

# **ESCUELA POLITÉCNICA NACIONAL**

## **FACULTAD DE INGENIERÍA DE SISTEMAS**

**Construcción de Software – GR2SW**



**ESCUELA  
POLITÉCNICA  
NACIONAL**

## **PROYECTO SEGUNDO BIMESTRE**

**Integrantes:**

Evelin Rocha

Profesor: Vicente Eguez.

Fecha de entrega: 23/01/2026

**Construcción y evolución de Software.**

**Proyecto:** Plataforma Multimedia Galeto

## 1. Introducción: Contexto, Problemática y Propósito

La plataforma **Galeto** surge como una solución tecnológica avanzada ante la creciente necesidad de espacios digitales que integren de manera cohesiva la gestión visual y la experiencia auditiva. En el panorama actual, los usuarios se enfrentan a plataformas fragmentadas; Galeto propone un ecosistema unificado donde la fotografía y la música convergen bajo estrictos estándares de seguridad, moderación y retroalimentación social dinámica.

El propósito fundamental de este despliegue técnico es demostrar cómo una arquitectura desacoplada, soportada por un ciclo de vida de desarrollo automatizado en Azure DevOps, puede evolucionar desde un módulo básico de autenticación hasta un sistema complejo de interacción en tiempo real. Este documento detalla la ingeniería detrás del carrusel dinámico, la seguridad basada en tokens y la gestión de requerimientos a través de cuatro sprints incrementales.

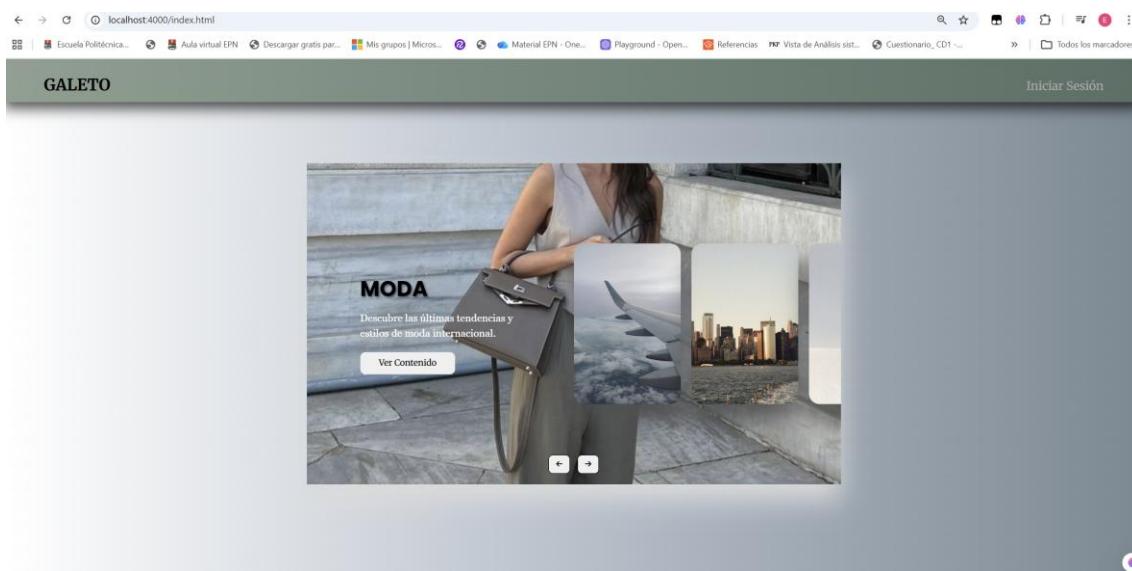


Ilustración 1. Ventana de inicio Galeto.

Ilustración 2. Gestión de Galeto en azureDevops..

## 2. Arquitectura del Proyecto y Estrategia de Integración

La robustez de Galeto reside en su arquitectura modular, diseñada para minimizar la deuda técnica y maximizar el rendimiento en el renderizado del lado del cliente.

