EmployeeDetailsMain.java

```
package com.cts.employeedetailsreport.client;
import com.cts.employeedetailsreport.skeleton.SkeletonValidator;
import com.cts.employeedetailsreport.service.HospitalManagement;
public class EmployeeDetailsMain {
 public static void main(String[] args) {
                        // CODE SKELETON - VALIDATION STARTS
                        // DO NOT CHANGE THIS CODE
                        new SkeletonValidator();
                        // CODE SKELETON - VALIDATION ENDS
   // TYPE YOUR CODE HERE
    new HospitalManagement();
  }
 }
```

==========

DBConnectionManager.java

```
package com.cts.employeedetailsreport.dao;

    import java.io.FileInputStream;
    import java.io.IOException;
    import java.sql.DriverManager;
    import java.sql.Connection;
    import java.util.Properties;


   import com.cts.employeedetailsreport.exception.InvalidEmployeeNumberException;
```

```
public class DBConnectionManager {



    private static Connection con = null;
    private static DBConnectionManager instance;
    public DBConnectionManager()  throws InvalidEmployeeNumberException
    {
        FileInputStream fis=null;
        try{
          fis=new FileInputStream("database.properties");
        Properties props=new Properties();
        props.load(fis);
        Class.forName(props.getProperty("DB_DRIVER_CLASS"));

    con=DriverManager.getConnection(props.getProperty("DB_URL"),props.getProperty("DB_U
SERNAME"),props.getProperty("DB_PASSWORD"));
        }catch(Exception e){
          e.printStackTrace();
        }finally{
          try{
            fis.close();


          }catch(IOException e){
            e.printStackTrace();


          }
        }
```

```
            //FILL THE CODE HERE
        }
        public static DBConnectionManager getInstance() throws InvalidEmployeeNumberException
{
            //FILL THE CODE HERE
            instance=new DBConnectionManager();
            return instance;
        }
        public Connection getConnection(){
           return con;
        }
 }
```

==========

DetailsDAO.java

```
package com.cts.employeedetailsreport.dao;

   import java.sql.Statement;
   import java.sql.Connection;
   import java.sql.SQLException;
   import java.util.List;

   import com.cts.employeedetailsreport.exception.InvalidEmployeeNumberException;
   import com.cts.employeedetailsreport.model.EmployeeDetails;

   public class DetailsDAO {
```

```
        public boolean insertEmployeeList(List <EmployeeDetails> eList) throws
InvalidEmployeeNumberException {


                boolean recordsAdded = false;

                DBConnectionManager db=new DBConnectionManager();

                DBConnectionManager.getInstance();

                Connection conne=db.getConnection();


                // FILL THE CODE HERE


  try{

    Statement st=conne.createStatement();

     for(int i=0;i<eList.size();i++)

     {

        String ins="INSERT INTO EmployeeDetails
VALUES("+eList.get(i).getEmployeeName()+","+eList.get(i).getEmployeeNumber()+","+eList.get(i).get
Level()+","+eList.get(i).getExtraWorkingHours()+","+eList.get(i).getTotalSalary();

        st.executeUpdate(ins);

     }

  conne.commit();

  recordsAdded=true;

  }

  catch(SQLException e){

    e.printStackTrace();


  try{

    conne.rollback();

  }catch(Exception k){

    k.printStackTrace();

  }

        }finally{

           try{
```

```
            conne.close();
          }catch(Exception e){
            e.printStackTrace();
          }
        }
        return recordsAdded;
   }
 }
```

==========

InvalidEmployeeNumberException.java

```
package com.cts.employeedetailsreport.exception;
 public class InvalidEmployeeNumberException extends Exception
   {

        String strMsg1;
        Throwable strMsg2;



        public InvalidEmployeeNumberException() {
                super();
        }
        public InvalidEmployeeNumberException(String strMsg1)
        {
                super(strMsg1);
        }

        public InvalidEmployeeNumberException(String strMsg1, Throwable strMsg2) {
                super();
                this.strMsg1 = strMsg1;
```

```
            this.strMsg2 = strMsg2;
        }
    }
```

==========

EmployeeDetails.java

```
package com.cts.employeedetailsreport.model;

    public class EmployeeDetails {
        private String employeeNumber;
       private String employeeName;
       private String level;
       private int extraWorkingHours;
       private double totalSalary;

   //Constructors

        public EmployeeDetails(String string1, String string2, String string3, int i,double sal) {
            this.employeeNumber=string1;
            this.employeeName=string2;
            this.level=string3;
            this.extraWorkingHours=i;
            this.totalSalary=sal;
        }

        public EmployeeDetails() {
        }

   //getters and setters
```

```
public String getEmployeeNumber() {
        return employeeNumber;
}

public void setEmployeeNumber(String employeeNumber) {
        this.employeeNumber = employeeNumber;
}

public String getEmployeeName() {
        return employeeName;
}

public void setEmployeeName(String employeeName) {
        this.employeeName = employeeName;
}

public String getLevel() {
        return level;
}

public void setLevel(String level) {
        this.level = level;
}

public int getExtraWorkingHours() {
        return extraWorkingHours;
}
public void setExtraWorkingHours(int extraWorkingHours) {
        this.extraWorkingHours = extraWorkingHours;
}
```

```
	public double getTotalSalary() {
		return totalSalary;
	}

	public void setTotalSalary(double totalSalary) {
		this.totalSalary = totalSalary;
	}

	@Override
	public String toString() {
		return "EmployeeDetails [employeeNumber=" + employeeNumber + ", employeeName=" + employeeName + ", level="
				+ level + ", extraWorkingHours=" + extraWorkingHours + ", totalSalary=" + totalSalary + "]";
	}

  }
```

