publications

MDPI

Article

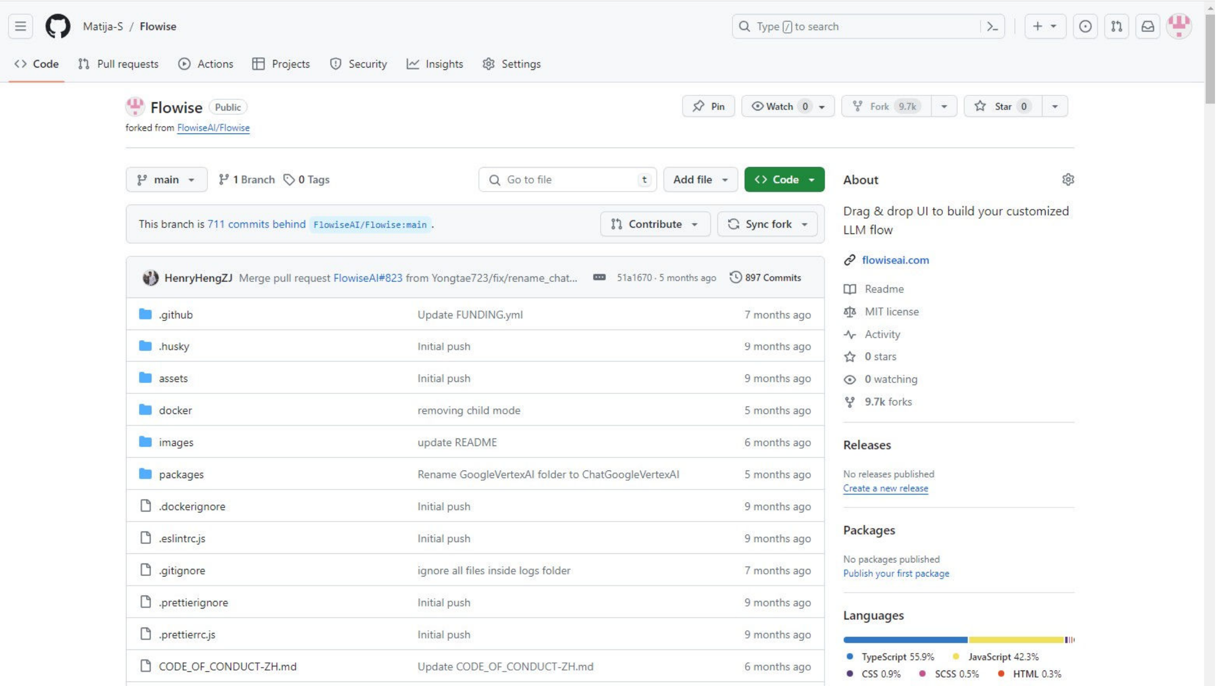
Application of ChatGPT in Information Literacy Instructional Design

Jelena Madunić [1] * and Matija Sovulj [2]

[1] University of Split Library; jmadunic@svkst.hr
[2] University of Split Library; msovulj@svkst.hr
* Author to whom correspondence should be addressed

Step-by-step guide: Developing a custom chat model

For new users, preliminary requirement is signing up for an OpenAI account, and a GitHub account. The first step is accessing Flowise repository at GitHub, as shown in Figure B1. In this application, Flowise provides drag & drop UI needed to build the customized LLM flow.



This branch is 711 commits behind `FlowiseAI/Flowise:main`.

HenryHengZJ Merge pull request FlowiseAI#823 from Yongtae723/fix/rename_chat... 51a1670 · 5 months ago 897 Commits

.github	Update FUNDING.yml	7 months ago
.husky	Initial push	9 months ago
assets	Initial push	9 months ago
docker	removing child mode	5 months ago
images	update README	6 months ago
packages	Rename GoogleVertexAI folder to ChatGoogleVertexAI	5 months ago
.dockerignore	Initial push	9 months ago
.eslintrc.js	Initial push	9 months ago
.gitignore	ignore all files inside logs folder	7 months ago
.prettierignore	Initial push	9 months ago
.prettierrc.js	Initial push	9 months ago
CODE_OF_CONDUCT-ZH.md	Update CODE_OF_CONDUCT-ZH.md	6 months ago

