

## REAL TIME SYSTEM: QuickTalk

### SOFTWARE ENGINEERING PROJECT CSC 335

#### Presented by Group 5:

- |                                |        |
|--------------------------------|--------|
| 1. CHINEDU, PROMISE OKAFOR     | 213930 |
| 2. OLATUNJI, MICHAEL OLUWAYEMI | 214903 |
| 3. OGUNESAN, RHODA OLUWATOSIN  | 214897 |
| 4. KAYODE, PETER TEMITOPE      | 208077 |
| 5. OKAFOR, LISA CHISOM         | 214901 |

## Table of Contents

<b>Introduction .....</b>	<b>6</b>
What are Real-Time System? .....	6
Hard Real-Time System .....	6
Soft Real-Time System .....	6
<b>Planning.....</b>	<b>7</b>
Introduction .....	7
Purpose of the System .....	7
Objective of the System .....	7
Scope of the System .....	9
Intended Audience.....	11
Management of the System .....	11
Project Charter.....	12
Work Breakdown Structure .....	13
Project Work Plan .....	14
Gantt chart .....	15
<b>Requirement Analysis.....</b>	<b>16</b>
Functional Requirements.....	16
Non-Functional Requirements.....	16
Architectural Requirements .....	19
<b>Feasibility Report.....</b>	<b>20</b>
Introduction .....	20
Study Approach.....	20

Project Description .....	20
Feasibility Analysis .....	20
<b>Data Model .....</b>	<b>25</b>
Class Diagram .....	25
Context Level Data Flow Diagram.....	26
Use Case Diagram .....	27
<b>Database Design .....</b>	<b>28</b>
Users.....	29
Translation Request .....	29
Translation History .....	30
Language Data.....	30
<b>User Interface Design .....</b>	<b>32</b>
1. The Splash Screen .....	32
2. Create Account .....	33
3. Log In .....	34
4. Home .....	35
5. Account.....	36
6. History .....	37
7. Logout.....	38
<b>Testing.....</b>	<b>40</b>
Unit Testing.....	40
Integration Testing .....	41

Functional Testing.....	41
Performance Testing.....	42
Compatibility Testing.....	42
<b>Maintenance .....</b>	<b>43</b>
Bug Fixes.....	43
Updates and Enhancements .....	43
Patching and Security Updates .....	43
Performance Optimization.....	44
Compatibility and Integration.....	44
Documentation Maintenance .....	44

# Introduction

## What are Real-Time System?

Real-time systems are computer systems that are designed to respond to events or input within a specified time constraint. These systems are often used in applications where timely and predictable responses are required, such as industrial control systems, aerospace and defense systems, medical devices, and multimedia applications. The defining characteristic of real-time systems is their ability to meet deadlines or time constraints. They are classified into two broad categories:

### Hard Real-Time System

In hard real-time systems, meeting the specified deadlines is critical. If a deadline is missed, it can lead to catastrophic consequences, such as system failure or loss of life. Examples include air traffic control systems, medical life support systems, and automotive safety systems.

### Soft Real-Time System

Soft real-time systems also have timing constraints, but missing occasional deadlines does not result in system failure or severe consequences. These systems can tolerate some degree of deadline violation, but performance degradation may occur. Examples include multimedia streaming, online gaming, and virtual reality applications.

With this in mind, as a partial requirement in passing the course, CSC 335 (Software Engineering), we present this project to the lecturer in charge, Dr. Angela Makolo.

# Planning

## Introduction

This project involves the development of Language translator mobile application which is a real time system responsible for interpreting voices in different languages, translating the voices into different languages, transcribing of words, translating words of various languages into desired languages.

## Purpose of the System

The system is developed to ease the mode of communication between people of different languages, thereby breaking language barrier while communicating. It facilitates efficient communication without any form of misinterpretation as fast as possible while the speaker is speaking. The availability of this service to users would help to foster communication with ease and mobility.

## Objective of the System

### 1. Enable Cross-Language Communication

The primary objective of the application is to facilitate communication between users who speak different languages. It allows users to engage in real-time conversations, overcoming language barriers through automatic translation.

### 2. Real-Time Translation

The application should provide accurate and fast real-time translation of messages exchanged between users. The translation should be seamless and enable smooth communication without significant delays.

### **3. Multilingual Support**

The application should support multiple languages to cater to a wide range of users. It should have a diverse language selection, allowing users to communicate in their preferred language.

### **4. User-Friendly Interface**

The application should have a user-friendly interface that is intuitive and easy to navigate. Users should be able to quickly understand how to use the chat features and access translation options effortlessly.

### **5. Accurate and Reliable Translation**

The translation feature should ensure high accuracy and reliability. It should effectively capture the intended meaning of the messages while maintaining context and tone, providing users with a clear understanding of each other's messages.

### **6. Customizable Translation Settings**

The application should offer customizable translation settings to cater to individual preferences. Users should have the ability to adjust translation accuracy levels, select preferred languages, and personalize the translation experience to suit their needs.

### **7. Privacy and Security**

The application should prioritize user privacy and data security. It should employ robust encryption measures to protect user conversations and sensitive information, ensuring a secure and private communication environment.

### **8. Cross-Platform Compatibility**

The application should be compatible with various devices and platforms, such as mobile phones, tablets, and desktop computers. This enables users to access the chat application conveniently from different devices.



## 9. Error Handling and Feedback

The application should handle translation errors gracefully and provide appropriate feedback to users. In cases where the translation may not be accurate or clear, users should be alerted and provided with alternative options or suggestions.

## 10. Continuous Improvement

The objective is to continually enhance the translation capabilities and overall performance of the application. Feedback from users should be collected and used to refine the translation algorithms, improve language support, and address any identified issues or limitations.

By achieving these objectives, a real-time language translation chat application can effectively bridge language gaps, promote cross-cultural understanding, and facilitate meaningful communication between individuals who speak different languages.

## Scope of the System

The system will be able to accept input such as either audio, text or both, from users in different languages, and then simultaneously translate the input into the user desired language audio or text continually. The system is built to be a real time system because while the input is being accepted from the user, translation into other desired language simultaneously takes place, given the desired result.

<b>Student Record Management System</b> <i>Project Scope Statement</i>		Prepared by: Lisa Okafor Date: 09 -08-2023
<b>General Project Information</b>		
<b>Project Name:</b>	Real Time Chat System	
<b>Sponsor:</b>	Computer Science Department, University of Ibadan	
<b>Project Manager:</b>	Lisa Okafor	

**Problem/Opportunity Statement**

The real-time chat system is a web-based application that allows users from different nations to communicate with each other through text messages in real-time. The goal of the chat system is to create a seamless and intuitive user experience for individuals and groups to communicate in real-time without the barrier of language.

**Project Objectives:**

The system will allow users to create accounts, find and connect with other users, join and participate in chat rooms, and exchange messages with one another.

**Project Description:**

A real-time language translation chat application: An application that allows users who speak different languages to communicate in real-time using automatic translation. Privacy and authority access and also enable students to interact with the system.

