

BUILD A FULL-STACK E-COMMERCE APPLICATION WITH OPENCODE

A two-week practical vibe-coding sprint



1 Course overview

1	Course overview	2
2	Build a full-stack e-commerce application	1
2.1	A practical AgileMinds Hub sprint using OpenCode	1
2.2	What you will build	1
3	What you will learn	1
4	The OpenCode workflow	2
4.1	The project instruction files	2
5	Course modules	3
5.1	Module 1 · Local setup and OpenCode	3
5.2	Module 2 · Requirements, design, and agent instructions	3
5.3	Module 3 · Next.js foundation	3
5.4	Module 4 · MongoDB and data models	3
5.5	Module 5 · Storefront and product catalog	3
5.6	Module 6 · Authentication and accounts	3
5.7	Module 7 · Shopping cart	4
5.8	Module 8 · Checkout, orders, and Paystack	4
5.9	Module 9 · Order history and email	4
5.10	Module 10 · Admin dashboard	4
5.11	Module 11 · Security, testing, and deployment	4
5.12	Module 12 · Final demo and handoff	4
6	The product journey	5
7	Payments and operations	5
8	Admin, security, and release	6
9	Tools and technologies	6
10	What you will have at the end	7
11	Who this course is for	7
12	Course completion standard	7

A focused two-week OpenCode sprint from local setup to a working e-commerce MVP.

2 Build a full-stack e-commerce application

2.1 A practical AgileMinds Hub sprint using OpenCode

Turn an idea into a working e-commerce application using **OpenCode**, **Next.js**, **MongoDB**, **Paystack**, and modern software-engineering practices.

This is the practical continuation of the **Software Engineering and Vibe Coding** course. Instead of only learning how software works, you will use OpenCode to set up a project, direct an AI coding agent, and build a functional product from scratch.

Format	Primary tool	Project
Maximum two-week sprint	OpenCode	Full-stack e-commerce MVP
Module-based learning	Next.js + MongoDB	Payments, orders, admin, deployment

2.2 What you will build

You will build a responsive customer storefront and a protected administrative workspace. Customers will browse products, create accounts, manage a cart, complete checkout, pay through Paystack test mode, receive order confirmation, and view order history. Administrators will manage products, categories, availability, and orders.

“The goal is not to type every line manually. The goal is to direct OpenCode effectively, provide the right context, review the implementation, test the result, and take responsibility for the product that is built.”

3 What you will learn

Set up the environment.

Use the terminal, install OpenCode, configure an AI model provider, create a project folder, and connect GitHub.

Build with Next.js.

Set up the App Router, TypeScript, layouts, components, server logic, validation, and application structure.

Build commerce flows.

Create the catalog, authentication, cart, checkout, orders, Paystack payment flow, and customer email.

Create project instructions.

Write `PRODUCT_REQUIREMENTS.md`, `DESIGN.md`, and `AGENTS.md` so the agent has a clear source of truth.

Connect the data.

Connect MongoDB and create the collections needed for users, products, carts, orders, and payments.

Prepare for release.

Add admin operations, security checks, testing, environment configuration, and deployment readiness.

