Employee Management Application

PRESENTED BY

NIKHITHA MOGER

SEGANTI MANIKANTA

NOMULA LOUKYA

ABBAGONI RISHITHA

Contents

TEST PLAN

REFERENCE DOCUMENT

CONSOLE PROJECT

AUTOMATION PROJECT

Test Plan

TEST PLAN

Project Name: Employee Management Application

Test Engineer: Manikanta

Date:13/01/23

Prepared by: Manikanta

1)Test Objective or Aim

The objective of the test is to verify the functionalities of **"Add Employee, Show all Employee, Get specific Employee, Delete Employee, Update Employee, Exit works according to the specification."**

2) Scope of testing

a) Within the scope features to be tested

Module	Application Roles	Description
Add Employee	Manager	❖ Manager should able to Add Employee.
Show all Employee	Manager	❖ Manager should able to View All Employee.
Get specific Employee	Manager	❖ Manager should be able Details of specific Employee.
Delete Employee	Manager	❖ Manager can able to delete Employee.
Update Employee	Manager	❖ Manager can able to update the Employee details.

b) Out of scope: These features are not be tested because they are not included in the software requirement specs

- Automation Testing
- Stress Testing
- Performance Testing

3) Test Strategy

a) *Levels of testing*

- System Testing: Testing the system as a whole, including the front-end and back-end functionality.

a) Types of testing

- Functional Testing: Testing the functional requirements and features of the system.

d) Configuration Management tool

- GIT-Code Configuration Management

e) Terminology

- Test plan
- Test Scenarios
- Test Cases
- Defect Log
- RTM

4) Exit and Entry criteria

- Entry Criteria –
  - Test environment ready.
  - Build available for testing.
  - Test data ready to test execution.
  - Test scenarios and test cases are created.
- Exit Criteria –
  - All the test cases are executed.
  - Defect not are of low severity.
  - Schedule didn't get extended.

5) Test deliverables

Before testing	During testing	After testing
Test plan document	Test tool	Test results
Requirements document	Test data	Defect reports
	RTM	

6)Roles and Responsibility

ROLES	Names	RESPONSIBILITIES
Test Engineer	Manikanta, Nikhita	Developed java code(for console and selenium automation project).
Preparing of Test Scenario, cases, defect log, RTM	Rishitha	Writing and executing the test cases & report the defects also identify the test design techniques.
Prepare Test plan	Loukya	Preparing test plan.

7) Risks and mitigation

- Meet outstanding prerequisites
- Redefine test data
- Review test plan and modify components (that is, scripts)
- Restore data and restart

8)Schedule

Task	Members	Estimate effort
Writing java code, database connection and database creation.	Nikhita, Manikanta	10man-hours
Test plan	Loukya	4 man-hour
Test scenario, test cases, defect log and RTM.	Rishitha	5 man-hour
Total		19 man-hour

9) Test Environment

- Database Server (MySQL Workbench) 8gb ram, 150gb hard disk, 3.2ghz.
- Java version “1.8.0_341”.
- Eclipse IDE.

10) Assumptions: Exploratory Testing would be carried out once the build is ready for testing

- Performance testing is not considered for this estimation.
- Test case design activities will be performed by QA Group
- Test environment and preparation activities will be owned by Nikhita Team

- Nikhita team will provide Defect fix plans based on the Defect meetings during each cycle to plan.

12) Approval Information

Project Manager: reviews the content of the Test Plan, Test Strategy and Test Estimates signs off on it.

Test Manager: Reviews the test cases, test Conditions and Test data, test report.

The Names and Titles of all persons who must approve this plan.

Signature:

Name:

Role:

Date:

13)Test Metrics

- Passed test Cases Percentage: (no. of passed test cases/no. of test cases executed) *100
- Failed test cases percentage: (no. of failed test cases/no. of test cases executed) *100
- Fixed defect percentage: (defects fixed / defects reported) *100
- Accepted defect percentage: no. of accepted defects/no. of defected reported) *100
- Defects deferred percentage: (defects deferred/defects reported) *100
- Critical Defects Percentage: (critical defects/total defects reported) *100

Reference Document

Project Name: Employee Management.

Date: 03/03/23

Revision History

Version No	Date	Prepared by / Modified by	Significant Changes
Employee 1.0	25/01/23	Nikhita, Manikanta	-
Employee 2.0	02/02/23	Nikhita, Manikanta	Changes at adding employee & Delete Employee

1.1 Project Overview

The project is a Employee management application that allows users to perform various functions such as adding Employee, viewing all the Employee, viewing specific Employee based on their id, deleting a specific Employee details based on their id and updating a specific Employee details using their id. The application is built using Java and uses MySQL database for storing customer information.

1.2 Scope

The application will allow users to perform the following functions:

- Add new Employee to the system database
- Viewing all the Employee's details in the database of the application
- Viewing a specific Employee details in the database of the application based on their id

- Deleting a specific Employee details from the database of the application based on their id
- Updating a specific Employee details in the database of the application based on their id
- Exiting the application after using the application

1.3 Out of Scope

The application will not include any additional functions such as finding salary of a Employee or changing the gender of Employee.

1.4 Intended Audience

The intended audience for this application is managers who will be responsible for managing Employee information

1.5 High Level Use Cases

- Add new Employee to the system
- Viewing all the Employee's details in the database of the application
- Viewing a specific Employee details in the database of the application based on their id
- Deleting a specific Employee details from the database of the application based on their id
- Updating a specific Employee details in the database of the application based on their id
- Exiting the application after using the application

1.6 Use Cases detailed

- **`Add Employee:`** Allows managers to add a new Employee to the system by entering their personal details.
- **`Show All Employee:`** Allows managers to view all Employee details from the database
- **`Get Employee based on id:`** Allows managers to view a specific Employee details from the database based on the Employee id
- **`Delete Employee:`** Allows managers to delete a specific Employee details from the database based on the Employee id
- **`Update Employee:`** Allows managers to update a specific Employee details in the database based on the Employee id
- **Exit:** Allows managers to exit the application after using.

1.7 User Interface Modules

The application will have a user interface that allows manager to perform the various functions as described in the use cases.

1.8 Technical Architecture

The application is built in Java, JDBC and uses MySQL database for storing employee information.

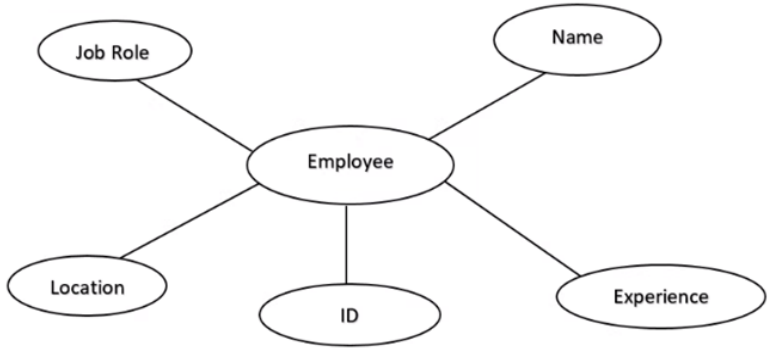
1.9 Technical Detailed Description

The application will use Java as the programming language and will make use of JDBC to connect to the MySQL database. The database will store customer information such as name, Experience, Job Role, Location. The application will also use a user interface to allow managers to perform the various functions described in the use cases.

1.10 Sequence Diagram

A sequence diagram would be a visual representation of the interactions between the different components of the system and the order in which these interactions occur.

1.11 Database Entity Relationship Diagram



1.12 Standards

The application will adhere to industry standards for Java and MySQL development.

1.13 Non Functional Requirements

- **Security:** The application will need to ensure that Employee information is protected and that only authorized users can access it.
- **Performance:** The application will need to respond quickly to user requests in order to provide a good user experience.
- **Scalability:** The application will need to be able to handle an increasing number of Employee details as joined to the database.

End of document ■

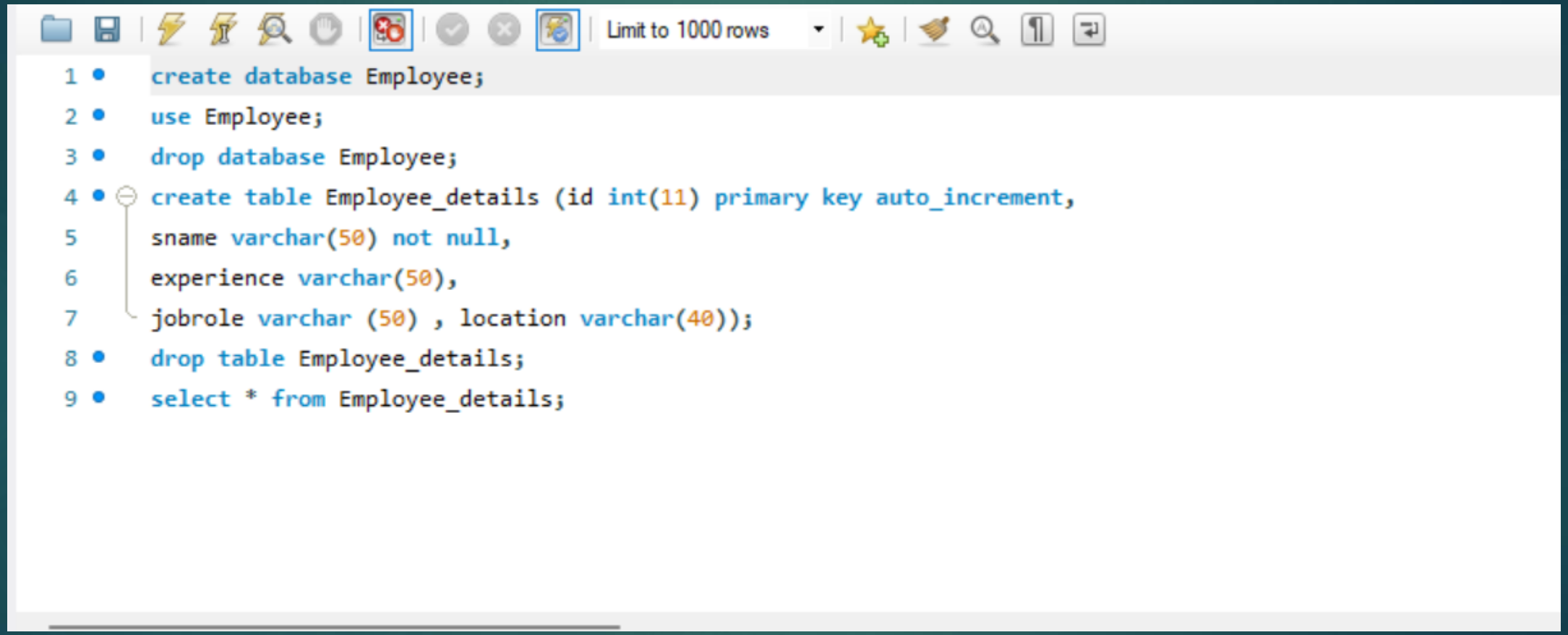
Project Overview

- Employee Management Application consists of Modules-Add Employee, show all Employee, Get the Employee based on the Id, Delete Employee, Update Employee and Exit. Each module runs accordingly as the user want to function the application.
- This application helps the user to make the list of Employees including their details in the database and they will be able to recollect details as required for them and store them Successfully. Maintenance of the application is too easy.

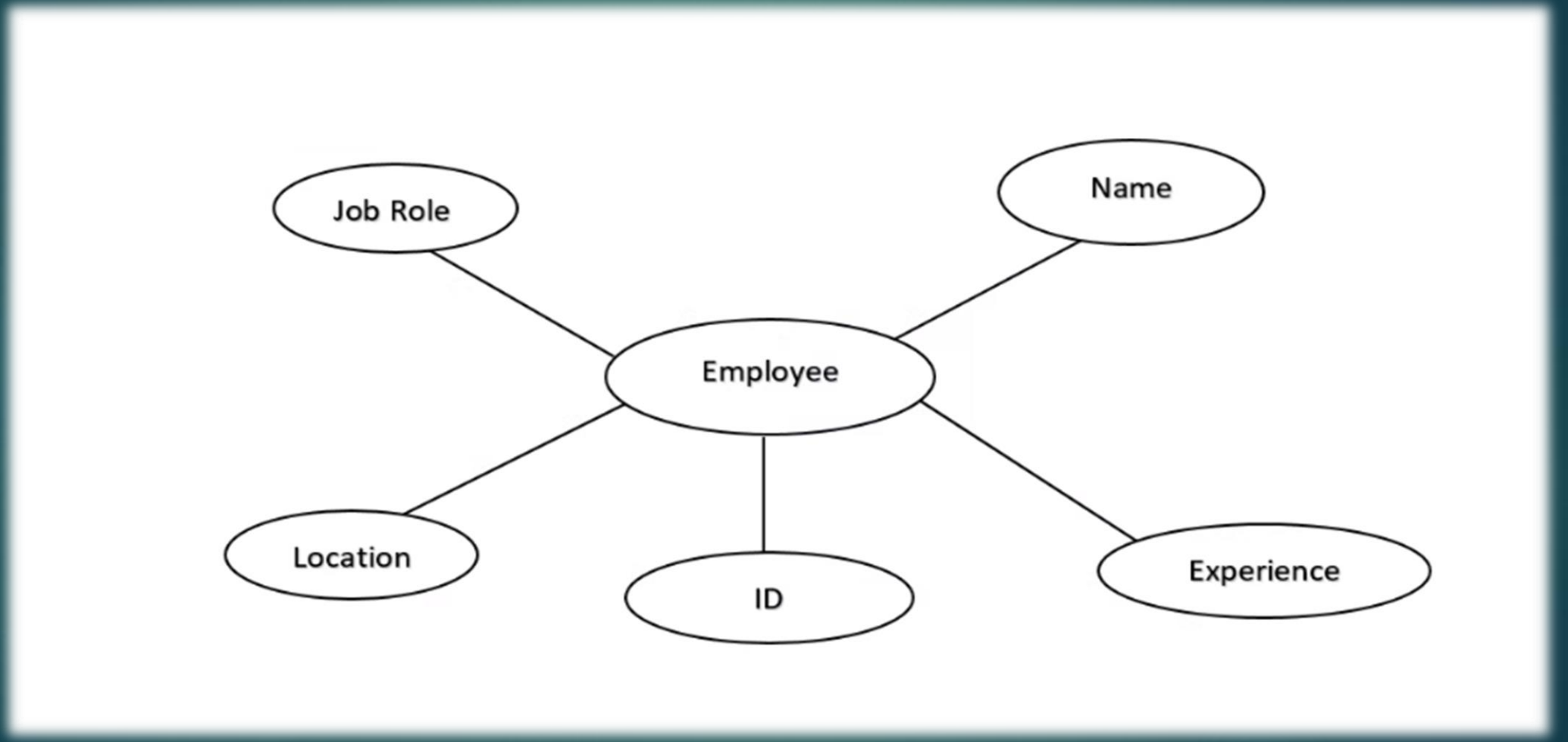
Modules in Employee Management

- Adding the Employee in Application.
- Showing all Employee of Application.
- Getting Employee by their Identity.
- Deletion of Employee details.
- Updating the Employee details.

