## 

## Software Testing Project Report

## An Overview of the Project

**Introduction**

The web application under testing is Java Web App for Employee Time Reporting.

The source code of the app can be found at following link:

<http://www.java2s.com/Open-Source/Java_Free_Code/Web_Application/Download_timesheet_Free_Java_Code.htm>

**Overview of the application being tested**

The project is a lighter version of a pay system for managing the time reporting of the employees.

Some of the application features are:

* Adding the new employees in the database and managing their roles.
* Time Logging for non-salaried employee (either by employee himself or by the admin)
* Configuring the database settings.
* Managing the groups in the company.
* Generate the ADP reports of the employees.

**Environment Setup**

The environment to run the application can be created by following these steps:

1. Download maven from here: <https://maven.apache.org/download.cgi>
2. Download and install the mysql workbench from here: <https://dev.mysql.com/downloads/installer/>
3. Download jdk1.8+
4. Clone <https://github.com/risaldar/PaySystem>
5. In the .\timesheet-master\build.bat, set the JAVA\_HOME to jdk path and similarly set MAVEN\_HOME to the maven path.
6. In the .\timesheet-master\run.bat, set the JAVA\_HOME and set CATALINA\_HOME to absolute path appended by ".\PaySystem\apache-tomcat-7.0.108-windows-x64\apache-tomcat-7.0.108".
7. Open Command prompt, navigate to project repository i.e .\Paysystem\timesheet-master\ and execute build.bat.
8. This will build the project.
9. Follow these steps for MySQL server

|  |
| --- |
| * Open ‘Run’ Window by using Win key + R * Type ‘services.msc’ * Now search for MySQL service based on the version that is installed. * Click on ‘stop’, ‘start’ or ‘restart’ the service option. |

1. Open mysql workbench, start a new local connection, provide root user password (which is set at time of MySQL installation) and enter following two queries:

DROP DATABASE paysystem;

Create DATABASE paysystem;

1. When the database is created for first time, only execute the create query.
2. Execute run.bat.

**Application Setup**

After the local server is running, go to <http://localhost:8090/> or you can just go to the application <http://localhost:8090/PaySystem>



Enter the company name, and then click next.

Then you will be redirected to add information about the database.  
To avoid confusion, database username and database password are kept same.



You will be redirected to add username and password for the user purpose. These are also kept same.



You will be redirected to the login page.



After clicking login, Login using the username you set earlier.



After login you will be directed to the dashboard. Below is the full dashboard.



**Testing Team**

Danish Hassan MSCS-20001

Muhammad Abu Bakar MSCS-20013

Muhammad Awais MSCS-20074

Musa Khan MSCS-20065

**References: List of documents, websites any other material to be referred.**

We have cloned the source code in our repository. <https://github.com/risaldar/PaySystem>

All the reference material can be found in the repository.

## List of Application Features to be tested

Following are the application features to be tested.

1. Wage can only be double (float) values.
2. Calculate hours worked for non-salaried and non-admin person.
3. Admin can log time for other non-salaried person.
4. Approval of timesheet by timesheet approver.
5. Non-salaried person can log his own working time and send for approval.
6. A non-salaried and non-regular cannot approve his own time-sheet.
7. Only paid hour type shall appear in ADP report.
8. Employs added in ADP report shall have all combinations of employee properties. (Salaried, active, role, group).

## List of Testing Techniques Used

Following Black box testing techniques will be applied on the above mentioned use cases:

1. Equivalence Class
2. Boundary Value Analysis
3. Decision Table Testing
4. Domain Analysis
5. Pair-Wise Testing

## Test Environment

Application execution environment

|  |  |
| --- | --- |
| Operating system | Windows |
| Application servers | apache-tomcat-7.0.108-windows-x64  MySQL Server |
| Tools | Maven  JDK1.8+  MySQL Workbench |

