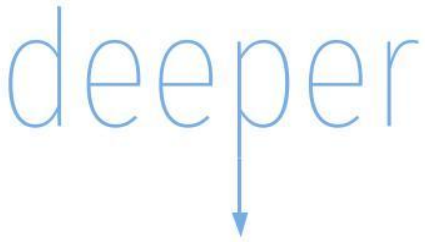
Boston University
Electrical & Computer Engineering
EC463 Capstone Senior Design Project

Final Project Testing Report

deeper



by

Team #6
deeper

Team Members

Nicole Kwon kwonn@bu.edu
Daniel Li dli0793@bu.edu
Harry Li harryli@bu.edu
Byron Mitchell byronmit@bu.edu

Required Materials:

Software:

- Back End:
    - Firebase Authentication
    - Firestore (Database)
- Smartphone Applications on iPhone and Android device (using Expo)
    - UI Interfacing

Pre-testing Setup Procedure:

deeper App:

1. Open project in Visual Studio Code
2. Run “yarn install” in terminal under the root directory
3. Run “expo start” in terminal under the folder “frontend” of the deeper app
4. Install Expo app on iPhone/Android device

Sign In and Sign Up Functionality:

1. Open up Firebase authentication

Community Functionality:

1. Open up the collection “posts” in the Firestore database

Journaling Functionality:

1. Open up the collection “journal” in the Firestore database

Testing Procedure:

deeper App:

1. Run the app by scanning the Expo barcode with an iPhone and/or Android device

Sign In Functionality:

1. Go to the sign in page
2. Put in your created credentials (i.e. email and password)
3. Submit and be prompted to the onboarding pages

Sign Up Functionality:

1. Go to the sign up page
2. Put in your unique credentials (i.e. first and last name, email, password)
3. Submit and be prompted to the onboarding pages
4. This new user should now be shown in the Firestore authentication

Community Functionality:

1. Go to the community tab in the app and click on the plus button
2. Upload a picture and create a post
3. The post should now show up on Firestore
4. Reload the application and the new post with the previous post feed will be on the community feed

Journaling Functionality:

1. Go to the journaling tab in the app and click on the plus button
2. Write in the text field and click on the plus button
3. The journal entry should now show up on Firestore
4. Click on the top right and the new post should be in the past entries

Setup:

There are five different tests taken to test the functional capability of our application–the deeper App, the sign in functionality, the sign out functionality, the community functionality, and the journaling functionality. The deeper App is cross-platform and will be presented through Expo and on an iPhone simulator. The sign in and sign out functionality are demonstrated through an already existing user signing in first, followed by signing up with a new user. There is also error checking implemented, and we will show them during the sign up process. In addition, the community functionality allows users to post to the community about anything. We will create a text input and photo upload, which will then be stored in Firestore; the database will display the feed in the community page. Finally, the journaling functionality will be shown through the user writing a passage, followed by the APIs of saving it and retrieving it through navigating to the past entries tab.

Measurements Taken:

deeper App:

1. After opening Visual Studio, we will run Expo, and the apps are able to run successfully and look the same on both an iPhone and an Android device, demonstrating our cross-platform functionality.
2. The user will be able to navigate through the different screens, from start to finish.
3. The user will be able to navigate through different pages by clicking on the tabs and buttons, represented by an icon or TouchableOpacity.
4. The overall UI will reflect the Figma design and more, which is shown on the side. Most of the different components from the Figma visualization (login page was removed due to current changes in the database being used) are included in our frontend code.

Login Functionality:

1. Explain the metrics and results behind logging in and Firebase

- To measure the functionality behind our authentication that uses Firebase, we look at the error checking in the sign-up/login page and that the correct information is stored/accessed in our database.

a. User should be able to create an account;
    - To demonstrate the error checking in our sign-up page, we left some blank sections in our sign-up form, which then prompted the user to completely fill out the form in order to register. Also, we wrote down different passwords for the password and confirm password sections, which then prompted the user to check the password fields as they don't match in order to register. After correctly filling out the form, we could see that the user's information was safely and correctly stored in the database by accessing Firestore.

b. User should be able to see if their login credentials are correct or not;
    - To demonstrate the error checking in our login page, we input an email address that isn't associated with any user in our database which prompts the user to use an existing account or to sign up with the provided email address. Also, by providing a correct email address, but the incorrect password prompts the user to enter the correct password associated with the account. After correctly login in, we see that the app correctly fetches the user's information as the app outputs it to the console where we can verify that it matches our database.

Community and Journaling Functionality:

1. We will measure if the posting functionality works by seeing if the post saves in the database.
2. Then we will see if the post and previous posts display in the community feed.

<u>Score Sheet</u>

deeper App:

Description	**Did it work? (y/n)**
The Expo apps should be able to run successfully.	Y
The user should be able to navigate through the different screens.	Y
The user should be able to navigate through different tabs by clicking on the buttons.	Y
The UI should reflect the general Figma design and more.	Y

Login Functionality:

Description	Did it work? (y/n)
The user can sign up for a new account.	Y
The user can log in to an existing account.	Y
The user will be denied if the password and email combo are wrong.	Y
The user cannot register a new account if the email address is already registered.	Y

Community Functionality:

Description	Did it work? (y/n)
The user can create a post.	Y
The post will save in the Firestore database.	Y
The feed will show all posts from the database.	Y
Adding a new post will display the new post in the feed.	Y

Journal Read and Save:

Description	Did it work? (y/n)
The user can create a journal and save it into a database.	Y
The journal can be retrieved.	Y
Return error message when no login cookie was detected.	Y