A Minor Project on

ChaTeX: A LaTeX Based Math Chat Application

Submitted in Partial Fulfillment of the

Requirements for the Degree of **Bachelor's**

Degree of Engineering in Computer

Engineering under Pokhara University

Submitted by:

Barsha Pokharel, 15307

Bikalpa Dhakal, 15395

Date:

14 Dec 2018

Department of Computer Engineering

NEPAL COLLEGE OF

INFORMATION TECHNOLOGY

Balkumari, Lalitpur, Nepal

ACKNOWLEDGEMENT

We express our deepest gratitude to Mr. Madan Kadariya whose suggestions and recommendations were extremely useful in figuring out the limitations of our application and making the application more user friendly He aided us significantly in the documentation phase, without which the completion of this project would be an uphill task.

We extend our sincere thanks to the college administration, respective departments and teachers for providing us with the healthy environment for the better growth of the project. We thank our parents for their continuous support and encouragement during the entire period of our project. Finally, we would like to thank our colleague for comprehending and supporting us for the completion of our project in limited time.

Barsha Pokharel (15307)
Bikalpa Dhakal (15395)

Nepal College of Information Technology

Balkumari, Lalitpur.

December 2018

ABSTRACT

ChaTeX is an android instant messaging application which have capability of rendering text message typeset in LaTeX. LaTeX is a documentation preparation system and typesetting of mathematical equations in plain text is one of its prominent features. Although there are many Instant Messaging apps available today, there doesn't exist a single mobile application, till this day that let users send LaTeX typeset text and render it. Even the web based LaTeX chat applications don't provide robust and secure mechanism of chatting because they create public chat rooms.

ChaTeX uses MathJax engine and several open source libraries to render LaTeX typeset text and Google Firebase for managing database, storage and authentication. In comparison to the existing web applications for the same purpose, ChaTeX distinguishes itself by use of friend request and user profile feature. The app aims to eradicate the trouble of having to write handwritten expressions and sending their pictures among students and scholars. After the successful completion of the project, it has been concluded that ChaTeX will be an inevitable tool for students in the field of science and mathematics.

Keywords: ChaTex, LaTeX, Android, Firebase, Instant Messaging, MathJax

TABLE OF CONTENTS

ABBREVIATIONS

API	Application Program Interface
CSS	Cascading Style Sheet
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
I/O	Input / Output
IDE	Integrated Development Environment
IM	Instant Messaging
IPC	Inter Process Communication
JDK	Java Development Kit
JSON	JavaScript Object Notation
MathML	Mathematical Markup Language
NOSQL	Not Only Structured Query Language
OpenGL | ES	Open Graphics Library for Embedded Systems
OS	Operating System
SDLC	Software Development Life Cycle
SGL	Scalable Graphics Library
SSL	Secure Socket Layer
UI	User Interface
URI	Uniform Resource Indicator
URL	Uniform Resource Locator
Wi-Fi	Wireless Fidelity
XML	Extensive Markup Language

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

ChaTeX is an android chat (Instant Messaging) application with the capability of rendering and displaying equations typeset in a popular typesetting system called LaTeX. This section aims to provide essential information about the needs, objectives, scope and limitations of ChaTeX.

1.1. PROBLEM STATEMENT

With its increasing scope and availability, internet is now one of the most used resources by the students. One feature of internet that almost everyone uses today is the instant messaging. Students frequently use chat applications for interacting among one another, asking questions, answering them, sending solutions and many more.

One of the problems students frequently encounter while chatting is the trouble of sending mathematical equations in their correct form. Although the commonly available chat apps like Facebook Messenger, Viber, WhatsApp, etc. allow users to communicate with text as well as multimedia messages, they don't fare good when the subject of the chat turns to mathematics. Students often share mathematical equations via pictures of handwritten equations. Sending pictures not only require more cost than sending text in terms of data bandwidth, but also adds overhead of image processing at both sender and receiver side.

There are a few websites in the internet which do provide the feature of chatting in LaTeX, but not a single reliable app exists in android world to do the same functionality till date. Those websites need users to have browsers installed in their devices and have boring UI. Moreover, they create public chat rooms so that anyone who knows the name of chat room enter the room and can access messages, which isn't so secure. ChaTeX aims to solve that problem by allowing users to interchange text messages along with mathematical equations typeset in LaTeX between explicitly registered users in the app. The LaTeX equations are rendered only when needed on both sender and receiver device and otherwise are transferred in plain text form with a special syntax, which makes the chat synchronization faster and smooth.

1.2. PROJECT OBJECTIVES

The following are the core objectives of this project:

1. To provide the users with the most practicable and effective solution to the problem of sending/receiving and discussing mathematical equations.
2. To encourage fruitful discussion among the students in the field of science and mathematics making use of the available tools and technologies to communicate, share, help, ask and answer the various problems that arise in the field.

1.3. SIGNIFICANCE OF STUDY

The study aims to find solution to the difficulty in sharing math equations across internet. Mathematical equations are not only confined to pure mathematics, but also appear frequently in every branch of pure and applied science. The development of ChaTeX will provide students an efficient way to discuss and eventually help students in efficient learning. Writing equations in LaTeX makes the equations more readable, understandable, intuitive and beautiful with just the right amount of symbol size and spacing, which also includes a wide range of character sets.

1.4. SCOPE AND LIMITATION

The main scope of ChaTeX is to allow users to use mathematical equations in chat more efficiently. The application is explicitly targeted to the students and teachers that frequently have to discuss about the mathematics using instant messaging. Moreover, the application is developed explicitly for the android platform to be run as a native android application. It is required that the user have some basic knowledge about LaTeX for exploiting the full capacity of the application, so the scope of application in present context is not too broad.

The following are the limitations of our application.

1. Users will be able to chat and converse the mathematical expressions, but the app will not focus on any way towards simplifying, error checking – either in logic or in mathematics and solution of the expressions thus typed.

2. As the application is explicitly developed as a native android application, it can't be run in any other platforms like iOS, Web, Windows etc.
3. The app developed at the end of this project will be a standalone application. So, the users will have to explicitly register separate accounts in the application itself. The application in the present scope is not designed as a plugin to another standalone chat application.
4. The application will not be able to share/forward the messages from the application to any other existing chat applications in TeX format because they don't have capability to render TeX typeset equation.

ChaTeX will not implement complex encrypting algorithms for data encryption, but use a simple open source encrypting library.

2. LITERATURE REVIEW

This section consists the literature study on the different topics like Instant Messaging, LaTeX, Firebase etc. It aims to provide readers a theoretical base for the project and also develop an understanding of the nature of the project.

2.1. ANDROID

Android, initially developed by Android Inc. and later acquired by Google in 2007, is an open source mobile operating system developed primarily for touchscreen mobile devices and tablets. Built upon a modified version of the Linux kernel, Android is currently one of the most used mobile operating systems in the world. In addition to the mobile devices, various other devices like televisions, cars, wrist watches, game consoles and digital cameras use Android today. The UI of android is written in Java, while its core and kernels are written in C and C++. Android 9.0 "Pie" is the current version of android, which was released in August 2018[1].

The functionality of the devices running Android can be extended by installation of several custom applications, commonly called as "apps", which are primarily written using Java. The applications run on top of Android Runtime and make use of core libraries set up by the application framework. On the foundation lies the Linux kernel, which acts as a bridge between the underlying hardware and the android library.

2.2. INSTANT MESSAGING

Instant messaging (IM) is an online chat mechanism that allows real-time text transmission between registered users over the Internet[2]. Unlike email messaging, instant messaging requires same set of software installed by all participants, and the transmission of text happens at real-time. Real-time text transfer means that there is no significant delay between sending and receiving message. Most primitive IM applications used push technology that transmitted text character. The latest IM services allow the transmission of not only text messages, but also multimedia messages. Some of the popular chat applications available in internet presently are Facebook Messenger, WeChat, WhatsApp, Line, Viber, Hangouts, etc.

2.2.1. EXISTING SYSTEMS LIKE CHATEX

There exist a few web applications in the internet that allow the users to chat, where the text typeset in LaTeX enclosed inside some special escape characters are automatically rendered in real-time. Two such web chat applications – namely MathIM and hack.chat are reviewed in this section.

2.2.1.1. MATH-IM

MathIM is an open source web chat application written in Scala developed by Tommy C. Li.[3] It uses MathJax engine to render LaTeX. To start chatting in MathIM, the user will browse to the URL https://mathim.com, where the user is asked for the the name of chat room that he/she wants to enter into. The user can enter into an existing chat room if the user already knows the name of the chat room. If new name is entered, however, a new chat room is created. The user will then send the name of the chat room, which obviously is unique for every chat rooms, to the person(s) he/she wants to chat.

Figure 1: User Interface of MathIM[4]

The figure illustrated above shows the basic user interface of MathIM. In MathIM, a user can enter the desired name of chat room that is to be entered to and type the nickname that appears with every chat message. The chat room URL for a chat room

with name ‘chat-room’ will appear as https://mathim.com/chat-room. The problem here is that anyone who knows the chat room name can directly enter the chat room, whether or not intended by the original participants of the chat.

2.2.1.2. HACK.CHAT

Hack.chat is another open source web chat application written by Andrew Belt[5]. The user interface looks very minimal, sometimes even mimicking the console or terminal of a computer. Like MathIM, it also creates chat room URL based on the name of the chat room created. A chat room with name ‘chat-room’ can be created or joined by going to https://hack.chat/?chat-room. It can be felt that even though the user interface of hack.chat is primitive than of MathIM, the rendering of LaTeX is much faster in hack.chat nonetheless.

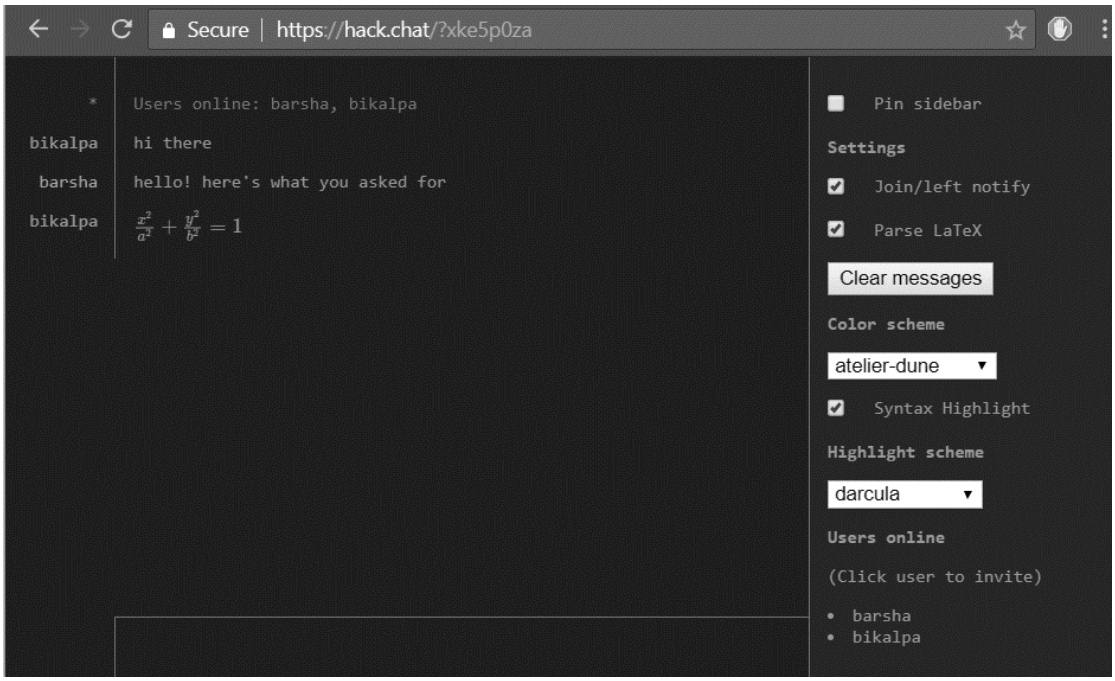


Figure 2: User Interface of hack.chat [6]

HackChat however automatically generates a random string to be used as a slug in the URL, so the chances of name collision is quite not significant. Similarly, more than one user can connect to the same chat channel via the same link used by the creator of the channel.

2.3. LATEX

LaTeX (a shortening of Lamport TeX) is a document preparation system for high-quality typesetting. Initially, LaTeX was used by mathematicians and computer scientists as a writing tool for standard typesetting. Later on, owing to the development of LaTeX itself, scholars started using it to write documents which include non-Latin scripts and complex mathematical expressions.[7]

This application uses the typesetting feature of LaTeX to beautifully represent complex math expressions in text form to equation form. The use of LaTeX typesetting in representing mathematical expressions not only make the equations more readable and intuitive, but also preserves the beauty of different symbols and non-Latin characters used. Mathematical equations typeset in LaTeX basically consists of several tags followed by backward slash '\' which have their own meaning in LaTeX. The whole equation body is often enclosed inside two '$' signs. The following table presents some typical equations written in LaTeX and their corresponding natural form.

Table 1: Some typical math equations in LaTeX and their rendered natural form[8]

LaTex text	**Output equation**
`$E = mc^2$`	$E = mc^2$
`$\frac{x^2}{a^2}+\frac{y^2}{b^2} = 1$`	$\frac{x^2}{a^2}+\frac{y^2}{b^2} = 1$
`$x= \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$`	$x= \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$
`$f(x)=\sum_{0}^{\infty}(a_ncosnx+b_nsinnx)$`	$f(x)=\sum_{0}^{\infty}(a_ncosnx+b_nsinnx)$
`$\Gamma(z)=\int_0^\infty x^{z-1} e^{-x} \, dx$`	$\Gamma(z)=\int_0^\infty x^{z-1} e^{-x} \, dx$

LaTeX has been a de-facto standard in the mathematics typesetting system across the world. It's capability to render the equations more swiftly and correctly has made it a prominent choice for integrating mathematics in web as well as mobile applications.

