



Documento de Propuesta de Diseño de Software I, II y III VIDEO RECORDING

AUTORES:

- Bruno Aguirre Mayerlis Paola
mbrunoaguirre53@correo.unicordoba.edu.co
- Briceño Ojeda Cesar Luis
cbricenoojeda42@correo.unicordoba.edu.co
- Moreno Zapata Yerson David
ymorenzapata@correo.unicordoba.edu.co
- Álvarez Ortega José Fernando
jalvarezortega69@correo.unicordoba.edu.co
- Flórez Causil Arturo José
aflorezcausil@correo.unicordoba.edu.co

TUTOR:

Alexander Enrique Toscano Ricardo.



BREVE RESEÑA

Diseñar un componente para la plataforma administradora de contenidos que se encargue de grabar pantalla, permitiendo la grabación de esta misma por completo o de una región en específico según la necesidad del usuario ofreciendo diferentes resoluciones y calidades de video con la posibilidad de grabar audio del sistema o del micrófono local almacenando las grabaciones en un formato de video compatible (MP4, AVI, etc.). Organizando las grabaciones por fecha y hora o nombres de archivo Ofreciendo opciones para compartir las grabaciones por correo electrónico, plataformas en la nube o enlaces directos, implementando herramientas básicas de edición como cortar, unir y eliminar partes de las grabaciones. Dando opción de añadir títulos, créditos y marcas de agua a las grabaciones.

Ofreciendo opciones para ajustar el brillo, contraste y volumen de las grabaciones y botones de fácil acceso para iniciar, detener y pausar la grabación.

Mostrar información detallada como el tiempo real de la duración de la grabación, la fecha (hora, día, mes, año), el peso del archivo, el formato y el espacio disponible en disco. Permitiendo la configuración de las opciones de grabación antes de iniciar el proceso

Tabla de contenido

Documento de Propuesta de Diseño de Software I, II y III	2
Tabla de contenido	4
Etapa 1 Diseño de la Aplicación y Análisis de Requisitos	7
Introducción	7
Propósito del Documento	7
Etapa 1 Diseño de la Aplicación y Análisis de Requisitos	7
Etapa 2: Persistencia de Datos con Backend – Servidor	7
Etapa 3: Consumo de Datos y Desarrollo Frontend – Cliente	8
Alcance del Proyecto	8
Definiciones y Acrónimos	9
Descripción General	10
Objetivos del Sistema	10
Interfaz	10
Conceptos de las entidades	11
Funcionalidad General	11
Usuarios del Sistema	11
Restricciones	11
Requisitos Funcionales	11
Casos de Uso	11
Descripción detallada de cada caso de uso	13
Diagramas de Secuencia	25
Prioridad de Requisitos	30
Requisitos No Funcionales	31

Requisitos de Desempeño	32
Requisitos de Seguridad	32
Requisitos de Usabilidad	32
Requisitos de Escalabilidad	32
Modelado E/R	32
Caracterización de los datos Diagrama de Entidad-Relación	32
Diagrama relacional	32
Descripción de Entidades y Relaciones	32
Reglas de Integridad	32
Anexos (si es necesario)	32
Diagramas Adicionales	32
Referencias	32
Etapa 2: Persistencia de Datos con Backend	33
Introducción	33
Diseño de la Arquitectura de Backend	33
Elección de la Base de Datos	33
Implementación del Backend	34
Conexión a la Base de Datos	34
Pruebas del Backend	34
Etapa 3: Consumo de Datos y Desarrollo Frontend	35
Introducción	35
Creación de la Interfaz de Usuario (UI)	36
Programación Frontend con JavaScript (JS)	36
Consumo de Datos desde el Backend	37
Interacción Usuario-Interfaz	37
Pruebas y Depuración del Frontend	37

Implementación de la Lógica de Negocio en el Frontend	38
Integración con el Backend	38

Etapa 1 Diseño de la Aplicación y Análisis de Requisitos

Introducción

Propósito del Documento

El presente documento tiene como finalidad documentar el proceso de diseño, análisis e implementación de software de tipo educativo, comercial, OVA, componente o módulo de aplicaciones. Se divide en tres etapas para facilitar el entendimiento y aplicación a gran escala en la asignatura de diseño de software.

Etapa 1 Diseño de la Aplicación y Análisis de Requisitos

Esta etapa cumple la tarea de recoger todas las competencias desarrolladas en todas las áreas de formación del currículo de la licenciatura en Informática y Medios Audiovisuales y ponerlas a prueba en el diseño y análisis de un producto educativo que se base en las teorías de aprendizaje estudiadas, articule las estrategias de enseñanza con uso de TIC y genere innovaciones en educación con productos interactivos que revelen una verdadera naturaleza educativa. Estos productos deben aprovechar las fortalezas adquiridas en las áreas de tecnología e informática, técnicas y herramientas, medios audiovisuales y programación y sistemas, para generar productos software interactivos que permitan a los usuarios disfrutar de lo que aprenden, a su propio ritmo. Todo esto en el marco de un proceso metodológico (metodologías de desarrollo de software como MODESEC, SEMLI, etc.) que aproveche lo aprendido en la línea de gestión y lo enriquezca con elementos de la Ingeniería de Software.

Etapa 2: Persistencia de Datos con Backend – Servidor

En la etapa 2 se continúa con los lineamientos de la etapa 1, para seguir adicionando elementos de diseño e implementación de software, enfocados en el desarrollo de APIs, servidores o microservicios que permitan soportar aplicaciones cliente del software educativo; en este sentido, el curso presenta los conceptos de los sistemas de bases de datos, su diseño lógico, la organización de los sistemas manejadores de bases de datos, los lenguaje de definición de datos y el lenguaje de manipulación de datos SQL y NoSQL; de tal manera que los estudiantes adquieran las competencias para analizar, diseñar y desarrollar aplicaciones para gestionar y almacenar grandes cantidades de datos, mediante el uso de técnicas adecuadas como el diseño y modelo lógico y físico de base datos, manejo de los sistemas de gestión de bases de datos, álgebra relacional, dominio del lenguaje SQL como herramienta de consulta, tecnología cliente / servidor; igualmente, se definirán los elementos necesarios para el acceso a dichas bases de datos, como la creación del servidor API, utilizando tecnologías de vanguardia como node.js, express, Nest.js, Spring entre otros; para, finalmente converger en el despliegue de la API utilizando servicios de hospedaje en la nube, preferiblemente gratuitos. También podrá implementar servidores o API's con inteligencia artificial o en su defecto crear una nueva capa que consuma y transforme los datos obtenidos de la IA. El desarrollo del curso se trabajará por proyectos de trabajo colaborativo que serán evaluados de múltiples maneras, teniendo en cuenta más el proceso que el resultado.

Etapa 3: Consumo de Datos y Desarrollo Frontend – Cliente

La etapa 3 el estudiante está en capacidad de establecer la mejor elección de herramientas de consumo de datos y técnicas en aras de lograr el mejor producto a nivel de software o hardware acorde a los requerimientos funcionales y no funcionales del problema a solucionar. En este punto el estudiante puede consumir los datos a través de un cliente que puede ser una aplicación de celular, una aplicación de escritorio, una página web, IoT(internet de las cosas) o incluso, artefactos tecnológicos. El diseño gráfico es de los requisitos esenciales en la capa de presentación, por lo tanto, se requieren los cursos de diseño gráficos vistos previamente. Los elementos anteriores nos permiten elegir el paradigma y tecnología para desarrollar nuestras aplicaciones, teniendo en cuenta que podríamos desarrollar aplicaciones de tipo cliente.

Alcance del Proyecto

El proyecto consiste en desarrollar un componente de grabación de vídeo, permitiendo la captura de pantalla completa o por zonas, grabación de audio del sistema o micrófono, con opciones de calidad y resolución ajustables, así como herramientas básicas de edición como recorte de video a futuro podría incluirse el envío de vídeos por correo electrónico, edición avanzada, visualización de datos de video y espacio disponible en el sistema; con el fin de mejorar la comunicación, el aprendizaje, la accesibilidad, la colaboración y la documentación de los usuarios.

El alcance del proyecto puede estar sujeto a ajustes según las necesidades y requerimientos adicionales que surjan durante el desarrollo. La flexibilidad y la comunicación efectiva serán clave para el éxito del proyecto. Cabe resaltar que la implementación del componente se llevará a cabo en tres fases durante tres semestres académicos. La primera fase se enfoca en la documentación detallada del componente y sus funcionalidades.

Para esta primera versión se trabajará las siguientes funcionalidades:

- Grabar pantalla completa
- Grabar pantalla por zona
- Capturar audio
- Iniciar grabación
- Pausar Grabación
- Finalizar grabación
- Guardar grabación.

Funcionalidades Futuras

- Descargar grabación
- Buscar Video
- Lista de videos
- Datos de video

Definiciones y Acrónimos

Grabación de pantalla: Proceso de capturar y guardar en formato de vídeo lo que se muestra en la pantalla de una computadora o dispositivo.