Application Database



Database Entity Diagram



Source code: Main Java

main.java

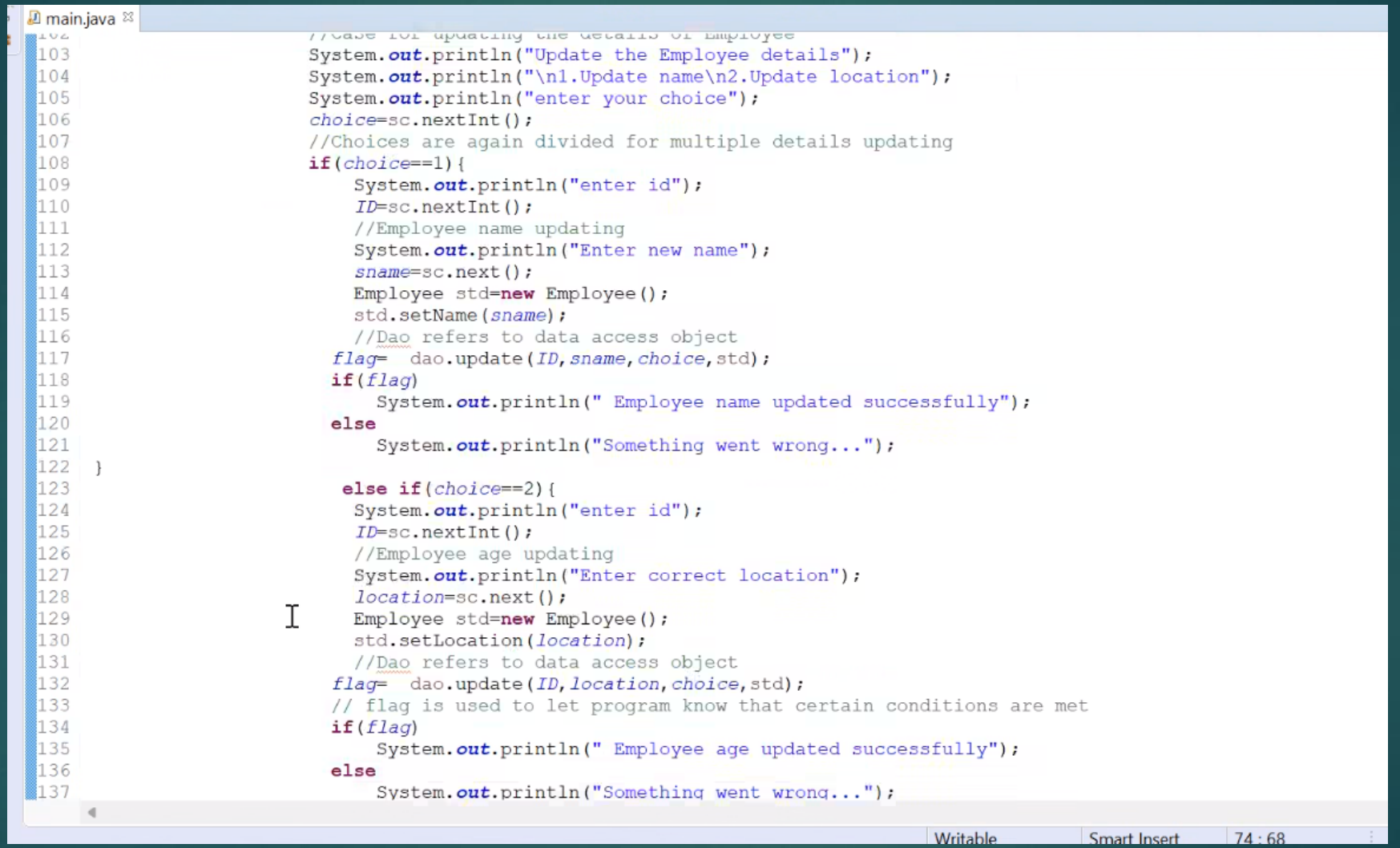
```
1  package main;
2
3  import java.util.Scanner;
4
5  import Employee.dao.EmployeeDao;
6  import Employee.dao.EmployeeDaoInterface;
7  import Employee.model.Employee;
8
9  public class main {
10     private static boolean f;
11     private static boolean ff;
12     private static boolean ans;
13     private static boolean flag;
14     public static int Id;
15     public static int ID;
16     public static int id;
17     public static int st;
18     public static int choice;
19     public static int ch;
20     public static String name;
21     public static String experience;
22     public static String jobrole;
23     public static String location;
24     public static String sname;
25     public static String Experience;
26
27     public static void main(String[] args) {
28         // Scanner class to give console based inputs
29         Scanner sc=new Scanner(System.in);
30
31         //Object creation
32         EmployeeDaoInterface dao=new EmployeeDao();
33         System.out.println("Welcome to Employee Management application");
34         while(true){
35             System.out.println("\n1.Add Employee" +
```

main.java

```
34          while(true){
35              System.out.println("\n1.Add Employee" +
36                      "\n2.Show All Employee" +
37                      "\n3.Get Employee based on id" +
38                      "\n4.Delete Employee"+
39                      "\n5.Update Employee" +
40                      "\n6.Exit");
41              //The main interface of application
42              System.out.println("Enter choice");
43              ch=sc.nextInt();
44              // switch cases for different modules of application
45              switch (ch){
46                  case 1:
47                      System.out.println("Add Employee");  // case for adding the Employee module
48                      System.out.println("Enter Employee name");
49                      name=sc.next();
50                      System.out.println("Enter Employee experience");
51                      experience=sc.next();
52                      System.out.println("Enter jobrole");
53                      jobrole=sc.next();
54                      System.out.println("Enter Location");
55  //                    location=sc.next();
56                      // object creation for Employee
57                      Employee st=new Employee(name,experience,jobrole,location);
58                      //along with parameters
59                      ans=dao.insertEmployee(st);
60                      if(ans)
61                          System.out.println("Employee details inserted Successfully!!!");
62                      else
63                          System.out.println("something went wrong, please try again");
64                      // break statement takes the flow of control out of the switch statement
65                      break;
66                  case 2:
67                      //Case of getting all the Employees from the database
68                      System.out.println("Show all Employee ");
```

main.java

```
69                  dao.showAllEmployee();
70
71                  break;
72              case 3:
73                  //Case of getting particular Employee details
74                  System.out.println("Get Employee based on id");
75                  //via id
76                  System.out.println("enter id");
77                  id=sc.nextInt();
78                 f = dao.showEmployeeById(id);
79                  //Condition if Employee ID is not available in database
80
81                if(f) {
82                    System.out.println("Employee with this id is available in our system");}
83                    else
84                        {System.out.println("Employee with this id is not available in our system");}
85                //break statement takes the flow of control out of switch statement
86                  break;
87              case 4:
88                   // Case for deletion of the students
89                  System.out.println("Delete Employee");
90                    System.out.println("Enter  ID to delete");
91                   Id = sc.nextInt();
92  //                 ff= dao.showEmployeeById(Id);
93                   if (ff) {
94                   ff = dao.delete(Id);
95                   System.out.println("Employee details deleted successfully...");
96                   } else {
97                   System.out.println("Employee with this ID is not available in our system");}
98                   // break statement takes the flow of control out of the switch statement
99                   break;
100
101             case 5:
102                 //Case for updating the details of Employee
103                 System.out.println("Update the Employee details");
```


main.java
System.out.println("Update the Employee details");
System.out.println("\n1.Update name\n2.Update location");
System.out.println("enter your choice");
choice=sc.nextInt();
//Choices are again divided for multiple details updating
if(choice==1){
System.out.println("enter id");
ID=sc.nextInt();
//Employee name updating
System.out.println("Enter new name");
sname=sc.next();
Employee std=new Employee();
std.setName(sname);
//Dao refers to data access object
flag= dao.update(ID,sname,choice,std);
if(flag)
System.out.println(" Employee name updated successfully");
else
System.out.println("Something went wrong...");
}
else if(choice==2){
System.out.println("enter id");
ID=sc.nextInt();
//Employee age updating
System.out.println("Enter correct location");
location=sc.next();
Employee std=new Employee();
std.setLocation(location);
//Dao refers to data access object
flag= dao.update(ID,location,choice,std);
// flag is used to let program know that certain conditions are met
if(flag)
System.out.println(" Employee age updated successfully");
else
System.out.println("Something went wrong...");
Writable
Smart Insert
74 : 68