According to a published article titled "Don't Format Manuscripts" published in the journal "The Scientist" (2009), the usage of LaTeX in some of the field of science and mathematics are summarized below.

Table 2: LaTeX usage statistics in some fields of science[17]

Field	LaTeX rate (% of paper submitted)
Mathematics	96.9
Statistics and probabiltity	89.1
Physics	74.0
Computer Science	45.8
Astronomy and Astrophyics	35.1

Source: François Brischoux, Pierre Legagneux. Don't Format Manuscripts. The Scientist 23, 24 (2009).

It can clearly be seen that almost all of the papers in the fields of science and mathematics are prepared using LaTeX documentation. This clearly justifies the usage of LaTeX as the typesetting system in our application.

2.3.1. KNUTH-PLASS ALGORITHM

The algorithm behind the text wrapping feature of LaTeX is the Knuth-Plass algorithm. This algorithm calculates when a long line should be broken to a new line so that the text as a whole seems uniform.

The paragraph is first divided into entities called boxes, which are the indivisible blocks that can be anything from a syllable to a character. The boxes that can be stretched and shrunken in size are called the glue which most often is the whitespace. The maximum ratio to which the glues are allowed to expand is called stretchability, whereas the

minimum ratio is called shrinkability The optional characters (like hyphen) are called as penalties.

Known to us all these parameters, the minimum width of the line can be calculated as:

$$width_{min} = width_{boxes} + num_{glue} \times width_{glue} \times shrinkability$$

Similarly the maximum width of the line can be calculated as:

$$width_{max} = width_{boxes} + num_{glue} \times width_{glue} \times stretchability$$

These expressions help us to determine where the line break should be. It is desirable that $width_{min} < width < width_{max}$.

2.4. MATHJAX

MathJax is an open source JavaScript library which displays the mathematical notations written in MathML or LaTeX in the web browsers. MathJax uses HTML and CSS with web fonts, instead of using bitmap images, so that the equations easily scale with surrounding text at all zoom level. It first downloads the webpage content, parse through the document for then typeset all the mathematical notations used in that document. The most recent version of MathJax library is MathJax v2.7.[9]

Today, most of the mathematical typesetting in the internet is based on the MathJax Library. The following popular websites and web applications use MathJax for rendering LaTeX.

1. The Annals of Mathematics
2. Scholarpedia
3. KAIST (Department of Mathematical Sciences, SouthKorea)
4. Math Overflow
5. math.stackexchange.com (a popular Q&A platform from the makers of StackOverflow)
6. ProofWiki
7. GeogebraWiki

The processing of a MathJax page involves three basic stages[10]. One of them is the pre-processing and the other two are processing stages. The following three paragraph describe each of these basic stages in brief.

Pre-processing identifies mathematical content on a page (MathML, TeX, different TeX-delimiters etc) and converts it into a standard input format (script-tags). While this pre-processing can be done server-side, it's not a bottleneck and very little performance is gained by optimizing here.

The Input-Jax processes the text input into MathJax's internal format, which is a very fast process. Since the input processors are modular, network latency can create delays as components are loaded as they are needed. This is the core problem of balancing modularity vs network activity and needs to be revisited as network speed and processing power develop.

The final stage of MathJax is the generation of its output either in HTML-CSS or SVG form. MathJax uses the popular Knuth-Plass algorithm for identifying line breaks and other formatting basics.[11] Knuth Plass algorithm is a bottom-up algorithm that determines the dimensions of the children before determining the width and height of the parent.

2.5. KATEX

KaTeX is another fast and easy-to-use Javascript library for rendering TeX math on the web, and is the first alternative to the MathJax library for the same purpose. KaTeX is faster than its other competitors, including MathJax, and is compatible in most of the web browsers used in these days. Also, it does server-side rendering and hence produce same output regardless of browser or environment.

The following table shows the processing times to load Interactive Mathematics'[12] "KaTeX and MathJax comparison demo" page (URL: https://www.intmath.com/cg5/katex-mathjax-comparison.php) stuffed with more than a hundred complex TeX codes to be loaded using both MathJax and KaTeX for different browsers.

Table 3: Comparision of MathJax and KaTeX rendering engines[12]

S.N.	Browser	Time Taken for MathJax	Time Taken for KaTeX
1.	Google Chrome (Windows)	1881 ms	121 ms
2.	Microsoft Edge(Windows)	4403 ms	289 ms

2.6. MATHVIEW

MathView is an open source library for displaying math formula in android apps written by Brian Lee.[13] There are two rendering engines available in the library: MathJax and KaTeX. The library supports android version 4.1 (Jelly Bean) and newer as per its documentation.

MathView is a custom View libaray, whose behavior is similar to to that of the popular TextView. The MathView renders the TeX or MathML code into math formula using one of the available engines. MathView inherited from Android WebView and use javascript (MathJax or KaTeX) to do the rendering.

2.7. GOOGLE FIREBASE

Firebase is a mobile and web application development platform developed by Firebase, Inc. in 2011, then acquired by Google in 2014.[14] Google Firebase is an awesome development platform that provides plethora of services like authentication, real-time database, storage, hosting, remote configuration etc. in an easy way which would otherwise require a lot of headache for app developers. Google firebase allows app developers to focus on app productivity and content quality by providing backend solutions by itself. App developers can simply use Firebase API and tools in their mobile or web applications and use built in features provided by Google.

The table that follows lists some of the major features of Google Firebase that were used in this application.

Table 4: Various features provided by Google Firebase[14]

SN	Product	Function
1.	Firebase Authentication	This feature allows developers to easily register the users of their application and then authenticate whenever they try to login to their app. Firebase currently supports authentication via Phone, Email, Google, Facebook, Twitter, etc.
2.	Firebase Realtime Database	The real-time database feature of Firebase allows developers to write their apps such that they make modifications to database in the real time or in other words, instantly.
3.	Cloud Storage for Firebase	Cloud Storage feature of Firebase allows developers to store various resources and files needed for their applications in the storage provided by Firebase itself. These resources could be images of users, various files needed to be accessed by users, etc.
4.	Cloud Functions for Firebase	This feature allows developers to run backend code without managing the servers. The Firebase functions are single purpose JavaScript functions which are run in secure Node.js runtime environment.
5.	Firebase Cloud Messaging	Firebase Cloud Messaging provides a reliable connection between server and client devices, which allows developers to deliver and receive messages and notifications.

Among the various available options for implementing the backend, Google Firebase was chosen, thanks to the following outstanding advantages:

1. Minimal effort required for installation and integration.
2. Free, and easy-to-use

3. Takes away a lot of overhead from developer's task list, which is implemented by the server itself.
4. Future proof and cross-platform API are available.
5. Firebase provides real time database with NoSQL
6. Easy to use authentication, cloud messaging and notification features.

3. METHODOLOGY

This section describes the different stages involved during the development phase of this chat application.

3.1. SOFTWARE DEVELOPMENT LIFE CYCLE

ChaTeX uses the iterative model of software development life cycle. In this model, a simple and primitive implementation of a very small set of software requirements is done at first, which is followed by the iterative enhancement in the primitive model until all requirements are fulfilled and the software is ready to be deployed. The following sub sections briefly describe various phases in iterative model of SDLC that was applied in development of ChaTeX.

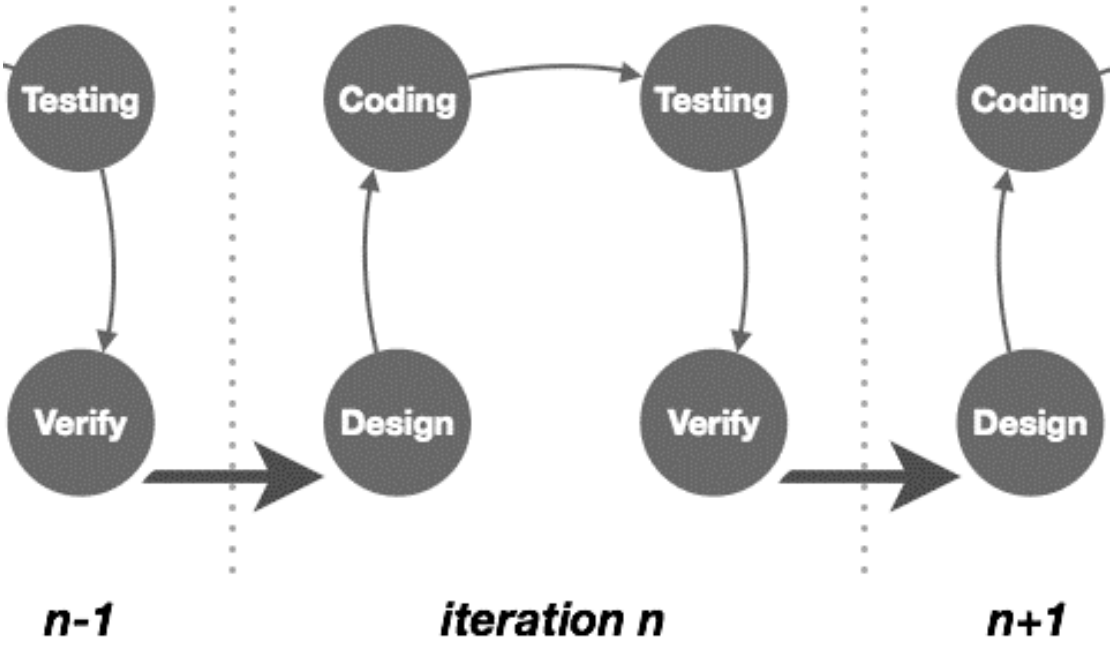


Figure 3: Iterative Model of Software Development Life Cycle. (Source: TutorialsPoint.com, https://www.tutorialspoint.com/software_engineering/software_development_life_cycle.htm)

3.1.1. REQUIREMENT ANALYSIS

In requirement analysis phase, the different requirements that the final product was supposed to fulfill were determined, gathered and analyzed. The various requirements for ChaTeX were user authentication, account registration, avatar and status features, last seen feature, friend request, push notifications, text messaging etc. With successive iteration, this phase also involved the identification of requirements which were not

fulfilled in the previous builds of the software, but are still needed for the development of final product.

3.1.2. DESIGN

In design phase, a software solution to meet the requirements gathered in the previous phase was designed. Design phase was largely comprised of designing firebase database schema, where different child objects were created for storing different aspects of information like users, messages, requests, profile pictures, etc.

3.1.3. IMPLEMENTATION

In implementation phase, the software was coded as per the design pattern mentioned earlier. This phase involved both the creation of new features, plus the upgrade and enhancement of previously coded feature as per the final requirements.

The Appendix I documents the source code written both in server side and client side during the development of ChaTeX.

3.1.4. TESTING AND REVIEW

In this phase, the source code was built and tested in some sample devices to find out any errors (both logical and syntactical) that exist in the application. The developer team tested the app by logging in from arbitrary accounts and sending requests and messages to each other. The primary device used in testing the app was Oneplus2 phone running on Android OS API level 23.

3.2. TOOLS AND TECHNOLOGIES USED

The development of ChaTeX was possible by use of several feature packed enterprise application software and tools which are briefly described in this section.

Google Firebase was chosen as the back-end solution and the front-end UI was constructed by using various open source libraries available in GitHub. The reasons for choosing Google Firebase among various other alternatives as the back-end are enlisted as below:

1. Availability to developers free of cost. The pricing starts only after our application crosses the maximum user limits.
2. Easy user registration and authentication with just a few lines of code.
3. Easy access to real-time database by use of Firebase UI library.
4. Very easy to integrate to an existing android application by use of Firebase Assistant tool in-built in Android Studio.
5. Built-in authentication via Facebook, Google, Twitter, and even phone.
6. Easy Cloud Messaging service to send push notifications to clients.
7. Feature of firebase functions, which can be used to write any server-side functions using Node.js

Table 5: Application software and tools used during development of ChaTeX

SN	**Name**	**Purpose/Description**
1.	Java Development Kit	A collection of libraries and tools required for compilation of code written in Java which is required by Android Studio for compilation of the Java code.
2.	Android Studio Bundle	A feature-pack IDE used for the development of android applications, which was used in writing source code, building and testing of the application.
3.	Google Firebase	A free platform that provides features like authentication, real-time database, cloud messaging, server-side functions, cloud messaging, etc. which was used as the backend.
4.	Adobe Photoshop	A de facto standard software in field of raster graphics editing, which was used in designing logos, icons, buttons etc.
5.	Sublime Text 3	A simple and feature packed text editor, which was used for typing and editing codes.
6.	Node.js	An open source, server-side JavaScript run-time environment, which was used in building firebase function for sending push notifications.

In addition to these tools, several open source libraries have been used in the source code of ChaTeX. The table that follows enlists the names of those libraries, the authors of the library and their purpose in our application. All libraries mentioned in the table are available in GitHub.

Table 6: Open source libraries used in ChaTeX

SN	Name of library	Author	Purpose
1.	Lapit Android Firebase Chat App	Akshaye JH (Akshayejh)	For basic UI and layout, for MessageAdapter and FirebaseMessagingService classes.
2.	CircleImageView	Henning Dodenhof (hdodenhof)	To create rounded images used as profile picture indicator and user thumbnails.
3.	Picasso	Square, Inc.	To load ImageView elements with the image whose download URL is provided.
4.	Android Image Cropper	Arthur (ArthurHub)	To get an image from local storage of client and then crop the image to required dimensions and quality.
5.	Compressor	Zetra (zetbaitsu)	To compress the image uploaded by user to a minimal size before updating profile picture or generating thumbnails.
6.	Firebase UI	Google Inc.	To easily connect to and use the Firebase API library.
7.	MathView	Brian Lee (kexanie)	To render the text typeset in LaTeX in its mathematical form.
8.	OkHttp	Square Inc.	Used to exchange media between front-end app and back-end server via HTTP.