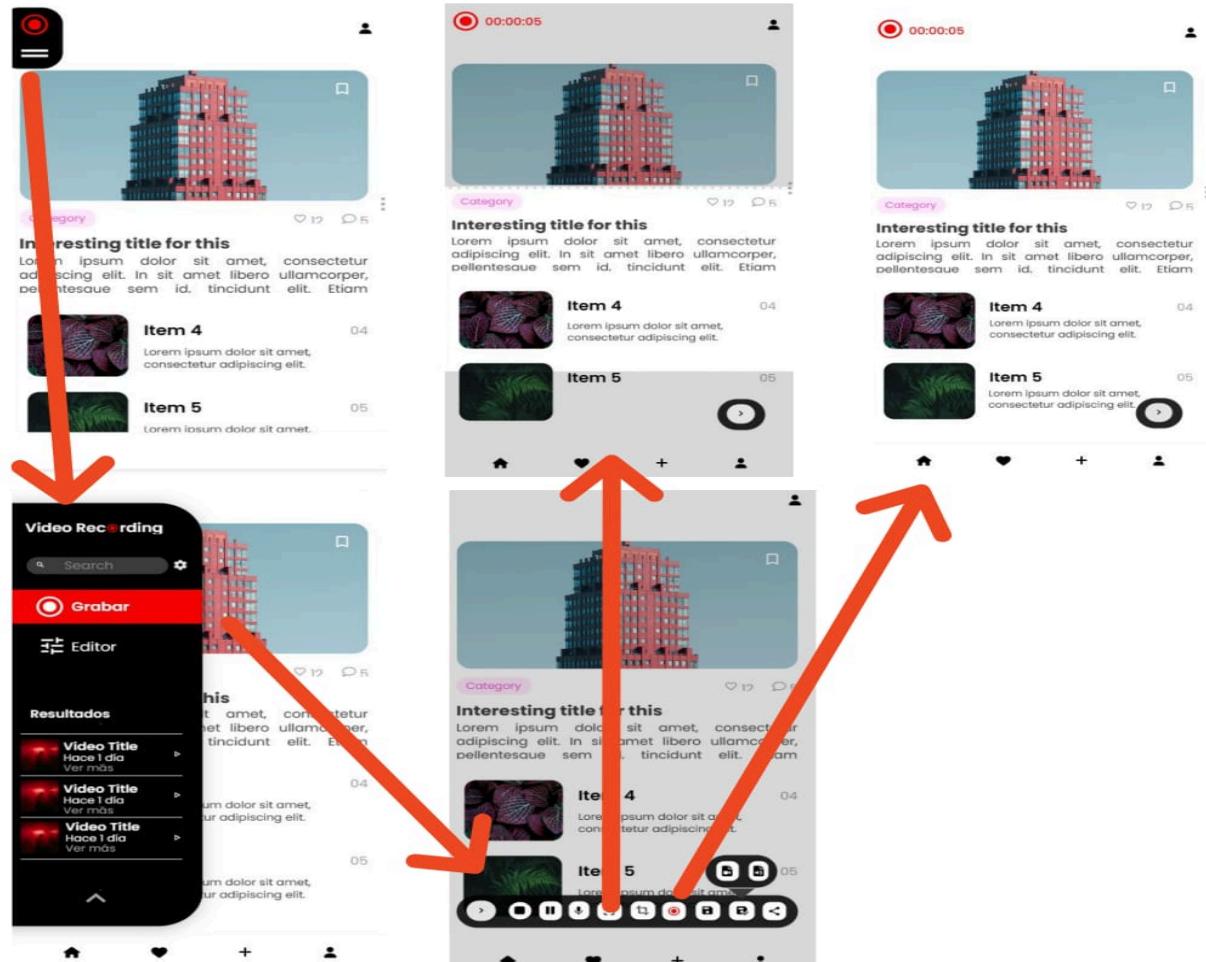
- 1. Resolución de vídeo:** La cantidad de píxeles que componen una imagen de vídeo, determinada por el número de píxeles en anchura y altura (por ejemplo, 1920x1080 píxeles para resolución Full HD).
- 2. Calidad de vídeo:** La medida de la claridad y fidelidad de una grabación de vídeo, que puede estar influenciada por factores como la resolución, la tasa de bits y el códec utilizado.
- 3. Audio del sistema:** El sonido generado por el sistema operativo y las aplicaciones en ejecución en el dispositivo.
- 4. Micrófono local:** Dispositivo de entrada de audio que captura sonido del entorno cercano al dispositivo, como la voz del usuario.
- 5. Formato de vídeo compatible:** Formato de archivo de vídeo que puede ser reproducido por una amplia gama de reproductores y dispositivos, como MP4 (MPEG-4), AVI (Audio Video Interleave), etc.
- 6. Organización por fecha y hora:** Agrupar las grabaciones de pantalla según el momento en que fueron realizadas, facilitando la búsqueda y gestión.
- 7. Edición de vídeo:** Proceso de modificar y mejorar el contenido de vídeo, que puede incluir recorte, unión, eliminación de partes, añadir títulos, créditos y marcas de agua, entre otros.
- 8. Brillo y contraste:** Parámetros que afectan la luminosidad y diferencia entre las partes más claras y oscuras de una imagen de vídeo.
- 9. Volumen de las grabaciones:** Nivel de sonido de las grabaciones de vídeo.
- 10. Inicio, pausa y detención de la grabación:** Acciones para iniciar, pausar y detener el proceso de grabación de pantalla.
- 11. Duración de la grabación:** El tiempo total que ha estado activa la grabación.
- 12. Peso del archivo:** El tamaño en bytes del archivo de vídeo grabado.
- 13. Espacio disponible en disco:** La cantidad de almacenamiento libre en el dispositivo donde se guardarán las grabaciones.
- 14. Opciones de grabación:** Configuraciones que el usuario puede ajustar antes de iniciar el proceso de grabación, como resolución, calidad, fuente de audio, entre otros.

Descripción General

Objetivos del Sistema

El objetivo principal de este componente dentro del software educativo es potenciar la capacidad de los usuarios para capturar y compartir contenido visual de sus pantallas de manera eficiente. Este componente busca fomentar la creatividad y mejorar la comunicación en el ámbito educativo, permitiendo a educadores y estudiantes crear y compartir material visual de alta calidad de forma sencilla. Facilita la colaboración en proyectos educativos al posibilitar la edición y compartición rápida de contenido entre usuarios

Interfaz



Conceptos de las entidades

Archivos

Los archivos representan los documentos multimedia que el sistema genera o almacena durante las sesiones de grabación. Cada archivo está vinculado a una grabación específica y puede contener información adicional para facilitar su identificación y uso posterior.

Grabaciones

Las grabaciones son el núcleo del sistema. Representan las sesiones en las que los usuarios capturan vídeo y audio desde pantallas configuradas previamente..

Audios

Los audios complementan las grabaciones al proporcionar pistas de sonido independientes que pueden ser procesadas o combinadas con los archivos de vídeo.

Pantallas

Las pantallas definen las configuraciones de visualización que el sistema captura durante una grabación. Pueden abarcar áreas completas o específicas del monitor del usuario.

Funcionalidad General

El sistema está diseñado para capturar y gestionar grabaciones de vídeo y audio de forma eficiente. A continuación, se detalla el flujo principal del sistema:

1. Inicio de sesión de grabación:

El usuario configura las pantallas y define el área que desea capturar. También puede elegir si desea incluir audio.

2. Proceso de grabación:

Durante la grabación, el sistema almacena los datos en tiempo real y los segmenta en archivos separados (video y audio). Las pantallas configuradas determinan las áreas específicas de captura.

3. Almacenamiento:

Al finalizar la grabación, los archivos generados se organizan en el sistema de almacenamiento, con rutas únicas que permiten su fácil recuperación. Cada archivo incluye metadatos para su clasificación (nombre, tamaño, descripción, etc.).

4. Gestión de grabaciones:

El sistema permite al usuario realizar las siguientes acciones:

- Consultar todas las grabaciones disponibles.
- Visualizar información detallada de una grabación específica.
- Editar datos asociados a la grabación (como la descripción o etiquetas).
- Eliminar grabaciones o archivos asociados

Usuarios del Sistema

Los siguientes usuarios pueden interactuar con el sistema dependiendo de las funcionalidades

Funcionalidades	Admin	Docente	Docente invitado	Alumno
Grabar pantalla completa en tiempo real	✓	✓	✓	✓
Grabar pantalla por zona en tiempo real	✓	✓	✓	✓
Capturar audio en tiempo real	✓	✓	✓	✓
Iniciar grabación	✓	✓	✓	✓
Pausar grabación	✓	✓	✓	✓
Finalizar grabación	✓	✓	✓	✓
Guarda grabación	✓	✓	✓	✓
Descargar grabación	✓	✓	✓	✓
Buscar video	✓	✓	✓	✓
Listar videos	✓	✓	✓	✓
Datos del video	✓	✓	✓	✓

Restricciones

Restricciones Técnicas

- **Formato de Archivos:**
 - a. Los videos deben ser almacenados en formato MP4 para asegurar compatibilidad con la mayoría de los reproductores.
 - b. Los audios deben utilizar formatos como MP3 o WAV para garantizar calidad y facilidad de uso.
- **Resolución de Pantallas:**
 - a. El sistema soportará grabaciones de hasta 4K, pero la resolución dependerá de la capacidad del dispositivo cliente.
- **Almacenamiento:**
 - a. Cada archivo generado no puede exceder los 5 GB debido a restricciones de almacenamiento en el servidor.
 - b. Los archivos deben ser almacenados en rutas únicas dentro de la estructura del sistema.
- **Duración Máxima de Grabaciones:**
 - a. Las grabaciones no podrán superar las 2 horas por sesión debido a limitaciones de procesamiento y espacio.
- **Conexión a Internet:**
 - a. Se requiere una conexión estable a internet para la sincronización de datos y almacenamiento en la nube.

