

# Informe de Laboratorio 9

## Tema: ANGULAR

**Nota**

Estudiante	Escuela	Asignatura
Sergio Hanco Mullisaca shanccom@unsa.edu.pe	Escuela Profesional de Ingeniería de Sistemas	Programación Web 2 Semestre: II Código:

Laboratorio	Tema	Duración
9	ANGULAR	04 horas

Semestre académico	Fecha de inicio	Fecha de entrega
2024 - A	Del 14 Junio 2024	Al 22 Junio 2024

### 1. Tarea

- Informe de laboratorio
- Video en Flip
- Ejercicios Propuestos

### 2. Equipos, materiales y temas utilizados

- VS
- Git 2.39.2.
- Cuenta en GitHub con el correo institucional.

### 3. URL de Repositorio Github

- URL del video en yt.
- <https://youtu.be/7F3h6xbS8TU>
- URL del video en flip.
- <https://flip.com/s/33zq-2NuehBW>
- URL del GITHUB.
- [https://github.com/shanccom/Programacion\\_Web\\_2.git](https://github.com/shanccom/Programacion_Web_2.git)

### 4. Actividades

#### 4.1. IMPLEMENTACION - JUEGO AHORCADO

Listing 1: ahorcado.component.ts

```
import { Component, OnInit } from '@angular/core';
import { CommonModule } from '@angular/common';

@Component({
  selector: 'app-ahorcado',
  standalone: true,
  imports: [CommonModule],
  templateUrl: './ahorcado.component.html',
  styleUrls: ['./ahorcado.component.css']
})

export class AhorcadoComponent implements OnInit {
  palabraSecreta: string = '';
  palabraMostrada: string = '';
  intentosRestantes: number = 6;
  letrasIntentadas: string[] = [];
  mensaje: string = '';
  letraIngresada: string = '';

  palabras: string[] = ['ANGULAR', 'JAVASCRIPT', 'COMPONENTE', 'HTML', 'CSS',
    'PROGRAMACION', 'CELULAR', 'LAPTOP'];

  ngOnInit(): void {
    this.seleccionarPalabra();
    this.actualizarPalabraMostrada();
  }

  seleccionarPalabra(): void {
    this.palabraSecreta = this.palabras[Math.floor(Math.random() * this.palabras.length)];
  }

  actualizarPalabraMostrada(): void {
    this.palabraMostrada = '';
    for (let letra of this.palabraSecreta) {
      if (this.letrasIntentadas.includes(letra)) {
        this.palabraMostrada += letra + ' ';
      }
    }
  }
}
```

```

    } else {
        this.palabraMostrada += ' _ ';
    }
}
}

intentar(letra: string): void {
    if (!this.letrasIntentadas.includes(letra) && letra.match(/^[A-Z]$/)) {
        this.letrasIntentadas.push(letra);
        if (!this.palabraSecreta.includes(letra)) {
            this.intentosRestantes--;
        }
        this.actualizarPalabraMostrada();
        this.verificarEstadoJuego();
    }
}

verificarEstadoJuego(): void {
    if (this.intentosRestantes <= 0) {
        this.mensaje = 'Perdiste! La palabra era: ' + this.palabraSecreta;
    } else if (this.palabraSecreta === this.palabraMostrada.replace(/ /g, '')) {
        this.mensaje = 'Ganaste!';
    }
}

reiniciarJuego(): void {
    this.intentosRestantes = 6;
    this.letrasIntentadas = [];
    this.seleccionarPalabra();
    this.actualizarPalabraMostrada();
    this.mensaje = '';
}

get imagenAhorcado(): string {
    return 'assets/ahorcado${6 - this.intentosRestantes}.png';
}
}

```

## HTML DEL COMPONENTE

Listing 2: ahorcado.component.html

```

<div class="juego-container">
    <h1>JUEGO DEL AHORCADO</h1>
    <img [src]="imagenAhorcado" class="ahorcado-image">
    <p>{{ mensaje }}</p>
    <p>Palabra a adivinar: <span>{{ palabraMostrada }}</span></p>
    <p class="intentos-restantes">Intentos restantes: <span>{{ intentosRestantes }}</span></p>
    <p class="letras-intentadas">Letras intentadas: <span>{{ letrasIntentadas.join(', ') }}</span></p>
    <div *ngIf="mensaje !== 'Ganaste!' && mensaje !== 'Perdiste!'">
        <p>INTENTA CON UNA LETRA:</p>
        <input type="text" #letraInput
            (keyup.enter)="intentar(letraInput.value.toUpperCase())">
        <button (click)="intentar(letraInput.value.toUpperCase())">Intentar</button>
    </div>
</div>

