AIGS 1003 MACHINE LEARNING

“Say My Name Project”

Group 1

Members:

Olabode Aremu

Sonali Jammichedu

Vennila

Neha Parveen Mohammed

Sana Naseem

Loyalist College, Belleville, Canada, 2023

Tables of Contents

Introduction

Our names are usually linked with our personal identities, often chosen with care and laden with meaning, although for many students and staff their college experience is mostly spent in correcting people to pronounce their name properly or they feel compelled to adapt their name or adopt a new name entirely so that they can fit in into it.

"Say My Name" application aims to bridge the gap to the above problem and provides a solution which would be useful for students and staff to correctly pronounce the names. This application takes username as input and returns with a human readable phonetic/ syllable along with the pronunciation of the user's name and ask's for the feedback.

Literature Review

Learning how to pronounce names correctly can be very challenging most of the times. Proper pronunciation of names is very important to have good meaning and correct syllables. The right pronunciation of names can help in achieving success in socializing with people. This process can be simplified through different tools.

Many applications do exist out there that currently help pronounce user's name correctly, but we have chosen to come with another one along with good features that makes it better than what currently exists. We came across a research paper on "Say My Name!" developed by researchers based in Brazil [1]. Basically, they have used three algorithms Soundex [5], Metaphone [6] and NYSIIS [7], which takes in an input and returns a code of phonetic encoding. Soundex algorithm allows to raise the level of similarity between the texts in SMS and their corresponding text in English or any other language, whereas NYSIIS algorithm is an alphabetic algorithm which is easy to implement, and which yields canonical index code like Soundex. However, NYSIIS differs from Soundex in that it retains information about the position of vowels in the encoded word by converting all vowels to the letter A [8]. To determine the complex phonetic names, they calculated the phonetic distance between the letters using the Jaro distance and the match rate index. Word complexity measure is also used to analyse the complexity of the name.

The application that we are working on will come up with good phonetics transcription to represent names in a way that captures their pronunciation. The text to speech library-Speech Module in Angular can handle different languages, accents, and variations in pronunciation, Natural Language Processing techniques and libraries to improve the accuracy of name pronunciation. We are also addressing user experience and accessibility which is addressing cultural sensitivity and inclusivity in pronunciation. An intriguing and user-friendly interface that initially captures user text and voice inputs, ultimately transcribing them into human-readable phonetics.

Objectives

1. To develop a user-friendly web application that accurately captures and transcribes user's names in human-readable format by taking student ID and username as an input.
2. To implement a phonetics-based algorithm to analyze and interpret different types of names to ensure correct human readable pronunciation. Ex. Carol [kar-uhl]
3. Provide an interactive platform where users may enter their names along with the student ID and get precise phonetic translations.
4. Capture user's voice to verify the propriety of the human readable phonetic spelling generated by the library (i.e. G2p) (upcoming feature).
5. Making sure proper flow and functional working of the website is achieved by handling different scenarios using different Exception Handling methods.
6. Verify the application's correctness and inclusivity by testing it with a variety of names.

Technologies and Libraries Used

Tool used for frontend and backend development: Visual Studio Code

Backend development: Python Language

Frontend development: Angular, Typescript, JavaScript, HTML/HTML5 (Hyper Text Markup Language), SCSS (Sassy Cascading Style Sheet), RxJs (Reactive Extensions for JavaScript), Bootstrap

Framework for Integrating backend with frontend: Flask – micro web framework.

Database: Using Excel (.csv) to store and retrieve data.

Libraries Used

Backend

Library	Usage
openpyxl	Helps to read and write excel files
G2p	Grapheme-to-Phoneme is a function that generates pronunciations (phonemes) for words

Flask	Integration of backend with frontend
Flask_cors	Flask extension for handling Cross Origin Resource Sharing (CORS)
Flask_restful	Extension by flask for building REST (Representational State Transfer) API's (Application Program Interface).
Re (Regex pattern)	To check whether a sequence of characters form a particular search pattern (In Say My Name project used for Exceptional Handling)
json	Syntax for storing and exchanging data. Text, written with JavaScript object notation.

