An Android Application to Integrate and Secure Social Networks and Emails

GRADUATE PROJECT REPORT

Submitted to the Faculty of
The School of Engineering & Computing Sciences
Texas A&M University-Corpus Christi
Corpus Christi, TX

In Partial Fulfillment of the Requirements for the Degree of
Master of Science in Computer Science

by

Anuraag Reddy Muppidi
Summer 2014

Committee Members

Dr. Mario Garcia
Committee Chairperson

Dr. Scott A. King
Committee Member
Abstract

Android is a mobile operating system based on the Linux kernel which is currently developed by Google. It is an open source operating system developed primarily for mobile handheld devices such as touch screen smart phones, tablets, wrist gear etc and since 2013, Android devices have been outselling their competitors IOS, Windows and Blackberry. Social networking and the use of emails on mobile devices have also increased greatly over the past few years with the rise in the popularity of smart phones and mobile devices such as tablets, causing the need for special applications to be built to serve the demands of the public.

OneNote application, for Android, has the most frequently used email and social media such as Gmail, Facebook, Twitter and LinkedIn integrated under one roof. Generally, when a user logs in to his/her application on a smartphone, it stays logged in. The user has to logout from all the social apps to ensure security. It also provides various setting options. To further increase the security of the application from attackers, two-step verification is implemented. This report will detail the processes required in the application.
# TABLE OF CONTENTS

Abstract ................................................................................................................................. ii

Table of Contents .................................................................................................................. iii

List of Figures .......................................................................................................................... v

List of Tables .......................................................................................................................... vi

1. Background ......................................................................................................................... 1
   1.1 Introduction ...................................................................................................................... 3
   1.2 Previous Work .................................................................................................................. 4
   1.3 Existing Application ....................................................................................................... 8
       1.3.1 APP Lock ............................................................................................................... 8
       1.3.2 Social Lock .......................................................................................................... 8
   1.4 Solution ......................................................................................................................... 9

2. Rationale ............................................................................................................................. 10
   2.1 Motivation .................................................................................................................... 10
   2.2 System ......................................................................................................................... 11
   2.3 Project Scope ............................................................................................................... 11
   2.4 Functionality ............................................................................................................... 12

3. System Design ................................................................................................................... 13
   3.1 System Architecture ..................................................................................................... 13
   3.2 Design Flow ................................................................................................................ 14
3.3 Use Case Diagram..................................................................................16
3.4 Sequence Diagram..................................................................................17
3.5 Modules......................................................................................................18
3.6 Class Diagrams..........................................................................................20

4. System Implementation ...............................................................................22
   4.1 Android Components..............................................................................22
   4.2 Requirements..........................................................................................22
   4.3 Outcome..................................................................................................22
   4.4 Android Application................................................................................22

5. Testing........................................................................................................38
   5.1 Registration............................................................................................38
   5.2 Login........................................................................................................39
   5.3 Login Verification....................................................................................40
   5.4 Social Networks.......................................................................................41
   5.5 Settings....................................................................................................42

6. Conclusion and Future Work.......................................................................44
7. Bibliography ...............................................................................................46
## LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure 3.1</th>
<th>System Architecture</th>
<th>13</th>
</tr>
</thead>
<tbody>
<tr>
<td>Figure 3.2</td>
<td>Design Flow</td>
<td>15</td>
</tr>
<tr>
<td>Figure 3.3</td>
<td>Use Case Diagram</td>
<td>16</td>
</tr>
<tr>
<td>Figure 3.4</td>
<td>Sequence Diagram</td>
<td>17</td>
</tr>
<tr>
<td>Figure 3.5</td>
<td>Class Diagram</td>
<td>21</td>
</tr>
<tr>
<td>Figure 4.1</td>
<td>OneNote Application Splash Screen</td>
<td>24</td>
</tr>
<tr>
<td>Figure 4.2</td>
<td>Checking connectivity at splash screen</td>
<td>25</td>
</tr>
<tr>
<td>Figure 4.3</td>
<td>No Network</td>
<td>25</td>
</tr>
<tr>
<td>Figure 4.4</td>
<td>Login screen</td>
<td>27</td>
</tr>
<tr>
<td>Figure 4.5</td>
<td>Sign up activity checks user details</td>
<td>27</td>
</tr>
<tr>
<td>Figure 4.6</td>
<td>Sign up screen</td>
<td>28</td>
</tr>
<tr>
<td>Figure 4.7</td>
<td>Random number generation</td>
<td>29</td>
</tr>
<tr>
<td>Figure 4.8</td>
<td>Verify login</td>
<td>29</td>
</tr>
<tr>
<td>Figure 4.9</td>
<td>Dashboard</td>
<td>30</td>
</tr>
<tr>
<td>Figure 4.10</td>
<td>Facebook login</td>
<td>31</td>
</tr>
<tr>
<td>Figure 4.11</td>
<td>Access token for Facebook login</td>
<td>31</td>
</tr>
<tr>
<td>Figure 4.12</td>
<td>Facebook user Profile</td>
<td>32</td>
</tr>
<tr>
<td>Figure 4.13</td>
<td>Load the user profile picture</td>
<td>32</td>
</tr>
<tr>
<td>Figure 4.14</td>
<td>Facebook share screen</td>
<td>33</td>
</tr>
<tr>
<td>Figure 4.15</td>
<td>Facebook recent posts</td>
<td>34</td>
</tr>
<tr>
<td>Figure 4.16</td>
<td>Twitter Share</td>
<td>35</td>
</tr>
<tr>
<td>Figure 4.17</td>
<td>Twitter file attachment</td>
<td>35</td>
</tr>
</tbody>
</table>
Figure 4.18 Compose email.................................................................36
Figure 4.19 Inbox and delete email.....................................................37
Figure 4.20 LinkedIn Authentication....................................................38
Figure 4.21 Settings screen.................................................................39
Figure 5.1(a) Unsuccessful Registration.................................................41
Figure 5.1(b) Successful Registration.....................................................41
Figure 5.2(a) Unsuccessful Login..........................................................42
Figure 5.2(b) Successful Login..............................................................42
Figure 5.3(a) Unsuccessful Login Verification.........................................43
Figure 5.3(b) Successful Login Verification.............................................43
Figure 5.4(a) Unsuccessful Facebook Login............................................44
Figure 5.4(b) Successful Facebook Login...............................................44
Figure 5.5 Change Password...............................................................45
Figure 5.10 Landscape screen view.....................................................45
1. BACKGROUND

