INSTALILY – PARTSELECT BOT – Aayushi Goenka

Ideal Approach:

Part Select must have a database of parts and models in a structured format considering their smooth front-end; this data should be extracted and put into a vector database where it can get indexed.
We should use a RAG pipeline for this approach
After the user submits a query, the pipeline 'RETRIEVES' the relevant documents from the vectorDB, 'AUGMENTS' the query with these documents and sends the new query to the LLM model for 'GENERATING' a response to it

End Result: The chatbot can summarize the structured data in a concise manner and provide all necessary info within the interface.

Current Approach (with certain limitations):

Considerations:
I was not able to scrape the data using a web scraper or a crawler (tried scrapy, bs4, axios, java – 403 error ; have included screenshots at the end) and used a defined json file I obtained online containing partselect data.
Tool choices:

- Flask backend for a lightweight solution with minimal dependencies
- Chroma with OpenAI embeddings to create a vector database
- Use RAG to augment the user_query with relevant context and answer related questions.

Note: I specifically chose to use **OpenAI** for the chat integration instead of DeepSeek because, ideally, the system should be backed by a comprehensive website scraping or a structured database to handle more nuanced queries that require rich contextual understanding. For instance, a question like *"Can you find me something similar to part XYZ?"* relies heavily on semantic similarity and contextual relevance.

DeepSeek, while powerful, doesn't currently support OpenAI embeddings. It works with **OllamaEmbeddings**, which are suited for local setups. However, for a public-facing application like the PartSelect website, a local-only chatbot isn't ideal. OpenAI offers better compatibility and scalability in this context.

- React for its reusable components
- Tailwind for pre-defined styles

Interface:

- Used Figma for initial setup
- Used the theme colors from the PartSelect website and the name “Lily” for InstaLily
- Is responsive and compatible with multiple screen types

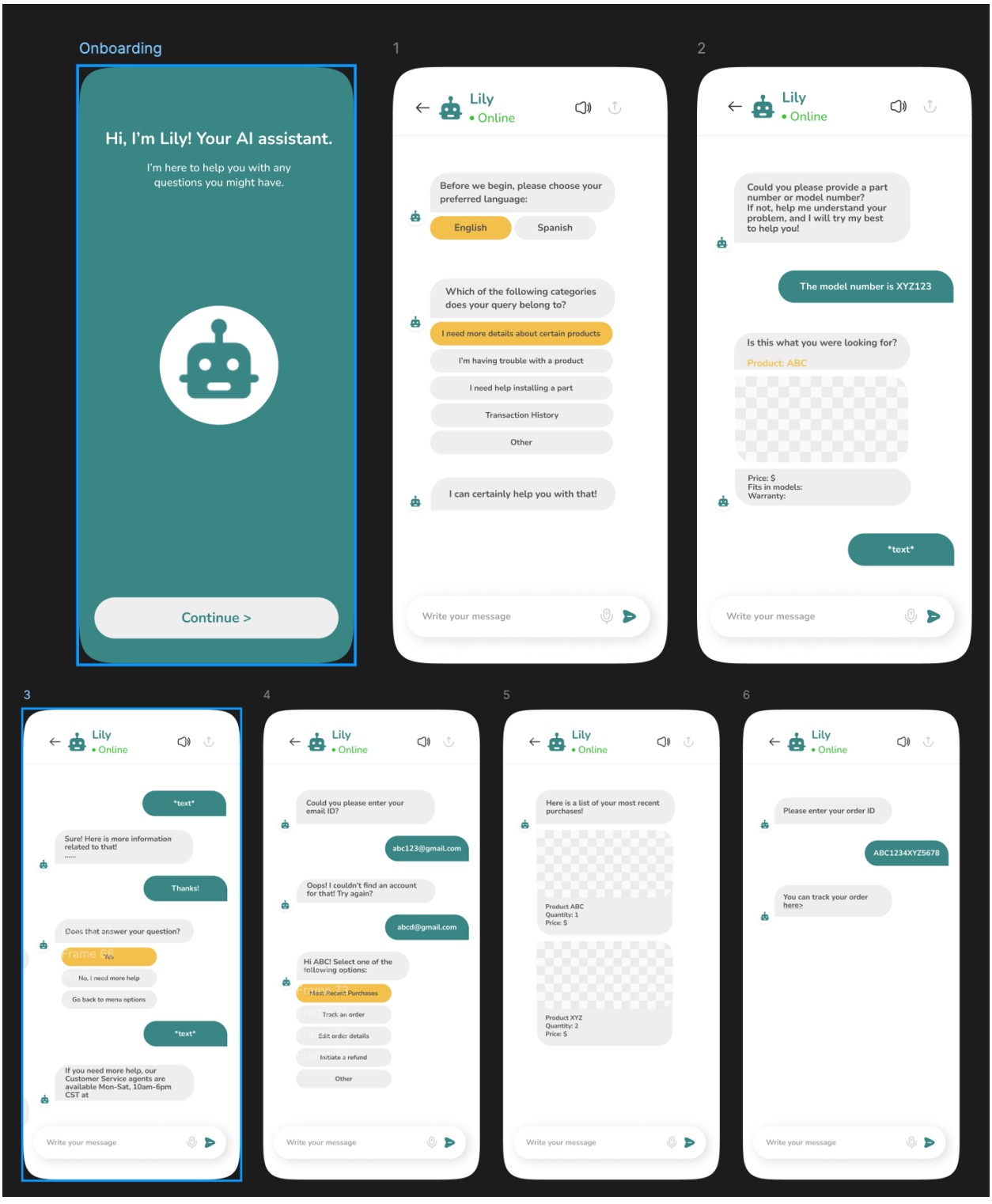
Assumptions:

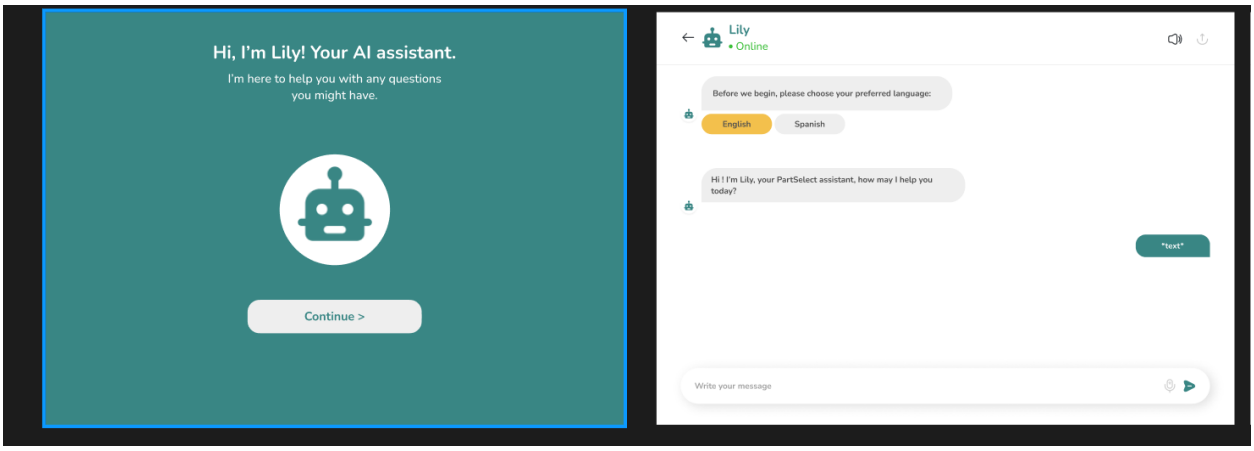
- Only text input from users no images
- Single response from chatbot

