Registro Diario de Avances

**Fecha:** 24/09/2025

**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 1 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:

* Iniciar sesión.
* Iniciar con roles diferenciados.
* implementación de la base de datos dentro del sistema.

# 3. Historias de Usuario y Tareas

| Backlog ID | Historia de Usuario | Tiempo Estimado | Fecha de Inicio | Fecha de Finalización |
| --- | --- | --- | --- | --- |
| HU-0.1 | Diseño de la base de datos. | 2 | 18/09/2025 | 18/09/2025 |
| HU-0.2 | Creación de la base de datos. | 1 | 18/09/2025 | 18/09/2025 |
| HU-0.3 | Normalización de la base de datos. | 3 | 18/09/2025 | 18/09/2025 |
| HU-1.7 | Acceso de administrador al sistema | 5 | 19/09/2025 | 20/09/2025 |
| HU-1.8 | Registro de fecha y hora de acceso del administrador | 6 | 20/09/2025 | 22/09/2025 |
| HU-1.9 | Restricción de acceso a perfiles de administrador | 5 | 20/09/2025 | 22/09/2025 |
| HU-1.10 | Cierre de sesión manual | 3 | 23/09/2025 | 23/09/2025 |
| HU-1.11 | Cierre de sesión por inactividad | 3 | 24/09/2025 | 24/09/2025 |

# 4. Evidencias de Código

## 4.1 Historia de Usuario 0.1: Diseño de la base de datos.

Archivo: Proyecto/backend/DB/prmartin (1).sql

SET SQL\_MODE = "NO\_AUTO\_VALUE\_ON\_ZERO";

START TRANSACTION;

SET time\_zone = "+00:00";

/\*!40101 SET @OLD\_CHARACTER\_SET\_CLIENT=@@CHARACTER\_SET\_CLIENT \*/;

/\*!40101 SET @OLD\_CHARACTER\_SET\_RESULTS=@@CHARACTER\_SET\_RESULTS \*/;

/\*!40101 SET @OLD\_COLLATION\_CONNECTION=@@COLLATION\_CONNECTION \*/;

/\*!40101 SET NAMES utf8mb4 \*/;

--

-- Base de datos: `prmartin`

--

-- --------------------------------------------------------

--

-- Estructura de tabla para la tabla `alumnos`

--

CREATE TABLE `alumnos` (

`IdAlumno` int(11) NOT NULL,

`IdUsuario` int(11) NOT NULL,

`AnioIngreso` int(11) NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_general\_ci;

-- --------------------------------------------------------

--

-- Estructura de tabla para la tabla `auditoria`

--

CREATE TABLE `auditoria` (

`IdAuditoria` int(11) NOT NULL,

`IdUsuario` int(11) DEFAULT NULL,

`Accion` varchar(150) NOT NULL,

`Entidad` varchar(50) NOT NULL,

`IdEntidad` int(11) DEFAULT NULL,

`Detalles` text DEFAULT NULL,

`DireccionIP` varchar(50) DEFAULT NULL,

`FechaEvento` datetime DEFAULT current\_timestamp()

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_general\_ci;

-- --------------------------------------------------------

--

-- Estructura de tabla para la tabla `calificaciones`

--

CREATE TABLE `calificaciones` (

`IdCalificacion` int(11) NOT NULL,

`IdMatricula` int(11) NOT NULL,

`Periodo` varchar(20) NOT NULL,

`Nota` decimal(5,2) NOT NULL CHECK (`Nota` >= 0 and `Nota` <= 20),

`FechaRegistro` datetime DEFAULT current\_timestamp()

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_general\_ci;

-- --------------------------------------------------------

--

-- Estructura de tabla para la tabla `cursos`

--

CREATE TABLE `cursos` (

`IdCurso` int(11) NOT NULL,

`NombreCurso` varchar(150) NOT NULL,

`IdDocente` int(11) NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_general\_ci;

-- --------------------------------------------------------

--

-- Estructura de tabla para la tabla `docentes`

--

CREATE TABLE `docentes` (

`IdDocente` int(11) NOT NULL,

`IdUsuario` int(11) NOT NULL,

`Especialidad` varchar(100) NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_general\_ci;

-- --------------------------------------------------------

--

-- Estructura de tabla para la tabla `flyway\_schema\_history`

--

CREATE TABLE `flyway\_schema\_history` (

`installed\_rank` int(11) NOT NULL,

`version` varchar(50) DEFAULT NULL,

`description` varchar(200) NOT NULL,

`type` varchar(20) NOT NULL,

`script` varchar(1000) NOT NULL,

`checksum` int(11) DEFAULT NULL,

`installed\_by` varchar(100) NOT NULL,

`installed\_on` timestamp NOT NULL DEFAULT current\_timestamp(),

`execution\_time` int(11) NOT NULL,

`success` tinyint(1) NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_general\_ci;