**Business Benefits:**

Reduced cost of manual record of data.  
Reduced delay in getting promotion, recruitment and leave.  
Improved decision making.

**Project Deliverables:**

Real-time chat system analysis and design  
Real-time chat system programs.  
Real-time chat system documentation.  
Training procedures.

**Estimated Project Duration:**

8 weeks

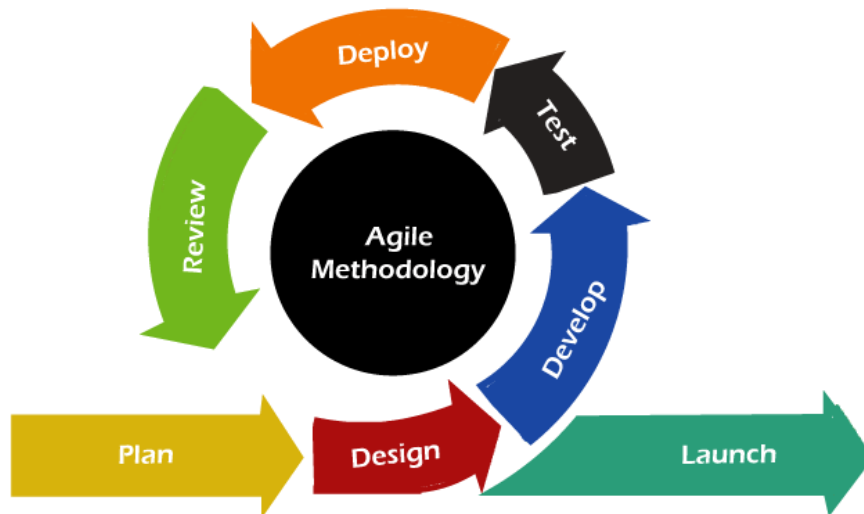
## Intended Audience

This system is aimed towards everyone who has basic knowledge of mobile phone operation, people who are languages students, people who interact with people of different ethnic groups/languages, people who communicate more often with aged people.

## Management of the System

The creation and management of the system right from when it is in the planning stages up to when it reaches the testing stage can be managed using the SDLC, the SDLC is process used to design, develop and test software. It has various models, and the Agile method will be employed for this system.

The Agile methodology involves breaking the project into phases and emphasizes continuous collaboration and improvement. The team follows a cycle of planning, executing, and evaluating.



## Project Charter

### *Project Charter*

Prepared: April 03, 2023

**Project Name:** Real Time System

**Customer:** MetaVerse

**Project Sponsor:** Dr Angela Makolo

### **Project Overview:**

This project will implement a real-time language translation chat application: An application that allows users who speak different languages to communicate in real-time using automatic translation. This type of chat application is important for cross-cultural communication and improving understanding between people who speak different languages.

### **Objectives:**

- Enable cross-language communication
- Real-time translation
- User-friendly chat interface
- Accurate and reliable translation

