Informe de Angular 1

Tema: Angular 1

Nota

Estudiante	Escuela	Asignatura
Victor Gonzalo Maldonado Vilca vmaldonadov@unsa.edu.pe	Escuela Profesional de Ingeniería de Sistemas	Programación Web 2 Semestre: III Código: 1702122

Tarea	Tema	Duración
09	Angular 1	2 horas

Semestre académico	Fecha de inicio	Fecha de entrega
2024 - A	Del 23 de mayo de 2024	Al 8 de julio de 2024

1. Introducción

Angular es un framework desarrollado por Google para crear aplicaciones web y móviles robustas y escalables. Se destaca por su arquitectura basada en componentes, enlace de datos bidireccional, y herramientas integradas para manejar formularios, enrutamiento y optimización de rendimiento. Es ampliamente utilizado por su capacidad de desarrollar aplicaciones de una sola página (SPAs) de manera eficiente y estructurada.

2. Objetivos

- Facilitar el desarrollo de aplicaciones de una sola página (SPA) con Angular.
- Mejorar la mantenibilidad y escalabilidad del código en Angular.
- Aumentar la productividad del desarrollador con Angular CLI y TypeScript.
- Implementar enlace de datos bidireccional eficiente en Angular.
- Ofrecer soporte robusto para formularios y validaciones en Angular.
- Optimizar el rendimiento de las aplicaciones web con Angular.

3. Tarea

- Volver a implementar las clases teóricas en un proyecto en github realizando commits de cada avance.
- Compartirlo con el profesor (usuario CarloCorrales010)

4. Entregables

- Informe hecho en Latex.
- URL: Repositorio GitHub.

5. Equipos, materiales y temas utilizados

- Angular
- Componentes
- Servicios
- Formularios
- Referencias

6. URL de Repositorio Github

- Link Repositorio GitHub.
- `https://github.com/Victor-Gonzalo-Maldonado-Vilca/Angular.git`

7. Desarrollo del trabajo

7.1. Capturas de la Actividad

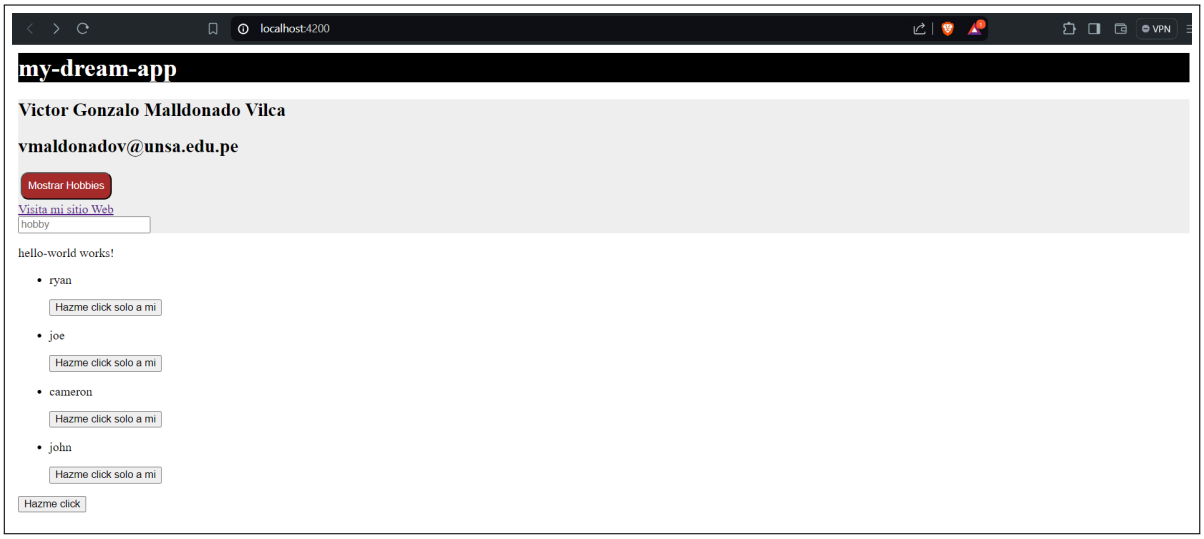
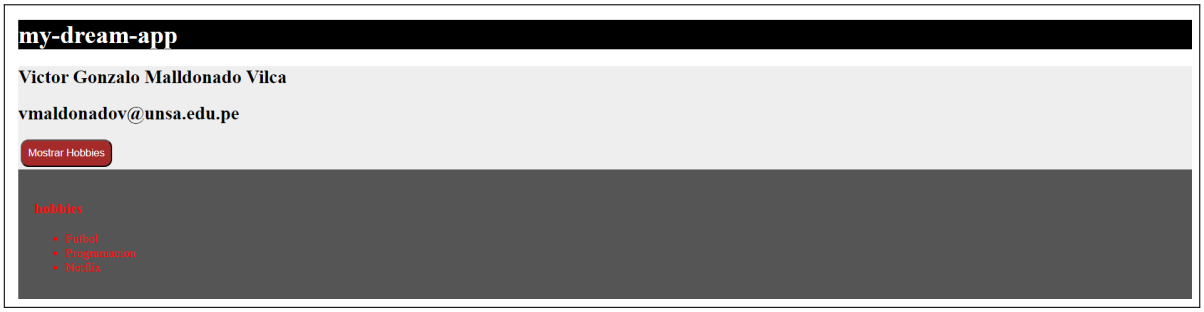


