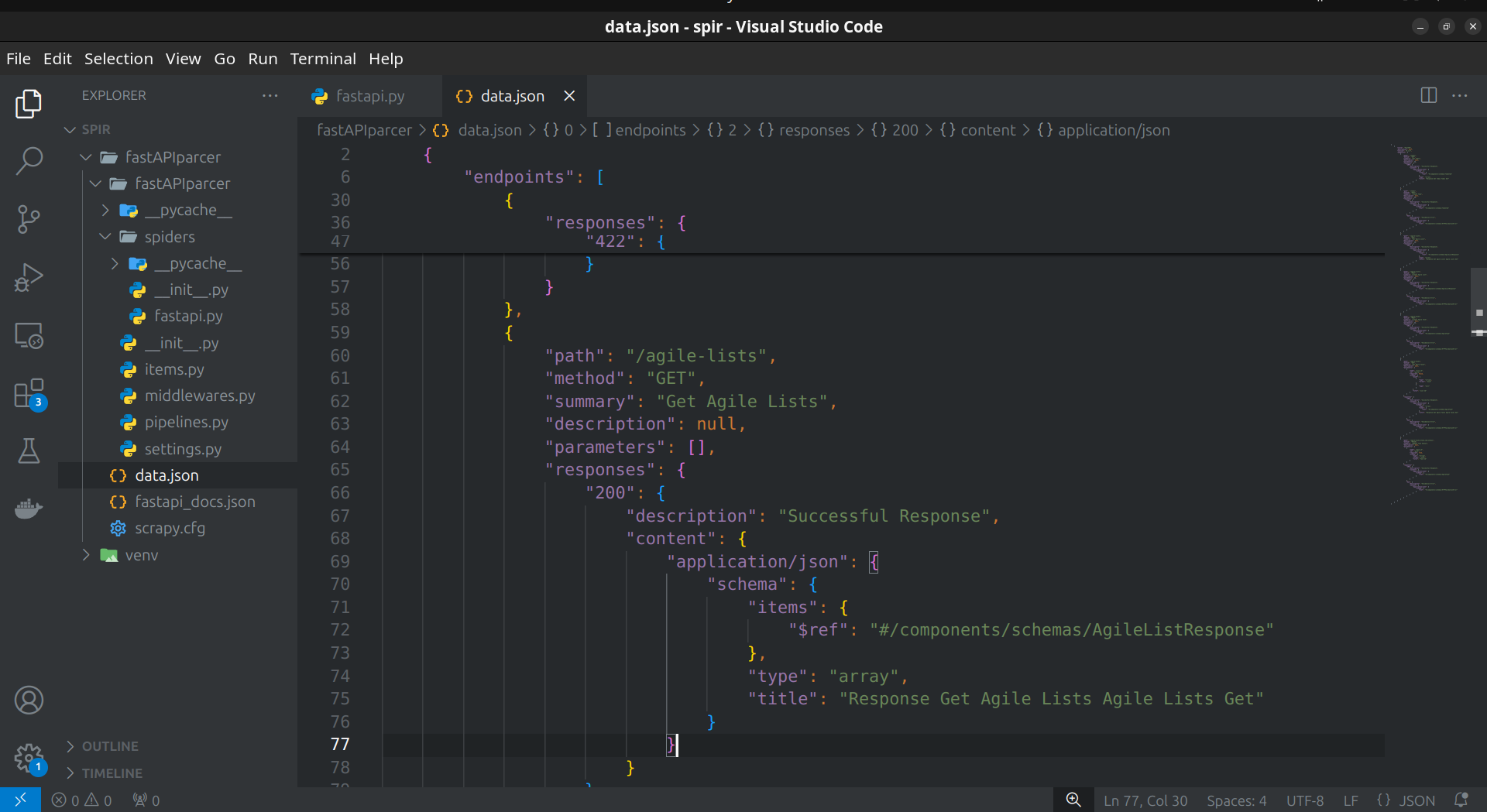
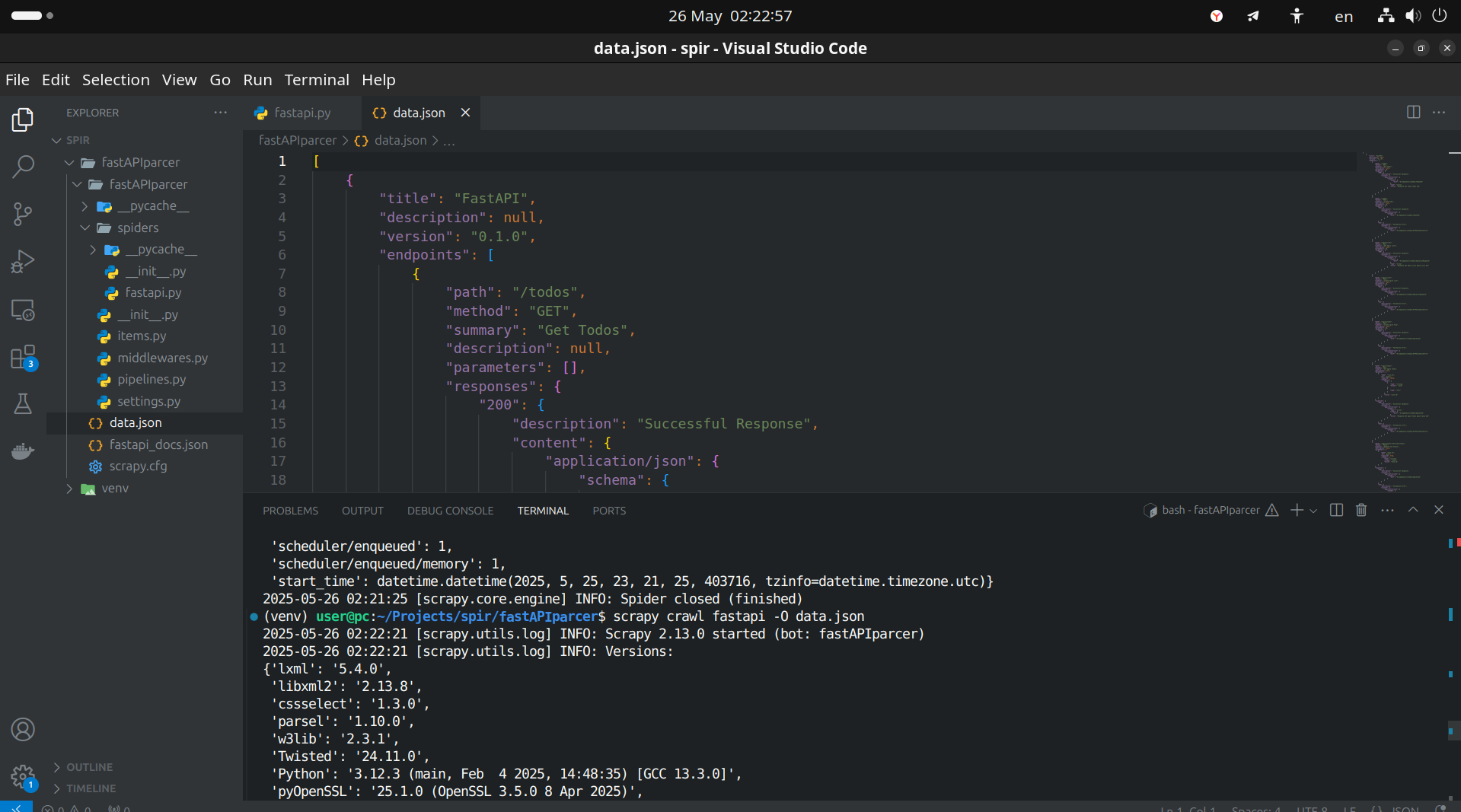
