Modules:

* Speech to Text
* FAQ Prediction
* AI

**Speech To Text**

Description: This module contains the code to transcribe user audio into text.

Functions:

* transcribeAudio(self, api :str="houndify") -> str

Description: Transcribe user audio into text using a specified API.

API List: IBM, Houndify

If invalid API picked, raises TRANSCRIPT ERROR

Returns the transcribed text if transcription successful.

* YesOrNo(self, stt\_api) -> bool

Description: Gets the user input on a Yes or No question when asked a Yes or No Question

Exceptions:

ConversationAttempt

If attempts at conversation failed twice

TranscriptError

If invalid transcript API select or API cannot transcribe user audio

HangUp

If user says “bye”, hangs up the call

**FAQ Prediction**

Description: Module to predict an answer to a user query based on the question set it’s been trained on

Functions:

* train(self, csv\_holder="training\_csv")

Using a specified parent folder containing Q&A files as CSV

Automatically imports them into the dataframe

and writes the embeddings for the question set

to a .pickle file.

This trains the model on the question set it is given

* pred\_ans(self, usr\_query: str)

Predicts answer based on user query

Question set converted to embeds in .train() which is loaded here

Convert user query to embeds and compare to question set

to find best match.

Returns a list with the answer and a success rating.

If question match was not enough, return 0 + “Repeat Question”

If question was match was satisfactory, return 1 + answer

**AI**

Descrition: Combines the two modules into one and initates calls using the provided student details. Appends call data to the CSV which can be used by the ML Model to perform Lead Scoring Analysis.

Functions:

* initConversation(self, phone\_number, prior\_know\_bool)

Initiates call using the supplied phone number. This will ask the user a set of preliminary questions to gauge if they knew about the university or not and if they have any questions they need answers for.

Returns call data along with a transcription of successful user communication

* answerUserQuestions(self)

Answers the user questions using STT and FAQ\_Predict

* VAQuestionOpportunity(self, user\_query)

If there is an opportunity for a follow-up question to gauge if the student is interested or not, it will ask the question and record student answer

* recordConversation(self, user\_conversation, phone\_number)

Records the transcribed successful user communication to a .text file uniquely named by their phone number. This file is further used for sentimental analysis

* getSentimentPolarity(self, text)

Returns sentiment polarity on the given text

* SentimentalAnalysis(self, phone\_number)

Opens the text file using the supplied phone number if the file exists and performs sentimental analysis to judge whether they are interested in enrolling in the university or not.

* run\_app(self)

Reads the provided Student Details CSV and initiates calls and records call data to the CSV which is then used for Lead Scoring Analysis.