



**San José State**  
UNIVERSITY

Term Project: TalesBook (Social Mobile App)

CMPE 277: Mobile App Development

(Thursday 6:00 PM to 8:45PM)

Submitted By, **TEAM 7**

<b>Name</b>	Venkata Narasa Kumar Kuchi	Neha Kumar	Sashank Malladi	Rahul Bharadwaj Vudutala
<b>Mail ID</b>	venkatanarasakumar .kuchi@sjsu.edu	Neha.kumar@sjsu. edu	sashank.malladi@ sjsu.edu	rahulbharadwaj.vudutal a@sjsu.edu
<b>SJSU ID</b>	010821876	010832523	010466651	010821980
<b>Major</b>	Software Engineering	Software Eng.	Computer Eng.	Software Engineering

Presented to

<b>Professor</b>	Charles Zhang Department of Computer Engineering, San Jose State University
<b>Term project</b>	TalesBook (Social Mobile App for Android)
<b>Date</b>	05-25-2017
<b>Version</b>	1.0

## Table of Contents

1. Motivation and introduction .....	3
2. High level and component level design .....	4
3. Technology choices .....	5
4. Description of features with final screenshots.....	6
4.1 Sign up and authentication .....	6
4.2 Email verification .....	8
4.2 User Timeline.....	9
4.3 Edit Profile (About Me) .....	9
4.4 My Posts .....	10
4.5 Follow People with Public Profiles.....	11
4.5 Friends .....	12
4.6 View Other Users profile.....	13
4.7 Private Messaging.....	14
4.8 Notifications of private messaging (via Email) if user is offline .....	15
5. Testing plan .....	15
6. Challenges and future work .....	18
7. Git repository for Source code .....	19

## 1. Motivation and introduction

As part of learning of for fulfilment of requirement for CMPE 277 Smartphone Application & Development, we have developed a native mobile app with full-fledged features for a social networking app. The application is hosted on cloud & is developed on Android SDK 21.

The app, henceforth called as **Talesbook** has all the features from user signup & authentication to profile management, to searching friends & sharing his views & pictures on his timeline.

Building the app gave in depth knowledge of android features & how functionalities can be built. There are more than one of doing same thing in android. Not only android, but we also learned about creating web services for backend & RDS for storing user information.

## 2. High level and component level design

There are basically seven main components which interact with each other & combinedly form the app.

1. User Nodes
2. Posts
3. Database node
4. Authentication system
5. Messaging/In Mail
6. Friends
7. Management of user profiles

