Module: ECWM604 Advanced Web Technology
Assignment No.: 2
Name: G.R.A. Aponso
Student No.: 2008087
Email: **rumeshaponso@gmail.com**

Table of Contents

1. Introduction

This report is about the implementation phase of the ECWM 604 – Advanced Web Technology module's assignment 2. It briefly describes the step by step process of the implementation of the MAXHRMS web application where it discuss about application features, additional features and changes made to the initial designs (Mock-ups, MVC class diagram and ER diagram in AWT assignment 1), problems encountered and solutions found, testing, results, evaluation and lastly future enhancement.

2. Features of the MAXHRMS Web Application

Table 1, show a comparison between the features which are stated in the AWT assignment 1 with the features which are implemented in the MAXHRMS web application. The colours such as green, orange and red are used to mark the status for the easy understanding of the reader.

■ – fully implemented, ■ – partially implemented, ■ – not implemented

ID	Original Feature	Status	Notes
FR1	Login/Logout	**Yes**	Successfully implemented according to the assignment 1 specification.
FR2	View My Profile	**Yes**	Successfully implemented according to the assignment 1 specification.
FR3	Change Password	**Yes**	Successfully implemented according to the assignment 1 specification.
FR4	View Employee Information	**Yes**	Successfully implemented according to the assignment 1 specification.
FR5	Edit Employee Information	**Yes**	Due to the time constraint, the profile picture upload feature has not been implemented as specified in assignment 1. However, other mentioned criteria are successfully implemented according to the assignment 1 specification.
FR6	Add New Employee	**No**	Due to time constraint this feature has not been implemented.
FR7	View Payroll Information	**Yes**	Successfully implemented according to the assignment 1 specification.
FR8	Add New Salary	**Yes**	Successfully implemented according to the assignment 1 specification.
FR9	View Profile Information	**Yes**	Successfully implemented according to the assignment 1 specification.

Table 1 - Original Feature vs. Implemented Feature of the MAXHRMS Web Application

Accordingly in Table 1, FR6 – Add New Employee was not implemented due to the limited time constraint given for this assignment. However, 95% of the FR5 – Edit Employee Information has been implemented successfully. Moreover, the functionality of FR5 and FR6 are more or less the same. Thus, it is another reason not to implement FR6, in order to save time to properly load test the web application and finish the final report in time.

Moreover, apart from the features in Table 2, the search employee feature has been implemented as specified in the AWT assignment 2. Both admin and normal users can use this feature to search an employee, where they can perform search operation by any combination of the employee number, first name, last name, department and title fields. The search form uses 'GET' request method to retrieve information. Accordingly, the user logins are included with specific user level/role with different access levels. Thus, only admin users can view salary information of the employees.

The non-functional requirements are also been considered such as security, efficiency and the user-friendliness of the web application. Thus, a substantial amount of time have been spent to check the error scenarios and to improve the security of the web application. Also, in AWT assignment 1, there were two non-functional requirements such as,

- The web application should use a REST API wherever possible.
- AJAX style of communication between browser and server.

Thus, these two non-functional requirements have been addressed in the implementation of the web application. As shown in Figure 1, the AJAX style communication between browser and server is maintained. In employee management page, AJAX used to load the data to data table. In the edit employee information page, AJAX used to send the data to the server for storing.

Figure 1 - AJAX usage in the Web Application

Figure 2, shows the REST methods that have been used in the web application. The left side of Figure 2 shows employee information 'READ'. Likewise, the right hand side of the Figure 2, shows the 'CREATE' of a salary record. The 'UPDATE' and 'DELETE' could not have been implement due to limited time constraints. Thus, update and delete features have not been considered in the web application.

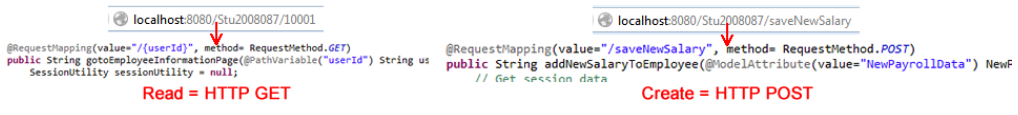

Figure 2 - REST usage in the Web Application

3. Additional Features and Changes Made to Initial Designs

As shown in Figure 3, the search feature is not in the initial design (mock-up), where it has been added to the final design as highlighted in screenshot. The reason for this change is that in the AWT assignment 2, there was requirement of search employee. However, the initial mock-up and MVC class diagram was not supported the required feature, thus the stated modification has been done to the actual view. Accordingly, as shown in Figure 3, the MVC class diagram was changed. The methods in red colour square were replaced with new methods in the 'EmployeeController' to handle the employee search function.