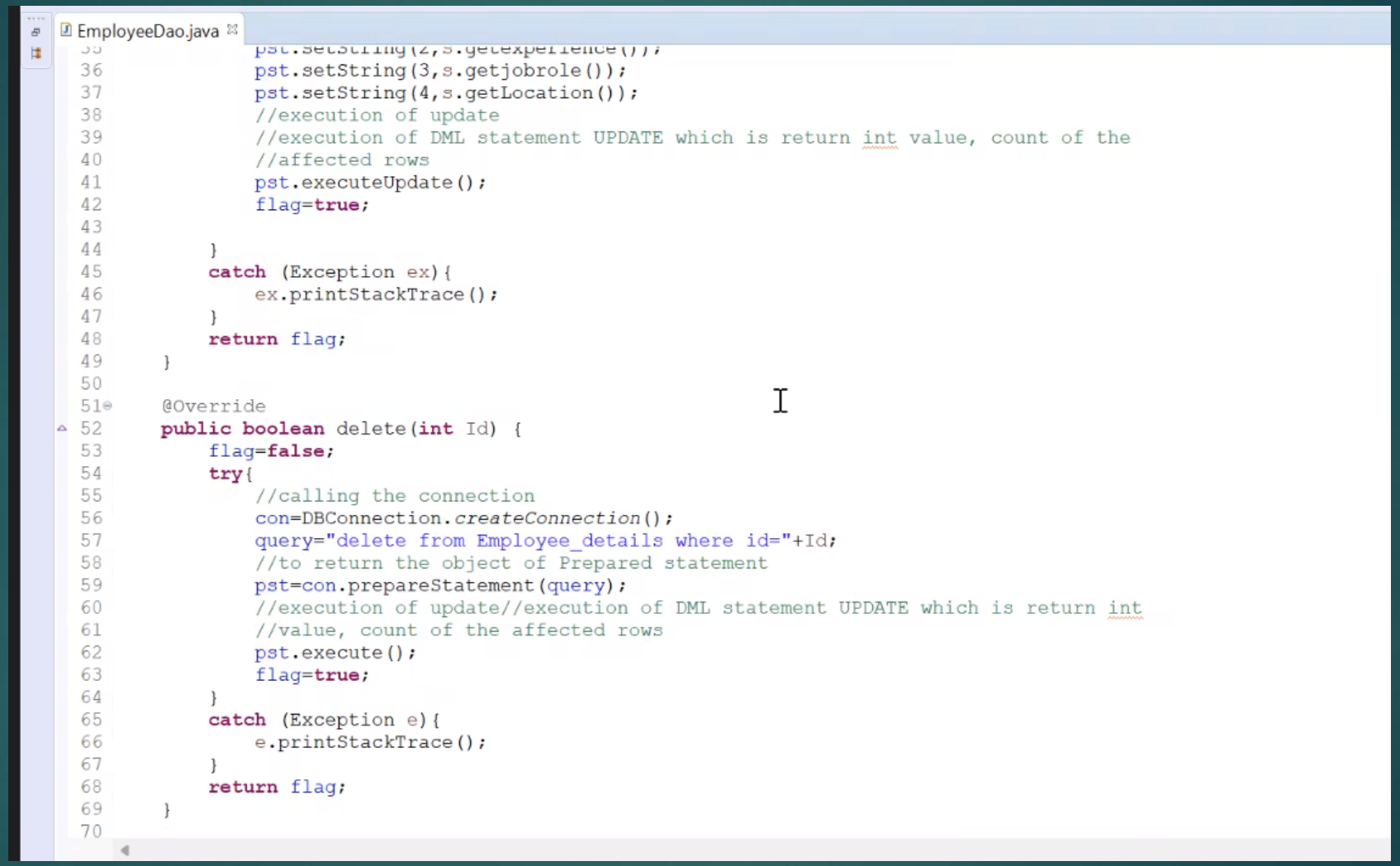
main.java

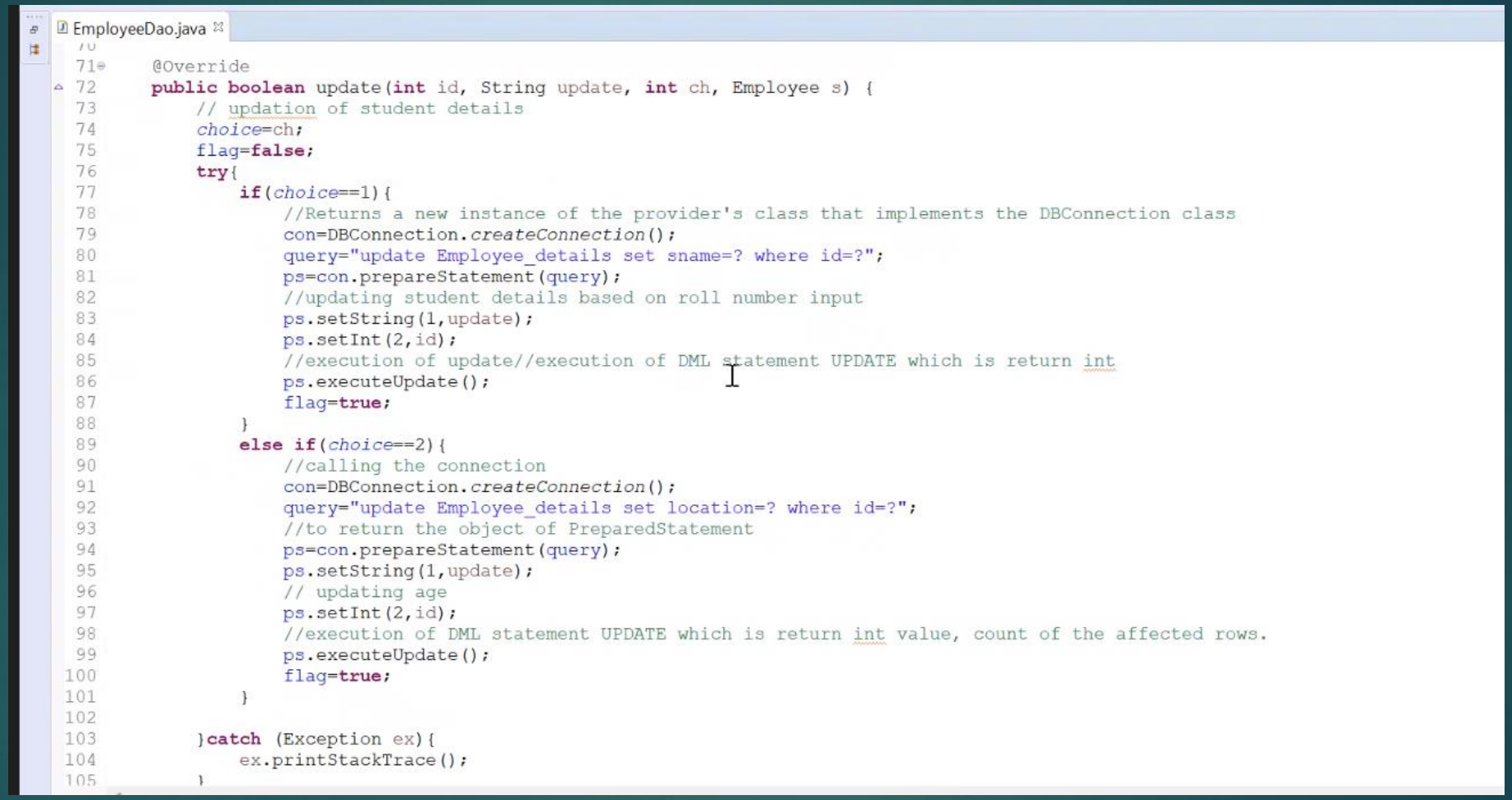
```
                    System.out.println("Something went wrong...");
    }
                 else if(choice==2){
                  System.out.println("enter id");
                  ID=sc.nextInt();
                  //Employee age updating
                  System.out.println("Enter correct location");
                  location=sc.next();
                  Employee std=new Employee();
                  std.setLocation(location);
                  //Dao refers to data access object
                 flag=  dao.update(ID,location,choice,std);
                 // flag is used to let program know that certain conditions are met
                 if(flag)
                     System.out.println(" Employee age updated successfully");
                 else
                     System.out.println("Something went wrong...");
    }
                //break statement takes the flow of control out of switch statement
                break;
            case 6:
                System.out.println("Thank You for using Employee management application!!!");
                //Terminates the current java running machine
                System.exit(0);
                //Exiting the application and terminating the program
            default:
                //to specify the set of statements to execute if
                // there is no case match
                System.out.println("Please enter valid choice....");
        }
    }


}
}
```

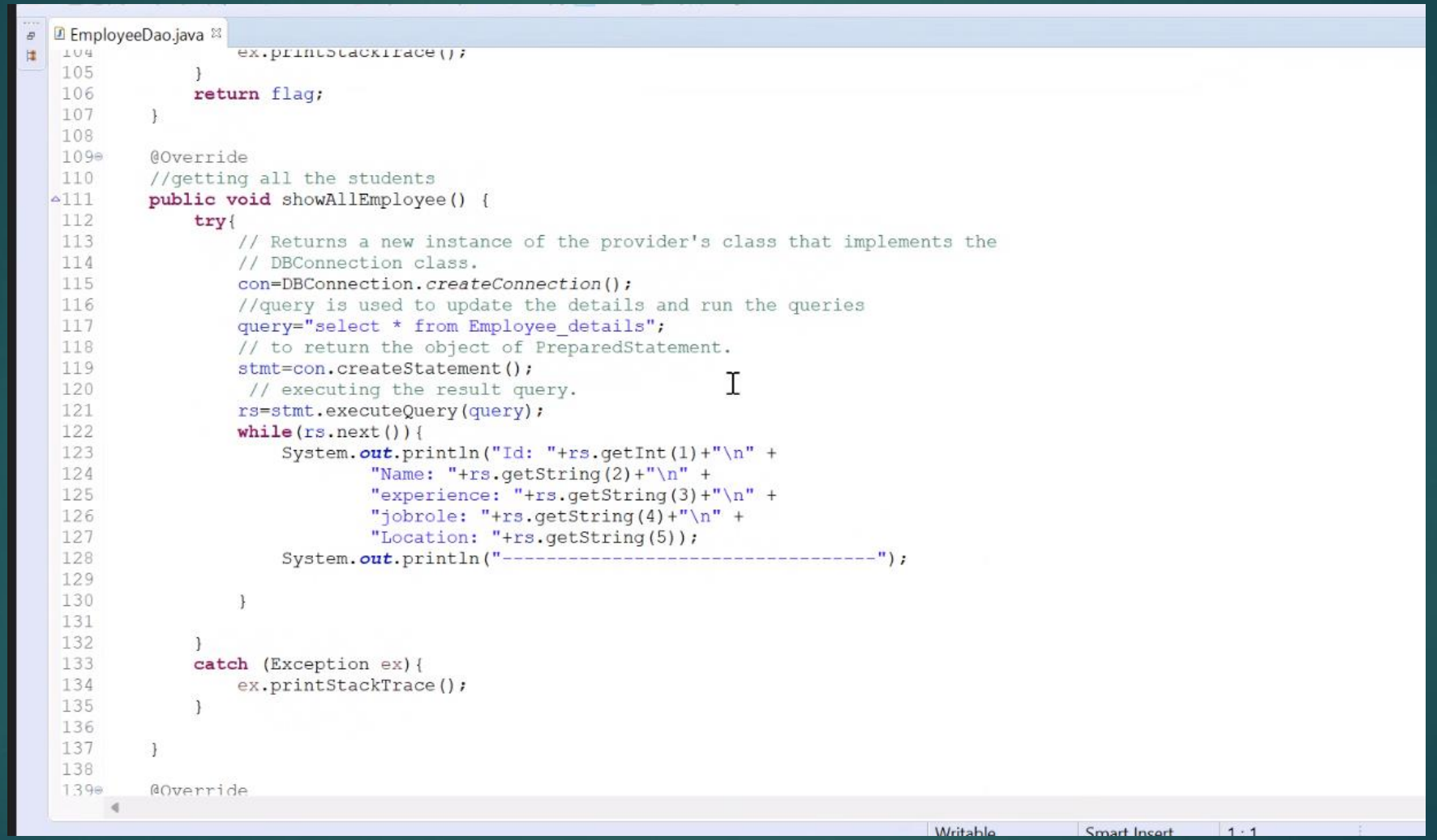
Source Code :EmployeeDao.java

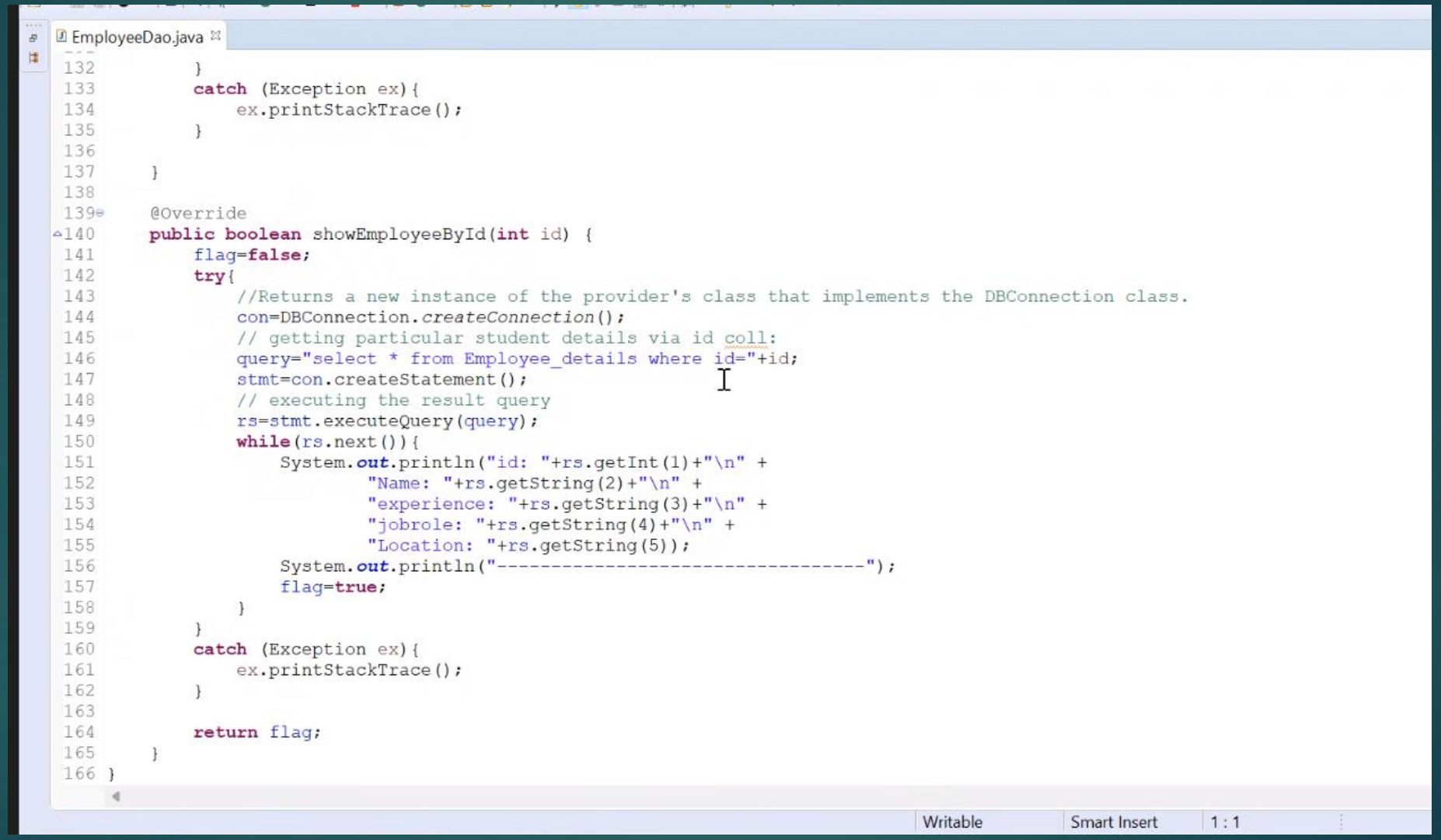
EmployeeDao.java

```
1  package Employee.dao;
2
3  import java.sql.Connection;
4  import java.sql.PreparedStatement;
5  import java.sql.ResultSet;
6  import java.sql.Statement;
7
8  import Employee.db.DBConnection;
9  import Employee.model.Employee;
10
11 public class EmployeeDao implements EmployeeDaoInterface{
12
13     public Statement stmt;
14     public PreparedStatement pst;
15     public PreparedStatement ps;
16     public Connection con;
17     public String query;
18     public ResultSet rs;
19     public boolean flag;
20     public static int choice;
21
22     @Override
23     public boolean insertEmployee(Employee s) {
24         flag=false;
25         try{
26             //returns a new instance of the provider's class that implements the
27             //DBconnection class
28             con= DBConnection.createConnection();
29             //inserting class
30             query="insert into Employee_details(sname,experience,jobrole,location) value(?,?,?,?)";
31             //Prepare statement object provides a feature to execute parameterized
32             //query. to return the object of preparestatement
33             pst=con.prepareStatement(query);
34             pst.setString(1,s.getName());
35             pst.setString(2,s.getexperience());
```


EmployeeDao.java
pst.setString(2,s.getExperience());
pst.setString(3,s.getjobrole());
pst.setString(4,s.getLocation());
//execution of update
//execution of DML statement UPDATE which is return int value, count of the
//affected rows
pst.executeUpdate();
flag=true;
}
catch (Exception ex){
ex.printStackTrace();
}
return flag;
}
@Override
public boolean delete(int Id) {
flag=false;
try{
//calling the connection
con=DBConnection.createConnection();
query="delete from Employee_details where id="+Id;
//to return the object of Prepared statement
pst=con.prepareStatement(query);
//execution of update//execution of DML statement UPDATE which is return int
//value, count of the affected rows
pst.execute();
flag=true;
}
catch (Exception e){
e.printStackTrace();
}
return flag;
}


EmployeeDao.java
@Override
public boolean update(int id, String update, int ch, Employee s) {
// updation of student details
choice=ch;
flag=false;
try{
if(choice==1){
//Returns a new instance of the provider's class that implements the DBConnection class
con=DBConnection.createConnection();
query="update Employee_details set sname=? where id=?";
ps=con.prepareStatement(query);
//updating student details based on roll number input
ps.setString(1,update);
ps.setInt(2,id);
//execution of update//execution of DML statement UPDATE which is return int
ps.executeUpdate();
flag=true;
}
else if(choice==2){
//calling the connection
con=DBConnection.createConnection();
query="update Employee_details set location=? where id=?";
//to return the object of PreparedStatement
ps=con.prepareStatement(query);
ps.setString(1,update);
// updating age
ps.setInt(2,id);
//execution of DML statement UPDATE which is return int value, count of the affected rows.
ps.executeUpdate();
flag=true;
}
}catch (Exception ex){
ex.printStackTrace();
}


