**FbShare: An Android Mobile Cloud App to Share Files**

**Course Project Report for ESE 543: Mobile Cloud Computing At Stony Brook University**

**Compiled and submitted by:**

**Shashank Rao (111471609)**

**Abstract:**

This project report summarizes the work done with regards to the android app: FbShare. FbShare is an app that uses a user’s Facebook account to login, make file transactions with his friends and then notify these friends through Facebook interactions. It uses Firebase Realtime Database and Firebase Storage for handling all the app data, including the user data. The report also includes some basic evaluations of the app.

**Introduction:**

In today’s world, technology is almost completely assimilated with everyday life. People wake up in the morning usually to a smartphone alarm, read news on various smartphone apps and use various other smartphone apps to perform various other day to day tasks. On the online social front, Facebook has emerged as one of the, if not the biggest sensation of these times. Almost every individual has a Facebook account where he/she notifies the world of his/her daily tasks and emotions. Almost every detail, from a simple visit to the coffee shop to visiting ‘The Louvre’ is “checked-in” and posted on Facebook. Similarly every occasion in the user’s life is also shared on Facebook.

FbShare attempts to use this interactivity between Facebook users to provide notifications whenever the user shares a file with his/her Facebook friends. FbShare allows the user to share five different types of files - pdf, docx, image, audio and video - and whenever the user wishes to share a file, Facebook posts are made to notify the people with whom he/she is sharing the file regarding the file. It allows for easy notifications by the user to his friends. The philosophy behind FbShare is that people are more aware of Facebook notifications more than any other form of notifications, since almost every user has a virtual presence on Facebook which makes them try to keep updated on every facet of his/her friend’s life. The user can also share links he/she finds interesting directly with his/her friends through this app.

FbShare uses Firebase Realtime Database and Firebase Storage to provide backend storage for the app and its users. The user information, groups informations as well as the download URLs for files uploaded and shared between the users are all stored on the Firebase Realtime Storage. Firebase Storage provides the actual storage of the files which are uploaded. As a result, the app is very light and requires very little storage space on the user’s smartphone.

**Design:**

FbShare was designed with two goals in mind - one, to provide easy user interaction, and two, to be lightweight. As a result, the app that was to be created had to have the following design parameters - it should not have a large number of variables and/or text on the screen and no user/app data had to be stored on the app. The following flowchart gives an insight into the app flow which the user experiences:

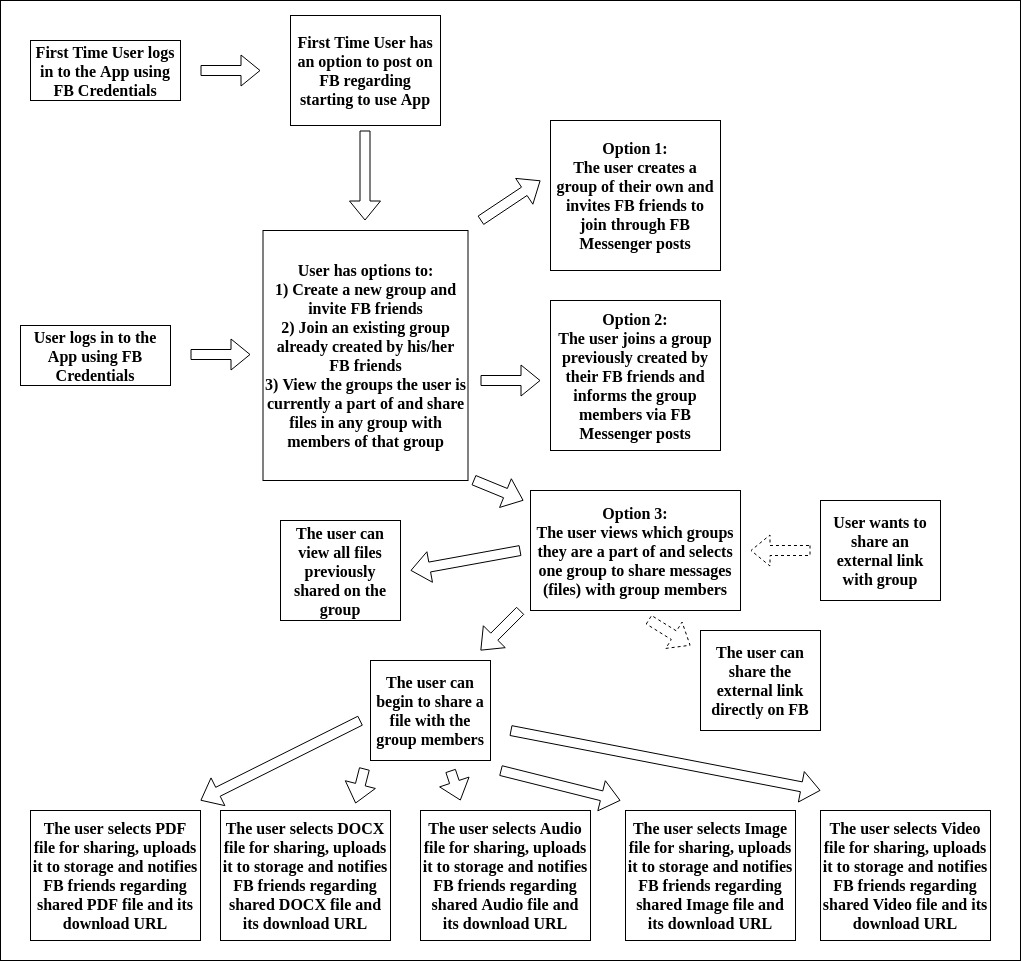


Fig. 1: Functional flow of the App for the User

As can be seen from the flow, user interface is kept simple and the user is only kept aware of all the necessary details required for him/her to perform actions, i.e., create a group between FB friends or join an existing group created by an FB friend or enter the group page and start sharing files. These functions do not require a lot of user input and user operation is kept as simple as possible. This achieves the easy user interaction objective.

One of the main reasons user interaction is so easy is because most of the details regarding the user is collected from Facebook and stored in the background using Firebase database and storage. Since no data is stored by the app, but rather in the cloud, the app created is very light. This fulfills the second objective too.

**Implementation:**

This section gives a brief overview into how the app is implemented. Before starting the android development, the developer has to create an account with Firebase as well as with Facebook Developer, since Firebase is being used for storing all the data, and Facebook is being used for notifications as well as to obtain user details like user profile, user friends and user email. Once these accounts have been created and the app has been registered with both of them, the developer can begin android development. The following steps describe each of the Activities that have been created for the FbShare app:

1. FacebookLoginActivity: This is the first activity that is called when the app is loaded. The user interface only consists of a simple Facebook login button, which leads to the Facebook login page of the native Facebook app installed on the smartphone. Once the user enters his/her credentials, he/she is directed to the next page. Behind the scenes however, there is a lot of processing that takes place. OpenGraph API calls are made to the Facebook APIs which returns the user’s data (profile, friends list and email). The user is then authenticated with Firebase Authorization, which is required for storing the sensitive user data. Once authorization is received, the Facebook data obtained for the user is stored in the Firebase Realtime Database (FbUserData and FbIdToUserIdMap parent nodes are created with respective child nodes holding user data) for later extraction and processing. During this login process, it is also checked if the user is logging in for the first time. If he/she is, then the control flow is directed to FirstTimeLoginActivity, else the control flow is directed to MainActivity.
2. FirstTimeLoginActivity: In this activity, the user is given an option to share a post on Facebook stating that he/she has begun to use this app. This activity makes use of a ShareDialog object (which is a Facebook object) in order to create the Facebook post. If the user does not want to post on Facebook, he/she can choose so. Once posting (or not), the control flow is directed to MainActivity.
3. MainActivity: This is the main screen activity of the app, and the user can select whether to create a new group, join an existing group, view the groups he/she is a member of or logout from the app. Depending on the selection, the control flow is directed towards CreateNewGroupActivity, JoinExistingGrpActivity or ViewExistingGroupsActivity. The logout button, removes user’s Firebase authentication and logs him/her out of the app.
4. CreateNewGroupActivity: This activity is used by the user to create new groups. The user inputs only the Group name; checks are performed on this name to ensure no duplication. If the name has not been used previously, the user is directed to the next screen. Behind the scenes, the Group details are stored in Firebase Realtime Database (UserGroups parent node which contains all group details). Duplication check is also performed by querying the database. After creation of group, control flow is directed towards InviteFriendsToNewGroup activity.
5. InviteFriendsToNewGroup: This activity is used to ask whether the user wants to invite his Facebook friends to the new group created. In this activity, MessageDialog object (also a Facebook object) is used. This allows user to message his friends using Facebook Messenger App rather than post on Facebook. After the user completes messaging, the control flow is directed back towards MainActivity.
6. JoinExistingGrpActivity: This activity is used to present all the groups created by the user’s friends to the user. The user can see group names as well as the group creators (Group Administrators). The user interface uses FirebaseRecylerAdapter object to load the group details directly from the database and display in RecyclerView and CardView interface. The user can click on any card to be directed to the Group Joining Confirmation page, i.e., the control flow is directed towards JoinGrpConfirmationActivity.
7. JoinGrpConfirmationActivity: This activity is used to provide the details of the group (name, admin and group members) to the user and the user can provide confirmation as to whether he wants to or doesn’t want to join the group. If he/she decides to join the group, the control flow is shifted to InformGrpMembersAfterJoining, else it is shifted back to JoinExistingGrpActivity.
8. InformGrpMembersAfterJoining: This activity once again uses MessageDialog object to post messages to the user’s friends regarding him/her joining the group (using Facebook Messenger app). After this, the control flow is directed towards MainActivity.
9. ViewExistingGroupsActivity: This activity is used to give the user a list of all the groups of which he/she is a member. The user interface again uses FirebaseRecyclerAdapter object to load group details. Once the user clicks on any card, he/she will be directed to the Group Activities screen, i.e., the control flow will shift to ShareInGrpActivity. ViewExistingGroupsActivity is also the activity that is opened when an authorized user tries to share an external link through the app.
10. ShareInGrpActivity: This is the main group activities screen, where the user can choose to decide whether to share a file from his/her device, share an external link or view all files/links shared within the selected group. According, the control flow will be directed towards UploadFileOntoStorageActivity, ShareOutsideLinkUrlActivity or ViewAllMsgsInGrp activities respectively.
11. UploadFileOntoStorageActivity: This is the activity that gives users a choice to upload and share a file of their choice (PDF, DOCX, Image, Audio, Video) with the other members of the group selected. The user can first select a file using a chooser. Once the file is selected, the upload button is enabled. Once the file is uploaded, the share with Facebook button gets enabled and user is able to share the download URL of the uploaded file for other users (for PDF, DOCX and Audio files) using ShareLinkContent object of Facebook APIs, or is able to share the image/video directly on Facebook (using SharePhoto and ShareVideo objects of Facebook APIs). When the file is uploaded, it gets stored in Firebase Storage along the file path fixed for each group by the developer. The download URLs of the files uploaded are also stored in Firebase Realtime Database (at the parent node FilesSharedUrl). When the user clicks the share to Facebook button, the control flow is directed towards SharedFilesPostOnFB activity.
12. SharedFilesPostOnFB: This activity is used to perform a customized Facebook post for each of the different files being shared. The group members (who are friends of the user) are tagged along with the post. ShareDialog object is again used along with a couple of extra parameters. After the post has been made, the user can share another file by pressing the Continue button or can exit to the main screen, i.e., the control flow is directed towards UploadFileOntoStorageActivity or MainActivity.
13. ShareOutsideLinkUrlActivity: This activity is used to share an external link from third-party apps on FbShare. Once again, ShareDialog object is used for posting on Facebook and the control flow is directed towards MainActivity afterwards.
14. ViewAllMsgsInGrp: This activity is used to list the hyperlinks to all the files uploaded in the group. It is also displayed using FirebaseRecyclerAdapter as a RecyclerView and CardView.

