Arab International University

Faculty of Informatics and Communication Engineering

Junior Project Report on

Speech Training App for Stroke Patients

Submitted to

Department of Informatics Engineering

in partial fulfillment of the requirement for the Degree of Bachelor in

Informatics Engineering

Submitted by

Aous Abdulhamid **Mohammed Shahrour**

Under the Supervision of

Eng. Massa Albaali

February 2022

Faculty of Informatics & Communication Engineering

CERTIFICATE OF APPROVAL

The undersigned certify that they have read and recommended to the Department of Informatics Engineering for acceptance, a project report entitled Speech Training App for Stroke Patients Submitted by: Ahmad Salamoun, Aws Abdulhamed, Jawad Murad, Marina Heloun, Mohammed Shahrour, in partial fulfillment for the degree of Bachelor of Engineering in Informatics.

Supervisor	**Head of Department**
Eng. Massa Albaali	**Dr. Said Desouki**

Acknowledgments

It gives us immense pleasure to express our deepest sense of gratitude and sincere thanks to our highly respected and esteemed guide Dr. **Massa Albaali** for her valuable guidance, encouragement, and help in completing this work, her useful suggestions for this whole work and cooperative behavior are sincerely acknowledged.
We are also grateful for her constant support and guidance.

In the end, we would like to express our sincere thanks to all our families, friends, and others who helped us directly or indirectly during this project work.

Abstract

Fluencia is an app to train people who stutter to get rid of stuttering. It gives them exercises that can help them to control their breathing rhythm, and to know some other techniques to use while talking to achieve more fluency.

Exercises are given based on features that we extracted from the recordings we asked them to record. Features are to know whether they have word repetition and/or letters repetition or not to know if their speaking speed rate is above or under average, and if they have grammatical mistakes.

We also embedded a Chatbot to give the user an engaging experience and to feel free to talk to a virtual friend whenever he likes. The Chatbot can interact and engage in various talks to encourage the patient to have a conversation to achieve confidence at last.

Contents

Abbreviations

SLP	Speech-Language Pathologist
DAF	Delayed Auditory Feedback
API	Application Programming Interface
WER	Word Error Rate
WPM	Words per Minute
ANN	Artificial Neural Network
UI	User Interface
JSON	JavaScript Object Notation
WSDL	Web Services Description Language
XML	Extensible Markup Language
ONEIROS	Open-ended Neuro-Electronic Intelligent Robot Operating System
SDK	Software Development Kit
WSGI	Web Server Gateway Interface
NLTK	Natural Language Toolkit
CPU	Central Processing Unit

Keywords

Chabot, ASR, TTS, NLP, Deep Learning, Stroke, API

Introduction

The project objective is to get a voice record to specify the kind of problem he has in his/her way of talking, later we can offer him/her exercises based on his/her kind of problem.

Training and exercises are offered by the Chatbot then, on the patient's demand. The Chatbot can show him/her and guide him/her to use these exercises each day until he finishes the required exercises that should allow him/her to get over his/her troubles.

In creating the chatbot, deep learning techniques are used so that the chatbot can answer any question from a medical trainee on a particular case.

Chapter 1: Project Description

1.1 Background

The project objective is to get a voice record to specify the kind of problem has in his/her way of talking, later we can offer him/her exercises based on his/her kind of problem. Exercises should help the patient to get over his/her troubles.

1.2 Problem Statement

The patient faces difficulties in speaking. The aim is to help him/her to pursue his/her communication goals and the life that he wants to live.

1.3 Project Objective

The objective of the project is to build a mobile application that helps people who stutter to be able to improve their flow of speech. Speaking slowly and deliberately can reduce stress and the symptoms of a stutter.

1.4 Project Scope

The system can be used by every stutter to learn to speak more slowly and addressing anxiety issues are among the techniques an SLP can use with both children and adults.

1.5 Project Features

Provide patients exercises to practice based on his/her troubles.

Chatbot to engage and talk with.

A guided training plan that adapts as the patient trains.

Statistics and assessments.

1.6 Project Feasibility

This system can be used by any stutter to guide and train him/her to talk slowly and deliberately to achieve fluency and confidence in speaking.

The main purpose is to help the patient by training him/her, and by allowing him/her to save a lot of money because of the high cost of treatment by the SLP.

1.7 System Requirements

The objective of the project is to build a mobile application that allows people who stutter to improve and manage their pronunciation and allow them to practice exercises. To help them to get rid of their difficulties.

This system can be used by any stutter to guide and train him/her to talk slowly and deliberately to achieve fluency and confidence in speaking.

Chapter 2: Theoretical Study

2.1 Stroke

A stroke occurs when the blood supply to part of your brain is interrupted or reduced, preventing brain tissue from getting oxygen and nutrients. Brain cells begin to die in minutes. A stroke is a medical emergency, and prompt treatment is crucial.

2.1.1 Complications

Difficulty talking or swallowing. A stroke might affect control of the muscles in your mouth and throat, making it difficult for you to talk clearly, swallow or eat. You also may have difficulty with language, including speaking or understanding speech, reading, or writing.

2.2 Stuttering

It's also called stammering or childhood-onset fluency disorder — is a speech disorder that involves frequent and significant problems with normal fluency and flow of speech. People who stutter know what they want to say, but have difficulty saying it.

2.2.1 Treatment

Stuttering is not curable. However, multiple things can be done to help a person who stutters pursue their communication goals and the life that they want to live.

2.2.2 Diagnoses

Stuttering is usually diagnosed by a speech-language pathologist. This is a health professional who is trained to test and treat people with voice, speech, and language disorders. If you or your child stutters, your regular health care provider may give you a referral to a speech-language pathologist.

Chapter 3: Literature Review

3.1 Related Works

The system can be used by every stutter to learn to speak more slowly and addressing anxiety issues are among the techniques an SLP can use with both children and adults.

An anti-stuttering app may enable them to be their guide in the quest for fluency through guided training exercises curated by professionals.

3.1.1 Stamurai

Stamurai is a Mobile app to help you learn and practice speech therapy exercises for stuttering including the following features:

1. You can practice reading aloud, record your voice and observe the dis-fluencies.
2. You can enjoy guided meditation. App-guided meditation will teach you coastal breathing techniques ubiquitous for fluency.
3. Practice breathing exercises and controlled exhalation while speaking.
4. You will get to know the function and challenges of your speech mechanism by answering a few simple questions.
5. Use delayed auditory feedback(DAF) to compensate for auditory processing defects
6. Log daily ratings such as those used in Lidcombe Program

3.1.2 BeneTalk

BeneTalk is a habit-building app made for people who stutter by people who stutter. Use BeneTalk to learn, practice, and ingrain communication habits to express yourself with less tension and struggle.

BeneTalk has the following features:

1. Cutting-edge voice recognition technology.
2. Real-time personalized visual feedback on your speech.
3. Practice exercises: choose between reading or freestyle modes.
4. View feedback reports to self-reflect on your practice.
5. Listen back to your recordings to self-evaluate.
6. Daily tips to help you along the way.

3.2 Comparison

Features	**BeneTalk**	**Stamurai**	**Fluencia**
Voice Recognition	**Provide**	**Provide**	**Provide**
Free Platform	**Free**	**Paid**	**Free**
ChatBot	**None**	**None**	**Includes**

Chapter 4: Features

4.1 Repetition

This feature will check if the user has repeated consecutive words in his/her speech. After Speech-to-Text conversion, the text will be passed to this method to check whether it contains a word repeated consecutively by comparing each word in the string with its following word and so on.

Repeated words will be stored in a list to be shown later to the user in the analytical report. The text will get stripped from the repeated words and will be corrected to be shown to the user so he can correct his/her mistakes and know what words he repeats the most.

4.2 Grammatical Mistakes

In this feature, we show the user his/her percentage mistake of the total word that he/she spoken by Algorithm Called The Levenshtein distance.

The Levenshtein distance it’s Calculate the Word Error Rate =Number of errors (Substitution + insertions + deletions) / Spoken Words (total words).

After we know the WER, we used TextBlob which is a Python library for processing textual data that has a feature of spelling correction, so we can correct all the word errors.

4.3 Speed

This feature is to measure the speaking speed of the patient and calculate how many words per minute are spoken depending on the speaking average rate.

-How to calculate your speaking rate:

Speaking rate is often expressed in words per minute (WPM). To calculate this value, you’ll need to record your voice talking for a few minutes and then add up the number of words in your speech. Divide the total number of words by the number of minutes your speech took to be recorded.

Speaking rate (wpm) = Total Words / Number of Minutes

For example, if your speech was 4 minutes 30 seconds, you need to divide the number of words that person talks by 4.5.

Average speech rates testing:

- In presentations people can talk between 100 - 150 wpm at a comfortable pace.
- In conversational people can talk between 120 - 150 wpm.
- In audiobooks people can talk between 150 - 160 wpm, which is the upper range that people comfortably hear and vocalize words.
- In radio hosts and podcasters people can talk between 150 - 160 wpm.
- In auctioneers people can talk can speak at about 250 wpm.
- In commentators people can talk between 250- 400 wpm.

4.4 Delay

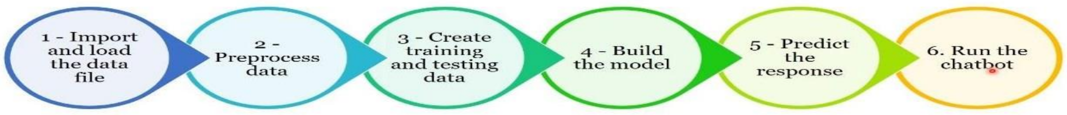
This feature will check if the user has repeated characters in his speech. After converting speech to text, the text will be passed to this method to check whether the word contains consecutive letters.

The word with repeated letters will be stored in a list to be shown later to the user in the analytical report. The text will be stripped of the repeating characters and will be corrected to display to the user so that he can correct his mistakes and find out which characters he repeats the most.

Chapter 5: ChatBot

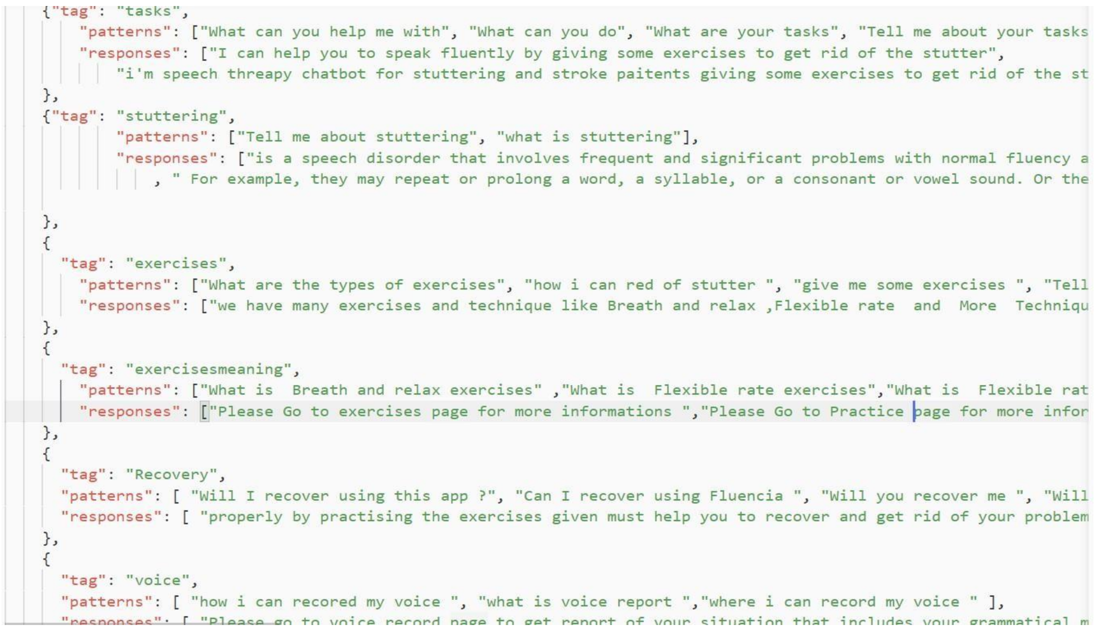
5.1 Description

In this chapter, we are going to build a Chabot using deep learning techniques. The Chabot will be trained on the dataset which contains categories (intents), patterns, and responses. We use a special artificial neural network (ANN) to classify which category the user's message belongs to and then we will give a random response from the list of responses.