--

-- Volcado de datos para la tabla `flyway\_schema\_history`

--

INSERT INTO `flyway\_schema\_history` (`installed\_rank`, `version`, `description`, `type`, `script`, `checksum`, `installed\_by`, `installed\_on`, `execution\_time`, `success`) VALUES

(1, '1', '<< Flyway Baseline >>', 'BASELINE', '<< Flyway Baseline >>', NULL, 'root', '2025-09-21 23:06:46', 0, 1);

-- --------------------------------------------------------

--

-- Estructura de tabla para la tabla `matriculas`

--

CREATE TABLE `matriculas` (

`IdMatricula` int(11) NOT NULL,

`IdAlumno` int(11) NOT NULL,

`IdCurso` int(11) NOT NULL,

`FechaMatricula` date DEFAULT current\_timestamp()

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_general\_ci;

-- --------------------------------------------------------

--

-- Estructura de tabla para la tabla `refreshtokens`

--

CREATE TABLE `refreshtokens` (

`IdRefreshToken` int(11) NOT NULL,

`IdUsuario` int(11) NOT NULL,

`Token` varchar(255) NOT NULL,

`IssuedAt` datetime DEFAULT NULL,

`ExpiresAt` datetime NOT NULL,

`Revoked` tinyint(1) DEFAULT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_general\_ci;

-- --------------------------------------------------------

--

-- Estructura de tabla para la tabla `respuestasia`

--

CREATE TABLE `respuestasia` (

`IdRespuesta` int(11) NOT NULL,

`IdSolicitud` int(11) NOT NULL,

`RespuestaJSON` text NOT NULL,

`ModeloIA` varchar(100) DEFAULT NULL,

`Confianza` decimal(5,2) DEFAULT NULL,

`FechaRespuesta` datetime DEFAULT current\_timestamp()

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_general\_ci;

-- --------------------------------------------------------

--

-- Estructura de tabla para la tabla `roles`

--

CREATE TABLE `roles` (

`IdRol` int(11) NOT NULL,

`NombreRol` varchar(20) NOT NULL,

`nombre\_rol` varchar(20) NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_general\_ci;

--

-- Volcado de datos para la tabla `roles`

--

INSERT INTO `roles` (`IdRol`, `NombreRol`, `nombre\_rol`) VALUES

(1, 'ADMIN', ''),

(2, 'DOCENTE', ''),

(3, 'ALUMNO', '');

-- --------------------------------------------------------

--

-- Estructura de tabla para la tabla `solicitudesia`

--

CREATE TABLE `solicitudesia` (

`IdSolicitud` int(11) NOT NULL,

`IdDocente` int(11) NOT NULL,

`IdCurso` int(11) NOT NULL,

`IdAlumno` int(11) DEFAULT NULL,

`DatosEntrada` text NOT NULL,

`Estado` varchar(20) DEFAULT 'PENDIENTE',

`FechaSolicitud` datetime DEFAULT current\_timestamp()

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_general\_ci;

-- --------------------------------------------------------

--

-- Estructura de tabla para la tabla `testsgardner`

--

CREATE TABLE `testsgardner` (

`IdTest` int(11) NOT NULL,

`IdAlumno` int(11) NOT NULL,

`Respuestas` text DEFAULT NULL,

`Puntajes` text DEFAULT NULL,

`FechaAplicacion` datetime DEFAULT current\_timestamp()

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_general\_ci;

-- --------------------------------------------------------

--

-- Estructura de tabla para la tabla `usuarios`

--

CREATE TABLE `usuarios` (

`IdUsuario` int(11) NOT NULL,

`NombreUsuario` varchar(80) NOT NULL,

`CorreoElectronico` varchar(150) NOT NULL,

`ContrasenaHash` varchar(255) NOT NULL,

`IdRol` int(11) NOT NULL,

`FechaCreacion` datetime DEFAULT current\_timestamp(),

`contrasena\_hash` varchar(255) NOT NULL,

`correo\_electronico` varchar(150) NOT NULL,

`fecha\_creacion` datetime(6) DEFAULT NULL,

`nombre\_usuario` varchar(80) NOT NULL,

`id\_rol` int(11) NOT NULL,

`activo` bit(1) DEFAULT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_general\_ci;

--

-- Índices para tablas volcadas

--

--

-- Indices de la tabla `alumnos`

--

ALTER TABLE `alumnos`

ADD PRIMARY KEY (`IdAlumno`),

ADD UNIQUE KEY `IdUsuario` (`IdUsuario`);

--

-- Indices de la tabla `auditoria`

--

ALTER TABLE `auditoria`

ADD PRIMARY KEY (`IdAuditoria`),

ADD KEY `IdUsuario` (`IdUsuario`);

--

-- Indices de la tabla `calificaciones`

--

ALTER TABLE `calificaciones`

ADD PRIMARY KEY (`IdCalificacion`),

ADD KEY `IdMatricula` (`IdMatricula`);