Backend: Libraries Explored

Library	Usage
pyttsx3	It is a python text-to-speech conversion library
eng_to_ipa	This python library helps us to convert English text into International Phonetic Alphabet (IPA) which is a system of symbols that represents the speech

Frontend

Library	Usage
RecordRTC	Captures the user's voice and in the form of an audio file
ToastrService	Used to apply toaster messages based on user action
HttpClient	Performs hyper text transfer protocol (http) requests
NgxUiLoaderModule	Used to apply a loader to the application, in scenarios such as late API response.
MatIconModule	Provides wide range of icons, compatible with angular.
MatCardModule	Provides card styling to the User Interface
SpeechModule	Used for speech synthesis, coverts text to speech
BrowserAnimationModule	Enables animation in Angular application. Provides set of function and decorators to create and control animations as well as browser specific implementation of the Web animations API

RxJs	Provides subjects, observables, and other methods to handle asynchronous or callback-based code

Technical Analysis

1. **Testing Data:** For this project, we have some testing data which is already stored in the Excel which we are using to implement the logic for generating human-readable phonetics. Besides this an excel file was provided by the college to test the given application.

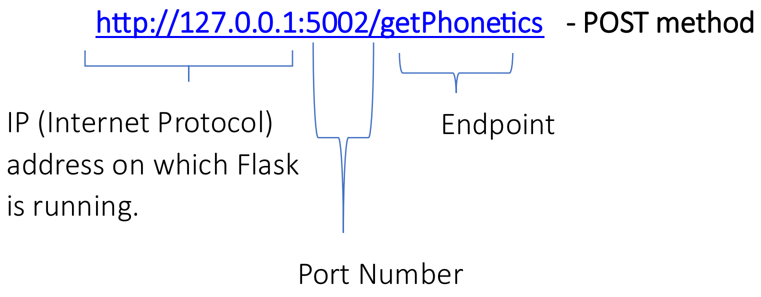
2**. Layout:** For the layout and the flow of the application please refer to the GitHub repository link which is provided in the reference section or the topic following this. (References point 4)

3. **Technologies Used:** For backend we have used python, which is quite scalable language, expansive and ready-to-use library collection makes easier to develop any web application. For the frontend we used Angular, Typescript, JavaScript, HTML/HTML5 (Hyper Text Transfer Protocol), SCSS (Sassy Cascading Style Sheet), RxJs (Reactive Extensions for JavaScript), Bootstrap. Visual Studio Code (VSCode) is lightweight, fast, open-source and cross-platform, which makes it quite popular IDE to use.

4. **Scalability:** This application is quite scalable in terms of adding new functionalities and technology. It aims to be compatible while integrating/embedding with any other application or product.

Application Program Interface (API) Details

API used to interact python with Angular – to get the human-readable phonetics from backend:



http://127.0.0.1:5002/getPhonetics **- POST method**

IP (Internet Protocol)
address on which Flask
is running.

Endpoint

Port Number

The API requires request object to get the phonetics from the backend to return with a response object. The structure of request and response is given below:

requestObject:

```
{
“studentID”: “500226345”,
“studentName”: “Joseph”
}
```

responseObject:

```
{
“phoneticsSpelling”: “jo-se-ph”,
“status”: “success”,
“message”: “success”,
}
```

API used to get the feedback from the user and send it to the backend:

http://127.0.0.1:5002/userFeedback- POST Method

requestObject:

```
{
“studentID”: “500226345”,
“studentName”: “Joseph”
“feedback”:”1”
}
```

