COMP 1531 Group Project, 18s2

Aims:

The aim of this group project is to enable students to consolidate their knowledge in the fundamental principles of Software Engineering and apply the theoretical concepts to a “hands-on” software engineering problem. The project will enable students to:

- Develop problem-solving skills to solve ‘real-world’ software engineering problems; analyse the problem domain, design and develop a solution to the problem
- Learn to work effectively as part of a team by managing your project, planning, and allocation of responsibilities among the members of your team
- Gain experience in collaborating through the use of a source control
- Apply appropriate design practices and methodologies in the development of their solution

Due Dates:

- Milestone 1: 11:59 PM Sunday 2nd September (Week 6) (Feedback: Week 6 Lab)
- Milestone 2: 09:59AM Monday 17th September (Week 9) (Feedback: Week 9 Lab)
- Final milestone: Final milestone: 09:59AM Monday 15th October (Week 11) (Demonstration: Week 12 Lab)
- Moodle Peer assessment: 11:55PM Sunday 21st October

Value: 30%

Background

Your consultancy firm has been called in to Australia’s largest healthcare enterprise, MediSoft, for a new software project. The client is interested in developing an online healthcare appointment management system (HAMS) that would streamline the process of patients booking healthcare appointments and

general practitioners sending referrals to specialists. The client feels that such a system would help to market and increase the likelihood of attracting more customers to the health centres affiliated with MediSoft.

Preliminary Requirements

Health-Care Centres

The e-health system stores all health-care centres (which can be a hospital or a medical centre) that are affiliated with MediSoft and health-care providers working for these health-care centres. A health centre may employ one or more health providers, and consequently, the set of services provided by a health centre depends on the group of providers working for that centre.

User Types

The users of the proposed system can be grouped broadly into two categories, namely patients and health-care providers. Every user registered on the system has their full name, email address, and their phone number stored in the system. Patients additionally have their Medicare card number recorded as well, while health providers have their provider number stored on the system.

Health-care providers can be further categorised by their professions, that is, the type of healthcare service they provide. For example, a provider may be classified as a general practitioner (GP), a pharmacist, a physiotherapist or a pathologist. Every health provider has exactly one health profession. A health provider may work for one or more health centres. Furthermore, different health providers may have different workings hours; and if a provider is affiliated with multiple centres, then the provider will have different working hours associated with each centre. (*You can assume that these hours do not conflict*)

Every user registered on the e-health system can update their personal information through a profile page. For health-care providers, the profile page additionally displays their current rating and the health-centres they are associated with.

Patients

Searching for a Healthcare Service to Book an Appointment

On the proposed system, the patients can search for a health provider or a health centre that suits their interest using the search engine provided. The search engine has the following options:

- search for a particular health-care centre by their name (e.g., Garners Medical Centre)
- search for health-care centres by suburb (e.g., Kensington)

- search for a particular type of service (e.g., GP) (e.g., a search for a 'GP' service will list all the GPs currently registered on the system).

- search for health-care providers by their name (e.g., Jackson)

For any of the four search options, the search should not only return the exact matches but 'near-matches' also. For example, if there is a health centre called "Royal Randwick Hospital", then it should be returned as a result of the patient searching for 'centres by name' with the keyword "royal rand". Each entry of the search result should contain brief details about the centre/provider, and a link to the corresponding profile page.

Viewing Health Centre Profile

The profile page of a health centre should display details about the centre and its customer rating. Furthermore, there should be a list of all health providers working for the centre, categorised by the type of service they provide. For each provider, there should be a link to the provider's profile as well as a button to proceed to booking an appointment with the provider at the centre.

Viewing Health Provider Profile

Similarly, the profile page of a health provider should display details about the health provider, including their average rating. Furthermore, the page should display a list of all affiliated health centres. For each centre, there should be a link to the centre's profile, as well as a button to proceed to booking an appointment with the provider at the centre.

Booking an Appointment

Once the patient clicks a button to proceed onto booking an appointment, they should be presented with a form that consists of the following components.

- A user-friendly widget that lets the patient select the time slots for the appointment. Every day is partitioned into 48 equally-sized time slots, each of which are 30 minutes long in length. The patient can select **available** time-slots e.g, [08:00-08:30]

- A section for the patient to optionally write a brief reason of visit, e.g. "I want to take a blood test".

After selecting the time slots, the patient can click 'book' to finalise the booking. Once the patient has successfully booked the appointment, a confirmation is displayed to the user.

Provide Rating

Both health providers and health centres have an average rating that is displayed in the search results and their profiles. In addition to viewing profiles, patients should be able to rate any provider or centre registered on the HAMS. If a patient rates the same provider/centre more than once, only the patient's most recent rating should count towards their overall average rating.

