

III GitHub Copilot: Funcionalidades básicas

Versión V1.1

Índice

01 Funcionalidades básicas

Leyenda

A nivel de Contenido



Cuadros de Contexto



Problemas

Texto

Objetivos

Texto

Ejemplos explicativos



Detalles

A nivel de Tipo de Slide



Tipo "Curiosidades"



Tipo "Ejemplo"

Los recursos utilizados se han obtenido de

- <https://storyset.com> : Ilustraciones customizables gratuitas
- <https://lexica.art> : IA que genera imágenes
- <https://www.freepik.es>: Iconos

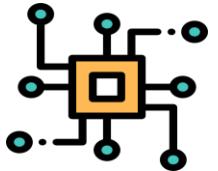
01 Funcionalidades básicas

01

01 Completado de código

01 Funcionalidades
básicas

01.01



¿Qué es el completado de código (code completion)?

Completado de código

Capacidad del asistente de IA para **predecir** y **sugerir** **automáticamente código** mientras el **desarrollador escribe** en **tiempo real**

Características:

- *Genera códigos a partir de comentarios, líneas parciales o nombre de funciones*
- *Trabaja en tiempo real -> sugiere código al momento*
- *Basado en un corpus muy potente*
- *Contextual: se basa en el código antes y después del cursor*
- *Entiende el lenguaje natural*
- *Se adapta y mejora sus sugerencias conforme interactúa*

Objetivo: mejorar la experiencia de desarrollador (acelerar el desarrollo, reducir los errores sintácticos, sugerir buenas prácticas, facilitar tareas repetitivas, enfoque en la lógica del problema, etc.)

Ejemplo: Es como el móvil que te sugiere la siguiente palabra al escribir un mensaje

¿Cómo se usa?

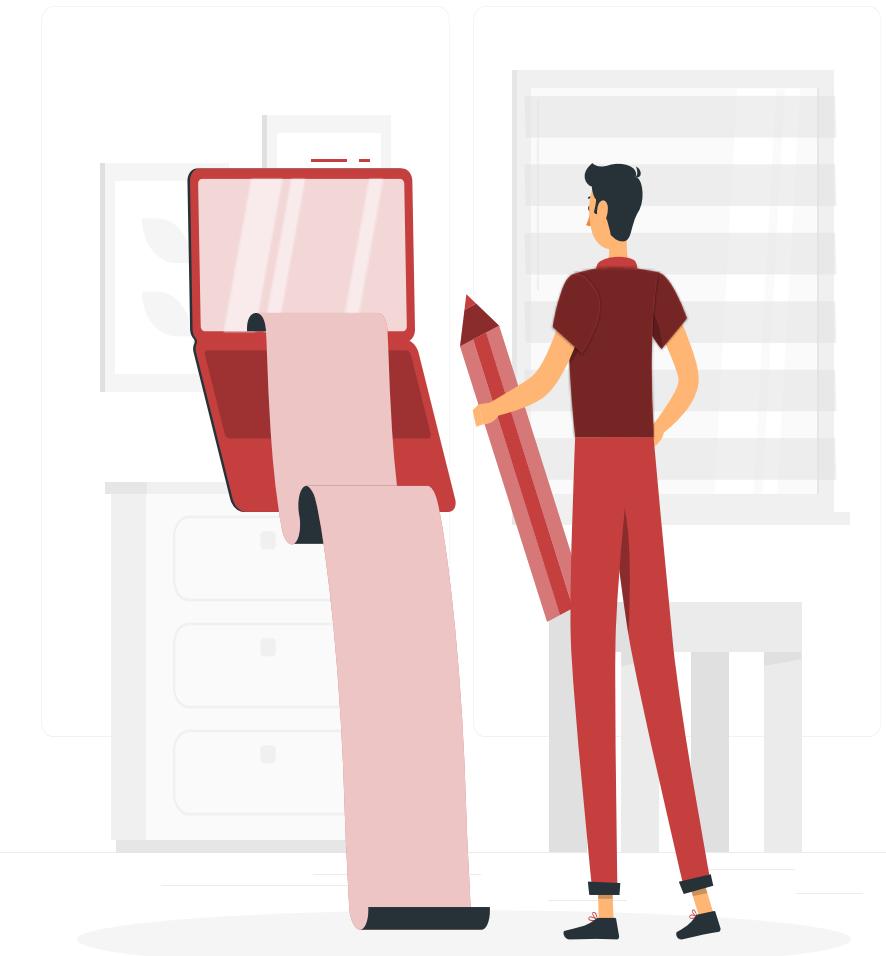
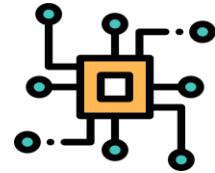
1. Empezar a programar en un IDE compatible con la extensión de Copilot
2. Las sugerencias irán apareciendo a medida que se escribe

Best Practices

- Escribir nombres de variables o comentarios descriptivos para guiar las sugerencias
 - Si los resultados no son adecuados entonces se aconseja volver a refinar
- Usar convenciones en la nomenclatura
- Revisar en detalle las sugerencias antes de aceptarlas

Tipos de completado de código

Funcionalidades principales



Autocompletar en línea (línea o bloque de código)

Autocompletar código repetitivo

Convertir un comentario a código

- Diferentes tipos de estrategias "comment-drive"

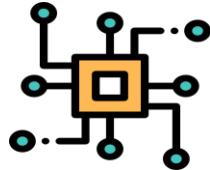
Mostrar alternativas

Completado de archivos enteros (whole file)

...

Atajos de teclado para GitHub Copilot en VSC

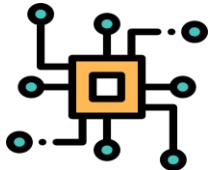
Completado de código



Tarea	Window	macOS	Linux
Aceptar sugerencia de código en línea	Tab	Tab	Tab
Rechazar sugerencia de código en línea	Esc	Esc	Esc
Mostrar siguiente sugerencia	Alt +]	Option +]	Alt +]
Mostrar sugerencia anterior	Alt + [Option + [Alt + [
Aceptar la siguiente palabra en la sugerencia	Ctrl + ->	Ctrl + ->	Ctrl + ->
Activar sugerencia	Alt + \	Option + \	Alt + \
Abrir 10 sugerencias en un panel independiente	Ctrl + Enter	Ctrl + Enter	Ctrl + Enter

El problema de los atajos de teclado

Completado de código



Existen atajos de teclado que pueden no funcionar según el entorno y sobre todo si existen otros atajos de teclado creados o bien usados por otras extensiones.

