SE1 Assignment Report

QUESTION 1 - REQUIREMENTS

Q1A. IDENTIFY FIVE ISSUES WITH THE PROPOSED FUNCTIONAL/NON-FUNCTIONAL REQUIREMENTS IN THE BOOKING BRIEF

ISSUE #1: Sub-functional requirements of FR2 ("Users shall be given appropriate roles") may cause 4 different sub issues in the Booked application.

There are 4 different sub functional requirements explained in the FR2. All of them have issues that can cause a severe problem in the management of the Booked application. FR2 mainly focuses on giving appropriate roles to the users. Lead Volunteers shall be able to assign multiple roles to users such as Member, Facilitator, and Volunteer. Also, Lead Volunteers shall inherit the permissions of the Volunteer role. The issue is that authorization details of these roles are not clearly explained in the FR2 and in FR3 requirement description document.

What is the authorization and permission level of these roles? Can the Volunteer role make all C.R.U.D operations in the application? If they can, assigning the volunteer role to a user can be dangerous. What are the exact roles of Volunteer, Facilitator, Member? A Member who has a volunteer role can deliberately delete all the books in the application. In this possible scenario all books in the system need to be found in a backup database. If there is no backup database application can lose all books inside of it. Similarly, what is the authorization level of Facilitator role and Volunteer role. Authorization levels of all users need to be explained completely and consistently under FR2 and FR3 with different sub requirements.

ISSUE #2: FR3.2.1 ("Volunteers shall be able to edit a booking.") might cause three different sub issues in the Booked application.

Specifically, FR3.2 and FR3.3 shall cause severe problems. According to FR3.2, Volunteers shall be able to edit a booking, however, the scope and limitations of the edit function are not clearly defined. Which of the details of a booking (given in FR3.1.3) are allowed to be edited by Volunteers? Do they change the time or location of the meeting without sending a notification to all registered Members? Do they add any constraint to meetings, such as people who can attend who has just PhD degree without asking facilitators? In FR3.3 Volunteers shall be able to delete a booking and system sends email to registered members. Even though email is sent to members, there should be time limitation in this scenario. The system should not allow to cancel a booking within the last 12 hours before meeting. Otherwise, a member who lives outside of Bath may travel to Booked in Bath for the booking which was already cancelled. At least they could have bought tickets for travel planning to this activity, in this case, the system should support online meetings to attendees instead of onsite meetings.

ISSUE #3: 3rd issue might cause three different sub issues in the booked application related to NFR5.2 ("The system should be written in a universally available language")

"The system should be written in a universally available language" is an ambiguous requirement that can imply various meanings. What is the description of universally available language? Is it implied that the documentation of the system be written in a universally available language (e.g. English), or does it mean that the application shall be developed in a universally available

programming language? If programming language is implied, this application can be developed in various programming languages. One can use one of the many programming languages in the backend and various languages in the front-end development. Even one can write the system as a mobile application by using Objective C or React OS, it can be developed as an embedded system. All these confusions can cause a problem in the development team. Hence, this non-functional requirement needs to be clearly defined again.

ISSUE #4: NFR4.1 ("75% of users should be able to register for an activity in under 2 minutes") is not a clear requirement.

The term "registration for an activity" is not clearly defined, therefore may lead to confusion in the development phase. Is it the entire time spent from logging in to Booked application up until to clicking on Register button, or is it just the duration of clicking on Register button? If it is the second one, the registration activity (clicking the button and getting a successful return message) should be completed in less than 1 second. If the first definition is true for registration, a ballpark of 30 seconds should be the maximum time for this kind of activity. Also, 75 percent is not a very high ratio. For a successful adoption and high acceptance rate by users, it needs to at least be more than 95 percent. Otherwise, application can be useless, and people may stop using application because of performance problems. When a person wants to register, purchase or like something in a modern application (amazon, Netflix, Instagram) it can be completed within 1 second.

ISSUE #5: In NFR1, definitions of requirements are not clearly stated.

In NFR1.1, while creating a password, are the characters requested as in ASCII table or are they just numbers or letters? NFR1.2 is not enough to make a strong password. Passwords should include at least one number, one capital letter, one small letter, 2 consecutive numbers and one special character to be more secure. NFR1.3 can be an option that can be chosen by user. User can choose 3 to 6 to 9 to 12 months password chance duration. 12 months of password change period is a long duration for a reliable system.

Q1B. A LIST OF PERTINENT SYSTEM AND USER REQUIREMENTS FOR FR4

NEW USER REQUIREMENTS FOR FR4

Table 1 User Requirements with Pre- and Post-conditions for FR4

Code	User Requirement	Pre-Condition	Post-Condition
FR4	The system shall support book loans.		
FR4.1	The system shall allow permitted users to loan books		
FR4.1.1	Facilitators shall be able to search, reserve, loan, purchase, request book(s)		
FR4.1.2	Lead Volunteers shall be able to search, reserve, loan, purchase, request book(s)		
FR4.1.3	Volunteers shall be able to search, reserve, loan, purchase, request book(s)		

