Ain Shams University

Faculty of Engineering

My Gym System

Submitted By:

Ayman Hesham 16P3037

Yara Hossam 16P3002

Nada Tarek 16P6053

Noha Ibrahim 16P2001

Caroleen Mourad 16P6019

Submitted to:

Dr. Gamal Ibrahim Abdelshafy

Eng.Lara Wahib

Spring 2018

Abstract

Physical fitness is very important for a healthy and tension free life. "My Gym" program is a complete gym system which cares about all of its members and their memberships, employees, and activities. It is a system that is designed to manage day to day business by allowing the user to add or remove members, search for a member by searching for their ID, or to view the list of members. The system is designed to be user friendly that requires minimal user training.

Also, it allows us to add or remove an employee which helps in storing information about employees and coaches working in the gym for the safety of trainees, besides searching for an employee and viewing list of employees. Furthermore, for ensuring the progress of the trainees, the system can follow up each member through the member's Inbody to make sure that the trainee is approaching their target within the shortest time that suits their body and health. It also views list of all Inbodies and you can search for a member's Inbody by using the member's ID so that all of the member's information appear in the search result. And finally, for providing exciting activities for members, fitness classes can be added/ removed for each member, besides searching for a fitness class and viewing list of fitness classes.

Also, the system can prevent user's mistakes as it prevents him from typing letters in the fields that are required to be filled with numbers, for maintaining the accuracy upon filling information.

Finally, any changes in the database depend on the employee since he's the only one who can access and change in it according to members' and employees' verbal requests

Table of Contents

1. Introduction

1.1 Purpose

The purpose of the project is to develop a software system that facilitates the data storage and data usage to store the records of the clients; their information, InBodies and Fitness classes that they attend, in order to guarantee each trainee's progress. The staff (employees) has the ability to access the database and make changes to it. Also, for the safety of trainees it gathers information about employees and trainers.

1.2 List of Definitions

InBody: It is a test that provides a comprehensive view of body composition balance.

Java.awt, javax.swing: Packages provided in java language.

BMI: (Body Mass Index)It is the measure of weight relative to the height.

PBF: Percentage of Body Fat.

1.3 Scope

Our proposed “My Gym System” is for those who own a gym business and need to organize all the data, so we created this application to store all the information needed about members and the activities inside the gym.

1.4 Overview

The system handles the information of people working and training in the gym so we provided detailed description of the application throughout this document starting from general description ,System requirements,Use case diagram

2. General Description

2.1 Product Perspective

Java and an operating system are recommended for usage of the application.

2.2 General Capabilities

The Application is capable of storing thousands of information about the members. It contains the ability of adding and removing members, Fitness classes, Inbodies and employees. It can also provide lists for all members, employees, Inbodies and fitness classes.

2.3 General Constraints

There are a few constraints that the user must be aware of, such as that all of the information is required, nothing is optional. Also, The letters cannot be entered in weight ,height or number of months of registeration.

2.4 User Characteristics

This application can be used only by adults for those who run a gym business.

2.5 Environment Description

Since it is a software program therefor the place must contain a PC (a computer or a laptop) and for sure a suitable source of electricity and it must be kept away from reach of children. Also, it has to have enough memory to store information.

2.6 Assumptions and dependencies

The only person allowed to access this app and make changes is a specific employee in the gym as he takes the paper filled applications from the receptionist and transfers them to the database of the gym by storing all of the information on the app.

2.7 Other resources needed

It's recommended to use Netbeans IDE 0.2 .

The used libraries:

- Java.awt
- Java.awt.event
- Javax.swing
- Java.util

3. System Requirements

3.1 Functional Requirements

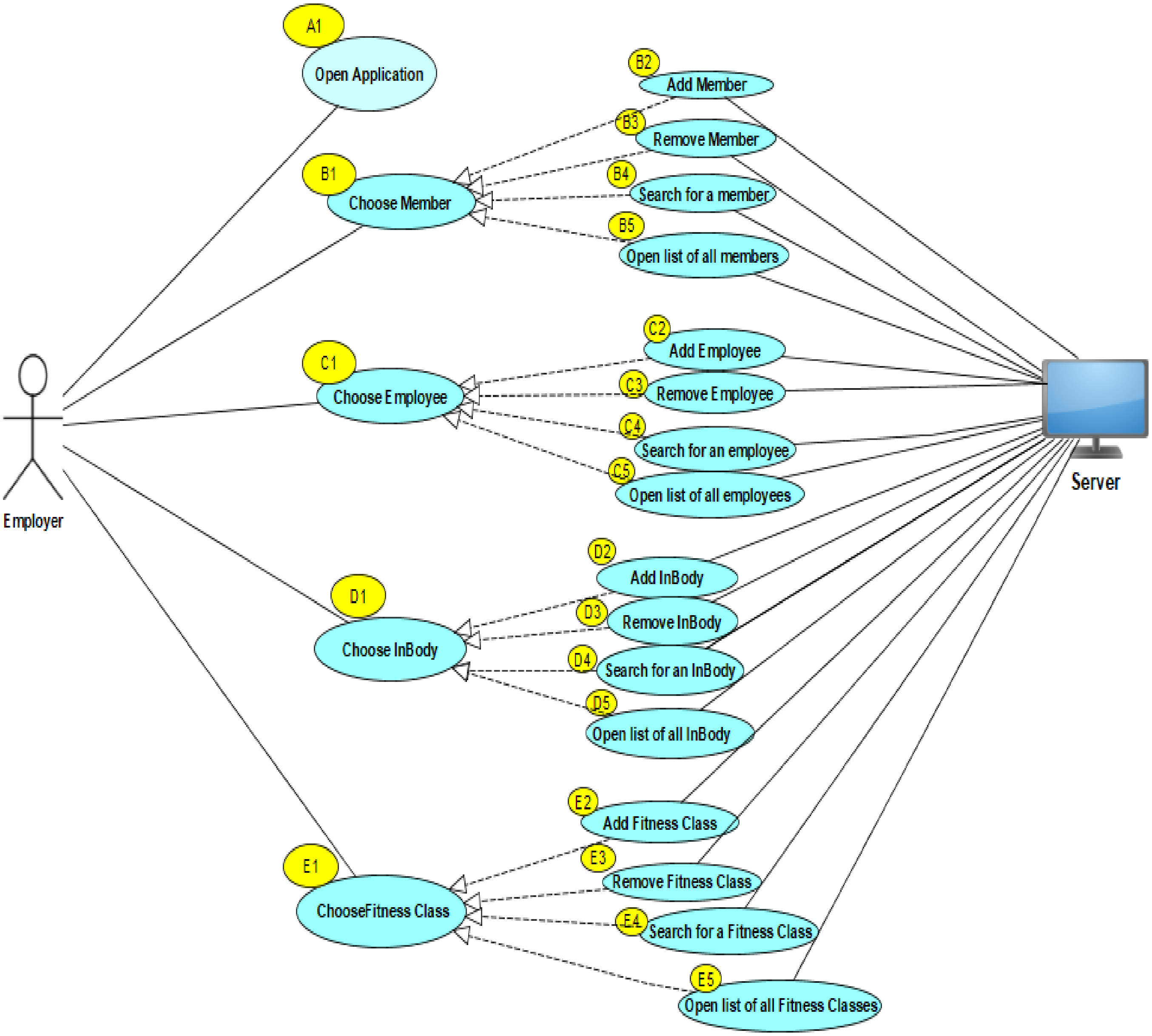
The system provides:

- Adding, removing and searching for a member.
- Adding, removing and searching for an employee.
- Adding, removing and searching for an Inbody.
- Adding, removing and searching for fitness class**.**
- Checking the lists of all members, employees, Inbodies and fitness classes**.**

3.2 Non-Functional Requirements

- Large Data memory.
- Java Language.
- Available 24 hours per day.
- User friendly.

4. Use Case Diagram



5. Narrative Description of Use Cases

1st use case:

Use case name: open app

Related requirement: B1,C1,D1,E1

Goal in context: The app opens and ready to click on menu

Pre-condition: The app must be downloaded first on the computer

Successful end condition: The app opens

Failed end condition: The app does not open

Primary actor: Employer

Secondary actor: -

Trigger: the app is loading

Included case: --

Main flow: The user opens the computer then right click choose open then the app opens

Extensions: Computer shut downs

2nd use case:

Use case name:Choose member

Related requirement: B2

Goal in context: Apply operations in member menu

Pre-condition: Open the app

Successful end condition: the menu opened so chose an option

Failed end condition: couldn't open the menu

Primary actor: Server

Secondary actor: Employer

Trigger: Decide which menu to click on

Included case: --

Main flow: The employer opens the app then decide which menu he want to click on then select from the drop down menu.

Extensions: The application freezes.

3rd use case:

Use case name: Add member

Related requirement: B1

Goal in context: The system adds the new member selected by employer

Pre-condition: Click on add member

Successful end condition: The member is registered

Failed end condition: It doesn’t register the member

Primary actor: Server

Secondary actor: Employer

Trigger: the app is loading

Included case: B1

Main flow: The employer opens the app then selects the add member from the drop down menu after that the system registers the member

Extensions: the system collapse

4th use case:

Use case name: Remove member

Related requirement: B1

Goal in context: The system removes the new member selected by employer

Pre-condition: Click on remove member

Successful end condition: The member is removed

Failed end condition: Couldn’t remove the member

Primary actor: Server

Secondary actor: Employer

Trigger: the app is loading

Included case: B1

Main flow: The employer opens the app then selects remove member from the drop down menu after that the system removes the member information

Extensions: the system collapse

5th use case:

Use case name: Search member

Related requirement: B1

Goal in context: The system searches for the member

Pre-condition: Click on search for a member

Successful end condition: The information of the member appears

Failed end condition: Couldn’t search for the member

Primary actor: Server

Secondary actor: Employer

Trigger: the app is loading

Included case: B1

Main flow: The employer opens the app then selects search for a member from the drop down menu after that he writes the member ‘s ID the system provides the member information

Extensions: the system collapse

6th use case:

Use case name: Open list of all members

Related requirement: B1

Goal in context: The system shows list of all the members

Pre-condition: Click on search for a member

Successful end condition: list of all the members is opened

Failed end condition: Couldn’t open the list of all the members

Primary actor: Server

Secondary actor: Employer

Trigger: the app is loading

Included case: B1

Main flow: The employer opens the app then selects list of all members from the drop down menu after that list of all the members registered in the gym appears.

Extensions: the system collapse

7th use case:

Use case name: Choose employee

Related requirement:C2

Goal in context: Apply Operations in employee menu

Pre-condition: Open the app

Successful end condition: the employee menu opened so chose an option

Failed end condition: couldn’t open the menu

Primary actor: Server

Secondary actor: Employer

Trigger: the app is loading

Included case: --

Main flow: The Employer opens the app then decide which menu he wants to click on then selects it

Extensions: the app freezes

8th use case:

Use case name: add emloyee

Related requirement:C1

Goal in context: The system adds the new employee

Pre-condition: Opens the app

Successful end condition: The employee is added

Failed end condition: The employee is not added

Primary actor: Server

Secondary actor: Employer

Trigger: waiting for the system to add the employee

Included case: C1

Main flow: The employer opens the app then selects add employee menu from the employee drop down menu after that the system registers the employee

Extensions: the system collapses

9th use case:

Use case name: Remove employee

Related requirement: C1

Goal in context: The system removes the new employee selected by employer

Pre-condition: Click on member menu

Successful end condition: The employee is removed

Failed end condition: Couldn’t remove the employee

Primary actor: Server

Secondary actor: Employer

Trigger: the app is loading

Included case: C1

Main flow: The employer opens the app then selects remove employee from the drop down menu after that the system removes the employee information

Extensions: the system collapse

10th use case:

Use case name: Search employee

Related requirement: C1

Goal in context: The system searches for the employee

Pre-condition: Click on search for an employee

Successful end condition: The information of the employee appears

Failed end condition: Couldn’t search for the employee

Primary actor: Server

Secondary actor: Employer

Trigger: the app is loading

Included case: C1

Main flow: The employer opens the app then selects search for an employee from the drop down menu after that he writes the employee ‘s ID the system provides the employee information

Extensions: the system collapse

11th use case:

Use case name: Open list of all employees

Related requirement: C1

Goal in context: The system shows list of all the employees

Pre-condition: Click on search for an employee

Successful end condition: list of all the employees is opened

Failed end condition: Couldn’t open the list of all the employees

Primary actor: Server

Secondary actor: Employer

Trigger: the app is loading

Included case: C1

Main flow: The employer opens the app then selects list of all employees from the drop down menu after that list of all the employees registered in the application appears.

Extensions: the system collapse

12th use case:

Use case name: Select InBody

Related requirement: D2

Goal in context: Apply Operations in InBody menu

Pre-condition: Open the app

Successful end condition: the InBody menu opened so chose an option

Failed end condition: couldn’t open the InBody menu

Primary actor: Employer

Secondary actor: Server

Trigger: Decide which menu to click on

Included case: --

Main flow: The employer opens the app then decide which menu he want to click on then selects InBody menu

Extensions: The application freezes.

13th use case:

Use case name: Add InBody

Related requirement: D1

Goal in context: The system adds the InBody selected by employer

Pre-condition: Click on add InBody

Successful end condition: new InBody is registered

Failed end condition: It doesn’t register the Inbody

Primary actor: Server

Secondary actor: Employer

Trigger: the app is loading

Included case: D1

Main flow: The employer opens the app then selects add InBody from the drop down menu after that the system registers the Inbody

Extensions: the system collapse

14th use case:

Use case name: Remove InBody

Related requirement: D1

Goal in context: The system removes the new InBody selected by employer

Pre-condition: Click on InBody menu

Successful end condition: The InBody is removed

Failed end condition: Couldn’t remove the InBody

Primary actor: Server

Secondary actor: Employer

Trigger: the app is loading

Included case: D1

Main flow: The employer opens the app then selects remove InBody from the drop down menu ,then write InBody's ID to the InBody's Information.

Extensions: the system collapse

15th use case:

Use case name: Search InBody

Related requirement: D1

Goal in context: The system searches for the InBody

Pre-condition: Click on search for an InBody

Successful end condition: The information of the InBody appears

Failed end condition: Couldn't search for the InBody

Primary actor: Server

Secondary actor: Employer

Trigger: the app is loading

Included case: C1

Main flow: The employer opens the app then selects search for an InBody from the drop down menu after that he writes the InBody 's ID then the system provides the InBody details

Extensions: the system collapse

16th use case:

Use case name: Open list of all InBodies

Related requirement: D1

Goal in context: The system shows list of all the InBodies

Pre-condition: Click on search for an InBody

Successful end condition: list of all the InBodies is opened

Failed end condition: Couldn't open the list of all the InBodies

Primary actor: Server

Secondary actor: Employer

Trigger: the app is loading

Included case: D1

Main flow: The employer opens the app then selects list of all InBodies from the drop down menu after that list of all the InBodies registered in the application appears.

Extensions: the system collapse

17th use case:

Use case name: Select Fitness class

Related requirement: E2

Goal in context: Apply Operations in Fitness Class menu

Pre-condition: Open the app

Successful end condition: the Fitness class menu opened so chose an option

Failed end condition: couldn't open the Fitness Class menu

Primary actor: Employer

Secondary actor: Server

Trigger: Decide which menu to click on

Included case: --

Main flow: The employer opens the app then decide which menu he want to click on then select Fitness class menu

Extensions: The application freezes.

18th use case:

Use case name: Add Fitness Class

Related requirement: E1

Goal in context: The system adds the Fitness Class selected by employer

Pre-condition: Click on add Fitness Class

Successful end condition: new Fitness Class is registered

Failed end condition: It doesn’t register the Fitness Class

Primary actor: Server

Secondary actor: Employer

Trigger: the app is loading

Included case: E1

Main flow: The employer opens the app then selects add Fitness Class from the drop down menu after that the system registers the Fitness Class

Extensions: the system collapse

19th use case:

Use case name: Remove Fitness Class

Related requirement: E1

Goal in context: The system removes the new Fitness Class selected by employer

Pre-condition: Click on Fitness Class menu

Successful end condition: The Fitness Class is removed

Failed end condition: Couldn’t remove the Fitness Class

Primary actor: Server

Secondary actor: Employer

Trigger: the app is loading

Included case: D1

Main flow: The employer opens the app then selects remove Fitness Class from the drop down menu ,then write Fitness Class’ ID then the system provides the fitness class information

Extensions: the system collapse

20th use case:

Use case name: Search Fitness Class

Related requirement: E1

Goal in context: The system searches for the Fitness Class

Pre-condition: Click on search for a fitness class

Successful end condition: The information of the fitness class appears

Failed end condition: Couldn't search for the fitness class

Primary actor: Server

Secondary actor: Employer

Trigger: the app is loading

Included case: E1

Main flow: The employer opens the app then selects search for fitness class from the drop down menu after that he writes the fitness class' ID then the system provides the fitness class details

Extensions: the system collapse

21st use case:

Use case name: Open list of all fitness classes

Related requirement: E1

Goal in context: The system shows list of all the Fitness Classes

Pre-condition: Click on search for a fitness class

Successful end condition: list of all the fitness classes is opened

Failed end condition: Couldn't open the list of all the Fitness Classes

Primary actor: Server

Secondary actor: Employer

Trigger: the app is loading

Included case: E1

Main flow: The employer opens the app then selects list of all Fitness classes from the drop down menu after that list of all the Fitness classes registered in the application appears.

Extensions: the system collapse

6. Requirements validation

Functional requirements:

- Adding, removing and searching for a member.
- Adding, removing and searching for an employee.
- Adding and removing and searching for an InBody.
- Adding, removing and searching for fitness class.
- Checking the lists of all members, employees, Inbodies and fitness classes.

Requirements Traceability Matrix

Req. ID	1	2	3	4	5
1					
2					
3	✓				
4	✓				
5	✓	✓	✓	✓	

8. Class Model

Stage 1:

Menu lists on top of the window of System of the gym control the selection of what we want to do, either with members, employees, fitness classes or inbodies.

Stage 2:

Menu lists on top of the window of System of the gym control the selection of what we want to do, either with members, employees, fitness classes or inbodies. Each menu list drop menu items when clicked to give us the facilities provided for the pressed menu list. Member menu list helps us add member and view list of members. Employee menu list helps us add employee and view list of employees. Fitness class menu list helps us add fitness class and view list of fitness class. And finally, Inbody menu list help us add inbody and view list of inbodies.

Each menu list closes when something is selected from it and a new window of the selected item opens. If nothing is clicked, the window's shape stays the way it has opened.

Stage 3:

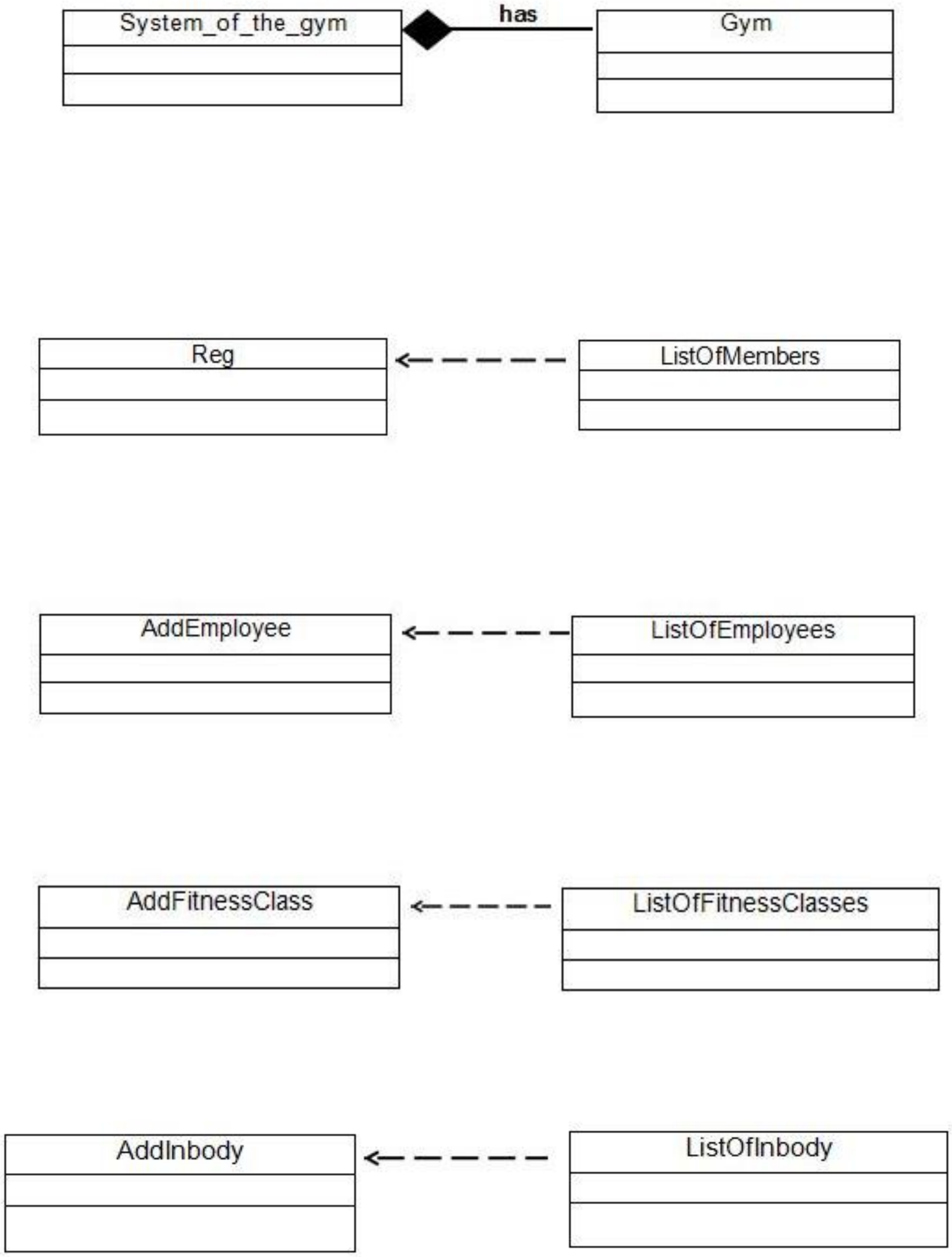
The nouns that can be identified are: **menu lists, top of the window, System of the gym, selection, menu items, clicked, facilities, add member, list of members, add employee, list of employees, add fitness class, list of fitness classes, add inbody, list of inbodies, selected, new window, shape.**

Exclude nouns that lie outside the problem boundary: **top of the window, selection, facilities, selected**.

