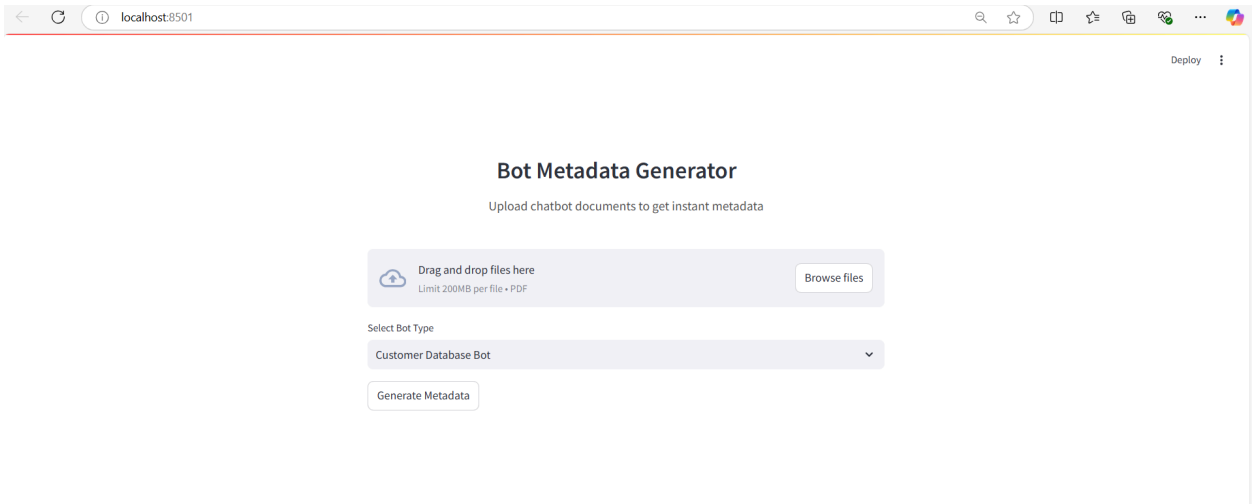
Auto Metadata Generation Web Tool Documentation

Overview

The Metadata Web App Tool is a user-friendly application that allows you to upload PDF documents containing chatbot-related information and automatically generates aggregated metadata for specific chatbot types. The generated metadata includes capabilities, descriptions, and specialization keywords, which can be downloaded in JSON format.

How to Use the Metadata Web App Tool

Step 1: Accessing the Tool

1. Open the web app in your browser by running the command “streamlit run python_webapp.py”
2. You will see the main interface with a clean and minimal design, featuring an upload area, bot type selector, and a "Generate Metadata" button.
3. 

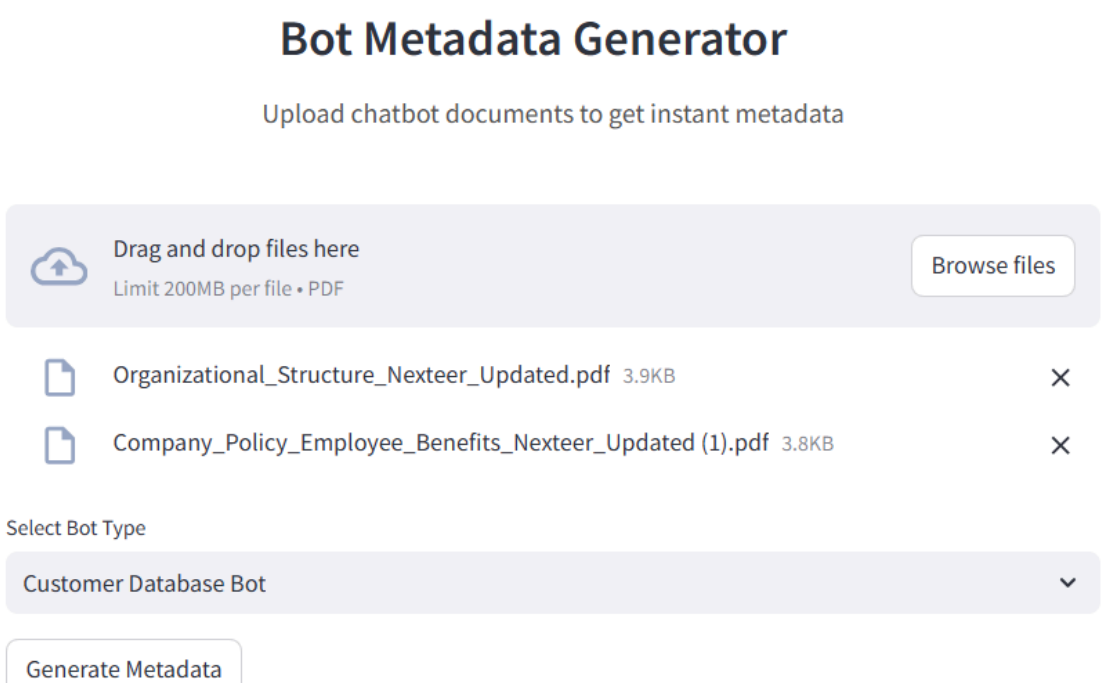
Step 2: Uploading Documents

1. In the "Upload PDF files" section:
    - Drag and drop one or more PDF files into the upload box.
    - Alternatively, click the Browse files button to select files from your computer.
2. Ensure that the uploaded documents are in the PDF format and contain relevant content about the chatbot.

