Informe de Laboratorio 09

Tema: Angular

Nota

Estudiante	Escuela	Asignatura
- David Alfredo Huamani Ollachica - Marco Antonio Suarez Huamaní - Rafael Diego Nina Calizaya - Angel Paul Apaza Nazareth	Escuela Profesional de Ingeniería de Sistemas	Programación Web 2 Semestre: III Código: 1702122

Laboratorio	Tema	Duración
09	Angular	04 horas

Semestre académico	Fecha de inicio	Fecha de entrega
2024 - A	Del 3 Julio 2024	Al 6 Julio 2024

1. Tarea

- URL GitHub del Projecto Angular `https://github.com/dev1d123/lab09`

```
* 47c604b (HEAD -> main, origin/main, origin/HEAD) Corrigiendo los fallos de
 logica al finalizar el juego
* 6436932 Implementacion de un listener para que funcione con el teclado
* 1b75f74 Agregando mas estilos para el teclado y un boton para regresar al
home
* 96cb4ca Agregando las imagenes que se usaran en los intentos del ahorcado,
 ademas de modificar y agregar cosas en el componente home
* 901ff41 actualizacion de estilos del juego
* fc7efef añadiendo  home y juego con sus html, css y ts
* 1cd2b74 Subiendo archivos iniciales
* 7e4d684 Initial commit
```

Figura 1: Imagen de los commits realizads en el repositorio

2. Codigo fuente

2.1. Componentes usados

- En la estructura del proyecto, hay dos componentes principales: Game y Home.
- Cada componente tiene su propio conjunto de archivos que siguen este esquema:
  - Archivo CSS (`.css`): Define los estilos específicos del componente.
  - Archivo HTML (`.html`): Define la estructura y el contenido del componente.
  - Archivo de pruebas (`.spec.ts`): Define las pruebas unitarias para el componente.
  - Archivo TypeScript (`.ts`): Define la lógica del componente, incluyendo sus propiedades y métodos.
- Esta organización permite mantener el código modular y separado, facilitando el mantenimiento y la escalabilidad del proyecto.

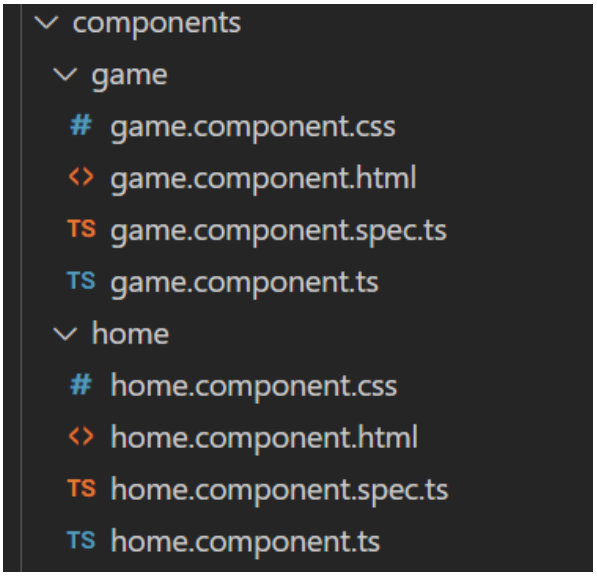

Figura 2: Diagrama de la base de datos

2.2. Componente Home

- El componente home se centra en la presentacion del juego, mostrando los nombres de los integrantes junto con las instrucciones del juego, ademas se añade un boton para pasar a la siguiente pestaña.
- **HTML**:
  - **Título Principal:** Un encabezado (`<h1>`) con el texto "Juego del ahorcado".
  - **Descripción:** Un párrafo (`<p>`) que invita al usuario a poner a prueba sus habilidades de adivinanza.
  - **Integrantes:**
    - Subtítulo (`<h2>`) con el texto Ïntegrantes".

- Lista (`<ul>`) que contiene los nombres de los miembros del equipo, cada uno en un elemento de lista (`<li>`) individual.

