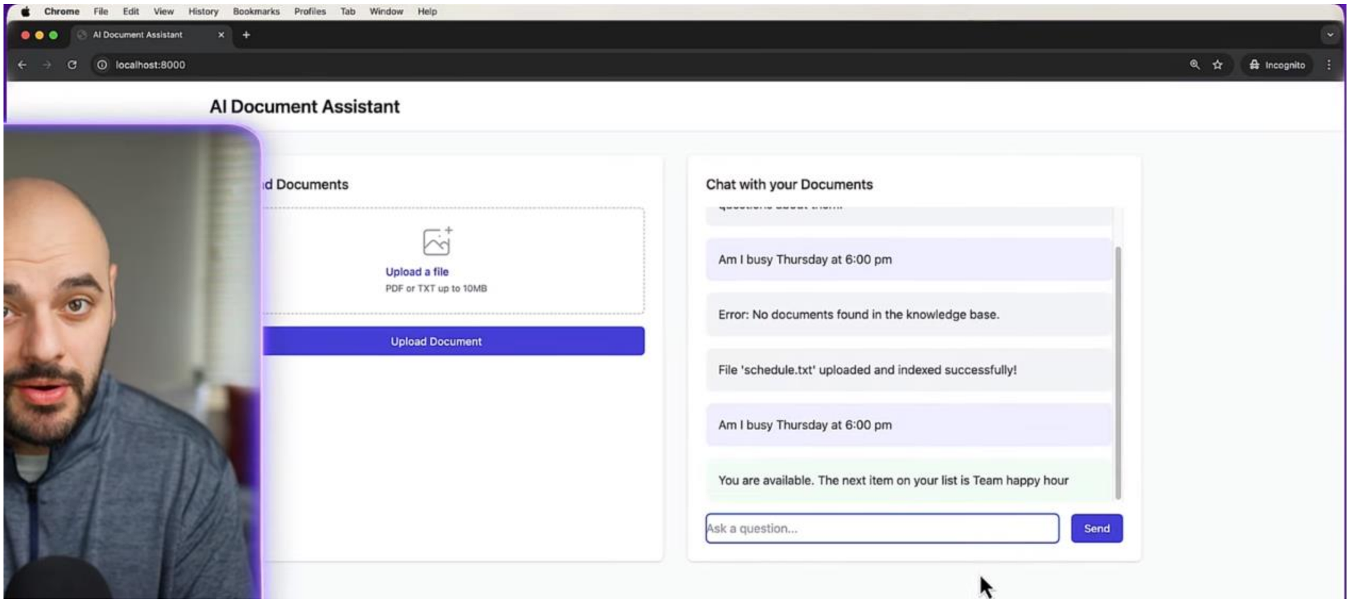
The BEST Way to Create an Offline RAG AI with FastAPI & PydanticAI

Eric Roby
23.6K subscribers

Join | Subscribe | 117 | Share | Ask | Download | ···

3,148 views Apr 17, 2025 #ai #fastapi #codingwithroby
AI Agents number one responsibility is to do AI automation. To make the AI automation personal, you want to implement RAG. So, in this PydanticAI and FastAPI tutorial we show you exactly how to automate your personal life with an AI RAG.

00:00 - Introduction
00:22 - What Are We Going To Be Doing
01:40 - Go Over Code
07:34 - Run Project with Swagger
08:35 - Run Project with UI



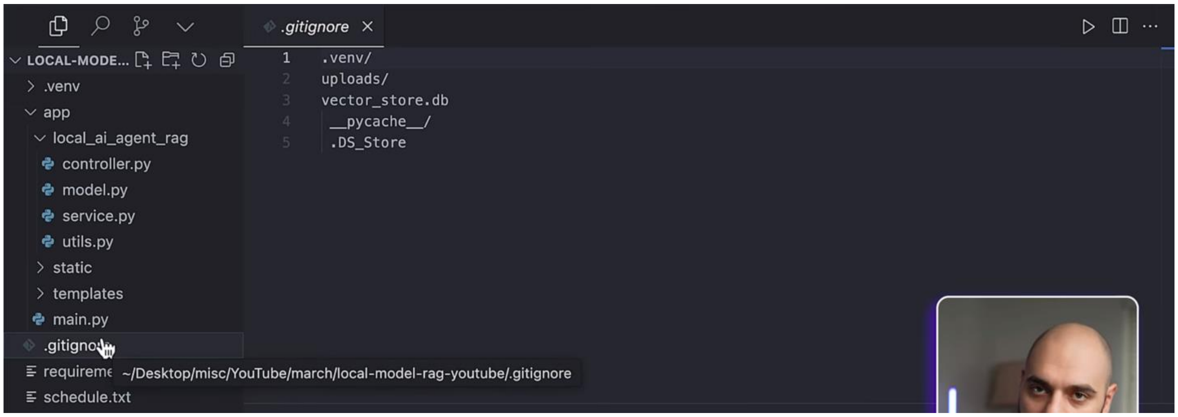
Am I busy Thursday at 6:00 pm

Error: No documents found in the knowledge base.

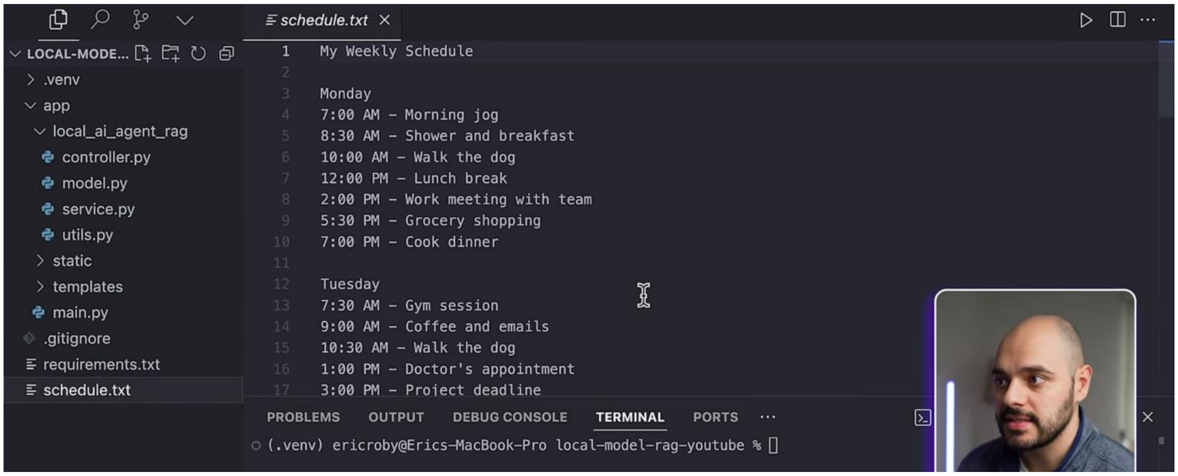
File 'schedule.txt' uploaded and indexed successfully!

Am I busy Thursday at 6:00 pm

You are available. The next item on your list is Team happy hour

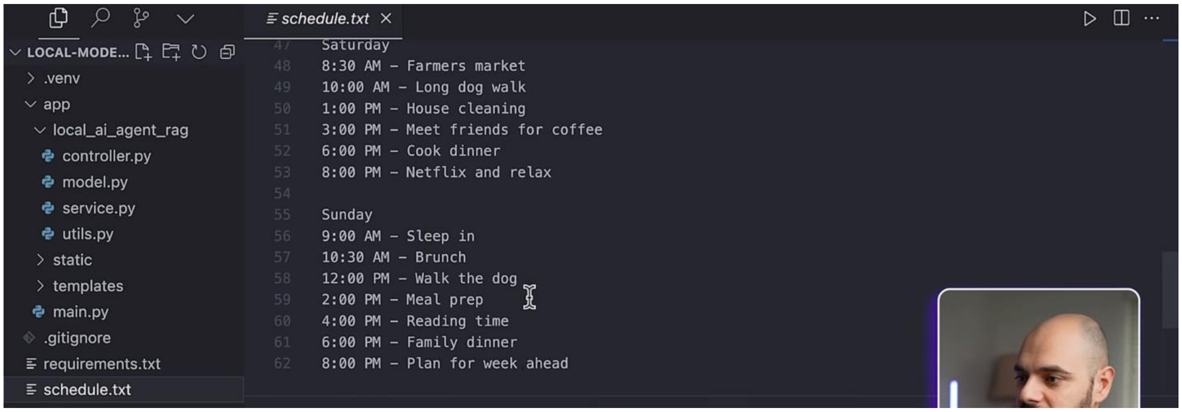


```
.venv/
uploads/
vector_store.db
 __pycache__/
 .DS_Store
```

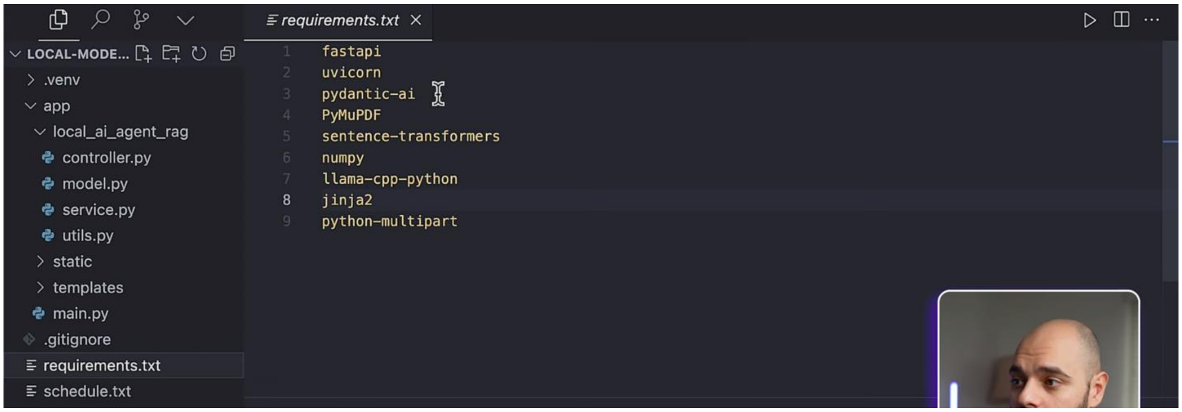


```
My Weekly Schedule

Monday
7:00 AM – Morning jog
8:30 AM – Shower and breakfast
10:00 AM – Walk the dog
12:00 PM – Lunch break
2:00 PM – Work meeting with team
5:30 PM – Grocery shopping
7:00 PM – Cook dinner

Tuesday
7:30 AM – Gym session
9:00 AM – Coffee and emails
10:30 AM – Walk the dog
1:00 PM – Doctor's appointment
3:00 PM – Project deadline
```

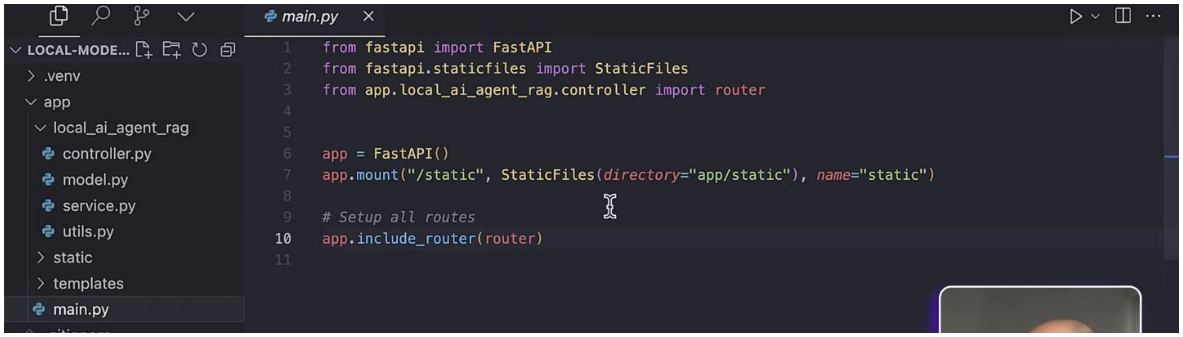
```
(.venv) ericroby@Erics-MacBook-Pro local-model-rag-youtube %
```



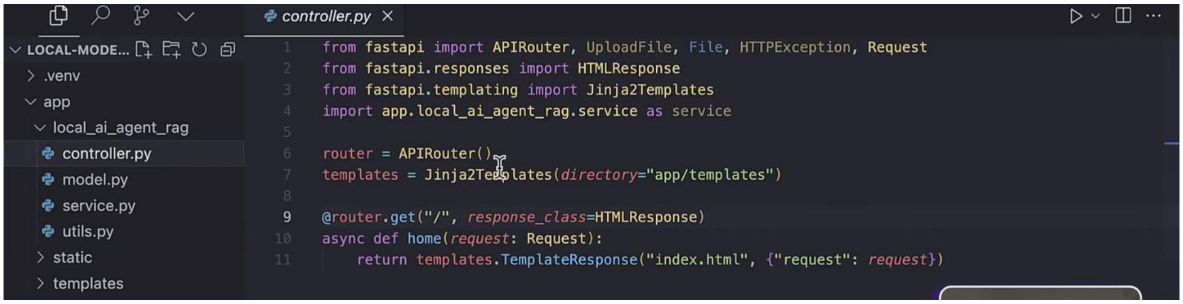
```
Saturday
8:30 AM – Farmers market
10:00 AM – Long dog walk
1:00 PM – House cleaning
3:00 PM – Meet friends for coffee
6:00 PM – Cook dinner
8:00 PM – Netflix and relax

Sunday
9:00 AM – Sleep in
10:30 AM – Brunch
12:00 PM – Walk the dog
2:00 PM – Meal prep
4:00 PM – Reading time
6:00 PM – Family dinner
8:00 PM – Plan for week ahead
```



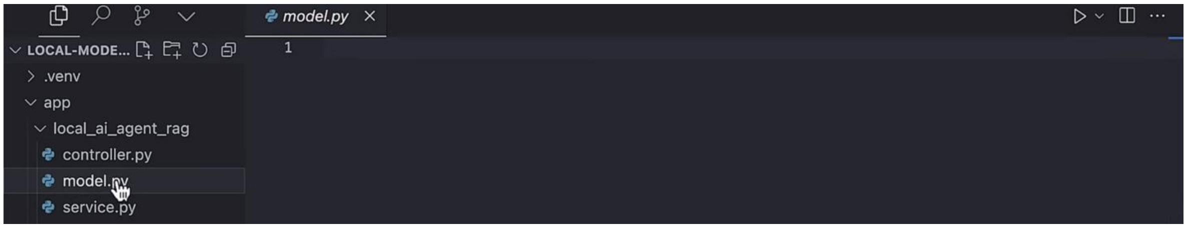
```
fastapi
uvicorn
pydantic-ai
PyMuPDF
sentence-transformers
numpy
llama-cpp-python
jinja2
python-multipart
```

