Sir Syed University of Engineering & Technology (SSUET)

Software Engineering Department

Course Name: Data Structures & Algorithm (SE-203L)

Semester: 3rd
Batch: 2023F

PROJECT REPORT

Project Title: Self-Fit

Project By:

Hassan Ahmed
Yousuf Ashher Siddiqui
Syed Hussain Ali
M Umer Intesab

Report Prepared by Hassan

TABLE OF CONTENTS

S. No.	TOPIC	Page No.
1	INTRODUCTION OF THE PROJECT	
2	ALGORITHM / PSEUDOCODE OF EACH OPERATION	
3	DATA STRUCTURES USED IN PROJECT	
4	SYSTEM FLOW DIAGRAM	
5	USER GUIDE	
6	CONCLUSION	
7	FUTURE EXPANSION OF THE PROJECT	

Self-Fit: A Personal Fitness Application

1. Introduction

Self-Fit is a simple and easy-to-use fitness app made in Java. It helps users stay healthy by offering custom workout plans, diet suggestions, BMI checks, and progress tracking.

Users can **sign up** by entering their name, phone number, email, and password, and also select their fitness level (beginner, intermediate, or advanced). There is a **login page** for returning users, along with options to **delete or recover** their account.

The app uses **Java data structures** and a **MySQL database** to save and manage user data smoothly.
The main goal is to make fitness simple, personalized, and helpful for everyone

2. Why We Developed Self-Fit?

As more people understand the importance of staying fit, gyms and fitness clubs are looking for better ways to support their members. Many fitness apps today lack useful features, work slowly on some devices, or don't give personalized advice to users.

Self-Fit is here to change that. It's a smart and easy-to-use app designed to help gyms and fitness clubs run smoothly. **Self-Fit** can track members' progress, analyze their fitness data, and give recommendations based on their goals. By focusing on speed, simplicity, and reliable features, **Self-Fit** makes it easier for gyms to provide a better experience and help their members stay motivated.

3. Objectives of Self-Fit

1. **Workout Planning (Default + Personalized)** – Users can follow default workout routines or get personalized plans based on their preferences, weight, and BMI.
2. **BMI Calculator** – Calculates Body Mass Index to recommend suitable workout and diet options.
3. **Custom Workout Addition** – Users can add or modify their workout routines according to their goals.
4. **Real-Time Progress Tracking** – Tracks steps taken, calories burned, and exercise performance.
5. **User-Friendly Interface** – Simple and easy for beginners as well as advanced users.
6. **Scalable Design** – Easily upgradable for future features like wearable device support.
7. **Accessibility Support** – Voice guidance and text-to-speech options for users with special needs.
8. **Gym Integration** – Allows gyms to manage members, trainers, class schedules, equipment usage, promotions, and send personalized updates.

2. ALGORITHM / PSEUDOCODE OF EACH OPERATION

1.LinkedList Operations

Used In Login Class and BMI History Class Operation:

Insertion

```
Algorithm InsertLinkedList(head, value):
    Input: Head of the LinkedList, value to insert
    Output: LinkedList with value inserted at the end
    new_node = Create new node with value
    If head == null:
        head = new_node
    Else:
        current = head
        While current.next != null:
            current = current.next
        current.next = new_node
End Algorithm
```

Traversing and Display

```
Algorithm TraverseLinkedList(head):
    Input: Head of the LinkedList
    Output: Display all elements in the LinkedList
    current = head
    While current != null:
        Print current.value
        current = current.next
End Algorithm
```

Searching

```
Algorithm SearchLinkedList(head, target):
    Input: Head of the LinkedList, target element
    Output: Node containing target or null if not found
    current = head
    While current != null:
        If current.value == target:
            Return current
        current = current.next
    Return null
End Algorithm
```

2. Queue Operations

Used in BMI Calculation Class and Registration Class

Traversing

```
Algorithm TraverseQueue(queue):
    Input: Queue
    Output: Display all elements in the queue
    For each element in queue:
        Print element
End Algorithm
```

Enqueue (Insertion)

```
Algorithm EnqueueQueue(queue, value):
    Input: Queue, value to enqueue
    Output: Queue with value added at the rear
    queue.enqueue(value)
End Algorithm
```

Dequeue (Deletion)

```
Algorithm DequeueQueue(queue):
    Input: Queue
    Output: Dequeued value from the front
    If queue is empty:
        Return "Queue is empty"
    Return queue.dequeue()
End Algorithm
```

Peek

```
Algorithm PeekQueue(queue):
    Input: Queue
    Output: Front element of the queue
    If queue is empty:
        Return "Queue is empty"
    Return queue.front()
End Algorithm
```

3. Stack Operations

Used in BMI Calculation Class , Delete Profile Class and Retrieve Profile Class

Traversing

```
Algorithm TraverseStack(stack):
    Input: Stack
    Output: Display all elements in the stack
    For each element in stack (from top to bottom):
    Print element
End Algorithm
```

PUSH (Insertion)

```
Algorithm PushStack(stack, value):
    Input: Stack, value to push
    Output: Stack with value added at the top
    stack.push(value)
End Algorithm
```

POP (Deletion)

```
Algorithm PopStack(stack):
    Input: Stack
    Output: Popped value from the top
    If stack is empty:
        Return "Stack is empty"
    Return stack.pop()
End Algorithm
```

Top

```
 Algorithm PeekStack(stack):
        Input: Stack
        Output: Top element of the stack
         If stack is empty:
             Return "Stack is empty"
         Return stack.top()
 End Algorithm
```

4.Array Operations

Used in BMI Calculation Class and BMI History Class

Traversing

```
Algorithm TraverseArray(array):
    Input: Array
    Output: Display all elements in the array
    For each element in array:
        Print element
End Algorithm
```

Insertion

```
Algorithm InsertArray(array, value, position):
    Input: Array, value to insert, position to insert at
    Output: Array with value inserted at the given position
    If position < 0 OR position > length of array:
        Return "Invalid position"
    Shift elements from position to right
    Insert value at position
End Algorithm
```

Sorting (Ascending)

```
Algorithm BubbleSortArray(array):
    Input: Array
    Output: Sorted array in ascending order
    For i from 0 to length of array - 1:
        For j from 0 to length of array - i - 1:
            If array[j] > array[j+1]:
                Swap array[j] and array[j+1]
End Algorithm
```

Conclusion:

Linked List *(Used in Login Class and BMI History Class)*

- **Operations:**
  - **Insertion**: Inserts a value at the end of the linked list.
  - **Traversing and Display**: Iterates through the linked list and prints elements.
  - **Searching**: Searches for a specific value in the linked list.
- **Usage:** Ideal for dynamic data storage where insertions and deletions occur frequently.