About

Drag & drop UI to build your customized LLM flow

flowiseai.com
Readme
MIT license
Activity
0 stars
0 watching
9.7k forks

Releases

No releases published
Create a new release

Packages

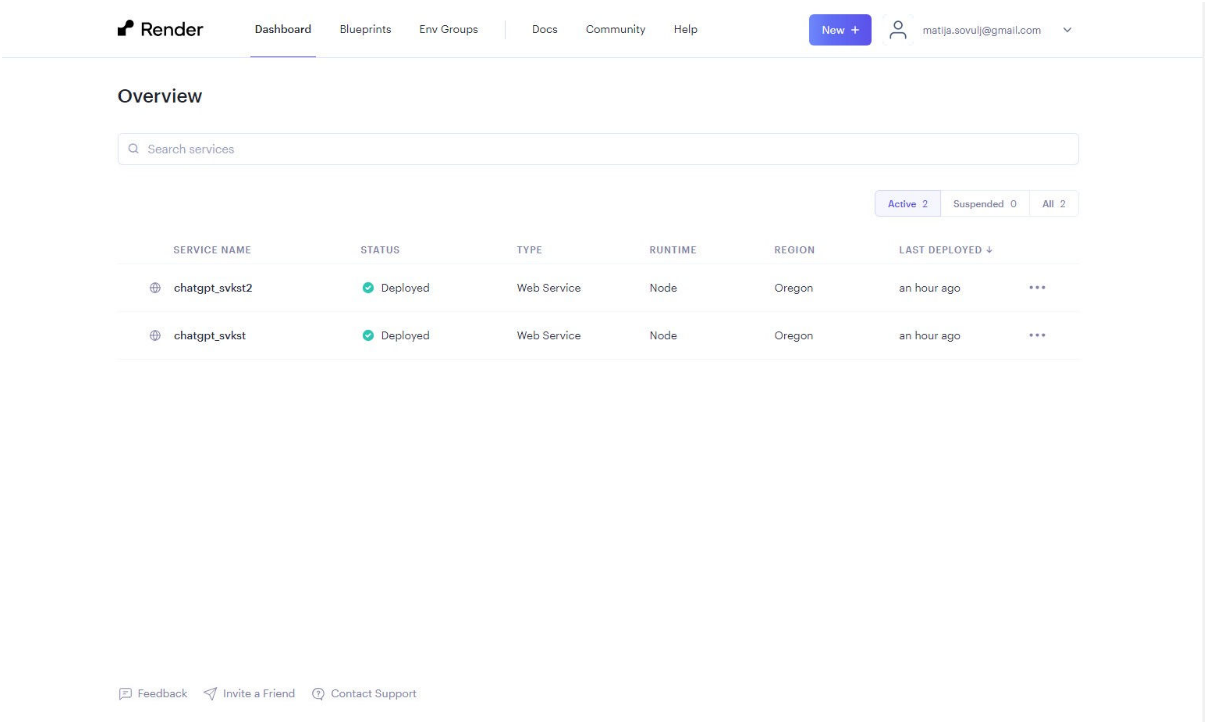
No packages published
Publish your first package

Languages

TypeScript 55.9%
JavaScript 42.3%
CSS 0.9%
SCSS 0.5%
HTML 0.3%

Figure S1. Flowise repository at GitHub.

The second step requires a server, i.e. location where the application will be uploaded. The service used for this application is Render. The Render dashboard provides options for two functions needed for our custom chatbot: connecting with GitHub and deployment of web service. Render dashboard is shown in Figure B2.

