TKR COMMUNITY

[1]Mr B Narsingam, [2]B.Sumanth, [3]D.Ganesh Chandu, [4]K.Sandeep

[1]Assistant Professor, Dept.of CSE, Teegala Krishna Reddy Engineering College, Meerpet, Hyderabad,

tkrcsebnarsingam@gmail.com

[2]BTech student, Dept.of CSE, Teegala Krishna Reddy Engineering College, Meerpet, Hyderabad,
sumanthbelladhi@gmail.com

[3]BTech student, Dept.of CSE, Teegala Krishna Reddy Engineering College, Meerpet, Hyderabad,
ganeshchandu29@gmail.com

[4]BTech student, Dept.of CSE, Teegala Krishna Reddy Engineering College, Meerpet, Hyderabad, :
sandeepkotakonda4648@gmail.com

Abstract*: This is an application where people who are recognized as a part of TKR institute can access to the application. This is an instant messaging and social network platform, which allows users to register and connect to one another for easy communication within campus. WhatsApp social network enables sharing of different-type data and resource, ensuring effective communication and interaction, and even the creating joint activities within created groups. With this high level of communication and interaction environment, it is seen that higher education institutions have started to adopt mobile technologies to meet the needs and expectations of students. But it had a bad impact on students like: Students, who are dependent on instant messaging, may be unsuccessful in academic processes. Negative results such as inability to concentrate, reduced learning skills and productivity may occur in students, who are deprived of sleep due to active WhatsApp use. Receiving too many messages over WhatsApp may negatively affect learning. Active topic may be easily left out of account, because of unnecessary comments in WhatsApp environment. So, an educational institute must not encourage students to use such platforms.*

Keywords*: WhatsApp social network, Negative results, communication and interaction environment.*

I. INTRODUCTION

The use of social media is increased with the increase in population. In recent years, Chat applications have improved and made substantial improvements to the social media due to its distinctive characteristics, which attract audiences. It offers real-time messages and provides various services like text, images, data, etc. In addition, cross platforms including Android and iOS are supported. There are now 100 million mobile users who use monthly chat applications. Security is of utmost importance in chat applications but few take this seriously. It is very important to make more secure application for better communication. In, today's time where most of the chat application use Transport layer protocol for security. It is difficult to ensure that the data is secured. As, the service provider has the full access to the entire message sent and received by their user.

II. LITERATURE SURVEY

The development and design of the student management system based on the network environment

The paper discusses the method of the management information in higher education. On the basis of a comprehensive investigation and analysis on the student management in higher education, we establish the models of the college students' management information by adopting the advanced information technology, and construct the student management information platform. Moreover, we analyze the characteristics of the information management in higher education, and elaborate the methods to solve the difficulties confronting in the students management of the higher education. Finally, the key method and technology to carry out the information management platform are presented.

Android-based Attendance Management System

In this paper a method of taking attendance by employing an application running on the Android platform is proposed in this paper. This application, once installed can be used to download the students list from a designated web server. Based on the downloaded list of students, the device will then act like a scanner to scan each of the student cards one by one to confirm and verify the student's presence. The devices camera will be used as a sensor that will read the barcode printed on the students' cards. The updated attendance list is then uploaded to an online database and can also be saved as a file to be transferred to a PC later on. This system will help to eliminate the current problems, while also promoting a paperless environment at the same time. Since this application can be deployed on lecturers own existing Android devices, no additional hardware cost is required.

The Designment of Student Information Management System Based on B/S Architecture

This paper uses the B/S structure to design the student information management system, and explains the system design principle, system plan and structure, the function module of information system according to current university student information management needs. It provides an interactive students management platform for the information of a large number of students and the management of students.

Research and Implementation of Web Services in Android Network Communication Framework Volley

This paper includes combination of Web Services and mobile devices will promote the development of mobile applications. Volley framework Google 2013 proposed has the advantages of convenient use and network request faster, but it does not support Web Services. Extension of Volley, to support the Web Services, which ca n facilitate the Web Services application development, but also can improves the access performance of Web Services. On the basis of analysis and research of the Volley, Ksoap2 and Java Web Services, through the implementation of the Http Stack interface and the expansion of JSON Object Request to realize support for Web Services. The scheme uses JSON format to transfer data, support S SL/TLS protocol requests, custom parameter, sets or gets the request header. This scheme is good compatibility, easy to use, suitable for application on Android platform.