4 The OpenCode workflow

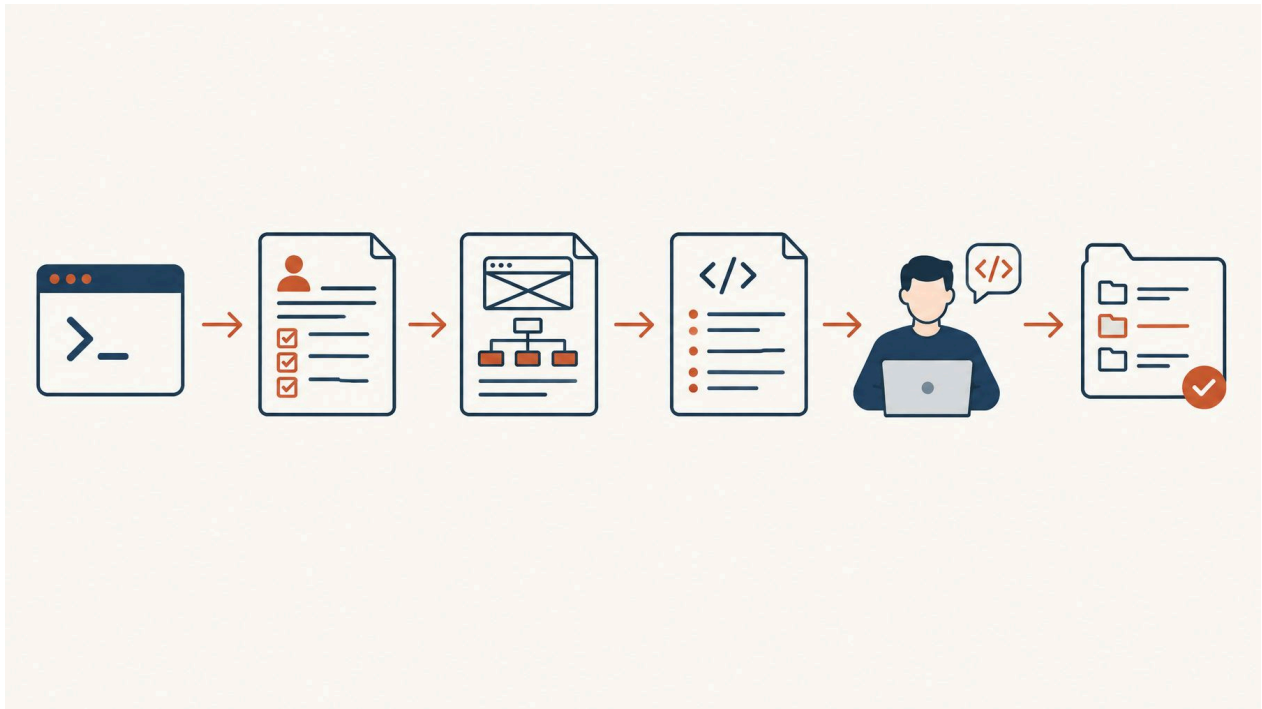


Figure 1: A repeatable loop from local setup and project context to implementation, review, and a working repository.

OpenCode is used throughout the project as the primary AI coding environment. The learner follows a simple loop: inspect the project, plan the change, approve the scope, implement one feature, test the result, review the files, and record what remains.

4.1 The project instruction files



Figure 2: Three files give OpenCode the context it needs to build consistently.

File	Purpose
PRODUCT_REQUIREMENTS.md	Defines what the product must do, its scope, user flows, features, rules, and acceptance criteria.
DESIGN.md	Defines how the product must look and behave, including visual tokens, layouts, components, states, and accessibility.
AGENTS.md	Defines how OpenCode should work inside the repository, including architecture, security, testing, and reporting rules.

5 Course modules

The course is organized into implementation modules rather than weeks. Learners may complete multiple modules in a day during the two-week sprint.

5.1 Module 1 · Local setup and OpenCode

Set up the terminal, project folder, GitHub repository, OpenCode, and an AI model provider such as Claude or another supported provider.

Outcome: You can open the project locally, run commands, launch OpenCode, and direct an AI coding agent.

5.2 Module 2 · Requirements, design, and agent instructions

Gather the e-commerce requirements, define the MVP, map the user journeys, create PRODUCT_REQUIREMENTS.md, write DESIGN.md, and create AGENTS.md.

Outcome: You have a clear product scope, visual system, and engineering instruction system before implementation begins.

5.3 Module 3 · Next.js foundation

Create the Next.js project with TypeScript, App Router, styling, layouts, navigation, shared components, loading states, and error handling.

Outcome: You have a clean full-stack application foundation ready for features.

5.4 Module 4 · MongoDB and data models

Connect MongoDB and create the models for users, products, categories, carts, orders, payments, and addresses.

Outcome: Your application can store and retrieve the data required for commerce workflows.

5.5 Module 5 · Storefront and product catalog

Build the homepage, product listings, categories, search, filters, product cards, product details, availability states, and responsive layouts.

Outcome: Customers can discover and understand the products in your store.

5.6 Module 6 · Authentication and accounts

Build registration, login, logout, sessions, customer profiles, protected routes, roles, and access control.

Outcome: Customers can securely create accounts and access their own information.

5.7 Module 7 · Shopping cart

Build add-to-cart, quantity updates, item removal, cart persistence, empty states, stock checks, and server-side validation.

Outcome: Customers can create and manage a reliable shopping cart.

5.8 Module 8 · Checkout, orders, and Paystack

Build checkout, secure total calculation, order creation, Paystack transaction initialization, payment verification, webhooks, payment states, and duplicate-event protection.

Outcome: Customers can complete a test purchase and the system can safely record and verify the payment.

5.9 Module 9 · Order history and email

Build order confirmation, order history, order details, status updates, and transactional email using Google SMTP or an approved provider.

Outcome: Customers can track their purchases and receive relevant notifications.

5.10 Module 10 · Admin dashboard

Build protected admin pages for creating products, editing products, managing categories, updating availability, viewing orders, and changing fulfillment status.

Outcome: Store administrators can manage the core operations of the platform.

5.11 Module 11 · Security, testing, and deployment

Review secrets, authentication, authorization, validation, payment security, testing, production configuration, and deployment.

Outcome: Your application is tested, secured, documented, and ready to deploy or demonstrate as a working MVP.

5.12 Module 12 · Final demo and handoff

Present the customer journey, admin journey, architecture, OpenCode workflow, major decisions, known limitations, and future improvements.

Outcome: You can explain, demonstrate, and continue improving the product you built.

6 The product journey



Figure 3: The customer journey moves from discovery to product understanding, cart, checkout, and confirmation.

“Set up your environment → define the product → guide OpenCode → build the storefront → connect the database → add authentication → build the cart → implement checkout → integrate Paystack → manage orders → secure and test the application → deploy and present the product.”