## Test Cases (Blackbox Testing)

### **Use Case # 1: Wage can only be double (float) values.**

We shall use Equivalence class and boundary value analysis technique to test this feature. Since no description is provided to user on web page, it is assumed that valid input to this function shall be any non-negative real value. So we define two class in this case,

Equivalence Class Partitioning:

|  |  |  |
| --- | --- | --- |
| **Class #** | **Class Type: Valid Class (VC) / Invalid Class (IC)** | **Description** |
| C1 | IC | All negative real numbers i.e. Wage < 0.0 |
| C2 | VC | All non-negative real numbers i.e. Wage >= 0.0 |

Boundary Value Analysis. (Assume smallest part to be 1/100 of fractional part)

|  |  |  |  |
| --- | --- | --- | --- |
| **Boundary #** | **Boundary Types: Valid Boundary (VB) / Invalid Boundary (IB)** | **Description** | **Class Reference** |
| B1 | IB | Wage = -0.01 | C1 |
| B2 | VB | Wage = 0.00 | C1 |
| B3 | VB | Wage = 0.01 | C1 |
| B4 | IB | Wage = -0.01 | C2 |
| B5 | VB | Wage = 0.00 | C2 |
| B6 | VB | Wage = 0.01 | C2 |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Scenario ID** | | Manage Account | | | | **Test Case ID** | 001 | |
| **Test Case Objective** | | Test current user’s wage entry functionality in account management. | | | | **Test Priority** | High | |
| **Test browser** | | Chrome | | | |  |  | |
| **Pre-condition** | | Registered user should be logged in to the Pay system web portal and should be at his dashboard view page. | | | | **Post-condition** | NA | |
|  | | | | | | | | | | |
| **Step No** | **Action** | | **Inputs** | | **Expected Output** | **Actual Output** | **Test Result** | **Test Comments** | |
| 1 | Go to ‘Manage Account’ section | | N/A | | http://localhost:8090/PaySystem/manageUser.jsp |  |  |  | |
| 2 | Enter Wage and press Submit button | | Wage: 0.0 | | Wage should be stored in database and user should go back to his dashboard <http://localhost:8090/PaySystem/dashboard.jsp> |  |  |  | |
| 3 | Go to ‘Manage Account’ section | | N/A | | Wage should be same as previously stored in database |  |  |  | |
| **Overall Result** | | | | *Passed Failed Not Executed* | | | | | | |

### **Use Case # 2: Calculate hours worked for non-salaried and non-admin person.**

We shall use Equivalence class and boundary value analysis technique to test this feature.

On the user page there is an option of PM check box, precondition is that the both boxes shall be unchecked for time started and for time ended.

Time ended and duration of lunch break shall have fixed values of 17:00 and 1:00 respectively.

Equivalence Class Partitioning:

|  |  |  |
| --- | --- | --- |
| **Class #** | **Class Type: Valid Class (VC) / Invalid Class (IC)** | **Description** |
| C1 | VC | Time started between 0:00 – 24:00 |
| C2 | IC | Time started <= -1:00 |
| C3 | IC | Time started >= 25:00 |
| C4 | IC | When time started is a string |

Boundary Value Analysis. (Assume smallest increment to be 1 hour)

|  |  |  |  |
| --- | --- | --- | --- |
| **Boundary #** | **Boundary Types: Valid Boundary (VB) / Invalid Boundary (IB)** | **Description** | **Class Reference** |
| B1 | VB | Time started = 0:00 | C1 |
| B2 | VB | Time started = 24:00 | C1 |
| B3 | IB | Time started = -1:00 | C2 |
| B4 | IB | Time started = 25:00 | C3 |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Scenario ID** | | Enter Time Started | | | | **Test Case ID** | 002 | |
| **Test Case Objective** | | Test the hours calculation utility in time entry | | | | **Test Priority** | High | |
| **Test browser** | | Chrome | | | |  |  | |
| **Pre-condition** | | Non – Salaried user should be logged in to the Pay system web portal and should be at his dashboard view page.  <http://localhost:8090/PaySystem/dashboard.jsp>  Go to ‘Enter Time’ section | | | | **Post-condition** | NA | |
|  | | | | | | | | | | |
| **Step No** | **Action** | | **Inputs** | | **Expected Output** | **Actual Output** | **Test Result** | **Test Comments** | |
| 1 | Enter Time Started  Time ended =17:00  Duration of Lunch =1:00  Click calculate | | Time started: 12:00 | | The total hours in between those time started and time ended subtracted by lunch break duration shall appear in “Hours Worked” section on the same page. | The total hours in between those time started and time ended subtracted by lunch break duration are appeared in “Hours Worked” section on the same page. | Pass | N/A | |
| **Overall Result** | | | | *Passed Failed Not Executed* | | | | | | |

