Automatic evaluation[+]
LoanManagement/src/com/dao/CustomerDAO.java

```java
package com.dao;

import java.util.*;

import com.exception.LoanException;
import com.model.Customer;

public class CustomerDAO {

	List<Customer> customerList = new ArrayList<>();

	public void addCustomer(Customer customerObj){
		customerList.add(customerObj);
	}

	public List<Customer> viewAllCustomer(){
		if(customerList.isEmpty()){
			return null;
		}
		return customerList;
	}

	public boolean deleteCustomer(int customerId){
		//boolean flag=false;
		for(Customer c : customerList){
			if(c.getCustomerId()==customerId){
				customerList.remove(c);
				return true;
			}
		}
		return false;
	}

	public boolean validateCustomerID(int customerId){

		try {
			if(customerId>=1000 && customerId<=9999)
			{
				return true;
			}
			else{
				throw new LoanException("Customer Id is invalid");
			}
		} catch (LoanException e) {
			// TODO Auto-generated catch block
			e.printStackTrace();
		}
		return false;
	}

}

```

LoanManagement/src/com/dao/LoanDAO.java

```java
package com.dao;

```

```

import java.util.ArrayList;
import java.util.List;

import com.exception.LoanException;
//import com.model.Customer;
import com.model.Loan;

//import java.io.IOException;

public class LoanDAO {

    List<Loan> loanList = new ArrayList<>();

    public void issueLoan(Loan loanObj){
        loanList.add(loanObj);
    }

    public List<Loan> viewLoanByType(int loanType) throws LoanException {

        if(loanList.isEmpty()){
            return null;
        }
        else
        {
            List<Loan> temp= new ArrayList<>();
            for(Loan loan : loanList){
                if(loan.getLoanType().equals(loanType)){
                    temp.add(loan);
                }
            }

            //try{
                if(temp==null)
                {
                    // to check if temp is empty
                    throw new LoanException("No loans available with type "+loanType);
                }
            //}
            //catch(LoanException e){

            //}
            /*inally{
                return temp;
            }*/
            else{
                return temp;
            }
        }
    }

}

```
LoanManagement/src/com/dao/PaymentDAO.java

```
package com.dao;

```

```java
import java.util.ArrayList;
import java.util.List;

import com.model.Payment;
//import com.dao.LoanDAO;
//import com.dao.CustomerDAO;

public class PaymentDAO {

    private List<Payment> paymentList = new    ArrayList<>();

    //private CustomerDAO custDAO;


    public void setPaymentList(List<Payment> paymentList ) {
        this.paymentList= paymentList;
    }

    public List<Payment> getPaymentList() {
        return paymentList;
    }

      public void makePayment(Payment obj){
        paymentList.add(obj);
      }

     /* public void updatePayment(int paymentId,double amount){

      } */

}

```

LoanManagement/src/com/exception/LoanException.java

```java
package com.exception;

public class LoanException extends Exception {
    public LoanException(String msg){
        super(msg);
    }
}

```

LoanManagement/src/com/model/Customer.java

```java
package com.model;

public class Customer {

    int customerId;
    String customerName;
    String address;
    String panNumber;
    String emailId;

    public Customer(int customerId, String customerName, String address, String panNumber) {
        //String pan="";
        this.customerId = customerId;
        this.customerName = customerName;
```

```
		this.address = address;
		this.panNumber = panNumber;
	}

	public Customer() {
		//default
	}

	public int getCustomerId() {
		return customerId;
	}

	public void setCustomerId(int customerId) {
		this.customerId = customerId;
	}

	public String getCustomerName() {
		return customerName;
	}

	public void setCustomerName(String customerName) {
		this.customerName = customerName;
	}

	public String getAddress() {
		return address;
	}

	public void setAddress(String address) {
		this.address = address;
	}

	public String getPanNumber() {
		return panNumber;
	}

	public void setPanNumber(String panNumber) {
		this.panNumber = panNumber;
	}

	public String getEmailId() {
		return emailId;
	}

	public void setEmailId(String emailId) {
		this.emailId = emailId;
	}

}

