PepperAI

AI chat for Pepper robot

Usage & Developer Guide

Version: 1.3.1
Last Updated: 19.6.2025
Platform: Android (6)
Created by: Juho Maijala, Topi Nikula and Luka Matikainen

Table of contents

Introduction

PepperAIBot is an Android app that lets you chat with Pepper, the humanoid robot from SoftBank Robotics.

The app is built to converse with the robot without need for complicated setup or too much technical know-how. Just add an AI API of your choosing and start chatting.

What can you do with it?

- **Talk to Pepper using your voice.** No typing required.
- **Get AI-generated answers.** Pepper responds using large language models for human-like replies.
- **Expand it if you're a developer.** The app is also designed to be extended with your own code if you want to build more features.

Who is it for?

- People who use Pepper in classrooms, shops, events, or public spaces.
- Developers who want to add voice and AI capabilities to their Pepper projects.
- Anyone curious about making robots talk more like humans.

(Disclaimer: This was a school project and we (the developers) did not have background in Android app coding so there could be some bugs in our code.)

User guide

Step by step guide to get you started.

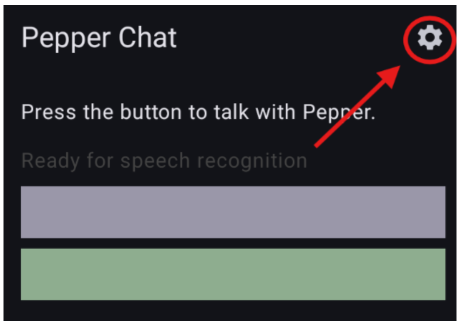
Step 1: Preparing the Pepper robot

- Turn on the robot. (There is power button under the tablet)
- Connect it to internet.
- Download and install the latest PepperAIBot apk from Github: https://github.com/Juh0m/PepperAIBot/releases
  (If you can't download and install it this way, there is guide for installing it from another device in the Guide for Developers.)
- Launch the app: PepperAIBot.

(If Pepper robot's head doesn't rise you have to restart it.)

Step 2: Preparing the PepperAIBot app

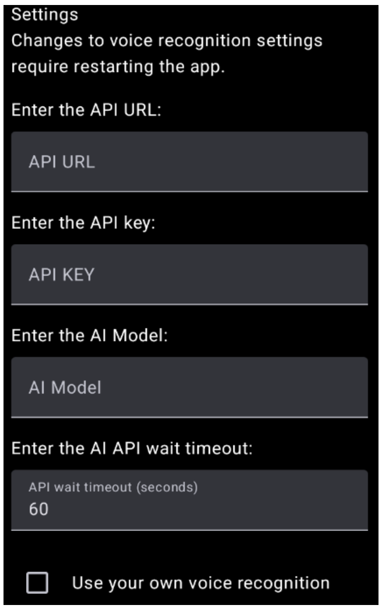
- When the app launches, open the settings panel from the top right of your screen.


Pepper Chat

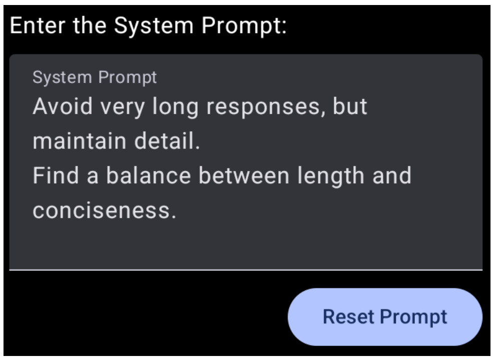
Press the button to talk with Pepper.

Ready for speech recognition

- Add API URL (for example, https://api.com/), API Key (if needed) and AI model (if needed).
- Pepper doesn't have the capacity to host it's own AI chat locally so you need to have AI chat API ready for the app to function. We recommend buying a subscription to OpenAI's API or hosting your own. Guide to hosting your own API can be found in this document.

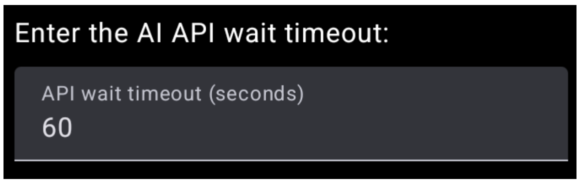

Settings
Changes to voice recognition settings require restarting the app.
Enter the API URL:
API URL
Enter the API key:
API KEY
Enter the AI Model:
AI Model
Enter the AI API wait timeout:
API wait timeout (seconds)
60
☐ Use your own voice recognition

- If you want to change the system prompt you can do it here:

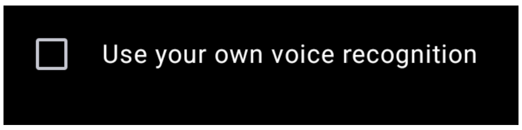

Enter the System Prompt:
System Prompt
Avoid very long responses, but maintain detail.
Find a balance between length and conciseness.
Reset Prompt

- Reset Prompt button just sets the prompt back to the default Pepper prompt
- System prompt tells the AI how it should act or how it should speak. It will apply through the whole conversation until changed.