Anализar
detalladamente
el conflicto

Verificar las
condiciones de
ejecución

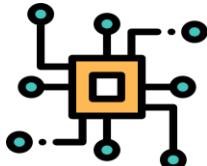
Identificar la
propiedad de la
acción

...

Command	Keybinding	When	Source
.NET: Delete	Delete	focusedView == 'solutionExplorer' && selectedSolutionExplor...	C# Dev Kit
.NET: Remove	Delete	focusedView == 'solutionExplorer' && selectedSolutionExplor...	C# Dev Kit
.NET: Remove	Delete	focusedView == 'solutionExplorer' && selectedSolutionExplor...	C# Dev Kit
.NET: Remove	Delete	focusedView == 'solutionExplorer' && selectedSolutionExplor...	C# Dev Kit
.NET: Rename Solution Folder...	F2	focusedView == 'solutionExplorer' && selectedSolutionExplor...	C# Dev Kit
.NET: Rename...	F2	focusedView == 'solutionExplorer' && selectedSolutionExplor...	C# Dev Kit
Accept Inline Completion	^ ⇧ 7	accessibleViewIsShown && accessibleViewCurrentProviderId ==...	System
Accept Inline Suggestion	Tab	inlineEditisVisible & tabShouldAcceptInlineEdit && !editor...	System
Accept Inline Suggestion	Tab	inInlineEditsPreviewEditor	System
Accept Next Word Of Inline Suggestion	⌘ →	inlineSuggestionVisible && !accessibilityModeEnabled && !ed...	System
Accessible Diff Viewer: Go to Next Difference	F7	isInDiffEditor	System
Accessible Diff Viewer: Go to Previous Difference	⇧ F7	isInDiffEditor	System
Add Cursor Above	⌃ ⌘ ↑	editorTextFocus	System
Add Cursor Below	⌃ ⌘ ↓	editorTextFocus	System
Add Cursors to Line Ends	⌃ ⌘	editorTextFocus	System
Add Line Comment	⌘ K ⌘ C	editorTextFocus && !editor_READONLY	System
Add Selection to Next Find Match	⌘ D	editorFocus	System
Ansible Lightspeed: Inline suggestion accept	Tab	inlineSuggestionVisible && editorLangId == 'ansible'	Ansible
Ansible Lightspeed: Inline suggestion hide	Escape	inlineSuggestionVisible && editorLangId == 'ansible'	Ansible
Ansible Lightspeed: Inline suggestion trigger	⌃ .	config.ansible.lightspeed.enabled && config.ansible.lightsp...	Ansible
AREPL: eval python in real time (current doc)	⌃ ⌘ A	!inQuickOpen && !terminalFocus	AREPL for python
AREPL: eval python in real time (new doc)	⌃ ⌘ R	!inQuickOpen && !terminalFocus	AREPL for python
AREPL: execute the current block of code	⌘ Enter ⌘ E	editorTextFocus && editorLangId == 'python'	User
AREPL: trigger a run in the current AREPL session	⌃ ⌘ .	!inQuickOpen && !terminalFocus && editorLangId == 'python'	AREPL for python
Auto Fix...	⌃ ⌘ .	textInputFocus && !editor_READONLY && supportedCodeAction =~...	System
AWS: Open with Workflow Studio	⌃ ⌘ V	editorTextFocus && isCloud9 && editorLangId == 'asl' edi...	AWS Toolkit
AWS: Sync SAM Application (formerly Deploy)	⌃ ⌘ S	isCloud9 && workspaceContains:/*template.yaml isCloud9 ...	AWS Toolkit
Branches	3	config.gitlens.views.scm.grouped.views.branches && !gitlens...	GitLens — Git superch...
Calls: Show Call Hierarchy	⌃ ⌄ H	editorHasCallHierarchyProvider	Reference Search Vie...

Ejemplo de autocompletado en línea

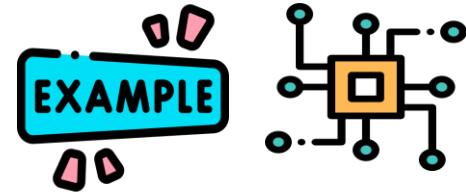
Completado de código



```
JS api.js  U ● JS example.js JS delete.js 1, U ●  
workspace-github-copilot > solutions > JS delete.js > ...  
1  function calculateSum(var1, var2){  
    // Enter your code here  
    const sum = var1 + var2;  
    return sum;  
}  
2  <- #1-5 function calculateSum(var1, var2)
```

Ejemplo de autocompletado desde un comentario

Completo de código



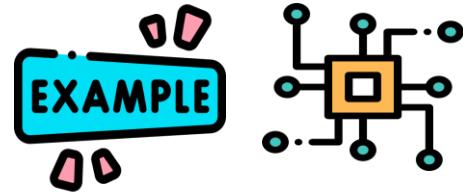
```
JS api.js  U ●   JS delete.js U ●   ⬇ exerc-01-03-comment-to-code.md 2, U

workspace-github-copilot > solutions > JS delete.js

1 // Una función que calcule la suma de dos números
2 function suma(a, b) {
    return a + b;
}
3
```

Ejemplo de autocompletado varias sugerencias

Completo de código



JS example2.js 2, U ● JS example.js Keyboard Shortcuts exerc-01-01-cc ...

workspace-github-copilot > solutions > JS example2.js > ...

```
1 function calculateSecondsBetweenDates
```

GitHub Copilot Suggestions for example2.js

GitHub Copilot Suggestions

10 Suggestions

Suggestion 1

```
function calculateSecondsBetweenDates (date1, date2) {
    // Convertir las fechas a segundos
    const seconds1 = Math.floor(new Date(date1).getTime() / 1000);
    const seconds2 = Math.floor(new Date(date2).getTime() / 1000);

    // Calcular la diferencia en segundos
    return Math.abs(seconds1 - seconds2);
}
```

Accept suggestion 1

Suggestion 2

```
function calculateSecondsBetweenDates (date1, date2) {
    // Convertir las fechas a objetos Date
    const d1 = new Date(date1);
    const d2 = new Date(date2);

    // Calcular la diferencia en milisegundos
    const differenceInMilliseconds = Math.abs(d2 - d1);

    // Convertir milisegundos a segundos
    const differenceInSeconds = Math.floor(differenceInMilliseconds / 1000);

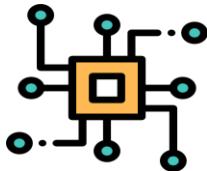
    return differenceInSeconds;
}
```

Accept suggestion 2

02 Copilot Chat

01 Funcionalidades
básicas

01.02



¿Qué es copilot chat?