- **Botón de Navegación:** Un botón (`<button>`) con el texto .Empezar a Jugar"que, al ser presionado, redirige al usuario a otra pestaña mediante la directiva `routerLink` de Angular.
- **Instrucciones:**
  - Subtítulo (`<h2>`) con el texto Instrucciones".
  - Lista (`<ul>`) que proporciona una guía paso a paso sobre cómo jugar el juego del ahorcado. Cada paso está contenido en un elemento de lista (`<li>`).

Figura 3: Ejecucion de la pestaña home

Listing 1: src.component.html

```
<div class="container">
    <h1 class="my-4">Juego el ahorcado </h1>
    <p class="mt-4">Pon a prueba tus habilidades de adivinanza con este entretenido
        juego de palabras!</p>
    <h2 class="my-4">Integrantes</h2>
    <ul class="list-group">
      <li class="list-group-item">David Alfredo Huamani Ollachica</li>
      <li class="list-group-item">Marco Antonio Suarez Huaman</li>
      <li class="list-group-item">Rafael Diego Nina Calizaya</li>
      <li class="list-group-item">Angel Paul Apaza Nazareth</li>
    </ul>
    <button routerLink="/game" class="btn btn-primary">Empezar a Jugar</button>
      <div class="instructions">
        <h2>Instrucciones</h2>
        <ul>
          <li>Presiona "Empezar a Jugar" para iniciar una nueva partida.</li>
          <li>Selecciona letras para adivinar la palabra oculta.</li>
          <li>Si adivinas una letra correctamente, aparecer en su lugar en la
              palabra.</li>
          <li>Si adivinas incorrectamente, perders un intento y el dibujo del ahorcado
              se completar un poco ms.</li>
```

```
        <li>Ganas si adivinas la palabra completa antes de que se terminen los
            intentos.</li>
        <li>Pierdes si se completan todos los intentos sin adivinar la palabra.</li>
      </ul>
    </div>
</div>
```

2.3. Componente Game

El componente `GameComponent` es una implementación de un juego del ahorcado utilizando Angular. A continuación se describe en detalle el código de este componente, incluyendo su lógica y estilos.

2.3.1. Estructura del Componente

Listing 2: src.component.ts

```
   import { Component, OnInit, HostListener } from '@angular/core';
import { Router } from '@angular/router';

@Component({
  selector: 'app-game',
  templateUrl: './game.component.html',
  styleUrls: ['./game.component.css']
})
export class GameComponent implements OnInit {
  alphabet: string[] = 'QWERTYUIOPASDFGHJKLZXCVBNM'.split('');
  words: string[] = ['ANGULAR', 'COMPONENTE', 'SERVICIO', 'DIRECTIVA', 'MODULO'];
  hiddenWord: string[] = [];
  incorrectLetters: string[] = [];
  remainingAttempts: number = 6;
  wins: number = 0;
  losses: number = 0;
  hangmanImage: string = '';
  showModal: boolean = false;
  modalMessage: string = '';

  private selectedWord: string = '';
  constructor(private router: Router) {}

  ngOnInit(): void {
    this.resetGame();
  }

  guessLetter(letter: string): void {
    if(this.gameEnd()){
      return;
    }
    if (this.selectedWord.includes(letter)) {
      for (let i = 0; i < this.selectedWord.length; i++) {
        if (this.selectedWord[i] === letter) {
          this.hiddenWord[i] = letter;
        }
      }
```

```
  } else {
    this.incorrectLetters.push(letter);
    this.remainingAttempts--;
  }

  this.updateHangmanImage();
  this.checkGameStatus();
}

isLetterGuessed(letter: string): boolean {
  return (this.hiddenWord.includes(letter) || this.incorrectLetters.includes(letter));
}

resetGame(): void {
  this.selectedWord = this.words[Math.floor(Math.random() * this.words.length)];
  this.hiddenWord = Array(this.selectedWord.length).fill('_');
  this.incorrectLetters = [];
  this.remainingAttempts = 6;
  this.updateHangmanImage();
}

updateHangmanImage(): void {
  this.hangmanImage = `assets/img${6 - this.remainingAttempts}.png`;
}

checkGameStatus(): void {
  if (this.hiddenWord.join('') === this.selectedWord) {
    this.wins++;
    this.showModalMessage('Ganaste!');
  } else if (this.remainingAttempts === 0) {
    this.losses++;
    this.showModalMessage('Perdiste. La palabra era ' + this.selectedWord);
  }
}
gameEnd(): boolean {
  return this.remainingAttempts === 0;
}
showModalMessage(message: string): void {
  this.modalMessage = message;
  this.showModal = true;
  setTimeout(() => {
    this.showModal = false;
  }, 3000);
}
navigateHome(){
  this.router.navigate(['/']);
}

@HostListener('window:keydown', ['$event'])
handleKeyboardEvent(event: KeyboardEvent) {
  const letter = event.key.toUpperCase();
  if (!this.gameEnd() && this.alphabet.includes(letter) &&
      !this.hiddenWord.includes(letter)) {
    this.guessLetter(letter);
  }
}
```

```
}
```

