

Práctica no. 9 Middleware Express

Preguntas:

1. ¿Qué es un middleware?

Es una función que se ejecuta antes o durante el flujo de una petición en express.

2. ¿Cuál es el significado del término "MEAN stack" ?

Es el acrónimo formado por las iniciales de las cuatro tecnologías principales que entran en juego: MongoDB, Express, Angular y Node.js y permite crear aplicaciones distribuidas utilizando el mismo lenguaje JavaScript en todas sus fases y capas.

3. ¿Cuáles son las características/funcionalidades que Express ofrece como middleware?

El "middleware" es ampliamente utilizado en las aplicaciones de Express: desde tareas para servir archivos estáticos, a la gestión de errores o la compresión de las respuestas HTTP.

Sus características principales son: hacer cambios a una petición, ejecutar código, realizar cambios a la petición o al objeto pedido, puede también finalizar el ciclo de petición-respuesta.

4. ¿Qué es una ruta en express?

Es la dirección textual del puerto a la que tendremos que acceder desde el navegador para llegar a este manejador.

5. ¿Qué es un "route handler" en express y para qué se utiliza?

El manejador de ruta, en español, es todo el código que te dice de dónde viene, hacia dónde va y que va a mandar en la resp.

6. ¿Cuáles son los 2 parámetros necesarios para crear una ruta para el método GET con express. ejemplo: app.get()?

La sintaxis para la ruta del método GET es la siguiente:

```
app.get (ruta [, middleware], devolución de llamada [, devolución de llamada ...])
```

7. ¿Qué es lo que hace la instrucción res.send?

Como puede deducirse por el nombre, envía una respuesta cuando se mande a llamar por el navegador web.

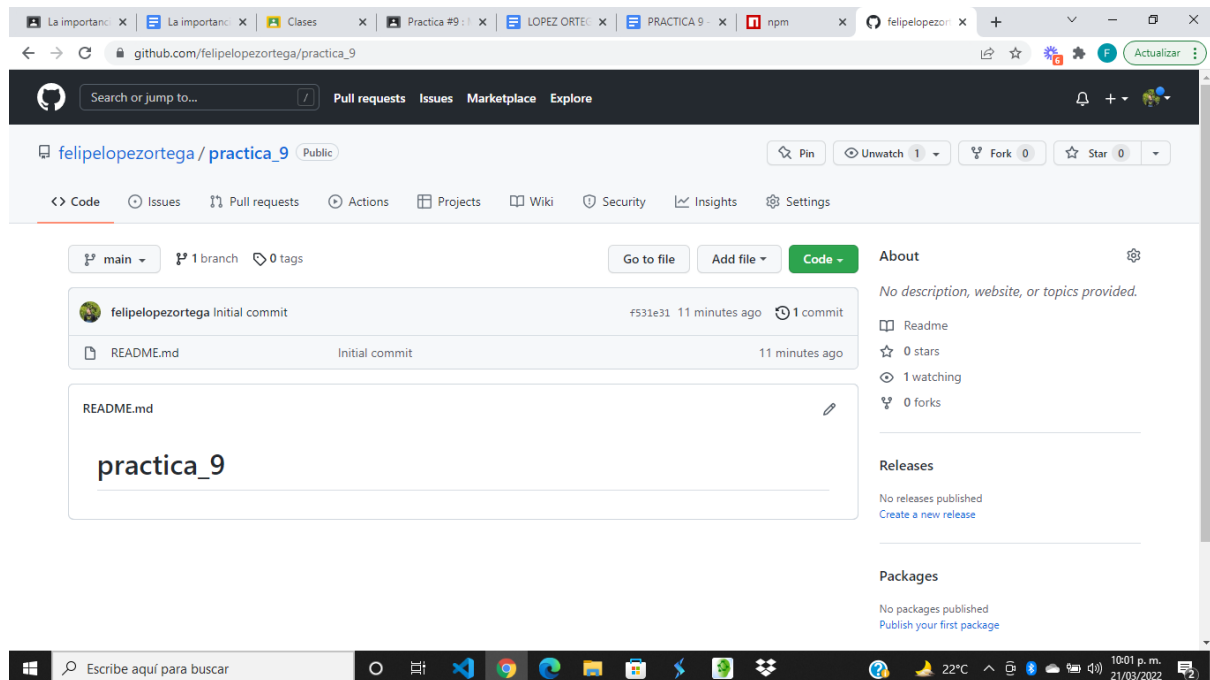
8. ¿Cuál es el motivo por el que express es tan popular?

La razón por la que Express es el marco web más popular es que facilita el desarrollo de aplicaciones web, sitios web y API. También ofrece una colección subyacente de topografías. Con Express.js, podrá perfeccionar diferentes aspectos de la aplicación web. Puede determinar configuraciones como la ubicación de las plantillas que se usarán para la respuesta o el puerto que se usará para establecer una conexión.