```
LoanManagement/src/com/model/Loan.java
```
package com.model;

public class Loan {
	int loanNumber;
	String loanType="Vehicle";
	Customer customer;
```

```java
double loanAmount;
double balanceLoanAmount;
static final double MAXLOANAMOUNT=1500000;

public Loan(){
        //default
}

public Loan(int loanNumber, String loanType, Customer customer, double loanAmount) {
        this.loanNumber = loanNumber;
        this.loanType = loanType;
        this.customer = customer;
        this.loanAmount = loanAmount;
        balanceLoanAmount=loanAmount;
}

public int getLoanNumber() {
        return loanNumber;
}

public void setLoanNumber(int loanNumber) {
        this.loanNumber = loanNumber;
}

public String getLoanType() {
        return loanType;
}

public void setLoanType(String loanType) {
        this.loanType = loanType;
}

public Customer getCustomer() {
        return customer;
}

public void setCustomer(Customer customer) {
        this.customer = customer;
}

public double getLoanAmount() {
        return loanAmount;
}

public void setLoanAmount(double loanAmount) {
        this.loanAmount = loanAmount;
}

public double getBalanceLoanAmount() {
        return balanceLoanAmount;
}

public void setBalanceLoanAmount(double balanceLoanAmount) {
        this.balanceLoanAmount = balanceLoanAmount;
}

```

```

    public boolean checkBalanceAmount(double amountPaid){
        //float gst=3;
        return (balanceLoanAmount >= amountPaid);
        //double amount = balanceLoanAmount+balanceLoanAmount*gst/100;
        //if(balanceLoanAmount >= amountPaid)
        //    return true;
        //else
        //    return false;
    }




}

```
LoanManagement/src/com/model/Payment.java
```
package com.model;

import java.time.LocalDate;

public class Payment {

    int paymentId;
    LocalDate dateOfPayment;
    Loan loanObj;
    double amount;
    String paymentMode;
    static final float GSTPERCENTAGE=2;

    public Payment(int paymentId, LocalDate dateOfPayment, Loan loanObj, double amount, String paymentMode) {

        this.paymentId = paymentId;
        this.dateOfPayment = dateOfPayment;
        this.loanObj = loanObj;
        this.amount = amount;
        this.paymentMode = paymentMode;
    }

    public int getPaymentId() {
        return paymentId;
    }

    public void setPaymentId(int paymentId) {
        this.paymentId = paymentId;
    }

    public LocalDate getDateOfPayment() {
        return dateOfPayment;
    }

    public void setDateOfPayment(LocalDate dateOfPayment) {
        this.dateOfPayment = dateOfPayment;
    }

    public Loan getLoanObj() {
        return loanObj;
    }

    public void setLoanObj(Loan loanObj) {
        this.loanObj = loanObj;
    }

    public double getAmount() {
        return amount;
    }

    public void setAmount(double amount) {
        this.amount = amount;
    }

    public String getPaymentMode() {
        return paymentMode;
    }

    public void setPaymentMode(String paymentMode) {
        this.paymentMode = paymentMode;
    }

    public static float getGstpercentage() {
        return GSTPERCENTAGE;
    }



}

```
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

```
package com.dao;

import java.time.LocalDate;
import java.util.ArrayList;
import java.util.List;

import com.model.Flight;
import com.exception.InvalidFlightException;
import com.model.BookFlight;
import com.model.Customer;

public class BookFlightDAO {

    List<BookFlight> bookingList = new ArrayList<>();

    public boolean bookAFlight(final Customer customer,final Flight flight,final LocalDate dateOfbooking, LocalDate flightDate,int noOfPassengers){

        BookFlight bookObj = new BookFlight(customer,flight,dateOfbooking,flightDate,noOfPassengers);

        //boolean flag=true;
        //boolean result = bookingList.add(bookObj);
        //return result;
        return bookingList.add(bookObj);
    }

    public List<BookFlight>viewBookingByFlight(Flight flightObj) throws InvalidFlightException {
    final List<BookFlight> temp = new ArrayList<>();

        for(final BookFlight booking : bookingList){
            if(booking.getFlight().getFlightId()==flightObj.getFlightId()) {
                temp.add(booking);
            }
        }
        if(temp.isEmpty()){
            throw new InvalidFlightException("No booking for Flight "+flightObj.getFlightId());
        }
        return temp;

    }
    }