2.3.2. Explicación del Código

- `alphabet`: Array que contiene las letras del alfabeto.
- `words`: Array que contiene las palabras que se usarán en el juego.
- `hiddenWord`: Array que representa la palabra oculta con guiones bajos.
- `incorrectLetters`: Array que contiene las letras incorrectas adivinadas por el jugador.
- `remainingAttempts`: Número de intentos restantes antes de perder el juego.
- `wins`: Contador de juegos ganados.
- `losses`: Contador de juegos perdidos.
- `hangmanImage`: Ruta de la imagen del ahorcado.
- `showModal`: Booleano que controla la visibilidad del modal de mensajes.
- `modalMessage`: Mensaje a mostrar en el modal.
- `selectedWord`: Palabra seleccionada para el juego actual.

2.3.3. Métodos

- `ngOnInit`: Inicializa el juego llamando a `resetGame`.
- `guessLetter`: Maneja la lógica de adivinar una letra.
- `isLetterGuessed`: Verifica si una letra ya ha sido adivinada.
- `resetGame`: Reinicia el juego con una nueva palabra.
- `updateHangmanImage`: Actualiza la imagen del ahorcado basada en los intentos restantes.
- `checkGameStatus`: Verifica el estado del juego (ganado o perdido).
- `gameEnd`: Verifica si el juego ha terminado.
- `showModalMessage`: Muestra un mensaje en un modal.
- `navigateHome`: Navega a la página de inicio.
- `handleKeyboardEvent`: Maneja eventos de teclado para adivinar letras.

2.3.4. HTML del Componente

Listing 3: src.component.html

```
<div class="container">
  <div class="game-header">
    <h1>Juego del Ahorcado</h1>
    <p>Encuentra la palabra:</p>
    <div class="word">
      <span *ngFor="let char of hiddenWord">{{ char }}</span>
    </div>
  </div>

  <button class="btn-exit" (click)="navigateHome()">
    Regresar al inicio
  </button>
```

```
  <div class="game-body">
    <img [src]="hangmanImage" alt="Hangman" class="hangman-image">
    <p>Intentos restantes: {{ remainingAttempts }}</p>
    <p>Letras incorrectas: {{ incorrectLetters.join(', ') }}</p>
    <div class="keycontainer">
      <div class="letters">
        <button class="letter btn" *ngFor="let letter of alphabet"
            (click)="guessLetter(letter)" [disabled]="isLetterGuessed(letter)">
           <div class="tecla caracter">
              {{ letter }}
           </div>
        </button>
      </div>
    </div>
  </div>

  <div class="game-footer">
    <p class="win-counter">Ganaste: {{ wins }}</p>
    <p class="lose-counter">Perdiste: {{ losses }}</p>
    <button (click)="resetGame()" class="btn btn-primary">Jugar de nuevo</button>
  </div>

  <div class="modal" [ngClass]="{'show': showModal}" role="dialog" aria-hidden="true">
    <div class="modal-dialog" role="document">
      <div class="modal-content">
        <div class="modal-header">
          <h5 class="modal-title">{{ modalMessage }}</h5>
        </div>
      </div>
    </div>
  </div>
</div>
```