Code	User Requirement	Pre-Condition	Post-Condition
FR4.1.4	Members shall be able to search, reserve, loan, purchase, request book(s)		
FR4.1.5	Lead Volunteers can set a ban on users for book loans and reservations	User has not returned a book within 2 months OR the user does not return books on time for 3 consecutive times	User is not eligible for book loan
FR4.2	The system shall allow permitted users to search for books online	User is logged in	Search log is updated
FR4.2.1	User shall be able to search by keywords	User is logged in	Search log is updated
FR4.2.2	User shall be able to search by author	User is logged in	Search log is updated
FR4.3	The system shall list the search results	User is logged in and search button is clicked	Search result is displayed
FR4.3.1	Search result list shall be able to be sorted by relevance		
FR4.3.2	Search result list shall be able to be sorted in title alphabetical order		
FR4.3.3	Search result list shall display the expected return date of already loaned books		
FR4.4	The system shall allow user to reserve the selected book for loan		
FR4.4.1	If the selected book is available, user clicks the "Reserve" button to reserve the book for the next 24 hours	- User is logged in and eligible for loan - Book is available - Reserve button is clicked	Selected book's status is changed to "Reserved" for next 24h
FR4.4.2	Reservation shall be automatically cancelled if the book is not collected from the library by user in 24h	- Book is reserved - User does not collect the book within 24 hours	Selected book's status is changed to "Available"
FR4.4.3	If the selected book is already reserved or loaned, user can click the "Notify me" button to get in queue for loan	- User is logged in and eligible for loan - Book is reserved or loaned by another user	- User is logged to queue list for email notification
FR4.4.4	An alert email is sent to the user when the loaned book is returned to the library or the reservation on the book has expired	- A book has been returned or reservation has expired - There are users in the queue list for this book	- Book's status is changed to "Available" - Email log and queue list is updated

Code	User Requirement	Pre-Condition	Post-Condition
FR4.5	The system shall limit number of books to be loaned or reserved by one user at the same time		
FR4.5.1	A message shall be displayed to user when the loan or reservation limit has been reached	- loaned and reserved books limit is exceeded - User clicks "Reserve" button	- A message is prompted to user, indicating the limit has been reached.
FR4.6	The system shall display the list of books loaned, reserved and purchased by the user	User is logged in	
FR4.6.1	The system shall display the list of loaned books which are overdue for return	User is logged in	
FR4.6.2	The system shall display a list of all loaned and reserved books sorted by return date	User is logged in	
FR4.6.3	The system shall display a list of purchased books/articles	User is logged in	
FR4.6.4	The system shall send a daily reminder email to users for upcoming books	User has unreturned books due in 7 days	Email log is updated
FR4.7	Users shall be able to buy pdf books, e-books and articles from the library	User is logged in Book/article is available in digital format	
FR4.7.1	User shall click "Purchase digital version" button	Book/article is available in digital format	
FR4.7.1.1	Volunteers should be able to scan and convert paper-based books and articles to pdf versions	Title is not available in pdf format	Title is created in pdf format for sale
FR4.7.2	System shall ask the user for Credit Card	Purchase Digital Version button is clicked	
FR4.7.2.1	User shall enter Credit Card Type		
FR4.7.2.2	User shall enter Credit Card Owner		
FR4.7.2.3	User shall enter Credit Card Number		
FR4.7.2.4	User shall enter CVV number		
FR4.7.2.5	User shall enter Card Expiry Date		
FR4.7.3	User clicks "Complete Purchase" button		
FR4.7.4	After clicking Complete Purchase button, a verification code is sent to the members phone	Member has a valid phone number in system	SMS is sent
FR4.7.5	User enters verification number in the system and transaction continues	Verification is successful	

Code	User Requirement	Pre-Condition	Post-Condition
FR4.7.5.1	If any of the payment details is not complete or validated, system displays a warning message to the user	- Complete Purchase button is clicked - Payment details are incomplete, OR - Payment details could not be verified	Purchase status is marked as Unsuccessful
FR4.7.5.2	If payment is successful, a confirmation message is displayed to the user and digital invoice is sent to user email address.	- Complete Purchase button is clicked - Payment details are complete, and - Payment details are verified	- Purchase status is marked as Successful - Email log is updated
FR4.8	The system shall allow for new book addition requests		
FR4.8.1	Users shall be able to fill out a book request form online and submit the form to the library	- User is logged in - Form is submitted	- Requested books list is updated

Table 2 Conflicts & Resolutions of FR4

Code	Conflicts & Resolution
FR4.1.5	- Conflict: A lead volunteer might violate the book return rules - Resolution: automation of user banning
FR4.4.3	- Conflict: User who currently loaned the book may click and extend the loan duration - Resolution: User who currently has the book, cannot click Notify Me

Table 3 Constraints of requirements of FR4

Code	Constraints
FR4.4.3	A user can click Notify Me only once during a loan period
FR4.7.1.1	Page count validation is needed
FR4.7.5	Verification code needs to be entered within 120 sec.
FR4.7.5.2	Digital invoice creation shall be completed within 5 minutes

NEW SYSTEM REQUIREMENTS FOR FR4

Code	System Requirement Detail
NFR4.4	The "Complete purchase" button shall complete the transaction under 1 seconds for all users.
NFR4.5	Accessibility options (for visually impaired) should be available optionally for all users

Code	System Requirement Detail
NFR4.6	Search engine for book search should perform the search and return a result page in under 5 seconds for 95% of all users, upon clicking the search button
NFR6	The system should show users an up-to-date list of books
NFR6.1	The system should refresh its list of books' loan, purchase and reservation status and user balances at least every 5 minutes.
NFR7	The system should be compliant with national accounting procedures
NFR7.1	The system should create an invoice with a unique invoice number for each purchase under 5 min.
NFR8	The system should be integrated with finance system
NFR8.1	Invoices generated in finance software should flow to and be available in Booked application

