



INFORME DE LABORATORIO

INFORMACIÓN BÁSICA					
ASIGNATURA:	Programacion Web 2				
TÍTULO DE LA PRÁCTICA:	<i>Laboratorio 9 - Angular</i>				
NÚMERO DE PRÁCTICA:	9	AÑO LECTIVO:	2025	NRO. SEMESTRE:	1
FECHA DE PRESENTACIÓN	29/06/2025	Repositorio	https://github.com/JesusFSP/Curso-PWeb2.git		
INTEGRANTE (s): Silva Pino Jesus Francisco				NOTA:	
DOCENTE(s): CARLO JOSE LUIS CORRALES DELGADO					

SOLUCIÓN Y RESULTADOS
<p>I. SOLUCIÓN DE EJERCICIOS/PROBLEMAS</p> <h3>1. Configuración Inicial del Proyecto</h3> <p>Pasos realizados:</p> <ol style="list-style-type: none"> 1. Creación del proyecto Angular usando Angular CLI. 2. Configuración de la estructura de archivos principal. 3. Establecimiento de las dependencias básicas. <p>Funcionalidad implementada:</p> <ul style="list-style-type: none"> • Base estructural para una aplicación Angular moderna. • Configuración inicial de rutas y módulos. • Preparación del entorno de desarrollo.

	<p>UNIVERSIDAD NACIONAL DE SAN AGUSTIN FACULTAD DE INGENIERÍA DE PRODUCCIÓN Y SERVICIOS ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMA</p>	
<p>Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación</p>		
<p>Aprobación: 2025/05/03</p>	<p>Código: GUIA-PRLE-001</p>	<p>Página: 2</p>

2. Implementación del Componente Principal

Archivos modificados:

- app.ts (Lógica principal)
- app.html (Template principal)
- app.css (Estilos principales)



Funcionalidades implementadas:

// Componente principal que gestiona:
 // - Datos del usuario (nombre, email, página web)
 // - Lista de hobbies
 // - Lista de usuarios
 // - Funciones para manipular datos

```
export class App {
  title = 'my-dream-app';
  name: string;
  email: string;
  webpage: string;
  hobbies: string[];
  users: string[];

  constructor() {
    this.name = "Usuario Ejemplo";
    this.email = "usuario@example.com";
    this.webpage = "https://www.example.com";
    this.hobbies = ["Deportes", "Programación", "Música"];
    this.users = ['usuario1', 'usuario2', 'usuario3'];
  }

  // Métodos para gestión de datos
  newHobby(hobby: any) {
    this.hobbies.push(hobby.value);
```

	<p align="center">UNIVERSIDAD NACIONAL DE SAN AGUSTIN FACULTAD DE INGENIERÍA DE PRODUCCIÓN Y SERVICIOS ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMA</p>	
<p align="center">Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación</p>		
<p>Aprobación: 2025/05/03</p>	<p>Código: GUIA-PRLE-001</p>	<p align="right">Página: 3</p>

```

hobby.value = "";
}

deleteUser(user: string) {
  this.users = this.users.filter(u => u !== user);
}

addUser(newUser: any) {
  this.users.push(newUser.value);
  newUser.value = "";
  newUser.focus();
}
}

```

3. Creación del Componente HelloWorld

Archivos creados:

- hello-world.ts (Componente)
- hello-world.html (Template)
- hello-world.css (Estilos)

Funcionalidades implementadas:

```

// Componente secundario que:
// - Recibe datos del componente padre
// - Implementa interacción básica
export class HelloWorldComponent {
  @Input() nameUser: string = "";

  sayHello(nameUser: string) {
    alert(`Hola ${nameUser}`);
  }
}

```

	<p>UNIVERSIDAD NACIONAL DE SAN AGUSTIN FACULTAD DE INGENIERÍA DE PRODUCCIÓN Y SERVICIOS ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMA</p>	
<p>Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación</p>		
<p>Aprobación: 2025/05/03</p>	<p>Código: GUIA-PRLE-001</p>	<p>Página: 4</p>

4. Integración de Componentes

Pasos realizados:

1. Importación del componente secundario en el principal.
2. Configuración del sistema de templates.
3. Implementación de data binding entre componentes.

Implementación:

```
<!-- En app.html -->
<div *ngFor="let user of users">
  <app-hello-world [nameUser]="user"></app-hello-world>
</div>
```

5. Configuración de Rutas y Módulos

Archivos configurados:

- app.config.ts (Configuración principal)
- main.ts (Punto de entrada)



Estructura implementada:

```
// Configuración básica de la aplicación
export const appConfig: ApplicationConfig = {
  providers: [
    provideRouter([]) // Sistema de rutas
  ]
};
```

6. Implementación de Funcionalidades Adicionales

Características implementadas:

1. Sistema de gestión de hobbies:
 - Adición dinámica de nuevos hobbies.
 - Visualización condicional.
2. Sistema de gestión de usuarios:
 - Adición de nuevos usuarios.