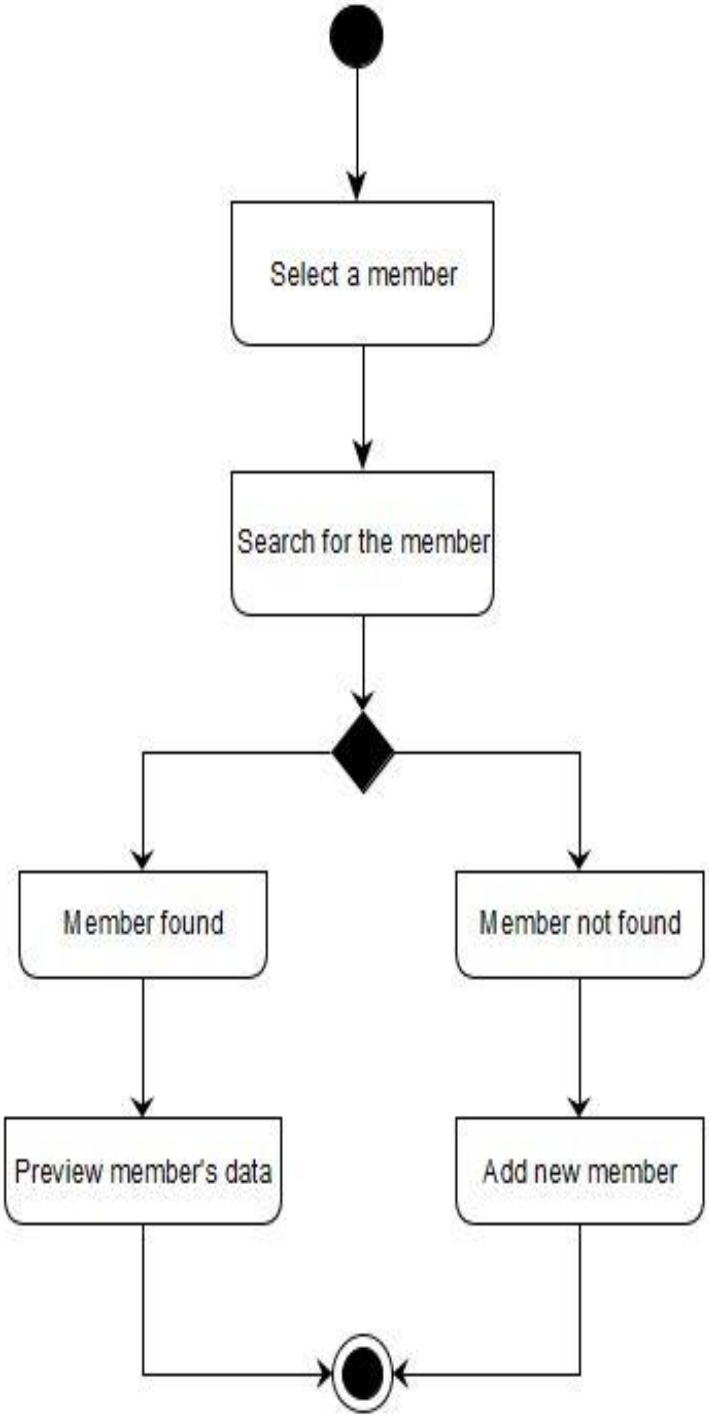
Exclude abstract nouns that represent ideas or quantities that have no physical existence, these may become attributes: **menu lists, menu items, clicked, new window, shape.**

Hence, candidate classes are: **System of the gym, gym, add member, list of members, add employee, list of employees, add fitness class, list of fitness classes, add Inbody, list of Inbodies.**

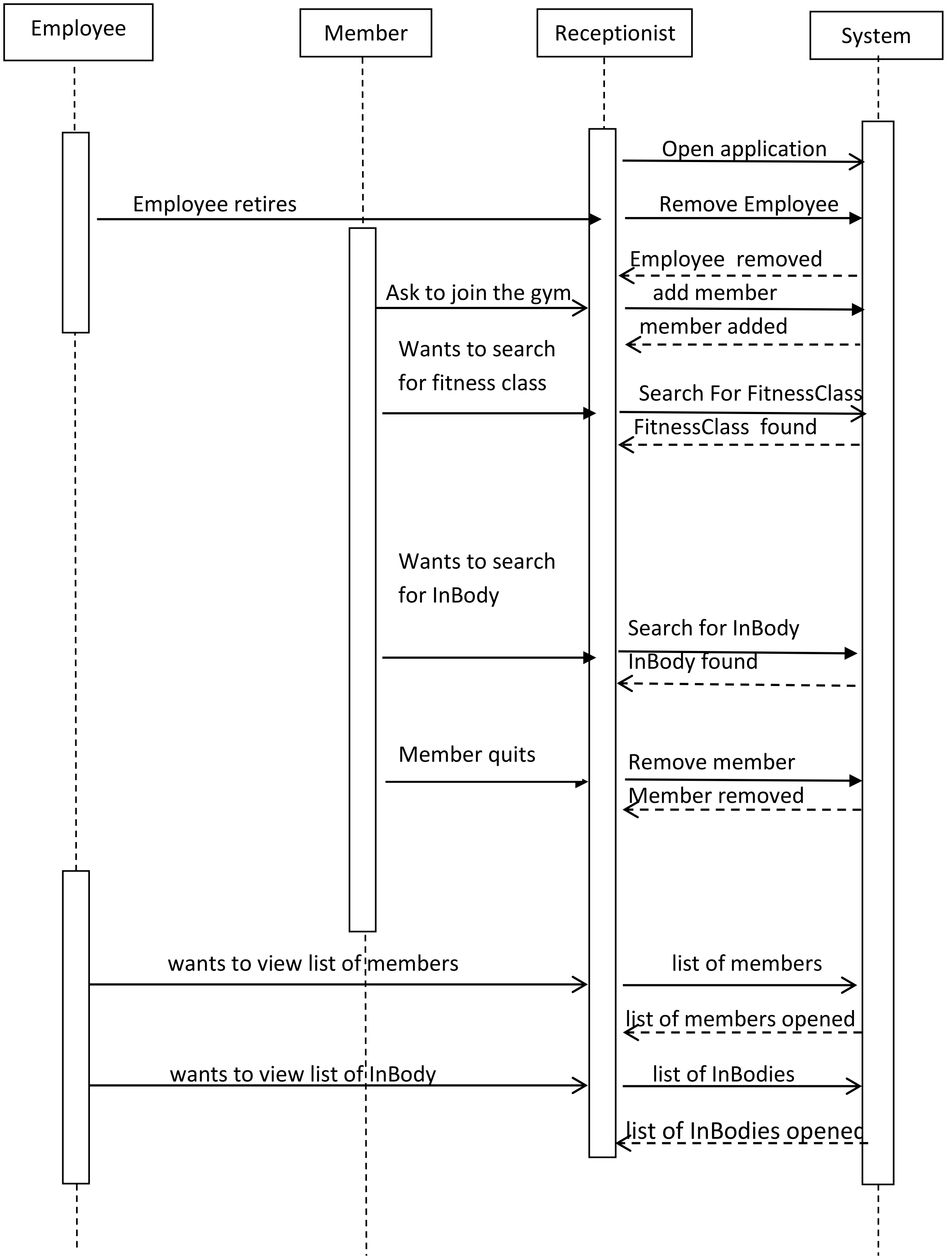
Class Diagram



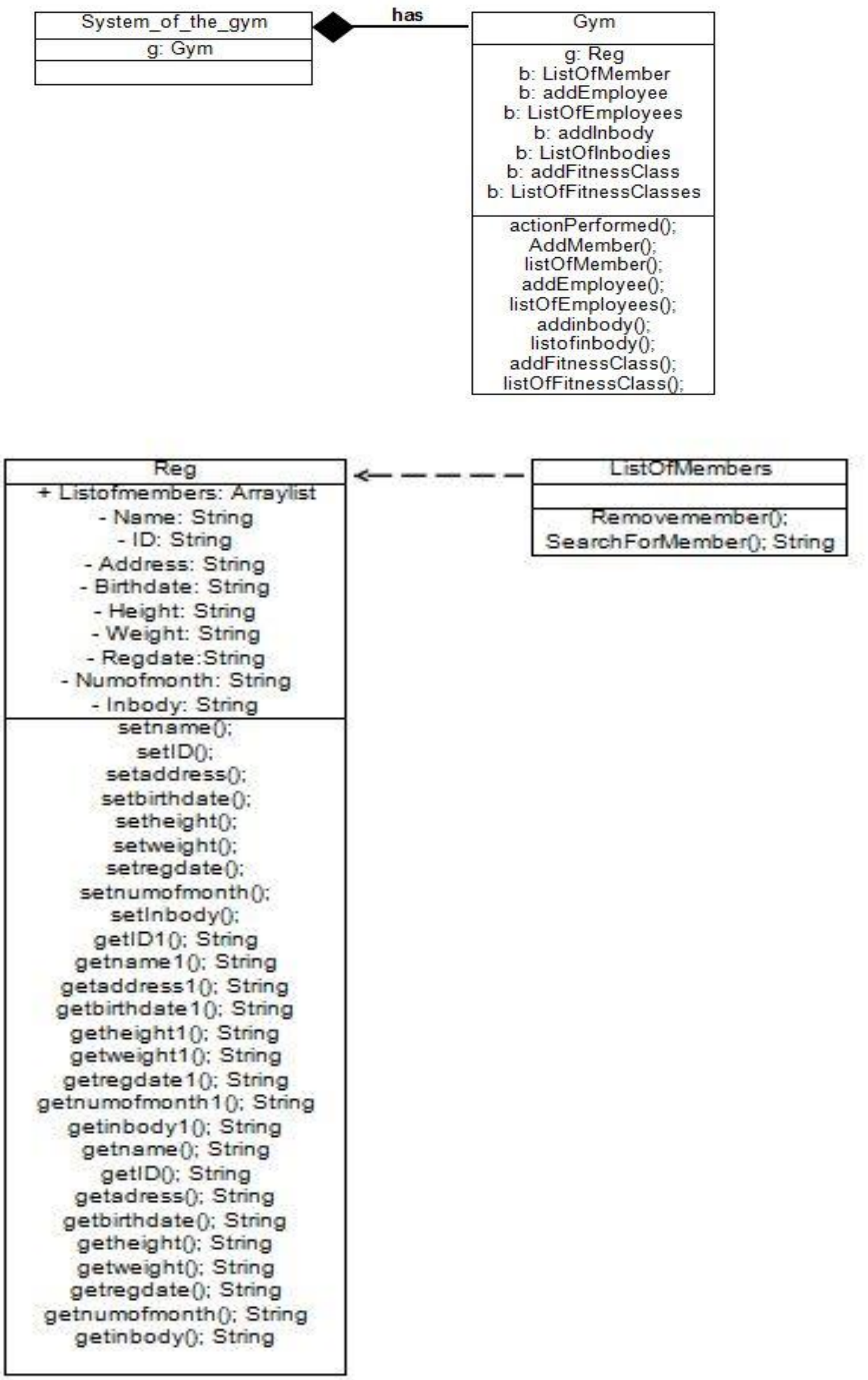
9. State diagram

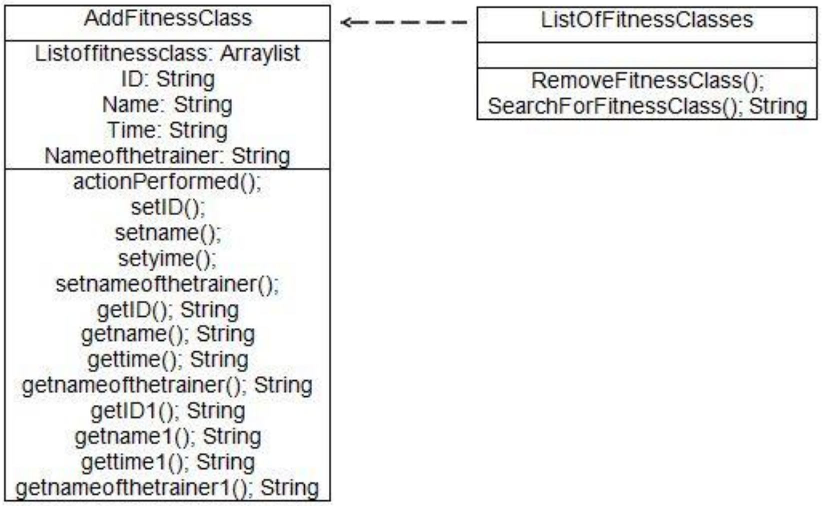


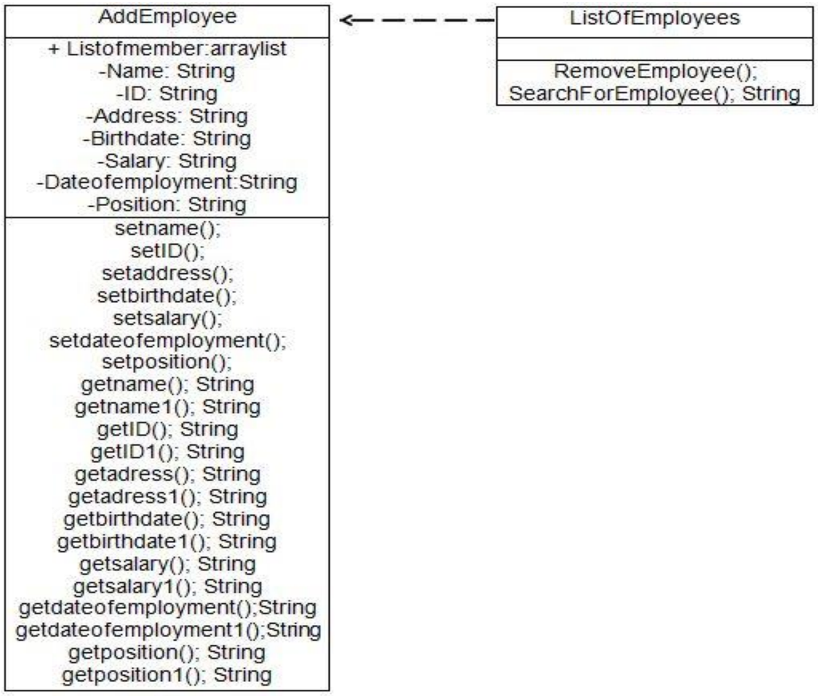
10. Interaction (Sequence) diagram

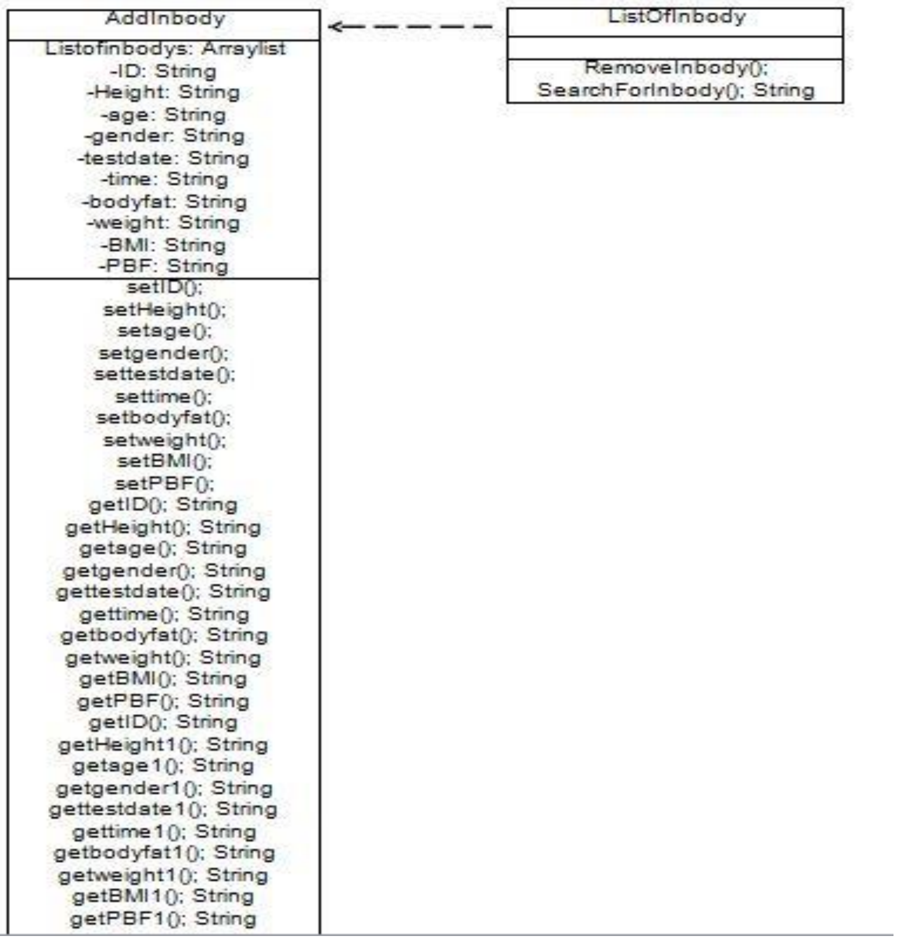


11. Detailed class diagram

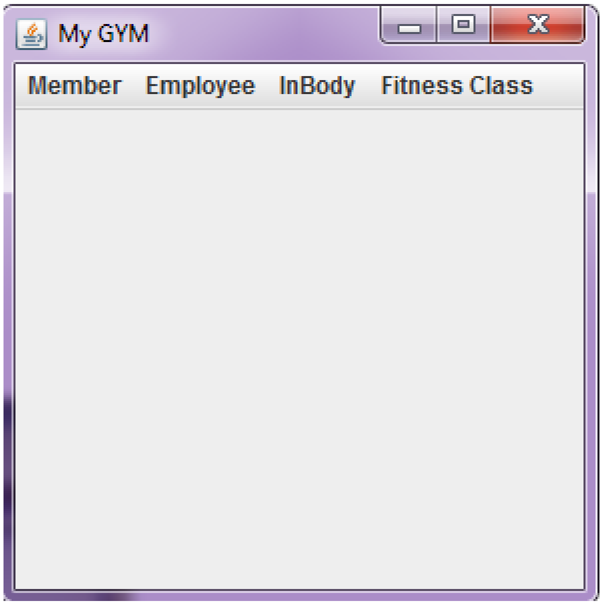



AddFitnessClass
Listoffitnessclass: Arraylist
ID: String
Name: String
Time: String
Nameofthetrainer: String
actionPerformed();
setID();
setname();
setyime();
setnameofthetrainer();
getID(); String
getname(); String
gettime(); String
getnameofthetrainer(); String
getID1(); String
getname1(); String
gettime1(); String
getnameofthetrainer1(); String
ListOfFitnessClasses
RemoveFitnessClass();
SearchForFitnessClass(); String


AddEmployee
+ Listofmember:arraylist
-Name: String
-ID: String
-Address: String
-Birthdate: String
-Salary: String
-Dateofemployment:String
-Position: String
setname();
setID();
setaddress();
setbirthdate();
setsalary();
setdateofemployment();
setposition();
getname(); String
getname1(); String
getID(); String
getID1(); String
getadress(); String
getadress1(); String
getbirthdate(); String
getbirthdate1(); String
getsalary(); String
getsalary1(); String
getdateofemployment();String
getdateofemployment1();String
getposition(); String
getposition1(); String
ListOfEmployees
RemoveEmployee();
SearchForEmployee(); String


AddInbody
Listofinbodys: Arraylist
-ID: String
-Height: String
-age: String
-gender: String
-testdate: String
-time: String
-bodyfat: String
-weight: String
-BMI: String
-PBF: String
setID();
setHeight();
setage();
setgender();
settestdate();
settime();
setbodyfat();
setweight();
setBMI();
setPBF();
getID(); String
getHeight(); String
getage(); String
getgender(); String
gettestdate(); String
gettime(); String
getbodyfat(); String
getweight(); String
getBMI(); String
getPBF(); String
getID(); String
getHeight1(); String
getage1(); String
getgender1(); String
gettestdate1(); String
gettime1(); String
getbodyfat1(); String
getweight1(); String
getBMI1(); String
getPBF1(); String
ListOfInbody
RemoveInbody();
SearchForInbody(); String

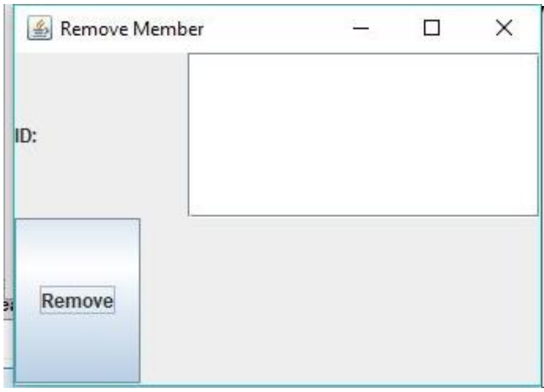
11. User Interface Design:

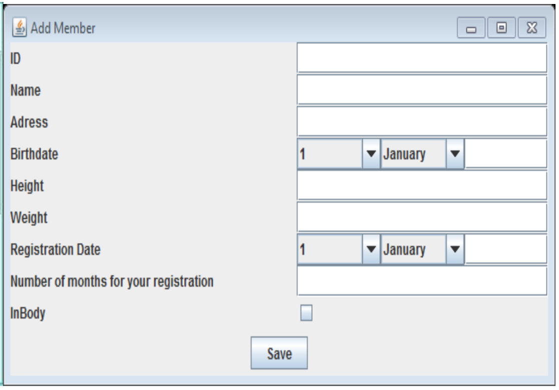


System home page

Purpose: The starting point of the My GYM system.

