# Proyecto web blog con astro

## Entorno de desarrollo:

para iniciar a trabajare en astro es necesario tener instalado algunas cosas como:

-node y npm

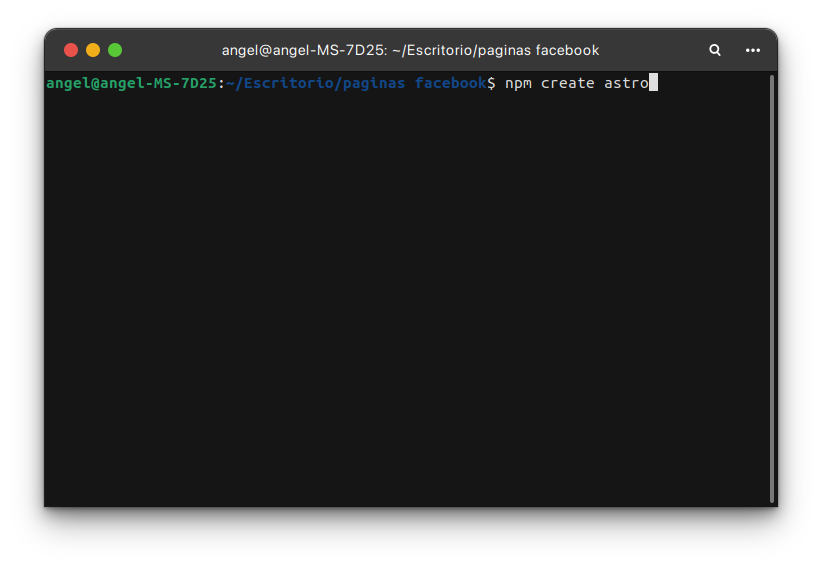
-editor de codigo, en este caso vscode

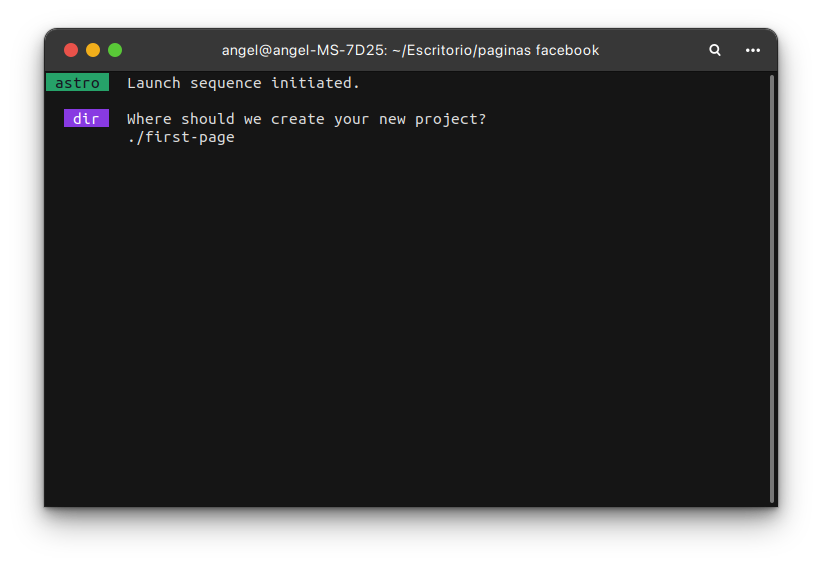
-git

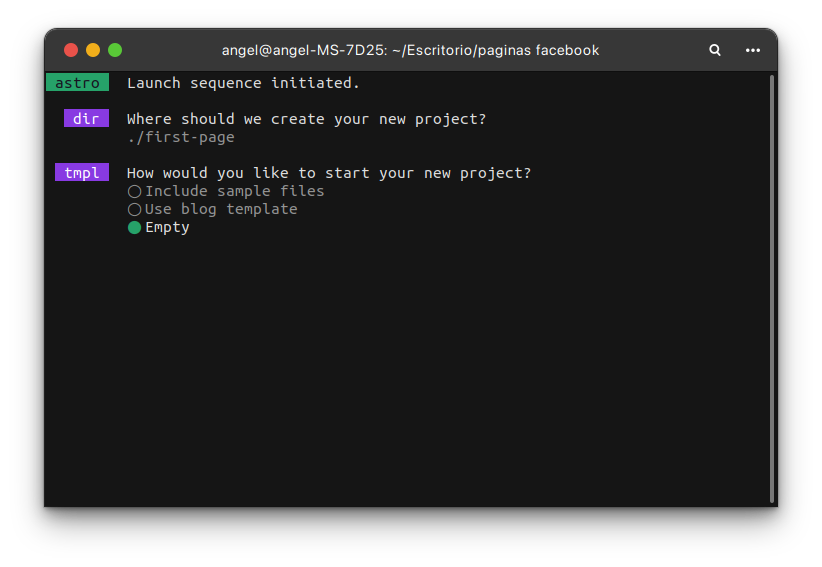
## Configuración inicial:

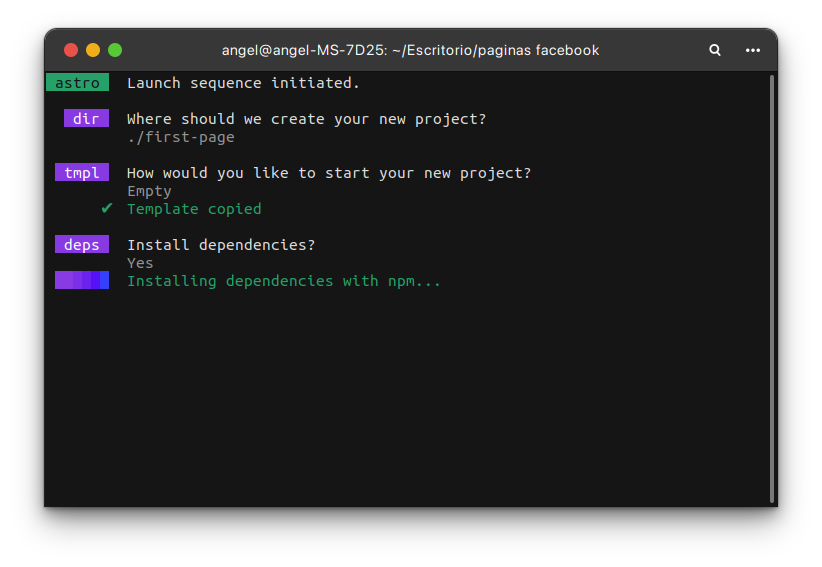
para iniciar con la creacion de un proyecto en la ruta ya establecida en astro es necesario crear dicho proyecto mediante el siguiente comando:

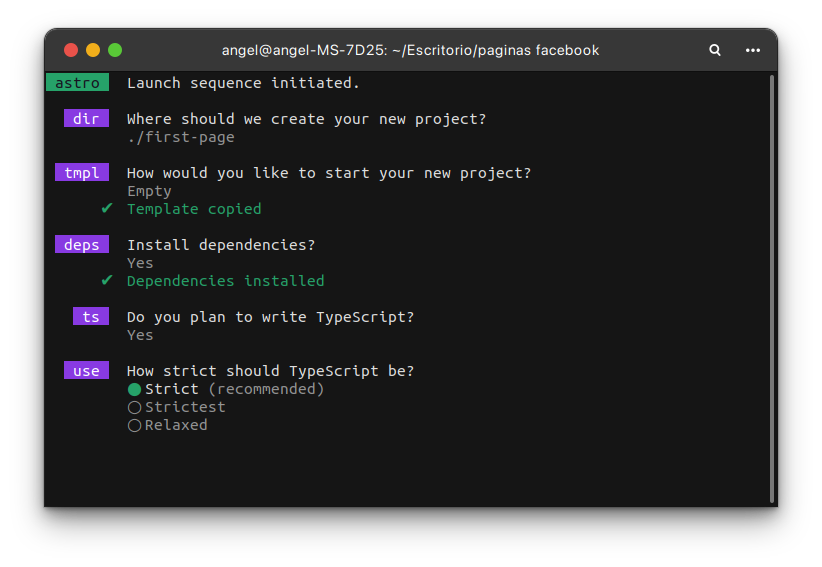
npm create astro

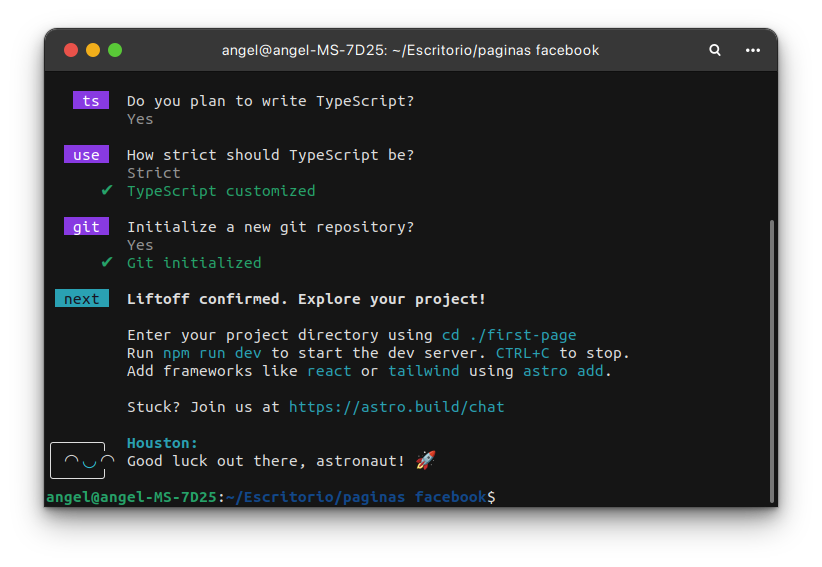
les preguntara como quieren que se llame la carpeta que contendrá el proyecto:

despues de eso les preguntara que si es necesario incluir un ejemplo, un blog o dejarlo vacio, queda a criterio pero es recomendable dejarlo vació para entender mejor la logica.

Instalamos las dependencias

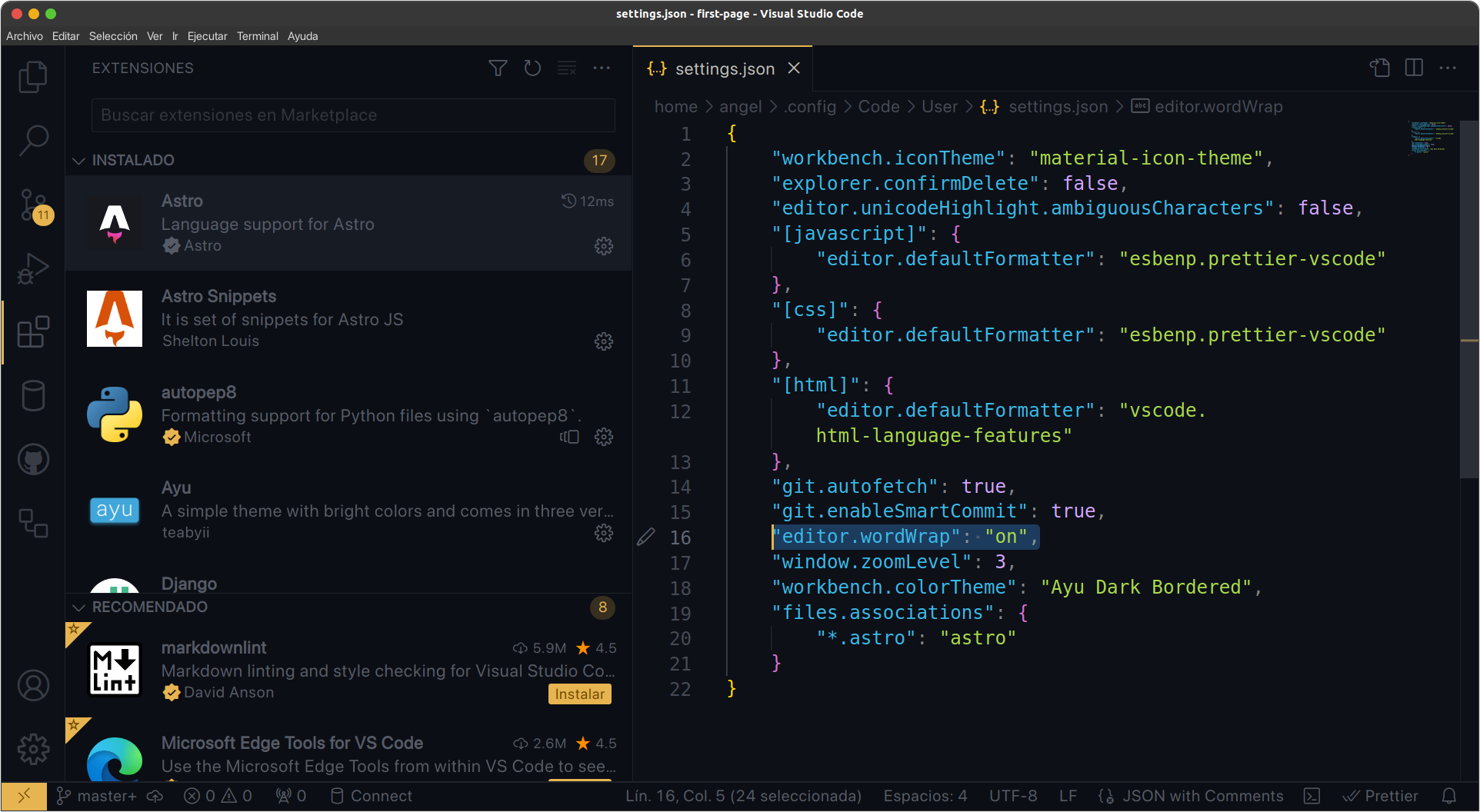
marcamos el uso rcomendado de typescript para que en un futuro que el proyecto cresca nos sirva de ayuda.