EmployeeDao.java
ex.printStackTrace();
}
return flag;
}
@Override
//getting all the students
public void showAllEmployee() {
try{
// Returns a new instance of the provider's class that implements the
// DBConnection class.
con=DBConnection.createConnection();
//query is used to update the details and run the queries
query="select * from Employee_details";
// to return the object of PreparedStatement.
stmt=con.createStatement();
// executing the result query.
rs=stmt.executeQuery(query);
while(rs.next()){
System.out.println("Id: "+rs.getInt(1)+"\n" +
"Name: "+rs.getString(2)+"\n" +
"experience: "+rs.getString(3)+"\n" +
"jobrole: "+rs.getString(4)+"\n" +
"Location: "+rs.getString(5));
System.out.println("----------------------------------");
}
}
catch (Exception ex){
ex.printStackTrace();
}
}
@Override
Writable
Smart Insert
1 : 1


EmployeeDao.java
132 }
133 catch (Exception ex){
134 ex.printStackTrace();
135 }
136
137 }
138
139 @Override
140 public boolean showEmployeeById(int id) {
141 flag=false;
142 try{
143 //Returns a new instance of the provider's class that implements the DBConnection class.
144 con=DBConnection.createConnection();
145 // getting particular student details via id coll:
146 query="select * from Employee_details where id="+id;
147 stmt=con.createStatement();
148 // executing the result query
149 rs=stmt.executeQuery(query);
150 while(rs.next()){
151 System.out.println("id: "+rs.getInt(1)+"\n" +
152 "Name: "+rs.getString(2)+"\n" +
153 "experience: "+rs.getString(3)+"\n" +
154 "jobrole: "+rs.getString(4)+"\n" +
155 "Location: "+rs.getString(5));
156 System.out.println("----------");
157 flag=true;
158 }
159 }
160 catch (Exception ex){
161 ex.printStackTrace();
162 }
163
164 return flag;
165 }
166 }
Writable
Smart Insert
1 : 1

Source Code :EmployeeDaoInterface.java

EmployeeDaoInterface.java

```java
package Employee.dao;

import Employee.model.Employee;

public interface EmployeeDaoInterface {
    // specifies the behavior of a class by providing an abstract type
    /* Interfaces are used in Java to achieve abstraction. (A process of hiding the
    implementation details and showing only functionality to the user)*/
    public boolean insertEmployee(Employee s);
    public boolean delete(int id);
    public boolean update(int id,String update,int ch,Employee s);
    public void showAllEmployee();
    public boolean showEmployeeById(int id);
}
```

Source Code: DBConnection.java

DBConnection.java

```java
package Employee.db;

import java.sql.Connection;
import java.sql.DriverManager;

public class DBConnection {
    //global declaration of variable
    static Connection con;

    public static Connection createConnection(){

        try{
            //used to get instance of the class with specified class name
            Class.forName("com.mysql.jdbc.Driver");
            //username of mysql
            String user="root";
            //password of mysql
            String pass="Mani1007";
            //provides a way identifying a database so that the appropriate driver
            //recognizes it and connects to it
            String url = "jdbc:mysql://localhost:3306/Employee?autoReconnect=true&useSSL=false";
            //Driver manager class attempts to establish a connection to the database
            con= DriverManager.getConnection(url,user,pass);

        }
        catch(Exception ex){
            ex.printStackTrace();
        }

        return con;
    }

}
```

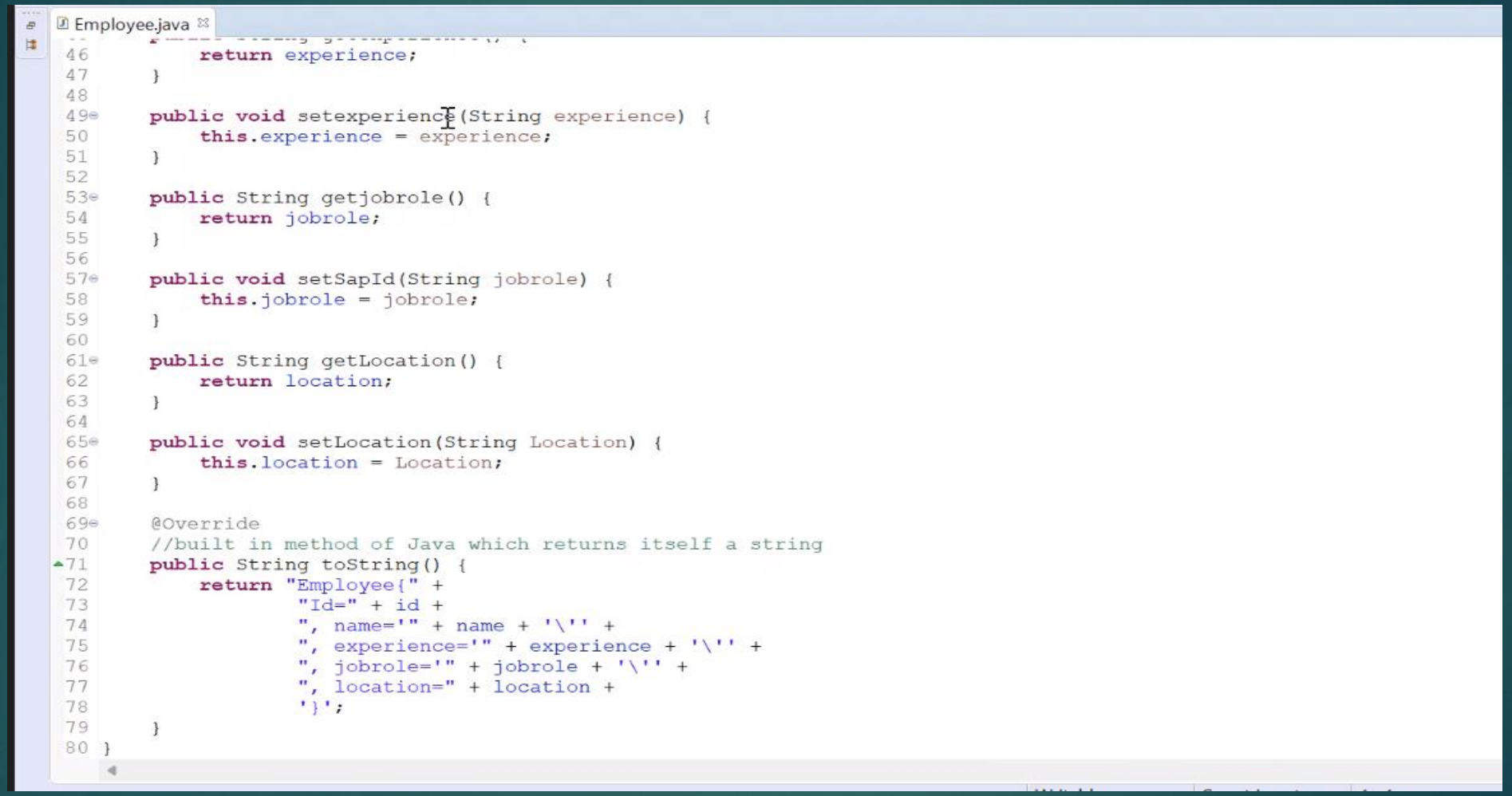
Source Code : Employee.java

Employee.java

```java
package Employee.model;

public class Employee {
    private int id;
    private String name;
    private String experience;
    private String jobrole;
    private String location;

    public Employee(){

    }

    public Employee(String name, String experience, String jobrole, String location) {
        this.name = name;
        this.experience = experience;
        this.jobrole = jobrole;
        this.location = location;
    }

    public Employee(int id, String name, String experience, String jobrole, String location) {
        this.id = id;
        this.name = name;
        this.experience = experience;
        this.jobrole = jobrole;
        this.location = location;
    }
    //getters and setter of constructor
    public int getid() {
        return id;
    }

    public void setid(int id) {
        this.id = id;
    }
```

Employee.java

```
35      }
36
37      public String getName() {
38          return name;
39      }
40
41      public void setName(String name) {
42          this.name = name;
43      }
44
45      public String getexperience() {
46          return experience;
47      }
48
49      public void setexperience(String experience) {
50          this.experience = experience;
51      }
52
53      public String getjobrole() {
54          return jobrole;
55      }
56
57      public void setSapId(String jobrole) {
58          this.jobrole = jobrole;
59      }
60
61      public String getLocation() {
62          return location;
63      }
64
65      public void setLocation(String Location) {
66          this.location = Location;
67      }
68
69      @Override
```


Employee.java
46 return experience;
47 }
48
49 public void setexperience(String experience) {
50 this.experience = experience;
51 }
52
53 public String getjobrole() {
54 return jobrole;
55 }
56
57 public void setSapId(String jobrole) {
58 this.jobrole = jobrole;
59 }
60
61 public String getLocation() {
62 return location;
63 }
64
65 public void setLocation(String Location) {
66 this.location = Location;
67 }
68
69 @Override
70 //built in method of Java which returns itself a string
71 public String toString() {
72 return "Employee{" +
73 "Id=" + id +
74 ", name='" + name + '\'' +
75 ", experience='" + experience + '\'' +
76 ", jobrole='" + jobrole + '\'' +
77 ", location=" + location +
78 '}';
79 }
80 }

Console View: Adding Employee

Entering Valid Details of module:

```
1.Add Employee
2.Show All Employee
3.Get Employee based on id
4.Delete Employee
5.Update Employee
6.Exit
Enter choice
1
Add Employee
Enter Employee name
Manikanta
Enter Employee experience
8
Enter jobrole
Tester
Enter Location
chennai
Loading class `com.mysql.jdbc.Driver'. This is deprecated. The new
Employee details inserted Successfully!!!

1.Add Employee
2.Show All Employee
3.Get Employee based on id
4.Delete Employee
5.Update Employee
6.Exit
Enter choice
```

Entering Invalid choice of module:

```
1.Add Employee
2.Show All Employee
3.Get Employee based on id
4.Delete Employee
5.Update Employee
6.Exit
Enter choice
1
Add Employee
Enter Employee name
Manikanta
Enter Employee experience
8
Enter jobrole
Tester
Enter Location
Loading class `com.mysql.jdbc.Driver'. This is deprecated. The new
Employee details inserted Successfully!!!
```

Console View: Show All Employees

All Employees details are shown

```
1.Add Employee
2.Show All Employee
3.Get Employee based on id
4.Delete Employee
5.Update Employee
6.Exit
Enter choice
2
Show all Employee
Id: 1
Name: abc
experience: 13
jobrole: tester
Location: chennai
-----------------------------------
Id: 4
Name: 2
experience: 12
jobrole: tester
Location: channai
-----------------------------------
Id: 5
Name: 2
experience: 12
jobrole: tester
Location: null
-----------------------------------
Id: 6
Name: abc
```

Console View: Get Employee based on Id

Entering Valid id :

```
1.Add Employee
2.Show All Employee
3.Get Employee based on id
4.Delete Employee
5.Update Employee
6.Exit
Enter choice
3
Get Employee based on id
enter id
17
id: 17
Name: Manikanta
experience: 8
jobrole: Tester
Location: null
---------------------------------
Employee with this id is available in our system

1.Add Employee
2.Show All Employee
3.Get Employee based on id
4.Delete Employee
5.Update Employee
6.Exit
Enter choice
```

Entering Invalid id:

Task List | Outline | Console

Main [Java Application] C:\Program Files\Java\jre1.8.0_351\bin\javaw.exe (Feb 18, 2023, 3:43:29 PM

```
6.Exit
Enter choice

3
Get Employee based on id
enter id
143
Employee with this id is not available in our system

1.Add Employee
2.Show All Employee
3.Get Employee based on id
4.Delete Employee
5.Update Employee
6.Exit
Enter choice
4
Delete Employee
Enter  ID to delete
17
Employee with this ID is not available in our system
```

Console View: Delete Employee based on Id

Deleting Employee by valid id :

```
1.Add Employee
2.Show All Employee
3.Get Employee based on id
4.Delete Employee
5.Update Employee
6.Exit
Enter choice
5
Update the Employee details

1.Update name
2.Update location
enter your choice
4

1.Add Employee
2.Show All Employee
3.Get Employee based on id
4.Delete Employee
5.Update Employee
6.Exit
Enter choice
```

Deleting Employee by Invalid id :

```
Enter choice
4
Delete Employee
Enter  ID to delete
17
Employee with this ID is not available in our system

1.Add Employee
2.Show All Employee
3.Get Employee based on id
4.Delete Employee
5.Update Employee
6.Exit
Enter choice
4
Delete Employee
Enter  ID to delete
143
Employee with this ID is not available in our system

1.Add Employee
2.Show All Employee
3.Get Employee based on id
4.Delete Employee
5.Update Employee
6.Exit
Enter choice
```

Console View: Update Employee based on Id

Updating Employee by Valid choice:

```
1.Add Employee
2.Show All Employee
3.Get Employee based on id
4.Delete Employee
5.Update Employee
6.Exit
Enter choice
5
Update the Employee details

1.Update name
2.Update location
enter your choice
1
enter id
16
Enter new name
Rishitha
Employee name updated successfully

1.Add Employee
2.Show All Employee
3.Get Employee based on id
4.Delete Employee
5.Update Employee
6.Exit
Enter choice
```

Updating Employee by Invalid choice:

```
1.Add Employee
2.Show All Employee
3.Get Employee based on id
4.Delete Employee
5.Update Employee
6.Exit
Enter choice
5
Update the Employee details

1.Update name
2.Update location
enter your choice
4

1.Add Employee
2.Show All Employee
3.Get Employee based on id
4.Delete Employee
5.Update Employee
6.Exit
Enter choice
```

Console View: Exit

Exit Command by Valid choice:

```
1.Add Employee
2.Show All Employee
3.Get Employee based on id
4.Delete Employee
5.Update Employee
6.Exit
Enter choice
6
Thank You for using Employee management application!!!
```

