FACULTY OF COMPUTING
UTM Johor Bahru

SECJ2154-04 OBJECT ORIENTED PROGRAMMING (PENGATURCARAAN BERORIENTASIKAN OBJEK)

SEMESTER 2 2023/2024

GROUP PROJECT

SPORTS TEAM MANAGEMENT SYSTEM

NO	NAME	MATRIC NO
1	SERI NUR AYUNI BINTI SALIMI	A22EC0268
2	PUTERI NURIN I’RDINA BINTI FADZIL	A22EC0261
3	FARAH HAZIRAH NISHA BINTI ABD LATIF	A22EC0159
4	SARAH SOFEA BINTI ANUAR	A22EC0104

2/SECR

SECTION 04

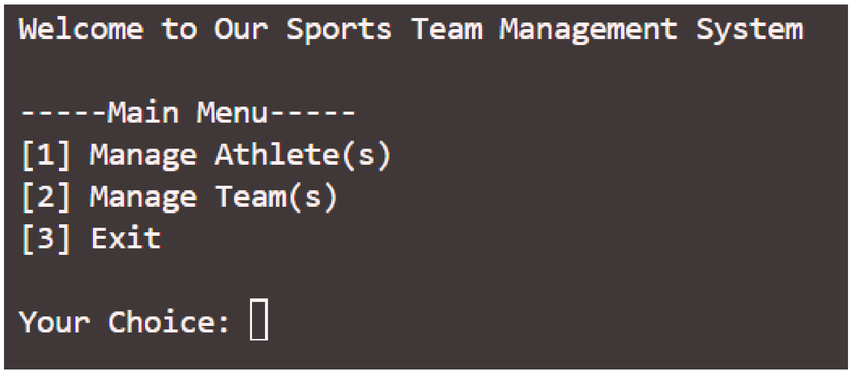
Lecturer:
DR. LIZAWATI BINTI MI YUSUF

The work in this page has been done by: Farah Hazirah and Sarah Sofea   Date:21/6/24

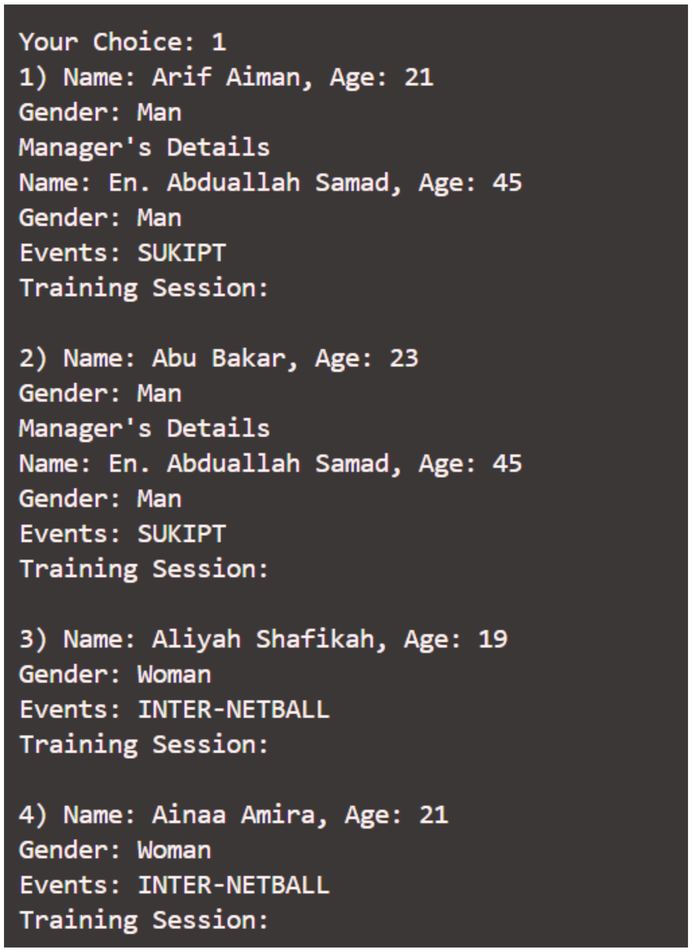
SECTION A: PROJECT DESCRIPTION

Synopsis: (farah hazirah)

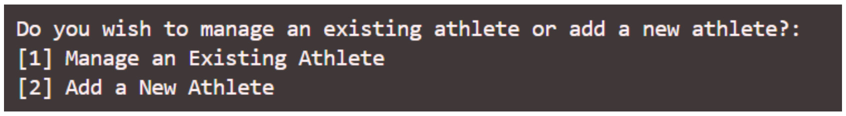
The Sports Team Manager allows users to store data about sports management including the athletes, coach, manager, training session, sports, and events in a more organized way. For instance, we can view, add and remove the athlete's details, teams, team's member and the events of the sport and also their training. First of all, the system will prompt the user to insert the number that is provided in the system (1-3) which will manage the athlete, manage the team, and exit the system. The system will prompt the user to either want to use the existing athlete/team data or want to add a new athlete/team. Next the user will be given 8 options to manage the athlete and team which are to recruit an athlete into the team, remove an athlete from the team, add a training session, remove a training session, add a sports event, cancel a sports event, display team details and exit the athlete/team management menu. All of the options have two options inside them which are either want to use the existing data or add new data except for canceling a sports event, remove an athlete from the team, remove a training session and exit option.



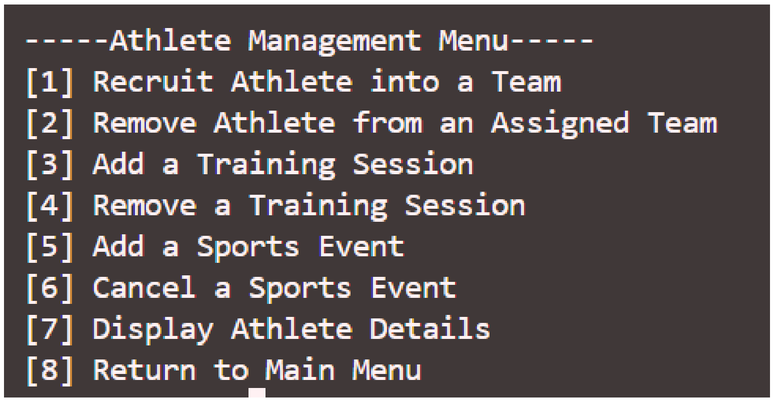
```
Welcome to Our Sports Team Management System

-----Main Menu-----
[1] Manage Athlete(s)
[2] Manage Team(s)
[3] Exit

Your Choice: 
```



