Title: CS653 Project: AugmentedBOI

Project ID: 7

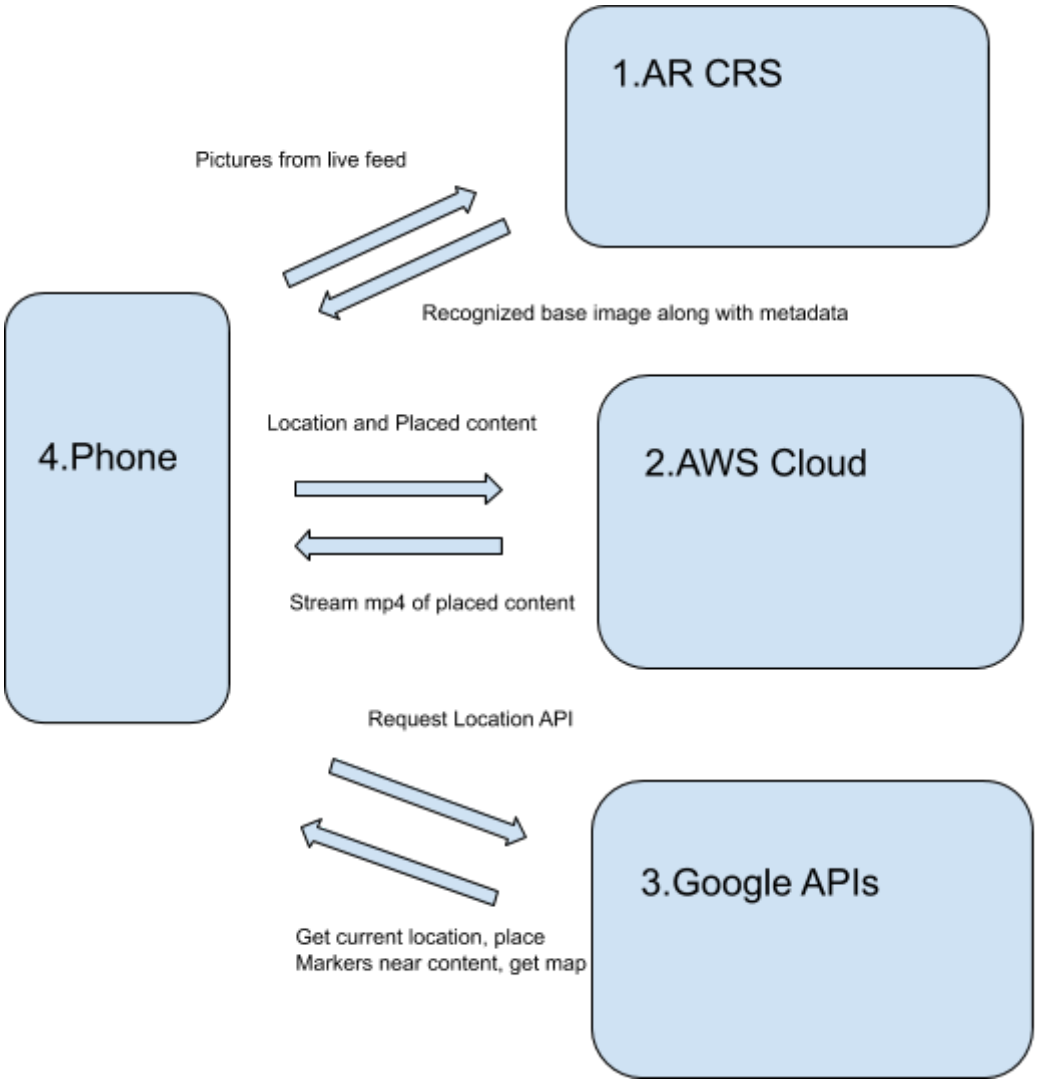
Objective:

Geo location based Augmented Reality App

Team:

Name	Roll number	email ID	Efforts
Surender Singh Lamba	140050075	140050075@iitb.ac.in	High
Chaitanya Rajesh Banala	140050073	140050073@iitb.ac.in	High
Sri Surya Teja Navuluri	140050055	140050055@iitb.ac.in	Medium

Architecture:



1. AR CRS/ Cloud Recognition System:

Provided by EasyAR SDK, this provides cloud storage for saving base image over which content needs to be displayed and provides feature extraction for low latency recognition. On recognition, returns metadata of the url to stream from AWS cloud which is placed over image in live camera feed to provide AR.

