Bilkent University

Department of Computer Engineering

Senior Design Project

Low-Level Design Report

Project: Signify

Team Members: Ali Taha Dinçer, Çağlar Çankaya, İrem Ecem Yelkanat, Muhammed Naci Dalkıran, Sena Korkut

Supervisor: Ayşegül Dündar

Jury Members: Erhan Dolak and Tağmaç Topal

This report is submitted to the Department of Computer Engineering of Bilkent University in partial fulfillment of the requirements of the Senior Design Project course CS491/2.

1. Introduction

In society, people who are hearing impaired and/or speech impaired have difficulty expressing themselves and communicating with other people because most people lack knowledge of sign language. Even though the improvements in technology have changed the way people live and made the lives of people easier by, for example, transforming mobile phones from sound devices into multi-functional devices, communicating with people who are hearing impaired and/or speech impaired continues to be a problem in many areas including social and technical contexts. These communication activities include social life, healthcare, career development, and education. Furthermore, as a result of the Covid19 pandemic that started in late 2019, obligations regulated by most countries, including wearing face masks and social and physical distancing, have increased the communication and social challenges for hearing impaired people. For example, wearing face masks has led to some negative impacts in communication with other people as it eliminates speech perception by visual features through lipreading. Additionally, a considerable amount of face-to-face communications have turned into virtual communications, which results in more hardship for the hearing impaired and/or speech impaired people as some of the most used virtual communication services like Zoom, Microsoft Teams or Skype do not support sign language.

According to the World Health Organization (WHO), 5% of the people on the earth are hearing impaired, which is more than 350 million people [1] and will exceed 700 million by 2050 [2]. Considering that, sustainability of the social lives of hearing-impaired and/or speech impaired people will be an essential issue in the future. Therefore, we propose a solution to this problem named Signify. Signify is a mobile application with the main aim of helping hearing and/or speech impaired people in their social lives by translating sign language into text along with speech and text-speech to sign language translation in real-time.

This report consists of the object design trade-offs, interface documentation guidelines, engineering standards, definition, acronyms, packages we will use in the project, class interface diagrams, the explanation of the diagrams and functions in the diagram, glossary, and references of the report.

1.1. Object Design Trade-offs

1.1.1. Security vs. Portability

Computation for machine learning models requires high computational power. Using third-party computational power (computation in the cloud/server) leads to a security concern since the user's audio, the video should be sent to cloud/server. To handle this security concern, we plan to use local computational power on the user's phone, therefore, target devices are restricted. This hinders portability.

1.1.2. Rapid Development vs. Functionality

This project is a one-year project; therefore, we should develop the application rapidly. During the project implementation, we aim to implement our core functionalities as soon as possible. However, because of the short time we have, some of the planned functionalities might be ignored. For example, for sign language generation, the model can generate sign language; but it might not have a style transfer.

1.1.3. Efficiency vs. Portability

The application can be used efficiently on the targeted devices which provide high computational power. But if the device cannot provide the required computational power, the efficiency of the application might be decreased. For example, some computations might take a long time, the accuracy of the ML models might be decreased. Because we want that all users most efficiently use the application, we restricted the target devices.

1.2. Interface Documentation Guidelines

We will use the following convention for class descriptions.

Class Name	Explanation for the class	
Attributes	attributeName: type	Explanation for the attribute
Methods	methodName(args): return type	Explanation for the method

"Class Name" is the name of the class described, "Attributes" are listed as names and types followed by their explanations, "Methods" are listed as signatures and return types followed by their explanations.