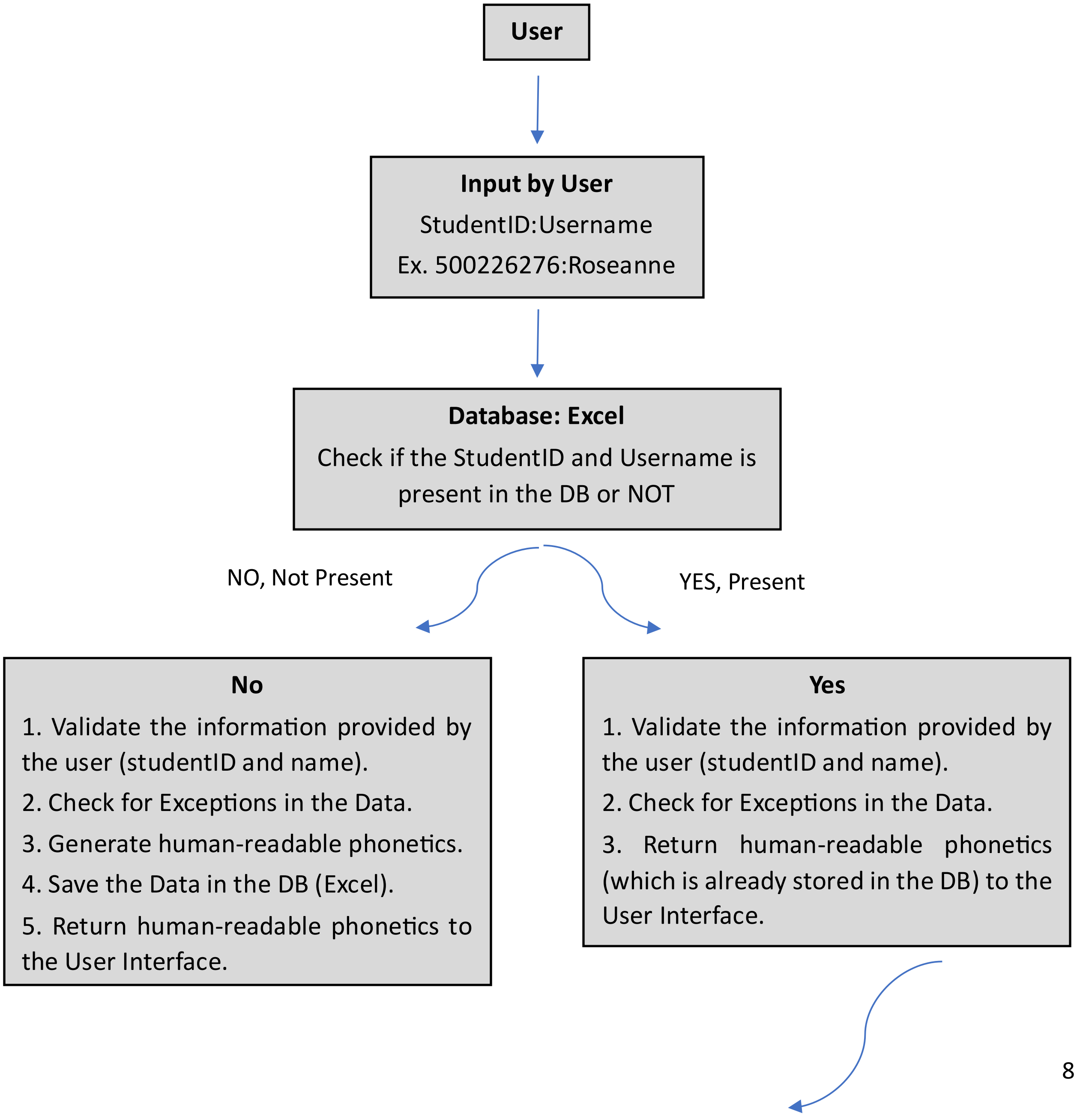
responseObject:

```
{
“status”: “success”,
"message": "success",

}
```