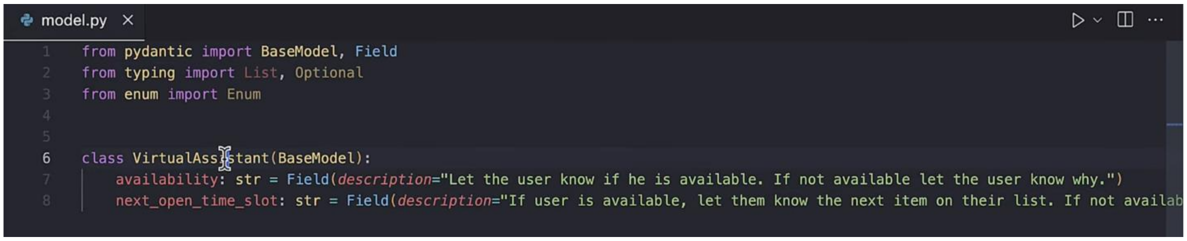


```python
from fastapi import FastAPI
from fastapi.staticfiles import StaticFiles
from app.local_ai_agent_rag.controller import router


app = FastAPI()
app.mount("/static", StaticFiles(directory="app/static"), name="static")

# Setup all routes
app.include_router(router)
```

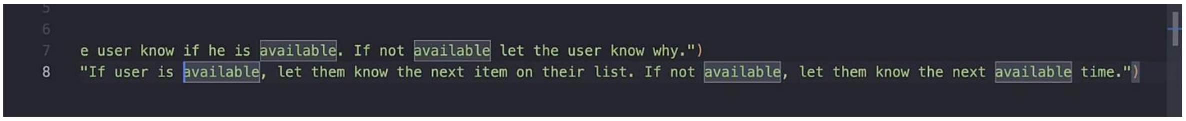


```python
from fastapi import APIRouter, UploadFile, File, HTTPException, Request
from fastapi.responses import HTMLResponse
from fastapi.templating import Jinja2Templates
import app.local_ai_agent_rag.service as service

router = APIRouter()
templates = Jinja2Templates(directory="app/templates")

@router.get("/", response_class=HTMLResponse)
async def home(request: Request):
    return templates.TemplateResponse("index.html", {"request": request})
```

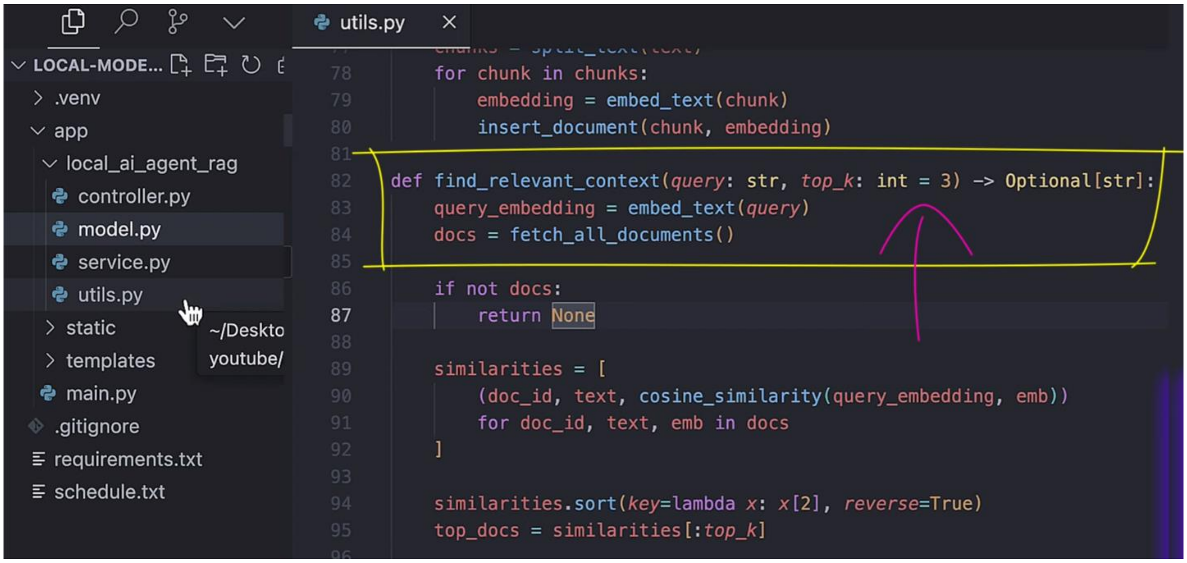




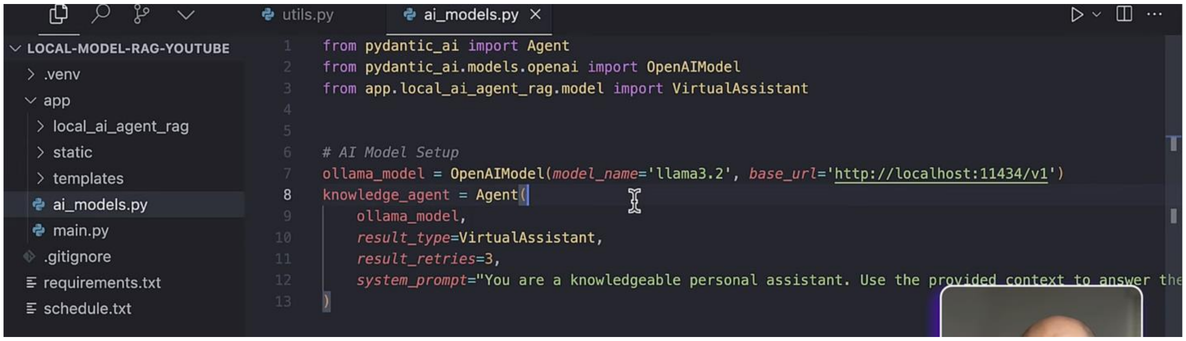
```python
from pydantic import BaseModel, Field
from typing import List, Optional
from enum import Enum


class VirtualAssistant(BaseModel):
    availability: str = Field(description="Let the user know if he is available. If not available let the user know why.")
    next_open_time_slot: str = Field(description="If user is available, let them know the next item on their list. If not availab
```



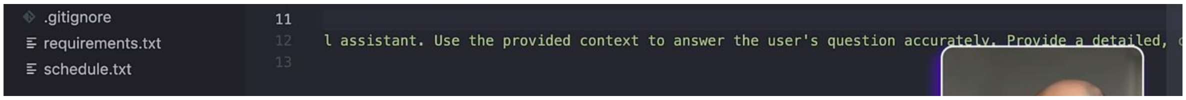
```python
e user know if he is available. If not available let the user know why.")
"If user is available, let them know the next item on their list. If not available, let them know the next available time.")
```



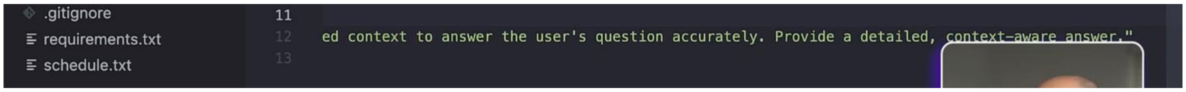
```python
    for chunk in chunks:
        embedding = embed_text(chunk)
        insert_document(chunk, embedding)

def find_relevant_context(query: str, top_k: int = 3) -> Optional[str]:
    query_embedding = embed_text(query)
    docs = fetch_all_documents()

    if not docs:
        return None

    similarities = [
        (doc_id, text, cosine_similarity(query_embedding, emb))
        for doc_id, text, emb in docs
    ]

    similarities.sort(key=lambda x: x[2], reverse=True)
    top_docs = similarities[:top_k]
```



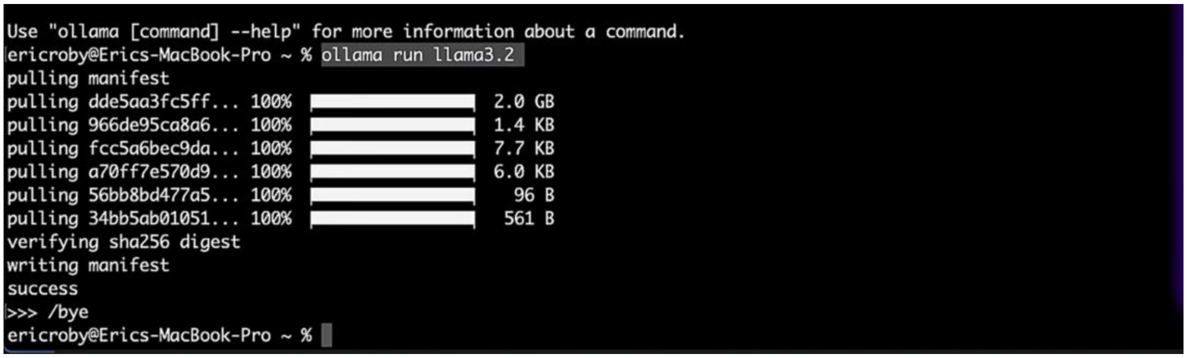
```python
from pydantic_ai import Agent
from pydantic_ai.models.openai import OpenAIModel
from app.local_ai_agent_rag.model import VirtualAssistant


# AI Model Setup
ollama_model = OpenAIModel(model_name='llama3.2', base_url='http://localhost:11434/v1')
knowledge_agent = Agent(
    ollama_model,
    result_type=VirtualAssistant,
    result_retries=3,
    system_prompt="You are a knowledgeable personal assistant. Use the provided context to answer the
)
```



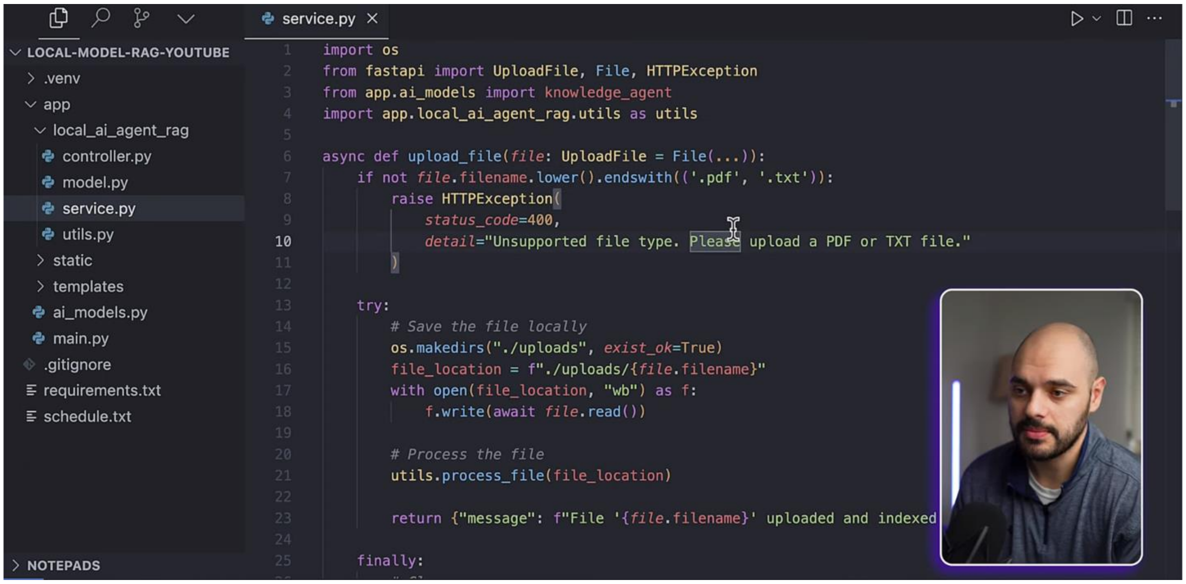
```python
l assistant. Use the provided context to answer the user's question accurately. Provide a detailed, c
```



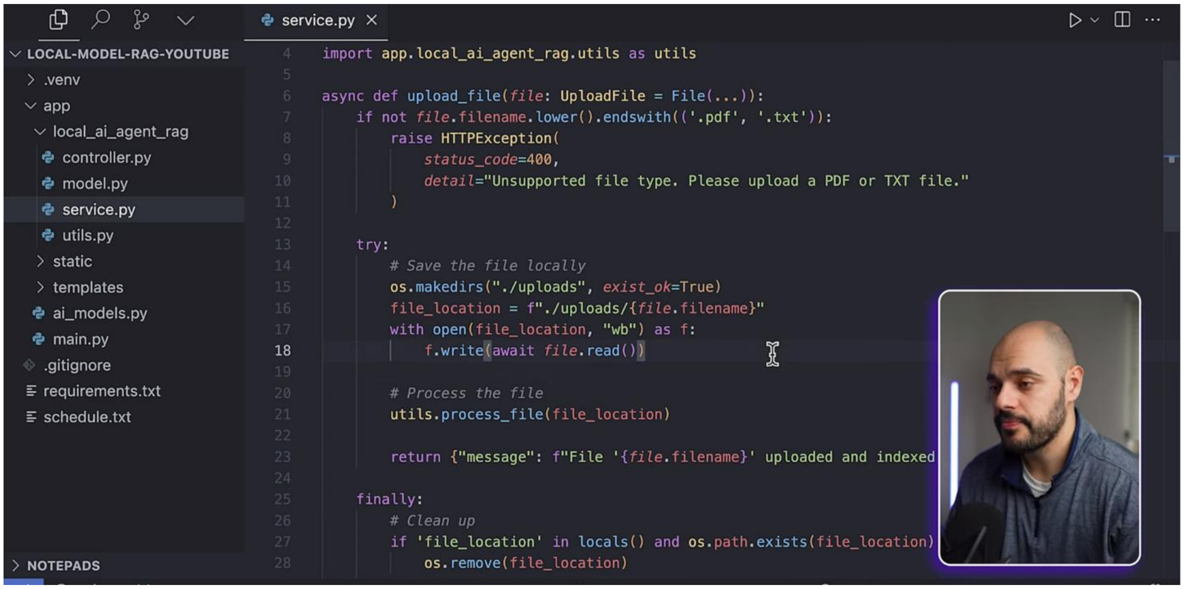
```python
ed context to answer the user's question accurately. Provide a detailed, context-aware answer."
```



