# Squire – AI Assistant Documentation

## Team Members

**Jayden Sipe – 100% contribution**

## Brief Overview

Squire is an AI (Artificial Intelligence) assistant built with modularity in mind to be able to add extra functionality easily. It was built using Python, along with spaCy, TensorFlow and LangChain.

## Overview of Functions

Firstly, one must train Squire using the intents.json file located in training/intents. This trains Squire to differentiate different commands. The default command it will fall back to if it does not understand the user's input is the ChatGPT integration, allowing it to function as a general assistant.

The next step is called the Handler phase. This initializes Squire by instantiating the UI (User Interface) thread and houses the functions necessary to pass input between the UI and Language threads.

Once a user inputs a command into chat box, Squire will process that command by trying to understand what the user's intent was. If an intent is not found, it will default to ChatGPT. If an intent is found, it will match the correct external function call. I was able to provide *LIFX* integration and *OpenWeatherMap*. The *LIFX* Integration allowed me to control certain lights in my house and the *OpenWeatherMap* integration allows you to view weather data about locations.

Squire will then reply using either preset responses or ChatGPT to fulfill the user's request.

## Implementation Discussion

To further extend Squire's functionality, one may add more external functionality.

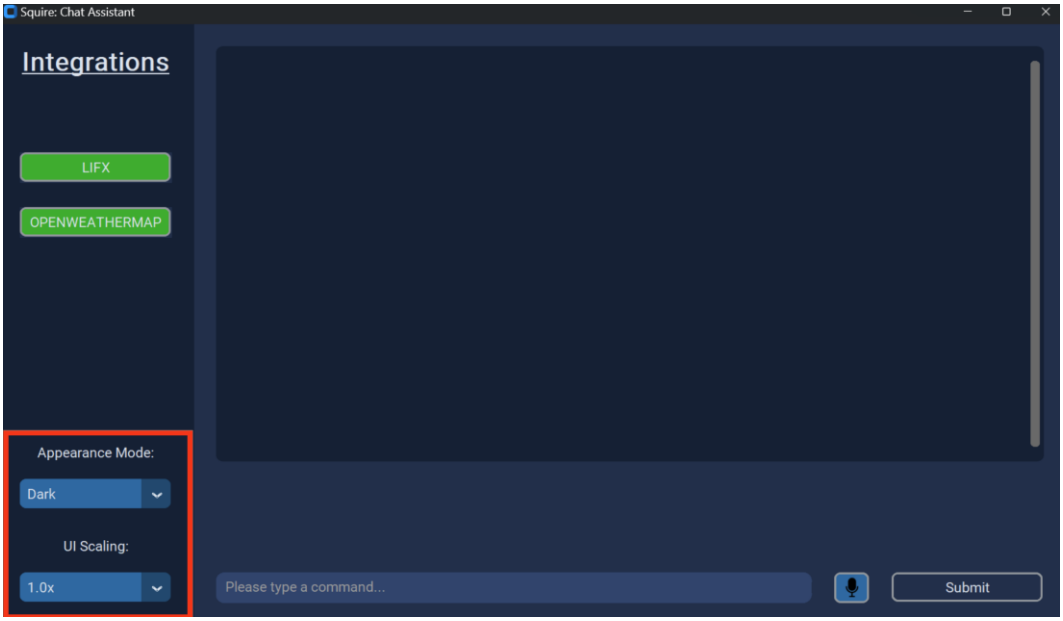
This involves:

1. Adding intents to *intents.json*, making sure to note *tag* added.
2. Create a new class in *external/apis*, passing in a name to use in settings.json.
3. Modify brain.py to load in external dependency and *_match_intent* function to catch your intent and send to the correct external class.

