Relatório 11 - Prática: Criando agentes com CrewAI e Streamlit (IV)

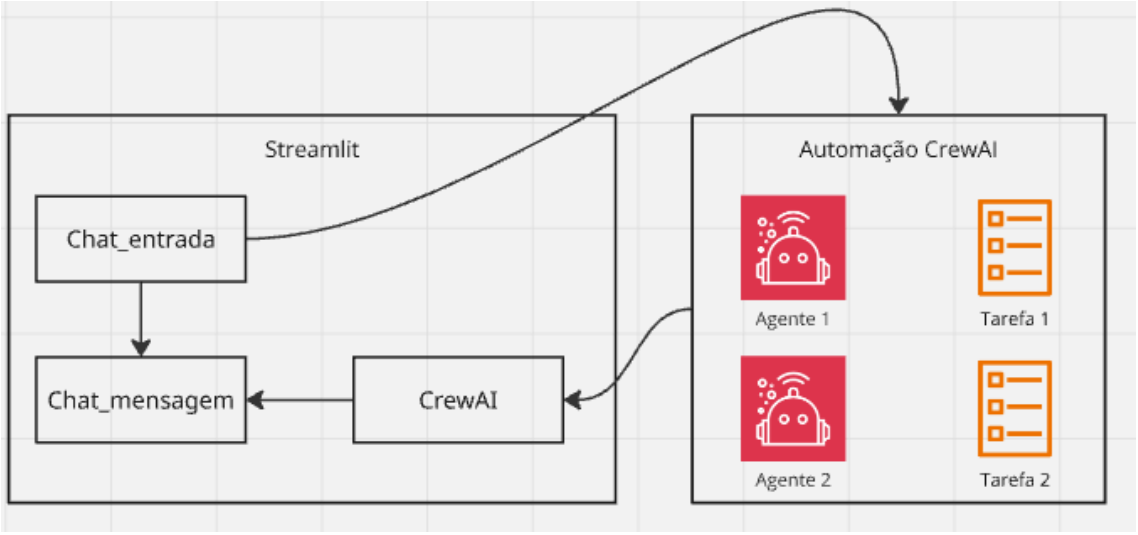
<Paulo Victor Dos Santos>

Descrição da atividade

A atividade do card 11 é uma atividade prática voltada a usar agentes com a ferramenta streamlit.

O streamlit facilita na criação de aplicativos web através da linguagem python, é possível criar prótotipos web programando em python, sem a necessidade de técnicas avançadas de frontend.

O primeiro vídeo demonstra de forma simples como utilizar o streamlit com agentes inteligentes, trata-se de uma aula prática com um caso de uso chamado “CrewAI Writing Studio”. Neste caso de uso são utilizados dois agentes, um é escrito de texto para blog e o outro agente é o revisor do texto escrito.



O fluxo acima exibe a interação do aplicativo. A parte de automação consta com dois agentes e a resposta final é exibida de forma sequencial na interface criada pelo streamlit.

O segundo vídeo do card 11, exibe um aplicativo de planejamento de férias, chamado VacAIgent. Neste aplicativo o usuário seleciona a cidade em que se encontra, a cidade de destino no qual tem interesse em viajar, as datas de interesse e um campo textual no qual pode inserir mais detalhes sobre as supostas férias.

Trata-se de um código que utiliza agentes de forma hierárquico, isso facilita a definição do que os agentes precisam de fato executar, trazendo velocidade nas respostas. Ou seja, o usuário pré selecionar alguns dados cruciais, pode fazer com que a aplicação seja mais rápida, uma vez que os agentes já possuem bastante informação, facilitando na tomada de decisão.

O último vídeo é uma solução interessante e bastante útil. Esta solução tem como escopo a pesquisa por profissionais a partir da descrição de uma vaga ou da capacitação do suporto profissional da área de tecnologia no qual deseja contratar.
O sistema pode fazer buscas no linkedin ou na internet para procurar o profissional mais adequado.

