

**College of Engineering**

**COSC 448: Mobile Programing 2**

**Final Project**

**Weightage 35%**

**Submission: 20 April 2024**

**Poster Session: 21 April 2024**

Project Title: BookEase SwiftUI App

Description:

**BookEase** is a versatile booking application developed using SwiftUI, offering users a seamless platform to make reservations for various services such as restaurants, hotels, events, and appointments. The application encompasses a range of functionalities including a splash screen for an engaging start, asynchronous loading of images for service listings, fetching booking data from a server in JSON format, adopting MVVM architecture for robust code organization, leveraging User Defaults for personalized preferences, Swift Data for efficient data management, integration of Maps and Location Services for venue navigation, photo capture and display for user profiles or service images, and robust authentication and security features to safeguard user information.

By encompassing these features, **BookEase** will offer users a comprehensive booking experience across various services, ensuring ease of use, security, and reliability.

Assessed Learning Outcomes:

• A1. Demonstrate advanced knowledge and understanding of mobile applications development framework and process.

• A2. Use specialized programming tools to design, implement, test, debug and deploy mobile applications.

• B1. Develop mobile applications that solve a real-world problem using a suitable programming language.

• B2. Use different approaches to design a mobile application and communicate design ideas to the target audience.

• C1. Demonstrate responsibility and leadership within a team to achieve the goal of having a mobile application solution to a real-world problem.

Core Features:

1. **Splash Screen using DispatchQueue**
   * Design an attractive splash screen using SwiftUI.
   * Implement DispatchQueue to display the splash screen for a specific duration.
   * Transition to the main view of the application after the splash screen period.
2. **Asynchronous Image Loading from URL**
   * Develop a ViewModel to manage asynchronous loading of images from URLs.
   * Populate SwiftUI views with images fetched asynchronously from URLs.
3. **Asynchronous Data Fetching and JSON Format**
   * Establish a network service to asynchronously fetch booking data from a remote server.
   * Utilize Codable to parse JSON data received from the server.
   * Update ViewModel to handle asynchronous data fetching and update SwiftUI views accordingly.
4. **MVVM in SwiftUI for a Better Architecture**
   * Architect the application using the MVVM pattern for improved separation of concerns.
   * Define models to represent booking entities and data structures.
   * Implement ViewModels to manage business logic and data flow independently of views.
5. **User Defaults**
   * Implement User Defaults to store user preferences and booking-related settings.
   * Enable functionalities for saving and retrieving data using User Defaults.
6. **Swift Data**
   * Utilize Swift Data for efficient storage and retrieval of images or other binary data.
   * Implement methods to store and retrieve images or user data using Swift Data.
7. **Maps and Location Services**
   * Integrate MapKit to display maps within the application interface.
   * Utilize Core Location for retrieving the user's current location.
   * Implement features such as venue location display and navigation.
8. **Capturing and Displaying Photo**
   * Develop functionality to capture user photos using the device's camera or photo library.
   * Display captured photos within the application, e.g., for user profiles or service images.
9. **Authentication and Security**
   * Implement robust authentication mechanisms using Firebase Authentication or similar services.
   * Ensure secure transmission of user data over the network using HTTPS.
   * Implement data encryption techniques to protect sensitive user information stored locally.

**Requirements:**

1. **Project Document**

A comprehensive document that outlines various aspects of your project. It should include:

• **Project Overview:** A summary of what the project is about, its objectives, and its scope.

• **Technical Specifications:** Details about the technologies used (SwiftUI, data storage methods, etc.), architecture, and any specific libraries or frameworks.

• **Design Documentation:** Sketches, wireframes, or mockups of the user interface.

• **Implementation Details:** A step-by-step explanation of how the functionality is implemented.

• **Testing and Validation:** Information on how the application was tested (e.g., unit tests, user testing).

• **Challenges and Solutions:** Any challenges faced during the development process and how they were resolved.

• **Future Enhancements:** Ideas or plans for future updates or improvements.

1. **Xcode Project File**

This is the actual software project file that you will work on using Xcode. It should include:

• **Source Code:** All the Swift code files, including the SwiftUI views for the user interface, data models, and any other logic.

• **Assets:** Images, icons, or any other media used in the project.

• **Configuration Files:** Project settings, plist files, etc.

• **Dependencies:** If you're using any external libraries or packages, they should be included or referenced here.

1. **Poster**

This is likely a visual representation of your project, designed to summarize and showcase it effectively. It should include:

• **Project Title:** Clearly displayed at the top.

• **Visual Elements:** Screenshots of your app, diagrams, or any other relevant images.

• **Key Features:** Bullet points or short sections highlighting the main features of your project

• **Technologies Used:** Icons or names of the technologies (SwiftUI, etc.).

• **Contact Information:** Include your contact information or QR code linking to the project or your portfolio.

**Project instructions:**

The following instructions should be followed:

1. The project team (Maximum Three students) will hold weekly status meetings, chaired by the project manager (the choice is one of your team members).
2. In your project your contribution must be recorded, i.e. each participant should describe his/her contribution and must be documented and submitted in the project report as an *appendix*.
3. Submit your project report and your presentation slides by 21th April.
4. Each team project requested to present the project (15-20 minutes) on 22th April.
5. The total score is 35, making up 35% of the overall course weightage.

Rubric:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Criteria** | **Scale** | **Maximum Points** | **CLOs** | **Comment/s** |
| **Functionality** | Does the application fulfill its core purpose of enabling users to make bookings services? | 2 |  |  |
| Are all the essential features implemented, including splash screen, image loading, data fetching, MVVM architecture, user preferences, data management, maps, photo capture, authentication, and security? | 2 |  |  |
| **User Experience (UX):** | Is the application intuitive and easy to navigate? | 2 |  |  |
| Are the UI elements well-designed and consistent throughout the application? | 2 |  |  |
| **Performance:** | How responsive is the application to user input? | 2 |  |  |
| Does the application handle errors gracefully and recover from failures effectively? | 2 |  |  |
| **Scalability and Maintainability:** | Is the codebase organized and maintainable, following MVVM architecture principles? | 2 |  |  |
| Is the code well-documented and easy for other developers to understand and contribute to? | 2 |  |  |
| **Data Management:** | Are user preferences and settings persisted correctly using User Defaults? | 2 |  |  |
| Is data stored efficiently using Swift Data? | 2 |  |  |
| **Integration of External Services:** | Are external services such as MapKit and Core Location integrated seamlessly into the application? | 2 |  |  |
| **Authentication and Security:** | Is user authentication implemented securely, protecting user credentials and data? | 2 |  |  |
| **Error Handling and Resilience:** | How does the application handle unexpected errors or network failures? | 1 |  |  |
| Is there a robust error-handling mechanism in place to notify users and recover from errors gracefully? | 1 |  |  |
| **Testing:** | Has the application been thoroughly tested for functionality, usability, and performance? | 1 |  |  |
| **Poster Presentation:** | * **Excellence presentation** **(5/5)** with appropriate explanation (5 points) * **Good presentation (3-4)**   with some mistakes   * **Bad presentation (1-2)** with many mistakes | 5 |  |  |
| **Teamwork and Collaboration:** | * **Excellent (3):** Actively and effectively collaborates within the team, demonstrating strong communication and contributing significantly to the project. * **Good (2):** Participates well in the team environment, making effective contributions to the database development project. * **Fair (1):** Engages to some extent, but collaboration and contributions may be limited or sporadic. | 3 |  |  |
|  | **Total** | **35** |  |  |