**PAYROLL MANAGEMENT SYSTEM (PMS)**

**SPECIFICATION-BASED TEST DOCUMENT**

**Prepared By Team 1:**

Ciana Rogers

Juan Diaz

Ivan Rojas

Dwayne Thomas

Daykel Muro

**Prepared For:**

Professor Peter Clarke

Fundamentals of Software Testing, CEN4072 - U01C

Florida International University, Summer 2018

July 14, 2018

**Abstract**

The purpose of this paper is intended to describe a specification-based testing approach applied to the software Payroll Management Software (PMS) - which will serve as validation of its intended usage. Intended usage is determined from the use cases provided by the development team and verified to have been implemented by the testing team.

The tools used to perform all this testing cases were: Eclipse IDE, JUnit, Mockito, PowerMock, Cobertura, eCobertura, EclEmma, CodeCover, and Rational Functional Tester. For each of these categories the document goes over the criteria and procedures for testing, the test cases, and the results of running the tests. All tools, software and terminology used, as well as, all work and preparation done is documented.

## Chapter 1: Introduction

This project is designed around the Payroll Management Software which will be referenced as PMS. This chapter will show an overview of different testing aspects such as: requirements of the system, the overall testing approach, terminology used, and document organization.

### 1.1 Overview of The System

Payroll is a functional requirement for every company or organization - the more employees are in a company’s payroll, the higher the complexity of paying said employees . Based upon some details like basic pay and taxes, salary calculation will be computed. Taking these client inputs, the system creates pay charges, payslips, and all timetables for debit and credit installments. The Payroll Management Software shall provide all these functionalities.

### 1.2 Requirements of the System

Functional requirements are presented in section 1.2.1, while non-functional requirements are presented in section 1.2.2. Subsequent sections present overall testing approach, terminology, and document organization of chapters 2 through 7.

#### 1.2.1 Functional Requirements

|  |  |
| --- | --- |
| PMS\_01\_Login | The system shall display the homepage of the Payroll management system. |
| PMS\_02\_DeleteEmp | The system shall allow the employer to delete the employee. |
| PMS\_03\_SearchEmpTS | The system shall employer to search the Employee’s timesheet information. |
| PMS\_04\_ModifyTS | The system shall modify the employee’s timesheet information. |
| PMS\_05\_ApproveTS | The system shall display the homepage of the Payroll management system. |
| PMS\_06\_Tax | The system shall calculate the tax amount that needs to be withheld from the modeler’s paycheck. |
| PMS\_07\_CalcSal | The system shall display the calculated salary of the respective employee according to the timesheet per week schedule. |
| PMS\_08\_SaveTS | The system shall allow to save the Timesheet. |
| PMS\_ 09\_PayCheck | The system shall generate a downloadable PDF file to show the paycheck of an employee. |
| PMS\_ 10\_DirectDeposit | The system shall allow the employer to deposit the salary in to the employee bank account. |
| PMS\_11\_AddDept | The system shall allow the employer to add a new department. |
| PMS\_12\_RemoveDept | The system shall allow the employer to remove the department. |
| PMS\_13\_AddEmployee | The system shall allow the employer to add a new employee. |
| PMS\_15\_RemoveEmployee | The system shall the employer to remove an employee from a department. |
| PMS\_16\_UpdateEmployee | The system shall allow the employee to change/update its information. |
| PMS\_17\_WorkProfile | The system shall allow to see the work profile of an employee. |
| PMS\_18\_SubmitTimesheet | The system shall allow the employee to submit the timesheet. |
| PMS\_19\_ForgetPasswor | The system shall generate password automatically when an employee is newly registered. |
| PMS\_20\_changepwd | The system shall allow the employee to change the password. |
| PMS\_21\_logout | The system shall allow the employee/employer to logout of the system and directs it to homepage. |
| PMS\_001\_Injector | The system shall allow the employee/employer to logout of the system and directs it to homepage. |
| PMS\_002\_PwdNotificator | The system shall notify the employee/employer through an email when the password is changed. |
| PMS\_003\_Multi.login | The system shall deny simultaneous logins. |
| PMS\_004\_duplicate | The system shall deny the duplicate submission of timesheets. |
| PMS\_005\_security | The system shall ask security question, when an employee tries to retrace his/her password. |
| PMS\_006\_Timeout | The system shall log out automatically if there is no certain action for a specific period. |
| PMS\_007\_textToLabel | The system shall change the input to label when an employee submits the timesheet. |

#### 1.2.2 Non-Functional Requirements

These requirements are composed of reliability, usability, security, maintainability, scalability, and performance. These attributes pose the constraints on the Payroll Management System. These have the same importance of functional requirements as they ensure the efficiency and performance.

**Usability:** No basic training is required for an individual to use this application.

**Reliability:** Mean Time to Failure – for most use cases one failure per week of operation is within acceptable parameters. Average Downtime – for most use cases one hour per month of operation is within acceptable parameters. The PMS\_05\_ApproveTS, PMS\_17\_WorkProfile have possibility of failure when parameters are not accepted.

**Performance:** Most use cases have requests that return within average of 8 seconds. The PMS\_05\_ApproveTS, PMS\_06\_Tax, PMS\_03\_SearchEmpTS and PMS\_10\_DirectDeposit respond within 10 seconds and PMS\_006\_Timeout occurs with 2 minutes.

**Supportability:** The system shall support all modern browser implementations including mobile. The system is constructed purely in Java 8 and thus will be portable provided an appropriate environment. The system is guaranteed to run on Google Chrome, Mozilla Firefox (desktop).

**Implementation:** Client is implemented with HTML. Customer mandates the implementation to support mobile devices and that the server be constructed using Java and the Eclipse IDE

### 1.3 Overall Testing Approach

The program under test is validated using specification-based testing. The software architecture is MVC with Java class facades for the model and controller portions of the architecture. The view is represented by the client web browser.

Test set selection is done through equivalence partitioning, boundary analysis, and ad-hoc input selection. Furthermore, test cases fall into one of three categories: unit testing, subsystem testing, and system testing. For unit and subsystem testing, testing tools like JUnit, Mockito are leveraged. For system testing, the IBM Rational Functional tester application is employed. In addition, test cases include 3 sunny case and 3 rainy day instances for each feature tested in all the categories.

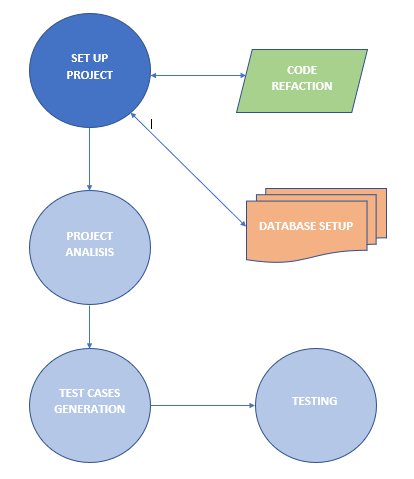
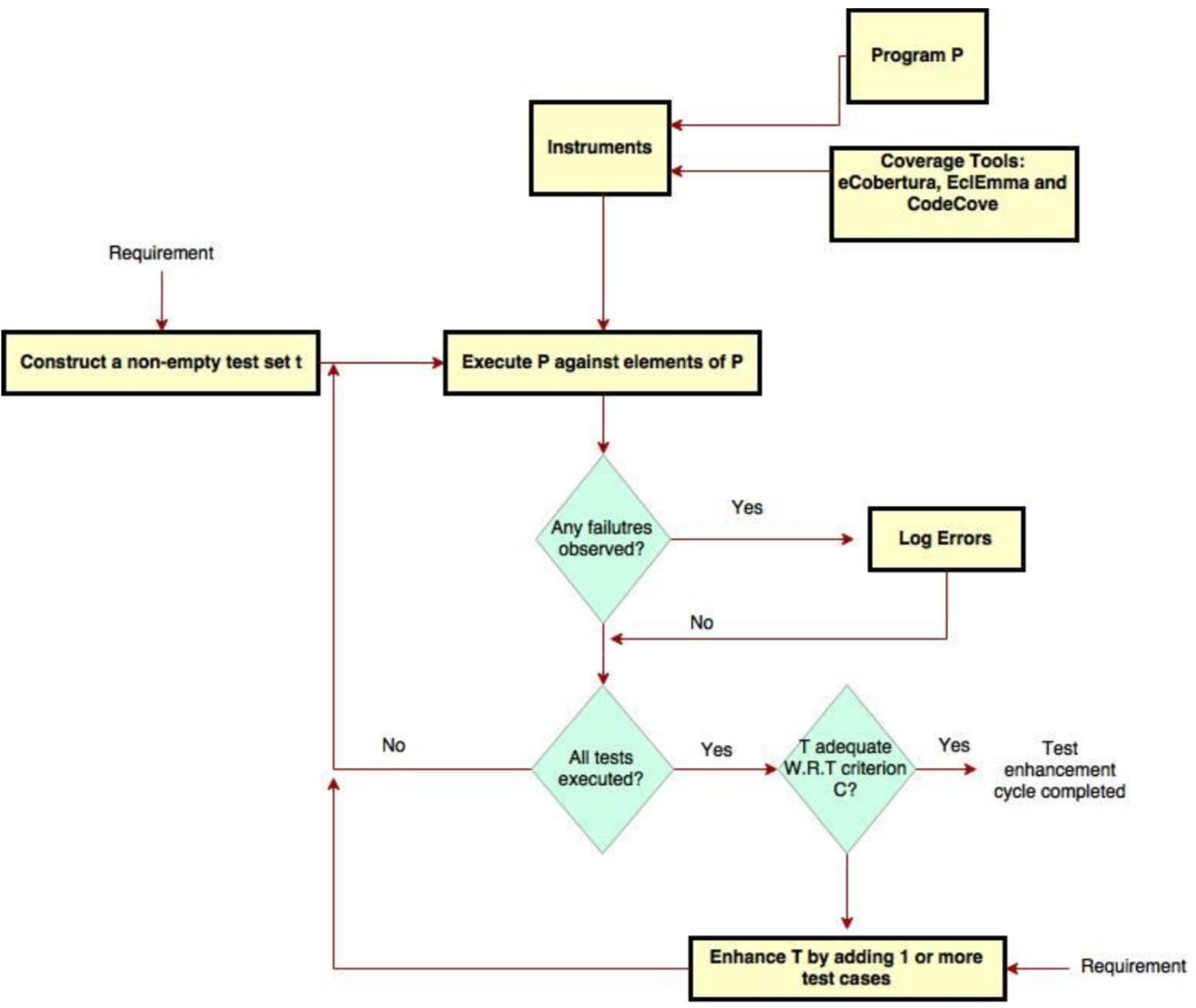


Figure 1.3.1: Initial Test plan flowchart



**Figure 1.3.2**: Testing Approach flowchart (Errors were not removed, but recommendations were made).

### 1.4 Terminology

|  |  |
| --- | --- |
| **PMS** | Payroll Management Software . |
| **JUnit** | A framework used in unit testing for the Java programming language. |
| **RFT** | Rational Functional Tester (IBM). Software used for automated systems testing that that mimic the actions and assessments of a human tester. |
| **FIU** | Florida International University. |
| **JRE** | Java Runtime Environment. |
| **Facade** | A software design pattern that involves the refactoring of connections between subsystems so that only the facade is exposed. |
| **Mockito** | A mocking framework for unit tests written in Java. |
| **PowerMockito** | Extends Mockito functionality with several new features such as mocking static and private methods. |
| **Tomcat** | Server used to host web-applications. |
| **Mysql** | Relational database system. |
| **SRD** | Software Requirement Document, used to determine requirements for the system. |

### 1.5 Document Organization

The document is organized into 9 chapters, without the introduction, that defines how the system is tested Chapter 1 gives an overview of the program under test. Chapter 2 presents the overall testing approach. Chapters 3 exercises individual classes to accomplish unit testing. Chapter 4 documents the exercising of the controller facade as a means of subsystem testing, and Chapter 5 exercises the program as whole from the perspective of the user as part of system testing.

Chapter 6 gives a summary on the results of all the tests, Chapter 7 reports the risks and contingencies in testing the system. Chapter 8 has the signatures of all team member approving of this document, Chapter 9 has all the terms used in this document and their definitions. Chapter 10 is the appendix, it contains additional information and images on references to tools used, as well as a diary of meetings and the testing schedule.

## Chapter 2: Specification Test Plan

This chapter will outline the test plan for the project which includes the roles of each member of the team, hardware and software requirements, reference materials, tested features of the project, features that were not tested and the work breakdown for this phase.

**2.1 Organization**

The table below shows the roles of the members of the team for each phase.

|  |  |  |
| --- | --- | --- |
| Team 1 Phase 1 | | |
| **Team Member** | **Team Role** | **Tasks Performed** |
| Ciana Rogers | Minute Taker | Systems Testing |
| Juan Diaz | Project Leader | SubSystem Testing |
| Ivan Rojas |  | Systems Testing |
| Dwayne Thomas | Time Keeper | Unit Testing |
| Daykel Muro |  | SubSystem Testing |
| Yovanni Jones |  | Unit Testing |

**Table 2.1**: Team roles for phase 2, Summer 2018

**2.2 Hardware and Software Requirements**

The following hardware was used to perform the tests. To replicate test results, a similar machine in terms of hardware must be used.

**Hardware Requirements**

**Machine 1 (Laptop)**

Processor: 2.3GHz dual-core Intel Core i5 4260U

RAM: 8.0 GB

Graphics Card: Intel Iris Plus Graphics 640

Hard Drive: 256 SDD

**Machine 2 (Laptop)**

Processor: Intel® Core™ i7-7500U

RAM: 8.0 GB

Graphics Card: Intel® HD Graphics 620

Hard Drive: 750GB HDD

**Machine 3 (Laptop)**

Processor: 2.23GHz dual Core Intel Core i5 5200U

Ram: 8.00 GB

Graphics Card: Intel HD Graphics 550

Hard Drive: 500GB SSD

**Machine 4 (Laptop)**

Processor: Intel Core i7-4710HQ CPU @ 2.50GHz

Ram: 16.00 GB

Graphics Card: Nvidia Geforce GTX 860m

Hard Drive: 2TB SSHD

**Software Requirements:**

o Apache Tomcat (Version 7.2): server to host web applications.

o Eclipse Oxygen (Version 4.7.3a with Java EE 8): IDE for Java and used for subsystems testing and unit testing.

o Firefox Browser Version (60.0.2): used for automated testing.

o JUnit (Version 2.0): framework for unit testing in Java.

o Mockito (Version 1.7.1): framework for mocking in Java.

o MySQL (Version 6.3) : database for the program.

o Rational Functional Tester (Version 9.2.0): used for systems testing.

o Google Chrome 67.0.3396.87 (Official Build) (64-bit): used for systems testing.

o “IBM Rational Functional Tester for Google Chrome” extension v1.3.7: enables IBM Rational Functional Tester support for web applications.

**2.3 Test Reference Items**

List of test reference items from the testing process:

* Apache Tomcat (Version 7.0) - Server to host web application.
* Eclipse Oxygen (4.7.1, Java EE) - used for automated subsystems and unit testing.
* Payroll Manager System Final User Guide PDF - user manual for the application.
* Payroll Manager System, CEN 5011 U01/RXA-Fall 2015 Final Document, authors: Joma Rodriguez, Sharan Tej Kondumuru, Amit H Shenoy, Prafulla P Ghadage, Srinivas Reddy Manda, Sai Chaithra Allala, and Yao Xiao - final documentation for application.
* Java JDK (Version 1.8.0\_72) - Java kit for developing Java applications.
* JUnit (Version 4.0) - framework for unit testing in Java.
* Mockito (Version 1.7.1) - framework for mocking in Java.
* Rational Functional Tester - used for systems testing in Java.

**2.4 Tested Features**

Those are the functionalities that were being tested:

PMS\_01\_DeleteEmp : Delete Employee

PMS\_02\_Login : Login

PMS\_03\_SearchEmp : Search Employee

PMS\_04\_ModifyTS : Modify Timesheet

PMS\_05\_ApproveTS : Approve Timesheet

PMS\_06\_CalPay : Calculate Salary

PMS\_09\_PayCheck : Paycheck

PMS\_11\_AddDept : Add Department

PMS\_13\_AddEmployee: Add Employee

PMS\_14\_UpdateEmployer : Update Employer

PMS\_19\_ForgotPwd : Forgot Password

PMS\_21\_Logout : Logout

**2.5 Features Not Tested**

PMS\_07\_GrossSal : Gross Salary

PMS\_08\_Net Sal : Net Salary

PMS\_10\_DirectDep : Direct Deposit

PMS\_12\_RemoveDept : Remove Department

PMS\_15\_RemoveEmployee : Remove Employee

PMS\_18\_NewEmployee : New Employee

PMS\_20\_PwdNotifier : Password Notifier

PMS\_001\_Injector : Injector

PMS\_103\_MultiLogin : Multiple Logins

PMS\_106\_Timeout : Timeout

**2.6 Work Breakdown**

Below is showed the breakdown for the assigned task for deliverable number 2

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Work Breakdown Phase 1** | | | | | | |
| ID | Task | Duration | Start | End | Assignee | Dependencies |
| 1 | Review program documents | 5 days | 6/14 | 5/18 | All |  |
| 2 | Deploy application to local machines | 5 days | 6/21 | 6/25 | All |  |
| 3 | Create DB schema | 5 days | 6/21 | 6/25 | Juan |  |
| 4 | Create model facade | 4 days | 6/28 | 7/1 | Ivan |  |
| 5 | Create controller facade | 4 days | 6/28 | 7/1 | Juan |  |
| 6 | Chapter 1 and 2 | 5 days | 7/4 | 7/8 | Daykel and Ciana |  |
| 7 | Chapter 3 | 5 days | 7/4 | 7/8 | Dwayne |  |
| 8 | Chapter 4 | 5 days | 7/4 | 7/8 | Juan and Daykel |  |
| 9 | Chapter 5 | 5 days | 7/4 | 7/8 | Ciana and Ivan |  |
| 10 | Chapter 6 - 10 | 2 days | 7/11 | 6/12 | All |  |
| 11 | Create presentation | 1 day | 7/15 | 7/19 | All |  |

## Chapter 3: Unit Test Cases

**3.1 Unit Test Cases**

In this unit we are going to present the test cases , identified and declare it’s objectives.

**3.1.1 Test Identification and Objective**

|  |  |
| --- | --- |
| ID | Objective |
| unitTest\_deleteempcontroller\_Sunny01 | To verify that the deleteempcontroller controller successfully lets the user delete an employee when the user enter a valid employee ID |
| unitTest\_deleteempcontroller\_Sunny02 | To verify that the deleteempcontroller controller successfully does not let the user delete an employee when the user enter a invalid employee ID |
| unitTest\_deleteempcontroller\_Sunny03 | To verify that the deleteempcontroller controller successfully does not let the user delete an employee when the user does not enter employee ID |
| unitTest\_deleteempcontroller\_Rainy01 | To verify that the deleteempcontroller controller unsuccessfully lets the user delete an employee when the user enter a valid employee ID |
| unitTest\_deleteempcontroller\_Rainy02 | To verify that the deleteempcontroller controller unsuccessfully does not let the user delete an employee when the user enter a invalid employee ID |
| unitTest\_deleteempcontroller\_Rainy03 | To verify that the deleteempcontroller controller unsuccessfully does not let the user delete an employee when the user does not enter a employee ID |
| unitTest\_Login\_Sunny01 | To verify that login controller successfully lets the user login with a valid username, password, and Id. |
| unitTest\_Login\_Sunny02 | To verify that login controller successfully does not let the user login with an invalid username, password, and Id. |
| unitTest\_Login\_Sunny03 | To verify that login controller successfully does not let the user login when the user enter a empty username, password, or Id. |
| unitTest\_Login\_Rainy01 | To verify that login controller unsuccessfully lets the user login with a valid username, password, and Id. |
| unitTest\_Login\_Rainy02 | To verify that login controller unsuccessfully does not let the user login with an invalid username, password, and Id. |
| unitTest\_Login\_Rainy03 | To verify that login controller unsuccessfully does not let the user login when the user enter an empty username, password, or Id. |
| unitTest\_SearchEmployee\_Sunny01 | To verify that searchEmployee controller successfully lets the user search for an employee when the user enters a valid employee Id. |
| unitTest\_SearchEmployee\_Sunny02 | To verify that searchEmployee controller successfully lets the user search for an employee when the user enters a valid employee Id. |
| unitTest\_SearchEmployee\_Sunny03 | To verify that searchEmployee controller successfully lets the user search for an employee when the user enters a valid employee Id. |
| unitTest\_SearchEmployee\_Rainy01 | To verify that searchEmployee controller unsuccessfully takes the user to the right page after entering a valid employee Id. |
| unitTest\_SearchEmployee\_Rainy02 | To verify that searchEmployee controller unsuccessfully takes the user to the right page after entering a valid employee Id. |
| unitTest\_SearchEmployee\_Rainy03 | To verify that searchEmployee controller unsuccessfully takes the user to the right page after entering a valid employee Id. |
| unitTest\_Authentication\_Sunny01 | To verify that the Authentication controller successfully lets the user change password by entering a valid employee id, uid, s1, a1 ,s2, a2, s3, a3,oldpass, newpass, and current password. |
| unitTest\_Authentication\_Sunny02 | To verify that the Authentication controller successfully does not let the user change password by entering an invalid uid. |
| unitTest\_Authentication\_Sunny03 | To verify that the Authentication controller successfully lets the user change password when the new password and current password are not the same. |
| unitTest\_Authentication\_Rainy01 | To verify that the Authentication controller unsuccessfully lets the user change password by entering a valid employee id, uid, s1, a1 ,s2, a2, s3, a3,oldpass, newpass, and current password. |
| unitTest\_Authentication\_Rainy02 | To verify that the Authentication controller unsuccessfully does not let the user change password by entering an invalid uid. |
| unitTest\_Authentication\_Rainy03 | To verify that the Authentication controller unsuccessfully lets the user change password when the new password and current password are the same. |

**3.1.2 Test Criteria and Procedures**

The test criteria of the test method are that the input should be the HTTPServletResponse redirecting to a page that displays an error message or a successful message. This is determined by the result of the model facade methods. These methods from the model would take the inputs in a form of a String that could be found in the database.

The procedure would include using Junit as the main testing framework for all the test cases. These test cases would include a setup and a tear down method. PowerMockito would then be used in the test methods to mock the HTTPServletRequest, HTTPServletResponse, session, and PrintWriter. Then the test methods would be used to verify if a mock object was used and if the controller redirected the HTTPServletResponse to a page that displays an error message or a successful message.