```

FlightMgmt/src/com/dao/CustomerDAO.java

```
package com.dao;

import java.util.*;

import com.exception.InvalidCustomerException;
import com.model.Customer;

```

```
public class CustomerDAO {

    List<Customer>customerList = new ArrayList<>();

    public void addCustomer(Customer customer){
        customerList.add(customer);
    }

    public Customer viewCustomerByUserName(String userName){
        try{
            if(customerList.isEmpty()){
                throw new InvalidCustomerException("User Name is invalid");
            }
            else{
                for(Customer c : customerList) {
                    if(c.getUserName().equals(userName)){
                        return c;
        }
                }
        }
    }
        catch(Exception e){
            System.out.println(e.getMessage());
        }
        return null;
    }

    public boolean validateCustomer(String userName,String password){
        //boolean flag=false;

        for(Customer c : customerList) {
            if(c.getUserName().equals(userName) && c.getPassword().equals(password)) {
                return true;
            }
        }

        return false;
    }

}

```

FlightMgmt/src/com/dao/FlightDAO.java

```
package com.dao;

import java.util.*;

import com.exception.*;
import com.model.*;

public class FlightDAO {

    List<Flight>flightList = new ArrayList<>();

    public void addCourse(Flight flightObj){
        flightList.add(flightObj);
    }

    public List<Flight>viewFligtBySourceDestination(String source,String destination) throws InvalidFlightException {
        List<Flight>temp=new ArrayList<>();
        for(Flight f : flightList) {
            if(f.getSource().equals(source) && f.getDestination().equals(destination)) {
                temp.add(f);
            }
        }
        if(temp.isEmpty()){
            throw new InvalidFlightException("No Flight with source "+source+" and destination "+destination);
        }
        else{
            return temp;
    }
    }

}

```

FlightMgmt/src/com/exception/InvalidCustomerException.java

```
package com.exception;

public class InvalidCustomerException extends Exception {

    public InvalidCustomerException(String msg){
        super(msg);
    }

}

```

FlightMgmt/src/com/exception/InvalidFlightException.java

```
package com.exception;

public class InvalidFlightException extends Exception {

    public InvalidFlightException(String msg){
        super(msg);
    }

}

```

FlightMgmt/src/com/model/BookFlight.java

```
package com.model;

import java.time.LocalDate;
//import java.util.Date;

public class BookFlight {

    private Customer customer;
    private Flight flight;
    private LocalDate dateOfbooking;
    private LocalDate flightDate;
    private int noOfPassengers;
    private double totalFare;

```

```
	//public BookFlight(){
	//
	//}

	public BookFlight(Customer customer, Flight flight, LocalDate dateOfbooking, LocalDate flightDate,
			int noOfPassengers) {
		super();
		this.customer = customer;
		this.flight = flight;
		this.dateOfbooking = dateOfbooking;
		this.flightDate = flightDate;
		this.noOfPassengers = noOfPassengers;
		this.totalFare = totalFare;
	}

	public Customer getCustomer() {
		return customer;
	}

	public void setCustomer(Customer customer) {
		this.customer = customer;
	}

	public Flight getFlight() {
		return flight;
	}

	public void setFlight(Flight flight) {
		this.flight = flight;
	}

	public LocalDate getDateOfbooking() {
		return dateOfbooking;
	}

	public void setDateOfbooking(LocalDate dateOfbooking) {
		this.dateOfbooking = dateOfbooking;
	}

	public LocalDate getFlightDate() {
		return flightDate;
	}

	public void setFlightDate(LocalDate flightDate) {
		this.flightDate = flightDate;
	}

	public int getNoOfPassengers() {
		return noOfPassengers;
	}

	public void setNoOfPassengers(int noOfPassengers) {
		this.noOfPassengers = noOfPassengers;
	}

	public double getTotalFare() {
        return totalFare;
    }

    public void setTotalFare(double totalFare) {
        this.totalFare = totalFare;
    }

    public float calculateTotalFare(){
        return (float)noOfPassengers*flight.getFlightFare();
        //return totalFare;
    }

    public boolean validateNoOfPassengers(){
        boolean flag=false;
        if(noOfPassengers >0 && noOfPassengers <= 30) {
            flag=true;
        }
            return flag;
    }
}