Práctica

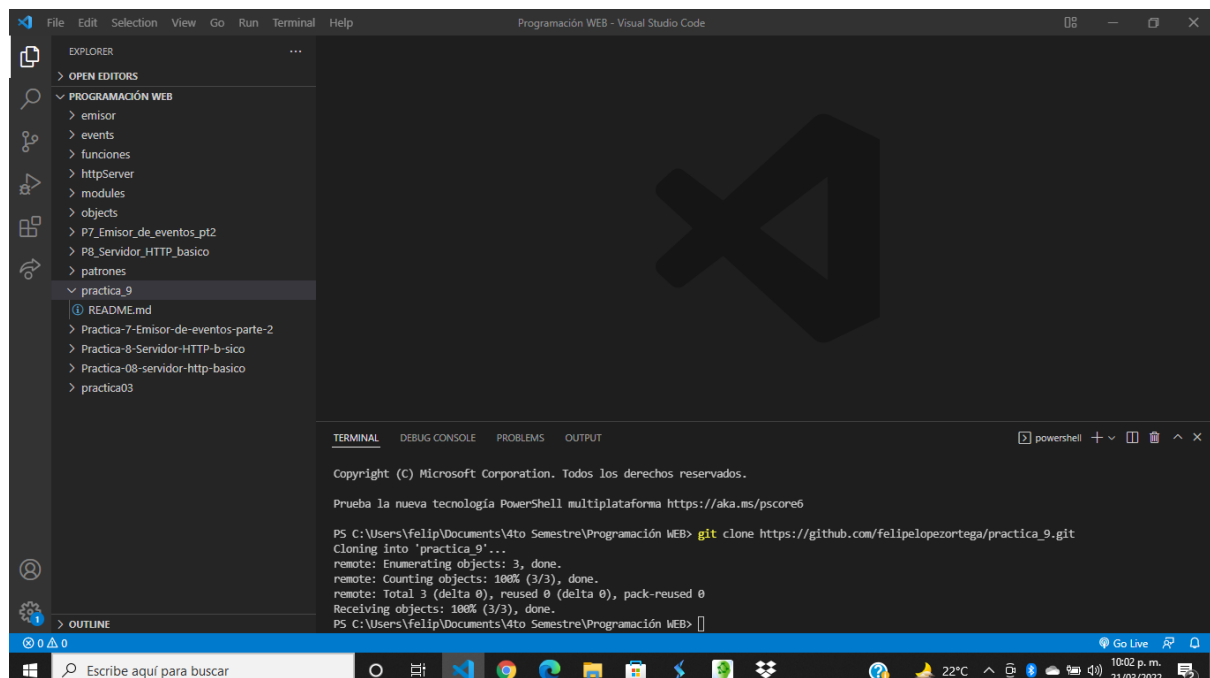
Paso 1.

Se creó el repositorio en Github.



Paso 2.

Se clonó a visual studio code.



Se inicializó npm y se descargó el paquete de express

La importancia de la programación web

Clases

Practica #9: Programación web

PRACTICA 9 - Programación web

LOPEZ ORTEGA, FELIPE

express - npm

heloelopezor

npmjs.com/package/express

Actualizar

express

Fast, unopinionated, minimalist web framework for **node**.

npm v4.17.3

downloads 95M/month

linux passing

windows passing

coverage 100%

```
const express = require('express')
const app = express()

app.get('/', function (req, res) {
  res.send('Hello World')
})

app.listen(3000)
```

Installation

This is a **Node.js** module available through the **npm registry**.

Before installing, **download and install Node.js**. Node.js 0.10 or higher is required.

Install

```
> npm i express
```

Repository

github.com/expressjs/express

Homepage

expressjs.com/

Weekly Downloads

19,930,201

Version	License
4.17.3	MIT

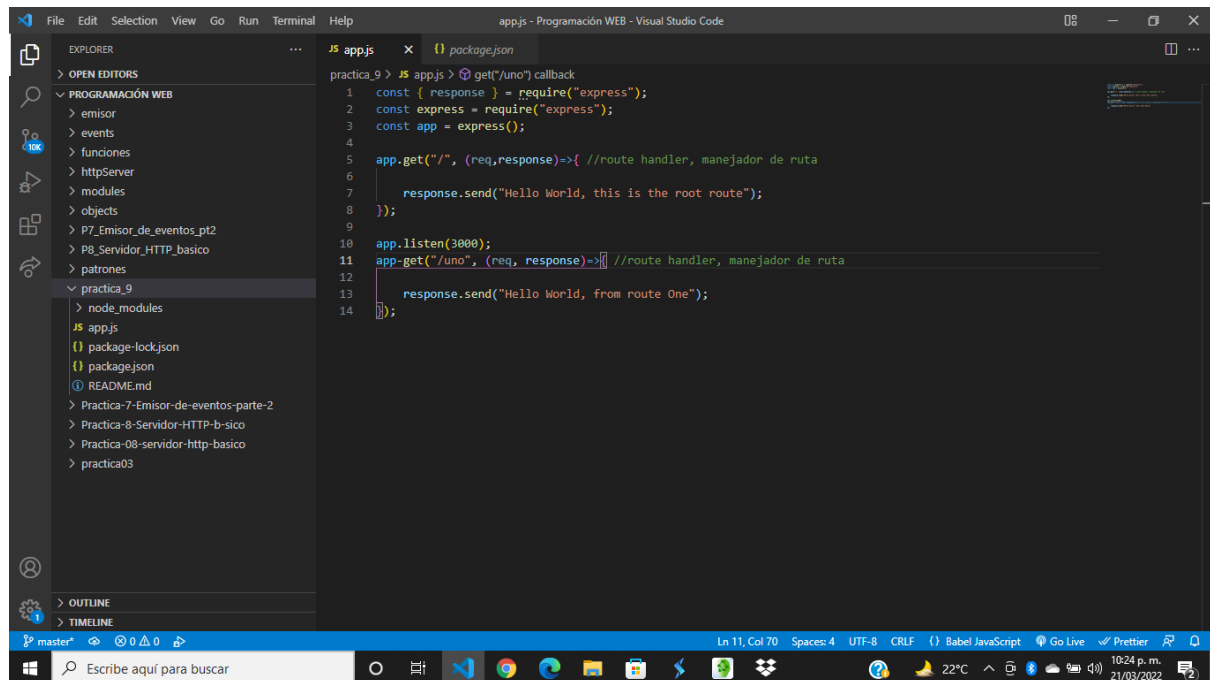
Unpacked Size	Total Files
210 kB	16

Issues	Pull Requests
107	49

Last publish

Paso 4.