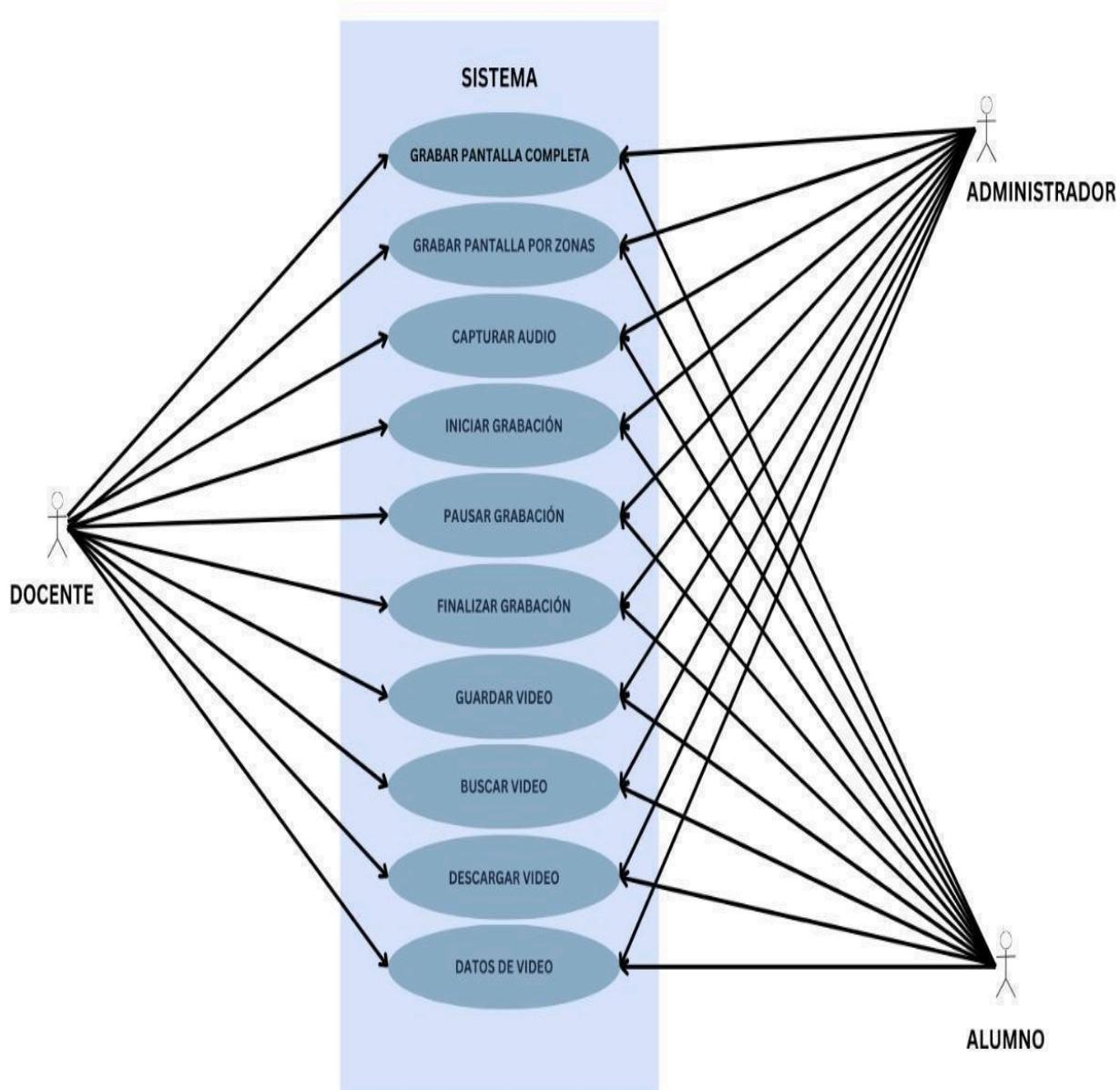
Restricciones Funcionales

- **Simultaneidad:**
 - a. Un usuario no puede iniciar más de una grabación a la vez.
- **Acceso a Pantallas:**
 - a. Solo se permite grabar pantallas completas o áreas específicas configuradas previamente. No se puede capturar múltiples áreas simultáneamente.
- **Integridad de Datos:**
 - a. Todos los archivos generados deben incluir metadatos obligatorios, como nombre, ruta, tamaño y descripción.
- **Compatibilidad:**
 - a. El sistema debe ser accesible desde navegadores modernos como Google Chrome, Mozilla Firefox y Microsoft Edge, con versiones actualizadas.

Requisitos Funcionales

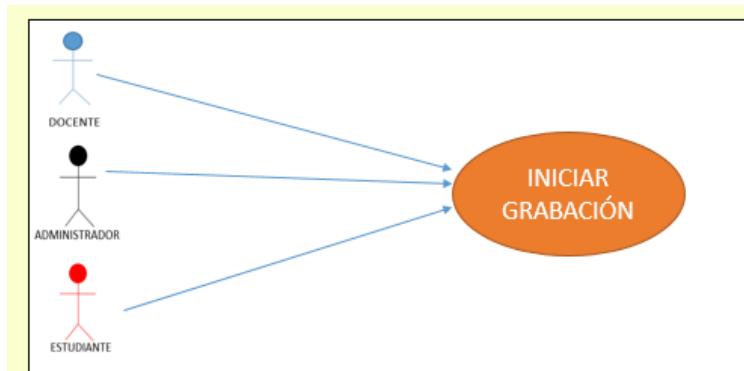
Casos de Uso

- Grabar pantalla completa
- Grabar pantalla por zona
- Capturar audio
- Iniciar grabación
- Pausar Grabación
- Finalizar grabación
- Guardar grabación
- Descargar grabación
- Buscar Video
- Lista de videos
- Datos de video



Descripción detallada de cada caso de uso

CU-1



Urgencia: 5

Esfuerzo: 4

SOIG: Selecciona la opción iniciar grabación

IG: Iniciar la grabación

IIG: Indica que inicio la grabación

Iniciar grabación
Flujo:Iniciar grabación
Prueba: Variable iniciar grabación .

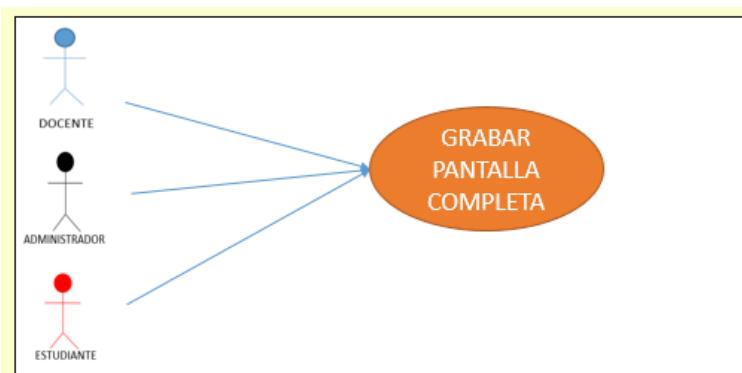
Iniciar grabación
Flujo: SOIG, IG, IIG

CASO No. 1 Iniciar grabación

ID:	CU-1	
Nombre	Iniciar grabación	
Actores	Docente – Docente invitado -Administrador-Alumno	
Objetivo	Permitir iniciar una grabación	
Urgencia	5	
Esfuerzo	4	
Precondiciones	Estar autenticado, permisos para grabar audio y video otorgados	
Flujo Normal	Actores	Sistema
	Selecciona la opción “Iniciar grabación”	
		Inicia la grabación
		Indica que inició la grabación

Flujo Alternativo 1	Selecciona la opción "Iniciar grabación"	
		Muestra Advertencia (Falta de espacio, puede grabar, pero no puede guardarse)
	Da la opción de liberar espacio	

CU-2



Urgencia: 5

Esfuerzo: 3

IGPC: Iniciar Grabación de Pantalla Completa

CU-1: Se efectúa el caso de uso 1

VG: Visualiza la grabación

DG: Detiene la grabación

OGG: Da la opción de guardar grabación

SGG: Selección guardar la grabación

MOAG: Muestra opciones de almacenamiento para la grabación

SGG: Selección donde se guardara la grabación

PGG: Pulsa guardar la grabación

GG: Guarda la grabación

Grabar pantalla completa

Flujo: Grabar pantalla completa

Prueba: variable de grabar pantalla completa

GRABAR PANTALLA COMPLETA

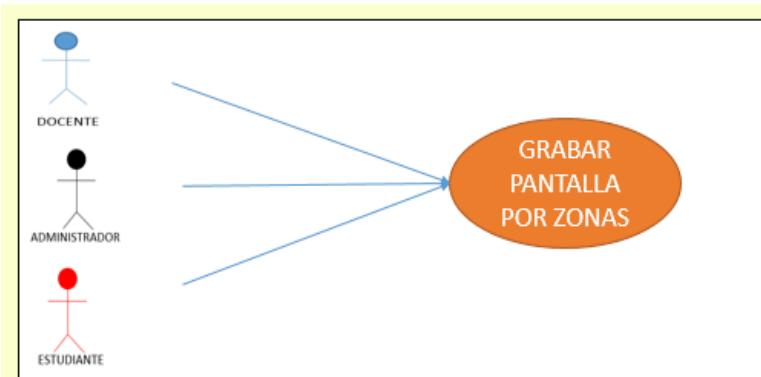
Flujo: IGPC, CU-1, VG, DG, OGG, SGG, MOAG, SGG, PGG, GG

CASO No. 2 Grabar Pantalla completa

ID:	CU-2
Nombre	Grabar Pantalla Completa
Actores	Docente – Docente invitado -Administrador-Alumno
Objetivo	Permitir Grabar pantalla completa
Urgencia	5
Esfuerzo	3

Precondiciones	Estar autenticado, deben haber sido otorgados los permisos de grabar audio y video	
Flujo Normal	Actores	Sistema
	Iniciar grabación de pantalla completa	
		Se Efectúa el CU-1
	Visualiza la grabación	
	Detiene la grabación	
		Da opción de guardar la grabación
	Selección opción guardar la grabación	
		Muestra opciones de almacenamiento para la grabación
	Selecciona donde se guardará la grabación	
	Pulsa Guardar la grabación	
		Guarda la grabación
Flujo Alternativo 1	Inicia la grabación	
		Se Efectúa FLUJO 1 ALTERNATIVO CU1
	Da la opción de liberar espacio	

CU-3



Urgencia: 4
Esfuerzo: 3

SOGZ: Selecciona la opción grabar por zonas
MBDZG: Muestra botón de selección para definir la zona a grabar
DZ: Define la zona
IGZD: Inicia la grabación de la zona definida
IIG: Indica que inició la grabación
DG: Detiene la grabación
CU- 2: Se efectúa el CU-2
VDG: Visualiza datos de la grabación

Grabar pantalla por zonas
Flujo: Grabar pantalla por zonas
Prueba: Seleccionar zona

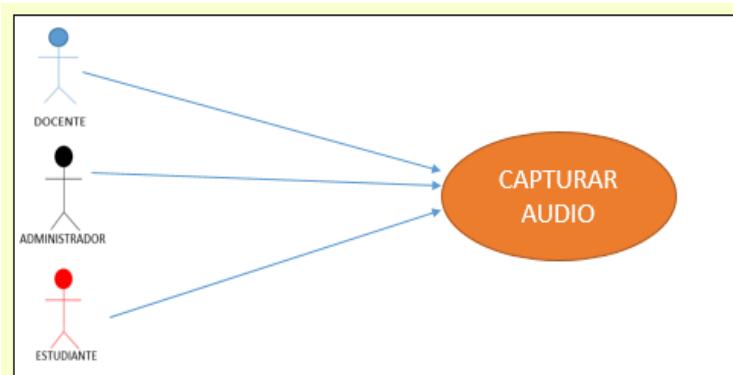
Grabar pantalla por zonas
Flujo: SOGZ, MBDZG, DZ, IGZD, IIG, DG, CU- 2, VDG

CASO No. 3 Grabar Pantalla por zonas

ID:	CU-3	
Nombre	Grabar Pantalla por zonas	
Actores	Docente – Docente invitado -Administrador-Alumno	
Objetivo	Permitir Grabar zonas específicas de la pantalla	
Urgencia	4	
Esfuerzo	3	
Precondiciones	Estar autenticado, haber definido la zona a grabar, permisos de grabar audio y video otorgados	
Flujo Normal	Actores	Sistema
	Selecciona la opción “Grabar por zonas”	

		Muestra botón de selección para definir la zona a grabar
	Define la zona	
		Inicia la grabación de la zona definida
		Indica que inició la grabación
	Detiene la grabación	
		Se Efectúa CU 2
	Visualiza datos de guardado	
Flujo Alternativo 1	Define la zona a grabar	
		Muestra error (Zona fuera de límite)
	Redefine la zona	

CU-4



Urgencia:4
Esfuerzo: 2

SOCA: Selecciona la opción Capturar audio
IMGA: Inicia módulo de grabación de audio
MV: Muestra módulo de volumen
DG: Detiene la grabación
OGAG: Da la opción de guardar audio grabado
SGA: Selecciona guardar audio
MOA: Muestra opciones de almacenamiento
SLA: Selecciona lugar de almacenamiento
GA: Guarda el audio

Capturar audio
Flujo: Capturar audio
Prueba: Audio grabado.

