“AÑO DE LA RECUPERACIÓN Y CONSOLIDACIÓN DE LA ECONOMÍA PERUANA”

Título del Proyecto: Plataforma de Tutorías Virtuales para Estudiantes en Riesgo

ODS Vinculado: 4 - Educación de Calidad

ASIGNATURA:

ANÁLISIS Y DISEÑO DE SOFTWARE

NRC: 62152

DOCENTE:

Rosario Delia Osorio Contreras

INTEGRANTES:

- Antezana Alonzo Alexandra Dayana
- Molina Bendezu Leonel
- Tito Llactahuaman Angel Xaviel
- Velasquez Colorado Lionel

Aplicaciones:

Jira: Enlace de jira

Modelio: Enlace de modelio

Canva: Enlace de canva

Github: Enlace

Huancayo, 2025

Aplicaciones

Jira:https://continental-team-tebflnru.atlassian.net/jira/software/projects/PDTV/boards/100/backlog?epics=visible&selectedIssue=PDTV-11&atlOrigin=eyJpIjoiYjUxZDg0NDhmNTE4NDA4MGE2ODhjYTVlYTQ3NTc2ZDkiLCJwIjoiaiJ9

Canva:https://www.canva.com/design/DAG0gw6JEHw/JsPAo3ViGVL86QsIcmtkCw/edit?utm_content=DAG0gw6JEHw&utm_campaign=designshare&utm_medium=link2&utm_source=sharebutton

Git:
https://github.com/AlexandraAntezana/-Plataforma-de-Tutor-as-Virtuales-para-Estudiantes-en-Riesgo_ODS4

ÍNDICE

UNIDAD I – FUNDAMENTOS Y MODELADO INICIAL

CAPÍTULO 1. PRESENTACIÓN DEL PROYECTO

- 1.1. ODS Vinculado: Educación de Calidad (ODS 4)
- 1.2. Organización o Institución Beneficiaria
- 1.3. Problema Identificado
- 1.4. Solución Propuesta

CAPÍTULO 2. ANÁLISIS DE NECESIDADES Y REQUERIMIENTOS

- 2.1. Descripción del Problema
- 2.2. Necesidades de los Usuarios
- 2.3. Requerimientos Funcionales (RF)
- 2.4. Requerimientos No Funcionales (RNF)
- 2.5. Requerimientos de Dominio

CAPÍTULO 3. MODELOS INICIALES DEL SISTEMA

- Modelo funcional (diagrama de contexto)
- Casos de uso generales
- Modelo de procesos
- Modelo de datos (Modelo E-R)

UNIDAD II – MODELOS DE DISEÑO Y METODOLOGÍA ÁGIL

CAPÍTULO 4. MODELOS DE DISEÑO

- Modelo estructural (diagrama de clases inicial)
- Modelo de interacción (diagrama de secuencia)

CAPÍTULO 5. METODOLOGÍA DE TRABAJO (SCRUM)

- 5.1. Definición de la Metodología Ágil Usada
- 5.2. Backlog del Producto (Épicas e Historias de Usuario)
- 5.3. Planificación de Sprints
- 5.4. Herramientas Utilizadas

CONCLUSIONES Y RECOMENDACIONES

- Conclusiones del Equipo
- Lecciones Aprendidas
- Recomendaciones para Futuras Mejoras del Sistema

REFERENCIAS BIBLIOGRÁFICAS

ANEXOS

UNIDAD I – FUNDAMENTOS Y MODELADO INICIAL

CAPÍTULO 1. PRESENTACIÓN DEL PROYECTO

1.1. ODS Vinculado: Educación de Calidad (ODS 4)

La educación es reconocida como un derecho humano fundamental y un elemento indispensable para el desarrollo sostenible, tal como lo establece la Agenda 2030. Este objetivo busca "garantizar una educación inclusiva, equitativa y de calidad y promover oportunidades de aprendizaje durante toda la vida para todos", con especial énfasis en las poblaciones vulnerables.

1.2. Organización o Institución Beneficiaria

La institución beneficiaria es la Universidad Continental - Sede Huancayo. Los usuarios principales son estudiantes de pregrado con bajo rendimiento, docentes que ofrecen tutorías, y coordinadores académicos que supervisan el progreso estudiantil. También puede aplicarse a otras universidades e institutos de la región.

1.3. Problema Identificado

Las desigualdades educativas estructurales persisten en múltiples dimensiones, afectando el acceso, la participación y los resultados de aprendizaje, especialmente en contextos de vulnerabilidad socioeconómica 2. Estudios recientes demuestran que la falta de personalización en los procesos de enseñanza-aprendizaje contribuye significativamente al bajo rendimiento académico y al abandono escolar temprano 3. En América Latina, esta problemática se ve agravada por la limitada disponibilidad de sistemas de apoyo educativo personalizado fuera del horario escolar convencional.

1.4. Solución Propuesta

Desarrollamos una plataforma integral de tutorías virtuales que implementa modelos de diferenciación educativa basados en los principios establecidos por Tomlinson 4, donde el contenido, proceso y producto educativo se adaptan a las necesidades específicas de cada estudiante. La plataforma incorpora sistemas de recomendación inteligentes fundamentados en técnicas de filtrado colaborativo y basado en contenido 5, permitiendo emparejamientos óptimos entre tutores y estudiantes.

CAPÍTULO 2. ANÁLISIS DE NECESIDADES Y REQUERIMIENTOS

2.1. Descripción del Problema

El proceso actual de tutorías presenta deficiencias significativas en su organización y seguimiento, lo que genera barreras de acceso al apoyo académico 2. Estudiantes con bajo rendimiento enfrentan limitaciones para

acceder a refuerzo educativo fuera del horario escolar, agravando las desigualdades educativas existentes 3.

Esto genera tres consecuencias críticas:

1. Estudiantes abandonan la búsqueda de ayuda por la complejidad del proceso
2. Tutores potenciales (estudiantes destacados o docentes) no tienen forma de ofrecer sus servicios organizadamente
3. La institución no puede identificar ni intervenir con estudiantes que requieren atención urgente

2.2. Necesidades de los Usuarios

Estudiantes:

- Encontrar tutores calificados rápidamente según la materia
- Agendar sesiones en horarios flexibles sin complicaciones
- Acceder a tutorías desde cualquier ubicación
- Revisar el historial de sus sesiones y materiales compartidos
- Calificar la calidad de las tutorías recibidas

Tutores:

- Publicar su disponibilidad horaria de forma clara
- Gestionar solicitudes de tutoría eficientemente
- Recibir recordatorios de sesiones programadas
- Acceder a materiales o notas previas del estudiante
- Construir reputación mediante valoraciones

Coordinadores Académicos:

- Visualizar reportes de estudiantes en riesgo académico
- Monitorear la frecuencia y efectividad de las tutorías
- Identificar materias con mayor demanda de apoyo
- Exportar datos para análisis institucional

2.3. Requerimientos Funcionales (RF)

Tabla 1. Requerimientos Funcionales del Sistema

Código	Requerimiento	Prioridad
RF01	El sistema debe permitir registro e inicio de sesión con roles (estudiante, tutor, coordinador)	Alta
RF02	El sistema debe permitir búsqueda y agendamiento de sesiones de tutoría por materia y horario	Alta
RF03	El sistema debe integrar videollamadas y enviar notificaciones por correo	Alta
RF04	El sistema debe registrar historial de sesiones y permitir calificar tutores	Media
RF05	El sistema debe generar reportes de estudiantes en riesgo para coordinadores	Alta

2.4. Requerimientos No Funcionales (RNF)

Tabla 2. Requerimientos No Funcionales del Sistema

Código	Requerimiento	Descripción
RNF 01	Usabilidad	Interfaz intuitiva con máximo 3 clics para agendar una sesión

RNF 02	Rendimiento	Tiempo de respuesta menor a 2 segundos para búsquedas
RNF 03	Seguridad	Autenticación mediante JWT y encriptación de contraseñas
RNF 04	Compatibilidad	Responsive y funcional en navegadores principales (Chrome, Firefox, Safari, Edge)

