

IA para programadores

Fundamentos +
herramientas + prácticas
reales

Lleva tu equipo de “*vibe coding*” a un
proceso ordenado y productivo con
IA integrada.



💡 ¿Por qué IA en desarrollo?

La IA no reemplaza al programador,
lo potencia.

Sin estructura (PRD, SAFe,
agentes, IDE configurado), solo
es “jugar con prompts”.

Objetivo:

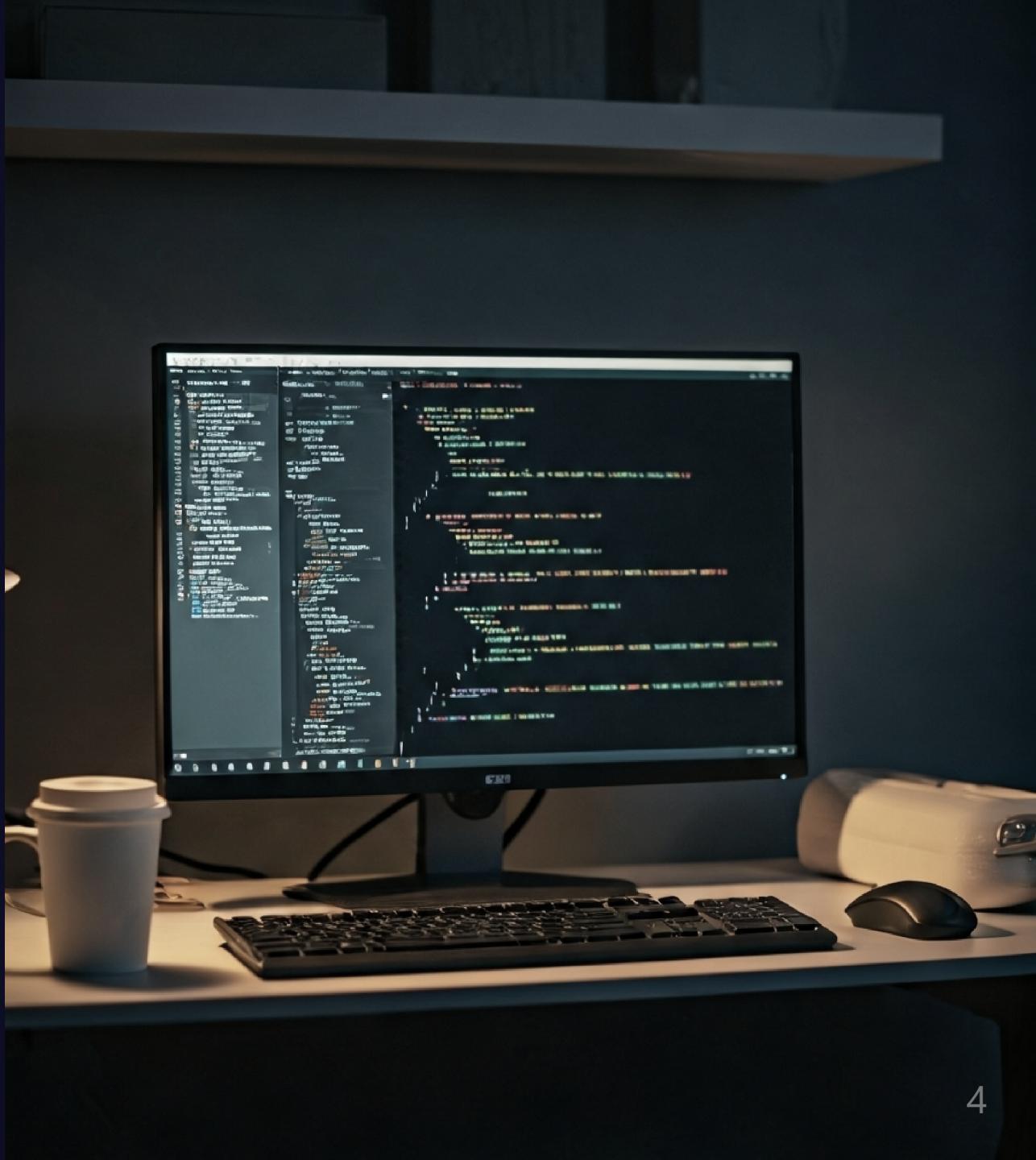
Al final del entrenamiento tendrás un
proceso claro y herramientas prácticas para
integrar IA en tu flujo de trabajo,
mejorando productividad y calidad.



