IERG4999 SJ01 Final Year Project II

Interim Report

Cross-platform Application Development

Group: S1

1155144676 Yeung Tang

Date: 12/3/2023

A FINAL YEAR PROJECT REPORT
SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF BACHELOR OF INFORMATION ENGINEERING
DEPARTMENT OF INFORMATION ENGINEERING
THE CHINESE UNIVERSITY OF HONG KONG

1. Abstract

Passing reading materials such as UGFN/H books from senior to junior years is a tradition in CUHK, but due to COVID-19, it is harder to meet new friends[1]. As a result, most students, after finishing the course, simply throw away the textbooks as they have no one to pass them on to. This results in a waste of paper and money since those books are quite thick which consume a lot of paper to print and it cost around $100 HKD per book.

To address the waste and encourage students at CUHK to meet each other, we have decided to develop an application that allows people to trade things they no longer need for something useful while promising social interactions and a chance to make new friends. Our application also promotes a green campus[2].

2. Background

During pandemic, students are hard to make friends and have a normal social life. They have to wear masks and restricted from face-to-face contacts with a brunch of friends, make them hard to meet new people.[1] In a study among university students in the United States at the beginning of the pandemic, 86% reported feeling socially isolated.[2] Therefore, students switched their focus to social media applications to get to know new people. Additionally, 70.8% of the students stated that the time they spent on social media increased during the COVID-19 pandemic [3].

3. Introduction

We have decided to develop a cross-platform application that encourages students to exchange their ‘waste’. This exchange platform is primarily based on books, so we will include a book scanning function and an instant chat and trade list, etc. We will then test the app and system to ensure that the application works as expected and well preformed.

4. Methodologies

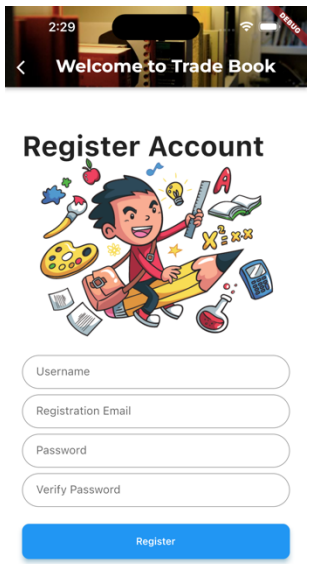
This project contains three main components. The first part is the UI/UX, which is the front-end for users to interact with. The next part is the Node.js part, which provides APIs that help us create, read, update, and delete data for the final part, which is the MongoDB database.

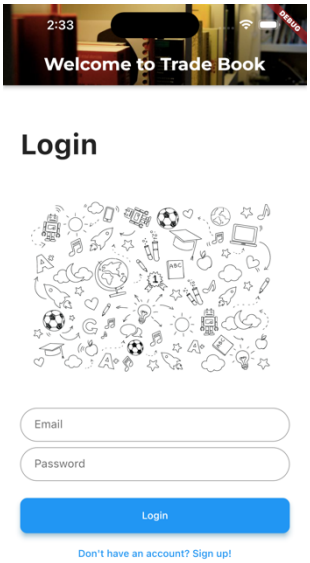
4.1 UI/UX for user to interact with

The client-side application is mainly used by the end-user. End-user may create their account and view their account details. Other than managing their account, the app serves as the medium to communicate with the server, e.g., user can submit photos and upload the books, or real-time messaging. We planned to implement the application with Visual Studio with Dart for the application logic and some online tools to design the user interface.
Below shows some proposed features/screens in our client-side application:

Register Page

This page is for new users to register their account. They simply need to input their username, email, and password, and an account will be registered.





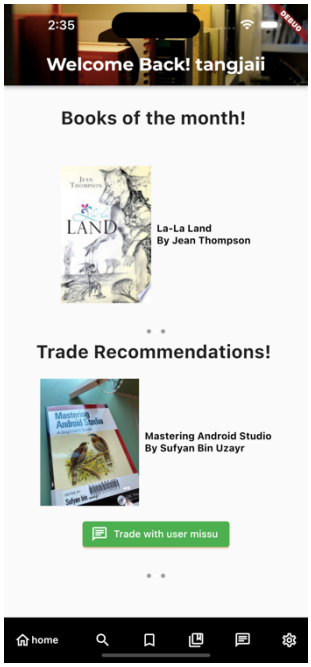
Login Page

This page is for users to log in to their account. A "remember me" checkbox for saving user credentials will be implemented later for fast access. For now, users will need to enter their email address and password to log in. After inputting the email and password, the system will check to see if the user has the correct credentials. If yes, it will redirect the user to the home page. This page also offers a button for new users to register a new account.

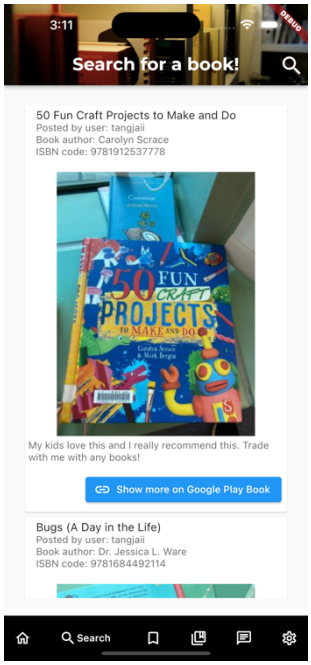
Home Page

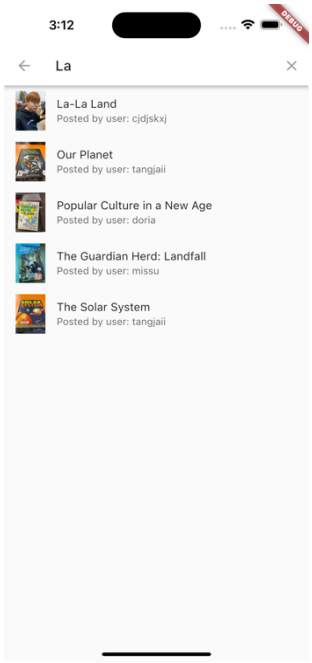
The home page displays books that we recommend to users, as well as books that are currently popular. Through this page, users can easily find books that interest them. For both rows, a carousel slider is used to display a list of books.