Copilot chat

Capacidad del asistente de IA para **proporcionar** una **funcionalidad conversacional** en el **editor** facilitando a los **desarrolladores interactuar** en **lenguaje natural** para: resolver dudas, generar código, explicar fragmentos, corregir errores, debatir sobre arquitectura, etc.

Características:

- *Respuestas en lenguaje natural*
- *Resolución de dudas de programación en tiempo real*
- *Contextualización avanzada (código abierto en el editor)*
- *Facilita la asistencia técnica*
- *Soporte didáctico (tutor técnico)*

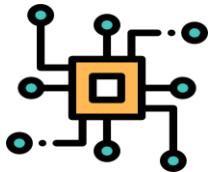
Disponible desde: IDE, GitHub.com, GitHub Mobile y Windows Terminal

Objetivo: mejorar la experiencia de desarrollador (acelerar la resolución de problemas, fomentar el aprendizaje continuo, asistente conversacional, etc.)

Ejemplo: Es como tener un mentor senior a tu lado al que puedes preguntarle lo que quieras sobre el código, en cualquier momento

¿Cómo se usa?

- IDE: Abrir el panel de Copilot Chat (Ver > GitHub Copilot Chat)
- GitHub.com: Hacer clic en el ícono de la burbuja de chat
- GitHub Mobile: Disponible en la app de GitHub Mobile
- Terminal: Usar el CLI de GitHub con la extensión de Copilot



Tipos de Copilot Chat

Copilot chat

```
routes.py 3, M
workspace-ia-lab > sense-api > src > api > examples > routes.py > ...
Víctor Javier Madrid Triviño, 19 months ago | 1 author (Víctor Javier Madrid Triviño)
1 from os import environ
2 import json
3     (import) APIRouter: Any
4 from fastapi import APIRouter, HTTPException, Path
5
6
7 router = APIRouter()
8
9
10    You, 1 second ago * Uncommitted changes
11 @router.get("/one/hello")
12 def read_root():
13     if "HELLOWORLD_ENV" in environ:
14         txt = environ.get("HELLOWORLD_ENV")
15     else:
16         txt = "HELLOWORLD_ENV not found!"
17     return {"HELLOWORLD_ENV": "{}".format(txt): "from /one/hello"}
18
19
20 @router.get("/get_api_key")
21 def read_api_key():
22     api_key = ""
23
24     try:
25         with open("/app/secrets/appconfig.conf") as f:
26             js = json.load(f)
27             api_key = js["api_key"]
28             # Do something with the file
29
30     except IOError:
31         logger.exception(e)
32         print("/app/secrets/appconfig.conf not accessible")
33
34     return {"API_KEY": "{}".format(api_key)}
```

CHAT

Crea un ejemplo de hola mundo en Python

GitHub Copilot

Aquí tienes un ejemplo básico de un programa "Hola, Mundo" en Python:

```
# Hola Mundo en Python
print("Hola, Mundo")
```

Guarda este código en un archivo con extensión .py, por ejemplo, `hola_mundo.py`. Luego, puedes ejecutarlo en la terminal con el siguiente comando:

```
python3 hola_mundo.py
```

Add Context... routes.py Current file

Edit files in your workspace in agent mode

Age... GPT... *

```
5
6
7 router = APIRouter()
8
9
10    You, 1 second ago * Uncommitted changes
11 @router.get("/one/hello")
12 def read_root():
13     if "HELLOWORLD_ENV" in environ:
14         txt = environ.get("HELLOWORLD_ENV")
15     else:
16         txt = "HELLOWORLD_ENV not found!"
17     return {"HELLOWORLD_ENV": "{}".format(txt): "from /one/hello"}
18
```

Ask Copilot

Claude Sonnet 3.7

Víctor Javier Madrid Triviño, 19 months ago • Initial

@router.get("/one/hello")

def read_root():

if "HELLOWORLD_ENV" in environ:

txt = environ.get("HELLOWORLD_ENV")

else:

txt = "HELLOWORLD_ENV not found!"

return {"HELLOWORLD_ENV": "{}".format(txt): "from /one/hello"}

Copilot Chat en línea

Ask Copilot

GPT-4.1

● const Add Context... routes.py Current file

> sense-api > src > api > examples > routes.py > ...