After the advances in technology, we can say with certainty that smart phones have replaced the standard mobile phones in the market. A combination of Personal Digital Assistant and cellular phone, smart phone use has increased over the past few years. In 2013, about 90 per cent of the mobile phones purchased were smart phones and between 2012 and 2014, the number of smart phone users increased by about 0.75 billion. The total usage is now estimated (as of 2014) to be around 1.75 billion (Pewresearch 2014).

The smart phones are built on a mobile operating system with computing capabilities and advanced connectivity (Pewresearch 2014). They are very portable and help the users perform their tasks easily and simultaneously. Instead of carrying Personal Digital Assistants or personal computers, the user can simply carry the smart phone, which is much more portable and are capable of multitasking – they can send a text message or an email or make a phone call from the smart phone; it doesn’t simply perform one duty only.

Smartphones use a variety of operating systems. Some of the very popular mobile operating systems are Apple’s iOS, Windows, Nokia’s Symbian, RIM’s BlackBerry OS, Android etc. Android is a Linux based operating system and was first used on a smart phone in October 2008 on HTC’s Dream. Since then, there was no stopping it and the number of phones using Android OS increased greatly.

These smartphones have a lot of functionality in them. The users can customize User Interfaces and make them more convenient depending on their use. They have different kinds of menu screens and home screens. The menu screens have all the
applications on the phone and the user can add shortcuts to these applications on the home screen. These phones display live content, videos, notifications from various applications etc. They display notifications from text messages, emails, social networking apps etc. A click on these notifications will re-direct the user to the specific application from which the notification came.

Android has its compatibility with different hardware and it supports features such as online gaming, video streaming, video calling, Wi-Fi Internet, multitasking etc. Android also has the capability of printing the screen via screenshots. The phone captures an image of the entire screen as it is with a command. Android has attracted much interest in development because the following facilities it provides:

- it is an open source operating system.
- it can be easily customized and social network APIs are easily available for integration, and
- the use of android operating system on mobile phones has increased greatly over the past few years and it also has the highest number of applications available.

The Android operating system allows different third party developers to develop applications for it, unlike Apple where the developer has to be registered with the development site. There are different kinds of applications available and they are divided into categories on the Play Store, which is where the user generally downloads applications for Android phones. The different categories of applications include news, photography, sports, productivity, music, health, lifestyle etc.

There were about 1 million applications available for Android as of 2013 and the numbers of applications downloaded from the Google Play Store were about 48 billion.
The users will download applications from the play store and use them to their convenience.

1.1 Introduction

Security in smart phones and tablets can be applied at various levels; the security of the phone in general, and the security of applications in the phone such as social networking apps, online banking apps, email apps, online shopping apps etc. Logged in social networking apps and email apps are very vulnerable to misuse. These apps may contain a lot of sensitive information and secure data. Vulnerabilities in these apps will attract misuse of the data and information in these apps.

As of now, the android operating system has individual apps for each of the social networking sites. The user has to go to each of the apps to access them. These smart phones have been built with multi-tasking abilities to switch between apps while the other apps are still running. For example, a user can talk on the speakerphone while looking at the Facebook app.

When a user is logged in to the email or social app, all the notifications and updates within the app can be accessed. When the app is exited, it will not logout. The email and social apps on these smart phones are live apps. Once logged in, they are always logged in and the user has to log out to ensure security.