1.3. Engineering Standards

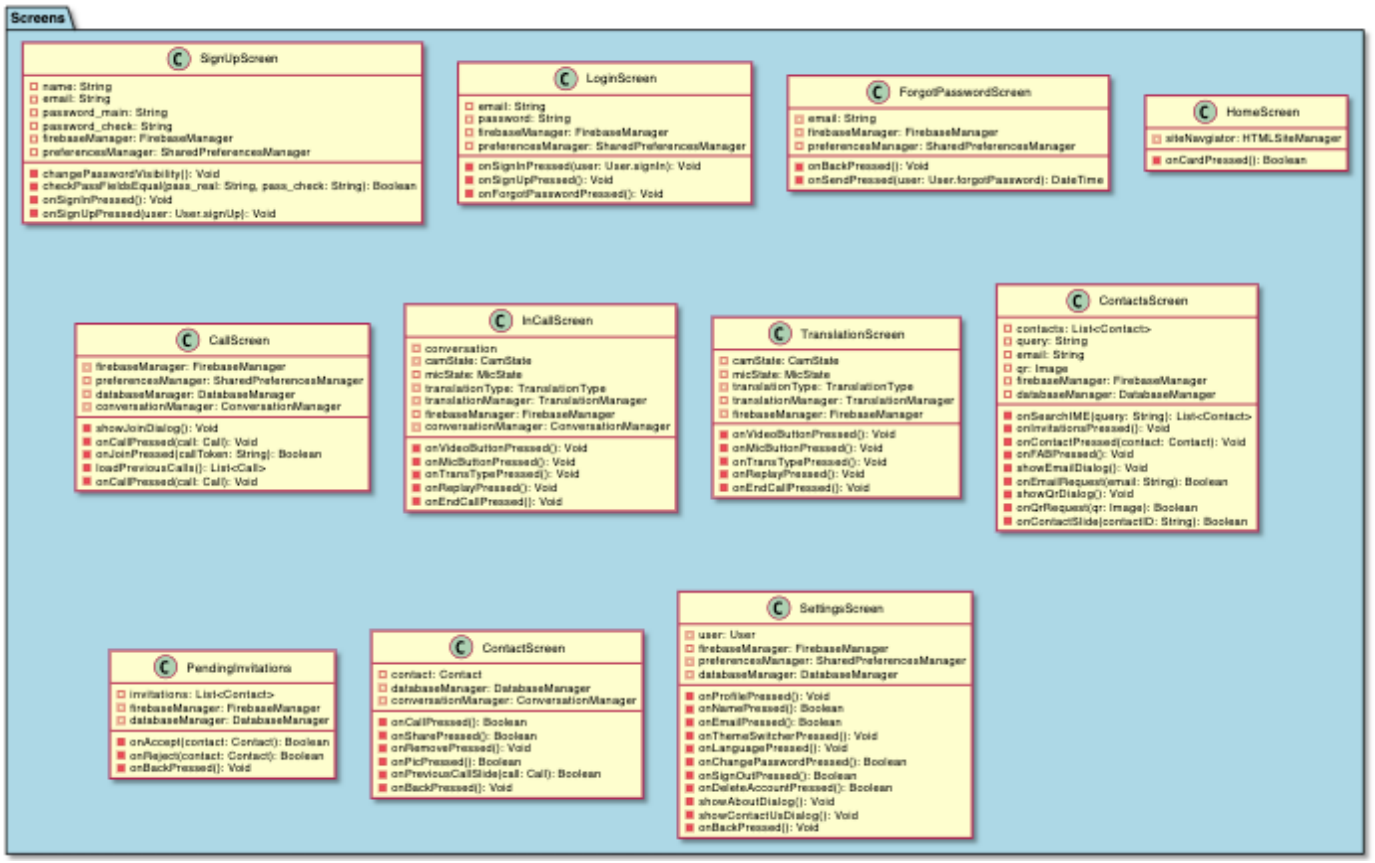
This report follows the Unified Modeling Language (UML) [3] standards to visualize the design of the system and represent class interfaces. Additionally, IEEE referencing style standards [4] for citations are used throughout the report for all of the citations.

1.4. Definitions, Acronyms, and Abbreviations

- **Online conference:** A feature where people can easily communicate online through our application.
- **Real-time communication:** A feature where people can easily communicate in real life through our application.
- **Bidirectional translation:** Translation from text to sign language and from sign language to text.
- **Web-RTC:** Name of the real time communication system and provider.
- **ASL:** American Sign Language, the chosen sign language for the application.
- **UI:** The user interface of the application.
- **DB:** Firebase Database that holds user-related and call related data.
- **ML:** Machine learning.
- **GAN:** Generative adversarial networks that are used to generate an ASL animation from the text information.
- **Model data:** Data that is used for the design and improvement of ML models.

2. Packages

2.1. Screens



SignupScreen: Displays the sign up screen before entering the application. App will fallback to this screen if the user has not signed up or logged in.

LoginScreen: Displays the login screen for the users who already have an account. App will fallback to this screen if the user logged off from the app.

ForgotPasswordScreen: Displays the screen to receive mail regarding resetting the account if the user already has an account.

HomeScreen: The main screen after logging or signing in to the application. This screen will welcome the user unless the user decides to logout.

CallScreen: Displays the call screen with call requests and history.

InCallScreen: Displays the call screen in the online conference joined or created.

TranslationScreen: Displays the translation screen with camera and microphone options for the translation in real-time.