### **Stakeholders and Responsibilities**

#### **Stakeholders Responsibility**

#### **Role**

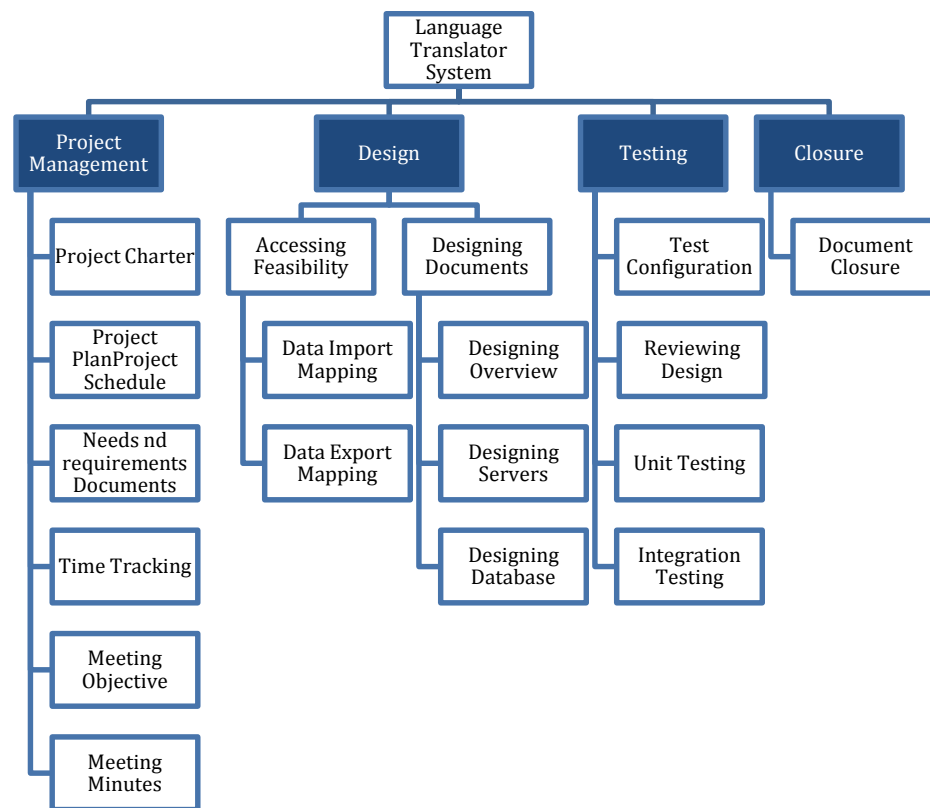
Dr. Angela Makolo

VP

Project

Vision,

## Work Breakdown Structure



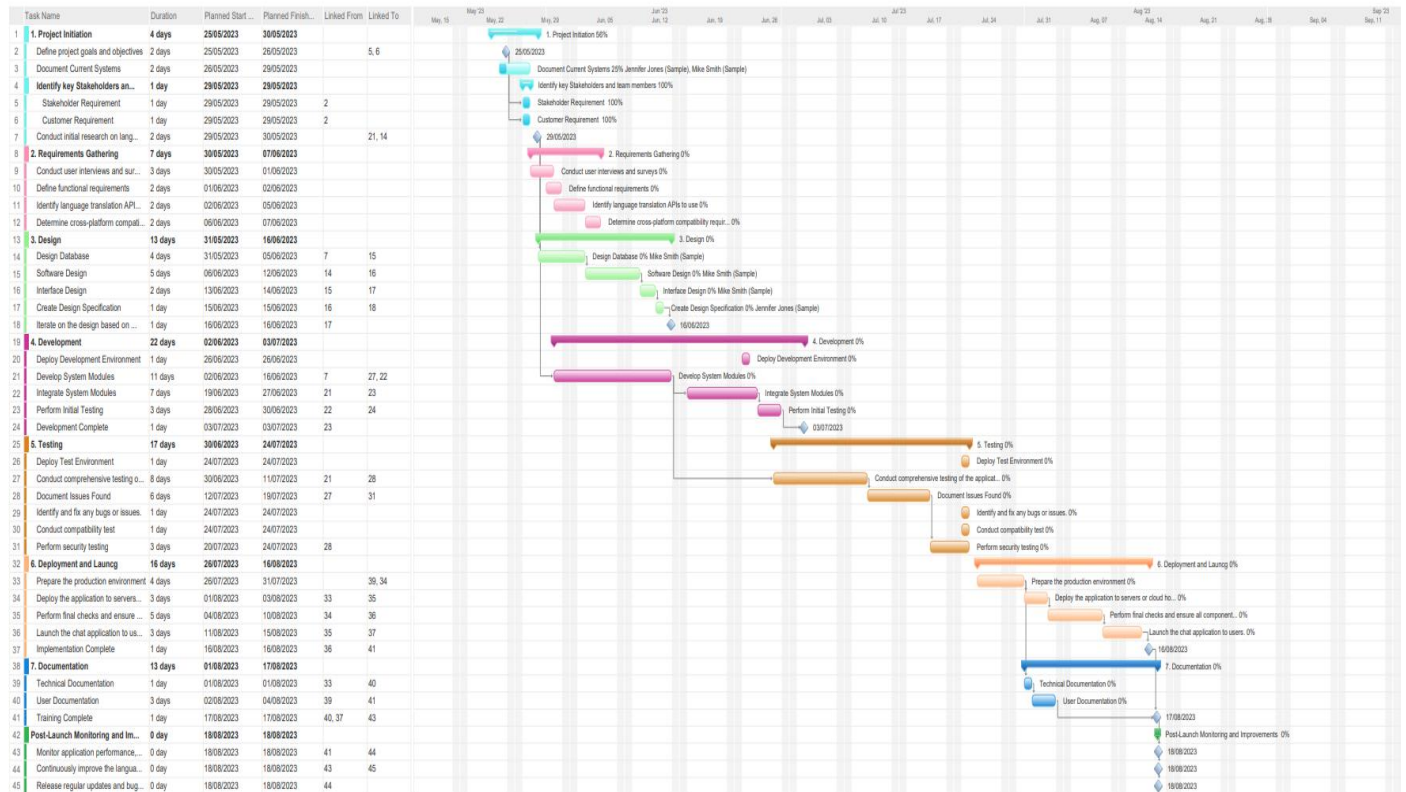
## Project Work Plan

S/ N	TASK NAME	DURATI ON (Days)	START	FINISH	CRITICAL PATH	PREDECESSOR
1	REQUIREMENTS COLLECTION	6	01\04\2023	07\04\2023	✓	-
2	SYSTEM ANALYSIS	6	07\04\2023	13\04\2023	✓	1
3	SYSTEM DESIGN	6	13\04\2023	19\04\2023	✓	2
4	USER DOCUMENTATION	13	19\04\2023	02\05\2023		3
5	CODING	8	02\05\2023	10\05\2023	✓	3
6	IMPLEMENTATION	5	10\05\2023	15\05\2023	✓	5
7	TESTING	2	15\05\2023	17\05\2023	✓	6
8	INSTALLATION	1	17\05\2023	18\05\2023	✓	7
9	TRAINING OF USERS	1	18\05\2023	19\05\2023	✓	4,8

## Gantt chart

## New Project - 20 May 2023

Page 1



# Requirement Analysis

The below requirements of this system to be developed were gathered using previously existing real time language translation applications such as Google translate, Hi Translate, All Language translate. Interviewing of potential customers were also done to gather all their requirements, and hence, the requirements were compiled and now documented below.

## Functional Requirements

1. The system allows users to register and create account.
2. The system allows users to log in and log out of the application for security purposes.
3. As a real time system, the QuickTalk allows for typing texts and audio, and getting them translated simultaneously.
4. QuickTalk allows user to see list of previously translated texts or audio.

## Non-Functional Requirements

### 1. Performance

- QuickTalk would be able to handle a minimum of 1000 concurrent users at any given time.
- QuickTalk would be have a maximum response time of 0.5 second for sending and receiving messages.
- The performance of the Application would be determined by it responsive time, time to complete the given task.
- When QuickTalk is made to start up it would not take more than 3 second to load initial screen.



## 2. Scalability

- Quick Talk would be able to adopt itself to increased usage or able to handle more data as time progress.
- When the user data(caches , stored data etc.) increases, the app would be capable of handling them without delay by optimizing the way storage is done and accessed

## 3. Responsiveness

QuickTalk would be responsive to the user Input or to any external interrupt which is of highest priority and return back to same state. I.e. When it gets interrupted by call, then app would be able to save state and return to same state/ page which was there before it got interrupted. User would be able to understand the flow of App easily i.e. Users would be able to use App without any guideline or help from experts/manuals. QuickTalk would have a proper and easy user interface

## 4. Usability

User would be able to understand the flow of App easily i.e. Users would be able to use App without any guideline or help from experts/manuals. QuickTalk would have a proper and easy user interface

## 5. Reliability

The application should be reliable to perform the business, i.e. when user perform some important action it should be acknowledged with confirmation.

## 6. Security

- The system will use HTTPS for secure communication between clients and the server.
- User passwords will be hashed and salted before being stored in the database.
- User sessions will be managed using JWT tokens to prevent unauthorized access to user accounts.
- QuickTalk will have rate limiting to prevent brute force attacks.

- All the app data should be secured and be encrypted with minimum needs so that it's protected from outside environment also from internal attack.
- All authentication token would be saved on local device for comparison and need user permission to gain access. The application should offer customizable translation settings to cater to individual preferences. Users should have the ability to adjust translation accuracy levels select preferred languages, and personalize the translation experience to suit their needs.

## 7. Screen Adaptation

Since now a days lot of mobile devices comes with different screen sizes and layout, Quick Talk would be able to render it's layout to different screen sizes. Along with automatic adjustment of Font size and image rendering.

## 8. Network Coverage

QuikTalk would work well with Wi-Fi but would be able to handle slow connection while experience Wi-Fi black spots or when connected to mobile Network. The app would be able to look out for WiFi if not available then automatically switch to mobile network.

## 9. Accessibility

QuickTalk would be made available 24/7 with a maximum downtime of 5 minutes per month.

## Architectural Requirements

- QuickTalk will be built using a client-server architecture.
- The server will be responsible for managing user accounts, translating messages and audios, storing messages and audios, and handling requests from clients.
- The clients will be responsible for displaying the interface to the users, sending messages and audios to the server, and receiving messages from the server.
- QuickTalk will use a RESTful API for communication between the clients and server.
- QuickTalk will use web sockets for real-time communication between clients and the server.

# Feasibility Report

## Introduction

The purpose of this feasibility study is to assess the viability of developing and implementing a language translating application. This report provides an analysis of the technical, economic, operational, schedule, and legal feasibility of the proposed chatting application.

## Study Approach

We conducted interviews with stakeholders, performed market research on existing chatting applications, and consulted with technical experts to gather relevant information for the feasibility study.

## Project Description

The proposed language translating application aims to facilitate real-time communication and collaboration among employees within our organization. The application will provide features such as instant messaging, file sharing, audio interpretation, and integration with existing tools.