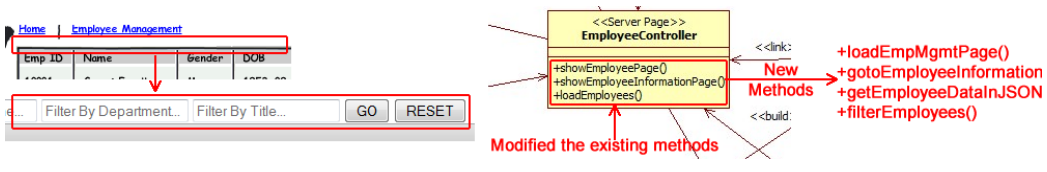

Figure 3- Search Functionality Modification | MVC Class Diagram Modification

As shown in Figure 4, the payroll information screen has been changed. The 'Emp. No.:' combo box is changed into a text field and the employee name is moved to data table. The reason for this change is, when loading all the 300K employee numbers to the combo box in mock-up, the application got non-responsive while injecting all the records into the view. Thus, to handle the issue stated change has been made.

Figure 4 - Payroll View Modification

Apart from the major changes shown in Figure 3 and Figure 4, the implemented web application is comparatively the same with the initial designs (mock-ups). Additionally, alignments and positioning of buttons and text fields are changed in some places such as Add New Payroll and Edit Employee pages to enhance the user experience.

The changes made to MVC class diagram and ER diagram are in MVC Class Diagram, ER Diagram and Coding section.

4. Screenshots of the MAXHRMS Web Application

Figure 5, shows the login page and the common home page of both admin user and normal user. When the user clicks on a particular icon or a link, the user's user level will be checked. Accordingly, if the users have access to the particular view, then the resulted page will be shown. Otherwise 'Access Denied Page' will be shown.

