Product Requirements Document

Project: AI Driven Writing Assistant
Team: Aziksa AI Architects (AAA)
Sponsor: Aziksa
Authors:
Jonathan Cheng, **jonathancheng@ucsb.edu**
Paritosh Jha, **paritoshjha@ucsb.edu**
Tanish Kumar, **tanish_kumar@ucsb.edu**
Aashay Parab, **aashay_parab@ucsb.edu**
Vyomsarit Singh, **vyomsaritsingh@ucsb.edu**
Alexander Ward, **alexanderward@ucsb.edu**

INTRODUCTION

Problem

English is becoming the global language of business. More and more multinational companies are mandating English as the common corporate language. This is done to facilitate communication and performance across geographically diverse functions and business endeavors. Thus, it is essential to be able to speak fluent business language in English. However, there exists a gap in the language fluency level for non-native English-speaking corporate employees to perform their duties. These employees are being contracted on the expectation that they will be able to perform their duties in their native tongue and English. Creating a tool that will assist them while they reach fluency is a market necessity at the moment.

Project Significance

About 1.5 billion individuals worldwide spoke English as a second or native language in 2023. 1.08 billion people are non-native speakers. However, the general consensus is that it takes between five to seven years to achieve advanced fluency, and the business world expects advanced fluency from their employees, especially in business and social contexts. Thus, the gap in language fluency level for employees is a massive roadblock for workforce career growth.

Current Solutions

Currently, the gap in language fluency is being solved in a piecewise approach: using a combination of translation (Google Translate/DeepL), peer help, email template, asking supervisor to review communication, and language training. Language learning apps like Duolingo and Babbel aim to teach non-native speakers to increase English proficiency through daily exercises and games.

Vendors are introducing AI-driven writing apps which are still in very early development phases. While some of these solutions are effective, many of them still require a high level of

English fluency to write the email, prompt the app, or understand and correctly address the feedback. Moreover, they do not address the time spent by employees learning the language and achieving fluency, during which company productivity and communications may suffer.

Assumptions

Our system is built on a few significant assumptions. Primarily, we assume that our user is a non-native English speaker, and requires some form of assistance with forming a specifically tailored email. Additionally, although our user is not expected to be a native speaker, we still expect them to have a baseline understanding of the language, as our first UI will be in English. The user is also expected to be able to run an application on iOS, as the product will be deployed as an app.

Project Outcome

Our goal is to leverage an LLM (Llama2) to develop an English learning and writing assistant for non-native language speakers in the corporate workforce who are in a business context. We will implement an application and an LLM API to handle features and finetune the model - providing prompt text cleaning, adding context and examples, adding output formatting instructions, determining the user goal, and adding business roles like customer service representatives, sales executives, etc.

The product is designed to support our users along every step of the email prompting and writing process. Users will be able to tailor their emails based on the writer's role, email type, audience, and key points to be included. Additionally, the product will provide correctness, clarity, engagement, and tone detection checks on the generated text. The writing assistant will be available in an app that will provide users with a UI to enter their prompt information. This process will offload the prompt burden from the user onto our APIs, allowing non-native English speakers to form a fluent and professional piece of writing without needing to articulately express their thoughts in English, which is required to interact with traditional AI tools such as ChatGPT and Bard AI.