```

```
</div>
<button (click)="reiniciarJuego()">Reiniciar Juego</button>
</div>
```

## CSS DEL COMPONENTE

Listing 3: ahorcado.component.css

```
body {
  text-align: center;
}

.juego-container {
  display: flex;
  flex-direction: column;
  justify-content: center;
  align-items: center;
  padding: 20px;
}

h1 {
  color: #2c3e50;
}

p {
  font-size: 1.2em;
}

input[type="text"] {
  font-size: 1.2em;
  padding: 5px;
  margin-right: 10px;
  border: 1px solid #ccc;
  border-radius: 4px;
}

button {
  font-size: 1.2em;
  padding: 5px 20px;
  color: #000000;
  background-color: #f5df90;
  border: none;
  border-radius: 0px;
  cursor: pointer;
  transition: background-color 0.3s ease;
  margin-top: 10px;
}

button:hover {
  background-color: #3498db;
}

button:disabled {
  background-color: #bdc3c7;
  cursor: not-allowed;
}
```

```
div {  
    margin-top: 20px;  
}  
  
.letras-intentadas {  
    margin-top: 10px;  
}  
  
.letras-intentadas span {  
    display: inline-block;  
    margin: 0 5px;  
    font-weight: bold;  
    color: #e74c3c;  
}  
  
.ahorcado-image {  
    width: 100px;  
    height: auto;  
    display: block;  
    margin: 20px auto;  
}
```

#### PRIMERA IMPLEMENTACION



The screenshot shows a web browser window with the address bar displaying 'localhost:4200'. The page title is 'Juego del Ahorcado'. The interface includes a text input field for the word to guess, followed by a series of dashes indicating the word's length. Below this, it shows 'Intentos restantes: 6'. There is a label 'Letras intentadas:' followed by a list of letters. At the bottom, there is a label 'Intenta con una letra:' followed by a text input field, an 'Intentar' button, and a 'Reiniciar Juego' button.

< > ↺ 📖 ⓘ localhost:4200

## Juego del Ahorcado

Palabra a adivinar: \_ \_ \_ \_ \_

Intentos restantes: 6

Letras intentadas:

Intenta con una letra:

## IMPLEMENTACION FINAL



### JUEGO DEL AHORCADO

1

Palabra a adivinar: \_ \_ \_ \_ \_

Intentos restantes: 6

Letras intentadas:

INTENTA CON UNA LETRA:

Intentar

Reiniciar Juego

## 5. Rúbricas

### 5.1. Entregable Informe

Tabla 1: Tipo de Informe

<b>Informe</b>	
<b>Latex</b>	El informe está en formato PDF desde Latex, con un formato limpio (buena presentación) y fácil de leer.

### 5.2. Rúbrica para el contenido del Informe y demostración

- El alumno debe marcar o dejar en blanco en celdas de la columna **Checklist** si cumple 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 los ítems.
- El alumno debe autocalificarse en la columna **Estudiante** de acuerdo a la siguiente tabla:

Tabla 2: Niveles de desempeño

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

Tabla 3: Rúbrica para contenido del Informe y demostración

Contenido y demostración		Puntos	Checklist	Estudiante	Profesor
<b>1. GitHub</b>	Hay enlace URL activo del directorio para el laboratorio hacia su repositorio GitHub con código fuente terminado y fácil de revisar.	2	X	2	
<b>2. Commits</b>	Hay capturas de pantalla de los commits más importantes con sus explicaciones detalladas. (El profesor puede preguntar para refrendar calificación).	4	X	4	
<b>3. Código fuente</b>	Hay porciones de código fuente importantes con numeración y explicaciones detalladas de sus funciones.	2	X	2	
<b>4. Ejecución</b>	Se incluyen ejecuciones/pruebas del código fuente explicadas gradualmente.	2	X	1	
<b>5. Pregunta</b>	Se responde con completitud a la pregunta formulada en la tarea. (El profesor puede preguntar para refrendar calificación).	2	X	2	
<b>6. Fechas</b>	Las fechas de modificación del código fuente están dentro de los plazos de fecha de entrega establecidos.	2	X	2	
<b>7. Ortografía</b>	El documento no muestra errores ortográficos.	2	X	2	
<b>8. Madurez</b>	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	
<b>Total</b>		20		19	