Python Itinerary Generator Checklist

Steps to build a dynamic itinerary planner with AI and web scraping.

Based on Building A Python Itinerary Generator With Ai And Location Scraping Cursor Gemini 2 5 Flash by Casey Builds It

---

1. Project Setup and Core Features

Define the foundational elements of your itinerary generator.

- ■ Set up a Python environment for the project.
- ■ Define core feature set: date selection.
- ■ Define core feature set: party composition (adults, kids, dogs).
- ■ Define core feature set: budget level.
- ■ Define core feature set: custom activity requests.
- ■ Implement PDF export functionality.
- ■ Integrate with an LLM for conversational feature definition (e.g., Gemini 2.5 Flash).
- ■ Use an IDE like Cursor for real-time testing and debugging.

2. Location Data Scraping Strategy

Plan how to dynamically pull location-based information.

- ■ Identify free data sources: OpenStreetMap.
- ■ Identify free data sources: Wikipedia.
- ■ Identify free data sources: DuckDuckGo.
- ■ Develop logic to pull real businesses, restaurants, and activities.
- ■ Test scraping against a specific location (e.g., Colorado Springs).
- ■ Test scraping against a location with ambiguous place names (e.g., Portland, Maine).
- ■ Compare free scraping methods against paid API options (e.g., Google Places API).
- ■ Analyze cost implications of API usage for scaling.

3. AI Integration and LLM Usage

Leverage AI for defining and refining itinerary generation.

- ■ Feed reference material from existing itinerary features into Cursor.
- ■ Define initial features conversationally with the LLM.
- ■ Test LLM-generated suggestions for itinerary content.
- ■ Compare LLM performance and cost (e.g., Gemini Flash vs. Claude).
- ■ Use LLM for debugging assistance during the build.

4. Testing and Debugging

Identify and resolve issues during the development process.

- [ ] Perform live testing against diverse locations.
- [ ] Debug broken location scrapes.
- [ ] Address issues with ambiguous or duplicate place names.
- [ ] Validate the usability of generated itineraries.
- [ ] Test PDF export for accuracy and formatting.
- [ ] Compare results from different scraping methods.

5. Enhancements and Future Development

Explore potential upgrades to the itinerary generator.

- [ ] Integrate a Weather API for improved day-by-day planning.
- [ ] Implement GPS-based distance filtering for realistic suggestions.
- [ ] Incorporate Event APIs to find local concerts and happenings.
- [ ] Develop database caching for previously scraped areas.
- [ ] Plan architecture for converting the script into a web application.
- [ ] Consider user authentication for a web app version.
- [ ] Add options for different travel styles or preferences.

6. Code and Tooling

Select and utilize appropriate tools for development.

- [ ] Choose Python as the primary programming language.
- [ ] Utilize Cursor IDE for integrated AI assistance.
- [ ] Leverage Gemini 2.5 Flash or similar LLM for AI tasks.
- [ ] Implement web scraping libraries (e.g., BeautifulSoup, Scrapy).
- [ ] Use PDF generation libraries (e.g., ReportLab, FPDF).
- [ ] Consider version control (e.g., Git).
- [ ] Set up a virtual environment for dependencies.

Resources & Further Reading

Use these pages while working through the checklist:

- Casey Builds It

https://caseybuildsit.com/the-files/

- Casey Builds It

https://caseybuildsit.com/make-contact/

- Casey Builds It

https://caseybuildsit.com/

- Downloads

https://caseybuildsit.com/downloads/

Casey Builds It | -
- | caseybuildsit.com
Build dynamic itineraries using Python, AI, and free data sources.