PROJECT 2 FINAL REPORT
EMBEDDING CLOUD SERVICES FOR ANDROID APPLICATION USING REST

TEAM 1:

-Ankita Kapratwar
(SJSU ID- 009413469)

-Anumeha Shah
(SJSU ID- 009423973)

-Swapna Kulkarni
(SJSU ID-009264905)

TABLE OF CONTENTS

INTRODUCTION

This project targets on developing an android application which will allow users to upload, list and play the video to cloud in a secure way. We have implemented REST web services to provide an android-cloud connectivity system turning an android device

into an extension of powerful cloud based services on a popular cloud computing platform Google App Engine. This will help the users of android devices to leverage the power of cloud platform and will provide scalability, accessibility and unlimited storage.

INTENDED USERS

The beauty of this project lies in the scope of the type of users who can use this application. This project has the following users/customer

1. Layman android phone user
2. Photographer/Video Maker
3. Any user who likes to take random videos

FUNCTIONALITY MATRIX

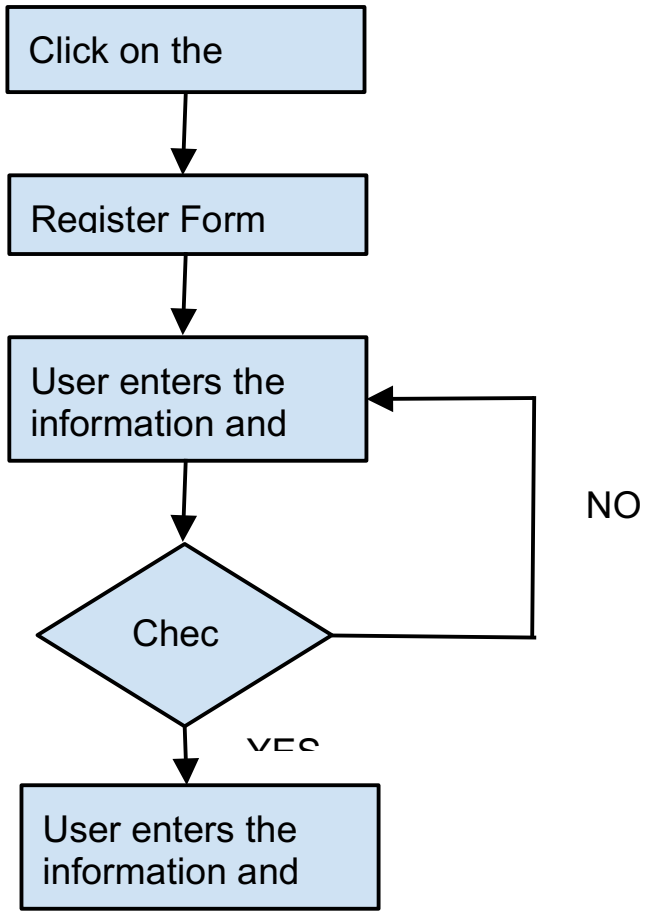
Feature	Description	Status
Restricted Access	Access to the data in cloud will be restricted by the userID and password	Done

Encryption	The data travels from from user to cloud in an encrypted form	Done
Storage Security	Secure hash check for passwords and passwords are stored in a hash form	Done
REST API Implementation	REST API's for connecting to the Google App Engine	Done
User Registration	Registers an new user for the application	Done
Authentication	Every user for the application will be authenticated	Done
Video Upload	The video successfully gets uploaded on the cloud	Done
List and play public videos	The user views the list of all the public videos and play them	Done
List and play private videos	The user views the list of his private videos and plays them	Done
Logout	User can log out from the application	Done
Blob store JAVA API	Store the video files as blob on Google App Engine	Done

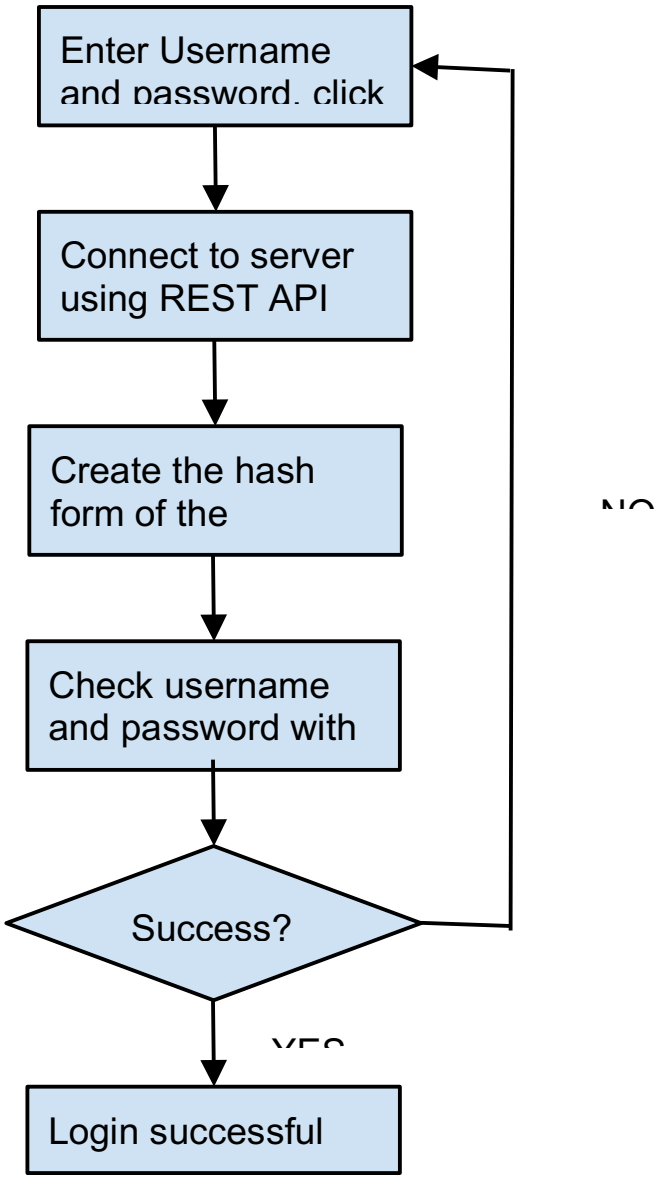
FINAL PROJECT DESIGN

- **Flow Diagrams**

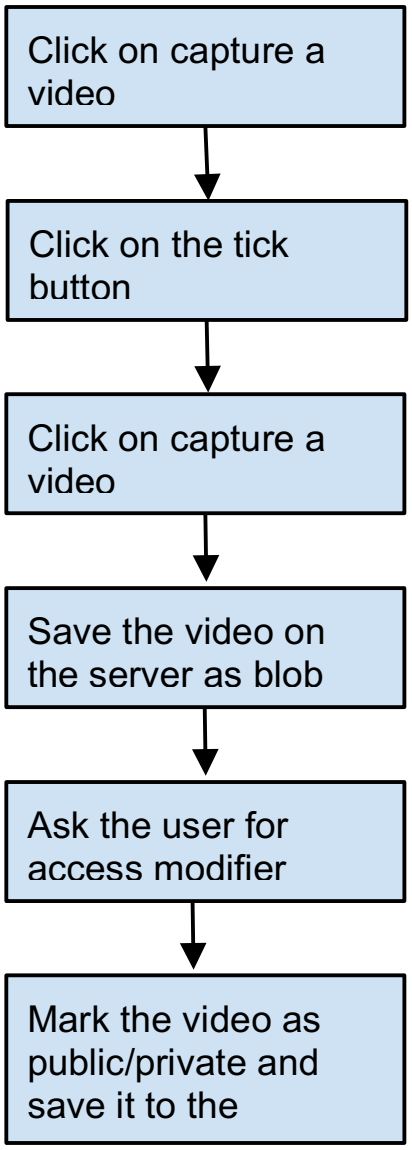
1) **Register a new user**


Click on the
Register Form
User enters the information and
NO
Chec
YES
User enters the information and

