GYMWORKOUTPLANNERANDT RACKER

[Document subtitle]

Contents

Introduction:

The Gym Workout Planner and Tracker is aimed at developing a Desktop Java Application that will monitor the workouts of participants under the guidance of specialized coaches. The primary goal of this application is to help participants achieve their fitness goals by providing tailored workout programs and nutrition plans. The application will also act as a platform for coaches to manage and track the progress of their participants.

Program structure and relationships:

The program consists of several classes that represent different entities and functionalities. Here is the program structure and relationships:

1. User class:

- Represents a user in the system, with common attributes like name, username, password, and profile.

- Has subclasses Coach and Participant, which inherit from User.

2. Profile class:

- Represents the profile of a user and contains attributes like health information, goals, and progress.

- User class has a profile object.

3. Coach class:

- Represents a coach in the system.

- Inherits from User class and has additional attributes like the number of years of experience and a list of participants.

- Has methods to view participants' profiles and add workout and nutrition programs.

4. Participant class:

- Represents a participant in the system.

- Inherits from User class and has additional attributes like a coach, cardio program, strength program, and specialized nutrition program.

- Has methods to select a coach, view workout and nutrition programs, and track progress.

5. WorkoutProgram interface:

- Represents a workout program, with methods for adding and removing exercises and viewing the program.

6. NutritionProgram interface:

- Represents a nutrition program, with methods for adding and removing meals and viewing the program.

7. BasicNutritionProgram class:

- Implements the NutritionProgram interface and represents a basic nutrition program.

- Has a list of meals.

8. SpecializedNutritionProgram class:

- Implements the NutritionProgram interface and represents a specialized nutrition program.

- Has a list of meals.

9. CardioProgram class:

- Implements the WorkoutProgram interface and represents a cardio workout program.

- Has a list of cardio exercises and additional attributes like program duration and intensity level.

10. StrengthProgram class:

- Implements the WorkoutProgram interface and represents a strength workout program.

- Has a list of strength exercises and additional attributes like weight and monthly repetitions.

11. Exercise class:

- Represents an exercise, with attributes like name, muscle group, repetitions, and duration.

12. Meal class:

- Represents a meal, with attributes like name, ingredients, and nutritional information.

13. Observer interface:

- Represents an observer who receives and handles notifications.

- Has update method to handle notifications.

14. WorkoutObserver class:

- Implements the Observer interface and represents an observer for workout program notifications.

15. NutritionObserver class:

- Implements the Observer interface and represents an observer for nutrition program notifications.

16. WorkoutProgramStore class:

- Stores the available workout program types.

- Has a static map that maps program types to their corresponding instances.

- Provides a method to create a new instance of a desired workout program type.

The relationships between these classes can be summarized as follows:

- Coach and Participant inherit from User.

- Coach has a list of Participant objects.

- Participant has a Coach object and can have CardioProgram, StrengthProgram, and SpecializedNutritionProgram objects.

- User has a Profile object.

- NutritionProgram and WorkoutProgram are interfaces implemented by BasicNutritionProgram, SpecializedNutritionProgram, CardioProgram, and StrengthProgram.

- Observer is an interface implemented by WorkoutObserver and NutritionObserver.

- WorkoutObserver and NutritionObserver are responsible for receiving and handling program notifications.

Functionalities:

The Gym Workout Planner and Tracker offers a range of functionalities for both coaches and participants. Let's take a closer look at each user's capabilities:

1. Coach:
    - View Participants' Profile: Coaches will have access to the profiles of their participants, including their health information. This will provide coaches with the necessary background information to customize workout programs and nutrition plans.
    - Add Workout Programs: Coaches can create and assign specific workout programs to each participant. These programs can be adjusted on a monthly basis to account for individual progress and goals.
    - Add Nutrition Programs: Coaches will design and prescribe nutrition plans for their participants. The nutrition programs will be tailored to complement the workout programs and maximize the effectiveness of the participants' fitness journey.
    - View Participant Progress: Coaches will be able to track the progress of their participants, receiving updates on the completion of workouts and adherence to nutrition plans. This information will allow coaches to assess the effectiveness of their programs and make any necessary adjustments.
2. Participant:
    - Add Health Information: Participants will provide their health information during the sign-up process. This information will help coaches personalize the workout and nutrition programs to cater to the participant's needs and capabilities.
    - Update Health Information: Participants can update their health information whenever necessary, ensuring that coaches have up-to-date and accurate details to create effective programs.
    - Select Coach: Participants will have the ability to choose their preferred coach from a list of available coaches. This allows participants to connect with a coach who aligns with their fitness goals and preferences.
    - View Workout Program: Participants will have access to their personalized workout program created by the coach. The program will outline the specific exercises and routines to be followed on a daily basis.
    - View Nutrition Program: The application will present the nutrition programs designed by the participant's chosen coach. These plans will encompass dietary guidance that supports the workout program and overall fitness objectives.

- View and Confirm Progress: Participants will have a dashboard where they can view their progress and confirm the completion of workouts and adherence to nutrition plans. This will provide a sense of accomplishment and motivation to continue their fitness journey.

Difficulties:

1. Complexity in tracking multiple workout programs: One of the key challenges is managing multiple workout programs for each participant. With different workout programs running simultaneously, ensuring accurate tracking of progress can be complex. To overcome this, the application needs to have a systematic approach to organize and manage these programs, providing clear indicators of completion and progress.
2. Difficulty in managing multiple nutrition programs: Similar to workout programs, managing multiple nutrition programs for each participant can be challenging. The application needs to ensure that the correct nutrition program is assigned to each participant and aligns with their corresponding workout program. Coordination between coaches and participants is crucial to overcome this difficulty.
3. Daily achievement confirmation: Obtaining reliable and accurate confirmation of daily achievements from the participants can be a challenge. Self-reporting by participants may be subjective and prone to errors. Implementing mechanisms such as checkboxes or digital signatures can help minimize inaccuracies and provide a standardized system for participants to confirm their daily progress.
4. Viewing progress and fitness history: Maintaining a comprehensive record of participant progress and fitness history requires careful management. The application needs to store and display this information in an organized and easily accessible manner. Coaches and participants should have the ability to track their progress over time, view previous achievements, and identify areas for improvement.

Design patterns used:

Prototype design pattern is used in the classes WorkoutProgram, CardioProgram, StrengthProgram, and WorkoutProgramStore.

The prototype design pattern allows the creation of new objects by cloning existing ones. It provides a way to create objects without specifying the exact class of the object that will be created.

In the code, the WorkoutProgram, CardioProgram, and StrengthProgram interfaces define the necessary methods for creating workout programs. The classes CardioProgram and StrengthProgram implement the WorkoutProgram interface and override the necessary methods.

They also implement the **clone()** method, which allows creating new instances of the respective program using a prototype object.

The **WorkoutProgramStore** class acts as a **factory** for creating new instances of workout programs. It uses a map to store instances of different workout program prototypes. The **make(String type) method** in the WorkoutProgramStore class retrieves the respective prototype object from the map and clones it to create a new instance of the workout program.

The **Observer design pattern** is used in the classes Observer, WorkoutObserver, and NutritionObserver.

The Observer design pattern establishes a one-to-many relationship between objects, so that when one object changes its state, all its dependents are notified and updated automatically.

In the given code, the Observer interface defines the update(String message) method that is implemented by the WorkoutObserver and NutritionObserver classes. These classes act as observers and implement the logic to receive and display notifications about changes in workout programs and nutrition programs, respectively.

The other design patterns used in the code are:

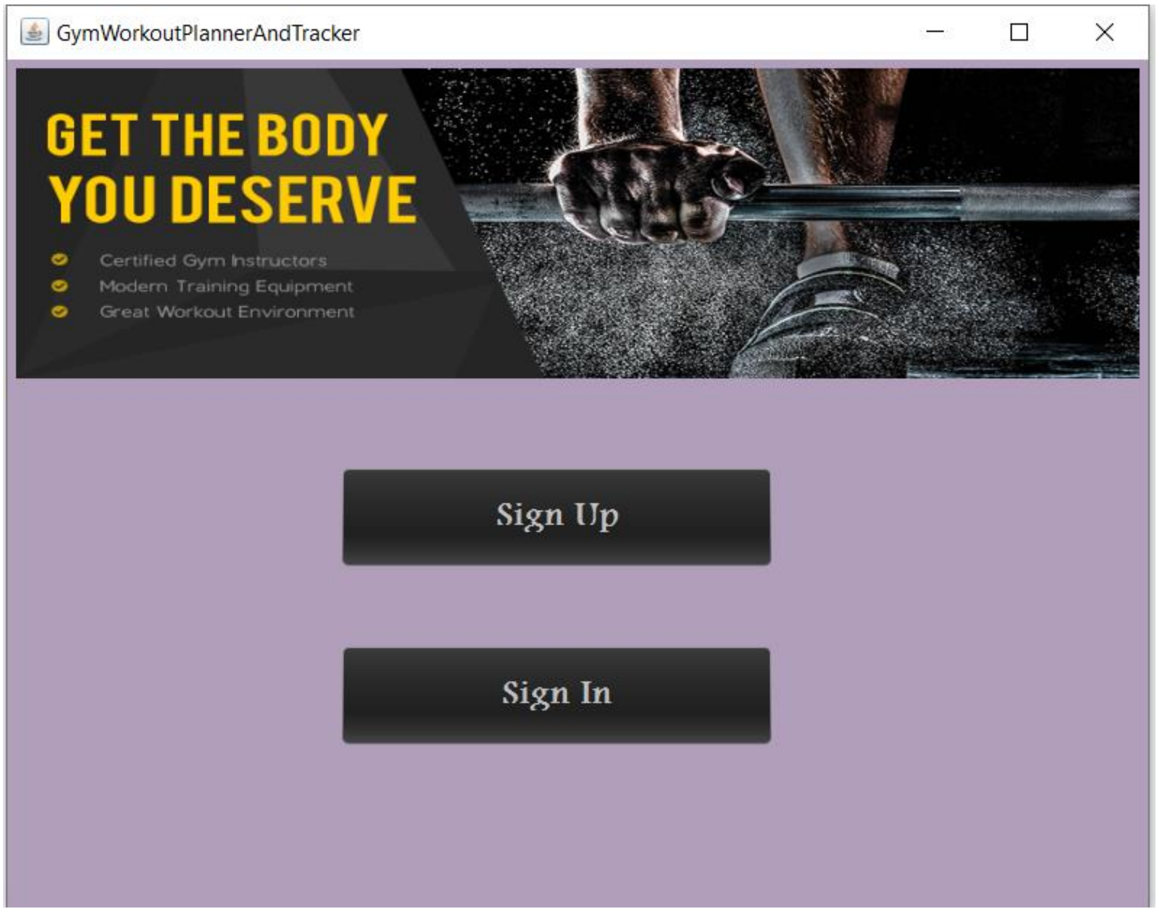
- Serialization: The classes that implement the Serializable interface are using the Serialization design pattern to convert object instances into a byte stream for storage or transmission over a network. This allows objects to be saved and restored back to their original state.

Overall, the code demonstrates the use of the Prototype design pattern for creating workout programs and the Observer design pattern for receiving notifications about changes in workout and nutrition programs.