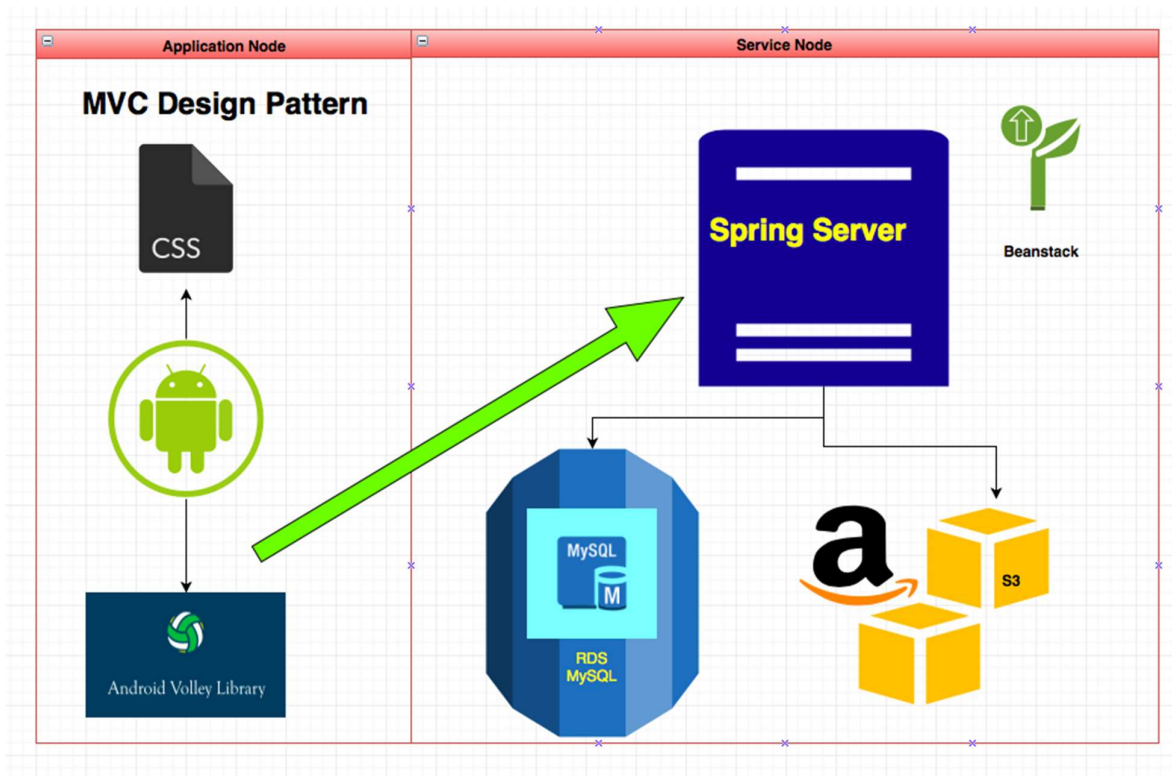


Fig 1. High level architecture design

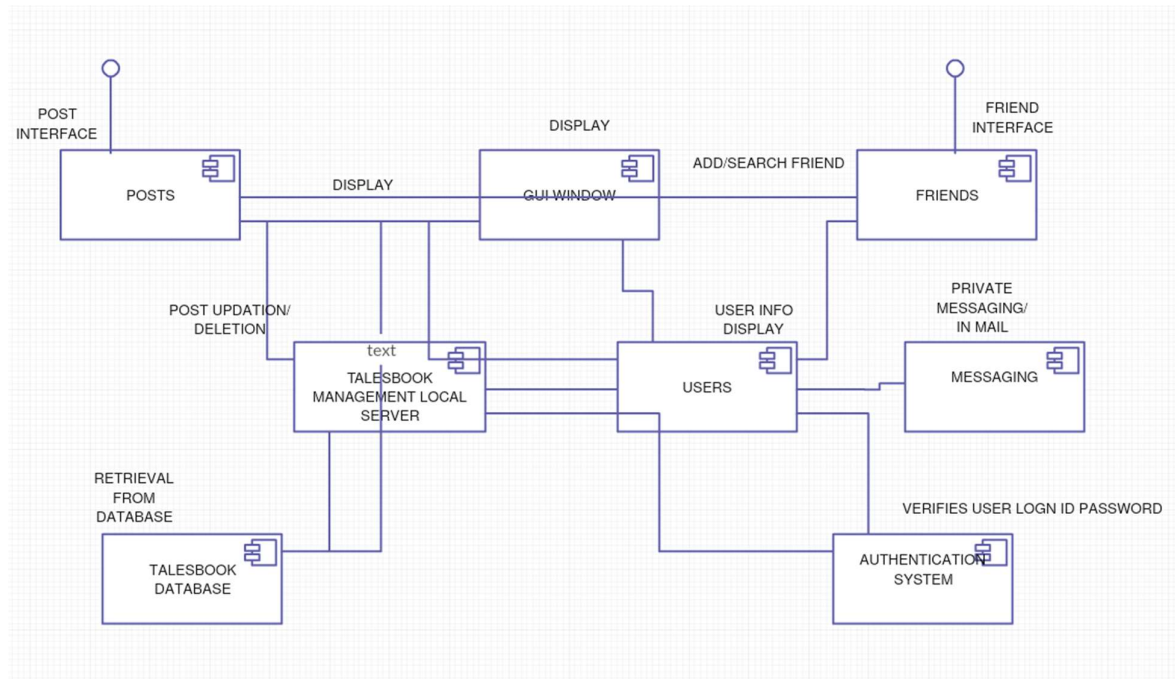


Fig 2: Component level design

### 3. Technology choices

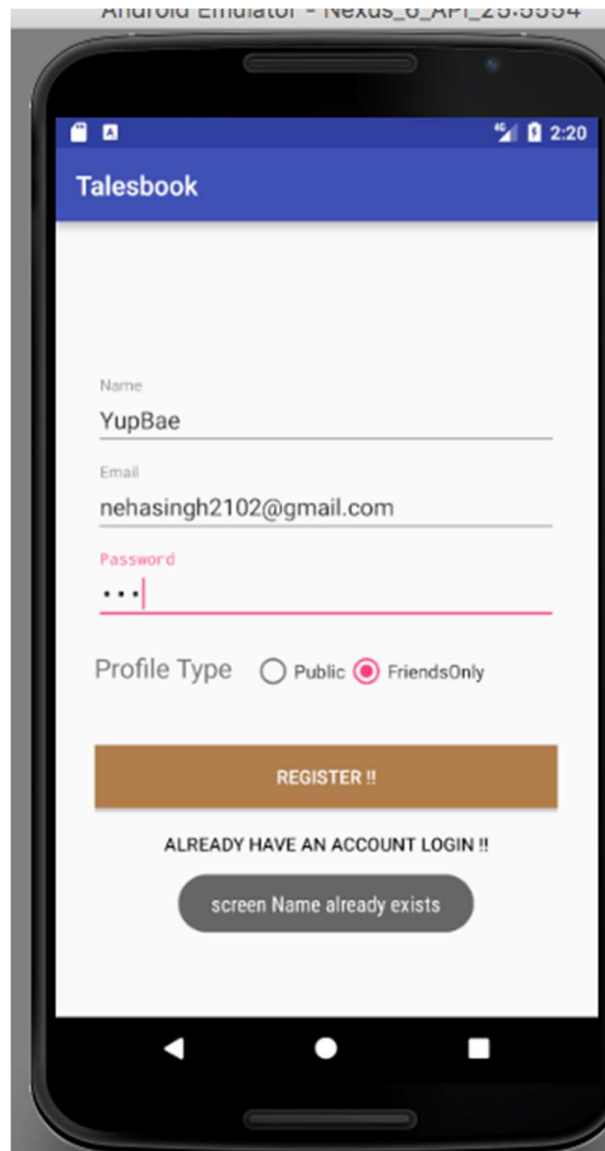
As shown in high level diagram. We have used Java for developing android platform for the application. We have used RESTFUL web services for creating the backend services. The server is hosted on Elastic Beanstalk on amazon AWS.

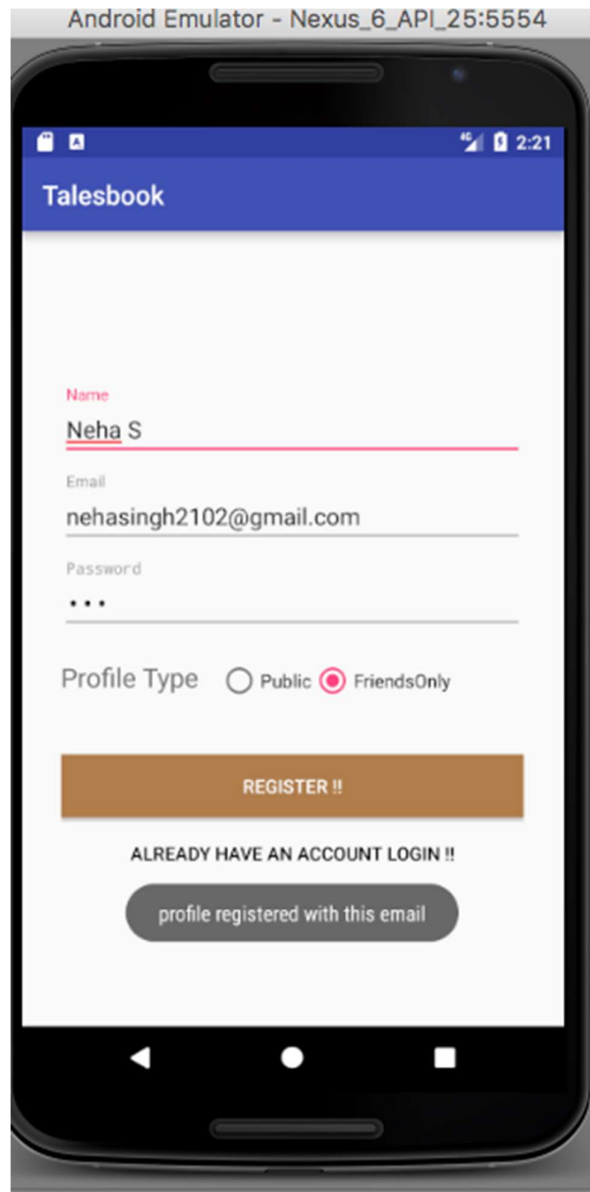
1. **Volley** : Volley is an HTTP library that makes networking for Android apps easier and most importantly, faster because of its asynchronous nature.
2. We are using **Amazon S3** for **storing the images** uploaded by used. It makes the data access faster & solves problem of storage.
3. **Database** used is **Amazon RDS**, SQL for storing all the user information, profile, posts, friends & profile settings. Database is also used for verification of users.
4. **JavaMailSender & SimpleMailMessage** for Email verification
5. For **messaging & In-mail** features also we have used **JavaMailSender & SimpleMailMessage**
6. **Auto-scaling** feature of **AWS** for creating more instances based on rules set.