Frontend is running on http://localhost:4200/

Application Flow



User

Input by User

StudentID:Username

Ex. 500226276:Roseanne

Database: Excel

Check if the StudentID and Username is present in the DB or NOT

NO, Not Present

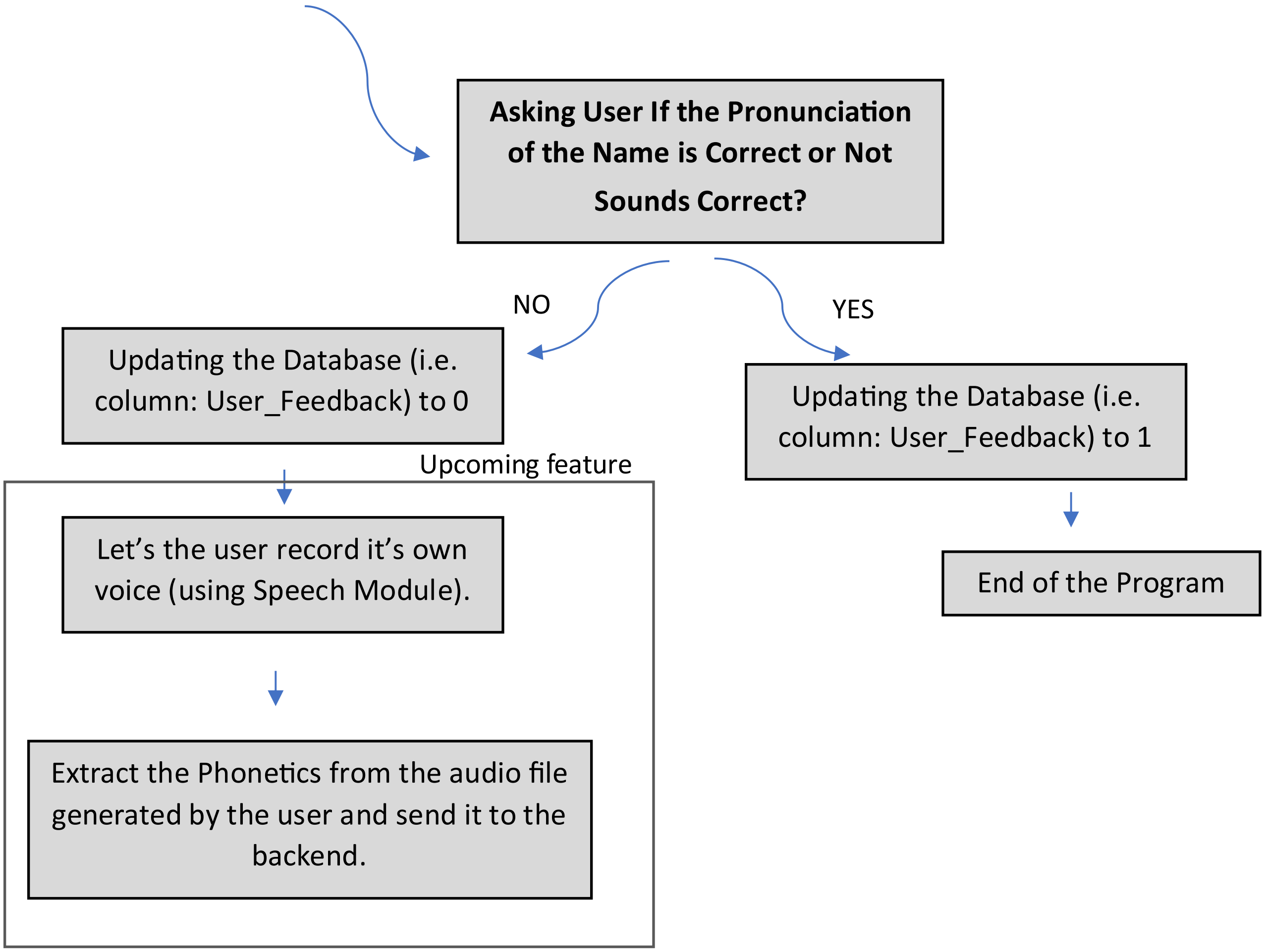
YES, Present

No

1. Validate the information provided by the user (studentID and name).
2. Check for Exceptions in the Data.
3. Generate human-readable phonetics.
4. Save the Data in the DB (Excel).
5. Return human-readable phonetics to the User Interface.

Yes

1. Validate the information provided by the user (studentID and name).
2. Check for Exceptions in the Data.
3. Return human-readable phonetics (which is already stored in the DB) to the User Interface.


Asking User If the Pronunciation of the Name is Correct or Not Sounds Correct?
NO
YES
Updating the Database (i.e. column: User_Feedback) to 0
Updating the Database (i.e. column: User_Feedback) to 1
Upcoming feature
Let's the user record it's own voice (using Speech Module).
End of the Program
Extract the Phonetics from the audio file generated by the user and send it to the backend.

__Explanation__

Converting the text to transcription and text to speech:

Considering the excel as the Database which has the columns such as "Student name, Student_ID", we have developed the code using "openpyxl" to read and manipulate the excel, in such a way that, if the user inputs their Student ID along with the Name in the application, the program fetches the database by iterating over each row until it finds the match. Once it finds the matching Student ID and Name, it will give us output in the form of human-readable phonetics. Ex. Carol [kar-uhl]

Updating the Database for the new user:

The program keeps iterating over the rows until it reaches the maximum number of rows and if it doesn't find the matching name in the database, it will validate the information provided by the user, generates the transcription, and will update the database. Once the database is updated it will send the human-readable phonetics to the frontend.

Saving the transcribed name:

While fetching for the student ID and user's name in the Database, parallelly it will also save the transcribed name in the database for the respective user.

Exception Handling:

Once the student ID and username is passed to the backend via the API, the exception handling method in python will check if the student ID consist of any other character other than digits, if yes, then it will throw an error by sending a failure message to the User Interface. It will also check for the username whether it consists of any other character other than capital and small alphabets.