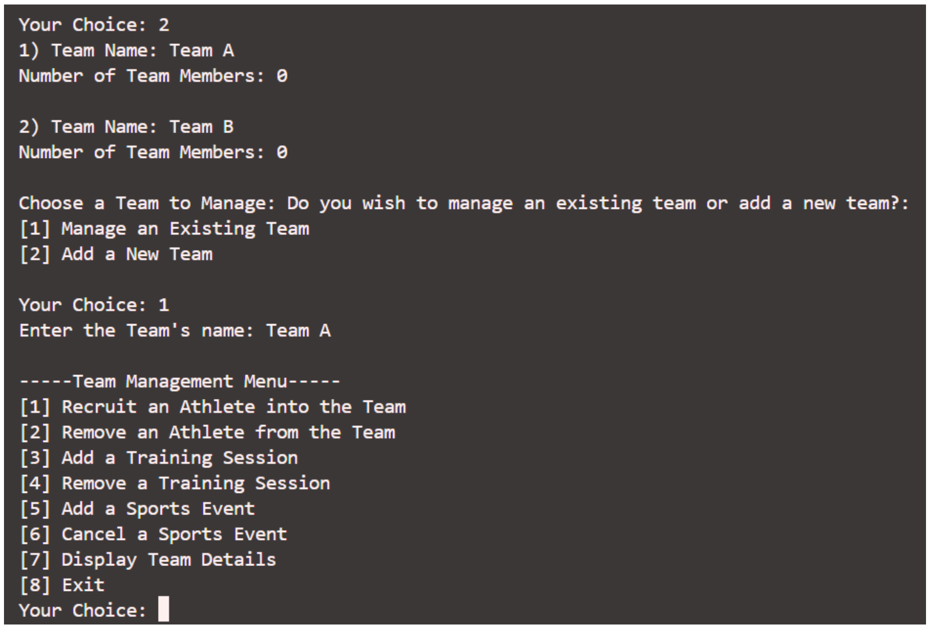
```
Your Choice: 1
1) Name: Arif Aiman, Age: 21
Gender: Man
Manager's Details
Name: En. Abduallah Samad, Age: 45
Gender: Man
Events: SUKIPT
Training Session:

2) Name: Abu Bakar, Age: 23
Gender: Man
Manager's Details
Name: En. Abduallah Samad, Age: 45
Gender: Man
Events: SUKIPT
Training Session:

3) Name: Aliyah Shafikah, Age: 19
Gender: Woman
Events: INTER-NETBALL
Training Session:

4) Name: Ainaa Amira, Age: 21
Gender: Woman
Events: INTER-NETBALL
Training Session:
```



```
Do you wish to manage an existing athlete or add a new athlete?:
[1] Manage an Existing Athlete
[2] Add a New Athlete
```



```
-----Athlete Management Menu-----
[1] Recruit Athlete into a Team
[2] Remove Athlete from an Assigned Team
[3] Add a Training Session
[4] Remove a Training Session
[5] Add a Sports Event
[6] Cancel a Sports Event
[7] Display Athlete Details
[8] Return to Main Menu
```



```
Your Choice: 2
1) Team Name: Team A
Number of Team Members: 0

2) Team Name: Team B
Number of Team Members: 0

Choose a Team to Manage: Do you wish to manage an existing team or add a new team?:
[1] Manage an Existing Team
[2] Add a New Team

Your Choice: 1
Enter the Team's name: Team A

-----Team Management Menu-----
[1] Recruit an Athlete into the Team
[2] Remove an Athlete from the Team
[3] Add a Training Session
[4] Remove a Training Session
[5] Add a Sports Event
[6] Cancel a Sports Event
[7] Display Team Details
[8] Exit
Your Choice: 
```

Objective and Scope: (farah hazirah)

The aim is to apply for a sports team manager based on what we learned in the Object-Oriented Programming which are encapsulation, association, aggregation, composition, inheritance, polymorphism and exception handling. The program provides a menu- driven interface for performing some operations similar as adding, deleting, and viewing athletes, training, events, members of the team to the system, for a based on provided options, and displaying the updated data. In this coding we use the arraylist for our program as it should be able to handle increments of data volume efficiently. First of foremost, we definitely implemented encapsulation and hiding concepts in our system of each class as it was the basic concept in Object Oriented. Aggregation and composition also have been implemented in our coding for the management of the team class(aggregation), athlete class(aggregation) and event class(composition). While inheritance was implemented at coach class, athlete class and manager class that inherit the data from the person class. For the association we use unidirectional and bidirectional concepts so that the object's class can call the methods in other objects. For the polymorphism, we also implemented in the Person class, Coach class and Manager class that have the same method but different behaviour or action. Furthermore, our system has exception handling to prevent the program crashing, as if we don't have this process, exceptions will disrupt our process in the system. The system also has a user-friendly interface with a menu-driven system that determines users through the whole process until they choose to exit the system.

Workflow: (farah hazirah)

All of the data is already declared in the main method class including athlete, manager, coach, events and so on. The system also will provide the menus to prompt the user to choose the option given either want to manage athletes, teams or exit the system. There are also have some operations like inserting new data if the user wants to add on new data in the sports team system. For the Athlete, coach and managers classes, the management of athletes, coaches and managers, each with their own details for instance name, age and gender while for the coaches, experience and expertise will be allowed by the system. For the organization of the team class, team class can be created or managed with the athletes that are assigned to specific teams. It is also associated with sports and also can have training sessions. Next, for the events, it was organized and tracked by including the details like name, date, venue, hour, minutes and also associated with sports. Athletes can be linked to this class that they participate in. Last but not least, the training sessions are to make a schedule for teams and athletes with details on training's date, hour, minutes, duration of training, description and also the venue.

OO Concepts: (sarah sofea)

1.	**Classes and Object**	● **Class** - Blueprint that describes the structure and behavior shared by all objects of a specific kind. - For this management system, we already decided to use 9 classes which consist of class Person,Coach,Manager,Athletes,Team,TrainingSession,Event,Sport and also SportsTeamManagement (main method). - Each class consists of their own attribute and method. ● **Object** - Instances of classes created during runtime. - Each object has its own attributes and can perform methods defined in its class.
2.	**Constructor**	- Special kind of methods for creating instance of the class - Each class that we created have one constructor - Constructors are called when the keyword 'new' is used to create an object.
3.	**Encapsulation and Data Hiding**	- For data hiding using encapsulation, we declared each attribute in each class as private. - This will prevent direct access to the attribute hence ensuring the data integrity. - To access this attribute we need to use an accessor or also known as getter (ex: getName() in 'Person'