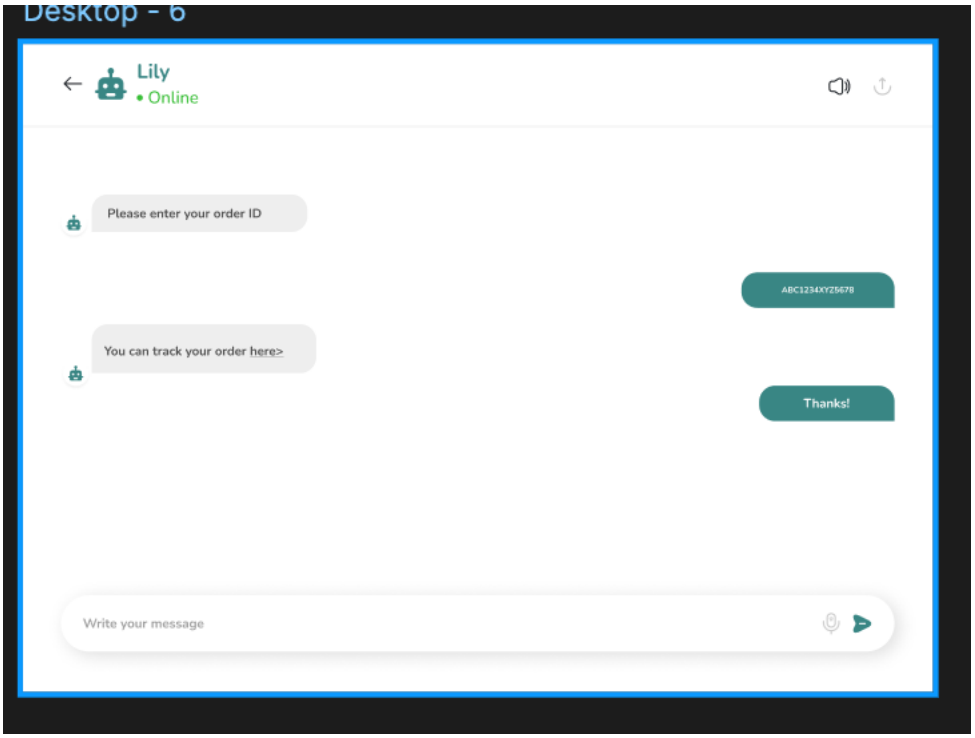
Future considerations (Scalability):