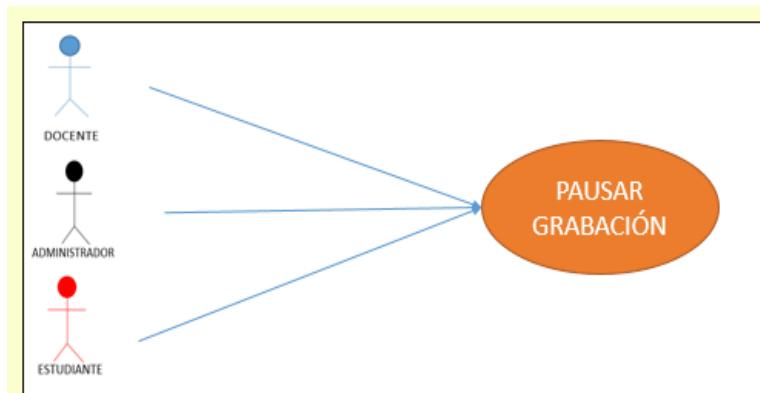
Capturar Audio
Flujo: SOCA, IMGA, MV, DG, OGAG, SGA, MOA, SLA, GA

CASO No. 4 Capturar audio

ID:	CU-4	
Nombre	Capturar audio	
Actores	Docente - Docente invitado -Administrador-Alumno	
Objetivo	Permitir capturar solo el audio	
Urgencia	4	
Esfuerzo	2	
Precondiciones	Estar autenticado, permisos para grabar audio otorgados	
Flujo Normal	Actores	Sistema
	Selecciona la opción "Capturar audio"	
		Inicia módulo de grabación de audio
		Muestra módulo de volumen
	Detiene la grabación	
		Da la opción de guardar audio grabado
	Selecciona guardar audio	
		Muestra opciones de almacenamiento
	Selecciona lugar de almacenamiento	
Flujo Alternativo 1	Guarda el audio	
	Selecciona la opción de capturar audio	

		Muestra error (Micrófono no disponible)
	Configura micrófono, Intenta de nuevo	

CU- 5



Urgencia: 3
Esfuerzo: 2

SOPG: Selecciona la opción pausar grabación
PG: Pausa la grabación
IGP: Indica que la grabación ha sido pausada
IGN: Inicia grabación nuevamente
SG: Sigue grabando por donde iba

Flujo: Pausar grabación
Prueba: Variable pausar grabación.

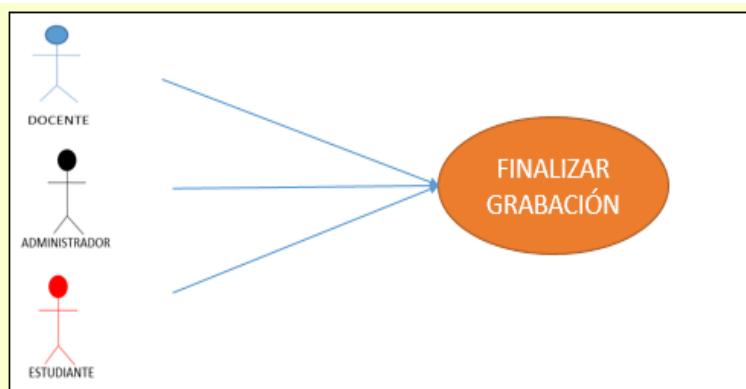
Pausar grabación
Flujo: SOPG, PG, IGP, IGN, SG

CASO No. 5 Pausar grabación

ID:	CU-5	
Nombre	Pausar grabación	
Actores	Docente - Docente invitado -Administrador-Alumno	
Objetivo	Permitir pausar una grabación en curso	
Urgencia	3	
Esfuerzo	2	
Precondiciones	Estar autenticado, una grabación en curso	
Flujo Normal	Actores	Sistema
	Selecciona la opción “Pausar grabación”	
		Pausa la grabación
		Indica que la grabación ha sido pausada
	Inicia grabación nuevamente	
		Sigue grabando por donde iba
Flujo Alternativo 1	Selecciona la opción “Pausar grabación”	

		Muestra advertencia (Grabación avanzada, desea pausar?)
	Cancela la operación	

CU- 6



Urgencia: 5
Esfuerzo: 2

SOFG: Selecciona la opción Finalizar grabación
MOA: Muestra Opción de almacenamiento
SLA: Selecciona lugar de almacenamiento
SGG: Selecciona guardar grabación
GG: Guarda la grabación

Finalizar grabación
Flujo: Finalizar grabación
Prueba: Video finalizado

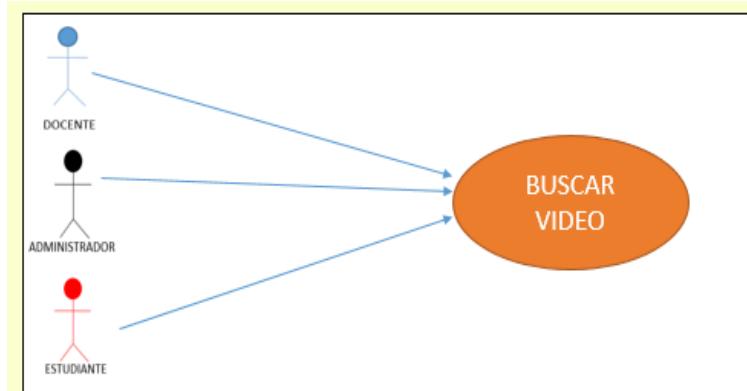
Finalizar grabación
Flujo: SOFG, MOA, SLA, SGG, GG

CASO No. 6 Finalizar grabación

ID:	CU-6	
Nombre	Finalizar grabación	
Actores	Docente – Docente invitado -Administrador-Alumno	
Objetivo	Permitir finalizar una grabación en curso	
Urgencia	5	
Esfuerzo	2	
Precondiciones	Estar autenticado, una grabación en curso	
Flujo Normal	Actores	Sistema
	Selecciona la opción “Finalizar grabación”	
		Muestra Opción de almacenamiento
	Selecciona lugar de almacenamiento	
	Selecciona guardar grabación	Guarda la grabación
Flujo Alternativo 1	Selecciona la opción “Finalizar grabación”	

		Muestra error. Problemas de almacenamiento)
	Cancela la operación	

CU- 7



Urgencia: 3

Esfuerzo: 2

SOBV: Selecciona la opción Buscar videos

ICB: Indica los criterios de la búsqueda

MR: Muestra los resultados

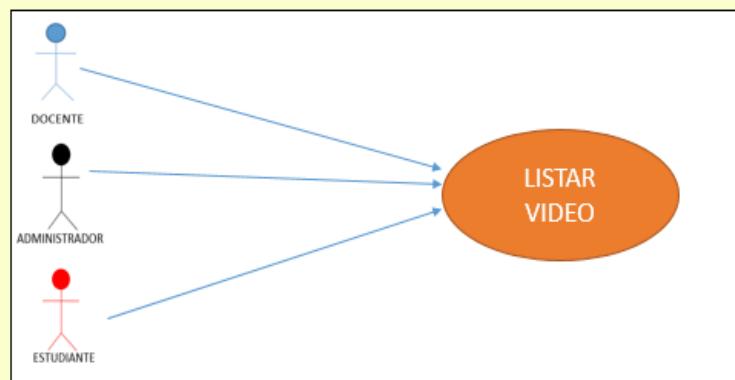
Buscar video
Flujo: Buscar video
Prueba: Variable buscar video.

Buscar video
Flujo: SOBV, ICB, MR

CASO No. 7 Buscar video

ID:	CU-7	
Nombre	Buscar video	
Actores	Docente – Docente invitado -Administrador-Alumno	
Objetivo	Permitir buscar videos en el sistema	
Urgencia	3	
Esfuerzo	2	
Precondiciones	Estar autenticado	
Flujo Normal	Actores	Sistema
	Selecciona la opción “Buscar videos”	
	Indica los criterios de la búsqueda	
		Muestra los resultados
Flujo Alternativo 1	Selecciona la opción “Buscar videos”	
	Define los criterios de búsqueda	
		Muestra error (No hay resultados)
	Cancela la operación	

CU- 8



Urgencia: 3

Esfuerzo: 2

SOLV: Selecciona la opción listar videos

MLVD: Muestra lista de videos disponibles

SV: Selección video

VV: Visualiza video

Listar video

Flujo: Listar video

Prueba: Lista de videos.

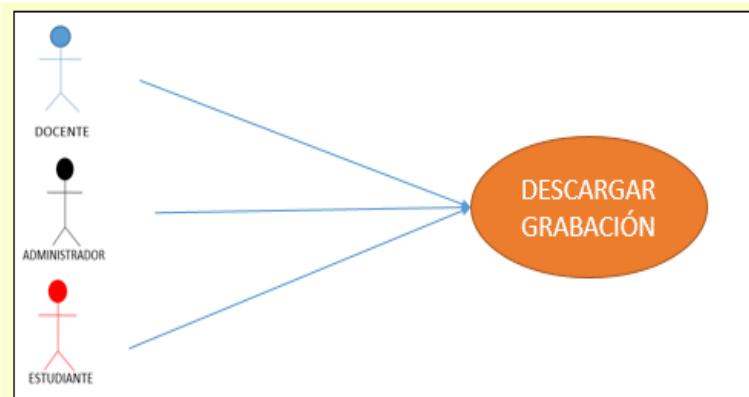
Listar video

Flujo: SOLV, MLVD, SV, VV

CASO No. 8 Listar video

ID:	CU-8	
Nombre	Listar video	
Actores	Docente – Docente invitado -Administrador-Alumno	
Objetivo	Permitir buscar videos en el sistema	
Urgencia	3	
Esfuerzo	2	
precondiciones	Estar autenticado	
Flujo Normal	Actores	Sistema
	Selecciona la opción “listar videos”	
		Muestra lista de videos disponibles
	Selección video	
	Visualiza video	
Flujo Alternativo 1	Selecciona la opción “Listar videos”	
		Muestra error (Lista vacía)
	cancela la operación	

CU- 9



Urgencia: 4

Esfuerzo: 3

PLG: Presiona lista de grabaciones

MLGD: Muestra las grabaciones disponibles

SG: Selecciona la grabación

PAD: Prepara el archivo para descargar

SCU: Selecciona carpeta de ubicación

D: Descarga

VD: Visualiza descarga

Flujo: Descargar grabación

Prueba: Variable descargar grabación.