Queue *(Used in BMI Calculation Class and Registration Class)*

- **Operations:**
  - **Enqueue (Insertion)**: Adds an element at the rear of the queue.
  - **Dequeue (Deletion)**: Removes an element from the front of the queue.
  - **Peek**: Retrieves the front element without removing it.
  - **Traversing**: Displays all elements in the queue.
- **Usage:** Follows **FIFO (First-In-First-Out)**, making it useful for handling requests or tasks in sequence.

Stack *(Used in BMI Calculation Class, Delete Profile Class, and Retrieve Profile Class)*

- **Operations:**
  - **Push (Insertion)**: Adds an element at the top of the stack.
  - **Pop (Deletion)**: Removes an element from the top of the stack.
  - **Peek (Top Element)**: Retrieves the top element without removing it.
  - **Traversing**: Displays elements from top to bottom.
- **Usage:** Follows **LIFO (Last-In-First-Out)**, making it useful for **undo/redo operations, function calls, and backtracking algorithms**.

Array *(Used in BMI Calculation Class and BMI History Class)*

- **Operations:**
  - **Insertion**: Inserts an element at a specific position.
  - **Sorting (Bubble Sort - Ascending Order)**: Sorts elements in ascending order.
  - **Traversing**: Displays all elements in the array.
- **Usage:** Provides **fast random access** but has a fixed size, making it ideal for storing and processing data efficiently.

5. DATA STRUCTURES USED IN PROJECT

Mention all the data structures used in the project.

1. Stack

- **Where Used**: `Delete_JF` and `Retrieval_JF` classes also in BMI_CountJF class .
- **Purpose**:
    - Temporarily stores deleted user profiles to allow for retrieval (undo functionality).
    - Utilizes **Last-In-First-Out (LIFO)** behavior.
    - A stack is used to store the existing BMI values and dates in **Last-In-First-Out (LIFO)** behavior.
    - It allows easy management of the most recent values and enables undo functionality by popping the latest values.

2. Arrays

- **Where Used**: BMI_Count JF Class
- **Purpose**:
    - Make out the String Value in database and make it an array of 7 BMI Values as they are separated by comma
    - Enable iteration for Further Operations