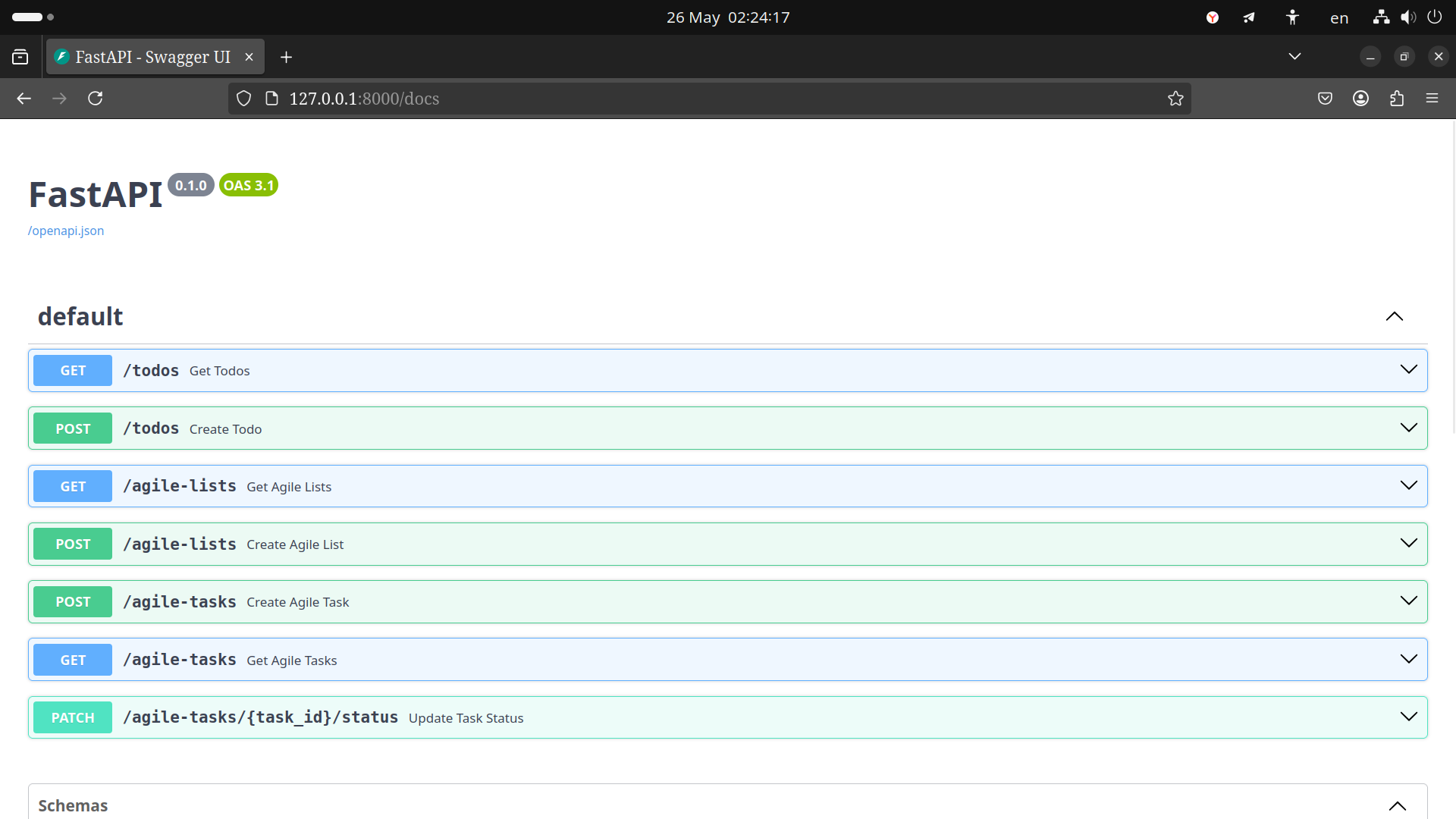
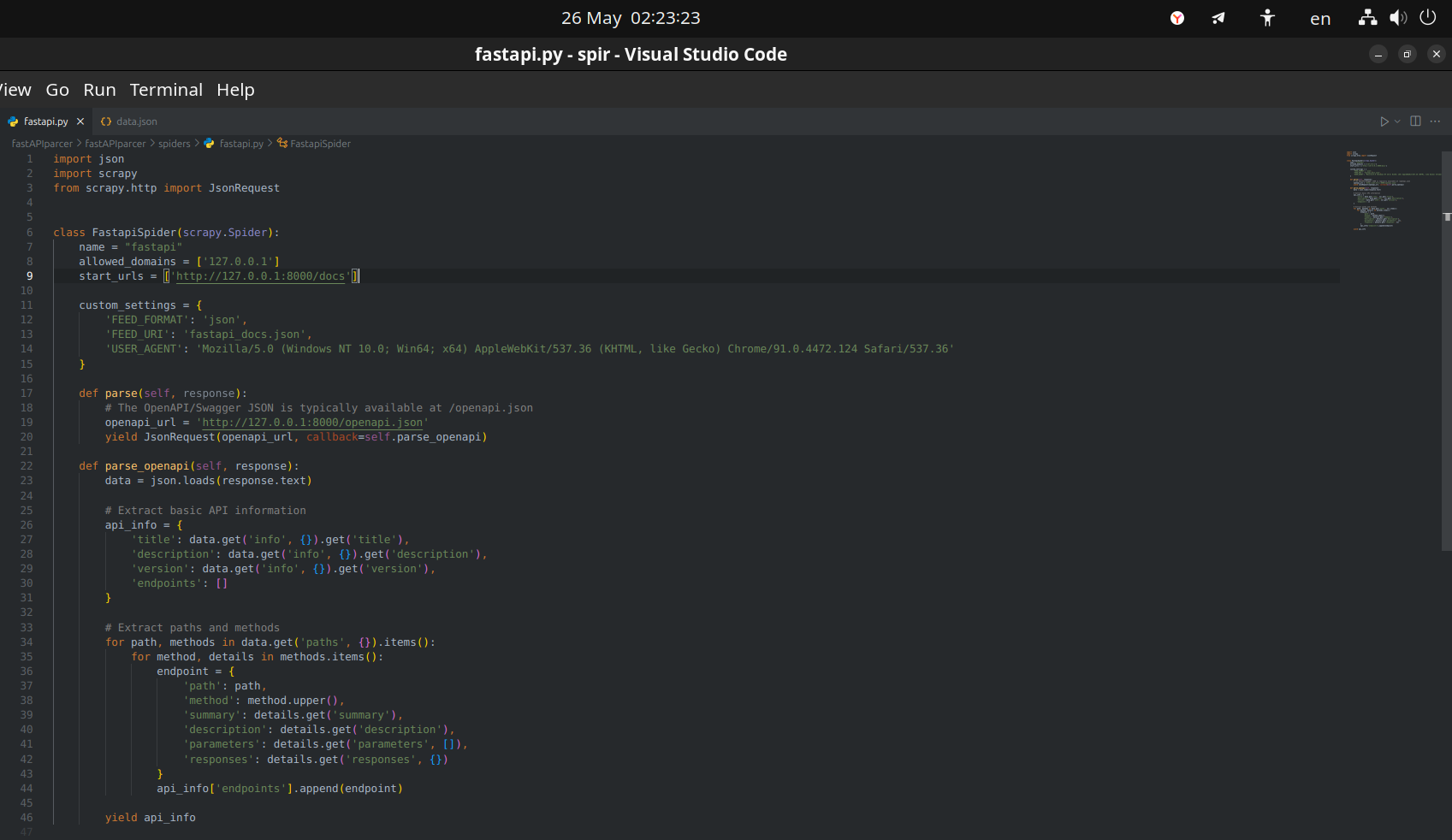
