



Generación de código en Angular con GitHub Copilot

¡Bienvenido al curso de Generación de código en Angular con GitHub Copilot!

El objetivo de este curso es proporcionarte las herramientas necesarias para que, mediante **prompts e interacciones** con GitHub Copilot, puedas comenzar a generar código de manera eficiente en tu día a día. Esta capacitación consta de cuatro módulos diseñados para guiarte **desde la creación de un proyecto hasta la integración completa de servicios y componentes**.

Prerrequisitos:

Para aprovechar al máximo este curso, necesitas:

- Tener una cuenta de GitHub Copilot.
- GitHub Copilot instalado y funcionando en tu entorno de desarrollo.
- Visual Studio Code (VSCode) como tu IDE de desarrollo.
- Familiaridad con el Framework Angular.
- Ejecutar el servidor de simulación de api customer, según pasos del modulo 0

Instrucciones:

1. Crea un proyecto y levanta el servidor de simulación de api según el Módulo 0.
2. Navega por los módulos 1, 2 y 3 y sigue el paso a paso.
3. Carga las evidencias en la evaluación para aprobar el curso.

^ Estructura del Curso

Módulo 0: Creación de Proyecto de Ejemplo

Objetivo: Aprender a generar un proyecto Angular desde cero usando GitHub Copilot.

Contenido:

1. **Generación del proyecto Angular:** Aprende a usar el comando `ng new` para crear un nuevo proyecto Angular con la ayuda de GitHub Copilot.
2. **Interacción con comandos:** Descubre cómo insertar comandos directamente en la terminal desde GitHub Copilot para agilizar tu flujo de trabajo.

Módulo 1: Generar un Servicio con GitHub Copilot

Objetivo: Crear y consumir servicios desde un proyecto Angular utilizando GitHub Copilot.

Contenido:

1. **Generación de servicios:** Aprende a generar un servicio en Angular con el comando `ng generate service`.
2. **Consumo de APIs:** Utiliza GitHub Copilot para escribir el código necesario para consumir una API, creando modelos de datos y manejando las respuestas del servidor.

Módulo 2: Trabajo en Componentes con GitHub Copilot

Objetivo: Crear y gestionar componentes en Angular utilizando GitHub Copilot.

Contenido:

1. **Generación de componentes:** Crea componentes en tu proyecto Angular utilizando prompts específicos.
2. **Integración de routing:** Aprende a incorporar componentes en el routing de tu aplicación y a consumir servicios desde los componentes.
3. **Refactorización de código:** Utiliza GitHub Copilot para mejorar y refactorizar el código de tus componentes.

Módulo 3: Trabajo en HTML con GitHub Copilot

Objetivo: Generar y estilizar código HTML en tus proyectos Angular con la ayuda de GitHub Copilot.

Contenido:

1. **Generación de vistas:** Aprende a mostrar datos en tus componentes utilizando tablas y otros elementos HTML.
2. **Estilización:** Crea estilos personalizados para tus componentes usando prompts de GitHub Copilot.

Prepárate para descubrir cómo generar código de manera más rápida y eficiente.

¡Comencemos!

Módulo 0: Creación de proyecto de ejemplo

^ Generando un proyecto angular


Explicación paso a paso:

- Abre Visual Studio Code y asegúrate de que GitHub Copilot esté activado.
- Escribe el prompt en el chat de GitHub Copilot.
- GitHub Copilot te proporcionará una explicación del comando `ng new` y cómo usarlo.
- Selecciona "Insert Into Terminal" para ejecutar el comando directamente en la terminal y generar un nuevo proyecto Angular.

Existen varias formas de interactuar con GitHub Copilot. La más sencilla es cuando nos proporciona información sobre un comando o un marco de trabajo específico, como Angular. Esto significa que podemos obtener ayuda directa sobre cómo usar comandos o características de Angular.


Por ejemplo, si trabajaremos en un proyecto de Angular y necesitamos recordar cómo se utiliza el comando `ng new` para generar un nuevo proyecto, GitHub Copilot puede proporcionarnos esa información de manera instantánea mediante su chat con el siguiente prompt.

```
explícame cómo funciona el comando ng new
```



id24134-corporativo_sangroup

explicame como funciona el comando ng new



GitHub Copilot

El comando `ng new` es un comando de Angular CLI que se utiliza para crear un nuevo proyecto de Angular. Aquí está el desglose de cómo funciona:

- `ng`: Es el comando de la interfaz de línea de comandos de Angular (CLI). Se utiliza para interactuar con Angular.
- `new`: Es un subcomando de Angular CLI que indica que quieres crear un nuevo proyecto.
- `ng new my-app`: Este comando crea un nuevo proyecto de Angular con el nombre "my-app".

El comando `ng new` también tiene varias opciones que puedes usar para personalizar la creación de tu proyecto. Algunas de las más comunes incluyen:

- `--style`: Permite especificar el preprocesador de hojas de estilo que quieres usar (css, scss, sass, less, styl).
- `--routing`: Genera un módulo de enrutamiento para el proyecto.
- `--skip-tests`: No genera archivos de prueba para el nuevo proyecto.



Por ejemplo, si quieres crear un nuevo proyecto de Angular con SCSS como preprocesador de hojas de estilo y un módulo de enrutamiento, puedes usar el siguiente comando:

```
ng new my-app --style=scss --routing
```

Este comando creará un nuevo proyecto de Angular con SCSS y un módulo de enrutamiento.