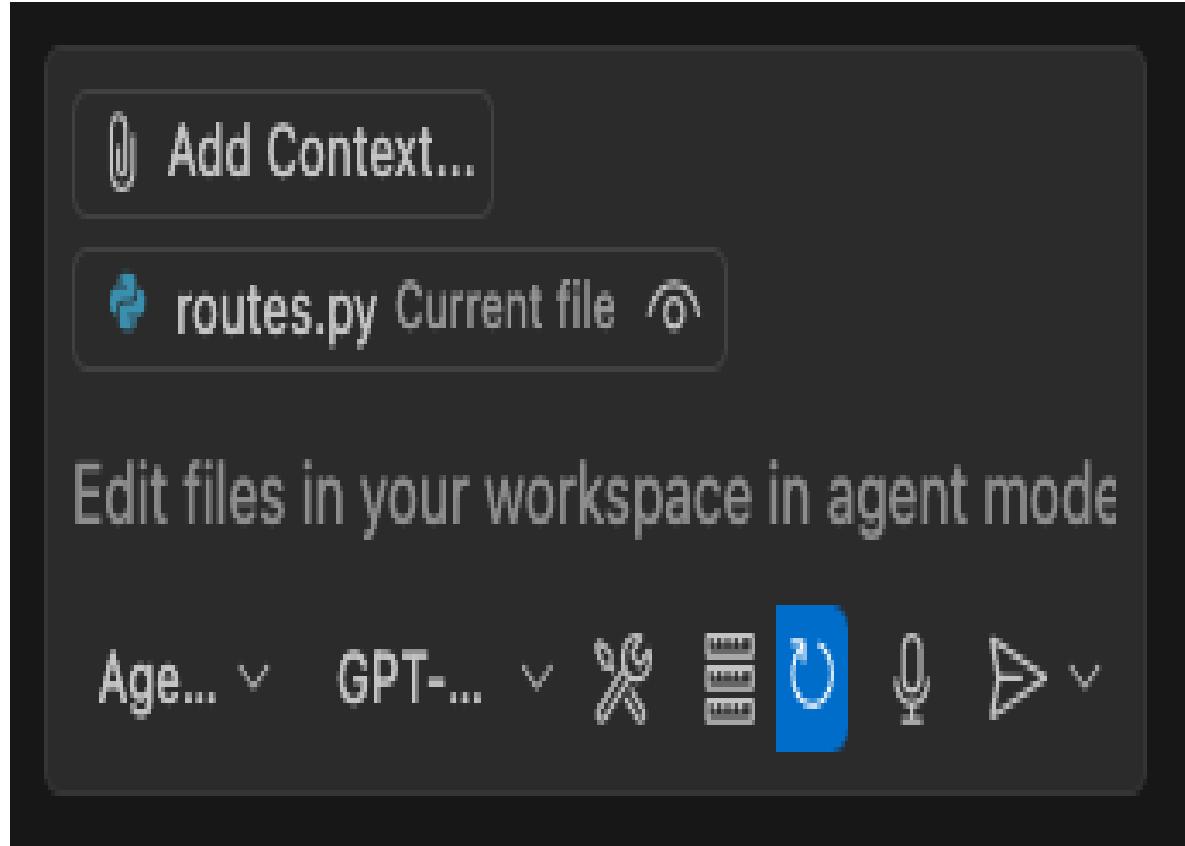
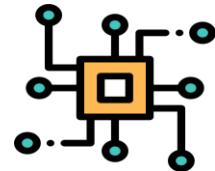
```
fastapi import APIRouter, HTTPException, Path
```

Panel Copilot Chat

Quick Chat

Configuración Copilot Chat

Copilot chat



Modo del Chat (Ask, Edit, Agent y Custom)

- **Ask:** preguntar y para respuestas generales y explicaciones
- **Edit:** editar y para interactuar con el código en el IDE
- **Agent:** tareas complejas que involucran planificación y ejecución de comandos
- **Custom:** personalización del chat

Modelo

- Por defecto
- Premium

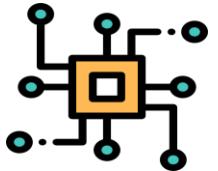
Herramientas

Contexto

- Modo Observación
- Modo Adjunto

Ser ordenado es un premio

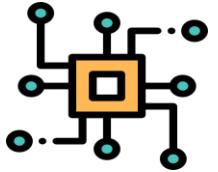
Copilot chat



Se aconseja solamente tener abiertos y/o cargados los archivos que tienen algo que aportar en la sugerencia

Modos Copilot Chat

Copilot chat



Screenshot of the Visual Studio Code documentation page for "Built-in chat modes".

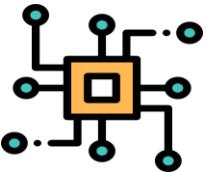
The page is titled "Built-in chat modes" and includes the following content:

- Chat modes**: A table listing three modes:
 - Ask mode**: Optimized for answering questions about codebase, coding, and technology concepts. Use ask mode to understand how a piece of code works, brainstorm software design ideas, or explore new technologies. Open ask mode in [Stable](#) | [Insiders](#).
 - Edit mode**: Optimized for making code edits across multiple files in your project. VS Code directly applies the code changes in the editor, where you can review them in-place. Use edit mode for coding tasks when you have a good understanding of the changes that you want to make, and which files you want to edit. Open edit mode in [Stable](#) | [Insiders](#).
 - Agent mode**: Optimized for making autonomous edits across multiple files in your project. Use agent mode for coding tasks when you have a less well-defined task that might also require running terminal commands and tools. Open agent mode in [Stable](#) | [Insiders](#).
- Custom chat modes**: A section for creating custom chat modes.

The left sidebar shows the navigation menu for the "GITHUB COPILOT" section, including "Chat Modes" which is currently selected.

Funciones

Copilot chat



Preguntas y
respuestas
interactivas (Q&A)

Respuestas
contextuales
(context-aware
responses)

Conversaciones
multivuelta

Explicación de
código

Guía práctica

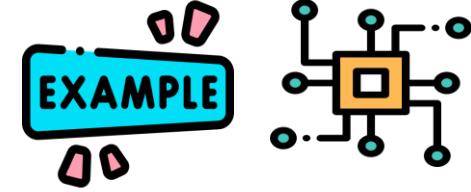
Generación de
pruebas

Ayuda a la
depuración

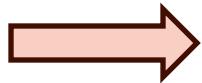
...

