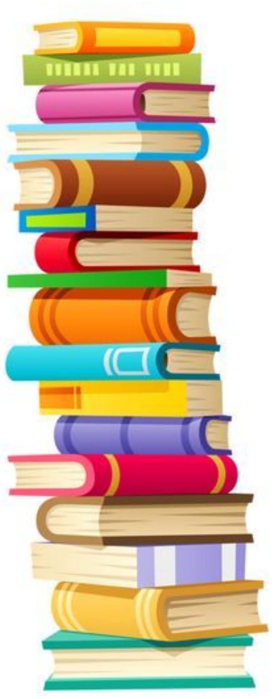


TOWER

Architecture and Design Overview

Youssef Elmougy, Michael Cappeller, Liam Carlton, Raj Bedi

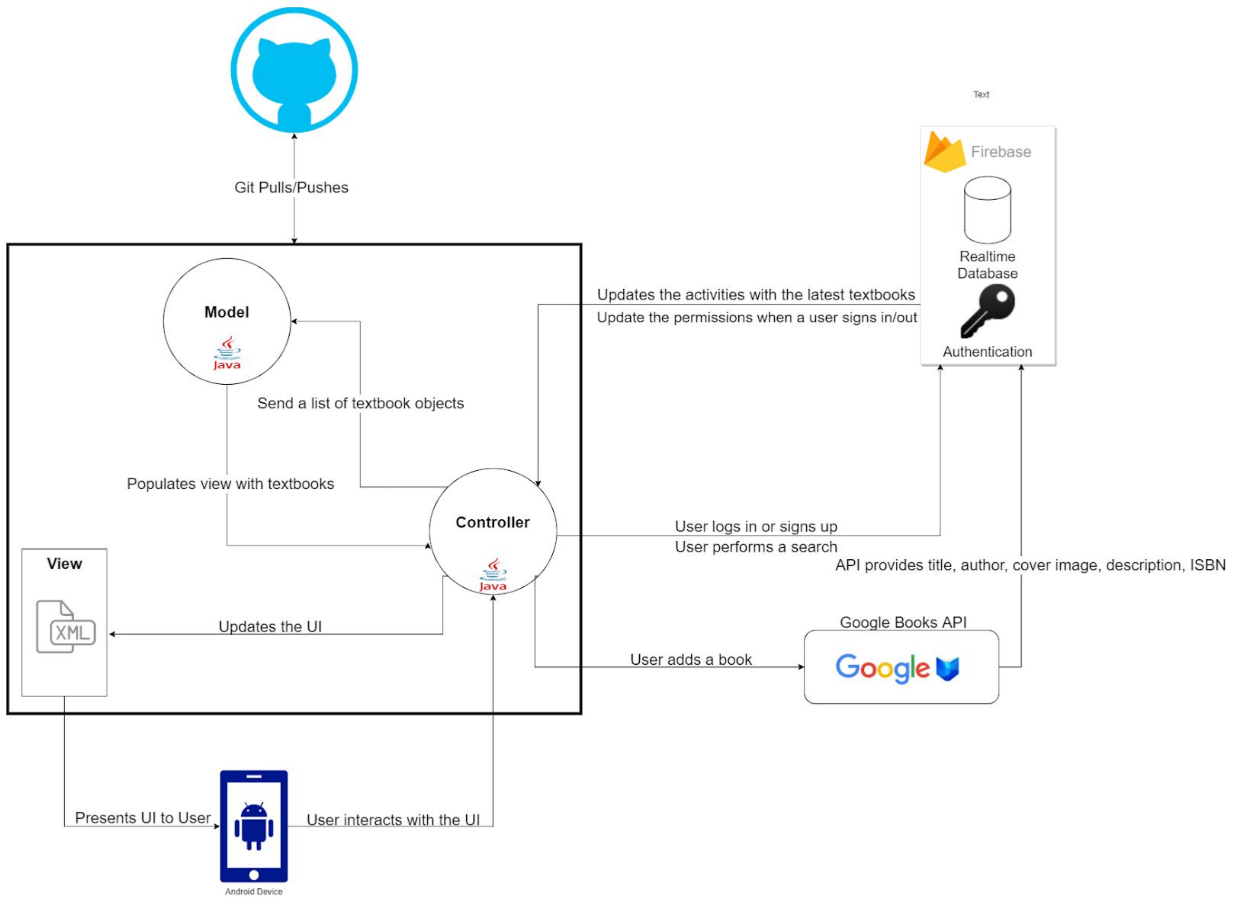
Version 2

Spring 2020

Table Of Contents

1. Introduction

This paper will outline the architectural pattern used for creating the TOWER app. Our application will be designed in accordance with the Model-View-Controller architecture pattern, with the exception being our utilization of two external software: Firebase and the Google Books API. The application will follow a thin-client model where most of the code relies heavily on data from an external source. All backend services such as database management and user authentication will run in the cloud via Firebase. The database will store textbook information as well as account information in the Firebase Realtime Database as a JSON file, and the application will have methods that access this information in order to display information to the user. The application will also utilize a google API to allow users to upload textbooks faster and with access to more information. Below is a box and line diagram that roughly illustrates the architectural pattern we employed.


