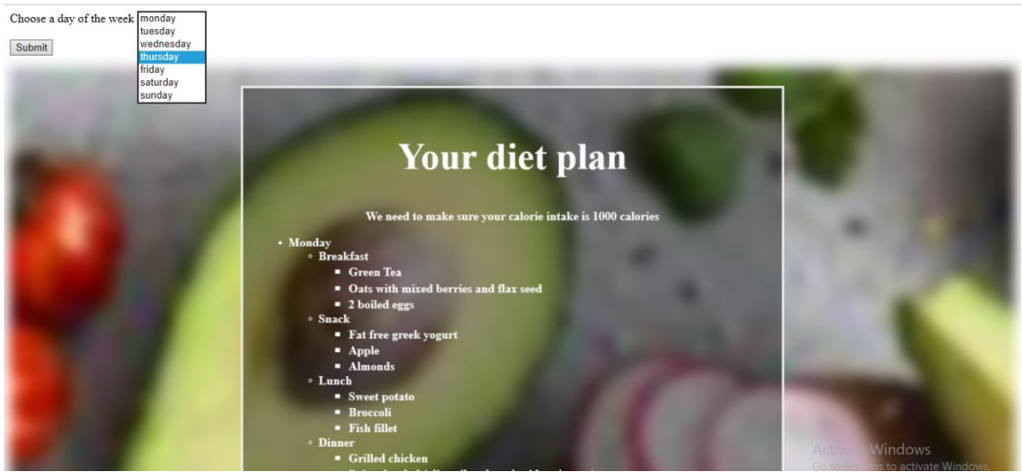
Once the user selects the exercise and presses submit the user is led to a page displaying information for that specific exercise.



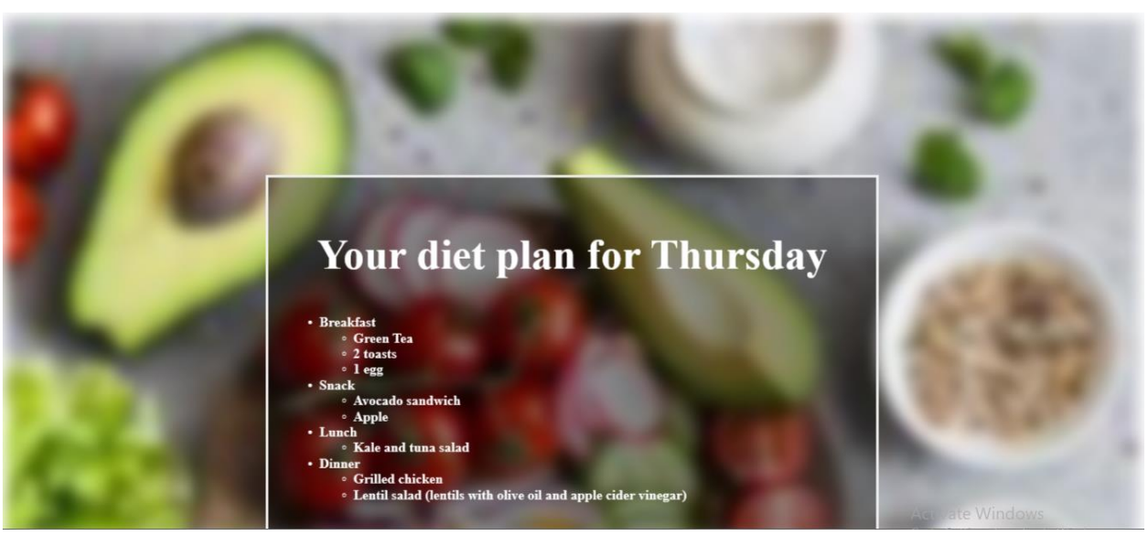
DIET_PLAN_REPORT:



This page displays the calorie intake for the user by fetching it from the data base. This page also displays diet plan for the whole week. In the top left corner, we're giving the user an option to select any day of the week to display diet plan for that day specifically.



Once the user selects a day and presses submit, user is led to a page displaying diet plan for that specific day.