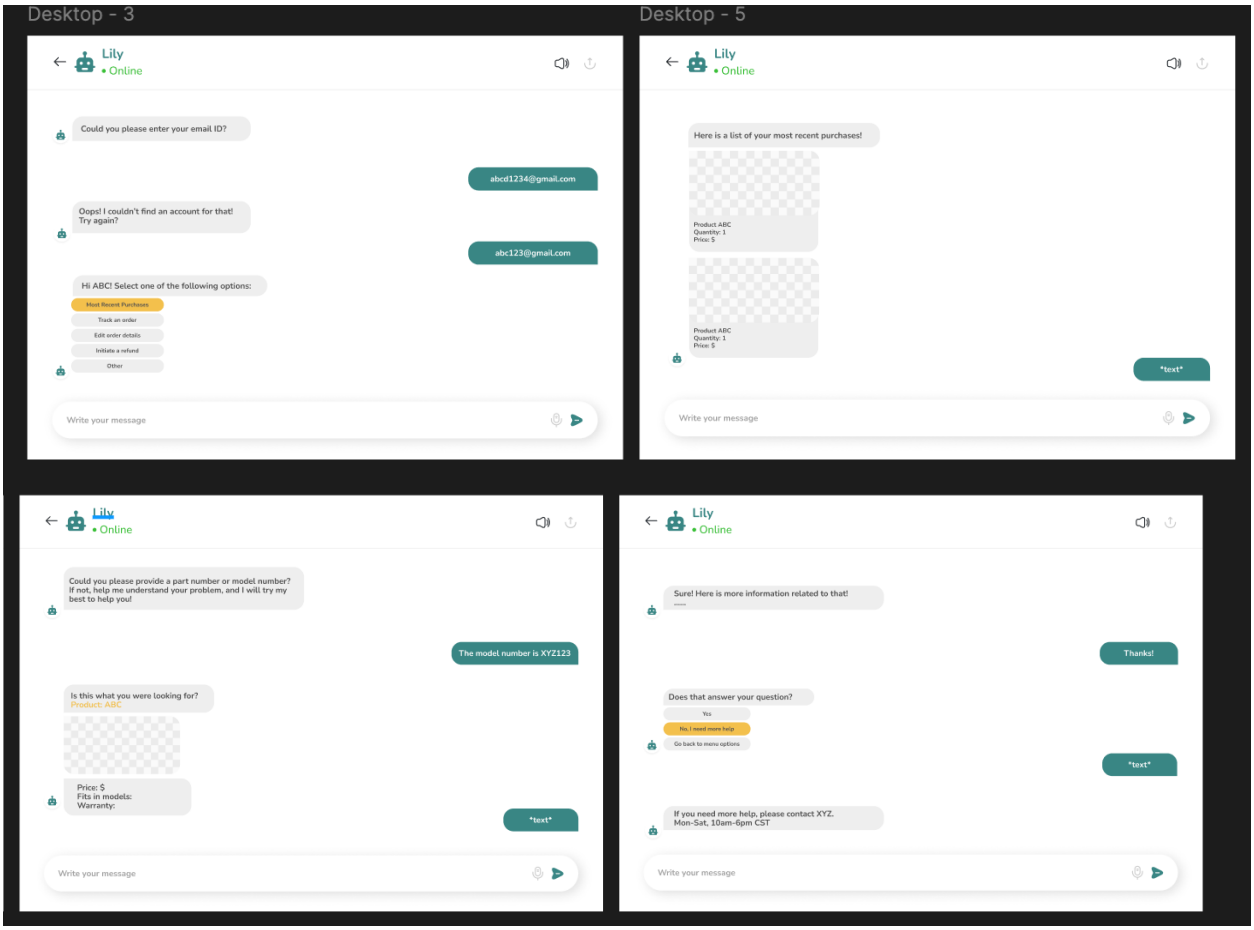
- Disable the input field when the chatbot is sending an answer
- Include image uploads – user can upload an image of the part / model if they don’t know what they’re looking for
- Include a Stripe/Razorpay integration to make the agent place an order on behalf of the user – requires authentication
- Store memory per session/user ID for a more tailored experience in a multi-user environment – use session data to pull up most recent purchases / order tracking
- Include language preferences example: English, Spanish
- To improve accuracy and relevance, it’s important to build **dedicated pipelines for different appliances**, each powered by its own dataset. By identifying the appliance type directly from the user query, we can route the request to the appropriate pipeline — enabling more precise, context-aware responses.
  This approach is conceptually similar to a **Mixture of Experts (MoE) architecture**, where specialized models (or pipelines) are selectively activated based on the input, ensuring better performance and domain-specific understanding.

Interface design (including some future considerations):