Text
Firebase
Realtime Database
Authentication
Git Pulls/Pushes
Model
Java
Controller
View
XML
Updates the activities with the latest textbooks
Update the permissions when a user signs in/out
Send a list of textbook objects
Populates view with textbooks
User logs in or signs up
User performs a search
API provides title, author, cover image, description, ISBN
Updates the UI
Google Books API
Google
User adds a book
Presents UI to User
User interacts with the UI
Android Device

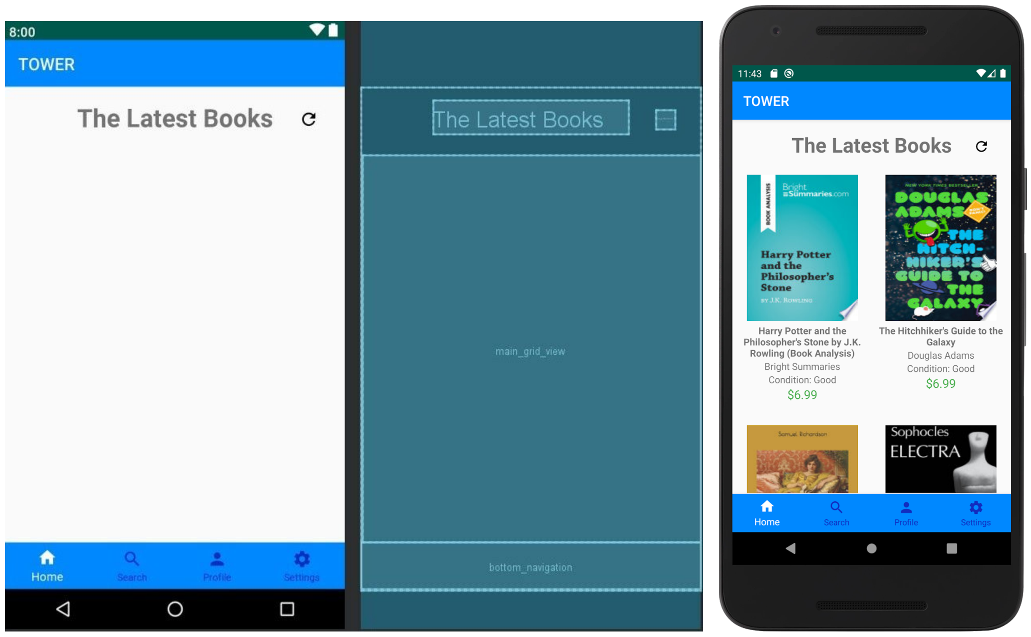
2. Architecture Overview

In this section we will briefly outline our approach to describing the architecture as well as the thought process that led us to take this approach.

Our main goal is to thoroughly establish the MVC architecture pattern as a genuine guiding force in designing our project. To accomplish this, we've dedicated sections 3-7 of this paper to exhaustively describe each of the applications major activities (an "activity" is basically the Android terminology equivalent of a "screen"). In general, each of these activities contains a view component that decides what the user can see and touch, a controller component that decides what happens when the user touches something, and a model component that does the thing the controller tells it to do (which usually involves updating the view component with the new changes). It should be noted that in some events, the controller is able to simply update the view directly without needing to invoke the model. Since we use fairly consistent notations, it will be obvious when such a situation comes up.

3. Home

3.1. View Component



Upon opening the app, the user is greeted to the home page, which will display a scrollable list of textbooks as well as a navigation menu. On this screen, the user has access to the following options:

3.1.1. Tap on any of the books to reveal additional info.

3.1.2. Tap the refresh button near the top-right corner of the screen.

3.1.3. Tap any of the four icons on the navigation menu near the bottom.

3.2. Controller Component

Whenever the user takes any of the actions mentioned above, the controller will be fed the information and a corresponding event will fire.

3.2.1. Upon tapping one of the books, one of two things will happen. If the user is not logged in, the controller will directly update the view to display the "Login/Signup" page. If they are already logged in, the controller will directly update the view to display the "Specific Textbook" page corresponding to that book.