```
FlightMgmt/src/com/model/Customer.java
```
package com.model;

public class Customer {

    private String customerId;
    private String customerName;
    private String emailId;
    private String userName;
    private String password="FH782";

    public Customer(String customerName, String emailId, String userName) {
        this.customerId = customerId;
        this.customerName = customerName;
        this.emailId = emailId;
        this.userName = userName;
    }

    public String getCustomerId() {
        return customerId;
    }

    public void setCustomerId(String customerId) {
        this.customerId = customerId;
    }

    public String getCustomerName() {
        return customerName;
    }

    public void setCustomerName(String customerName) {
        this.customerName = customerName;
    }

    public String getEmailId() {
        return emailId;
```

```java
    }

    public void setEmailId(String emailId) {
        this.emailId = emailId;
    }

    public String getUserName() {
        return userName;
    }

    public void setUserName(String userName) {
        this.userName = userName;
    }

    public String getPassword() {
        return password;
    }

    public void setPassword(String password) {
        this.password = password;
    }



}

```

FlightMgmt/src/com/model/Flight.java

```java
package com.model;

public class Flight {

    public int flightId;
    private String flightName;
    private String source;
    private String destination;
    private float flightFare;
    private int noOfSeats = 40;
    static final String COMPANYNAME = "Aviva Airlines";

    public Flight(int flightId, String flightName, String source, String destination, float fare) {

        this.flightId = flightId;
        this.flightName = flightName;
        this.source = source;
        this.destination = destination;
        this.flightFare = fare;

    }

    public int getFlightId() {
        return flightId;
    }

    public void setFlightId(int flightId) {
        this.flightId = flightId;
    }

    public String getFlightName() {
        return flightName;
    }

    public void setFlightName(String flightName) {
        this.flightName = flightName;
    }

    public String getSource() {
        return source;
    }

    public void setSource(String source) {
        this.source = source;
    }

    public String getDestination() {
        return destination;
    }

    public void setDestination(String destination) {
        this.destination = destination;
    }

    public float getFlightFare() {
        return flightFare;
    }

    public void setFare(float fare) {
        this.flightFare = fare;
    }

    public int getNoOfSeats() {
        return noOfSeats;
    }

    public void setNoOfSeats(int noOfSeats) {
        this.noOfSeats = noOfSeats;
    }

    public static String getCompanyname() {
        return COMPANYNAME;
    }
}

```

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; }
    }
}

StoreSchedular.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

```java
package project;
public class Employee
{
private String employeeId;
private String employeeName;
private String emailId;
private String designation;
public Employee(String employeeId, String employeeName,
String emailId, String designation) {
this.employeeId = employeeId;
this.employeeName = employeeName;
this.emailId = emailId;
this.designation = designation;
}
public String getEmployeeId() {
return employeeId;
}
public void setEmployeeId(String employeeId) {
this.employeeId = employeeId;
}
public String getEmployeeName() {
return employeeName;
}
public void setEmployeeName(String employeeName) {
this.employeeName = employeeName;
}
public String getEmailId() {
return emailId;
}
public void setEmailId(String emailId) {
this.emailId = emailId;
}
public String getDesignation() {
return designation;
}
public void setDesignation(String designation) {
this.designation = designation;
}
@Override
public String toString() {
return "Employee [employeeId=" + employeeId + ",employeeName=" + employeeName + ", emailId=" + emailId+ ", designation=" + designation + "]";
}
}
```

ProjectAllocationRefactoring/src/project/EmployeeDAO.java

```java
package project;
import java.util.ArrayList;
import java.util.List;
public class EmployeeDAO {
private final List<Employee> employeeList = new
ArrayList<>();
public void addEmployee(Employee employee) {
employeeList.add(employee);
}
public void removeEmployee(Employee employee) {
```

```
employeeList.remove(employee);
}
public void viewEmployee() {
for (Employee employee : employeeList) {
System.out.println("Employee Id:" +
employee.getEmployeeId());
System.out.println("Employee Name:" +
employee.getEmployeeName());
System.out.println("Email Id:" +
employee.getEmailId());
System.out.println("Designation: " +
employee.getDesignation());
}
}
}
```