## 4. Description of features with final screenshots

### 4.1 Sign up and authentication

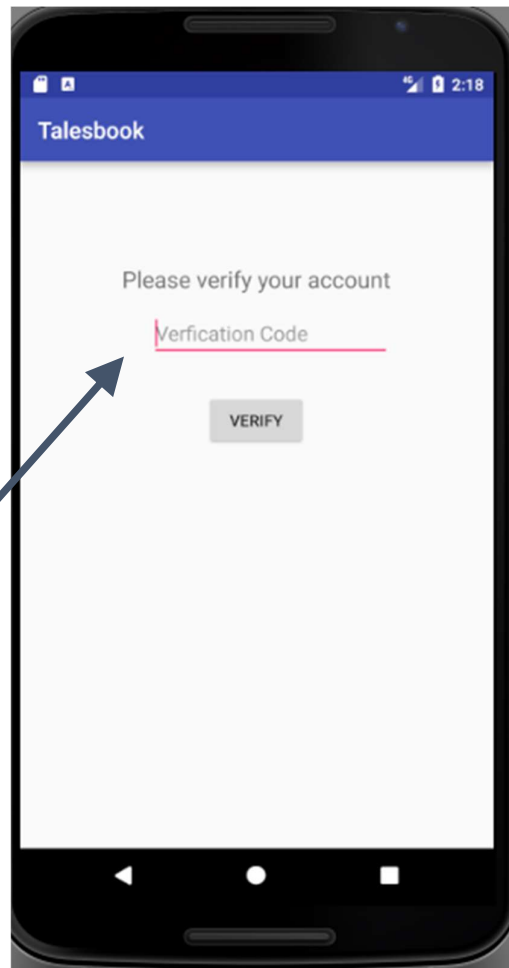
Implemented user signup, a user can sign up using his email & select a unique username for his profile. If a username is already taken, a popup message will be shown. Also, email needs to be unique every time for creating a new user.





## 4.2 Email verification

Once a profile is created, a user gets a verification email with a code. When user enters that code, the user is verified & he can login using his credentials (email/unique username)



Welcome to 277\_ SocialApp Inbox x



277socialapp@gmail.com

to me ▾

Please verify your account with following code888982

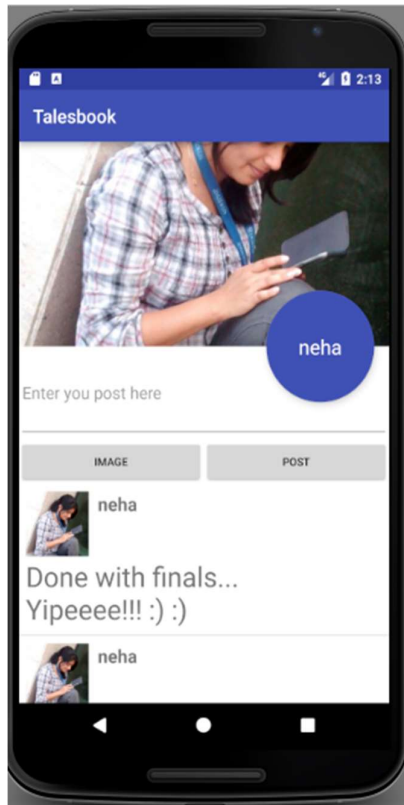


Click here to [Reply](#) or [Forward](#)



## 4.2 User Timeline

Users will be able to see their timeline once they login. They can post (text + images) as their posts. The timeline is a list of all posts posted by himself, friends, and people whom the user is following.