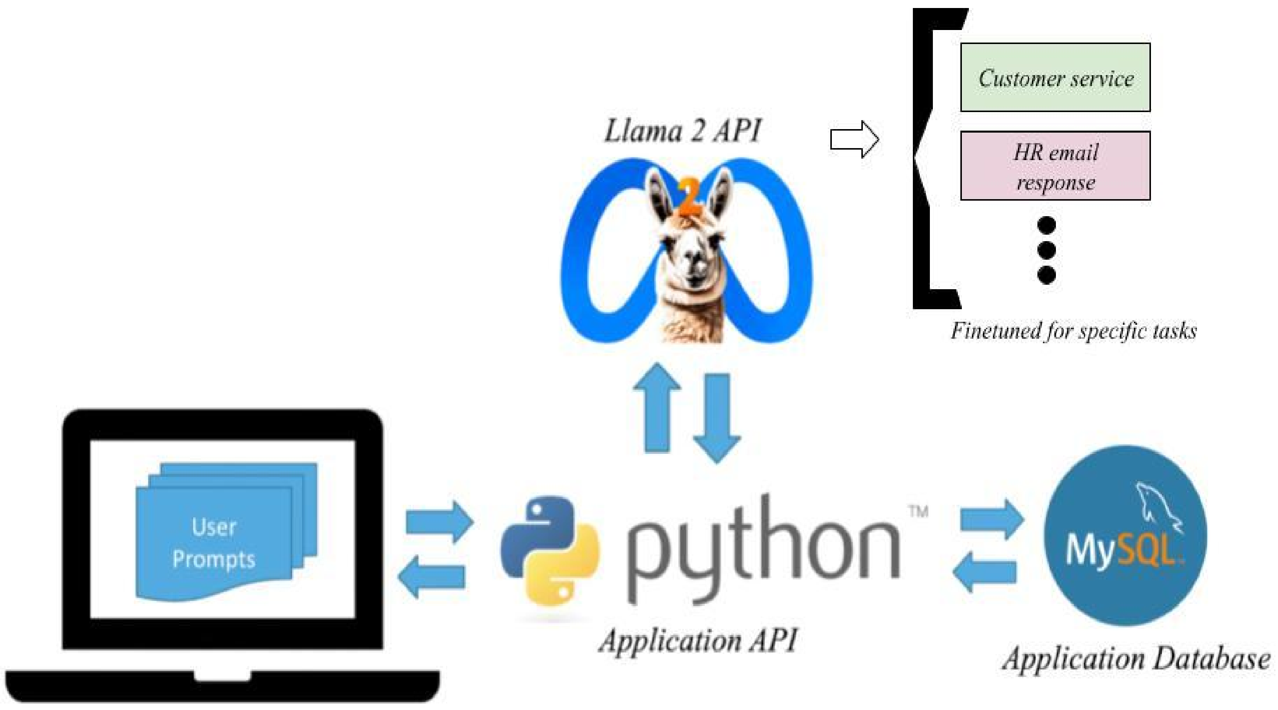
Project Milestones

For the first quarter, our primary goal is to implement a working backend for our writing assistant and to deploy the model to production. Some key milestones include fine-tuning the LLM for data privacy and personalized feedback by providing context and examples, outputting formatting instructions, and better-determining user goals. This will be achieved via an API, and by creating several endpoints designed to handle specific features and parameters in the request.

For the second quarter, we will focus on developing the mobile app. Our milestones include: designing the frontend of the writing assistance app (on Figma), developing the app primarily through React Native, connecting the app with the LLM, and integrating the app with email tools (with assistance from Aziksa mentors to handle user information). Ultimately, our goal is to release our application to production by the end of the Winter Quarter.