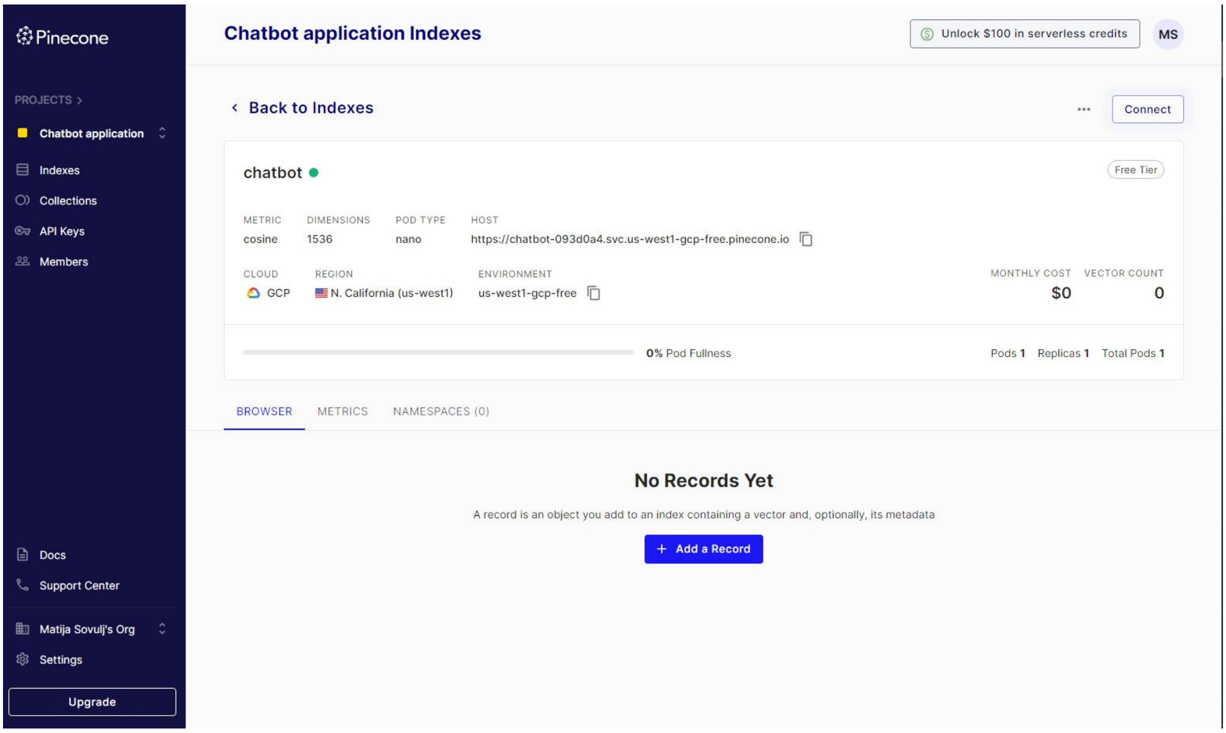


SERVICE NAME	STATUS	TYPE	RUNTIME	REGION	LAST DEPLOYED ↓
chatgpt_svkst2	Deployed	Web Service	Node	Oregon	an hour ago
chatgpt_svkst	Deployed	Web Service	Node	Oregon	an hour ago

Figure S2. Render dashboard.

OpenAI API keys allow developers to access OpenAI's models through the API and build applications that can generate text, images, and code. To get an OpenAI API key, developersneed to sign up for an OpenAI account and request an API key from the OpenAI dashboard. In our case, the OpenAI API key will be needed in the next steps, in the process of creating credentials for Flowise.

Pinecone vector database is needed to upload the .pdf documents which represent our customdataset. First, an index is created on Pinecone dashboard, as shown in Figure B3. Pinecone uses the term ‘index’ as equivalent to repository. Before the upload process, it is necessary tocreate API key on Pinecone as well, to be used for Flowise.