III. PROPOSED SYSTEM

In the proposed system we are developing an application, which is accessible to recognized members of TKR Institution we can use the identity-based AES Encryption Algorithm to provide the data security, it can handle the collusion attack and it can control efficient access control on the data we are sharing

MODULES:

- Registration: The user has to register him into the system to generate the Unique Id.

- Login: The user has to login into the system providing his username and password keeping all the data secure.

- Inbox: The user gets to see a list of messages which were encrypted and only sent to him i.e. to his account. The Inbox will only have the senders name and date/time.

- Message: The user is allowed to view the complete message which is already decrypted as he selects the messages from the Inbox.

• Group Messages: As the recipient’s Id is difficult to keep in mind the system allows the user to assign group.

• Send Message: The user can send messages which will be encrypted once he sends it, here the user should add mobile number and the receipts id or he can make use of the data saved in favorites.

• Sent: The user can view the messages that he has sent i.e. only the encrypted ones in the sent folder.

SYSTEM ARCHITECTURE

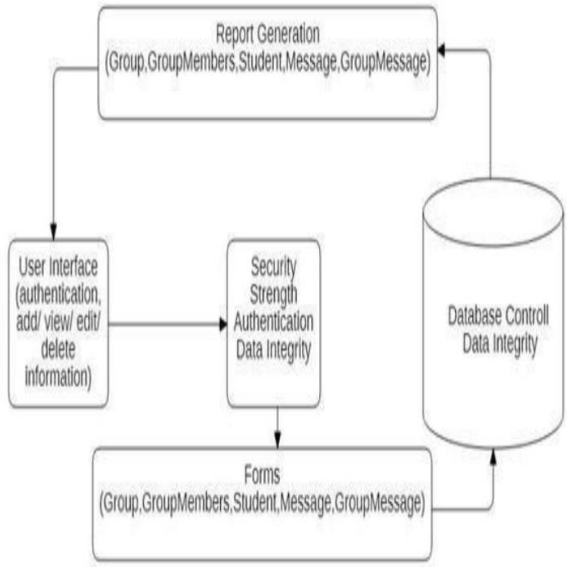

Report Generation
(Group,GroupMembers,Student,Message,GroupMessage)
User Interface (authentication, add/ view/ edit/ delete information)
Security Strength Authentication Data Integrity
Database Controll Data Integrity
Forms
(Group,GroupMembers,Student,Message,GroupMessage)

Fig.1 Proposed architecture

IV. IMPLEMENTATION

Android Technical Environment

Android apps are written in the Java programming language. The Android SDK tools compile your code—along with any data and resource files—into an APK: an Android package, which is an archive file with an .apk suffix. One APK file contains all the contents of an Android app and is the file that Android-powered devices use to install the app.

Once installed on a device, each Android app lives in its own security sandbox: The Android operating system is a multi-user Linux system in which each app is a different user.

By default, the system assigns each app a unique Linux user ID (the ID is used only by the system and is unknown to the app). The system sets permissions for all the files in an app so that only the user ID assigned to that app can access them. Each process has its own virtual machine (VM), so an app's code runs in isolation from other apps.

By default, every app runs in its own Linux process. Android starts the process when any of the app's components need to be executed, then shuts down the process when it's no longer needed or when the system must recover memory for other apps.

In this way, the Android system implements the principle of least privilege. That is, each app, by default, has access only to the components that it requires to do its work and no more. This creates a very secure environment in which an app cannot access parts of the system for which it is not given permission.

The manifest file in which you declare components and required device features for your app. Resources that are separate from the app code and allow your app to gracefully optimize its behavior for a variety of device configurations.

App components are the essential building blocks of an Android app. Each component is a different point through which the system can enter your app. Not all components are actual entry points for the user and some depend on each other, but each one exists as its own entity and plays a specific role—each one is a unique building block that helps define your app's overall behavior. There are four different types of app components. Each type serves a distinct purpose and has a distinct lifecycle that defines how the component is created and destroyed. Here are the four types of app components.

