

Automatic evaluation[+]

LoanManagement/src/com/dao/CustomerDAO.java

```
1 package com.dao;
2
3 import java.util.*;
4
5 import com.exception.LoanException;
6 import com.model.Customer;
7
8 public class CustomerDAO {
9
10     List<Customer> customerList = new ArrayList<>();
11
12     public void addCustomer(Customer customerObj){
13         customerList.add(customerObj);
14     }
15
16     public List<Customer> viewAllCustomer(){
17         if(customerList.isEmpty()){
18             return null;
19         }
20         return customerList;
21     }
22
23     public boolean deleteCustomer(int customerId){
24         //boolean flag=false;
25         for(Customer c : customerList){
26             if(c.getCustomerId()==customerId){
27                 customerList.remove(c);
28                 return true;
29             }
30         }
31         return false;
32     }
33
34     public boolean validateCustomerId(int customerId){
35
36         try {
37             if(customerId>=1000 && customerId<=9999)
38             {
39                 return true;
40             }
41             else{
42                 throw new LoanException("Customer Id is invalid");
43             }
44         } catch (LoanException e) {
45             // TODO Auto-generated catch block
46             e.printStackTrace();
47         }
48         return false;
49     }
50
51 }
52
```

LoanManagement/src/com/dao/LoanDAO.java

```
1 package com.dao;
2
```



```

3
4 import java.util.ArrayList;
5 import java.util.List;
6
7 import com.exception.LoanException;
8 //import com.model.Customer;
9 import com.model.Loan;
10
11 //import java.io.IOException;
12
13 public class LoanDAO {
14
15     List<Loan> loanList = new ArrayList<>();
16
17     public void issueLoan(Loan loanObj){
18         loanList.add(loanObj);
19     }
20
21     public List<Loan> viewLoanByType(int loanType) throws LoanException {
22
23         if(loanList.isEmpty()){
24             return null;
25         }
26         else
27         {
28             List<Loan> temp= new ArrayList<>();
29             for(Loan loan : loanList){
30                 if(loan.getLoanType().equals(loanType)){
31                     temp.add(loan);
32                 }
33             }
34
35             //try{
36                 if(temp==null)
37                 {
38                     // to check if temp is empty
39                     throw new LoanException("No loans available with type
40 "+loanType);
41                 }
42             //}
43             //catch(LoanException e){
44
45             //}
46             /*inally{
47                 return temp;
48             }*/
49             else{
50                 return temp;
51             }
52         }
53     }
54 }
55
LoanManagement/src/com/dao/PaymentDAO.java
1 package com.dao;
2

```

```

3 import java.util.ArrayList;
4 import java.util.List;
5
6 import com.model.Payment;
7 //import com.dao.LoanDAO;
8 //import com.dao.CustomerDAO;
9
10 public class PaymentDAO {
11
12     private List<Payment> paymentList = new ArrayList<>();
13
14     //private CustomerDAO custDAO;
15
16
17     public void setPaymentList(List<Payment> paymentList ) {
18         this.paymentList= paymentList;
19     }
20
21     public List<Payment> getPaymentList() {
22         return paymentList;
23     }
24
25     public void makePayment(Payment obj){
26         paymentList.add(obj);
27     }
28
29     /* public void updatePayment(int paymentId,double amount){
30
31     } */
32
33 }
34
LoanManagement/src/com/exception/LoanException.java
1 package com.exception;
2
3 public class LoanException extends Exception {
4     public LoanException(String msg){
5         super(msg);
6     }
7 }
8
LoanManagement/src/com/model/Customer.java
1 package com.model;
2
3 public class Customer {
4
5     int customerId;
6     String customerName;
7     String address;
8     String panNumber;
9     String emailId;
10
11     public Customer(int customerId, String customerName, String address, String
panNumber) {
12         //String pan="";
13         this.customerId = customerId;
14         this.customerName = customerName;

```

```

15         this.address = address;
16         this.panNumber = panNumber;
17     }
18
19     public Customer() {
20         //default
21     }
22
23     public int getCustomerId() {
24         return customerId;
25     }
26
27     public void setCustomerId(int customerId) {
28         this.customerId = customerId;
29     }
30
31     public String getCustomerName() {
32         return customerName;
33     }
34
35     public void setCustomerName(String customerName) {
36         this.customerName = customerName;
37     }
38
39     public String getAddress() {
40         return address;
41     }
42
43     public void setAddress(String address) {
44         this.address = address;
45     }
46
47     public String getPanNumber() {
48         return panNumber;
49     }
50
51     public void setPanNumber(String panNumber) {
52         this.panNumber = panNumber;
53     }
54
55     public String getEmailId() {
56         return emailId;
57     }
58
59     public void setEmailId(String emailId) {
60         this.emailId = emailId;
61     }
62
63 }
64

```

LoanManagement/src/com/model/Loan.java

```

1 package com.model;
2
3 public class Loan {
4     int loanNumber;
5     String loanType="Vehicle";
6     Customer customer;