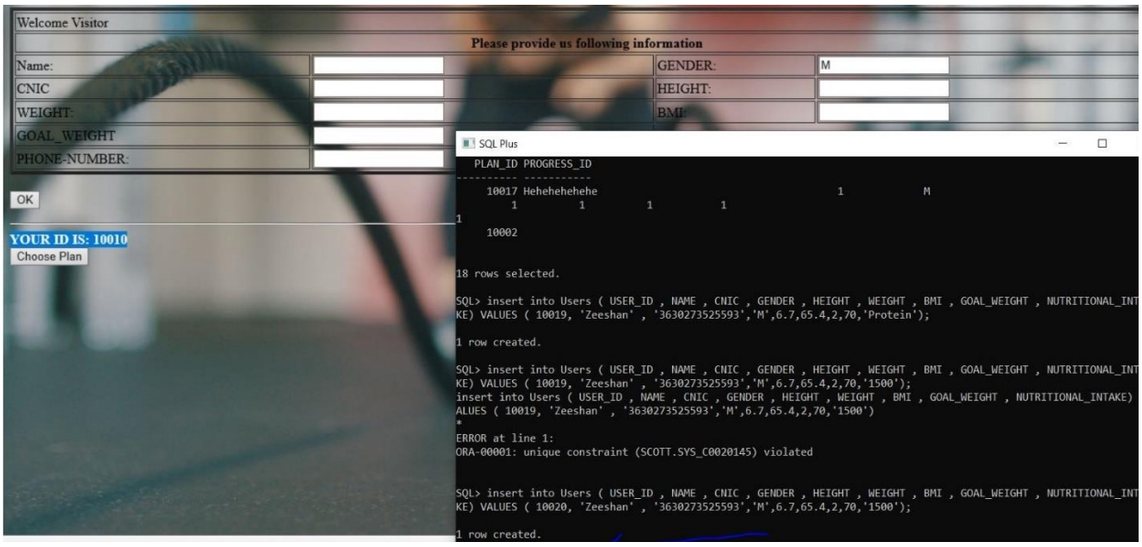


8: REPORT SQL

In this part we will discuss various SQL queries linked to forms and reports in depth.

A: REGISTRATION:

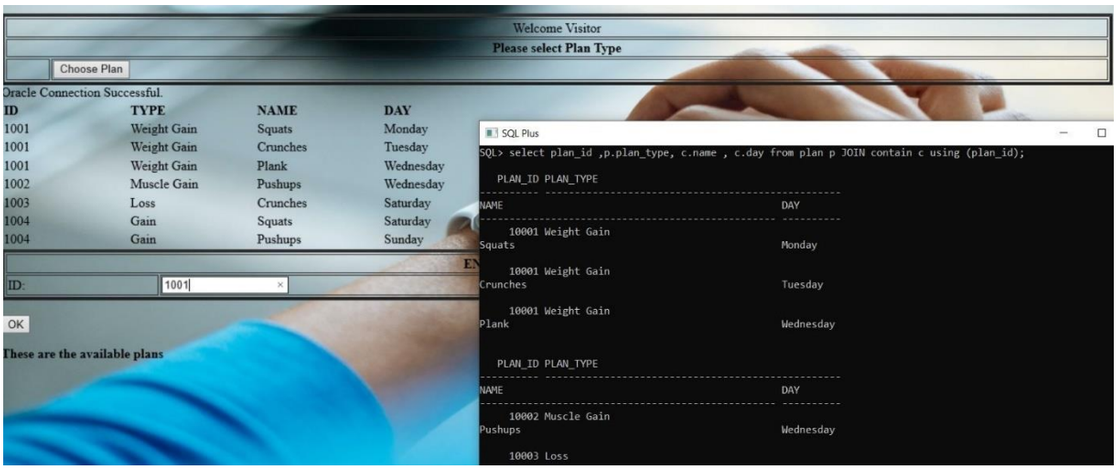
We will start off with the queries associated with Registration form.



This query inserts the user data into the User table. The user_id is a unique 5-digit number, name is a user name, cnic is user national identity number , Gender is to determine whether a user is female or male. On the basis of BMI, the user will be assigned an optimal nutritional intake.