Step 3: Selecting a Bot Type

1. Use the dropdown menu under "Select Bot Type" to choose the type of chatbot for which you want to generate metadata. The options include:
    - Customer Database Bot
    - Organizational Information Bot
    - Internet Search Bot
    - Custom Bot

2. If you select Custom Bot, a text input field will appear for you to manually enter the bot's name.



Step 4: Generating Metadata

1. Click the Generate Metadata button to process the uploaded documents and generate metadata.
2. The app will extract text from the uploaded PDFs and call an AI model to generate metadata fields:
    - Capabilities: Lists specific functionalities the bot should have.
    - Description: Summarizes the bot's purpose and scope.
    - Specialization Keywords: Extracts keywords describing the bot's expertise and focus.

Step 5: Viewing and Downloading Metadata

1. After processing is complete, the metadata will be displayed on the screen in a JSON format.
2. To download the metadata:
    - Click the Download Metadata button.
    - Save the file as a .json file on your computer. The file will be named based on the bot type, e.g., Customer_Database_Bot_metadata.json.

```
"Specialization Keywords" :
"company policies, employee benefits, working hours, attendance policy, paid time off, sick leave, parental leave, bereavement leave, leave of absence, remote work policy, code of conduct, health insurance, retirement plans, savings plans, professional development, wellness programs, life insurance, disability insurance, employee assistance program, organizational structure, executive leadership, manufacturing, logistics, supply chain management, product development, engineering, quality assurance, human resources, reporting structure, employee engagement, flexible work hours, PTO accrual, hybrid work model, counseling services, tuition reimbursement, wellness credit, fitness center, financial planning, mental health support, family counseling."
}
```


Download Metadata

API Documentation for Auto Metadata Generation

API Documentation for Metadata Generation API (Future Extension)

Overview

This documentation outlines how the Metadata Web App functionality can be extended to support an API-based workflow. The envisioned API will allow developers to upload multiple documents, specify a chatbot type, and receive aggregated metadata programmatically. While this functionality is not currently implemented, the following serves as a guide for how it can be integrated into the existing system.

Proposed Base URL:https://api.example.com/v1/metadata

Proposed Endpoints

1. Generate Metadata: POST /generate

This endpoint will allow users to send multiple documents and a chatbot type as input and receive metadata as the output.

Request Format

Headers:

- Content-Type: application/json
- Authorization: Bearer YOUR_API_KEY

Body (JSON):

```
{
  "documents": [
    "base64_encoded_pdf_1",
    "base64_encoded_pdf_2"
  ],
  "chatbot_type": "Customer Database Bot"
}
```

Parameter	Type	Required	Description
documents	Array (String)	Yes	An array of Base64-encoded PDF documents.
chatbot_type	String	Yes	The type of chatbot. Options: Customer Database Bot, Organizational Information Bot, Internet Search Bot, or a custom name.

Response Format

Success (200):


```
{
  "status": "success",
  "metadata": {
    "Capabilities": "Handles customer queries, fetches purchase history, updates customer profiles.",
    "Description": "This bot assists in managing customer data and streamlining customer support processes.",
    "Specialization Keywords": "customer, database, queries, purchase history, customer support"
  }
}
```


Field	Type	Description
Capabilities	String	A list of the bot's specific functionalities derived from the documents.
Description	String	A summary of the bot's purpose and scope.
Specialization Keywords	String	Comma-separated keywords describing the bot's expertise.

Error (400):


```
{
  "status": "error",
  "message": "Invalid input. Please provide valid Base64-encoded documents and chatbot type."
}
```

How It Can Be Extended

Step 1: Modify the Code to Accept API Requests

- Use a framework like Flask or FastAPI to handle HTTP POST requests.
- Add a route, e.g., /generate, that accepts JSON payloads containing Base64-encoded PDFs and chatbot type.

Step 2: Decode Base64-Encoded PDFs

- Extract and decode the Base64 strings into PDF files.

Example:

```
import base64

def decode_base64_to_pdf(base64_string, output_path):
    with open(output_path, "wb") as pdf_file:
        pdf_file.write(base64.b64decode(base64_string))
```

Step 3: Integrate with Existing Text Extraction and Metadata Generation Logic

- Use the extract_text_from_pdfs function to process the decoded PDFs.

  Code Link: https://github.com/Rugz007/capstone/blob/main/python_webapp.py#L19

- Pass the combined text and chatbot type to generate_aggregate_metadata.

  Code Link: https://github.com/Rugz007/capstone/blob/main/python_webapp.py#L32

Step 4: Return Metadata as JSON

- Modify the API to return metadata as a JSON response, as shown in the response format.