**Java Application on Android Programming**

Khai Phan Thanh (100901164)

Object-Oriented Software Development II - Java (BTP400 - NCC)

Professor Peter Liu

March 26th, 2020

Word Count: 2616

**General Reporting and Executive Summary**

Java has been the top-five programming language for a very long time and the following reasons. Java is a platform-independent language, meaning that it can run on many machines and in any platform (Windows, Mac, Linux or UNIX) as long as the device has a Java Runtime Environment (JRE). Additionally, Java offers a massive class library that greatly simplifies the job of the programmer, saving time and being more efficient in completing tasks. One of the advantages that Java has over other languages is the ability to create a Mobile Application, whereas other languages only can create a Desktop Application.

In most Mobile Applications nowadays, especially Android, about 90% of them are written in Java. With that saying, we find a great advantage to discover and learn about Mobile Development as it will be a great asset to have in our careers. The approach we have is to create an APIs functionality search for users. Before the user can use the API functionality, they need to create an account. Then the app will get the user data, which will transfer to an online Cloud Database via an integrated connection. The app will also fetch the data from the Cloud Database upon verification when the user chooses to log in. After user login, they will have several functionalities to choose from, and each function is an API. API will take the user's argument, transfer it to the database and return a matching result (if any/applicable). Additionally, we will be working with Travel APIs that will handle flight and hotel searches to find the cheapest dates, prices, and routes for a trip.

For the task, we would need to learn a new technology platform (Android), a new codebase (Android Studio) and APIs that will use in the app. There is an Android Developer Fundamentals course offered online made by the Google Developers Training team, which will help us learn the essential concepts needed in our app. At the end of this applied research, we would want to deliver why Java is a preferred language for the Android app and how beneficial it is for programmers from the back-end to front-end in development. Below is a timeline which we expect to get the progress of research and application completed.

**Expected Timeline:**

|  |  |  |
| --- | --- | --- |
| **Week** | **Date** | **Deliverable** |
| 8 | **February 29(Saturday)** | Executive Summary (Team) Completed.  Start research on APIs. |
| 9 | **March 3 (Tuesday)** | Getting Start with the app, begin with the UI (Front-End side).  Implement Cloud Database for registering and login. |
| 9 | **March 7 (Saturday)** | Cloud Database ready to store and fetch data from the app.  Decide on APIs and functionality to use. |
| 10 | **March 10 (Tuesday)** | ~~Implementing Hotel API.~~  **\*\*Revised: Implementing Flight API\*\*** |
| 10 | **March 14**  **(Saturday)** | Finish and Tested API #1.  ~~Implementing Flight API.~~  **\*\*Revised: Implement Hotel API\*\*** |
| 11 | **March 17 (Tuesday)** | Progress Report (Team) Checkpoint.  Finish and Tested all APIs, and wrap up with any extra details |
| 11 | **March 21 (Saturday)** | Final Draft for the Report ready.  All APIs function is ready to go. |
| 12 | **March 24(Tuesday)** | Final Report "report is ready to revise."  Enhance UI for APIs results. |
| 12 | **March 28(Saturday)** | Final Report and Application (Team) Completed. |

**Progress Report Checkpoint:**

At this point, according to the Expected Timeline, I have fully developed a Cloud Database that allowed the program to send and get information of user (username and password) back-to-back to stored and compare, respectively. One of the problems I currently experience is access to the Database table via a web browser. At the moment, it seems that the website is down for some reason (probably for maintenance), but the app is still able to send and fetch data to the server, so I assume everything is fine (that the database is not corrupt or shutdown).

For the 2 APIs, I was planning to start with the Hotel API and then Flight API. Still, then the progress of finding a Hotel API is a bit more challenging, so I've decided to start with a Flight API because it seems that there are more API for flight information than a hotel. The one I chose for flight is from Skyscanner. The problem I have with the Flight APIs is that they do not provide any real-time database. What that means is, the price might change variously throughout the date, but because this is a free one, it will just display an estimated cost. The real-time database might available in the paid version, from what I know. Because of that, I would assume the Hotel APIs would provide the same type of data as well, and indeed, I don't want to spend $50 or $100 for the APIs to get the real-time data.

At this moment, we are on track with the timeline we have expected. Only one minor change is the switch of Flight API and Hotel API, so what we have left at this point is the implementation of Hotel API and some final touch on the UI.