It becomes convenient for the users to access their email and post to social networks from one app. This will not only save the time for the user, but will also keep those apps secure by ensuring a single touch logout.
1.2 Previous Work

There has been a lot of development going on in the field of android operating system and smart phones. Over the past 5 years, there have been tremendous developments and great innovations in this field, including hundreds of thousands of apps that were developed in this time, with each app intending to serve the user better. The usage and interest in the app depends on the user’s smartphone usage, however, and all the apps may not be suitable to all the users.

Over a few years, Android and Apple’s iOS have been the two most popular operating systems used in smartphones. A survey of Norwegian mobile phone users was conducted and observations were made. The observations suggest that Apple has a wider network of people for social networks with Apple users having more friends than Android users. Perhaps this is because Apple IOS came to existence before the Android operating system but despite this, the growth in the Android operating system is still very high. The use of Android and Apple also varied from urban to rural areas (Johannes Bjelland 2012).

Android mobile devices have become a popular alternative to traditional PCs. The rise in the tasks being performed on these devices suggests that sensitive information is sometimes stored on these devices. Due to this, they have become targets for criminal exploitation. The existing security can be changed or improved by android modifications, which can reduce the potential of criminals to do harm.

Permissions removal is proposed, where a reverse engineering process is applied to remove the app’s permissions to a source. The repackaged app can run on all devices that the original app had supported. A study of certain social networking apps for Android
devices shows the difficulty of permissions removal, varying between the type of permissions and how the permission is integrated within the app (Quang Do 2014).

Human beings have become very social, in terms of how much they interact with others outside of people within their family. They constantly want to stay in touch with their parents or children or relatives or friends or work colleagues. Meet You is an app where the users are alerted when their friends or family are around. Linking the GPS with the social network app can do this. The user can do a check-in by using the social app when he/she is at a particular place. This would let the users friends know that the person is around that location. This feature could help the user keep track of a friend’s location and also let the users friends know about the location (Alexandra-Mihaela Siriteanu 2013). But this information would need to be private and secure.

The social networking applications are built on certain architecture. There is a new way to develop social networking apps that are context-aware, via Mobile Semantic Web. These applications change the way of communication by making use of the user’s context; micro blogging for example, which is an efficient way to convey the current activity and situation by using user’s context.

There is a difference between using social networking on static computers and mobile devices. Though the current mobile devices have good connectivity and accuracy, the users contexts may change very frequently, hence an agent-based system architecture based on distributed platform is required for developing semi-automated and semantic social networking applications (Juwel Rana 2009).
Social networking information is being used in ways that it might not have been intended originally. Due to the increased use of smart phones capable of running applications that can access the social network information such as the user’s location and preferences, the other applications are aware of the user’s preferences and location. For example, apps including Spotify and Shazam are capable of posting on behalf of the user in to the Facebook account. Other external apps can also access social networks such as Facebook and post on the user’s behalf. This needs permissions from the app though. The user can change the settings inside the app and block the app from accessing the social network.

Location based services need to query the mobile devices for the user’s social network information without disclosing the user’s identity or compromising the user’s privacy. These solutions will play a very important role, as the mobile social networks tend to grow exponentially (Aaron Beach 2009).

People all over the world use social networks. The users distributed worldwide, use different service providers. There may be trust issues with the independent service providers worldwide, hence it becomes hard to explicitly set-up their public keys with the conventional PK infrastructure. Identity based encryption schemes enable secure communication with social networking applications based on user identities like telephone numbers. By using these identity-based functions, user can send messages using these social networks all over the world, securely (Yongdong Wu 2013).

Face recognition is another method used to improve the security in android mobile smart phones. The face recognition feature captures the image of the user and sets it as the password. When the user tries to unlock the phone, the camera captures the image of
the person trying to unlock and matches it with the password image. If these two images match, then the phone unlocks. This feature has some negative sides, as the person’s image can be captured in conditions where the person is not willing (Antonia Rana 2013).

In the last 20 years, both mobile communication and Internet grew in parallel. There are many sites that are designed for mobile phones. These sites allow the user to login with a user name and a password. Captcha does not require any user name and password. Instead of entering a password, the user is shown a difficult code and he has to enter that code. The code is sent along with the users mobile phone IMEI number and when the server matches it with the information with in it, it allows the user to enter the site. This provides the user with a password less login. This has been implemented on the Nokia N71 phone (Mohammad Shirali-Shahreza 2007).

1.3 Existing applications

Android has over a million different applications and each of these applications help the user solve problems in a different way. One of our problems is to secure the social networking apps with a single user ID and password. Other developers have tried to solve this issue in a different way.

1.3.1 APP Lock

APP Lock is an app that is used to lock the applications on a phone. The user can use this app to select the applications that he/she wants to lock. Any social network applications like Facebook, Twitter, LinkedIn, Gmail, and Whatsapp can be locked with a passcode. When the user tries to enter these applications, it prompts the user to enter the set passcode, thus keeping the apps secure and private.
There are some negative sides to this application. The user has to enter the passcode each time he opens a secured application, and when the user is chatting or interacting with people on different social networks, he has to switch between the applications too often. Due to this, the app lock launches too many times and there is a lot of context switching which, as a result, slows down the phone and kills too much battery. It also becomes quite an effort for the user to enter the passcode so many times; it wastes a lot of time and frustrates the user. (DoMobileLab 2014).