Fundamentos de IA aplicada al desarrollo

Panorama actual de IA en programación

- IA como copiloto, no como reemplazo.
- Productividad vs. dependencia.
- Casos reales:
 - GitHub Copilot
 - ChatGPT
 - Ollama





Conceptos clave en IA para programadores

- LLM (Large Language Models): cómo funcionan en palabras simples.
- Inferencia y endpoints: qué son, cómo se consumen.
- Prompt Engineering: arte de hablarle a la IA para obtener valor real.
- Agentes y orquestación: qué son, por qué importan.

Preparando el terreno



🔧 Set up del entorno local

- **IDE (VS Code):** instalación y extensiones útiles
 - Copilot
 - CodeGPT
 - MCP clients
- **Ollama:** instalar y correr modelos localmente.
- **Integración:** APIs y endpoints de IA.

Buenas prácticas de seguridad y privacidad

- Cuándo usar modelos locales vs. en la nube.
- Gestión de datos sensibles.

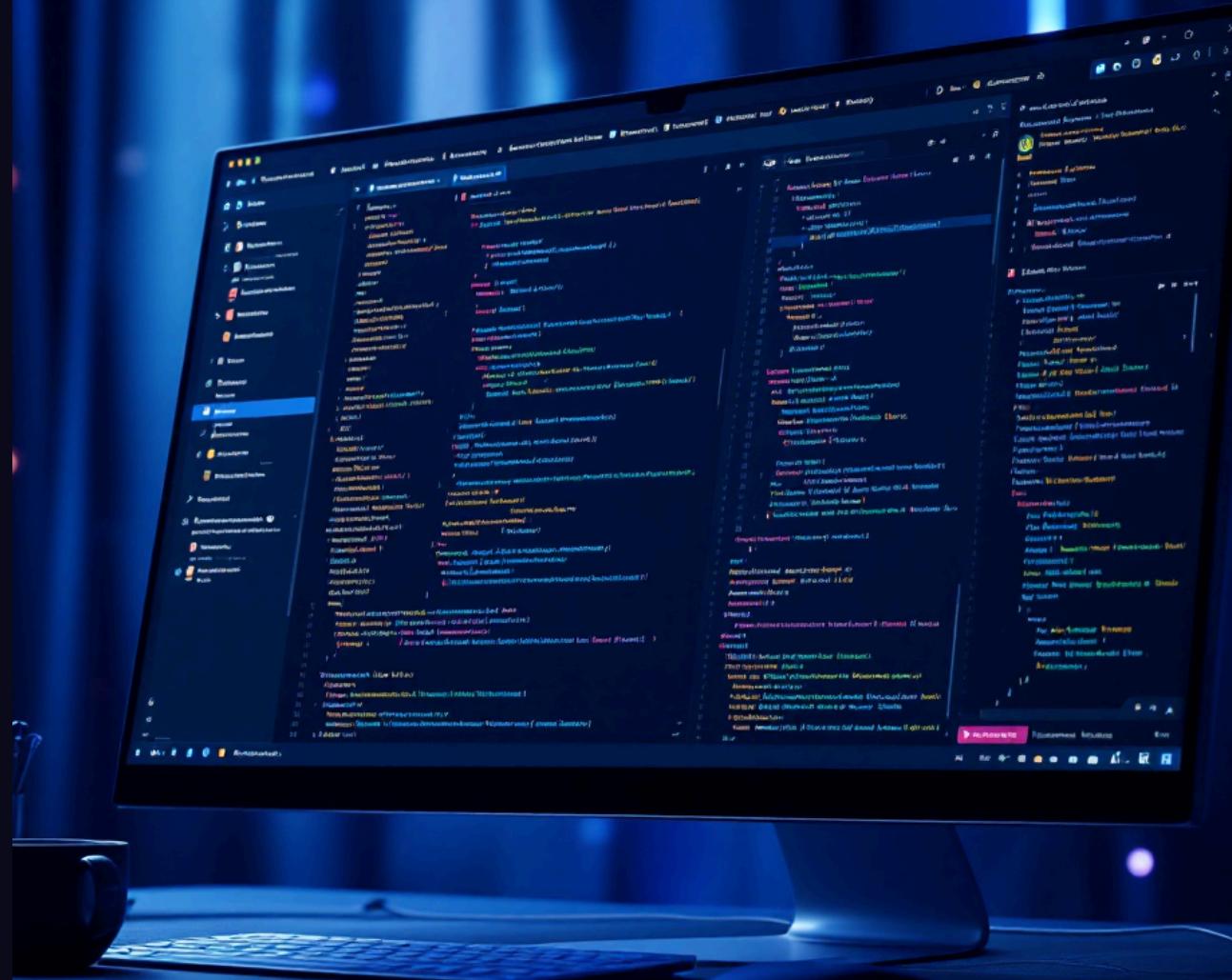


IA dentro del flujo de trabajo



Del PRD al código con IA

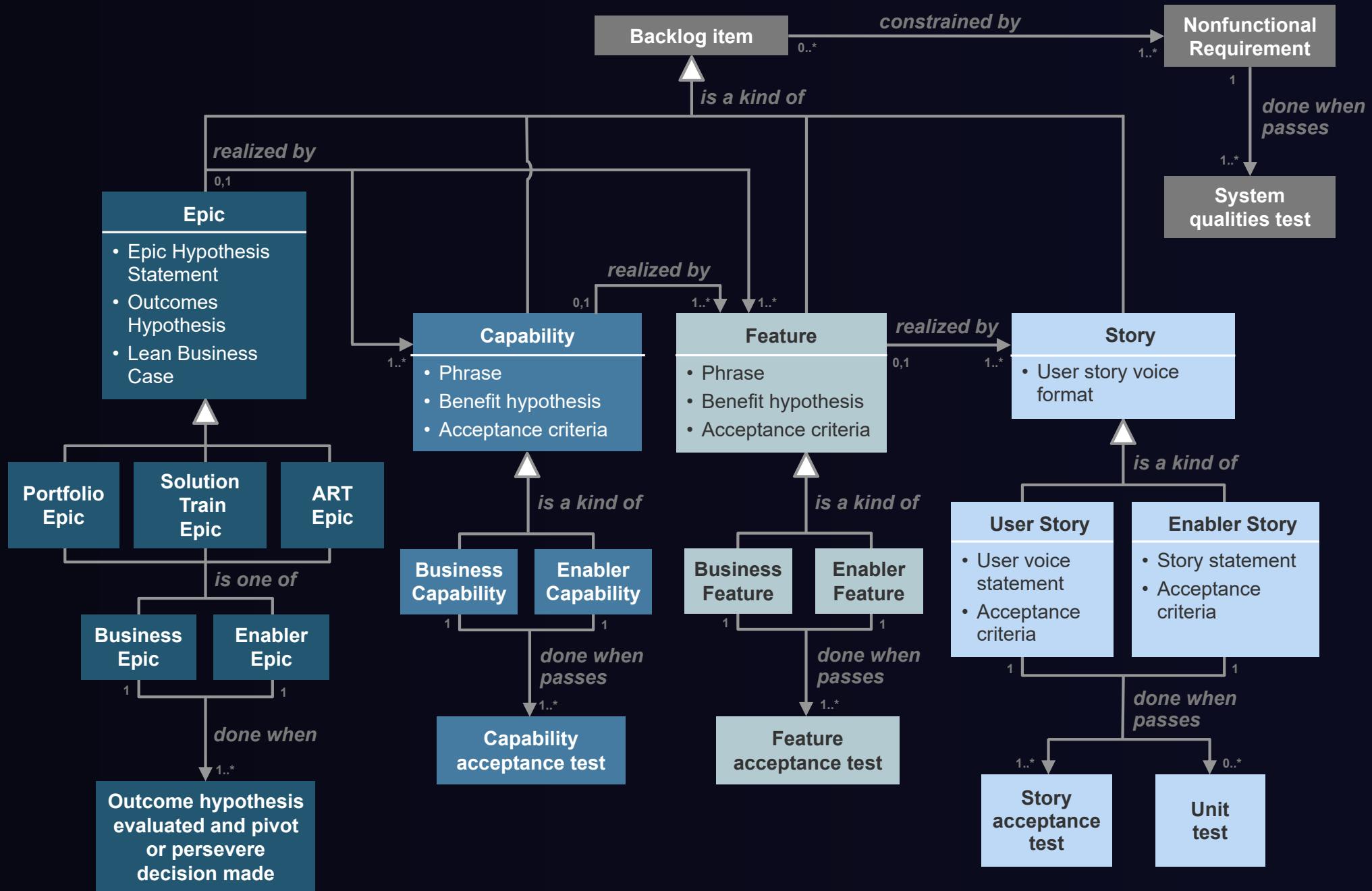
- Repaso del SAFe Requirements Model para proyectos pequeños.
- Traducir un *Product Requirement Document* en tareas técnicas con IA.
- Refinar requerimientos y dividir en historias/tareas.