Se creó un archivo llamado “app.js” y se implementó el siguiente código.

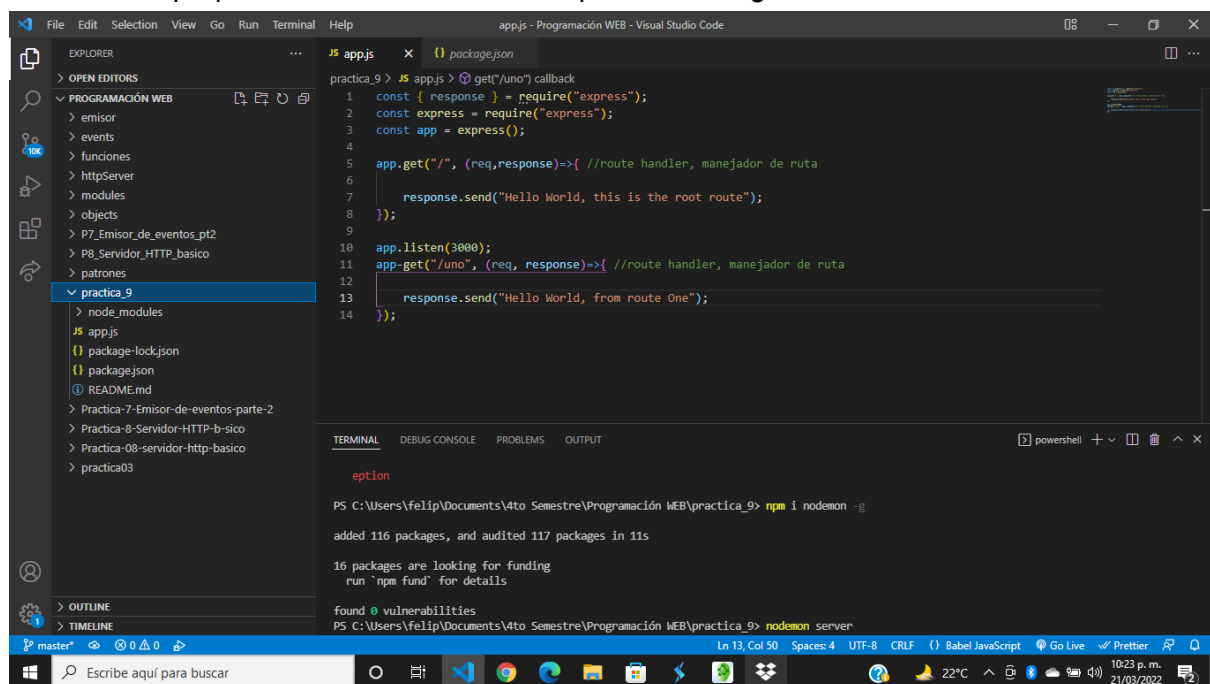


The screenshot shows the Visual Studio Code editor with the file explorer on the left displaying a project structure under 'practica_9'. The main editor window shows the 'app.js' file with the following code:

```
1 const { response } = require("express");
2 const express = require("express");
3 const app = express();
4
5 app.get("/", (req, response) => { //route handler, manejador de ruta
6
7     response.send("Hello World, this is the root route");
8 });
9
10 app.listen(3000);
11 app.get("/uno", (req, response) => { //route handler, manejador de ruta
12
13     response.send("Hello World, from route One");
14 });
```

Paso 5.