Copilot te proporcionará la información necesaria, y podrás optar por copiar el comando o insertarlo directamente en la terminal para crear tu nuevo proyecto.

Esta respuesta nos permite interactuar directamente con el comando para poder hacer uso de este mediante las opciones:

	Copy	Copiará el comando en la papelera para su posterior uso
	Insert Into Terminal	insertará en comando directamente en la terminal para su uso inmediato

Al insertar el comando en la terminal podremos generar nuestro nuevo proyecto.

```
id24134@VIRT103054 MINGW64 /c/copilot/taller
$ ng new my-app --style=scss --routing
CREATE my-app/angular.json (3207 bytes)
CREATE my-app/package.json (1070 bytes)
CREATE my-app/README.md (1052 bytes)
CREATE my-app/tsconfig.json (783 bytes)
CREATE my-app/.editorconfig (274 bytes)
CREATE my-app/.gitignore (604 bytes)
CREATE my-app/browserlistrc (703 bytes)
CREATE my-app/karma.conf.js (1423 bytes)
CREATE my-app/tsconfig.app.json (287 bytes)
CREATE my-app/tsconfig.spec.json (333 bytes)
CREATE my-app/src/favicon.ico (948 bytes)
CREATE my-app/src/index.html (291 bytes)
CREATE my-app/src/main.ts (372 bytes)
CREATE my-app/src/polyfills.ts (2820 bytes)
CREATE my-app/src/styles.scss (80 bytes)
CREATE my-app/src/test.ts (788 bytes)
CREATE my-app/src/assets/_gitkeep (0 bytes)
CREATE my-app/src/environments/environment.prod.ts (51 bytes)
CREATE my-app/src/environments/environment.ts (658 bytes)
CREATE my-app/src/app/app-routing.module.ts (245 bytes)
CREATE my-app/src/app/app.module.ts (393 bytes)
CREATE my-app/src/app/app.component.html (24617 bytes)
CREATE my-app/src/app/app.component.spec.ts (1073 bytes)
CREATE my-app/src/app/app.component.ts (211 bytes)
CREATE my-app/src/app/app.component.scss (0 bytes)
✓ Packages installed successfully.
```

Y así con ayuda de GitHub Copilot podríamos generar un proyecto en angular para trabajar.

^ Levantar servidor para simulación de API

para levantar el servidor simulado de api customer tienes 2 opciones

- Opción 1: Para versiones de angular 15 en adelante puedes usar un json-server

1. Ejecuta el comando para crear el proyecto

```
ng new apiserver
```

2. Ejecuta el comando en la carpeta del proyecto

```
npm i json-server
```

3. Crea en la raíz del proyecto el archivo db.json con la siguiente estructura:

```
{
  "customers": [
    { "idCliente": "876543215", "name": "David Brown", "mail": "David@mail.com", "phone": "1234567890", "address": "123 Main St" },
    { "idCliente": "123456789", "name": "John Doe", "mail": "John@mail.com", "phone": "0987654321", "address": "456 Elm St" },
    { "idCliente": "987654321", "name": "Jane Smith", "mail": "Jane@mail.com", "phone": "6789012345", "address": "789 Oak St" },
    { "idCliente": "543216789", "name": "Mary Johnson", "mail": "Mary@mail.com", "phone": "5432167890", "address": "987 Pine St" }
  ]
}
```

4. Levanta el servidor con el comando

```
json-server --watch db.json
```

5. El endpoint quedará disponible en

```
http://localhost:3000/customers/
```

- Opción 2: Puedes descargar y ejecutar este proyecto en express

1. Descarga el siguiente proyecto

[api-mock-node.zip](#)

2. Una vez descomprimido en tu equipo ejecuta el comando

```
node server-node-api.js
```

3. El endpoint quedará disponible en

```
http://localhost:3000/customers/
```

Módulo 1: Generar un servicio con GitHub Copilot

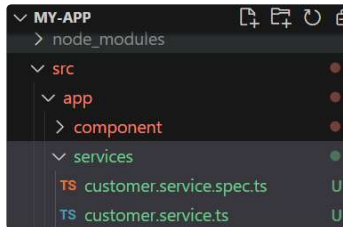
^ Creación de un servicio

Explicación paso a paso:

- Escribe el comando en la terminal de VSCode.

- GitHub Copilot generará un nuevo servicio en la estructura app/services.
- Este servicio se utilizará para consumir una API externa.

Para consumir una api desde un servicio, podemos generar código mediante prompts de GitHub Copilot. Previamente debemos generar un servicio en nuestro proyecto mediante el comando *ng generate service services/customer*, esto nos creara un servicio en la estructura app/services.