Attached at the end of this Progress Report are pictures for what I have completed so far. At this moment, the apps contain the Main Activity to display the first page of the app, the Login stage. User will have an option to create an account if they do not have one at the moment or login if the user is already own an account. After logging in, there will be two options for a user to choose. At the moment, only the Search Flights option is working, and the option Search Hotels will show up as a blank page as it is still in development.

My partner and I were contributed evenly to this project. Carl is responsible for the front-end user, which consist of all of the field placement and the UI. I am doing the back-end, where I will handle all of the Java processes such as field placement data fetching and pass that to either API or Cloud Database.

In summarize of all the methods I have done, start with Cloud Database; after I convert the field data into String, I will pass it to RegisterRequest and LoginRequest depends on the purpose and each class will have a method to POST the data into the server. The difference in each request is that the Register will POST the data to the server and store it into the database. Login will only POST the data, and then it will fetch the data that exists in the table and compare each field and return the state of the object. If the purpose is equal to one of the fields in the database, the return state will be 'success,' and the user will be logged in. For APIs, there are some arguments that I have to add to filter the results (such as price limit and dates), and there is a specific format for that. In each APIs, there is a particular key, and hostname used to access the database. After the result has been filtered, it will store into an object (this case will be Flight object) and use that to display into Detailed format, which has been prepared.

**Final Report**

1. **Research Outcome**

According to our expected timeline, we have completed all of the required tasks we have planned. The first task is Login/Register account via a Cloud Database that stored information of the account (username and password). The second task is about APIs implementation for Flight and Hotel to sort the result in the database. For the second task, we would consider it's only complete at about 80% to 90%. As mentioned previously in the Progress Report, most of the APIs available in the market for free at the moment only include a demo or standard result database, not the real-time result change. They also tend to have an average success rate only with a long delay time.

1. **Research Experience**

Cloud Database is the most straightforward task to work on, in my opinion, because there is a lot of servers offered a trial on their database (with some constrain on the number of tables, connections and size). Still, for school work-related that should be enough. The one that we picked is PHPMyAdmin because it seems to be the easiest and fastest one to access (in both Android app and website management) with a built-in login and register method available in the system. However, the PhPMyAdmin appears to have a minor problem though it doesn't affect the overall experience. The server seems to work fine, but the website to access the database usually under maintenance, but the program is still able to access the database, so that's a good sign.

|  |  |  |
| --- | --- | --- |
| **API Type** | **Source** | **API Link** |
| Cloud Database | PhPMyAdmin | Database (<https://databases.000webhost.com/index.php>)  **Username**: id11798421\_register **Password**: id11798421\_register |

And as mentioned previously in the Progress Report about the APIs connection and issue, it seems that the Flight API has more demand and usage than Hotel API. We were having trouble trying to find a work and stable API for the Hotel at first, but then after the 3rd round of searching (with the first 2 round before we decided to switch to Flight API), we finally manage to found one. Below is the table and information about the APIs that we use. Each API is connected to a source server contains database of corresponding API type. The Latency is the delay time server response to the request, and the success rate is the percentage rate server returns a result.

|  |  |  |  |
| --- | --- | --- | --- |
| **API Type** | **Source** | **API Link** | **Latency / Success Rate** |
| Flight | skyscanner | Rapid API ([https://rapidapi.com/skyscanner/api/skyscanner-flight-search? endpoint=5aa1eab3e4b00687d3574 279](https://rapidapi.com/skyscanner/api/skyscanner-flight-search?%20endpoint=5aa1eab3e4b00687d3574%20279)) | 466ms / 78% |
| Hotel | hotel api dojo | Rapid API ([https://rapidapi.com/apidojo/api/hotels4?endpoint=a piendpoint\_2bd1c370-acdc-45e5-9791-7fb2b497dfb3](https://rapidapi.com/apidojo/api/hotels4?endpoint=a%20piendpoint_2bd1c370-acdc-45e5-9791-7fb2b497dfb3)) | 1780ms / 96% |

In the description, the Cloud Database has a Database name id11798421\_register, and within that, it has a table name “user” that stores the username, password, and confirms password information. This information will be added when the user is registered or fetch out to compare when the user is login.

For Flight API, there is a lot of argument to be filled in the link format.But to simplify the work for the user and us, we only take the most important one is the Departure Date, the Return Date, the Origin and Destination (there are some more arguments such as time-zone, language, currency,…). After we get all the data from the user, we will parse the argument into the link and send the link to the API server, which will simplify the Database and return the result.