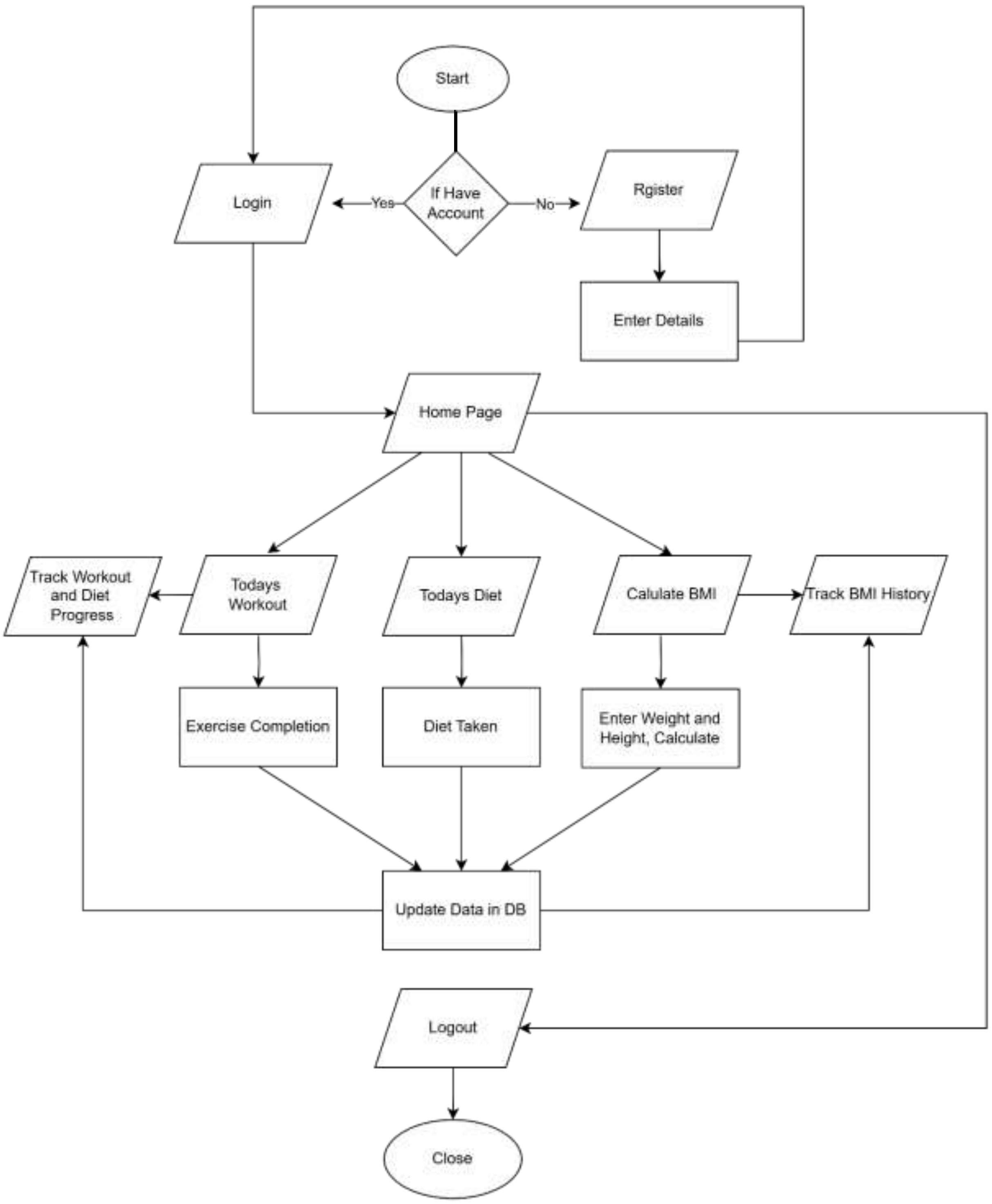
3. Queue

- **Where Used**: BMI_CountJF class
- **Purpose**:
    - A queue is used to store the BMI values and dates, implementing **First-In-First-Out (FIFO)** behavior.
    - It ensures that the oldest BMI value is removed when the queue exceeds the size limit of 7.

4. Linked List

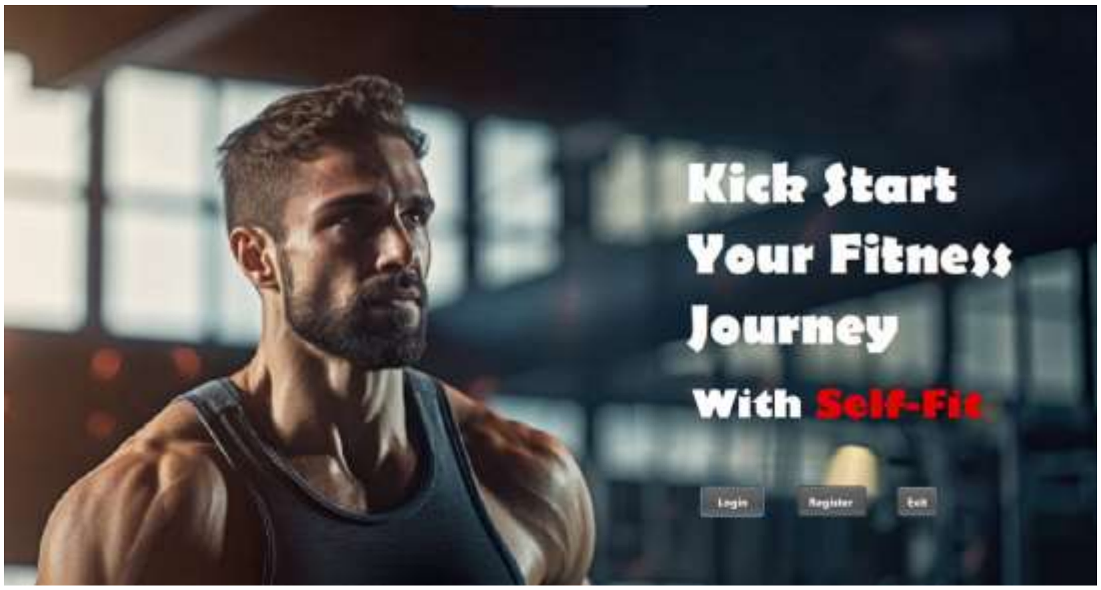
- **Where Used**: BMI_H and Login_JF classes
- **Purpose**:
    - To Collect Values from the Queue
    - To Put up the Output Labels by being travesed
    - After Storing all Names and passwords in Db
    - Searching for the user in those Names and Password Respectively