2.AWS Cloud:

Content to be displayed over base image is uploaded here and it undergoes resizing, compression and finally processed with FFMPEG in cloud to convert everything into mp4. Cloud also has authentication system for app which makes use of token for communication from the point of login. This also provides all the locations of nearby markers so those can be shown to user on device. In return the phone can stream the available mp4s directly available in cloud when a base image is found without downloading.

3.Google APIs:

Google's location and Maps api has been used to display a customized map along with markers whose location displays where all the AR content is available. This api also provides current location of user making use of GPS and Network providers.

4.Phone:

Phone has a flow for authentication and registration. The home screen is a custom stylized map which shows current location which is updated every 5 seconds and is refocused to a certain zoom level every 2 seconds in case the user wanders off. The other markers representing nearby AR content is updated every 30 seconds. It is taken care to carefully close location service when activity is inactive.

We further have a scan option which scans for base images in live feed using the CRS and streams content from AWS if something is found within the bounds of image.

The Create button from home prompts the user to take a base image and then presents four options of Draw over image, place picture, place video or place gif. Drawing makes use of phone's inbuilt gallery of Google photos app, others enable user to select content from gallery. This content is now pushed to AWS, then a post request sends location and url of uploaded content for processing. After this, a third request sends base picture to AR CRS for feature extraction along with AWS url in metadata.

Web APIs:

AR Target:
POST /targets/ application/json : pushes ar target image
Body:
{
"image":"Base64encodedstring",
"active":"1",
"name":"targetName",
"size":"5",
"meta":"AWS_MP4_URL",
"type":"ImageTarget",
"date": "2016-05-27T09:15:39.559Z",
"appKey": "test_app_key",
"signature": "sha1 signature"
}
Login:
POST application/json
{email:email, password:password}
– Response: application/json
{statuscode:Bool, key:key}

Register:
POST application/json
{email:email, password:password}
Response: application/json
{statuscode:Bool}

Content upload
POST application/json
{content:URL_UPLOADED_CONTENT, extension:"EXT", key:key, latitude:lat, longitude:long}
Response: application/json
{statuscode:Bool}

Marker download
GET application/json
{key:key, latitude:lat, longitude:long}
Response: application/json
{statuscode:Bool, nResults:N,results:[{lat:LAT, long:LONG} {lat:LAT, long:LONG} {lat:LAT, long:LONG}]

AWS uses TransferUtility class for multipart upload of content Async.

Hardware and Software Pre-requisites:

Hardware:

Phone supporting Android SDK minimum level 23 is required.

Software:

All additional libraries included apart from base libraries for android end used are listed:

'com.github.shem8:material-login:2.1.1'

'com.google.android.gms:play-services-location:11.8.0'

'com.google.android.gms:play-services:11.8.0'

'com.google.android.gms:play-services-maps:11.8.0'

'commons-io:commons-io:1.3.2'

'org.jcodec:jcodec:0.2.2'

'org.jcodec:jcodec-javase:0.2.2'

'org.jcodec:jcodec-android:0.2.2'

'com.squareup.okhttp3:okhttp:3.10.0'

'com.google.guava:guava:24.0-android'

'commons-codec:commons-codec:1.9'

'com.amazonaws:aws-android-sdk-core:2.3.3'

'com.amazonaws:aws-android-sdk-s3:2.3.3'

'Com.amazonaws:aws-android-sdk-ddb:2.3.3'

For AWS cloud, FFMPEG is used for encoding to mp4, interface is designed in Django.