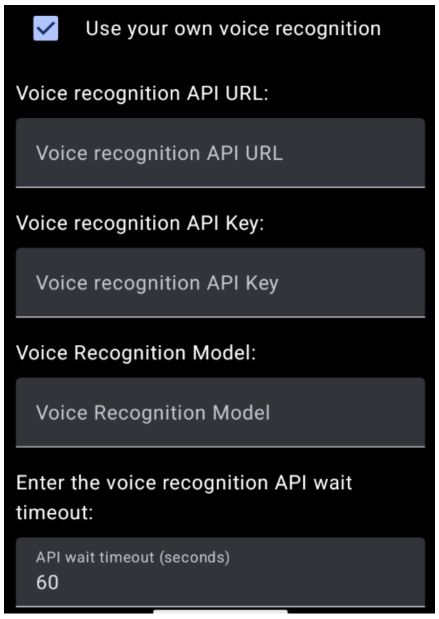
- Change the API timeout time if needed
- This adjusts the time before the API request gives up.


Enter the AI API wait timeout:
API wait timeout (seconds)
60

- If you want to change voice recognition model click the checkbox
- Peppers resources can't handle very good voice recognition models, so it's recommended that you use your own voice recognition model. Guide for hosting your own model can be found in this document.



- Add voice recognition API URL, API Key and Model (API Key and Model not implemented yet)
-



☑ Use your own voice recognition

Voice recognition API URL:

Voice recognition API URL

Voice recognition API Key:

Voice recognition API Key

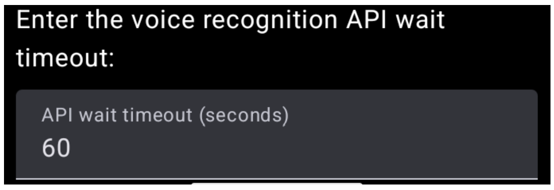
Voice Recognition Model:

Voice Recognition Model

Enter the voice recognition API wait timeout:

API wait timeout (seconds)
60

- Change the API timeout time if needed
- This adjusts the time before the API request gives up



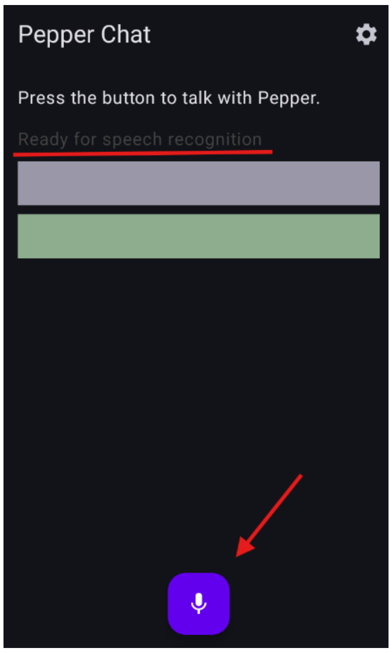
Enter the voice recognition API wait timeout:

API wait timeout (seconds)
60

- If you added or edited your own voice recognition API settings, you have to restart the app for it to take effect.
- Press the back button on the tablet to return to app's main screen.

Step 3: Start chatting

- When the speech recognition model is loaded, press the microphone button and start talking. (Default speech recognition model only understands English)
- Just tap the microphone button, no need to press it down
- Pepper listens and when it says "stopped" the AI will start generating an answer. (If you use your own voice recognition model you need to click the button again when you stop talking.)


Pepper Chat
Press the button to talk with Pepper.
Ready for speech recognition

- Pepper needs and eye contact for it to speak. It needs to focus on you.
- Depending on your API's hardware, internet speed or the prompt given the latency could be quite long, so just wait a moment and it should give you an answer or the API will timeout. If the API times out increase the timeout time or check for errors.

Troubleshooting:

1. Check for typos in settings.
2. Check if your API is getting the requests.
3. Check Pepper's internet connection
4. Restart the app.
5. Have you tried to restart Pepper?

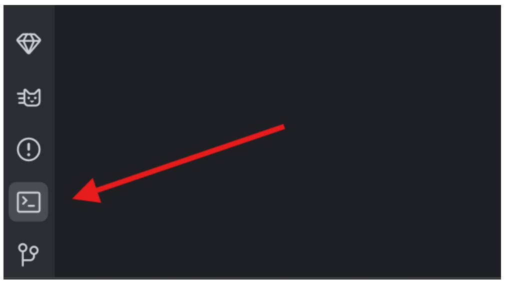
Guide for developers

The code can be found in https://github.com/Juh0m/PepperAIBot/tree/main
This guide is made primarily for Windows users. If there's something in this guide that does not apply to Linux, you need to figure that out on your own.

PepperAIBot is built with Android Studio. PepperSDK was not used for this project, and the interaction with Pepper is done using the QiSDK library. It's currently only compatible with OpenAI's API or other similar APIs, for example LM Studio in OpenAI compatibility mode.

When uploading the app, you can either use either the Run button in Android studio or adb commands. Both require adb connection to Pepper. You can also connect with PepperSDK, but that is abandoned software and does not work well, and as such I can't recommend its usage.

To get adb connection, you need to be on the same LAN as Pepper. You need it's IP, which you will find in the notifications of Pepper's tablet. Then, open the terminal in the bottom left corner of Android Studio.



In the terminal, type "adb connect {pepper's IP}.