The same concept applies to Hotel API, which will take the Destination, Number of Guests, Check-In Date and Check-Out Date, parse the arguments into the link and send that to the API server to filter the Database. One major thing to notice is that at some point during the day, the result could take longer than the latency average and at some point, it would be faster if the user go back and hit on the search again.

Most of the work we did isn’t include in our Java learning in class. Most of the content is beyond class learning level and so for most of the coding, we have to figure out on our own. For example, the Register and Login class, when they send the account object to the server for either store or compare, wil returned a JSON object that store the response status and the program will depend on the status to decide on the next appropriate step. For APIs, we are doing something similar to the Server and Client, as we learnt recently. Still, instead of using a local Client-Server connection, we have to use OkHttpClient to call into the server. In each call, it must have a URL (containing the arguments for the server to sort the Database), which is a host link to the server and a unique key to ensure we have the permission to access. Same as the Database, all of API will return a JSON Object, which contains the status responded from the server and an object. If the respond equals with success, an object with non-null data will be return and otherwise.

1. **Team Work Experience**

For this project, my teammates and I decided that each of us is going to handle a separate part that most certainly independent from each other. Carlianz is responsible for the connection of front-end users with the program, which is the XML files and some part of the .java files. I was responsible for the back-end program I manage all of the coding logic for connection between the program and the API or Database.

During this group work time, we were plan to see at least two days a week so we can work on the design, logic and catch-up on each other's code but because of the pandemic, we have to stay at home and communication online, and that's made the job difficult. In the past, whenever one of us have a problem with the code, we all sit down and find a way to resolve it, whether it is front-end or back-end. But now, we can only call or send messages to describe the issue and sometimes it's hard to imagine and understand. But the more we do, the more we get used to the new changing and get adapted to that well. In the future, it is always better to do group work in person because it delivers better communication and make progress go smoother.

On the scale from 0 to 10, I would rate my teammate a good 8. The reason I took off 2 marks is that we rarely communicate via WhatsApp unless I start it first and address the issue, which is a bit inconvenient for me because I'm not sure if whether my teammate knows the information or not without continually communicating.

1. **Future Research**

In the future, for the APIs, we would want to add more options for the user when finding the result, such as currency selection, flight/hotel rating preferences or many more. In addition to that, we will surely work on the UI to make it looks much better instead of plain colour and template since we have a time limit and the pandemic that leads to the difficulty of communication.

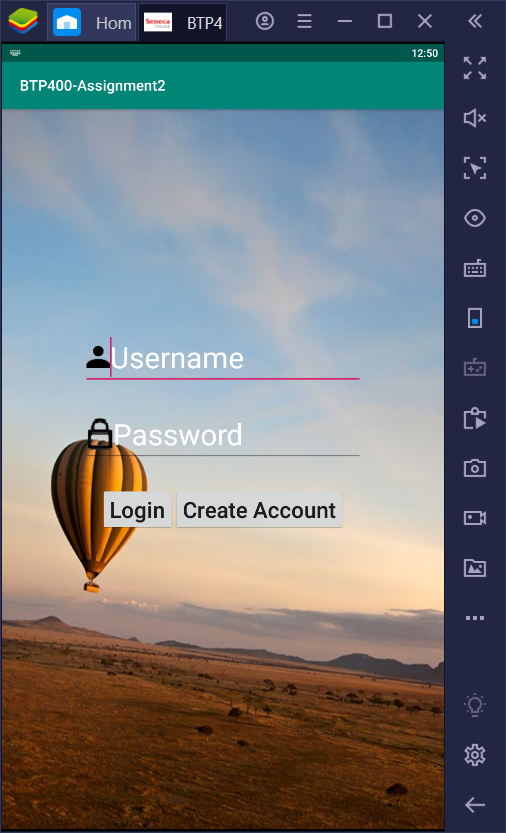
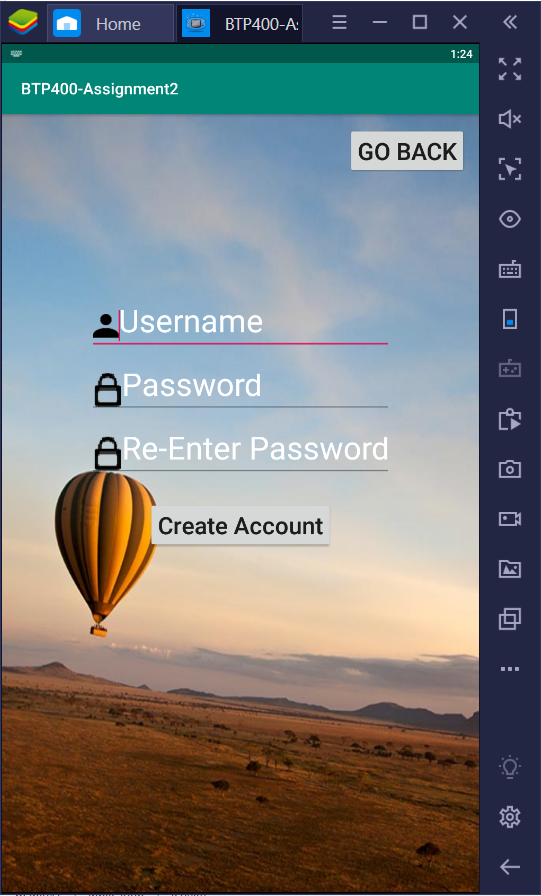
We have a plan to implement a calendar into the app. What that means is, user can book that flight and hotel, and it will automatically add to the calendar. The calendar will be about the same type and layout with the calendar implemented for the Flight and Hotel dates. However, this calendar seems to take a bit more time to implement because we have to get the dates and the information and transfer it over to our calendar. Then we also have to set an event to appear on each calendar day that even occurs. Moreover, when the user logout and login again, they can still retrieve the information, and when a different user login, it will show a different event.

