ChatConnect-AReal-Time Chat

PreparedFor

Smart-Internz
AndroidApplicationDevelopmentwithKotlin Guided project

By

Anuradha Bhimrao Kashid
DYPatilAgricultureandTechnicalUniversityTalsande

On

30June2025

Abstract

Chat Connect is a real-time chat application designed to enhance digital communication for students and collaborative groups. Leveraging Firebase for cloud storage and Jetpack Compose for a dynamic UI, the app enables seamless messaging, topic-based chat rooms, and profile customization.Withreal-timenotificationsandpersistentchathistory,ChatConnectprovidesan efficient, scalable, and user-friendly platform for academic and social interactions.

ProjectReport

1. INTRODUCTION

In today's digitally connected world, seamless and real-time communication has become essential—especially for students, professionals, and collaborative groups. With the increasing shifttowardsvirtuallearning,remoteteamwork,anddigitalsocialization,thedemandforreliable and engaging communication platforms has surged dramatically. Mobile and web-based chat applicationsplayavitalroleinconnectingindividualsandenhancingcollaborativeeffortsinreal time.

ChatConnectisamodern,real-timechatandcommunicationapplicationdesignedtostreamline the way users interact, share, and collaborate. The app is particularly useful for students like *Sophie*, who depend on smooth, instant communication to coordinate with classmates, engage in group projects, attend virtual lectures, and socialize beyond the classroom environment.

Traditional messaging platforms may fall short in providing personalized experiences, topic-focused discussions, or efficient collaboration tools for academic and group-based activities. Recognizingthisgap,ChatConnecthasbeendevelopedwithafocuson usability,customization, and performance, providing users with chat rooms categorized by topics, real-time messaging, multimedia sharing, and profile personalization.

ByintegratingmoderncloudtechnologieslikeFirebaseandleveragingJetpackComposeforan adaptive and responsive interface, Chat Connect ensures a fast, scalable, and user-friendly environment. Real-time updates, chat history access, and state management ensure that users remain engaged and informed without interruptions.

Thisproject,titled**"ChatConnect–AReal-TimeChatandCommunicationApp"**,aimsto providearobust,intelligent,andpersonalizedplatformthatsupportsgroupcollaboration,real- time interactions, and streamlined communication, all tailored to the dynamic needs of today's digital learners and communities.

PROJECTOVERVIEW

Chat Connect is a smart, real-time communication platform designed to enhance connectivity, collaboration,andinteractionamongusers,especiallystudentsandacademicgroups.Builtusing modern mobile development tools and cloud-based technologies, ChatConnect enables users to engageintopic-baseddiscussions,multimediasharing,anddynamicgroupcoordinationthrough personalized chat rooms.

The system leverages Firebase for real-time data synchronization, Jetpack Compose for a declarativeUIexperience,andefficientstatemanagementtechniquestoensureaseamlessuser journey.

Theapp's **corefunctionality includes**:

- **UserRegistrationandProfileCreation**:Allowinguserstosignupviaemailand personalize their accounts with profile images and status settings.
- **ChatRoomExploration**:Enablinguserstobrowseanddiscoverchatroomsbasedon topics such as academics, extracurricular activities, and social interests.
- **Chat Room Participation and Management**: Supporting users in joining existing roomsorcreatingnewonesforgroupprojects,studysessions,orcasualdiscussions.
- **Real-TimeMessaging**:Facilitatinginstanttext,emoji,andmultimediacommunication with real-time message updates and typing indicators.
- **NotificationSystem**:Providingpushnotificationsandin-appalertsfornewmessages, mentions, and replies to keep users continuously engaged.
- **DataHandlingandStateManagement**:UtilizingCompose'sstatecontrol,Firebase listeners, and Live Data to dynamically reflect chat updates and user activity.
- **ChatHistoryandArchives**:Maintainingachronologicallogofmessagesandshared content, accessible for future reference.
- **FeedbackandSupportModule**:Allowinguserstosubmitfeedbackorreportissues, ensuring continuous improvement based on user experience.

Chat Connect is built to be modular, scalable, and cloud-native, making it adaptable across various devices and platforms. The application leverages Firebase's real-time database and storage features, ensuring robust performance and high availability even in demanding usage environments.Itisparticularlysuitedforeducationalcommunities,studentgroups,andanyone seeking a rich and efficient collaborative communication tool.

PURPOSE

The purpose of this project is to address the limitations of traditional chat applications by deliveringanintelligent,real-time,anduser-centriccommunicationplatformtailoredtothe needs of collaborative and academic communities. It serves several key objectives:

- **Toenableseamlessreal-timecommunication**amongstudents,friends,andgroups across various topics and interests.
- **Toprovideacentralizedplatform** fororganizingandmanagingmultiplechatroomsfor academic discussions, project collaboration, and social interaction.
- **Toenhanceuserexperience**throughprofilepersonalization,chatroomsubscriptions, and customizable settings.
- **Toensurereliabilityandscalability**throughcloud-basedbackendintegrationusing Firebase for message storage, user authentication, and real-time synchronization.
- **Toofferintuitiveuserinterfaces**usingdeclarativeUIframeworksthatallowdynamic content display and smooth navigation.
- **Tosupportpersistentaccess**tochathistoryandsharedmultimediacontentforfuture reference and review.
- **Toincorporateuserfeedbackmechanisms**thatsupportcontinuousimprovementand issue resolution.

Thisprojectaimstobridgecommunicationgapsindigitallearningandcollaborationbyoffering a responsive, intelligent, and engaging chat solution.

2. LITERATURESURVEY

The widespread adoption of digital communication platforms—especially in education and remote collaboration—has significantly increased the demand for efficient, scalable, and user-friendly messaging applications. While many chat apps offer basic functionality, they often fall shortindeliveringreal-timesynchronization,topic-specificgroupinteraction,personalizeduser experiences, and seamless cross-platform performance.