Iniciamos nuestro repositorio y ya desde este momento podremos empezar a realizar nuestro proyecto con libertad.

Abrimos el proyecto desde vscode y podemos ver la estructura del proyecto el cual en mayor medida utilizaremos la carpeta de public y el src.

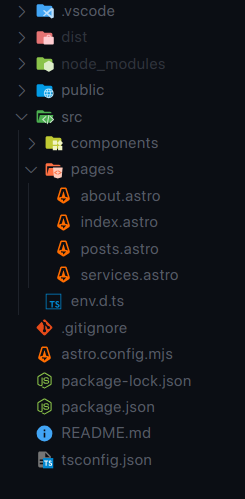
Nota:

para que nuestro vscode se adapte de mejor manera recomendamos añadir las estenciones siguientes ademas de la configuracion de json en el user json:

el wordwrap nos ayudara a formatear el codigo de acuerdo al tamaño de la pantalla de nuestro editor de manera dinamica.

## Paginas en Astro:

para que nosotros creamos nuevas paginas en astro es muy sencillo como el simple hecho de agregar un nuevo archivo en la carpeta **src/pages** y automáticamente ya existiría en las rutas una nueva pagina la cual podemos visitar.

Cabe recalcar que una estructura típica en los archivos de astro estarán conformados por lo siguiente.

*---*

const title = "Mi primera pagina"