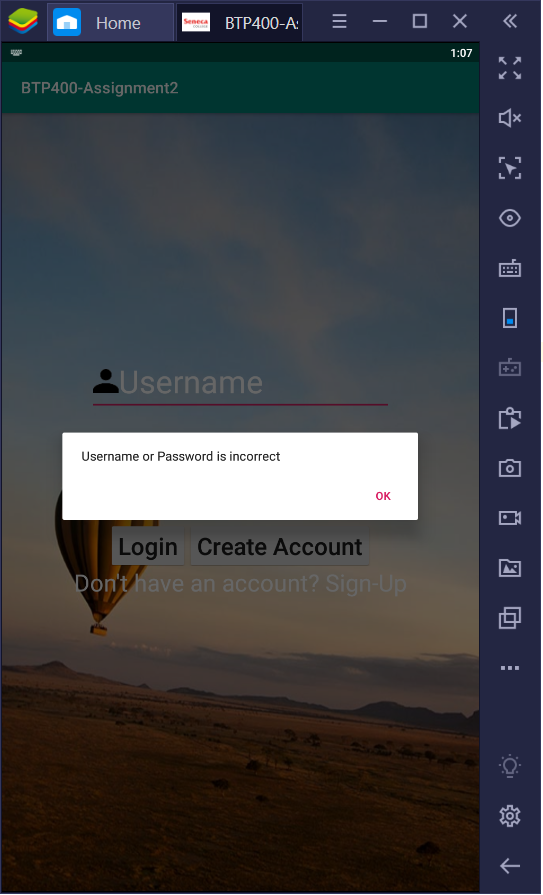
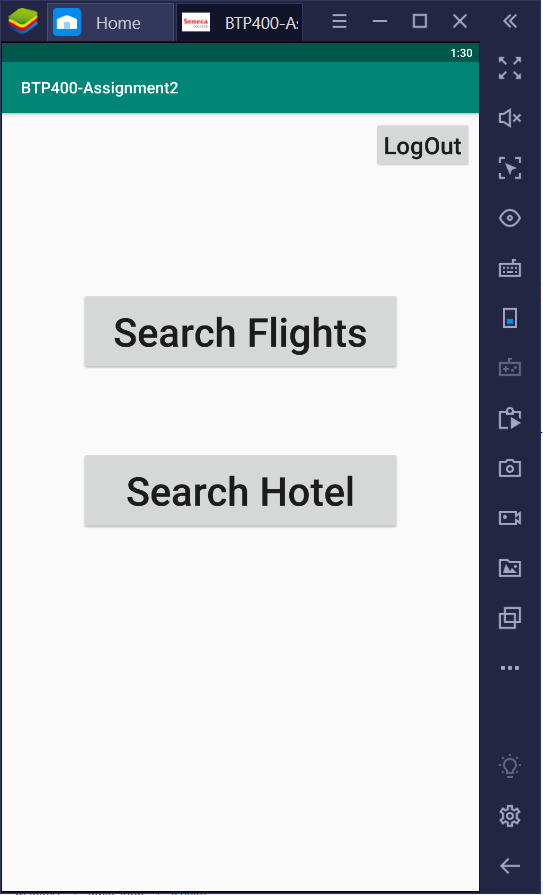
1. **Link Accessibility**

