💡 Using AI & Building Prompts for Backend Development

🤖 What is AI-Assisted Coding?

AI-assisted coding is when you use tools like GitHub Copilot or ChatGPT to help you write and understand code. Instead of trying to figure everything out on your own, you can type in natural-language questions or requests (called *prompts*). The AI can then suggest code, explain errors, or guide you step by step.

In this hackathon, you'll use AI as a tool to help you through the process of building your project. This guide focuses on how to get the most out of AI when working on the backend of your website

👩‍💻 How to Write Good Prompts

When talking to an AI, the way you phrase your request (*prompt*) makes a big difference. Here are some tips:

1. **Be specific** – Clearly describe what you want the AI to do.
2. **Include context** – Mention the language, framework, or database you're using.
3. **Ask for explanations** – Treat the AI like a tutor when you need clarity.
4. **Iterate and refine** – Improve your prompt if the first answer isn't right.
5. **Break tasks into steps** – Ask one piece at a time instead of everything at once.
6. **Set the format** – Tell the AI how you want the answer (e.g., code + plain English explanation)

⚡Prompt Cheat Sheet

Taking the tips mentioned above, here are some example prompts under the contexts of the WiT starter repository.

Goal	Weak Prompt	Strong Prompt (for this repo)	Location & Use
Create a backend route	"Make a user API."	"Generate a FastAPI endpoint for customer registration. It should accept firstName, lastName, email, and phone, and save them in MongoDB."	python-starter/src/fastapiproject/routers/customer_router.py → Use when adding or modifying the /customers **POST** endpoint.
Database schema	"Make a database."	"Design a MongoDB collection schema for customers using Pydantic. Include fields: firstName, lastName, email, phone, and dateOfBirth."	python-starter/src/fastapiproject/model/customer.py → Use when designing or updating the **Customer Pydantic model** for MongoDB.
Debugging code	"Fix my code."	"When the application is calling /products I'm not getting any products returned. Here's the code and error: [paste code + error]. Can you explain and fix?"	python-starter/src/fastapiproject/routers/product_router.py or python-starter/src/fastapiproject/routers/customer_router.py → Use when troubleshooting **route errors**.
Explanations	"What does this do?"	"Explain how this put router is fetching all customers line by line, as if I'm new to backend development: [paste code]."	python-starter/src/fastapiproject/routers/customer_router.py → Use when asking for **line-by-line explanations** (e.g., get_all_customers).

Step-by-step build	"Make me a backend."	"Help me build a FastAPI app step by step. First, show me how to set up the project and install dependencies. Then, guide me to create a customer route."	python-starter/src/fastapiproject/main.py and python-starter/src/fastapiproject/routers/customer_router.py → Use when **scaffolding** the app and adding first routes.
Adding features	"Add login."	"Update my FastAPI app to include a login route. It should check the email and password against a users collection in MongoDB and return a JWT if valid."	New file: python-starter/src/fastapiproject/routers/auth_router.py → Use when implementing **authentication**.
Format answers	"Write code for login."	"Write the code for a login endpoint in FastAPI. First show me the code, then explain how it works in plain English."	python-starter/src/fastapiproject/routers/auth_router.py → Use when requesting **code + explanation** for authentication routes.