Assignment 3

Instructions and Grading Criteria

- This is an **individual** assessment. Please review the college's **Academic Integrity Policy** to ensure that you are completing your work in an academically honest manner.
- The majority of grades are assigned based on the correct completion of the required functionality.
- The user interface of your application must be reasonably polished, easy to understand, and readable. Use reasonably pleasant colors and typography. Ensure your font sizes are legible.
- In addition to the required functionality, learners are expected to use the coding conventions demonstrated in class, meaningful variable naming, and clearly organized code. Comments are helpful but not required.

Submission Checklist

For your submission to be graded, provide:

- ☐ a **zip** file of your project
  - Create an XCode project named named **A4-FIRSTNAME**, where you replace **FIRSTNAME** with your name. For example: **A4-TOMMY**
  - After completing your project, create a zip of your project. Name the zip file: **A4-FIRSTNAME.zip**, where you replace **FIRSTNAME** with your name. Example: **A4-TOMMY.zip**

- ☐ a **screen recording** with an **app demo** and **project explanation**. 8-15 min max.
  - Upload your as an unlisted link to Youtube
  - Paste the link to the submission comments.
  - App demo explanation: Click here
  - Project explanation: Click here

Academic Integrity

- This is an individual assessment.
- Permitted activities: Usage of Internet to search for syntax only; usage of course materials
- Not permitted:
  - Communication with others (both inside and outside the class)
  - Discussion of solution or approaches with others; sharing/using a "reference" from someone
  - Searching the internet for full or partial solutions
  - Sharing of resources, including links, computers, accounts
  - Usage of generative AI tools, example: ChatGPT, CoPilot, the built-in XCode AI code generation tools, etc

Problem Description

Create a multiscreen application that simulates borrowing books from a library:

- Login Screen: Enables the user to login or create a new library account
- Books List Screen: Displays a list of books in the library

The app must meet these technical requirements:

- Data persistence must be implemented using CoreData
- User interface must be created with Storyboards (no SwiftUI)

1. Database

The app's database must consist of the following entities:

a. **User Entity:** Every user has a username and password
b. **Book Entity:** Every book has a book's title, author, and borrower. (A borrower is the username of the person who borrowed the book. If the book is not borrowed, then the borrower field is set to an empty string)

2. Login Screen

Here's how the screen must be implemented:

When the screen loads:

- Check if there are users and books in the app's database.
- If yes, then do nothing.
- If not, then load:
    - Two (2) users of your choice
    - Three (3) books of your choice. By default, set each book's **borrower** field to an empty string.

Display a login form

Provide a form for the user to login to the application. The form must have:

- ☐ Textboxes to enter a username and password
- ☐ Login button

Login Button

When the button is pressed, attempt to login the user with the provided credentials.

- If the username and password match a user in the database, navigate to the BooksList screen.
- Otherwise, display an error message.

Books List Screen

This screen contains a UI for displaying a list of library books. The screen must contain:

- ☐ Welcome message displaying the *name* of the currently logged in user.
- ☐ Tableview showing the list of books in the database. For each book, display the book's title, author and a message with the book's status
    - If the book does **not** have a borrower, then the message is "Available, click to checkout."
    - If the current user borrowed the book, then the message is "Checked out, click to return"
    - If the book was borrowed by another user, then the status is "Unavailable"
- ☐ Logout button

Clicking on a book in the tableview

The user interacts with a book by tapping on a row in the tableview.

- Clicking on an "Available" book will update the book's borrower field to the currently logged in user and refresh the tableview.
- Clicking on an "Unavailable" book should display an appropriate message.
- Clicking on a "Checked Out" book must reset the book's borrower field to an empty string and refresh the tableview.

Logout Button

When the button is pressed, simulate a user logout by doing the following:

- Remove the current screen from the navigation stack
- Go back to the Login screen

HINT: Consider using the `popViewControllerAnimated` function.

App Demonstration Checklist

During your app demonstration, show the following use cases:

- ☐ Login with an invalid credentials. What happens?
- ☐ Login with valid credentials. What happens?
- ☐ Using user #1, borrow a book. What happens?
- ☐ Using user #2, attempt to borrow the book that user #1 checked out. What happens?

If your app does not work or crashes, show what happens if you run the application. Ensure you clearly show the resulting crash/error message.

If your app is partially complete, demonstrate only the parts that you implemented.

END OF ASSESSMENT