When the user clicks on a book, it will prompt up a browser and shows the detailed information about the book on Google Book Store. If the user clicks on the "Trade with user xxx" button, it will then open the ‘tradebucket’, which is the trade interface for the user to select books to trade with another user.



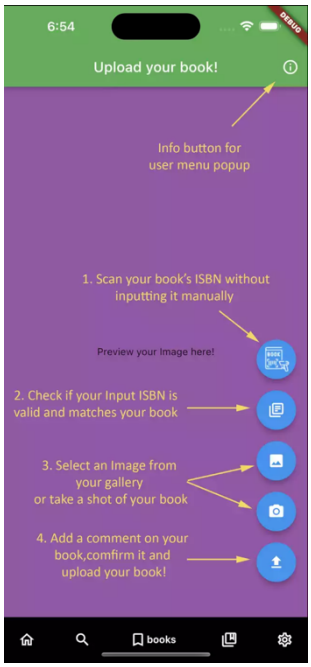
Search Page





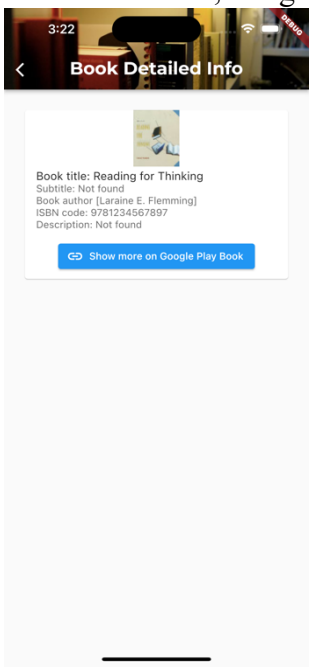
The search page is where users can search for books in our database. If any user has a book that is available for trade, they can search for the book here by clicking the search icon in the top right corner. Upon clicking the search icon, a page will slide up for searching the book name.

Users can also scroll through the search page without inputting any keywords to browse all of the available trade books.



Upload Page

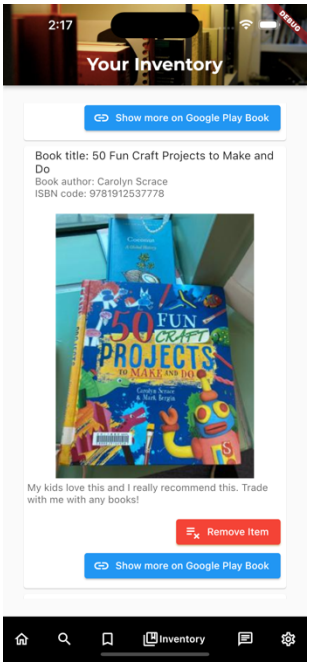
This page is mainly for users to upload their books to our database so that we can record them and make the books available for trade. It consists of 5 buttons: Scan ISBN, Scanned Book Details, Image Picker, Camera and Upload. If the user completes the process and presses the "Upload Book" button, the book with the image, ISBN, and user credentials will be uploaded to our database. Details of each button and APIs related will be shown during the final report.

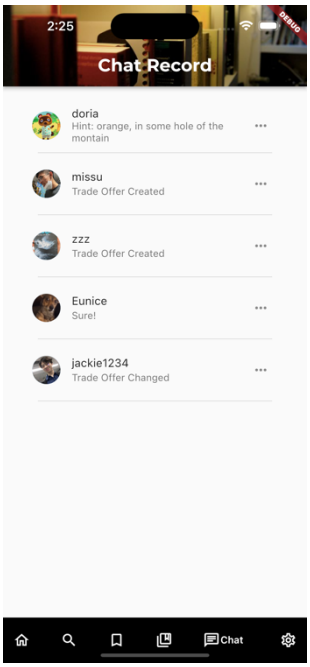


Scanned book detail example:

Inventory Page

This page is for users to browse and modify their own inventory. From here, users can edit the comments/tags on their book, view more details about their uploaded book on Google Book Store, or remove their uploaded book from their trade inventory.