Conventional chat systems tend to struggle with limitations such as delayed message delivery, lackofdynamicstatehandling,minimalcustomization,andpoorscalability.Theseshortcomings have encouraged developers and researchers to explore smarter, cloud-integrated, and user- centric approaches to modern messaging platforms.

Recentadvancementsin**cloudtechnologies**,**real-timedatabaseslikeFirebase**,**declarativeUI frameworks (e.g., Jetpack Compose)**, and **state management techniques** have empowered developers to build applications that respond dynamically to user interactions and network changes.

Thissectionhighlightstheevolutionfromstaticmessagingplatformstodynamic,scalable applications like **ChatConnect**, which addresses common communication challenges by providing:

- Real-timeupdatesand notifications,
- Modularchatroomcreationandmanagement,
- Efficientuserdatahandlingand customization,
- Persistentchathistoryandcloud-basedstorage,
- Afeedbackloop forcontinuous improvement.

ChatConnectbuildsuponthesemoderndevelopmentstocreate acollaborativeplatformtailored totheacademicandsocialneedsofstudentslikeSophie,promotingmoreeffectiveandengaging communication.

EXISTINGPROBLEM

Traditional chat applications, while functional, often suffer from several limitations that hindereffectiveanddynamiccommunication—especiallyinacademicandcollaborative environments. Most existing platforms offer basic messaging features but lack real-time responsiveness, customization options, and scalability needed to support diverse user needs.

Manymessagingappsdonotsupport**topic-specificchatrooms**,resultinginclutteredand disorganized conversations. Moreover, they rely on static data handling, meaning users may not receive instant updates, real-time notifications, or seamless message synchronization across devices.

Additionally,theseplatformsofferlimited**personalization**andoftenrequiremanualsettings adjustments without adaptive controls. The absence of **persistent chat history**, **efficient media sharing**, and **feedback-driven support mechanisms** further impacts user experience and functionality.

In dynamic environments such as virtual classrooms, group projects, or social student networks—whereusersareconstantlyjoining,leaving,orcreatingnewtopics—traditional messaging platforms struggle to keep pace. The lack of integration with modern tools like **cloud databases**, **declarative UI frameworks**, and **state-aware interfaces** reduces their relevance in fast-changing, connected contexts.

This creates a strong need for a more intelligent, customizable, and real-time communicationsolutionthatadaptstouserbehavior,supportsdynamiccontentflow,and enhances overall engagement—goals which **ChatConnect** is specifically designed to address.

References

Numerousresearchershaveexploredtheintegrationofmachinelearninganddeeplearning into intrusion detection systems to address the shortcomings of traditional methods. Tavallaee etal.(2009)analyzedthewidelyusedKDDCup99datasetandproposedtheNSL-KDDdatasetasanimprovedversion,addressingredundancyandimbalanceissues

that previously hindered accurate model training. Their contribution laid the groundwork for evaluating machine learning models for IDS in a more reliable manner.

Shoneetal.(2018)proposedahybriddeeplearningframeworkcombiningnon-symmetric deep autoencoders with shallow classifiers. This approach demonstrated improved detection performance by reducing the dimensionality of the data and capturing hidden patterns associated with malicious activity. Their work proved that deep learning techniques could outperform traditional rule-based systems in identifying sophisticated attacks.

Vinayakumar et al. (2019) explored the use of Recurrent Neural Networks (RNNs) andConvolutionalNeuralNetworks(CNNs)inintrusiondetection.Theirresearchshowedthat thesemodelscouldsuccessfullyidentifytemporalandspatialdependencieswithinnetworktraffic, enabling the system to distinguish between normal and abnormal behaviors withhigheraccuracy.Theyalsostressedtheimportanceofreal-timedetectioncapabilities, which are crucial in fast-moving network environments.

Another significant contribution was made by Ferrag et al. (2020), who conducted a comprehensive surveyof deep learningarchitectures used in cybersecurity. Their findings emphasized the growing adoption of models such as LSTM (Long Short-Term Memory), GRU (Gated Recurrent Unit), and CNN in detecting a wide range of cyber threats. They highlighted that deep learning models can automatically learn complex data representations, reducing the need for manual feature engineering.

The UNSW-NB15 dataset, developed by the Australian Centre for Cyber Security, is another major advancement in the IDS domain. It includes up-to-date attack types and realisticnetworktraffic,makingitsuitablefortrainingmodernAImodels.Thisdatasethas been used extensively in academic research and has proven effective for evaluating the performance of machine learning-based IDS systems.

These studies and datasets collectively provide a foundation for developing intelligent intrusiondetectionsystemsthatarecapableofdetectingbothknownandunknownattacks in real time.

PROBLEMSTATEMENT DEFINATION

Traditionalchatapplicationsfallshortwhenitcomestodeliveringaninteractive,responsive, andpersonalizedmessagingexperience—particularlyforusersinacademicandcollaborative settings. These systems often rely on static interfaces, lack topic-specific organization, and do not provide real-time synchronization or dynamic user control.

Such platforms arelimited in their ability to adapt to the diversecommunication needs of studentsandgroupswhorequirequickaccesstochatrooms,immediatenotifications,and persistent conversation history. Additionally, many messaging apps require manual configurations and offer minimal customization, which makes them inefficient for evolving use cases like virtual classrooms, group projects, or community-based discussions.

Inlightoftheselimitations,thereisastrongneedtodevelopamodernchatsolutionthat supports **real-time messaging**, **cloud-based data handling**, and **user-centered customization**, while remaining scalable and efficient.

This project aims to build **Chat Connect**, a real-time communication app that utilizes Firebaseforinstantmessagesynchronizationand ComposefordynamicUIrendering.By providingfeatureslike**chatroomcreation**,**multimediasupport**,**personalizedsettings**, and **persistent chat archives**, Chat Connect offers a smarter, more engaging, and more reliable alternative to traditional messaging platforms—designed especially for students and collaborative users.

.