Explained Output:

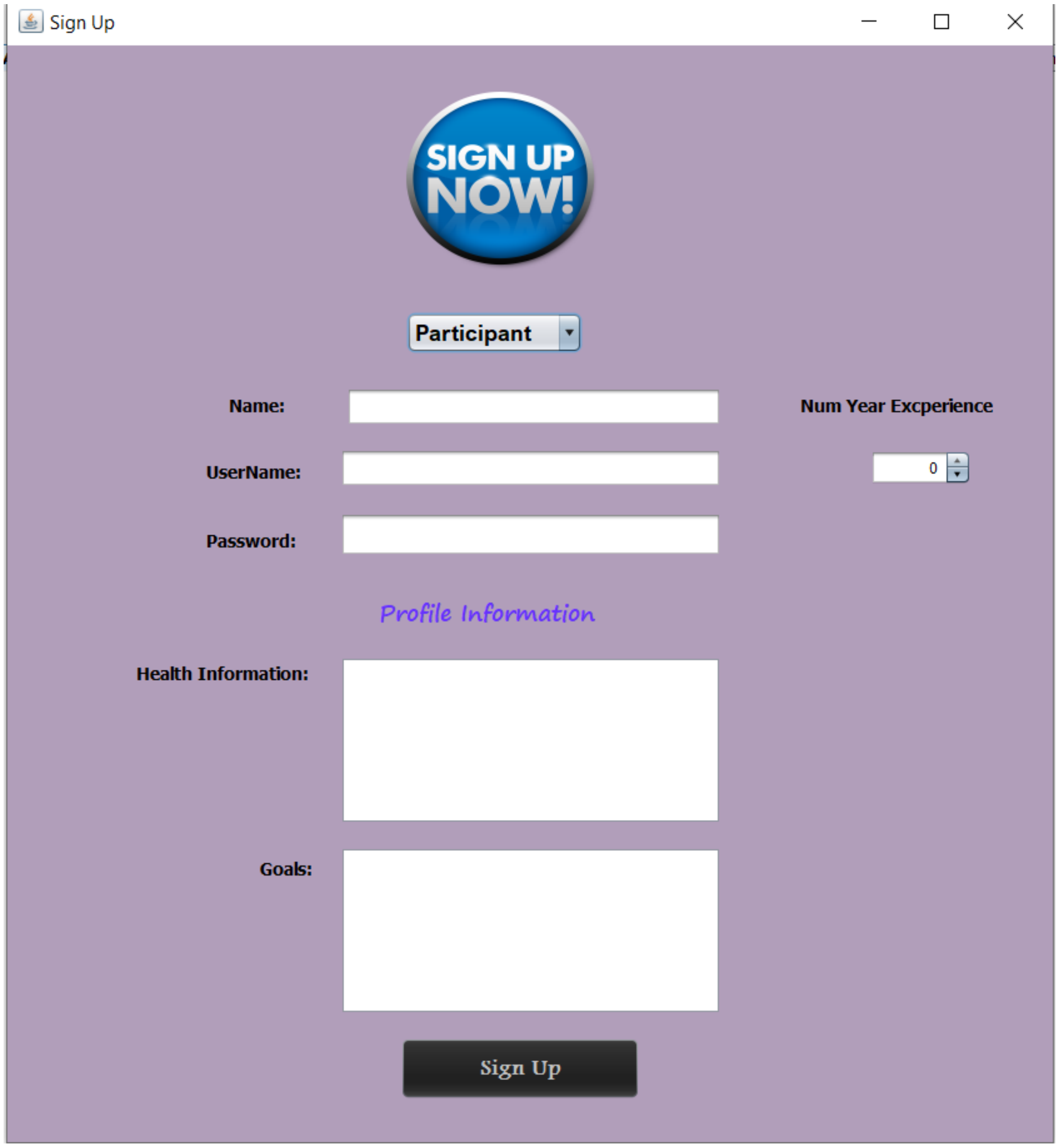
We use binary files to save and loads users and their information in the system.

MainFrame:


GymWorkoutPlannerAndTracker
GET THE BODY
YOU DESERVE
Certified Gym Instructors
Modern Training Equipment
Great Workout Environment
Sign Up
Sign In

In this frame user can sign up or login to the system, we have to know that when this main Frame load the **users** binary file is loaded as arraylist to the system and when user trying to close the program we ask him to save the changes or not, then save users arraylist as binary **users.ser**

Signup Frame:


Sign Up

Participant

Name:

UserName:

Password:

Num Year Excperience

0

Profile Information

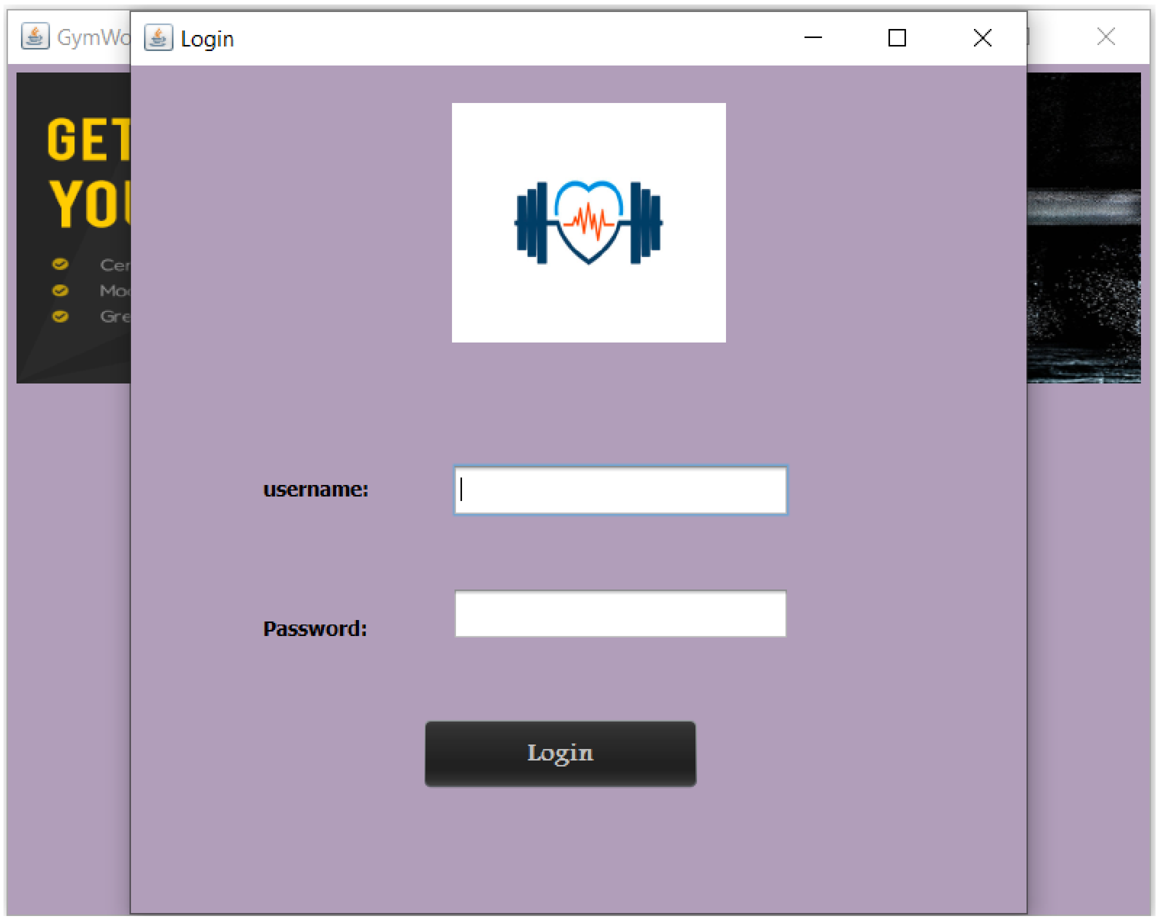
Health Information:

Goals:

Sign Up

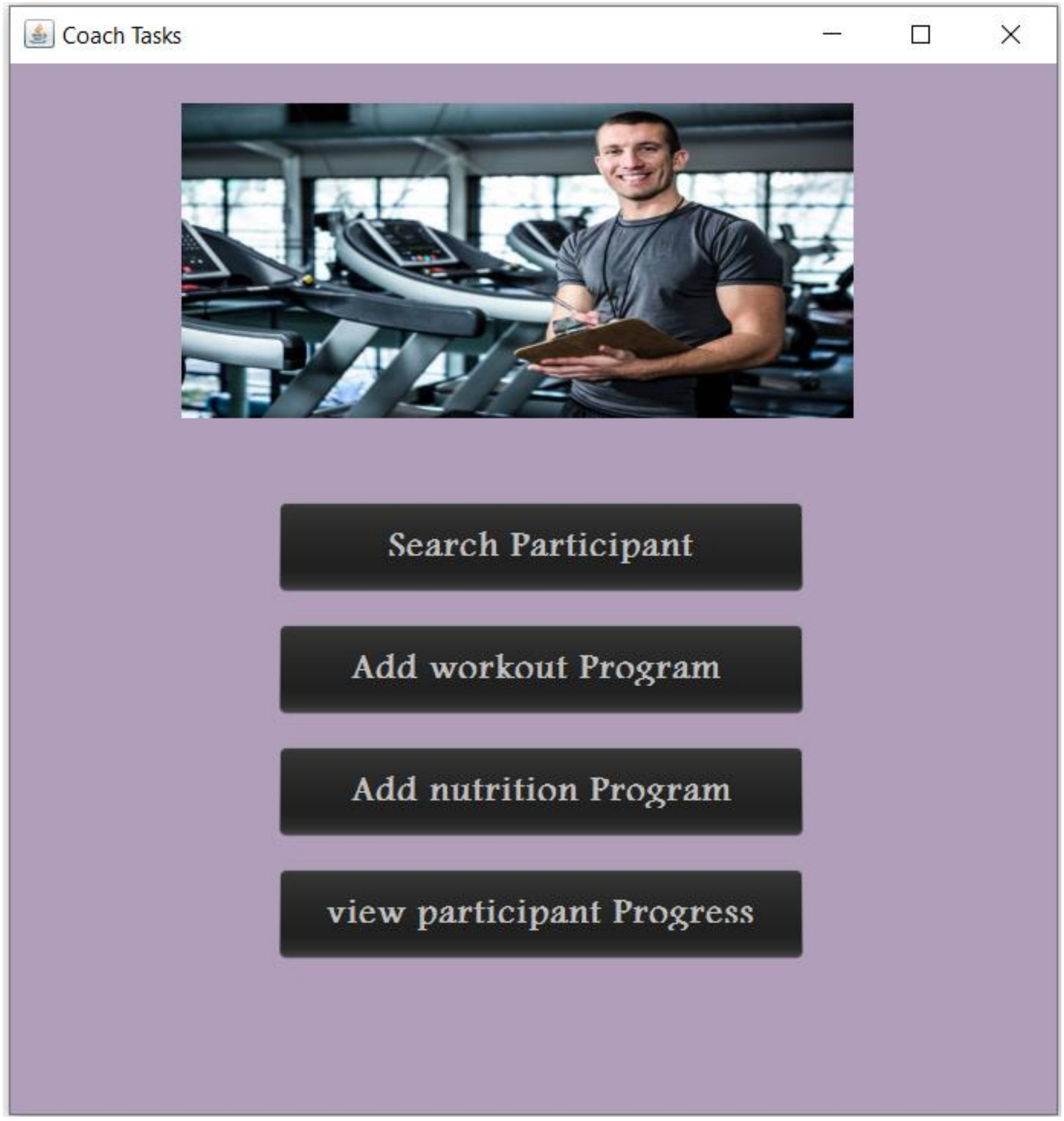
In this frame user can signup as participant or as coach where coach has number of year experience.