9.	SimpleEncryptionLib	Sattar Hummatli (hummatli)	Used to encrypt messages before storing them, and then decrypt them before showing.

3.3. PERFOMANCE ANALYSIS

At the completion of project, the evaluation was done based on performance and feasibility of the deliverable obtained. As a sample test, two users were logged in to the application, and some sample text written in LaTeX syntax is interchanged. The time delay between message synchronization and the rendering efficiency of equations were also considered, and a satisfactory result was obtained. The two users tried to change their respective profile pictures and change the status. To test the push notification feature while receiving request, two different client phones were used.

The time taken for a message to reach from sender to the receiver were taken into account, especially while sending image messages. After the analysis of performance, the application was found to fulfill all the proposed requirements and features.

3.4. TIME SCHEDULE

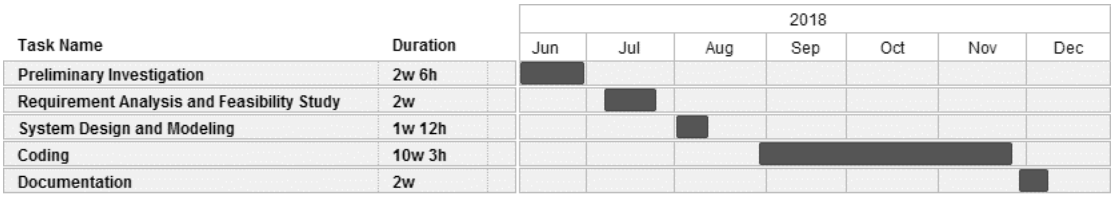


Figure 4: Time Schedule for project

The project was divided into five phases which were carried out as shown in the Gannt above. The major phases were preliminary investigation, requirement analysis and feasibility study, system design and modeling, coding and documentation. Several iterations were carried out as per the iterative software development life cycle.

4. REQUIREMENT ANALYSIS

The core requirements that are inevitable for the achievement of the objectives were identified so that valuable time will not be spent on unnecessary requirement specification. Requirement engineering specifies the softwire's operational characteristics and establishes constraints that the software must meet.

At the initial phase of the requirement analysis, requirements are first listed and then modelled by using various UML modelling diagrams.

The following were extracted as the requirements of the application to be built.

1. The user can download the application and either login to pre-existing account or create a new user account.
2. The user can prompt to sign in via Google, Facebook, Twitter etc. and also via email registration and verification.
3. The user will see his/her recent chats, the list of his/her friends and pending friend requests.
4. The user should be able to search through a list of registered users and then be able to send friend request to others. Similarly, other users should be able to receive or decline it.
5. The user should be able to send text as well as mathematical equations to the user that is friend to him/her.
6. The user should be able to message comfortably even if he/she is not an expert in LaTeX syntax.

4.1. USE CASES

The main purpose of the use cases is to demonstrate the different ways that a user might interact with the system. The few diagrams that follow illustrate different ways of interaction of user and system in the application.

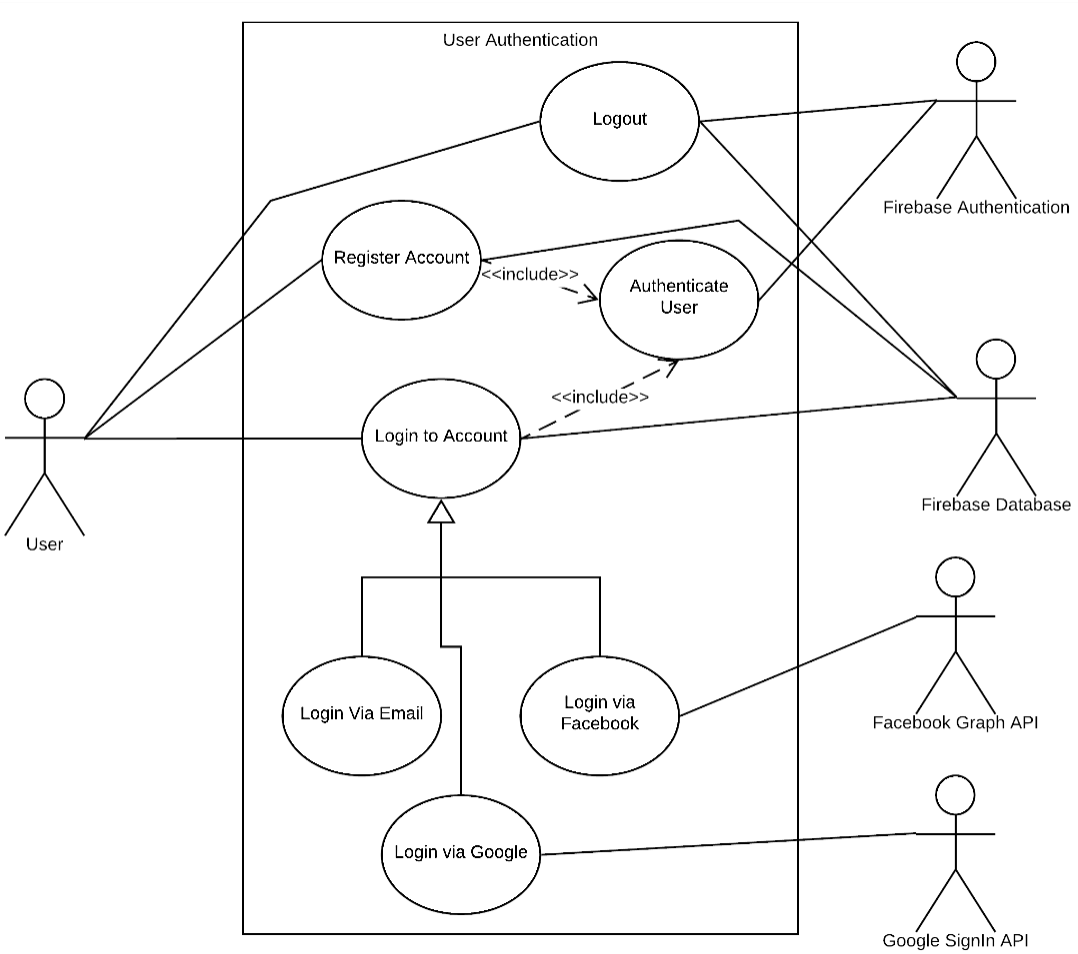


Figure 5: Use Case Diagram for User Authentication

Use Case: Authenticate User	
Actors	User, Firebase Authentication, Facebook Graph API, Google Sign-In Client, Firebase Realtime Database
Description	When the user first launches the application, Firebase Authentication checks whether the user is logged in or not. If the user is not logged in, he/she is taken to the Login / Register form. The user can choose to login/register via email, Facebook or Google. In case the user selects to authenticate via email, he/she provides email and password and then tap the login button. The Firebase Authentication then authenticates the credentials by checking whether they match the existing credentials for the current user and then update the UI correspondingly. In case the user chooses to sign in via Google, a separate intent is started and

	the Google Sign-In API authenticates the Google account of the user. In case the Facebook is chosen as the method of authentication, the Facebook Graph API authenticates the user and then provide Firebase Authentication with the valid credentials. If the user has authenticated to the app for the first time, his/her details are saved in the Firebase Realtime Database. Once the user is logged in, he/she can log out any time he/she likes.
Pre-Conditions	The user should have either a valid email account, a Google account or a Facebook account
Post Condition	The user should be navigated to the main activity if the login is successful.

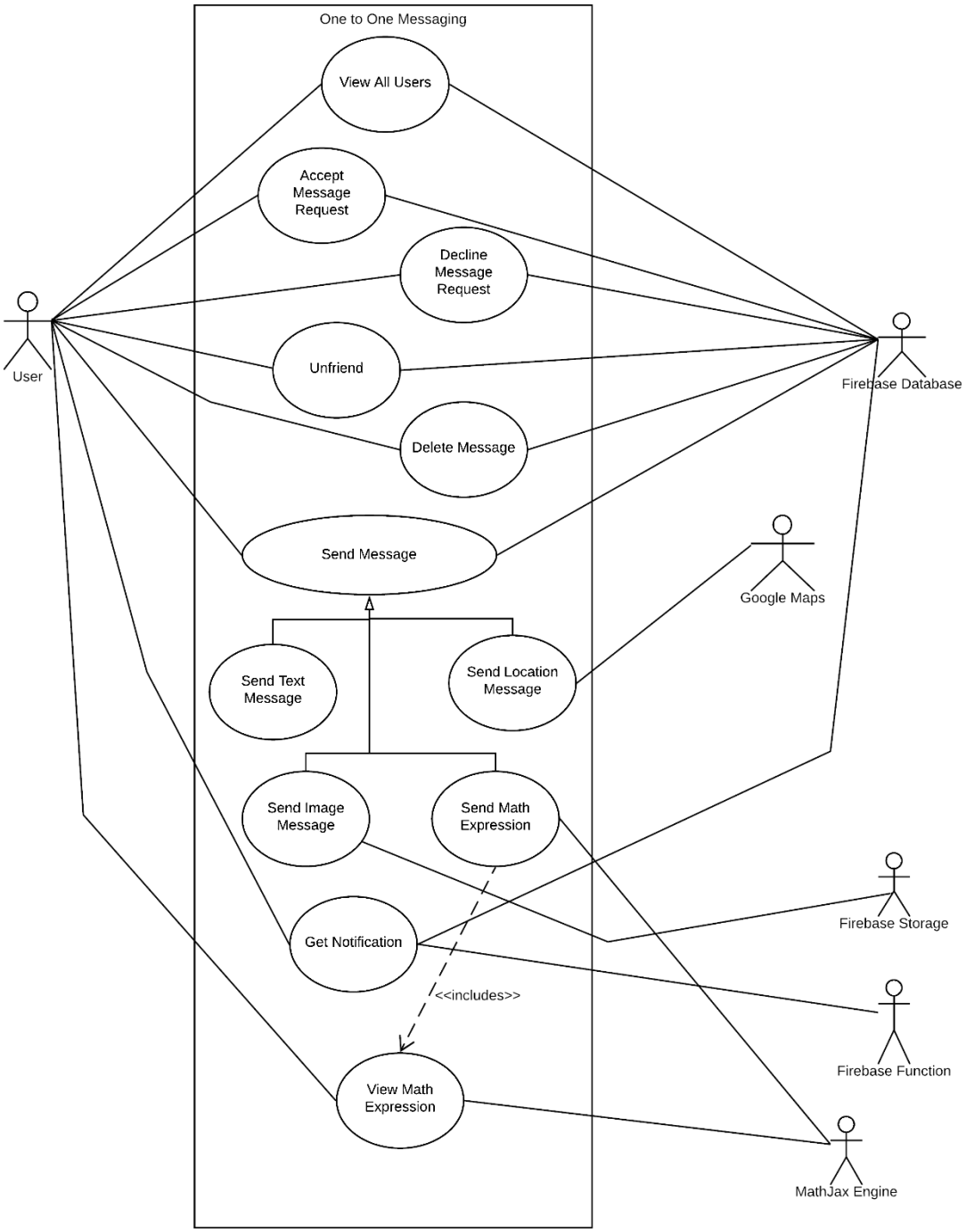


Figure 6: Use Case Diagram for One to One messaging

Use Case: One to One Messaging	
Actors	User, Firebase Realtime Database, Google Maps, MathJax Engine
Description	Once the user is logged in, she should be able to see the list of all the users she has connected to. The user can also view all other users of the application and search the user they are looking for. Since the user can send message request to anyone who is registered to the application, the receiver has to accept the message request for the two of them to have conversation. The receiver may also decline the message request. Once the receiver has accepted the request and the user opens the chat UI, she views the conversation that she has with that specific user. A user can unfriend her existing friends too. Inside the chat UI, the user can send either normal text messages, image, location message or a mathematical expression. In case of image messaging, the user can either select an image from the local storage or take a new picture from the camera. In case of location messaging, the user views a map where she can select a location to send to the other user. The current location of the user can also be sent if the user client has GPS enabled. In case the user selects to send a mathematical expression, the user is presented with a UI where she can type math expressions by help of guided buttons and also view suggestions. (See Use Case: Writing Math Expressions). Once a message is sent, a push notification is sent to the receiver.
Pre-Conditions	The user should be logged in.
Post Condition	A message is sent from the sender account to receiver account, and the receiver account receives a notification.