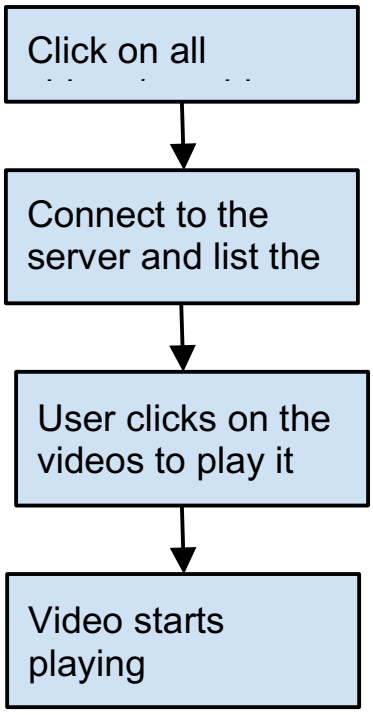
2) Login



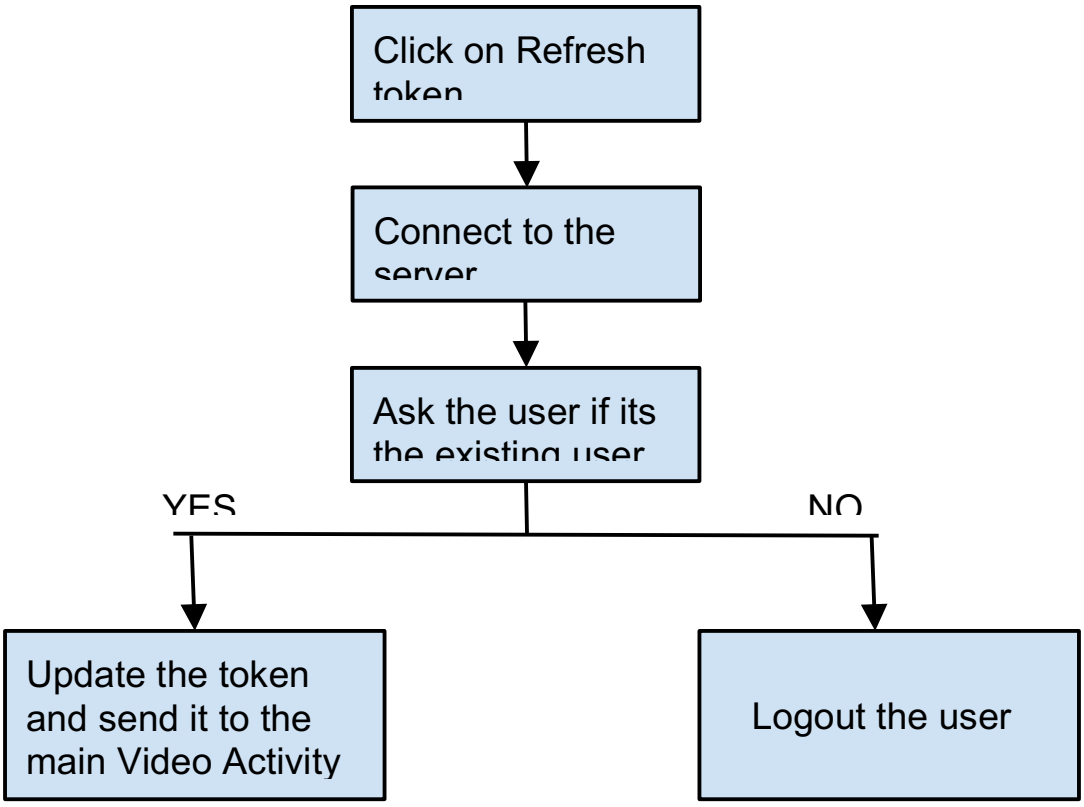
3) Upload a video


Click on capture a video
Click on the tick button
Click on capture a video
Save the video on the server as blob
Ask the user for access modifier
Mark the video as public/private and save it to the

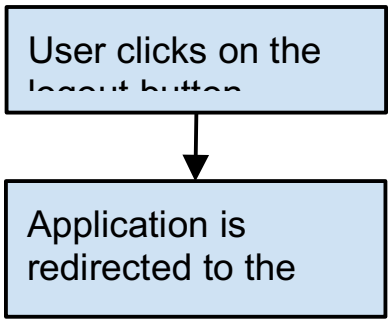
4) Play Video



4) Refresh token



5) Logout



MAJOR DESIGN DECISION

- **Cloud Platform**

There were mainly two platforms available for the backend in our application: Google App Engine and Amazon's EC2. The Google's App Engine is a Platform as a Service (PaaS) that lets you build and run applications on Google's infrastructure. App Engine applications are easy to build, easy to maintain, and easy to scale as your traffic and data storage needs change. With App Engine, there are no servers for you to maintain. Amazon Elastic Cloud Compute is an Amazon's web service which provide a cloud computing platform for a developers, with resources required for the application that are scalable for computing purposes to the developers. This service gives an ease to your application to configure the computing capacity both up and down. This is a pay as you go service provided by Amazon. The EC2 helps to make your application resistant to failures and prevents them from a predefined set of scenarios for failure.

For our project we decided to use the Google's App Engine for designing the backend of our application. This decision was driven due to various factors:

a. The ease with which the SSL can be enabled in Google App Engine compared to Amazon's EC2 is more.
b. The database designing in Google App Engine is less complex due to the entity value pair storage method
c. More familiarity with Google App Engine.

- **Blobstore**

Blobstore JAVA API has been used to store video object as blob and serve as blob. Google App Engine use Blobstore API to serve large files such as videos and images. A video file can be uploaded to server from Android application in two steps.

Create an upload URL : blobstoreService.createUploadUrl method create an upload URL and return back to Android client.

Upload Video : Video file is uploaded to the returned URL in step one.The Blobstore API create blob and also created meta data for blob and store to Datastore. The API then returns blobkey which is used to serve the blob.

FINAL PROJECT IMPLEMENTATION

Project Specifications

1. Operating System: Windows/ MacOS
2. Platform: Google Cloud Platform
3. Development Environment: Eclipse IDE, Android SDK
4. Framework for REST Api's: Servlet + Google App Engine APIs
5. Language Used: Java
6. BlobStore Service of Google App Engine to store video objects

API Details- The request response details are as follows

1. **Login Request Data : https://1-dot-assignment2-905.appspot.com/rest/user**

   {"emailId":"value", "passWord":value", "requestType: "login"}

   Login Response Data

   {"responseType" : "login","success" : "1","error" : "0","userName" : "anumeha shah","userId" : "1234567","accessToken" : "234567","errorMsg" : ""}

   If user is able to login successfull then
   - error should be set to 0
   - success should be set to 1
   - errorMsg should be set to ""
   - successMsg should be set to some msg such as User is able to login successfully

   If user is not able to login successful and there is some error such as user does not exists etc
   - error sets to 1

- success sets to 0
- errorMsg sets to “error message value”
- successMsg sets to “”;

2. Register Request Data https://1-dot-assignment2-905.appspot.com/rest/user

{"requestType":"register","emailId":"emailId","lname":"zxc","passWord":"passWord","fname":"zx”}

Request Data fields details
fname : user’s first name
lname : user’s last name
emailId: user’s email id
passWord: user’s password
requestType : register

Register Response Data

{"responseType":"register","success":"1","error":"0","successMsg":"Registered successfully Please Log in to access the app","errorMsg":""}

If user is able to register successfully

- set success = 1
- set error = 0
- set successMsg = “some msg”
- set errorMsg = “”;

If user is not able to register successfully

- set success = 0
- set error = 1
- set successMsg = “”
- set errorMsg = “some value.”;

3. createURL request data for video upload

{"userName":"abcde","emailId":"abcde@gmail.com","userId":"abcde","accessToken":"0.8435933116974902","requestType":"upload"}

createURL response data

{"responseType":"upload","name":"test1.mp4","tmp_name":"https://1-dot-assignment2-905.appspot.com/_ah/upload/AMmfu6bpqMYke2HmvuVD3cDnGC6VBoVTMQs8wCa0HSCEoql-s5J9BN6MvFT7RQa1JifDIbH_p4_wnp-kufVFE3CMIF13bK2RvqbUvMfIPk1U16mdXDN7LVs7E8uo-aEcoL1JwX4RSuJuDqIzQIEJiGBjAwRFtsATyA/ALBNUaYAAAAAVIje-r5Yh6P0E9u1uJ-DS0iPVrP3eRdL/","success":1,"error":0,"URL":"https://1-dot-assignment2-905.appspot.com/_ah/upload/AMmfu6bpqMYke2HmvuVD3cDnGC6VBoVTMQs8wCa0HSCEoql-s5J9BN6MvFT7RQa1JifDIbH_p4_wnp-kufVFE3CMIF13bK2RvqbUvMfIPk1U16mdXDN7LVs7E8uo-aEcoL1JwX4RSuJuDqIzQIEJiGBjAwRFtsATyA/ALBNUaYAAAAAVIje-r5Yh6P0E9u1uJ-DS0iPVrP3eRdL/"}

4. **uploadVideo Request**
   Upload video file using content-type multipart/form-data

   uploadVideo Response
   {"responseType":"upload","name":"test1.mp4","success":1,"error":0,"successMsg":"Hello","bk":{"blobKey":"AMIfv96-NonW7TfgqJhX6GlZe12c9V-Z7GHxf6Bv7mp-aKjr3_YLdcbP6R2r--FraJFtunKgr_shUekBW4Iyru2D8Ft2145gibilEon7RHUQHhByA3T3YkmEmWJq-UsIXwe6bDQcRAYkjdHlUiVimHVwWGIv6Axkw2kHhNEz_Kbo6ZbHaU5V5ks"}}

5. **Video permission change Request Data**

   {"userName" : "value","accessToken" : "value","userId" : "value","requestType" : "permission", “private” : “value”, “emailId” : “emailid” }

   the value of permission can be public or private

   Video permission change Response Data

   {"responseType":"permissions","success":"1","error":"0","successMsg":"Permission of the video file changed successfully","errorMsg":""}

