Resume Tailor Agent

This tool tailors your resume to specific job descriptions while keeping the **original formatting** intact.

It works locally (free, with Ollama) or with APIs (OpenAI, Anthropic, etc.).

🚀 Features

- Reads your resume in `.docx` format
- Reads job description from `.txt` or input string
- Calls an LLM (local or API) to rewrite **summary, skills, and experiences**
- Outputs a new `.docx` resume with identical formatting
- **Web interface** with Streamlit for easy use
- **CLI interface** for automation and scripting

📂 Project Structure

Plain Text

```
resume-tailor-agent/
|── data/
|   ├── resume_template.docx   # your base resume (sample included)
|   ├── jd.txt                 # job description (sample included)
|
|── output/
|   └── resume_tailored.docx   # generated tailored resume
|
|── src/
|   ├── __init__.py
|   ├── main.py                # CLI entry script
|   ├── resume_parser.py       # extract/update resume sections
|   ├── llm_interface.py       # wrapper for local/remote LLMs
|   └── utils.py               # helper functions
|
|── streamlit_app.py           # web interface
|── test_parser.py             # test resume parsing
|── test_llm.py                # test LLM integration
|── requirements.txt
|── README.md
```

🛠️ Installation

Prerequisites

- Python 3.8 or higher
- pip package manager

Quick Setup

1. Clone or download this project:
2. Create virtual environment:
3. Install requirements:

🧠 LLM Options

You can run this tool with either:

Option 1: Free Local (Recommended)

- Install Ollama (works great on Mac/Linux)
- Pull a model:
- This runs **completely free on your machine**

Option 2: Paid APIs

- **OpenAI** (GPT-4, GPT-3.5-turbo)
- **Anthropic** (Claude 3.5 Sonnet, Claude 3 Haiku)

Set your API key as an environment variable:

```bash
export OPENAI_API_KEY="your_key_here"
export ANTHROPIC_API_KEY="your_key_here"
```

▶️ Usage

Web Interface (Recommended)

Start the Streamlit web app:

Bash

```
streamlit run streamlit_app.py
```

Then open your browser to `http://localhost:8501` and:

1. Upload your resume (.docx)
2. Paste or upload job description
3. Select your LLM model
4. Click "Tailor My Resume"
5. Download your tailored resume

Command Line Interface

Run the tool from the project root:

```bash
python src/main.py --resume data/resume_template.docx --jd data/jd.txt --out
output/resume_tailored.docx --model local
```

Arguments:

- `--resume` / `-r` : Path to your resume (.docx)
- `--jd` / `-j` : Path to job description (.txt)
- `--jd-text` : Job description as text string (alternative to --jd)
- `--out` / `-o` : Output path for tailored resume
- `--model` / `-m` : LLM model (`local` , `openai` , `anthropic`)
- `--verbose` / `-v` : Enable detailed logging
- `--test-llm` : Test LLM connection

Examples:

```bash
# Using local Ollama
python src/main.py -r data/resume.docx -j data/job.txt -o
output/tailored.docx -m local

# Using OpenAI
python src/main.py -r data/resume.docx -j data/job.txt -o
output/tailored.docx -m openai
```

```
# Using job description as text
python src/main.py -r data/resume.docx --jd-text "Software Engineer
position..." -o output/tailored.docx
```

🧪 Testing

Test Resume Parsing

Bash

```bash
python test_parser.py
```

Test LLM Integration

Bash

```bash
python test_llm.py
```

Test LLM Connection

Bash

```bash
python src/main.py --test-llm --model local
```

🔧 Configuration

Environment Variables

- `OPENAI_API_KEY` : Your OpenAI API key
- `ANTHROPIC_API_KEY` : Your Anthropic API key

Supported File Formats

- **Input Resume**: `.docx` (Microsoft Word)
- **Job Description**: `.txt` (plain text) or direct text input
- **Output Resume**: `.docx` (preserves original formatting)

🛠️ Development

Project Architecture

- **`resume_parser.py`** : Handles .docx file parsing and updating
- **`llm_interface.py`** : Unified interface for different LLM providers
- **`utils.py`** : Helper functions and utilities
- **`main.py`** : CLI orchestration
- **`streamlit_app.py`** : Web interface

Adding New LLM Providers

1. Add provider configuration to `llm_interface.py`
2. Implement provider-specific method (e.g., `_run_newprovider`)
3. Update `supported_models` dictionary
4. Add to CLI choices in `main.py`

Extending Functionality

The modular design makes it easy to:

- Add new resume sections
- Support additional file formats
- Implement new LLM providers
- Add more sophisticated prompt engineering

🚀 Roadmap

- [x] ~~Basic tailoring (summary + skills + experience)~~
- [x] ~~CLI interface with comprehensive options~~
- [x] ~~Streamlit web interface~~
- [x] ~~Multiple LLM provider support~~
- [ ] Cover letter generator
- [ ] Job application tracker
- [ ] Batch processing for multiple jobs
- [ ] Resume templates and themes
- [ ] Advanced prompt customization

🐛 Troubleshooting

Common Issues

"Could not connect to Ollama"

- Make sure Ollama is installed and running
- Run `ollama serve` to start the service
- Pull a model: `ollama pull mistral`

"OpenAI client not initialized"

- Set your API key: `export OPENAI_API_KEY="your_key"`
- Check your API key is valid
- Ensure you have sufficient credits

"Failed to load resume document"

- Ensure file is in `.docx` format (not `.doc`)
- Check file is not corrupted
- Verify file path is correct

"No content sections found in resume"

- Ensure resume has clear section headers (Summary, Skills, Experience)
- Check that sections contain text content
- Review the sample resume format

Getting Help

1. Check the logs with `--verbose` flag
2. Test individual components with test scripts
3. Verify your LLM connection with `--test-llm`
4. Review the sample files in `data/` directory

📄 License

🤝 Contributing

Contributions are welcome! Please:

1. Fork the repository
2. Create a feature branch
3. Make your changes
4. Add tests if applicable
5. Submit a pull request

✅ With this setup, you have a **clean modular project** that:

- Works **free on your local machine**
- Can be **extended into a paid SaaS product**
- Lets you **swap LLMs with one line change**
- Provides both **CLI and web interfaces**
- Is **ready for production deployment**