User Interaction: The user should choose from the four options above what he wants to deal with.





ID

Name

Adress

Birthdate 1 January

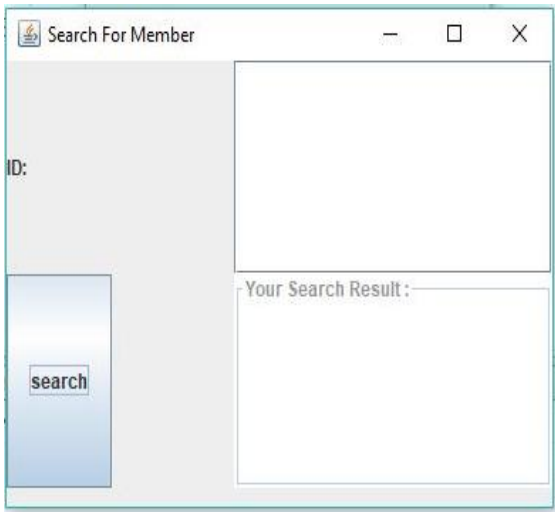
Height

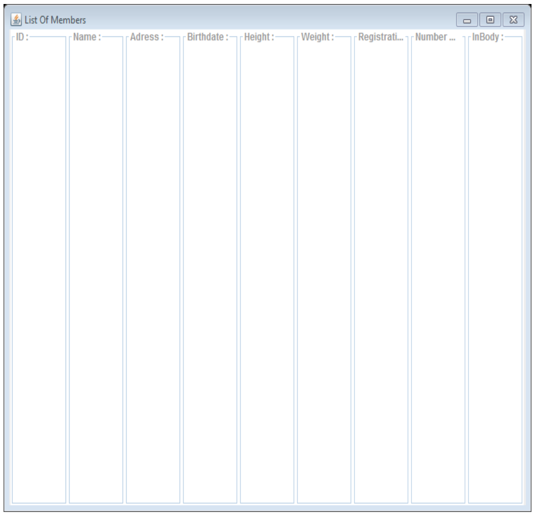
Weight

Registration Date 1 January

Number of months for your registration

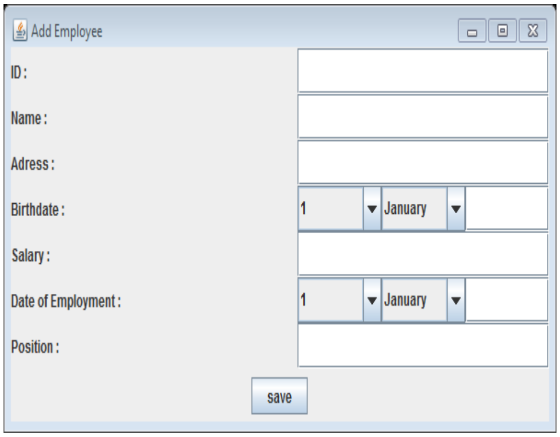
InBody □





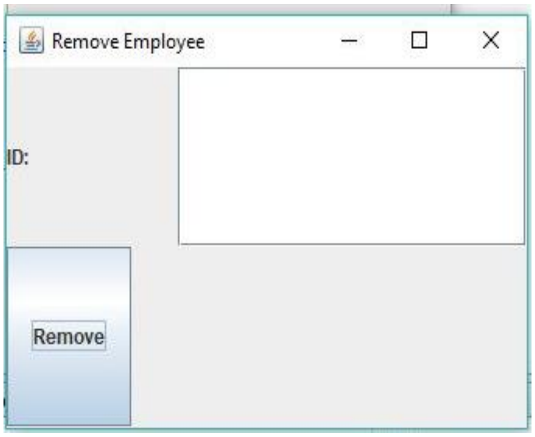
Purpose : The user can add new member to the gym, remove an old one, search for one, and open a list of all members enrolled in the gym

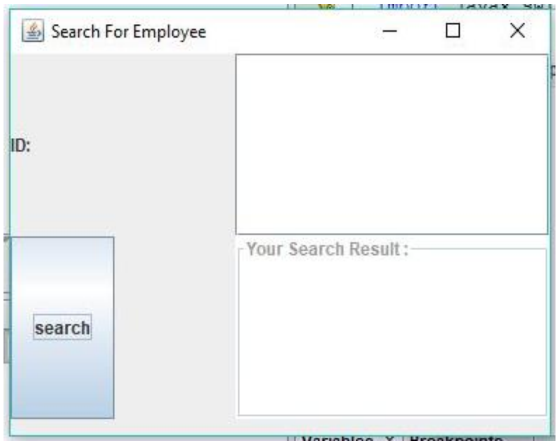
User Interaction : The user should fill the data of the member and add/remove/search for him.

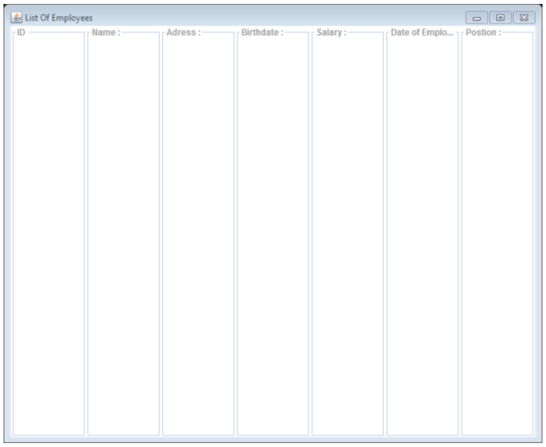


Birthdate : 1 January

Date of Employment : 1 January

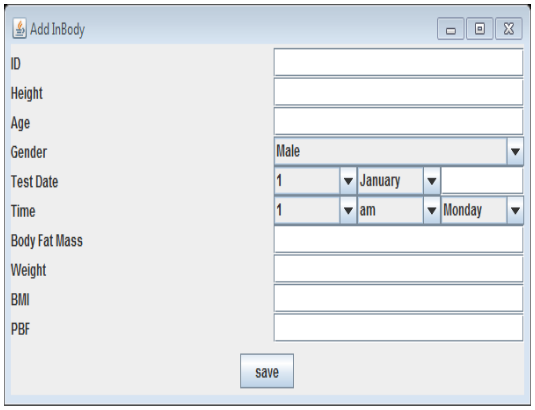




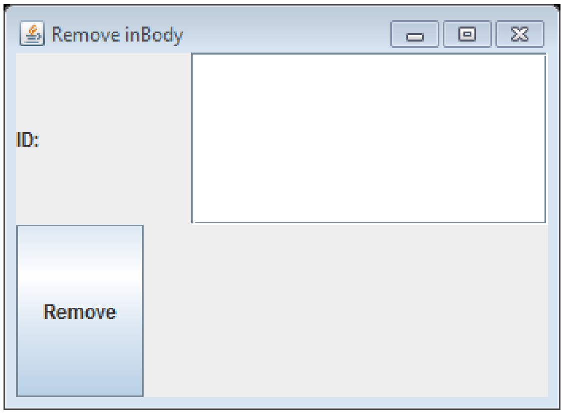


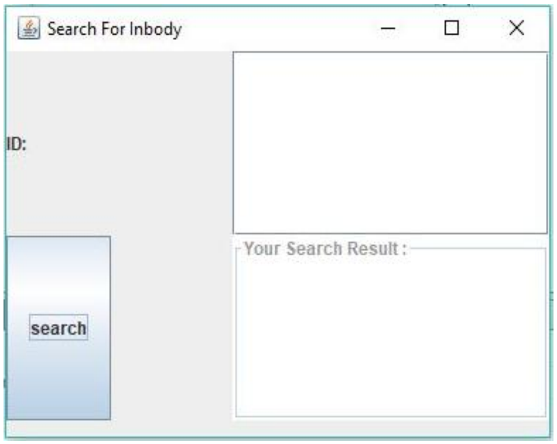
Purpose : The user can add new employee, remove an old one, search for one, and open a list of all employees working in the gym.

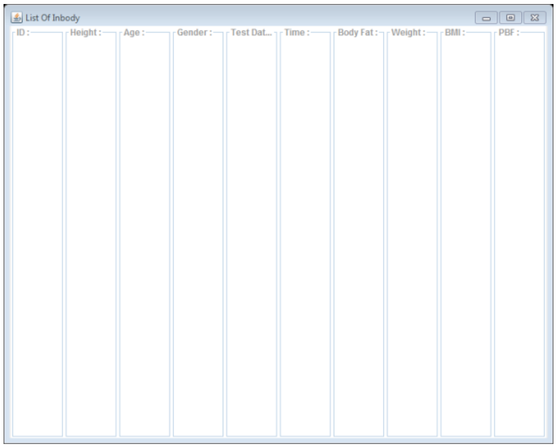
User Interaction : The user should fill the data of the employee and add/remove/search for him.



ID
Height
Age
Gender Male
Test Date 1 January
Time 1 am Monday
Body Fat Mass
Weight
BMI
PBF

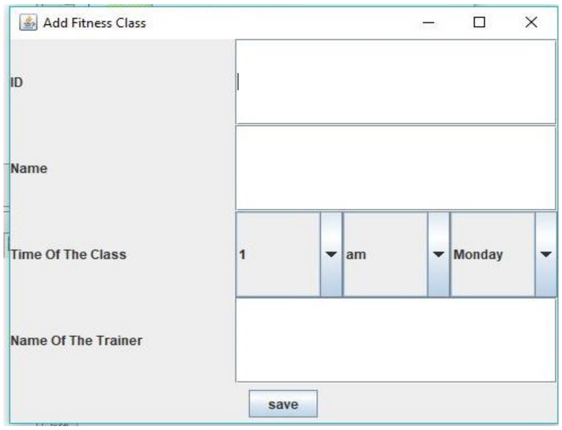


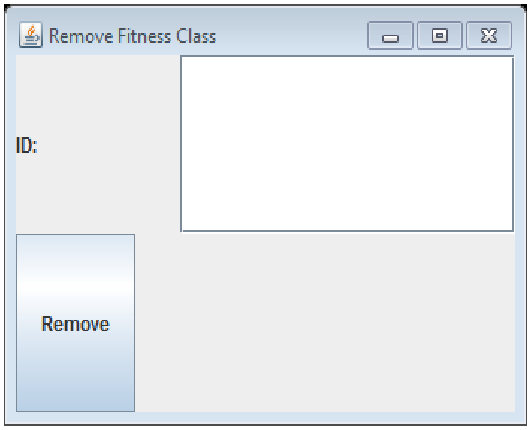


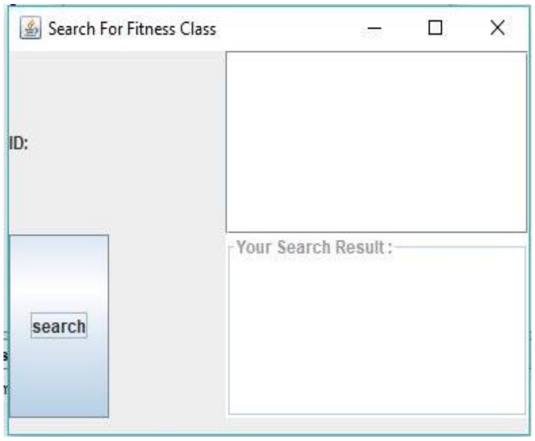


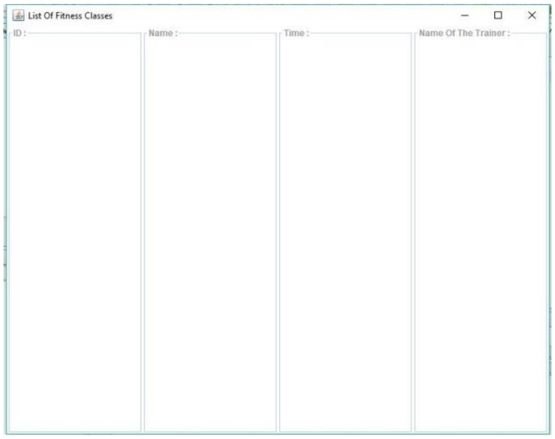
Purpose : The user can add inbody for a member, remove an old one, search for one, and open a list of all inbody for all members enrolled in the gym.

User Interaction : The user should fill the data of the member and add/remove/search for his info.





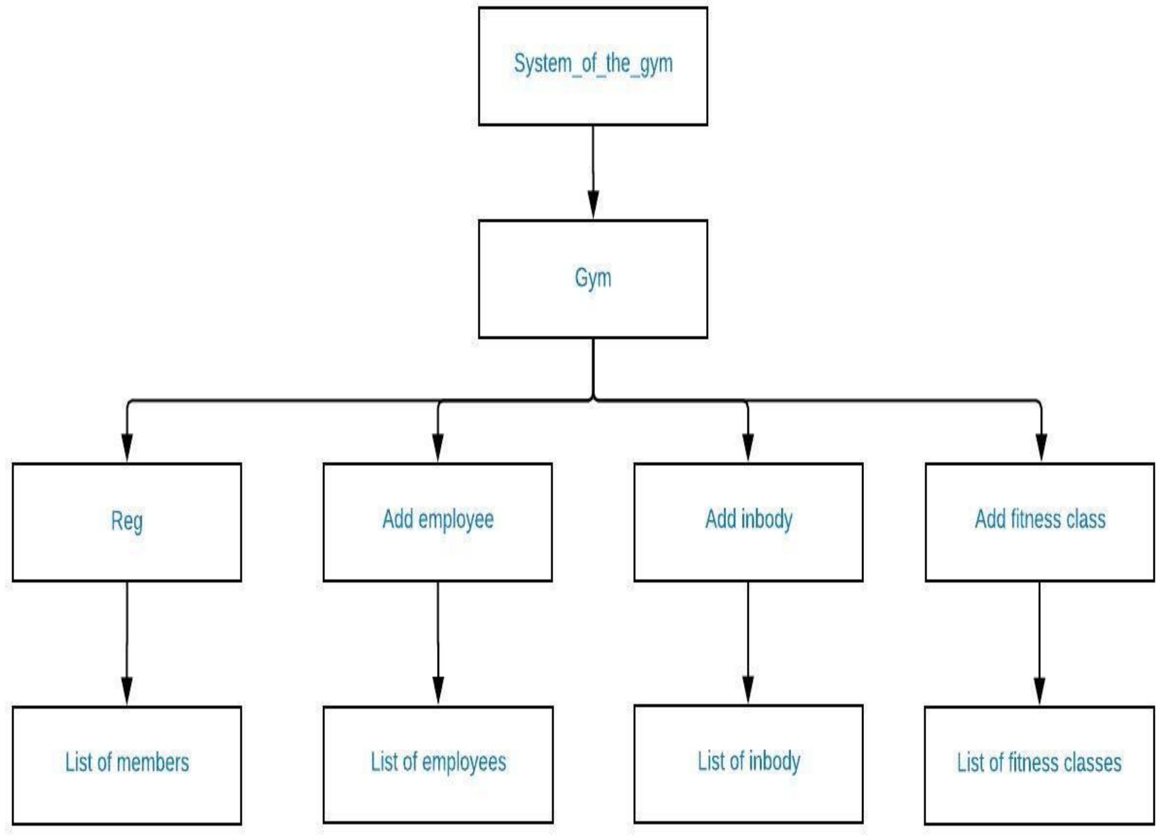




Purpose : The user can add a fitness class for a member, remove an old one, search for one, and open a list of all fitness classes available in the gym.

User Interaction : The user should fill the data of the fitness class and add/ remove/ search for a fitness class.

