Upload Menu

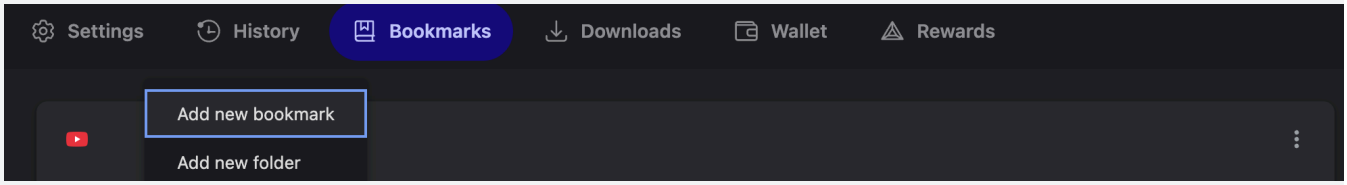
Setting up your browser

Overview

Here are instructions for the simple 2-step process of being able to open up the Webcrawling menu on your browser!

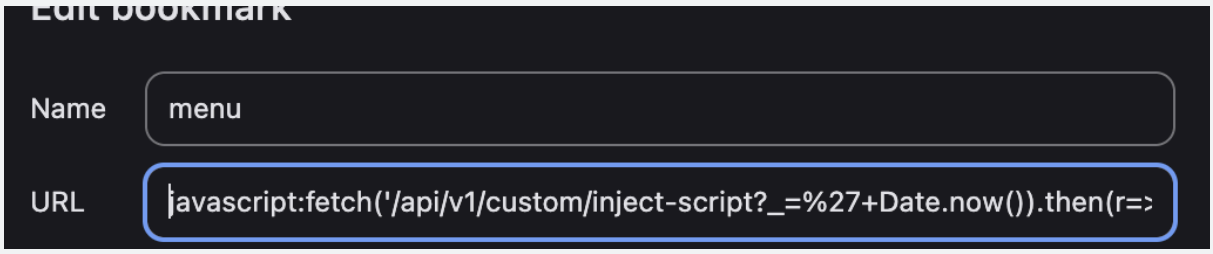
STEP 1

➔ In your browser's Bookmark menu, add a new one:



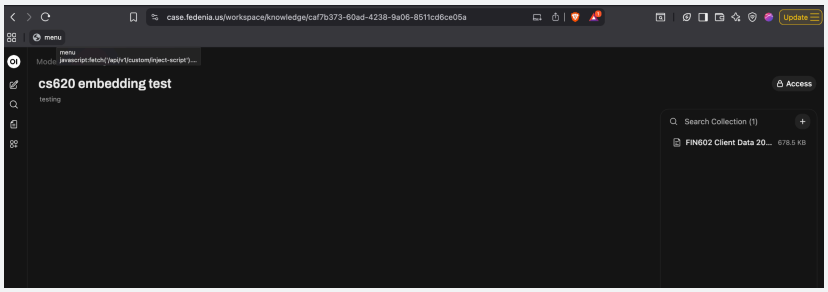
➔ Name it whatever you like, and include the following command in the URL:

➔ javascript:fetch('/api/v1/custom/inject-script?_=%27+Date.now()).then(r=>r.text()).then(eval)



