LEEDS METROPOLITAN UNIVERSITY

GRADUATE SCHOOL OFFICE
Coursework Submission Form

P03-F01

INFORMATION

STUDENT TO COMPLETE IN FULL (To be fastened securely to the front of all coursework. It is your responsibility to also attach any additional items. E g. CD, etc.

Please tick if you are:

Foundation ☐ Diploma ☐ Undergraduate ☐ Postgraduate ☒ Others ☐

Please Use Block Capitals

1	**Module Code & Title**	**Advanced Software Engineering**
2	**Course**	**MSC IN SOFTWARE ENGINEERING**
3	**Name of Lecturer**	**Mr. Riyaz Ahamed**
4	**Submission Date**	**25th April, 2016**

Please tick if this assignment is re-submission ☐

Please ensure that you have signed the declaration below before submitting your assignment. For group assignments, all students in the group must sign the declaration.

I/We understand that:
This assessment item is entirely **my/our** own original work, except where **I/we** have acknowledged use of source material [such as books, journal articles, other published material, the Internet, and the work of other student/s or any other person/s]. This assessment item has not been submitted for assessment for academic credits in this, or any other course at FTMS College or elsewhere.

I/We understand that:
The assessor of this assessment item may, for the purpose of assessing this item, reproduce this assessment item and provide a copy to another member of the college. The assessor may communicate a copy of this assessment item to a **plagiarism** checking service (which may retain a copy off the assessment item on its database for the purpose of future plagiarism checking).

STUDENT NAME	Student ID	Signature	Date
AMIR ANSARI	77172892		
EMMANCIPATE TAKUNDA MUSEMWA	77172238		
UADIALE HENRY AKHERE	77172169		

See information about plagiarism & academic misconduct from the Student Handbook

First marker's comment				
Second marker's comment				
Assessment Board/External Examiner:		**First Marker**	**Second Marker**	**Agreed Marks**
	TotalMarks			
	Signature&Date			

TABLE OF CONTENT

Contents

1. Introduction

Information is one of the most valuable things of the current information society. Fast access to needed information is critical in business at the present time. Companies have to spend huge amount of money to store the information about the employee and do changed when it's needed. The payroll system is an account system which stores the information of the employees at a one place and only transaction related action will be occurred here and if the any changes required in the future it can be easily updated.There are two accounts in IBM Payroll System and they have different permission to perform action.

First one is admin account which is the most important account in this payroll system and he can involve all the section of the employees and can perform some operation here. For example, hire the employee, delete the employee details, update salary etc. Manager account which is the second account in this system can access only few section of the employees. For example, expenditure report, print etc. Payroll management system will produce summary or salary report of all the employees or a specific employee at a specific date. The result of payroll accounting system is exact records to the costs connected with a wide range of pay, and other additional payments and remunerations to the workers. Payroll administration plays a key role in a business development. Any issues with the payrolls of an organization can prompt the workers disappointment that can have an immediate bearing on the proficiency of their administrations. A perfect payroll system influences each part of a business, from the spirit of workers to the sustainability and stability of the organization.

1.1 Aim

The goal of the IBM Payroll system is to develop an account system which can store all the information about the employees and can perform a quick action related to transaction and a it will also keep the history of all the transaction which can be accessed quickly it will save much money and time during the paying salary to the employees. For example, Allowance, Deduction or other benefits will be added automatically and system will generate net salary.

1.2 Objective

The main objective of the project is to develop an **"IBM Payroll System"** which provides more security during every transaction of the company to the employees and only those users can see the data of the employees who have appropriate permission. The basic idea behind the project is to reduce time during the payment of the employees or wasting a time during searching the allowance of a specific employee so in this system thousands of employees can be searched within a minute. This is a separate section so it will not effect of the companies' work so no more employee needed to work in this account so it will reduce the number of employees.

1.3 Requirements

Functional Requirements

Functional requirements explain a function of a system and its components. It describes the inputs the system will accept and the output it will produce. The table below shows the functional requirement of thesystem.

No.	Requirement Name	Description	Priority
1	Add New Employee	The system must allow the administrator to add new employees to the payroll system	H
2	Update Employee Details	The system must allow the administrator to update employee details	H
3	Log in	The system must allow users to log in with a unique username and password	H
4	Calculate Allowances	The system must be able to calculate total allowances and add them to the gross salary	H
5	Calculate Deductions	The system must be able to calculate total deductions and subtract	H

		them from the Payroll System	
6	Make Payment	The system must be able to make Payments to the employees	H
7	Generate Payslip	The system must be able to generate a payslip for an employee	H
8	Generate Overall Expenditure Report	The system must be able to calculate the overall company expenditure report	M
9	Generate Expenditure Report By Month	The system must be able to generate expenditure report by month	H
10	Print Reports	The system must be able to print generated reports	H
11	Logout User	The system must be able to logout or kill logged in user session.	H

Non- Functional Requirements

Non-functional requirement do not explain behavior but rather explain the ways that can be used to qualify the operations of the system. The table below shows the non-functional requirements for the online ordering system.

No.	**Requirement Name**	**Description**	**Why needed**
1	Availability	**Hours of Operation**- the system should be available 24/7 excerpt maintaining times. **Locations of Operation-** The system will be available to everyone online but only allows access to people in a	-To serve customers at any time -deliver distance restrictions

		100mile radius from IBM offices	
2	Security	Must allow users to login with a unique user name and password and restrict some areas to people with authority to access them only like the administrator	Prevent fraud and manipulation of information
3	Performance	**Response time-** application must load and refresh fast. **Processing time-** application must perform calculations fast. **Query and report times**- Application must load initial and subsequent loads fast	Provide results fast to user
4	Capacity	**Throughput-** the system should be able to handle over 100 transactions per hour **Storage-**the system should be able to store 300GB of data The system should have room to grow	Amount of items the system can handle can be over capacitated d
5	Reliability	**Mean time between failures –** less than 4,000 hours per year. The system must be reliable. **Mean time to recover-** if down the system must take less than an hour to recover	Customers need to access the system at all times.

6	Compatibility	The system should be compatible with **Shared applications- it** should communicate with flash players and web browsers etc **3rd party applications- it** should live amicably with antivirus software **Operating systems-** it should be able to run on window 7 and above, and above, linux and mac OS. **Platforms-** it should work on different hardware platforms and mobile devices	To make the system work on different situations.
7	Maintainability	The system must be easy to maintain, upgrade and grow	For development and error correction
8	Usability	**Look and feel-** the interface must be user friendly with the colors, text, space, keyboard shortcuts all welcoming for the user.	Easy and simple to use
9	Audit	System should allow auditing of some data elements like payment details	For security reasons.

1.4 Feasibility Study

This Payroll System is feasible in many ways which can be described from the following points.

- **Response Time**: The system has quick data access so theresponse lime is very less.
- **Data Integrity:** This portal shall have a centralized data which will be accurate.
- **Reliability:** A system is reliable if it fulfills requirements andrecognizes and prevents errors. This project is reliable for thisway.
- **Communicativeness:** This system will be accessible throughInternet.
- **Environment Factor:** The portal will save paper because ofdata files stores on hardware. This one decreases air pollution.
- **Compatibility:** As the project will be accessible through Internet, so it will be made to compatible with the standard hardware and software.
- **Social Objective:** In today's busy and last world where there is a chance to transfer payment in wrong account so it will prevent all the problems which can happened during the transaction.

1.5 Scope

This project is developed keeping in mind all the future facilities of Payroll system and it is the new trend of internet banking where company can also perform same operation like bankingsystem from their place. This system will help to employee and their salary will be transferred after every month in their account with some allowance or deduction without any mistaken because everythingwill be generated by the system. In this system there are many post of the employees so salary will be generated according to the post so it will save the time and

money of the company and only one accountant can perform all the operation in the transaction.This site is developed keeping all facilities of the Payroll System in mind but if in the future some changes required in this system so it can changed quickly because a lot of components developed for this project can be reused for future projects with a little change.

1.5 Platform Specification

Every project needs some tools and programming language that has to be decided in advance during the analysis. This project also uses quite a number of tools as well as languages which are explained below in detail. This project can work on both Windows and Linux platforms but I am developing and testing only on windows platform:

1.5.1 Tools Used

- **Java Development kit (JDK)** -> is a tool which is used to development the JAVA SE, JAVA ME, or JAVA program in a system etc.

- **Java Runtime Environment (JRE) ->** this application is used to run the java program on a system not for development. JDK is important for development so for this project both Software needs. This project will be required JDK 1.7 and JRE 7 and also it depends on the tomcat server.

- **Eclipse:** This software is used to implement the JAVA, JSP or any other programming languages. Web application eclipse IDE for Java EE Developers will be needed because this is a web application. Eclipse Mars.1 (4.5.1) is used to implement this project.

- **Tomcat:** This is server which is used to deploy the code and output can be seen on the browser. Tomcat 6.0 or above version is required for this project.

-
- **MySQL**: This is a database which is used to store the data about the client or another information at one place and can be accessed easily.

- **Web Browser**: There are two web browser to see the output first one internal which is embedded with eclipse and other is external browser (Google Chrome, Firefox etc.).

1.5.2 Languages Used

The project uses the **J2EE platform** and some another technologies will be used for a little work and some will be used throughout. The following is a complete list of all technologies and their use in the project in detail:

- **JAVA –** It is one of the most powerful Object Oriented Platform Independent Language. Java is used in most of the project in one form or another.

- **JSP –** Java Server Pages or JSP for short is a server-side technology that takes Java language with its inherent simplicity, and uses it to create highly interactive and flexible web applications. JSP will be used in the project to create most of the web pages that interact with the user.

- **JavaScript –** JavaScript is used for client- side scripting. It enables the web pages to have some programmatic functionality in the browser. JavaScript works with and can manipulate the HTML page in which it is embedded. JavaScript is used in the project to validate the data entered in the web pages by the user before it is send to the server.
- **Servlet –** A Servlet is a Java program that generates dynamic web content. They are written using the Java Servlet API and are managed by a Servlet container such as

Tomcat. The Servlet processes the user request, builds a response, and passes it to the container which it back to the user. I have used a Servlet for making the controller in the web based system. It processes the user requests and gives appropriate response according to them.

- **HTML –** HTML or Hyper Text markup Language is used creating web pages. HTML pages are static and do not interact with the user but can be made interactive by adding JSP elements them. Most of the web pages in the project are designed in HTML and after that JSP elements are added to them.

- **XML –** Extensible Markup Language or XML for short has become the de facto standard for data interchange on the Internet. We have not used it directly but a lot of configuration files of Apache Tomcat Server as well as Apache James E-mail server are written in this format.

1.6 Security Issue

Portal is secure both client side and server side no one can see the employees' details except admin and manager. Admin and manager have different types of accessibility but before they perform any operation with the employees, they should have some unique username and password. Care of security is taken for individual servers as well as the whole system. Steps are taken to counter three types of security issues which are as follows:

- ❖ Unauthorized access to database server.
- ❖ Unauthorized access to payroll system.
- ❖ Steps taken against hacking of system.

1.6.1 Data Integrity & Validation Checks

Data integrity is very important in any project because invalid data is of no use so various measures are taken for maintaining data integrity. In web based payroll system the most important data for the smooth functioning of the system is the data contained of the employee. There are two steps for maintaining data integrity:

- **Value Constraints and Ranges**
- **Validation Checks**

1.7 Value Constraints and Ranges

The first step for maintaining data integrity is to identify various value ranges of various attributes. This is done in the design phase in the payroll system admin can make changes to the data constraints are defined on the data entered by the admin which are as follows:

Attribute	**Value constraint and range**
username	User id contain a-z characters.
password	Contains password a-z character and 0-9 number
Employee ID:	Employee ID can contain a-z and 0-9 characters.
First Name	Only can contains a-z character
Last Name:	Can contains only a-z character
Gender	Can contain only Male or Female
Date Of Birth:	Must be in date format MM/DD/YYYY
Email Address:	An email address.
Phone Number:	Can contains only numeric value 0-9
Home Address:	Can contain any character
Designation:	Can select any post
Department:	Can select any department

Date Joined	Must be in date format MM/DD/YYYY

1.8 Validation Checks

Validation checks are the second step in maintaining data integrity. These are implemented in the coding phase of the project development. When the admin enters data validation checks are performed over it before using it. If it is found to violate its value range then an appropriate message is shown to the admin.

The project developed also performs a lot of validations on the data entered. Some of them are given below:

Attribute	**Validation Checks Performed**
username	Is thoroughly checked for value constraints before checking it with the server.
password	Checked only at login time to be more than six characters.
Employee ID:	Only administrator can give any employee id
First Name	Checked for blank entry.
Last Name:	Checked for blank entry.
Gender	Select any value
Date Of Birth:	Drop down list for entering the value.
Email Address:	Checked for blank entry.
Phone Number:	Checked for blank entry.
Home Address:	Checked for blank entry.
Designation:	Drop down list for entering the value.
Department:	Drop down list for entering the value.
Date Joined	Drop down list for entering the value.

2.0 System Analysis and Design

- **Output**: The project is created for report and printouts as output of the portal's all static or dynamic web pages.
- **Inputs**: Input of the project through form of Java Servlets under the following points.
- **Accuracy**: Data will be always accurate because data enter through textbox and combo box with fixed length and data type also. Wrong data not accepted by the project because each field uses data validation.
- **Timeliness**: Data is access very fast because the database is MySQL and not complex calculation present in the project.
- **Proper format**: For proper format input and output HTML, FrontPage and Java Servlet are used.

2.1 System Design

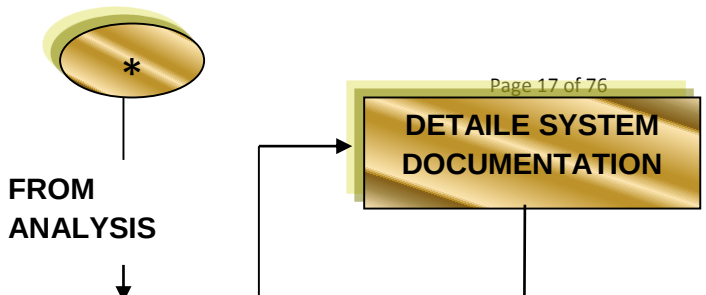


Figure 1.1: Belong to system design

2.2 Data Flow Diagrams

A data flow diagram is a graphical representation that depicts information flow and the transformations that are applied as data move from input to output. The DFD can be partitioned into levels that represent increasing information flow and functional detail.

LOGIN PROCESS

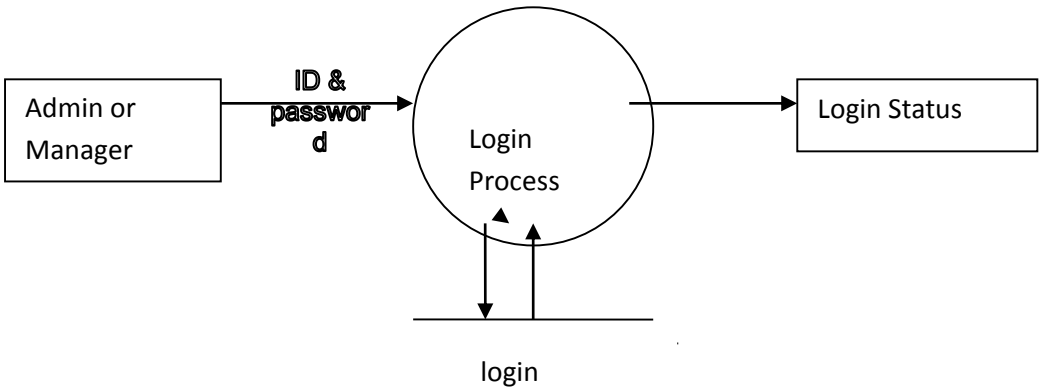


Fig 1.2 Login Process

LOGOUT PROCESS

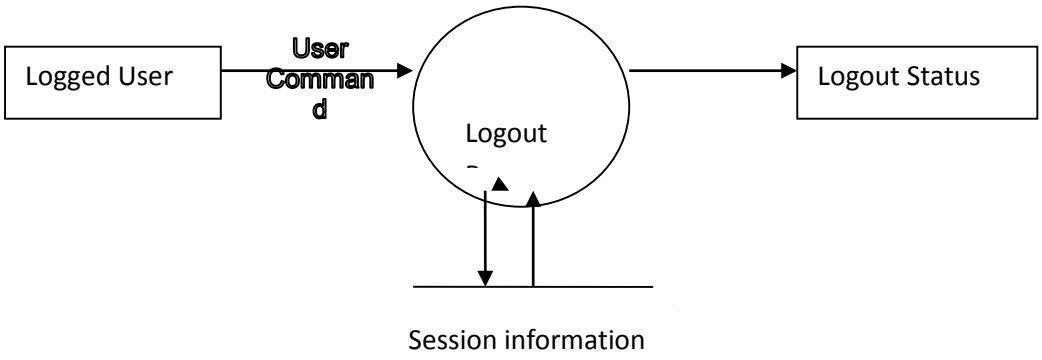
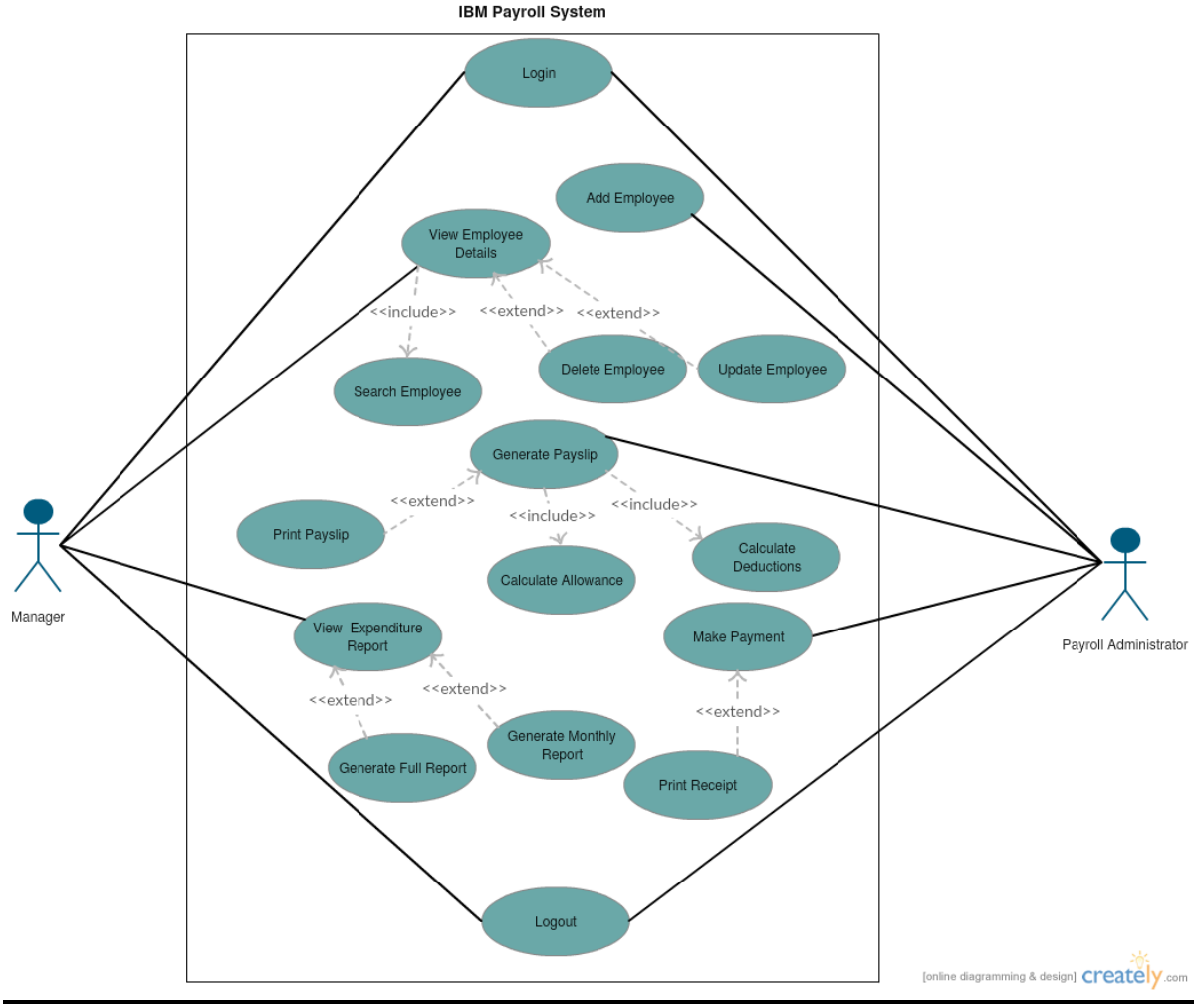


Fig 1.3 Logout Process

2.3 Use Case Diagram



Term	Synonym	Description
Administrator	Admin	An individual who maintains and monitors the database of the system and also perform main functions of the payroll system
Manager	Manager	A user who is only interested in viewing and printing reports. This user views and prints expenditure reports for accounting purposes

LIST OF USE CASES

Use case name	Use case Description	Actor
Log in	The system must allow users to log in with a unique username and password	Administrator (primary business) Manager (primary business)
Logout	This use case destroys logged in users session	Administrator (primary business) Manager (primary business)
Add Employee	The use case allows the administrator to add new employee to the payroll system.	Administrator (primary business)
View Employee Details	The use case allows the administrator to view employee details	Administrator (primary business)
Update Employee	The use case allows the administrator to update employee details	Administrator (primary business)
Search Employee	The use case allows the admin to search for an employee by employee ID	Administrator (primary business)
Delete Employee	The use case allows the administrator to delete an employee's details from the payroll system.	Administrator (primary business)
Generate Payslip	The use case allows the user to generate a payslip for a particular user.	Administrator (primary business)
Calculate Allowance	The use case allows calculation and addition of monthly allowances for a particular employee	Administrator (primary business)
Calculate Deductions	The use case allows calculation and subtraction of monthly deductions for a particular employee	Administrator (primary business)

Use case name	Use case Description	Actor
View Expenditure Report	The use case allows the user to view payroll expenditure reports	Administrator (primary business) Manager (primary business)
Generate Full Report	The use case allows the user to view total money spent on employees	Administrator (primary business) Manager (primary business)
Generate Monthly Report	The use case allows the use to view expenditure reports by month	Administrator (primary business) Manager (primary business)
Make Payment	The use case allows the money to be paid to the employee's account.	Administrator (primary business)
Print Payment Receipt	The use case allows the user to print the payment receipt.	Administrator (primary business)

2.4 Class Diagram

Class Diagram provides an overview of the target system by describing the objects and classes inside the system and the relationships between them (Visual-paradigm.com, n.d.). In the class diagram they are two main users of the system, the manage and the administrator. Both of this classes inherit attributes from the User super class. The administrator interacts with all the other classes while the manage can only view reports.

