WatsLoo

Private Github repository link: https://github.com/646-group8/mobile-app

Mengyao Zhang m473zhan@uwaterloo.ca 20912885
Zishuo Xu z463xu@uwaterloo.ca 20900288
Yitao Hou y39hou@uwaterloo.ca 20615100
Lu Wang l647wang@uwaterloo.ca 20896426
Yaowen Mei y5mei@uwaterloo.ca 20470193
Ting Gu t39gu@uwaterloo.ca 20924386

1. Demo Summary

Watsloo is a tour-guide application that accompanies users to explore the history and story of every corner of the UW campus and allows users to share their own stories.

We will demonstrate the functions including:

1) The app requests permission from the camera and location when users open it.
2) The overview pattern of Google Map of University of Waterloo campus.
3) The meaningful spots in UW campus have been correctly marked.
4) Users can use the map smoothly and change the view of the map, to discover more spots.
5) Users can click the specific spot to find a bunch of stories related to the spot.
6) Users can click on a story's title and read the story in detail. There are also many vivid pictures in the spot's detail page.
7) Users can click the "Submit My Own Story" button to upload their own stories about the spot. The operation is simple, what they need to do is edit the story's title and description.
8) Users can also take interesting pictures and upload to the story if they want.
9) At last, we will show the data structure in firebase, which is our app's database.

These functions above are generally real except for the loading of data. As we haven't completed the function of requesting data from firebase, these data of spots are hard-coded data. You can think of the process of querying data from firebase as simulated. But the function of uploading data has been realized and the part of uploading is real. Apart from this, all shown interfaces and their jump relationship are real.

2. Status report

2.1 Map part

This is the main interface to users of our app. Currently, we have achieved the overview pattern by importing a map using Google Map API and putting several markers as the preliminary sample spots. We also achieved functions that users can click on the spot and read interesting stories related to the spot.

Difficulties: We met a few problems when configuring the Google Key and Google map as developers of this part are in China now.

future plan:

1) Furtherly achieve the following pattern to show and update the user's location in real-time.
2) Read the stories from firebase. Optimize the user interface in a more elaborate way.

2.2 Contribution part

Users can upload their own stories with this activity. Our current progress are:

- Ask permission and obtain the user's realtime GPS information
- Ask permission and read a picture from album and extract the GPS information from the picture
- Ask permission and take a picture via systerm's camera app
- Upload GPS information, user input stories, and a picture to our Firebase database

The difficulties we are facing are:

- Reading GPS information from an EXIF picture is considered as a "sensitive" activity after Android API 29 (therefore it requires a special method), our code currently only tested on API31, we need to find a way to make it compatible with API 24, and all the way up to API29, this is going to be difficult.

Our plan for the next month is:

- Test compatibility and find solutions so that our code can work with lower API levels,
- Provide a user interface so that users can drag and drop a pin on google map to input the GPS value.

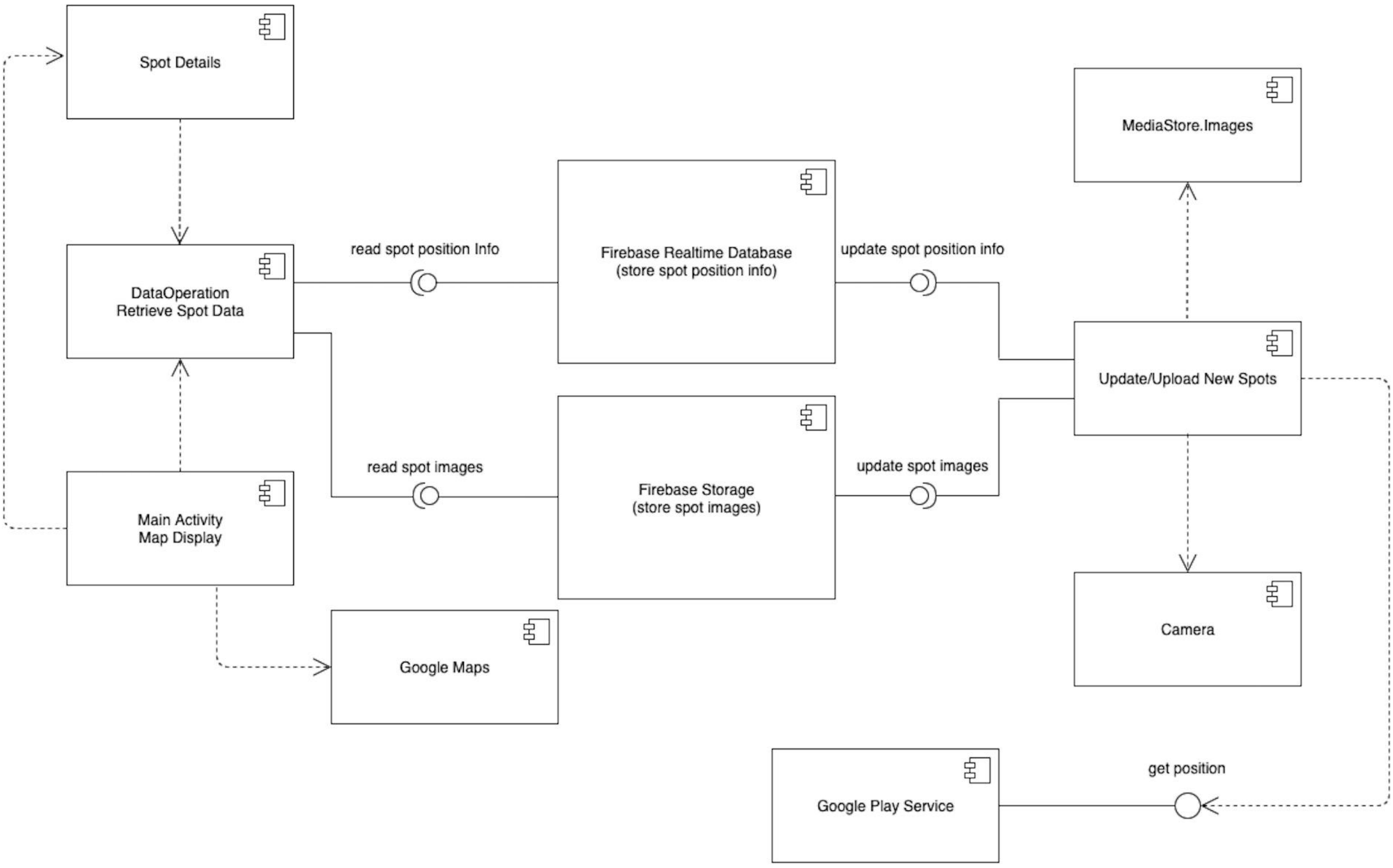
2.3 Firebase part

We chose Firebase realtime database, a NoSQL cloud database, to store app data. We constructed a class to represent items and encapsulated item upload/download methods for other components to use.

We face several difficulties, including how to choose the most reasonable abstraction for items and how to use the APIs in the Firebase SDK correctly and efficiently.

Our next plan includes:

1) May design a more suitable representation for items.
2) Deal with concurrent modification and guarantee that only one user modifies the data at one time.
3) Implement a more efficient add story method without getting existed stories before adding.



Our system follows the repository architectural style. The repositories are Firebase Realtime Database and Firebase Storage. They execute on Google Firebase in cloud server. Other components execute on the local mobile app of users.