Exit Command by Invalid choice:

```
Welcome to Employee Management application

1.Add Employee
2.Show All Employee
3.Get Employee based on id
4.Delete Employee
5.Update Employee
6.Exit
Enter choice
7
Please enter valid choice....

1.Add Employee
2.Show All Employee
3.Get Employee based on id
4.Delete Employee
5.Update Employee
6.Exit
Enter choice
```

Excel View of Cover Page

	Employee Management Application		
	Prepared By	Reviewed By	Approved By
Name	Team 5	Saba Rauf	Nagesgwar Rao
Role	Test Engineer	Senior Test	Test Manager
Date	2/18/2023		
Signature	xxxx		

Excel View of Test Scenario

Employee Management Application					
Module Name	Scenario ID	Scenario Name	Scenario Description	Reference Document	Requirement ID
Adding the Employee	EM_01	Employee Management Application - VALID (Entering valid data)	To verify Employee Management Application- Adding the Employee details to the database with valid input.	Employee Management Requirement Document	R01
	EM_02	Employee Management Application - INVALID (Entering invalid data)	To verify proper error message is displayed in Employee Management Application while Adding the Employee with invalid inputs.	Employee Management Requirement Document	R01
Show All Employee	EM_03	Employee Management Application - VALID (Entering valid data)	To verify Employee Management Application- Viewing all the Employee by Show all Employee with valid input.	Employee Management Requirement Document	R02
Get Specific Employee	EM_04	Employee Management Application - VALID (Entering valid data)	To verify Employee Management Application- Get Specific Employee from database with valid input.	Employee Management Requirement Document	R03
	EM_05	Employee Management Application - INVALID (Entering invalid data)	To verify proper error messageis displayed in Employee Management Application while getting Specific Employee from database with invalid input.	Employee Management Requirement Document	R03
Delete Employee	EM_06	Employee Management Application - VALID (Entering valid data)	To verify Employee Management Application- Deleting Employee from database with valid input.	Employee Management Requirement Document	R04
	EM_07	Employee Management Application - INVALID (Entering invalid data)	To verify proper error message is displayed in Employee Management Application- Deleting Employee from database with invalid input.	Employee Management Requirement Document	R04
Update Employee	EM_08	Employee Management Application - VALID (Entering valid data)	To verify Employee Management Application- Updating Employee with valid input.	Employee Management Requirement Document	R05
	EM_09	Employee Management Application - INVALID (Entering invalid data) data(Name,location)	To verify proper error message is displayed in Employee Management Application- Updating Employee with invalid input.	Employee Management Requirement Document	R05
Exit	SM_10	Employee Management Application - VALID (Entering valid data)	To verify Employee Management Application- Exit functionality with valid input.	Employee Management Requirement Document	R06
	SM_11	Employee Management Application - INVALID (Entering invalid data)	To verify proper error message is displayed in Employee Management Application- Exit functionality with invalid input.	Employee Management Requirement Document	R06

Excel View of Test Cases

Test Case_ID	Test Description	Pre-requisitte	Test Data Parameters	Steps to execute	Test Data	Expected Result	Actual Result	Test Result	Defect_Id
Test_Case_001	Validate Adding Employee with Valid data	User should open Console Project Application in Eclipse IDE	1)Enter <Click> 2)Add Employee<Click> 3)Enter Employee Name<Text> 4)Enter Employee Experience<Text> 5)Enter Jobrole<Text> 6)Enter Location<Text>	1)Open Eclipse 2)Select Console main Class 3)Run the code 4)Select the choice add Employee 5)Enter Employee name 6)Enter Employee experience 7)Enter jobrole 8)Enter Location 9)Click on enter	1)Vicky 2)3 3)Tester 4)Hyderabad	1)Eclipse should be opened 2)Console project main class should be opened 3)Code should Run 4)Add Employee option should be implemented after selecting. 5)User should able to give his/her name 6)User should able to give his/her experience 7)User should able to give his/her jobrole 8)User should able to give his/her Location 9)User should click Enter 10)User should get Successfully added Employee details	1)Eclipse is opened 2)Console project main class is opened 3)Code is Running 4)Add Employee option implemented after selecting. 5)User is able to give his/her name 6)User is able to give his/her experience 7)User able to give his/her jobrole **8)User not able to give his/her Location** 9)User clicks Enter 10)User has got "Successfully added Employee details"	Fail	D_01
Test_Case_002	Validate Adding Employee with Invalid data	User should open Console Project Application in Eclipse IDE	1)Enter <Click> 2)Add Employee<Click> 3)Enter Employee Name<Text> 4)Enter Employee Experience<Text> 5)Enter Jobrole<Text> 6)Enter Location<Text>	1)Open Eclipse 2)Select Console main Class 3)Run the code 4)Select the choice add scholar 5)Enter Scholar name 6)Enter Employee experience 7)Enter jobrole 8)Enter Location 9)Click on enter	1)Vicky 2)3 3)Tester 4)Hyderabad	1)Eclipse should be opened 2)Console project main class should be opened 3)Code should Run 4)Add Employee option should be implemented after selecting. 5)User should able to give his/her name 6)User should able to give his/her experience 7)User should able to give his/her jobrole 8)User should able to give his/her Location 9)User should click Enter 10)User should get Error message	1)Eclipse is opened 2)Console project main class is opened 3)Code is Running 4)Add Employee option implemented after selecting. 5)User is able to give his/her name 6)User is able to give his/her experience 7)User is able to give his/her jobrole **8)User not able to give his/her Location** 9)User clicks Enter 10)User has got error message	Fail	D_02
Test_Case_003	Validate Show all Employee	User should open Console Project Application in Eclipse IDE	1)Enter <Click> 2)Show all Employee<click>	1)Open Eclipse 2)Select Console main Class 3)Run the code 4)Select choice Show all Employee	1)2	1)Eclipse should be opened 2)Console project main class should be opened 3)Code should Run 4)Show All Employee option should be implemented after selecting. 5)Enter option number 6)User should click Enter 7)It should display all the Employee available in the database	1)Eclipse is opened 2)Console project main class is opened 3)Code is Running 4)Show All Employee option should be implemente after selecting. 5)User clicks Enter 6)Displayed Employee details that are available in the database	Pass	
Test_Case_004	Validate Get Specific Employee with valid data	User should open Console Project Application in Eclipse IDE	1)Enter <Click> 2)Get specific Employee<select> Enter Id<Text>	1)Open Eclipse 2)Select Console main Class 3)Run the code 4)Select choice Get Specific Employee 5)Enter Id	1)5	1)Eclipse should be opened 2)Console project main class should be opened 3)Code should Run 4)Get Specific Employee option should be implemented 5)User should Enter Id 6)User should click Enter 7)User should get info of entered Id	1)Eclipse is opened 2)Console project main class is opened 3)Code is Running 4)Get Specific Employee option is implemented 5)User Entered Id 6)User should click Enter 7)Employee details based on ID shown	Pass	

Excel View of Test Cases

38	Test_Case_005	Validate Get Specific Employee with invalid data	User should open Console Project Application in Eclipse IDE	1)Enter <Click> 2)Get specific Employee<select> Enter Id<Text>	1)Open Eclipse 2)Select Console main Class 3)Run the code 4)Select choice Get Specific Employee 5)Enter Id	1)77	1)Eclipse should be opened 2)Console project main class should be opened 3)Code should Run 4)Get Specific Employee option should be implemented 5)User should Enter Id 6)User should click Enter 7)User should get error message saying "Employee with this id is not available in our system"	1)Eclipse is opened 2)Console project main class is opened 3)Code is Running 4)Get Specific Employee option is implemented 5)User Entered Id 6)User should click Enter 7))User has got error message "Employee with this id is not available in our system".	Pass
47	Test_Case_006	Validate Delete Employee with valid data	User should open Console Project Application in Eclipse IDE	1)Enter <Click> 2)Delete Employee<select> Enter Id<Text>	1)Open Eclipse 2)Select Console main Class 3)Run the code 4)Select choice Delete 5)Enter Id	1)3	1)Eclipse should be opened 2)Console project main class should be opened 3)Code should Run 4)Delete Employee option should be implemented 5)User should Enter Id 6)User should click Enter 7)User should get message "Employee details deleted successfully"	1)Eclipse is opened 2)Console project main class is opened 3)Code is Running 4)Delete Employee option is implemented 5)User Entered Id 6)User clicks Enter **7)User got message "Employee details not found"**	Fail
56	Test_Case_007	Validate Delete Employee with invalid data	User should open Console Project Application in Eclipse IDE	1)Enter <Click> 2)Delete Employee<select> Enter Id<Text>	1)Open Eclipse 2)Select Console main Class 3)Run the code 4)Select choice Delete 5)Enter Id	1)77	1)Eclipse should be opened 2)Console project main class should be opened 3)Code should Run 4)Delete Employee option should be implemented 5)User should Enter Id 6)User should click Enter 7)User should get message "Employee with this ID is not available"	1)Eclipse is opened 2)Console project main class is opened 3)Code is Running 4)Delete scholar option is implemented 5)User Entered Id 6)User clicks Enter 7)User got message "Employee details not found"	pass
65	Test_Case_008	Validate update Employee with valid data	User should open Console Project Application in Eclipse IDE	1)Enter <Click> 2)Update Employee<select> 3)Update name<Text> 4)Update location<Text>	1)Open Eclipse 2)Select Console main Class 3)Run the code 4)Select choice Update 5)Select option 1 Name	1)Vicky	1)Eclipse should be opened 2)Console project main class should be opened 3)Code should Run 4)Update Employee option should be implemented 5)User should Select option 1 6)User should enter new name 7)User should get message "Employee Name updated successfully"	1)Eclipse is opened 2)Console project main class is opened 3)Code is Running 4)Update Employee option is implemented 5)User Selected option 1 6)User entered new name 7)User got message "Employee Name updated successfully"	Pass
74	Test_Case_009	Validate update Employee with invalid data	User should open Console Project Application in Eclipse IDE	1)Enter <Click> 2)Update Employee<select> 3)Update name<Text>	1)Open Eclipse 2)Select Console main Class 3)Run the code 4)Select choice Update 5)Select option 1 Name	1)vicky vicky	1)Eclipse should be opened 2)Console project main class should be opened 3)Code should Run 4)Update Employee option should be implemented 5)User should Select option 1 6)User should enter new name 7)User should get message "Something went wrong..."	1)Eclipse is opened 2)Console project main class is opened 3)Code is Running 4)Update Employee option is implemented 5)User Selected option 1 6)User entered new name 7)User got message "Something went wrong..."	Pass

Excel View of Test Cases

	B	C	D	E	F	G	H	I	J
83	Test_Case_010	Validate update Employee with valid data	User should open Console Project Application in Eclipse IDE	1)Enter <Click> 2)Update Employee<select> 3)Update location<Text>	1)Open Eclipse 2)Select Console main Class 3)Run the code 4)Select choice Update 5)Select option 2 location	1)Hyderabad	1)Eclipse should be opened 2)Console project main class should be opened 3)Code should Run 4)Update scholar option should be implemented 5)User should Select option 2 6)User should enter new Location 7)User should get message "Employee Location updated successfully"	1)Eclipse is opened 2)Console project main class is opened 3)Code is Running 4)Update Employee option is implemented 5)User Selected option 2 6)User entered new Location 7)User got message "Employee Location updated successfully"	Pass
92	Test_Case_011	Validate update Employee with valid data	User should open Console Project Application in Eclipse IDE	1)Enter <Click> 2)Update Employee<select> 3)Update location<Text>	1)Open Eclipse 2)Select Console main Class 3)Run the code 4)Select choice Update 5)Select option 2 location	1)dghdg	1)Eclipse should be opened 2)Console project main class should be opened 3)Code should Run 4)Update Employee option should be implemented 5)User should Select option 2 6)User should enter new Location 7)User should get message "Something went wrong..."	1)Eclipse is opened 2)Console project main class is opened 3)Code is Running 4)Update Employee option is implemented 5)User Selected option 2 6)User entered new Location 7)User got message "Something went wrong..."	Pass
101	Test_Case_012	Validate Exit with valid data	User should open Console Project Application in Eclipse IDE	1)Enter <Click> 2)Exit<select>	1)Open Eclipse 2)Select Console main Class 3)Run the code 4)Select choice Exit	1)6	1)Eclipse should be opened 2)Console project main class should be opened 3)Code should Run 4)Exit option should be implemented 5)User should Select Exit 6)User should get message "Thank You for using Employee management application!!!"	1)Eclipse is opened 2)Console project main class is opened 3)Code is Running 4)Exit option is implemented 5)User Selected Exit 6)User should get message "Thank You for using Employee management application!!!"	Pass
110	Test_Case_013	Validate Exit with invalid data	User should open Console Project Application in Eclipse IDE	1)Enter <Click> 2)Exit<select>	1)Open Eclipse 2)Select Console main Class 3)Run the code 4)Select choice Exit	1)7	1)Eclipse should be opened 2)Console project main class should be opened 3)Code should Run 4)Exit option should be implemented 5)User should Select Exit 6)User should get message "Please enter valid choice...."	1)Eclipse is opened 2)Console project main class is opened 3)Code is Running 4)Exit option is implemented 5)User Selected Exit 6)User should get message "Please enter valid choice...."	Pass

