Misty Receptionist

Max McGill

Problems-

Connecting PC to Misty

Misty has enterprise wifi already setup but the IP address occasionally changes

Solution:

Plug an empty usb with a folder labelled misty into the back of misty and take it out after one minute.

Check the new file called deviceconfig.json

Ctrl F and type IP to find the IP address and enter it into the code

Misty-SDK installation and MistyPy

MistyPy is not a full misty API, it only contains the pythonicated names and isn’t necessary to run custom misty programs in python

It is simply a naming convention designed to be installed over misty-sdk

* misty.TakePicture() [misty-sdk]
* misty.take\_picture() [MistyPy]

Official misty python programs use MistyPy naming conventions

Solution:

Install misty-sdk only

Image Data

Misty cannot simply save image data from her to the pc so image data can’t just be processed.

Solution:

Take picture with misty, store the base64 data, then convert the base64 data into usable data.

with open(file, "wb") as f:

f.write(base64.b64decode(read.json()["result"]["base64"]))

f.close()

This decodes and stores the data in a png file.

Chat GPT Subscription

Chat GPT requires payment to use its API key services

Solution:

GPT4ALL is a local GPT-model-running API which means you can run GPT models on your hardware as opposed to in a browser from an organization’s server. Using the llama models trained by Meta, we have access to a conversational AI

File access denied

Libllamamodel or similar model files cannot be opened

Solution:

Nvidia Cuda installation

Windows:

<https://developer.nvidia.com/cuda-downloads?target_os=Windows&target_arch=x86_64&target_version=11&target_type=exe_local>

during setup select express or if you select custom make sure you install drivers

**WARNING:** This will overwrite your current pc drivers (display and audio). However, these drivers are necessary for it to work.

Linux:

sudo apt install nvidia-cuda-toolkit

Temporary Speech to Text Name

For whatever reason, the temporary speech to text algorithm automatically expands my name “Max” into “Maximum”. This occurs because it reads “Max.” with the period at the end as a shortened word and thus tries to expand it.

Solution:

This implementation for speech to text is only a temporary feature for testing while Misty is not accessible.

### Besides its not a bad nickname lmao :]

Code Notes

This section is to provide explanation to some more general aspects of the code that isn’t important to its functionality but helps the reader to better understand the code.

Dev Commands

Dev commands are a class of debug features.

Dev commands are used all over the program and are not for the user communicating with Misty but actually are for helping the operator to understand what Misty is doing.

They are located within the dev\_commands class. Depending on whether Misty is connected a specific subclass of this is chosen.

All dev commands output a short string like “GPT” or “Global” which shows what aspect of the program is being run where global refers to the main function, GPT refers to the conversational capabilities, and QR refers to the QR detection function.

This is followed by a line separator (‘ | ‘) and then the system message. The message can either be debug notes or simply outputting user input/ Misty Speech.

Dev\_speak() does the same as the dev() command but it also uses Misty’s speech API for text to speech or if Misty isn’t connected then it uses a custom library.

Dev\_inp() provides formatting for user input so that it has a label like ‘GPT’ and ‘Global’. Primarily ‘Usr’ is used to indicate user input.

# need to test mistys microphone to see whether using her speech to text api is worth it

Methods

Dev\_Commands

Dev\_commands is a class that contains methods for printing a formatted statement to the terminal [ dev() ], taking and formatting the users text input [ dev\_typeinp() ], and updating the visitor\_count and ID list in visitor\_info.json [ update\_count() ].

Dev\_nomisty

Dev\_nomisty is a subclass of dev\_commands which uses custom libraries to replace some of Misty’s functions for when she isn't connected.

Dev\_speak():

This uses the pyttsx3 library which can be used to say anything using your pc’s default sound output option. It takes a string as input and prints the string to the terminal and says it aloud.

Dev\_inp():

This uses the speech\_recognition library that can be installed using pip install SpeechRecognition. This speech to text algorithm uses your pc’s default microphone and uses Google's speech recognition API (requires Internet) to return the user input.

Dev\_misty

Dev\_misty is another subclass of dev\_commands which uses Misty’s API as input and output.

Dev\_speak():

Dev\_speak does the same as in dev\_nomisty in that it takes string input and outputs it to the terminal and as audio. However, with misty connected it can use Misty's function misty.speak(“”) which uses her text to speech program.

Dev\_inp():

Dev\_inp() uses the misty.capture\_speech command from Misty’s API which returns an audio clip of everything the user said after saying “Hey Misty”. This audio clip is then pushed through Google's speech recognition API and returned.

QR

Take a picture using the Misty API and store its base64 encoded data as a variable ‘read’. Decode that data using the base64 library and append it into a new file labelled ‘qr\_camera.png’.

Push that image file through a function that finds the QR code with in it. If no QR code is found, then return an error. Otherwise, return the ‘data’ variable which contains the QR information.

If ‘data’ is equal to ‘quit’ then close the QR function returning Null. Otherwise, if the data is not empty check the QR code which should contain the user’s information and check whether it is an id that has already been viewed. If it hasn’t been checked, increment the visitor count, add their id to the viewed list, and return the name and id number.

If by this point no valid QR code has been located, then go back to the start and take another picture.

GPT

Greets the user and lets them ask anything. Enters a loop so they may ask multiple questions or encase something goes wrong. Take user input, generate a response using the local GPT model, and then return it to the user through the terminal and text to speech.

Enter a new follow up loop that asks if they have any more questions. If they do, then go back and let them ask another question. If not, then say goodbye to the user and exit the function. Otherwise, if the input was not valid then ask if they have any more questions again.

MAIN

Loads the GPT Model (Llama 3.2-1B), then asks the user if they wish to use the text only interface if they wish to run the program silently. If so use the text only developer commands, otherwise attempt to connect to Misty and use the misty developer commands. If that fails then use the no\_misty developer commands.