5.2 Data Collection

This data set we make it and its type is JSON format (JavaScript ObjectNotation) include tag, patterns, and response.



```
  {"tag": "tasks",
     "patterns": ["What can you help me with", "What can you do", "What are your tasks", "Tell me about your tasks
     "responses": ["I can help you to speak fluently by giving some exercises to get rid of the stutter",
         "i'm speech threapy chatbot for stuttering and stroke paitents giving some exercises to get rid of the st
  },
  {"tag": "stuttering",
         "patterns": ["Tell me about stuttering", "what is stuttering"],
         "responses": ["is a speech disorder that involves frequent and significant problems with normal fluency a
             , " For example, they may repeat or prolong a word, a syllable, or a consonant or vowel sound. Or the

  },
  {
    "tag": "exercises",
     "patterns": ["What are the types of exercises", "how i can red of stutter ", "give me some exercises ", "Tell
     "responses": ["we have many exercises and technique like Breath and relax ,Flexible rate  and  More  Techniqu
  },
  {
    "tag": "exercisesmeaning",
     "patterns": ["What is  Breath and relax exercises" ,"What is  Flexible rate exercises","What is  Flexible rat
     "responses": ["Please Go to exercises page for more informations ","Please Go to Practice page for more infor
  },
  {
    "tag": "Recovery",
    "patterns": [ "Will I recover using this app ?", "Can I recover using Fluencia ", "Will you recover me ", "Will
    "responses": [ "properly by practising the exercises given must help you to recover and get rid of your problem
  },
  {
    "tag": "voice",
    "patterns": [ "how i can recored my voice ", "what is voice report ","where i can record my voice " ],
    "responses": [ "Please go to voice record page to get report of your situation that includes your grammatical m
```

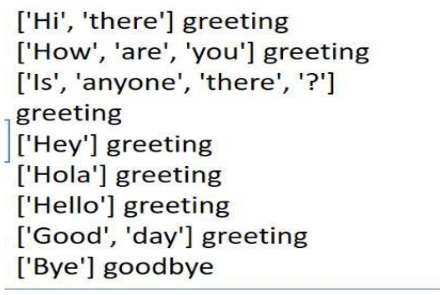
5.3 Preprocessing data

When working with text data, we need to perform various preprocessing on the data before we make a deep learning model. Based on the requirements we need to apply various operations to preprocess the data.

Tokenizing is the first thing you can do on text data. Tokenizing is the process of breaking the whole text into small parts like words.

Here we iterate through the patterns and tokenize the sentence using nltk.word_tokenize() function and append each word in the words list. We also create a list of classes for our tags.

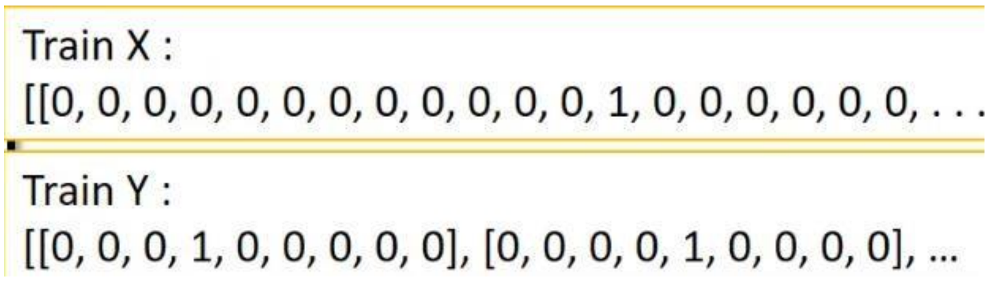
we will lemmatize each word and remove duplicate words from the list. Lemmatizing is the process of converting a word into its lemma form and then creating a pickle file **Training.py (** words.pkl, classes.pkl **):** to store the Python objects that we will use while predicting.



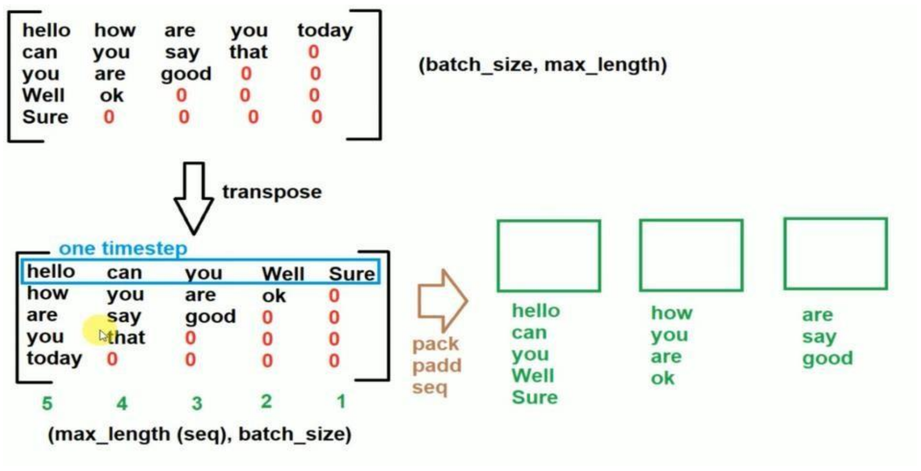
```
['Hi', 'there'] greeting
['How', 'are', 'you'] greeting
['Is', 'anyone', 'there', '?']
greeting
['Hey'] greeting
['Hola'] greeting
['Hello'] greeting
['Good', 'day'] greeting
['Bye'] goodbye
```

5.4 Create training and testing data

We will now create the training data in which we will provide the input and the output. Our input will be the pattern and output will be the class our input pattern belongs to. But the model doesn’t understand the text so we will convert text into numbers.



```
Train X :
[[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, . . .
Train Y :
[[0, 0, 0, 1, 0, 0, 0, 0, 0], [0, 0, 0, 0, 1, 0, 0, 0, 0], ...
```

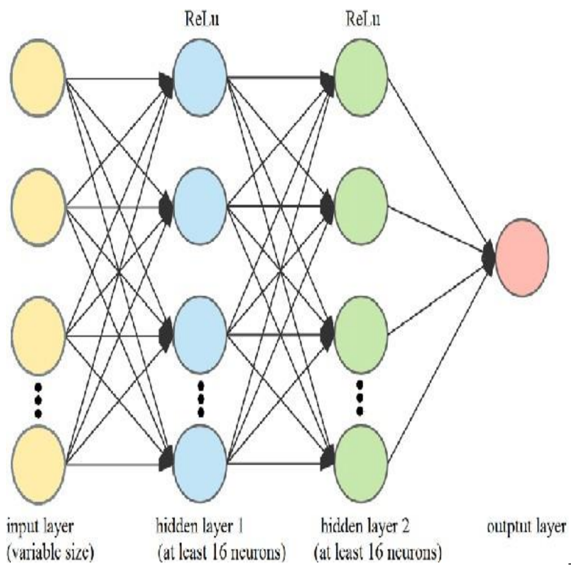


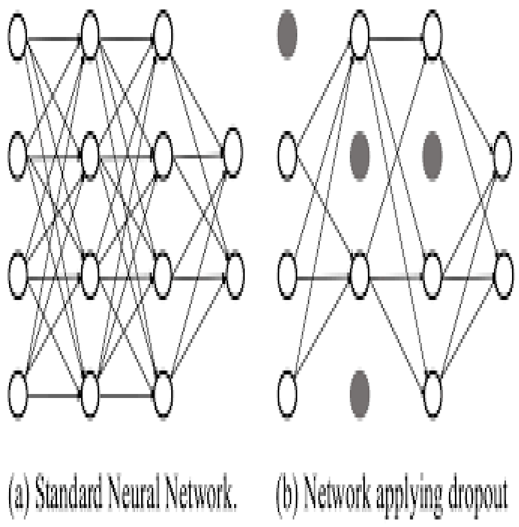
5.5 Model Architecture:

We will build a deep neural network that has 3 layers **First layer** 128 neurons**, the second layer** 64 neurons, and the **third layer** output layer contains several neurons Equal to the number of intents to predict output intent with softmax

We use the Keras sequential for this and Compile our model using Stochastic gradient descent With Nesterov accelerated gradient explained in chapter 5.5.1 gives good results for this model and it's how we cancel overfitting.

Deep Learning Sequential Neural Network and Dropout example:



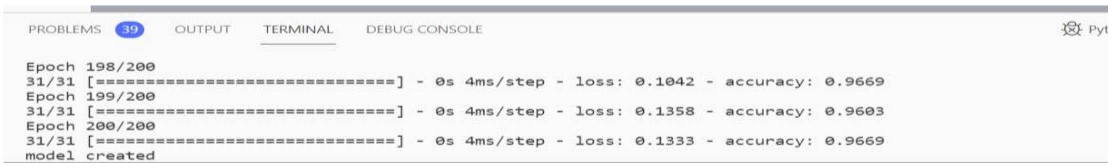


Activation layer: ReLU produces outputs faster than other common activation functions for both feedforward and backpropagation.

Softmax activation: we will use the Softmax activation function in the output layer in the above example. The Softmax activation function calculates the relative probabilities. That means it uses the value of Z21, Z22, Z23 to determine the final probability value, the softmax activation function works. Similar to the sigmoid activation function the SoftMax function returns the probability of each class. Here is the equation for the SoftMax activation function.

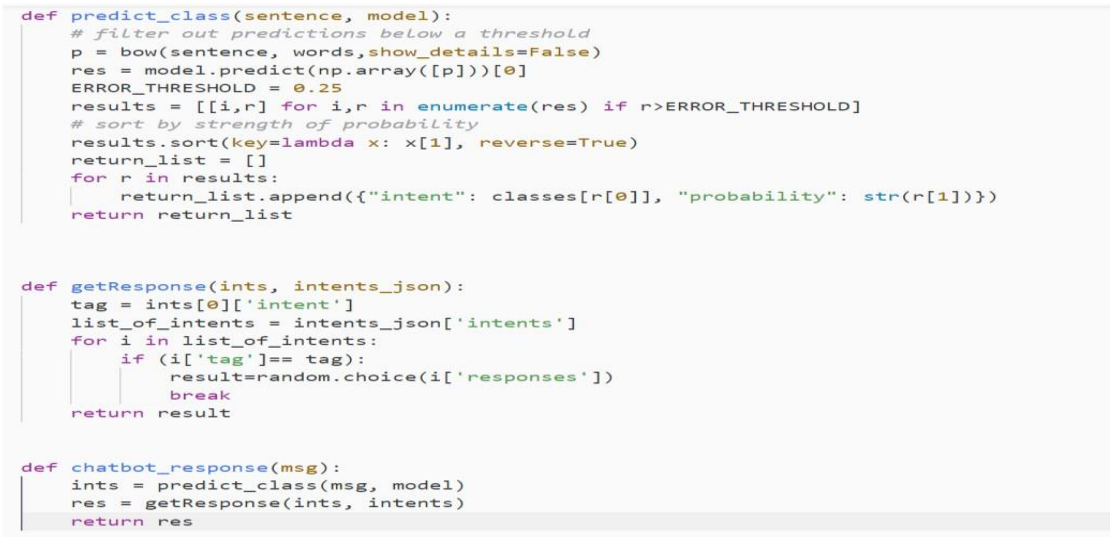
$$softmax(z_i) = \frac{exp(z_i)}{\sum_j exp(z_j)}$$

After training the model for 200 epochs, we achieved 96% accuracy on our model.