It should say "Failed to authenticate to {pepper's IP}. This is expected. You will also see a popup on Pepper's tablet asking if you want to enable USB debugging. Press "OK". You are now connected to Pepper and you should see "ARTNCORE LPT_200AR" on the left side of the run button. If this did not work, see possible issues on the next page:

Possible issues:

- You don’t have SDK platform tools installed. This is required for ADB. You can download this in Android Studio -> Tools -> SDK Manager -> SDK Tools -> Android SDK Platform Tools.

- (Windows only) If you have SDK platform tools installed but ADB does not work, check that adb is in %PATH%. You can see path by typing echo %PATH% into a terminal. If you don’t see platform tools there, add it.

- You didn’t type the right IP address. In the notifications, there are two different IP:s. Use the one that is *not* for robot browser. If you did use that, check for typos.

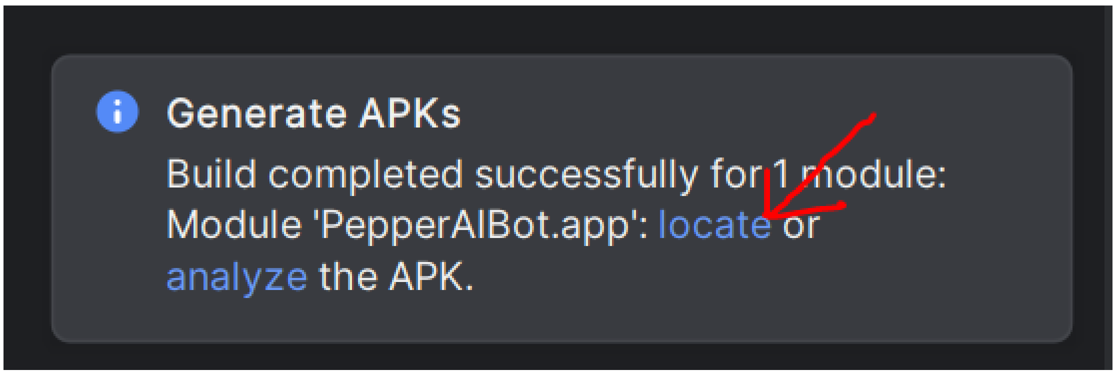
- Pepper has USB debugging disabled in it’s settings. Enable USB debugging.

- You aren’t connected to the same LAN as Pepper. Connect to the same LAN.

- Sometimes Pepper might just refuse connection for seemingly no reason. Try rebooting Pepper, it might help.

To run the app on Pepper using the “run” button, press the “run” button. Android studio will build the app and then attempt to run it on Pepper. This might work, but in our experience more often than not this does not work, as you will get a timeout after 5 minutes. Due to Pepper being old and very slow, this is often not enough time to upload the app. If it worked, the app will be installed on Pepper and will be automatically opened.

If that did not work, you can use adb to upload the .apk instead. First, press Build -> Generate app bundles of APKs -> Generate APKs. This will compile the app into an .apk file. After it finishes, you'll see a popup in the bottom right corner. On that popup should be a "locate" button.



Generate APKs
Build completed successfully for 1 module:
Module 'PepperAIBot.app': locate or analyze the APK.

Press that and it'll open file explorer to the directory the apk is in. In there, open a terminal. In that terminal, type "adb push –a –p "{name of your apk}" "/sdcard/Download/".

(For example: adb push –a –p "PepperAIBot.apk" "/sdcard/Download/)

–a preserves permissions, -p displays progress. This may take a while.

You may push the file into another directory, but in our experience trying to install the apk on pepper in other directories just gave a "this file can't be opened" error, and the only way to install an apk seems to be using the Downloads app on Pepper.

If this error happens even if you pushed the apk to the download folder, try copying the file from the download folder to the download folder. That may seem pointless, but for some reason it worked for us

Pepper:

To talk pepper needs to have eye contact with you. When Pepper gains eye contact, the function onRobotFocusGained is called. There we can get qiContext, which is necessary for interaction between the app and Pepper (for example making Pepper talk). This is stored in a variable that can be used in other functions, such as robotSay().

Pepper can talk in multiple languages, depending on the locale set in code. You can find how this is done in the robotSay() function on our code.

You can also change how fast Pepper talks, but we haven't implemented this, so you'll have to do that yourself.

HTTP:

For HTTP requests we use okhttp and Retrofit.

If you want to make API requests to APIs that aren't compatible with OpenAI's API, you need to create new data classes and interfaces.

UI:

The UI is created using Jetpack Compose. We currently have a forced dark mode as we don't have time for anything else. The UI code is at the bottom of each code file.

Voice recognition on Pepper(speech-to-text):

The speech-to-text is done using Vosk and the small model-en-us model. If you want to use a different Vosk model, replace the model-en-us.zip file in the assets directory and change the extractAssets() function call in initVosk to use the new zip file. The big models need more memory than Pepper has, so they cannot be used.

The models are available on https://alphacephei.com/vosk/models

The app does not currently support any other speech-to-text models on Pepper. In case you want to use something else, use a self-hosted one instead. For that, we made an API for that and instructions for using it are in this document.

Self-hosted speech-to-text

The application records an .aac file and sends it to the url specified in the settings. The API should receive the .aac audio file, transcribe it to text and return the text.

If you use this option, you aren't restricted to english only, for example the API we created supports multiple languages. That API is available at :link

This does not work on Android 10+ as write_external_storage does not give these permissions anymore (This does not matter on Pepper as the tablet is Android 6).

Other:

The values for all settings are stored in sharedPreferences.