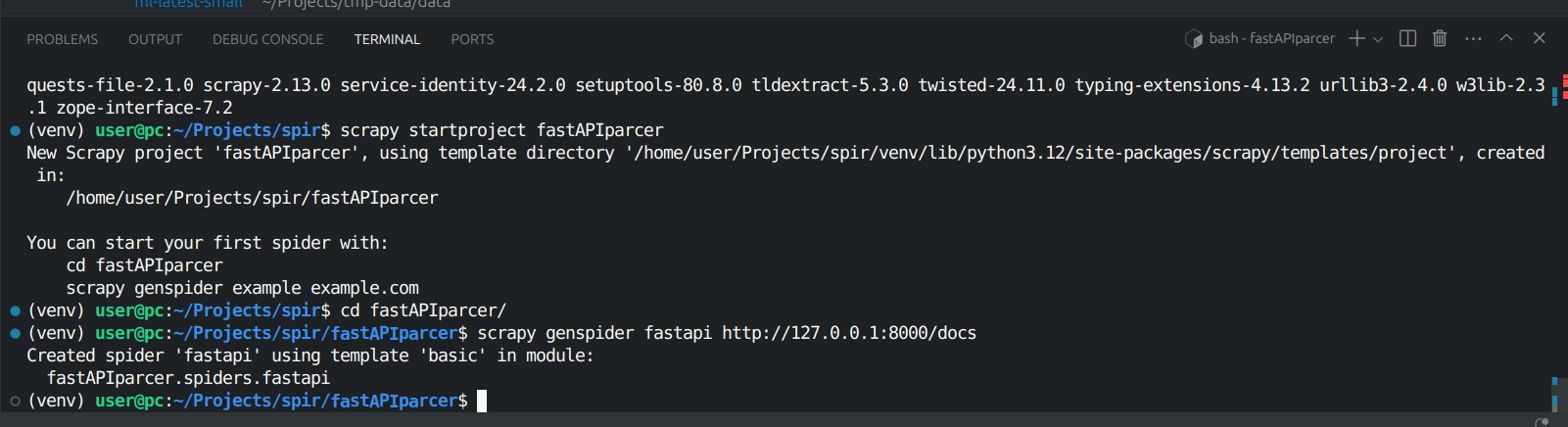
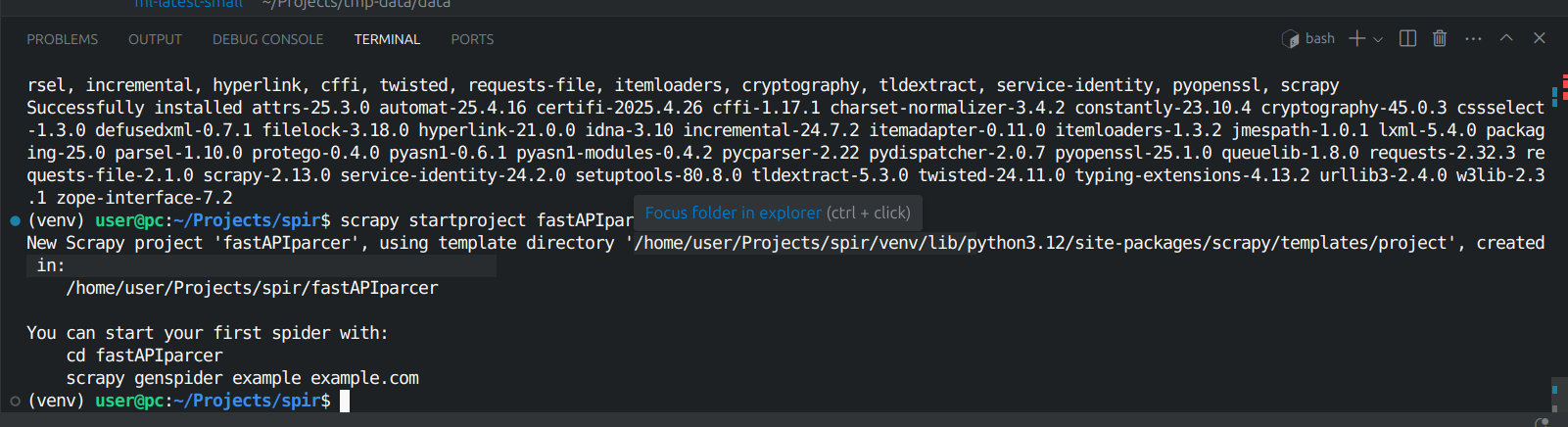
