

Informe de Laboratorio 09

Tema: Angular

Nota

Estudiante	Escuela	Asignatura
Luis Guillermo Luque Condori, Fernando Miguel Garambel Marín, William Herderson Choquehuanca Berna, Jeans Anthony Ajra Huacso lluquecon@unsa.edu.pe fgarambel@unsa.edu.pe jajra@unsa.edu.pe wchoquehuancab@unsa.edu.pe	Escuela Profesional de Ingeniería de Sistemas	Laboratorio de P.Web Semestre: III

Laboratorio	Tema	Duración
09	Angular	04 horas

Semestre académico	Fecha de inicio	Fecha de entrega
2024 - A	Del 18 de junio de 2024	Al 22 de junio de 2024

1. REPOSITORIO GitHub

- <https://github.com/nuevo637/Lab-09-Angular>

2. Videos

- A
- F
- L
- W

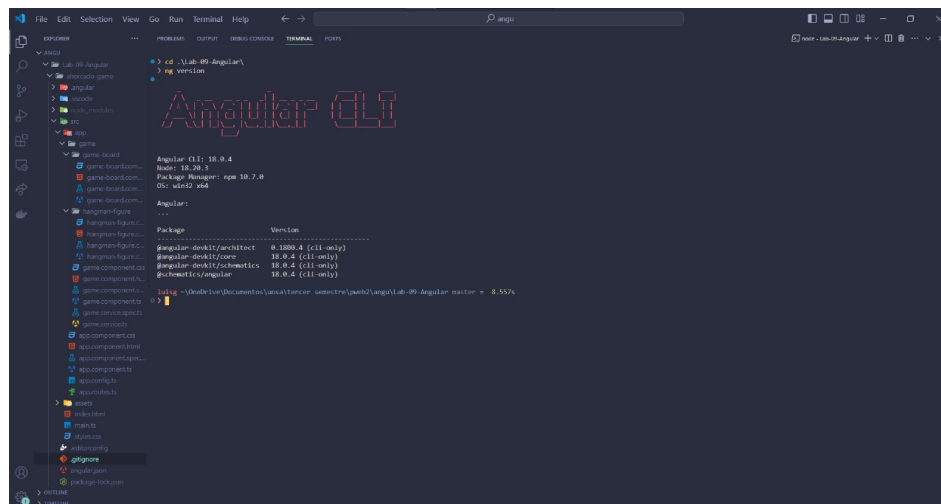
3. Introducción Angular

- Primero tenemos que tener una version de TypeScript de la 17 en adelante para poder instalar Angular.
- Luego usamos los siguientes comandos para instalarlo y verificar la instalación.

Listing 1: Comandos para instalar Angular

```
$ npm install -g @angular/cli
$ ng version
```

- si nos sale la siguiente imagen angular estará instalado



- para crear un proyecto en angular usamos la siguiente linea de comandos.

Listing 2: Crear Proyecto en Angular

```
$ ng new ahorcado-game
```

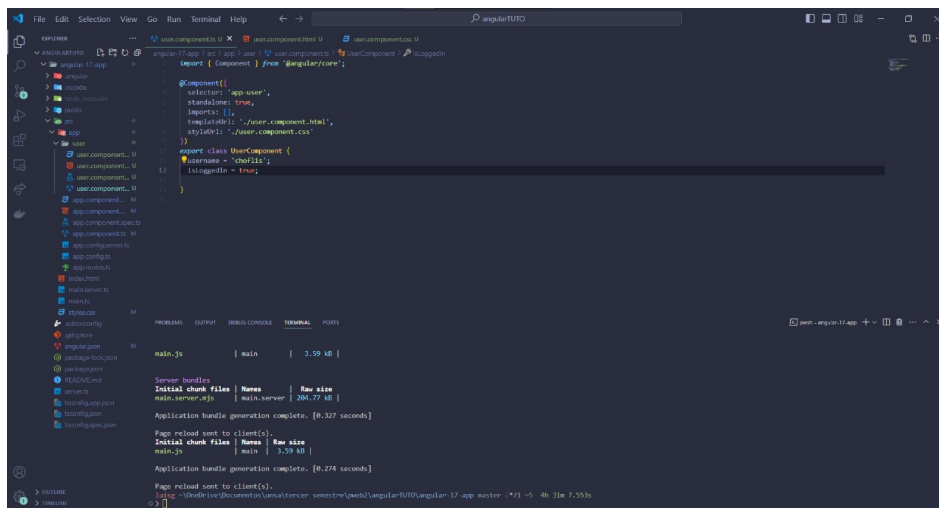
- Luego observaremos un archivo padre que seria nuestra app donde tendremos estilos que son genericos, html y un archivo de componentes que nos ayudaran a juntar mas componentes después.

