PrivateGPT

Querying own files with GPT locally using any GPT model (Retrieval-augmented generation (RAG))
Supports text formats only (pdf, doc, txt, etc), limited support for tables, but not csv data or large tables.

https://github.com/zylon-ai/private-gpt

CPU only docker

Docker version available.

```
#Install docker for user
curl -fSSL get.docker.com | sh
sudo apt-get install -y uidmap
dockerd-rootless-setuptool.sh install

#prepare, using Mistral-7B-OpenOrca Model, remove the LLAMACPP entries to use default
mkdir ~/aimodels
chmod 777 ~/aimodels/
cat << EOF >> docker-privategpt.yml
services:
  # https://hub.docker.com/r/3x3cut0r/privategpt
  privategpt:
    image: 3x3cut0r/privategpt:latest
    container_name: privategpt
    environment:
      LLAMACPP_LLM_HF_REPO_ID: "TheBloke/Mistral-7B-OpenOrca-GGUF"
      LLAMACPP_LLM_HF_MODEL_FILE: "mistral-7b-openorca.Q5_K_M.gguf"
      LLAMACPP_EMBEDDING_HF_MODEL_NAME: "BAAI/bge-large-en-v1.5"
      EMBEDDING_INGEST_MODE: "parallel"
      EMBEDDING_COUNT_WORKERS: "4"
    volumes:
      - /home/$USER/aimodels:/home/worker/app/models
    ports:
      - 8080:8080/tcp
EOF

docker compose -f docker-privategpt.yml up -d
docker logs --follow privategpt
```

NVIDIA docker

PrivateGPT Docker with NVIDIA GPU support - needs some adjustments to settings and docker