Host your own AI chat API

PepperAI uses large language models (LLM) to process text and respond to the user's prompts. An LLM is the type of "AI" that popular programs like ChatGPT use to produce text. Pepper itself cannot run an LLM due to its limited and aging hardware, hence why a separate personal computer is required to run one. This section of our guide will focus on the hardware and software requirements of running a large language model locally.

Hardware

The most important component in a computer for running an LLM is the graphics card, often referred to as a GPU. Due to a GPU's importance in these tasks, low-end systems with integrated GPUs may not be able to run an LLM sufficiently fast, if at all. Our software was chosen with NVIDIA GPUs in mind. NVIDIA is currently the most convenient choice of GPU manufacturer for LLMs, as most LLM backend software is designed with NVIDIA in mind. We recommend using newer NVIDIA GeForce RTX-series GPUs for PepperAI to maximize performance. RTX 20-series and later is ideal due to their increased focus on AI workloads compared to previous generations. For our development process, a GeForce RTX 5070 Ti was used.

A GPU's performance and VRAM affect its performance in LLM workloads. VRAM is crucial for storing the LLM itself, hence why more VRAM is beneficial. The RTX 5070 Ti mentioned above has 16 gigabytes of VRAM. We do not recommend GPUs with less than 8 gigabytes of VRAM for running PepperAI, as their low VRAM capacity limits the size of both the LLM and speech-to-text models which can be run. A smaller LLM and speech-to-text model result in more factual errors, less intellect and more misheard words. Attempting to run overly large models in too little VRAM can lead to severe reductions in speed and responsiveness. For example, our test hardware often takes less than ten seconds to respond to a user's prompt if both the LLM and STT model fit in its VRAM. If they do not fit, Pepper may take up to 20-30 seconds to respond, with the time only increasing if Pepper's response is particularly long.

Software

For this project, we chose LM Studio as an easy-to-use Windows-compatible program as the platform to run LLMs on. LLMs require a backend program to be used, which have varying levels of features and complexity. LM Studio is a more user-friendly option. It offers a server function, letting the user have their LLM be usable and reachable by another computer, or in this case, Pepper. This feature is the basis of Pepper's LLM functionality.

Our chosen LLM for testing was Google's Gemma 3 12B QAT. In LM Studio, the model is named "gemma-3-12b-it-qat" and was published by lmstudio-community. It was chosen for its intellect and accuracy despite being able to run on a consumer GPU with 16 GB of VRAM. If you have access to a GPU with 16 GB or more of VRAM, we recommend using the model mentioned above. However, you must consider the VRAM limitations of running both an LLM and an STT model on the same GPU. Both together must fit within the VRAM of the GPU for optimal performance. Reducing the size of the LLM results in less intelligent and accurate answers. Reducing the size of the STT model results in more misheard words, and possibly slower speech recognition.

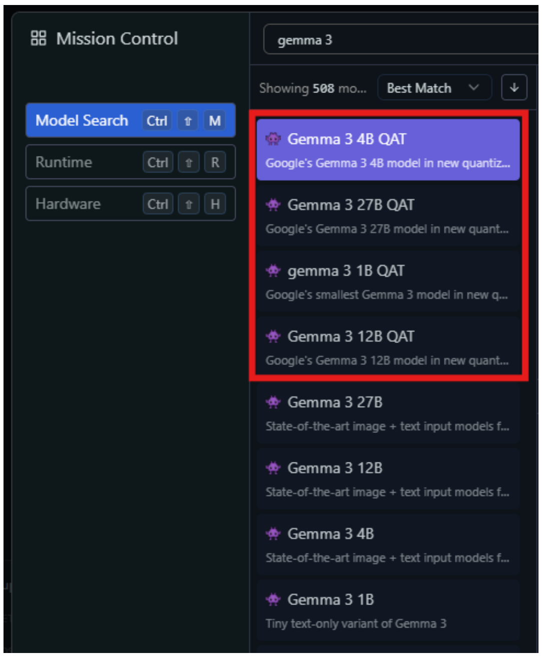
Guide:

Step 1: Downloading and installing LM Studio

- Go to the official site and download LM Studio.
- Install LM Studio.
- When prompted to install LM Studio's suggested LLM, click "Skip" from the top right corner of the window. This model is unnecessary.

Step 2: Downloading and preparing the LLM

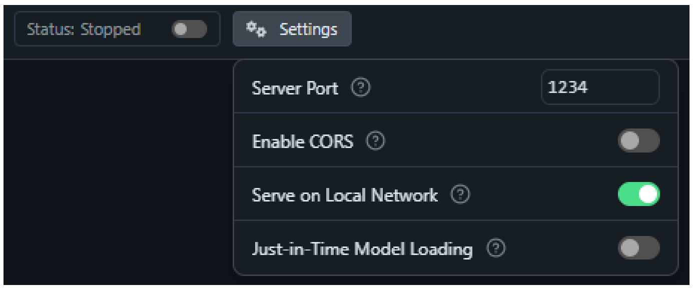
- Select “Power User” from the bottom of the window. This ensures the menus in later instructions are not hidden.
- Enter the “Discover” tab from the left side of the window.
- Search for “Gemma 3 QAT” in the search bar to find the right models. The right models on the list may have either “google” or “lmstudio-community” under them.


