



# TECNOLÓGICO NACIONAL DE MÉXICO

INSTITUTO TECNOLÓGICO DE CIUDAD JUÁREZ

DEPARTAMENTO DE SISTEMAS COMPUTACIONALES

**Desarrollo de Aplicaciones Web Reactivas**

Grupo: M6

## Instalación de Node.js y Angular CLI Unidad 2

FECHA DE ENTREGA: 11 DE NOVIEMBRE DE 2024

**Profesor:**

Jaime Alonso Ramos Silva

**Alumno:**

Torres Santos José Ángel

# Índice

<b>1. Introducción</b>	<b>3</b>
<b>2. Instalación de Node.js y Angular</b>	<b>4</b>
2.1. Node.js . . . . .	4
2.2. Angular CLI . . . . .	6
<b>3. Creando proyecto</b>	<b>7</b>
3.1. Lanzando el proyecto de manera local . . . . .	8
<b>4. Componentes y Templates</b>	<b>9</b>
4.1. Componentes . . . . .	9
4.2. Templates . . . . .	9
4.3. Relación entre Componentes y Templates . . . . .	9

# Desarrollo de Aplicaciones Web Reactivas

## Instalación de Node.js y Angular CLI

Torres Santos José Ángel

11 de noviembre de 2024

### Resumen

El objetivo de este reporte es, realizar la instalación de Node.js y Angular así como otras características para un correcto funcionamiento para este ecosistema en Ubuntu. Este laboratorio muestra también un proyecto de ejemplo, confirmado de que todo este en correcto funcionamiento.

## 1. Introducción

Node.js es el motor que impulsa la parte del servidor de muchas aplicaciones web modernas. Su capacidad para manejar múltiples conexiones de manera eficiente y su modelo de E/S no bloqueante lo convierten en una herramienta ideal para construir aplicaciones en tiempo real. Al utilizar Node.js, los desarrolladores pueden crear servidores que respondan rápidamente a las acciones del usuario, lo que es fundamental para lograr una experiencia de usuario fluida y dinámica.

Angular, por su parte, es el marco de trabajo perfecto para construir interfaces de usuario interactivas y responsivas. Con su enfoque basado en componentes y su potente sistema de plantillas, Angular facilita la creación de aplicaciones web de una sola página (SPA) que se sienten más como aplicaciones nativas. Al combinar Angular con Node.js, los desarrolladores pueden construir aplicaciones web reactivas completas, donde los cambios en los datos se reflejan instantáneamente en la interfaz de usuario sin necesidad de recargar toda la página.

Node.js y Angular son tecnologías complementarias que, al trabajar juntas, permiten a los desarrolladores crear aplicaciones web reactivas que ofrecen una experiencia de usuario superior. Node.js se encarga de la lógica del servidor y la gestión de las conexiones en tiempo real, mientras que Angular se enfoca en la construcción de interfaces de usuario dinámicas y responsivas.

Repositorio: <https://github.com/JoseTorresMX/primera-app-angular.git>

## 2. Instalación de Node.js y Angular

### 2.1. Node.js

Para iniciar, necesitamos descargar Node.js, pero como estamos en Ubuntu el proceso sera por medio de *Bash*, es decir, en la terminal.

