Solution Architecture for Gen AI

Ram N Sangwan

- **Solution Guidelines - Well-Architected principles**
- **Chat Session management**
- **Architectures for Various Use Cases**
- **Manage Token limitation**
- **Deployment Standards – Cloud or On-Prem**
- **Private GPT Standards**

Solution Guidelines - Well-Architected principles

Typically focus on key areas to ensure the technology is used effectively and responsibly.

Some insights:

Choice of Foundation Model: depends domain, data, and the specific use case.

Accessibility and Control: on Prem for full control or using them as a managed cloud service for speed and simplicity.

The decision will impact factors like infrastructure management, cost predictability, and complexity.

Solution Guidelines - Well-Architected principles

Architecture Components: data processing layer, a generative model layer, a feedback & improvement layer, and a deployment & integration layer.

Operational and Model Risks: Models must be managed to control privacy and ensure security. Risks also include potential biases in foundation models and producing unverified or misleading outputs.

Data and Infrastructure Strategy: Many organizations struggle with siloed, rapidly changing, or low-quality data, which impacts the performance of AI models. It's essential to have a solid data and infrastructure strategy to support the demands of Gen AI.

Regulatory Compliance and Ethical Considerations

Chat Session Management

Guidelines that could be useful

1. **Ensure User Privacy and Data Security**.
2. **Maintain Transparency**.
3. **Promote Ethical Interactions**.
4. **Manage Expectations**: cut-off knowledge date and its inability to access or retrieve real-time information without specific tools.
5. **Feedback Mechanism**.

Standard Architectures for various use cases

Generative Adversarial Networks (GANs)

Use Case

- Excellent for generating photorealistic images, art, and even enhancing low-resolution images.

Strengths

- Capable of producing high-quality and highly realistic outputs.

Weaknesses

- Training can be unstable and often requires careful tuning of hyperparameters.

Compared to other models, GANs can generate the most visually compelling results but are often harder to train.

Variational Autoencoders (VAEs)

Use Case

- Used for image generation, but with a focus on encoding an input into a latent space and then reconstructing it.

Strengths

- More stable training than GANs and can learn to represent complex probability distributions.

Weaknesses

- Tend to produce less sharp images compared to GANs.

VAEs are more stable and easier to train than GANs but do not match the output quality of GANs in terms of image crispness.

Long Short-Term Memory Networks (LSTM)

Use Case

- Suitable for time-series prediction, music composition, and text generation with a focus on sequences.

Strengths

- Good at capturing time-dependent properties in sequential data.

Weaknesses

- Struggles with long-range dependencies and is less effective than transformer models for longer sequences.

LSTMs are more efficient than transformers but lack the same performance on complex tasks requiring understanding of long contexts.

Auto-Regressive Models (e.g., PixelRNN, WaveNet)

Use Case

- PixelRNN is used for image generation, while WaveNet is used for generating audio, such as human-like speech or music.

Strengths

- Can produce high-quality outputs by modelling the probability distribution of a sequence one element at a time.

Weaknesses

- The sequential nature of prediction can be slow and computationally expensive

They are excellent in their respective domains (images and audio) but are generally slower than parallelizable models due to their sequential processing nature.

Comparative Summary

- **GANs** are the best for generating realistic images but are difficult to train.

- **LSTMs** are a go-to for simpler sequential tasks where computational efficiency is a concern, though they may falter with very long sequences.

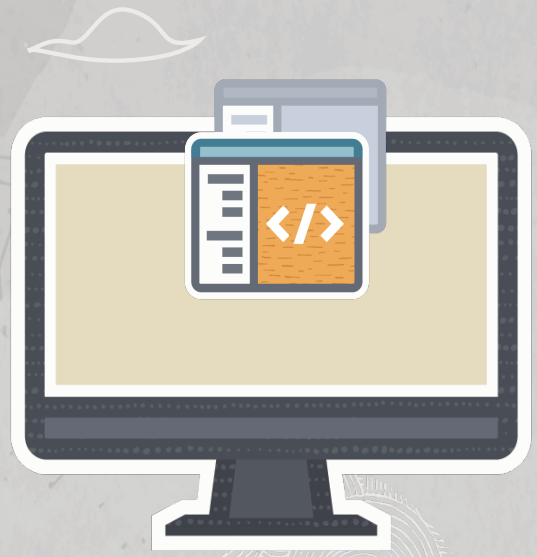
- **VAEs** offer a balance between image quality and training stability and are excellent for tasks that require understanding the underlying data distribution.