### 2.1. Frontend: Ingeniería de Interacción y Manipulación del DOM

El frontend ha sido concebido bajo el paradigma de **JavaScript Vanilla (ES6+)**, priorizando la velocidad de carga y la eficiencia de los recursos:

- **Algoritmo de Carrusel Infinito (Sprint 3 - HU O5H9):** A diferencia de las soluciones convencionales que clonian elementos consumiendo memoria adicional, se desarrolló un algoritmo de rotación física de nodos. Al interactuar con los selectores `.next` o `.prev`, el sistema captura el nodo desbordado y lo reinsera mediante `appendChild` o `prepend` en el extremo opuesto del contenedor padre. Esto garantiza que el flujo multimedia sea perpetuo sin importar la densidad de la carga de datos.

```

let next = document.querySelector('.next')
let prev = document.querySelector('.prev')

next.addEventListener('click', function () {
  let items = document.querySelectorAll('.item')
  document.querySelector('.slide').appendChild(items[0])
})

prev.addEventListener('click', function () {
  let items = document.querySelectorAll('.item')
  document.querySelector('.slide').prepend(items[items.length - 1]) // here the length of items = 6
})

```

*Ilustración 3. Implementación del carrusel.*

- **Modularidad Estética y CSS Avanzado:** El sistema visual se basa en una jerarquía de selectores de alta especificidad. Se implementó una biblioteca de variables CSS para el manejo de la paleta de colores y tipografía (Poppins/Merriweather), asegurando que cualquier cambio de marca pueda ejecutarse modificando un solo archivo raíz, sin afectar la estructura del layout.

## 2.2. Capa de Seguridad, Identidad y API

- **Protocolo de Autenticación JWT (Sprint 1 - HU O11H6):** La seguridad no se delega únicamente al servidor. El frontend implementa un interceptor de lógica que gestiona **JSON Web Tokens**. Tras un login exitoso contra la API en localhost:4000, el token se almacena encriptado en el localStorage. Una función dedicada (decodeJWT) extrae el perfil del usuario para habilitar o restringir componentes visuales (como el botón de borrado para administradores) en milisegundos.

```

// Lanza un tarea y configuración final de mailer
// Mailer significa que los archivos subidos se manejarán correctamente.
const mailer = mailer({
  storage: storage,
  filefilter: filefilter,
  limits: { filesize: 5 * 1024 * 1024 } // Límite de SMB
});

// -----
// RUTAS DE AUTENTICACIÓN (sin cambios funcionales)
// -----

// ruta /register (OOGH y Online)
app.post('/register', (req, res) => {
  const {username, email, password} = req.body;
  if (!username || !email || !password)
    return res.status(400).json({ success: false, message: 'Faltan campos obligatorios.' });

  try {
    const saltRounds = 10;
    const passwordHash = await bcrypt.hash(password, saltRounds);
    const pool = await poolPromise;
    await pool.request()
      .query(`INSERT INTO users (username, email, passwordhash) VALUES ('${username}', '${email}', '${passwordHash}')`)
      .on('error', (err) => {
        console.error(`Error en el registro: ${err}`);
        if (err.number === 2627 || err.message.includes('UNIQUE KEY constraint')) {
          if (err.message.includes('email'))
            message = 'El correo electrónico ya está registrado.';
          else if (err.message.includes('username'))
            message = 'El nombre de usuario ya no está disponible.';
          else
            message = 'El nombre de usuario o correo electrónico ya están en uso.';
        }
        res.status(400).json({ success: false, message: message });
      });
  } catch (err) {
    console.error(`Error en el registro: ${err}`);
    if (err.number === 2627 || err.message.includes('UNIQUE KEY constraint')) {
      if (err.message.includes('email'))
        message = 'El correo electrónico ya está registrado.';
      else if (err.message.includes('username'))
        message = 'El nombre de usuario ya no está disponible.';
      else
        message = 'El nombre de usuario o correo electrónico ya están en uso.';
    }
    res.status(400).json({ success: false, message: message });
  }
});

// ruta /login
app.post('/login', async (req, res) => {
  const {username, password} = req.body;
  if (!username || !password) return res.status(400).json({ success: false, message: 'Faltan credenciales.' });

  try {
    const pool = await poolPromise;
    const result = await pool.request()
      .input('username', sql.VarChar(100), username)
      .query(`SELECT UserID, Username, PasswordHash, RoleID FROM users WHERE Username = '${username}'`);
    const user = result.recordset[0];
    if (!user) return res.status(400).json({ success: false, message: 'Email o contraseña incorrectos.' });
  } catch (err) {
    console.error(`Error en la autenticación: ${err}`);
    res.status(400).json({ success: false, message: 'Error en la autenticación.' });
  }
});

