**To Build the Solution Using Object-Oriented Programming in Java**

Course-end Project 1

Description

To build the Solution using Object Oriented Programming in Java.

**Prerequisites:**

* OOPS
* Java

**Description:**

This assignment is designed to help understand working with OOPS in Java and how the polymorphic behavior can be implemented.

**Problem Statement:**

Yamin is a Gym Instructor. He takes multiple sessions of Zumba during the day, for which he has divided his participants into batches running in the morning and evening hours. To manage his participants, he needs a software solution to record the details of his participants and batches. Luckily his friend Mike runs a software company. Therefore, Yamin has requested the company to develop a software solution to manage his participants and batches.

The company aims to develop a core java based solution using OOPS.

**Tasks:**

Perform the below activities as a solution:

* Create a Java Project in Eclipse
* Create a Participant Class
* Create a Batch Class
* Create 1 to many relationship mapping for the Batch and Participant
* Take the inputs from the User using Scanner class to create Participant and Batch Objects
* Implement Method callbacks using Run time Polymorphism to implement. When a batch starts, the Participant should receive a message
* Run the Code in the main class

**Code:**

**=======**

**import** java.util.ArrayList;

**import** java.util.Scanner;

**class** Participant {

**private** String name;

**public** Participant(String name) {

**this**.name = name;

}

**public** String getName() {

**return** name;

}

**public** **void** receiveMessage(String message) {

System.***out***.println(name + " received message: " + message);

}

}

**class** Batch {

**private** String name;

**private** ArrayList<Participant> participants;

**public** Batch(String name) {

**this**.name = name;

participants = **new** ArrayList<>();

}

**public** **void** addParticipant(Participant participant) {

participants.add(participant);

}

**public** **void** start() {

**for** (Participant participant : participants) {

participant.receiveMessage("Batch " + name + " has started.");

}

}

}

**public** **class** Main {

**public** **static** **void** main(String[] args) {

Scanner scanner = **new** Scanner(System.***in***);

System.***out***.println("Enter the name of the batch:");

String batchName = scanner.nextLine();

Batch batch = **new** Batch(batchName);

System.***out***.println("Enter the number of participants:");

**int** numParticipants = scanner.nextInt();

scanner.nextLine(); // Consume newline

**for** (**int** i = 0; i < numParticipants; i++) {

System.***out***.println("Enter participant name " + (i + 1) + ":");

String participantName = scanner.nextLine();

Participant participant = **new** Participant(participantName);

batch.addParticipant(participant);

}

batch.start();

scanner.close();

