



YAMIX

Manual Técnico

Juan Camilo Arbelaez Diaz
Juan Esteban Betancur Hoyos
Christian Mathiws Torres Serna

Tecnología en análisis y desarrollo de software, centro de servicios y gestión empresarial

10 de diciembre del 2024



Tabla de contenidos

Tabla de contenidos	2
Introducción	3
1. Requerimientos del Sistema	4
1.1 Requerimientos de Hardware	4
1.2 Requerimientos de Software	4
2. Técnicas de recolección de datos	5
3. Ficha de proyecto	6
3.1 Planteamiento del problema	6
3.2 Justificación	7
3.3 Objetivos	7
3.3.1 General:	8
3.3.2 Específicos:	8
3.4 Alcance del proyecto	8
4. Mapa de procesos	10
5. Facilitación gráfica	11
6. Story mapping	12
7. Wireframe (Figma)	13
8. Matriz historia de usuarios	19
9. Product Backlog Priorizado	20
10. Plataforma de desarrollo	21
10.1 Sistema operativo	21
10.2 Diagrama de Despliegue de Hardware	22
11. Diagrama UML	23
11.1 Diagramas de casos de uso	23
11.2 Documentación de casos de uso.	26
11.3 Diagrama de clases	36
11.4 Diagrama de componentes	37
11.5 Diagrama de despliegue	38
12. Prototipo	39
13. Modelo de la base de datos	44
13.1 Modelo Relacional	44
13.2 Modelo Físico (Script)	
-	44
13.3 Diccionario de datos	74
14. Seguridad	76
15. Consideraciones especiales para la configuración	78
16. Migración	82
17. Backups	83



Introducción

Yamix: Club de Artes Marciales y Más en Copacabana, Antioquia

Yamix es un destacado club dedicado a la práctica de artes marciales mixtas, defensa personal, parkour y boxeo, ubicado en el municipio de Copacabana, Antioquia. Bajo la dirección de Yefferson Hernández Muñoz, el club opera principalmente en la sede Deportiva IDEM, ofreciendo a sus miembros un espacio para entrenar y desarrollar habilidades físicas y de autodefensa.

Actualmente, Yamix gestiona sus clases e inscripciones mediante WhatsApp, un sistema que permite a los interesados consultar horarios y decidir sobre su inscripción de manera directa. Sin embargo, este método presenta desafíos en la organización de horarios, manejo de clases y gestión de inscripciones, lo que motiva la búsqueda de soluciones más eficientes para optimizar sus procesos administrativos.



1. Requerimientos del Sistema

1.1 Requerimientos de Hardware

A fin de garantizar un rendimiento óptimo y garantizar una experiencia de usuario excelente, se establecen los siguientes requerimientos mínimos de hardware para el sistema **YAMIX** (Estos datos son estimados y podrían variar dependiendo de la carga del navegador o sistema operativo).

- **Procesador:** Procesador Intel Core i3 o procesador AMD Ryzen 3 (o superiores).
- **Memoria RAM:** 2GB o superior.
- **Almacenamiento:** HDD/SSD/M.2 de 100 GB o superior.
- **Periféricos:** Mouse, teclado y pantalla.

1.2 Requerimientos de Software

Para asegurar un funcionamiento adecuado y la plena utilización de todas las características de **YAMIX**, se detallan a continuación los requisitos de software necesarios:

- **Sistema operativo:** Windows (32/64 bits), MacOs, Linux (cualquier distribución con soporte de navegador web el cual soporte HTML 5).
- **Navegador web:** Cualquier navegador web el cual soporte Html5.
- **Conexión a internet:** Se requiere una conexión a internet permanente para hacer uso de las funciones en línea del software **YAMIX**.



2. Técnicas de recolección de datos

técnicas de recolección de información que utilizamos con nuestra cliente fue la **mesa redonda**, la cual se llevó a cabo con el objetivo principal de intercambiar ideas y opiniones sobre los **requisitos funcionales y no funcionales** necesarios para el desarrollo del aplicativo web/móvil. Esta mesa redonda se realizó el **17 de marzo de 2024** y permitió a todos los participantes exponer sus puntos de vista de manera abierta y colaborativa, fomentando una conversación enriquecedora. Durante esta sesión, se abordaron aspectos clave relacionados con las funcionalidades esenciales del sistema, así como los requisitos no funcionales, tales como el rendimiento, la seguridad y la usabilidad. La diversidad de opiniones contribuyó a un análisis más profundo y detallado de las necesidades del proyecto, lo que ayudó a establecer una visión compartida entre todas las partes involucradas.

Además, realizamos una **lluvia de ideas** con nuestra cliente el **20 de septiembre de 2024**, en la cual cada participante tuvo la oportunidad de proponer nuevas ideas que pudieran contribuir a mejorar el funcionamiento del aplicativo. Esta técnica permitió una generación rápida de ideas innovadoras, que se centraron en optimizar tanto el diseño como la experiencia del usuario. Durante la sesión, surgieron propuestas relacionadas con la interfaz, la navegación, la integración de funcionalidades adicionales y posibles mejoras en la velocidad y rendimiento del sistema. La lluvia de ideas fue fundamental para identificar oportunidades de mejora y generar soluciones creativas que podrían implementarse en la versión final del aplicativo.



3. Ficha de proyecto

Yamix es un club de artes marciales mixtas, defensa personal, parkour y boxeo ubicado en Copacabana, Antioquia. El club, dirigido por Yefferson Hernández Muñoz, ofrece sus servicios principalmente en la sede Deportiva IDEM. Actualmente, el club gestiona la asignación de sus clases a través de WhatsApp. Los clientes interesados preguntan sobre las clases disponibles y toman decisiones de inscripción basadas en la información proporcionada. Sin embargo, enfrenta varios problemas para la gestión de los horarios, clases y las inscripciones.

3.1 Planteamiento del problema

Yamix es un club de artes marciales mixtas, defensa personal, parkour y boxeo ubicado en Copacabana, Antioquia. El club, dirigido por Yefferson Hernández Muñoz, ofrece sus servicios principalmente en la sede Deportiva IDEM. Actualmente, el club gestiona la asignación de sus clases a través de WhatsApp. Los clientes interesados preguntan sobre las clases disponibles y toman decisiones de inscripción basadas en la información proporcionada. Sin embargo, enfrenta varios problemas para la gestión de los horarios, clases y las inscripciones. Estos problemas se desglosan de la siguiente manera:

Gestión de Horarios de Clases:

Yamix no cuenta con un sistema de agendamiento de horarios que permita la administración de las clases. Esto genera dificultad en la programación de las clases, ya que las mismas pueden variar de manera constante.

Inscripciones por WhatsApp:A1

Las inscripciones se llevan a cabo exclusivamente a través de WhatsApp. Esto limita la capacidad del club para expandir sus servicios y promocionar sus productos de manera efectiva, ya que carece de una plataforma centralizada para la gestión de inscripciones y la difusión de información relevante, impactando estos problemas en la eficiencia operativa de Yamix.

Sistema de agendamiento de horarios:

La planificación de clases y la satisfacción de los clientes, no cuentan con un sistema de información en línea para poder realizar un registro de las clases y el seguimiento a sus preferencias en cuanto a los servicios registrados buscando mayor visibilidad del mercado a través del mismo.

Sistema de visualización de implementos



los estudiantes enfrentan dificultades sobre dónde adquirir los implementos necesarios para cada disciplina. La falta de orientación y recursos centralizados respecto a proveedores confiables y lugares específicos donde obtener los implementos adecuados puede obstaculizar la participación y el proceso de cada estudiante.

3.2 Justificación

La creación de este aplicativo web- mobile responde a la necesidad crítica de mejorar la gestión y visibilidad del club Yamix ubicado en Copacabana, Antioquia. Actualmente, el club enfrenta limitaciones significativas en la administración de sus procesos y en la interacción con sus clientes.

Mejora de la Visibilidad:

Yamix busca ampliar su presencia y visibilidad en el mercado. La plataforma web propuesta permitirá una exposición más efectiva de sus servicios y productos, facilitando el acceso a información clave para los clientes potenciales.

Eficiencia y Rapidez en los Procesos:

Con la implementación de esta plataforma, los procesos de administración, inscripciones y gestión de estudiantes serán eficientes y rápidos. Esto mejorará la experiencia tanto para el club como para los clientes, reduciendo los tiempos de espera y los esfuerzos administrativos.

Mayor Interacción con el Cliente:

La plataforma brindará una mayor interacción con los clientes, permitiéndoles tomar decisiones informadas sobre las actividades e implementos que ofrece Yamix con terceros. Esto fomentará una relación más cercana y colaborativa entre el club y su base de clientes.

Gestión administrativa:

La plataforma ofrecerá a Yamix una herramienta para administrar sus inscripciones y el calendario de actividades, incluyendo eventos y clases. Esto simplificará la gestión interna y permitirá una planificación más efectiva.

Experiencia del Cliente Mejorada: Los clientes tendrán acceso a una plataforma en línea que les brindará una experiencia más conveniente y enriquecedora. Podrán realizar inscripciones de manera sencilla y acceder a información actualizada sobre las actividades del club, lo que mejorará su satisfacción.

3.3 Objetivos

3.3.1 General:



Desarrollar un aplicativo web-Mobile que le permita al club Yamix a optimizar la gestión de horario de clases, inscripciones, la visualización del catálogo de implementos y control de asistencia de los estudiantes registrados en el aplicativo, con el fin de mejorar la eficiencia operativa y la interacción con los usuarios.

3.3.2 Específicos:

Gestionar la configuración de los permisos asociados a los roles de acuerdo a las políticas del club.

Gestionar la cuenta de los usuarios de acuerdo con la seguridad del aplicativo.

Gestionar el proceso de servicios que gestiona los subprocessos de clases, inscripciones, calendario e implementos.

Gestionar el subprocesso de agendamiento de horarios.

Gestionar el subprocesso de eventos.

Gestión de horarios de clase y asistencias (en caso del maestro) por medio de un aplicativo móvil.

3.4 Alcance del proyecto

Proceso de configuración:

Subproceso de Gestión de Roles:

En este proceso, se definen los roles que estarán disponibles en la plataforma. Los roles determinan las funciones y responsabilidades que tendrán los usuarios.

Subproceso de Gestión de Permisos:

En este proceso, se definen los permisos que estarán disponibles en la plataforma. Los permisos determinan qué acciones o funciones pueden realizar los usuarios en la plataforma.

Proceso de usuarios

Subproceso de Gestión de Usuarios:



Se centra en la administración de los usuarios dentro del sistema de Yamix. Inicia con la creación de nuevas cuentas de usuario y abarca la actualización de la información de cada perfil.

Subproceso de Gestión de Acceso:

Este proceso permite a los administradores y usuarios autorizados gestionar su propia cuenta de usuario, realizar cambios en la información de perfil y configurar preferencias personales.

Proceso de Servicios

Subproceso de clases:

Comprende la gestión de clases y categorización de grupos ofrecidos por la aplicación.

Subproceso de inscripción:

Comprende la gestión de inscripciones realizadas por los estudiantes.

Subproceso de instructores:

Comprende la gestión de instructores.

Subproceso de gestión de calendario:

Comprende la gestión del calendario de actividades, que abarca la programación de clases y eventos.

Subproceso de implementos:

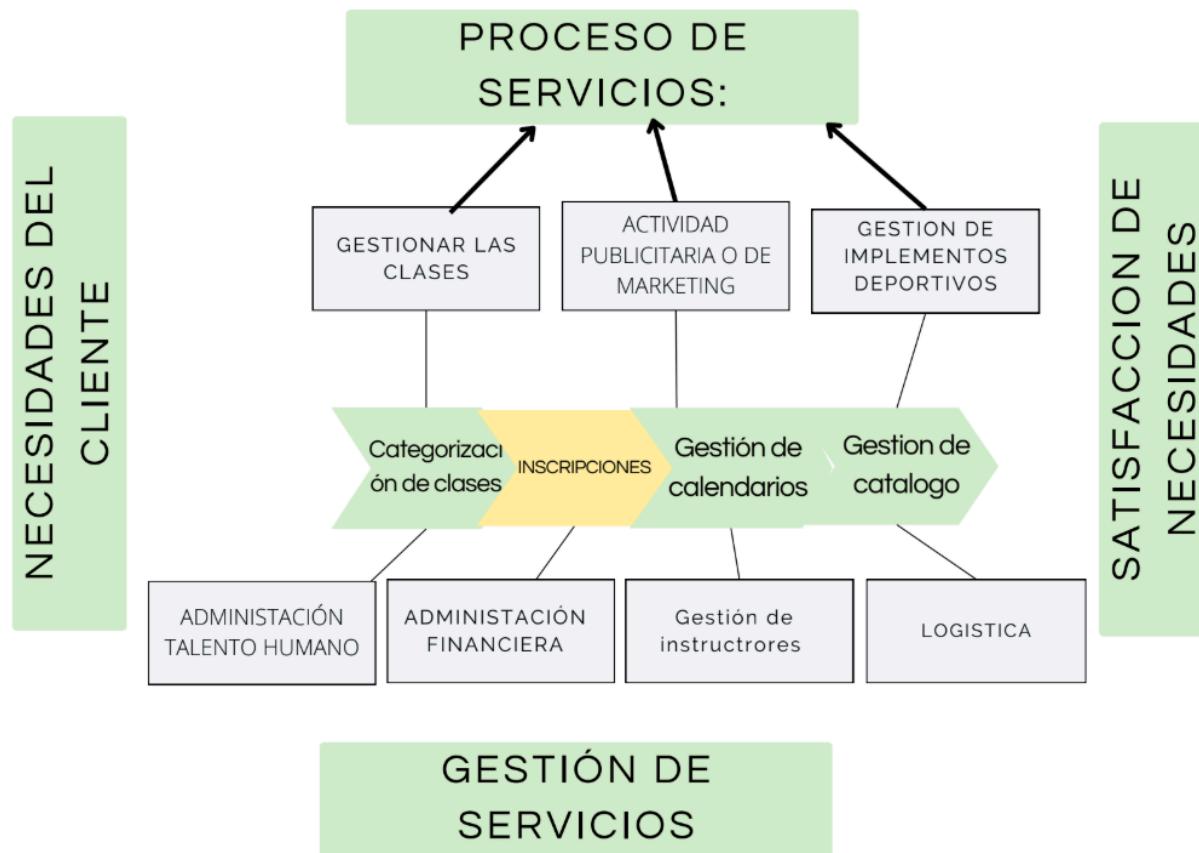
Comprende la información de alternativas de compra de implementos por terceros autorizados.

Proceso desempeño organizacional:

Subproceso estadístico: Comprende los datos estadísticos del aplicativo.

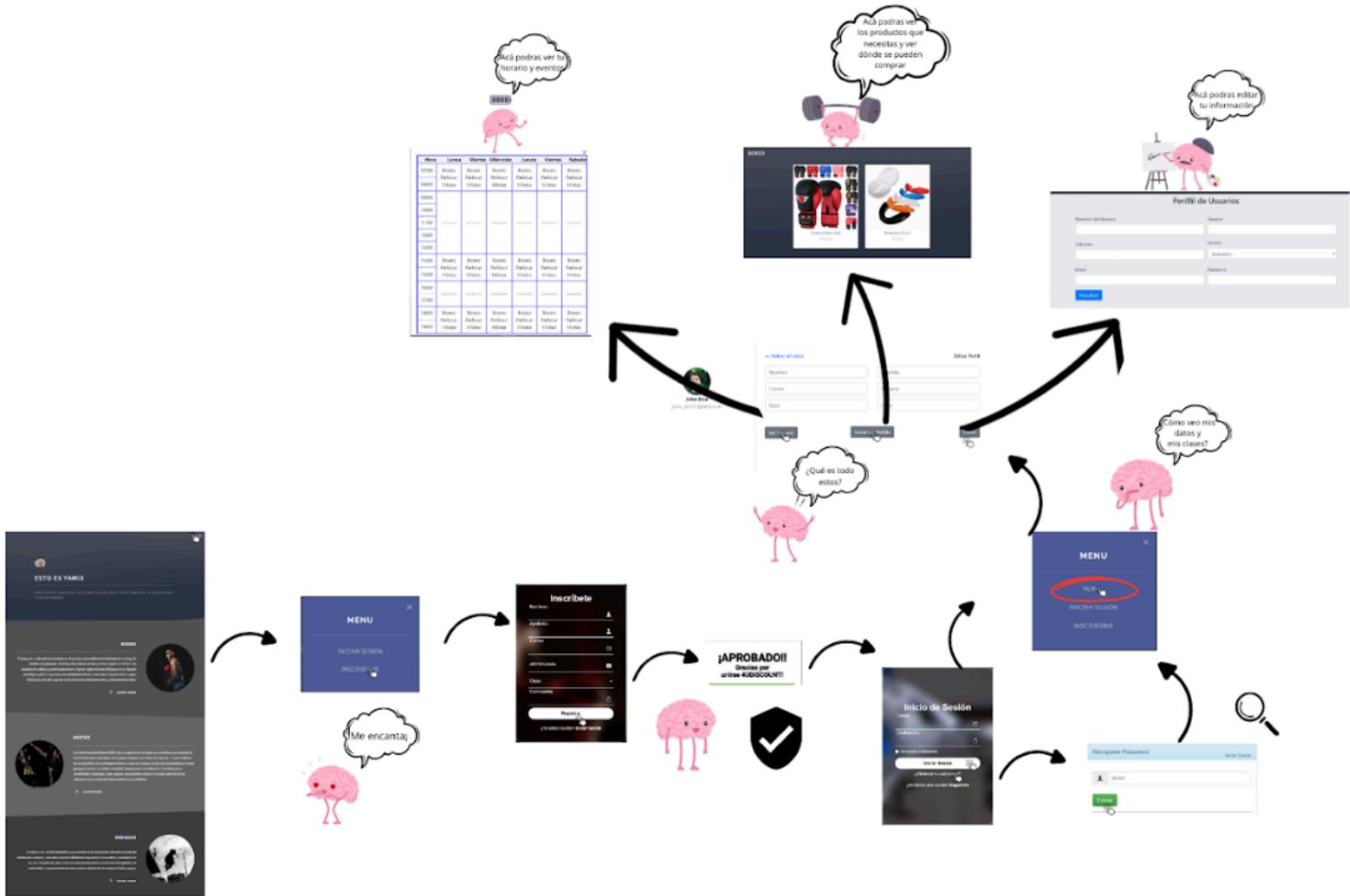
4. Mapa de procesos

Figura 1. Mapa de procesos.