Frontend:

We have used "TypeScript" to develop the Frontend of the application. Once we run the application, a text box appears on the screen asking the user to input their student Id and name separated by a colon (':'). Once the user inputs their information, the application displays the phonetic spelling of the name along with the audio of how the name will be pronounced.

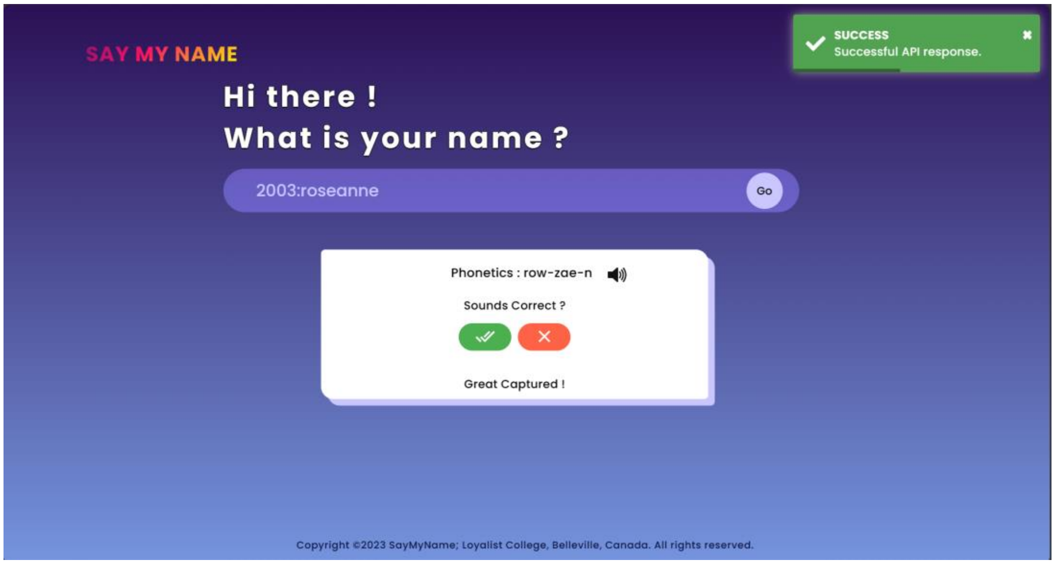
Then an option appears on the screen asking if the user is satisfied by how their name is pronounced. If the user is satisfied, then they should click on the "Tick mark" button and the program ends there. If the user is not satisfied, then they should click on the "Incorrect" button and an option will be displayed on the screen asking if the user can input their name by saying it out (record their voice) to the application. Once they click on the button, the user needs to say their name out load and clear. Once they are finished, they need to stop the recording. The user can hear the recorded audio and if not satisfied with the pronunciations, it can re-record its voice until the user is satisfied.

Exceptional Handling in Frontend:

Validation of student ID and username is taken care off, from the frontend by comparing the student ID with appropriate regex pattern to test whether the student Id contains any other character other than numbers. The username is being compared to a regex pattern which will check for errors other than capital and small alphabets only.

User Interface

The below image shows the user interface of Say My Name which has been developed so far.



SUCCESS
Successful API response.

Hi there !
What is your name ?

2003:roseanne

Phonetics : row-zae-n

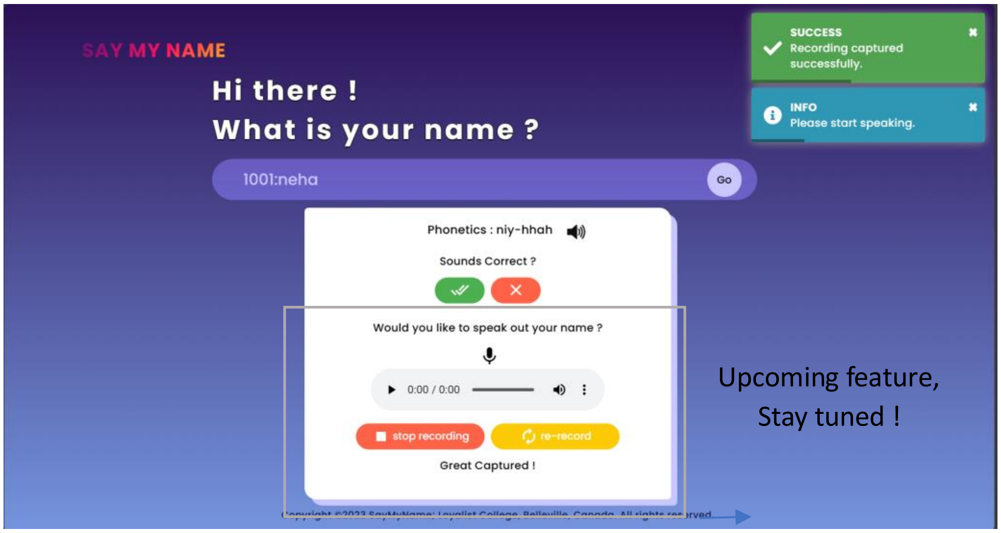
Sounds Correct ?