TRACEABILITY MATRIX AND DEPENDENCIES FOR NEW USER AND SYSTEM REQUIREMENTS

D: Dependent R: Relationship

USER REQUIREMENTS TRACEABILITY MATRIX

Code	FR4.1	FR4.2	FR4.3	FR4.4	FR4.5	FR4.6	FR4.7	FR4.8	NFR4.4	NFR4.5	NFR4.6	NFR6.1	NFR7.1	NFR8.1
FR4	R	R	R	R			R	R						
FR4.1														
FR4.1.1														
FR4.1.2														
FR4.1.3														
FR4.1.4														
FR4.1.5												D		
FR4.2	D		R											
FR4.2.1														
FR4.2.2														
FR4.3		D									R	D		
FR4.3.1														
FR4.3.2														
FR4.3.3												D		
FR4.4	D													
FR4.4.1	D				D							D		
FR4.4.2												D		
FR4.4.3	D											D		
FR4.4.4	D											D		
FR4.5												D		

Code	FR4.1	FR4.2	FR4.3	FR4.4	FR4.5	FR4.6	FR4.7	FR4.8	NFR4.4	NFR4.5	NFR4.6	NFR6.1	NFR7.1	NFR8.1
FR4.5.1														
FR4.6												D		
FR4.6.1												D		
FR4.6.2												D		
FR4.6.3												D		
FR4.6.4												D		
FR4.7									D					
FR4.7.1														
FR4.7.1.1														
FR4.7.2														
FR4.7.2.1														
FR4.7.2.2														
FR4.7.2.3														
FR4.7.2.4														
FR4.7.2.5														
FR4.7.3														
FR4.7.4														
FR4.7.5														
FR4.8	D													
FR4.8.1														

SYSTEM REQUIREMENTS TRACEABILITY MATRIX

Code	FR4.1	FR4.2	FR4.3	FR4.4	FR4.5	FR4.6	FR4.7	FR4.8	NFR4.4	NFR4.5	NFR4.6	NFR6.1	NFR7.1	NFR8.1
NFR4.4							R							
NFR4.5														
NFR4.6			R											
NFR6	R		R	R	R	R								
NFR6.1	R		R	R	R	R								
NFR7							R							
NFR7.1							R							
NFR8							R							
NFR8.1							R							

Q1C. MEASURABILITY

4 of the user requirements are investigated in terms of Measurability criterion (Validity, Consistency, Completeness, Realism, Verifiability).

Code	Validity		Consistency		Completeness		Realism		Verifiability	
FR4.1	"the volunteers would like to move the library's book loaning scheme online"	✓	No conflict	✓	Shown in Traceability Matrix, pre- and post-conditions, conflict checks	✓	Can be translated into a system feature	✓	Can be tested	✓
FR4.2	Searching is a subprocess of book loaning and a valid requirement	✓	No conflict	✓	Shown in Traceability Matrix, pre- and post-conditions, conflict checks	✓	Can be translated into a system feature	✓	Can be tested	✓
FR4.3	Listing search results is a subprocess of book loaning	✓	No conflict	✓	Shown in Traceability Matrix, pre- and post-conditions, conflict checks	✓	Can be translated into a system feature	✓	Can be tested	✓
FR4.4	Reserving a book before collecting it is a subprocess of loaning	✓	No conflict	✓	Shown in Traceability Matrix, pre- and post-conditions, conflict checks	✓	Can be translated into a system feature	✓	Can be tested	✓