2.3.5. CSS del Componente

Listing 4: src.component.css

```
@import url('https://fonts.googleapis.com/css2?family=Baskervville+SC&display=swap');

.container {
    text-align: center;
    margin-top: 20px;
    position: relative;
    font-family: 'Arial', sans-serif;
    background: linear-gradient(to right, #f80707 0%, #38fa0c 100%);
    padding: 20px;
    border-radius: 15px;
    box-shadow: 0 4px 10px rgba(0, 0, 0, 0.2);
    animation: fadeIn 1s ease-in-out;
  }

  @keyframes fadeIn {
    from {
      opacity: 0;
      transform: translateY(-20px);
```

```
  }
  to {
    opacity: 1;
    transform: translateY(0);
  }
}

.word {
  font-size: 32px;
  margin-bottom: 40px;
  letter-spacing: 10px;
  font-weight: bold;
  color: #4a4a4a;
  text-shadow: 1px 1px 2px rgba(0, 0, 0, 0.1);
  background: rgba(89, 249, 9, 0.8);
  padding: 10px 20px;
  border-radius: 10px;
  display: inline-block;
}

.hangman-image {
  width: 200px;
  height: auto;
  margin-bottom: 20px;
  transition: transform 0.3s ease, box-shadow 0.3s ease;
  border-radius: 10px;
  box-shadow: 0 4px 10px rgba(0, 0, 0, 0.2);
}

.hangman-image:hover {
  transform: scale(1.1);
  box-shadow: 0 6px 15px rgba(0, 0, 0, 0.3);
}
.keycontainer{
  display: flex;
  align-items: center;
  justify-content: center;
  max-width: 700px;
  margin: 0 auto;
}
.letters{
  display: flex;
  gap: 20px;
  flex-wrap: wrap;
  margin: auto;
  max-width: 100%;
  justify-content: center;

}
.letters button {
  font-family: "Baskervville SC", serif;
  font-weight: 400;
  font-style: normal;
  font-size: 1rem;
  background-color: beige;
  color: #4A4A4A;
```

```
  border: 1px black solid;
  box-shadow: 5px 5px #4A4A4A;
  border-radius: 1rem;
  min-width: 50px;
  max-width: 50px;
  min-height: 50px;
  max-height: 50px;
  cursor: pointer;
  transition: transform 0.3s ease, color 0.3s ease, background-color 0.3s ease;
}

.letters button:hover {
  background-color: #4A4A4A;
  color: beige;
  transform: scale(1.2);
}

.letters button:disabled {
  background-color: #ccc;
  color: #999;
  cursor: not-allowed;
  transform: scale(1);
  box-shadow: none;
}

.game-footer {
  margin-top: 20px;
  display: flex;
  justify-content: center;
  gap: 20px;
}

.win-counter {
  position: absolute;
  top: 10px;
  left: 10px;
  color: green;
  font-weight: bold;
  font-size: 18px;
  background: rgba(255, 255, 255, 0.8);
  padding: 5px 10px;
  border-radius: 5px;
}

.lose-counter {
  position: absolute;
  top: 10px;
  right: 10px;
  color: red;
  font-weight: bold;
  font-size: 18px;
  background: rgba(255, 255, 255, 0.8);
  padding: 5px 10px;
  border-radius: 5px;
