# Milestone 1 - Report

**CMPT 276 @ Simon Fraser University**

**Fall 2023**


| **Group 32** |
| --- |
| **Zheyuan Ou** |
| **Damir Zharikessov - 301541028** |
| **Victor Nguyen - 301458739** |
| **Tegvaran Sooch - 301418178** |

# Table of Contents

# Overview

Our project's primary goal is to develop a web-based application that offers valuable support to students who are non-native English speakers and those with visual impairments. The core concept revolves around harnessing the capabilities of the OpenAI API for language translation and the SpeechSynthesis API for text-to-speech conversion to enhance the accessibility and comprehensibility of academic assignments.

For non-native English speakers, we envision a tool that enables them to copy text from their assignments and gain a clear understanding of the content in their native language. This facilitates improved comprehension of their academic tasks. Additionally, our application will extend its utility to translating lecture notes into the user's preferred language, further enhancing learning experiences.

Furthermore, we are introducing a speech-to-speech translation feature that allows students to engage in seamless conversations with their peers. This feature ensures effective communication and mutual understanding between individuals working on collaborative projects, bridging language barriers and promoting teamwork.

Now that we have presented an overview of our project and its core objectives, it is essential to delve into the methodology that will guide our development process. In the following section, we will explain our choice of the Software Development Life Cycle (SDLC) model

# Choice of SDLC Model:

We have opted for the Extreme Programming (XP) methodology for our project due to our team's composition. Specifically, we have two team members experienced in JavaScript, while the other two are unfamiliar with it.

To address this, we are leveraging XP's pair programming practice, where each experienced JavaScript developer is paired with a less experienced counterpart. This ensures that the learning curve is smoother for those new to JavaScript, as they have immediate guidance and problem-solving support. Pair programming not only accelerates learning but also enhances code quality and issue resolution. It aligns with our goal of enabling every team member to contribute to the project actively.

In addition to pair programming, we are incorporating other XP practices like test-driven development, continuous integration, and small, iterative releases. Our choice of XP emphasizes collaboration, flexibility, and efficient knowledge sharing to deliver a high-quality application.

# User Stories:

Here are 4 user stories that highlight use cases for our application:

**User Stories for OpenAI API:**

1) As a student who is new to an English-speaking country, I need the ability to easily translate my assignments into my first language. This will help me better comprehend the assignment requirements and ensure that I can perform the tasks effectively.
2) As a student who does not have a lot of time and is new to an English-speaking country, I want to be able to summarize my assignments in my language so that I can get a quick overview of them.

**User Stories for SpeechSynthesis API:**

1) As a student who is new to an English-speaking country, I seek the functionality to express ideas in my native language and have them instantly converted to English. This feature is essential for effective communication within my team and ensures that language barriers do not hinder our collaboration.
2) As a language learner, I want the ability to convert text-based assignments and study materials into speech so that I can practice listening to and improving my pronunciation in the language I'm learning.

Now that we have defined the user stories that capture the specific needs of our project, let's delve into the technology stack we have chosen to bring these stories to life. The technology stack serves as the foundation that will enable us to implement the features and functionalities required to address these user stories effectively.

# Technology Stack

We have selected **JavaScript** as our primary programming language due to its suitability for web application development. Additionally, we will leverage **Bootstrap** to enhance the user interface with features such as navigation bars and aesthetically pleasing buttons.

Furthermore, we will incorporate **Node.js** into our technology stack, as the OpenAI API operates using Node.js for seamless integration.

For version control and task management, we will utilize **Git** and **GitHub**, enabling us to efficiently track changes and create task tickets for project milestones.

For web hosting, we will utilize **GitHub Pages** to host our website. This platform provides a straightforward and convenient way to make our application accessible to users online and it is free for students.

Now that we've covered our technology stack, let's shift our focus to the APIs that will empower our application's capabilities. In the following section, we will introduce the chosen APIs and elaborate on why we have selected them to enhance our project.

# Chosen APIs

Here are the two APIs we chose for this project and why we chose them:

**OpenAI API:** This API allows us to seamlessly integrate ChatGPT 3.5 into our web application, enabling both text translation and AI-powered text summarization. We chose this API for its robust language processing capabilities, ensuring accurate assignment translation.

