**Project Assessment**

**GENERAL INSTRUCTIONS:**Please carefully read the below instructions

The objective of this assessment is to check your ability to complete a project as per the provided “Project Design”.

**You are expected to –**

1.      Write the source code for the classes, methods and packages **EXACTLY** as mentioned in the “**Project Design**” section.

2.      Ensure that the names of the packages, classes, methods and variables **EXACTLY MATCH** with the names specified in the “Project Design” section.

3.      Understand the project requirements and ACCORDINGLY WRITE the code and logic in the classes and methods so as to meet all given requirements.

**Creating the project and testing it –**

1.      You are expected to create your project locally using eclipse (or any other IDE) on your desktop.

2.      Once you are ready with the code, you should upload the src folder of your project in .zip format, using the “Upload Zip File” button.

IMPORTANT NOTE 1 : The extension of the zip file should be ONLY .zip (any other zip formats such as .7z  will produce unexpected results)

IMPORTANT NOTE 2 : The .zip file should contain zip of ONLY the src folder structure from your project. (If the zip file has anything other than the src folder structure, the result will be unexpected. Do not zip the entire project folder structure. Just do the zip of the src folder structure and upload it)

IMPORTANT NOTE 3 : The name of the .zip file should be <your employee number>.zip For e.g., if your emp no. is 12345, the zip file should be named 12345.zip.

3.      After uploading the zip file, you can click on “Compile & Test” button and the assessment engine will compile your source code and test it using its pre-defined test-cases.

4.      If some of the test-cases fail, you can make the fixes in your source code locally on your desktop, and again repeat the above two steps.

5.      Once you are finished with all the fixes, you can click on “Final Submission” button, which will show you the final result/score.

**NOTE that –**

6.      The assessment engine will create objects and invoke methods as per the project design, and while doing so, it will use your packages, classes and methods. If your packages, classes and methods have a name mismatch or method prototype mismatch with respect to the expected “Project Design”, the tool will show it as an ERROR. If your packages, classes and methods match as per the names but do not perform the expected functionality, the tool will show it as a FAILURE.

7.      Unless specified in the Project Design, DO NOT use**System.exit(0)** anywhere in your code. Using**System.exit(0)** in your project code will cause the CPC test engine to exit and it will not be able to run all test-cases.

^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^

**Bus Schedule Management System**

**Project Objective:**

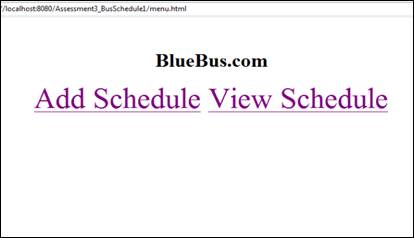
BlueBus.com is an online bus ticket booking system. They wanted to develop an administrator module to add schedules of different buses. This is only a part of their application.

**Note: This application will cover only limited functionalities for the assessment.**

**Sample Output:**

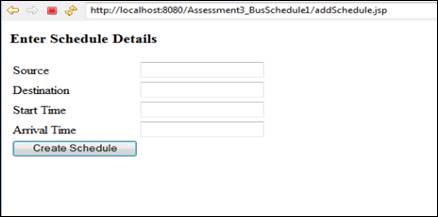
Note: The below mentioned HTML pages that you will create are not going to be tested by the CPC application, so please**don’t spend more time on decorating the UI pages**. The samples are shown for reference purpose only to give you a quick view of what is expected from the Application.

**menu.html**



1.      If “Add Schedule” link is clicked, the following screen is shown

**addSchedule.jsp**



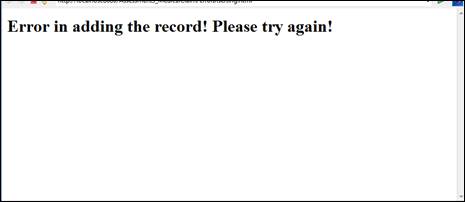
On Successful Insert, the user should be directed to the success.jsp page.