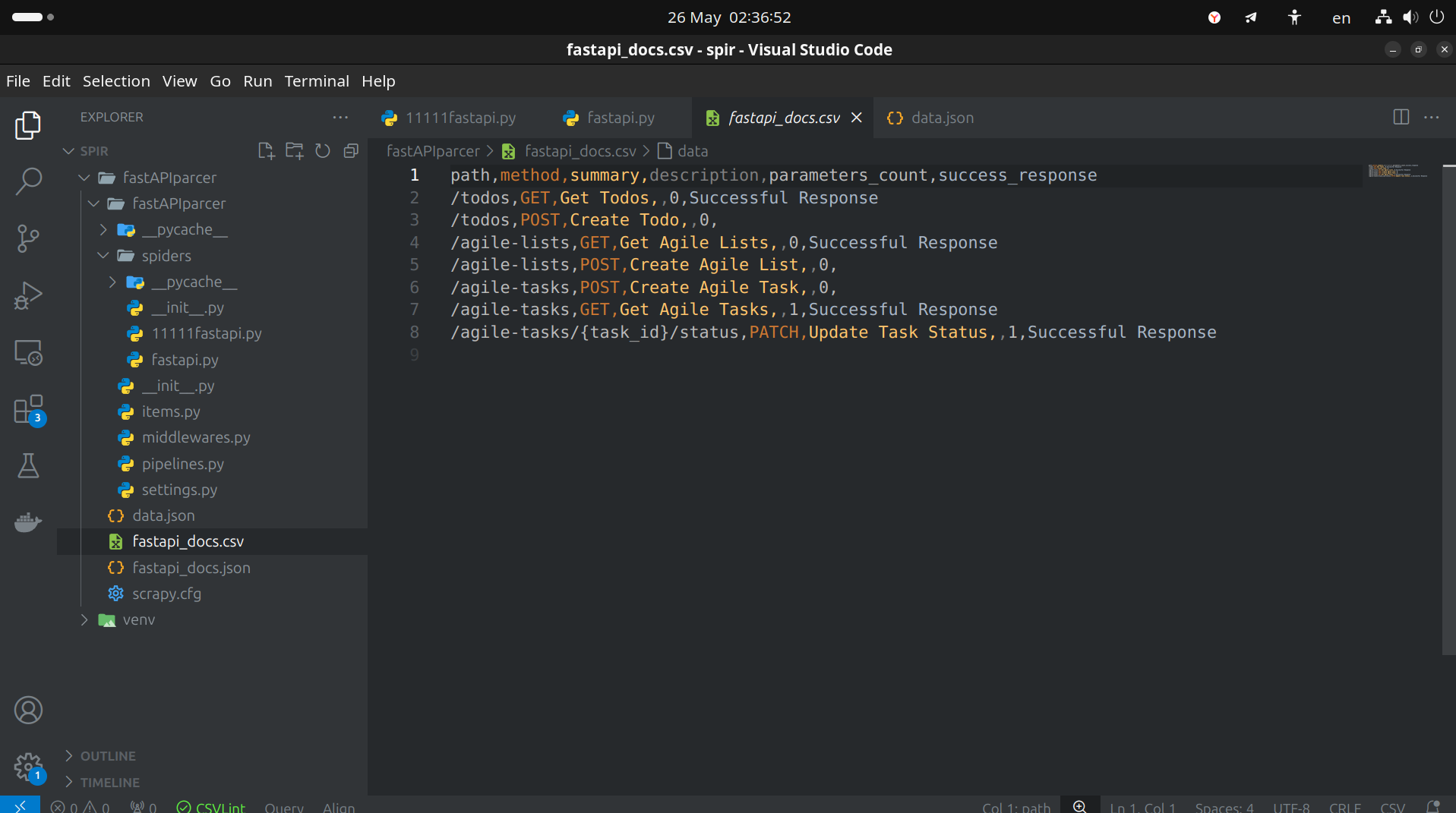
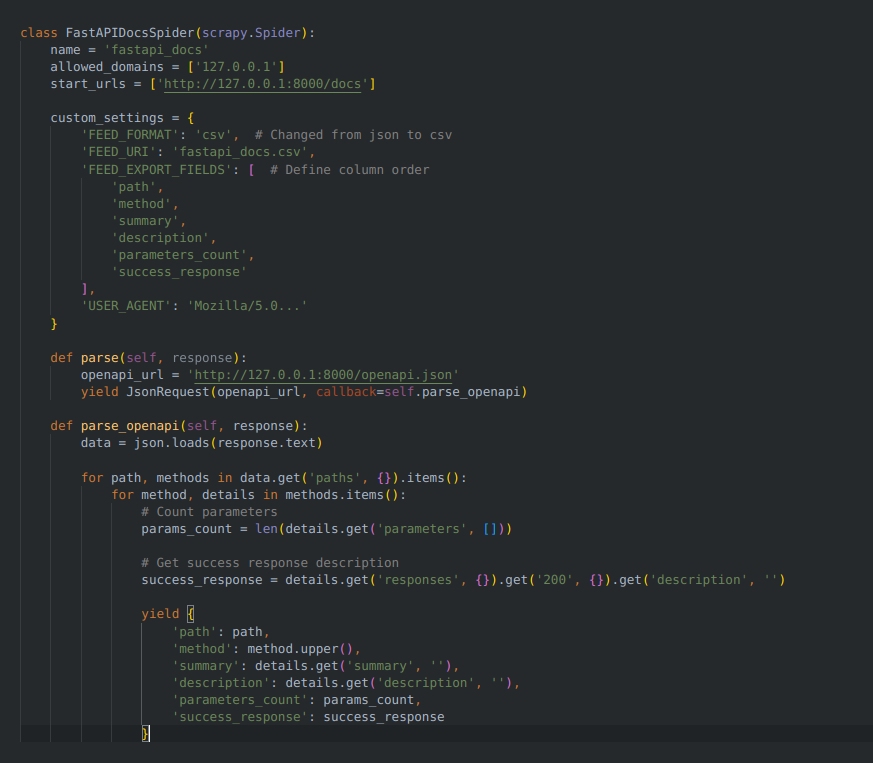
**Задание 1. Разработать модуль spider для получения данных с веб-приложений в формализованном виде**