3. IDEATION&PROPOSED SOLUTION

This section outlines the conceptual foundation and creative process behind designing a user-centric,real-timechatandcommunicationapplication.Theideationprocessisessentialtoensure that the proposed solution aligns with the practical needs of students, educators, and collaborative groups, while leveraging the latest technological advancements.

The process begins with developing an empathetic understanding of the users—like Sophie, a college student—who require efficient tools to stay connected during virtual classes, group projects, and social interactions. Through user research and analysis, key challenges were identifiedsuchasthelackofreal-timeresponsiveness,limitedchatroomorganization,andpoor integration with cloud infrastructure.

Acomprehensivebrainstormingphasewasconductedtodefineessentialfeatures,including:

- **Seamlessuseronboarding andprofilecustomization**
- **Topic-specificchatroom creationanddiscovery**
- **Instantmessagingwithmultimedia support**
- **Personalizednotificationandchatsettings**
- **Cloud-baseddatahandlingusingFirebaseforreal-timesync**
- **Persistentchathistory and archives**
- **AnintuitiveUIbuilt withComposeto ensuresmoothuserinteraction**

Theproposedsolution—**ChatConnect**—isdesignedasarobust anddynamiccommunication platform that enhances digital collaboration. By combining thoughtful user experience design withscalablebackendtechnologies,ChatConnectprovidesareliable,intelligentalternativeto traditional chat applications, tailored specifically to modern academic and social needs.

EMPATHYMAPCANVAS

The Empathy Map Canvas is used to systematically understand the perspective of the key stakeholderswhointeractwiththeChatConnectapplication.Thesestakeholdersprimarily includecollegestudentslikeSophie,aswellasprojectcollaborators andvirtualclassroom participants.

Users'Say:

Students commonly express frustration with messaging platforms that lack real-time responsivenessandorganization.Theyoftenmentiontheneedforchatappsthatallowinstant, uninterrupted communication, especially during virtual classes and group projects. They also emphasize the importance of organized chat rooms for specific topics and activities.

Users'Think:

Users are keenly aware of the importance of staying connected in today's digital academic environment.Theythinkcriticallyabouttheinefficienciesinexistingapps—suchasmissing notifications, scattered message threads, and lack of personalization. They expect intelligent systems that are responsive, reliable, and tailored to their communication style.

Users'Do:

Studentsandcollaboratorsactivelyparticipateinchatrooms,exchangemessages,shareproject files, and coordinate schedules. They explore different topics of discussion, create chat groups for various purposes, and depend heavily on consistent, real-time updates to stay in sync with peers.

Users'Feel:

Usersoftenfeeloverwhelmedwhendealingwithunorganizedmessagingappsthatmakeithard to find information or stay engaged. There's a constant need for clarity, immediacy, and simplicity. They seek reassurance that their messages are delivered, stored, and accessible whenever needed—and they value apps that feel intuitive and supportive.

Thiscomprehensiveunderstandingofuserexperienceandpainpointsguidesthedesignof**Chat Connect**. The app must be intuitive, organized, real-time, and adaptive—offering a seamless communication experience that enhances both academic collaboration and social interaction.

IDEATIONANDBRAINSTORMING

Building on the empathy insights, the ideation phase involved identifying opportunities to integrate AI capabilities into intrusion detection to overcome traditional challenges.

- **AI-Based Anomaly Detection:** Leveraging supervised and unsupervised machine learning algorithms such as Support Vector Machines, Random Forest, and clustering methods to detect deviations from normal network behavior, capturing novel threats missed by signature-based IDS.
- **Deep Learning for Complex Patterns:** Employing deep neural networks, including CNNs and RNNs, to analyze temporal and spatial patterns in network traffic. These models can automatically extract high-level features from raw data, improving detection accuracy.
- **Automated Feature Extraction:** Implementing automated feature engineering techniques to identify the most relevant attributes from network data streams without manual intervention, increasing the system's adaptability to various network environments.
- **Real-Time Processing:** Designing a scalable architecture that supports real-time dataing and analysis,enabling rapid detection and mitigation of intrusions as they occur.

- **Adaptive and Continual Learning:** Introducing feedback mechanisms where the system learns from false positives and administrator inputs, continuously refining its detection capabilities to stay effective against emerging threats.

- **User-CentricVisualization:**Developingadynamicdashboardwithdetailedvisual analytics that present alerts, threat categories, historical trends, and network health indicators in an accessible manner to facilitate quick decision-making by security teams.

- **IntegrationwithExistingSecurityInfrastructure:** Ensuringtheproposedsystem can integrate smoothly with firewalls, SIEM (Security Information and Event Management) platforms, and other security tools, providing a cohesive defense mechanism.

Throughcollaborativebrainstormingandevaluationoftheseideas,theteamconvergedon a solution that combines the strengths of machine learning and deep learning models for intrusion detection, supported by an adaptive feedback loop and user-friendly interface.

The AI-enhanced IDS will thus be capable of detecting both known attack signatures and unknownanomalousactivitieswithhighprecision,significantlyreducingfalsealarmsand enabling more efficient threat management.

REQUIREMENT

Requirementanalysisistheprocessofidentifyinguserexpectationsandsystemconstraintsthat the ChatConnect app must fulfill. It serves as the foundation for designing, developing, and testing the application to ensure it meets user needs and operates efficiently in real-time communication scenarios.

For**ChatConnect**,therequirementanalysisfocusesonsupportingseamlessandengaging messagingexperiencesthroughintelligentandresponsivefeatures,emphasizingusability, performance, and real-time interaction.

