

Proyecto Final de Lightning-um

Instalación de `#nvm` `#nodejs` `#angular`

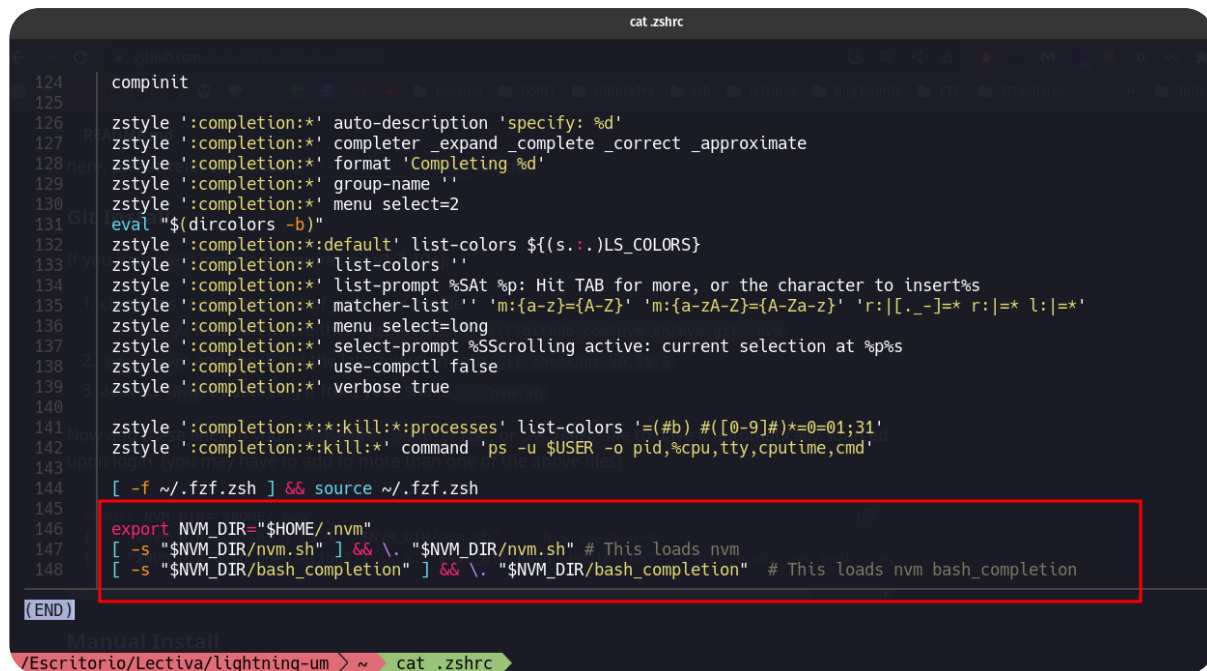
Repositorio [GITHUB](#)

Ejecutamos los siguientes comandos desde la terminal

```
export NVM_DIR="$HOME/.nvm" && (  
  git clone https://github.com/nvm-sh/nvm.git "$NVM_DIR"  
  cd "$NVM_DIR"  
  git checkout `git describe --abbrev=0 --tags --match "v[0-9]*" $(git rev-list --tags --max-count=1)`  
) && \. "$NVM_DIR/nvm.sh"
```

Esto lo colocamos en `.zshrc`

```
export NVM_DIR="$HOME/.nvm"  
[ -s "$NVM_DIR/nvm.sh" ] && \. "$NVM_DIR/nvm.sh" # This loads nvm  
[ -s "$NVM_DIR/bash_completion" ] && \. "$NVM_DIR/bash_completion" # This loads nvm  
bash_completion
```