Se instala el paquete de nodem mediante, npm i nodem -g

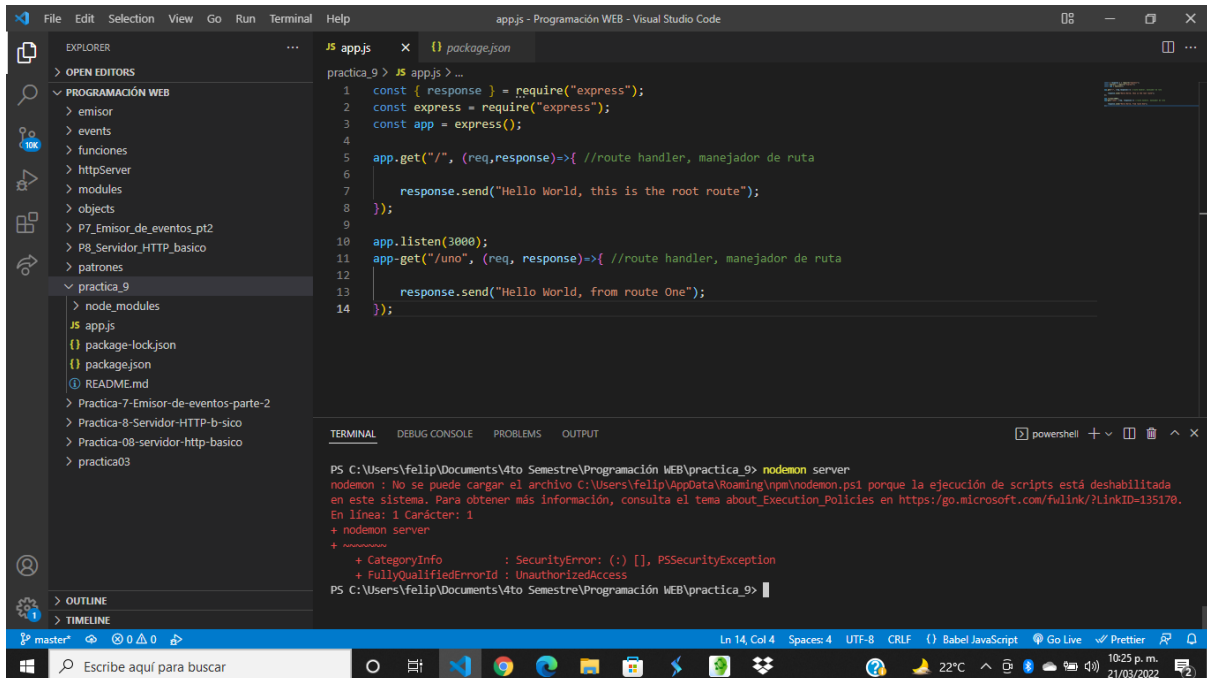


The screenshot shows the Visual Studio Code editor with the terminal window open at the bottom. The terminal displays the output of the command 'npm i nodemon -g' executed in a PowerShell prompt. The output indicates that 116 packages were added and 117 packages were audited. It also shows a message about 16 packages looking for funding and a link to run 'npm fund' for details. Finally, it shows that 0 vulnerabilities were found.

```
PS C:\Users\felip\Documents\4to Semestre\Programación WEB\practica_9> npm i nodemon -g
added 116 packages, and audited 117 packages in 11s
16 packages are looking for funding
run 'npm fund' for details
found 0 vulnerabilities
PS C:\Users\felip\Documents\4to Semestre\Programación WEB\practica_9> nodemon server
```

Paso 6.

Intentando ejecutar el código, apareció el siguiente mensaje de error:



The screenshot shows the Visual Studio Code interface. The Explorer pane on the left shows a project structure with a folder named 'practica_9' containing 'app.js' and 'package-lock.json'. The main editor shows the content of 'app.js', which is a simple Express.js server. The Terminal pane at the bottom shows the command 'node server' being executed, which results in a PowerShell error: 'SecurityError: (:) [], PSSecurityException'.

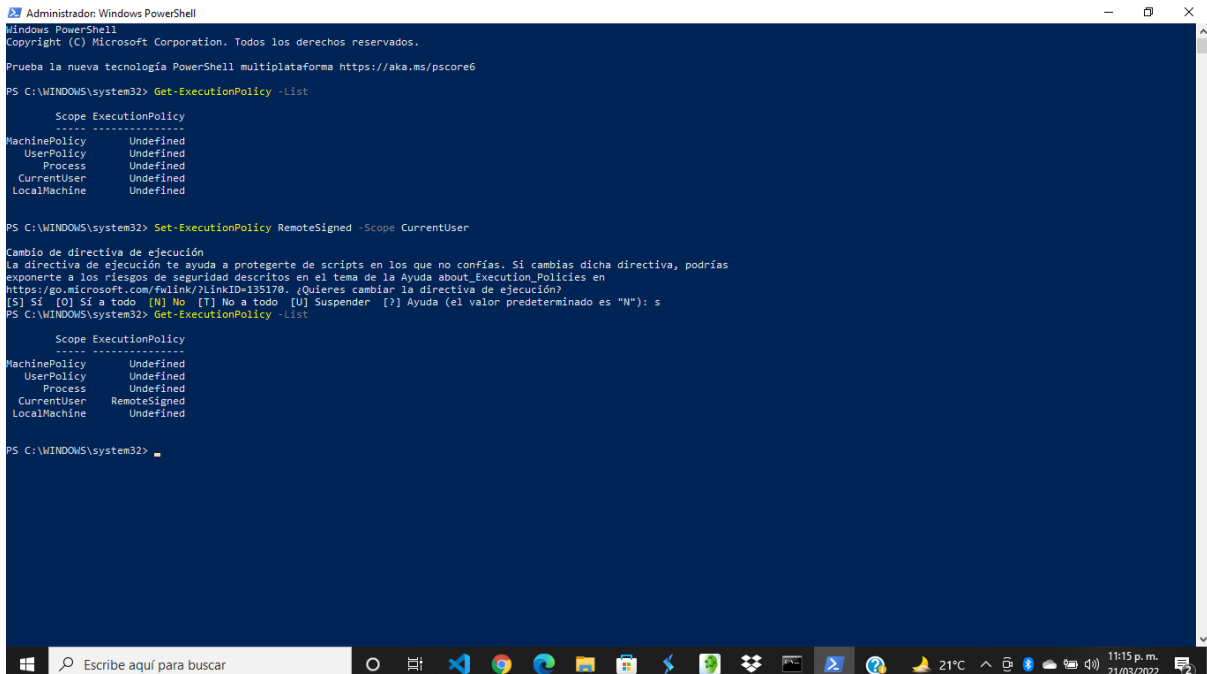
```
practica_9 > JS app.js > ...
1  const { response } = require("express");
2  const express = require("express");
3  const app = express();
4
5  app.get("/", (req, response) => { //route handler, manejador de ruta
6
7      response.send("Hello World, this is the root route");
8  });
9
10 app.listen(3000);
11 app.get("/uno", (req, response) => { //route handler, manejador de ruta
12
13     response.send("Hello World, from route One");
14 });
```

```
PS C:\Users\felip\Documents\4to Semestre\Programación WEB\practica_9> node server
node : No se puede cargar el archivo C:\Users\felip\AppData\Roaming\npm\node_modules\node\bin\node.exe porque la ejecución de scripts está deshabilitada
en este sistema. Para obtener más información, consulta el tema about_Execution_Policies en https://go.microsoft.com/fwlink/?LinkID=135170.
En línea: 1 Carácter: 1
+ node server
+ ~~~~~
+ CategoryInfo          : SecurityError: (:) [], PSSecurityException
+ FullyQualifiedErrorId : UnauthorizedAccess
PS C:\Users\felip\Documents\4to Semestre\Programación WEB\practica_9>
```

Se procedió a buscarle solución en la web.

Paso 7.

El problema se solucionó con el código que se enseña a continuación en el power shell de windows siendo ejecutado como administrador.



The screenshot shows a Windows PowerShell window running several commands to change the execution policy. The 'Get-ExecutionPolicy -List' command shows the current policy for various scopes. The 'Set-ExecutionPolicy RemoteSigned -Scope CurrentUser' command is used to change the policy. The 'Get-ExecutionPolicy -List' command is run again to confirm the change.