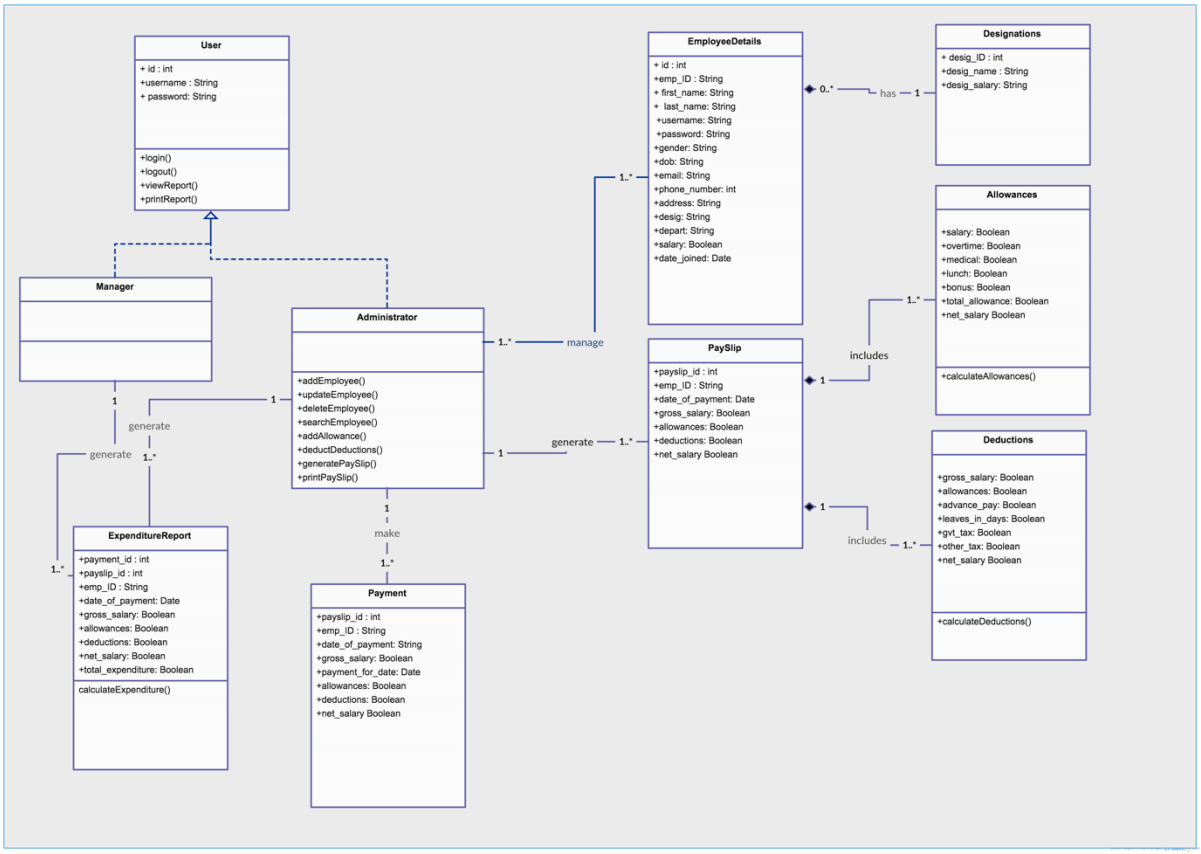


Fig 1.5 Class Diagram

3.0 Implementation

There are two accounts in the home page of the IBM payroll system which is playing different roles and only valid user can access these accounts and perform some actions.

A. Admin Account
B. Manager Account

3.1 Admin Account

After entering the a valid username and password, admin can perform these actions

- Admin Login
- Admin Menu
- Add Employee
- Update Employee
- Search Record
- Delete Record
- Allowance
- Deduction
- Make Payment
- Expenditure Report
- Print Report

3.1.1 Admin Login

This system is more secure and if the user enters a wrong user id and password or leave the blank space it will show the error only valid admin can access this account.

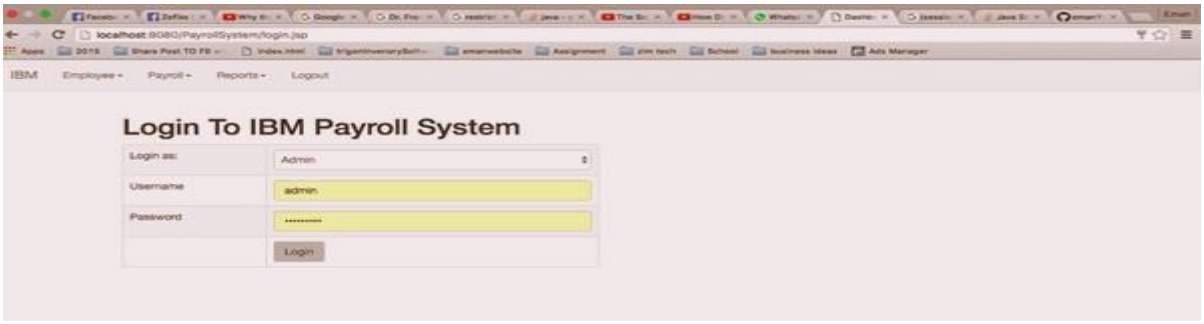


Fig 3.1.1 admin login

3.1.2 Home menu Admin

After successfully login admin can perform different actions which are showing in the payroll menu image.

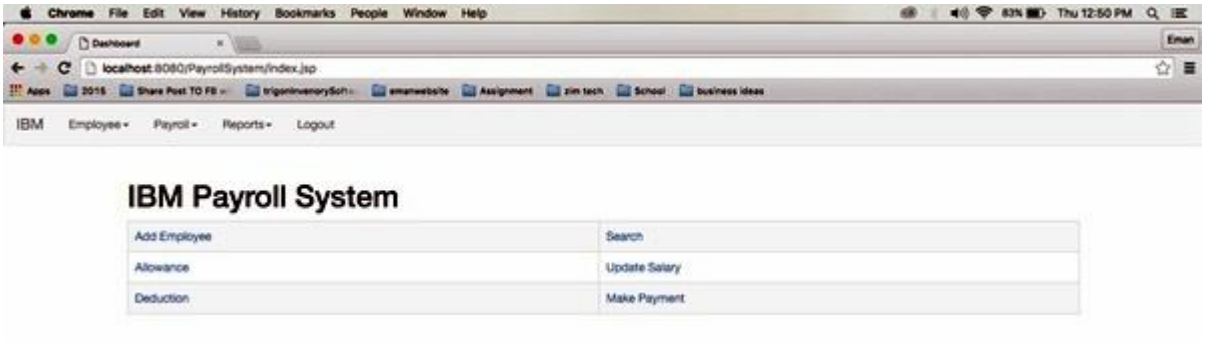


Fig 3.1.2 Admin home page

3.1.3 Add Employee

Admin hire the new employees and fill their details in a proper form at the add employee page and can also provide the salary of the employee.

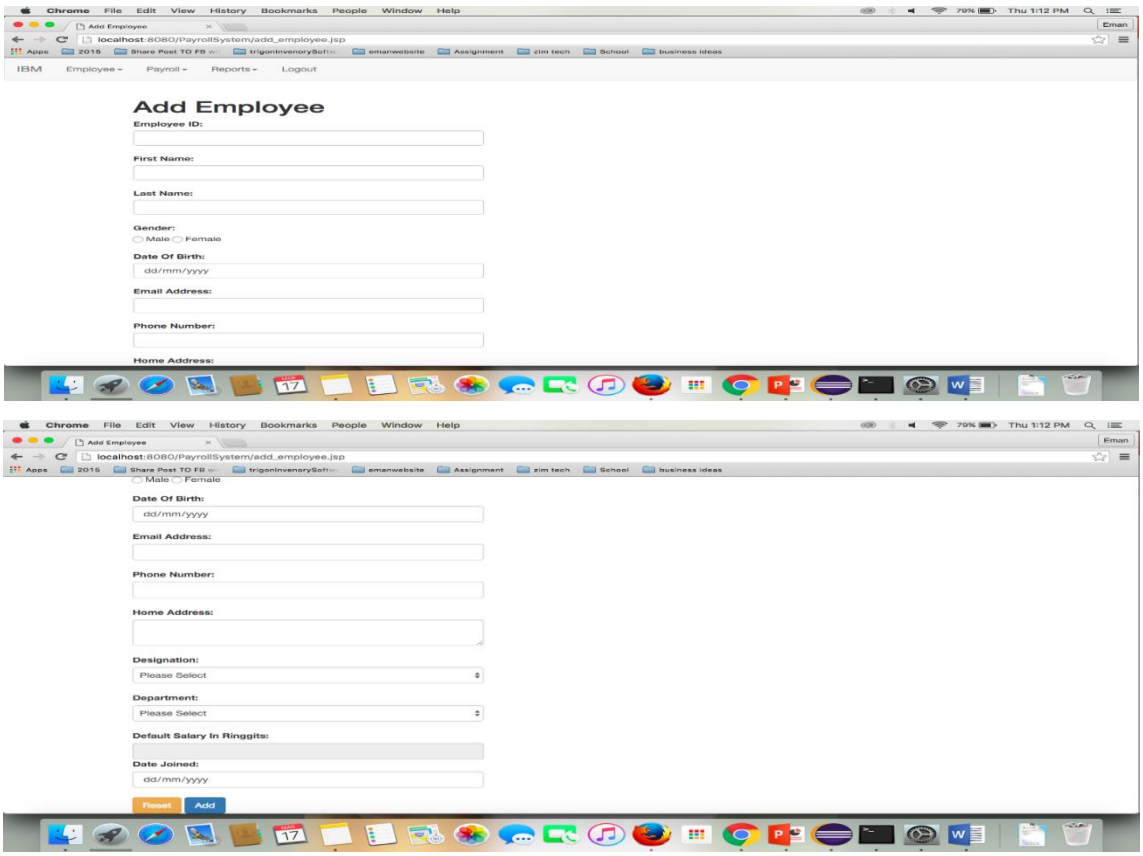


Fig 3.1.3 information about the employee

3.1.4 Update Employee

Employees' details are not fixed in this payroll system if in the future any correction needed so it can be easily updated by the admin. For example, name correction, salary increment, department change but Employee Id will be fixed it cannot be changed because it is a primary key in this system.

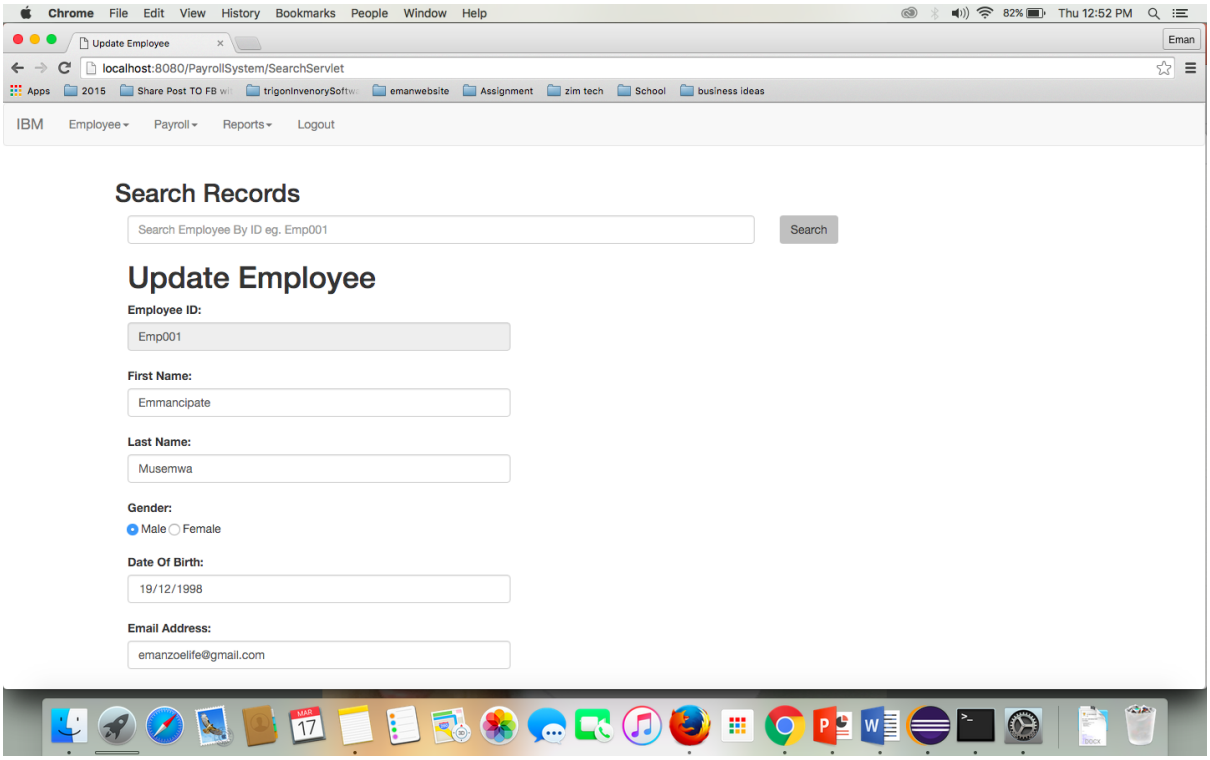


Fig 3.1.4 Update Data

3.1.5 Search Record

This system is providing search box where admin can easily get the information about the particular employee after entering a valid employee id and if the admin enter wrong employee id it will not show any results so will save the much time of the admin during updating any record of the employee.

Fig 3.1.5 Searching employee

3.1.6 Delete Record

This section will delete the record of the particular employee after searching employee id on the search box only admin can delete the records or employees can also approach to the admin for resign the job.

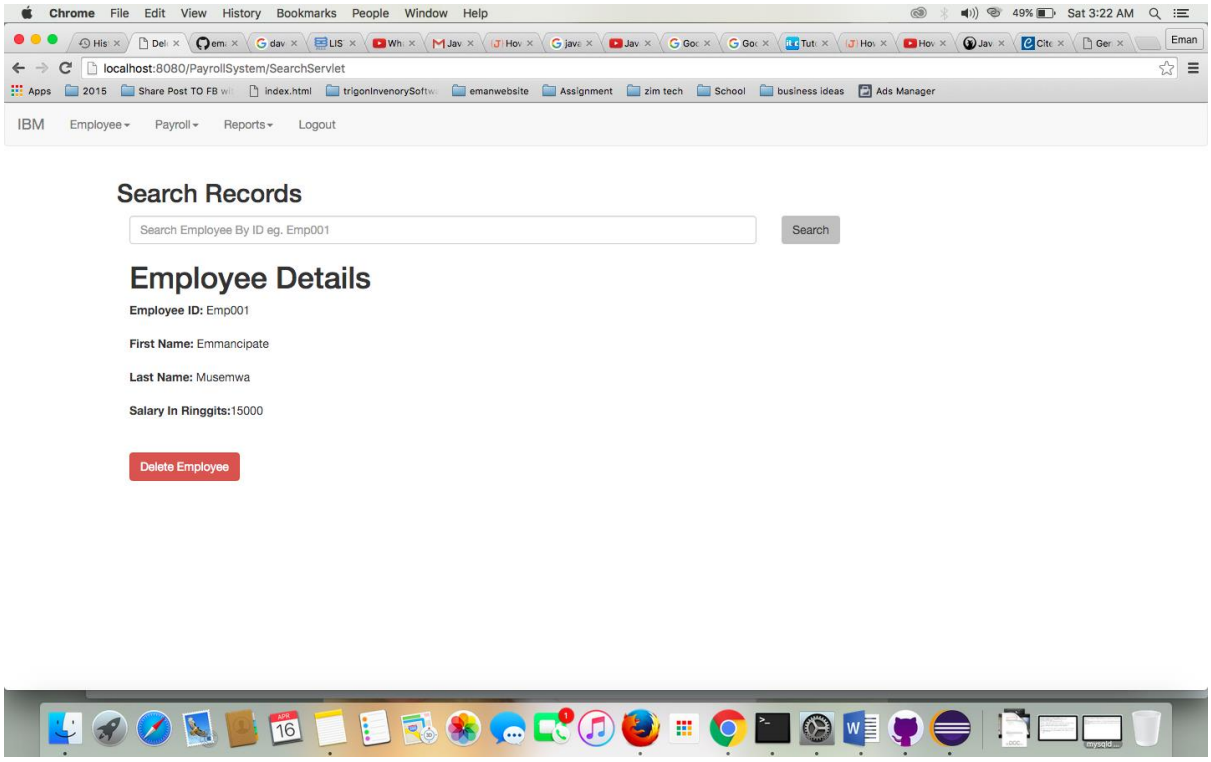


Fig 3.1.6 Delete employee record

3.1.7 Calculate Allowance

Admin provide the extra salary or all other benefits if the employees do the overtime work and it can be calculated easily because in this system most of the things are inbuilt which will calculate the exact amount of allowance and will add in the salary

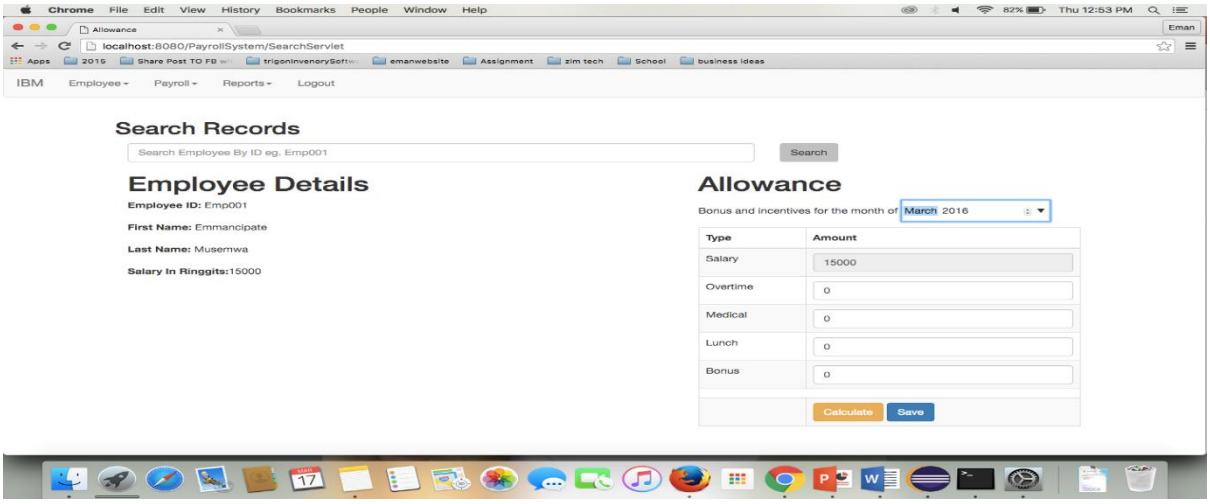


Fig 3.1.7 Allowance Calculation

3.1.8 Calculate Deduction

The same operation will be done as allowance but here admin will subtract all the amount from the salary which is taken in advance. For example, advance pay, leaves in days etc.

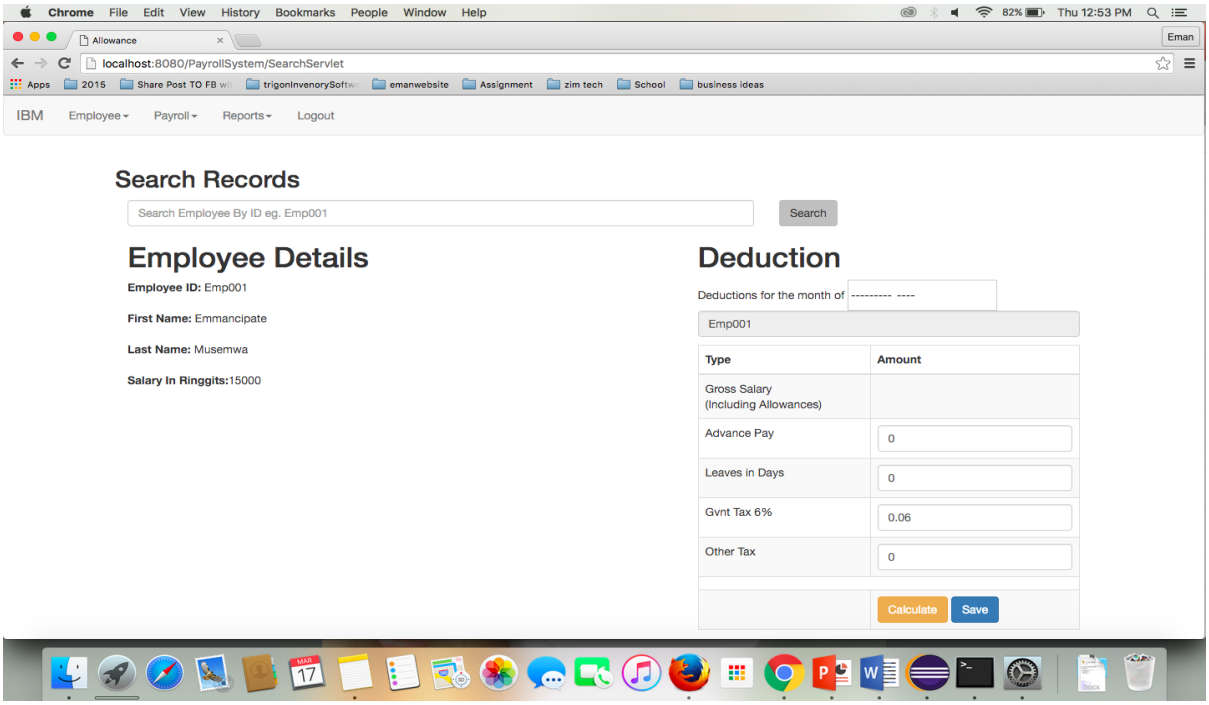


Fig 3.1.8 Deduction Calculation

3.1.9 Make Payment

Admin will make a payment after all allowance and deduction amount from the salary for a particular employee and a specific month or can also print the receipt of it.