Ejemplo Copilot Chat – Panel de Chat

Copilot chat

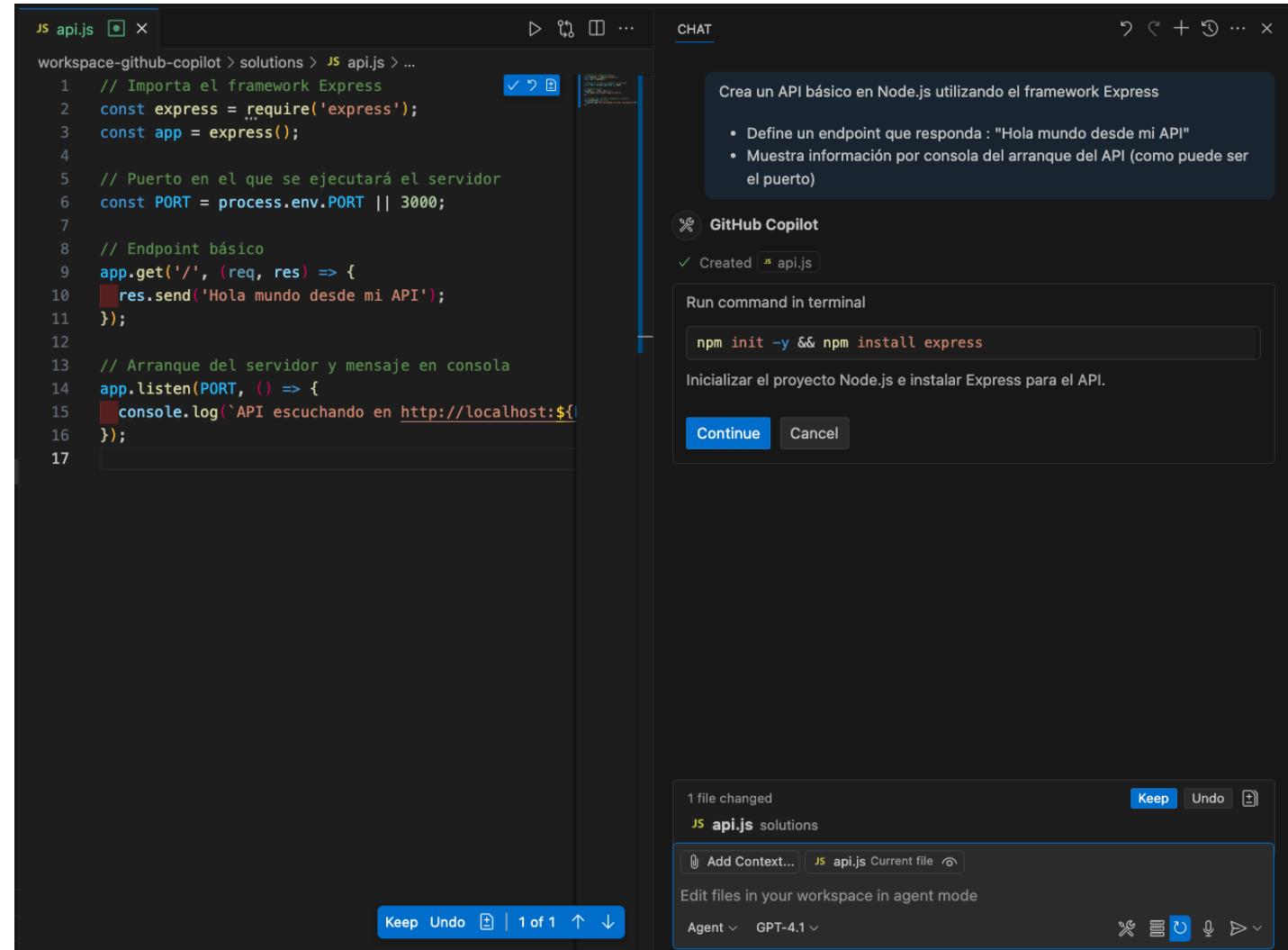


Prompt



Crea un API básico en Node.js utilizando el framework Express

- Define un endpoint que responda : "Hola mundo desde mi API"
- Muestra información por consola del arranque del API (como puede ser el puerto)



The screenshot shows a dark-themed VS Code interface. On the left is a code editor with a file named "api.js" containing Node.js code for an Express API. On the right is the GitHub Copilot Chat panel. The prompt in the Chat panel asks to create a basic API using Express, defining an endpoint to return "Hello world" and logging startup information to the console. Below the Chat panel is a terminal window showing the command "npm init -y && npm install express". At the bottom, there's a workspace summary showing one file changed, and various status icons and buttons like "Keep", "Undo", and "Agent GPT-4.1".

```
js api.js ✘ x workspace-github-copilot > solutions > JS api.js ...  
1 // Importa el framework Express  
2 const express = require('express');  
3 const app = express();  
4  
5 // Puerto en el que se ejecutará el servidor  
6 const PORT = process.env.PORT || 3000;  
7  
8 // Endpoint básico  
9 app.get('/', (req, res) => {  
10   res.send('Hola mundo desde mi API');  
11 });  
12  
13 // Arranque del servidor y mensaje en consola  
14 app.listen(PORT, () => {  
15   console.log(`API escuchando en http://localhost:${PORT}`);  
16 });  
17
```

CHAT

Crea un API básico en Node.js utilizando el framework Express

- Define un endpoint que responda : "Hola mundo desde mi API"
- Muestra información por consola del arranque del API (como puede ser el puerto)

GitHub Copilot

Created JS api.js

Run command in terminal

```
npm init -y && npm install express
```

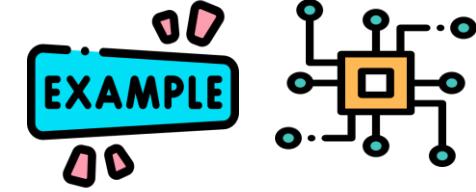
Inicializar el proyecto Node.js e instalar Express para el API.

Continue Cancel

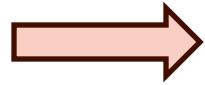
1 file changed
JS api.js solutions
Add Context... JS api.js Current file
Edit files in your workspace in agent mode
Agent GPT-4.1