ACTIVITIES

An activity represents a single screen with a user interface. For example, an email app might have one activity that shows a list of new emails, another activity to compose an email, and another activity for reading emails. Although the activities work together to form a cohesive user experience in the email app, each one is independent of the others. As such, a different app can start any one of these activities (if the email app allows it). For example, a camera app can start the activity in the email app that composes new mail, in order for the user to share a picture.

An activity is implemented as a subclass of Activity and you can learn more about it in the Activities developer guide.

SERVICES

A service is a component that runs in the background to perform long-running operations or to perform work for remote processes. A service does not provide a user interface. For example, a service might play music in the background while the user is in a different app, or it might fetch data over the network without blocking user interaction with an activity. Another component, such as an activity, can start the service and let it run or bind to it in order to interact with it. A service is implemented as a subclass of Service and you can learn more about it in the Services developer guide.

CONTENT PROVIDERS

A content provider manages a shared set of app data. You can store the data in the file system, an SQLite database, on the web, or any other persistent storage location your app can access. Through the content provider, other apps can query or even modify the data (if the content provider allows it). For example, the Android system provides a content provider that manages the user's contact information. As such, any app with the proper permissions can query part of the content provider (such as Contacts Contract. Data) to read and write information about a particular person.

BROADCAST RECEIVERS

A broadcast receiver is a component that responds to system-wide broadcast announcements. Many broadcasts originate from the system—for example, a broadcast announcing that the screen has turned off, the battery is low, or a picture was captured. Apps can also initiate broadcasts—for example, to let other apps know that some data has been downloaded to the device and is available for them to use. Although broadcast receivers don't display a user interface, they may create a status bar notification to alert the user when a broadcast event occurs. More commonly, though, a broadcast receiver is just a "gateway" to other components and is intended to do a very minimal amount of work. For instance, it might initiate a service to perform some work based on the event.

ACTIVATING COMPONENTS

Three of the four component types—activities, services, and broadcast receivers—are activated by an asynchronous message called an intent. Intents bind individual components to each other at runtime (you can think of them as the messengers that request an action from other components), whether the component belongs to your app or another.

An intent is created with an Intent object, which defines a message to activate either a specific component or a specific type of component—an intent can be either explicit or implicit, respectively.

THE MANIFEST FILE

Before the Android system can start an app component, the system must know that the component exists by reading the app's AndroidManifest.xml file (the "manifest" file). Your app must declare all its components in this file, which must be at the root of the app project directory. The manifest does a number of things in addition to declaring the app's components, such as: Identify any user permissions the app requires, such as Internet access or read-access to the user's contacts. Declare the minimum API Level required by the app, based on which APIs the app uses. Declare hardware and software features used or required by the app, such as a camera, Bluetooth services, or a multi touch screen. API libraries the app needs to be linked against (other than the Android framework APIs), such as the Google Maps library and more Declaring components.

DECLARING APP REQUIREMENTS

There are a variety of devices powered by Android and not all of them provide the same features and capabilities. In order to prevent your app from being installed on devices that lack features needed by your app, it's important that you clearly define a profile for the types of devices your app supports by declaring device and software requirements in your manifest file. Most of these declarations are informational only and the system does not read them, but external services such as Google Play do read them in order to provide filtering for users when they search for apps from their device.

APP RESOURCES

An Android app is composed of more than just code—it requires resources that are separate from the source code, such as images, audio files, and anything relating to the visual presentation of the app. For example, you should define animations, menus, styles, colors, and the layout of activity user interfaces with XML files. Using app resources makes it easy to update various characteristics of your app without modifying code and—by providing sets of alternative resources—enables you to optimize your app for a variety of device configurations (such as different languages and screen sizes). For every resource that you include in your Android project, the SDK build tools define a unique integer ID, which you can use to reference the resource from your app code or from other resources defined in XML. For example, if your app contains an image file named logo.png (saved in the res/drawable/ directory), the SDK tools generate a resource ID named R.drawable. Logo, which you can use to reference the image and insert it in your user interface