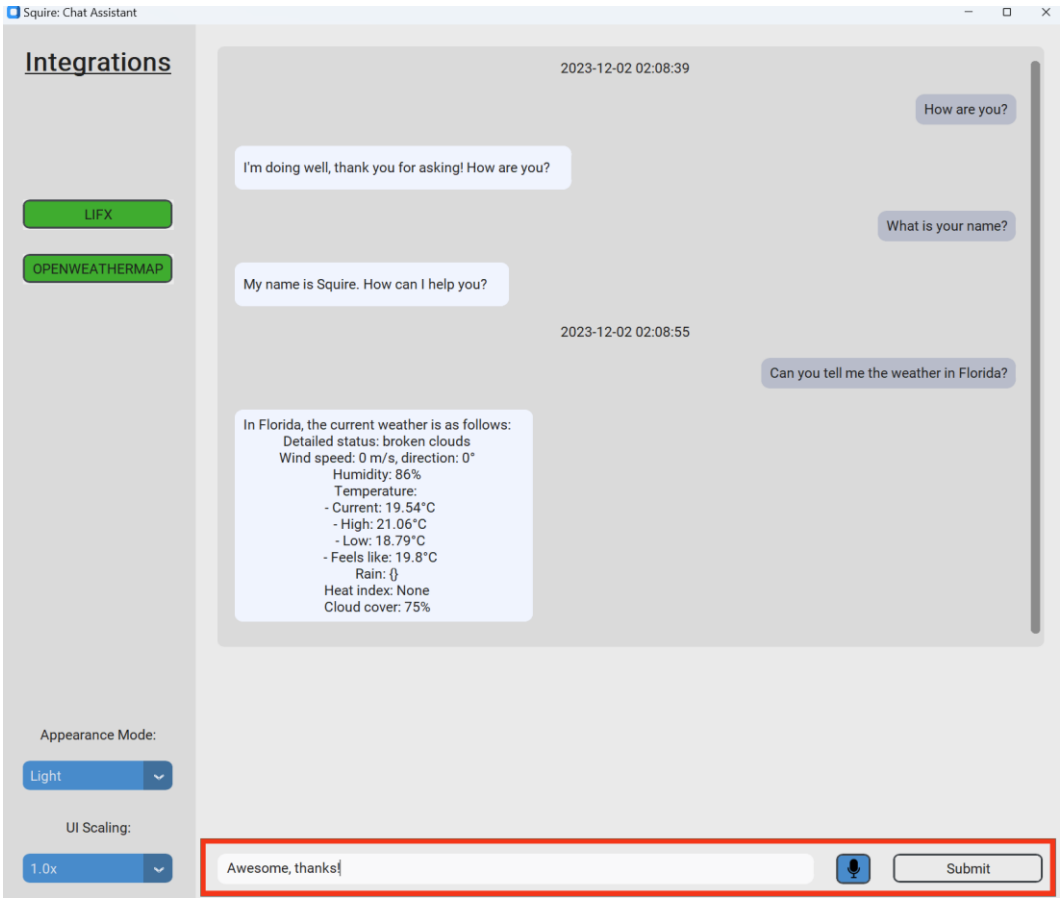
# Usage Documentation

**Download for Squire:**
https://drive.google.com/file/d/1VuctQlYW2bRbtDCnS_qUuuUwfy0XDKfe/view?usp=sharing

Once Squire is downloaded, open the folder, and run **main.exe**.

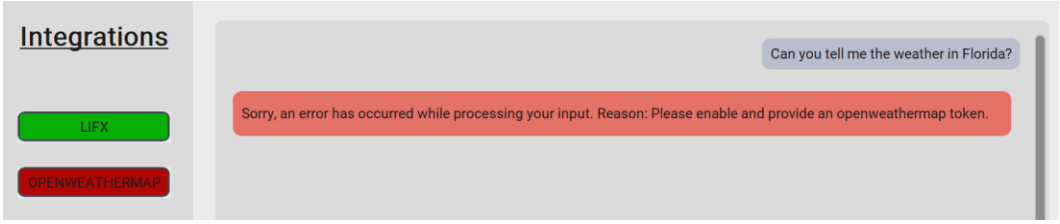




You will be met with Squire's UI. On the *bottom left-hand* side, you can configure the Appearance Mode and UI Scaling of the application.

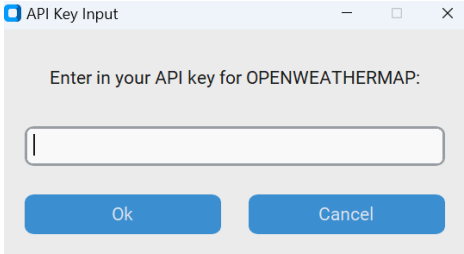



The main functionality of Squire is through the *bottom half* of the application. Here, a user can type in or use their microphone to input a command to Squire.

To use the microphone input, click on the microphone symbol and speak into the microphone.




Squire also has the functionality to include external APIs and functionality once implemented in the code. Here we can see the *LIFX* external functionality is ENABLED and has an API key included in the *settings.json*. The *OpenWeatherMap* functionality is DISABLED and Squire lets the user know they must provide an API key by clicking on the button and entering one.




For testing considerations, I would use the ChatGPT functionality by asking it whatever you please. To test the integration functionality, I would sign up with an account on https://openweathermap.org/ and get a free API key. Once the API key is entered, you may query Squire, for example: “What is the weather like in Florida?”