Figura 1: Ejecución 1

Figura 2: Ejecución 2



\caption{Presionando el botón "Mostrar Hobbies"}

7.2. Componente app

7.2.1. Archivo TypeScript

- **Class AppComponent:** La clase AppComponent contiene propiedades como title, name, email, webpage, hobbies, y showHobbies. En el constructor, se inicializan algunas de estas propiedades. showHobbies se inicializa como false para ocultar los hobbies al principio.

```
10  export class AppComponent {
11    title = 'my-dream-app';
12    name: string;
13    email; //string;
14    webpage: string;
15    hobbies: string[];
16    showHobbies: boolean;
17
18    constructor(){
19      console.log('Constructor Working...');
20      this.name = 'Victor Gonzalo Malldonado Vilca';
21      this.email = 'vmaldonadov@unsa.edu.pe';
22      this.webpage = 'http://www.unsa.edu.pe';
23      this.hobbies = ['Futbol', 'Programacion', 'Netflix'];
24      this.showHobbies = false;
25    }
```

- **Métodos y Propiedades Adicionales:**
  1.-toggleHobbies(): Este método cambia el estado de showHobbies entre true y false, alternando la visibilidad de los hobbies en la interfaz de usuario.
  2.-newHobby(hobby: any): Agrega un nuevo hobby al arreglo hobbies cuando se envía un formulario, luego limpia el campo de entrada y previene el comportamiento predeterminado de recargar la página.

```
34    toggleHobbies(){
35      this.showHobbies = !this.showHobbies;
36    }
37    newHobby(hobby: any) {
38      this.hobbies.push(hobby.value);
39      hobby.value = '';
40      return false;
41    }
```

- **Gestión de Usuarios:**
  1.-users: Arreglo que contiene nombres de usuarios.
  2.-sayHello(): Muestra un mensaje de alerta cuando se hace clic en el botón correspondiente.
  3.-deleteUser(user: any): Elimina un usuario específico del arreglo users.
  4.-addUser(newUser: any): Agrega un nuevo usuario al arreglo users y limpia el campo de entrada.

```
43    //User
44    users = ['ryan', 'joe', 'cameron', 'john'];
45    activated = false;
46
47    sayHello() {
48      alert("Hola desde app.component");
49    }
50
51    deleteUser(user: any) {
52      for(let i = 0; i < this.users.length; i++){
53        if(this.users[i] == user){
54          this.users.splice(i,1);
55        }
56      }
57    }
58
59    addUser(newUser: any){
60      //console.log(newUser.value);
61      this.users.push(newUser.value);
62      newUser.value = '';
63      newUser.focus();
64      return false;
65    }
```

7.2.2. Archivo html

- **Uso de Directivas y Propiedades en Angular:** router-outlet es una directiva de Angular que se utiliza para renderizar componentes dinámicamente basados en la ruta actual del enrutador.

```
1    <router-outlet></router-outlet>
```