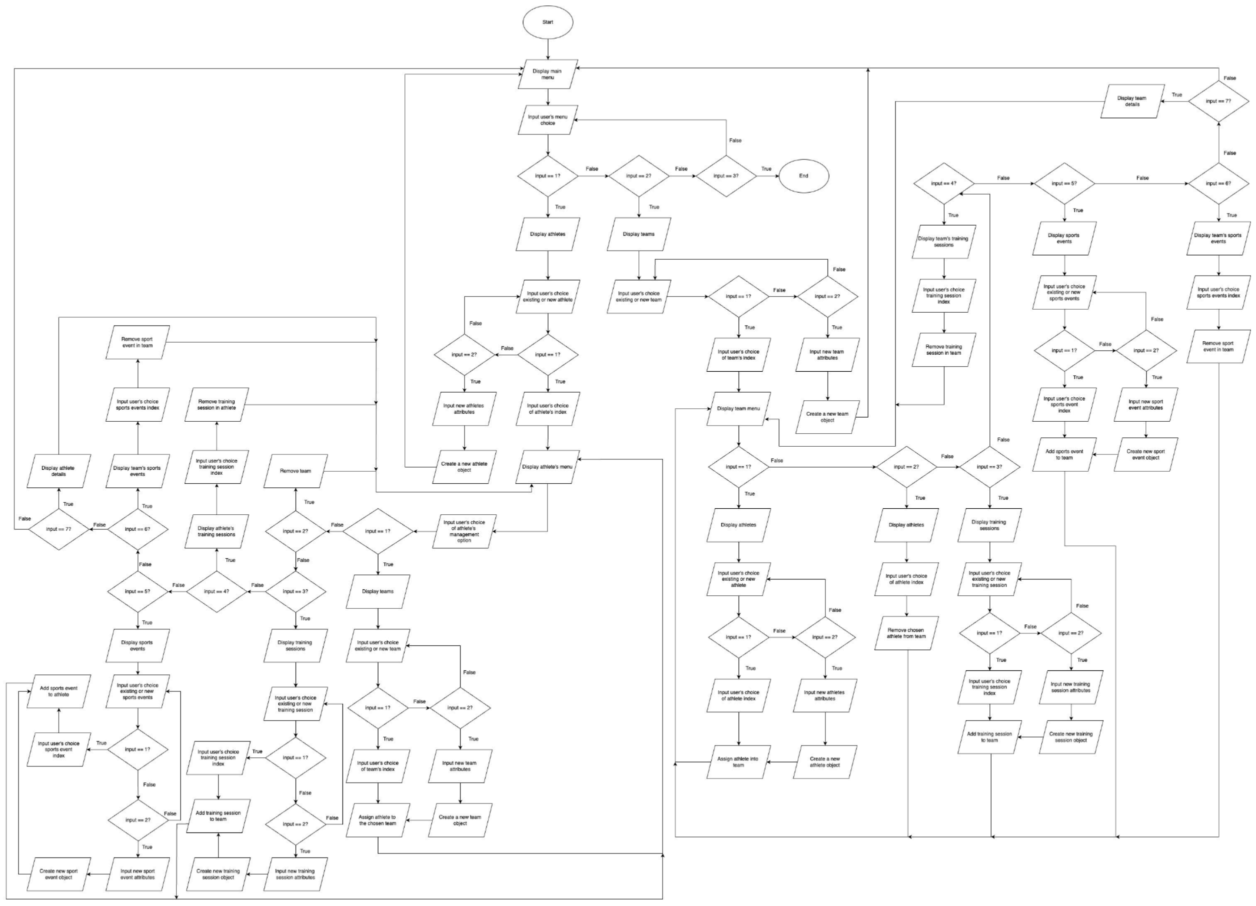
		class).
4.	**Association**	- Represent general binary relationships that describe an activity between two classes. - We implement both unidirectional(one-way) and bidirectional navigability. - By using association, it allows objects to call methods in other objects. As an example, addTrainingSession(TrainingSession) method can be called in ‘Coach’. - It enables one class to gain information about another and interact with its objects.
5.	**Composition**	- Composition is a strong form of aggregation. - The parts (object of another class) belong to one whole class and does not change during the execution. [Strong Ownership]. - As an example, ‘Athlete’ class needs at least one ‘Sport’ object. An ‘Athlete’ cannot exist without being associated with a ‘Sport’.
6.	**Aggregation**	- Aggregation is a special form of association, “has-a” relationship. - Relationship where one whole class contains parts (objects of another class). [Weak Ownership] - As an example, ‘Athlete’ class does not necessarily need to have a ‘Manager’ object. It still can execute even if the ‘Athlete’ does not have a ‘Manager’. - Allows part objects to exist independently of the whole object.
7.	**Inheritance**	- Inheritance implements the “is-a” relationship that

		consists of superclass(general class) and subclass(specialized class). - Process which a new class is created from anothe class. - Allow subclass to inherits the attributes and methods of the superclass. - As an example, ‘Person’ is a superclass that allow other classes such as ‘Coach’, ‘Manager’ and ‘Athlete’ to inherit all the attributes and methods.
8.	**Polymorphism**	- Polymorphism is when the same method call can lead to different behaviors. - Allows methods to be overridden. - Polymorphism also uses dynamic binding that determines the method to be invoked during runtime based on the object type not the reference type. - ‘getInfo()’ method can lead to different behaviors depending on which it is invoked. As an example, if it is called by object ‘Coach’, it will return name, gender, age, experience and expertise. Meanwhile, if it is called by object ‘Manager’, it will return name, gender, age and office.
9.	**Exception Handling**	- For exception handling, we decided to use a do…while loop. - The user will be ask to enter choice 1-3 (Manage Athlete(s), Manage Team(s) and Exit) only, if the user enter any other number, the system will keep looping until the correct input is entered.

The work in this page has been done by: Everyone involve Date:20/06/24

SECTION B: CLASS DESIGNS

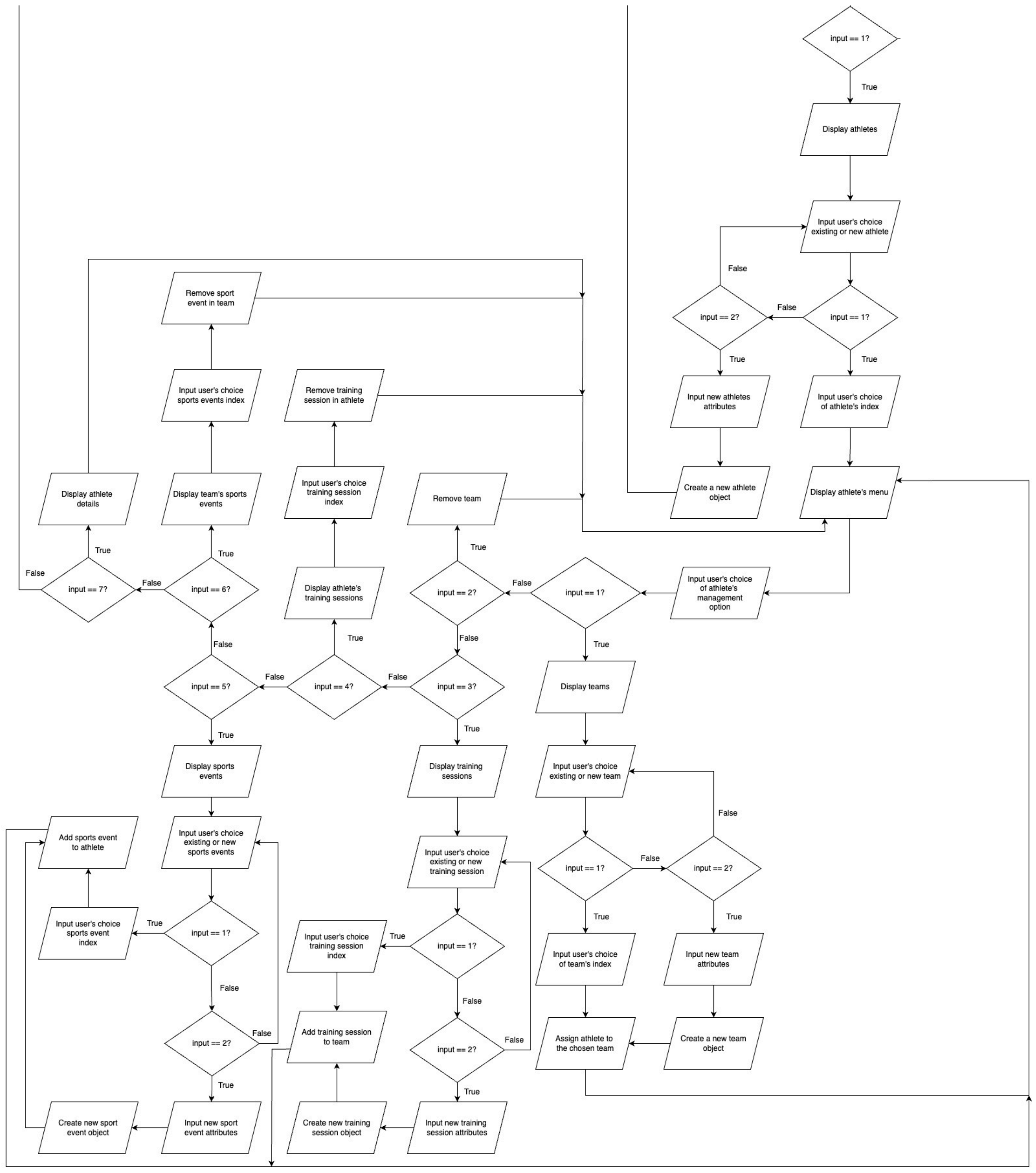
1. Flowchart (Puteri Nurin I'rdina)