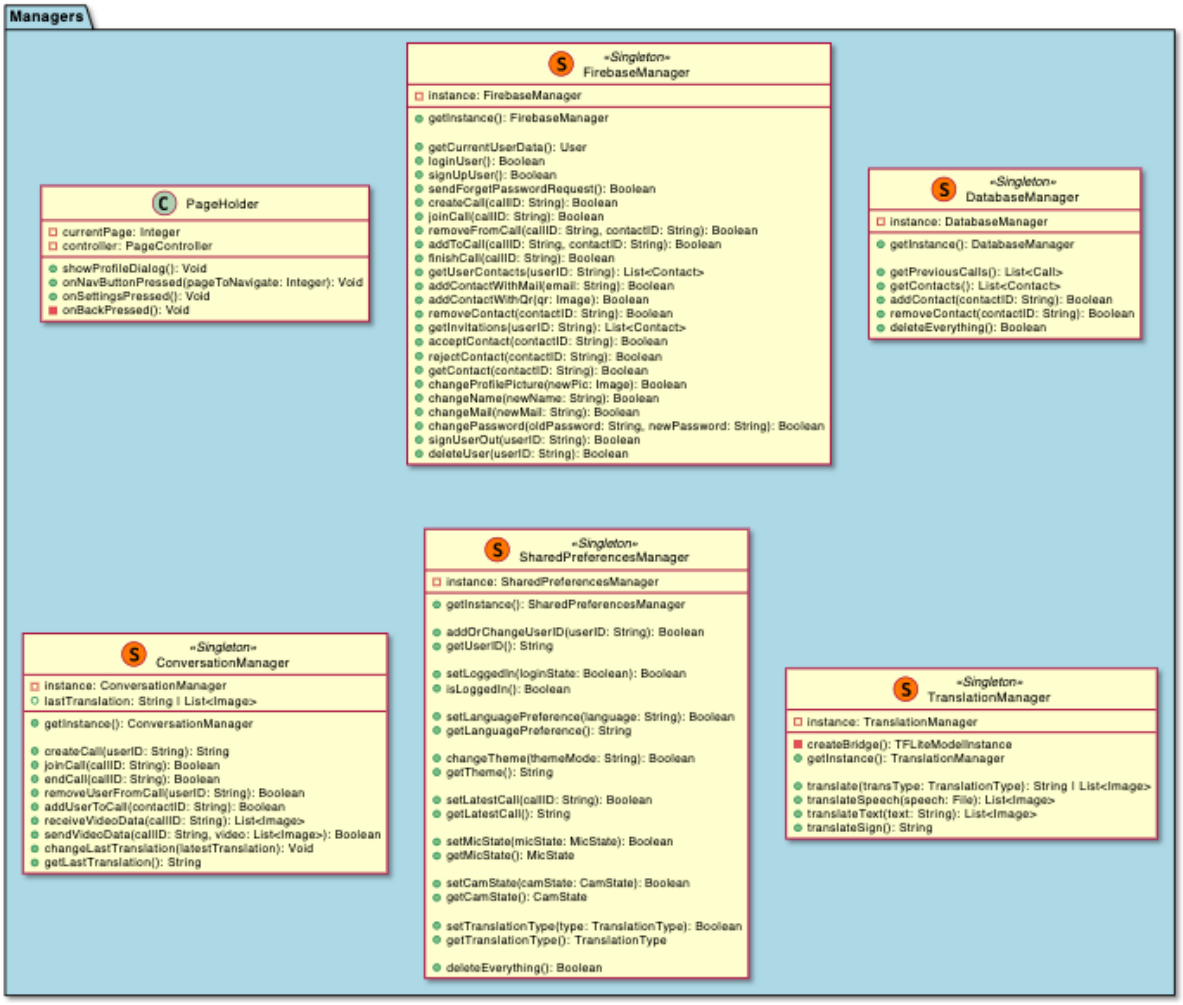
ContactsScreen: Displays the contacts of the user.

PendingInvitations: Displays pending contact invitations.

ContactScreen: Displays a specific contact with the profile information and call history.

SettingsScreen: Displays the settings for the users to customize the interface, or change the profile information according to their preferences.

2.2. Managers



PageHolder: Main manager to handle screen translation throughout the app. Basically, a super-class that contains each screen and handles the navigation.

FirebaseManager: Updates Firebase and retrieves information according to the user.

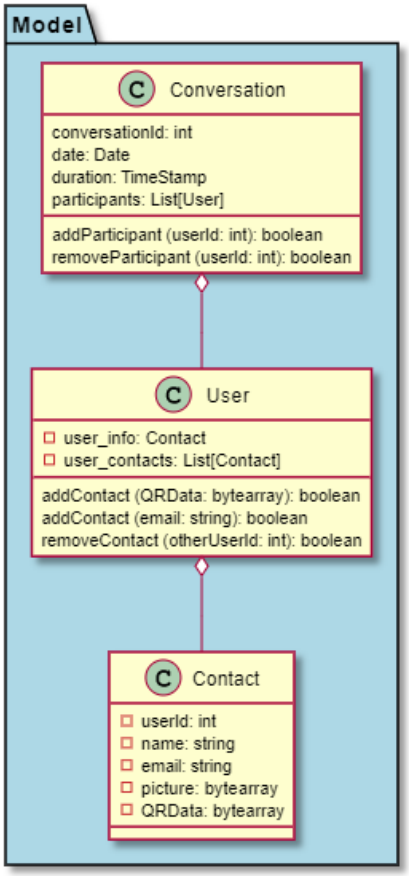
DatabaseManager: Updates the database and retrieves information about the contacts according to the user.

ConversationManager: Manages the requirements of the online conference.

SharedPreferencesManager: Manages the interface settings according to user preferences.