```

Ilustración 4. Implementación del protocolos de autenticación.

- Filtros de Integridad en el Registro (Sprint 1 - HU O2H1):** Se diseñó un motor de validación basado en Expresiones Regulares (Regex) que actúa como primer muro de defensa. Este motor no solo valida la estructura sintáctica del email, sino que restringe el acceso a dominios de confianza (@gmail, @outlook, @hotmail), reduciendo drásticamente la creación de cuentas "spam".

```

// --- validación de email ---
function validarEmail(email) {
  const regex = `/^([A-Z0-9_ñ]+)@outlook|gmail|hotmail\.com$/i`;
  if (regex.test(email))
    return true;
  else {
    showModa(`Por favor, ingrese un email válido con uno de los siguientes dominios: Outlook, Gmail, Hotmail`); Katuska, 2 months ago + funcionalidad registro e inicio sesión
    return false;
  }
}

// -----
// manejo completo del registro (envio con fetch)
// -----
formRegister.addEventListener('submit', async (event) => {
  event.preventDefault(); // detener el envío por defecto

  const username = document.querySelector('.form-container.sign-up input[name="username"]').value;
  const email = emailInputRegister.value;
  const password = passwordInputRegister.value;

  // validar Email
  if (!validarEmail(email))
    return;

  // validar Longitud de Contraseña
  if (password.length < 8)
    showModa(`La contraseña debe tener al menos 8 caracteres. Por favor, ingrese una contraseña más segura.`);
  return;

  // enviar al Backend
  try {
    const response = await fetch(`${API_URL}/register`, {
      method: 'POST',
      headers: { 'Content-Type': 'application/json' },
      body: JSON.stringify({ username, email, password })
    });

    const data = await response.json();

    if (response.ok) { // Status 200-299
      showModa(data.message);
      // redirigir al panel de login
      container.classList.remove('active');
    } else {
      // muestra error de unicidad (email ya existe)
      showModa(data.message || 'Error desconocido en el registro.');
    }
  } catch (error) {
    console.error('Error de red/servidor:', error);
    showModa('Error de conexión con el servidor. Verifique que el backend esté corriendo en puerto 4000.');
  }
});

```

*Ilustración 5. Implementación de la integridad del registro.*

### **3. Estrategia de Pipelines y Ciclo de Vida (Azure Pipelines)**

La evolución del software en Galeto se rige por un pipeline de **CI/CD** que garantiza que ningún código llegue a producción sin ser sometido a un riguroso escrutinio automatizado.

#### **3.1. Integración Continua (CI): Calidad en el Origen**

Cada vez que un desarrollador realiza un *commit* hacia **Azure Repos**, el pipeline ejecuta un flujo de trabajo de tres etapas:

1. **Análisis Estático (Linting):** Se audita el código CSS y JS para prevenir malas prácticas (como estilos "inline" o variables globales) que degraden el rendimiento.
2. **Pruebas de Validación Lógica:** Se ejecutan pruebas de caja negra sobre las funciones críticas (ej. validación de contraseña de 8 caracteres). Si una prueba falla, el build se rompe inmediatamente y se notifica al equipo.
3. **Gestión de Artefactos:** Se generan paquetes optimizados donde los scripts son minificados para reducir el ancho de banda necesario en el cliente final.

