Semantic Contextual PDF Retrieval (SCPR)

User Manual

Abstract

This manual provides a complete guide to the installation, setup, and operation of the Semantic Contextual PDF Retrieval (SCPR) application. It is intended for new users and system administrators. The guide covers all prerequisites, the setup of both the backend and frontend components, and basic troubleshooting steps.

Contents

1 Introduction

1.1 What is SCPR?

Welcome to the Semantic Contextual PDF Retrieval (SCPR) application. SCPR is a powerful, full-stack, AI-powered tool that allows you to ”chat” with your PDF documents.

Instead of manually searching through hundreds of pages, you can simply upload a PDF and ask questions in plain English. The application uses a sophisticated Retrieval-Augmented Generation (RAG) pipeline to find the most relevant information within your document and generate a concise, accurate answer.

1.2 How It Works

The application operates in two main stages:

1. **Indexing (Upload):** When you upload a PDF, the backend server:
   - Extracts all text, using OCR (Optical Character Recognition) for scanned images.
   - Splits the text into small, semantically meaningful chunks.
   - Converts these chunks into vector embeddings (numerical representations) using a Google AI model.
   - Stores these vectors in a local ‘ChromaDB‘ database.
2. **Querying (Chat):** When you ask a question:
   - Your question is also converted into a vector.
   - The system searches the database to find the text chunks that are most semantically similar to your question.
   - These relevant chunks (the ”context”) are passed, along with your original question, to a Google Gemini chat model.
   - The model generates a new answer based *only* on the provided context, ensuring all information is sourced directly from your document.

1.3 Who is this Manual For?

This manual is intended for end-users, system administrators, or developers who wish to run the SCPR application in a local environment for personal use or for demonstration purposes.

2 System Requirements (Prerequisites)

Before proceeding with the installation, you must ensure your system meets the following requirements.

- **Python 3.9 or newer.**
- **Node.js 18 or newer** (for the React frontend).
- **Git** (for cloning the project repository).
- **Tesseract OCR Engine:** This is a critical system-level dependency that must be installed separately from the Python packages. The application will fail to start without it.

- **Windows:** Download and run the installer from the official [Tesseract GitHub page](https://github.com/UB-Mannheim/tesseract/wiki).
- **macOS (using Homebrew):** `brew install tesseract`
- **Linux (Debian/Ubuntu):** `sudo apt-get install tesseract-ocr`

3 Installation & Setup

This section guides you through the complete setup for both the backend and frontend.

3.1 Step 1: Clone the Repository

First, open your terminal, navigate to the directory where you want to store the project, and clone the repository.

```
# Replace with your actual repository URL
git clone https://github.com/your-username/scpr-project.git
cd scpr-project
```

This will create a main project folder (e.g., `scpr-project`) containing the backend and `frontend` sub-folders.

3.2 Step 2: Backend Setup (Flask Server)

All backend commands should be run from the backend directory.

1. **Navigate to the backend folder:**

   ```
   cd backend
   ```

2. **Create and activate a Python virtual environment:**

   ```
   # On Windows
   python -m venv venv
   .\venv\Scripts\activate

   # On macOS or Linux
   python3 -m venv venv
   source venv/bin/activate
   ```

3. **Install all required Python dependencies:**

   ```
   pip install -r requirements.txt
   ```

4. **Create the environment file:** Create a new file named .env in the backend folder. You must obtain an API key from [Google AI Studio](https://aistudio.google.com/app/apikey).

   Add your key to the .env file as shown below:

   ```
   GEMINI_API_KEY=your_google_api_key_here
   ```

3.3 Step 3: Frontend Setup (React App)

All frontend commands should be run from the `frontend` directory.

1. **Navigate to the frontend folder (from the root project directory):**

   ```
   # From the root folder (e.g., scpr-project)
   cd frontend
   ```

2. **Install all Node.js dependencies:**

   ```
   npm install
   ```

3. **(Optional) Create environment file:** For robust communication with the backend, create a file named `.env.local` in the `frontend` folder and add the following line:

   ```
   REACT_APP_API_BASE_URL=http://127.0.0.1:5000
   ```

4 Running the Application

To run the application, you must have **two separate terminals** open simultaneously.

4.1 Terminal 1: Run the Backend (Flask)

1. Open a terminal and navigate to the backend folder.
2. Activate your virtual environment (if not already active).
3. Run the application.

```
cd path/to/your-project/backend
.\venv\Scripts\activate  (Or source venv/bin/activate)
python app.py
```

You should see output indicating the server is running on http://127.0.0.1:5000.

4.2 Terminal 2: Run the Frontend (React)

1. Open a **new** terminal and navigate to the `frontend` folder.
2. Run the start script.

```
cd path/to/your-project/frontend
npm start
```

This command will automatically open the SCPR application in your default web browser at http://localhost:3000.

5 How to Use SCPR

1. **Access the Application:** Open your web browser and navigate to http://localhost:3000.
2. **Upload a PDF:** Click the "Upload" or "Browse" button and select a PDF file from your computer. The system will begin indexing the file. You will see a success message when the document is ready.
3. **Ask Questions:** Type your question into the chat input box at the bottom of the screen and press Enter. The AI will retrieve the relevant context from your document and generate an answer.

6 Testing the Backend (Optional)

If you are a developer, you can verify the backend's functionality by running the included test suite.

1. Open a terminal in the `backend` directory.
2. Activate your virtual environment.
3. Install the development dependencies:

   ```
   pip install -r requirements-dev.txt
   ```

4. Run `pytest`:

   ```
   pytest -v
   ```

   All tests should pass, confirming the API is working correctly.

7 Troubleshooting

- **Error: Tesseract not found or "No text found in PDF."**
  This is the most common error. It means the Tesseract OCR engine is not installed correctly or is not in your system's PATH. Please reinstall Tesseract using the instructions in Section 2 and restart your computer.
- **Error: 500 Server Error on upload.**
  Check your backend terminal. This often means your `GEMINI_API_KEY` in the `.env` file is missing or invalid.
- **Error: "Connection Refused" or "Network Error" in the browser.**
  This means your backend (Flask) server is not running. Make sure your backend terminal (Terminal 1) is open and running `python app.py` without errors.