- **Transformers** are the gold standard for text-related tasks requiring the understanding of complex contexts but come with high computational costs.

- **Auto-Regressive Models** like PixelRNN and WaveNet excel in their specific domains of image and audio generation, respectively, but are not as fast as parallel processing models.

Manage Token limitations

Manage Token limitations



Summarize Content

Before inputting content into LLM, summarize it to reduce the number of tokens.

Chunking

Process each chunk individually and then synthesize the outputs.

Focused Prompts

Craft your prompts to be as focused as possible.

Iterative Refinement

Iteratively refine the output by asking follow-up questions.

Use Other Models

Consider using smaller models wherever possible, that are more token-efficient.

Increase Efficiency with Commands

Use the model's understanding of commands to execute complex tasks in a token-efficient manner.

Manage Token limitations

Pre-Processing

Use **external tools** for pre-processing tasks like extracting text from images or simplifying sentences to make them more concise.

Post-Processing

After getting the output, use post-processing to stitch together and make sense of the information if you had to break it into parts.

Pipeline Approaches

Create a pipeline that uses different models for different tasks, reserving the large language model for the most complex parts of the task.

Cache Responses

If you are likely to ask the same or similar questions, cache the responses so you don't have to use tokens for the same query again.

Optimize API Usage

If you are using an API, ensure that your API calls are optimized to send and receive as much relevant information as possible within the token limits.

Deployment Standards – Cloud or On-Prem

Deployment Standards – Cloud or On-Prem

- According to 451 Research, 90% of companies use cloud services in some form.
- However, the same report found that 60% of workloads are still run on-premise, indicating a balance between the two.
- In terms of costs, **OCI** can range from a few hundred dollars per month to tens of thousands of dollars per month depending on size of projects.
- However, on-premise solutions can have an upfront cost of several thousand dollars, with additional ongoing costs for maintenance and updates.

Cloud vs. On-Premise Hosting for AI Applications

- These are often compared to renting vs. buying a home.
- Cloud hosting is a lot like renting; the stay of AI applications is as long as the contract terms dictate.
- The maintenance of the hardware is the responsibility of the hosting provider.
- On-premise hosting, on the other hand, is like buying a home; the application can stay on the hardware as long as business requires it.

Cloud vs. On-Premise

Scalability

- On-premise hosting offers complete control over the hardware, which means that the administrators of a company can tightly control updates.
- But on-premise hosting does require advanced planning to scale hardware.
- This is because it requires time to gather the necessary data for updating it.
- Cloud resources can be rapidly adjusted to accommodate specific demands and increase the scalability of hardware.

Cloud vs. On-Premise

Security

- Full control over data stored on enterprise premises. Hosting providers must keep their systems updated and data encrypted to avoid breaches.
- Still, your company can’t be sure where your data is stored and how often it is backed up; data is also accessible by third parties.

Cloud vs. On-Premise

Data Gravity

"*The ability of data to attract applications, services, and other data towards itself*"

It is among the most important factors to be considered while choosing between cloud and on-premise platforms.

- Considering the costs of training neural networks, companies may want to deploy their AI applications on-premises.
- If the data required to build AI applications resides on the cloud, then it's best to deploy applications there.
- The location of the largest source of data for an enterprise determines the location of its most critical applications.
- Oracle cloud offering **"Cloud@Customer"** can proved to be best here.

The Case for Cloud-Based AI Applications

- Instead of building out a massive data centre, you can use the infrastructure someone else already maintains.
- One reason why AI has become so pervasive is cloud providers offering plug-and-play AI cloud services, with enough compute power and pre-trained models to launch AI applications.
- In many cases the pre-trained models or storage requirements of the cloud can be cost-prohibitive; higher GPU counts get expensive fast and training large datasets on the public cloud can be too slow.
- Still, the cloud can often be the best option in terms of “testing the waters” of AI and experimenting with which AI initiatives work best for an organization.