O último vídeo do card 11 apresenta conceitos de utilização de containers com agentes. Além da introdução aos containers, a aula demonstra como utilizar a biblioteca FASTAPI, o que facilita a integração e interoperabilidade com outros sistemas através de REST.

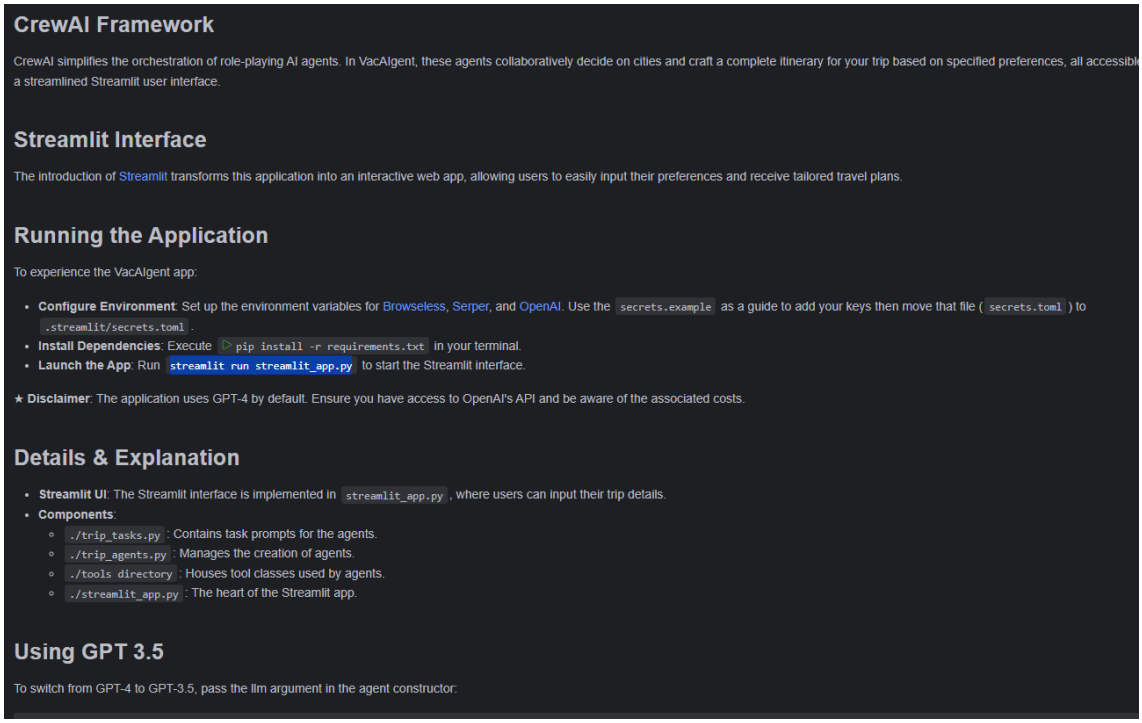
A prática aplicada no card 11 toma como base o sistema inicial apresentado que foi personalizado para atuar como um sistema de diagnóstico e laudos médicos.

A ideia é que o usuário possa descrever alguns sintomas e o sistema possa dar possíveis indicações de diagnóstico ou laudos, como se trata de um experimento a ideia é observar o comportamento dos agentes e suas interações na área médica.

O intuito do teste realizado é a preparação para o trabalho final do Fastcamp, com a experiência adquirida com sistemas distribuídos, containers, FastApi e Streamlit, eu possa utilizar todas essas tecnologias para criar uma ferramenta útil com toda abrangência possível.

Dificuldades

O código do segundo vídeo está disponível em um repositório na descrição do vídeo. Fiz o clone do repositório e segui as recomendações do arquivo README, conforme a imagem abaixo.



CrewAI Framework

CrewAI simplifies the orchestration of role-playing AI agents. In VacAIgent, these agents collaboratively decide on cities and craft a complete itinerary for your trip based on specified preferences, all accessible a streamlined Streamlit user interface.