```
Administrador Windows PowerShell
Windows PowerShell
Copyright (C) Microsoft Corporation. Todos los derechos reservados.

Prueba la nueva tecnología PowerShell multiplataforma https://aka.ms/pscore6

PS C:\WINDOWS\system32> Get-ExecutionPolicy -List

Scope ExecutionPolicy
-----
MachinePolicy Undefined
UserPolicy Undefined
Process Undefined
CurrentUser Undefined
LocalMachine Undefined

PS C:\WINDOWS\system32> Set-ExecutionPolicy RemoteSigned -Scope CurrentUser

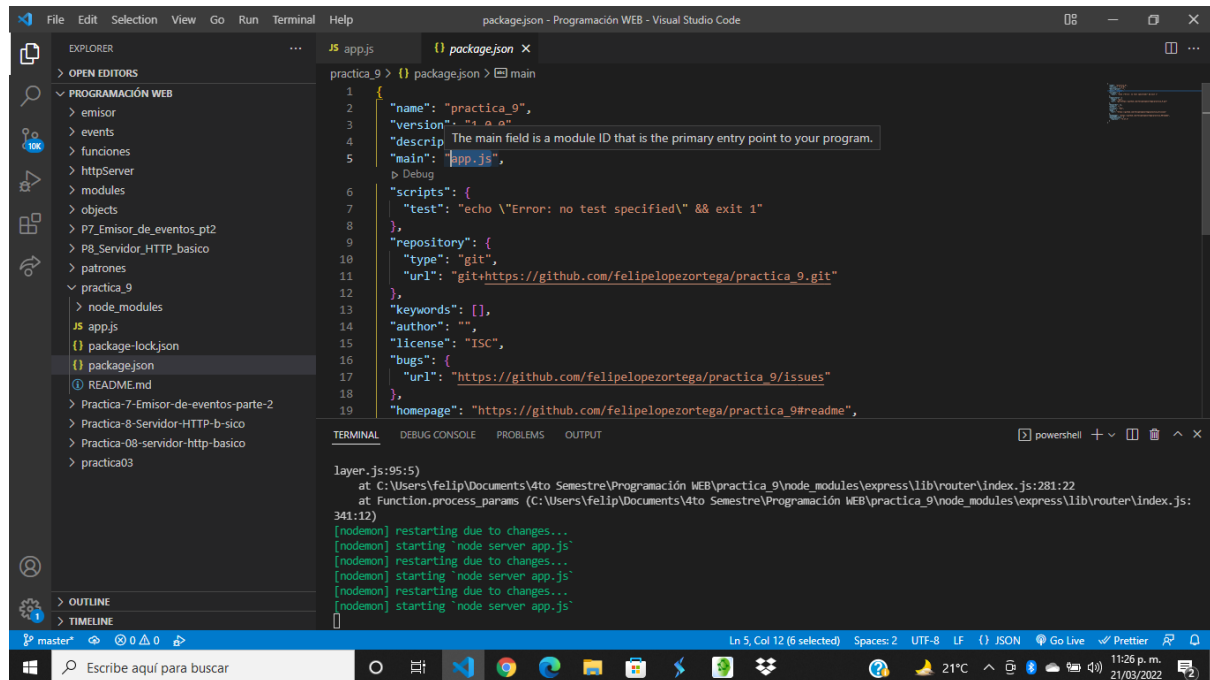
Cambio de directiva de ejecución
La directiva de ejecución te ayuda a protegerte de scripts en los que no confías. Si cambias dicha directiva, podrías
exponerte a los riesgos de seguridad descritos en el tema de la Ayuda about_Execution_Policies en
https://go.microsoft.com/fwlink/?LinkID=135170. ¿Quieres cambiar la directiva de ejecución?
[?] Sí [0] Sí a todo [N] No [1] No a todo [U] Suspender [?] Ayuda (el valor predeterminado es "N"): s
PS C:\WINDOWS\system32> Get-ExecutionPolicy -List