--

-- Indices de la tabla `cursos`

--

ALTER TABLE `cursos`

ADD PRIMARY KEY (`IdCurso`),

ADD KEY `IdDocente` (`IdDocente`);

--

-- Indices de la tabla `docentes`

--

ALTER TABLE `docentes`

ADD PRIMARY KEY (`IdDocente`),

ADD UNIQUE KEY `IdUsuario` (`IdUsuario`);

--

-- Indices de la tabla `flyway\_schema\_history`

--

ALTER TABLE `flyway\_schema\_history`

ADD PRIMARY KEY (`installed\_rank`),

ADD KEY `flyway\_schema\_history\_s\_idx` (`success`);

--

-- Indices de la tabla `matriculas`

--

ALTER TABLE `matriculas`

ADD PRIMARY KEY (`IdMatricula`),

ADD UNIQUE KEY `IdAlumno` (`IdAlumno`,`IdCurso`),

ADD KEY `IdCurso` (`IdCurso`);

--

-- Indices de la tabla `refreshtokens`

--

ALTER TABLE `refreshtokens`

ADD PRIMARY KEY (`IdRefreshToken`),

ADD UNIQUE KEY `Token` (`Token`),

ADD KEY `IdUsuario` (`IdUsuario`);

--

-- Indices de la tabla `respuestasia`

--

ALTER TABLE `respuestasia`

ADD PRIMARY KEY (`IdRespuesta`),

ADD KEY `IdSolicitud` (`IdSolicitud`);

--

-- Indices de la tabla `roles`

--

ALTER TABLE `roles`

ADD PRIMARY KEY (`IdRol`),

ADD UNIQUE KEY `NombreRol` (`NombreRol`);

--

-- Indices de la tabla `solicitudesia`

--

ALTER TABLE `solicitudesia`

ADD PRIMARY KEY (`IdSolicitud`),

ADD KEY `IdDocente` (`IdDocente`),

ADD KEY `IdCurso` (`IdCurso`),

ADD KEY `IdAlumno` (`IdAlumno`);

--

-- Indices de la tabla `testsgardner`

--

ALTER TABLE `testsgardner`

ADD PRIMARY KEY (`IdTest`),

ADD KEY `IdAlumno` (`IdAlumno`);

--

-- Indices de la tabla `usuarios`

--

ALTER TABLE `usuarios`

ADD PRIMARY KEY (`IdUsuario`),

ADD UNIQUE KEY `NombreUsuario` (`NombreUsuario`),

ADD UNIQUE KEY `CorreoElectronico` (`CorreoElectronico`),

ADD UNIQUE KEY `UKduxldumspflsqyka52vo72hse` (`correo\_electronico`),

ADD UNIQUE KEY `UKof5vabgukahdwmgxk4kjrbu98` (`nombre\_usuario`),

ADD KEY `IdRol` (`IdRol`);

--

-- AUTO\_INCREMENT de las tablas volcadas

--

--

-- AUTO\_INCREMENT de la tabla `alumnos`

--

ALTER TABLE `alumnos`

MODIFY `IdAlumno` int(11) NOT NULL AUTO\_INCREMENT;

--

-- AUTO\_INCREMENT de la tabla `auditoria`

--

ALTER TABLE `auditoria`

MODIFY `IdAuditoria` int(11) NOT NULL AUTO\_INCREMENT;

--

-- AUTO\_INCREMENT de la tabla `calificaciones`

--

ALTER TABLE `calificaciones`

MODIFY `IdCalificacion` int(11) NOT NULL AUTO\_INCREMENT;

--

-- AUTO\_INCREMENT de la tabla `cursos`

--

ALTER TABLE `cursos`

MODIFY `IdCurso` int(11) NOT NULL AUTO\_INCREMENT;

--

-- AUTO\_INCREMENT de la tabla `docentes`

--

ALTER TABLE `docentes`

MODIFY `IdDocente` int(11) NOT NULL AUTO\_INCREMENT;

--

-- AUTO\_INCREMENT de la tabla `matriculas`

--

ALTER TABLE `matriculas`

MODIFY `IdMatricula` int(11) NOT NULL AUTO\_INCREMENT;

--

-- AUTO\_INCREMENT de la tabla `refreshtokens`

--

ALTER TABLE `refreshtokens`

MODIFY `IdRefreshToken` int(11) NOT NULL AUTO\_INCREMENT;

--

-- AUTO\_INCREMENT de la tabla `respuestasia`

--

ALTER TABLE `respuestasia`

MODIFY `IdRespuesta` int(11) NOT NULL AUTO\_INCREMENT;

--

-- AUTO\_INCREMENT de la tabla `roles`

--

ALTER TABLE `roles`

MODIFY `IdRol` int(11) NOT NULL AUTO\_INCREMENT, AUTO\_INCREMENT=7;

--

-- AUTO\_INCREMENT de la tabla `solicitudesia`

--

ALTER TABLE `solicitudesia`

MODIFY `IdSolicitud` int(11) NOT NULL AUTO\_INCREMENT;