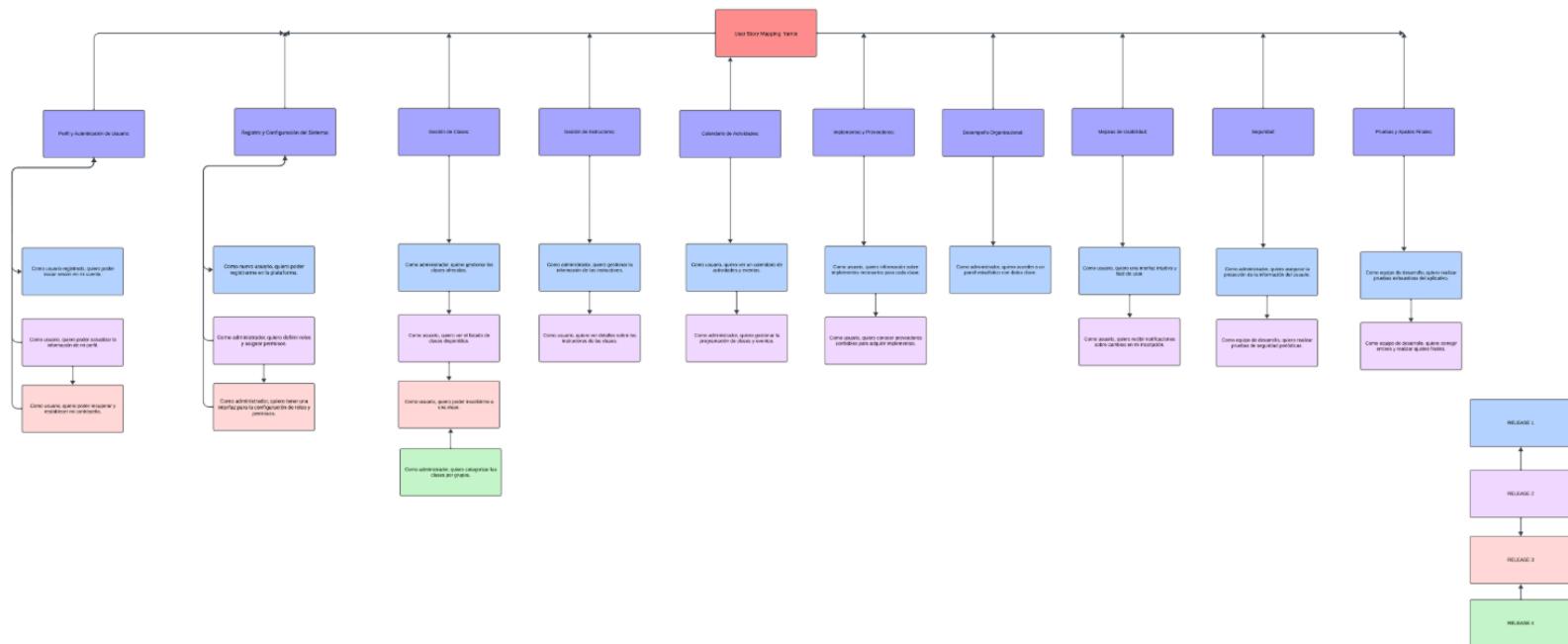
5. Facilitación gráfica

Figura 2. Facilitación gráfica.



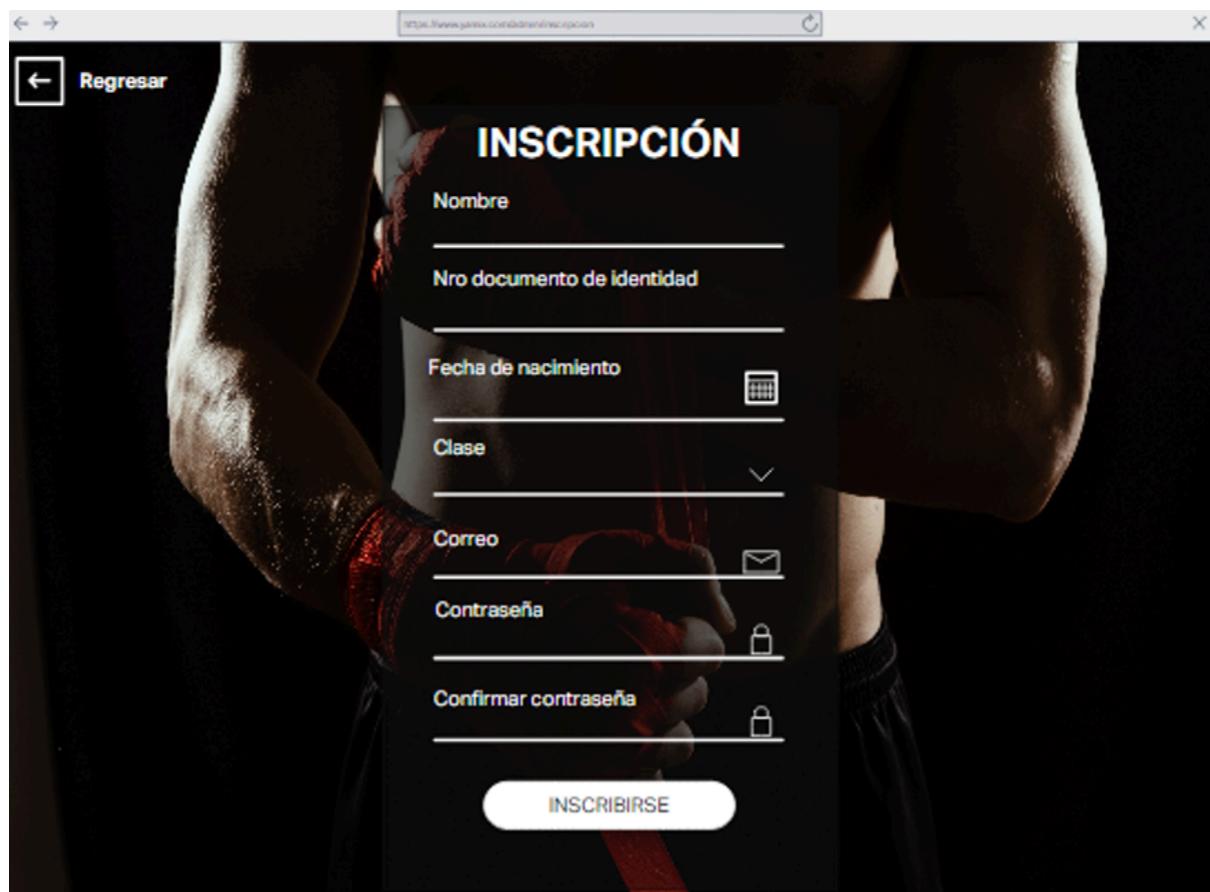
6. Story mapping

Figura 3. Story Mapping



7. Wireframe (Figma)

Figura 4. Wireframe (Figma)



INICIO DE SESIÓN

Correo

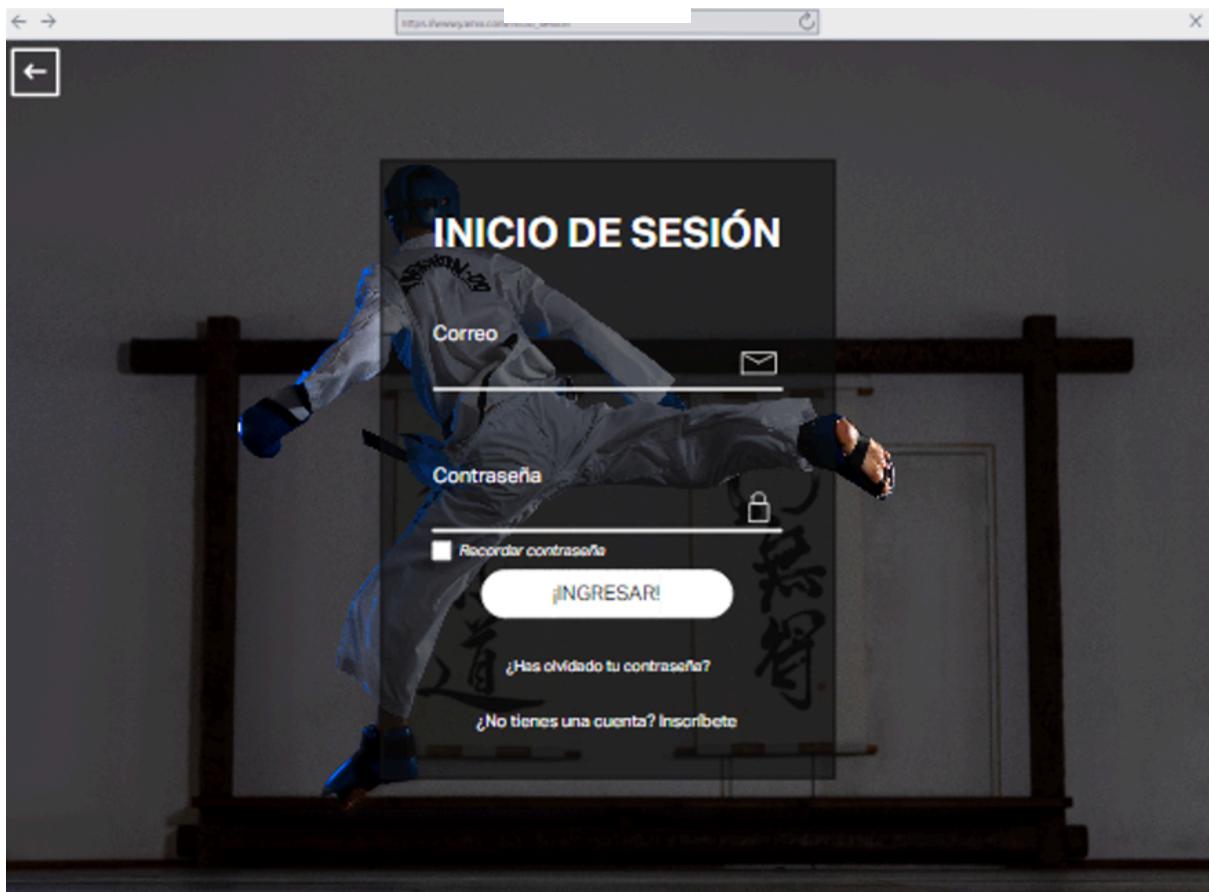
Contraseña 

Recordar contraseña

¡INGRESAR!

[¿Has olvidado tu contraseña?](#)

[¿No tienes una cuenta? Inscríbete](#)



Regresar

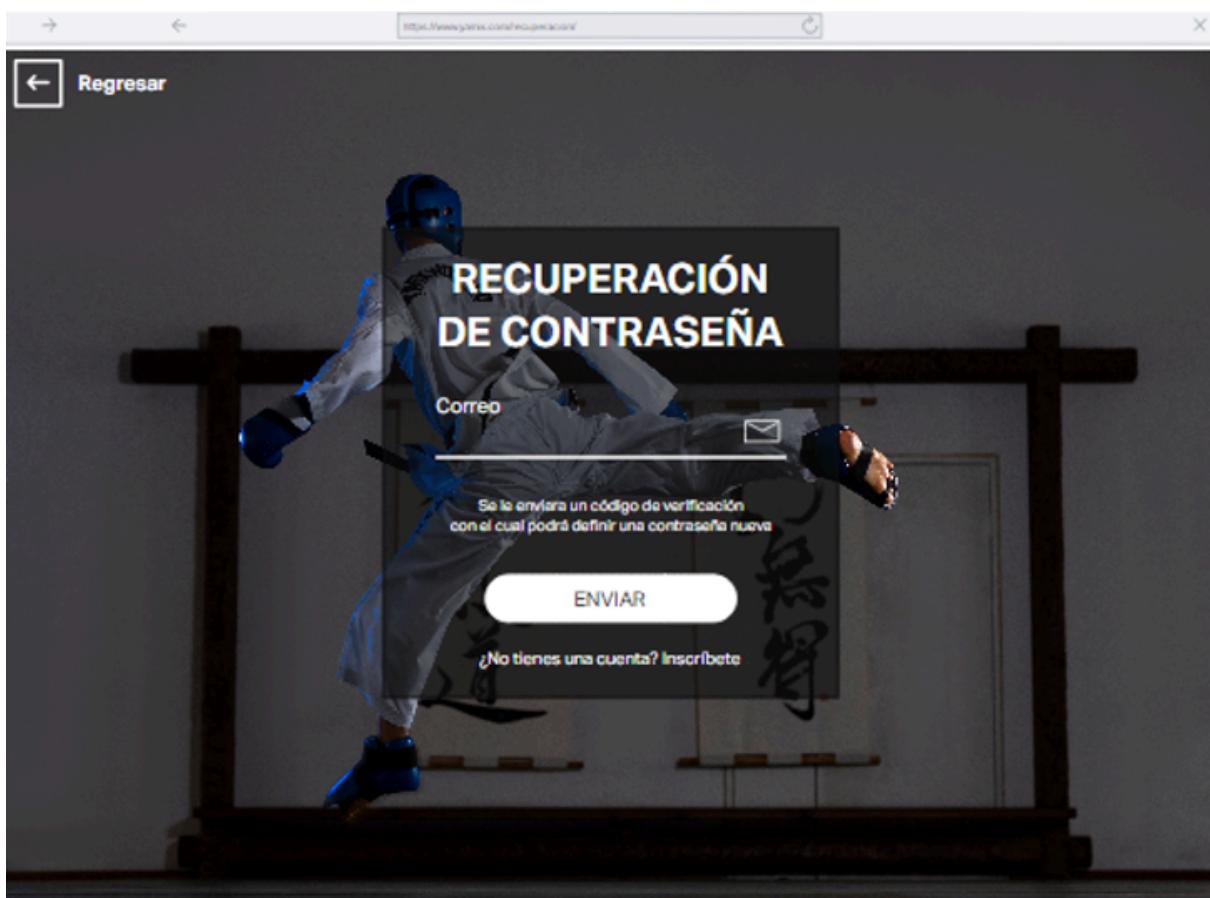
RECUPERACIÓN
DE CONTRASEÑA

Correo

Se le enviará un código de verificación con el cual podrá definir una contraseña nueva

ENVIAR

[¿No tienes una cuenta? Inscríbete](#)



← Regresar

Perfil

Administrador



Nombre: Juan Felipe Moreno
Nro documento de identificación: 123.456.789
Edad: 20
Fecha de nacimiento: 16 de agosto del 2003
Correo: juanfemoreno2002@gmail.com

[EDITAR](#)

[Cambiar Contraseña](#)

Documentación

Documento de identidad escaneado

Certificado EPS

[Nuevo Archivo](#)

[Nuevo Archivo](#)

← Regresar

Perfil

Administrador



Nombre: Juan Felipe Moreno
Nro documento de identificación: 123.456.789
Edad: 20
Fecha de nacimiento: 16 de agosto del 2003
Correo: juanfemoreno2002@gmail.com

[EDITAR](#)

[Cambiar Contraseña](#)

Documentación

Documento de identidad escaneado

Certificado EPS

[Nuevo Archivo](#)

[Nuevo Archivo](#)

Yamix
Administrador

Usuarios
 Maestros
 Estudiantes

Clases
 Artes Mixtas
 Boxeo
 Parkour

Eventos

Implementos

Inscripciones

Informes

Usuarios

Usuarios Yamix

Busqueda rápida

Aprendices

Maestros



Yamix
Administrador

Usuarios
 Maestros
 Estudiantes

Clases
 Artes Mixtas
 Boxeo
 Parkour

Eventos

Implementos

Inscripciones

Informes

Clases

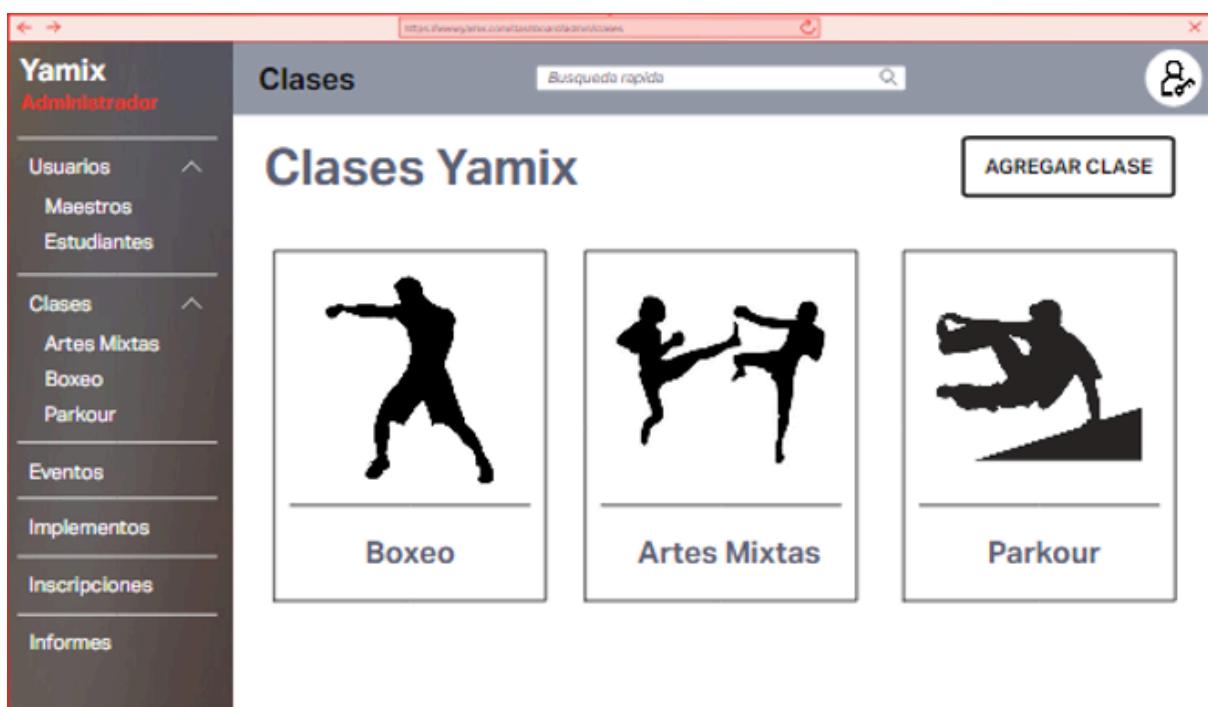
Clases Yamix

AGREGAR CLASE

Boxeo

Artes Mixtas

Parkour



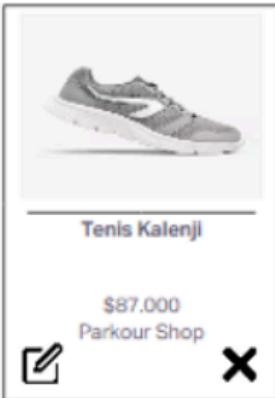
Yamix Maestro

Estudiantes Clase Eventos Implementos Inscripciones Informes

Implementos

Busqueda rápida

Implementos Parkour



Tenis Kalenji

\$87.000 Parkour Shop

AGREGAR IMPLEMENTO

Yamix Maestro

Estudiantes Clase Eventos Implementos Inscripciones Informes

Informes

Busqueda rápida

Informes

INFORMES PASADOS

Informe de pago mensual	Informe de pago anual
 <p>78% pagos 22% no pagos</p> <p>Total en el mes: \$360.000</p> <p>Solicitar informe imprimible: <input type="button" value="PDF"/></p>	 <p>93% pagos 7% no pagos</p> <p>Total en el año: \$4.140.000</p> <p>Solicitar informe imprimible: <input type="button" value="PDF"/></p>

https://www.yamix.com/estadocuentro

Yamix
Maestro

Estudiantes
Clase
Eventos
Implementos
Inscripciones
Informes

Informes

Busqueda rápida

Historial de informes

Año	Creación	Num. informes mensuales	Documentos comprimidos
2022	12/12/2022	11	
2021	21/12/2021	10	
2020	21/12/2020	8	