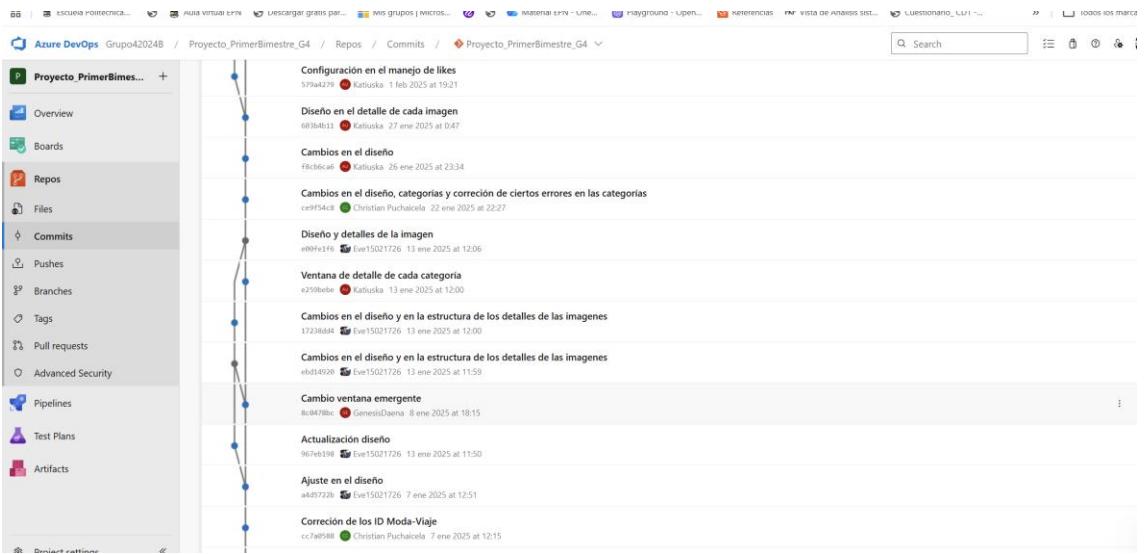


Ilustración 6. Integración continua.

### 3.2. Entrega Continua (CD): Despliegue Controlado

- Ambientes de Staging:** Se utiliza un entorno espejo donde se prueban las funcionalidades multimedia antes del lanzamiento oficial. Aquí se validó la interactividad del Sprint 4.
- Aprobaciones por Puertas (Approval Gates):** Se configuraron aprobaciones manuales obligatorias. Un líder técnico debe certificar visualmente que el diseño responsive en 480px es funcional antes de que el pipeline libere la versión en la rama main.

### 4. Gestión de Historias de Usuario (Azure Boards)

La trazabilidad entre el negocio y el código es el pilar de **Azure Boards**. Cada ticket representa una necesidad real extraída de los documentos de Sprint:

Fase	Enfoque	de Historias		de Resultado Tangible
		Ingeniería	Usuario	
		Vinculadas		
Sprint 1	Seguridad Estructural	O2H1, O11H6	O6H3,	Registro seguro, login con JWT y perfil de usuario persistente.
Sprint 2	Interacción y Socialización	O7H4, O4H8	O2H7,	Sistema de likes dinámico, asociación música-imagen y feed público.
Sprint 3	Multimedia y UX	O5H9, O17H12	O7H15,	Ventanas emergentes de audio y sistema de votación blindado por ID.
Sprint 4	Moderación y Feedback	O7H17, O10H17	O8H16,	Panel de notificaciones con estados de lectura y logout seguro.

## 6. Estrategia de Flujos de Desarrollo (Azure Repos)

Se adoptó un modelo de ramificación **GitFlow** avanzado para permitir el trabajo en paralelo de múltiples funcionalidades:

- **Feature Branching:** Cada funcionalidad se aisló en su propia rama. Por ejemplo, la rama feature/notificaciones permitió iterar sobre el panel dinámico sin poner en riesgo la estabilidad del login.
- **Pull Requests (PR) y Code Review:** Las fusiones a la rama develop requieren que el código sea revisado por otro desarrollador. Se busca optimización de

selectores, eliminación de código muerto y cumplimiento de la guía de estilos de Galeto.