--

-- AUTO\_INCREMENT de la tabla `testsgardner`

--

ALTER TABLE `testsgardner`

MODIFY `IdTest` int(11) NOT NULL AUTO\_INCREMENT;

--

-- AUTO\_INCREMENT de la tabla `usuarios`

--

ALTER TABLE `usuarios`

MODIFY `IdUsuario` int(11) NOT NULL AUTO\_INCREMENT;

--

-- Restricciones para tablas volcadas

--

--

-- Filtros para la tabla `alumnos`

--

ALTER TABLE `alumnos`

ADD CONSTRAINT `alumnos\_ibfk\_1` FOREIGN KEY (`IdUsuario`) REFERENCES `usuarios` (`IdUsuario`);

--

-- Filtros para la tabla `auditoria`

--

ALTER TABLE `auditoria`

ADD CONSTRAINT `auditoria\_ibfk\_1` FOREIGN KEY (`IdUsuario`) REFERENCES `usuarios` (`IdUsuario`);

--

-- Filtros para la tabla `calificaciones`

--

ALTER TABLE `calificaciones`

ADD CONSTRAINT `calificaciones\_ibfk\_1` FOREIGN KEY (`IdMatricula`) REFERENCES `matriculas` (`IdMatricula`);

--

-- Filtros para la tabla `cursos`

--

ALTER TABLE `cursos`

ADD CONSTRAINT `cursos\_ibfk\_1` FOREIGN KEY (`IdDocente`) REFERENCES `docentes` (`IdDocente`);

--

-- Filtros para la tabla `docentes`

--

ALTER TABLE `docentes`

ADD CONSTRAINT `docentes\_ibfk\_1` FOREIGN KEY (`IdUsuario`) REFERENCES `usuarios` (`IdUsuario`);

--

-- Filtros para la tabla `matriculas`

--

ALTER TABLE `matriculas`

ADD CONSTRAINT `matriculas\_ibfk\_1` FOREIGN KEY (`IdAlumno`) REFERENCES `alumnos` (`IdAlumno`),

ADD CONSTRAINT `matriculas\_ibfk\_2` FOREIGN KEY (`IdCurso`) REFERENCES `cursos` (`IdCurso`);

--

-- Filtros para la tabla `refreshtokens`

--

ALTER TABLE `refreshtokens`

ADD CONSTRAINT `refreshtokens\_ibfk\_1` FOREIGN KEY (`IdUsuario`) REFERENCES `usuarios` (`IdUsuario`);

--

-- Filtros para la tabla `respuestasia`

--

ALTER TABLE `respuestasia`

ADD CONSTRAINT `respuestasia\_ibfk\_1` FOREIGN KEY (`IdSolicitud`) REFERENCES `solicitudesia` (`IdSolicitud`);

--

-- Filtros para la tabla `solicitudesia`

--

ALTER TABLE `solicitudesia`

ADD CONSTRAINT `solicitudesia\_ibfk\_1` FOREIGN KEY (`IdDocente`) REFERENCES `docentes` (`IdDocente`),

ADD CONSTRAINT `solicitudesia\_ibfk\_2` FOREIGN KEY (`IdCurso`) REFERENCES `cursos` (`IdCurso`),

ADD CONSTRAINT `solicitudesia\_ibfk\_3` FOREIGN KEY (`IdAlumno`) REFERENCES `alumnos` (`IdAlumno`);

--

-- Filtros para la tabla `testsgardner`

--

ALTER TABLE `testsgardner`

ADD CONSTRAINT `testsgardner\_ibfk\_1` FOREIGN KEY (`IdAlumno`) REFERENCES `alumnos` (`IdAlumno`);

--

-- Filtros para la tabla `usuarios`

--

ALTER TABLE `usuarios`

ADD CONSTRAINT `usuarios\_ibfk\_1` FOREIGN KEY (`IdRol`) REFERENCES `roles` (`IdRol`);