compose files to use model variables properly:

```
git clone https://github.com/zylon-ai/private-gpt
cd private-gpt/

mkdir -p /opt/privategpt-storage/{local_data,models}
chown 1000:1000 /opt/privategpt-storage -R

# from PR https://github.com/zylon-ai/private-gpt/pull/1655/files
# and https://github.com/zylon-ai/private-gpt/issues/1405

cat << EOD >> Dockerfile.local.gpu
FROM nvidia/cuda:12.2.2-devel-ubuntu22.04 as base

# For tzdata
ENV DEBIAN_FRONTEND="noninteractive" TZ="Etc/UTC"

# Install Python 3.11 and set it as default
RUN apt-get update && \
    apt-get install -y software-properties-common && \
    add-apt-repository ppa:deadsnakes/ppa && \
    apt-get update && \
    apt-get install -y python3.11 python3.11-venv python3-pip && \
    ln -sf /usr/bin/python3.11 /usr/bin/python3 && \
    python3 --version

# Install poetry
RUN pip install pipx
RUN python3 -m pipx ensurepath
RUN pipx install poetry
ENV PATH="/root/.local/bin:$PATH"
ENV PATH=".venv/bin/:$PATH"

# Dependencies to build llama-cpp
RUN apt update && apt install -y \
  libopenblas-dev\
  ninja-build\
  build-essential\
  pkg-config\
  wget\
  gcc

# https://python-poetry.org/docs/configuration/#virtualenvsin-project
ENV POETRY_VIRTUALENVS_IN_PROJECT=true

##########################
FROM base as dependencies
##########################

WORKDIR /home/worker/app
COPY pyproject.toml poetry.lock ./
```

```
RUN poetry config installer.max-workers 10
RUN poetry install --extras "ui embeddings-huggingface llms-llama-cpp
vector-stores-qdrant"

# Enable GPU support
ENV LLAMA_CUBLAS=1
RUN CMAKE_ARGS='-DLLAMA_CUBLAS=on' FORCE_CMAKE=1 poetry run pip install --
upgrade --force-reinstall --no-cache-dir llama-cpp-python

################
FROM base as app
################

ENV PYTHONUNBUFFERED=1
ENV PORT=8080
EXPOSE 8080

# Prepare a non-root user
RUN adduser worker
WORKDIR /home/worker/app

RUN mkdir -p local_data; chown -R worker local_data
RUN mkdir -p models; chown -R worker models
COPY --chown=worker --from=dependencies /home/worker/app/.venv/ .venv
COPY --chown=worker private_gpt/ private_gpt
COPY --chown=worker fern/ fern
COPY --chown=worker *.yaml *.md ./
COPY --chown=worker scripts/ scripts
COPY --chown=worker pyproject.toml poetry.lock ./

# Copy the entry point script into the container and make it executable
COPY --chown=worker entrypoint.sh /entrypoint.sh
RUN chmod +x /entrypoint.sh

ENV PYTHONPATH="$PYTHONPATH:/private_gpt/"

#USER worker

#ENTRYPOINT ["/entrypoint.sh", "python", "-m", "private_gpt"]
ENTRYPOINT /entrypoint.sh python -m private_gpt
EOD

cat << EOD >> entrypoint.sh
#!/bin/sh

## Download the embedding and model files
echo "Running setup script"
poetry run python scripts/setup

## Execute the main container command
```

```
exec "$@"
EOD


cat << EOD >> settings-docker.yaml
server:
  env_name: ${APP_ENV:prod}
  port: ${PORT:8080}

llm:
  mode: ${PGPT_LLM_MODE:mock}

embedding:
  mode: ${PGPT_EMBEDDING_MODE:sagemaker}

llamacpp:
  prompt_style: ${PGPT_PROMPT_STYLE:mistral}
  llm_hf_repo_id: ${PGPT_HF_REPO_ID:TheBloke/Mistral-7B-Instruct-v0.2-GGUF}
  llm_hf_model_file: ${PGPT_HF_MODEL_FILE:mistral-7b-instruct-
v0.2.Q4_K_M.gguf}

huggingface:
  embedding_hf_model_name: ${PGPT_EMBEDDING_HF_MODEL_NAME:BAAI/bge-small-en-
v1.5}

sagemaker:
  llm_endpoint_name: ${PGPT_SAGEMAKER_LLM_ENDPOINT_NAME:}
  embedding_endpoint_name: ${PGPT_SAGEMAKER_EMBEDDING_ENDPOINT_NAME:}

ollama:
  llm_model: ${PGPT_OLLAMA_LLM_MODEL:mistral}
  embedding_model: ${PGPT_OLLAMA_EMBEDDING_MODEL:nomic-embed-text}
  api_base: ${PGPT_OLLAMA_API_BASE:http://ollama:11434}
  embedding_api_base: ${PGPT_OLLAMA_EMBEDDING_API_BASE:http://ollama:11434}
  tfs_z: ${PGPT_OLLAMA_TFS_Z:1.0}
  top_k: ${PGPT_OLLAMA_TOP_K:40}
  top_p: ${PGPT_OLLAMA_TOP_P:0.9}
  repeat_last_n: ${PGPT_OLLAMA_REPEAT_LAST_N:64}
  repeat_penalty: ${PGPT_OLLAMA_REPEAT_PENALTY:1.2}
  request_timeout: ${PGPT_OLLAMA_REQUEST_TIMEOUT:600.0}

ui:
  enabled: true
  path: /
EOD

cat << EOD >> docker-compose-gpu-wulf.yaml
services:
  private-gpt-gpu:
    container_name: private-gpt-gpu
    restart: unless-stopped
```

```
    build:
      dockerfile: Dockerfile.local.gpu
    volumes:
      - /opt/privategpt-storage/local_data/:/home/worker/app/local_data
      - /opt/privategpt-storage/models/:/home/worker/app/models
    ports:
      - 8001:8080
    environment:
      PORT: 8080
      PGPT_PROFILES: docker
      PGPT_LLM_MODE: llamacpp
      PGPT_EMBEDDING_MODE: huggingface

      #Microsoft Phi-3 Mini 4k
      PGPT_HF_REPO_ID: "microsoft/Phi-3-mini-4k-instruct-gguf"
      #PGPT_HF_MODEL_FILE: "Phi-3-mini-4k-instruct-fp16.gguf"
      PGPT_HF_MODEL_FILE: "Phi-3-mini-4k-instruct-q4.gguf"
      PGPT_PROMPT_STYLE: "chatml"

      #Meta Llama 3
      #PGPT_HF_REPO_ID: "QuantFactory/Meta-Llama-3-8B-Instruct-GGUF"
      #PGPT_HF_MODEL_FILE: "Meta-Llama-3-8B-Instruct.Q5_K_M.gguf"
      #PGPT_PROMPT_STYLE: "llama3"

      #OpenOrca Mistral 7B
      #PGPT_HF_REPO_ID: "TheBloke/Mistral-7B-OpenOrca-GGUF"
      #PGPT_HF_MODEL_FILE: "mistral-7b-openorca.Q5_K_M.gguf"
      #PGPT_PROMPT_STYLE: "mistral"

      PGPT_EMBEDDING_HF_MODEL_NAME: "BAAI/bge-small-en-v1.5"
      #PGPT_EMBEDDING_HF_MODEL_NAME: "BAAI/bge-large-en-v1.5"

      TOKENIZERS_PARALLELISM: True
      #PGPT_NGL: 20

      PGPT_MAX_NEW_TOKENS: 512
      PGPT_CONTEXT_WINDOW: 3900
      PGPT_TEMPERATURE: 0.1

      EMBEDDING_INGEST_MODE: "simple"
      EMBEDDING_COUNT_WORKERS: "2"
      HUGGINGFACE_TOKEN: "token to download gated models from huggingface"
      PYTORCH_CUDA_ALLOC_CONF: "max_split_size_mb:256"
    deploy:
      resources:
        reservations:
          devices:
            - driver: nvidia
              count: 1
              capabilities: [gpu]
EOD
```

```
docker build -f Dockerfile.local.gpu .
docker compose -f docker-compose-gpu-wulf.yaml up -d
docker logs -n 20 --follow private-gpt-gpu
nvtop
```