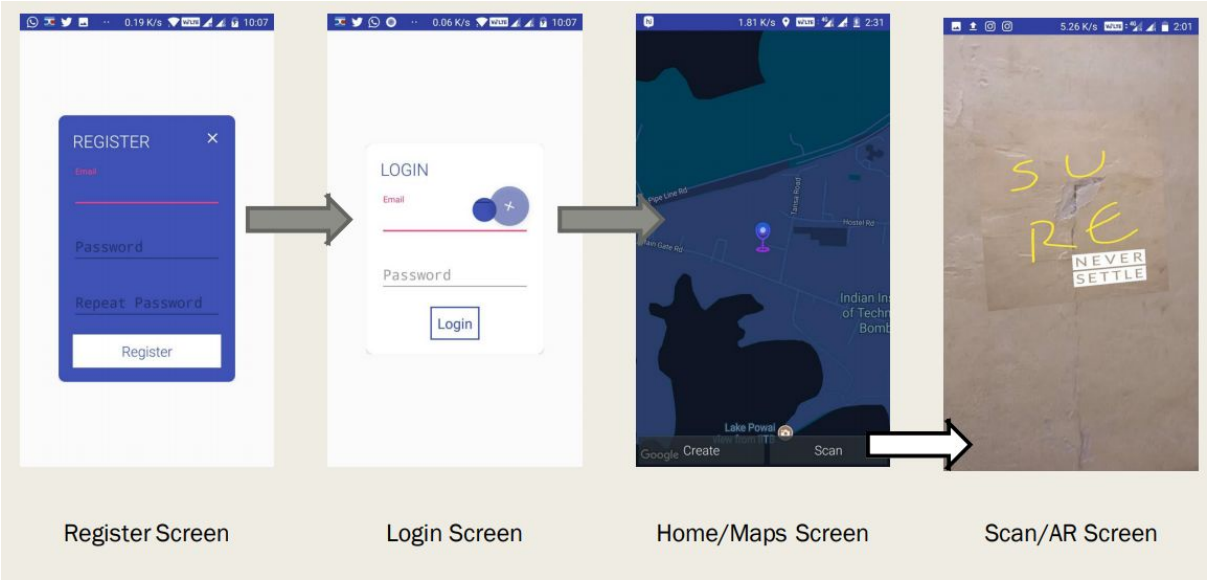
Progress After Midsem Demo:

At the time of midsem, we had no logins, no markers and everything of content was local and encoding with FFMPEG was done locally on shell with binary which was dependent on Chip architecture and was failing on CPUs other than ARM.

