```
!pip install crewai crewai-tools
!pip install PyPDF2
```

```
import os
from crewai import Agent, Task, Crew
from PyPDF2 import PdfReader
from crewai_tools import SerperDevTool, WebsiteSearchTool
```

```
os.environ["SERPER_API_KEY"] = "7911be9c4d1c2c9835b082a6243417d20a089a08"
os.environ["OPENAI_API_KEY"] = "sk-proj-L3ccq_uCe3XFfvTkELZnGHBZsjMSrRpcACaFuqTPtKd7rebQT6QYPHU34GT3BlbkFJTNAK55Ug9EmNdkghi5iW3Y_0EKYHJyUzDN
```

```
pdf = 'WM17S.pdf'
pages_to_extract = [1, 3]
```

```
raw_text = ''
with open(pdf, 'rb') as file:
    reader = PdfReader(file)
    for page_num in pages_to_extract:
        page = reader.pages[page_num - 1]
        content = page.extract_text()
        if content:
            raw_text += content

print(raw_text)
```

```
search_tool = SerperDevTool()
web_search_tool = WebsiteSearchTool()
```

Create Agents

```
bloodtest_analyst = Agent(
    role='Blood Test Analyst',
    goal='Analyze the blood test report and summarize the findings.',
    backstory='A medical expert specializing in blood test analysis.',
    verbose=True,
    allow_delegation=False
)
```

```
article_researcher = Agent(
    role='Article Researcher',
    goal='Search for health articles based on blood test results.',
    backstory='An expert researcher proficient in finding health-related articles.',
    tools=[search_tool, web_search_tool],
    verbose=True,
    allow_delegation = False,
)
```

```
health_advisor = Agent(
    role='Health Advisor',
    goal='Provide health recommendations based on the articles found.',
    backstory='A health advisor with extensive knowledge in providing health advice.',
    verbose=True,
    allow_delegation = False,)
```

Tasks

```
analyze_bloodtest_task = Task(
    description= f'You have to analyze the blood test report from "{raw_text}" blood report',
    expected_output='A summary of the blood test results.',
    agent=blood_test_analyst,
)
```

```
find_articles_task = Task(
    description='Search for health articles based on the blood test analysis.',
    expected_output='A list of relevant health articles with links.',
    agent=article_researcher,
    context=[analyze_blood_test_task]
)
```

```
provide_recommendations_task = Task(
    description='Provide health recommendations based on the articles found.',
    expected_output='Health recommendations with links to the articles.',
    agent=health_advisor,
    context=[find_articles_task]
)
```

Crew

```
crew = Crew(
    agents=[bloodtest_analyst, article_researcher, health_advisor],
    tasks=[analyze_bloodtest_task, find_articles_task, provide_recommendations_task],
    verbose=2
)
```

```
WARNING:opentelemetry.trace:Overriding of current TracerProvider is not allowed
```

Kickoff

```
crew.kickoff()
```

Start coding or generate with AI.

Start coding or generate with AI.

Start coding or generate with AI.