UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA

FACULTY OF ENGINEERING, BUILT ENVIRONMENT & IT

DEPARTMENT OF COMPUTER SCIENCE

COS 301 MINI-PROJECT:

Mouthpiece Documentation

COMPUTER SCIENCE

Contents

1 Introduction

Mouthpiece is a mobile application that allows real time lip syncing animation based off the user's voice. The aim of the application is to animate the movements of a user's mouth in real time using a set of mouth images which corresponds to different sounds and will be displayed when the user produces that sound, this is achieved either using a volume based conversion approach or a formant based conversion approach that is facilitated by the use of a recurrent neural network (RNN), the options can be chosen by the user.

The Mouthpiece mobile application is able to work offline by storing a voice profile of the user if a formant based conversion option is selected by the user or when using the volume based conversion option. The mobile application is supported by an online web companion application that allows the registered user to upload custom mouth packs, which are a set of mouth images that can be downloaded and used on the mobile application's mouth animation.

2 Systems

The Mouthpiece mobile application is built up from multiple subsystems that defines its functionalities and features. The systems are not independent on each other but are separated into different subsystems to show the different functionalities of each part of the whole system. The following is a list of subsystems of the mobile application and systems that supports the Mouthpiece mobile application:

2.1 Android

The Android system is developed using the Flutter framework. This system is the core system that all other subsystems will be based on and will be used to call the other subsystems to create the whole system. This system deals with the integration of the other subsystems, calling of APIs, displaying of views, navigation of views, displaying the mouth images to form an animation from using the converter's functionalities, and event handlers of all widgets.

2.2 Sharing API

The sharing API system is a REST API system that deals with storing of mouth pack's images and all other related data in a compressed format within a database. The system allows an API access for the android and web system to upload and download of the mouth pack's images by giving a JSON response with the mouth pack's information and where to download the specific mouth pack.

2.3 User management

The user management system is a REST API system that deals with the storing of users' information and all other related data in a database. The system provides an API for the other systems to allow the creation of user profiles, updates of user profiles, querying of users profiles, and authentications of users through JSON responses.

2.4 Converter

The converter system deals with the real time lip syncing animations of the application by returning the corresponding mouth image index according to the voice model when using formant based, or according to the volume level in a volume based approach. The converter system converts sound data to a format readable for the neural network to train the voice model and uses the trained voice model to get the correct mouth image index and return the index in real time, if the a volume based approach is used, it will not use the voice model but return the mouth image index based on each mouth volume range and return the corresponding mouth image based on the volume range.

2.5 Notification

The notification system deals with the local push notifications, network push notifications, and email notifications based off on relevant triggers that can occur from downloading of mouth packs, uploads of mouth packs, creations of profiles etc. as well as tips in a form of local push notifications.

2.6 Neural Network

The neural network system is a recurrent neural network (RNN) that deals with the training of the users' voice profile, hosted on a separate server, which the converter system will uploads a form of voice data to the neural network server to train the voice data and the neural network system will return a voice model back to the converter system to use. This system is used for the formant based conversions approach of the application.

2.7 Web application

The web application is not part of the mobile application but as a companion application. This system allows the creations, querying, updating of a user's profile through the use of user management API system, and featuring the functionalities of uploading mouth packs, searching of mouth packs with filtering preferences through the use of the sharing API system.

3 Structure

The Mouthpiece mobile application is developed using the Flutter framework, the following is the application structure and its definition of the mobile application.

```
lib/
     core/
          data/
          enum/
          models/
          services/
               authentication/
               converter/
               permissions/
               api.dart
          viewmodels/
     ui/
          shared/
          views/
          widgets/
          router.dart
     locator.dart
     main.dart
test/
```

3.1 Application Structure

The following are the names of the folders and files and definition of its functionalities:

- **lib/** - Contains all the functionalities of the mobile application and the building of the Flutter widgets.
- **core/** - Contains all the files that is associated with the logical processing of the application.
- **data/** - Contains the data that is retrieved from external sources through API calls.
- **enum/** - Contains all the enum variables.
- **models/** - Contains all the plain data models

- **services/** - Contains the dedicated files that will handle all the business logical processing.
- **authentication/** - Contains the files that will use the API to get the users details and update it accordingly.
- **converter/** - Contains the files used for volume based conversion or formant based conversions.
- **permission/** - Contains the files dealing with permissions in regards to the mobile device.
- **api.dart** - File used to request and serialies data.
- **viewmodels/** - Contains the provider models for each of the widget views. It is used for dependency injection and state management of the application.
- **ui/** - Contains all the files associated with the building of the views and handling of the navigation.
- **shared/** - Contains files that is used in multiple other user interface files.
- **views/** - Contains the files for the application views. Each file is built based on stateless and stateful application widgets.
- **widgets/** - Contains the widget files that should be used across the application and are too big to keep in the view file.
- **router.dart** - Contains the routes of the application and imports of all the views and screens.
- **locator.dart** - File that is used to locate the services and models based on the get_it package. It is a way to decouple the interface (abstract base class) from a concrete implementation from everywhere in the application over the interface. More information can be found from IoC Containers, Dependency Injections, and Get_it package.
- **main.dart** - File that is the entry point for the instantiation of the application.
- **test/** - Contains all the unit test files for testing the functions of the applications based off of the functions expected output.

4 Resources

- https://github.com/gjcsup/cos301_teamgamma.git
- Flutter framework
- FireBase framework
- Mouthpiece web companion application