```



```

7  double loanAmount;
8  double balanceLoanAmount;
9  static final double MAXLOANAMOUNT=1500000;
10
11  public Loan(){
12      //default
13  }
14
15  public Loan(int loanNumber, String loanType, Customer customer, double
loanAmount) {
16      this.loanNumber = loanNumber;
17      this.loanType = loanType;
18      this.customer = customer;
19      this.loanAmount = loanAmount;
20      balanceLoanAmount=loanAmount;
21  }
22
23  public int getLoanNumber() {
24      return loanNumber;
25  }
26
27  public void setLoanNumber(int loanNumber) {
28      this.loanNumber = loanNumber;
29  }
30
31  public String getLoanType() {
32      return loanType;
33  }
34
35  public void setLoanType(String loanType) {
36      this.loanType = loanType;
37  }
38
39  public Customer getCustomer() {
40      return customer;
41  }
42
43  public void setCustomer(Customer customer) {
44      this.customer = customer;
45  }
46
47  public double getLoanAmount() {
48      return loanAmount;
49  }
50
51  public void setLoanAmount(double loanAmount) {
52      this.loanAmount = loanAmount;
53  }
54
55  public double getBalanceLoanAmount() {
56      return balanceLoanAmount;
57  }
58
59  public void setBalanceLoanAmount(double balanceLoanAmount) {
60      this.balanceLoanAmount = balanceLoanAmount;
61  }
62

```

```

63
64 public boolean checkBalanceAmount(double amountPaid){
65     //float gst=3;
66     return (balanceLoanAmount >= amountPaid);
67     //double amount = balanceLoanAmount+balanceLoanAmount*gst/100;
68     //if(balanceLoanAmount >= amountPaid)
69     //    return true;
70     //else
71     //    return false;
72 }
73
74
75
76
77 }
78
LoanManagement/src/com/model/Payment.java
1 package com.model;
2
3 import java.time.LocalDate;
4
5 public class Payment {
6
7     int paymentId;
8     LocalDate dateOfPayment;
9     Loan loanObj;
10    double amount;
11    String paymentMode;
12    static final float GSTPERCENTAGE=2;
13
14    public Payment(int paymentId, LocalDate dateOfPayment, Loan loanObj, double
amount, String paymentMode) {
15
16        this.paymentId = paymentId;
17        this.dateOfPayment = dateOfPayment;
18        this.loanObj = loanObj;
19        this.amount = amount;
20        this.paymentMode = paymentMode;
21    }
22
23    public int getPaymentId() {
24        return paymentId;
25    }
26
27    public void setPaymentId(int paymentId) {
28        this.paymentId = paymentId;
29    }
30
31    public LocalDate getDateOfPayment() {
32        return dateOfPayment;
33    }
34
35    public void setDateOfPayment(LocalDate dateOfPayment) {
36        this.dateOfPayment = dateOfPayment;
37    }
38
39    public Loan getLoanObj() {

```

```
40     return loanObj;
41 }
42
43 public void setLoanObj(Loan loanObj) {
44     this.loanObj = loanObj;
45 }
46
47 public double getAmount() {
48     return amount;
49 }
50
51 public void setAmount(double amount) {
52     this.amount = amount;
53 }
54
55 public String getPaymentMode() {
56     return paymentMode;
57 }
58
59 public void setPaymentMode(String paymentMode) {
60     this.paymentMode = paymentMode;
61 }
62
63 public static float getGstpercentage() {
64     return GSTPERCENTAGE;
65 }
66
67
68
69 }
70
```

Grade

Reviewed on Tuesday, 4 May 2021, 2:29 AM by Automatic grade

Grade 100 / 100

Assessment report

[+]Grading and Feedback



Automatic evaluation[+]

FlightMgmt/src/com/dao/BookFlightDAO.java

```
1 package com.dao;
2
3 import java.time.LocalDate;
4 import java.util.ArrayList;
5 import java.util.List;
6
7 import com.model.Flight;
8 import com.exception.InvalidFlightException;
9 import com.model.BookFlight;
10 import com.model.Customer;
11
12 public class BookFlightDAO {
13
14     List<BookFlight> bookingList = new ArrayList<>();
15
16     public boolean bookAFlight(final Customer customer,final Flight flight,final
LocalDate dateOfbooking, LocalDate flightDate,int noOfPassengers){
17
18         BookFlight bookObj = new
BookFlight(customer,flight,dateOfbooking,flightDate,noOfPassengers);
19
20         //boolean flag=true;
21         //boolean result = bookingList.add(bookObj);
22         //return result;
23         return bookingList.add(bookObj);
24     }
25
26     public List<BookFlight>viewBookingByFlight(Flight flightObj) throws
InvalidFlightException {
27         final List<BookFlight> temp = new ArrayList<>();
28
29         for(final BookFlight booking : bookingList){
30             if(booking.getFlight().getFlightId()==flightObj.getFlightId()) {
31                 temp.add(booking);
32             }
33         }
34         if(temp.isEmpty()){
35             throw new InvalidFlightException("No booking for Flight
"+flightObj.getFlightId());
36         }
37         return temp;
38     }
39 }
40 }
41
42
43
```

FlightMgmt/src/com/dao/CustomerDAO.java

```
1 package com.dao;
2
3 import java.util.*;
4
5 import com.exception.InvalidCustomerException;
6 import com.model.Customer;
7
```




```

8 public class CustomerDAO {
9
10     List<Customer>customerList = new ArrayList<>();
11
12     public void addCustomer(Customer customer){
13         customerList.add(customer);
14     }
15
16     public Customer viewCustomerByUserName(String userName){
17         try{
18             if(customerList.isEmpty()){
19                 throw new InvalidCustomerException("User Name is invalid");
20             }
21             else{
22                 for(Customer c : customerList) {
23                     if(c.getUserName().equals(userName)){
24                         return c;
25                     }
26                 }
27             }
28         }
29         catch(Exception e){
30             System.out.println(e.getMessage());
31         }
32         return null;
33     }
34
35     public boolean validateCustomer(String userName,String password){
36         //boolean flag=false;
37
38         for(Customer c : customerList) {
39             if(c.getUserName().equals(userName)                &&
c.getPassword().equals(password)) {
40                 return true;
41             }
42         }
43
44         return false;
45     }
46
47 }
48