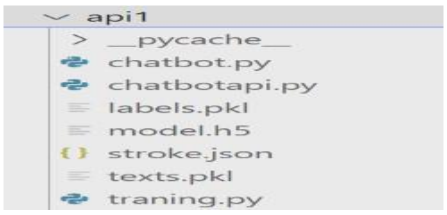
```
Epoch 198/200
31/31 [==============================] - 0s 4ms/step - loss: 0.1042 - accuracy: 0.9669
Epoch 199/200
31/31 [==============================] - 0s 4ms/step - loss: 0.1358 - accuracy: 0.9603
Epoch 200/200
31/31 [==============================] - 0s 4ms/step - loss: 0.1333 - accuracy: 0.9669
model created
```

5.6 Predicting the response:



```
def predict_class(sentence, model):
    # filter out predictions below a threshold
    p = bow(sentence, words,show_details=False)
    res = model.predict(np.array([p]))[0]
    ERROR_THRESHOLD = 0.25
    results = [[i,r] for i,r in enumerate(res) if r>ERROR_THRESHOLD]
    # sort by strength of probability
    results.sort(key=lambda x: x[1], reverse=True)
    return_list = []
    for r in results:
        return_list.append({"intent": classes[r[0]], "probability": str(r[1])})
    return return_list


def getResponse(ints, intents_json):
    tag = ints[0]['intent']
    list_of_intents = intents_json['intents']
    for i in list_of_intents:
        if (i['tag']== tag):
            result=random.choice(i['responses'])
            break
    return result


def chatbot_response(msg):
    ints = predict_class(msg, model)
    res = getResponse(ints, intents)
    return res
```

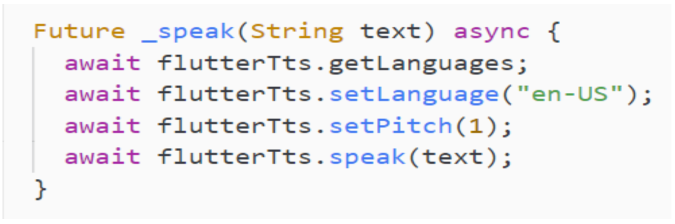
5.5 File Structure and Linking With API:



```
api1
  > __pycache__
  chatbot.py
  chatbotapi.py
  labels.pkl
  model.h5
  stroke.json
  texts.pkl
  traning.py
```

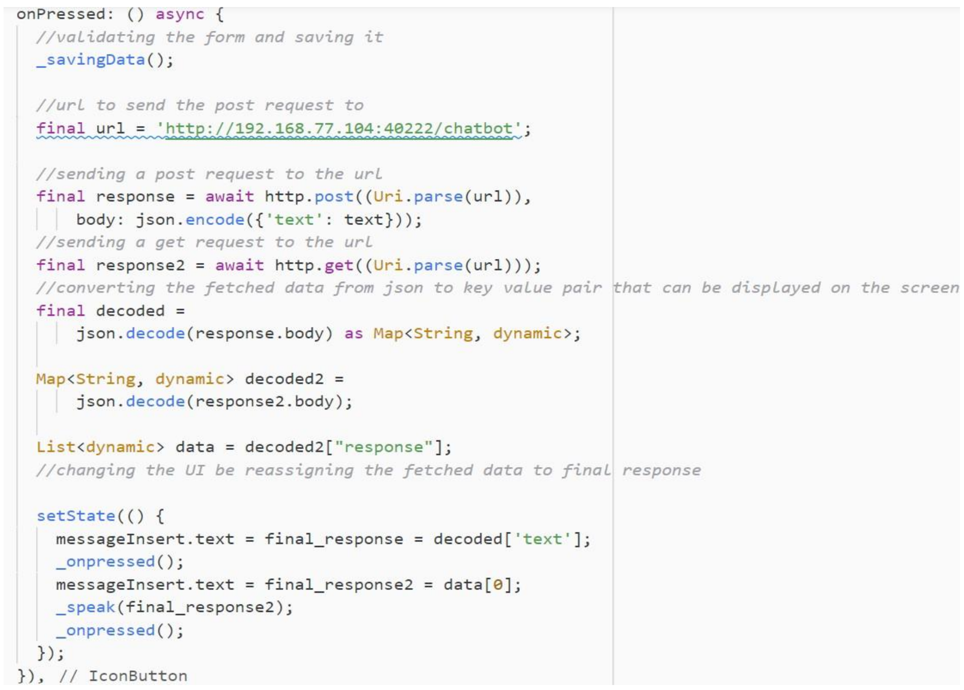
- In Front-End we used that API to get the response from flask

And we Transfer the response from text to speech using flutter TTS package:

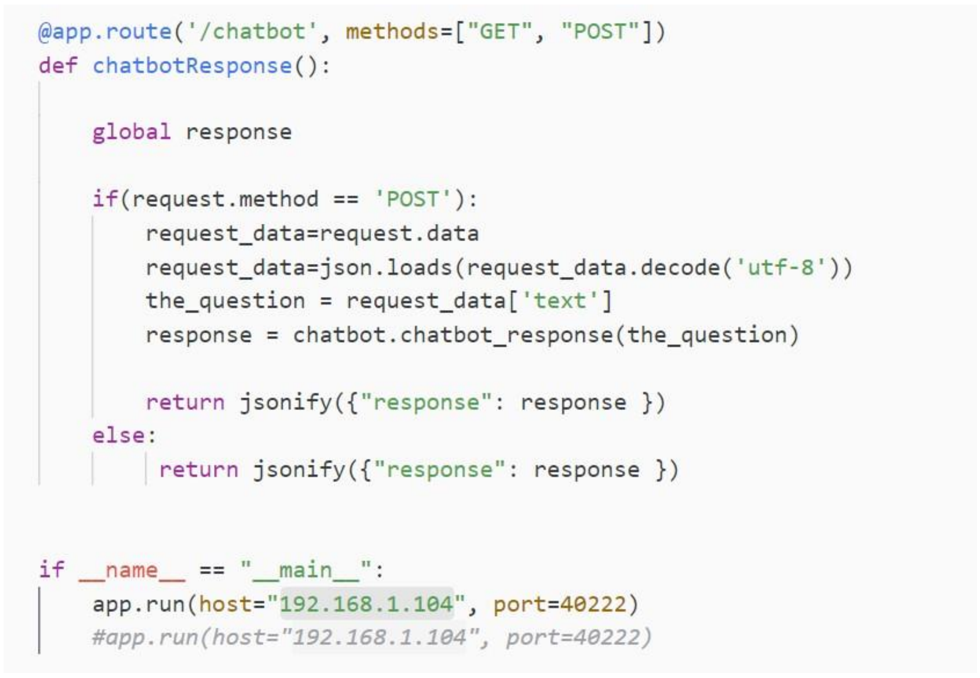


```
Future _speak(String text) async {
  await flutterTts.getLanguages;
  await flutterTts.setLanguage("en-US");
  await flutterTts.setPitch(1);
  await flutterTts.speak(text);
}
```

APIs in flutter:

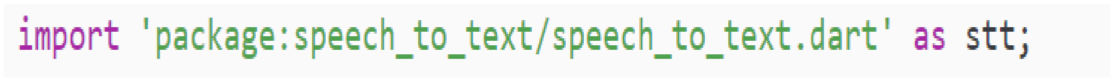

```dart
onPressed: () async {
  //validating the form and saving it
  _savingData();

  //url to send the post request to
  final url = 'http://192.168.77.104:40222/chatbot';

  //sending a post request to the url
  final response = await http.post((Uri.parse(url)),
      body: json.encode({'text': text}));
  //sending a get request to the url
  final response2 = await http.get((Uri.parse(url)));
  //converting the fetched data from json to key value pair that can be displayed on the screen
  final decoded =
      json.decode(response.body) as Map<String, dynamic>;

  Map<String, dynamic> decoded2 =
      json.decode(response2.body);

  List<dynamic> data = decoded2["response"];
  //changing the UI be reassigning the fetched data to final response

  setState(() {
    messageInsert.text = final_response = decoded['text'];
    _onpressed();
    messageInsert.text = final_response2 = data[0];
    _speak(final_response2);
    _onpressed();
  });
}), // IconButton
```

Flask API:


```python
@app.route('/chatbot', methods=["GET", "POST"])
def chatbotResponse():

    global response

    if(request.method == 'POST'):
        request_data=request.data
        request_data=json.loads(request_data.decode('utf-8'))
        the_question = request_data['text']
        response = chatbot.chatbot_response(the_question)

        return jsonify({"response": response })
    else:
        return jsonify({"response": response })


if __name__ == "__main__":
    app.run(host="192.168.1.104", port=40222)
    #app.run(host="192.168.1.104", port=40222)
```

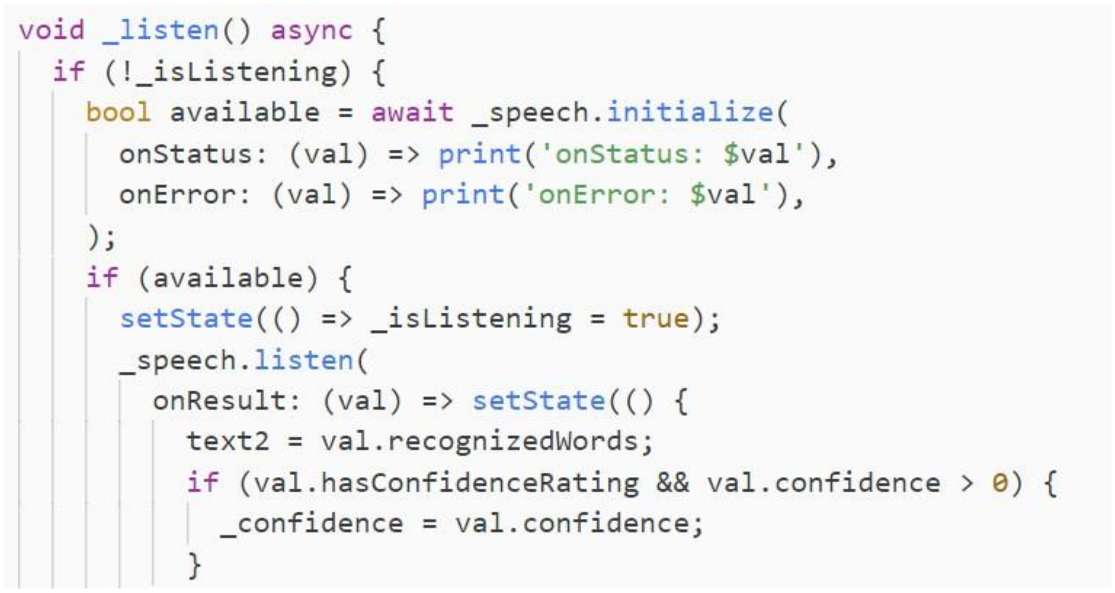
5.6 Speech Recognition

- We use a speech to text package to convert speech to text



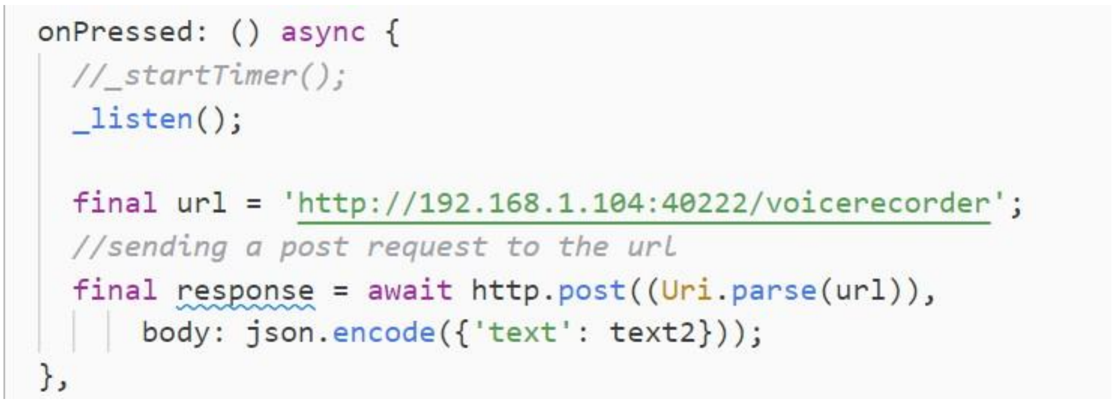
```dart
import 'package:speech_to_text/speech_to_text.dart' as stt;
```

