A Project Report

On

AN ONLINE CLOUD BASED CHAT AND LEARN APPLICATION USING FIREBASE CLOUD MESSAGING

Submitted in partial fulfillment of the
requirements for the award of the degree of

MASTER OF COMPUTER APPLICATIONS

SUBMITTED BY

REVU SARATH TEJA

(20A91F0048)

Under the Esteemed Guidance of

Mr.K.C.PRADEEP, M.Tech

Assistant Professor

DEPARTMENT OF MCA

ADITYA ENGINEERING COLLEGE

An Autonomous Institution

(Approved by AICTE, Affiliated to JNTUK & Accredited by NBA, NAAC with 'A' Grade)
Recognized by UGC under the sections 2(f) and 12(B) of the UGC act 1956
SURAMPALEM- 533 437, E.G.Dt,ANDHRA PRADESH.

2020-2022

ADITYA ENGINEERING COLLEGE

An Autonomous Institution

(Approved by AICTE, Affiliated to JNTUK & Accredited by NBA, NAAC with 'A' Grade)
Recognized by UGC under the sections 2(f) and 12(B) of the UGC act 1956
SURAMPALEM- 533437, E.G.Dt, ANDHRA PRADESH.

DEPARTMENT OF MCA

CERTIFICATE

This is to certify that the project work entitled, **"AN ONLINE CLOUD BASED CHAT AND LEARN APPLICATION USING FIREBASE CLOUD MESSAGING"**, is a bonafide work carried out by **REVU SARATH TEJA** bearing Regd.No: **20A91F0048** submitted to the requirements for the award of the Computer Applications in partial fulfilment of the requirements for the award of degree of **MASTER OF COMPUTER APPLICATIONS** from Aditya Engineering College, Surampalem for the academic year **2021-2022**.

Project Guide
Mr.K.C.Pradeep, M.Tech
Assistant Professor,
Department of MCA,
Aditya Engineering College,
Surampalem-533437.

Head of the Department
Mrs.D.Beulah,M.Tech.(Ph.D)
Associate Professor,
Department of MCA,
Aditya Engineering College,
Surampalem-533437.

External Examiner

DECLARATION

I here declare that the project entitled "**AN ONLINE CLOUD BASED CHAT AND LEARN APPLICATION USING FIREBASE CLOUD MESSAGING**", is done and submitted to **Aditya Engineering College , Surampalem** has been carried out by me alone under the guidance of **Mr. K.C.PRADEEP.**

Place: Surampalem

Date:

(REVU SARATH TEJA)
Reg No: 20A91F0048.

ACKNOWLEDGEMENT

The satisfaction and euphoria that accompany the successful completion of any task would be incomplete without the mention of people who made it possible, whose constant guidance and encouragement crowned our efforts with success. It is a pleasant aspect that I have now the opportunity to express my gratitude for all of them.

The first person I would like to thank is my guide **Mr. K.C.Pradeep, M.Tech, Assistant Professor, Department of MCA, Aditya Engineering College, Surampalem.** His wide knowledge and logical way of thinking have made a deep impression on me. His understanding, encouragement and personal guidance have deliverd the basis for this project. He is a source of inspiration for innovative ideas and his kind support is well known to all his students and colleagues.

I wish to thank **Mrs D.Beulah, M.Tech, (Ph.D), Head of the Department, Aditya Engineering College, Surampalem,** who has extended her support for the success of this project.

I also wish to thank **Dr. M.Sreenivasa Reddy, M.Tech, Ph.D, Principal, Aditya Engineering College, Surampalem**, who has extended his support for the success of this project.

I would like to thank all the **Management & Technical Supporting Staff** for their timely help throughout the project.

ABSTRACT

ABSTRACT

The emergence of computer networks and telecommunication technologies allows people to communicate in a new way. Chatting is a technique of exploitation technology to bring individuals and ideas organized regardless of imaginable boundary. Humans are social being that need to communicate regularly. One of the common methods to communicate is to take advantage of the advancement of the digital technology, such as chatting application.

I shaped this application with the partnership conventional to not just be another chat submission, but to enhance a level of clean UI (user interface) or a solid app structure purpose over a secure broadcast network. The Previously excited Chat Submissions momentously influenced the project of the UI (user interface) to give it users a `renewed but familiar UI (user interface) devise. Therefore, to utilize the current digital technology, it is easy to develop a chatting application which has some functions to facilitate the members to communicate among themselves and assist their jobs with add on activities such as payment, learning, to-do list, and games.

I developed such a chattering application using any Java IDE and database. People could use this application to do several things for playing game, learning, make payments, and remainder list. Components used in this project are Java, Android Studios IDE, Google Firebase, Hypervisor.

CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SCREENS

CHAPTER-1
INTRODUCTION

1.INTRODUCTION

1.1 Brief information about the project

The purpose of consultation calls and conversation is to bring individuals and ideas combined, regardless of what is going on more reserved. Irrespective of the fact that the mechanism has endured for quite some time, we have not long ago validated it. The devise illustrates the converse Mariana operation, which runs on any mobile on the Pall network. It may function as a social-networking tool that automates forth movement and, therefore, enables its users to intercommunicate associated erudition technologies. It offers an awful one-stop search expertise to keep bearing with folks you know.

1.2 Motivation and contribution of the project

Internet technologies have enhanced people's ability to access the web fluently. Thanks to technology, we can virtualize still more benefits offered by the cyberspace. Internet contact becomes a part of the everyday lives of people who use it. Usually, when communicating with diverse, one has a face-to-face convention to convey the message.

The objective and goal of creating this work is to deliver a neat UI and set up it easy to make known with personal anywhere in the world by transmitting and entering dispatches in real time with some characteristic like enjoyable elbow grease, to-do inventory and give a defragment entrance to remuneration the person we are associated with employing of pall coffers like google Firebase.

Advantages

- A converse operation that allows one to communication or communicate with a person in real-time. In some cases, it fancily the ground on where guests, implicit leads, prospects could ask inquiries about products or employments, and you can respond incontinent.
- Businesses started transferring emails to guests and implicit guests right after they learned that people use converse operations as a means to intercommunicate with musketeers and family.
- You can seamlessly blend a converse operation into your everyday dockets.
- Using of learn option in menu will helps to keep your mind engaged by learning and also body active. It assists you to increase new guests, develops your intellect to knob a wide range of tasks, and keeps your neurotic trails active.
- Make deadlines for each thing using to-do list point.

- Even though it's pleasant to be fixed, it's not all the time pragmatic. Whenever training your to-do catalog, seek to contain acceptable perspective about anything you can order in a day.

1.3 Objective of the project

Messaging : Theme texting is a moderately new method of virtual message. It uses nearly carefully contemporaneous arrangement chats that license dual particular to connect in real time. In texting, you have a pitch of conversation that you're pledged in. This feature is important as you can enhance people.

Todo list : Owning a list of whole one's prerequisite to-do note dejected in one place wherewithal you shouldn't neglect any considerable. By priority-setting of assignment in the catalog, you formulate the require in which you're exit to do them and can promptly see what desires your hasty consideration and what commission you can get out until a little later. One of the most substantial reasons you should use a to-do list is that it will help you stay organised. When you engrave all your common assignments in a list, they appear to be more governable. When you've turn into a clear structure of the errands you've got to organize and individuals are finished, it helps you stay condensed. While release up space in your head for different more imaginative work.

Payment Gateway : The compensation entry is a manner that lets people to pay for stuff directly upon the operation and endorse other persons to accept virtual compensation. The material of a remuneration entry is the functional that encompass and transmit remuneration data from the customer to the dealer and also transfer the rush deal or lacking return to the visitant. A compensation entry authenticates the customer's card specifics securely, secures the assets that are available, and final allow dealers to get paid. It action as a ophthalmic within a seller's site and its customer. It encipher sensitive credit card details, frosting that information is passed safely from the client to the acquisition bank, via the marketer..

Learn : Computer software scheme is the help of modern life. Imagine for a minute what would take place if all supercomputer abruptly disappeared the next day. No cyberspace, no information, no connection, no convenience, computer programming is a fundamental skill for so many dissimilar applications, not objective software growth or high-technology study into simulated intelligence. It makes investment more accessible, glint our supply lines, and generate those imdelivernt online knowledge. Programming means your required breeches are one click away, and authority can open services quick and more efficiently during a mess.

Paint : Analog painting is a pro rata new, but an already incorporated art form. It's a intermediate that usually unite a computer, a visuals tablet, and programme of choice. The master uses outline and illustration skills with the manner that arrive with the graphics tablet to create paintings within a digital art software.

1.4 Organization of project

- **Chapter -2 :** The **Literature survey** consists of inscribes, cultured articles, and any other sources relevant to a selective issue, area of examine, or thought, and by therefore performing, furnish an description, theoretical, and substantial, estimate of these works in relative to the inspect problem entity explored.

- **Chapter -3 :** The **System Analysis** is a technique of meeting and comprehension facts, distinctive of the difficulty, and crash of a scheme into its portions. I performed a system examination for the goal of examination in a organization or its pieces in order to systematize its purpose.

- **Chapter -4 :** The **System devise** dwell of determine elements of a arrangement like component, structure, elements and their interfaces and data for a organization based on the specific requirements.

- **Chapter -5 :** The **Technology Description** consists of technology used in this project.

- **Chapter -6 :** The **Sample code** consists mainly of sample code for few modules.

- **Chapter -7 :** The **Testing** consists of testing techniques and test cases for modules.

- **Chapter -8 :** The **Screenshots** consists of Output screens of the project.

Conclusion: This is the central conclusion of aforementioned project is to allow user a way to make known and learn.

CHAPTER-2
LITERATURE SURVEY

2. LITERATURE SURVEY

The literature review plays a very vital role in the research process. It is a basis from where research thoughts are careworn and advanced toward conception and finally theories. It also delivers the explorer a bird's-eye view about the done in that area so far. A survey gives an overlooking of a field and is thus differentiated from a sort of study which consists of a microscopic examination of a turf; it is a map rather than a detailed plan. Swinging on what they perceive in the literature review, a geographer will understand where his/her research stands.

1. The Impact Of The Chat: A brief Literature Review by Vicky, Chau Ka Ki

Internet communication is getting more and more popular among the society. Apart from utilizing telephone set or machines and transmission email, public can now express with each other using the conversation technology system. The chat, in fact, is a kind of Internet technology that endorses human-to-human communication. ICQ, for instance, is one of the new chat. Over the previous two years, with the progressive level of technology, there is a growing trend of using ICQ for message. Through ICQ, client can chat, direct mails, records and URLs or perform competitions with other users in real time. Because of the propagation of using the chat like ICQ, studies have been absorbed mainly on its impression on our society.

Much of the effort stresses the upright impact of the chat. Hauben's (1997) lettering recommended that as the impact or inspiration of first impressions is detached, users are free to communicate deprived of fears, limits or uneasiness through the chat. This statement essentially points out the main reason for the progressive use of the chat. Individual one gain, however, seems insufficient to attract such a huge number of handlers to use the chat, so it seems that there may be other benefit. Accordingly, Sicklier (1997) appealed that individuals can communicate online with others who have parallel goals and interests, thus they will augment their life and the announcement will be more productive and more pleasing than. Although Licklider is actually the forecaster of the Net, it seems that the chat really has this benefit.

Some studies, nevertheless, have taken a diverse approach by observing not so much on the rewards of the conversation, but focusing more on its connected difficulties. Randall (1997), for occurrence, references that difficulties have actually been existed. First of all, there is no hesitation that the chat users will not use their real individualizes for communication. They will rather create a new cyberspace individualizes which are very dissimilar from their real ones. Because of this, Randall argued that such conduct makes people hard to switch back and forth between these two characteristics. To him, those who have industrialized multiple cyberspace individualizes for Internet communication are the most cultured speakers on the Internet. In fact, Randall questions whether it is reliable to create a new identity when interactive through the chat.

In Randall's lookout, on the other hand, the determination of people who use the chat is for mingling. But he accentuated that such kind of mingling is different from that in the real world, as the former only involves the conversation of words with other handlers, but the latter means to interrelate with others face-to-face.

While instructors and academic are awaiting internet education to be consisted, Randall has formerly revealed apprehension on the significance of using such kind of education. In his point of view, I will concentrate the outmoded teacher dominance of the classroom on no matter this education is workable or not, because of the poor financial condition of the government. It seems that unemployment will be occasioned in the near upcoming.

To sum up, the chat has decent affected on the culture, but problems happen at the same time. Though, these problems are not thoughtful, in fact. Therefore, even if these problems exist uninterruptedly, the chat knowledge will still become dominant in our lives, and it has previously begun essentially.

2. Commercial Chat apps in 2010s

With the beginning of smartphones, chat apps lingered to flourish; in 2013, chat apps finally exceeded SMS in message volume. By 2015, WhatsApp alone held 30 billion messages per day; SMS logged only 20 billion. And in the seasonal worker of 2016, Facebook Messenger hit one billion users.

When developing one's approach for messaging applications, it's energetic to select the right platform mix for the embattled populace, based on TWO core standards:

Regional Strongholds: Only a minor group of applications like Viber, Facebook Messenger, and WhatsApp may be supposed to be truly worldwide—and even those platforms fight in certain countries. Meanwhile, messengers like WeChat, LINE, and Kakao Talk totally control specific markets but have negligible traction in others.

Demographics: It's a mutual delusion that messaging apps are a consistently millennial singularity. Some apps like Snapchat and LINE reorient both undeveloped and female, but others like Tango (which boasts 100 scullion monthly, active users, by approximation) mostly appeal to those matured 25–54 and powerfully over-index with Latino and African-American users.

FRA: The social broadcasting landscape is entering a retro of hyper disintegration that may be a trial to producers: Facebook, Twitter, and Instagram continue to appear large, but social media directors can now promote official stations on roughly 10 chat apps with over 50 million monthly, active handlers each.

Analytics: For administrations habituated to robust, real-time data, the lack of decent analytics tools for messaging apps remnants a main warning to acceptance. The trial is twofold: Robust analytics consoles take time to shape, and numerous messengers are privacy centric by nature.

Audience development: With billions of lively manipulators across numerous main chat apps, there is the occasion in building large spectators fairly fast on numerous platforms.

Higher engagement: Since numerous chat apps deliver originators with push announcements or chatbot experiences (programmable robots that opposite with operators), they can bring significantly advanced appointment rates.

A chance to connect with users in a new way: Messaging apps proposal a congregation of landscapes not unobtainable on social systems or other podiums. Programmers can imaginatively influence these tools to meet people in new customs.

While messaging is presently an obviously defined purpose of exact apps, they expected the forthcoming to be single, wherein the ability is to disintegrate into approximately all digital know ledges and services. The point anywhere a messaging app starts and ends will begin to distortion. Already, app cataloging is getting thornier, particularly as social media platforms inform their in-app messaging competences, moving them earlier to chat app involvements.

CHAPTER-3
SYSTEM ANALYSIS

3. SYSTEM ANALYSIS

3.1 Existing system

Earlier, there was no manner of online communication among users. In large or small administrations, communication between operators modelled an encounter. There was a prerequisite to record these infrastructures and stock the data for additional evaluation. The awareness is to mechanize the prevailing Simple Chat Room system and establish the users to exploit the software so that their appreciated information is received alphanumerically and can be established for additional management determinations. There was no connected method of communicating with diverse users. There were various unlike interfaces accessible in the market, but this technique of using cloud database makes the information protected and easy to deal with.

In this age of technological development, there are abundant messaging apps that contributors can pick from. They used the messaging apps for conversation purposes. They style communication relaxed with recommendation to such as immediate messaging and speech calls. It can connect the messaging apps on a smartphone irrespective of the smartphone operating system. There are mobile devices that use Windows, Android or iOS operating schemes. Presently, apps are arranged to transfer on the smartphone and talk straight away. Here is a slope of the greatest popular messaging apps worldwide.

Messenger : we can download the app for free from your particular working stock. Currently, it has concluded one hundred billion users. Messenger is a creation of Facebook. The app, also familiar as Facebook Messenger, permit Facebook users to interlinking in existent time over their smartphones. The moral news is that one does not have to be recorded in to Facebook. All the operating systems sustenance this app. However, it has the similar purpose but bears an unlike look depending on the operating system in use.

WhatsApp Messenger : Just similar to the Facebook Messenger, over one billion operators have transferred WhatsApp. As a matter of experience, it is the other largest messaging app universally. After it's achieved by Facebook, the dispatching application has full-grown in leaps and also boundary. It has increased more uses earned to its better-quality structures. What makes WhatsApp viewpoint outward from the repose of the applications is the circumstances that it does not need any notice. Furthermore, it is free and can be charged effortlessly.

WeChat : We drop the roots of this messaging application to China. WeChat is a social discussion tool with supplementary structures. By trembling your smartphone, you can find a new friend nearby. Otherwise, one can use the GPS function too. The app is free, giving growth to the more number of subscribers.

Line : With over 200 million active users, Line is a permitted app that enables users to make free calls and send messages. It is a widespread app in the Asian countries. The messaging app has plentiful and astonishing features, such as line group, video and voice calls, line stickers and line timelines.

Telegram Messenger : It has thought-provoking structures such as the bots. It also has unified mixing between various devices, PCs and tablets included. We store the chats in the cloud and synchronized across all listed devices. However, the major challenge confronted in this app is its failing crypto engraving. It is gaining some level of admiration, though scorn of the contests.

IMO : IMO is an application that has come an extended way. Presently, the app permits operators to make free video and voice calls together with instant messaging. It may not be a widespread app, particularly to the young and teenagers.

Skype : Skype is the preferred video calling app universally. There is a specialized demand too. It proposals instant messaging, which is actually well-organized. It makes transporting files easy, too.

Snapchat : This is an application that has fully fledged extremely over the ages. Youngsters enclosed this deserved to its astonishing Snapchat facts. Instant messaging is the catch. It allows live video chats. Snapchat is free to use.

Viber : Viber is a communal chatting app. It endure some comparability to the other texting applications. Viber's operative can make express claim, send messages and divide pictures with their networks. In 2003, there were 174 million Viber users worldwide. The number of operators has grown over the years. Finally, the app can be uploaded for free.

KIK : It is also one of the widespread messaging app. Unlike other messaging apps, it does not hinge on the phone number of contacts. The operators have usernames, and a group chat can hold up to 50 people. Besides distribution of text messages, KIK operators can share memes, pictures, and videos. The app has redefined immediate messaging. We can download it for free from the individual operating system store, as well as you can use KIK online on the PC.

The messaging apps have developed a significant aspect of our everyday lives. They help us to keep in trace with people who are close or in dissimilar physical locations. The apps have structures that make message exciting outside the consistent messaging services. Operators can share photos, make voice calls and video calls too. The apps have dissimilar supplementary feature giving the user diverse looks and tastes.

3.2 PROPOSED SYSTEM

The proposed system is android application where a user can use it for interactive with the people, to make easy payments, marking off to-do lists, learn to program lessons like java, python and can also play game inside the application. This Simple Chat application will enable the user to chat with the registered users in the application. The server or database used for this application is a product of google cloud system, which is Firebase real-time databases. Which delivers the user to communicate with instant messaging services. We made this application upon on the idea of messenger application to deliver the user entertainment by different way by making some games to play. Also, at the same term, user need to get their work done without addicting to the application by providing to-do list. This application is a social-networking tool that leverages on technology advancement, thereby allowing its users to communicate.