```

FlightMgmt/src/com/dao/FlightDAO.java

```

1 package com.dao;
2
3 import java.util.*;
4
5 import com.exception.*;
6 import com.model.*;
7
8 public class FlightDAO {
9
10     List<Flight>flightList = new ArrayList<>();
11
12     public void addCourse(Flight flightObj){
13         flightList.add(flightObj);
14     }

```



```

15
16 public List<Flight>viewFlightBySourceDestination(String source,String destination)
throws InvalidFlightException {
17     List<Flight>temp=new ArrayList<>();
18     for(Flight f : flightList) {
19         if(f.getSource().equals(source) && f.getDestination().equals(destination)) {
20             temp.add(f);
21         }
22     }
23     if(temp.isEmpty()){
24         throw new InvalidFlightException("No Flight with source "+source+" and
destination "+destination);
25     }
26     else{
27         return temp;
28     }
29 }
30
31 }
32

```

FlightMgmt/src/com/exception/InvalidCustomerException.java

```

1 package com.exception;
2
3 public class InvalidCustomerException extends Exception {
4
5     public InvalidCustomerException(String msg){
6         super(msg);
7     }
8
9 }
10

```

FlightMgmt/src/com/exception/InvalidFlightException.java

```

1 package com.exception;
2
3 public class InvalidFlightException extends Exception {
4
5     public InvalidFlightException(String msg){
6         super(msg);
7     }
8
9 }
10

```

FlightMgmt/src/com/model/BookFlight.java

```

1 package com.model;
2
3 import java.time.LocalDate;
4 //import java.util.Date;
5
6 public class BookFlight {
7
8     private Customer customer;
9     private Flight flight;
10    private LocalDate dateOfbooking;
11    private LocalDate flightDate;
12    private int noOfPassengers;
13    private double totalFare;
14

```



```

15 //public BookFlight(){
16 //
17 //}
18
19 public BookFlight(Customer customer, Flight flight, LocalDate dateOfbooking,
LocalDate flightDate,
20         int noOfPassengers) {
21     super();
22     this.customer = customer;
23     this.flight = flight;
24     this.dateOfbooking = dateOfbooking;
25     this.flightDate = flightDate;
26     this.noOfPassengers = noOfPassengers;
27     this.totalFare = totalFare;
28 }
29
30 public Customer getCustomer() {
31     return customer;
32 }
33
34 public void setCustomer(Customer customer) {
35     this.customer = customer;
36 }
37
38 public Flight getFlight() {
39     return flight;
40 }
41
42 public void setFlight(Flight flight) {
43     this.flight = flight;
44 }
45
46 public LocalDate getDateOfbooking() {
47     return dateOfbooking;
48 }
49
50 public void setDateOfbooking(LocalDate dateOfbooking) {
51     this.dateOfbooking = dateOfbooking;
52 }
53
54 public LocalDate getFlightDate() {
55     return flightDate;
56 }
57
58 public void setFlightDate(LocalDate flightDate) {
59     this.flightDate = flightDate;
60 }
61
62 public int getNoOfPassengers() {
63     return noOfPassengers;
64 }
65
66 public void setNoOfPassengers(int noOfPassengers) {
67     this.noOfPassengers = noOfPassengers;
68 }
69
70 public double getTotalFare() {

```

```

71     return totalFare;
72 }
73
74 public void setTotalFare(double totalFare) {
75     this.totalFare = totalFare;
76 }
77
78 public float calculateTotalFare(){
79     return (float)noOfPassengers*flight.getFlightFare();
80     //return totalFare;
81 }
82
83 public boolean validateNoOfPassengers(){
84     boolean flag=false;
85     if(noOfPassengers >0 && noOfPassengers <= 30) {
86         flag=true;
87     }
88     return flag;
89 }
90 }
91
FlightMgmt/src/com/model/Customer.java
1 package com.model;
2
3 public class Customer {
4
5     private String customerId;
6     private String customerName;
7     private String emailId;
8     private String userName;
9     private String password="FH782";
10
11     public Customer(String customerName, String emailId, String userName) {
12         this.customerId = customerId;
13         this.customerName = customerName;
14         this.emailId = emailId;
15         this.userName = userName;
16     }
17
18     public String getCustomerId() {
19         return customerId;
20     }
21
22     public void setCustomerId(String customerId) {
23         this.customerId = customerId;
24     }
25
26     public String getCustomerName() {
27         return customerName;
28     }
29
30     public void setCustomerName(String customerName) {
31         this.customerName = customerName;
32     }
33
34     public String getEmailId() {
35         return emailId;

```

```

36 }
37
38 public void setEmailId(String emailId) {
39     this.emailId = emailId;
40 }
41
42 public String getUser_name() {
43     return user_name;
44 }
45
46 public void setUser_name(String user_name) {
47     this.user_name = user_name;
48 }
49
50 public String getPassword() {
51     return password;
52 }
53
54 public void setPassword(String password) {
55     this.password = password;
56 }
57
58
59
60 }
61
FlightMgmt/src/com/model/Flight.java
1 package com.model;
2
3 public class Flight {
4
5     public int flightId;
6     private String flightName;
7     private String source;
8     private String destination;
9     private float flightFare;
10    private int noOfSeats = 40;
11    static final String COMPANYNAME = "Aviva Airlines";
12
13    public Flight(int flightId, String flightName, String source, String destination, float
fare) {
14
15        this.flightId = flightId;
16        this.flightName = flightName;
17        this.source = source;
18        this.destination = destination;
19        this.flightFare = fare;
20
21    }
22
23    public int getFlightId() {
24        return flightId;
25    }
26
27    public void setFlightId(int flightId) {
28        this.flightId = flightId;
29    }