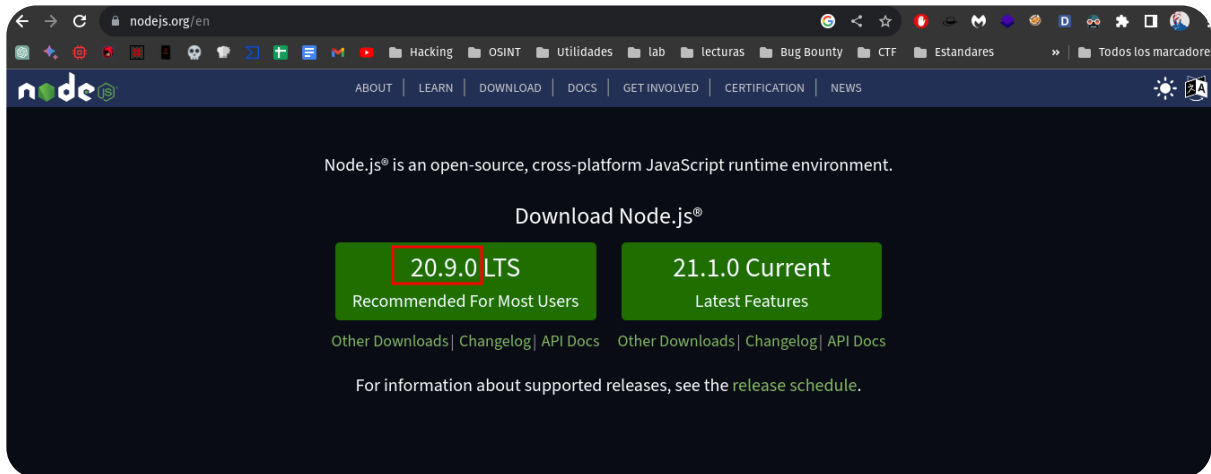


```
cat .zshrc  
  
124 compinit  
125  
126 zstyle ':completion:*' auto-description 'specify: %d'  
127 zstyle ':completion:*' completer _expand _complete _correct _approximate  
128 zstyle ':completion:*' format 'Completing %d'  
129 zstyle ':completion:*' group-name ''  
130 zstyle ':completion:*' menu select=2  
131 eval "$(dircolors -b)"  
132 zstyle ':completion:*:default' list-colors ${(s.:).LS_COLORS}  
133 zstyle ':completion:*' list-colors ''  
134 zstyle ':completion:*' list-prompt %Sat %p: Hit TAB for more, or the character to insert%s  
135 zstyle ':completion:*' matcher-list '' 'm:{a-z}={A-Z}' 'm:{a-zA-Z}={A-Za-z}' 'r:|[_-]* r:|=* l:|=*'  
136 zstyle ':completion:*' menu select=long  
137 zstyle ':completion:*' select-prompt %SScrolling active: current selection at %p%s  
138 zstyle ':completion:*' use-compctl false  
139 zstyle ':completion:*' verbose true  
  
140  
141 zstyle ':completion:*:kill:*:processes' list-colors '=(#b) #([0-9])*=0=01;31'  
142 zstyle ':completion:*:kill:*' command 'ps -u $USER -o pid,%cpu,tty,cputime,cmd'  
143  
144 [ -f ~/.fzf.zsh ] && source ~/.fzf.zsh  
145  
146 export NVM_DIR="$HOME/.nvm"  
147 [ -s "$NVM_DIR/nvm.sh" ] && \. "$NVM_DIR/nvm.sh" # This loads nvm  
148 [ -s "$NVM_DIR/bash_completion" ] && \. "$NVM_DIR/bash_completion" # This loads nvm bash_completion  
  
(END)  
Manual Install  
/Escritorio/Lectura/lightning-um > ~ cat .zshrc
```

Recargamos la configuración de zsh `source .zshrc`

1. Instalación de node.js

Después de esto instalamos la última versión de node.js



Con el comando `nvm install v20.9.0` recargamos y miramos con `node -v`

2. instalación de npm -v

tiramos el siguiente comando `npm install -g @angular/cli@latest`

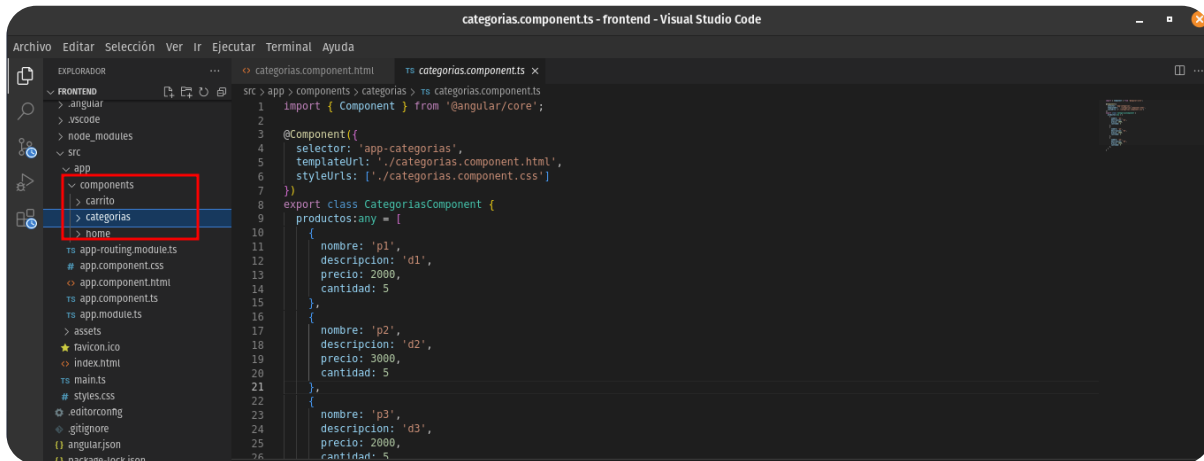
Después de esto pasamos a crear nuestro proyecto en Angular

