Documentation for Resume Parser

Overview

The Resume Parser project is a Flask-based web application designed to parse resumes in PDF or DOCX format. It extracts text, identifies named entities using a custom NER model, and stores the extracted data in a MongoDB database. Additionally, it utilizes Cloudinary for file storage.

Features

- **Upload Resumes**: Supports PDF and DOCX file formats.
- **Text Extraction**: Extracts text from uploaded files.
- **Named Entity Recognition (NER)**: Processes text with a SpaCy NER model to identify entities like names, organizations, etc.
- **Data Storage**: Stores extracted text and recognized entities in a MongoDB collection.
- **Cloud Storage**: Uses Cloudinary for secure file storage.

Setup and Installation

Prerequisites

Ensure the following are installed:

- Python 3.7 or later
- MongoDB
- Cloudinary account
- Libraries listed in `requirements.txt`

Steps

Clone the Repository
git clone https://github.com/Rsoniie/Resume_Parser.git

1. cd Resume_Parser

Set Up Virtual Environment
python -m venv venv

2. source venv/bin/activate # On Windows: venv\Scripts\activate
3. **Install Dependencies**
   pip install -r requirements.txt

Configure Environment Variables Create a `.env` file and add:
CLOUDINARY_CLOUD_NAME=<your_cloudinary_name>
CLOUDINARY_API_KEY=<your_cloudinary_api_key>
CLOUDINARY_API_SECRET=<your_cloudinary_api_secret>

4. **Prepare the NER Model**
   - Unzip `nlp_ner_model.zip` into the project directory.
   - Ensure `app.py` references the correct model directory:
     nlp = spacy.load('nlp_ner_model')
5. **Run the Application**
   python app.py
   The app will run at: `http://127.0.0.1:3000`

Endpoints

1. `/` (GET)

- Renders the homepage.

2. `/upload` (POST)

- Accepts file uploads (PDF/DOCX).

Returns:
```
{
  "message": "File uploaded successfully.",
  "filename": "example.pdf",
  "file_url": "https://cloudinary.com/..."

}
```

3. `/extract_text` (POST)

- Extracts text and performs NER on the last uploaded file.

Returns:
```
{
  "message": "Text extraction and entity recognition were successful.",
  "text": "Extracted text here...",
  "entities": [
    {"entity": "John Doe", "label": "PERSON"},
    {"entity": "Google", "label": "ORG"}
  ],
  "document_id": "mongodb_document_id"

}
```

File Structure

```
Resume_Parser/
├── .env
├── .gitignore
├── app.py
├── requirements.txt
├── templates/
│   └── index.html
├── nlp_ner_model/
├── static/
│   └── css/
├── uploads/
```

Key Files

1. `app.py`

- Core application logic.
- Handles routes for file upload and text extraction.

2. `.env`

- Contains environment variables for Cloudinary and MongoDB configuration.

3. `requirements.txt`

Lists all Python dependencies required for the project:
Flask==2.2.3

```
python-dotenv==1.0.0
cloudinary==1.30.0
spacy==3.5.0
PyPDF2==3.0.1
python-docx==0.8.1
pymongo==4.5.0
```

4. `README.md`

- General project overview and setup instructions.

5. `nlp_ner_model/`

- Directory containing the custom SpaCy NER model.

Troubleshooting

1. **MongoDB Issues**:
   - Ensure MongoDB server is running.
   - Verify `MONGODB_URI` in `.env`.
2. **Cloudinary Upload Errors**:
   - Check API credentials in `.env`.
3. **NER Model Loading Errors**:
   - Confirm `nlp_ner_model` directory exists with the necessary files.

Author

- **Roshan Soni**

License

- This project is licensed under the MIT License.