8. Matriz historia de usuarios

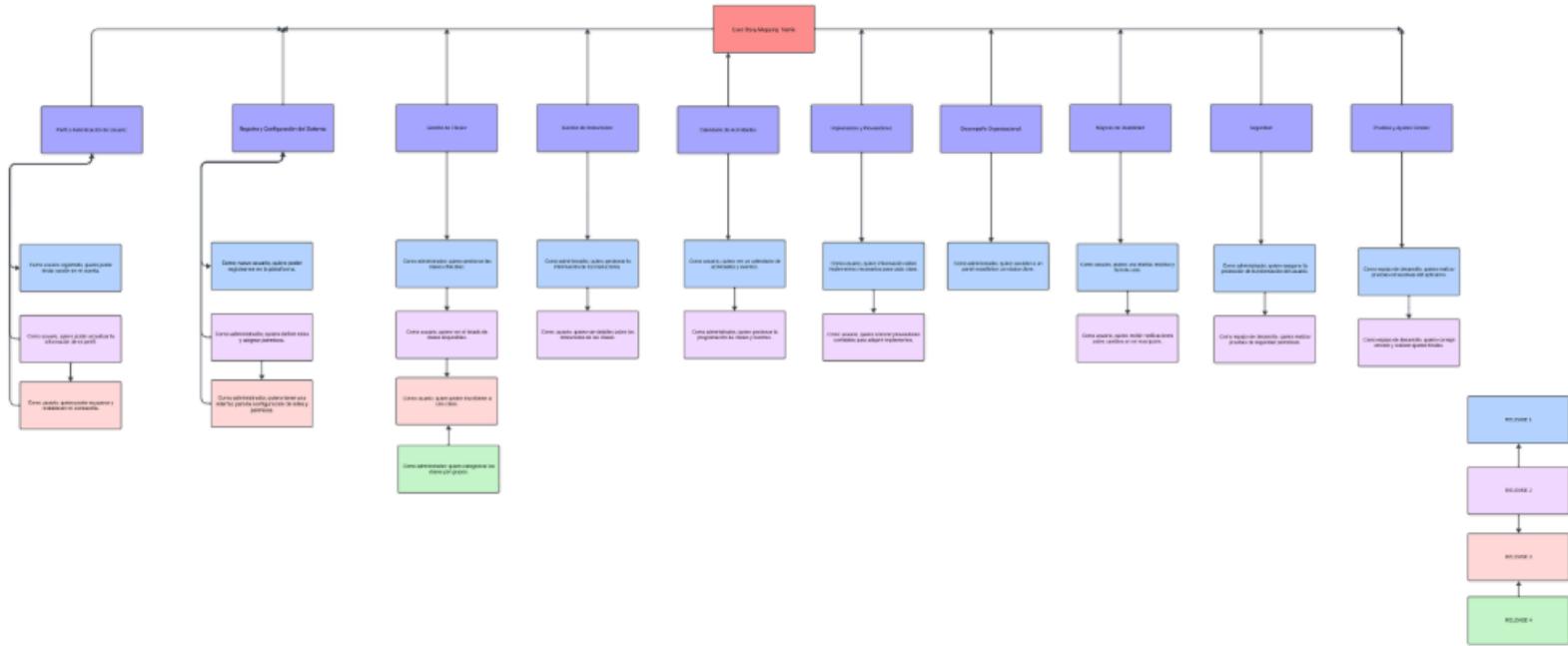
Figura 5. Historias de usuarios.

Proceso	Subproceso	Epica	Código	Título Historia de Usuario	Yo como	ROL DE	Deseo -	OBJETIVO	Para poder	BENEFICIO	Código Criterio	Criterios de Aceptación		
login o Inicio de Sesión	Gestión de Usuarios	Yo como aplicativo necesito ser capaz de gestionar el ingreso de los usuarios por medio de un inicio de sesión	Yo como	Administrador	quiero	poder registrarme en el aplicativo para poder	empezar a gestionar mis servicios a la compañía cuando lo necesite	HU_01	poder registrarme en el aplicativo para poder empezar a gestionar mis servicios a la compañía cuando lo necesite	podr haber un espacio donde solo se podan registrar los administradores	CA_01	Debe haber un espacio donde solo se podan registrar los administradores		
			Yo como	Administrador	quiero	sesión con mi usuario y contraseña	para poder	poder ingresar al aplicativo y gestionar diferentes procesos	HU_02	Cuando el administrador haya agregado el nombre de usuario, la contraseña y la contraseñ La contraseña deberá cumplir con unos requisitos de registro para que la misma sea	CA_02	Cuando el administrador haya agregado el nombre de usuario, la contraseña y la contraseñ		
			Yo como	Usuario Registrado	quiero	mi contraseña cuando	para poder	volver a ingresar al aplicativo sin ningún inconveniente	HU_03	El usuario va a tener botones de opción para ingresar desde gmail o hotmail si el usuario y la contraseña son incorrectas aparecerá un mensaje que dice al	CA_03	La contraseña deberá cumplir con unos requisitos de registro para que la misma sea		
			Yo como	Usuario Registrado	quiero	ir sesión en la	para poder	asegurar que nadie más pueda acceder a mi cuenta	HU_04	Cuando la persona ingresa a la aplicación y los datos sean correctos la aplicación le Deberá ingresar el correo con el cual se registró al aplicativo	CA_04	El usuario va a tener botones de opción para ingresar desde gmail o hotmail		
										cuando olvide la misma, para poder volver a ingresar al aplicativo sin	CA_05	Si el usuario y la contraseña son incorrectas aparecerá un mensaje que dice al		
										recordarla o registrarse de nuevo	CA_06	Cuando la persona ingresa a la aplicación y los datos sean correctos la aplicación le		
										cuando olvide la misma, para poder volver a ingresar al aplicativo sin	CA_07	Deberá ingresar el correo con el cual se registró al aplicativo		
										recordarla o registrarse de nuevo	CA_08	Si los datos que ingresó son correctos y coinciden, se enviará a su correo un pin el		
										cuando olvide la misma, para poder volver a ingresar al aplicativo sin	CA_09	Si los datos que ingresó son incorrectos o no coinciden aparecerá una ventana		
										recordarla o registrarse de nuevo	CA_10	El botón de cerrar sesión debe ser llamativo y fácil de encontrar en la aplicación		
										cuando olvide la misma, para poder volver a ingresar al aplicativo sin	CA_11	Cuando de click en cerrar sesión aparecerá un mensaje que dice: "esta seguro de		
										querer sacar tu sesión	CA_12	Cuando cierra sesión la aplicación te redirigirá a la ventana de inicio de sesión		

<https://docs.google.com/spreadsheets/d/1hqsXhNnDD5DkkqHBBXf2ra04KXdLZZzn/edit?usp=sharing&ouid=114759258205466794658&rtpof=true&sd=true>

9. Product Backlog Priorizado

Figura 6. Product Backlog Priorizado





10. Plataforma de desarrollo

10.1 Sistema operativo

1. Visual Studio Code.

Visual Studio Code es un editor de código fuente desarrollado por Microsoft para Windows, Linux, macOS y Web. Incluye soporte para la depuración, control integrado de Git, resaltado de sintaxis, finalización inteligente de código, fragmentos y refactorización de código

2. XAMPP.

XAMPP es un paquete de software libre, que consiste principalmente en el sistema de gestión de bases de datos MySQL, el servidor web Apache y los intérpretes para lenguajes de script PHP y Perl. El nombre es en realidad un acrónimo: X, Apache, MariaDB/MySQL, PHP, Perl.

3. Postman.

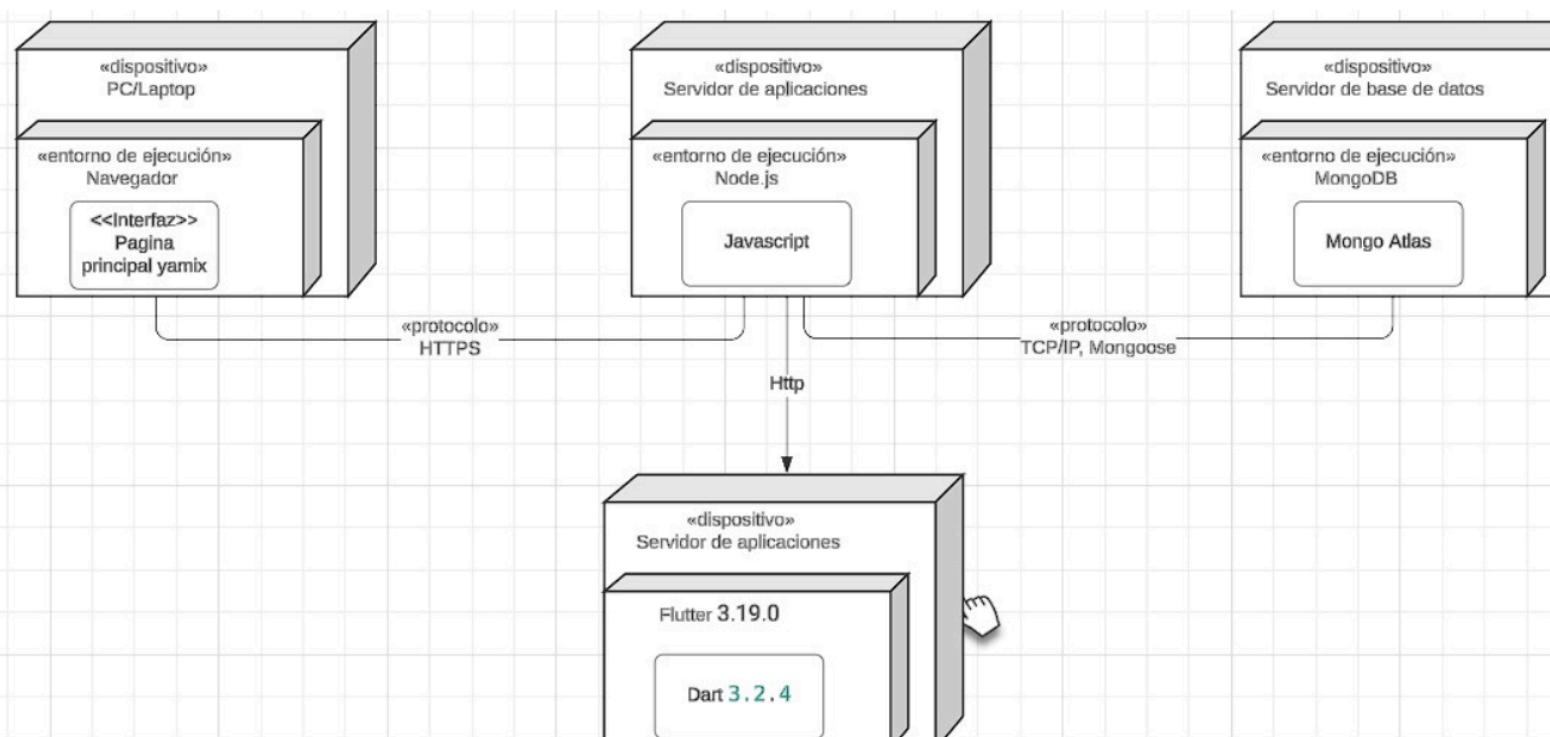
Permite la realización de peticiones a endpoints de backend.

4. Node JS.

Entorno en tiempo de ejecución multiplataforma, de código abierto, para la capa del servidor basado en el lenguaje de programación JavaScript, asíncrono, con E/S de datos en una arquitectura orientada a eventos y basado en el motor V8 de Google.