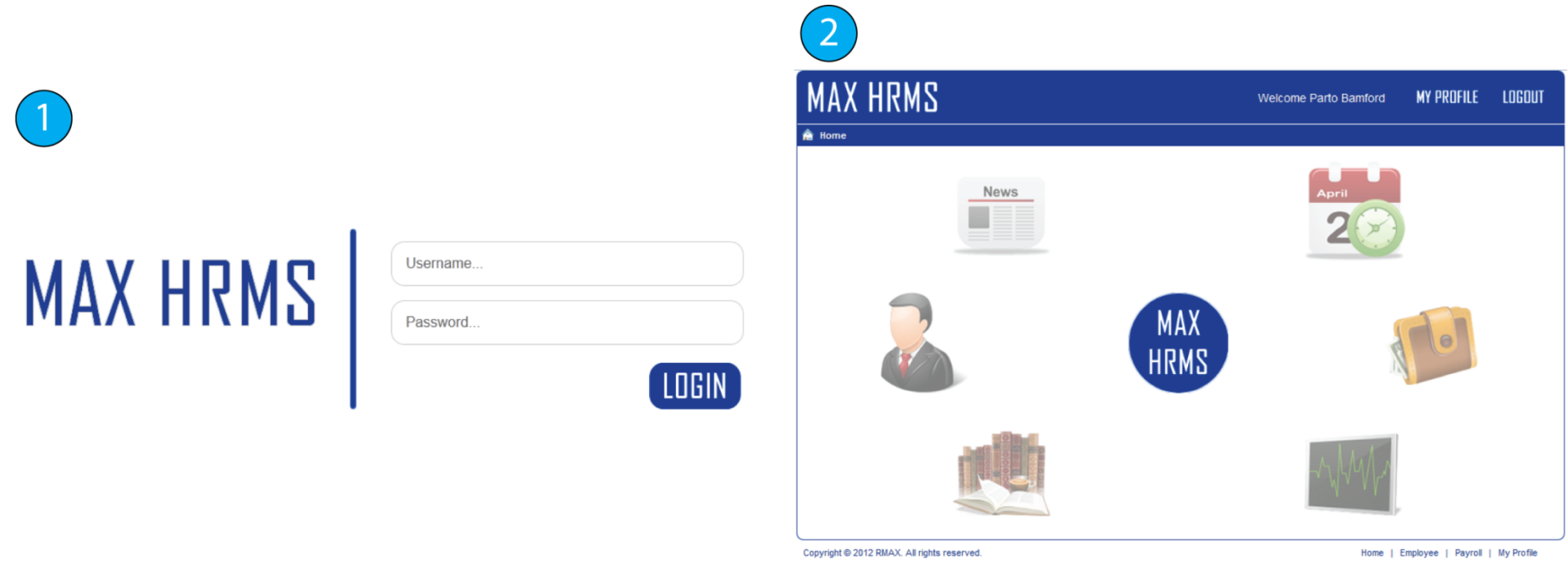

Figure 5 - 1 - Login Page | 2 - Common Home Page

Figure 6, shows the common my account page and the employee management page of an admin user. If the user is a normal user, then the highlighted (red colour square) add new employee icon will be hidden.

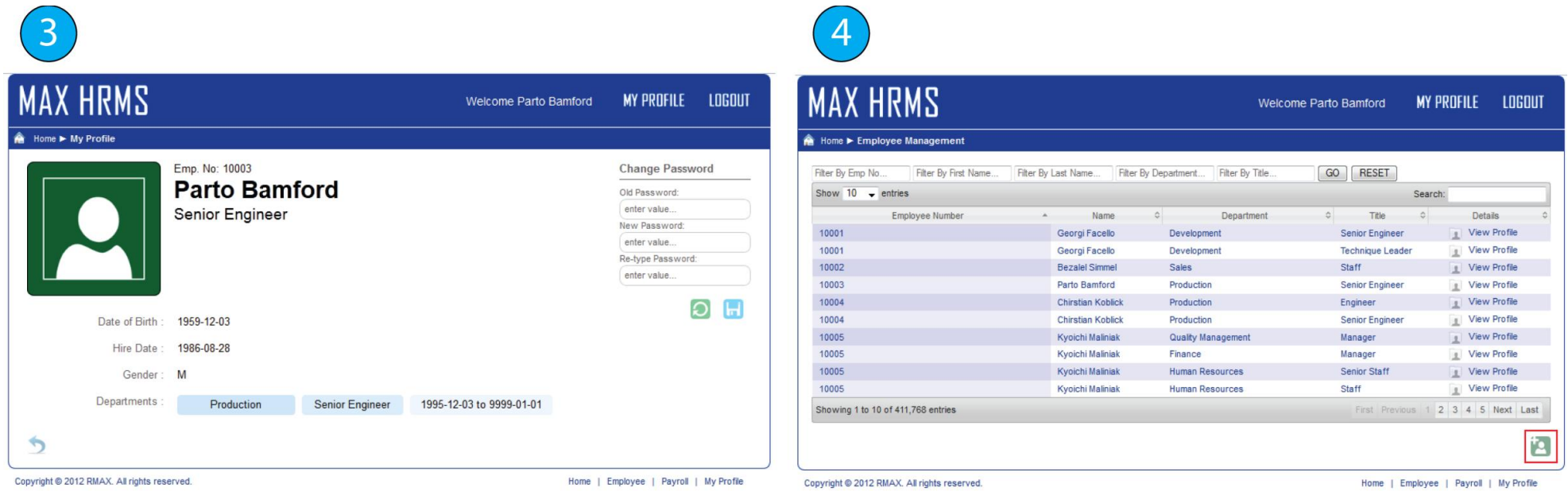

Figure 6 - 3 - My Profile Page | 4 - Employee Management Page (Admin User)

Figure 7, shows the employee information page and the edit employee page of an admin user. If the user is a normal user, then the highlighted (red colour square) edit employee information icon will be hidden. Similarly, a normal user cannot view/access edit employee information page.