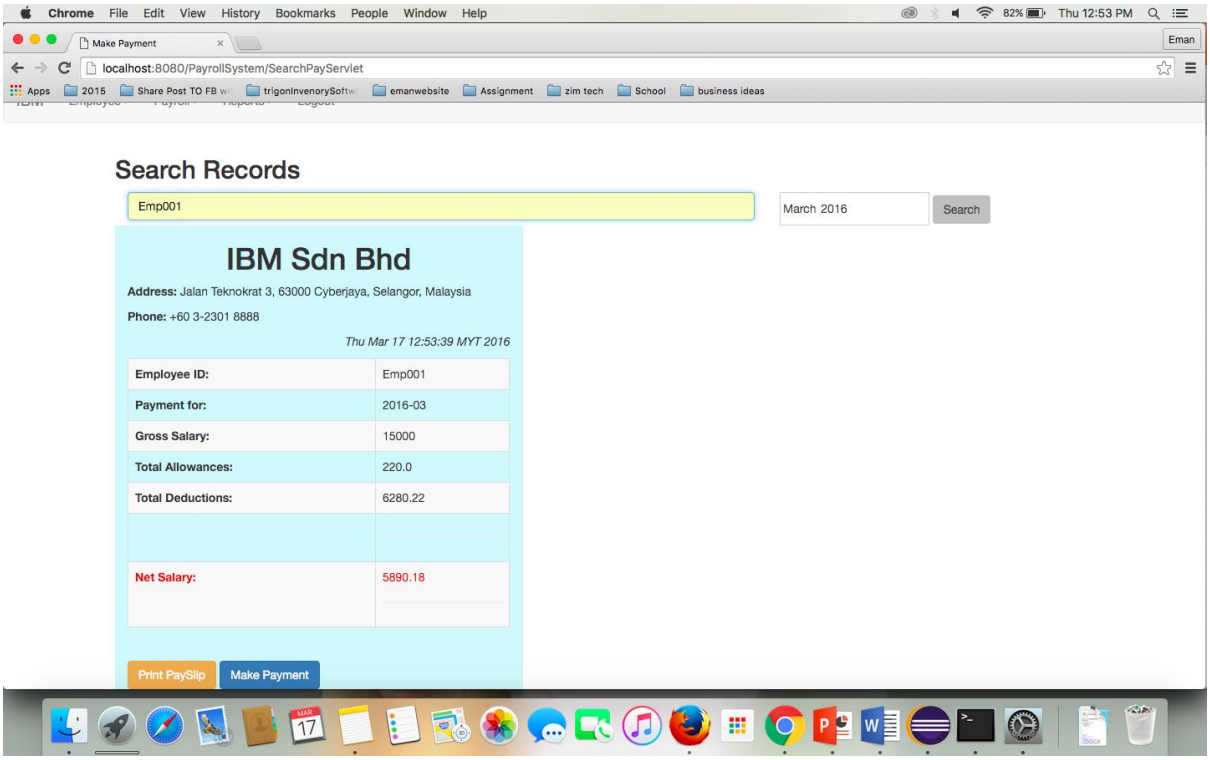


Fig 3.1.9 Make payment of particular employee

3.1.10 ExpenditureReport

Admin can see the expenditure report of all the employees which they have made a payment or can also see the expenditure report of a particular month.

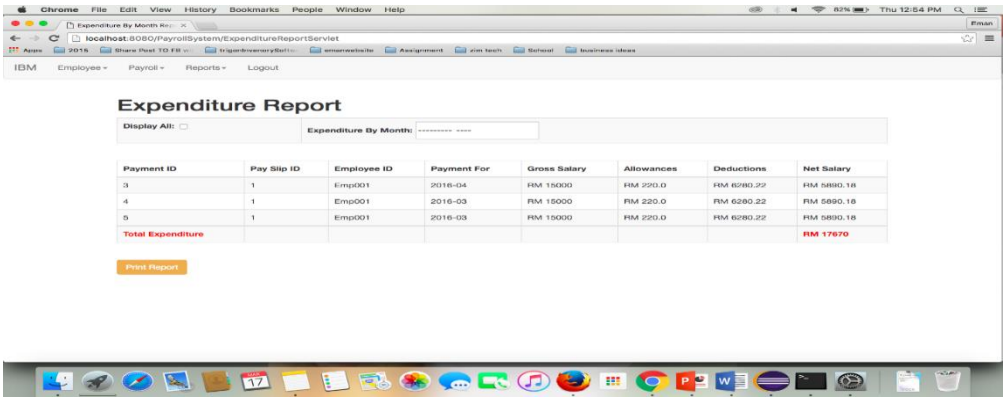


Fig 3.1.10 Expenditure report

3.1.11 Print Report

Admin can also print the employee expenditure report if the employees needed.

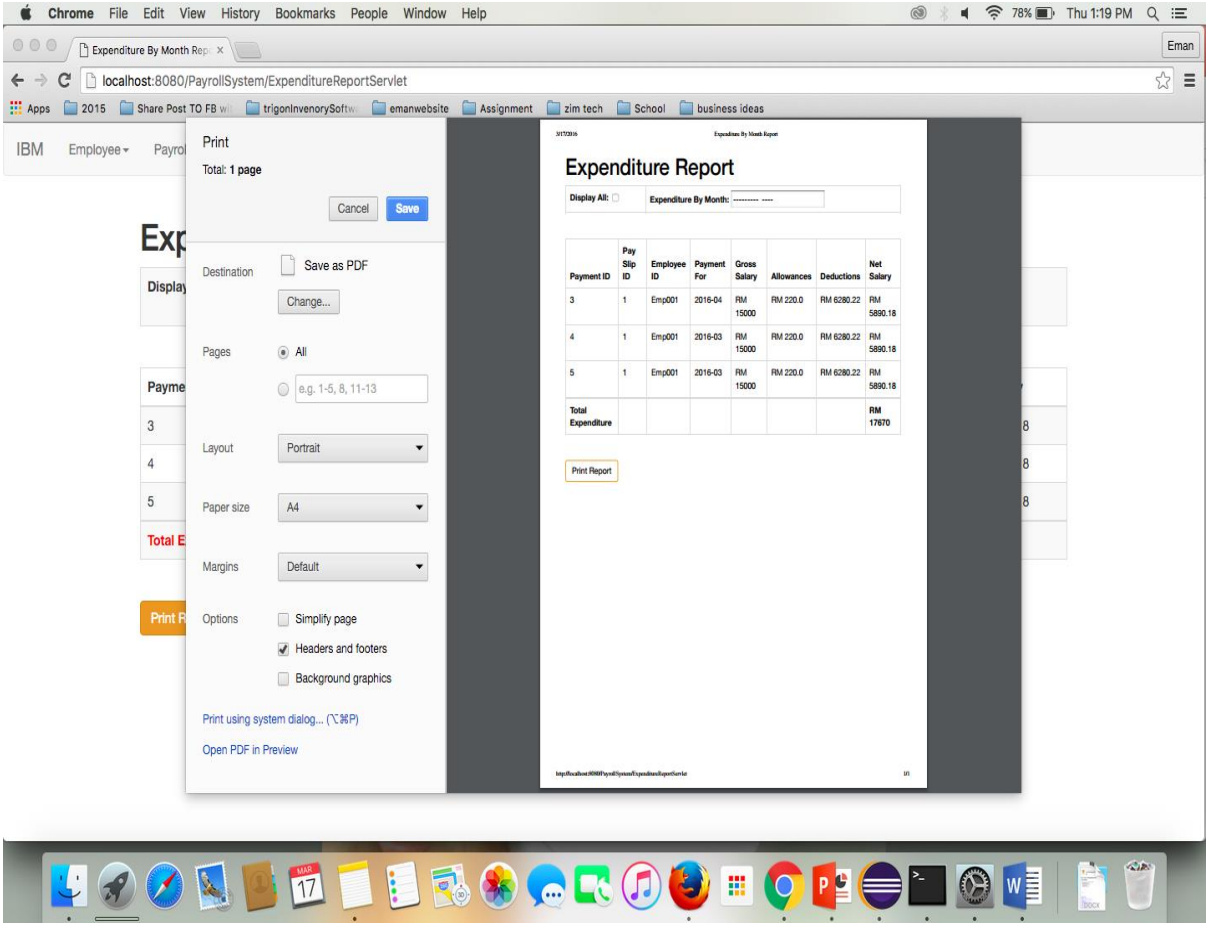


Fig 3.1.21 Print report

3.2 Manager Account

Manager can perform some task which are given below after successfully login

- Expenditure Report
- Expenditure Report by Month
- Print the Report

Fig 3.2 manager Account

4.0 Testing

It is the job of programmer to test, as far as possible, that all parts of the programs work correctly. It should be realized that complete testing is not possible except in the case of the most trivial program. One can never be completely certain that all errors have been removed, but sufficient test can be performed to give a reasonable measure of confidences in the program.

4.1 Unit Testing

In unit testing the programmer tests all the individual modules that are making the system. Unit testing gives stress on the modules independently of each other. This helps in easily detecting the logical errors in the modules. In my project all the individual files that were coded were tested individually for any errors. They were tested for the inputs they take and were tested to see what happens when we enter an illegal value for a field. After the results of an individual module were found satisfactory

4.2 Manual testing

First this payroll system tested manually to find the error during entering any unexpected value into the textbox so every value is tested to check that it value is in valid form or invalid and data is submitting on the database.

4.2.1 LEGEND

A lot of symbols are used in the testing data and reports which are given as follows:

1. N/A : Not Applied.
2. A : Apply
3. S : Select this field.
4. BFP : Blank Field Prohibited.
5. BFA : Blank Field can be applied.
6. MBE : Must Be Existing.
7. EMA : Error Message Appeared.
8. D: Default Value

4.3 IBM Payroll System Classes

UNIT TESTING OVERALL RESULT			
S.No.	**Module Description**	**Result**	**Successful/Failed**

1	Login Admin	As Expected	Successful
2	Login Manager	As Expected	Successful
3	Add Employee	As Expected	Successful
4	Update Employee	As Expected	Successful
5	Search Record	As Expected	Successful
6	Delete Record	As Expected	Successful
7	Allowance	As Expected	Successful
8	Deduction	As Expected	Successful
9	Make Payment	As Expected	Successful
10	Expenditure Report	As Expected	Successful
11	Expenditure Report By Month	As Expected	Successful
12	Logout	As Expected	Successful

UNIT Test Case Status:	***Successful***

4.3.1 Module Name: Admin Login

ARRAY OF VALUES								
Field Name	**Blank Entry**	**Numeric Entry**	**Alphabet Entry**	**@ Entry**	**Special Symbol**	**Expected Result**	**Observed Result**	**Test Result**
Username	BFP	N/A	A	N/A	N/A	EMA	EMA	**Pass**

Password	BFP	A	A	N/A	N/A	EMA	EMA	**Pass**

Module Test Status:	***Successful***

4.3.2 Module Name: Manager Login

ARRAY OF VALUES								
Field Name	**Blank Entry**	**Numeric Entry**	**Alphabet Entry**	**@ Entry**	**Special Symbol**	**Expected Result**	**Observed Result**	**Test Result**
Username	BFP	N/A	A	N/A	N/A	EMA	EMA	**Pass**
Password	BFP	A	A	N/A	N/A	EMA	EMA	**Pass**

Module Test Status:	***Successful***

4.3.3 Module Name: Add Employee

ARRAY OF VALUES								
Field Name	**Blank Entry**	**Numeric Entry**	**Alphabet Entry**	**@ Entry**	**Special Symbol**	**Expected Result**	**Observed Result**	**Test Result**
Employee ID	BFP	A	A	N/A	N/A	EMA	EMA	**Pass**

First Name	BFP	N/A	A	N/A	N/A	EMA	EMA	**Pass**
Last Name:	BFP	N/A	A	N/A	N/A	EMA	EMA	**Pass**
Gender:	BFP	N/A	S	N/A	N/A	EMA	EMA	**Pass**
Date Of Birth:	BFP	S	N/A	N/A	N/A	EMA	EMA	**Pass**
Email Address:	BFP	A	A	A	A	EMA	EMA	**Pass**
Phone Number	BFP	A	N/A	N/A	N/A	EMA	EMA	**Pass**
Home Address	BFP	A	A	A	A	EMA	EMA	**Pass**
Designation	BFP	N/A	S	N/A	N/A	EMA	EMA	**Pass**
Department	BFP	N/A	S	N/A	N/A	EMA	EMA	**Pass**
Default Salary In Ringgits	BFP	D	N/A	N/A	N/A	EMA	EMA	**Pass**
Date Joined:	BFP	S	N/A	N/A	N/A	EMA	EMA	**Pass**

Module Test Status:	***Successful***

4.3.4 Module Name: Update Employee

ARRAY OF VALUES								
Field Name	**Blank Entry**	**Numeric Entry**	**Alphabet Entry**	**@ Entry**	**Special Symbol**	**Expected Result**	**Observed Result**	**Test Result**
Employee ID	BFP	A	A	N/A	N/A	EMA	EMA	**Pass**

First Name	BFP	N/A	A	N/A	N/A	EMA	EMA	**Pass**
Last Name:	BFP	N/A	A	N/A	N/A	EMA	EMA	**Pass**
Gender:	BFP	N/A	S	N/A	N/A	EMA	EMA	**Pass**
Date Of Birth:	BFP	S	N/A	N/A	N/A	EMA	EMA	**Pass**
Email Address:	BFP	A	A	A	A	EMA	EMA	**Pass**
Phone Number	BFP	A	N/A	N/A	N/A	EMA	EMA	**Pass**
Home Address	BFP	A	A	A	A	EMA	EMA	**Pass**
Designation	BFP	N/A	S	N/A	N/A	EMA	EMA	**Pass**
Department	BFP	N/A	S	N/A	N/A	EMA	EMA	**Pass**
Default Salary In Ringgits	BFP	D	N/A	N/A	N/A	EMA	EMA	**Pass**
Date Joined:	BFP	S	N/A	N/A	N/A	EMA	EMA	**Pass**

Module Test Status:	***Successful***

4.3.5 Module Name: Search Records

ARRAY OF VALUES								
Field Name	**Blank Entry**	**Numeric Entry**	**Alphabet Entry**	**@ Entry**	**Special Symbol**	**Expected Result**	**Observed Result**	**Test Result**
Employee ID	BFP	A	A	N/A	N/A	EMA	EMA	**Pass**

Module Test Status:	***Successful***

4.3.6 Module Name: Delete Records

ARRAY OF VALUES								
Field Name	**Blank Entry**	**Numeric Entry**	**Alphabet Entry**	**@ Entry**	**Special Symbol**	**Expected Result**	**Observed Result**	**Test Result**
Employee ID	BFP	A	A	N/A	N/A	EMA	EMA	**Pass**

Module Test Status:	***Successful***

4.3.7 Module Name: Allowance

ARRAY OF VALUES								
Field Name	**Blank Entry**	**Numeric Entry**	**Alphabet Entry**	**@ Entry**	**Special Symbol**	**Expected Result**	**Observed Result**	**Test Result**
Employee ID	BFP	A	A	N/A	N/A	EMA	EMA	**Pass**
First Name	BFP	N/A	D	N/A	N/A	EMA	EMA	**Pass**

Last Name:	BFP	N/A	D	N/A	N/A	EMA	EMA	**Pass**
Salary	BFP	D	N/A	N/A	N/A	EMA	EMA	**Pass**
Default Salary In Ringgits	BFP	D	N/A	N/A	N/A	EMA	EMA	**Pass**
Bonus and incentives for month of	BFP	S	S	N/A	N/A	EMA	EMA	**Pass**
Overtime	BFP	A	N/A	N/A	N/A	EMA	EMA	**Pass**
Medical	BFP	A	N/A	N/A	N/A	EMA	EMA	**Pass**
Lunch	BFP	A	N/A	N/A	N/A	EMA	EMA	**Pass**
Bonus	BFP	A	N/A	N/A	N/A	EMA	EMA	**Pass**
Total Allowance	BFP	D	N/A	N/A	N/A	EMA	EMA	**Pass**
Net Salary	BFP	D	N/A	N/A	N/A	EMA	EMA	**Pass**

Module Test Status:	***Successful***

4.3.8 Module Name: Deduction

ARRAY OF VALUES								
Field Name	**Blank Entry**	**Numeric Entry**	**Alphabet Entry**	**@ Entry**	**Special Symbol**	**Expected Result**	**Observed Result**	**Test Result**
Employee ID	BFP	A	A	N/A	N/A	EMA	EMA	**Pass**
First Name	BFP	N/A	D	N/A	N/A	EMA	EMA	**Pass**

Last Name:	BFP	N/A	D	N/A	N/A	EMA	EMA	**Pass**
Salary In Ringgits	BFP	D	N/A	N/A	N/A	EMA	EMA	**Pass**
Deduction for the month of	BFP	S	S	N/A	N/A	EMA	EMA	**Pass**
Gross Salary(Including Allowances)	BFP	D	N/A	N/A	N/A	EMA	EMA	**Pass**
Advance Pay	BFP	A	N/A	N/A	N/A	EMA	EMA	**Pass**
Leaves in Days	BFP	A	N/A	N/A	N/A	EMA	EMA	**Pass**
Government Tax in 6%	BFP	D	N/A	N/A	N/A	EMA	EMA	**Pass**
Other Tax	BFP	A	N/A	N/A	N/A	EMA	EMA	**Pass**
Total Deduction	BFP	D	N/A	N/A	N/A	EMA	EMA	**Pass**
Net Salary	BFP	D	N/A	N/A	N/A	EMA	EMA	**Pass**

Module Test Status:	***Successful***

4.3.9 Module Name: Make Payment

ARRAY OF VALUES								
Field Name	**Blank Entry**	**Numeric Entry**	**Alphabet Entry**	**@ Entry**	**Special Symbol**	**Expected Result**	**Observed Result**	**Test Result**
Employee ID	BFP	A	A	N/A	N/A	EMA	EMA	**Pass**
Date	BFP	S	S	N/A	N/A	EMA	EMA	**Pass**

Module Test Status:	*Successful*

4.3.10 Module Name: Expenditure Report

ARRAY OF VALUES								
Field Name	**Blank Entry**	**Numeric Entry**	**Alphabet Entry**	**@ Entry**	**Special Symbol**	**Expected Result**	**Observed Result**	**Test Result**
Display All	BFP	N/A	N/A	N/A	A	EMA	EMA	**Pass**
Expenditure By Month	BFP	S	S	N/A	N/A	EMA	EMA	**Pass**
Payment ID	BFP	D	N/A	N/A	N/A	EMA	EMA	**Pass**
Pay Slip ID	BFP	D	N/A	N/A	N/A	EMA	EMA	**Pass**
Employee ID	BFP	D	D	N/A	N/A	EMA	EMA	**Pass**
Payment For	BFP	D	N/A	N/A	D	EMA	EMA	**Pass**
Gross Salary	BFP	D	D	N/A	N/A	EMA	EMA	**Pass**
Allowances	BFP	D	D	N/A	N/A	EMA	EMA	**Pass**
Deductions	BFP	D	D	N/A	N/A	EMA	EMA	**Pass**
Net Salary	BFP	D	D	N/A	N/A	EMA	EMA	**Pass**

Module Test Status:	*Successful*

4.311 Module Name: Expenditure Report by Month

ARRAY OF VALUES								
Field Name	**Blank Entry**	**Numeric Entry**	**Alphabet Entry**	**@ Entry**	**Special Symbol**	**Expected Result**	**Observed Result**	**Test Result**
Date	BFP	S	S	N/A	N/A	EMA	EMA	**Pass**

Module Test Status:	*Successful*

4.3.12Module Name: Logout

ARRAY OF VALUES								
Field Name	**Blank Entry**	**Numeric Entry**	**Alphabet Entry**	**@ Entry**	**Special Symbol**	**Expected Result**	**Observed Result**	**Test Result**
Logout	BFP	N/A	N/A	N/A	N/A	EMA	EMA	**pass**

Module Test Status:	*Successful*

4.4 Junit Testing

Joint testing is an automated testing which is fast in comparison to the manual testing and perform by the system. It will test all the class individually of the IBM Payroll System.

4.4.1 Admin Account

In the admin account username and password are required so it will test suit will test both class together.

adminaccount.java

```
package adminlogin;

public class adminaccount {

    private String username;
    private String password;


       public adminaccount(String username,String password){
          this.username = username;
          this.password=password;
       }

       public String UserName(){
          System.out.println(username);
          return username;
       }

       public String Password(){

          System.out.println(password);
          return password;
       }
    }
```

Fig 4.4.1 admin class

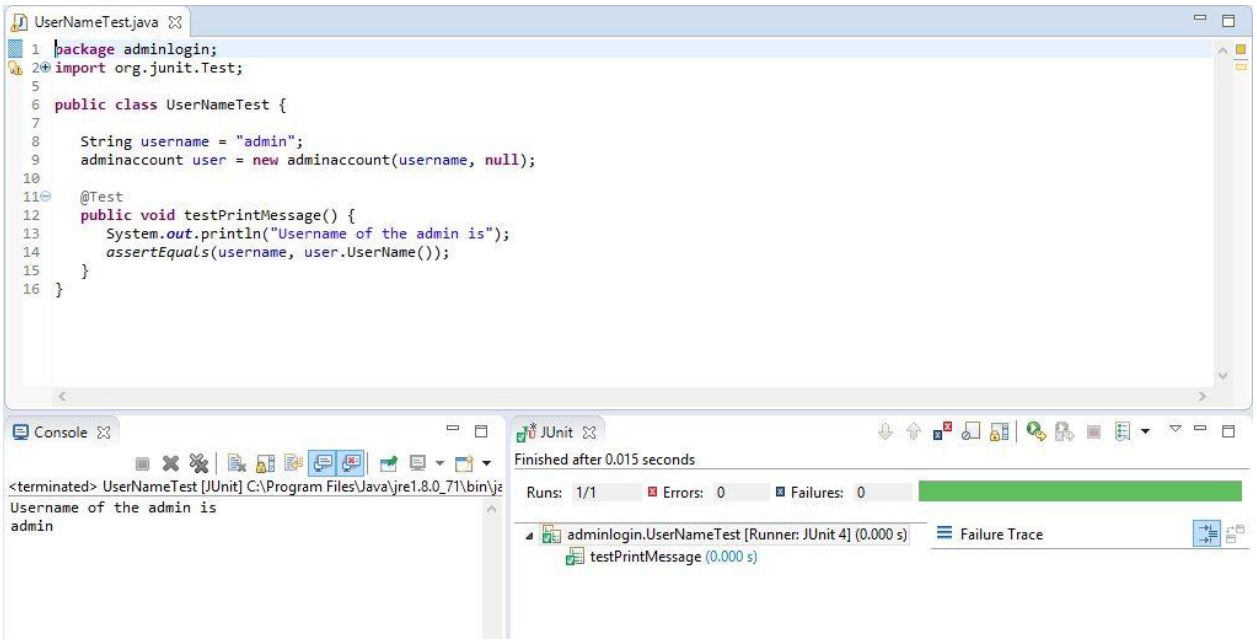


Fig 4.4.2 Test admin's username

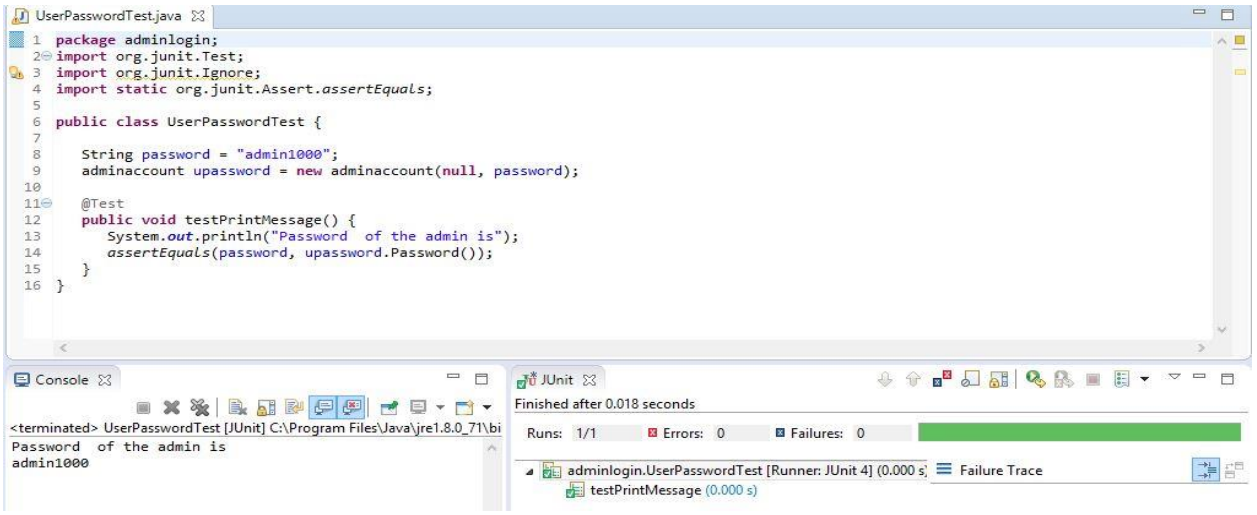


Fig 4.4.3 Test password

Here there are two different test so test suit will be needed to run both test classes together to check the username and password of the admin.