12. Client-Object Relation Diagram:



Client-Object relation diagram

13. Detailed Design:

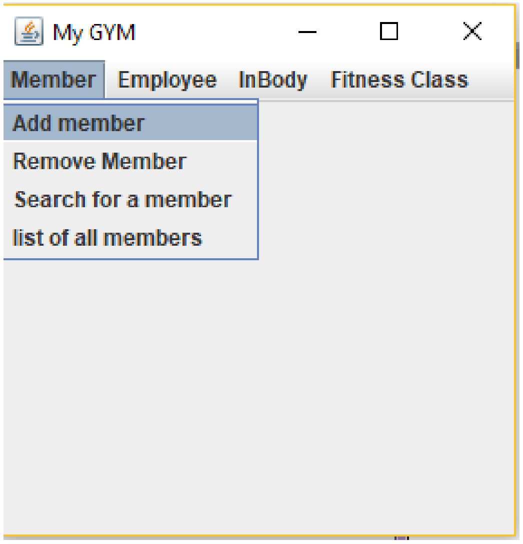
```
Void system home page (void)
{
   While(True)
   {
      If (Member button is clicked)
      {
         The user should choose to add, remove, search, or open a list of all members;
```

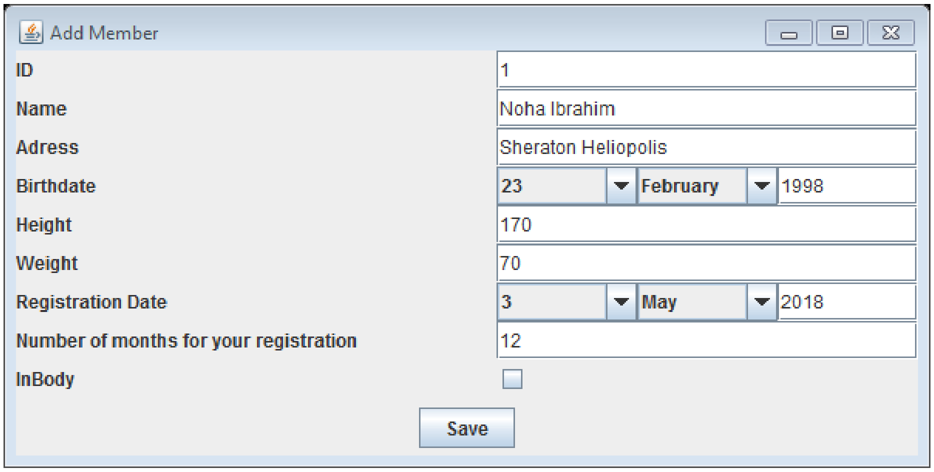
```
    If (add new member is chosen)
    {
        Registration window opens;
        The user should start the new member’s id, name and other info then click save;
        The new member should be added to the array list of members;
    }
    If (remove member is chosen)
    {
        Remove member window opens;
        The user should enter the Id of the member he wants to remove then click remove;
        The member should be removed from the array list of members;
    }
    If (Search for a member is chosen)
    {
        A window should appear asking the user to enter the Id of the member;
        The user should enter the Id and click search;
        All the member’s info should appear in a text box;
    }
    If (List of members is chosen)
    {
        A window should appear that has a list of all members and their info;
    }
}
If (Employee button is clicked)
{
    The user should choose to add, remove, search, or open a list of all employees;
    If (add new employee is chosen)
    {
        New employee window opens;
        The user should enter the new employee’s id, name and other info then click save;
        The new employee should be added to the array list of employees;
    }
    If (remove employee is chosen)
    {
        Remove employee window opens;
        The user should enter the Id of the employee he wants to remove then click remove;
        The employee should be removed from the array list of members;
    }
    If (Search for an employee is chosen)
    {
        A window should appear asking the user to enter the Id of the employee;
        The user should enter the Id and click search;
        All the employee’s info should appear in a text box;
    }
    If (List of employees is chosen)
    {
        A window should appear that has a list of all employees  and their info;
    }
}


If (Inbody button is clicked)
{
    The user should choose to add, remove, search, or open a list of inbody;
    If (add new inbody is chosen)
    {
        inbody window opens;
        The user should enter the member’s id and inbody info  then click save;
        The member’s inbody should be added to the array list of inbody;
    }
    If (remove inbody is chosen)
    {
        Remove inbody window opens;
        The user should enter the Id of the member he wants to remove his inbody then click remove;
        The inbody should be removed from the array list of inbody;
    }
    If (Search for inbody is chosen)
    {
        A window should appear asking the user to enter the Id of the member;
        The user should enter the Id and click search;
        All the member’s inbody info should appear in a text box;
    }
    If (List of inbody is chosen)
    {
        A window should appear that has a list of all members’ ids and their inbody info;
    }
}
If (Fitness class button is clicked)
{
```

```
            The user should choose to add, remove, search, or open a list of fitness classes;
            If (add new fitness class is chosen)
            {
               Fitness class window opens;
               The user should enter info about the fitness class and the trainer then click save;
               The new fitness class should be added to the array list of fitness classes;
            }
            If (remove fitness class is chosen)
            {
               Remove fitness class window opens;
               The user should enter the id of the fitness class he wants to remove then click remove;
               The fitness class should be removed from the array list of fitness classes;
            }
            If (Search for a fitness class is chosen)
            {
               A window should appear asking the user to enter the Id of the fitness class;
               The user should enter the Id and click search;
               All the fitness class’s info should appear in a text box;
            }
            If (List of fitness classes is chosen)
   {
   A window should appear that has a list of all fitness classes and their info;
            }
          }
        }
```

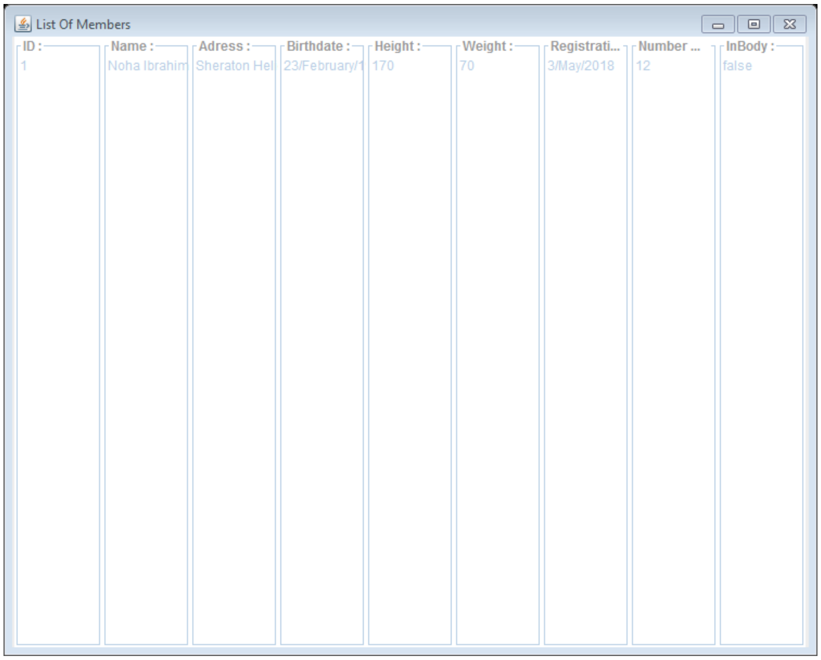
14. Testing:

i. Add new member:



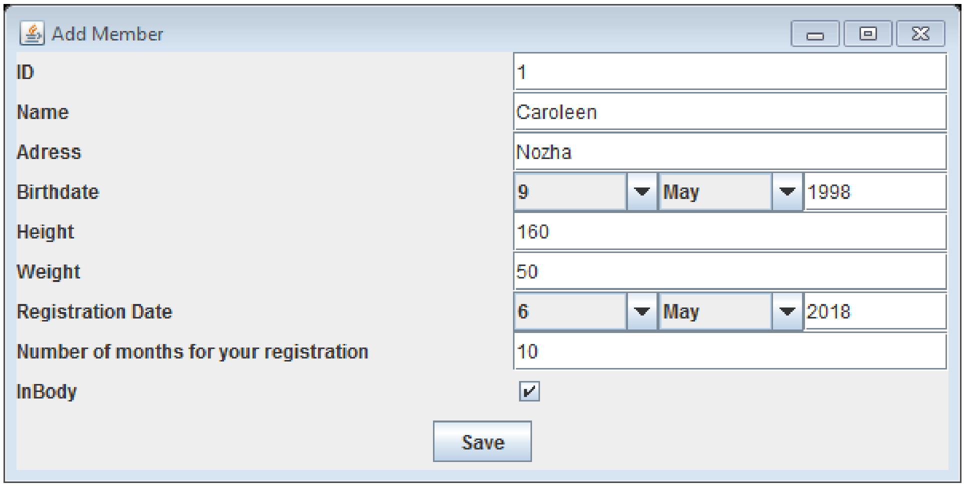

Add Member

ID	1		
Name	Noha Ibrahim		
Adress	Sheraton Heliopolis		
Birthdate	23	February	1998
Height	170		
Weight	70		
Registration Date	3	May	2018
Number of months for your registration	12		
InBody	□		

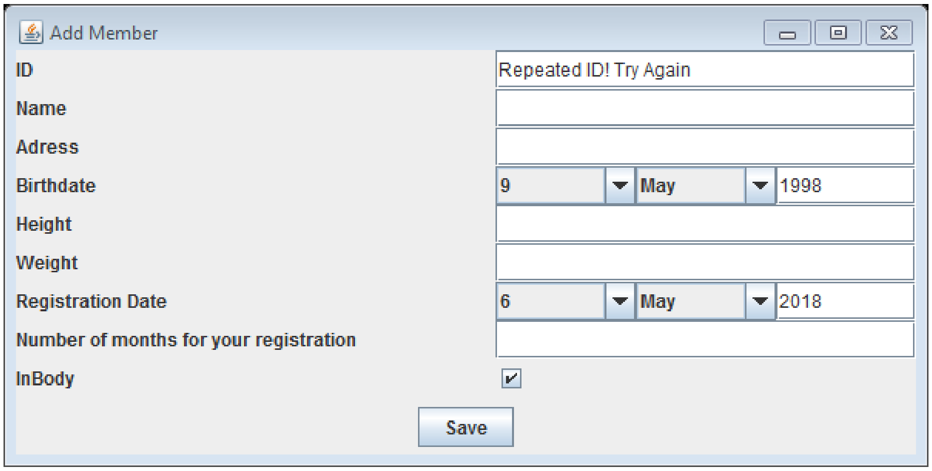


ID :	Name :	Adress :	Birthdate :	Height :	Weight :	Registrati...	Number ...	InBody :
1	Noha Ibrahim	Sheraton Hel	23/February/1	170	70	3/May/2018	12	false

ii. Add new member with an existing id:

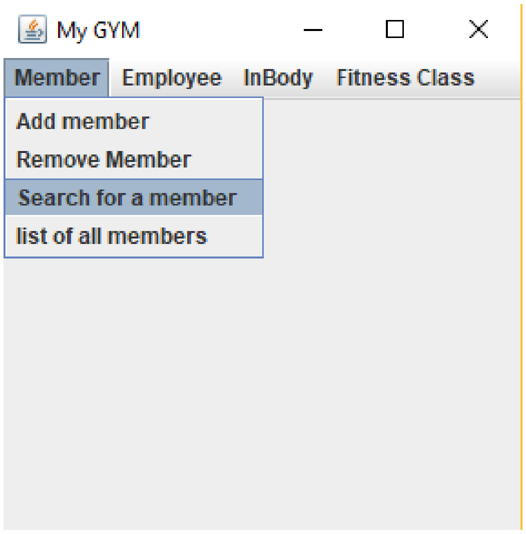


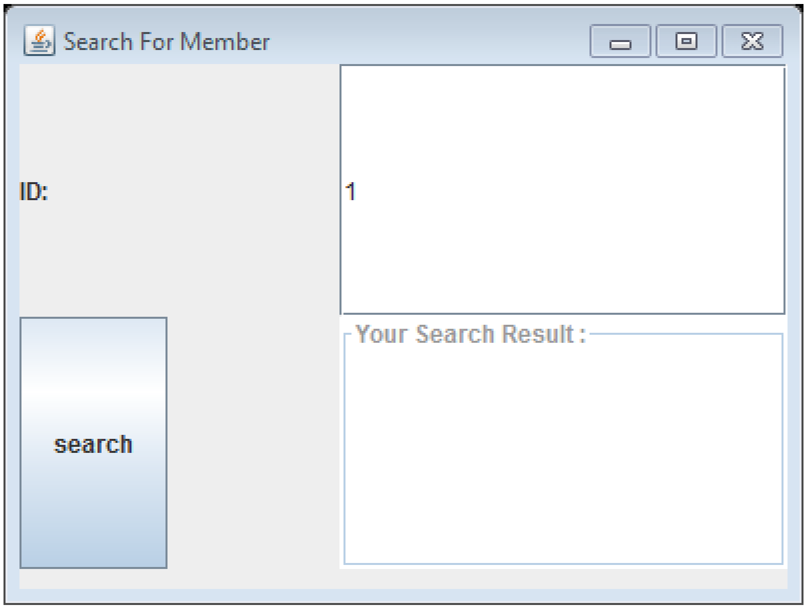
ID	1
Name	Caroleen
Adress	Nozha
Birthdate	9 May 1998
Height	160
Weight	50
Registration Date	6 May 2018
Number of months for your registration	10
InBody	☑



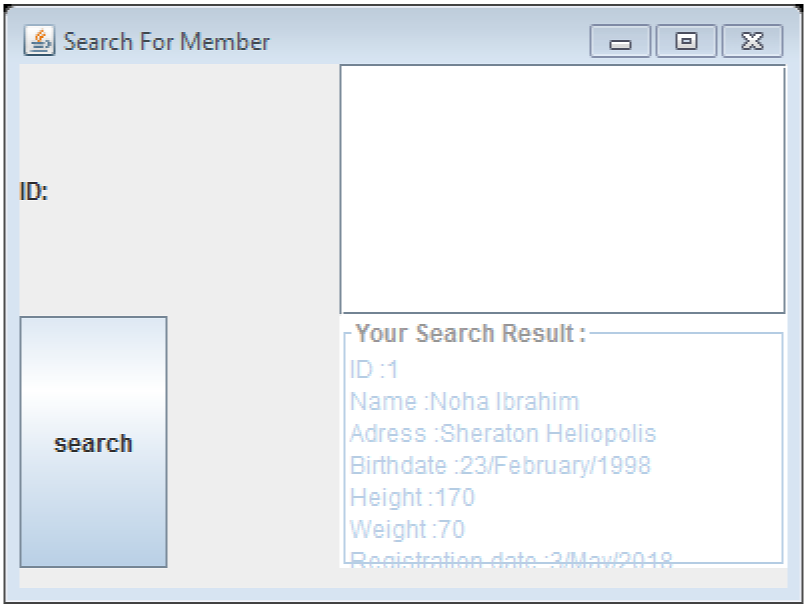
ID	Repeated ID! Try Again
Name	
Adress	
Birthdate	9 May 1998
Height	
Weight	
Registration Date	6 May 2018
Number of months for your registration	
InBody	☑

iii. Search for a member





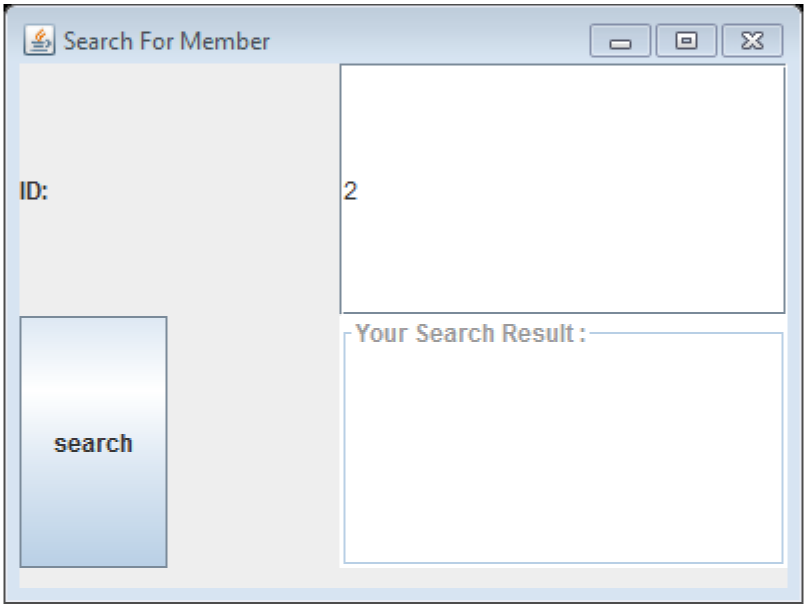
ID: 1



Your Search Result :

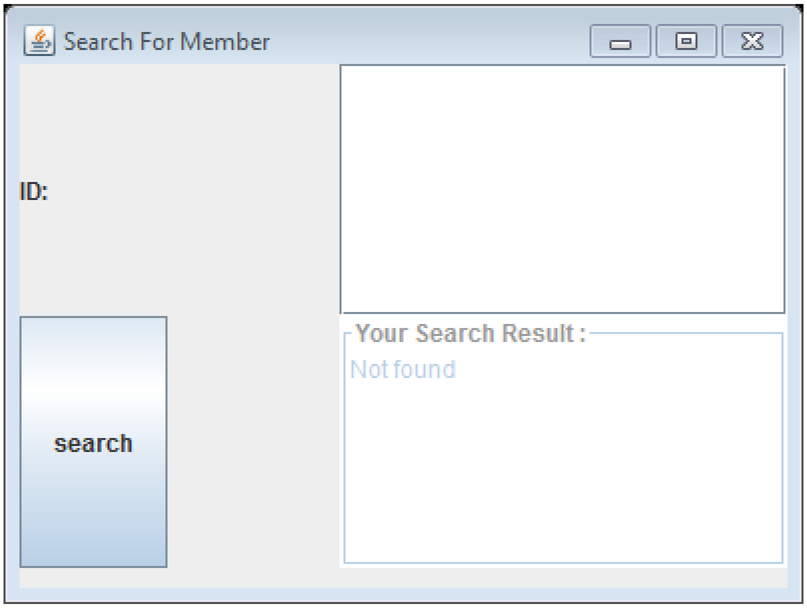
```
ID :1
Name :Noha Ibrahim
Adress :Sheraton Heliopolis
Birthdate :23/February/1998
Height :170
Weight :70
Registration date :3/May/2018
```

iv. Search for a member that is not on the list:

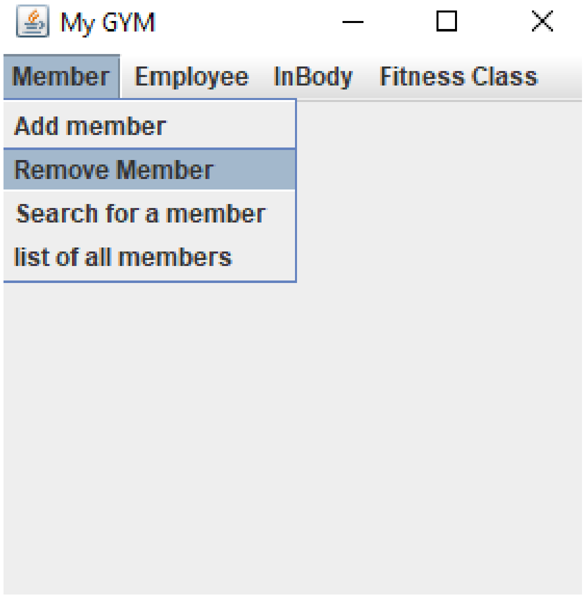


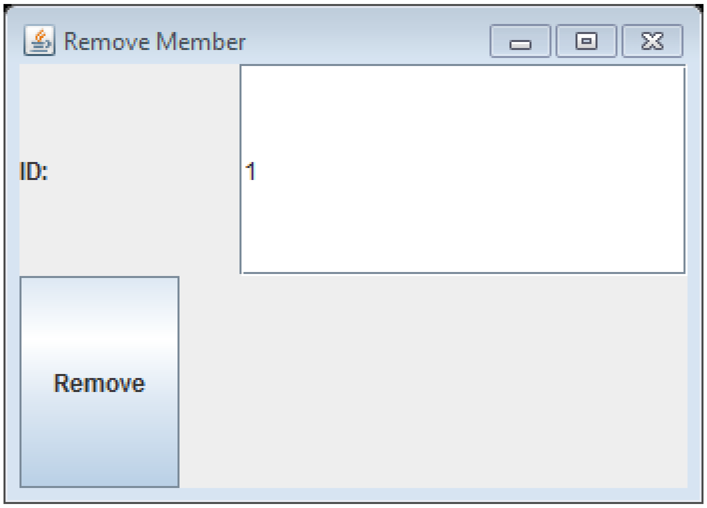
ID: 2

Your Search Result :

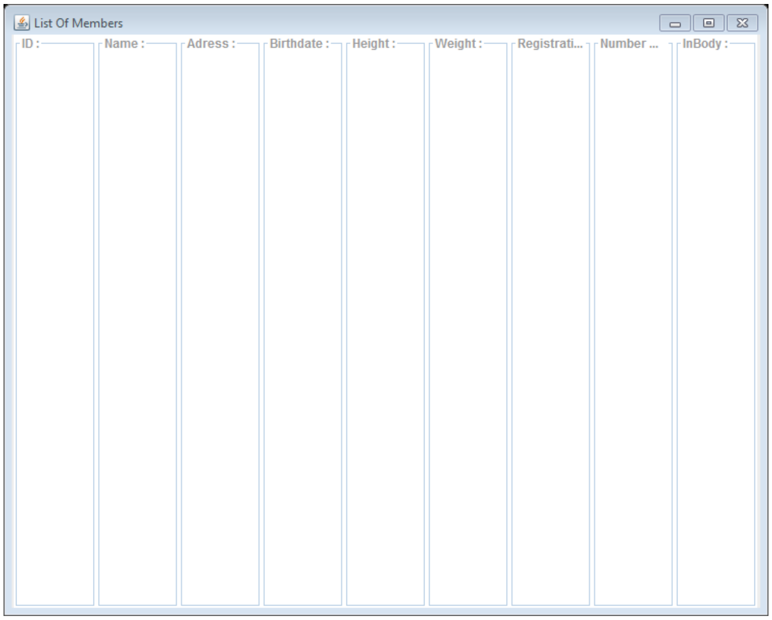


v. Remove member:



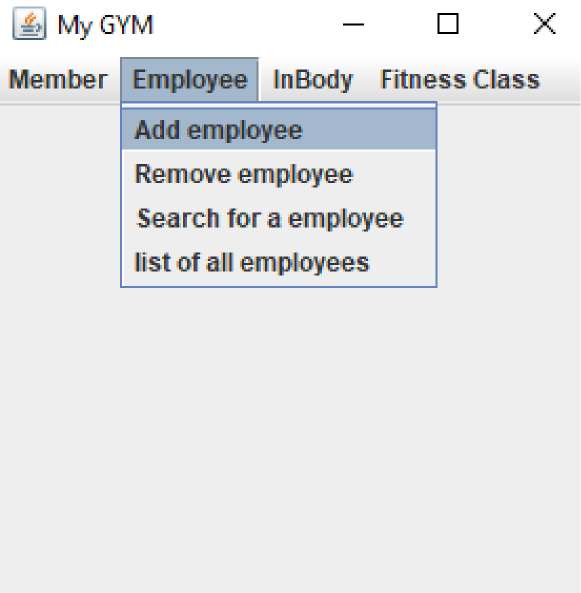

Remove Member

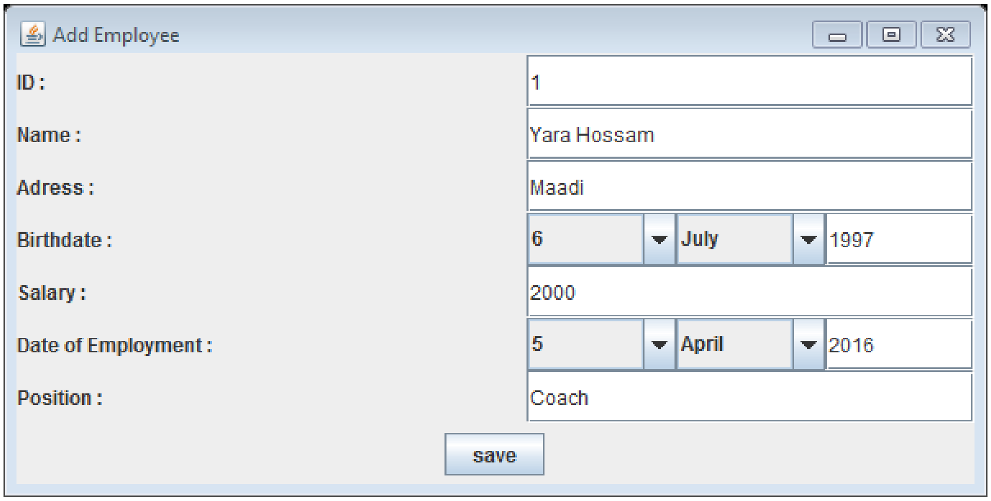
ID: 1


List Of Members

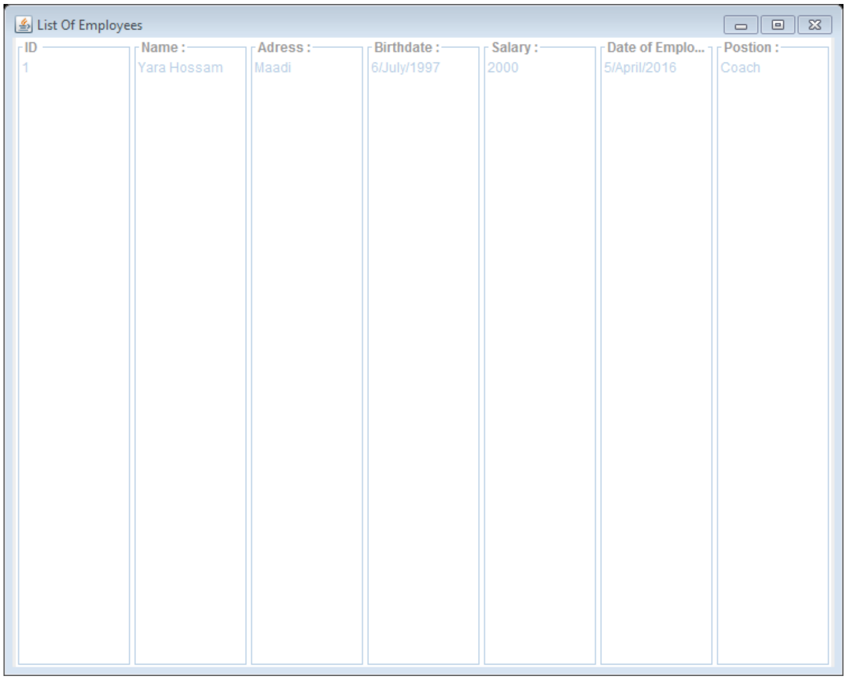
ID :	Name :	Adress :	Birthdate :	Height :	Weight :	Registrati...	Number ...	InBody :

vi. Add employee:





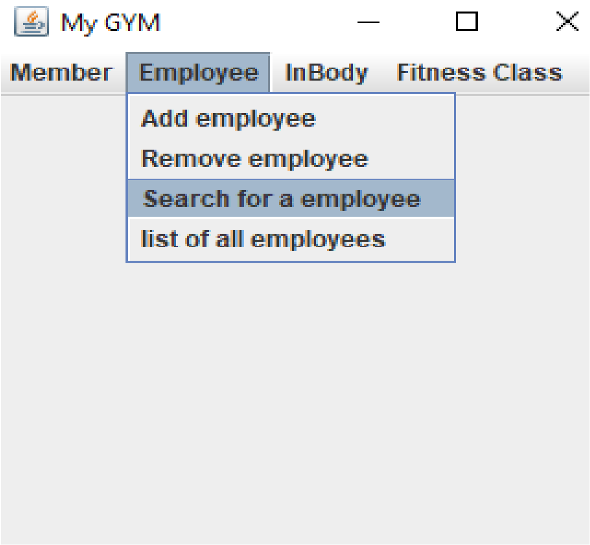
ID :	1
Name :	Yara Hossam
Adress :	Maadi
Birthdate :	6 July 1997
Salary :	2000
Date of Employment :	5 April 2016
Position :	Coach

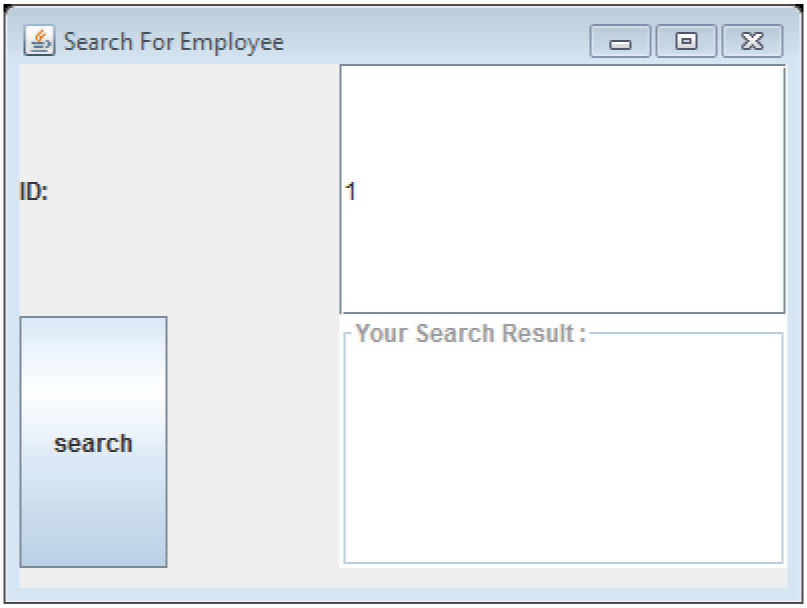


List Of Employees

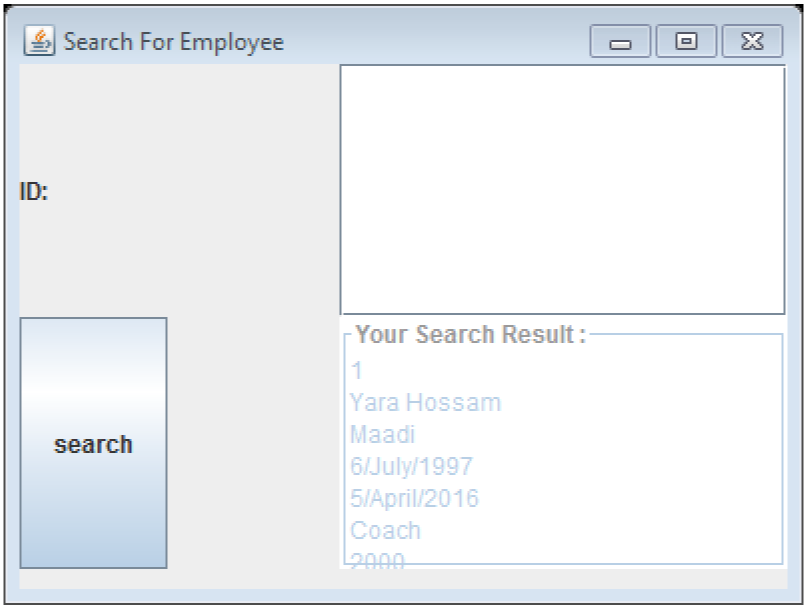
ID	Name :	Adress :	Birthdate :	Salary :	Date of Emplo...	Postion :
1	Yara Hossam	Maadi	6/July/1997	2000	5/April/2016	Coach

vii. Search for employee:



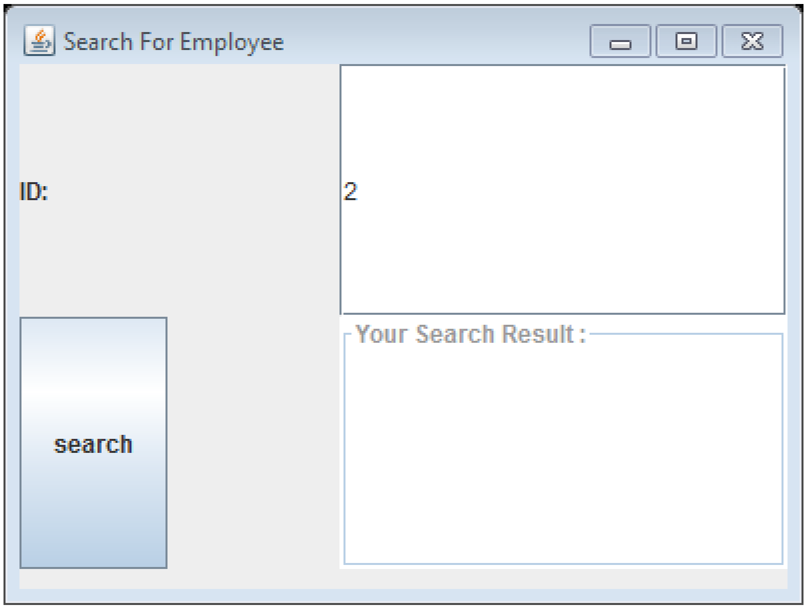


1

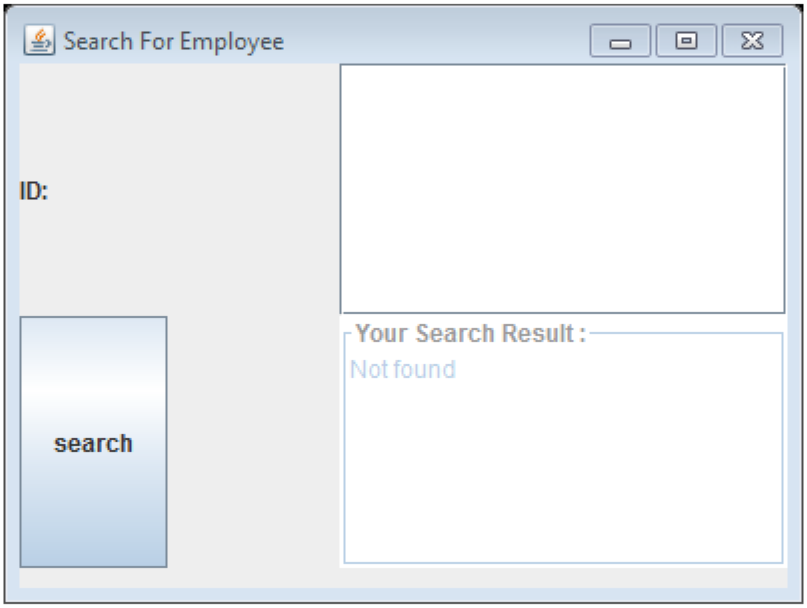


```
1
Yara Hossam
Maadi
6/July/1997
5/April/2016
Coach
2000
```

viii. Search for an employee that is not on the list:



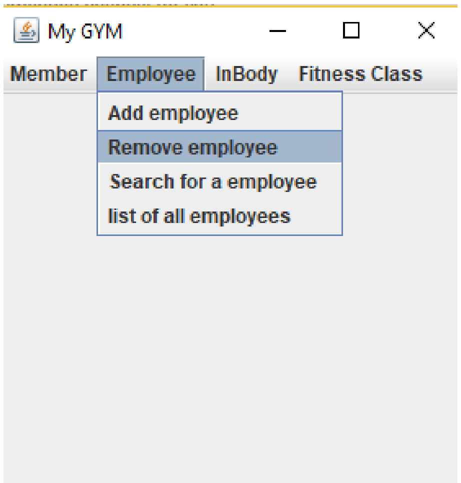
ID: 2

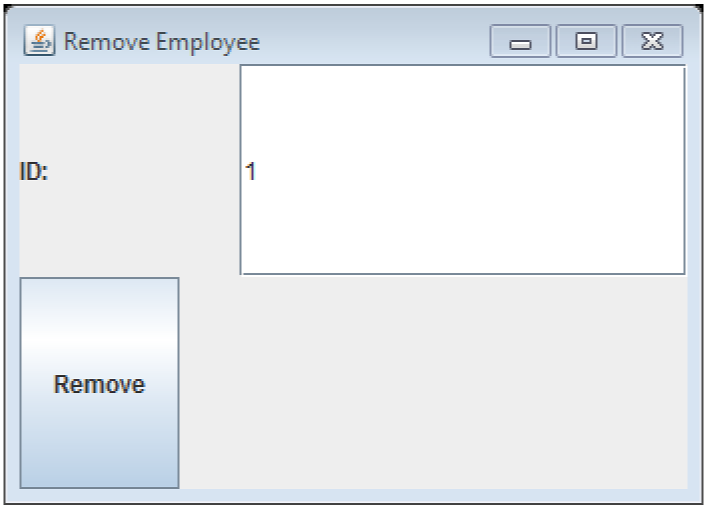


ID:

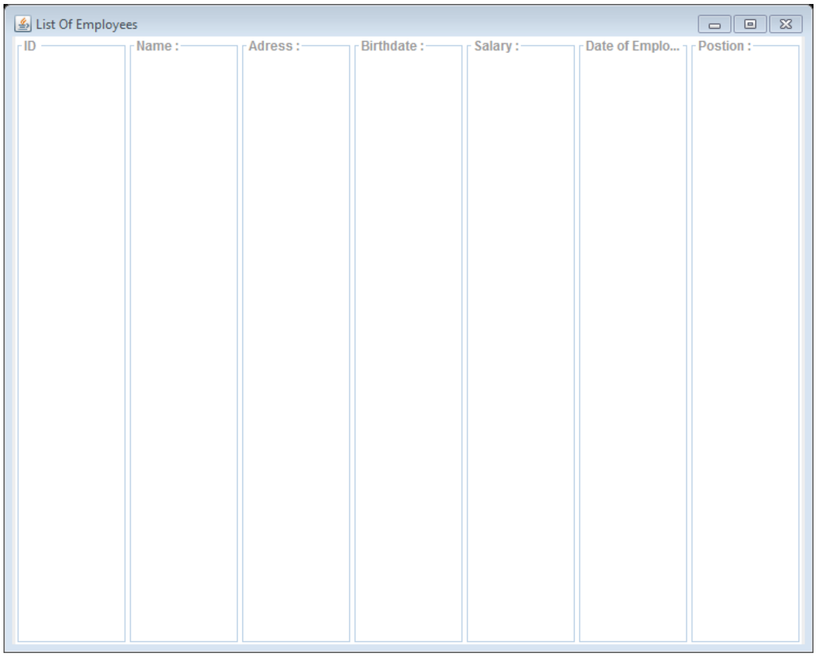
Not found

ix. Remove employee

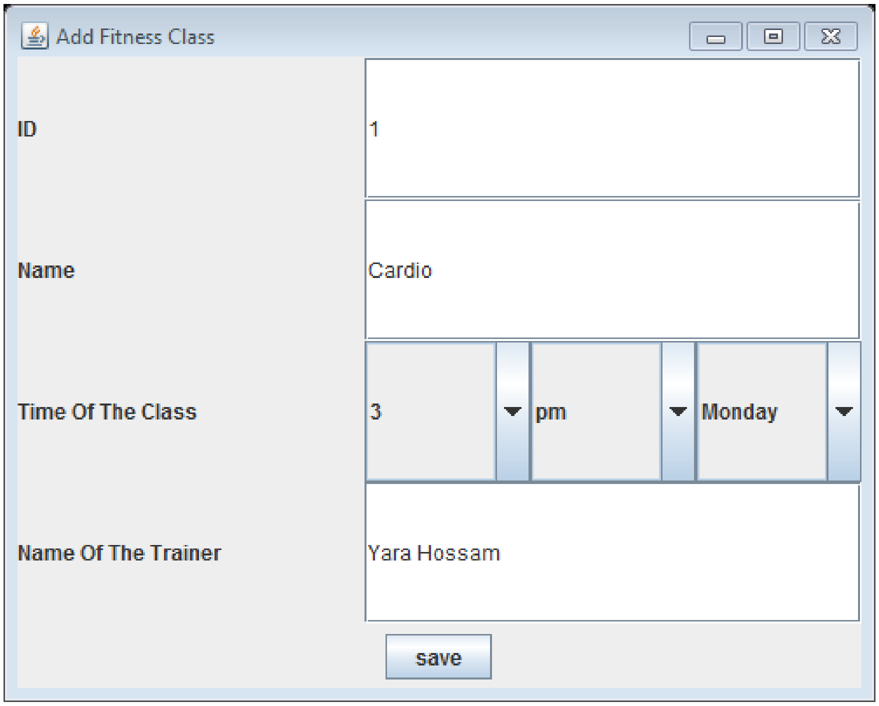




ID: 1

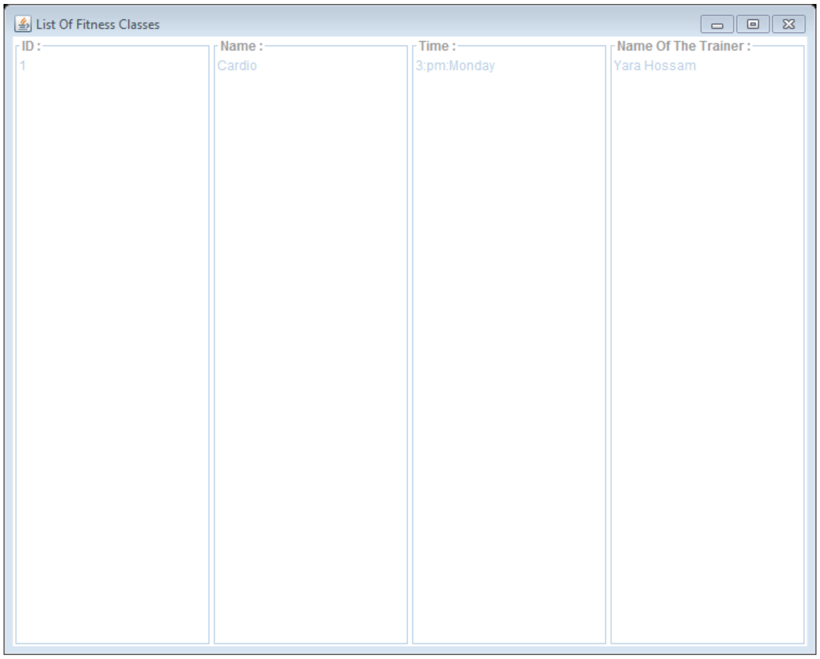


x. Add fitness class:



Add Fitness Class

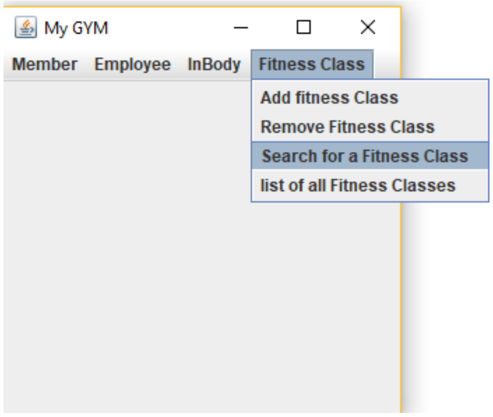
ID	1
Name	Cardio
Time Of The Class	3 pm Monday
Name Of The Trainer	Yara Hossam