Создание проекта



**Задание 2. Подготовить датасет и произвести его анализ**



# Анализ документации FastAPI

# Загрузка данных и импорт библиотек

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

# Настройки отображения

plt.style.use('ggplot')

pd.set\_option('display.max\_colwidth', 30)

pd.set\_option('display.width', 100)

# Загрузка данных

df = pd.read\_csv('fastapi\_docs.csv')

# 1. Первичный анализ структуры API

print("=== Основная информация ===")

print(f"Всего эндпоинтов: {len(df)}")

print(f"Уникальных путей: {df['path'].nunique()}")

print("\nМетоды API:")

print(df['method'].value\_counts())

# Создаем категории из путей

df['category'] = df['path'].apply(lambda x: x.split('/')[1] if len(x.split('/')) > 1 else 'root')

print("\nРаспределение по категориям:")

print(df['category'].value\_counts())

# 2. Визуализация распределения

fig, axes = plt.subplots(2, 2, figsize=(14, 10))

# Распределение методов

df['method'].value\_counts().plot(

kind='pie',

autopct='%1.1f%%',

colors=['#66b3ff','#99ff99','#ffcc99'],

ax=axes[0, 0]

)

axes[0, 0].set\_title('Распределение HTTP методов')

axes[0, 0].set\_ylabel('')