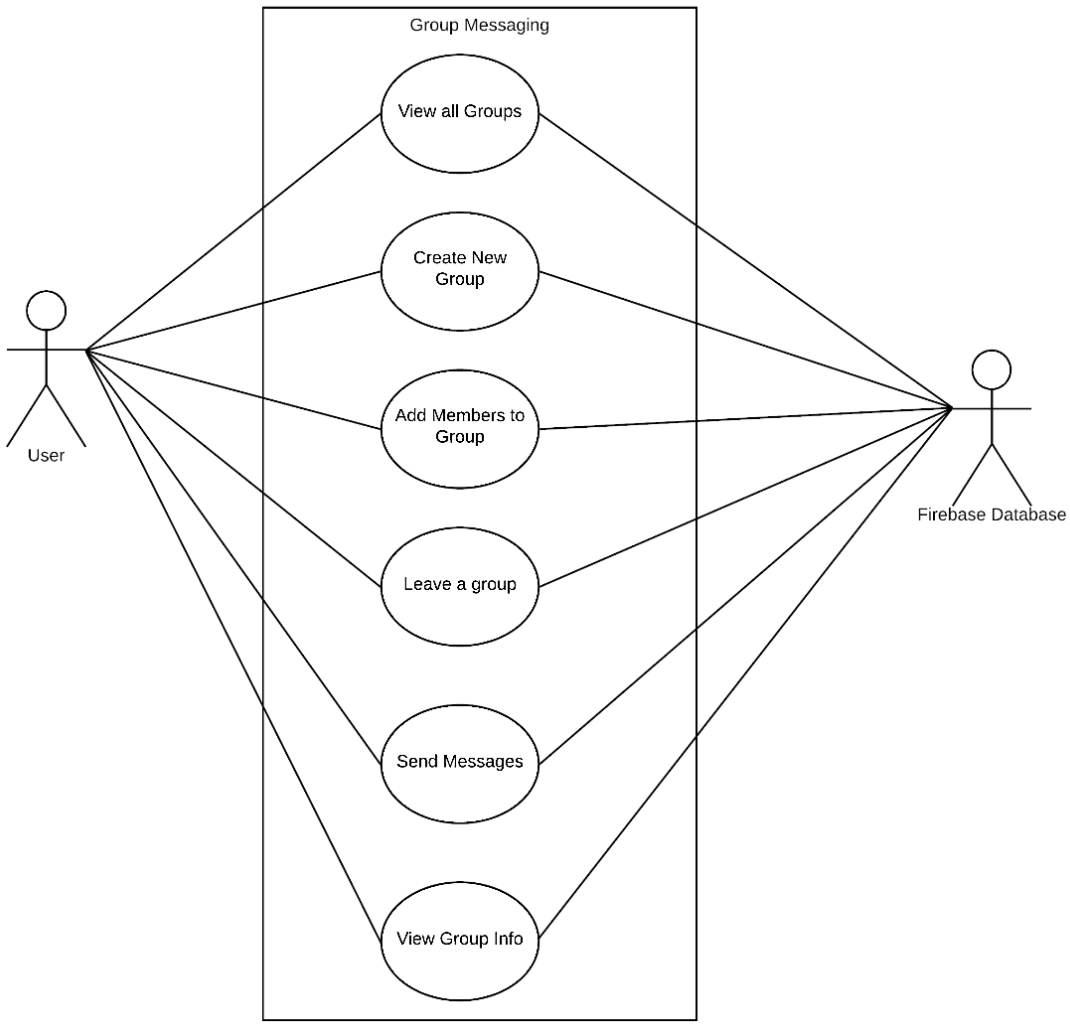


Figure 7: Use Case Diagram for Group Messaging

Use Case: Group Messaging	
Actors	User, Firebase Realtime Database
Description	In the application's main UI, the user can view all the groups that she is member of. The user can create a new group and add any number of users that she is friends with. Any user can leave the group at any time. The member information of the group is written simultaneously to the Firebase Database at the time of creation of group and also while adding new members. The members of the group can view the information about the group. Once a user is a member of a group, she can send message to the group as a whole so that every member of the group get the message.
Pre-Conditions	The user should be logged in.

Post Condition	A message is sent from the sender account to a group, and every member of group receive the message.

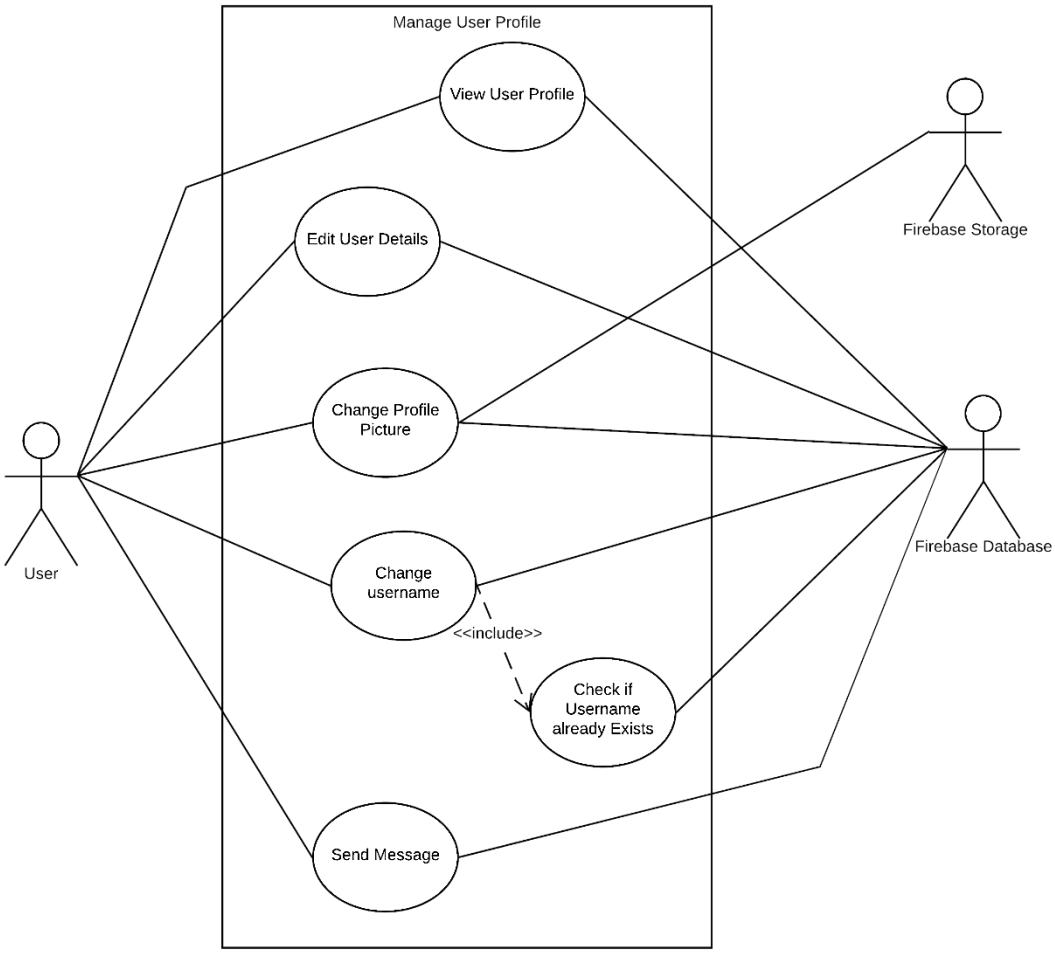


Figure 8: Use Case Diagram for managing user profile

Use Case: Manage User Profile	
Actors	User, Firebase Realtime Database, Firebase Storage
Description	Whenever the user creates an account in the application for the first time, she creates a profile which contains the profile picture, email, username and personal details like name, address, etc. A user can view any other user's profile in the application. She may also choose to send a message to that user, but the two of them cannot have two-way conversation until the receiver accepts the message request. A user can also edit her profile anytime in the

	future, but the username should be unique and not taken by any other users in the application. If the user changes her profile picture, the selected picture is uploaded to Firebase Storage.
Pre-Conditions	The user should be logged in. The username should be unique.
Post Condition	The profile of the user with unique username is updated.

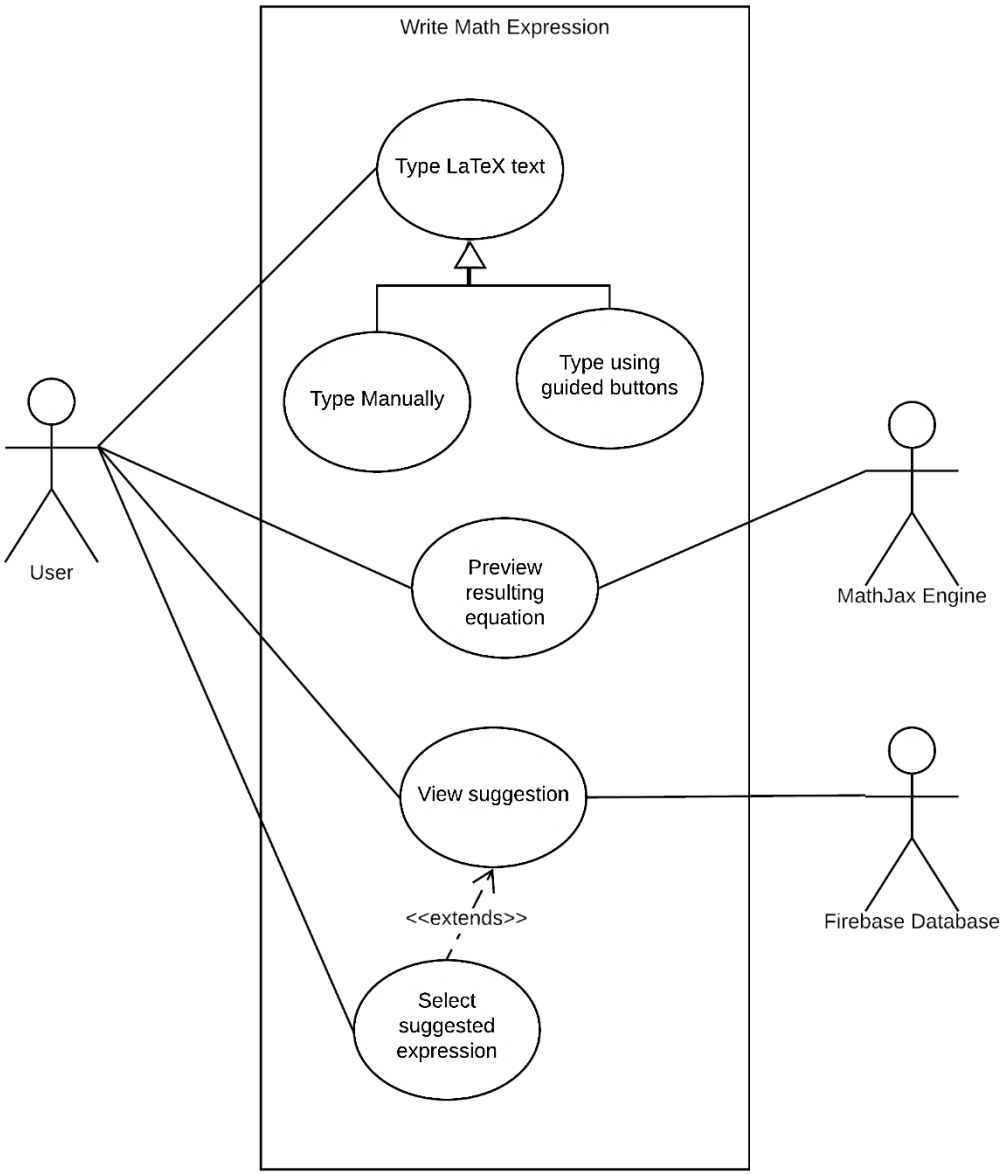


Figure 9: Use Case Diagram for writing math equation

Use Case: Write Math Equation	
Actors	User, Firebase Realtime Database, MathJax Engine
Description	When the user wants to write math equations, she is navigated to a separate activity where there are guided buttons with mathematical symbols labelled on them. If the user presses in any one of the buttons, the LaTeX code that corresponds to the symbol labelled on the button is typed automatically in the message box. The user can also view the preview of the mathematical equation she is typing. Moreover, the user is provided with the suggestions regarding what equation she wishes to type, based on the list of frequently typed equations and the past equation message history.
Pre-Conditions	The user must be logged in.
Post Condition	The user writes the desired mathematical expression.

5. DESIGN

This section documents the various steps taken during the design phase of ChaTeX application.

5.1. E-R DIAGRAM

The main purpose of E-R diagram is to illustrate an information system's entities and the relationships between those entities graphically. The following ER diagram shows the various entities and the relationships among them in the application.

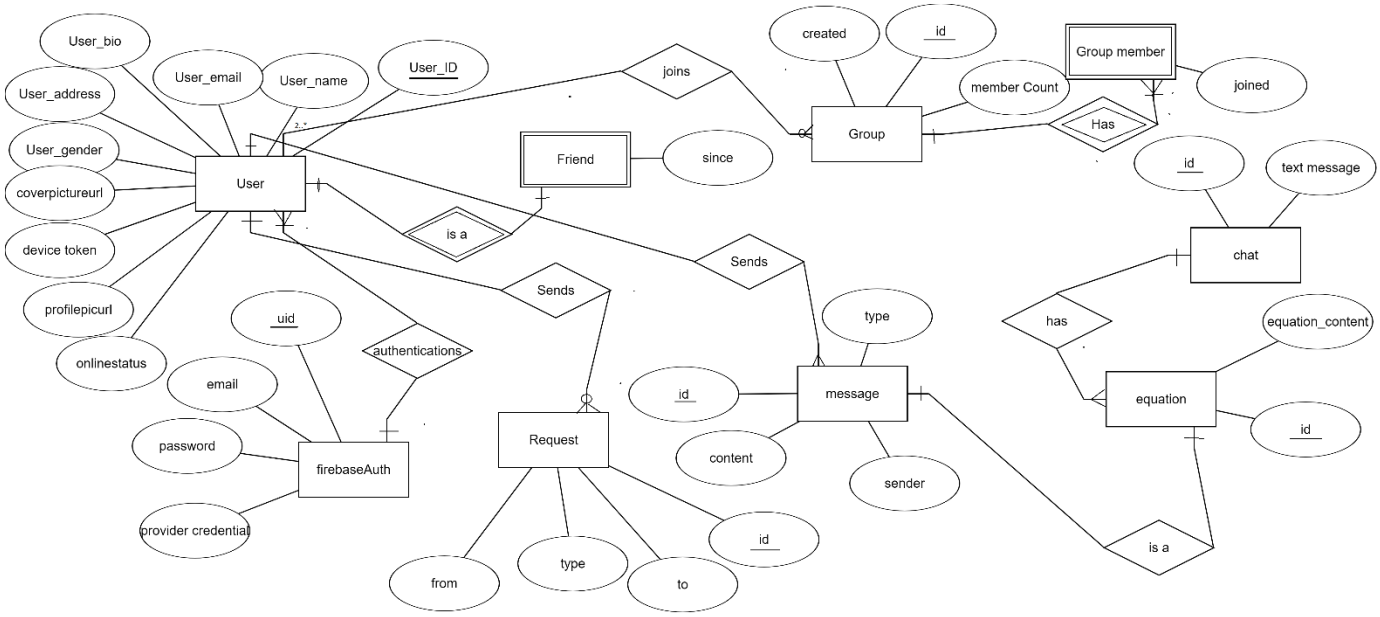


Figure 10: ER diagram of ChaTeX

5.2. CLASS DIAGRAM

The main purpose of class diagram is to describe the structure of a system by showing the system's classes, their attributes, operations and the relationships among objects. The following class diagram shows the overall schema of the objects used in the application.