2.5. Requerimientos de Dominio

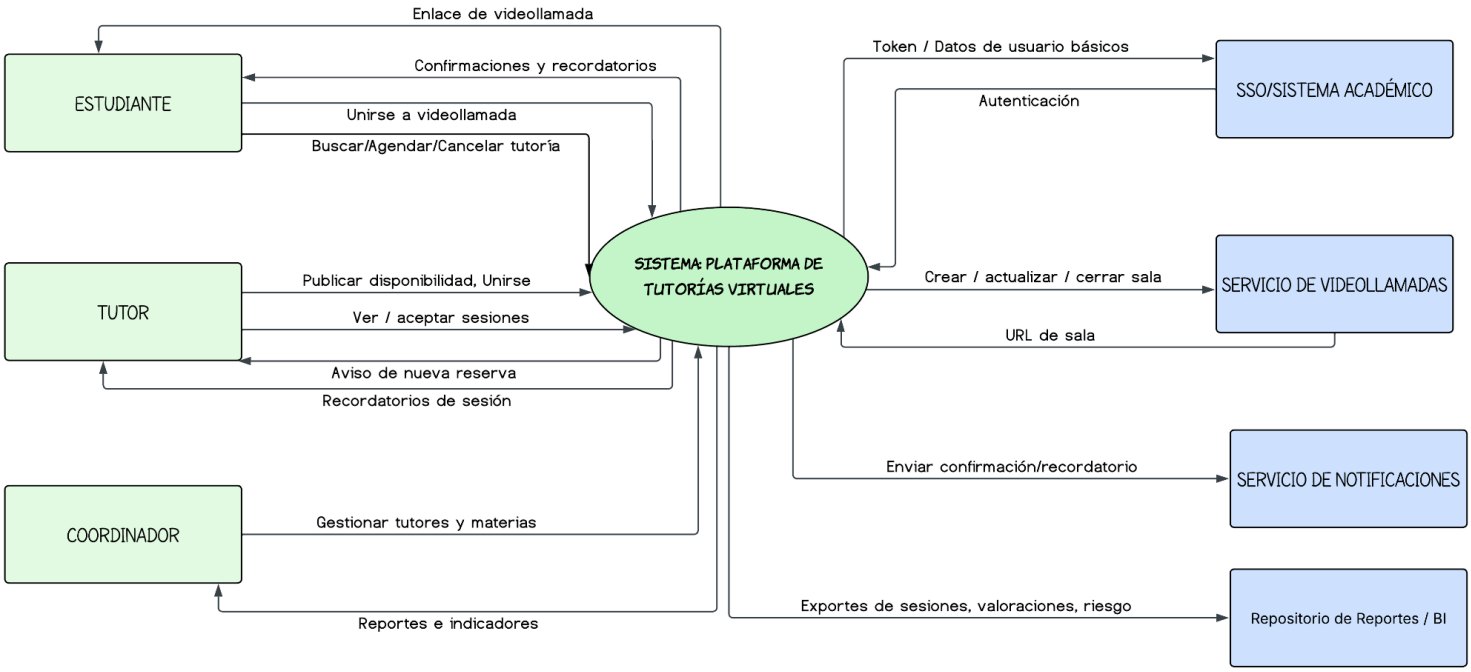
Tabla 3. Requerimientos de Dominio

Código	Requerimiento	Justificación
RD01	Una sesión de tutoría debe durar mínimo 30 minutos	Estándar académico para sesiones efectivas
RD02	Los estudiantes pueden cancelar sesiones con mínimo 2 horas de anticipación	Política institucional de respeto al tiempo del tutor
RD03	Los tutores deben tener promedio mínimo de 15 en la materia que enseñan	Requisito de calidad académica

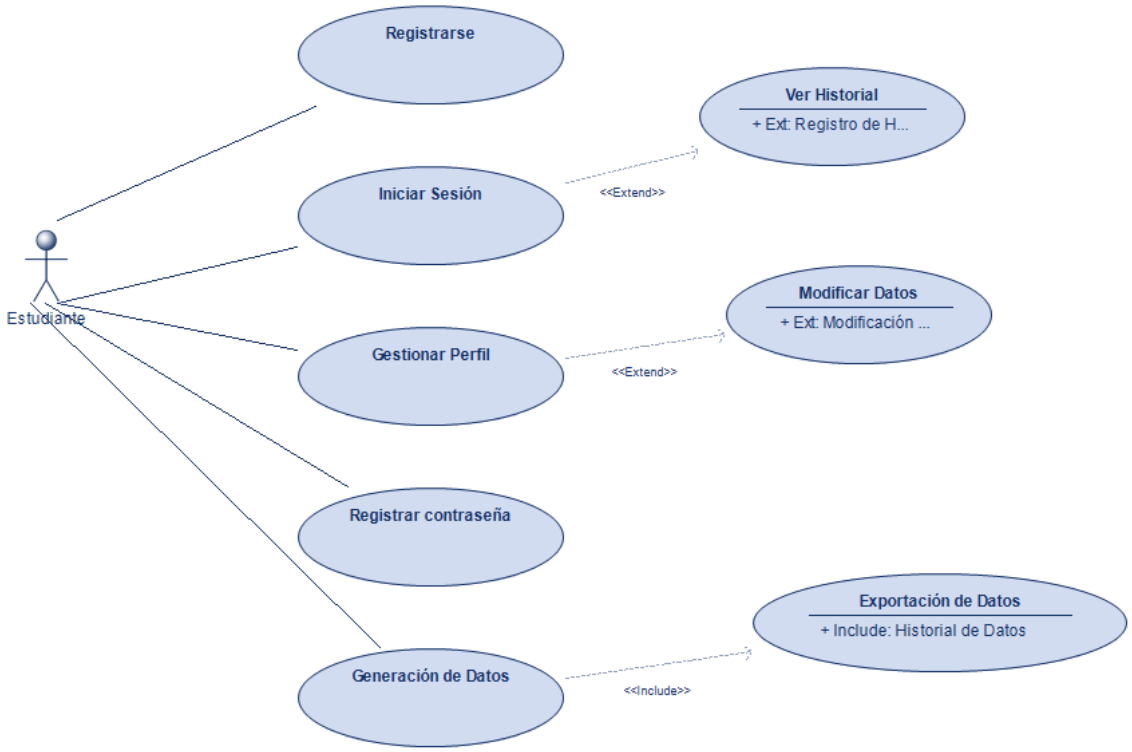
RD04	Las sesiones se programan entre 7:00 - 22:00 horas en bloques de 30 minutos	Horario operativo universitario extendido

Capítulo 3. Modelos Iniciales del Sistema

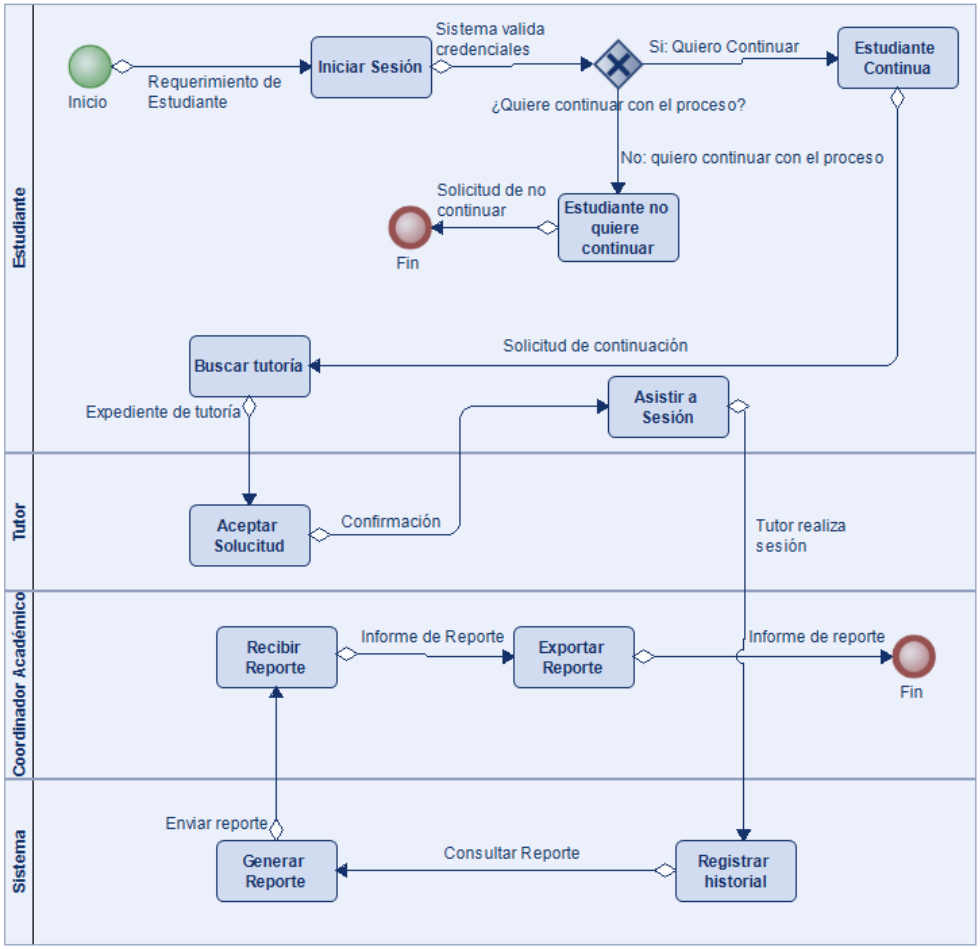
Modelo funcional (diagrama de contexto)



