guayerd

Fundamentos IA

Introducción IA y datos

Clase 5

En colaboración con

IBM **SkillsBuild**

¡Bienvenidos!

¿Nos presentamos?

- ¿Qué recuerdan de la clase anterior?
- ¿Qué esperan aprender?
- ¿Tienen alguna pregunta?

guayerd

En colaboración con

IBM **SkillsBuild**

Contenidos

Por temas

05
- Copilot Chat y prompts
- Demo asincrónica

06
- Limpieza y transformación

07
- Estadística aplicada

08
- Visualización

guayerd

En colaboración con

IBM **SkillsBuild**

Objetivos de la clase

- Copilot Chat
- Prompting y escritura efectiva

guayerd

En colaboración con

IBM **SkillsBuild**

Introducción a la IA y los datos

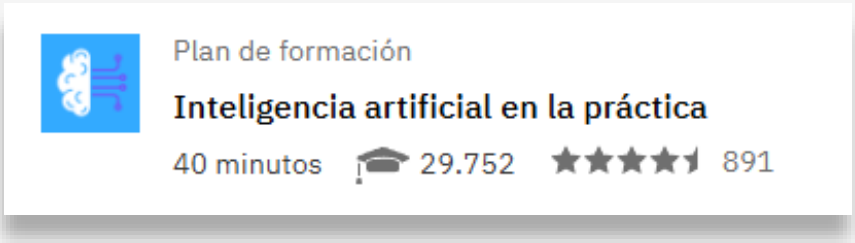
Copilot Chat y prompts

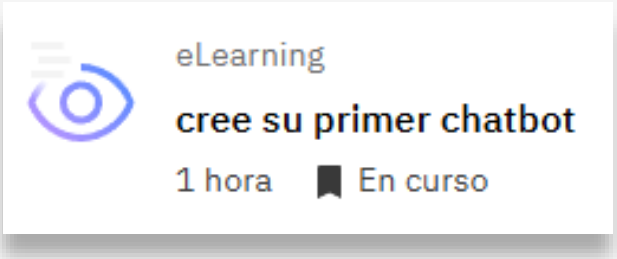
guayerd

En colaboración con

IBM **SkillsBuild**

Plataforma Skill Build: IA generativa





guayerd

En colaboración con

IBM **SkillsBuild**

¿Qué es Copilot Chat?

Herramienta de IA integrada en entornos de desarrollo.

- Asiste en redacción, análisis y programación
- Interactúa mediante lenguaje natural
- Disponible en VS Code, GitHub y otros entornos

Acceso en VS Code

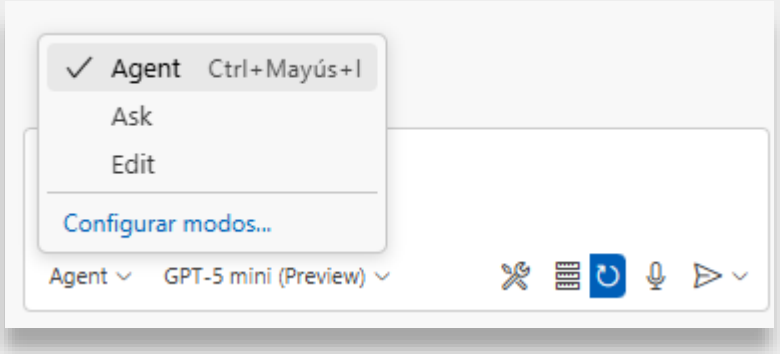
- **Activación:** Panel lateral o ctrl+shift +I
- **Requisitos**
  - Extensión instalada de Copilot
  - Sesión iniciada con cuenta de GitHub

Formas de utilizarlo

- **Autocompletado inteligente**
  - Sugiere código al escribir
  - Uso: Tab para aceptar, Esc para rechazar
  - Alternativas: Alt +] y Alt + [
- **Inline chat**
  - Conversación en el editor
  - Activar: Ctrl + I
  - Ejecutar: Ctrl + Enter
- **Chat completo**
  - Modo Agent, Ask y Edit
  - Activar: Ctrl + Alt + I
- **Acciones rápidas**
  - Icono de bombilla cuando detecta mejoras