Descarga: <https://nodejs.org/en/download/package-manager>

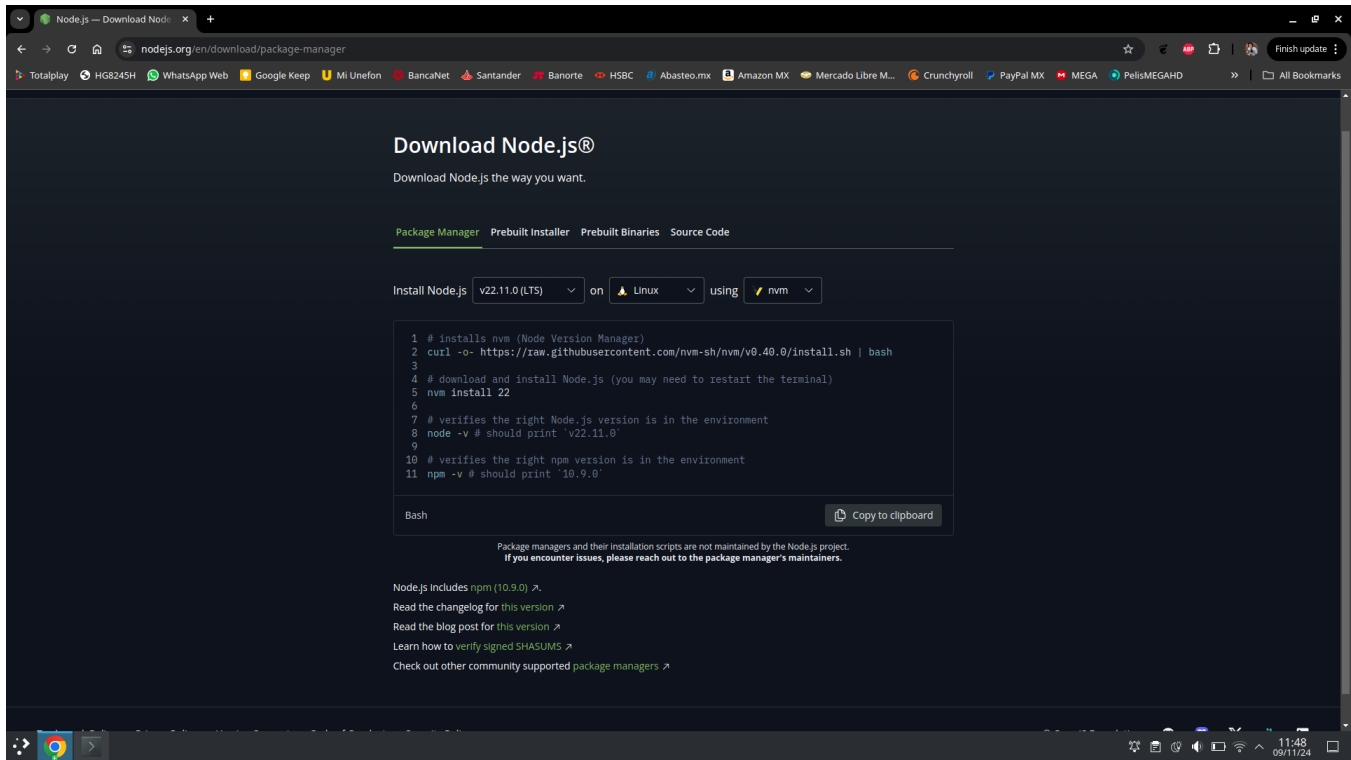


Figura 1: Pagina para de instalación y descarga de Node.js según la plataforma

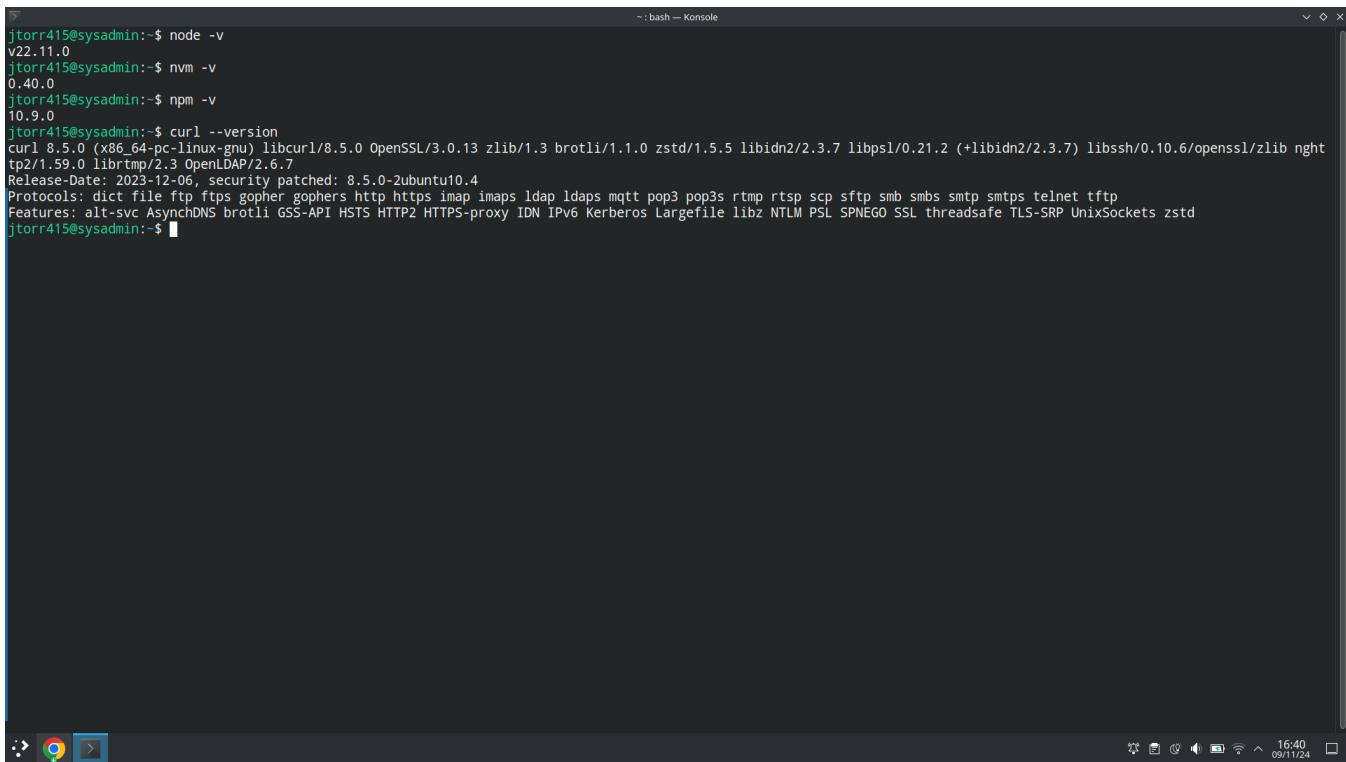
```
# installs nvm (Node Version Manager)
curl -o- https://raw.githubusercontent.com/nvm-sh/nvm/v0.40.0/install.sh | bash

# download and install Node.js (you may need to restart the terminal)
nvm install 22

# verifies the right Node.js version is in the environment
node -v # should print 'v22.11.0'

# verifies the right npm version is in the environment
npm -v # should print '10.9.0'
```

En caso de no tener instalado **curl** y/o **nvm**, es necesario instalarlo mediante la siguiente documentación: Para *curl*: <https://gcore.com/learning/how-to-install-curl-on-ubuntu/> Para *nvm*: <https://www.freecodecamp.org/news/node-version-manager-nvm-install-guide/>