Test Case Variations

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case #** | **Inputs** | **Expected Output** | **Actual Output** | **Test Result** | **Test Comments (Class/Boundary reference)** |
| 1 | Time Started = 12:00  Time ended =17:00  Duration of Lunch =1:00 | 4 | 4 | Pass | C1 (point within the class) |
| 2 | Time Started = -12:00 Time ended =17:00  Duration of Lunch =1:00 | Error : Input is incorrect.  Started time cannot be negative. | 28 | Fail | C2 (point within the class) |
| 3 | Time Started = 30:00 Time ended =17:00  Duration of Lunch =1:00 | Error :  Input is incorrect.  Started time cannot be greater than 24:00. | -14 | Fail | C3 (point within the class) |
| 4 | Time Started = “abc” Time ended =17:00  Duration of Lunch =1:00 | Error :  Input is incorrect.  Started time cannot be a string | Error: Input is incorrect | Pass | C4 (point within the class) |
| 5 | Time Started = 0:00 Time ended =17:00  Duration of Lunch =1:00 | 16 | 16 | Pass | C1/B1 (point on the boundary)  C2/B3 (point above the boundary) |
| 6 | Time Started = 1:00 Time ended =17:00  Duration of Lunch =1:00 | 15 | 15 | Pass | C1/B1 (point above the boundary) |
| 7 | Time Started = -1:00 Time ended =17:00  Duration of Lunch =1:00 | Error : Input is incorrect.  Started time cannot be negative. | 17 | Fail | C1/B1 (point below the boundary)  C2/B3 (point on the boundary) |
| 8 | Time Started = 24:00 Time ended =17:00  Duration of Lunch =1:00 | Error : Input is incorrect.  Started time cannot be greater than time ended. | -8 | Fail | C1/B2 (point on the boundary)  C3/B4 (point below the boundary) |
| 9 | Time Started = 25:00 Time ended =17:00  Duration of Lunch =1:00 | Error : Input is incorrect.  Started time cannot be greater than time ended. | -9 | Fail | C1/B2 (point above the boundary)  C3/B4 (point on the boundary) |
| 10 | Time Started = 23:00 Time ended =17:00  Duration of Lunch =1:00 | Error : Input is incorrect.  Started time cannot be greater than time ended. | -7 | Fail | C1/B2 (point below the boundary) |
| 11 | Time Started = -2:00 Time ended =17:00  Duration of Lunch =1:00 | Error : Input is incorrect.  Started time cannot be negative. | 18 | Fail | C2/B3 (point below the boundary) |
| 12 | Time Started = 26:00 Time ended =17:00  Duration of Lunch =1:00 | Error : Input is incorrect.  Started time cannot be greater than time ended. | -10 | Fail | C3/B4 (point above the boundary) |