QUESTION 2 - ARCHITECTURAL DESIGN

Q2A. DIAGRAMS

1- MVC (MODEL VIEW CONTROLLER) ARCHITECTURE

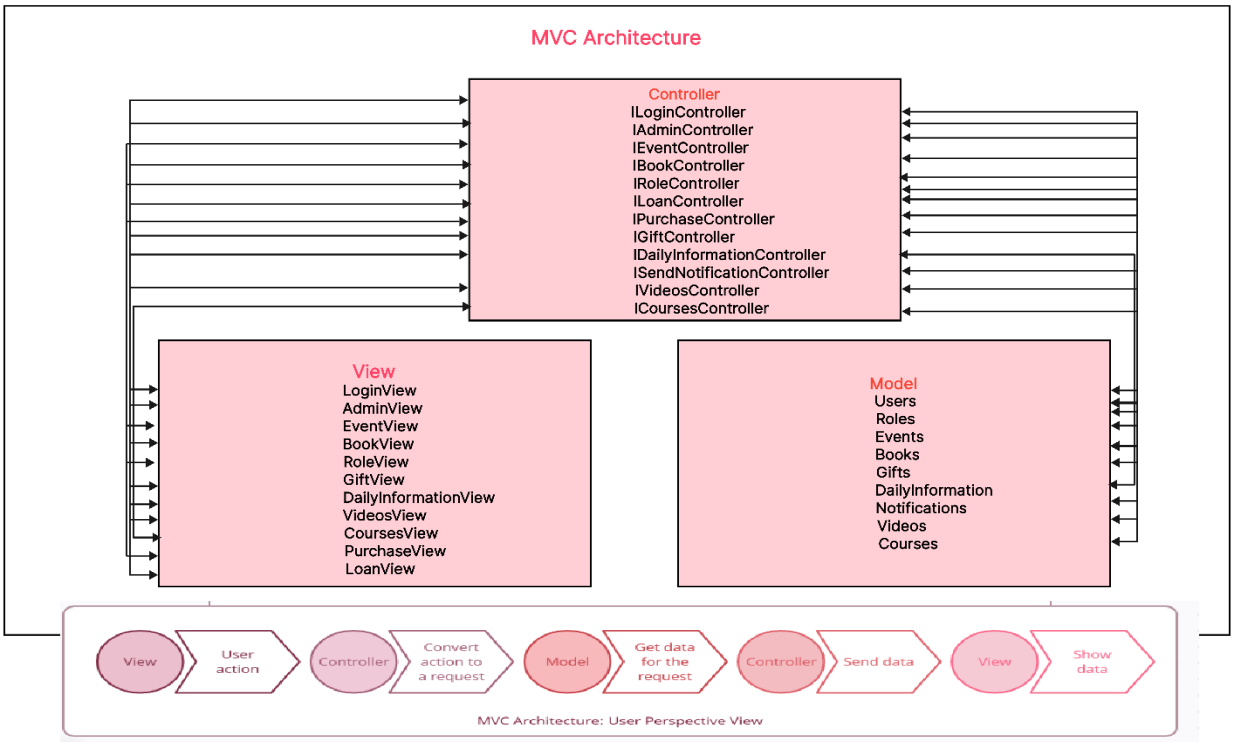


Figure 1 MVC Diagram

The model component within the MVC architecture encapsulates the entities, representing distinct objects within the system. Each entity comprises various pertinent attributes, colloquially referred to as columns. For the sake of clarity, these attributes are conceptualized as objects. As depicted in the aforementioned diagram, the Model component facilitates the transmission or persistence of data to the underlying database, while Controllers oversee the management of Create, Read, Update, and Delete (CRUD) operations. Views, on the other hand, serve to render graphical user interfaces within the application. Certain views may serve as subcomponents within the overarching system architecture. For instance, 'giftView' functions as a subordinate entity within the broader 'bookView'. Users engage with the system by initiating actions such as purchasing a book via interactive elements presented on the interface, subsequently marking the transaction as completed. In such instances, methods within the 'IPurchaseController' interface handle the user request, effectuating the necessary data manipulation within the 'books' entity. This entails the instantiation of new transactions and the population of relevant attributes, including indicators such as 'isPurchased' and 'isGifted'.

Reference: Fowler, M. (2002). Patterns of Enterprise Application Architecture. Addison-Wesley.

2- CLIENT-SERVER ARCHITECTURE

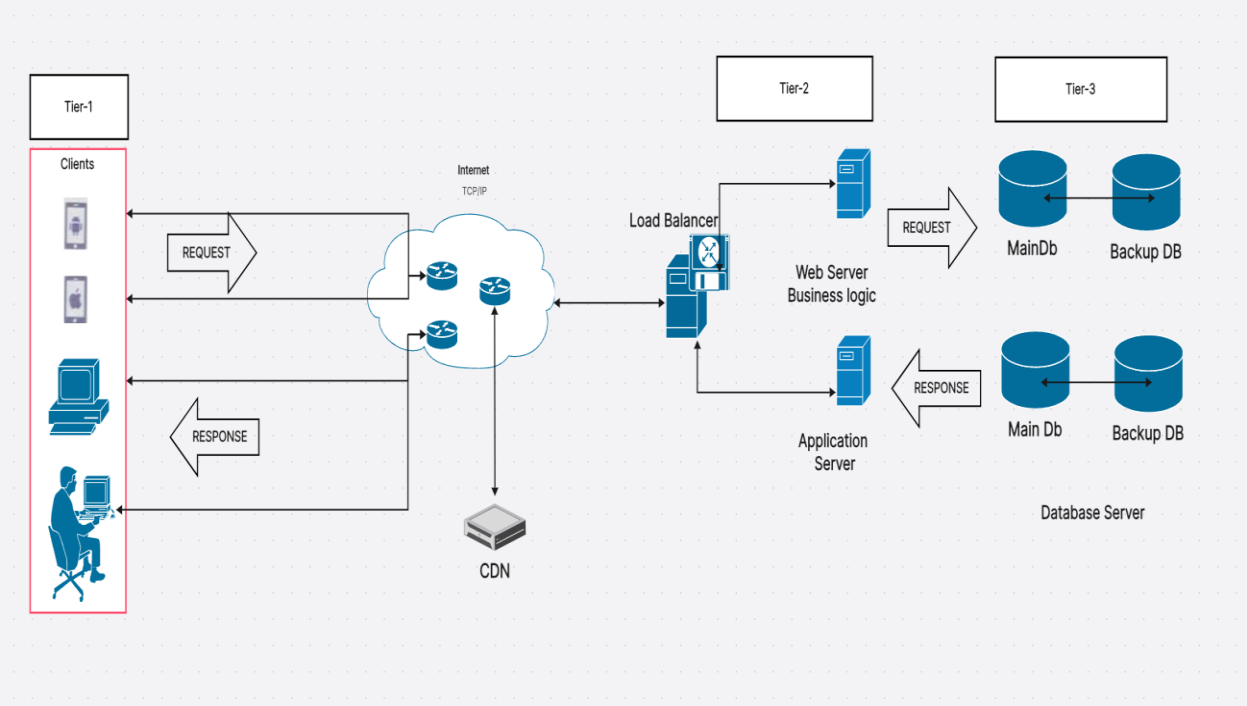


Figure 2 Client-Server Diagram

In the 3-Tiered client-server architecture, various components such as services, objects, and modules inherent to the Model-View-Controller (MVC) architecture find application. For the sake of brevity, a comprehensive reiteration of the design is omitted. Initially, the client initiates a

request for data from the server, typically by entering the application's URL in a web browser. This request is transmitted over the Internet and registered by a load balancer, which then appropriately directs the request to an available server. Subsequently, the server undertakes the processing of these client requests. Moreover, the server engages in querying the database to retrieve the requisite information. Upon receiving the queried data from the database, the server processes it and subsequently transmits the processed data back to the client. This iterative process persists until the client ceases to issue requests.