==========

HospitalManagement.java

```
package com.cts.employeedetailsreport.service;

    import java.util.ArrayList;
    import java.util.List;
    import com.cts.employeedetailsreport.exception.InvalidEmployeeNumberException;
    import com.cts.employeedetailsreport.model.EmployeeDetails;
    import com.cts.employeedetailsreport.util.ApplicationUtil;
```

```
public class HospitalManagement {
        private List<String>employeeRecords;
        public List<String> getEmployeeRecords(){
           return employeeRecords;
        }
        public void setEmployeeRecords(List<String>employeeRecords){
           this.employeeRecords=employeeRecords;
        }

public static ArrayList <EmployeeDetails> buildEmployeeList(List <String> employeeRecords) {


                final String COMMADELIMITER = ",";
                ArrayList <EmployeeDetails> empList = new ArrayList<EmployeeDetails>();


        //fill the code here
                int listSize=employeeRecords.size();
                int i=0;
                EmployeeDetails empdet;
                while(listSize-->0){
                   String[]
employeeDetailsString=employeeRecords.get(i++).split(COMMADELIMITER);
                   try{
                      if(ApplicationUtil.validate(employeeDetailsString[0])){
                         int extraHours=Integer.parseInt(employeeDetailsString[3]);
                         double sal=calculateTotalSalary(employeeDetailsString[2],extraHours);
                         empdet =new
EmployeeDetails(employeeDetailsString[0],employeeDetailsString[1],employeeDetailsString[2],extra
Hours,sal);
                         empList.add(empdet);
                      }
```

```
                    }
                    catch(InvalidEmployeeNumberException in){
                       System.out.print(in);
                    }
                       }
                    return empList;
          }


          public boolean addEmployeeList(String inputFeed) throws
InvalidEmployeeNumberException
          {
          //fill the code here
        try{
      this.setEmployeeRecords(ApplicationUtil.readFile(inputFeed));
         return true;

      }
      catch(Exception e){
         e.printStackTrace();
      }
      return false;
          }

          public static double calculateTotalSalary(String level,int extraWorkingHours)
      {
                    double sal=0.0;
          //fill the code here
        if(level.equals("level1")){
            sal=75000+(1000*extraWorkingHours);
         }
```

```
        else if(level.equals("level2")){
            sal=50000+(1000*extraWorkingHours);
        }
        else if(level.equals("level3")){
            sal=35000+(1000*extraWorkingHours);
        }
        else if(level.equals("level4")){
            sal=25000+(1000*extraWorkingHours);
        }
                return sal;
    }

}
```

==========

ApplicationUtil.java

```
package com.cts.employeedetailsreport.util;
  import java.util.ArrayList;
  import java.io.FileInputStream;
  import java.io.FileNotFoundException;
  import java.io.InputStreamReader;
  import java.nio.charset.StandardCharsets;
  import java.util.List;
  import java.io.BufferedReader;
  import com.cts.employeedetailsreport.exception.InvalidEmployeeNumberException;

  public class ApplicationUtil {
```

```
    public static List<String> readFile(String filePath) throws
FileNotFoundException,InvalidEmployeeNumberException

            {

                List<String> employeeList=new ArrayList<String>();


                        // FILL THE CODE HERE

          try(BufferedReader br=new BufferedReader(new InputStreamReader(new
FileInputStream(filePath),StandardCharsets.UTF_8));){

            String line;

            while((line=br.readLine())!=null){

               employeeList.add(line);

            }

          }

          catch(Exception e){


          }

                return employeeList;


            }

            public static boolean validate(String employeeNumber) throws
InvalidEmployeeNumberException

                {

                boolean val=false;

                        // FILL THE CODE HERE

            int n=employeeNumber.length();

            if(n!=7)throw new InvalidEmployeeNumberException("Invalid Employee Number");


            char[] charArray=employeeNumber.toCharArray();


             if(charArray[0]!='P'&&charArray[1]!='R')throw new
InvalidEmployeeNumberException("Invalid Employee Number");;
```

```
        for(int i=2;i<n;i++)
        {
          if(!Character.isDigit(charArray[i]))
          {
           throw new InvalidEmployeeNumberException("Invalid Employee Number");
          }
        }

      val=true;
            return val;
            }

}
```