10.2 Diagrama de Despliegue de Hardware

Figura 7. Diagrama de despliegue



11. Diagrama UML

11.1 Diagramas de casos de uso

Figura 8. Diagramas de casos General

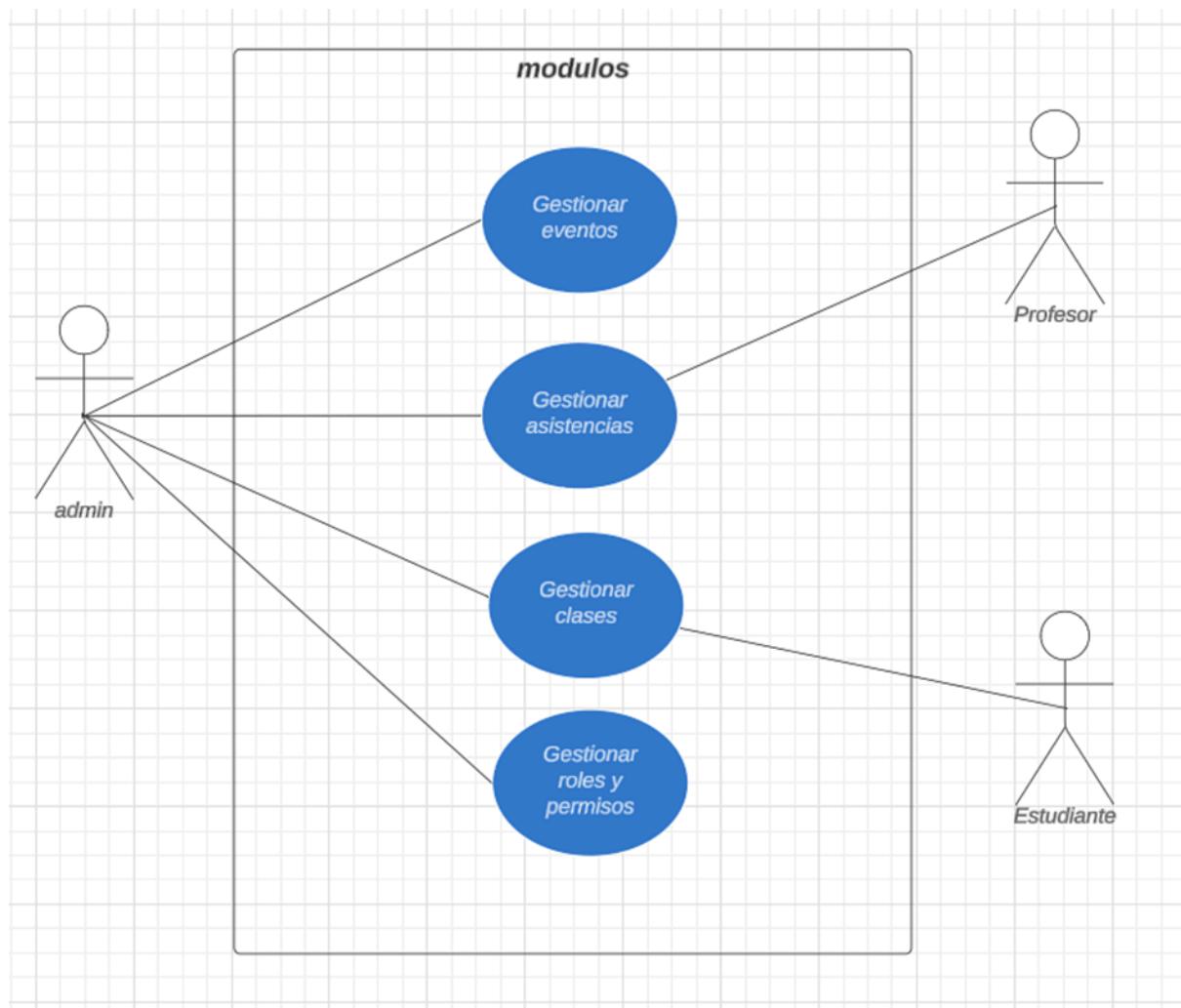
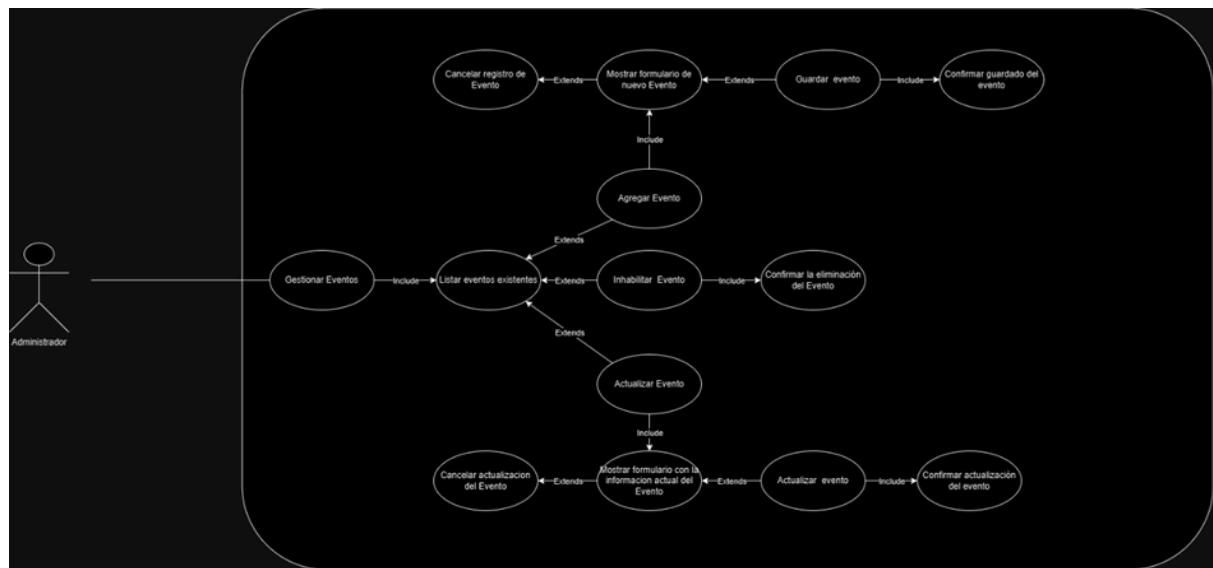
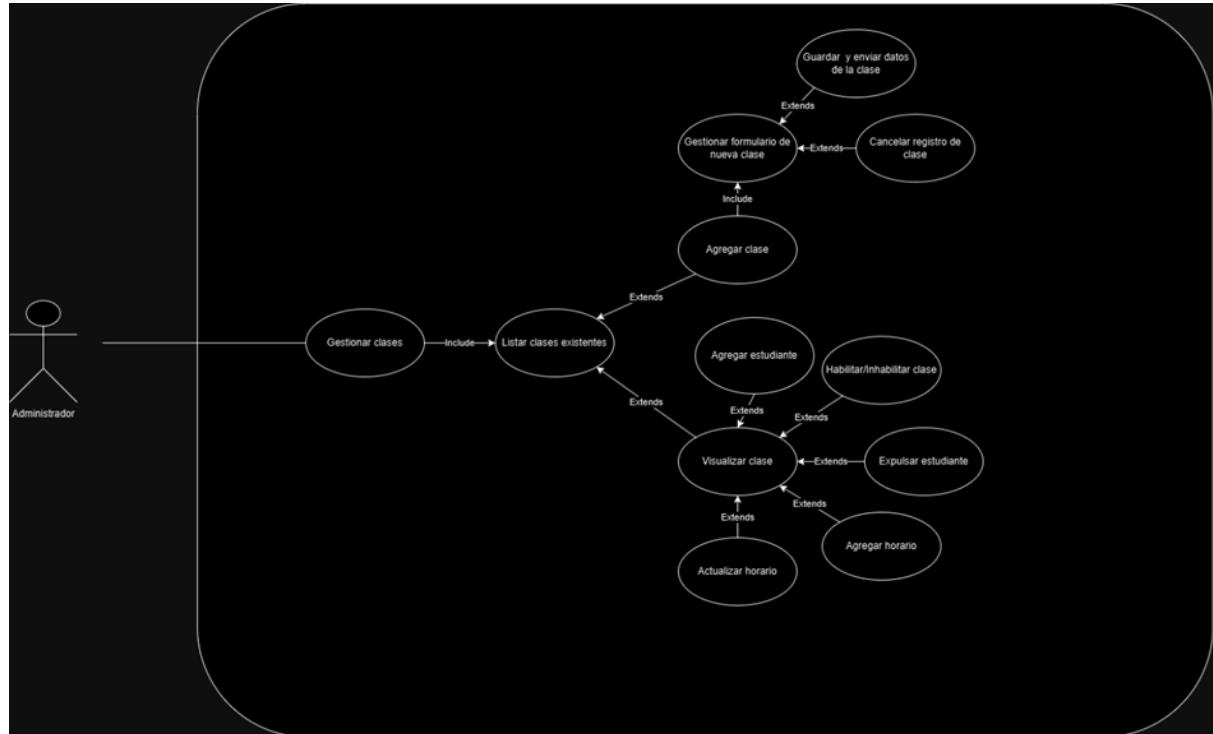
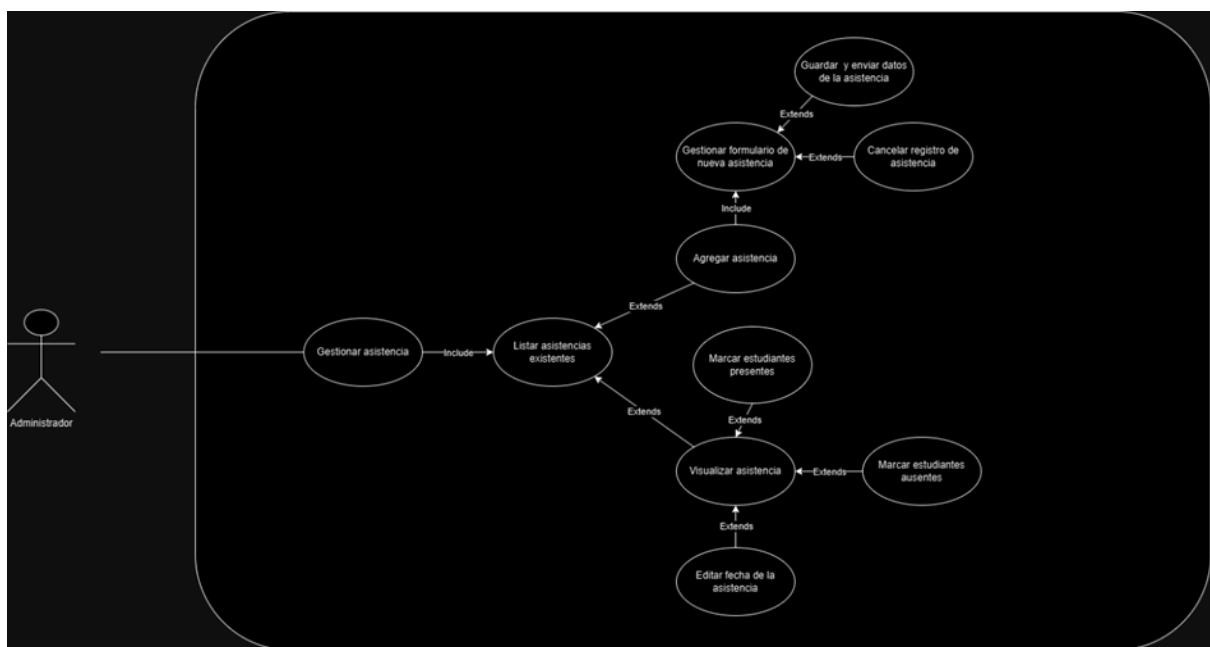
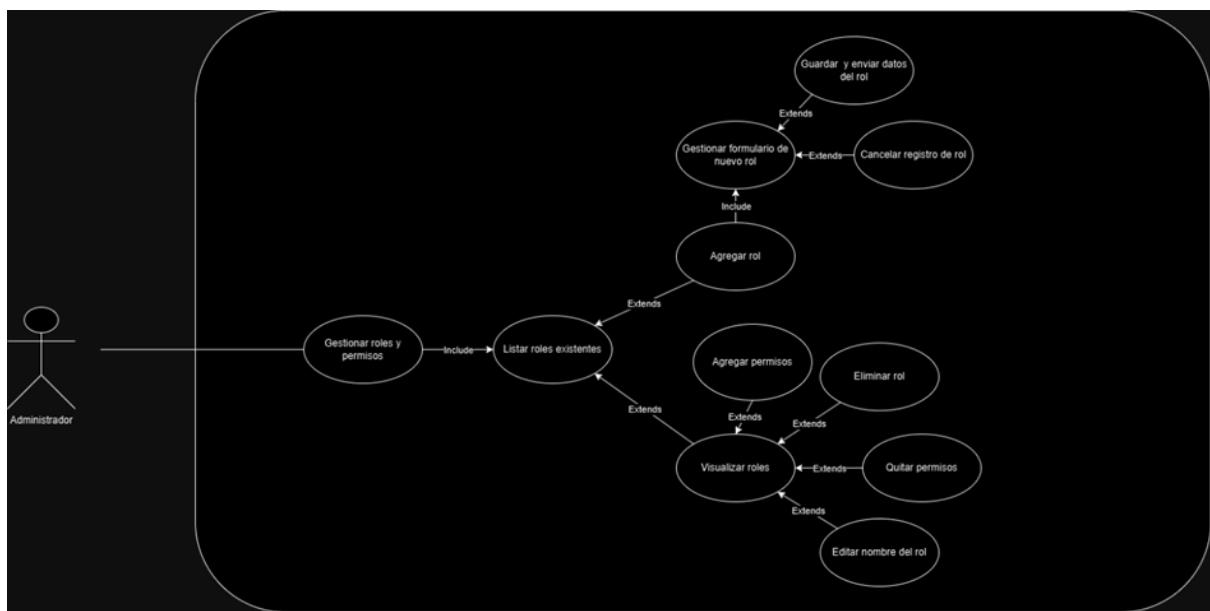


Figura 9. Casos de Usos Específicos.







11.2 Documentación de casos de uso.

Escenario	CU01 Clases
	CU01.1 Gestión de Clases

CU -	CU.01.1.1 Crear una nueva Clase	
Dependencias	CU 01 Clase CU 01.1 Gestión Clase	
Pre – Condiciones	El Usuario debe haber iniciado sesión Acceso a módulo de clases	
Descripción	El sistema deberá comportarse como se describe en el siguiente caso de uso para crear una clase nueva dentro del sistema.	
Actores	Usuario	
Secuencia Normal Crear clase	Paso	Acción
	1	El Usuario ingresa al módulo de clases, y selecciona crear una nueva clase
	2	El sistema muestra formulario de nueva clase

	3	El Usuario diligencia los campos del formulario (curso, hora de inicio, hora final, profesor asignado, estudiantes a inscribir a la clase)
	4	El sistema valida la información del formulario
	5	El sistema crea la nueva clase
	6	El sistema arroja un mensaje al usuario de registro exitoso

Post - Condición	El sistema ha registrado una nueva clase	
Excepciones	Paso	Acción
	3 y 4	3.1. Si alguna casilla del formulario está vacía o tiene información inválida
	1	El sistema devuelve un mensaje al usuario, señalando que campo está vacío o contiene información inválida
	2	El sistema no permitirá crear la clase
	4.1	Si ya hay una clase creada en el mismo rango de tiempo
	1	El sistema devuelve un mensaje al usuario, señalando que ya existe una clase en ese horario

		2	El sistema devuelve al Usuario al paso 2
Documentación Regla de Negocio Normatividad			El sistema podrá crear una nueva clase siempre que esta cumpla con las condiciones establecidas en la academia para evitar sobrecargar el espacio disponible para la formación.
Comentarios			
Responsables			Equipo de desarrollo

CU.01.1.2 Listar Clases	
CU -	
Dependencias	CU 01 Clases CU 01.1 Gestión Clases
Pre – Condiciones	El Usuario debe haber iniciado sesión Acceso a módulo de Clases

Descripción	El sistema deberá comportarse como se describe en el siguiente caso de uso para listar las Clases registrados dentro del sistema.	
Actores	Usuario	
Secuencia Normal Listar Clases	Paso	Acción
	1	El Usuario ingresa al módulo de Clases, y se listarán todas las clases registrados con sus respectivas acciones de visualizar, editar e inhabilitar, además de una barra de búsqueda.
Post - Condición	El sistema ha listado todas las Clases	
Documentación Regla de Negocio Normatividad	El sistema podrá crear una nueva clase siempre que esta cumpla con las condiciones establecidas en la academia para evitar sobrecargar el espacio disponible para la formación.	
Comentarios		
Responsables	Equipo de desarrollo	

CU - CU.01.1.3 Visualizar Estudiantes de una Clase	
Dependencias	CU 01 Clases CU 01.1 Clases
Pre – Condiciones	El Usuario debe haber iniciado sesión Acceso a módulo de clases

Descripción	El sistema deberá comportarse como se describe en el siguiente caso de uso para visualizar una clase dentro del sistema.	
Actores	Usuario	
	Paso	Acción
Secuencia Normal visualizar estudiantes de una clase	1	El Usuario ingresa al módulo de clases, y se listarán todas las clases registradas con sus respectivas acciones de visualizar, editar e inhabilitar, además de una barra de búsqueda.
	2	El Usuario selecciona el botón de visualizar estudiantes
	3	El sistema mostrará todos los estudiantes tanto los que participan de la clase como los que no participan
	4	El sistema permitirá tanto agregar estudiantes de la clase como sacar estudiantes de la clase
Post - Condición	El Usuario ha visualizado los estudiantes de la clase	
Documentación Regla de Negocio Normatividad	El sistema podrá crear una nueva clase siempre que esta cumpla con las condiciones establecidas en la academia para evitar sobrecargar el espacio disponible para la formación.	
Comentarios		
Responsables	Equipo de desarrollo	

CU -	CU.01.1.4 Buscar Clase		
Dependencias	CU 01 Clases CU 01.1 Gestión Clases		
Pre – Condiciones	El Usuario debe haber iniciado sesión Acceso a módulo de clases		
Descripción	El sistema deberá comportarse como se describe en el siguiente caso de uso para buscar clases dentro del sistema.		
Actores	Usuario		
Secuencia Normal Buscar clase	Paso	Acción	
	1	El Usuario ingresa al módulo de clases, y se listarán todas las clases registrados con sus respectivas acciones de visualizar, editar e inhabilitar, además de una barra de búsqueda.	
	2	El Usuario ingresa la información a buscar perteneciente a una clase en la barra de búsqueda.	
	3	El sistema muestra todas las clases que tengan la información buscada, así como sus respectivas acciones de visualizar, editar e inhabilitar.	
Post - Condición	El sistema ha buscado clases		
Excepciones	Paso	Acción	
	1 y 2	1.1. Si no hay clases registradas en el sistema	
		1	El sistema devuelve un mensaje al administrado, indicando que no hay clases registradas en el sistema.

		2.1. Si la información buscada no está asociada a ninguna clase
	1	El sistema devuelve un mensaje al Usuario, indicando que no hay clases registradas en el sistema que tengan esta información
Documentación Regla de Negocio Normatividad	El sistema podrá crear una nueva clase siempre que esta cumpla con las condiciones establecidas en la academia para evitar sobrecargar el espacio disponible para la formación.	
Comentarios		
Responsables	Equipo de desarrollo	

CU - CU.01.1.5 Editar una Clases	
Dependencias	CU 01 Clases CU 01.1 Gestión Clases
Pre – Condiciones	El Usuario debe haber iniciado sesión Acceso a módulo de clases
Descripción	El sistema deberá comportarse como se describe en el siguiente caso de uso para editar una clase dentro del sistema.
Actores	Usuario

Secuencia Normal Editar clase	Paso	Acción
	1	El Usuario ingresa al módulo de clases, y se listarán todas las clases registradas con sus respectivas acciones de visualizar, editar e inhabilitar, además de una barra de búsqueda, y selecciona el botón de editar en la clase deseada.
	2	El sistema muestra formulario de nueva clase con la información actual de esta clase
	3	El Usuario edita los campos del formulario (curso, hora de inicio, hora final, profesor asignado, estudiantes a inscribir a la clase)
	4	El sistema valida la información del formulario
	5	El sistema edita la información de la clase
	6	El sistema arroja un mensaje actualizado al usuario de datos exitoso
Post - Condición El sistema ha editado un paciente		
Excepciones	Paso	Acción
	1, 3, 4	1.1. Si no hay clases registradas en el sistema
		1 El sistema devuelve un mensaje al Usuario, indicando que no hay clases registradas en el sistema.
		3.1. Si alguna casilla del formulario está vacía o tiene información inválida

	1	El sistema devuelve un mensaje al usuario, señalando que campo está vacío o contiene información inválida
	2	El sistema no permite editar la clase
	4.1.	Si el rango de horas ya pertenece a una clase existente
	1	El sistema devuelve un mensaje al usuario, señalando que ya existe una clase con ese horario.
	2	El sistema regresa al Usuario al paso 2
Documentación Regla de Negocio Normatividad	El sistema podrá crear una nueva clase siempre que esta cumpla con las condiciones establecidas en la academia para evitar sobrecargar el espacio disponible para la formación.	
Comentarios		
Responsables	Equipo de desarrollo	

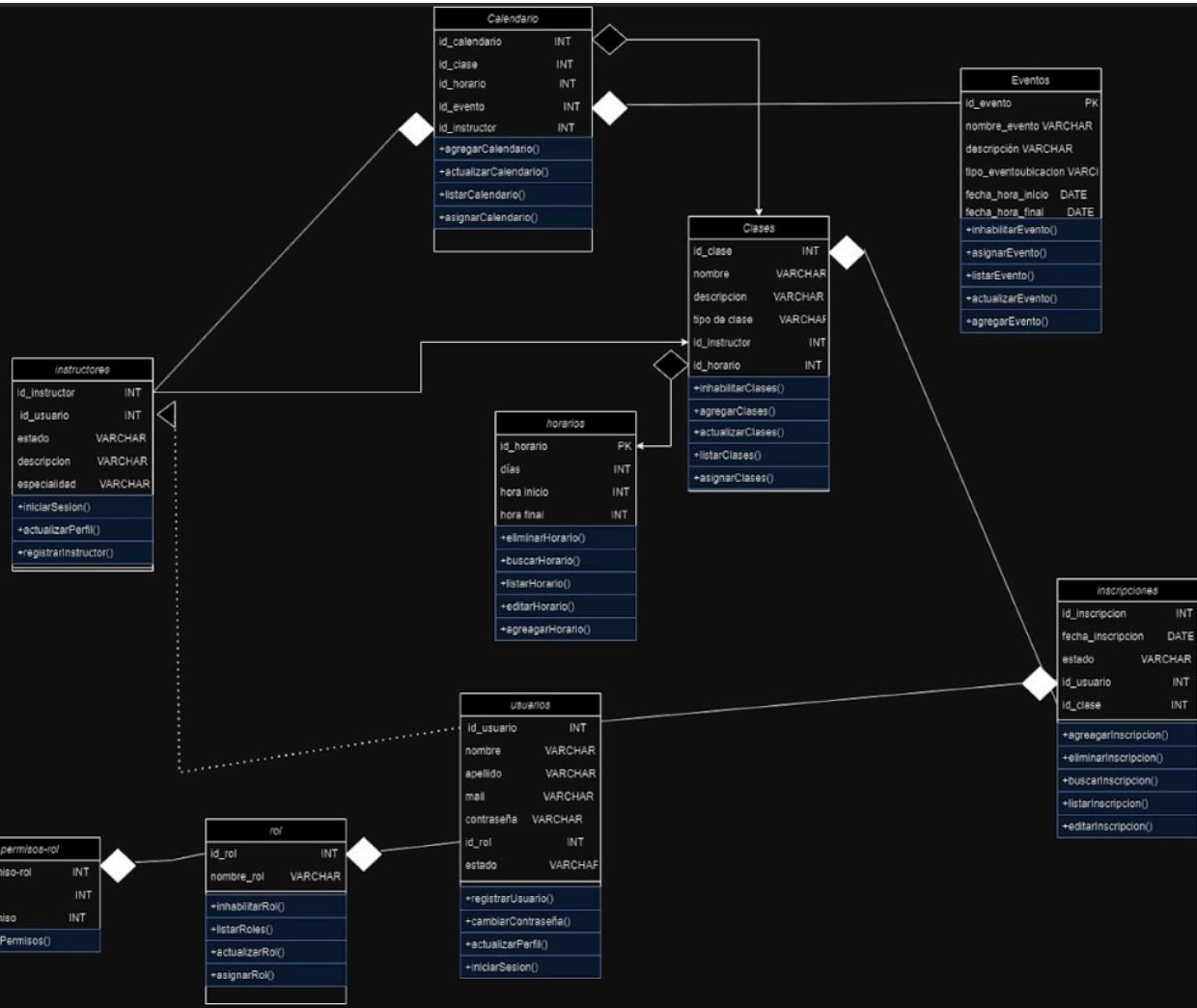
CU - CU.01.1.6 Inhabilitar una Clase

Dependencias	CU 01 Clases CU 01.1 Gestión Clases	
Pre - Condiciones	El Usuario debe haber iniciado sesión Acceso a módulo de clases	
Descripción	El sistema deberá comportarse como se describe en el siguiente caso de uso para eliminar un paciente dentro del sistema.	
Actores	Usuario	
Secuencia Normal Inhabilitar Clase	Paso	Acción
	1	El Usuario ingresa al módulo de clases, y se listarán todos los pacientes registrados con sus respectivas acciones de visualizar, editar e inhabilitar, además de una barra de búsqueda.
	2	El sistema arroja un mensaje al Usuario de habilitación/inhabilitación de clase exitoso
	3	El sistema regresa al Usuario al paso 1
Post - Condición	El sistema ha inhabilitado una clase	
Excepciones	Paso	Acción
	1	1.1. Si no hay clases registradas en el sistema
		1 El sistema devuelve un mensaje al Usuario, indicando que no hay clases registradas en el sistema.