Advantages

The main advantages belongings I hope to accomplish with this application are,

- Speed in usage
- Easy and friendly UI
- Privacy Protection
- Promoting Unity
- Economic boost
- Easy learning
- Stress reliever

3.3 Feasibility Study

We reviewed the feasibility of the project in this phase and we put the commercial proposal forth with a very wide-ranging plan for the project and several cost calculate. During system analysis, the practicability study of the planned arrangement is to be out. This is to confirm that the proposed system is not a load for the company.

For the feasibility analysis, some considerate of the major requirements of the scheme is essential. The key deliberations involved in the feasibility analysis are:

- Economical Feasibility
- Technical Feasibility
- Operational Feasibility

Economic Feasibility

Someone can industrialize technically a system and such will be cast-off if established must still be a virtuous speculation for the organization. In the economical feasibility, we appraise the expansion cost by creating the system against the definitive benefit resultant from the new systems. Financial benefits require equal or outshine the costs. Economic usefulness defines whether the predictable benefit equals or surpasses the predictable costs. It is also usually referred to as cost/benefit analysis. The process is to determine the supports, and the savings expected from the system and equivalence them with the costs. It forestalled a proposed system to offset the costs.

This is a minor project with no cost of development. I relaxed the system to understand and use. Consequently, there exists no need to assign on training to use the arrangement. This arrangement has the potential to grow by adding functionalities for students as strongly as teachers. This can, Hence, the project could have economic benefits in the future.

Technical Feasibility

We distributed this study to test the specialized feasibility, that is, the specialized conditions of the system. Any system industrialized mustn't have a high demand for the available procedural coffers. This can achieve high demands on the available specialized coffers. This may beget high demands being placed on the customer. The advanced system must have a modest demand, we need as only minimum or null changes for enforcing this fashion. Also, whether the devise is possible in terms of labor force, and moxie, to handle the completion of the devise. It considers determining coffers for the proposed system.

As we develop the system using java, it's platform independent. Thus, the druggies of the system can have average processing capabilities running on any small specified platform. The technology is one of the rearmost, hence the system is also technically doable.

Operational Feasibility

The study's feature is to assess the user's level of understanding of the system. This includes training the user to administer the technology professionally. The user should not feel threatened by the system, but rather embrace it as a need. The operational feasibility of a given system is a measure of how successfully it addresses difficulties with the operators. The operational feasibility of the project is dependent on the human resources available for the project and entails determining if we will embrace the system care if it is created and implemented. The amount of adoption by users is entirely dependent on the methods used to educate the operator about the system and familiarise them with the application.

3.4 Functional Requirements

Login: The login delivers a form to users with email, password and forgot fields. If the user enters the valid username/password combination, they will be boosting access to the website with additional resources.

Sign Up: The sign-up delivers a form to users with username, email, and password fields. If the user enters the valid combination of details, then they will be registered to the application with additional resources.

Forgot: The forgot delivers a form to the users with email fields. If the user enters the valid combination of mail address then they will get a message to their registered mail address where there itself the user can make changes to their password of their account.

Add list: The add list is a to-do list feature that displays a form with empty fields to users. If the user enters the data, it will be concealed within the phone's memory.

Game: The game is characteristic providing the user with a game to play.

Transaction: The method is for pre-authorized transactions, which delivers a communication device to the vendor and the card owner. The card owner initializes the transaction by communicating with the card number and storing a piece of information that categorizes the specific transaction made by the authorized user of the card at a later time. Also delivers UPI payments for the users.

The following are the functional requirements of the system:

- The system shall deliver the user the ability to create a new account.
- The system shall deliver the user the ability to log in with the username and password chosen at the time the account was created.
- The system shall deliver the sender's communication to the receiver immediately if the receiver is networked.
- The system shall deliver sender's message to the receiver once the receiver is back online if the message was sent when receiver's offline

3.5 Non-Functional Requirements

The following are the non-functional requirements of the system:

- The system shall be a web-based application that can offer all the purposes over the cyberspace.
- The system shall transport messages in the same order it becomes sent out.
- The system shall assurance the distribution. And ought detect originator if message is not delivered efficaciously.
- The system ought be extensible and full-bodied.
- The system take be value skillfully. It shall not cost a lot if there are not many operators.
- The system shall deliver the message moderately fast.

3.6 Performance Requirements

The presentation of the application is controlled, fashioned on the time delay and bundle drop rate for each note and the response time for user's actions.

- The login and account data loading time should be less than 3 seconds.
- The time in between message directed and established should be less than 1 seconds when total online operator's number < 10000, and less than 10 seconds with unlimited number of total online operators.
- The released package rate for video/audio chat should be less than 1%

3.7 Database Requirements

Google Firebase is used to store all the information we want to keep, especially data that does not need to be changed frequently, such as user authentication and authorization information, such as username, password. The project uses the cloud real-time database to use both storage and caching.

3.8 Requirements Specification

3.8.1 Hardware Requirements

The Hardware Requirements to develop the application :

- Processor : 2.0 GHz or above
- RAM : 8 GB RAM or more
- Hard Disk : 50 GB of available disk space minimum

The Hardware Requirements for the application :

- Device : Android Version 5.0 or above
- Processor : 1.5 GHz or above
- RAM : 2 GB RAM
- Hard Disk : 8 GB of available disk space minimum

3.8.2 Software Requirements

The Software Requirements to Develop & Run the application :

- Language : JDK (8.1)
- IDE : Android Studios (Chipmunk)
- Operating System : Windows(8/10), Android 5.0 or above

CHAPTER-4
SYSTEM DESIGN

4.SYSTEM DESIGN

4.1 Introduction

The decision of the intention stage is to arrange a solution of the delinquent deviseated by the essential document. This phase is the primary step in shock from the unruly section for the retort domain. In accessory terms, launch with what is needful project takes us towards how to gratified the inevitably. The devise of a organization is conceivably the most grave factor liking the renown of the program it has a central effect on the next step, generally testing, support. The relinquish of such phase is the venture paper. This chronicle is alike to a pattern for the retort and is exerted later in the direction of capital punishment, experiment and continuation. The devise movement is much split into two individual phases, System Plan and Complete devise.

System devise also called top-level devise objective to place the components that ought to be within the system, the stipulation of those components, and the way they interrelate through single additional to deliver the wanted results. At the conclusion of the system devise all the most data structures, file formats, output performances, and therefore the foremost modules within the system and their stipulation are obvious. During, Comprehensive devise, the within logic of every of the components stated in system devise is set. During this stage, the particulars of the info of a module is sometimes laid out in a high-level strategy description language, which is self-governing of the target language during which the software will in conclusion be performed. In system devise the main target is on identifying the modules, whereas during comprehensive devise the main target is on fashioning the logic for every of the components. In other words, in system devise the eye is on what components are needed, while in detailed devise how the modules can be implemented in software is the issue.

devise is anxious with identifying software components, specifying relationships among components. Specifying package building and providing blueprint for the document segment. Modularity is one among the desirable properties of enormous systems. It implies that the system is split into several parts. In such a way, the interaction between parts is minimal, clearly specified. During the system devise activities, Developers cross the gap amidst the wants' specification, produced during requirements elicitation and analysis, and therefore the system that's delivered to the user. devise is that the place where the standard is fostered in development. Software devise may be a process through which requirements are translated into a representation of software.

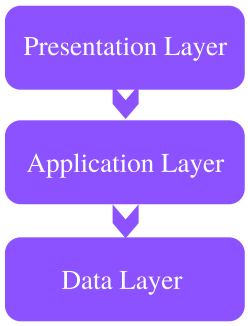


Fig -1 : System devise Layers

Presentation Layer

This layer is responsible for the chromatic interface. All the factors that druggies see and interact with within the operation are in this subcaste.

Application Layer

Application layer controls the overall functionality of the given application. Functionality similar as entering into the system is each done in this subcaste.

Data Layer

In this layer, Data and Information are stored and recaptured in the database. The names, images of operators are stored in the database. Once the user sends or receives mails, it forwards the data to operators matched in the database.

4.2 System Architecture

While the arrangement devise of a talk application is special in how it agreement with the distinctive business requirements, you'll be able to constantly break it all this way downhearted to two main element: the chat guest and also the chat waitperson. The chat guest is what the user encounter. In a desktop, network or smartphone chat demand, the chat guest is accountable for interface with the program (i.e. your computer, browser or device). Relationship include dispensation of push notifications, exhibiting data to the operator, and storing messages and records. Once you blood type the message and hit send, the chat client transmits that message to the opposite major component: the chat server.

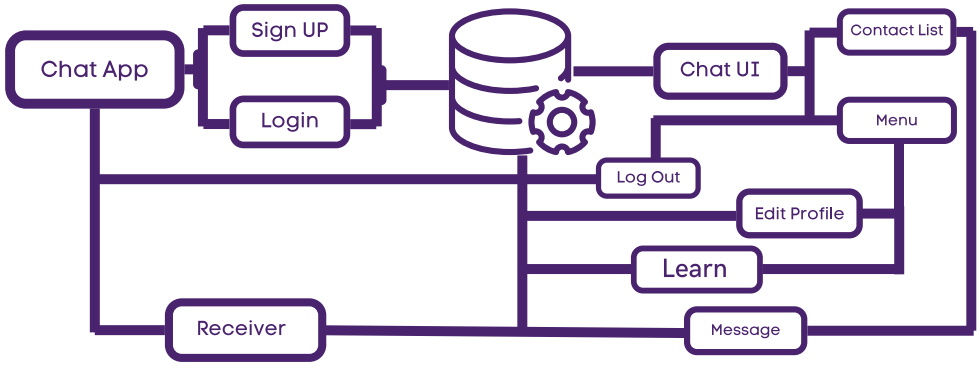


Fig -2 : System Architecture

The chat server is simply that, a server (or usually many servers) that hosts all the software, frameworks and databases necessary for the chat app to work. This server, or pool of servers, is liable for securely receiving a message, identifying the right recipient, queuing the message so forwarding the message to the recipient's chat client. The chat server's resources can include a REST API, a Web Socket server, an AWS instance, Google fire base for media storage, etc.

The Android software stack usually consists of a Linux kernel and an assembly of C/C++ libraries that is uncovered through an application framework that delivers services, and management of the applications and run time. Android was shaped on the open source kernel of Linux. One main reason for selecting this kernel was that it deliverd confirmed core structures on which to develop the Android operating system. The features of Linux kernel are:

Security: The Linux kernel handles the security among the application and the system.

Memory Management: It proficiently handles the memory management, thereby providing the autonomy to develop our apps.

Process Management: It manages the process well, allots resources to processes every time they need them.

Network Stack: It excellently switches the network communication.

Driver Model: It certifies that the application works. Hardware constructors can figure their motorists into the Linux build.

Running on the top of the kernel, the Android framework was established with numerous features. It contains of innumerable C/C++ core libraries with several of open source tools. Some of these are:

The Android runtime: The Android runtime contain of essential libraries of Java and ART(the Android Run-Time). Older versions of Android (4 and before versions) had Davis runtime.

Open GL(graphics library): This cross-language, cross-platform application driver boundary (API) is used to produce 2D and 3D computer visuals.

WebKit: This open source web browser engine delivers all the functionality to establish web content and to simplify page loading.

Media frameworks: These libraries let you play and record audio and video.

Secure Socket Layer (SSL): These public libraries are there for Internet security.

Android Runtime is the third segment of the architecture. It offers one of the important components which is called Davis Virtual Machine. It acts like Java Virtual Machine, which is intended particularly for Android. Android uses its own convention VM, planned to ensure that several instances run efficiently on a single device.

The Delvik VM customs the stratagem's underlying Linux kernel to switch low-level functionality, counting safekeeping, threading and memory management.

The Android crew has built on a recognized set of confirmed libraries, built in the contextual, and all of it This is uncovered through Android boundaries. These edges warp up all the various public library and make them beneficial for the Developer. They don't have to shape any of the functionality deliverd by the android. Some of these boundaries include:

Activity Manager: It accomplishes the activity development and the motion stack.

Telephony Manager: It offers admittance to telephony facilities as connected subscriber information, such as phone numbers.

View System: It shapes the operator interface by conduct the views and arrangements.

Location manager: It finds the device's physical location.

Android applications can be originated at the uppermost layer. At application layer, we inscribe our application to be installed on this layer only. Examples of submissions are Games, Mails, Contacts etc.

Cloud computing is the distribution of computation facilities—including servers, storage, databases, networking, software, analytics, and astuteness—over the Cyberspace ("the cloud") to proposition quicker revolution, flexible resources, and economies of scale. Cloud computing technology is used by both minor and huge administrations to store the data in cloud and contact it from anywhere at anytime by means of the internet connection. Physical phenomenon computing architecture is a grouping of service-oriented architecture and event-driven architecture.

Cloud computing architecture devise is divided mainly into the following two parts -

- Front End
- Back End

The below diagram shows the complete architecture of cloud computing system

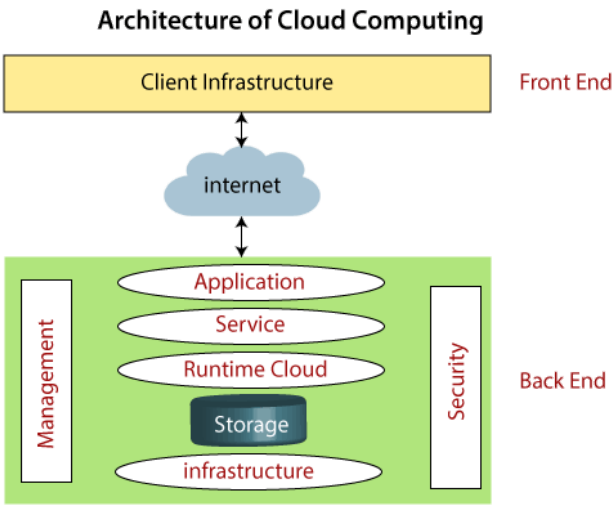


Fig -3 : Cloud Architecture

The front end is used by the client. It covers client-side edges and applications that are essential to access the cloud computing platforms. The front end embraces web servers (including Chrome, Firefox, Internet Explorer, etc.), reedy & portly clients, tablets, and mobile devices.

The back end is used by the service end of the deliverr. It achieves all the capitals that are essential to offer cloud computing services. It comprises a vast amount of data storage, security mechanism, virtual machines, organizing models, servers, traffic control mechanisms, etc.

The foremost component of the project which is cloud storage FCM depend on the following set of components that build, transport, and receive messages:

Firebase isn't just a somewhat normal database. As a real-time, scalable backend, fire base offer the apparatuses you need to quickly build rich, collective applications that can serve millions of operators. Many Firebase-powered apps contain of only client code, and don't need anything other than Firebase and a way to allocate your app to work.

Tooling to combine or build communication requirements. The Notification's composer offers a GUI-based option for making notification requests. For full computerization and support for all message types, you must shape message requests in an important server environment that provisions the Firebase Admin SDK or the FCM server protocols. This atmosphere could be Cloud Functions for Firebase, App Engine, or your own app server.

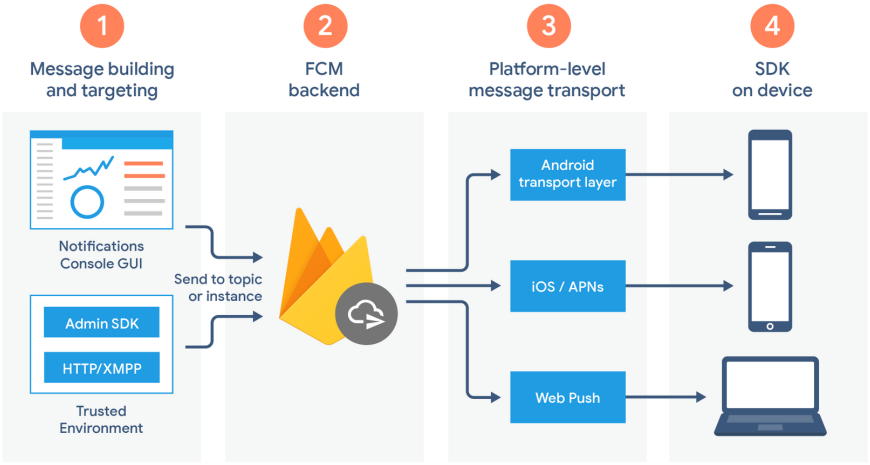


Fig -4 : Firebase Architecture

The FCM backend, which admits message requests, performs fan out of messages via topics, and generates message metadata such as the message ID.

A platform-level transportation layer, which directions the message to the embattled device, handles message delivery, and put on platform-specific configuration where suitable. This transport layer includes:

- Android transport layer (ATL) for Android diplomatic with Google Play Services.
- Apple Push Notification service (APNs) for Apple campaigns.
- Web push protocol for web apps.

The FCM SDK on the operator's device, where the announcement is showed, or the message is touched giving to the app's foreground state any applicable submission logic. An architecture in achievement would be clients placing tasks for the server to process in a file. The user can have single or more servers picking off items from the file whenever they have resources available, and then place back the result into your Firebase database, so the clients can deliver them.

Architecture Description

The system architecture will consists of the following steps:

- Installing Application.
- Login / Sign up.
- Data Storage.
- Chat UI.
- Contact list.
- Messaging.
- Menu (features of applications)
- Logout.

4.3 Modules Description

The complete components of system can be divided in following modules:

- User Management Module
- User Registration Module
- User Validation Module
- Change Password Module
- Administration Module
- Server Module
- Payment module
- Library management Module

User Management Module

The User Management Module enables the managing scope of to access the application and the content available and visible to dedicated users. This module has two submodules which are named as registration and validation modules.

User Registration Module

You can enable users to create a new user account by clicking New User Registration Module. This Module helps you to create and devise the perfect user registration forms on the application. Also have the option of redirecting the user to Custom URL, auto-login, and sending an email after successful registration using this module.

User Validation Module

The Validation's module delivers a relaxed out-of-the-box method to confirm that the content of a message in your movement matches a specified set of measures. The main advantage this has over with filter out is traceability, as strainers all raise indistinguishable exclusions, making it hard for the user to know where the exclusion was caused. Validators, on the further hand, raise an exemption with a meaningful message involved. User can optionally modify this message and even the type of exclusion user want it to throw.

Change Password Module

Users can modify their passwords each time they want by using Self Provision Password Reset. Self Service Password Reset allows managers to modify the password change practice for the operators from the supplication to the end. The Change Password component allows you to organize actions the users must perform before fluctuating their password.

Administration Module

The Administration Unit, only for administrators operators, is proposed for the startup of the basic conformation and the consecutive alterations. You practice the applications in the Administration unit for system administration responsibilities. You can perform functions such as creating and deleting of information in database.

Server Module

The Server Module allow the operators to communicate through the server, also consist of the Broadcasting Module where it is used for chatting like broadcasting and display their inbox and outbox and list of users.

Payment Module

A payment unit is a group of payment structures and settings, repeatedly made available by third parties. For example, Razor Pay, PayPal, Amazon Payment, Gpay are instances of payment modules. Payment Segment used During counter, where a user will send the payment data to a gateway company which can verify a operation for any of the major credit cards and return a pass or fail.

Library Module

Library Management Module deals with the build of the project dependencies, which are used to create and use functions or methods of specific features in the applications. The library management used for these projects is Gradle build where we can add dependencies in order to use their functions and functions example of dependencies are fire base, EasyUpiPayments and many more functions are used to use their functions to add features for the applications.