^ Creación de modelo a partir de Json

Explicación paso a paso:

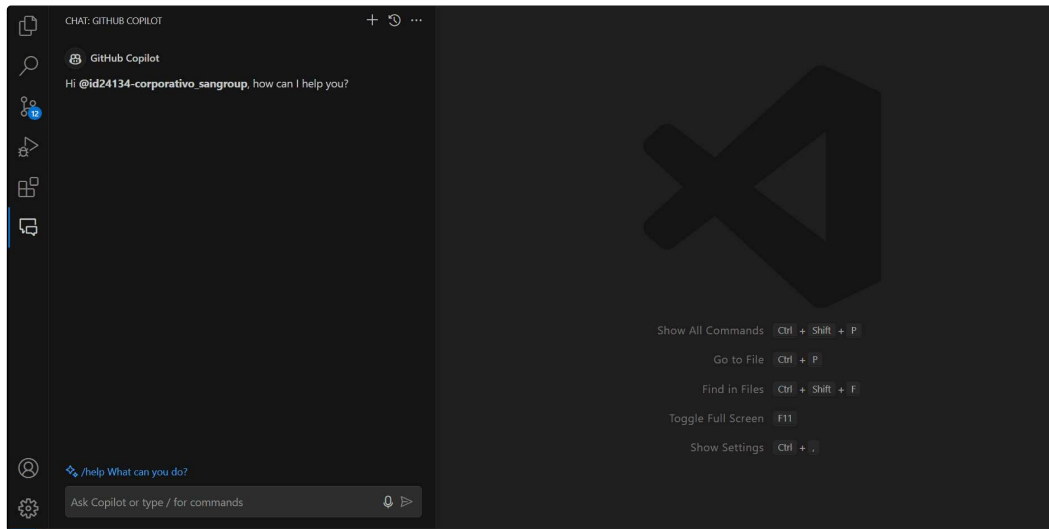
- Escribe el prompt en el chat de GitHub Copilot.
- GitHub Copilot generará un modelo de datos basado en la estructura proporcionada.
- Si la sugerencia es correcta, selecciona "Insert Into New File" para crear el archivo de modelo en tu proyecto.

Bajo este archivo customer.service.ts generaremos el código necesario mediante GitHub Copilot para consumir el endpoint. <http://localhost:3000/customers>

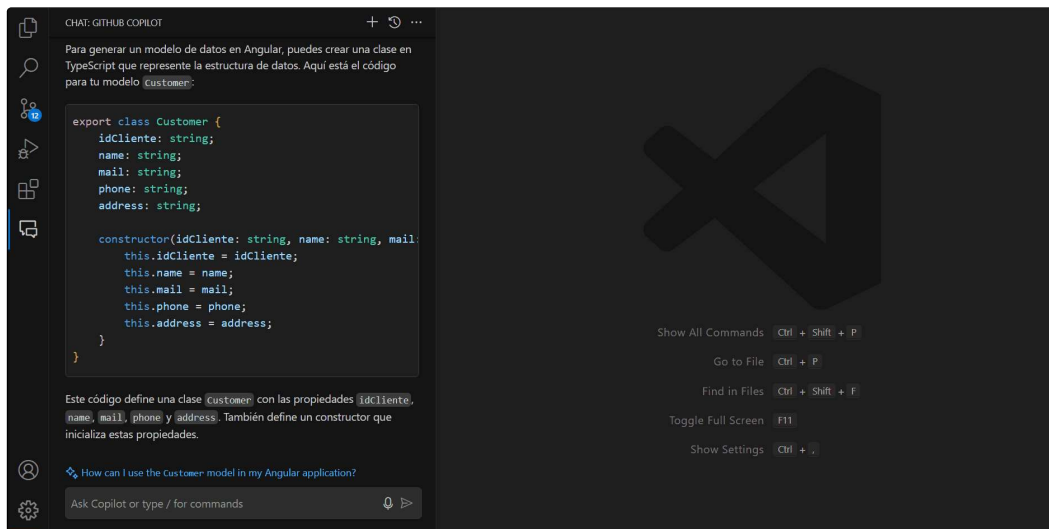
Primero que todo, debemos generar el modelo que nos permitirá contener la información obtenida desde la API, para esto mediante el prompt obtendremos un nuevo archivo respetando estructura del resultado de la API.

```
{
  "idCliente": "876543215",
  "name": "David Brown",
  "mail": "David@mail.com",
  "phone": "1234567890",
  "address": "123 Main St"
},
```

prompt: *genera un modelo de datos de nombre customer en angular para la siguiente estructura de datos de ejemplo { "idCliente": "876543215", "name": "David Brown", "mail": "David@mail.com", "phone": "1234567890", "address": "123 Main St" }*



Una vez generada la sugerencia y si se adapta a la solución esperada, podremos generar feedback en la herramienta mediante el botón de **pulgar arriba** y además aceptarla generando el archivo mediante la opción **insert into new file**.



^ Consumir una API con apoyo de GitHub Copilot

Explicación paso a paso:

- Escribe el prompt en el chat de GitHub Copilot.
- GitHub Copilot generará el código para consumir la API utilizando el modelo customer.ts.
- Inserta el código generado en el archivo de servicio correspondiente.