}
```

```css
.modal {
  display: none;
  position: fixed;
  top: 0;
  left: 0;
  width: 100%;
  height: 100%;
  background-color: rgba(0, 0, 0, 0.5);
  align-items: center;
  justify-content: center;
  z-index: 9999;
  animation: fadeInModal 0.5s ease forwards;
}

.modal.show {
  display: flex;
}

@keyframes fadeInModal {
  from {
    opacity: 0;
  }
  to {
    opacity: 1;
  }
}

.modal-dialog {
  background-color: white;
  padding: 20px;
  border-radius: 10px;
  box-shadow: 0 4px 8px rgba(0, 0, 0, 0.2);
  animation: slideInModal 0.5s ease forwards, swing 2s infinite;
}

@keyframes slideInModal {
  from {
    transform: translateY(-50%);
    opacity: 0;
  }
  to {
    transform: translateY(0);
    opacity: 1;
  }
}

@keyframes swing {
  20% {
    transform: rotate(15deg);
  }
  40% {
    transform: rotate(-10deg);
  }
  60% {
    transform: rotate(5deg);
  }
```

```
  80% {
    transform: rotate(-5deg);
  }
  100% {
    transform: rotate(0deg);
  }
}

.btn {
  padding: 10px 20px;
  font-size: 16px;
  border: none;
  border-radius: 5px;
  background-color: #007bff;
  color: #fff;
  cursor: pointer;
  transition: background-color 0.3s ease, transform 0.3s ease, box-shadow 0.3s ease;
  box-shadow: 0 2px 5px rgba(0, 0, 0, 0.1);

}

.btn:hover {
  background-color: #0056b3;
  transform: scale(1.05);
  box-shadow: 0 4px 10px rgba(0, 0, 0, 0.2);
}

@media (max-width: 600px) {
  .letters button {
    font-size: 16px;
    padding: 8px 12px;
  }

  .word {
    font-size: 24px;
  }

  .btn {
    font-size: 14px;
    padding: 8px 16px;
  }
}

.btn-exit {
  position: fixed;
  top: var(--button-top-position);
  left: 5px;
  background-color: #ff4c4c;
  color: white;
  border: none;
  border-radius: 5px;
  cursor: pointer;
  box-shadow: 0 4px 6px rgba(0, 0, 0, 0.1);
  font-size: 16px;
  transition: all 0.3s ease-in-out;
}
```

```
.btn-exit:hover {
    background-color: #ff1a1a;
    box-shadow: 0 6px 8px rgba(0, 0, 0, 0.2);
    transform: scale(1.1);
}

.btn-exit:active {
    background-color: #e60000;
    box-shadow: 0 2px 4px rgba(0, 0, 0, 0.2);
    transform: scale(0.9);
}

@keyframes fadeIn {
    0% {
        opacity: 0;
        transform: translateY(-20px);
    }
    100% {
        opacity: 1;
        transform: translateY(0);
    }
}

.btn-exit {
    animation: fadeIn 0.8s ease-out;
}

@keyframes pulse {
    0% {
        box-shadow: 0 0 0 0 rgba(255, 76, 76, 0.7);
    }
    70% {
        box-shadow: 0 0 0 20px rgba(255, 76, 76, 0);
    }
    100% {
        box-shadow: 0 0 0 0 rgba(255, 76, 76, 0);
    }
}

.btn-exit {
    animation: pulse 1.5s infinite;
}
```