V. RESULTS

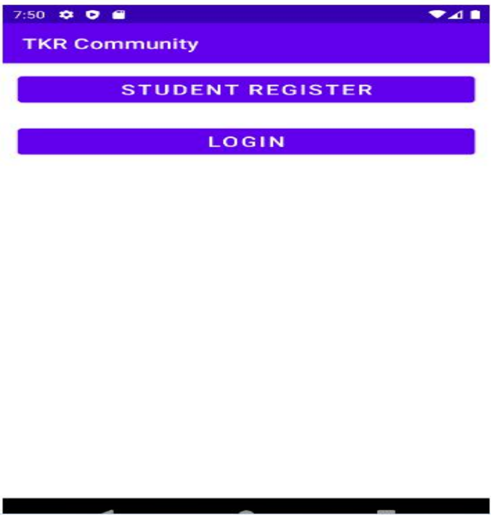
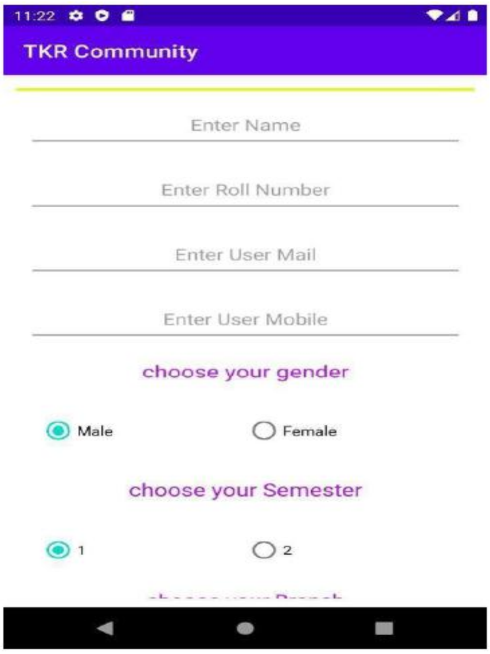