Mission Control
gemma 3
Showing 508 mo...
Best Match
Model Search Ctrl ⇧ M
Runtime Ctrl ⇧ R
Hardware Ctrl ⇧ H
Gemma 3 4B QAT
Google's Gemma 3 4B model in new quantiz...
Gemma 3 27B QAT
Google's Gemma 3 27B model in new quant...
gemma 3 1B QAT
Google's smallest Gemma 3 model in new q...
Gemma 3 12B QAT
Google's Gemma 3 12B model in new quant...
Gemma 3 27B
State-of-the-art image + text input models f...
Gemma 3 12B
State-of-the-art image + text input models f...
Gemma 3 4B
State-of-the-art image + text input models f...
Gemma 3 1B
Tiny text-only variant of Gemma 3

- Download your LLM of choosing.

Step 3: Setting up the LLM API server

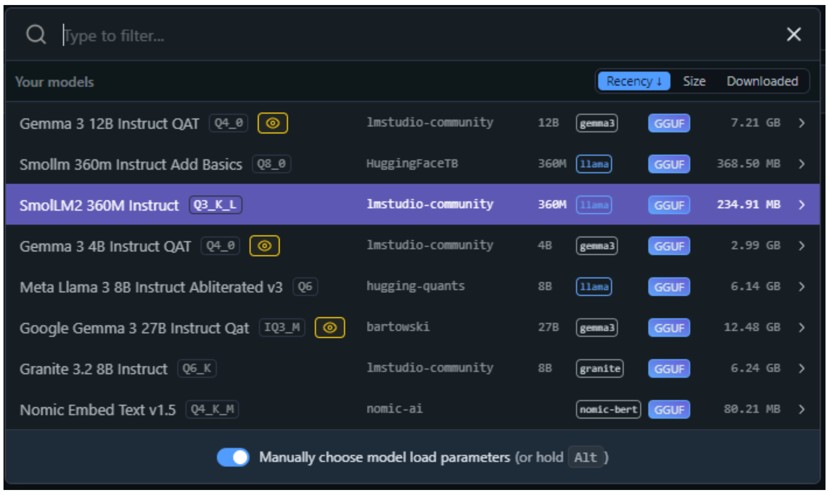
- Enter the “Developer” tab on the left side of the window.
- Click “Settings” near the top left of the window.



Server Port 1234
Enable CORS: off
Serve on Local Network: on
Just-in-Time Model Loading: off

- Deselect “Just-in-Time Model Loading”.
- Select “Server on Local Network”.
- Click “Select a model to load”.

- Select the LLM you downloaded in step 2.



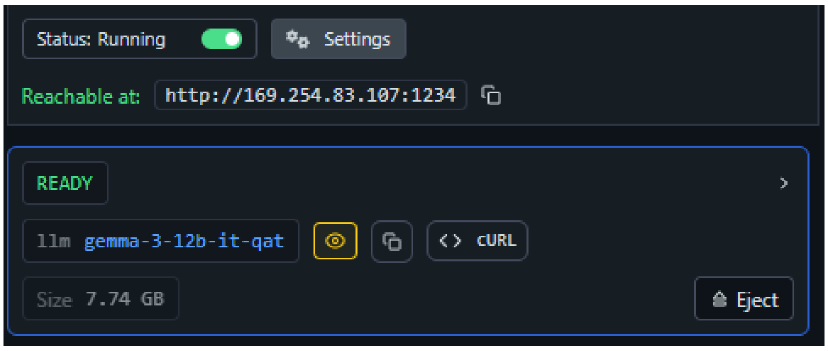
Type to filter...

Your models — Recency ↓ | Size | Downloaded

Gemma 3 12B Instruct QAT	Q4_0	lmstudio-community	12B	gemma3	GGUF	7.21 GB
Smollm 360m Instruct Add Basics	Q8_0	HuggingFaceTB	360M	llama	GGUF	368.50 MB
SmolLM2 360M Instruct	Q3_K_L	lmstudio-community	360M	llama	GGUF	234.91 MB
Gemma 3 4B Instruct QAT	Q4_0	lmstudio-community	4B	gemma3	GGUF	2.99 GB
Meta Llama 3 8B Instruct Abliterated v3	Q6	hugging-quants	8B	llama	GGUF	6.14 GB
Google Gemma 3 27B Instruct Qat	IQ3_M	bartowski	27B	gemma3	GGUF	12.48 GB
Granite 3.2 8B Instruct	Q6_K	lmstudio-community	8B	granite	GGUF	6.24 GB
Nomic Embed Text v1.5	Q4_K_M	nomic-ai		nomic-bert	GGUF	80.21 MB

Manually choose model load parameters (or hold Alt)

- Click the "Status" toggle. This option should say "Running" for the server function to be active.



Status: Running | Settings

Reachable at: http://169.254.83.107:1234

READY

llm gemma-3-12b-it-qat | cURL

Size 7.74 GB | Eject

Your computer is now hosting the LLM at its own IP address. LM Studio uses port 1234 by default, but this can be changed in the settings menu mentioned above in step 3. Please note that Pepper and your host computer *must* be connected to the same network. LM Studio often shows your computer's IP address and port, but it may sometimes be inaccurate. In this case, find your IPv4 address in the computer's network settings, and use it instead with the port number added to the end.

Step 4: Recommendations

In the "Load" tab of the model settings, you can enable different features that affect the performance of the model. We recommend lowering the context length to 1024-2048, as higher values have no benefit on Pepper. A higher value would only use VRAM and risk slowing Pepper down.