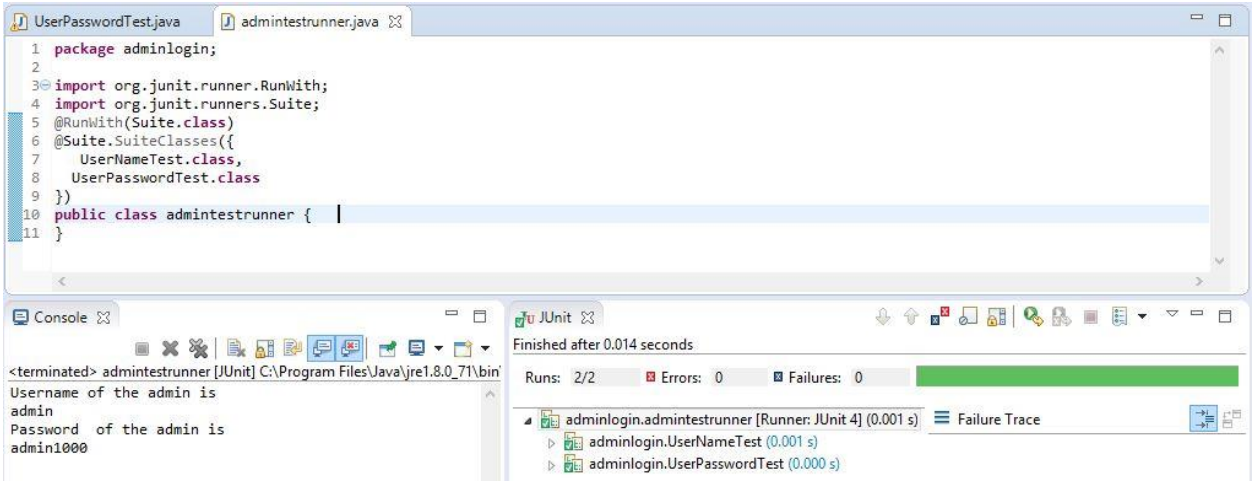


Fig 4.4.4 test suit admin login test

Junit Test	Description	Expected result	Actual result	Time elapsed
Admin Login	To check the username and password of the admin	Username and password are valid	Test passed	0.01 and 0.000 second

4.4.5 Manager Login

It will check the username and password of the manager account using test suit class.

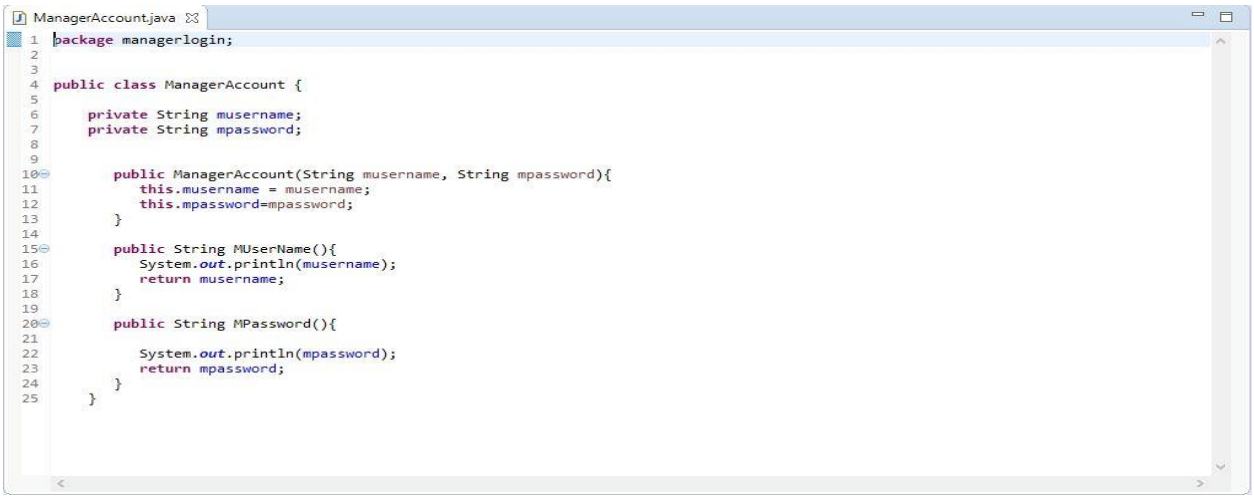


Fig 4.4.5 manager class

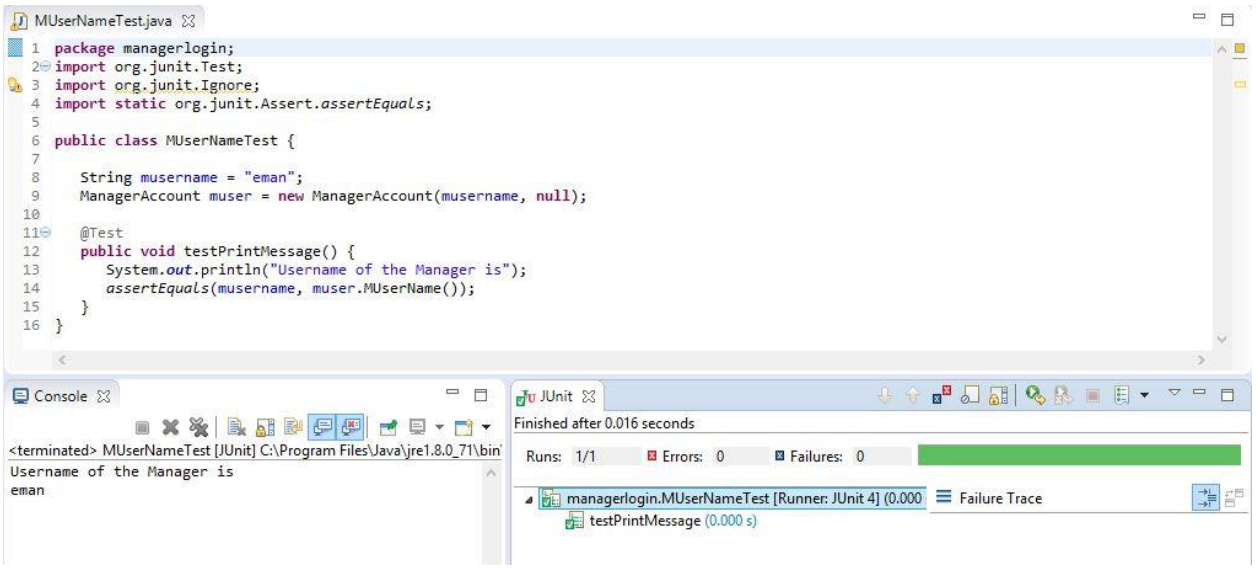


Fig 4.4.6 Test username of the manager

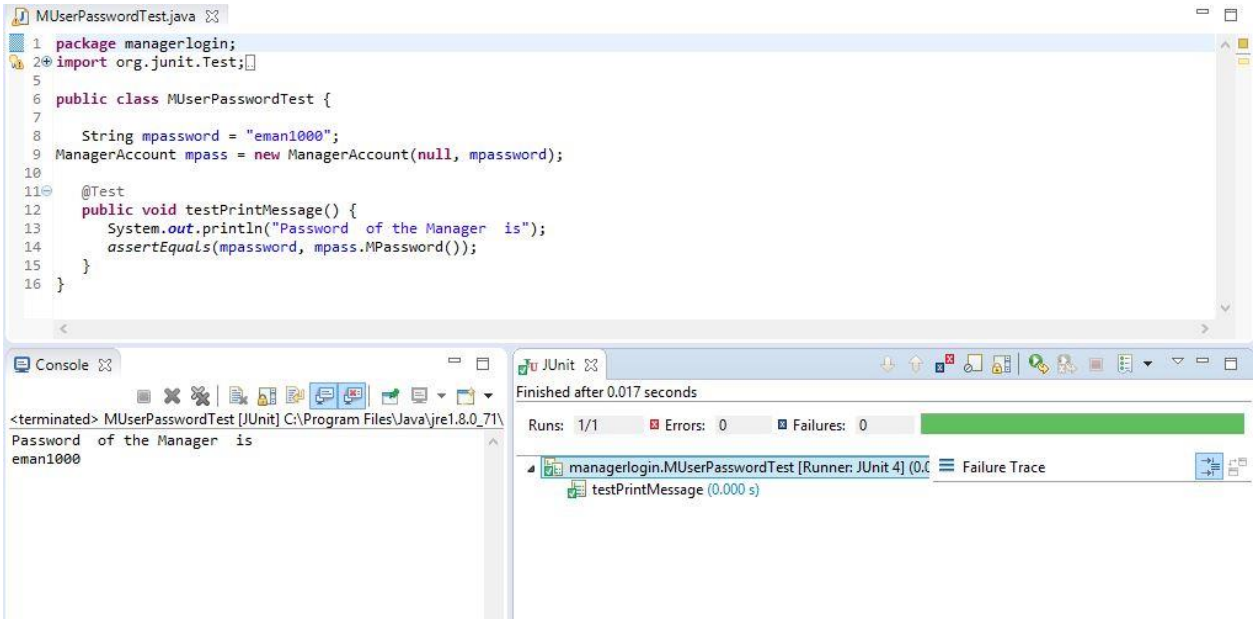


Fig 4.4.7 test password of the manager

Here is also two different test classes so test suit will be needed so check the data.

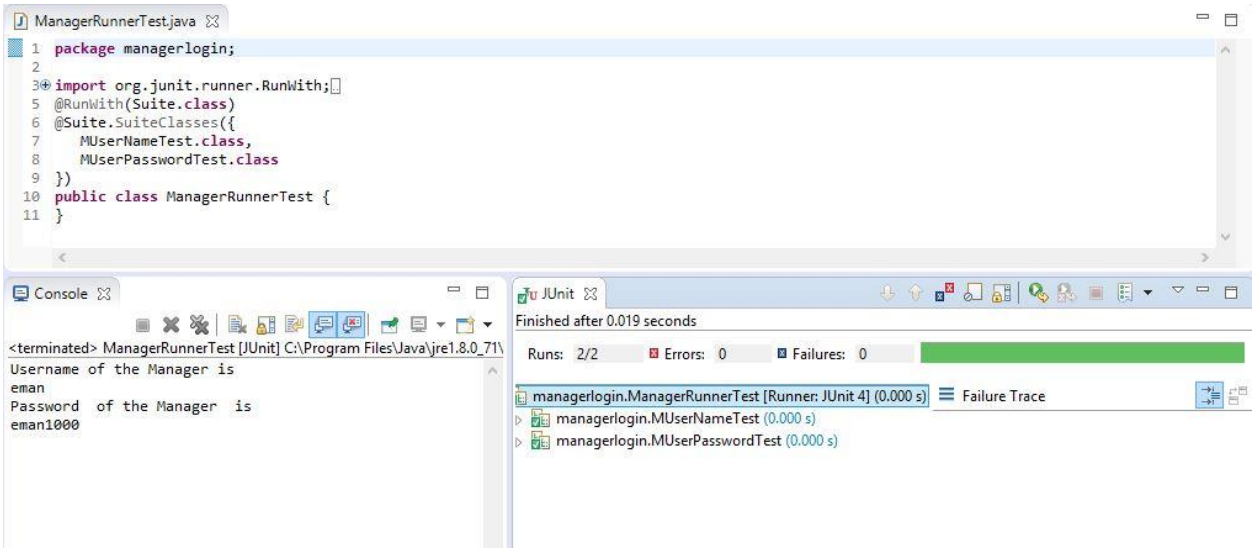


Fig 4.4.8 Test suit of manager class

Junit Test	Description	Expected result	Actual result	Time elapsed
Manager Login	To check the username and password of the admin	Username and password are valid	Test passed	0.00 and 0.000 second

4.5.1 Add Employee Class

Here system will test all the array values of the text box which are entered by the admin.

Fig 4.5.1 Actual format of the value which is needed to fill in the text box.

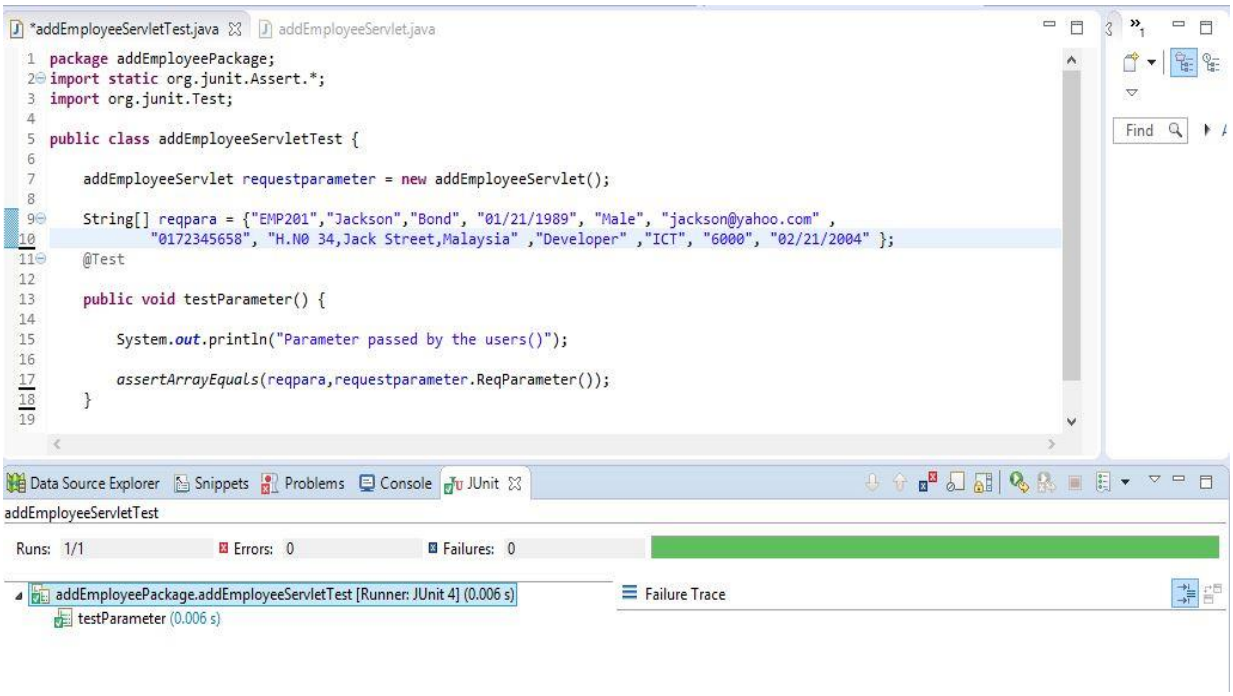


Fig 4.5.2 test the array values which are entered by the admin.

Junit Test	Description	Expected result	Actual result	Time elapsed
Add Employee Details	To check all the input parameter which are passed by the admin .It will check that these data type are valid format or not.	All the inputs are valid	Test passed	0.006 seconds

Test all the conditions for the input parameter. For example if user will not enter any input in a form else he enters wrong data type.

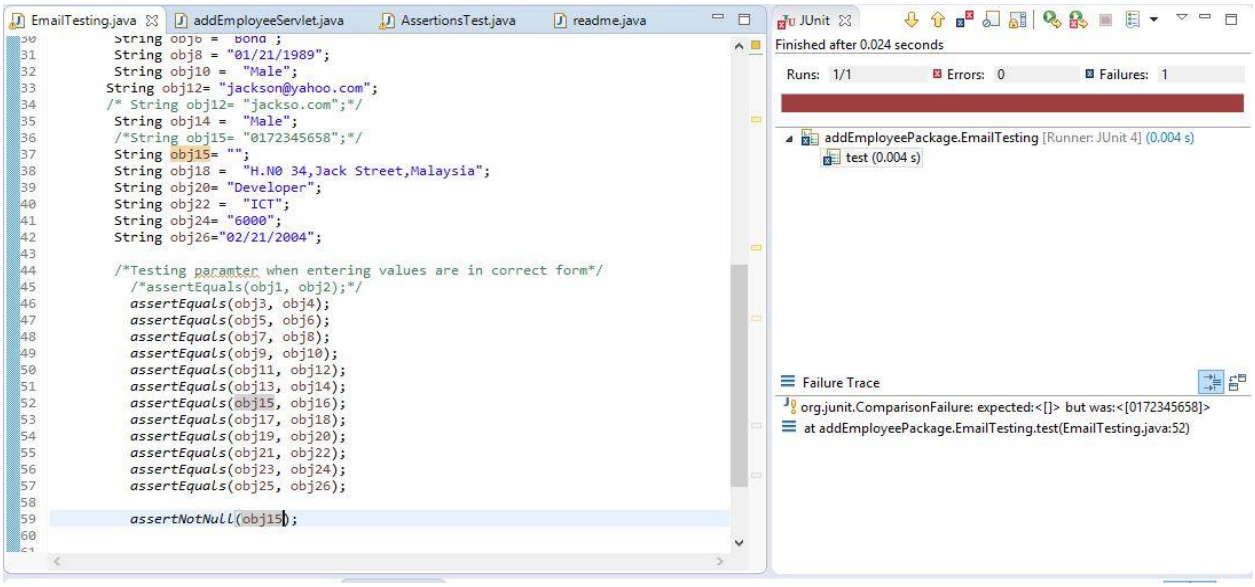


Fig 4.5.3 test blank value

Junit Test	Description	Expected result	Actual result	Time elapsed
Mobile number	Check mobile number column has blank value or not which is obj15	Mobile has not blank value	Test Failed (Admin did not enter any mobile number)	0.004 seconds

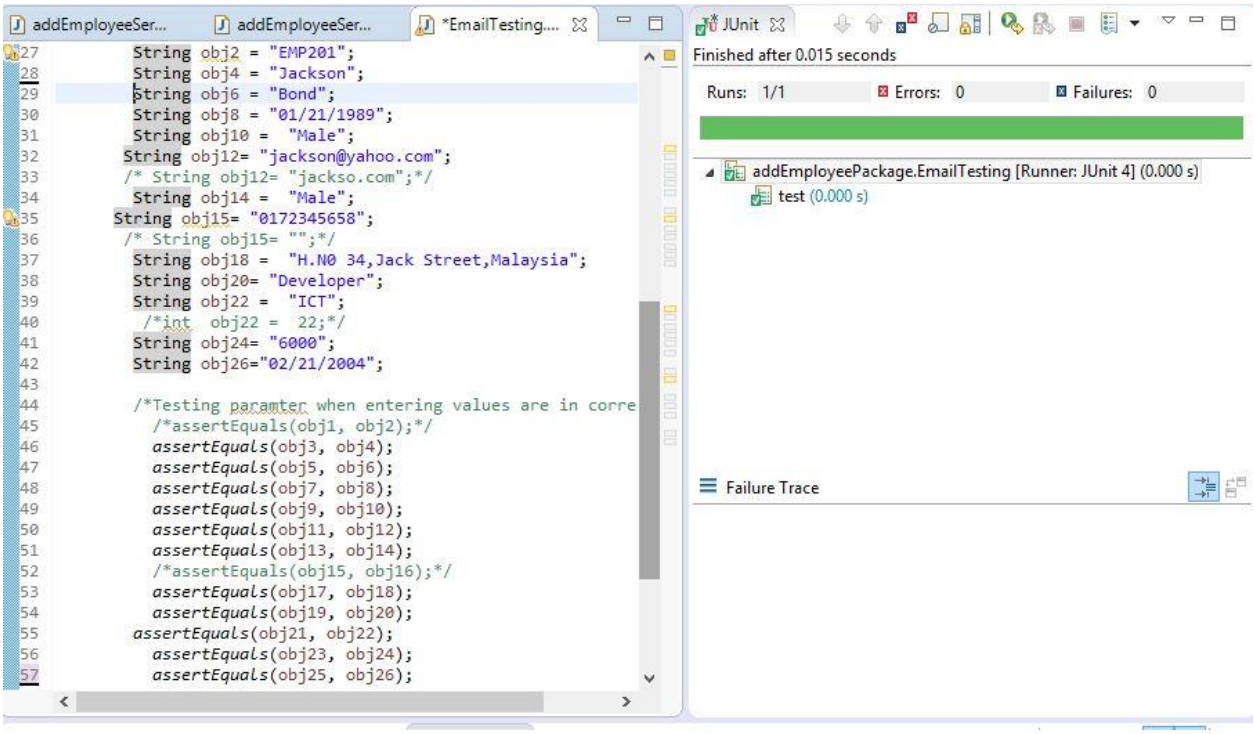


Fig 4.5.3 test again after changing some value

Junit Test	Description	Expected result	Actual result	Time elapsed
Mobile Number	Check mobile number column	Admin has entered correct mobile number	Test passed	0.000 second

Validation on Email id which is obj12

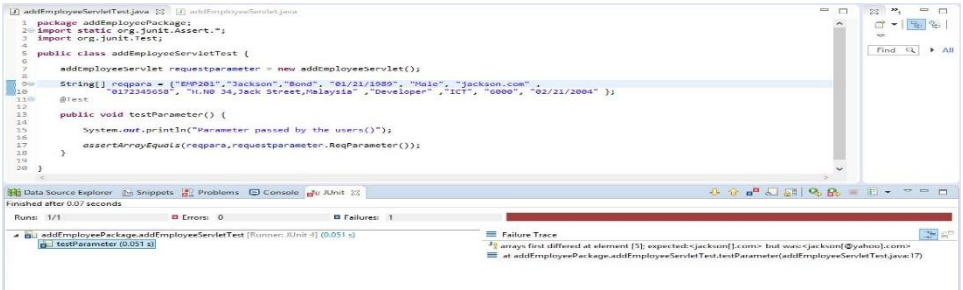


Fig 4.5.4 wrong email id

Junit Test	Description	Expected result	Actual result	Time elapsed
Email validation	Check that admin has entered correct email id format	Correct email id	Test Failed	0.0051 seconds

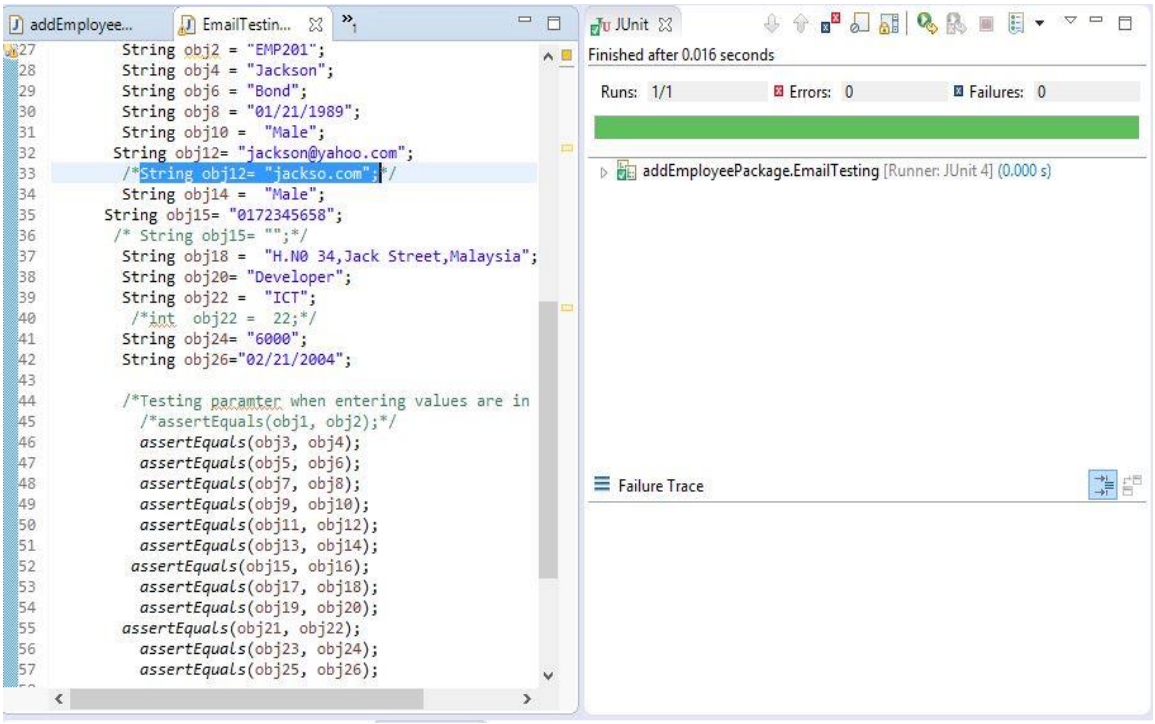


Fig 4.5.5 test email id

Junit Test	Description	Expected result	Actual result	Time elapsed
Validation Email Id	Check that email id is a valid form	Email id is valid	Test passed	0.000 second

4.6.1Allowance Amount

It will test the actual allowance amount and system generated allowance.