NGL settings patch

To add the amount of layers loaded in the GPU for llamacpp, apply this NGL option patch, then add "PGPT_NGL: 20" to the docker compose environment section with 20 being the amount of layers or -1 for all.

```
cat << EOD >> ngl-settings-option.patch
diff --git a/private_gpt/components/llm/llm_component.py
b/private_gpt/components/llm/llm_component.py
index baffa4e..e8bddd2 100644
--- a/private_gpt/components/llm/llm_component.py
+++ b/private_gpt/components/llm/llm_component.py
@@ -57,7 +57,7 @@ class LLMComponent:
                     "top_k": settings.llamacpp.top_k,  # ollama and llama-
cpp
                     "top_p": settings.llamacpp.top_p,  # ollama and llama-
cpp
                     "repeat_penalty": settings.llamacpp.repeat_penalty,  #
ollama llama-cpp
-                    "n_gpu_layers": -1,
+                    "n_gpu_layers": settings.llamacpp.ngl,
                     "offload_kqv": True,
                 }
                 self.llm = LlamaCPP(
diff --git a/private_gpt/settings/settings.py
b/private_gpt/settings/settings.py
index 051cfca..701a1a9 100644
--- a/private_gpt/settings/settings.py
+++ b/private_gpt/settings/settings.py
@@ -145,6 +145,11 @@ class LlamaCPPSettings(BaseModel):
         1.1,
         description="Sets how strongly to penalize repetitions. A higher
value (e.g., 1.5) will penalize repetitions more strongly, while a lower
value (e.g., 0.9) will be more lenient. (Default: 1.1)",
     )
+    ngl: int = Field(
+        -1,
+        description="Number of layers loaded in GPU (Default: -1)",
+    )
+


 class HuggingFaceSettings(BaseModel):
```

```
diff --git a/settings.yaml b/settings.yaml
index e881a55..c4c86cb 100644
--- a/settings.yaml
+++ b/settings.yaml
@@ -60,6 +60,7 @@ llamacpp:
   top_k: 40             # Reduces the probability of generating nonsense. A higher value (e.g. 100) will give more diverse answers, while a lower value (e.g. 10) will be more conservative. (Default: 40)
   top_p: 1.0            # Works together with top-k. A higher value (e.g., 0.95) will lead to more diverse text, while a lower value (e.g., 0.5) will generate more focused and conservative text. (Default: 0.9)
   repeat_penalty: 1.1   # Sets how strongly to penalize repetitions. A higher value (e.g., 1.5) will penalize repetitions more strongly, while a lower value (e.g., 0.9) will be more lenient. (Default: 1.1)
+  ngl: ${PGPT_NGL:-1}   # Sets number of layers offloaded to gpu

 embedding:
   # Should be matching the value above in most cases
EOD

git apply ngl-settings-option.patch
```