Ahora generaremos mediante GitHub Copilot el consumo de la api customer mediante el chat inline sobre el archivo de servicio previamente creado

prompt: *genera el código dentro de CustomerService para consumir la api http://localhost:3000/customers/ usando el modelo customer.ts*

Módulo 2: Trabajo en componentes con GitHub Copilot

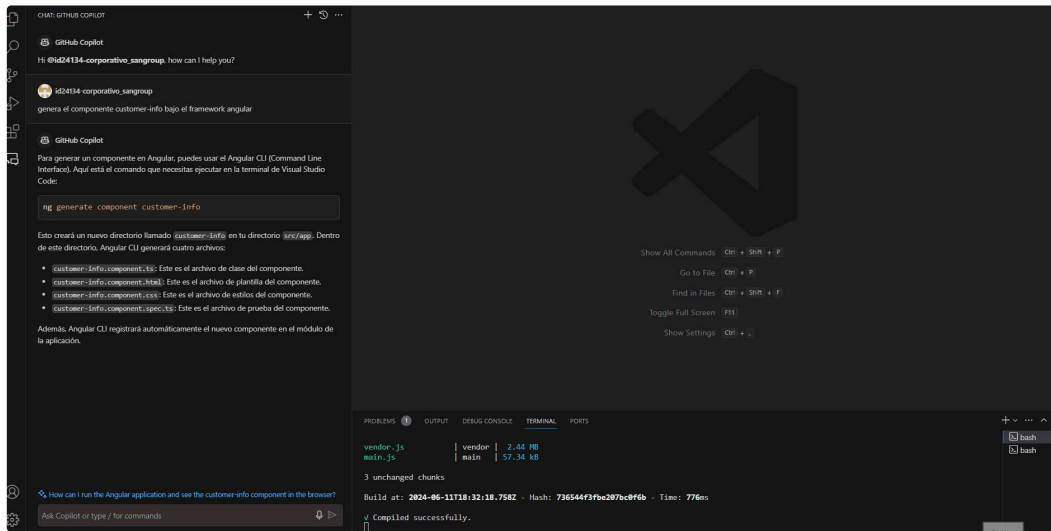
^ ¿Cómo generar código de componentes?

Explicación paso a paso:

- Escribe el prompt en el chat de GitHub Copilot.
- GitHub Copilot generará los pasos necesarios para crear el componente customer-home.
- Selecciona "Insert Into Terminal" para ejecutar el comando directamente y crear el componente.

Para crear un componente en nuestro proyecto podemos utilizar el siguiente prompt *genera el componente customer-home bajo el framework angular* recuerda que debes seguir los principios de las 4S.

Este prompt te entregará los pasos necesarios para la generación de este componente mediante angular CLI, además de una breve explicación del procedimiento, al seleccionar la opción *insert into terminal* podremos ejecutar el comando directamente sin la necesidad de volver a escribirlo



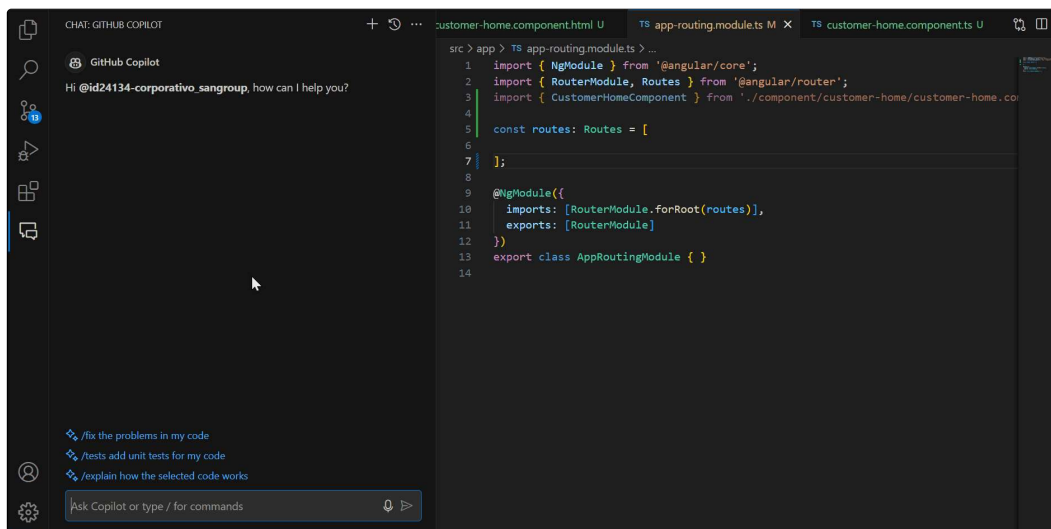
^ Incorporación al routing y consumo del servicio desde componente

Explicación paso a paso:

- Escribe el prompt en el chat de GitHub Copilot .
- GitHub Copilot generará el código necesario para añadir el componente customer-home al routing.
- Inserta el código en el archivo de routing de la aplicación.
- Escribe el prompt en el chat inline de GitHub Copilot .
- GitHub Copilot generará el código necesario para añadir el componente customer-home al path:".
- Inserta el código en el archivo de routing de la aplicación.

Incorporación de routing

Si requerimos de una explicación mientras solicitamos el apoyo de GitHub Copilot mediante el chat con el prompt *genera el routing de #file:customer-home.component.ts*, GitHub Copilot no entregara de manera rápida la forma de conectar nuestro componente al routing de la aplicación, junto a la explicación del paso a paso, cabe destacar que mediante el comando `#file` le podremos indicar a GitHub Copilot el scope a tomar en cuenta.