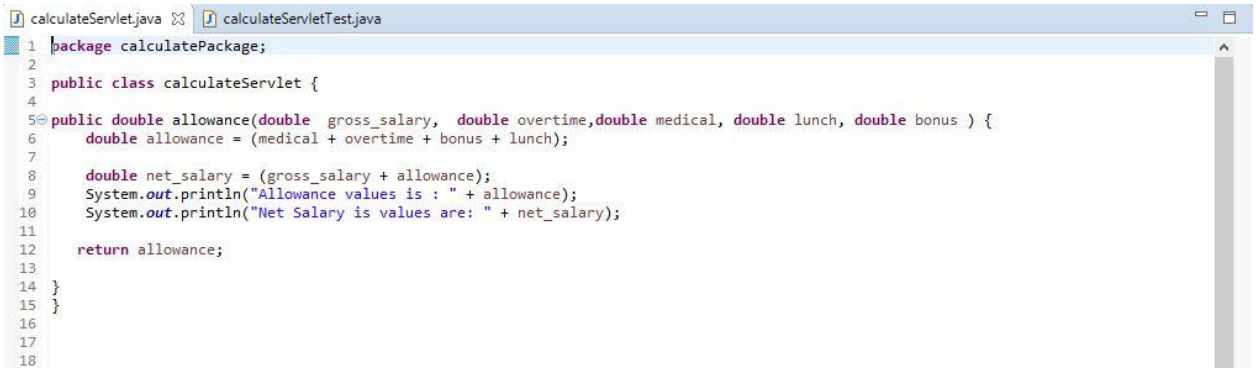


Fig 4.6.1 Array values of the allowance

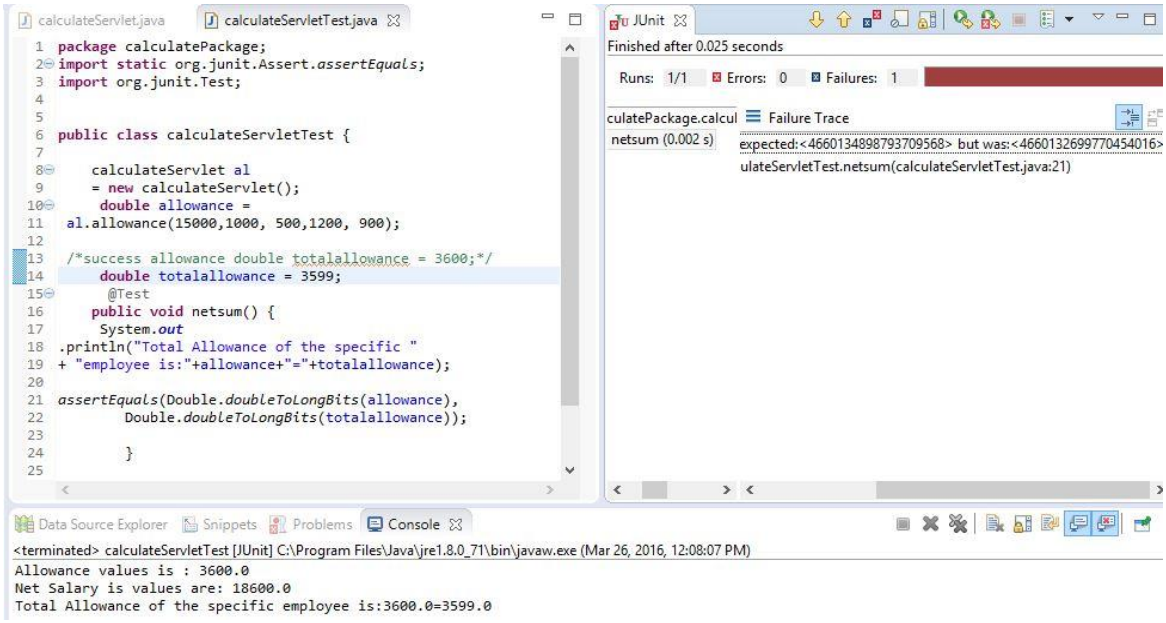


Fig 4.6.2 Comparing two values

Junit Test	Description	Expected result	Actual result	Time elapsed
Employee' Allowance	Calculation of net salary and allowance with comparing actual amount.	Allowance value has successfully calculated.	Test Failed (system generated value is different actual value)	0.002 second

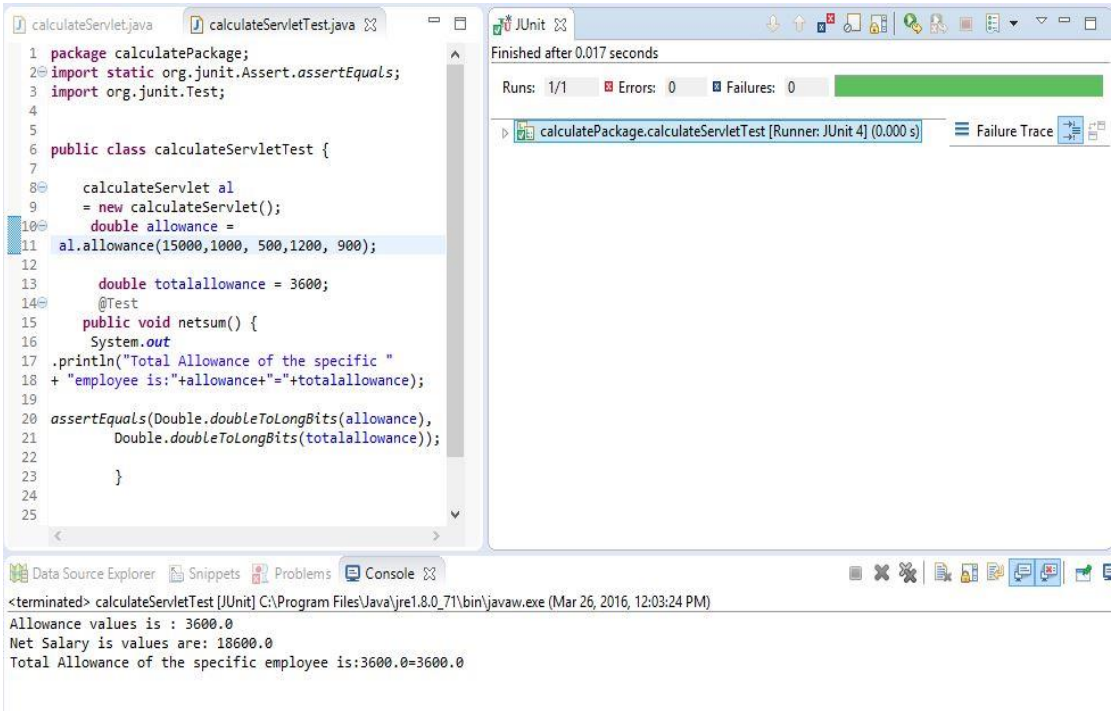
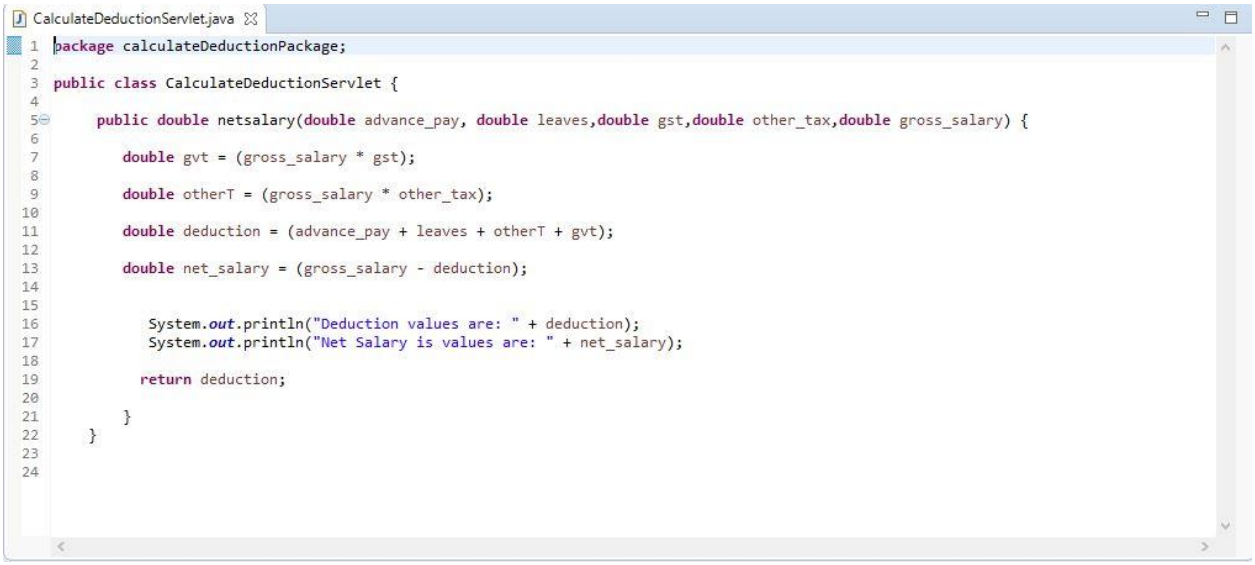


Fig 4.6.3 Test allowance of the employee

Junit Test	Description	Expected result	Actual result	Time elapsed
Employee' Allowance	Calculation of net salary and allowance with comparing actual amount.	Allowance value has successfully calculated.	Test passed (Actual value is same as system generated value)	0.000 second

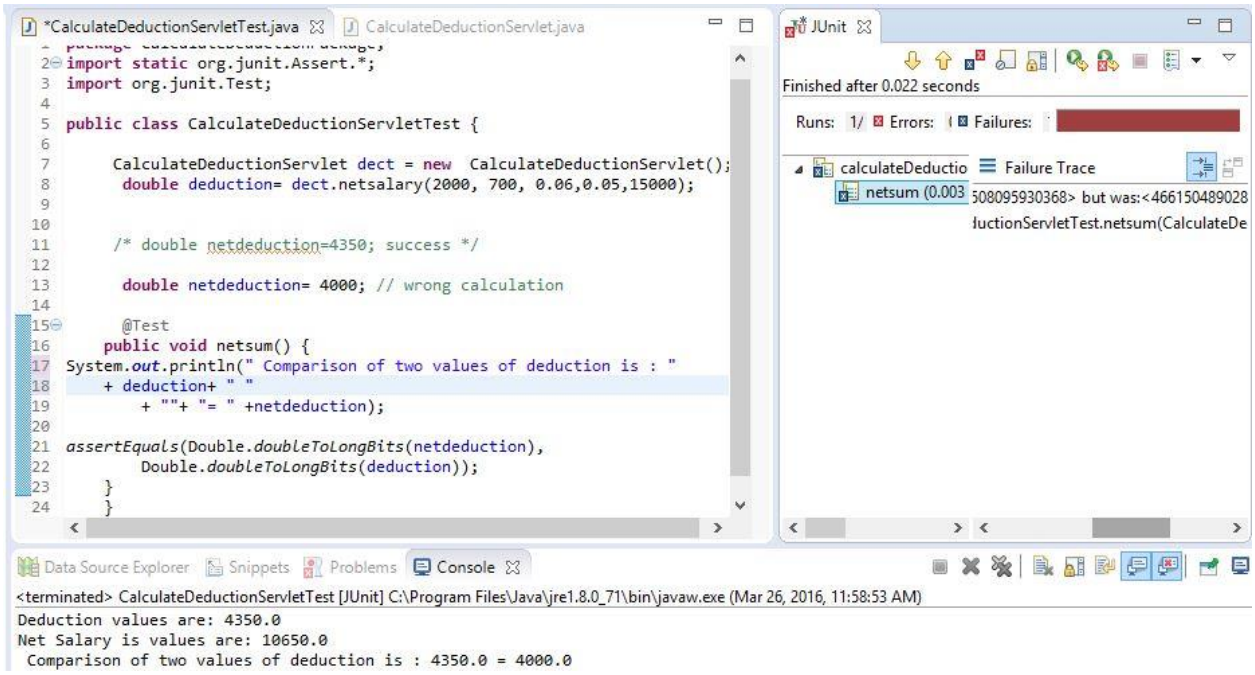
4.7.1 Deduction Amount

It will test the deduction amount system generated with the actual amount.



```
CalculateDeductionServlet.java

1  package calculateDeductionPackage;
2
3  public class CalculateDeductionServlet {
4
5      public double netsalary(double advance_pay, double leaves,double gst,double other_tax,double gross_salary) {
6
7         double gvt = (gross_salary * gst);
8
9         double otherT = (gross_salary * other_tax);
10
11        double deduction = (advance_pay + leaves + otherT + gvt);
12
13        double net_salary = (gross_salary - deduction);
14
15
16           System.out.println("Deduction values are: " + deduction);
17           System.out.println("Net Salary is values are: " + net_salary);
18
19          return deduction;
20
21        }
22    }
23
24
```

Fig 4.7.1 Array values of the deduction which have to perform operation.



```
*CalculateDeductionServletTest.java   CalculateDeductionServlet.java

2  import static org.junit.Assert.*;
3  import org.junit.Test;
4
5  public class CalculateDeductionServletTest {
6
7      CalculateDeductionServlet dect = new  CalculateDeductionServlet();
8       double deduction= dect.netsalary(2000, 700, 0.06,0.05,15000);
9
10
11     /* double netdeduction=4350; success */
12
13      double netdeduction= 4000; // wrong calculation
14
15      @Test
16    public void netsum() {
17 System.out.println(" Comparison of two values of deduction is : "
18     + deduction+ " "
19        + ""+ "= " +netdeduction);
20
21 assertEquals(Double.doubleToLongBits(netdeduction),
22        Double.doubleToLongBits(deduction));
23    }
24    }
```

```
JUnit
Finished after 0.022 seconds
Runs: 1/ Errors: Failures:
calculateDeductio   Failure Trace
netsum (0.003   508095930368> but was:<466150489028
                fuctionServletTest.netsum(CalculateDe
```

```
Data Source Explorer  Snippets  Problems  Console
<terminated> CalculateDeductionServletTest [JUnit] C:\Program Files\Java\jre1.8.0_71\bin\javaw.exe (Mar 26, 2016, 11:58:53 AM)
Deduction values are: 4350.0
Net Salary is values are: 10650.0
 Comparison of two values of deduction is : 4350.0 = 4000.0
```

Fig 4.7.2 check the actual value and system generated value.

Junit Test	Description	Expected result	Actual result	Time elapsed
Deduction	Comparing the actual amount of deduction with the system generated.	System generated deduction value is same as actual value.	Test Failed	0.002 second

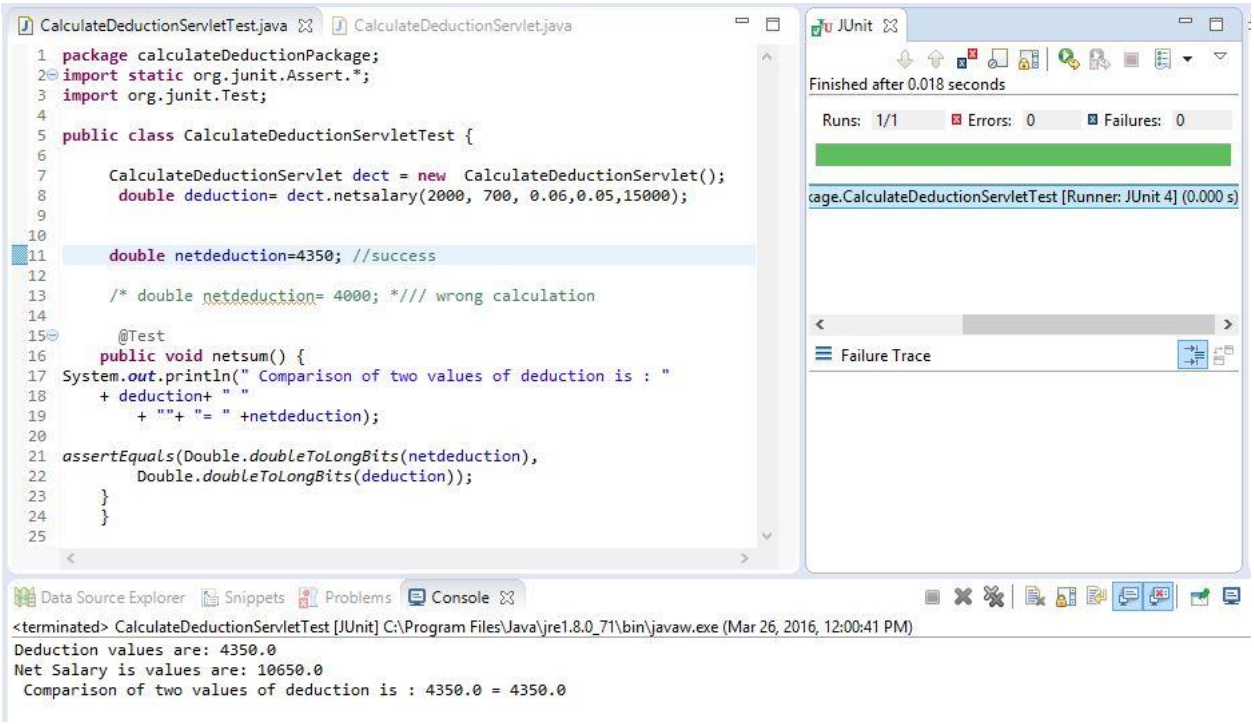


Fig 4.7.3 check the actual deduction value and system generated value.

Junit Test	Description	Expected result	Actual result	Time elapsed
Deduction	Calculate deduction amount and comparing with actual amount	System generated deduction value is same as actual value.	Test passed	0.000 second

4.8.1Expenditure Report By Month

It will test net salary after all the allowance and deduction from the salary of an employee for a particular month.