3- MONOLITHIC LAYERED ARCHITECTURE

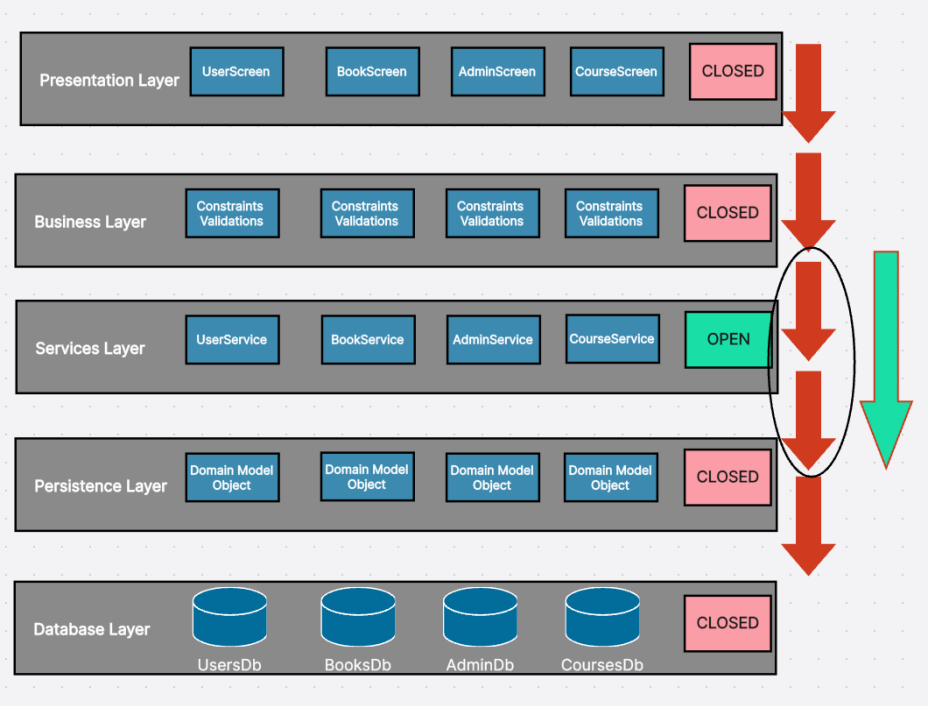
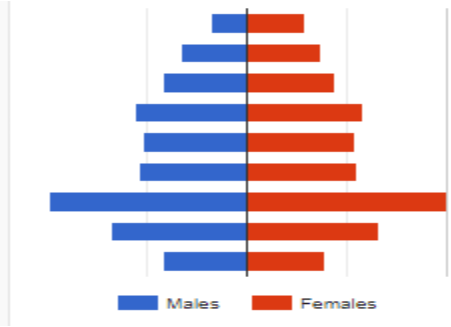


Figure 3 Monolithic Layered Architecture Diagram

In layered architecture, requests flow downwards through the structure, typically starting at the top layer. The concept of open and closed layers arises when attempting to access lower layers. A closed layer necessitates passing through it to reach the layer below, leading to potential confusion. To address this, the option of opening layers, such as the services layer, can be considered. This allows developers the flexibility to bypass certain layers, when necessary, as advocated by Mark Richards for providing guidance to development teams. Layered architecture offers advantages in terms of cost-effectiveness and simplicity due to its monolithic nature, making it suitable for scenarios with budget and time constraints. Hence, it remains a practical choice under such circumstances.

Q2B. JUSTIFICATIONS

I recommend MVC architecture as the preferred architectural pattern for booked application due to its numerous advantages over client-server and layered architecture. Firstly, MVC is easily comprehensible, with many object-oriented programming languages offering MVC model development environments that require minimal code. For example, in .NET, developers can utilize .NET Razor Pages MVC project, swiftly establishing the MVC model infrastructure. Similarly, in Java, the Spring Boot environment enables the creation of an archetype web application, facilitated by Maven, which streamlines the MVC project setup. In contrast, implementing booked application with client-server architecture lacks readily available infrastructure and necessitates manual code development for layered architecture. This demands considerable expertise, typically requiring the guidance of a senior developer to construct and instruct junior developers in code implementation. Consequently, such endeavors incur both time and financial resources. Furthermore, the MVC model presents cost efficiency as it allows deployment from a powerful single computer, obviating the need for extensive server procurement. This approach aligns with the findings of citypopulation.de, which underscores the economic benefits of MVC implementation.



Age Distribution (C 2021)	
80+ years	4,712
70-79 years	6,962
60-69 years	8,540
50-59 years	11,431
40-49 years	10,522
30-39 years	10,778
20-29 years	19,831
10-19 years	13,289
0-9 years	8,031

In Bath, a city with a population of 100 thousand, residents typically find local events for book purchases or attendance. An optimistic projection suggests that around 50 percent of the user base, approximately 50 thousand members (with a variance of 10 percent), would actively engage with this application. Given its relatively modest scale compared to larger enterprises in sectors such as banking or e-commerce, minimal server capacity is expected, obviating the need for substantial investments. Developers engaged in client-server projects are required to navigate web sockets and TCP/IP network layers. Conversely, in prewritten MVC modules like Razor Pages, developers primarily focus on writing APIs and managing CRUD operations through JSON file parsing. With an abundance of available courses, developers can readily grasp MVC design in object-oriented programming languages, expediting development by adapting example modules to their codebase. This inherent efficiency in MVC significantly accelerates both the development and management of the application.

