GitHub Username: mihajul

Book Companion

Description

Have you ever read a book but forgot most of its contents after a few days/weeks ?
Or maybe want a refresher on a book but you do not want to re-read the whole thing again ?

This app acts like an notepad in which you store and organize notes when you read a book.
The Book Companion app helps you takes notes while reading a book and store them in an organized fashion indexing your notes by book, chapter, topic and keywords.

Intended User

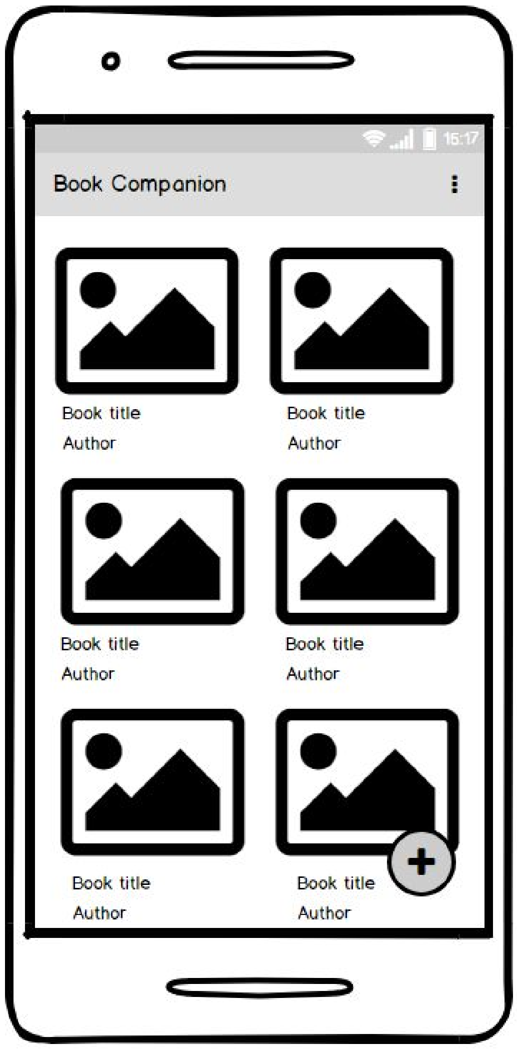
This app can be used by anybody who reads books, but mostly students or people which read specialized literature that they want to summarize or remember afterwards.

Features

- Keep track of your books collection
- Take text or image notes from the book contents
- Be able to review the stored information efficiently
- App will have a widget which will display a random quote from your notes.