Start
Display main menu
Input user's menu choice
input == 1?
input == 2?
input == 3?
End
Display athletes
Display teams
Input user's choice existing or new athlete
Input user's choice existing or new team
Input user's choice of athlete's index
Input new athletes attributes
Create a new athlete object
Display athlete's menu
Input user's choice of athlete's management option
Remove team
Display athlete details
Display team's sports events
Remove sport event in team
Remove training session in athlete
Display athlete's training sessions
Display sports events
Display training sessions
Add sports event to athlete
Create new sport event object
Input new sport event attributes
Add training session to team
Create new training session object
Input new training session attributes
Assign athlete to the chosen team
Create a new team object
Input new team attributes
Input user's choice of team's index
Display team menu
Assign athlete into team
Remove chosen athlete from team
Display team's training sessions
Remove training session in team
Display team details
Display team's sports events
Input user's choice sports events index
Add sports event to team
input == 4?
input == 5?
input == 6?
input == 7?
True
False

Figure 1: Overall Flowchart of the System

The figure above illustrates the overview flow of the flowchart which incorporates multiple layers of data handling, loop, conditional statements and others.



input == 1?
True
Display athletes
Input user's choice existing or new athlete
False
input == 2?
False
input == 1?
True
True
Input new athletes attributes
Input user's choice of athlete's index
Create a new athlete object
Display athlete's menu
Remove sport event in team
Input user's choice sports events index
Remove training session in athlete
Input user's choice training session index
Display athlete details
Display team's sports events
Remove team
Display athlete's training sessions
input == 7?
input == 6?
input == 2?
input == 1?
Input user's choice of athlete's management option
input == 5?
input == 4?
input == 3?
Display teams
Display sports events
Display training sessions
Input user's choice existing or new team
Add sports event to athlete
Input user's choice existing or new sports events
Input user's choice existing or new training session
input == 1?
input == 2?
Input user's choice sports event index
Input user's choice training session index
Input user's choice of team's index
Input new team attributes
Add training session to team
input == 2?
Assign athlete to the chosen team
Create a new team object
Create new sport event object
Input new sport event attributes
Create new training session object
Input new training session attributes

Figure 2: Details of Athlete Management Flow

Figure above illustrates the flow of athlete management option flow, after each option is executed the program will return to the athlete menu display via loop. The options offered for athlete management menu are recruitment and removal of the athlete from a team, joining and canceling a training session, joining and canceling a sport event and displaying athletes details.

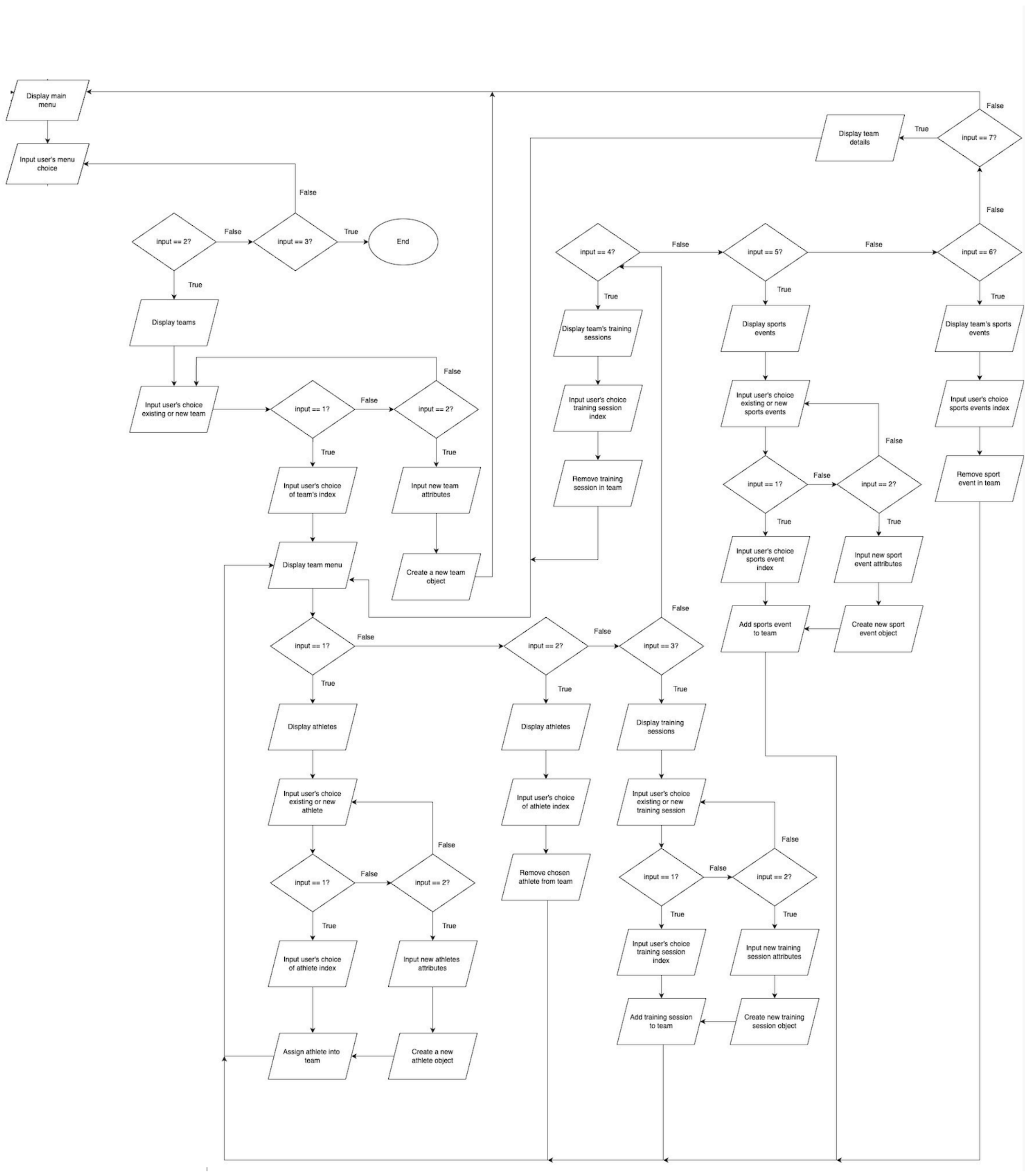

Display main menu
Input user's menu choice
input == 2?
False
input == 3?
True
End
False
True
Display teams
Input user's choice existing or new team
input == 1?
False
input == 2?
False
True
True
Input user's choice of team's index
Input new team attributes
Display team menu
Create a new team object
input == 1?
False
input == 2?
False
input == 3?
False
True
True
True
Display athletes
Input user's choice existing or new athlete
input == 1?
False
input == 2?
False
True
True
Input user's choice of athlete index
Input new athletes attributes
Assign athlete into team
Create a new athlete object
Display athletes
Input user's choice of athlete index
Remove chosen athlete from team
Display training sessions
Input user's choice existing or new training session
input == 1?
False
input == 2?
False
True
True
Input user's choice training session index
Input new training session attributes
Add training session to team
Create new training session object
input == 4?
False
input == 5?
False
input == 6?
False
input == 7?
False
True
Display team details
True
Display team's training sessions
Input user's choice training session index
Remove training session in team
True
Display sports events
Input user's choice existing or new sports events
input == 1?
False
input == 2?
False
True
True
Input user's choice sports event index
Input new sport event attributes
Add sports event to team
Create new sport event object
True
Display team's sports events
Input user's choice sports events index
Remove sport event in team