## Feasibility Analysis

### 1. Technical Feasibility

The key functionalities and features real-time messaging, group chats, multimedia sharing, translation, notifications, user authentication, and integration with other systems or platforms. QuickTalk application will target mobile (iOS, Android). Having considered factors such as developer expertise, scalability, performance, and compatibility with the chosen platform(s), the programming languages to be used include JavaScript (with frameworks like React or Angular), Swift, Kotlin, or Python.

The database requirements for storing user profiles, chat histories, and other relevant data could either be MySQL or MongoDB. The technology or protocol required for real-time communication within this application could include WebSockets, WebRTC, or third-party communication APIs (e.g., Pusher, Socket.IO). For scalability and performance, techniques like load balancing, caching, and horizontal scaling would be considered. The design principles would be of responsive layouts, and interactive elements to create an intuitive and user-friendly interface. A cloud-based services (e.g., AWS, Google Cloud) or on-premises infrastructure could be used.

Having considered all these technical requirements, the project is due feasible in the technical aspect.

## 2. Economic Feasibility

### 1. Cost Estimation

**Software Development Costs:** Based on consultations with development teams and vendors, the estimated cost for developing the translating application is N2500000, which includes frontend and backend development, UI/UX design, and testing.

**Infrastructure Setup Costs:** The estimated cost for setting up the necessary servers, databases, and networking infrastructure is N1500000. This includes hardware, software licenses, and cloud services.

**Licensing Fees:** To integrate certain third-party tools and services, licensing fees amounting to N500000 per year will be incurred.

**Additional Expenses:** Other hardware or equipment requirements, such as cameras or microphones for video chat features, are estimated to cost N500000.

## 2. Return on Investment (ROI) Analysis

The initial investment for developing and launching the chatting application is estimated at N20000000.

Based on revenue projections and estimated costs, we anticipate achieving breakeven within 24 months.

The projected ROI over a period of 3 years is estimated at 300%, indicating a positive return on the investment made.

## 3. Cost-Benefit Analysis

The implementation of the language translating application offers numerous benefits, including improved communication, enhanced collaboration, and increased productivity.

Quantifiable benefits include estimated time savings of 8 hours per week for employees, leading to the cost-benefit analysis demonstrates that the anticipated benefits outweigh the costs associated with developing and operating the application.


## 4. Risk Assessment

Identified risks include potential competition from existing language translating applications, evolving user preferences, and technological advancements impacting the market demand.

Mitigation strategies, such as continuous product enhancements, marketing campaigns, and strategic partnerships, have been identified to minimize these risks.

## 5. Financial Projections

Financial projections indicate that the chatting application will generate total revenue of N30000000 in the first year, with an average annual growth rate of 25% over the subsequent 5 years. Projected costs and expenses are estimated to increase in line with revenue growth, ensuring sustainable financial performance.



## 6. Financial Viability Assessment

Based on the comprehensive economic analysis, the language translating application is deemed financially viable and promising. Key financial metrics, such as the payback period, NPV, and IRR, meet or exceed the predetermined success criteria, demonstrating the economic feasibility of the project.

## 3. Operational Feasibility

As a result of the project team interviews with key stakeholders, such as project managers, team members, and executives, we came to understand their current language translation techniques and process. They identify that the existing process relies heavily on manual interpretation from a hired language interpreter which often leads to miscommunication, delays, and difficulties in tracking progress.

Based on this information, we define the operational objectives of the new application, such as improving efficiency in communication, improving real-time collaboration, streamlining task assignment, and providing centralized project documentation. Additionally, we will evaluate the user adoption challenges, conduct training sessions, and establish a change management plan to ensure a smooth transition to the new application. Through continuous improvement efforts and feedback mechanisms, we will iteratively enhance the application based on user feedback, addressing operational challenges and improving overall project management efficiency.

## 4. Schedule Feasibility

Having to carefully and meticulously define the Project Milestones, estimate development Effort, identify dependencies, check out resource allocations, consider risk assessment and mitigation, prioritize features and scope, consider the development methodology, and would ensure regular monitoring and reporting, the project is schedule feasibility is on the positive side.

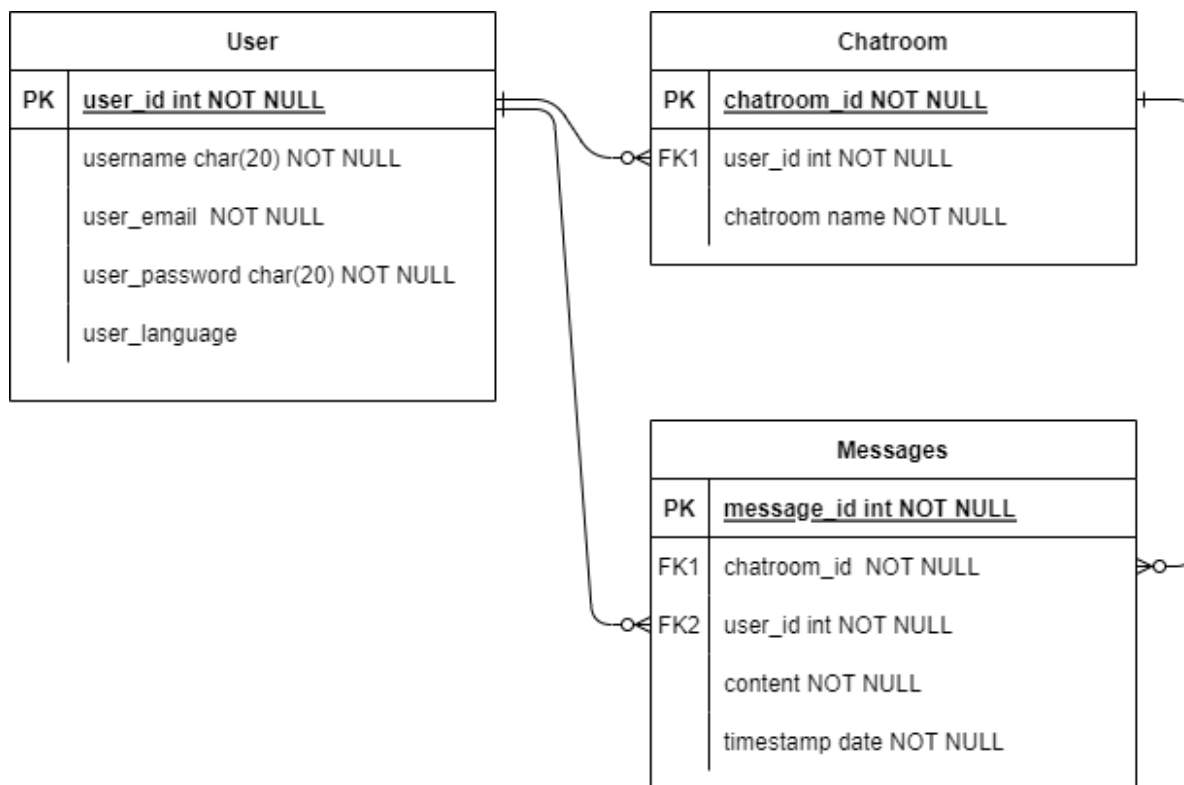
## 5. Legal Feasibility

Having carefully Identified applicable Laws and regulations, evaluated Data Protection and Privacy, considered Intellectual Property Rights, analyzed Electronic Communications and Security, assessed Compliance with Industry-Specific Regulations, considered Legal Liability and Terms of Use, stayed updated with legal changes, and consulted Legal Experts, it is deemed fit that QuickTalk is positive in legal feasibility.