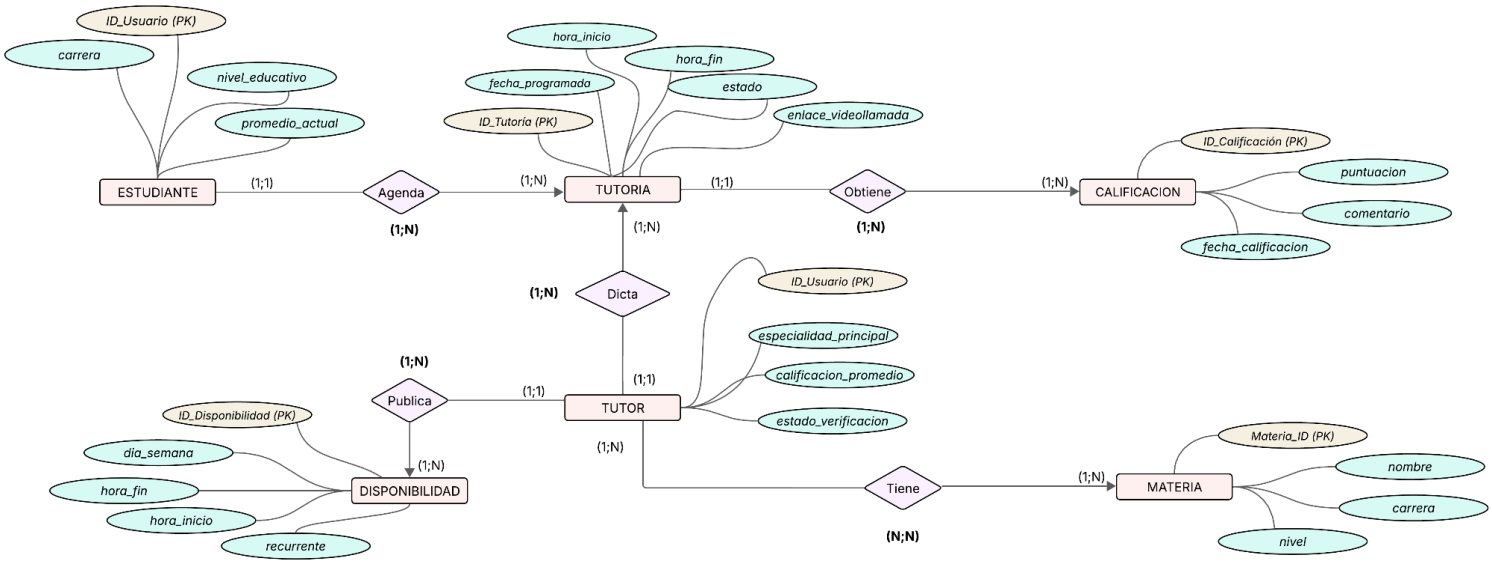
Casos de uso generales:


Registrarse
Ver Historial
+ Ext: Registro de H...
Iniciar Sesión
<<Extend>>
Estudiante
Modificar Datos
+ Ext: Modificación ...
Gestionar Perfil
<<Extend>>
Registrar contraseña
Exportación de Datos
+ Include: Historial de Datos
Generación de Datos
<<Include>>

Modelo de procesos



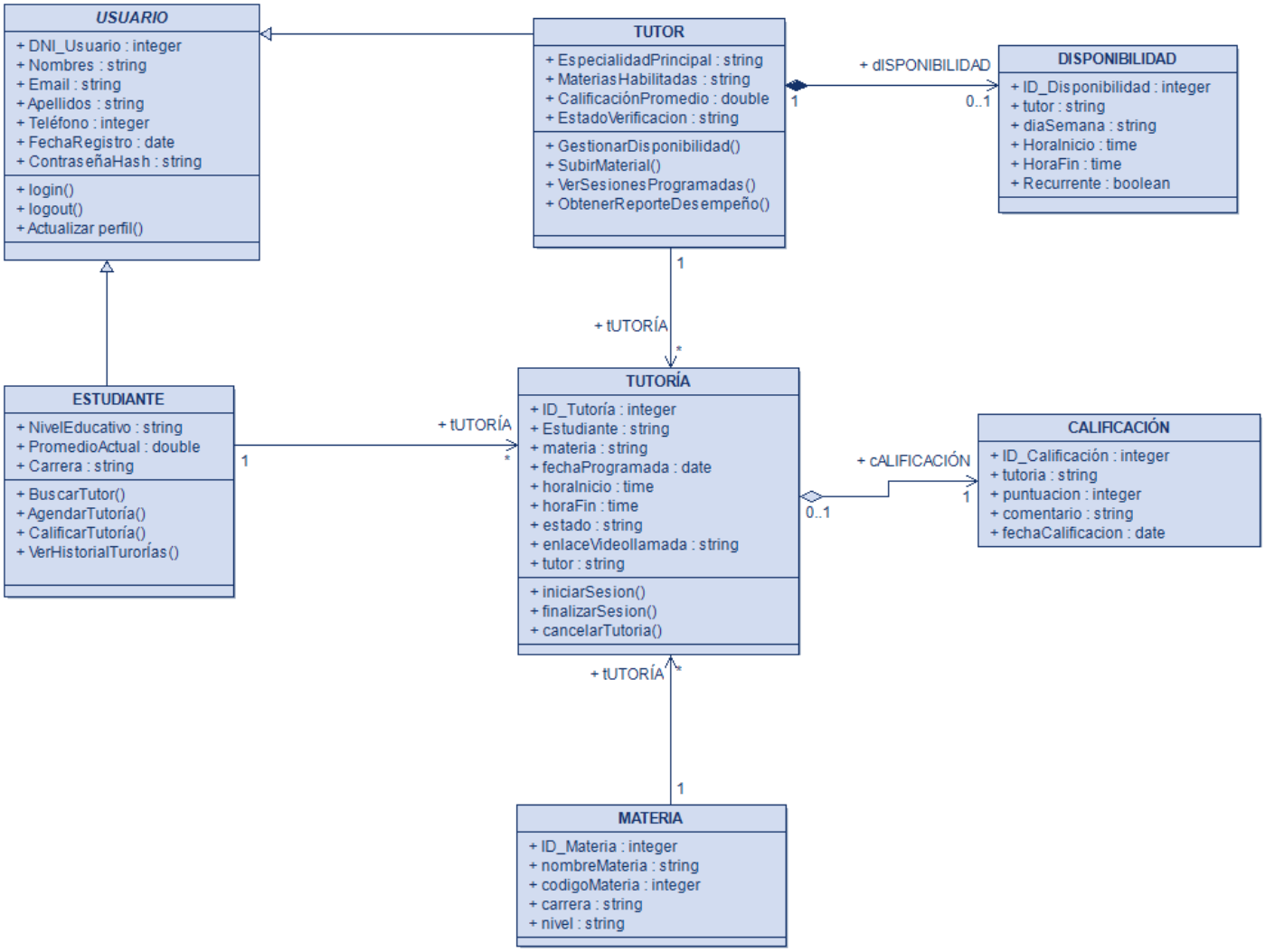
Modelo de datos (Modelo E-R)



Unidad II – Modelos de Diseño y Metodología Ágil (Semanas 5–7)

Capítulo 4. Modelos de Diseño

Modelo estructural (diagrama de clases inicial):



Modelo de interacción (diagrama de secuencia):

Diagrama de secuencia – Agendar tutoría. El Estudiante inicia sesión y busca tutores; la UI envía la solicitud al InscripcionService, que consulta disponibilidad en la base de datos. Al seleccionar tutor y horario, el servicio verifica conflictos: si hay cupo, genera el enlace de videollamada, registra la sesión y envía la confirmación al Estudiante y al Tutor; si no hay cupo, retorna “horario no disponible”.