1.3.2 Social Lock

Social Lock is an app that is used to lock all the social networking apps on the phone. This app will have a pattern lock that can be set at the beginning and that lock will secure all the social networking apps. Social networking apps need ease of operation. The user should be able to move between the apps and back and forth easily. This kind of a lock will make the user suffocate by prompting him/her to enter the lock code each time.

Sometimes people have conversations with one person on one app and another person on the other app. When the number of persons the user is talking to increases, the switching between the apps increases. There is a need for the user to have all these apps under one roof and also secure them with one password, so that the user can log in and log out of all the apps easily (1BNSoftware 2014).

1.4 Solution

This project will build an application in which the favorite social networking apps and emails apps like Facebook, Gmail, Twitter and LinkedIn are integrated into one app.
This application will have login page and a registration page. At the registration, the user has to register to the social networking apps by providing the ID and passwords. The database will store all the registration details by creating tables. The user should also set up an ID and password for this application. After this process, whenever the user logs in to the application, Gmail and their other social networks can be accessed to send emails, check inbox, reply to mails and post to the network.

It also has settings options where screen orientation; notifications services (enable and disable) are available. This way the user will not need to log out from all the applications separately and make posts from one application. This will save a lot of time and also keep the social networking apps secure. This application will also use a two-step verification using random 4-digit code, thereby restricting attacks from hackers.
2. RATIONALE

Due to the increase in the development of communication and mobile phones, the usage of these phones has increased so much that it is inconvenient for a person in today's world to live without a smartphone. Smart phones have been becoming compact and sleek from the time they came into existence. They are designed differently depending on the use of the person. For example, some smart phones have smaller screens to increase the mobility and ease of operation. Where as, some smart phones have bigger screens to increase the pleasure of watching videos, playing games etc.

The social networking apps contain the entire social networking history of the user. From the time a user registered to a particular social network to the present date, all their activity is stored on its server, hence there is a great need to protect these applications. Also, to increase the ease of operation of these networks, integration of multiple social networks to a single app is important.

2.1 Motivation

From the time the smart phones came in to usage, there have always been problems with the security of these phones. There has never been a phone that is completely secure. Social networks are one of the most vulnerable places, where the user has a lot of personal and sensitive information that needs extra security and protection.

As of 2013, 73% of the adults of the United States use social networks and are very active. 25% of the messages sent from social networks were unauthorized. 43% of the inappropriate comments made on these networks were from unauthorized access. 62
out of 100 times, logged in social networks were used to play pranks and other inappropriate activities. Friends of the users made 36% of the pranks and fake statuses. Out of the total changes in personal details on the social profiles, 24% were unauthorized and were made by someone else. 47% of the pranks played were abusive and embarrassing (IndiaGarner 2013).

Some of these pranks led to violent actions from the users and some led to suicides and negative activity due to the irreversible nature of them, hence there is a great need to secure these social networks, as well as a great need to have all the social media under one roof, so that it becomes more convenient and easy for the user to post.

2.2 System

This application has the frequently used email and social media Gmail, Facebook, Twitter and LinkedIn integrated under one roof. Generally, when a user logs in to his application on a smart phone, it stays logged in. The user has to logout from all the social apps to ensure security. It also provides various setting options. To further increase the security of the application from attackers, two-step verification is implemented.

2.3 Project Scope

This is an android application that has to be downloaded and installed on a phone. It will have the selected social networks integrated with it. The user can post to social networks directly from this application. Once installed and registered, it will automatically login the user to all the social networks and a logout of the application will log the user out from all the applications.
The application can be distributed via email and is a .apk file. The users can download the application and run it with a single touch. Once this is done, the user can register to the social apps and set up a login ID and password.

### 2.4 Functionality

The android application will have the following functions that the user can use, to improve the security and also the ease of operation:

1. Single touch login and logout.
2. Single Interface.
4. Two-step verification.
3. SYSTEM DESIGN

This chapter discusses the design of the architecture of the entire system. This chapter also discusses the flow diagrams, use case diagram, modules, important components and class diagrams.

3.1 System Architecture

![System Architecture Diagram]

**Figure 3.1 System Architecture of OneNote.**

Figure 3.1 shows the architecture for OneNote application. This application is a stand-alone app and it does not have its own server. It makes use of the SQLite database server to store the usernames and passwords temporarily. The app connects to the Gmail server using SMTP and IMAP protocols with secure socket layer port 465 and 993 respectively. The incoming mail connections use IMAP while the outgoing mail connections use SMTP. The Login validation checks the username and password entered with the username and password in the database and confirms or rejects login
accordingly. For social networks, access tokens are used to control the session. The authorization for social networks is done using their own APIs. For file attachments, Bitmap is used to decode the file path and compress the attachments. Broadcast receiver runs in the background to check for new emails received and the notification manager gives the new notifications to the app. These components and modules are discussed clearly, later in this chapter.