	<p align="center">UNIVERSIDAD NACIONAL DE SAN AGUSTIN FACULTAD DE INGENIERÍA DE PRODUCCIÓN Y SERVICIOS ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMA</p>	
<p align="center">Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación</p>		
<p>Aprobación: 2025/05/03</p>	<p>Código: GUIA-PRLE-001</p>	<p align="right">Página: 5</p>

- Eliminación de usuarios existentes.
 - Data binding bidireccional.
3. Integración de componentes:
- Comunicación padre-hijo.
 - Paso de parámetros.
 - Manejo de eventos.

7. Estructura Final del Proyecto

src/app/

```

├─ app.html (Template principal)
├─ app.ts (Componente principal)
├─ app.css (Estilos globales)
├─ app.config.ts (Configuración)
├─ hello-world/
|   ├─ hello-world.html (Template)
|   ├─ hello-world.ts (Componente)
|   └─ hello-world.css (Estilos)
└─ ...

```

8. Funcionalidades Clave Implementadas

1. Componente Principal:

- Gestión centralizada del estado de la aplicación.
- Comunicación con componentes hijos.
- Manejo de formularios.

2. Componente HelloWorld:

- Recepción de datos desde componente padre.
- Emisión de eventos.
- Presentación de información.

3. Sistema de Templates:

- Directivas estructurales (*ngFor, *ngIf).
- Data binding (interpolación, property binding).

- Manejo de eventos.

my-dream-app

Jesus Silva

jsilva@unsa.edu.pe

hobbies:

- Fútbol
- Programación
- Netflix

[Visita mi sitio Web](#)**Desde app.component**

- Jesus
- joe
- cameron
- john

hello-world works! Jesus

II. CONCLUSIONES

1. Arquitectura y Organización del Código

El proyecto demostró la efectividad de la arquitectura basada en componentes de Angular, permitiendo una clara separación de responsabilidades. La estructura modular facilitó:

- **Mantenibilidad:** Cada componente con su lógica, template y estilos asociados
- **Escalabilidad:** Fácil adición de nuevas funcionalidades mediante componentes

	<p>UNIVERSIDAD NACIONAL DE SAN AGUSTIN FACULTAD DE INGENIERÍA DE PRODUCCIÓN Y SERVICIOS ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMA</p>	
<p>Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación</p>		
<p>Aprobación: 2025/05/03</p>	<p>Código: GUIA-PRLE-001</p>	<p>Página: 7</p>

adicionales

- **Reusabilidad:** Componentes como HelloWorld pueden ser utilizados en múltiples partes de la aplicación

2. Efectividad del Data Binding

La implementación demostró las ventajas del sistema de data binding de Angular:

- **Sincronización automática** entre modelo y vista
- **Reducción de código boilerplate** para actualizar la UI
- **Doble vía efectiva** en formularios y componentes interactivos

3. Gestión de Estado

El proyecto mostró diferentes enfoques para la gestión de estado:

- **Estado local** en componentes (hobbies, usuarios)
- **Comunicación entre componentes** mediante @Input()
- **Eventos personalizados** para interacción usuario-componente

4. Desarrollo de Componentes

La creación del componente HelloWorld evidenció:

- **Encapsulación efectiva** de funcionalidad específica
- **Comunicación clara** con el componente padre
- **Fácil integración** en la estructura general

5. Buenas Prácticas Implementadas

El proyecto incorporó múltiples prácticas recomendadas:

1. Separación de preocupaciones:

- Lógica en TypeScript
- Presentación en HTML
- Estilos en CSS dedicados

2. Uso de directivas estructurales:

	<p>UNIVERSIDAD NACIONAL DE SAN AGUSTIN FACULTAD DE INGENIERÍA DE PRODUCCIÓN Y SERVICIOS ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMA</p>	
<p>Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación</p>		
<p>Aprobación: 2025/05/03</p>	<p>Código: GUIA-PRLE-001</p>	<p>Página: 8</p>

- *ngFor para listas dinámicas
- *ngIf para renderizado condicional

6. Lecciones Aprendidas

1. **Configuración inicial** es crítica para el éxito del proyecto
2. **Planificación de componentes** ahorra tiempo en desarrollo
3. **Documentación del código** facilita el mantenimiento
4. **Pruebas tempranas** previenen problemas complejos

7. Valor Agregado del Proyecto

Esta implementación sirvió como:

- **Base demostrativa** de conceptos Angular fundamentales
- **Plantilla reusable** para futuros desarrollos
- **Ejemplo práctico** de arquitectura frontend moderna
- **Punto de partida** para funcionalidades más avanzadas

RETROALIMENTACIÓN GENERAL

REFERENCIAS Y BIBLIOGRAFÍA