3.2.2. Upon tapping the refresh button, the controller calls upon the model to scan Firebase for any changes made to the textbook database since the last time the page was refreshed (an initial refresh occurs when the app is launched)

3.2.3. Upon tapping any of the four icons in the navigation menu, the controller will update the view to display a page corresponding to whichever icon was tapped. The four pages that can be reached this way are Home, Search, Profile, and Help. Note, this navigation menu is included in every activity for the users convenience. However, since it always does the same thing, it will not be discussed in subsequent sections.

3.3. Model Component

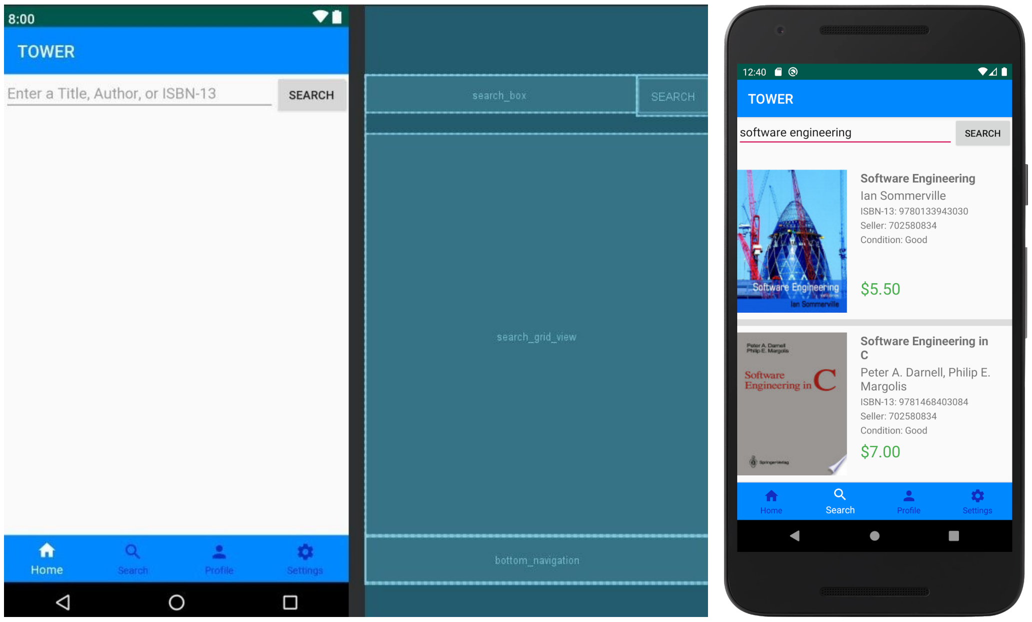
3.3.1. The model does not need to be invoked in this case, as the controller simply updates the view directly.

3.3.2. The model is invoked by iterating through the Firebase Database and updating the current grid to display any changes such as if a book has been added or removed. The application does this initially at launch and then keeps the grid static until the user hits the refresh button. When the refresh button is hit, the application clears the grid and then remakes it reflecting the new changes.

3.3.3. The model does not need to be invoked in this case, as the controller simply updates the view directly.

4. Search

4.1. View Component



Software Engineering
Ian Sommerville
ISBN-13: 9780133943030
Seller: 702580834
Condition: Good
$5.50

Software Engineering in C
Peter A. Darnell, Philip E. Margolis
ISBN-13: 9781468403084
Seller: 702580834
Condition: Good
$7.00

The search page will contain an empty search box and button upon arrival. Once a user makes a search then the view will update to show results based on the user's search query. On this screen the user has access to the following options:

4.1.1. Type in the title, author, or ISBN of the textbook they are searching for into a text field.

4.1.2. Tap the search button.

4.2. Controller Component

4.2.1. The controller will monitor the text field to determine whether or not it is empty.

4.2.2. Upon pressing the search button, so long as the text field isn't empty, the controller will invoke the model to deploy the search algorithm on the textbook database.