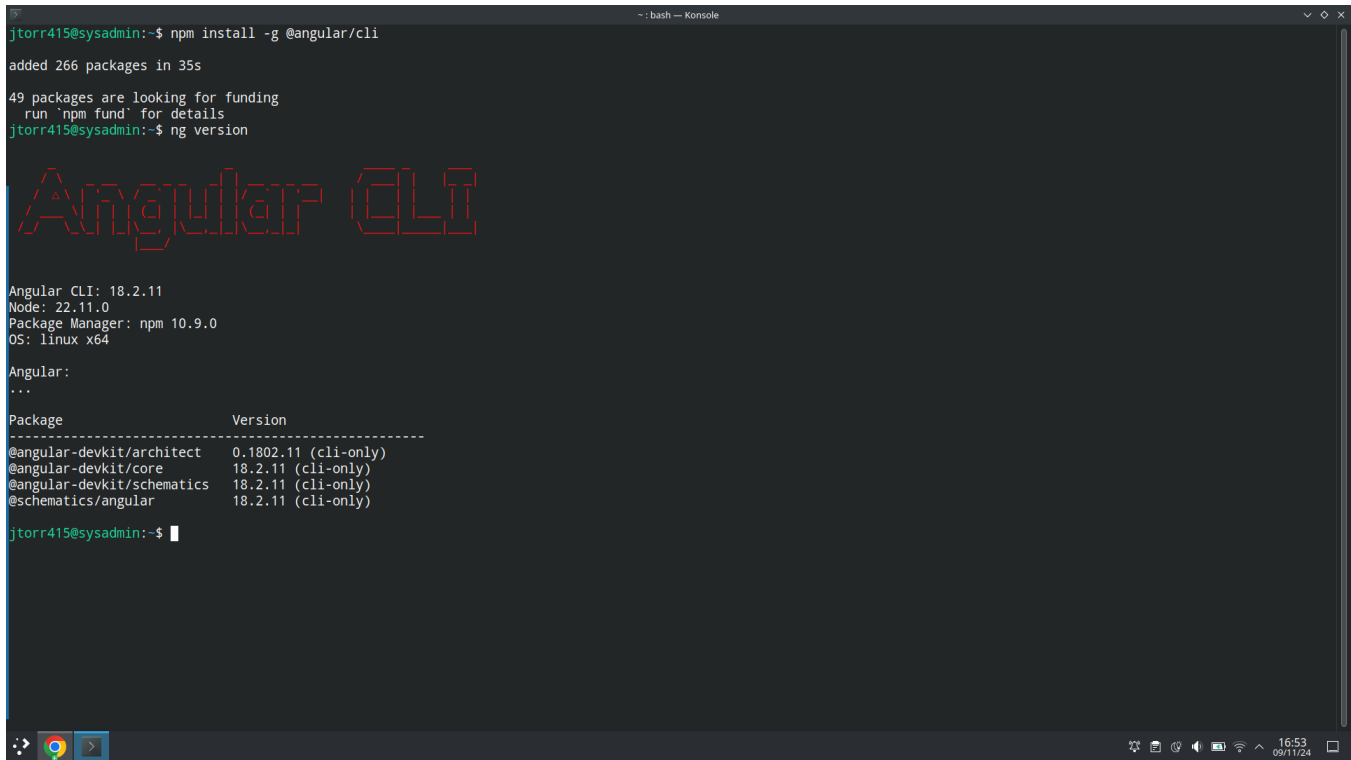


```
jtorr415@sysadmin:~$ node -v
v22.11.0
jtorr415@sysadmin:~$ nvm -v
0.40.0
jtorr415@sysadmin:~$ npm -v
10.9.0
jtorr415@sysadmin:~$ curl --version
curl 8.5.0 (x86_64-pc-linux-gnu) libcurl/8.5.0 OpenSSL/3.0.13 zlib/1.3 brotli/1.1.0 zstd/1.5.5 libidn2/2.3.7 libpsl/0.21.2 (+libidn2/2.3.7) libssh/0.10.6/openssl/zlib nghttp2/1.59.0 librtmp/2.3 OpenLDAP/2.6.7
Release-Date: 2023-12-06, security patched: 8.5.0-2ubuntu10.4
Protocols: dict file ftp ftps gopher gophers http https imap imaps ldap ldaps mqtt pop3 pop3s rtmp rtsp scp sftp smb smbs smtp smtps telnet tftp
Features: alt-svc AsynchDNS brotli GSS-API HSTS HTTP2 HTTPS-proxy IDN IPv6 Kerberos Largefile libz NTLM PSL SPNEGO SSL threadsafe TLS-SRP UnixSockets zstd
```