TranslationManager: Manages the translation among text, speech and video. Work as a bridge to main machine learning mechanics.

2.3. Model

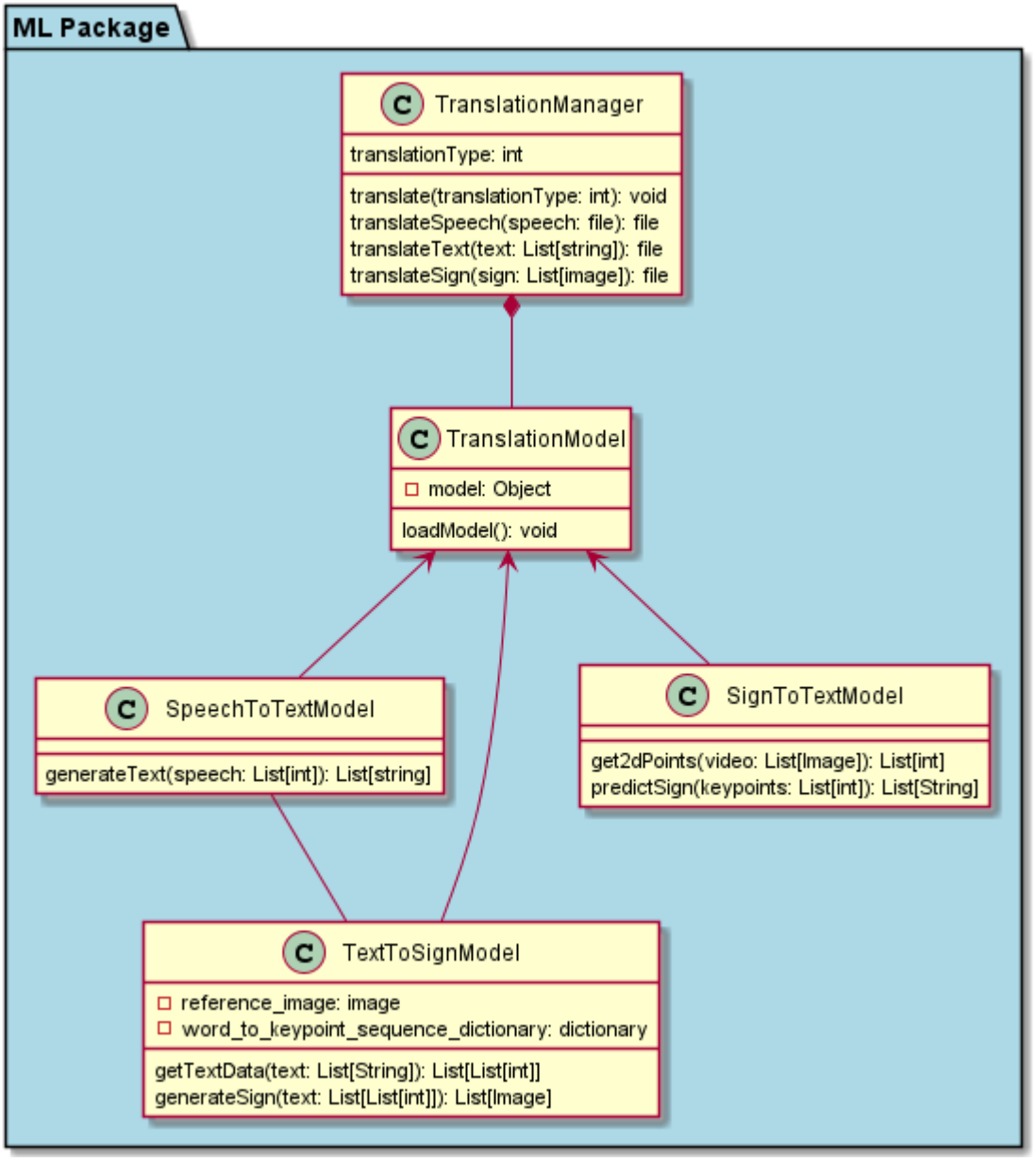


Conversation: Holds the information about the online conference.

User: Holds the information about the user.

Contact: Holds the information about a contact of the user.

2.4. Machine Learning



TranslationManager (Python): Manages the translation models which are text-to-sign, sign-to-video, and speech-to-text.

TranslationManager (Dart): Creates a bridge and handles the I/O operations between Flutter interface and Python interface.

TranslationModel: Superclass of all translation models.

SpeechToTextModel: Translates speech to text.

SignToTextModel: Translates sign language videos to texts.

TextToSignModel: Translates texts to sign languages.