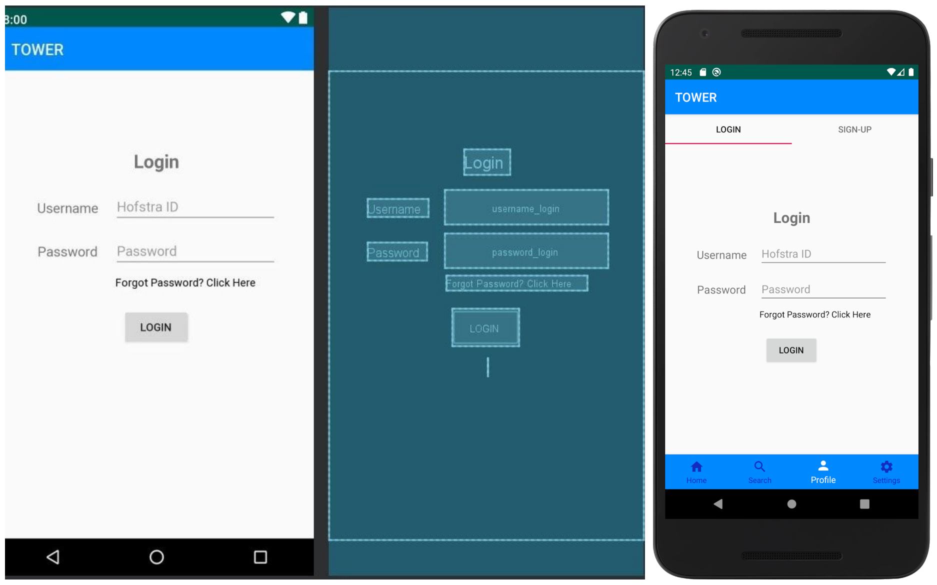
4.2.3. The search algorithm will ultimately generate a scrollable list of textbooks that the user may tap on. Doing so will cause the controller to update the view to display the "Specific Textbook" page corresponding to that book.

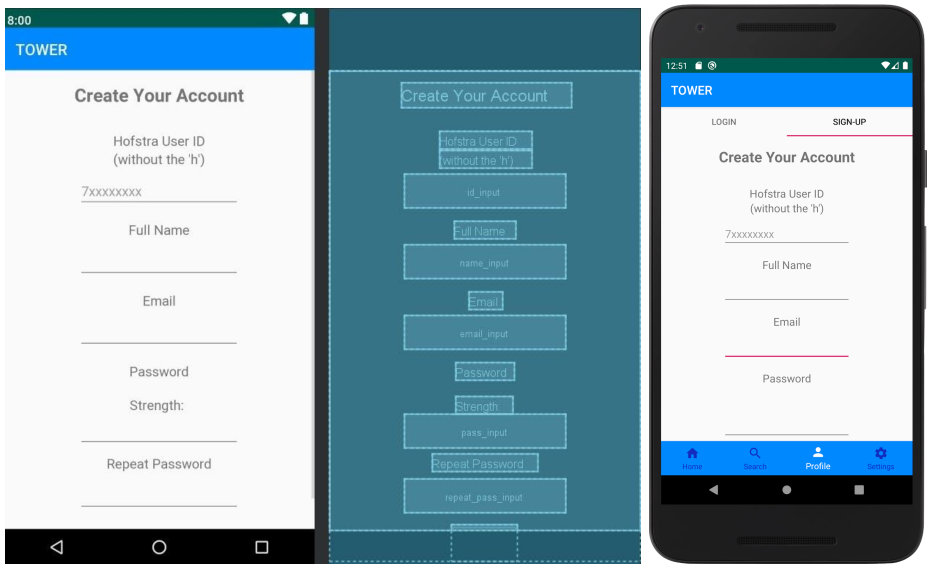
4.3. Model Component

4.3.1. To initiate the search algorithm, the model first makes a call to Firebase. Firebase then responds by returning a JSON file containing relevant data for all the items in our textbook database. The model then uses this JSON data to generate a corresponding ArrayList of textbook objects. Next, the model sorts this list of books based on how "similar" they are to the search query supplied by the user. This is done by first assigning each textbook an integer equal to the length of the longest substring shared between it and the search query. Then, the ArrayList is reordered from greatest to least in terms of the new integer value pinned to each book. Afterwards, the model feeds this new ArrayList into an adaptor class called "Textbook Adapter". The Textbook Adapter then uses the first 40 textbooks in this list (we prevent potential slowdowns by eliminating less relevant results) to generate the schematics for what's called a gridView. Finally, these schematics are passed back to the view, causing the screen to display the book cover, author, ISBN, seller, condition and price of each textbook in an aesthetically pleasing fashion.

5. Log in/Sign up

5.1. View Component





The Profile Page will be where the user has the ability to login or create an account. The user will be able to navigate between these two options through clicking one of the two tabs on the top of the screen. Depending on the tab selected, the user will be able to do the following:

5.1.1. When signing up, type their username, full name, email, password, and re-typed password into the corresponding fields and tap the sign up button.

5.1.2. When logging in, type their username (aka Hofstra ID) and password into the corresponding text fields and tap the login button.

5.2. Controller Component

5.2.1. Sign-Up

5.2.1.1. When the sign-up button is pressed, the controller calls upon the model to determine whether or not the information supplied by the user is valid. Additionally, the model will be invoked to evaluate the strength of the supplied password in real time.