# Распределение по категориям

df['category'].value\_counts().plot(

kind='bar',

color='skyblue',

ax=axes[0, 1]

)

axes[0, 1].set\_title('Количество эндпоинтов по категориям')

axes[0, 1].set\_ylabel('Количество')

# Параметры

sns.countplot(

data=df,

x='parameters\_count',

palette='pastel',

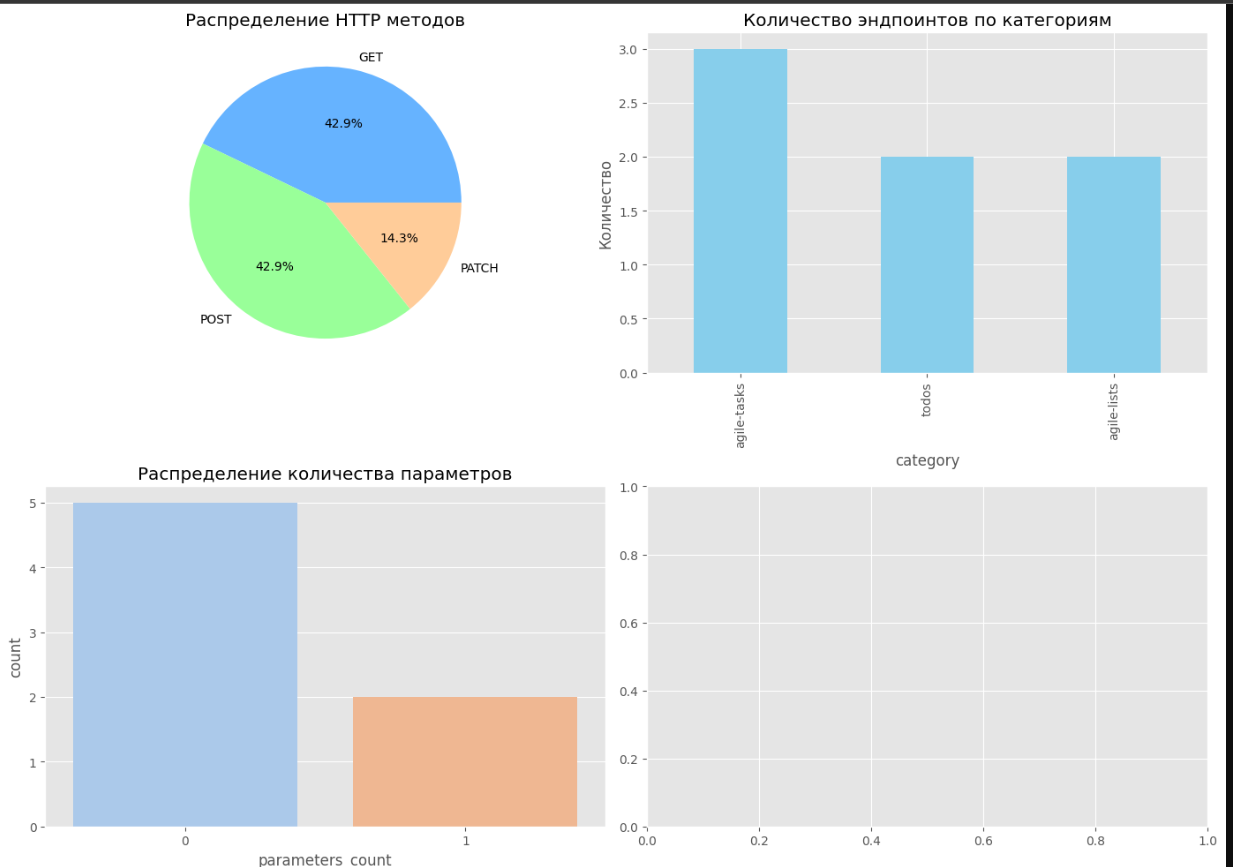
ax=axes[1, 0]

)

axes[1, 0].set\_title('Распределение количества параметров')

plt.tight\_layout()

plt.show()



**Задание 3. Изучить возможности средства Metasploit**

Metasploit Framework — это набор инструментов, позволяющих выполнять сбор информации, сканирование, эксплуатацию, разработку эксплойтов, пост-эксплуатацию и многое другое. Хотя основное применение Metasploit Framework связано с тестированием на проникновение, он также полезен для исследования уязвимостей и разработки эксплойтов.