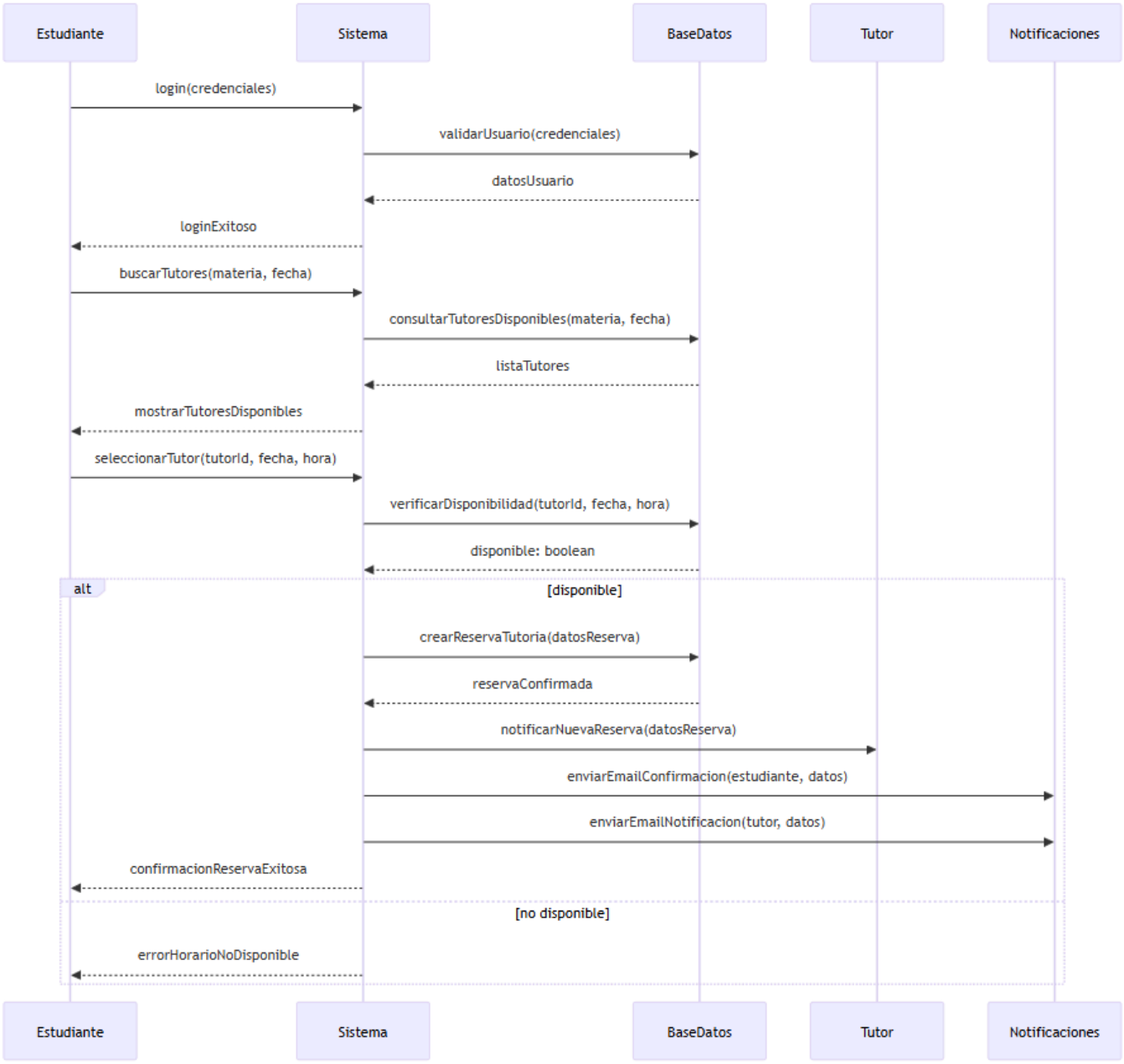
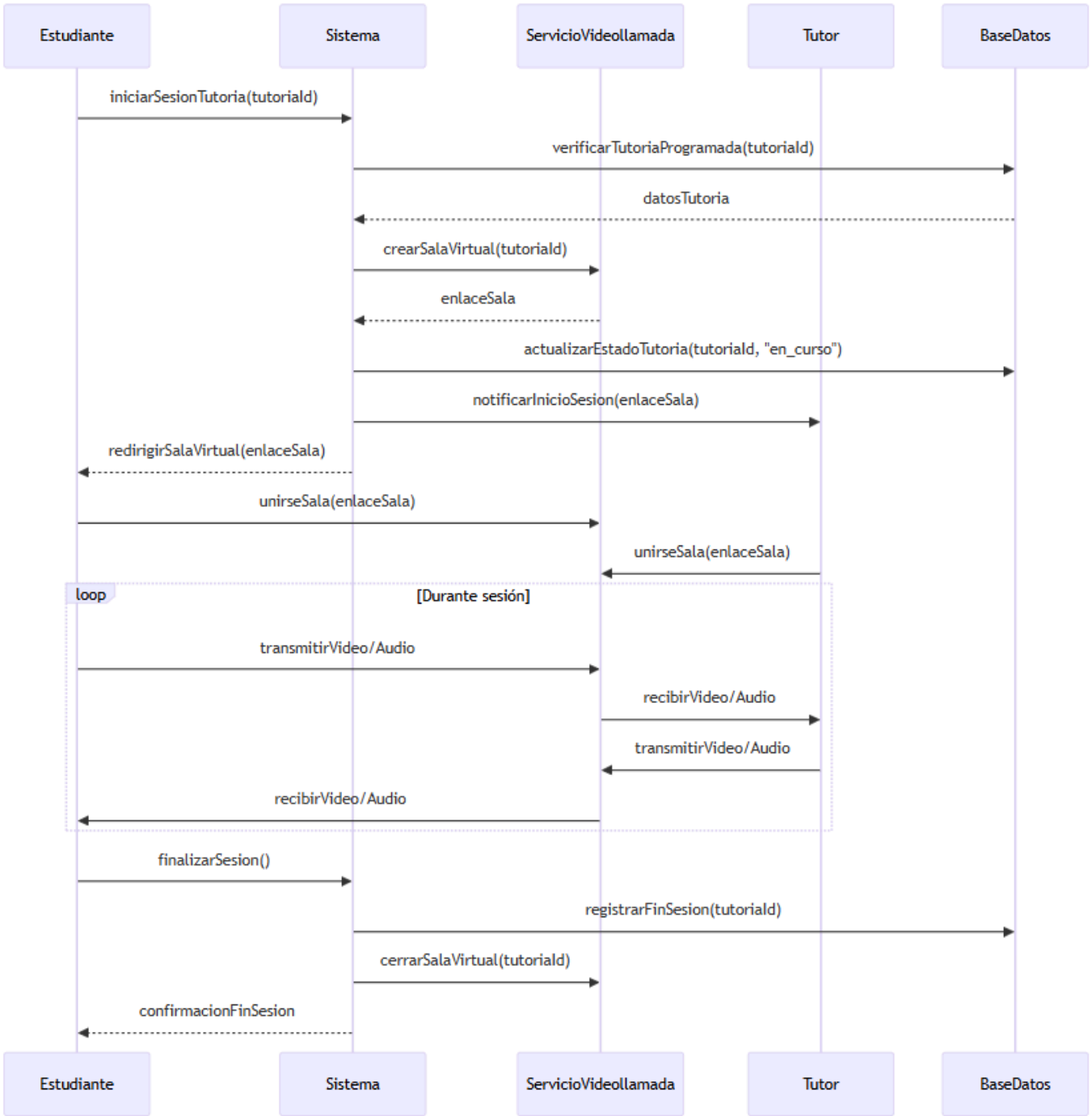


Diagrama de secuencia – Iniciar videollamada de tutoría. El Estudiante solicita iniciar sesión; el Sistema valida la tutoría, crea la sala virtual, redirige con el enlace, ambos participantes se unen, se transmite media (loop) y al finalizar se registra el fin y se cierra la sala.



CAPÍTULO 5. METODOLOGÍA DE TRABAJO (SCRUM)

5.1. Definición de la Metodología Ágil Usada

Para el desarrollo de la Plataforma de Tutorías Virtuales se implementó **SCRUM** como marco de trabajo ágil. SCRUM permite entregas incrementales del producto mediante ciclos cortos llamados sprints, facilitando la adaptación a cambios y la retroalimentación continua.

Roles del equipo:

- **Product Owner:** Coordinador académico que define prioridades del backlog
- **Scrum Master:** Rosario Osorio Contreras (facilita el proceso y elimina impedimentos)
- **Development Team:** Antezana Alexandra, Molina Leonel, Tito Angel, Velasquez Lionel

Eventos SCRUM implementados:

- Sprint Planning: Planificación al inicio de cada sprint (2 semanas)
- Daily Stand-up: Reuniones diarias de 15 minutos (virtuales)
- Sprint Review: Demostración del incremento al finalizar el sprint
- Sprint Retrospective: Análisis de mejoras del proceso

5.2. Backlog del Producto (Épicas e Historias de Usuario)

Épicas del Producto

EP01 - Gestión de Usuarios Permite el registro, autenticación y administración de perfiles de estudiantes, tutores y coordinadores.

EP02 - Sistema de Tutorías Facilita la búsqueda, agendamiento y ejecución de sesiones de tutoría virtual.

EP03 - Seguimiento y Reportes Proporciona herramientas de monitoreo y análisis para coordinadores académicos.