- Los componentes nos ayudan a dar valores y lógica a nuestra página web.
- Ahora para crear un componente usaremos la siguiente línea de comandos.

Listing 3: Crear Componentes

```
$ ng generate component game-board
$ ng generate component hangman-figure
```

- Esto nos ayuda para poder trabajar por partes y no tenerlo todo en un solo código además al momento de cambiar estilos es más fácil.
- En nuestro caso usamos 2 componentes uno para la lógica y otro para crear el muñeco de ahorcado.

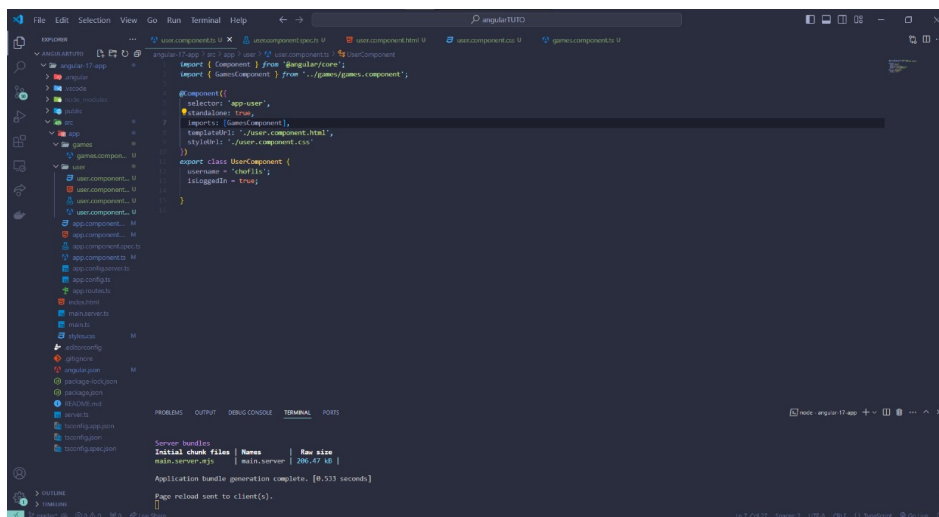


The screenshot shows a VS Code editor with a file explorer on the left displaying a project structure with various components like 'user.component.ts', 'game-board.component.ts', and 'hangman-figure.component.ts'. The main editor shows the code for 'user.component.ts' with the following content:

```
@Component({
  selector: 'app-user',
  standalone: true,
  imports: [],
  templateUrl: './user.component.html',
  styleUrls: ['./user.component.css']
})
export class UserComponent {
  username = 'churli';
  isLoggedIn = true;
}
```

The terminal at the bottom shows the output of the Angular CLI command 'ng generate component game-board', indicating that the component was successfully generated.

- Luego probaremos el servidor que crea angular, para que funcione un componente hijo en un componente padre este tiene que ser importado en el componente padre.



The screenshot shows a VS Code editor with a file explorer on the left displaying a project structure. The main editor shows the code for 'user.component.ts' with the following content:

```
@Component({
  selector: 'app-user',
  standalone: true,
  imports: [GameComponent],
  templateUrl: './user.component.html',
  styleUrls: ['./user.component.css']
})
export class UserComponent {
  username = 'churli';
  isLoggedIn = true;
}
```