7. SYSTEM FLOW DIAGRAM

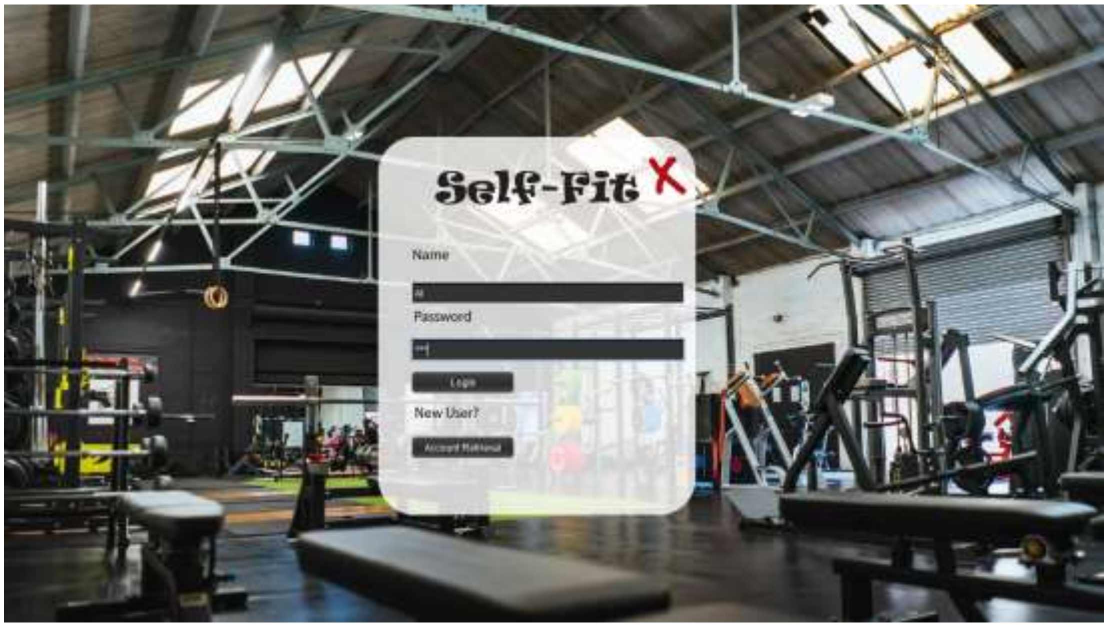


8. USER GUIDE

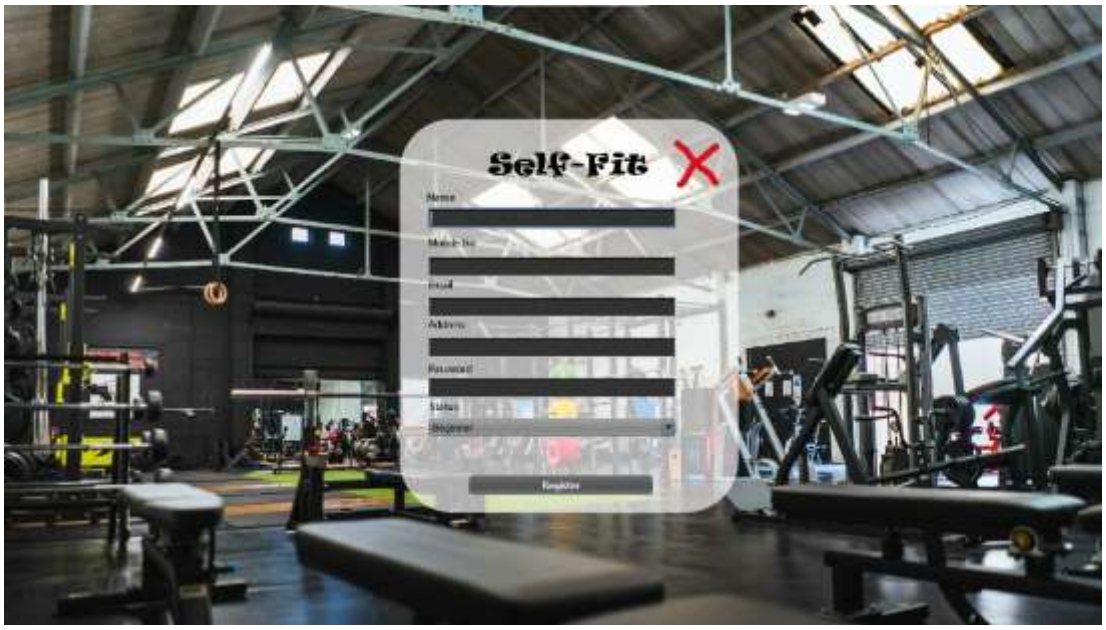
1. Welcome and Entry Page: It is the Entry Window of Self-Fit that allow you to register as new or Login with your credentials either you are admin or user.



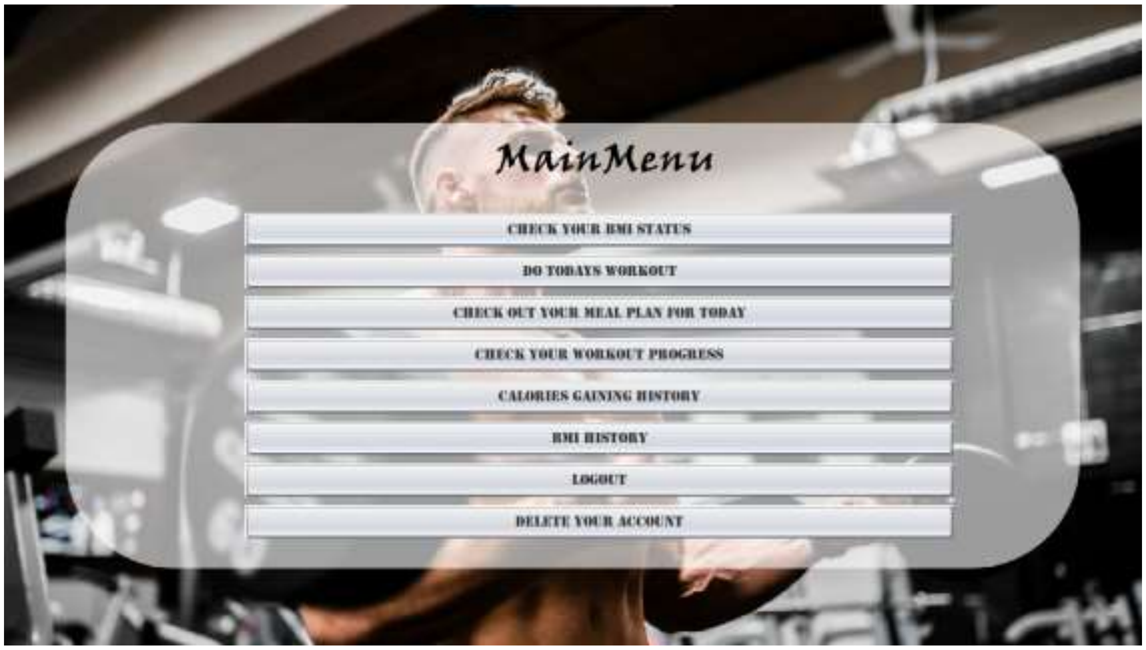
2. Login Page: This page represents the first thing about our website. It leads on to the login point for its personnel; it takes up the username and password , also provide an option to retrieve your previous profile