Figure 3: Details of Team Management and Exit Flow

Figure above illustrates the flow of team management option flow, after each option is executed the program will return to the team menu display via loop. Similar to athlete management flow, the options offered for team management menu are recruitment and removal of the athlete from a team, joining and canceling a training session, joining and canceling a sport event and displaying team details. The program also offers an option to exit the program.

2. Class Diagram (everyone involve)

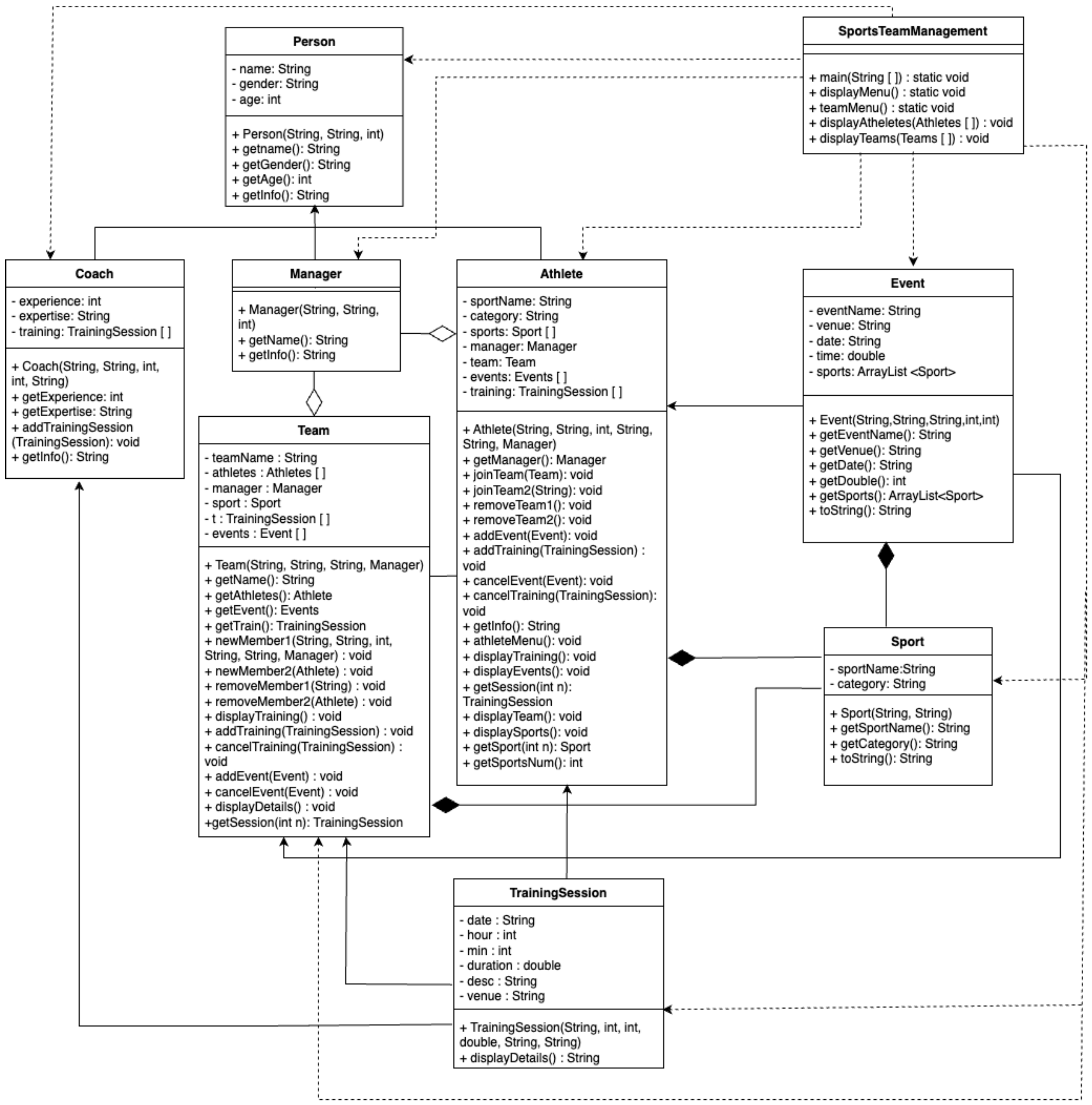

Person
- name: String
- gender: String
- age: int
+ Person(String, String, int)
+ getname(): String
+ getGender(): String
+ getAge(): int
+ getInfo(): String
SportsTeamManagement
+ main(String [ ]) : static void
+ displayMenu() : static void
+ teamMenu() : static void
+ displayAtheletes(Athletes [ ]) : void
+ displayTeams(Teams [ ]) : void
Coach
- experience: int
- expertise: String
- training: TrainingSession [ ]
+ Coach(String, String, int, int, String)
+ getExperience: int
+ getExpertise: String
+ addTrainingSession (TrainingSession): void
+ getInfo(): String
Manager
+ Manager(String, String, int)
+ getName(): String
+ getInfo(): String
Athlete
- sportName: String
- category: String
- sports: Sport [ ]
- manager: Manager
- team: Team
- events: Events [ ]
- training: TrainingSession [ ]
+ Athlete(String, String, int, String, String, Manager)
+ getManager(): Manager
+ joinTeam(Team): void
+ joinTeam2(String): void
+ removeTeam1(): void
+ removeTeam2(): void
+ addEvent(Event): void
+ addTraining(TrainingSession) : void
+ cancelEvent(Event): void
+ cancelTraining(TrainingSession): void
+ getInfo(): String
+ athleteMenu(): void
+ displayTraining(): void
+ displayEvents(): void
+ getSession(int n): TrainingSession
+ displayTeam(): void
+ displaySports(): void
+ getSport(int n): Sport
+ getSportsNum(): int
Event
- eventName: String
- venue: String
- date: String
- time: double
- sports: ArrayList <Sport>
+ Event(String,String,String,int,int)
+ getEventName(): String
+ getVenue(): String
+ getDate(): String
+ getDouble(): int
+ getSports(): ArrayList<Sport>
+ toString(): String
Team
- teamName : String
- athletes : Athletes [ ]
- manager : Manager
- sport : Sport
- t : TrainingSession [ ]
- events : Event [ ]
+ Team(String, String, String, Manager)
+ getName(): String
+ getAthletes(): Athlete
+ getEvent(): Events
+ getTrain(): TrainingSession
+ newMember1(String, String, int, String, String, Manager) : void
+ newMember2(Athlete) : void
+ removeMember1(String) : void
+ removeMember2(Athlete) : void
+ displayTraining() : void
+ addTraining(TrainingSession) : void
+ cancelTraining(TrainingSession) : void
+ addEvent(Event) : void
+ cancelEvent(Event) : void
+ displayDetails() : void
+getSession(int n): TrainingSession
Sport
- sportName:String
- category: String
+ Sport(String, String)
+ getSportName(): String
+ getCategory(): String
+ toString(): String
TrainingSession
- date : String
- hour : int
- min : int
- duration : double
- desc : String
- venue : String
+ TrainingSession(String, int, int, double, String, String)
+ displayDetails() : String