**SpeechSynthesis API**: This API enables us to utilize both text-to-speech and speech-to-speech functionality. We chose this API because it is compatible with popular modern web browsers such as Chrome, Firefox, and Safari, ensuring accessibility across various devices.

With the capabilities of both the APIs outlined, let's now move forward to explore the specific features and functionalities that we have planned for each of these APIs

# Planned Features for each API

Here are the features we plan on implementing for each API:

**OpenAI API:**

1) Language Translation: Translate text from English to the user's preferred language.
2) Assignment Summarization: Provide concise summaries of text in the user's preferred language.
3) Language Detection: Automatically detect the source language of a text if the language provided is not English.

**SpeechSynthesis API:**

1) Multilingual Text-to-Speech: Enable users to listen to the translated text.
2) Multilingual Speech-to-text: Allow users to speak in their native language and have it automatically transcribed into text.
3) Customization of Speech Voice: Provide settings to adjust the voice pitch and speech speed for text-to-speech and speech-to-speech functionalities, allowing users to customize their experience.

# Work Breakdown Structure

Link to WBS

For the creation of the Work Breakdown Structure (WBS), our team conducted a collaborative effort to systematically break down the project tasks. The WBS is structured into two main sections: Planning and Implementation.

## Planning

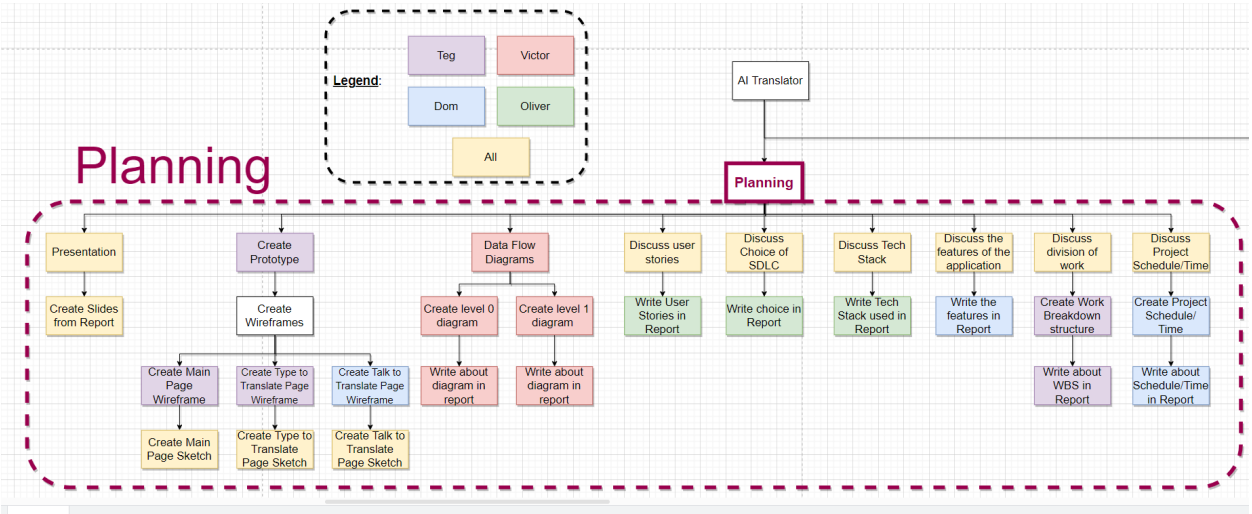


*Figure 1 - WBS (Planning section)*

This section encompasses all the tasks completed during the planning phase (Milestone 1) of the project. The color-coded boxes indicate the responsible person for each section.

### Implementation

This section includes all the work that will be done in the implementation (Milestone 2) part of the project. Check the link to view a clearer WBS.