- Protección de Ramas:** La rama main es inviolable; solo permite fusiones mediante PRs que hayan superado exitosamente el Pipeline de CI.

ID	HISTORIAS DE USUARIO	SEMANA 1					SEMANA 2				
		LUNES	MARTES	MIERCOLES	JUEVES	VIERNES	SABADO	LUNES	MARTES	MIERCOLES	JUEVES
SPRINT 1 (10/11 - 16/10)	Como usuario, deseo registrarme en la aplicación para poder acceder a todas las funcionalidades que ofrece. <b>O2H1</b>										
	Como usuario, deseo acceder a mi perfil para ver mis fotos y editar mi información personal. <b>O6H3</b>										
	Como desarrollador, quiero que las contraseñas se almacenen cifradas para proteger la información de los usuarios. <b>O11H6</b>										
	Como usuario no registrado, deseo ver el feed público para conocer el contenido antes de registrarme. <b>O4H8</b>										
SPRINT 2 (17/10 - 23/10)	Como administrador, deseo ver el contenido subido y la lista de usuarios para gestionarlos. <b>O12H2</b>										
	Como usuario, deseo dar "like" a una foto para interactuar con otros usuarios. <b>O7H4</b>										
	Como usuario, deseo asociar tres canciones a una foto para mejorar la experiencia del usuario. <b>O2H7</b>										
	Como usuario, quiero que el sistema valide los formatos de archivo para asegurarme de que mis archivos se puedan subir correctamente sin errores. <b>O3H10</b>										

Ilustración 7. Planificación del sprint 1 y 2.

ID	HISTORIAS DE USUARIO	SEMANA 1					SEMANA 2				
		LUNES	MARTES	MIERCOLES	JUEVES	VIERNES	SABADO	LUNES	MARTES	MIERCOLES	JUEVES
SPRINT 3 (10/11 - 16/10)	Como usuario, deseo reproducir las canciones asociadas desde una ventana emergente sin salir del sitio. <b>O5H9</b>										
	Como usuario, quiero eliminar o editar mis propios comentarios para corregir errores o información equivocada. <b>O10H5</b>										
	Como usuario, deseo poder emitir solo un voto por canción, para garantizar la equidad y evitar votos duplicados. <b>O7H15</b>										
	Como usuario, deseo votar por la canción que más me guste para compartir gustos musicales. <b>O17H12</b>										
	Como usuario, deseo editar o eliminar mis propias fotos para corregir errores o borrar contenido viejo. <b>O8H13</b>										
SPRINT 4 (17/10 - 23/10)	Como usuario, deseo que el sistema me muestre un error si intento subir imágenes o audios demasiado pesados. <b>O17H14</b>										
	Como administrador, deseo eliminar contenido publicado desde mi perfil, para mantener la calidad y el control de la plataforma. <b>O8H16</b>										
	Como usuario, deseo disponer de un botón de cerrar sesión visible en todas las pestañas, para salir de la aplicación de forma segura en cualquier momento. <b>O10H17</b>										
	Como usuario, deseo que al hacer clic en el logo Picsound sea redirigido al panel de categorías, sin importar la pestana en la que me encuentre, para mejorar la navegación. <b>O8H18</b>										
	Como administrador, deseo presentar una interfaz adecuada para poder generar una experiencia adecuada. <b>O01H1</b>										

Ilustración 8. Planificación del sprint 3 y 4.

## 7. Control de Calidad, Revisiones y Rendimiento

El aseguramiento de la calidad (QA) en Galeto trasciende las pruebas básicas, enfocándose en la experiencia de usuario final:

1. **Auditoría de Rendimiento:** Se verificó que la manipulación del DOM en el carrusel no generara fugas de memoria (*memory leaks*), manteniendo un uso de CPU estable durante transiciones largas.
2. **Validación de UX Responsiva:** Se ejecutaron pruebas en diversos dispositivos móviles. El CSS se ajustó para que elementos críticos, como el panel de notificaciones (.notifications-panel), mantuvieran un scroll independiente y no bloquearan la navegación principal.
3. **Trazabilidad Boards-Repos:** Se implementó una política de commits donde cada cambio debe referenciar el ID del Work Item de Azure Boards. Esto permite realizar auditorías de código meses después y entender el motivo de cada cambio.