### **Test Case # 3: Admin can log time for other non-salaried person.**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Scenario ID** | | Manage Time | | | | **Test Case ID** | 003 | |
| **Test Case Objective** | | Test the time entry functionality for user with administrator role while logging time for other users. | | | | **Test Priority** | High | |
| **Test browser** | | Chrome | | | |  |  | |
| **Pre-condition** | | Registered user should be logged in to the Pay system web portal and should be at his dashboard view page. | | | | **Post-condition** | NA | |
|  | | | | | | | | | | |
| **Step No** | **Action** | | **Inputs** | | **Expected Output** | **Actual Output** | **Test Result** | **Test Comments** | |
| 1 | Launch the application | | https://www.gmail.com | | Gmail home page | Gmail home page | Pass |  | |
| 2 | Enter correct Email & Password and press Sign in button | | Email id : test@xyz.com  Password: \*\*\*\*\*\* | | Page with latest emails is displayed | Page with latest emails is displayed | Pass |  | |
| 3 |  | |  | |  |  |  |  | |
| **Overall Result** | | | | *Passed Failed Not Executed* | | | | | | |

### **Test Case # 4: Approval of timesheet by timesheet approver.**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Scenario ID** | | Manage Time | | | | **Test Case ID** | 004 | |
| **Test Case Objective** | | Test the logged time entry approval functionality for user with role time sheet approver for time entries logged by other users who are non-salaried. | | | | **Test Priority** | High | |
| **Test browser** | | Chrome | | | |  |  | |
| **Pre-condition** | | Registered user should be logged in to the Pay system web portal and should be at his dashboard view page. | | | | **Post-condition** | NA | |
|  | | | | | | | | | | |
| **Step No** | **Action** | | **Inputs** | | **Expected Output** | **Actual Output** | **Test Result** | **Test Comments** | |
| 1 | Launch the application | | https://www.gmail.com | | Gmail home page | Gmail home page | Pass |  | |
| 2 | Enter correct Email & Password and press Sign in button | | Email id : test@xyz.com  Password: \*\*\*\*\*\* | | Page with latest emails is displayed | Page with latest emails is displayed | Pass |  | |
| 3 |  | |  | |  |  |  |  | |
| **Overall Result** | | | | *Passed Failed Not Executed* | | | | | | |

### **Test Case # 5: Non-salaried person can log his own working time and send for approval.**

We shall use Decision Table for testing this function. It is not a complex condition but systematic approach to identification of test cases is still applicable.

Decision Table

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | **Rule-1** | **Rule-2** |
| **Condition #** | **Input Conditions** |  |  |
| C1 | user is non-salaried | YES | NO |
| **Action #** | **Output Actions** |  |  |
| A1 | User can log time entry | YES | NO |
| A2 | User entry shall be accessible for approval to time sheet approver | YES | NO |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Scenario ID** | | Enter Time | | | | **Test Case ID** | 005 | |
| **Test Case Objective** | | Test the personal time entry functionality for non-salaried user. | | | | **Test Priority** | High | |
| **Test browser** | | Chrome | | | |  |  | |
| **Pre-condition** | | Registered user should be logged in to the Pay system web portal and should be at his dashboard view page. | | | | **Post-condition** | NA | |
|  | | | | | | | | | | |
| **Step No** | **Action** | | **Inputs** | | **Expected Output** | **Actual Output** | **Test Result** | **Test Comments** | |
| 1 | Go to dashboard and find section ‘Enter Time’ | | N/A | | Section ‘Enter Time’ is available on user dashboard |  |  |  | |
| 2 | Go to section ‘Enter Time’ | | N/A | | http://localhost:8090/PaySystem/timeEntering.jsp |  |  |  | |
| 3 | Select Date, Type and enter Hours worked. Press ’Submit Hours’ button. | | Date: Current Date  Type: Office Hours  Hours Worked: 8 | | A pop-up comes up saying ‘Successfully submitted the hours.’ |  |  |  | |
| 4 | Press ‘OK’ on pop-up | | N/A | | User is taken back to Enter time section. |  |  |  | |
| 5 | Press ‘Logout’ | | N/A | | http://localhost:8090/PaySystem/logout.jsp |  |  |  | |
| 6 | Login as employee with rights to approve the filled in time entry. | | User Name: admin  Password: admin | | http://localhost:8090/PaySystem/index.jsp |  |  |  | |
| 7 | Go to Manage Time section | | N/A | | http://localhost:8090/PaySystem/manageTime.jsp |  |  |  | |
| 8 | Select ‘Employee’ as previously logged in user and approve his time entry | | Employee: developer\_1 | | http://localhost:8090/PaySystem/manageTime.jsp |  |  |  | |
| **Overall Result** | | | | *Passed Failed Not Executed* | | | | | | |

