**Scope of Work: Scoring Oral Skills Project**

Professional Computing Group 10

Client: Dr Marinella Caruso

**Section 1: Project Goal**

This project aims to help students learning the Italian language (particularly beginners) to develop their speaking and pronunciation skills. The application should enable students to listen to pre-recorded italian speech, and then make their own attempt to repeat this speech. Students will be given an opportunity to report how well they believe they did, and are then scored using an audio similarity API.

The application’s main goal is to provide an innovative way for students to practise and obtain feedback on their Italian speaking and pronunciation. It is anticipated that the application will improve student’s confidence in their speaking abilities. Our client, Dr. Caruso, reports that beginner students often underestimate their own speaking skills. Students will hopefully be encouraged when they see that many of their attempts have been graded higher by the computer than by themselves. For those students who do not receive as high scores as they may have hoped, the project provides confirmation that this is an area that they need to improve on.

The platform’s secondary goal is to provide data for the investigation of whether repetition of strings of words impacts the learning of pronunciation and fluency in Italian, and whether an innovative way of receiving feedback motivates students in their learning.

**Section 2: Proposed Application**

**2.1 Overview**

The application will provide students with an interface to sign up and log in to the platform. The standard user home page will provide access to any speaking tasks assigned by their course instructor and past tasks that can be repeated for their own practise, as well as display all of the students' recorded results.

Inside an assigned task, there will be a number of speaking questions. Each question will have four “stages”:

1. The user will be presented with a playable audio where a native italian speaker speaks a word/phrase. When the user is ready, they can continue to the next step.
2. The user can record their own attempt at repeating the native speaker’s phrase. They can re-record their attempt as many times, until they are satisfied.
3. The user is asked to rate how well they feel their speech compares to that of the native speaker, this forms the self-evaluation score.
4. A speech similarity score is calculated, determining the similarity of the user’s audio to the given Italian audio. This is shown to the user along with their self-evaluative score, and other relevant insights.

The administrator home page will provide the course instructor (or anyone with admin privileges) the means to assign and manage tasks, and to view student results and insights. The client has also expressed a desire for analytics related to both student and class performance — which should be accessible from here.

**2.2 Functional Requirements**

**2.2.1 Requirements for Individual Speaking Activity**

| **Feature Name** | Play Audio |
| --- | --- |
| **Description** | Users must be able to listen to pre-recorded italian speech |
| **User Problem** | Students require audio output in order for them to copy the native speaker’s pronunciation |
| **Subfeatures** | * Users may play & pause audio * Users may listen to audio multiple times |
| **Considerations** | * N/A |
| **Minimum Viable Product (MVP)** | Ability to play supplied audio tracks |

| **Feature Name** | Record & Submit Audio |
| --- | --- |
| **Description** | Users must be able to record their own attempt of speaking italian |
| **User Problem** | Students require ability to submit their own speaking attempt so it can be compared and graded against the original |
| **Subfeatures** | * Users can make 3 attempts at recording * Audio should not be able to be uploaded as a file, only recorded |
| **Considerations** | * They can choose their best recording to submit |
| **Minimum Viable Product (MVP)** | Ability to record & submit audio track. |

| **Feature Name** | Self Evaluation Score (Input) |
| --- | --- |
| **Description** | Users must be able to input a numerical score reflecting how well they believe their speaking attempt compares to that of the original audio |
| **User Problem** | Students often underestimate their own speaking skills. The aim is for many students to be pleasantly surprised when they see how highly the computer grades them compared to their own estimate. |
| **Subfeatures** | * Score should be a simple numerical value scale |
| **Considerations** | * Value should be validated and only accepted if appropriate |
| **Minimum Viable Product (MVP)** | Ability to enter a numerical value within a specified range. |

| **Feature Name** | Similarity Score (Output) |
| --- | --- |
| **Description** | Users should receive a score (feedback) that indicates how closely their attempt compares to the pre-recorded audio. |
| **User Problem** | Students require feedback so they know how well their submission compares to a native speaker. |
| **Subfeatures** | * Score should be a simple numerical value scale |
| **Considerations** | * Possible use of audio similarity API’s |
| **Minimum Viable Product (MVP)** | Use of audio similarity API to compare student submission to pre-recorded audio, and output a numerical score(s) that is somehow representative of student performance. |

| **Feature Name** | Test Submission/Analytics Page |
| --- | --- |
| **Description** | Users should be able to view their similarity scores, along with their self-evaluation score and have this data displayed so that they can visualise their progress, and other relevant analytics. |
| **User Problem** | Students need to see an overview of their results of the test. |
| **Subfeatures** |  |
| **Considerations** | * Could include average grade as well as individual question scores |
| **Minimum Viable Product (MVP)** | Students are displayed their average similarity score and self score after completing a test. |