Figura 2: *Verificando versiones*

## 2.2. Angular CLI

A continuación, instalamos Angular CLI mediante *bash*:



```
jtorr415@sysadmin:~$ npm install -g @angular/cli
added 266 packages in 35s
49 packages are looking for funding
  run 'npm fund' for details
jtorr415@sysadmin:~$ ng version

Angular CLI
Angular CLI: 18.2.11
Node: 22.11.0
Package Manager: npm 10.9.0
OS: linux x64

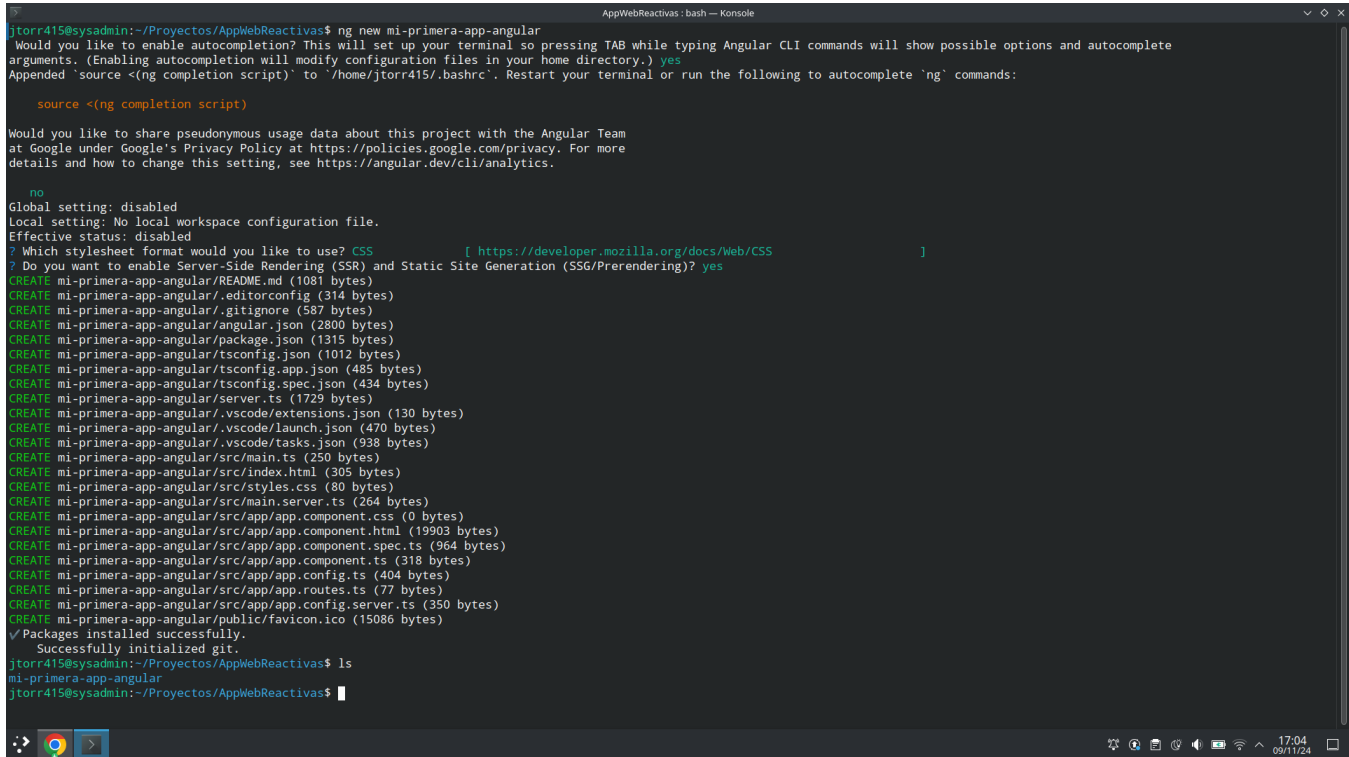
Angular:
...
Package      Version
-----
@angular-devkit/architect    0.1802.11 (cli-only)
@angular-devkit/core         18.2.11 (cli-only)
@angular-devkit/schematics   18.2.11 (cli-only)
@schematics/angular          18.2.11 (cli-only)

jtorr415@sysadmin:~$
```