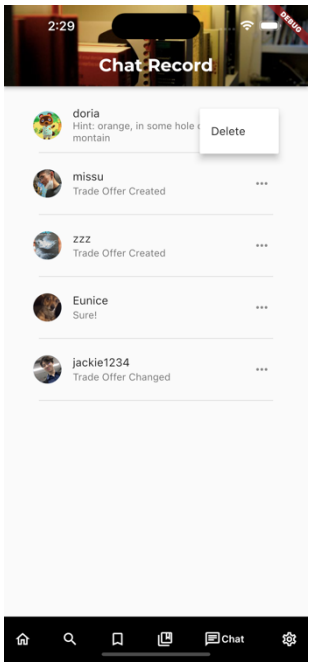


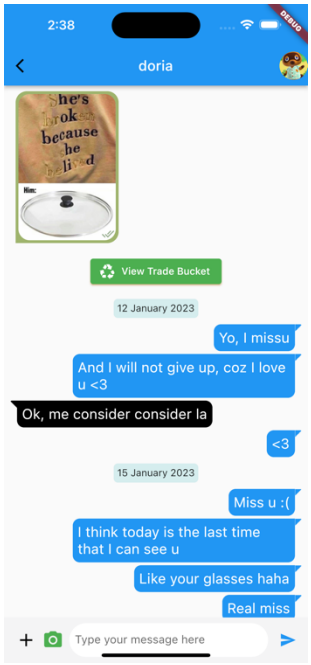


Chat Record Page

This page lists the chat contacts of other users if a chatter record was found for the user. The user can also delete the chatter record by pressing the three dots at the end of each chat contact and selecting the "Delete" button.

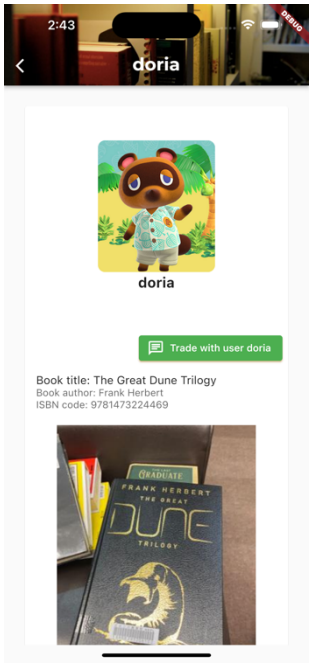
How to delete a chat:



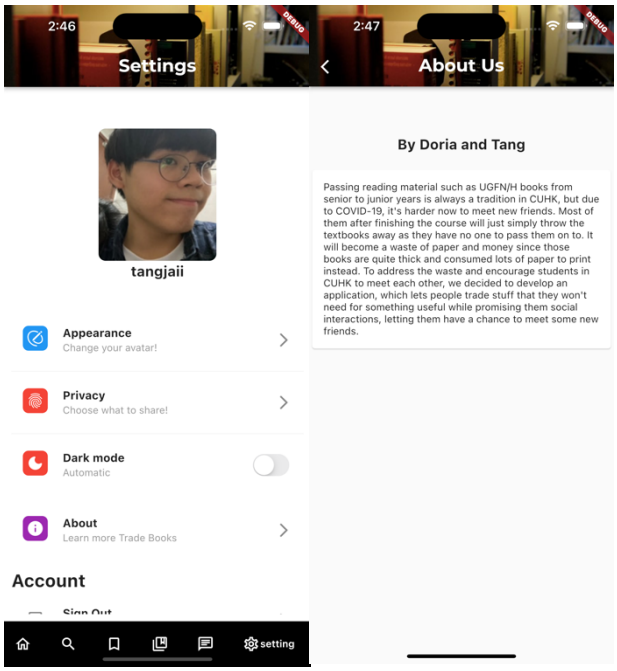


Chatter Page

This page shows the conversation between users. Users can send text messages as well as emojis. They can also send images from their photo gallery or take a picture from their camera to better describe something and communicate further. Users can also click on the user icon to learn more about the user they are chatting with, such as what other books they offer.



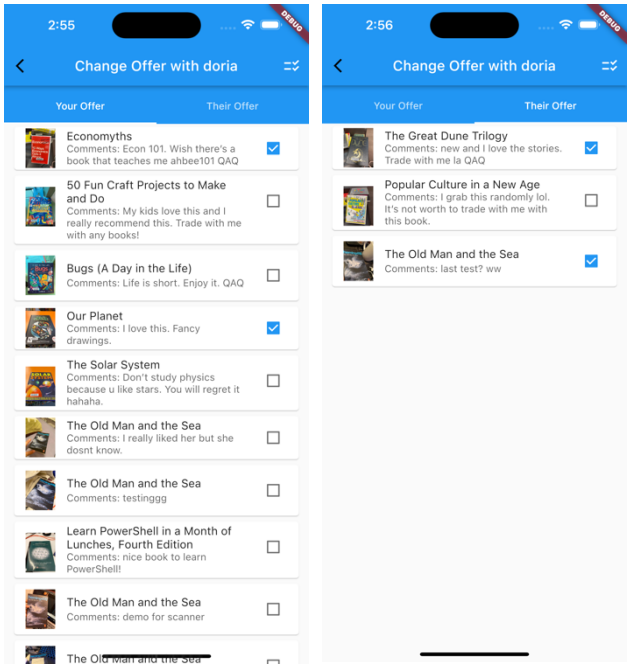
User doria for example after pressing her icon:



Setting Page

This page is for users to make modifications to their use of the application. Users can change their avatar, settings for what they want to share, enable dark theme, and know more about the project/developer through the "About Us" button. Users can also change their username and password and log off from their account on this page.