## 8. Herramientas, Conexiones y Evidencias Técnicas

El ecosistema de trabajo se orquestó mediante conexiones inteligentes:

- **Azure Boards ↔ Azure Repos:** Permite la vinculación automática; cada vez que se abre un PR, el estado de la Historia de Usuario cambia automáticamente a "En Revisión".
- **Postman ↔ API Testing:** Se integraron colecciones de Postman para validar los endpoints de la API (puerto 4000), garantizando que los datos JSON de las notificaciones se entreguen con la estructura correcta.
- **Anexos de Evidencia:** Se deben insertar aquí las capturas de los Sprints 1, 2, 3 y 4 que muestran los tableros de avance y las metas cumplidas.

ID	Title	Assigned To	State	Area Path	Tags
110	BUG EN CATEGORIAS	GENESIS DAENA VASCON...	Proposed	Proyecto_PrimerBimestre_G4	
109	CARGA DE IMAGENES	ALISON KATIUSKA LITA US...	Proposed	Proyecto_PrimerBimestre_G4	
56	Galería de Imagenes	CHRISTIAN GEOVANNY P...	Active	Proyecto_PrimerBimestre_G4	
41	Gestión de Usuarios	EVELIN XIMENA ROCHA R...	Active	Proyecto_PrimerBimestre_G4	
108	MUTIPICIDAD DE BOTÓN ME GUSTA	CHRISTIAN GEOVANNY P...	Proposed	Proyecto_PrimerBimestre_G4	
57	Navegación y experiencia del usuario	ALISON KATIUSKA LITA US...	Active	Proyecto_PrimerBimestre_G4	
58	Requerimientos no funcionales	GENESIS DAENA VASCON...	Active	Proyecto_PrimerBimestre_G4	

Ilustración 9. IWork items.

Active	Resolved	0/s	Closed
41 Gestión de Usuarios EVELIN XIMENA ROCHA ROCHA	57 Navegación y experiencia del usuario ALISON KATIUSKA LITA USHINA	1/1	
56 Galería de Imagenes CHRISTIAN GEOVANNY PUCHACELA CURIPOMA			
58 Requerimientos no funcionales GENESIS DAENA VASCONCE ESCOBAR			

Ilustración 10. Board.

Order	Work Item Type	Title	State	Effort	Business Area	Tags
1	Epic	> Gestión de Usuarios	Active		Business	
2	Epic	> Navegación y experiencia del usuario	Active		Business	
3	Epic	> Galería de Imagenes	Active		Business	
4	Epic	> Requerimientos no funcionales	Active		Business	

Ilustración 11. Backlogs.

**Sprint 1 Fundamentos del registro y autenticación segura**

Order	Title	State	Assigned To
1	RF01 Registro de Usuarios	Closed	GENESIS DAEN...
	Configuración de la base de datos para usuarios y contr...	Closed	GENESIS DAEN...
	• Implementación de un sistema de validación de emails.	Closed	ALISON KATIUS...
	• Configuración de seguridad para contraseñas.	Closed	GENESIS DAEN...
2	RF02 Inicio de Sesión	Closed	CHRISTIAN GE...
	• Implementación de formulario de inicio de sesión.	Closed	CHRISTIAN GE...
	• Validación de correo electrónico y contraseña.	Closed	CHRISTIAN GE...
	• Mensajes de error para credenciales incorrectas.	Closed	CHRISTIAN GE...
	• Redirección a la página principal si el inicio de sesión es...	Closed	CHRISTIAN GE...
+ 3	RNF01 Seguridad de Contraseñas	Closed	EVELIN XIMEN...
	Implementar validación de contraseña en el backend	Closed	EVELIN XIMEN...
	Evitar almacenamiento en texto plano	Closed	EVELIN XIMEN...

