Sprint Planning Document (Sprint 2)
Sprint Goal Backlog (Sprint 2)

February 19, 2025 – March 25, 2025

Trang Do, Donovan Kohler, Samuel Kwon, Raudel Armenta, Anthony Rutherford

High-level Project Overview

Project Mission:

- The focus of our project is to explore what is possible with the cutting-edge Apple Vision Pro. We aim to build an app that collects data from the Vision Pro and utilizes Hugging Face models for inference to provide a kind of distributed intelligence. This project will also set up initial infrastructure for future development with the Vision Pro and cloud computing.

Problems We Are Solving:

- People don’t know exactly what the Vision Pro can be used for
- Developing on the Vision Pro is relatively new and a sandbox environment to play around with the headset would make further development easier

Project Overview (High-Level Features):

- **VisionOS app - Discover:**
    - **Photo Library:** Users can select an image from the Photos app on the Apple Vision Pro and display the image.
    - **Main Camera Access:** Users can capture an image using the main camera from the Apple Vision Pro and display the captured image.
    - **Connect to WebSocket**: Users can send the selected image to AWS Lambda function via WebSocket. Then, the app can receive the response from the function and display the result to users.
    - **Speech Recognition:** Users can talk to the headset and apply speech recognition to send a request to OpenAI. Then, the app can receive the response from the OpenAI and display the result to users.
- **Cloud Computing**
    - **Cloud Service**: AWS, using either EC2 instance or Lambda functions
    - **Protocol:** WebSockets Secure, as required by project sponsor
    - **AI/ML:** Hugging Face models
    - **Flow**:
        - Client establishes WebSocket connection with AWS
        - Client sends message over connection
        - AWS receives message and uses Hugging Face models to infer and provide result
        - Result sent back to user over connection

Sprint 2 Planning

Sprint 2 Goals:

1. **Attempt to install dependencies within AWS lambda (pivot to EC2 if necessary)**
2. **Get our code running in AWS so frontend can integrate**
3. **Add speech recognition to the visionOS app**
4. **Make the project website more appealing, interactive, and user-friendly**
5. **Display the result from the data processing in AWS to the user in the visionOS app**

Sprint 2 Deliverables:

- **Install dependencies on AWS lambda**
  - **Assigned:** Sam Kwon, Donovan Kohler, Anthony Rutherford
  - Attempted to install dependencies on AWS lambda function:
    - Tried zip file of dependencies
    - Tried AWS S3 bucket with dependencies
    - Tried AWS lambda layers
    - Researched Docker use
  - Concluded that AWS lambda is not the right fit for what we need due to large dependencies and lack of persistence
- **Setup EC2 instance**
  - **Assigned:** Anthony Rutherford, Sam Kwon, Donovan Kohler
  - Get permission from sponsor to spend money on more compute
  - Create and configure EC2 instance
- **Port code over to EC2, install dependencies in EC2, hit with Postman**
  - **Assigned:** Sam Kwon
  - Port code into EC2 instance
  - Install dependencies
  - Run code and hit with postman
- **Update the project website to add animations and make it more appealing, interactive and user-friendly**
  - **Assigned:** Trang Do
  - Create a website using NextJS
  - Design the landing page about project description and used technology
  - Add subpage about the team member information
  - Add subpage about deliverables

    - Use Framer Motion to add animation to each component and use Tailwind CSS to style each element
- **Create components to display results from the server after sending the data**
    - **Assigned:** Trang Do
    - Create a new view called ResultTableView to display the objects detected and their confidence level from the data processing in AWS to the user
- **Refactoring the visionOS code to follow Model-View-ViewModel (MVVM) architecture**
    - **Assigned:** Trang Do
    - Follow SwiftUI tutorials online to understand SwiftUI better
    - Refactor the code to improve readability and follow MVVM architecture
- **Apply speech recognition on the visionOS app**
    - **Assigned:** Raudel Armenta
    - Created a new button to translate speech to text.
    - Researched how to send requests to OpenAI.