Descargar grabación

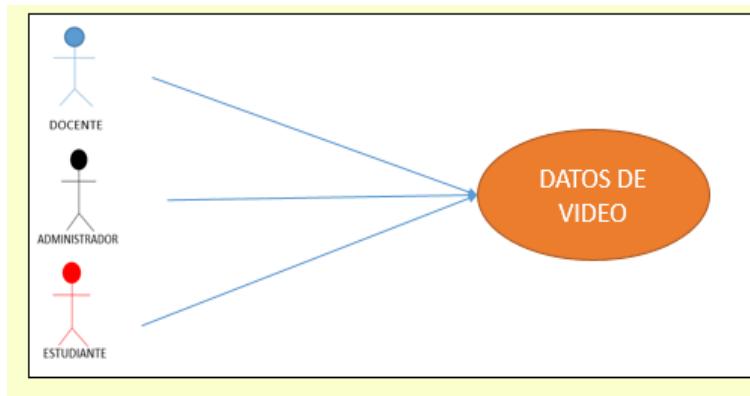
Flujo: PLG, MLGD, SG, PAD, SCU, D, VD

CASO No. 9 Descargar grabación

ID:	CU-9	
Nombre	Descargar grabación	
Actores	Docente - Docente invitado -Administrador-Alumno	
Objetivo	Permitir descargar una grabación	
Urgencia	4	
Esfuerzo	3	
precondiciones	Estar autenticado, tener una grabación disponible	
Flujo Normal	Actores	Sistema
	Presiona lista de grabaciones	Muestra las grabaciones disponibles
	Selecciona la grabación	Prepara el archivo para descargar
	Selecciona carpeta de ubicación	Descarga
	Visualiza Descarga	

Flujo Alternativo 1	Selecciona grabación para descargar	
		Muestra error (Problemas de conexión)
	Cancela la operación	

CU- 10



Urgencia: 3
Esfuerzo: 2

SOVDV: Selecciona la opción ver datos de video
MDVS: Muestra los datos de video seleccionado
VD: Visualiza datos

Flujo: Datos de video
Prueba: Variable datos de video.

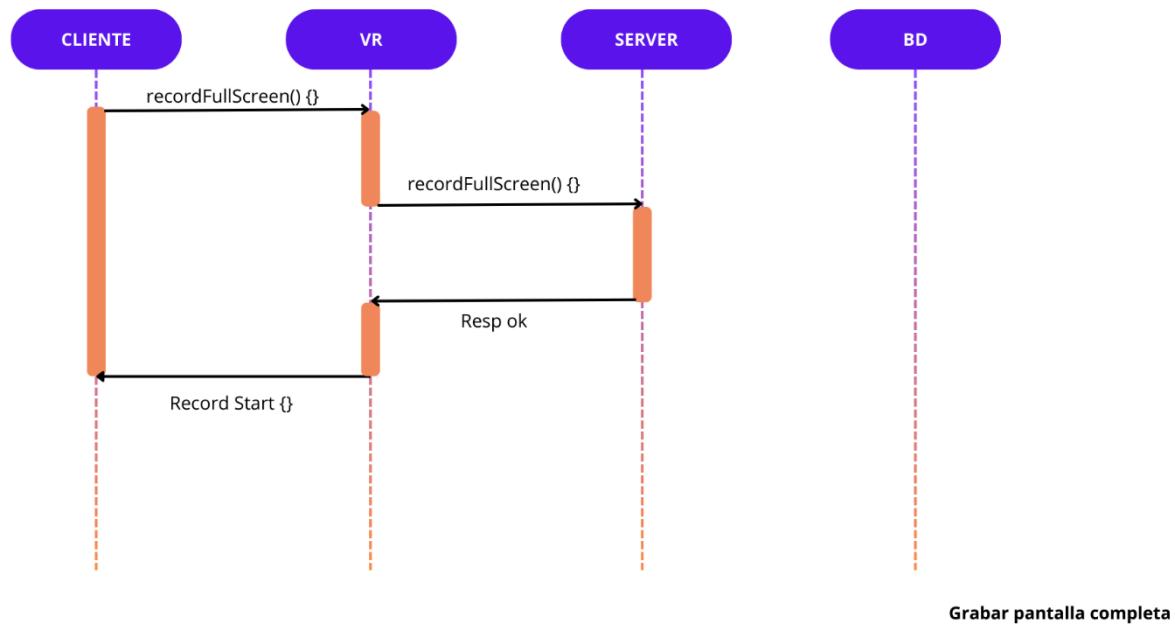
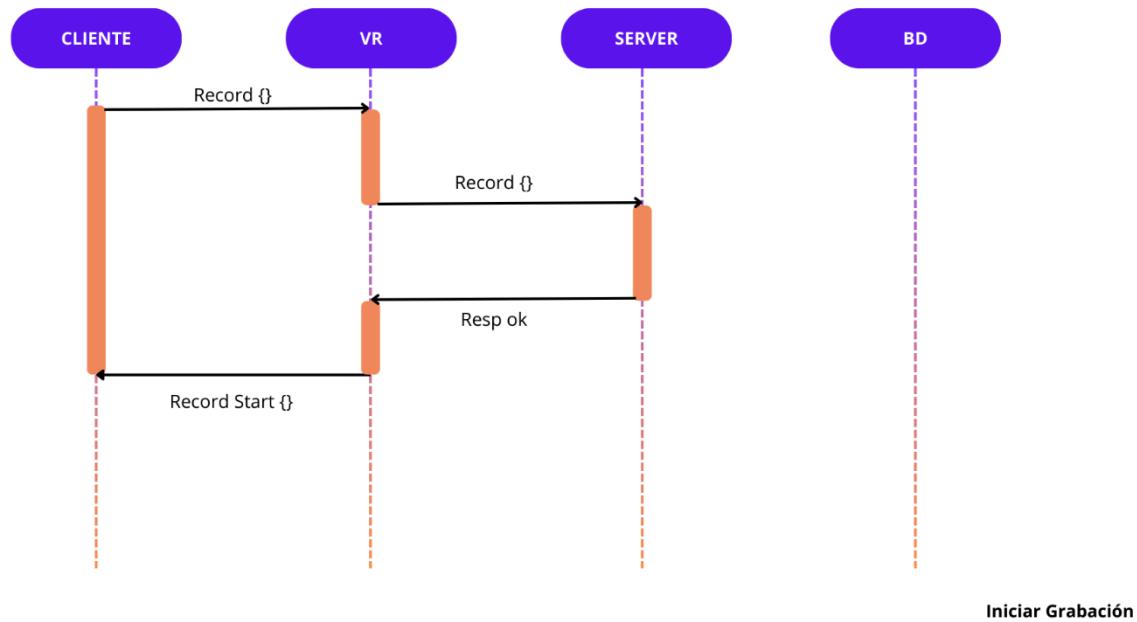
Datos de video
Flujo: SOVDV, MDVS, VD

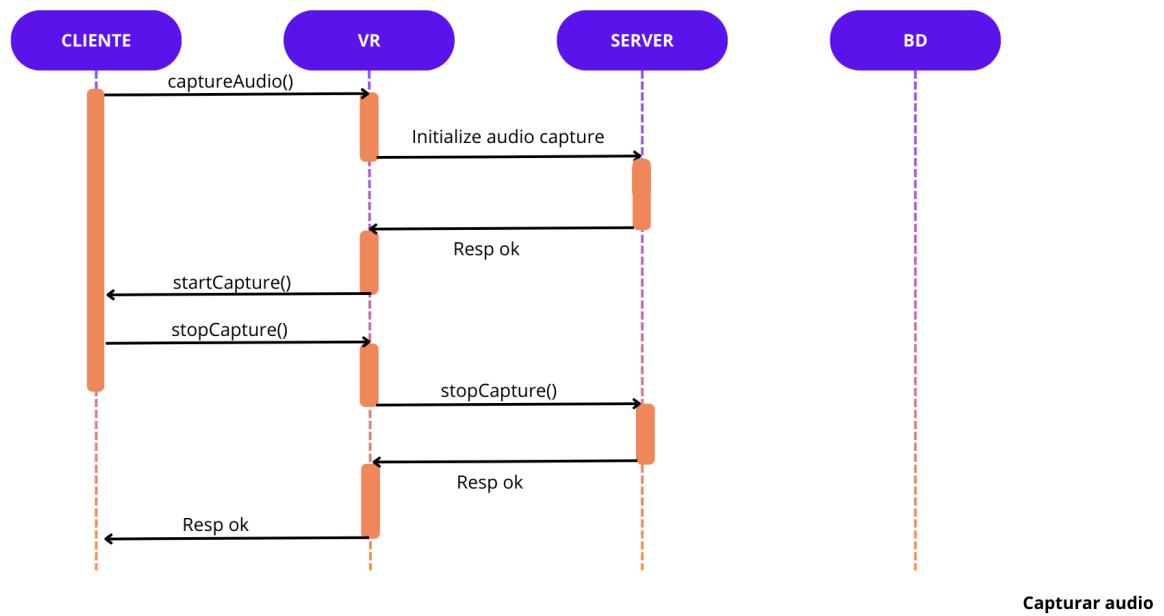
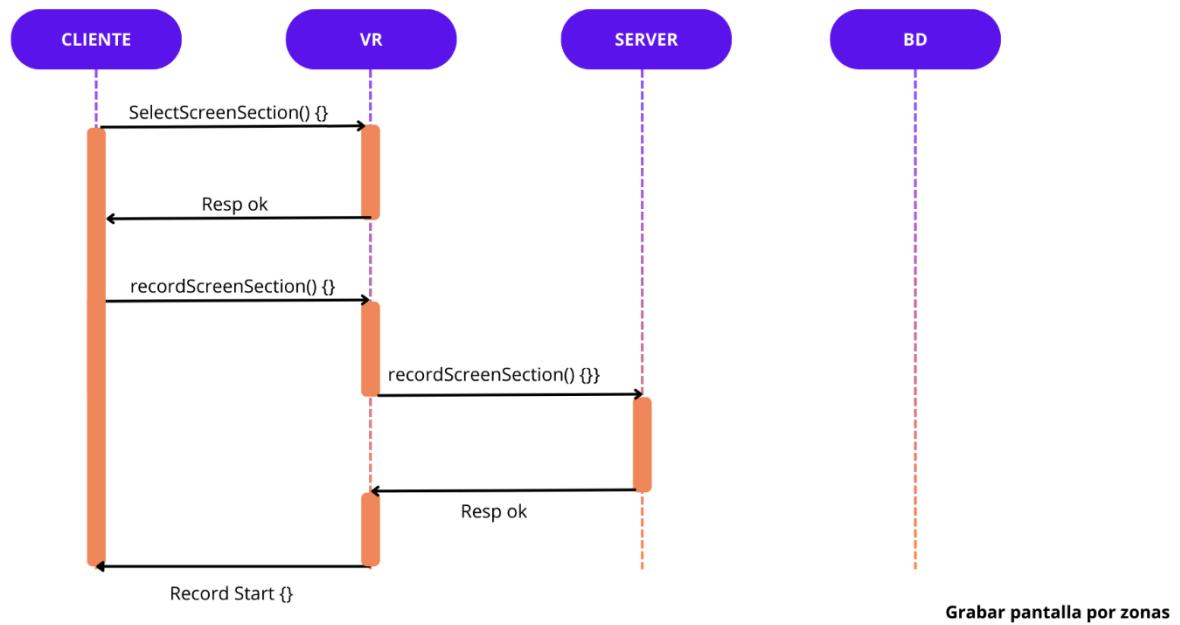
CASO No. 10 Datos de video