3. Class Interfaces

3.1. Screens

SignUpScreen	Displays the sign up screen before entering the application.	
Attributes	name: String	Name input by the user.
	email: String	E-mail input by the user.
	password_main: String password_check: String	Password and password check input by the user.
	firebaseManager:FirebaseManager	Instance of a Firebase Manager class.
	preferencesManager:SharedPreferencesManager	Instance of a Shared Preferences Manager class.
Methods	changePasswordVisibility(): void	The option to make the password visible or invisible during user input.
	checkPasswordEquals(): boolean	Checks if the password main and password check matches.
	onSignInPressed():void	Calls Firebase instance and SharedPreferences instance and invokes sign-in operations including saving user token to preferences instance.
	onSignUpPressed(user: User.signUp): void	Navigates to LoginScreen without losing the already entered user info by using the User<SignUpType> constructor.

LogInScreen	Displays the login screen for the users who already have an account.	
Attributes	email: String	E-mail input by the user.
	password: String	Password input by the user.
	firebaseManager:FirebaseManager	Instance of a Firebase Manager class.
	preferencesManager:SharedPreferencesManager	Instance of a Shared Preferences Manager class.
Methods	onSignInPressed(user: User.signIn): void	Navigates to SignInScreen without losing the already entered user info by using the User<SignInType> constructor.
	onSignUpPressed(): void	Calls Firebase instance and SharedPreferences instance and invokes sign-up operations including saving already existing user token to preferences instance.
	onForgotPasswordPressed(): void	Navigates to ForgetPasswordScreen.

ForgotPasswordScreen	Displays the screen to receive mail regarding resetting the account if the user already has an account.	
Attributes	email: String	E-mail input by the user.
	firebaseManager:FirebaseManager	Instance of a Firebase Manager class.
	preferencesManager:SharedPreferencesManager	Instance of a Shared Preferences Manager class.
Methods	onBackPressed(): void	Goes back to the previous page when it is called.
	onSendPressed(user:User.forgotPassword): DateTime	Sends e-mail to the user when it is called. Returns a DateTime instance of the epoch time that the request sent in order to calculate the time to show the “resend” option.

HomeScreen	The main screen after logging in to the application.	
Attributes	siteNavigator: HTMLSiteManager	Site navigator attribute.
Methods	onCardPressed():boolean	Open the URL by using the HTMLSiteManager instance with respect to the clicked card content.

CallScreen	Displays the call screen with call requests and history.	
Attributes	firebaseManager: FirebaseManager	Instance of a Firebase Manager class.
	preferencesManager: SharedPreferencesManager	Instance of a Shared Preferences Manager class.
	databaseManager: DatabaseManager	Instance of a Database Manager class.
	conversationManager: ConversationManager	Instance of a Conversation Manager class.
Methods	showJoinDialog():void	Shows the joining dialog when it is called.
	onCallPressed(call:Call):void	Creates a call when it is called by using a Call instance containing the users' ID and created call instance ID.
	onJoinPressed(callToken:String): boolean	Joins the user to an ongoing call when it is called by using the callToken. Returns true if joining is successful, false otherwise.
	loadPreviousCalls():List<Call>	Retrieves the list of previous calls of the user.

InCallScreen	Displays the call screen in the online conference joined or created.

Attributes	camState:CamState	State of the camera: CamState.ON: Camera is On CamState.OFF: Camera is Off
	micState: MicState	State of the microphone: MicState.ON: Microphone is On MicState.OFF: Microphone is Off
	translationType: TranslationType	Type of the translation: TranslationType.SETT : speech to text TranslationType.SITT : sign to text TranslationType.TTS: text to sign
	translationManager:TranslationManager	Instance of Translation Manager class.
	firebaseManager:FirebaseManager	Instance of Firebase Manager class.
	conversationManager:ConversationManager	Instance of Conversation Manager class.
Methods	onVideoButtonPressed():void	Changes the state of the camera to CamState.ON if the camera is off and vice-versa.
	onMicButtonPressed():void	Changes the state of the microphone to MicState.ON if the microphone is off and vice-versa.
	onTransTypePressed():void	Changes the state of the translation type to other types sequentially.
	onReplayPressed():void	Replays the last translation when it is called.
	onEndCallPressed():void	Ends the call when it is called. Will invoke Firebase and Conversation Manager instances to handle the finishing operations.

TranslationScreen	Displays the translation screen with camera and microphone options for the translation in real life.	
Attributes	camState:CamState	State of the camera: CamState.ON: Camera is On CamState.OFF: Camera is Off
	micState: MicState	State of the microphone: MicState.ON: Microphone is On MicState.OFF: Microphone is Off
	translationType: TranslationType	Type of the translation: TranslationType.SETT : speech to text TranslationType.SITT : sign to text TranslationType.TTS: text to sign
	translationManager:Translation Manager	Instance of Translation Manager class.
	firebaseManager:FirebaseManager	Instance of Firebase Manager class.
Methods	onVideoButtonPressed():void	Changes the state of the camera to CamState.ON if the camera is off and vice-versa.
	onMicButtonPressed():void	Changes the state of the microphone to MicState.ON if the microphone is off and vice-versa.
	onTransTypePressed():void	Changes the state of the translation type to other types sequentially.
	onReplayPressed():void	Replays the last translation when it is called.