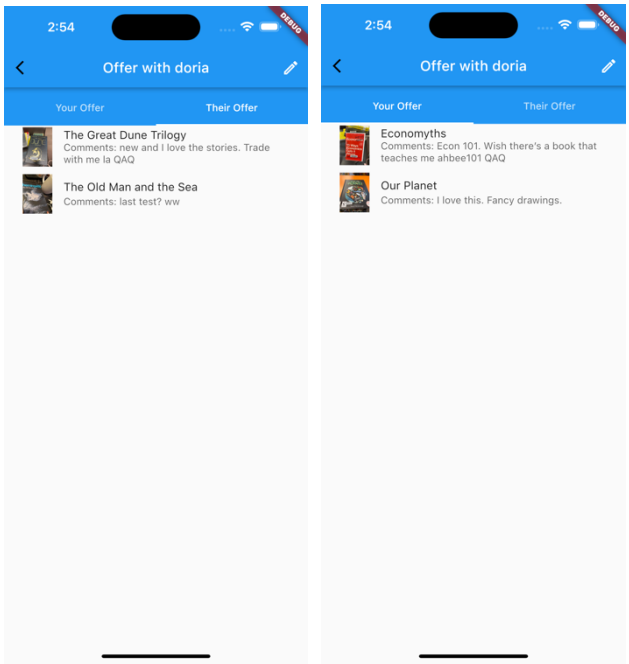
Trade interface (Implementing)



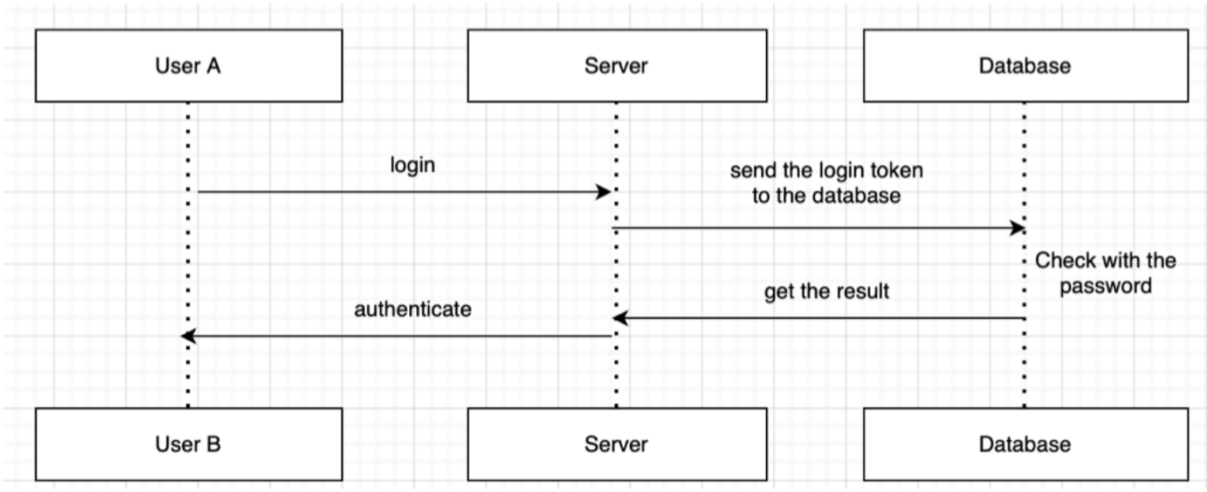
Users can send a trade request to other users containing the books they want to give and receive from the other user. When the user wants to initiate a request, this interface will appear. For this page, the user will first need to select one or more items from their inventory for trading. Afterwards, they can select what they want from the other user's inventory/trade list. After confirming, the trade request will be sent to the other user.

This page will also be reused when one party wants to change the trade offer, making a counter-offer.

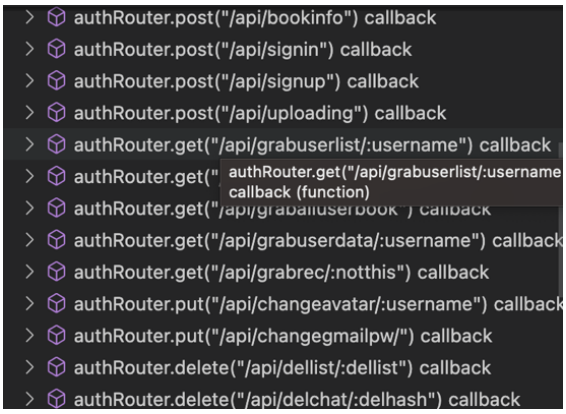
Example of an offer :



4.2 NodeJS as middleware



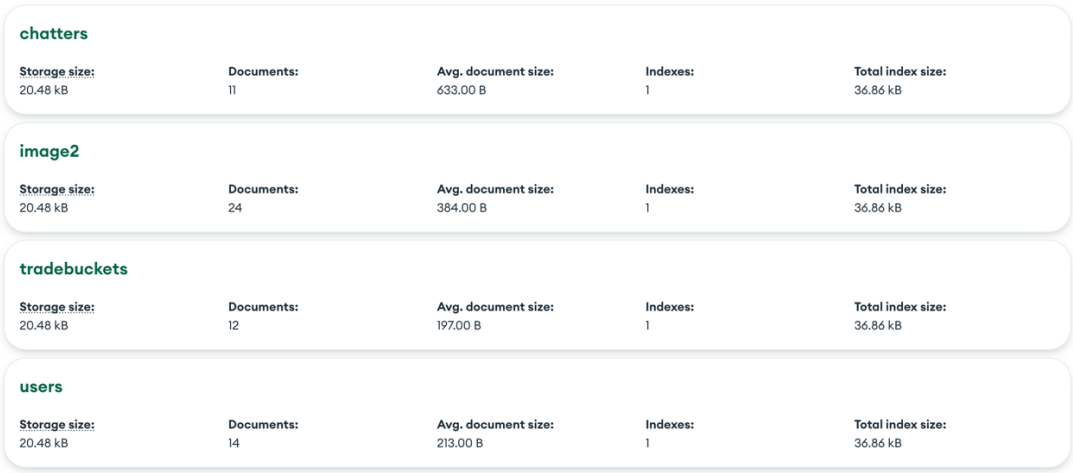
The graph above is a simple way to demonstrate how the data flows between these three main components for user registration and login to the app. More detailed and well-explained breakdown of each APIs will be shown in the final report for this project.




```
646   });
647   module.exports = authRouter; //allow public access
```

There are more than 20 APIs for this app to CRUD the database for different purpose

4.3 MongoDB as Database



We use MongoDB as our database since MongoDB is a non-relational database[4] which provides us the flexibility to implement the data flow we wanted. The image above shows the data we are storing in MongoDB, which can be divided into four main parts. The first part is users, which stores the user credentials after they register an account and verifies the user's credentials upon login. It also stores the user's avatar and username.