4.4 UML Diagrams

- The unified modelling verbal allows the software program locomotive engineer to express an analysis typical model using the modeling symbolization that is administered by a set of syntactical semantic and pragmatic rules.
- A UML system is characterized using five diverse opinions that define the system from distinctly dissimilar perspective. UML is exactly constructed through two different way of domains.

- UML Analysis study modeling, these concentrations on the user model and structural model opinions of the system.
- UML devise modelling, which emphasis on the communication modelling, execution modeling and environmental model views.

These are divided mainly into the following type case diagrams:

- Use case diagram
- Class diagram
- Sequence diagram
- Collaboration diagram
- Activity diagram

4.4.1 Use Case Diagram

Use Case diagrams recognize the functionality deliverd by the system (use cases), the operators who interrelate with the structure (actors), and the connotation among the operators and the functionality. Use Cases are used in the Examination phase of software development to articulate the high-level requirements of the system. The main goal line of Use Case diagrams include:

- Providing a high-level opinion of what the organization does.
- Recognizing the users ("actors") of the system.
- Defining areas needing human-computer interfaces.

Graphical Notation: The elementary workings of Use Case diagrams are the Actor, the Use Case, and the Association.

Actor	An Actor, as declared, is a operator of the system, and is portrayed using a stick figure. The role of the operator is written below the icon. Actors are not accurate to humans. If a system interconnects with an alternative application, and imagines input or transports output, then that application can also be considered an actor.	

Use Case	A Use Case is a functionality that tendered by the system; Use Cases are portrayed with a hollow shaped figure. The deviseation of the use case is printed within the circle	 **Use Case**
Directed Association	These Connotations are castoff to link Actors with Use Cases, and deviseate to that an Actor contributes to the Use Case in some form.	

Behind each Use Case is a sequence of movements to achieve the proper functionality, as well as alternative trails for instances where validation fails, or errors occur. These activities can be further defined in a Use Case explanation. Since this is not addressed in UML, there are no canons for Use Case metaphors. However, there are some communal templates can follow, and whole books on the subject lettering of Use Case description.

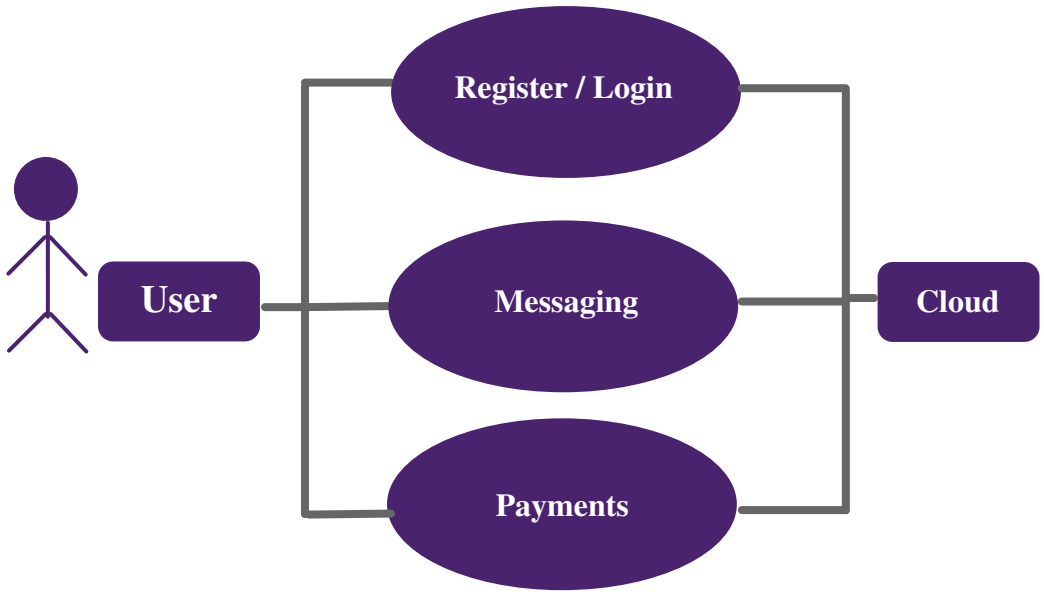


Fig 4.4.1.1 Use Case Diagram

Description:

The above shown diagram represents the use case diagram of this project. In the above use case diagram it describes about the relation between the user, login and verification.

Use case Templates

Usecase name	Login
Participating actors	User
Flow of events	The system asks the user to enter his/her name and credentials. The user enters his/her name and credentials. The system validates and checks the entered name and password and logs the operator into the system.
Entry condition	None
Exit condition	If the use case is successful and matched in the database, the user is logged and enters into the system. Else the state is constant.

Table 4.4.1.1 Use case template for Login

Usecase name	Verification
Participating actors	User
Flow of events	The system asks the user to enter his/her value of details. The user types his/her details. The system validates and checks the entered details and verifies the process.
Entry condition	None
Exit condition	If the use case is successful, the operator is a transaction into the system. Else, the system state will be unchanged.

Table 4.4.1.2 Use case template for Verification

Level 0	Level 1	Level 2	actor
Chat application	Authentication system	Register	User
Chat application	contact form	Friend list	User
Entry condition	chat form	logout	User
Exit condition	maintenance	user profile Database	admin

Authentication Service

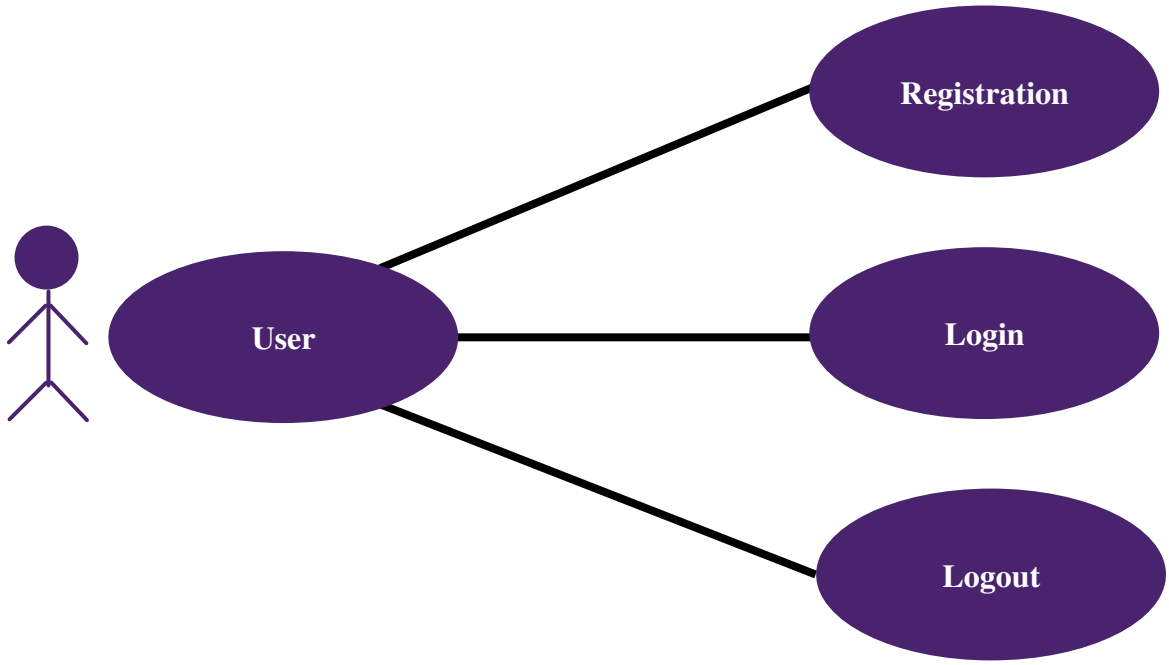


Figure 4.4.1.2: Use Case Diagram of Authentication Service for User

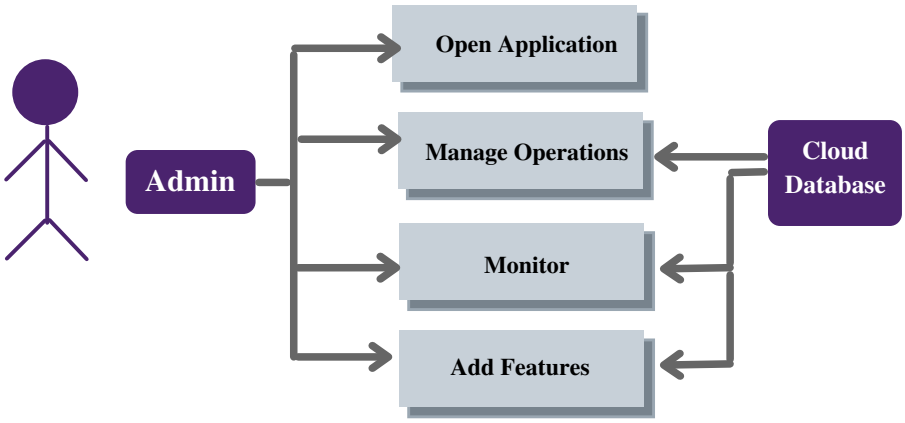
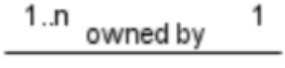


Figure 4.4.1.3: Use Case Diagram of Admin Control Service

4.4.2 Class Diagram

Class diagrams recognize the class building of a system, counting the properties and methods of each session. Also portrayed are the various associations that can happen between classes, such as an inheritance relationship. Part of the admiration of Class diagrams branches from the fact that many CASE tools, such as Balanced XDE, will auto-generate code in a diversity of languages, these tools can coordinate models and code, reducing the workload, and can also produce Class diagrams from object-oriented code.

Graphical Notation: The elements on a Class diagram are classes and the relationships among them.

Class	Classes are the blocks in object-based technique scripting programming. A Class is showed using a cube separated into three sections. The top unit is the name of the Class. The middle unit defines the parcels of the Class. The last section lists the styles of the class.	
Association	An Association is a general form relationship between two classes, and is modeled by a line connecting the two classes. This figure can be good with the type of relationship, and can also feature a multifariousness rule(e.g. one- to- one, one- to- numerous, numerous- to- numerous) for the relationship established.	

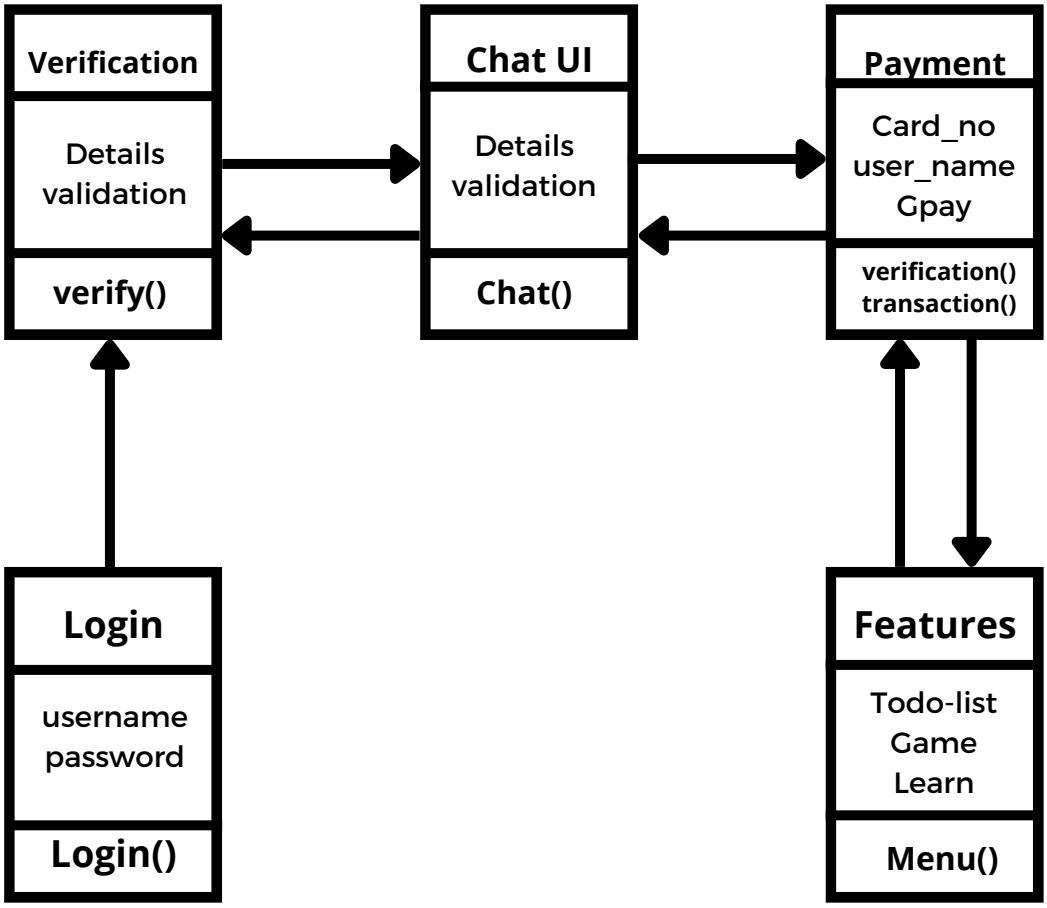


Fig 4.4.2.1 Class Diagram for System

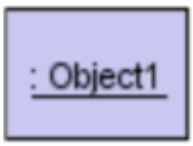
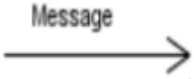
Description:

In the above diagram represents the class diagram of this project, "An Online Cloud Based Chat and Learn Application Using Firebase Cloud Messaging". In the above class diagram, it describes the relation between login, transaction, verification, features and security info.

4.4.3 Sequence Diagram

Sequence diagrams document the communications amongst classes to accomplish a result, such as a use case. Because UML is planned for object-oriented programming, these communications amongst classes are known as messages. The Sequence diagram lists objects straight, and time precipitously, and models these messages concluded time.

Graphical Notation: In a Sequence diagram, classes and actors are listed in columns, with vertical lifelines and denoting the overtime of the object concluded time.

Objects	Objects are examples of the classes, and they are arranged horizontally as shown. The graphic representation for an Entity is a class (a rectangle) with the name preceded by the object name (optional) and a punctuation	
Lifeline	The Salvation recognizes the existence presence of the object over time. The notation for every Lifeline is a vertical-dotted line ranging from an object.	
Activation	Activations, demonstrated as quadrangular containers on the salvation, deviseates when the object is accomplishing an action.	
Message	Messages, demonstrated as straight parallel arrows amongst Activations, direct to the communications amongst objects.	

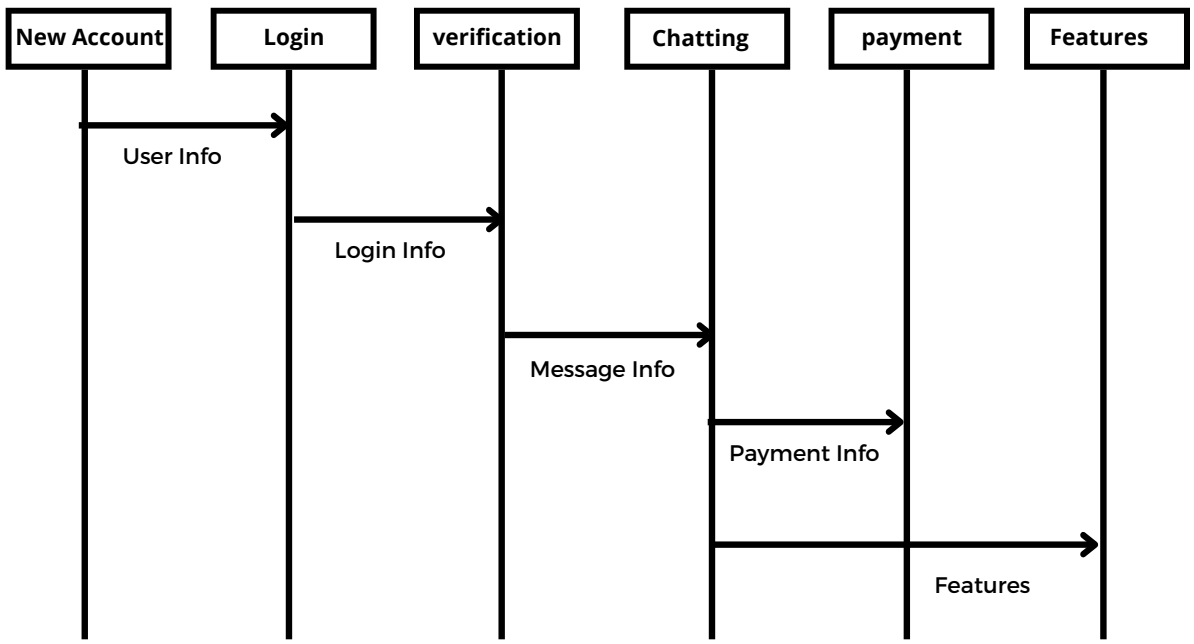


Fig 4.4.3.1 Sequence Diagram for System

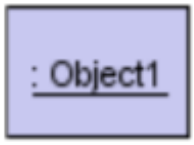
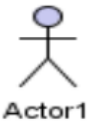
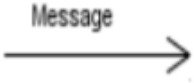
Description

In the above drawing stand for the sequence diagram of the project "An Online Cloud Based Chat and Learn Application Using Firebase Cloud Messaging". In the preceding sequence diagram, it describes the new account, login, transaction, verification, security and complete transaction acts as objects.

4.4.4 Collaboration Diagram

Like the other Behavioral diagrams, Collaboration diagrams model the connections among substances. This type of illustration is a cross among an object diagram and a sequence diagram. Unlike the Sequence drawing, which internal representation the communication in a column and row type format, the Collaboration diagram uses the free-form arrangement of objects as found in an Object diagram. These kinds exist calmer to see all connections involving a particular object.

Graphical Notation:

Objects	Objects are occurrents of forms, and stay one of the nonfictional prose kinds that can be complicated in communications. An Object is haggard as a polygonal shaped box, with the class term inside started with the object name (optional) and a semicolon.	
Actor	Actors can similarly interconnect with Objects, so they too can be recorded on Collaboration diagrams. A Performer is portrayed by a stick figure.	
Message	Messages, showed as darts among objects, and characterized with a gathering number. Indicate the transportations among objects.	

Description

In the below diagram represents the Collaboration diagram of the project "An Online Cloud Based Chat and Learn Application Using Firebase Cloud Messaging". In the above Collaboration diagram, it describes the new account, login, transaction, verification, security and complete transaction acts as objects.

