Booked Library System

Assignment Writeup Guidelines

Software Engineering I

1 Introduction

Software engineering is concerned with numerous activities that stretch far beyond the programming and debugging of code. It is a collective term that encompasses a range of processes designed to provide context and validation to development, and to better prepare for and sustain the production of software. In fact, programming ability may be far down the list of requisite skills when looking to recruit software engineers. A software engineer is an expert in identifying the need for software, designing appropriate solutions, and documenting design efforts.

1.1 Expectation of Written Representation at an MSc Level

It goes without saying that academic writing uses formal vocabulary as compared to the vocabulary used in day-day communication. Your aim should be to make your text as clear as possible – to present your ideas clearly and concisely and to avoid ambiguity or redundancy. The tone of academic writing is formal with clear focus on the issue or topic rather than your opinion. Academic writing is defined by conventions rather than specific rules. Here is a link that would help you build your written skills.

As future software engineering leaders, we aspire our students to work to a high professional standard and use relevant formal vocabulary to engage in a critical/analytical discourse as part of their report specification. It also goes without saying that **grammar and punctuation** need to be impeccable and that you need to be clear on the accuracy of the required vocabulary for this module. If in doubt please refer to the Sommerville textbook or ask the module leader.

1.2 Submission Report

This section details the report deliverable written to high professional standard. At least that is what aspire from our students. Your report must be a maximum of **15 pages**, including diagrams and references. An allowance of +10% is made for this page limit. The marker will not read beyond this. Your report should be written using font size 12 (or other standard size).

Please include appropriate section headings and page numbers in the footer. The core course material is sufficient to pass this assignment well. Reading extra material can be helpful to inspire and refine your solutions. Write from what you know using formal vocabulary and formal definitions, and then move on if needed and if time allows. You are not required to read beyond the core material, but any additional sources of information should be correctly referenced. Your report should include the sections below. The first page needs to be a cover page with you the assignment title **SE1 Assignment Report** and must not include your name anywhere and the first page is not counted as part of the page count of your written report.

Observing Conventions. Please make sure that you use the correct question numbering shown in the brief and **NOT introduce your own question numbering**.

2 Assignment

2.1 Question 1 - Requirements, 40%

Scope of the brief. It is important to establish the scope of the booked scenario, so you can make some reasonable assumptions about the booked scenario.

The **proposed requirements** placed in the booked library brief have several issues. There are deliberate omissions in the scenario so that you can recognize what the issues may be. You also have the room to be creative in your approach. You could come up with anything else that you think would be a good and justifiable addition to the system requirements for book loans.

In a real-world software engineering project, there would be some interaction with a client, who may want more or less involvement in the process of the system. In this assignment, the role of the client is played by members of the teaching team, who have prepared the brief provided and are now passing decisions to you.

- **Q1a.** Identify **five** issues with the proposed functional/non-functional requirements in the Booking brief. We want you to reflect and identify what these issues might be and write about them; provide justifications for each point. These issues need to be specific and not based on your opinion. Once you have identified an issue, you will need to explain using reasoning why it is an issue - i.e. what is wrong with it. [**10%**]

  [**Hint:** After reading the requirements, **reflect** and think if the requirement has any omissions, inaccuracies, is it valid?, is it realistic? - can it be implemented into a system feature right from the description of it itself?, does it aim to fulfill what the booked brief is asking of it. Does it answer the question of **what** the system is meant to do and **NOT how** the feature is to be developed and coded (this is not part of requirement analysis). These are some hints and this list is not exhaustive, so think carefully and strategically. What we are interested in is your ability to recognize an issue and make a reasoned judgement for it, not simply state that there is an issue

because in your opinion there is an issue.

- **Q1b.** A list of **pertinent system and user requirements** for Functional Requirement 4 (book loans) and be clear about separating them into user requirements and system requirements. The following components are expected as part of your answer. [**15%**].
  - Traceability Matrix (User and System)
  - Conflict Checks - Identified and Resolved.
  - Pre-Post Conditions Identified
  - Dependencies Identified
  - Constraints on the Requirements Identified
- During the weekly class discussions on the Online Forums, we shall show you how to present system and user requirements and give you ample examples.
- Do not include all system/user requirements, just new ones that you have created.

Q1c. In your requirements, you need to demonstrate that you have provided consistent requirements that show a clear and complete distinction between **what the system shall accomplish (requirement)** and **how a feature is implemented(design)**. You must also show a clear distinction between the **system requirements and user requirements**. Design requirements are outside the scope of this brief and therefore you are not expected to write about these.

Measurability, can be shown with the above five and below.

Validity. The requirements reflect the current needs of the users, bearing in mind that these may have changed over time and through the refinement itself. The final proposed requirements need to be at the highest level of refinement.

Consistency. There are no contradictory descriptions or constraints in the document. For example, all sub-requirements of system requirements are also system requirements and the same be the case of user-requirements.

