GitHub Username: kishanrraval

Book Desk

Description

Step into the fictitious and mind wandering world of books. If you find yourself lost, scan the ISBN code of the book or search the book name for a complete description as well as the overview of the book along with the author.

Intended User

This app is meant for use by book lovers and readers ranging from the age of 10-90.

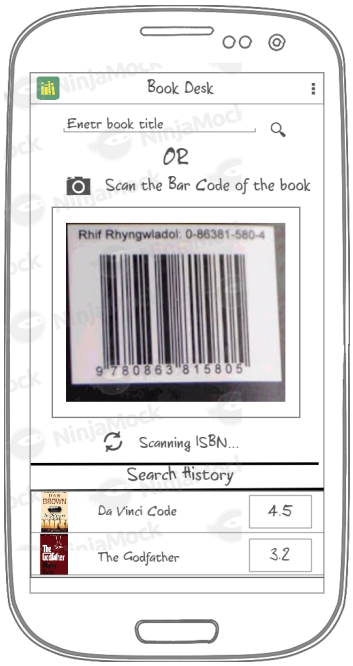
Features

The application provides various features like:

- Descriptions about a book are searched by scanning the book's ISBN code.
- It works on Android phones as well as tablets.
- Not only the book details are available, but also the details about the author.
- The data will be backed up locally as well as on the cloud, which enables the user to login from multiple devices.

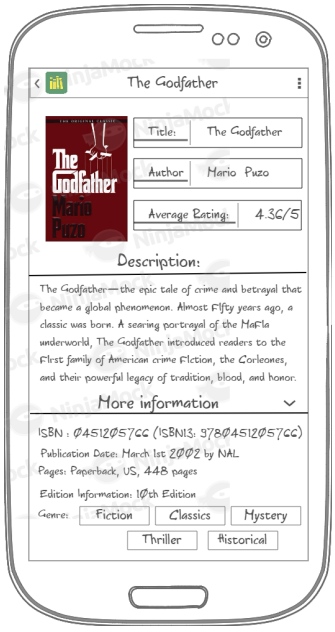
User Interface Mocks

Screen 1



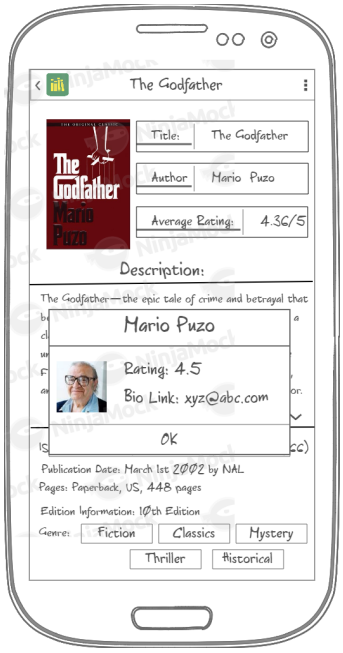
This UI showcases the home page of the application. User can search any book by providing ISBN number through EditText or by scanning the ISBN bar code available on the back of the book. Every time a user searches a book, it will be added to their personal history and will be showcased on the bottom of the home-screen.

Screen 2



This UI will be displayed when the user selects a book either from the history or by scanning the ISBN code. The UI contains detailed description and overview of the book along with book ratings.

Screen 3



A dialog will pop-up when the user clicks on the author. This UI displays the rating of the author and provides a link of his bio.

Screen 4

A widget which will show past searches that user has made. Each item will display image of book, title of book and average rating of the book.

Key Considerations

How will your app handle data persistence?

First, we will use Firebase Authentication for login. The received data would be stored on firebase database, so that if same user logs in from other device, then he can get his previous app experience back. Then I'll store all the search history details to firebase database as well. Other data which is given from GoodReads API will be handled by AsyncTask.

Describe any edge or corner cases in the UX.

We will store searched ISBN number on savedInstanceState, so that even if the phone gets tilted, user can see same description of the book after tilting the phone.

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

- Glide: Showing images of book using URL to ImageView.

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

- Firebase Database: To store data that is needed to be stored on cloud.
- FIrebase Authentication: To authenticate user which can help to use the same app from multiple device concurrently.

Next Steps: Required Tasks

Task 1: Project Setup

- Setting up the project on Android Studio
- Creating a new project on firebase console and doing connection of app to firebase

Task 2: Project Setup

- Adding login functionality to app using firebase authentication
- Creating a UI for authentication
- Get user's identity details on successful authentication
- Show error message on unsuccessful authentication
- Adding sign out button on menu list and implementing the sign out functionality

Task 3: Implementing the first screen which includes Camera integration

- Creating UI according to the mock
- Adding camera to UI itself rather than using native camera app
- Auto detecting BarCode without clicking the capture button

Task 4: Storing and retrieving data from Firebase

- Storing user's authentication details as well as the search history in firebase database
- Retrieving the data of the search history of the user from the Firebase database and displaying in the homepage
- Adding the books (from the search history) dynamically to linear layout

Task 4: Implementing AsyncTask adapter and fetching book details from Goodreads API

- Generating appropriate URL using searched ISBN number and developer key
- Initiating connection and fetching the data though AsyncTask adapter
- Getting needed data from OnPostExecute method
- Handling exceptions and errors and display appropriate message to the user

Task 5: Displaying the fetched data to description UI

- Implementing layout according to UI mock
- Displaying fetched data systematically on the user Interface

Task 6: Implementing tablet mode UI

- Creating a separate layout resource directory and adding a new UI which contains both Screens in a single UI
- Handling the fragments and other UI elements properly.

Task 7: Adding widget to the app

- Designing the widget layout
- Creating a class that extends AppWidgetProvider
- Providing AppWidgetProviderInfo metadata
- Declaring Widget in the Application Manifest