3.2 Design Flow

Figure 3.2 shows the design flow for the application. The activity starts with the SplashScreenActivity class. This activity runs while the AppDataBaseHelper loads the database tables. The LoginActivity has options for the user to signin or signup. The user has to signup first. Hence, the app goes to SignUpActivity class. In this class, the user enters the details such as username, password, confirm password, email and phone number. The AppDataBaseHelper uses the phone number provided as userUniquePhnumber. It stores the data entered against the users userUniquePhnumber. Whenever the user logs in to the app from the LoginActivity, the AppDataBaseHelper loads the user data against the userUniquePhnumber.

A two step verification is done by generating a random 4-digit number in the VerificationActivity class. The randomly generated code is sent to the user as a text message to the userUniquePhnumber. The user has to enter the code at the login verification screen to enter the application.

MainDashBoardActivity has the buttons for the social networks, Gmail and settings. It also has a back button and a logout button.
The social networks use their own API for authorization. Facebook provides APPID, consumer key and secret key, while LinkedIn and Twitter has the consumer key and secret key. These keys are used to connect with the social network APIs for authorization. The session is controlled using the access tokens and expire time which are stored in the SharedPreferences for which we have an object mprefs.

Figure 3.2 Design Flow Diagram of OneNote
3.3 Use-case Diagram

Figure 3.3 shows the use case diagram for the system with three actors and components. The actors are the user, database and notifications. The use case diagram shows the interaction between the actors and the components.

Figure 3.3 Use case Diagram of OneNote.
3.4 Sequence diagram

Figure 3.4 shows the sequence diagram for the activities in the OneNote Application.

Figure 3.4 Sequence Diagram for OneNote.
3.5 Modules

This system made used of several modules for functioning of the app, such as:

1. SQLite Database (AppDataBaseHelper)
2. Noticeboard receiver
3. SMTP and IMAP for Gmail
4. Access tokens for social network sessions
5. Bitmap for file attachments

1. SQLite Database:

The application needs a database to store the user details, recent posts and others. SQLite database is a server less, transactional SQL database engine. AppDataBaseHelper class is used for the SQLite database. An object helper is created to use this database in other classes. It stores the details of the user against the userUniquePhnumber, which is the number provided by the user at signup. It uses several strings to store user details session IDs, Gmail inbox details, Facebook upload details and others in tables. When the user logs out the session is cleared.

```
db.execSQL("CREATE TABLE " + fbUploadsTable + " (" + fbUniqueID + " TEXT," + fbFilePath + " TEXT," + fbMessage + " TEXT," + fbDate + " TEXT)"");
```

The database tables for fbUploadsTable have columns as fbUniqueID, fbFilePath, fbMessage and fbDate which stores the respective information. It also stores the information in other tables similarly.

2. Noticeboard Receiver:
The noticeboard receiver extends to the broadcast receiver. Broadcast receiver is an android component that runs in the background to receive any new data such as new emails. It uses a method email manager to connect to the Gmail server using SMTP and IMAP protocols to get the new emails. A method initprotocol() is created to establish a session with the Gmail server. getInboxMails() is used to store the imap session in store, which is a java data type. Inbox is then retrieved after the connection is established. Notification manager uses the notification service to notify new emails. Inbox mails are stored in the inbox tab by using store.getFolder(inbox).

3. SMTP and IMAP for Gmail:

IMAP and SMTP use standard Simple Authentication and Security Layer, via native IMAP AUTHENTICATE and SMTP AUTH commands, to authenticate users. The incoming connection for IMAP use port 993 and SMTP use port 465 for outgoing connections. IMAP and SMTP protocols are used via secure sockets layer (SSL).

4. Access Tokens for social network sessions:

Social networks use their own API for login authorization. Hence, each session for the social networks has an access token and session expire time. Session expire time is set to 0, which means the session does not expire until logout is done at that particular social network. When the user logs out, the access token is cleared.

5. Bitmap for file attachments:
The pictures on the phone use a lot of memory. It becomes difficult to set the image on the phone in the particular space, hence Bitmap is used to decode the file path and set the scale to a required size.

3.6 Class Diagrams

Figure 3.5 shows the class diagram for the OneNote application main activity.
Figure 3.5: Class diagram for Application Activity.
4. SYSTEM IMPLEMENTATION

4.1 Android Components

The SDKs of each of the social networks such as Facebook, Twitter and LinkedIn are used to develop the interface for these apps. These apps are all integrated to one single interface. The SQLite database is used to store the usernames and passwords of the apps temporarily. The notifications manager is used to provide new notification whenever the account gets a new email. Connectivity manager checks the connectivity of the phone and verifies the Internet connection. Broadcast receiver keeps the app running in the background to support the notification manager.