Figura 3: *Instalando Angular CLI*

### 3. Creando proyecto

Para crear nuestro primer proyecto, se es necesario usar el comando **ng new** seguido del nombre del proyecto, si el nombre del proyecto tiene espacios, procura usar guiones en lugar de espacios:



```
jtorr415@sysadmin: ~/Proyectos/AppWebReactivas$ ng new mi-primer-app-angular
Would you like to enable autocomplete? This will set up your terminal so pressing TAB while typing Angular CLI commands will show possible options and autocomplete arguments. (Enabling autocomplete will modify configuration files in your home directory.) yes
Appended 'source <(ng completion script)>' to '/home/jtorr415/.bashrc'. Restart your terminal or run the following to autocomplete 'ng' commands:

source <(ng completion script)

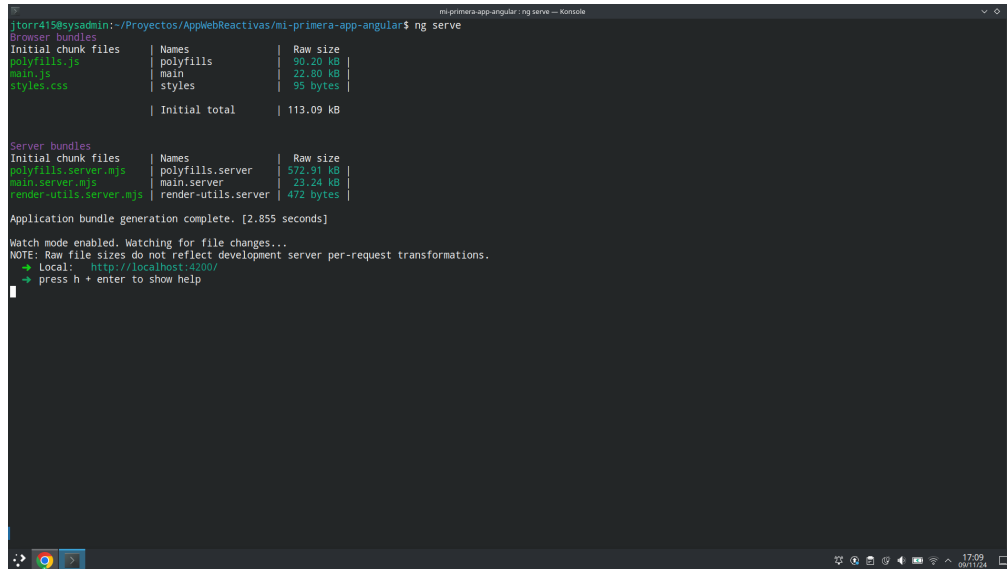
Would you like to share pseudonymous usage data about this project with the Angular Team at Google under Google's Privacy Policy at https://policies.google.com/privacy. For more details and how to change this setting, see https://angular.dev/cli/analytics.

no
Global setting: disabled
Local setting: No local workspace configuration file.
Effective status: disabled
? Which stylesheet format would you like to use? CSS [ https://developer.mozilla.org/docs/Web/CSS ]
? Do you want to enable Server-Side Rendering (SSR) and Static Site Generation (SSG/Prerendering)? yes
CREATE mi-primer-app-angular/README.md (1081 bytes)
CREATE mi-primer-app-angular/.editorconfig (314 bytes)
CREATE mi-primer-app-angular/.gitignore (587 bytes)
CREATE mi-primer-app-angular/angular.json (2800 bytes)
CREATE mi-primer-app-angular/package.json (1315 bytes)
CREATE mi-primer-app-angular/tsconfig.json (1012 bytes)
CREATE mi-primer-app-angular/tsconfig.app.json (485 bytes)
CREATE mi-primer-app-angular/tsconfig.spec.json (434 bytes)
CREATE mi-primer-app-angular/server.ts (1729 bytes)
CREATE mi-primer-app-angular/.vscode/extensions.json (130 bytes)
CREATE mi-primer-app-angular/.vscode/launch.json (470 bytes)
CREATE mi-primer-app-angular/.vscode/tasks.json (938 bytes)
CREATE mi-primer-app-angular/src/main.ts (250 bytes)
CREATE mi-primer-app-angular/src/index.html (305 bytes)
CREATE mi-primer-app-angular/src/styles.css (80 bytes)
CREATE mi-primer-app-angular/src/main.server.ts (264 bytes)
CREATE mi-primer-app-angular/src/app/app.component.css (0 bytes)
CREATE mi-primer-app-angular/src/app/app.component.html (19903 bytes)
CREATE mi-primer-app-angular/src/app/app.component.spec.ts (964 bytes)
CREATE mi-primer-app-angular/src/app/app.component.ts (318 bytes)
CREATE mi-primer-app-angular/src/app/app.config.ts (404 bytes)
CREATE mi-primer-app-angular/src/app/app.routes.ts (77 bytes)
CREATE mi-primer-app-angular/src/app/app.config.server.ts (350 bytes)
CREATE mi-primer-app-angular/public/favicon.ico (15086 bytes)
✓ Packages installed successfully.
Successfully initialized git.
jtorr415@sysadmin: ~/Proyectos/AppWebReactivas$ ls
mi-primer-app-angular
jtorr415@sysadmin: ~/Proyectos/AppWebReactivas$
```