**3.1.3 Test Cases**

|  |  |
| --- | --- |
| Test ID: | unitTest\_deleteempcontroller\_Sunny01 |
| Purpose: | To verify that the deleteempcontroller controller successfully lets the user delete an employee when the user enter a valid employee ID. |
| Test Setup: | HTTPServletRequest was mocked to return valid caller, eid value when called inside controller facade.  HTTPServletResponse was mocked to capture state modified by controller facade.  Stub for Model Facade was created to contains valid employee ID found in the database. |
| Input: | Caller:delete\_emp  eid: 1 |
| Expected Output | PrintWriter object prints "<script language='javascript'>window.alert('Employee Details Deleted');window.location.replace('emplrhome.jsp');</script>" |

|  |  |
| --- | --- |
| Test ID: | unitTest\_deleteempcontroller\_Sunny02 |
| Purpose: | To verify that the deleteempcontroller controller successfully does not let the user delete an employee when the user enter a invalid employee ID. |
| Test Setup: | HTTPServletRequest was mocked to return valid caller, eid value when called inside controller facade.  HTTPServletResponse was mocked to capture state modified by controller facade.  Stub for Model Facade was created to contains valid employee ID found in the database. |
| Input: | Caller:delete\_emp  eid: 2 |
| Expected Output | HTTP Response object redirects to "error.jsp?msg="+ del.getResult() page |

|  |  |
| --- | --- |
| Test ID: | unitTest\_deleteempcontroller\_Sunny03 |
| Purpose: | To verify that the deleteempcontroller controller successfully does not let the user delete an employee when the user does not enter employee ID. |
| Test Setup: | HTTPServletRequest was mocked to return valid caller, eid value when called inside controller facade.  HTTPServletResponse was mocked to capture state modified by controller facade.  Stub for Model Facade was created to contains valid employee ID found in the database. |
| Input: | Caller:delete\_emp  eid: |
| Expected Output | HTTP Response object redirects to "error.jsp?msg="+ del.getResult() page |

|  |  |
| --- | --- |
| Test ID: | unitTest\_deleteempcontroller\_Rainy01 |
| Purpose: | To verify that the deleteempcontroller controller unsuccessfully lets the user delete an employee when the user enter a valid employee ID |
| Test Setup: | HTTPServletRequest was mocked to return valid caller, eid value when called inside controller facade.  HTTPServletResponse was mocked to capture state modified by controller facade.  Stub for Model Facade was created to contains valid employee ID found in the database. |
| Input: | Caller:delete\_emp  eid: 1 |
| Expected Output | HTTP Response object redirects to "error.jsp?msg="+ del.getResult() page |

|  |  |
| --- | --- |
| Test ID: | unitTest\_deleteempcontroller\_Rainy02 |
| Purpose: | To verify that the deleteempcontroller controller unsuccessfully does not let the user delete an employee when the user enter a invalid employee ID. |
| Test Setup: | HTTPServletRequest was mocked to return valid caller, eid value when called inside controller facade.  HTTPServletResponse was mocked to capture state modified by controller facade.  Stub for Model Facade was created to contains valid employee ID found in the database. |
| Input: | Caller:delete\_emp  eid: 2 |
| Expected Output | PrintWriter object prints "<script language='javascript'>window.alert('Employee Details Deleted');window.location.replace('emplrhome.jsp');</script>" |

|  |  |
| --- | --- |
| Test ID: | unitTest\_deleteempcontroller\_Rainy03 |
| Purpose: | To verify that the deleteempcontroller controller unsuccessfully does not let the user delete an employee when the user does not enter a employee ID |
| Test Setup: | HTTPServletRequest was mocked to return valid caller, eid value when called inside controller facade.  HTTPServletResponse was mocked to capture state modified by controller facade.  Stub for Model Facade was created to contains valid employee ID found in the database. |
| Input: | Caller:delete\_emp  eid: |
| Expected Output | PrintWriter object prints "<script language='javascript'>window.alert('Employee Details Deleted');window.location.replace('emplrhome.jsp');</script>" |

|  |  |
| --- | --- |
| Test ID: | unitTest\_Login\_Sunny01 |
| Purpose: | To verify that emplologin controller lets the user successfully login with a valid username, password, and Id. |
| Test Setup: | HTTPServletRequest was mocked to return valid caller, eid, username, and password value when called inside controller facade.  HTTPServletResponse was mocked to capture state modified by controller facade.  Stub for Model Facade was created to contains valid employee id, username, and found in the database. |
| Input: | Caller: emplogin  eid: 1  uname: emp1  pwd: pass |
| Expected Output | assertEquals("success", loh.getResult());  verify(rep).sendRedirect("employeehome.jsp"); |

|  |  |
| --- | --- |
| Test ID: | unitTest\_Login\_Sunny02 |
| Purpose: | To verify that emplologin controller does not let the user successfully login with an invalid username, password, and Id. |
| Test Setup: | HTTPServletRequest was mocked to return valid caller, eid, username, and password value when called inside controller facade.  HTTPServletResponse was mocked to capture state modified by controller facade.  Stub for Model Facade was created to contains valid employee id, username, and found in the database. |
| Input: | Caller: uemplogin  eid: 1  uname: emp2  pwd: pass |
| Expected Output | assertEquals("fail", loh.getResult());  verify(rep).sendRedirect("error.jsp?msg=" + loh.getResult()); |

|  |  |
| --- | --- |
| Test ID: | unitTest\_Login\_Sunny03 |
| Purpose: | To verify that emplologin controller does not let the user successfully login when the user enter a empty username, password, or Id. |
| Test Setup: | HTTPServletRequest was mocked to return valid caller, eid, username, and password value when called inside controller facade.  HTTPServletResponse was mocked to capture state modified by controller facade.  Stub for Model Facade was created to contains valid employee id, username, and found in the database. |
| Input: | Caller: emplogin  eid: 1  uname: emp2  pwd: pass |
| Expected Output | assertEquals("fail", loh.getResult());  verify(rep).sendRedirect("error.jsp?msg=" + loh.getResult()); |

|  |  |
| --- | --- |
| Test ID: | unitTest\_Login\_Rainy01 |
| Purpose: | To verify that emplologin controller lets the user unsuccessfully login with a valid username, password, and Id. |
| Test Setup: | HTTPServletRequest was mocked to return valid caller, eid, username, and password value when called inside controller facade.  HTTPServletResponse was mocked to capture state modified by controller facade.  Stub for Model Facade was created to contains valid employee id, username, and found in the database. |
| Input: | Caller: emplogin  eid: 1  uname: emp1  pwd: pass |
| Expected Output | assertEquals("fail", loh.getResult());  verify(rep).sendRedirect("error.jsp?msg=" + loh.getResult()); |

|  |  |
| --- | --- |
| Test ID: | unitTest\_Login\_Rainy02 |
| Purpose: | To verify that emplologin controller does not let the user unsuccessfully login with an invalid username, password, and Id. |
| Test Setup: | HTTPServletRequest was mocked to return valid caller, eid, username, and password value when called inside controller facade.  HTTPServletResponse was mocked to capture state modified by controller facade.  Stub for Model Facade was created to contains valid employee id, username, and found in the database. |
| Input: | Caller: emplogin  eid: 1  uname: emp2  pwd: pass |
| Expected Output | assertEquals("success", loh.getResult());  verify(rep).sendRedirect("employeehome.jsp"); |

|  |  |
| --- | --- |
| Test ID: | unitTest\_Login\_Rainy03 |
| Purpose: | To verify that emplologin controller does not let the user unsuccessfully login when the user enter a empty username, password, or Id. |
| Test Setup: | HTTPServletRequest was mocked to return valid caller, eid, username, and password value when called inside controller facade.  HTTPServletResponse was mocked to capture state modified by controller facade.  Stub for Model Facade was created to contains valid employee id, username, and found in the database. |
| Input: | Caller: emplogin  eid: 1  uname: emp2  pwd: pass |
| Expected Output | assertEquals("success", loh.getResult());  verify(rep).sendRedirect("employeehome.jsp"); |

|  |  |
| --- | --- |
| Test ID: | unitTest\_SearchEmployee\_Sunny01 |
| Purpose: | To verify that searchEmployee controller successfully lets the user search for an employee when the user enters a valid employee Id. |
| Test Setup: | HTTPServletRequest was mocked to return valid caller, eid value when called inside controller facade.  HTTPServletResponse was mocked to capture state modified by controller facade.  PrintWriter was mock to write character stream  Stub for Model Facade was created to contain valid employee id, found in the database. |
| Input: | Caller: emplrlogin  eid: 1 |
| Expected Output | verify(print).print("<script language='javascript'>window.alert('time sheet submitted');history.back();</script>"); |

|  |  |
| --- | --- |
| Test ID: | unitTest\_SearchEmployee\_Sunny02 |
| Purpose: | To verify that searchEmployee controller successfully lets the user search for an employee when the user enters a valid employee Id. |
| Test Setup: | HTTPServletRequest was mocked to return valid caller, eid value when called inside controller facade.  HTTPServletResponse was mocked to capture state modified by controller facade.  PrintWriter was mock to write character stream  Stub for Model Facade was created to contain valid employee id, found in the database. |
| Input: | Caller: emplrlogin  eid: 2 |
| Expected Output | verify(print).print("<script language='javascript'>window.alert('time sheet submitted');history.back();</script>"); |

|  |  |
| --- | --- |
| Test ID: | unitTest\_SearchEmployee\_Sunny03 |
| Purpose: | To verify that searchEmployee controller successfully lets the user search for an employee when the user enters a valid employee Id. |
| Test Setup: | HTTPServletRequest was mocked to return valid caller, eid value when called inside controller facade.  HTTPServletResponse was mocked to capture state modified by controller facade.  PrintWriter was mock to write character stream  Stub for Model Facade was created to contain valid employee id, found in the database. |
| Input: | Caller: emplrlogin  eid: 3 |
| Expected Output | verify(print).print("<script language='javascript'>window.alert('time sheet submitted');history.back();</script>"); |

|  |  |
| --- | --- |
| Test ID: | unitTest\_SearchEmployee\_Rainy01 |
| Purpose: | To verify that searchEmployee controller unsuccessfully takes the user to the right page after entering a valid employee Id. |
| Test Setup: | HTTPServletRequest was mocked to return valid caller, eid value when called inside controller facade.  HTTPServletResponse was mocked to capture state modified by controller facade.  PrintWriter was mock to write character stream  Stub for Model Facade was created to contain valid employee id, found in the database. |
| Input: | Caller: emplrlogin  eid: 1 |
| Expected Output | verify(print).print("<script language='javascript'>window.alert('time sheet not submitted');history.back();</script>"); |

|  |  |
| --- | --- |
| Test ID: | unitTest\_SearchEmployee\_Rainy02 |
| Purpose: | To verify that searchEmployee controller unsuccessfully takes the user to the right page after entering a valid employee Id. |
| Test Setup: | HTTPServletRequest was mocked to return valid caller, eid value when called inside controller facade.  HTTPServletResponse was mocked to capture state modified by controller facade.  PrintWriter was mock to write character stream  Stub for Model Facade was created to contain valid employee id, found in the database. |
| Input: | Caller: emplrlogin  eid: 2 |
| Expected Output | verify(print).print("<script language='javascript'>window.alert('time sheet not submitted');history.back();</script>"); |

|  |  |
| --- | --- |
| Test ID: | unitTest\_SearchEmployee\_Rainy03 |
| Purpose: | To verify that searchEmployee controller unsuccessfully takes the user to the right page after entering a valid employee Id. |
| Test Setup: | HTTPServletRequest was mocked to return valid caller, eid value when called inside controller facade.  HTTPServletResponse was mocked to capture state modified by controller facade.  PrintWriter was mock to write character stream  Stub for Model Facade was created to contain valid employee id, found in the database. |
| Input: | Caller: emplrlogin  eid: 3 |
| Expected Output | verify(print).print("<script language='javascript'>window.alert('time sheet not submitted');history.back();</script>"); |

|  |  |
| --- | --- |
| Test ID: | unitTest\_Authentication\_Sunny01 |
| Purpose: | To verify that the Authentication controller successfully lets the user change password by entering a valid employee id, uid, s1, a1 ,s2, a2, s3, a3,oldpass, newpass, and current password. |
| Test Setup: | HTTPServletRequest was mocked to return valid caller, employee id, uid, s1, a1 ,s2, a2, s3, a3,oldpass, newpass, and current password value when called inside controller facade.  HTTPServletResponse was mocked to capture state modified by controller facade.  PrintWriter was mock to write character stream  Stub for Model Facade was created to contain valid employee id, uid, s1, a1 ,s2, a2, s3, a3,oldpass, newpass, and current password, found in the database. |
| Input: | Caller: changepassword  String eid = "1";  String uid = "dthom289";  String s1 = "Favorite Color";  String a1 = "red";  String s2 = "First PEt Name";  String a2 = "shelly";  String s3 = "Favorite movie";  String a3 = "Avengers";  String oldpass = "1222";  String newpass = "1224";  String cpass = "1224"; |
| Expected Output | assertEquals("success", Auth.getResult());  verify(print).print("<script language='javascript'>window.alert('Password changed');window.location.replace('employeehome.jsp');</script>"); |

|  |  |
| --- | --- |
| Test ID: | unitTest\_Authentication\_Sunny02 |
| Purpose: | To verify that the Authentication controller successfully does not let the user change password by entering an invalid uid |
| Test Setup: | HTTPServletRequest was mocked to return valid caller, employee id, uid, s1, a1 ,s2, a2, s3, a3,oldpass, newpass, and current password value when called inside controller facade.  HTTPServletResponse was mocked to capture state modified by controller facade.  PrintWriter was mock to write character stream  Stub for Model Facade was created to contain valid employee id, uid, s1, a1 ,s2, a2, s3, a3,oldpass, newpass, and current password, found in the database. |
| Input: | Caller: changepassword  String eid = "1";  String uid = "dthom249";  String s1 = "Favorite Color";  String a1 = "red";  String s2 = "First PEt Name";  String a2 = "shelly";  String s3 = "Favorite movie";  String a3 = "Avengers";  String oldpass = "1222";  String newpass = "1224";  String cpass = "1224"; |
| Expected Output | assertEquals("fail", Auth.getResult());  verify(rep).sendRedirect("error.jsp?msg=Error :" + Auth.getResult()); |

|  |  |
| --- | --- |
| Test ID: | unitTest\_Authentication\_Sunny03 |
| Purpose: | To verify that the Authentication controller successfully lets the user change password when the new password and current password are not the same. |
| Test Setup: | HTTPServletRequest was mocked to return valid caller, employee id, uid, s1, a1 ,s2, a2, s3, a3,oldpass, newpass, and current password value when called inside controller facade.  HTTPServletResponse was mocked to capture state modified by controller facade.  PrintWriter was mock to write character stream  Stub for Model Facade was created to contain valid employee id, uid, s1, a1 ,s2, a2, s3, a3,oldpass, newpass, and current password, found in the database. |
| Input: | Caller: changepassword  String eid = "1";  String uid = "dthom289";  String s1 = "Favorite Color";  String a1 = "red";  String s2 = "First PEt Name";  String a2 = "shelly";  String s3 = "Favorite movie";  String a3 = "Avengers";  String oldpass = "1222";  String newpass = "1295";  String cpass = "1224"; |
| Expected Output | assertEquals("fail", Auth.getResult());  verify(print).print("<script language='javascript'>window.alert('Passwords not matched');window.location.replace('changepassword.jsp');</script>"); |

|  |  |
| --- | --- |
| Test ID: | unitTest\_Authentication\_Rainy01 |
| Purpose: | To verify that the Authentication controller unsuccessfully lets the user change password by entering a valid employee id, uid, s1, a1 ,s2, a2, s3, a3,oldpass, newpass, and current password. |
| Test Setup: | HTTPServletRequest was mocked to return valid caller, employee id, uid, s1, a1 ,s2, a2, s3, a3,oldpass, newpass, and current password value when called inside controller facade.  HTTPServletResponse was mocked to capture state modified by controller facade.  PrintWriter was mock to write character stream  Stub for Model Facade was created to contain valid employee id, uid, s1, a1 ,s2, a2, s3, a3,oldpass, newpass, and current password, found in the database. |
| Input: | Caller: changepassword  String eid = "1";  String uid = "dthom249";  String s1 = "Favorite Color";  String a1 = "red";  String s2 = "First PEt Name";  String a2 = "shelly";  String s3 = "Favorite movie";  String a3 = "Avengers";  String oldpass = "1222";  String newpass = "1224";  String cpass = "1224"; |
| Expected Output | assertEquals("fail", Auth.getResult());  verify(print).print("error.jsp?msg=Error :" + Auth.getResult()); |

|  |  |
| --- | --- |
| Test ID: | unitTest\_Authentication\_Rainy02 |
| Purpose: | To verify that the Authentication controller unsuccessfully does not let the user change password by entering an invalid uid |
| Test Setup: | HTTPServletRequest was mocked to return valid caller, employee id, uid, s1, a1 ,s2, a2, s3, a3,oldpass, newpass, and current password value when called inside controller facade.  HTTPServletResponse was mocked to capture state modified by controller facade.  PrintWriter was mock to write character stream  Stub for Model Facade was created to contain valid employee id, uid, s1, a1 ,s2, a2, s3, a3,oldpass, newpass, and current password, found in the database. |
| Input: | Caller: changepassword  String eid = "1";  String uid = "dthom249";  String s1 = "Favorite Color";  String a1 = "red";  String s2 = "First PEt Name";  String a2 = "shelly";  String s3 = "Favorite movie";  String a3 = "Avengers";  String oldpass = "1222";  String newpass = "1224";  String cpass = "1224"; |
| Expected Output | assertEquals("success", Auth.getResult());  verify(print).print("<script language='javascript'>window.alert('Password changed');window.location.replace('employeehome.jsp');</script>"); |

|  |  |
| --- | --- |
| Test ID: | unitTest\_Authentication\_Rainy03 |
| Purpose: | To verify that the Authentication controller unsuccessfully lets the user change password when the new password and current password are not the same. |
| Test Setup: | HTTPServletRequest was mocked to return valid caller, employee id, uid, s1, a1 ,s2, a2, s3, a3,oldpass, newpass, and current password value when called inside controller facade.  HTTPServletResponse was mocked to capture state modified by controller facade.  PrintWriter was mock to write character stream  Stub for Model Facade was created to contain valid employee id, uid, s1, a1 ,s2, a2, s3, a3,oldpass, newpass, and current password, found in the database. |
| Input: | Caller: changepassword  String eid = "1";  String uid = "dthom289";  String s1 = "Favorite Color";  String a1 = "red";  String s2 = "First PEt Name";  String a2 = "shelly";  String s3 = "Favorite movie";  String a3 = "Avengers";  String oldpass = "1222";  String newpass = "1224";  String cpass = "1224"; |
| Expected Output | assertEquals("fail", Auth.getResult());  verify(print).print("<script language='javascript'>window.alert('Passwords not matched');window.location.replace('changepassword.jsp');</script>"); |

**3.1.4 Actual Test Results**

|  |  |  |
| --- | --- | --- |
| ID | Results | |
| unitTest\_deleteempcontroller\_Sunny01 | Pass | |
| unitTest\_deleteempcontroller\_Sunny02 | Pass | |
| unitTest\_deleteempcontroller\_Sunny03 | Pass | |
| unitTest\_deleteempcontroller\_Rainy01 | Fail |
| unitTest\_deleteempcontroller\_Rainy02 | Fail |
| unitTest\_deleteempcontroller\_Rainy03 | Fail |
| unitTest\_Login\_Sunny01 | Pass |
| unitTest\_Login\_Sunny02 | Pass |
| unitTest\_Login\_Sunny03 | Pass |
| unitTest\_Login\_Rainy01 | Fail |
| unitTest\_Login\_Rainy02 | Fail |
| unitTest\_Login\_Rainy03 | Fail |
| unitTest\_SearchEmployee\_Sunny01 | Pass |
| unitTest\_SearchEmployee\_Sunny02 | Pass |
| unitTest\_SearchEmployee\_Sunny03 | Pass |
| unitTest\_SearchEmployee\_Rainy01 | Fail |
| unitTest\_SearchEmployee\_Rainy02 | Fail |
| unitTest\_SearchEmployee\_Rainy03 | Fail |
| unitTest\_Authentication\_Sunny01 | Pass |
| unitTest\_Authentication\_Sunny02 | Pass |
| unitTest\_Authentication\_Sunny03 | Pass |
| unitTest\_Authentication\_Rainy01 | Fail |
| unitTest\_Authentication\_Rainy02 | Fail |
| unitTest\_Authentication\_Rainy03 | Fail |

## Chapter 4 Subsystem Testing

This chapter summarizes all the subsystems test cases, test criteria, procedures, and test results for the **Application Logic** package subsystem.

**4.1. Subsystem Test Cases**

There is a total of twelve test cases for subsystems testing. These tests are designed to show that the connections between subsystems and **Application Logic** are functioning properly. This section goes over the test cases, their objectives, and criteria, as well as the procedures used in testing.

### 4.1.1 Test Identification and Objective (Summary)

The case list below shows the ID and objectives of each test cases performed for the controller subsystem. Where the object mentions valid values, it refers to specific instances of Model classes attributes that were hard corded into the stubs. Refer to Appendix F for the state of the database stubbed in.