nos dirigimos a ruta desde donde vamos a trabajar en mi caso `cd Escritorio/Lectiva` y creamos la estructura del proyecto con `ng new frontend`

Posterior a esto creamos la index de las páginas, en nuestro caso creamos tres de la siguiente manera desde la terminal.

1 `ng g c components/home` 2 `ng g c components/categorias` 3 `ng g c componets/carrito`

En este punto cuando ustedes abren el proyecto con visual studio code les debería aparecer en la estructura, acá es donde estructuramos el diseño, utilizar HTML ,Css.

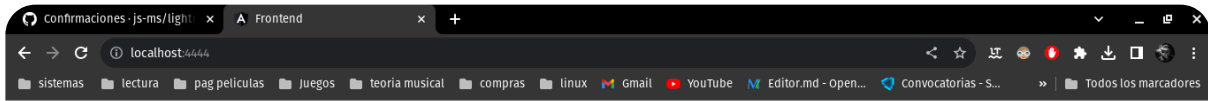


Posterior a esto montamos un servidor local , con `ng` que nos permite ver en tiempo real lo que estamos realizando en frontend.

Desde terminal ejecutamos el comando `ng serve --port 4444` el puerto puede ser cualquiera.

Nota: De ejecutar desde la ruta donde está el proyecto.

```
> cd frontend
> ls
node_modules {} angular.json @ package.json {} tsconfig.app.json {} tsconfig.spec.json
src @ package-lock.json * README.md {} tsconfig.json
> ng serve --port 4444
:: Generating browser application bundles (phase: setup)...]
```



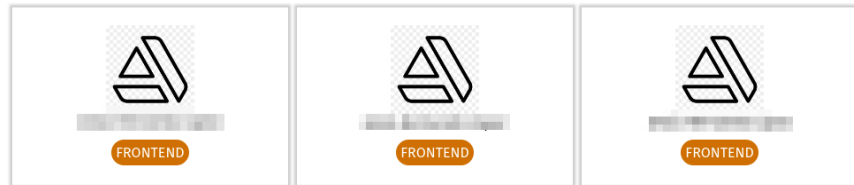
Proyecto

Lorem ipsum dolor sit amet consectetur adipisicing elit. Quibusdam, repudiandae facilis. Natus, cumque quos minus dolor deleniti quas beatae minima nihil cum dolorum obcaecati tempora maxime mollitia culpa, labore sed.

Modi fuga ratione dolore molestias et, libero hic, ipsum quaerat similique eum suscipit voluptate quam maiores, impedit fugit atque nobis sed exercitationem optio pariatur. Itaque numquam nisi odit tempore repudiandae!

Rerum eligendi neque harum aperiam, provident dicta. Quae dolorem veniam non tenetur sequi voluptatem obcaecati nisi aut temporibus, nemo quo quaerat officia officiis voluptates. Exercitationem laboriosam magni dolores quia necessitatibus?

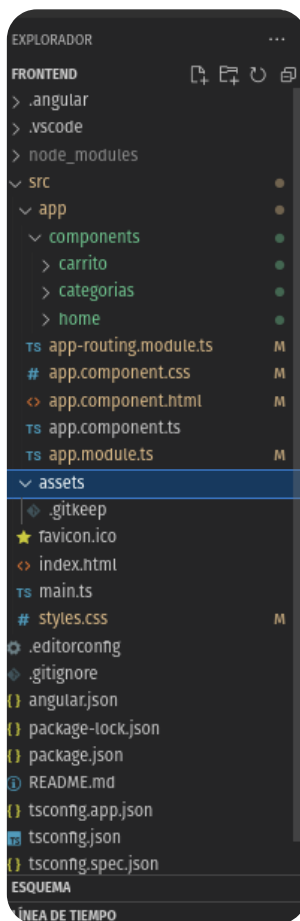
Aut sequi odio nesciunt atque? Qui molestias in, ducimus alias quam quis inventore praesentium ex suscipit sapiente magni aliquam necessitatibus ea ipsum commodi maiores nulla corporis. Iste magni accusantium voluptates.