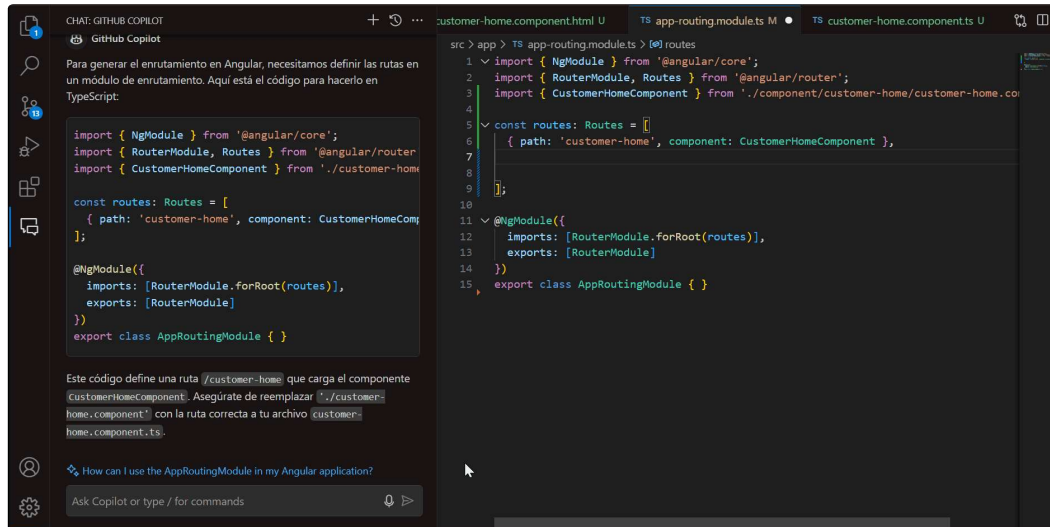


Para tener en consideración que al utilizar la opción *insert at cursor* esta pegará todo el código disponible de la sugerencia desde donde este el cursor posicionado.

Otra forma de incorporar al routing el componente es mediante el chat inline (ctrl + i), en donde podremos indicar el prompt *genera el código para incorporar el componente customer-home al path:"en el archivo app-routing.module.ts"*, mediante la sugerencia en el código mediante un comentario descriptivo

```
//genera el código para incorporar el componente customer-home al path:"
// en el archivo app-routing.module.ts
```

Entregándonos el siguiente resultado:



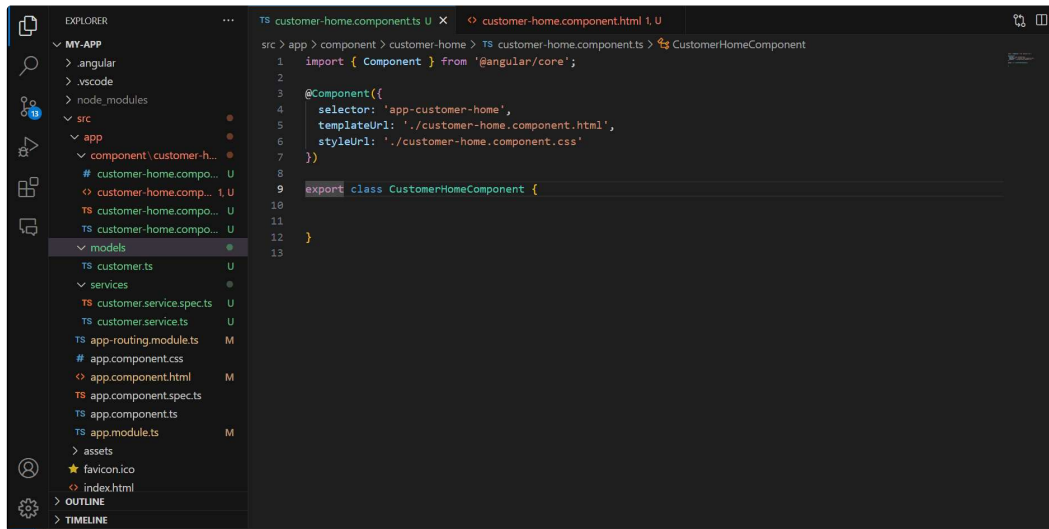
^ Consumo de servicio en componente

Explicación paso a paso:

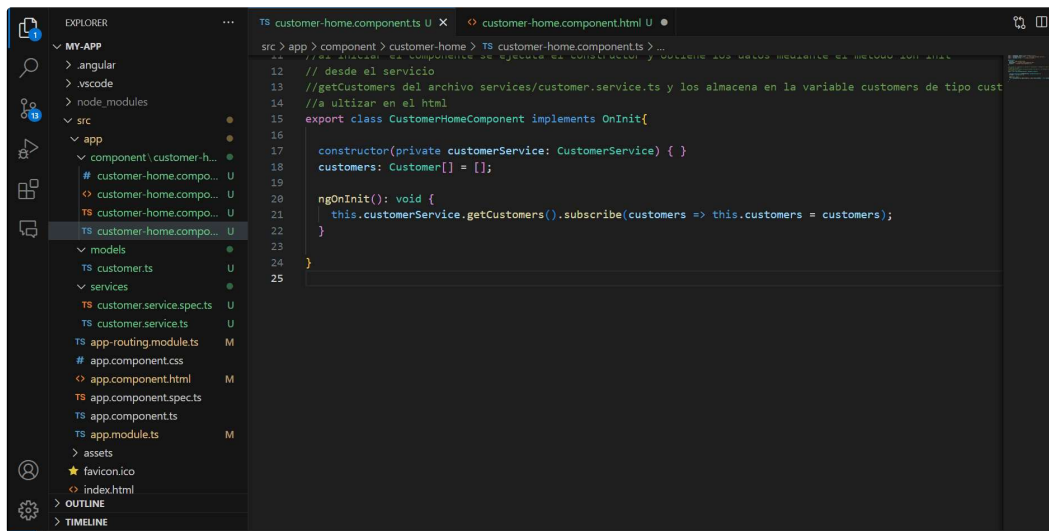
- Escribe el prompt en el mismo código mediante comentarios en el archivo.
- GitHub Copilot generará el código para consumir el servicio `getCustomers` y almacenar los datos en una variable.
- Inserta el código en el componente correspondiente.