Figure 4: Class Diagram of The Sports Team Management Program

Lastly, the figure above illustrates the class diagram of the sports team management program. It consists of multiple classes such as Person, Coach, Manager, Athlete, Team, TrainingSession, Sport, Event and SportsTeamManagement. This program employs multiple relationships such as association, aggregation and composition. Each class also employs various private attributes and public methods and constructors.

Class Person: //ziera

Attributes	Description
name	The name of athlete, coach and manager
gender	The gender of athlete, coach and manager
age	The age of athlete, coach and manager
Methods	**Description**
Person	To construct the Person's objects and assign values to the Person's attributes
getName	To get access name attributes
getGender	To get access gender attributes
getAge	To get access age attributes
getInfo	To display the name, gender, and age

Class Coach:

Attributes	Description
experience	The experience of a coach being a coach
expertise	The sport expertise of a coach
training	The association concept to update the TrainingSession class
Methods	**Description**
Coach(String, String, int, int, String)	To construct the Coach's objects and assign values to the Coach's attributes
getExperience()	To get access experience attributes
getExpertise()	To get access expertise attributes
addTrainingSession(TrainingSession)	To update the training session
getInfo()	To display the Coach's details (name, gender, age, experience and expertise)

Class Athlete:

Attributes	Description
sportName	The name of the athlete’s sport
category	The category of the athlete’s sport
sports	List of sports
manager	Athlete’s manager (athlete does not necessarily need to have manager)
team	Athlete’s team (athlete does not necessarily need to have team)
events	List of events
training	List of training session
Methods	**Description**
Athlete(String, String, int, String, String, Manager)	To construct the Athlete’s objects and assign values to the Athlete’s attributes
joinTeam(Team)	For Athlete join Team by Object
joinTeam2(String)	For Athlete join Team by Name
removeTeam1()	To remove Athlete in Team
removeTeam2()	To remove Team
addEvent(Event)	To add Event that the Athlete join
cancelEvent(Event)	To cancel Event that the Athlete join
addTraining(TrainingSession)	To add training session for the Athlete
cancelTraining(TrainingSession)	To cancel training session for the Athlete
getInfo()	To get all the Athlete’s details
athleteMenu()	To print Athlete Management Menu
displayTraining()	To display Training Session for the Athlete
displayEvents()	To display Event for the Athlete

getSession(int)	To access Training Session attributes
displayTeam()	To display if Athlete has no team
displaySports()	To display Sports for the Athlete
getSport(int)	To access Sport attributes
getSportNum()	To return number of sports an Athlete join

Class Manager:

Methods	Description
Manager	To construct Manager’s object and assign values to the Manager's attributes
getRoom()	To get room’s number of manager’s office
getInfo()	To get manager’s details

Class Team:

Attributes	Description
teamName	The team’s name
athletes	List of Athletes in the Team
manager	The manager for the team (does not necessarily to have a manager)
sport	The sport that the Team join
t	List of TrainingSeesion for the team
events	List of Event for the team
Methods	**Description**

Team(String, String, String, Manager)	To construct Team's object and assign values to the Team's attributes
newMember1(String, String, int, String, String, Manager)	To add Athlete in team by attributes
newMember2(Athlete)	To add Athlete in team by object
removeMember1(String)	To remove Athlete from the Team by using Athlete's name
removeMember2(Athlete)	To remove Athlete from the team by using object
getName()	To get access to teamName attributes
addTraining(TrainingSession)	To add training session for the Team
cancelTraining(TrainingSession)	To cancel training session for the Team
displayTraining()	To display all the Training Session that the Team have
displayDetails()	To display details about the Team
addEvent(Event)	To add Event that the Team join
cancelEvent(Event)	To cancel Event that the Team join
getSession(int)	To access Training Session attributes

Class TraningSession:

Attributes	**Description**
date	The date of the training session
hour	The time of the event occur in hour, using 24 hours format (eg: **08**:30)
min	The time of the event occur in min, using 24 hours format (eg: 08:**30**)
duration	The duration of the training session
desc	The description on what are they going to do

	during training session
venue	The venue of the training session
Methods	**Description**
TrainingSession(String, double, double, double, String, String)	To construct the TrainingSession's objects and assign values to the TrainingSession's attributes
getDetails()	To display details about the Training Session

Class Sport:

Attributes	**Description**
sportName	The sport's name
category	The category for the sport's (eg: U23, U21)
Methods	**Description**
Sport(String, String)	To construct Sport's object and assign values to the Sport's attributes
getSportName()	To get access to sportName attributes
getCategory()	To get access to category attributes
toString()	To return sportName and category attributes.

Class Event:

Attributes	**Description**
eventName	The event's name

venue	The venue of the event occur
date	The date of the event occur
time	The time of the event occur using 24 hours format (eg: 08:30)
sports	The sports that occur during the event
Methods	**Description**
Event (String, String, String, int, int)	To construct Event's object and assign values to the Event's attributes
getEventName()	To get access to eventName attributes
getVenue()	To get access to venue attributes
getDate()	To get access to date attributes
getTime()	To get access to time attributes
getSports()	To get access to sports attributes
toString()	To return all the attributes declared in this class

The work in this page has been done by: Seri Nur Ayuni Date:21/06/24

SECTION C: SOURCE CODE AND USER MANUAL

(Texts in **Blue** and **Bold** are user inputs)

Main Menu (3 Options)

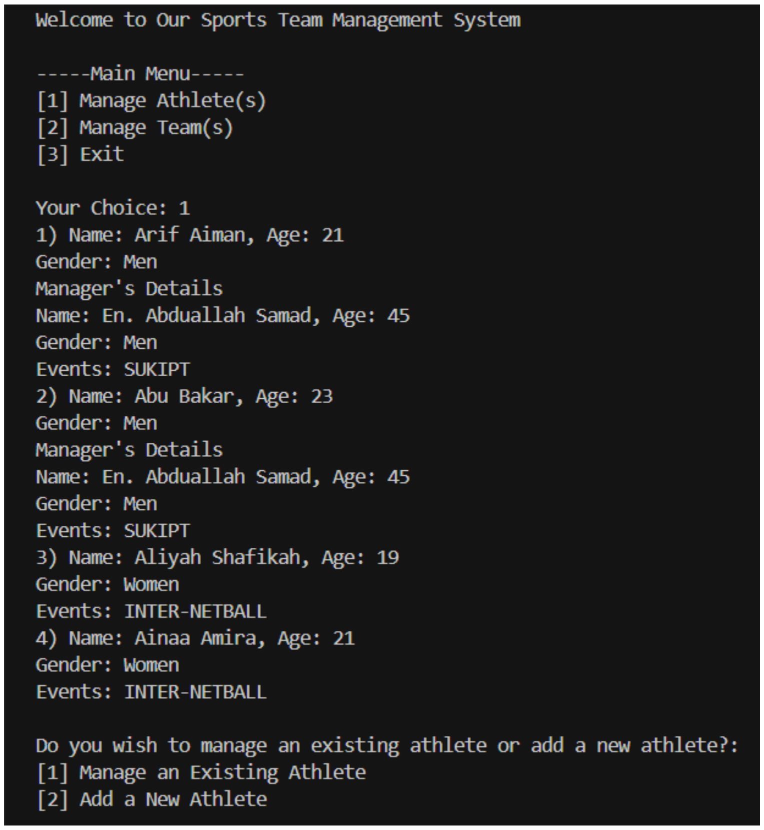
1. Manage Athlete(s)