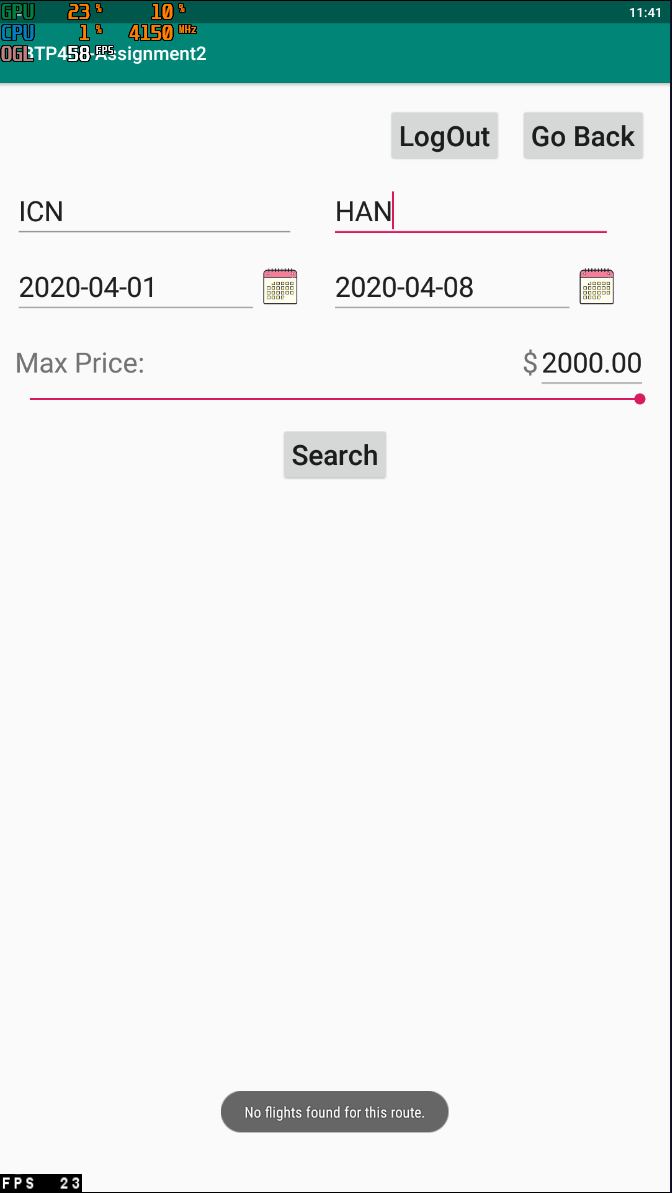
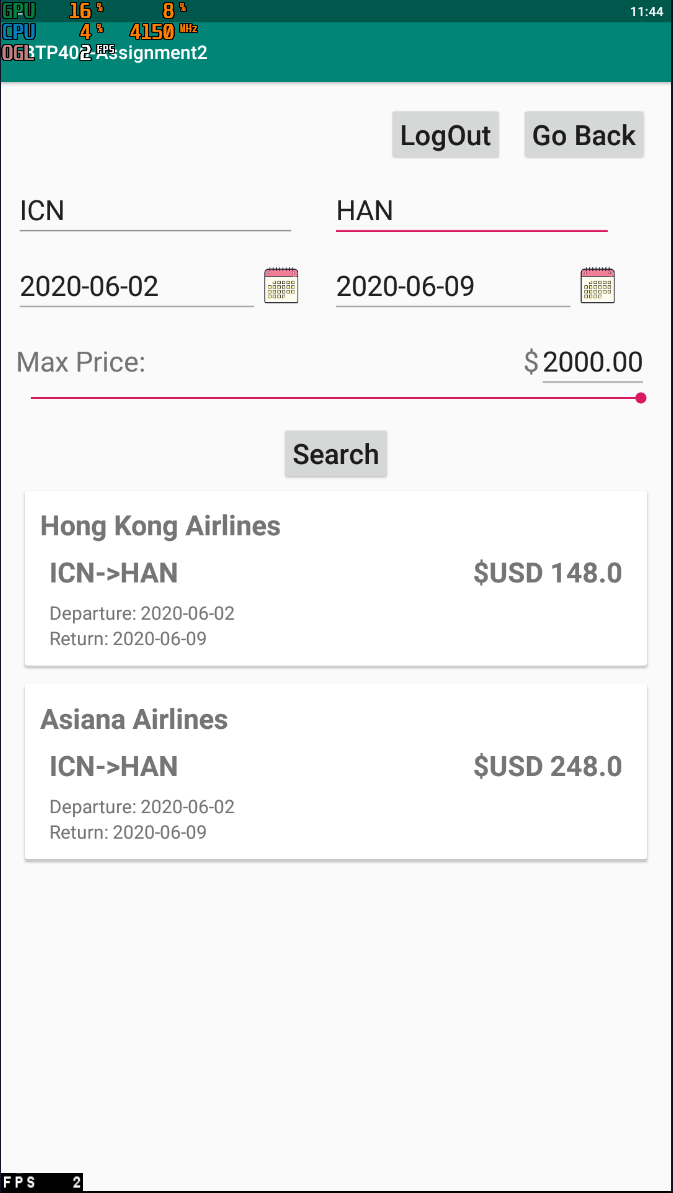
* **Github:** <https://github.com/phanthanhkhai480/BTP400/tree/master/Assignment2>