Modos de conversación

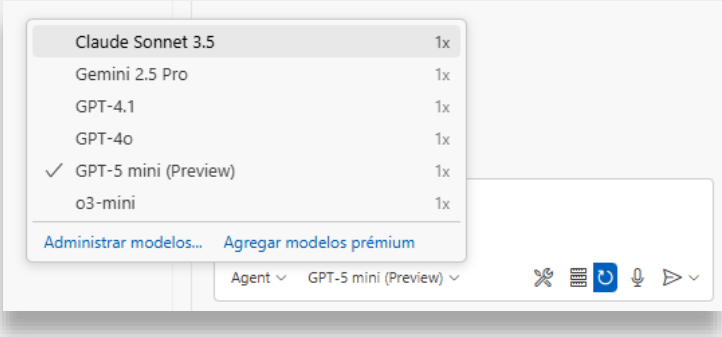
- **Agent:** Tareas complejas y de seguimiento
- **Ask:** Consultas rápidas (modo más común)
- **Edit:** Modificaciones específicas



Se cambian desde el desplegable del chat

Selector de modelos

Elegir el motor de IA (GPT-5 mini, Gemini 2.5 Pro, Claude Sonnet 3.5) según velocidad o profundidad.



Motor de IA

Tabla comparativa de modelos

Modelo	Velocidad	Profundidad	Multimodalidad	Ideal para...
Claude Sonnet 3.5	Media	Muy alta	Solo texto	Análisis profundo, explicaciones complejas, tareas de codificación avanzadas
Gemini 2.0 Flash	Muy alta	Alta	Texto, imagen, audio, video	Interacciones rápidas, agentes conversacionales, tareas multimodales
GPT-4.1	Media	Muy alta	Texto e imagen	Razonamiento complejo, generación de código, documentación técnica
GPT-4o	Alta	Muy alta	Texto, imagen, audio	Conversaciones fluidas, tareas creativas, asistencia técnica con contexto
GPT-5 mini	Alta	Alta	Texto e imagen	Buen balance entre velocidad y calidad, ideal para clases y prototipos
o3-mini	Muy alta	Media	Texto	Consultas simples, respuestas inmediatas, tareas livianas

Comandos

Acciones rápidas sobre el código (/)

- /explain: explica código seleccionado
- /fix: sugiere correcciones para errores
- /test: genera casos de prueba
- /doc: crea documentación
- /optimize: mejora rendimiento del código

Contexto de trabajo (@)

- @workspace: usar todo el proyecto
- @file: solo el archivo actual
- @selection: código seleccionado
- @terminal: interactuar con la terminal
- @vscode: ayuda sobre VS Code

Referencias (#)

- Permiten mencionar issues, pull request o archivos
- Ejemplo: #42 para referirse a un issue

Comandos

Acciones rápidas sobre el código (/)

- /explain: explica código seleccionado
- /fix: sugiere correcciones para errores
- /test: genera casos de prueba
- /doc: crea documentación
- /optimize: mejora rendimiento del código

Contexto de trabajo (@)

- @workspace: usar todo el proyecto
- @file: solo el archivo actual
- @selection: código seleccionado
- @terminal: interactuar con la terminal
- @vscode: ayuda sobre VS Code

Referencias (#)

- Permiten mencionar issues, pull request o archivos
- Ejemplo: #42 para referirse a un issue

Comandos

Acciones rápidas sobre el código (/)

- /explain: explica código seleccionado
- /fix: sugiere correcciones para errores
- /test: genera casos de prueba
- /doc: crea documentación
- /optimize: mejora rendimiento del código

Contexto de trabajo (@)

- @workspace: usar todo el proyecto
- @file: solo el archivo actual
- @selection: código seleccionado
- @terminal: interactuar con la terminal
- @vscode: ayuda sobre VS Code

Referencias (#)

- Permiten mencionar issues, pull request o archivos
- Ejemplo: #42 para referirse a un issue

Comandos

Acciones rápidas sobre el código (/)

- /explain: explica código seleccionado
- /fix: sugiere correcciones para errores
- /test: genera casos de prueba
- /doc: crea documentación
- /optimize: mejora rendimiento del código

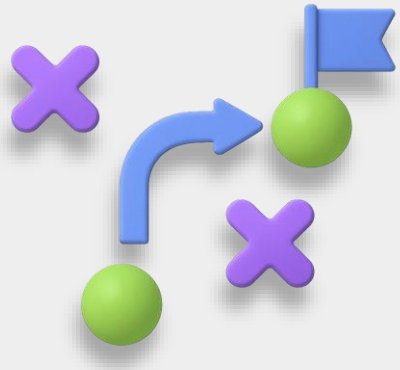
Contexto de trabajo (@)

- @workspace: usar todo el proyecto
- @file: solo el archivo actual
- @selection: código seleccionado
- @terminal: interactuar con la terminal
- @vscode: ayuda sobre VS Code

Referencias (#)

- Permiten mencionar issues, pull request o archivos
- Ejemplo: #42 para referirse a un issue

Contexto

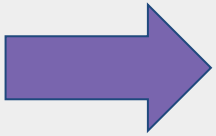


El contexto **es la información que Copilot toma como referencia** para generar respuestas.