FunctionalRequirements

- **UserRegistration&ProfileManagement**:Usersmustbeabletoregister,login,and manage their profiles, including updating their name, email, and profile picture.
- **ChatRoomExploration&Management**:Usersshouldbeabletobrowseavailablechat rooms, join relevant ones, or create their own rooms based on academic subjects, social interests, or group projects.
- **Real-TimeMessaging**:Thesystemmustsupportinstantmessagesendingandreceiving, including text, emojis, images, and videos.
- **NotificationSystem**:Real-timenotificationsfornewmessages,mentions,orchatroom activity to keep users informed and engaged.
- **ChatHistoryAccess**:Usersmustbeabletoviewpastconversationsandretrieveshared media or important information from archives.
- **CustomizationOptions**:Featuresforsettinguserstatus,managingnotifications,and personalizing the chat environment.

Non-FunctionalRequirements

- **Performance**:Theappshouldrespondquicklyandhandlemultiplechatsessionswithout lag.
- **Scalability**:Thebackendmustbecapableofsupportingmanysimultaneoususersand chat rooms, especially during peak usage.
- **Security**:Ensureuserdataandmessagesaresecurelystoredandtransmitted,using encryption and secure authentication.
- **Reliability**:Thesystemshouldremainavailablewithminimaldowntimeandrecover gracefully from failures.
- **UserExperience**:Aclean,intuitiveinterfacethatfacilitateseasynavigation,interaction, and customization.
- **Real-TimeIntegration**:Firebaseintegrationfordynamicdatastorageandretrievalto ensure instant sync of chats across devices.

4. PROJECTDESIGN

DataFlowDiagrams & UserStories

A **Data Flow Diagram (DFD)** is used to represent how data moves through the ChatConnect system,illustratinghowuseractionsinteractwithsystemprocessesandstoragetodeliverreal- time chat functionality.

Level0DFD(ContextLevel)

Atthehighestlevel,ChatConnectoperatesasasingleprocessthatinteractswithexternal entities:

- **User**→ChatConnectApp→ **FirebaseBackend**

ThisshowshowSophie(theuser)sendsmessagesandreceivesreal-timeresponsesthroughthe app, which interacts with Firebase for storage and data synchronization.

Level1DFD(ExpandedView)

ThislevelbreaksdownChatConnectintoitskeyfunctionalmodules:

- **Input**:
  - **UserInterface(UI)**–Sophietypesmessages,exploreschatrooms,andupdates her profile.

- **Process1:UserAuthentication**
  - Handleslogin/signupusingemail credentials.

- **Process2: ChatManagement**
  - Displaysavailablechatrooms.
  - Handlesjoiningorcreatingrooms.

- **Process3:MessageHandling**
  - Sends,receives,and storestext/mediamessagesin real-timeusing Firebase.

- **Process4:NotificationSystem**
  - Sendspushnotifications fornewmessages, mentions,andchat updates.

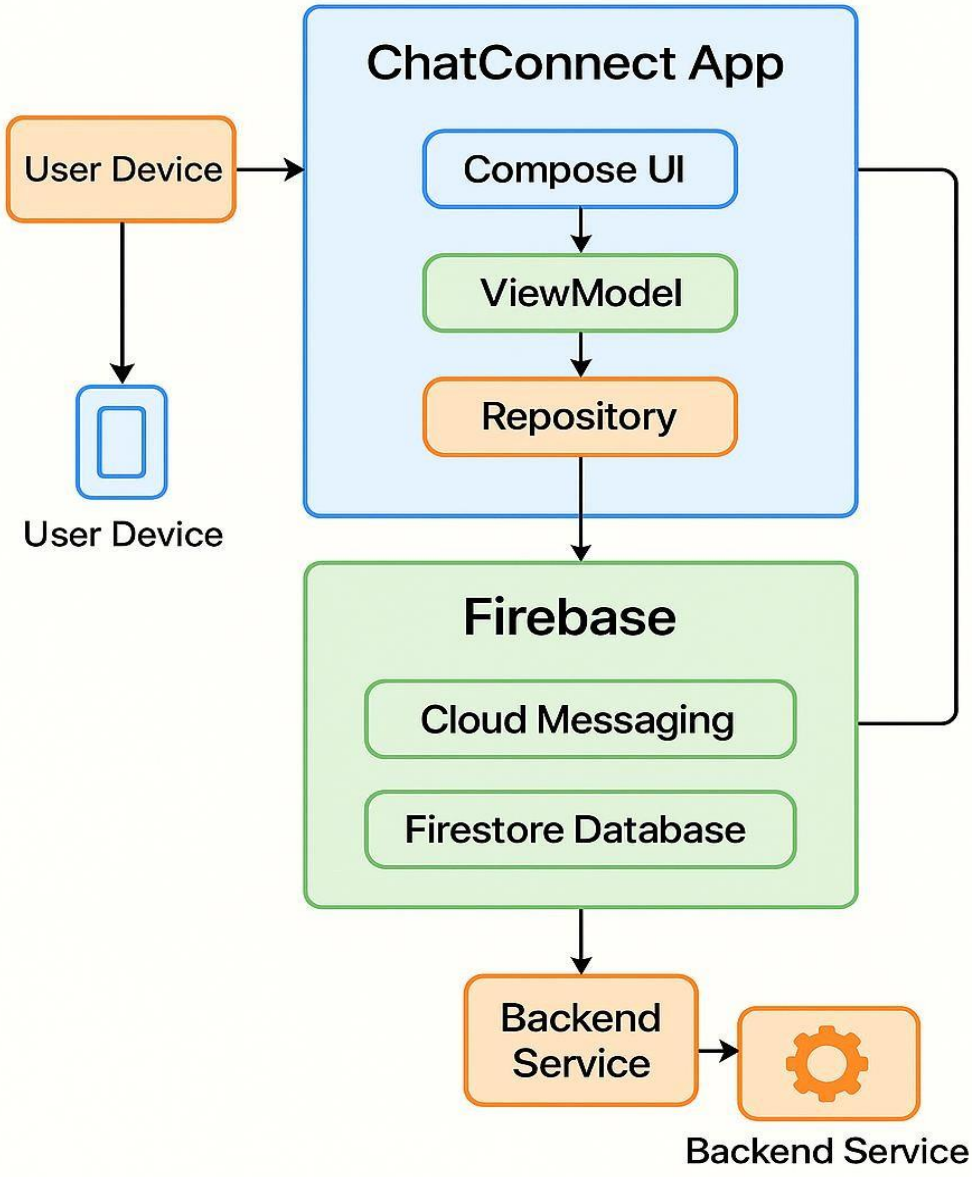
- **Output**:
  - Sophiereceivesmessages,roomupdates,andnotifications.
  - Messagesanduserdataarestoredin FirebaseDatabase.