Viewing Appointments History

A patient should be able to view their own appointments history, a chronological list of their current appointments with the health providers on HAMS. Each entry on the list should contain links to the profile pages of the corresponding health provider and the corresponding health centre.

Health Providers

Viewing Appointments History

Similarly, health providers should also have access to their own appointments history, a chronological list of all current appointments made by patients with the provider. Each entry on the list should contain a link to the corresponding patient's profile page. On the patient's profile page, the provider should be able to see details about the patient's past history.

Viewing Patient History

Registered health-care providers will also be able to use the e-health system. A health-care provider is able to view all their patient appointments. During consultation with a patient, the health-care provider is able to:

- record notes of the patient's visit
- view the past history of the patient (i.e, notes taken at the previous visit).
- record any medication prescribed to the patient

Login and Authentication

The HAMS must only be accessible by the pre-registered users, whose login credentials will be provided in `csv` files. Hence, no registration is required yet. Users should be able to log into the system with their email and password. Once a user has logged into the system, they should be granted access to the rest of the features in system already described above.

Group Project Requirements

Team Registration

For this project, you will need to organise yourself into teams of 4 (no more than 4), and all the team members must be from the same lab session. You will do this under the guidance of your tutor in the week 4 lab. While they will try to accommodate your preferences for teammates, they may have to make adjustments to ensure teams have 4 members.

Once your team has been formed one member from the team will need to register it:

- Go to https://cgi.cse.unsw.edu.au/~cs1531/github/run.cgi/login and sign in
- Click on the 'Group Work' button. This will show you a form that you must fill out
- Select your tutorial from the left drop-down menu and think of a team name (do NOT use any emojis or similar special characters)
- Submit the form.
- Follow the link in the notification to add your other team members. This will create a team within the unsw-cs1531 GitHub organisation

Please ensure that your team is registered by end of week 4.

Creation of Team Repository

Each team will need to create a repository to collaborate on their group project. This can be done as follows:

- Go to https://cgi.cse.unsw.edu.au/~cs1531/github/run.cgi/login and sign in
- Click on the 'Group Project' button.
- Select your tutorial group (on the left drop-down menu) and your team (on the right drop-down menu)
- Click on Import

This will create a repository under the team and also load a set of `csv` files as listed below:

- *patient.csv* - login credentials of patients
- *provider.csv* - login credentials of health-care providers
- *health_centres.csv* - list of health centres on the HAMS system
- *provider_health_centres.csv* - lists the health-centres that a provider works for

Implementation Guidelines

- Keeping mind that an Agile Software Development style has been chosen for this project, your team will be required to build and deliver the project in iterations. Each iteration will deliver a part of the requirements of the project during which the team members are expected to carry out all the SDLC activities, namely analysis, design, coding and testing. At the end of the iteration, you (as a team) will demonstrate to your lab class the functionality implemented during that iteration cycle. Your team must bear in mind that project requirements may be subject to change and enhancements to functionalities may be made at the end of the iteration. You will need to carefully design the solution for your current iteration, such that the solution is extensible to accommodate these changes. Deliverables for each iteration will be outlined at the start of each iteration cycle.

- For this iteration, no sophisticated authentication is required to be implemented. When the URL of the HAM system is specified, e.g., http://localhost:8080/hams, this should launch a login page that prompts the user to enter their email and password

- For this iteration, any kind of persistence mechanism may be used (e.g., Python's **pickle**)

- This project must be designed using **object-oriented design** principles and implemented using **Python/Flask/Jinja2**. It is recommended that students build their front-end using HTML and CSS only. If students choose to use other CSS frameworks (e.g. Material CSS, Bootstrap, etc.) to build the UI for the application then it must be kindly noted that support will be offered by course staff only on the core technology stack namely HTML, CSS, Python, Flask and Jinja2.

- To implement authentication, we strongly recommend that the "Flask Login" extension is used.

- All necessary artifacts for this project e.g., CSV file containing the emails and passwords will be available to you through your repository when you import the project (step b above)

- What has been provided to you is a problem statement; a set of high-level requirements from the customer. Your team must analyse the problem statement and go through a process of eliciting more formal requirements to develop the final set of user-stories and acceptance tests e.g., one of the requirements in the specification states – "Clicking on a particular GP should provide more details about the provider." What details need to be displayed has not been elaborated. This must be elicited as part of your requirements analysis.

- The model solution demonstrated in the lectures was only a guide to demonstrate "one" possible way of implementing the customer's requirements. You are required to design your solution and present in week 12.

- At the end of each deliverable (weeks 6, 9, & 12) tutors will check the team's GitHub repositories to ensure that all members of the team have contributed equally to the project and appropriate branches are created by each team member.