If permission changed successfully then mark the video as public or private accordingly

- set success = 1
- set error = 0
- set successMsg = “some msg”
- set errorMsg = “”;

If permission does not successfully give following response

- set success = 0
- set error = 1
- set successMsg = “”
- set errorMsg = “some value.”;

6. All Video Request Data

{“userName”:”value”, “userId”:value”, “accessToken: “value”, “requestType: “getAllVideos”}

All Video Response Data

{"videos":
[
{""blobKey":"AMIfv96-NonW7TfgqJhX6GlZe12c9V-Z7GHxf6Bv7mp-aKjr3_YLdcbP6R2r--FraJFtunKgr_shUekBW4Iyru2D8Ft2145gibilEon7RHUQHhByA3T3YkmEmWJq-UsIXwe6bDQcRAYkjdHlUiVimHVwWGIv6Axkw2kHhNEz_Kbo6ZbHaU5V5ks""},
{""blobKey":"AMIfv96-NonW7TfgqJhX6GlZe12c9V-Z7GHxf6Bv7mp-aKjr3_YLdcbP6R2r--FraJFtunKgr_shUekBW4Iyru2D8Ft2145gibilEon7RHUQHhByA3T3YkmEmWJq-UsIXwe6bDQcRAYkjdHlUiVimHVwWGIv6Axkw2kHhNEz_Kbo6ZbHaU5V5ks""},
{""blobKey":"AMIfv96-NonW7TfgqJhX6GlZe12c9V-Z7GHxf6Bv7mp-aKjr3_YLdcbP6R2r--FraJFtunKgr_shUekBW4Iyru2D8Ft2145gibilEon7RHUQHhByA3T3YkmEmWJq-UsIXwe6bDQcRAYkjdHlUiVimHVwWGIv6Axkw2kHhNEz_Kbo6ZbHaU5V5ks""},
],
"responseType":"getAllVideos","success":"1","error":"0","successMsg":"Downloaded successfully","errorMsg":"Error in downloading"}

7. My Video Request Data

{“userName”:”value”, “userId”:value”, “accessToken: “value”, “requestType: “getMyVideos”}

My Video Response Data

{"videos":
[
{""blobKey":"AMIfv96-NonW7TfgqJhX6GlZe12c9V-Z7GHxf6Bv7mp-aKjr3_YLdcbP6R2r--FraJFtunKgr_shUekBW4Iyru2D8Ft2145gibilEon7RHUQHhByA3T3YkmEmWJq-UsIXwe6bDQcRAYkjdHlUiVimHVwWGIv6Axkw2kHhNEz_Kbo6ZbHaU5V5ks""},
{""blobKey":"AMIfv96-NonW7TfgqJhX6GlZe12c9V-Z7GHxf6Bv7mp-aKjr3_YLdcbP6R2r--FraJFtunKgr_shUekBW4Iyru2D8Ft2145gibilEon7RHUQHhByA3T3YkmEmWJq-UsIXwe6bDQcRAYkjdHlUiVimHVwWGIv6Axkw2kHhNEz_Kbo6ZbHaU5V5ks""},
{""blobKey":"AMIfv96-NonW7TfgqJhX6GlZe12c9V-Z7GHxf6Bv7mp-aKjr3_YLdcbP6R2r--FraJFtunKgr_shUekBW4Iyru2D8Ft2145gibilEon7RHUQHhByA3T3YkmEmWJq-UsIXwe6bDQcRAYkjdHlUiVimHVwWGIv6Axkw2kHhNEz_Kbo6ZbHaU5V5ks""},
],
"responseType":"getMyVideos","success":"1","error":"0","successMsg":"Downloaded successfully","errorMsg":"Error in downloading"}

- When user clicks on All Video or My Video.
- It fetches blobKey from the server
- When user clicks on any blobKey, It fetches video data from server
- Stores video data to byte array
- Converts byte array to video file and store temporarily on sd card
- Serves the video from temporary file
- Delete the file once user goes back from play video page to video activity page.

8. **Refresh Token Request Data**

   {“userName”:”value”, “userId”:value”, “accessToken: “value”, “requestType: “refresh”, “emailId” : “emailid”}

   Refresh Token Response Data

   {"responseType":"refresh","success":"1","error":"0","successMsg":"Token has been refreshed successfully","errorMsg":"","userName":"anumeha shah","userId":"1234567","accessToken":"234567"}

9. **Log Out Request Data**

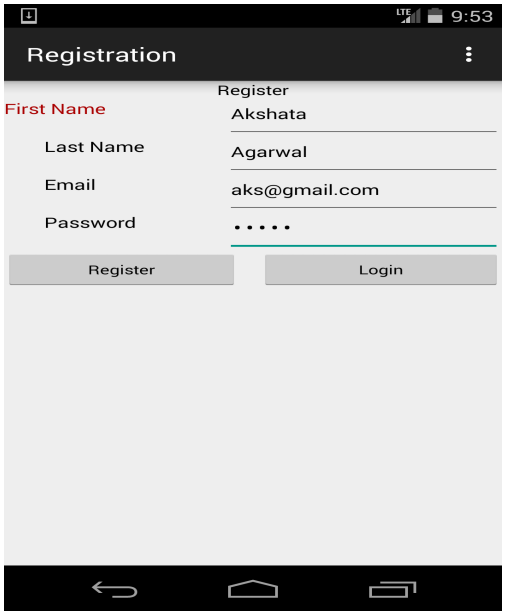
{“userName”:”value”, “userId”:value”, “accessToken: “value”, “requestType: “logout”}

Log Out Response Data

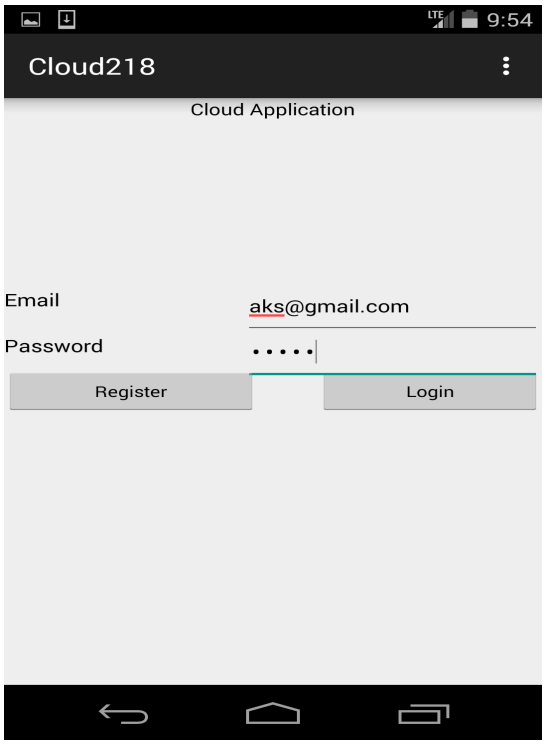
{"responseType":"logout","success":"1","error":"0","successMsg":"You have been logout from the app successfully","errorMsg":""}