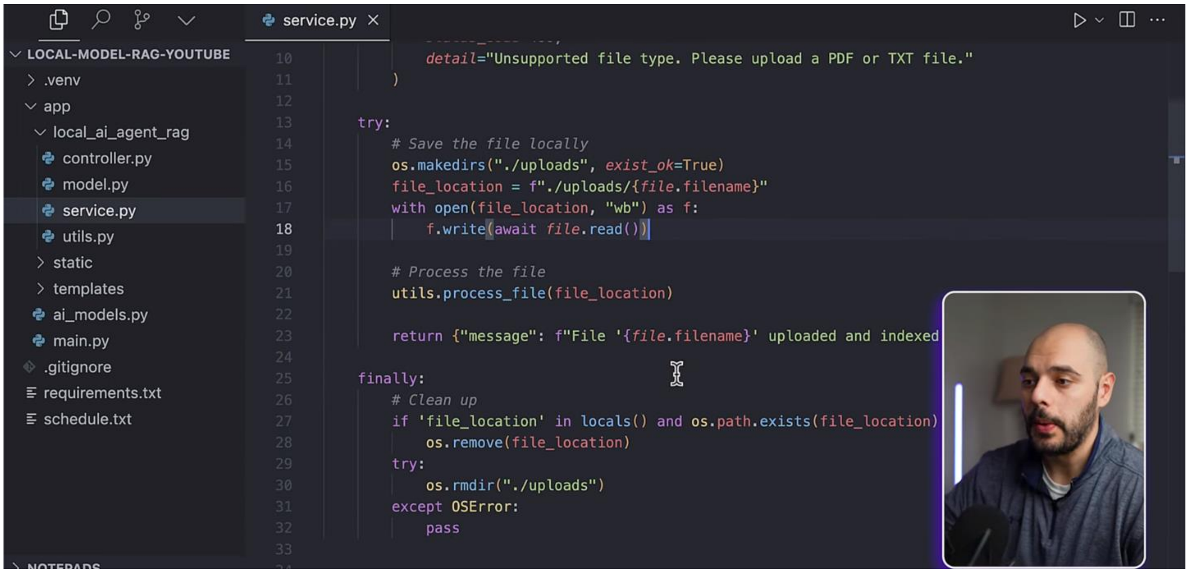
```
Use "ollama [command] --help" for more information about a command.
ericroby@Erics-MacBook-Pro ~ % ollama run llama3.2
pulling manifest
pulling dde5aa3fc5ff... 100% ▕████████████████▏ 2.0 GB
pulling 966de95ca8a6... 100% ▕████████████████▏ 1.4 KB
pulling fcc5a6bec9da... 100% ▕████████████████▏ 7.7 KB
pulling a70ff7e570d9... 100% ▕████████████████▏ 6.0 KB
pulling 56bb8bd477a5... 100% ▕████████████████▏   96 B
pulling 34bb5ab01051... 100% ▕████████████████▏  561 B
verifying sha256 digest
writing manifest
success
>>> /bye
ericroby@Erics-MacBook-Pro ~ %
```



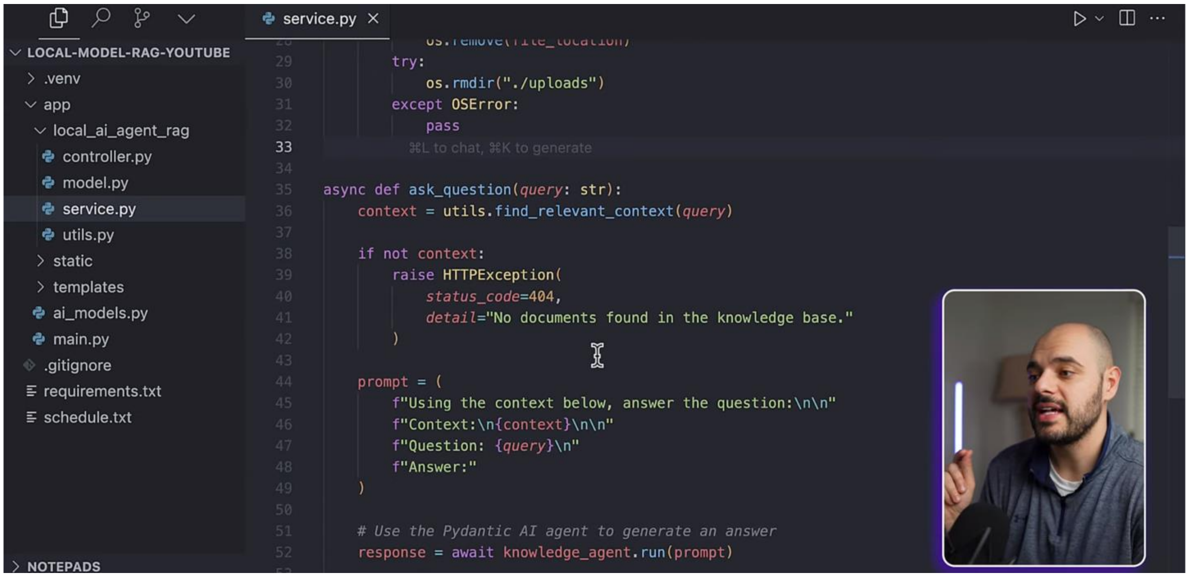
```python
import os
from fastapi import UploadFile, File, HTTPException
from app.ai_models import knowledge_agent
import app.local_ai_agent_rag.utils as utils

async def upload_file(file: UploadFile = File(...)):
    if not file.filename.lower().endswith(('.pdf', '.txt')):
        raise HTTPException(
            status_code=400,
            detail="Unsupported file type. Please upload a PDF or TXT file."
        )

    try:
        # Save the file locally
        os.makedirs("./uploads", exist_ok=True)
        file_location = f"./uploads/{file.filename}"
        with open(file_location, "wb") as f:
            f.write(await file.read())

        # Process the file
        utils.process_file(file_location)

        return {"message": f"File '{file.filename}' uploaded and indexed

    finally:
```



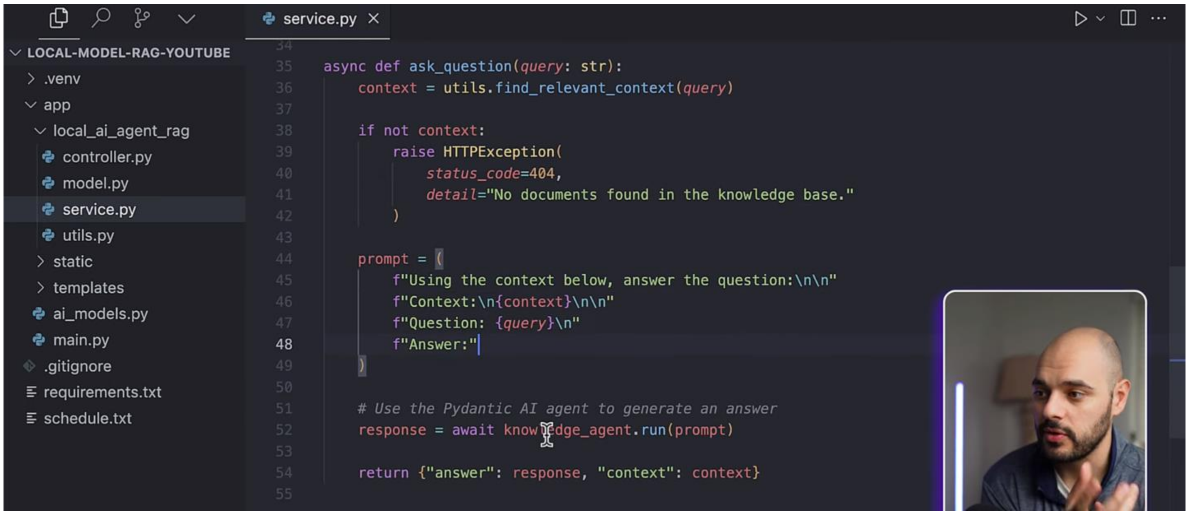
```python
import app.local_ai_agent_rag.utils as utils

async def upload_file(file: UploadFile = File(...)):
    if not file.filename.lower().endswith(('.pdf', '.txt')):
        raise HTTPException(
            status_code=400,
            detail="Unsupported file type. Please upload a PDF or TXT file."
        )

    try:
        # Save the file locally
        os.makedirs("./uploads", exist_ok=True)
        file_location = f"./uploads/{file.filename}"
        with open(file_location, "wb") as f:
            f.write(await file.read())

        # Process the file
        utils.process_file(file_location)

        return {"message": f"File '{file.filename}' uploaded and indexed

    finally:
        # Clean up
        if 'file_location' in locals() and os.path.exists(file_location)
            os.remove(file_location)
```



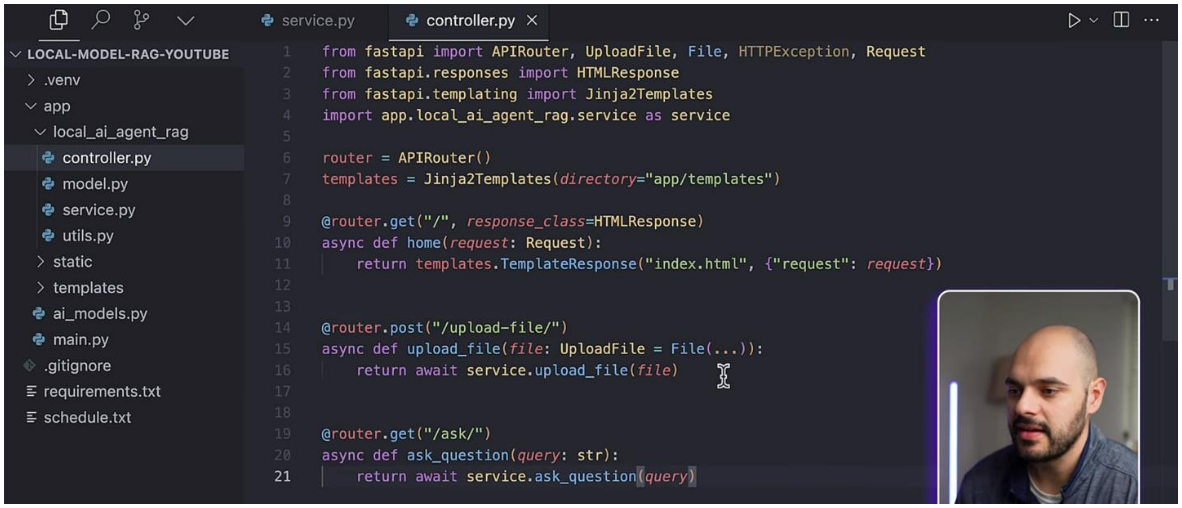
```python
            detail="Unsupported file type. Please upload a PDF or TXT file."
        )

    try:
        # Save the file locally
        os.makedirs("./uploads", exist_ok=True)
        file_location = f"./uploads/{file.filename}"
        with open(file_location, "wb") as f:
            f.write(await file.read())

        # Process the file
        utils.process_file(file_location)

        return {"message": f"File '{file.filename}' uploaded and indexed

    finally:
        # Clean up
        if 'file_location' in locals() and os.path.exists(file_location)
            os.remove(file_location)
        try:
            os.rmdir("./uploads")
        except OSError:
            pass
```



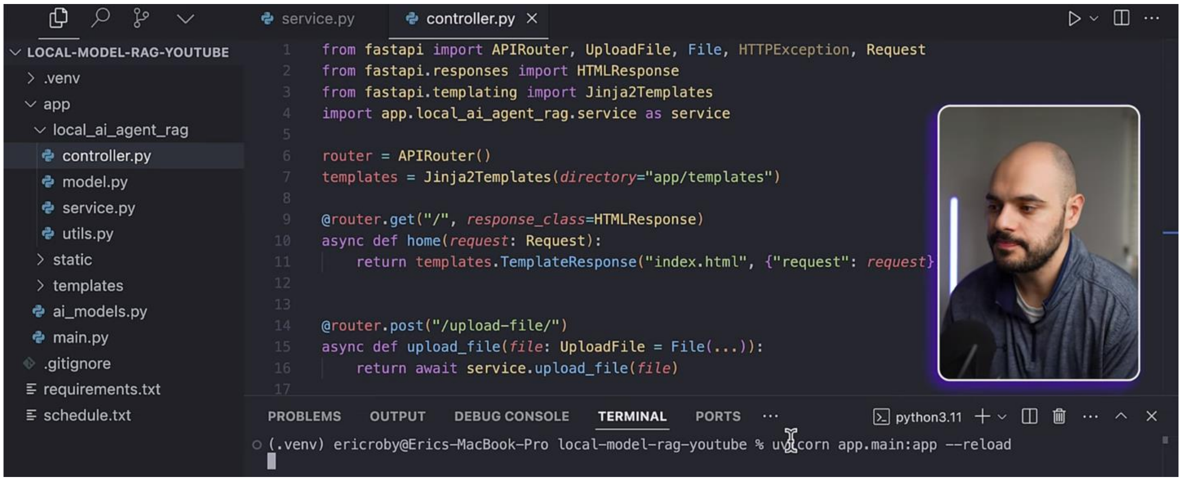
```python
        try:
            os.rmdir("./uploads")
        except OSError:
            pass


async def ask_question(query: str):
    context = utils.find_relevant_context(query)

    if not context:
        raise HTTPException(
            status_code=404,
            detail="No documents found in the knowledge base."
        )

    prompt = (
        f"Using the context below, answer the question:\n\n"
        f"Context:\n{context}\n\n"
        f"Question: {query}\n"
        f"Answer:"
    )

    # Use the Pydantic AI agent to generate an answer
    response = await knowledge_agent.run(prompt)
```



```python
async def ask_question(query: str):
    context = utils.find_relevant_context(query)

    if not context:
        raise HTTPException(
            status_code=404,
            detail="No documents found in the knowledge base."
        )

    prompt = (
        f"Using the context below, answer the question:\n\n"
        f"Context:\n{context}\n\n"
        f"Question: {query}\n"
        f"Answer:"
    )

    # Use the Pydantic AI agent to generate an answer
    response = await knowledge_agent.run(prompt)

    return {"answer": response, "context": context}
```

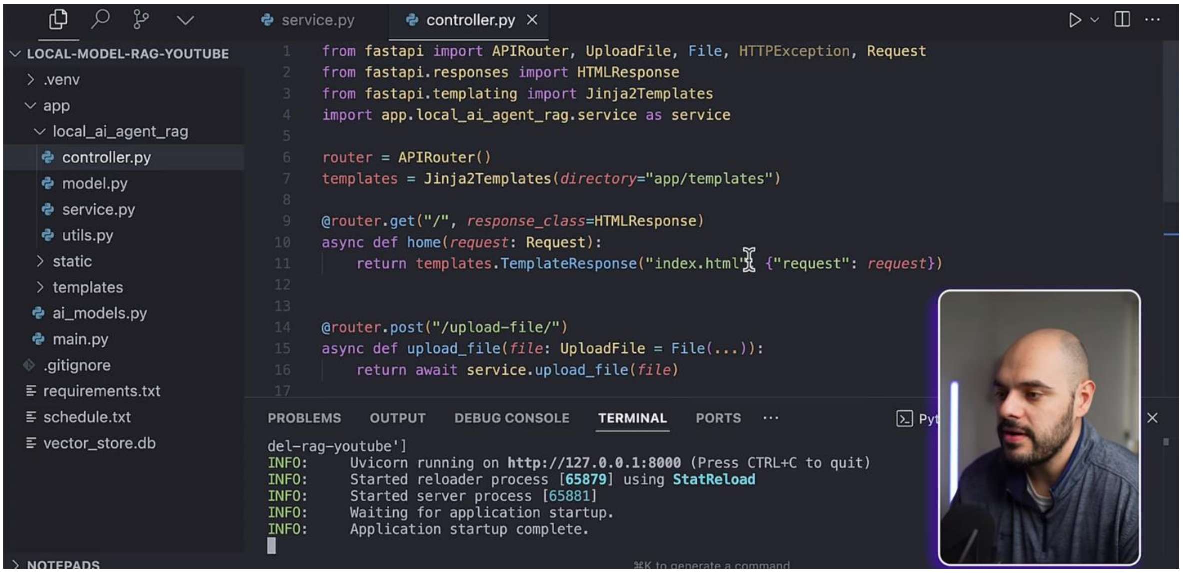


```python
from fastapi import APIRouter, UploadFile, File, HTTPException, Request
from fastapi.responses import HTMLResponse
from fastapi.templating import Jinja2Templates
import app.local_ai_agent_rag.service as service

router = APIRouter()
templates = Jinja2Templates(directory="app/templates")

@router.get("/", response_class=HTMLResponse)
async def home(request: Request):
    return templates.TemplateResponse("index.html", {"request": request})


@router.post("/upload-file/")
async def upload_file(file: UploadFile = File(...)):
    return await service.upload_file(file)


@router.get("/ask/")
async def ask_question(query: str):
    return await service.ask_question(query)
```