Historias de Usuario Priorizadas

ID	Historia de Usuario	Épica	Prioridad	Puntos
HU 01	Como estudiante quiero registrarme en el sistema para acceder a las tutorías	EP01	Alta	5
HU 02	Como estudiante quiero buscar tutores por materia para encontrar ayuda específica	EP02	Alta	8
HU 03	Como estudiante quiero agendar una sesión para recibir tutoría en horario disponible	EP02	Alta	13
HU 04	Como tutor quiero publicar mi disponibilidad para que estudiantes me encuentren	EP01	Alta	8
HU 05	Como estudiante quiero recibir notificaciones por correo para recordar mis sesiones	EP02	Media	5
HU 06	Como tutor quiero acceder a videollamada integrada para realizar la tutoría virtual	EP02	Alta	13
HU 07	Como estudiante quiero calificar al tutor para ayudar a otros estudiantes	EP02	Media	5
HU 08	Como coordinador quiero ver reportes de estudiantes en riesgo para intervenir oportunamente	EP03	Alta	13

Criterios de Aceptación (Ejemplo - HU03):

- El estudiante puede seleccionar fecha y hora del calendario del tutor
- El sistema valida que no haya conflictos de horario
- Se envía confirmación automática por correo a ambas partes
- La sesión queda registrada en el historial del estudiante

5.3. Planificación de Sprints

Sprint 1 (Semanas 1-2)

Objetivo: Implementar funcionalidades básicas de gestión de usuarios y búsqueda de tutores

Historias incluidas:

- HU01: Registro de usuarios (5 puntos)
- HU02: Búsqueda de tutores (8 puntos)
- HU04: Publicación de disponibilidad (8 puntos)

Capacidad del equipo: 21 puntos de historia

Entregables:

- Módulo de autenticación funcional
- Base de datos con tablas de usuarios y disponibilidad
- Interfaz de búsqueda de tutores con filtros

Definición de Done:

- Código revisado y aprobado por el equipo
- Pruebas unitarias ejecutadas exitosamente
- Funcionalidad desplegada en entorno de desarrollo
- Documentación técnica actualizada

Sprint 2 (Semanas 3-4)

Objetivo: Desarrollar sistema de agendamiento y videollamadas

Historias incluidas:

- HU03: Agendamiento de sesiones (13 puntos)
- HU05: Notificaciones por correo (5 puntos)
- HU06: Integración de videollamadas (13 puntos)

Capacidad del equipo: 31 puntos de historia

Entregables:

- Calendario interactivo de agendamiento
- Sistema de notificaciones automáticas

- Integración con API de Google Meet
- Módulo de gestión de sesiones

Riesgos identificados:

- Complejidad de integración con API externa de videollamadas
- Posibles problemas de sincronización de horarios

Mitigación:

- Investigación previa de documentación de APIs
- Implementación de sistema de logs para debugging

5.4. Herramientas Utilizadas

Gestión de Proyecto:

- **Jira:** Gestión del backlog, sprints y seguimiento de historias de usuario
- **Confluence:** Documentación técnica y actas de reuniones

Diseño y Modelado:

- **Draw.io:** Creación de diagramas UML (casos de uso, secuencia, actividad)
- **Dbdiagram.io:** Diseño del modelo entidad-relación de la base de datos
- **Figma:** Prototipado de interfaces de usuario

Desarrollo:

- **Visual Studio Code:** IDE principal para desarrollo
- **GitHub:** Control de versiones y colaboración en código
- **Postman:** Pruebas de APIs REST

Comunicación:

- **Discord:** Reuniones diarias y comunicación del equipo
- **Google Meet:** Sprint reviews y retrospectivas
- **WhatsApp:** Coordinación rápida y notificaciones urgentes

Unidad III – Diseño de Software

Capítulo 6. Diseño de Arquitectura y Patrones

6.1 Estrategia de diseño del software

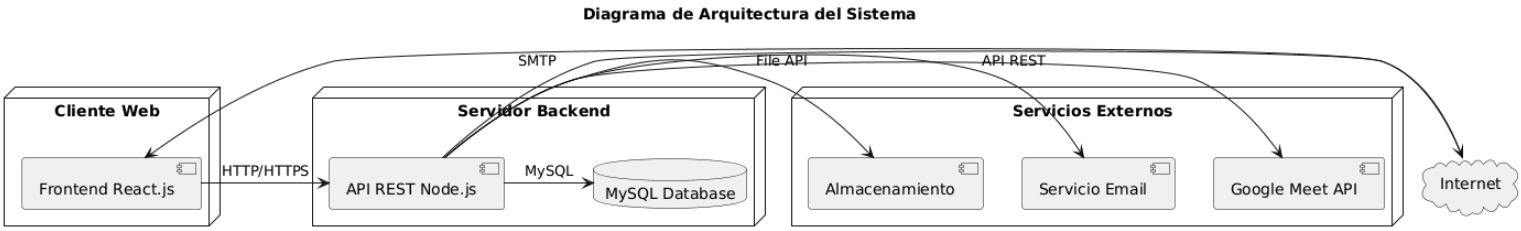
La estrategia de diseño adoptada para la Plataforma de Tutorías Virtuales sigue un enfoque basado en componentes con separación clara de responsabilidades. Se implementa una arquitectura en capas que facilita el mantenimiento, escalabilidad y testabilidad del sistema.

6.2 Tipo de arquitectura del sistema

Arquitectura de Referencia: MVC (Modelo-Vista-Controlador) en el frontend y Arquitectura en Capas (Layered Architecture) en el backend.

Justificación:

- Separación de preocupaciones: Cada capa tiene responsabilidades específicas
- Escalabilidad: Permite escalar componentes de forma independiente
- Mantenibilidad: Facilita la implementación de cambios y correcciones
- Testabilidad: Cada capa puede ser probada de forma aislada



6.3 Patrones de diseño aplicados

1. Patrón Repository

- Aplicación: Acceso a datos en la capa de persistencia
- Beneficio: Abstraction de la lógica de acceso a datos
- Implementación: Clases Repository para cada entidad principal (Usuario, Tutoría, Sesión)

2. Patrón Factory

- Aplicación: Creación de objetos complejos (notificaciones, sesiones de videollamada)
- Beneficio: Centraliza la lógica de creación
- Implementación: NotificationFactory, SessionFactory

3. Patrón Strategy

- Aplicación: Algoritmos de recomendación de tutores
- Beneficio: Intercambiable entre diferentes estrategias
- Implementación: TutorMatchingStrategy interface con implementaciones concretas

4. Patrón Observer

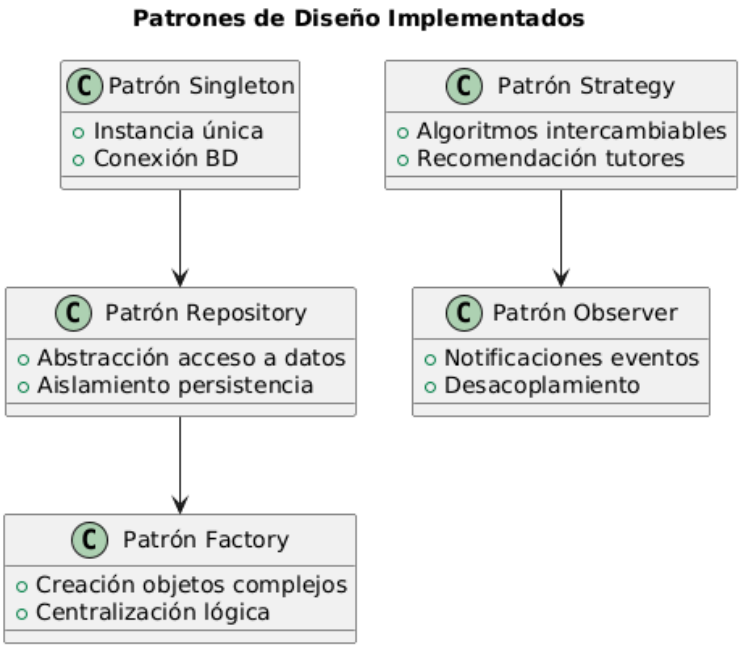
- Aplicación: Sistema de notificaciones
- Beneficio: Desacoplamiento entre emisores y receptores de eventos
- Implementación: Eventos de sistema (nueva sesión, recordatorio, calificación)

5. Patrón Singleton

- Aplicación: Conexión a base de datos, configuración del sistema
- Beneficio: Una única instancia global
- Implementación: DatabaseConnection, AppConfig

6.4 Diseño estructural

Diagrama de Componentes:



Capítulo 7. Diseño Detallado de la Base de Datos

7.1 Modelo lógico y físico

Modelo Lógico Normalizado (3FN):

Entidades Principales:

- Usuario (id, email, password, rol, nombre, apellido, telefono, estado)
- Estudiante (id, usuario_id, codigo, promedio_general, estado_academico)
- Tutor (id, usuario_id, especialidades, promedio_minimo, estado_verificacion)
- Materia (id, codigo, nombre, descripcion, creditos)
- TutorEspecialidad (id, tutor_id, materia_id, promedio_materia)
- SesionTutoria (id, estudiante_id, tutor_id, materia_id, fecha_hora, duracion, estado, enlace_videollamada)
- Calificacion (id, sesion_id, puntuacion, comentarios, fecha_calificacion)
- Disponibilidad (id, tutor_id, dia_semana, hora_inicio, hora_fin, recurrente)

