GitHub Username: EddyMM

Bookworm

Description

- ➔ Bookworm is an Android Application that lists the **best selling books** available based on the New York times ratings
- ➔ It ensures that book enthusiasts are kept **up to date on the best quality books** to read
- ➔ It allows readers to **bookmark books** based on the synopsis
- ➔ It can keep track of the readers bookmarks online so that they never lose their favorite books on uninstallation or switching devices

N/B: Based on the New York Times API
https://developer.nytimes.com/docs/books-product/1/overview

Intended User

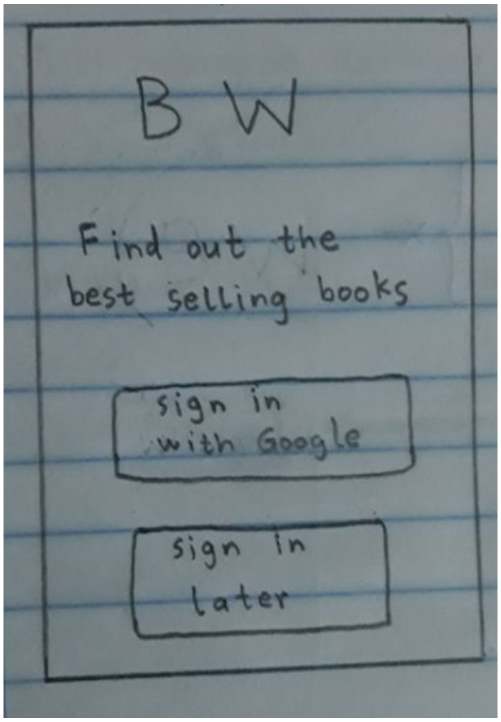
Book enthusiasts

Features

- ➔ **List** best selling books
- ➔ **Bookmark** books based on readers assessment of synopsis
- ➔ **Keep track** of bookmarks **online** through their GMail Account