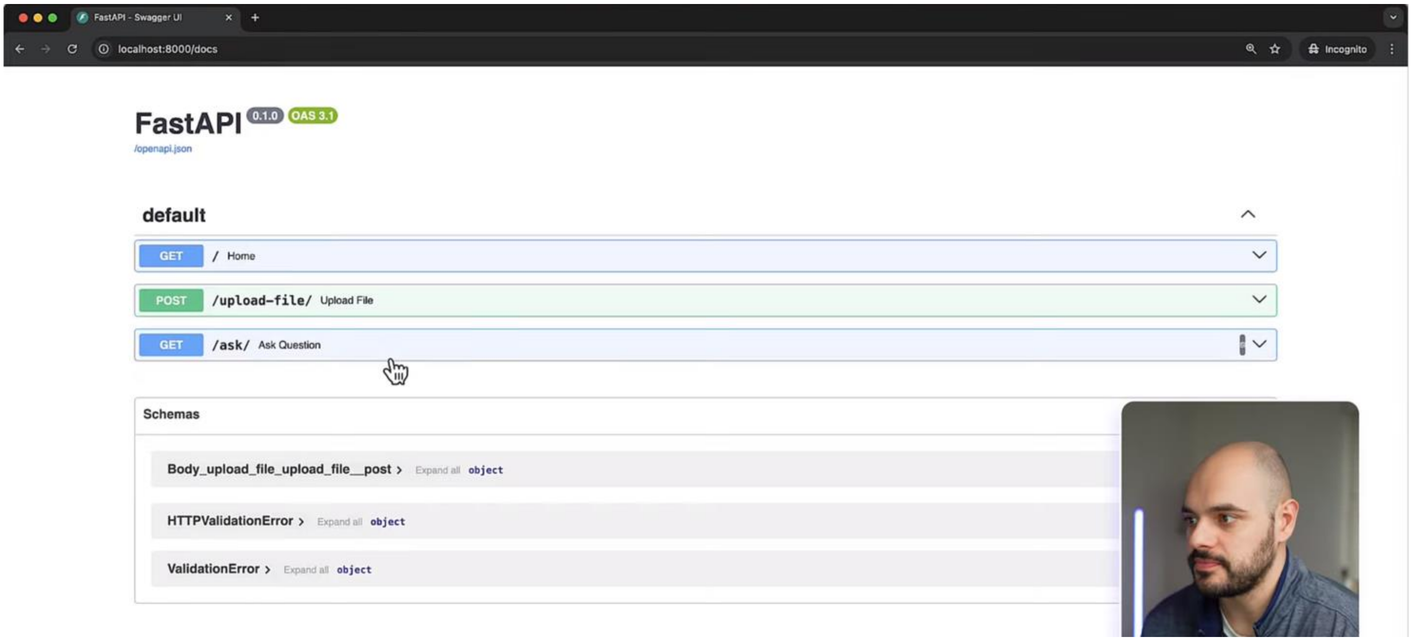
```python
from fastapi import APIRouter, UploadFile, File, HTTPException, Request
from fastapi.responses import HTMLResponse
from fastapi.templating import Jinja2Templates
import app.local_ai_agent_rag.service as service

router = APIRouter()
templates = Jinja2Templates(directory="app/templates")

@router.get("/", response_class=HTMLResponse)
async def home(request: Request):
    return templates.TemplateResponse("index.html", {"request": request}


@router.post("/upload-file/")
async def upload_file(file: UploadFile = File(...)):
    return await service.upload_file(file)
```

```
(.venv) ericroby@Erics-MacBook-Pro local-model-rag-youtube % uvicorn app.main:app --reload
```



```python
from fastapi import APIRouter, UploadFile, File, HTTPException, Request
from fastapi.responses import HTMLResponse
from fastapi.templating import Jinja2Templates
import app.local_ai_agent_rag.service as service

router = APIRouter()
templates = Jinja2Templates(directory="app/templates")

@router.get("/", response_class=HTMLResponse)
async def home(request: Request):
    return templates.TemplateResponse("index.html", {"request": request})


@router.post("/upload-file/")
async def upload_file(file: UploadFile = File(...)):
    return await service.upload_file(file)
```

```
del-rag-youtube']
INFO:     Uvicorn running on http://127.0.0.1:8000 (Press CTRL+C to quit)
INFO:     Started reloader process [65879] using StatReload
INFO:     Started server process [65881]
INFO:     Waiting for application startup.
INFO:     Application startup complete.
```



FastAPI - Swagger UI

localhost:8000/docs

Incognito

FastAPI 0.1.0 OAS 3.1

/openapi.json

default

GET / Home

POST /upload-file/ Upload File

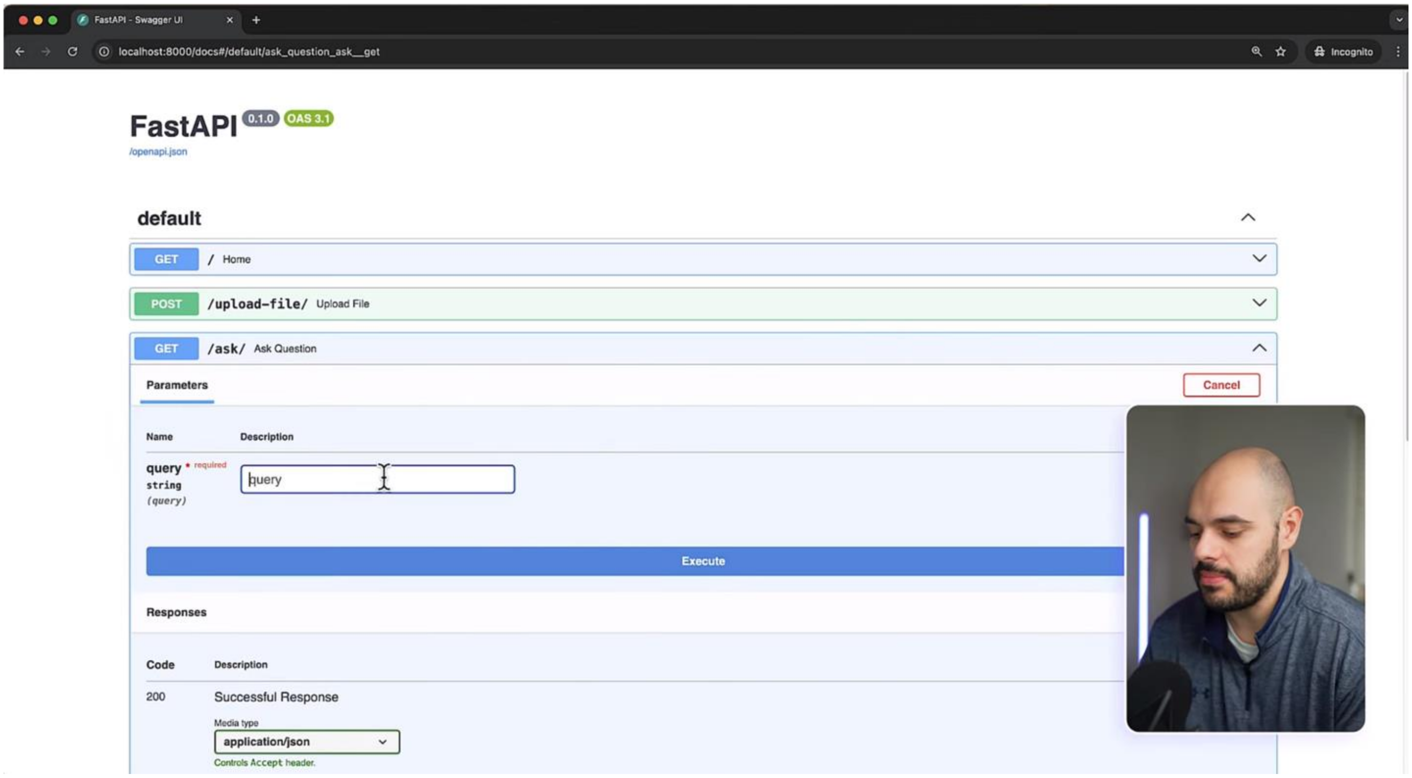
GET /ask/ Ask Question

Schemas

Body_upload_file_upload_file__post > Expand all object

HTTPValidationError > Expand all object

ValidationError > Expand all object



FastAPI - Swagger UI

localhost:8000/docs#/default/ask_question_ask__get

Incognito

FastAPI 0.1.0 OAS 3.1

/openapi.json

default

GET / Home

POST /upload-file/ Upload File

GET /ask/ Ask Question

Parameters

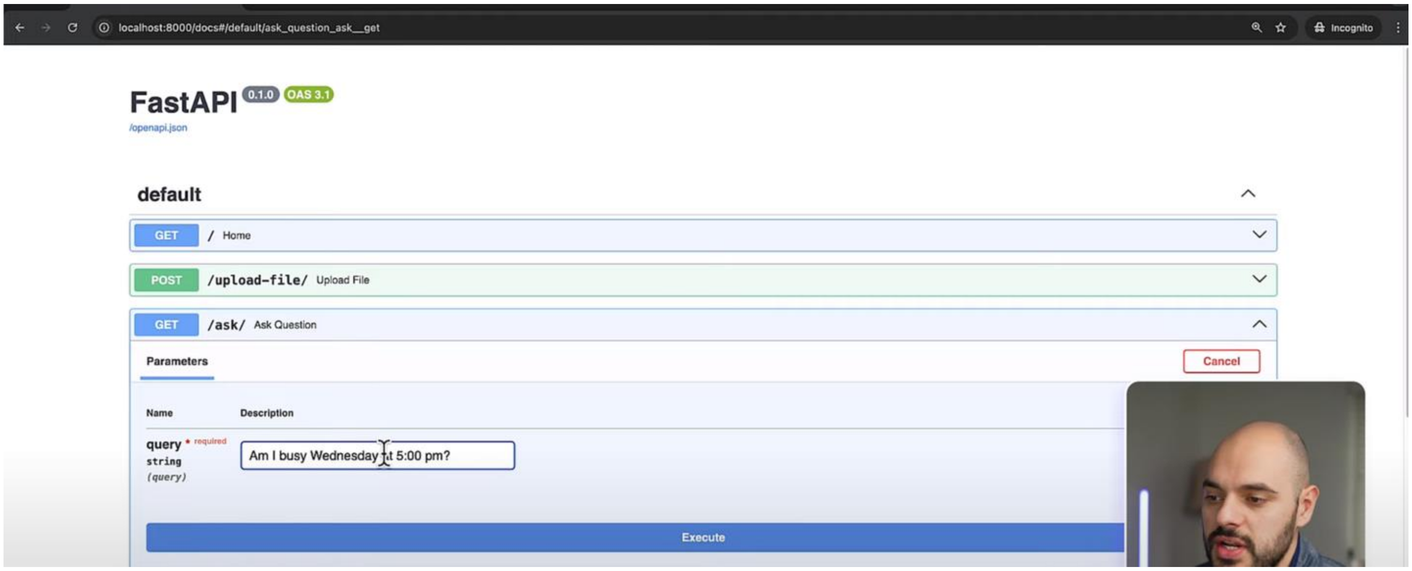
Cancel

Name	Description
query * required string *(query)*	query

Execute

Responses

Code	Description
200	Successful Response Media type application/json Controls Accept header.



FastAPI 0.1.0 OAS 3.1
/openapi.json

default

GET / Home

POST /upload-file/ Upload File

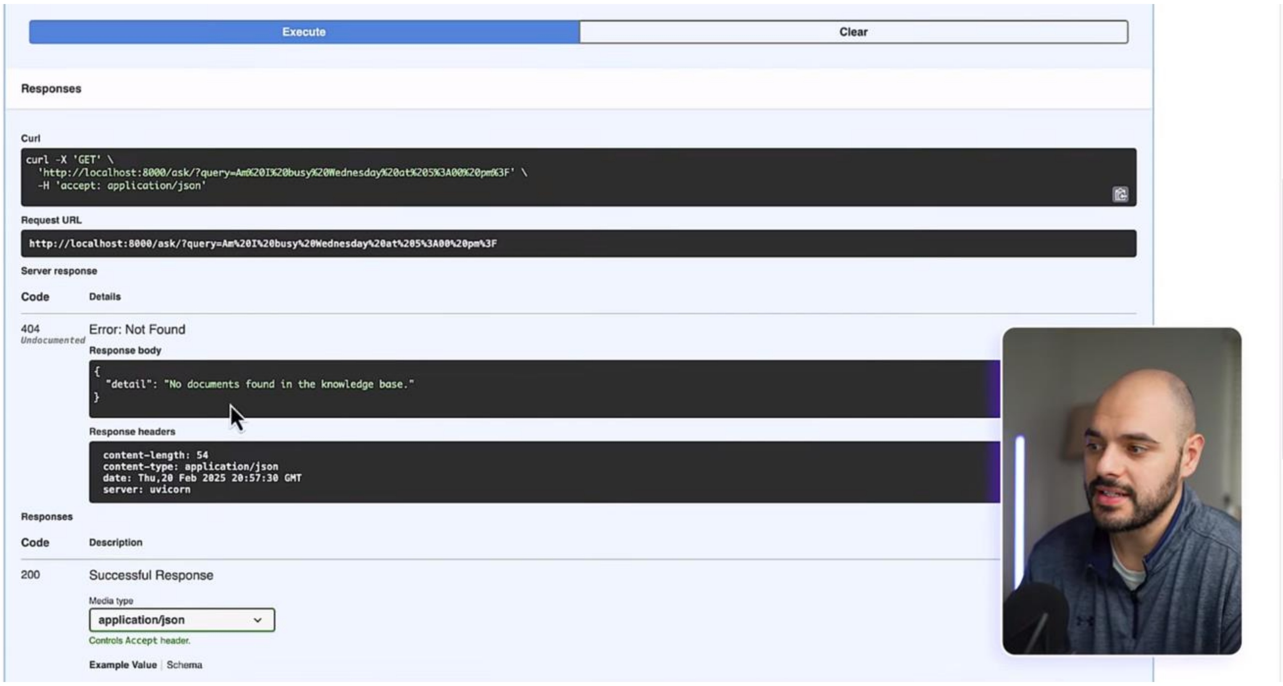
GET /ask/ Ask Question

Parameters

Cancel

Name	Description
query * required string *(query)*	Am I busy Wednesday at 5:00 pm?