Q2C. DECISION

1. **Testability**: MVC facilitates testability by separating the application into three distinct components: Model, View, and Controller. Each component can be tested independently,

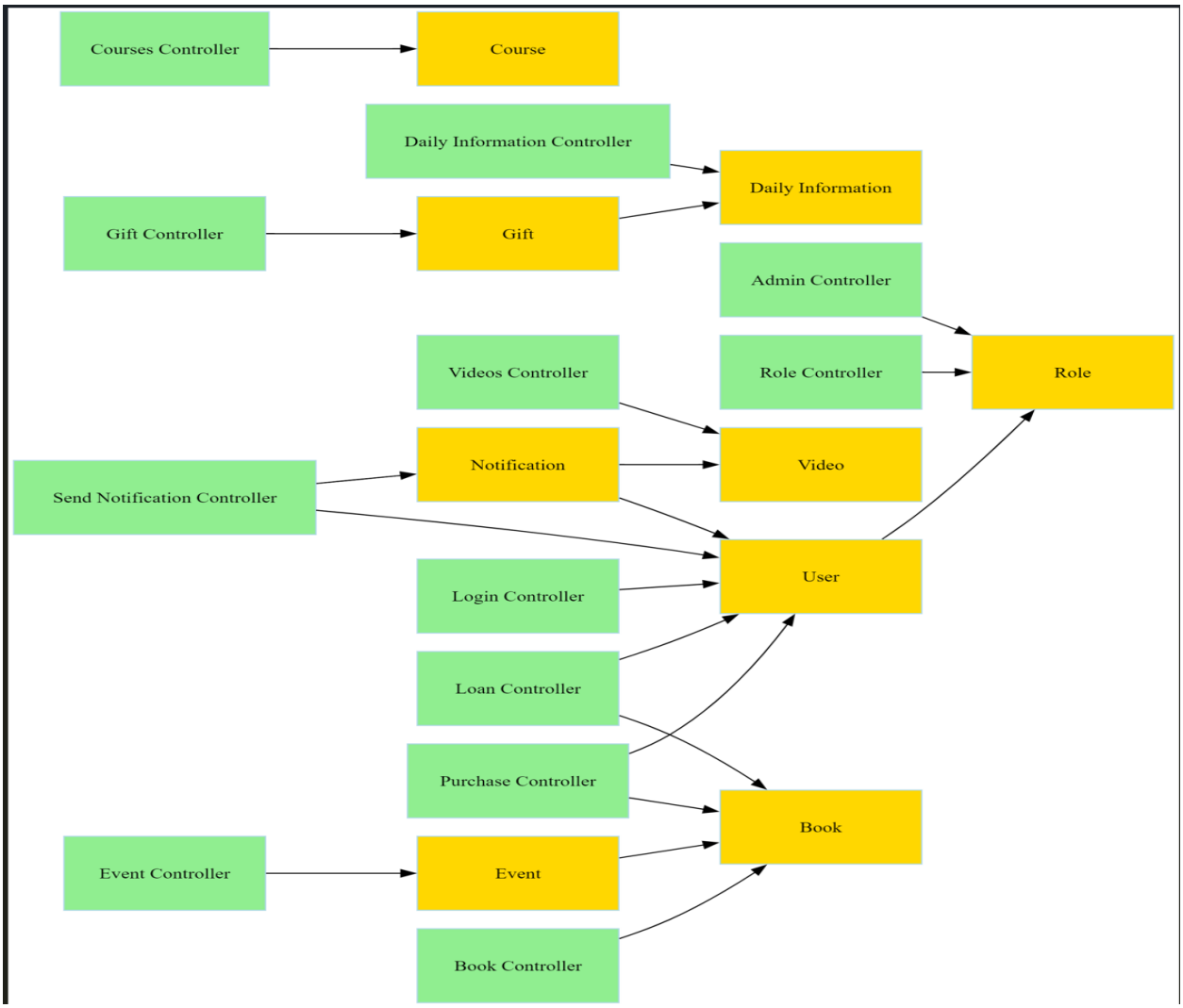
making it easier to write unit tests and perform integration testing. This modularity enhances software quality by allowing developers to identify and fix issues more efficiently.

2. **Scalability:** MVC supports scalability by enabling developers to scale individual components as needed. For instance, during periods of high user traffic, developers can scale the Controller or View components independently to handle the increased load. This flexibility ensures that the application can accommodate growth without sacrificing performance.
3. **Security:** MVC promotes security through its clear separation of concerns. By separating the presentation layer (View) from the business logic (Controller) and data access (Model), MVC helps mitigate security vulnerabilities such as injection attacks and unauthorized data access. This architectural pattern enhances application security by enforcing strict boundaries between different components.
4. **Agility:** MVC enhances agility by facilitating rapid development and iteration. The separation of concerns allows developers to make changes to one component without impacting others. This modularity streamlines the development process, enabling faster iterations and easier adaptation to change requirements. As a result, MVC enables teams to deliver high-quality software more quickly and efficiently.