|  |  |
| --- | --- |
| ID | Objective |
| subSystem\_emplrlogincontroller\_Sunny01 | To verify that employer authentication controller lets user with the valid username, password combination into the system. |
| subSystem\_emplrlogincontroller\_Sunny01 | To verify that employer authentication controller lets user with the valid username, password combination into the system. |
| subSystem\_emplrlogincontroller\_Sunny01 | To verify that employer authentication controller lets user with the valid username, password combination into the system. |
| subSystem\_emplrlogincontroller\_Rainy01 | To verify that employer authentication controller redirects to Error page when using credentials not in the system. |
| subSystem\_emplrlogincontroller\_Rainy02 | To verify that employer authentication controller redirects to Error page when using credentials not in the system. |
| subSystem\_emplrlogincontroller\_Rainy03 | To verify that employer authentication controller redirects to Error page when using credentials not in the system. |
| subSystem\_login\_Sunny01 | To verify that employee authentication controller redirects to “employeehome” page when using valid credentials. |
| subSystem\_login\_Sunny02 | To verify that employee authentication controller redirects to “employeehome” page when using valid credentials. |
| subSystem\_login\_Sunny03 | To verify that employee authentication controller redirects to “employeehome” page when using valid credentials. |
| subSystem\_login\_Rainy01 | To verify that employee authentication controller redirects to “error.jsp” page when using invalid credentials. |
| subSystem\_login\_Rainy02 | To verify that employee authentication controller redirects to “error.jsp” page when using invalid credentials. |
| subSystem\_login\_Rainy03 | To verify that employee authentication controller redirects to “error.jsp” page when using invalid credentials. |
| subSystem\_Edit\_Timesheet\_Control\_Sunny01 | To verify that an employer logged-in is shown a successful message if the timesheet was updated. |
| subSystem\_Edit\_Timesheet\_Control\_Sunny02 | To verify that an employer logged-in is shown a successful message if the timesheet was updated. |
| subSystem\_Edit\_Timesheet\_Control\_Sunny03 | To verify that an employer logged-in is shown a successful message if the timesheet was updated. |
| subSystem\_Edit\_Timesheet\_Control\_Rainy01 | To verify that a value in the incorrect time-format causes a failure to update the employee time sheet. |
| subSystem\_Edit\_Timesheet\_Control\_Rainy02 | To verify that a value in the incorrect time-format causes a failure to update the employee time sheet. |
| subSystem\_Edit\_Timesheet\_Control\_Rainy03 | To verify that a value in the incorrect time-format causes a failure to update the employee time sheet. |
| subSystem\_deleteempcontroller\_Sunny01 | To verify that a valid emID passed results in employee record being deleted. |
| subSystem\_deleteempcontroller\_Sunny02 | To verify that a valid emID passed results in employee record being deleted. |
| subSystem\_deleteempcontroller\_Sunny03 | To verify that a valid emID passed results in employee record being deleted. |
| subSystem\_deleteempcontroller\_Rainy01 | To verify that a invalid eid passed results in fault message. |
| subSystem\_deleteempcontroller\_Rainy02 | To verify that a invalid eid passed results in fault message. |
| subSystem\_deleteempcontroller\_Rainy03 | To verify that a invalid eid passed results in fault message. |
| subSystem\_Get\_Profile\_Sunny01 | To verify that an employee profile can be updated for an existing employee. |
| subSystem\_Get\_Profile\_Sunny02 | To verify that an employee profile can be updated for an existing employee. |
| subSystem\_Get\_Profile\_Sunny03 | To verify that an employee profile can be updated for an existing employee. |
| subSystem\_Get\_Profile\_Rainy01 | To verify that an employee profile cannot be updated for an eid that does not exists. |
| subSystem\_Get\_Profile\_Rainy02 | To verify that an employee profile cannot be updated for an eid that does not exists. |
| subSystem\_Get\_Profile\_Rainy03 | To verify that an employee profile cannot be updated for an eid that does not exists. |
| subSystem\_empregistration\_Sunny01 | To verify that an employee with an existing employeeID is able to register |
| subSystem\_empregistration\_Sunny02 | To verify that an employee with an existing employeeID is able to register |
| subSystem\_empregistration\_Sunny03 | To verify that an employee with an existing employeeID is able to register |
| subSystem\_empregistration\_Rainy01 | To verify that an employee without an existing employeeID is not able to register. |
| subSystem\_empregistration\_Rainy02 | To verify that an employee without an existing employeeID is not able to register. |
| subSystem\_empregistration\_Rainy03 | To verify that an employee without an existing employeeID is not able to register. |
| subSystem\_Search\_Employee\_Sunny01 | To verify that an existing employee is able to submit a timesheet for an employer to review. |
| subSystem\_Search\_Employee\_Sunny02 | To verify that an existing employee is able to submit a timesheet for an employer to review. |
| subSystem\_Search\_Employee\_Sunny03 | To verify that an existing employee is able to submit a timesheet for an employer to review. |
| subSystem\_Search\_Employee\_Rainy01 | To Verify that an employee without timesheet is not shown the “timesheet submitted” message. |
| subSystem\_Search\_Employee\_Rainy02 | To Verify that an employee without timesheet is not shown the “timesheet submitted” message. |
| subSystem\_Search\_Employee\_Rainy03 | To Verify that an employee without timesheet is not shown the “timesheet submitted” message. |
| subSystem\_forgotpasswordcontroller\_Sunny01 | To Verify that an employee that has registered is able to retrieve their password by providing eid, uid, and answers to 3 security questions |
| subSystem\_forgotpasswordcontroller\_Sunny02 | To Verify that an employee that has registered is able to retrieve their password by providing eid, uid, and answers to 3 security questions |
| subSystem\_forgotpasswordcontroller\_Sunny03 | To Verify that an employee that has registered is able to retrieve their password by providing eid, uid, and answers to 3 security questions |
| subSystem\_forgotpasswordcontroller\_Rainy01 | To Verify that an employee that has not registered or that does not provide the correct answers to the security fields is not given a password. |
| subSystem\_forgotpasswordcontroller\_Rainy02 | To Verify that an employee that has not registered or that does not provide the correct answers to the security fields is not given a password. |
| subSystem\_forgotpasswordcontroller\_Rainy03 | To Verify that an employee that has not registered or that does not provide the correct answers to the security fields is not given a password. |
| subSystem\_Paycheck\_Generator\_Sunny01 | To Verify that modifications to the Paycheck are completed. |
| subSystem\_Paycheck\_Generator\_Sunny02 | To Verify that modifications to the Paycheck are completed. |
| subSystem\_Paycheck\_Generator\_Sunny03 | To Verify that modifications to the Paycheck are completed. |
| subSystem\_Paycheck\_Generator\_Rainy01 | To Verify that wrong modifications to the Paycheck are no completed. |
| subSystem\_Paycheck\_Generator\_Rainy02 | To Verify that wrong modifications to the Paycheck are no completed. |
| subSystem\_Paycheck\_Generator\_Rainy03 | To Verify that wrong modifications to the Paycheck are no completed. |
| subSystem\_Authentication\_Sunny01 | To Verify that updated password operation is done with proper parameters (Security Questions). |
| subSystem\_Authentication\_Sunny02 | To Verify that updated password operation is done with proper parameters (Security Questions). |
| subSystem\_Authentication\_Sunny03 | To Verify that updated password operation is done with proper parameters (Security Questions). |
| subSystem\_Authentication\_Rainy01 | To Verify that updated password operation is not allowed without proper parameters (Security Questions). |
| subSystem\_Authentication\_Rainy02 | To Verify that updated password operation is not allowed without proper parameters (Security Questions). |
| subSystem\_Authentication\_Rainy03 | To Verify that updated password operation is not allowed without proper parameters (Security Questions). |

### 4.1.2. Test Criteria and Procedures

**Test Criteria**

Each controller method’s last action is redirection to an error page or any other page. Therefore, the test criterion chosen is that tests exercise the two possible paths. Valid values should redirect to a page that is not an error. Invalid values should redirect to a page that is an error. The page should be determined by examining the last line of code of the method under test.

Input values are all considered to be of String type by the controller methods. As such, input values for tests are partitioned into two equivalence classes. Values that would be in the database and values that would not. In addition, whenever an integer is a String, we treat the String as an integer and perform boundary analysis on the valid partition.

**Procedures**

Every method in the controller subsystem was exercised by mocking the HTTPServletRequest and HTTPServletResponse classes, along other HTTP bases classes such as PrintWriter and Session because the program under test is a web application. Aforementioned classes are implemented by a web server supporting Java servlets (e.g TomCat) during production.

When a test case refers to “valid” inputs, it refer to values that the application would have found in the database, but that instead been setup in the stubs of the Model controller classes for the purpose of these tests. Appendix F contains the state of the database that the mocks are based on.

A test is considered to pass if we can verify that the expected web page is displayed given some parameters. This represents the least intrusive way of verifying results as we need only check the state of the subsystem as opposed to a value returned. Had we chosen to verify a value returned, this would have introduced unnecessary complexity in adding attribute values and methods not present in the original implementation of the software.

### 4.1.3 Test Cases

|  |  |
| --- | --- |
| Test ID: | subSystem\_emplrlogincontroller\_Sunny01 |
| Purpose: | To verify that employer authentication controller lets user with the valid username, password combination into the system. |
| Test Setup: | HTTPServletRequest was mocked to return valid caller, username, and password values when called inside controller facade.  HTTPServletResponse was mocked to capture state modified by controller facade.  Stub for Employer Model was created to contains valid username-password combinations found in the database. Refer to Appendix F. |
| Input: | Caller: emplrlogin  Username: galaxy  Password: deathstar |
| Expected Output | HTTP Response object redirects to “emplrhome.jsp” page |

|  |  |
| --- | --- |
| Test ID: | subSystem\_emplrlogincontroller\_Sunny02 |
| Purpose: | To verify that employer authentication controller lets user with the valid username, password combination into the system. |
| Test Setup: | HTTPServletRequest was mocked to return valid caller, username, and password values when called inside controller facade.  HTTPServletResponse was mocked to capture state modified by controller facade.  Stub for Employer Model was created to contains valid username-password combinations found in the database. Refer to Appendix F. |
| Input: | Caller: emplrlogin  Username: galaxy  Password: deathstar |
| Expected Output | HTTP Response object redirects to “emplrhome.jsp” page |

|  |  |
| --- | --- |
| Test ID: | subSystem\_emplrlogincontroller\_Sunny03 |
| Purpose: | To verify that employer authentication controller lets user with the valid username, password combination into the system. |
| Test Setup: | HTTPServletRequest was mocked to return valid caller, username, and password values when called inside controller facade.  HTTPServletResponse was mocked to capture state modified by controller facade.  Stub for Employer Model was created to contains valid username-password combinations found in the database. Refer to Appendix F. |
| Input: | Caller: emplrlogin  Username: galaxy3  Password: deathstar3 |
| Expected Output | HTTP Response object redirects to “emplrhome.jsp” page |

|  |  |
| --- | --- |
| Test ID: | subSystem\_emplrlogincontroller\_Rainy01 |
| Purpose: | To verify that employer authentication controller redirects to Error page when using credentials not in the system. |
| Test Setup: | HTTPServletRequest was mocked to return valid caller, username values when called inside controller facade. Mocked to return invalid password value when called inside the controller face.  HTTPServletResponse was mocked to capture state modified by controller facade.  Stub for Employer Model was created to contains valid username-password combinations found in the database. Refer to Appendix F. |
| Input: | Caller: emplrlogin  Username: galaxy  Password: wrongpassword |
| Expected Output | HTTP Response object redirects to “error.jsp” page |

|  |  |
| --- | --- |
| Test ID: | subSystem\_emplrlogincontroller\_Rainy02 |
| Purpose: | To verify that employer authentication controller redirects to Error page when using credentials not in the system. |
| Test Setup: | HTTPServletRequest was mocked to return valid caller, password values when called inside controller facade. Mocked to return invalid username value when called inside the controller face.  HTTPServletResponse was mocked to capture state modified by controller facade.  Stub for Employer Model was created to contains valid username-password combinations found in the database. Refer to Appendix F. |
| Input: | Caller: emplrlogin  Username:wrongusername  Password: deathstar |
| Expected Output | HTTP Response object redirects to “error.jsp” page |

|  |  |
| --- | --- |
| Test ID: | subSystem\_emplrlogincontroller\_Rainy03 |
| Purpose: | To verify that employer authentication controller redirects to Error page when using credentials not in the system. |
| Test Setup: | HTTPServletRequest was mocked to return valid caller value inside controller facade. Mocked to return invalid username, password values when called inside the controller face.  HTTPServletResponse was mocked to capture state modified by controller facade.  Stub for Employer Model was created to contains valid username-password combinations found in the database. Refer to Appendix F. |
| Input: | Caller: emplrlogin  Username: <empty string>  Password: <empty string> |
| Expected Output | HTTP Response object redirects to “error.jsp” page |

|  |  |
| --- | --- |
| Test ID: | subSystem\_login\_Sunny01 |
| Purpose: | To verify that employee authentication controller redirects to “employeehome” page when using valid credentials. |
| Test Setup: | HTTPServletRequest was mocked to return valid caller, empid, username, and password values when called inside controller facade.  HTTPServletResponse was mocked to capture state modified by controller facade.  Stub for Employer Model was created to contains valid empid-username-password combinations found in the database. Refer to Appendix F. |
| Input: | Caller: emplogin  empid: 1  Username: emp1  Password: pass1 |
| Expected Output | HTTP Response object redirects to “employeehome” page |

|  |  |
| --- | --- |
| Test ID: | subSystem\_login\_Sunny02 |
| Purpose: | To verify that employee authentication controller redirects to “employeehome” page when using valid credentials. |
| Test Setup: | HTTPServletRequest was mocked to return valid caller, empid, username, and password values when called inside controller facade.  HTTPServletResponse was mocked to capture state modified by controller facade.  Stub for Employer Model was created to contains valid empid-username-password combinations found in the database. Refer to Appendix F. |
| Input: | Caller: emplogin  empid: 5  Username: emp5  Password: pass5 |
| Expected Output | HTTP Response object redirects to “employeehome” page |

|  |  |
| --- | --- |
| Test ID: | subSystem\_login\_Sunny03 |
| Purpose: | To verify that employee authentication controller redirects to “employeehome” page when using valid credentials. |
| Test Setup: | HTTPServletRequest was mocked to return valid caller, empid, username, and password values when called inside controller facade.  HTTPServletResponse was mocked to capture state modified by controller facade.  Stub for Employer Model was created to contains valid empid-username-password combinations found in the database. Refer to Appendix F. |
| Input: | Caller: emplogin  empid: 10  Username: emp10  Password: pass10 |
| Expected Output | HTTP Response object redirects to “employeehome” page |

|  |  |
| --- | --- |
| Test ID: | subSystem\_login\_Rainy01 |
| Purpose: | To verify that employee authentication controller redirects to “error.jsp” page when using invalid credentials. |
| Test Setup: | HTTPServletRequest was mocked to return valid caller, username, and password values when called inside controller facade.  HTTPServletRequest was mocked to return invalid caller, empid value when called inside controller facade.  HTTPServletResponse was mocked to capture state modified by controller facade.  Stub for Employer Model was created to contains valid empid-username-password combinations found in the database. Refer to Appendix F. |
| Input: | Caller: emplogin  empid: 192  Username: emp1  Password: pass1 |
| Expected Output | HTTP Response object redirects to “error.jsp” page |

|  |  |
| --- | --- |
| Test ID: | subSystem\_login\_Rainy02 |
| Purpose: | To verify that employee authentication controller redirects to “error.jsp” page when using invalid credentials. |
| Test Setup: | HTTPServletRequest was mocked to return valid caller, username, and password values when called inside controller facade.  HTTPServletRequest was mocked to return invalid caller, empid value when called inside controller facade.  HTTPServletResponse was mocked to capture state modified by controller facade.  Stub for Employer Model was created to contains valid empid-username-password combinations found in the database. Refer to Appendix F. |
| Input: | Caller: emplogin  empid: -10  Username: emp1  Password: pass1 |
| Expected Output | HTTP Response object redirects to “error.jsp” page |

|  |  |
| --- | --- |
| Test ID: | subSystem\_login\_Rainy03 |
| Purpose: | To verify that employee authentication controller redirects to “error.jsp” page when using invalid credentials. |
| Test Setup: | HTTPServletRequest was mocked to return valid caller, username, and password values when called inside controller facade.  HTTPServletRequest was mocked to return invalid caller, empid value when called inside controller facade.  HTTPServletResponse was mocked to capture state modified by controller facade.  Stub for Employer Model was created to contains valid empid-username-password combinations found in the database. Refer to Appendix F. |
| Input: | Caller: emplogin  empid: 0  Username: emp1  Password: pass1 |
| Expected Output | HTTP Response object redirects to “error.jsp” page |

|  |  |
| --- | --- |
| Test ID: | subSystem\_Edit\_Timesheet\_Control\_Sunny01 |
| Purpose: | To verify that an employer logged-in is shown a successful message if the timesheet was updated. |
| Test Setup: | HTTPServletRequest was mocked to return valid caller, timeRecordID, clockin, lunchout, lunchin, clockout, and totalhours values. It was also mocked to return a mocked Session object.  HTTPServletResponse was mocked to capture state and to return a mocked PrintWriter object.  Stub for TimeSheet Model was created to accept all the passed values as long as they were in correct time format and valid timerecordID exists. Refer to Appendix F. |
| Input: | caller = view\_timesheet  etsid1 = 1111  intime1 = 11:00:00  lunchout = 14:00:00  lunchin1 = 15:00:00  outtime = 17:00:00  thours1 = 5.00 |
| Expected Output | Window Alert with “time sheet updated” is displayed. |

|  |  |
| --- | --- |
| Test ID: | subSystem\_Edit\_Timesheet\_Control\_Sunny02 |
| Purpose: | To verify that an employer logged-in is shown a successful message if the timesheet was updated. |
| Test Setup: | HTTPServletRequest was mocked to return valid caller, timeRecordID, clockin, lunchout, lunchin, clockout, and totalhours values. It was also mocked to return a mocked Session object.  HTTPServletResponse was mocked to capture state and to return a mocked PrintWriter object.  Stub for TimeSheet Model was created to accept all the passed values as long as they were in correct time format and valid timerecordID exists. Refer to Appendix F. |
| Input: | caller = view\_timesheet  etsid1 = 2222  intime1 = 00:00:00  lunchout = 14:00:00  lunchin1 = 15:00:00  outtime = 24:00:00  thours1 = 24.00 |
| Expected Output | Window Alert with “time sheet updated” is displayed. |

|  |  |
| --- | --- |
| Test ID: | subSystem\_Edit\_Timesheet\_Control\_Sunny03 |
| Purpose: | To verify that an employer logged-in is shown a successful message if the timesheet was updated. |
| Test Setup: | HTTPServletRequest was mocked to return valid caller, timeRecordID, clockin, lunchout, lunchin, clockout, and totalhours values. It was also mocked to return a mocked Session object.  HTTPServletResponse was mocked to capture state and to return a mocked PrintWriter object.  Stub for TimeSheet Model was created to accept all the passed values as long as they were in correct time format and valid timerecordID exists. Refer to Appendix F. |
| Input: | caller = view\_timesheet  etsid1 = 1111  intime1 = 00:00:00  lunchout = 14:00:00  lunchin1 = 14:00:00  outtime = 14:00:00  thours1 = 14.00 |
| Expected Output | Window Alert with “time sheet updated” is displayed. |

|  |  |
| --- | --- |
| Test ID: | subSystem\_Edit\_Timesheet\_Control\_Rainy01 |
| Purpose: | To verify that a value in the incorrect time-format causes a failure to update the employee time sheet. |
| Test Setup: | HTTPServletRequest was mocked to return valid caller, timeRecordID, clockin, lunchout, lunchin, clockout, and totalhours values. It was also mocked to return a mocked Session object.  HTTPServletResponse was mocked to capture state and to return a mocked PrintWriter object.  Stub for TimeSheet Model was created to accept all the passed values as long as they were in correct time format and valid timerecordID exists. Refer to Appendix F. |
| Input: | caller = view\_timesheet  etsid1 = 1111  intime1 = <empty string>  lunchout = 14:00:00  lunchin1 = 14:00:00  outtime = 14:00:00  thours1 = 14.00 |
| Expected Output | HTML page with “Error: fail” is displayed. |

|  |  |
| --- | --- |
| Test ID: | subSystem\_Edit\_Timesheet\_Control\_Rainy02 |
| Purpose: | To verify that a value in the incorrect time-format causes a failure to update the employee time sheet. |
| Test Setup: | HTTPServletRequest was mocked to return valid caller, timeRecordID, clockin, lunchout, lunchin, clockout, and totalhours values. It was also mocked to return a mocked Session object.  HTTPServletResponse was mocked to capture state and to return a mocked PrintWriter object.  Stub for TimeSheet Model was created to accept all the passed values as long as they were in correct time format and valid timerecordID exists. Refer to Appendix F. |
| Input: | caller = view\_timesheet  etsid1 = 1111  intime1 = <empty string>  lunchout = 14:00:00  lunchin1 = 14:00:00  outtime = 14:00:00  thours1 = 14.00 |
| Expected Output | HTML page with “Error: fail” is displayed. |

|  |  |
| --- | --- |
| Test ID: | subSystem\_Edit\_Timesheet\_Control\_Rainy03 |
| Purpose: | To verify that a value in the incorrect time-format causes a failure to update the employee time sheet. |
| Test Setup: | HTTPServletRequest was mocked to return valid caller, timeRecordID, clockin, lunchout, lunchin, clockout, and totalhours values. It was also mocked to return a mocked Session object.  HTTPServletResponse was mocked to capture state and to return a mocked PrintWriter object.  Stub for TimeSheet Model was created to accept all the passed values as long as they were in correct time format and valid timerecordID exists. Refer to Appendix F. |
| Input: | caller = view\_timesheet  etsid1 = 1111  intime1 = test  lunchout = 14:00:00  lunchin1 = 14:00:00  outtime = 14:00:00  thours1 = 14.00 |
| Expected Output | HTML page with “Error: fail” is displayed. |

|  |  |
| --- | --- |
| Test ID: | subSystem\_deleteempcontroller\_Sunny01 |
| Purpose: | To verify that a valid emID passed results in employee record being deleted. |
| Test Setup: | HTTPServletRequest was mocked to return valid caller and eid value.  HTTPServletResponse was mocked to capture state and to return a mocked PrintWriter object.  Stub for Employee Model was created to return “success” if eid exists or “fail” if it does not. Refer to Appendix F. |
| Input: | caller = delete\_emp  eid=1 |
| Expected Output | Window alert with “employee details deleted message appears.” |

|  |  |
| --- | --- |
| Test ID: | subSystem\_deleteempcontroller\_Sunny02 |
| Purpose: | To verify that a valid emID passed results in employee record being deleted. |
| Test Setup: | HTTPServletRequest was mocked to return valid caller and eid value.  HTTPServletResponse was mocked to capture state and to return a mocked PrintWriter object.  Stub for Employee Model was created to return “success” if eid exists or “fail” if it does not. Refer to Appendix F. |
| Input: | caller = delete\_emp  eid=5 |
| Expected Output | Window alert with “employee details deleted message appears.” |

|  |  |
| --- | --- |
| Test ID: | subSystem\_deleteempcontroller\_Sunny03 |
| Purpose: | To verify that a valid emID passed results in employee record being deleted. |
| Test Setup: | HTTPServletRequest was mocked to return valid caller and eid value.  HTTPServletResponse was mocked to capture state and to return a mocked PrintWriter object.  Stub for Employee Model was created to return “success” if eid exists or “fail” if it does not. Refer to Appendix F. |
| Input: | caller = delete\_emp  eid=10 |
| Expected Output | Window alert with “employee details deleted message appears.” |

|  |  |
| --- | --- |
| Test ID: | subSystem\_deleteempcontroller\_Rainy01 |
| Purpose: | To verify that a invalid eid passed results in fault message. |
| Test Setup: | HTTPServletRequest was mocked to return valid caller value and invalid eid value.  HTTPServletResponse was mocked to capture state and to return a mocked PrintWriter object.  Stub for Employee Model was created to return “success” if eid exists or “fail” if it does not. Refer to Appendix F. |
| Input: | caller = delete\_emp  eid = -10 |
| Expected Output | Response object redirects to “error.jsp” page. |

|  |  |
| --- | --- |
| Test ID: | subSystem\_deleteempcontroller\_Rainy02 |
| Purpose: | To verify that a invalid eid passed results in fault message. |
| Test Setup: | HTTPServletRequest was mocked to return valid caller value and invalid eid value.  HTTPServletResponse was mocked to capture state and to return a mocked PrintWriter object.  Stub for Employee Model was created to return “success” if eid exists or “fail” if it does not. Refer to Appendix F. |
| Input: | caller = delete\_emp  eid = 0 |
| Expected Output | Response object redirects to “error.jsp” page. |

|  |  |
| --- | --- |
| Test ID: | subSystem\_deleteempcontroller\_Rainy03 |
| Purpose: | To verify that a invalid eid passed results in fault message. |
| Test Setup: | HTTPServletRequest was mocked to return valid caller value and invalid eid value.  HTTPServletResponse was mocked to capture state and to return a mocked PrintWriter object.  Stub for Employee Model was created to return “success” if eid exists or “fail” if it does not. Refer to Appendix F. |
| Input: | caller = delete\_emp  eid = 15 |
| Expected Output | Response object redirects to “error.jsp” page. |

|  |  |
| --- | --- |
| Test ID: | subSystem\_Get\_Profile\_Sunny01 |
| Purpose: | To verify that an employee profile can be updated for an existing employee. |
| Test Setup: | HTTPServletRequest was mocked to return valid caller, eid, fname, lname, gen, dob, job, contact, e-mail, addr, accno, bname values.  HTTPServletResponse was mocked to capture state and to return a mocked PrintWriter object.  Stub for Employee Model was created to return “success” if eid exists or “fail” if it does not. Refer to Appendix F. |
| Input: | caller = update\_emp\_rec  eid = 1  fname=1  lanem=1  gen=1  dob=1  job=1  contact=1  email=1  addr=1  bname=1 |
| Expected Output | PrintWriter object display “Employee Details Updated” windows and then the “update\_emp\_jsp” web page. |

|  |  |
| --- | --- |
| Test ID: | subSystem\_Get\_Profile\_Sunny02 |
| Purpose: | To verify that an employee profile can be updated for an existing employee. |
| Test Setup: | HTTPServletRequest was mocked to return valid caller, eid, fname, lname, gen, dob, job, contact, e-mail, addr, accno, bname values.  HTTPServletResponse was mocked to capture state and to return a mocked PrintWriter object.  Stub for Employee Model was created to return “success” if eid exists or “fail” if it does not. Refer to Appendix F. |
| Input: | caller = update\_emp\_rec  eid = 5  fname=1  lanem=1  gen=1  dob=1  job=1  contact=1  email=1  addr=1  bname=1 |
| Expected Output | PrintWriter object display “Employee Details Updated” windows and then the “update\_emp\_jsp” web page. |