- **Variables de Componente:** Define dos secciones (black y ash) con estilos CSS aplicados. title , name , y email son interpolaciones que muestran datos dinámicos desde el componente.

```
8    <div class="black">
9      <h1> {{ title }} </h1>
10   </div>
```

```
11    <div class="ash">
12      <h2>{{ name }}</h2>
13      <h2>{{ email }}</h2>
```

- **Control de Estado y Eventos:** toggleHobbies() se invoca al hacer clic en el botón, mostrando/ocultando la lista de hobbies basado en showHobbies. Los hobbies se muestran en una lista ul cuando showHobbies es verdadero.

```
14    <button (click)="toggleHobbies()" class="btn">Mostrar Hobbies</button>
15    <div *ngIf="showHobbies" class="componente2">
16      <h3>hobbies</h3>
17      <ul>
18        <li *ngFor="let hobby of hobbies">{{ hobby }}</li>
19      </ul>
20    </div>
```

- **Enlaces y Formularios:** Un enlace a un sitio web externo se abre en una nueva pestaña. El formulario permite agregar nuevos hobbies cuando se envía (submit).

```
21      <a href="{{ webpage }}" target="_blank">Visita mi sitio Web</a>
22      <form (submit)="newHobby(hobby)">
23        <input #hobby type="text" placeholder="hobby"/>
24      </form>
25    </div>
```

- **Componente Personalizado:** Se incluye el componente personalizado ¡app-hello-world¿, mostrando el contenido `"Aqui"` dentro de él.

```
27    <app-hello-world>Aqui</app-hello-world>
```

- **Uso de Componentes y Propiedades:** Se muestra una lista de usuarios utilizando el componente app-user, pasando [nameUser] como propiedad para cada usuario. Al hacer clic en el botón, se llama a sayHello().

```
29    <!--User-->
30    <div>
31      <ul>
32        <li *ngFor='let user of users'>
33          <app-user [nameUser]="user"></app-user>
34        </li>
35      </ul>
36      <button (click)="sayHello()">Hazme click</button>
37    </div>
```

- **Agregar y Borrar Elementos:** Permite agregar nuevos usuarios a la lista users mediante un formulario. Cada usuario tiene un botón para borrarlo, llamando a deleteUser(user) al hacer clic.

```
39    <!--Agregar Elementos-->
40    <h1>Desde app.component</h1>
41    <form (submit)="addUser(newUser)">
42      <input #newUser type="text"/>
43      <button>Agregar Usuario</button>
```

```
44    </form>
45    <!--Borrar Elementos-->
46    <ul>
47      <li *ngFor="let user of users">
48        {{ user }} <button (click)="deleteUser(user)">Borrar Usuario</button>
49      </li>
50    </ul>
```

7.2.3. Archivo css

- El estilo **.black** establece un fondo negro con texto blanco.

```
1   .black {
2     background-color: black;
3     color: white;
4   }
```

- El estilo **.ash** define un fondo gris claro con texto negro.

```
5   .ash {
6     background-color: #eee;
7     color: black;
8   }
```

- El estilo **.componente2** utiliza un fondo gris oscuro con texto rojo y tiene un relleno interior de 20px.

```
9   .componente2 {
10    background-color: #555;
11    color: red;
12    padding: 20px;
13  }
```

- El estilo **.btn** aplica un relleno interior de 8px, texto blanco sobre fondo marrón con bordes redondeados de 10px y se muestra como un bloque con un margen de 3px.

```
14  .btn {
15    padding: 8px;
16    color: white;
17    background-color: brown;
18    display: block;
19    margin: 3px;
20    border-radius: 10px;
21  }
```

7.3. Componente hello-world

7.3.1. Archivo TypeScript

- **Descripción:** La clase HelloWorldComponent en Angular está diseñada para funcionar como un componente básico dentro de una aplicación. Implementa la interfaz OnInit, lo que implica que tiene acceso al ciclo de vida del componente para ejecutar lógica especializada en su inicialización.

El constructor actualmente no tiene contenido, lo que significa que no realiza ninguna acción específica al crearse una instancia de este componente.

