Purwadhika
Digital Technology School

Full Stack AI Software Development

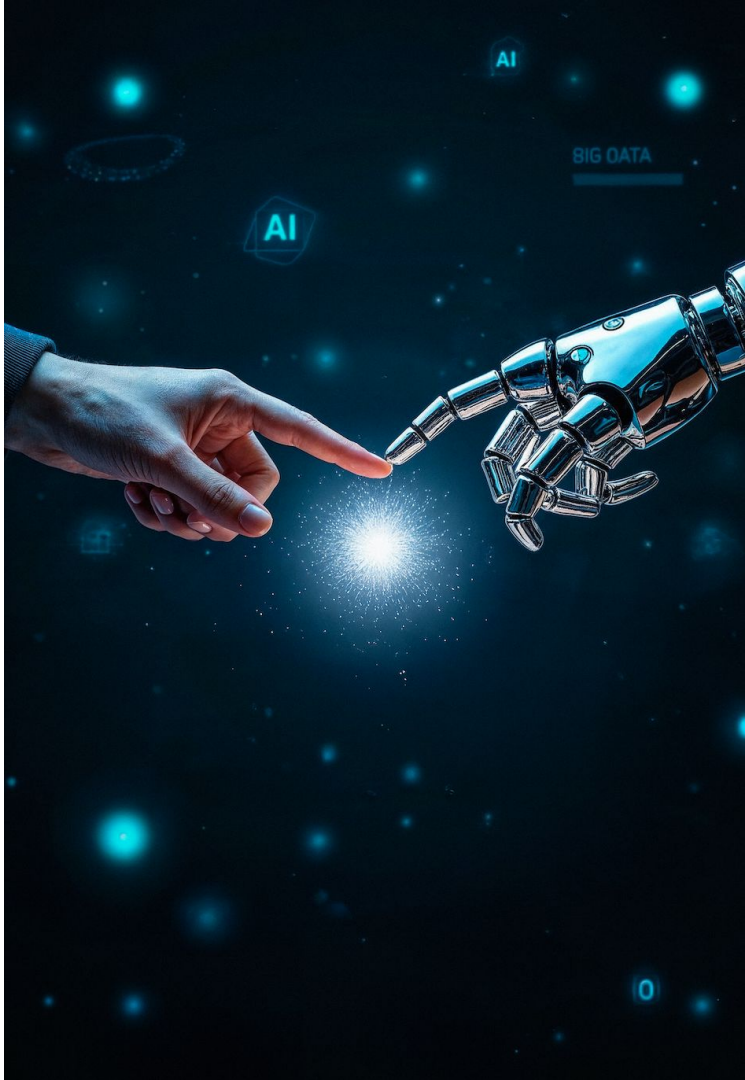
AI-Assisted Software Development

Job Connector Program

What is AI-Assisted Development?

AI-Assisted Software Development is the practice of using Models to accelerate the software lifecycle. It is not about letting AI do your job. It is about shifting your role from **Typing Syntax** to **Designing Logic**.

Think of it like the shift from **"Mental Math"** to using a **"Calculator."** You still need to know what equation to solve, but you no longer need to do the long division by hand.



Shift in Responsibility

In traditional coding, you spend 80% of your time typing boilerplate and 20% solving problems.

Old Way

- Memorizing syntax
- Looking up docs
- Writing code manually

New Way (AI-Assisted)