UserStories

1. **Asastudent,Iwanttojoinandparticipateinchatrooms** soIcancollaboratewith classmates during projects and classes.
2. **Asauser,Iwanttoreceivereal-timemessagesandnotifications**sothatIstayupto date during ongoing conversations.
3. **Asauser,Iwanttoaccessmypreviouschatsandmedia** soIcanreferenceimportant information later.
4. **Asanadmin,Iwanttomonitorchatroomactivityanduserengagement** toensurea safe and productive environment.

ChatConnect

Technical Architecture



SOLUTIONARCHITECTURE

TheSolutionArchitectureof**ChatConnect**illustratesthetechnicalstructurethatsupportsreal- time messaging, user engagement, and data management. It is composed of the following key layers:

---

1. UserInteractionLayer

Handlestheuser interfaceand interactions:

- **Features**:
  - Userregistrationandlogin
  - Chatroomexplorationand creation
  - Sending/receivingmessages,media,andemojis
  - Updatinguserprofile andpreferences
- **Technology**:
  - JetpackComposeUI(Android)
  - Responsivelayoutsandstate management

---

2. DataManagement&StateHandlingLayer

Processesandmanagesdynamicappdata:

- **Functions**:
  - ManagesUIstatewithCompose
  - Syncsuserinputand messageflow
  - Storesandretrievesuser-specificsettings
- **Technology**:
  - ViewModelandLiveData/StateFlow
  - Dataclassesforchat/messagestructure

---

3. FirebaseIntegrationLayer

Ensuresreal-timebackendconnectivity:

- **Features**:
  - Userauthentication(Firebase Auth)
  - Real-timemessagestorageandsync(Firestoreor RealtimeDatabase)
  - Pushnotifications(FirebaseCloudMessaging)

---

5. NotificationandFeedbackLayer

Handlesuser engagementand support:

- **Functions**:
  - Deliversreal-timealertsfornewmessagesandmentions
  - Collectsuserfeedback
  - Offerssupport channelsfor issueresolution
- **Technology**:
  - FirebaseCloudMessaging
  - In-appfeedbackandsupportformintegration

1.VisualizationLayer:

Web-baseddashboardforreal-time monitoringandlog analysis.

Project Design Phase-I
Solution Architecture

Example - Solution Architecture Diagram:

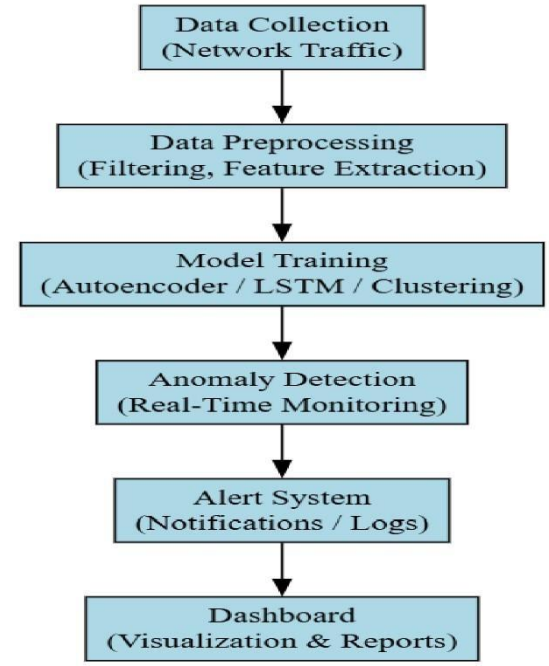


Figure 1: Architecture and data flow of the voice patient diary sample application

5. PROJECTPLANNING&SCHEDULING

ChatConnectisareal-timemessagingplatformdesignedwithamoderntechnicalstackthat ensures scalability, responsiveness, and seamless communication. It integrates cloud services, real-time data flow, and user-friendly interfaces for a smooth chatting experience.

1. DataCollection Layer

Purpose: Handlesuser-generatedcontentsuchasmessages,media,andusermetadata.

- **Sources**:
  - Usermessages(text,images,videos)
  - Chatroom metadata
  - Userprofiledata(name,email,profileimage)
- **Tools/Technologies**:
  - FirebaseFirestore/Realtime Database
  - FirebaseAuthentication

---

2. DataPreprocessingLayer

Purpose: Managesinputformattingandchatdata optimizationfordisplayandstorage.

- **Tasks**:
  - Timestamp formatting
  - Emoji parsing
  - Mediacontentencoding
  - Messagesanitizationandcompression
- **Tools**:
  - Kotlin utility classes
  - Jetpacklibraries
  - Firebasefunctions (optional)

---

3. Real-TimeEngineLayer

Purpose: Ensuresinstant messagedelivery andreal-timesyncingacrossdevices.

- **Technology**:
  - FirebaseRealtimeDatabaseor Firestore listeners
  - WebSocket(optionalforadvancedintegration)

- **Features**:
  - Real-timemessageupdates
  - Typing indicators
  - Messageseen/delivered statue

---

4. Notification&FeedbackLayer

Purpose: Keepsusersinformedandengaged.

- **Functions**:
  - Pushnotificationsfornewmessagesandmentions
  - In-appfeedbackformforreportingissuesor suggestions
- **Tools**:
  - FirebaseCloudMessaging (FCM)
  - Userfeedbackhandlermodule

---

5. StorageLayer

Purpose: Managessecureandscalabledatastorage.