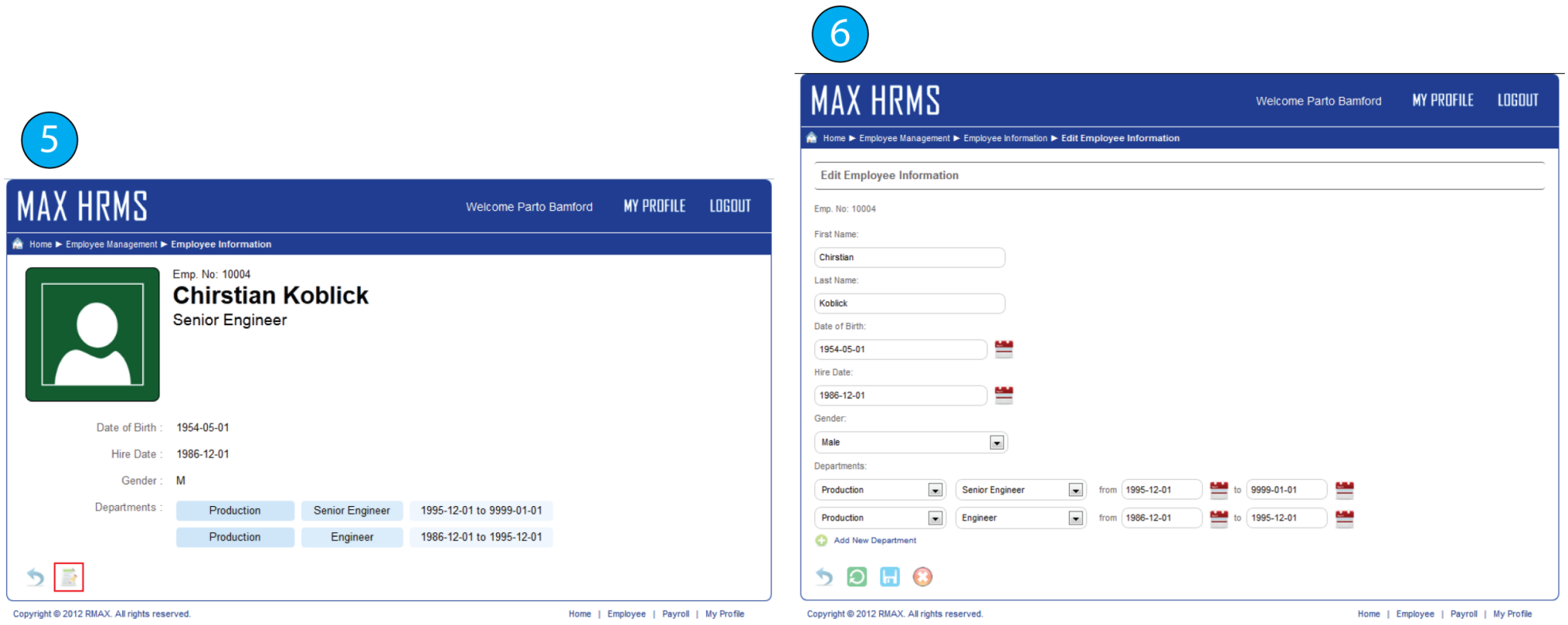

Figure 7 - 5 - Employee Information Page (Admin User) | 6 - Edit Employee Page (Admin User)

Figure 8, shows the payroll management page of the admin user, site under construction page and access denied page. A normal user cannot view/access payroll management page.

