# Project Summary Document

| | |
|---|---|
| Team Number | Team 33 |
| Project Title | Framework for Automated Student Assessment using NLP - Audio |
| Document | Project Summary Document |
| Creation date | 2023/04/23 |
| Created By | Devisetti Sai Asrith, Geethika Thota, Sidhi Panda, P Pavan |
| Client | Arjun Rajasekar, RCTS |

**Brief problem statement**

This application allows students to improve their skills through ongoing daily assessment and being supervised by an experienced teacher. Doing this physically is time-consuming and is an administrative burden.

The current work proposes to automate the test-taking/grading process by using the audio of the student-teacher interactions during class, to perform this assessment. The work shall start with a limited set of known questions with clear answers in the form of multiple choices or True/False.

**Tools Used:**

| Tool Used | Purpose |
|---|---|
| React JS | Develop a demonstrative UI, which is self explanative |
| Node JS | Build the backend and enable connection to the database |
| Whisper | Speech-to-Text Conversion Module |
| Word2Vec | Compare two text files and check the similarity |
| MongoDB | Store the Question and Answers |

**API keys used:**

For the implementation of the backend, the following API routes were defined:

- "**/question**": Using the POST method, we use this route to post a true/false question and corresponding answer into the database.
- "**/oneword**": Using the POST method, we use this route to post a one-word answer question and corresponding answer into the database.
- "**/api/send-truefalse**": Using the POST method, we use this route to post the link of the submissions stored in the local storage for the take test → True/False page.
- "**/api/send-oneword**": Using the POST method, we use this route to post the link of the submissions stored in the local storage for the take test → OneWordAnswer page.
- "**/run-python**": Using the POST method, we use this route to post the output of an external Python file.

For making Network requests, we used "**fetch**" in the React files.

**Instructions to run the Web Application:**

- To Run Frontend: "npm start"
- To Run Backend: "nodemon server.js"

To run External Python Script

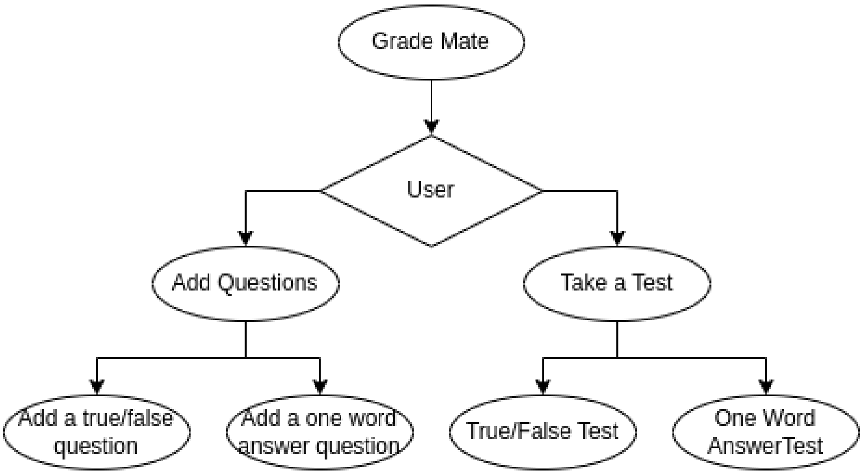
- For True/False Answer Submission: export FLASK_APP=mongoquestions_tf.py; export FLASK_ENV=development; flask run
- For OneWord Answer Submission: export FLASK_APP=mongoquestions_one.py; export FLASK_ENV=development; flask run --port=XXXX

Replace XXXX with a free port that is not occupied.

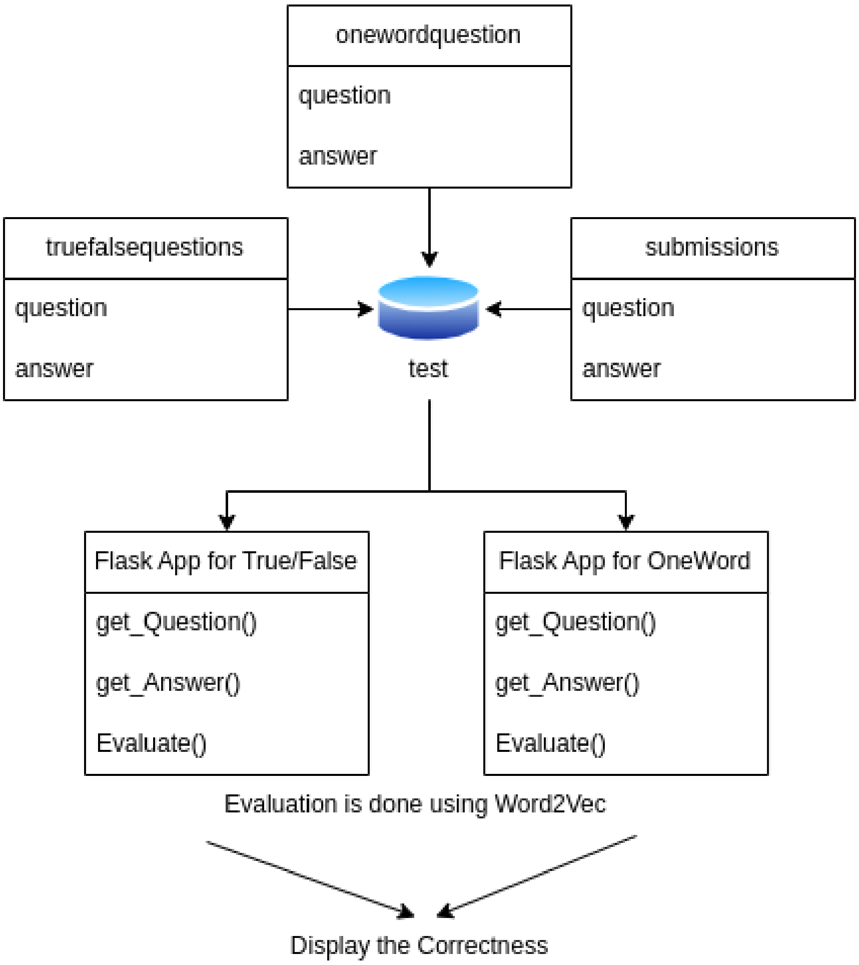
**External Modules to be installed:**

- Whisper Module (ASR module): Speech-to-text conversion module.
- Word2Vec (Deep Learning module): Word comparator to find the similarities.

**User Usecase Flow:**



**DataBase Flow:**


onewordquestion
question
answer
truefalsequestions
question
answer
test
submissions
question
answer
Flask App for True/False
get_Question()
get_Answer()
Evaluate()
Flask App for OneWord
get_Question()
get_Answer()
Evaluate()
Evaluation is done using Word2Vec
Display the Correctness