Execute



Execute Clear

Responses

Curl

```
curl -X 'GET' \
  'http://localhost:8000/ask/?query=Am%20I%20busy%20Wednesday%20at%205%3A00%20pm%3F' \
  -H 'accept: application/json'
```

Request URL

```
http://localhost:8000/ask/?query=Am%20I%20busy%20Wednesday%20at%205%3A00%20pm%3F
```

Server response

Code	Details
404 *Undocumented*	Error: Not Found

Response body

```
{
  "detail": "No documents found in the knowledge base."
}
```

Response headers

```
 content-length: 54
 content-type: application/json
 date: Thu,20 Feb 2025 20:57:30 GMT
 server: uvicorn
```

Responses

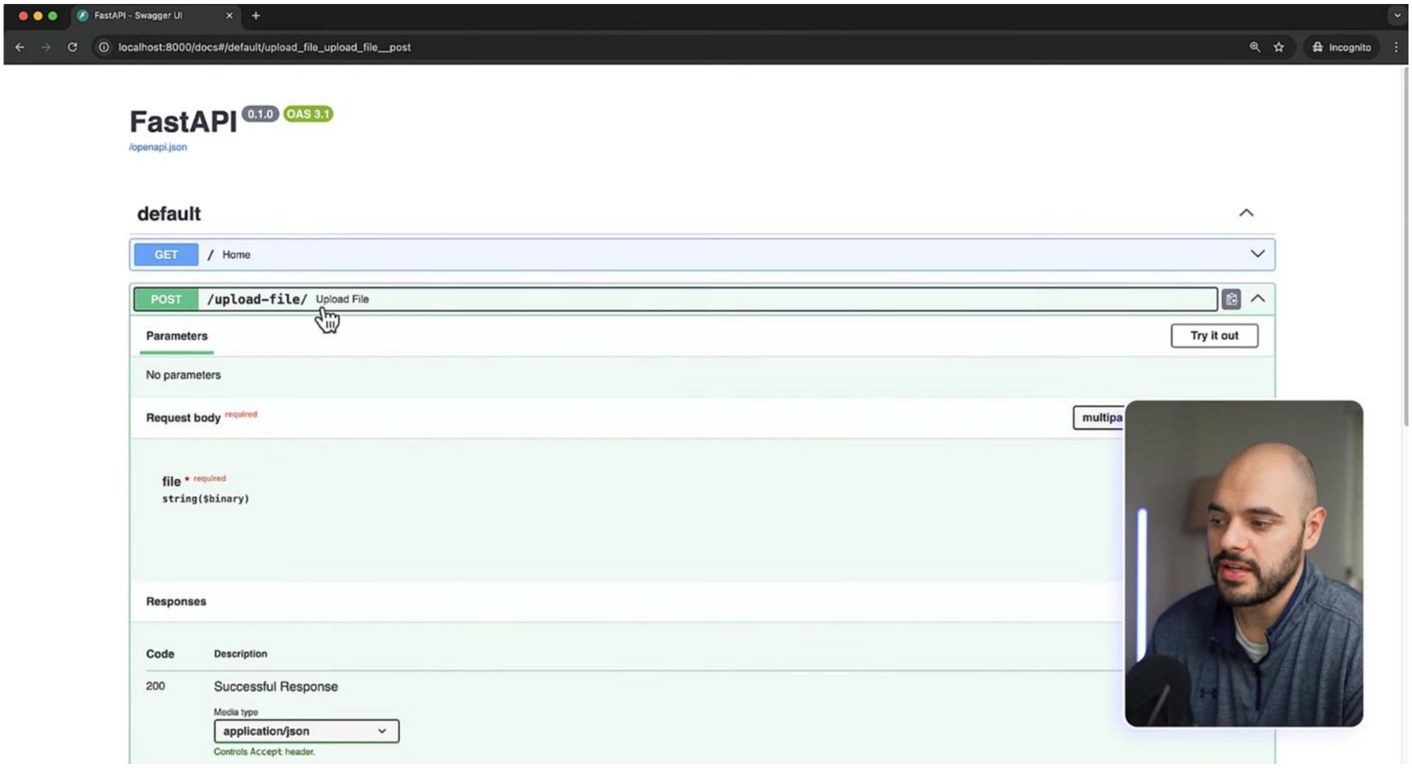
Code	Description
200	Successful Response

Media type

application/json

Controls Accept header.

Example Value | Schema



FastAPI 0.1.0 OAS 3.1

/openapi.json

default

GET / Home

POST /upload-file/ Upload File

Parameters

Try it out

No parameters

Request body required

multipa

file * required
string($binary)

Responses

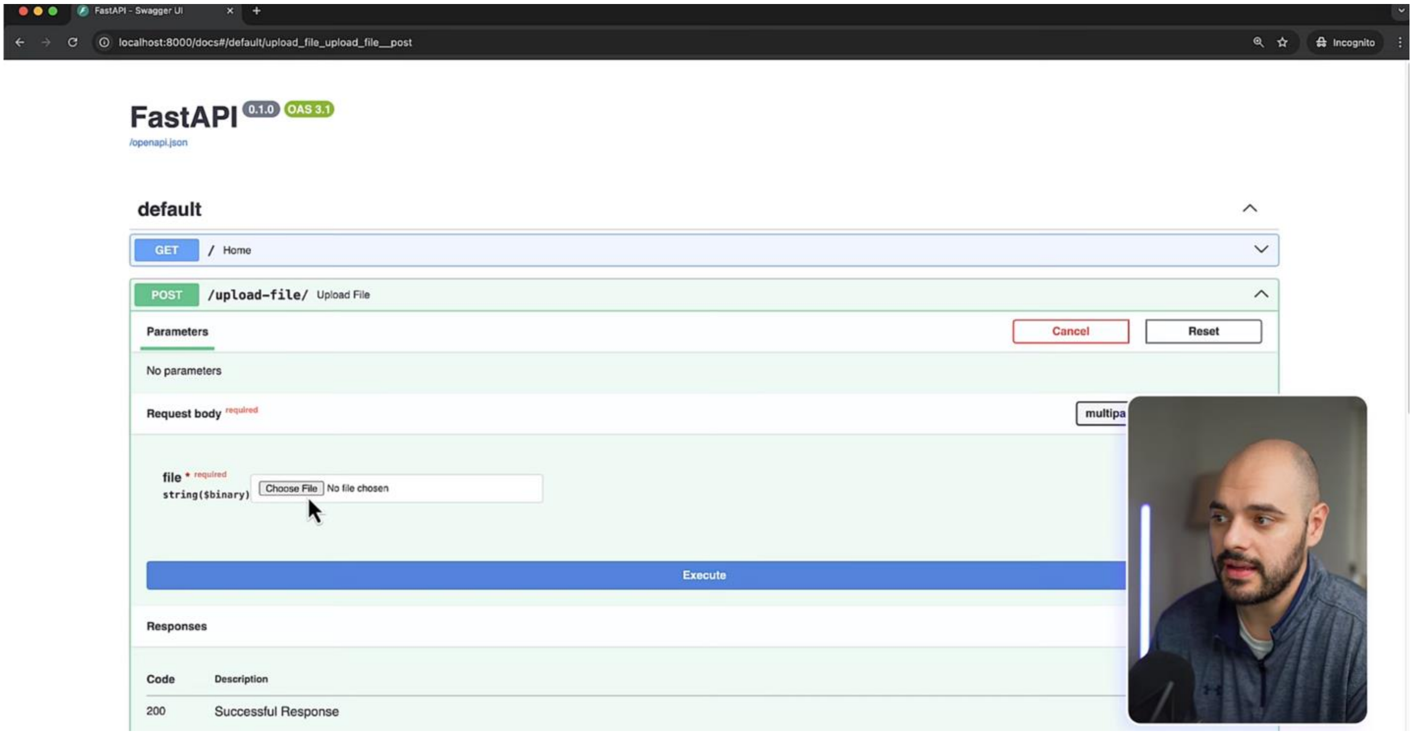
Code Description

200 Successful Response

Media type

application/json

Controls Accept header.



FastAPI 0.1.0 OAS 3.1

/openapi.json

default

GET / Home

POST /upload-file/ Upload File

Parameters

Cancel Reset

No parameters

Request body required

multipa

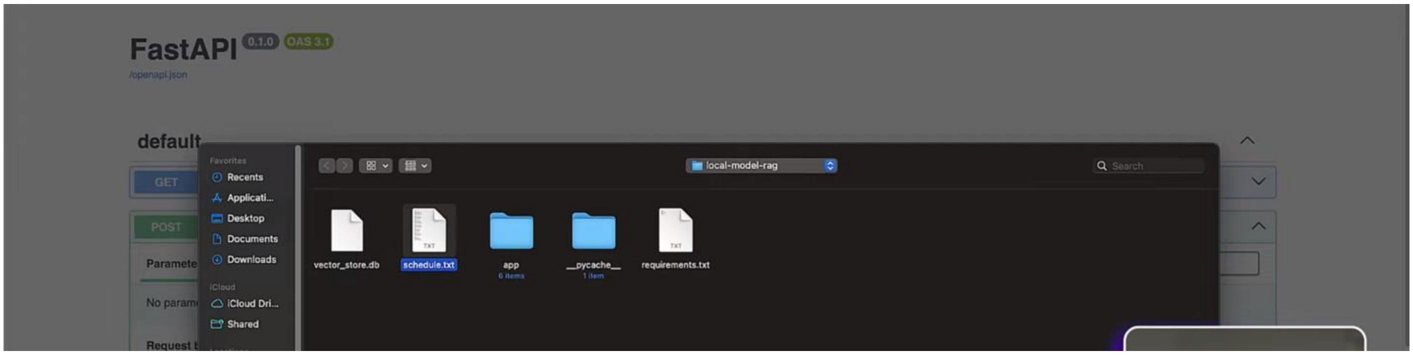
file * required
string($binary) Choose File No file chosen