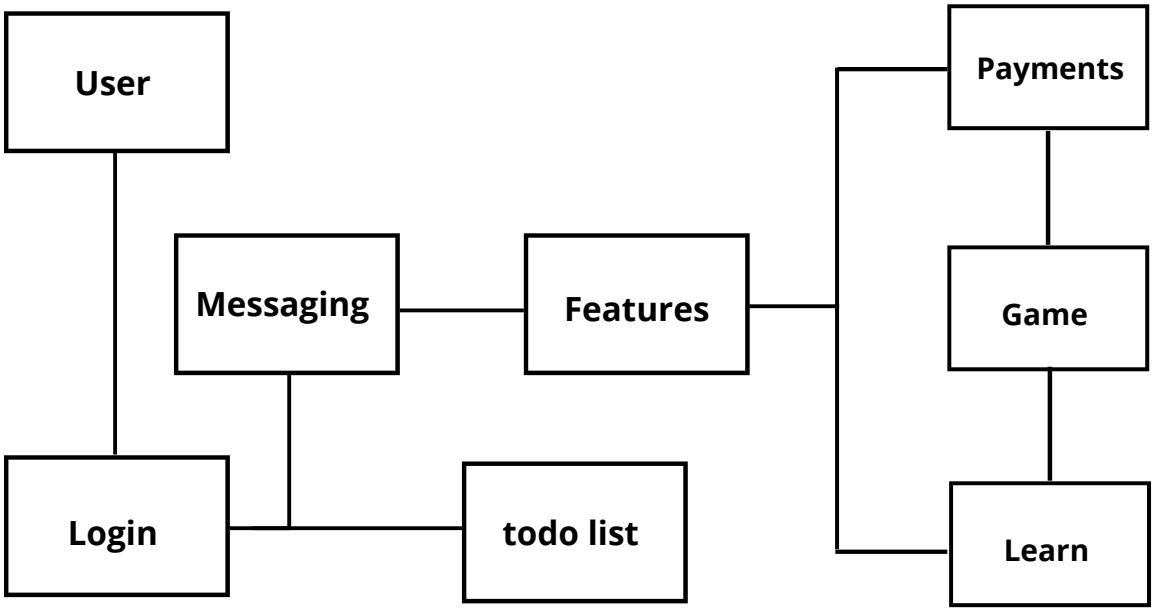


Fig 4.4.4.1 Collaboration Diagram for System

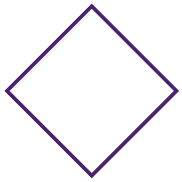
4.4.5 Activity Diagram

This demonstrates the flow of events within the system. The activities that happen within a use case or within an object's behavior typically happen in a sequence. An activity diagram is intended to be a simplified look at what happens during an operation or a process.

Each activity is qualified by a smooth-edged parallelogram the permission within an activity goes to compilation and then an automatic broadcast to the next activity occurs. An arrow represents the changeover from one activity to the following. An activity diagram describes a system in terms of actions. Activities are the state that characterises the execution of a set of processes. These are similar to flow chart diagram and data flow character diagrams.

Starting point	This is main starting point of process.	
Action	An accomplishment represents the implementation of a nuclear action, naturally the supplication of an operation. An achievement is an unpretentious action with an entry achievement whose only exit evolution is prompted by the contained event of effecting the accomplishment of the entry action.	

Description

End point	Any termination point epitomizes the last or "final" activity of the envelopment merged activity. There might be further than one ending point at any level, portentous that the merged activity can end in dissimilar ways or conditions. When a last activity is touched and there is no other encircling activity, it means that the entire activity has accomplished its conversions and no more transitions can occur.	
Decision	An activity diagram premium rate choice when sentry conditions are used to specify different conceivable transitions that hinge on Boolean conditions of the owing object.	
Synchronization bar	The fork will split the condition into various conditions and the intersection will merge the multiple states into solitary states	
Swim lane	The plunge lane will divide the commercial objects into connotation full order.	
Transition	A conversion is a directed affiliation between a source state vertex and a goal state vertex. It may be part of a complex transition, which receipts the inert machine from one static conformation to another, on behalf of the complete response of the static apparatus to a particular event instance.	

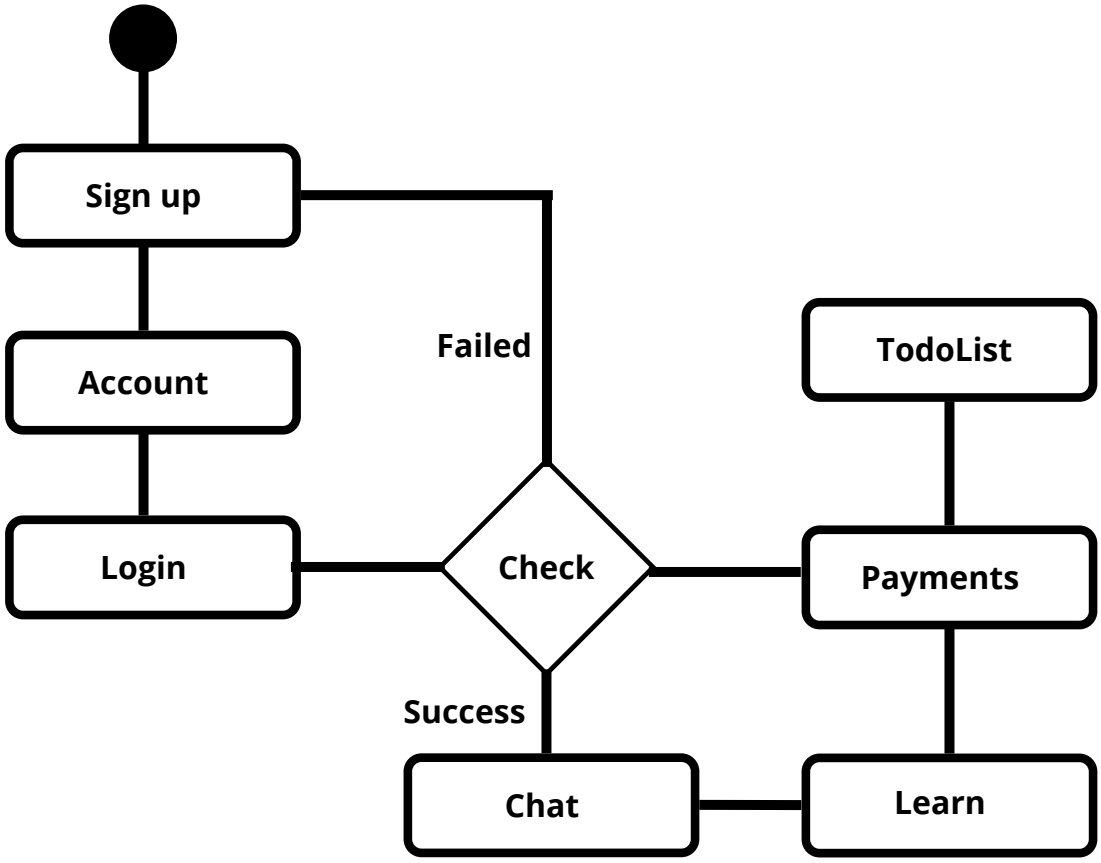


Fig 4.4.5.1 Activity diagram for system

Description

In the above diagram represents the Activity diagram of the project "An Online Cloud Based Chat and Learn Application Using Firebase Cloud Messaging". In the above Activity diagram, it describes the card and user activities are shown. The user creates a new account, opens the account, login, purchases the product, pays for the product, verifies the transactions, check for security – whether the transaction is valid or not, security is added to the transaction and transaction is complete. Once the user logged into the account, user able to use the features of applications like learn, To-do List and paint as features of the application.

CHAPTER-5
TECHNOLOGY DESCRIPTION

5 .TECHNOLOGY DESCRIPTION

Tools and Technologies

Tools and techniques used in the project are described in this section of the thesis. This project focused was mainly focused on Java Programming.

5.1 Introduction to java

JAVA was technologically advanced by James Gosling at Sun Microsystems Inc in the year 1995, later developed by Oracle Corporation. It is a simple programming language. Java styles writing, compiling, and debugging programming relaxed. It helps to generate reusable code and flexible programs.

Java is a class-based, object-oriented programming language and is intended to have as few operation dependencies as conceivable. A general-purpose programming language made for deviseers to write once run anywhere that is compiled, Java code can run on all platforms that support Java. Java applications are compiled to byte code that can run on any Java Virtual Machine environment. The syntax of Java JDK is so similar to c/c++.

History

Java's history is very thought-provoking. It is a programming language fashioned and formed in 1991. James Gosling, Mike Sheridan, and Patrick Naught on, a team of Sun engineers known as the Green team, started the Java language in 1991. Sun Microsystems released its first community claim in 1996 as Java 1.0. It delivers no-cost -run-times on popular stages. Java compiler was re-scripted in Java by Arthur Van Hoff to firmly comply with its stipulation. With the arrival of Java 2, new versions had multiple conformations built for different kinds of platforms.

In 1997, Sun Microsystems advanced the ISO canons body and later pompous Java, but it soon removed from the process. At one time, Sun made maximum of its Java applications available deprived of charge, despite their registered software status. Sun generated revenue from Java through the marketing of licenses for dedicated products such as the Java Enterprise System.

On November 13, 2006, Sun unconfined much of its Java virtual machine as free, open-source software. On May 8, 2007, Sun finished the development, making all of its JVM's core code obtainable under open-source distribution terms.

The principles for generating java were unassuming, robust, secured, high performance, portable, multithreaded, read, dynamic, etc. In 1995 Java was industrialized by James Gosling, who is known as the Father of Java. Presently, Java is used in mobile devices, internet programming, games, e-business, etc.

Before learning Java, one should be familiar with these most common terms of Java.

1. Java Virtual Machine(JVM): This is generally referred to as JVM. There are of mainly three execution phases of a program. They are scripting, compile and running the program

- Writing a program is done by a java programmer by operators.
- The compilation is done by the JAVA C compiler, which is a mainly Java compiler included in the Java development kit (JDK). It takes the Java program as input and obtains byte code as a result.
- In the running stage of a program, JVM runs the byte code generated by the compiler.

Now, we understood that the purpose of Java Virtual Machine is to perform the byte code produced by the compiler. Every Operating System has a dissimilar JVM, but the output they crop after the implementation of byte code is the same across all the operating systems. This is why Java is recognized as a platform-independent language.

2. Byte code in the Development process: As conversed, the Java compiler of JDK compiles the java source code into byte code so that it can be performed by JVM. It is protected as a class case by the compiler. To view the byte code, a disassembled similar Java can be used.

3. Java Development Kit(JDK): Though we were with the term JDK when we learn about byte code and JVM. So, as the appellation suggests, it is a whole Java improvement kit that includes the lot including compiler, Java Runtime Environment (JRE), java debuggers, java docs, etc. For the program to implement in java, we need to connect JDK on our electronic computer in order to produce, compile and run the java program.

4. Java Runtime Environment (JRE): JDK contains JRE. JRE connection on our computers allows the java program to run, nevertheless, we keep compile it. JRE contains of a browser, JVM, applet supports, and plugins. For successively the java program, a computer needs JRE.

5. Garbage Collector: In Java, a computer operator can't delete the objects. To delete or remember that memory, JVM has a program called Garbage Collector. Garbage Collectors can recall the objects that are not referenced. So Java styles the life of a programmer easy by management memory management. Though, computer operator should be cautious about their code whether they are by means of objects that have been used for a long time. Because Garbage cannot recuperate the memory of objects being referenced.

6. Class Path: The class path is the folder path wherever the java runtime and Java compiler aspect for a class files to freight. By avoidance, JDK delivers countless libraries. If you want to contain external libraries, they should be supplementary to the class path.

5.2 Features of java

1. Platform Independent: Compiler changes source code to byte code and then the JVM performs the byte code generated by the compiler. This byte code can run on any platform be it Windows, Linux, macOS which resources if we compile a program on Windows, then we can run it on Linux and vice versa. Individually operating system has a different JVM, but the output formed by all the OS is the same after the implementation of byte code. That is why we sound java a platform-independent language.

2. Object-Oriented Programming Language: Establishing the program in the terms of an assortment of objects is a way of object-oriented programming, each of which signifies an instance of the class.

The four main concepts of study of Object-Oriented programming are:

- Abstraction
- Encapsulation
- Inheritance
- Polymorphism

3. Simple: Java is one of the simple and easy languages as it does not have difficult features like pointers, operator overloading, multiple inheritances, Explicit memory allocation.

4. Robust: Java language is full-bodied, which means reliable. It is established in such a way that it puts a lot of exertion into checking faults as early as possible, that is why the java compiler is clever to detect uniform those errors that are not easy to distinguish by another programming language. The core topographies of java that make it robust are garbage collection, Exception Handling, and memory allocation.

5. Secure: In java, we don't have pointers, so we cannot contact out-of-bound arrays , it displays ArrayIndexOutOfBound Exception if we attempt to ensure so. That's why numerous security defects like stack corruption or buffer run-off are impossible to exploit in Java.

6. Distributed: We can generate scattered applications using the java programming language. Remote Technique Petition and Enterprise Java Beans are rummage-sale for producing distributed applications in java. The java programs can be simply circulated on single or further systems that are linked to each other through an internet connection.

7. Multithreading: Java provisions multithreading. It is a Java feature that lets simultaneous execution of two or more parts of a program for extreme consumption of CPU.

8. Portable: As we know, java code written on one machine can be executed on another machine. The platform-independent feature of java in which its platform-independent byte code can be engaged to any platform for execution varieties java portable.

9. High Performance: Java architecture is well-defined in such a way that it diminishes overhead throughout the runtime and at some time java uses Just In Time (JIT) compiling program where the compiler compiles code on-demand fundamentals where it only accumulates those methods that are called making applications to execute faster.

10. Dynamic flexibility: Java being completely object-oriented gives us the flexibility to add classes, new methods to existing classes and even create new classes through subclasses. Java even supports functions written in other languages such as C, C++ which are referred to as native methods.

11. Sandbox Execution: Java programs track in a dispersed space that lets user implement their applications deprived of affecting the underlying system with help of a byte code verifier. Byte code verifier also offers additional security, as its role is to check the code for any defilement of admittance.

12. Write Once Run Anywhere: As deliberated above, java application produces a '.class' file which resembles to our applications (program) but comprises code in binary format. It offers ease, architecture-neutral effortlessness as byte code is not reliant on any machine architecture. It is the key reason why java is used in the enterprising IT industry altogether worldwide.

13. Power of compilation and interpretation: Most languages are intended with determination, either they are compiled language or they are interpreted language. But java join in arising massive power as Java compiler compiles the source code to byte code and JVM implements this byte code to machine OS-dependent workable code.

5.3 XML

XML (Extensible Markup Language) is a markup language comparable to HTML, but short of predefined tags to use. As an alternative, you define your own tags deviseed precisely for your needs. This is an authoritative way to store data in a presentation that can be stored, examined, and shared. Furthermost prominently, since the fundamental format of XML is homogeneous, if you share or transmit XML transversely systems or platforms, either nearby or over the internet, the addressee can still parse the data due to the identical XML syntax.

There are numerous languages built on XML, together with XHTML, MathML, SVG, RSS, and RDF. You can also describe your own structure of an XML document. The entire structure of XML and XML-based languages is built on tags.XML - statement is not a tag. It is used for the broadcast of the meta-data of a document.

```
<?xml version="1.0" encoding="UTF-8"?>
```

XML's suppleness has countless benefits. It lets you hand over data between corporate databases and Websites, deprived of losing vital descriptive information. It lets you automatically modify the presentation of data, somewhat than display the identical page to all comers. And it makes explores more efficient because search engines can classify through detailed tags rather than long pages of text.

5.4 Applications of XML

XML brings sophisticated data coding to websites, it helps companies integrate their information flows. By creating one set of XML tags for all corporate data, information is shared seamlessly among websites, databases, and other back-end systems. But the radical power of XML deceits in backup transactions among businesses. When a company sells an honest or service to a distinct company, an honest deal of knowledge must be exchanged—about prices, terms, stipulation, delivery schedules, and so on. HTML's one-size-fits-all nature makes such exchanges difficult, if not impossible, over the net. With XML, all the specified information could also be shared electronically, allowing complex deals to be closed with none of human intervention. That's why business-to-business Web markets, like those pass Ariba and Commerce One, already depend upon XML to automatically match buyers and sellers. Within the not-too-distant future, your company is additionally judged by the content of its XML tags.

Platform Independent and Language Independent: The foremost advantage of XML is that you can practice it to take data from a program like Microsoft SQL, adapt it into XML, then share that XML with further programs and platforms. You can interconnect between dual platforms, which are mostly very difficult.

The foremost thing which makes XML actually powerful is its international acceptance. Many establishments use XML boundaries for databases, programming, office application mobile phones and additional. This is due to its platform independent feature of system.

Basics of User Interface(UI)

Basically in Android XML is employed to implement the UI-related data. So understanding the core a part of the UI interface with relevance XML is very important. The computer program for an Android App is made because of the hierarchy of main layouts, widgets. The layouts are View Group objects or containers that control how the kid view should be positioned on the screen. Widgets here are view objects, like Buttons and text boxes.

Different Types of XML Files That Used in Android Studio

1. Layout XML files in android

The Layout XML files are accountable for the particular computer program of the applying. It embraces altogether the widgets or views like Buttons, Text Views, Edit Texts, etc. which are defined under the View Groups.

2. AndroidManifest.xml file

This file describes the essential information about the application's, just like the application's package names which matches code's namespaces, a component of the applying like activities, services, broadcast receivers, and content deliverrs. Permission required by the user for the appliance features also mentioned during this XML file.

3. strings.xml file

This file comprehends texts for all the Text Views thingamabobs. This permits reusability of cipher, and also benefits in the localization of the application with diverse languages. The strings definite in these documentations can be used to swap all hardcoded text in the whole application.

4. themes.xml file

This file describes the base theme and modified themes of the submission. It also used to describe elegance and aspects for the UI (User Interface) of the application. By essential elegance, we can adapt how the views or thingamabobs expression on the Operator Interface.

5. Drawable XML files

These are the XML files that deliver visuals to elements like custom background for the buttons and its undulation effects, also numerous ramps can be fashioned. This also holds the trajectory graphics like representations. Using these files convention layouts can be built for Edit Texts.

6. colors.xml file

The colors.xml file is accountable to hold all the sorts of colors mandatory for the application. It may be main product color and its alternatives and inferior brand color and its alternatives. The colors help support the product of the applications. So the colors need to be definite watchfully as they are in authority for the User Understanding. The colors need to be demarcated in hex code format.

7. dimens.xml file

As the file name itself recommends that the file is responsible to embrace the entire magnitudes for the views. It may be the elevation of the Button, packing of the views, the margin for the views, etc. The extents need to in the setup of pixel compactness(DP) values. Which changes all the hard-coded dp ideals for the views. This file desires to be twisted separately in the ideal's folder.

5.5 Gradle

Gradle is a build mechanization tool identified for its elasticity to build software program. A build mechanization tool is used to automate the construction of applications. The building process embraces compiling, involving, and packaging the code. The process becomes more dependable with the assistance of build automation tools.

It is widespread for its capability to build mechanization in languages like Java, Scala, Android, C/C++, and Groovy. This tool provisions groovy based Domain Detailed Language over XML. Gradle delivers construction, testing, and installing software on several platforms. The tool is widespread for edifice any software and large developments. Gradle consist of the pros of Ant and Maven and curbs the cons of both.

Around principal motivations to use Gradle are:

- The tool have attention on maintainability, usability, extensibility, performance, and athleticism.
- It is well-known to be highly customizable when it ascends to different outlines, dealing with various know ledges. We may use Gradle in several customs, like Java projects, Android projects, and Groovy projects.

- Gradle is widespread to deliver high-speed performance, approximately twice as fast as Maven.
- The tools sustenance a wide assortment of IDE's, which offer a healthier operator experience, as diverse people prefer working on a diverse IDE. It offers the users that like to toil on the terminal with the command-line edge, which offers topographies like Gradle tasks, command line completion, etc.
- Gradle obstacles all the matters confronted on other build tools like Maven and ANT.