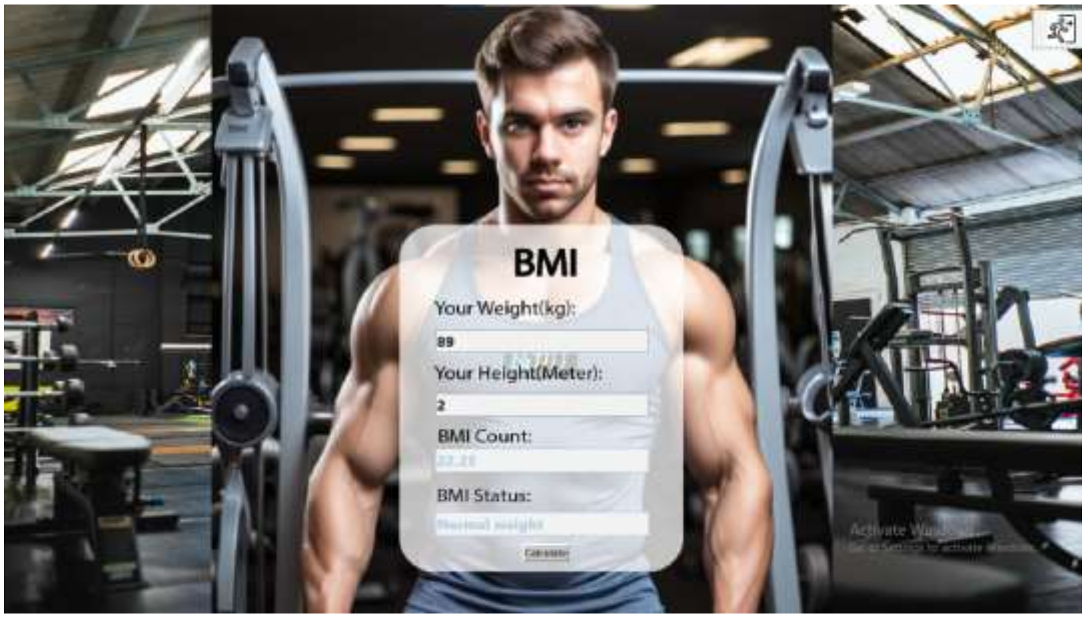
3. **Register Option:** This page represents registering new profile for Self-Fit . It requires username, password, email, address and Status, theses information are all mandatory.



4. **User Home Page:** This page shows us what user can see and access. He can see exercise and meal plans , track his BMI Count and keep track of these



5. BMI Calculation: This option will help you check your BMI Count



BMI

Your Weight(kg):

89

Your Height(Meter):

2

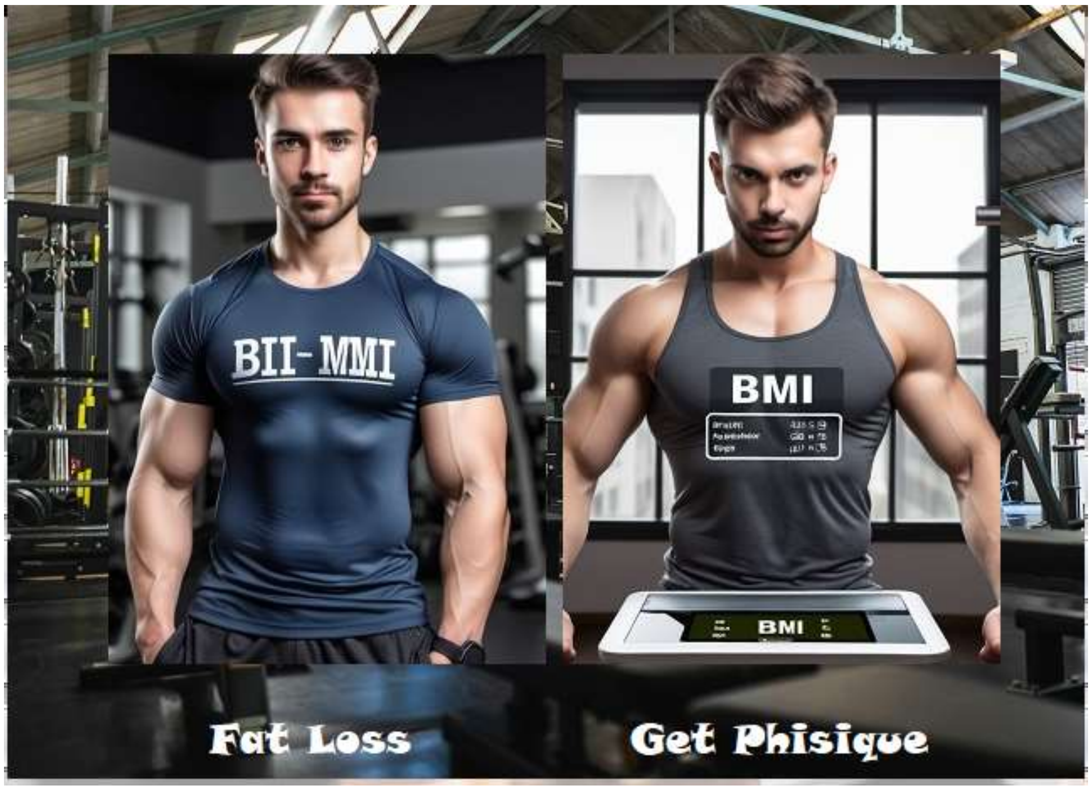
BMI Count:

22.25

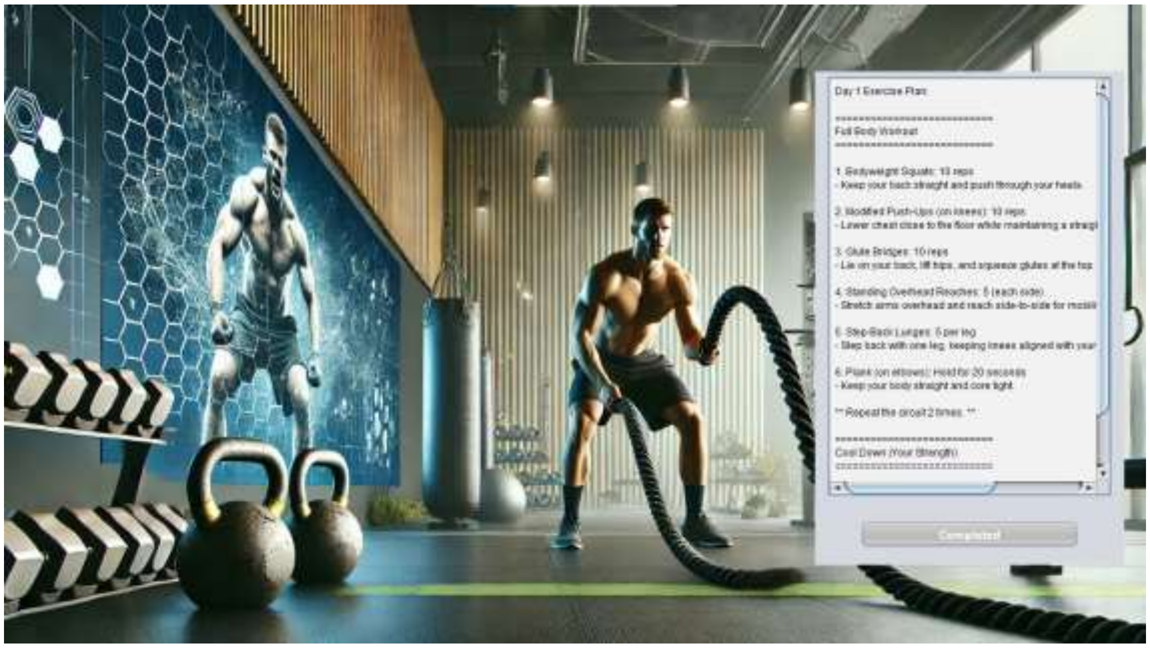
BMI Status:

Normal weight

6. **Select Workout Plan:** When you Register your self for the first you will be asked what kind of workplan you want , then according to your selection a plan for thirty days will be provided to you

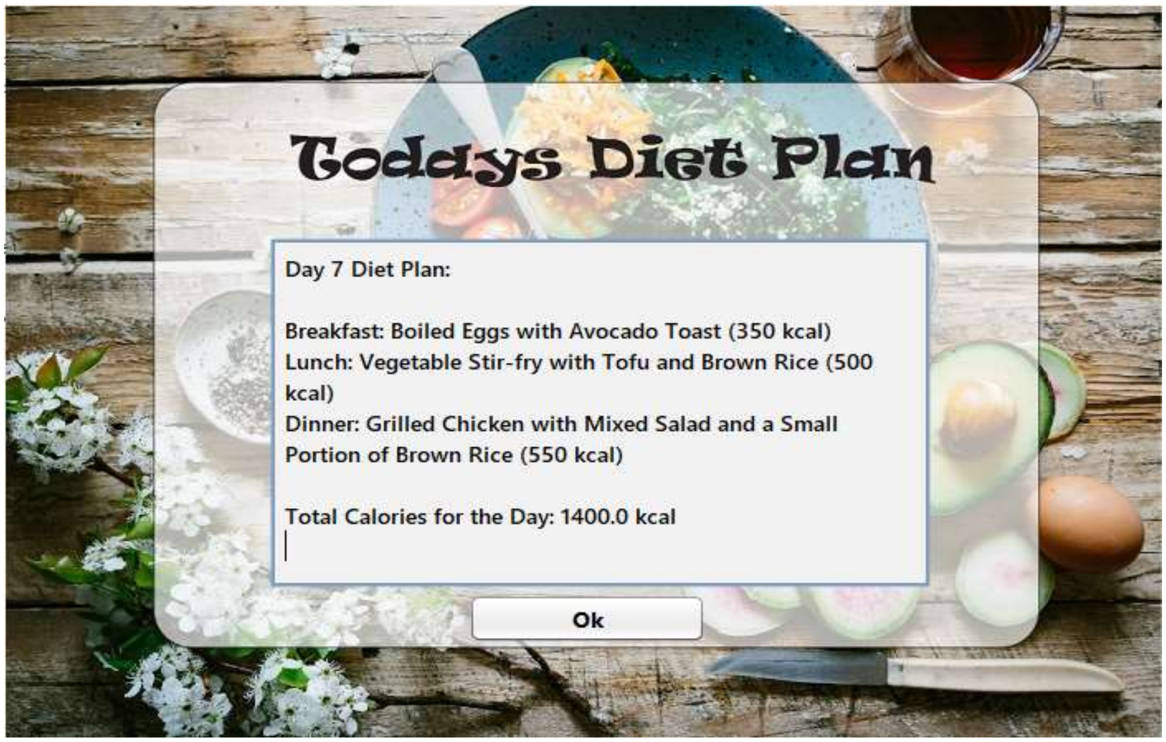


7. **Workout Plan**: This particular window allows you to see what you have to do according to your Exercise plan



