**:**

**🛠 Cursor AI Best Practices SOP (Optimized for Large Projects)**

**📌 Purpose**

This SOP ensures **efficient and structured use of Cursor AI** for large projects. By leveraging **incremental development, structured project documentation, and .cursorrules integration**, developers can maximize AI-assisted coding accuracy and consistency while minimizing errors and context loss.

**🚀 Setup & Configuration**

**1️⃣ Install & Initialize Cursor**

* Download Cursor from [Cursor Official Site](https://cursor.sh) and sign in.
* Open your project in Cursor and ensure AI features are enabled.
* **Regularly Reindex the Codebase**:
  + **Settings → Cursor Settings → Features → Resync Index**
  + Prevents Cursor from referencing **deleted or moved files**, which can lead to incorrect suggestions.

**2️⃣ Essential Project Files for AI Context Awareness**

To **enhance AI accuracy**, create the following Markdown files in your project root and reference them in .cursorrules.

**📌 Project\_Milestones.md**

* Defines **project phases** and **milestones** with requirements.
* **Ask Cursor to update this file** after each session to track progress.
* **Example Structure:**

markdown

CopyEdit

## Phase 1: Manual Operations & Testing

- Basic API integration

- Database schema setup

- OpenAI integration

- Manual data collection & analysis

## Learnings & Architecture Decisions

### Worker Architecture

\*\*Current Setup:\*\* Single worker handling API calls

\*\*Next Steps:\*\* Split into separate workers:

- `call-worker`: API & data collection

- `analysis-worker`: OpenAI integration

- `api-worker`: Admin dashboard

**📌 Documentation.md**

* Maintain an **updated record of functions, schemas, endpoints, and request/response formats**.
* **Example Structure:**

markdown

CopyEdit

## API Endpoints

- `/user/login` → Authenticates user, returns JWT

- `/posts/get` → Fetches recent posts, paginated

**📌 .cursorrules**

* **Enhance AI adherence** to project-specific requirements.
* **Reference milestone & documentation files** for structured AI assistance.
* **Example Ruleset:**

yaml

CopyEdit

project\_management:

- Reference `Project\_Milestones.md` for feature implementations

- Reference `Documentation.md` for API endpoints & request/response formats

- Ensure new code aligns with defined milestones

- Follow the established database schema

- Consider cost optimizations defined in metrics

- Maintain consistency with existing components

* **Leverage the** [**Awesome Cursor Rules GitHub**](https://github.com/your-link-here) for pre-built rulesets specific to your tech stack.

**🧑‍💻 Optimizing AI-Assisted Workflows**

**✅ Structured Requests for Accuracy**

* **Avoid vague prompts**, use milestone-based requests:  
  ❌ *"Build a dashboard."*  
  ✅ *"Create an admin dashboard with a login system using Next.js. Follow the database schema in Documentation.md."*
* **Break down large features into increments**: *Use Cursor Prompt:*

*"Let’s work on the next milestone."*  
Cursor will **prioritize development tasks** in milestone order.

* **Adopt "Sync" Prompts to Improve AI Awareness**:
  + **Before coding**, use:  
    *"@Take a Breath: Check Project\_Milestones.md and Documentation.md. Ensure everything is updated before proceeding."*
  + **Before switching to a new feature**, use:  
    *"@Get up to speed: Review project context and suggest the next step before coding."*
* **Treat Cursor like an intern, not a senior developer**:
  + Request **features step by step** instead of **entire applications**.

**🔍 Debugging & Fine-Tuning Cursor Output**

**🚨 Debugging Strategy**

* Debug **incrementally** to prevent AI mistakes from cascading:  
  ✅ **Step 1:** Test API requests before integrating UI.  
  ✅ **Step 2:** Validate database queries before connecting endpoints.  
  ✅ **Step 3:** Ensure UI renders correctly before styling.

**🔹 Prevent AI Hallucinations & Outdated References**

* **Reindex Codebase** regularly:
  + **Settings → Cursor Settings → Features → Resync Index**
  + Prevents Cursor from suggesting **deleted or moved functions**.
* **Use an Error Log File to Manage Debugging Context**:
  + Instead of pasting logs into chat, store them in this.log.
  + Reference in chat:  
    *"Check this.log for the latest errors and suggest fixes."*
* **Improve .cursorrules iteratively**:  
  *If AI isn’t following rules, ask:*

*"AI is not adhering to X rule in my .cursorrules file. How can I adjust it for better compliance?"*

**⚡ AI Model Optimization in Cursor**

* **Switch AI models** if responses are inconsistent (Cursor supports GPT-4, Claude, Mixtral).
* **Use structured outputs** for better categorization:

json

CopyEdit

{

"category": "advice\_request",

"confidence": 0.92

}

* **Keep files below 300-500 lines** to prevent AI from overcomplicating solutions.
* **Avoid AI feature creep** – prevent it from deleting or refactoring components **beyond the current task scope**.

**🎨 Enhancing UI with Cursor & Vercel v0**

* **Use Cursor for functionality**, **Vercel v0 for UI design**.
* Provide **clear styling instructions**:  
  ❌ *"Make this look better."*  
  ✅ *"Match the homepage’s UI style, ensuring proper spacing and readable typography."*

**📈 Continuous Improvement & Best Practices**

📌 **Commit working code to Git regularly** – prevent AI from accidentally overwriting good progress.  
📌 **Update .cursorrules and documentation** to reflect evolving project needs.  
📌 **Use Project\_Milestones.md to track progress** and **maintain feature implementation order**.  
📌 **Reindex Cursor frequently** to **prevent outdated AI suggestions**.  
📌 **Leverage Awesome Cursor Rules** for **stack-specific rule enhancements**.  
📌 **Use error logs instead of pasting debug output into chat** to prevent AI from losing project context.

**📌 Summary**

Cursor AI is a **powerful assistant**, but **structured workflows, incremental development, and strong project documentation** are key to ensuring AI-generated code is **efficient, maintainable, and error-free**.

🔹 **Use "sync" prompts to reset AI context.**  
🔹 **Leverage Project\_Milestones.md for incremental development.**  
🔹 **Keep .cursorrules optimized and up-to-date.**  
🔹 **Reindex frequently to prevent outdated references.**  
🔹 **Commit working code regularly to prevent AI from overriding good changes.**

**💡 Following these best practices ensures a productive AI-powered coding experience! 🚀**