Gradle shapes are castoff to define a project and its responsibilities. At least one Gradle build file is positioned in the origin ring binder of the project. A task signifies the work that a Gradle build has to accomplish, e.g., compiling the source cipher of the program. You can implement multiple tasks at a time under a single build file. These tasks can be enthusiastically created and protracted at runtime.

Features of Gradle

High Performance

Gradle rapidly appearances the task by considering the productivity from the previous implementations. The jobs whose contributions are changed are the only ones that are performed. This helps in evading pointless tasks and produces developed performance.

delivers Support

Known to deliver provision, we use Gradle for ANT build developments. Tasks can be imported from ANT build projects then can be reclaimed in Gradle. Gradle also provisions Maven fountains that are made to distribute and raise dependencies of the project.

Multi-Project Build Software

Gradle guarantees wonderful sustenance for multi-project builds. These projects can cover a root development and any sum of subprojects. Gradle provisions limited builds, which states that the tool will control if a project on which our project be contingent, needs any kind of rebuilding. In case the project needs transformation, Gradle will do that earlier building any additional projects.

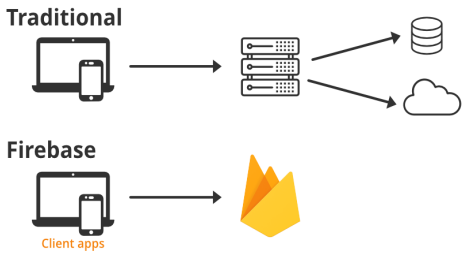
5.6 Android Studios

Android Studio is the authorized integrated development environment (IDE) for Google's Android operating system, built on JetBrains' IntelliJ IDEA software besides deviseed exactly for Android expansion. It is accessible for transfer on Windows, macOS and Linux based operating systems or as a subscription-based package in 2020. It is an auxiliary for the Eclipse Android Development Tools (E-ADT) as the main IDE for native Android application growth.

Android Studio was broadcasted on May 16, 2013, at the Google I/O consultation. It was in initial contact broadcast stage preliminary from version 0.1 in May 2013, at one time arrived beta stage preliminary from version 0.8 which was unconfined in June 2014. The first stable figure was unconfined in December 2014, initial from version 1.0.

5.7 Firebase

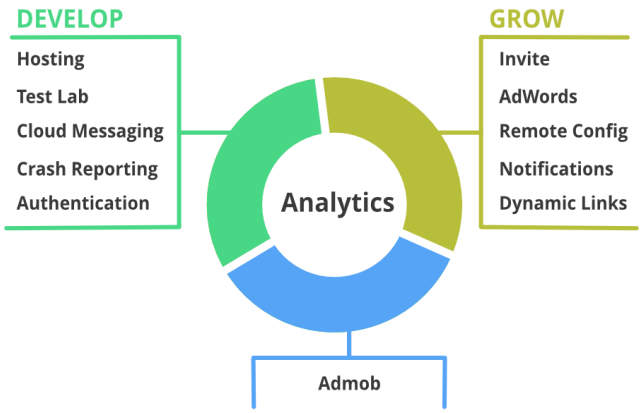
Firebase is a creation of Google which assistance developers to shape, manage, and grow their apps effortlessly. It helps deviseers to build their applications quicker and in a more safe way. No software devise is required on the Firebase sideways, which varies it easy to use its topographies more proficiently. It offers services to android, IOS, web, and unity. It offers a cloud storage. It customs NoSQL for the database for the storing of data.



Firebase originally was an online chat service benefactor to numerous websites over API and ran with the name Evolve. It became popular as developers secondhand it to exchange application information like a game state-run in actual time across their operators more than the chats. This occasioned in the departure of the Evolve architecture, and it's chat system. The Evolve architecture was further changed by its founders James Tamplin and Andrew Lee, to come again up-to-date day Firebase is in the year 2012.

Features of Firebase

Mainly, there are 3 categories in which firebase delivers its services.



Build better applications

This feature mostly contains backend facilities that help deviseers to build and accomplish their applications in a better means. Services encompassed under this feature are :

Real-time Database: The Firebase Real-time Database is a cloud-based NoSQL catalog that manages your statistics at the heated rapidity of milliseconds. In modest term, it can be measure out as a big JSON file.

Cloud Fire store: The cloud fire store is a NoSQL file record that offers services like store, sync, and request over the application on a large scale. It provisions data in the method of objects also identified as documents. It has a key-value couple and can stock all types of data like, strings, binary data, and even JSON trees.

Authentication: Firebase Authentication provision delivers relaxed to use UI libraries and SDKs to validate users to your app. It decreases the manpower and exertion obligatory to grow and uphold the user confirmation service. It even grips tasks like integrating accounts, which if completed physically can be hectic.

Remote Config: The remote configuration package benefits in publishing apprises to the user directly. The changes can vary from shifting machines of the UI to altering the performance of the applications. These are frequently used while issuing seasonal proposals and insides to the application that has an uncomplete life.

Hosting: Firebase offers hosting of applications with rapidity and safety. It can be used to host Stationary or Eventful web site and microservices. It has the aptitude of hosting an application with a solo command.

Firebase Cloud Messaging(FCM): The FCM service offers an assembly among the server and the application end operators, which can be used to obtain and direct messages and notifications. These networks are dependable and battery-efficient.

CHAPTER-6
SAMPLE CODE

6. SAMPLE CODE

1. Splash screen (Main.java)

```
Public class Main Activity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        getSupportActionBar().hide();
        getWindow().addFlags(WindowManager.LayoutParams.FLAG_FULLSCREEN);

        new Handler().postDelayed(new Runnable() {


            @Override
            public void run() {
                Intent i = new Intent(MainActivity.this, LoginActivity.class);
                startActivity(i);
                finish();
            }
        }, 3000);


    }
}
```

2. Home Activity (Home.java)

```
public class HomeActivity extends AppCompatActivity {
    FirebaseAuth auth;
    RecyclerView rv;
    FirebaseUser user;
    DatabaseReference reference;
    FirebaseDatabase database;
    String userName;

    List<String> list;
    UsersAdapter adapter;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_home);



        rv = findViewById(R.id.rv);
        rv.setLayoutManager(new LinearLayoutManager(this));

        list = new ArrayList<>();
        auth = FirebaseAuth.getInstance();
        database = FirebaseDatabase.getInstance();
        reference = database.getReference();
        user = auth.getCurrentUser();
    reference.child("Users").child(user.getUid()).child("username").addValueEventListener(new
ValueEventListener() {
        @Override
        public void onDataChange(@NonNull DataSnapshot snapshot) {
            userName = snapshot.getValue().toString();
            getUsers();
            adapter = new UsersAdapter(list,userName,HomeActivity.this);
            rv.setAdapter(adapter);
        }
 @Override
 public void onCancelled(@NonNull DatabaseError error) {
 }
 });
}
```

```
public void getUsers()
{
   reference.chilr("User").addChild Event Listener(new Child EventListener() {
       @Override
       public void onChildAdd(@NonNull Data Snapshot snapshot, @Nullable String previous
ChildName) {
          String key = snapshot.get Key();
          if(!key.equals(user.get Uid()))
          {
             list.add(key);
             adapter.notifyData SetChange();
          }
       }
       @Override
       public void onChild Chang(@NonNull Data Snapshot snapshot, @Nullable String
previousChild Name) {
       }
       @Override
       public void on Child Remove(@NonNull Data Snapshot snapshot) {
       }
       @Override
       public void onChildMo(@NonNull Data Snapshot snapshot, @Nullable String previousChild
Name) {
       }
       @Override
       public void on Cancel(@NonNull Database Error error) {
       }
   });
}
@Override
public boolean on Create Options Menu(Menu menu) {
   menuInflater.inflate (R_menu.chat_menu,menu);
   return super.onCreate_OptionsMenu(menu);
}
@Override
public boolean onOptions_Item_Selected(@NonNull MenuItem item)
{
 if(item.getItemId() == R.id.action_profile)
{
 startActivity(new Intent(Home_Activity.this,Profile_Activity.class));
 }
```

```
        if(item.getItemId() == R.id.todo)
        {
            startActivity(new Intent(HomeActivity.this,TodoActivity.class));
        }
        if(item.getItemId() == R.id.games)
        {
            startActivity(new Intent(HomeActivity.this,GameActivity.class));
        }
        return super.onOptionsItemSelected(item);
    }
}
```

3. Login Activity (LoginActivity.java)

```
public class LoginActivity extends AppCompatActivity {
    private EditText editTextEmail,editTextPassword;
    private TextView buttonSignup,textViewForgot;
    private Chip buttonSingin;
    FirebaseAuth auth;
    FirebaseUser firebaseUser;
    @Override
    protected void onStart() {
        super.onStart();
        firebaseUser =FirebaseAuth.getInstance().getCurrentUser();
        if(firebaseUser != null)
        {
            Intent intent = new Intent(LoginActivity.this,HomeActivity.class);
            startActivity(intent);
        }
    }
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_login);
        getSupportActionBar().hide();
        editTextEmail = findViewById(R.id.editTextEmail);
        editTextPassword = findViewById(R.id.editTextPassword);
        buttonSignup = findViewById(R.id.buttonSignup);
        textViewForgot = findViewById(R.id.textViewForgot);
        buttonSingin = findViewById(R.id.buttonForgot);
```

```
 buttonSignup.setOnClickListener(new View.OnClickListener() {
          @Override
          public void onClick(View view) {
             startActivity(new Intent(LoginActivity.this,SignupActivity.class));
          }
       });
       textViewForgot.setOnClickListener(new View.OnClickListener() {
          @Override
          public void onClick(View view) {
             startActivity(new Intent(LoginActivity.this,ForgotActivity.class));
          }});
      buttonSingin.setOnClickListener(new View.OnClickListener(){
          @Override
          public void onClick(View view) {
             String emai = editTextEmail.getText().toString();
             String pssd = editTextPassword.getText().toString();
             if(!emai.equals("") && !pssd.equals("")){
                signin(emai,pssd);
             }
             else
             {
                Toast.makeText(LoginActivity.this,"Please enter valid email and
password",Toast.LENGTH_SHORT).show();
        }});} public void signin(String email,String password)
       auth.signInWithEmailAndPassword(email, password).addOnCompleteListener(new
OnCompleteListener<AuthResult>() {
 @Override
 public void onComplete(@NonNull Task<AuthResult> task) {
 if(task.isSuccessful()){
 startActivity(new Intent(LoginActivity.this,HomeActivity.class));
 Toast.makeText(LoginActivity.this,"logged in sucessfully",Toast.LENGTH_SHORT).show();
 }
 else{
 Toast.makeText(LoginActivity.this,"Login failed",Toast.LENGTH_SHORT).show();
 }
 }});}}
```

```
public class ForgotActivity extends AppCompatActivity {
    private EditText editTextForgot;
    private Chip buttonForgot;
    FirebaseAuth auth;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_forgot);
        getSupportActionBar().hide();
        editTextForgot = findViewById(R.id.editTextForgot);
        buttonForgot = findViewById(R.id.buttonForgot);
        auth = FirebaseAuth.getInstance();
        buttonForgot.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                String email = editTextForgot.getText().toString();
                if(!email.equals(""))
                {
                    passwordReset(email);
                }
            }});
    }
    public void passwordReset(String email){
auth.send.PasswordResetEml(emailadress).add OnCompleteListener(new
OnCompleteListener<Void>() {
            @Override
            public void onCompleted(@NonNull Task1<Void> task) {
                if(task.isSuccessful())
                { Toast.makeText(ForgotActivity.this,"successfully sent an
email",Toast.LENGTH_SHORT).show();
                }
                else
                { Toast.makeText(ForgotActivity.this,"Check",Toast.LENGTH_SHORT).show();
                }
            }
        });
    }
}
```

```
public class MyChatActivity extends AppCompatActivity {
    private ImageView imageViewBack;
    private TextView textViewChat,pay;
    private RecyclerView rvChat;
    private EditText editTextMessage;
    private FloatingActionButton fab;

    String RecieverName;
    String userName;
    FirebaseDatabase database;
    FirebaseAuth firebaseAuth;

    DatabaseReference reference;
    MessageAdapter adapter;
    List<ModelClass> list;
    @Override
    protected void onCreate(Bundle saved_Instance_State) {
        super.onCreate(saved_Instance_State);
        setContentView(R.layout.activity_my_chat);
        getSupportActionBar().hide();

        database = FirebaseDatabase.getInstance();
        firebaseAuth = FirebaseAuth.getInstance();
        reference = database.getReference();

        userName = getIntent().getStringExtra("username");
        RecieverName = getIntent().getStringExtra("otherName");
        imageViewBack = findViewById(R.id.imageViewBack);
        textViewChat = findViewById(R.id.textViewChat);
        pay = findViewById(R.id.pay);
        rvChat = findViewById(R.id.rvChat);
        editTextMessage = findViewById(R.id.editTextTextMessage);
        fab = findViewById(R.id.fab);
        rvChat.setLayoutManager(new LinearLayoutManager(this));
        list = new ArrayList<>();
        textViewChat.setText(""+RecieverName);
        pay.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
        startActivity(newIntent
(MyChatActivity.this,PayActivity.class));
    }
});
```

```
 imageViewBack.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            startActivity(new Intent(MyChatActivity.this,HomeActivity.class));
        }
    });
    fab.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            String message = editTextMessage.getText().toString();
            if(!message.equals(""))
            {
                sendMessage(message);
                editTextMessage.setText("");
            }
        }
    });
    getMessage();
}
public void sendMessage(String message)
{
    String key =
reference.child("Messages").child(userName).child(RecieverName).push().getKey();
    Map<String,Object> messageMap =new HashMap<>();
    messageMap.put("message",message);
    messageMap.put("from",userName);
 reference.child("Messages").child(userName).child(RecieverName).child(key).setValue(message
Map).addOnCompleteListener(new OnCompleteListener<Void>() {
        @Override
        public void onComplete(@NonNull Task<Void> task) {
            if(task.isSuccessful())
            {
reference.child("Messages").child(RecieverName).child(userName).child(key).setValue(message
Map);
            }

        }
    });
}
```

```
    public void getMessage()
    {
reference.child("Messages").child(userName).child(Reciever_Name).addChild_EventListener(ne
w ChildEventListener() {
            @Override
            public void onChild_Add(@NuLLNull Data_Snapshot snapshot, @Nullable String
previousChildName) {
                ModelClass modelClass = snapshot.getValue(Model_Class.class);
                list.add(model_Class);
                adapter.notifyData_SetChanged();
                rvChat.scrollToPosition(list.size()-1);
            }
            @Override
            public void onChildChanged(@NonNull DataSnapshot snapshot, @Nullable String
previousChildName) {
            }
            @Override
            public void onChildRemoved(@NonNull DataSnapshot snapshot) {
            }
            @Override
            public void onChildMoved(@NonNull DataSnapshot snapshot, @Nullable String
previousChildName) {
            }
            @Override
            public void onCancelled(@NonNull DatabaseError error) {
           }
        });
        adapter = new MessageAdapter(list,userName);
        rvChat.setAdapter(adapter);
    }


}
```

4. Pay Activity (PaymentActivity.java)

```
public class PayActivity extends App_Compat_Activity {
    private Button pays;
    private TextView G_pay;
    @Override
    protected void onCreate(Bundle saved_InstanceState) {
        pays = findViewById(R.id.pays);
        Gpay = findViewById(R.id.G_pay);

        Gpay.setOnClickListener(new View.OnClick_Listener() {
            @Override
            public void onClick(View view) {
                Intent launch_Intent =
getPackage_Manager().getLaunch_IntentFor_Package("com.google.android.gms.pay");
                if(launch_Intent != null)
                {
                    start_Activity(launch_Intent);
                }
                else{
                    Toast.makeText(Pay_Activity.this,"App not found",Toast.LENGTH_LONG).show();
                }}});
        pays.setOn_Click_Listener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                AlertDialog.Builder alert_Dialog = new AlertDialog.Builder(Pay_Activity.this);
                alertDialog.setTitle("Status")
                .setMessage("Payment Successful").set_PositiveButton("Okay", new
DialogInterface.OnClickListener() {
                        @Override
                        public void onClick(Dialog_Interface dialogInterface, int i) {

                            dialogInterface.cancel();
                        }
                    }).show();
                alertDialog.create();
            }
        });
    }
```

5. Todo Activity (TodoActivity.java)

```
public class TodoActivity extends AppCompatActivity {
    private TextView user_id;
    private EditText item_1;
    private Button add_btn;
    private ListView list;
    String Name;
    ArrayList<String> itemList = new ArrayList<>();
    ArrayAdapter<String> arrayAdapter;
    @Override
    protected void on_Create(Bundle saved_InstanceState) {
        super.on_Create(saved_Instance_State);
        setContent View(R.layout._activity_to-do);
        getSupportActionBar().hide();
        item = findViewById(R.id.item);
        addbtn = findViewById(R.id.add_btn);
        list = findViewById(R.id.list);
android.R.layout.simple_list_item_1,android.R.id.text1,itemList);
        list.setAdapter(arrayAdapter);
        addbtn.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                String itemName = item.getText().toString();
                itemList.add(itemName);
                item.setText("");
FileHelper.writeData(itemList,getApplication_Context());
                arrayAdapter.notifyData_Set_Changed();
            }
        });
        list.setOnItem_ClickListener(new AdapterView.OnItem_Click_Listener() {
            @Override
            public void onItemClick(AdapterView<?> adapter_View, View view, int position, long l)
{
                AlertDialog.Builder alert = new AlertDialog.Builder(To-do_Activity.this);
                alert.setTitle("Delete");
                alert.set_Message("Sure Want to delete this item?");
                alert.setCancelable(false);
```

```
    @Override
            alert.setNegativeButton("No", new DialogInterface.OnClickListener() {
                @Override
                public void onClick(DialogInterface dialogInterface, int i) {
                    dialogInterface.cancel();
                }
            });
            alert.setPositiveButton("Yes", new DialogInterface.OnClickListener() {
                @Override
                public void onClick(DialogInterface dialogInterface, int i) {
                    itemList.remove(position);
                    arrayAdapter.notifyDataSetChanged();
                    FileHelper.writeData(itemList,getApplicationContext());
                }
            });
            AlertDialog alertDialog = alert.create();
            alertDialog.show();
        }
    });


    }
}
```

CHAPTER-7
TESTING

7. TESTING

7.1 Introduction

In overall, software engineers differentiate software program liabilities from software letdowns. In case of a letdown, the software does not do what the user imagines. A fault is a program writing error that may or may not really apparent as a failure. A fault can also be labelled as an error in the precision of the semantic of a computer program. A fault will become a catastrophe if the exact totaling conditions are met, one of them being that the defective portion of computer software implements on the CPU. A fault can also turn into a failure when the software is ported to a different hardware platform or a different compiler, or when the software gets protracted. Software testing is the technical investigation of the product under test to deliver investors with quality related information.

System Testing and Implementation

The determination is to workout the different portions of the module code to notice coding errors. After this the modules are slowly combined into subsystems, which are then combined themselves too ultimately forming the complete system. During this combination of module integration testing is accomplished. The goal of this is to detect deceitful errors, while concentrating the interconnection among modules. After the system was put composed, system challenging is performed. Here the system is tested against the system requirements to see if all necessities were met, and the system performs as detailed by the necessities. Finally, accepting testing is achieved to establish to the client for the operation of the system.