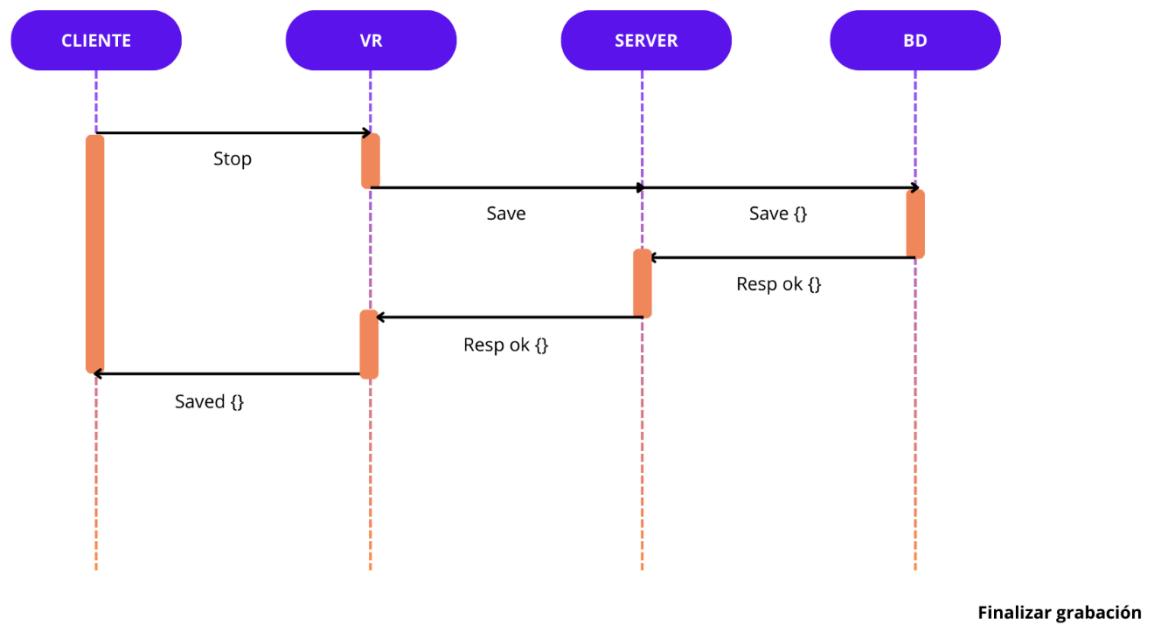
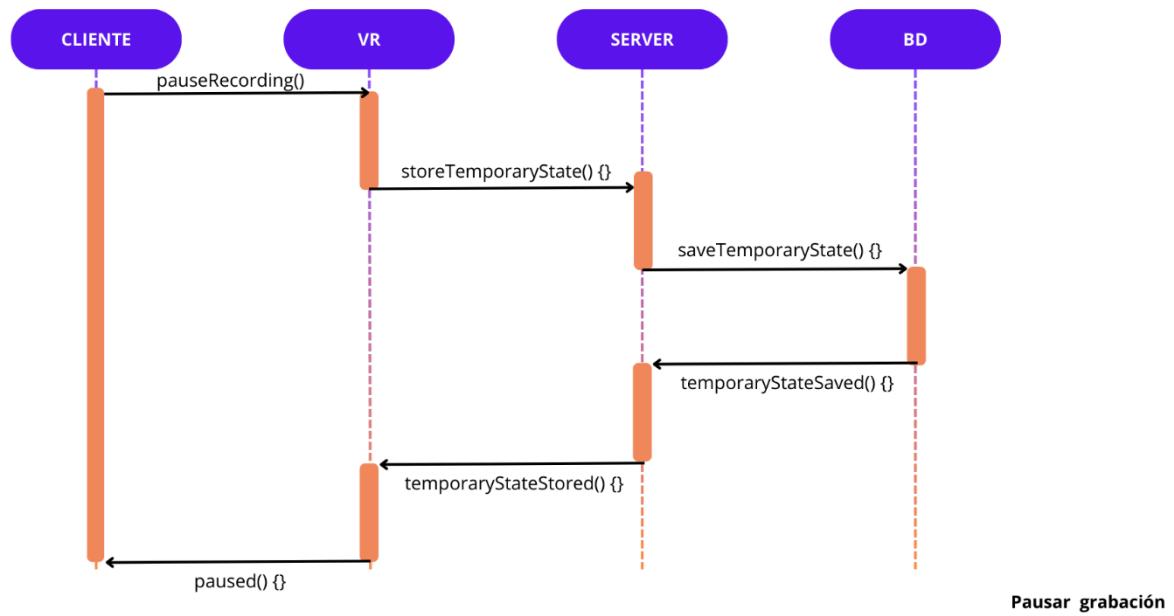
ID:	CU-10	
Nombre	Datos de video	
Actores	Docente – Docente invitado -Administrador-Alumno	
Objetivo	Permitir ver los datos de un video grabado	
Urgencia	3	
Esfuerzo	2	
precondiciones	Estar autenticado, tener una grabación disponible	
Flujo Normal	Actores	Sistema
	Selecciona opción de ver datos de video	
		Muestra los datos del video seleccionado
	Visualiza los datos	
Flujo Alternativo 1	Selecciona opción para ver datos de video	
		Muestra error (Datos no disponibles)

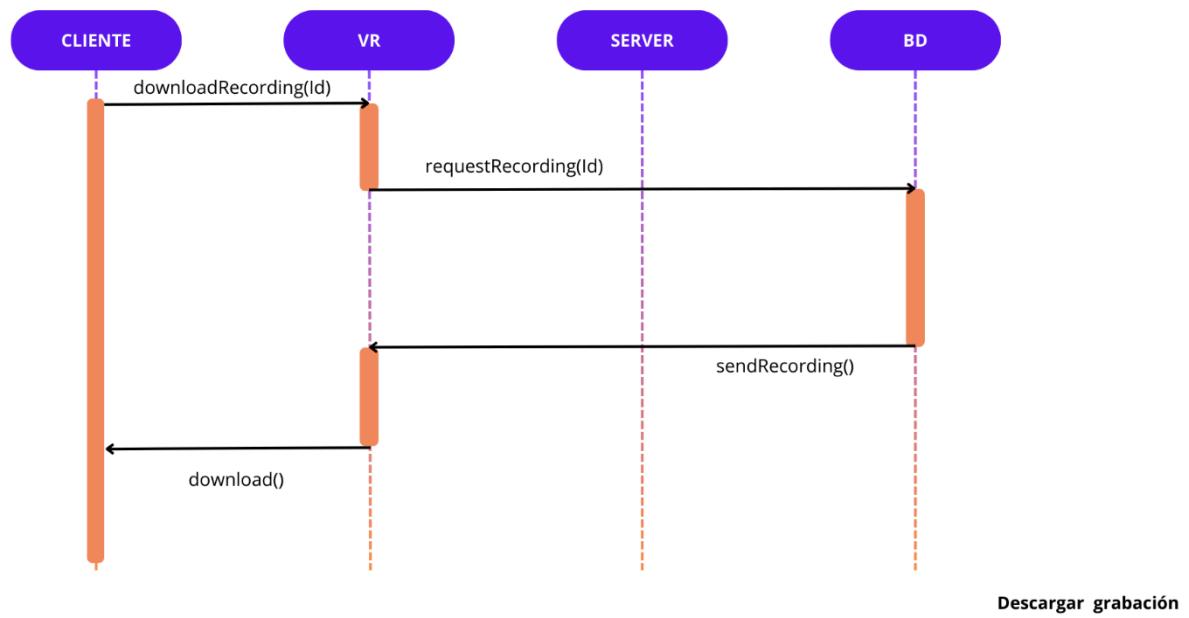
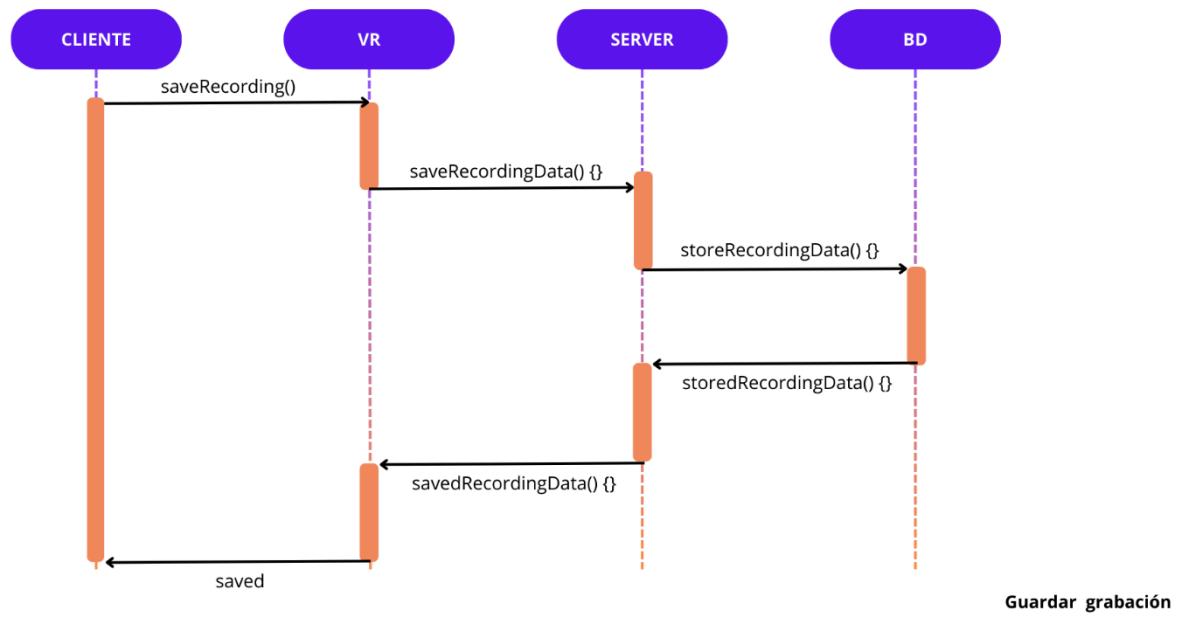
cancela la operación