SAFe Requirements Model

The SAFe Requirements Model is a hierarchical model that defines requirements at the Epic, Feature, and Story levels. It is based on the **INVEST** model for writing good user stories. The SAFe Requirements Model is a living document that will be updated as the product evolves and new requirements are identified.

[SAFe Requirements Model](#)





Aplicando IA en el ciclo de desarrollo

- **Diseño:** arquitecturas, diagramas, documentación.
- **Codificación:** generación, refactorización, optimización.
- **Testing:** pruebas unitarias e integración automáticas.
- **Documentación:** README, comentarios, guías de usuario.

Agentes y MCP

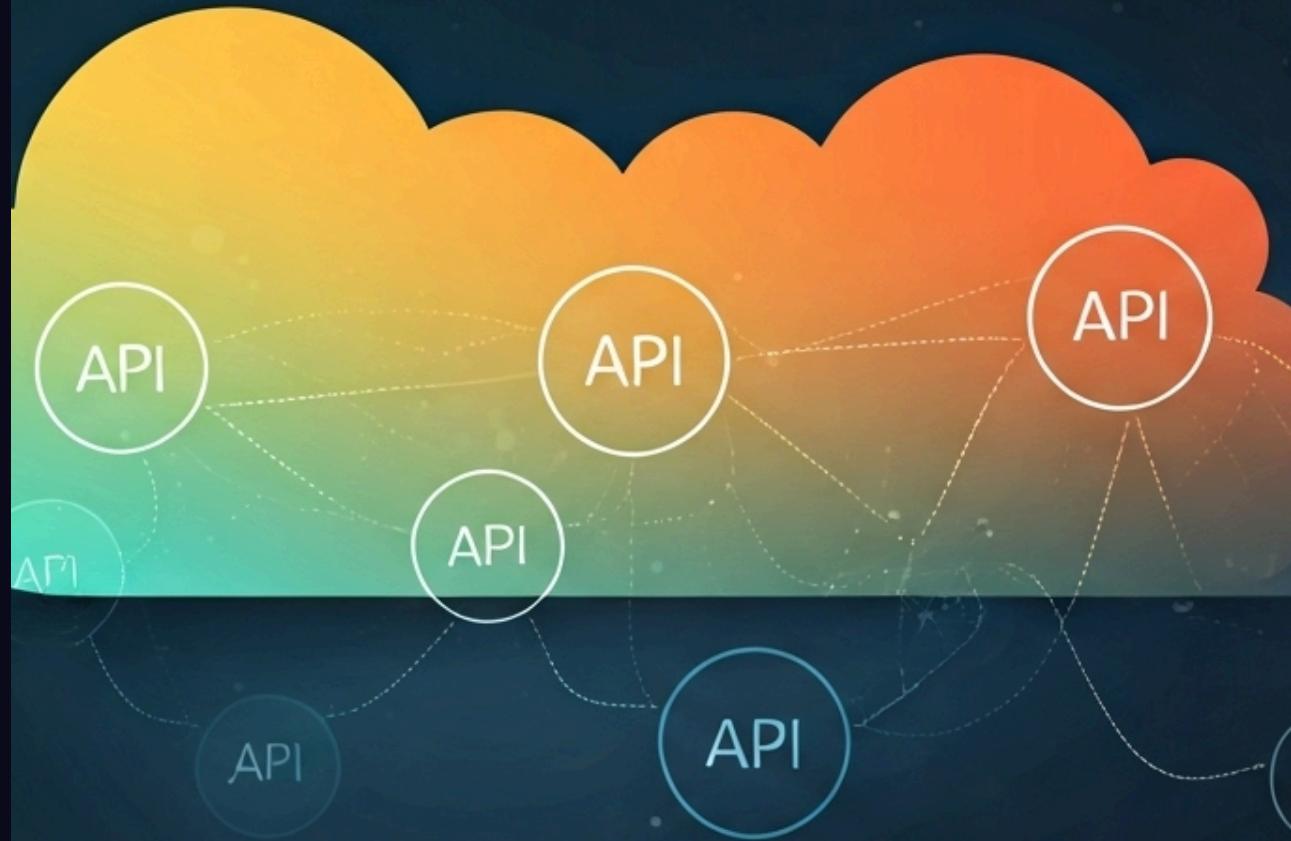


¿Qué son los agentes?

- Diferencia entre LLM básico y agente autónomo.
- Ejemplos:
 - LangChain
 - MCP (Model Context Protocol)
 - A2A (Agent-to-Agent)

🔗 MCP en acción

- MCP es el “Zapier de agentes”.
- Integrar agentes con herramientas y APIs.
- Ejemplo: conectar un LLM con un repositorio de código.



Casos prácticos y workshop



1: PRD → Código base

- Definir un PRD simple (ej. app de notas para vendedores).