|  |  |
| --- | --- |
| Test ID: | subSystem\_Get\_Profile\_Sunny03 |
| Purpose: | To verify that an employee profile can be updated for an existing employee. |
| Test Setup: | HTTPServletRequest was mocked to return valid caller, eid, fname, lname, gen, dob, job, contact, e-mail, addr, accno, bname values.  HTTPServletResponse was mocked to capture state and to return a mocked PrintWriter object.  Stub for Employee Model was created to return “success” if eid exists or “fail” if it does not. Refer to Appendix F. |
| Input: | caller = update\_emp\_rec  eid = 10  fname=1  lanem=1  gen=1  dob=1  job=1  contact=1  email=1  addr=1  bname=1 |
| Expected Output | PrintWriter object display “Employee Details Updated” windows and then the “update\_emp\_jsp” web page. |

|  |  |
| --- | --- |
| Test ID: | subSystem\_Get\_Profile\_Rainy01 |
| Purpose: | To verify that an employee profile cannot be updated for an eid that does not exists. |
| Test Setup: | HTTPServletRequest was mocked to return invalid eid value.  HTTPServletRequest was mocked to return valid caller, fname, lname, gen, dob, job, contact, e-mail, addr, accno, bname values.  HTTPServletResponse was mocked to capture state and to return a mocked PrintWriter object.  Stub for Employee Model was created to return “success” if eid exists or “fail” if it does not. Refer to Appendix F. |
| Input: | caller = update\_emp\_rec  eid = -10  fname=1  lanem=1  gen=1  dob=1  job=1  contact=1  email=1  addr=1  bname=1 |
| Expected Output | Response is redirect to “"error.jsp?msg=fail” page. |

|  |  |
| --- | --- |
| Test ID: | subSystem\_Get\_Profile\_Rainy02 |
| Purpose: | To verify that an employee profile cannot be updated for an eid that does not exists. |
| Test Setup: | HTTPServletRequest was mocked to return invalid eid value.  HTTPServletRequest was mocked to return valid caller, fname, lname, gen, dob, job, contact, e-mail, addr, accno, bname values.  HTTPServletResponse was mocked to capture state and to return a mocked PrintWriter object.  Stub for Employee Model was created to return “success” if eid exists or “fail” if it does not. Refer to Appendix F. |
| Input: | caller = update\_emp\_rec  eid = 0  fname=1  lanem=1  gen=1  dob=1  job=1  contact=1  email=1  addr=1  bname=1 |
| Expected Output | Response is redirect to “"error.jsp?msg=fail” page. |

|  |  |
| --- | --- |
| Test ID: | subSystem\_Get\_Profile\_Rainy03 |
| Purpose: | To verify that an employee profile cannot be updated for an eid that does not exists. |
| Test Setup: | HTTPServletRequest was mocked to return invalid eid value.  HTTPServletRequest was mocked to return valid caller, fname, lname, gen, dob, job, contact, e-mail, addr, accno, bname values.  HTTPServletResponse was mocked to capture state and to return a mocked PrintWriter object.  Stub for Employee Model was created to return “success” if eid exists or “fail” if it does not. Refer to Appendix F. |
| Input: | caller = update\_emp\_rec  eid = 122  fname=1  lanem=1  gen=1  dob=1  job=1  contact=1  email=1  addr=1  bname=1 |
| Expected Output | Response is redirect to “"error.jsp?msg=fail” page. |

|  |  |
| --- | --- |
| Test ID: | subSystem\_empregistration\_Sunny01 |
| Purpose: | To verify that an employee with an existing employeeID is able to register |
| Test Setup: | HTTPServletRequest was mocked to return valid caller, eid, uid, pwd, s1, a1, s2, a2, s3, and a3 values.  HTTPServletResponse was mocked to capture state.  Stub for Security\_Question model was created to return “success” if eid exists or “fail” if it does not. Refer to Appendix F. |
| Input: | caller = registration  eid = 1  uid=emp1  pwd=pass1  s1=Favorite Color  a1=blue  s2=First PEt Name  a2=sparky  s3=hackers  a3=2018-06-02 |
| Expected Output | Response redirects to “"emplogin.jsp” page. |

|  |  |
| --- | --- |
| Test ID: | subSystem\_empregistration\_Sunny02 |
| Purpose: | To verify that an employee with an existing employeeID is able to register |
| Test Setup: | HTTPServletRequest was mocked to return valid caller. eid, uid, pwd, s1, a1, s2, a2, s3, and a3 values.  HTTPServletResponse was mocked to capture state.  Stub for Security\_Question model was created to return “success” if eid exists or “fail” if it does not. Refer to Appendix F. |
| Input: | caller = registration  eid = 7  uid=emp1  pwd=pass1  s1=Favorite Color  a1=blue  s2=First PEt Name  a2=sparky  s3=hackers  a3=2018-06-02 |
| Expected Output | Response redirects to “"emplogin.jsp” page. |

|  |  |
| --- | --- |
| Test ID: | subSystem\_empregistration\_Sunny03 |
| Purpose: | To verify that an employee with an existing employeeID is able to register |
| Test Setup: | HTTPServletRequest was mocked to return valid caller. eid, uid, pwd, s1, a1, s2, a2, s3, and a3 values.  HTTPServletResponse was mocked to capture state.  Stub for Security\_Question model was created to return “success” if eid exists or “fail” if it does not. Refer to Appendix F. |
| Input: | caller = registration  eid = 10  uid=emp1  pwd=pass1  s1=Favorite Color  a1=blue  s2=First PEt Name  a2=sparky  s3=hackers  a3=2018-06-02 |
| Expected Output | Response redirects to “"emplogin.jsp” page. |

|  |  |
| --- | --- |
| Test ID: | subSystem\_empregistration\_Rainy01 |
| Purpose: | To verify that an employee without an existing employeeID is not able to register. |
| Test Setup: | HTTPServletRequest was mocked to return valid caller.  HTTPServletRequest was mocked to return invalid uid.  HTTPServletResponse was mocked to capture state.  Stub for Security\_Question model was created to return “success” if eid exists or “fail” if it does not. Refer to Appendix F. |
| Input: | caller = registration  eid = -100  uid=emp1  pwd=pass1  s1=Favorite Color  a1=blue  s2=First PEt Name  a2=sparky  s3=hackers  a3=2018-06-02 |
| Expected Output | Response redirects to “"error.jsp?msg=fail” page. |

|  |  |
| --- | --- |
| Test ID: | subSystem\_empregistration\_Rainy02 |
| Purpose: | To verify that an employee without an existing employeeID is not able to register. |
| Test Setup: | HTTPServletRequest was mocked to return valid caller.  HTTPServletRequest was mocked to return invalid uid.  HTTPServletResponse was mocked to capture state.  Stub for Security\_Question model was created to return “success” if eid exists or “fail” if it does not. Refer to Appendix F. |
| Input: | caller = registration  eid = 0  uid=emp1  pwd=pass1  s1=Favorite Color  a1=blue  s2=First PEt Name  a2=sparky  s3=hackers  a3=2018-06-02 |
| Expected Output | Response redirects to “"error.jsp?msg=fail” page. |

|  |  |
| --- | --- |
| Test ID: | subSystem\_empregistration\_Rainy03 |
| Purpose: | To verify that an employee without an existing employeeID is not able to register. |
| Test Setup: | HTTPServletRequest was mocked to return valid caller.  HTTPServletRequest was mocked to return invalid uid.  HTTPServletResponse was mocked to capture state.  Stub for Security\_Question model was created to return “success” if eid exists or “fail” if it does not. Refer to Appendix F. |
| Input: | caller = registration  eid = 99  uid=emp1  pwd=pass1  s1=Favorite Color  a1=blue  s2=First PEt Name  a2=sparky  s3=hackers  a3=2018-06-02 |
| Expected Output | Response redirects to “"error.jsp?msg=fail” page. |

|  |  |
| --- | --- |
| Test ID: | subSystem\_Search\_Employee\_Sunny01 |
| Purpose: | To verify that an existing employee is able to submit a timesheet for an employer to review. |
| Test Setup: | HTTPServletRequest was mocked to return valid empid value.  HTTPServletResponse was mocked to capture state and to return a mocked PrintWriter object.  TimeSheet object was mocked to contain valid empid values. Refer to Appendix F. |
| Input: | eid = 1 |
| Expected Output | PrinrWriter displays window alert “timesheet submitted”. |

|  |  |
| --- | --- |
| Test ID: | subSystem\_Search\_Employee\_Sunny02 |
| Purpose: | To verify that an existing employee is able to submit a timesheet for an employer to review. |
| Test Setup: | HTTPServletRequest was mocked to return valid empid value.  HTTPServletResponse was mocked to capture state and to return a mocked PrintWriter object.  TimeSheet object was mocked to contain valid empid values. Refer to Appendix F. |
| Input: | eid = 4 |
| Expected Output | PrinrWriter displays window alert “timesheet submitted”. |

|  |  |
| --- | --- |
| Test ID: | subSystem\_Search\_Employee\_Sunny03 |
| Purpose: | To verify that an existing employee is able to submit a timesheet for an employer to review. |
| Test Setup: | HTTPServletRequest was mocked to return valid empid value.  HTTPServletResponse was mocked to capture state and to return a mocked PrintWriter object.  TimeSheet object was mocked to contain valid empid values. Refer to Appendix F. |
| Input: | eid = 10 |
| Expected Output | PrintWriter displays window alert “timesheet submitted”. |

|  |  |
| --- | --- |
| Test ID: | subSystem\_Search\_Employee\_Rainy01 |
| Purpose: | To Verify that an employee without timesheet is not shown the “timesheet submitted” message. |
| Test Setup: | HTTPServletRequest was mocked to return valid empid value.  HTTPServletResponse was mocked to capture state and to return a mocked PrintWriter object.  TimeSheet object was mocked to contain valid empid and timesheet values. Refer to Appendix F. |
| Input: | eid =-10 |
| Expected Output | PrintWriter does not displays window alert “timesheet submitted”. |

|  |  |
| --- | --- |
| Test ID: | subSystem\_Search\_Employee\_Rainy02 |
| Purpose: | To Verify that an employee without timesheet is not shown the “timesheet submitted” message. |
| Test Setup: | HTTPServletRequest was mocked to return valid empid value.  HTTPServletResponse was mocked to capture state and to return a mocked PrintWriter object.  TimeSheet object was mocked to contain valid empid and timesheet values. Refer to Appendix F. |
| Input: | eid = 0 |
| Expected Output | PrintWriter does not displays window alert “timesheet submitted”. |

|  |  |
| --- | --- |
| Test ID: | subSystem\_Search\_Employee\_Rainy03 |
| Purpose: | To Verify that an employee without timesheet is not shown the “timesheet submitted” message. |
| Test Setup: | HTTPServletRequest was mocked to return valid empid value.  HTTPServletResponse was mocked to capture state and to return a mocked PrintWriter object.  TimeSheet object was mocked to contain valid empid and timesheet values. Refer to Appendix F. |
| Input: | eid = 11 |
| Expected Output | PrintWriter does not displays window alert “timesheet submitted”. |

|  |  |
| --- | --- |
| Test ID: | subSystem\_forgotpasswordcontroller\_Sunny01 |
| Purpose: | To Verify that an employee that has registered is able to retrieve their password by providing eid, uid, and answers to 3 security questions |
| Test Setup: | HTTPServletRequest was mocked to return valid caller, eid, uid, s1, a1, s2, a2, s3, and a3 values.  HTTPServletResponse was mocked to capture state.  Employee object was mocked to contain valid authentication values for existing users. Refer to Appendix F. |
| Input: | caller = forgotpassword  eid = 1  uid=epm1  s1=Favorite Color?  a1=blue  s2=First PEt Name  a2=sparky  s3=Favorite movie  a3=hackers |
| Expected Output | Password “pass1” is displayed on new page. |

|  |  |
| --- | --- |
| Test ID: | subSystem\_forgotpasswordcontroller\_Sunny02 |
| Purpose: | To Verify that an employee that has registered is able to retrieve their password by providing eid, uid, and answers to 3 security questions |
| Test Setup: | HTTPServletRequest was mocked to return valid caller, eid, uid, s1, a1, s2, a2, s3, and a3 values.  HTTPServletResponse was mocked to capture state.  Employee object was mocked to contain valid authentication values for existing users. Refer to Appendix F. |
| Input: | caller = forgotpassword  eid = 5  uid=epm5  s1=Favorite Color?  a1=blue  s2=First PEt Name  a2=sparky  s3=Favorite movie  a3=hackers |
| Expected Output | Password “pass5” is displayed on new page. |

|  |  |
| --- | --- |
| Test ID: | subSystem\_forgotpasswordcontroller\_Sunny03 |
| Purpose: | To Verify that an employee that has registered is able to retrieve their password by providing eid, uid, and answers to 3 security questions |
| Test Setup: | HTTPServletRequest was mocked to return valid caller, eid, uid, s1, a1, s2, a2, s3, and a3 values.  HTTPServletResponse was mocked to capture state.  Employee object was mocked to contain valid authentication values for existing users. Refer to Appendix F. |
| Input: | caller = forgotpassword  eid = 10  uid=epm10  s1=Favorite Color?  a1=blue  s2=First PEt Name  a2=sparky  s3=Favorite movie  a3=hackers |
| Expected Output | Password “pass10” is displayed. |

|  |  |
| --- | --- |
| Test ID: | subSystem\_forgotpasswordcontroller\_Rainy01 |
| Purpose: | To Verify that an employee that has not registered or that does not provide the correct answers to the security fields is not given a password. |
| Test Setup: | HTTPServletRequest was mocked to return invalid uid.  HTTPServletRequest was mocked to return valid caller, eid, s1, a1, s2, a2, s3, and a3 values.  HTTPServletResponse was mocked to capture state and return a mocked PrintWriter object.  Employee object was mocked to contain valid authentication values for existing users. Refer to Appendix F. |
| Input: | caller = forgotpassword  eid = 10  uid=emp100  s1=Favorite Color?  a1=blue  s2=First PEt Name  a2=sparky  s3=Favorite movie  a3=hackers |
| Expected Output | Window Alert displays “User Details Not Found” |

|  |  |
| --- | --- |
| Test ID: | subSystem\_forgotpasswordcontroller\_Rainy02 |
| Purpose: | To Verify that an employee that has not registered or that does not provide the correct answers to the security fields is not given a password. |
| Test Setup: | HTTPServletRequest was mocked to return invalid eid.  HTTPServletRequest was mocked to return valid caller, uid, s1, a1, s2, a2, s3, and a3 values.  HTTPServletResponse was mocked to capture state and return a mocked PrintWriter object.  Employee object was mocked to contain valid authentication values for existing users. Refer to Appendix F. |
| Input: | caller = forgotpassword  eid = 100  uid=emp1  s1=Favorite Color?  a1=blue  s2=First PEt Name  a2=sparky  s3=Favorite movie  a3=hackers |
| Expected Output | Window Alert displays “User Details Not Found” |

|  |  |
| --- | --- |
| Test ID: | subSystem\_forgotpasswordcontroller\_Rainy03 |
| Purpose: | To Verify that an employee that has not registered or that does not provide the correct answers to the security fields is not given a password. |
| Test Setup: | HTTPServletRequest was mocked to return invalid eid, and uid.  HTTPServletRequest was mocked to return valid caller, s1, a1, s2, a2, s3, and a3 values.  HTTPServletResponse was mocked to capture state and return a mocked PrintWriter object.  Employee object was mocked to contain valid authentication values for existing users. Refer to Appendix F. |
| Input: | caller = forgotpassword  eid = <empty string>  uid= <empty string>  s1=Favorite Color?  a1=blue  s2=First PEt Name  a2=sparky  s3=Favorite movie  a3=hackers |
| Expected Output | Window Alert displays “User Details Not Found” |

|  |  |
| --- | --- |
| Test ID: | subSystem\_Paycheck\_Generator\_Sunny01() |
| Purpose: | To Verify that modifications to the Paycheck are completed |
| Test Setup: | HTTPServletRequest was mocked to return success. With normal and extra values.    HTTPServletResponse was mocked to capture state and to return a printer write object. |
| Input: | caller = paymode  normal = “15”  extra = “10” |
| Expected Output | Response is redirect to an “window. alert ('Pay mode added'); window. location. replace('paymode.jsp’)” page |

|  |  |
| --- | --- |
| Test ID: | subSystem\_Paycheck\_Generator\_Sunny02() |
| Purpose: | To Verify that modifications to the Paycheck are completed |
| Test Setup: | HTTPServletRequest was mocked to return success. With normal and extra values.    HTTPServletResponse was mocked to capture state and to return a printer write object. |
| Input: | caller = paymode  normal = “1”  extra = “4999” |
| Expected Output | Response is redirect to an “window. alert ('Pay mode added'); window. location. replace('paymode.jsp’)” page |

|  |  |
| --- | --- |
| Test ID: | subSystem\_Paycheck\_Generator\_Sunny03() |
| Purpose: | To Verify that modifications to the Paycheck are completed |
| Test Setup: | HTTPServletRequest was mocked to return success. With normal and extra values.    HTTPServletResponse was mocked to capture state and to return a printer write object. |
| Input: | caller = “paymode”  normal = “152”  extra = “89” |
| Expected Output | Response is redirect to an “window. alert(‘Pay mode added'); window. location. replace('paymode.jsp’)” page |

|  |  |
| --- | --- |
| Test ID: | subSystem\_Paycheck\_Generator\_Rainy01() |
| Purpose: | To Verify that wrong modifications to the Paycheck are no completed. |
| Test Setup: | HTTPServletRequest was mocked to return success. With normal and extra values.    HTTPServletResponse was mocked to capture state and to return a printer write object. |
| Input: | caller = paymode  normal = “-1”  extra = “10000” |
| Expected Output | Response is redirect to "error.jsp?msg=fail"” page |

|  |  |
| --- | --- |
| Test ID: | subSystem\_Paycheck\_Generator\_Rainy02() |
| Purpose: | To Verify that wrong modifications to the Paycheck are no completed. |
| Test Setup: | HTTPServletRequest was mocked to return success. With normal and extra values.    HTTPServletResponse was mocked to capture state and to return a printer write object. |
| Input: | caller = paymode  normal = “8000”  extra = “300” |
| Expected Output | Response is redirect to "error.jsp?msg=fail"” page |

|  |  |
| --- | --- |
| Test ID: | subSystem\_Paycheck\_Generator\_Rainy03() |
| Purpose: | To Verify that wrong modifications to the Paycheck are no complete. |
| Test Setup: | HTTPServletRequest was mocked to return success. With normal and extra values.    HTTPServletResponse was mocked to capture state and to return a printer write object. |
| Input: | caller = paymode  normal = “52”  extra = “10000” |
| Expected Output | Response is redirect to "error.jsp?msg=fail"” page |

|  |  |
| --- | --- |
| Test ID: | subSystem\_Authentication\_Sunny01() |
| Purpose: | To Verify that updated password operation is done with proper parameters (Security Question). |
| Test Setup: | HTTPServletRequest was mocked to return success when updated password with proper questions and answers. Method only checked for passwords.    HTTPServletResponse was mocked to capture state and to return a printer write object. |
| Input: | caller = "changepassword"  s1 = Favorite Color  a1= blue  s2 =First PET name  a2= sparky  s3 = hackers  a3= us  oldpass= “day”  newpass= “yo”  cpass= “yo” |
| Expected Output | Response is redirect to an “window.alert('Password changed');window.location.replace('employeehome.jsp')” page |

|  |  |
| --- | --- |
| Test ID: | subSystem\_Authentication\_Sunny02() |
| Purpose: | To Verify that updated password operation is done with proper parameters (Security Question). |
| Test Setup: | HTTPServletRequest was mocked to return success when updated password with proper questions and answers. Method only checked for passwords.    HTTPServletResponse was mocked to capture state and to return a printer write object. |
| Input: | caller = "changepassword"  s1 = Favorite Color  a1= blue  s2 =First PET name  a2= sparky  s3 = hackers  a3= us  oldpass= “night”  newpass= “me”  cpass= “me” |
| Expected Output | Response is redirect to an “window.alert('Password changed');window.location.replace('employeehome.jsp')” page |

|  |  |
| --- | --- |
| Test ID: | subSystem\_Authentication\_Sunny03() |
| Purpose: | To Verify that updated password operation is done with proper parameters (Security Question). |
| Test Setup: | HTTPServletRequest was mocked to return success when updated password with proper questions and answers. Method only checked for passwords.    HTTPServletResponse was mocked to capture state and to return a printer write object. |
| Input: | caller = "changepassword"  s1 = Favorite Color  a1= blue  s2 =First PET name  a2= sparky  s3 = hackers  a3= us  oldpass= “all”  newpass= “yes”  cpass= “yes” |
| Expected Output | Response is redirect to an “window.alert('Password changed');window.location.replace('employeehome.jsp')” page |

|  |  |
| --- | --- |
| Test ID: | subSystem\_Authentication\_Rainy01() |
| Purpose: | To Verify that updated password operation is not allowed done without proper parameters (Security Question). |
| Test Setup: | HTTPServletRequest was mocked to return success when updated password with proper questions and answers. Method only checked for passwords.    HTTPServletResponse was mocked to capture state and to return a printer write object. |
| Input: | caller = "changepassword"  s1 = Favorite Color  a1= blue  s2 =First PET name  a2= sparky  s3 = hackers  a3= us  oldpass= “all”  newpass= “no”  cpass= “yes” |
| Expected Output | Response is redirect to an “window.alert('Passwords not matched');window.location.replace('changepassword.jsp')” page |

|  |  |
| --- | --- |
| Test ID: | subSystem\_Authentication\_Rainy02() |
| Purpose: | To Verify that updated password operation is not allowed done without proper parameters (Security Questions). |
| Test Setup: | HTTPServletRequest was mocked to return success when updated password with proper questions and answers. Method only checked for passwords.    HTTPServletResponse was mocked to capture state and to return a printer write object. |
| Input: | caller = "changepassword"  s1 = Favorite Color  a1= blue  s2 =First PET name  a2= meow(expected sparky)  s3 = hackers  a3= us  oldpass= “all”  newpass= “yes”  cpass= “yes” |
| Expected Output | Response is redirect to an “window.alert('Passwords not matched');window.location.replace('changepassword.jsp')” page |

|  |  |
| --- | --- |
| Test ID: | subSystem\_Authentication\_Rainy03() |
| Purpose: | To Verify that updated password operation is not allowed done without proper parameters (Security Questions). |
| Test Setup: | HTTPServletRequest was mocked to return success when updated password with proper questions and answers. Method only checked for passwords.    HTTPServletResponse was mocked to capture state and to return a printer write object. |
| Input: | caller = "changepassword"  s1 = Favorite Color  a1= blue  s2 =First PET name  a2= meow (expected sparky)  s3 = hackers  a3= us  oldpass= “all”  newpass= “no”  cpass= “yes” |
| Expected Output | Response is redirect to an “window.alert('Passwords not matched');window.location.replace('changepassword.jsp')” page |

|  |  |
| --- | --- |
| Test ID: | subSystem\_Forgot\_Password\_Sunny01() |
| Purpose: | To Verify that get password operation is done with proper parameters (Securities Questions). |
| Test Setup: | HTTPServletRequest was mocked to return success when get password with proper questions and answers.    HTTPServletResponse was mocked to capture state and send a redirect. |
| Input: | caller = "changepassword"  s1 = Favorite Color  a1= blue  s2 =First PET name  a2= sparky  s3 =hackers  a3= us |
| Expected Output | Response is redirect to an “"error.jsp?msg=Your password is="” page |

|  |  |
| --- | --- |
| Test ID: | subSystem\_Forgot\_Password\_Sunny02() |
| Purpose: | To Verify that get password operation is done with proper parameters (Securities Questions). |
| Test Setup: | HTTPServletRequest was mocked to return success when get password with proper questions and answers.    HTTPServletResponse was mocked to capture state and send a redirect. |
| Input: | caller = "changepassword"  s1 = Favorite Color  a1= green  s2 =First PET name  a2= meow  s3 = hackers  a3= them |
| Expected Output | Response is redirect to an “"error.jsp?msg=Your password is="” page |

|  |  |
| --- | --- |
| Test ID: | subSystem\_Forgot\_Password\_Sunny03() |
| Purpose: | To Verify that get password operation is done with proper parameters (Securities Questions). |
| Test Setup: | HTTPServletRequest was mocked to return success when get password with proper questions and answers.    HTTPServletResponse was mocked to capture state and send a redirect. |
| Input: | caller = "changepassword"  s1 = Favorite Color  a1= yellow  s2 =First PET name  a2= shy  s3 = hackers  a3= we |
| Expected Output | Response is redirect to an “"error.jsp?msg=Your password is="” page |

|  |  |
| --- | --- |
| Test ID: | subSystem\_Forgot\_Password\_Rainy01() |
| Purpose: | To Verify that get password operation is done with proper parameters (Security Questions). |
| Test Setup: | HTTPServletRequest was mocked to return success when get password with proper questions and answers.    HTTPServletResponse was mocked to capture state and to return a printer write object. |
| Input: | caller = "changepassword"  s1 = Favorite Color  a1= yellow (expected green)  s2 =First PET name  a2= shy (expected sparky)  s3 = hackers  a3= we (expected them) |
| Expected Output | Response is redirect to an “"window.alert('user details not found');window.location.replace('emplogin.jsp')” page |

|  |  |
| --- | --- |
| Test ID: | subSystem\_Forgot\_Password\_Rainy02() |
| Purpose: | To Verify that get password operation is done with proper parameters (Security Questions). |
| Test Setup: | HTTPServletRequest was mocked to return success when get password with proper questions and answers.    HTTPServletResponse was mocked to capture state and to return a printer write object. |
| Input: | caller = "changepassword"  s1 = Favorite Color  a1= yellow  s2 =First PET name  a2= meow (expected sparky)  s3 = hackers  a3= them |
| Expected Output | Response is redirect to an “"window.alert('user details not found');window.location.replace('emplogin.jsp')” page |

|  |  |
| --- | --- |
| Test ID: | subSystem\_Forgot\_Password\_Rainy03() |
| Purpose: | To Verify that get password operation is done with proper parameters (Security Questions). |
| Test Setup: | HTTPServletRequest was mocked to return success when get password with proper questions and answers.    HTTPServletResponse was mocked to capture state and to return a printer write object. |
| Input: | caller = "changepassword"  s1 = Favorite Color  a1= green  s2 =First PET name  a2= fox (expected sparky)  s3 = hackers  a3= us (expected them) |
| Expected Output | Response is redirect to an “"window.alert('user details not found');window.location.replace('emplogin.jsp')” page |