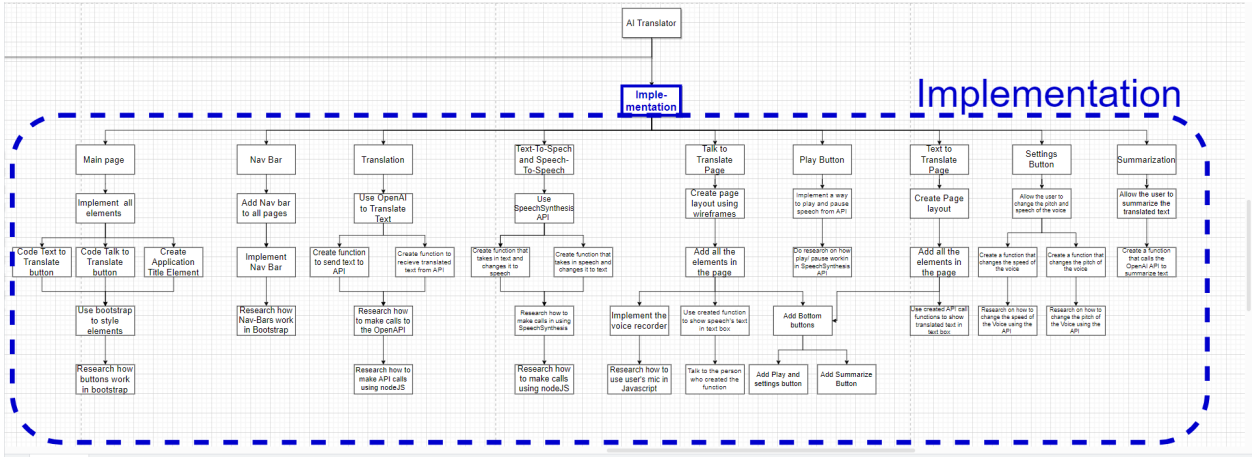


*Figure 2 - WBS (Implementation Section)*

The work in this section has not been subdivided yet, the general approach for task distribution is as follows: team members unfamiliar with JavaScript and HTML will focus on page layouts, while those with JavaScript expertise will handle API calls and function development.

With the tasks and responsibilities defined in our Work Breakdown Structure, our next step is to outline the project schedule and timeline. In the following section, we will present a detailed plan for how we intend to execute and complete the various project components within the allocated timeframes.

# Project Schedule/ Timeline

The team will follow the high-level schedule presented below. In addition to the sprint structure, we will be holding team meetings to ensure effective communication and progress tracking.

**Meeting Schedule:**

- Mondays (2 pm): These meetings will serve as planning sessions to discuss task assignments and responsibilities for the upcoming week.

- Thursdays (12 pm): These meetings are designed to facilitate progress updates, address any issues, and offer assistance to team members who may be facing challenges or blockers.

The graphical representation of our project schedule can be found in the chart below.

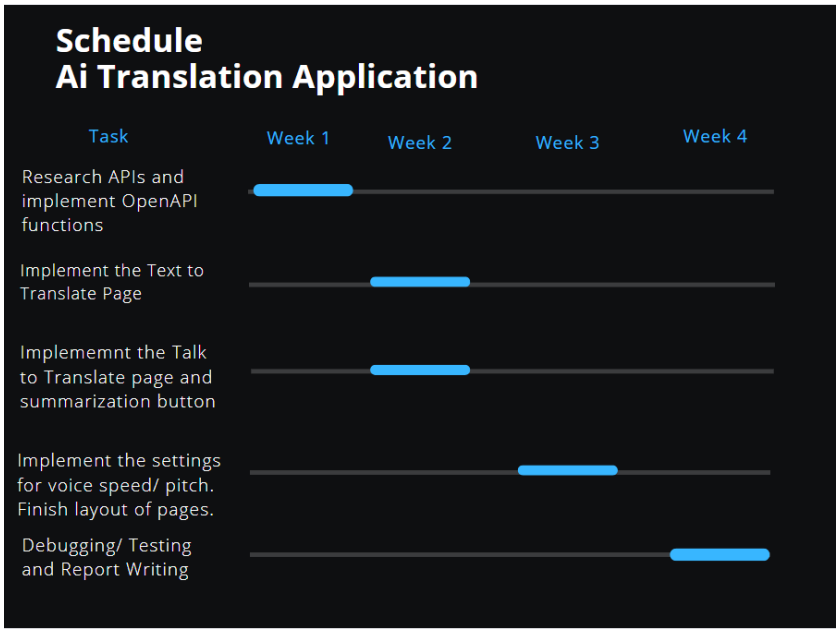


*Figure 3 - High-level Schedule*

We plan on completing the “Text to Translate” Page for the Project Check-in.

# Wireframes

Here are the wireframes that were created using Figma presented on a MacBook Air.