Fig.2 Homepage



11:22

TKR Community

Enter Name

Enter Roll Number

Enter User Mail

Enter User Mobile

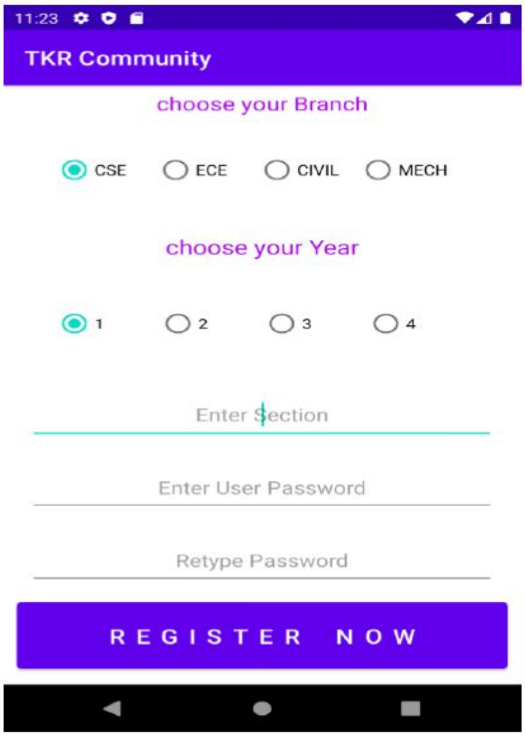
choose your gender

● Male ○ Female

choose your Semester

● 1 ○ 2

Fig.3 Registration



11:23

TKR Community

choose your Branch

● CSE ○ ECE ○ CIVIL ○ MECH

choose your Year

● 1 ○ 2 ○ 3 ○ 4

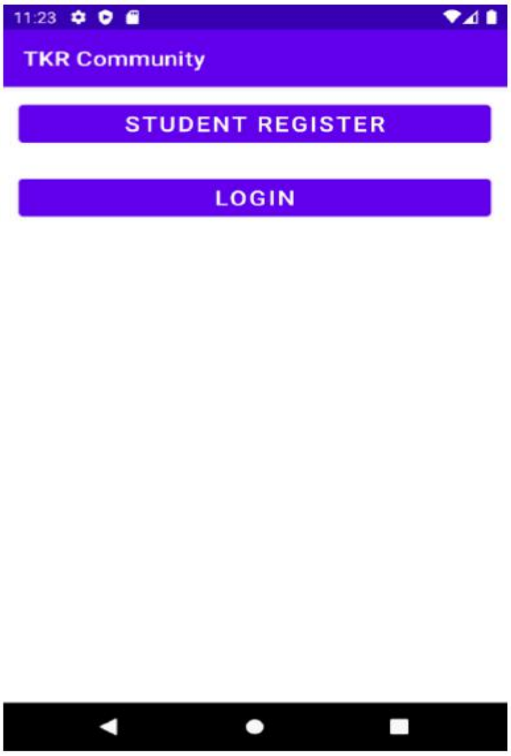
Enter Section

Enter User Password

Retype Password

REGISTER NOW

Fig.4 Registration



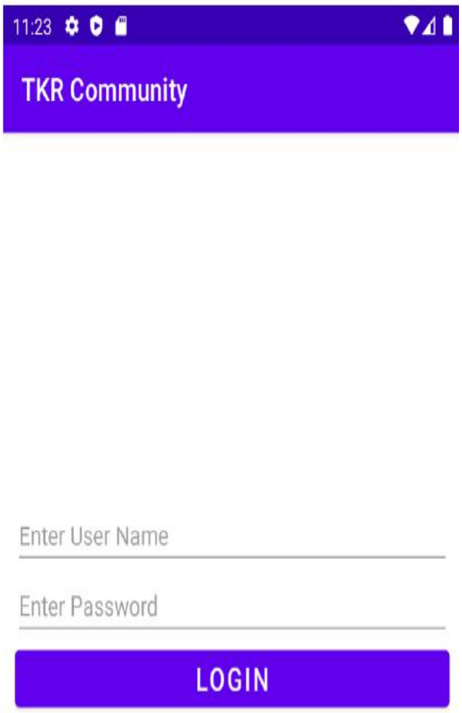
11:23

TKR Community

STUDENT REGISTER

LOGIN

Fig.5 APP Interface



11:23

TKR Community

Enter User Name

Enter Password

LOGIN

Fig.6 Login page

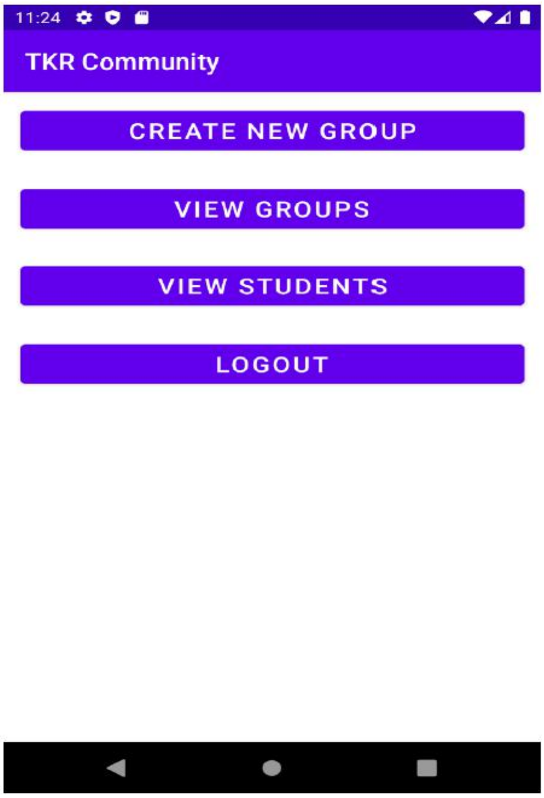


Fig.7 Admin Interface

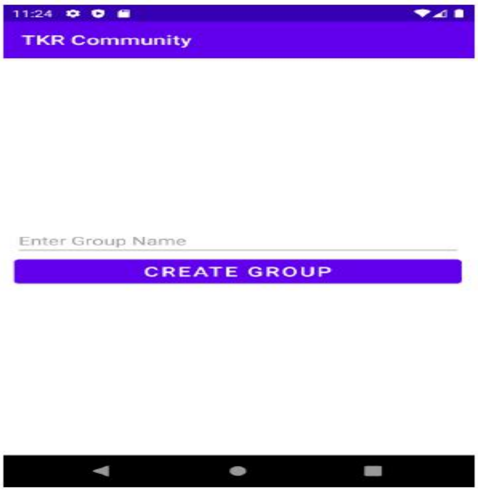
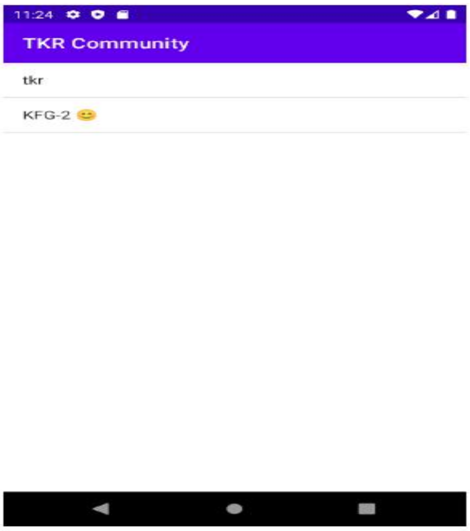