7.2 Script SQL

```
-- Creación de la Base de Datos

CREATE DATABASE plataforma_tutorias;

USE plataforma_tutorias;


-- Tabla Usuario

CREATE TABLE usuario (

    id INT PRIMARY KEY AUTO_INCREMENT,

    email VARCHAR(255) UNIQUE NOT NULL,

    password_hash VARCHAR(255) NOT NULL,

    rol ENUM('estudiante', 'tutor', 'coordinador') NOT NULL,

    nombre VARCHAR(100) NOT NULL,
```

```
    apellido VARCHAR(100) NOT NULL,
    telefono VARCHAR(20),
    estado ENUM('activo', 'inactivo', 'pendiente') DEFAULT 'pendiente',
    fecha_creacion TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    fecha_actualizacion TIMESTAMP DEFAULT CURRENT_TIMESTAMP ON UPDATE
CURRENT_TIMESTAMP
);

-- Tabla Estudiante
CREATE TABLE estudiante (
    id INT PRIMARY KEY AUTO_INCREMENT,
    usuario_id INT UNIQUE NOT NULL,
    codigo_estudiante VARCHAR(20) UNIQUE NOT NULL,
    promedio_general DECIMAL(3,2),
    estado_academico ENUM('regular', 'riesgo', 'graduado') DEFAULT 'regular',
    FOREIGN KEY (usuario_id) REFERENCES usuario(id) ON DELETE CASCADE
);

-- Tabla Tutor
CREATE TABLE tutor (
    id INT PRIMARY KEY AUTO_INCREMENT,
    usuario_id INT UNIQUE NOT NULL,
    especialidades TEXT,
    promedio_minimo DECIMAL(3,2) DEFAULT 15.00,
    estado_verificacion ENUM('pendiente', 'aprobado', 'rechazado') DEFAULT 'pendiente',
    calificacion_promedio DECIMAL(2,1) DEFAULT 0.0,
    sesiones_completadas INT DEFAULT 0,
```

```
    FOREIGN KEY (usuario_id) REFERENCES usuario(id) ON DELETE CASCADE
);

-- Tabla Materia
CREATE TABLE materia (
    id INT PRIMARY KEY AUTO_INCREMENT,
    codigo VARCHAR(10) UNIQUE NOT NULL,
    nombre VARCHAR(100) NOT NULL,
    descripcion TEXT,
    creditos INT NOT NULL,
    estado ENUM('activa', 'inactiva') DEFAULT 'activa'
);

-- Tabla Sesión de Tutoría
CREATE TABLE sesion_tutoria (
    id INT PRIMARY KEY AUTO_INCREMENT,
    estudiante_id INT NOT NULL,
    tutor_id INT NOT NULL,
    materia_id INT NOT NULL,
    fecha_hora DATETIME NOT NULL,
    duracion_minutos INT DEFAULT 30,
    estado ENUM('programada', 'en_curso', 'completada', 'cancelada') DEFAULT
'programada',
    enlace_videollamada VARCHAR(500),
    calificacion_id INT,
    fecha_creacion TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (estudiante_id) REFERENCES estudiante(id),
```

```
    FOREIGN KEY (tutor_id) REFERENCES tutor(id),
    FOREIGN KEY (materia_id) REFERENCES materia(id)
);

-- Índices para optimización
CREATE INDEX idx_sesion_fecha ON sesion_tutoria(fecha_hora);
CREATE INDEX idx_sesion_estado ON sesion_tutoria(estado);
CREATE INDEX idx_usuario_email ON usuario(email);
CREATE INDEX idx_estudiante_codigo ON estudiante(codigo_estudiante);
```

7.3 Procedimientos almacenados, vistas y triggers

Procedimientos Almacenados:

```
-- SP para agendar sesión con validaciones
DELIMITER $$
CREATE PROCEDURE sp_agendar_sesion(
    IN p_estudiante_id INT,
    IN p_tutor_id INT,
    IN p_materia_id INT,
    IN p_fecha_hora DATETIME,
    IN p_duracion INT
)
BEGIN
    DECLARE v_conflict_count INT;
    DECLARE v_tutor_available INT;
```

```
    -- Verificar conflictos de horario
    SELECT COUNT(*) INTO v_conflict_count
    FROM sesion_tutoria
    WHERE tutor_id = p_tutor_id
    AND estado IN ('programada', 'en_curso')
    AND fecha_hora = p_fecha_hora;

    -- Verificar disponibilidad del tutor
    SELECT COUNT(*) INTO v_tutor_available
    FROM disponibilidad
    WHERE tutor_id = p_tutor_id
    AND DAYNAME(p_fecha_hora) = dia_semana
    AND TIME(p_fecha_hora) BETWEEN hora_inicio AND hora_fin;

    IF v_conflict_count = 0 AND v_tutor_available > 0 THEN
        INSERT INTO sesion_tutoria (estudiante_id, tutor_id, materia_id, fecha_hora,
duracion_minutos)
        VALUES (p_estudiante_id, p_tutor_id, p_materia_id, p_fecha_hora, p_duracion);

        SELECT LAST_INSERT_ID() as sesion_id, 'success' as status;
    ELSE
        SELECT NULL as sesion_id, 'conflict' as status;
    END IF;
END$$
DELIMITER ;

-- SP para generar reporte de estudiantes en riesgo
```

```
DELIMITER $$
CREATE PROCEDURE sp_report_estudiantes_riesgo()
BEGIN
    SELECT
        e.id,
        u.nombre,
        u.apellido,
        e.codigo_estudiante,
        e.promedio_general,
        COUNT(st.id) as sesiones_solicitadas,
        AVG(c.puntuacion) as satisfaccion_promedio
    FROM estudiante e
    JOIN usuario u ON e.usuario_id = u.id
    LEFT JOIN sesion_tutoria st ON e.id = st.estudiante_id
    LEFT JOIN calificacion c ON st.id = c.sesion_id
    WHERE e.estado_academico = 'riesgo'
    GROUP BY e.id
    ORDER BY e.promedio_general ASC;
END$$
DELIMITER ;
```

Vistas:

```
-- Vista para dashboard de coordinadores
CREATE VIEW vw_dashboard_coordinador AS
SELECT
    COUNT(DISTINCT e.id) as total_estudiantes,
    COUNT(DISTINCT t.id) as total_tutores,
```

```
    COUNT(DISTINCT st.id) as sesiones_mes,
    AVG(c.puntuacion) as satisfaccion_promedio,
    COUNT(DISTINCT CASE WHEN e.estado_academico = 'riesgo' THEN e.id END) as
estudiantes_riesgo
FROM estudiante e
CROSS JOIN tutor t
CROSS JOIN sesion_tutoria st
LEFT JOIN calificacion c ON st.id = c.sesion_id
WHERE st.fecha_hora >= DATE_SUB(NOW(), INTERVAL 1 MONTH);


-- Vista para disponibilidad de tutores
CREATE VIEW vw_tutores_disponibles AS
SELECT
    t.id as tutor_id,
    u.nombre,
    u.apellido,
    m.nombre as materia,
    d.dia_semana,
    d.hora_inicio,
    d.hora_fin
FROM tutor t
JOIN usuario u ON t.usuario_id = u.id
JOIN tutor_especialidad te ON t.id = te.tutor_id
JOIN materia m ON te.materia_id = m.id
JOIN disponibilidad d ON t.id = d.tutor_id
WHERE u.estado = 'activo'
AND t.estado_verificacion = 'aprobado';
```

Triggers:

```
-- Trigger para actualizar calificación promedio del tutor
DELIMITER $$
CREATE TRIGGER tr_actualizar_calificacion_tutor
AFTER INSERT ON calificacion
FOR EACH ROW
BEGIN
    UPDATE tutor t
    JOIN sesion_tutoria st ON t.id = st.tutor_id
    SET t.calificacion_promedio = (
        SELECT AVG(puntuacion)
        FROM calificacion c
        JOIN sesion_tutoria s ON c.sesion_id = s.id
        WHERE s.tutor_id = t.id
    ),
    t.sesiones_completadas = (
        SELECT COUNT(*)
        FROM sesion_tutoria s
        WHERE s.tutor_id = t.id
        AND s.estado = 'completada'
    )
    WHERE st.id = NEW.sesion_id;
END$$
DELIMITER ;

-- Trigger para validar horario de sesión
```

```
DELIMITER $$

CREATE TRIGGER tr_validar_horario_sesion

BEFORE INSERT ON sesion_tutoria

FOR EACH ROW

BEGIN

   IF TIME(NEW.fecha_hora) < '07:00:00' OR TIME(NEW.fecha_hora) > '22:00:00'
THEN

      SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Las sesiones solo pueden
programarse entre 7:00 y 22:00 horas';

   END IF;



   IF NEW.duracion_minutos < 30 THEN

      SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'La duración mínima de una
sesión es 30 minutos';

   END IF;

END$$

DELIMITER ;
```

7.5 Seguridad y respaldo

Estrategia de Seguridad:

- Encriptación de contraseñas con bcrypt
- Tokens JWT para autenticación
- Validación de entrada contra SQL injection
- Roles y permisos basados en RBAC (Role-Based Access Control)

Capítulo 8. Diseño Detallado de sistemas en red y móviles

8.1 Modelo de Comunicación

En la plataforma de tutorías, los distintos componentes se comunican entre sí mediante una arquitectura cliente-servidor.
El frontend (React.js) envía solicitudes al backend (Node.js) usando una API REST que intercambia datos en formato JSON. El servidor se conecta a la base de datos MySQL, y además maneja servicios externos como Google Meet para las videollamadas.

Protocolos principales:

- **HTTP/HTTPS:** comunicación segura entre el navegador y el servidor.
- **WebSocket:** para enviar notificaciones en tiempo real a tutores y estudiantes.
- **SMTP:** para correos de confirmación de tutorías.
- **OAuth 2.0:** para conectarse de forma segura con Google Meet.