ContactsScreen	Displays the contacts of the user.	
Attributes	contacts:List<Contact>	Contact list of the user.
	query:String	Search query input string.

	email:String	E-mail of the user.
	qr:Image	Qr code of the user.
	firebaseManager:FirebaseManager	Instance of Firebase Manager class.
	databaseManager:DatabaseManager	Instance of Database Manager class.
Methods	onSearchIME(query: String): List<Contact>	Search a contact from the list according to the given query.
	onFABPressed():void	Opens the modal sheet containing options to add a new contact.
	onContactPressed(contact:Contact):void	Goes to the contact screen for the specific contact chosen.
	showEmailDialog():void	Opens a dialog in order to get the user input containing email of the contact to add.
	onEmailRequest(email:String):boolean	Sends an invitation request to the new added contact through Firebase channels by using the given email.
	showQrDialog():void	Opens a dialog in order to get the user input containing Qr of the contact to add..
	onQrRequest(qr:Image):boolean	Sends an invitation request to the new added contact through Firebase channels by using the given Qr.
	onContactSlide(contactID:String):boolean	Delets the slided contact instance from the users' contacts.

PendingInvitations	Displays pending contact invitations.	
Attributes	invitations:List<Contact>	List of the contact requests.
	firebaseManager:FirebaseManager	Instance of a Firebase Manager class.

	databaseManager:DatabaseManager	Instance of a Database Manager class.
Methods	onAccept(contact:Contact):boolean	Accepts the contact request when it is called.
	onReject(contact:Contact):boolean	Rejects the contact request when it is called.
	onBackPressed():void	Goes back to the previous page when it is called.

ContactScreen	Displays a specific contact with the profile information and call history.	
Attributes	contact:Contact	The current contact
	databaseManager:DatabaseManager	Instance of a Database Manager class.
	conversationManager:ConversationManager	Instance of a Conversation Manager class.
Methods	onCallPressed():boolean	Creates a call with the contact when it is called.
	onSharePressed():boolean	Opens sharing options when it is called.
	onRemovePressed():void	Removes the contact when it is called.
	onPicPressed():boolean	Shows the profile picture of the contact in full-screen mode.
	onPreviousCallSlide(call:Call):boolean	Deletes the slided previous call instance from all the saved places.
	onBackPressed():void	Goes back to the previous page when it is called.

SettingsScreen	Displays the settings for the users to customize the interface, or change the profile information according to their preferences.	
Attributes	user:User	Current user.
	databaseManager:DatabaseManager	Instance of a Database Manager class.
	firebaseManager:FirebaseManager	Instance of a Firebase Manager class.
	preferenceManager:SharedPreferencesManager	Instance of a Shared Preferences Manager class.
Methods	onProfilePressed():void	Shows a modal sheet in order to select a new profile picture that is taken from the camera or the gallery. The sheet also contains a selection for deleting the profile picture of the user.
	onNamePressed():boolean	Handles the input to change the name when it is called.
	onEmailPressed():boolean	Handles the input to change the email when it is called.
	onThemeSwitcherPressed():void	Handles the input to change the theme when it is called.
	onLanguagePressed():void	Handles the input to change the language when it is called.
	onChangePasswordPressed():boolean	Navigates to the change password options.
	onSignOutPressed():boolean	Signs the user out when it is called.
	onDeleteAccountPressed():boolean	Deletes the account of the user when it is called.
	showAboutDialog():void	Shows the about dialog when it is called.
	onBackPressed():void	Goes back to the previous page when it is called.

3.2. Managers

PageHolder	Main manager to handle screen translation throughout the app. Basically, a super-class that contains each screen and handles the navigation.	
Attributes	currentPage: Integer	Holds the id of the current page.
	controller:PageController	Instance of PageController class.
Methods	showProfileDialog():void	Shows the dialog containing user information and Qr Code of the user.
	onNavButtonPressed(pageToNavigate: Integer):void	Handles the navigation from currentPage to given page.
	onSettingsPressed():void	Navigates to the settings screen when it is called.
	onBackPressed():void	Closes the application.

FirabaseManager	Manages the information updates and retrieval from Firebase. All methods of this class use Firebase to operate requests.	
Attributes	instance:FirebaseManager	Instance of the class itself
Methods	getInstance():FirebaseManager	Retrieves the instance of the class.
	getCurrentUserData():User	Retrieves the current data of the user.
	loginUser():boolean	Logs in the user.
	signUpUser():boolean	Signs up the user and updates Firebase.
	sendForgetPasswordRequest():boolean	Sends a request to Firebase in order to send a reset password mail.
	createCall(callID:String):boolean	Saves the shareable callID to the Firebase.
	joinCall(callID:String):boolean	Saves userID to the given shareable callID which exists in