Max New Tokens / Context Size / Temperature settings patch

To be able to set Max New Tokens, Context Size and Temperature in the docker compose file as variables, the settings.yaml file needs to be adjusted.

docker compose file additions:

```
    environment:
      PGPT_MAX_NEW_TOKENS: 512
      PGPT_CONTEXT_WINDOW: 3900
      PGPT_TEMPERATURE: 0.1
```

```
cat << EOD >> token-ctx-temp-settings-option.patch
diff --git a/settings.yaml b/settings.yaml
index e881a55..8666b86 100644
--- a/settings.yaml
+++ b/settings.yaml
@@ -37,10 +37,10 @@ ui:
 llm:
   mode: llamacpp
   # Should be matching the selected model
-  max_new_tokens: 512
-  context_window: 3900
+  max_new_tokens: ${PGPT_MAX_NEW_TOKENS:512}
+  context_window: ${PGPT_CONTEXT_WINDOW:3900}
   tokenizer: mistralai/Mistral-7B-Instruct-v0.2
```

```
-  temperature: 0.1      # The temperature of the model. Increasing the
temperature will make the model answer more creatively. A value of 0.1 would
be more factual. (Default: 0.1)
+  temperature: ${PGPT_TEMPERATURE:0.1}      # The temperature of the model.
Increasing the temperature will make the model answer more creatively. A
value of 0.1 would be more factual. (Default: 0.1)

 rag:
   similarity_top_k: 2
EOD

git apply token-ctx-temp-settings-option.patch
```

CSS Customisation

To adjust the main input box and fix mobile/low height browser window issue of the input box wrapping to the right, some css trickery is required. The last three css lines are added to privategpt/ui/ui.py for this:

privategpt/ui/ui.py

```
def _build_ui_blocks(self) -> gr.Blocks:
        logger.debug("Creating the UI blocks")
        with gr.Blocks(
            title=UI_TAB_TITLE,
            theme=gr.themes.Soft(primary_hue=slate),
            css=".logo { "
            "display:flex;"
            "background-color: #000;"
            "height: 90px;"
            "border-radius: 8px;"
            "align-content: center;"
            "justify-content: center;"
            "align-items: center;"
            "font-size: xxx-large;"
            "color: #fff;"
            "font-weight: bold;"
            "}"
            ".logo img { height: 100% }"
            ".contain { display: flex !important; flex-direction:
column !important; }"
            "#component-0, #component-3, #component-10, #component-8  {
height: 100% !important; }"
            "#chatbot { flex-grow: 1 !important; overflow: auto
!important;}"
            "#col { height: calc(100vh - 112px - 16px) !important; }"
            "#component-24 > label:nth-child(1) > textarea:nth-child(2)
{ min-height: 100px !important; }"
            "#component-24 { min-width: min(260px, 100%) !important;}"
```

```
            "#col { min-height:750px !important; }",
        ) as blocks:
            with gr.Row():
```