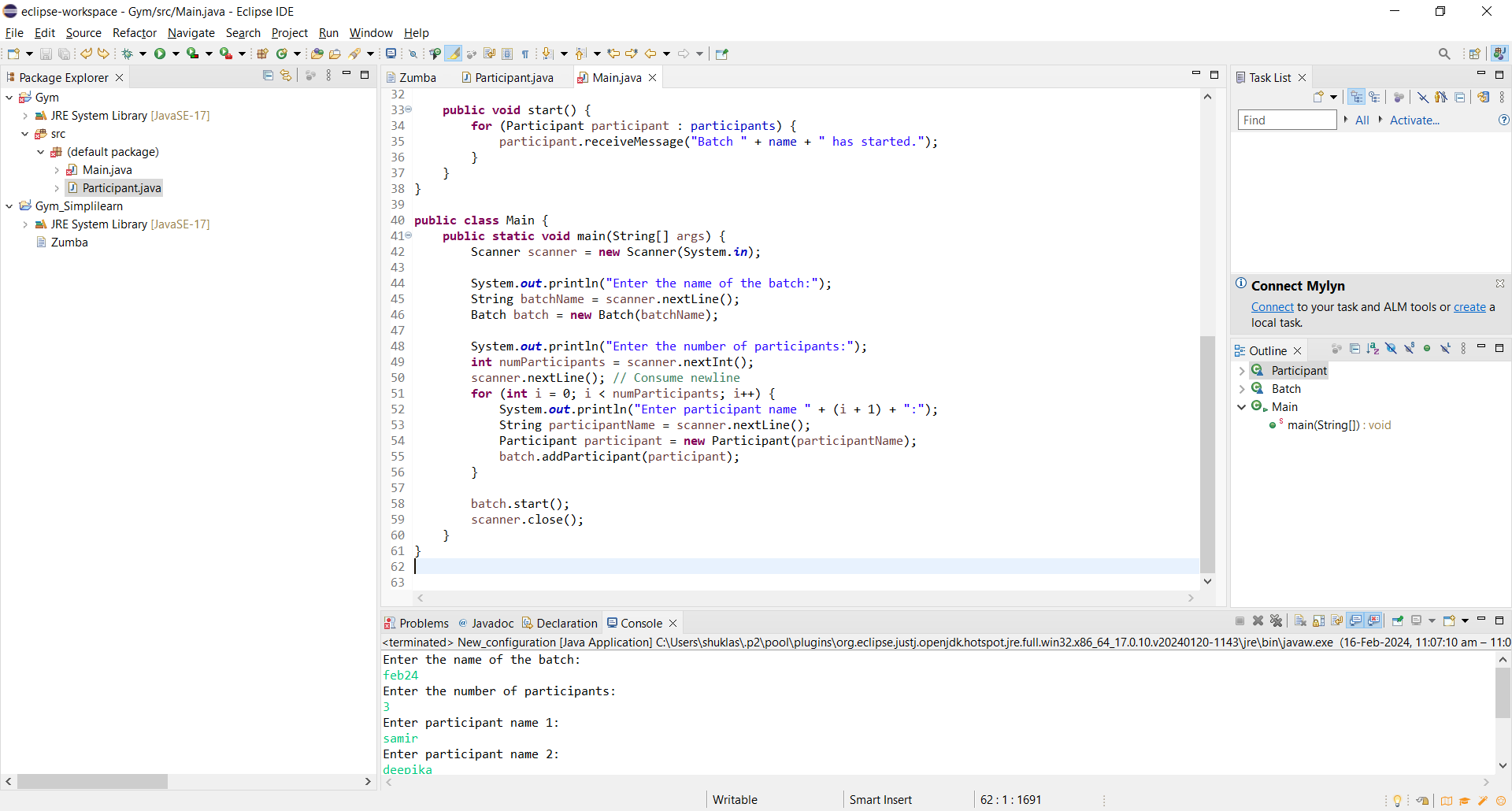
}

}

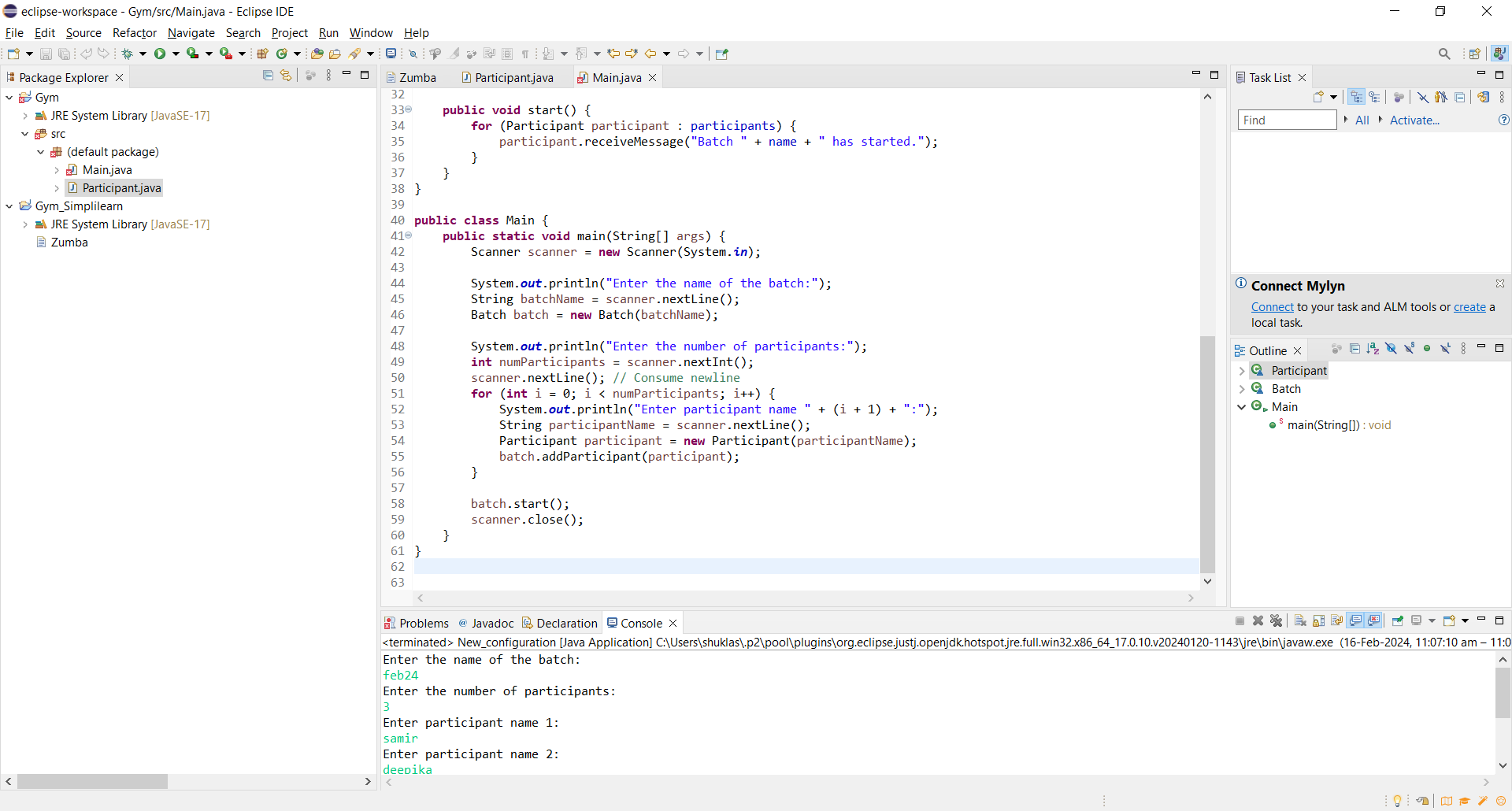
Screenshots

A screenshot of a computer

Description automatically generated



Output



A screenshot of a computer

Description automatically generated