```

```

30
31 public String getFlightName() {
32     return flightName;
33 }
34
35 public void setFlightName(String flightName) {
36     this.flightName = flightName;
37 }
38
39 public String getSource() {
40     return source;
41 }
42
43 public void setSource(String source) {
44     this.source = source;
45 }
46
47 public String getDestination() {
48     return destination;
49 }
50
51 public void setDestination(String destination) {
52     this.destination = destination;
53 }
54
55 public float getFlightFare() {
56     return flightFare;
57 }
58
59 public void setFare(float fare) {
60     this.flightFare = fare;
61 }
62
63 public int getNoOfSeats() {
64     return noOfSeats;
65 }
66
67 public void setNoOfSeats(int noOfSeats) {
68     this.noOfSeats = noOfSeats;
69 }
70
71 public static String getCompanyname() {
72     return COMPANYNAME;
73 }
74 }
75

```

Grade

Reviewed on Wednesday, 12 May 2021, 10:43 PM by Automatic grade

Grade 100 / 100

Assessment report

[+]Grading and Feedback



Program.cs.....

```
using System;
using StoresScheduleSystemBLL;

namespace StoresScheduleSystemConsoleUi
{
    class Program
    {
        static void Main(string[] args)
        {
            string Continue = "y";
            StoreScheduler StoreScheduler = new StoreScheduler();
            if (Continue.ToLower() == "n")
            {
                System.Environment.Exit(1);
            }
            else
            {
                while (Continue.ToLower() == "y")
                {
                    Console.WriteLine("-----");
                    Console.WriteLine("===== GHMC Stores Scheduler =====");
                    Console.WriteLine("-----");
                    try
                    {
                        FindAndUpdateStore(StoreScheduler);
                    }
                    catch (Exception ex)
                    {
                        Console.WriteLine("Error : " + ex.Message);
                    }
                    finally
                    {
                        Console.WriteLine();
                        Console.Write("Do you want to continue for another store(y/n) :");
                        Continue = Console.ReadLine();
                    }
                }
            }
        }
    }
}
```

```

private static void FindAndUpdateStore(StoreScheduler StoreScheduler)
{

    StoreScheduler storeScheduler = new StoreScheduler();
    Console.WriteLine("Enter store id");
    int id = Convert.ToInt32(Console.ReadLine());

    Store store = storeScheduler.GetStoreById(id);
    if (store.StoreId == id)
    {
        Console.WriteLine("Found a store with following details");
        Console.WriteLine("StoreName : " + store.StoreName);
        Console.WriteLine("OwnerName : " + store.OwnerName);
        Console.WriteLine("MobileNo : " + store.MobileNo);
        Console.WriteLine("StoreAddress : " + store.StoreAddress);
        Console.WriteLine("SellsEssentials : " + store.SellsEssentials);
        Console.WriteLine("OpeningTime : " + store.OpeningTime);
        Console.WriteLine("ClosingTime : " + store.ClosingTime);
        Console.WriteLine("Enter the Timeslot assigned to store (TimeSlotA/TimeSlotB)");
        try
        {
            string timeslot = Console.ReadLine();

            storeScheduler.AssignTimings(store, timeslot);
            storeScheduler.UpdateStoreTimings(store, timeslot);

            Console.WriteLine("Time slot updated for the store {0}", store.StoreName);

        }
        catch (Exception e)
        {
            Console.WriteLine(e.Message);
        }
    }
    else
    {
        Console.WriteLine("No store found for store id : {0}", id);
    }
}
}

```



```
}
```

Store.cs.....

.....

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace StoresScheduleSystemBLL
{
    public class Store
    {
        public int StoreId { get; set; }
        public string StoreName { get; set; }
        public string OwnerName { get; set; }
        public string MobileNo { get; set; }
        public string StoreAddress { get; set; }
        public bool SellsEssentials { get; set; }
        public string OpeningTime { get; set; }
        public string ClosingTime { get; set; }
    }
}
```

StoreScheduler.cs.....

.....

```
using System;
using System.Data;
using StoresScheduleSystemDAL;

namespace StoresScheduleSystemBLL
{
    public class StoreScheduler
    {
        StoresScheduleDAO StoresScheduleDAO = new StoresScheduleDAO();
        public StoreScheduler()
        {
            // Instantiate fields here
        }
        public Store AssignTimings(Store store, string timeSlot)
        {
            TimeSlotA timeSlotA = new TimeSlotA();
        }
    }
}
```

```

TimeSlotB timeSlotB = new TimeSlotB();
if (timeSlot == "timeSlotA" || timeSlot=="TimeSlotA")
{
    timeSlotA.SetIsEssentialItemsStore(Convert.ToBoolean(store.SellsEssentials));
    store.OpeningTime = timeSlotA.GetOpeningTime();
    store.ClosingTime = timeSlotA.GetClosingTime();
}
else if(timeSlot == "timeSlotB" || timeSlot == "TimeSlotB")
{
    timeSlotB.SetIsEssentialItemsStore(Convert.ToBoolean(store.SellsEssentials));
    store.OpeningTime = timeSlotB.GetOpeningTime();
    store.ClosingTime = timeSlotB.GetClosingTime();
}
else
{
    throw new ArgumentException("Error : Invalid time slot");
}