4.2 Requirements

The most important requirement for this system is an Android device with version 4.0 or higher. A developing environment such as eclipse is needed.

4.3 Outcome

The outcome of this project will be OneNote.apk. This file can be downloaded and installed on android devices. It can be distributed via email for testing and usage purposes.

4.4 Android Application

The application can be opened from the home screen. This application implementation can be divided into the following stages:

1. Login
2. Registration
3. Login verification
4. Dashboard
5. Social Networks
6. Settings

![OneNote application splash screen](image)

**Figure 4.1 OneNote application splash screen**

When the user opens the OneNote application, the OnCreate() shows a splash screen while the app establishes a connection with the database and creates tables. So, splash screen activity is run in the foreground until the app data is initialized as shown in Figure 4.1.

```java
if (Utility.haveNetworkConnection((ConnectivityManager) getSystemService(CONNECTIVITY_SERVICE))) {
    Intent intent = new Intent(SplashScreenActivity.this, LoginActivity.class);
```
startActivity(intent);
SplashScreenActivity.this.finish();
}
else
{
showDialog(1);
}

**Figure 4.2** Checking connectivity at splash screen

The app checks for the Internet connection. If the connection is established, startActivity(intent) calls SplashScreenActivity.java class and opens the app, else showDialog() pops up a dialog showing “no Internet connection” as shown in Figure 4.3. Figure 4.2 shows the commands for checking the connectivity.

**Figure 4.3** No Network

1. Login:

This page uses three buttons backButton, signinBtn and signupBtn. OnCreate() is used to open the page with its layout and contents. It uses TextView as titleTV which is set to ‘user login’. This page needs two text editors userNameET and pwdET to enter
username and password respectively. AppDataBaseHelper class is used to verify the username and password.

If the ID of the button pressed is equal to the backBtnID, the login activity closes and goes to the previous page. If the ID of the button pressed is equal to the signinBtn_ID, then the strings userNameET and pwdET are checked with the database via helper which is the object created for AppDatabaseHelper.java class. The phone number registered with the username and password is checked with the useruniquePhnumber. When both the numbers are same, VerificationActivity class is called using Intent and the user will see login verification page.

If the fields are entered incorrectly or not entered completely, showDialog() pops up messages ‘Please enter valid details’ or ‘Username and password should not be empty” respectively. Similarly when the ID of the button pressed is equal to the signupBtn_ID, SignUpScreenActivity class is called. Figure 4.4 shows the login screen with the buttons, title and text editors.

2. Registration:

OnCreate() opens the registration screen with backButton, signupBtn, titleTV. It also has strings nameET, passwordET, emailET, phnoET, confirmET as text fields. These are used to enter the user details. The details entered are stored in AppDataBaseHelper class via object helper. Figure 4.6 shows the registration screen.

```java
if(isEmailValid(email))
{
    if(!helper.checkUserSignUp(phno))
    {
        helper addUserDetails(new UserDetails(name,pwd,email,phno,""));
        showDialog(5);
    }
}
```
The user details entered are checked for the conditions shown in Figure 4.5 and 
showDialog() is used to display the messages accordingly.

showDialog(1) – ‘Please enter all details’
showDialog(2) – ‘Password and confirm password should be matched’
showDialog(3) – ‘Enter valid email id’
showDialog(4) – you already registered’
showDialog(5) – ‘you successfully signed up’

When showDialog(5) appears, user presses OK button and OnClick() will dismiss the 
alert and closes the SignUpScreenActivity.
3. Login Verification:

OnCreate() launches the page with backButton, verifyBtn, titleTV title and verifyET text editor. verifyEV has the 4-digit random code. When the id of the button pressed is equal to verifyBtnID, the condition verifyText = randomNumber is checked. If true, MainDashBoardActivity class is called. If false, showDialog() displays the message ‘Please enter valid details’.

```java
Random r = new Random();
List<Integer> codes = new ArrayList<Integer>();
for (int i = 0; i < 10; i++)
{
    int x = r.nextInt(9999);
    while (codes.contains(x))
    {
        x = r.nextInt(9999);
    }
    codes.add(x);
}
```
Figure 4.7 Random number generation.

The random 4-digit code is generated by using the commands shown in Figure 4.7 and is sent to the phone number used at signup. Figure 4.8 shows the login verification screen.

4. Dashboard:

`OnCreate()` launches the dashboard with `backButton`, `logoutBtn`, `gmailBtn`, `twitterbtn`, `linkedInBtn`, `facebookBtn`, `settingsBtn`, `titleTV`. The `titleTV` for this screen is Dashboard.

If the ID of the button pressed is equal to the `gmailBtnID`, then the app checks for connectivity and calls the `GmailLoginActivity` class. If no connectivity, `showDialog()` shows the ‘No internet connection’ message. Other social networks are selected similarly.