- Puede ajustarse con comandos @
- Cuanto más preciso sea el contexto, más relevantes y útiles serán las respuestas

Identifica el comando

- Entender código complejo
- Solicitar ayuda sobre VS Code
- Identificar y corregir un bucle infinito
- Generar casos de prueba
- Buscar en todo el proyecto
- Crear documentación



- /explain
- @
- /fix
- /tes
- @w pace
- /doc

Prompting

Es el proceso de **redactar instrucciones** claras para obtener respuestas útiles de una IA.

- Precisión en el lenguaje = mejores resultados
- Evitar ambigüedades y redundancias
- Usar formato estructurado cuando sea posible
- Incluir contexto relevante

Se recomienda incluir en el proyecto un archivo Markdown (.md) con lineamientos claros sobre cómo debe asistir Copilot

Prompts

Estructura

Contexto + **Objetivo** + **Formato** + **Restricciones**

Contexto	Objetivo	Formato	Restricciones
Ejemplo	**Ejemplo**	**Ejemplo**	**Ejemplo**
“Soy principiante en Python”	“Explica cómo crear una función que calcule promedios”	“Con ejemplos de código comentado”	“Usando sólo conceptos básicos”

Prompts

Tipos

- **Consulta:** “¿Qué es?”, “¿Cómo funciona?”
- **Creación:** “Crea”, “desarrolla”
- **Análisis:** “Analiza este código”, “identifica errores”
- **Optimización:** “Mejora este código”, “sugiere alternativas”
- **Documentación:** “Documenta esta función”, “crea un README”

+info

Prompts

Recomendaciones

Reglas generales

- Ser específico
- Proporcionar contexto
- Definir resultado esperado
- Incluir ejemplos

Mejores prácticas

- **Iterativos:** refinar con nuevas preguntas
- **Chaining:** usar la salida de uno como entrada del siguiente
- **Role prompting:** asignar un rol
- **Constraint prompting:** definir límites

Prompts

Metodologías

PTCF = Persona – Task – Context – Format

Esta técnica ayuda a construir prompts más claros, específicos y útiles para modelos de lenguaje como Copilot, ChatGPT, Claude, etc.

Elemento	Descripción	Ejemplo práctico
Persona	Define el rol que debe asumir el modelo	“Sos profesor de física explicando a estudiantes de ingeniería”
Task	Especifica la tarea que debe realizar el modelo	“Explicá cómo funciona un puente Wheatstone”
Context	Proporciona información adicional relevante para la tarea	“Los estudiantes tienen conocimientos básicos de circuitos”
Format	Indica cómo debe estructurarse la respuesta	“Usá pasos numerados y agregá una analogía visual”

Prompts

Metodologías

PTCF = Persona – Task – Context – Format
Esta técnica ayuda a construir prompts más claros, específicos y útiles para modelos de lenguaje como Copilot, ChatGPT, Claude, etc.

Ejemplo completo usando PTCF

Persona: Sos profesor de física explicando a estudiantes de ingeniería
Task: Explicá cómo funciona un puente Wheatstone
Context: Los estudiantes tienen conocimientos básicos de circuitos
Format: Usá pasos numerados y agregá una analogía visual

Prompts

Metodologías

ROCEF = Rol – Objetivo – Contexto – Ejemplo – Format

Es una estructura de prompting muy poderosa para diseñar instrucciones claras, útiles y bien contextualizadas para modelos de lenguaje. Es especialmente efectiva en entornos educativos, técnicos y colaborativos.

Elemento	Descripción	Ejemplo práctico
Rol	Define el rol que debe asumir el modelo	“Sos profesor de física explicando a estudiantes de ingeniería”
Objetivo	Especifica la tarea que debe realizar el modelo	“Explicá cómo interpretar un histograma”
Contexto	Proporciona información adicional relevante para la tarea	“Los estudiantes están aprendiendo sobre distribución de datos”
Ejemplo	Incluye un caso concreto o muestra de lo que se espera	“Usá un histograma de alturas de estudiantes como ejemplo”
Formato	Indica cómo debe estructurarse la respuesta	“Usá pasos numerados y agregá una analogía visual”

Prompts

Metodologías

ROCEF = Rol – Objetivo – Contexto – Ejemplo – Format
Es una estructura de prompting muy poderosa para diseñar instrucciones claras, útiles y bien contextualizadas para modelos de lenguaje. Es especialmente efectiva en entornos educativos, técnicos y colaborativos.