return store;
}

public Store GetStoreById(int storeId)
{
    Store Store = new Store();
    DataTable dt = StoresScheduleDAO.FindStore(storeId);
    if(dt!=null && dt.Rows.Count>0)
    {
        Store.MobileNo = dt.Rows[0]["MobileNo"].ToString();
        Store.StoreName = dt.Rows[0]["StoreName"].ToString();
        Store.OwnerName = dt.Rows[0]["OwnerName"].ToString();
        Store.StoreId = Convert.ToInt32(dt.Rows[0]["StoreId"]);
        Store.StoreAddress = dt.Rows[0]["StoreAddress"].ToString();
        Store.SellsEssentials = Convert.ToBoolean(dt.Rows[0]["SellsEssentials"]);
        Store.OpeningTime = dt.Rows[0]["OpeningTime"].ToString();
        Store.ClosingTime = dt.Rows[0]["ClosingTime"].ToString();

    }
    else
    {
        return null;
    }

return Store;
}

```

```

    }

    public bool UpdateStoreTimings(Store store, string timeSlot)
    {
        bool IsUpdated = false;
        // Donot change method signature
        // Implement code here

        TimeSlotA timeSlotA = new TimeSlotA();
        TimeSlotB timeSlotB = new TimeSlotB();
        if (timeSlot == "timeSlotA" || timeSlot=="TimeSlotA")
        {
            timeSlotA.SetIsEssentialItemsStore(Convert.ToBoolean(store.SellsEssentials));
            store.OpeningTime = timeSlotA.GetOpeningTime();
            store.ClosingTime = timeSlotA.GetClosingTime();
        }
        else if(timeSlot == "timeSlotB" || timeSlot == "TimeSlotB")
        {
            timeSlotB.SetIsEssentialItemsStore(Convert.ToBoolean(store.SellsEssentials));
            store.OpeningTime = timeSlotB.GetOpeningTime();
            store.ClosingTime = timeSlotB.GetClosingTime();
        }
        else
        {
            throw new Exception("Error : Invalid time slot");
        }
        StoresScheduleDAO storesScheduleDAO = new StoresScheduleDAO();

        int a = storesScheduleDAO.UpdateStore(store.StoreId, store.OpeningTime,
store.ClosingTime);
        if (a > 0)
            IsUpdated = true;

        return IsUpdated;
    }
}

```

```

StoresScheduleDAO.cs.....
.....
using System;
using System.Collections.Generic;
using System.Data;

```

```

using System.Data.SqlClient;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Configuration;

namespace StoresScheduleSystemDAL
{
    public class StoresScheduleDAO
    {
        public SqlConnection connection;
        public SqlCommand command;
        public SqlDataAdapter adapter;
        public string connstring =
ConfigurationManager.ConnectionStrings["StoresConnection"].ConnectionString;

        public StoresScheduleDAO()
        {
            // Instantiate fields here
        }

        public DataTable FindStore(int storeId)
        {
            DataTable table = new DataTable();

            connection = new SqlConnection(connstring);
            string query = "select * from Stores where storeId= " + storeId;

            connection.Open();

            adapter = new SqlDataAdapter(query, connection);

            adapter.Fill(table);

            connection.Close();

            return table;
        }

        public int UpdateStore(int storeId, string openingTime, string closingTime)
        {

```

```

        int RowsAffected;

        connection = new SqlConnection(connstring);
        string query = "update dbo.Stores set OpeningTime= " + openingTime + "
,ClosingTime= " + closingTime + " where StoreId= " + storeId;
        connection.Open();
        command = new SqlCommand(query, connection);
        RowsAffected = command.ExecuteNonQuery();
        connection.Close();

        return RowsAffected;
    }
}

```

TimeSlot.cs.....

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace StoresScheduleSystemBLL
{
    public class TimeSlot
    {
        protected bool SellsEssentialItems;

        public void SetIsEssentialItemsStore(bool isEssentialItemsStore)
        {
            SellsEssentialItems = isEssentialItemsStore;
        }

        public virtual string GetOpeningTime()
        {
            return "08:00 AM";
        }

        public virtual string GetClosingTime()

```

```

        {
            return "08:00 PM";
        }
    }
}
TimeSlotA.cs.....
.....

```

```
using System;
```

```
namespace StoresScheduleSystemBLL
```

```

{
    public class TimeSlotA:TimeSlot
    {

        public override string GetOpeningTime()
        {
            // Donot change method signature
            // Implement code here

            if (SellsEssentialItems)
            {
                return "08:00 AM";
            }
            else
            {
                return "10:00 AM";
            }

        }

        public override string GetClosingTime()
        {

            if (SellsEssentialItems)
            {

                return "02:00 PM";
            }
            else
            {
                return "02:00 PM";
            }
        }
    }
}

```

```

    }

    }
}

TimeSlotB.cs.....
.....