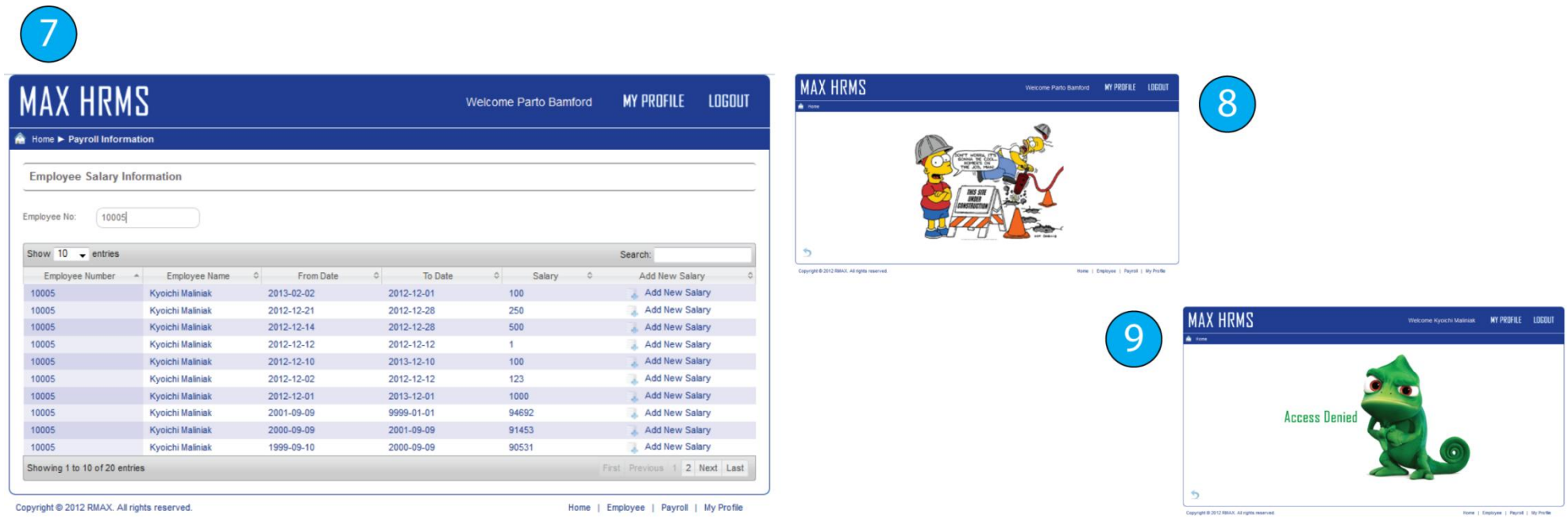

Figure 8 - 7 - Payroll Management Page (Admin User) | 8 - Site Under Construction Page | 9 - Access Denied Page

5. User Authentication

The user authentication process is successfully implemented as discussed in AWT assignment 1. There are two different user roles with different access levels. Moreover, the user passwords are encrypted and salted using 'SHA-256' hash function. As shown in Figure 9, the left hand side image shows the view of a normal user with no such add new icon, where the right hand side image shows an admin user's view with add new employee icon.

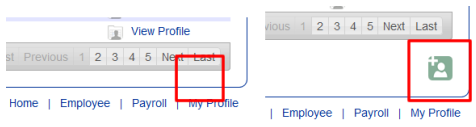

Figure 9 - Employee Management Page Add New Employee Icon

As shown in Figure 10, the left hand side image shows the view of a normal user with no such edit information icon, where the right hand side image shows an admin user's view with edit employee icon.

Figure 10 - Employee Information Page Edit Employee Information Icon

Accordingly, there are other features such as payroll information, where the user authorisation will be checked before display the page. Furthermore, as discussed in AWT assignment 1 and as shown in Figure 11, SSL certificate has been used in the server to enhance the security of the web application.

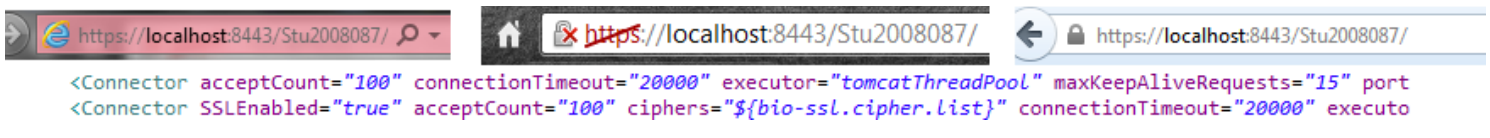

Figure 11 - Running Web App in Browsers | left to right - Internet Explorer - Chrome - Firefox | down – The Tomcat Server Configuration

6. MVC Class Diagram, ER Diagram and Coding

In order to implement the MVC class diagram which was designed in AWT assignment 1, Spring MVC has been used. To create the project the Spring Tool Suite was used. As shown in Figure 12, the MVC class structure has been followed throughout the project. All the controller classes are in '*.controller' package. All the views are in the 'views' folder in the 'webapp'. To implement model classes two packages were used named '*.model.entityclasses' and '*.model.viewdto'. One of the reason to use such two sub packages is to simplify the class structure and for better understanding. Other reason is entity classes are directly mapped to employees sample database tables. Hence, those classes are somewhat complex, contains lot of unnecessary data and sometimes it is difficult to map them to the actual view. Thus as a solution for this matter, a simple class structure called 'viewdto' has been introduced. These classes are very simple and include only the needed fields to render the view. These 'viewdto' classes are only used to pass the values to views, whereas 'entityclasses' are used when data storing and data retrieving from the database.