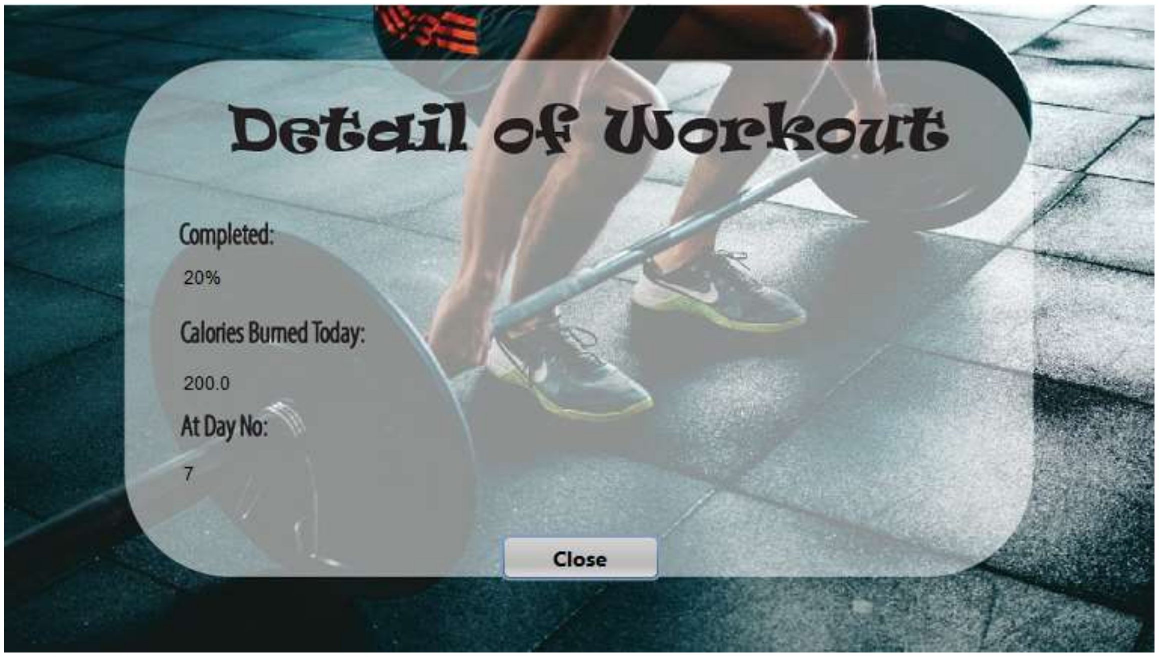
```
Day 1 Exercise Plan:

============================
Full Body Workout
============================

1. Bodyweight Squats: 10 reps
- Keep your back straight and push through your heels.

2. Modified Push-Ups (on knees): 10 reps
- Lower chest close to the floor while maintaining a straig

3. Glute Bridges: 10 reps
- Lie on your back, lift hips, and squeeze glutes at the top

4. Standing Overhead Reaches: 5 (each side)
- Stretch arms overhead and reach side-to-side for mobili

5. Step-Back Lunges: 5 per leg
- Step back with one leg, keeping knees aligned with your

6. Plank (on elbows): Hold for 20 seconds
- Keep your body straight and core tight.

** Repeat the circuit 2 times. **

============================
Cool Down (Your Strength)
============================
```

8. **Diet Plan :** Similar to workout plan this is used to follow a certain diet plan



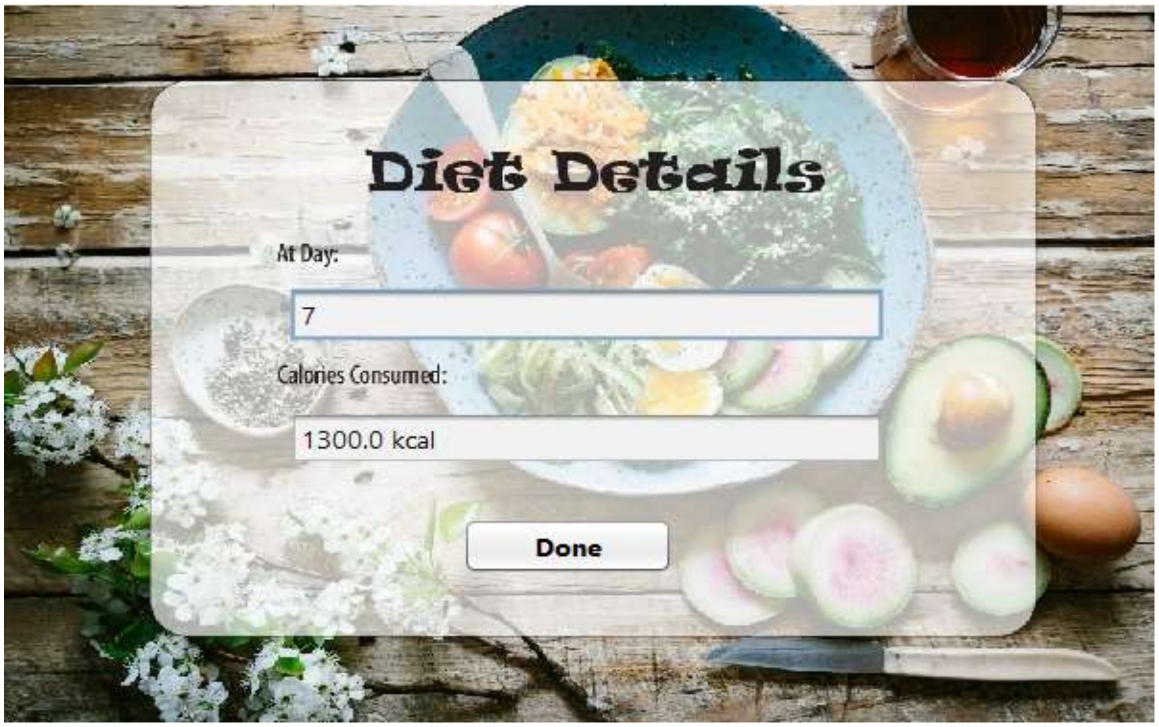
```
Day 7 Diet Plan:

Breakfast: Boiled Eggs with Avocado Toast (350 kcal)
Lunch: Vegetable Stir-fry with Tofu and Brown Rice (500
kcal)
Dinner: Grilled Chicken with Mixed Salad and a Small
Portion of Brown Rice (550 kcal)

Total Calories for the Day: 1400.0 kcal
```

9. Tracking Workout Progress : This helps you to see the progress of your workout plan



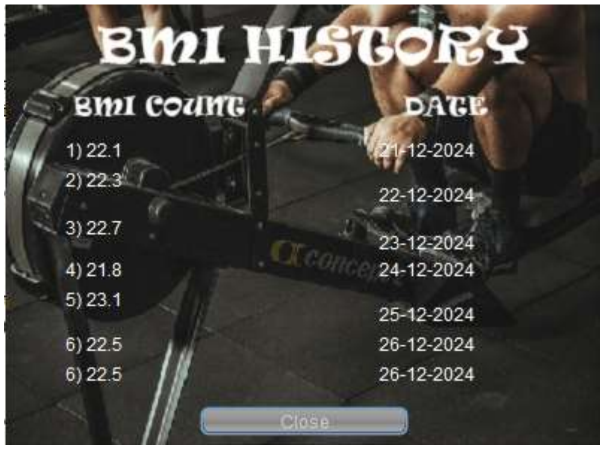
Completed:
20%
Calories Burned Today:
200.0
At Day No:
7