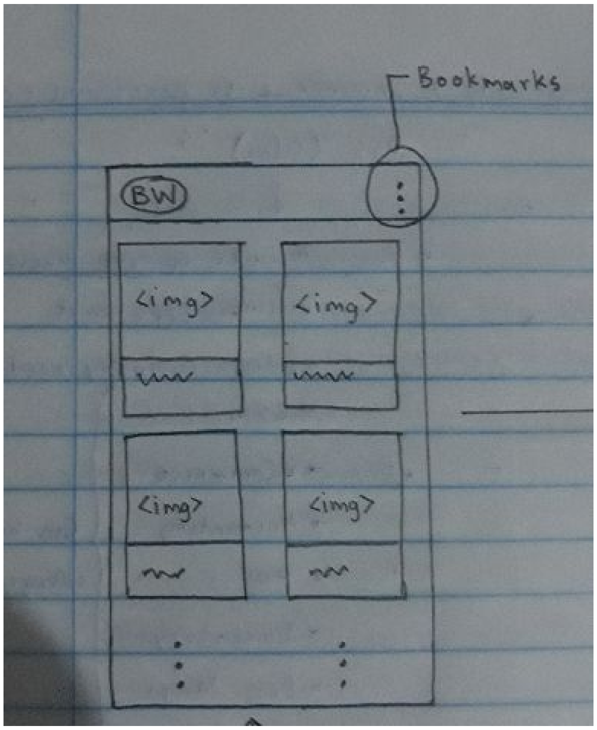
User Interface Mocks

Introduction Screen

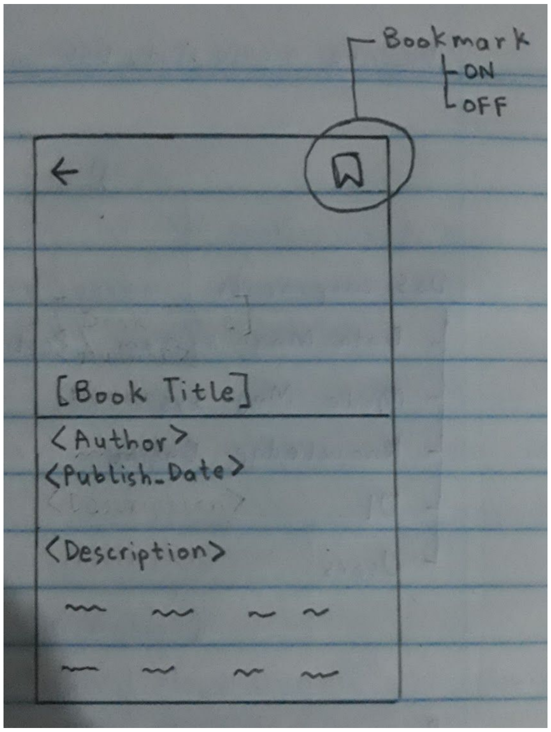


- ➔ Introduction to the app.
- ➔ Describes what the app does and an optional sign in
- ➔ Should show up once per install

Books List Screen



- ➔ List of best selling books
- ➔ Each book is represented using a card view
- ➔ Bookmarked books list should be accessible from the menu



Book Detail Screen

- ➔ Details about the book
- ➔ Option to bookmark the book for later reference

Key Considerations

How will your app handle data persistence?

- ➔ Bookmarked books will be stored using **Firebase Realtime Database** to ensure the reader has **online and offline access** to them with **syncing** being handled
- ➔ **User account** login credentials will be managed by **Firebase Authentication Service** so need to store login details locally

Describe any edge or corner cases in the UX.

Google sign-in dialog box will show up when user tries to bookmark a book when they are **not currently logged in**

Describe any libraries you'll be using and share your reasoning for including them.

- ➔ **Retrofit:** The books will be fetched from online servers that expose the information using a REST API hence a REST API client library such as retrofit would be suitable
- ➔ **Picasso:** The books have have images of their cover. Rendering these images and handling caching is easier using Picasso
- ➔ **Firebase SDK:** To store bookmarks offline and online, Firebase Realtime DB is appropriate. Furthermore, user authentication is simple when using Firebase Authentication service. These services can be accessed using the Firebase SDK library.

Describe how you will implement Google Play Services or other external services.

Firebase Realtime Database

- ➔ Add the library in gradle
- ➔ Create book models
- ➔ Create a firebase DB Reference with persistence enabled to benefit from offline storage
- ➔ Use LiveData DataSnapshots and Event listeners to sync data between the UI and Data Layer

Firebase Authentication Service

- ➔ Add the library in gradle
- ➔ Get instance of FirebaseAuth
- ➔ Use GoogleSignIn option
- ➔ Use instance to keep track of user login and logout as bookmarks are stored using their login id

Next Steps: Required Tasks

Task 1: Project Setup

- Add appropriate libraries (Support, Design)
- Set a common support library version

Task 2: Implement Introduction Activity

- Build UI for the Introduction Activity
- Perform Google Sign In by using Firebase Authentication API
  - Requires initializing the firebase instance

Task 3: Books List Activity

- Build UI for the Books List Activity
  - Involves handling API calls for the books using the API: https://developer.nytimes.com/docs/books-product/1/overview
- Use a GridLayout for the books

Task 4: Book Detail Activity and Fragment

- Build UI for the Book Detail Activity
- Use CoordinatorLayout and AppBarLayout
- Load book title as App Bar Image
- Include the bookmark menu icon
  - Changes color based on ON and OFF
  - Save book details using Firebase Realtime DB on ON
  - Remove book details from Firebase Realtime DB DB on OFF

Submission Instructions

- After you've completed all the sections, download this document as a PDF [File → Download as PDF]
  - Make sure the PDF is named "**Capstone_Stage1.pdf**"
- Submit the PDF as a zip or in a GitHub project repo using the project submission portal

If using GitHub:

- Create a new GitHub repo for the capstone. Name it "**Capstone Project**"
- Add this document to your repo. Make sure it's named "**Capstone_Stage1.pdf**"