*---*

<html lang="en">

<head>

<meta charset="utf-8" />

<link rel="icon" type="image/svg+xml" href="/favicon.svg" />

<meta name="viewport" content="width=device-width" />

<meta name="generator" content={Astro.generator} />

<title>{title}</title>

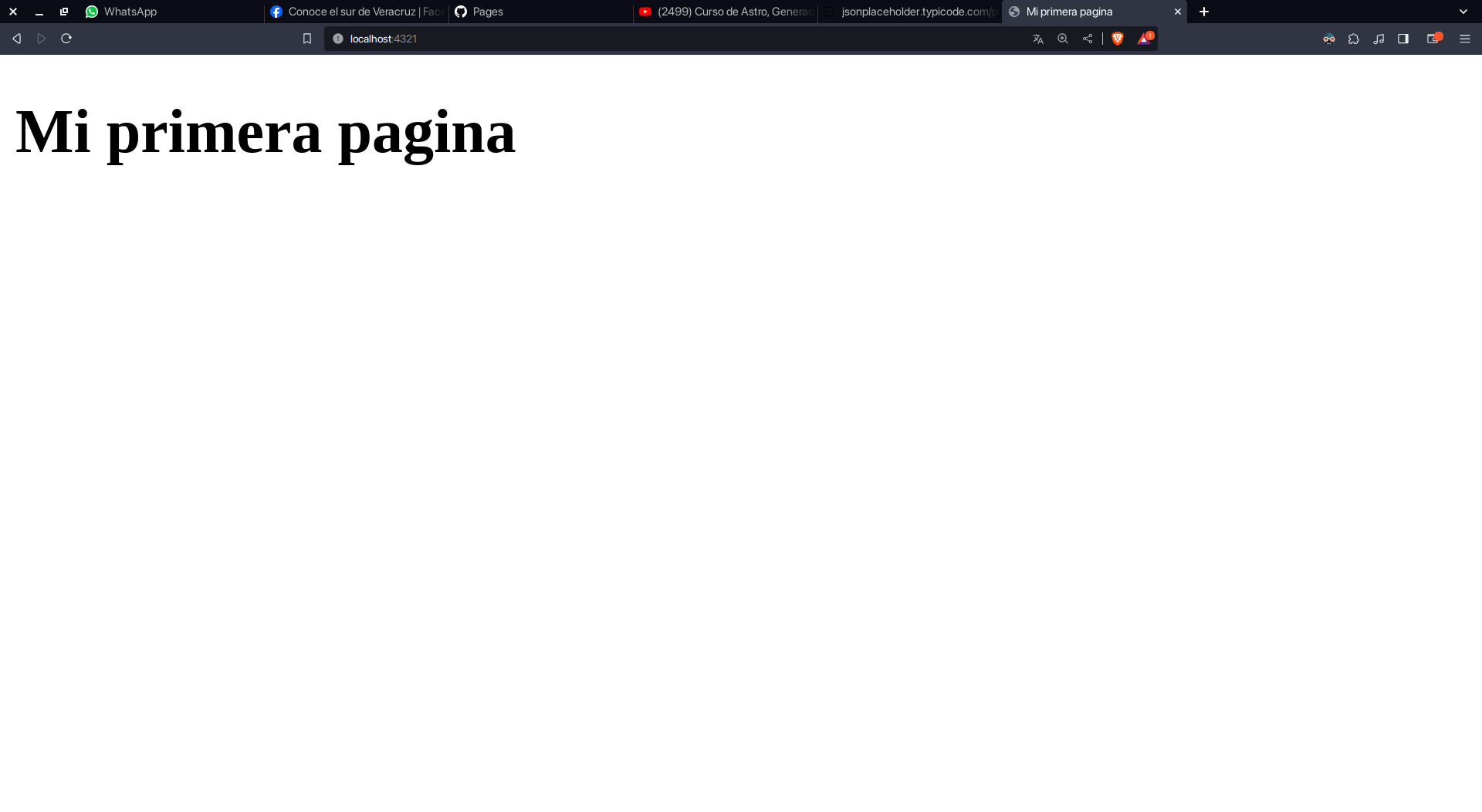
</head>

<body>

<h1>{title}</h1>

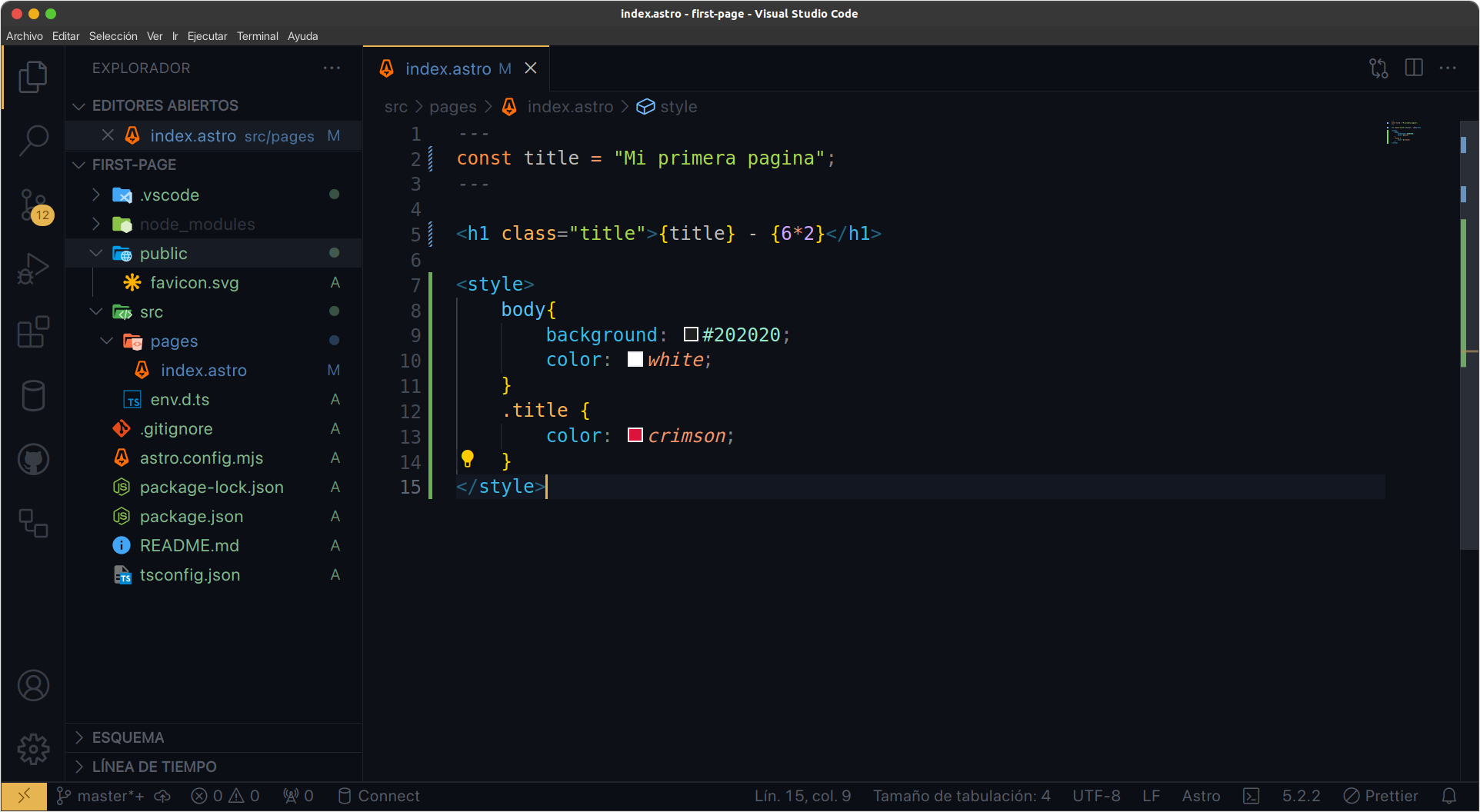
</body>

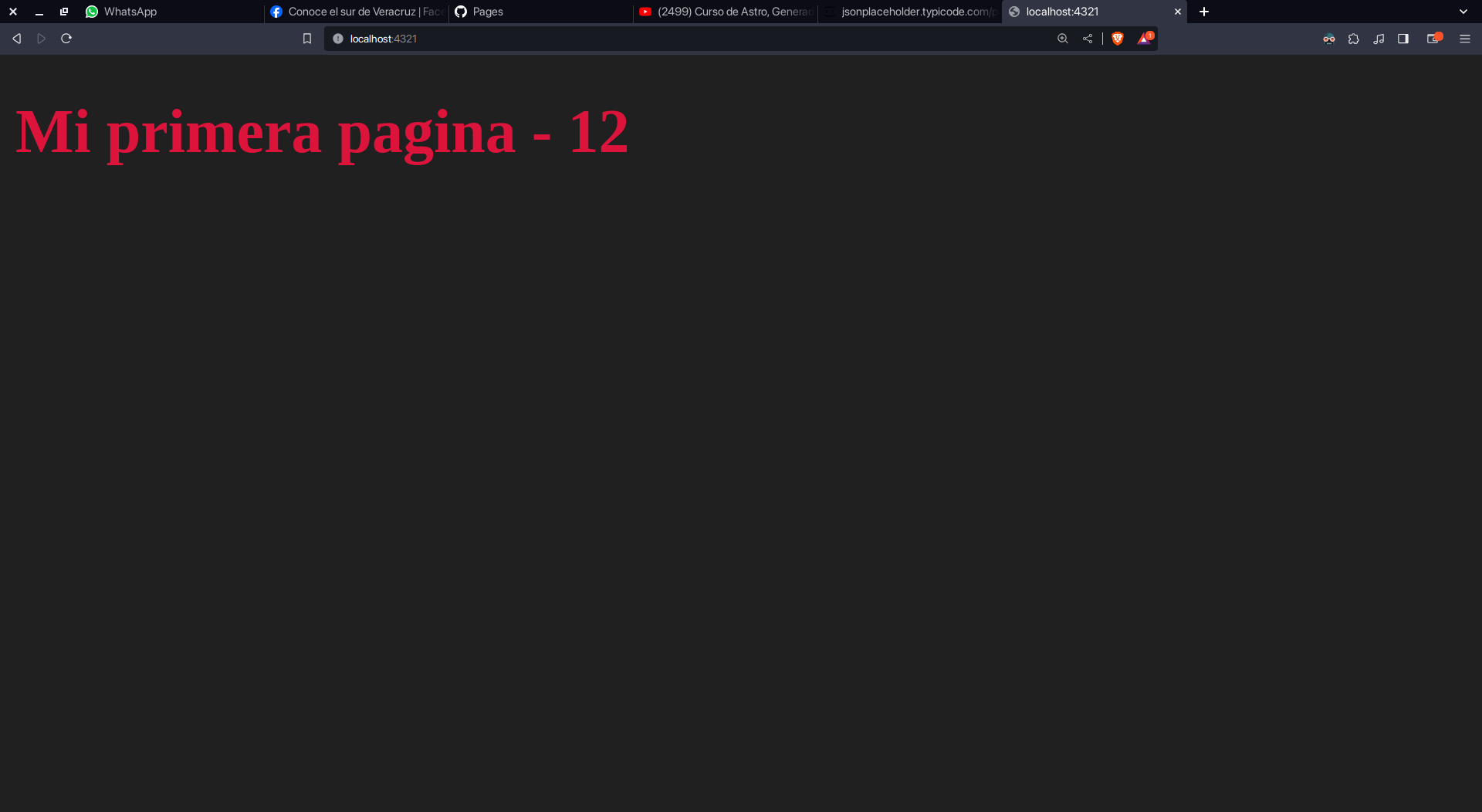
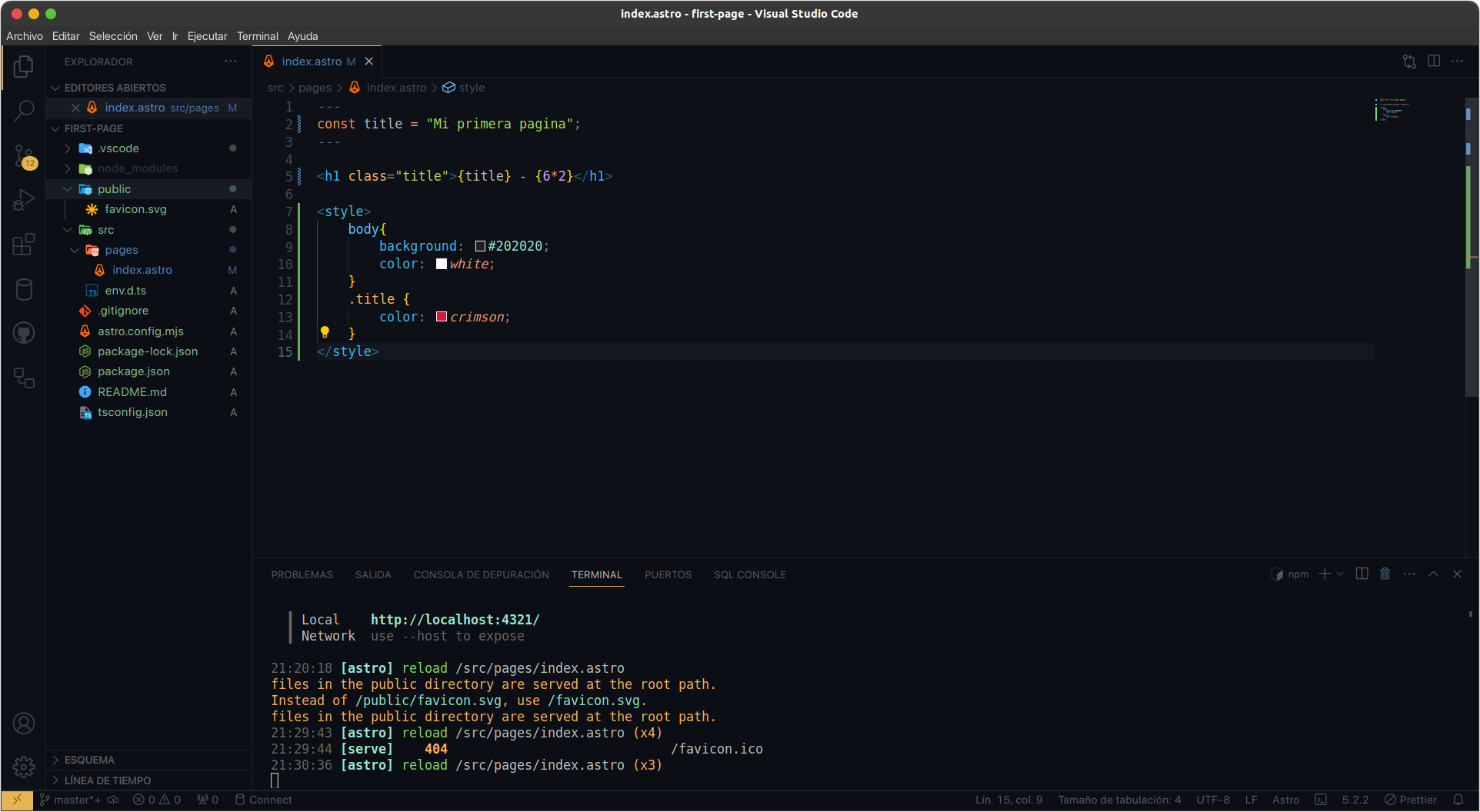
</html>