Login Frame:



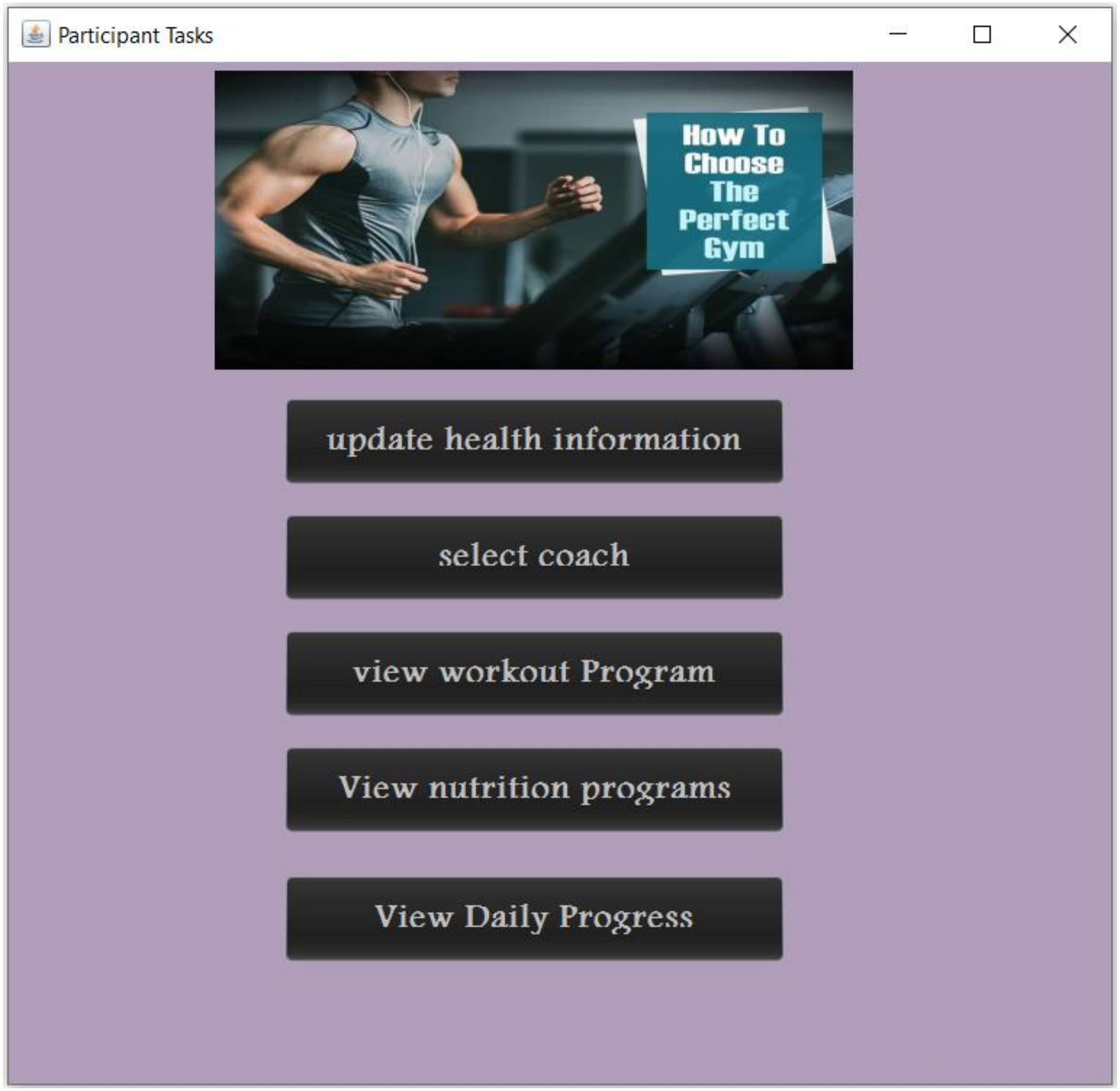
In this frame both participants and coach can access to the system by username and password.

Coach tasks Frame:



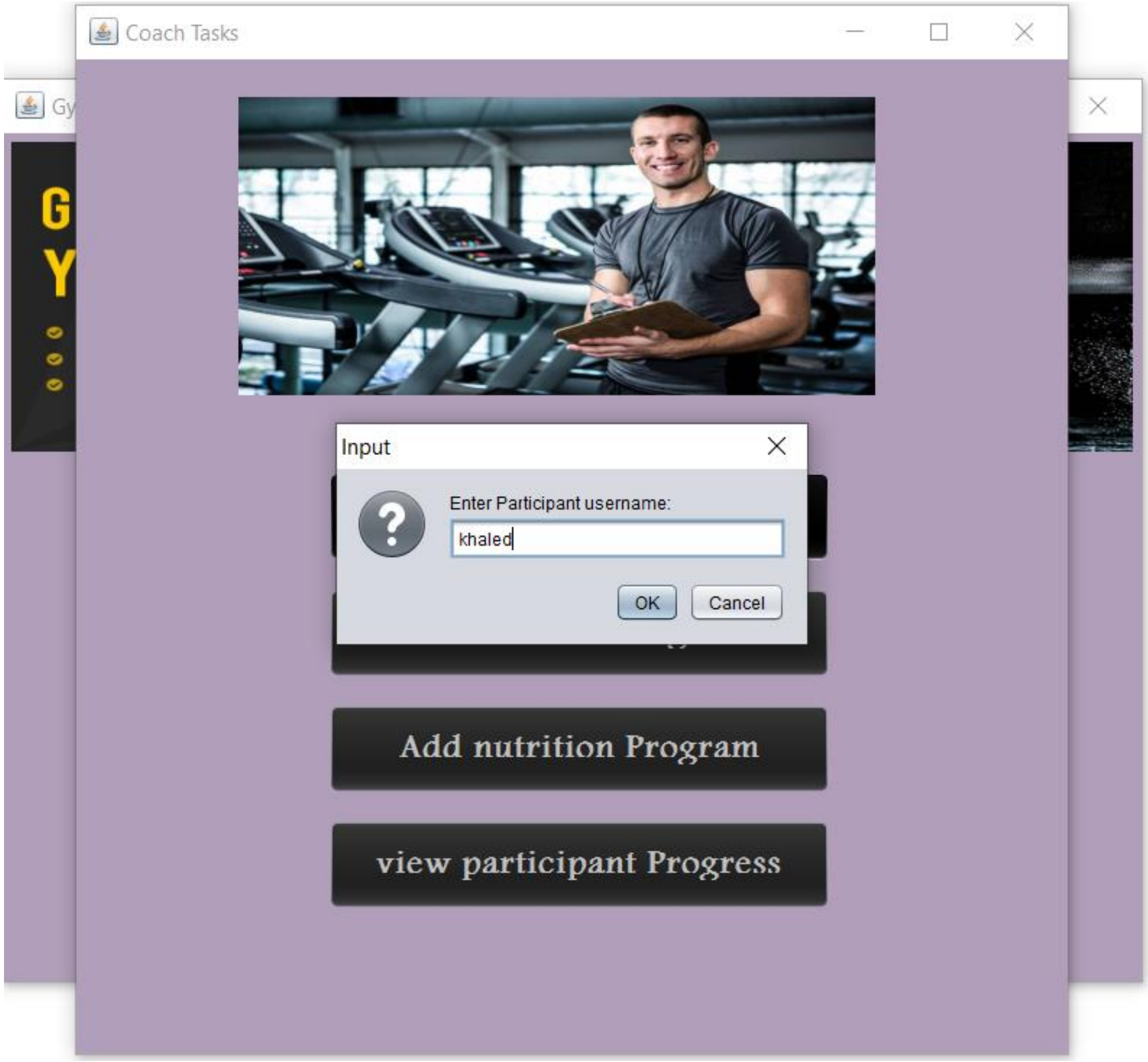
This frame shows the functionalities of the coach in the system.

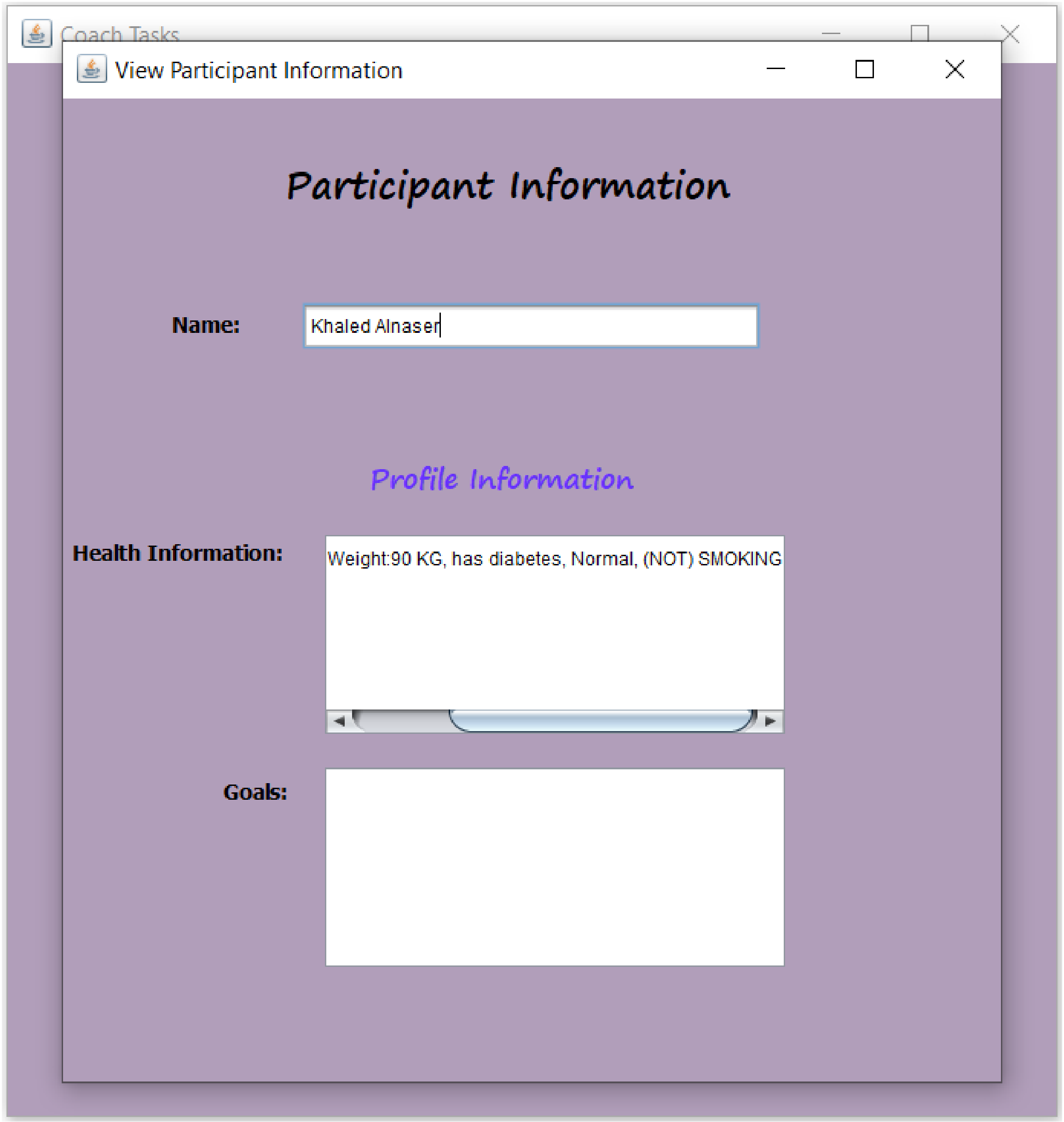
Participant task frame:



This frame shows the functionalities of the participant in the system.