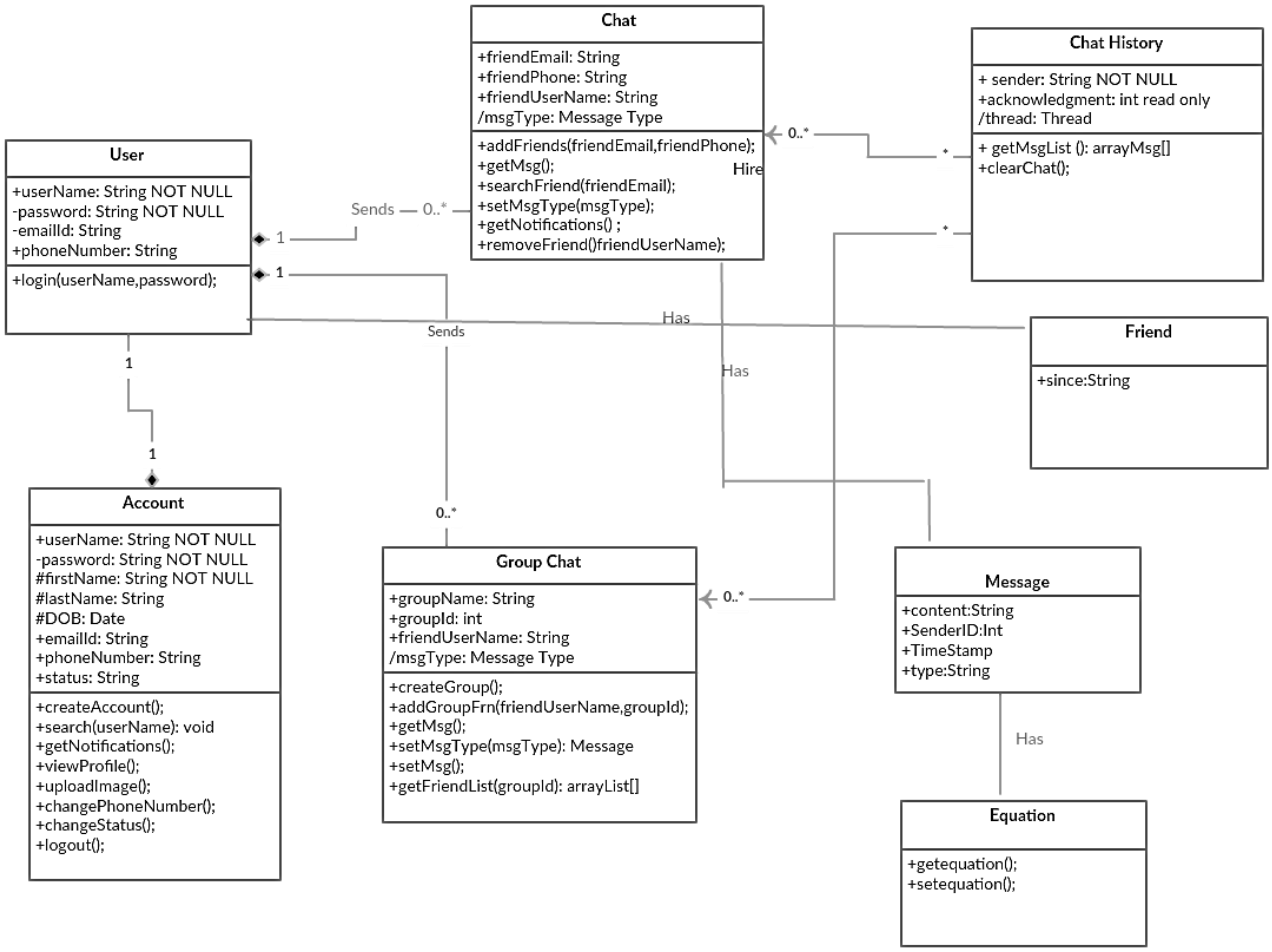


Figure 11: Class Diagram for ChaTeX

5.3. DATABASE SCHEMA

Google Firebase uses a NOSQL database which is quite different from the traditional relational model of database. All data in Firebase Realtime Database are stored as JSON objects. For example, a message node may be of the structure:

message:{

id : sNXdttesXzdtujgei,

content : “Hello, There!”,

sender : Taeisute78etSggDT,

type : “text”,

timestamp : 1544271426

}

The database model used by Firebase is quite like hierarchical database model, where data are added to a JSON tree. Hence, the overall schema of the database is very different from the relational schema which is best described using a tree. Appendix II shows the NoSQL Firebase Realtime Database schema used in the application.

5.4. SEQUENCE DIAGRAMS

A sequence diagram shows the dynamic view of the interaction between the various objects during runtime. The following sequence diagrams show how the client and the different objects of the application interact with each other in different use case scenarios.