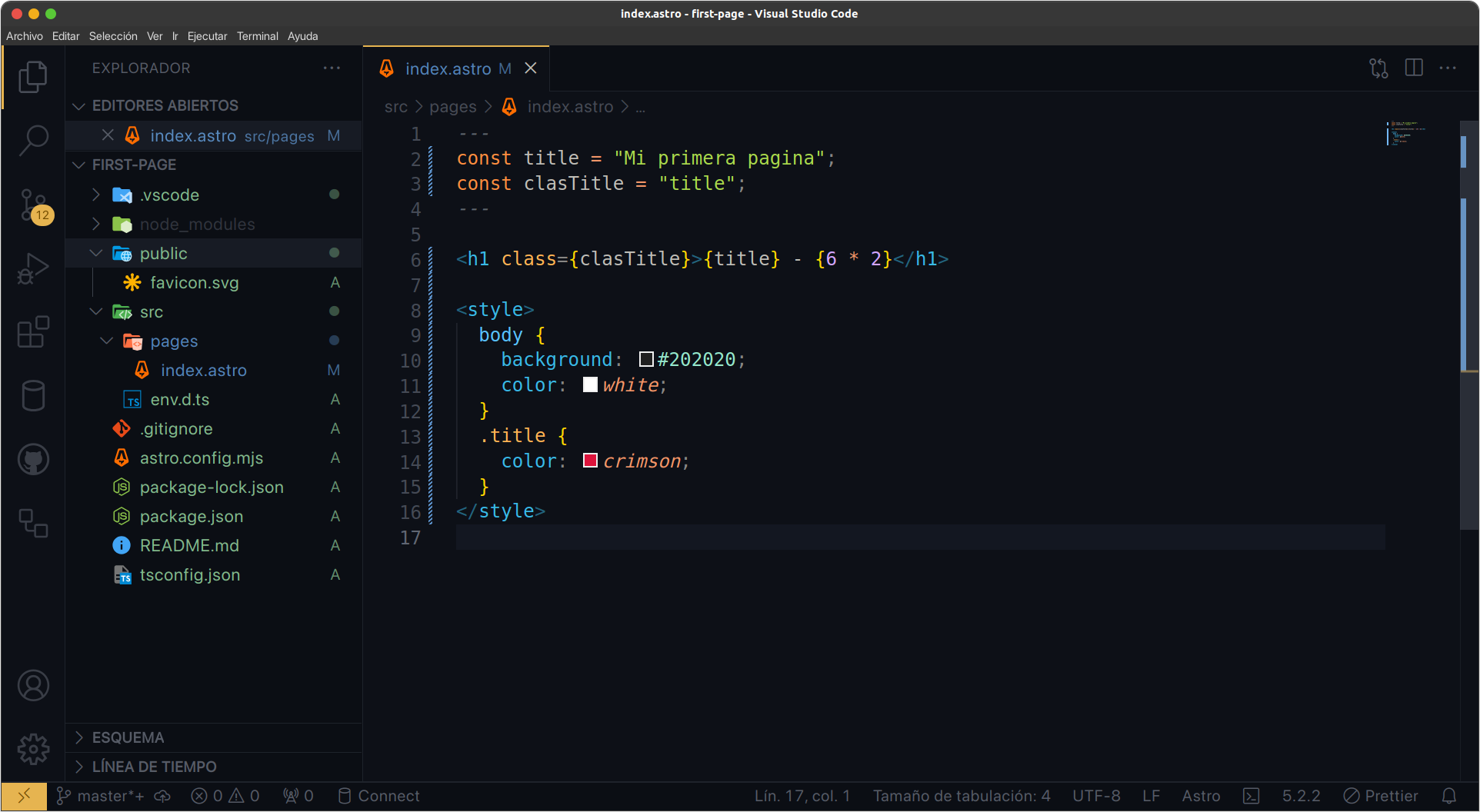
entre “---” el codigo que usara logica de js para la conexión de datos el cual la estructura de astro podra leer entre llaves y se vera de la siguiente manera:

## Sintaxis JSX en astro build & fetch:

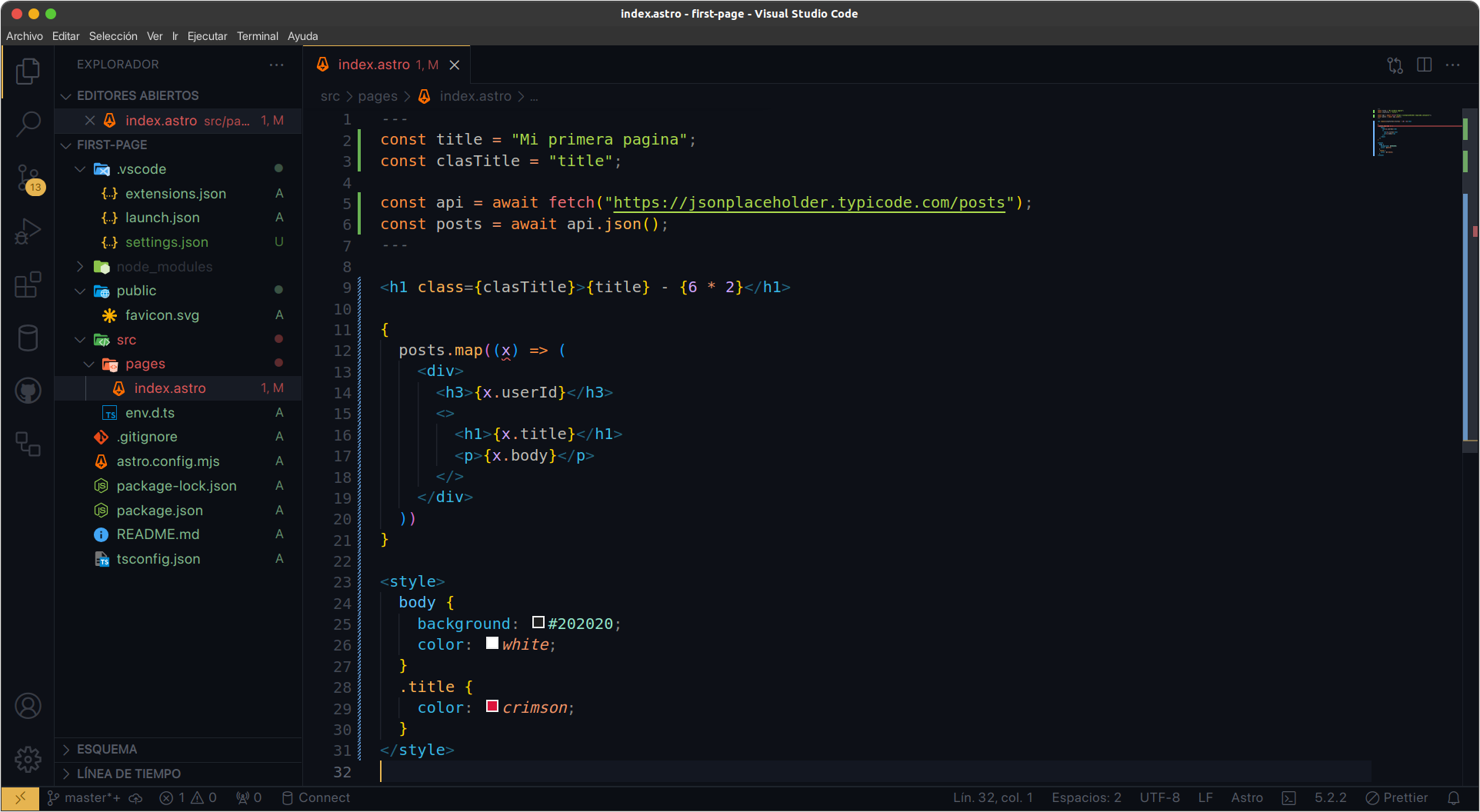
Hay que recalcar que astro puede interpretar sintaxis de javascript que posteriormente el framework traspasara a html puro pero que en el desarrollo nos ayudara mucho para mostrar informacion, por ejemplo usando las llaves {}.



como puedes ver al inicio se utiliza codigo de js pero al final astro convierte todo a contenido estatico de html con el fin de volver ligero el contenido.

Podemos ademas de eso utilizar la sintaxis para añadir las clases directamente desde una constate.

Otra cosa que podemos hacer y que nos ayudara de grana manera es la obtención de datos desde una api y recorriendolo en nuestra pagina de manera rapida y sencilla por ejemplo utilizando la api de <https://jsonplaceholder.typicode.com/posts>

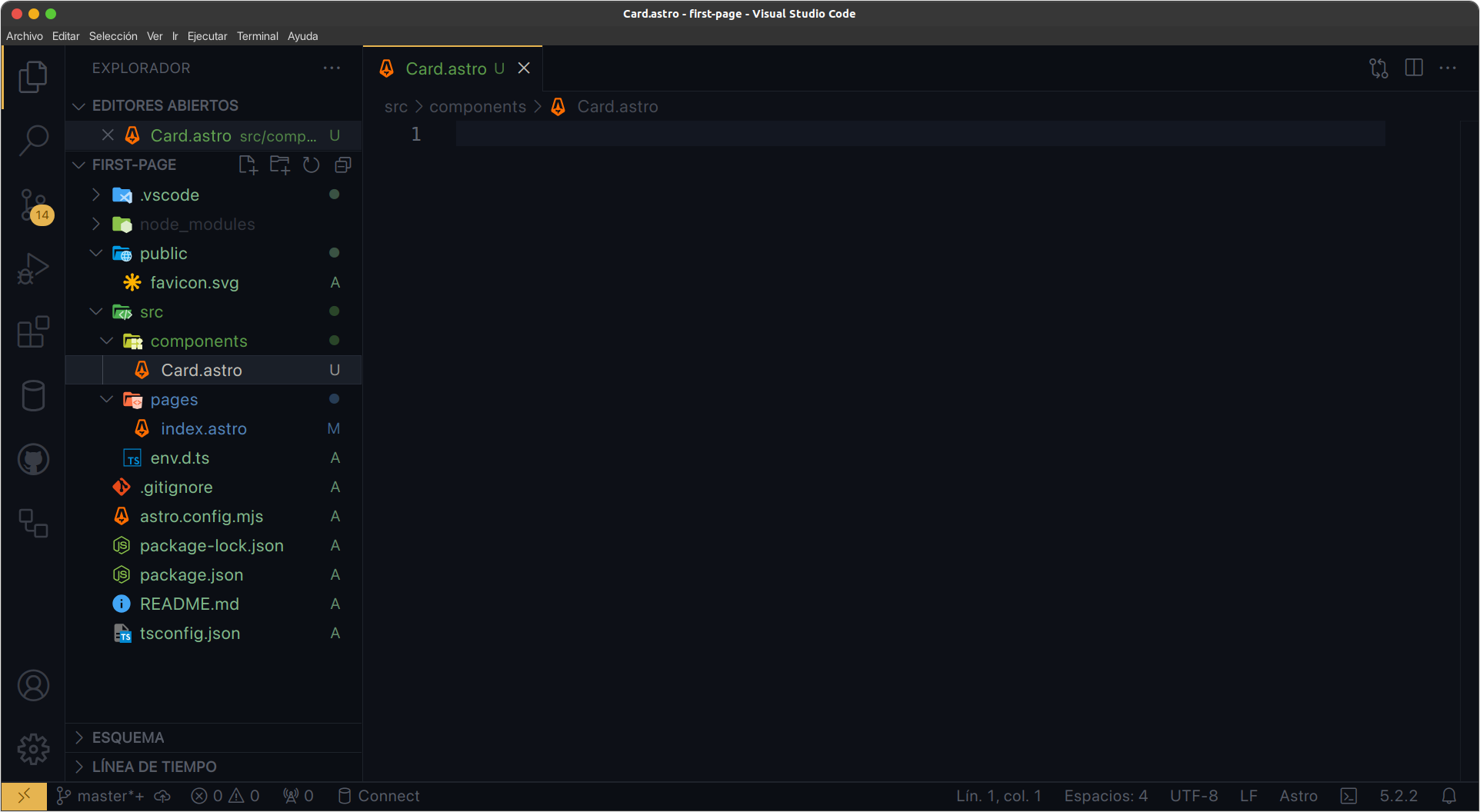


de esta forma podremos recorrer datos que al final Astro va a pasar a simple y crudo html.