### 4.2 Test Results

|  |  |
| --- | --- |
| TestID | Result |
| subSystem\_emplrlogincontroller\_Sunny01 | Fail |
| subSystem\_emplrlogincontroller\_Sunny02 | Fail |
| subSystem\_emplrlogincontroller\_Sunny03 | Fail |
| subSystem\_emplrlogincontroller\_Rainy01 | Pass |
| subSystem\_emplrlogincontroller\_Rainy02 | Pass |
| subSystem\_emplrlogincontroller\_Rainy03 | Pass |
| subSystem\_login\_Sunny01 | Pass |
| subSystem\_login\_Sunny02 | Pass |
| subSystem\_login\_Sunny03 | Pass |
| subSystem\_Edit\_Timesheet\_Control\_Sunny01 | Pass |
| subSystem\_Edit\_Timesheet\_Control\_Sunny02 | Pass |
| subSystem\_Edit\_Timesheet\_Control\_Sunny03 | Pass |
| subSystem\_Edit\_Timesheet\_Control\_Rainy01 | Pass |
| subSystem\_Edit\_Timesheet\_Control\_Rainy01 | Pass |
| subSystem\_Edit\_Timesheet\_Control\_Rainy01 | Pass |
| subSystem\_deleteempcontroller\_Sunny01 | Pass |
| subSystem\_deleteempcontroller\_Sunny02 | Pass |
| subSystem\_deleteempcontroller\_Sunny03 | Pass |
| subSystem\_deleteempcontroller\_Sunny01 | Pass |
| subSystem\_deleteempcontroller\_Sunny02 | Pass |
| subSystem\_deleteempcontroller\_Sunny03 | Pass |
| subSystem\_deleteempcontroller\_Rainy01 | Pass |
| subSystem\_deleteempcontroller\_Rainy02 | Pass |
| subSystem\_deleteempcontroller\_Rainy03 | Pass |
| subSystem\_Get\_Profile\_Sunny01 | Pass |
| subSystem\_Get\_Profile\_Sunny02 | Pass |
| subSystem\_Get\_Profile\_Sunny03 | Pass |
| subSystem\_Get\_Profile\_Rainy01 | Pass |
| subSystem\_Get\_Profile\_Rainy02 | Pass |
| subSystem\_Get\_Profile\_Rainy03 | Pass |
| subSystem\_empregistration\_Sunny01 | Pass |
| subSystem\_empregistration\_Sunny02 | Pass |
| subSystem\_empregistration\_Sunny03 | Pass |
| subSystem\_empregistration\_Rainy01 | Pass |
| subSystem\_empregistration\_Rainy02 | Pass |
| subSystem\_empregistration\_Rainy03 | Pass |
| subSystem\_Search\_Employee\_Sunny01 | Pass |
| subSystem\_Search\_Employee\_Sunny02 | Pass |
| subSystem\_Search\_Employee\_Sunny03 | Pass |
| subSystem\_Search\_Employee\_Rainy01 | Fail |
| subSystem\_Search\_Employee\_Rainy02 | Fail |
| subSystem\_Search\_Employee\_Rainy03 | Fail |
| subSystem\_forgotpasswordcontroller\_Sunny01 | Pass |
| subSystem\_forgotpasswordcontroller\_Sunny02 | Pass |
| subSystem\_forgotpasswordcontroller\_Sunny03 | Pass |
| subSystem\_forgotpasswordcontroller\_Rainy01 | Pass |
| subSystem\_forgotpasswordcontroller\_Rainy02 | Pass |
| subSystem\_forgotpasswordcontroller\_Rainy03 | Pass |
| subSystem\_Paycheck\_Generator\_Sunny01() | Pass |
| subSystem\_Paycheck\_Generator\_Sunny02() | Pass |
| subSystem\_Paycheck\_Generator\_Sunny03() | Pass |
| subSystem\_Paycheck\_Generator\_Rainy01() | Pass |
| subSystem\_Paycheck\_Generator\_Rainy02() | Pass |
| subSystem\_Paycheck\_Generator\_Rainy03() | Pass |
| subSystem\_Authentication\_Sunny01() | Pass |
| subSystem\_Authentication\_Sunny02() | Pass |
| subSystem\_Authentication\_Sunny03() | Pass |
| subSystem\_Authentication\_Rainy01() | Pass |
| subSystem\_Authentication\_Rainy02() | Pass |
| subSystem\_Authentication\_Rainy03() | Pass |
| subSystem\_Forgot\_Password\_Sunny01() | Pass |
| subSystem\_Forgot\_Password\_Sunny02() | Pass |
| subSystem\_Forgot\_Password\_Sunny03() | Pass |
| subSystem\_Forgot\_Password\_Rainy01() | Pass |
| subSystem\_Forgot\_Password\_Rainy02() | Pass |
| subSystem\_Forgot\_Password\_Rainy03() | Pass |

## Chapter 5: System Testing

### 5.1.1 Test Identification and Objective (Summary)

The case list below shows the ID and objectives of each test cases performed for the PMS. Refer to Appendix G for the state the database was reinitialized to prior to each System test.

|  |  |
| --- | --- |
| ID | Objective |
| System\_emplrlogin\_Sunny01 | To verify that emplrlogin.jsp redirects to emplrhome.jsp page when using valid credentials. |
| System\_emplrlogin\_Sunny02 |
| System\_emplrlogin\_Sunny03 |
| System\_emplrlogin\_Rainy01 | To verify that emplrlogin.jsp redirects to Error page when using invalid credentials. |
| System\_emplrlogin\_Rainy02 |
| System\_emplrlogin\_Rainy03 |
| System\_login\_Sunny01 | To verify that emplogin.jsp redirects to employeehome.jsp page when using valid credentials. |
| System\_login\_Sunny02 |
| System\_login\_Sunny03 |
| System\_login\_Rainy01 | To verify that emplogin.jsp redirects to “error.jsp” page when using invalid credentials. |
| System\_login\_Rainy02 |
| System\_login\_Rainy03 |
| System\_Edit\_Timesheet\_Sunny01 | To verify that an employer logged-in is shown a successful message if the timesheet was updated. |
| System\_Edit\_Timesheet\_Sunny02 |
| System\_Edit\_Timesheet\_Sunny03 |
| System\_Edit\_Timesheet\_Rainy01 | To verify that a value in the incorrect time-format causes a failure to update the employee time sheet. |
| System\_Edit\_Timesheet\_Rainy02 |
| System\_Edit\_Timesheet\_Rainy03 |
| System\_deleteemp\_Sunny01 | To verify that if an employer passes in a valid empID, the record is deleted. |
| System\_deleteemp\_Sunny02 |
| System\_deleteemp\_Sunny03 |
| System\_deleteemp\_Rainy01 | To verify that an employer who passes an invalid empid receives an error message. |
| System\_deleteemp\_Rainy02 |
| System\_deleteemp\_Rainy03 |
| System\_Get\_Profile\_Sunny01 | To verify that an employer can update the profile for an existing eid |
| System\_Get\_Profile\_Sunny02 |
| System\_Get\_Profile\_Sunny03 |
| System\_Get\_Profile\_Rainy01 | To verify that an employer receives an error when trying to update a profile for an unregistered eid. |
| System\_Get\_Profile\_Rainy02 |
| System\_Get\_Profile\_Rainy03 |
| System\_empregistration\_Sunny01 | To verify that an employee with an existing employeeID is able to register |
| System\_empregistration\_Sunny02 |
| System\_empregistration\_Sunny03 |
| System\_empregistration\_Rainy01 | To verify that an employee without an existing employeeID is not able to register. |
| System\_empregistration\_Rainy02 |
| System\_empregistration\_Rainy03 |
| System\_Search\_Employee\_Sunny01 | To verify that an employer can search for a registered empid. |
| System\_Search\_Employee\_Sunny02 |
| System\_Search\_Employee\_Sunny03 |
| System\_Search\_Employee\_Rainy01 | To Verify that an employer receives an error when searching for an unregistered empid. |
| System\_Search\_Employee\_Rainy02 |
| System\_Search\_Employee\_Rainy03 |
| System\_forgotpassword\_Sunny01 | To Verify that an employee that has registered is able to retrieve their password by providing eid, uid, and answers to 3 security questions |
| System\_forgotpassword\_Sunny02 |
| System\_forgotpassword\_Sunny03 |
| System\_forgotpassword\_Rainy01 | To Verify that an employee that has not registered or that does not provide the correct answers to the security fields is not given a password. |
| System\_forgotpassword\_Rainy02 |
| System\_forgotpassword\_Rainy03 |
| System\_Paycheck\_Generator\_Sunny01 | To verify that a logged in employee can view their corresponding pay slips. |
| System\_Paycheck\_Generator\_Sunny02 |
| System\_Paycheck\_Generator\_Sunny03 |
| System\_Paycheck\_Generator\_Rainy01 | To Verify that an employee not logged in is not shown anyone’s payslip. |
| System\_Paycheck\_Generator\_Rainy02 |
| System\_Paycheck\_Generator\_Rainy03 |
| System\_Approve\_TS\_Sunny01 | To verify a logged in employer can approve a time sheet of their choice. |
| System\_Approve\_TS\_Sunny02 |
| System\_Approve\_TS\_Sunny03 |
| System\_Approve\_TS\_Rainy01 | To verify a logged in employer can approve none, multiple, or all timesheets without error. |
| System\_Approve\_TS\_Rainy02 |
| System\_Approve\_TS\_Rainy03 |
| System\_Add\_Emp\_Sunny01 | To verify an employer can add an employee with a valid, not in use, employee id. |
| System\_Add\_Emp\_Sunny03 |
| System\_Add\_Emp\_Sunny03 |
| System\_Add\_Emp\_Rainy01 | To verify an employer receives an error when trying to add an employee with an employee ID already in use. |
| System\_Add\_Emp\_Rainy02 |
| System\_Add\_Emp\_Rainy03 |

### 5.1.2. Test Criteria and Procedures

**Test Criteria**

Each of implemented use cases results in a redirection to an error or the requested page. Therefore, the test criterion chosen is that tests will exercise the two possible paths. Valid values should redirect to the requested page. Invalid values should redirect to an error page. The results of the tests are determined by examining the URL, and in some cases the visible text, of the displayed web page.

Input values are all considered to be of type String by the Rational Functional Tester. As such, input values for tests are partitioned into two equivalence classes. Values that would be in the database and values that would not. In addition, whenever an integer is a String, we treat the String as an integer and perform boundary analysis on the valid partition.

**Procedures**

The System was setup using tomcat Server, and a mySQL database, both running on localhost. The tests were run using Rational Functional Tester v9.2.0 (RFT) and Google Chrome v67.0.3396.87 with the “Functional Tester for Google Chrome” extension enabled. Respective data pools for Sunny and Rainy day scenarios were also created and used to execute the test cases. The input of the test cases shows which values were used for that test.

When a test case refers to “valid” inputs, it refers to values that the application would find in the PMS database. Appendix G contains the initial state of the database that each system test was executed from.

A test is considered to pass if we can verify that the expected web page is displayed given some parameters. This represents the least intrusive way of verifying results as we need only to check the URL or text returned by the system, rather than the contents of the database.

### 5.1.3 Test Cases

|  |  |
| --- | --- |
| Test ID: | System\_emplrlogin\_Sunny01 |
| Purpose: | To verify that emplrlogin.jsp redirects to emplrhome.jsp page when using valid credentials. |
| Test Setup: | The database is reinitialized to initial state.  User Opens application in browser on home page.  User navigates to emplrlogin.jsp |
| Input: | Click on text\_user text box.  Enter “galaxy” for username.  Click on text\_pwd text box.  Enter “deathstar” for password.  Click on button\_submit button. |
| Expected Output | The browser loads the “emplrhome.jsp” page |

|  |  |
| --- | --- |
| Test ID: | System\_emplrlogin\_Sunny02 |
| Purpose: | To verify that emplrlogin.jsp redirects to emplrhome.jsp page when using valid credentials. |
| Test Setup: | The database is reinitialized to initial state.  User Opens application in browser on home page.  User navigates to emplrlogin.jsp |
| Input: | Click on text\_user text box.  Enter “galaxy1” for username.  Click on text\_pwd text box.  Enter “deathstar1” for password.  Click on button\_submit button. |
| Expected Output | The browser loads the “emplrhome.jsp” page |

|  |  |
| --- | --- |
| Test ID: | System\_emplrlogin\_Sunny03 |
| Purpose: | To verify that emplrlogin.jsp redirects to emplrhome.jsp page when using valid credentials. |
| Test Setup: | The database is reinitialized to initial state.  User Opens application in browser on home page.  User navigates to emplrlogin.jsp |
| Input: | Click on text\_user text box.  Enter “galaxy2” for username.  Click on text\_pwd text box.  Enter “deathstar2” for password.  Click on button\_submit button. |
| Expected Output | The browser loads the “emplrhome.jsp” page |

|  |  |
| --- | --- |
| Test ID: | System\_emplrlogin\_Rainy01 |
| Purpose: | To verify that emplrlogin.jsp redirects to Error page when using invalid credentials. |
| Test Setup: | The database is reinitialized to initial state.  User Opens application in browser on home page.  User navigates to emplrlogin.jsp |
| Input: | Click on text\_user text box.  Enter “galaxy” for username.  Click on text\_pwd text box.  Enter “wrongpassword” for password.  Click on button\_submit button. |
| Expected Output | The browser loads the “error.jsp” page. |

|  |  |
| --- | --- |
| Test ID: | System\_emplrlogin\_Rainy02 |
| Purpose: | To verify that emplrlogin.jsp redirects to Error page when using invalid credentials. |
| Test Setup: | The database is reinitialized to initial state.  User Opens application in browser on home page.  User navigates to emplrlogin.jsp |
| Input: | Click on text\_user text box.  Enter “wrongusername” for username.  Click on text\_pwd text box.  Enter “deathstar” for password.  Click on button\_submit button. |
| Expected Output | The browser loads the “error.jsp” page. |

|  |  |
| --- | --- |
| Test ID: | System\_emplrlogin\_Rainy03 |
| Purpose: | To verify that emplrlogin.jsp redirects to Error page when using invalid credentials. |
| Test Setup: | The database is reinitialized to initial state.  User Opens application in browser on home page.  User navigates to emplrlogin.jsp |
| Input: | Click on text\_user text box.  Enter “ ” , a blank space, for username.  Click on text\_pwd text box.  Enter “ ”, a blank space, for password.  Click on button\_submit button. |
| Expected Output | The browser loads the “error.jsp” page. |

|  |  |
| --- | --- |
| Test ID: | System\_login\_Sunny01 |
| Purpose: | To verify that emplogin.jsp redirects to employeehome.jsp page when using valid credentials. |
| Test Setup: | The database is reinitialized to initial state.  User Opens application in browser on home page.  User navigates to emplogin.jsp |
| Input: | Click on text\_eid text box  Enter “1” for Employee ID  Click on text\_user text box.  Enter “emp1” for username.  Click on text\_pwd text box.  Enter “pass1” for password.  Click on button\_login button. |
| Expected Output | The browser loads the “employeehome” page |

|  |  |
| --- | --- |
| Test ID: | System\_login\_Sunny02 |
| Purpose: | To verify that emplogin.jsp redirects to employeehome.jsp page when using valid credentials. |
| Test Setup: | The database is reinitialized to initial state.  User Opens application in browser on home page.  User navigates to emplogin.jsp |
| Input: | Click on text\_eid text box  Enter “5” for Employee ID  Click on text\_user text box.  Enter “emp5” for username.  Click on text\_pwd text box.  Enter “pass5” for password.  Click on button\_login button. |
| Expected Output | The browser loads the “employeehome” page |

|  |  |
| --- | --- |
| Test ID: | System\_login\_Sunny03 |
| Purpose: | To verify that emplogin.jsp redirects to employeehome.jsp page when using valid credentials. |
| Test Setup: | The database is reinitialized to initial state.  User Opens application in browser on home page.  User navigates to emplogin.jsp |
| Input: | Click on text\_eid text box  Enter “10” for Employee ID  Click on text\_user text box.  Enter “emp10” for username.  Click on text\_pwd text box.  Enter “pass10” for password.  Click on button\_login button. |
| Expected Output | The browser loads the “employeehome” page |

|  |  |
| --- | --- |
| Test ID: | System\_login\_Rainy01 |
| Purpose: | To verify that emplogin.jsp redirects to “error.jsp” page when using invalid credentials. |
| Test Setup: | The database is reinitialized to initial state.  User Opens application in browser on home page.  User navigates to emplogin.jsp |
| Input: | Click on text\_eid text box  Enter “192” for Employee ID  Click on text\_user text box.  Enter “emp1” for username.  Click on text\_pwd text box.  Enter “pass1” for password.  Click on button\_login button. |
| Expected Output | The browser loads the “error.jsp” page. |

|  |  |
| --- | --- |
| Test ID: | System\_login\_Rainy02 |
| Purpose: | To verify that emplogin.jsp redirects to “error.jsp” page when using invalid credentials. |
| Test Setup: | The database is reinitialized to initial state.  User Opens application in browser on home page.  User navigates to emplogin.jsp |
| Input: | Click on text\_eid text box  Enter “1” for Employee ID  Click on text\_user text box.  Enter “emp10” for username.  Click on text\_pwd text box.  Enter “pass1” for password.  Click on button\_login button. |
| Expected Output | The browser loads the “error.jsp” page. |

|  |  |
| --- | --- |
| Test ID: | System\_login\_Rainy03 |
| Purpose: | To verify that emplogin.jsp redirects to “error.jsp” page when using invalid credentials. |
| Test Setup: | The database is reinitialized to initial state.  User Opens application in browser on home page.  User navigates to emplogin.jsp |
| Input: | Click on text\_eid text box  Enter “1” for Employee ID  Click on text\_user text box.  Enter “emp1” for username.  Click on text\_pwd text box.  Enter “pass10” for password.  Click on button\_login button. |
| Expected Output | The browser loads the “error.jsp” page. |

|  |  |
| --- | --- |
| Test ID: | System\_Edit\_Timesheet\_Sunny01 |
| Purpose: | To verify that an employer logged-in is shown a successful message if the timesheet was updated. |
| Test Setup: | The database is reinitialized to initial state.  User Opens application in browser on home page.  User navigates to emplrlogin.jsp  User enters “galaxy” for username and “deathstar” for password  Browser loads emplrhome.jsp  User navigates to “view\_timesheet.jsp”  User selects “All” from dropdown and submits |
| Input: | Select text in text\_intime for the first row and replace with “11:00:00”  Select text in text\_lunchout for the first row and replace with “14:00:00”  Select text in text\_lunchin for the first row and replace with “15:00:00”  Select text in text\_outime for the first row and replace with “17:00:00”  Select text in text\_thours for the first row and replace with “5.00”  Click Update timesheet button. |
| Expected Output | The browser loads “view\_timesheet.jsp?eid=All&submit=Get+Details” |

|  |  |
| --- | --- |
| Test ID: | System\_Edit\_Timesheet\_Sunny02 |
| Purpose: | To verify that an employer logged-in is shown a successful message if the timesheet was updated. |
| Test Setup: | The database is reinitialized to initial state.  User Opens application in browser on home page.  User navigates to emplrlogin.jsp  User enters “galaxy” for username and “deathstar” for password  Browser loads emplrhome.jsp  User navigates to “view\_timesheet.jsp”  User selects “All” from dropdown and submits |
| Input: | Select text in text\_intime for the first row and replace with “00:00:00”  Select text in text\_lunchout for the first row and replace with “14:00:00”  Select text in text\_lunchin for the first row and replace with “15:00:00”  Select text in text\_outime for the first row and replace with “24:00:00”  Select text in text\_thours for the first row and replace with “24.00”  Click Update timesheet button. |
| Expected Output | The browser loads “view\_timesheet.jsp?eid=All&submit=Get+Details” |