Chatbot application Indexes

chatbot

METRIC	DIMENSIONS	POD TYPE	HOST
cosine	1536	nano	https://chatbot-093d0a4.svc.us-west1-gcp-free.pinecone.io

CLOUD	REGION	ENVIRONMENT	MONTHLY COST	VECTOR COUNT
GCP	N. California (us-west1)	us-west1-gcp-free	$0	0

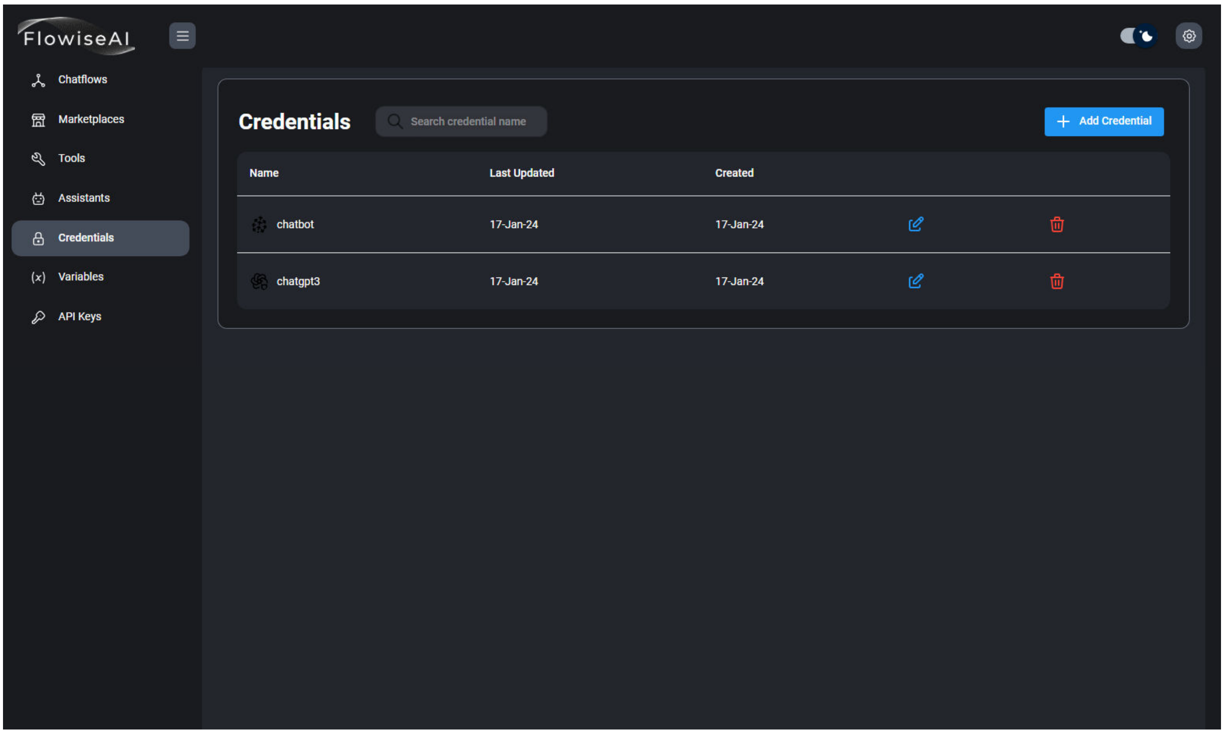
0% Pod Fullness Pods 1 Replicas 1 Total Pods 1

No Records Yet

A record is an object you add to an index containing a vector and, optionally, its metadata

Figure S3. Pinecone vector database.

On the FlowiseAI dashboard, credentials are created for both OpenAI and Pinecone, asshown in Figure B4. In this step, OpenAI and Pinecone API keys are used.