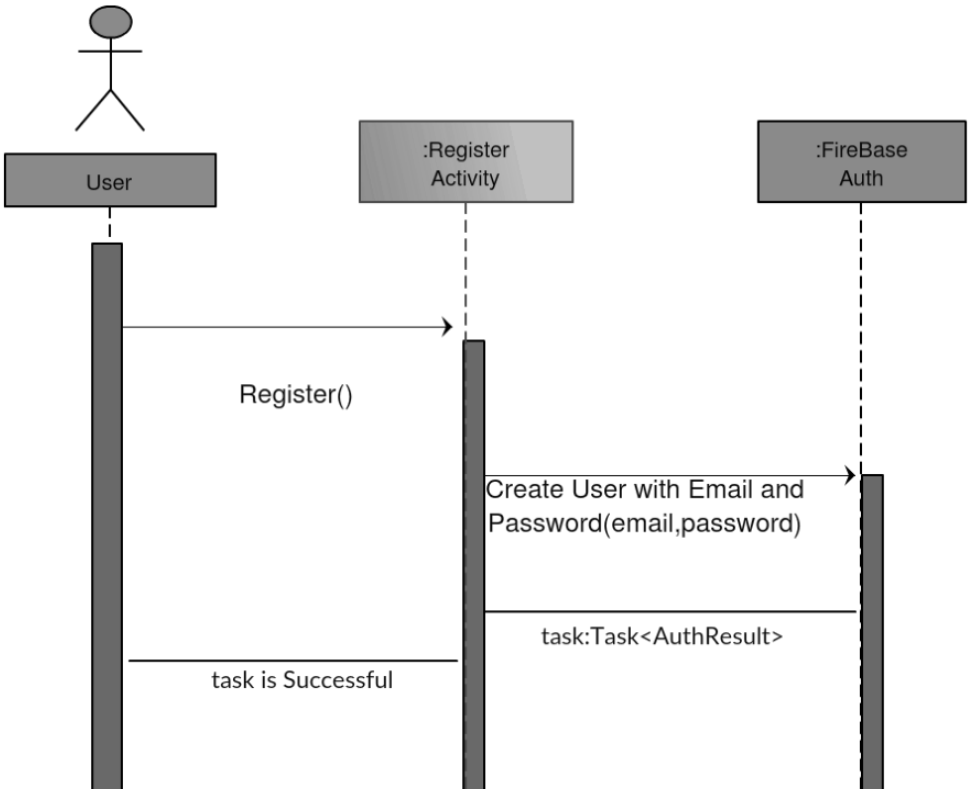


Figure 12: Sequence diagram for registering via email

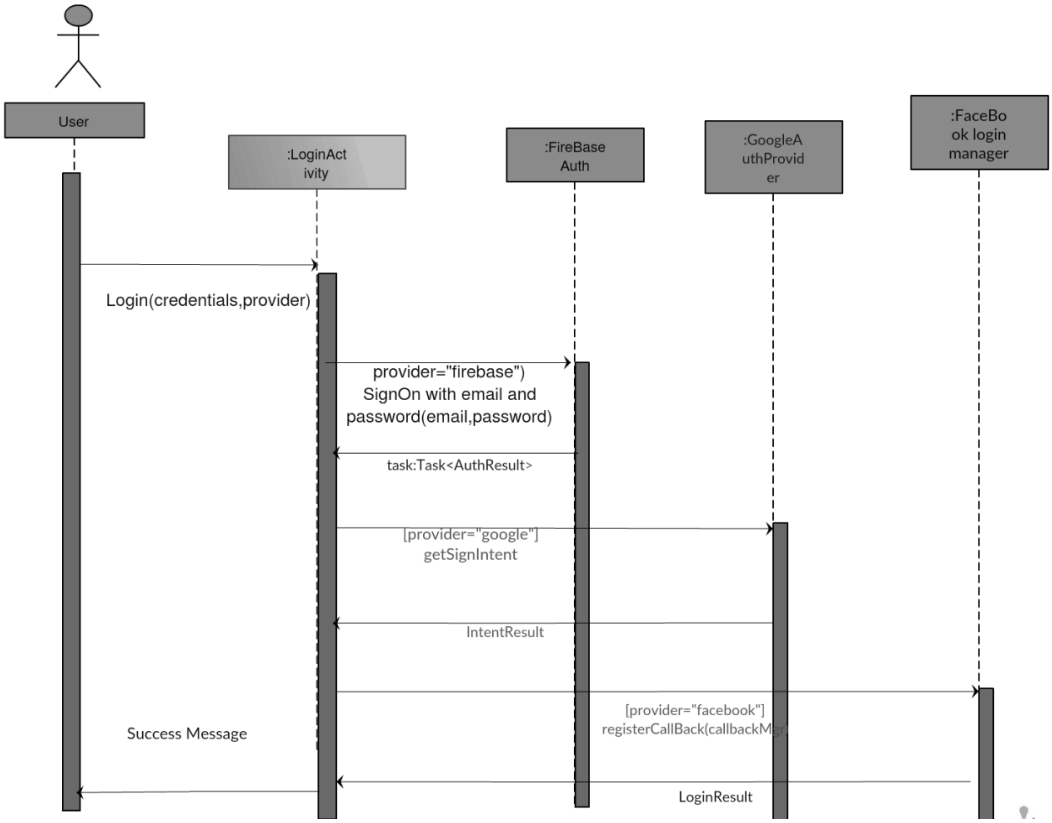


Figure 13: Sequence diagram for login

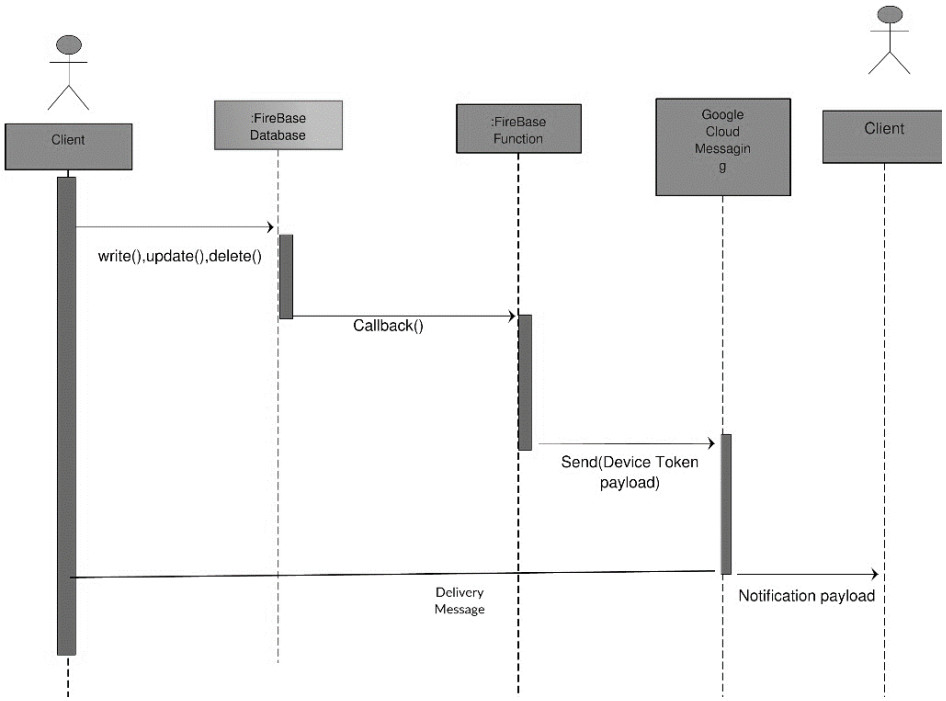


Figure 14: Sequence diagram for push notification

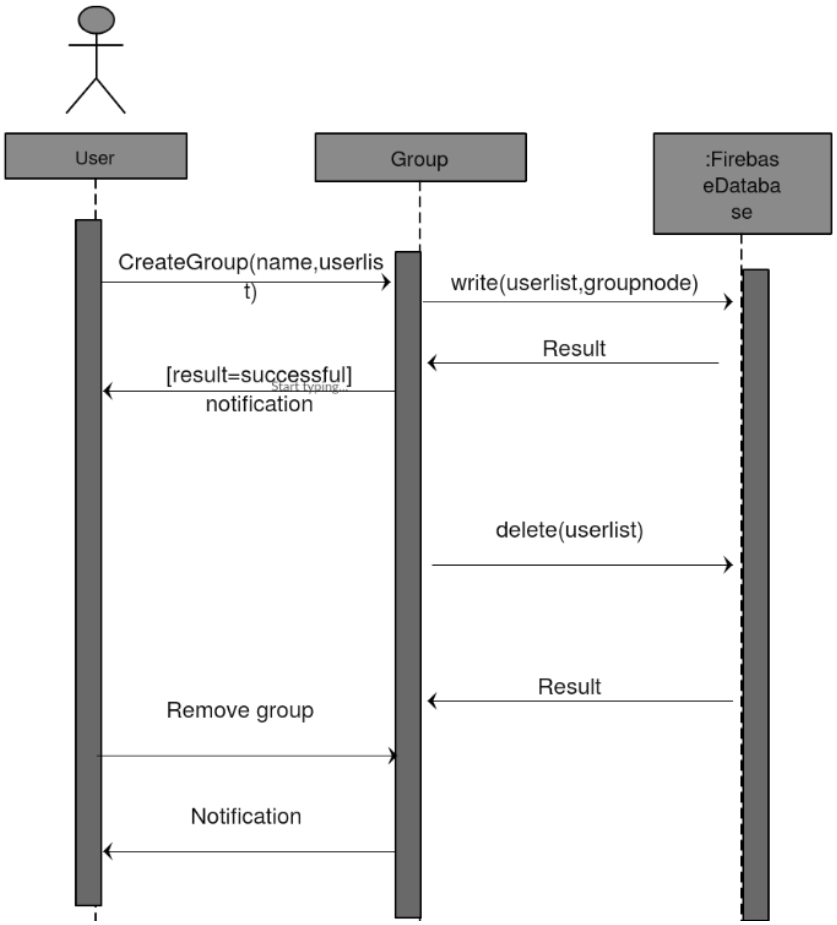


Figure 15: Sequence diagram for group messaging

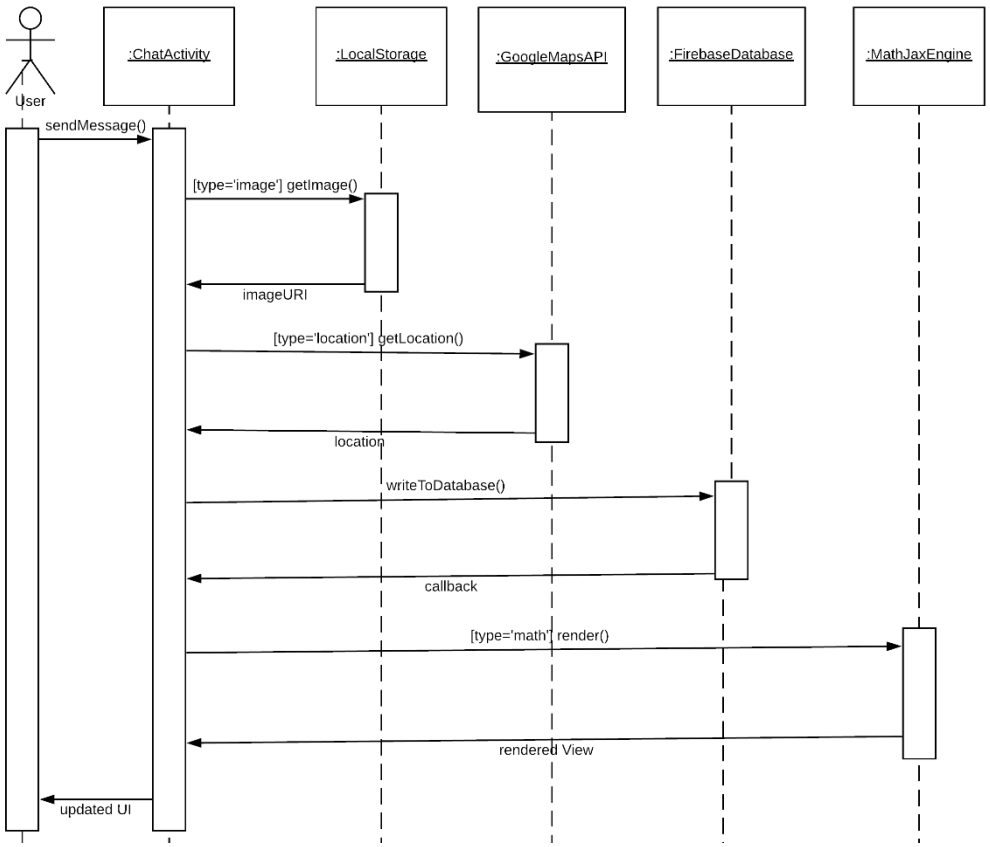


Figure 16: Sequence diagram for one to one messaging

6. CODING AND IMPLEMENTATION

After figuring out the logical solution of the enlisted requirements, the app was set up and coded. The following sections describe the various steps that were required during the implementation phase of ChaTeX.

6.1. PROJECT SETUP

First and foremost, a new project was set up in the Android Studio IDE with following specifications:

Package name: com.thecoffeecoders.chatex

Android Studio Version: 3.2.1

JRE version: 1.8.0_152

Minimum API Level: 21 (Android 5.0 Lollipop)

6.2. FIREBASE INTEGRATION

The Android Studio has a built-in tool named Firebase Assistant in Tools menu. The Firebase assistant eases the integration of Firebase to an android project drastically, as it sets up dependencies and all configurations needed in server side automatically. In addition, we have used another library named FirebaseUI, which makes it easy for us to connect and work with Firebase API. The following dependencies related to Firebase were added to the gradle file of the application.

Table 7: Firebase dependencies added in ChaTeX

Dependency	**Version**
Firebase Auth	16.0.3
Firebase Database	16.0.3
Firebase Storage	16.0.3

Firebase UI Database	4.2.1
Firebase Messaging	16.0.3

6.3. UI DESIGN

This phase involves the design of various layouts to be displayed in the front-end when user browses to several activities and fragments. These layouts are designed in Android Studio IDE and are written in XML.

The various layouts created during the coding phases and their purposes are enlisted in the table that follows. The screenshots of the these layouts are provided in Appendix-I.

Table 8: Layouts used in the application

SN	Layout Name	Purpose
1.	activity_chat.xml	Contains a RecyclerView that displays all text messages sent and received by a user.
2.	activity_developers.xml	Display information and images of the developers of the app.
3.	activity_login.xml	The login page where user can enter email and password and tap on Login button.
4.	activity_main.xml	The main activity layout that opens by default whenever the application is launched. This activity contains chat messages, friends list and request list.
5.	activity_register.xml	The register page where user can enter name, email and password and tap on register button.
6.	activity_write_equation.xml	User can write mathematical equations.

7.	activity_send_location.xml	This page is where user can send locations.
8.	activity_forgot_password.xml	The password is recovered in case you forget.
9.	activity_edit_profile.xml	Users can edit their info in this page.
10.	activity_add_group.xml	Users can create group adding their friends.
11.	app_bar_layout.xml	A app bar that appears at the top of every page which may contain a title and a back button.
12.	chat_custom_bar.xml	A custom app bar which appears only on Chat Activity page.
13.	cardview_group_row.xml	A single layout for a Group object displayed in the GroupsFragment that contains the name and image of the group.
14.	fragment_chats.xml	A fragment that lists all the conversation made by the current user with the last message sent to or received from that user.
15.	fragment_friends.xml	A fragment that lists all the users who are currently friends with this user.
16.	fragment_requests.xml	A fragment that lists all the users who have either received request from current user or sent request to the current user.
17.	fragment_find_friends.xml	A fragment that has searching feature.
19.	fragment_groups.xml	A fragment that lists all the groups to which the user is a member.
20.	message_single_layout.xml	The layout of a single message sent by either sender or the receiver, which contains the

		message, sender/receiver name, avatar, sent images, etc.
21.	nav_header_main.xml	The header layout for the app's navigation drawer.
22.	row_single_friend.xml	A single layout that displays a friend's name, username and the profile picture.
23.	users_single_layout.xml	The layout of a single user entry in fragement_chats, fragment_friends or fragment_requests.
24.	row_single_equation	A single layout that displays a mathematical equation.

6.4. REGISTRATION AND AUTHENTICATION

After the dependencies for Firebase Authentication are correctly set up, the registration and login feature were implemented. The Firebase API provides classes like FirebaseUser and FirebaseAuth to work on with authentication and registration feature. The listener class FirebaseUser.AuthStateListener listens for the change in authentication state of the user, and then fire event if any change occurs.

A new user with an email and password can easily be created by invoking a call to method createUserWithEmailAndPassword(String email, String password),which takes the email and password values as parameters. (See Appendix III)

The method signInWithEmailAndPassword(String email, String password) was invoked to authenticate the users with their email and password. The parameters email and password are provided by fetching text input by the user in the EditText view in the LoginActivity.

6.5. USER PROFILE AND AVATAR

The users of ChaTeX can maintain their own avatar and status. The display name of the user is same as the name entered by him/her during registration, which can be changed later. The users can navigate to change their profile from MainActivity by clicking on profile picture image on the navigation drawer.

The change of the user profile is done by navigating to another activity named EditProfileActivity, which allows user to type in his new details and then update it. The open source library Android Image Cropper is used to let users select image from the local storage and crop it to a square shape. The code snippet that illustrates how a local file can be fetched from local storage and cropped is documented on Appendix III.

After the image is cropped to required size, the image is then compressed to reduce the original file size. We have used open source library named 'Compressor' to reduce the size of the image cropped by the user. The image was then uploaded to Firebase Cloud Storage, and then their download URLs were stored in the respective fields of the 'Users' object in the real-time database. Whenever the image was needed to be loaded to a ImageView, the URLs of the images were obtained from the database and the image represented by that URL was downloaded dynamically with Glide open source library. In addition to these details, the device token id of a user is also stored in the database. This token is very helpful in sending notification to a user device.

6.6. SENDING MESSAGE REQUESTS

When a user sends a message to another user that is not already friend to the user, that message is sent as a message request to the other user. For this, the message is written to the 'requests' node in the database rather than to the 'chat' node. When the other user sees the message, she is given two options: either to accept the message request (which will automatically add that user in her friend list) or decline the message request. Once the user accepts the message request, the two users can have two way conversation.

6.7. SENDING TEXT MESSAGES

The information about the text messages sent by the users are stored in an object named 'Chat' and the actual messages are stored in object named 'messages' in the firebase

real-time database. The 'Chat' object stores information about the time of message departure and the seen status of the message. A green dot can be seen by the side of the user's name if the user is currently online. The changes in these database objects are automatically detected by the event listeners and the chat UI is updated accordingly.

The 'messages' database object stores sent and received messages of both sender and receiver. The key value pairs of information stored inside the child entry of 'messages' object are:

Table 9: The message information entry stored in the Firebase Database

Key	**Value**
from	UserID of the user who sent this message
message	The encrypted contents of the message
seen	True/False according to whether the receiver has seen the message or not.
time	Server timestamp when this message was sent.
type	'text'/'image' according to the type of the message sent.

6.8. RENDERING LATEX TEXT

Each message that contains LaTeX text in the conversation are displayed in a custom view called MathView, which is provided by the open source library called 'MathView'. The text message is checked at first using Java Regular Expressions to find out whether it contains text typeset in LaTeX or not. If yes, the text is displayed in MathView and the TextView is hidden; if no, the TextView becomes visible and MathView becomes invisible.

MathView is very similar to TextView in its structure and methods but the difference is that it renders the LaTeX typeset text passed into it. Whenever a message is sent or received, he setText(String text) method of MathView is used to pass a LaTeX typeset text typed by the user into the view object, and the text is automatically rendered. Two different engines, namely MathJax and Katex can be used to render the equations, among which MathJax engine is used in this project.

MathView supports both single line equations and inline equations. Single line equations should be typed inside $$ $$ characters as like in $$x^2$$, and the inline equations should be typed inside \(.. \) characters as like in \(x^2\). The user can click on the math toggle button once to automatically insert \(\) into the message, or long click in it to insert $$ $$. The inline equations appear in the same line as the other text characters of the message, whereas the single line equation is displayed in a new line separated from the other text characters of the image.

6.9. SENDING IMAGE MESSAGES

Each single message layout has a by-default hidden ImageView, which is only made visible when the user selects to send image message by clicking on plus icon in ChatActivity page. The user can select an image from the local storage to send, after which the selected image is uploaded to the Firebase Cloud Storage and the download URL of the image message is sent to the receiver. The receiver then updates the hidden ImageView in the ChatActivity page with the image located at the URL received as a message and makes the ImageView visible.

6.10. SENDING LOCATION MESSAGES

The user can send their current as well as any other location to the other user from the integrated Google Maps. For this, the user will be navigated to a separate activity called SendLocationActivity which will take the location coordinate of the place in map where the user taps and then sends that location to the other user. The other user will be able to view the location in app as well as in Google Maps app if it is installed in the user's device.

7. TESTING AND REVIEW

This section describes the process involved in various tests conducted in the application. The specification of the devices in which the app was tested on is listed on the table that follows.

Table 10: Devices used for testing the application

SN	Device Name	Specifications
1.	OnePlus 2	Manufacturer: OnePlus ROM: Pixel Experience | Beta Android Version: 9.0 (Pie) API Level: 28
2.	Android Studio Emulator	Android Version: 8.1 (Oreo) API Level: 27

7.1 HARDWARE AND SOFTWARE SPECIFICATIONS

The following are the different specifications both in hardware and software levels that a target device must fulfill to run ChaTeX successfully.

Minimum Space Required: 50MB or higher

Operating System: Android

Target API level: 28

Minimum API Level: 21

App Permissions: Read and Write External Storage, Access Internet, Camera, Location

Google Play Services must be installed.

7.2 TEST CASES

The following test cases were used while unit testing the application.

Table 11: Test cases used for unit testing.