5.2.1.2. If the users input is determined invalid, the controller updates the view to display a message detailing what went wrong. (Fields left empty, password being different than the re-typed password, etc…)

5.2.1.3. If the input is determined valid, the user will automatically be logged in and the controller will prompt the view to display the add book page

5.2.2. Login

5.2.2.1. When the login button is pressed, as long as neither of the fields are empty, the controller calls upon the model to determine whether or not the credentials supplied by the user match any of those in our database.

5.2.2.2. Additionally, there is the option to reset your password via the email address attached to the account. This utilizes Firebase Authentication which sends an email to the accounts email with instructions to reset the password.

5.3. Model Component

5.3.1. Signup

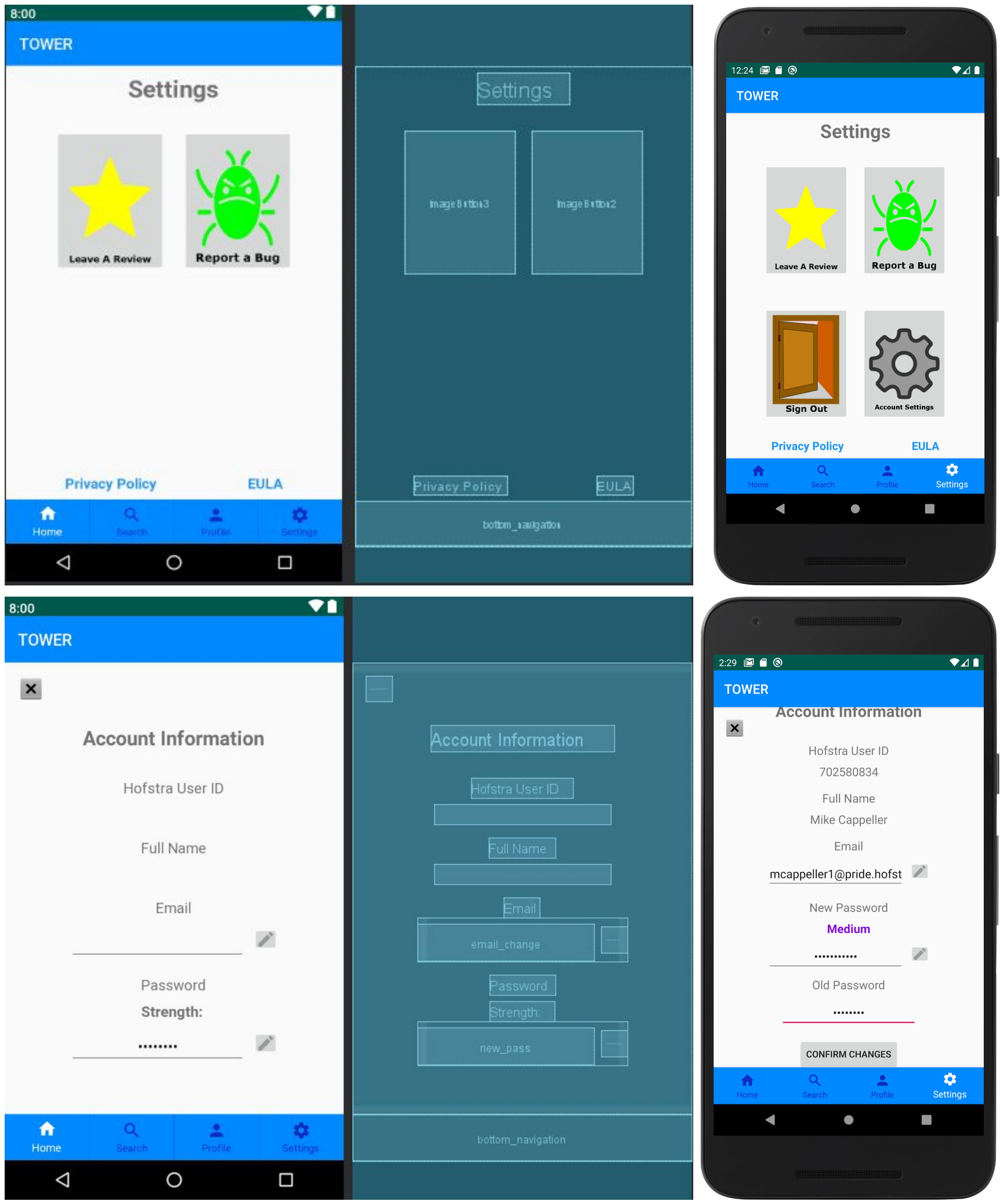
5.3.1.1. Firebase Authentication and Firebase Database take the information sent from the controller as a final step before officially creating the account. Firebase checks to see if there is already a user with that Hofstra ID or email address and throws an exception if that occurs. If an account is created successfully then the user is added to Firebase where their password is encrypted and hidden from even the administrators of the application. The user is then signed in and the view is updated.