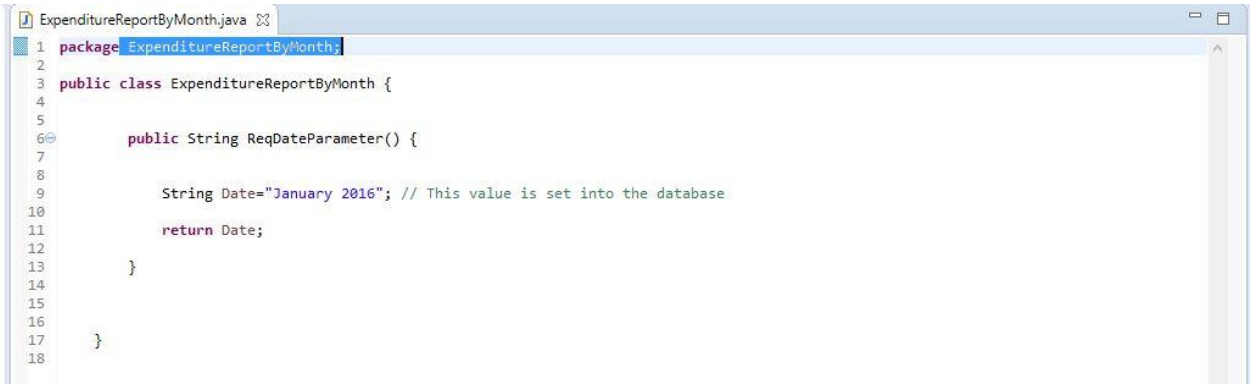


Fig 4.8.1 Array value of the expenditure class for a specific month

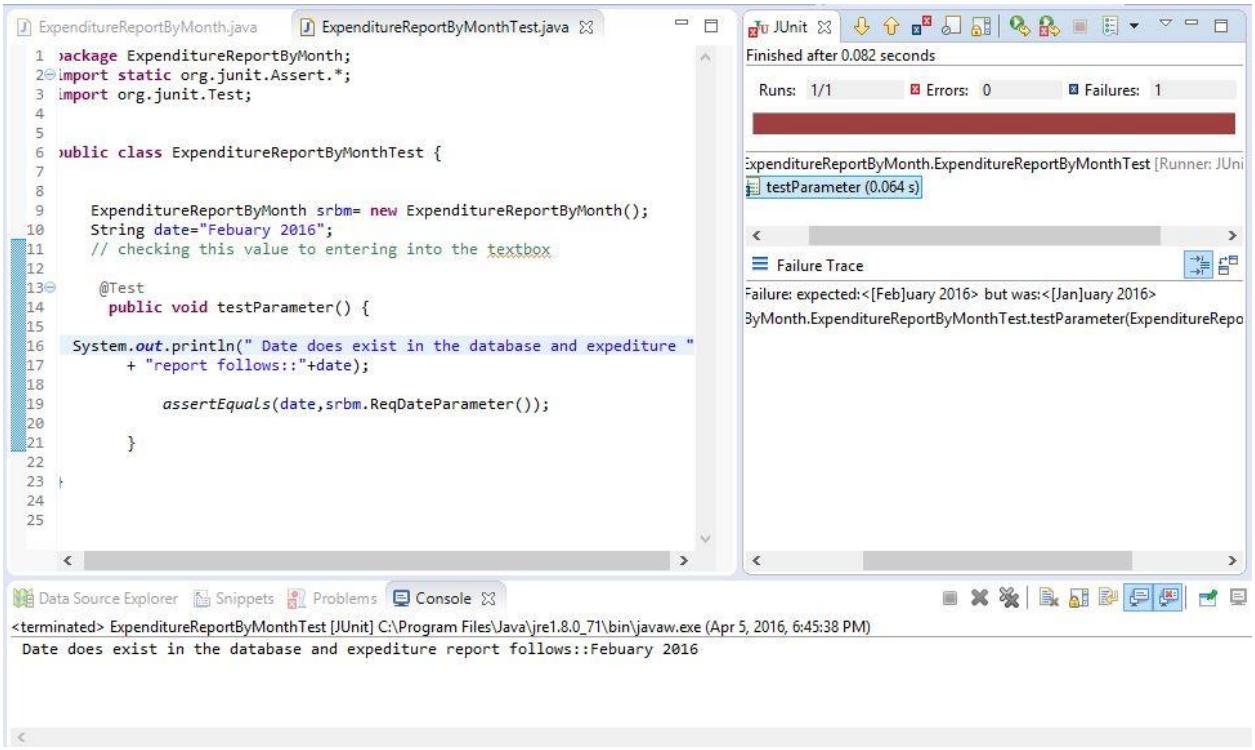


Fig 4.8.2 test the a specific date

Junit Test	Description	Expected result	Actual result	Time elapsed
Expenditure ReportBy Month	Calculate the expenditure for a specific date.	Date is exist in the database for a specific employee and expenditure value is something	Test Failed (Date does not exist which is entered by the admin)	0.0064 second

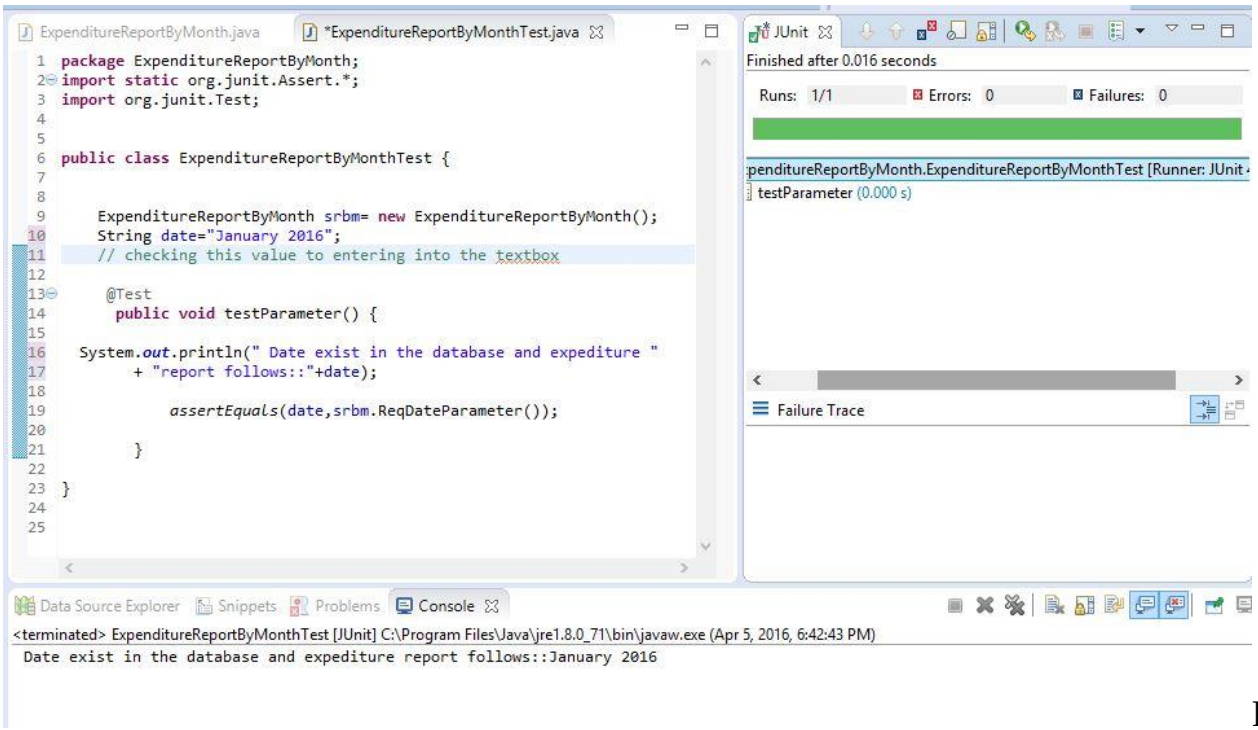


Fig 4.8.3

Test validate date

Junit Test	Description	Expected result	Actual result	Time elapsed
Expenditure ReportBy Month	Check the expenditure report of a specific date.	Date is exist in the database for a specific employee	Test passed (Date exist which is entered by the admin)	0.0064 second

4.9.1 Expenditure Report

It will calculate the expenditure of a particular employee or all the employees.

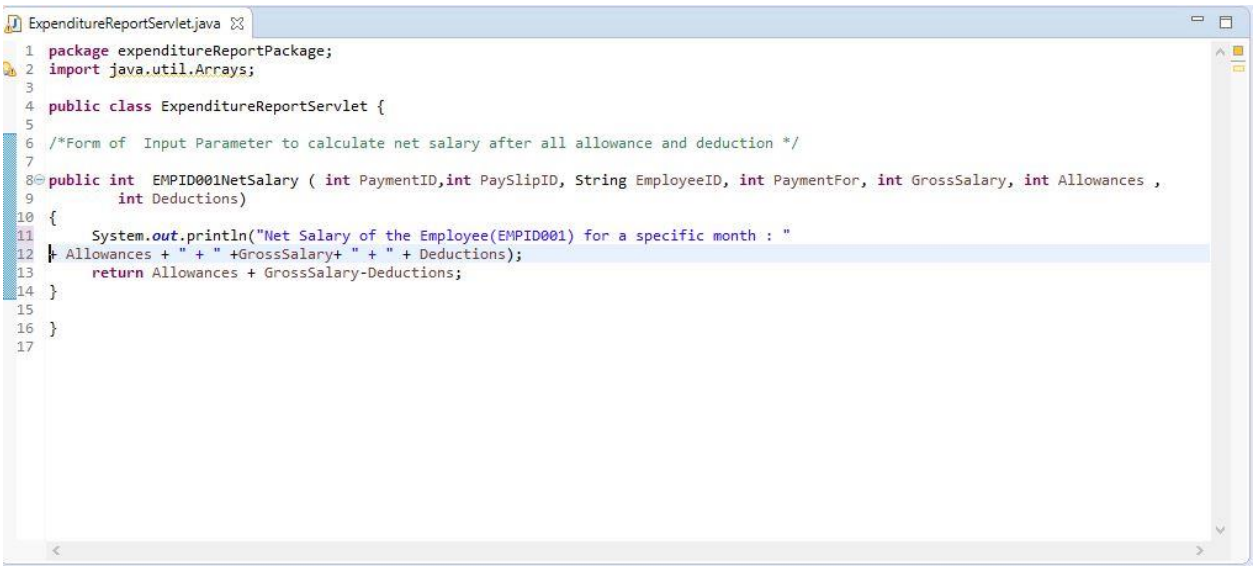


Fig 4.9.1 Array values of the expenditure

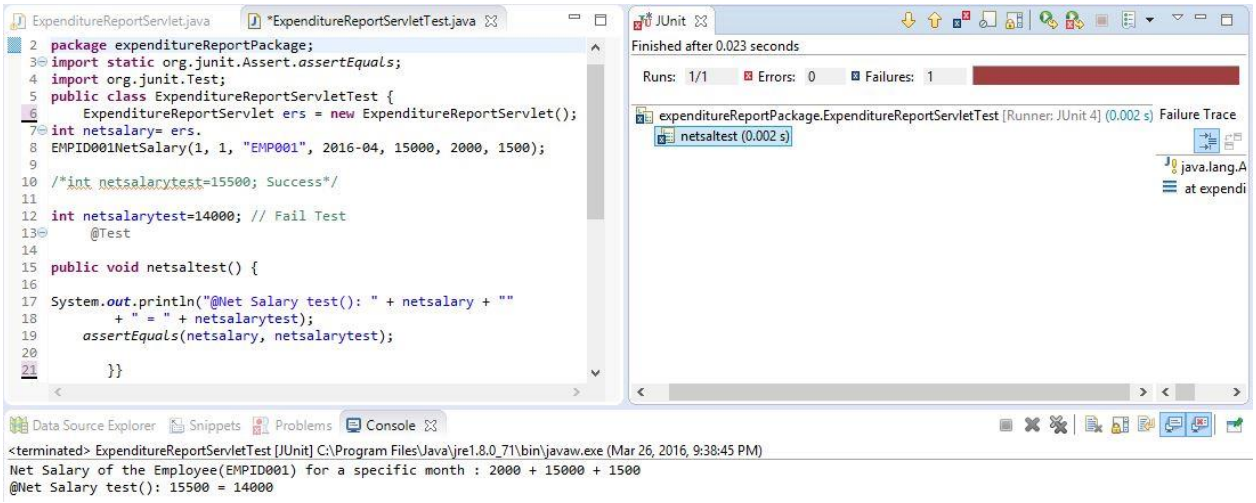


Fig 4.9.2 test net salary

Junit Test	Description	Expected result	Actual result	Time elapsed

Expenditure Report	Comparing two salaries where one is system generated and other is actual value	Both values are same	Test Failed (system generated and actual value are different)	0.002 second

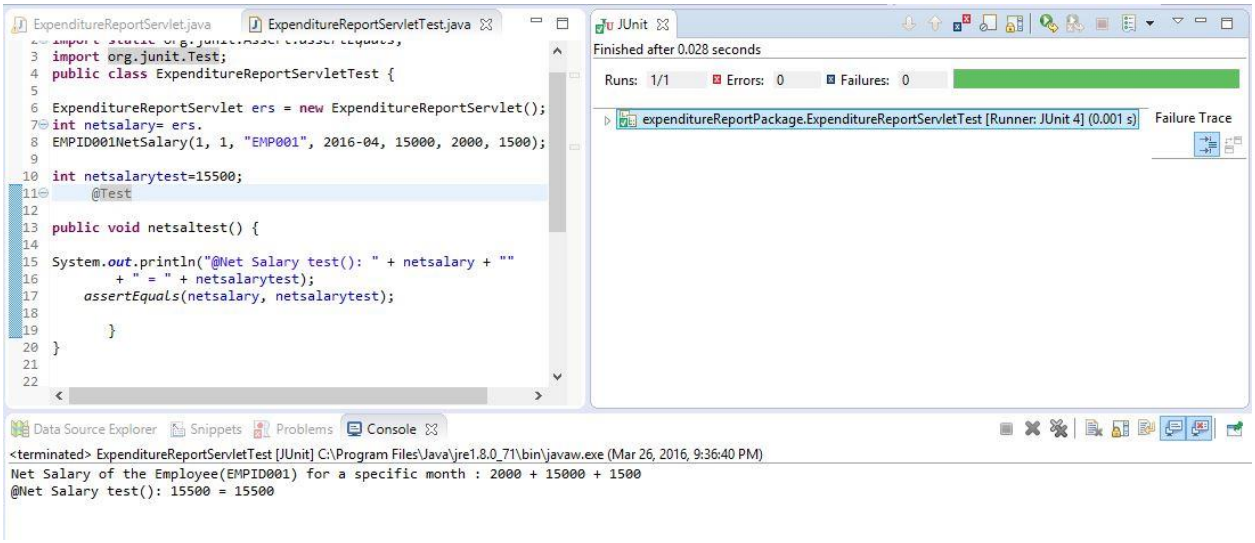


Fig 4.9.3testing two values

Junit Test	Description	Expected result	Actual result	Time elapsed
Expenditure Report	Check the expenditure report auto generated with actual net salary	Both values are same	Test Passed (system generated and actual value are same)	0.001 second

4.10.1 Search Records

It will try to find the specific employee with the help of the employee Id and will return some results.

SearchRecords.java

```
1  package searchrecords;
2
3
4  public class SearchRecords {
5      public String ReqParameterEmpId() {
6
7
8          String EmpId="Emp001";
9
10         return EmpId;
11
12     }
13
14
15
16 }
17
```

Fig 4.10.1Array value of the employee

SearchRecords.java | SearchRecordsTest.java

```
1  package searchrecords;
2  import static org.junit.Assert.*;
3  import org.junit.Test;
4
5  public class SearchRecordsTest {
6
7      SearchRecords sr= new SearchRecords();
8
9      String empid="Emp009";
10     @Test
11      public void testParameter() {
12
13
14          System.out.println("Empoyee ID Emp009 does not exist in the database");
15
16            assertEquals(empid,sr.ReqParameterEmpId());
17
18        }
19
20 }
21
```

JUnit

Finished after 0.031 seconds

Runs: 1/1 Errors: 0 Failures: 1

searchrecords.SearchRecordsTest [Runner: JUnit 4]
testParameter (0.006 s)

Failure Trace

org.junit.ComparisonFailure: expected:<Emp00[9]> but
at searchrecords.SearchRecordsTest.testParameter(Sea

Data Source Explorer | Snippets | Problems | Console

```
<terminated> SearchRecordsTest [JUnit] C:\Program Files\Java\jre1.8.0_71\bin\javaw.exe (Apr 5, 2016, 10:54:48 AM)
Empoyee ID Emp009 does not exist in the database
```

Fig 4.10.2 check the employee id

Junit Test	Description	Expected result	Actual result	Time elapsed
Search Employee Id	Check the that specific employee id exist or not	Email Id exist	Test Failed (Email Id does not exist in the database)	0.006 second

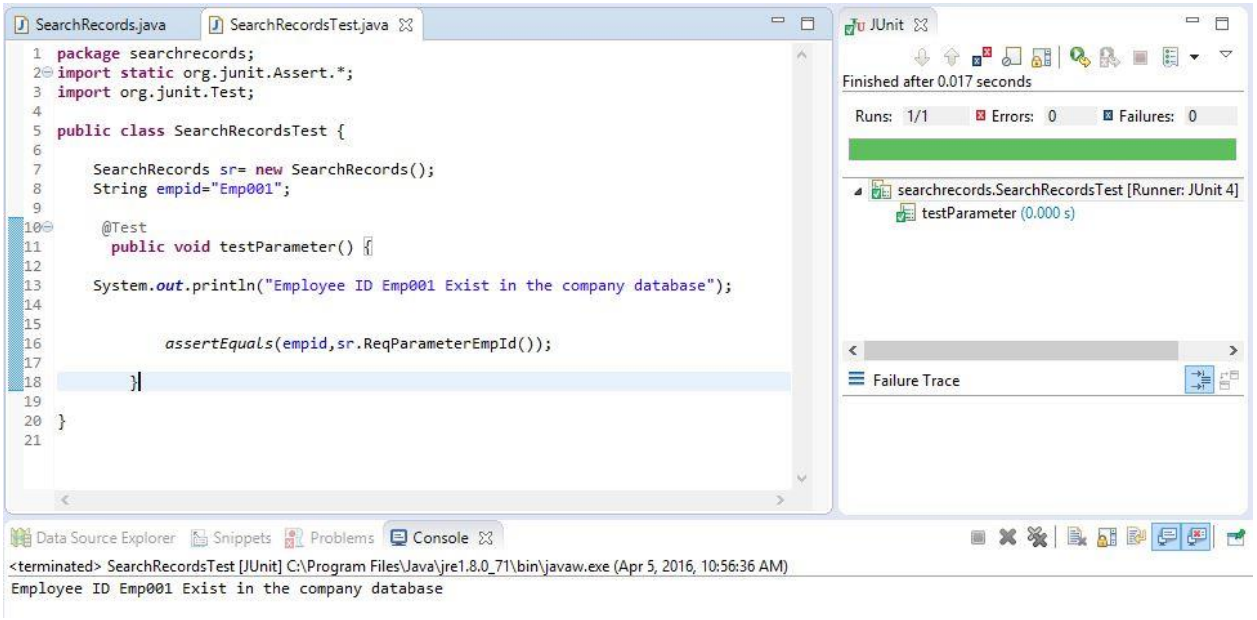


Fig 4.10.3 Test Employee Id exist or not

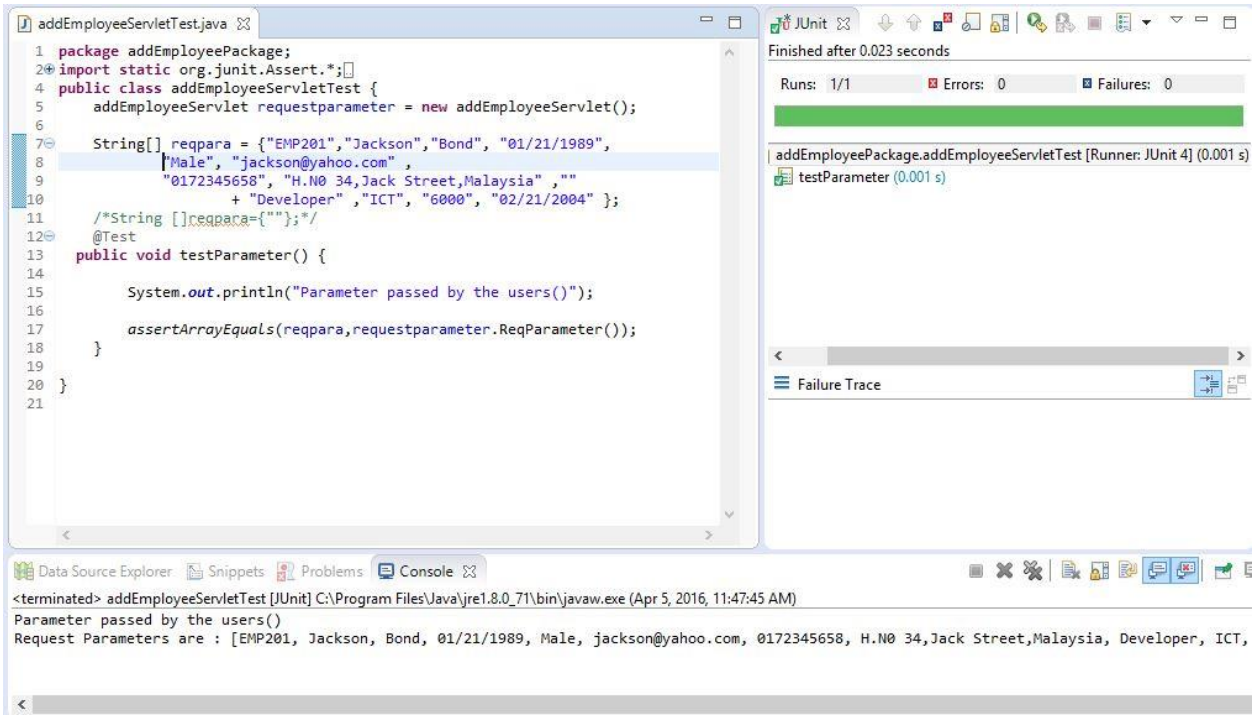
Junit Test	Description	Expected result	Actual result	Time elapsed
Search Employee Id	Check the that specific employee id exist or not	Email Id exist	Test Passed (Email Id exist in the database)	0.000 second

4.11.1 Update Employee

Test after the updating employee details.

```
package UpdateEmployees;

import java.util.Arrays;

public class UpdateEmployee {

	public String[] ReqParameterUpdate() {

String[] requestparameterforupdate = { "EMP201","Jackson","Bond", "01/21/1989", "Male", "jackson@yahoo.com" ,
		"0172345658", "H.N0 34,Jack Street,Malaysia" ,"Developer" ,"ICT", "6000", "02/21/2004" };
    // Employee Id fixed

System.out.println("Request Parameters are for update : "   + Arrays.toString(requestparameterforupdate ));

		return requestparameterforupdate;


	}

    }
```

Fig 4.11.1 exist array values of the employee



```
package addEmployeePackage;
import static org.junit.Assert.*;
public class addEmployeeServletTest {
    addEmployeeServlet requestparameter = new addEmployeeServlet();

    String[] reqpara = {"EMP201","Jackson","Bond", "01/21/1989",
            "Male", "jackson@yahoo.com" ,
            "0172345658", "H.N0 34,Jack Street,Malaysia" ,""
                    + "Developer" ,"ICT", "6000", "02/21/2004" };
    /*String []reqpara={""};*/
    @Test
  public void testParameter() {

        System.out.println("Parameter passed by the users()");

        assertArrayEquals(reqpara,requestparameter.ReqParameter());
    }

}
```

```
Finished after 0.023 seconds
Runs: 1/1   Errors: 0   Failures: 0
addEmployeePackage.addEmployeeServletTest [Runner: JUnit 4] (0.001 s)
testParameter (0.001 s)
Failure Trace
```

```
<terminated> addEmployeeServletTest [JUnit] C:\Program Files\Java\jre1.8.0_71\bin\javaw.exe (Apr 5, 2016, 11:47:45 AM)
Parameter passed by the users()
Request Parameters are : [EMP201, Jackson, Bond, 01/21/1989, Male, jackson@yahoo.com, 0172345658, H.N0 34,Jack Street,Malaysia, Developer, ICT,
```