Test Case #	**Category**	**Test Scenario**	**Expected Result**	**Actual Result**
1.	General UI	Launch application from the app drawer	App launch successful	App launch successful
2.		Uninstall, reinstall and launch the application.	Open Login Form	Login form opens
3.		Login to the account for the first time.	UI shows empty view in main UI	UI shows empty view in main UI
4.		Navigate to 'Find Friends' menu	UI shows the registered users of the application	UI shows the registered users of the application
5.		Navigate to 'Find Friends' menu and tap on a random user.	UI shows the user's profile	UI shows the user's profile
6.		When viewing user's profile, tap on message button	Shows Chat UI with the corresponding user	Shows Chat UI with the corresponding user
7.	Login / Register	Enter unregistered email and tap login button	Show 'wrong credentials' dialog	Shows 'Wrong credentials' dialog
8.		Enter registered email but incorrect password	Show incorrect credentials dialog	Shows 'the credentials you

				provided do not match'
9.		Uninstall Facebook application, launch the application and tap Facebook icon to log in via Facebook.	Facebook Login page opens in a WebView	Facebook Login page opens in a WebView
10.		Remove Google account from the phone, launch the application and tap the Google icon to log in via Google	Phone asks to add a new Google account	Phone asks to add a new Google account to continue
11.		Register account by an email that is already registered.	Show 'Email already registered dialog'	Shows 'Email already registered dialog'
12.		Register account by providing correct email and password credentials	Account is successfully created and user is taken to EditProfileActivity	Account is successfully created, and user is taken to EditProfileActivity
13.	Edit Profile	Tap 'Update profile' button leaving the fields blank.	Prompt that the fields cannot be empty	Prompts that the fields cannot be empty
14.		Type preexisting username and tap 'update Profile' button	Prompt that the username is already taken	Prompts that the username is already taken
15.	Chat UI	Type nothing and hit send button	Nothing happens	Nothing happens
16.		Type a text message and then hit send button	The text message is sent to the other user	The text message is sent to the other user

17.		Tap '+' icon, select 'Image' option, select an image from gallery	The selected image is sent to the other user	The selected image is sent to the other user
18.		Tap '+' icon, select 'Math' option,	A new UI shows up with buttons labelled with mathematical symbols.	A new UI shows up with buttons labelled with mathematical symbols.
19.		Tap '+' icon, select 'Location' option, select an image from gallery	UI shows scrollable google map	UI shows scrollable google map
20.		Tap GPS icon in the google maps UI	The pointer changes its position to the current location	The pointer changes its position to the current location
21.	Typing Math Expression	Tap on a few guided buttons	The UI shows the preview of the typed expression.	The UI shows the preview of the typed expression.
22.		Alter the syntax of LaTeX in the EditText	The preview no longer shows the expected mathematical expression	The preview no longer shows the expected mathematical expression
23.		Type an equation and tap send button	The typed expression is sent to the other user	The typed expression is sent to the other user
24.	Group Chat	Create a new group without providing the group name	Shows prompt 'name cannot be empty'	Shows prompt 'name cannot be empty'
25.		Create a group without adding any one of the friends	Shows 'Add at least one another person' dialog	Shows 'Add at least one another person' dialog

26.		Create a group with all of the friends selected to be added	A new group is created with all friends as its member.	A new group is created with all friends as its member.

8. CONCLUSION

The major problem that motivated for the development of ChaTeX was the difficulty that exist today while sending mathematical equations and expressions in chat among the students and scholars. As ChaTeX can successfully send and receive messages like any other chat applications and in addition render the text typeset in LaTeX, the authors do attempt to give a conclusion that ChaTeX is a great success achieved in solving the problems stated earlier. While it can be generally agreed that the application is simple and primitive, nonetheless it meets all the basic criterions that were proposed in the project proposal and moves one great step further in the instant messaging industry. In addition to the LaTeX rendering, the feature of sending image messages was also added later, which certainly proves the capability of ChaTeX. In the present context where there is no such application available in Android platform, ChaTeX is a historical milestone in a way towards versatility in Instant Messaging.

9. RECOMMENDATIONS

Due to time limits and weightage of the project, the application has some distinct aspects that can be improved further. This section provides some recommendations that can further enhance the usability, capability and versatility of the application.

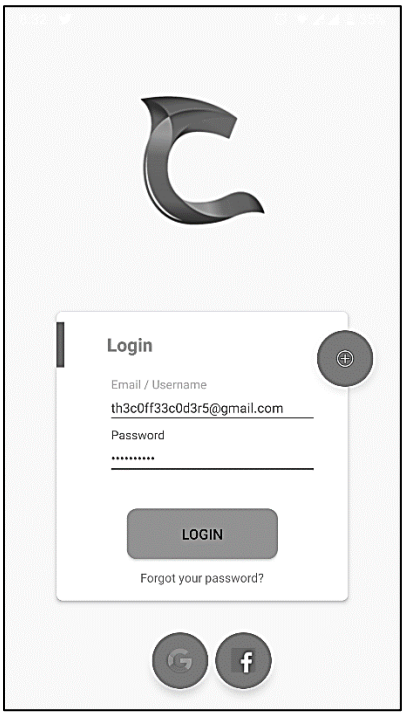
1. Even though the users of ChaTeX can type some simple equations using guided buttons, it is still assumed that the users know the basics of LaTeX and its syntax to type the required equation in the correct format. A naïve user who doesn't have knowledge about LaTeX would find the application less useful because he/she must first learn the syntax of LaTeX. This can be improved using handwriting recognition APIs like MyScript Developer where users can draw the desired form of equation on the screen, which is then recognized by the recognition algorithms and automatically converted to LaTeX text.
2. The database structure used in ChaTeX is a very primitive database, which can certainly be improved by using more normalized forms and reducing the redundancy in the fields.
3. Features like sending videos, sending contact cards, graffiti, handwriting, gifs, emojis, etc. can be added to make the application more useful.
4. The authentication methods can be extended to other alternatives like Phone authentication, Twitter, etc.
5. After the optimizations, the application will be published to the Google Play Store soon.

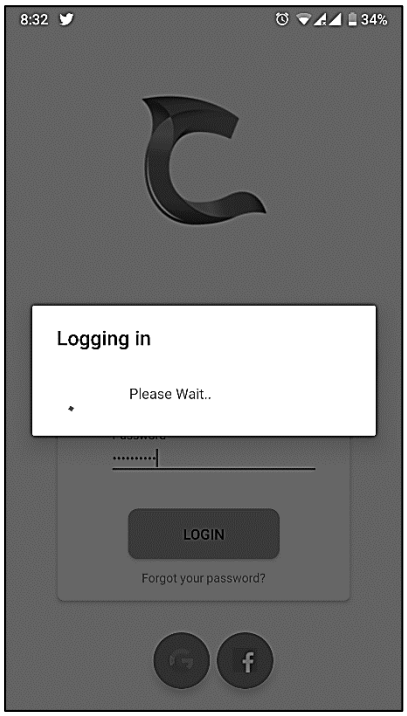
10. REFERENCES

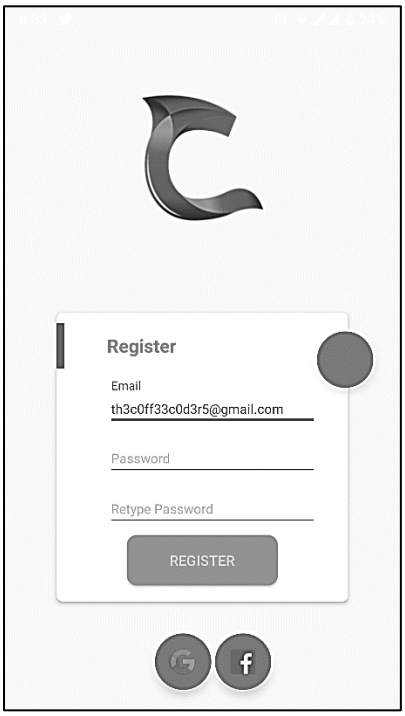
1. "MathIM: Math Typeset Chat" [Online] https://mathim.com/
2. "Instant Messaging: Wikipedia" [Online] https://en.wikipedia.org/wiki/Instant_messaging [Accessed: 03-Sep-2017]
3. "LaTeX: Wikipedia" [Online] https://en.wikipedia.org/wiki/LaTeX [Accessed: 03-Sep-2017]
4. Online LaTeX Equation Editor: Codegogs. [Online] https://www.codecogs.com/latex/eqneditor.php
5. "Firebase: Wikipedia" [Online] https://en.wikipedia.org/wiki/Firebase [Accessed: 03-Sep-2017]
6. Google Firebase Documentation: Google LLC. [Online] https://firebase.google.com/docs/ [Accessed: 01-Nov-2017]
7. "Android (Operating System): Wikipedia" [Online] https://en.wikipedia.org/wiki/Android_(operating_system) [Accessed:10-Jan-2018]
8. "UML Use Case Diagram tutorial": LucidChart [Online] https://www.lucidchart.com/pages/uml/use-case-diagram [Accessed: 10-jan-2018]
9. FirebaseUI-Android, Google LLC. [Online] https://github.com/firebase/FirebaseUI-Android [Accessed: 01-Jan-2018]
10. MathJax [Online] https://www.mathjax.org/
11. "Docs | Node.js" Node.js foundation. [Online] https://nodejs.org/en/docs/ [Acessed: 01-Jan-2018]
12. Lapit Android Firebase Chat App: Akshaye JH [Online] https://github.com/akshayejh/Lapit---Android-Firebase-Chat-App
13. "Firebase Chat App – Android Studio | One to One Chat Application": TVAC Studio, (YouTube) [Online] https://www.youtube.com/playlist?list=PLGCjwl1RrtcQ3o2jmZtwu2wXEA4OIIq53
14. CirceImageView: Henning Dodenhof (hdodenhof) [Online] https://github.com/hdodenhof/CircleImageView
15. Picasso: Square Inc. [Online] https://github.com/hdodenhof/CircleImageView
16. Android Image Cropper, Arthur (ArthurHub) [Online] https://github.com/hdodenhof/CircleImageView
17. Compressor: Zetra (zetbaitsu) [Online] https://github.com/zetbaitsu/Compressor
18. MathView: Brian Lee (kexanie) [Online] https://github.com/kexanie/MathView
19. OkHttp: Square Inc. [Online] http://square.github.io/okhttp/
20. SimpleEncryptionLib: Sattar Hummatli (hummatli) [Online] https://github.com/hummatli/SimpleEncryptionLib

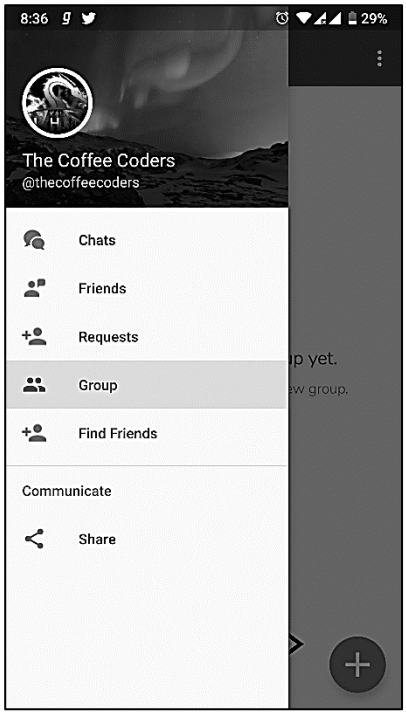
APPENDIX - I

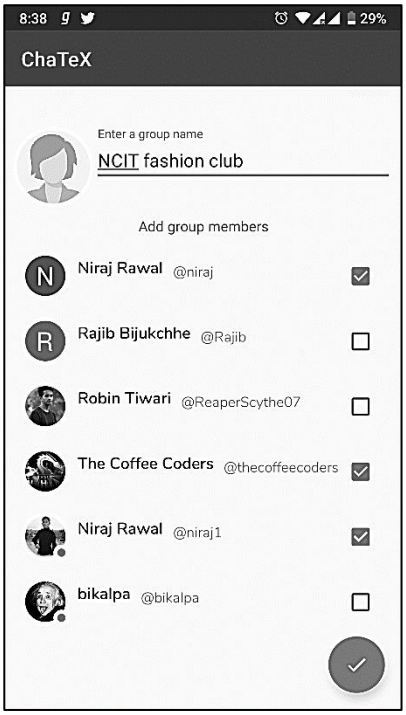
This appendix lists the screenshots of various screens of the application UI. Please note that the UI could be quite different on different user devices because the layouts depend on screen size, pixel quantity, etc. of the target android device.