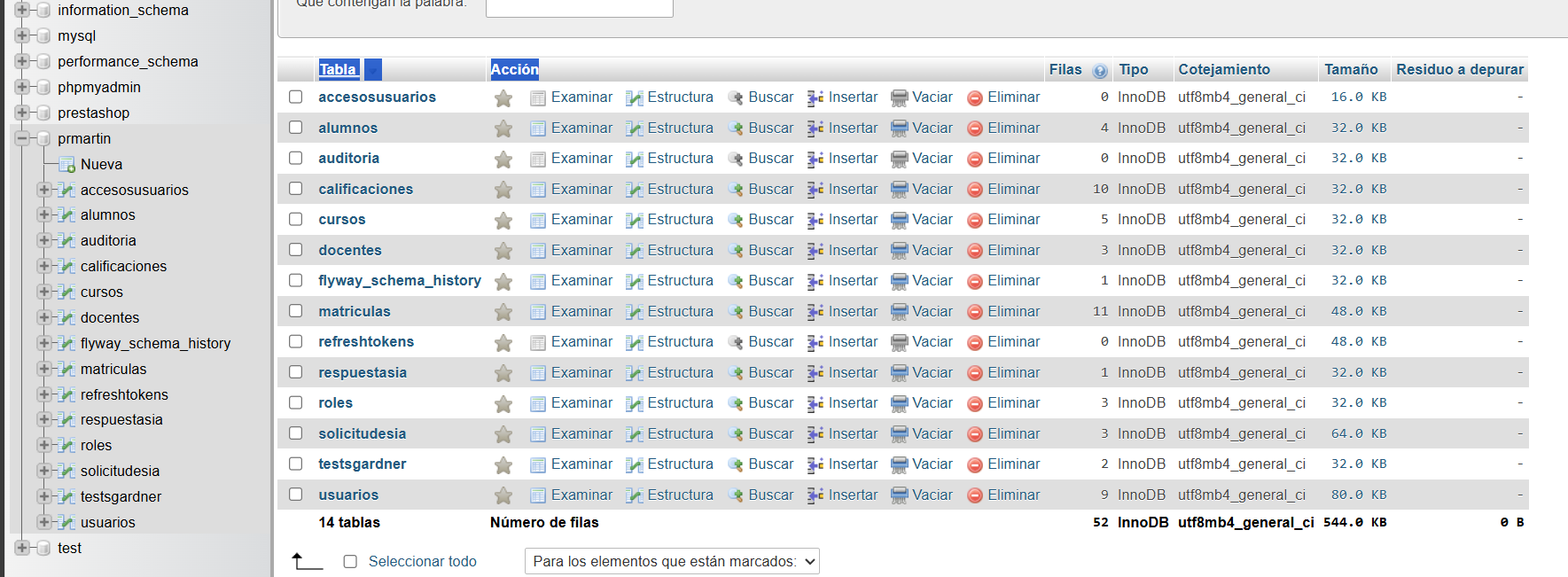
COMMIT;

/\*!40101 SET CHARACTER\_SET\_CLIENT=@OLD\_CHARACTER\_SET\_CLIENT \*/;