### **Test Case # 6: A non-salaried and non-regular cannot approve his own time-sheet.**

We shall use Decision Table for testing this function. It is not a complex condition but systematic approach to identification of test cases is still applicable. In current use case, we shall only consider Role of ‘Time Sheet Approver’ as non-‘Regular’.

Decision Table

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  | **Rule-1** | **Rule-2** | **Rule-3** | **Rule-4** |
| **Condition #** | **Input Conditions** |  |  |  |  |
| C1 | user role == ‘Time Sheet Approver’ | YES | YES | NO | NO |
| C2 | user is non-salaried | YES | NO | YES | NO |
| **Action #** | **Output Actions** |  |  |  |  |
| A1 | User can log time entry | YES | NO | NO | NO |
| A2 | User entry shall be accessible for approval to time sheet approver | NO | NO | NO | NO |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Scenario ID** | | Enter Time / Manage Time | | | | **Test Case ID** | 006 | |
| **Test Case Objective** | | Test the time entry approval functionality against self-approval for non-salaried and non-regular employees. | | | | **Test Priority** | High | |
| **Test browser** | | Chrome | | | |  |  | |
| **Pre-condition** | | Registered user should be logged in to the Pay system web portal and should be at his dashboard view page. | | | | **Post-condition** | NA | |
|  | | | | | | | | | | |
| **Step No** | **Action** | | **Inputs** | | **Expected Output** | **Actual Output** | **Test Result** | **Test Comments** | |
| 1 | Go to dashboard and find section ‘Enter Time’ | | N/A | | Section ‘Enter Time’ is available on user dashboard |  |  |  | |
| 2 | Go to section ‘Enter Time’ | | N/A | | http://localhost:8090/PaySystem/timeEntering.jsp |  |  |  | |
| 3 | Select Date, Type and enter Hours worked. Press ’Submit Hours’ button. | | Date: Current Date  Type: Office Hours  Hours Worked: 8 | | A pop-up comes up saying ‘Successfully submitted the hours.’ |  |  |  | |
| 4 | Press ‘OK’ on pop-up | | N/A | | User is taken back to Enter time section. |  |  |  | |
| 5 | Press ‘Dashboard’ | | N/A | | http://localhost:8090/PaySystem/dashboard.jsp |  |  |  | |
| 7 | Go to Manage Time section | | N/A | | http://localhost:8090/PaySystem/manageTime.jsp |  |  |  | |
| 8 | Select ‘Employee’ as self and approve time entry | | Employee: time\_sheet\_approver\_1 | | ‘Employee’ drop down menu shall not have option to select current user. |  |  |  | |
| **Overall Result** | | | | *Passed Failed Not Executed* | | | | | | |

### **Test Case # 7: Only paid hour type shall appear in ADP report.**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Scenario ID** | | Report | | | | **Test Case ID** | 007 | |
| **Test Case Objective** | | Test the ADP report functionality which should only include logged time entries which include paid hours. | | | | **Test Priority** | High | |
| **Test browser** | | Chrome | | | |  |  | |
| **Pre-condition** | | Registered user should be logged in to the Pay system web portal and should be at his dashboard view page. | | | | **Post-condition** | NA | |
|  | | | | | | | | | | |
| **Step No** | **Action** | | **Inputs** | | **Expected Output** | **Actual Output** | **Test Result** | **Test Comments** | |
| 1 | Launch the application | | https://www.gmail.com | | Gmail home page | Gmail home page | Pass |  | |
| 2 | Enter correct Email & Password and press Sign in button | | Email id : test@xyz.com  Password: \*\*\*\*\*\* | | Page with latest emails is displayed | Page with latest emails is displayed | Pass |  | |
| 3 |  | |  | |  |  |  |  | |
| **Overall Result** | | | | *Passed Failed Not Executed* | | | | | | |