If the user selects logout button, `showDialog()` shows the dialog ‘Do you want to
logout’ with two buttons yes and no. If yes, app is logged out and the session details in database are cleared with the help of userUniquePhNumber. If no, then the alert is dismissed and the app goes back to dashboard. Figure 4.9 shows the dashboard screen.

![Dashboard screen](image)

**Figure 4.9 Dashboard screen.**

The user can choose one of the options from the dashboard and perform the required activity on the application.

5. Social Networks:

Facebook developers site provides the APPID. This APP_ID is given to the facebook object. With the Facebook object, methods required can be called from Facebook API. Figure 4.10 shows the Facebook login screen.
Figure 4.10 Facebook login.

```java
mPrefs = getPreferences(MODE_PRIVATE);
String access_token = mPrefs.getString("access_token", null);
long expires = mPrefs.getLong("access_expires", 0);
```

Figure 4.11 Access token for Facebook login.

Every session will have an access token. We can get the access token from the
getAccessToken() method. The access token will be stored in to the shared preferences
file. Every session will have an expire time. We can get the expire time from the
getAccessExpires() method. The access token will be stored in to the shared preferences
file. The session can be restored from the access token and expire time. These can be
retrieved from the stored shared preferences file as shown in Figure 4.11.
OnClickListener is set for the logout button, which monitors the user input. facebook.logout(this) method is used to logout from Facebook account. The access token and access expire time are cleared from preferences file.

```java
String strUserProfilePic = "http://graph.facebook.com/" + strId + "/picture?type=large";
imageLoader.displayImage(strUserProfilePic.replace(" ", "%20"), ivProfile, options);
```

**Figure 4.13 Load the user profile picture.**

The code in Figure 4.13 is used to load the user profile picture to the image view using imageLoader. Graph API is used to get data in and out of Facebook. Figure 4.12 shows the User profile.
In the Figure 4.14, there is an image button, edit text and share button. OnClickListeners are set for image button and share button. Text entered in the edit text is retrieved using the getText() method. Path of the image is stored in a variable and the image is converted into bitmap format and then compressed. Using the access token, the message with an image is posted on to Facebook. The post is also saved into the AppDataBaseHelper() against the userUniquePhnumber; which would be displayed in recent posts tab as shown in Figure 4.15. The getFacebookUploads() method is used to get the recently made posts to the recent posts tab.

The consumer key and the secret key are provided by the twitter developer website. ConnectivityManager checks for the network connection. Consumer key and secret key are passed to twitt() object, which are then passed to the TwitterApp object. Twitter API is used to authorize the given credentials. Figure 4.17 shows the twitter share screen.
The text entered in the text editor is passed to twitt.share() method when the network is available. Otherwise, a message “No NetworkConnection Available” is toasted.

```java
File source = new File(sourceImagePath);
File dir = new File(sdCard.getAbsolutePath() + "/TwitterPictures/", source.getName());
if (source.exists()) {
    FileChannel src = new FileInputStream(source).getChannel();
    FileChannel dst = new FileOutputStream(dir).getChannel();
    dst.transferFrom(src, 0, src.size());
    src.close();
    dst.close();
}
```

**Figure 4.15 Facebook recent posts.**

**Figure 4.16 Twitter file attachment**
The file path for attachment is obtained from the code in Figure 4.16 based on user selection. The path is then given to BitmapFactory. Using the bitmap, image can be set to the image view. Twitter recent posts are obtained from the database using the userUniquePhoneNumber.

For Gmail, the username and password are retrieved from the edit text and given to the EmailManager. After checking the username and password with EmailManager, GMailDashBoardActivity class is called. The username and password for Gmail are saved to helper object. The login details are saved in the helper object against the UserUniquePhoneNumber. Figure 4.18 shows the compose email screen.
User profile information is obtained from GmailProfileActivity using the data passed to it from the GmailDashBoardActivity.

For composing email, username and password are obtained from helper object. These details are given to the SMTP authenticator. SMTP creates an instance of a session. Using MimeMessage, it delivers the email to the recipient. EmailManager provides the username, password and urlserver to the EmailAuthenticator, which checks the credentials. IMAP session is created to retrieve inbox. Figure 4.19 shows the inbox and delete email screen.

The LinkedIn API provides authorization. LinkedIn consumer key and secret key are used to communicate with LinkedIn API, similar to Twitter. Each session has an access token, which are obtained after the authorization. Profile information is obtained as follows.
Text from the edit text is retrieved. Using the OAuthConsumer with the help of LinkedIn consumer key, secret key and access token, for a successful authentication, text is posted using an http client as shown in Figure 4.20. The posts are then saved in the helper object against the userUniquePhnumber.

```java
OAuthConsumer consumer = new CommonsHttpOAuthConsumer(
    Config.LINKEDIN_CONSUMER_KEY,
    Config.LINKEDIN_CONSUMER_SECRET);
consumer.setTokenWithSecret(accessToken.getToken(),
    accessToken.getTokenSecret();
DefaultHttpClient httpclient = new
DefaultHttpClient();
HttpPost post = new HttpPost("https://api.linkedin.com/v1/people/~/shares");
```
6. Settings:

The settings screen has options such as turning email notifications on and off, changing the user password and screen view options. Figure 4.21 shows the settings screen.