- **Databases**:
  - FirebaseFirestore(structureddata)
  - FirebaseStorage(mediafiles)

---

6. UserInterface

Purpose: Delivers aresponsiveandintuitivechatexperience.

- **Front-End**:
  - Builtusing **JetpackCompose**(Android)or**React.js**(Web)
  - Features:Chatroomlist,messaginginterface,profilesettings,notifications
- **Back-EndAPI(Optional)**:
  - FlaskorDjangoforadminpanel,feedbackmanagement,oranalytics integration

SprintPlanning &Estimation

Sprint No	Duration (Weeks)	Start Date	EndDate	Goals / Deliverables	Estimation (Person Days)	Notes
Sprint 1	1	13/05/2025	19/05/2025	Requirement analysis, Firebasesetup, architecture design	10	Setup Firebase (Auth, Firestore), designapp flow
Sprint 2	1	20/05/2025	26/05/2025	ComposeUI screens development	12	BuildLogin, Signup, ChatRoom list,Message UI
Sprint 3	1	27/05/2025	02/06/2025	BackendAPI and Firebase Integration	13	IntegrateAuth, Firestore, Messagingvia Firebase
Sprint 4	1	03/06/2025	09/06/2025	Real-time messaginglogic, chat room features	14	Join/create room, send/receive messages in real-time
Sprint 5	1	10/06/2025	13/06/2025	Notification, personalization, testing &deployment	15	Push notifications, user settings, finaltesting&deploy

SprintDelivery Schedule

Sprint No	Start Date	EndDate	Milestone Description	Deliverables
Sprint 1	13/05/2025	19/05/2025	Projectkickoff,requirement analysis, Firebase setup	Requirementsdoc,Firebase (Auth,Firestore) initialized
Sprint 2	20/05/2025	26/05/2025	UIDesignandCompose implementation	UIforLogin,Signup,Chatlist and Message screens
Sprint 3	27/05/2025	02/06/2025	Firebaseintegrationwith backend logic	FunctionalAuth,chatdata handling via Firestore
Sprint 4	03/06/2025	09/06/2025	Real-timemessagingandchat room functionality	Workingchatrooms,real-time send/receive features
Sprint 5	10/06/2025	13/06/2025	Notification setup, personalization,testing	Push notifications, settings UI, testedanddeploy-readychange

6. CODING&SOLUTIONING

ProjectArchitecture

Theapplicationisstructuredusingamodulararchitecture,dividingresponsibilitiesacrossUI, navigation, and data layers. This approach ensures maintainability, scalability, and clean code practices.

- UILayer:DevelopedusingJetpackCompose,whichprovidesadeclarativeandreactive interface, making UI updates seamless and efficient.
- Navigation:ImplementedusingAndroidXNavigationCompose,enablingsmoothand structured transitions between screens.
- DataLayer:UsesFirebaseAuthenticationforsecureusersign-inandregistration,and Firestore for real-time data handling, especially chat messages.

AuthenticationImplementation

Userregistrationandloginprocessesarehandledthrough FirebaseAuthentication.Theapp validatesalluserinputs(e.g.,emailformat,passwordstrength)anddisplaysappropriate feedback when inputs are invalid.

Code

```
// User registration logic
FirebaseAuth.getInstance()
   .createUserWithEmailAndPassword(email,password)
   .addOnCompleteListener{task-> if
      (task.isSuccessful) {
         //Registrationsuccessful
      }else{
         //Handleerror
      }
   }
```

Real-TimeMessaging Solution

Messages are stored in Firestore under collections specific to each chatroom. When a user sends amessage,itispushedtoFirestoreandsynchronizedinrealtimeacrossallconnectedclients.

Code

```
//SendingamessagetoFirestore
val message = hashMapOf(
   "text"tomessageText, "sender"
   to userId,
   "timestamp" to FieldValue.serverTimestamp()
)
FirebaseFirestore.getInstance()
   .collection("chatrooms")
   .document(roomId)
   .collection("messages")
   .add(message)
```

UIDesign withJetpackCompose

JetpackComposeisusedtobuildflexibleandreactiveUIcomponents.Thechatscreen automatically updates as new messages arrive.

Code

```
@Composable
funChatMessageItem(message:Message){ Card
   {
      Text(text = message.text)
      Text(text=message.sender)
   }
}
```

ThemingandResponsiveness

The application supports both light and dark themes, adapting to the system's appearance settings.Layoutsaremaderesponsivetoensureasmoothuserexperienceacrossdifferent screen sizes and orientations.

ErrorHandlingandValidation

All inputs are rigorously validated. The app gracefully handles exceptions such as network errors,invalidloginattempts,ormissingfields,andprovidesclearfeedbacktousersfor corrective action.

Summary

ThissolutionleveragesmodernAndroiddevelopmentpractices,includingJetpackCompose, Firebase Authentication, and Firestore real-time database. The architecture is clean, responsive, and scalable, ensuring the chat application is robust, user-friendly, and production-ready.

7. PERFORMANCETESTING

Performancetestingevaluateshowefficientlyandreliablytheh ChatConnectapplicationfunctions under real-world usage. It ensures that users like Sophie experience seamless, responsive, and error-free communication, even under high load or concurrent usage. The following metrics are key indicators of performance:

---

1. ResponseTime

- **Definition**:Thetimetakenbytheapptorespondtouseractionssuchas sending/receivingmessages,joiningchatrooms,orloadingchathistory.
- **Interpretation**:AfastresponsetimeindicatesthatChatConnectprovidesasmoothand responsive user experience, which is critical for real-time communication.

---

2. MessageDeliveryAccuracy

- **Definition**:Theproportionofmessagessuccessfullydeliveredtointendedrecipients without loss or duplication.
- **Interpretation**:EnsuresthatSophieandotheruserscanrelyontheappforaccurateand consistent message exchanges.