**success.jsp**



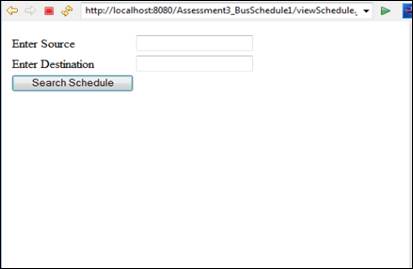
If there is any error while adding the record, the following page should be displayed.

**errorInserting.html**



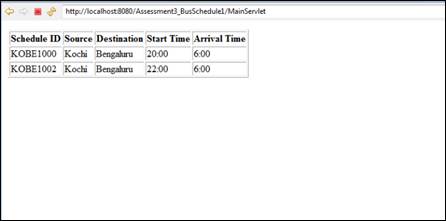
2.      If “View Schedule” link is Clicked on menu.html, the below given page should be displayed

**viewSchedule.jsp**



If matching records exists in the database, display the schedule details in the below given page.

**displaySchedule.jsp**



If no matching records exists in the database, display the following message in displaySchedule.jsp



**Project Design:**

**A. Database Design: You can skip this step if you have created this user earlier**

**1.      Create a new user in database [ To be done in the backend by using sql commands ]**

a)      **Note:**Do **NOT** use the default scott/tiger account of oracle for this project.  You will have to create a new user in the below mentioned format.

b)      Username/password :  B<batchnumber><employeeid>

For example, if your batch number is **39806** and Employee number is **12345**, then the oracle user should be **B3980612345** and the password should be **B3980612345**

c)      For JDBC connection, only use **orcl** as service name and **1521** as port number

**2.      Steps for creating a new user**

a)      Open command prompt

b)      **Type** Sqlplus / as sysdba

c)      **Create user <username> identified by <password>;**     [ For example to create a user named “test” with password “test” : create user test identified by test; ]

d)      **Grant connect, resource to <username>;**  [ E.g: grant connect, resource to test;]

e)      Commit;

f)       Exit;

**3.      Create Table [ To be done using sql commands, after logging-in as the new user that has been created in above step ]**

**Table Name: SCHEDULE\_TBL**

|  |  |  |
| --- | --- | --- |
| **Column** | **Datatype** | **Description** |
| **SCHEDULEID** | VARCHAR2(8 BYTE) | Primary Key |
| **SOURCE** | VARCHAR2(30 BYTE) | Not null |
| **DESTINATION** | VARCHAR2(30 BYTE) | Not null |
| **STARTTIME** | VARCHAR2(5 BYTE) |  |
| **ARRIVALTIME** | VARCHAR2(5 BYTE) |  |

**Sequence Name: SCHEDULE\_SEQ**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sequence Name** | **Minimum Value** | **Max Values** | **Incremental value** | **Start Value** |
| **SCHEDULE\_SEQ** | 1000 | 9999 | 1 | 1000 |

**B. System Design: Server Side**

|  |  |
| --- | --- |
| **Name of the package** | **Usage** |
| com.wipro.bus.util | Contains the class that establishes the database connection |
| com.wipro.bus.bean | Contains all the bean classes |
| com.wipro.bus.dao | Contains the DAO classes that performs the real JDBC operations |
| com.wipro.bus.service | Contains the administrator class that receives input from Servlets and then invokes the respective DAO class methods |
| com.wipro.bus.servlets | Contains the MainServlet that performs the role of a Controller |

**Package: com.wipro.bus.util**

|  |  |  |
| --- | --- | --- |
| **Class** | **Method and Variables** | **Description** |
| **DBUtil** |  | DB connection class |
|  | static Connection**getDBConnection**() | Establish a connection to the database and return the java.sql.Connection reference |

|  |  |  |
| --- | --- | --- |
| **Class** | **Method and Variables** | **Description** |
| **InvalidInputException** |  | Exception Class |
|  | String toString() | Return the error message:  **“Invalid Input”** |
|  |  |  |