The Case for On-Premise AI

- There’s a whole ecosystem of tools built for on-premise infrastructure that can work with mass amounts of compute power–which can be very expensive in the cloud.
- Thus, it is more economical to do this on-premise or prefer a capital expense to an operational expense model.
- If your organization wants to get more involved in this or roll-out AI at scale, then it may make more sense to invest in on-premises infrastructure instead of consuming cloud-based services.

Private GPT Standards

What is Private GPT?

- Private GPT allows enterprises to tap into the remarkable capabilities of large language models while prioritizing privacy and security.
- Private GPT operates entirely on-premises within an organization’s own servers and data centres.

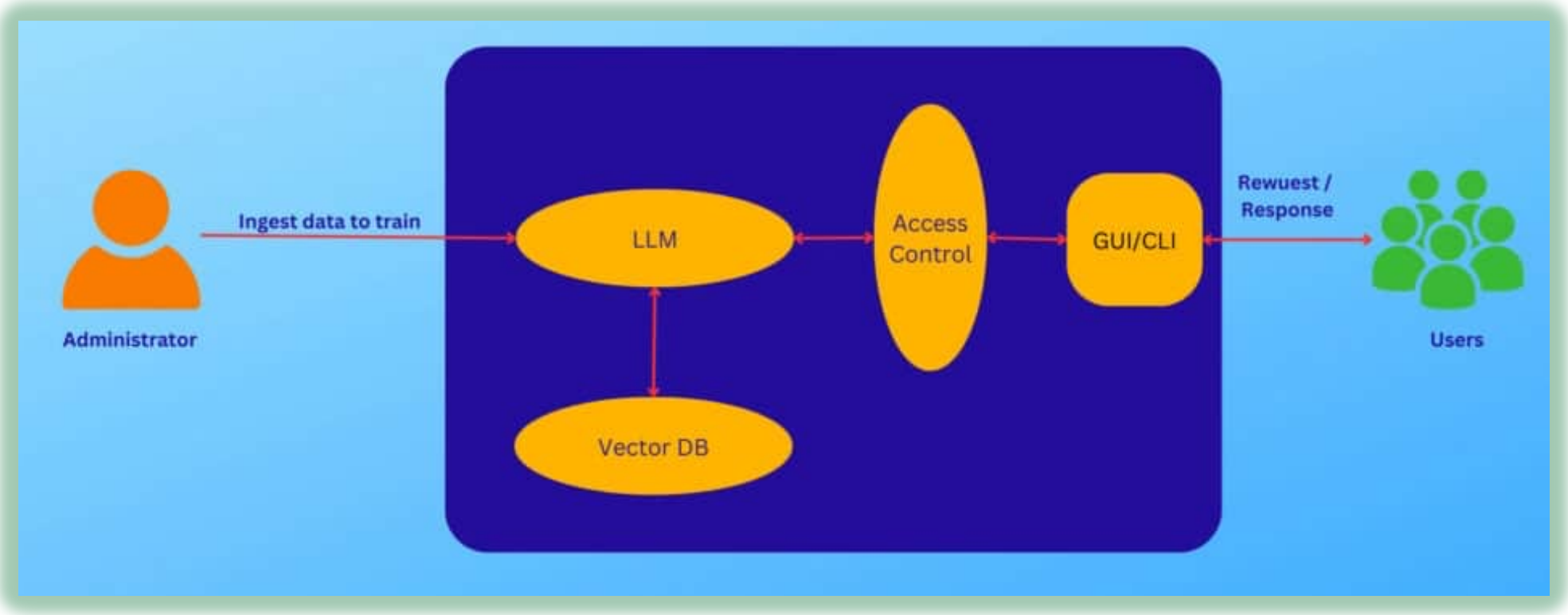
Private GPT Use Cases

- **Knowledge Management**.
- **Customer Support**.
- **Content Creation**.
- **Data Analysis**.
- **Automate Workflows**

Components of Private GPT and How Does it Work?