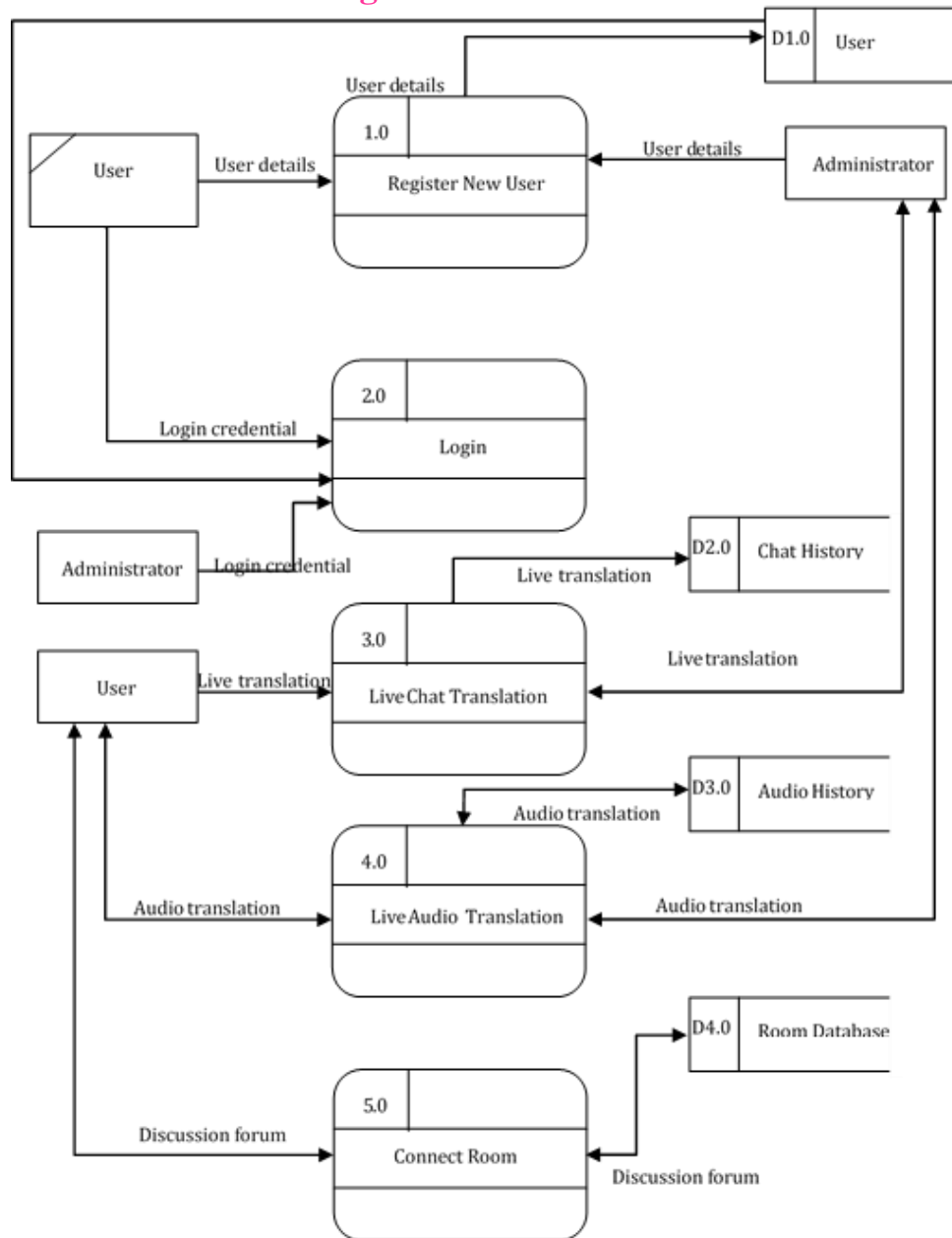


# Data Model

## Class Diagram



## Context Level Data Flow Diagram



## Use Case Diagram



# Database Design

This documentation offers a comprehensive outline of the database structure and features of QuickTalk. This application is specifically developed to facilitate the translation of text or speech between different languages, empowering users to effectively communicate despite language barriers. The database assumes a vital role in the storage and administration of essential data required for seamless translation operations.

QuickTalk leverages a relational database management system (RDBMS) to efficiently store and structure its data. The choice of RDBMS may vary depending on implementation preferences, including popular options like MySQL, PostgreSQL, or Microsoft SQL Server. However, for this project we used MySQL due to the following reasons.

- Ease of use
- High level of security
- High performance
- MySQL is open source
- Data security
- High Efficiency

QuickTalk's database consists of several tables that store different types of data required for translation. The following sections describe each table and its associated columns:

## Users

The “Users” table stores information about the app's users, including their unique identifiers, authentication credentials, and user preferences. It helps to manage user accounts and personalize the translation experience, providing a seamless experience to each user.

The Columns present in the “Users” table include:

- ``user_id``: The ``user_id`` is a *primary key*, which is a unique identifier for each user.
- ``username``: This represents each user's username.
- ``password``: To ensure secure authentication, the app encrypts the passwords of each user, which is represented in the ``password`` column.
- ``email``: This stores each user's email address.
- ``preferences``: In order to provide a seamless and personalized experience to each user, we provide a means of storing each user's preferences for language settings and other customizations.

## Translation Request

The “Translation Requests” table maintains a log of translation requests made by users. Each request encompasses the source text, source language, target language, and any supplementary details pertinent to the translation process. The Columns present in the “Translation Requests” table include:

- ``request_id``: A unique identifier is assigned to each translation request as its primary key tagged as the ``request_id``.
- ``user_id``: This is a *foreign key* referencing the user who made the request.
- ``source_text``: The text the user desires to translate.
- ``source_language``: The language of the source text.
- ``target_language``: The desired language for the translation.
- ``additional_details``: This includes optional additional information or instructions provided by the user.

## Translation History

The “Translation History” table preserves a comprehensive log of completed translations, including the original source text, translated text, and pertinent details. This functionality empowers users to conveniently revisit and review their previous translation when desired.

The Columns present in the “Translation History” table include:

- `translation\_id`: A unique identifier for each translation which serves as a primary key.
- `request\_id`: A foreign key referencing the original translation request.
- `source\_text`: The original text that was translated.
- `translated\_text`: The resulting translated text.
- `source\_language`: The language of the original text.
- `target\_language`: The language of the translation.
- `timestamp`: Timestamp of the translation completion

## Language Data

The "Language Data" table stores relevant information regarding the supported languages within the app. It comprises language codes, names, and other pertinent details necessary for language selection and facilitating accurate translations.