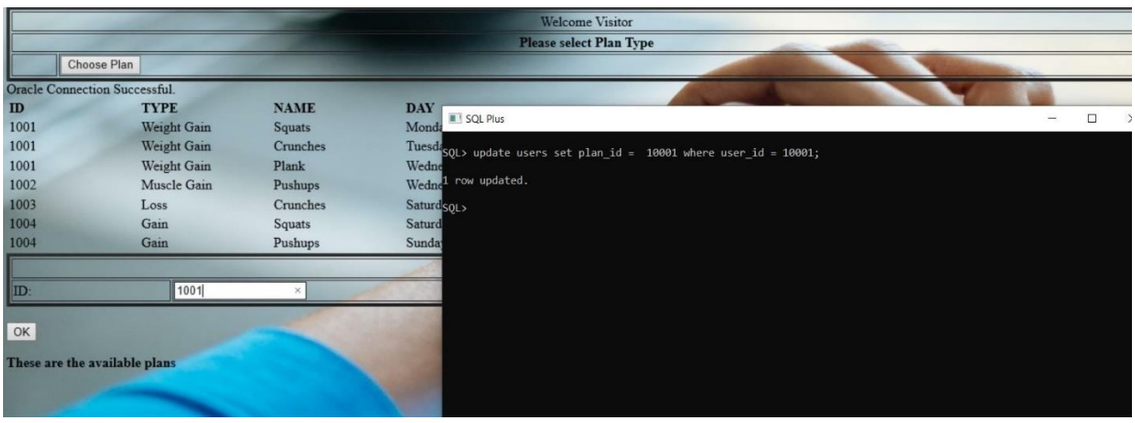
B: PLAN SELECTION:

Next the user needs to select a workout plan. So according to our assumption, user can select any plan from already stored plans.



User will be displayed all available plans. User will have to add a plan id and that plan will be chosen for that specific user.

After pressing the OK button, the user plan id will be updated as follows :

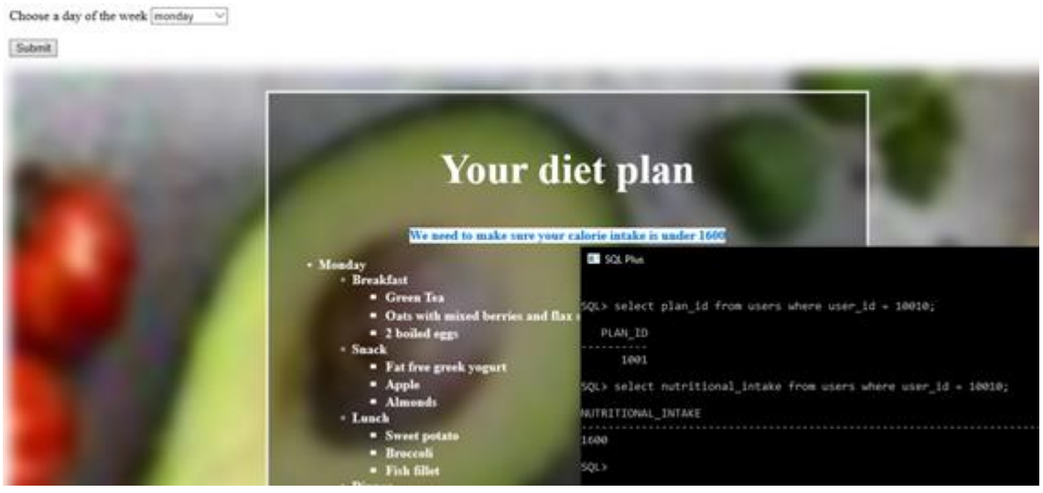


Plan_id is a foreign key in USER table , its value will be updated when a user enters the value from the displayed values.

C: VIEW DIET PLAN:

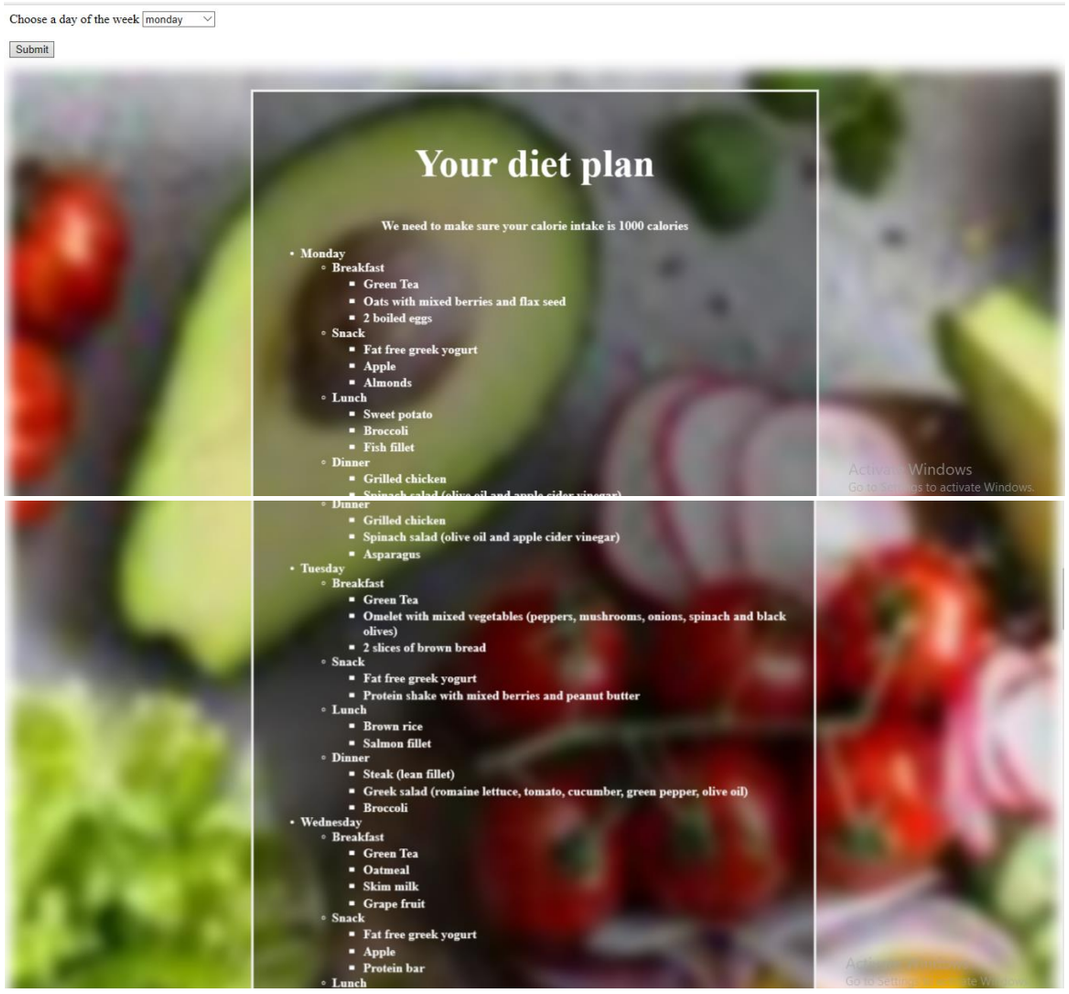
On pressing “View Diet Plan”, user is led to the login page.

Once the user inserts their id and presses “press me” their user id will be sent to Diet Plan Report page and they will be led to this page as well.