Para consumir el servicio podemos apoyarnos de GitHub Copilot de la siguiente forma, ingresamos al componente creado y describimos en el componente lo que se requiere de la siguiente forma:

```
//al iniciar el componente se ejecuta el constructor y obtiene los datos del servicio
//getCustomers del archivo services/customer.service.ts y los almacena en la variable customers
```

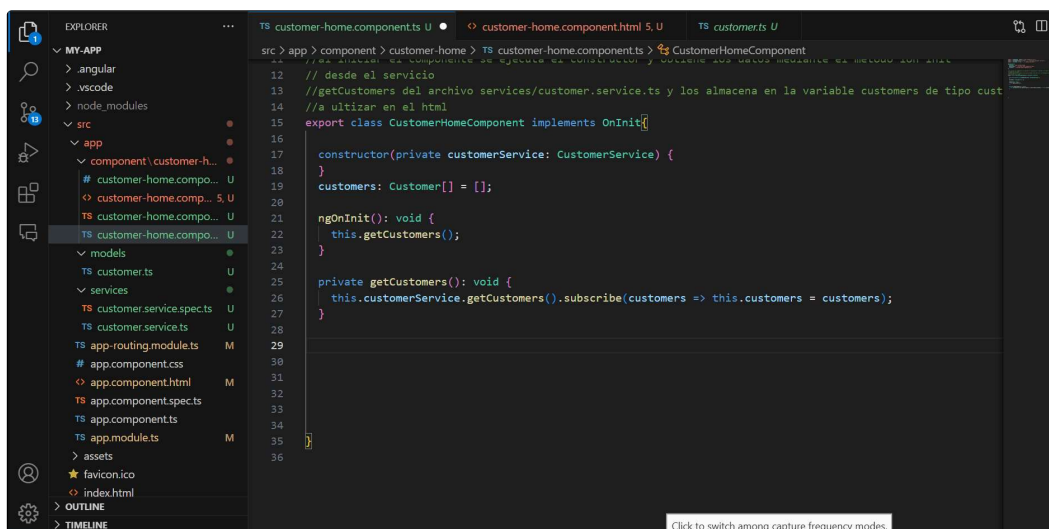


También mediante GitHub Copilot podemos refactorizar código de la siguiente manera mediante el chat inline utilizamos la acción /fix refactoriza el código hacia un nuevo método



Agregamos un método que nos entregue el detalle de customer para utilizarlo en la página integrando comentarios al código y corregimos algunos errores con la acción /fix

```
//método para expandir la información de un cliente en específico
//recibe como parámetro el cliente seleccionado
//y se le asigna a la variable customerSelected
```



Módulo 3: Generación de código HTML con apoyo de GitHub Copilot

^ Generación de código HTML

Explicación paso a paso:

- Escribe el prompt en el chat inline de GitHub Copilot.
- GitHub Copilot generará el código HTML necesario para mostrar una tabla dinámica.
- Inserta el código en el archivo HTML del componente.

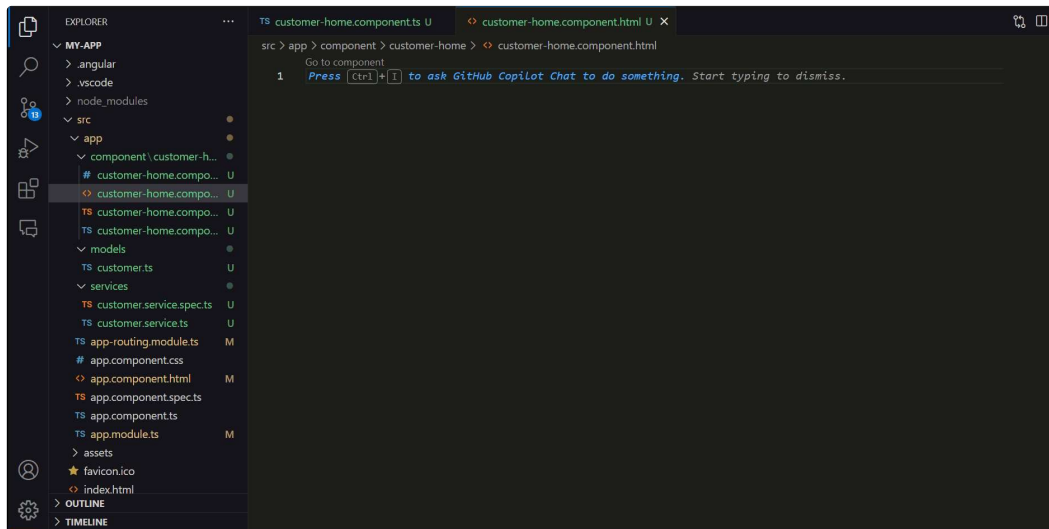
Trabajemos con los consumos de APIs generados en el módulo anterior.