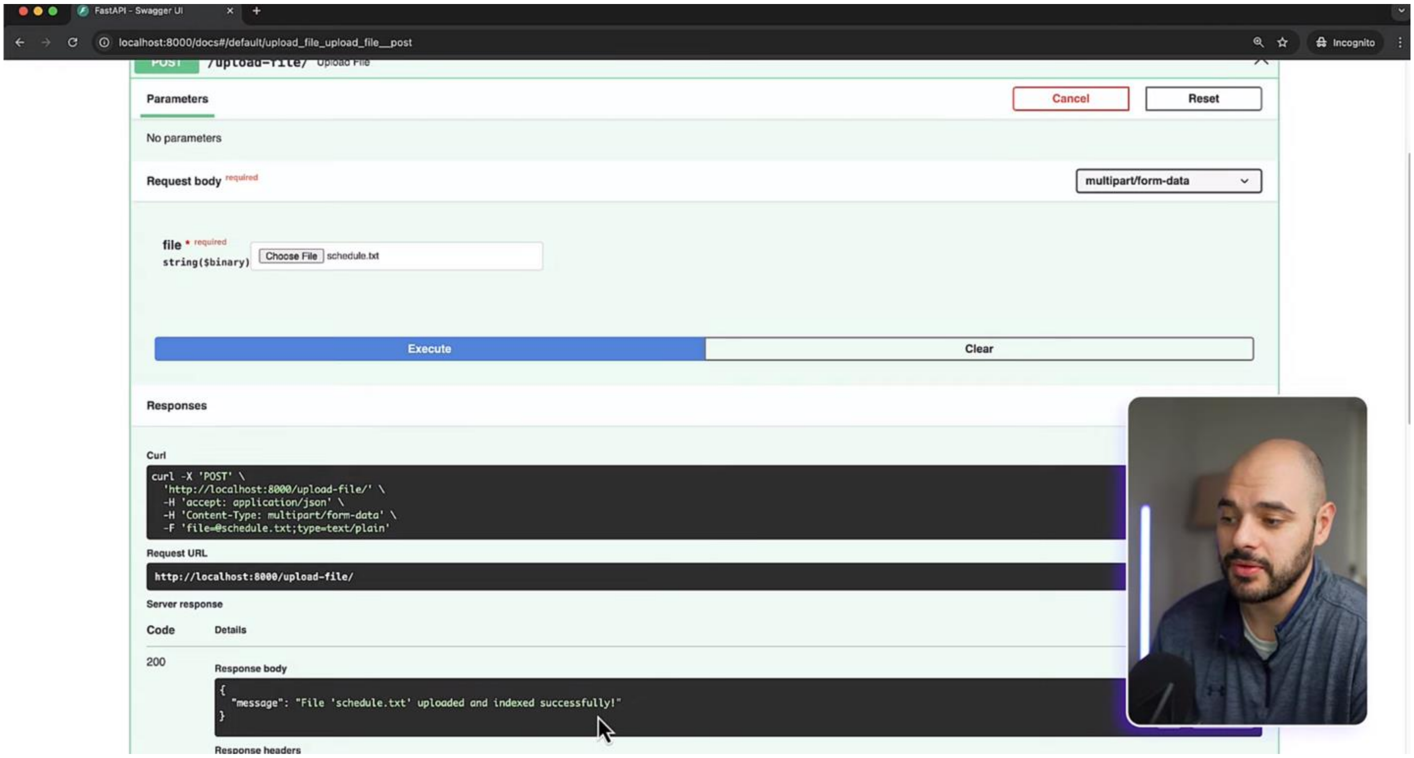
Execute

Responses

Code Description

200 Successful Response





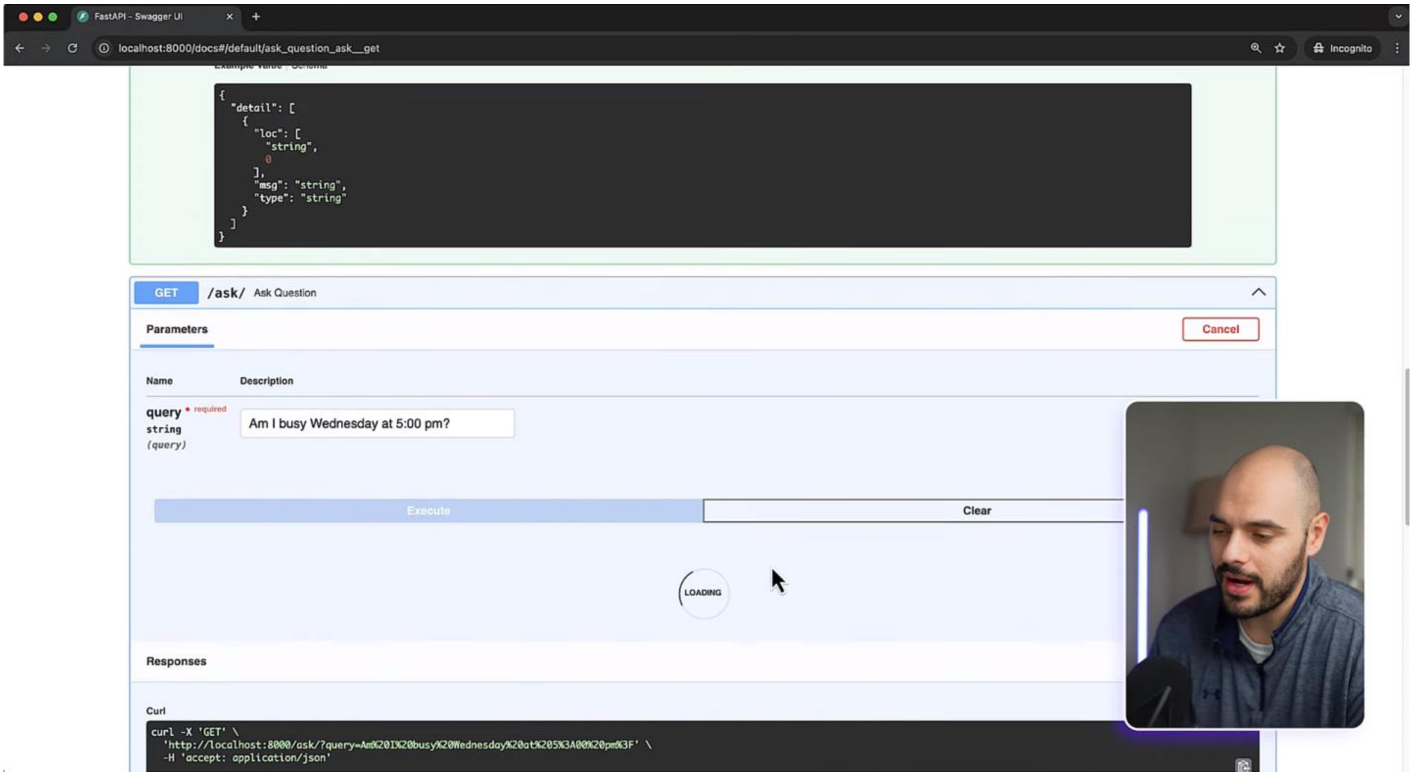
file: schedule.txt

```
curl -X 'POST' \
  'http://localhost:8000/upload-file/' \
  -H 'accept: application/json' \
  -H 'Content-Type: multipart/form-data' \
  -F 'file=@schedule.txt;type=text/plain'
```

Request URL: `http://localhost:8000/upload-file/`

Code: 200

```
{
  "message": "File 'schedule.txt' uploaded and indexed successfully!"
}
```

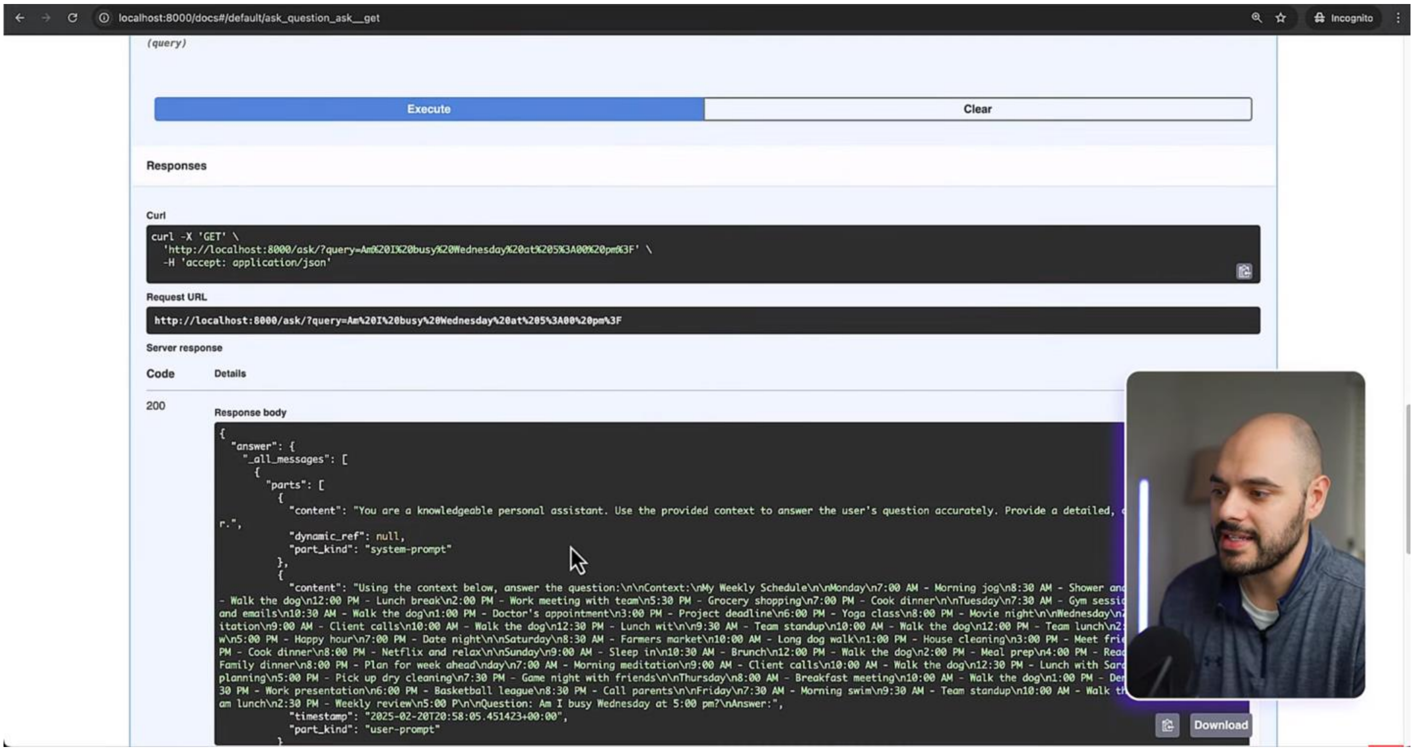


```
{
  "detail": [
    {
      "loc": [
        "string",
        0
      ],
      "msg": "string",
      "type": "string"
    }
  ]
}
```

query: Am I busy Wednesday at 5:00 pm?

```
curl -X 'GET' \
  'http://localhost:8000/ask/?query=Am%20I%20busy%20Wednesday%20at%205%3A00%20pm%3F' \
  -H 'accept: application/json'
```

We should now get a response based on the text that we just submitted.



Responses

Curl

```
curl -X 'GET' \
  'http://localhost:8000/ask/?query=Am%20I%20busy%20Wednesday%20at%205%3A00%20pm%3F' \
  -H 'accept: application/json'
```

Request URL

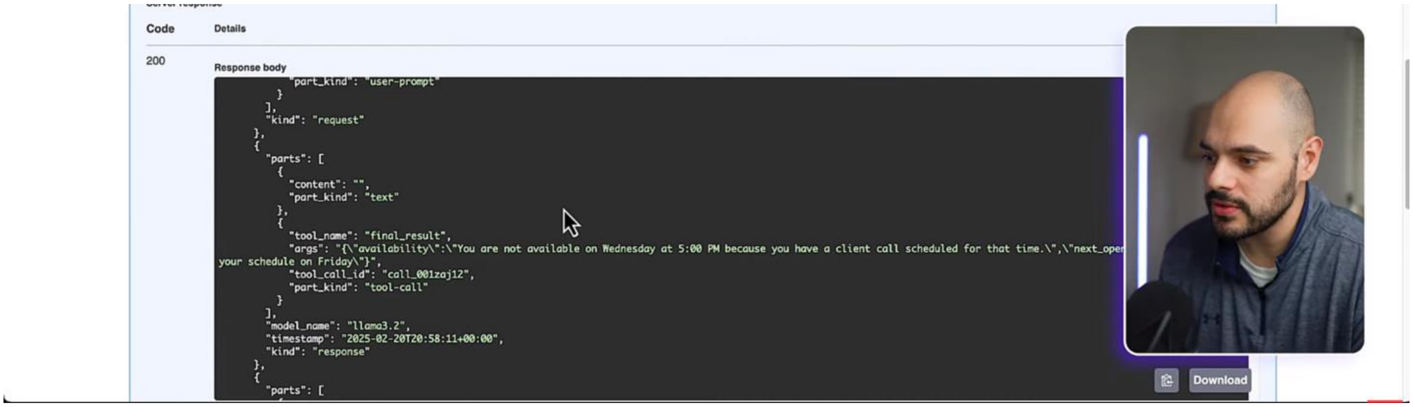
```
http://localhost:8000/ask/?query=Am%20I%20busy%20Wednesday%20at%205%3A00%20pm%3F
```

Server response

Code Details

200 Response body

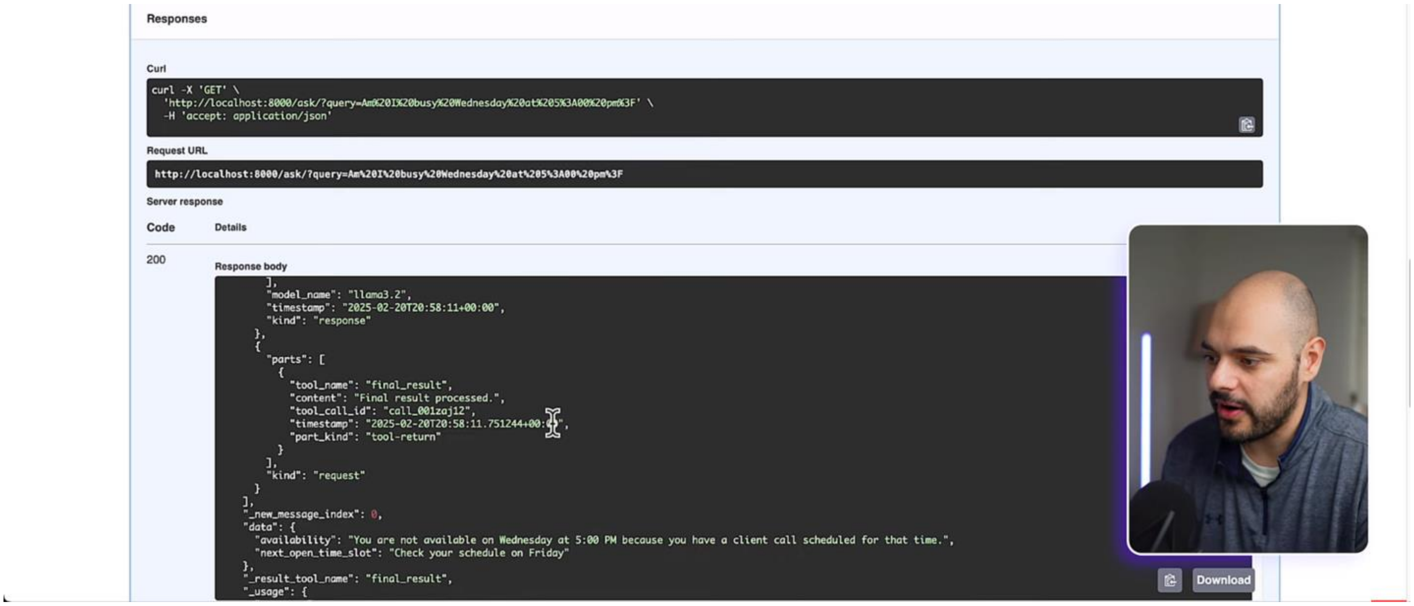
```
{
  "answer": {
    "_all_messages": [
      {
        "parts": [
          {
            "content": "You are a knowledgeable personal assistant. Use the provided context to answer the user's question accurately. Provide a detailed, c
r.",
            "dynamic_ref": null,
            "part_kind": "system-prompt"
          },
          {
            "content": "Using the context below, answer the question:\n\nContext:\nMy Weekly Schedule\n\nMonday\n7:00 AM - Morning jog\n8:30 AM - Shower and
- Walk the dog\n12:00 PM - Lunch break\n2:00 PM - Work meeting with team\n5:30 PM - Grocery shopping\n7:00 PM - Cook dinner\n\nTuesday\n7:30 AM - Gym sessio
and emails\n10:30 AM - Walk the dog\n1:00 PM - Doctor's appointment\n3:00 PM - Project deadline\n6:00 PM - Yoga class\n8:00 PM - Movie night\n\nWednesday\n7
itation\n9:00 AM - Client calls\n10:00 AM - Walk the dog\n12:30 PM - Lunch wit\n\n9:30 AM - Team standup\n10:00 AM - Walk the dog\n12:00 PM - Team lunch\n2:
w\n5:00 PM - Happy hour\n7:00 PM - Date night\n\nSaturday\n8:30 AM - Farmers market\n10:00 AM - Long dog walk\n1:00 PM - House cleaning\n3:00 PM - Meet frie
PM - Cook dinner\n8:00 PM - Netflix and relax\n\nSunday\n9:00 AM - Sleep in\n10:30 AM - Brunch\n12:00 PM - Walk the dog\n2:00 PM - Meal prep\n4:00 PM - Read
Family dinner\n8:00 PM - Plan for week ahead\nday\n7:00 AM - Morning meditation\n9:00 AM - Client calls\n10:00 AM - Walk the dog\n12:30 PM - Lunch with Sara
planning\n5:00 PM - Pick up dry cleaning\n7:30 PM - Game night with friends\n\nThursday\n8:00 AM - Breakfast meeting\n10:00 AM - Walk the dog\n1:00 PM - Den
30 PM - Work presentation\n6:00 PM - Basketball league\n8:30 PM - Call parents\n\nFriday\n7:30 AM - Morning swim\n9:30 AM - Team standup\n10:00 AM - Walk th
am lunch\n2:30 PM - Weekly review\n5:00 P\n\nQuestion: Am I busy Wednesday at 5:00 pm?\nAnswer:",
            "timestamp": "2025-02-20T20:58:05.451423+00:00",
            "part_kind": "user-prompt"
          }
```



Code Details

200 Response body

```
            "part_kind": "user-prompt"
          }
        ],
        "kind": "request"
      },
      {
        "parts": [
          {
            "content": "",
            "part_kind": "text"
          },
          {
            "tool_name": "final_result",
            "args": "{\"availability\":\"You are not available on Wednesday at 5:00 PM because you have a client call scheduled for that time.\",\"next_open
your schedule on Friday\"}",
            "tool_call_id": "call_001zaj12",
            "part_kind": "tool-call"
          }
        ],
        "model_name": "llama3.2",
        "timestamp": "2025-02-20T20:58:11+00:00",
        "kind": "response"
      },
      {
        "parts": [
```