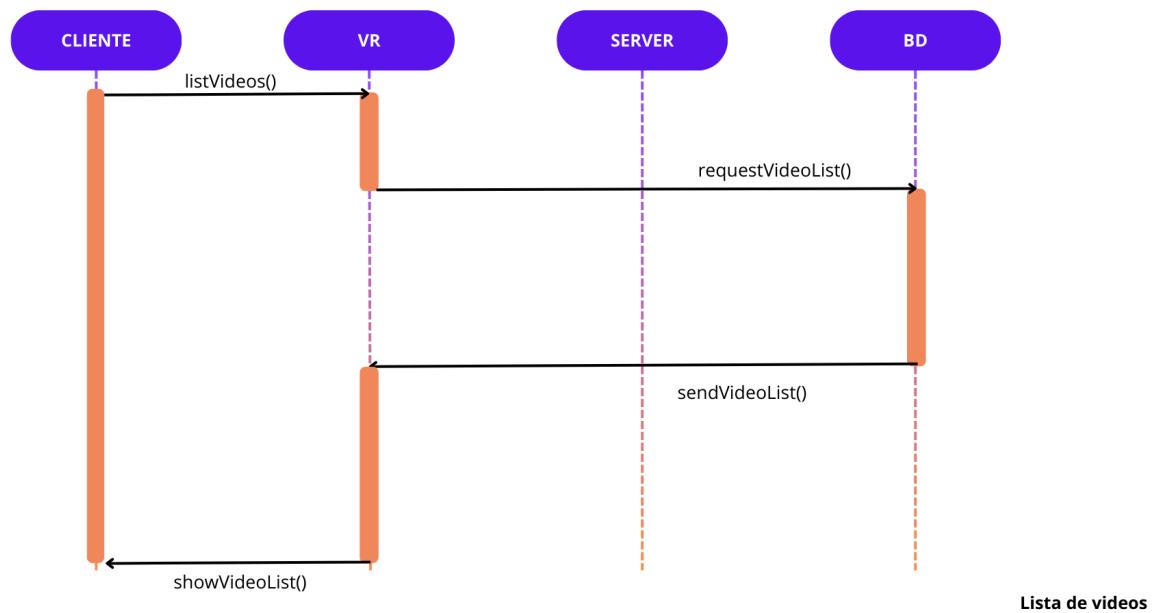
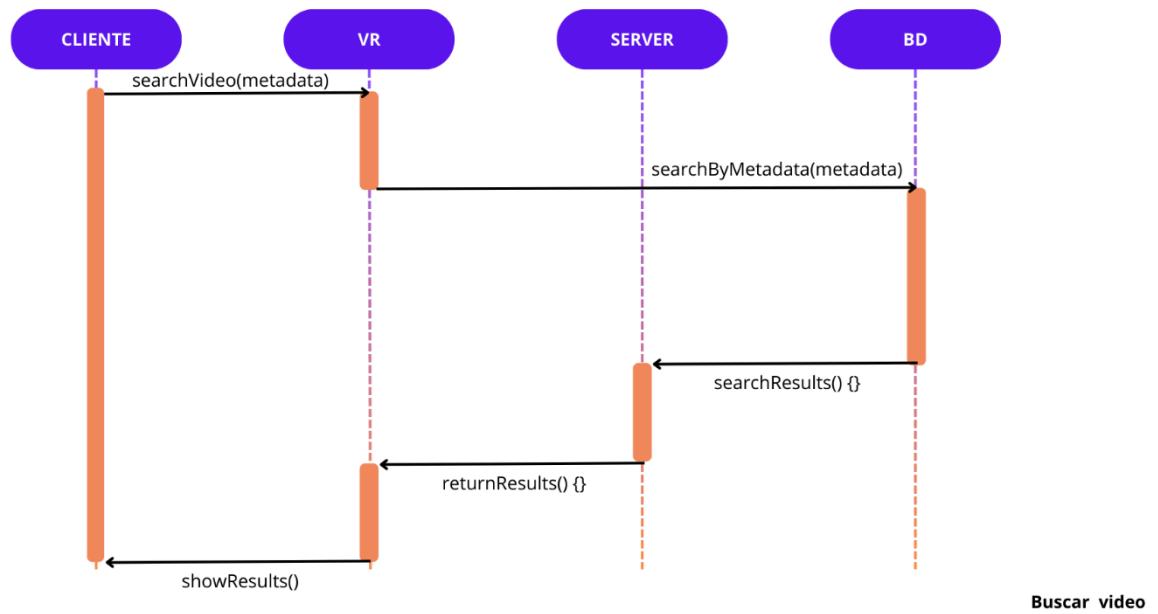
Diagramas de Secuencia











Prioridad de Requisitos

A partir del análisis de requerimientos, funcionalidades y el proceso de diseño thinking, se concreta la siguiente matriz de prioridad de requerimientos.

Para la interpretación se tiene en cuenta la siguiente escala con sus valores.

Eje de Urgencia:

- Obligatoria (5)
- Alta (4)
- Moderada (3)
- Menor (2)
- Baja (1)

Eje de Esfuerzo:

- Muy alto (5)
- Alto (4)
- Medio (3)
- Bajo (2)

	Urgencia					
Esfuerzo		1-Baja	2-Menor	3-Moderada	4-Alta	5-Obligatoria
	5-Muy alto	5	10	15	20	25
	4-Alto	4	8	12	16	20
	3-Medio	3	6	9	12	15
	2-Bajo	2	4	6	8	10
				CU-7 CU-5 CU-8	CU-4	CU-6

				CU-10		
1-Muy bajo	1	2	3	4	5	

Requisitos No Funcionales

Rendimiento:

- El sistema debe poder grabar pantallas a una tasa mínima de 30 cuadros por segundo (FPS) con resoluciones de hasta 1920x1080 (Full HD).
- El tiempo de respuesta del sistema para iniciar o pausar una grabación debe ser inferior a 2 segundos.
- Las herramientas de edición (cortar, unir, eliminar) deben ejecutarse en menos de 5 segundos para videos de hasta 10 minutos.

Escalabilidad:

- El sistema debe ser capaz de manejar múltiples usuarios concurrentes sin afectar el rendimiento, especialmente en la carga y descarga de grabaciones desde la nube.
- Se debe permitir la posibilidad de aumentar la capacidad de almacenamiento para los videos según la demanda.

Usabilidad:

- La interfaz de usuario debe ser intuitiva y accesible para usuarios con diferentes niveles de experiencia técnica.
- Las opciones clave (iniciar, pausar, detener grabación) deben ser accesibles con un máximo de dos clics o interacciones.
- El sistema debe ofrecer atajos de teclado para iniciar, pausar y detener grabaciones.

Seguridad:

- El sistema debe cumplir con los estándares de cifrado para la transmisión y almacenamiento de los videos en la nube (SSL/TLS).
- Solo los usuarios autenticados deben tener acceso a las grabaciones.
- Debe haber un control de acceso que garantice que solo ciertos usuarios puedan realizar acciones específicas (ej. administrador puede eliminar videos, usuarios estándar sólo pueden visualizar).

Compatibilidad:

- El sistema debe ser compatible con los principales navegadores web (Chrome, Firefox, Edge, Safari).
- Las grabaciones deben guardarse en formatos de video ampliamente aceptados como MP4 y AVI, que sean reproducibles en cualquier dispositivo moderno.

Portabilidad:

- La aplicación debe ser capaz de ejecutarse en diferentes sistemas operativos, incluyendo Windows, macOS y Linux.
- La arquitectura debe permitir su fácil integración con otros módulos o plataformas de administración de contenidos mediante APIs.

Fiabilidad:

- El sistema debe asegurar una tasa de éxito del 99.9% en la grabación y almacenamiento de videos.
- En caso de fallo de grabación, el sistema debe notificar al usuario y permitir la recuperación automática de archivos temporales.

Mantenibilidad:

- El sistema debe seguir principios de diseño modular para facilitar futuras actualizaciones de funcionalidades, tanto en la parte de frontend como backend.
- Debe contar con una documentación detallada para desarrolladores que incluya instrucciones de configuración, instalación y pruebas.

Disponibilidad:

- El sistema debe estar disponible al 99.5% del tiempo, con interrupciones permitidas solo para tareas de mantenimiento planificado.

Tolerancia a fallos:

- En caso de que el sistema se cierre inesperadamente, debe haber mecanismos para reanudar la grabación desde el último punto guardado o recuperar grabaciones no finalizadas.

Requisitos de Desempeño

1. **Velocidad:** Latencia máxima de 100 ms para la grabación y procesamiento de video; herramientas de edición deben procesar un video de 10 minutos en menos de 5 segundos.
2. **Uso de memoria:** No debe superar el 60% de la memoria disponible en grabaciones Full HD.
3. **Almacenamiento:** Videos de 10 minutos en Full HD no deben ocupar más de 1 GB.
4. **Tiempo de respuesta:** Iniciar o detener la grabación debe tardar menos de 2 segundos; exportar un video de 5 minutos debe tomar menos de 10 segundos.
5. **Rendimiento en tiempo real:** Sin interrupciones en la grabación, con sincronización perfecta entre audio y video.
6. **Subida a la nube:** Tiempo de subida menor a 15 segundos por cada 100 MB de video.
7. **Optimización de recursos:** Uso del procesador no debe superar el 70% en grabaciones Full HD.
8. **Ancho de banda:** Transmisión eficiente sin saturar la red y reproducción en línea con latencia inferior a 200 ms.
9. **Capacidad concurrente:** Soporte para al menos 50 usuarios concurrentes sin pérdida de calidad.
10. calidad.