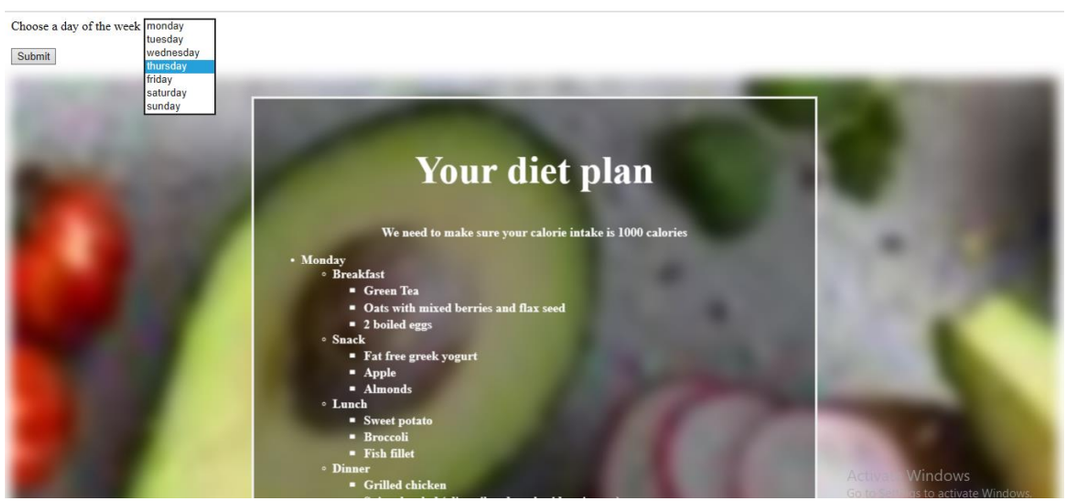


This report page displays meal plan for the whole week. Apart from this it also displays the required calorie intake for the user as well. An SQL query has also been displayed showing the data being fetched from the data base. The SQL query used is “select nutritional intake from users where user_id = 10010;”.

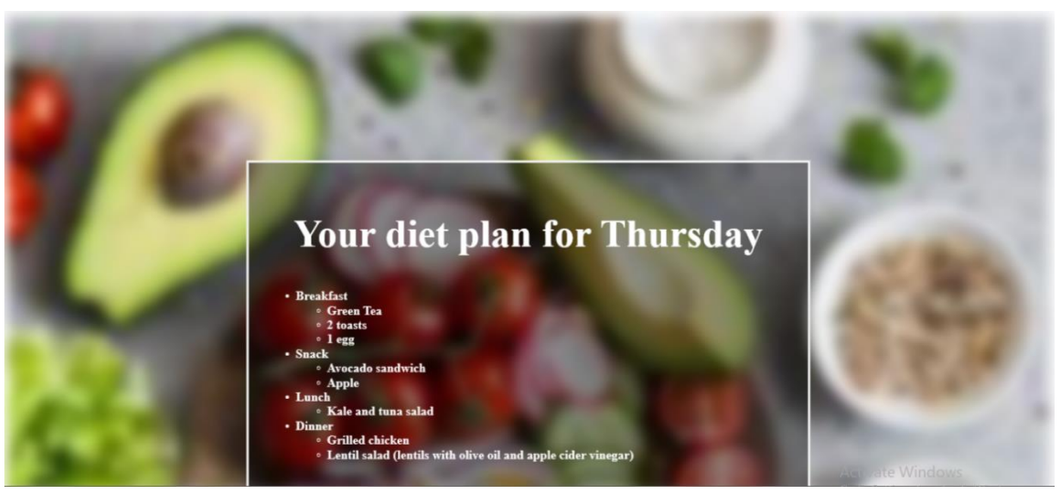
A user who previously got registered, and got assigned the id 10010; their nutritional intake would be displayed using this query.



On the top left corner, we're giving the user an option to select any day of the week to display diet plan for that day specifically.



Once the user selects a day and presses submit, user is led to a page displaying diet plan for that specific day.

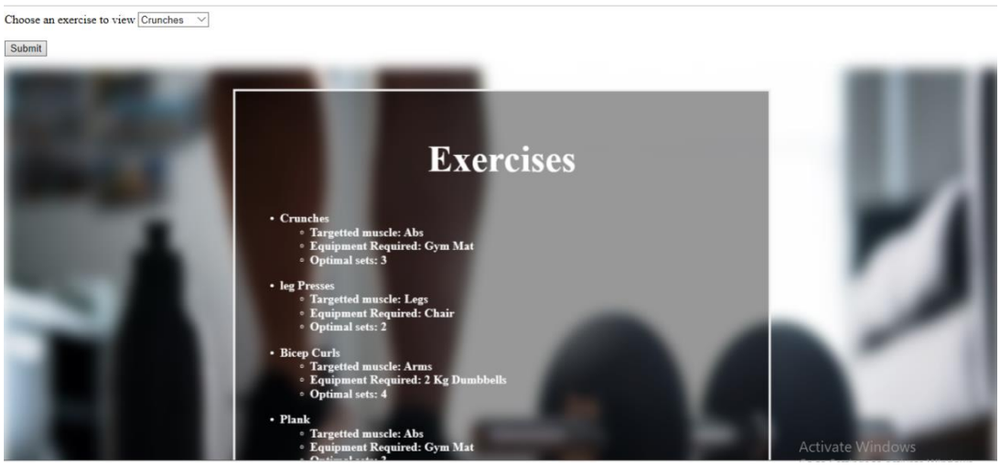


D: VIEW EXERCISES:

Now back to Home Screen 1.