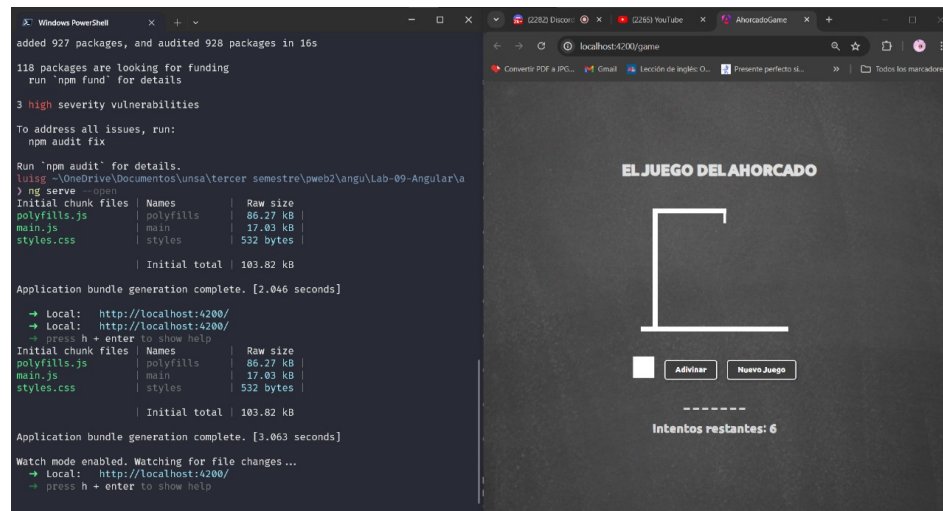
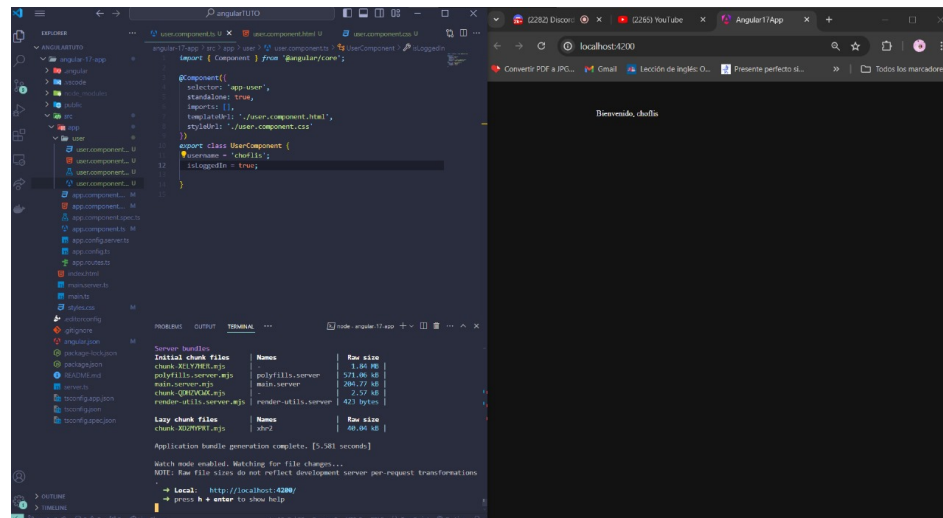
The terminal at the bottom shows the output of the Angular CLI command 'ng generate component game-board', indicating that the component was successfully generated.

- ahora el comando para correr el servidor.