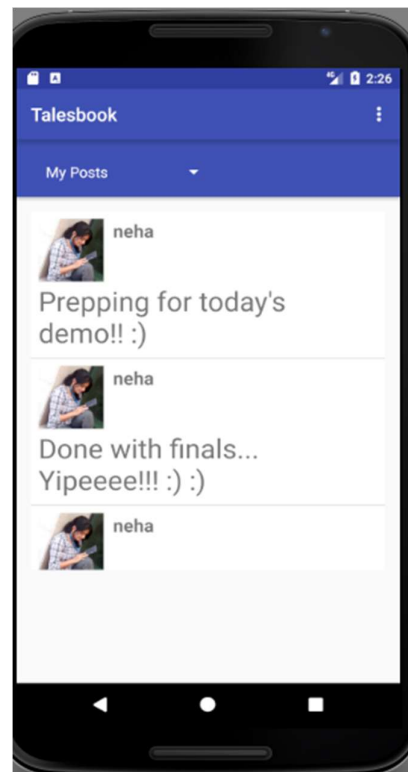
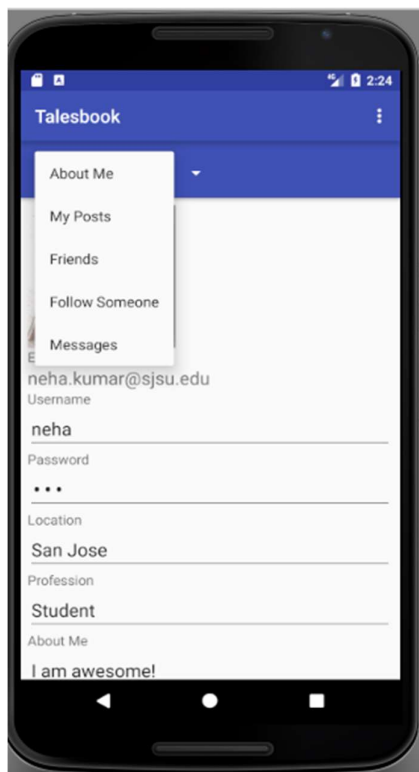
## 4.3 Edit Profile (About Me)

User can change the profile (image, username, password. Location, etc.) via edit profile. The About me page can be started by clicking on profile picture.






#### 4.4 My Posts

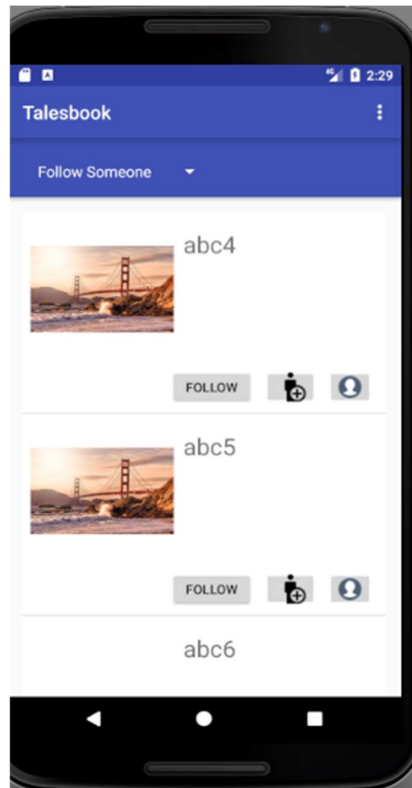
My posts are the posts posted by user. Use can see all his posts here.



#### 4.5 Follow People with Public Profiles

User can follow people or send friend request to other users whose profile type is public. The functionalities of buttons respectively,