- Here we get the text and we save it in text2 variable



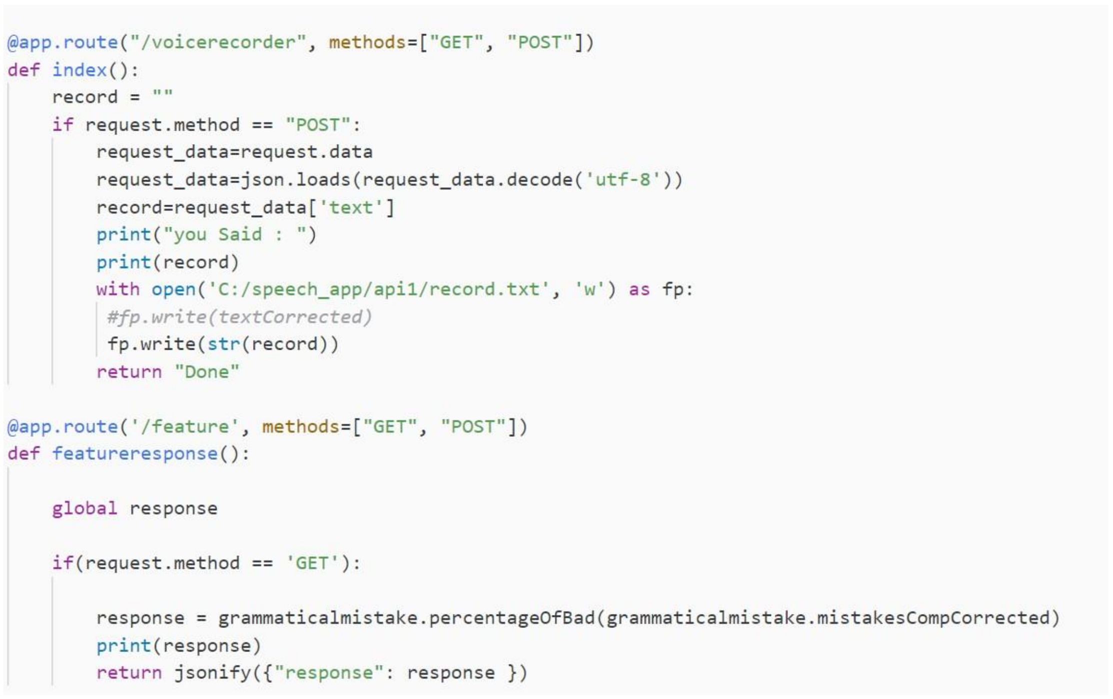
```dart
void _listen() async {
  if (!_isListening) {
    bool available = await _speech.initialize(
      onStatus: (val) => print('onStatus: $val'),
      onError: (val) => print('onError: $val'),
    );
    if (available) {
      setState(() => _isListening = true);
      _speech.listen(
        onResult: (val) => setState(() {
          text2 = val.recognizedWords;
          if (val.hasConfidenceRating && val.confidence > 0) {
            _confidence = val.confidence;
          }
```

- Then we post the text2 to flask API



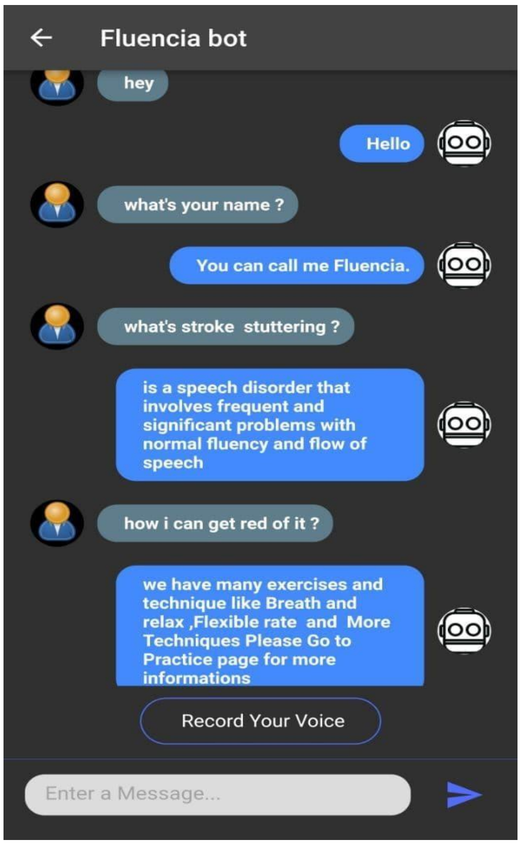
```dart
onPressed: () async {
  //_startTimer();
  _listen();

  final url = 'http://192.168.1.104:40222/voicerecorder';
  //sending a post request to the url
  final response = await http.post((Uri.parse(url)),
      body: json.encode({'text': text2}));
},
```

- Here is the flask API get the text2 and save it and pass text to our four features Grammatical Mistakes, Speed, Repetition, and Delay.



```python
@app.route("/voicerecorder", methods=["GET", "POST"])
def index():
    record = ""
    if request.method == "POST":
        request_data=request.data
        request_data=json.loads(request_data.decode('utf-8'))
        record=request_data['text']
        print("you Said : ")
        print(record)
        with open('C:/speech_app/api1/record.txt', 'w') as fp:
         #fp.write(textCorrected)
         fp.write(str(record))
        return "Done"

@app.route('/feature', methods=["GET", "POST"])
def featureresponse():

    global response

    if(request.method == 'GET'):

        response = grammaticalmistake.percentageOfBad(grammaticalmistake.mistakesCompCorrected)
        print(response)
        return jsonify({"response": response })
```

5.7 Testing


Fluencia bot

hey

Hello

what's your name ?

You can call me Fluencia.

what's stroke stuttering ?

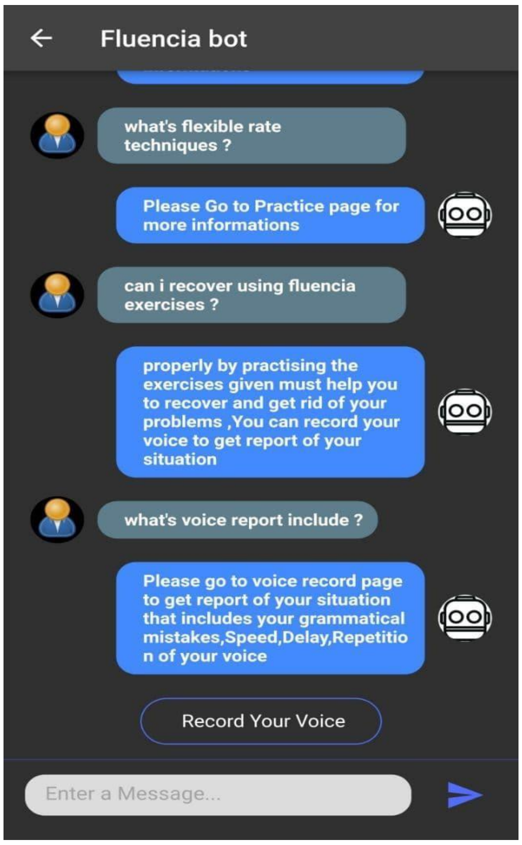
is a speech disorder that involves frequent and significant problems with normal fluency and flow of speech

how i can get red of it ?

we have many exercises and technique like Breath and relax ,Flexible rate and More Techniques Please Go to Practice page for more informations

Record Your Voice

Enter a Message...


Fluencia bot

what's flexible rate techniques ?

Please Go to Practice page for more informations

can i recover using fluencia exercises ?

properly by practising the exercises given must help you to recover and get rid of your problems ,You can record your voice to get report of your situation

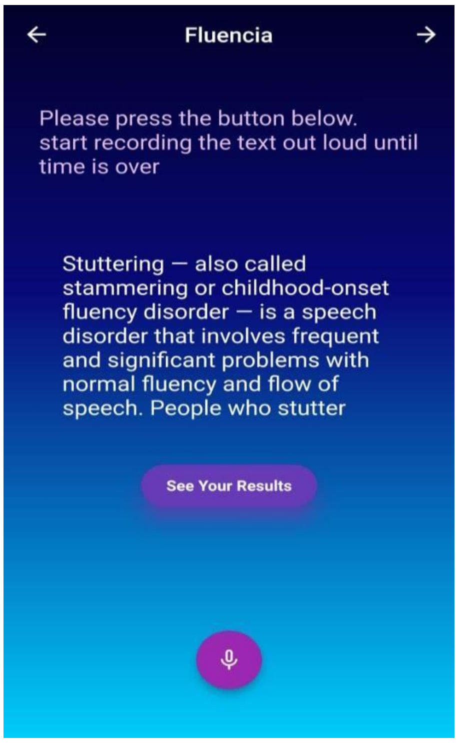
what's voice report include ?

Please go to voice record page to get report of your situation that includes your grammatical mistakes,Speed,Delay,Repetition of your voice

Record Your Voice

Enter a Message...