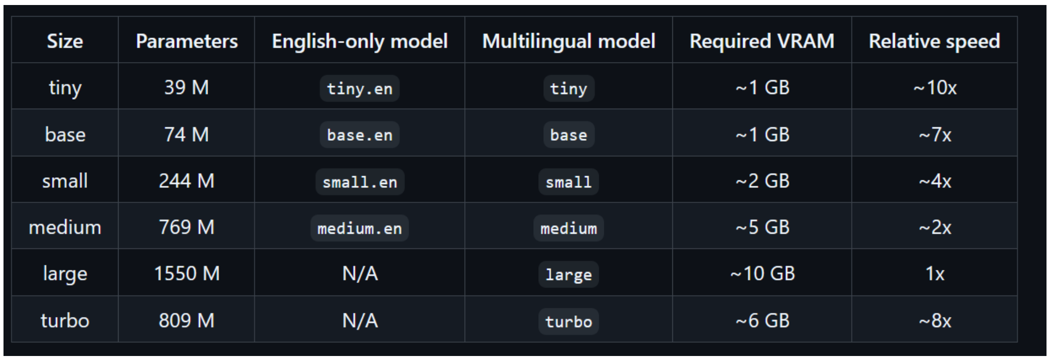
Host your own speech-to-text API

We strongly recommend using a STT (speech-to-text) API instead of the Vosk STT that runs on Pepper. Due to Pepper's weak and old hardware, it is impossible to run a good STT on it, so the one on it makes a ton of errors and just does not work properly if you have a strong accent. Also, it only understands English. Fortunately for you, we made our own API using Whisper, a STT model by OpenAI, which can be found here: Juh0m/Whisper-stt-api-pepper: Python Whisper STT API for our Pepper app

This API takes between 1 and 8 GB of VRAM, depending on your needs. Out of the box, without changing the model it will take about 6 GB. If you don't have enough for that, you can change models.

The parameter count roughly corresponds to how good the speech recognition is, as the parameter count increases it will make less mistakes. Relative speed is how fast it is compared to the large model. Required VRAM is how much VRAM it will take, the rest you can ignore.

We strongly recommend the turbo model as it finds a great balance between speed and accuracy, while still taking less VRAM than the large model. We haven't tested the smaller models, so we can't guarantee that they are accurate.

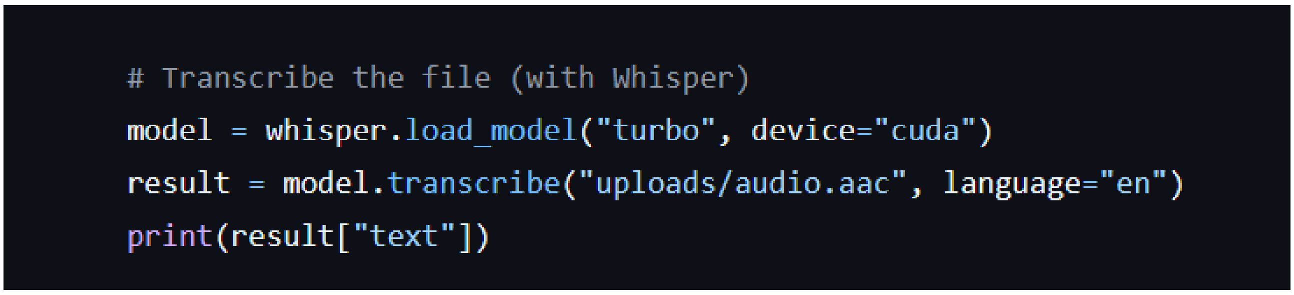


Size	Parameters	English-only model	Multilingual model	Required VRAM	Relative speed
tiny	39 M	`tiny.en`	`tiny`	~1 GB	~10x
base	74 M	`base.en`	`base`	~1 GB	~7x
small	244 M	`small.en`	`small`	~2 GB	~4x
medium	769 M	`medium.en`	`medium`	~5 GB	~2x
large	1550 M	N/A	`large`	~10 GB	1x
turbo	809 M	N/A	`turbo`	~6 GB	~8x

(Do the step-by-step guide first before changing this)

To change the model, open the api.py file in any text editor (For example, notepad or notepad++).

Locate these lines in the code. To change the model, change the “turbo” seen in the below image to the model you want. Then, save the file. Next time you start the API, the model will automatically be downloaded.



```
# Transcribe the file (with Whisper)
model = whisper.load_model("turbo", device="cuda")
result = model.transcribe("uploads/audio.aac", language="en")
print(result["text"])
```