- Follow user 
- Add friend 
- View profile 

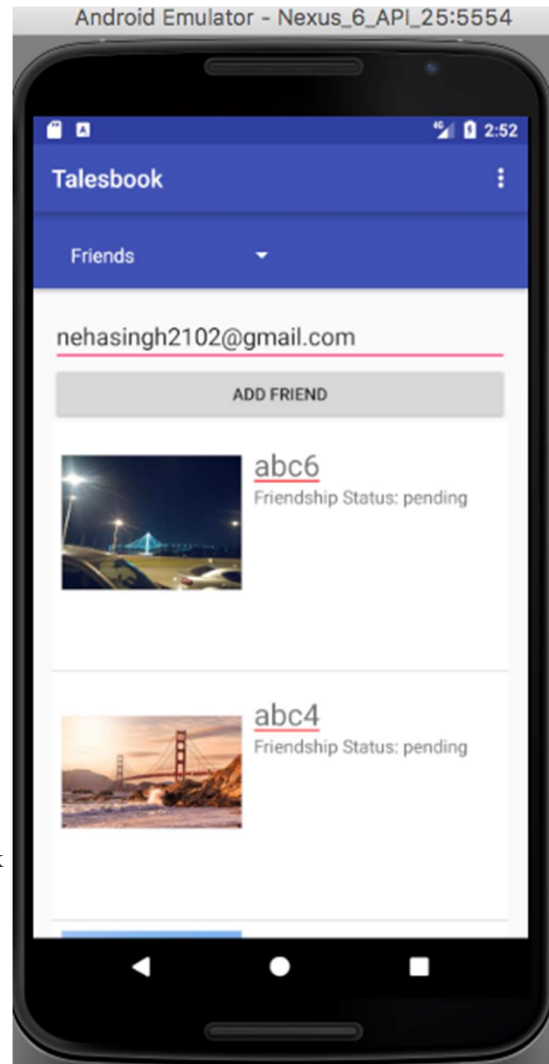
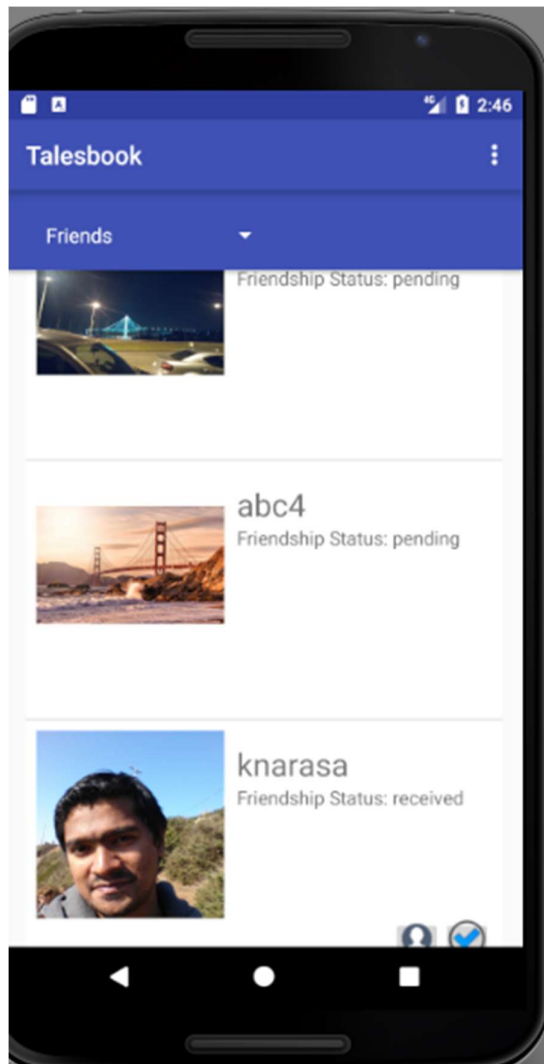


## 4.5 Friends

Users can view all friends (Friendship established, Friend request pending by current user, Friendship request sent by current user and pending with opposite user)


The functionalities based on type of friend and the status are

- View Profile (of friend) 
- Approve friend 
- Add Friend (User can add the friend by entering his friend's email. This will send a friend request to the other user and list gets updated with pending)