- Architectural planning
- Reviewing code
- Debugging logic

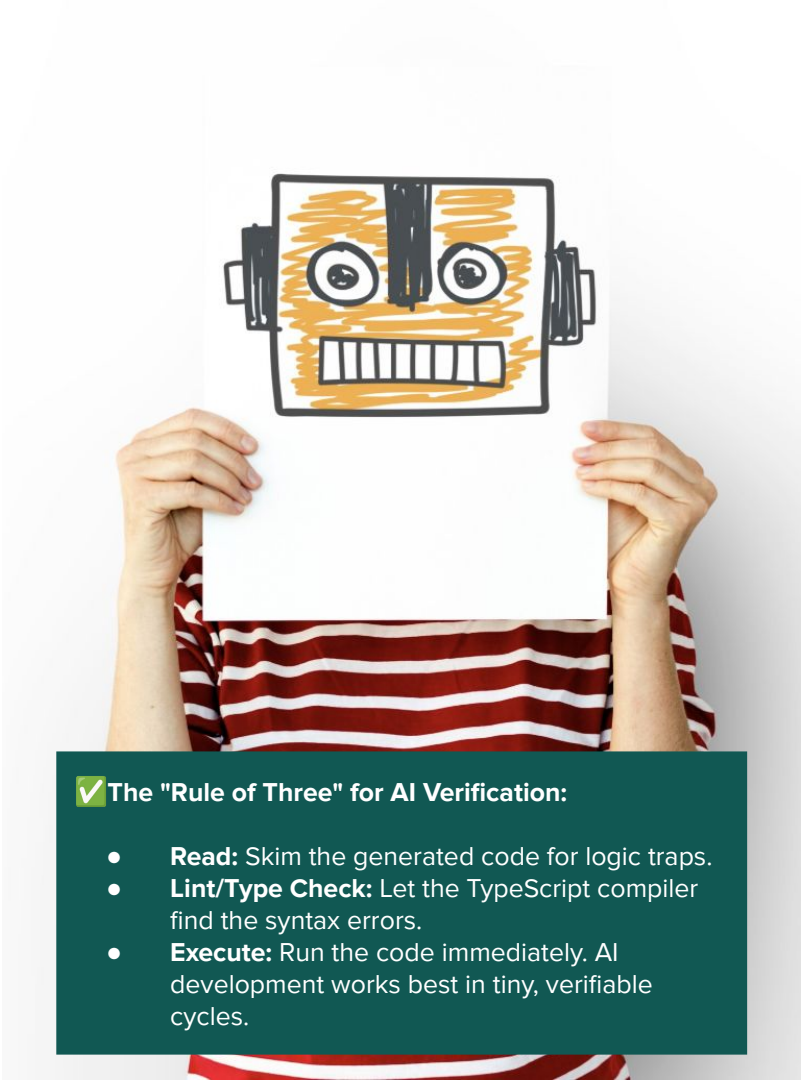
Human in the Loop

AI models are probabilistic, not deterministic. They predict the next likely word based on training data. This means they can:

- **Hallucinate:** Invent libraries that don't exist.
- **Make Security Errors:** Suggest insecure code (e.g., skipping input validation).
- **Lose Context:** Forget previous instructions in long conversations.

You are the Pilot. The AI is the Co-Pilot.

You must never commit code you do not understand or cannot explain.



✅**The "Rule of Three" for AI Verification:**

- **Read:** Skim the generated code for logic traps.
- **Lint/Type Check:** Let the TypeScript compiler find the syntax errors.
- **Execute:** Run the code immediately. AI development works best in tiny, verifiable cycles.

The Tool (Cursor)

We will use **Cursor**, an IDE built specifically for this workflow. It is a fork of VS Code that integrates AI into the core editor.

https://cursor.com/

The 4 Modes of AI in Cursor

You don't just **"chat"**. You assign the AI a specific role based on the complexity of the task.

Plan (The Architect)

- Role: Engineering Manager.
- Power: Read-only. It analyzes your request and creates a step-by-step checklist of how to solve the problem.
- Use Case: "Plan the database schema for a generic e-commerce app." (Always do this before building).

Agent (The Builder)

- Role: Developer.
- Power: Can write code, create/delete files, and run terminal commands autonomously.
- Use Case: "Build this entire Login page component from scratch."

Debug (The Fixer)

- Role: QA Engineer.
- Power: Focuses purely on stack traces, error logs, and logic flaws. It ignores design.
- Use Case: "Why is my server crashing with error 500?"