Explicación de la estructura del proyecto

Repositorio del código: [github-js-ms/lightning-um](#)

Carpeta Frontend



- El archivo `package.json` es el archivo donde se manejan todas las librerías.
 - Por ejemplo, `angular route` para generar rutas de nuestros componentes creados.
 - Aquí estarán almacenadas todas las librerías que importemos para el proyecto.
 - Algo a tener en cuenta es que esto se puede hacer de forma automática desde terminal para que se agregue a este directorio.

En la carpeta `SRC` tenemos lo siguiente, encontraremos una carpeta `assets` donde normalmente se almacena toda la información, imágenes, estilos, etc. El siguiente es el fichero `index.html` donde centraremos una línea que nos llama la atención, definida como `root`, la que se encarga de renderizar toda la página que hemos montado. El fichero `typescript` también nos permite renderizar en fin.

En este caso lo que nos interesa es la carpeta `app`, donde tenemos el `component.html`, para mostrar toda la estructura de lo que tenemos enrutado en este caso las tres categorías mencionadas anteriormente. Otro fichero sería el `app.component.ts` que tiene la lógica y la URL, por último acá debemos definir una variable `componet`, para los selectos que está en el `index.html`. Otro componente que estamos definiendo desde acá es para la URL y el estilo.

Archivo `package.json` :

El archivo `package.json` es fundamental en cualquier proyecto de `Node.js`. Contiene información sobre el proyecto y las dependencias que utiliza. Por ejemplo, si estamos trabajando con `Angular`, aquí es donde especificamos las versiones de `Angular` y otras bibliotecas que estamos utilizando en nuestro proyecto. También se pueden definir scripts personalizados que facilitan tareas como iniciar el servidor o compilar el código.

Carpeta src :

En la carpeta `src` encontramos varios elementos clave:

1. Carpeta `assets` :

Aquí es donde almacenamos recursos como imágenes, estilos, fuentes, entre otros. Estos recursos pueden ser utilizados en nuestro proyecto y se pueden importar en nuestros componentes y plantillas.

2. Archivo `index.html` :

Este archivo es el punto de entrada de nuestra aplicación web. Contiene el marcado HTML inicial y se encarga de cargar los archivos JavaScript y CSS necesarios para nuestra aplicación. También es donde se define el punto de montaje de la aplicación, comúnmente conocido como "root", donde Angular renderizará la interfaz de usuario.

3. Archivos TypeScript:

Los archivos TypeScript contienen el código fuente de nuestra aplicación. Pueden incluir componentes, servicios, directivas, entre otros. TypeScript es un lenguaje de programación que compila a JavaScript y es utilizado para desarrollar aplicaciones Angular de manera más estructurada y con características adicionales.

4. Carpeta `app` :

En esta carpeta encontramos los elementos principales de nuestra aplicación Angular:

- **`component.html` :** Aquí se define la estructura de los componentes. Este archivo contiene el marcado HTML que será renderizado cuando el componente se utilice en la aplicación.
- **`app.component.ts` :** En este archivo se encuentra la lógica del componente. Aquí definimos propiedades y métodos que serán utilizados en el componente. También se pueden manejar eventos y comunicarse con servicios.
- **Otras carpetas y archivos:** Dependiendo de la complejidad del proyecto, es posible que tengamos más carpetas y archivos en la carpeta `app`, como subcomponentes, servicios, directivas, entre otros.

Es importante mencionar que Angular es un framework para construir aplicaciones web de una sola página (SPA) y se basa en el lenguaje TypeScript. Utiliza componentes para construir la interfaz de usuario y ofrece una amplia gama de funcionalidades para desarrollar aplicaciones modernas y escalables.

```
src > app > ts app.component.ts > AppComponent
1  import { Component } from '@angular/core';
2
3  @Component({
4    selector: 'app-root',
5    templateUrl: './app.component.html',
6    styleUrls: ['./app.component.css']
7  })
8  export class AppComponent {
9    | title = 'frontend';
10 }
11
```

Otro directorio a tener en cuenta, el `app.modelo`, que básicamente nos permite gestionar todo, donde importamos los módulos y declaramos los componentes.

Como explicación final en cada componente a nivel de visualización está con `css` y `html`. Si tiene dudas en fragmentos de código, le recomendaría pasárselo a una IA, le argumenta mucho mejor de que se trata para no dar mucha largue a este archivo.

1. Material de apoyo

- [Curso de Angular youtube ↗](#)
- [Aprender un poco de síntesis web ↗](#)