ProjectAllocationRefactoring/src/project/Project.java

```
package project;

public class Project{
    String projectId;
    String projectName = new String("");
    String projectManagerName;
    int duration;
    String startDate;
    String endDate;

    public Project(){

    }
    public Project(String projectId, String projectName, String projectManagerName, int duration, String startDate,String endDate) {
            super();
            this.projectId = projectId;
            this.projectName = projectName;
            this.projectManagerName = projectManagerName;
            this.duration = duration;
            this.startDate = startDate;
            this.endDate = endDate;
    }
public String getProjectId() {
return projectId;
}
public void setProjectId(String projectId) {
this.projectId = projectId;
}
public String getProjectName() {
return projectName;
}
public void setProjectName(String projectName) {
this.projectName = projectName;
}
public String getProjectManagerName() {
return projectManagerName;
}
public void setProjectManagerName(String projectManagerName)
{
this.projectManagerName = projectManagerName;
}
public int getDuration() {
return duration;
}
public void setDuration(int duration) {
this.duration = duration;
}
public String getStartDate() {
return startDate;
}
public void setStartDate(String startDate) {
this.startDate = startDate;
}
public String getEndDate() {
return endDate;
}
public void setEndDate(String endDate) {
this.endDate = endDate;
}
@Override
public String toString() {
return "Project [projectId=" + projectId + ",projectName=" + projectName + ", projectManagerName="+ projectManagerName + ", duration=" +duration + ", startDate=" + startDate + ", endDate=" + endDate+ "]";
}
}





```
ProjectAllocationRefactoring/src/project/ProjectAllocation.java

```
package project;
import java.util.Date;
public class ProjectAllocation {
private Employee employee;
private Project project;
private int projectAllocationId;
private String moduleName;
private static final int NO_OF_PROJECTS_WORKING_IN_PARALLEL
= 0;
private Date allocationDate;
private static final int NO_OF_HOURS_ALLOCATED = 160;
public ProjectAllocation(Employee employee, Project project,
int projectAllocationId, String moduleName,
Date allocationDate) {
this.employee = employee;
this.project = project;
this.projectAllocationId = projectAllocationId;
this.moduleName = moduleName;
this.allocationDate = allocationDate;
}
public Employee getEmployee() {
return employee;
}
public void setEmployee(Employee employee) {
this.employee = employee;
}
public Project getProject() {
return project;
}
public void setProject(Project project) {
this.project = project;
}
public int getProjectAllocationId() {
return projectAllocationId;
}
public void setProjectAllocationId(int projectAllocationId)
{
this.projectAllocationId = projectAllocationId;
}
public String getModuleName() {
return moduleName;
}
public void setModuleName(String moduleName) {
this.moduleName = moduleName;
}
public Date getAllocationDate() {
return allocationDate;
}
public void setAllocationDate(Date allocationDate) {
this.allocationDate = allocationDate;
}
public static int getNoOfProjectsWorkingInParallel() {
return NO_OF_PROJECTS_WORKING_IN_PARALLEL;
}
public static int getNoOfHoursAllocated() {
return NO_OF_HOURS_ALLOCATED;
}
@Override
public String toString() {
return "ProjectAllocation [employee=" + employee + ",project=" + project + ", projectAllocationId="+ projectAllocationId + ", moduleName=" +moduleName + ", allocationDate=" + allocationDate + "]";
}
}
```