Ilustración 12. Sprint 1.

**Sprint 2 Gestión de contenido multimedia galería y carga**

Order	Title	State	Assigned To
2	RF10 Botón de 'Perfil' para Acceso a Información Personal d...	Closed	EVELIN XIMEN...
	• Agregar el botón Perfil a la barra de navegación	Closed	EVELIN XIMEN...
	• Configurar el backend para cargar los datos del usuario	Closed	EVELIN XIMEN...
3	RF08 Botón de navegación por categorías en la página web	Closed	EVELIN XIMEN...
	• Agregar el botón de Categorías	Closed	EVELIN XIMEN...
	• Implementar navegación a cada categoría	Closed	EVELIN XIMEN...
4	RF09 Funcionalidad de Retorno a 'Inicio' desde la Barra de ...	Closed	ALISON KATIUS...
	• Agregar el botón Inicio en la barra de navegación	Closed	ALISON KATIUS...
	• Configurar la redirección al hacer en el botón	Closed	ALISON KATIUS...
+ 5	RF07 Implementación de Botón de 'Me Gusta' en Fotos	Closed	EVELIN XIMEN...
	• Agregar el ícono de Me Gusta en cada foto	Closed	EVELIN XIMEN...
	• Implementar funcionalidad de dar Me Gusta	Closed	EVELIN XIMEN...
	• Implementar validaciones en el lado del servidor	Closed	ALISON KATIUS...
	• MÚTIPICIDAD DE BOTÓN ME GUSTA	Proposed	CHRISTIAN GE...
6	RNF07 Integridad de Datos	Closed	ALISON KATIUS...

Ilustración 13. Sprint 2.

**Sprint 3 Interacción del usuario votaciones y preferencias**

Order	Title	State	Assigned To
1	RF05 Carga de canciones a la imagen	Closed	CHRISTIAN GE...
	• CARGA DE IMAGENES	Proposed	ALISON KATIUS...
	• BUG EN CATEGORIAS	Proposed	GENESIS DAEN...
2	RF06 Votación de Canciones Asociadas a la Imagen	Closed	CHRISTIAN GE...

*Ilustración 14. Sprint 3.*

*Ilustración 15. Sprint 4..*

Name	Last change	Commits
.vscode	8 dic 2024	<a href="#">1c137b8d</a> Actualización para el registrar e ingresar con el uso de "node" Christian...
ímágenes	1 feb 2025	<a href="#">579a4272</a> Configuración en el manejo de likes Katiuska
node_modules	27 ene 2025	<a href="#">4e538be1</a> Manejo Likes GenesisDaen...
public	24 may 2025	<a href="#">5bea84a2</a> modificado modal.css Katiuska
.gitignore	25 ago 2025	<a href="#">4f813723</a> Renamed .gitignore.txt to .gitignore ALISON KATIUSKA LITA USHINA
ts app.js	24 may 2025	<a href="#">2656eb51</a> añadir archivo netlify.toml para desplegar Katiuska
netlify.toml	24 may 2025	<a href="#">2656eb51</a> añadir archivo netlify.toml para desplegar Katiuska
package-lock.json	10 dic 2024	<a href="#">639a8398</a> Nuevas funcionalidades añadidas Christian Puchacela
package.json	24 dic 2024	<a href="#">5f489381</a> Rediseño en la base de datos para solucionar la duplicidad de imágenes...

*Ilustración 16. Repositorio.*

## 9. Conclusiones y Visión Evolutiva del Software

La arquitectura técnica y la metodología aplicadas bajo el stack de Azure DevOps han transformado una idea conceptual en un producto de software robusto, escalable y mantenable. Galetto no solo cumple con las funcionalidades solicitadas en los cuatro sprints, sino que establece una base técnica sólida para futuras expansiones, como la integración de microservicios de streaming o sistemas de moderación automatizados por

IA. La trazabilidad lograda garantiza que la evolución del software sea transparente, segura y orientada siempre al valor del usuario final.