**Project Outline: AI-Powered Website Generator with Database**

**Project Goal**

Create a tool that asks a user a few questions about their website, then uses AI to:

1. Generate the website structure (HTML + CSS).
2. Populate content automatically (e.g., summaries, text, placeholders).
3. Set up a simple database to store dynamic content (like blog posts, user info, or product data).

**1. User Interaction**

* The program runs in a **terminal/CLI** or simple web form.
* Questions to ask:
  1. What is the website for? (business, blog, portfolio, e-commerce)
  2. Number of pages (Home, About, Contact, Blog, etc.)
  3. Should it have a login/signup system?
  4. Do you want sample content generated automatically?
  5. Theme/style preferences (light, dark, custom colors)

**2. AI Planning Module**

* Inspired by **DesignerAgent**:
  + Maps user answers into a **website plan**:
  + plan = {
  + "pages": ["home", "about", "contact", "blog"],
  + "login\_required": True,
  + "theme": "dark",
  + "generate\_content": True
  + }
* Could use a **lightweight LLM or prompt-based AI** to summarize info for the pages.
  + Example: User says “My website is about cats.”
  + AI generates a paragraph for “About” and sample blog post titles.

**3. AI Content Generation**

* Inspired by **AI search summarizer**:
  + Generate simple text snippets for each page.
  + Optionally generate “blog posts” or “products” for the database.
* Example workflow:
  + Ask user: “What is your business/product about?”
  + Use AI to generate 2-3 paragraphs for the homepage.
  + Use AI to create 2-3 sample items in a database table.

**4. Coder Module**

* Inspired by **CoderAgent**:
  + Create HTML/CSS files for each page.
  + Add navigation links based on pages.
  + Optionally include a **login/signup page** (basic HTML form).
  + Link CSS with a theme chosen by the user.

**5. Database Integration**

* Use **SQLite** (for simplicity) to store:
  + Blog posts or product listings
  + User info (optional, if login is enabled)
* Populate database with AI-generated sample data.
* Optionally, create a **small backend API** (Flask or FastAPI) to display database content dynamically on the website.

**6. File Structure**

project/

│

├─ builder.py # Main program, asks questions, runs agents

├─ agents/

│ ├─ designer\_agent.py

│ ├─ coder\_agent.py

│ └─ content\_agent.py # AI content generator

├─ data/

│ ├─ output/ # Generated HTML/CSS files

│ └─ database.db # SQLite database

└─ templates/ # Optional HTML/CSS templates

**7. Optional Features**

* Auto-generate favicon/logo using AI.
* Generate meta tags for SEO automatically.
* Add basic JavaScript for interactive elements (like a slideshow).
* Export the whole project as a ready-to-upload folder.

**8. Learning Outcomes**

* Combining **user input → AI planning → AI content generation → website assembly → database population**.
* Understanding **basic full-stack flow**:
  + Frontend (HTML/CSS)
  + Backend (Python + SQLite)
  + AI-driven content
* Hands-on experience with **modular AI agents** orchestrating a larger project.