Private GPT is requires multiple components to work together to function.

- **Private LLMs**: Private GPT supports proprietary models like GPT4ALL and LLAMA.



Components of Private GPT and How Does it Work?

Internal Data Sources

- This includes the organization's documents, emails, chat logs, databases, and other private data sources on which the LLM can be trained.

Vector Database

- Private GPT uses Chroma vector database as the default DB.

LLM Interface

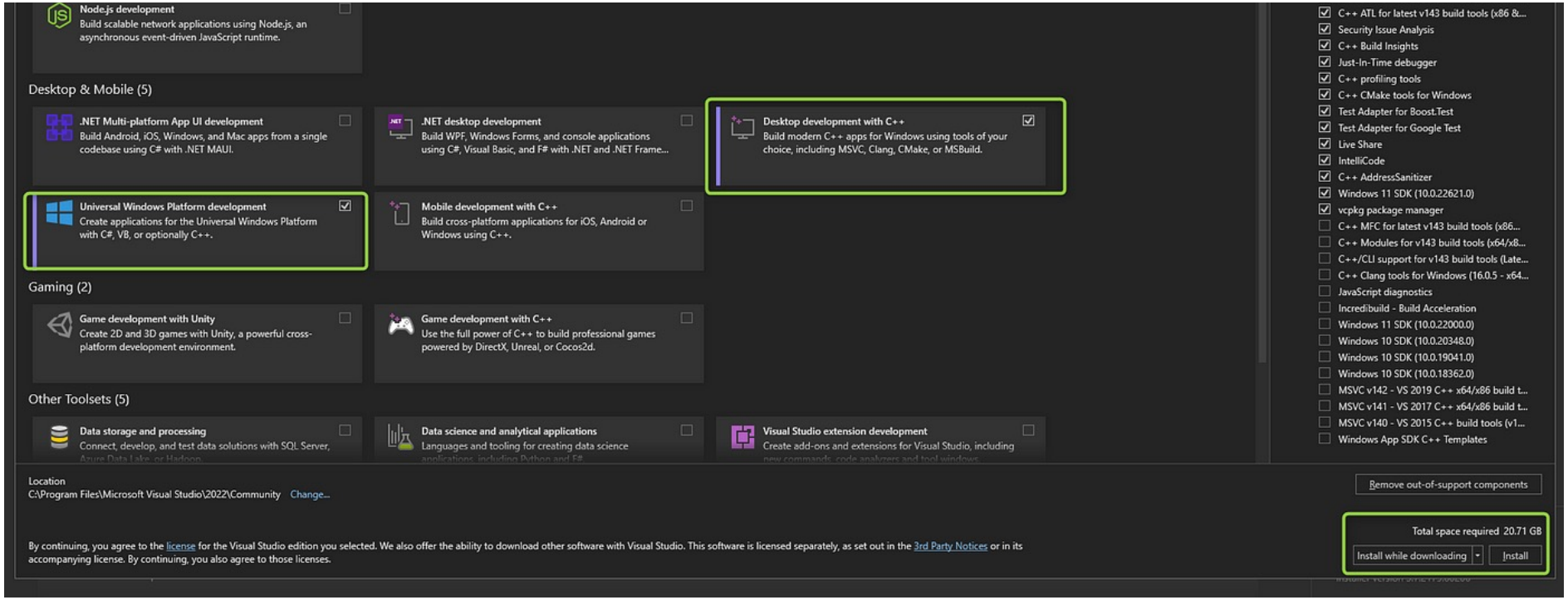