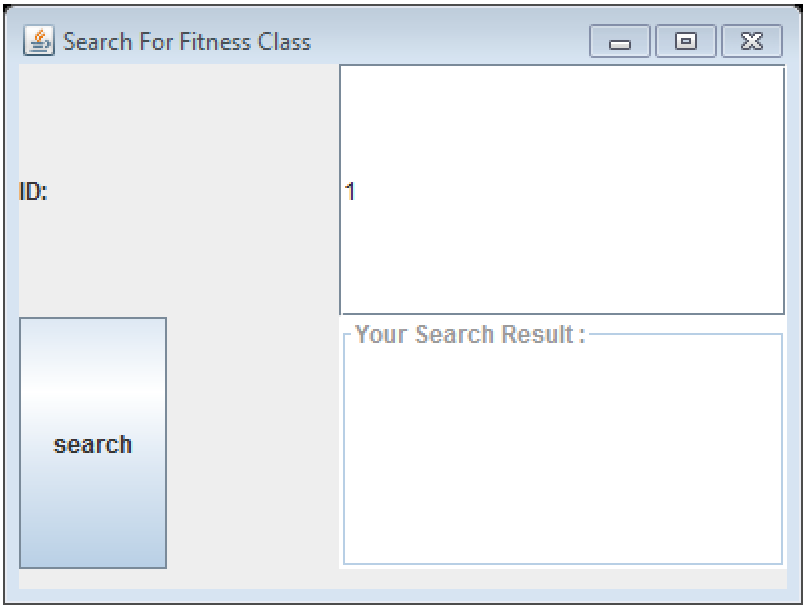


List Of Fitness Classes

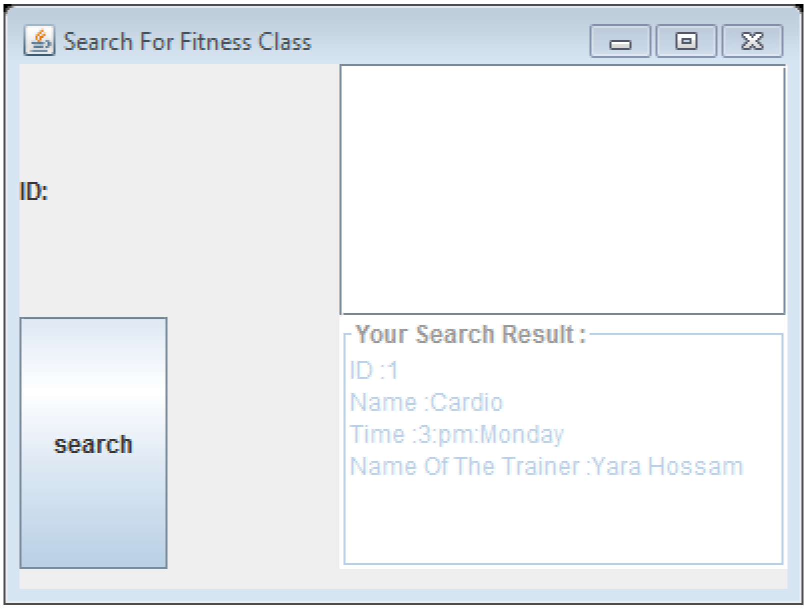
ID :	Name :	Time :	Name Of The Trainer :
1	Cardio	3:pm:Monday	Yara Hossam

xi. Search for a fitness class:



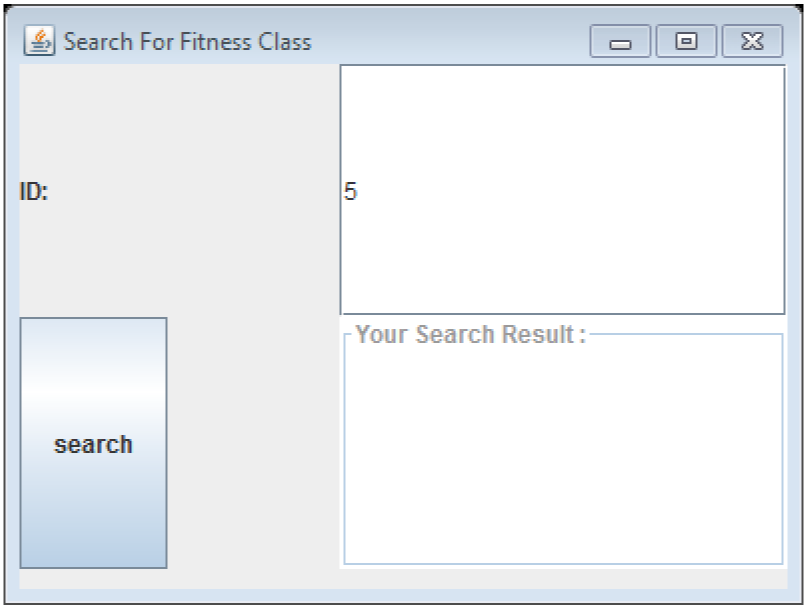


ID: 1

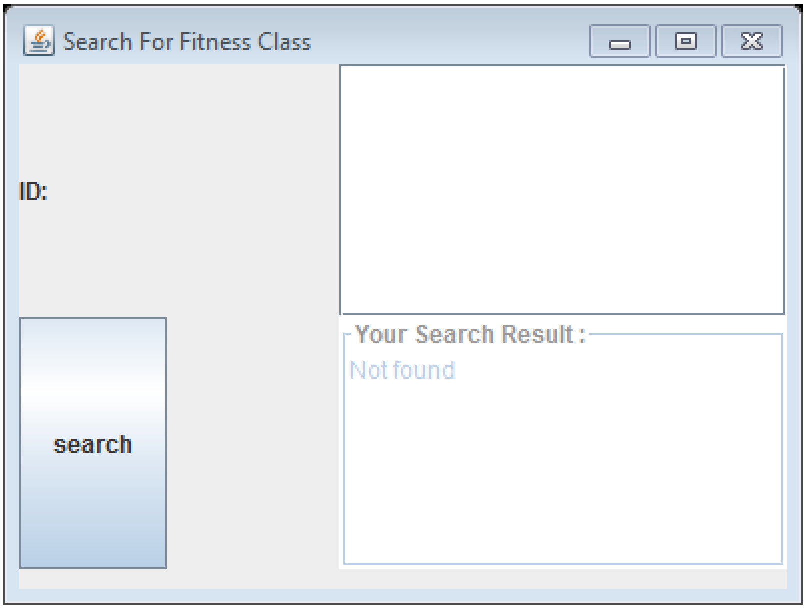


ID :1
Name :Cardio
Time :3:pm:Monday
Name Of The Trainer :Yara Hossam

xii. Search for a fitness class that is not on the list:

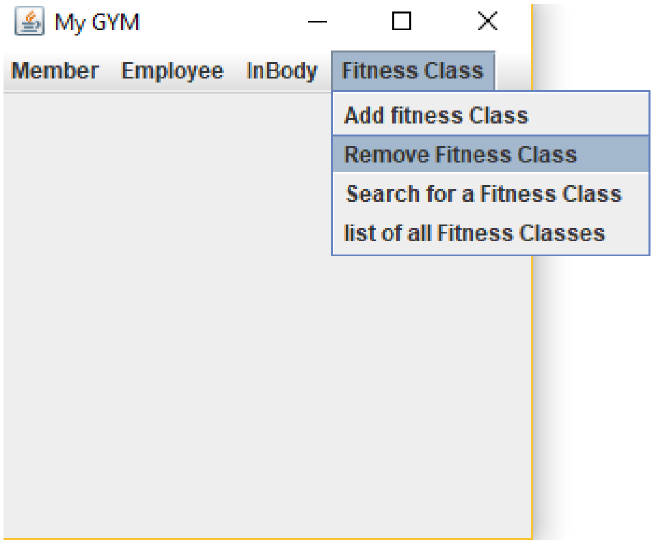


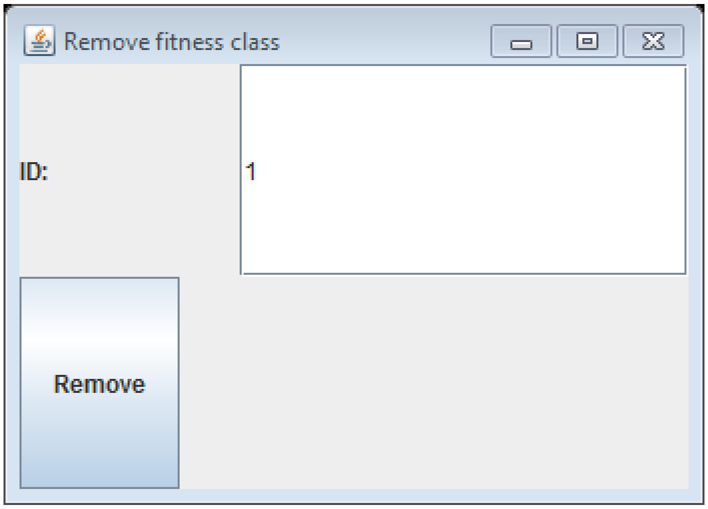
ID: 5



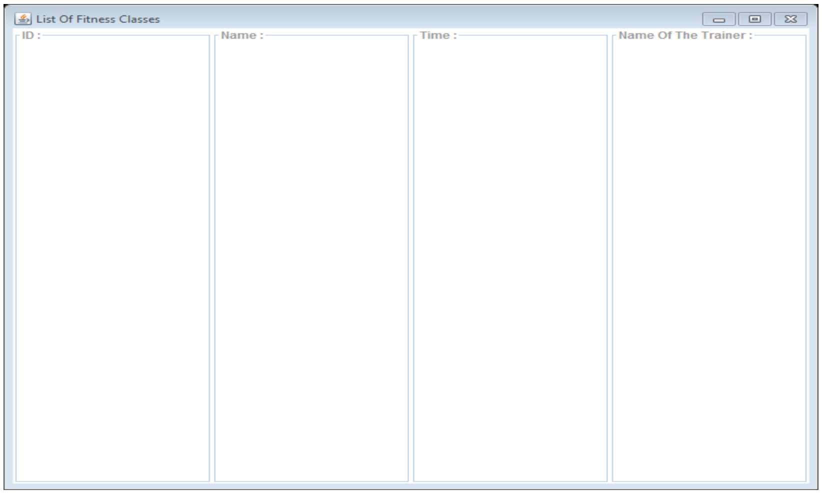
Your Search Result :
Not found

xiii. Remove fitness class:

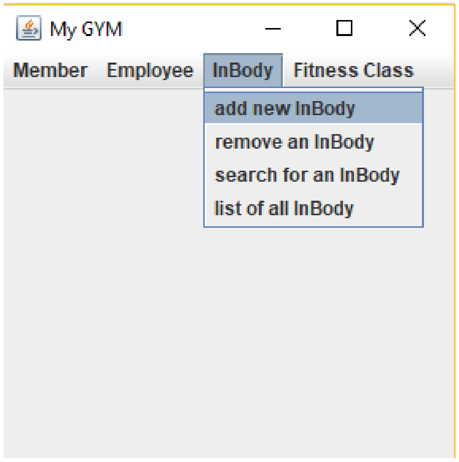


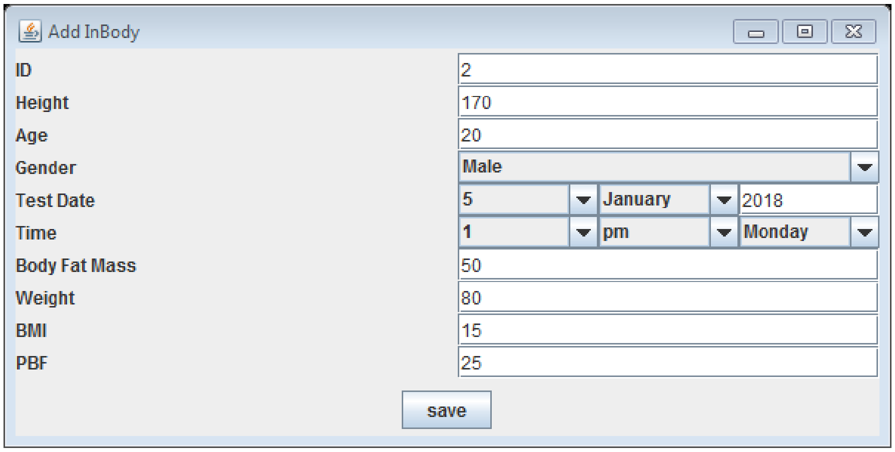


ID: 1

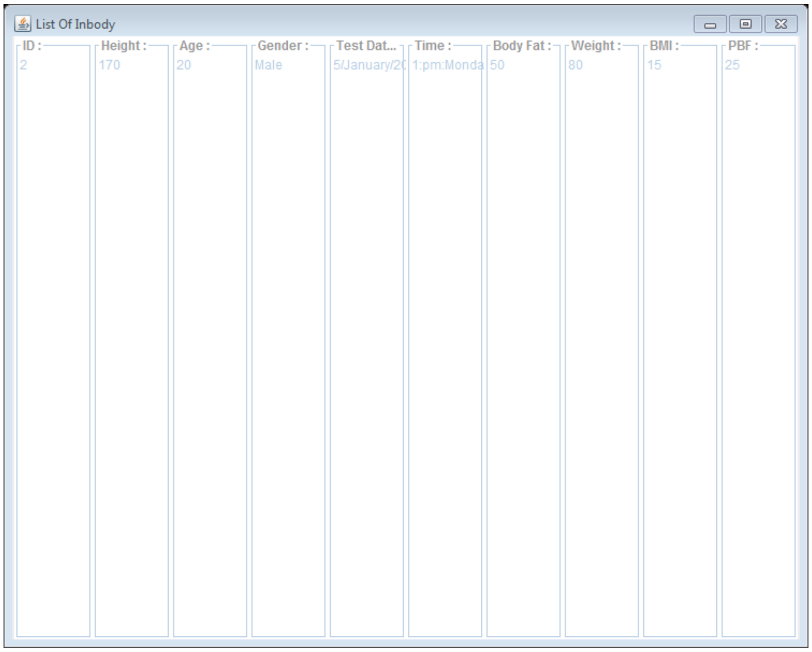


xiv. Add InBody:



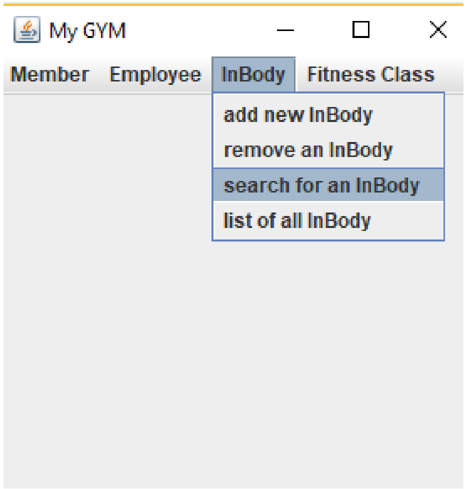

Add InBody

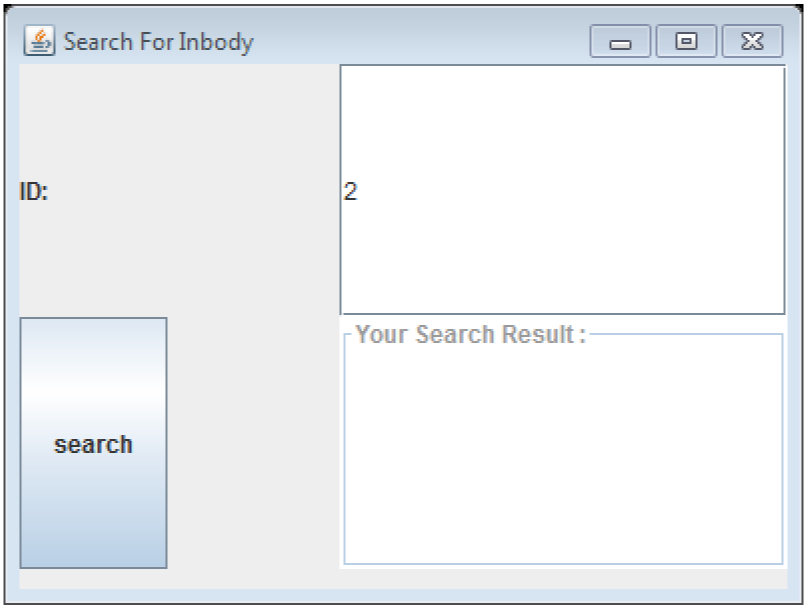
ID	2
Height	170
Age	20
Gender	Male
Test Date	5 January 2018
Time	1 pm Monday
Body Fat Mass	50
Weight	80
BMI	15
PBF	25


List Of Inbody

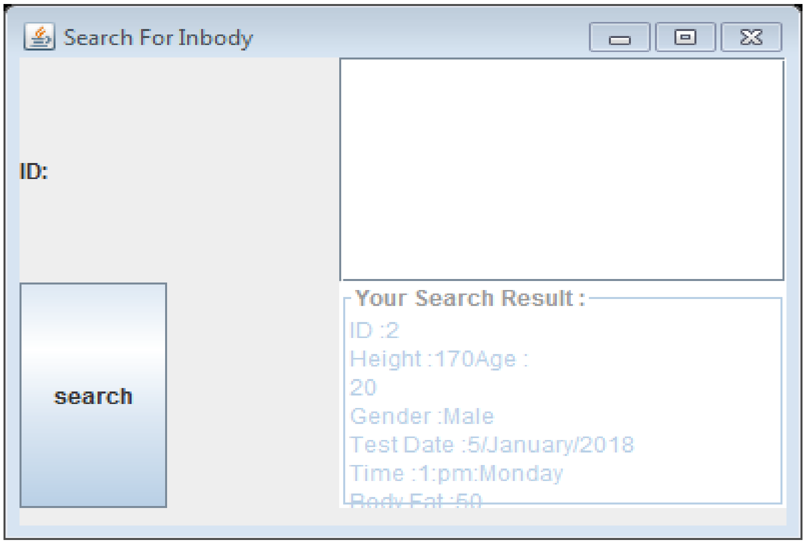
ID :	Height :	Age :	Gender :	Test Dat...	Time :	Body Fat :	Weight :	BMI :	PBF :
2	170	20	Male	5/January/2(	1:pm:Monda	50	80	15	25

xv. Search for InBody:



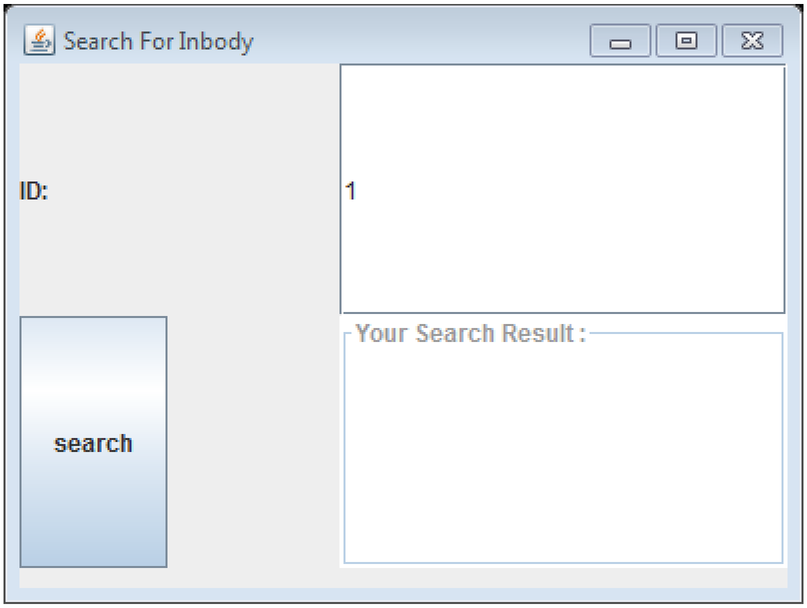


ID: 2



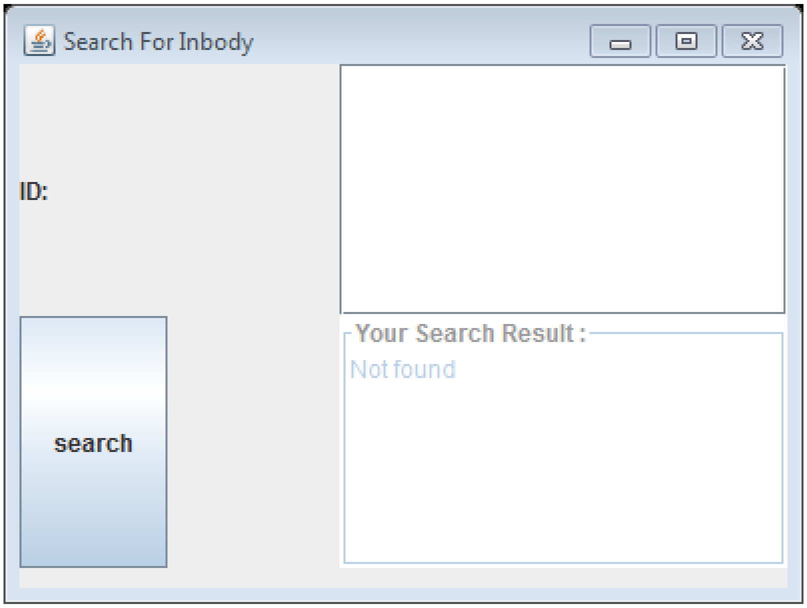
Your Search Result :
ID :2
Height :170Age :
20
Gender :Male
Test Date :5/January/2018
Time :1:pm:Monday
Body Fat :50

xvi. Search for inbody that is not in the list:



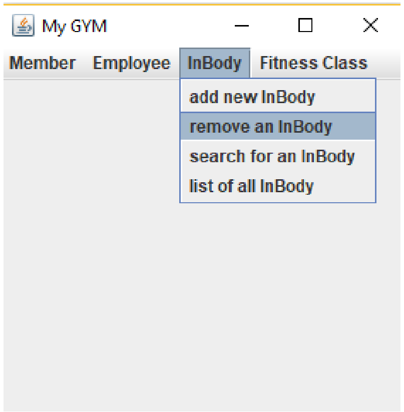
1

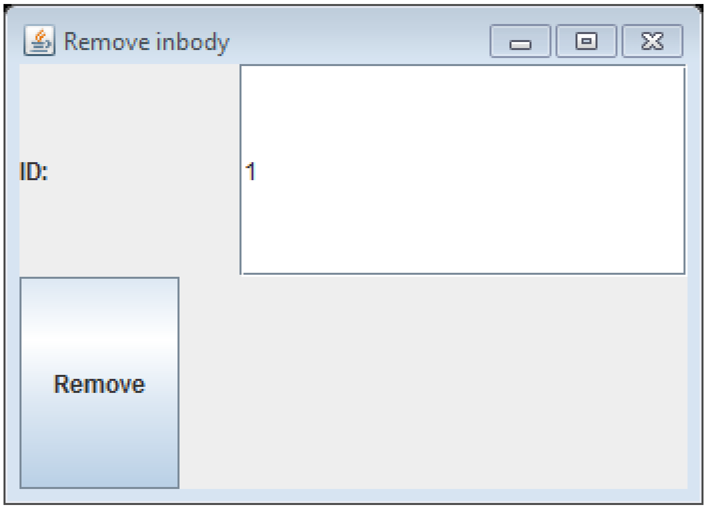
Your Search Result :



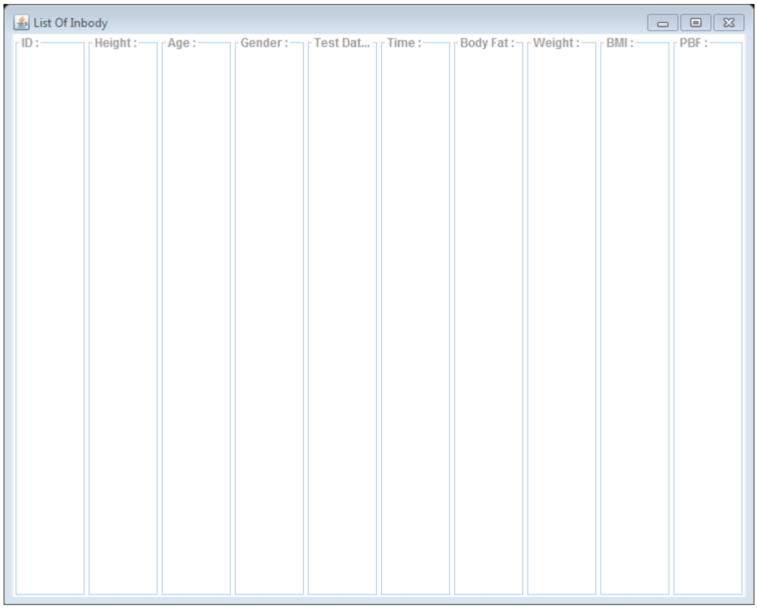
Your Search Result :
Not found

xvii. Remove InBody:





ID: 1



15.Estimated Project Cost:

PERS = 1
RCPX = 1
RUSE = 0.5
PDIF = 1
PREX = 1
FCIL = 1
SCED = 1

- ***M = PERS×RCPX×RUSE×PDIF×PREX×FCIL×SCED = 0.5***

Precedentedness = 1

Development flexibility = 4

Architecture/risk solution= 3

Team cohesion =5

Process maturity =3

- **B = (sumOfFactors/100) + 1.01 = 1.17**

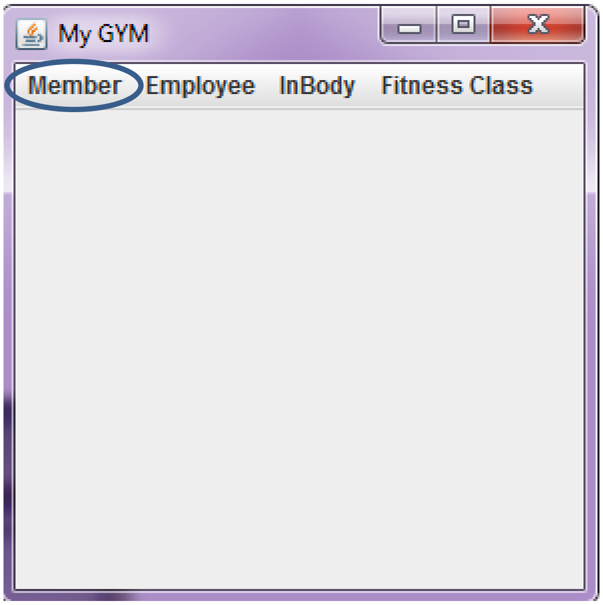
- **PM = A × Size^B × M = 2.94 x 2.47^1.17 x 0.5 = 4.23**

16.User guide:

1. Add / remove / search for a member / open a list of all members:

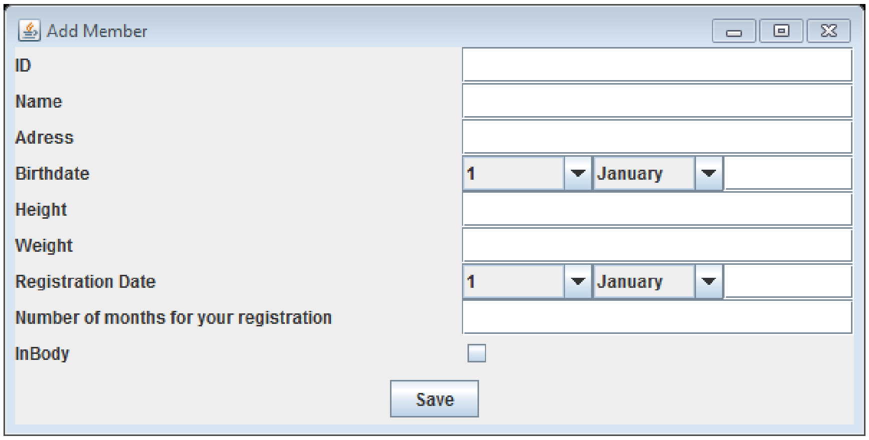
1- Choose member:

2- A list including add member, remove member, search for a member, list of members will appear.



- **To add member:**

1) Choose add member.
2) A window will appear asking you to fill in member's info.



ID

Name

Adress

Birthdate 1 January

Height

Weight

Registration Date 1 January

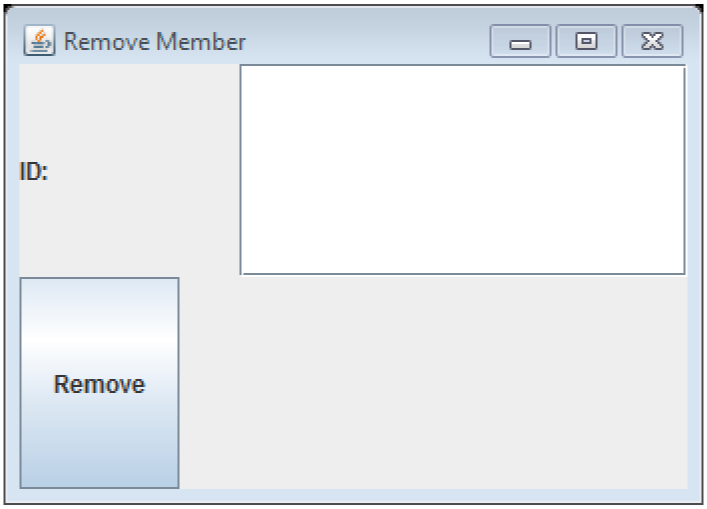
Number of months for your registration

InBody □

Save

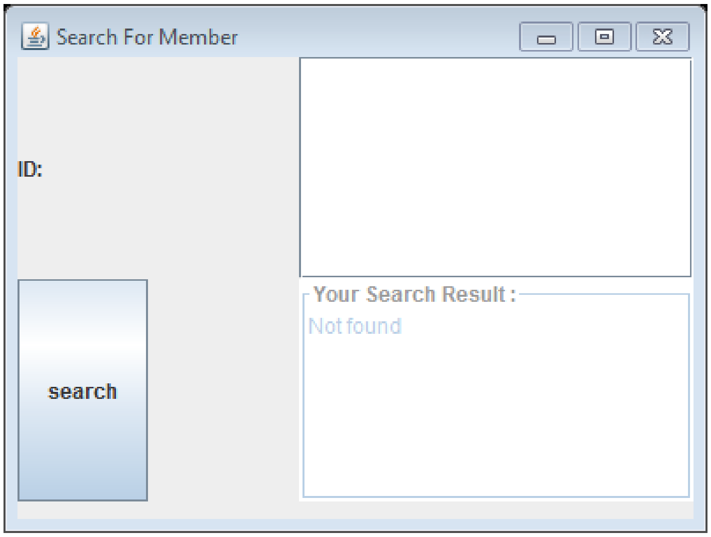
3) Fill in the info.
4) Click save.

- **To remove member:**
  1) Chose remove member.
  2) A window will appear asking you to enter the member's id.



  3) Enter the member's id.
  4) Click remove.

- **To search for a member:**
  1) Choose search for a member.
  2) A window will appear asking you to enter member's id.

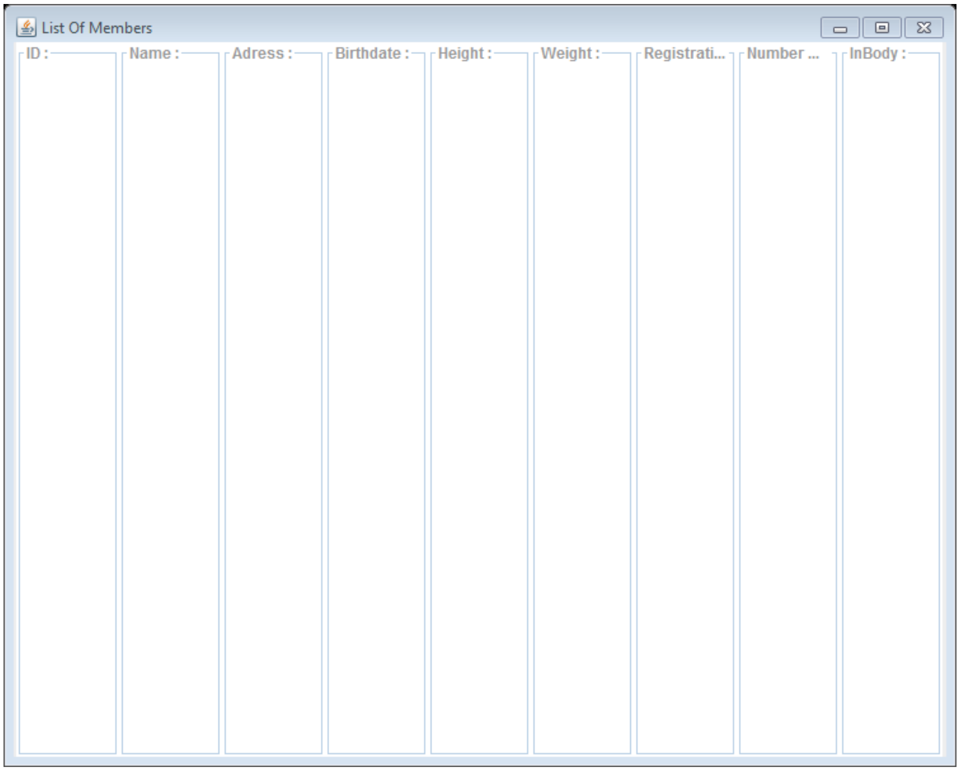


3) Enter the member's id.

4) Click search.

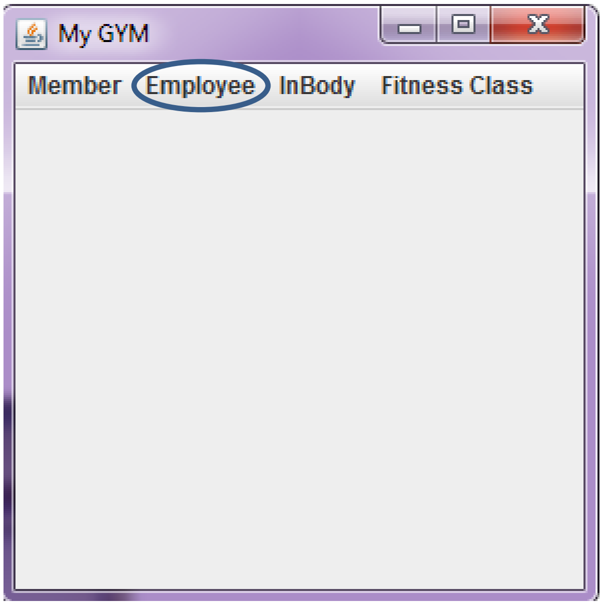
- **To open a list of all members:**

1) Choose list of members
2) A window will appear with a list of all members enrolled in the gym.

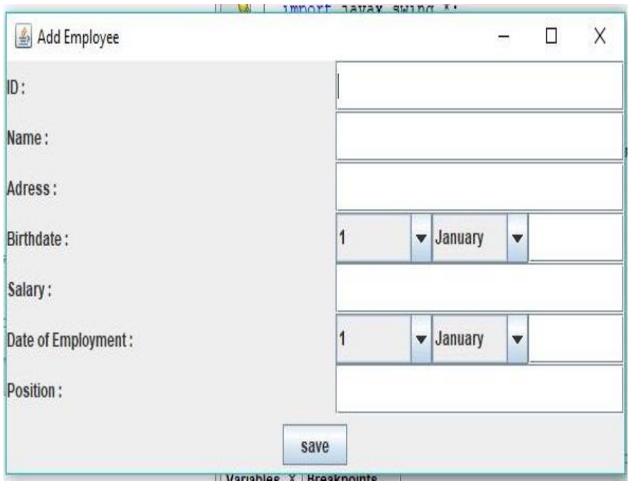


2. Add / remove / search for a member / open a list of all employees:

1- Choose employee:
2- A list including add employee, remove member, search for a member, list of employees will appear.



- **To add employee:**
  1) Choose add employee.
  2) A window will appear asking you to fill in employee's info.



Add Employee

ID :

Name :

Adress :

Birthdate : 1 January

Salary :

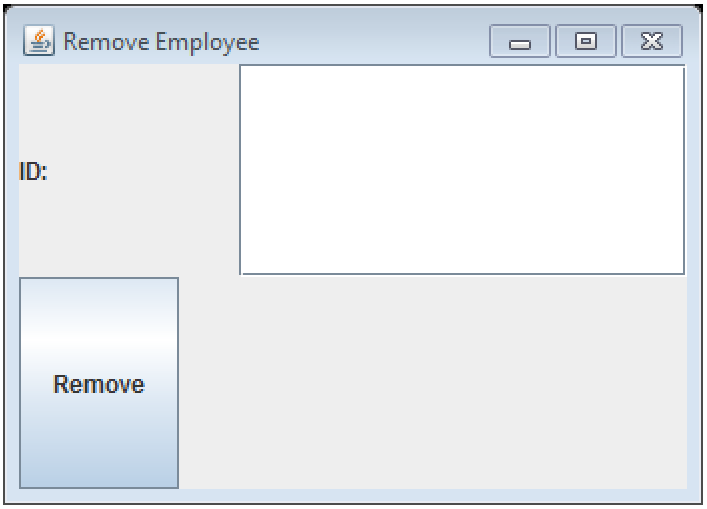
Date of Employment : 1 January

Position :

save

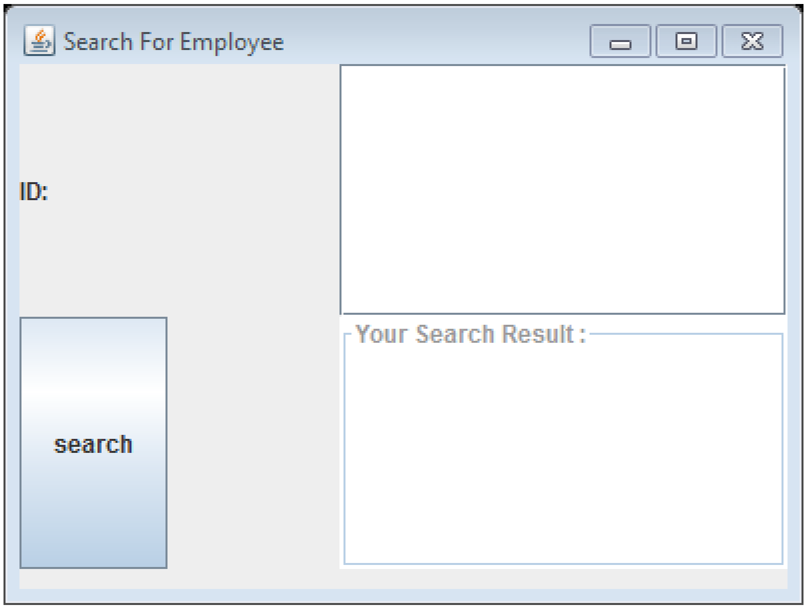
  3) Fill in the info.
  4) Click save.

- **To remove employee:**
  1) Chose remove employee.
  2) A window will appear asking you to enter the employee’s id.



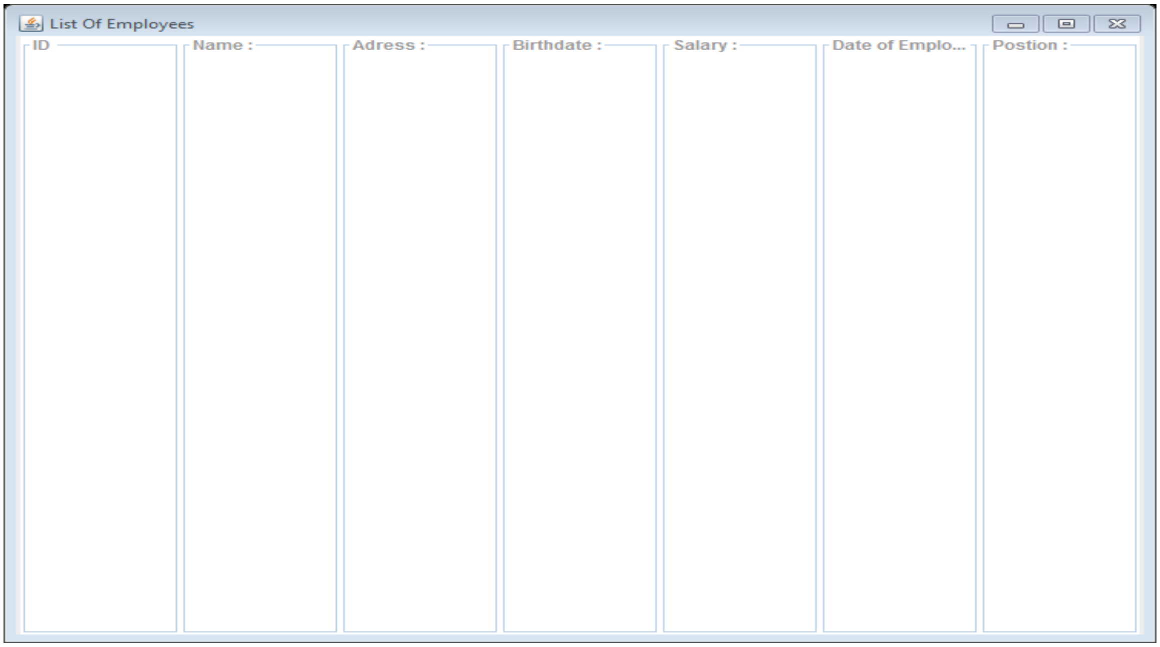
  3) Enter the member’s id.
  4) Click remove.

- **To search for an employee:**
  1) Choose search for a employee.
  2) A window will appear asking you to enter employee's id.