For the challenging to be positive, proper assortment of the test case is essential. There are two diverse approaches for choosing a test case. The software or the module to be tested is preserved as a black box, and the test cases are obvious based on the conditions of the system or module. For this motive, this form of testing is also called “black box testing”.

The emphasis here is on testing the external conduct of the system. In structural testing, the test cases are definite based on the logic of the component to be tested. A common method here is to accomplish some type of coverage of the declarations in the code. The two procedures of testing are complementary: one tests the external behavior, the further tests the internal structure.

Testing is a tremendously critical and time-consuming action. It requires appropriate planning of the complete testing process. The testing process often twitches with the test plan. This plan classifies all testing associated activities that must be achieved and specifies the schedule, allocates the properties, and specifies guidelines for testing.

This strategy recognizes all testing connected activities that must be achieved and specifies the schedule, allocates the resources, and postulates strategies for testing. The test plan stipulates conditions that must be tested; diverse units to be tested, and the manner in which the module will be joined together. Then for a diverse test unit, a test case specification document is formed, which lists all the diverse test cases, together with the predictable outputs, that will be castoff for testing. Throughout the testing of the component, the stated test cases are performed, and the definite results are associated with the predictable outputs. The final output of the testing stage is the testing description and the error description, or a set of such intelligences. Each test report contains a set of test cases and the result of implementing the code with the test cases. The error report defines the errors come across and the action taken to remove the error.

Testing Techniques

Testing is a process, which discloses errors in the program. It is the main quality portion employed throughout software development. During testing, the program is implemented with a set of circumstances known as test cases and the output is appraised to determine whether the driver is performing as predictable. To make sure that the system does not have faults, the diverse levels of testing approaches that are functional at differing phases of software development are:

Black Box Testing

In this approach, some test cases are created as input circumstances that fully execute all functional necessities for the program. This challenging has been uses to find faults in the following categories:

- Improper or missing functions
- Interface faults
- Errors in data structure or external catalog access
- Presentation errors
- Loading and termination errors.

In this difficult only the production is patterned for perfection. The reasonable flow of the information is not checked.

White Box Testing

In this testing, the test cases are engendered on the judgement of each component by drawing flow diagrams of that component and logical conclusions are confirmed on all the cases. It has been uses to produce the test cases in the following cases:

- Assurance that all autonomous paths have been implemented.
- Execute all reasonable conclusions on their correct and untrue sides.
- Perform all loops at their limitations and within their operational.
- Perform internal statistics structures to guarantee their validity.

Testing Strategies

Unit testing

Unit testing contains the strategy of test cases that authenticate that the inner package logic is functioning properly, and that program participation produce legal outputs. All choice branches and interior code flow should be authenticated. It is the challenging of separate software units of the application . It is done after the accomplishment of a distinct unit before incorporation. This is a physical difficult, that depend on information of its building and is offensive. Unit tests achieve basic tests at constituent level and test a detailed occupational process, request, and/or system outline formation. Unit tests guarantee that each single path of a business procedure performs precisely to the documented stipulations and contains clearly distinct inputs and predictable results.

This Organization contains of 3 modules. Those are standing module, route discovery module, review module. Each module is taken as a component and verified. Identified errors are modified, and executable unit are attained.

Integration testing

Integration tests are intended to test combined software components to control if they run as one package. Testing is happening driven and is more worried with the basic consequence of screens or grounds. Integration tests establish that even though the components were separately satisfaction, as shown by positively unit testing, the grouping of components is correct and steady. Integration testing is precisely aimed at exposing the difficulties that arise from the mixture of components.

System Testing

System testing safeguards that the whole united software organization meets necessities. It tests a formation to confirm known and expected results. An instance of system testing is the configuration-oriented system combination test.

System testing is created on process metaphors and flows, highlighting pre-driven process links and incorporation points.

Functional Testing

Functional tests offer orderly protests that purposes tested are available as quantified by the occupational and procedural requirements, system certification, and user manuals.

Functional testing is centered on the following items

- Valid Input : recognized classes of valid contribution must be acknowledged.
- Invalid Input : recognized classes of invalid input obligation to be rejected.
- Functions : recognized functions must be trained.

- Output : : recognized classes of application outputs must be trained..
- Systems/Procedures : Interfacing schemes or actions must be invoked.

Organization and groundwork of functional examinations is focused on necessities, important functions, or singular test cases. In addition, methodical coverage affecting to classify Business procedure flows, data arenas, predefined procedures, and successive processes.

7.2 Sample Test Case Specification

Test Number	Test Case	Output	Result
T1	Open project in Android studios	Project opens Successfully	Pass
T1	Open project in Android studios	Fails to Open Project	Fail
T2	Gradle Build Project	automate the creation of application	Pass
T2	Gradle Build Project	Fail to syncs the files	Fail
T3	Cloud Connection	Connected to Firebase	Pass
T4	Creating User	New User in Database	Pass
T4	Creating User	Database Error	Fail
T5	Message	Successfully receiving and sending messages	Pass
T5	Message	Adapter Fails to connect	Fail

Table 7.2 Test case Specification

Test Number	Test Case	Output	Result
T6	Profile Update	Name changed	Pass
T6	Profile Update	Name Unchanged	Fail
T7	Lost password mail service	Mail received	Pass
T7	Lost password mail service	Fails to send mail	Fail

Table 7.2 Test case Specification

7.3 Test Case Screens :

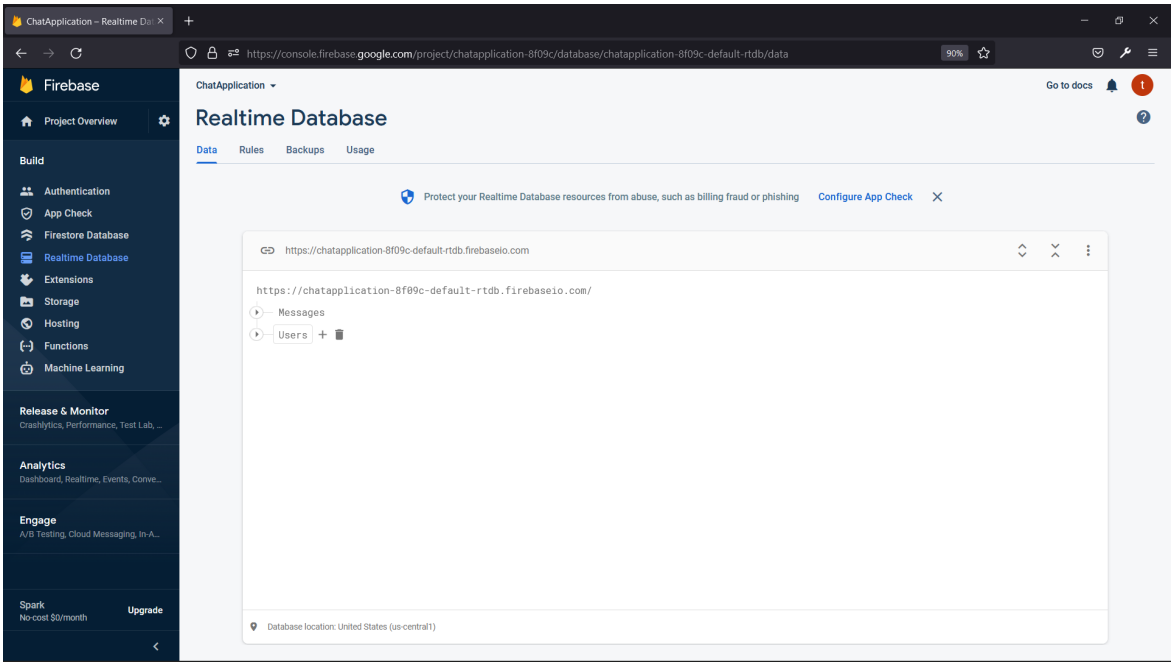


Fig 7.1 Screenshot for Successful Build of Database connection

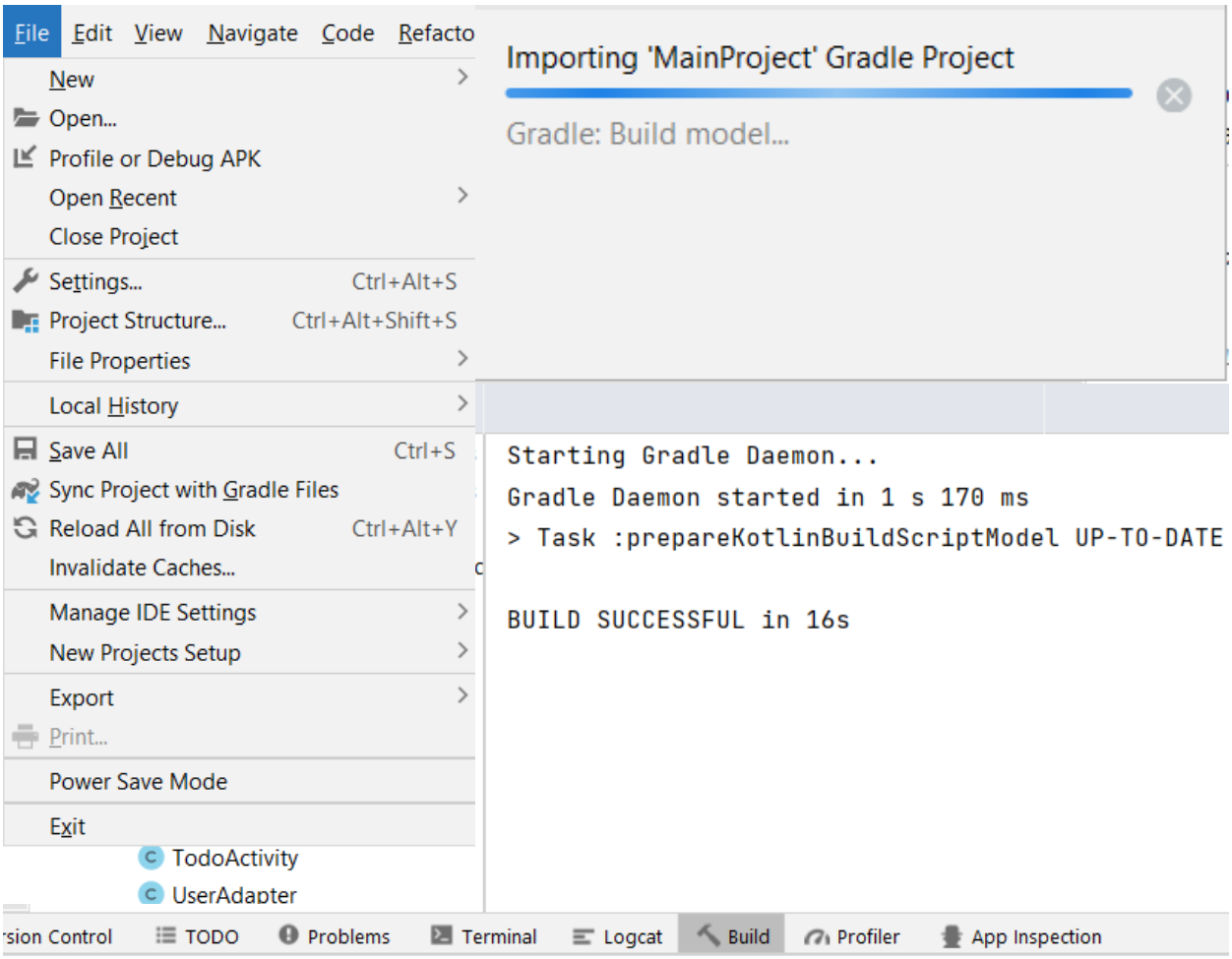


Fig 7.2 Screenshot for Successful Build of Gradle

Description

The Gradle build is a procedure of generating a Gradle development. When we track a Gradle expertise, it will expression for a file called build. Gradle in the existing directory. This case is also called the Gradle shape script. The build conformation, tasks, and plugins are labelled in this file. The build script describes the project and its tasks.

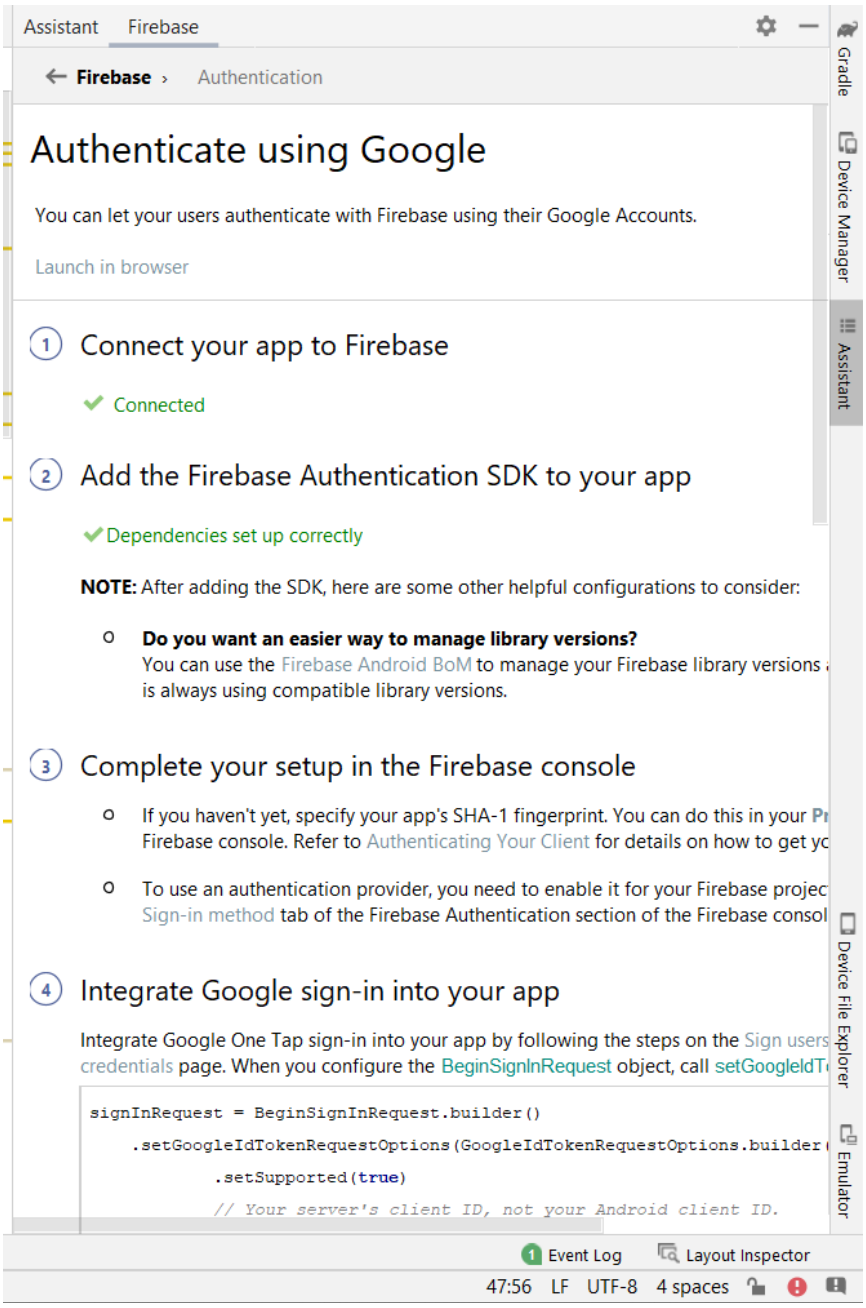


Fig 7.3 Screenshot for Connection to Firebase

Description

The Firebase Real-time Catalog is a cloud-hosted database. The information is stored as JSON and synchronized in real-time to each connected user. When you size a cross-platform app with our Apple platforms, Android, and JavaScript SDKs, all the patrons share one Real-time Database occurrence and mechanically receive apprises with the newest data.

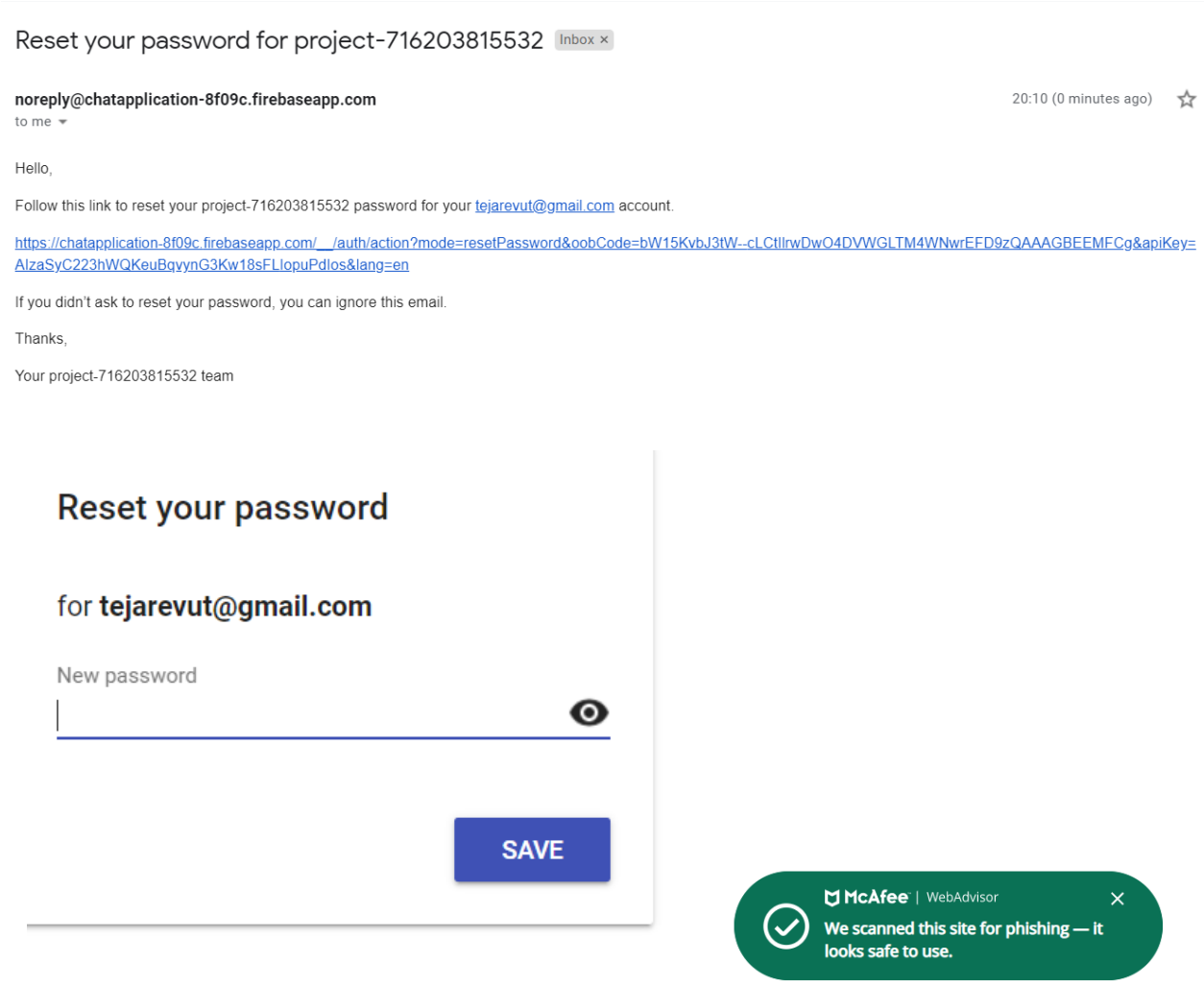


Fig 7.4 Screenshot of mail service for Forgot Activity

Description

Firebase is a mobile and web application expansion platform. It offers amenities that a web application or mobile application might want and need. Firebase offers email and password confirmation without any overhead of edifice backend for user authentication. Firebase would send user a reset password email with a link which guides operator to a reorganized password webpage. The defaulting email template is polyglot and customizable.