10. Tracking Diet Progress: Similar to Workout Progress Tracking



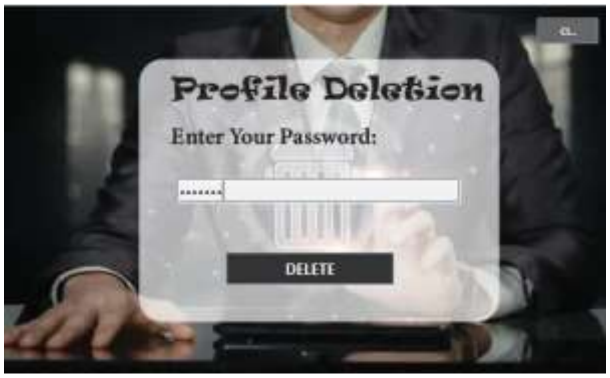
At Day:
7
Calories Consumed:
1300.0 kcal

11. **BMI History Tracker:** This Window particularly allows you to have a look on last 7 entries of your BMI Count


BMI HISTORY

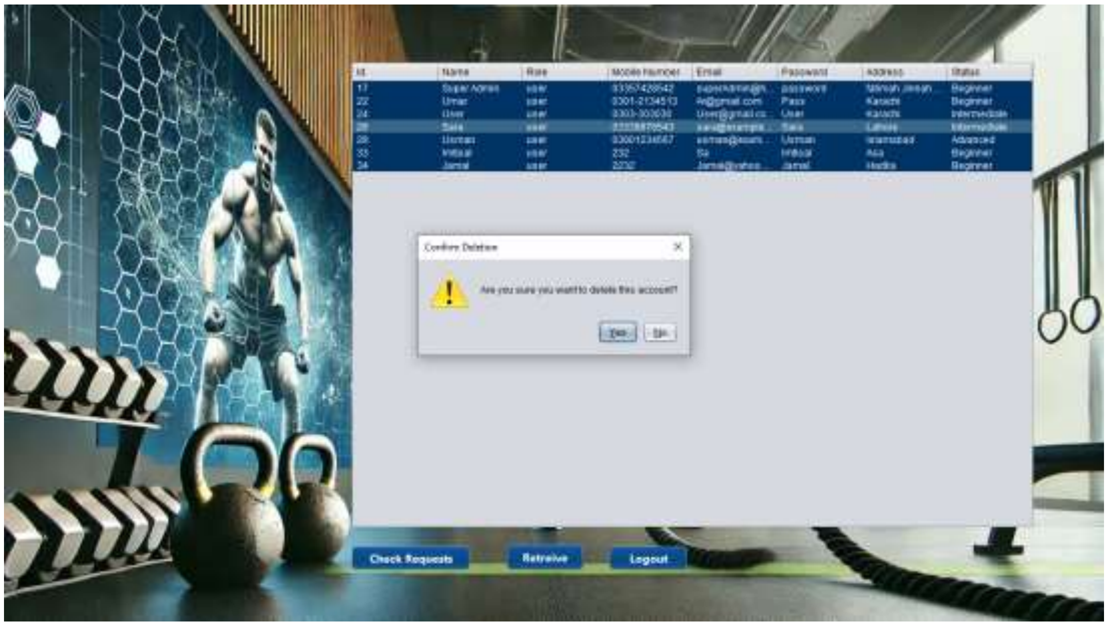
BMI COUNT	DATE
1) 22.1	21-12-2024
2) 22.3	22-12-2024
3) 22.7	23-12-2024
4) 21.8	24-12-2024
5) 23.1	25-12-2024
6) 22.5	26-12-2024
6) 22.5	26-12-2024

12. **Profile Deletion:** This Window particularly allow user to delete account of Self-Fit



11.Account Retrieval : A request for Account Retrieval can be done by User from login page, if he/her deleted account if Admin considered it the account can be recovered

12 .Admin Panel : An Admin Panel is there for the Management of the Fitness Club or Gym , who will have authority to delete or retrieve account and have the data of user with them



Id	Name	Role	Mobile Number	Email	Password	Address	Status
17	Super Admin	user	03357428542	superAdmin@h...	password	Mansah Jinnah ...	Beginner
22	Umar	user	0301-2134513	Ar@gmail.com	Pass	Karachi	Beginner
24	User	user	0303-303030	User@gmail.co...	User	Karachi	Intermediate
28	Sara	user	03338879543	sara@example...	Sara	Lahore	Intermediate
29	Usman	user	03001234567	usman@exam...	Usman	Islamabad	Advanced
33	Intisal	user	232	isa	Intisal	Asia	Beginner
34	Jamal	user	2232	Jamal@yahoo...	Jamal	Madina	Beginner

Confirm Deletion

Are you sure you want to delete this account?

9. CONCLUSION

The Self-Fit application successfully integrates various data structures, such as Stack, Arrays, Queue, and Linked List, to provide an efficient, fast, and user-friendly fitness experience. By implementing these structures, we were able to optimize performance, manage user data securely, and ensure smooth progress tracking for personalized fitness plans. The app offers features like BMI calculations, real-time updates, and data security, making it a comprehensive solution for users at different fitness levels. Overall, Self-Fit serves as a reliable fitness companion, aiming to make health management both accessible and effective for users.

10. FUTURE EXPANSION OF YOUR PROJECT

In the future, Self-Fit can be enhanced by adding support for wearable fitness devices, helping users track their health and activities in real time. We also plan to introduce personalized workout videos and social media integration, so users can stay motivated and share their progress. More detailed health analytics will give better insights into fitness and well-being. The app's AI can also be improved to give smarter, personalized recommendations based on user habits, fitness level, and progress. Lastly, we aim to make the app available on iOS and connect it with health services to reach a wider audience and improve accessibility.