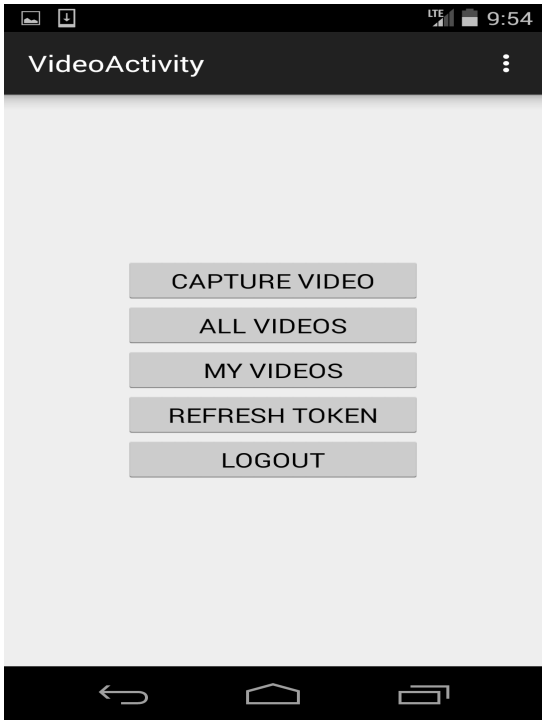
Screenshots

1. Register

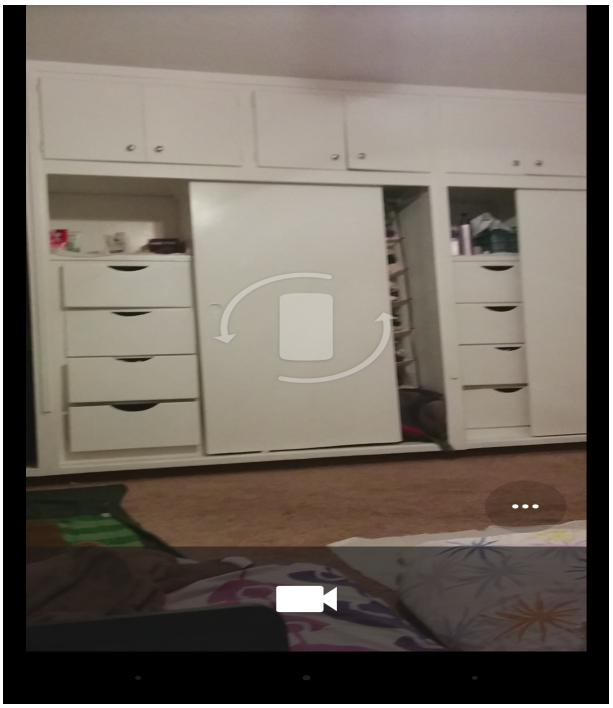


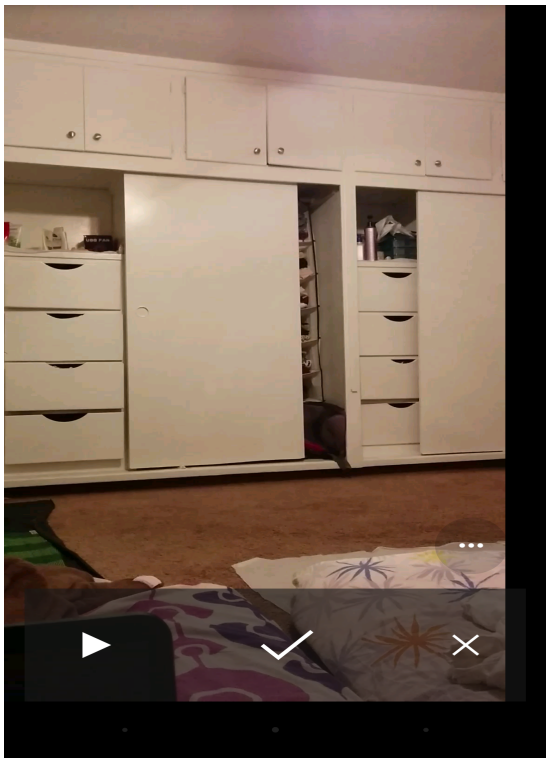
2. Login



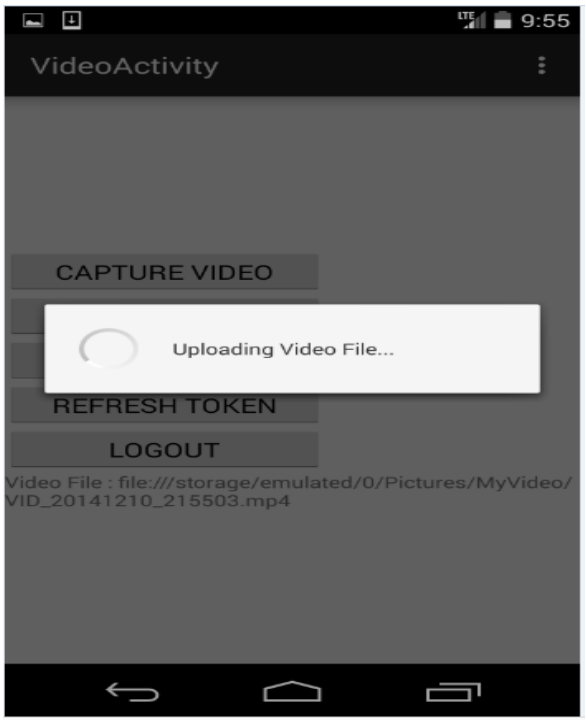


3. Capture Video

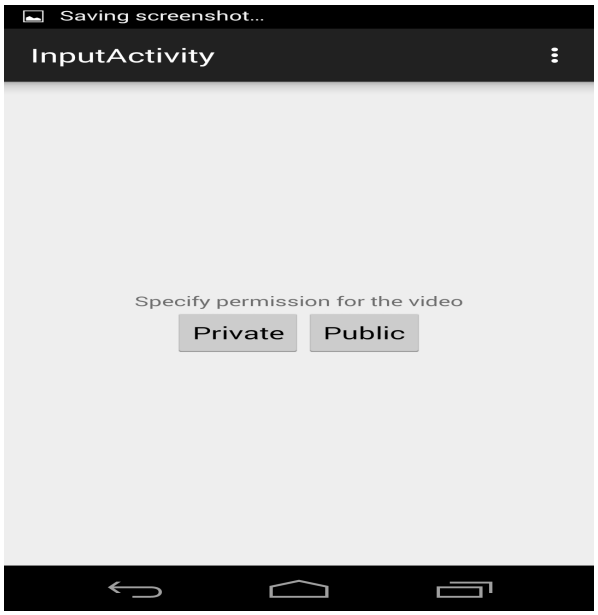


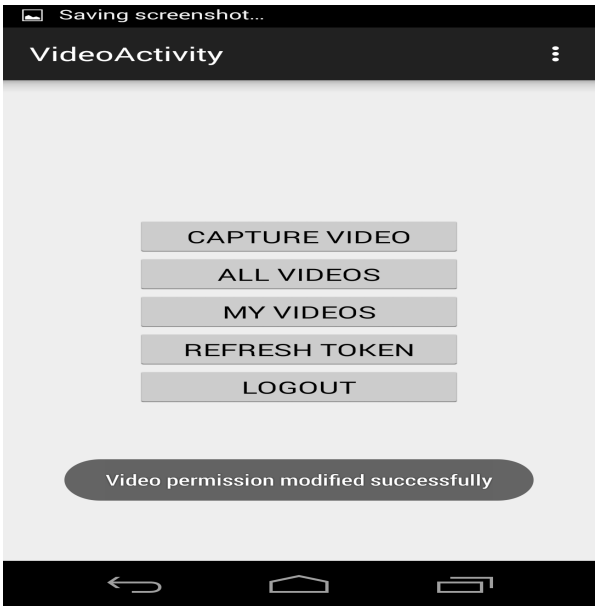


4. Uploading Video

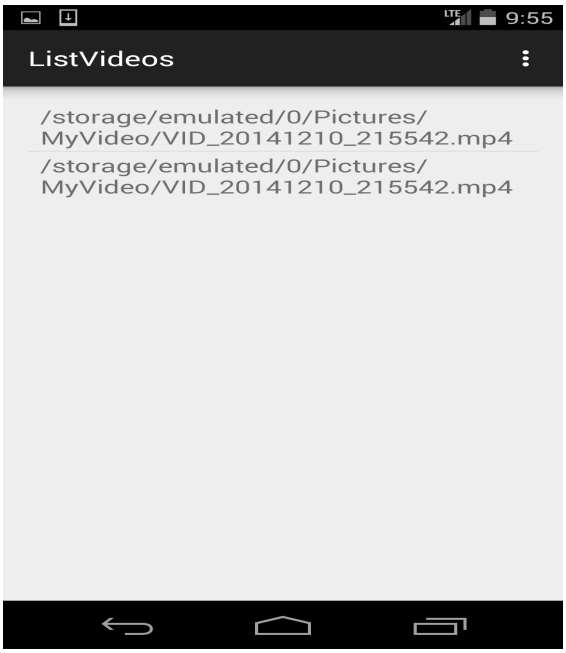


5. Access Permissions

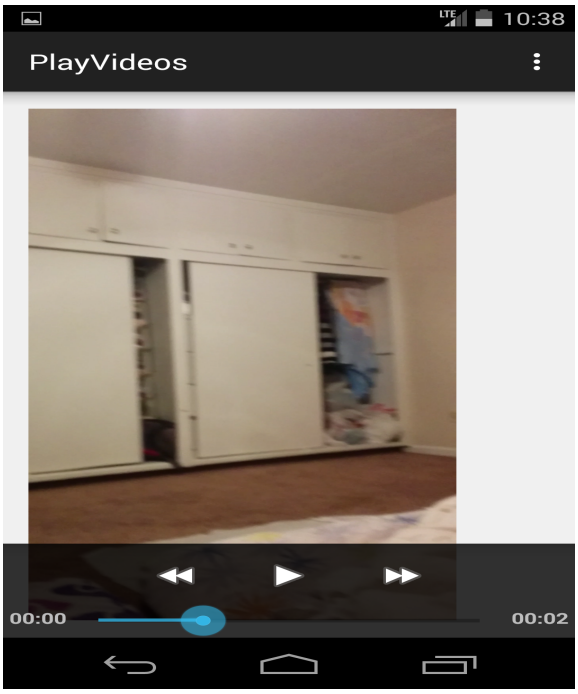




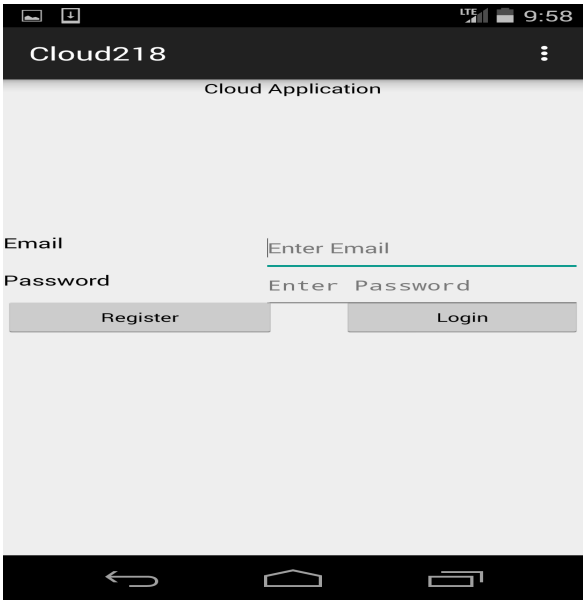
6. Listing Videos



7. Playing Videos



8. Logout



HOW OUR APPLICATION IS DIFFERENT FROM OTHERS

1. REST APIs: We have implemented REST API's to connect to cloud

2. Access Metrics: Access to other user's data is based on access metrics (public, Private) .The user is able to mark his data as private or public. If the data is marked as public then it can be seen by other users as well.
3.Very Secure: We have implemented security features such as encryption using ssl , authentication, secure hashing and token refreshing enablement.