Scope ExecutionPolicy
-----
MachinePolicy Undefined
UserPolicy Undefined
Process Undefined
CurrentUser RemoteSigned
LocalMachine Undefined

PS C:\WINDOWS\system32>
```

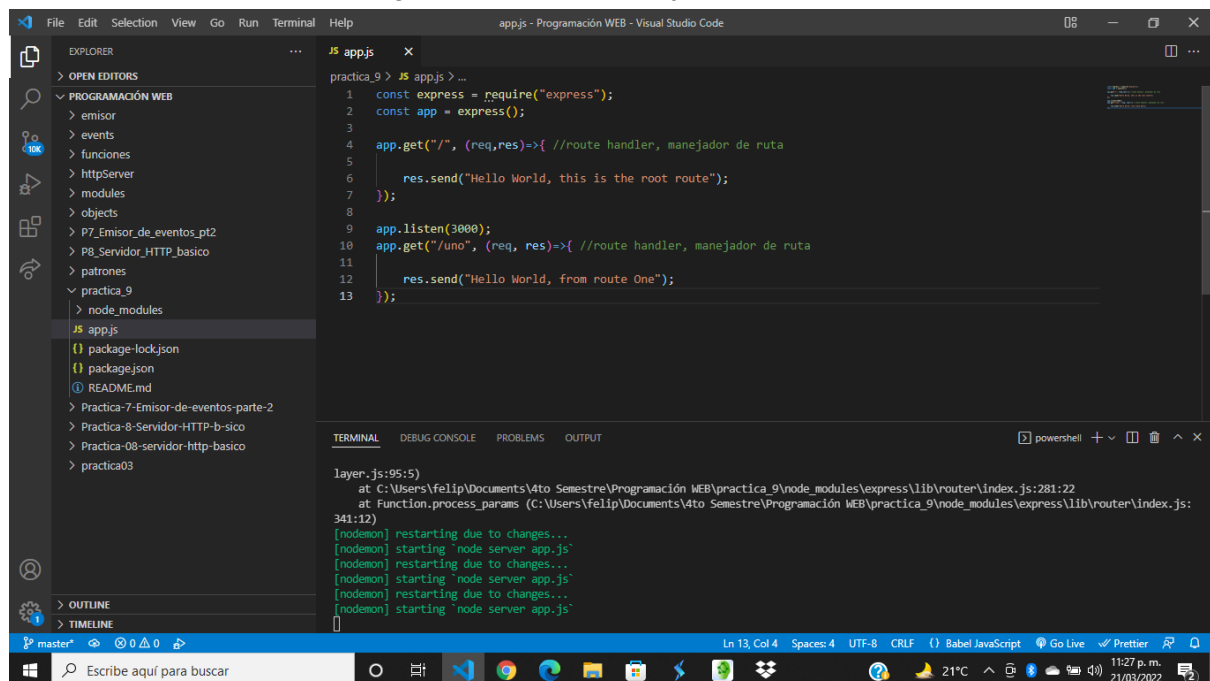
Paso 8

Se cambió el dominio de main en el paquete instalado de “index.js” al que tenemos nosotros, “app.js”.



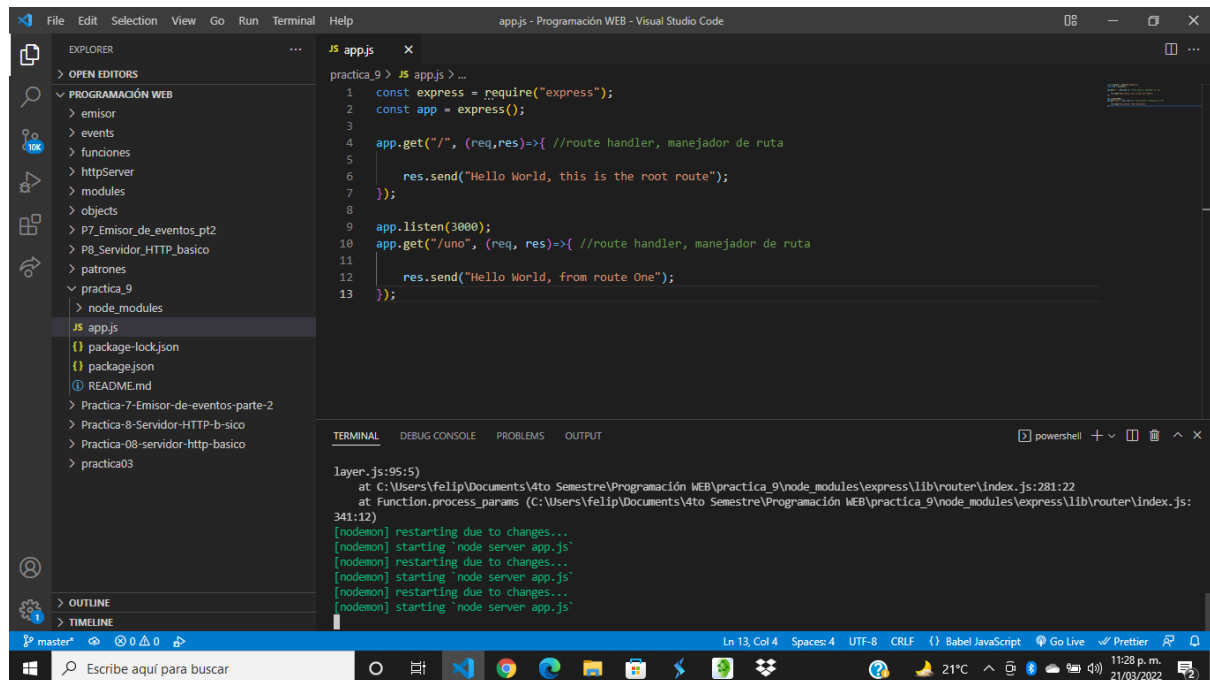
Paso 9

Corrección de errores de código en el archivo “app.js”.



Paso 10.

Ejecutamos en la terminal el comando nodemon server.



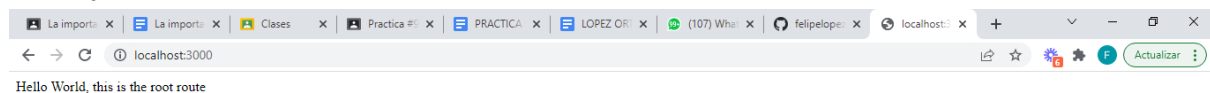
The screenshot shows the Visual Studio Code interface. On the left, the Explorer sidebar shows a project structure with folders like 'PROGRAMACIÓN WEB', 'practica_9', and 'node_modules'. The main editor displays a file named 'app.js' with the following code:

```
1 const express = require("express");
2 const app = express();
3
4 app.get("/", (req,res)=>{ //route handler, manejador de ruta
5
6     res.send("Hello World, this is the root route");
7 });
8
9 app.listen(3000);
10 app.get("/uno", (req, res)=>{ //route handler, manejador de ruta
11
12     res.send("Hello World, from route One");
13 });
```

At the bottom, the Terminal window shows the command 'practica_9 > JS app.js > ...' and the output of the nodemon command, which includes the file path and the message 'Hello World, this is the root route'.

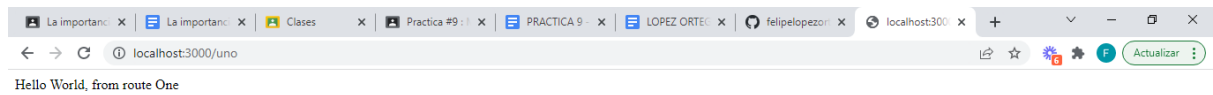
Paso 11

En nuestro navegador web, se colocó la leyenda “localhost:3000” y al dar enter salió el mensaje.



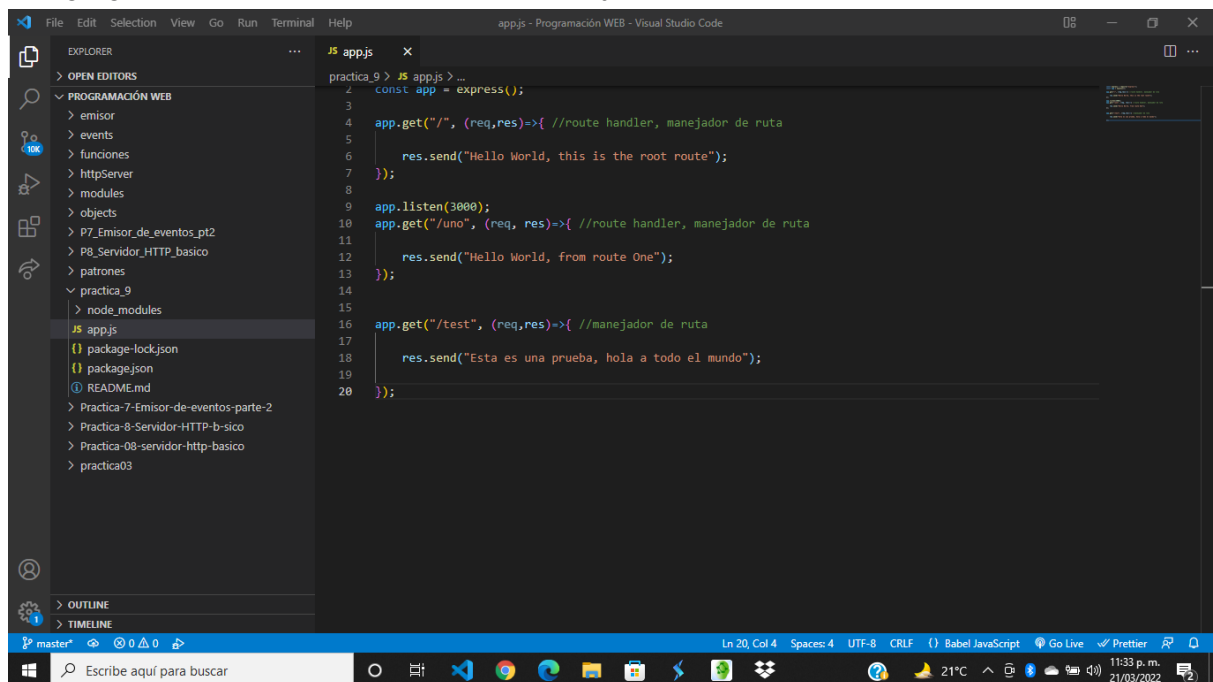
Paso 12

Se cambió de ruta a la 1.



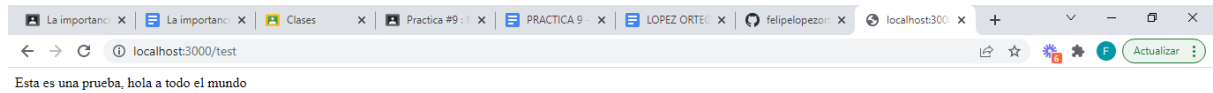
Paso 13

Se agregó una ruta de prueba al archivo "app.js".



Paso 14

En el navegador web, ahora se accedió a esta nueva ruta llamada “test”.



Paso 15

Se agrega contenido HTML al código de la prueba (“test”) y se corre nuevamente en el navegador.

The screenshot displays the Visual Studio Code editor with a project named 'app.js - Programación WEB'. The Explorer sidebar on the left shows a file tree for 'PROGRAMACIÓN WEB' with various folders and files, including 'app.js'. The main editor window shows the content of 'app.js', which is a simple Express.js application:

```
13 });
14
15
16 app.get("/test", (req,res)=>{ //manejador de ruta
17
18     res.send("<h1>Esta es una prueba, hola a todo el mundo</h1>");
19
20 });
21
22
```

Below the editor, the TERMINAL panel shows the output of running the application:

```
7:13)
at Route.dispatch (C:\Users\felip\Documents\4to Semestre\Programación WEB\practica_9\node_modules\express\lib\router\route.js:112:3)
at Layer.handle [as handle_request] (C:\Users\felip\Documents\4to Semestre\Programación WEB\practica_9\node_modules\express\lib\router\layer.js:95:5)
at C:\Users\felip\Documents\4to Semestre\Programación WEB\practica_9\node_modules\express\lib\router\index.js:281:22
at Function.process_params (C:\Users\felip\Documents\4to Semestre\Programación WEB\practica_9\node_modules\express\lib\router\index.js:341:12)
[nodemon] restarting due to changes...
[nodemon] starting 'node server app.js'
```

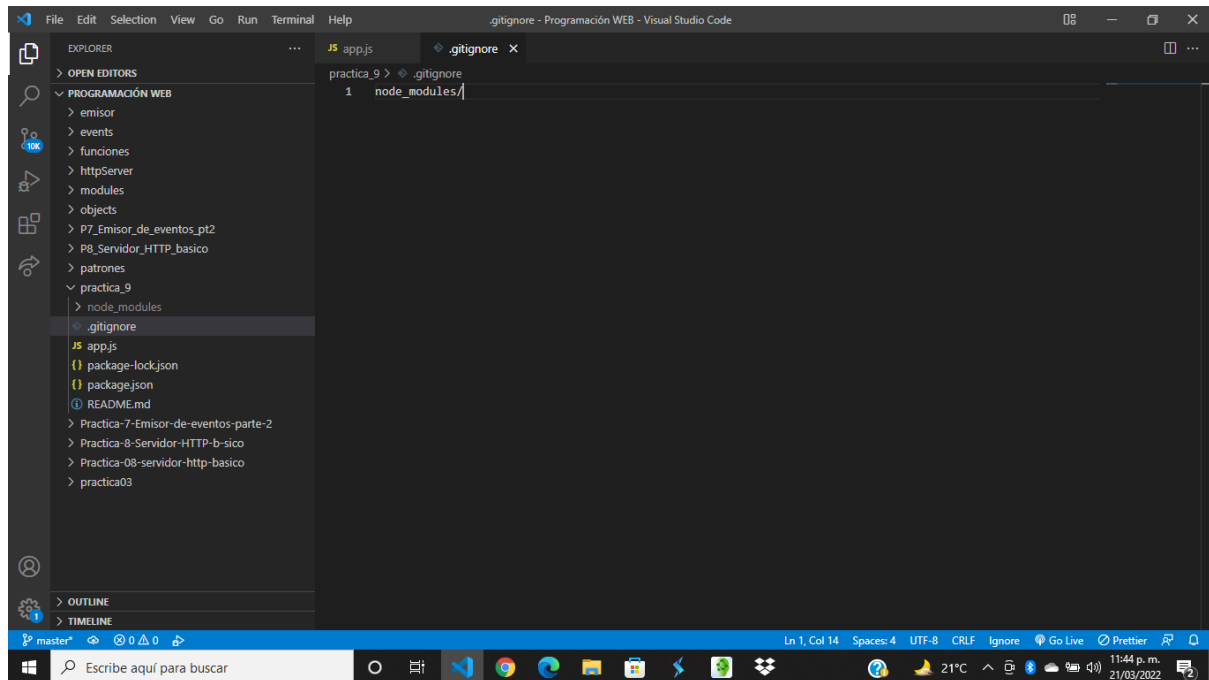
The bottom of the image shows a web browser window with the address bar set to 'localhost:3000/test'. The page content displays the text:

Esta es una prueba, hola a todo el mundo



Paso 16

Se crea el archivo “.gitignore” para no mandar al repositorio el contenido del archivo “node_modules”.



Paso 17

Se sube todo al repositorio.