Completeness. The system requirements include all functions and constraints, are ac- curate and without omissions. A traceability matrix with conflict checks, identified and resolved, dependencies, pre-conditions and post-condition checks identified is a good way to measure if a given requirement is complete or not.

Realism. Your system requirements are realistic given technology, schedule, and budget (not relevant in the case of booked brief). The central question you need to ask is if I take this system requirement could it be translated into a system feature in software using a suitable programming language?

Verifiability. Tests can be written to demonstrate the system meets each requirement to reduce potential for disputes and hence you need to demonstrate that each system requirements are a testable system feature.

You will be creating the details of these requirements yourself, so the book loaning and any other associated processes are up to you. You are not expected to modify the existing set of requirements but can indicate if you think your requirements will affect existing ones. This will also be a good way to check if your new requirements have a dependency check, pre-conditions, post-conditions, conflicts and inconsistencies between the new requirements and the ones you have identified.

Measurability can be demonstrated by showing that your requirements have achieved all the of the above as well as meeting the requirements validation criteria set out on the course page. You do need to keep in mind that we do not expect you to demonstrate every single requirement to be measured but you can make a judgement on choosing at least **4** requirements where you have shown that you have measured the requirements. [**15%**].

2.2 Question 2 - Architectural Design, 30%

This section must include

- **Q2a. Diagrams**. An architectural design for the following 3 architectures using suitable notation: Model View Controller, Client-Server Architecture, Layered Architecture. These architectures diagrams need to reflect how you would design them for booked brief. So, the more detailed and specific you can be, the better it is for you.
- **Q2b. Justifications**. A written justification which includes a comparison between above 3 architectures. It is important that you understand how to write justifications; simply describing your viewpoint is not enough.
- **Q2c. Decision.** In this section, you need to state your decision for the choice of architecture that you propose. Make a case for using critical analytical arguments to propose a particular system architecture for booked brief in your conclusion.
- You can compare the architectures on the following criteria: **Availability Reliability, Testability, Scalability, Security, Agility, Fault Tolerance, Elasticity, Recoverability, Performance, Deployment, Learnability**. These are the 12 main characteristics, and I would not expect you to go beyond these but should you have a valid point of comparison then please state what the criteria is for comparison.

  Your work should be a critique rather than simply doing a descriptive writing of your points and you have been shown in the study skills post online and what is the difference between descriptive writing and critical writing. I have explained the criteria here below.

  - **Availability**. How long the system will need to be available (if 24/7 , steps need to be in place to allow the system to be up and running quickly in case of any

failure). So, which of the 3 architectures offer the best availability for the booked brief?

- **Recoverability**. This is dealing with the business continuity requirements (e.g in the case of a disaster, how quickly is the system required to be online again? This will affect the backup strategy and the requirements for duplicated hardware. So which architecture styles will support faster recoverability when it comes to the booked brief?
- **Elasticity**. It is the ability of the system architecture to handle bursts of re- quests/traffic. This means can a particular system architecture cope with an exponential increase in the number of requests, so which of the 3 architectures are suitably elastic for the booked brief?
- **Agility**. Agility is the ability for the system architecture to embrace and respond to change in software processes due to significant changes in the business environment or perhaps even new business demands. So which of the 3 architectures are the most agile when it comes to booked scenario?
- **Learnability**. Learnability is how easy it is for the users to learn to use the software and another expansion of the definition is how the developed software can integrate itself into the wider software systems architecture, so another definition is for the software to learn about its environment in order to become self-configuring or self-optimizing - this is particular the case when new modules are integrated with existing software modules. So which architecture makes it easier for new modules to be easily integrated?
- **Reliability**. A Software architecture's reliability is the probability of failure-free operation of a computer program for a specified period in a specified environment. Reliability is a customer-oriented view of software quality. A software architecture's reliability is measured in the operational environment with excel- lent accuracy. So which of the three architectures would be the most reliable in the case of booked library scenario?
- **Testability**. Software testability is the degree to which a software system or a unit under test supports its own testing. To predict and improve software testa- bility, a large number of techniques and metrics have been proposed by both practitioners and researchers in the last several decades. Which of the 3 archi- tectures support testing with ease and how would tests run on one architecture differ from those on another?
- **Security**. Software security is an idea implemented to protect software against malicious attack and other hacker risks so that the software continues to function correctly under such potential risks. Security is necessary to provide integrity, authentication and availability. Which of the 3 architectures are the most secure in the case of the booked scenario?
- **Deployment**. Software deployment refers to the process of running an application on a server or device. Software deployment refers to the process of making the application work on a target device, whether it be a test server, production

environment or a user's computer or mobile device. How do these 3 architectures affect software deployment when it comes to booked scenario?