|  |  |
| --- | --- |
| Test ID: | System\_Edit\_Timesheet\_Sunny03 |
| Purpose: | To verify that an employer logged-in is shown a successful message if the timesheet was updated. |
| Test Setup: | The database is reinitialized to initial state.  User Opens application in browser on home page.  User navigates to emplrlogin.jsp  User enters “galaxy” for username and “deathstar” for password  Browser loads emplrhome.jsp  User navigates to “view\_timesheet.jsp”  User selects “All” from dropdown and submits |
| Input: | Select text in text\_intime for the first row and replace with “00:00:00”  Select text in text\_lunchout for the first row and replace with “14:00:00”  Select text in text\_lunchin for the first row and replace with “14:00:00”  Select text in text\_outime for the first row and replace with “14:00:00”  Select text in text\_thours for the first row and replace with “14.00”  Click Update timesheet button. |
| Expected Output | The browser loads “view\_timesheet.jsp?eid=All&submit=Get+Details” |

|  |  |
| --- | --- |
| Test ID: | System\_Edit\_Timesheet\_Rainy01 |
| Purpose: | To verify that a value in the incorrect time-format causes a failure to update the employee time sheet. |
| Test Setup: | The database is reinitialized to initial state.  User Opens application in browser on home page.  User navigates to emplrlogin.jsp  User enters “galaxy” for username and “deathstar” for password  Browser loads emplrhome.jsp  User navigates to “view\_timesheet.jsp”  User selects “All” from dropdown and submits |
| Input: | Select text in text\_intime for the first row and inputkey “{BKSP}”  Select text in text\_lunchout for the first row and replace with “14:00:00”  Select text in text\_lunchin for the first row and replace with “14:00:00”  Select text in text\_outime for the first row and replace with “14:00:00”  Select text in text\_thours for the first row and replace with “14.00”  Click Update timesheet button. |
| Expected Output | The browser loads “error.jsp” |

|  |  |
| --- | --- |
| Test ID: | System\_Edit\_Timesheet\_Rainy02 |
| Purpose: | To verify that a value in the incorrect time-format causes a failure to update the employee time sheet. |
| Test Setup: | The database is reinitialized to initial state.  User Opens application in browser on home page.  User navigates to emplrlogin.jsp  User enters “galaxy” for username and “deathstar” for password  Browser loads emplrhome.jsp  User navigates to “view\_timesheet.jsp”  User selects “All” from dropdown and submits |
| Input: | Select text in text\_intime for the first row and replace with “00:00:00”  Select text in text\_lunchout for the first row and replace with “14:00:00”  Select text in text\_lunchin for the first row and replace with “ ”  Select text in text\_outime for the first row and replace with “24:00:00”  Select text in text\_thours for the first row and replace with “24.00”  Click Update timesheet button. |
| Expected Output | The browser loads “error.jsp” |

|  |  |
| --- | --- |
| Test ID: | System\_Edit\_Timesheet\_Rainy03 |
| Purpose: | To verify that a value in the incorrect time-format causes a failure to update the employee time sheet. |
| Test Setup: | The database is reinitialized to initial state.  User Opens application in browser on home page.  User navigates to emplrlogin.jsp  User enters “galaxy” for username and “deathstar” for password  Browser loads emplrhome.jsp  User navigates to “view\_timesheet.jsp”  User selects “All” from dropdown and submits |
| Input: | Select text in text\_intime for the first row and replace with “test”  Select text in text\_lunchout for the first row and replace with “14:00:00”  Select text in text\_lunchin for the first row and replace with “14:00:00”  Select text in text\_outime for the first row and replace with “14:00:00”  Select text in text\_thours for the first row and replace with “14.00”  Click Update timesheet button. |
| Expected Output | The browser loads “error.jsp” |

|  |  |
| --- | --- |
| Test ID: | System\_deleteemp\_Sunny01 |
| Purpose: | To verify that if an employer passes in a valid empID, the record is deleted. |
| Test Setup: | The database is reinitialized to initial state.  User Opens application in browser on home page.  User navigates to emplrlogin.jsp  User enters “galaxy” for username and “deathstar” for password  Browser loads emplrhome.jsp  User navigates to “delete\_emp.jsp” |
| Input: | Click on Employee ID text box  Enter “1”  Click “Delete” |
| Expected Output | The browser loads “delete\_emp.jsp” |

|  |  |
| --- | --- |
| Test ID: | System\_deleteemp\_Sunny02 |
| Purpose: | To verify that if an employer passes in a valid empID, the record is deleted. |
| Test Setup: | The database is reinitialized to initial state.  User Opens application in browser on home page.  User navigates to emplrlogin.jsp  User enters “galaxy” for username and “deathstar” for password  Browser loads emplrhome.jsp  User navigates to “delete\_emp.jsp” |
| Input: | Click on Employee ID text box  Enter “5”  Click “Delete” |
| Expected Output | The browser loads “delete\_emp.jsp” |

|  |  |
| --- | --- |
| Test ID: | System\_deleteemp\_Sunny03 |
| Purpose: | To verify that if an employer passes in a valid empID, the record is deleted. |
| Test Setup: | The database is reinitialized to initial state.  User Opens application in browser on home page.  User navigates to emplrlogin.jsp  User enters “galaxy” for username and “deathstar” for password  Browser loads emplrhome.jsp  User navigates to “delete\_emp.jsp” |
| Input: | Click on Employee ID text box  Enter “10”  Click “Delete” |
| Expected Output | The browser loads “delete\_emp.jsp” |

|  |  |
| --- | --- |
| Test ID: | System\_deleteemp\_Rainy01 |
| Purpose: | To verify that an employer who passes an invalid empid receives an error page. |
| Test Setup: | The database is reinitialized to initial state.  User Opens application in browser on home page.  User navigates to emplrlogin.jsp  User enters “galaxy” for username and “deathstar” for password  Browser loads emplrhome.jsp  User navigates to “delete\_emp.jsp” |
| Input: | Click on Employee ID text box  Enter “-10”  Click “Delete” |
| Expected Output | The browser loads “error.jsp” |

|  |  |
| --- | --- |
| Test ID: | System\_deleteemp\_Rainy02 |
| Purpose: | To verify that an employer who passes an invalid empid receives an error page. |
| Test Setup: | The database is reinitialized to initial state.  User Opens application in browser on home page.  User navigates to emplrlogin.jsp  User enters “galaxy” for username and “deathstar” for password  Browser loads emplrhome.jsp  User navigates to “delete\_emp.jsp” |
| Input: | Click on Employee ID text box  Enter “ ”  Click “Delete” |
| Expected Output | The browser loads “error.jsp” |

|  |  |
| --- | --- |
| Test ID: | System\_deleteemp\_Rainy03 |
| Purpose: | To verify that an employer who passes an invalid empid receives an error page. |
| Test Setup: | The database is reinitialized to initial state.  User Opens application in browser on home page.  User navigates to emplrlogin.jsp  User enters “galaxy” for username and “deathstar” for password  Browser loads emplrhome.jsp  User navigates to “delete\_emp.jsp” |
| Input: | Click on Employee ID text box  Enter “9999”  Click “Delete” |
| Expected Output | The browser loads “error.jsp” |

|  |  |
| --- | --- |
| Test ID: | System\_Get\_Profile\_Sunny01 |
| Purpose: | To verify that an employer can update the profile for a valid eid |
| Test Setup: | The database is reinitialized to initial state.  User Opens application in browser on home page.  User navigates to emplrlogin.jsp  User enters “galaxy” for username and “deathstar” for password  Browser loads emplrhome.jsp  User navigates to “update\_emp.jsp” |
| Input: | Click on Employee ID text box and Enter “1”  Click “Search”  Select text in Contact text box and replace with “2020”  Click “Update” |
| Expected Output | The browser loads “update\_emp\_rec.jsp” |

|  |  |
| --- | --- |
| Test ID: | System\_Get\_Profile\_Sunny02 |
| Purpose: | To verify that an employer can update the profile for a valid eid |
| Test Setup: | The database is reinitialized to initial state.  User Opens application in browser on home page.  User navigates to emplrlogin.jsp  User enters “galaxy” for username and “deathstar” for password  Browser loads emplrhome.jsp  User navigates to “update\_emp.jsp” |
| Input: | Click on Employee ID text box and Enter “5”  Click “Search”  Select text in Contact text box and replace with “600”  Click “Update” |
| Expected Output | The browser loads “update\_emp\_rec.jsp” |

|  |  |
| --- | --- |
| Test ID: | System\_Get\_Profile\_Sunny03 |
| Purpose: | To verify that an employer can update the profile for a valid eid |
| Test Setup: | The database is reinitialized to initial state.  User Opens application in browser on home page.  User navigates to emplrlogin.jsp  User enters “galaxy” for username and “deathstar” for password  Browser loads emplrhome.jsp  User navigates to “update\_emp.jsp” |
| Input: | Click on Employee ID text box and Enter “10”  Click “Search”  Select text in Contact text box and replace with “300”  Click “Update” |
| Expected Output | The browser loads “update\_emp\_rec.jsp” |

|  |  |
| --- | --- |
| Test ID: | System\_Get\_Profile\_Rainy01 |
| Purpose: | To verify that an employer receives an error when trying to update a profile for an unregistered eid. |
| Test Setup: | The database is reinitialized to initial state.  User Opens application in browser on home page.  User navigates to emplrlogin.jsp  User enters “galaxy” for username and “deathstar” for password  Browser loads emplrhome.jsp  User navigates to “update\_emp.jsp” |
| Input: | Click on Employee ID text box and Enter “-10”  Click “Search” |
| Expected Output | The browser loads “update\_emp\_rec.jsp” with visible text “Empid not found.” |

|  |  |
| --- | --- |
| Test ID: | System\_Get\_Profile\_Rainy02 |
| Purpose: | To verify that an employer receives an error when trying to update a profile for an unregistered eid. |
| Test Setup: | The database is reinitialized to initial state.  User Opens application in browser on home page.  User navigates to emplrlogin.jsp  User enters “galaxy” for username and “deathstar” for password  Browser loads emplrhome.jsp  User navigates to “update\_emp.jsp” |
| Input: | Click on Employee ID text box and Enter “ ”  Click “Search” |
| Expected Output | The browser loads “update\_emp\_rec.jsp” with visible text “Empid not found.” |

|  |  |
| --- | --- |
| Test ID: | System\_Get\_Profile\_Rainy03 |
| Purpose: | To verify that an employer receives an error when trying to update a profile for an unregistered eid. |
| Test Setup: | The database is reinitialized to initial state.  User Opens application in browser on home page.  User navigates to emplrlogin.jsp  User enters “galaxy” for username and “deathstar” for password  Browser loads emplrhome.jsp  User navigates to “update\_emp.jsp” |
| Input: | Click on Employee ID text box and Enter “9999”  Click “Search” |
| Expected Output | The browser loads “update\_emp\_rec.jsp” with visible text “Empid not found.” |

|  |  |
| --- | --- |
| Test ID: | System\_empregistration\_Sunny01 |
| Purpose: | To verify that an employee with an existing employeeID is able to register |
| Test Setup: | The database is reinitialized to initial state.  User Opens application in browser on home page.  User navigates to “registration.jsp” |
| Input: | Click at employee ID textbox and enter “101”  Click at username textbox and enter “emp101”  Click at password textbox and enter “pass101”  Click Security Question 1 dropdown and select “Favorite Color?”  Click at answer1 textbox and enter “blue”  Click Security Question 1 dropdown and select “First Pet name?”  Click at answer1 textbox and enter “sparky”  Click Security Question 1 dropdown and select “Favorite movie?”  Click at answer1 textbox and enter “hackers”  Click Register |
| Expected Output | The browser loads “emplogin.jsp” |

|  |  |
| --- | --- |
| Test ID: | System\_empregistration\_Sunny02 |
| Purpose: | To verify that an employee with an existing employeeID is able to register |
| Test Setup: | The database is reinitialized to initial state.  User Opens application in browser on home page.  User navigates to “registration.jsp” |
| Input: | Click at employee ID textbox and enter “102”  Click at username textbox and enter “emp102”  Click at password textbox and enter “pass102”  Click Security Question 1 dropdown and select “Favorite Color?”  Click at answer1 textbox and enter “red”  Click Security Question 1 dropdown and select “First Pet name?”  Click at answer1 textbox and enter “sparky”  Click Security Question 1 dropdown and select “Favorite movie?”  Click at answer1 textbox and enter “hackers the movie”  Click Register |
| Expected Output | The browser loads “emplogin.jsp” |

|  |  |
| --- | --- |
| Test ID: | System\_empregistration\_Sunny03 |
| Purpose: | To verify that an employee with an existing employeeID is able to register |
| Test Setup: | The database is reinitialized to initial state.  User Opens application in browser on home page.  User navigates to “registration.jsp” |
| Input: | Click at employee ID textbox and enter “110”  Click at username textbox and enter “emp110”  Click at password textbox and enter “pass110”  Click Security Question 1 dropdown and select “Favorite Color?”  Click at answer1 textbox and enter “yellow”  Click Security Question 1 dropdown and select “First Pet name?”  Click at answer1 textbox and enter “sparky”  Click Security Question 1 dropdown and select “Favorite movie?”  Click at answer1 textbox and enter “Find Nemo.”  Click Register |
| Expected Output | The browser loads “emplogin.jsp” |

|  |  |
| --- | --- |
| Test ID: | System\_empregistration\_Rainy01 |
| Purpose: | To verify that an employee without an existing employeeID is not able to register. |
| Test Setup: | The database is reinitialized to initial state.  User Opens application in browser on home page.  User navigates to “registration.jsp” |
| Input: | Click at employee ID textbox and enter “-20”  Click at username textbox and enter “emp101”  Click at password textbox and enter “pass101”  Click Security Question 1 dropdown and select “Favorite Color?”  Click at answer1 textbox and enter “blue”  Click Security Question 1 dropdown and select “First Pet name?”  Click at answer1 textbox and enter “sparky”  Click Security Question 1 dropdown and select “Favorite movie?”  Click at answer1 textbox and enter “hackers”  Click Register |
| Expected Output | The browser loads “error.jsp” |

|  |  |
| --- | --- |
| Test ID: | System\_empregistration\_Rainy02 |
| Purpose: | To verify that an employee without an existing employeeID is not able to register. |
| Test Setup: | The database is reinitialized to initial state.  User Opens application in browser on home page.  User navigates to “registration.jsp” |
| Input: | Click at employee ID textbox and enter “0”  Click at username textbox and enter “emp102”  Click at password textbox and enter “pass102”  Click Security Question 1 dropdown and select “Favorite Color?”  Click at answer1 textbox and enter “red”  Click Security Question 1 dropdown and select “First Pet name?”  Click at answer1 textbox and enter “sparky”  Click Security Question 1 dropdown and select “Favorite movie?”  Click at answer1 textbox and enter “hackers the movie”  Click Register |
| Expected Output | The browser loads “error.jsp” |

|  |  |
| --- | --- |
| Test ID: | System\_empregistration\_Rainy03 |
| Purpose: | To verify that an employee without an existing employeeID is not able to register. |
| Test Setup: | The database is reinitialized to initial state.  User Opens application in browser on home page.  User navigates to “registration.jsp” |
| Input: | Click at employee ID textbox and enter “200000000000000000000000000”  Click at username textbox and enter “emp110”  Click at password textbox and enter “pass110”  Click Security Question 1 dropdown and select “Favorite Color?”  Click at answer1 textbox and enter “yellow”  Click Security Question 1 dropdown and select “First Pet name?”  Click at answer1 textbox and enter “sparky”  Click Security Question 1 dropdown and select “Favorite movie?”  Click at answer1 textbox and enter “Find Nemo.”  Click Register |
| Expected Output | The browser loads “error.jsp” |

|  |  |
| --- | --- |
| Test ID: | System\_Search\_Employee\_Sunny01 |
| Purpose: | To verify that an employer can search for a registered empid |
| Test Setup: | The database is reinitialized to initial state.  User Opens application in browser on home page.  User navigates to emplrlogin.jsp  User enters “galaxy” for username and “deathstar” for password and submits  Browser loads emplrhome.jsp  User navigates to “search\_emp.jsp” |
| Input: | Click on Employee ID text box and Enter “1”  Click “Search” |
| Expected Output | The browser loads “show\_emp\_rec.jsp” with “Employee Details” as visible text |

|  |  |
| --- | --- |
| Test ID: | System\_Search\_Employee\_Sunny02 |
| Purpose: | To verify that an employer can search for a registered empid |
| Test Setup: | The database is reinitialized to initial state.  User Opens application in browser on home page.  User navigates to emplrlogin.jsp  User enters “galaxy” for username and “deathstar” for password and submits  Browser loads emplrhome.jsp  User navigates to “search\_emp.jsp” |
| Input: | Click on Employee ID text box and Enter “5”  Click “Search” |
| Expected Output | The browser loads “show\_emp\_rec.jsp” with “Employee Details” as visible text |

|  |  |
| --- | --- |
| Test ID: | System\_Search\_Employee\_Sunny03 |
| Purpose: | To verify that an employer can search for a registered empid |
| Test Setup: | The database is reinitialized to initial state.  User Opens application in browser on home page.  User navigates to emplrlogin.jsp  User enters “galaxy” for username and “deathstar” for password and submits  Browser loads emplrhome.jsp  User navigates to “search\_emp.jsp” |
| Input: | Click on Employee ID text box and Enter “10”  Click “Search” |
| Expected Output | The browser loads “show\_emp\_rec.jsp” with “Employee Details” as visible text |

|  |  |
| --- | --- |
| Test ID: | System\_Search\_Employee\_Rainy01 |
| Purpose: | To Verify that an employer receives an error when searching for an unregistered empid. |
| Test Setup: | The database is reinitialized to initial state.  User Opens application in browser on home page.  User navigates to emplrlogin.jsp  User enters “galaxy” for username and “deathstar” for password and submits  Browser loads emplrhome.jsp  User navigates to “search\_emp.jsp” |
| Input: | Click on Employee ID text box and Enter “-10”  Click “Search” |
| Expected Output | The browser loads “show\_emp\_rec.jsp” with “Empid not found.” as visible text |

|  |  |
| --- | --- |
| Test ID: | System\_Search\_Employee\_Rainy02 |
| Purpose: | To Verify that an employer receives an error when searching for an unregistered empid. |
| Test Setup: | The database is reinitialized to initial state.  User Opens application in browser on home page.  User navigates to emplrlogin.jsp  User enters “galaxy” for username and “deathstar” for password and submits  Browser loads emplrhome.jsp  User navigates to “search\_emp.jsp” |
| Input: | Click on Employee ID text box and Enter “ ”  Click “Search” |
| Expected Output | The browser loads “show\_emp\_rec.jsp” with “Empid not found.” as visible text |

|  |  |
| --- | --- |
| Test ID: | System\_Search\_Employee\_Rainy03 |
| Purpose: | To Verify that an employer receives an error when searching for an unregistered empid. |
| Test Setup: | The database is reinitialized to initial state.  User Opens application in browser on home page.  User navigates to emplrlogin.jsp  User enters “galaxy” for username and “deathstar” for password and submits  Browser loads emplrhome.jsp  User navigates to “search\_emp.jsp” |
| Input: | Click on Employee ID text box and Enter “9999”  Click “Search” |
| Expected Output | The browser loads “show\_emp\_rec.jsp” with “Empid not found.” as visible text |

|  |  |
| --- | --- |
| Test ID: | System\_forgotpassword\_Sunny01 |
| Purpose: | To Verify that an employee that has registered is able to retrieve their password by providing eid, uid, and answers to 3 security questions |
| Test Setup: | The database is reinitialized to initial state.  User Opens application in browser on home page.  User navigates to emplogin.jsp  User clicks on “Forgot Password?”  Browser loads forgotpassword.jsp |
| Input: | Click at employee ID textbox and enter “1”  Click at username textbox and enter “emp1”  Click Security Question 1 dropdown and select “Favorite Color?”  Click at answer1 textbox and enter “blue”  Click Security Question 1 dropdown and select “First Pet name?”  Click at answer1 textbox and enter “sparky”  Click Security Question 1 dropdown and select “Favorite movie?”  Click at answer1 textbox and enter “hackers”  Click “Get Password” |
| Expected Output | The browser loads “error.jsp” with “Your password=” in the visible text. |

|  |  |
| --- | --- |
| Test ID: | System\_forgotpassword\_Sunny02 |
| Purpose: | To Verify that an employee that has registered is able to retrieve their password by providing eid, uid, and answers to 3 security questions |
| Test Setup: | The database is reinitialized to initial state.  User Opens application in browser on home page.  User navigates to emplogin.jsp  User clicks on “Forgot Password?”  Browser loads forgotpassword.jsp |
| Input: | Click at employee ID textbox and enter “10”  Click at username textbox and enter “emp10”  Click Security Question 1 dropdown and select “Favorite Color?”  Click at answer1 textbox and enter “blue”  Click Security Question 1 dropdown and select “First Pet name?”  Click at answer1 textbox and enter “sparky”  Click Security Question 1 dropdown and select “Favorite movie?”  Click at answer1 textbox and enter “hackers”  Click “Get Password” |
| Expected Output | The browser loads “error.jsp” with “Your password=” in the visible text. |

|  |  |
| --- | --- |
| Test ID: | System\_forgotpassword\_Sunny03 |
| Purpose: | To Verify that an employee that has registered is able to retrieve their password by providing eid, uid, and answers to 3 security questions |
| Test Setup: | The database is reinitialized to initial state.  User Opens application in browser on home page.  User navigates to emplogin.jsp  User clicks on “Forgot Password?”  Browser loads forgotpassword.jsp |
| Input: | Click at employee ID textbox and enter “2”  Click at username textbox and enter “emp2”  Click Security Question 1 dropdown and select “Favorite Color?”  Click at answer1 textbox and enter “blue”  Click Security Question 1 dropdown and select “First Pet name?”  Click at answer1 textbox and enter “sparky”  Click Security Question 1 dropdown and select “Favorite movie?”  Click at answer1 textbox and enter “hackers”  Click “Get Password” |
| Expected Output | The browser loads “error.jsp” with “Your password=” in the visible text. |

|  |  |
| --- | --- |
| Test ID: | System\_forgotpassword\_Rainy01 |
| Purpose: | To Verify that an employee that has not registered or that does not provide the correct answers to the security fields is not given a password. |
| Test Setup: | The database is reinitialized to initial state.  User Opens application in browser on home page.  User navigates to emplogin.jsp  User clicks on “Forgot Password?”  Browser loads forgotpassword.jsp |
| Input: | Click at employee ID textbox and enter “-1000”  Click at username textbox and enter “emp1”  Click Security Question 1 dropdown and select “Favorite Color?”  Click at answer1 textbox and enter “blue”  Click Security Question 1 dropdown and select “First Pet name?”  Click at answer1 textbox and enter “sparky”  Click Security Question 1 dropdown and select “Favorite movie?”  Click at answer1 textbox and enter “hackers”  Click “Get Password” |
| Expected Output | The browser loads “forgotpassword.jsp” again. |

|  |  |
| --- | --- |
| Test ID: | System\_forgotpassword\_Rainy02 |
| Purpose: | To Verify that an employee that has not registered or that does not provide the correct answers to the security fields is not given a password. |
| Test Setup: | The database is reinitialized to initial state.  User Opens application in browser on home page.  User navigates to emplogin.jsp  User clicks on “Forgot Password?”  Browser loads forgotpassword.jsp |
| Input: | Click at employee ID textbox and enter “10”  Click at username textbox and enter “emp1”  Click Security Question 1 dropdown and select “Favorite Color?”  Click at answer1 textbox and enter “blue”  Click Security Question 1 dropdown and select “First Pet name?”  Click at answer1 textbox and enter “sparky”  Click Security Question 1 dropdown and select “Favorite movie?”  Click at answer1 textbox and enter “hackers”  Click “Get Password” |
| Expected Output | The browser loads “forgotpassword.jsp” again. |