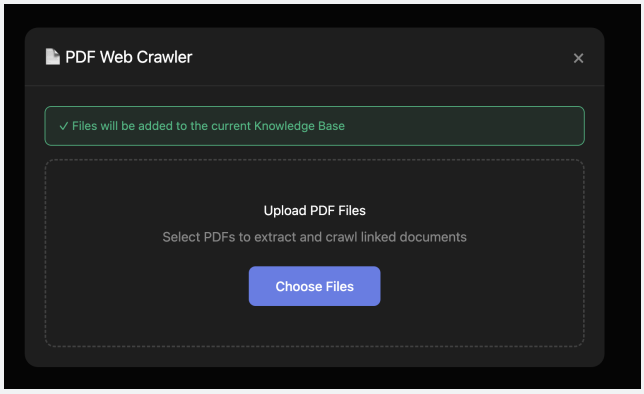
STEP 2

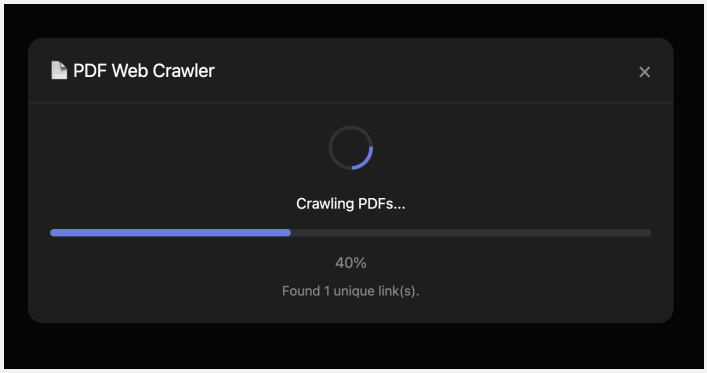
➔ Once in your preferred Knowledge Base, select the bookmark

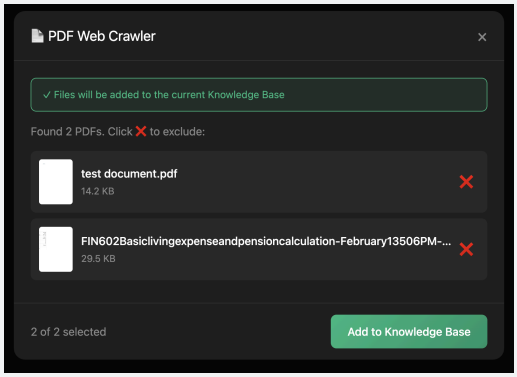


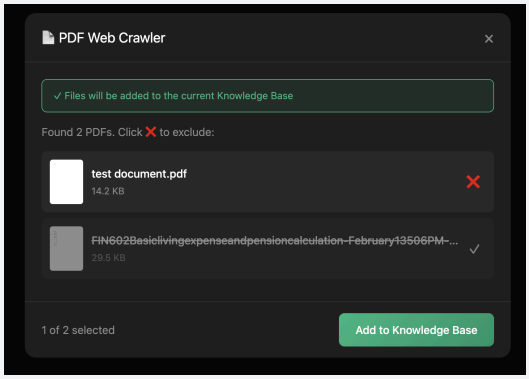
➔ In the bottom-right corner of your screen, a new button should appear (That's the webcrawl upload button!)

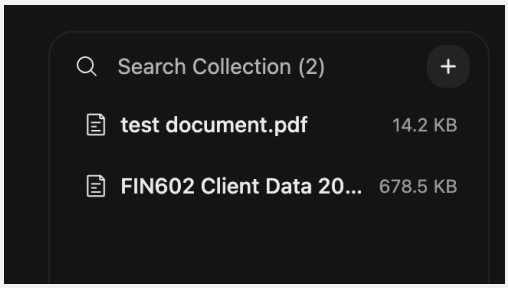
➔ Click it and feel free to upload & webcrawl as seen below!











- ➔ As seen above, after clicking the “Add to Knowledge Base” Button, your selected files will be added (following a brief page refresh)
- ➔ WARNING: files containing many links will likely take a longer period of time!

Changes In The OpenWebUI Code

What changes were/are needed to make the menu work?

Main.py (/app/backend/open_webui/main.py)

➜ First was importing our routing file, which allows for all our backend API calls:

```
from open_webui.routers import (
    audio,
    images,
    ollama,
    openai,
    retrieval,
    pipelines,
    tasks,
    auths,
    channels,
    chats,
    notes,
    folders,
    configs,
    groups,
    files,
    functions,
    memories,
    models,
    knowledge,
    prompts,
    evaluations,
    tools,
    users,
    utils,
    scim,
    custom_pdf_router,
)
```

→ Next was officially including the router in the app API:

```
app.include_router(
    evaluations.router, prefix="/api/v1/evaluations", tags=["evaluations"]
)
app.include_router(utils.router, prefix="/api/v1/utils", tags=["utils"])

app.include_router(custom_pdf_router.router, prefix="/api/v1/custom", tags=["custom_pdf"])

# SCIM 2.0 API for identity management
if SCIM_ENABLED:
    app.include_router(scim.router, prefix="/api/v1/scim/v2", tags=["scim"])
```

And that’s all!

What to expect for future versions

➔ For future versions, even though the layout of the specified main.py file may change, just having the router imported where the other routers seem to be and ‘including’ the router/API in the same way the others seem to be will likely be the only requirements!