5.3.2. Login

5.3.2.1. Since Firebase only allows users to sign in using their email address and our app requires that users login with their Hofstra ID, it is necessary to map each email address with a Hofstra ID in the database. After making this translation, the user can now successfully login using their Hofstra ID and password. If the information is valid the user interface is updated to reflect the updated permissions a user has.

6. Settings

6.1. View Component



Hofstra User ID
702580834
Full Name
Mike Cappeller
Email
mcappeller1@pride.hofst
New Password
Medium
Old Password

The settings is where the user can see all of their account information as well as information about the application. In this activity the user will have the ability to:

6.1.1. Tap leave a review.

6.1.2. Tap report a bug.

6.1.3. Tap Privacy Policy

6.1.4. Tap EULA

6.1.5. If a user is signed in:

6.1.5.1. Tap Sign Out

6.1.5.2. Tap Account Settings

6.1.6. In Account Settings:

6.1.6.1. A user can view the name and email attached to their account.

6.1.6.2. Tap a button to edit their email.

6.1.6.3. Tap a button to edit their password.

6.1.6.4. Tap a button to confirm any changes.

6.2. Controller Component

6.2.1. Tapping leave a review will tell the controller to open the Google Play Store where they can leave feedback of what they think of the application.

6.2.2. Tapping report a bug will tell the controller to open an Android dialog box where a user can then email the Tower Support Team to report any concerns or issues they are facing with the application/

6.2.3. Tapping privacy policy tells the controller to start a new activity called “PrivacyPolicy” where there is a PDF stored locally of the application’s official privacy policy. The controller updates the view directly.

6.2.4. Tapping EULA tells the controller to start a new activity called “EULA” where there is a PDF stored locally of the application’s official end-user license agreement. The controller updates the view directly.

6.2.5. If a user is signed in:

6.2.5.1. Tapping sign out tells the controller to call upon the model to notify is that the user is no longer logged in. The controller then updates the view to reflect these changes.

6.2.5.2. Tapping account settings tells the controller to start an activity called “AccountSettings” and this will update the view directly.

6.2.6. In account settings:

6.2.6.1. The controller calls upon the model to scan Firebase for the information about the user that is currently logged in.

6.2.6.2. The controller updates the view directly and allows for a user to enter in a different email.

6.2.6.3. The controller updates the view directly and allows for a user to enter in a different password.

6.2.6.4. The controller calls upon the model to update the Firebase Database with the new information.

6.3. Model Component

6.3.1. The model does not need to be invoked in this case, as the controller simply updates the view directly.

6.3.2. The model does not need to be invoked in this case, as the controller simply updates the view directly.

6.3.3. The model does not need to be invoked in this case, as the controller simply updates the view directly.

6.3.4. The model does not need to be invoked in this case, as the controller simply updates the view directly.

6.3.5. If a user is signed in:

6.3.5.1. The model notifies Firebase that the user is no longer signed in. The model then updates the view to reflect that the user is no longer signed into their account.

6.3.5.2. The model does not need to be invoked in this case, as the controller simply updates the view directly.

6.3.6. In account settings:

6.3.6.1. The model connects to the Firebase Database and retrieves the email and Hofstra ID of the user currently logged in.