|  |  |
| --- | --- |
| Test ID: | System\_forgotpassword\_Rainy03 |
| Purpose: | To Verify that an employee that has not registered or that does not provide the correct answers to the security fields is not given a password. |
| Test Setup: | The database is reinitialized to initial state.  User Opens application in browser on home page.  User navigates to emplogin.jsp  User clicks on “Forgot Password?”  Browser loads forgotpassword.jsp |
| Input: | Click at employee ID textbox and enter “2”  Click at username textbox and enter “emp2”  Click Security Question 1 dropdown and select “Favorite Color?”  Click at answer1 textbox and enter “blue”  Click Security Question 1 dropdown and select “First Pet name?”  Click at answer1 textbox and enter “Turbo”  Click Security Question 1 dropdown and select “Favorite movie?”  Click at answer1 textbox and enter “hackers”  Click “Get Password” |
| Expected Output | The browser loads “forgotpassword.jsp” again. |

|  |  |
| --- | --- |
| Test ID: | System\_Paycheck\_Generator\_Sunny01 |
| Purpose: | To verify that a logged in employee can view their corresponding pay slips. |
| Test Setup: | The database is reinitialized to initial state.  User Opens application in browser on home page.  User navigates to emplogin.jsp |
| Input: | Click at employee ID textbox and enter “1”  Click at username textbox and enter “emp1”  Click at password and enter “pass1”  Click Login and browser loads employeehome.jsp  User navigates to “viewemppayslip.jsp” |
| Expected Output | The browser loads “viewemppayslip.jsp” with “Emp ID : 1” in the visible text |

|  |  |
| --- | --- |
| Test ID: | System\_Paycheck\_Generator\_Sunny02 |
| Purpose: | To verify that a logged in employee can view their corresponding pay slips. |
| Test Setup: | The database is reinitialized to initial state.  User Opens application in browser on home page.  User navigates to emplogin.jsp |
| Input: | Click at employee ID textbox and enter “5”  Click at username textbox and enter “emp5”  Click at password and enter “pass5”  Click Login and browser loads employeehome.jsp  User navigates to “viewemppayslip.jsp” |
| Expected Output | The browser loads “viewemppayslip.jsp” with “Emp ID : 5” in the visible text |

|  |  |
| --- | --- |
| Test ID: | System\_Paycheck\_Generator\_Sunny03 |
| Purpose: | To verify that a logged in employee can view their corresponding pay slips. |
| Test Setup: | The database is reinitialized to initial state.  User Opens application in browser on home page.  User navigates to emplogin.jsp |
| Input: | Click at employee ID textbox and enter “1”  Click at username textbox and enter “emp1”  Click at password and enter “pass1”  Click Login and browser loads employeehome.jsp  User navigates to “viewemppayslip.jsp” |
| Expected Output | The browser loads “viewemppayslip.jsp” with “Emp ID: 10” in the visible text |

|  |  |
| --- | --- |
| Test ID: | System\_Paycheck\_Generator\_Rainy01 |
| Purpose: | To Verify that an employee not logged in is not shown anyone’s payslip. |
| Test Setup: | The database is reinitialized to initial state.  User Opens application in browser on home page. |
| Input: | User navigates to “viewemppayslip.jsp” without logging in |
| Expected Output | The browser loads “viewemppayslip.jsp” without any Employee pay details |

|  |  |
| --- | --- |
| Test ID: | System\_Paycheck\_Generator\_Rainy02 |
| Purpose: | To Verify that an employee that has logged out is not shown anyone’s payslip. |
| Test Setup: | The database is reinitialized to initial state.  User Opens application in browser on home page. |
| Input: | User navigates to “emplogin.jsp”  Click at employee ID textbox and enter “1”  Click at username textbox and enter “emp1”  Click at password and enter “pass1”  Click Login and browser loads employeehome.jsp  Click Logout and browser loads index.jsp  User navigates to “viewemppayslip.jsp” |
| Expected Output | The browser loads “viewemppayslip.jsp” without any Employee pay details |

|  |  |
| --- | --- |
| Test ID: | System\_Paycheck\_Generator\_Rainy03 |
| Purpose: | To Verify that an employer that has logged in is not shown anyone’s employee payslip. |
| Test Setup: | The database is reinitialized to initial state.  User Opens application in browser on home page. |
| Input: | User navigates to “emplrlogin.jsp”  Click at username textbox and enter “galaxy”  Click at password and enter “deathstar”  Click Login and browser loads “emplrhome.jsp”  User navigates to URL “viewemppayslip.jsp” |
| Expected Output | The browser loads “viewemppayslip.jsp” without any Employee pay details |

|  |  |
| --- | --- |
| Test ID: | System\_Approve\_TS\_Rainy01 |
| Purpose: | To verify a logged in employer can approve no timesheet without error |
| Test Setup: | The database is reinitialized to initial state.  User Opens application in browser on home page.  User navigates to “emplrlogin.jsp”  Click at username textbox and enter “galaxy”  Click at password and enter “deathstar”  Click Login and browser loads “emplrhome.jsp” |
| Input: | User navigates to URL “approve\_ts.jsp”  User click Approve |
| Expected Output | The browser loads “approve\_ts.jsp” |

|  |  |
| --- | --- |
| Test ID: | System\_Approve\_TS\_Rainy02 |
| Purpose: | To verify a logged in employer can approve 5 time sheets without error |
| Test Setup: | The database is reinitialized to initial state.  User Opens application in browser on home page.  User navigates to “emplrlogin.jsp”  Click at username textbox and enter “galaxy”  Click at password and enter “deathstar”  Click Login and browser loads “emplrhome.jsp” |
| Input: | User navigates to URL “approve\_ts.jsp”  User clicks approval for the from the third to the eighth row  User click Approve |
| Expected Output | The browser loads “approve\_ts.jsp” |

|  |  |
| --- | --- |
| Test ID: | System\_Approve\_TS\_Rainy03 |
| Purpose: | To verify a logged in employer can approve all time sheets without error |
| Test Setup: | The database is reinitialized to initial state.  User Opens application in browser on home page.  User navigates to “emplrlogin.jsp”  Click at username textbox and enter “galaxy”  Click at password and enter “deathstar”  Click Login and browser loads “emplrhome.jsp” |
| Input: | User navigates to URL “approve\_ts.jsp”  User clicks approval for all rows  User click Approve |
| Expected Output | The browser loads “approve\_ts.jsp” |

|  |  |
| --- | --- |
| Test ID: | System\_Add\_Emp\_Sunny01 |
| Purpose: | To verify an employer can add an employee with a valid, not in use, employee id. |
| Test Setup: | The database is reinitialized to initial state.  User Opens application in browser on home page.  User navigates to “emplrlogin.jsp”  Click at username textbox and enter “galaxy”  Click at password and enter “deathstar”  Click Login and browser loads “emplrhome.jsp” |
| Input: | User navigates to URL “add\_emp.jsp”  Click at Employee ID textbox and enter “1001”  Click at first name textbox and enter “Mike”  Click at last name textbox and enter “Mke1001”  Click at D.O.B. textbox and enter “06252050”  Click at job textbox and enter “StormTroop”  Click at contact textbox and enter “895”  Click at email textbox and enter “Mike@1001.com”  Click at address textbox and enter “12345 SW 100”  Click at account number textbox and enter “12345”  Click at Bank name textbox and enter “Empire”  Click at “Add” |
| Expected Output | The browser loads “emplrhome.jsp” |

|  |  |
| --- | --- |
| Test ID: | System\_Add\_Emp\_Sunny02 |
| Purpose: | To verify an employer can add an employee with a valid, not in use, employee id. |
| Test Setup: | The database is reinitialized to initial state.  User Opens application in browser on home page.  User navigates to “emplrlogin.jsp”  Click at username textbox and enter “galaxy”  Click at password and enter “deathstar”  Click Login and browser loads “emplrhome.jsp” |
| Input: | User navigates to URL “add\_emp.jsp”  Click at Employee ID textbox and enter “1010”  Click at first name textbox and enter “Mike”  Click at last name textbox and enter “Mke1001”  Click at D.O.B. textbox and enter “06252050”  Click at job textbox and enter “StormTroop”  Click at contact textbox and enter “895”  Click at email textbox and enter “Mike@1001.com”  Click at address textbox and enter “12345 SW 100”  Click at account number textbox and enter “12345”  Click at Bank name textbox and enter “Empire”  Click at “Add” |
| Expected Output | The browser loads “emplrhome.jsp” |

|  |  |
| --- | --- |
| Test ID: | System\_Add\_Emp\_Sunny03 |
| Purpose: | To verify an employer can add an employee with a valid, not in use, employee id. |
| Test Setup: | The database is reinitialized to initial state.  User Opens application in browser on home page.  User navigates to “emplrlogin.jsp”  Click at username textbox and enter “galaxy”  Click at password and enter “deathstar”  Click Login and browser loads “emplrhome.jsp” |
| Input: | User navigates to URL “add\_emp.jsp”  Click at Employee ID textbox and enter “50”  Click at first name textbox and enter “Mike”  Click at last name textbox and enter “Mke1001”  Click at D.O.B. textbox and enter “06252050”  Click at job textbox and enter “StormTroop”  Click at contact textbox and enter “895”  Click at email textbox and enter “Mike@1001.com”  Click at address textbox and enter “12345 SW 100”  Click at account number textbox and enter “12345”  Click at Bank name textbox and enter “Empire”  Click at “Add” |
| Expected Output | The browser loads “emplrhome.jsp” |

|  |  |
| --- | --- |
| Test ID: | System\_Add\_Emp\_Rainy01 |
| Purpose: | To verify an employer receives an error when trying to add an employee with an employee ID already in use. |
| Test Setup: | The database is reinitialized to initial state.  User Opens application in browser on home page.  User navigates to “emplrlogin.jsp”  Click at username textbox and enter “galaxy”  Click at password and enter “deathstar”  Click Login and browser loads “emplrhome.jsp” |
| Input: | User navigates to URL “add\_emp.jsp”  Click at Employee ID textbox and enter “1”  Click at first name textbox and enter “Mike”  Click at last name textbox and enter “Mke1001”  Click at D.O.B. textbox and enter “06252050”  Click at job textbox and enter “StormTroop”  Click at contact textbox and enter “895”  Click at email textbox and enter “Mike@1001.com”  Click at address textbox and enter “12345 SW 100”  Click at account number textbox and enter “12345”  Click at Bank name textbox and enter “Empire”  Click at “Add” |
| Expected Output | The browser loads “error.jsp?msg=Employee%20Registration%20Failed” |

|  |  |
| --- | --- |
| Test ID: | System\_Add\_Emp\_Rainy02 |
| Purpose: | To verify an employer receives an error when trying to add an employee with an employee ID already in use. |
| Test Setup: | The database is reinitialized to initial state.  User Opens application in browser on home page.  User navigates to “emplrlogin.jsp”  Click at username textbox and enter “galaxy”  Click at password and enter “deathstar”  Click Login and browser loads “emplrhome.jsp” |
| Input: | User navigates to URL “add\_emp.jsp”  Click at Employee ID textbox and enter “5”  Click at first name textbox and enter “Mike”  Click at last name textbox and enter “Mke1001”  Click at D.O.B. textbox and enter “06252050”  Click at job textbox and enter “StormTroop”  Click at contact textbox and enter “895”  Click at email textbox and enter “Mike@1001.com”  Click at address textbox and enter “12345 SW 100”  Click at account number textbox and enter “12345”  Click at Bank name textbox and enter “Empire”  Click at “Add” |
| Expected Output | The browser loads “error.jsp?msg=Employee%20Registration%20Failed” |

.

|  |  |
| --- | --- |
| Test ID: | System\_Add\_Emp\_Rainy03 |
| Purpose: | To verify an employer receives an error when trying to add an employee with an employee ID already in use. |
| Test Setup: | The database is reinitialized to initial state.  User Opens application in browser on home page.  User navigates to “emplrlogin.jsp”  Click at username textbox and enter “galaxy”  Click at password and enter “deathstar”  Click Login and browser loads “emplrhome.jsp” |
| Input: | User navigates to URL “add\_emp.jsp”  Click at Employee ID textbox and enter “10”  Click at first name textbox and enter “Mike”  Click at last name textbox and enter “Mke1001”  Click at D.O.B. textbox and enter “06252050”  Click at job textbox and enter “StormTroop”  Click at contact textbox and enter “895”  Click at email textbox and enter “Mike@1001.com”  Click at address textbox and enter “12345 SW 100”  Click at account number textbox and enter “12345”  Click at Bank name textbox and enter “Empire”  Click at “Add” |
| Expected Output | The browser loads “error.jsp?msg=Employee%20Registration%20Failed” |

### 5.2 Test Results

|  |  |
| --- | --- |
| ID | Results |
| System\_emplrlogin\_Sunny01 | Pass |
| System\_emplrlogin\_Sunny02 | Pass |
| System\_emplrlogin\_Sunny03 | Pass |
| System\_emplrlogin\_Rainy01 | Pass |
| System\_emplrlogin\_Rainy02 | Pass |
| System\_emplrlogin\_Rainy03 | Pass |
| System\_login\_Sunny01 | Pass |
| System\_login\_Sunny02 | Pass |
| System\_login\_Sunny03 | Pass |
| System\_login\_Rainy01 | Pass |
| System\_login\_Rainy02 | Pass |
| System\_login\_Rainy03 | Pass |
| System\_Edit\_Timesheet\_Sunny01 | Pass |
| System\_Edit\_Timesheet\_Sunny02 | Pass |
| System\_Edit\_Timesheet\_Sunny03 | Pass |
| System\_Edit\_Timesheet\_Rainy01 | Pass |
| System\_Edit\_Timesheet\_Rainy02 | Pass |
| System\_Edit\_Timesheet\_Rainy03 | Pass |
| System\_deleteemp\_Sunny01 | Pass |
| System\_deleteemp\_Sunny02 | Pass |
| System\_deleteemp\_Sunny03 | Pass |
| System\_deleteemp\_Rainy01 | Pass |
| System\_deleteemp\_Rainy02 | Pass |
| System\_deleteemp\_Rainy03 | Pass |
| System\_Get\_Profile\_Sunny01 | Pass |
| System\_Get\_Profile\_Sunny02 | Pass |
| System\_Get\_Profile\_Sunny03 | Pass |
| System\_Get\_Profile\_Rainy01 | Pass |
| System\_Get\_Profile\_Rainy02 | Pass |
| System\_Get\_Profile\_Rainy03 | Pass |
| System\_empregistration\_Sunny01 | Pass |
| System\_empregistration\_Sunny02 | Pass |
| System\_empregistration\_Sunny03 | Pass |
| System\_empregistration\_Rainy01 | Pass |
| System\_empregistration\_Rainy02 | Pass |
| System\_empregistration\_Rainy03 | Pass |
| System\_Search\_Employee\_Sunny01 | Pass |
| System\_Search\_Employee\_Sunny02 | Pass |
| System\_Search\_Employee\_Sunny03 | Pass |
| System\_Search\_Employee\_Rainy01 | Pass |
| System\_Search\_Employee\_Rainy02 | Pass |
| System\_Search\_Employee\_Rainy03 | Pass |
| System\_forgotpassword\_Sunny01 | Pass |
| System\_forgotpassword\_Sunny02 | Pass |
| System\_forgotpassword\_Sunny03 | Pass |
| System\_forgotpassword\_Rainy01 | Pass |
| System\_forgotpassword\_Rainy02 | Pass |
| System\_forgotpassword\_Rainy03 | Pass |
| System\_Paycheck\_Generator\_Sunny01 | Pass |
| System\_Paycheck\_Generator\_Sunny02 | Pass |
| System\_Paycheck\_Generator\_Sunny03 | Pass |
| System\_Paycheck\_Generator\_Rainy01 | Pass |
| System\_Paycheck\_Generator\_Rainy02 | Failed |
| System\_Paycheck\_Generator\_Rainy03 | Pass |
| System\_Approve\_TS\_Sunny01 | Pass |
| System\_Approve\_TS\_Sunny02 | Pass |
| System\_Approve\_TS\_Sunny03 | Pass |
| System\_Approve\_TS\_Rainy01 | Pass |
| System\_Approve\_TS\_Rainy02 | Pass |
| System\_Approve\_TS\_Rainy03 | Pass |
| System\_Add\_Emp\_Sunny01 | Pass |
| System\_Add\_Emp\_Sunny02 | Pass |
| System\_Add\_Emp\_Sunny03 | Pass |
| System\_Add\_Emp\_Rainy01 | Pass |
| System\_Add\_Emp\_Rainy02 | Pass |
| System\_Add\_Emp\_Rainy03 | Pass |

## Chapter 6: Test Summary Report

The following tests cases did not produce the expected output and failed to meet test criteria. Observation and possible solutions are presented in juxtaposition.

### Unit Test Cases

|  |  |
| --- | --- |
| Test Case ID(s): | unitTest\_deleteempcontroller\_Rainy01  unitTest\_deleteempcontroller\_Rainy02  unitTest\_deleteempcontroller\_Rainy03 |
| Purpose: | To verify that the deleteempcontroller controller successfully lets the user delete an employee when the user enter a valid employee ID and to verify the controller successfully does not let the user delete an employee when the user does not enter a employee ID or when the user enter an invalid employee ID. |
| Observations: | deleteempcontroller controller was not able to find that specific employee ID in the database. |
| Proposed Solution: | include an assertEquals and verify method in the test case to make sure that the controller objects are mock and if the result is successful. |

### Subsystem Test Cases

|  |  |
| --- | --- |
| Test Case ID(s): | subSystem\_Search\_Employee\_Rainy01  subSystem\_Search\_Employee\_Rainy01  subSystem\_Search\_Employee\_Rainy01 |
| Purpose: | To Verify that an employee without timesheet is not shown the “timesheet submitted” message. |
| Observations: | The controller method name, search\_employee, is not representative of it’s function, which might cause confusion.  The controller always return a window that display “timesheet submitted” even when the input values are incorrect. |
| Proposed Solution: | Refactor the controller method name to represent it’s function (e.g. submit\_timesheet) and include logic that is able to confirm if the timesheet submitted was successful. If successful display “timesheet submitted” else display error message (e.g “error submitting timesheet). |

|  |  |
| --- | --- |
| Test Case ID(s): | subSystem\_emplrlogincontroller\_Sunny01  subSystem\_emplrlogincontroller\_Sunny02  subSystem\_emplrlogincontroller\_Sunny03  subSystem\_emplrlogincontroller\_Rainy01  subSystem\_emplrlogincontroller\_Rainy02  subSystem\_emplrlogincontroller\_Rainy03 |
| Purpose: | To verify that employer authentication controller lets user with the valid username, password combination into the system. |
| Observations: | While the controller does verify that the passed credentials are valid, the result action is to redirect to the employer home page which does not verify the session or credentials. This results in anyone knowing the URL to be granted access. |
| Proposed Solution: | Once the user is authenticated, a session object should be created and the employer home page should check that the session contains valid credentials. |

### System Test Cases

|  |  |
| --- | --- |
| Test Case ID(s): | System\_Paycheck\_Generator\_Rainy02 |
| Purpose: | To Verify that an employee that has logged out is not shown anyone’s payslip. |
| Observations: | After a user has logged out, if they manually navigate to “viewemppayslip.jsp” their timesheet still appears. |
| Proposed Solution: | Have “logout.jsp” clear HTTPsession session attributes when the user logs out, as “viewempayslip.jsp” uses those session variables to retrieve the corresponding payslips for the currently logged in employee. |

## Chapter 7: Risks and Contingencies

Potential risks that could arise during the testing process include:

1) Lack of experience with web applications, software engineering, databases, and testing tools resulting in the project not being finished on time.

2) Poor time management resulting in project not completed or finished with substandard quality.

3) Conflicting schedules and large geographical distance between members could be an impediment for team collaboration.

Contingency plans for potential risks include:

1) Verify early in the testing process that at least one team member has taken databases, software engineering, or familiar with database. If at least one team member does not have experience in the subjects mentioned above, then enough time should be scheduled for a person to learn the basics of the subject so it can be taught to the others. If no experience with any of the testing tools, team members should be assigned a tool to learn so they may teach the others.

2) Project tasks should be started early in the process to avoid rushing near the dateline. Having a weekly standup where tasks are discussed and assigned should provide a sense of urgency and responsibility to all members.

3) Team meetings should be planned for before or after lectures since all team members would be in the same location. This also has the added benefit of coinciding with the professor Clarke’s office hours. If this is not possible, then virtual meetings should be schedule in the evening or weekends where members can join in. Tools that facilitate remote learning like google drive, microsoft office online, etc should be leveraged.

## Chapter 8: Approvals

|  |  |  |
| --- | --- | --- |
| Approved By | Signature | On (Date) |
| Ciana | *Ciana Rogers* | 06/15/18 |
| Ivan | *Ivan Rojas* | 06/15/18 |
| Daykel | *Daykel Muro* | 06/15/18 |
| Dwayne | *Dwayne Thomas* | 06/15/18 |
| Juan Diaz | *Juan Diaz* | 06/15/18 |

## Chapter 9

These are the definition for the terms used in this document.

|  |  |
| --- | --- |
| ISC | International Science Consortium. |
| Junit | A framework used in unit testing for the Java programming language. |
| RFT | Rational Functional Tester (IBM). Software used for automated systems testing that that mimic the actions and assessments of a human tester. |
| FIU | Florida International University |
| JRE | Java Runtime Environment |
| Facade | A software design pattern that involves the refactoring of connections between subsystems so that only the facade is exposed. |
| Mockito | A mocking framework for unit tests written in Java. |
| PowerMockito | Extends Mockito functionality with several new features such as mocking static and private methods. |
| Project Professional | Tool used to keep track of team meetings and work. |
| Tomcat | Server used to host web-applications. |
| Mysql | Relational database system. |
| SRD | Software Requirement Document |
| Mocked | The action of creating an instance of object with specific values using mockito. |

## Chapter 10

### Appendix A

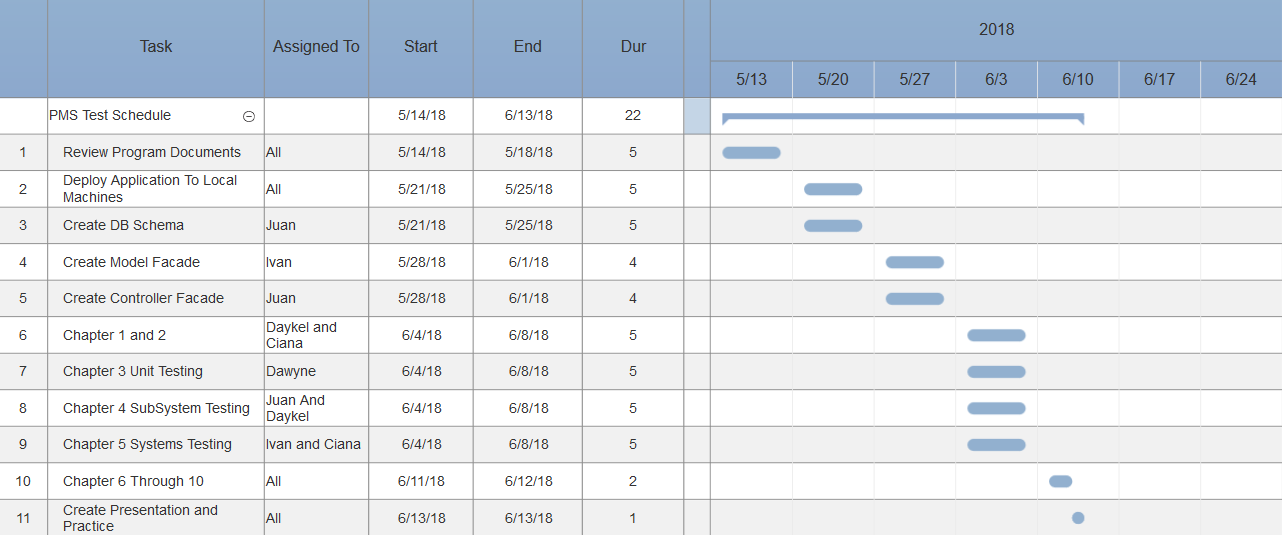
****

Figure 10.1.1: Test Schedule as a Gantt Chart

### Appendix B

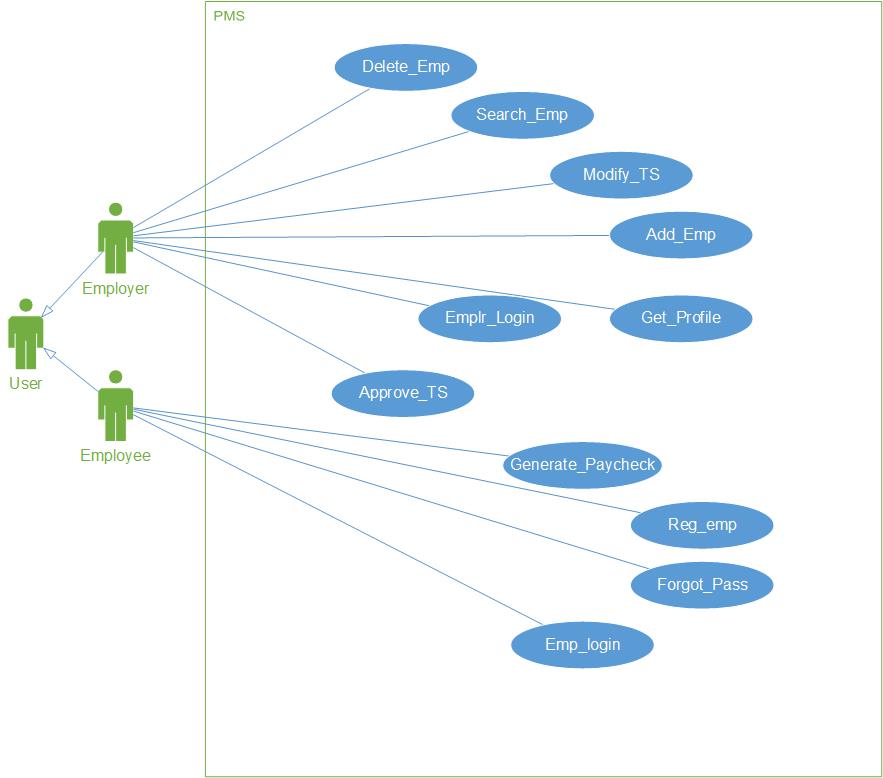


Figure 10.2.1: Use cases Implemented by the system

### Appendix C

Example of well documented test drivers, stubs used during unit and subsystem testing.