View participant start of enter username of participant to see relevant information:





Coach Tasks

View Participant Information

Participant Information

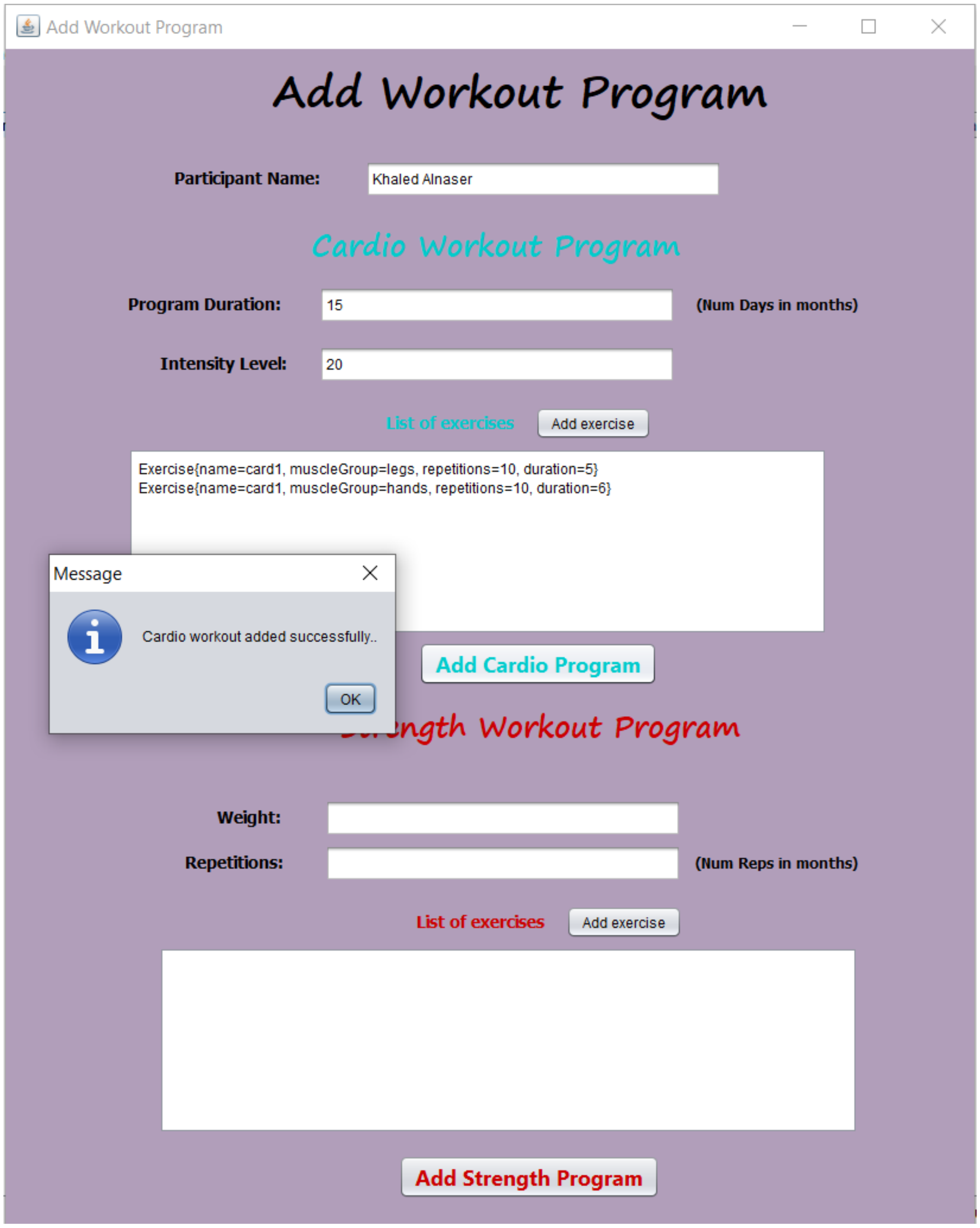
Name: Khaled Alnaser

Profile Information

Health Information: Weight:90 KG, has diabetes, Normal, (NOT) SMOKING

Goals:

This frame show details of target participant.



Add Workout Program

Add Workout Program

Participant Name: Khaled Alnaser

Cardio Workout Program

Program Duration: 15 (Num Days in months)

Intensity Level: 20

List of exercises

```
Exercise{name=card1, muscleGroup=legs, repetitions=10, duration=5}
Exercise{name=card1, muscleGroup=hands, repetitions=10, duration=6}
```

Message

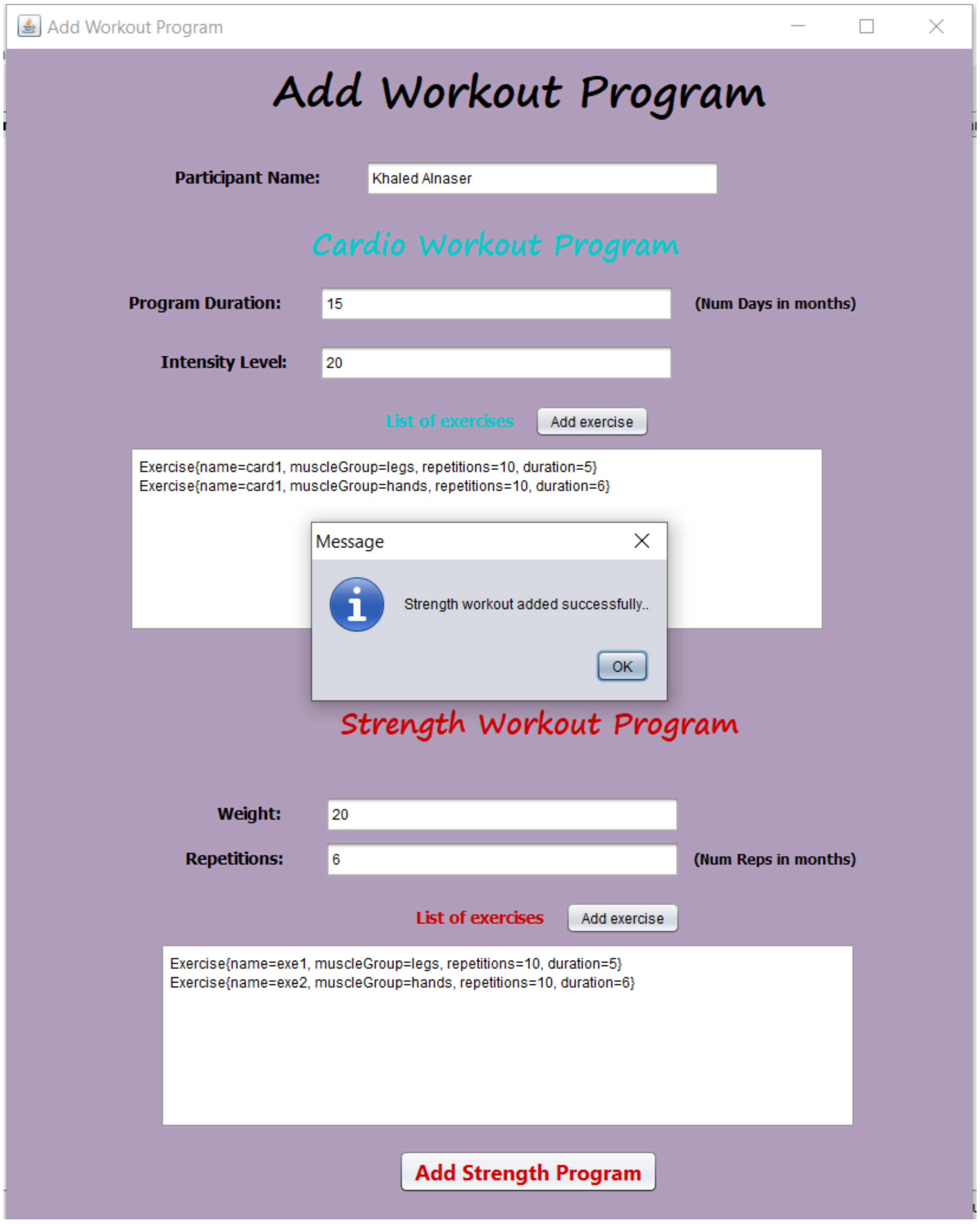
Cardio workout added successfully..

Weight:

Repetitions: (Num Reps in months)

List of exercises

Add cardio program



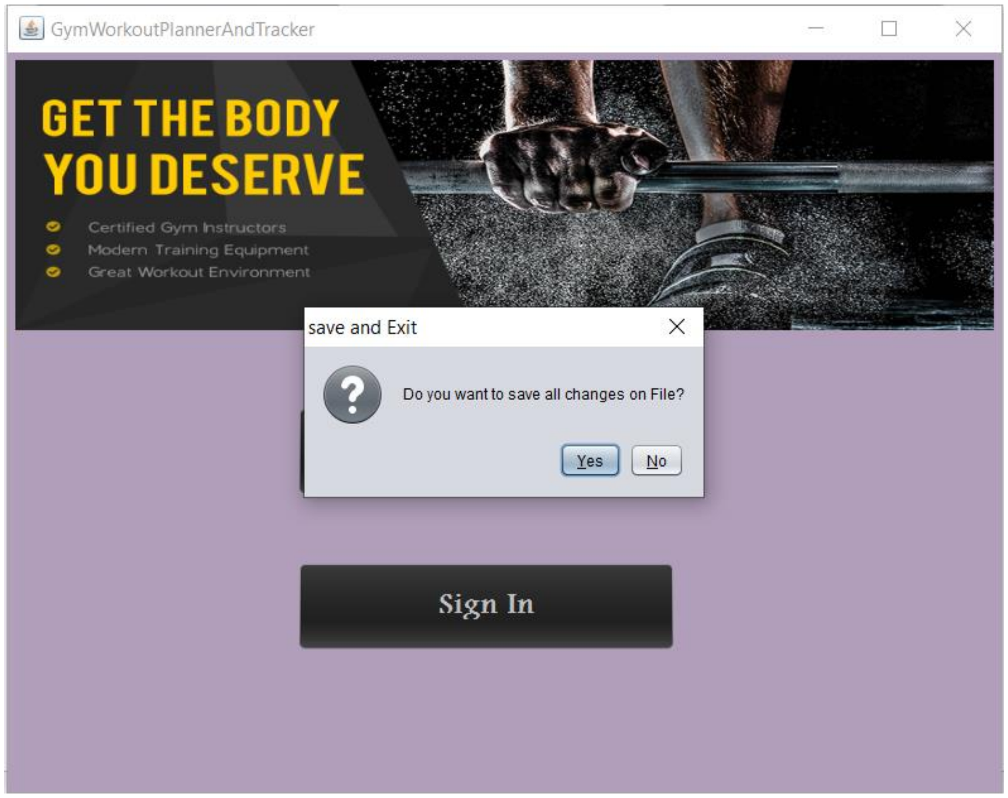
```
Exercise{name=card1, muscleGroup=legs, repetitions=10, duration=5}
Exercise{name=card1, muscleGroup=hands, repetitions=10, duration=6}
```

Strength workout added successfully..

```
Exercise{name=exe1, muscleGroup=legs, repetitions=10, duration=5}
Exercise{name=exe2, muscleGroup=hands, repetitions=10, duration=6}
```