- **Código:**

```
8   export class HelloWorldComponent implements OnInit{
9
10    constructor(){ }
11
12    ngOnInit(): void {
13
14    }
15  }
```

7.3.2. Archivo html

- **Descripción:** El archivo solo contiene un parrafo.
- **Código:**

```
1   <p>hello-world works!</p>
```

7.4. Componente user

7.4.1. Archivo TypeScript

- **Descripción:** El UserComponent es un componente que permite mostrar información de usuarios. Está diseñado para recibir datos de entrada a través del decorador @Input(), especificamente nameUser, que puede contener cualquier tipo de dato. Durante la inicialización del componente, se ejecuta el método ngOnInit(), ideal para realizar tareas como inicialización de datos o llamadas a servicios. Además, incluye el método sayhello(nameUser: any), el cual despliega una alerta saludando al usuario cuyo nombre se le pasa como argumento.
- **Código:**

```
8   export class UserComponent implements OnInit{
9     @Input() nameUser: any;
10
11    ngOnInit() {}
12
13    sayhello(nameUser: any){
14      alert("Hola " + nameUser);
15    }
16  }
```

7.4.2. Archivo html

- **Descripción:** El código HTML muestra dinámicamente el nombre del usuario (nameUser) dentro de un párrafo (`<p>`). Un botón permite al usuario recibir un saludo personalizado al hacer clic en él, utilizando la función sayhello(nameUser) del componente.

- **Código:**

```
8     <p>{{ nameUser }}</p>
9     <button (click)=sayhello(nameUser)>Hazme click solo a mi</button>
```

8. Conclusiones

- Angular facilita la creación modular y reutilizable de aplicaciones web.
- Permite la manipulación dinámica de datos en la interfaz de usuario mediante enlaces bidireccionales y interpolación.
- Las directivas y eventos mejoran la interactividad al agregar comportamiento a elementos DOM.
- La inyección de dependencias simplifica la gestión de dependencias y promueve un diseño modular.
- Angular proporciona un ciclo de vida claro para los componentes, facilitando la gestión de estados y eventos.
- Ofrece herramientas para optimizar el rendimiento, como la detección de cambios y estrategias de renderizado eficientes.
- El entorno de desarrollo Angular CLI permite una creación eficiente y depuración de aplicaciones web complejas.

8.1. Rúbrica para el contenido del Informe y demostración

- El alumno debe marcar o dejar en blanco en celdas de la columna **Checklist** si cumplio con el ítem correspondiente.
- Si un alumno supera la fecha de entrega, su calificación será sobre la nota mínima aprobada, siempre y cuando cumpla con todos lo items.
- El alumno debe autocalificarse en la columna **Estudiante** de acuerdo a la siguiente tabla:

Tabla 1: Niveles de desempeño

	Nivel			
Puntos	Insatisfactorio 25 %	En Proceso 50 %	Satisfactorio 75 %	Sobresaliente 100 %
2.0	0.5	1.0	1.5	2.0
4.0	1.0	2.0	3.0	4.0

Tabla 2: Rúbrica para contenido del Informe y demostración

Contenido y demostración		Puntos	Checklist	Estudiante	Profesor
1. GitHub	Hay enlace URL activo del directorio para el laboratorio hacia su repositorio GitHub con código fuente terminado y fácil de revisar.	2	X	2	
2. Commits	Hay capturas de pantalla de los commits más importantes con sus explicaciones detalladas. (El profesor puede preguntar para refrendar calificación).	4	X	4	
3. Código fuente	Hay porciones de código fuente importantes con numeración y explicaciones detalladas de sus funciones.	2	X	2	
4. Ejecución	Se incluyen ejecuciones/pruebas del código fuente explicadas gradualmente.	2	X	2	
5. Pregunta	Se responde con completitud a la pregunta formulada en la tarea. (El profesor puede preguntar para refrendar calificación).	2	X	2	
6. Fechas	Las fechas de modificación del código fuente estan dentro de los plazos de fecha de entrega establecidos.	2	X	2	
7. Ortografía	El documento no muestra errores ortográficos.	2	X	2	
8. Madurez	El Informe muestra de manera general una evolución de la madurez del código fuente, explicaciones puntuales pero precisas y un acabado impecable. (El profesor puede preguntar para refrendar calificación).	4	X	4	
Total		20		20	

9. Referencias

- https://v17.angular.io/guide/architecture
- https://v17.angular.io/guide/binding-syntax