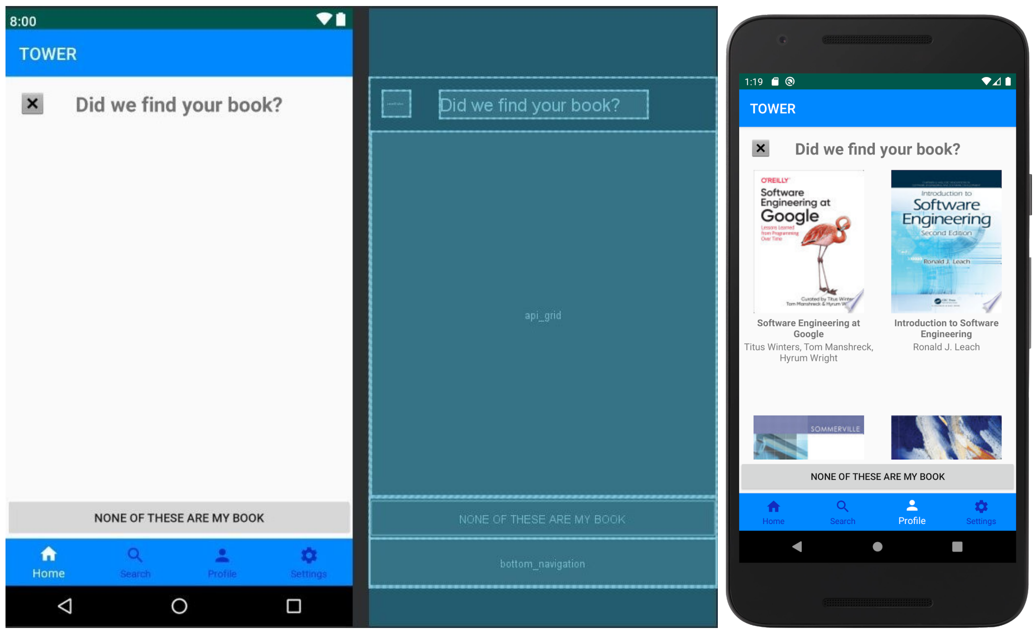
6.3.6.2. The model does not need to be invoked in this case, as the controller simply updates the view directly.

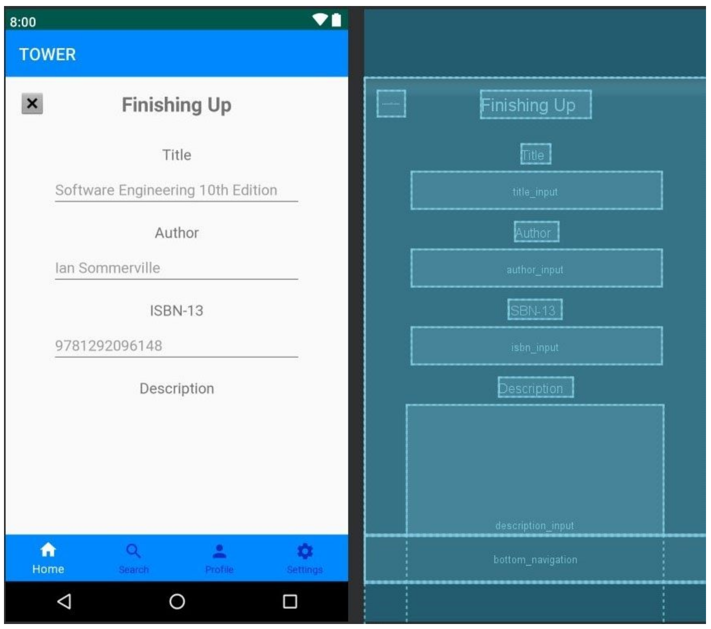
6.3.6.3. The model does not need to be invoked in this case, as the controller simply updates the view directly.

6.3.6.4. The model calls upon the Firebase to verify that this information is valid. If the information is valid, then the Firebase Database and Firebase Authentication will update their information to reflect the changes to the email and/or password.

7. Add Book

7.1. View Component


8:00
TOWER
Did we find your book?
NONE OF THESE ARE MY BOOK
Home
Search
Profile
Settings
Did we find your book?
api_grid
NONE OF THESE ARE MY BOOK
bottom_navigation
1:19
TOWER
Did we find your book?
Software Engineering at Google
Titus Winters, Tom Manshreck, Hyrum Wright
Introduction to Software Engineering
Ronald J. Leach
NONE OF THESE ARE MY BOOK
Home
Search
Profile
Settings



Finishing Up

Title

Software Engineering 10th Edition

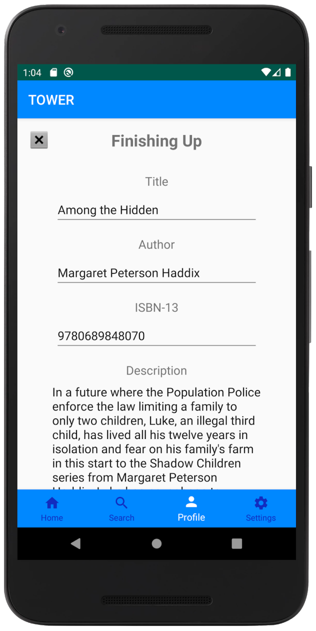
Author

Ian Sommerville

ISBN-13

9781292096148

Description



Finishing Up

Title

Among the Hidden

Author

Margaret Peterson Haddix

ISBN-13

9780689848070

Description

In a future where the Population Police enforce the law limiting a family to only two children, Luke, an illegal third child, has lived all his twelve years in isolation and fear on his family's farm in this start to the Shadow Children series from Margaret Peterson

The sell page will consist of two parts. The first being where the user checks to see if their book exists on the Google Books Database. The second being where the user enters or changes information about the book they are uploading. On these two activities the user will have the ability to:

7.1.1. Look at a display of books that are relevant to their search query.

7.1.2. Tap on any book to receive more information including a description and ISBN.

7.1.3. Tap on a button labeled "None of these are my Books"

7.1.4. Enter information about the book in addition to setting the price and condition.

7.1.5. Tap a button to begin listing the book on the application.

7.2. Controller Component

7.2.1. After a user has made their search, the controller will call upon the model to scan the Google Books API for books matching their search query.