Responses

Curl

```
curl -X 'GET' \
  'http://localhost:8000/ask/?query=Am%20I%20busy%20Wednesday%20at%205%3A00%20pm%3F' \
  -H 'accept: application/json'
```

Request URL

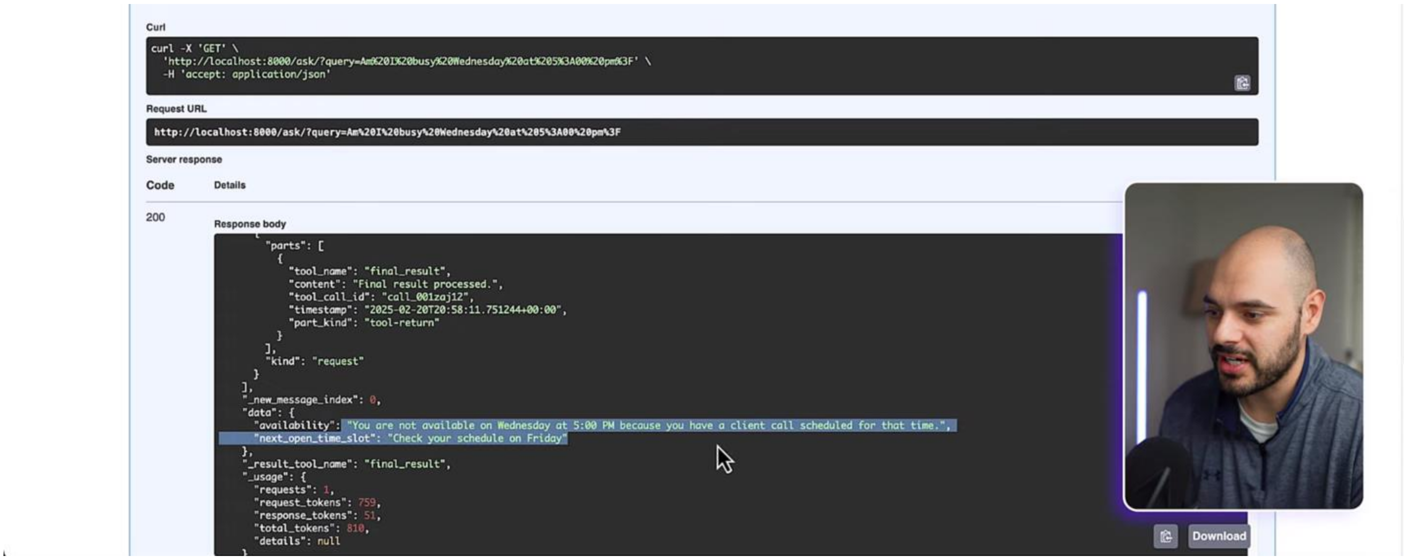
```
http://localhost:8000/ask/?query=Am%20I%20busy%20Wednesday%20at%205%3A00%20pm%3F
```

Server response

Code Details

200 Response body

```
        ],
        "model_name": "llama3.2",
        "timestamp": "2025-02-20T20:58:11+00:00",
        "kind": "response"
      },
      {
        "parts": [
          {
            "tool_name": "final_result",
            "content": "Final result processed.",
            "tool_call_id": "call_001zaj12",
            "timestamp": "2025-02-20T20:58:11.751244+00:00",
            "part_kind": "tool-return"
          }
        ],
        "kind": "request"
      }
    ],
    "_new_message_index": 0,
    "data": {
      "availability": "You are not available on Wednesday at 5:00 PM because you have a client call scheduled for that time.",
      "next_open_time_slot": "Check your schedule on Friday"
    },
    "_result_tool_name": "final_result",
    "_usage": {
```



Curl

```
curl -X 'GET' \
  'http://localhost:8000/ask/?query=Am%20I%20busy%20Wednesday%20at%205%3A00%20pm%3F' \
  -H 'accept: application/json'
```

Request URL

```
http://localhost:8000/ask/?query=Am%20I%20busy%20Wednesday%20at%205%3A00%20pm%3F
```

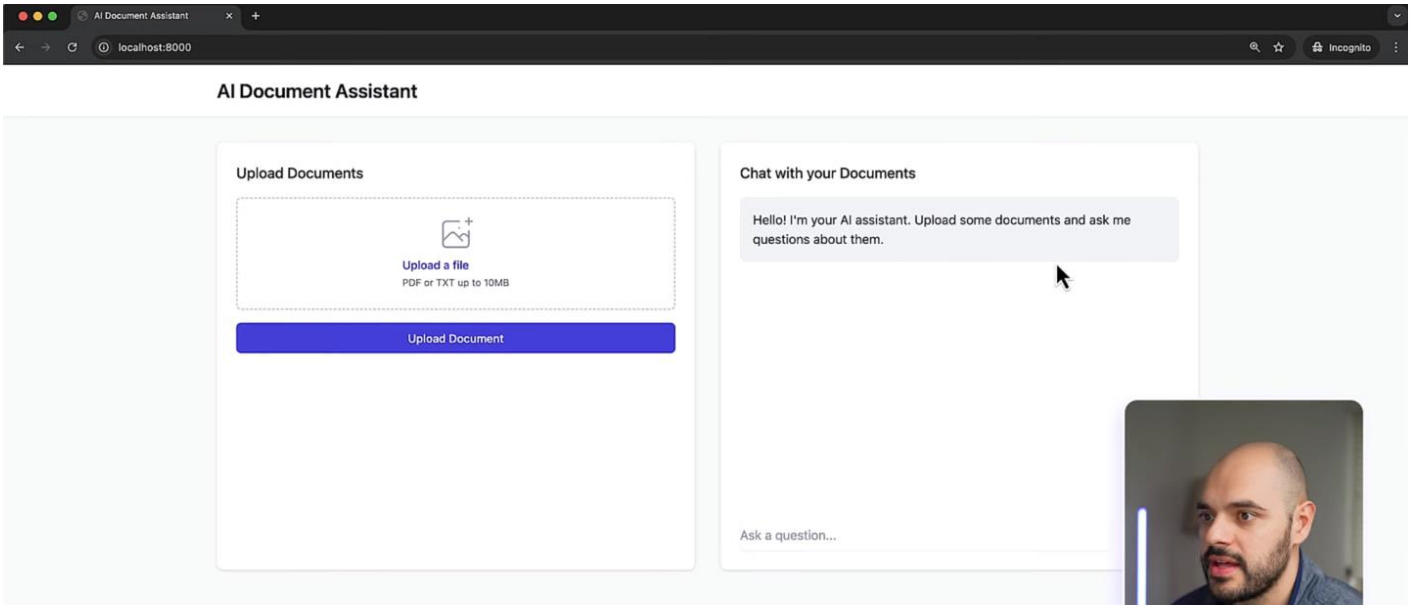
Server response

Code Details

200

Response body

```
        "parts": [
          {
            "tool_name": "final_result",
            "content": "Final result processed.",
            "tool_call_id": "call_001zaj12",
            "timestamp": "2025-02-20T20:58:11.751244+00:00",
            "part_kind": "tool-return"
          }
        ],
        "kind": "request"
      }
    ],
    "_new_message_index": 0,
    "data": {
      "availability": "You are not available on Wednesday at 5:00 PM because you have a client call scheduled for that time.",
      "next_open_time_slot": "Check your schedule on Friday"
    },
    "_result_tool_name": "final_result",
    "_usage": {
      "requests": 1,
      "request_tokens": 759,
      "response_tokens": 51,
      "total_tokens": 810,
      "details": null
```

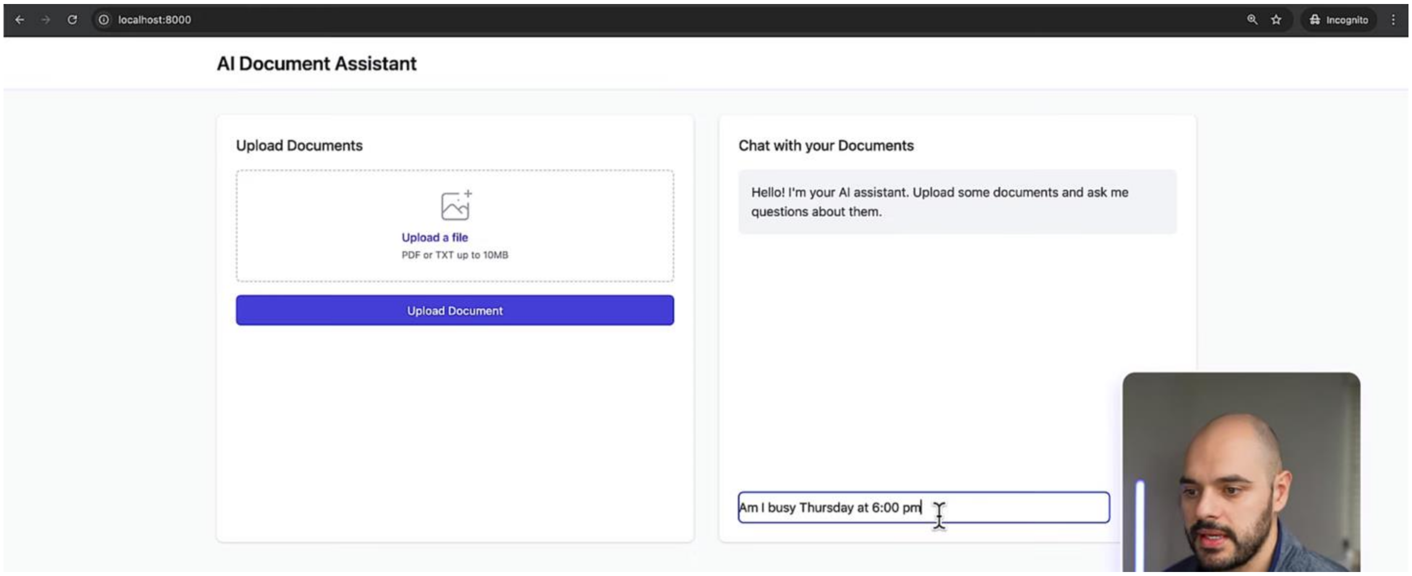


AI Document Assistant

Upload Documents

Chat with your Documents

Hello! I'm your AI assistant. Upload some documents and ask me questions about them.



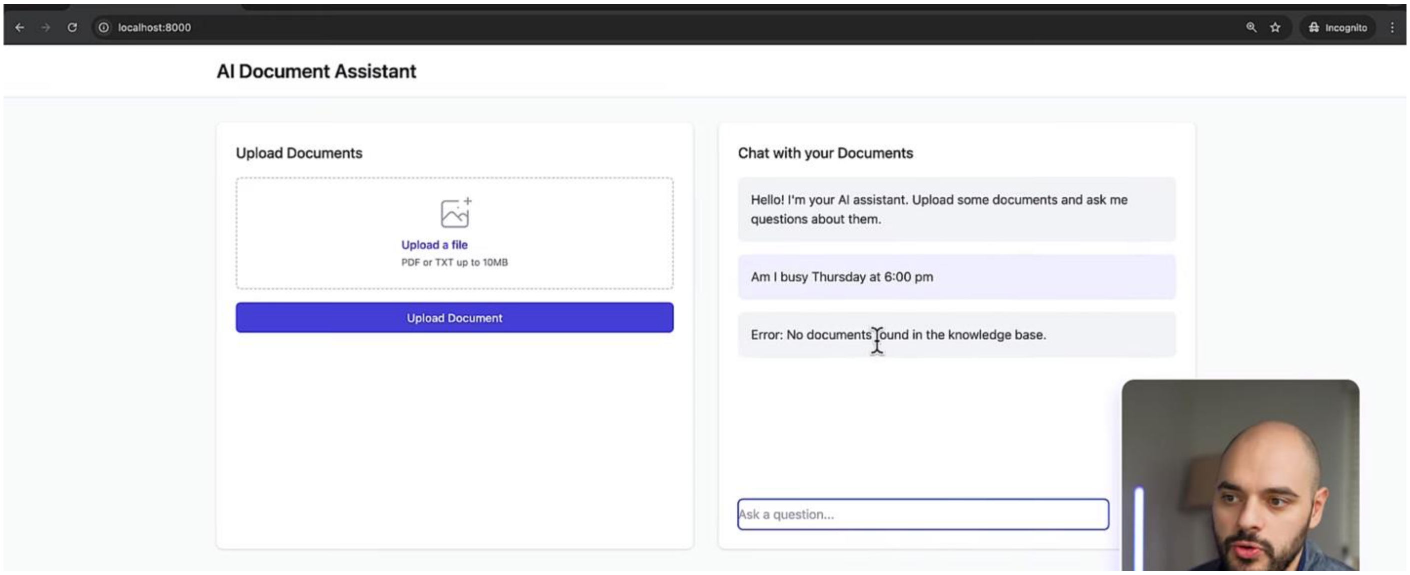
AI Document Assistant

Upload Documents

Chat with your Documents

Hello! I'm your AI assistant. Upload some documents and ask me questions about them.