## Main Page

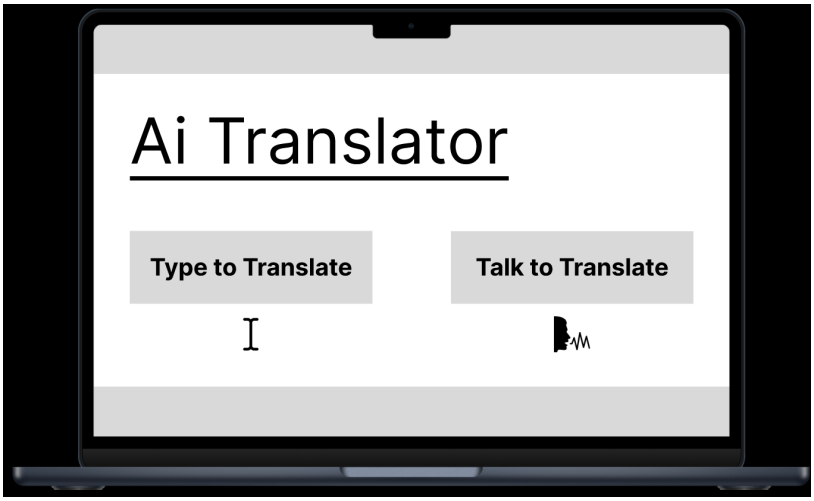


*Figure 4 - Main Page*

This page serves as a welcome page where users can select if they want to translate text (Type to Translate) or if they want to translate speech (Talk to translate).

## Type to Translate

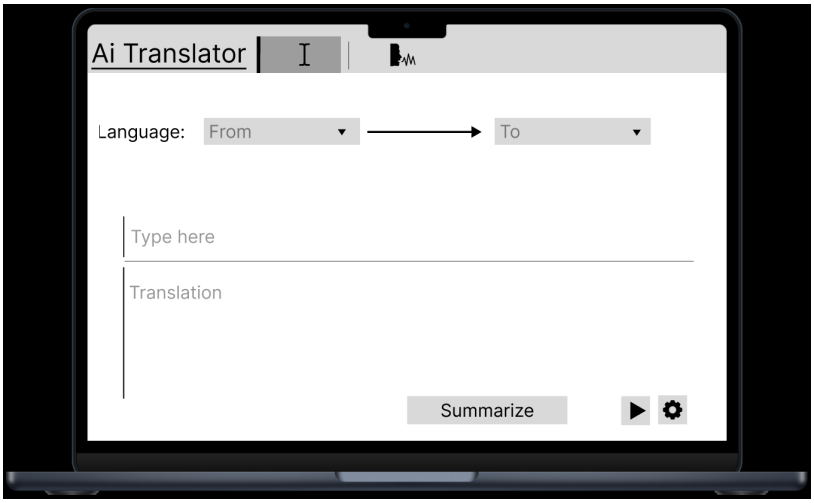


*Figure 5 - Type to Translate*

This wireframe depicts the user interface for the 'Type to Translate' feature, allowing users to enter text in their native language for real-time translation.

The link to the prototype on Figma can be found here: link

### Talk to Translate

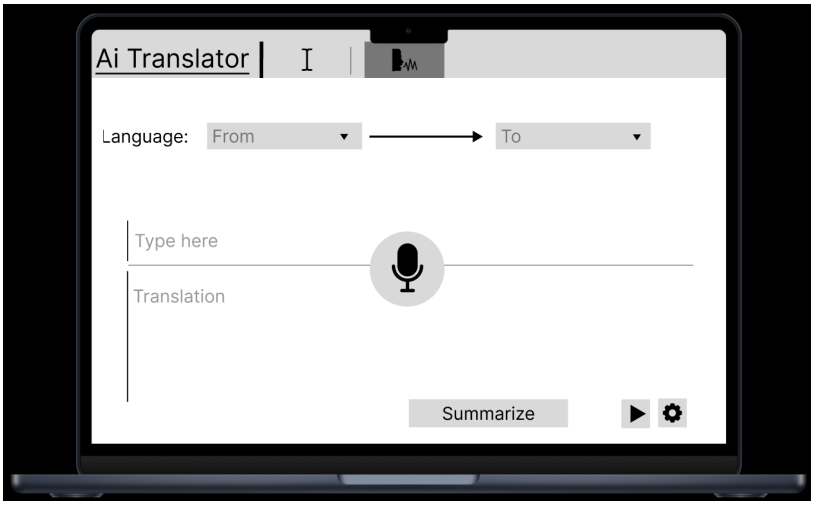


*Figure 6 - Speak to Translate*

This wireframe illustrates the user interface for the 'Talk to Translate' feature, enabling users to speak in their native language for instant speech-to-speech translation.