8.2 Diseño del Sistema Web o Móvil

El sistema está pensado como una web adaptable a móviles, usando diseño *responsive* para ajustarse a distintos tamaños de pantalla.
Los usuarios pueden acceder desde un celular, laptop o PC sin perder funcionalidad.

Tecnologías utilizadas:

- **Frontend:** React.js, TailwindCSS y Redux para manejo de estado.
- **Backend:** Node.js con Express y MySQL.
- **Autenticación:** mediante JWT, asignando roles según el tipo de usuario.

8.3 Gestión de Datos en Red

El sistema maneja datos distribuidos entre el servidor y la nube. Cuando un estudiante agenda una tutoría, la información se guarda en la base de datos central y se sincroniza con las notificaciones en tiempo real. Si hay problemas de conexión, los datos se conservan temporalmente hasta que se restablezca la red.

8.4 Seguridad en Red y Móviles

Para proteger la información de los usuarios se aplican varios mecanismos:

- Cifrado de contraseñas con bcrypt.
- Validación de usuarios con tokens JWT.
- Comunicación protegida con HTTPS y SSL/TLS.
- Control de acceso según rol (estudiante, tutor, coordinador).
- Políticas de respaldo y copias automáticas de la base de datos.

8.5 Justificación Técnica

El diseño detallado de sistemas en red y móviles garantiza que la plataforma de tutorías virtuales funcione de forma segura, confiable y adaptable a los diferentes entornos tecnológicos.

- **Confiabilidad:** El sistema mantiene su funcionamiento incluso ante fallos de red, gracias al uso de peticiones asincrónicas y la gestión de errores en el servidor.
- **Seguridad:** Se aplican protocolos de comunicación seguros (HTTPS) y mecanismos de autenticación mediante JWT, además del cifrado de contraseñas con bcrypt, para proteger los datos de los usuarios.
- **Escalabilidad:** La arquitectura basada en API REST permite que la plataforma crezca, añadiendo más módulos o usuarios sin afectar su rendimiento.
- **Mantenibilidad:** El desarrollo se estructura bajo el patrón MVC (Modelo–Vista–Controlador), que separa la interfaz, la lógica de negocio y la

gestión de datos, facilitando la actualización y mantenimiento del sistema.

- **Extensibilidad:** El diseño modular permite incorporar nuevas funciones, como un sistema de chat, aplicación móvil nativa o integración con herramientas externas (Google Meet, correo institucional, etc.).

CONCLUSIONES Y RECOMENDACIONES

Conclusiones del Equipo

1. La Plataforma de Tutorías Virtuales representa una solución efectiva para democratizar el acceso al apoyo académico, eliminando barreras geográficas y temporales que afectan a estudiantes en riesgo.
2. La aplicación de la metodología SCRUM permitió al equipo trabajar de forma organizada con entregas incrementales, facilitando la adaptación a cambios y la detección temprana de problemas técnicos.
3. El análisis detallado de requerimientos fue fundamental para identificar las necesidades reales de los tres tipos de usuarios (estudiantes, tutores y coordinadores), evitando funcionalidades innecesarias.
4. La integración de tecnologías modernas como React.js, Node.js y APIs de videollamadas garantiza que la plataforma sea escalable y mantenible a largo plazo.
5. El proyecto contribuye directamente al ODS 4 al mejorar la calidad educativa mediante herramientas tecnológicas accesibles que reducen la deserción universitaria.

Lecciones Aprendidas

1. **Comunicación constante:** Las reuniones diarias fueron clave para mantener al equipo sincronizado y resolver impedimentos rápidamente.
2. **Priorización efectiva:** Definir correctamente las historias de usuario de alta prioridad permitió entregar valor desde los primeros sprints.

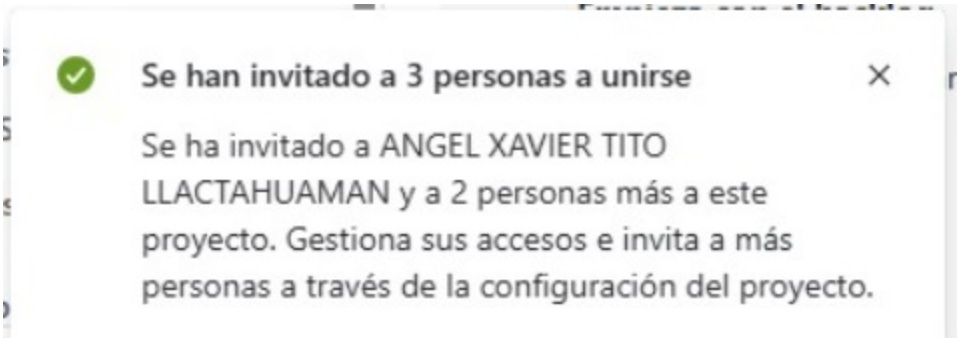
3. **Investigación previa:** Dedicar tiempo a investigar las APIs de videollamadas antes del Sprint 2 evitó bloqueos técnicos durante el desarrollo.
4. **Documentación continua:** Mantener actualizada la documentación en Confluence facilitó la incorporación de nuevos miembros y la consulta de decisiones previas.
5. **Flexibilidad:** Ser capaces de adaptar el plan cuando surgieron imprevistos fue esencial para cumplir los objetivos del proyecto.

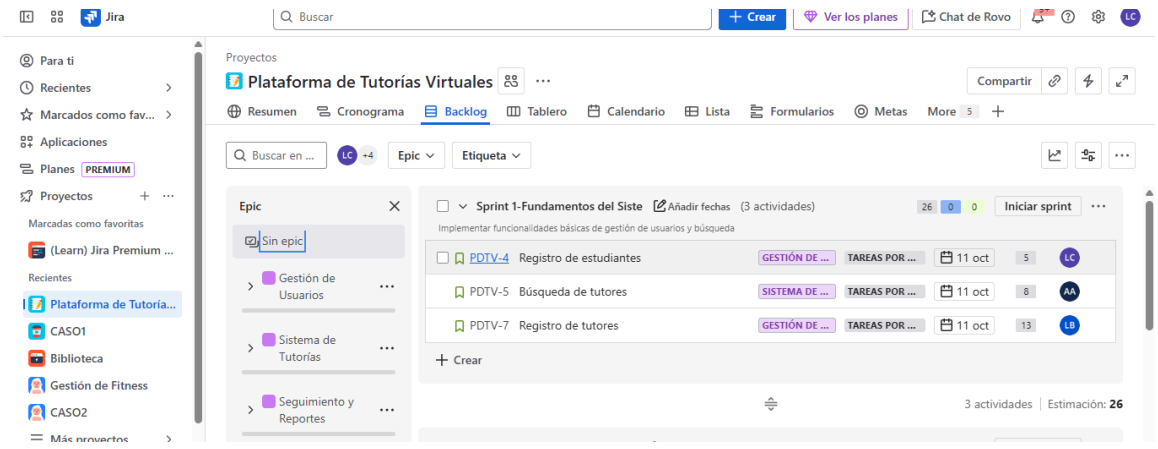
Recomendaciones para Futuras Mejoras del Sistema

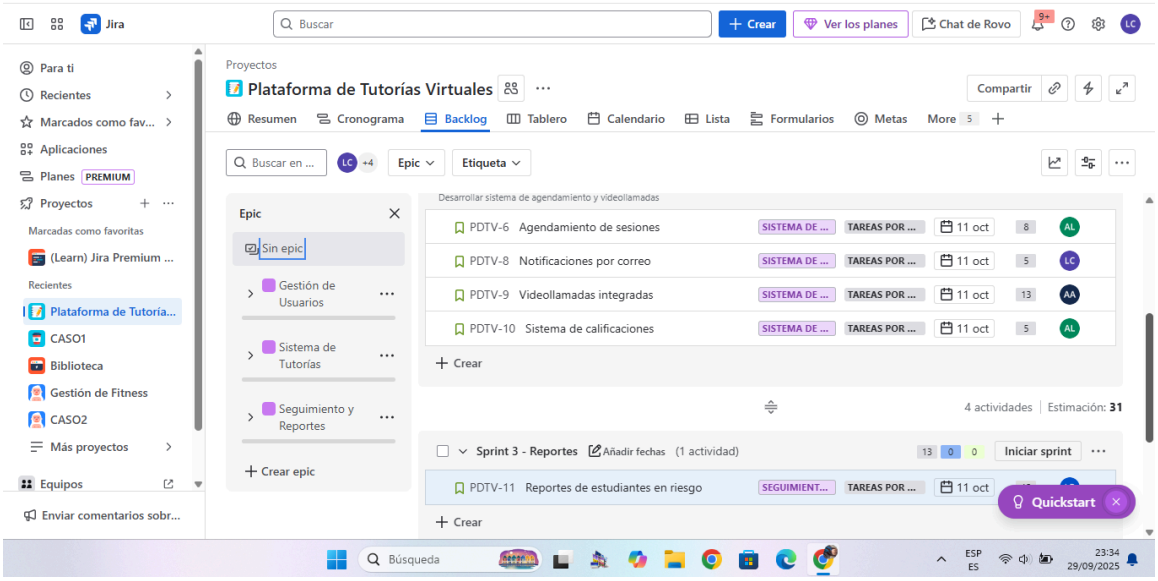
1. **Inteligencia Artificial:** Implementar un sistema de recomendación que sugiera tutores basándose en el historial académico y preferencias del estudiante.
2. **Gamificación:** Agregar insignias y puntos para tutores destacados, incentivando la participación y calidad del servicio.
3. **Aplicación móvil nativa:** Desarrollar versiones para iOS y Android que mejoren la experiencia del usuario en dispositivos móviles.
4. **Sistema de pagos:** Incorporar pasarela de pagos para tutorías privadas fuera del programa institucional, generando un modelo de negocio sostenible.
5. **Analítica avanzada:** Implementar dashboards con métricas predictivas que identifiquen estudiantes en riesgo antes de que reprueben asignaturas.
6. **Integración con LMS:** Conectar la plataforma con sistemas de gestión académica (Canvas, Moodle) para sincronizar automáticamente las notas y detectar necesidades de tutoría.
7. **Grabación de sesiones:** Permitir grabar tutorías (con consentimiento) para que los estudiantes puedan repasar el contenido posteriormente.
8. **Chat en tiempo real:** Agregar mensajería instantánea para consultas rápidas que no requieran una sesión completa de videollamada.