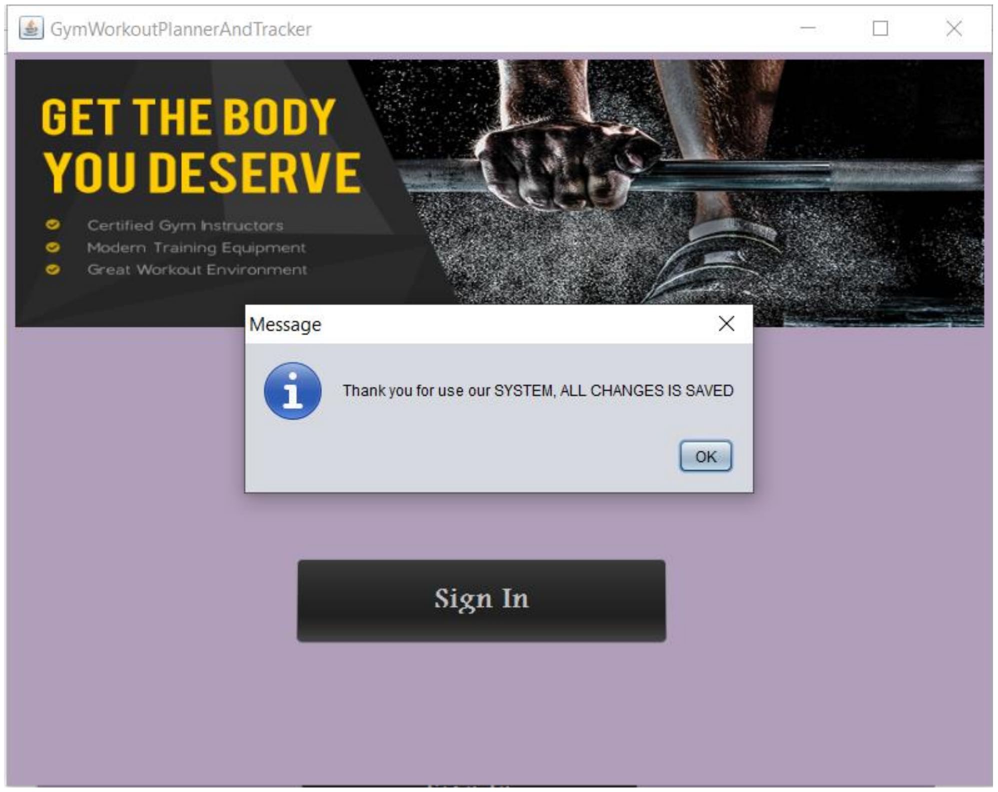
Add strength program

We check save changes before end the program:


GymWorkoutPlannerAndTracker
GET THE BODY YOU DESERVE
Certified Gym Instructors
Modern Training Equipment
Great Workout Environment
Yes
No
Sign In

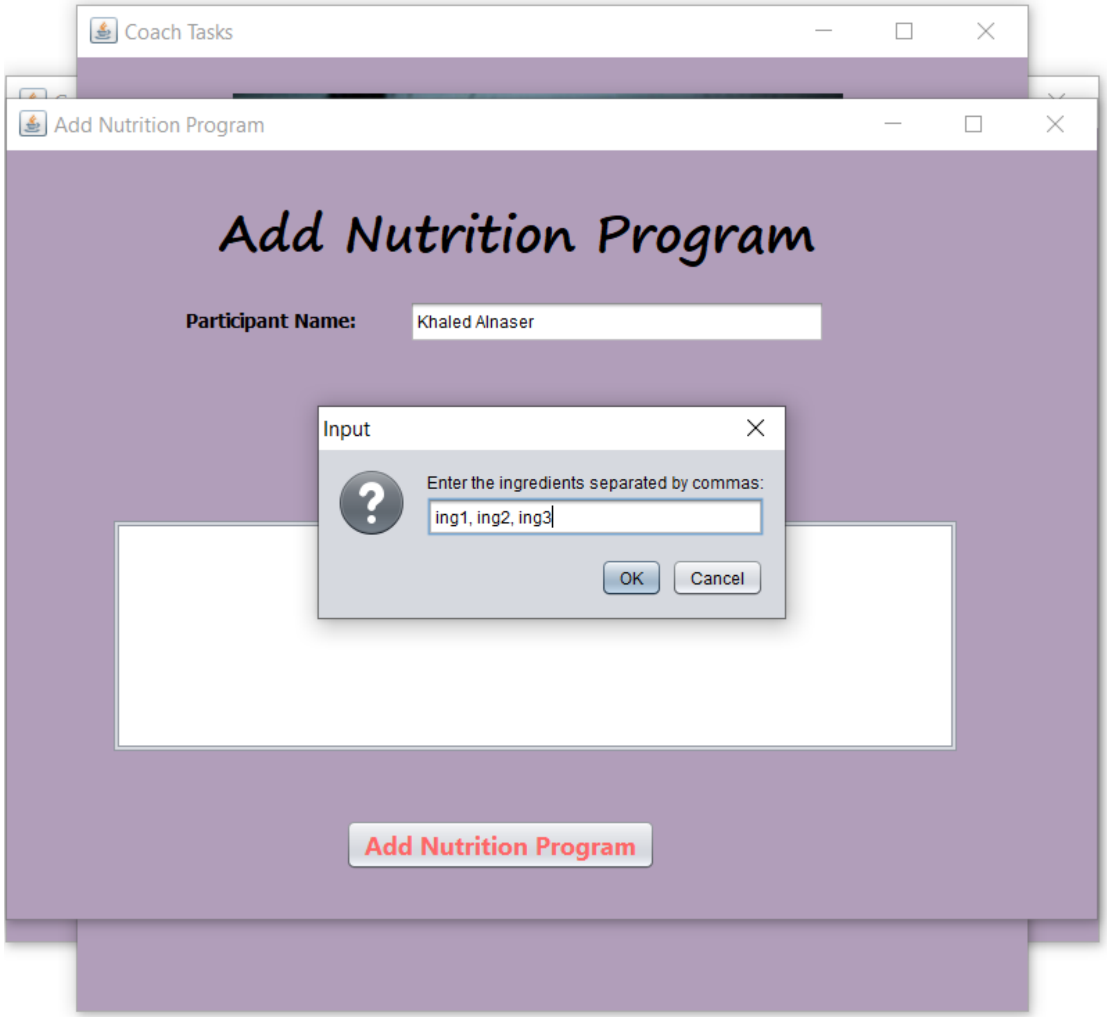
save and Exit

Do you want to save all changes on File?


GymWorkoutPlannerAndTracker
GET THE BODY YOU DESERVE
Certified Gym Instructors
Modern Training Equipment
Great Workout Environment
OK
Sign In

Message

Thank you for use our SYSTEM, ALL CHANGES IS SAVED


Coach Tasks

Add Nutrition Program

Add Nutrition Program

Participant Name: Khaled Alnaser

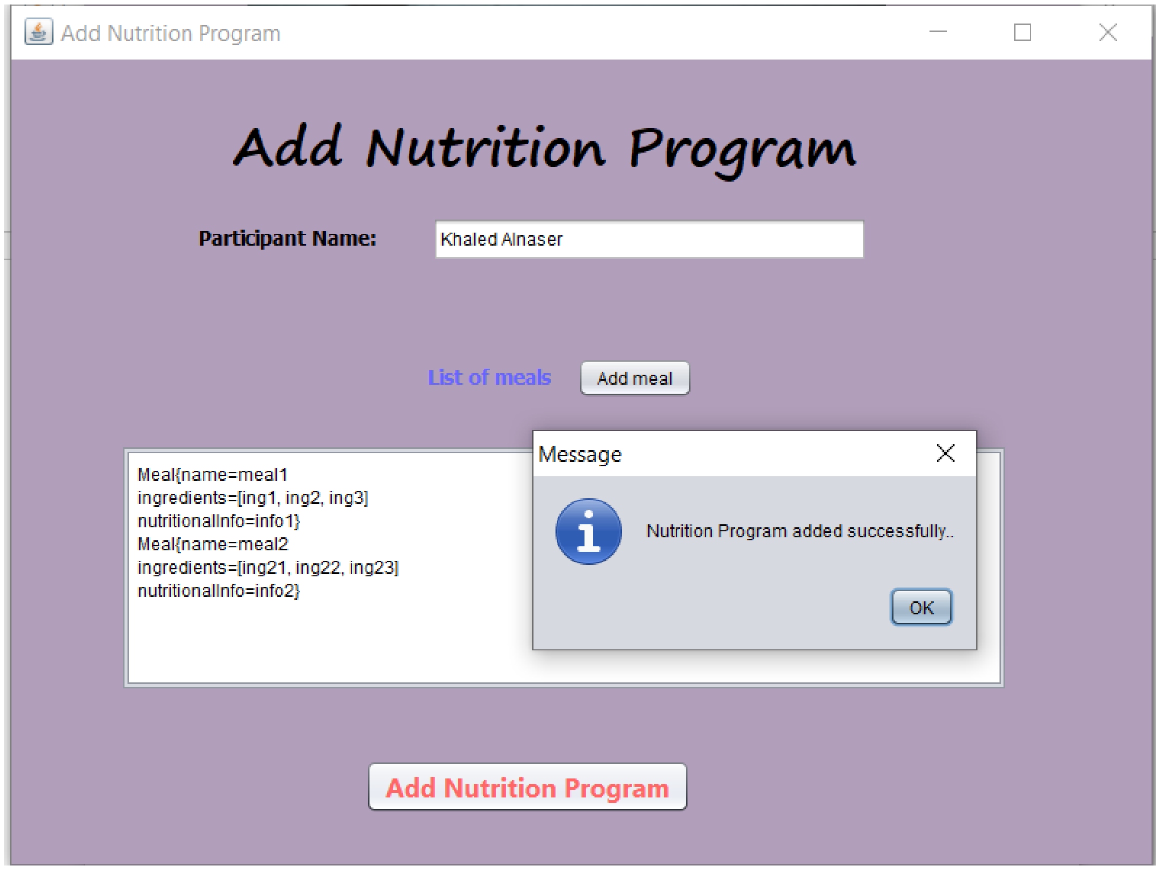
Input

Enter the ingredients separated by commas:

ing1, ing2, ing3

Note, when add ingredients place comma between them.

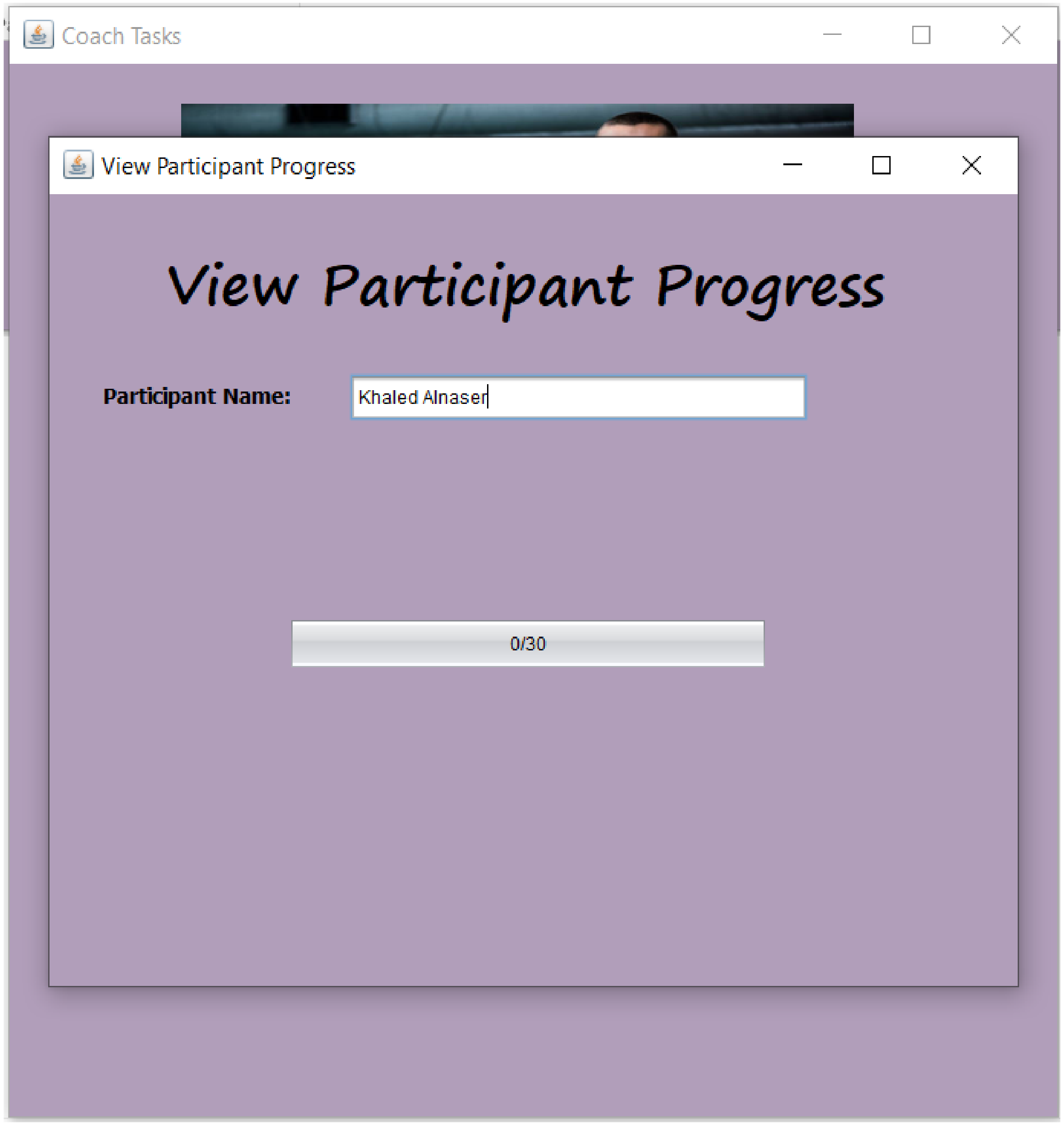
Add Nutrition Program:



Khaled Alnaser

```
Meal{name=meal1
ingredients=[ing1, ing2, ing3]
nutritionalInfo=info1}
Meal{name=meal2
ingredients=[ing21, ing22, ing23]
nutritionalInfo=info2}
```

Nutrition Program added successfully..