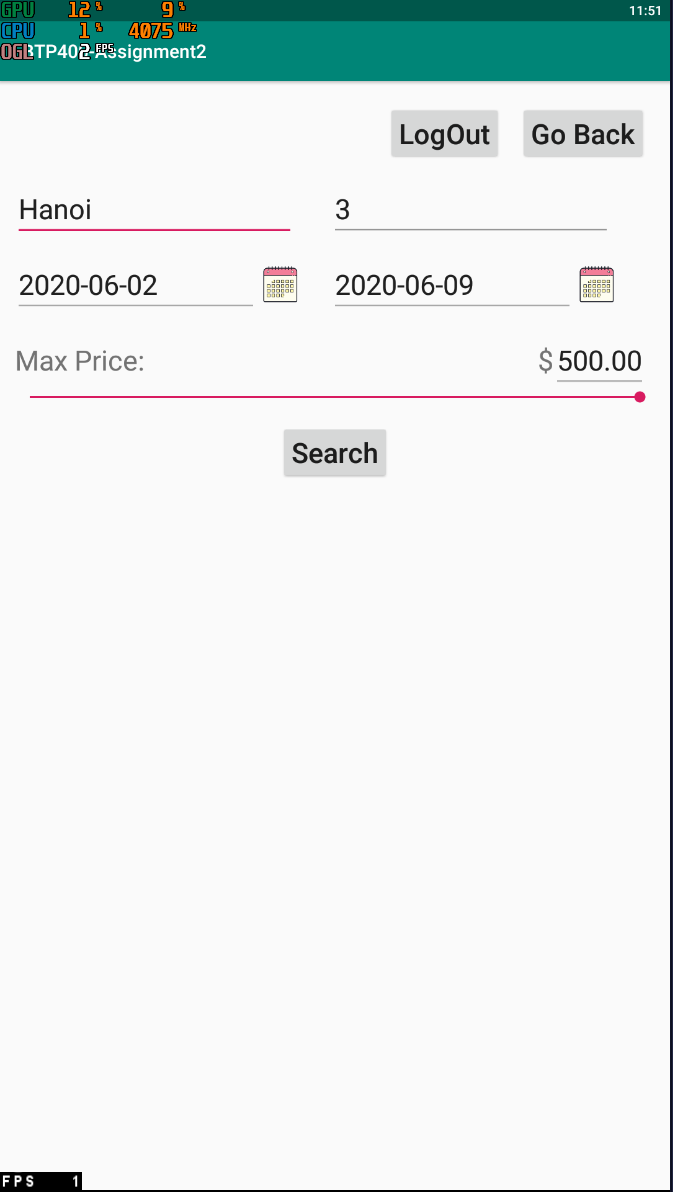
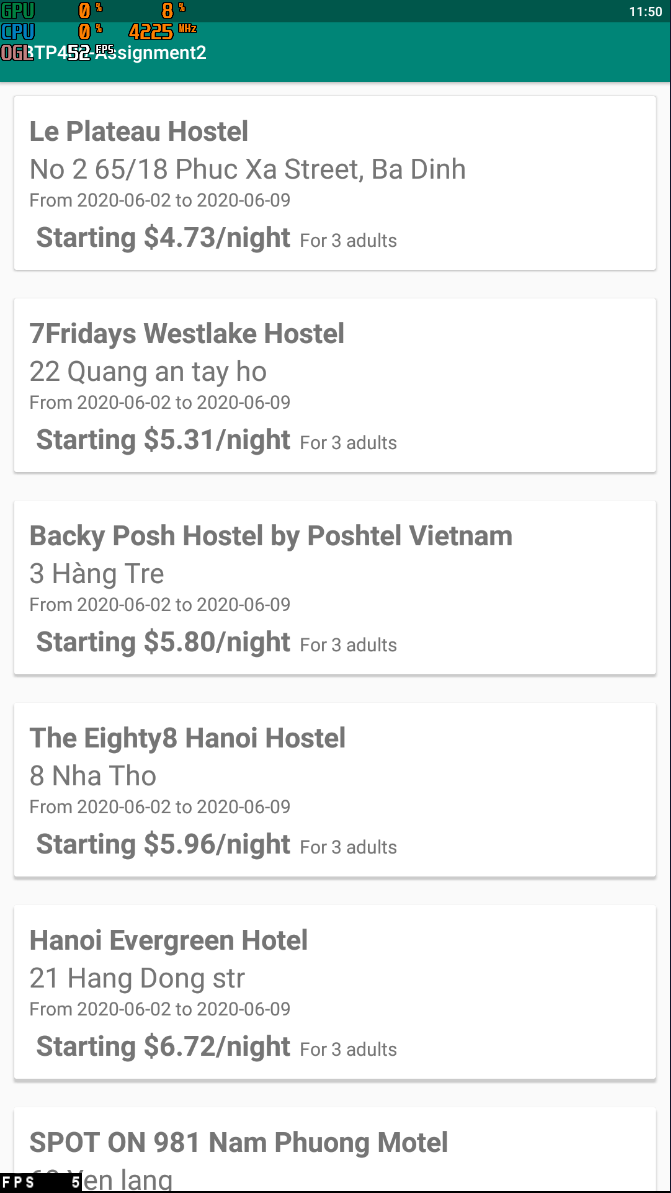
1. **Screenshot**