```

```
using System;
```

```

namespace StoresScheduleSystemBLL
{
    public class TimeSlotB:TimeSlot
    {

        public override string GetOpeningTime()
        {

            if (SellsEssentialItems)
            {
                return "02:00 PM";
            }
            else
            {
                return "04:00 PM";
            }

        }

        public override string GetClosingTime()
        {

            if (SellsEssentialItems)
            {
                return "08:00 PM";
            }
            else
            {
                return "08:00 PM";
            }

        }

    }
}

```

}
}

Automatic evaluation[+]

ProjectAllocationRefactoring/src/project/Employee.java

```
1 package project;
2 public class Employee
3 {
4     private String employeeId;
5     private String employeeName;
6     private String emailId;
7     private String designation;
8     public Employee(String employeeId, String employeeName,
9 String emailId, String designation) {
10 this.employeeId = employeeId;
11 this.employeeName = employeeName;
12 this.emailId = emailId;
13 this.designation = designation;
14 }
15 public String getEmployeeId() {
16 return employeeId;
17 }
18 public void setEmployeeId(String employeeId) {
19 this.employeeId = employeeId;
20 }
21 public String getEmployeeName() {
22 return employeeName;
23 }
24 public void setEmployeeName(String employeeName) {
25 this.employeeName = employeeName;
26 }
27 public String getEmailId() {
28 return emailId;
29 }
30 public void setEmailId(String emailId) {
31 this.emailId = emailId;
32 }
33 public String getDesignation() {
34 return designation;
35 }
36 public void setDesignation(String designation) {
37 this.designation = designation;
38 }
39 @Override
40 public String toString() {
41     return "Employee [employeeId=" + employeeId + ",employeeName=" +
employeeName + ", emailId=" + emailId + ", designation=" + designation + "]",
42 }
43 }
```

ProjectAllocationRefactoring/src/project/EmployeeDAO.java

```
1 package project;
2 import java.util.ArrayList;
3 import java.util.List;
4 public class EmployeeDAO {
5     private final List<Employee> employeeList = new
6 ArrayList<>();
7     public void addEmployee(Employee employee) {
8 employeeList.add(employee);
9 }
10 public void removeEmployee(Employee employee) {
```



```

11 employeeList.remove(employee);
12 }
13 public void viewEmployee() {
14 for (Employee employee : employeeList) {
15 System.out.println("Employee Id:" +
16 employee.getEmployeeId());
17 System.out.println("Employee Name:" +
18 employee.getEmployeeName());
19 System.out.println("Email Id:" +
20 employee.getEmailId());
21 System.out.println("Designation: " +
22 employee.getDesignation());
23 }
24 }
25 }

```

ProjectAllocationRefactoring/src/project/Project.java

```

1 package project;
2
3 public class Project{
4     String projectId;
5     String projectName = new String("");
6     String projectManagerName;
7     int duration;
8     String startDate;
9     String endDate;
10
11     public Project(){
12
13     }
14     public Project(String projectId, String projectName, String projectManagerName, int
duration, String startDate,String endDate) {
15         super();
16         this.projectId = projectId;
17         this.projectName = projectName;
18         this.projectManagerName = projectManagerName;
19         this.duration = duration;
20         this.startDate = startDate;
21         this.endDate = endDate;
22     }
23     public String getProjectId() {
24     return projectId;
25 }
26     public void setProjectId(String projectId) {
27     this.projectId = projectId;
28 }
29     public String getProjectName() {
30     return projectName;
31 }
32     public void setProjectName(String projectName) {
33     this.projectName = projectName;
34 }
35     public String getProjectManagerName() {
36     return projectManagerName;
37 }
38     public void setProjectManagerName(String projectManagerName)
39 {
40     this.projectManagerName = projectManagerName;

```



```

41 }
42 public int getDuration() {
43 return duration;
44 }
45 public void setDuration(int duration) {
46 this.duration = duration;
47 }
48 public String getStartDate() {
49 return startDate;
50 }
51 public void setStartDate(String startDate) {
52 this.startDate = startDate;
53 }
54 public String getEndDate() {
55 return endDate;
56 }
57 public void setEndDate(String endDate) {
58 this.endDate = endDate;
59 }
60 @Override
61 public String toString() {
62 return "Project [projectId=" + projectId + ", projectName=" + projectName + ",
projectManagerName="+ projectManagerName + ", duration=" +duration + ", startDate=" +
startDate + ", endDate=" + endDate+ "]";
63 }
64 }
65
66
67
68
69

```

ProjectAllocationRefactoring/src/project/ProjectAllocation.java

```

1 package project;
2 import java.util.Date;
3 public class ProjectAllocation {
4 private Employee employee;
5 private Project project;
6 private int projectAllocationId;
7 private String moduleName;
8 private static final int NO_OF_PROJECTS_WORKING_IN_PARALLEL
9 = 0;
10 private Date allocationDate;
11 private static final int NO_OF_HOURS_ALLOCATED = 160;
12 public ProjectAllocation(Employee employee, Project project,
13 int projectAllocationId, String moduleName,
14 Date allocationDate) {
15 this.employee = employee;
16 this.project = project;
17 this.projectAllocationId = projectAllocationId;
18 this.moduleName = moduleName;
19 this.allocationDate = allocationDate;
20 }
21 public Employee getEmployee() {
22 return employee;
23 }
24 public void setEmployee(Employee employee) {
25 this.employee = employee;

```



```

26 }
27 public Project getProject() {
28 return project;
29 }
30 public void setProject(Project project) {
31 this.project = project;
32 }
33 public int getProjectAllocationId() {
34 return projectAllocationId;
35 }
36 public void setProjectAllocationId(int projectAllocationId)
37 {
38 this.projectAllocationId = projectAllocationId;
39 }
40 public String getModuleName() {
41 return moduleName;
42 }
43 public void setModuleName(String moduleName) {
44 this.moduleName = moduleName;
45 }
46 public Date getAllocationDate() {
47 return allocationDate;
48 }
49 public void setAllocationDate(Date allocationDate) {
50 this.allocationDate = allocationDate;
51 }
52 public static int getNoOfProjectsWorkingInParallel() {
53 return NO_OF_PROJECTS_WORKING_IN_PARALLEL;
54 }
55 public static int getNoOfHoursAllocated() {
56 return NO_OF_HOURS_ALLOCATED;
57 }
58 @Override
59 public String toString() {
60 return "ProjectAllocation [employee=" + employee + ",project=" + project + ",
projectAllocationId="+ projectAllocationId + ", moduleName=" +moduleName + ",
allocationDate=" + allocationDate + "];
61 }
62 }