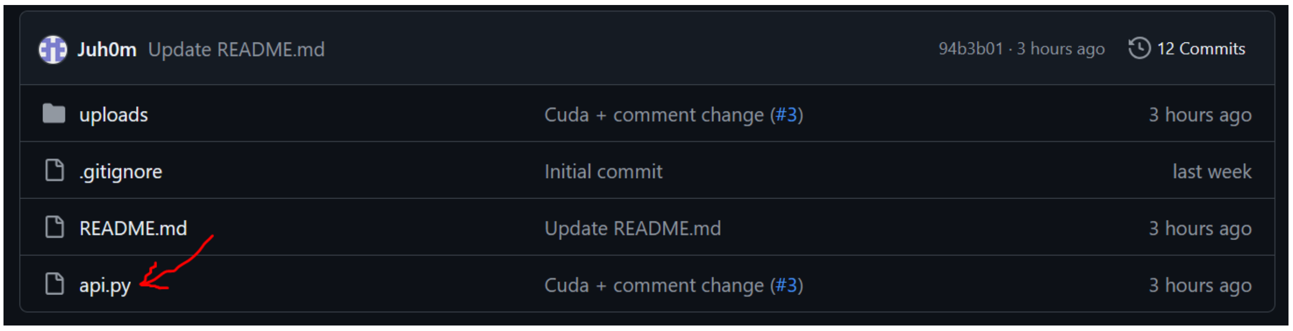
Here’s a step-by-step guide on how to get it running on your machine:

This guide is primarily meant for Windows users. Linux users should know the Linux equivalents of the commands presented here. For Mac users, I unfortunately cannot provide help with this topic.

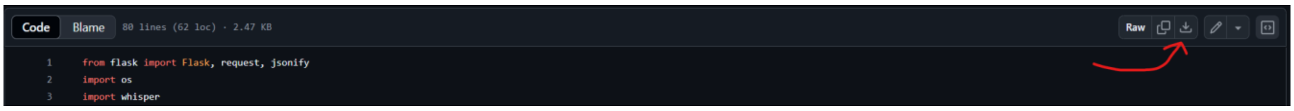
Firstly, you need to install Python 3.11 (or any other version between 3.8 and 3.11).

If you already have a newer version of Python, you don’t need to uninstall that, but you will need to use different commands for installing packages (which will also be provided on the guide). You can check the newest version by opening a terminal (For example, command prompt. To open command prompt, you can search “cmd” using the Windows search). In there, type py –version. If it gives an error, you likely don’t have Python. In that case, go to https://www.python.org/downloads/release/python-3110/. There, click on the Windows installer (64-bit).

Next step is to get the api.py from our GitHub repository at Juh0m/Whisper-stt-api-pepper: Python Whisper STT API for our Pepper app. You only need the single file, and the easiest way to download that is to click on it and press the “download raw file” button. You may also clone the repository if you know how to do that.



Juh0m Update README.md	94b3b01 · 3 hours ago	12 Commits
uploads	Cuda + comment change (#3)	3 hours ago
.gitignore	Initial commit	last week
README.md	Update README.md	3 hours ago
api.py	Cuda + comment change (#3)	3 hours ago



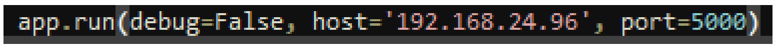
```
from flask import Flask, request, jsonify
import os
import whisper
```

Now, save the file you downloaded somewhere outside of the download folder.

Next, we'll install some required dependencies. You'll need to open a terminal. Again, this can be done by typing "cmd" into the search bar. In the terminal, copy these commands:

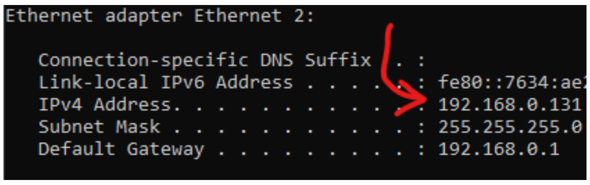
Use the blue commands if you have a newer python than 3.11 also installed, otherwise use the green ones.

1. pip install flask (py –3.11 -m pip install flask) This installs flask for python, which we used for the API calls.
2. pip install openai-whisper (py –3.11 -m pip install openai-whisper) This is the speech-to-text model we used. Installing this one may take a while.
3. If you want to use your computer's graphics card instead of the cpu, you need to install pytorch cuda toolkit, for that use pip install torch torchvision torchaudio –index-url https://download.pytorch.org/whl/cu128 (py –3.11 -m pip install torch torchvision torchaudio –index-url https://download.pytorch.org/whl/cu128) If this does not work because your graphics card does not support it (or if you don't have one), you can do everything on the cpu instead. For this to work, change
4. Open the api.py file you downloader earlier in any text editor. If you don't have anything else notepad will do the job. At the bottom of the file, you will see this line:

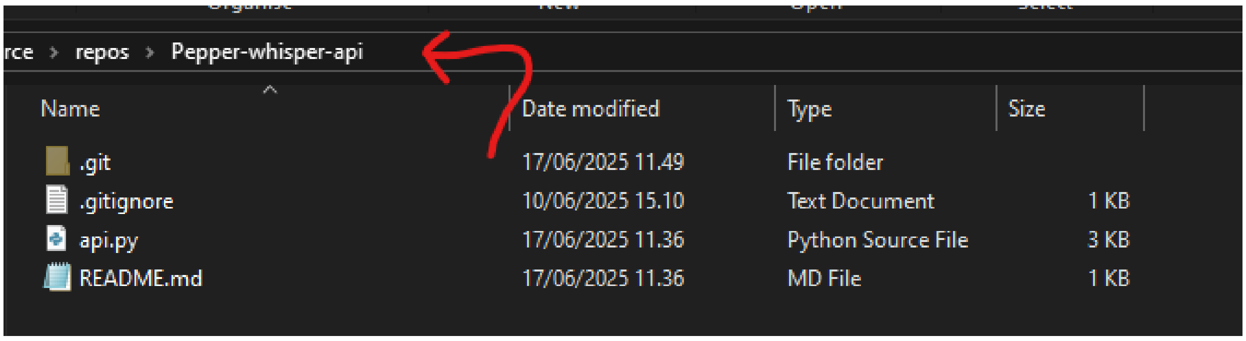
   ```
   app.run(debug=False, host='192.168.24.96', port=5000)
   ```

   This IP has to be changed to your IP. You can find your IP by opening a terminal and using the command "ipconfig". There, find your IPv4 address. Replace the IP in the code to the IP you see in the terminal.