Once the user press ‘View Exercises’ user is led to the Exercise Report page.

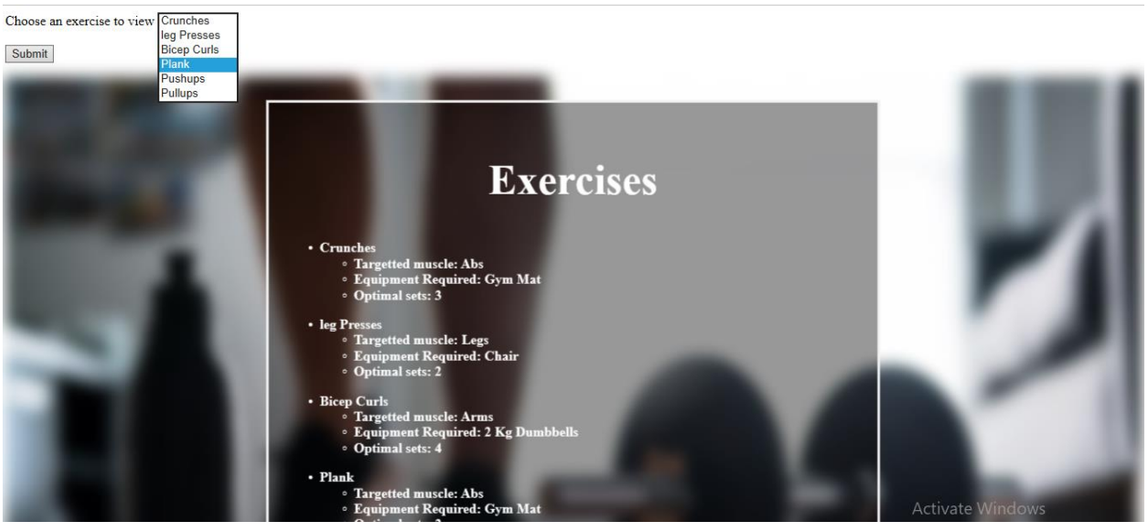
This report shows all the exercises that we have stored in our data base.



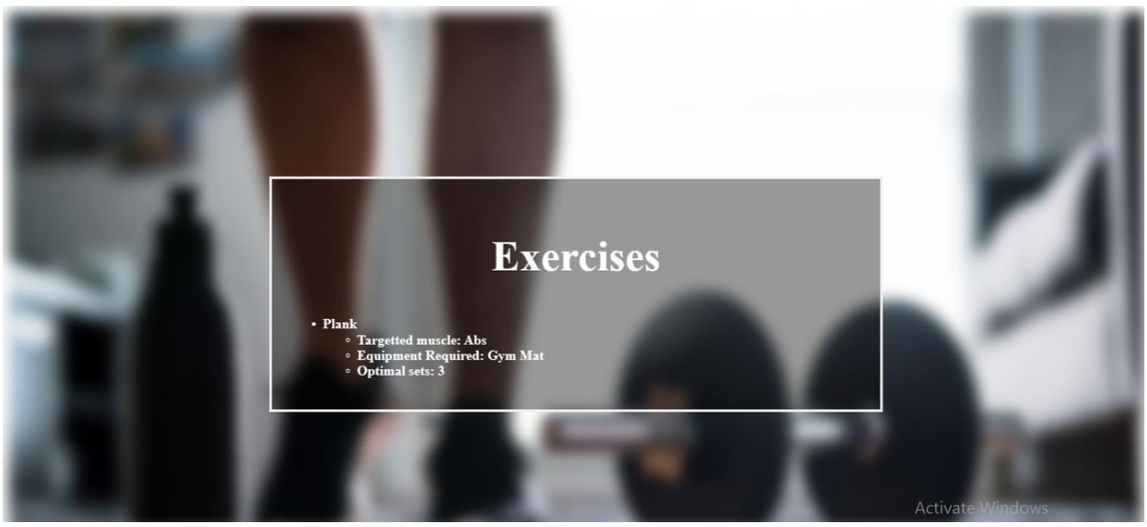
This page displays all the data regarding exercises stored in our data base. The information stored includes name of the exercises, targeted muscle, equipment required and optimal sets.