Log Book

You are required to maintain a log book through the entire project that records:

- date of regular, stand-up meetings
- summary of decisions made in stand-up meetings, requirements elicited and key design decisions (hand-written user-stories, CRC cards etc.)
- responsibilities allocated to each team member and tasks to be accomplished for the next meeting
- progress of tasks using a velocity chart (a hand-drawing will suffice, no sophisticated tool needed), summary of decisions made in stand-up meetings
- milestones achieved
- reflection if assigned tasks (decided from last meeting) have been achieved
- any obstacles

Marks will be awarded for the log book

Assessment

You will be assessed on your ability to apply what you have learnt in this course as well as your ability to work in a team and produce a significant piece of software. In cases where the client has not been explicit in their requirements, you will need to make your own design decisions with your team. It is important that you communicate regularly with your team members as well as document your design to avoid confusion about the choices you have made. You will keep a list of assumptions you have made and submit them as part of your final submission in your log book. You will also be asked to justify these decisions when you demonstrate in week 12.

You are expected to use git appropriately by committing consistently and using feature branches to manage significant changes. It is an important part of this project for you (as a team) to manage your own time. You need to start early and work consistently: do not be overly optimistic about what you can achieve (remember you are all doing other courses with other assignments). It is essential for you as a team to monitor your own progress and it is often necessary to revise the plan as you progress.

While it is up to you how to divide the work between you, all members must take on the developer role and work consistently throughout the project.

Each team member generally receives the same mark, but if the moodle peer assessment tool or the logs of your GitHub repository indicate an imbalance in the amount of work done, the total mark may be scaled to match actual contribution.

Group Project Scheduled Deliverables

The planned dates for the different stages of the project deliverables are outlined below.

- Week 6 Lab Session: User Stories Submission (15%)
- Week 9 Lab Session: Demonstration of iteration 1 (25%)
- Week 12 Lab Session: Final Group Project Demonstration (60%)

Week 6 Lab Session: Requirements Analysis and Domain Modelling

Prior to your lab session, your team must schedule collaboration sessions, to have initial high-level visioning discussions during which you will brainstorm to identify epic stories from the customer requirements and their key features, break-down high-level user stories to smaller user-stories and detail each user story to identify key conditions of satisfaction, error-conditions etc. Based on the requirements analysis, produce a domain model as a UML class diagram. Your team is expected to produce:

- High Level **Epic Stories** from the problem statement
- Each epic story broken into **user stories** – Each user-story must define:
    - a unique story identifier (e.g., UC1)
    - a short description of the feature based on Role-Goal-Benefit template (Refer to the RGB model described in the lectures)
    - an estimate for the implementation of the user story in user story points (e.g., UC1 = 2 User story points, where each point = 2.5 hours)
    - priority of implementation
- acceptance criteria for each user story (Refer to the 3 C's model described in the lectures)
- Log book

Please note, the above artifacts will need to be submitted using GIVE by week 6, Sunday 11:59 pm. The submission guidelines for week 6 deliverables will be outlined over the next week.

Week 9 Lab Session: Demonstration of Iteration 1

- The code you wish to submit for this milestone should be in a branch named `release`. You should not push to this branch after the deadline. Your tutor will check to make sure you have not done this. You may continue to push to other branches as you work on your project between the deadline and when you demonstrate to your tutor.
- UML **class diagram** that clearly shows:

    - all classes with attributes, methods and right access modifiers
    - relationships (inheritance, association, aggregation and composition).
    - cardinality must also be indicated

- Your UML class diagram should be a PDF file at the root of your repository named `design.pdf`.
- During this lab session, you will demonstrate the first iteration of your working software. For this first iteration, the customer has requested that they be able to see an instance of working software that meets some of the requirements outlined in the problem statement as outlined below:
- Allow a pre-registered user (patient or health-provider) to log into the system
- Enable a patient to do search (as described above), book an appointment with a particular health-care provider and view their current appointments
- View profile pages of health-care centres and health-care providers
- Health-care providers should be able to view their current appointments.

The clients have agreed that other functionality related to recording of medication prescribed to patients, recording patient history at each visit (i.e recording notes about the visit), maintaining history of past visits (i.e. being able to view past notes), are lower priority and can be viewed in the next iteration. The task for your team is to ***select the necessary user-stories from the backlog of user-stories developed in week 6*** and implement them to meet the customer's goals. The additional document deliverables required for this iteration will be outlined later.

Week 12 Lab Session: Final Group Project presentation

Whatever you have in the `release` branch of your repository at the time of the deadline for this milestone will be considered your final submission. Include the following deliverables:

- The complete working code for your application
- An up-to-date UML class diagram

You will demonstrate your application to your tut group in Week 12. You may be asked to justify your design decisions and explain how you worked as a team.

Peer assessment

To ensure fair allocation of marks, you are required to complete a form on Moodle where you specify how much each team member contributed to the project and any comments you have about them. Information on how you can access this form will be released closer to Week 12.