The “Language Data” table contains the following columns:

- `language\_id`: A unique identifier for each language - `language\_id` is a primary key.
- `language\_code`: It represents the language code for a particular language according to standard language code conventions.
- `language\_name`: It represents the name of the language.
- `is\_supported`: A flag indicating if the language is currently supported in the app.

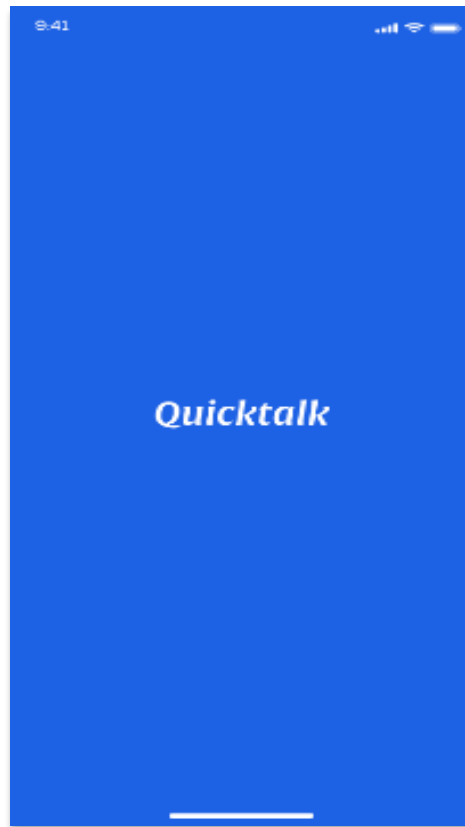
This documentation provides an overview of QuickTalk's database structure and functionality. It covers essential tables for managing user information, translation requests, history, and language data. Understanding this structure helps enhance the app's translation capabilities efficiently, and also enables developers and maintainers of the app to successfully manage and enhance the app's translation capabilities proficiently.

# User Interface Design

The interface design has 7 flows to express the functionalities of the software.

## 1. The Splash Screen

To introduce users the app, the splash screen was designed. A re-assuring screen was added – showing the benefits of having the app. Attached to this screen are buttons to create account and join back if account had been created before.

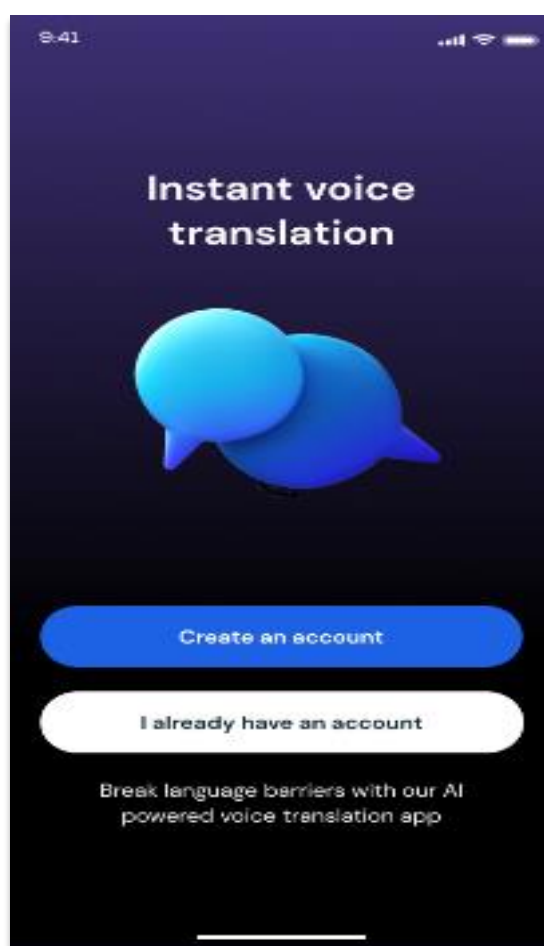




## 2. Create Account

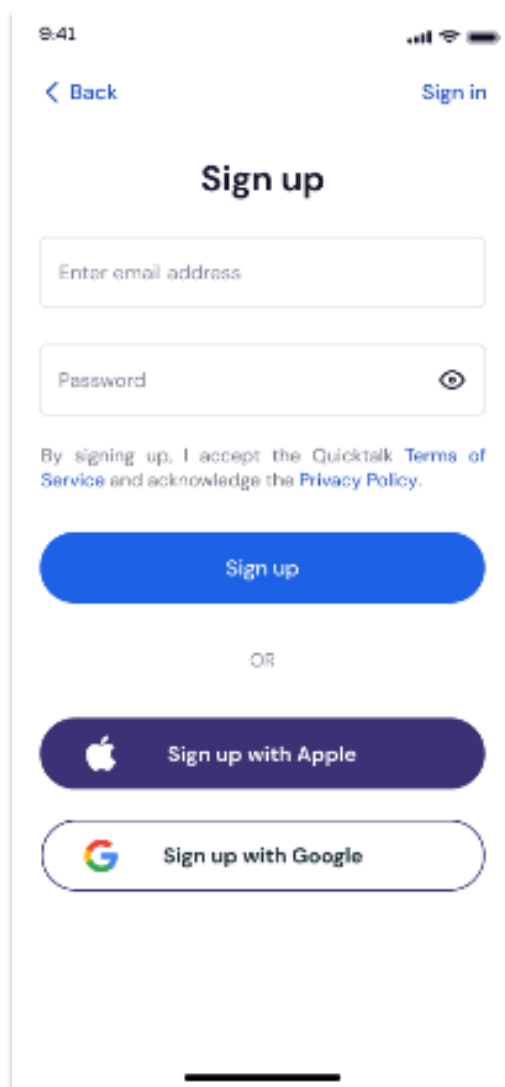
The second flow is to create account. It has 3 different options to allow flexibility.

The user can choose to create with email address and password or with a Google account or with an Apple account. Attached to this screen are the Privacy Policy and Terms of Service.



### 3. Log In

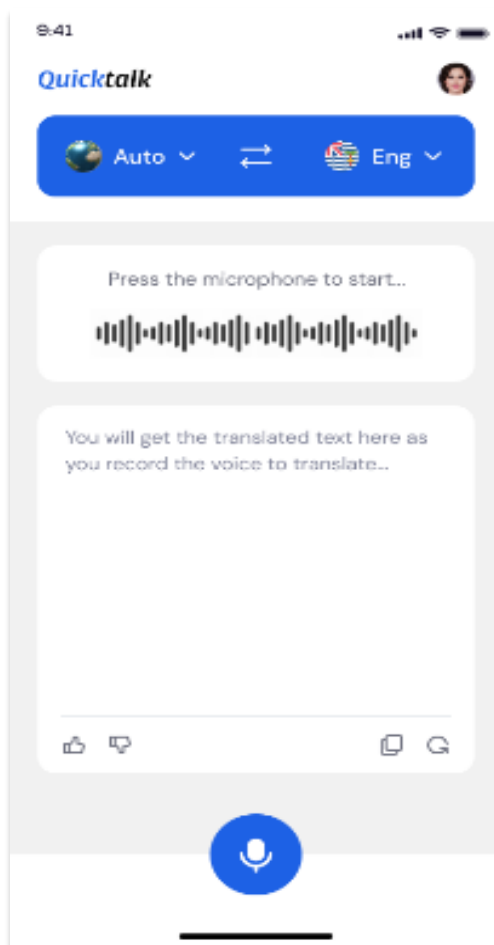
This flow is for registered users. They can log in with the method used to create account initially.