- **Fault Tolerance**. Software architecture's fault tolerance is the ability of the architecture to not be affected by the fault, and what it takes for the architecture system to recover from a fault that is happening or has already happened in either the software or hardware or in some component of the system architecture in which the software is running in order to provide service in accordance with the specification. So which of these architectures will be highly fault tolerant when it comes to the booked scenario?
- **Scalability**. Architecture scalability is the flexibility and ability of the architecture to grow or shrink to meet changing demands on a business. Architecture scalability is critical to support growth, but also to pivot during times of uncertainty and scale back operations as needed. So, you need to think which architecture styles support better scalability for the booked scenario?
- **Performance**. Performance is an indicator of how well a software systems architecture and its component meet its requirements for timeliness. Timeliness is measured in terms of response time or throughput. The response time is the time required to respond to a request. It may be the time required for a single trans- action, or the end-to-end time for a user task. For example, we may require that an online system provide a result within one-half second after the user presses the "enter" key. For embedded systems, it is the time required to respond to events, or the number of events processed in a time interval. The throughput of a system is the number of requests that can be processed in some specified time interval. So, how would the 3 mentioned architectures fare when it comes to the booked scenario?

Your justifications should include any **4 out of 12** of the above criteria for comparisons against alternatives. Please be aware that opinionated justifications lacking above mentioned standards for comparison make it harder for us to give you credit for your answers.

Out-of-scope. Hybrid models are out of scope of our syllabus, so please do not discuss these.

Additional Reference Text. The book Software Engineering by Google https://www.amazon.co.uk/Software-Engineering-Google-Lessons-Programming/dp/B08VKJXVHK/ref=sr_1_1?keywords=software+engineering+at+google&qid=1646230724&sprefix=software+en%2Caps%2C64&sr=8-1 is a good resource if you want to refer for some of your questions but we do not expect you to use that as your base textbook.

2.3 Questions 3 - System Models, 20%

This section should include:

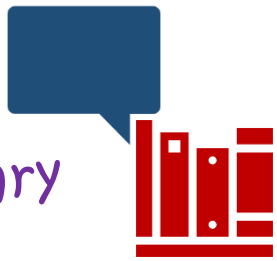
- **Q3a. Diagrams.** Draw three system models applied to the booked scenario. These are: Class Diagrams, Sequence Diagrams and Business Process Diagrams. The system models should use an accurate and relevant notation. It will not be possible to model the entire system in a 15 page document, so be selective and model a few key functionalities. The system models should include suitable and accurate notations for your choice of modelling, use of suitable abstractions and connection to requirements.
- **Q3b. Justification.** The justifications of your system models must include a balanced critique of the aforementioned system models.
- **Q3c. Decision.** In this section, you need to state your decision for the choice of system models that you propose. Make a case for using critical analytical arguments to propose a particular system model for booked brief in your conclusion.

2.4 Presentation Report, 10%

The overall quality of your presentation will be graded for professional presentation, coherence, grammar, punctuation and the quality of your written report.

Software Brief and Requirements

Booked – Not just a library



Overview

Booked is a library based in the centre of London and has been providing book services to local residents since 1957. Due to a declining interest in book loans, the library was due to shut down earlier this year. However, a small group of volunteers have decided to take over the operation of the library and begun to run a range of activities in the library alongside its continued book loan services. They hope to turn Booked into a hub for community activities, and to bring in a range of . instructors and facilitators to introduce new activities to Bath

The lead volunteer, Ms Jekyll, has reviewed the procedures currently in place to book out the library and found them to be paper-based. A diary, kept at the front desk of the library during opening hours, is used to record bookings for the communal spaces. However, some of the facilitators coming in from outside Bath have reported difficulties accessing the diary, or getting to Booked and finding that an activity is already taking place. Volunteers also report that the diary can go missing, or that there can be confusion regarding which activities are booked.

The volunteers have decided that the library should transition to a digital system that allows volunteers and facilitators to manage bookings online, as well as to support other routine operations at the library.

Proposed Solution

The volunteers are looking for a solution to replace the current paper-based booking system. They are concerned about on-site security, so would prefer an externally hosted service rather than protecting and maintaining on-site hardware.

The software should allow volunteers and activity facilitators to book time slots for activities outside of the library opening hours (09:30 – 12:30; 13:00 – 17:00). Some of these activities are sign-up only, so members of the community will need some way to register and reserve a place. Volunteers should also be able to delete and edit bookings.

In addition, the volunteers would like to move the library's book loaning scheme online. The software should be integrated with the booking system so that they have a "one stop shop" for the library. They also think that linking activities to related books would be a good way to increase the number of loans in the library.

Requirements

Your software engineering team has also conducted requirement elicitation activities and generated the following set of functional and non-functional requirements from interviews with the clients. However, they have not generated FR4: “The system shall support book loans”.

R#. Requirement description.

