INFORMATICS & COMPUTATIONAL SCIENCES

MOHANLAL SUKHADIA UNIVERSITY, UDAIPUR

BACHELOR OF COMPUTER APPLICATIONS

(A choice Based Credit System)

B.C.A 6th Semester (2024-25)

Major Project Synopsis

On

"Conversa"

Submitted to:
Dr. Avinash Panwar

Submitted by:
Kavish Menaria

Project Title

Conversa *– Your space to speak!*

Project Logo

Group Members

- Kavish Menaria
- Khushal Dangi

Introduction of the Project:

Conversa is a real-time chat application developed for iOS using Swift. It provides users with an instant messaging platform that allows them to communicate in real time. The application uses Firebase Authentication for user sign-in and Firebase Firestore for efficient data storage and synchronization.

This chat application is designed with a simple and intuitive UI**,** making it easy for users to send and receive messages instantly. Firebase's cloud-based architecture ensures seamless message delivery and data synchronization across devices, eliminating the need for a traditional backend server.

Additionally, Conversa integrates Firebase Cloud Messaging (FCM) to send push notifications, ensuring that users are notified of new messages even when the app is in the background.

By leveraging Firebase, this project benefits from scalability, security, and real-time updates, making it an efficient solution for personal or professional communication. The NoSQL structure of Firebase Firestore allows for flexible data management, making message retrieval and updates faster compared to traditional relational databases.

Technology Stack Description:

The project is built using modern development tools and frameworks to ensure optimal performance and scalability.

Front-End:

- **SwiftUI:** Used for designing the user interface (UI).
- **Swift:** The programming language used for iOS app development.
- **SF Symbols:** Ensures a clean and modern UI.

Back-End (Cloud Services):

- **Firebase Authentication:** Provides a secure login and sign-up system.
- **Firebase Firestore:** A NoSQL cloud database that stores user profiles and chat messages.
- **Firebase Cloud Messaging (FCM):** Enables push notifications for new messages.

Development Tools:

- **X Code:** The official IDE for iOS app development.
- **Firebase Console:** Used for managing authentication, database, and cloud messaging services.
- **Git**: Used for version control and collaborative development

Advantages of Using Firebase:

The use of Firebase in Conversa provides multiple benefits over traditional backend solutions:

1. **Real-Time Data Synchronization:** Messages are instantly updated across all devices using Firebase Firestore.
2. **Scalability:** Firebase can handle multiple concurrent users without requiring dedicated backend servers.
3. **Security:** Firebase Authentication ensures user credentials are protected with encryption.
4. **Cloud-Based Storage:** Eliminates the need for local database management by using Firestore for storing chat data.
5. **Automatic Data Backup:** Data is stored in the cloud, preventing data loss.
6. **Efficient Push Notifications:** Firebase Cloud Messaging (FCM) ensures users receive real-time notifications for new messages.
7. **Reduced Development Effort:** Firebase handles authentication, database, and messaging services, reducing the need for backend code.

8. **Cost-Effective:** The free tier provides sufficient resources for a small-scale chat application, with pay-as-you-go pricing for larger projects.

Aim of the Project:

The aim of this project is to develop a real-time iOS chat application using **Swift** and **SwiftUI**, focusing on secure, responsive, and engaging communication between users. The key objectives include:

- Developing a personal chat system for one-on-one communication.
- Supporting image and audio messaging to enhance user interaction.
- Implementing Firebase Authentication for secure user registration and login.
- Using Firebase Firestore to store, sync, and retrieve chat messages in real-time.
- Enabling search functionality by username to easily find and connect with other users.
- Providing push notifications to alert users of new messages instantly.
- Building a profile view interface for user details and account management.
- Allowing users to delete older messages or clear entire chat history**.**
- Integrating block and unblock features to control unwanted communication.
- Ensuring data privacy and message security by leveraging Firebase rules and encryption.
- Designing an interactive and smooth UI using SwiftUI for modern, intuitive navigation.
- Maintaining low latency and high performance with Firebase's real-time sync.
- Supporting scalability to handle concurrent users and growing message volume efficiently

Project Features

- **User Authentication:** Secure user registration and login using Firebase Authentication with email/password or other supported providers.
- **Real-Time Messaging:** Instant delivery of personal messages using Firebase Firestore, with live updates across devices.
- **Media Sharing:** Supports sending and receiving images and audio messages, enhancing communication flexibility.
- **User Profiles:** Users can view and update their profile details including display name and profile picture.
- **Online/Offline Status:** Shows real-time user activity status to indicate if a contact is currently online.