Encryption

Access Control

Private GPT on Your Windows PC

Install Visual Studio 2022 Components

1. Universal Windows Platform development
2. Desktop Development with C++



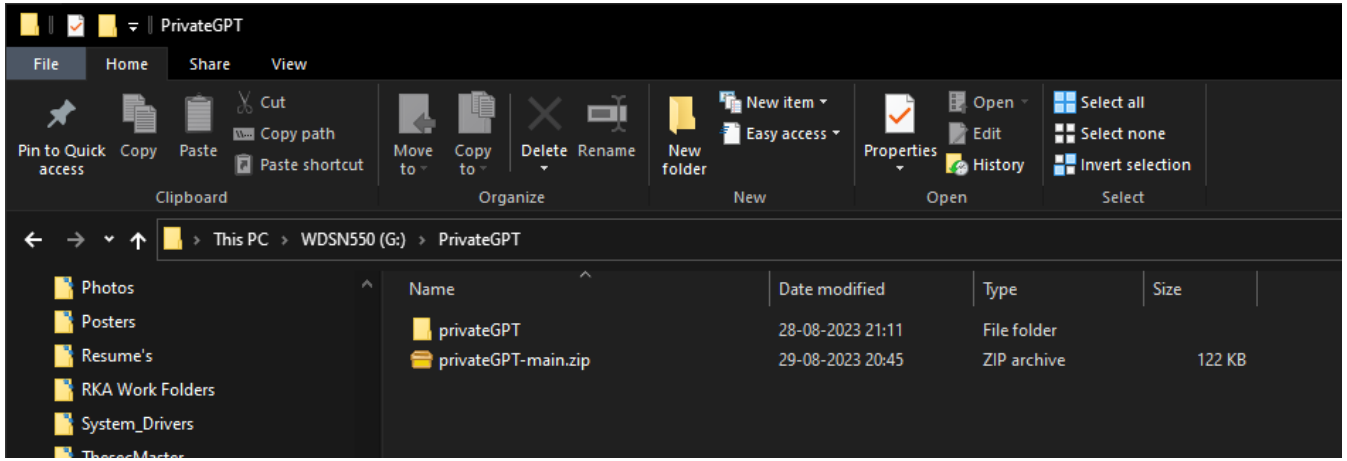
Private GPT on Your Windows PC

Private GPT requires Python 3.10 or later.

```
C:\WINDOWS\system32>python --version
Python 3.11.4

C:\WINDOWS\system32>
```

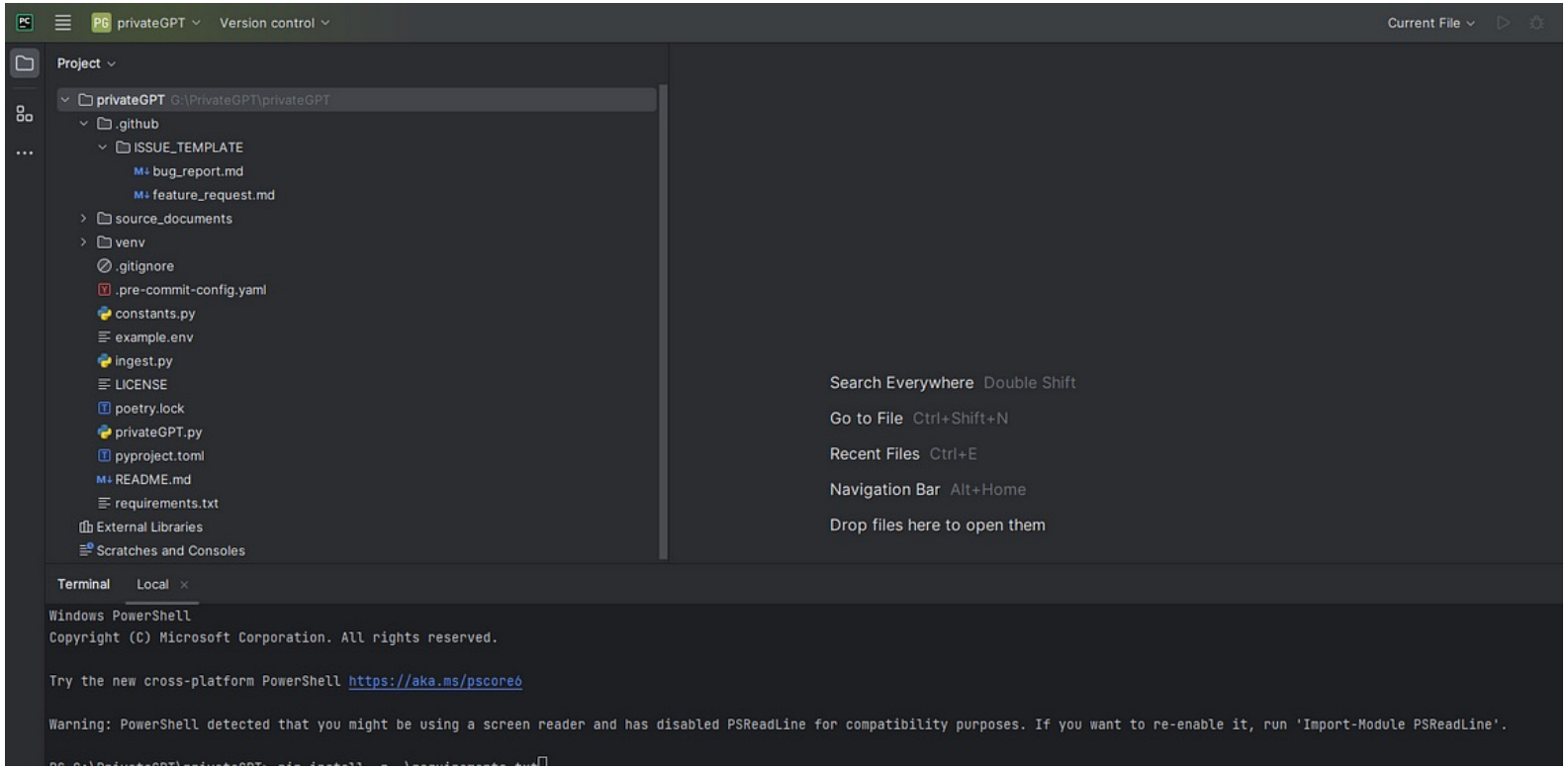
Download the GPT Source with - git clone https://github.com/imartinez/privateGPT.git