2.3.6. Explicación de los Estilos

- `.container`: Estilos para el contenedor principal, incluyendo fuente, colores de fondo y animación de entrada.
- `.word`: Estilos para la palabra oculta, con espaciado entre letras y efectos de sombra.
- `.hangman-image`: Estilos para la imagen del ahorcado, incluyendo tamaño, transición y efectos de hover.
- `.keycontainer`: Estilos para el contenedor de teclas.
- `.letters`: Estilos para las letras del teclado virtual, con efectos de hover y deshabilitado.

- `.game-footer`: Estilos para el pie de página del juego.
- `.win-counter` y `.lose-counter`: Estilos para los contadores de victorias y derrotas.
- `.modal`: Estilos para el modal, incluyendo animaciones de entrada.
- `.btn`: Estilos para los botones, con efectos de hover y transiciones.
- `.btn-exit`: Estilos para el botón de salida, con animaciones de entrada y efectos de pulsación.

Este componente implementa todas las funcionalidades necesarias para jugar al ahorcado, manejando eventos de teclado, actualizando la interfaz de usuario y mostrando mensajes de victoria o derrota de manera efectiva.

Figura 4: Ejecución del juego 1

3. Rúbricas

3.1. Sobre el 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.

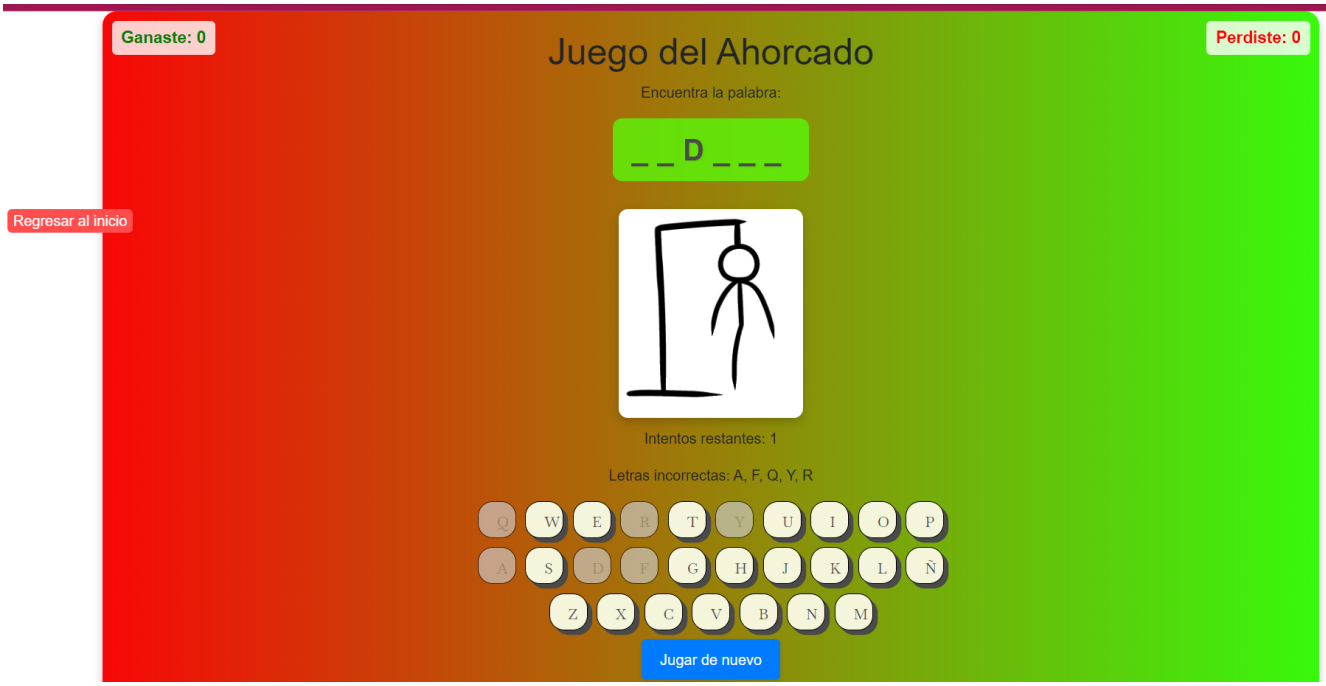

Figura 5: Ejecución del juego 2

Figura 6: Ejecución del juego 3

Figura 7: Ejecución del juego 4

Figura 8: Ejecución del juego 5

3.2. 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 2: 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 3: 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	3	
Total		20		19	

4. Referencias

- `https://angular.dev/`
- `https://docs.angular.lat/`