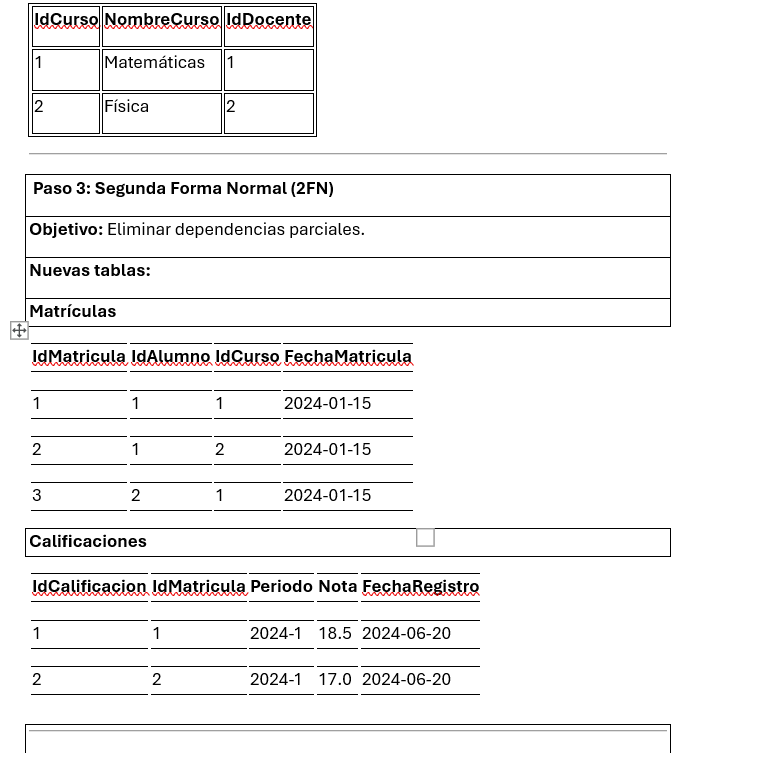
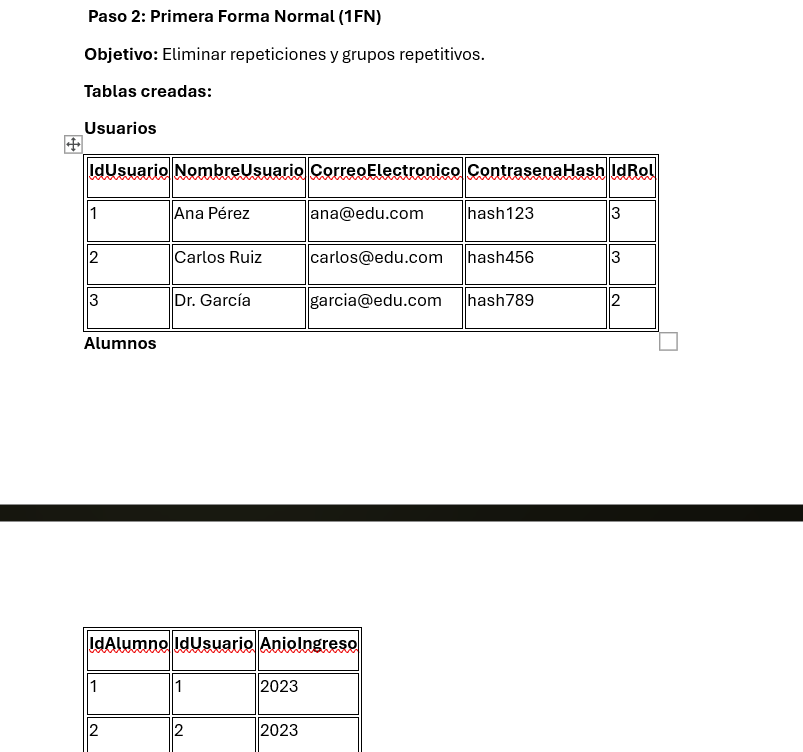
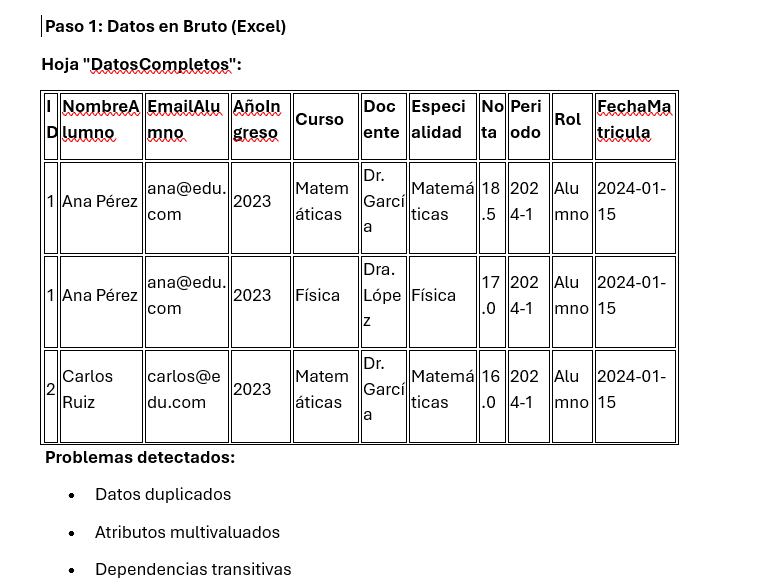
/\*!40101 SET CHARACTER\_SET\_RESULTS=@OLD\_CHARACTER\_SET\_RESULTS \*/;

/\*!40101 SET COLLATION\_CONNECTION=@OLD\_COLLATION\_CONNECTION \*/;

## 4.2 Historia de Usuario 0.2: Creación de la base de datos.



## 4.3 Historia de Usuario 0.3: Normalización de la base de datos.



## 4.4 Historia de Usuario 1.7: Acceso de administrador al sistema.

Archivo: backend/DB

@Bean

public SecurityFilterChain filterChain(HttpSecurity *http*) throws Exception {

http

.cors(cors -> cors.configurationSource(corsConfigurationSource()))

.csrf(csrf -> csrf.disable())

.sessionManagement(session -> session.sessionCreationPolicy(SessionCreationPolicy.STATELESS))

.authorizeHttpRequests(auth -> auth

*// Endpoints públicos*

.requestMatchers("/api/auth/\*\*").permitAll()

.requestMatchers("/api/test/\*\*").permitAll()

.requestMatchers("/actuator/health").permitAll()

.requestMatchers("/actuator/info").permitAll()

.requestMatchers("/swagger-ui/\*\*").permitAll()

.requestMatchers("/v3/api-docs/\*\*").permitAll()

.requestMatchers("/swagger-resources/\*\*").permitAll()

.requestMatchers("/webjars/\*\*").permitAll()

.requestMatchers("/").permitAll()

.requestMatchers("/error").permitAll()

*// Endpoints de administrador requieren autenticación*

.requestMatchers("/api/admin/\*\*").hasRole("ADMINISTRADOR")

.requestMatchers("/api/docente/\*\*").hasAnyRole("ADMINISTRADOR", "DOCENTE")

.requestMatchers("/api/alumno/\*\*").hasAnyRole("ADMINISTRADOR", "DOCENTE", "ALUMNO")

.requestMatchers("/api/auditoria/\*\*").hasRole("ADMINISTRADOR")

*// Actuator endpoints protegidos*

.requestMatchers("/actuator/\*\*").hasRole("ADMINISTRADOR")

*// Todos los demás endpoints requieren autenticación*

.anyRequest().authenticated()