A mobile app sign-up screen with a white background. At the top, the status bar shows the time 9:41, signal strength, Wi-Fi, and battery. Below the status bar, there is a blue '< Back' link on the left and a blue 'Sign in' link on the right. The main heading 'Sign up' is centered in bold black text. Below the heading, there are two input fields: 'Enter email address' and 'Password'. The password field has a toggle icon (an eye) on the right. Below the input fields, there is a line of text: 'By signing up, I accept the Quicktalk Terms of Service and acknowledge the Privacy Policy.' Below this text is a large blue button with the text 'Sign up'. Below the button is the word 'OR' in a small, gray font. Below 'OR' are two more buttons: a dark blue button with the Apple logo and the text 'Sign up with Apple', and a white button with a gray border, the Google logo, and the text 'Sign up with Google'. At the bottom of the screen, there is a black horizontal line representing the home indicator.

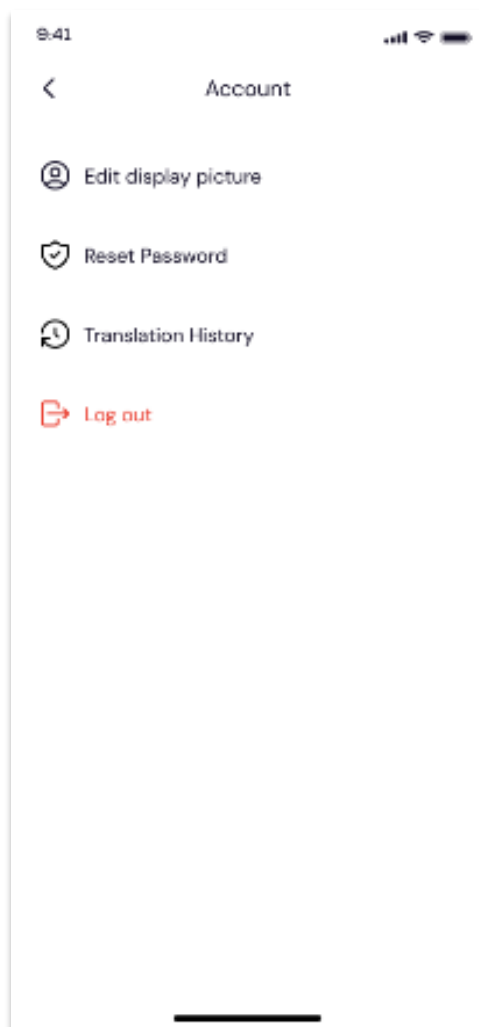
## 4. Home

This flow is to show the functionalities of the buttons, icons and picture on the screen. From this screen, the users can explore the app by recording voice for translation, refresh the translation, copy the translation, like and dislike, pause and play, access account too.



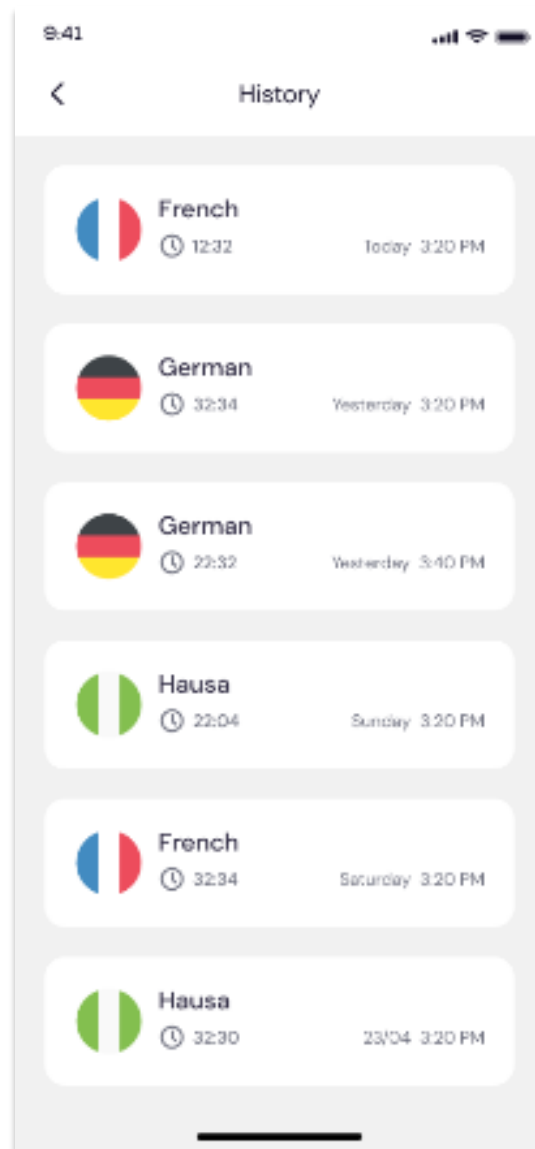
## 5. Account

The flow is to allow customization of app by adding, editing or removing display picture; reset password. Users can also access translation history and log out from this screen.



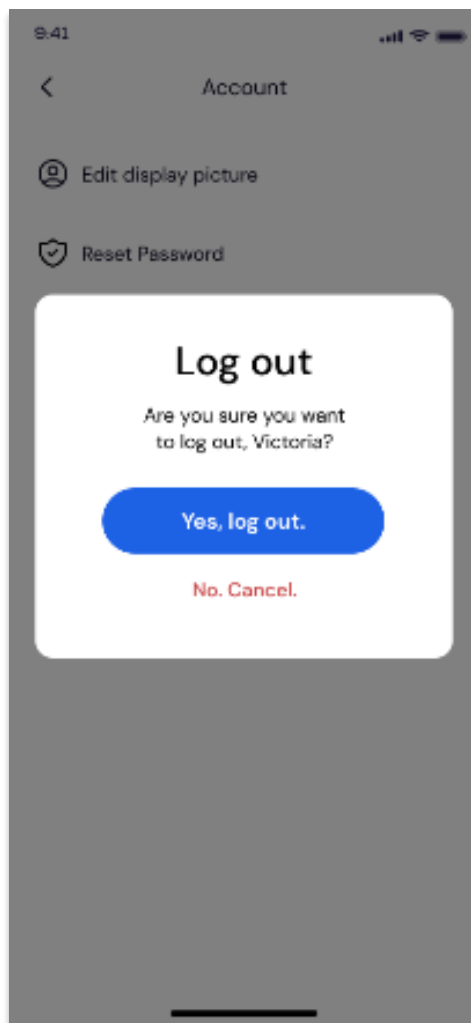
## 6. History

This flow is from Account. It can be used to access previous translations.