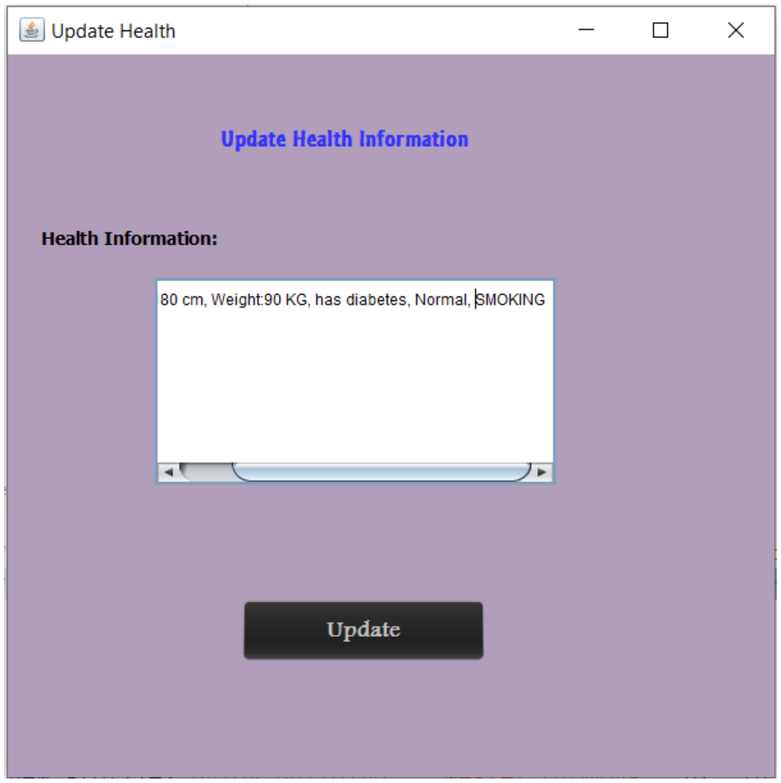


View Participant Progress

Participant Name: Khaled Alnaser

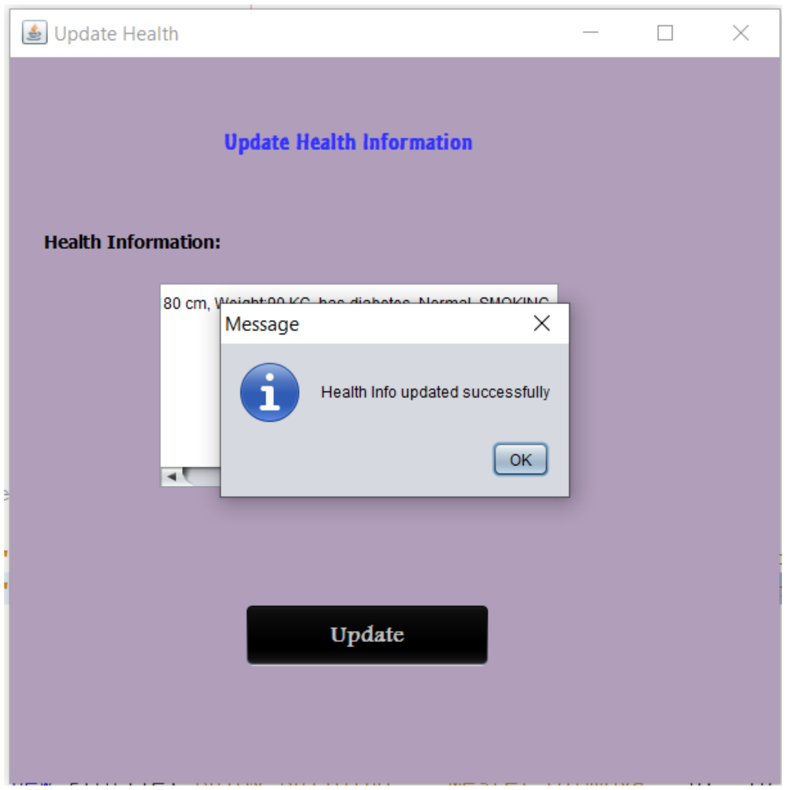
0/30

View participant progress



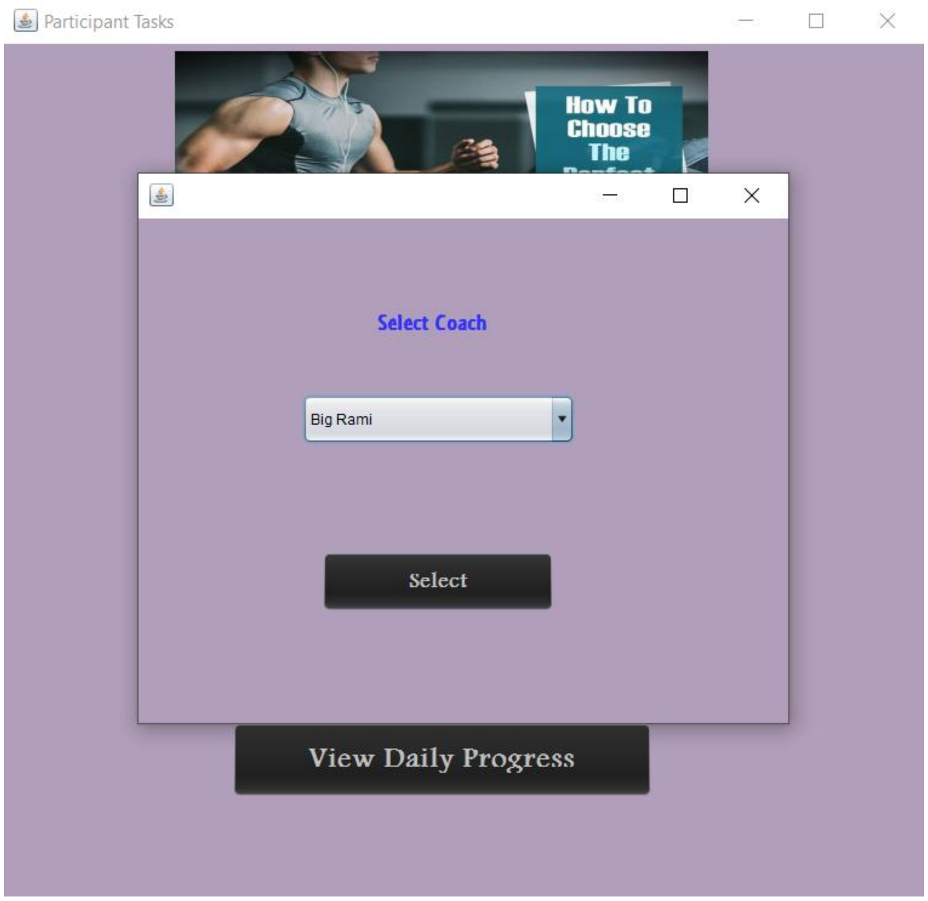
80 cm, Weight:90 KG, has diabetes, Normal, SMOKING