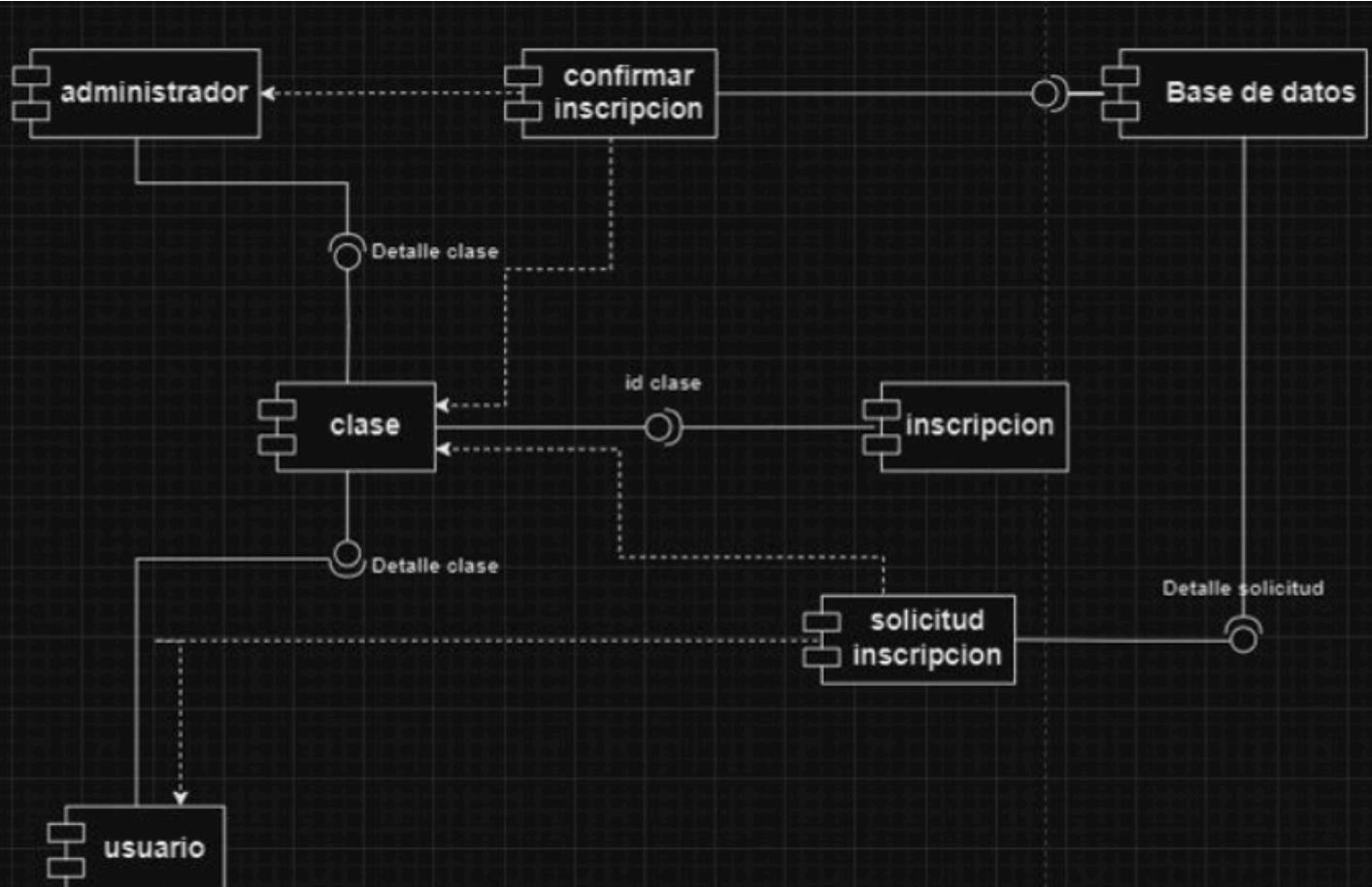
Documentación Regla de Negocio Normatividad	El sistema podrá crear una nueva clase siempre que esta cumpla con las condiciones establecidas en la academia para evitar sobrecargar el espacio disponible para la formación.
Comentarios	
Responsables	Equipo de desarrollo

11.3 Diagrama de clases

Figura 10. Diagrama de clases

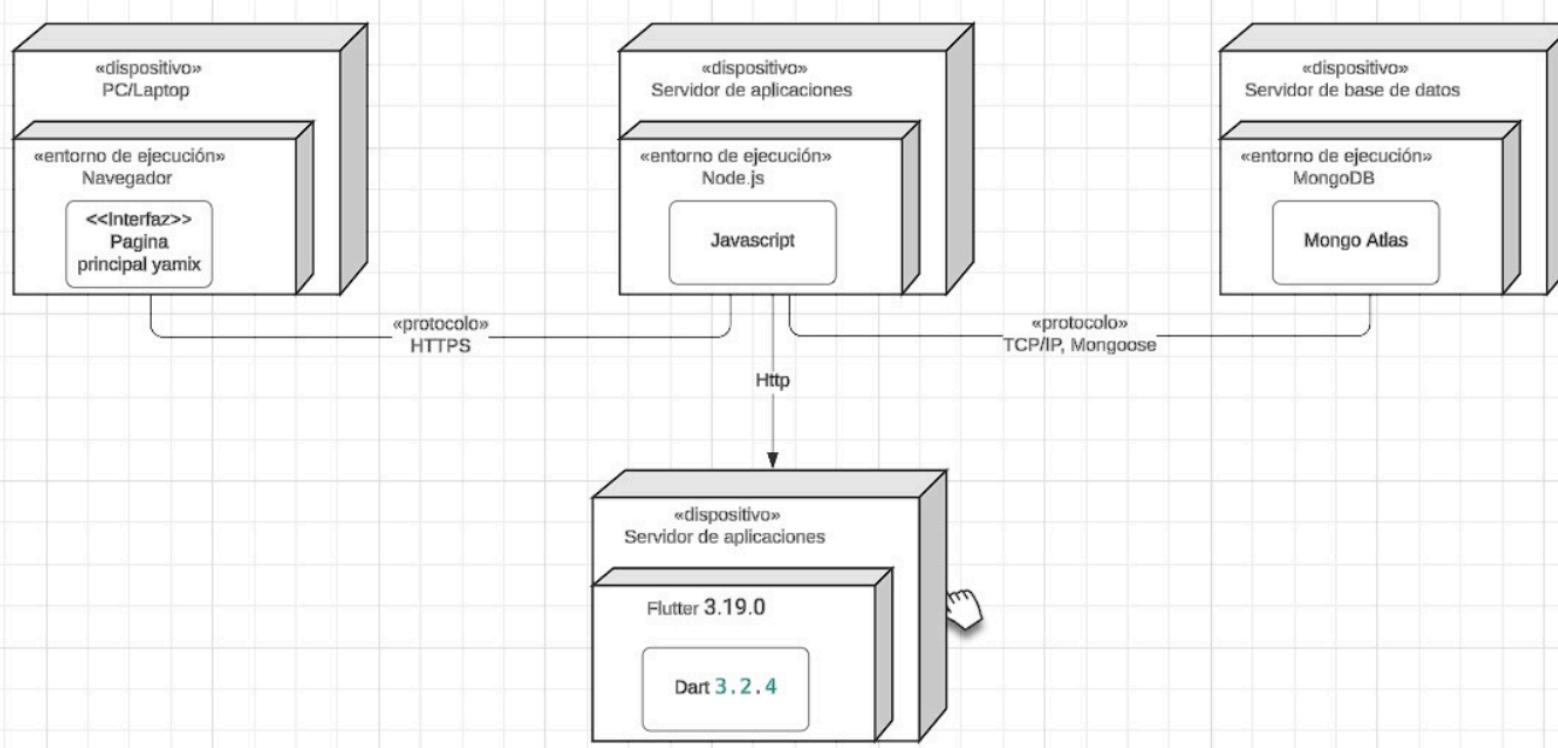


11.4 Diagrama de componentes



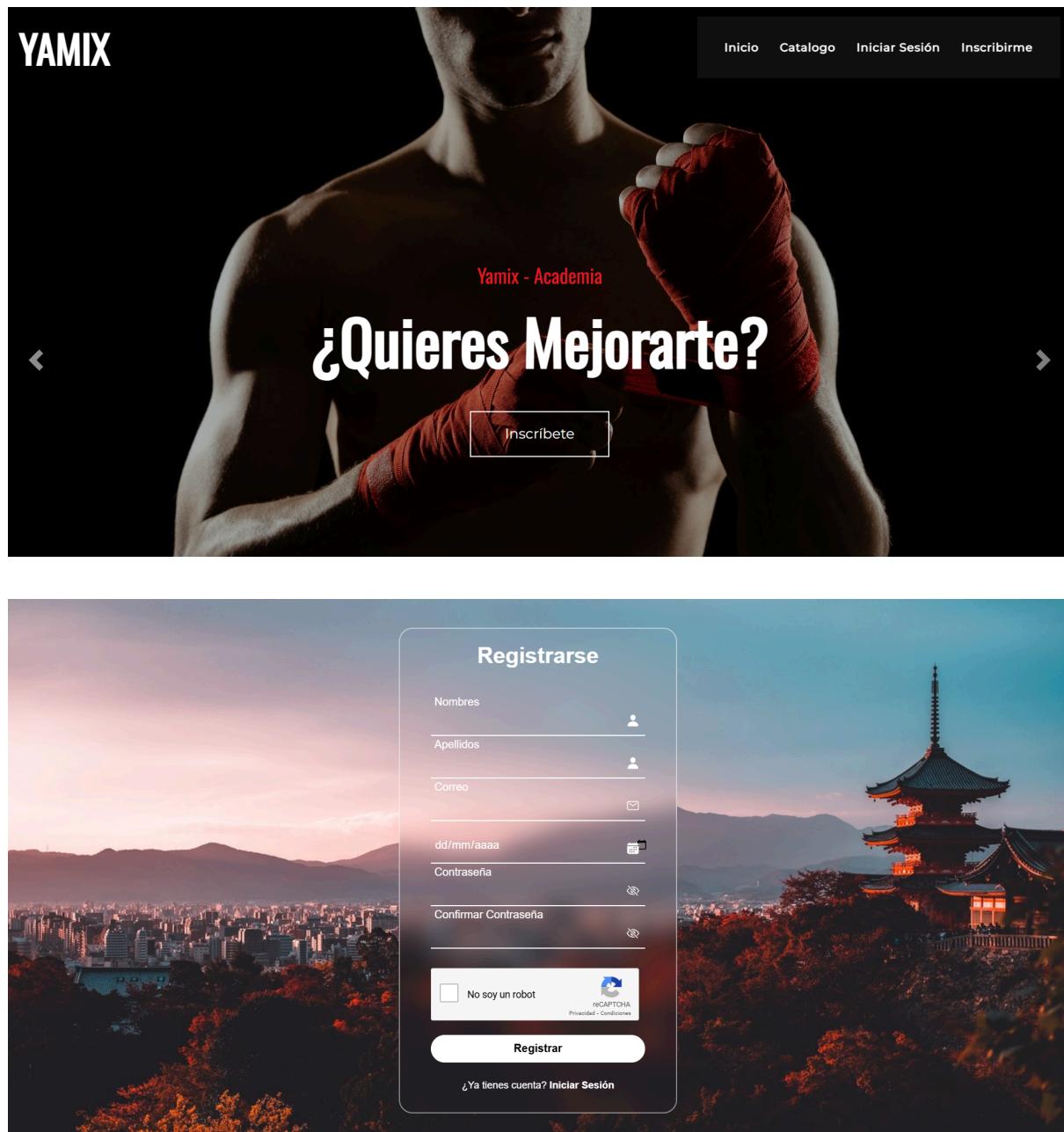
11.5 Diagrama de despliegue

Figura 12. Diagrama de despliegue



12. Prototipo

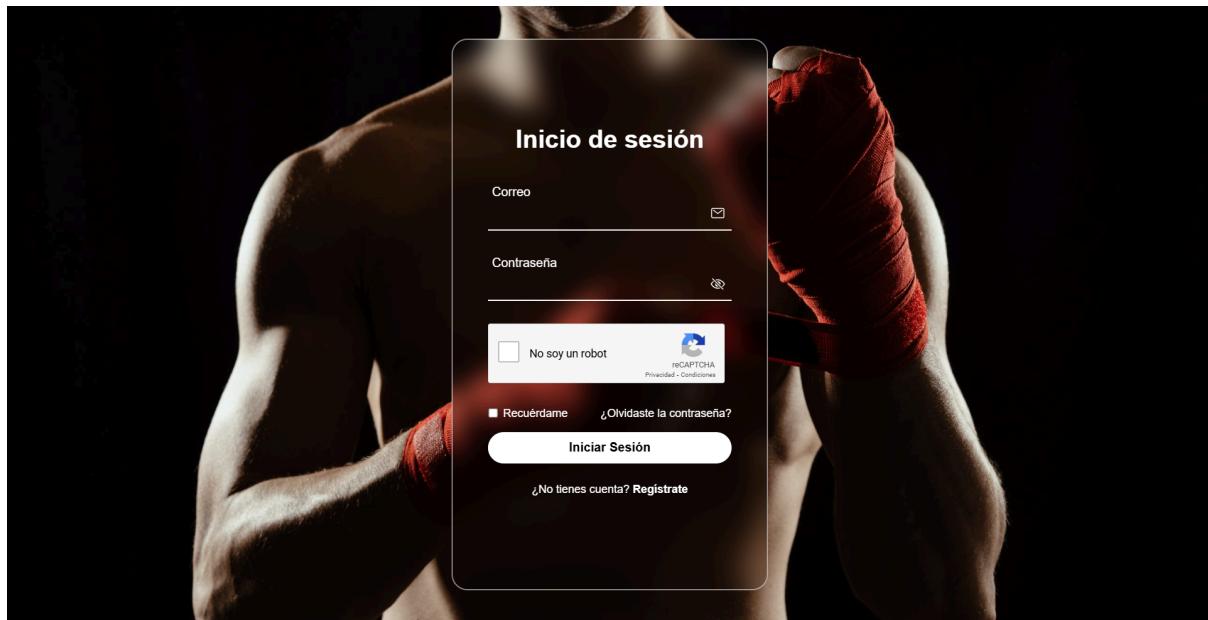
Figura 13. Prototipo



The image displays two wireframes of a web application interface for 'YAMIX'.

Home Screen: The top half shows a dark background featuring a muscular torso and arms of a person wearing red boxing wraps. The text 'YAMIX' is in the top left corner, and the navigation bar at the top right includes 'Inicio', 'Catalogo', 'Iniciar Sesión', and 'Inscribirme'. The central text reads 'Yamix - Academia' and '¿Quieres Mejorarte?'. A large button labeled 'Inscríbete' is positioned below the main text. Navigation arrows are located on the left and right sides of the main image.

Registration Screen: The bottom half shows a wireframe of a registration form titled 'Registrarse'. The form fields include 'Nombres' (with a person icon), 'Apellidos' (with a person icon), 'Correo' (with an envelope icon), 'dd/mm/aaaa' (with a calendar icon), 'Contraseña' (with an eye icon), 'Confirmar Contraseña' (with an eye icon), and a 'No soy un robot' checkbox followed by a reCAPTCHA field. A 'Registrar' button is at the bottom. The background of this screen is a scenic view of a traditional pagoda at dusk.