MainActivity.java RegisterActivity.java

 LoginError SecondActivity.java

FlightActivity.java (Result Not Found) FlightListAdapter.java (Result Found) HotelActivity.java HotelListAdapter.java (Result Found)

** **

**Resources**

ADMEC Institute. “Role of Java in Android App Development.” *ADMEC MULTIMEDIA INSTITUTE*, 24 Apr. 2018, [www.admecindia.co.in/blog/role-java-android-app-development](http://www.admecindia.co.in/blog/role-java-android-app-development).

Goswami, Himalaya. "Why Is Java Preferred for Developing an Android App? - Quora." *Quora.com*, April 4th. 2017, [www.quora.com/Why-is-Java-preferred-for-developing-an-Android-app](http://www.quora.com/Why-is-Java-preferred-for-developing-an-Android-app).

Kanjilal, Joydip. "Java Mobile Programming for Android." *Java Mobile Programming for Android - Developer.com*, February 26th. 2016, [www.developer.com/java/j2me/java-mobile-programming-for-android.html](http://www.developer.com/java/j2me/java-mobile-programming-for-android.html).

Cloud Database: (000webhost PhP)

Database List: <https://www.000webhost.com/members/website/received-blueprints/database>

Database Table: <https://databases-auth.000webhost.com/signon.php?sid=aXdPMDk3c204V2xWMDBSN0RKSWtvc3dWb0doNklubkx3S1JkblJKZkJIZGd2K203T0J4dllvOVZGcVgrbTFLR1R1RVBLK25rSjUrbjB4bFB3SG1odlE9PQ==>

APIs:

Flight: [https://rapidapi.com/skyscanner/api/skyscanner-flight-search?endpoint= 5aa1eab3e4b00687d3574279](https://rapidapi.com/skyscanner/api/skyscanner-flight-search?endpoint=%205aa1eab3e4b00687d3574279)

Hotel:<https://rapidapi.com/apidojo/api/hotels4?endpoint=apiendpoint_2bd1c370-acdc-45e5-9791-7fb2b497dfb3>