ork

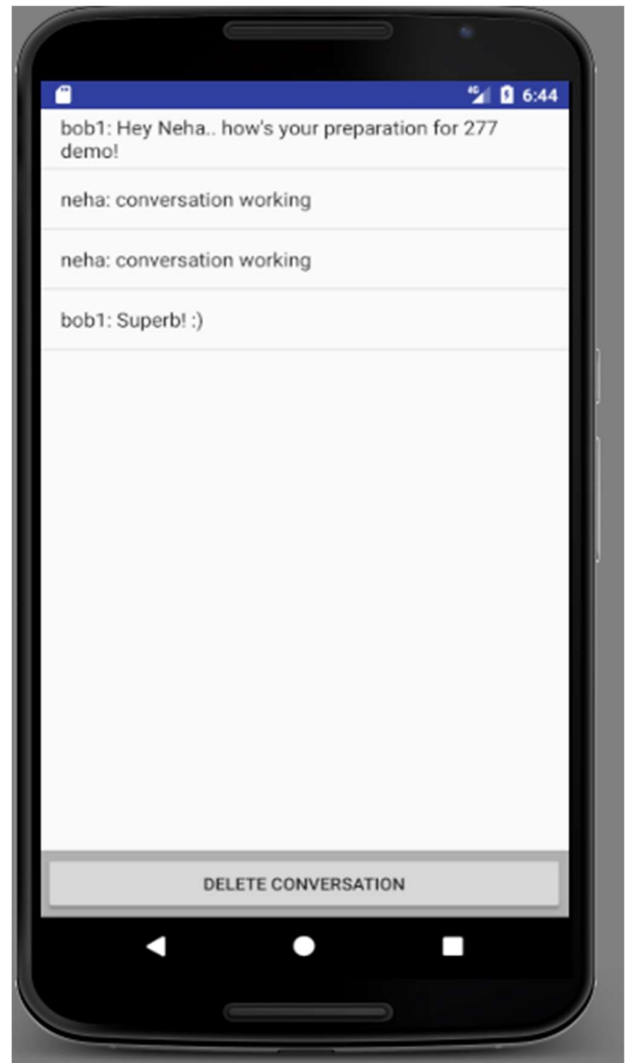
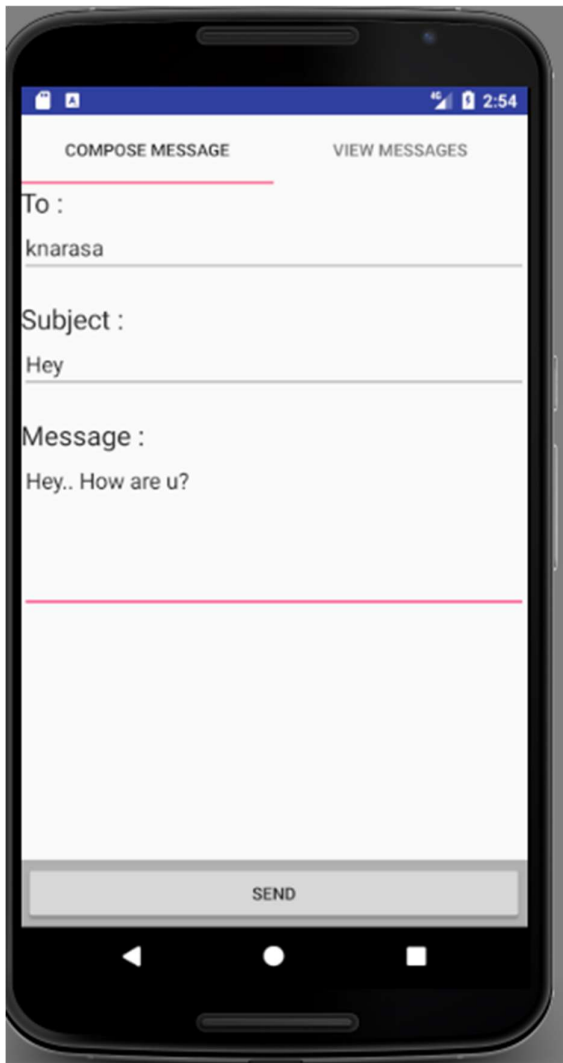
#### 4.6 View Other Users profile

- The current user can see Friends profile or user with public profile type by clicking on View profile icon (  ). User can also send a private message by clicking on Message button.
- User can also visit the other users post by selecting the spinner from About → Posts.



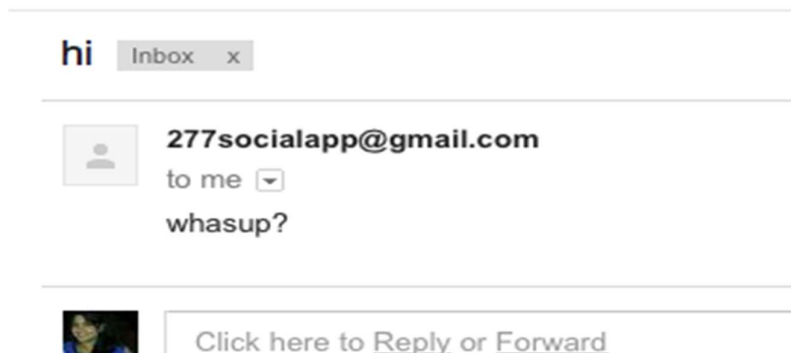
#### 4.7 Private Messaging

Any user can send private message to any of his friend or any person whose profile is public. This message will be saved under history grouped by each user. The user can delete the conversation by different users.



#### 4.8 Notifications of private messaging (via Email) if user is offline

Users can get notification when they are offline by email.