4.11.3testing before the updating records of the employee

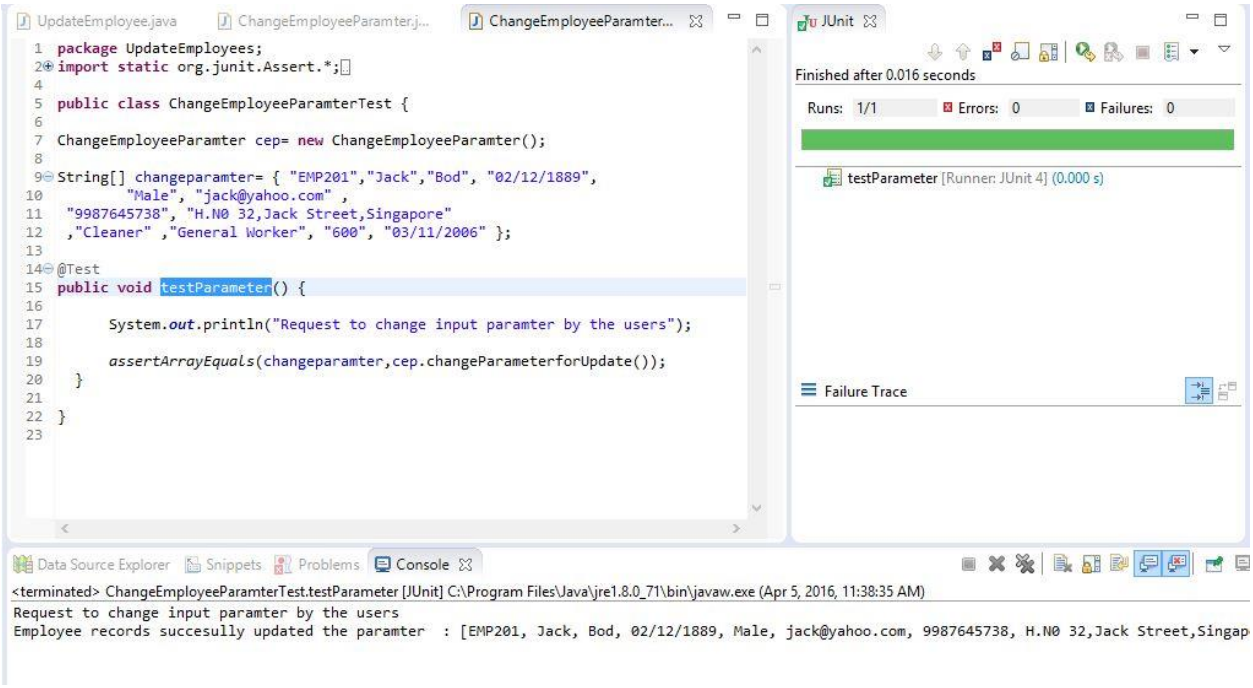
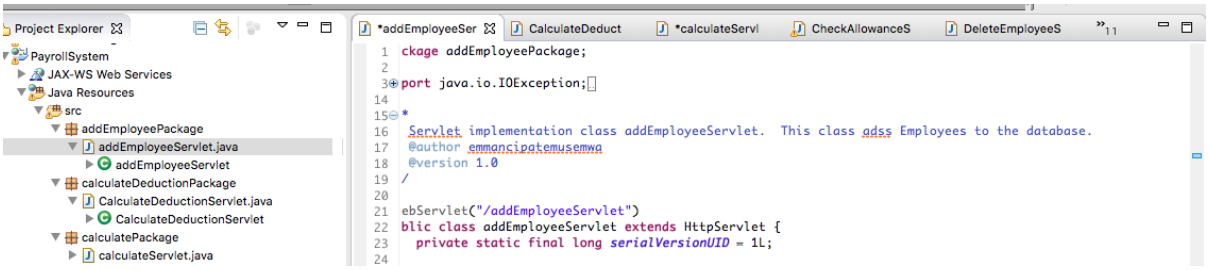


Fig 4.11.4 testing after updating some value of the employees

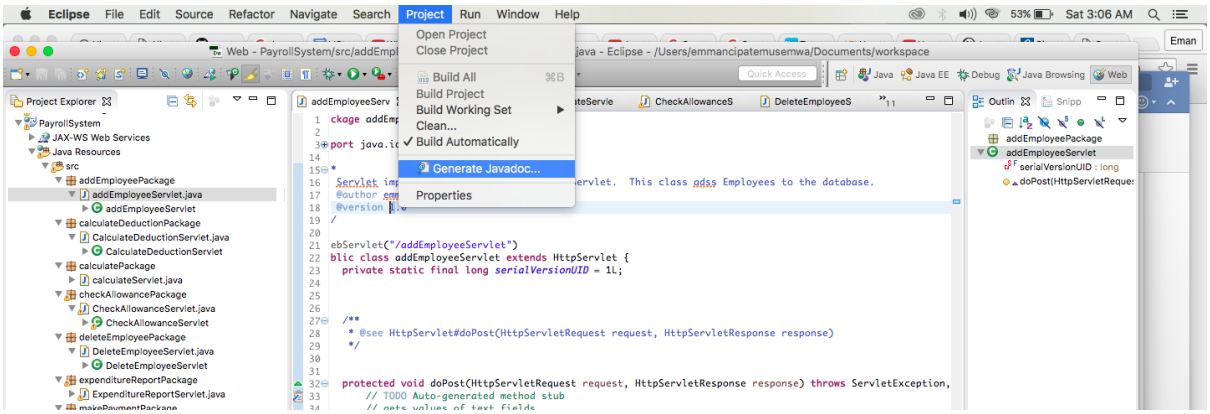
Junit Test	Description	Expected result	Actual result	Time elapsed
Update Records of the Employee	Update the some value of the employee	Employees record updated	Test Passed	0.000 second

5.0 Javadoc

Javadoc is a tools which is used to generate the API documentation which help during the developing an application and it is HTML format and it generates doc comments from source code. The following example of generating javadoc from the eclipse software.



5.1 Generating Java doc



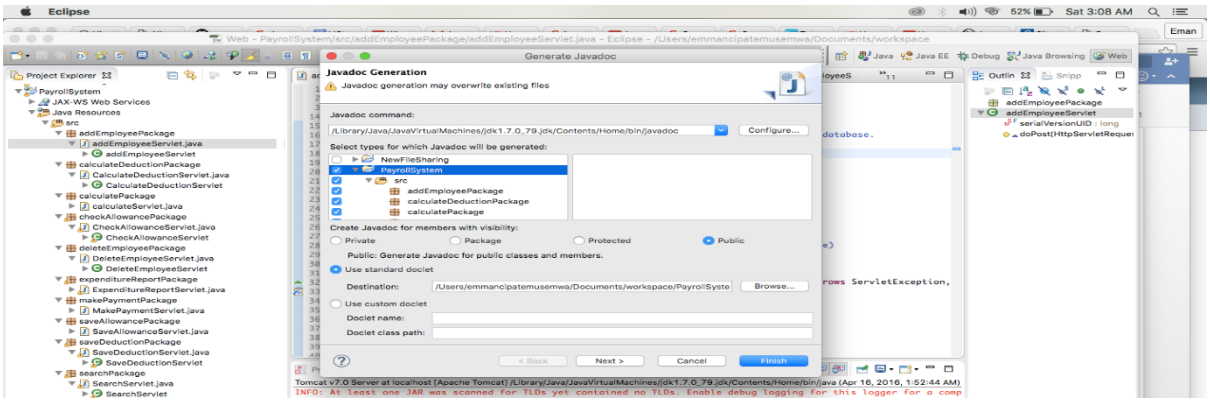
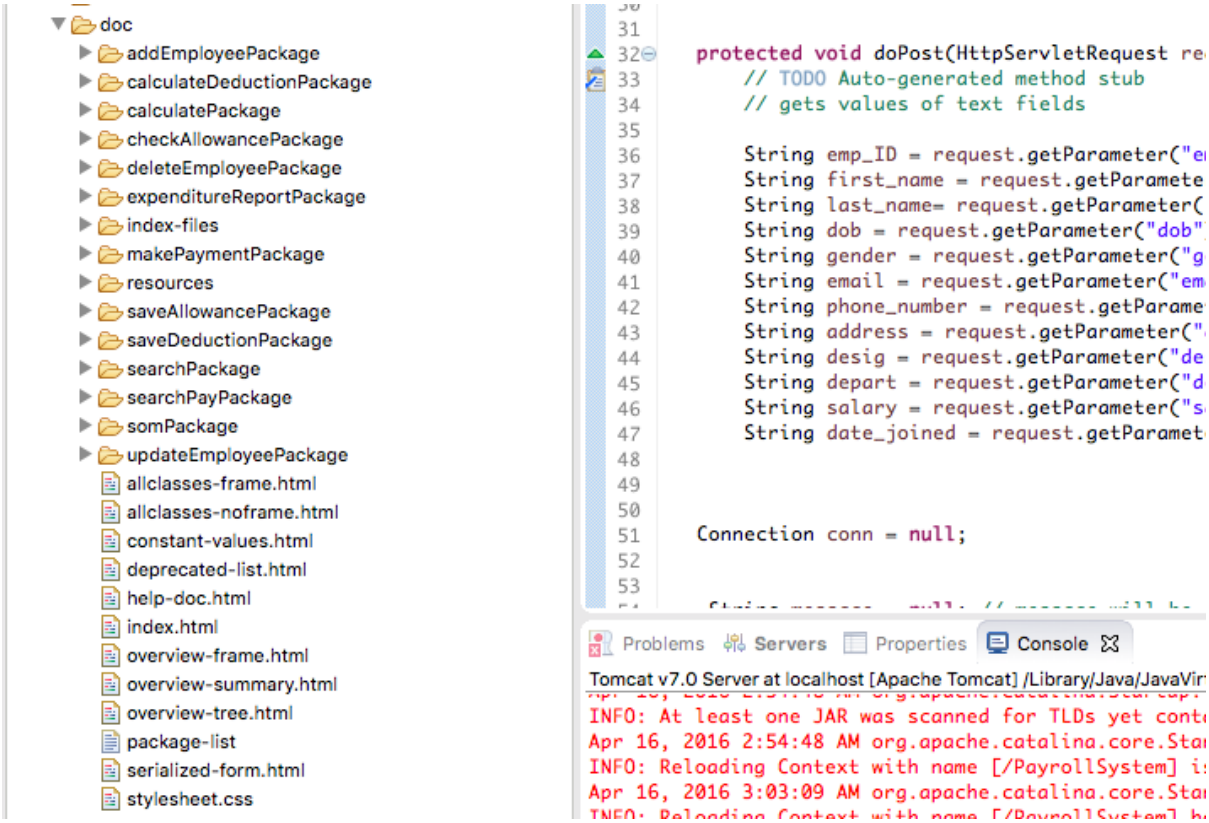


Fig 5.2 Generating Javadoc



5.3 Generated Java Doc

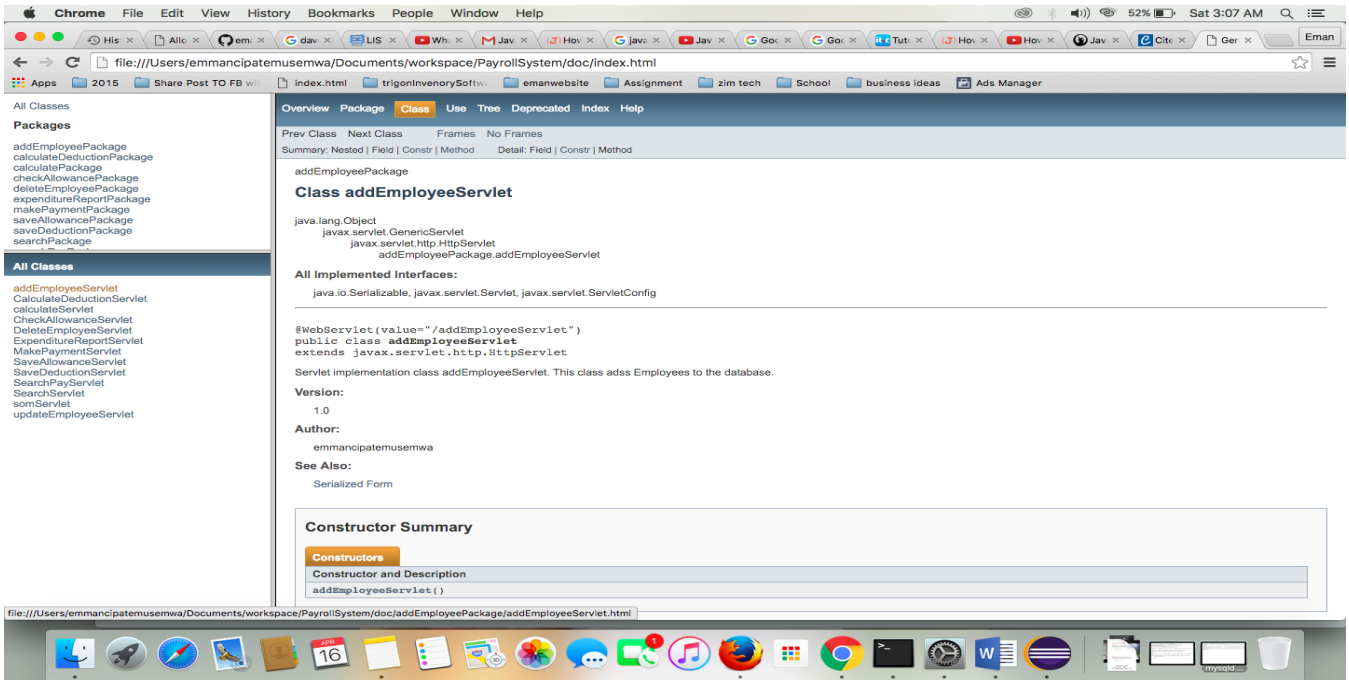
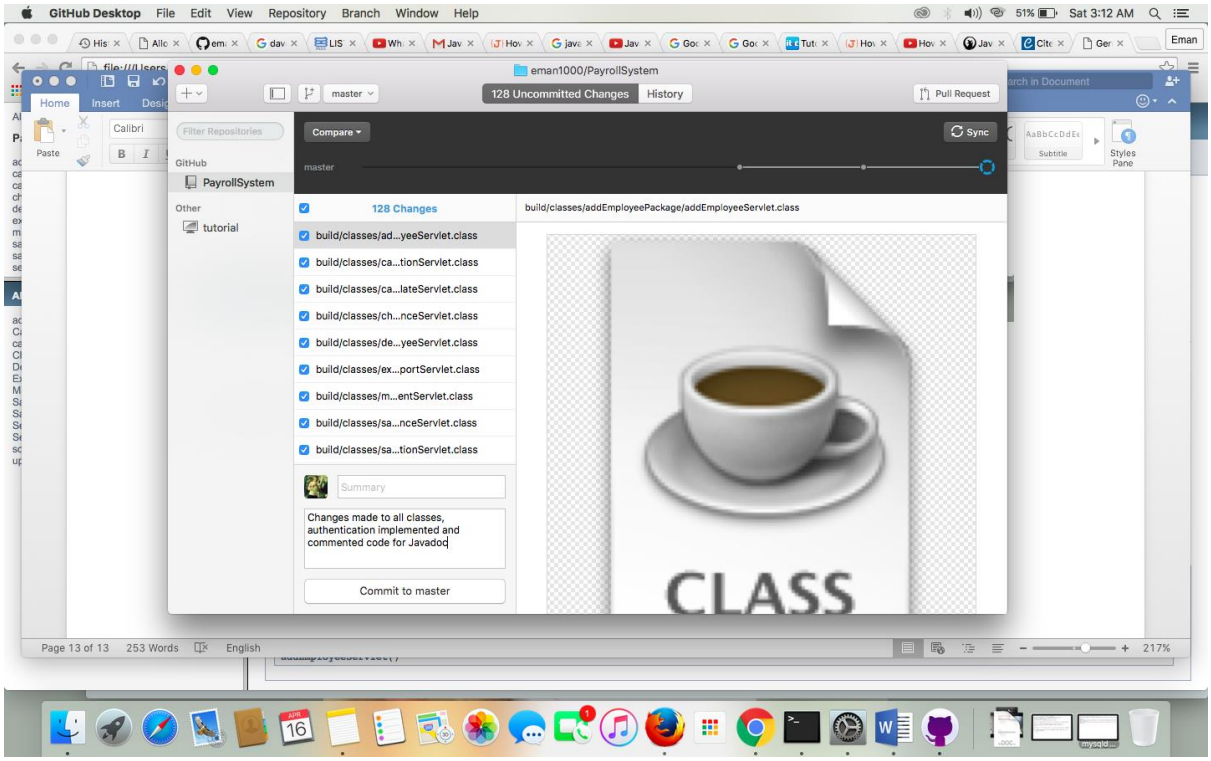


Fig 5.4 Generated java doc

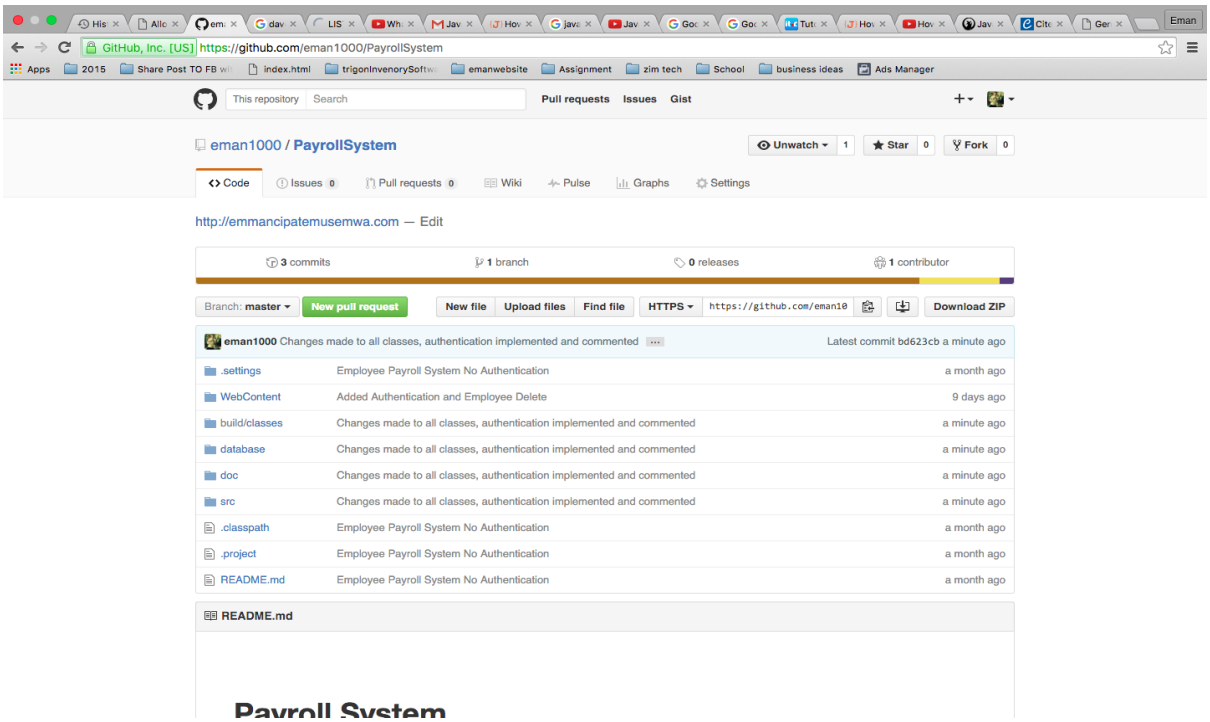
6.0 Versions Control

Here is the link to the version control on github https://github.com/eman1000/PayrollSystem