		the Firebase.
	removeFromCall(callID:String, contactID:String):boolean	Removes the userID from the given shareable callID which exists in Firebase.
	addToCall(callID:String, contactID:String):boolean	Add a contact to the call with respect to the given callID which exists in Firebase.
	finishCall(callID:String):boolean	Deletes the callID from the Firebase.
	getUserContacts(userID:String): List<Contact>	Retrieves the contact list of the user from the Firebase.
	addContactWithMail(email:String):boolean	Sends a request to the contact that exists in Firebase with the given e-mail.
	addContactWithQr(qr:Image):boolean	Sends a request to the contact that exists in Firebase with the given qr code.
	removeContact(contactID:String):boolean	Removes a contact from the user's contact list stored in Firebase.
	getInvitations(userID:String):List<Contact>	Retrieves contact requests waiting for approval in the Firebase for the user.
	acceptContact(contactID:String): boolean	Accepts the contact request and adds the contact to the user’s contacts stored in Firebase.
	rejectContact(contactID:String): boolean	Rejects the contact request and deletes the invitation from the list stored in Firebase.
	getContact(contactID:String):boolean	Retrieves specific contact information by using the given contactID from the Firebase.
	changeProfilePicture(newPic:Image):boolean	Changes the profile picture of the user stored in Firebase.
	changeName(newName:String): boolean	Changes the name of the user stored in Firebase.
	changeMail(newMail:String):boolean	Changes the email of the user stored in Firebase.

	changePassword(old:String, new:String):boolean	Changes the password of the user stored in Firebase.
	signUserOut(userID:String):boolean	Signs the user out and invokes Firebase for the logout event.
	deleteUser(userID:String):boolean	Deletes the user and all the information about it from the Firebase.

DatabaseManager	Updates the database and retrieves information about the contacts according to the user.	
Attributes	instance: DatabaseManager	Instance of the class itself.
Methods	getInstance(): DatabaseManager	Retrieves the instance of the class.
	getPreviousCalls():List<Call>	Retrieves the list of previous calls.
	getContacts():List<Contact>	Retrieves the list of contacts from the database.
	addContact(contactID:String):boolean	Saves the given contactID to the database.
	deleteEverything():boolean	Clears the database including all saved instances of previous calls and saved contacts.

ConversationManager	Manages the online conference calls.	
Attributes	instance:ConversationManager	Instance of the class itself.
	lastTranslation: String | List<Image>	Holds the last translation information. It can be a text translated from video data or a video generated from text.
Methods	getInstance():ConversationManager	Retrieves the instance of the class.
	createCall(userID:String):String	Creates a call on behalf of the user.

	joinCall(callID:String):boolean	Joins the user to the call with the given callID which was created before.
	endCall(callID:String):boolean	Ends the call.
	removeUserFromCall(userID:String):boolean	Removes the user from the call.
	addUserToCall(contactID:String):boolean	Adds a user to the call.
	receiveVideoData(callID:String): List<Image>	Receives the video data coming from the server and feeds to the UI.
	sendVideoData(callID:String, video:List<Image>):boolean	Sends the video output coming from the camera to share the video to the server that the call is going on.
	changeLastTranslation(latestTranslation):void	Updates the last translated data.
	getLastTranslation():String	Retrieves the last translation information.

SharedPreferences Manager	Manages the interface settings according to user preferences.	
Attributes	instance: SharedPreferencesManager	Instance of the class itself.
Methods	getInstance(): SharedPreferencesManager	Retrieves the instance of the class.
	addOrChangeUserID(userID:String):boolean	Saves the UserID created by a Firebase instance to preferences store. If an existing value is found, an update operation will be done.
	getUserID():String	Retrieves the user ID.
	setLoggedIn(loginState:boolean):boolean	Sets the logged in state of the user.
	isLoggedIn():boolean	Checks if the user is logged in.

	setLanguagePreference(language :String):boolean getLanguagePreference():boolea n	Retrieves and updates the language preferences of the user according to the input.
	changeTheme(themeMode:Strin g):boolean getTheme():String	Retrieves and updates the application's theme according to the input.
	setLatestCall(callID:String):bool ean getLatestCall():String	Retrieves and updates the latest call of the user.
	setMicState(micState:MicState): boolean getMicState():MicState	Retrieves and updates the latest state of the microphone.
	setCamState(camState:CamState):boolean getCamState():CamState	Retrieves and updates the latest state of the camera.
	setTranslationType(type:Translat ionType):boolean getTranslationType():boolean	Retrieves and updates the latest translation type of the user.
	deleteEverything():boolean	Clears the shared preferences data store, including deleting every key and their corresponding values.