7 Payments and operations

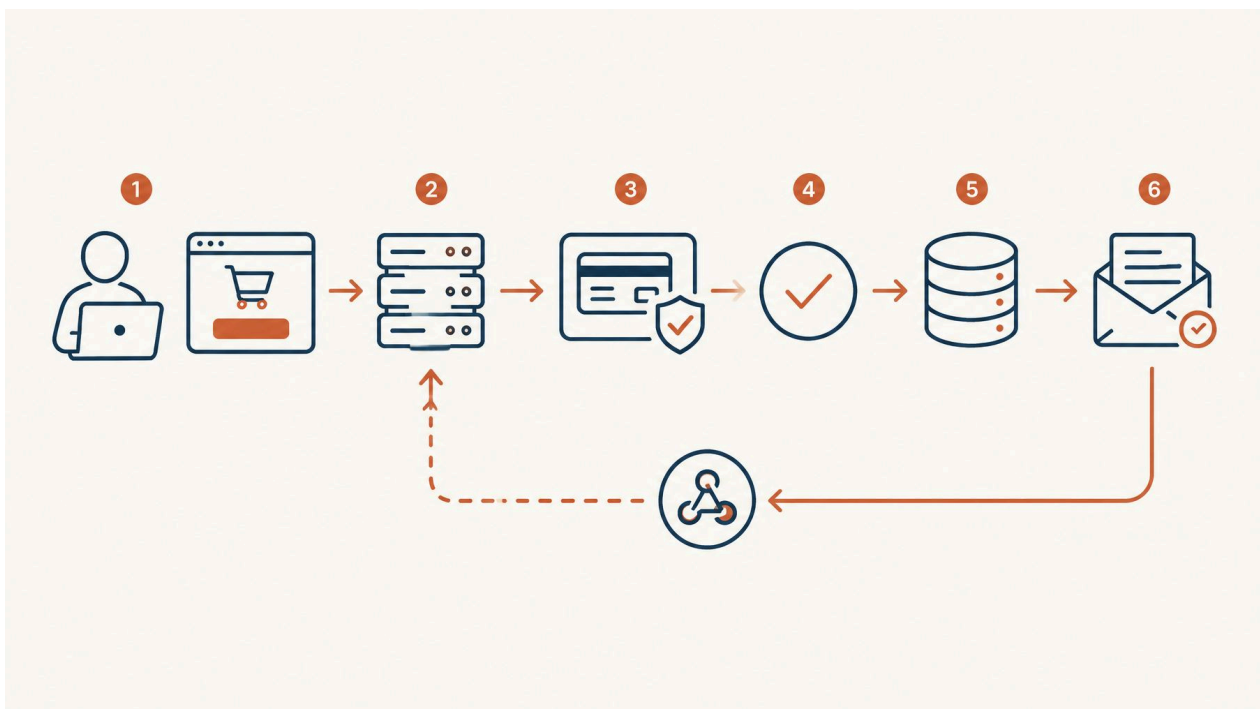


Figure 4: The payment flow connects checkout, server-side verification, order records, notifications, and webhook events.

The payment module teaches learners to initialize Paystack transactions from the server, verify payment status and amount, handle webhook events, prevent duplicate fulfillment, and show the right state when payment succeeds, fails, is abandoned, or remains uncertain.

8 Admin, security, and release



Figure 5: The final stage connects store operations, security review, testing, deployment, and monitoring. Learners finish by building the admin dashboard, reviewing environment variables and secrets, testing the important customer and payment journeys, preparing production configuration, and deploying or documenting a deployment-ready application.

9 Tools and technologies

Area	Technology
AI coding environment	OpenCode
AI model	Claude or another supported provider
Application framework	Next.js with TypeScript and the App Router
Frontend	React and the approved design system
Database	MongoDB
Payment gateway	Paystack
Email	Google SMTP or an approved provider

Version control	Git and GitHub
Deployment	A Next.js-compatible hosting provider

10 What you will have at the end

By completing the course, you will have:

1. A working full-stack e-commerce MVP.
2. A GitHub repository with your project source code.
3. `PRODUCT_REQUIREMENTS.md`, `DESIGN.md`, and `AGENTS.md` files.
4. A MongoDB-backed product and order system.
5. A working customer storefront and admin dashboard.
6. A Paystack test-payment flow with server-side verification.
7. Authentication, cart, checkout, orders, and email notifications.
8. Testing and security review evidence.
9. A deployed application or deployment-ready project.
10. The confidence to use OpenCode to build and improve future products.

11 Who this course is for

This course is for founders, product managers, aspiring developers, designers, and beginners who want to use AI to build a real software product instead of only experimenting with isolated code examples.

You should understand the basic concepts covered in the **Software Engineering and Vibe Coding course**. You do not need to be an expert developer, but you should be comfortable using a computer, following terminal instructions, reading basic code, and reviewing AI-generated work.

12 Course completion standard

To complete the course successfully, you must demonstrate a customer journey from product discovery through cart, checkout, Paystack test payment, order confirmation, and order history. You must also demonstrate a protected admin dashboard, product and order management, project documentation, basic testing, security checks, and deployment or deployment readiness.

Understand the foundation. Direct the agent. Build the product.

Prerequisite: Software Engineering and Vibe Coding course · AgileMinds Hub