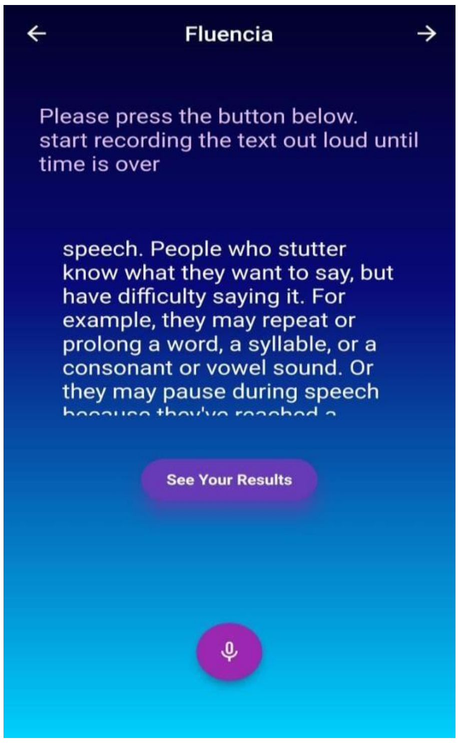
5.7.1 Voice Recording Interface


Fluencia

Please press the button below.
start recording the text out loud until time is over

Stuttering — also called stammering or childhood-onset fluency disorder — is a speech disorder that involves frequent and significant problems with normal fluency and flow of speech. People who stutter

See Your Results

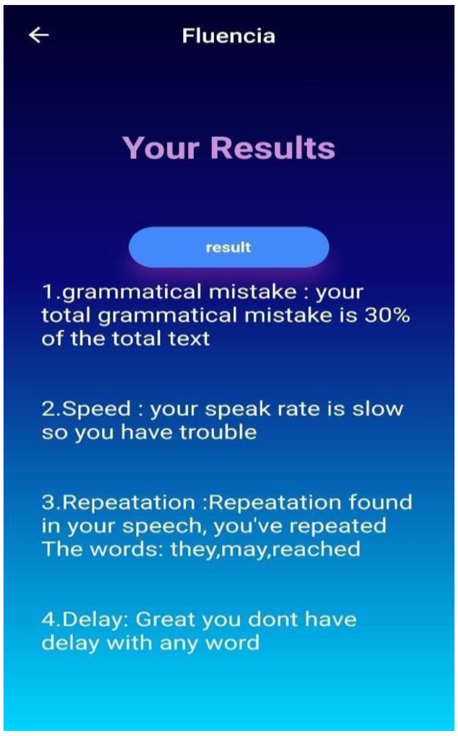

Fluencia

Please press the button below.
start recording the text out loud until time is over

speech. People who stutter know what they want to say, but have difficulty saying it. For example, they may repeat or prolong a word, a syllable, or a consonant or vowel sound. Or they may pause during speech

See Your Results

5.7.2 Result Interface


Fluencia

Your Results

result

1.grammatical mistake : your total grammatical mistake is 30% of the total text

2.Speed : your speak rate is slow so you have trouble

3.Repeatation :Repeatation found in your speech, you've repeated The words: they,may,reached

4.Delay: Great you dont have delay with any word

Chapter 6: System Analysis

6.1 Functional Requirements:

The user should be able to do the following:

1- Create an account.
2- Login to account.
3- Chat with ChatBot.
4- Ask to show exercises.
5- Submit a voice record.
6- Get a detailed report about his/her recording.
7- View suitable exercises for him/her.
8- Practice exercisers.
9- Track his/her progress.

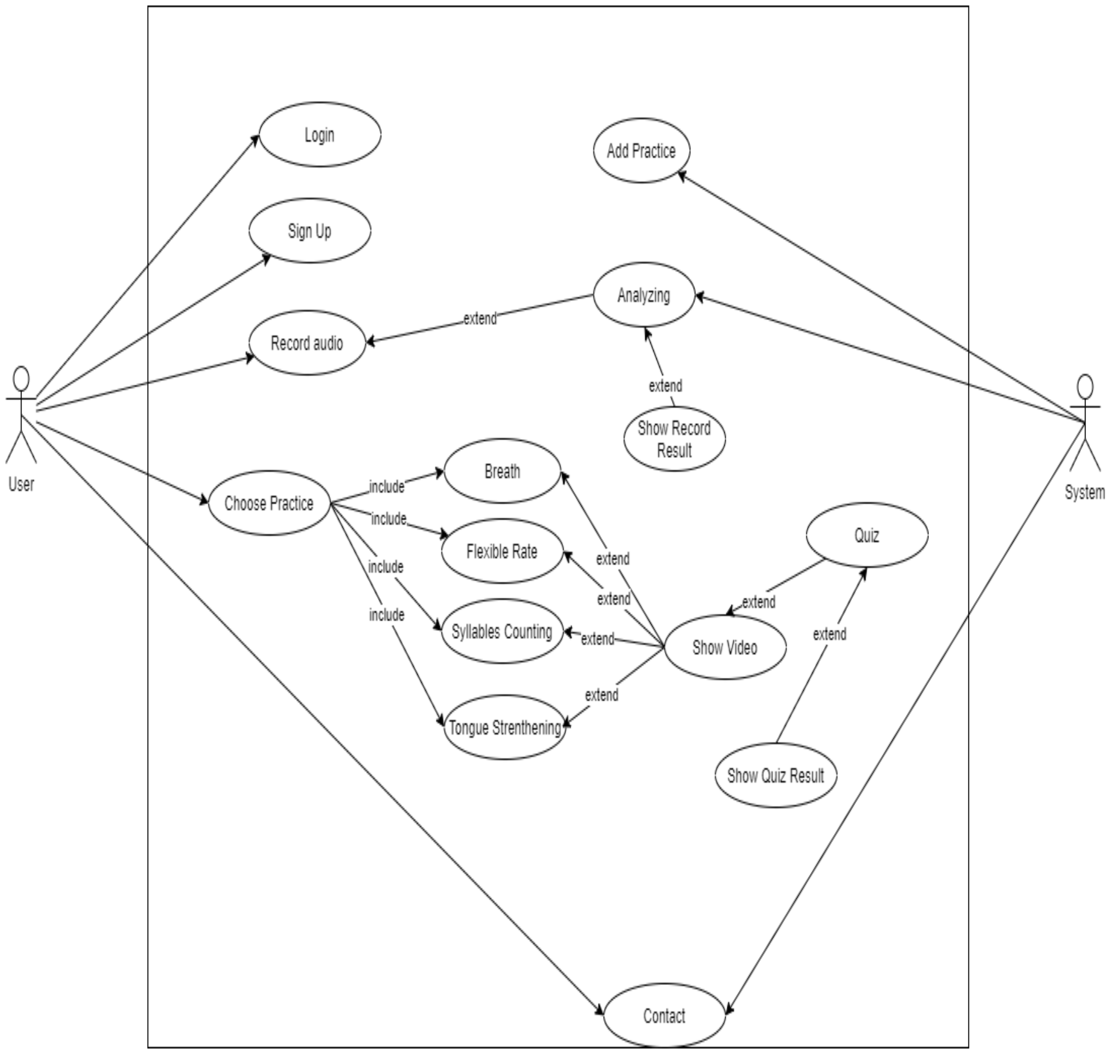
6.2 Non-Functional Requirements:

1- Security
2- Performance
3- Transparency
4- Availability

6.3 Use Cases

The use case is a list of actions or event steps typically defining the interactions between a role and a system to achieve a goal. The actor can be a human or other external system. In systems engineering, use cases are used at a higher level than within software engineering, often representing missions or stakeholder goals. The detailed requirements may then be captured in the Systems Modeling Language or as contractual statements.

After specifying the functional requirements of the application, we will represent the use case diagram that connects those functionalities to show how the flow of information within the application is done.



6.4 Use Case Specification

Use case name	**Login**	
Actor	User	
Pre-condition	Not Login	
Post-condition	The user is logged in	
The flow of events:	**User**	**System**
	1 - The user presses the login button. 3 -The user fills in his/her account in the field. 4- Submit.	2 -An interface appears containing a user account entry field. 5- The system processes user information and shows a result according to its correctness.
Critical scenario:	**Scenario**	**Response**
	1- The user entered the wrong account or does not exist	1-show error message

Use case name	**Sign Up**	
Actor	User	
Pre-condition	The user doesn’t have an account.	
Post-condition	The user is registered in the system.	
Flow of events:	**User**	**System**
	1 - The user presses the sign-up button. 3 - The user fills in his/her information in the field. 4- Submit.	2 - An interface appears containing a user information entry field. 5 - The system processes user information and shows a result according to its correctness
Critical scenario:	**Scenario**	**Response**
	1- The user entered the wrong password or does not exist	1- Show error message

Use case name	**Record Audio**	
Actor	User	
Pre-condition	User Login	
Post-condition	The user recorded the audio and save it in the system	
Flow of events:	**User**	**System**
	1 - The user presses the record button. 3 - The user speaks. 4- Submit.	2 – The system record audio. 5 - The system processes user records and shows a result according to its correctness.
Critical scenario:	**Scenario**	**Response**
	1- The user did not log in.	1- Redirect the user to the login page.

Use case name	**Choose Practice**	
Actor	User	
Pre-condition	User Login	
Post-condition	The user selects the exercise he/she wants	
Flow of events:	**User**	**System**
	1 - The user presses the practice button.	2 – An interface appears containing a practice.
Critical scenario:	**Scenario**	**Response**
	1- The user does not speak in the recording	1-show error message

Use case name	**Analyzing**	
Actor	System	
Pre-condition	User Login	
Post-condition	The audio is recorded by the user	
Flow of events:	**User**	**System**
	1 - The user is recording the voice.	2 – The system analyzes the sound and finds errors.
Critical scenario:	**Scenario**	**Response**
	1- Registration is empty.	1-show error message.

Use case name	**Show Record Result**	
Actor	User / System	
Pre-condition	User Login	
Post-condition	Audio recorder and analyzer	
Flow of events:	**User**	**System**
	1- The user records the audio. 3 - The results are shown to the user.	2 – The system analyzes the sound and shows the results.
Critical scenario:	**Scenario**	**Response**
	1- There is no registered record.	1-show error message.

Use case name	**Show Video**	
Actor	User / System	
Pre-condition	User Login	
Post-condition	The user is practicing the exercises	
Flow of events:	**User**	**System**
	1- The user enters the exercises page. 3 - The video is shown to the user.	2 – The system displays a video demonstration of the exercises.
Critical scenario:	**Scenario**	**Response**
	1- Use exercises without watching the video.	1-show error message.

Use case name	**Quiz**	
Actor	User	
Pre-condition	User login	
Post-condition	The user has practiced my exercises	
Flow of events:	**User**	**System**
	1- The user is training his/her exercises. 3 - The user makes the quiz.	2 – Give the user a quiz.
Critical scenario:	**Scenario**	**Response**
	1- The user is not trained.	1-Show error message.

Use case name	**Show Quiz Result**	
Actor	User	
Pre-condition	User Login	
Post-condition	The user did the test.	
Flow of events:	**User**	**System**
	1- The user makes the quiz 3 - The result is shown to the user	2 – The system analyzes the quiz and shows the results
Critical scenario:	**Scenario**	**Response**
	1- The user is not advanced to the quiz.	1-Not showing results.