)

@Override

protected void doFilterInternal(HttpServletRequest *request*, HttpServletResponse *response*,

FilterChain *filterChain*) throws ServletException, IOException {

final String authorizationHeader = request.getHeader("Authorization");

if (authorizationHeader != null && authorizationHeader.startsWith("Bearer ")) {

String jwt = authorizationHeader.substring(7);

try {

String username = jwtUtil.extractUsername(jwt);

Integer idUsuario = jwtUtil.extractIdUsuario(jwt);

String rol = jwtUtil.extractRol(jwt);

if (username != null && SecurityContextHolder.getContext().getAuthentication() == null) {

if (jwtUtil.validateToken(jwt, username)) {

*// Crear autenticación con el rol correcto*

UsernamePasswordAuthenticationToken authToken =

new UsernamePasswordAuthenticationToken(

username,

null,

Collections.singletonList(new SimpleGrantedAuthority("ROLE\_" + rol))

);

*// Agregar detalles adicionales*

authToken.setDetails(idUsuario);

SecurityContextHolder.getContext().setAuthentication(authToken);

}

}

} catch (Exception *e*) {

logger.error("Error al procesar JWT: " + e.getMessage());

}

}

## 4.5 Historia de Usuario 1.8: Registro de fecha y hora de acceso del administrador.

Archivo: backend/DB

*// Listar todos los administradores (usuarios con rol ADMIN)*

@GetMapping("/administradores")

@PreAuthorize("hasRole('ADMIN')")

public ResponseEntity<List<Usuario>> obtenerAdmins() {

List<Usuario> admins = usuarioService.findAllByRoleName("ADMIN");

return ResponseEntity.ok(admins);

}

*// Obtener admin por id (filtro por usuario y rol)*

@GetMapping("/administradores/{id}")

@PreAuthorize("hasRole('ADMIN')")

public ResponseEntity<Usuario> obtenerAdminPorId(@PathVariable Integer *id*) {

Optional<Usuario> usuario = usuarioService.findByIdIfRoleName(id, "ADMIN");

return usuario.map(ResponseEntity::ok).orElse(ResponseEntity.notFound().build());

}

*// Crear admin (crea un Usuario y le asigna el rol ADMIN)*

@PostMapping("/administradores")

@PreAuthorize("hasRole('ADMIN')")

public ResponseEntity<?> crearAdmin(@Valid @RequestBody UsuarioDto *dto*) {

Map<String, Object> resp = new HashMap<>();

try {

if (usuarioService.existsByNombreUsuario(dto.getNombreUsuario())) {

resp.put("exito", false);

resp.put("mensaje", "El nombre de usuario ya existe");

return ResponseEntity.badRequest().body(resp);

}

if (usuarioService.existsByCorreoElectronico(dto.getCorreoElectronico())) {

resp.put("exito", false);

resp.put("mensaje", "El correo electrónico ya existe");

return ResponseEntity.badRequest().body(resp);

}

Usuario creado = usuarioService.createAdminFromDto(dto);

resp.put("exito", true);

resp.put("mensaje", "Administrador creado exitosamente");

resp.put("administrador", creado);

return ResponseEntity.created(URI.create("/api/admin/administradores/" + creado.getIdUsuario())).body(resp);

} catch (Exception *e*) {

resp.put("exito", false);

resp.put("mensaje", "Error al crear administrador: " + e.getMessage());

return ResponseEntity.status(500).body(resp);

}

}

## 4.6 Historia de Usuario 1.9: Restricción de acceso a perfiles de administrador.

Archivo: backend/DB

@Bean

public SecurityFilterChain filterChain(HttpSecurity *http*) throws Exception {

http

.cors(cors -> cors.configurationSource(corsConfigurationSource()))

.csrf(csrf -> csrf.disable())

.sessionManagement(session -> session.sessionCreationPolicy(SessionCreationPolicy.STATELESS))

.authorizeHttpRequests(auth -> auth

*// Endpoints públicos*

.requestMatchers("/api/auth/\*\*").permitAll()

.requestMatchers("/api/test/\*\*").permitAll()

.requestMatchers("/actuator/health").permitAll()

.requestMatchers("/actuator/info").permitAll()

.requestMatchers("/swagger-ui/\*\*").permitAll()

.requestMatchers("/v3/api-docs/\*\*").permitAll()

.requestMatchers("/swagger-resources/\*\*").permitAll()

.requestMatchers("/webjars/\*\*").permitAll()

.requestMatchers("/").permitAll()

.requestMatchers("/error").permitAll()

*// Endpoints de administrador requieren autenticación*

.requestMatchers("/api/admin/\*\*").hasRole("ADMINISTRADOR")

.requestMatchers("/api/docente/\*\*").hasAnyRole("ADMINISTRADOR", "DOCENTE")

.requestMatchers("/api/alumno/\*\*").hasAnyRole("ADMINISTRADOR", "DOCENTE", "ALUMNO")