ProjectAllocationRefactoring/src/project/ProjectAllocationDAO.java

```
package project;
import java.util.ArrayList;
import java.util.List;
public class ProjectAllocationDAO {
private final List<ProjectAllocation> projectAllocationList
= new ArrayList<>();
public void addProjectAllocation(ProjectAllocation
projectAllocation) {
projectAllocationList.add(projectAllocation);
}
public void removeProjectAllocation(ProjectAllocation
projectAllocation) {
projectAllocationList.remove(projectAllocation);
}
public void viewProjectAllocation() {
if (projectAllocationList.isEmpty())
{
```

```
System.out.println("Project Allocation List is empty");
}
else {
for (ProjectAllocation projectAllocation :
projectAllocationList) {
System.out.println("Project Allocation Id:"
+ projectAllocation.getProjectAllocationId());
System.out.println("Project Id:" +
projectAllocation.getProject().getProjectId());
System.out.println("Employee Id:" +
projectAllocation.getEmployee().getEmployeeId());
System.out.println("Allocation Date:" +
projectAllocation.getAllocationDate());
System.out.println("Module Name:" +
projectAllocation.getModuleName());
}
}
}
}
```
ProjectAllocationRefactoring/src/project/ProjectDAO.java
```
package project;
import java.util.ArrayList;
import java.util.List;
public class ProjectDAO {
private final List<Project> projectList = new ArrayList<>();
public void addProject(Project project) {
projectList.add(project);
}
public void removeProject(Project project) {
projectList.remove(project);
}
public void viewProject() {
for (Project project : projectList) {
System.out.println("Project Id:" +
project.getProjectId());
System.out.println("Project Name:" +
project.getProjectName());
System.out.println("Project Manager Name:" +
project.getProjectManagerName());
System.out.println("Duration:" +
project.getDuration());
System.out.println("Start Date:" +
project.getStartDate());
System.out.println("End Date:" +
project.getEndDate());
}
}
}
```
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

```java
package com.dao;


import java.util.List;

import com.exception.InvalidCourseException;
import com.exception.InvalidStudentException;
import com.model.Course;
import com.model.Student;



public class Academy {

    final StudentDAO studentDAO=new StudentDAO();
    final CourseDAO courseDAO = new CourseDAO();


    public void addStudent(Student studentObj){
        studentDAO.addStudent(studentObj);
    }

    public Student viewStudentById(int studentid) throws InvalidStudentException{
        return studentDAO.viewStudentById(studentid);
    }

    public void addCourse(Course courseObj){
        courseDAO.addCourse(courseObj);
    }

    public List<Course> viewCourseByFees(float fees) throws InvalidCourseException {
        return courseDAO.viewCourseByFees(fees);
    }


}

```

ELearningApp/src/com/dao/CourseDAO.java

```java
package com.dao;

import java.util.ArrayList;
import java.util.List;

import com.exception.InvalidCourseException;

import com.model.Course;

public class CourseDAO {

    List<Course> courseList = new ArrayList<>();

    public void addCourse(Course courseObj){
        courseList.add(courseObj);
    }

    public List<Course> viewCourseByFees(float fees) throws InvalidCourseException {
        List<Course> temp=new ArrayList<>();
        for(Course c : courseList){
            if(c.getFees()>=fees)
            {
                temp.add(c);
            }
        }
        if(temp==null) //to check of the size of the list is 0
        {
            throw new InvalidCourseException("No course with fees greater than "+fees);
        }
        else
        {
            return temp;
        }
    }

}

```

ELearningApp/src/com/dao/RegistrationDAO.java

```java
package com.dao;

import java.time.LocalDate;
import java.util.ArrayList;
import java.util.List;

import com.model.Course;
import com.model.Registration;
import com.model.Student;

public class RegistrationDAO {

    List<Registration> regList = new ArrayList<>();

    public void registerStudentToCourse(Student student,Course course,LocalDate dor){

        Registration r = new Registration(student,course,dor,'X');

        int count=0;


        for(Registration reg : regList)
        {
            if(reg.getCourseObj().courseId==course.courseId)
            {
                count++;
            }
        }
        if(course.getMaxstrengthpermitted()>count)
        {
            regList.add(r);
        }
    }

}

```
ELearningApp/src/com/dao/StudentDAO.java
```java
package com.dao;

import java.util.ArrayList;
import java.util.List;

import com.exception.InvalidStudentException;
import com.model.Student;

public class StudentDAO {

    List<Student> studentList = new ArrayList<>();

    public void addStudent(Student studentObj){
        studentList.add(studentObj);
    }

    public Student viewStudentById(int studentid) throws InvalidStudentException{
        if(studentList.isEmpty())
        {
            throw new InvalidStudentException("Student list is empty");
        }
        else
        {
            ;

            for(Student s : studentList)
            {
                if(s.getStudentId().equals(studentid))
                {
                    return s;
                }
            }
        }
        return null;
    }
}