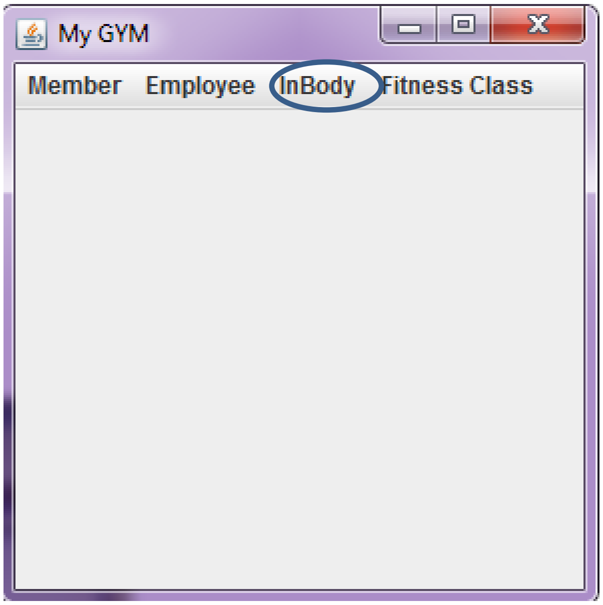
  3) Enter the member's id.
  4) Click search.

- **To open a list of all employees:**
  1) Choose list of employees.
  2) A window will appear with a list of all employees working in the gym.

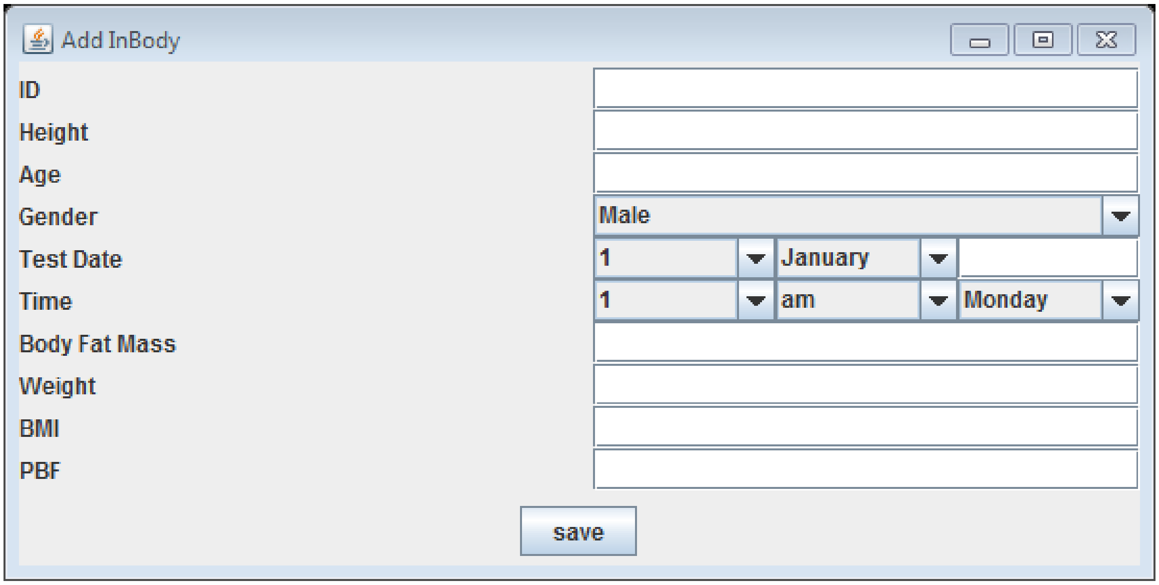


2. Add / remove / search for inbody / open a list of inbody:

1- Choose inbody.
2- A list including add inbody, remove inbody, search for inbody, list of inbody will appear.



- **To add InBody:**
  1) Choose add inbody.
  2) A window will appear asking you to fill in mebmbe's inbody info.

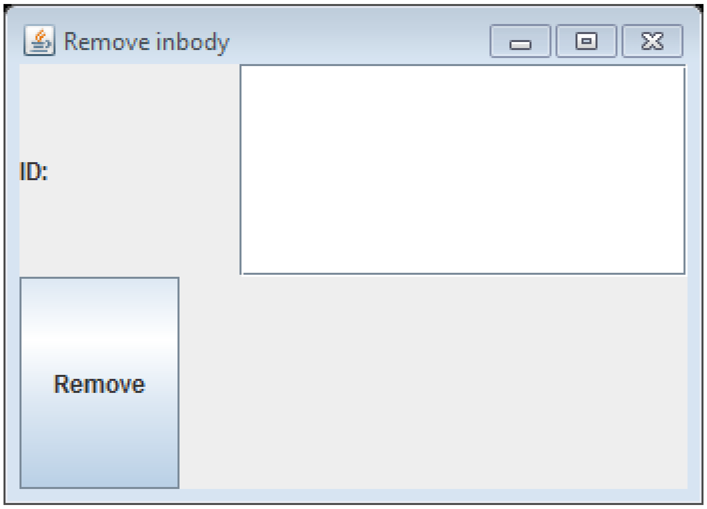


Add InBody

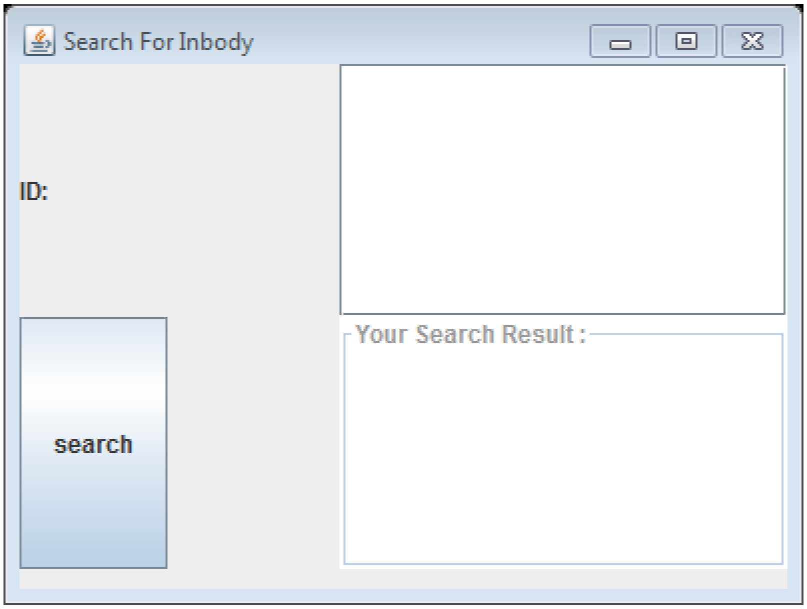
ID			
Height			
Age			
Gender	Male		
Test Date	1	January	
Time	1	am	Monday
Body Fat Mass			
Weight			
BMI			
PBF			

  3) Fill in the info.
  4) Click save.

- **To remove InBody:**
  1) Chose remove inbody.
  2) A window will appear asking you to enter the member’s id.

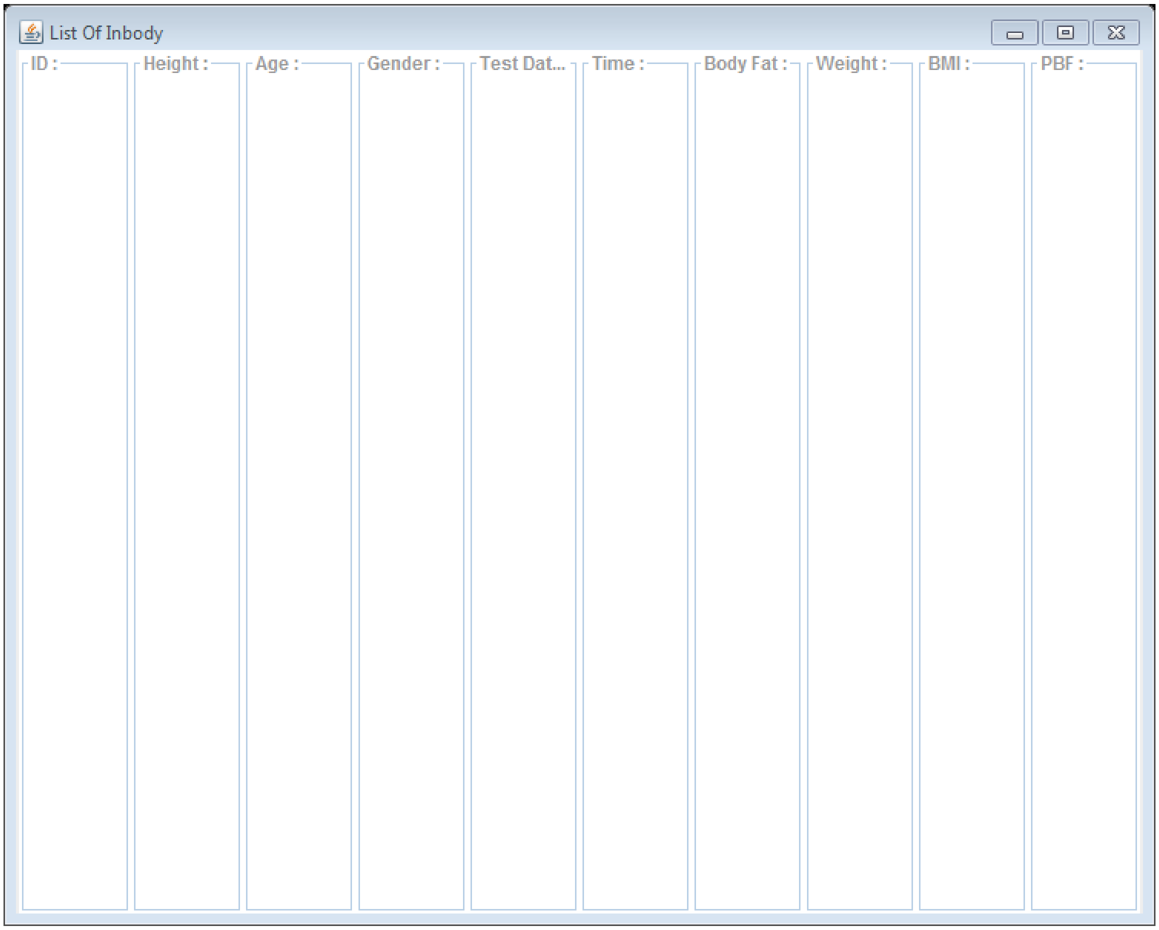


  3) Enter the member’s id.
  4) Click remove.
- **To search for an InBody:**
  1) Choose search for inbody.
  2) A window will appear asking you to enter member’s id.



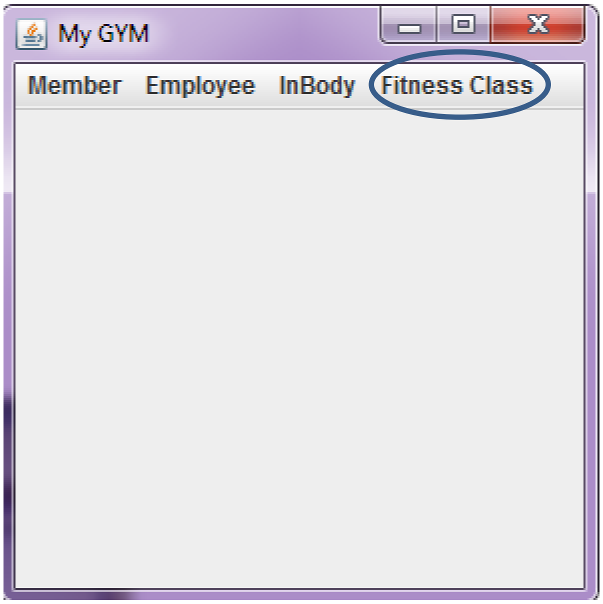
  3) Enter the member’s id.
  4) Click search.

- **To open a list of all InBody:**
    1) Choose list of inbody.
    2) A window will appear with a list of all members' inbody in the gym.

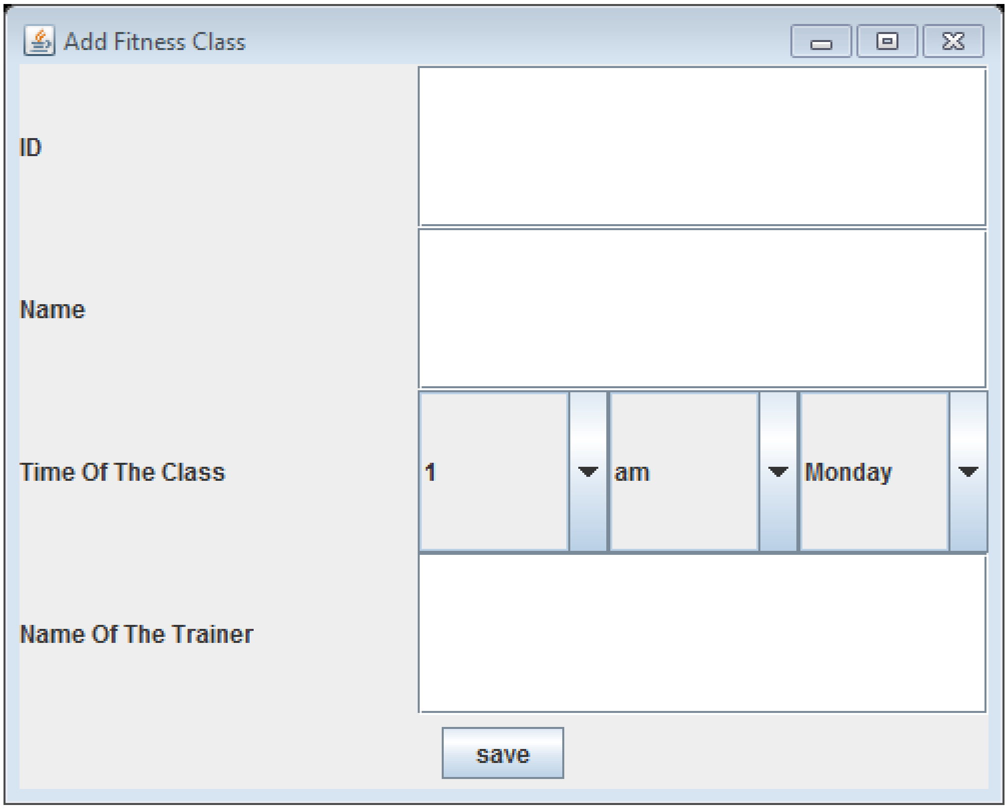


3.Add / remove / search for a fitness class / open a list of all fitness classes:

1- Choose fitness class:
2- A list including add fitness class, remove fitness class, search for a fitness class, list of fitness classes will appear.

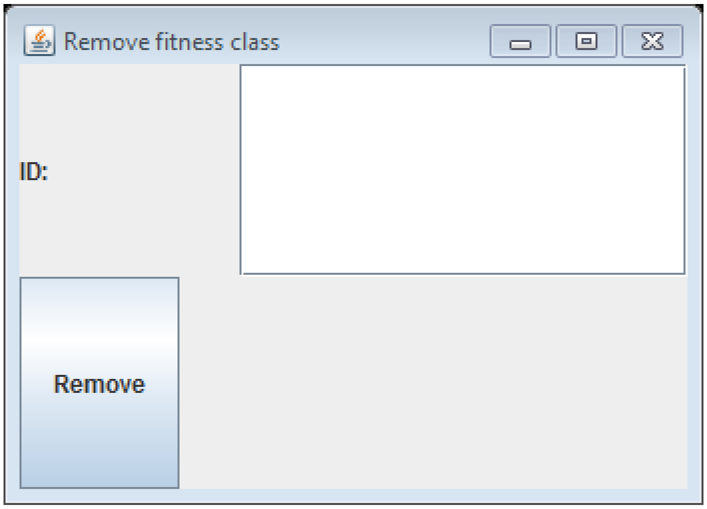


- **To add fitness class:**
  1) Choose add fitness class.
  2) A window will appear asking you to fill in fitness class's info.



  3) Fill in the info.
  4) Click save.

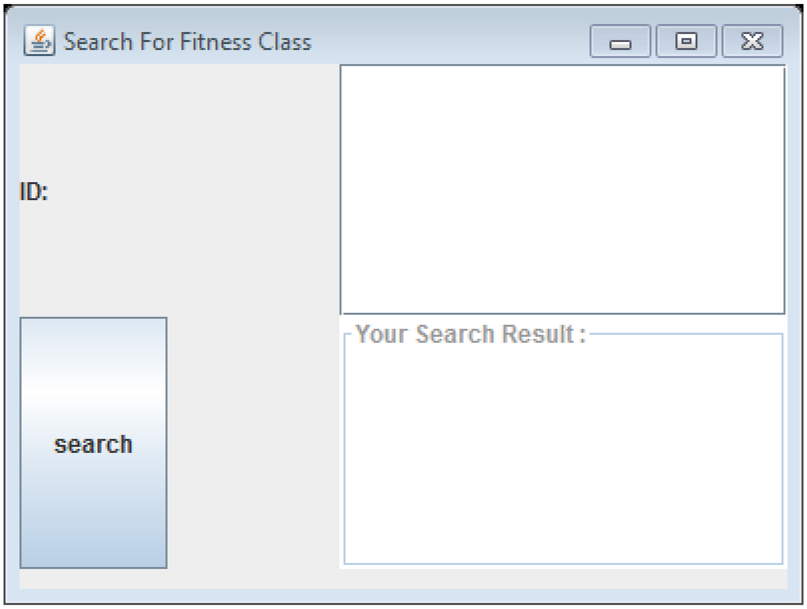
- **To remove fitness class**:
    1) Chose remove fitness class.
    2) A window will appear asking you to enter the fitness class 's id.



3) Enter the fitness class's id.
4) Click remove.

- **To search for a fitness class:**

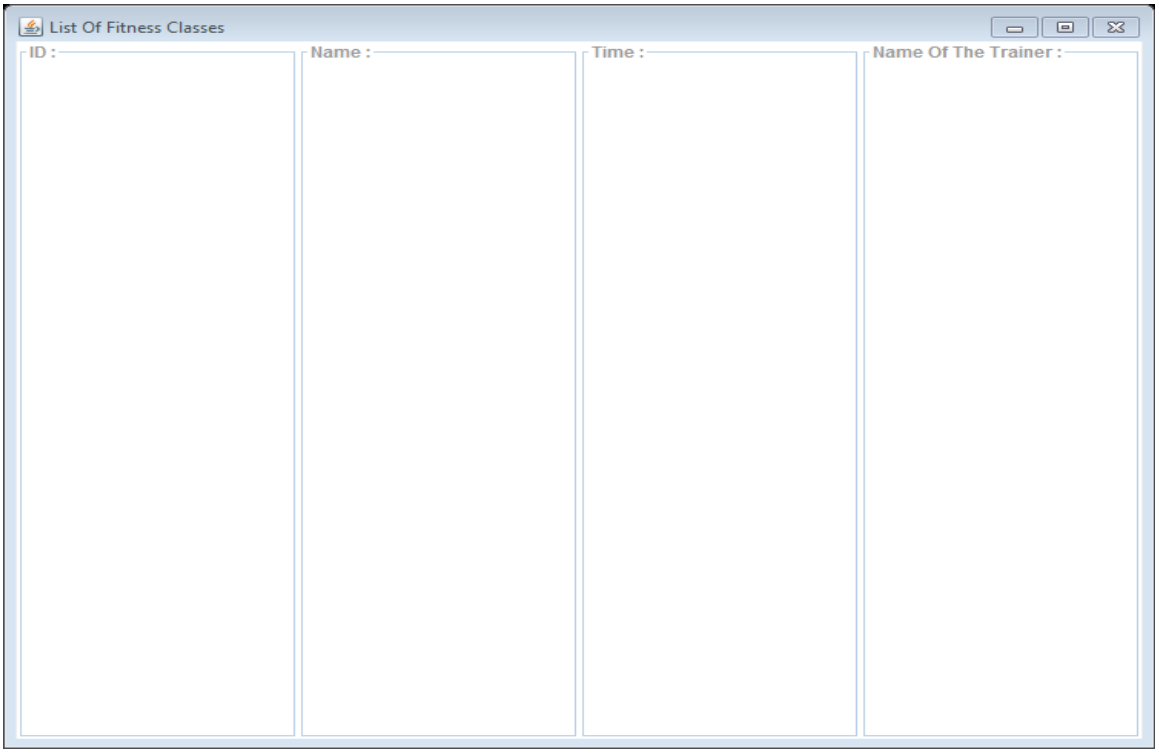
    1) Choose search for a fitness class.
    2) A window will appear asking you to enter fitness class's id.



3) Enter the fitness class's id.
4) Click search.

- **To open a list of all fitness class:**

1) Choose list of fitness classes.
2) A window will appear with a list of all fitness classes in the gym.


List Of Fitness Classes
ID :
Name :
Time :
Name Of The Trainer :