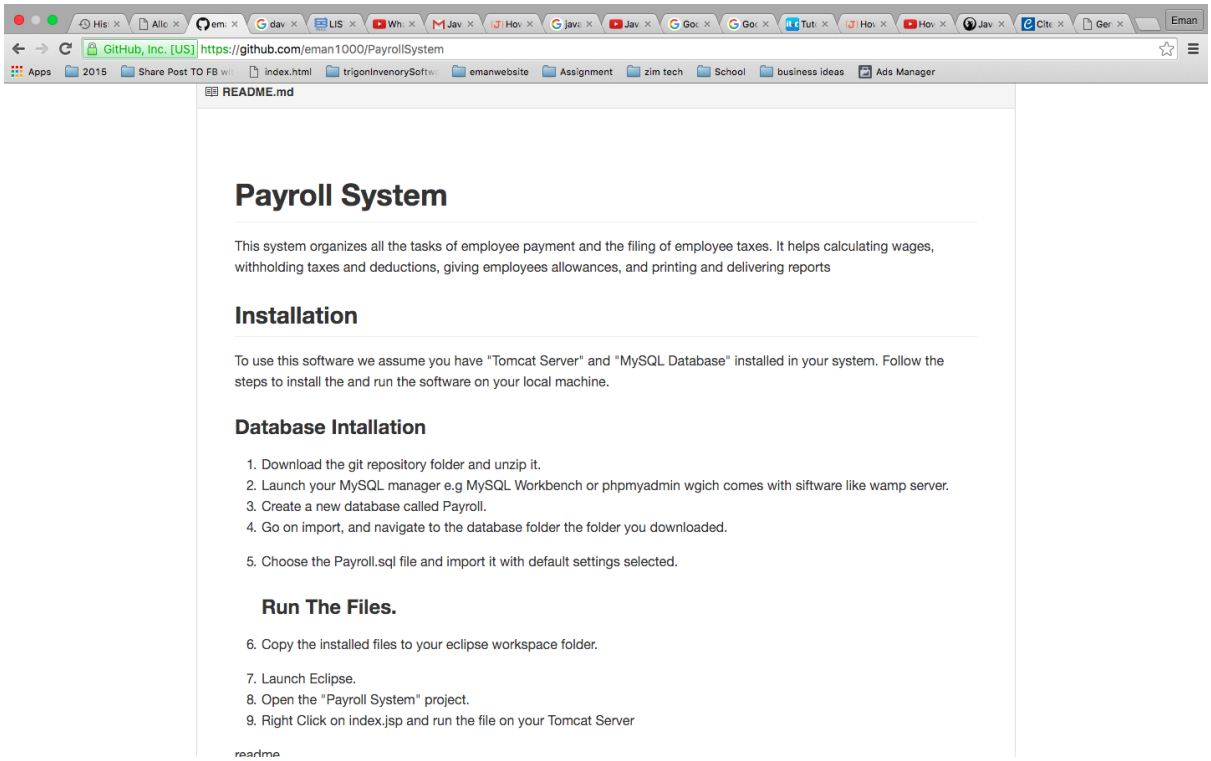
Image showing committing to github



Commits in GitHub Image showing commited files in github



6.1 Readme file in GitHub

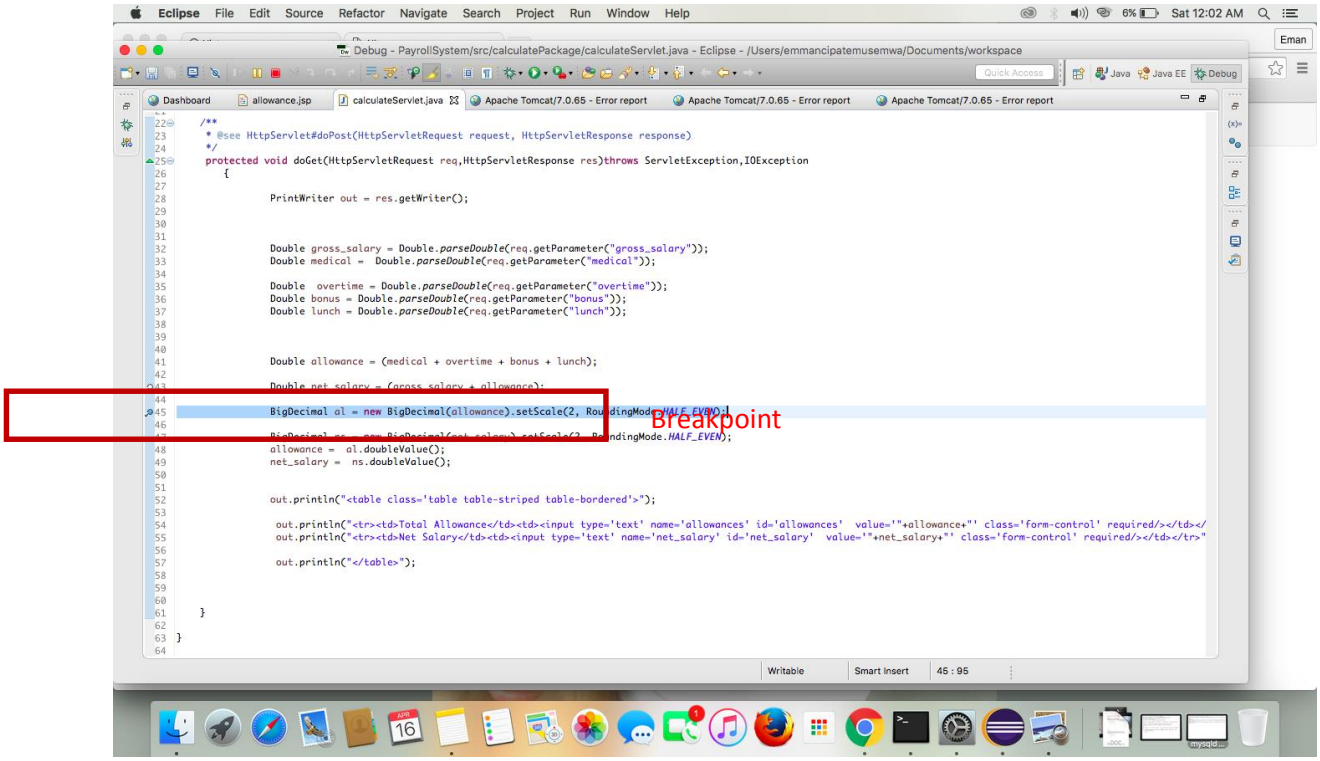


7.0 Debugging

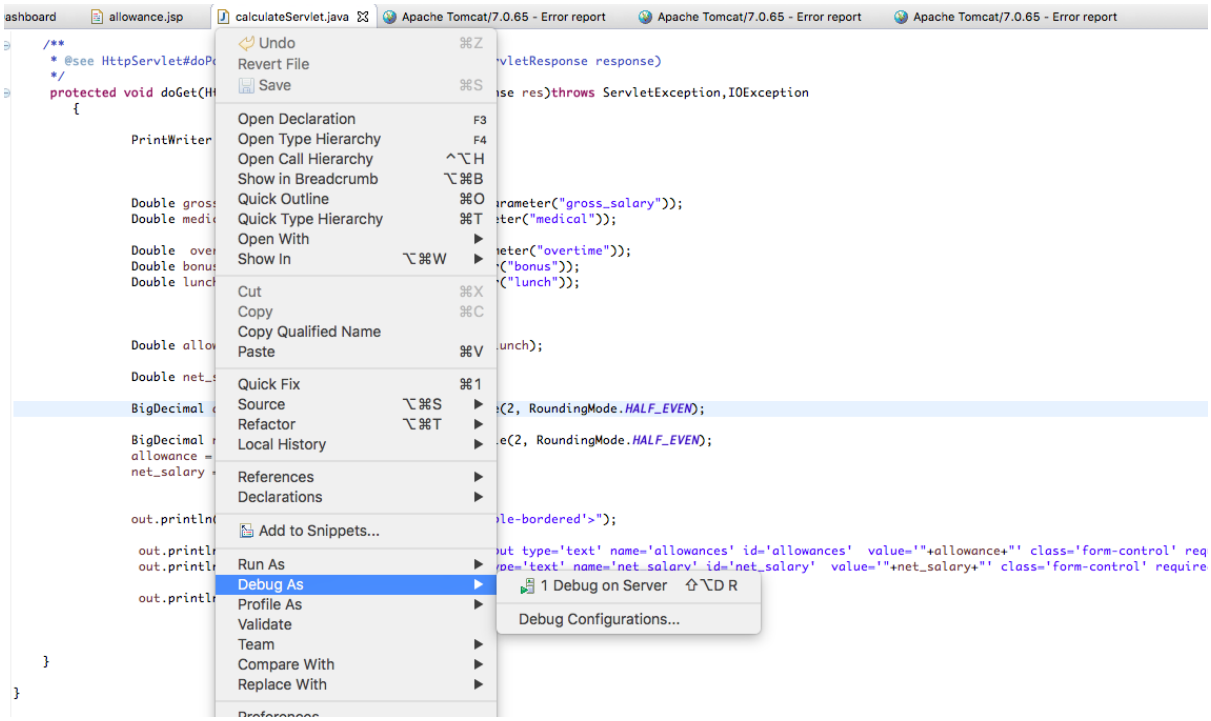
According to (Vogel, 2013), Debugging allows you to run a program interactively while watching the source code and the variables during the execution. In this section calculateServlet.java is going to be debugged. The servlet is responsible for calculating allowance. The breakpoint or watchpoint will help analyse the values of valriables.

7.1 Breakpoint

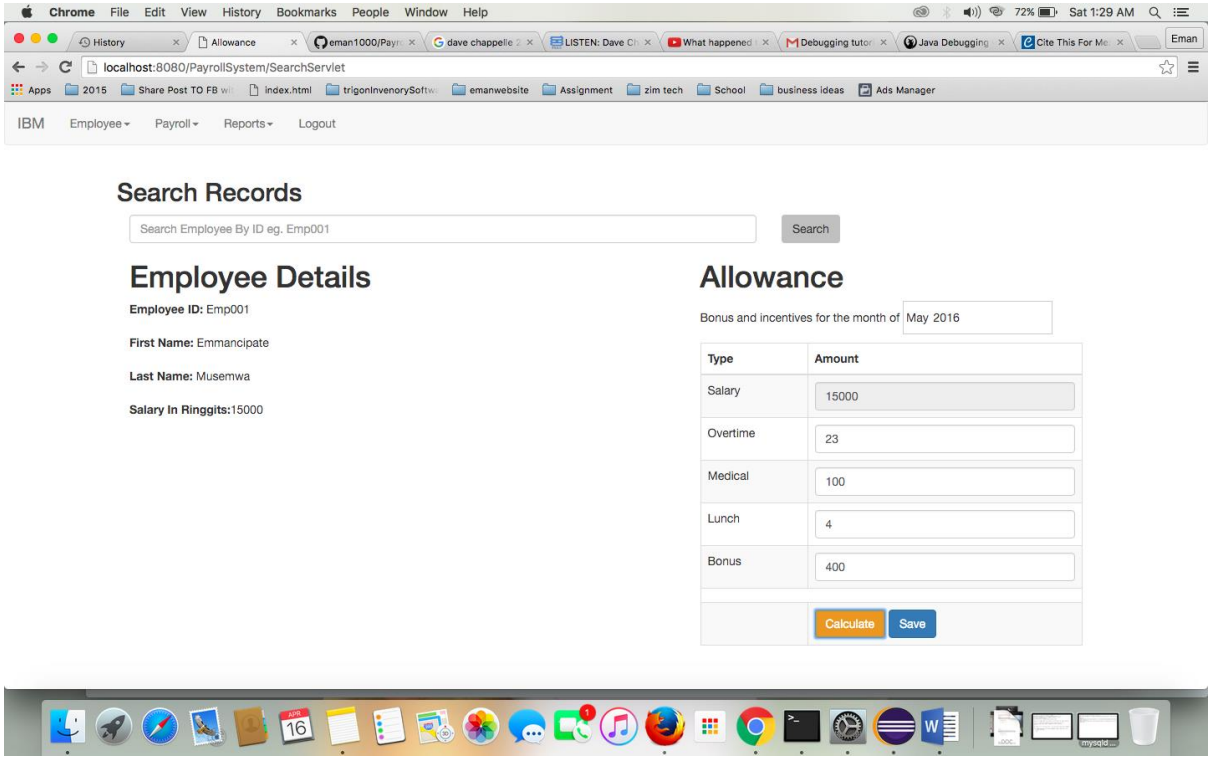
The following image shows a breakpoint specified on line 45. By breakpoints in the source code you specify where the execution of the program should stop (Vogel, 2013).



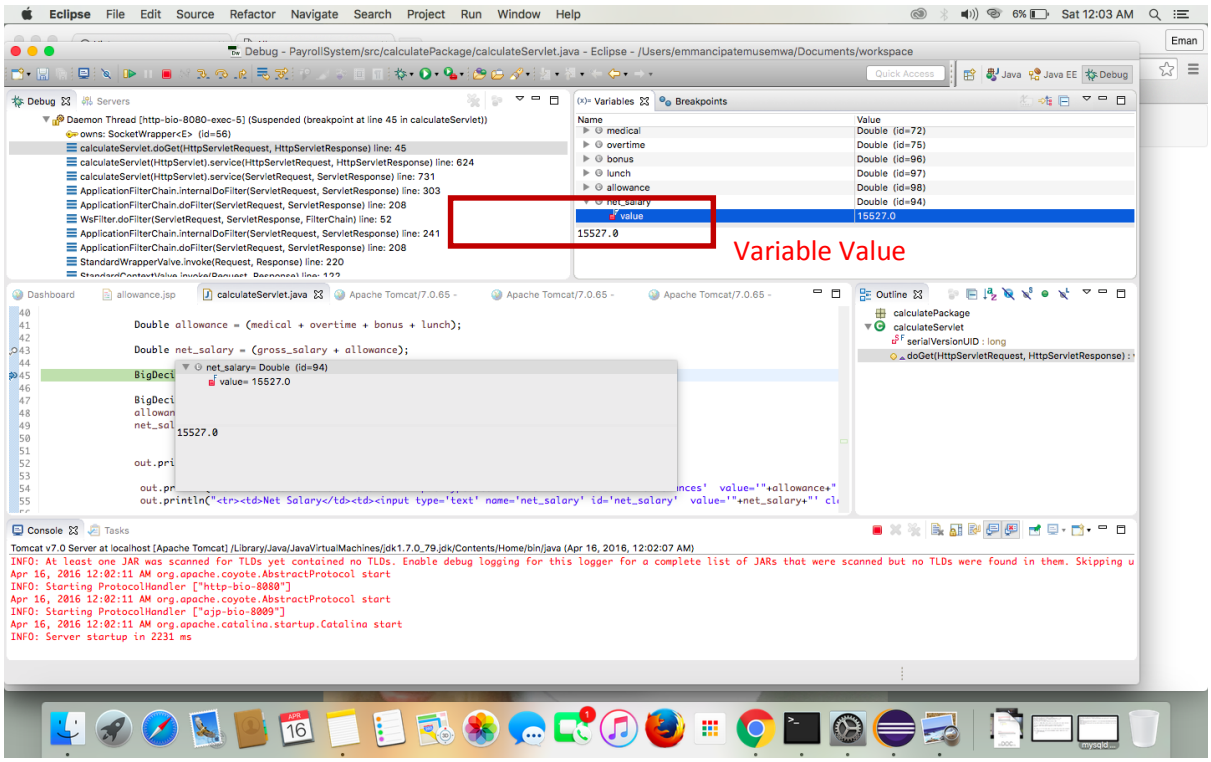
Now that a break point has been specified we can now debug the java file as follows



Once the debug in running we can use the system as normal, when the breakpoint specified is reached the program will stop giving us the current values of variables.



Once the calculate button is clicked, the calculate servlet will be run and the results are as follows.



7.2 Agile Approach

Agile methodology is the new trend in the software industry to develop a software and also popular due to its flexibility and adaptability. Agile model is a methodology to develop a software so it is also known as agile software development life cycle and it is the combination of the iterative and incremental process and it is used to deliver the software to the customers quickly. There are some strategy to work with Agile, here a product is divided into small fragments and each fragments have their life cycle. For example, planning, analysis, design, coding, unit testing etc. and the each iteration customers and stakeholder can see the project and also can give some suggestion.

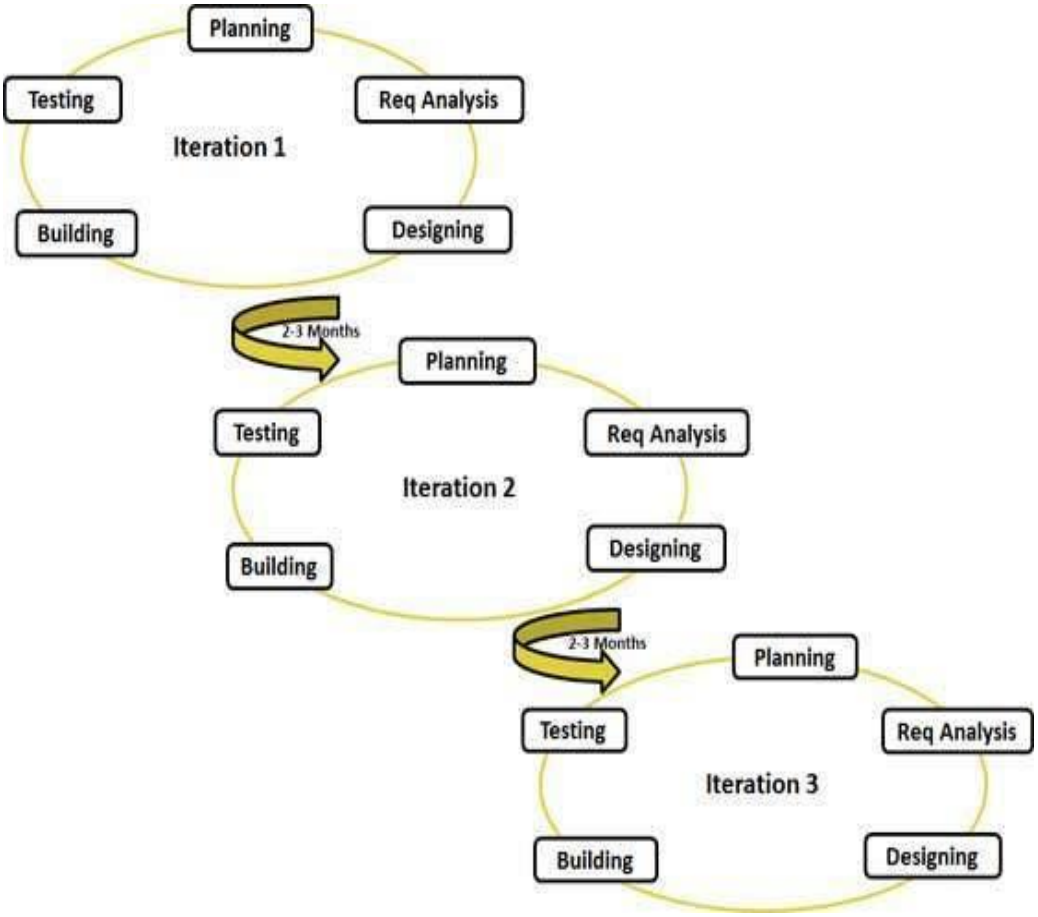


Fig 7.2 Agile SDLC (tutorialspoint, 2016)

Every project is based on some circumstances so during the development of the project will be needed some existing methodology with different form or new method. In the Agile approach, every part of thecompletion of a project is fixeddate.

7.3 Comparing Agile Approach with the Traditional Approach

Agile Software Development Life Cycle (SDLC) follows the different strategy to develop the software comparing the traditional SDLC. For example, waterfall model. Waterfall model follows the predictive approach to the development and required full documentation and planning. In this approach, full information needed from the customers before developing the software which is called the requirements analysis of a project as well planning. The most

beneficial part of this model is that everything is documented based so if in the future any changes needed it can be changed easily with the help of documentation but the agile approach is different but fast. here there is no need all the requirements from the customers at a time and also no such a planning is needed for the development of a software .Customers can come at any time and can go through different stages of a project if they think any changes needed it will be changed and it is based on adaptive approach. Customer interaction is important in this model but the disadvantage of this model is that there is no documentation so if there is any changes needed in the future so same team member will be needed who have developed this instated of the new developer.

7.3.1 Following are the Agile principles are:

- **Individuals and interactions** - In this approach self-organization and motivation are important because it is team work so everyone has to consider as best member of a team.

- **Working software** – Working Software is the best demo to presenting as a part of the project in front of the customer so customers can understands easily instead to write everything in the document.

- **Customer collaboration** –The continuous interaction is important with customers during the developing of a software to full fill all the requirements instead of the beginning of the software.

- **Responding to change** – Every part of the project is completing with different team so if any changes will needed the customer can go that a specific team and can approach some changes in a particular section and it will be changed quickly.

The most popular agile methods include Rational Unified Process (1994), Scrum (1995), Crystal Clear, Extreme Programming (1996), Adaptive Software Development, Feature Driven Development, and Dynamic Systems Development Method (DSDM) (1995). These are now collectively referred to as agile methodologies, after the Agile Manifesto was published in 2001 (tutorialspoint, 2016).

8.0 Conclusion

Technology has given us a lot of advantages which can be used for different work and its make a life easier also save much money and time. The flexibility of internet has changed the users' mind and now they can interact with the bank easily and can do any type of the transaction. The IBM Payroll System also based same principle of the internet banking which also deals all transaction related to the employees. This system is more flexible with employees and can show all the transaction within a minute and this separate section of the company. It has more demand in the future to save the money in every transaction.

Coding Standards

(Mytton, 2004) stated in his article that best applications are coded properly. He then went on to explain that “properly” does not only mean the code does its job, but it is also easy to maintain, add to and debug. Coding standards document what developers should do while writing there code. Instead of following individual ways of coding all developers will follow a standard way. Other developers who look t the code know what to expect through the entire application (Mytton, 2004).

Commenting

It is very fundamental to comment on code. (Mytton, 2004) stated that Looking through unfamiliar code is much easier if it is laid out well and everything is neatly commented with details that explain any complicated constructs and the reasoning behind them. Below is an example of commented code for the IBM Payroll System. The “//” characters are used for single line comments in java and they have been used throughout the project to explain blocks of code

```
53
54      String message = null; // message will be sent back to client
55
56            try {
57             // 1. Get a connection to database
58             DriverManager.registerDriver(new com.mysql.jdbc.Driver());
59             conn = DriverManager.getConnection(
60                     "jdbc:mysql://localhost:3306/Payroll", "root", "123456");
61
62
63                // constructs SQL statement
64
65                String sql = "INSERT INTO employees_tb (emp_ID, first_name, last_name, gender, dob, email, phone_numbe
66                     + "values (? ,? , ?, ?, ?, ?, ?, ?, ?, ?, ?)";
67                PreparedStatement statement = conn.prepareStatement(sql);
68                statement.setString(1, emp_ID);
69                statement.setString(2, first_name);
70                statement.setString(3, last_name);
71                statement.setString(4, gender);
72                statement.setString(5, dob);
73                statement.setString(6, email);
74                statement.setString(7, phone_number);
75                statement.setString(8, address);
76                statement.setString(9, desig);
77                statement.setString(10, depart);
78                statement.setString(11, salary);
79                statement.setString(12, date_joined);
80
81
82             // sends the statement to the database server
83                int row = statement.executeUpdate();
84                if (row > 0) {
85                    message = "Employee Data Saved in the Database";
86                }
87            } catch (SQLException ex) {
88                message = "ERROR: " + ex.getMessage();
89                ex.printStackTrace();
90            } finally {
91                if (conn != null) {
92                    // closes the database connection
93                    try {
94                        conn.close();
95                    } catch (SQLException ex) {
```

Code Paragraphs and indentation

Blocks of code should be separated into paragraphs for easy readability. Once the end of a paragraph is reached, just like in a word document, one can easily know that a new function or activity is starting. In the image below code paragraphs and indention can be clearly seen in use. If an external programmer looks into this code it would be easy for him/her to understand it.

```
        // gets values of text fields
        String payslip_id = request.getParameter("payslip_id");
        String emp_ID = request.getParameter("emp_ID");
        String payment_for = request.getParameter("payment_for");
        String gross_salary = request.getParameter("gross_salary");
        String allowances = request.getParameter("allowances");
        String deductions = request.getParameter("deductions");
        String net_salary = request.getParameter("net_salary");


      //Connect to the database
       Connection conn = null;
       String message = null; // message will be sent back to client
           try {
            // 1. Get a connection to database
            DriverManager.registerDriver(new com.mysql.jdbc.Driver());
            conn = DriverManager.getConnection(
                    "jdbc:mysql://localhost:3306/Payroll", "root", "123456");

               // constructs SQL statement
               String sql = "INSERT INTO payments (payslip_id, emp_ID, payment_for, gross_salary, allowances, deductio
                    + "values (? ,? , ?, ?, ?, ?, ?)";
               PreparedStatement statement = conn.prepareStatement(sql);
               statement.setString(1, payslip_id);
               statement.setString(2, emp_ID);
               statement.setString(3, payment_for );

               statement.setString(4, gross_salary);
               statement.setString(5, allowances);
               statement.setString(6, deductions);
               statement.setString(7, net_salary);

]
            // sends the statement to the database server
               int row = statement.executeUpdate();
               if (row > 0) {
                   message = "Payment Made to the Employee";
               }
           } catch (SQLException ex) {
               message = "ERROR: " + ex.getMessage();
               ex.printStackTrace();
           } finally {
               if (conn != null) {
                   // closes the database connection
```

Functions and Variables Naming

Variables and functions ought to be named consistently throughout the code. They should also be named according to the function they are preforming. They should briefly describe the data they contain (Mytton, 2004). The image below shows variables declared according to the data they contain.

```
//variable declaration
Double advance_pay = Double.parseDouble(req.getParameter("advance_pay"));
Double leaves =  Double.parseDouble(req.getParameter("leaves"));
Double  gst = Double.parseDouble(req.getParameter("gst"));
Double other_tax = Double.parseDouble(req.getParameter("other_tax"));
Double gross_salary = Double.parseDouble(req.getParameter("gross_salary"));Double gvt = (gross_salary
Double otherT = (gross_salary * other_tax);
Double deduction = (advance_pay + leaves + otherT + gvt);
Double net_salary = (gross_salary - deduction);
```

The screenshot below shows a class declaration which is name exactly according to its function.

```
18  */
19 //Deduction Calculation Class
20 @WebServlet("/CalculateDeductionServlet")
21 public class CalculateDeductionServlet extends HttpServlet {
22     private static final long serialVersionUID = 1L;
23
```

Reference

[1] Vogel, L. (2013). Java Debugging with Eclipse - Tutorial. [online] Vogella.com. Available at: http://www.vogella.com/tutorials/EclipseDebugging/article.html [Accessed 15 Apr. 2016].

[2] Visual-paradigm.com. (n.d.). Class Diagram - UML Diagrams - Unified Modeling Language Tool. [online] Available at: https://www.visual-paradigm.com/VPGallery/diagrams/Class.html [Accessed 15 Apr. 2016].

[3] tutorialspoint. (2016, April 17). Retrieved from http://www.tutorialspoint.com/: http://www.tutorialspoint.com/sdlc/sdlc_agile_model.htm

[4] Mytton, D. (2004). Why You Need Coding Standards. [online] SitePoint. Available at: http://www.sitepoint.com/coding-standards/ [Accessed 22 Apr. 2016].