.requestMatchers("/api/auditoria/\*\*").hasRole("ADMINISTRADOR")

*// Actuator endpoints protegidos*

.requestMatchers("/actuator/\*\*").hasRole("ADMINISTRADOR")

*// Todos los demás endpoints requieren autenticación*

.anyRequest().authenticated()

)

## 4.7 Historia de Usuario 1.10: Cierre de sesión manual.

Archivo: inicio.component.html

inicio.component.ts

</div>

<button class="logout-button" (click)="logout()">Cerrar Sesión</button>

</div>

logout() {

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

Swal.fire({

title: 'Sesión cerrada',

text: 'Ahora no tienes acceso a la aplicación',

icon: 'success',

confirmButtonText: 'Aceptar',

});

}

## 4.8 Historia de Usuario 1.11: Cierre de sesión por inactividad

Archivo: app.component.ts

export class AppComponent {

title = 'Proyecto\_IA';

*private* timeoutId: any;

*private* *readonly* inactivityTime = 15 \* 60 \* 1000; *// 15 minutos*

constructor(*private* router: Router) {

this.resetTimer();

}

@HostListener('document:mousemove')

@HostListener('document:keydown')

@HostListener('document:click')

@HostListener('document:scroll')

resetTimer() {

if (this.router.url === '/login') {

clearTimeout(this.timeoutId);

*return*;

}

clearTimeout(this.timeoutId);

this.timeoutId = setTimeout(() => this.logout(), this.inactivityTime);

}

logout() {

if (this.router.url !== '/login') {

localStorage.clear();

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

Swal.fire({

title: 'Sesión cerrada',

text: 'Se detecto inactividad, Se ha cerrado la sesión',

icon: 'warning',

confirmButtonText: 'Aceptar',

});

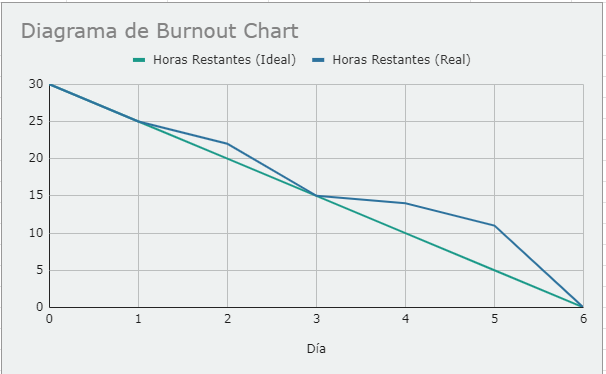
}

}

}

# 5. Burndown Chart

| Backlog ID | Historia de Usuario | Tiempo Estimado | 17/09/2025 | 18/092025 | 19/09/2025 | 20/09/2025 | 22/09/2025 | 23/09/2025 |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Dia1** | **Dia2** | **Dia3** | **Dia4** | **Dia5** | **Dia6** |
| HU-0.1 | Diseño de la base de datos. | 2 | 2 |  |  |  |  |  |
| HU-0.2 | Creación de la base de datos. | 1 | 1 |  |  |  |  |  |
| HU-0.3 | Normalización de la base de datos. | 3 | 2 | 1 |  |  |  |  |
| HU-1.7 | Acceso de administrador al sistema | 5 |  | 1 | 4 |  |  |  |
| HU-1.8 | Registro de fecha y hora de acceso del administrador | 6 |  | 1 | 3 | 1 | 1 |  |
| HU-1.9 | Restricción de acceso a perfiles de administrador | 5 |  |  |  |  | 2 | 3 |
| HU-1.10 | Cierre de sesión manual | 3 |  |  |  |  |  | 3 |
| HU-1.11 | Cierre de sesión por inactividad | 3 |  |  |  |  |  | 3 |
| Tiempo de trabajo | | 28 | 23 | 20 | 13 | 12 | 9 | 0 |



# 6. Análisis del Sprint

* Se cumplió con todas las tareas estimadas.
* Los días 2, 4 y 5 tuvieron contratiempos respecto al alcance estimado.
* La implementación de la base de datos se tuvo que replantear debido al agrupamiento de tablas para registros dentro de la aplicación, tomando la decisión de mantener la base de datos intacta y agregar tablas no relacionadas con el resto de la base de datos.
* El flujo de trabajo fue un poco abrupto debido a los contratiempos del desarrollo y falta de tiempo de los integrantes.

# 7. Conclusiones

El Sprint 1 se completó exitosamente, implementando todas las tareas propuestas y poniendo en mesa la posibilidad de extender nuevas funciones para la implementación futura, Se avanzó el apartado del login casi por completo y también parte de la visualización del administrador.

El equipo demostró capacidad de entrega puntual y buena gestión de riesgos menores. Las métricas indican que la carga pudo estar mejor distribuida y los objetivos del sprint se cumplieron completamente.