**Subsystem Example 1**

**Test Driver (including Mock)**

package controller;

import static org.mockito.Mockito.\*;

import java.io.PrintWriter;

import javax.servlet.http.\*;

import org.junit.After;

import org.junit.Before;

import org.junit.Test;

public class SubSystemTests {

HttpServletRequest request;

HttpServletResponse response;

HttpSession session;

PrintWriter printWriter;

ControllerFacade controllerFacade;

@Before

public void setUp() throws Exception {

controllerFacade = new ControllerFacade();

request = mock(HttpServletRequest.class);

response = mock(HttpServletResponse.class);

session = mock(HttpSession.class);

printWriter = mock(PrintWriter.class);

}

@After

public void tearDown() throws Exception {

request = null;

response = null;

session = null;

printWriter = null;

controllerFacade = null;

}

@Test

public void subSystem\_deleteempcontroller\_Sunny01() throws Exception {

//Test Setup

when(request.getParameter("caller")).thenReturn("delete\_emp");

when(request.getParameter("eid")).thenReturn("1");

when(response.getWriter()).thenReturn(printWriter);

//Test

controllerFacade.doGet(request, response);

verify(printWriter).print("<script language='javascript'>window.alert('Employee Details Deleted');window.location.replace('emplrhome.jsp');</script>");

}

**Employee Stub**

public class Employee {

ArrayList<String> employees = new ArrayList<String>();

public void setData() {

for (int i=1; i<11; i++)

{

String empid = Integer.toString(i);

employees.add(empid);

}

}

public String deleteEmp(String empid)

{

String result="fail";

this.setData();

if (employees.contains(empid)) {

result = "success";

} else {

result = "fail";

}

return result;

}

}

**Unit Test Example 1**

**Test Driver (including Mock)**

public class TestCase {

HttpServletRequest req;

HttpServletResponse rep;

HttpSession session;

PrintWriter print;

deleteempcontroller del;

@Before

public void setUp() throws IOException{

req = mock(HttpServletRequest.class);

rep = mock(HttpServletResponse.class);

print = mock(PrintWriter.class);

del = new deleteempcontroller();

}

@After

public void tearDown() {

req = null;

rep = null;

}

@Test

public void unitTest\_deleteempcontroller\_Sunny01() throws Exception, IOException {

String eid = "1";

when(rep.getWriter()).thenReturn(print);

when(req.getParameter("eid")).thenReturn(eid);

del.doGet(req,rep);

assertEquals("success", del.getResult());

verify(print).print("<script language='javascript'>window.alert('Employee Details Deleted');window.location.replace('emplrhome.jsp');</script>");

}

}

**deleteempcontroller Stub**

public void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {

PrintWriter pw=response.getWriter();

String eid = request.getParameter("eid");

//pw.print("ok");

//model.ModelFacade obj=new model.ModelFacade();

//String result=obj.deleteEmp(eid);

if(eid.equals(empid)) {

result = "success";

}

if(result.equals("success")) {

pw.print("<script language='javascript'>window.alert('Employee Details Deleted');window.location.replace('emplrhome.jsp');</script>");

}

else{

response.sendRedirect("error.jsp?msg="+result);

}

}

### Appendix D

Screenshots of the GUI testing tools used for one system test case.

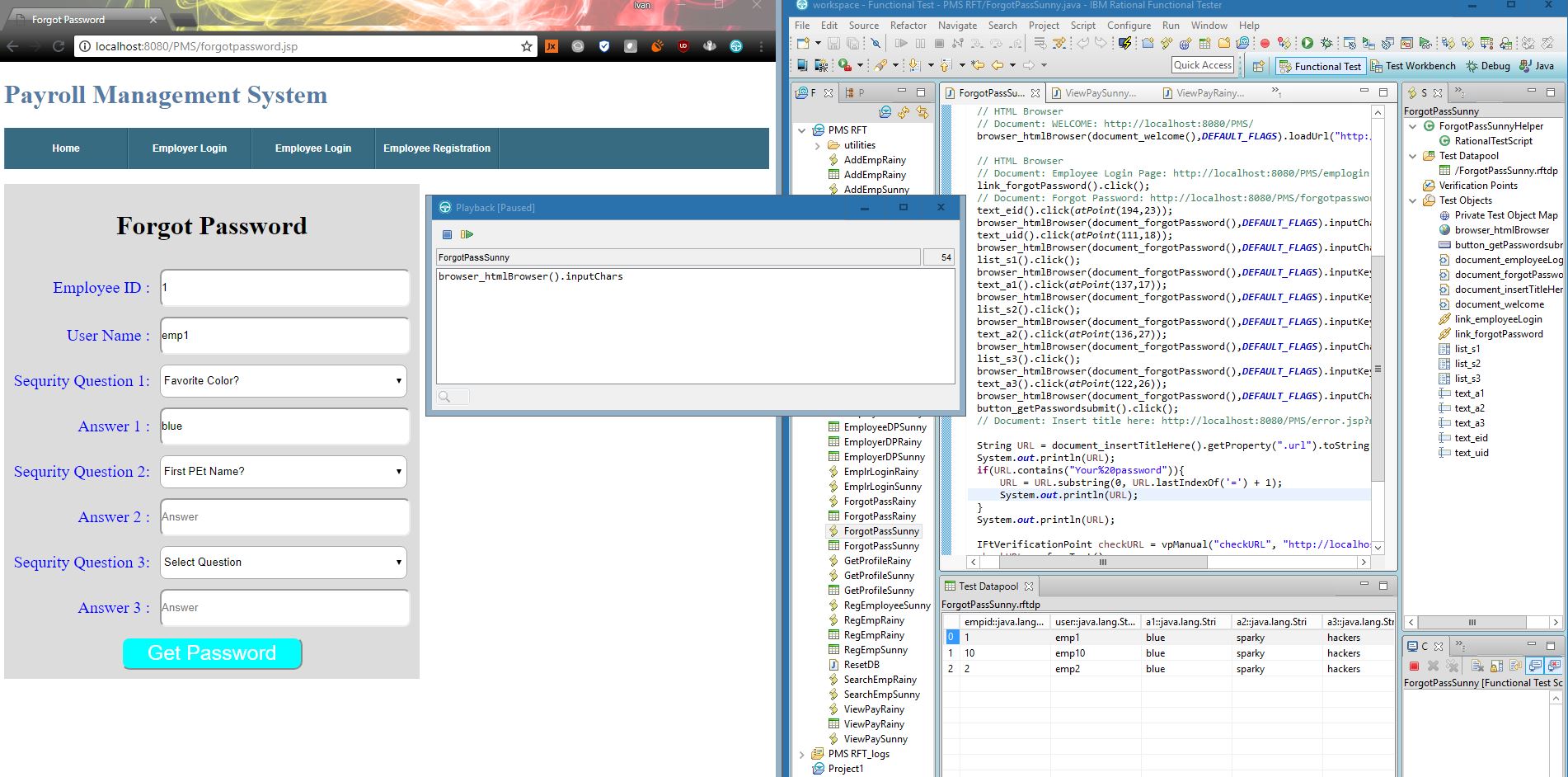


Figure 10.4.1: RFT performing ForgotPasswordSunny, with browser in use on the left, and RFT with datapool on right.

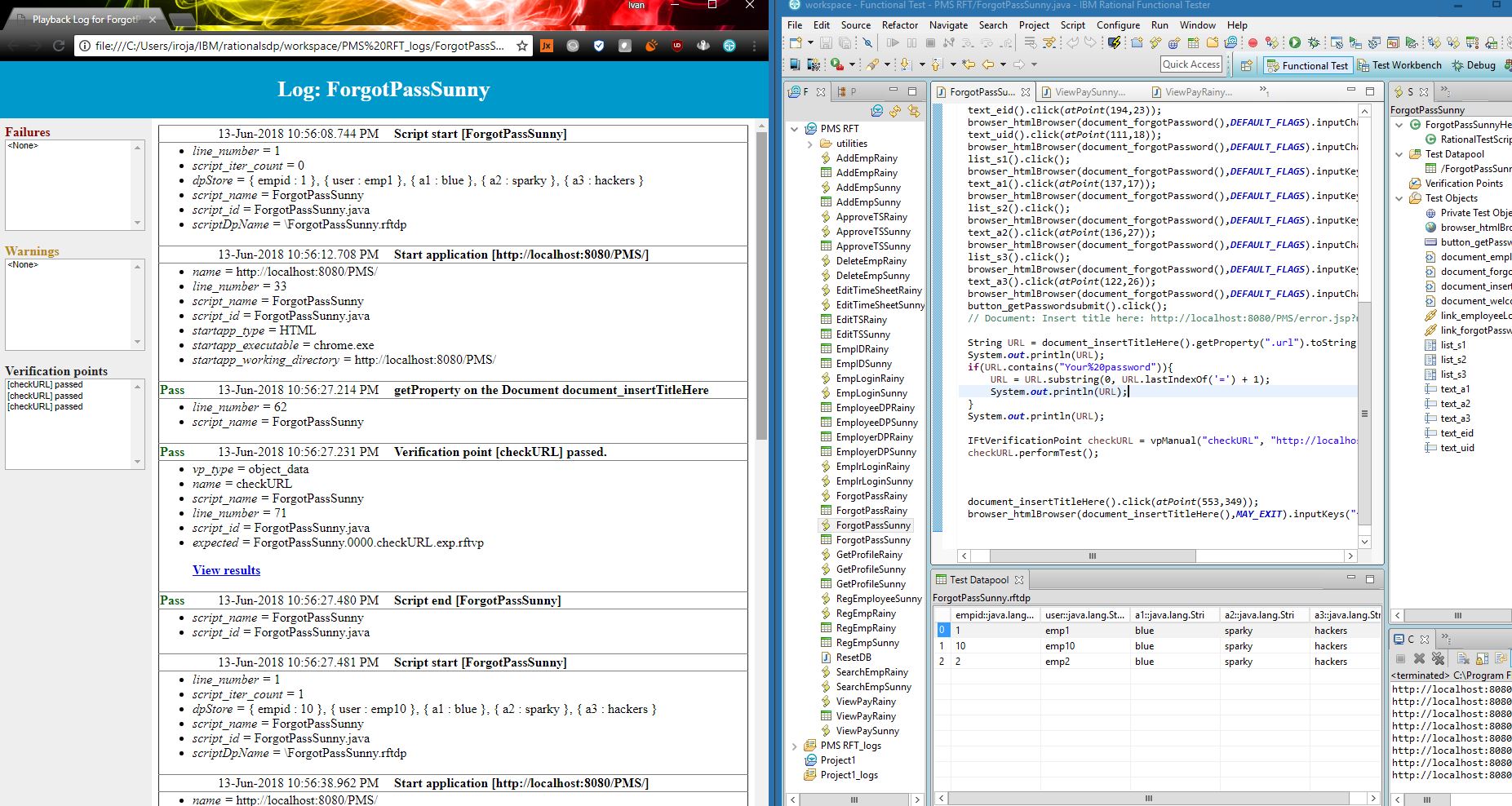


Figure 10.4.2: Log Results for ForgotPassSunny on left, with RFT and datapool on right.

### Appendix E

Diary of meeting and tasks.

Date: 05/20/2018

Location: Online - discordapp.com

Start time: 12:45 PM EST

End time: 1:15 PM EST

Attended: Dwayne, Juan, Daykel, Ivan, Ciana

Late: None

Agenda:

* Discuss where everyone is at in the project and assign responsibilities
* Pick a time for meet-ups
* Discuss test cases assigned by professor
* Go over Roles
* Go over Tools

Summary of discussion:

* Daykel commits to have the application and database running by 05/22/2018
* Dwayne commits to have application running by 05/22/2018 and the database 05/24/2018
* Ivan commits to have application and db server running by end of day 05/20/2018
* Ciana commits to have application and db server running by end of day 05/20/2018
* Moving forward we’re going to be meeting Thursdays after class for at least an hour.
* Juan has already done a sunny and rainy day test case
* Link that Ciana provided is permanent and will be used for online-meetings (<https://discord.gg/uejmkr>).
* All roles have been assigned.
* Mockito is not mentioned in the project specs so we need to ask the professor. ?????

Assigned tasks:

* Ciana agrees to do another sunny and rainy day
* Dwayne to learn J - unit and teach the group
* Ciana and Ivan to learn RFT and teach the group
* Ciana also to learn Cobertura and teach the group
* Juan to learn Code Cover and teach the group
* Daykel to learn eclipse EclEmma and teach the group

Date: 05/24/2018

Location: PG6 Lobby

Start time: 06:25 PM EST

End time: 7:15 PM EST

Attended: Dwayne, Juan, Daykel, Ivan, Ciana

Absent: Ciana

Agenda:

* Discuss where everyone is at in the project and go over missing items and set a due date.

Summary of discussion:

* Ivan, Daykel, and Dwyane is all done with app and db running.
* Controller Facade is all done.
* Ciana might still need to establish connection to DB

Assigned tasks:

* Model Facade to be worked on by Ivan, Ciana, Daykel and due on Tuesday.
* Dwayne to have the application and db running by next class.
* Source Code with controller face is completed, please download from Google Drive.
* Dwayne to review document specs and data by Tuesday.
* Juan to upload the pictures of the previous test in google drive.

Date: 05/31/2018

Location: PG6 Lobby

Start time: 06:30 PM EST

End time: 7:15PM EST

Attended: Dwayne, Juan, Daykel, Ciana

Absent: Ivan

Agenda:

* Discuss where everyone is at in the project and go start assigning items.

Summary of discussion:

* Everyone in the team has the app and facades (model and controller) up.
* Dwayne worked with professor Clarke in figuring out how to do test cases.

Assigned tasks:

* Ciana and Daykel will begin working on the paper this weekend
* Ciana and Ivan to finish writing test cases for system by next Thursday (06/07/2018)
* Juan and Daykel to finish writing test cases for subsystem by Thursday (06/07/2018)
* Dwayne to do half of the unit cases by Thursday (06/07/2018)

Date: 06/07/2018

Location: PG6 Lobby

Start time: 6:30 PM EST

End time: 7:15PM EST

Attended: Dwayne, Juan, Daykel, Ivan, Ciana

Late: NoneDate:

Agenda:

* Go over deliverable 1 and assign tasks

Summary of discussion:

* Dwayne is still working on learning Junit and Mockito
* Juan has begun to write the tests using Junit and Mockito and will make adjustment to meet professor’s suggestions
* Daykel began to write chapter 1 and 2 of the deliverable
* Ciana and Ivan still need to learn RFT

Assigned tasks:

* Everyone in the team to have their respective chapter (unit, subsystem, systems) testing completed by 06/12 so we can begin work on the presentation on 06/13
* Ciana and Ivan will continue to work on Chapter 1 and 2 and finish by 06/10
* Chapters 6-9 to be done in a group effort by whoever has bandwidth after completing chapters they are responsible for.
* Chapter 10 also needs input from every group so everyone is responsible for their share.
* Juan to do the test schedule (Appendix A).

### Appendix F

State of Database for SubSystem Testing

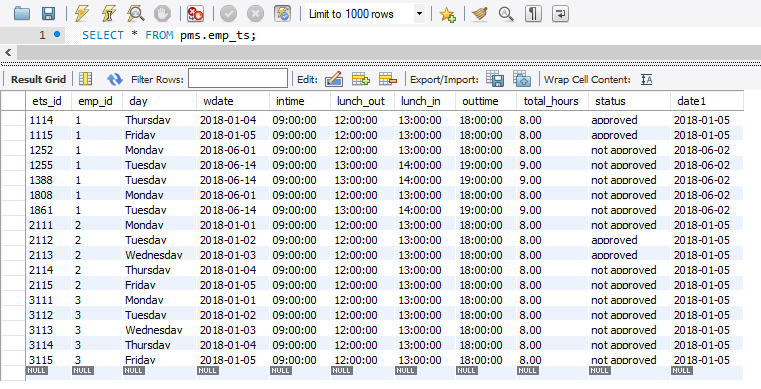


Figure 10.6.1: Employee Timesheets Submitted

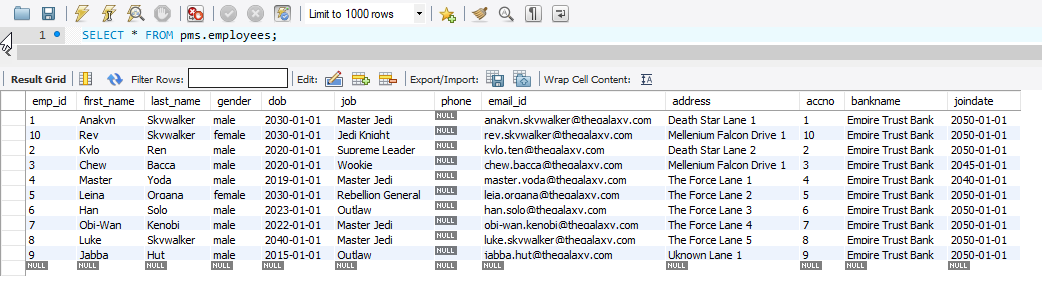


Figure 10.6.2: Employees

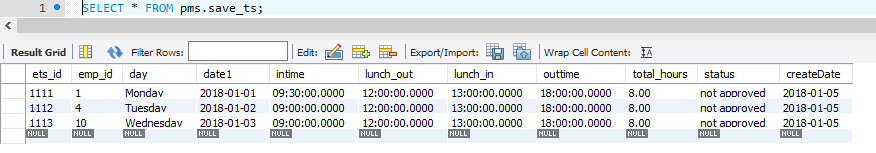


Figure 10.6.3: Employee Timesheet Not Submitted

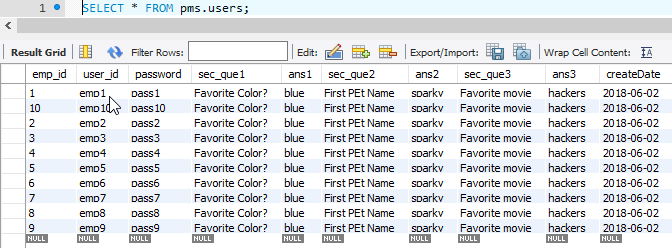


Figure 10.6.4: Employee Credentials

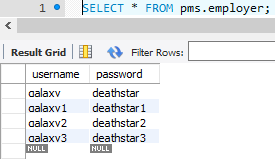


Figure 10.6.5: Employer Credentials

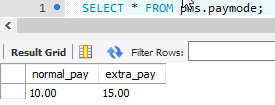


Figure 10.6.6: Pay Modes

### Appendix G

State of Database for System Testing

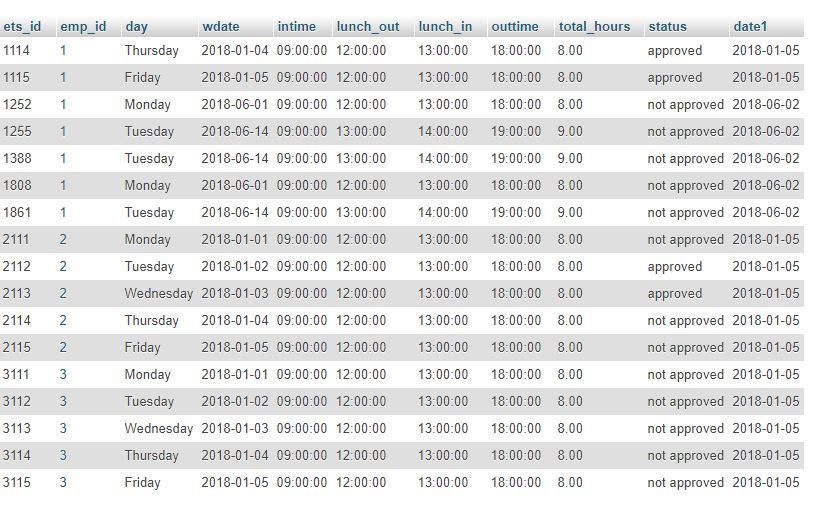


Figure 10.7.1: Employee Timesheets Submitted



Figure 10.7.2: Employees

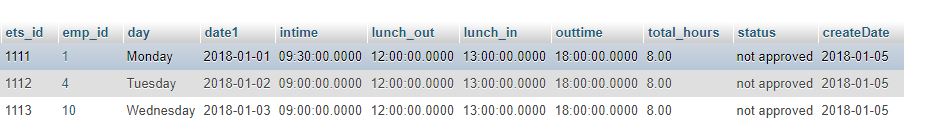


Figure 10.7.3: Employee Timesheet Not Submitted



Figure 10.7.4: Employee Credentials



Figure 10.7.5: Employer Credentials



Figure 10.7.6: Pay Modes