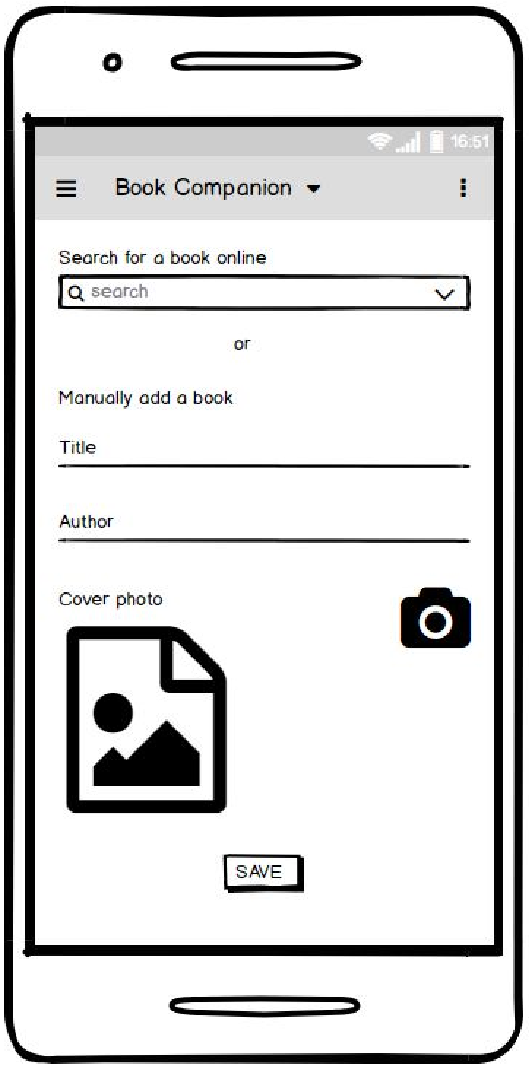
User Interface Mocks

Screen 1



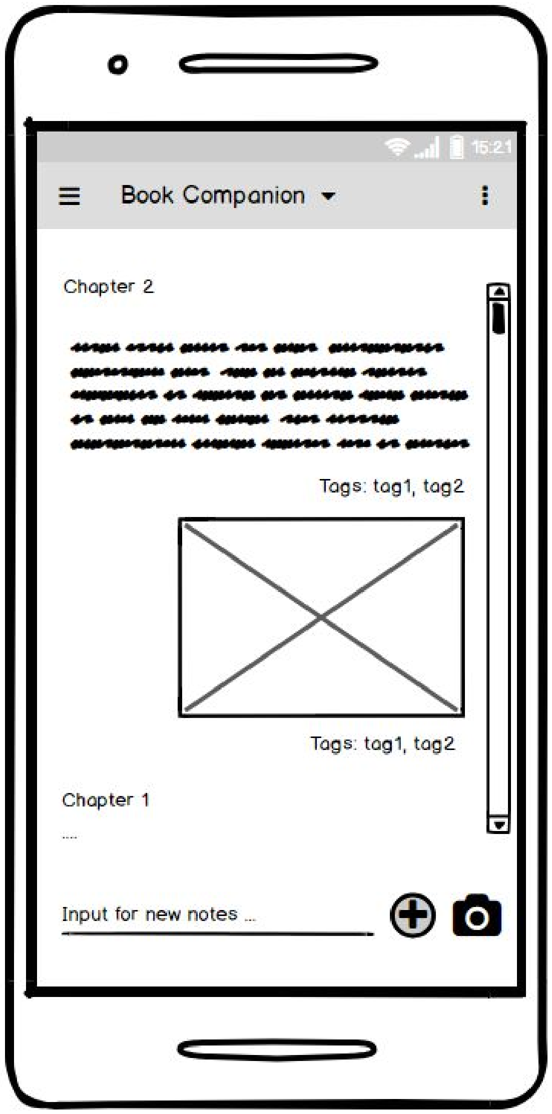
This is the first screen of the app, showing the list of books you have in your database, allowing sorting options (order alphabetically or by date). This screen also has a FAB allowing to add a new book to your collection.

Screen 2



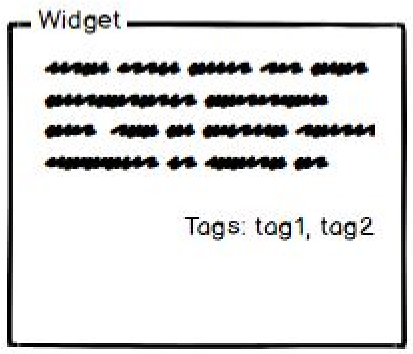
This form allows you to add a new book to your collection.
You can either search for a book online to auto complete the fields easier, or manually input the required data.

Screen 3



This is the screen that allows you to view notes and take new ones (either in text or image format - you can take a picture of a page/section of the book). You can also add tags to the notes to find them quickly later.

Widget



The widget will display a random quote from your notes and will update daily.
On clicking the text it will launch the NoteDetail activity for the current book.

Key Considerations

How will your app handle data persistence?

All the data will be stored locally in a SQL database and accessible through a content provider..

Describe any edge or corner cases in the UX.

I may have to design an additional popup dialog (or new activity) to enter more details about a current note you are taking, such as tags, chapter, etc.

We should also take into account the fact that we will place ads into our app so we need to make room for ads in the UI, or figure out a flow in which we display ads to the users at certain times (making sure we are not annoying)

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

Picasso for the loading of images. Simplifies all the work in loading images from the internet, resizing, showing placeholders in case of errors, etc.

Below is a table with all the dependencies and their versions:

Library	Version
Android support library	27.1.1
Picasso (com.squareup.picasso:picasso)	2.5.2
Google Admob (com.google.android.gms:play-services-ads)	15.0.0
Google Services (com.google.gms:google-services)	3.0.0
Google Analytics (com.google.android.gms:play-services-analytics)	10.2.4
Constraint Layout (com.android.support.constraint:constraint-layout)	1.1.2

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

As external services we will use:

- Google Admob for ads.
- Google Analytics to gather data about application usage
- An external API to get information about a book based on the title or ISBN.
  Google books API seeems a good place to start.
  Example query:
  https://www.googleapis.com/books/v1/volumes?q=1932073485

  We will implement requests to the Google Books API using an AsyncTask and AsyncTaskLoader. Every time the user types something in the search bar when adding a new book we will trigger the api call, show a spinner indicating we a loading data, and then display the results in a grid.

Other Considerations:

- App is written solely in the Java Programming Language
- App keeps all strings in a strings.xml file and enables RTL layout switching on all layouts (maybe follow guidelines here: https://android.jlelse.eu/rtl-support-on-android-here-is-all-you-need-know-e13f2df512e2)
- App will have support for accesibility. We will include content descriptions for all images and UI elements.

Next Steps: Required Tasks

Task 1: Project Setup

1. Start with an Android Studio new project wizard, selecting the master/detail flow template.
2. Add additional libraries. Find out what dependencies we need (like constraint layout, picasso, etc.) and add them to the build.gradle file
3. Setup google services, obtain API keys for them, check the licenses and terms of usage.

Task 2: Defining entities, database, content provider.

1. We should define the java classes and enums that we will use to represent the main objects for the app like: book, note, note types (text or image), tags, photos, etc.

2. We should then proceed to implement the database in which we will store all the information and the content provider through which we will access all the data.

Task 3: Implement UI for Each Activity and Fragment

1. Build UI for MainActivity. Implement RecyclerView and Loader for getting data from ContentProvider
2. The FAB action should open a form in which you can add a new book by searching in the Google Book API or by manually filling the title, author and cover image.
3. Build UI for NotesActivity. Implement the scrolling view for the current notes and the fields that permit adding a new note (either by text or taking a photo). The layout should be similar with the WhatsApp interface for sending and receiving messages.
4. Implement the UI for the widget which will display a random quote from your notes.

Task 4: Adding Google Services

1. Identify the best location and timing for the Google Ads and add the code to serve ads.
2. Add the code to connect to the Google Analytics API and define events we want to log.

Task 5: Preparing for release

- Implement some unit and integration tests to check that everything works as planned.
- Generate signing keys for the app.
- Check the gradle setup and see if we need additional goals, and make sure the packaging and signing configurations are set up.