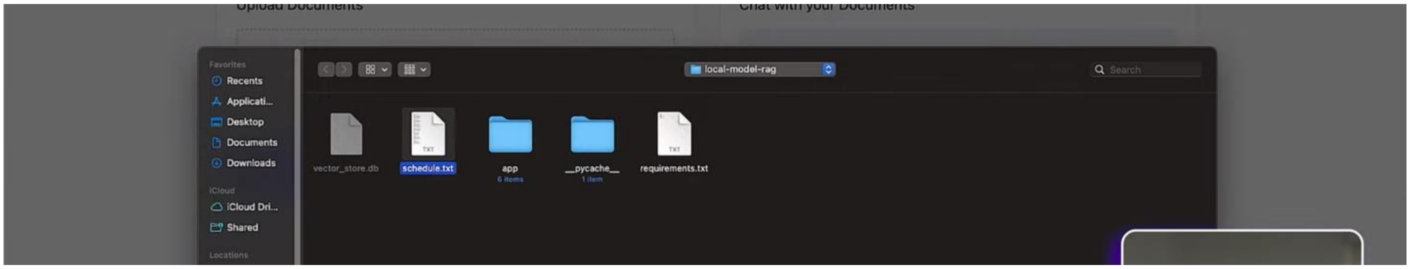
Am I busy Thursday at 6:00 pm

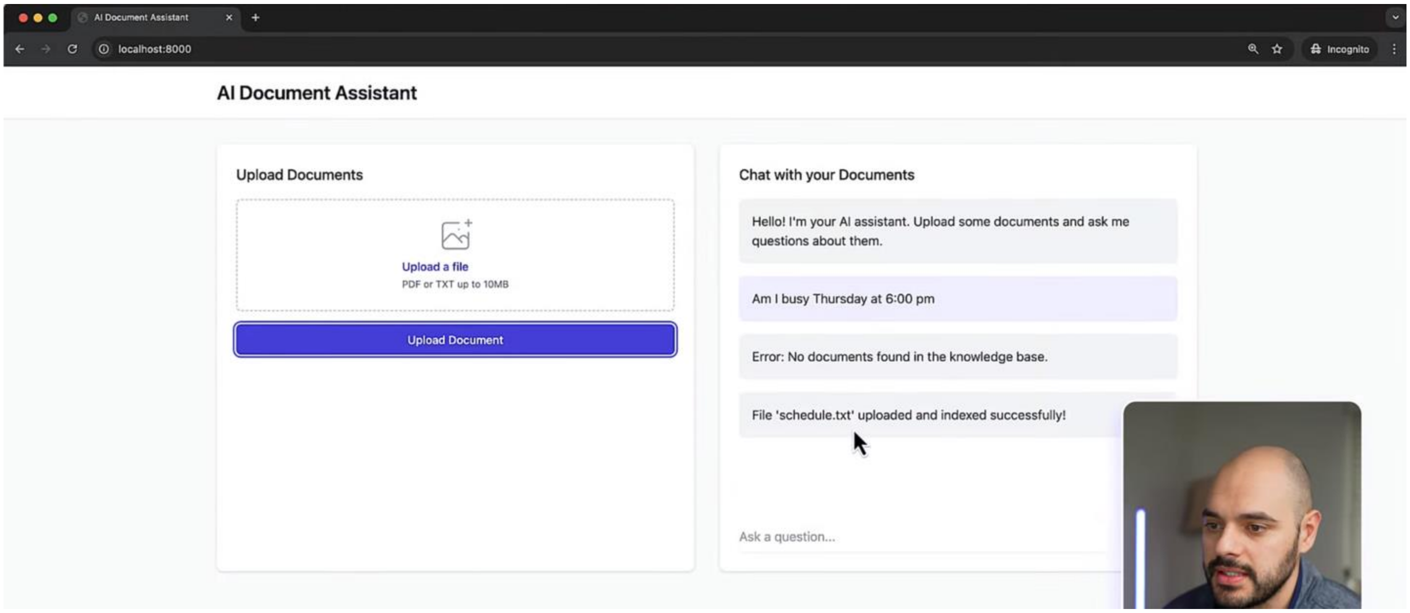


Hello! I'm your AI assistant. Upload some documents and ask me questions about them.

Am I busy Thursday at 6:00 pm

Error: No documents found in the knowledge base.



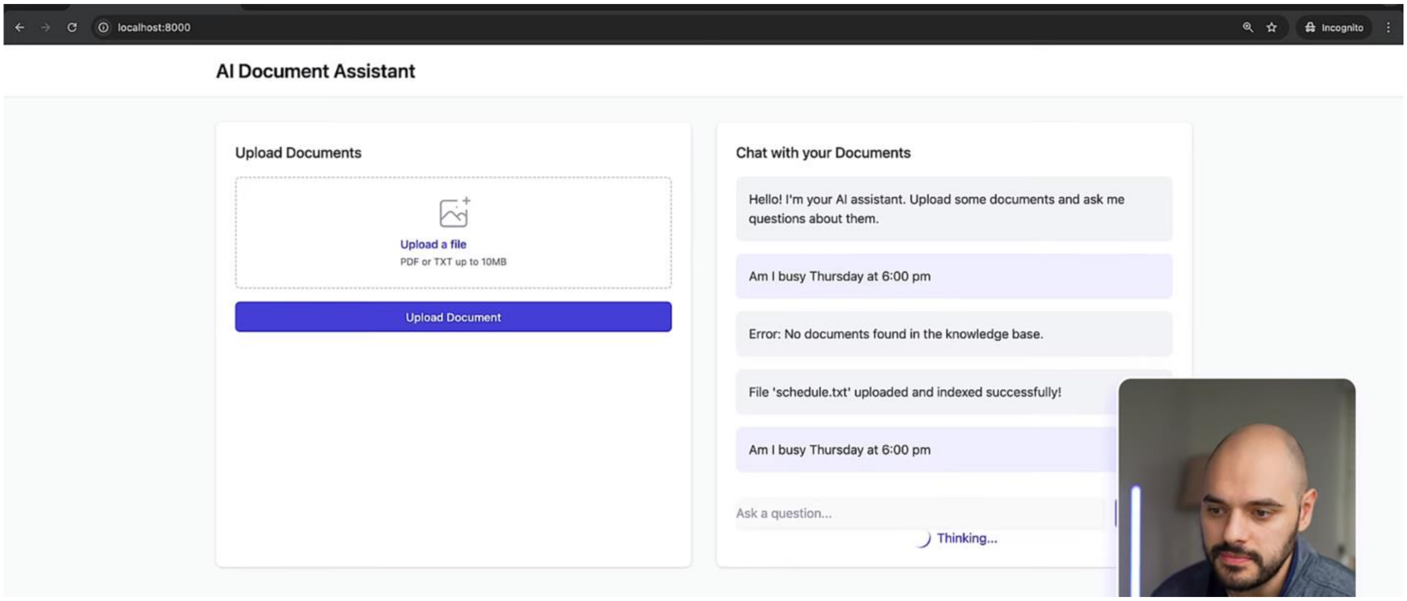


Hello! I'm your AI assistant. Upload some documents and ask me questions about them.

Am I busy Thursday at 6:00 pm

Error: No documents found in the knowledge base.

File 'schedule.txt' uploaded and indexed successfully!



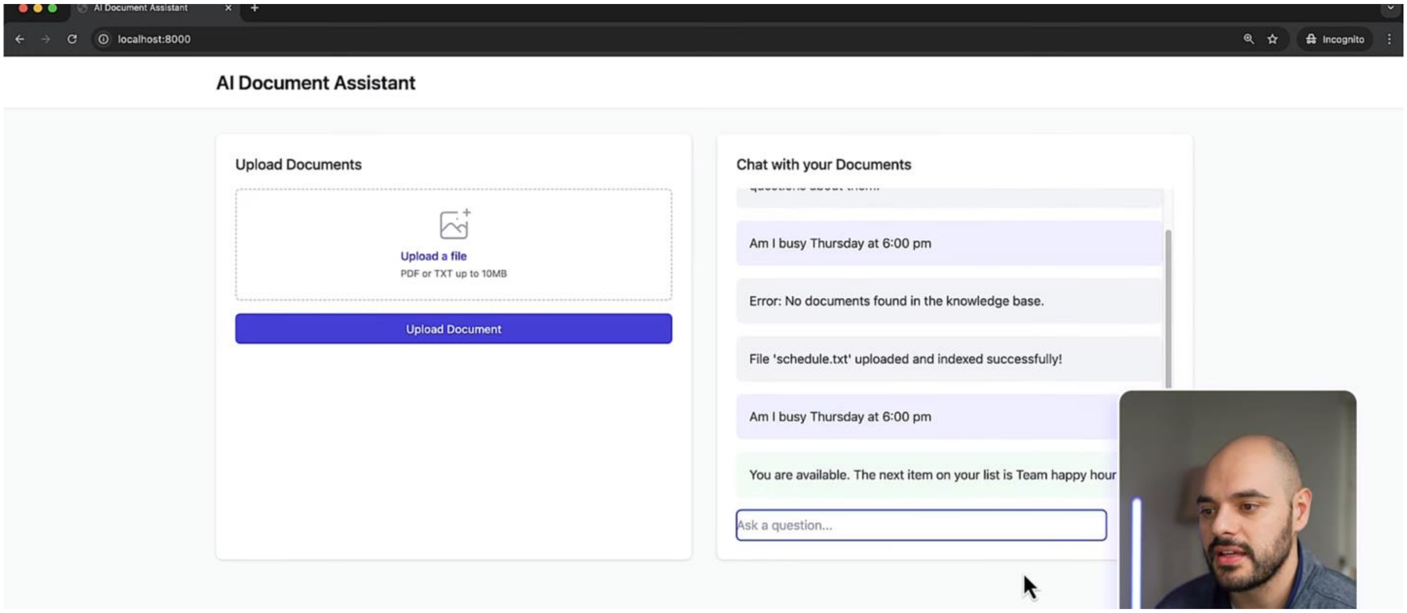
Hello! I'm your AI assistant. Upload some documents and ask me questions about them.

Am I busy Thursday at 6:00 pm

Error: No documents found in the knowledge base.

File 'schedule.txt' uploaded and indexed successfully!

Am I busy Thursday at 6:00 pm



questions about them.

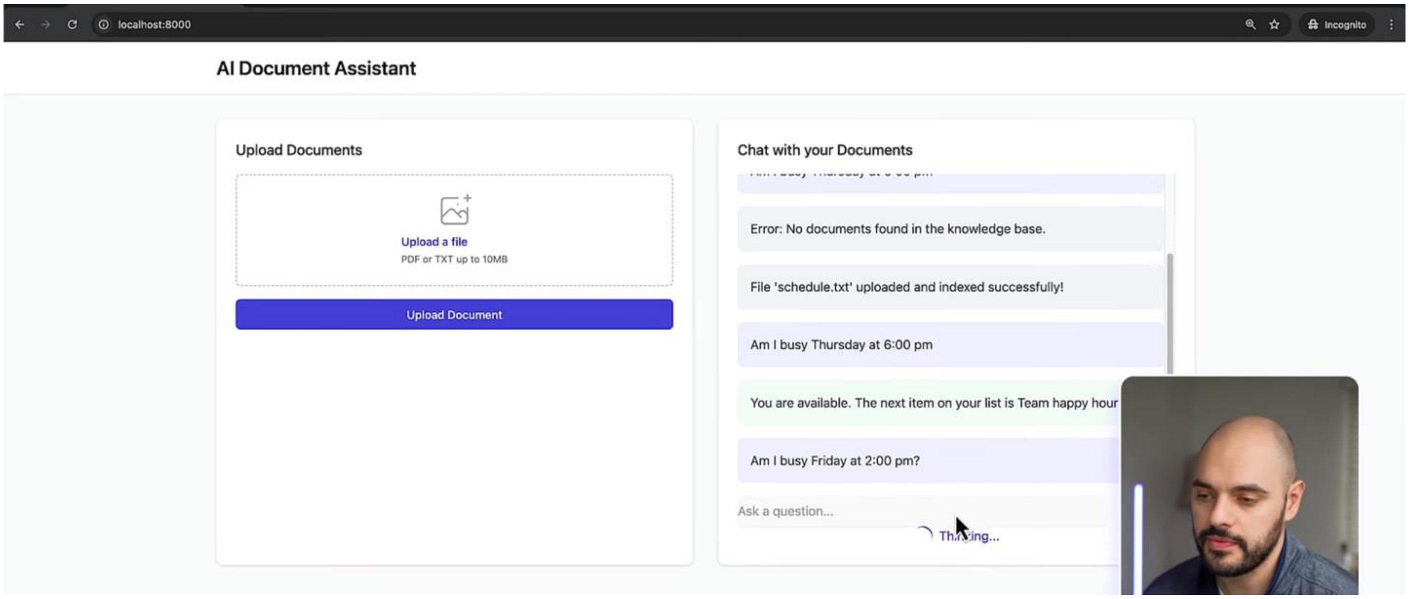
Am I busy Thursday at 6:00 pm

Error: No documents found in the knowledge base.

File 'schedule.txt' uploaded and indexed successfully!

Am I busy Thursday at 6:00 pm

You are available. The next item on your list is Team happy hour



Am I busy Thursday at 6:00 pm

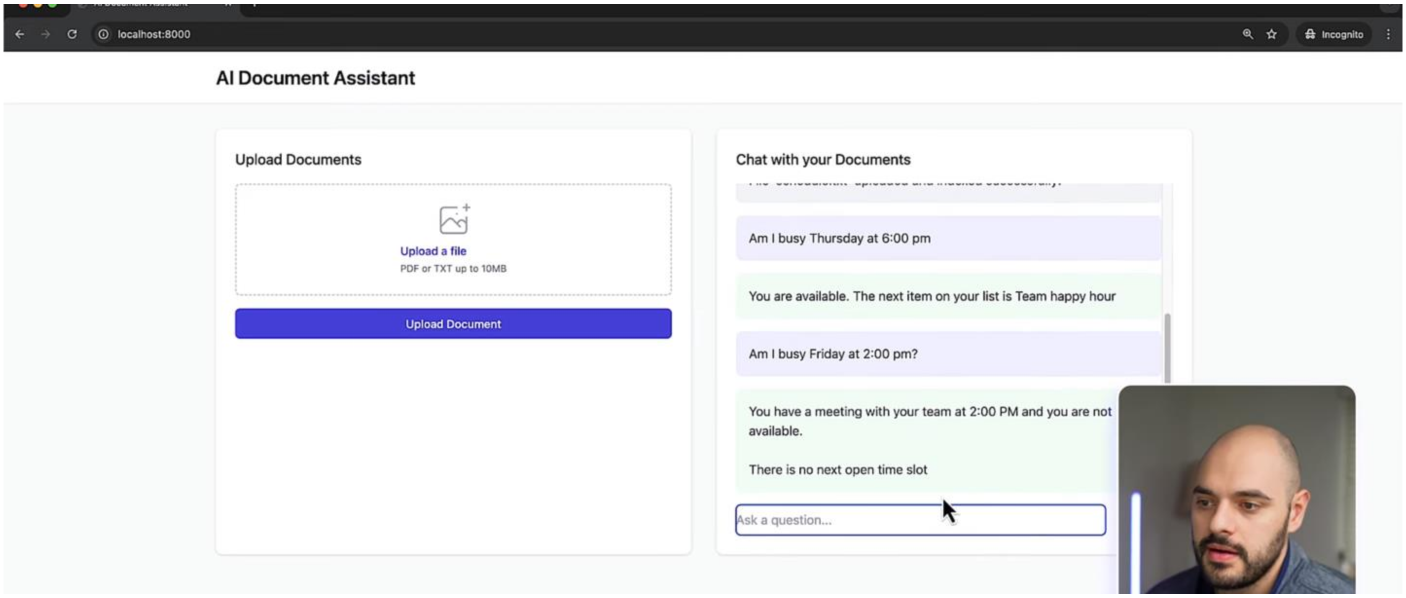
Error: No documents found in the knowledge base.

File 'schedule.txt' uploaded and indexed successfully!

Am I busy Thursday at 6:00 pm

You are available. The next item on your list is Team happy hour

Am I busy Friday at 2:00 pm?



File 'schedule.txt' uploaded and indexed successfully!

Am I busy Thursday at 6:00 pm

You are available. The next item on your list is Team happy hour

Am I busy Friday at 2:00 pm?

You have a meeting with your team at 2:00 PM and you are not available.

There is no next open time slot