Registro Diario de Avances

**Fecha:** 26/09/2025

**Autor:**

**Danluis Enrique Romani Mora**

**Eduardo Chaparro Huaman**

**Fabrizcio Argandoña Montalvo**

**Versión:** 1.0

# 1. Introducción

En este documento se muestra el avance de tareas respectivas al Sprint 2 con lo que también se detallan las historias de usuarios pertenecientes y el esfuerzo realizado para completar cada una de ellas, con evidencias del código utilizado, burndown chart y el análisis respectivo al rendimiento del equipo.

# 2. Lineamientos de Diseño

Desarrollar una primera versión funcional de la aplicación que permita a los usuarios:

* Validación de Credenciales
* Módulo de recuperación de contraseñas
* Acceso del docente al sistema.

# 3. Historias de Usuario y Tareas

| Backlog ID | Historia de Usuario | Tiempo Estimado | Fecha de Inicio | Fecha de Finalización |
| --- | --- | --- | --- | --- |
| HU-1.1 | Acceso de docentes al sistema | 9 | 24/09/2025 | 28/09/2025 |
| HU-1.2 | Validación de las credenciales del docente | 10 | 26/09/2025 | 29/09/2025 |
| HU-1.3 | Recuperación de contraseña para docentes | 11 | 28/09/2025 | 30/09/2025 |

# 4. Evidencias de Código

## 4.1 Historia de Usuario 1.1: Acceso de docentes al sistema.

Archivo: backend/a

Diseñar pantalla de login con campos de usuario y contraseña.

<body>

<div class="overlay"></div>

<header>

<h1>INICIO DE SESIÓN</h1>

<br>

<h2>Ingrese sus datos</h2>

<div class="logoCole"></div>

</header>

<div class="login-container" [ngClass]="{'fade-mode': animar}">

<div style="display: flex; justify-content: space-between; gap: 10px;">

<button class="crear-button"

[ngClass]="{'active': tipoLogin === 'docente'}"

(click)="setLogin('docente')">Docentes</button>

<button class="crear-button"

[ngClass]="{'active': tipoLogin === 'alumno'}"

(click)="setLogin('alumno')">Alumnos</button>

<button class="crear-button"

[ngClass]="{'active': tipoLogin === 'admin'}"

(click)="setLogin('admin')">Administrador</button>

</div>

<input type="text" class="input-field" placeholder="Usuario" [(ngModel)]="usuario">

<input type="password" class="input-field" placeholder="Contraseña" [(ngModel)]="contrasena">

<button class="login-button" (click)="login()">Ingresar</button>

</div>

</body>

## 4.2 Historia de Usuario 1.2: Validación de las credenciales del docente.

package com.prsanmartin.appmartin.entity;

import jakarta.persistence.\*;

import lombok.AllArgsConstructor;

import lombok.Data;

import lombok.NoArgsConstructor;

@Entity

@Table(name = "Docentes")

@Data

@NoArgsConstructor

@AllArgsConstructor

public class Docente {

@Id

@GeneratedValue(strategy = GenerationType.*IDENTITY*)

@Column(name = "IdDocente")

private Integer idDocente;

@OneToOne(fetch = FetchType.*LAZY*)

@JoinColumn(name = "IdUsuario", nullable = false)

private Usuario usuario;

@Column(name = "Especialidad", nullable = false, length = 100)

private String especialidad;

}

package com.prsanmartin.appmartin.entity;

import jakarta.persistence.\*;

import lombok.AllArgsConstructor;

import lombok.Data;

import lombok.NoArgsConstructor;

import java.time.LocalDateTime;

@Entity

@Table(name = "Usuarios")

@Data

@NoArgsConstructor

@AllArgsConstructor

public class Usuario {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

@Column(name = "IdUsuario")

private Integer idUsuario;

@Column(name = "NombreUsuario", nullable = false, unique = true, length = 80)

private String nombreUsuario;

@Column(name = "CorreoElectronico", nullable = false, unique = true, length = 150)

private String correoElectronico;

@Column(name = "ContrasenaHash", nullable = false, length = 255)

private String contrasenaHash;

@ManyToOne(fetch = FetchType.LAZY)

@JoinColumn(name = "IdRol", nullable = false)

private Rol rol;

@Column(name = "FechaCreacion")

private LocalDateTime fechaCreacion;

private Boolean activo;

public Boolean getActivo() {

return activo;

}

public void setActivo(Boolean activo) {

this.activo = activo;

}

}

## 4.3 Historia de Usuario 1.3: Recuperación de contraseña para docentes.

Archivo Frontend . recuperar-contrasena.component.html

<div class="overlay"></div>

<div class="panel-container">

<div class="panel-header">

<h2>Recuperar Contraseña</h2>

<button class="actividad-button cancelar" (click)="regresar()">Cancelar</button>

</div>

<form (ngSubmit)="confirmar()" #formContrasena="ngForm" class="formulario">

*<!-- Nueva contraseña -->*

<div class="form-group">

<label for="nueva">Nueva Contraseña</label>

<input type="password" id="nueva" name="nueva" [(ngModel)]="nuevaContrasena"

required minlength="6" pattern="^(?=.\*[0-9]).{6,}$" #nuevaCtrl="ngModel" />

<div class="error" \*ngIf="nuevaCtrl.invalid && (nuevaCtrl.dirty || nuevaCtrl.touched)">

<span \*ngIf="nuevaCtrl.errors?.['required']">La contraseña es obligatoria.</span>