ManualTecnico| X YAMIX - Docum... X Corona Admin X Yamix - Google... X Refinamiento - C... X Matriz Historias X Descargar archi... X ChatGPT X +

← → G △ No seguro yamix.online/dashboard

WhatsApp Web YouTube Facebook Netflix OPGG Twitter Porfessor.g - Bú... Changelog - League... Gmail Maps Iniciar Sesión Traducir Domicilios | Trello > Todos los favoritos

YAMIX 

Opciones ▾

Navegación

- Dashboard
- Permisos
- Usuarios
- Asistencia
- Clases
- Cursos
- Calendario
- Catálogo

Estudiantes registrados

Usuarios por Curso

Categoría	Usuarios
Parkour	5.8
Boxeo	2.8
Mixedas	4.8
Yoga	0.0
Tekondo	3.8
Defensa Personal	0.0

Ampliar Gráfica

Asistencias de Estudiantes

Filtrar por mes: Selección un mes ▾

Presentes (Cyan line) | Ausentes (Red line)

Ampliar Gráfica

Estado de Usuarios

Estado de Usuarios

Categoría	Usuarios
Activo	12.5

Estudiantes Menores de Edad

Menores de Edad (Yellow) | Mayores de Edad (Blue)

Buscar  Buscar 

Esp LAA  12:33 a.m.  10/12/2024 



YAMIX

Navegación

- Dashboard
- Permisos
- Usuarios
- Asistencia
- Clases
- Cursos
- Calendario
- Catálogo

Opciones ▾

Roles

Mostrar 10 registros

#	Rol	Permiso(s)	Acción
1	Administrador	Asistencia admin · Asistencia profesor · Calendario · Catalogo admin · Clases · Cursos · Dashboard · Perfil · Permisos · Usuarios	
2	Profesor	Asistencia profesor · Perfil	
3	Estudiante	Perfil	
4	Miembro	Perfil	
5	Secretaria	Sin permisos	

Mostrando 1 a 5 de 5 registros

Anterior 1 Siguiente

YAMIX

Navegación

- Dashboard
- Permisos
- Usuarios
- Asistencia
- Clases
- Cursos
- Calendario
- Catálogo

Opciones ▾

Clases

Mostrar 10 registros

Curso	Hora Inicio	Hora Final	Instructor	Estado	Estudiantes	Acción
Boxeo	14:00:00	16:00:00	Juan Esteban	activo	+	
Defensa Personal	20:30:00	21:30:00	Yefferson	activo	+	
Mixtas	08:00:00	10:00:00	Yefferson	activo	+	
Parkour	11:00:00	13:00:00	Yefferson	activo	+	
tekondo	19:00:00	20:00:00	Yefferson	activo	+	
Yoga	17:00:00	18:00:00	Yefferson	activo	+	

Mostrando 1 a 6 de 6 registros

Anterior 1 Siguiente



YAMIX

Hoy

diciembre 2024

Semana Día

Navegación

- Dashboard
- Permisos
- Usuarios
- Asistencia
- Clases
- Cursos
- Calendario**
- Catálogo

LUN.	MAR.	MIÉ.	JUE.	VIE.	SÁB.	DOM.
25	26	27	28	29	30	1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
X 0 demostración de yoga				X 0 Clase gratis de zumba		
16	17	18	19	20	21	22
X 12 clase demostración boxeo				X 0 Clase gratis de parkour		
23	24	25	26	27	28	29
X 0 tekondo gratis				X 8 Torneo mixtas		
30	31	1	2	3	4	5
X Torneo mix	X 0 Feliz año					

Nombre:
Juan Camilo Arbelaez Diaz

Correo:
camiloardi22@gmail.com

Fecha de nacimiento:
2/21/2003

Rol:
Administrador

[Edit](#)

Horario de clases

Tus Clases

[Mostrar todas las clases](#)

Horarios	Lunes	Martes	Miercoles	Jueves	Viernes
----------	-------	--------	-----------	--------	---------



CATALOGO

Home / YAMIX

A screenshot of a website header. The background is dark with a red boxing glove graphic. The word "CATALOGO" is in large white capital letters at the top center. Below it is a smaller navigation link "Home / YAMIX".

Productos recomendados

Todos los cursos ▾

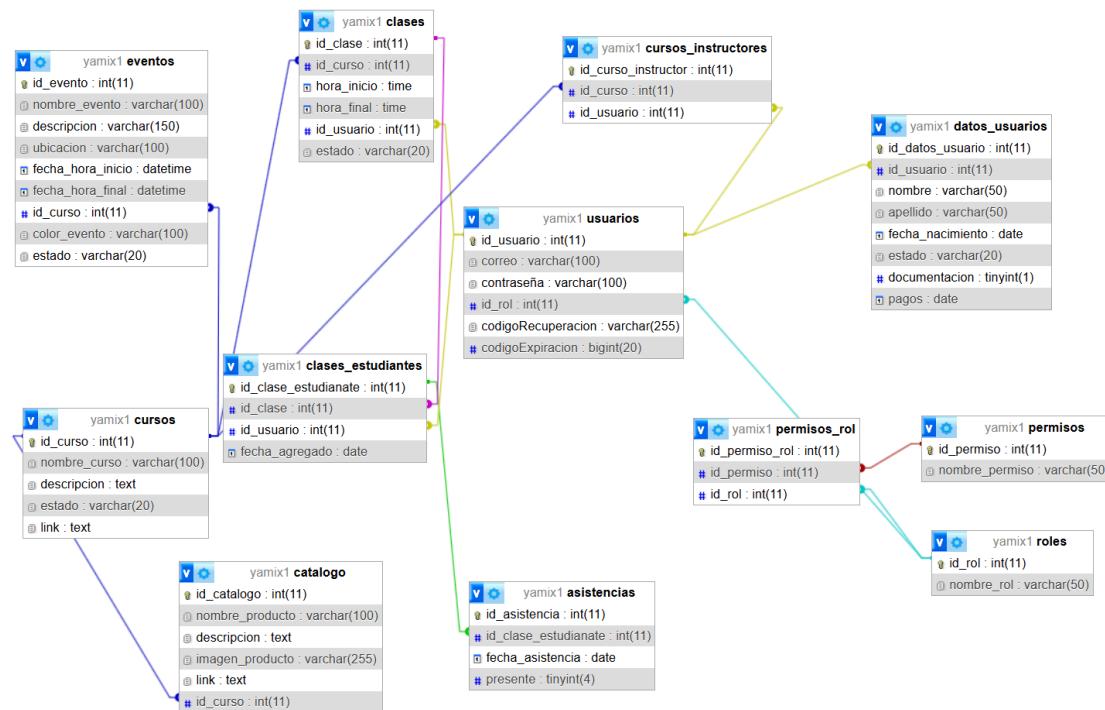


https://drive.google.com/file/d/1JteDk-jtsYv-iN-C0GBip7Zo_pJ2n3w8/view?usp=drive_link

13. Modelo de la base de datos

13.1 Modelo Relacional

Figura 14. Modelo relacional



13.2 Modelo Físico (Script)

```

-- Base de datos: `yamix`

-- 
create database yamix;

use yamix;

```



```
-- Estructura de tabla para la tabla `asistencias`  
--  
  
CREATE TABLE `asistencias` (  
  
    `id_asistencia` int(11) NOT NULL,  
  
    `id_clase` int(11) DEFAULT NULL,  
  
    `fecha_asistencia` date DEFAULT NULL  
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;  
  
--  
  
-- Volcado de datos para la tabla `asistencias`  
--  
  
INSERT INTO `asistencias`(`id_asistencia`, `id_clase`,  
`fecha_asistencia`) VALUES  
(44, 19, '2024-12-03'),  
(45, 19, '2024-11-11'),  
(46, 21, '2024-12-02'),  
(47, 19, '2024-11-12'),  
(48, 20, '2024-12-04'),  
(49, 19, '2024-12-05'),  
(50, 21, '2024-12-05'),  
(51, 24, '2024-12-07'),
```



```
(52, 24, '2024-12-07');

-- Estructura de tabla para la tabla `asistencias_estudiantes` 

CREATE TABLE `asistencias_estudiantes` (
    `id_asistencia` int(11) DEFAULT NULL,
    `id_usuario` int(11) DEFAULT NULL,
    `presente` tinyint(1) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;

-- Volcado de datos para la tabla `asistencias_estudiantes` 

INSERT INTO `asistencias_estudiantes` (`id_asistencia`, `id_usuario`, `presente`) VALUES
(44, 53, 1),
(44, 54, 1),
(44, 55, 1),
(45, 53, 1),
(45, 54, 0),
```