Ejemplo Copilot Chat – Editor Inline Chat

Copilot chat

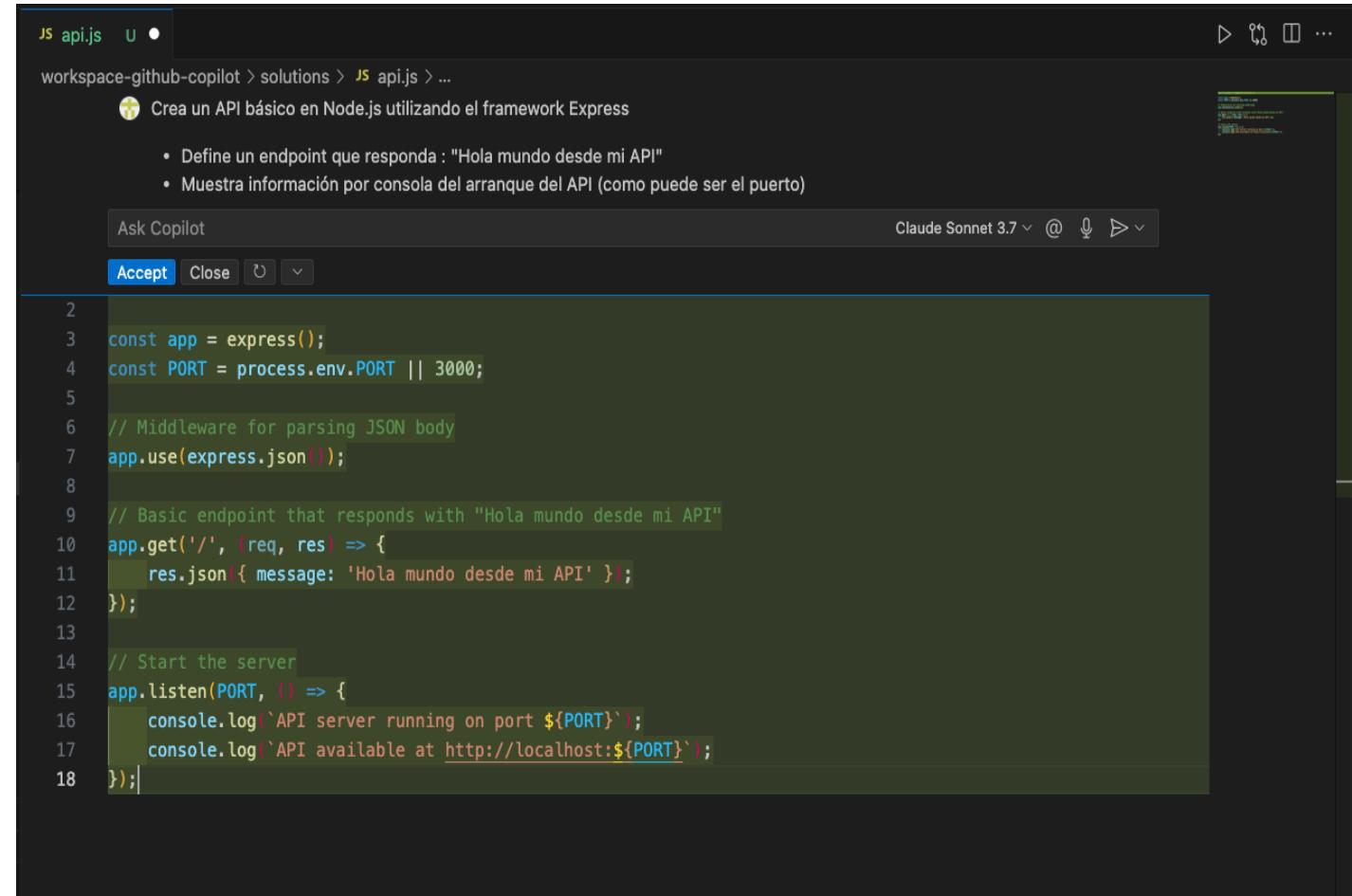


Prompt



Crea un API básico en Node.js utilizando el framework Express

- Define un endpoint que responda : "Hola mundo desde mi API"
- Muestra información por consola del arranque del API (como puede ser el puerto)

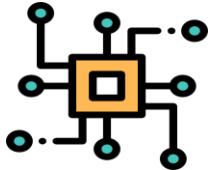


```
JS api.js  U • workspace-github-copilot > solutions > JS api.js > ...
Crea un API básico en Node.js utilizando el framework Express
• Define un endpoint que responda : "Hola mundo desde mi API"
• Muestra información por consola del arranque del API (como puede ser el puerto)

Ask Copilot
Accept Close Claude Sonnet 3.7 @ ▶ ▾
2
3 const app = express();
4 const PORT = process.env.PORT || 3000;
5
6 // Middleware for parsing JSON body
7 app.use(express.json());
8
9 // Basic endpoint that responds with "Hola mundo desde mi API"
10 app.get('/', (req, res) => {
11   res.json({ message: 'Hola mundo desde mi API' });
12 });
13
14 // Start the server
15 app.listen(PORT, () => {
16   console.log(`API server running on port ${PORT}`);
17   console.log(`API available at http://localhost:${PORT}`);
18 })|
```

Mejoras de la conversación con Copilot Chat

Copilot chat



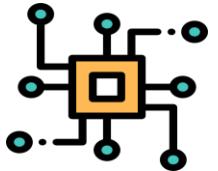
Participantes

Comandos

VARIABLES DE CHAT

Best Practices

Copilot Chat



Buen contexto

Buena pregunta

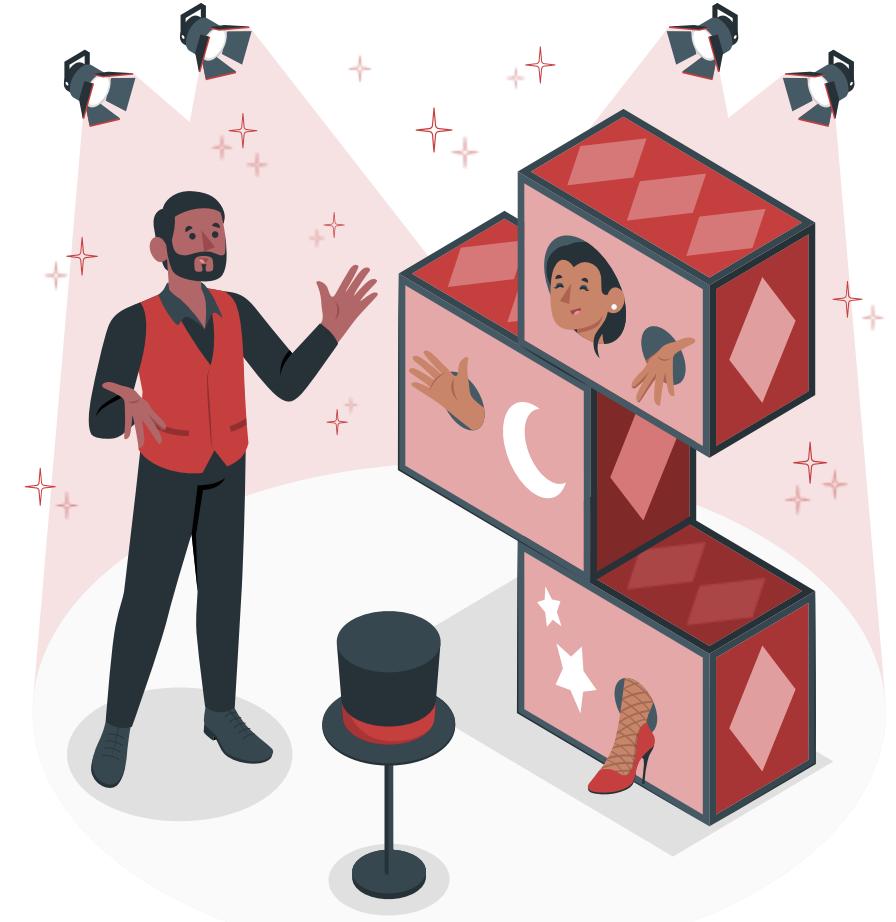
Utilizar adecuadamente: participantes, comandos y variables de chat