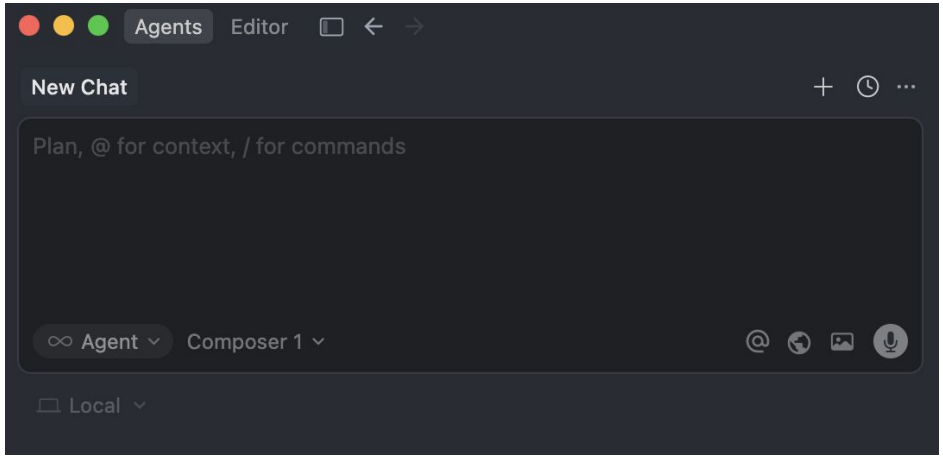
Ask (The Consultant)

- Role: Mentor.
- Power: Standard chat. Answers questions about concepts or libraries.
- Use Case: "What is the difference between interface and type in TypeScript?"

Mastering Cursor Composer

What is Composer?

Composer is Cursor's agentic interface designed for multi-file software engineering. Unlike a standard AI chat that answers questions, Composer acts as an engineer that can plan, edit, delete, and create files across your entire project structure simultaneously.



CURSOR

Mastering Cursor Composer

The Mental Model: Brain vs. Body

To understand Composer, distinct between the Model and the Interface:

- **The Brain (The AI Model):** This is the intelligence (e.g., Claude 3.5 Sonnet, GPT-4o, or the new Cursor 2.0 "Composer" model). It handles logic and reasoning.
- **The Body (Composer):** This is the agent that gives the brain "arms." It allows the AI to reach into your file system, read context, and apply changes directly to the disk.

Essential Shortcuts

- **Ctrl / Cmd + K (Inline Edit):** "Change this specific block of code." (Refactoring, fixing typos).
- **Ctrl / Cmd + L (Composer / Sidebar Chat):** "Talk to the whole codebase." (Architecture, explanation).
- **Tab (Super-Autocomplete):** Cursor predicts your next cursor position, not just the next word. It can write entire functions instantly.

The Mindset (Prompt Engineering)

The Golden Rule:

***"Context is King"**. AI cannot read your mind. It can only read your Context Window.*

The Prompt Framework: *CIC*

To get working code on the first try, use the CIC structure:

- **Context:** "I am using Express, Prisma, and Tailwind..." (Tell it the stack).
- **Instruction:** "Create a function to..." (Tell it the goal).
- **Constraint:** "Use try/catch blocks. Do not use any types." (Tell it the rules).

Practical Workflow

Build a Full Stack "Product Management" App with Cursor

- **Stack:** Express.js, PostgreSQL, Prisma, Vite (React), Tailwind CSS.
- **Tool:** Cursor Editor (features: Plan, Agent, Debug).

Core Philosophy: The Architect vs. The Builder

In this tutorial, **you are the Architect**. You define what needs to be built.

Cursor is the Builder. It writes the syntax, creates the files, and handles the boilerplate.

Phase 1: The Blueprint & Scaffolding

Step 1: Generate the Master Plan

- **Feature Used:** Composer (Cmd+L / Ctrl+L) in **Plan mode**
- **Goal:** Define the master plan in a markdown file, then use that plan to scaffold the folder structure.
- **Why this helps:** You now have a written contract. If the AI gets confused later, you can say ***"Read plan.md and fix this."***

✨**PROMPT**: Create a file named plan.md in the root. Write a comprehensive detailed plan for a **"Product Management Dashboard"** application.

- Tech Stack: Express, Prisma (Postgres), Vite (React), Tailwind CSS.
- Features:
  - Database: Product model (id, name, sku, category, price, stock, status).
  - Backend: REST API with full CRUD endpoints.
  - Frontend: Dashboard with a data table, "Add Product" modal, and "Edit" capabilities.
  - Use typescript as the main programming language.
- Structure: Monorepo (root, /server, /client).
- Implementation Steps: List the steps to build this from scratch.

Phase 2: Setup, Migrate, Run and Bug Fixing

Next steps

- **Install dependencies:** npm install (from root)
- Set up PostgreSQL database and configure **server/.env**
- **Run migrations:** cd server && npx prisma migrate dev
- **Start development:** npm run dev (from root)

Exercise

Implement category management in the product management dashboard.

Thank You!