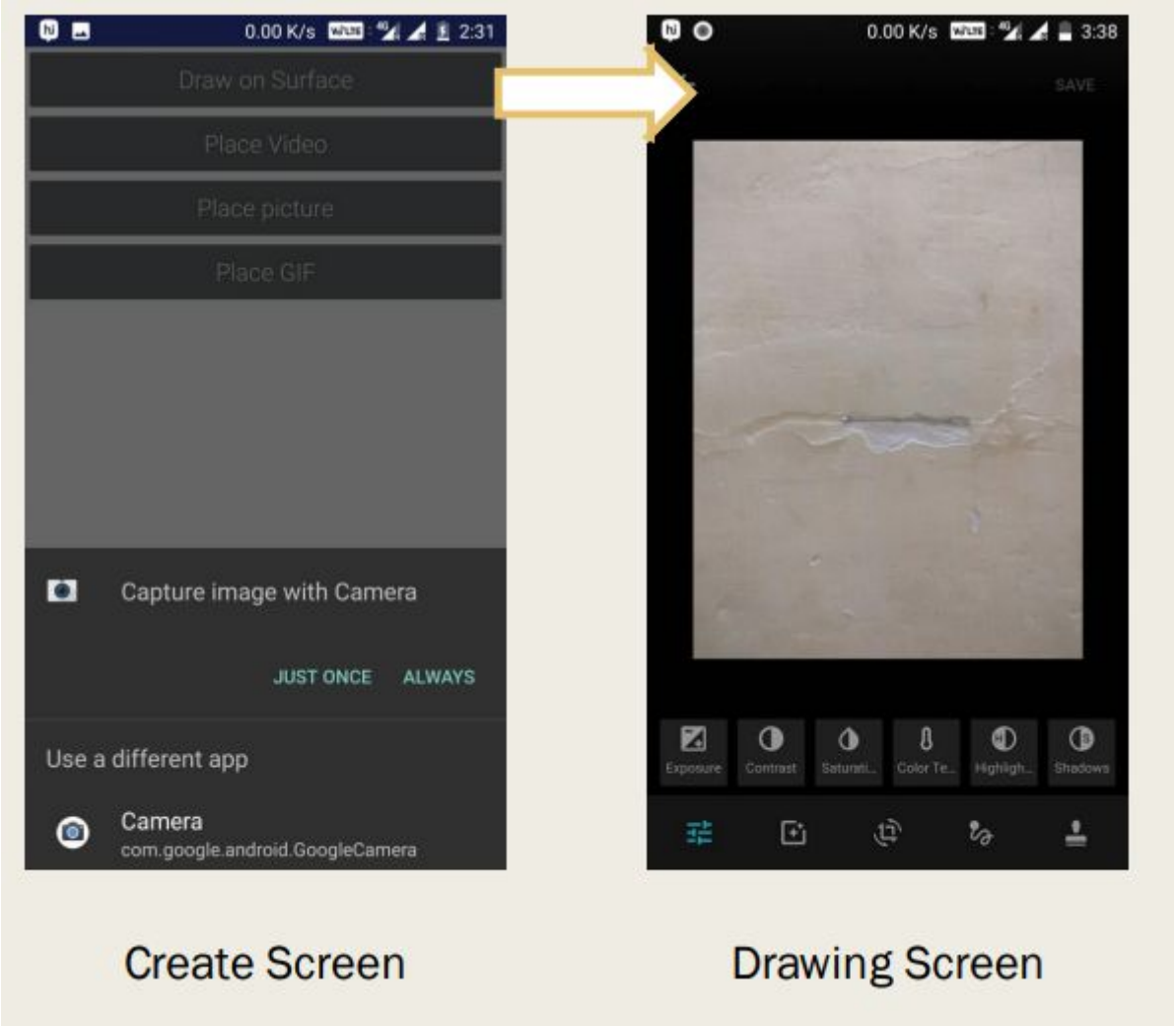
During demo, we were asked to complete the cloud implementation and provide reliable recognition capability as it was not working sometimes and were asked to look into further applications.

After midsem, we built the whole cloud infrastructure and moved all the processing part to cloud and brought the app size from 55 MB to 14 MB. We set up uploading to buckets directly to handle large multipart uploads instead of re-implementing in Django and to avoid Base64 conversion which was highly dependent on SDK versions and was in tug of war between android and java libraries. Further logins and markers were implemented to show where all places near the user, AR content is available. A suggestion during midsem demo discussion of implementing a live news over a newspaper article was tried using a sequence of Google OCR through Cloud Vision API and then scraping google search results for youtube results but was scraped as there is no use of having it in an AR heavy app.

Usage of the app:



If instead you want to create, press Create from Map screen:



Link to source code and executable:

https://drive.google.com/file/d/1e2w5_hRWzs4RmZG3Bcjd0tyYZ8pF0xT5/view?usp=sharing

Unfinished tasks:

From the initial proposal, some parts were dropped in favor of adding more types of content and having the current features done effectively. The proposal to add a welcome screen and how-to flow was dropped in favor of intuitive UX and adding news videos as suggested in midsem discussion was dropped because of reasons mentioned before in this doc.