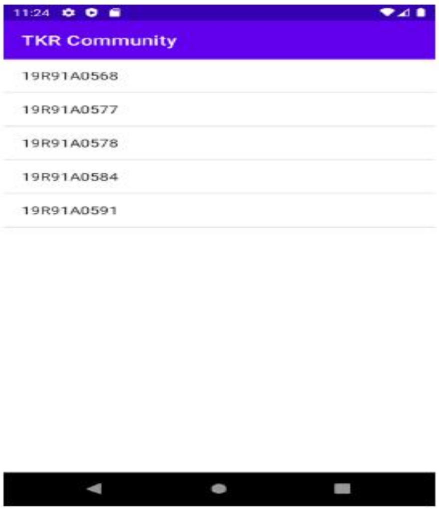
Fig.8 Create group



tkr

KFG-2 😊

Fig.9 Groups



19R91A0568

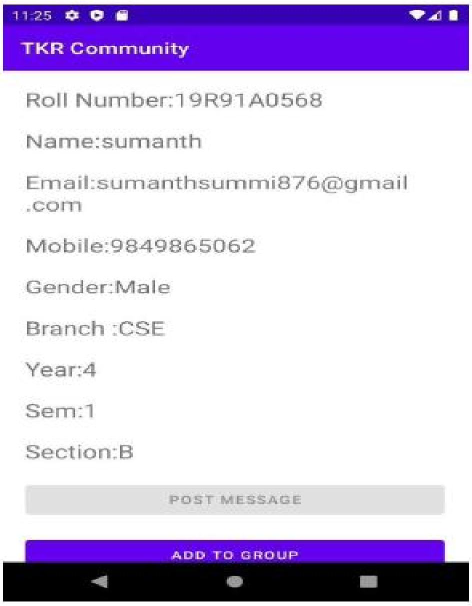
19R91A0577

19R91A0578

19R91A0584

19R91A0591

Fig.10 Roll Numbers of students



Roll Number:19R91A0568

Name:sumanth

Email:sumanthsummi876@gmail.com

Mobile:9849865062

Gender:Male

Branch :CSE

Year:4

Sem:1

Section:B

Fig.11 Student Detials

VI. CONCLUSION

This android application 'TKR COMMUNITY' will provide features that will connect students & teachers. This is an instant messaging and social network platform, which allows users to register and connect to one another for easy communication within campus .The main aim of this application is to be independent from online social media's which is making a bad impact on student's concentration.

REFERENCES

1. https://developer.android.com/docs(Official Google Documentation for app developers) • https://stackoverflow.com/ (Reference for handling errors)
2. https://gits-cm.com/book/en/v1/Getting-Started-AboutVersion-Control(Reference for understating Version Control tool Git.)
3. https://firebase.google.com/docs(Reference for Google Firebase)
4. https://gradle.org (Gradle build tool)
5. https://www.tutorialspoint.com/android/index.htm(Android Tutorial)
6. http://www.flexworkresearch.org/uploads/publication/document/4957/LaborMarketInstitutionsLiteratureReview.pdf&sa=D&source=hangouts&ust=1572775231341000&usg= FQjCNFeIddlSov5fDLZKJmNOqGjmScwDQ(UNReport2013) https://www.iza.org/publications/dp/4890/behind-thelighthouse-effect. (J30)
7. Prasadu Peddi (2019), "Data Pull out and facts unearthing in biological Databases", International Journal of Techno-Engineering, Vol. 11, issue 1, pp: 25-32.