Excel View of Defect Log

1	Employee Management Application																
2	Defect_ID	Test Case I	Re-Produc	Defect Description	Defect In cycle	Defect in version	Severity	Priority	Steps to reproduce	Defect Status	Reported By	Reported Date	Phase of origin	DefectType	Cause of Defect	Resolution	Resolved By	Resolved Date	Remarks
3	D_01	TC_001	Always	The user tries to Add the details of Employee.(Valid inputs) **Expected Result:**Application should take all the details of Employee and displaye the details. **Actual Result:**Application didn't took employee's location details. **Test Data:** Name:Vicky experience: jobrole:	Test Cycle_1	1	High	High	Open the Eclipse IDE Run the project as Java Application Enter the Get Employee based on Id Enter Valid Employee Id	New	Manikanta	2/16/2023	Add Employee Module	Immidiate	Employee Location is not taken	Immediately Taken	Nikhitha	2/18/2023	Nill
10	D_02	TC_002	Always	The user tries to Add the details of Employee.(Invalid inputs) **Expected Result:**Application should take all the details of Employee and displaye the details. **Actual Result:**Application should throw error message.it didn't took employee's location details. Test Data: Name:Vicky experience: jobrole:	Test Cycle_1	1	High	High	Open the Eclipse IDE Run the project as Java Application Enter the Get Employee based on Id Enter Invalid Employee Id	New	Manikanta	2/16/2023	Add Employee Module	Immidiate	Employee Location is not taken	Immediately Taken	Nikhitha	2/18/2023	Nill
16	D_03	TC_006	Always	The user should able to get error message by invalid Id in Delete Employee option **Expected Result:**Application should delete the scholar and throw message "Employee with this ID is not available" **Actual Result:**"Employee details deleted successfully" Test Data:77	Test Cycle_1	1	High	High	Open the Eclipse IDE Run the project as Java Application Enter the Choice Delete Employee Enter Invalid Id	New	Manikanta	2/16/2023	Delete Employee Module	Immidiate	Delete Employee module is not throwing error message	Immediately Taken	Nikhitha	2/18/2023	Nill

Excel View of RTM

1	Employee Management Application				
2	Requirement_ID	Test Scenario_ID	Test Case_ID	Test Result	Defect ID
3	R01	EM_01	TC_01	**Fail**	**D_01**
4	R01	EM_02	TC_02	**Fail**	**D_02**
5	R02	EM_03	TC_03	Pass	
6	R03	EM_04	TC_04	Pass	
7	R03	EM_05	TC_05	Pass	
8	R04	EM_06	TC_06	**Fail**	**D_03**
9	R04	EM_07	TC_07	Pass	
10	R05	EM_08	TC_08	Pass	
11	R05	EM_09	TC_09	Pass	
12	R05	EM_10	TC_10	Pass	
13	R05	EM_11	TC_11	Pass	
14	R06	EM_12	TC_12	Pass	
15	R06	EM_13	TC_13	Pass	

MODULES OF AUTOMATION TESTING

1. Signing in
2. Buy option
3. Add to cart
4. Help and contact

SPECIFICATIONS OF PROJECT

1. Page Object Model (POM) without Page Factory
2. Reading Data From Excel
3. Driven Testing
4. Extent Report Generations
5. TestNG Frame work

Source Code(Automation): Add to cart (main)

addtocart1.java

```java
package main;

import java.io.IOException;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.testng.annotations.Test;
import com.aventstack.extentreports.ExtentReports;
import com.aventstack.extentreports.ExtentTest;
import com.aventstack.extentreports.reporter.ExtentHtmlReporter;

import pageobjects.addtocartPageObjects;

public class addtocart1 {

    //declaring the variables globally
    static ExtentHtmlReporter spark;
    static ExtentReports extent;
    static ExtentTest test;
    private static WebDriver driver;
    static String browser = "Chrome";

@Test
public void addtocart_Test() throws IOException, InterruptedException{

System.setProperty("webdriver.chrome.driver", "C:\\Users\\Seganti.Manikanta\\Downloads\\chromedriver_win32 (1)\\chromedriver.exe");
//object creation for the report
 extent=new ExtentReports();
//setting path for the extent report
 spark=new ExtentHtmlReporter("C:\\HCL training\\program\\core java\\LAB3\\LAB4\\CONSOLE PROJECT\\ETSY\\reportaddtocart1.html");
extent.attachReporter(spark);
 test=extent.createTest("Verify the add to cart functionality of etsy");// here we have to give of extent report

```

Source Code(Automation): Add to cart (main)

```
31  test=extent.createTest("Verify the add to cart functionality of etsy");// here we have to give of extent report
32
33    driver = new ChromeDriver();
34     //opening the Url
35    driver.get("https://www.etsy.com/");
36    System.out.println("Title is:"+driver.getTitle());
37
38    addtocartPageObjects addtocartPageObj = new addtocartPageObjects(driver);
39          //Clicking on jewellery button
40         Thread.sleep(5000);
41         addtocartPageObj.clickjewelleryButton();
42         //Clicking on shop this item button
43         addtocartPageObj.clickshopthisitemButton();
44         //Clicking on add to basket button
45         Thread.sleep(5000);
46         addtocartPageObj.clickaddtobasketButton();
47
48         driver.close();
49
50     //To erase any previous data on the report and create a new report.
51    extent.flush();
52  }
53 public static WebDriver getDriver() {
54      return driver;
55  }
56 public static void setDriver(WebDriver driver) {
57      addtocart1.driver = driver;
58  }
59  }
60
```

Source Code(Automation): Add to cart(Page object)

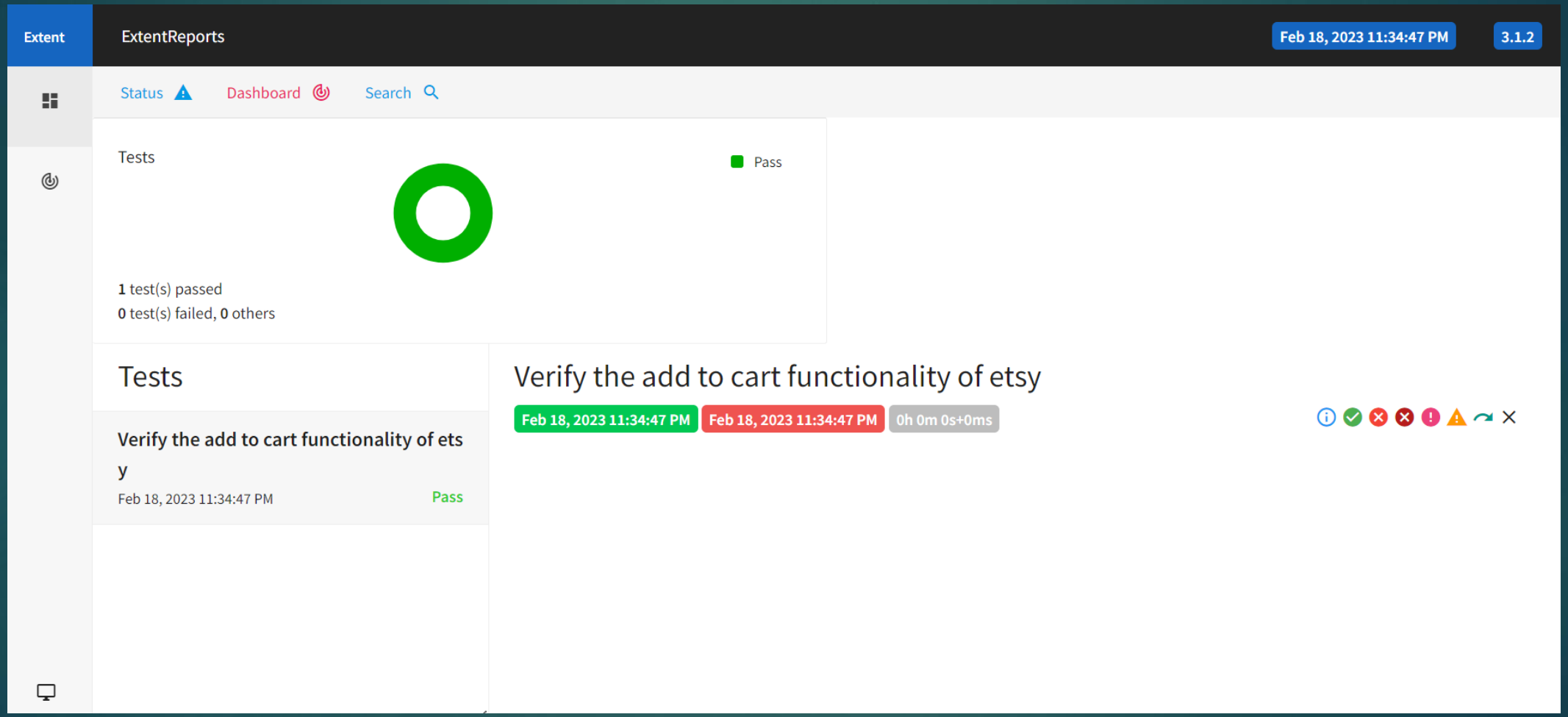
addtocartPageObjects.java

```
1  package pageobjects;
2
3  import org.openqa.selenium.By;
4  import org.openqa.selenium.WebDriver;
5  import org.testng.Assert;
6
7  public class addtocartPageObjects {
8
9      WebDriver driver;
10     By button_jewellery = By.xpath("//*[@id=\"content\"]/div/div[1]/div[2]/div/ul/li[2]/a");
11     By button_shopthisitem = By.xpath("//*[@id=\"content\"]/div/div[1]/div/div[3]/div[4]/div/div/div/div[2]/a");
12     By button_addtobasket = By.cssSelector("#listing-page-cart > div.wt-mb-xs-6.wt-mb-lg-0 > div:nth-child(1) > div.wt-display-flex-xs.wt-flex-dire
13     By button_wedding = By.xpath("//*[@id=\"content\"]/div/div[1]/div[2]/div/ul/li[1]/a");
14     By button_HomeandLiving = By.xpath("//*[@id=\"content\"]/div/div[1]/div[2]/div/ul/li[4]/a/div[1]/div/img");
15     By button_Wallart = By.xpath("//*[@id=\"content\"]/div/div[1]/div[2]/div/ul/li[5]/a");
16
17
18
19     public addtocartPageObjects (WebDriver driver) {
20         this.driver = driver;
21         }
22     public void clickjewelleryButton() {
23         //Assertion
24         Assert.assertTrue(driver.findElement(button_jewellery).isDisplayed(), "\" jewellery\" button is displayed");
25         //Clicking on jewellery button
26     driver.findElement (button_jewellery).click();
27     }
28     public void clickshopthisitemButton() {
29         //Assertion
30         Assert.assertTrue(driver.findElement(button_shopthisitem).isDisplayed(), "\" shop this item\" button is displayed");
31         //Clicking on shop this item button
32     driver.findElement (button_shopthisitem).click();
33     }
34     public void clickaddtobasketButton() {
35         //Assertion
```

Source Code(Automation): Add to cart (Page object)

```
32      driver.findElement (button_shopthisitem).click();
33      }
34      public void clickaddtobasketButton() {
35          //Assertion
36          Assert.assertTrue(driver.findElement(button_addtobasket).isDisplayed(), "\" add to basket\" button is displayed");
37          //Clicking on add to basket button
38      driver.findElement (button_addtobasket).click();
39      }
40      public void clickweddingButton() {
41          //Assertion
42          Assert.assertTrue(driver.findElement(button_wedding).isDisplayed(), "\" wedding\" button is displayed");
43          //Clicking on wedding button
44      driver.findElement (button_wedding).click();
45      }
46      public void clickHomeandLivingButton() {
47          //Assertion
48          Assert.assertTrue(driver.findElement(button_HomeandLiving).isDisplayed(), "\" Home and Living\" button is displayed");
49          //Clicking on wedding button
50      driver.findElement (button_HomeandLiving).click();
51      }
52      public void clickWallartButton() {
53          //Assertion
54          Assert.assertTrue(driver.findElement(button_Wallart).isDisplayed(), "\" Wall art\" button is displayed");
55          //Clicking on Home and Living button
56      driver.findElement (button_Wallart).click();
57      }
58  }
59
```

Extent Report: Add to cart



Emailable-Report: Add to cart

Test	# Passed	# Skipped	# Retried	# Failed	Time (ms)	Included Groups	Excluded Groups
Default suite							
Default test	1	0	0	0	29,426		

Class	Method	Start	Time (ms)
Default suite			
Default test — passed			
main.addtocart1	addtocart_Test	1676744548201	29373

Default test

main.addtocart1#addtocart_Test

Source Code(Automation): Buy module(main)

buy1.java | buy2.java | buy3.java | testng.xml

```
1  package main;
2
3  import java.io.IOException;
4  import org.openqa.selenium.WebDriver;
5  import org.openqa.selenium.chrome.ChromeDriver;
6  import org.testng.annotations.Test;
7  import com.aventstack.extentreports.ExtentReports;
8  import com.aventstack.extentreports.ExtentTest;
9  import com.aventstack.extentreports.reporter.ExtentHtmlReporter;
10
11 import pageobjects.buyPageObjects;
12
13 public class buy1 {
14
15 //  private static final TimeUnit TimeUnitSECONDS = null;
16     //declaring the variables globally
17     static ExtentHtmlReporter spark;
18     static ExtentReports extent;
19     static ExtentTest test;
20     private static WebDriver driver;
21     static String browser = "Chrome";
22
23 @Test
24 public void buy_Test() throws IOException, InterruptedException{
25
26 System.setProperty("webdriver.chrome.driver",  "C:\\Users\\Seganti.Manikanta\\Downloads\\chromedriver_win32 (1)\\chromedriver.exe");
27 //object creation for the report
28  extent=new ExtentReports();
29 //setting path for the extent report
30  spark=new ExtentHtmlReporter("C:\\HCL training\\program\\core java\\LAB3\\LAB4\\CONSOLE PROJECT\\ETSY\\reportbuy1.html");
31 extent.attachReporter(spark);
32  test=extent.createTest("Verify the buy functionality of etsy");// here we have to give of extent report
33
34    driver = new ChromeDriver();
35     //opening the Url
```

Source Code(Automation): Buy module(main)

```
35      //opening the Url
36     driver.get("https://www.etsy.com/");
37     System.out.println("Title is:"+driver.getTitle());
38
39     buyPageObjects buyPageObj = new buyPageObjects(driver);
40           //Clicking on jewellery button
41          Thread.sleep(5000);
42          buyPageObj.clickjewelleryButton();
43          //Clicking on shop this item button
44          buyPageObj.clickshopthisitemButton();
45          //Clicking on add to basket button
46          Thread.sleep(5000);
47          buyPageObj.clickaddtobasketButton();
48          Thread.sleep(5000);
49          //Clicking on check out button
50          buyPageObj.clickcheckoutButton();
51          //Clicking on Buy button
52          buyPageObj.clickbuyButton();
53
54          driver.close();
55
56      //To erase any previous data on the report and create a new report.
57     extent.flush();
58  }
59 public static WebDriver getDriver() {
60      return driver;
61  }
62 public static void setDriver(WebDriver driver) {
63      buy1.driver = driver;
64  }
65  }
66
67
```

