COS30041 Creating Secure and Scalable Software

Software Proposal for Distinction (D)

Prepared by: <Your name, student id>

Intended Grade: <D or HD> [In case of HD, your tutor will take into your HD Research proposal for consideration as one big project to judge your HD proposal]

Submission for Feedback (Week 6 – 7) and Final Approval (Week 8, Fri, 5:00pm)

[Note: Based on past experience, it will take at least 2 to 3 submissions to make it right. So, start early]

Instructions - This document is for students aiming to achieve Distinction (D) or above.

For **D grade**, a student needs to do a software application that can demonstrate their skills in developing enterprise application using **a variety of technologies discussed in this subject**. In judging whether your application will be approved or not, the lecturer will be looking into whether the student can utilize various technologies (e.g. as a guide 70%+ of those discussed in the subject) in the application.

For **HD grade**, a student needs to complete the software for D grade as well as a research report. For detailed requirements of the research report, please see the HD Research Proposal Template document in 81_HD_Task_5.4.

Intended Learning Outcomes (extracted from Unit Outline)

1. Build and deploy secure and scalable application using contemporary frameworks
2. Explain and apply strategies, patterns and frameworks to address a range of scalability issues
3. Explain and apply strategies, patterns and frameworks to address a range of security issues
4. Use contemporary tools to evaluate the scalability and security of applications

SOFTWARE PROPOSAL for Distinction

Software Title: Staffboard

Introduction

Staffboard is a web-based enterprise java application for sharing contact information between departments within an organisation. Staffboard provides a web-based portal for viewing the contact information, a login gated interface for updating the contact information, and an admin interface for changing the layout and display of the contact information.

Business Scenario

Within an emergency department communication is critical to maintain and manage patient flow. To assist in this the clerical team are responsible for updating an excel spreadsheet with the current shifts contact details for the nurses and clinicians which are then displayed in a web page via an excel macro. A second excel file contains a list of contact information outside of the department, including for other wards and hospitals.

There are several issues with the current setup:

1. Persistence

   Currently the excel files are stored in a network drive and staff are prohibited from saving the file, as if the file is saved locally the upload macro will not work. As a result, the file must always remain open, and if the file is accidentally closed all the data must be manually re-entered.

2. Accessibility

   Currently only the in department contact information is broadly available to the emergency department staff members. The other ward numbers and hospital contact information is in another excel file on a network folder that most of the nursing staff are unfamiliar with, so they often need to ask the clerks for this information. None of this information is shared outside of the department.

3. Accountability

   There is no login process for accessing the excel files, and as such there is no way of tracking which staff member made changes.

4. Source of data is centralized

   Since only a single group of employees manage these files, any changes must be made by them. Often this results in lost data as the staff members will often either forget to ask the clerks to update their information or they will update their information in the patient management system instead, which results in the department having multiple sources with conflicting information.

Functionalities that you want to implement

Below is a list of functionalities for the software:

F1. Users will belong to one of three user groups: employee, manager, admin
F2. Employee's and managers will belong to one or more departments
F3. Administrators can create and delete departments
F4. Administrators can authorize an employee as a manager & vice versa
F5. Administrators can set which departments managers can edit the contacts of
F6. Managers can add contacts to a department
F7. Managers can choose whether the role can be edited
F8. Managers can set which employees have edit permission
F9. All users can view the contact information for the departments they belong to
F10. Managers can define the contacts that exist for the department
F11. Employees with edit permission can edit the contact numbers for a department
F12. All users can reset their password via email

Functionality and Technology Traceability Matrix

The following table shows the relevant technologies discussed in this subject that could be used to implement the functionalities as suggested in the Functionalities section above. If you plan to use some other technologies that has not been discussed in the subject, please specify.

At this stage, just the nature of individual software component is enough. There is no need to do "design" on the software component level (e.g. the names and responsibilities of the classes, and how these classes interact with each other). Having said that, it is nice to have but not necessary.

Functionality	Related technologies (discussed in this subject + others, please specify) <<e.g. Data Tier – Entity Class using ORM [no JDBC – too old☺]; Business Tier – EJB (Stateless, Stateful, Singleton), JavaMail API; Web Tier – JSF, Managed Beans, AJAX, JSON; Others – PrimeFaces [on top of JSF]; use of JSON for Web Services [instead of using DTO for communication between servers]>>

F1	User table and groups defined in java db, web.xml, glassfish jdbc pool, resource, and realm configured to restrict access based on user group.
F2	User table in Java db will reference department table via a junction table between the two.
F3-F11	JSF web pages will display the UI, security roles in web.xml will restrict user access to pages, managed beans will hold and save user session data with role restrictions placed on methods to prevent user from performing actions not allowed by their role. Stateless beans will persist and retrieve data from database, JSON will be used in place a DTO. As a stretch goal, AJAX will be used to make the application a single page website.
F13	Will use the email functionality from tutorial 5.1 / 5.2C