Great Captured !

Copyright ©2023 SayMyName; Loyalist College, Belleville, Canada. All rights reserved.

Upcoming Feature

In the next phase of the project, we aim to implement a functionality where, if the user is not satisfied with the human-readable phonetics and the pronunciation, user has an option to record his own voice such that we will be extracting the phonetics from that to satisfy the pronunciation which user has input. We also plan to give an option to re-record the voice until the user is satisfied.



SUCCESS
Recording captured successfully.

INFO
Please start speaking.

Hi there !
What is your name ?

1001:neha

Phonetics : niy-hhah

Sounds Correct ?

Would you like to speak out your name ?

Great Captured !

Copyright ©2023 SayMyName; Loyalist College, Belleville, Canada. All rights reserved.

Upcoming feature,
Stay tuned !

Challenges

Frontend:

- After having worked on User Interface we have arrived at two-point approach in developing User Interface.
    1. Manual Typing
    2. Voice Command (Work in Progress)
- Faced some technical issues while developing the methods mentioned above. Could not find proper library for speech synthesis which could accurately give the pronunciation of human-readable phonetics.
- Finding the right set of libraries to capture user's voice from the frontend was a major task. Finally, we ended up using RecordRTC to capture user's voice from the frontend.

Backend:

In python we have achieved in getting the phonetics correctly i.e., the transcription part, but instead it was expected to give a human readable syllable. After research and development on using different libraries, we implemented using ipa and g2p which does not give the exact syllable spelling (Work in Progress, Approach – We will try to make a direct call to dictionary.com which gives the exact human-readable phonetics).

Integration:

To find the exact framework and compatibility of python with angular for integration was a major task. We found a framework called Flask, which is compatible with python as well as angular but, to interact with Application Programming Interface (API) was difficult task to achieve.

Conclusion

Through this project we have gained a lot of experience in problem solving, working as a team, and making joints and connections between each process. We now have a deep understanding of how important it is to pronounce the name right. We were able to apply some knowledge based on our professional experience in coding and logical thinking in this project. Other than this, we were able to achieve the human-readable phonetics and store it in the Database. Operations such as insert and update to the Database have been implemented along with exception handling. Besides that, we continue towards completion of this project.

References

1. Gresta, Remo, and Elder Cirilo. "Say my name! An empirical study on the pronounceability of identifier names." Anais do IX Workshop de Visualização, Evolução e Manutenção de Software. SBC, 2021. Retrieved from https://sol.sbc.org.br/index.php/vem/article/view/17218/17056
2. Yoshida, M. T. (2018). Choosing technology tools to meet pronunciation teaching and learning goals. Catesol Journal, 30(1), 195-212. Retrieved from https://eric.ed.gov/?id=EJ1174226
3. Besides this, the project source code and details can be found on the mentioned repository.
4. nsana1999/AIGS_1003_Say_My_Name_Group1 (github.com)
5. David Pinto, Darnes Vilariño, Yuridiana Alemán, Helena Gómez, Nahun Loya,and Héctor Jiménez-Salazar. 2012. The Soundex phonetic algorithm revisited forSMS text representation. InInternational Conference on Text, Speech, and Dialogue.Springer, 47–55
6. Lawrence Philips. 2000. The Double Metaphone Search Algorithm.C/C++ UsersJ.18, 6 (June 2000), 38–43.
7. Chakkrit Snae. 2007. A comparison and analysis of name matching algorithms.International Journal of Applied Science. Engineering and Technology4, 1 (2007),252–257
8. L. E. Gill, M. J. Goldacre, H. M. Simmons, G. A. Bettley, and M. Griffith, “Computerised Linkage of Medical Records: Methodological Guidelines,” Journal of Epidemiology and Community Health, vol. 47, pp. 316-319, 1993