```

ProjectAllocationRefactoring/src/project/ProjectAllocationDAO.java

```

1 package project;
2 import java.util.ArrayList;
3 import java.util.List;
4 public class ProjectAllocationDAO {
5 private final List<ProjectAllocation> projectAllocationList
6 = new ArrayList<>();
7 public void addProjectAllocation(ProjectAllocation
8 projectAllocation) {
9 projectAllocationList.add(projectAllocation);
10 }
11 public void removeProjectAllocation(ProjectAllocation
12 projectAllocation) {
13 projectAllocationList.remove(projectAllocation);
14 }
15 public void viewProjectAllocation() {
16 if (projectAllocationList.isEmpty())
17 {

```

```

18 System.out.println("Project Allocation List is empty");
19 }
20 else {
21 for (ProjectAllocation projectAllocation :
22 projectAllocationList) {
23 System.out.println("Project Allocation Id:"
24 + projectAllocation.getProjectAllocationId());
25 System.out.println("Project Id:" +
26 projectAllocation.getProject().getProjectId());
27 System.out.println("Employee Id:" +
28 projectAllocation.getEmployee().getEmployeeId());
29 System.out.println("Allocation Date:" +
30 projectAllocation.getAllocationDate());
31 System.out.println("Module Name:" +
32 projectAllocation.getModuleName());
33 }
34 }
35 }
36 }

```

ProjectAllocationRefactoring/src/project/ProjectDAO.java

```

1 package project;
2 import java.util.ArrayList;
3 import java.util.List;
4 public class ProjectDAO {
5 private final List<Project> projectList = new ArrayList<>();
6 public void addProject(Project project) {
7 projectList.add(project);
8 }
9 public void removeProject(Project project) {
10 projectList.remove(project);
11 }
12 public void viewProject() {
13 for (Project project : projectList) {
14 System.out.println("Project Id:" +
15 project.getProjectId());
16 System.out.println("Project Name:" +
17 project.getProjectName());
18 System.out.println("Project Manager Name:" +
19 project.getProjectManagerName());
20 System.out.println("Duration:" +
21 project.getDuration());
22 System.out.println("Start Date:" +
23 project.getStartDate());
24 System.out.println("End Date:" +
25 project.getEndDate());
26 }
27 }
28 }

```

Grade

Reviewed on Thursday, 11 March 2021, 10:30 AM by Automatic grade

Grade 92.73 / 100

Assessment report

[+]SOURCE CODE ANALYZER REPORT

[+]Grading and Feedback



Automatic evaluation[-]

Proposed grade: 100.0 / 100

Result Description

[+]Grading and Feedback

ELearningApp/src/com/dao/Academy.java

```
1 package com.dao;
2
3
4 import java.util.List;
5
6 import com.exception.InvalidCourseException;
7 import com.exception.InvalidStudentException;
8 import com.model.Course;
9 import com.model.Student;
10
11
12
13 public class Academy {
14
15     final StudentDAO studentDAO=new StudentDAO();
16     final CourseDAO courseDAO = new CourseDAO();
17
18
19     public void addStudent(Student studentObj){
20         studentDAO.addStudent(studentObj);
21     }
22
23     public Student viewStudentById(int studentid) throws InvalidStudentException{
24         return studentDAO.viewStudentById(studentid);
25     }
26
27     public void addCourse(Course courseObj){
28         courseDAO.addCourse(courseObj);
29     }
30
31     public List<Course> viewCourseByFees(float fees) throws InvalidCourseException {
32         return courseDAO.viewCourseByFees(fees);
33     }
34
35
36 }
37
```

ELearningApp/src/com/dao/CourseDAO.java

```
1 package com.dao;
2
3 import java.util.ArrayList;
4 import java.util.List;
5
6 import com.exception.InvalidCourseException;
7
8 import com.model.Course;
9
10 public class CourseDAO {
11
12     List<Course> courseList = new ArrayList<>();
13
14     public void addCourse(Course courseObj){
```



```

15     courseList.add(courseObj);
16 }
17
18 public List<Course> viewCourseByFees(float fees) throws InvalidCourseException {
19     List<Course> temp=new ArrayList<>();
20     for(Course c : courseList){
21         if(c.getFees()>=fees)
22         {
23             temp.add(c);
24         }
25     }
26     if(temp==null) //to check of the size of the list is 0
27     {
28         throw new InvalidCourseException("No course with fees greater than
"+fees);
29     }
30     else
31     {
32         return temp;
33     }
34 }
35
36 }
37

```

ELearningApp/src/com/dao/RegistrationDAO.java

```

1 package com.dao;
2
3 import java.time.LocalDate;
4 import java.util.ArrayList;
5 import java.util.List;
6
7 import com.model.Course;
8 import com.model.Registration;
9 import com.model.Student;
10
11 public class RegistrationDAO {
12
13     List<Registration> regList = new ArrayList<>();
14
15     public void registerStudentToCourse(Student student,Course course,LocalDate dor){
16
17         Registration r = new Registration(student,course,dor,'X');
18
19         int count=0;
20
21
22         for(Registration reg : regList)
23         {
24             if(reg.getCourseObj().courseId==course.courseId)
25             {
26                 count++;
27             }
28         }
29         if(course.getMaxstrengthpermitted()>count)
30         {
31             regList.add(r);
32         }
33     }
34 }
35