TASKS FOR EACH MEMBER

Team Member	Task
Ankita Kapratwar	1. Listing of public and private videos for a user 2. Feature to play a video when a user clicks on the video url from list. 3. Connecting to the backend for the all videos and my videos functionality 4. Testing the application for video capture on the android device 5. Project report and presentation 6. Final project integration
Anumeha Shah	1. Designing the Login and Registration Functionality 2. Designing the capture and upload video functionality and public and private permission functionality 3. Refreshing access tokens 4. Logout functionality of application

	5. Client server communication 6. Project report and presentation. 7. Final project integration
Swapna Kulkarni	1. Build REST APIs 2. Application backend design 3. Client Server communication 4. Project report and presentation 5. Final Project Integration 6. Blobstore API implementation

PROJECT MILESTONES

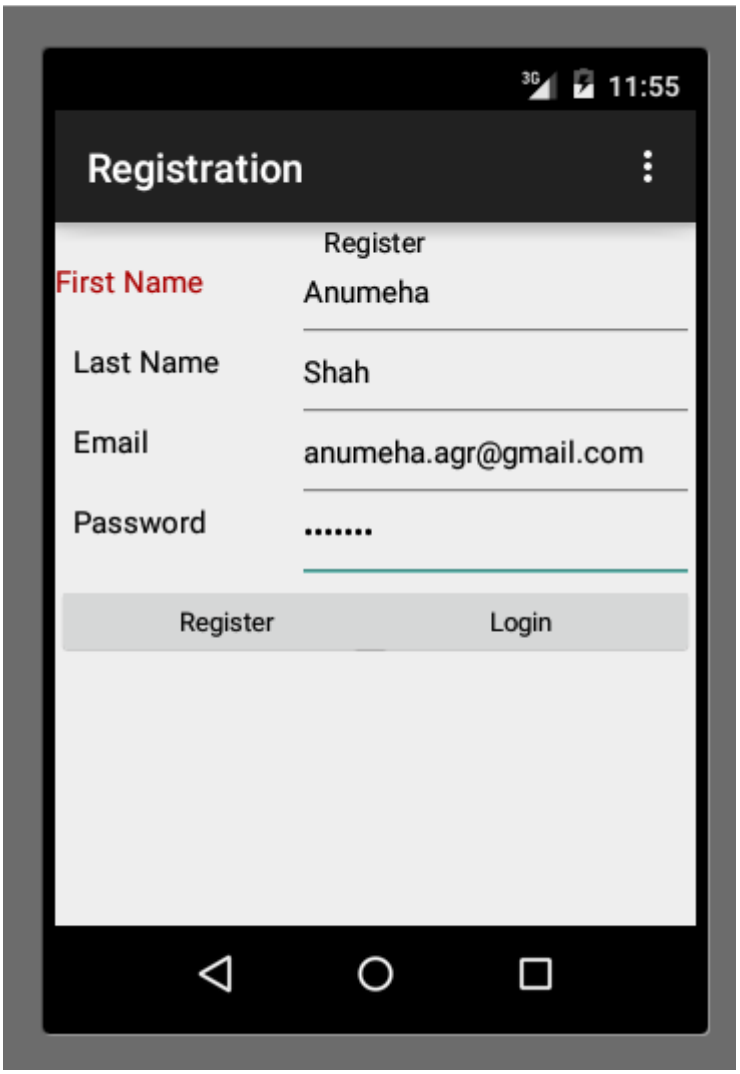
Date	Task
10/6	Preparing project plan.
10/8	Software/ plugins/SDK installation.
10/14	Frontend for Login and Registration Activity and implementation of REST Apis
10/17	Midterm Report
11/1	Implementation of login and registration activity. Testing it on a dummy backend server in php.
11/15	Implementation of capture video activity and testing it on real emulator and android device
11/22	Implementation of upload video activity testing it on android device with the backend server
11/29	Implementation of video list and playing

	activity and testing on real device with backend connectivity
11/1	Code integration for above functionalities
11/3	Logout functionality and refreshing access tokens
11/7	Project Integration, final testing, report and presentation submission

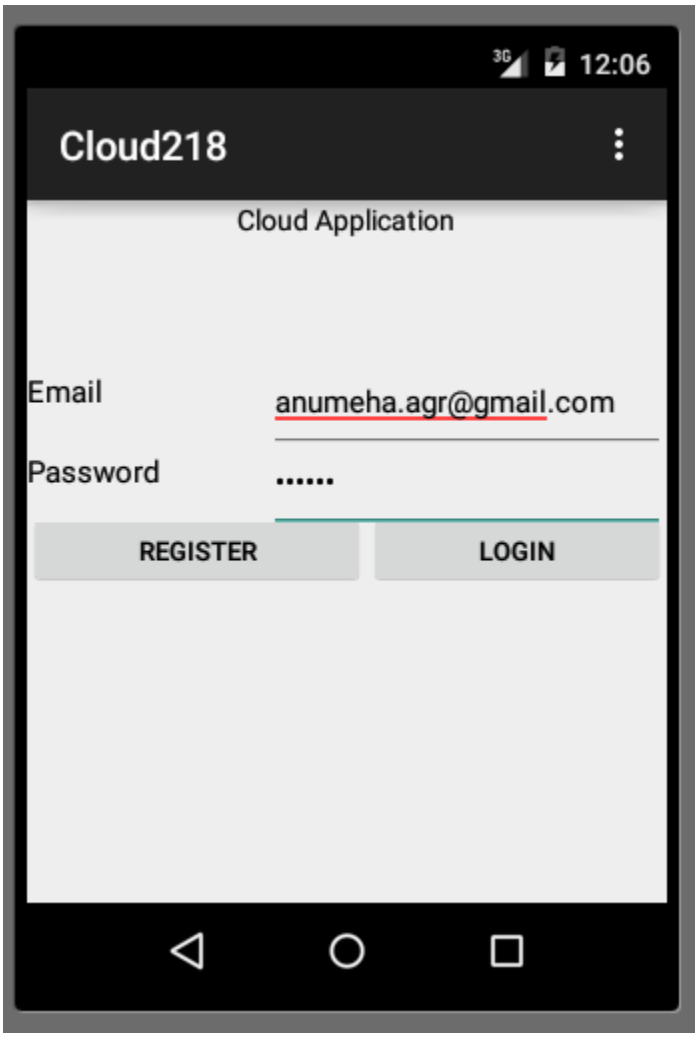
TEST PLAN EXECUTION

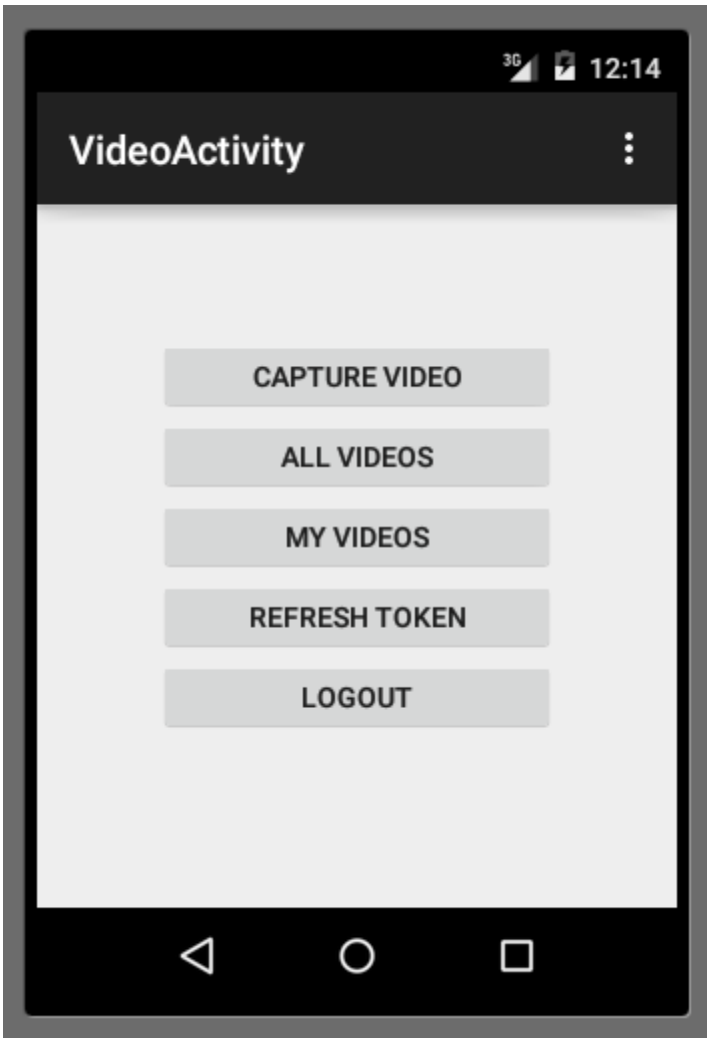
We initially tested the android frontend on our local machine with a dummy server. The screenshots are as follows:

1) REGISTER

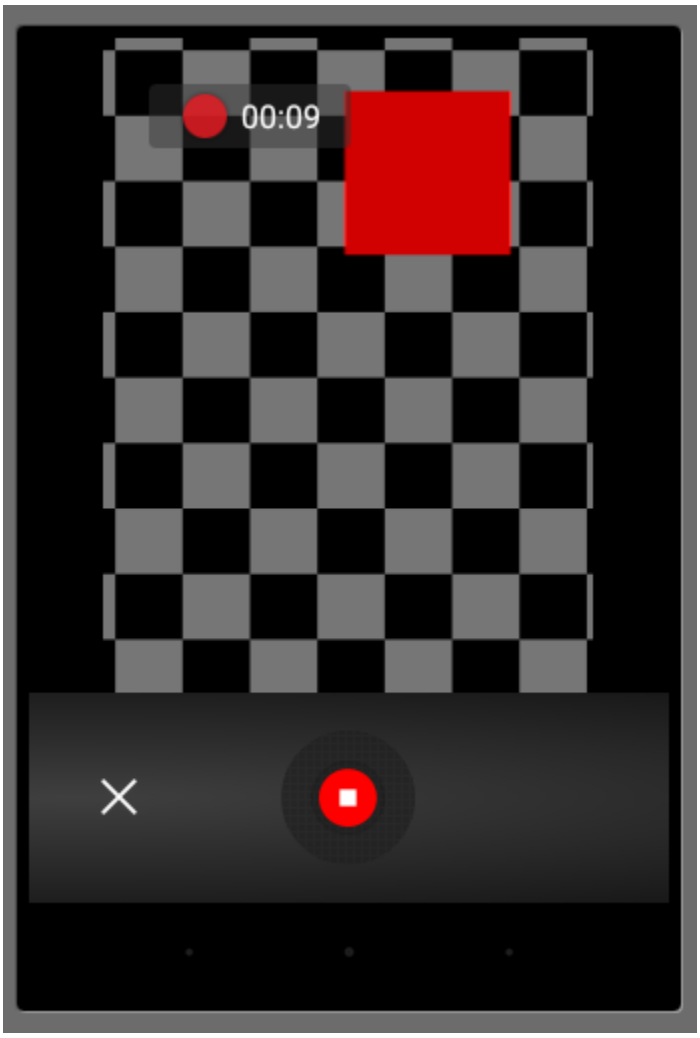

11:55
Registration
Register
First Name
Anumeha
Last Name
Shah
Email
anumeha.agr@gmail.com
Password
.....
Register
Login

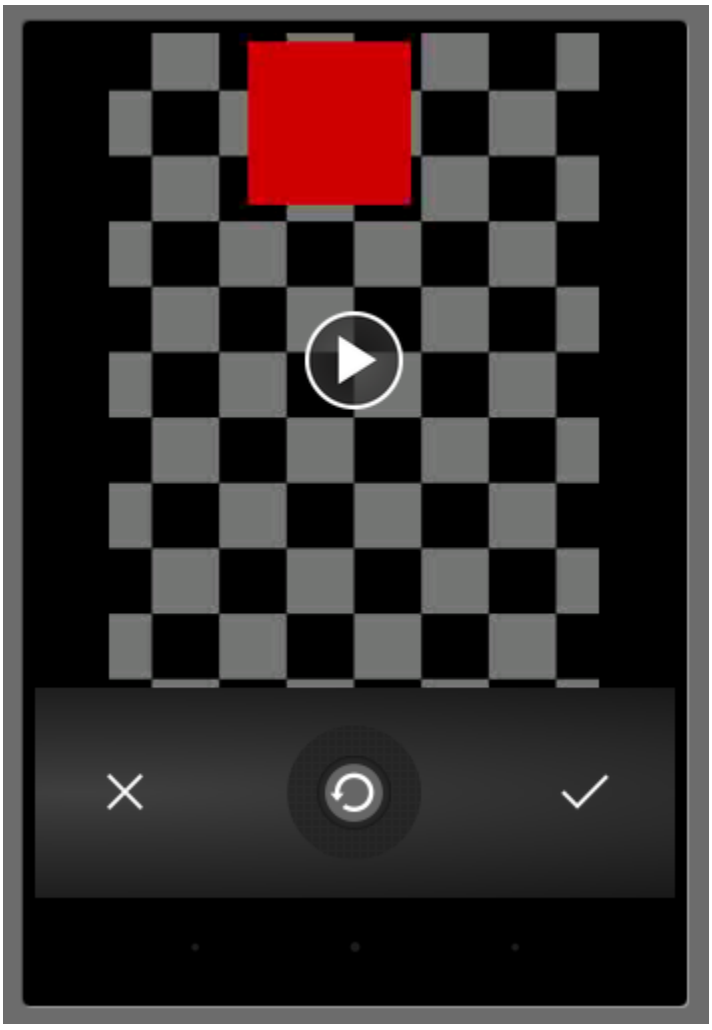
2) LOGIN


12:06
Cloud218
Cloud Application
Email
anumeha.agr@gmail.com
Password
.....
REGISTER
LOGIN