See example below:

```
_id: ObjectId('63ad9c9e1aea9704abae64bf')
name: "doria"
email: "doria@gmail.com"
password: "$2a$08$rUMswEdhf6Xch30jz63XnOnJ.rWsI7aNsPlheU6q6XlySzt.qMCQC"
address: "https://i.imgur.com/k2XHNOy.jpg"
type: "user"
__v: 0
```

The second part is called ‘image2’, which is the place to store the book information that the user uploads. It contains the book information and the uploader's information.

See example below:


```
_id: ObjectId('63b294c15bae82ed201fde0a')
name: "yjhKj12"
La-La Land e: "doria"
url: "https://i.imgur.com/yjhKj12.jpg"
dbISBN: "9781473224469"
delhash: "jQFrWUKgOR3YKeN"
comments: "new and I love the stories. Trade with me la QAQ"
googlelink: "http://books.google.com.hk/books?id=rPufswEACAAJ&dq=isbn:9781473224469…"
author: "Frank Herbert"
booktitle: "The Great Dune Trilogy"
__v: 0
```

The third part is called "chatters," which is the place to store user chat records, such as the context they are chatting about, images, and timestamps.

See example below:


```
_id: ObjectId('63bfecf4d8df937aabad67d6')
self: "tangjaii"
notself: "doria"
randomhash: "52633"
lastdate: "2023-03-08"
▾ chatter: Array
  ▾ 0: Object
      dates: "2023-01-12"
  ▾ 1: Object
      user: "tangjaii"
      text: "Yo, I missu"
  ▸ 2: Object
  ▸ 3: Object
  ▸ 4: Object
  ▸ 5: Object
  ▸ 6: Object
  ▸ 7: Object
```


The last part is called "tradebuckets," which is the place for storing the offers that each user made, such as which book they are willing to give and take. This is used in the trade bucket list for displaying the list of offered books between users.

See example below:

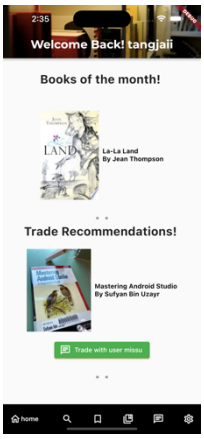
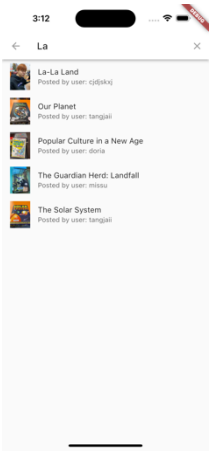
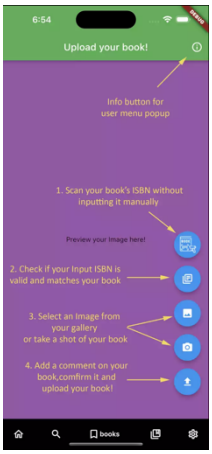
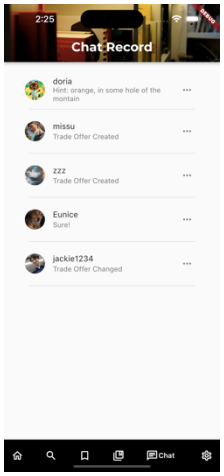
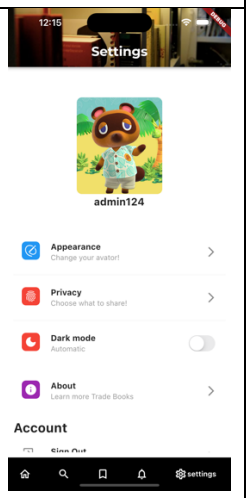

```
_id: ObjectId('63fe1fcf709483fbd2df091e')
missu "tangjaii"
notself: "doria"
randomhash: "12345"
lastdate: "2023-01-06"
lastuser: "doria"
status: "inprogress"
editing: "no"
▾ selflist: Array
    0: "ddAn97U"
    1: "TLdnhpD"
▾ notselflist: Array
    0: "yjhKj12"
    1: "RoJVmbT"
__v: 9
```


More detail of the usage for these data will be provided in the final report.

5. Testing

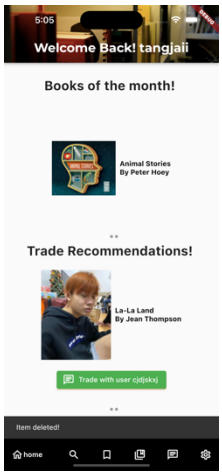
There will be four main parts to test out. The first will be UI/UX testing, which is mainly to test the front-end of the application for the user. The next part will be functional testing, such as testing different buttons to see if they work on other devices and as intended. The third part will be middleware testing, which is to test the API on node server and see if it is on Desired State Configuration [5] for Database. The last part will be beta testing, where we will ask some of our friends to test the app and provide feedback to see if there are any improvements needed. All of tests and procedure will be list out in a more detailed way during the final report.