```

```

33 }
34
35 }
36
ELearningApp/src/com/dao/StudentDAO.java
1 package com.dao;
2
3 import java.util.ArrayList;
4 import java.util.List;
5
6 import com.exception.InvalidStudentException;
7 import com.model.Student;
8
9 public class StudentDAO {
10
11     List<Student> studentList = new ArrayList<>();
12
13     public void addStudent(Student studentObj){
14         studentList.add(studentObj);
15     }
16
17     public Student viewStudentById(int studentid) throws InvalidStudentException{
18         if(studentList.isEmpty())
19         {
20             throw new InvalidStudentException("Student list is empty");
21         }
22         else
23         {
24             ;
25
26             for(Student s : studentList)
27             {
28                 if(s.getStudentId().equals(studentid))
29                 {
30                     return s;
31                 }
32             }
33         }
34         return null;
35     }
36 }
37
ELearningApp/src/com/exception/InvalidCourseException.java
1 package com.exception;
2
3 public class InvalidCourseException extends Exception {
4
5     public InvalidCourseException(String msg){
6         super(msg);
7     }
8
9 }
10
ELearningApp/src/com/exception/InvalidStudentException.java
1 package com.exception;
2
3 public class InvalidStudentException extends Exception {

```



```

4
5 public InvalidStudentException(String msg){
6     super(msg);
7 }
8
9 }
10
ELearningApp/src/com/model/Course.java
1 package com.model;
2
3 public class Course {
4
5     public int courseId;
6     private String courseName;
7     private float fees;
8     private int duration = 10;
9     private String trainerIncharge;
10    static final int MAXSTRENGTHPERMITTED = 100;
11    static String academyName="Akshara Acadey";
12
13    public int getCourseId() {
14        return courseId;
15    }
16    public void setCourseId(int courseId) {
17        this.courseId = courseId;
18    }
19    public String getCourseName() {
20        return courseName;
21    }
22    public void setCourseName(String courseName) {
23        this.courseName = courseName;
24    }
25    public float getFees() {
26        return fees;
27    }
28    public void setFees(float fees) {
29        this.fees = fees;
30    }
31    public int getDuration() {
32        return duration;
33    }
34    public void setDuration(int duration) {
35        this.duration = duration;
36    }
37    public String getTrainerIncharge() {
38        return trainerIncharge;
39    }
40    public void setTrainerIncharge(String trainerIncharge) {
41        this.trainerIncharge = trainerIncharge;
42    }
43    public static int getMaxstrengthpermitted() {
44        return MAXSTRENGTHPERMITTED;
45    }
46
47 }
48
ELearningApp/src/com/model/Registration.java

```

```

1 package com.model;
2
3 import java.time.LocalDate;
4
5 public class Registration {
6
7     private Student studentObj;
8     private Course courseObj;
9     private LocalDate dateOfRegistration;
10    private char grade;
11
12
13    public Registration(Student studentObj, Course courseObj, LocalDate
dateOfRegistration, char grade) {
14        this.studentObj = studentObj;
15        this.courseObj = courseObj;
16        this.dateOfRegistration = dateOfRegistration;
17        this.grade = grade;
18    }
19
20    public Student getStudentObj() {
21        return studentObj;
22    }
23
24    public void setStudentObj(Student studentObj) {
25        this.studentObj = studentObj;
26    }
27
28    public Course getCourseObj() {
29        return courseObj;
30    }
31
32    public void setCourseObj(Course courseObj) {
33        this.courseObj = courseObj;
34    }
35
36    public LocalDate getDateOfRegistration() {
37        return dateOfRegistration;
38    }
39
40    public void setDateOfRegistration(LocalDate dateOfRegistration) {
41        this.dateOfRegistration = dateOfRegistration;
42    }
43
44    public char getGrade() {
45        return grade;
46    }
47
48    public void setGrade(char grade) {
49        this.grade = grade;
50    }
51
52    public void calculateGrade(int mark){
53
54        if(mark >= 90)
55        {
56            setGrade('O');

```

```

57     }
58     else if(mark >= 70)
59     {
60         setGrade('A');
61     }
62     else
63     {
64         setGrade('B');
65     }
66 }
67 }
68
ELearningApp/src/com/model/Student.java
1 package com.model;
2
3 public class Student {
4
5     private String studentId;
6     private String studentName;
7     private String phoneNumber;
8     private String emailId;
9
10    public Student(String studentId, String studentName, String phoneNumber, String
emailId) {
11        this.studentId = studentId;
12        this.studentName = studentName;
13        this.phoneNumber = phoneNumber;
14        this.emailId = emailId;
15    }
16
17    public String getStudentId() {
18        return studentId;
19    }
20
21    public void setStudentId(String studentId) {
22        this.studentId = studentId;
23    }
24
25    public String getStudentName() {
26        return studentName;
27    }
28
29    public void setStudentName(String studentName) {
30        this.studentName = studentName;
31    }
32
33    public String getPhoneNumber() {
34        return phoneNumber;
35    }
36
37    public void setPhoneNumber(String phoneNumber) {
38        this.phoneNumber = phoneNumber;
39    }
40
41    public String getEmailId() {
42        return emailId;
43    }

```



```
44
45 public void setEmailld(String emailld) {
46     this.emailld = emailld;
47 }
48
49 }
50
```

Grade

Reviewed on Tuesday, 4 May 2021, 2:25 AM by Automatic grade

Grade 100 / 100

Assessment report

[-]Grading and Feedback

Good Programming Practice - 100.0 / 100(Success)