Figura 4: *Creando proyecto*

### 3.1. Lanzando el proyecto de manera local

A continuación, lanzamos el proyecto con los comandos **ng serve**, este nos crea un servidor (localhost) local, el cual se le adjunta un puerto en el cual podemos acceder a nuestro proyecto:



```
jtorr415@sysadmin:~/Proyectos/AppWebReactivas/mi-primera-app-angular$ ng serve
Initial chunk files | Names | Raw size
polyfills.js | polyfills | 90.20 kB
main.js | main | 22.80 kB
styles.css | styles | 95 bytes
| Initial total | 113.09 KB

Server bundles
Initial chunk files | Names | Raw size
polyfills.server.mjs | polyfills.server | 572.91 kB
main.server.mjs | main.server | 23.24 kB
render-utils.server.mjs | render-utils.server | 472 bytes

Application bundle generation complete. [2.855 seconds]

Watch mode enabled. Watching for file changes...
NOTE: Raw file sizes do not reflect development server per-request transformations.
→ Local: http://localhost:4200/
→ press h + enter to show help
```

Figura 5: *Creando y lanzando el servidor*

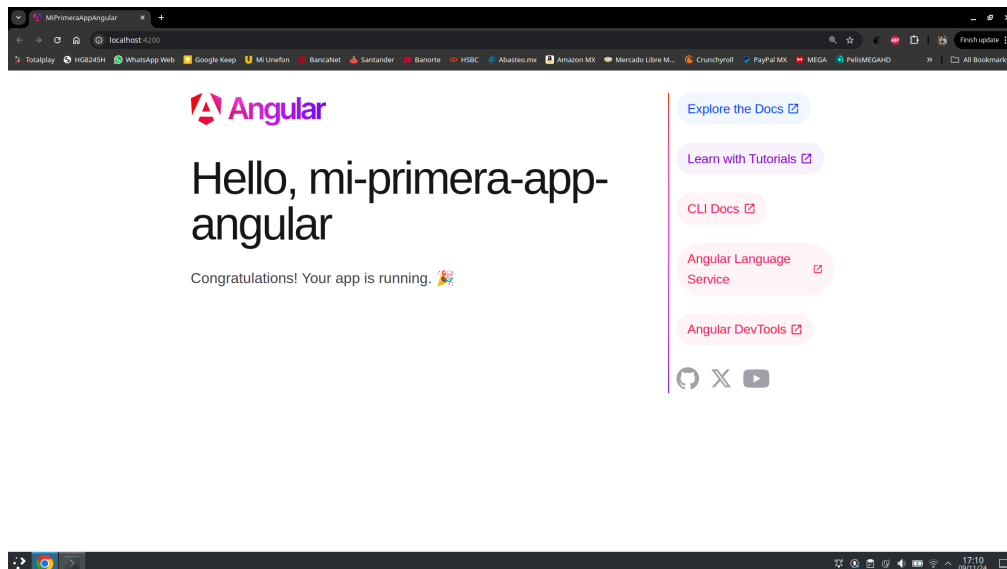


Figura 6: *Visualizando proyecto en nuestro servidor local*

Con esto terminamos la primera parte de este laboratorio.



## 4. Componentes y Templates

En Angular, los componentes y templates son conceptos fundamentales que trabajan en conjunto para construir las interfaces de usuario de nuestras aplicaciones.

### 4.1. Componentes

Imagina que estás construyendo una casa. Los componentes son como los ladrillos de esa casa. Cada componente representa una parte independiente y reutilizable de la interfaz de usuario, como un botón, una lista, un formulario, un encabezado, etc.

- Estructura: un componente esta compuesto por tres partes; plantilla, clase TypeScript y estilo.
- Reutilización: La gran ventaja de los componentes es que pueden ser reutilizados en diferentes partes de la aplicación, lo que promueve la modularidad y la organización del código.
- Encapsulación: Cada componente tiene su propio ámbito, lo que significa que los cambios dentro de un componente no afectan a otros componentes, a menos que estén diseñados para interactuar.

### 4.2. Templates

Los templates son como los planos de construcción de cada componente. Son archivos HTML que definen la estructura visual y el contenido del componente. Dentro de los templates, podemos utilizar:

- Interpolación de datos: Mostrar datos de la clase TypeScript del componente directamente en el HTML.
- Directivas: Agregar comportamiento a los elementos del DOM, como la directiva ngFor para iterar sobre listas o la directiva ngIf para mostrar u ocultar elementos condicionalmente.
- Eventos: Capturar eventos del usuario, como clics, cambios de valor, etc., y ejecutar la lógica correspondiente en la clase TypeScript.