Chapter 7: System Design

7.1 Collaboration Diagram

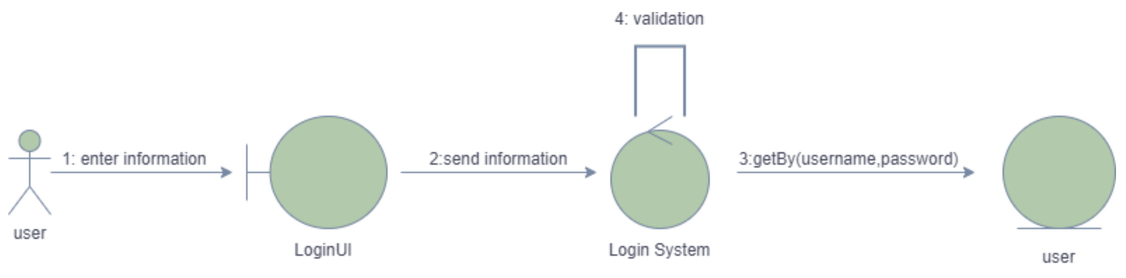


Figure 1.1 : User Login

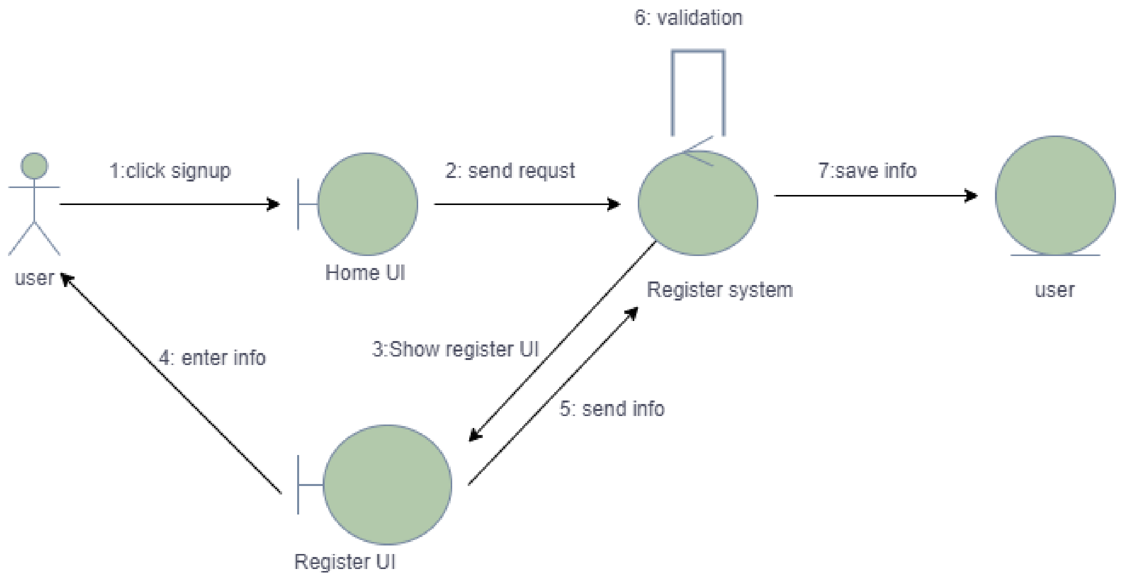


Figure 1.2 : User Signup

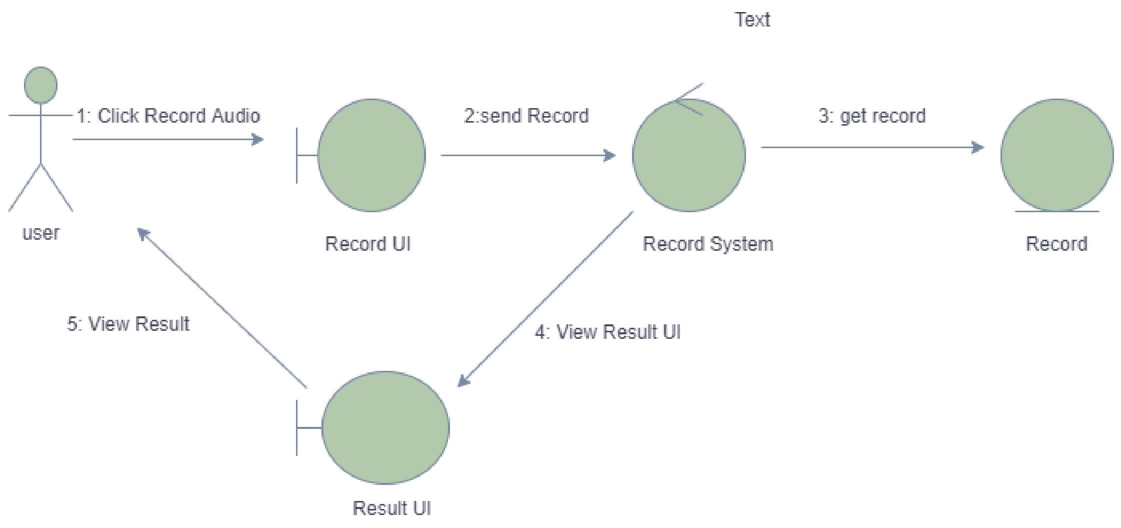


Figure 1.3 : Record Audio

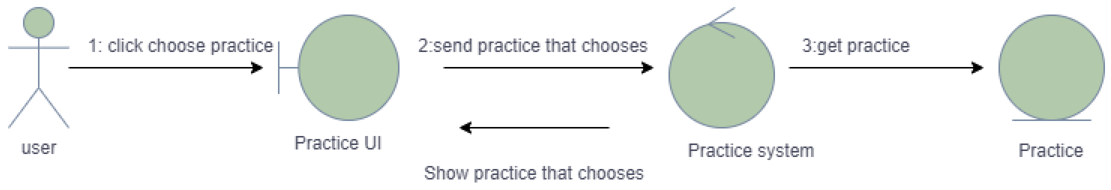
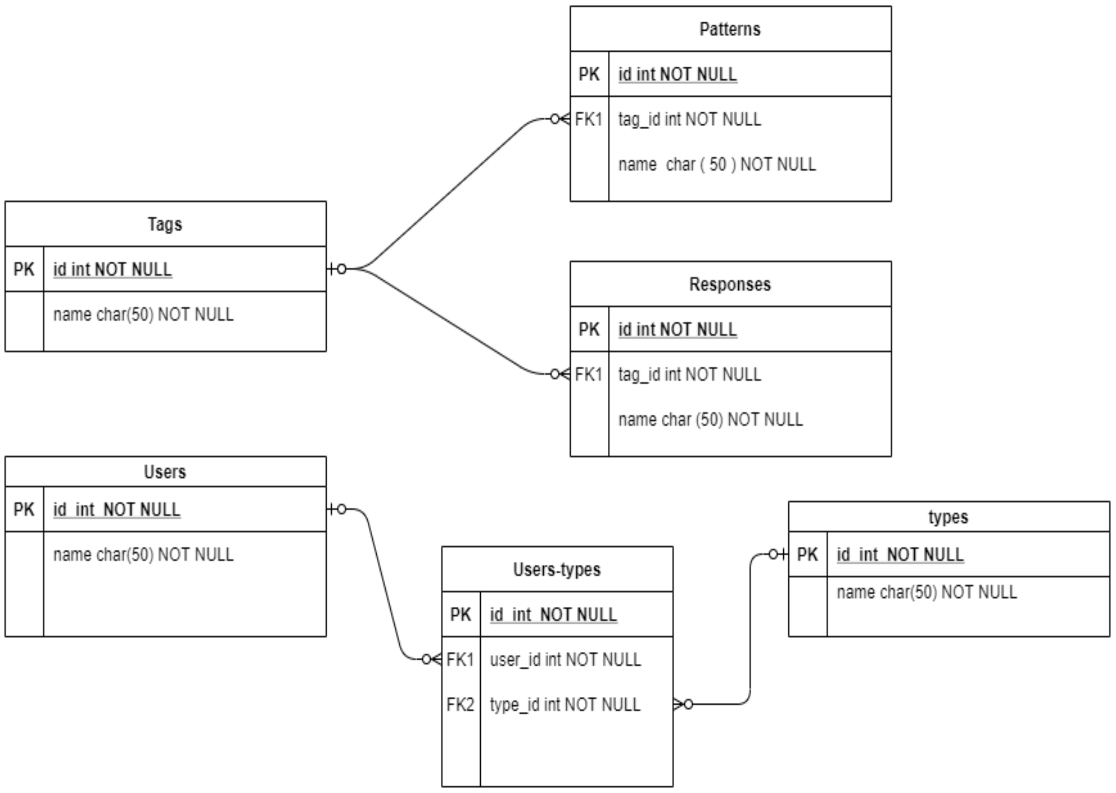