Основные компоненты Metasploit Framework:

* msfconsole: Основной интерфейс командной строки.
* Модули: Вспомогательные модули, такие как эксплойты, сканеры, полезные нагрузки и т.д.
* Инструменты: Автономные инструменты для исследования уязвимостей, оценки уязвимостей или тестирования на проникновение.
* Эксплойт: Код, использующий уязвимость в целевой системе.
* Уязвимость: Ошибка в проектировании, коде или логике целевой системы. Эксплуатация уязвимости может привести к раскрытию конфиденциальной информации или выполнению кода на целевой системе.
* Полезная нагрузка: Эксплойт использует уязвимость, но для достижения желаемого результата (например, получения доступа к системе) требуется полезная нагрузка. Это код, который будет выполнен на целевой системе.

Вспомогательные модули (Auxiliary)

Здесь находятся поддерживающие модули, такие как сканеры, краулеры и фаззеры.

Кодировщики (Encoders)

Кодировщики позволяют закодировать эксплойт и полезную нагрузку, чтобы обойти сигнатурные антивирусные решения. Однако их эффективность ограничена, так как антивирусы могут выполнять дополнительные проверки.

Обход (Evasion)

В отличие от кодировщиков, модули обхода напрямую пытаются обойти антивирусное ПО с разной степенью успеха.

Эксплойты (Exploits)

Эксплойты, организованные по целевым системам.

NOPs (No Operation)

NOPs (от англ. "No Operation") — это инструкции, которые ничего не делают. В процессорах Intel x86 они представлены как 0x90, и процессор пропускает один цикл. Они часто используются для выравнивания размера полезной нагрузки.

Полезные нагрузки (Payloads)

Полезные нагрузки — это код, который выполняется на целевой системе. Эксплойты используют уязвимости, но для достижения цели (например, получения оболочки или выполнения команды) требуется полезная нагрузка.

Полезные нагрузки делятся на:

* Адаптеры (Adapters): Обёртки для преобразования полезных нагрузок в разные форматы.
* Одиночные (Singles): Самодостаточные полезные нагрузки (например, добавление пользователя или запуск notepad.exe).
* Стадии (Stagers и Stages): Stagers устанавливают канал связи, а Stages загружают основную часть полезной нагрузки

Пост-эксплуатация (Post)

Модули пост-эксплуатации полезны на завершающем этапе тестирования на проникновение.

**Задание 4. Изучить возможности средств OSINT**

1. Maltego

**Назначение**:

Maltego — это мощный инструмент для анализа связей и визуализации данных. Он помогает исследовать взаимосвязи между людьми, доменами, IP-адресами, email-адресами и другими сущностями в интернете.

**Особенности**:

* Автоматизирует сбор данных из открытых источников.
* Позволяет строить графы связей между объектами.
* Интегрируется с различными API и базами данных.

**Пример использования:**

Поиск всех связанных доменов с определённой компанией или выявление связей между сотрудниками организации.

2. Shodan

**Назначение:**

Shodan — это поисковая система для интернета вещей (IoT), которая индексирует устройства, подключённые к интернету (серверы, камеры, роутеры и т.д.).

**Особенности**:

* Позволяет находить уязвимые устройства по ключевым словам (например, "default password")
* Предоставляет информацию о сервисах, открытых портах и ПО на устройствах.
* Полезен для пентестеров и исследователей безопасности.

**Пример использования:**

Поиск незащищённых веб-камер или серверов с устаревшим ПО.

3. Google Dorks

**Назначение:**

Google Dorks — это специальные поисковые запросы, которые позволяют находить скрытую или чувствительную информацию в интернете с помощью Google.

**Особенности:**

* Использует операторы Google (например, site:, filetype:, intitle:).
* Может выявлять открытые базы данных, конфиденциальные документы и уязвимые страницы.

**Пример использования:**

Поиск файлов с паролями (filetype:txt password) или списков пользователей на определённом сайте (site:example.com intext:"username").

4. Recon-ng

**Назначение:**

Recon-ng — это фреймворк для проведения разведки и сбора информации, написанный на Python.

**Особенности:**

* Модульная структура (есть модули для работы с WHOIS, DNS, соцсетями и т.д.).
* Автоматизирует рутинные задачи разведки.
* Интегрируется с внешними API (например, Shodan).

**Пример использования:**

Сбор информации о домене, включая поддомены, email-адреса и связанные аккаунты в соцсетях.

5. Harvester (TheHarvester)

**Назначение:**

TheHarvester — это инструмент для сбора email-адресов, поддоменов, IP-адресов и другой информации о цели.

**Особенности:**

* Агрегирует данные из поисковиков (Google, Bing), PGP-серверов и соцсетей.
* Прост в использовании (запускается из командной строки).
* Часто используется на начальном этапе пентеста.

**Пример использования:**

Поиск всех email-адресов, связанных с доменом компании.