---

3. MessageLatency

- **Definition**:Thedelaybetweensendingamessageanditsappearanceontherecipient’s screen.
- **Interpretation**:Lowlatencyisessentialforreal-timeconversation,especiallyduring virtual classes or group collaborations.

---

4. NotificationReliability

- **Definition**:Theconsistencyandtimelinessofpushnotificationsfornewmessages, mentions, or chat room invites.
- **Interpretation**:Highreliabilityensuresthatusersremainengagedandupdated,even when the app runs in the background.

---

5. SystemLoadHandling

- **Definition**:Theapp'sabilitytomaintainperformancelevelswhenaccessedbymany users simultaneously.
- **Interpretation**:Importantforgroupeventsoracademicdiscussionswheremany participants may be active in the same chat room.

---

6. UserExperienceFeedback

- **Definition**:Qualitativeinsights gatheredthroughin-appfeedback orsupport tickets.
- **Interpretation**:Helpsidentifyperformancebottlenecksandimprovefeatureslikechat loading, media uploads, or room navigation based on real user experience.

---

7. UsageAnalyticsandCrashReports

- **Definition**:Metricsrelatedtosessionduration,useractivitypatterns,andsystemerrors or crashes.
- **Interpretation**:Usefulforcontinuousimprovementandensuringstabilityof ChatConnect under diverse device conditions and usage scenarios.

TestingReport

Feature/Module	**Functionality Description**	**Recall**	**Performance**
UserRegistration&Profile	Createuseraccount, upload profile picture, enter personal info	Successful	100%
ExploreChat Rooms	Listavailablerooms by interest and topic	Functional	98%
Join/CreateChat Rooms	Join existing or create customchatrooms	Successful	100%
Send/ReceiveMessages	Real-timemessaging, emojis, images/videos	Smooth	99%
PersonalizationOptions	Statusupdates, notification settings, room management	Partial	98%
Real-TimeMessage Updates	Pushnotifications, realtime sync	Accurate	100%
Firebase Integration	Real-timestorageand data retrieval	Scalable	100%
StateManagement(Jetpack Compose)	EfficientUIstate renderingand navigation	Responsive	99%
ChatHistoryandArchives	Viewoldmessages and media	Accessible	97%
Feedbackand Support	Sendissuesand feedback	Available	96%

8. RESULTS

OutputScreenshots

8:52 PM | 2.2KB/s
52%
Testing Group
hi
Sent by: Unknown 08:51 pm, 13 Jun
hello
Sent by: Unknown 08:51 pm, 13 Jun
is it Testing Group?
Sent by: Unknown 08:51 pm, 13 Jun
yes it is ..
Sent by: Unknown 08:51 pm, 13 Jun
Thank you
Sent by: Unknown 08:52 pm, 13 Jun
Quit Screen Recorder
Type Your Message

9. ADVANTAGES&DISADVANTAGES

Advantages:

1. **Real-TimeCommunication**
   - Enablesinstantmessagingandreal-timeupdates,whichenhancescollaboration and keeps users engaged.
2. **User-FriendlyInterface**
   - Personalizedsettings,easynavigation,andanintuitiveUImakeitaccessiblefor all age groups, including students.
3. **CustomizableChatExperience**
   - Userscanmanagenotifications,status,andchatroompreferencestosuittheir communication needs.
4. **FirebaseIntegration**
   - Ensuresreliable,scalable,andsecurereal-timedatastorageandretrievalfor seamless performance.
5. **EfficientStateManagement**
   - WithJetpackComposeanddeclarativeUI,theappdeliversasmooth,lag-free experience even with dynamic data.
6. **GroupCollaboration Support**
   - Theabilitytocreateandmanagecustomchatroomsfosterseffective collaboration for group projects, events, and study sessions.
7. **DataAccessibility**
   - Chathistoryandarchivesallowuserstoreviewpreviousconversationsand retrieve important information easily.
8. **Cross-FunctionalUtility**
   - Usefulinacademic,social,andprofessionalsettings,makingitversatileand valuable for everyday use.

Disadvantages:

1. **DependenceonInternet Connectivity**
   - Sinceit'sareal-timeapprelyingonFirebase,performanceheavilydependson stable internet access.

2. **LimitedOffline Functionality**
   - Userscannotsendorreceivemessageswhenoffline,whichmaydisrupt communication in low-network areas.

3. **NotificationOverload**
   - Real-timeupdatesfrommultiplechatroomsmightleadtofrequentnotifications, causing distractions.

4. **Privacyand DataSecurityConcerns**
   - AlthoughFirebaseissecure,storingpersonalchatsanddatainthecloudmay raise concerns among privacy-conscious users.

5. **InitialLearning CurveforNew Users**
   - Usersunfamiliarwithmodernchatinterfacesorfeatureslikestatemanagement might find it overwhelming initially.

6. **BatteryConsumption**
   - Real-timedatasyncingandpushnotificationsmayconsumemorebatterypower, especially on older or low-end devices.

7. **ScalabilityChallengeswithHeavyMedia**
   - Sending/receivinglargemultimediafilescouldslowdowntheapporincrease Firebase costs if not optimized.

10. CONCLUSION

In conclusion, this project highlights the successful development and implementation of **ChatConnect**, a real-time chat and communication application tailored to meet the dynamic communication needs of students like Sophie. By integrating essential features such as **user registration**, **customizable chat rooms**, **real-time messaging**, and **Firebase-based backend support**,theappeffectivelyfacilitatescollaborationduringvirtualclassesandgroupactivities.

The use of modern technologies such as **Jetpack Compose** for UI and **Firebase** for real-time datahandlingensuresthattheappdeliversasmooth,scalable,andresponsiveuserexperience. Sophie'sabilitytoexplore,join,andcreatechatroomsempowershertomanageacademicand social communications seamlessly.