7.2.2. Upon tapping one of the books, the controller will directly update the view to display the "Specific Textbook" page.

7.2.3. Upon tapping the button labeled "None of these are my Books", the controller will update the view to display the "Add Book Final" page.

7.2.4. The controller will check the strength of a user’s password in realtime and call upon the model to show if a user has entered a password that is weak or strong. The controller will also check to see if a user has entered any empty input in which the controller will directly update the view by preventing the user to press the “Add Book” button.

7.2.5. The controller will check if the user has entered any invalid input such as a title with 1000 characters and throw an exception if necessary. If an exception is found, the controller directly notifies the view that the input is invalid. The controller will then call upon the model in order to add the book to the database.

7.3. Model Component

7.3.1. The Volley Library is used to make an HTTP GET request to the Google Books API. This returns a large JSON object which contains the information of 40 books related to what the user entered. Volley is then used again to filter through the JSON object to find the title, author, ISBN-13, description and image url.

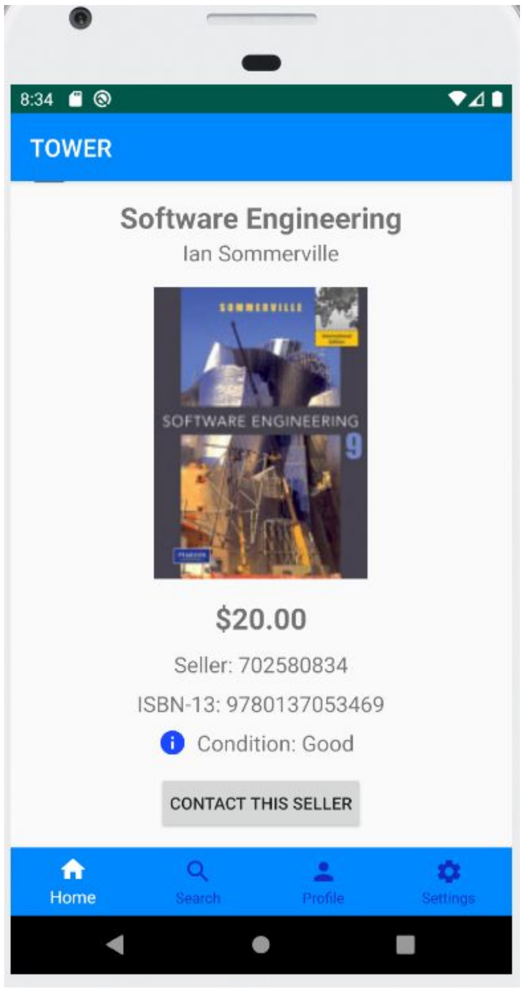
7.3.2. The model does not need to be invoked in this case , as the controller simply updates the view directly.

7.3.3. The model does not need to be invoked in this case , as the controller simply updates the view directly.

7.3.4. The model is invoked by calling upon the PasswordStrength class which takes the password a user enters and awards it a score based on how many special characters, numbers, and uppercase letters the password contains. If a password has a higher score than it is more secure and the model updates the view to show the user the strength of their password as they type it in.

7.3.5. The model is invoked by connecting to the Firebase Database and preparing it to add a new JSON object. First a unique key is generated for the textbook. Then the values of that unique key will be the values that the user provided which are title, author, ISBN, description, price and condition. If the book was a copy of a book from the Google Books API, then an image url will be provided. Otherwise a default book cover is used.

8. Specific Textbook



Software Engineering
Ian Sommerville

$20.00

Seller: 702580834

ISBN-13: 9780137053469

Condition: Good

8.1. View Component

8.1.1. This activity displays information about a specific textbook. Here, the user is able to tap the “contact this seller” button.

8.2. Controller Component

8.2.1. When the “contact this seller” button is pressed, the controller invokes the model to obtain the emails of both the buyer and the seller. The controller then exchanges the email addresses of both users, allowing them to carry out their transaction.

8.3. Model Component

8.3.1. The model sends a call to Firebase in order to retrieve the email addresses of the buyer and the seller.