

Python Itinerary Generator Checklist

Steps to build a dynamic itinerary planner with AI and location 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 Feature Definition

Define the initial scope and features for your itinerary generator.

- Identify the core purpose: a standalone Python itinerary planner.
- Define input parameters: dates, number of adults, kids, dogs, budget, custom activities.
- Determine output format: PDF export for finished itineraries.
- Establish data sourcing strategy: dynamic scraping vs. pre-scraped content.
- Set up development environment with Cursor and an LLM (e.g., Gemini 2.5 Flash).
- Feed reference material (e.g., from an existing Airbnb-style feature) into Cursor.
- Define core feature set conversationally with the LLM.

2. Data Sourcing and Scraping Strategy

Plan how to gather real-time location data for itinerary generation.

- Select free data sources: OpenStreetMap, Wikipedia, DuckDuckGo.
- Configure the tool to pull location-based ideas on the fly.
- Research and compare free scraping methods against paid APIs (e.g., Google Places API).
- Evaluate the trade-offs between free and paid data sources for usability and cost.
- Implement logic to find real businesses, restaurants, and activities.
- Address potential issues with ambiguous or duplicate place names during scraping.

3. AI Integration and LLM Usage

Leverage AI for itinerary generation and conversational development.

- Choose an LLM for AI-powered features (e.g., Google Gemini 2.5 Flash).
- Use the LLM for defining features conversationally.
- Test and debug itinerary generation logic with the LLM in real-time.
- Compare LLM costs (e.g., Gemini Flash vs. Claude) for scalability.
- Utilize AI to process user input and generate activity suggestions.

4. Development and Testing

Build and test the itinerary generator with real-world scenarios.

- Implement the core scaffolding for the itinerary planner.
- Test the tool live against different locations (e.g., Colorado Springs, Portland, Maine).
- Conduct debugging sessions for broken location scrapes.

- Verify PDF export functionality for generated itineraries.
- Ensure the tool handles various input combinations (dates, party size, budget).

5. Debugging and Refinement

Identify and resolve issues encountered during development.

- Address errors related to location scraping.
- Troubleshoot issues with ambiguous or duplicate place names.
- Optimize scraping logic for efficiency and accuracy.
- Compare results from different scraping methods.
- Refine AI prompts for better itinerary generation.
- Test edge cases and unexpected user inputs.

6. Future Enhancements and Scalability

Explore potential upgrades and pathways for the itinerary generator.

- Integrate a Weather API for dynamic day-by-day planning.
- Implement GPS-based distance filtering for realistic suggestions.
- Incorporate Event APIs to surface local happenings.
- Develop database caching for previously scraped areas.
- Plan the transition from a standalone script to a full web application.
- Consider user authentication and data management for a web app.

7. Cost and API Considerations

Evaluate the financial implications of API usage and data sourcing.

- Compare costs of free scraping versus paid APIs.
- Analyze LLM pricing models for different providers.
- Determine cost-effectiveness for personal projects versus scaled applications.
- Factor in API rate limits and usage policies.
- Plan for potential future API subscription costs.

Resources & Further Reading

Use these pages while working through the checklist:

■ Casey Builds It

<https://caseybuildsit.com/>

■ Casey Builds It

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

■ Downloads

<https://caseybuildsit.com/downloads/>

■ Casey Builds It

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

Casey Builds It | -
- | caseybuildsit.com

Build dynamic itineraries using AI, location scraping, and conversational coding.