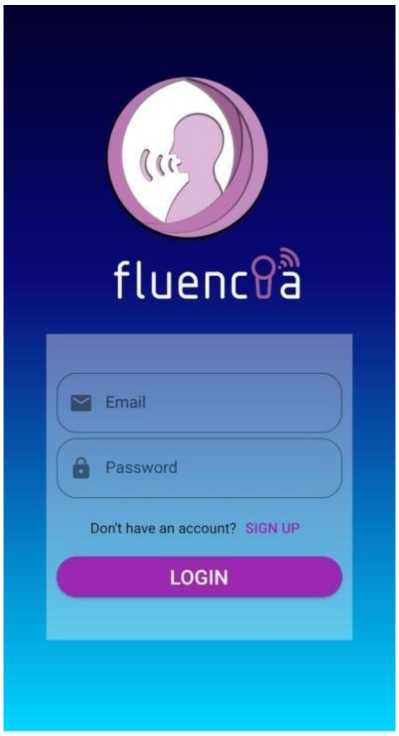
Figure 1.4 : Choose Practice

7.2 Entity Relationship Diagram

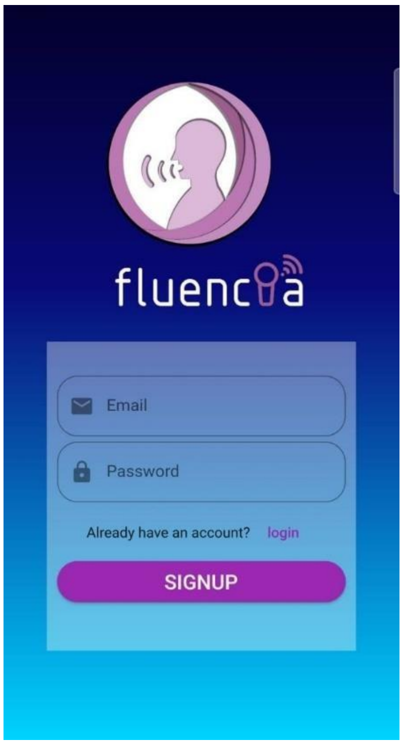


7.3 App Interfaces Design

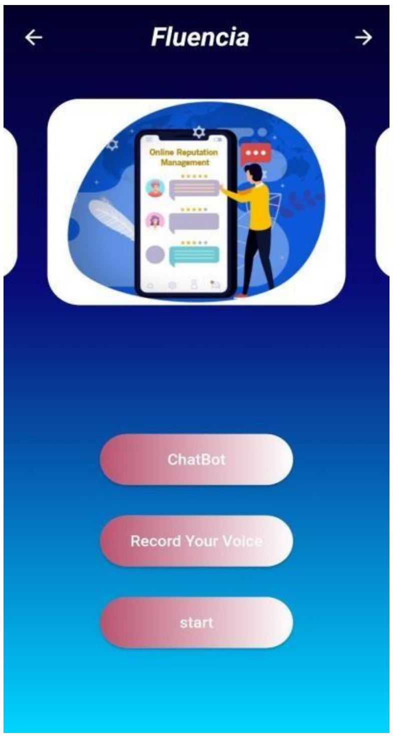
7.3.1 Login interface



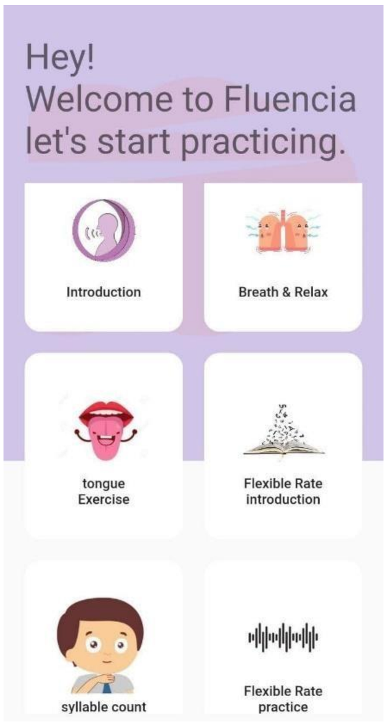
7.3.2 Sign-up interface



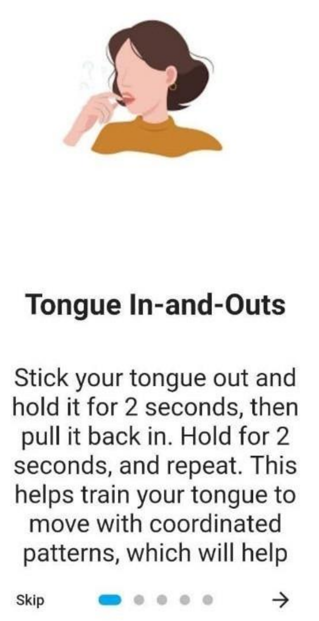
7.3.3 Main interface



7.3.4 Exercises interface

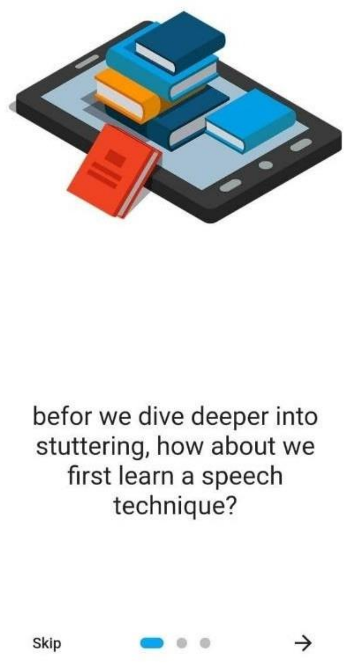


7.3.5 Tongue Exercise interface

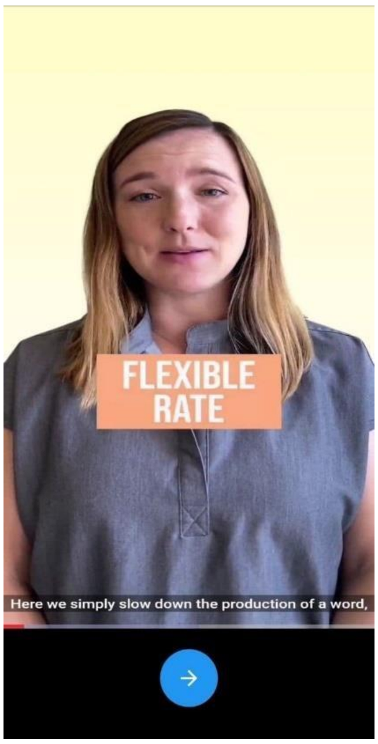

Tongue In-and-Outs

Stick your tongue out and hold it for 2 seconds, then pull it back in. Hold for 2 seconds, and repeat. This helps train your tongue to move with coordinated patterns, which will help

7.3.6 Offering an Exercise Interface

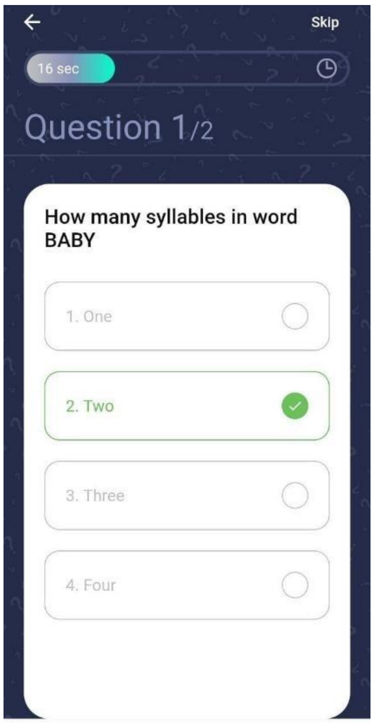

befor we dive deeper into stuttering, how about we first learn a speech technique?

7.3.7 Flexible Rate Exercise Interface



Here we simply slow down the production of a word,

7.3.8 Flexible Rate Quiz Interface



Question 1/2

How many syllables in word BABY

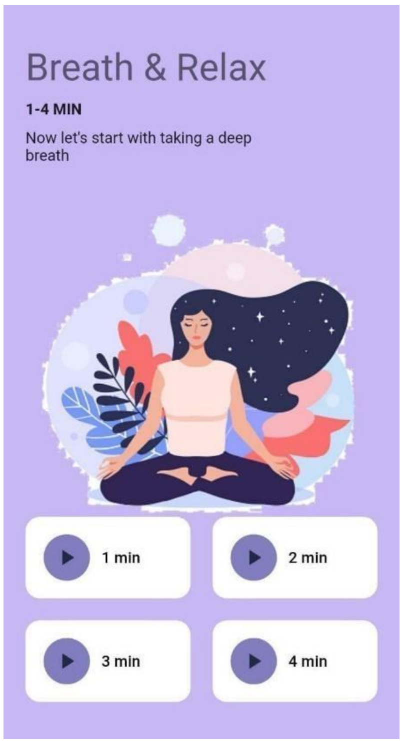
1. One

[✓ 2. Two]

3. Three

4. Four

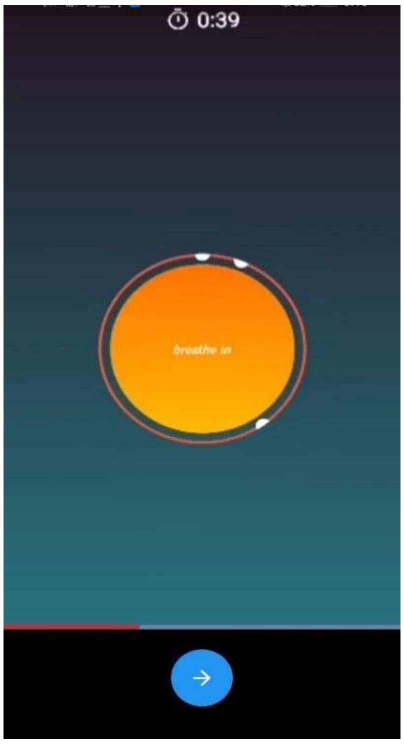
7.3.9 Breath Control Exercise Interface


Breath & Relax

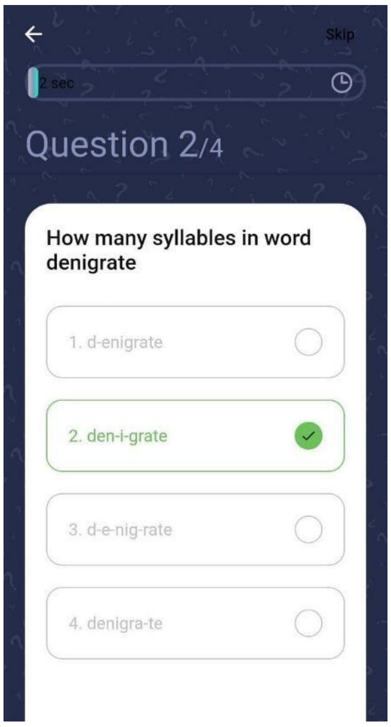
1-4 MIN

Now let's start with taking a deep breath

7.3.10 Breath Control Guidance Interface



7.3.11 Word Syllables Exercise Interface



Question 2/4

How many syllables in word denigrate

1. d-enigrate
[✓ 2. den-i-grate]
3. d-e-nig-rate
4. denigra-te

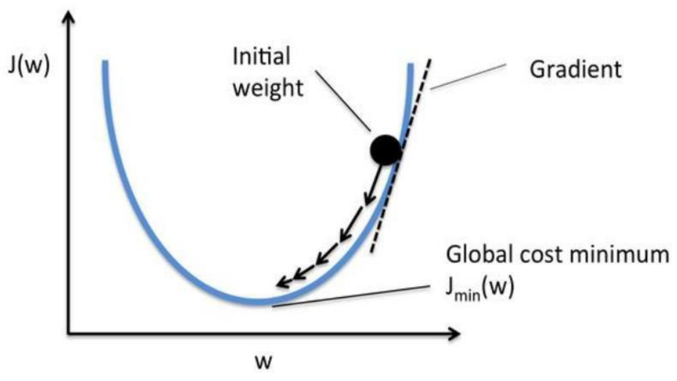
7.4 Algorithms

The system provides a facility to improve the pronunciation and speech of the trainee. The mobile application will help and guide him/her to achieve fluency. Using the ChatBot, practicing exercises and tracking his/her progress.

After getting a voice recording from the user, the system will convert his/her speech to a text, then the text will be analyzed and results will be shown to the user. On the user’s demand, exercises will be given to practice. Finishing one after another the user can record his/her voice again to see whether he improved or not, so he can decide whether to practice more or not.

7.4.1 Stochastic gradient descent (SGD)

Approximates the gradient using only one data point. Therefore, evaluating the gradient saves a lot of time compared to summing the overall data.
The gradient of the cost function will be calculated not for all elements in the sample, as it is done with the traditional gradient descent method, but for each element separately. The gradient calculated for a particular element is taken as an approximation of the real gradient. Weights in the model are recalculated by the calculated gradient for one element, which leads to the fact that the model is adjusted when moving from each successive element of the sample to the next.



In normal GD we take a step every time we go over the whole data but with SGD, we take a step based on every data line (we update the weights after each training sample) Using this introduces problems it can get stuck in flat areas or bounce around if the objective function returns noisy gradients.

- **Nesterov momentum:**

  Momentum is an approach that accelerates the progress of the search to skim across flat areas and smooth out bouncy gradients.
  The problem with Momentum is that it can overshoot because it uses the last gradient value as momentum.

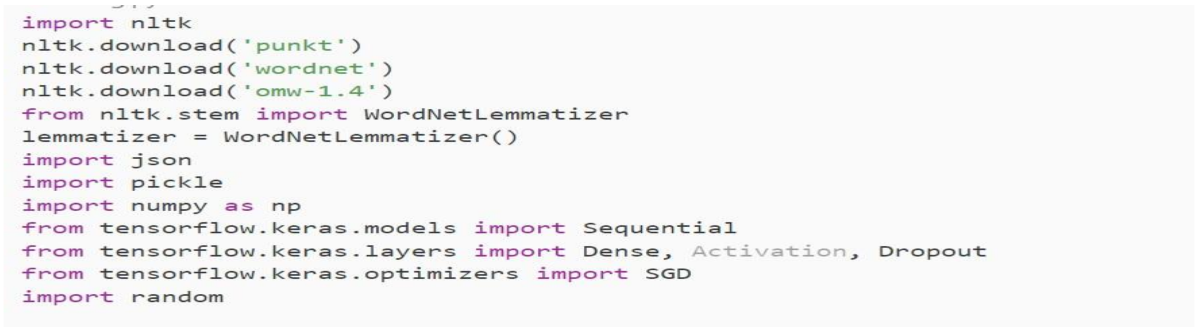
Nesterov momentum is an extension of momentum that involves calculating the decaying moving average of the gradients of projected positions in the search space rather than the actual positions themselves.
This has the effect of harnessing the accelerating benefits of momentum whilst allowing the search to slow down when approaching the optima and reducing the likelihood of missing or overshooting it.