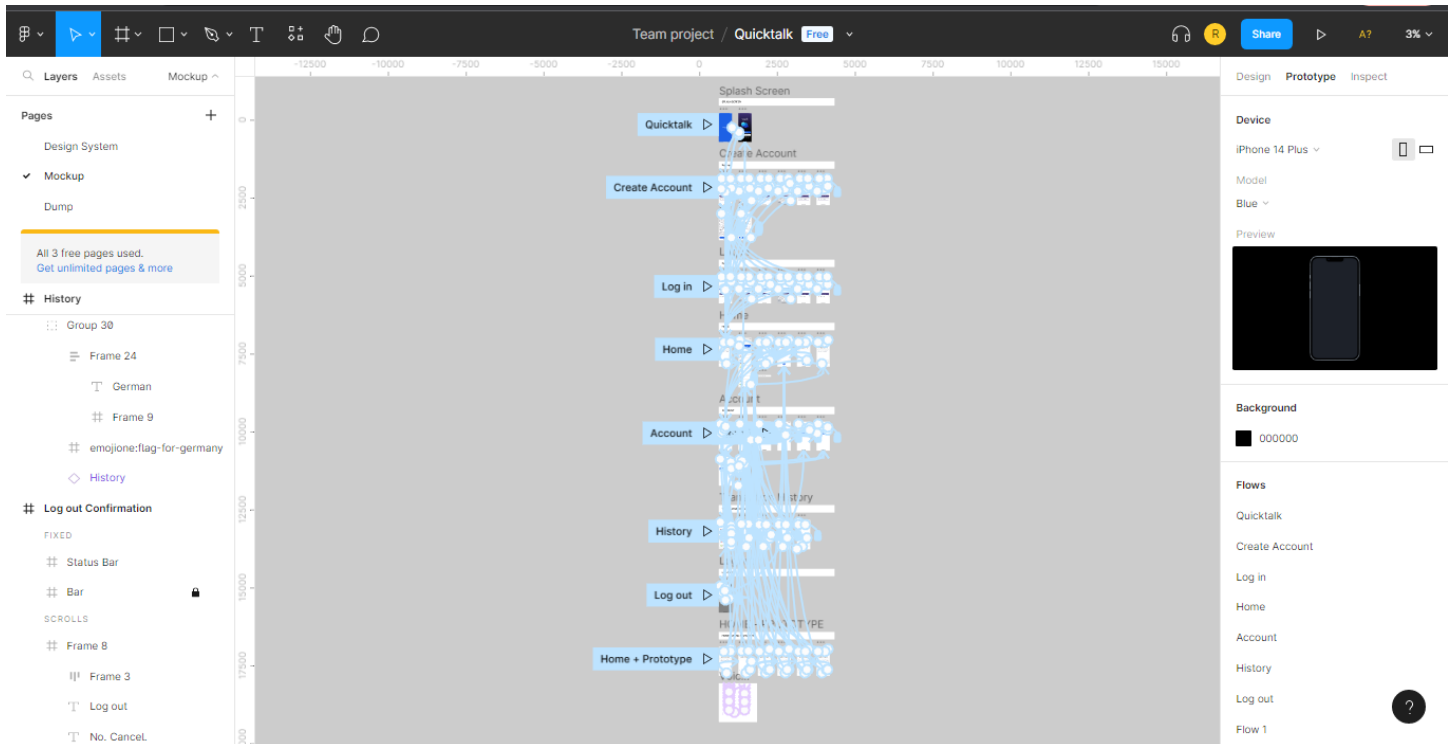


## 7. Logout

This is the last flow. It is for the users to leave the app without a mistake.



The prototype of the designs was done to show how interactive the app is, for the stakeholders to better and easily understand the software and for easy testing.



[Click here to inspect the Working Prototype](#)

# Testing

The purpose of this stage is to identify defects, errors, or issues in the software and ensure that it meets the specified requirements and quality standards before it is released to users. Importantly, this testing stage involved multiple iterations and cycles, since we used the agile development environment. Continuous testing and feedback loops were implemented to ensure that issues were identified early and addressed promptly. By conducting thorough testing during this stage, we enhanced the reliability, performance, and overall quality of the software before it is released to end-users.

## Unit Testing

### Test Case 1: Voice Recognition

**Input:** Audio file containing spoken words in English.

**Expected Result:** Accurate recognition of English words

### Test Case 2: Translation Algorithm

**Input:** English text "Hello, how are you?"

**Expected Result:** Translated text in French "Bonjour, comment ça va ?"

### Test Case 3: Text-to-Speech Conversion

**Input:** Translated text in Spanish "Hola, ¿cómo estás?"

**Expected Result:** Audio output with accurate Spanish pronunciation



## Integration Testing

### Test Case 4: Voice Interpretation and Translation

**Input:** Live voice input in Mandarin Chinese

**Expected Result:** Accurate interpretation and translation into English

### Test Case 5: Translation and Transcription Integration

**Input:** Live voice input in German

**Expected Result:** Accurate translation to English and transcription of German words

## Functional Testing

### Test Case 6: Language Translation Accuracy

**Input:** English text "I love pizza."

**Expected Result:** Translated text in Italian "Amo la pizza."

### Test Case 7: Word Transcription Accuracy

**Input:** Spoken word "Bonjour" in French

**Expected Result:** Accurate transcription of the word in written form

## Performance Testing

### Test Case 8: Response Time

**Input:** Live voice input in Spanish

**Expected Result:** Quick translation and response within a few

### Test Case 9: Resource Usage

**Input:** Continuous translation and transcription for 10 minutes

**Expected Result:** Stable performance without excessive battery drain or memory

## Compatibility Testing

### Test Case 10: iOS Compatibility

**Input:** Same test cases as above on an iOS device

**Expected Result:** Consistent functionality and performance on

### Test Case 11: Android Compatibility

**Input:** Same test cases as above on an Android device

**Expected Result:** Consistent functionality and performance on Android

# Maintenance

The maintenance phase would occur after the software has been deployed and is in use by the end-users. It involves managing and addressing issues that arise during the software's operational life cycle.

## Bug Fixes

As users interact with the software, they may encounter bugs or defects that need to be addressed. During the maintenance phase, we will analyze and fix reported bugs to ensure the software functions correctly.

## Updates and Enhancements

QuickTalk may require updates or enhancements to introduce new features, improve performance, or address changing user needs. These changes will be typically planned based on user feedback, market demands, or organizational requirements.

## Patching and Security Updates

Security vulnerabilities or critical issues may be discovered in QuickTalk. It's important that we will release patches or updates to address these vulnerabilities and ensure that QuickTalk remains secure.

## **Performance Optimization**

Over time, QuickTalk's performance may degrade due to increased data volume, changes in the operating environment, or inefficient algorithms. Performance optimization activities aim to improve the software's speed, efficiency, and resource utilization.

## **Compatibility and Integration**

As new technologies or platforms emerge, QuickTalk may need to be updated to remain compatible or integrated with other systems or devices. This ensures that the software can continue to function seamlessly in the evolving technological landscape.

## **Documentation Maintenance**

Documentation related to the QuickTalk, such as user manuals, technical guides, or system documentation, may need updates to reflect any changes or enhancements made.