CHAPTER-8
SCREENSHOTS

8. SCREENSHOTS

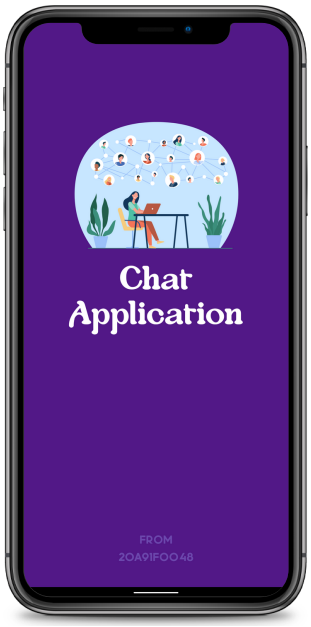


Fig 8.1 Screenshot of Splash screen

Description

A splash screen is a graphic regulator component consisting of a window holding an image, a logo, and the current version of the package or program. A splash screen can give the impression while a game or program is initiated. A splash page is an outline page on a website. Infrequently, a progress bar within the splash awning deviseates the loading development. A splash screen vanishes when the application's foremost window appears. Splash screens may be additional for a period of time and then substituted anew.

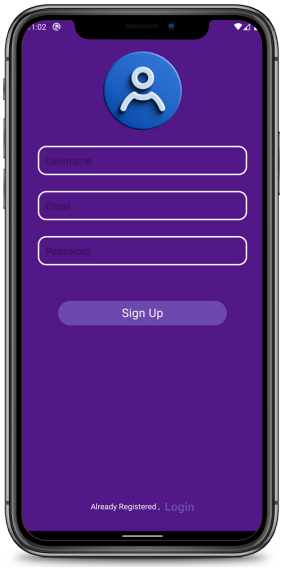


Fig 8.2 Screenshot of Sign Up

Fig 8.2.1 Screenshot of Created account

Description

Sign up is an action to catalogue yourself for a new interpretation. For example, you need to sign up beforehand you can contact your Gmail account which needs you to fill your particulars like name, address, email ID, contact number, and a password to log in. Just speaking, when you sign up for somewhat, you actually roll yourself as a new user. It is an accomplishment that better describes how you can cooperate with the websites. Sign up just means to generate an account.

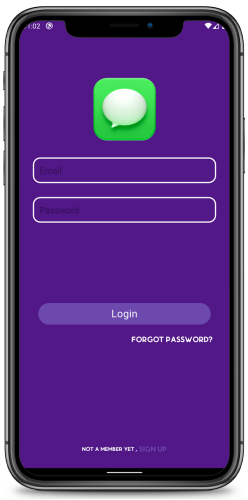


Fig 8.3 Screenshot of Login screen

Description

In computer security, logging in is the procedure by which a discrete gain's admittance to a computer system by recognizing and confirming themselves. The user identifications are typically some form of username and a corresponding password, and these authorizations themselves are sometimes mentioned to as a login.

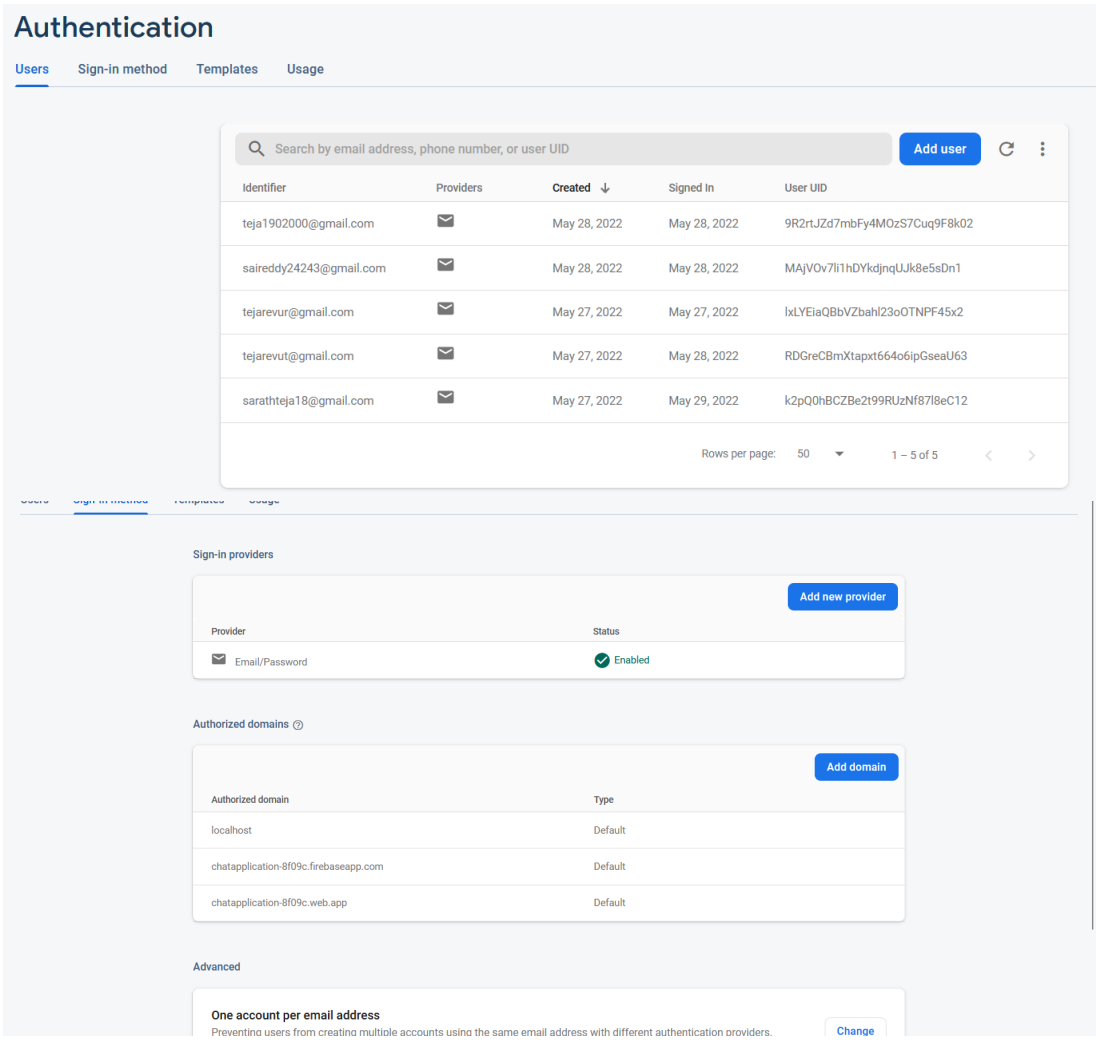


Fig 8.3.1 Screenshot of Authentication Identifiers

Description

Firebase Authentication purposes to make edifice secure verification systems easy, while refining the sign-in and onboarding knowledge for end users. It offers an end-to-end individuality solution, supporting email and password balance sheet, phone auth, and Google, Twitter, Facebook, and GitHub login, and more.

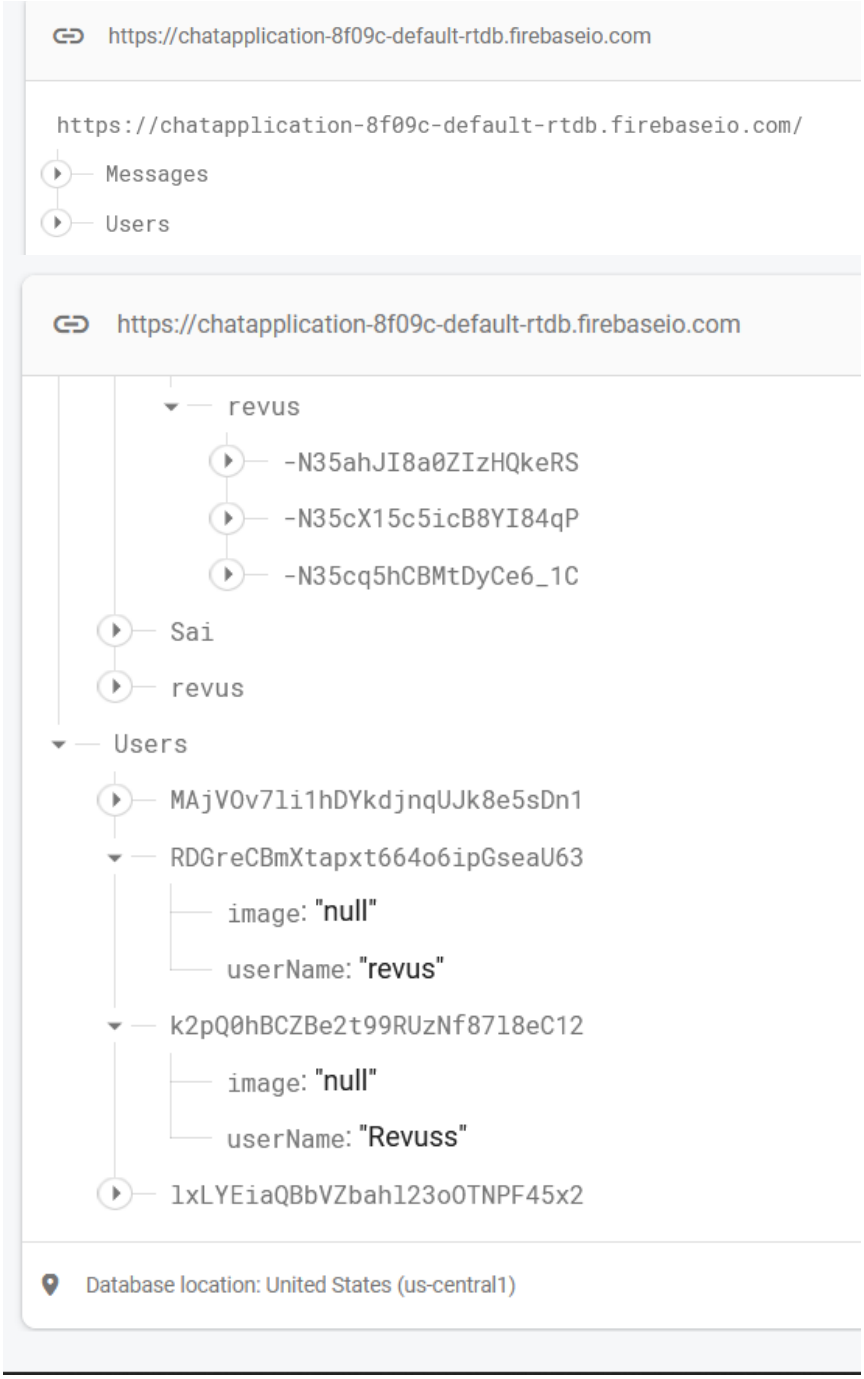


Fig 8.4 Screenshot of Realtime Database

Description

This application customs FCM record for packing of data. Firebase Authentication goals to make building safe verification systems relaxed, while refining the sign-in and onboarding knowledge for end operators. It offers an end-to-end identity explanation, subsidiary email and password books, phone auth, and Google, Twitter, Facebook, and GitHub login, and more.

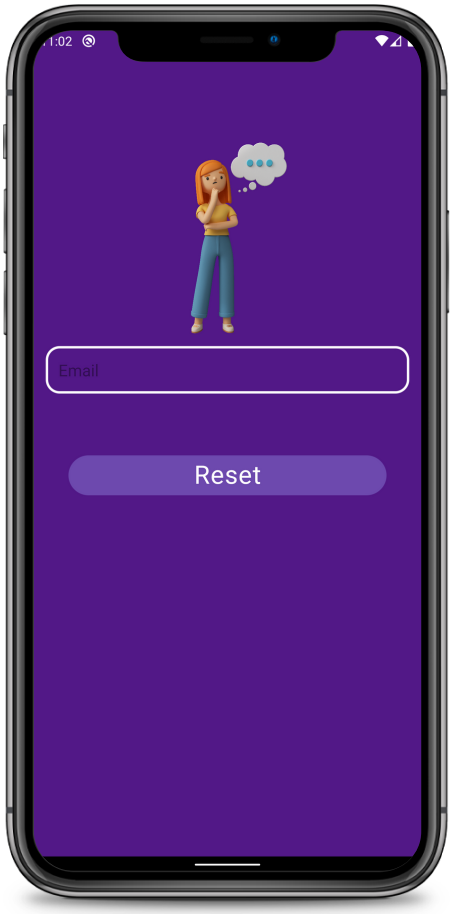


Fig 8.5 Screenshot of Forgot activity

Description

This activity lets operators who have elapsed their password to answer, retrieve, or reset it, usually by conveyance a secret link to their recorded e-mail. Once the account is tested, a new or temporary password is directed to the e-mail address related with it or even via text message.

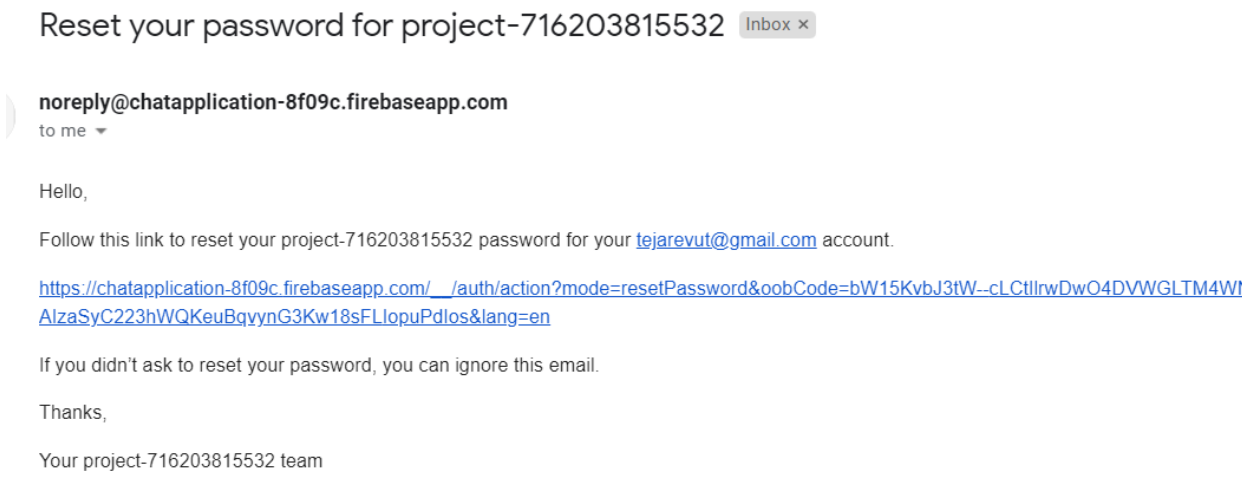


Fig 8.5.1 Screenshot of Forgot mail

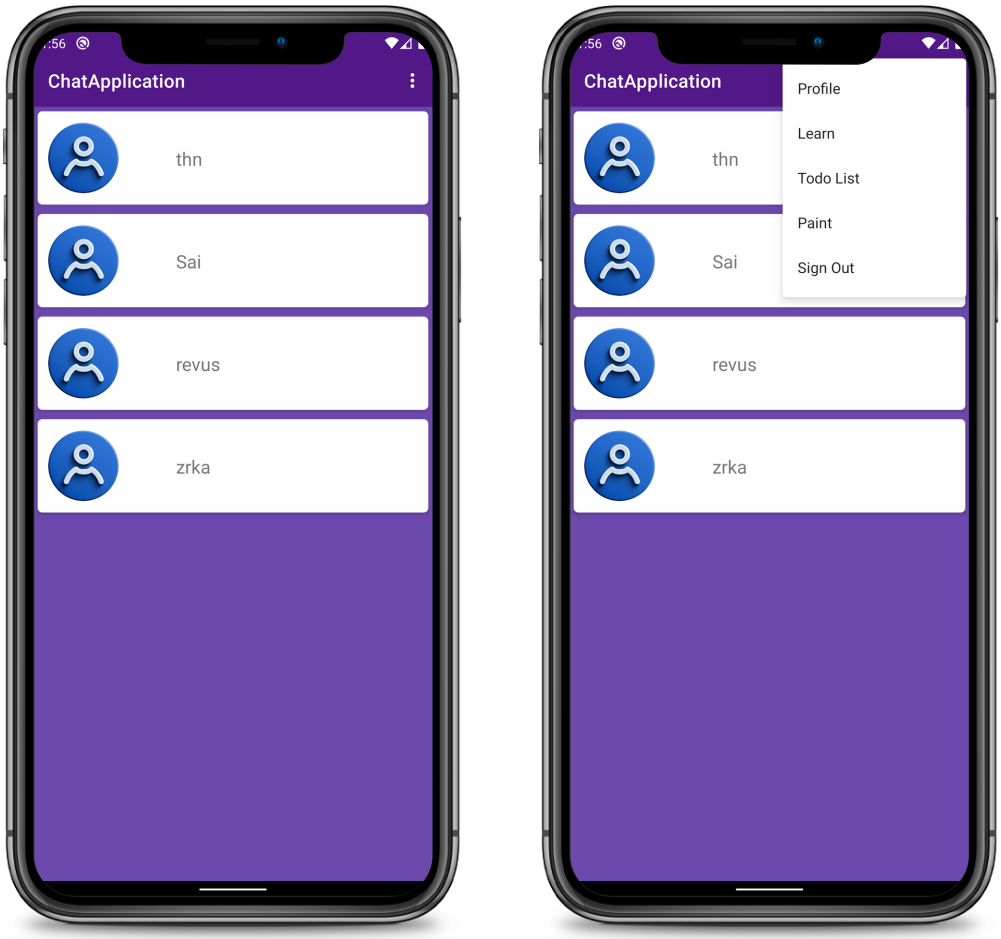
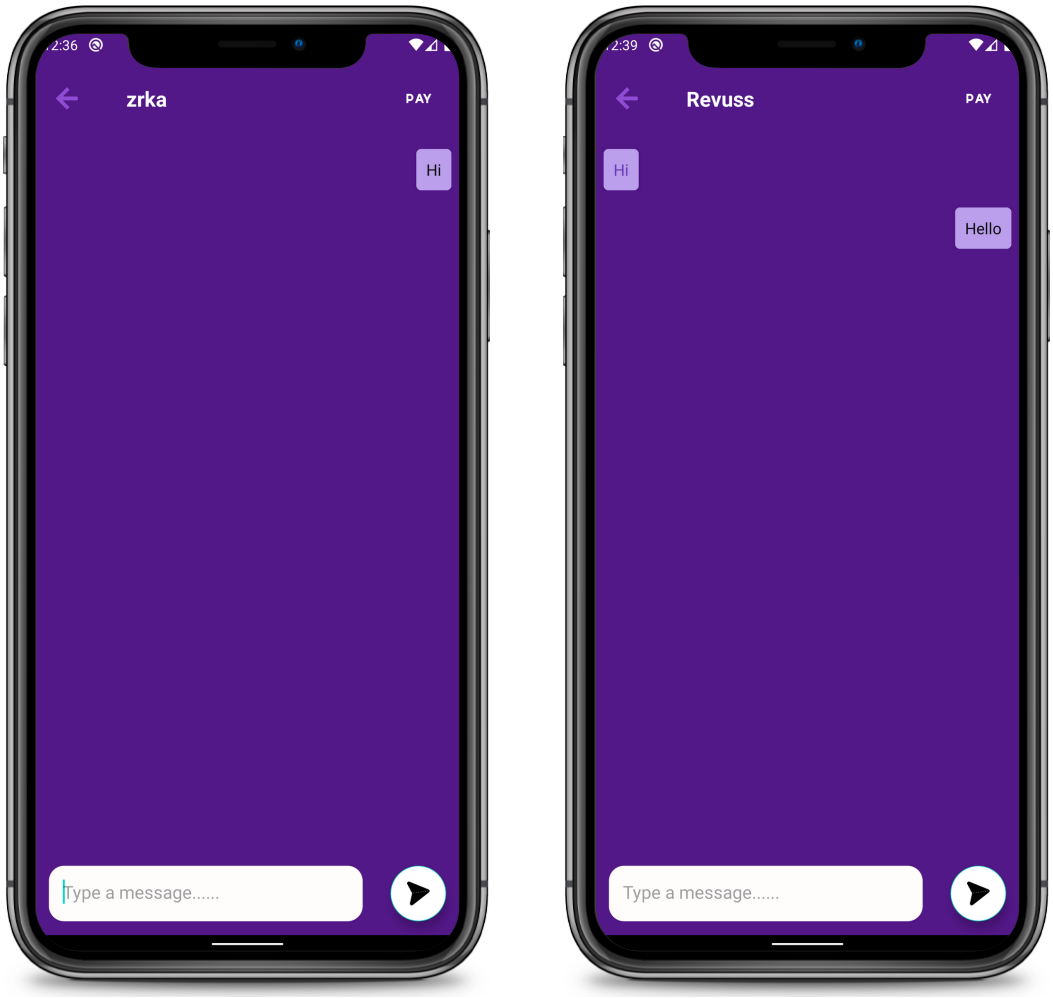


Fig 8.6 Screenshot of Home Screen and menu

Description

After successful authentication, the user will be redirected to the homepage of the application. This activity allows users to access to main screen of the application, containing of number of users in the database to chat with and contains features of the application as shown in the figure containing three dot icons. This icon contains of options like Profile, To-do List, Paint and Sign Out. This activity is the primary page of application which a visitor navigating from the login activity, and also serve as a landing page to attract the visitors. This activity can also be called as start page of the application as it is the first interface for user after authentication. Once the user click's on one of the user, the application redirects the user to the chat activity page for the selected user to communicate.