Update info

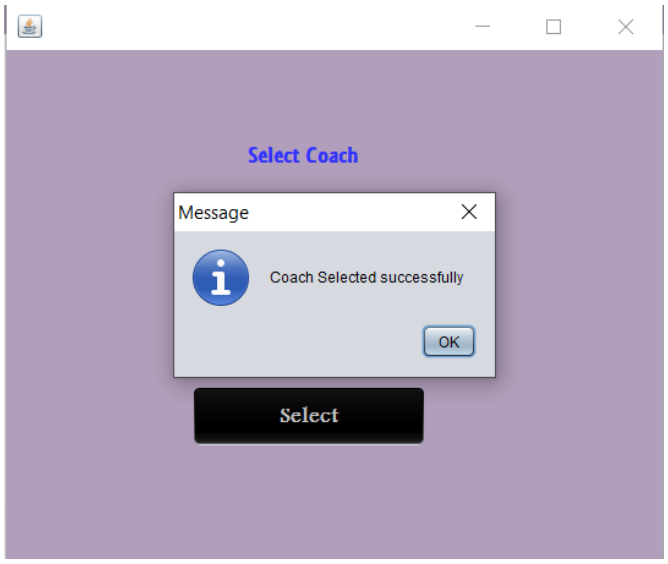


80 cm, Weight:90 KG, has diabetes, Normal, SMOKING

Health Info updated successfully

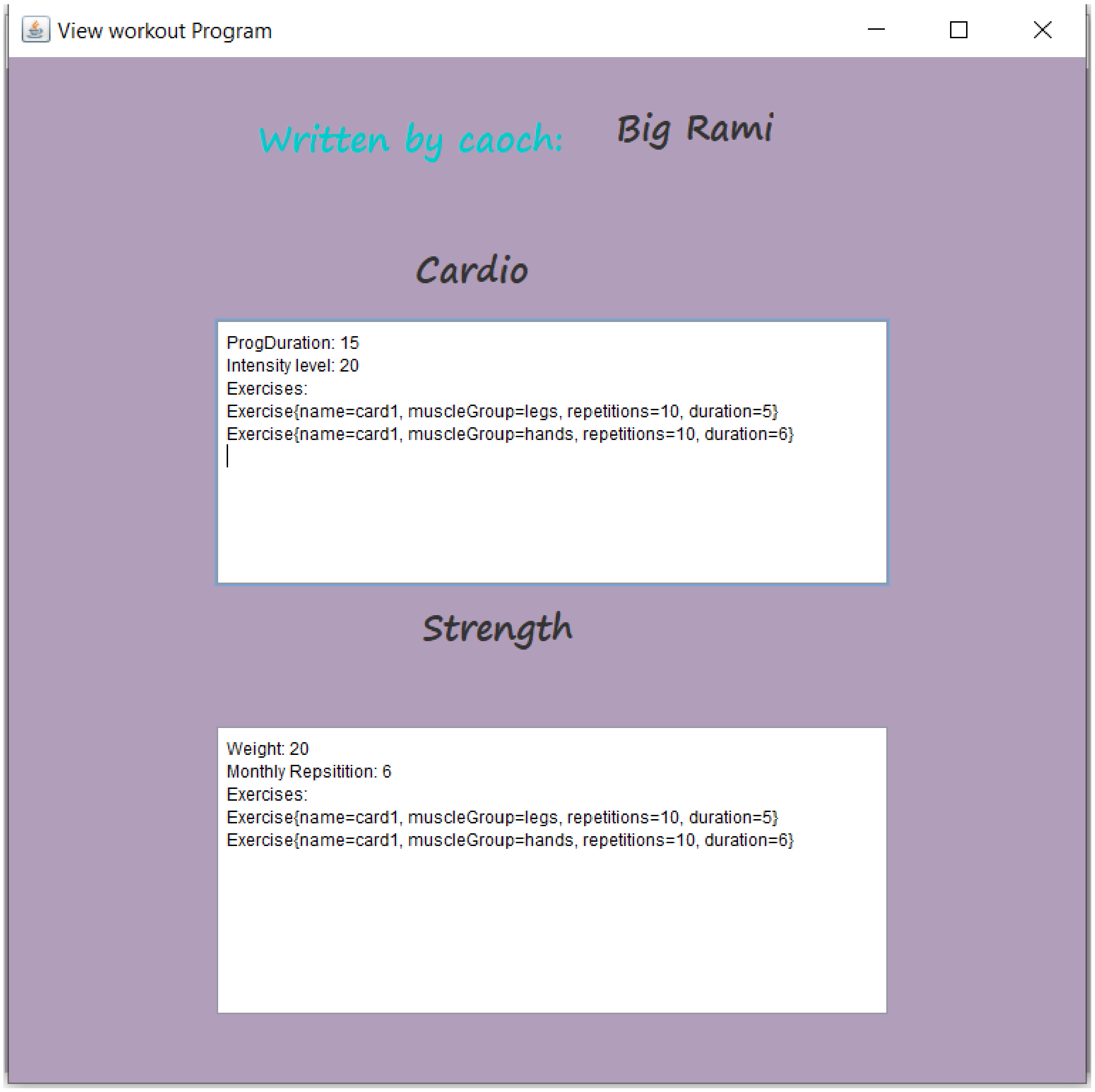


Select coach



Coach Selected successfully

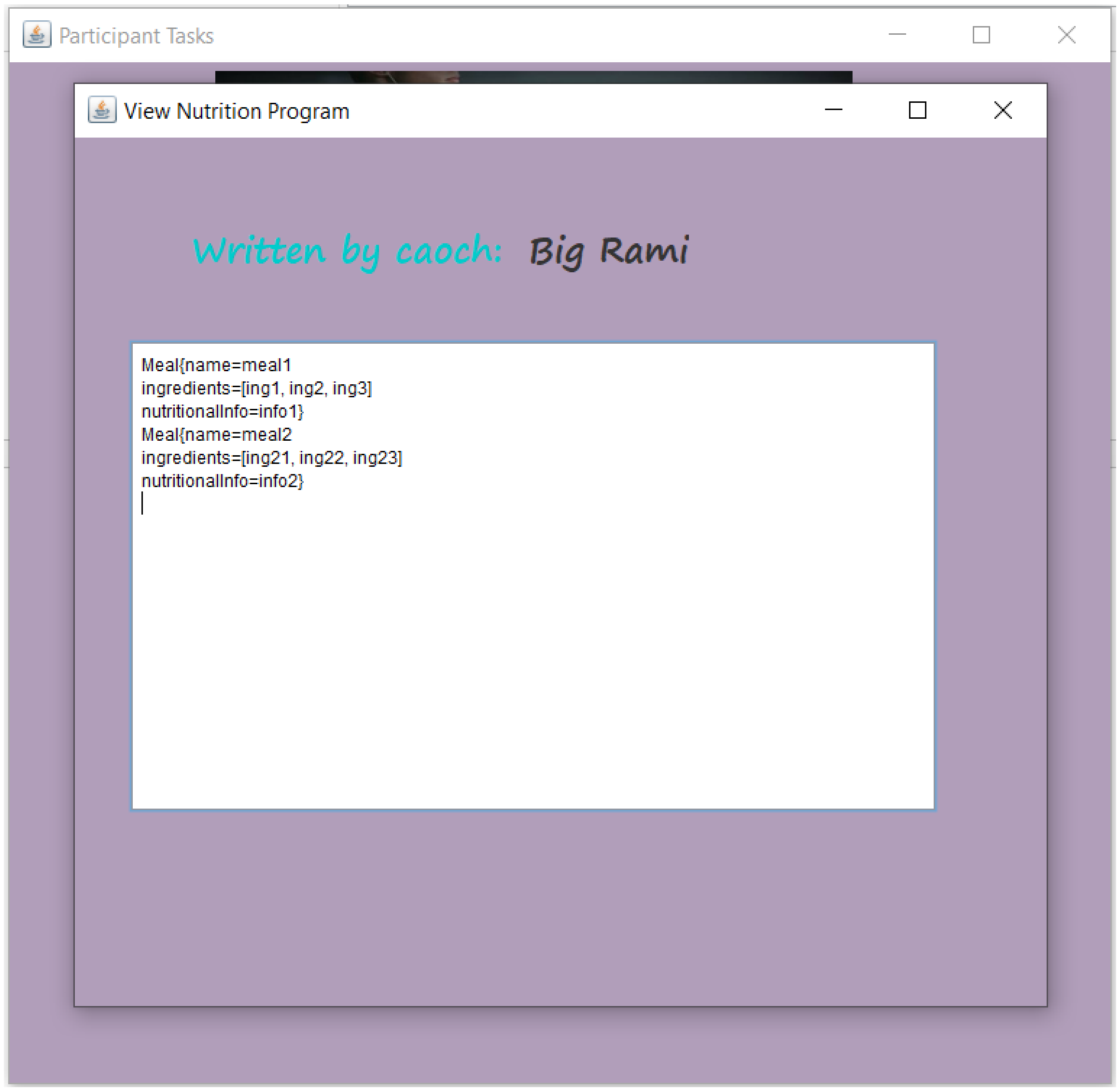
View workout program:



```
ProgDuration: 15
Intensity level: 20
Exercises:
Exercise{name=card1, muscleGroup=legs, repetitions=10, duration=5}
Exercise{name=card1, muscleGroup=hands, repetitions=10, duration=6}
```

```
Weight: 20
Monthly Repsitition: 6
Exercises:
Exercise{name=card1, muscleGroup=legs, repetitions=10, duration=5}
Exercise{name=card1, muscleGroup=hands, repetitions=10, duration=6}
```

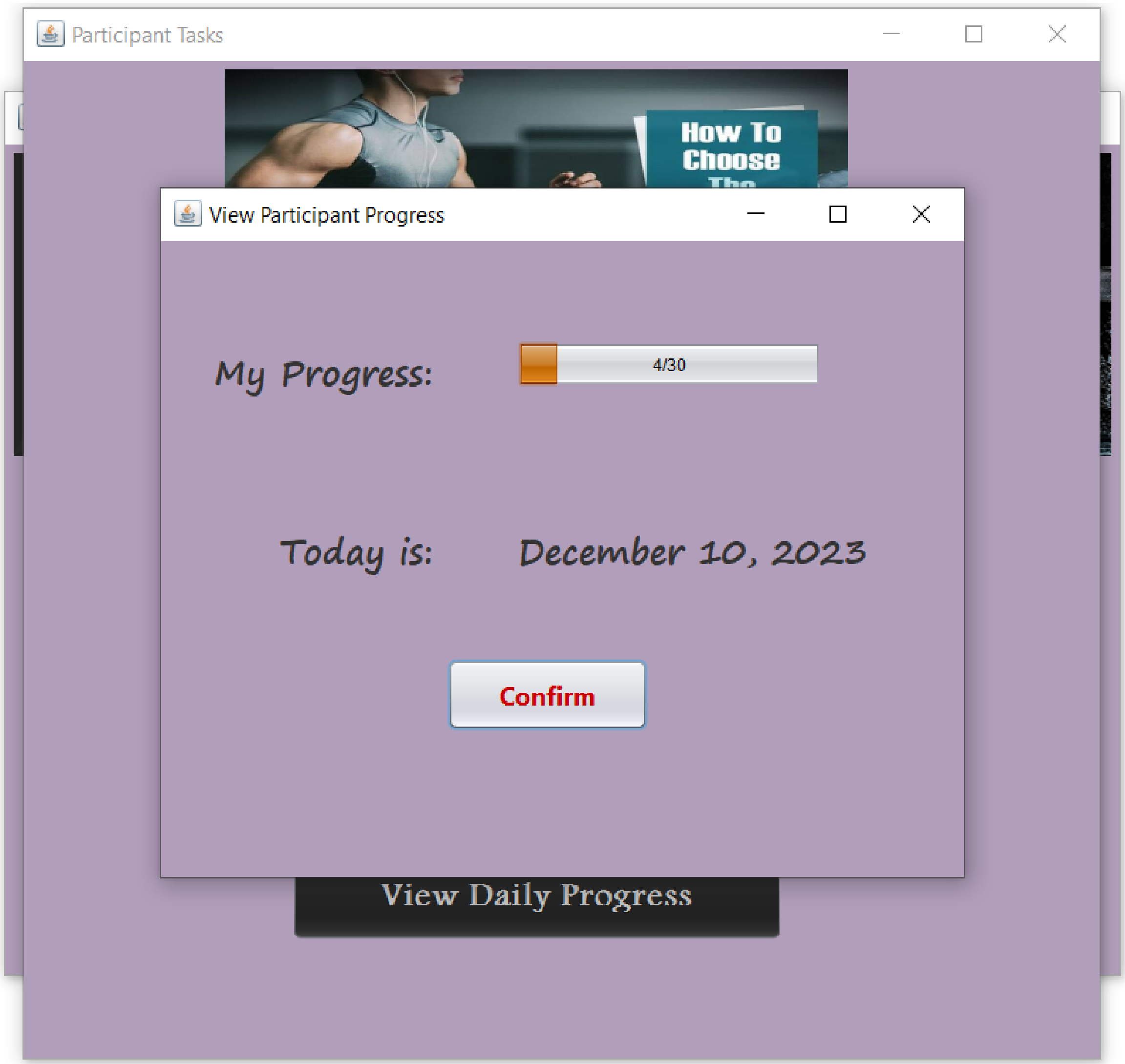
View workout programs.



```
Meal{name=meal1
ingredients=[ing1, ing2, ing3]
nutritionalInfo=info1}
Meal{name=meal2
ingredients=[ing21, ing22, ing23]
nutritionalInfo=info2}
```