Mostrar una lista de clientes

Para mostrar el listado de clientes en la página podemos generar el siguiente prompt generado por etapas enriqueciendo el prompt o corrigiéndolo si fuera necesario.

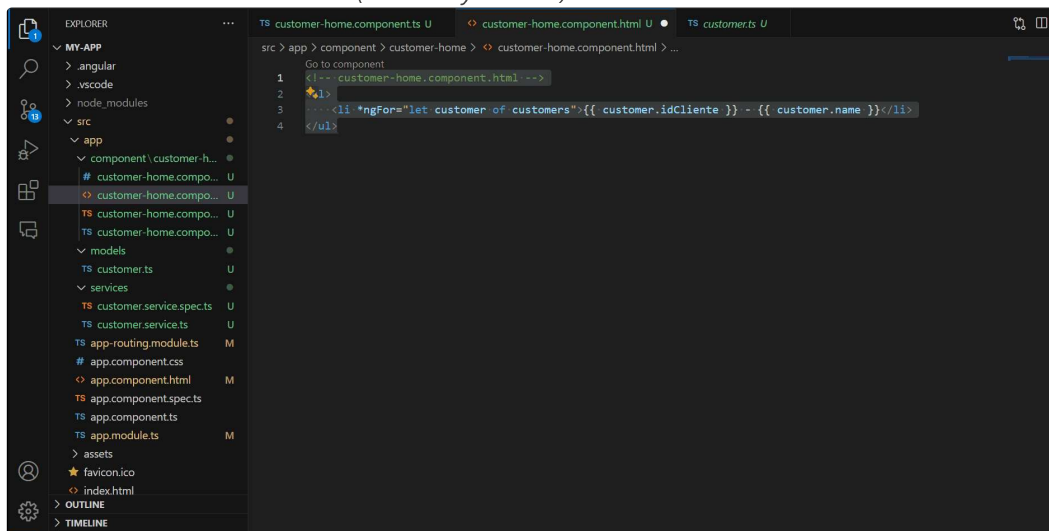
prompts:

muestra un listado de clientes



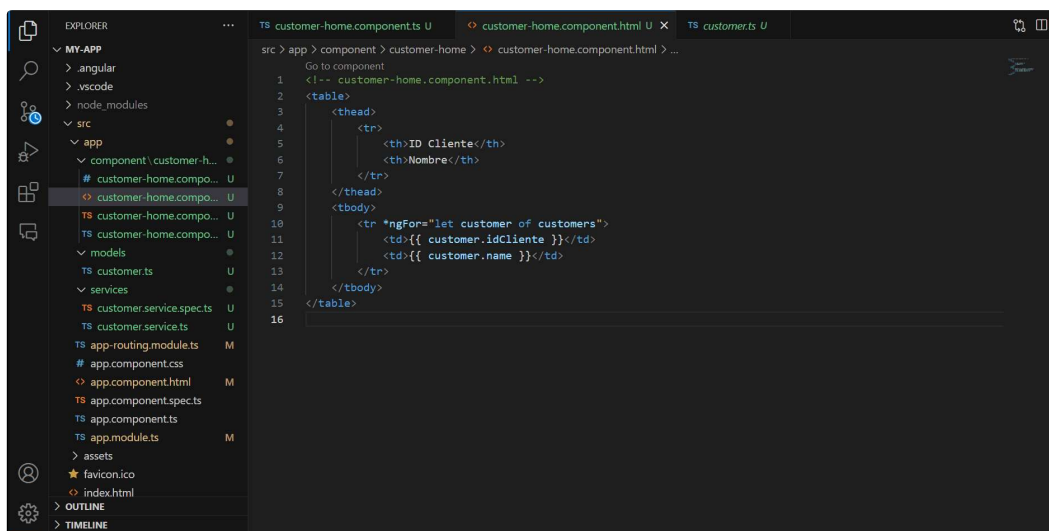
muestra un listado de los elementos de customers usando customer-home.component.ts

muestra un listado de los elementos de customers (idCliente y nombre) en una tabla dinámica



```
1 <!-- customer-home.component.html -->
2 <ul>
3   <li *ngFor="let customer of customers">{{ customer.idCliente }} - {{ customer.name }}</li>
4 </ul>
```

muestra un listado de los elementos de customers (idCliente y nombre) en una tabla dinamica que al seleccionar un registro se expanda y muestre los otros elementos de customer.ts (mail, phone, address)



```
1 <!-- customer-home.component.html -->
2 <table>
3   <thead>
4     <tr>
5       <th>ID Cliente</th>
6       <th>Nombre</th>
7     </tr>
8   </thead>
9   <tbody>
10    <tr *ngFor="let customer of customers">
11      <td>{{ customer.idCliente }}</td>
12      <td>{{ customer.name }}</td>
13    </tr>
14  </tbody>
15 </table>
16
```