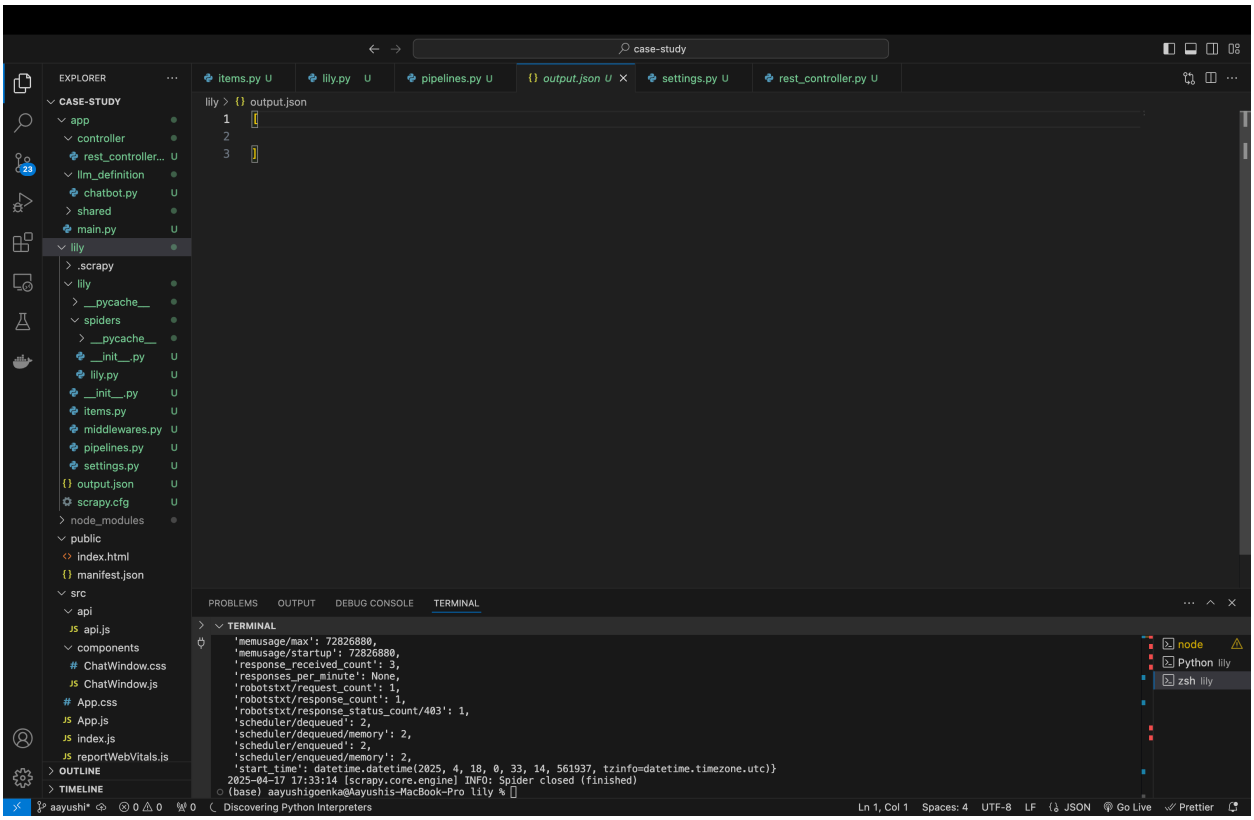

Onboarding
Hi, I'm Lily! Your AI assistant.
I'm here to help you with any questions you might have.
Continue >
1
Lily
• Online
Before we begin, please choose your preferred language:
English
Spanish
Which of the following categories does your query belong to?
I need more details about certain products
I'm having trouble with a product
I need help installing a part
Transaction History
Other
I can certainly help you with that!
Write your message
2
Lily
• Online
Could you please provide a part number or model number? If not, help me understand your problem, and I will try my best to help you!
The model number is XYZ123
Is this what you were looking for?
Product: ABC
Price: $
Fits in models:
Warranty:
text
Write your message
3
Lily
• Online
text
Sure! Here is more information related to that!
Thanks!
Does that answer your question?
No, I need more help
Go back to menu options
text
If you need more help, our Customer Service agents are available Mon-Sat, 10am-6pm CST at
Write your message
4
Lily
• Online
Could you please enter your email ID?
abc123@gmail.com
Oops! I couldn't find an account for that! Try again?
abcd@gmail.com
Hi ABC! Select one of the following options:
Most Recent Purchases
Track an order
Edit order details
Initiate a refund
Other
Write your message
5
Lily
• Online
Here is a list of your most recent purchases!
Product ABC
Quantity: 1
Price: $
Product XYZ
Quantity: 2
Price: $
Write your message
6
Lily
• Online
Please enter your order ID
ABC1234XYZ5678
You can track your order here>
Write your message