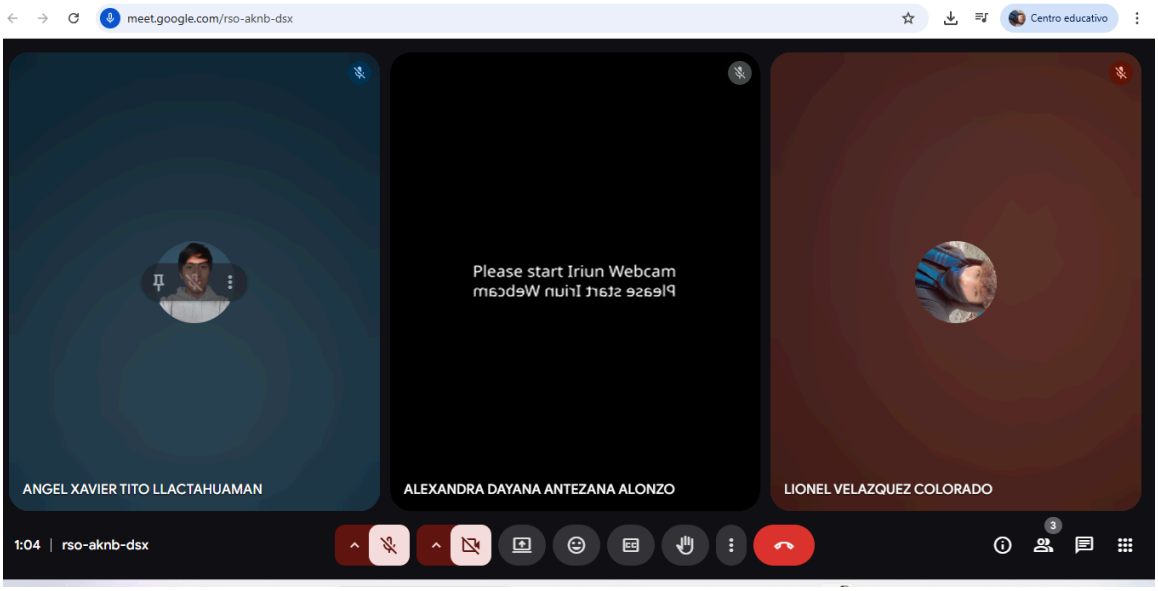
Anexos

Evidencias gráficas (capturas de Jira, capturas de GITHUB y commits, evidencias de trabajo en equipo).

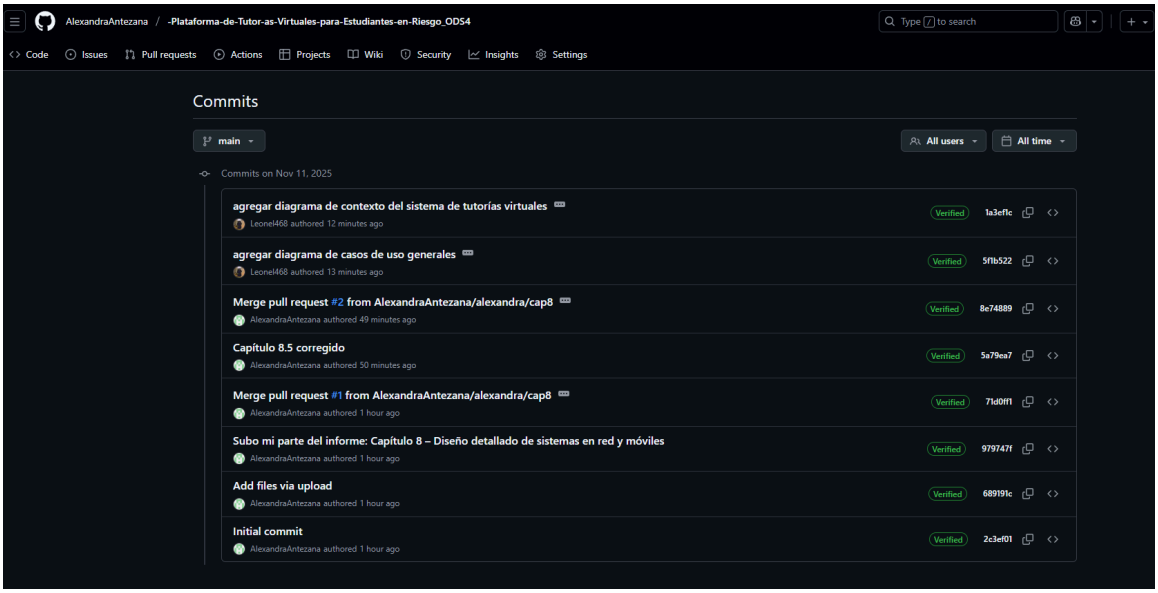

Se han invitado a 3 personas a unirse
Se ha invitado a ANGEL XAVIER TITO LLACTAHUAMAN y a 2 personas más a este proyecto. Gestiona sus accesos e invita a más personas a través de la configuración del proyecto.


Jira
Buscar
Crear
Ver los planes
Chat de Rovo
Para ti
Recientes
Marcados como fav...
Aplicaciones
Planes PREMIUM
Proyectos
Marcadas como favoritas
(Learn) Jira Premium ...
Recientes
Plataforma de Tutoría...
CASO1
Biblioteca
Gestión de Fitness
CASO2
Proyectos
Plataforma de Tutorías Virtuales
Compartir
Resumen
Cronograma
Backlog
Tablero
Calendario
Lista
Formularios
Metas
More
Buscar en ...
Epic
Etiqueta
Sin epic
Gestión de Usuarios
Sistema de Tutorías
Seguimiento y Reportes
Sprint 1-Fundamentos del Siste
Añadir fechas
(3 actividades)
Iniciar sprint
Implementar funcionalidades básicas de gestión de usuarios y búsqueda
PDTV-4 Registro de estudiantes
PDTV-5 Búsqueda de tutores
PDTV-7 Registro de tutores
11 oct
Crear
3 actividades | Estimación: 26


Jira
Buscar
Crear
Ver los planes
Chat de Rovo
Proyectos
Plataforma de Tutorías Virtuales
Compartir
Resumen
Cronograma
Backlog
Tablero
Calendario
Lista
Formularios
Metas
More
Epic
Etiqueta
Sin epic
Gestión de Usuarios
Sistema de Tutorías
Seguimiento y Reportes
Crear epic
Desarrollar sistema de agendamiento y videollamadas
PDTV-6 Agendamiento de sesiones
PDTV-8 Notificaciones por correo
PDTV-9 Videollamadas integradas
PDTV-10 Sistema de calificaciones
11 oct
Crear
4 actividades | Estimación: 31
Sprint 3 - Reportes
Añadir fechas
(1 actividad)
Iniciar sprint
PDTV-11 Reportes de estudiantes en riesgo
Quickstart
Equipos
Enviar comentarios sobr...
Búsqueda
ESP ES
23:34
29/09/2025



commits:



Referencias bibliográficas (ISO 690 numérico).

SACHS, J. D. La era del desarrollo sostenible. Barcelona: Deusto, 2015. 576 pp. ISBN: 9788423419939.

UNESCO. Informe de seguimiento de la educación en el mundo 2020: Inclusión y educación: todos sin excepción. París: UNESCO, 2020. 435 pp. ISBN: 9789233000885.

HARGREAVES, A. y FULLAN, M. Capital profesional: Transformar la enseñanza en cada escuela. Madrid: Morata, 2014. 256 pp. ISBN: 9788471128049.

TOMLINSON, C. A. El aula diversificada: Dar respuesta a las necesidades de todos los estudiantes. Barcelona: Octaedro, 2015. 216 pp. ISBN: 9788499216960.