Listing 4: Comandos del servidor

```
$ ng serve
$ ng serve --open
```

- Ahora podremos ver las modificaciones que hicimos



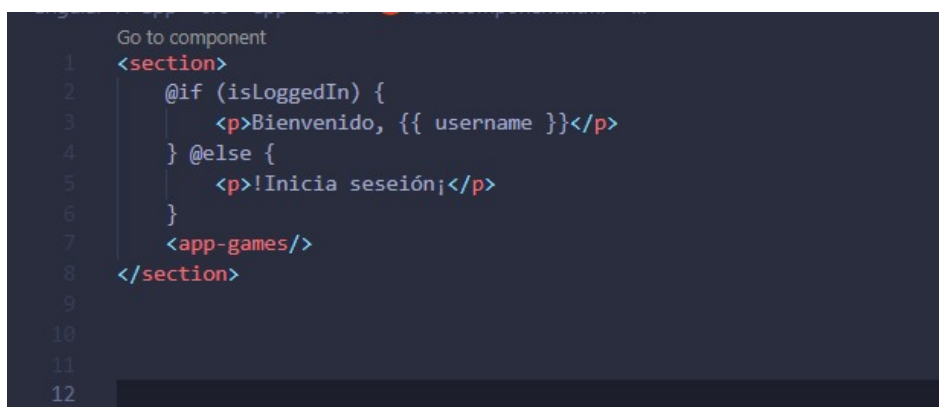
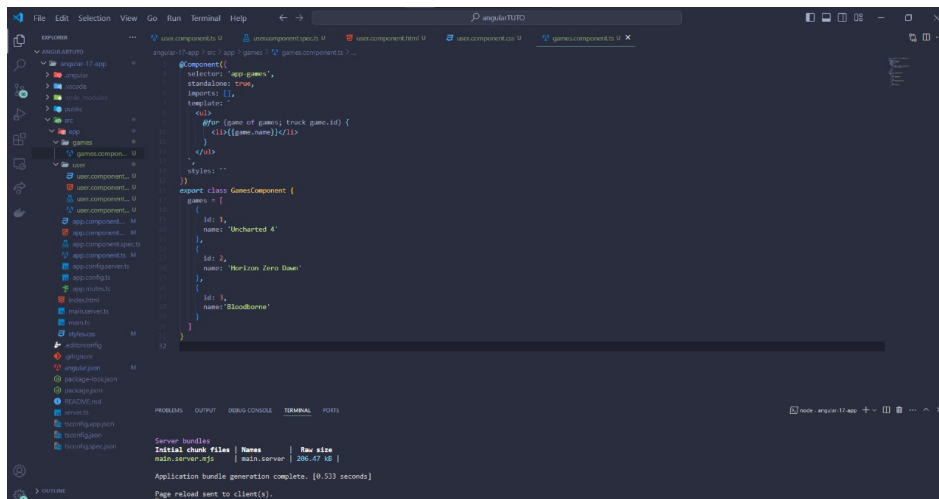
- Inicialmente nos sale una página que nos da la bienvenida, pero como hicimos modificaciones nos aparece las páginas creadas.
- Podemos usar los componentes que necesitamos, en caso no requieramos de uno, usamos la siguiente linea de comandos.

```
Creates a new, generic component definition in the given project.

Arguments:
  name The name of the component. [string]

Options:
  --help Shows a help message for this command in the console. [boolean] [default: true]
  -i, --interactive Enable interactive input prompts. [boolean] [default: false]
  -f, --dry-run Run through and reports activity without writing out results. [boolean] [default: false]
  --defaults Disable interactive input prompts for options with a default. [boolean] [default: false]
  --force Force overwriting of existing files. [boolean] [default: false]
  -t, --change-detection The change detection strategy to use in the new component. [string] [choices: "Default", "OnPush"] [default: "Default"]
  -b, --display-block Specifies if the style will contain ':host { display: block; }'. [boolean] [default: false]
  --export The declaring NgModule exports this component. [boolean] [default: false]
  --flat Create the new files at the top level of the current project. [boolean] [default: false]
  -s, --inline-style Include styles inline in the component.ts file. Only CSS styles can be included inline. By default, an external styles file is created and referenced in the component.ts file. [boolean] [default: false]
  -t, --inline-template Include template inline in the component.ts file. By default, an external template file is created and referenced in the component.ts file. [boolean] [default: false]
  -m, --module The declaring NgModule. [string] [default: false]
  -p, --prefix The prefix to apply to the generated component selector. [string] [default: false]
  --project The name of the project. [string] [default: false]
  --selector The HTML selector to use for this component. [string] [default: false]
  --skip-import Do not import this component into the outgoing NgModule. [boolean] [default: false]
  --skip-selector Specifies if the component should have a selector or not. [boolean] [default: false]
  --skip-tests Do not create "spec.ts" test files for the new component. [boolean] [default: false]
  --standalone Whether the generated component is standalone. [boolean] [default: true]
  --style The file extension or preprocessor to use for style files, or 'none' to skip generating the style file. [string] [choices: "css", "scss", "sass", "less", "none"] [default: "css"]
  --type Adds a developer-defined type to the filename, in the format "name.type.ts". [string] [default: "Component"]
  -v, --view-encapsulation The view encapsulation strategy to use in the new component. [string] [choices: "Emulated", "None", "ShadowDom"] [default: "Emulated"]
```

- Ahora para las funciones de for o de if en angular tiene una sintaxis interesante ya que funcionan de la siguiente manera "@for." "@if".



- Algo interesante de la ultima imagen es que para que un componente creado este en otro componente en el html tenemos que poner una referencia a ese componente en este caso la referencia es app-games.
- Por último podemos hacer que ocurran acciones como cuando damos un click o cosas asi, para esto necesitamos usar (click) dentro de nuestro código.

```
angular-17-app / src / app / user / user.component.html / ...
Go to component
1  <section>
2    @if (isLoggedIn) {
3      <p>Bienvenido, {{ username }}</p>
4      
7    } @else {
8      <p>!Inicia sesión;</p>
9    }
10   <app-games/>
11 </section>
12
13
14
15
```

4. Ahorcado-Game

5. Rúbricas

5.1. Entregable Informe

Tabla 1: Tipo de Informe

Informe	
Latex	El informe está en formato PDF desde Latex, con un formato limpio (buena presentación) y facil de leer.

6. Referencias

- Sobre nodejs
- <https://nodejs.dev/learn>
- <https://nodejs.org/en/docs/guides/getting-started-guide/>
- Sobre express
- <http://expressjs.com/es/>