- **SGD classification:**

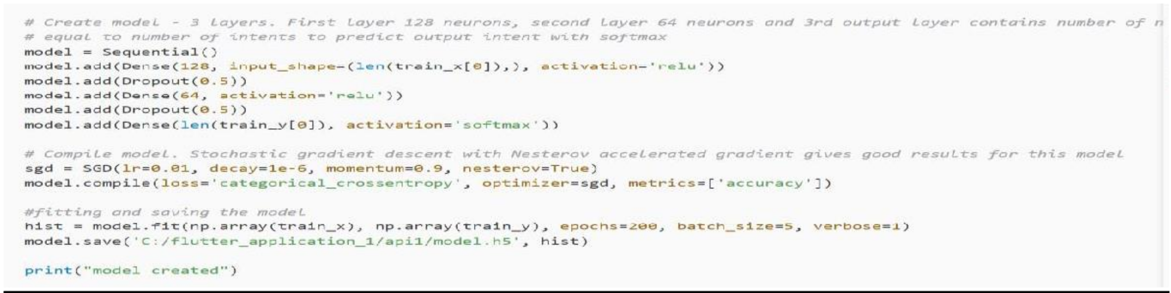
The class SGDClassifier implements a plain stochastic gradient descent learning routine that supports different loss functions and penalties for classification. Below is the decision boundary of an SGDClassifier trained with the hinge loss, equivalent to a linear SVM.

- **SGD Tips on Practical Use:**

Stochastic Gradient Descent is sensitive to feature scaling, so it is highly recommended to scale your data. For example, scale each attribute on the input vector X to [0, 1] or [-1, +1], or standardize it to have mean 0 and variance note that the same scaling must be applied to the test vector to obtain meaningful results. This can be easily done using Keras. optimizers.



```
import nltk
nltk.download('punkt')
nltk.download('wordnet')
nltk.download('omw-1.4')
from nltk.stem import WordNetLemmatizer
lemmatizer = WordNetLemmatizer()
import json
import pickle
import numpy as np
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Activation, Dropout
from tensorflow.keras.optimizers import SGD
import random
```



```
# Create model - 3 layers. First layer 128 neurons, second layer 64 neurons and 3rd output layer contains number of n
# equal to number of intents to predict output intent with softmax
model = Sequential()
model.add(Dense(128, input_shape=(len(train_x[0]),), activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(64, activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(len(train_y[0]), activation='softmax'))

# Compile model. Stochastic gradient descent with Nesterov accelerated gradient gives good results for this model
sgd = SGD(lr=0.01, decay=1e-6, momentum=0.9, nesterov=True)
model.compile(loss='categorical_crossentropy', optimizer=sgd, metrics=['accuracy'])

#fitting and saving the model
hist = model.fit(np.array(train_x), np.array(train_y), epochs=200, batch_size=5, verbose=1)
model.save('C:/flutter_application_1/api1/model.h5', hist)

print("model created")
```

Chapter 8: Tools and Environment

8.1 Flutter

Flutter is a free and open-source mobile UI framework created by Google and released in May 2017. In a few words, it allows you to create a native mobile application with only one codebase. This means that you can use one programming language and one codebase to create two different apps (for iOS and Android).

Flutter consists of two important parts:

An SDK (Software Development Kit): A collection of tools that are going to help you develop your applications. This includes tools to compile your code into native machine code (code for iOS and Android).

A Framework (UI Library based on widgets): A collection of reusable UI elements (buttons, text inputs, sliders, and so on) that you can personalize for your own needs.

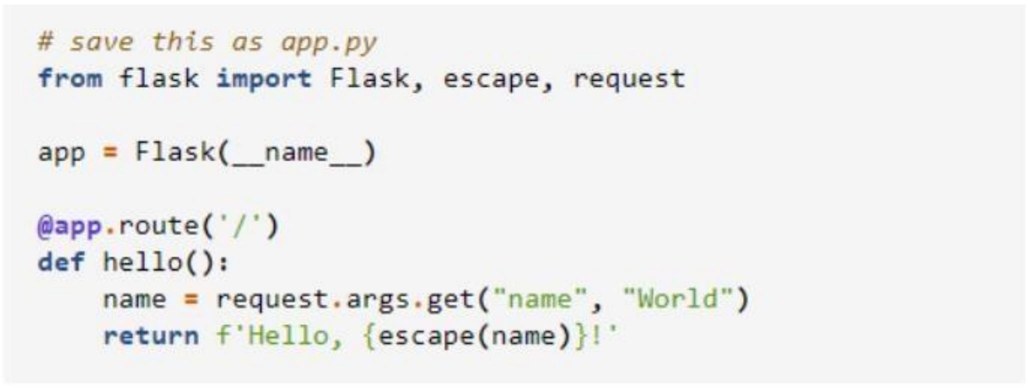
To develop with Flutter, you will use a programming language called Dart. The language was created by Google in October 2011, but it has improved a lot over these past years.

Dart focuses on front-end development, and you can use it to create mobile and web applications.

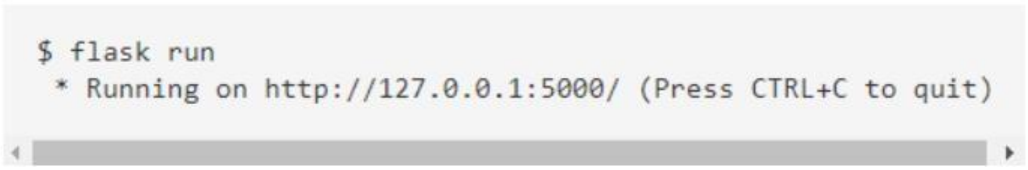
8.2 Flask

Flask is a lightweight WSGI web application framework. It is designed to make getting started quick and easy, with the ability to scale up to complex applications. It began as a simple wrapper around Werkzeug and Jinja and has become one of the most popular Python web application frameworks.

Flask offers suggestions but doesn't enforce any dependencies or project layout. It is up to the developer to choose the tools and libraries they want to use. There are many extensions provided by the community that makes adding new functionality easy.



```python
# save this as app.py
from flask import Flask, escape, request

app = Flask(__name__)

@app.route('/')
def hello():
    name = request.args.get("name", "World")
    return f'Hello, {escape(name)}!'
```



```
$ flask run
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

8.3 Python

PYTHON

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. It’s a high-level built-in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components. Python's simple, easy-to-learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms and can be freely distributed.

Often, programmers fall in love with Python because of the increased productivity it provides. Since there is no compilation step, the edit-test-debug cycle is incredibly fast. Debugging Python programs is easy: a bug or bad input will never cause a segmentation fault. Instead, when the interpreter discovers an error, it raises an exception. When the program doesn't catch the exception, the interpreter prints a stack trace. A source-level debugger allows inspection of local and global variables, evaluation of arbitrary expressions, setting breakpoints, stepping through the code a line at a time, and so on. The debugger is written in Python itself, testifying to Python's introspective power. On the other hand, often the quickest way to debug a program is to add a few print statements to the source: the fast edit-test-debug cycle makes this simple approach very effective

8.4 Anaconda

Anaconda is a free and open-source distribution of the programming languages Python and R (check out these Python online courses and R programming courses). The distribution comes with the Python interpreter and various packages related to machine learning and data science.

The idea behind Anaconda is to make it easy for people interested in those fields to install all (or most) of the packages needed with a single installation.

8.5 Postman

Postman is an API Development Environment that helps people to build, test, document, monitor, and publish documentation for their APIs.

Some of the main features of Postman are:

- Sending requests (with support for different authentication schemes, Cookies, certificates, headers, query parameters, request body, and SOAP with/without WSDL) and debugging and saving responses.
- Organizing your APIs into groups called Collections.

- Writing tests. The test scripts can run pre-request, after a response has been received, and can have looping and branching concepts.

8.6 Tensorflow

TensorFlow is an open-source software library for dataflow programming across a range of tasks. It is a symbolic math library and is also used for machine learning applications such as neural networks. It is used for both research and production at Google, often replacing its closed-source predecessor, Disbelief. Tensor Flow was developed by the Google Brain team for internal Google use. It was released under the Apache 2.0 open source license on November 9, 2015.

8.7 Keras

Keras is an open-source neural network library written in Python. It is capable of running on top of TensorFlow, Microsoft Cognitive Toolkit, Theano, or MXNet. Designed to enable fast experimentation with deep neural networks, it focuses on being user-friendly, modular, and extensible. It was developed as part of the research effort of project ONEIROS (Open-ended Neuro-Electronic Intelligent Robot Operating System), and its primary author and maintainer are François Cholet, a Google engineer. In 2017, Google's TensorFlow team decided to support Keras in TensorFlow's core library. Chollet explained that Keras was conceived to be an interface rather than a standalone machine-learning framework. It offers a higher-level, more intuitive set of abstractions that make it easy to develop deep learning models regardless of the computational backend used.

8.8 NLTK

The Natural Language Toolkit, or more commonly NLTK, is a suite of libraries and programs for symbolic and statistical natural language processing (NLP) for English written in the Python programming language. It was developed by Steven Bird and Edward Loper in the Department of Computer and Information Science at the University of Pennsylvania. NLTK includes graphical demonstrations and sample data. It is accompanied by a book that explains the underlying concepts behind the language processing tasks supported by the toolkit, plus a cookbook.

8.8 Threading in python

Threading in python is used to run multiple threads (tasks, function calls) at the same time. Note that this does not mean that they are executed on different CPUs.Python threads will NOT make your program faster if it already uses 100 % CPU time.

8.9 NumPy

NumPy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays. The ancestor of NumPy, Numeric, was created by Jim Hugunin with contributions from several other developers. In 2005, Travis Oliphant created NumPy by incorporating features of the competing Numarray into Numeric, with extensive modifications. NumPy is open-source software and has many contributors.

8.10 JSON

JSON (JavaScript Object Notation) is a minimal, readable format for structuring data. It is used primarily to transmit data between a server and web application, as an alternative to XML.

Chapter 9: Conclusion

9.1 Summary

The project objective is to get a voice record to specify the kind of problem the patient has in his/her way of talking, later we can offer him/her exercises based on his/her kind of problem.

Training and exercises are offered by the ChatBot then, on the patient's demand. The ChatBot can show him/her and guide him/her to use these exercises each day until he/she finishes the required exercises that should allow him/her to get over his/her troubles. In creating the ChatBot, deep learning techniques were used so that the ChatBot can answer any question from a medical trainee on a particular case.

9.2 Difficulties

- Learning about the syndrome and how it can be cured.
- The need to create a dataset to train the ChatBot.
- Send the recording from the front end to the back end.

9.3 Future Vision

There is a scope for further development in our project to a great extent, what we have planned for so far are the following:

- Add more exercises that can help the patient.
- Support more languages.
- Improve progress tracking.
- Switch to a speech-to-speech ChatBot.
- Add premium features that can be unlocked by committing to exercise daily.
- Improve speech analysis and give a better-detailed report.

References

Online Resources

[1] https://www.udemy.com/course/applied-deep-learning-build-a-chatbot-theory-application/learn/lecture/13462282#overview

[2] https://stackabuse.com/levenshtein-distance-and-text-similarity-in-python/

[3] https://holianh.github.io/portfolio/Cach-tinh-WER/

[4] https://stackabuse.com/spelling-correction-in-python-with-textblob/

[5] https://python.plainenglish.io/create-a-deep-learning-chatbot-with-python-and-flask-d75396a4382a

[6] https://virtualspeech.com/blog/average-speaking-rate-words-per-minute

[7] https://medicalfuturist.com/top-12-health-chatbots/

[8] https://en.wikipedia.org/wiki/Matplotlib

[9] https://en.wikipedia.org/wiki/TensorFlow

[10] https://en.wikipedia.org/wiki/numpy

[11] https://en.wikipedia.org/wiki/scikit-learn

[12] https://en.wikipedia.org/wiki/keras

[13] https://en.wikipedia.org/wiki/python

[14] https://en.wikipedia.org/wiki/pandas

[15] https://en.wikipedia.org/wiki/Glob_(programming)

[16] https://en.wikibooks.org/wiki/Python_Programming/Threading

[17] https://en.wikipedia.org/wiki/Natural_Language_Toolkit

[18] https://en.wikipedia.org/wiki/Word2vec