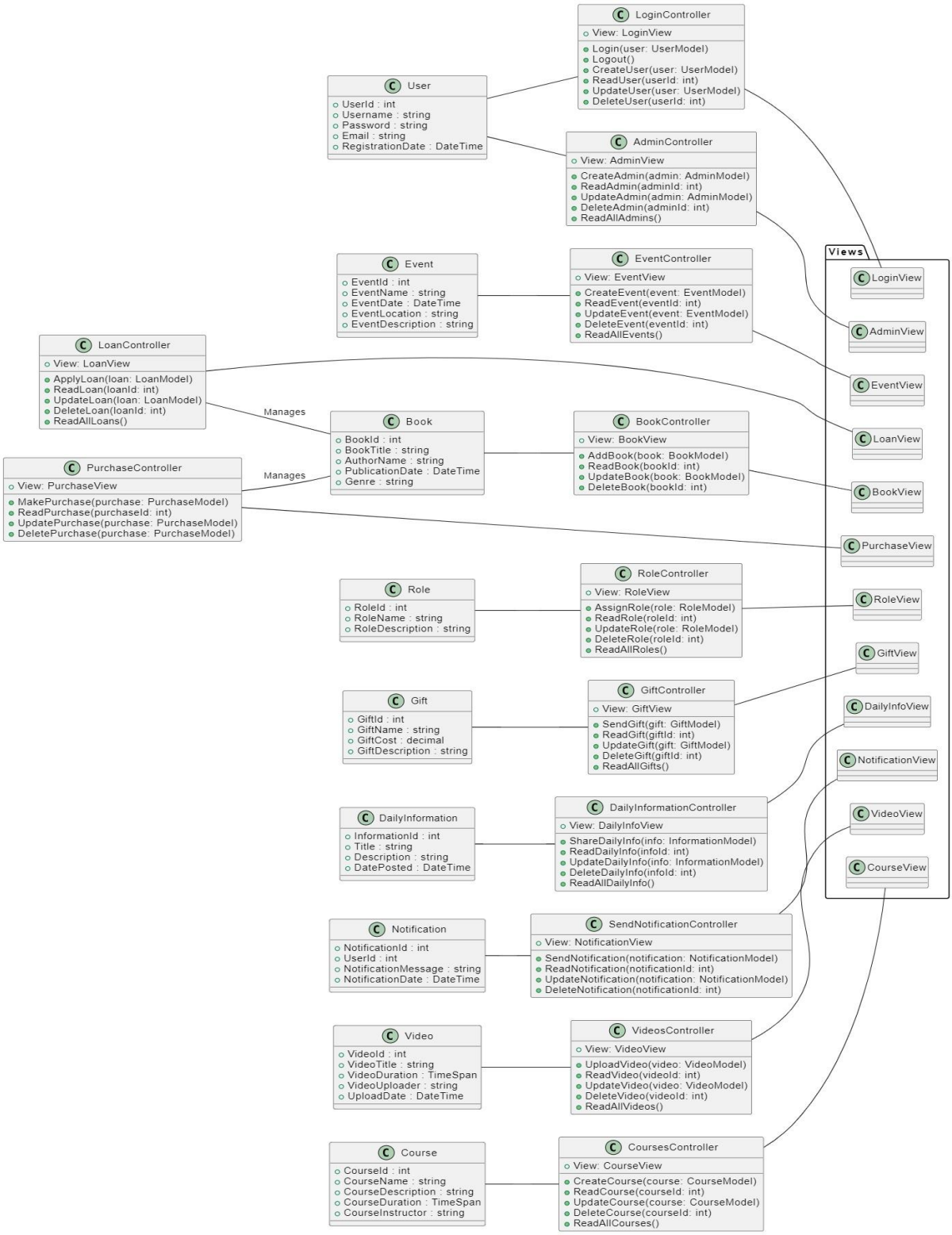
QUESTIONS 3 - SYSTEM MODELS

Q3A. DIAGRAMS

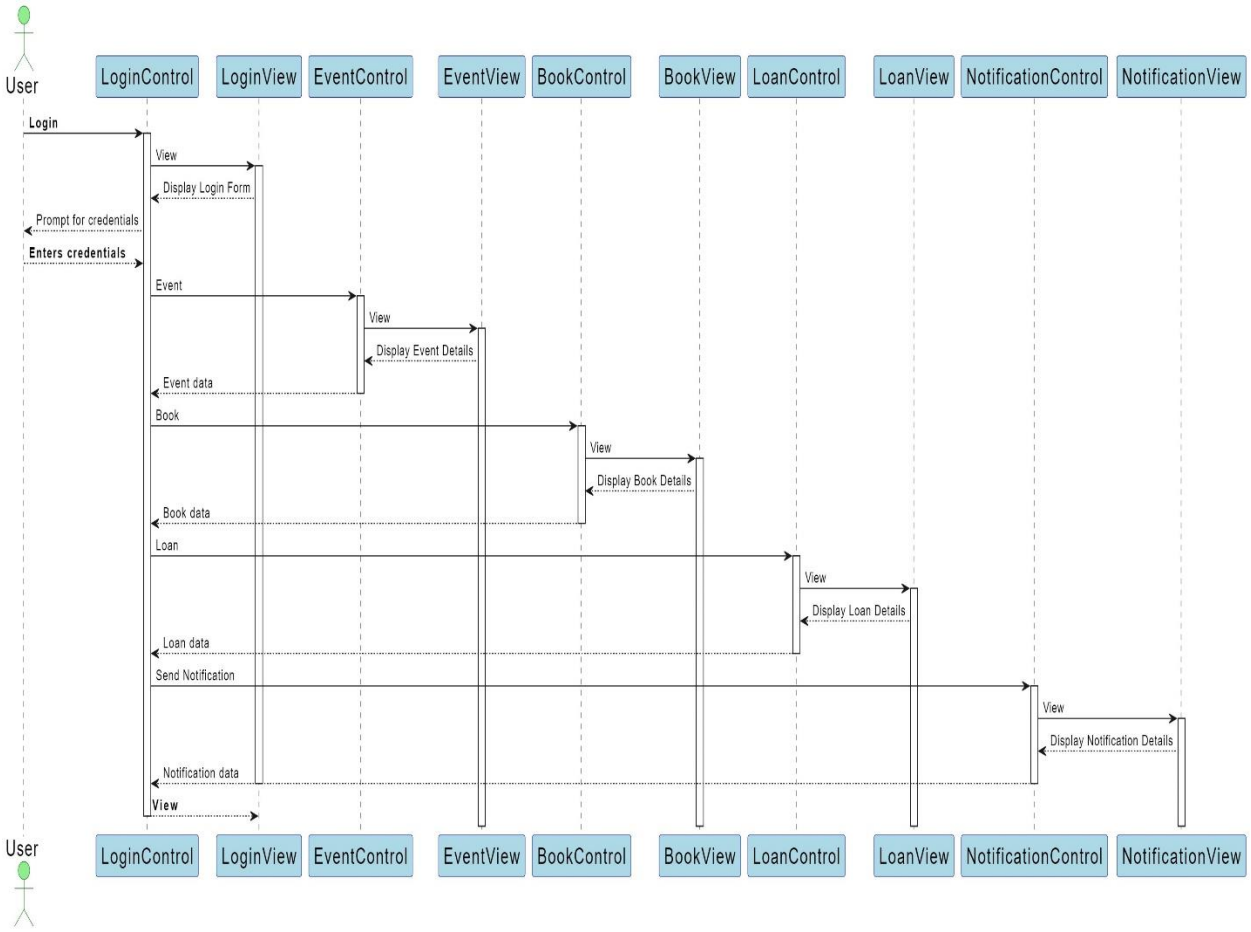
Business Process Diagram



UML Class Diagram



Sequence Diagram



Q3B. JUSTIFICATIONS

Sequence Diagrams, Business Process Diagrams, and UML Diagrams each offer unique perspectives on system modeling and analysis Sequence diagrams, focusing on dynamic object interactions, provide insights into runtime behavior and message passing within software systems. In contrast, Business Process Diagrams emphasize the flow of activities within software systems and emphasize the flow of activities within organizations, aiding in process optimization and cross-functional communication. UML Diagrams, as a standardized language for software modeling, encompass a broader range of perspectives, including structure, behavior, and interaction, facilitating comprehensive system documentation and design. While Sequence diagrams excel in depicting object-level interactions, Business Process Diagrams specialize in organizational workflows, and UML Diagrams provide a holistic view of software systems. Together, they form a versatile toolkit for understanding, communicating, and designing complex systems across various domains.

Q3C. DECISION

In proposing a system model for Booked Brief, the choice of UML diagrams should be driven by the specific aspects of the system that need to be modeled or communicated effectively. Considering the nature of Booked Brief, which likely involves interactions between users, scheduling of briefings, management of bookings, and possibly a database for storing relevant information, a combination of UML diagrams would be beneficial to capture the system comprehensively.

Firstly, to depict the structural aspects of the system, Class Diagrams are essential. They would help in identifying the key entities such as Users, Briefings, Bookings, and any other relevant classes along with their attributes and relationships. This would provide a clear understanding of the static structure of the system.

Secondly, to illustrate the dynamic behavior of the system, Sequence Diagrams would be invaluable. They would allow us to visualize the sequence of interactions between users, the system, and other components involved in the booking process. This includes actions such as user authentication, briefing selection, booking confirmation, and notification generation.