(45, 55, 1),

(46, 55, 1),

(46, 54, 1),

(46, 57, 0),

(47, 55, 0),

(47, 53, 1),

(47, 54, 0),

(48, 57, 1),

(49, 55, 0),

(49, 53, 1),

(49, 54, 0),

(50, NULL, 1),

(50, NULL, 0),

(50, NULL, 1),

(50, 54, 0),

(50, 55, 0),

(50, 57, 1),

(51, 54, 0),

(51, 57, 1),

(51, 55, 0),

(52, 55, 0),

(52, 57, 1),

(52, 54, 1),

(52, 63, 0);



```
--  
-- Estructura de tabla para la tabla `catalogo`  
--  
  
CREATE TABLE `catalogo` (  
  
    `id_catalogo` int(11) NOT NULL,  
  
    `nombre_producto` varchar(100) NOT NULL,  
  
    `descripcion` text DEFAULT NULL,  
  
    `imagen_producto` varchar(255) DEFAULT NULL,  
  
    `link` text NOT NULL,  
  
    `id_curso` int(11) DEFAULT NULL  
  
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;  
  
--  
-- Volcado de datos para la tabla `catalogo`  
--  
  
INSERT INTO `catalogo`(`id_catalogo`, `nombre_producto`,  
`descripcion`, `imagen_producto`, `link`, `id_curso`) VALUES  
(10, 'Guantes de box', 'box', '/uploads/1733254910761.webp',  
'https://articulo.mercadolibre.com.co/MCO-1784232978-guantes-de-boxeo-p  
rofesionales-negros-box-14-onzas-mma-elite-_JM#polyCard_client=search-n  
ordic&position=4&search_layout=grid&type=item&tracking_id=c1ddd41b-d965  
-4970-8c24-f84cbfa31ac2', 26),
```



```
(11, 'Guantes de MMA', 'estilo callejero',  
'uploads/1733254683945.jpg',  
'https://articulo.mercadolibre.com.co/MCO-614751817-guantes-mma-kick-bo  
xing-boxeo-artes-marciales-mixtas-force-_JM#polycard_client=search-nord  
ic&position=14&search_layout=grid&type=item&tracking_id=059eaebb-e2fa-4  
cd3-bd20-b7fcb5fb3194', 27),  
  
(12, 'Tenis para parkour', 'para saltar bien alto',  
'uploads/1733254829071.webp',  
'https://articulo.mercadolibre.com.co/MCO-2417670234-tenis-galaxy-6-gw3  
848-adidas-_JM#polycard_client=search-nordic&position=24&search_layout=  
grid&type=item&tracking_id=28e82f27-8648-42c8-8434-8f993cf9b238', 25),  
  
(13, 'Guantes', 'mejor agarre', '/uploads/1733255522401.jpeg',  
'https://articulo.mercadolibre.com.co/MCO-1340559585-guantes-con-munequ  
era-gimnasio-pesas-gym-sports-deporte-_JM#polycard_client=search-nordic  
&position=16&search_layout=grid&type=item&tracking_id=3dd2f944-4449-4ff  
c-abbf-81c40d598512', 25),  
  
(14, 'Tenis Para Martha', 'Tenis exclusivos para Martha',  
'uploads/1733264054709.jpeg', 'https://www.youtube.com/', 25),  
  
(15, 'Guantes', 'boxeo', '/uploads/1733605516025.jpg',  
'https://www.facebook.com/', 26),  
  
(16, 'juanes gay', 'qweqw', '/uploads/1733605549307.jpg',  
'https://www.facebook.com/', 26);  
  
-- -----  
  
--  
  
-- Estructura de tabla para la tabla `clases`  
  
--  
  
CREATE TABLE `clases` (  
  
    `id_clase` int(11) NOT NULL,  
  
    `id_curso` int(11) DEFAULT NULL,
```



```
 `hora_inicio` time DEFAULT NULL,  
  
 `hora_final` time DEFAULT NULL,  
  
 `id_usuario` int(11) DEFAULT NULL,  
  
 `estado` varchar(20) NOT NULL  
  
 ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;  
  
--  
  
-- Volcado de datos para la tabla `clases`  
  
--  
  
INSERT INTO `clases` (`id_clase`, `id_curso`, `hora_inicio`,  
 `hora_final`, `id_usuario`, `estado`) VALUES  
(19, 25, '08:00:00', '10:00:00', 52, 'activo'),  
  
(20, 26, '11:00:00', '11:30:00', 62, 'activo'),  
  
(21, 28, '16:00:00', '18:00:00', 52, 'activo'),  
  
(22, 26, '13:00:00', '15:00:00', 52, 'activo'),  
  
(23, 26, '18:01:00', '20:00:00', 56, 'activo'),  
  
(24, 30, '10:20:00', '10:50:00', 60, 'activo');  
  
-----  
  
--  
  
-- Estructura de tabla para la tabla `clases_estudiantes`  
  
--
```



```
CREATE TABLE `clases_estudiantes`  
(`id_clase_estudianate` int(11) NOT NULL,  
 `id_clase` int(11) DEFAULT NULL,  
 `id_usuario` int(11) DEFAULT NULL,  
 `fecha_agregado` date DEFAULT NULL  
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;  
  
--  
-- Volcado de datos para la tabla `clases_estudiantes`  
--  
  
INSERT INTO `clases_estudiantes`(`id_clase_estudianate`, `id_clase`,  
 `id_usuario`, `fecha_agregado`) VALUES  
(79, 19, 53, '2024-11-10'),  
(81, 19, 55, '2024-11-10'),  
(85, 19, 54, '2024-11-11'),  
(88, 20, 57, '2024-11-10'),  
(89, 21, 54, '2024-11-10'),  
(90, 21, 55, '2024-11-10'),  
(91, 21, 57, '2024-11-11'),  
(92, 20, 54, '2024-12-05'),  
(107, 22, 61, '2024-12-05'),  
(108, 21, 61, '2024-12-05'),  
(109, 19, 61, '2024-12-05'),  
(112, 22, 53, '2024-12-07'),
```



```
(113, 24, 55, '2024-12-07'),  
  
(114, 24, 54, '2024-12-07'),  
  
(115, 24, 57, '2024-12-07');  
  
--  
  
-- Estructura de tabla para la tabla `cursos`  
  
--  
  
CREATE TABLE `cursos` (  
  
    `id_curso` int(11) NOT NULL,  
  
    `nombre_curso` varchar(100) DEFAULT NULL,  
  
    `link` text NOT NULL,  
  
    `descripcion` text DEFAULT NULL,  
  
    `estado` varchar(20) NOT NULL  
  
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;  
  
--  
  
-- Volcado de datos para la tabla `cursos`  
  
--  
  
INSERT INTO `cursos` (`id_curso`, `nombre_curso`, `link`,  
    `descripcion`, `estado`) VALUES  
  
(25, 'Parkour', 'uploads\\1733248959775.jpeg', 'saltar', 'activo'),
```



```
(26, 'Boxeo', 'uploads\\1733251192077.jpg', 'boxeo en el ring',
'activo'),  
  
(27, 'Mixtas', 'uploads\\1733254151651.webp', 'Estilo callejero',
'activo'),  
  
(28, 'programacion', 'uploads\\1733262871586.webp', 'programar',
'activo'),  
  
(30, 'Kungfu', 'uploads\\1733603182561.jpg', 'ewqaeqqwe', 'activo');  
  
-- -- -- -- --  
  
-- Estructura de tabla para la tabla `cursos_instructores`  
  
-- -- -- -- --  
  
CREATE TABLE `cursos_instructores` (  
  
    `id_curso_instructor` int(11) NOT NULL,  
  
    `id_curso` int(11) DEFAULT NULL,  
  
    `id_usuario` int(11) DEFAULT NULL  
  
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;  
  
-- -- -- -- --  
  
-- Estructura de tabla para la tabla `datos_usuarios`  
  
-- -- -- -- --
```



```
CREATE TABLE `datos_usuarios` [
    `id_datos_usuario` int(11) NOT NULL,
    `id_usuario` int(11) DEFAULT NULL,
    `nombre` varchar(50) DEFAULT NULL,
    `apellido` varchar(50) DEFAULT NULL,
    `fecha_nacimiento` date DEFAULT NULL,
    `estado` varchar(20) DEFAULT NULL
] ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;

-- Volcado de datos para la tabla `datos_usuarios`

INSERT INTO `datos_usuarios` (`id_datos_usuario`, `id_usuario`, `nombre`, `apellido`, `fecha_nacimiento`, `estado`) VALUES
(1, 1, 'Juan Camilo', 'Arbelaez Diaz', '2001-02-21', 'habilitado'),
(49, 52, 'Andres', 'Gomez', '2003-12-04', 'habilitado'),
(50, 53, 'matias', 'serna', '2006-12-04', 'habilitado'),
(51, 54, 'santiago ', 'henao', '2005-12-12', 'habilitado'),
(52, 55, 'rodolfo', 'hernandez', '2005-12-06', 'habilitado'),
(53, 56, 'Matias', 'Arbelaez Diaz', '2004-12-04', 'habilitado'),
(54, 57, 'Martha', 'Gomez', '2003-12-12', 'habilitado'),
(55, 58, 'Juan', 'Betancur', '2003-12-04', 'espera'),
(56, 59, 'Mathiws', 'Arbelaez Diaz', '2001-11-11', 'habilitado'),
(57, 60, 'Christian Mathiws', 'Torres Serna', '2002-11-11', 'habilitado'),
```



```
(58, 61, 'Mathiws', 'Torres Seina', '2002-11-11', 'deshabilitado'),  
(59, 62, 'stephanie', 'Ramirez', '2003-02-21', 'habilitado'),  
(60, 63, 'Jaunes', 'Hoyos', '2003-12-12', 'habilitado');  
--  
-- Estructura de tabla para la tabla `documentos`  
--  
  
CREATE TABLE `documentos` (  
    `id_documento` int(11) NOT NULL,  
    `ruta_archivo` varchar(255) NOT NULL,  
    `id_usuario` int(11) NOT NULL  
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;  
--  
-- Volcado de datos para la tabla `documentos`  
--  
  
INSERT INTO `documentos`(`id_documento`, `ruta_archivo`, `id_usuario`) VALUES  
(50, '/uploads/documents/1733250648681.pdf', 54),  
(51, '/uploads/documents/1733262439435.pdf', 57),  
(66, '/uploads/documents/1733603665887.pdf', 54),
```



```
(67, '/uploads/documents/1733604948032.pdf', 63),  
  
(68, '/uploads/documents/1733604940508.pdf', 63),  
  
(69, '/uploads/documents/1733604948033.pdf', 63);  
  
--  
  
-- Estructura de tabla para la tabla `eventos`  
  
--  
  
CREATE TABLE `eventos` (  
  
    `id_evento` int(11) NOT NULL,  
  
    `nombre_evento` varchar(100) DEFAULT NULL,  
  
    `descripcion` varchar(150) DEFAULT NULL,  
  
    `ubicacion` varchar(100) DEFAULT NULL,  
  
    `fecha_hora_inicio` datetime DEFAULT NULL,  
  
    `fecha_hora_final` datetime DEFAULT NULL,  
  
    `id_curso` int(11) DEFAULT NULL,  
  
    `color_evento` varchar(100) NOT NULL,  
  
    `estado` varchar(20) NOT NULL  
  
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;  
  
--  
  
-- Volcado de datos para la tabla `eventos`  
  
--
```



```
INSERT INTO `eventos` (`id_evento`, `nombre_evento`, `descripcion`,  
`ubicacion`, `fecha_hora_inicio`, `fecha_hora_final`, `id_curso`,  
`color_evento`, `estado`) VALUES  
  
(9, 'Parkour municipal', 'torneo de parkour', 'a', '2024-12-27  
00:00:00', '2024-12-30 00:00:00', 25, '#9c27b0', 'inactivo'),  
  
(10, 'Torneo mixtas', 'Artes mixtas', 'Cancha La Pedrera Cl. 52 #44-05,  
Copacabana, Antioquia', '2024-12-24 00:00:00', '2024-12-27 00:00:00',  
27, '#2196F3', 'activo'),  
  
(11, 'Torneo boxeo', 'box', 'Cancha La Pedrera Cl. 52 #44-05,  
Copacabana, Antioquia', '2024-12-20 00:00:00', '2024-12-23 00:00:00',  
26, '#8BC34A', 'activo'),  
  
(12, 'programacion', 'dasdas', 'dasd', '2024-12-04 00:00:00',  
'2024-12-07 00:00:00', 28, '#009688', 'activo'),  
  
(13, 'Clase gratis box', 'boxeo gratis prueba', 'sdaasd', '2024-12-09  
00:00:00', '2024-12-12 00:00:00', 26, '#FFC107', 'activo');  
  
--  
  
-- Estructura de tabla para la tabla `permisos`  
  
--  
  
CREATE TABLE `permisos` (  
    `id_permiso` int(11) NOT NULL,  
    `nombre_permiso` varchar(50) DEFAULT NULL  
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;  
  
--
```



```
-- Volcado de datos para la tabla `permisos`  
--  
  
INSERT INTO `permisos` (`id_permiso`, `nombre_permiso`) VALUES  
(1, 'dashboard'),  
(2, 'permisos'),  
(3, 'usuarios'),  
(4, 'asistencia admin'),  
(5, 'clases'),  
(6, 'calendario'),  
(7, 'catalogo admin'),  
(8, 'asistencia profesor'),  
(9, 'perfil'),  
(10, 'cursos');  
  
-- -----  
--  
  
-- Estructura de tabla para la tabla `permisos_rol`  
--  
  
CREATE TABLE `permisos_rol` (  
  `id_permiso_rol` int(11) NOT NULL,  
  `id_permiso` int(11) DEFAULT NULL,  
  `id_rol` int(11) DEFAULT NULL
```



```
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;

-- Volcado de datos para la tabla `permisos_rol`


INSERT INTO `permisos_rol` (`id_permiso_rol`, `id_permiso`, `id_rol`)
VALUES

(457, 8, 2),
(458, 9, 2),
(459, 9, 1),
(460, 9, 4),
(461, 1, 3),
(462, 2, 3),
(463, 3, 3),
(464, 4, 3),
(465, 5, 3),
(466, 6, 3),
(467, 7, 3),
(468, 8, 3),
(469, 9, 3),
(470, 10, 3),
(472, 1, 8),
(473, 3, 8);
```



```
--  
-- Estructura de tabla para la tabla `roles`  
--  
  
CREATE TABLE `roles` (  
  
    `id_rol` int(11) NOT NULL,  
  
    `nombre_rol` varchar(50) DEFAULT NULL  
  
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;  
  
--  
-- Volcado de datos para la tabla `roles`  
--  
  
INSERT INTO `roles` (`id_rol`, `nombre_rol`) VALUES  
(1, 'Estudiante'),  
(2, 'profesor'),  
(3, 'administrador'),  
(4, 'miembro'),  
(8, 'secretaria');
```



```
-- Estructura de tabla para la tabla `usuarios`  
--  
  
CREATE TABLE `usuarios` (  
  
    `id_usuario` int(11) NOT NULL,  
  
    `correo` varchar(100) DEFAULT NULL,  
  
    `contraseña` varchar(100) DEFAULT NULL,  
  
    `id_rol` int(11) DEFAULT NULL,  
  
    `codigoRecuperacion` varchar(255) DEFAULT NULL,  
  
    `codigoExpiracion` bigint(20) DEFAULT NULL  
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;  
  
--  
  
-- Volcado de datos para la tabla `usuarios`  
  
--  
  
INSERT INTO `usuarios`(`id_usuario`, `correo`, `contraseña`, `id_rol`,  
    `codigoRecuperacion`, `codigoExpiracion`) VALUES  
(1, 'camiloardi22@gmail.com',  
    '$2a$10$NSYNov831oCoyr68nqnP4.IlnVahPI1e27kBVL92K9d1RYRPAaKIy', 3,  
    NULL, NULL),  
  
(52, 'mathiws112@gmail.com',  
    '$2a$10$tKg6bVGunLY7HnP12qLx.DQcZPQ3NWpvUMu4NRuoD3ff/t80.Fve', 2,  
    NULL, NULL),  
  
(53, 'juan2112@gmail.com',  
    '$2b$10$6EKfCeoj4ExBjxU2P6gV6uaVtfRPFqShqRv6OVE1JfOKwWw4L5Wo.', 1,  
    NULL, NULL),
```



```
(54, 'juan21@gmail.com',
'$2b$10$3p1I/UzLGZR1WMpYyaMFxOWnwn0N8x3lkaqU8JHCoFiQHDRgeF/Ta', 1,
NULL, NULL),

(55, 'juan13@gmail.com',
'$2b$10$s6eDGiR3HI3GBOHhP7N6ieUAM9djcLbvlipASROudRcu/OHVBkyCC', 1,
NULL, NULL),

(56, 'juan14@gmail.com',
'$2b$10$Ei655B/xgMYH09nOOCNvH.4vlCdBu06aubZB4Rop.6DFyvYKSG29C', 2,
NULL, NULL),

(57, 'juanesbetancur09@gmail.com',
'$2a$10$.8M7.kZrYlMqqqyPmI/G2e4uhIqHU2yLAqRwTyq8KUqXn38gqQq02', 1,
NULL, NULL),

(58, 'j@gmail.com',
'$2a$10$sb7DQxYxPqX.Drw/jfqqu0mUaXVgNyS1fHdO/KTq9s5mQMM01.dW', 3,
NULL, NULL),

(59, 'mathiwstw@gmail.com',
'$2b$10$AVIdje/ETCR2tLCRcoEJ9urdDq8uGTkgxFWbSyZcp00G.f70KwWPW', 2,
NULL, NULL),

(60, 'sas@gmail.com',
'$2b$10$/2YS/u7o9v.ooajs97T0seqbHUDZsuxvXJ8sAZCtt8MezfMP5VtrG', 2,
NULL, NULL),

(61, 'mathiwsts@gmail.com',
'$2b$10$NEPtam2EKNZz4dsrAtgz9eSbgAcTMs/6MFLSydbYjUOFAG9/gVwBO', 1,
NULL, NULL),

(62, 'stephanieramirezarredondo@gmail.com',
'$2b$10$fuEDEzDOhx4.9ohjyn8ouJi/BjQbchMcDVLfP.t/vESwyFx dmK.2', 2,
NULL, NULL),

(63, 'camiloardi32@gmail.com',
'$2a$10$uNseoInTvie/ZdIHZftckuX9njCc6w2PaC75Ph0ySmkJdMs1kVXgS', 1,
NULL, NULL);

-- 
-- Índices para tablas volcadas
```



```
--  
--  
-- Indices de la tabla `asistencias`  
--  
  
ALTER TABLE `asistencias`  
  
    ADD PRIMARY KEY (`id_asistencia`),  
  
    ADD KEY `id_clase` (`id_clase`);  
  
--  
-- Indices de la tabla `asistencias_estudiantes`  
--  
  
ALTER TABLE `asistencias_estudiantes`  
  
    ADD KEY `id_asistencia` (`id_asistencia`),  
  
    ADD KEY `id_usuario` (`id_usuario`);  
  
--  
-- Indices de la tabla `catalogo`  
--  
  
ALTER TABLE `catalogo`  
  
    ADD PRIMARY KEY (`id_catalogo`),  
  
    ADD KEY `id_curso` (`id_curso`);  
  
--  
-- Indices de la tabla `clases`
```



```
--  
  
ALTER TABLE `clases`  
  
    ADD PRIMARY KEY (`id_clase`),  
  
    ADD KEY `id_curso` (`id_curso`),  
  
    ADD KEY `id_usuario` (`id_usuario`);  
  
--  
  
-- Indices de la tabla `clases_estudiantes`  
  
--  
  
ALTER TABLE `clases_estudiantes`  
  
    ADD PRIMARY KEY (`id_clase_estudianate`),  
  
    ADD KEY `id_clase` (`id_clase`),  
  
    ADD KEY `id_usuario` (`id_usuario`);  
  
--  
  
-- Indices de la tabla `cursos`  
  
--  
  
ALTER TABLE `cursos`  
  
    ADD PRIMARY KEY (`id_curso`);  
  
--  
  
-- Indices de la tabla `cursos_instructores`  
  
--  
  
ALTER TABLE `cursos_instructores`  
  
    ADD PRIMARY KEY (`id_curso_instructor`),
```



```
ADD KEY `id_curso`(`id_curso`),  
    ADD KEY `id_usuario`(`id_usuario`);  
  
--  
  
-- Indices de la tabla `datos_usuarios`  
  
--  
  
ALTER TABLE `datos_usuarios`  
    ADD PRIMARY KEY (`id_datos_usuario`),  
    ADD KEY `id_usuario`(`id_usuario`);  
  
--  
  
-- Indices de la tabla `documentos`  
  
--  
  
ALTER TABLE `documentos`  
    ADD PRIMARY KEY (`id_documento`),  
    ADD KEY `id_usuario`(`id_usuario`);  
  
--  
  
-- Indices de la tabla `eventos`  
  
--  
  
ALTER TABLE `eventos`  
    ADD PRIMARY KEY (`id_evento`),  
    ADD KEY `id_curso`(`id_curso`);  
  
--
```



```
-- Indices de la tabla `permisos`  
--  
  
ALTER TABLE `permisos`  
  
ADD PRIMARY KEY (`id_permiso`);  
  
--  
  
-- Indices de la tabla `permisos_rol`  
--  
  
ALTER TABLE `permisos_rol`  
  
ADD PRIMARY KEY (`id_permiso_rol`),  
ADD KEY `id_permiso` (`id_permiso`),  
ADD KEY `id_rol` (`id_rol`);  
  
--  
  
-- Indices de la tabla `roles`  
--  
  
ALTER TABLE `roles`  
  
ADD PRIMARY KEY (`id_rol`);  
  
--  
  
-- Indices de la tabla `usuarios`  
--  
  
ALTER TABLE `usuarios`  
  
ADD PRIMARY KEY (`id_usuario`),  
ADD KEY `id_rol` (`id_rol`);
```

```
-- AUTO_INCREMENT de las tablas volcadas

-- AUTO_INCREMENT de la tabla `asistencias` 

ALTER TABLE `asistencias` 

    MODIFY `id_asistencia` int(11) NOT NULL AUTO_INCREMENT,
AUTO_INCREMENT=53;

-- AUTO_INCREMENT de la tabla `catalogo` 

ALTER TABLE `catalogo` 

    MODIFY `id_catalogo` int(11) NOT NULL AUTO_INCREMENT,
AUTO_INCREMENT=17;

-- AUTO_INCREMENT de la tabla `clases` 

ALTER TABLE `clases` 

    MODIFY `id_clase` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=25;
```



```
-- AUTO_INCREMENT de la tabla `clases_estudiantes`  
--  
  
ALTER TABLE `clases_estudiantes`  
  
    MODIFY `id_clase_estudianate` int(11) NOT NULL AUTO_INCREMENT,  
AUTO_INCREMENT=117;  
  
--  
  
-- AUTO_INCREMENT de la tabla `cursos`  
--  
  
ALTER TABLE `cursos`  
  
    MODIFY `id_curso` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=31;  
  
--  
  
-- AUTO_INCREMENT de la tabla `cursos_instructores`  
--  
  
ALTER TABLE `cursos_instructores`  
  
    MODIFY `id_curso_instructor` int(11) NOT NULL AUTO_INCREMENT;  
  
--  
  
-- AUTO_INCREMENT de la tabla `datos_usuarios`  
--  
  
ALTER TABLE `datos_usuarios`  
  
    MODIFY `id_datos_usuario` int(11) NOT NULL AUTO_INCREMENT,  
AUTO_INCREMENT=61;  
  
--
```



```
-- AUTO_INCREMENT de la tabla `documentos`  
--  
  
ALTER TABLE `documentos`  
  
    MODIFY `id_documento` int(11) NOT NULL AUTO_INCREMENT,  
AUTO_INCREMENT=70;  
  
--  
  
-- AUTO_INCREMENT de la tabla `eventos`  
  
--  
  
ALTER TABLE `eventos`  
  
    MODIFY `id_evento` int(11) NOT NULL AUTO_INCREMENT,  
AUTO_INCREMENT=14;  
  
--  
  
-- AUTO_INCREMENT de la tabla `permisos`  
  
--  
  
ALTER TABLE `permisos`  
  
    MODIFY `id_permiso` int(11) NOT NULL AUTO_INCREMENT,  
AUTO_INCREMENT=21;  
  
--  
  
-- AUTO_INCREMENT de la tabla `permisos_rol`  
  
--  
  
ALTER TABLE `permisos_rol`  
  
    MODIFY `id_permiso_rol` int(11) NOT NULL AUTO_INCREMENT,  
AUTO_INCREMENT=474;
```



```
--  
-- AUTO_INCREMENT de la tabla `roles`  
--  
  
ALTER TABLE `roles`  
  
MODIFY `id_rol` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=9;  
  
--  
--  
-- AUTO_INCREMENT de la tabla `usuarios`  
--  
  
ALTER TABLE `usuarios`  
  
MODIFY `id_usuario` int(11) NOT NULL AUTO_INCREMENT,  
AUTO_INCREMENT=64;  
  
--  
--  
-- Restricciones para tablas volcadas  
--  
  
--  
--  
  
-- Filtros para la tabla `asistencias`  
--  
  
ALTER TABLE `asistencias`  
  
ADD CONSTRAINT `asistencias_ibfk_1` FOREIGN KEY (`id_clase`)  
REFERENCES `clases` (`id_clase`);  
  
--  
--  
-- Filtros para la tabla `asistencias_estudiantes`
```



```
--  
  
ALTER TABLE `asistencias_estudiantes`  
  
    ADD CONSTRAINT `asistencias_estudiantes_ibfk_1` FOREIGN KEY  
(`id_usuario`) REFERENCES `usuarios` (`id_usuario`),  
  
    ADD CONSTRAINT `asistencias_estudiantes_ibfk_2` FOREIGN KEY  
(`id_asistencia`) REFERENCES `asistencias` (`id_asistencia`);  
  
--  
  
-- Filtros para la tabla `catalogo`  
  
--  
  
ALTER TABLE `catalogo`  
  
    ADD CONSTRAINT `catalogo_ibfk_1` FOREIGN KEY (`id_curso`) REFERENCES  
 `cursos` (`id_curso`);  
  
--  
  
-- Filtros para la tabla `clases`  
  
--  
  
ALTER TABLE `clases`  
  
    ADD CONSTRAINT `clases_ibfk_1` FOREIGN KEY (`id_curso`) REFERENCES  
 `cursos` (`id_curso`),  
  
    ADD CONSTRAINT `clases_ibfk_2` FOREIGN KEY (`id_usuario`) REFERENCES  
 `usuarios` (`id_usuario`);  
  
--  
  
-- Filtros para la tabla `clases_estudiantes`  
  
--  
  
ALTER TABLE `clases_estudiantes`
```



```
ADD CONSTRAINT `clases_estudiantes_ibfk_1` FOREIGN KEY (`id_clase`)
REFERENCES `clases` (`id_clase`),  
  
ADD CONSTRAINT `clases_estudiantes_ibfk_2` FOREIGN KEY (`id_usuario`)
REFERENCES `usuarios` (`id_usuario`);  
  
--  
  
-- Filtros para la tabla `cursos_instructores`  
  
--  
  
ALTER TABLE `cursos_instructores`  
  
ADD CONSTRAINT `cursos_instructores_ibfk_1` FOREIGN KEY (`id_curso`)
REFERENCES `cursos` (`id_curso`),  
  
ADD CONSTRAINT `cursos_instructores_ibfk_2` FOREIGN KEY
(`id_usuario`) REFERENCES `usuarios` (`id_usuario`);  
  
--  
  
-- Filtros para la tabla `datos_usuarios`  
  
--  
  
ALTER TABLE `datos_usuarios`  
  
ADD CONSTRAINT `datos_usuarios_ibfk_1` FOREIGN KEY (`id_usuario`)
REFERENCES `usuarios` (`id_usuario`);  
  
--  
  
-- Filtros para la tabla `documentos`  
  
--  
  
ALTER TABLE `documentos`  
  
ADD CONSTRAINT `documentos_ibfk_1` FOREIGN KEY (`id_usuario`)
REFERENCES `usuarios` (`id_usuario`) ON DELETE CASCADE ON UPDATE
CASCADE;
```



```
-- Filtros para la tabla `eventos`  
--  
  
ALTER TABLE `eventos`  
  
    ADD CONSTRAINT `eventos_ibfk_1` FOREIGN KEY (`id_curso`) REFERENCES  
    `cursos`(`id_curso`);  
  
--  
  
-- Filtros para la tabla `permisos_rol`  
--  
  
ALTER TABLE `permisos_rol`  
  
    ADD CONSTRAINT `permisos_rol_ibfk_1` FOREIGN KEY (`id_rol`) REFERENCES `roles`(`id_rol`),  
  
    ADD CONSTRAINT `permisos_rol_ibfk_2` FOREIGN KEY (`id_permiso`) REFERENCES `permisos`(`id_permiso`);  
  
--  
  
-- Filtros para la tabla `usuarios`  
--  
  
ALTER TABLE `usuarios`  
  
    ADD CONSTRAINT `usuarios_ibfk_2` FOREIGN KEY (`id_rol`) REFERENCES `roles`(`id_rol`);  
  
COMMIT;
```