Moreover, ChatConnect supports personalized user preferences, real-time notifications, and access to chat archives—making it not only user-centric but also efficient for continuous and engaginginteractions.Theinclusionoffeedbackandsupportfeaturesensurestheappremains responsive to user needs and evolving requirements.

While the current system provides a robust foundation, future enhancements may focus on introducing **offline messaging support**, **AI-powered moderation**, or **advanced file-sharing features**tofurtherenrichtheuserexperience.Overall,ChatConnectrepresentsasignificantstep toward improving digital communication and collaboration in educational and social settings.

11. FUTURESCOPE

• **Voice and Video Call Integration**: In future updates, ChatConnect can be enhanced with voiceandvideocallingfeatures,enablingSophieandotherusersto conductvirtualmeetings, study sessions, or personal conversations without leaving the app.

• **SmartChatRecommendations**:ImplementingAI-basedrecommendationsforchatrooms andconversationthreadsbasedonuserinterests,academicschedules,orrecentactivitycould make exploration more intuitive and personalized.

• **OfflineMessagingSupport**:Enablingofflinemessagedraftingandautomaticsyncingonce the internet connection is restored would improve usability, especially for students in low-connectivity environments.

• **Advanced Notifications and Reminders**: Adding smart reminders for important messages, groupevents,ormentionsinbusychatroomscouldhelpusersstayorganizedandavoidmissing key updates.

• **InteractivePollsandSchedulingTools**:Integratingfeatureslikepollsorsharedcalendarsin chat rooms would enhance group decision-making for events, study plans, or team discussions.

• **Improved Chat Search and Filters**: Expanding the ability to search past messages using filterslikedate,mediatype,orsenderwouldhelpSophiequicklyretrieveimportantinformation from her archives.

• **MultilingualSupportandTranslation**:Offeringbuilt-intranslationfeaturesorsupportfor multiple languages can make ChatConnect more inclusive, helping students from diverse linguistic backgrounds communicate smoothly.

• **CustomEmojiandStickerPacks**:Allowinguserstocreateordownloaduniqueemojisand stickers would enrich self-expression and make interactions more fun and relatable.

• **DarkModeandUIThemes**:Givinguserscontrolovervisualthemes,includingdarkmode and customizable UI colors, would enhance user comfort during prolonged use, especially at night.

• **Educational Tool Integration**: Future versions of ChatConnect can integrate with learning managementsystems(LMS),file-sharingtools,ordocumentcollaborationplatformstomakeita comprehensive academic communication hub.

• **EnhancedSecurityandPrivacySettings**:Offeringmoregranularprivacycontrols,suchas restricted group access or self-destructing messages, would increase user trust and protect sensitive information shared during group work or private discussions.

12. APPENDIX

SourceCode

1. MainActivity.kt

```kotlin
//filepath:app/src/main/java/com/example/myapplication/MainActivity.kt
```

```kotlin
class MainActivity : ComponentActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        FirebaseApp.initializeApp(this)
        setContent {
            NavComposeApp()
        }
    }
}
```

NavGraph.kt

```kotlin
//filepath:app/src/main/java/com/example/myapplication/nav/NavGraph.kt
@Composable
funNavGraph(navController:NavHostController){
    NavHost(navController,startDestination="login"){
        composable("login"){LoginScreen(navController)}
        composable("register"){RegisterScreen(navController)}
        composable("home"){HomeScreen(navController)}
        composable("chatroom/{roomId}"){backStackEntry->
            ChatRoomScreen(navController,
backStackEntry.arguments?.getString("roomId"))
        }
    }
}
```

3. LoginScreen.kt

```kotlin
//filepath:app/src/main/java/com/example/myapplication/view/LoginScreen.kt
@Composable
funLoginScreen(navController:NavController){
    //...existingcode...
    Button(onClick={/*Handlelogin*/}){ Text("Login")
    }
    //...existingcode...
}
```

4. RegisterScreen.kt

```kotlin
//filepath:app/src/main/java/com/example/myapplication/view/RegisterScreen.kt
@Composable
funRegisterScreen(navController:NavController){
    //...existingcode...
    Button(onClick={/*Handleregistration*/}){
        Text("Register")
    }
    //...existingcode...
}
```

5. HomeScreen.kt

```kotlin
//filepath:app/src/main/java/com/example/myapplication/view/HomeScreen.kt
@Composable
funHomeScreen(navController:NavController){
    //...existingcode... LazyColumn
    {
        items(chatRooms){room-
            >Text(room.name)
        }
    }
    //...existingcode...
}
```

6. ChatRoomScreen.kt

```kotlin
// filepath:
app/src/main/java/com/example/myapplication/view/ChatRoomScreen.kt
@Composable
funChatRoomScreen(navController:NavController,roomId:String?){
    //...existingcode... LazyColumn
    {
        items(messages){message->
```

```
            ChatMessageItem(message)
        }
    }
    //...existingcode...
}
```

7. ChatMessageItem.kt

```kotlin
// filepath:
app/src/main/java/com/example/myapplication/view/ChatMessageItem.kt
@Composable
funChatMessageItem(message:Message){
    Card {
        Text(text = message.text)
        Text(text=message.sender)
    }
}
```

8. Color.kt

```kotlin
//filepath:app/src/main/java/com/example/myapplication/ui/theme/Color.kt
val Purple200 = Color(0xFFBB86FC)
valPurple500=Color(0xFF6200EE)
valTeal200= Color(0xFF03DAC5)
```

9. Theme.kt

```kotlin
//filepath:app/src/main/java/com/example/myapplication/ui/theme/Theme.kt
@Composable
funChatoonTheme(content:@Composable()->Unit){
    MaterialTheme(
        colors = lightColors(),
        typography=Typography,
        shapes=Shapes, content
        = content
    )
}
```

10. Constants.kt

```kotlin
//filepath:app/src/main/java/com/example/myapplication/Constants.kt object
Constants {
    constvalUSERS_COLLECTION="users"
    constvalCHATROOMS_COLLECTION="chatrooms"
}
```