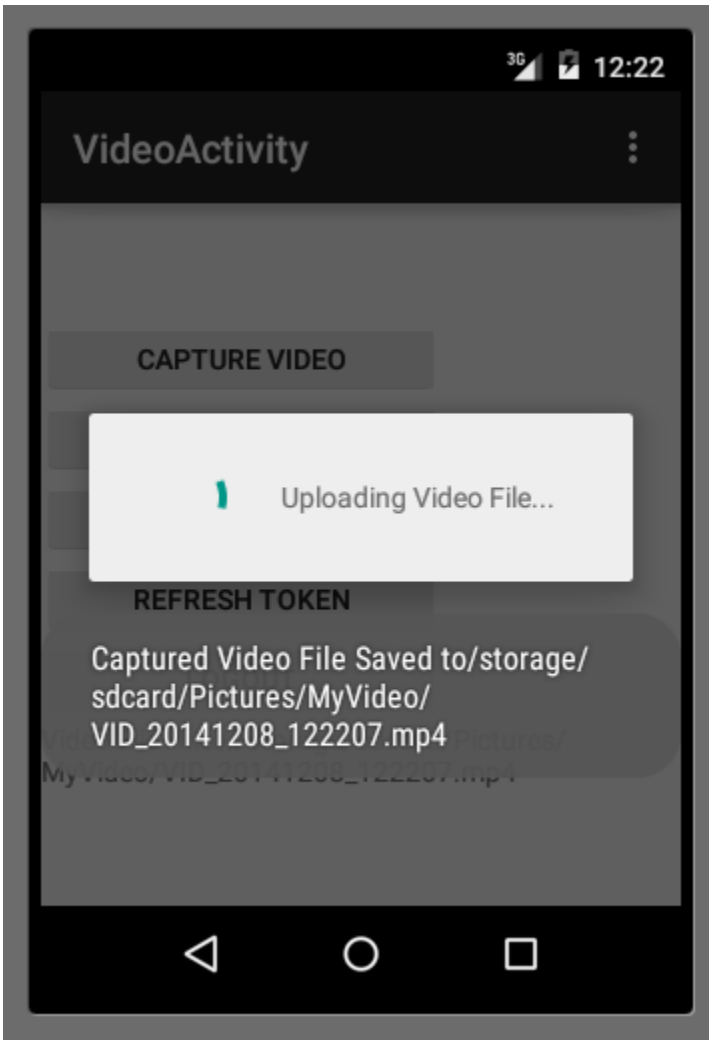


3) CAPTURE VIDEOS

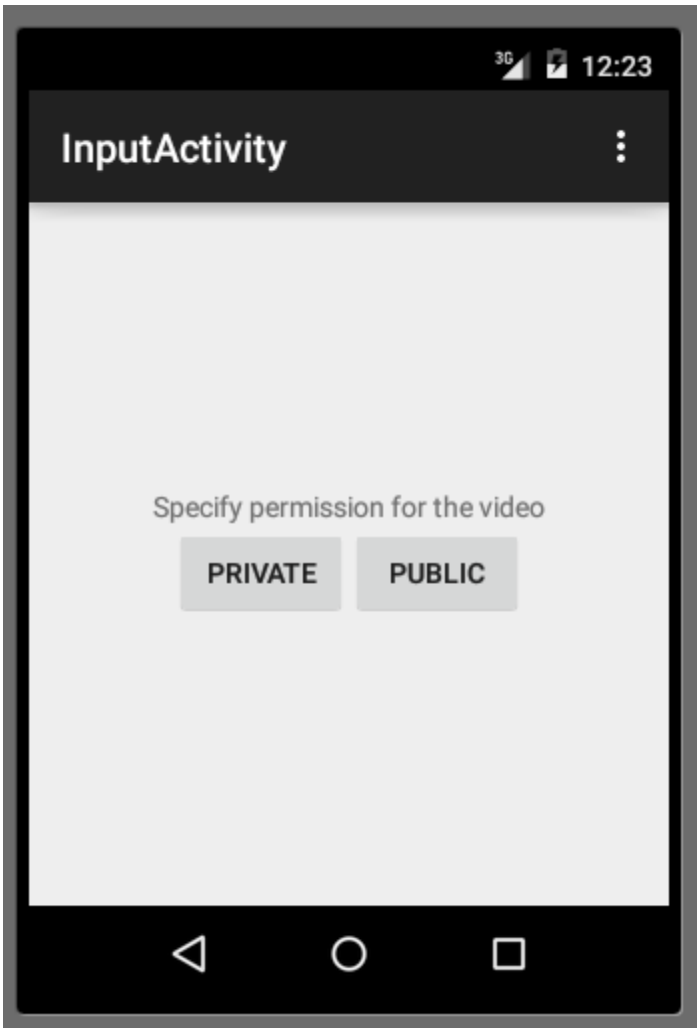

00:09



3) UPLOAD VIDEO

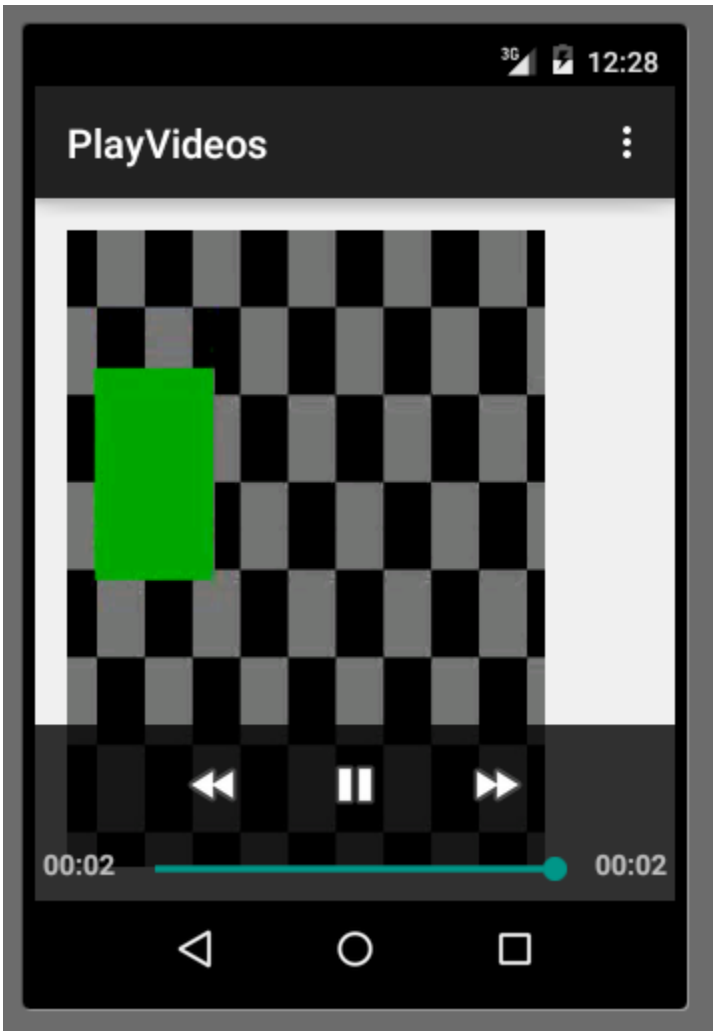


4) ACCESS PERMISSION


12:23
InputActivity
Specify permission for the video
PRIVATE
PUBLIC

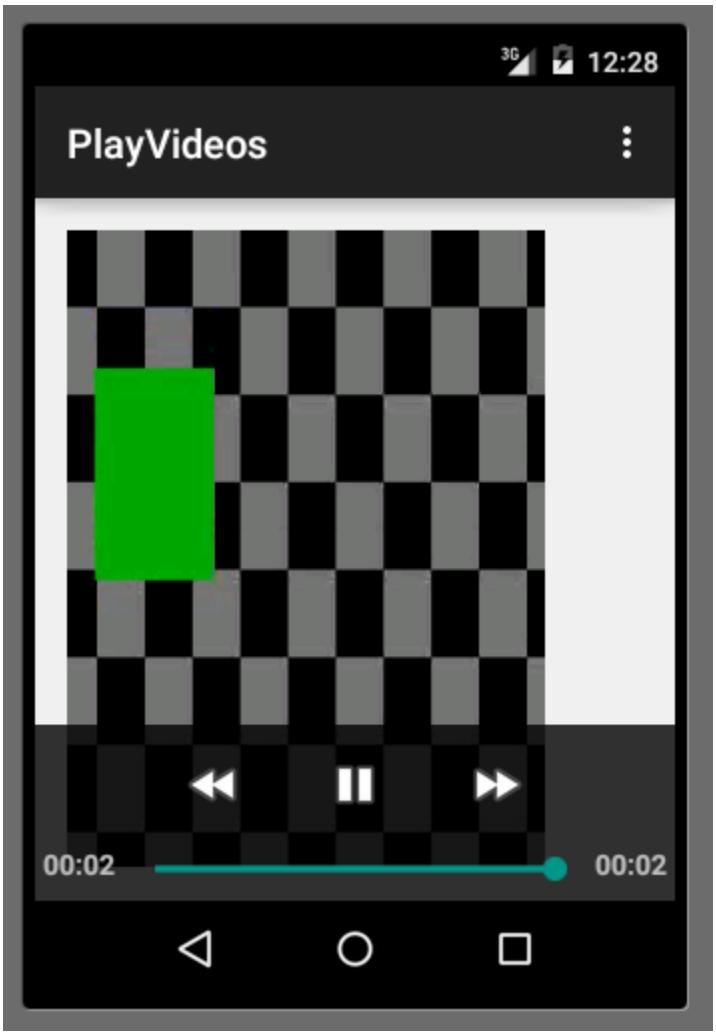
5) ALL VIDEOS


12:26
ListVideos
http://10.0.2.2/videouploads/
VID_20141202_163923.mp4
http://10.0.2.2/videouploads/
VID_20141202_175814.mp4
http://10.0.2.2/videouploads/
VID_20141202_235207.mp4
http://10.0.2.2/videouploads/
VID_20141203_180616.mp4