Welcome to Our Sports Team Management System

-----Main Menu-----
[1] Manage Athlete(s)
[2] Manage Team(s)
[3] Exit

Your Choice: **1**

Output:



```
Welcome to Our Sports Team Management System

-----Main Menu-----
[1] Manage Athlete(s)
[2] Manage Team(s)
[3] Exit

Your Choice: 1
1) Name: Arif Aiman, Age: 21
Gender: Men
Manager's Details
Name: En. Abduallah Samad, Age: 45
Gender: Men
Events: SUKIPT
2) Name: Abu Bakar, Age: 23
Gender: Men
Manager's Details
Name: En. Abduallah Samad, Age: 45
Gender: Men
Events: SUKIPT
3) Name: Aliyah Shafikah, Age: 19
Gender: Women
Events: INTER-NETBALL
4) Name: Ainaa Amira, Age: 21
Gender: Women
Events: INTER-NETBALL

Do you wish to manage an existing athlete or add a new athlete?:
[1] Manage an Existing Athlete
[2] Add a New Athlete
```

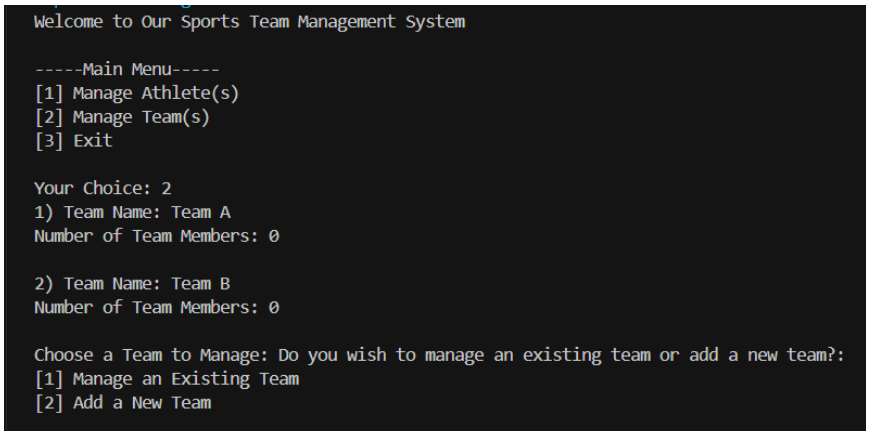
2. Manage Team(s)

Welcome to Our Sports Team Management System

-----Main Menu-----
[1] Manage Athlete(s)
[2] Manage Team(s)
[3] Exit

Your Choice: **2**

Output:



```
Welcome to Our Sports Team Management System

-----Main Menu-----
[1] Manage Athlete(s)
[2] Manage Team(s)
[3] Exit

Your Choice: 2
1) Team Name: Team A
Number of Team Members: 0

2) Team Name: Team B
Number of Team Members: 0

Choose a Team to Manage: Do you wish to manage an existing team or add a new team?:
[1] Manage an Existing Team
[2] Add a New Team
```

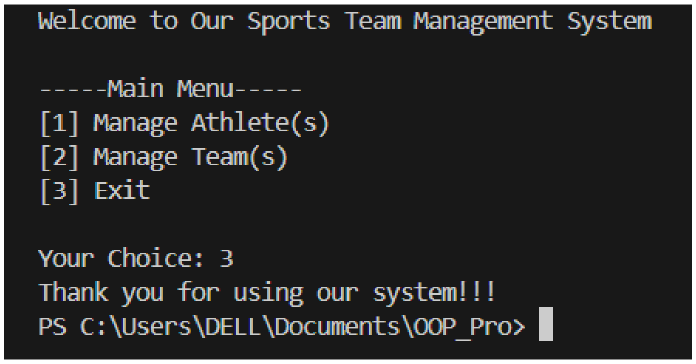
3. Exit

Welcome to Our Sports Team Management System

-----Main Menu-----
[1] Manage Athlete(s)
[2] Manage Team(s)
[3] Exit

Your Choice: **3**

Output:

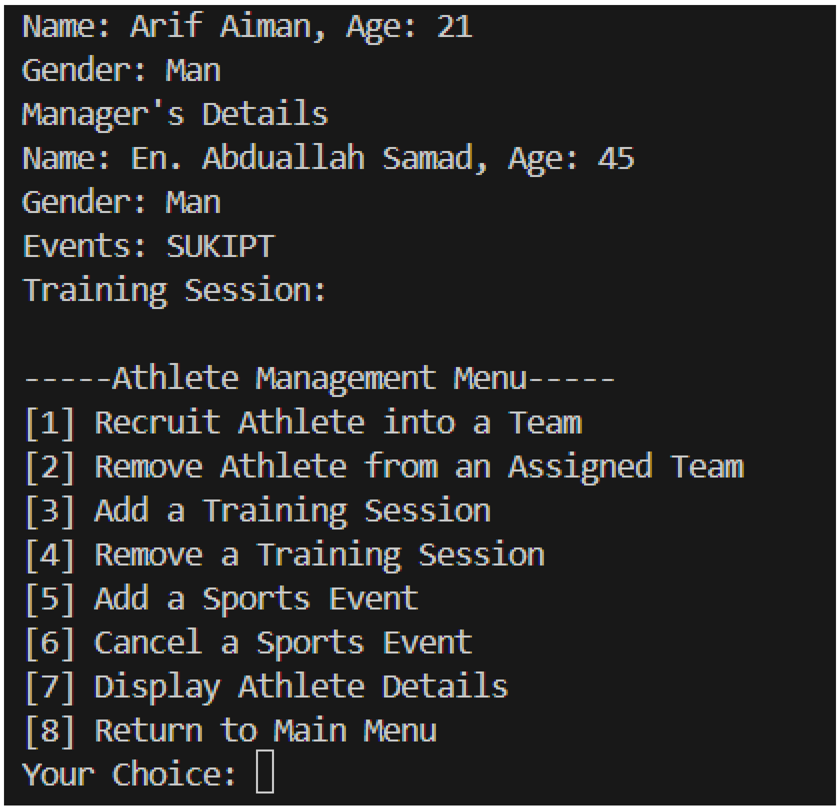


```
Welcome to Our Sports Team Management System

-----Main Menu-----
[1] Manage Athlete(s)
[2] Manage Team(s)
[3] Exit

Your Choice: 3
Thank you for using our system!!!
PS C:\Users\DELL\Documents\OOP_Pro>
```