Functional Requirements

FR1 The system shall allow users to register and use an account.

FR1.1 Accounts shall require relevant information.

FR1.1.1 An account shall have a first name.

FR1.1.2 An account shall have a surname.

FR1.1.3 An account shall have an e-mail address.

FR1.1.3.1 The system shall check for and not allow duplicate e-mail addresses when registering.

FR1.1.3.2 The system shall display an error message to users if an e-mail address is already in use.

FR1.1.3.3 The system should prompt the user to login if attempting to register with an e-mail address that is already in use.

FR1.1.4 An account shall have an address.

R.1.1.4.1 An address shall have an address line.

FR1.1.4.2 An address shall have a city.

FR1.1.4.3 An address shall have a postcode.

FR1.1.5 An account shall have a password.

FR1.2 Accounts shall be given a unique membership number.

FR1.3 The system shall require users to login to use its booking features.

FR1.3.1 The system shall require an e-mail address OR membership number to login.

FR1.3.2 The system shall require a password to login.

FR1.3.3 The system shall allow users to reset a password by sending an e-mail to their registered e-mail address.

FR2 Users shall be given appropriate roles.

FR2.1 Lead Volunteers shall be able to assign roles to users.

FR2.1.1 Lead Volunteers shall be able to assign multiple roles to users.

FR2.2 Lead Volunteers shall be able to assign the Volunteer role to a user.

FR2.2.1 Lead Volunteers shall inherit the permissions of the Volunteer role.

FR2.3 Lead Volunteers shall be able to assign the Facilitator role to a user.

FR2.4 Lead Volunteers shall be able to assign the Member role to a user.

FR3 The system shall be able to support activity bookings.

FR3.1 The system shall allow permitted users to create a booking.

FR3.1.1 Facilitators shall be able to create a booking.

FR3.1.2 Volunteers shall be able to create a booking.

FR3.1.3 A booking shall require relevant information.

FR3.1.3.1 A booking shall have an activity name.

FR3.1.3.2 A booking shall have an activity date.

FR3.1.3.3 A booking shall have an activity time.

FR3.1.3.4 A booking shall have an activity location.

FR3.1.3.5 A booking shall have an activity facilitator.

FR3.1.3.6 A booking shall have a list of registered members.

FR3.1.3.7 A booking shall have a maximum number of attendees.

FR3.1.4 The system shall not allow a booking to be made if the location is in use at the selected time.

FR3.2 The system shall allow permitted users to edit a booking.

FR3.2.1 Volunteers shall be able to edit a booking.

FR3.3 The system shall allow permitted users to delete a booking.

FR3.3.1 Volunteers shall be able to delete a booking.

FR3.3.2 Registered members should be e-mailed when a booking is deleted.

FR3.4 Bookings shall only be made for times outside of the library opening hours.

FR3.4.1 Bookings that are made for times during the library opening hours shall receive an error message.

FR3.5 The system shall allow Members to register for an activity.

FR3.5.1 The system shall display a list of bookings to Members.

FR3.5.1.1 The system shall display a list of bookings sorted by date (upcoming listed first)

FR3.5.2 The system shall allow a Member to register for a selected activity.

FR3.5.2.1 The system should not allow a Member to register for an activity if they are already registered for another activity at the same time.

FR3.5.2.2 A confirmation dialogue should be presented to the Member to confirm registration.

FR3.5.2.3 An e-mail should be sent to the Member when registration is complete.

FR3.5.2.4 An e-mail should be sent to the Facilitator when registration is complete.

FR3.5.2.5 The system should not allow a Member to register for an activity if the there are no more places available.

FR4 The system shall support book loans.

FR4.1 ….

Non-Functional Requirements

NFR1 User passwords shall be secure.

NFR1.1 Passwords shall contain 6-9 characters.

NFR1.2 Passwords shall contain at least one number.

NFR1.3 Users shall be required to change password once every 12 months.

NFR2 The system should show users an up-to-date list of activities.

NFR2.1 The system should refresh its list of activities at least every 5 minutes.

NFR2.2 The system should refresh the number of available spaces on an activity at least every 10 seconds.

NFR3 The system should be reliable.

NFR3.1 The system should be available at least 23 hours a day.

NFR3.2 Any downtime to the system should be limited to the hours between 00:00 and 06:00 GMT.

NFR4 The system should be usable.

NFR4.1 75% of users should be able to register for an activity in under 2 minutes.

NFR4.2 Tooltips should be available for every user interface widget.

NFR4.3 Help messages should be available on every page.

NFR5 The system should be portable.

NFR5.1 The system should be available on multiple platforms.

NFR5.1.1 The system should be available on Windows versions 7 onwards.

NFR5.1.2 The system should be available on iOS versions 11 onwards.

NFR5.1.3 The system should be available on Android versions 5.0 onwards.

NFR5.2 The system should be written in a universally available language.