Streamlit Interface

The introduction of Streamlit transforms this application into an interactive web app, allowing users to easily input their preferences and receive tailored travel plans.

Running the Application

To experience the VacAIgent app:

- **Configure Environment**: Set up the environment variables for Browseless, Serper, and OpenAI. Use the `secrets.example` as a guide to add your keys then move that file (`secrets.toml`) to `.streamlit/secrets.toml` .
- **Install Dependencies**: Execute `pip install -r requirements.txt` in your terminal.
- **Launch the App**: Run `streamlit run streamlit_app.py` to start the Streamlit interface.

★ **Disclaimer**: The application uses GPT-4 by default. Ensure you have access to OpenAI's API and be aware of the associated costs.

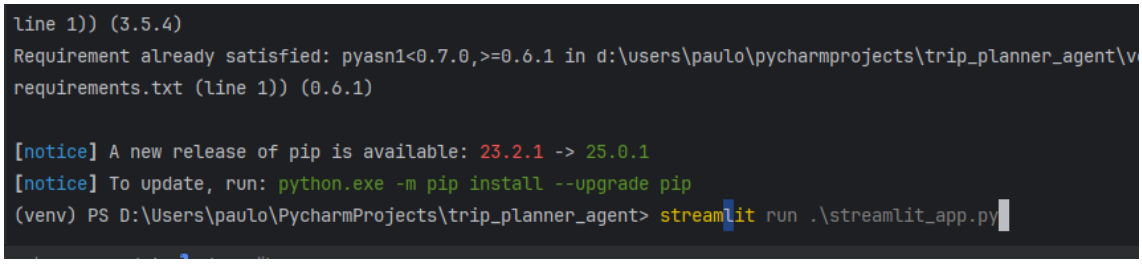
Details & Explanation

- **Streamlit UI**: The Streamlit interface is implemented in `streamlit_app.py` , where users can input their trip details.
- **Components**:
  - `./trip_tasks.py` : Contains task prompts for the agents.
  - `./trip_agents.py` : Manages the creation of agents.
  - `./tools directory` : Houses tool classes used by agents.
  - `./streamlit_app.py` : The heart of the Streamlit app.

Using GPT 3.5

To switch from GPT-4 to GPT-3.5, pass the llm argument in the agent constructor:

As orientações sugerem a instalação das bibliotecas através do arquivo de requirements, a edição do arquivo de chaves e finalmente a execução padrão do streamlit.



```
line 1)) (3.5.4)
Requirement already satisfied: pyasn1<0.7.0,>=0.6.1 in d:\users\paulo\pycharmprojects\trip_planner_agent\v
requirements.txt (line 1)) (0.6.1)

[notice] A new release of pip is available: 23.2.1 -> 25.0.1
[notice] To update, run: python.exe -m pip install --upgrade pip
(venv) PS D:\Users\paulo\PycharmProjects\trip_planner_agent> streamlit run .\streamlit_app.py
```

Porém mesmo seguindo as orientações e ainda trocando as versões do python o sistema quebra e não executa, muito provavelmente por incompatibilidade de alguma biblioteca, dado o tempo que o código foi publicado.

Por se tratar de uma solução mais organizada e mais complexa, não achei viável consertar o código ou buscar uma solução. Preferi focar nos outros entregáveis, dado o prazo das atividades.

Conclusões

Nota-se que através da utilização do streamlit é possível criar provas de conceito elaboradas para utilização de agentes. No último vídeo fica claro como é fácil e eficiente criar container com agentes, isso facilita a criação e utilização de agentes em arquiteturas modernas e distribuídas.

Além de facilitar na disponibilidade, a utilização de containers facilita a escalabilidade e a disponibilidade de soluções mais rebuscadas.

Referencias

https://www.youtube.com/watch?v=I-n9iTb6nis

https://www.youtube.com/watch?v=nKG_kbQUDDE

https://www.youtube.com/watch?v=I-n9iTb6nis

https://github.com/yeyu2/Youtube_demos