**2.2.2 Requirements for Data Storage & Access**

| **Feature Name** | Sign up & log in functionality |
| --- | --- |
| **Description** | Users must be able to sign-up to the website if they are new users, or log-in if they have an already existing account. |
| **User Problem** | Student submissions need to be attributable to the student who completed them. |
| **Subfeatures** |  |
| **Considerations** | * Involves database to facilitate * Potential separation by class/unit |
| **Minimum Viable Product (MVP)** | Ability to sign up & log in to website |

| **Feature Name** | Database & Information Storage |
| --- | --- |
| **Description** | Student attempts and scores should be stored by the system and attributable to their account. |
| **User Problem** | Student progress cannot be reviewed by the student or the instructor without information storage. |
| **Subfeatures** |  |
| **Considerations** |  |
| **Minimum Viable Product (MVP)** | Scores/attempt information stored in database and is attributable to the student who submitted them. |

**2.2.3 Requirements for Student Functionality**

| **Feature Name** | Student Interface |
| --- | --- |
| **Description** | Students should have a homepage that allows them to view their account, their past scores, and access tasks that have been set by their course instructor. |
| **User Problem** | Students require easy access to all the tasks assigned by their instructor over the course of the study period |
| **Subfeatures** |  |
| **Considerations** |  |
| **Minimum Viable Product (MVP)** | Student homepage with access to account (settings) and tasks assigned to their classes. |

| **Feature Name** | Task Repeatability |
| --- | --- |
| **Description** | For practice purposes, tasks should be able to be repeated even after the initial marked attempt (and even beyond the due date). |
| **User Problem** | Students should be provided plenty of opportunities to practise. |
| **Subfeatures** |  |
| **Considerations** |  |
| **Minimum Viable Product (MVP)** | After the first attempt, tasks can be repeated. |

**2.2.3 Requirements for Administrator Functionality**

| **Feature Name** | Administrator Interface |
| --- | --- |
| **Description** | Course instructor should have a homepage that allows them to:   * Review tasks they have set (and grades of students) * Review information & analytics regarding a particular class * View scores of a particular student |
| **User Problem** | Course instructor should be able to review student progress. |
| **Subfeatures** |  |
| **Considerations** | * Client mentions keen interest in analytics, excellent opportunity to implement them here |
| **Minimum Viable Product (MVP)** | Instructors can see scores of students for an assigned task. |

| **Feature Name** | Instructor Privileges |
| --- | --- |
| **Description** | Instructors should have special privileges that enable the administrator view. New instructor accounts should be able to be added. |
| **User Problem** | Course instructor should be able to review student progress. |
| **Subfeatures** | * Ability to add new administrator accounts |
| **Considerations** |  |
| **Minimum Viable Product (MVP)** | Instructors can see scores of students for all graded attempts. |

| **Feature Name** | Task Assignment |
| --- | --- |
| **Description** | Instructor needs to be able to create and assign tasks (quizzes) to classes for completion. |
| **User Problem** | Without tasks assigned to students (including audio files), they are not able to practise their speaking skills or submit any speaking attempts. |
| **Subfeatures** | * Grouping of students by class/unit * Implementation of due date |
| **Considerations** |  |
| **Minimum Viable Product (MVP)** | Instructors can create a task with at least one question and assign it to a group of students. Task should have a due date. |

**2.3 Non-Functional Requirements**

**2.3.1 User Interface Background & Requirements**

***Background***

The application is intended to be used in an educational setting by experienced Italian professors & teachers. It is aimed mostly towards beginner students, but the use of more complex audio tracks may allow easy expansion to higher level study groups.

The application is intended to be simple and intuitive with very little instruction required (instruction will mainly be related to sign-up at the beginning of the study period). This is particularly important as a complex interface will dissuade beginner learners from practising their skills.

***Requirement***

| **Feature Name** | Simple, professional, and intuitive interface |
| --- | --- |
| **Description** | Application should be simple to access and intuitive to operate. Appearance needs to be appropriate and professional. |
| **User Problem** | Students will have varying levels of technological familiarity. Tool is intended for educational use so professionalism is important. |
| **Subfeatures** |  |
| **Considerations** |  |
| **Minimum Viable Product (MVP)** | Interface that is easy to view and understand |

**2.3.2 Error Handling Requirements**

| **Feature Name** | Input Validation |
| --- | --- |
| **Description** | All inputs should be validated, and users should not be able to progress in a task without submitting a valid input. |
| **User Problem** | Invalid self evaluation score, or lack of audio submission, will mean that the task is unable to be marked. |
| **Subfeatures** |  |
| **Considerations** |  |
| **Minimum Viable Product (MVP)** | Self evaluation score is validated (error messages if necessary), and audio submission is required. Submission will not be possible if any input is not valid. |

**2.3.3 Security Requirements**

| **Feature Name** | Injection Attack Protection/Validation |
| --- | --- |
| **Description** | Sign-up and log-in inputs should be validated. Robust protection of database information is desired. |
| **User Problem** | Application should not be vulnerable to injection attacks |
| **Subfeatures** |  |
| **Considerations** |  |
| **Minimum Viable Product (MVP)** | Inputs validated and processed to prevent injection attacks. |

| **Feature Name** | Password Protection |
| --- | --- |
| **Description** | Passwords should not be stored as-is. In the event that the contents of the user table are compromised, passwords should not be able to be revealed/deciphered. |
| **User Problem** | Student and staff passwords are private and may be used for other applications and purposes. Users expect information to be protected. |
| **Subfeatures** |  |
| **Considerations** |  |
| **Minimum Viable Product (MVP)** | Passwords are hashed before being stored in the database. |