Proporcionar contexto en base al código (comentarios, naming, etc)

Proporcionar ejemplos del resultado que se espera

Control del histórico de conversaciones

Eliminar conversaciones "tontas"



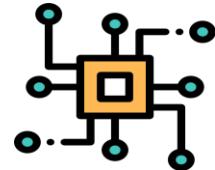
03 Copilot en línea de comandos

01 Funcionalidades básicas

01.03

¿Qué es Copilot en Línea de comandos (Command Line)?

Copilot en línea de comandos



Capacidad del asistente de IA para **añadir** cierta **inteligencia** de Copilot a la **línea de comandos**: asistir en comandos, explicaciones, automatización de tareas y navegación

SIN salir de la consola

Características:

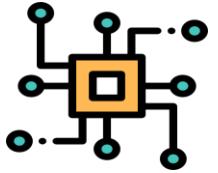
- *Es una extensión*
- *Sugerencias de comandos y argumentos*
- *Soporte didáctico*
- *Contextualización basada en el historial de comandos o código*
- *Compatible con Shell (bash, zsh, etc.)*
- *Etc.*

Objetivo: mejorar la experiencia de desarrollo (reducir la dependencia con el entorno gráfico, proporcionar ayuda contextual directamente desde el terminal, asistir a comandos complejos (Git, Docker, ...), mejorar la eficiencia en tareas tipo scripting / CLI Devops , etc.)

Ejemplo: Es como tener un copiloto en el terminal

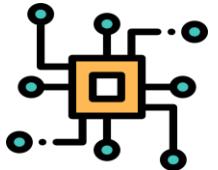
Instalar GitHub CLI

Copilot en línea de comandos



The screenshot shows the GitHub CLI website at cli.github.com. The main heading is "Take GitHub to the command line". Below it, a sub-headline says "GitHub CLI brings GitHub to your terminal. Free and open source." There are two buttons: "brew install gh" and "Download for Mac". A link "View installation instructions →" is also present. At the bottom, there's a terminal window showing the command "\$ gh pr status" and the output "Check on the status of your pull requests." followed by a list of pull requests from the "cli/cli" repository.

- Instalar GitHub CLI: [**https://cli.github.com**](https://cli.github.com)
- Autenticarse: **gh auth login**
- Instalar extensión de CLI para GitHub Copilot: **gh extension install github/gh-copilot**
- Actualizar Copilot en la CLI: **gh extension upgrade gh-copilot**



Acciones permitidas por GitHub CLI

Copilot en línea de comandos

```
› gh copilot
Your AI command line copilot.

Usage:
  copilot [command]

Examples:

$ gh copilot suggest "Install git"
$ gh copilot explain "traceroute github.com"

Available Commands:
  alias      Generate shell-specific aliases for convenience
  config     Configure options
  explain    Explain a command
  suggest    Suggest a command

Flags:
  -h, --help           help for copilot
  --hostname string   The GitHub host to use for authentication
  -v, --version        version for copilot

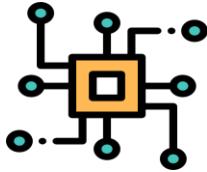
Use "copilot [command] --help" for more information about a command.
```



16:18:49

Línea de comandos (Command Line)

Copilot en línea de comandos



Comando	Propósito	Ejemplo de uso	Salida
gh copilot explain	Entender comandos shell	gh copilot explain "ls -la"	Explicación detallada del comando
gh copilot suggest	Obtener sugerencias de comando	gh copilot suggest "find TODO items"	Comandos shell sugeridos

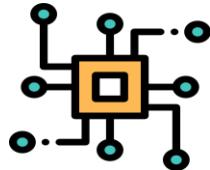
04 Smart Actions

01 Funcionalidades
básicas

01.04

¿Qué es un smart actions?

Smart Actions



Capacidad del asistente de IA para facilitar **sugerencias proactivas** basadas en el **contexto** del **código actual**

Características:

- *Proponen acciones rápidas: refactorizar, generar test, completar función, etc.*
- *Se integra de forma no intrusiva dentro del flujo de trabajo*
- *Adaptativas según el lenguaje, entorno y contexto del código*

¿Cómo se usa?

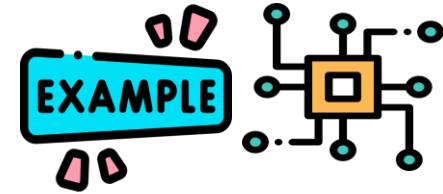
- Identificación en el editor /IDE mediante un ícono de “chispas”
- Pulsar sobre el ícono

Objetivo: mejorar la resolución de tareas de desarrollador (reducir el esfuerzo cognitivo y tiempo en tareas repetitivas o estructurales)

Ejemplo: Es como tener un ayudante que ve que escribes una función y te propone: ¿Genero los test unitarios? o ¿Quieres documentarla?

Ejemplo de smart actions

Smart Actions



```
7  router = APIRouter()
8
9  def get_item(item_id: int = Path(..., title="The ID of the item to get")): You, 4 seconds ago
10
11     Rewrite
12     ✨ Generate Documentation using Copilot
13     ✨ Generate Tests using Copilot
14
15     @ Move
16     🔐 Move symbol to...
17     🔐 Move symbol to new file NV")
18
19     else:
20
21         txt = "HELLOWORLD_ENV not found!"
22
23         return {"HELLOWORLD_ENV: {}".format(txt): "from /one/hello"}
```

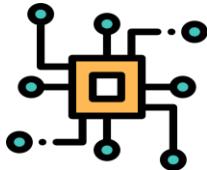
05 Copilot en code reviews

01 Funcionalidades básicas

01.05

¿Qué es Copilot en Revisión de código (Code Review)?