# Data Flow Diagrams

### High-Level Overview Diagram

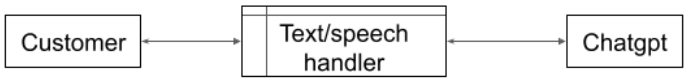


*Figure 7 - Level 0 DFD*

Our level 0 diagram displays the data starting from the customer to be sent to Chatgpt to be translated. Then afterwards, it is sent back to the customer in its translated state as text or speech.

## Detailed Diagram

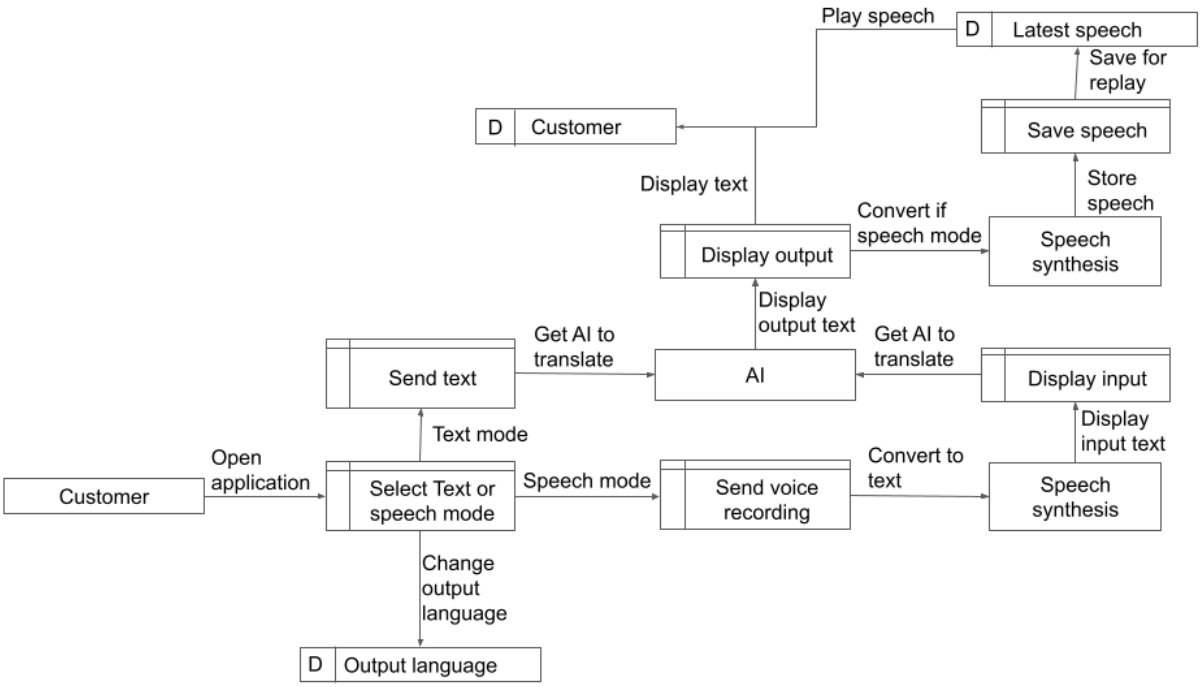


*Figure 8 - Level 1 DFD*

Our level 1 diagram shows the data process with a much closer inspection relating to our system. The data starts on the customer on the bottom left opening the application and choosing speech mode or text mode. After doing so, the customer could also change the output language to the language they want their message to be translated to. As for the modes themselves, text mode can immediately be sent to the AI and quickly sent to the customer as text. Speech mode on the other hand has to be converted into text in order to be fed to the AI. Along with displaying the output as text, we will also convert the message to a speech with the speech synthesis and save it in order to be replayed by the customer.