## Astro components

En astro se utiliza una filosofia de reutilizacion de codigo que nos facilitara de gran manera la creacion de las paginas webs que estamos creando así como otros frameworks de js, para eso utilizaremos algo que conocemos como componentes, eso de la siguiente manera.

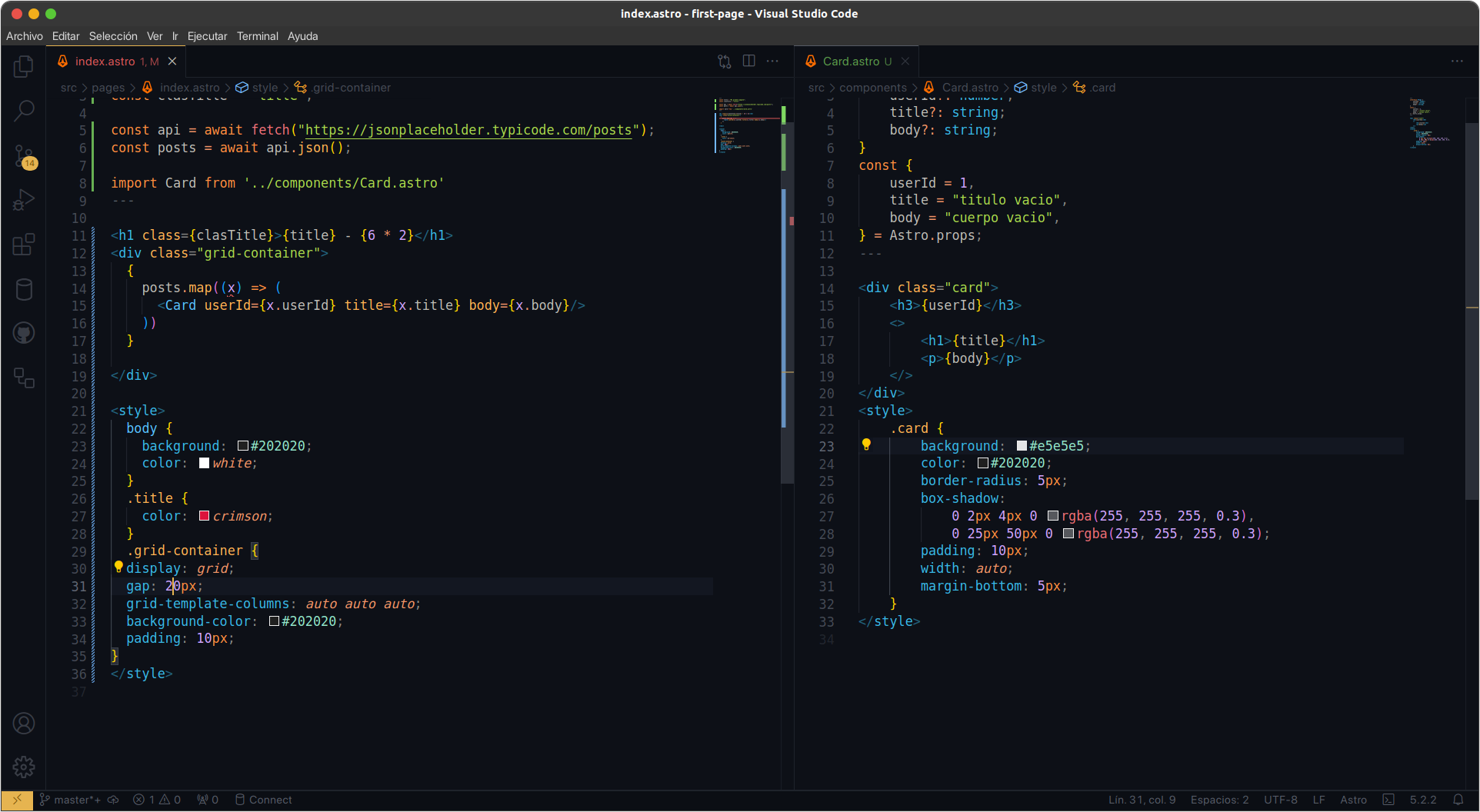
Primero que nada hay que localizarlos aparte de todo para tener un orden a la hora de crear nuevas porciones de código.

Como podemos verse agrego el componente dentro de una nueva carpeta, ademas de eso por conveniencia tipica la agregamos con la primera letra en mayuscula para identificar que es un componente y con extencion astro, desde aquí se puede agregar varios datos y se puede invocar desde distintas paginas de la siguiente manera.

import Card from '../components/Card.astro'

en la pagina utilizando el --- podemos llamar el componente en primera instancia le agregaremos un nombre y despues indicaremos la ubicación del mismo. En el componente podemos agregar el estilo que querramos que se agregue a lo que vamos a mostrar, tambien es preciso decir que el componente sera dinamico en contenido, es decir que la estructura visual sera la misma o diferira en algunas cosas ademas del contenido es por eso mismo que podemos agregar opciones dinamicas utilizando algo conocido como props el cual ayudara a diseñar el codigo que utilizemos.

Podemos observar primero:

en el componente se define que tipo de variables son las que se utilizaran mediante el uso de typescrip asignandoles a la vez que el valor ingresado puede ser nulo y de igual manera llenando esas constantes con los valores que los prop de astro les proporcionen y si el valor no es asignado les de uno predeterminado. Una vez definidas esas constantes podemos ubicarlas dentro del card y asignarles un estilo propio que se lo añadiremos despues.

Una vez diseñado el componente lo podemos llamar en alguna pagina como lo indicamos anteriormente y cargandolo en el map con los datos que corresponden a la api llamada con los atributos que definimos en el typescript de el componente.