For UI/UX testing, it will be something simple such as navigation testing as below:

Home Page	Search Page	Upload Page	Chat Record Page	Settings Page
				

The second part, which is functional testing, will involve testing buttons such as the upload button, delete button, logout button, and others to ensure that they are working as intended.

Example of deleting a book in inventory:

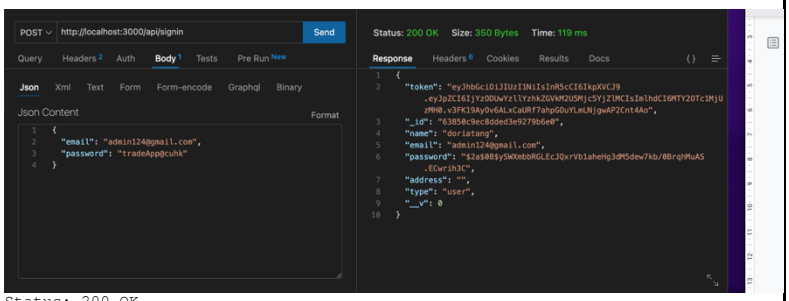


The third part will be testing out different APIs we built for the application. For example testing out the register, login, and the book info getter API.

Register API test:

The Backend server will send a POST request to the database and see if there is a match inside the database. It will then return a response.

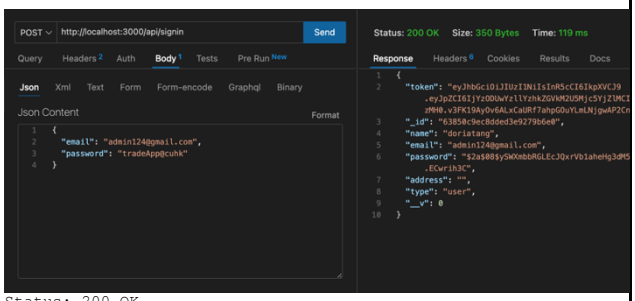
JSON Request	JSON Response	Testing Result

<pre>{
  "email":
"admin124@gmail.com",
  "password":
"tradeApp@cuhk"
}</pre>
	<pre>{
	  "name": "doriatang",
	  "email":
	"admin124@gmail.com",
	  "password":
	"$2a$08$ySWXmbbRGLEcJQxrVb1aheHg3dM5dew7kb/0BrqhMuAS.ECwrih3C",
	  "address": "",
	  "type": "user",
	  "__v": 0
	}</pre>
		
		Status: 200 OK

Login API test:

The Backend server will send a POST request to the database and see if there is a match inside the database. It will then return a response.

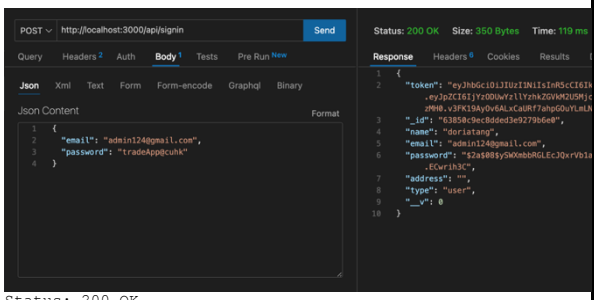
JSON Request	JSON Response	Testing Result

<pre>{
  "email":
"admin124@gmail.com",
  "password":
"tradeApp@cuhk"
}</pre>
	<pre>{
	  "token":
	"eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpZCI6IjYzODUwYzllYzhkZGVkM2U5Mjc5YjZlMCIsImlhdCI6MTY2OTc1MjUzMH0.v3FK19AyOv6ALxCaURf7ahpGOuYLmLNjgwAP2Cnt4Ao",
	  "_id": "63850c9ec8dded3e9279b6e0",
	  "name": "doriatang",
	  "email": "admin124@gmail.com",
	  "password":
	"$2a$08$ySWXmbbRGLEcJQxrVb1aheHg3dM5dew7kb/0BrqhMuAS.ECwrih3C",
	  "address": "",
	  "type": "user",
	  "__v": 0 }</pre>
		
		Status: 200 OK

Register API test:

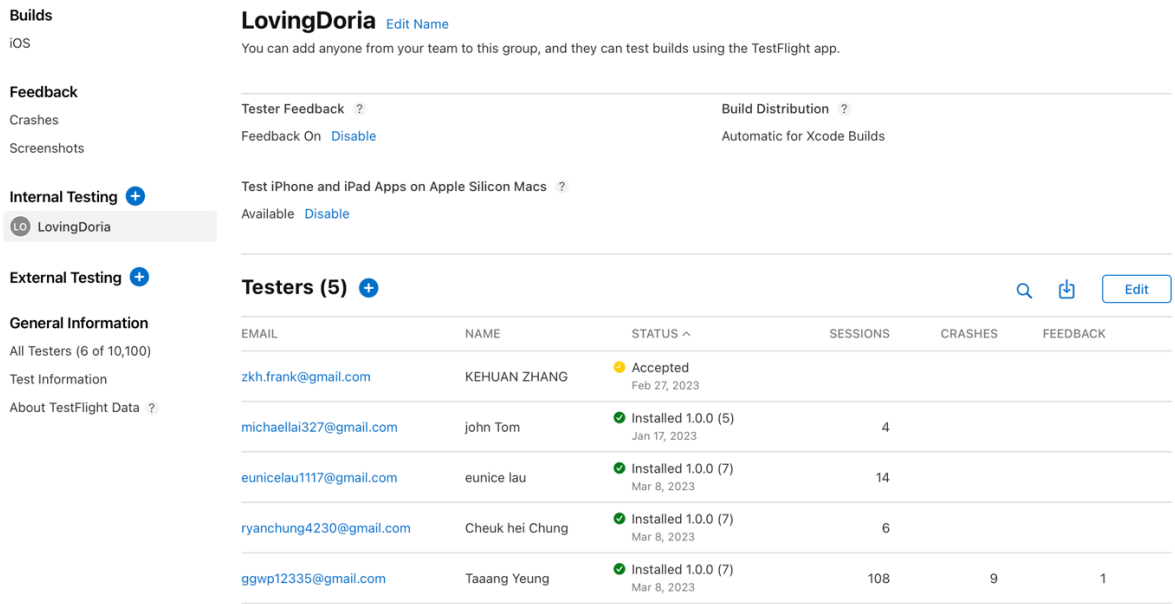
The Backend server will send a POST request to the database and see if there is a match inside the database. It will then return a response.