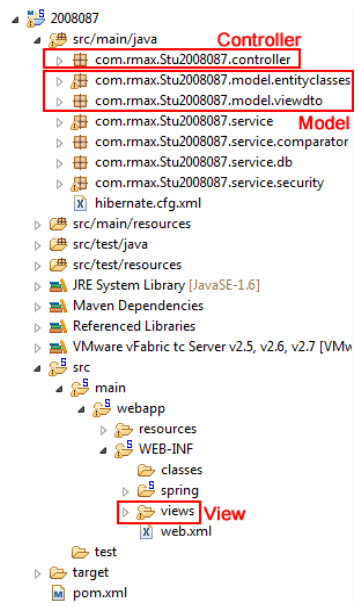

Figure 12 - MVC Class Structure

Also as shown in Figure 12, few modifications such as service layer have been added to the original MVC class diagram in AWT assignment 1. As shown in Figure 3 earlier, the employee controller's methods have been modified. Similarly, 'PayrollController' methods have been modified. However, the changes which have been made to the class diagram are not affects the overall design. Thus, the design could be considered as a good solution for the given problem.

The ER diagram designed in AWT assignment 1 has not been changed. Therefore the all the field, data types and lengths are same. Apart from that, as shown in Figure 13, few user account records were added to 'user_acc' table in order to test the web app.

emp_no	security	username	password	salt_value	userImgPath
10001	1	emp10001	241a2cfd2d3d3c3...	6e7a6b444745...	NULL
10003	2	emp10003	c8ba140af41e235...	78316b52694d...	NULL
10005	1	emp10005	0866ac7313636a...	7738682f514c6...	resources/image.

Figure 13 - Added User Account Records

While adding user account records, it is assumed that one user has only one user account. That he/she either could be an admin user or a normal user. However, at the same time he/she cannot have an admin account and a normal account. Thus, the user account records are unique.

7. Problems Encountered and Solutions Found

While implementation of the web application few problems were encountered. Most of the problems were small and managed to solve them by researching in the internet. Web sites such as www.w3schools.com, http://jquery.com and http://stackoverflow.com were very useful.

Generally, according to the MVC class diagram in AWT assignment 1, employee controller is always retrieves all the employee data before employee management page shown. This action took substantial amount of time and affected the efficiency of the application. Therefore, the choice between load employee details frequently or load employee details at the beginning was considered, in order to gain more efficiency. Finally the choice of loading data at the beginning was choose. The experience received during placement year was mainly useful to make the stated decision. As shown in Figure 14, singleton design pattern was used to tackle the data loading issue.