- Usar IA para generar el *scaffolding* inicial.



2: Refactorización con IA

- Tomar código desordenado.
- Usar IA para mejorar legibilidad, añadir pruebas y documentación.



3: Agente MCP + IDE

- Configurar un agente que ejecute tareas simples dentro del proyecto.
- Ejemplo: revisar pedidos de mercancía.

Productividad y cultura



🚫 Evita el “vibe coding” con IA

- No le pidas cualquier cosa a la IA sin proceso.
- Adopta estructura:
PRD → tareas → IA como copiloto → validación humana.



Medición del impacto

- Tiempo ahorrado.
- Calidad del código.
- Reducción de bugs.



目 Próximos pasos

- Especialización en agentes.
- Integración con pipelines CI/CD.
- Construcción de framework propio de buenas prácticas con IA.





Recursos

- Documentación oficial:
GitHub Copilot, ChatGPT,
Ollama.
- Tutoriales de MCP y agentes
([HAPI Stack](#)).
- Ejemplos de PRDs y cómo
desglosarlos en tareas técnicas.
- Plantillas de [prompts efectivos](#)
para distintas fases del
desarrollo.





¡Gracias!

La IA es poderosa, pero su verdadero valor se desbloquea cuando se integra de manera estructurada y consciente en el flujo de trabajo del equipo.

¡Potencia tu desarrollo con IA! 