JSON Request	JSON Response	Testing Result

<pre>{
  "email":
"admin124@gmail.com",
  "password":
"tradeApp@cuhk"
}</pre>
	<pre>{
	  "name": "doriatang",
	  "email": "admin124@gmail.com",
	  "password":
	"$2a$08$ySWXmbbRGLEcJQxrVb1aheHg3dM5dew7kb/0BrqhMuAS.ECwrih3C",
	  "address": "",
	  "type": "user",
	  "__v": 0
	}</pre>
		
		Status: 200 OK

For the last part of testing, we will distribute the application to our friends at CUHK to beta test it and provide feedback for using the application and see how it compares to other similar applications like Carousell, and whether they have met any new friends after using our application. This will help us determine if our application is a successful and promising product.

After receiving feedback, we can then try to improve the app further by adding new buttons, making processes smoother, and addressing any issues or concerns raised by the testers. The image below shows the current users for testing the application and reports of them using the app.



6. Conclusion

In this project, we have created a platform for users to exchange books with each other to prevent waste while also providing opportunities for social interactions. During this semester, we have made significant progress comparing to the previous semester and aims to finish the entire app by the end of the semester. We have already created the key functions for the app, such as chat, trade, search, and settings, all of which are powered by a database with live data.

The application provides a channel for CUHK students to trade and interact with each other. The provided functions include everything a user needs, starting from register/logging into their account, searching through the search bar or browsing our recommendation list to find the book they want to trade/offer. They can first upload the books they want to trade away and send out a trade request using the trade interface to make an offer to another. A chat will then create for each other, letting user to discuss about the trade, where to meet up for the trade and the condition of the book, and among other things.

In conclusion, we hope that our application can offer users a chance to interact with others in CUHK, making our application a social application as well as an app that promotes a green campus [6] and reduces waste in our society.

7. Future developments and improvements

7.1.1 Improvements with physical devices on the network configurations

As we are testing and deploying our codes on XCode using the build-in simulator, we do not need to manually configure the simulate device since it is already in developer mode. However, when we tried it on our physical phone, we encountered some issues regarding the network and the application was unable to send or receive any requests or responses from the server. To address this issue, we may need to configure the iOS configuration files so that we can have permission to communicate with the server on our phone. We will also take notes from here and remind ourselves when deploying on Android phones.

7.1.2 Improvements in how we implement the codes

Since we were mainly focused on the application architecture, many of our codes were messy which just to fulfill the data flow and basic functions. For example, we were sending data and uploading images directly from the client-side, which could raise some security concerns. As a result of the unorganized codes, we did not have a technical specification for the codes that we were implementing, and many of the codes were not reusable. To make the codes clearer and more reusable, we will set technical specifications for the codes and APIs that we are using, to organize our application in a more detailed and well-structured way.

7.2 Feature Directions

7.2.1 Expanding to other categories

We hope that our application will not just stop with books, but expand to other categories such as electronics, housewares, and other items that people commonly throw away when they are no longer needed. For example, many bed sheets or refrigerators are thrown away after people finish their undergraduate studies and no longer live in the dormitory, and our application hopes to address this type of waste.

7.2.2 Subscriptions

We hope to include subscriptions in our application since we want it to be self-sustainable. Including subscriptions can help pay for the fees for renting cloud services or other maintenance fees. Users can enjoy an ad-free experience and up to 20 item slots for trading upon subscribing to the monthly subscription.

8. Weekly logbook

Date	Work done during the week	Challenges/Problem	Task planned for next week
30/1	Finished the Inventory system for books, Including the upload and display function	Needs to figure out how are we going to store the book image that user uploaded	Set up AWS for node server to deploy (deprecate use of local node sever on Mac)
6/2	Finish setting up AWS node server and got an apple account	It took time for apple to process the account and migrating the server from US to SGX	Create search page for user to search through the book list and create 'offers'
13/2	Almost finish the search function and thee book list page	Indexing the books for user to search through and listing it out on the page	Finish up the search function and start working on the chat page 'chatter'
20/2	Fully finished the search function and built a JSON format for storing the chat content	Designing the way to store the chats	Finish up the "chatter" and try to have our first message sent out!
27/2	'Chatter' is now done and it can display basic text in the 'Chatter'	Getting the time chip create/display correct in 'Chatter'	Try to push it on Test flight using the apple account we got earlier, and let other user to test it
6/3	It's On! The app is now available on the test flight and we can try it on ourself/others iPhone for the app	Build error during the process of archiving the build, which it was solved eventually	Before showing to user for testing, we wanted to implement the 'trade bucket', which let user to trade easily between users

13/3	Implementing the trade bucket, which let users to choose what to trade or make a counteroffer	Creating a JSON file format for storing the trade info	Finish the bucket by next week and improve the pap further

9. Reference

[1] Fong, Ben Y. F. et al. “Relationships Between Physical and Social Behavioural Changes and the Mental Status of Homebound Residents in Hong Kong During the Covid-19 Pandemic.” International journal of environmental research and public health 17.18 (2020): 1–18. Web.