Sender **Reciever**

Fig 8.7 Screenshot of Chat Activity Screen

Description

A chat window is a window rummage-sale by a one-to-one program to allow a user to see messages he or she has established, as well as those mails that he or she has sent. This window is classically detached into deuce basic parts, the exhibition section that demonstrations the mails that have stood sent by respectively operator and the contribution piece that lets an operator understand the communication he or she is presently generating. A chat opening can be part of a chat package, such as immediate messaging programs, or may happen within a bigger framework, such as a connected computer game or social schooling website. In totalling to the exhibition section, this submission also contains an activity named Pay, which is used to make payments.

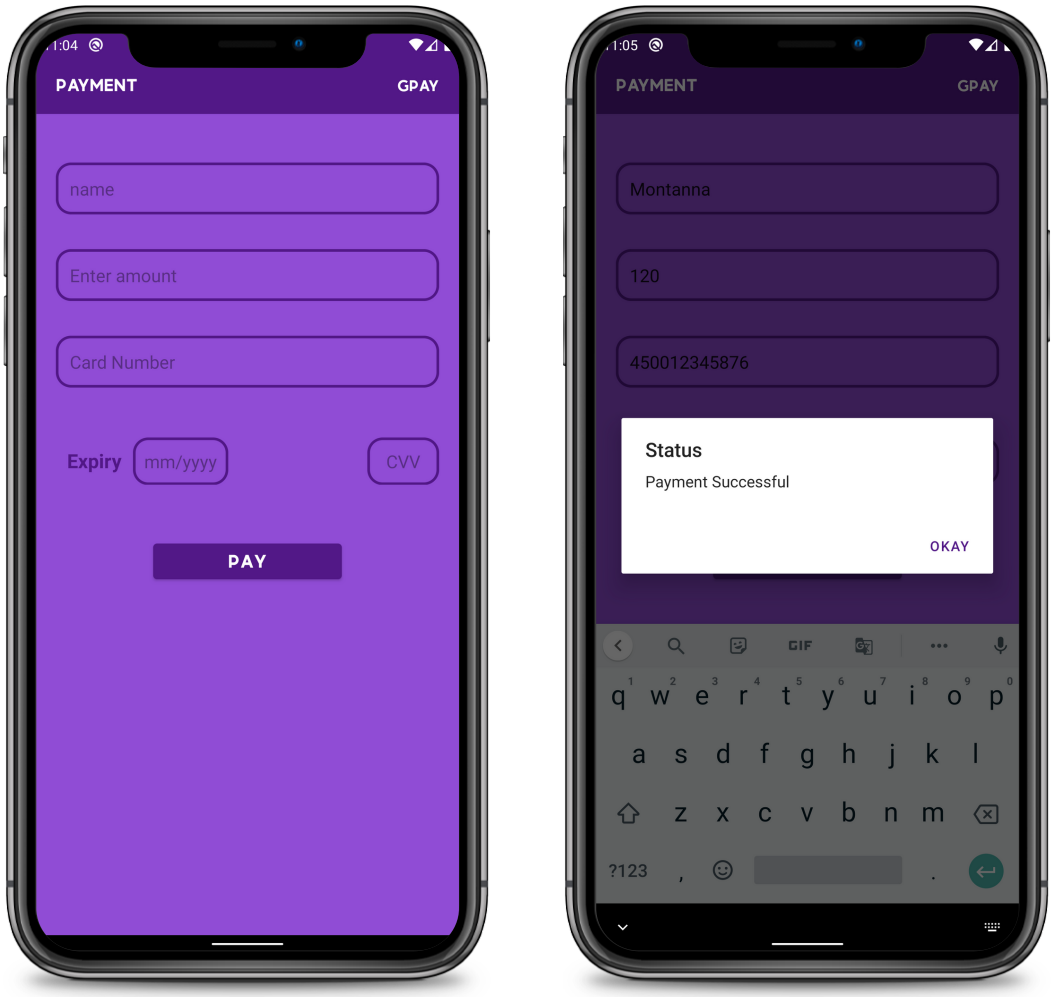


Fig 8.8 Screenshot of Payment Screen

Description

A payment entry is a merchant service providing by the application provision deliverr that authorizes credit card or direct payments processing. The payment entry may be deliverd by a bank to its patrons, but can be providing by a specialized monetary service deliverr as a separate provision, such as an immurement service deliverr. A payment entry enables a payment deal by the transfer of information between a compensation portal (such as a website, mobile phone or interactive voice response service) and the front end computer or receiving bank. Payment entries are a service that helps dealers inductee ecommerce, in-app, and point of sale expenditures for a comprehensive variety of some means. The gateway is not straight involved in the coinage flow; typically it is a web server to which a dealer's website or POS system is linked. A payment gateway often links several acquiring banks and compensation methods under one system.

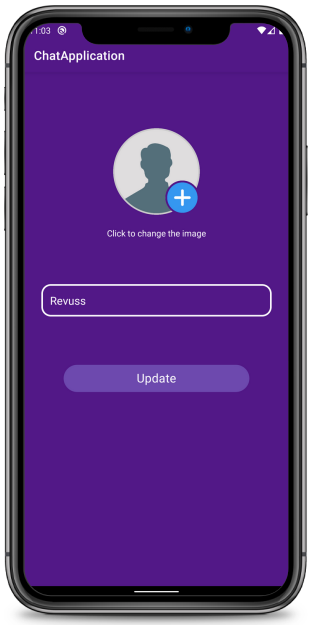


Fig 8.9 Screenshot of Profile Screen

Description

Update Profile is a simple Snippet that allows users who are logged in the front-end the ability to edit their profile and can change their username and profile image in the Cloud database.

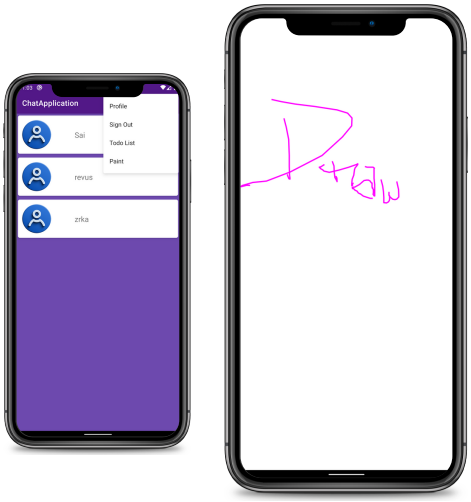


Fig 8.10 Screenshot of Paint feature

Description

This paint feature in this application allows the user to draw the shapes and sketches.

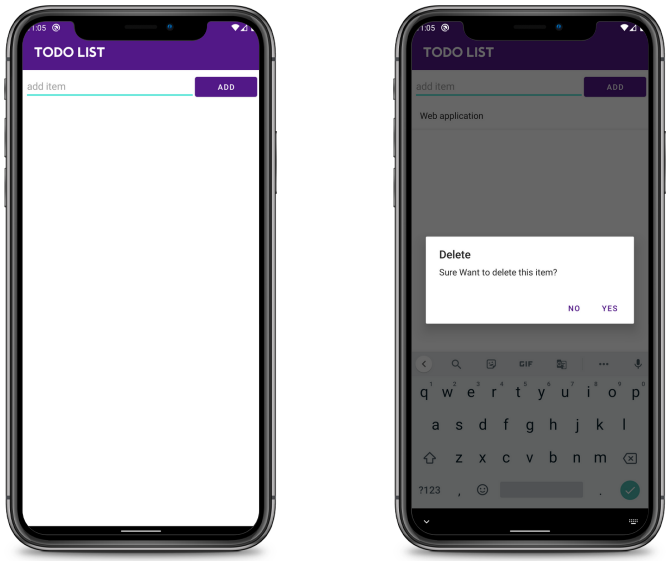


Fig 8.11 Screenshot of Todo List

Description

This activity delivers the user to make a list, where he or she can keep track of their work, with the most important tasks at the top of the list, and the least important tasks at the bottom. By keeping such a list, you make sure that your tasks are written down all in one place, so you don't forget anything important.

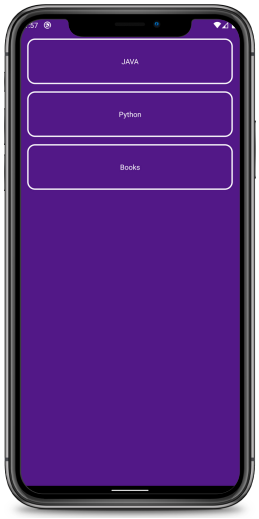


Fig 8.12 Screenshot of Learn

Description

Free Online Option Learning Modules in an Interactive manner to deliver knowledge about the basic programming knowledge about Java and Python Technologies with a playable video.

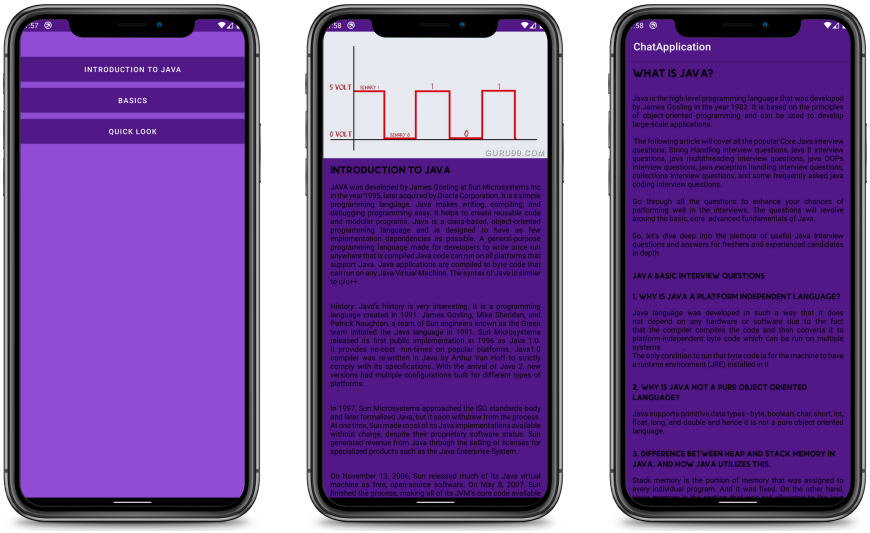


Fig 8.13 Screenshot of Java Panel

Description

This option Java delivers the user able to learn by accessing into the Java panel to learn with different types of topics deliverd.

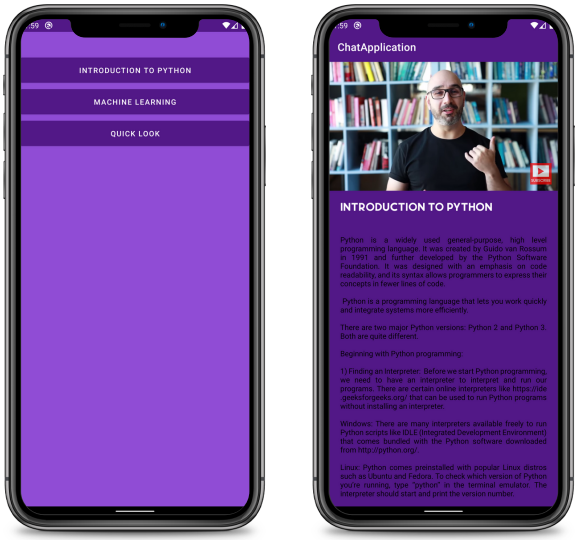


Fig 8.14 Screenshot of Python Panel

Description

This option Python delivers the user able to learn by accessing into the python panel to learn with different types of topics deliverd.

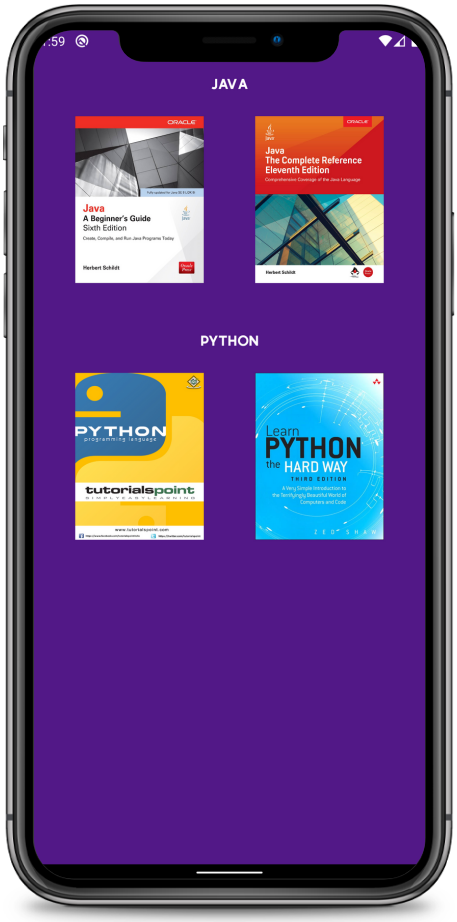


Fig 8.15 Screenshot of Free Book Store

Description

This program delivers the user not only learn by limited resources of topics deliverd.

CONCLUSION

CONCLUSION AND RECOMMENDATION

Conclusion

Chatting is a very mutual used application between the users. General operators use the immediate messaging services to interconnect with other separate users. In our project, we have providing with several enhanced structures for a chat application. The features like watercolour along through chat would be a amusing to use and interactive feature for a general user. The chat application is so aimed that the people could have a better practice of discussion. It has the possible to attract more and more operators to interrelate and connect. Social networks act mainly as transmission tools built around news feedstuffs filled with friends, family, and associates. In difference, messengers allow people to talk privately with the people they most care about. A recent Frank N. Magid Acquaintances survey of U.S. teens institute that only 9 percent deviseate Facebook as “safe” or “trustworthy.” Many CEOs of foremost messaging apps take a comparable stance, avoiding the collection of personal information and building stages free of hyper-targeted advertising. Chat apps may too feel “trustworthy” to young operators by contribution an amnesty from the watchful eye of parents and employers that have permeated traditional social networks.

Recommendation

In this proposed method, there are a few limits. Even though the application itself works well, abundant was learned during its growth. For this motive, we wrote a slope of possible developments/changes, some of which are informal to execute, others might require rewriting a significant quantity of the current source code. Apart from that, the Perfect application was too determined, which occasioned in many topographies not actuality able to be implemented during the course of the project.

The archetypal platform, labelled in the "Features" unit, had plenty of topographies. Many of them persist undone: Notifications, Status, Room roles. File sharing, Voice and video calls. Although our application already delivers the essentials to programmers who want to talk and share cipher themselves, having extra of these classical features complete would probably attract the consideration of more of them.

BIBLIOGRAPHY/REFERENCE

BIBLIOGRAPHY

Books referred :

- Java: The Complete Reference, Eleventh Edition, 11th Edition By Herbert Schild.
- GUI devise for Android Apps by Ryan Cohen, Tao Wang.
- Java Programming for Android Developers For Dummies By Burd.

References :

[1] Literature Survey : https://caes.hku.hk/acadgrammar/litrev/examples/ICQ1-Impact.htm (The Impact Of The Chat: A brief Literature Review by Vicky, Chau Ka Ki)

[2] Android API: http://developer.android.com/reference/packages.html

[3] Java 6 API: http://docs.oracle.com/javase/6/docs/api/

[4] Android Fundamentals: http://developer.android.com/guide/components/fundamentals.html

[5] The Java Tutorials: http://docs.oracle.com/javase/tutorial/

[6] Android User Interfaces: http://developer.android.com/guide/topics/ui/index.html

[7] Layout: http://developer.android.com/guide/topics/ui/declaring-layout.html

[8] Firebase : https://firebase.google.com/docs/reference/fcm/rest/v1/projects.messages

[9] Idys, E., & Sam, D.,(2020) “About native xmlhttp.” https://msdn.microsoft.com/enus/library/ms537505(v=vs.85).aspx. Accessed: 2016-12-18.

Websites referred :

- www.geeksforgeeks.org
- www.coursera.org
- www.Tutorialspoint.com
- www.wikipedia.org
- www.youtube.com

20A91F0048

ORIGINALITY REPORT

28%
SIMILARITY INDEX

PRIMARY SOURCES

1	origin.geeksforgeeks.org Internet	1487 words — 10%
2	www.geeksforgeeks.org Internet	525 words — 3%
3	www4.caes.hku.hk Internet	450 words — 3%
4	comp786.blogspot.com Internet	389 words — 2%
5	www.engadget.com Internet	319 words — 2%
6	www.simplilearn.com Internet	204 words — 1%
7	www.slideshare.net Internet	194 words — 1%
8	hbr.org Internet	132 words — 1%
9	pdfcoffee.com Internet	121 words — 1%
10	developer.mozilla.org Internet	