ASSIGNMENT 2

Instructions

- Using Visual Studio and Xamarin Forms, develop a cross platform mobile application per the specifications in this assessment. Regardless of whether your development machine has the necessary software to test on iOS, your submission must account for the iOS safe area.
- After developing the application, select **ONE** mobile platform that you will test your application on (iOS or Android). You are **NOT** required to test on both platforms.

    a. **iOS** apps will be tested on an **iPhone 15** simulator.
    b. **Android** apps will be tested on a **Google Pixel 7** emulator.

    .
- If you are unable to use these emulators on your computer, then please note which emulator you used in the submission comments.

Grading Criteria

- The majority of grades are assigned based on the correct completion of the required functionality.
- 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.
- The user interface of your application must be reasonably polished, easy to understand, and readable. Use reasonably pleasant colors and typography.

Submission Checklist

For your submission to be graded, you must provide a **zip** file of your project, and a **screen recording** demonstrating the functionality you implemented.

During the screen recording, you must verbally narrate/explain what you are doing while you are doing it (do not assume the instructor will understand what you are doing simply by watching you click things on the screen)

1. Creating Your Project

- Projects must be created using Visual Studio and the Xamarin framework
- When creating your project, name the project: **Library-firstName**.
- Regardless of which platform you are actually testing your application on, the project must contain the .Android and .IOS project folders as shown in class.

2. Before you submit:

- In Visual Studio, run **"Build > Clean xxxx"**, where xxxx is the name of your project. Do NOT re-run the application. Running this command will remove extraneous, compiler generated files from the project.
- After running Build > Clean xxx, create a zip file containing all project code.
- Name the project **Library-firstName.zip**. No 7zip or rar files are accepted.

3. Creating Your Screen Recording

- In the screen recording, demonstrate the app running on an emulator (or real device) and the functionalities you implemented.
- During the screen recording, you must verbally narrate/explain what you are doing while you are doing it (do not assume the instructor will understand what you are doing simply by watching you click things on the screen)
- You are also required to explain your code in detail.
- Max 10 mins.

4. Submitting the assignment:

- In the submission comments, note which mobile platform your application is designed to be run on.
- **Submit your zip file and screen recording** separately (do not zip them together).
- If your screen recording is too large, upload your screen recording to **OneDrive** and ensure that the link is set to: "Anyone with the link can view". Paste a link to the recording in the **submission comments**.

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

PROBLEM DESCRIPTION

In this project, you will build a cross platform mobile application that enables the user to interact with a list of books in a library.

The application consists of 2 screens:

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

The user interface of the application is provided on the course webpage.

Login Screen

1. The login screen consists of a form that enables the user to sign in with their username and password. You must use appropriate form field elements and keyboard types.

2. When the user presses LOGIN, perform validation on the form fields. If validation fails, display an appropriate error message. If a valid username and password was provided, navigate the user to the Books List screen.

3. There are two valid users in this library system:

username	password
peter	1234
mary	0000

All other username/password combinations are considered invalid.

4. The users should be statically coded into your Login Screen logic (not persisted to a database)

Books List Screen

1. This screen shows a **welcome message to the currently logged in user** and a **list of books** in the library. The list of books should be loaded from a database *immediately before* the BooksList page is visible on the screen.
2. When the user selects a book from the list, the screen should display the status of the book, ie:
   - "The book is available for borrowing."
   - "The book is checked out by [another user]" (replace [another user] with the name of the user)
   - "You checked out this book!"

3. The user can check out or return a book by using the **context actions** on the currently selected book.
4. **Borrowing a book** (checkout): A book can be borrowed if no one else has currently borrowed the book.
5. **Returning a book**: A book can only be returned by the user who borrowed it.
6. After borrowing or returning the book, the app should display a **popup message** informing the user of whether the borrow/return was successful. If not successful, display an appropriate error message (ie: the reason why the operation could not be completed).
    - In Xamarin, popup messages can be displayed using a **DisplayAlert** control: https://learn.microsoft.com/en-us/xamarin/xamarin-forms/user-interface/pop-ups#display-an-alert
7. If the person presses "back" on the Books List screen, they should be returned to the **Login Screen**. All login screen from fields and error messages must be cleared of any prior content.

Modelling a Book

1. Books must be modelled as a custom class. Each book has the following properties:
    - ISBN-10 number
    - Title
    - Author
    - Borrowing status (either the book is checked out or not). **By default**, **this is set to false**.
    - Borrower (The name of the user who checked out the book, if any). **By default, this is set to an empty string**.

Data Persistence

1. The application must implement data persistence using SQLite.
2. You must use the `sqlite-net-pcl` library.
3. The database must be prepopulated with a minimum of 4 books.
4. The initial creation of the database and population of its books should occur *immediately before* the login screen is visible on the screen.

Navigation

1. Navigation between pages must be performed using hierarchical navigation.

END OF ASSESSMENT