Private GPT on Your Windows PC

Import the PrivateGPT into an IDE

Install Required Python Packages with `pip install -r .\requirements.txt`

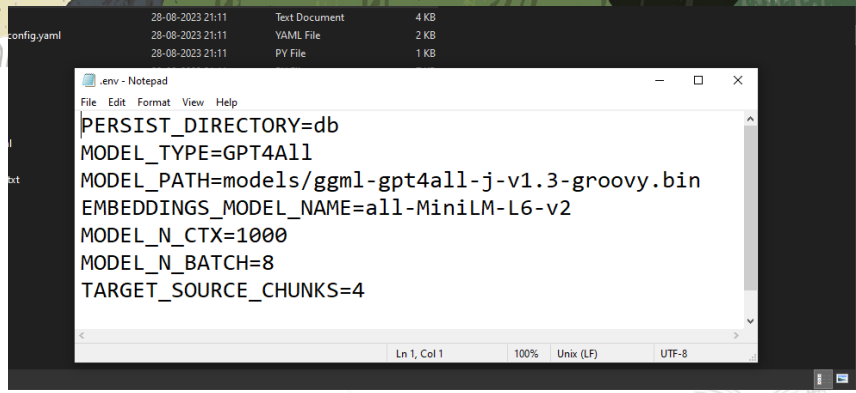


Private GPT on Your Windows PC

Download a Large Language Model

- The Private GPT code is designed to work with models compatible with GPT4All-J or LlamaCpp.
- Download whichever model you prefer based on size.
- Once downloaded, create a folder called `models` inside the `privateGPT` folder, and move the `.bin` file into it.

Private GPT on Your Windows PC



Configure Environment Variables

Rename `example.env` to `.env` (remove `example`) and open it in a text editor.

`MODEL_PATH`: Set this to the path to your language model file

`MODEL_TYPE`: supports LlamaCpp or GPT4All
`PERSIST_DIRECTORY`: is the folder you want your vectorstore in
`MODEL_PATH`: Path to your GPT4All or LlamaCpp supported LLM
`MODEL_N_CTX`: Maximum token limit for the LLM model
`MODEL_N_BATCH`: Number of tokens in the prompt that are fed into the model at a time.