13.3 Diccionario de datos

Figura 15. Diccionario de datos

Nombre de la tabla: USUARIOS							
Nombre del campo o Atributo:	Llave Primaria o Primary Key:	Tipo de Columna:	Valor por default o Defecto:	Longitud:	Valores Nulos:	Observación o Descripción:	Llave Secundaria o Foreign Key:
id_usuario	Si	Entero	No	11	Not null	Identificador único para cada usuario en la base de datos.	No
correo	No	VARCHAR	No	100	Not null	Dirección de correo electrónico asociada al usuario.	No
contraseña	No	VARCHAR	No	100	Not null	Contraseña encriptada utilizada para la autenticación de usuario.	No
id_rol	No	Entero	No	20	Not null	Referencia al rol asignado al usuario en la tabla ROLES.	Si
codigoRecuperacion	No	VARCHAR	No	20	Null	Código temporal utilizado para recuperar la cuenta en caso de olvido.	No
codigoExpiracion	No	BIGINT	No	20	Null	Código temporal utilizado para recuperar la cuenta en caso de expiración.	No

Nombre de la tabla: Clases							
Nombre del campo o Atributo:	Llave Primaria o Primary Key:	Tipo de Columna:	Valor por default o Defecto:	Longitud:	Valores Nulos:	Observación o Descripción:	Llave Secundaria o Foreign Key:
id_clase	Si	Entero	No	12	Not null	Identificador único para cada clase registrada en el sistema.	No
estado	No	VARCHAR	No	50	Not null	Estado actual de la clase, como activo, finalizado, o cancelado.	No
hora_inicio	No	TIME	No	80	Not null	Hora en la que comienza la clase.	No
hora_final	No	TIME	No	20	Not null	Hora en la que termina la clase.	No
id_usuario	No	Entero	No	11	Not null	Referencia al instructor o usuario asociado a la clase, caso de ser instructor.	Si
id_curso	No	Entero	No	11	Not null	Referencia al curso al que pertenece la clase, conectado si es instructor.	Si

Nombre de la tabla: ASISTENCIAS							
Nombre del campo o Atributo:	Llave Primaria o Primary Key:	Tipo de Columna:	Valor por default o Defecto:	Longitud:	Valores Nulos:	Observación o Descripción:	Llave Secundaria o Foreign Key:
id_asistencia	Si	Entero	No	12	Not null	Identificador único para cada registro de asistencia.	No
id_clase_estudiante	Si	Entero	No	12	Not null	Referencia a la relación entre clase y estudiante, conectado.	Si
fecha_asistencia	No	DATE	No	80	Not null	Fecha en la que se registra la asistencia del estudiante a la clase.	No
presente	No	TINYINT	No	1	Not null	Indica si el estudiante estuvo presente en la clase. Valor 1 si es presente.	No

Nombre de la tabla: CURSOS							
Nombre del campo o Atributo:	Llave Primaria o Primary Key:	Tipo de Columna:	Valor por default o Defecto:	Longitud:	Valores Nulos:	Observación o Descripción:	Llave Secundaria o Foreign Key:
id_curso	Si	Entero	No	12	Not null	Identificador único para cada curso.	No
link	No	VARCHAR	No	50	Not null	Enlace asociado al curso, como material de referencia.	No
nombre_curso	No	VARCHAR	No	80	Not null	Nombre del curso, que identifica de manera única su contenido.	No
descripcion	No	VARCHAR	No	80	Not null	Descripción del curso, que detalla su contenido, objetivos y demás.	No
estado	No	VARCHAR	No	80	Not null	Estado actual del curso, que puede ser "activo", "inactivo".	No

Nombre de la tabla: CURSO_INSTRUCTORES							
Nombre del campo o Atributo:	Llave Primaria o Primary Key:	Tipo de Columna:	Valor por default o Defecto:	Longitud:	Valores Nulos:	Observación o Descripción:	Llave Secundaria o Foreign Key:
id_curso_instructor	Si	Entero	No	12	Not null	Identificador único para cada relación entre un curso y un instructor.	No
id_curso	No	Entero	No	50	Not null	Referencia al curso relacionado, identificándolo en la tabla CURSOS.	Si
id_usuario	No	Entero	No	50	Not null	Referencia al usuario que actúa como instructor, identificándolo en la tabla USUARIOS.	Si

Nombre de la tabla: permisos_rol							
Nombre del campo o Atributo:	Llave Primaria o Primary Key:	Tipo de Columna:	Valor por default o Defecto:	Longitud:	Valores Nulos:	Observación o Descripción:	Llave Secundaria o Foreign Key:
id_permiso-rol	si	Entero	No	20	Not null	Identificador único para cada relación entre un permiso y un rol.	No
id_rol	No	Entero	No	20	Not null	Referencia al rol asociado, identificándolo en la tabla ROLES.	Si
id_permiso	No	Entero	No	11	null	Referencia al permiso asociado, identificándolo en la tabla PERMISOS.	No



Nombre de la tabla: PERMISOS

Nombre del campo o Atributo:	Llave Primaria o Primary Key:	Tipo de Columna:	Valor por default o Defecto:	Longitud:	Valores Nulos:	Observación o Descripción:	Llave Secundaria o Foreign Key:
id_permiso	Si	Entero	No	12	Not null	Identificador único para cada permiso definido en el sistema.	No
nombre_permiso	No	Char	No	50	Not null	Nombre descriptivo del permiso, indicando la acción o operación.	No

Nombre de la tabla: CATALOGO

Nombre del campo o Atributo:	Llave Primaria o Primary Key:	Tipo de Columna:	Valor por default o Defecto:	Longitud:	Valores Nulos:	Observación o Descripción:	Llave Secundaria o Foreign Key:
id_catalogo	Si	Entero	No	12	Not null	Identificador único para cada producto o recurso en el catálogo.	No
nombre_producto	No	VARCHAR	No	50	Not null	Nombre descriptivo del producto o recurso, por ejemplo, "Laptop".	No
descripcion	No	VARCHAR	No	20	Not null	Descripción del producto, detallando sus características.	No
link	No	VARCHAR	No	20	Not null	Enlace a una página de referencia o compra para el producto.	No
imagen_producto	No	VARCHAR	No	20	Not null	URL o ruta de la imagen representativa del producto en la web.	No
id_curso	No	Entero	No	11	null	Referencia al curso relacionado con el producto, identificándolo.	Si

Nombre de la tabla: eventos

Nombre del campo o Atributo:	Llave Primaria o Primary Key:	Tipo de Columna:	Valor por default o Defecto:	Longitud:	Valores Nulos:	Observación o Descripción:	Llave Secundaria o Foreign Key:
id_evento	Si	Entero	No	12	Not null	Identificador único para cada evento registrado.	No
nombre_evento	No	VARCHAR	No	50	Not null	Nombre descriptivo del evento, por ejemplo, "Clase de programación".	No
descripción	No	VARCHAR	No	80	Not null	Breve descripción o detalles adicionales sobre el evento.	No
ubicación	No	VARCHAR	No	20	Not null	Lugar físico o virtual donde se llevará a cabo el evento.	No
fecha_hora_inicio	No	DATETIME	No	11	null	Fecha y hora en la que comienza el evento.	No
fecha_hora_final	No	DATETIME	No	50	Not null	Fecha y hora en la que finaliza el evento.	No
id_curso	No	Entero	No	12	Not null	Referencia al curso asociado con el evento, identificándolo.	No
color_evento	No	VARCHAR	No	20	Not null	Código de color asociado al evento, útil para la visualización.	No
estado	No	VARCHAR	No	20	Not null	Estado actual del evento, como "activo", "cancelado" o "pendiente".	No

Nombre de la tabla: CLASES_ESTUDIANTES

Nombre del campo o Atributo:	Llave	Tipo de	Valor por	Longitud:	Valores	Observación o Descripción:	Llave
id_clase_estudiante	Si	Entero	No	12	Not null	Identificador único para la relación entre una clase y un estudiante.	No
id_clase	No	Entero	No	50	Not null	Referencia a la clase relacionada, identificándola en la tabla.	Si
id_usuario	No	Entero	No	50	Not null	Referencia al usuario estudiante relacionado, identificándolo.	Si
fecha_agregado	No	DATE	No		Not null	Fecha en la que el estudiante fue inscrito en la clase.	No

Nombre de la tabla: CATALOGO

Nombre del campo o Atributo:	Llave Primaria o Primary Key:	Tipo de Columna:	Valor por default o Defecto:	Longitud:	Valores Nulos:	Observación o Descripción:	Llave Secundaria o Foreign Key:
id_catalogo	Si	Entero	No	12	Not null	Identificador único para cada producto o recurso en el catálogo.	No
nombre_producto	No	VARCHAR	No	50	Not null	Nombre descriptivo del producto o recurso, por ejemplo, "Laptop".	No
descripcion	No	VARCHAR	No	20	Not null	Descripción del producto, detallando sus características.	No
link	No	VARCHAR	No	20	Not null	Enlace a una página de referencia o compra para el producto.	No
imagen_producto	No	VARCHAR	No	20	Not null	URL o ruta de la imagen representativa del producto en la web.	No
id_curso	No	Entero	No	11	null	Referencia al curso relacionado con el producto, identificándolo.	Si

Nombre de la tabla: DATOS_USUARIOS

Nombre del campo o Atributo:	Llave Primaria o Primary Key:	Tipo de Columna:	Valor por default o Defecto:	Longitud:	Valores Nulos:	Observación o Descripción:	Llave Secundaria o Foreign Key:
id_datos_usuario	Si	Entero	No	11	Not null	Identificador único para los datos adicionales de un usuario.	No
nombre	No	VARCHAR	No	100	Not null	Nombre del usuario.	No
apellido	No	VARCHAR	No	100	Not null	Apellido del usuario.	No
id_usuario	No	Entero	No	20	Not null	Referencia al usuario asociado, identificándolo en la tabla.	Si
fecha_nacimiento	No	VARCHAR	No	20	Null	Fecha de nacimiento del usuario.	No
estado	No	BIGINT	No	20	Null	Estado actual del usuario, como "activo", "inactivo", o "pendiente".	No
documentacion	No	TINYINT	No	1	Not null	Indicador (1 o 0) de si el usuario ha proporcionado la documentación.	No

<https://docs.google.com/spreadsheets/d/1u0-M2d9Ejwr0tPNdnps3EOLmHHV3f9lj/edit?gid=1530018788#gid=1530018788>



14. Seguridad

Figura 15. Seguridad

- **Algoritmo de encriptación:** Se ha usado un encriptador de contraseñas basado en la técnica **cryptjs.hash** (Estándar de cifrado avanzado).

```
const hashedPassword = await  
bcryptjs.hash(contraseña, saltRounds);
```

- **Manejo de tokens:** Se realiza la protección de peticiones exigiendo en cada una un **token**.

```
const jwt = require('jsonwebtoken');  
  
exports.verifyToken = (req, res, next) => {  
  
    const token = req.cookies.jwt; // Asegúrate de tener cookies  
    configuradas  
  
    if (!token) {  
  
        return res.status(401).json({ message: 'No estás autenticado' }) ;  
    }  
  
    jwt.verify(token, process.env.JWT_SECRET, (err, decodedToken) => {  
  
        if (err) {  
  
    }  
});
```



```
        console.error('Error de verificación del token:', err);

        return res.status(401).json({ message: 'Token no válido'
    }) ;

}

// Verifica que el token tenga el nombre del rol

const nombre_rol = decodedToken.rol;

if (!nombre_rol) {

    return res.status(403).json({ message: 'Rol no encontrado
en el token' });
}

// Guardar el nombre del rol en req.usuario

req.usuario = {

    id_usuario: decodedToken.id,
    rol: nombre_rol,
};

req.token = token;

next();
}) ;
} ;
```



15. Consideraciones especiales para la configuración

Figura 16. Consideraciones especiales para la configuración

Variables de entorno

```
const app = express();

// Configura cookie-parser para manejar cookies

app.use(cookieParser());

// Configura CORS para permitir solicitudes desde el frontend

app.use(cors({
    origin: ['http://yamix.online/'], // Cambia esto si tu frontend
    está en otra URL

    credentials: true // Permite el envío de cookies
}));

// Configura el motor de plantillas

app.set('view engine', 'ejs');

app.set('views', path.join(__dirname, 'views'));
```



```
// Configura la carpeta para archivos estáticos  
  
app.use(express.static('public'));  
  
  
// Configura el parseo de solicitudes URL-encoded y JSON  
  
app.use(express.urlencoded({ extended: true }));  
  
app.use(express.json());  
  
  
  
  
// Carga las variables de entorno  
  
dotenv.config({ path: './env/.env' });  
  
  
  
  
// Rutas  
  
  
  
  
// Configura las rutas  
  
  
  
  
app.use('/',rutas, auth,dashboard, usuarios, cursos, clases,  
roles,eventos, catalogo,asistencia);
```



```
// Middleware para manejar errores

app.use((err, req, res, next) => {

    console.error('Error en el servidor:', err);

    res.status(500).send('Error interno del servidor');

}) ;

// Inicia el servidor

app.listen(3000, () => {

    console.log('Server running in port http://localhost:3000/');
}) ;
```

Tokens de autenticación.

```
// Verifica que el token esté presente

if (!req.cookies.jwt) {

    throw new Error('Token no presente');

}
```



En el entorno de desarrollo, el servidor está configurado para utilizar el puerto 4000 por defecto. Esto permite a los desarrolladores acceder fácilmente a la aplicación localmente utilizando ese puerto específico. Sin embargo, cuando la aplicación es desplegada en un servidor en la nube, la gestión de puertos funciona de manera diferente. En lugar de especificar un puerto fijo, el servidor en la nube selecciona automáticamente un puerto libre disponible en su sistema interno para garantizar que no haya conflictos con otros servicios que puedan estar corriendo en la misma infraestructura



16. Migración

La API del sistema realiza una migración automática de datos cada vez que detecta una cambio en la estructura de las entidades (clases) del sistema haciendo uso de la sincronización de Sequelize:

```
Usuario

  .sync({ alter: true })

  .then((result) => {
    console.log("Modelo actualizado");
  })

  .catch((err) => { console.log(err);
}) ;
```



17. Backups

El servidor en la nube donde se desplegó la base de datos permite, por defecto, realizar backups automáticos de la información. Estos backups de bases de datos son de tipo completo, no incrementales. Esto significa que se realiza una copia completa de toda la base de datos diariamente, en lugar de solo guardar los cambios o diferencias desde el último backup. Este proceso se realiza diariamente para las bases de datos MySQL de forma predeterminada. Con un plan premium, esta periodicidad puede reducirse hasta llegar al nivel de hacer backups cada hora.

Especificación de Requerimientos:

Datos, D. de B. de D. D. de B. (s/f). Especificación de requerimientos. Ugr.es. Recuperado el 2 de abril de 2024, de <https://elvex.ugr.es/idbis/db/docs/design/2-requirements.pdf>

Node.js:

NodeJS. Index | Node.js v22.9.0 Documentation. Recuperado el 10 de diciembre de 2024, de <https://nodejs.org/en/docs>

Figma:

Wallace, D. F. E. (2016). Figma. Recuperado el 10 de diciembre de 2024, de <https://www.figma.com>

GitHub:



GitHub. GitHub Docs. Recuperado el 10 de diciembre de 2024, de <https://docs.github.com>

EJS (Embedded JavaScript Templates):

EJS. (s/f). EJS: Embedded JavaScript templates. Recuperado el 10 de diciembre de 2024, de <https://ejs.co>

Express.js. (s/f). EJS View Engine. Recuperado el 10 de diciembre de 2024, de <https://expressjs.com/en/resources/templates/ejs.html>

Postman:

Postman. (s/f). Postman Documentation. Recuperado el 10 de diciembre de 2024, de <https://www.postman.com/docs>

Postman. (2024). Getting Started with Postman. Recuperado el 10 de diciembre de 2024, de <https://www.postman.com/getting-started>

Google API:

Google Developers. (s/f). Google API Documentation. Recuperado el 10 de diciembre de 2024, de <https://developers.google.com/apis-explorer>

Google Cloud. (s/f). Google Cloud API Documentation. Recuperado el 10 de diciembre de 2024, de <https://cloud.google.com/apis>

Google. (s/f). Google API Console. Recuperado el 10 de diciembre de 2024, de <https://console.developers.google.com>

Trimestre 2 - Requisitos

1. Introducción
2. Técnicas de recolección aplicadas
 - 2.1 Técnica zzz
3. Ficha de proyecto



2.1. Planteamiento del problema

2.1.1 Justificación

2.2 Objetivos

2.2.1 Objetivo general

2.2.2 Objetivos específicos

2.3 Alcance del proyecto

4. Mapa de procesos

5. Facilitación gráfica

6. Matriz historias de usuario, épicas y criterios de aceptación

Trimestre 3 - Análisis

1. Story mapping

2. Wireframe (Balsamiq)

3. Product backlog priorizado

4. Diagrama de casos de uso

5. Documentación de casos de uso

6. Modelo lógico de la base de datos

7. Diagrama de clases

Trimestre 4 - Modelado

1. Prototipo Figma

2. Prototipo frontend

3. Modelo físico de base de datos

4. Script

5. Diccionario de base de datos



6. Refinamiento diagrama de clases
7. Diagrama de componentes
8. Diagrama de despliegue

Trimestre 5 - Construcción

1. Plataforma de desarrollo para móviles
2. Plataforma de desarrollo para web
3. Sistema operativo
4. Motor de base de datos

Trimestre 6

1. Requerimientos del sistema
 - 1.1 Requerimientos de hardware
 - 1.2 Requerimientos de software
2. Ayudas en línea (Herramienta tecnológica o microvideos de cada módulo)

Trimestre 7

1. Políticas de seguridad
2. Migraciones
3. Backups
4. Capacitación usuario