### 5. Testing plan

#### 5.1 Testing plan executed and results

##### Test case for Profile creation

1	Step 1: Use a new email address to sign up. Expected Results: A email with verification code will receive. Step 2: Input the verification code to login. Expect Results: Successful login and redirect to the home page of User	Pass
2	Step 1: User can view and update his profile at any time. Expected Results: The information will change accordingly after profile is being updated.	Pass
3	Step 1: User needs to select to unique screen name for him. If he selects already existing username, he will get a toast message Expected Results: A toast message is popped up incase a username is already taken	Pass

4	<p>Step 1: User needs to use new email address every time for creating a new account. If he uses an email id which is already used, he gets a toast message saying email is already used.</p> <p>Expected Results: A toast message is popped up incase an email is already used for creating an account.</p>	Pass
5.	<p>Step 1: User can set his profile picture.</p> <p>Expected Results: User is able to upload any picture as his profile picture.</p>	Pass

## 5.2 Test case for Post, Profile visibility & browsing public profiles

6.	<p>Step 1: User can choose to set his profile as public or visible to only friends.</p> <p>Expected Results: User is able to update his profile settings anytime.</p>	Pass
7.	<p>Step 1: User can post on his profile &amp; upload images as part of post.</p> <p>Expected Results: User is able to upload post &amp; upload picture.</p>	Pass
8.	<p>Step1: Once a user's profile is created, he can browse all the public profiles.</p> <p>Expected Results: User is able to see the posts of public profiles</p>	Pass
9.	<p>Step1: User can follow any user whose profile is public.</p> <p>Expected Results: Once a user follows a public profile, all his posts are visible to the user.</p>	Pass
10.	<p>Step1: User should see all the posts made by users who he is following.</p> <p>Expected Results: All posts of public profiles who he is following is visible on his timeline</p>	Pass



### 5.3 Test case for Friends

11.	Step 1: User can send friend request to anyone using his email address or unique username. Expected Results: User is able to send friend requests.	Pass
12.	Step 1: If Alice sends friend request to Bob, Bob gets a notification that he has received a friend request. Expected Results: User gets a friend request & same is visible in his friends tab which shows all his friends & pending friend requests.	Pass
13.	Step 1: Alice can see friends requests sent to others & it's status whether it's accepted or pending. Expected Results: Alice is able to view the status of her friend request sent to others.	Pass
14.	Step 1: Alice can see friends requests sent to her & she can chose to accept it or not. Expected Results: Alice is able to view the friend request sent to her.	Pass

### 5.4 Test case for Private Messaging

11.	Step 1: User can send private message to his friend or to any user whose profile is public. Expected Results: User gets a private message & same is visible in his messages	Pass
12.	Step 1: User gets in mail when he is not online & if someone has sent him a private message. Expected Results: An in mail message is sent to his registered email	Pass
13.	Step 1: When a user browses public profiles and picks one to send a private message, the screen name should be automatically populated Expected Results: Screen name is auto populated while sending a private message	Pass

14.	Step 1: User must be able to browse all his private messages, view the one selected, and delete those you do not want to keep. Expected Results: User is able to view & delete any message conversation.	Pass
10.	Step 1: User gets in mail when he is not online & if someone has sent him a private message. Expected Results: An in mail message is sent to his registered email	Pass

## 6. Challenges and future work

In this project, we got familiar with some key concept in Android development & how to use Spring framework, such as

- 1) Spring MVC, the model has no idea about the view part, the view only focuses on how to present data and the controller connects the model and view part.
- 2) JPA, a very useful technology to access data from database.
- 3) Transaction, four key properties of transaction is ACID which stands for Atomicity, Consistency, Isolation and Durability.
- 4) ORM, ManyToMany, ManyToOne, OneToMany and OneToOne relationships.

There were few challenges encountered while working on this Android application.

- Making asynchronous calls via Volley library to Server, and S3 for data retrieval
- Developing custom component for displaying posts and friends list
- Deleting messaged by one user should not delete history from other user in Private messaging
- Developing UI to fit in all types of mobile (in both orientations). Even though we have used Linear layout, the UI gets disturbed on rotating phone. This will be future scope.

In the future, we will beautify our web pages and implement the 1) interview function which allows company and seeker to arrange an interview online and 2) post function which enables seeker and company to share a new technology or an upcoming job affair.

## 7. Git repository for Source code

Please find below our team GitLab repository (Private repository) & APK file of our application is attached to source code directory in another submission.

- [GitLab Link For App](#)
- [GitLab Link For Services](#)