Optimal value differs a lot depending on the model (8 works well for GPT4All, and 1024 is better for LlamaCpp)

`EMBEDDINGS_MODEL_NAME`: SentenceTransformers embeddings model name

`TARGET_SOURCE_CHUNKS`: The amount of chunks (sources) that will be used to answer a question

Private GPT on Your Windows PC

Ingest Documents

- This preprocesses your files so Private GPT can search and query them. In the PyCharm terminal, run:

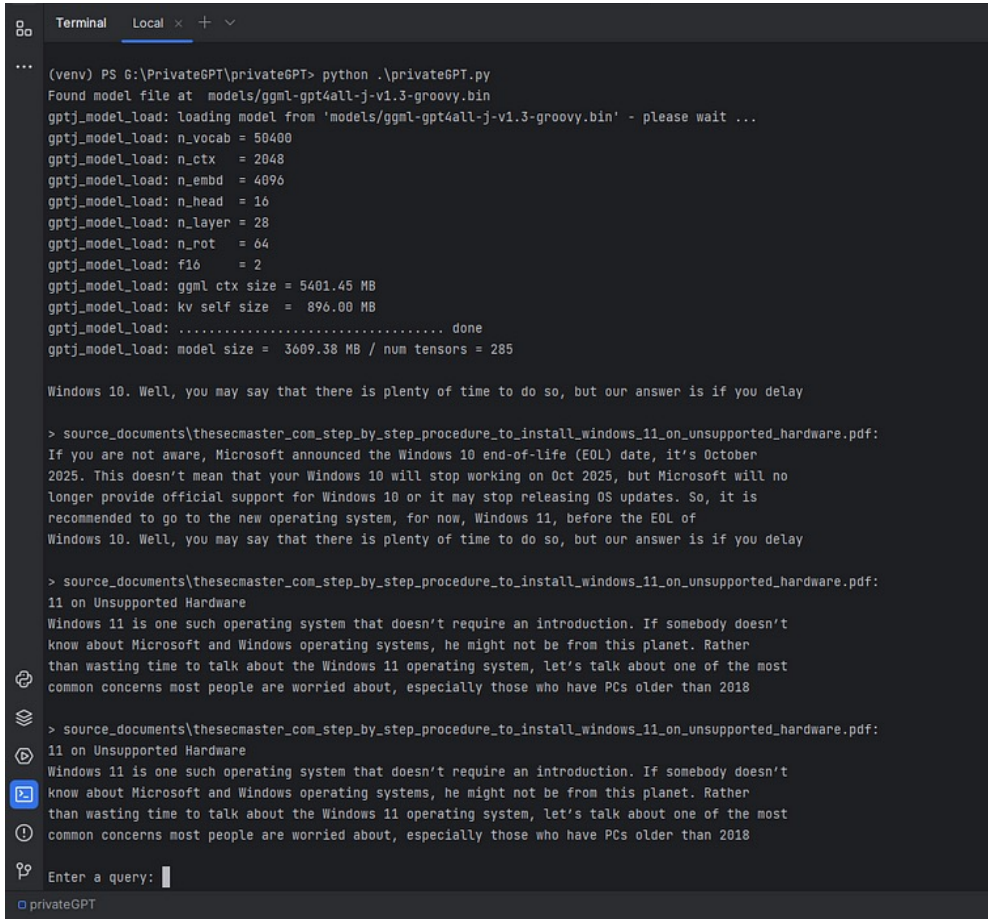
```
python .\ingest.py
```

- This will look for files in the source_documents folder, process them, and add them to the database.
- You can add .pdf, .docx, .txt, and other files in this folder ‘source_documents.’ The initial process may take some time, depending on how large your files are.
- Once it finishes, your documents are ready to query!

Private GPT on Your Windows PC

Query Your Documents - `python .\privateGPT.py`

- This will prompt you to enter a query. Type your question and hit enter.
- The model will think for 20–30 seconds and then return an answer by searching your ingested documents, along with context snippets.
- You can keep entering new questions or type `exit` to quit.



Thank You