All the changes made in the settings are saved to the helper object. Notification Service can be turned on and off from the mail updates. The setNotificationReciever() object is used to turn the updates on and off. Change in the password is updated in the helper via edit text.

![Settings screen](image)

**Figure 4.21 Settings screen.**

5. TESTING
The app has to be downloaded and installed on the Android mobile device.

Once the app is installed it can be tested for the following phases:

1. Registration
2. Login
3. Login verification
4. Social networks
5. Settings

5.1. Registration

The testing of registration phase has two test cases. The two test cases would be:

I. Unsuccessful Registration.
II. Successful Registration.

The app will provide slots to enter personal information such as username, password, confirm password, email and phone number during registration. The user has to enter all the details and then touch the sign up button. In case of absence of one of those details, the app will give a message showing ‘all fields are mandatory’ as in Figure 5.1(a); which means an unsuccessful registration.
When the user enters all the fields correctly, the app will display a message saying ‘you successfully signed up’ as in Figure 5.1(b); which means successful registration.

5.2. Login

The testing of the login phase is also divided into 2 test cases. The two cases would be:

I. Unsuccessful Login

II. Successful Login

When the registration phase is successful, the user will have a username and a password with which they can login to the app. When the user enters the credentials
incorrectly, the app will show a message saying ‘Please enter valid details’ as shown in Figure 5.2(a); which means the login was unsuccessful.

When the user enters the credentials correctly, the app will show the login verification page as in Figure 5.2(b); which means the login was successful in the first step. When the login is successful, the app will take the user to the second step of verification, which is called the login verification phase.

![Unsuccessful login](image1.png) ![Successful login](image2.png)

**Figure 5.2(a) Unsuccessful login 5.2(b) Successful login**

### 5.3. Login verification

The testing of login verification phase is also divided into two test cases. The test cases for login verification are:

I. Unsuccessful Login Verification

II. Successful Login Verification
When the user enters the correct credentials, the app will show the login verification page. At the same time, it also generates a random 4-digit code and sends it to the user via text message. The user has to enter the latest 4-digit code to enter the app. When the user enters the code incorrectly, the app will show the message saying ‘Please enter valid details’ as shown in figure 5.3(a); which means an unsuccessful login verification.

![Unsuccessful login verification](image1)

**Figure 5.3(a) Unsuccessful login verification**

When the user enters the correct 4-digit code, the app will show the dashboard as shown in Figure 5.3(b); which means the login verification was successful.

5.4. Social networks

Each of the social networks has its own testing cases. The social network phase has two test cases:

I. Unsuccessful Login

II. Successful login
When the user enters incorrect credentials, the app will show a message as shown in Figure 5.4(a); which means the login was unsuccessful.

When the user enters correct credentials, the app will show the Facebook page with different options as shown in Figure 5.4(b); which means the login was successful.

![Figure 5.4(a) Unsuccessful Facebook login. 5.4(b) Successful Facebook login.](image)

5.5. Settings

The settings phase has two test cases:

I. Change Password.

II. Screen View.

The change password phase is where the user can change the existing password to a new password by entering the details as shown in Figure 5.9.

The screen view phase is where the user can change the view of the screen from portrait mode to landscape mode and vice versa as shown in Figure 5.10.
Figure 5.5 Change password

Figure 5.6 Landscape screen view
6. CONCLUSION AND FUTURE WORK

6.1 Conclusion

The OneNote application is unique and it provides security via 2-step verification to the social networking apps. It also provides a single interface to the selected apps. SQLite database is used to store user and other details temporarily. The login to OneNote application will also login all the apps and vice versa. Generating a text message with a random 4-digit code provides the 2-step verification. This code will provide application access to the user. The user is able to post to Social networks. Users can also compose mails, check inboxes and reply to messages. The settings screen will provide other user compatibility options such as screen view, turning on notifications and changing the password. This app will need the user to signup first with his username phone number and email. Each user will have a userUniquePhnumber. The phone number cannot be used twice.

6.2 Future work

Number of social networks and emails can be increased for much wider usage. This will increase the number of users, as the users will have greater choice of social networks and emails. More attractive and sophisticated user interface can be developed to make the app look appealing. Adding animations and screen effects at the change of screens will make the app more attractive. Better user interface will attract more users. Number of features on the social networks can be increased such as chat, view friend profile and view the posts of other users. This will make it easy for the user to manage his social networks from single app instead of downloading individual apps. Fingerprint for second
step of verification instead of the 4-digit code to improve the speed of operation. The user will not have to open text messages to see the new 4-digit code each time. As of now, the android phones do not have the screens that can take fingerprint inputs. Shake the phone feature to logout, so peers will find it difficult to know that the user is logging out. The motion sensors could be used to implement this feature.
1 Bibliography


