## 

## 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. Open mysql workbench 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. Date hired should be less than or equal to full time date and time logging should be greater than or equal to date hired.

## 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)

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Scenario ID** | | Login | | | | **Test Case ID** | 001 | |
| **Test Case Objective** | | Test the login functionality | | | | **Test Priority** | High | |
| **Test browser** | | Chrome | | | |  |  | |
| **Pre-condition** | | Registered user | | | | **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 Variations

<<test the above use case with different inputs generated using a variety of black box testing techniques >>

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Step No | 1 | 2 | 3 | 4 |
| Inputs | https://www.gmail.com | Email id : **abc@xyz.com**  Password: \*\*\*\*\*\* |  |  |
| Expected Output | Gmail home page | Page with latest emails is displayed |  |  |
|  |  |  |  |  |
| Inputs | https://www.gmail.com | Email id : **123@xyz.com**  Password: \*\*\*\*\*\* |  |  |
| Expected Output | Gmail home page | **The system prints error, “invalid email address”** |  |  |

## 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.