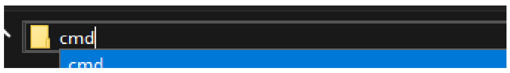
   ```
   Ethernet adapter Ethernet 2:

      Connection-specific DNS Suffix  . :
      Link-local IPv6 Address . . . . . : fe80::7634:ae
      IPv4 Address. . . . . . . . . . . : 192.168.0.131
      Subnet Mask . . . . . . . . . . . : 255.255.255.0
      Default Gateway . . . . . . . . . : 192.168.0.1
   ```

5. You can now start the API. Open a terminal in the same location where your api.py is. Type "cmd" into the bar shown on the image below to open the terminal. Press enter, and it should open.

   rce › repos › Pepper-whisper-api

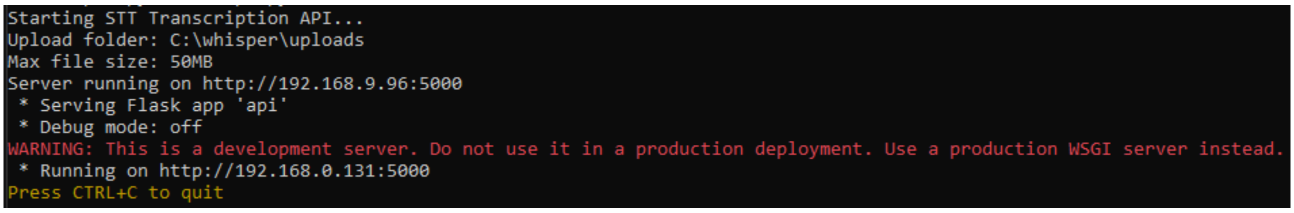
Name	Date modified	Type	Size
.git	17/06/2025 11.49	File folder	
.gitignore	10/06/2025 15.10	Text Document	1 KB
api.py	17/06/2025 11.36	Python Source File	3 KB
README.md	17/06/2025 11.36	MD File	1 KB

   cmd

   Then, type the command py api.py (or py –3.11 api.py) to start the speech-to-text.

If everything works, you'll see a message like this:

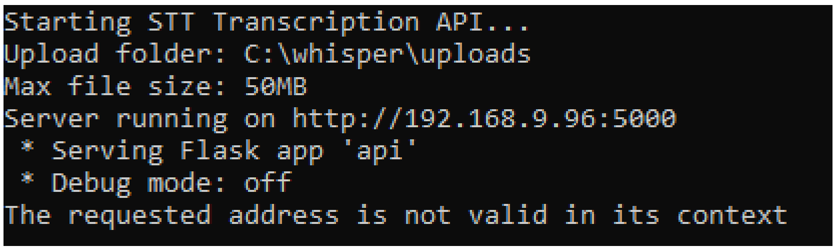


```
Starting STT Transcription API...
Upload folder: C:\whisper\uploads
Max file size: 50MB
Server running on http://192.168.9.96:5000
 * Serving Flask app 'api'
 * Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
 * Running on http://192.168.0.131:5000
Press CTRL+C to quit
```

This means the API is working correctly. Now, just make sure the checkbox on Use own voice recognition is checked on the Pepper app's settings. Then, type the URL you see in the terminal on the API url field. For example, in the image above the url would be http://192.168.0.131:5000/. Everything should now work. If something did not work, try the troubleshooting steps below:

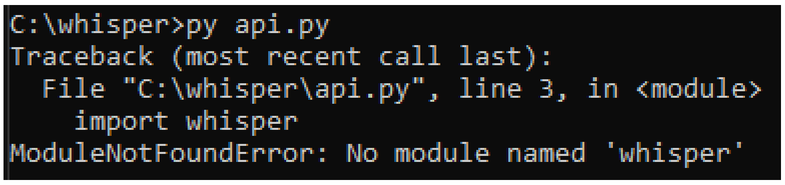
Troubleshooting:

- If the IP you set in the code is incorrect, you will see this error. Check your IP and try again.



```
Starting STT Transcription API...
Upload folder: C:\whisper\uploads
Max file size: 50MB
Server running on http://192.168.9.96:5000
 * Serving Flask app 'api'
 * Debug mode: off
The requested address is not valid in its context
```

- Whisper does not support the newest python versions, and this can cause problems if you accidentally use a newer version.



```
C:\whisper>py api.py
Traceback (most recent call last):
  File "C:\whisper\api.py", line 3, in <module>
    import whisper
ModuleNotFoundError: No module named 'whisper'
```

There are some issues with this API that we did not have time to fix:

- No rate limitations, if you send multiple requests to this API at the same time, it can take up a ton of resources and even freeze the computer. This should however never happen in normal usage with a single robot.
- Very resource-intensive and can be slow on a weak computer.
- IP must always be manually set in the code, this adds a bit of extra work when starting the API.