SYSTEM ARCHITECTURE OVERVIEW

High Level Diagram

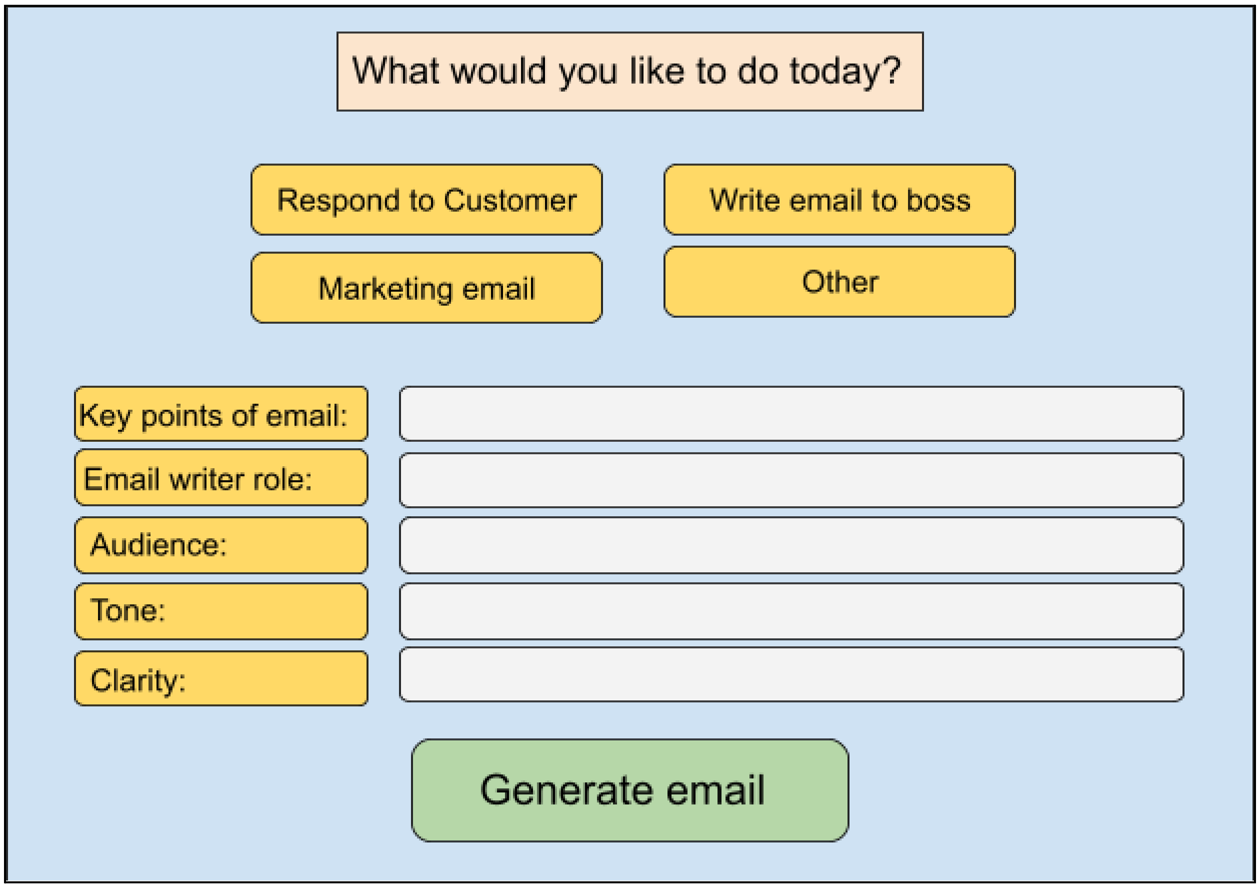


User prompt gets handled before going to the LLM API

- Prompt text cleaning
- Add context and examples to prompt
- Add output formatting instruction

Can be done via separate endpoints for particular features or using a parameter in the request

User Interaction and Design



REQUIREMENTS

User stories:

1. Create endpoint to fetch list of valid options for user to select from
   - Allow filtering based on category or other criteria
   - https://trello.com/c/OU4DLwK9
2. End point for constructing prompts
   - Implement an endpoint that takes the selected options as input and constructs a prompt string
   - https://trello.com/c/wkYMbqIe
3. Validate the selected options to ensure they are valid and compatible.
   - Check to make sure that selected input can work together i.e. no conflicting keywords which might mess up or confuse LLM.
   - https://trello.com/c/fUO1ttpG
4. Define the format of the response that includes the constructed prompt.

- Create a format from the selected input which will be imputed into the LLM, must be consistently formatted no matter the input
- https://trello.com/c/bdSmBkMt

5. Implement error handling for invalid input or server errors
    - Create checks for invalid input or server errors and add desired behavior such as prompting user of the error message
    - https://trello.com/c/evauhMzP
6. Ensure that the API is secure, using authentication and authorization mechanisms if needed
    - Create login for users, and make sure no API keys are leaked github or in webapp.
    - https://trello.com/c/uXAIQoLq
7. Optimize the API for performance to handle a potentially large number of concurrent requests.
    - Optimize runtimes of any computation (no exponential time functions)
    - https://trello.com/c/FWVnUYRa
8. Provide clear and comprehensive documentation for UI developers to integrate and use the API.
    - Create documentation for future use to expand on current project
    - https://trello.com/c/AHcKTlBp
9. Implement versioning for the API to support future updates without breaking existing integrations.
    - This can be done using our URI Path
    - https://trello.com/c/LBATu1Js
10. Create abusive language filter
    - Prevent abusive language input from user and output from LLM
    - https://trello.com/c/BWdlUPTm

APPENDICES

Generative AI Stack:

1. LLM - Llama2, Google Vertex AI
2. Cloud Environment - Google Cloud
3. Operating System - Linux
4. API Programming Language - Python, FastAPI, Uvicorn
5. Python Libraries:
    a. Pandas
    b. Numpy
    c. Matplotlib
    d. Sci-kit-learn
6. API Testing Tool - Postman

Web/Mobile App Stack:

1. Programming Languages - React Native, NodeJS, Javascript
2. Database - MySQL
3. Operating System - Linux

Source Control: Github

Issue/Bug Tracking: Github/Trello