```
ELearningApp/src/com/exception/InvalidCourseException.java
```java
package com.exception;

public class InvalidCourseException extends Exception {

    public InvalidCourseException(String msg){
        super(msg);
    }

}

```
ELearningApp/src/com/exception/InvalidStudentException.java
```java
package com.exception;

public class InvalidStudentException extends Exception {

    public InvalidStudentException(String msg){
        super(msg);
    }

}

```
ELearningApp/src/com/model/Course.java

```java
package com.model;

public class Course {

    public int courseId;
    private String courseName;
    private float fees;
    private int duration = 10;
    private String trainerIncharge;
    static final int MAXSTRENGTHPERMITTED = 100;
    static String academyName="Akshara Acadey";

    public int getCourseId() {
        return courseId;
    }
    public void setCourseId(int courseId) {
        this.courseId = courseId;
    }
    public String getCourseName() {
        return courseName;
    }
    public void setCourseName(String courseName) {
        this.courseName = courseName;
    }
    public float getFees() {
        return fees;
    }
    public void setFees(float fees) {
        this.fees = fees;
    }
    public int getDuration() {
        return duration;
    }
    public void setDuration(int duration) {
        this.duration = duration;
    }
    public String getTrainerIncharge() {
        return trainerIncharge;
    }
    public void setTrainerIncharge(String trainerIncharge) {
        this.trainerIncharge = trainerIncharge;
    }
    public static int getMaxstrengthpermitted() {
        return MAXSTRENGTHPERMITTED;
    }

}

```
ELearningApp/src/com/model/Registration.java

```
package com.model;

import java.time.LocalDate;

public class Registration {

    private Student studentObj;
    private Course courseObj;
    private LocalDate dateOfRegistration;
    private char grade;


    public Registration(Student studentObj, Course courseObj, LocalDate dateOfRegistration, char grade) {
        this.studentObj = studentObj;
        this.courseObj = courseObj;
        this.dateOfRegistration = dateOfRegistration;
        this.grade = grade;
    }

    public Student getStudentObj() {
        return studentObj;
    }

    public void setStudentObj(Student studentObj) {
        this.studentObj = studentObj;
    }

    public Course getCourseObj() {
        return courseObj;
    }

    public void setCourseObj(Course courseObj) {
        this.courseObj = courseObj;
    }

    public LocalDate getDateOfRegistration() {
        return dateOfRegistration;
    }

    public void setDateOfRegistration(LocalDate dateOfRegistration) {
        this.dateOfRegistration = dateOfRegistration;
    }

    public char getGrade() {
        return grade;
    }

    public void setGrade(char grade) {
        this.grade = grade;
    }

    public void calculateGrade(int mark){

        if(mark >= 90)
        {
            setGrade('O');
```

```java
        }
        else if(mark >= 70)
        {
            setGrade('A');
        }
        else
        {
            setGrade('B');
        }
    }
}

```
ELearningApp/src/com/model/Student.java
```java
package com.model;

public class Student {

    private String studentId;
    private String studentName;
    private String phoneNumber;
    private String emailId;

    public Student(String studentId, String studentName, String phoneNumber, String emailId) {
        this.studentId = studentId;
        this.studentName = studentName;
        this.phoneNumber = phoneNumber;
        this.emailId = emailId;
    }

    public String getStudentId() {
        return studentId;
    }

    public void setStudentId(String studentId) {
        this.studentId = studentId;
    }

    public String getStudentName() {
        return studentName;
    }

    public void setStudentName(String studentName) {
        this.studentName = studentName;
    }

    public String getPhoneNumber() {
        return phoneNumber;
    }

    public void setPhoneNumber(String phoneNumber) {
        this.phoneNumber = phoneNumber;
    }

    public String getEmailId() {
        return emailId;
    }

    public void setEmailId(String emailId) {
            this.emailId = emailId;
    }

}

```
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)