<span \*ngIf="nuevaCtrl.errors?.['minlength']">Debe tener al menos 6 caracteres.</span>

<span \*ngIf="nuevaCtrl.errors?.['pattern']">Debe incluir al menos un número.</span>

</div>

</div>

*<!-- Confirmar contraseña -->*

<div class="form-group">

<label for="confirmar">Confirmar Contraseña</label>

<input type="password" id="confirmar" name="confirmar" [(ngModel)]="confirmarContrasena"

required #confirmarCtrl="ngModel" />

<div class="error" \*ngIf="confirmarCtrl.invalid && (confirmarCtrl.dirty || confirmarCtrl.touched)">

<span \*ngIf="confirmarCtrl.errors?.['required']">Debe confirmar la contraseña.</span>

</div>

<div class="error" \*ngIf="confirmarContrasena && confirmarContrasena !== nuevaContrasena">

Las contraseñas no coinciden.

</div>

</div>

*<!-- Botón -->*

<button class="actividad-button" type="submit"

[disabled]="formContrasena.invalid || nuevaContrasena !== confirmarContrasena">

Confirmar

</button>

</form>

</div>

Archivo Frontend . recuperar-contrasena.component.ts

import { Component } from '@angular/core';

import { Router } from '@angular/router';

import { FormsModule } from '@angular/forms';

import { CommonModule } from '@angular/common';

import Swal from 'sweetalert2';

@Component({

selector: 'app-recuperar-contrasena',

templateUrl: './recuperar-contrasena.component.html',

styleUrls: ['./recuperar-contrasena.component.css'],

standalone: true,

imports: [FormsModule,CommonModule]

})

export class RecuperarContrasenaComponent {

nuevaContrasena: string = '';

confirmarContrasena: string = '';

constructor(*private* router: Router) { }

regresar() {

this.router.navigate(['/login']);

}

confirmar() {

*// Validar campos vacíos*

if (!this.nuevaContrasena || !this.confirmarContrasena) {

Swal.fire({

icon: 'error',

title: 'Campos vacíos',

text: 'Debe completar ambos campos.'

});

*return*;

}

*// Validar longitud mínima*

if (this.nuevaContrasena.length < 6) {

Swal.fire({

icon: 'error',

title: 'Contraseña inválida',

text: 'La contraseña debe tener al menos 6 caracteres.'

});

*return*;

}

*// Validar que tenga al menos un número*

*const* regexNumero = /\d/;

if (!regexNumero.test(this.nuevaContrasena)) {

Swal.fire({

icon: 'error',

title: 'Contraseña insegura',

text: 'La contraseña debe contener al menos un número.'

});

*return*;

}

*// Validar que coincidan*

if (this.nuevaContrasena !== this.confirmarContrasena) {

Swal.fire({

icon: 'error',

title: 'Error',

text: 'Las contraseñas no coinciden.'

});

*return*;

}

*// Si todo es válido*

Swal.fire({

icon: 'success',

title: 'Éxito',

text: 'Contraseña cambiada con éxito.'

}).then(() => {

this.router.navigate(['/login']);

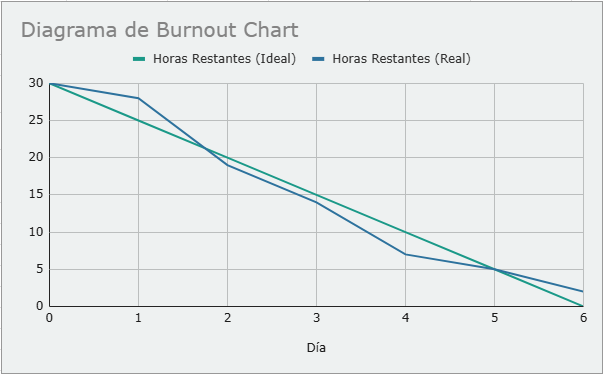
});

}

}

# 5. Burndown Chart

| Backlog ID | Historia de Usuario | Tiempo Estimado | 24/04/2025 | 25/04/2025 | 26/04/2025 | 28/04/2025 | 29/04/2025 | 30/04/2025 |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Dia1** | **Dia2** | **Dia3** | **Dia4** | **Dia5** | **Dia6** |
| HU-0.1 | Acceso de docentes al sistema | 9 | 2 | 3 | 2 | 1 |  |  |
| HU-0.2 | Validación de las credenciales del docente | 10 |  |  | 3 | 4 | 1 |  |
| HU-0.3 | Recuperación de contraseña para docentes | 11 |  |  |  | 2 | 1 | 3 |
| Tiempo de trabajo | | 30 | 28 | 19 | 14 | 7 | 5 | 2 |



# 6. Análisis del Sprint

* Se cumplieron con todas las tareas estimadas, pero tuvimos problemas con la gestión del tiempo..
* El progreso fue constante pero tuvimos problemas en la gestión de tiempos, por lo cual algunas tareas se retrasaron .
* Se logró modularizar el código y documentarlo en comentarios.
* Las estimaciones no fueron del todo precisas.

# 7. Conclusiones

El Sprint 2 concluyó exitosamente con todas las funcionalidades básicas implementadas. Se generaron componentes reutilizables, se probaron los flujos de usuario, y el código quedó preparado para futuras funcionalidades como edición y múltiples listas.

El equipo demostró capacidad de trabajo constante pero se vieron problemas en el momento de terminar el sprint en el tiempo acordado. Las métricas indican que la carga pudo estar mejor distribuida y los objetivos del sprint se pueden alcanzar con una mejor gestión..