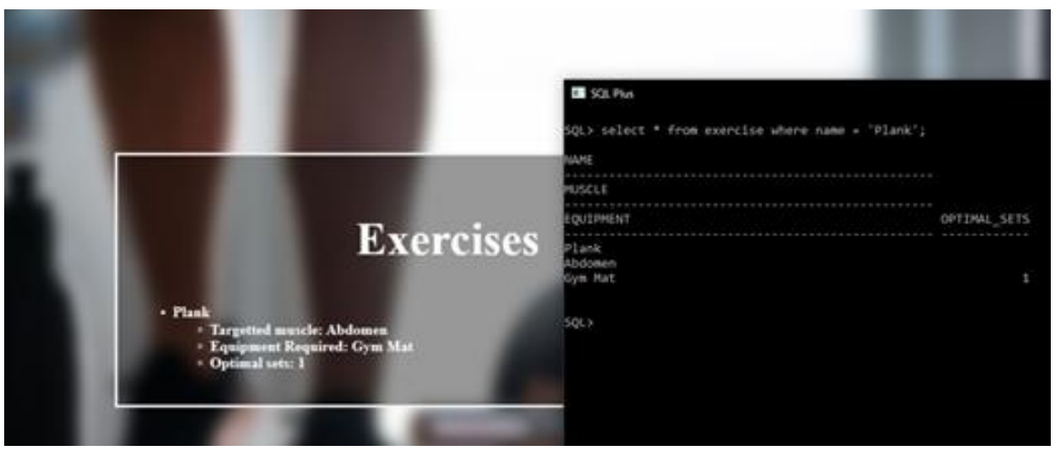
An **SQL query** has been displayed here; the SQL query is “select * from exercise;”. This query will display names of all the exercises along with their targeted muscle, equipment required and optimal sets.

In the top left corner, we’re giving user the option to select an exercise to view its information individually.



Once the user selects the exercise and presses submit the user is led to a page displaying information for that specific exercise.





An **SQL query** has been displayed. The SQL query used is “select * from exercise where name = ‘Plank’;”. This query displays information regarding the specific exercise named ‘Plank’.

E: VIEW PROGRESS:

Now back to Home Screen 1.

Once the user presses 'View Progress' they are led to the login page.

The login page asks for the user id. Once the user enters their user id and presses "Press Me" the user id will be sent to "View Progress" page and that page will be loaded.

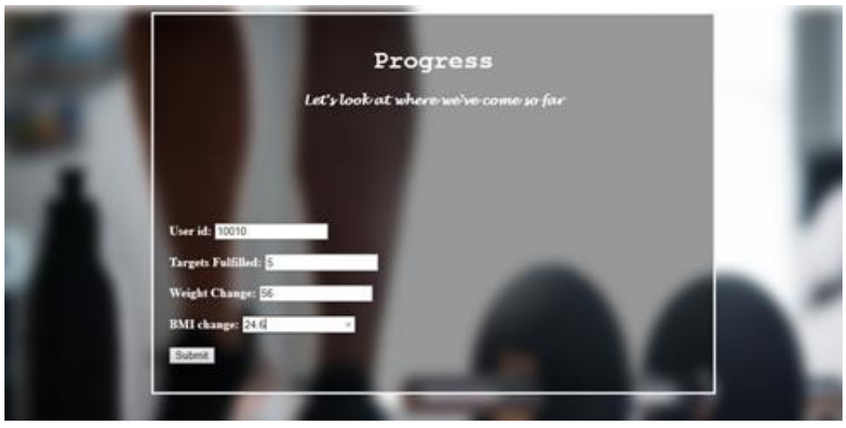
This page displays the user's progress. Initially when the user has just made an account their target fulfilled is 0 and their weight and BMI are the ones that they entered. An **SQL query** has also been displayed showing the progress data being fetched from the data base. The SQL query used is "select * from progress where progress_id = 10010".

F: EDIT PROGRESS:

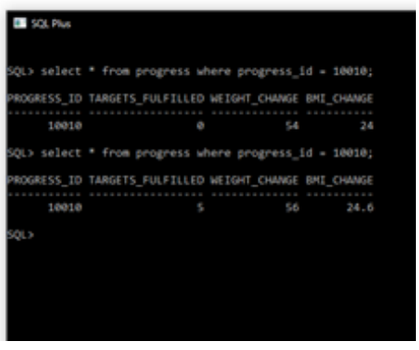
For editing the progress, we will go back to the Home Screen 1 and press Edit Progress.