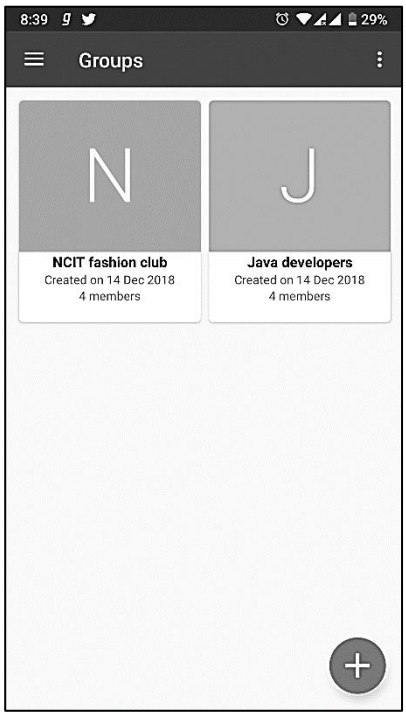


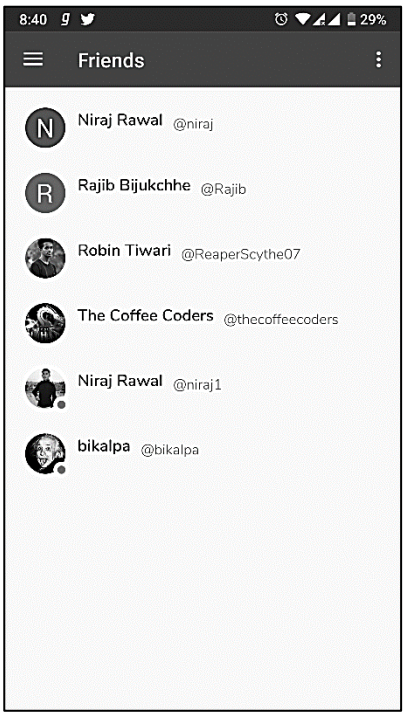




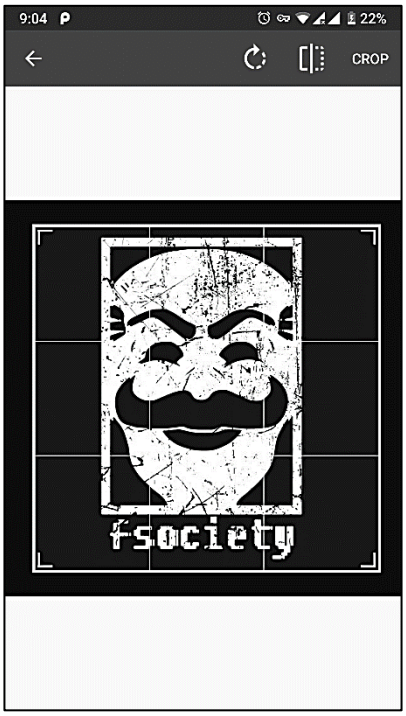

8:36
29%
The Coffee Coders
@thecoffeecoders
Chats
Friends
Requests
Group
Find Friends
Communicate
Share

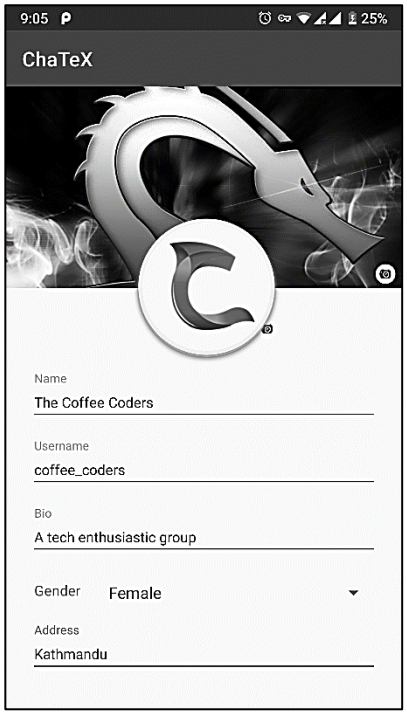

8:38
29%
ChaTeX
Enter a group name
NCIT fashion club
Add group members
Niraj Rawal @niraj
Rajib Bijukchhe @Rajib
Robin Tiwari @ReaperScythe07
The Coffee Coders @thecoffeecoders
Niraj Rawal @niraj1
bikalpa @bikalpa

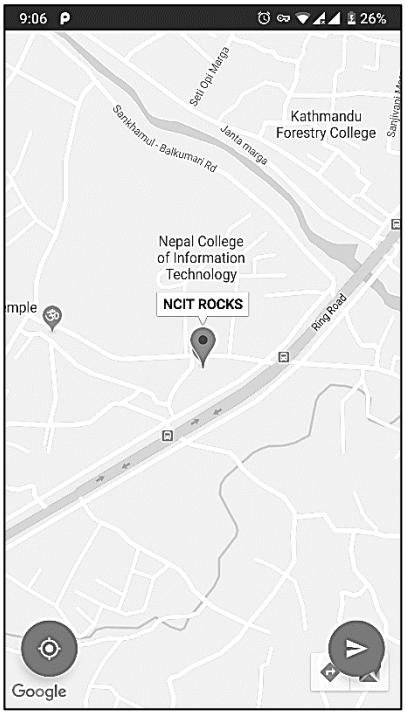

8:39
29%
Groups
N
J
NCIT fashion club
Created on 14 Dec 2018
4 members
Java developers
Created on 14 Dec 2018
4 members

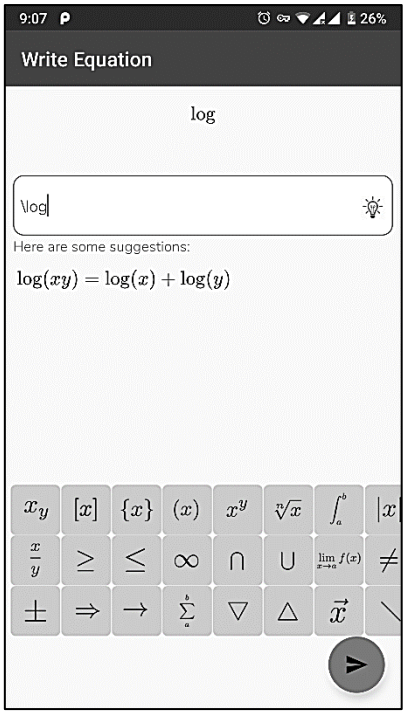

8:40
29%
Friends
Niraj Rawal @niraj
Rajib Bijukchhe @Rajib
Robin Tiwari @ReaperScythe07
The Coffee Coders @thecoffeecoders
Niraj Rawal @niraj1
bikalpa @bikalpa


9:03
21%
CROP
Select source
Camera
Camera
Files
Photos
Gallery
Bitmoji
ES File Explorer
Find command
FX File

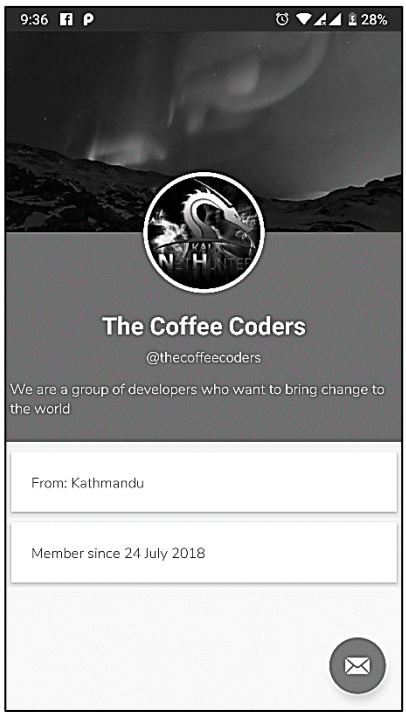

9:04
22%
CROP
fsociety

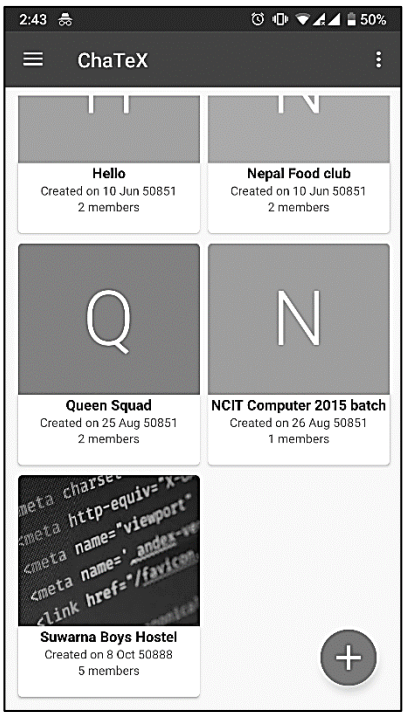

9:05
25%
ChaTeX
Name
The Coffee Coders
Username
coffee_coders
Bio
A tech enthusiastic group
Gender
Female
Address
Kathmandu


9:06
26%
Kathmandu Forestry College
Nepal College of Information Technology
NCIT ROCKS
Ring Road
Google


9:07
26%
Write Equation
log
\log
Here are some suggestions:
$\log(xy) = \log(x) + \log(y)$

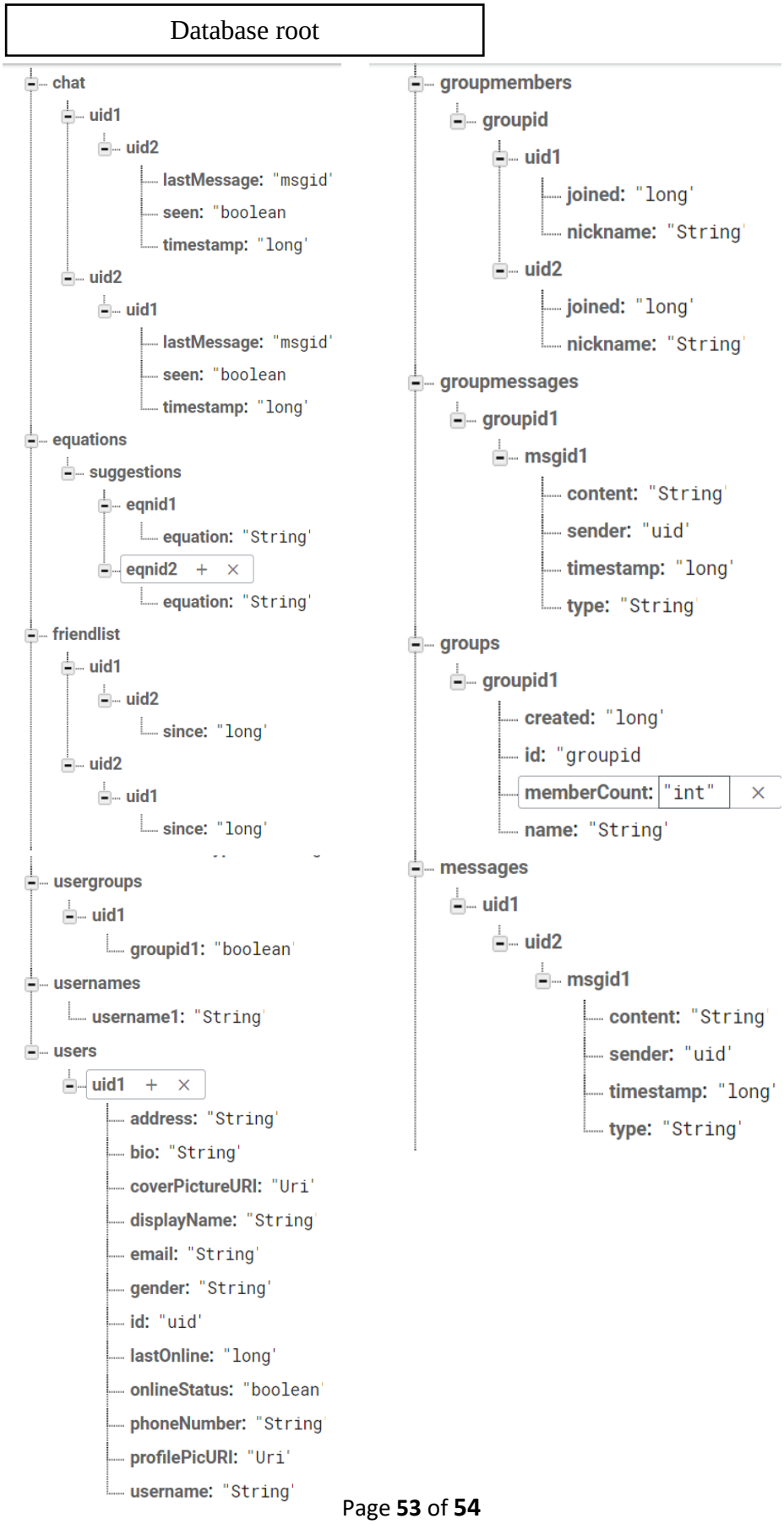

9:29
27%
The Coffee Coders
Active Now
Hi !
Hello, how are you?
09:15 AM
All good.
need help on this
Sure. But where are you?
09:25 AM
Location Message
Type a message..


9:36
28%
The Coffee Coders
@thecoffeecoders
We are a group of developers who want to bring change to the world
From: Kathmandu
Member since 24 July 2018


2:43
50%
ChaTeX
Hello
Created on 10 Jun 50851
2 members
Nepal Food club
Created on 10 Jun 50851
2 members
Q
Queen Squad
Created on 25 Aug 50851
2 members
N
NCIT Computer 2015 batch
Created on 26 Aug 50851
1 members
Suwarna Boys Hostel
Created on 8 Oct 50888
5 members

APPENDIX II

The following is the NoSQL database schema used in the application.



APPENDIX III

This section documents the code snippets that were used for some specific purposes in the application.

Registration and Authentication

The following method creates a new account using email and password.

```
mAuth.createUserWithEmailAndPassword(email, password)
.addOnCompleteListener(this, new OnCompleteListener<AuthResult>() {
        @Override
        public void onComplete(@NonNull Task<AuthResult> task) {
          if (task.isSuccessful()) {
                        //user created successfully
          }
          if (!task.isSuccessful()) {
            //error in creating user
          }
        }
});
```

The following method logs in the account with specified email and password.

```
mAuth.signInWithEmailAndPassword(email, password)
.addOnCompleteListener(this, new OnCompleteListener<AuthResult>() {
        @Override
        public void onComplete(@NonNull Task<AuthResult> task) {
          if(task.isSuccessful()){
                        //if signed  in successfully
          }
          if (!task.isSuccessful()) {
            //error in sign In
          }
        }
});
```

The following method is invoked in order to fetch an image from local storage and then crop it.

```
CropImage.activity()
      .setGuidelines(CropImageView.Guidelines.ON)
      .setAspectRatio(1,1) //we want only square images
      .setMinCropWindowSize(500,500)//but not less than 500x500
      .start(this);

if (requestCode == CropImage.CROP_IMAGE_ACTIVITY_REQUEST_CODE) {
   CropImage.ActivityResult result = CropImage.getActivityResult(data);
   if (resultCode == RESULT_OK) {
        //if image is successfully cropped
      } else {
        //if image couldn't be cropped successfully
      }
```