Main Prompt (2 Options)

1. **Manage an Existing Athlete**
   Do you wish to manage an existing athlete or add a new athlete?:
   [1] Manage an Existing Athlete
   [2] Add a New Athlete

   Your Choice: **1**
   Choose an Athlete to Manage: 1
   Output:

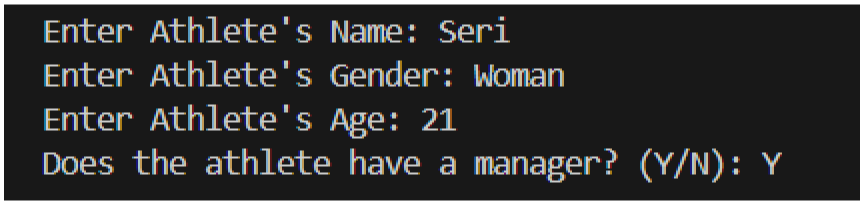


```
Name: Arif Aiman, Age: 21
Gender: Man
Manager's Details
Name: En. Abduallah Samad, Age: 45
Gender: Man
Events: SUKIPT
Training Session:

-----Athlete Management Menu-----
[1] Recruit Athlete into a Team
[2] Remove Athlete from an Assigned Team
[3] Add a Training Session
[4] Remove a Training Session
[5] Add a Sports Event
[6] Cancel a Sports Event
[7] Display Athlete Details
[8] Return to Main Menu
Your Choice: 
```

2. **Add a New Athlete**
   Do you wish to manage an existing athlete or add a new athlete?:
   [1] Manage an Existing Athlete
   [2] Add a New Athlete

   Your Choice: **2**

   Example Output: It will ask user to input for athlete's details


```
Enter Athlete's Name: Seri
Enter Athlete's Gender: Woman
Enter Athlete's Age: 21
Does the athlete have a manager? (Y/N): Y
```

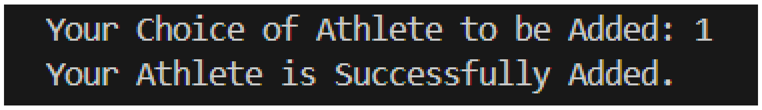
Team Management Options

1. **Recruit an Athlete into the Team**

   Do you wish to add an existing Athelete or add a new Athlete?
   [1] Add an Existing Athlete
   [2] Add a New Athlete

   1. **Add an Existing Athlete**
      - Insert index of athlete from existing team list to be assigned to the team

        Example Output:


```
Your Choice of Athlete to be Added: 1
Your Athlete is Successfully Added.
```

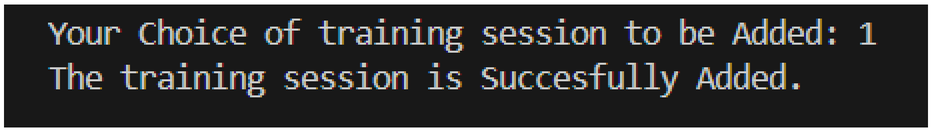
   2. **Add a New Athlete**
      - Insert new athlete's details in user prompts

2. **Remove Athlete from an the Team**
   - Remove the selected athlete from assigned team
3. **Add a Training Session**

   Do you wish to add an existing training session or add a new training session?
   [1] Add an Existing Training Session
   [2] Add a New Training Session

   1. **Add an Existing Training Session**
      - Insert index of training session from existing training session list to be assigned to the team

        Example Output:


```
Your Choice of training session to be Added: 1
The training session is Succesfully Added.
```

   2. **Add a New Training Session**
      - Insert new training session's details in user prompts

4. **Remove a Training Session**

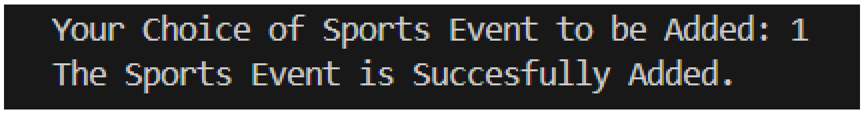
- Remove the selected training session from the team

5. **Add a Sports Event**

   Do you wish to add an existing Event or add a new Event?
   [1] Add an Existing Event
   [2] Add a New Event

   1. **Add an Existing Event**
      - Insert index of event from existing event list to be assigned to the team

        Example Output:



```
Your Choice of Sports Event to be Added: 1
The Sports Event is Succesfully Added.
```

   2. Add a New Event
      - Insert new event's details in user prompts

6. **Cancel a Sports Event**
   - Remove the selected event from the team
7. **Display Team Details**
   - Display all details of the team
8. **Return to Main Menu**

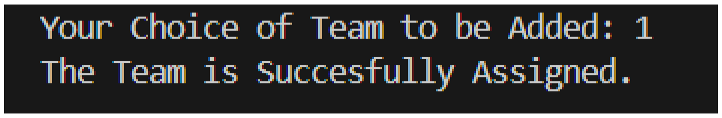
Athlete Management Options

9. **Recruit Athlete into a Team**

   Do you wish to add an existing team or add a new Team?
   [1] Add an Existing Team
   [2] Add a New Team

   1. **Add an Existing Team**
      - Insert index of team from existing team list to be assigned to the athlete

        Example Output:



```
Your Choice of Team to be Added: 1
The Team is Succesfully Assigned.
```

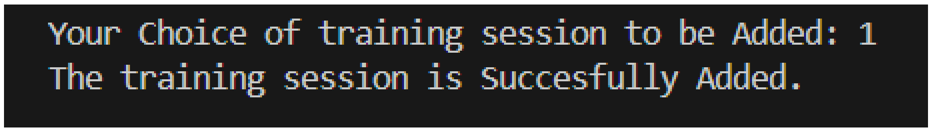
   2. Add a New Team
      - Insert new team's details in user prompts

10. **Remove Athlete from an Assigned Team**
    - Remove the selected athlete from assigned team
11. **Add a Training Session**

Do you wish to add an existing training session or add a new training session?
[1] Add an Existing Training Session
[2] Add a New Training Session

1. **Add an Existing Training Session**
   - Insert index of training session from existing training session list to be assigned to the athlete

   Example Output:



```
Your Choice of training session to be Added: 1
The training session is Succesfully Added.
```

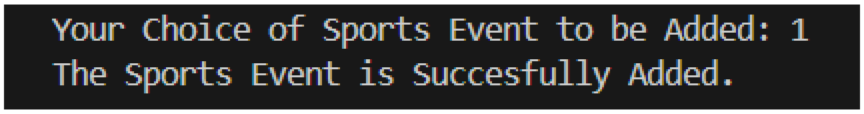
2. **Add a New Training Session**
   - Insert new training session's details in user prompts

12. **Remove a Training Session**
    - Remove the selected training session from the athlete
13. **Add a Sports Event**

Do you wish to add an existing Event or add a new Event?
[1] Add an Existing Event
[2] Add a New Event

1. **Add an Existing Event**
   - Insert index of event from existing event list to be assigned to the athlete

   Example Output:



```
Your Choice of Sports Event to be Added: 1
The Sports Event is Succesfully Added.
```

2. **Add a New Event**
   - Insert new event's details in user prompts

14. **Cancel a Sports Event**
    - Remove the selected event from the athlete
15. **Display Athlete Details**
    - Display all details of the athlete
16. **Return to Main Menu**