Once the user presses edit progress, they will be led to login page.

Once the user enters their user id and presses “press me” their user id will be sent to “log_file” page and that page will be loaded.



User will update the record for their progress. They will enter their user id, targets fulfilled, weight change and BMI change. Once the user presses submit, record will be updated in the data base.



The updated message will be displayed and an **SQL query** has also been displayed showing the record being updated in the data base. The query used is “select * from progress where progress_id = 10010;”. It can be seen that first query displayed initial progress values. But when the user updates the values the second query displays different result (the values that user updated).

G: VIEW EXERCISE PLAN:

Now back to Home Screen 1. We will select View Exercise Plan.

Once the user presses view exercise plan, they will be led to login page.

Once the user enters their user id and presses “press me” their user id will be sent to “Exercise Plan” page and that page will be loaded.

This page displays the exercise plan for the user. The page shows the day on which the user will be doing the exercise, the name of the exercise to be done, muscles targeted by that exercise, equipment required for that exercise and optimal sets for that exercise.

An SQL query has also been displayed showing the plan id that the user selected. Names of the exercises under that plan_id. The SQL query used is “select plan_id from users where user_id = 10010;” this query will display plan_id for that specific user.

We will use this plan id to display names and days that we need to do those exercises on in that plan. The query used to display the names and days is as follows.

“select * from contain where plan_id = 1001;”

Once we have all the names and of the exercises, we will use them to find the information regarding these exercises. As can be seen in the images below.

As this specific plan contains 3 exercises the SQL queries used are

“select * from exercise where name = ‘Crunches’;”

“select * from exercise where name = ‘Plank’;”

“select * from exercise where name = ‘Squats’;”

They display all the information regarding those exercises.

9: ASSUMPTIONS

Following assumptions were made for this project:

- The progress will be updated monthly only as in real life too, you cannot notice any visible change just after a week of workout.
- As people will be working-out from home, we will assume that they won’t have any proper gym machinery due to lack of space or financials.
- The BMI will be calculated automatically and the result will be updated in the Progress table.
- Workout Plan is on weekly basis.
- User can select any plan from the built-in plans.

10: FURTHER IMPROVEMENTS

For further improvement, one thing that could be done is to also store tutorials against each exercise so that people working out from home can easily exercise.

11: DEMONSTRATION VIDEO

We made a video of our screen recording of our project’s execution. We tested all possible queries and scenarios on it. The link to the video can be found as follows:

https://drive.google.com/file/d/1EtlXeKX-tllMVlrDwHbtcxMcAELGmRBX/view?usp=sharing

11: REQUIREMENTS TRACKING:

Project Functionality	Deliverables	Marks	Completed?
SQL	ERD	10	✓ Implemented and added in report.
	Table Description	5	✓ Implemented and added in report.
	Schema Diagram	5	✓ Implemented and added in report.
	DML+DDL	15	✓ Implemented and added in report and files attached separately too.
	Report SQL	15	✓ Implemented and added in report. Screenshots of interfaces and queries along with their outputs attached and added in report along with attaching separately too.
HTML Interface (Front-end)	Index Page	5	✓ Implemented and added in report along with attaching it separately in submission.
	Forms	15	✓ Implemented and added in report along with attaching it separately in submission.
	Reports	15	✓ Implemented and added in report along with attaching it separately in submission.
Query Optimization	All SQL queries must be following query optimization standards	5	✓ Implemented and maximum query optimization was assured.
Problem Comprehension	Degree of problem comprehension, and mapping to solution	5	✓ We tried our best to comprehend the problem completely followed by designing an efficient and complete solution for it.
PL-SQL	Use of PL-SQL (triggers, procedures, etc.)	5	✗ This couldn't be implemented as we tried implementing it but it kept giving errors and due to lack of debugging time, we had to drop this part.
Bonus	Any extra effort of use (Bootstrap, form validation, etc.)	10	✓ We designed our majority interfaces on bootstrap and attached the screenshots in this report along with adding the files in submission.
Demonstration Video	The video containing demonstration of our project	-	✓ We made a video in which we ran our project and tested all possible scenarios on it. Link to the video is attached.