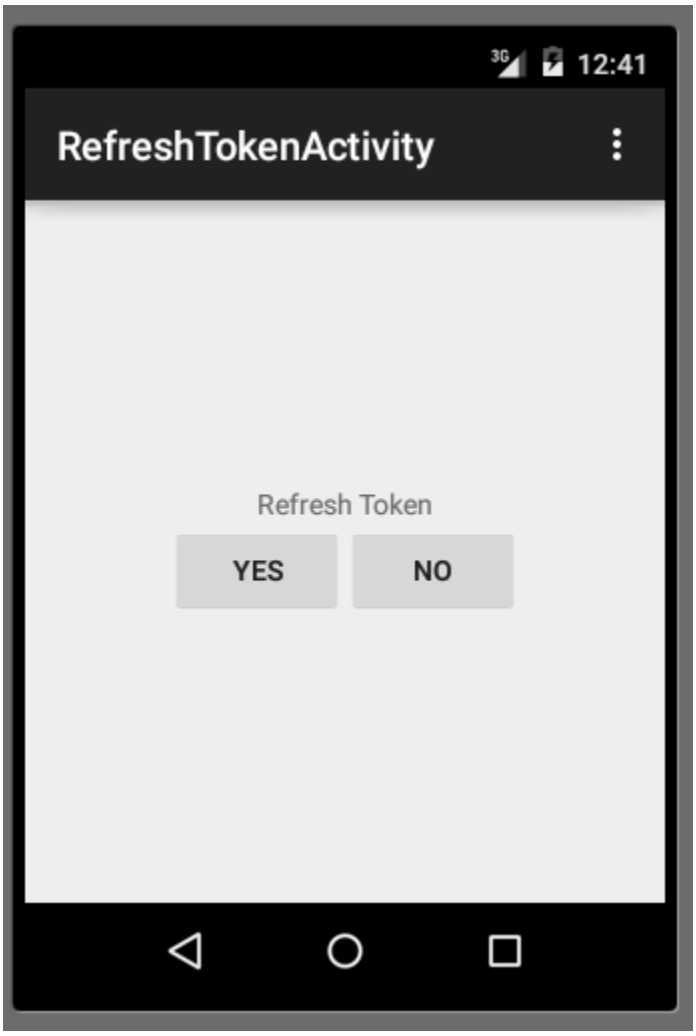


6) MY VIDEOS

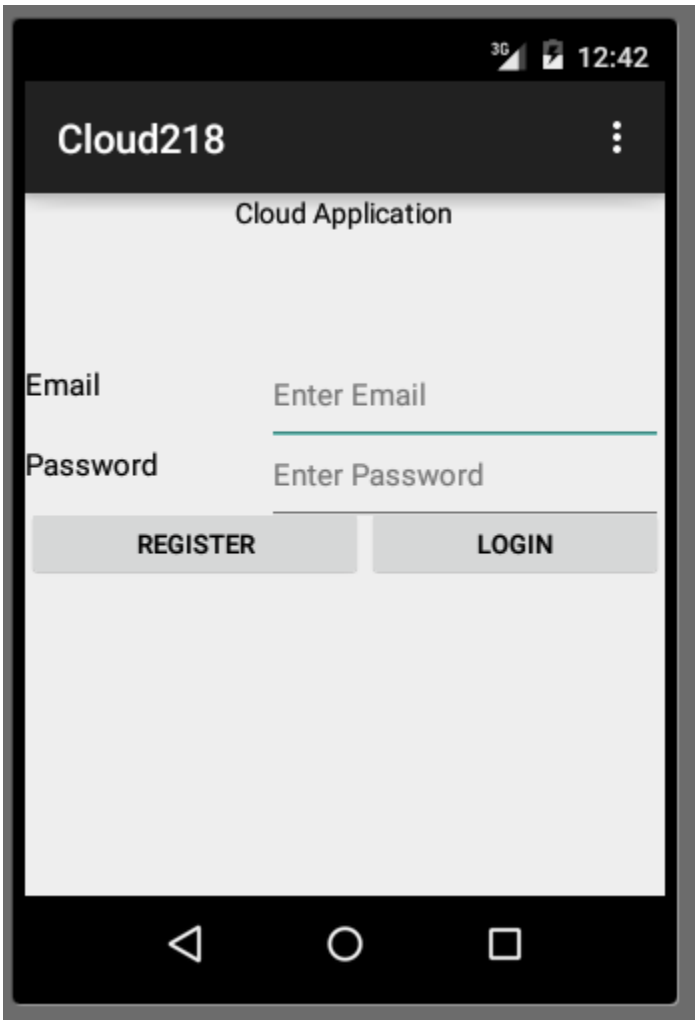

12:30
ListVideos
http://10.0.2.2/videouploads/
VID_20141202_235207.mp4
http://10.0.2.2/videouploads/
VID_20141203_180616.mp4



7) REFRESH TOKEN

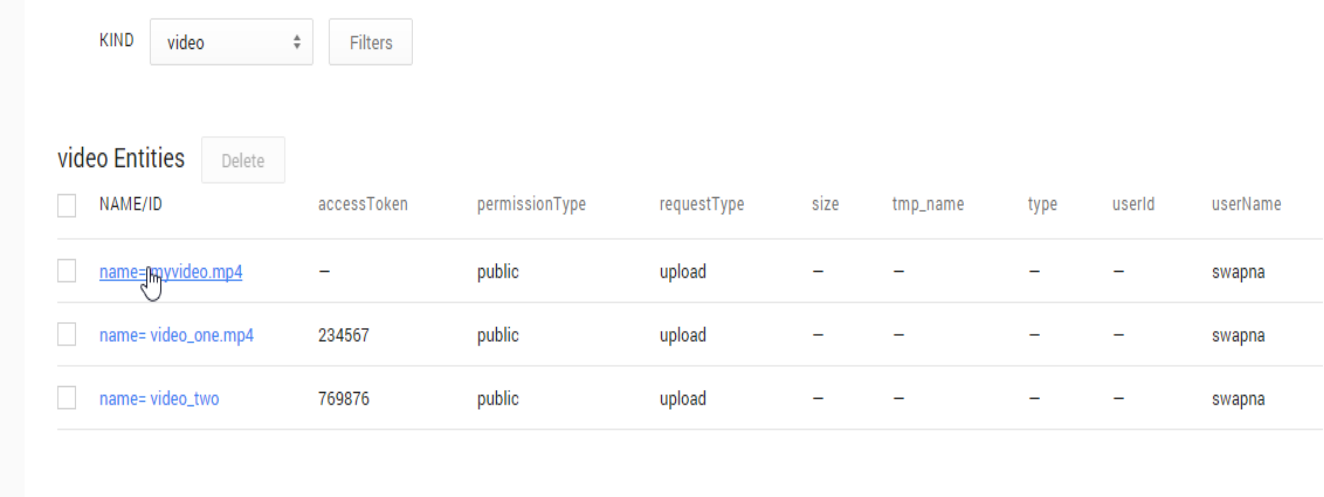

12:41
RefreshTokenActivity
Refresh Token
YES
NO

8) LOGOUT- On clicking the logout button the user is able to logout and gets the main homepage.

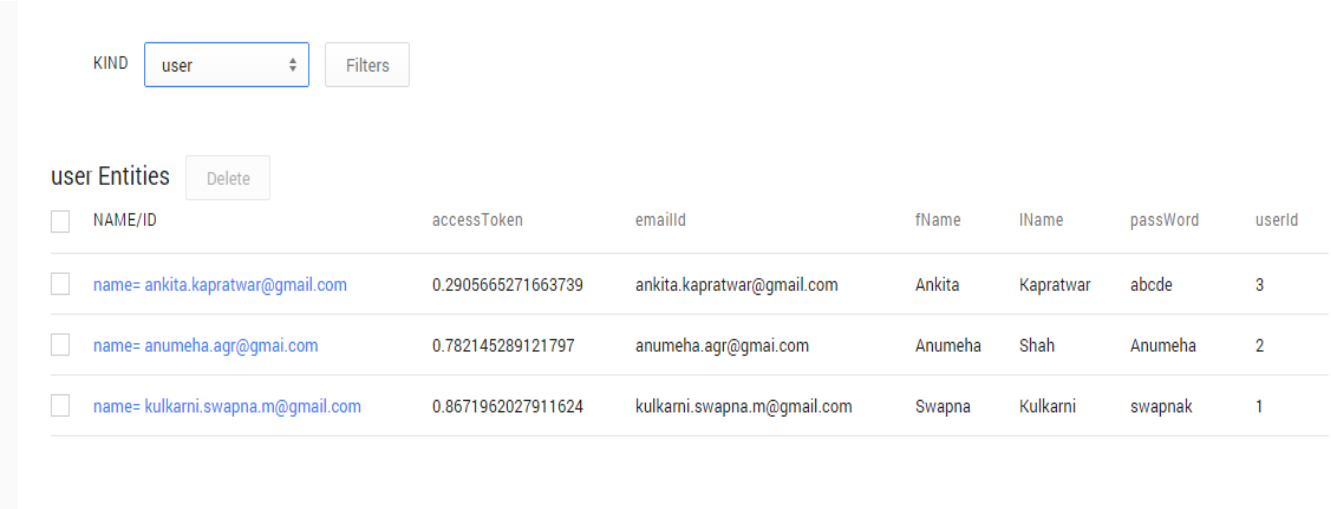


We deployed our application on Google App Engine. Following are the screenshots:

Video Upload and Permissions



User Registration and Login



PROJECT IMPROVEMENTS

We have the following suggestions for the improvement of the project:

1) Uploading the video- A facility to upload a video from the users SDCard can also be provided.

2) Refresh token - This facility can be implemented to refresh a token automatically without a button click.

3)Deleting a video- A functionality to only delete the private videos can be provided

CONCLUSION

From this project we learned Android and deploying rest apis on Google APP Engine, though the learning curve was steep we have successfully implemented the android application project with the features promised with our proposal. We have scope of improvements to extend the video functionality and security.

REFERENCES

[1] Android Programming: The Big Nerd Ranch Guide
[2] https://cloud.google.com/appengine/docs/java/blobstore/#Java_Serving_a_blob
[3] http://developer.android.com/training/sharing/receive.html
[4] http://androidexample.com/Create_A_Simple_Listview_-_Android_Example/index.php?view=article_discription&aid=65&aaid=90

[5] http://stackoverflow.com/questions/16246734/using-blobstore-with-google-cloud-endpoint-and-android
http://stackoverflow.com/questions/11406603/how-to-play-a-video-file-from-sd-card