TranslationManage r	Creates a bridge and handles the I/O operations between Flutter interface and Python interface.	
Attributes	instance: TranslationManager	Instance of the class itself.
Methods	createBridge(): TFLiteModelInstace	Creates a bridge between Python Manager instance and Dart Manager instance and returns it as a TFLite Model
	getInstance(): TranslationManager	Retrieves the instance of the class.
	translate(transType: TranslationType): String | List<Image>	Translates the given input and returns a string if the given translation type is TranslationType.SITT or

		TranslationType.SETT or a List of images which corresponds to a video if the translation type is TranslationType.TTS
	translateSpeech(speech:File): List<Image>	Translates given speech and returns a video output.
	translateText(text:String): List<Image>	Translates given test and returns a video output.
	translateSign(List<Image>): String	Translate given video in sign language to text.

3.3. Model

Conversation	A model class to hold information about online conversation (online conference call).	
Attributes	conversationId:int	Discriminative id for the call.
	date:Date	Date of the conversation created.
	duration:TimeStamp	Duration of the conversation after the creation.
	participants:List<User>	List of users participated in the call.
Methods	addParticipant(userId:int): boolean	Adds a user to the conversation.
	removeParticipant(userId:int):boolean	Removes a participant from the conversation.

User	A model class to represent the user.	
Attributes	user_info: Contact	Holds the information about user's id, name, e-mail, profile picture and qr code.
	user_contacts:List<Contact>	Holds the contact list of the user
Methods	addContact(QRData: bytearray): boolean	Adds the contact by qr code.

	addContact(email:String):boolean	Adds the contact by email
	removeContact(otherUserId:int):boolean	Removes the contact from the contacts list.

Contact	A model class that holds the information of the contact.	
Attributes	userId:int	Discriminative id of the contact.
	name:String	Name of the contact.
	email:String	E-mail of the contact.
	picture:bytearray	Profile picture of the contact.
	QRData:bytearray	Qr code assigned to the contact.

3.4. Machine Learning

TranslationManager	Manages the translation models which are text-to-sign, sign-to-video, and speech-to-text.	
Attributes	translationType: integer	Type of the translation: 0 : speech to text 1 : sign to text 2: text to sign
Methods	translate(translationType: int):void	Takes translation type and according to translation type calls the required function
	translateSpeech(speech: file):file	Takes speech file as a input and initialize SpeechToTextModel object to translate the speech to text
	translateText(text: List[String]):file	Takes text as a input and initialize TextToSignModel object to translate the text to sing language videos

	translateSign(sign:List[Image]):file	Takes sign language videos as a input and initialize SignToTextModel object to translate the sign language to text

TranslationModel	Superclass of all translation models.	
Attributes	model: Object	ML model accasible and different for all transational models
Methods	loadModel(): void	Load to translation model which model is required. The model path are static; therefore if the model is used, the model is loaded from static model path. Also, all sub-classes are override this method.

SignToTextModel	Translates sign language to text data. It is a subclass of the Translation Model class.	
Attributes	None	None
Methods	get2dPoints(video: List[Image]): List[int]	From input image of user extracts location of joints of hands which is used to process in prediction stage
	predictSign(keypoints: List[int])): List[String]	Pass the generated keypoint data through the model to get prediction of corresponding word related to movement.

SpeechToTextModel	Translates speech to text. It is a subclass of the Translation Model class.	
Attributes	None	None
Methods	generateText(speech List[int]):	Given speech data passed

	List[String]	through a model which creates text data in a list data structure.

TextToSignModel	Generates sign language video for given input word using the reference image of the user.	
Attributes	reference_image: Image	A picture of the user to pass to the GAN model as reference.
	word_to_keypoint_sequence_dictionary: Dictionary	A dictionary of a word and required location of joints to display that word.
Methods	getTextData(text: List[String]): List[List[int]]	Takes sentences from the user and returns the keypoints using the word_to_keypoint_sequence_dictionary attribute.
	generateSign(text: List[List[int]]): List[Image]	Using the keypoint sequence and reference image of the user generates a sequence of images that will be processed as a video.

4. References

[1] E. McPhillips , “World wide hearing loss: Stats from around the world,” *Audicus*, 14-Sep-2021. [Online]. Available: https://www.audicus.com/world-wide-hearing-loss-stats-from-around-the-world/. [Accessed: 05-Oct-2021].

[2] “Every 4th person to suffer hearing loss by 2050: Who,” *Down To Earth*. [Online]. Available: https://www.downtoearth.org.in/news/health/every-4th-person-to-suffer-hearing-loss-by-2050-who-75718. [Accessed: 05-Oct-2021].

[3] *What is Unified Modeling Language (UML)?* [Online]. Available: https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-uml/. [Accessed: 20-Feb-2021].

[4] “IEEE Referencing: Getting started with IEEE referencing,” *Library Guides*. [Online]. Available: https://libraryguides.vu.edu.au/ieeereferencing/gettingstarted#:~:text=%E2%80%9CIEEE%E2%80%9D%20stands%20for%20The%20Institute,paper%2C%20provided%20in%20square%20brackets.&text=This%20is%20known%20as%20an,of%20the%20work%20are%20provided. [Accessed: 20-Feb-2022].