Credentials

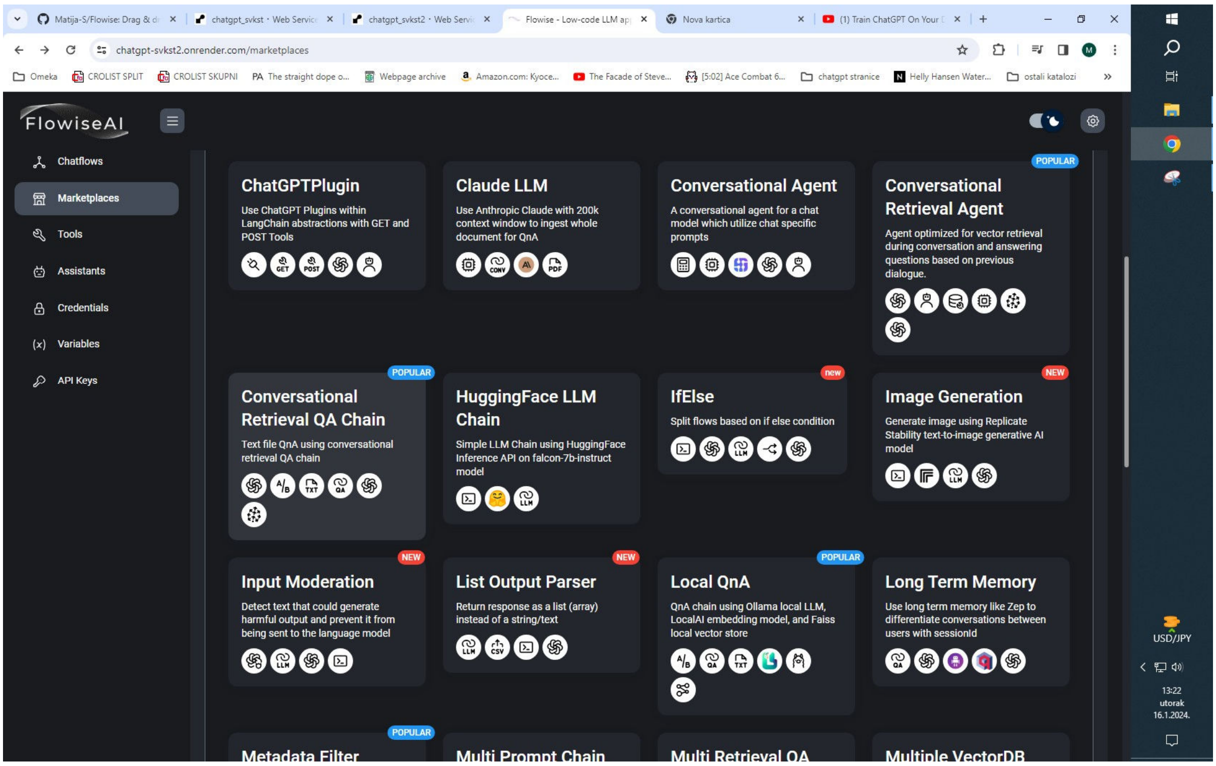
Name	Last Updated	Created
chatbot	17-Jan-24	17-Jan-24
chatgpt3	17-Jan-24	17-Jan-24

Figure S4. Flowise credentials.

The next step is choosing a pre-defined template from the Flowise Marketplace, shown in

Figure B5. The template used for our application is Conversational Retrieval QA Chain. Thistemplate is based on a method which combines retrieving the most relevant portions of text

from the documents, in combination with chat history. Flowise templates can be edited andadapted for specific purposes.



ChatGPTPlugin
Use ChatGPT Plugins within LangChain abstractions with GET and POST Tools

Claude LLM
Use Anthropic Claude with 200k context window to ingest whole document for QnA

Conversational Agent
A conversational agent for a chat model which utilize chat specific prompts

Conversational Retrieval Agent
Agent optimized for vector retrieval during conversation and answering questions based on previous dialogue.

Conversational Retrieval QA Chain
Text file QnA using conversational retrieval QA chain

HuggingFace LLM Chain
Simple LLM Chain using HuggingFace Inference API on falcon-7b-instruct model

IfElse
Split flows based on if else condition

Image Generation
Generate image using Replicate Stability text-to-image generative AI model

Input Moderation
Detect text that could generate harmful output and prevent it from being sent to the language model

List Output Parser
Return response as a list (array) instead of a string/text

Local QnA
QnA chain using Ollama local LLM, LocalAI embedding model, and Faiss local vector store

Long Term Memory
Use long term memory like Zep to differentiate conversations between users with sessionId