Source Code(Automation): Buy module(Page object)

buyPageObjects.java

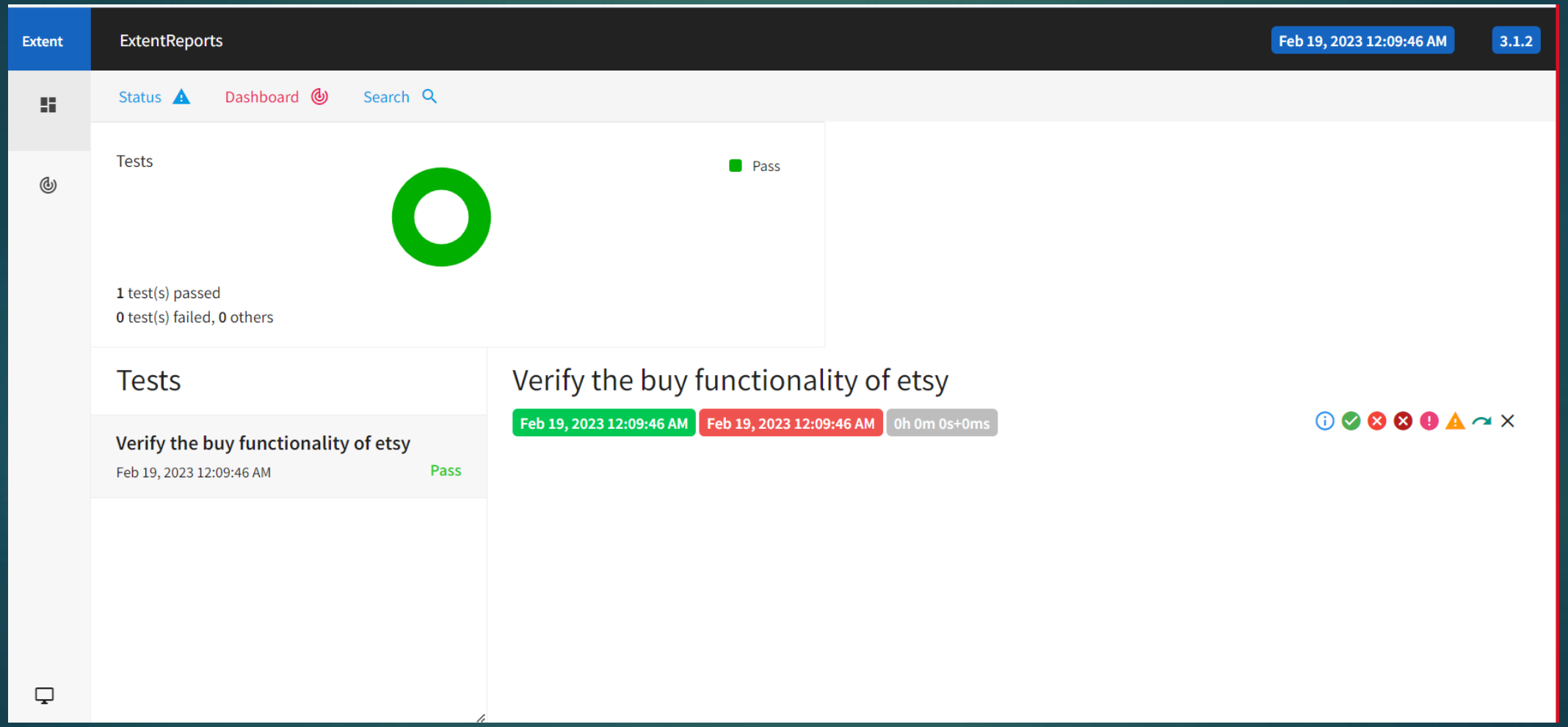
```java
package pageobjects;

import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.testng.Assert;

public class buyPageObjects {

    WebDriver driver;
    By button_jewellery = By.xpath("//*[@id=\"content\"]/div/div[1]/div[2]/div/ul/li[2]/a");
    By button_shopthisitem = By.xpath("//*[@id=\"content\"]/div/div[1]/div/div[3]/div[4]/div/div/div/div[2]/a");
    By button_addtobasket = By.xpath("(//button[@type=\"submit\"])[2]");
    By button_wedding = By.xpath("//*[@id=\"content\"]/div/div[1]/div[2]/div/ul/li[1]/a");
    By button_HomeandLiving = By.xpath("//*[@id=\"content\"]/div/div[1]/div[2]/div/ul/li[4]/a/div[1]/div/img");
    By button_Wallart = By.xpath("//*[@id=\"content\"]/div/div[1]/div[2]/div/ul/li[5]/a");
    By button_checkout = By.cssSelector("#atc-overlay-content > div.wt-display-flex-md.wt-flex-direction-column-md.wt-bb-xs.wt-pb-xs-4 > div > a'
    By button_buy = By.xpath("//*[@id=\"multi-shop-cart-list\"]/div[1]/div/div[1]/div/div/div/form/div[2]/div[1]/button");


    public buyPageObjects (WebDriver driver) {
        this.driver = driver;
        }
    public void clickjewelleryButton() {
        //Assertion
        Assert.assertTrue(driver.findElement(button_jewellery).isDisplayed(), "\" jewellery\" button is displayed");
        //Clicking on jewellery button
    driver.findElement (button_jewellery).click();
    }
    public void clickshopthisitemButton() {
        //Assertion
        Assert.assertTrue(driver.findElement(button_shopthisitem).isDisplayed(), "\" shop this item\" button is displayed");
        //Clicking on shop this item button
    driver.findElement (button_shopthisitem).click();
    }
    public void clickaddtobasketButton() {
```

Source Code(Automation): Buy module(Page object)

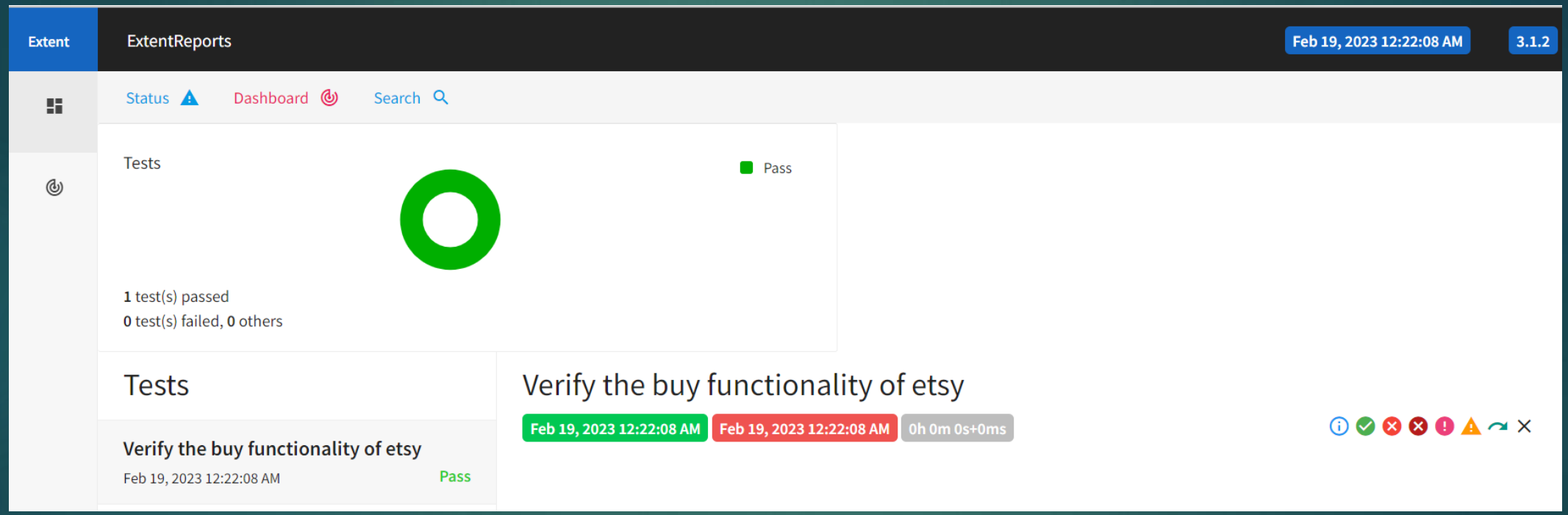
*buyPageObjects.java

```
        //Assertion
        Assert.assertTrue(driver.findElement(button_addtobasket).isDisplayed(), "\" add to basket\" button is displayed");
        //Clicking on add to basket button
    driver.findElement (button_addtobasket).click();
    }
    public void clickweddingButton() {
        //Assertion
        Assert.assertTrue(driver.findElement(button_wedding).isDisplayed(), "\" wedding\" button is displayed");
        //Clicking on wedding button
    driver.findElement (button_wedding).click();
    }
    public void clickHomeandLivingButton() {
        //Assertion
        Assert.assertTrue(driver.findElement(button_HomeandLiving).isDisplayed(), "\" Home and Living\" button is displayed");
        //Clicking on wedding button
    driver.findElement (button_HomeandLiving).click();
    }
    public void clickWallartButton() {
        //Assertion
        Assert.assertTrue(driver.findElement(button_Wallart).isDisplayed(), "\" Wall art\" button is displayed");
        //Clicking on Home and Living button
    driver.findElement (button_Wallart).click();
    }
    public void clickcheckoutButton() {
        //Assertion
        Assert.assertTrue(driver.findElement(button_checkout).isDisplayed(), "\" checkout\" button is displayed");
        //Clicking on Home and Living button
    driver.findElement (button_checkout).click();
    }
    public void clickbuyButton() {
        //Assertion
        Assert.assertTrue(driver.findElement(button_buy).isDisplayed(), "\" checkout\" button is displayed");
        //Clicking on Home and Living button
    driver.findElement (button_buy).click();
    }
```

Extent Report: Buy module



Emailable-Report: Buy module



Source Code(Automation): Help module(main)

help1.java

```java
package main;
import java.io.IOException;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.testng.annotations.Test;
import com.aventstack.extentreports.ExtentReports;
import com.aventstack.extentreports.ExtentTest;
import com.aventstack.extentreports.reporter.ExtentHtmlReporter;
import pageobjects.helpPageObjects;

public class help1 {

    //declaring the variables globally
    static ExtentHtmlReporter spark;
    static ExtentReports extent;
    static ExtentTest test;
    private static WebDriver driver;
    static String browser = "Chrome";

@Test
public void help_Test() throws IOException, InterruptedException{

System.setProperty("webdriver.chrome.driver",  "C:\\Users\\Seganti.Manikanta\\Downloads\\chromedriver_win32 (1)\\chromedriver.exe");
//object creation for the report
 extent=new ExtentReports();
//setting path for the extent report
 spark=new ExtentHtmlReporter("C:\\HCL training\\program\\core java\\LAB3\\LAB4\\CONSOLE PROJECT\\ETSY\\reporthelp1.html");
extent.attachReporter(spark);
 test=extent.createTest("Verify the help and contact functionality of etsy");// here we have to give of extent report

   driver = new ChromeDriver();
    //opening the Url
   driver.get("https://www.etsy.com/");
   System.out.println("Title is:"+driver.getTitle());

```

Source Code(Automation): Help module(main)

```
31      driver = new ChromeDriver();
32       //opening the Url
33      driver.get("https://www.etsy.com/");
34      System.out.println("Title is:"+driver.getTitle());
35
36      helpPageObjects helpPageObj = new helpPageObjects(driver);
37
38           Thread.sleep(5000);
39           //Clicking on help and contact button
40           helpPageObj.clickhelpButton();
41           Thread.sleep(5000);
42           //Clicking on purchasing button
43           helpPageObj.clickpurchasingButton();
44
45           driver.close();
46
47       //To erase any previous data on the report and create a new report.
48      extent.flush();
49  }
50  public static WebDriver getDriver() {
51       return driver;
52  }
53  public static void setDriver(WebDriver driver) {
54       help1.driver = driver;
55  }
56  }
57
```