Copilot en code reviews



Capacidad del asistente de IA para **facilitar tareas** de una **revisión de código** desde el IDE o bien desde las pull request de GitHub: analizar cambios, detectar errores, sugerir mejoras, generar comentarios técnicos, etc.

Características:

- *Generación automática de comentarios explicando mejoras o problemas*
- *Sugerencias de refactorización y mejores prácticas*
- *Soporte didáctico para comprender código ajeno*
- *Integración con Copilot Chat*
- *Lectura contextual de cambios de PRs*

Objetivo: mejorar la eficiencia de las revisiones (acelerar el proceso de revisión, mejorar la calidad del código, estandarizar recomendaciones y estilos, ayudar a los desarrolladores con menos experiencia, detectar errores, etc.)

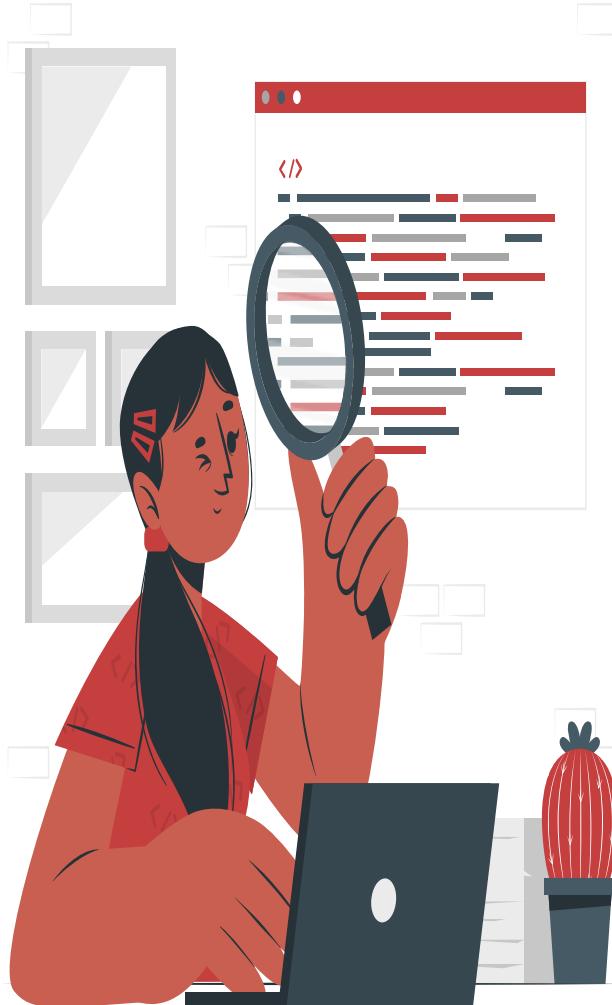
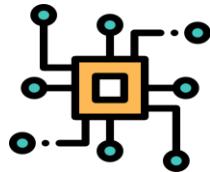
Ejemplo: Es como tener un revisor senior que te va dejando notas sobre las oportunidades de mejora y explicando el por qué

¿Cómo se usa?

- VS Code: Selecciona el código y usa "Copilot: Solicitar revisión de código"
- GitHub PR: Agrega a Copilot como revisor en las solicitudes de extracción
- Revisión específica: Resalta secciones específicas del código para obtener comentarios específicos

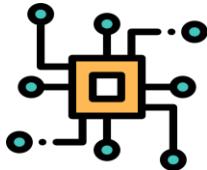
Tipos de revisión de código

Copilot en code reviews



Funcionalidad	Disponibilidad	Acción Requerida	Salida
Comentarios en línea	VS Code, GitHub PR	Seleccionar código + solicitar revisión	Sugerencias específicas con explicaciones
Aplicar cambios	VS Code, GitHub PR	Clic en el botón "Aplicar cambio"	Correcciones con un solo clic
Analisis de seguridad	Todas las plataformas	Automático durante la revisión	Advertencias de vulnerabilidad
Consejos de rendimiento	Todas las plataformas	Automático durante la revisión	Sugerencias de optimización

Ejemplo de revisión de código



Copilot en code reviews

```
14 @router.get("/one/hello")      Víctor Javier Madrid Triviño, 19 months ago • Initial  
15 def read_root():  
16     if "HELLOWORLD_ENV" in environ:  
17         txt = environ.get("HELLOWORLD_ENV")  
18     else:  
19         txt = "HELLOWORLD_ENV not found!"  
20     return {"HELLOWORLD_ENV": "{}".format(txt)}: "from /one/hello"
```

Code Review Comment (1 of 1)



GitHub Copilot



The returned dictionary key "HELLOWORLD_ENV: {}".format(txt) includes a colon and space, which is unconventional for JSON keys. Consider using "HELLOWORLD_ENV": txt for clarity and consistency.

Suggested change:

- ```
- return {"HELLOWORLD_ENV: {}".format(txt): "from /one/hello"}
+ return {"HELLOWORLD ENV": txt}
```

## 06 Pull Request Summaries

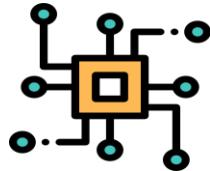
---

01 Funcionalidades  
básicas

01.06

# Resúmenes de Pull Request (Pull Request Summaries)

## Pull Request Summaries



**Capacidad** del asistente de código que **facilita generar automáticamente descripciones completas de solicitudes de incorporación de cambios (PR)**. Ahorrando tiempo y mejorando la documentación

Genera resúmenes automáticos y claros de los cambios realizados en un pull request

Características:

- *Genera descripciones en lenguaje natural del contenido de la PR*
- *Aumenta la comprensión del código revisado sin leer línea por línea*
- *Se integra directamente en GitHub dentro del flujo de revisión*
- *Puede actualizarse automáticamente conforme cambian los commits*

**Objetivo:** Ahorra tiempo a revisores y autores al ofrecer un resumen comprensible, facilitando decisiones rápidas sobre aprobación o cambio necesarios.

**Ejemplo:** Es como tener un “traductor de código a lenguaje humano” que resume el propósito y efecto de los cambios en una PR

knowmad mood



Spain · Portugal · Italy · United Kingdom · United States · Uruguay. Morocco

Email: [example@knowmadmood.com](mailto:example@knowmadmood.com)  
[www.knowmadmood.com](http://www.knowmadmood.com)