### **Test Case # 8: Employs added in ADP report shall have all combinations of employee properties.**

We shall apply Pair-Wise Testing technique here since the aim is to verify that all combinations of employee properties are included in ADP report. For sake of simplicity, we can only consider following 4 variables in employee attributes. Some combinations of attributes are not logical but are kept in place for sake of testing e.g. Employee with ‘Administrator’ role should not belong to any group other than ‘admin’.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Variable Names** | Group | Role | Active | Salaried |
| **Values (Choices) Count** | 3 | 6 | 2 | 2 |

Select Orthogonal Array: **L18 (3661)** since we have 3 variables which can be covered by array variables of 3 options and 1 variable with 6 options.

Map the Problem to Orthogonal Array:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Variable Names** | **Choices** | | |  |  |  |
| Group | [0] admin | [1] developer | [2] management |  |  |  |
| Role | [0] Administrator | [1] Executive | [2] Manager | [3] Assistant Manager | [4] Time Sheet Approver | [5] Regular Employee |
| Active | [0] active | [1] inactive | [2] active (repeated) |  |  |  |
| Salaried | [0] salaried | [1] non-salaried | [2] salaried (repeated) |  |  |  |

Test Inputs:

|  |  |  |  |
| --- | --- | --- | --- |
| **Group** | **Salaried** | **Active** | **Role** |
| admin | salaried | active | Administrator |
| developer | salaried | active | Executive |
| management | salaried | inactive | Manager |
| admin | non-salaried | active | Assistant Manager |
| management | non-salaried | active | Time Sheet Approver |
| developer | salaried | inactive | Regular Employee |
| management | non-salaried | active | Regular Employee |
| admin | salaried | inactive | Time Sheet Approver |
| developer | non-salaried | inactive | Administrator |
| management | salaried | active | Executive |
| developer | salaried | active | Assistant Manager |
| admin | salaried | active | Manager |
| management | salaried | inactive | Assistant Manager |
| developer | non-salaried | active | Manager |
| developer | salaried | active | Time Sheet Approver |
| admin | salaried | active | Regular Employee |
| admin | non-salaried | inactive | Executive |
| management | salaried | active | Administrator |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Scenario ID** | | Enter | | | | **Test Case ID** | 008 | |
| **Test Case Objective** | | Employs added in ADP report shall have all combinations of employee properties. (Salaried, active, role, group). | | | | **Test Priority** | High | |
| **Test browser** | | Chrome | | | |  |  | |
| **Pre-condition** | | All employees with properties in table of test input given above should be already created in pay system database.  Registered user should be logged in to the Pay system web portal and should be at his dashboard view page. | | | | **Post-condition** | NA | |
|  | | | | | | | | | | |
| **Step No** | **Action** | | **Inputs** | | **Expected Output** | **Actual Output** | **Test Result** | **Test Comments** | |
| 1 | Go to ‘Reports’ section | | N/A | | http://localhost:8090/PaySystem/reports.jsp |  |  |  | |
| 2 | Enter Batch ID and description, click next | | Batch ID: 1  Batch Description: Test Report | | http://localhost:8090/PaySystem/reports.jsp |  |  |  | |
| 3 | Click Finalize Data | | N/A | | ‘adpImport.csv’ file should be downloaded which lists all registered employees. |  |  |  | |
| **Overall Result** | | | | *Passed Failed* *Not Executed* | | | | | | |

## Test Cases (Whitebox Testing)

## List of Test Cases that you created because you think they are important; otherwise none of the formal techniques required you to create them

## Summary

Overall statistics, opinion about the quality of the system, number of use cases, test cases completed etc.

## Role/Responsibilities of each team member.