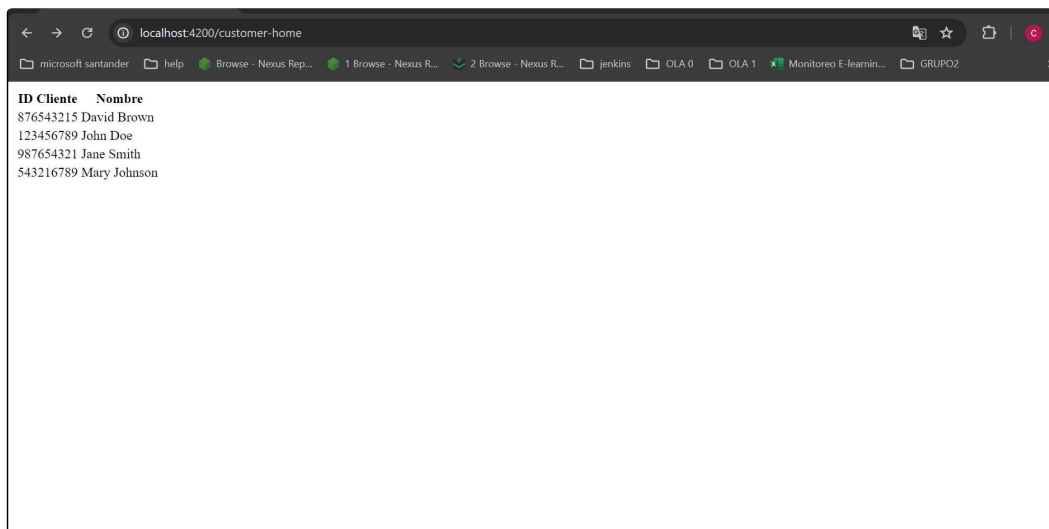
^ Aplicación de estilo sobre vista

Explicación paso a paso:

- Escribe el prompt en el chat de GitHub Copilot.
- GitHub Copilot generará el código CSS necesario para estilizar el componente.
- Inserta el código en un archivo CSS nuevo o existente en tu proyecto.

Trabajemos el estilo de la página para que tome un formato más banco generamos el siguiente prompt:

@workspace #file:customer-home.component.html Crea un archivo CSS para el componente de Angular que le dé un estilo bancario de Santander a la visualización de la información del cliente. Usa la paleta de colores del banco Santander. El diseño debe ser limpio y profesional.



^ Separando la información en tablas

Explicación paso a paso:

- Escribe el prompt en el chat de GitHub Copilot.
- GitHub Copilot sugerirá el código correspondiente.
- Inserta el código en la página HTML a modificar.

prompt: *@workspace genera una tabla html nueva que contenga el código seleccionado*

Evaluación: ¡RESPONDE PARA APROBAR EL CURSO!

Generación de código para Angular con GitHub Copilot

Instrucciones:

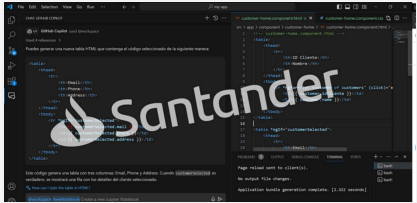
- 1. Sigue el paso a paso del curso.
- 2. Carga lo solicitado para aprobar.
- 3. Debes obtener mínimo 4 puntos para aprobar.

Hola, Cristian. Cuando envíe este formulario, el propietario verá su nombre y dirección de correo electrónico.

* Obligatorio

1

Carga la captura de un prompt utilizado en el proceso. (Pregunta no anónima) * (2 puntos)



⬆ Cargar archivo

Límite de número de archivos:1 Límite de tamaño del archivo individual: 10MB
Tipos de archivo permitidos: Word, Excel, PPT, PDF, Imagen, Vídeo, Audio

Si tienes problemas para ver la evaluación, ingresa al siguiente enlace mediante **Microsoft Edge**:

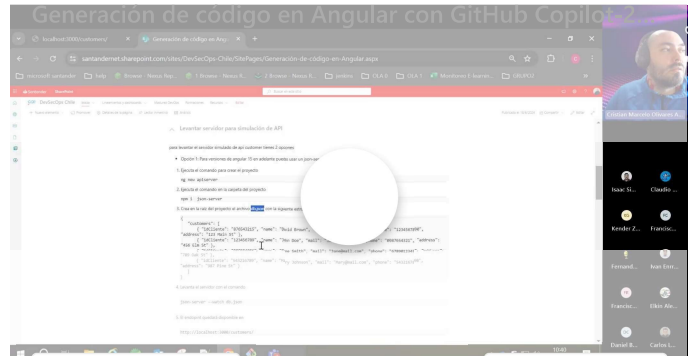
<https://forms.office.com/e/srX99Dc0Rd>

Si tienes dudas con este tema, contacta a



Cristian Olivares Araya

¡También puedes ver el video de la sesión práctica!
Donde pasamos por todos los módulos de este curso



¡Felicidades por completar el curso de Generación de Código en Angular con GitHub Copilot!

Esperamos que hayas encontrado útiles los módulos y que las herramientas proporcionadas te ayuden a ser más eficiente en tu trabajo diario. Sigue practicando y explorando las capacidades de GitHub Copilot para mejorar continuamente tus habilidades de programación.

¡Buena suerte en tus futuros proyectos y sigue creando código increíble con Angular!

¡Ayúdanos a mejorar!

Responde la encuesta de satisfacción con el curso aquí:



Microsoft Forms