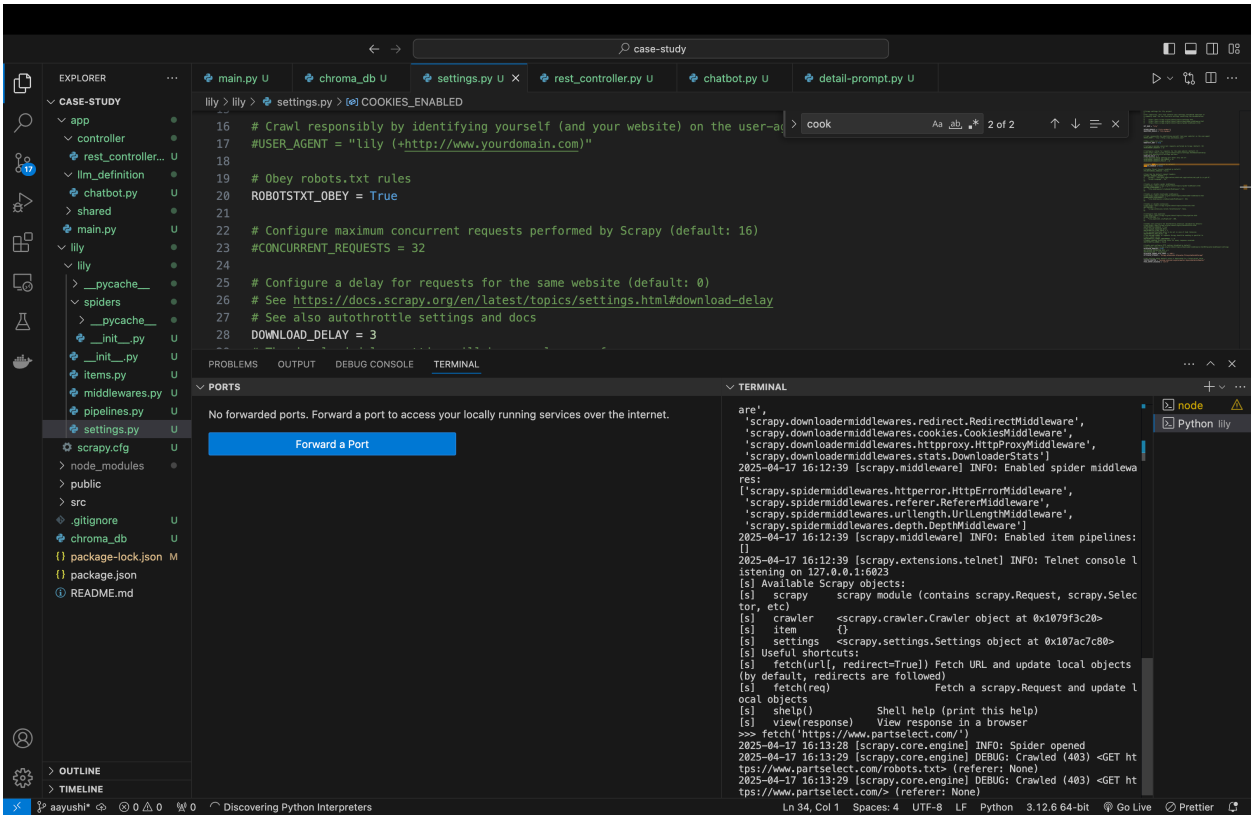

Hi, I'm Lily! Your AI assistant.
I'm here to help you with any questions you might have.
Continue >
Lily
• Online
Before we begin, please choose your preferred language:
English
Spanish
Hi ! I'm Lily, your PartSelect assistant, how may I help you today?
"text"
Write your message


Desktop - 6
Lily
• Online
Please enter your order ID
ABC1234XYZ5678
You can track your order here>
Thanks!
Write your message



SCRAPER SCREENSHOTS:


case-study
EXPLORER
CASE-STUDY
items.py U
lily.py U
pipelines.py U
output.json U
settings.py U
rest_controller.py U
lily > {} output.json
1 [
2
3]
app
controller
rest_controller... U
llm_definition
chatbot.py U
shared
main.py U
lily
.scrapy
lily
__pycache__
spiders
__pycache__
__init__.py U
lily.py U
__init__.py U
items.py U
middlewares.py U
pipelines.py U
settings.py U
output.json U
scrapy.cfg U
node_modules
public
index.html
manifest.json
src
api
api.js
components
ChatWindow.css
ChatWindow.js
App.css
App.js
index.js
reportWebVitals.js
OUTLINE
TIMELINE
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
TERMINAL
'memusage/max': 72826880,
'memusage/startup': 72826880,
'response_received_count': 3,
'responses_per_minute': None,
'robotstxt/request_count': 1,
'robotstxt/response_count': 1,
'robotstxt/response_status_count/403': 1,
'scheduler/dequeued': 2,
'scheduler/dequeued/memory': 2,
'scheduler/enqueued': 2,
'scheduler/enqueued/memory': 2,
'start_time': datetime.datetime(2025, 4, 18, 0, 33, 14, 561937, tzinfo=datetime.timezone.utc)}
2025-04-17 17:33:14 [scrapy.core.engine] INFO: Spider closed (finished)
(base) aayushigoenka@Aayushis-MacBook-Pro lily %
node
Python lily
zsh lily
aayushi*
Discovering Python Interpreters
Ln 1, Col 1
Spaces: 4
UTF-8
LF
{} JSON
Go Live
Prettier


case-study
EXPLORER
CASE-STUDY
main.py U
chroma_db U
settings.py U
rest_controller.py U
chatbot.py U
detail-prompt.py U
lily > lily > settings.py > COOKIES_ENABLED
cook
2 of 2
16 # Crawl responsibly by identifying yourself (and your website) on the user-a
17 #USER_AGENT = "lily (+http://www.yourdomain.com)"
18
19 # Obey robots.txt rules
20 ROBOTSTXT_OBEY = True
21
22 # Configure maximum concurrent requests performed by Scrapy (default: 16)
23 #CONCURRENT_REQUESTS = 32
24
25 # Configure a delay for requests for the same website (default: 0)
26 # See https://docs.scrapy.org/en/latest/topics/settings.html#download-delay
27 # See also autothrottle settings and docs
28 DOWNLOAD_DELAY = 3
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
No forwarded ports. Forward a port to access your locally running services over the internet.
Forward a Port
TERMINAL
are',
'scrapy.downloadermiddlewares.redirect.RedirectMiddleware',
'scrapy.downloadermiddlewares.cookies.CookiesMiddleware',
'scrapy.downloadermiddlewares.httpproxy.HttpProxyMiddleware',
'scrapy.downloadermiddlewares.stats.DownloaderStats']
2025-04-17 16:12:39 [scrapy.middleware] INFO: Enabled spider middlewares:
['scrapy.spidermiddlewares.httperror.HttpErrorMiddleware',
'scrapy.spidermiddlewares.referer.RefererMiddleware',
'scrapy.spidermiddlewares.urllength.UrlLengthMiddleware',
'scrapy.spidermiddlewares.depth.DepthMiddleware']
2025-04-17 16:12:39 [scrapy.middleware] INFO: Enabled item pipelines:
[]
2025-04-17 16:12:39 [scrapy.extensions.telnet] INFO: Telnet console listening on 127.0.0.1:6023
[s] Available Scrapy objects:
[s] scrapy scrapy module (contains scrapy.Request, scrapy.Selector, etc)
[s] crawler <scrapy.crawler.Crawler object at 0x1079f3c20>
[s] item {}
[s] settings <scrapy.settings.Settings object at 0x107ac7c80>
[s] Useful shortcuts:
[s] fetch(url[, redirect=True]) Fetch URL and update local objects (by default, redirects are followed)
[s] fetch(req) Fetch a scrapy.Request and update local objects
[s] shelp() Shell help (print this help)
[s] view(response) View response in a browser
>>> fetch('https://www.partselect.com/')
2025-04-17 16:13:28 [scrapy.core.engine] INFO: Spider opened
2025-04-17 16:13:29 [scrapy.core.engine] DEBUG: Crawled (403) <GET https://www.partselect.com/robots.txt> (referer: None)
2025-04-17 16:13:29 [scrapy.core.engine] DEBUG: Crawled (403) <GET https://www.partselect.com/> (referer: None)
node
Python lily
.gitignore
chroma_db
package-lock.json M
package.json
README.md
OUTLINE
TIMELINE
aayushi*
Discovering Python Interpreters
Ln 34, Col 1
Spaces: 4
UTF-8
LF
Python
3.12.6 64-bit
Go Live
Prettier