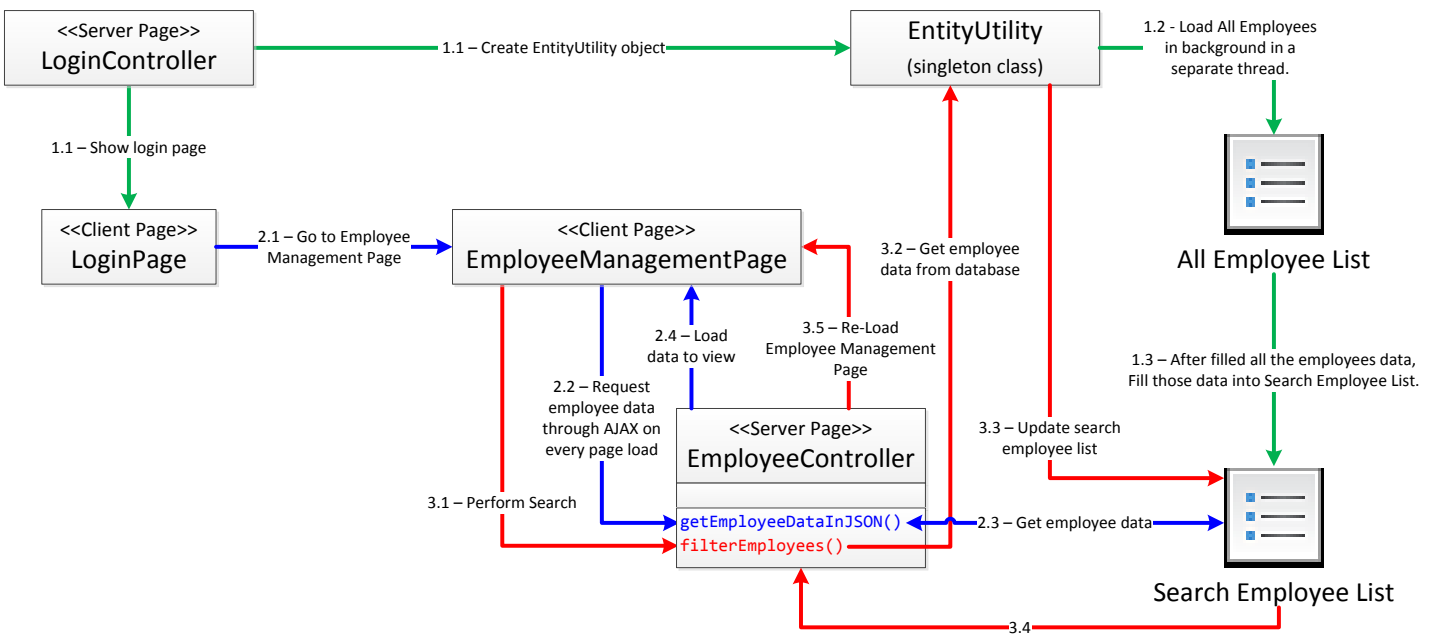

Figure 14 - Flow of Search Functionality

As shown in Figure 14, 'EntityUtility' is a singleton class, where it has two lists named 'AllEmployeeList' and 'SearchEmployeeList' as private variables. The application starts by executing the 'showIndexPage' controller method in 'LoginController'. As shown in green colour arrows, while loading Login Page, simultaneously an 'EntityUtility' singleton object will be built for the first time and 'AllEmployeeList' and 'SearchEmployeeList' will be filled in background using another thread. Loading of all employee data and search employee data takes approximately around 4 to 10 seconds. Afterwards, the flow of events follows the flow which describes in the Figure 14 by the specified numbering. There are few important facts such as, the 'getEmployeeDataInJSON' method will call by 'EmployeeManagementPage', every time on the page load via AJAX (refer blue lines in the Figure 14). Plus the 'getEmployeeDataInJSON'controller method always refers the pre-loaded 'SearchEmployeeList'. Thus, the efficiency of data loading is high, because the data is in the run time. Whenever, the user performs a search, the 'SearchEmployeeList' will be updated according to the combination of filtering values. If all the search parameters are not empty, then this action retrieves data from the database. If all the search parameters are empty, then the 'SearchEmployeeList' will filled with the data in 'AllEmployeeList'. After loading data to the 'SearchEmployeeList', the 'EmployeeManagementPage' will be reloaded. Then 'getEmployeeDataInJSON' will automatically triggered and the data table will be filled with appropriate search records.

Before use this solution the search operation took approximately 4 to 10 seconds at the first time. However, when number of search requests increases the amount of time taken for search operation increased by approximately 10 to 15 times longer than first time. On the other hand, with the solution discussed above, it always took approximately 4 to 10 seconds to do a search operation. Thus, the discussed solution is as improved solution for this scenario. For more statistical information about the efficiency of the search function, please refer Results section.

8. Testing

In order to evaluate the efficiency of the web application, a series of load tests were carried out. Since the web application is small, Apache Bench was used to perform the load tests. According to the research which carried out, there is a number of different load test tools available such as NeoLoad, LoadRunner, OpenLoad, etc. (load-testing-tools.com, http, 2012)

The load tests were based on a number of parameters such as number of requests to perform for the benchmarking session and number of multiple requests to perform at a time (number of concurrency requests). The load tests were carried out in two different machines (a local and a remote machine). Three different searches 'GET' requests were used to perform the benchmarking. The load test parameters were increased at each time. The results were recorded and presented in the Results section.