Ejemplo completo usando ROCEF

Rol: Sos profesor de estadística explicando a estudiantes de ingeniería
Objetivo: Explicá cómo interpretar un histograma
Contexto: Los estudiantes están aprendiendo sobre distribución de datos
Ejemplo: Usá un histograma de alturas de estudiantes como ejemplo
Formato: Usá pasos numerados y agregá una analogía visual

El dominio del prompting determina la efectividad de tu colaboración con IA.

¿Cómo pueden mejorarse estos prompts?

- “Haz una función que calcule algo matemático”
- “Este código tiene errores, ayúdame”
- “Explica machine learning con ejemplos y fácil”

Proyecto

Tienda Aurelion

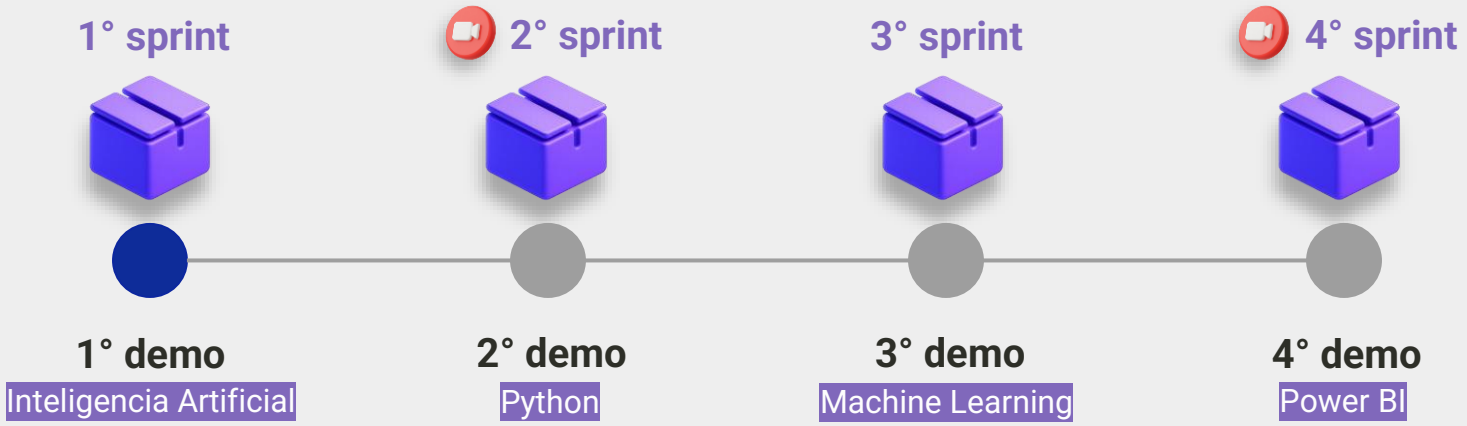
- **Documentación:** notebook Markdown
- **Desarrollo técnico:** programa Python
- **Visualización de datos:** dashboard en Power BI
- **Presentación oral:** problema, solución y hallazgos

guayerd

En colaboración con

IBM **SkillsBuild**

Sprints Project



guayerd

En colaboración con

IBM **SkillsBuild**

Características

1° demo: asincrónica

- **Tema:** Inteligencia Artificial
- **Medio:** Carpeta personal en Drive
- **Fecha límite:** 05/10
- **Devolución:** En salitas por equipo

Contenido

1° demo: asincrónica

Documentación (.md)

- Tema, problema y solución
- Dataset de referencia: fuente, definición, estructura, tipos y escala
- Información, pasos, pseudocódigo y diagrama del programa
- Sugerencias y mejoras aplicadas con Copilot

Programa (.py)

- Debe permitir obtener información del proyecto

Instrucciones (.md)

- Instrucciones para Copilot

Llamando a mi Copiloto

Trabajo en equipo

1. **Configurar Copilot Chat** en VS Code
2. **Crear carpeta del proyecto** en VS Code
3. Crear archivo **Instrucciones.md** con lineamientos
4. **Migrar programa .py** a la carpeta del proyecto
5. Generar **Documentación.md** con ayuda de Copilot
6. **Aplicar mejoras sugeridas por Copilot** en código y documentación

Retro

¿Cómo nos vamos?

- ¿Qué fue lo más útil de la clase?
- ¿Qué parte te costó más?
- ¿Qué te gustaría repasar o reforzar?

guayerd

En colaboración con

IBM **SkillsBuild**