[2] Yaser, Abu Zahrim. Green Engineering for Campus Sustainability. Ed. Abu Zahrim. Yaser. 1st ed. 2020. Singapore: Springer Singapore, 2020. Print.

[3] M. D. Hopp, M. Händel, S. Bedenlier, M. Glaeser-Zikuda, R. Kammerl, B. Kopp, and A. Ziegler, “The structure of social networks and its link to higher education students’ socio-emotional loneliness during covid-19,” *Frontiers in Psychology*, vol. 12, 2022.

[4] Medina, Juan Miguel, Ignacio J. Blanco, and Olga Pons. “A Fuzzy Database Engine for mongoDB.” International journal of intelligent systems 37.9 (2022): 5691–5724. Web.

[5] Sdwheeler, “Get started with desired state configuration (DSC) for Windows - PowerShell,” PowerShell | Microsoft Learn. [Online]. Available: https://learn.microsoft.com/en-us/powershell/dsc/getting-started/wingettingstarted?view=dsc-1.1.

[6] Sugiarto A, Lee CW, Huruta AD. “A Systematic Review of the Sustainable Campus Concept.” Behav Sci (Basel). (2022): 3.2. Learning Tool Dimension

The Chinese University of Hong Kong
Academic Honesty Declaration Statement

Submission Details

Student Name	YEUNG Tang (s1155144676)		
Year and Term	2022-2023 Term 2		
Course	IERG-4999-SJ01 Final Year Project II		
Assignment Marker	Professor ZHANG Kehuan		
Submitted File Name	1155144676_Sec_Interim_Report.pdf		
Submission Type	Individual		
Assignment Number	2	**Due Date (provided by student)**	2023-03-14
Submission Reference Number	3610773	**Submission Time**	2023-03-13 17:26:19

Agreement and Declaration on Student's Work Submitted to VeriGuide

VeriGuide is intended to help the University to assure that works submitted by students as part of course requirement are original, and that students receive the proper recognition and grades for doing so. The student, in submitting his/her work ("this Work") to VeriGuide, warrants that he/she is the lawful owner of the copyright of this Work. The student hereby grants a worldwide irrevocable non-exclusive perpetual licence in respect of the copyright in this Work to the University. The University will use this Work for the following purposes.

(a) Checking that this Work is original
The University needs to establish with reasonable confidence that this Work is original, before this Work can be marked or graded. For this purpose, VeriGuide will produce comparison reports showing any apparent similarities between this Work and other works, in order to provide data for teachers to decide, in the context of the particular subjects, course and assignment. However, any such reports that show the author's identity will only be made available to teachers, administrators and relevant committees in the University with a legitimate responsibility for marking, grading, examining, degree and other awards, quality assurance, and where necessary, for student discipline.

(b) Anonymous archive for reference in checking that future works submitted by other students of the University are original
The University will store this Work anonymously in an archive, to serve as one of the bases for comparison with future works submitted by other students of the University, in order to establish that the latter are original. For this purpose, every effort will be made to ensure this Work will be stored in a manner that would not reveal the author's identity, and that in exhibiting any comparison with other work, only relevant sentences/ parts of this Work with apparent similarities will be cited. In order to help the University to achieve anonymity, this Work submitted should not contain any reference to the student's name or identity except in designated places on the front page of this Work (which will allow this information to be removed before archival).

(c) Research and statistical reports
The University will also use the material for research on the methodology of textual comparisons and evaluations, on teaching and learning, and for the compilation of statistical reports. For this purpose, only the anonymously archived material will be used, so that student identity is not revealed.

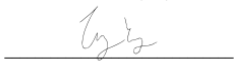
I confirm that the above submission details are correct. I am submitting the assignment for:

[X] an individual project.

I have read the above and in submitting this Work fully agree to all the terms. I declare that: (i) the assignment here submitted is original except for source material explicitly acknowledged; (ii) the piece of work, or a part of the piece of work has not been submitted for more than one purpose (e.g. to satisfy the requirements in two different courses) without declaration; and (iii) the submitted soft copy with details listed in the <Submission Details> is identical to the hard copy(ies), if any, which has(have) been / is(are) going to be submitted. I also acknowledge that I am aware of the University's policy and regulations on honesty in academic work, and of the disciplinary guidelines and procedures applicable to breaches of such policy and regulations, as contained in the University website http://www.cuhk.edu.hk/policy/academichonesty/.

I declare that I have not distributed/ shared/ copied any teaching materials without the consent of the course teacher(s) to gain unfair academic advantage in the assignment/ course.

I also understand that assignments without a properly signed declaration by the student concerned will not be graded by the teacher(s)



Signature (YEUNG Tang, s1155144676)

13/3

Date

Instruction for Submitting Hard Copy / Soft Copy of the Assignment

This signed declaration statement should be attached to the hard copy assignment or submission to the course teacher, according to the instructions as stipulated by the course teacher. If you are required to submit your assignment in soft copy only, please print out a copy of this signed declaration statement and hand it in separately to your course teacher.