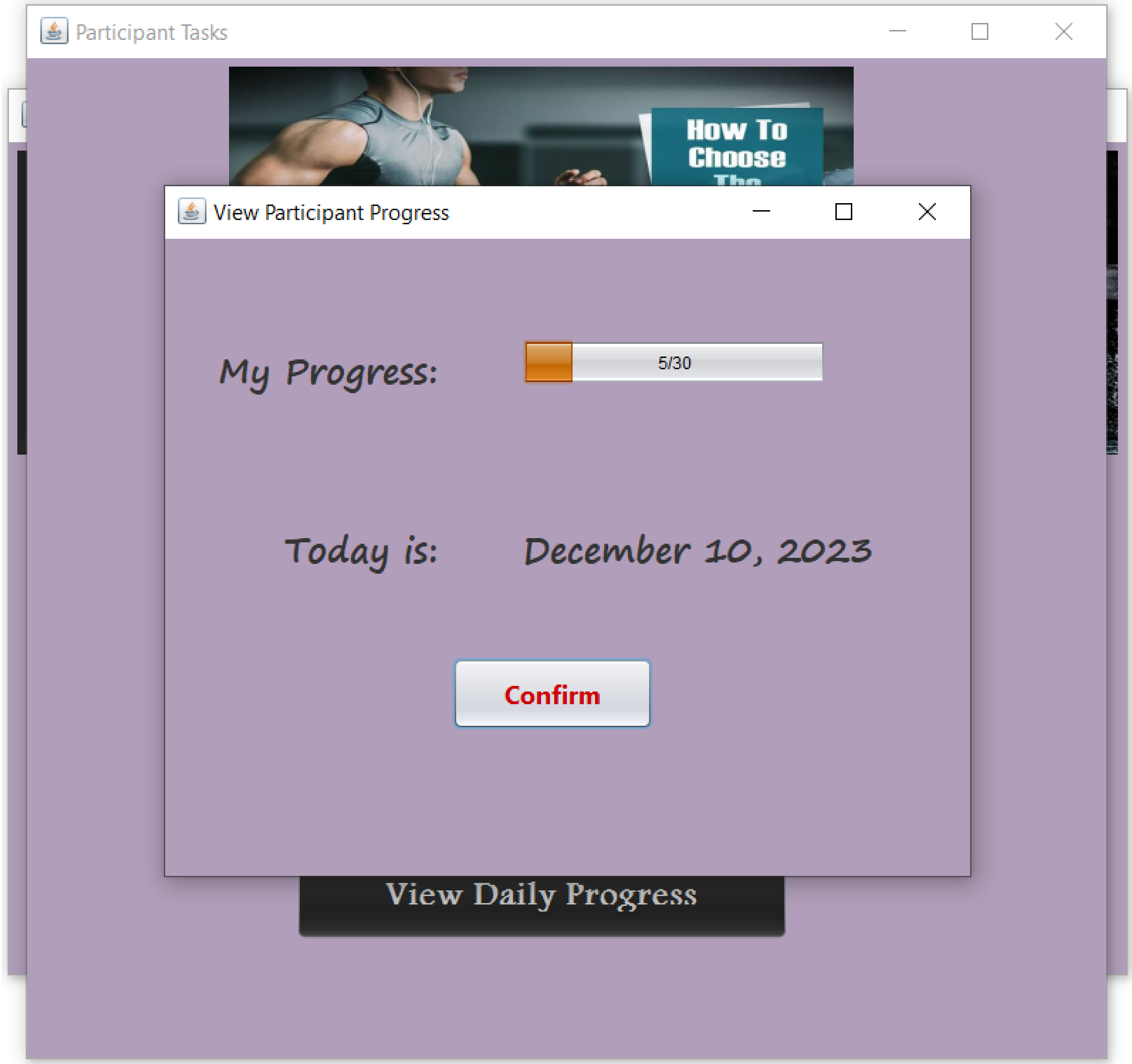
View nutrition info

Before confirm:

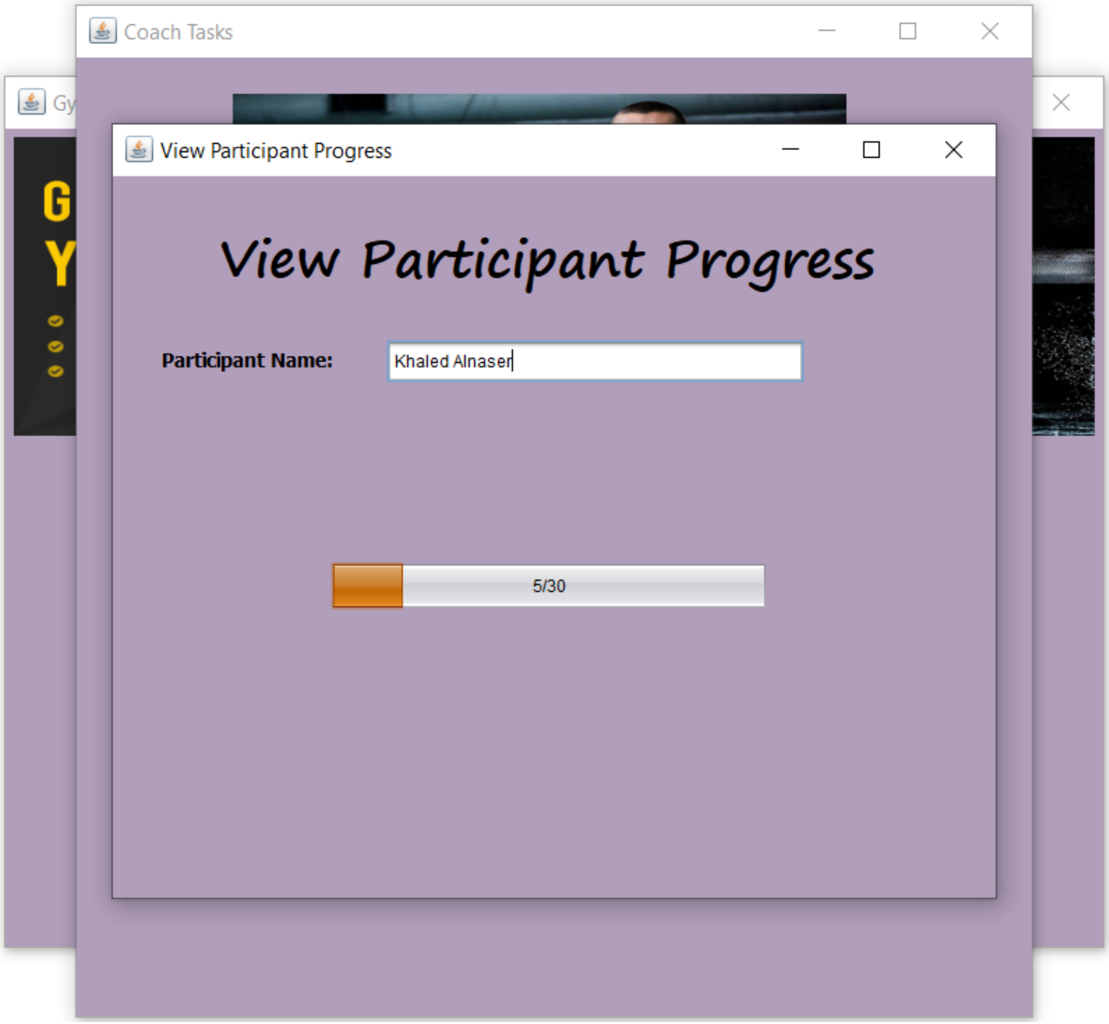


When applicant press confirm the daily achievement **updated**.

After confirm:



When coach check the progress:

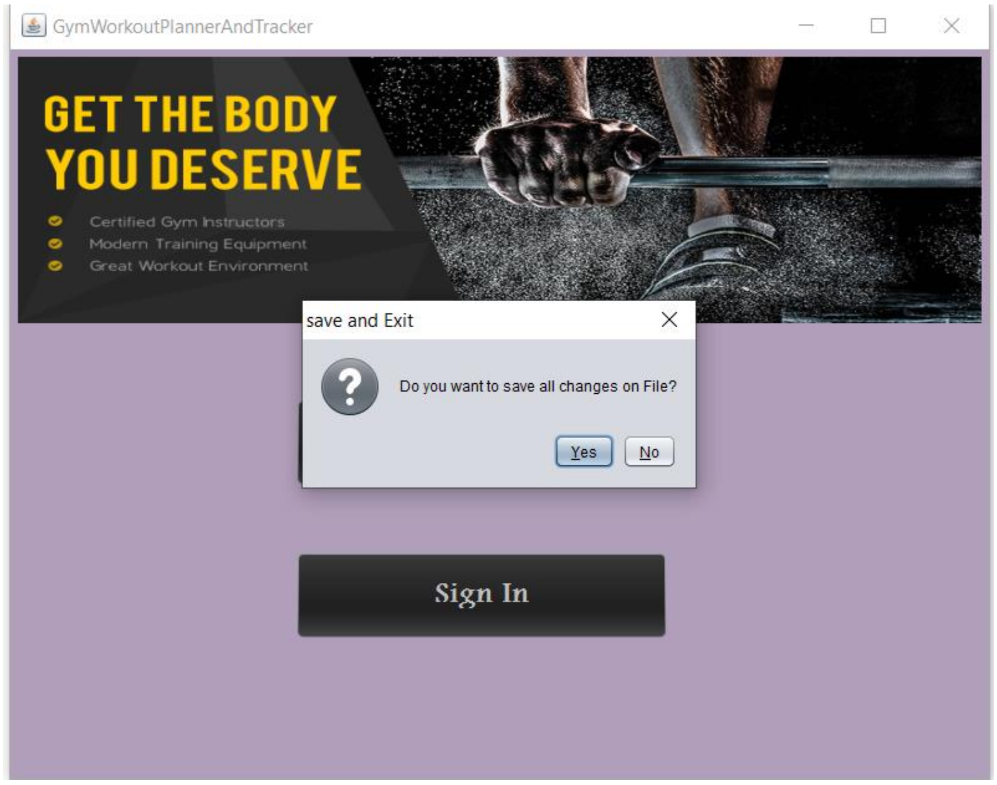


View Participant Progress

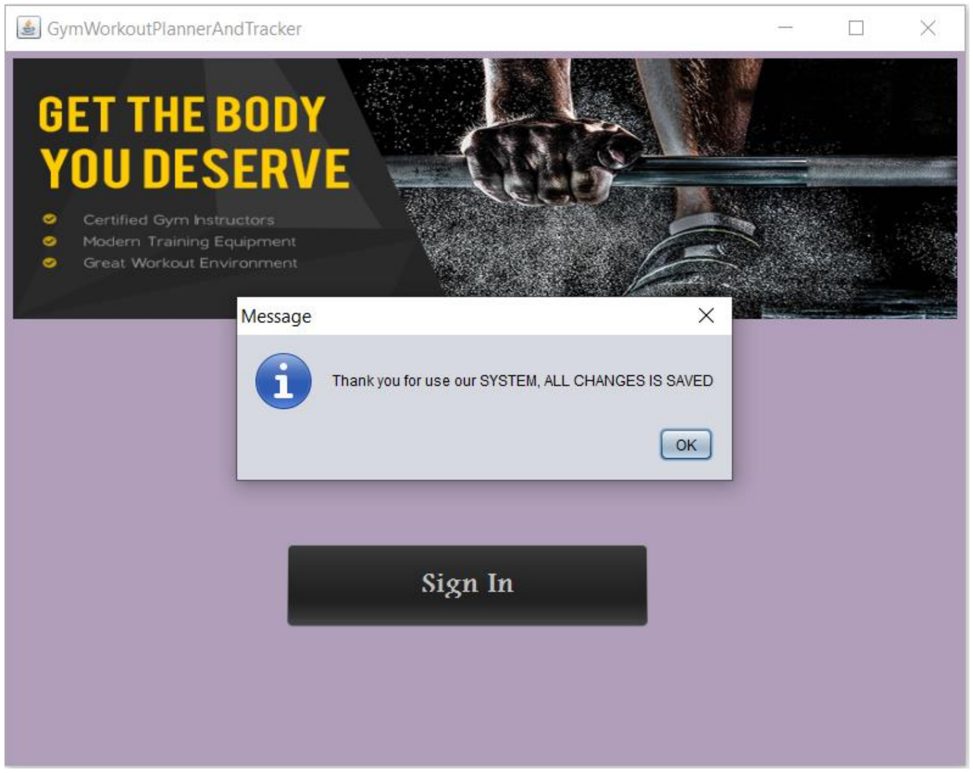
Participant Name: Khaled Alnaser

5/30

Check we save all changes:


GymWorkoutPlannerAndTracker
GET THE BODY YOU DESERVE
Certified Gym Instructors
Modern Training Equipment
Great Workout Environment
save and Exit
Yes
No
Sign In

Do you want to save all changes on File?


GymWorkoutPlannerAndTracker
GET THE BODY YOU DESERVE
Certified Gym Instructors
Modern Training Equipment
Great Workout Environment
Message
OK
Sign In

Thank you for use our SYSTEM, ALL CHANGES IS SAVED