**Package: com.wipro.bus.bean**

|  |  |  |
| --- | --- | --- |
| **Class** | **Method and Variable Names** | **Description** |
| **ScheduleBean** |  | **Bean Class** |
| String | **scheduleId** | ID number of the schedule |
| String | **source** | Source of bus schedule |
| String | **destination** | Destination of bus schedule |
| String | **startTime** | Start time of bus |
| String | **arrivalTime** | Arrival time of the bus |
|  | Setters and Getters for all properties | Using Eclipse, create getters and setters for all the properties |

**Package: com.wipro.bus.dao**

|  |  |  |
| --- | --- | --- |
| **Class** | **Method and Variables** | **Description** |
| **ScheduleDAO** |  | DAO class |
|  | **public**String createSchedule(ScheduleBean scheduleBean) |          This method receives a ScheduleBean object and inserts  the values in the database  .       If inserted the method should return "**SUCCESS**" |
|  | **public** String generateID(String source, String destination) |          This method receives two strings- source name and destination name and creates a scheduleId from it.           The Schedule ID is a combination of first two letters of source in uppercase, first two letters of destination in uppercase followed by 4 digit number generated from a sequence in database.           Eg: If source is Chennai and destination is Bengaluru then the scheduleId will be CHBE1234 where 1234 is the number generated by the sequence. |
|  | **public** ArrayList<ScheduleBean> viewSchedule(String source,String destination) |          Should fetch the records matching criteria and return the collection. |

**Package: com.wipro.bus.service**

|  |  |  |
| --- | --- | --- |
| **Class** | **Method and Variables** | **Description** |
| **Administrator** |  |  |
|  | **public** String addSchedule(ScheduleBean scheduleBean) |          This method receives a ScheduleBean object           Refer testcases given at the end of the document and perform the necessary validations.           If same source and destination is provided should give back a message “**Source and Destination Same**”           If validation is successful, then CallgenerateID(String source, String destination) from dao which will return a schedule ID and store the same in the bean.           InvokecreateSchedule(ScheduleBean scheduleBean) from dao and return the string value  For other user defined error messages, please look at the test cases given at the end of the document. |
|  | **public**ArrayList<ScheduleBean> viewSchedule(String source,String destination) |          This method receives the names of source and destination and returns an ArrayList of ScheduleBean with matching records. |

**Package: com.wipro.bus.servlets**

|  |  |  |
| --- | --- | --- |
| **Class** | **Method and Variables** | **Description** |
| **MainServlet** | **Servlet Class** |  |
|  | **public** String addSchedule(HttpServletRequest request) |          This method is invoked by the doPost(HttpServletRequest request, HttpServletResponse response) method of the Servlet Class if the Client has clicked on ‘Add Schedule’ link in the menu page           This method will create a ScheduleBean Object and will initialize all its properties with the values that are available in the request object           It will invoke the Administrator class’s**addSchedule(ScheduleBean scheduleBean)** and will return the String value received from String**addSchedule(ScheduleBean scheduleBean)** method |
|  | **public** ArrayList<ScheduleBean> viewSchedule(HttpServletRequest request) |          This method is invoked by the doPost(HttpServletRequest request, HttpServletResponse response) method of the Servlet Class if the Client has clicked on ‘View Schedule’ link in the menu page           This method will invoke the  viewSchedule(String source,String destination)method of the Administrator Class and it should return the ArrayList of ScheduleBean object received |
|  | **protected** **void**doPost(HttpServletRequest request, HttpServletResponse response) **throws**ServletException, IOException |          This method processes the Client’s request           It accesses a request parameter called ‘operation’ and based on the value of that parameter, it’ll decide whether to access the**addSchedule(HttpServletRequest request)** or**viewSchedule(HttpServletRequest request)** method           If the operation parameter value is ‘newSchedule’ then the following things needs to be done  1.      the**addSchedule(HttpServletRequest request)** method should be called  2.      If return value equals**“SUCCESS”** the servlet should redirect to **“success.jsp”**  3.      If return value equals **“FAIL”** or exception message then the servlet should redirect to**"errorInserting.html"**           If the operation parameter value is ‘viewSchedule’ then the following things needs to be done  1.     **viewSchedule(HttpServletRequest request)** method will be invoked  **2.**If the return value is**null** the servlet should redirect to**“displaySchedule.jsp”**displaying a message-          **‘No matching schedules exists! Please try again!’**  3.      Else the ArrayList of beans  should be set as an attribute to the request object and should forward the control to**“displaySchedule.jsp”** |