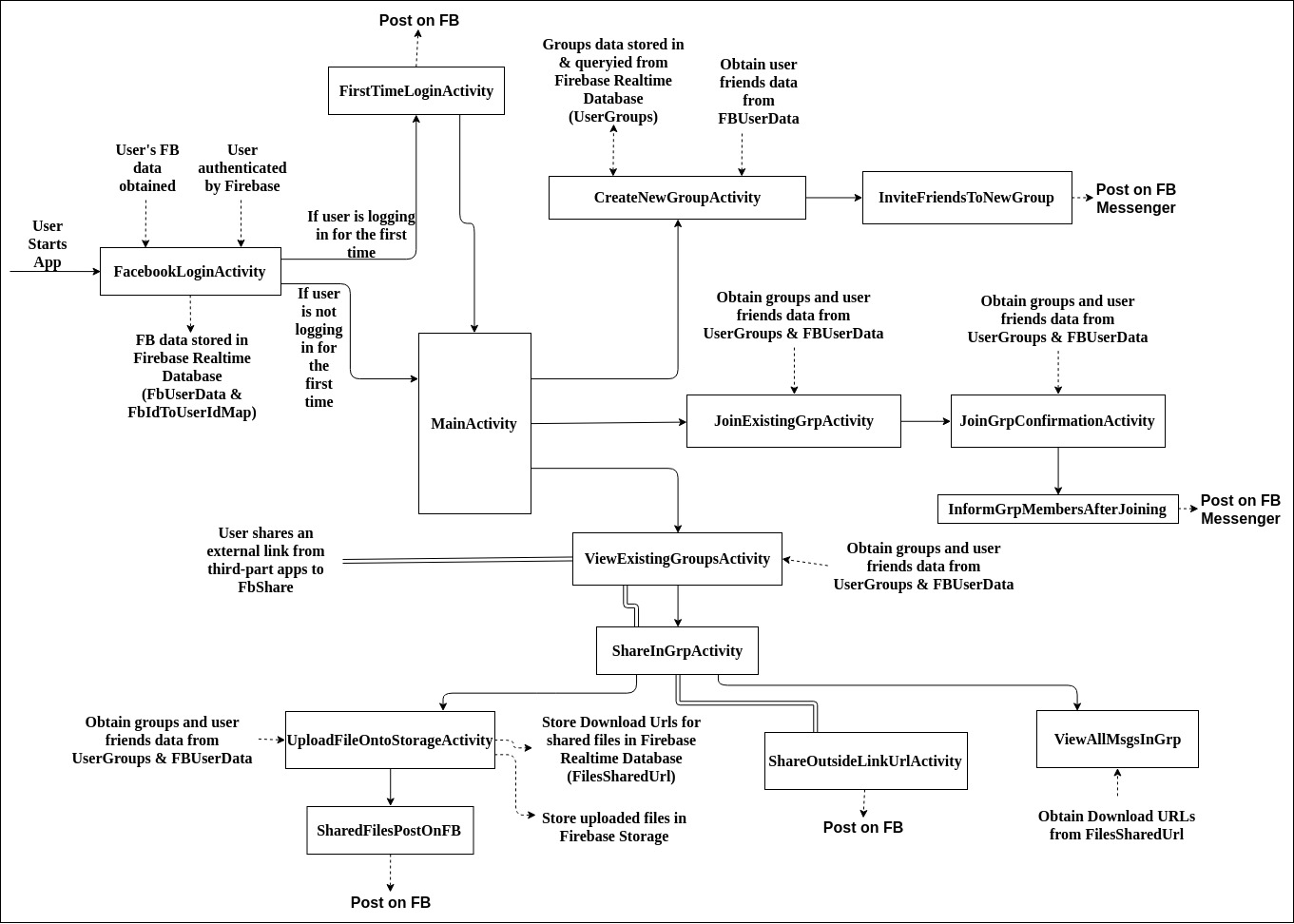
The following figure gives a flowchart depicting control flow in the android app: 

Fig. 2: Activity Control Flow Diagram

The following screenshots depict how data is stored in Firebase Realtime Database:

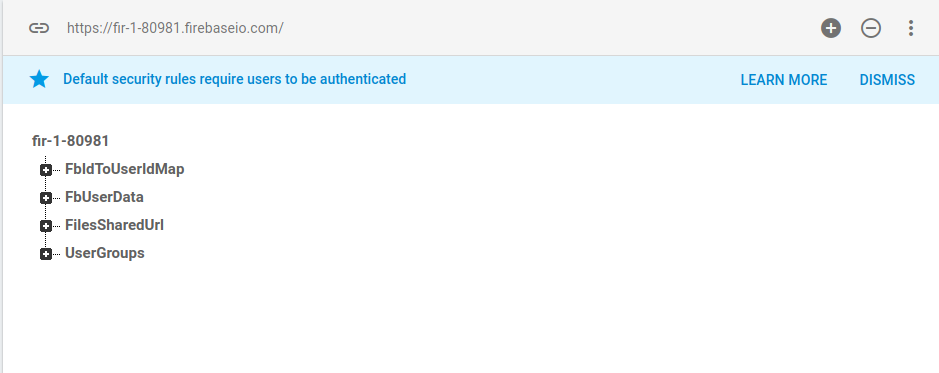


Fig. 3: Firebase Realtime Database Main parent nodes

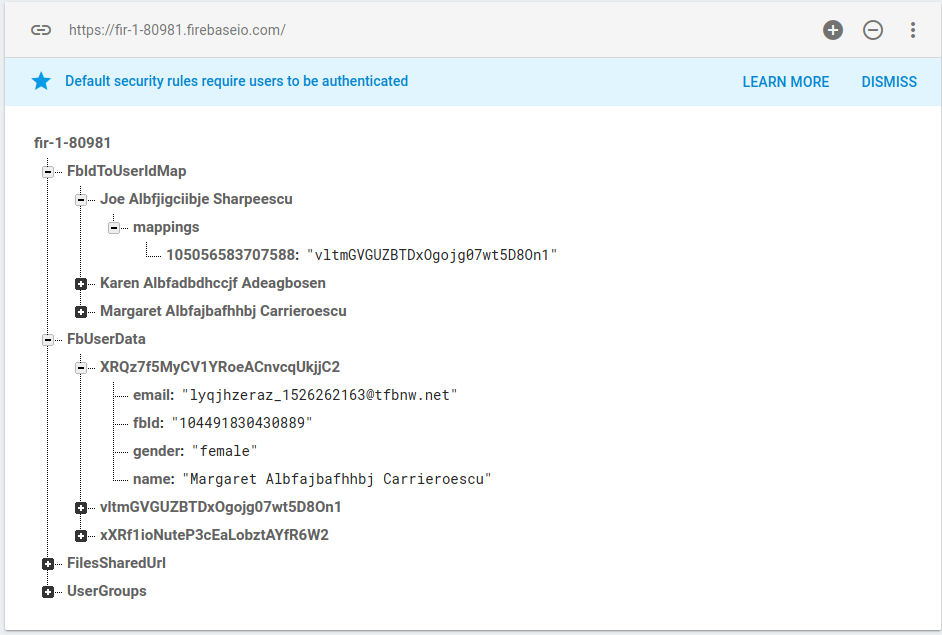


Fig. 4: Data stored in FbIdToUserMap and FbUserData nodes

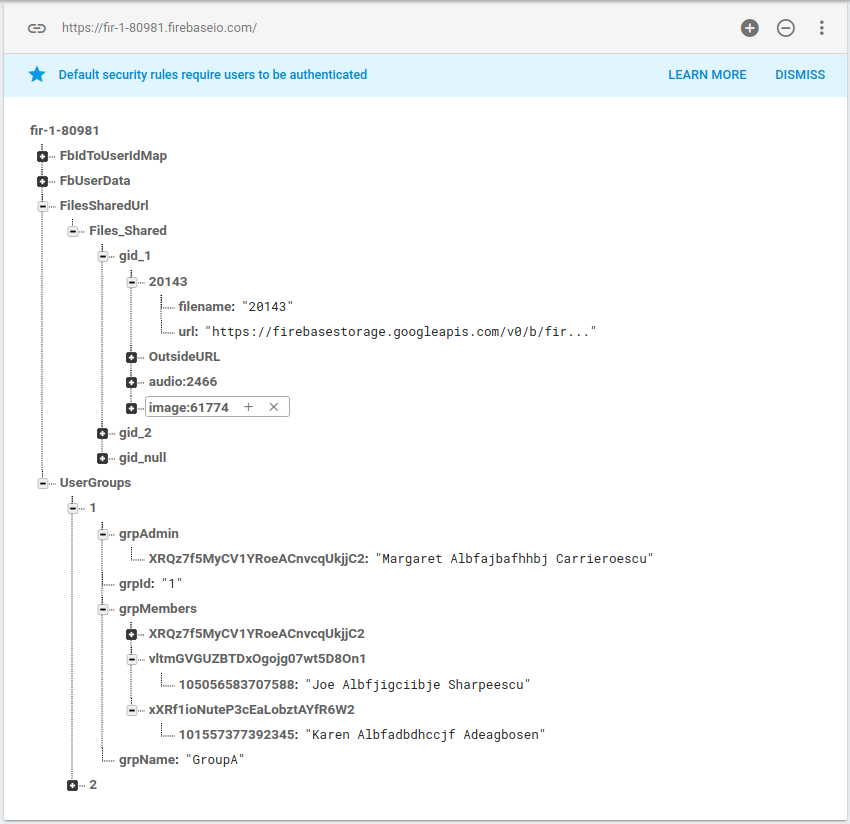


Fig. 5: Data stored in FilesSharedUrl and UserGroups nodes

The next screenshots depict how data is stored in Firebase Storage:

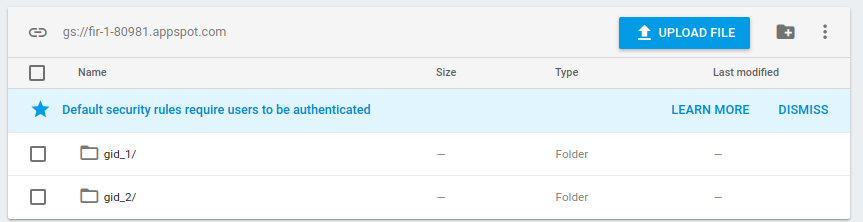


Fig. 6: Topmost layer of file structure(Each group has its own directory)

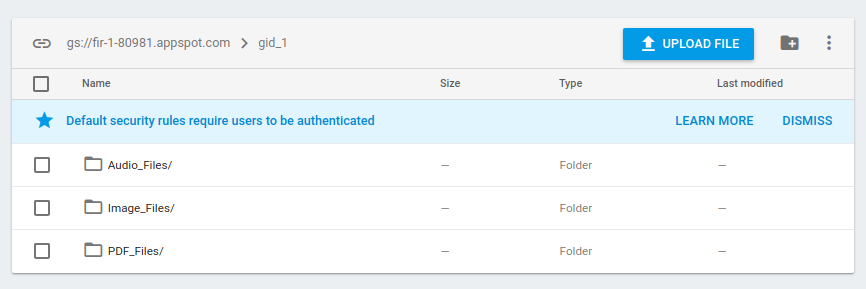


Fig. 7: File structure within first group (gid\_1); each different types of files have its own directory

**Basic Evaluations:**

1. Size of FbShare App on device: 6.02 MB
2. Maximum RAM usage of App recorded (during uploading an audio file of 142. MB and a video file of 582.92MB): 264 MB
3. Battery usage under high load (during uploading an audio file of 146.1 MB and a video file of 582.92MB): 3%
4. Timing tests: These tests were performed by three test users created on Facebook Developer site. The following is a screenshot of the users:

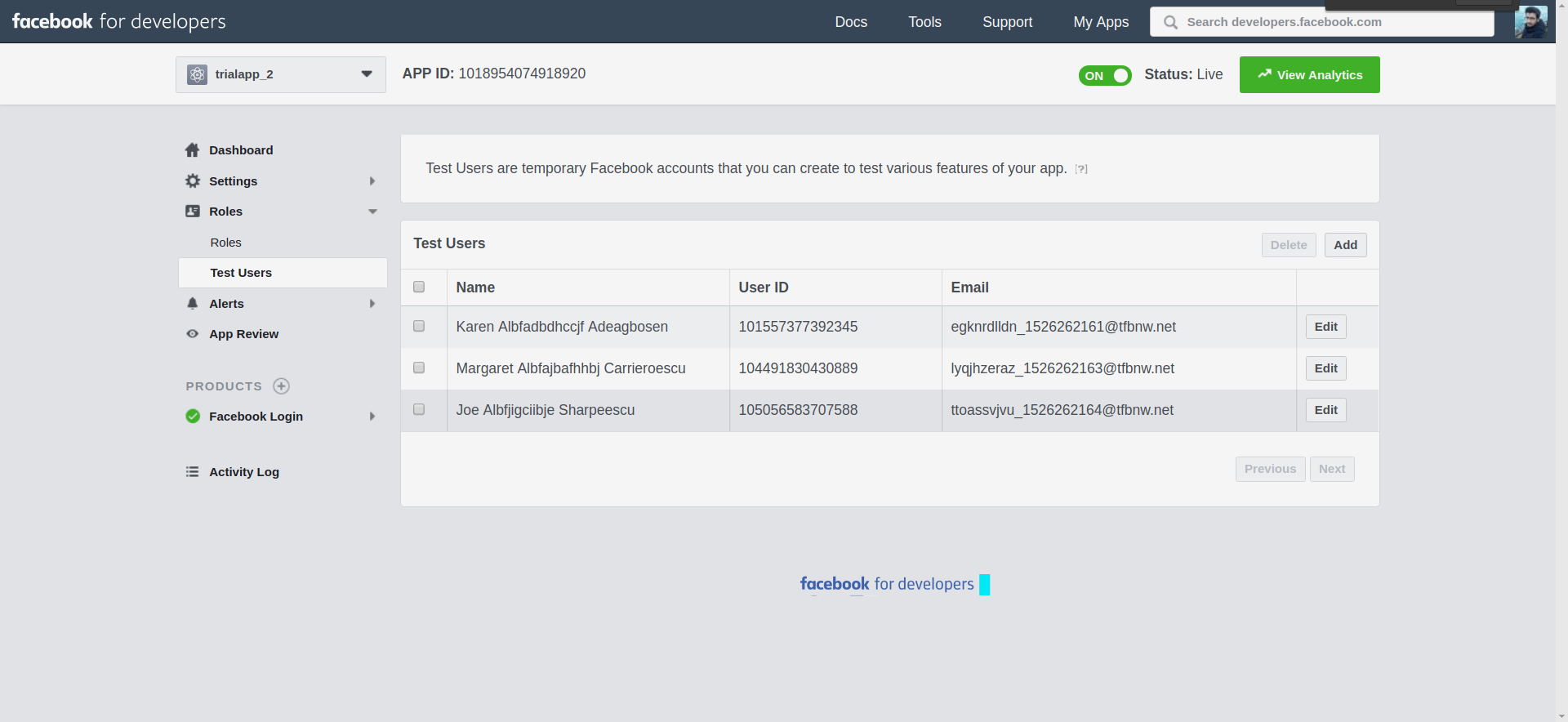


Fig. 8: Facebook Test users created and tested with

* 1. Average time taken for First-time user to navigate from Login screen to First-time Facebook post Activity - 3s
  2. Average time taken for user to navigate from Main screen to Create Group Facebook Messenger Post Activity - 4s
  3. Average time taken for user to navigate from Main screen to Join Group Facebook Messenger Post Activity - 6s
  4. Maximum time taken for user to navigate from Main screen to upload file(during upload of video file of 582.92 MB) - 3 minutes
  5. Average time taken for user to share an external link Facebook post: 4s

**Conclusions:**

The goal of the entire course project was to learn about android and cloud computing systems, which by building this app, was successfully accomplished. This is a really excellent way to gain hands-on practise and experience in building mobile cloud apps. The app FbShare was created and runs without errors and also performs the functionality it was supposed to achieve. Various techniques of interacting with cloud systems (Firebase Realtime Database and Firebase Storage) as well as with Facebook APIs (OpenGraph APIs) were learnt during the course of this project.