Metadata Filter

Multi Prompt Chain

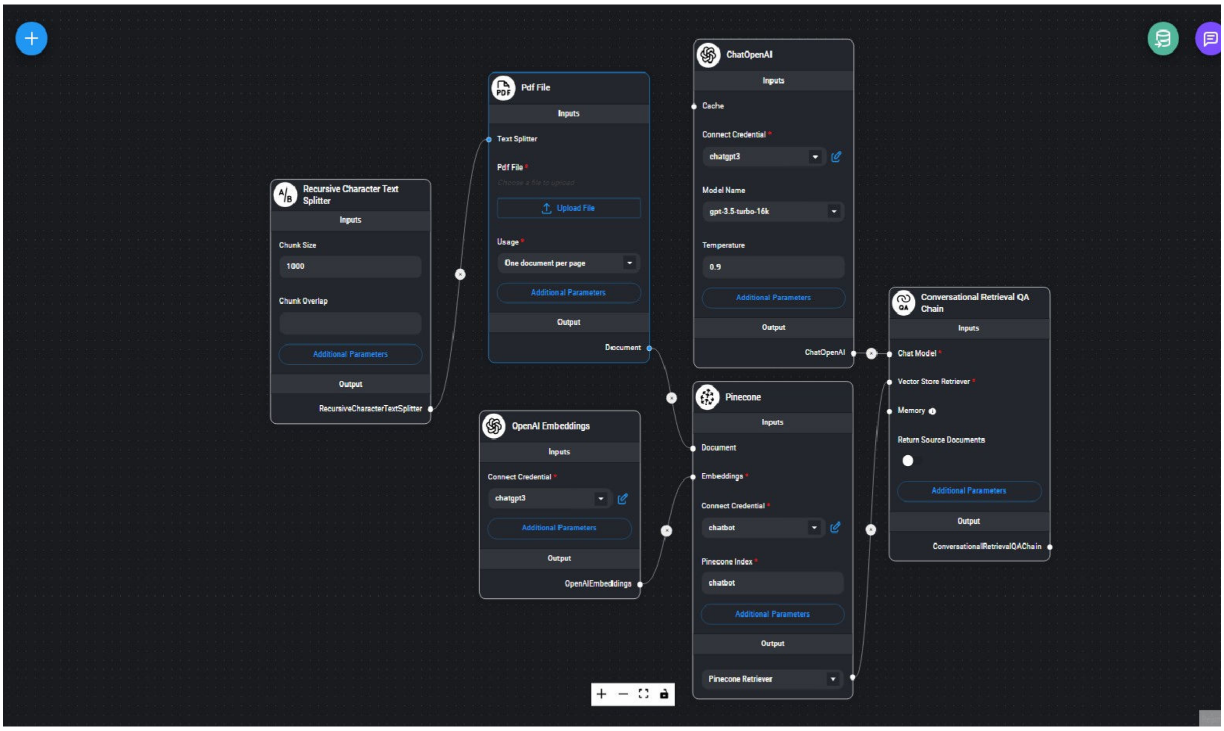
Multi Retrieval QA

Multiple VectorDB

Figure S5. Flowise Marketplace.

In the next step, on the Conversational Retrieval QA Chain template shown in Figure B6, document loader node is added, necessary for uploading the source documents which represent our modified dataset. By selecting Upload File section on that node, documents areadded to Pinecone server. For the purpose of testing this custom template, we chose a repository of required and supplementary course materials listed for the course “Academic Writing” (also used in the initial part of the research), in the format of .pdf files.

Selecting the Connect Credentials button on the OpenAI Embeddings node automatically connects the previously defined credentials for OpenAI and Pinecone.



Recursive Character Text Splitter
Chunk Size
1000
Chunk Overlap

Pdf File
Pdf File *
Choose a file to upload
Usage *
One document per page

ChatOpenAI
Connect Credential *
chatgpt3
Model Name
gpt-3.5-turbo-16k
Temperature
0.9

OpenAI Embeddings
Connect Credential *
chatgpt3

Pinecone
Connect Credential *
chatbot
Pinecone Index *
chatbot
Output
Pinecone Retriever

Conversational Retrieval QA Chain
Return Source Documents
○

Figure S6. Modified version of the Conversational Retrieval QA Chain template.

In the final version of the modified template, unnecessary nodes can be removed, to give a clearer view of the elements. To summarise, the basic elements of the final version are Pinecone server from which documents are retrieved, Open AI model and QA Chain which allows for communication via chatbot interface.