### 4.3. Relación entre Componentes y Templates

La relación entre componentes y templates es estrecha. La clase TypeScript del componente controla la lógica y los datos, mientras que el template muestra esa información al usuario. Cuando un usuario interactúa con un componente, se desencadena un evento que es capturado por la clase TypeScript, la cual puede actualizar los datos y, por lo tanto, la vista del componente.

A continuaciones, se agrega contenido al *componentes*, renderizar en el *template* y se verifica el resultado

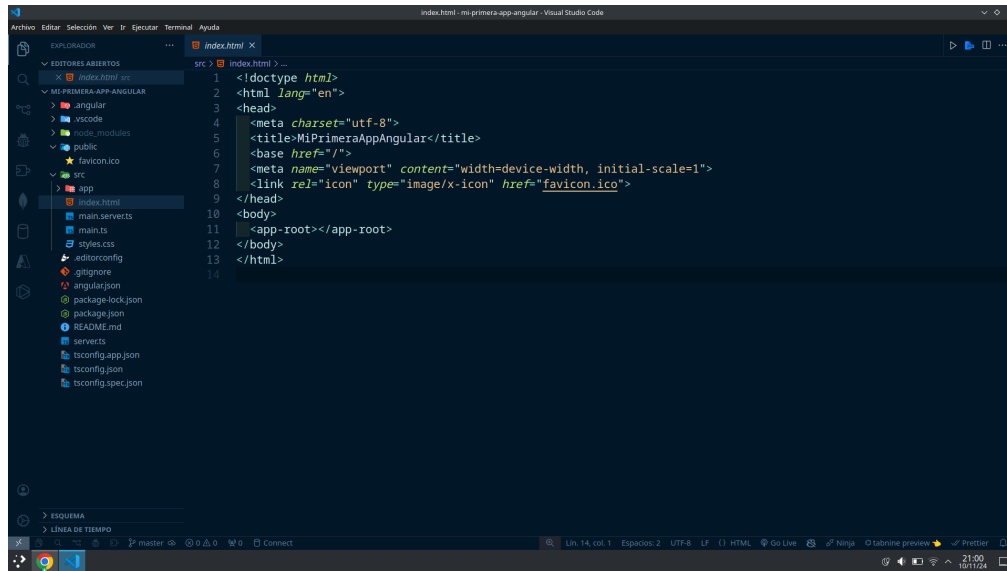


Figura 7: Proyecto en VSCode

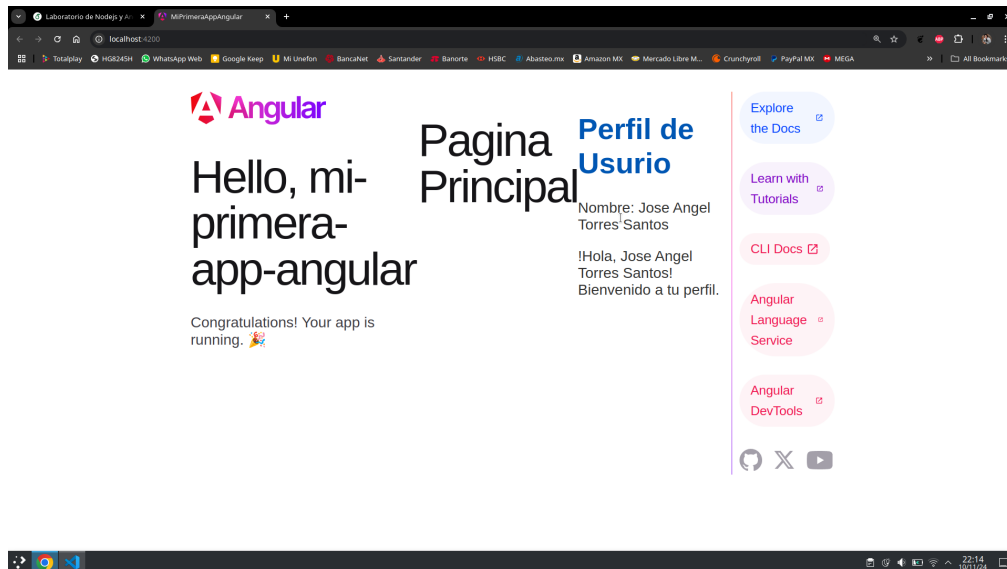


Figura 8: Pagina principal renderizada

Los créditos de las fotografías pertenecen a sus respectivos autores.