Additionally, Use Case Diagrams would be beneficial for identifying and communicating the functional requirements of the system from the users' perspective. They would help in capturing the various actors (such as Admins, Customers) and their interactions with the system, including actions like browsing available briefings, making bookings, and managing bookings.

Furthermore, considering the potential complexity of state transitions in the booking process (e.g., from available to booked, pending to confirmed), State Machine Diagrams could provide clarity on the various states the system can be in and the events that trigger transitions between these states.

In conclusion, a combination of Class Diagrams, Sequence Diagrams, Use Case Diagrams, and possibly State Machine Diagrams would provide a comprehensive and well-rounded system model for Booked Brief. Each diagram serves a distinct purpose in capturing different aspects of the system, ensuring clarity, understanding, and effective communication among stakeholders. By utilizing these UML diagrams, Booked Brief can be thoroughly analyzed, designed, and communicated, laying a solid foundation for its development and implementation.

REFERENCES

Fowler, M. (2002). *Patterns of Enterprise Application Architecture*, Addison-Wesley Professional.

Gamma, E. (1994), *Design Patterns: Elements of Reusable Object-Oriented Software*, Addison-Wesley Professional

Monson-Haefel, R. (2009), *97 Things Every Software Architect Should Know,* O'Reilly Media.

Richards, M. (2022), *Software Architecture Patterns,* 2nd Edition, O'Reilly Media

Bath (Bath and North East Somerset, South West England, United Kingdom) - population statistics, charts, map, location, weather and web information. (n.d.). https://www.citypopulation.de/en/uk/southwestengland/bath_and_north_east_somer/E63005227__bath/

PerfMatrix. (2023, August 2). Performance Testing vs Performance Engineering. PerfMatrix. https://www.perfmatrix.com/performance-testing-vs-performance-engineering/

Richards, M. (n.d.). My publications | Developer to Architect | Mark Richards. https://www.developertoarchitect.com/books.html