9. Results

Following search 'GET' request were load tested in the local machine and in a remote machine (the web site were accessed via Wi-Fi). The load test results shown in Table 2 with minimum, median and maximum request completion times. The evaluations of the results are in the Evaluation section.

-n - number of requests to perform for the benchmarking session, **-c** - number of multiple requests to perform at a time (number of concurrency requests)

Search 'GET' request URL	No. of search parameters	-n	-c	Local (time per request)			Remote (time per request)		
				min	median	max	min	median	max
http://localhost:8080/Stu2008087/filterEmployees?searchCriteriaEmpNo=&searchCriteriaFirstName=&searchCriteriaLastName=Facello&searchCriteriaDepartment=&searchCriteriaTitle=	1	1000	10	2809ms	4712ms	8824ms	2404ms	4707ms	10218 ms
http://localhost:8080/Stu2008087/filterEmployees?searchCriteriaEmpNo=&searchCriteriaFirstName=&searchCriteriaLastName=Facello&searchCriteriaDepartment=Human+Resources&searchCriteriaTitle=	2	5000	25	205ms	1021ms	1874ms	125ms	828ms	2875ms
http://localhost:8080/Stu2008087/filterEmployees?searchCriteriaEmpNo=&searchCriteriaFirstName=&searchCriteriaLastName=Facello&searchCriteriaDepartment=Human+Resources&searchCriteriaTitle=staff	3	10000	50	205ms	2074ms	4150ms	265ms	1703ms	3578ms

Table 2 - Apache Bench Load Test Results of the Search Functionality

Note: When testing in the remote machine, the search 'GET' request URL's 'localhost' was changed into the IP address of the hosted machine of the web application.

10. Evaluation of the Results

According to the results on Table 2, it is clear that when the number of request and the number of concurrency requests increases, the time taken per request to complete is increasing. Similarly, the number of search parameters are also clearly affects the completion time of a request. However, the maximum time taken to complete a request in both local and remote machines is less than or approximately equal to 10 seconds. Thus, search functionality could be considered as efficient. Furthermore, it would have been better, the testing could carry out further to gather more information and to improve the search functionality. However, due to the time limitations it was not possible.

When looking at the complete web application, there are a number of strengths, weaknesses and areas of improvements can be identified. The login, search and edit employee features could be considered as strengths, because those function are secure and user-friendly. The design was used to tackle the search function is good and test results shows that is it efficient. However, the web application is demonstrating only a tiny bit of the HR management system. Thus, it could be considered as a weakness. As a result, it leads to the areas of improvements which could add to the web application to enhance the user experience. For example, information such as news alerts and upcoming events in a calendar widget could add to the home page. Similarly, features such as language change, more interactive UIs, etc. could be considered as areas of improvements.

11. Future Enhancements

The HR management system which implemented under this assignment is just a basic prototype. Thus, features such as attendance, leavings, knowledge management, scheduling, announcements, etc. could be added to the existing web application. As well, the security aspect of the login function and information of the application could improve more. The report generation function could be added to the web application.

12. Research

In order to successfully implement the web application, a research carried out in different areas such as spring mvc, hibernate, jQuery widgets, load testing, hash functions and spring security. The research helped to overcome the most of the problem faced during the implementation of the web application. Apart from the research, the knowledge gained throughout the past few years were very useful.

13. References

Freitag, P. (2009). Using Apache Bench for Simple Load Testing. [Online]. <http://www.petefreitag.com/item/689.cfm> [Accessed date: 2012, Dec. 19]

How to install Apache Server on Windows. (2010). [Online]. ricocheting.com. <http://www.ricocheting.com/how-to-install-on-windows/apache> [Accessed date: 2012, Dec. 19]

Mkyong. (2010). Java SHA Hashing Example. [Online]. Mkyong.com. <http://www.mkyong.com/java/java-sha-hashing-example/> [Accessed date: 2012, Nov. 08]

Spring JDBC Example. (2012). [Online] tutorialspoint. <http://www.tutorialspoint.com/spring/spring_jdbc_example.htm> [Accessed date: 2012, Dec. 13]