WebDriver driver

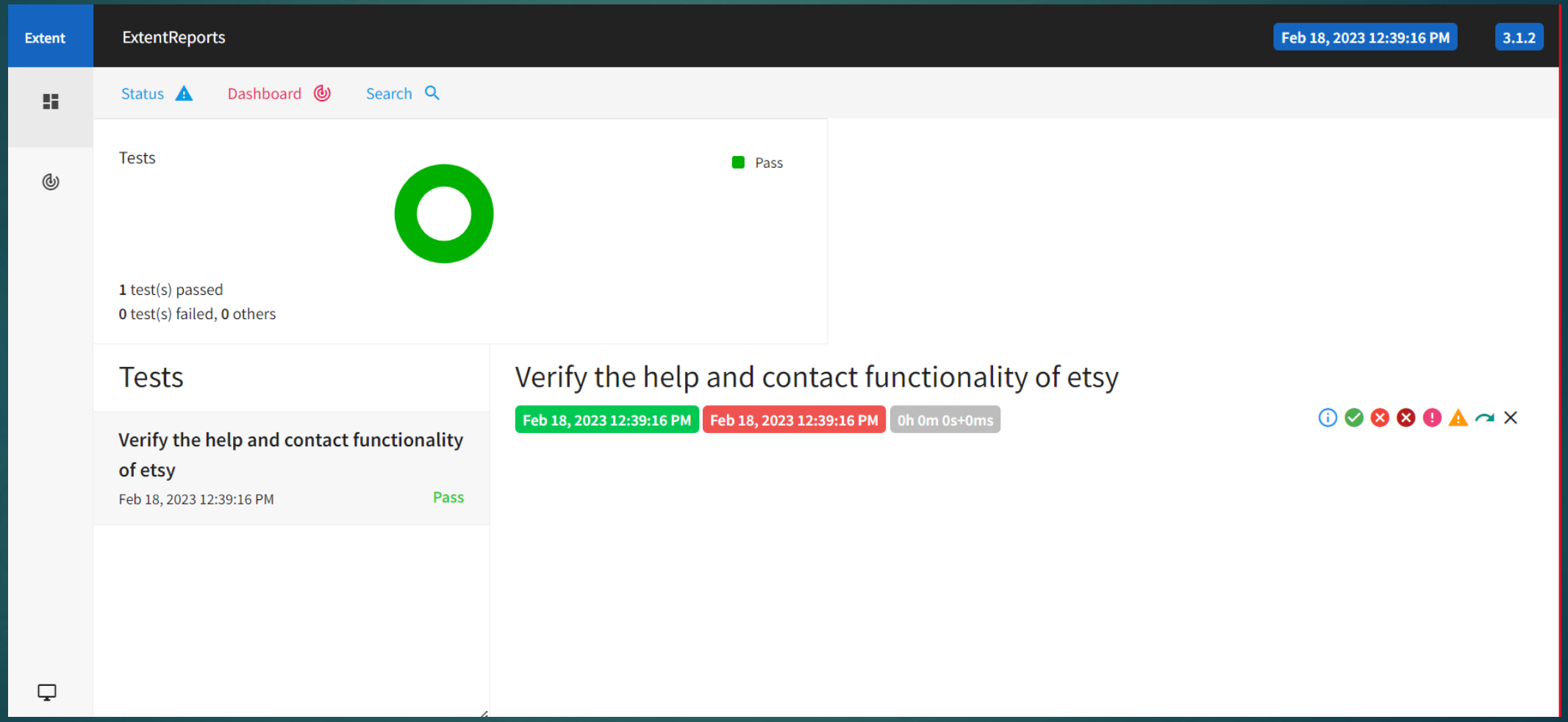
Press 'F2' for focus

Source Code(Automation): Help module(Page object)

helpPageObjects.java

```java
package pageobjects;

import org.openqa.selenium.By;

public class helpPageObjects {

    WebDriver driver;
    By button_help = By.cssSelector("#footer-extra-links-help > ul > li:nth-child(1) > a");
    By button_purchasing = By.cssSelector("body > main > div.wt-body-max-width.wt-pr-xs-4.wt-pl-xs-4 > div.home_page__categories-shopping > div.wt-

    public helpPageObjects (WebDriver driver) {
        this.driver = driver;
        }
    public void clickhelpButton() {
        //Assertion
        Assert.assertTrue(driver.findElement(button_help).isDisplayed(), "\" help and contact\" button is displayed");
        //Clicking on help and contact button
    driver.findElement (button_help).click();
    }
    public void clickpurchasingButton() {
        //Assertion
        Assert.assertTrue(driver.findElement(button_purchasing).isDisplayed(), "\" purchasing\" button is displayed");
        //Clicking on purchasing button
    driver.findElement (button_purchasing).click();
    }
}

```

Extent Report: Help module



Emailable-Report: Help module

Test	# Passed	# Skipped	# Retried	# Failed	Time (ms)	Included Groups	Excluded Groups
Default suite							
Default test	1	0	0	0	47,277		

Class	Method	Start	Time (ms)
Default suite			
Default test — passed			
main.help1	help_Test	1676746958680	47243

Default test

main.help1#help_Test

Source Code(Automation): sign in module(main)

signin1.java

```
1  package main;
2
3  import java.io.File;
4  import java.io.FileInputStream;
5  import java.io.IOException;
6  import java.util.concurrent.TimeUnit;
7
8  import org.apache.poi.xssf.usermodel.XSSFSheet;
9  import org.apache.poi.xssf.usermodel.XSSFWorkbook;
10 import org.openqa.selenium.WebDriver;
11 import org.openqa.selenium.chrome.ChromeDriver;
12 import org.testng.annotations.Test;
13 import com.aventstack.extentreports.ExtentReports;
14 import com.aventstack.extentreports.ExtentTest;
15 import com.aventstack.extentreports.reporter.ExtentHtmlReporter;
16 import pageobjects.SigninPageObjects;
17
18 public class signin1 {
19
20     //declaring the variables globally
21     static ExtentHtmlReporter spark;
22     static ExtentReports extent;
23     static ExtentTest test;
24     private static WebDriver driver;
25 //  static String browser = "Chrome";
26
27 @Test
28 public void sign_inTest() throws Exception{
29
30 System.setProperty("webdriver.chrome.driver",  "C:\\Users\\Seganti.Manikanta\\Downloads\\chromedriver_win32 (1)\\chromedriver.exe");
31 //object creation for the report
32  extent=new ExtentReports();
33 //setting path for the extent report
34  spark=new ExtentHtmlReporter("C:\\HCL training\\program\\core java\\LAB3\\LAB4\\CONSOLE PROJECT\\ETSY\\reportsignin1.html");
35 extent.attachReporter(spark);
```

Source Code(Automation): Sign in module(main)

*signin1.java

```
35 extent.attachReporter(spark);
36  test=extent.createTest("Verify the Sign in functionality of etsy");
37  File datafile = new File( "C:\\HCL training\\program\\core java\\LAB3\\LAB4\\CONSOLE PROJECT\\ETSY\\Etsy_excel\\Excel_data.xlsx");
38  FileInputStream fis=new FileInputStream(datafile);
39  XSSFWorkbook wb = new XSSFWorkbook(fis);
40 XSSFSheet sh = wb.getSheet("Sheet2");
41 //path of data from the excel the sheet
42 int rowcount =sh.getLastRowNum();
43    String[][] data1=new String[4][3];
44    System.out.println(rowcount);
45    for(int i=1;i<=rowcount-1;i++)
46    {
47   data1[i][0]=sh.getRow(i).getCell(0).getStringCellValue();
48    System.out.println(data1[i][0]);
49    data1[i][1]=sh.getRow(i).getCell(1).getStringCellValue();
50    System.out.println(data1[i][1]);
51   data1[i][2]=sh.getRow(i).getCell(2).getStringCellValue();
52    System.out.println(data1[i][2]);
53    driver = new ChromeDriver();
54     //opening the Url
55    driver.get("https://www.etsy.com/");
56    System.out.println("Title is:"+driver.getTitle());
57 SigninPageObjects signInPageObj = new SigninPageObjects(driver);
58 driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);
59         //Clicking on SignIn button
60         Thread.sleep(5000);
61         signInPageObj.clickSignInButton();
62         //Clicking on SignIn button
63         Thread.sleep(5000);
64         signInPageObj.clickregister();
65         //Getting data from excel
66         Thread.sleep(5000);
67         signInPageObj.setTextEmailaddress(data1[i][0]);
68         //Getting data from excel
69         Thread.sleep(5000);
```

Source Code(Automation): Sign in module(main)

```
68          //Getting data from excel
69          Thread.sleep(5000);
70          signInPageObj.setTextInfirstname(data1[i][1]);
71          //Getting data from excel
72          Thread.sleep(5000);
73          signInPageObj.setTextInPassword(data1[i][2]);
74          //Clicking on SignIn button
75          Thread.sleep(6000);
76          signInPageObj.clickregister1();
77
78          driver.close();
79  }
80      //To erase any previous data on the report and create a new report.
81     extent.flush();
82  }
83  public static WebDriver getDriver() {
84      return driver;
85  }
86  public static void setDriver(WebDriver driver) {
87      signin1.driver = driver;
88  }
89  }
90
```

Source Code(Automation): Sign in module(Page Objects)

SigninPageObjects.java

```
1  package pageobjects;
2
3  import org.openqa.selenium.By;
4  import org.openqa.selenium.WebDriver;
5  import org.testng.Assert;
6
7  public class SigninPageObjects {
8
9  WebDriver driver;
10 //Storing the web elements by methods
11 By button_signin = By.cssSelector("#gnav-header-inner > div.wt-flex-shrink-xs-0 > nav > ul > li:nth-child(1) > button");
12 By TextBox_Emailaddress = By.cssSelector("input#join_neu_email_field");
13 By TextBox_Password = By.cssSelector("input#join_neu_password_field");
14 By button_signin1 = By.cssSelector("#join-neu-form > div.wt-grid.wt-grid--block > div > div:nth-child(10) > div > button");
15 By button_register = By.cssSelector("#join-neu-form > div.wt-grid.wt-grid--block > div > div:nth-child(1) > div > button");
16 By TextBox_firstname = By.cssSelector("input#join_neu_first_name_field");
17 By button_register1 = By.cssSelector("#join-neu-form > div.wt-grid.wt-grid--block > div > div:nth-child(9) > div > button");
18 By button_cwg = By.xpath("//*[@id=\"join-neu-form\"]/div[3]/div[1]/div/button");
19 By button_uac = By.cssSelector("#yDmH0d");
20 By Textbox_email = By.cssSelector("#identifierId");
21 By button_next = By.cssSelector("#identifierNext > div > button");
22
23
24 public SigninPageObjects (WebDriver driver) {
25 this.driver = driver;
26 }
27 public void clickSignInButton() {
28     //Assertion
29     Assert.assertTrue(driver.findElement(button_signin).isDisplayed(), "\" signin\" button is displayed");
30     //Clicking on signin button
31 driver.findElement (button_signin).click();
32 }
33 public void setTextEmailaddress(String Text) {
34     //Assertion
35     Assert.assertTrue(driver.findElement(TextBox_Emailaddress).isDisplayed(), "\" Emailaddress\" Text is displayed");
```

Source Code(Automation): Sign in module(Page Objects)

SigninPageObjects.java

```
    //Assertion
    Assert.assertTrue(driver.findElement(TextBox_Emailaddress).isDisplayed(), "\" Emailaddress\" Text is displayed");
    //Entering on Emailaddress button
driver.findElement (TextBox_Emailaddress).sendKeys(Text);
}
public void setTextInPassword(String Text) {
    //Assertion
    Assert.assertTrue(driver.findElement(TextBox_Password).isDisplayed(), "\" Password TextBox\" Text is displayed");
    //Entering on password button
driver.findElement (TextBox_Password).sendKeys(Text);
}
public void clickSignInButton1() {
    //Assertion
    Assert.assertTrue(driver.findElement(button_signin1).isDisplayed(), "\" signin1\" button is displayed");
    //Clicking on sigin1 button
driver.findElement (button_signin1).click();
}
public void clickregister() {
    //Assertion
    Assert.assertTrue(driver.findElement(button_register).isDisplayed(), "\" register\" button is displayed");
    //Clicking on register button
driver.findElement (button_register).click();
}
public void setTextInfirstname(String Text) {
    //Assertion
    Assert.assertTrue(driver.findElement(TextBox_firstname).isDisplayed(), "\" firstname TextBox\" Text is displayed");
    //Entering on firstname button
driver.findElement (TextBox_firstname).sendKeys(Text);
}
public void clickregister1() {
    //Assertion
    Assert.assertTrue(driver.findElement(button_register1).isDisplayed(), "\" register1\" button is displayed");
    //Clicking on register1 button
driver.findElement (button_register1).click();
}
```

Source Code(Automation): Sign in module(Page Objects)

SigninPageObjects.java

```
62 }
63 public void clickregister1() {
64     //Assertion
65     Assert.assertTrue(driver.findElement(button_register1).isDisplayed(), "\" register1\" button is displayed");
66     //Clicking on register1 button
67 driver.findElement (button_register1).click();
68 }
69 public void clickcwg() {
70     //Assertion
71     Assert.assertTrue(driver.findElement(button_cwg).isDisplayed(), "\" continue with google\" button is displayed");
72     //Clicking on continue with google button
73 driver.findElement (button_cwg).click();
74 }
75 public void clickuac() {
76     //Assertion
77     Assert.assertTrue(driver.findElement(button_uac).isDisplayed(), "\" use another account\" button is displayed");
78     //Clicking on email button
79 driver.findElement (button_uac).click();
80 }
81 public void setTextInemail(String Text) {
82     //Assertion
83     Assert.assertTrue(driver.findElement(Textbox_email).isDisplayed(), "\" use another account\" button is displayed");
84     //Entering on email button
85 driver.findElement (Textbox_email).sendKeys(Text);
86 }
87 public void clicknext() {
88     //Assertion
89     Assert.assertTrue(driver.findElement(button_next).isDisplayed(), "\" next\" button is displayed");
90     //Clicking on next button
91 driver.findElement (button_next).click();
92 }
93
94 }
```

Extent Report: Sign in module

Emailable-Report: Sign in module

Test	# Passed	# Skipped	# Retried	# Failed	Time (ms)	Included Groups	Excluded Groups
Default suite							
Default test	1	0	0	0	145,423		

Class	Method	Start	Time (ms)
Default suite			
Default test — passed			
main.signin1	sign_inTest	1676747502960	145383

Default test

main.signin1#sign_inTest

Source-code: testing.xml

testng.xml

```xml
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE suite SYSTEM "https://testng.org/testng-1.0.dtd">
<suite name="Suite">
  <test thread-count="5" name="Test">
    <classes>
      <class name="main.signin3"/>
      <class name="main.signin1"/>
      <class name="main.signin2"/>
      <class name="main.buy4"/>
      <class name="main.buy3"/>
      <class name="main.addtocart2"/>
      <class name="main.addtocart1"/>
      <class name="main.buy2"/>
      <class name="main.addtocart4"/>
      <class name="main.addtocart3"/>
      <class name="main.help1"/>
      <class name="main.buy1"/>
    </classes>
  </test> <!-- Test -->
</suite> <!-- Suite -->
```

Emailable-Report: testing.xml

Test	# Passed	# Skipped	# Retried	# Failed	Time (ms)	Included Groups	Excluded Groups
Suite							
Test	12	0	0	0	690,278		

Class	Method	Start	Time (ms)
Suite			
Test — passed			
main.addtocart1	addtocart_Test	1676752154675	38930
main.addtocart2	addtocart_Test	1676752117346	37326
main.addtocart3	addtocart_Test	1676752281247	40614
main.addtocart4	addtocart_Test	1676752245241	36003
main.buy1	buy_Test	1676752358028	45080
main.buy2	buy_Test	1676752193607	51631
main.buy3	buy_Test	1676752073119	44224
main.buy4	buy_Test	1676752038419	34697
main.help1	help_Test	1676752321864	36160
main.signin1	sign_inTest	1676751800848	148353
main.signin2	sign_inTest	1676751949204	89212
main.signin3	sign_inTest	1676751712855	87986

GitHub link:

Thank you