SI 568
Philip Menchaca (pmench)

Transcript Editor: Project Timeline

Purpose

By leveraging the OpenAI API and OpenAI's Large Language Model, the time and cost of correcting interview transcripts will be greatly decreased. This will benefit journalists and others (e.g. nonprofits required to keep meeting minutes) who currently must spend sometimes significant time and money manually editing interview transcripts.

Outputs

- Output 1: Command-line tool that allows users to upload a transcript file and receive back a corrected version in text format.
- Output 2 (stretch goal): Command-line tool that allows users to upload an audio file and receive a transcript back.
- Output 3 (stretch goal): Command-line tool that allows users to receive a summary of an interview after uploading an interview transcript.

Outcomes

- Outcome 1: Time and cost efficiency of transcription will be improved.

Assumptions

- The OpenAI language model is capable of editing English-language text, improving grammar and spelling with a minimum number of errors.
- The OpenAI language model will not introduce an unacceptable number of errors (defined as making a transcript less accurate than the baseline text).
- Restrictions on the OpenAI API (rate and token limiting) will not prevent the tool from performing as intended.

Tasks and Resources Required

Research OpenAI API in-depth

- Requirements
    - OpenAI API documentation
    - Access to ChatGPT

Scope and generate ideas for tool's purpose

- Requirements
    - OpenAI API documentation

Write proposal

- Requirements:
    - Project purpose and description
    - Supporting documentation (e.g. market research)

SI 568
Philip Menchaca (pmench)

Obtain OpenAI API key, test transcript, and install OpenAI library

- Requirements:
    - OpenAI account
    - Machine-generated transcript in text format, suitable as a test case
    - Python development environment

Write code for minimally viable tool

- Requirements
    - OpenAI account
    - Machine-generated transcript in text format, suitable as a test case
    - Python development environment

Test different OpenAI models

- Requirements
    - Access to OpenAI API and documentation

Write code to tokenize and chunk longer text

- Requirements
    - Python development environment
    - Text exceeding 4096 or 2048 tokens
    - Text tokenizer (e.g. NLTK)

Test rate limiting and write code to manage limits if necessary

- Requirements
    - Python development environment

Refine code to optimize accuracy of output

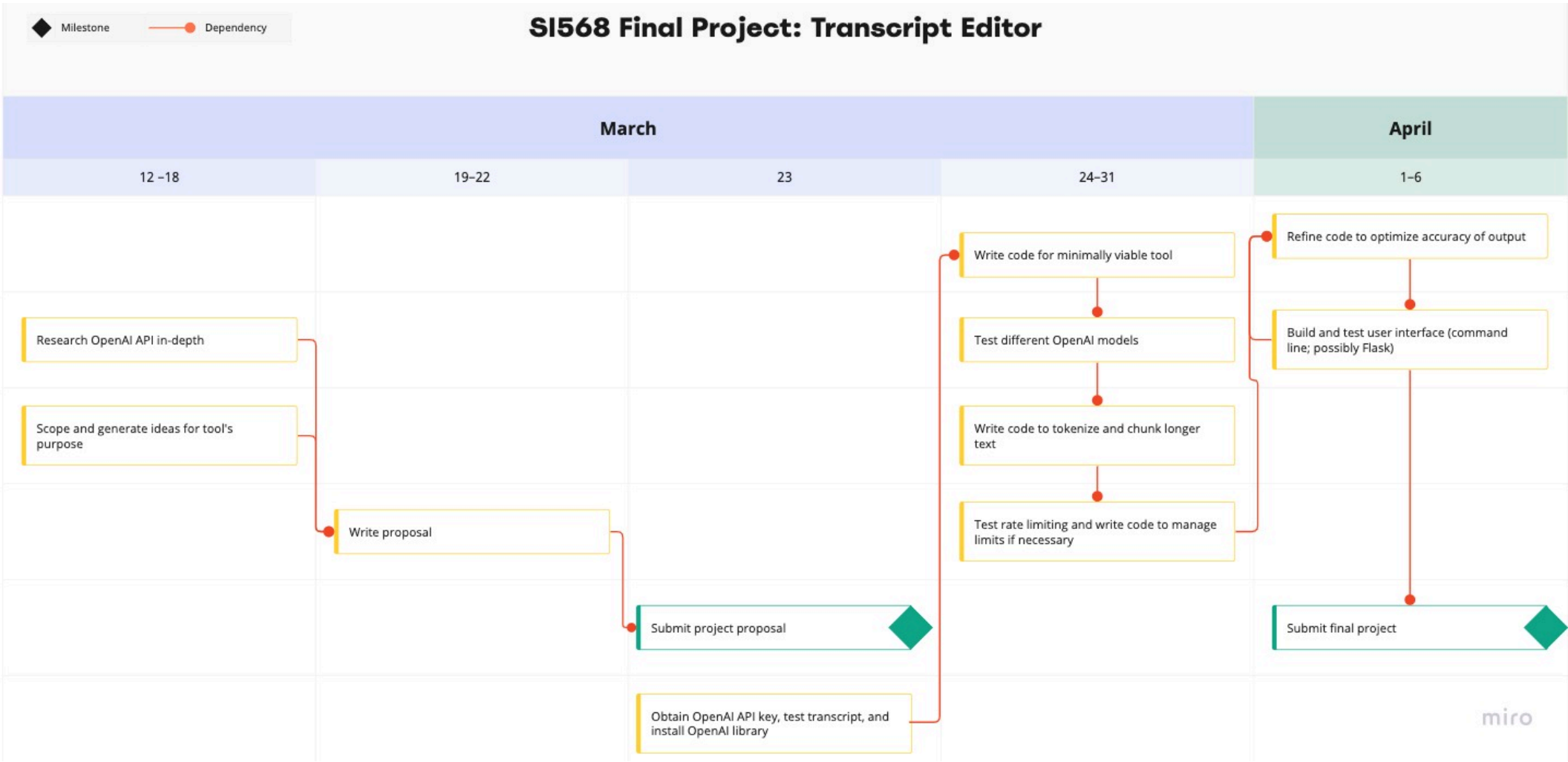
- Requirements
    - Python development environment
    - Working code that accepts a text file and returns an edited transcript

Build and test user interface (command line; possibly Flask)

- Requirements
    - Python development environment
    - Working code that accepts a text file and returns an edited transcript

<u>Timeline</u>

See attached.


Milestone
Dependency
SI568 Final Project: Transcript Editor
March
April
12 –18
19–22
23
24–31
1–6
Research OpenAI API in-depth
Scope and generate ideas for tool's purpose
Write proposal
Submit project proposal
Obtain OpenAI API key, test transcript, and install OpenAI library
Write code for minimally viable tool
Test different OpenAI models
Write code to tokenize and chunk longer text
Test rate limiting and write code to manage limits if necessary
Refine code to optimize accuracy of output
Build and test user interface (command line; possibly Flask)
Submit final project
miro