**B. System Design: UI Pages**

|  |  |  |
| --- | --- | --- |
| 1. menu.html | http://10.201.67.120:9090/pbl/cpc/Ass3Bus_files/image008.jpg |          Contains two links Add Schedule and View Schedule           Add Schedule link will take you to addSchedule.jsp          View Claim link will take you to viewSchedule.jsp |
| 2. addSchedule.jsp | http://10.201.67.120:9090/pbl/cpc/Ass3Bus_files/image009.jpg |         In the HTML form include this tag  <input type="hidden" name="operation" value="newSchedule">.This will help the MainServlet to understand that the add operation need to be done          When the submit button is clicked it invokes the post method of the MainServlet |
| 3.errorInserting.html | http://10.201.67.120:9090/pbl/cpc/Ass3Bus_files/image010.jpg |          If there is any error during record insertion, should display the  Following message    **“Error in adding the record! Please try again!”** |
| 4.success.jsp | http://10.201.67.120:9090/pbl/cpc/Ass3Bus_files/image011.jpg |          If the record is inserted then display the message- **‘Added successfully’** |
| 5. viewSchedule.jsp | http://10.201.67.120:9090/pbl/cpc/Ass3Bus_files/image012.jpg |          When the Search button is clicked it invokes the post method of the MainServlet           In the HTML form include this tag  <input type="hidden" name="operation" value="viewSchedule">.This will help the MainServlet to understand that the view operation need to be done. |
| 6. displaySchedule.jsp | http://10.201.67.120:9090/pbl/cpc/Ass3Bus_files/image013.jpg    http://10.201.67.120:9090/pbl/cpc/Ass3Bus_files/image014.jpg |          If the combination of source and destination exists, display the matching schedules.           Otherwise display**- ‘No matching schedules exists! Please try again!’** |

**Example Test Cases:**

|  |  |  |
| --- | --- | --- |
| **Testcase No** | **Testcase** | **Expected Result** |
| **1** | **Passing bean object with valid values to**addSchedule(ScheduleBean scheduleBean)**method of Administrator class** | **SUCCESS** |
| **2** | **Passing null bean object to**addSchedule(ScheduleBean scheduleBean)**method of Administrator class** | **The user define exception InvalidInputException is thrown and ‘INVALID INPUT’ is returned** |
| **3** | **Passing bean object with empty source or destination to**addSchedule(ScheduleBean scheduleBean) **method of Administrator class** | **The user define exception InvalidInputException is thrown and ‘INVALID INPUT’ is returned** |
| **4** | **Passing bean object with source or destination which has less than 2 letters  to**addSchedule(ScheduleBean scheduleBean)**method of Administrator class** | **The user define exception InvalidInputException is thrown and ‘INVALID INPUT’ is returned** |
| **5** | **Passing valid source and destination to**viewSchedule(String source,String destination)**method of Administrator class** | **ArrayList of ScheduleBean objects which is not null** |
| **6** | **Passing invalid source and destination to**viewSchedule(String source,String destination)**method of Administrator class** | **Null** |
| **7** | **Checking whether the servlet methods are implemented** | **IMPLEMENTED** |