- **Push Notifications:** Sends instant alerts when a new message is received, even when the app is in the background.
- **Search Functionality:** Allows users to search for other users by username or find previous messages within chats.
- **Chat Deletion and Clearing:** Users can delete specific messages or clear entire chat history for privacy and storage management.
- **Block/Unblock Contacts:** Provides the ability to block or unblock users to prevent unwanted communication.
- **Light & Dark Mode Support:** Enhances usability with a dynamic UI that supports both light and dark themes using SwiftUI.
- **Data Backup & Recovery:** All messages and media are backed up in Firestore, ensuring data persistence and recovery.
- **Smooth and Responsive UI:** Utilizes SwiftUI to deliver a modern, intuitive, and fluid user experience across iOS devices.
- **Scalability:** Built to handle a growing user base with scalable backend support from Firebase.

Project Modules

1. User Authentication Module

- Handles user registration, login, and logout using Firebase Authentication**.**
- Ensures password security and manages encrypted credentials via Firebase.

2. Chat Module

- Implements real-time personal chat using Firebase Firestore for instant messaging.
- Supports image and audio message sharing within chat conversations.
- Provides read receipts to track message delivery and seen status.
- Allows chat deletion, message clearing, and block/unblock functionality for better control and privacy.

3. Database Module

- Stores user profiles, chat messages, and media data using Firebase Firestore and Firebase Storage.
- Maintains chat history with efficient retrieval and organization.
- Enforces data integrity and access security using Firestore security rules.

4. UI/UX Module

- Designed with SwiftUI to deliver a clean, modern, and responsive interface.
- Includes profile view, emoji support, typing indicators, and online/offline status.
- Supports Light and Dark Mode, adaptive layouts, and smooth screen transitions.

5. Notification Module

- Integrates Firebase Cloud Messaging (FCM) to send push notifications for incoming messages.
- Displays badge icons for unread messages and handles background message alerts.
- Ensures timely delivery of notifications to improve user engagement.

6. Search & Profile Management Module

- Enables searching users by username for easy connectivity.
- Allows users to view and edit their profile, including profile picture and display name.
- Ensures profile updates are reflected instantly across the app.

Database Design

The chat application uses Firebase Firestore, a NoSQL cloud database, where data is structured into collections and documents instead of traditional relational database tables. The two primary collections used are Users and Messages, which are connected through UserID.

1. Users Table (Collection: users)

Field Name	Data Type	Description
Uid	String	Unique User ID (Firebase Auth UID)
Username	String	Unique username for search and display
email	String	Registered email address
profileImageUrl	String (URL)	Link to user's profile picture
isOnline	Boolean	True if user is currently online
lastSeen	Timestamp	Last active time
blockedUsers	Array[String]	List of user IDs that this user has blocked

2. Chats Table (Collection: chats)

Field Name	Data Type	Description
chatId	String	Combination of two user UIDs (e.g., uid1_uid2)
participants	Array[String]	Two UIDs involved in the conversation
lastMessage	String	Content of the last message sent
lastMessageTime	Timestamp	Time when last message was sent
isCleared	Map	Tracks if each user has cleared the chat

3. Messages Table (Subcollection: chats/{chatId}/messages)

Field Name	Data Type	Description
messageId	String	Auto-generated document ID
senderId	String	UID of the sender
receiverId	String	UID of the receiver
timestamp	Timestamp	Time when the message was sent
messageType	String	text, image, or audio
message	String	Text message or caption
mediaUrl	String (URL)	Link to media if type is image/audio
isDeleted	Boolean	True if the message is deleted (soft delete)

4. Device Tokens Table (Collection: deviceTokens)

Field Name	Data Type	Description
uid	String	User ID
deviceToken	String	FCM token used for push notifications

References:

1. developer.apple.com
2. developer.apple.com
3. firebase.google.com
4. swift.org