Requisitos de Seguridad

1. **Autenticación:** Los usuarios deben iniciar sesión con credenciales seguras (contraseñas o autenticación de dos factores) para acceder al sistema de grabación y edición.
2. **Autorización:** Diferentes niveles de acceso para usuarios según su rol (administradores, usuarios estándar, etc.), limitando funcionalidades según permisos.
3. **Cifrado de datos:** Los videos y archivos de audio deben estar cifrados tanto en almacenamiento como durante la transmisión para evitar accesos no autorizados.
4. **Protección de datos personales:** Cumplir con las normativas de privacidad (como GDPR) y evitar la captura de datos personales sin consentimiento.
5. **Almacenamiento seguro:** Los archivos se deben almacenar en servidores seguros con copias de seguridad y protección contra accesos no autorizados.
6. **Control de sesiones:** Las sesiones de usuario deben expirar después de un tiempo de inactividad y no se debe permitir múltiples inicios de sesión desde diferentes ubicaciones.
7. **Registro de actividad:** Registrar eventos críticos como inicio de sesión, intentos de acceso no autorizado, y modificaciones en archivos para auditorías de seguridad.
8. **Protección contra ataques:** Implementar protección contra ataques de fuerza bruta, inyecciones de código y accesos indebidos a las APIs del sistema.

9. **Respaldo y recuperación:** Garantizar que los datos de grabaciones y ajustes del usuario puedan ser restaurados en caso de pérdida o corrupción.

Requisitos de Usabilidad

1. **Interfaz intuitiva:** Se asegurará que el diseño sea limpio y sencillo, para que cualquier usuario, sin importar su nivel de experiencia, pueda navegar fácilmente por las opciones disponibles.
2. **Minimización de pasos:** Se asegurará que el flujo de trabajo esté diseñado de manera eficiente, permitiendo a los usuarios grabar y gestionar sus archivos con el menor número de pasos posible.
3. **Accesibilidad:** Se considerará la diversidad de los usuarios, incluyendo características como atajos de teclado y opciones de accesibilidad para facilitar el uso a personas con discapacidades.
4. **Coherencia visual y funcional:** Se mantendrá un diseño consistente en toda la plataforma, tanto en términos visuales como funcionales, para que los usuarios puedan familiarizarse rápidamente con la estructura y las herramientas.
5. **Interacción fluida:** Se asegurará que la respuesta de las acciones dentro de la plataforma sea rápida y coherente, evitando retrasos o comportamientos inesperados que puedan confundir al usuario.

Requisitos de Escalabilidad

1. **Adaptabilidad al crecimiento de usuarios:** Se asegurará que la plataforma soporte un aumento en el número de usuarios simultáneos sin degradar su rendimiento.
2. **Ampliación de funcionalidades:** Se diseñará la arquitectura de manera modular para permitir la incorporación de nuevas características sin afectar las existentes.
3. **Gestión eficiente de recursos:** Se optimizarán los recursos del sistema para manejar una mayor carga de trabajo a medida que la demanda crezca.
4. **Soporte para múltiples resoluciones y calidades:** Se garantizará que la plataforma maneje eficientemente grabaciones en diferentes resoluciones y calidades, sin comprometer la experiencia del usuario.

Modelado E/R

Caracterización de los datos Diagrama de Entidad-Relación

1. Archivo

- **Propósito:** Almacena información general sobre archivos relacionados con grabaciones y audios.
- **Atributos:**
 - **ID** (int): Identificador único del archivo.
 - **Nombre** (varchar): Nombre del archivo.
 - **Ruta** (varchar): Ubicación del archivo en el sistema de almacenamiento.
 - **Tamaño** (float): Tamaño del archivo, posiblemente en MB.
 - **Descripcion** (varchar): Descripción opcional del contenido del archivo.
 - **ID_Grabaciones** (int): Llave foránea que vincula este archivo con una

- grabación específica.
- **ID_Audio** (int): Llave foránea que vincula este archivo con un audio específico.

2. Grabaciones

- **Propósito:** Contiene información detallada sobre las grabaciones realizadas.
- **Atributos:**
 - **ID** (int): Identificador único de la grabación.
 - **Formato** (varchar): Tipo de formato del archivo (ej. MP4, AVI).
 - **Duracion** (float): Duración de la grabación en segundos.
 - **Area** (float): Área cubierta por la grabación, posiblemente en términos espaciales (px^2).
 - **Timestamp_inicio** (float): Marca temporal de inicio.
 - **Timestamp_pausa** (float): Marca temporal de pausa.
 - **Timestamp_finalizacion** (float): Marca temporal de finalización.
 - **Tipo** (varchar): Tipo de grabación (ej. video, sesión de pantalla).
 - **Estado** (varchar): Estado actual de la grabación (ej. activo, archivado).
 - **Palabras_claves** (varchar): Palabras clave asociadas para facilitar la búsqueda.
 - **Fecha** (datetime): Fecha de creación o modificación de la grabación.
 - **ID_Pantalla** (int): Llave foránea que vincula esta grabación con los datos de la pantalla usada.

3. Audios

- **Propósito:** Detalla información sobre archivos de audio asociados.
- **Atributos:**
 - **ID** (int): Identificador único del audio.
 - **Formato** (varchar): Tipo de formato del archivo de audio (ej. MP3, WAV).
 - **Duracion** (float): Duración del audio en segundos.
 - **Timestamp_inicio** (float): Marca temporal de inicio del audio.
 - **Timestamp_pausa** (float): Marca temporal de pausa del audio.
 - **Timestamp_finalizacion** (float): Marca temporal de finalización del audio.
 - **Palabras_claves** (varchar): Palabras clave asociadas al contenido del audio.
 - **ID_Grabaciones** (int): Llave foránea que vincula el audio con una grabación específica.

4. Pantalla

- **Propósito:** Define las características de la pantalla utilizada en una grabación.
- **Atributos:**
 - **ID** (int): Identificador único de las características de la pantalla.
 - **Ancho** (float): Ancho de la pantalla en píxeles.
 - **Alto** (float): Alto de la pantalla en píxeles.
 - **Ini_X** (float): Coordenada inicial en el eje X de la grabación.
 - **Ini_Y** (float): Coordenada inicial en el eje Y de la grabación.
 - **Fin_X** (float): Coordenada final en el eje X de la grabación.
 - **Fin_Y** (float): Coordenada final en el eje Y de la grabación.

Relaciones entre las entidades:

1. Archivo:

- Relación con **Grabaciones** mediante **ID_Grabaciones**.
- Relación con **Audios** mediante **ID_Audio**.

2. Grabaciones:

- Relación con **Pantalla** mediante **ID_Pantalla**.

3. Audios:

- Relación con **Grabaciones** mediante **ID_Grabaciones**.

Diagrama de Entidad-Relación

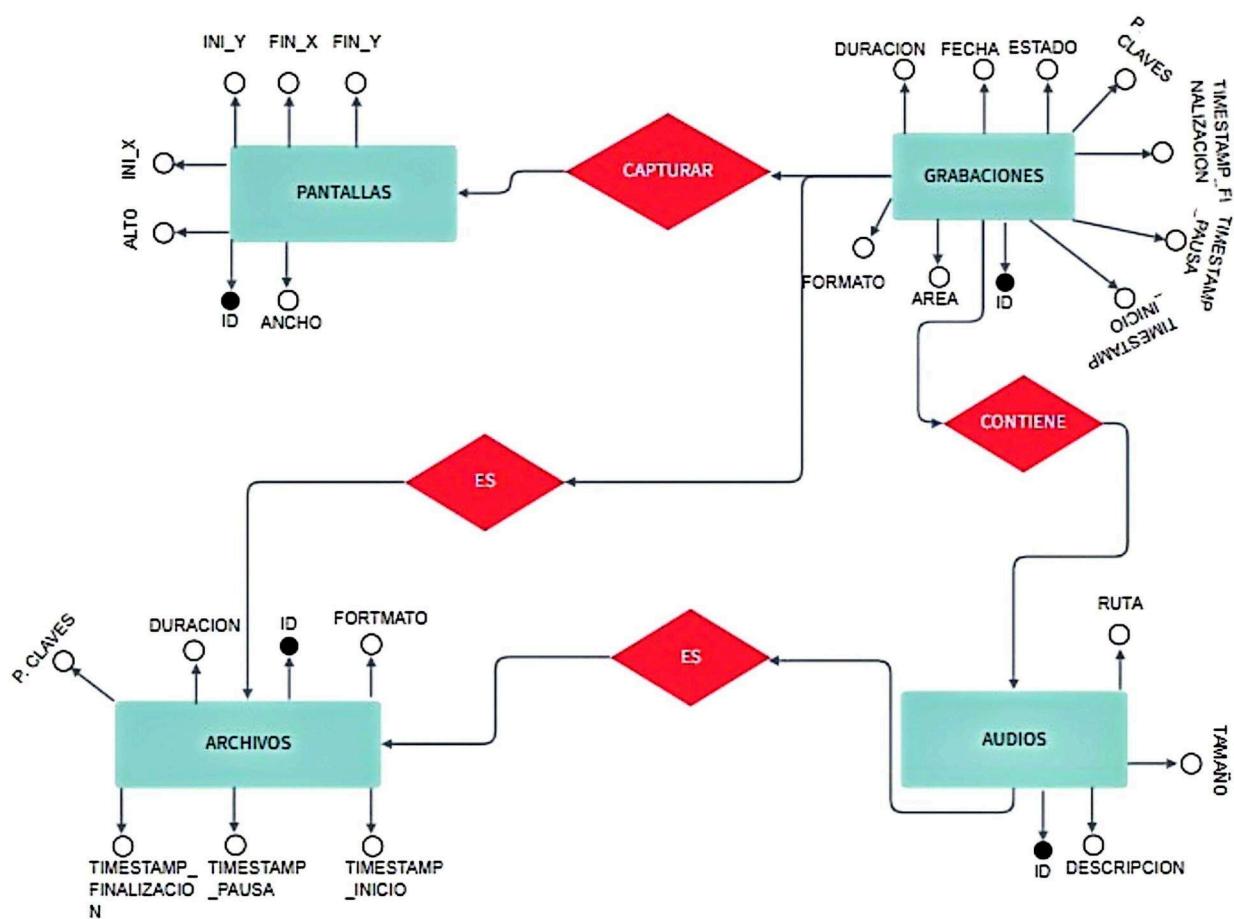
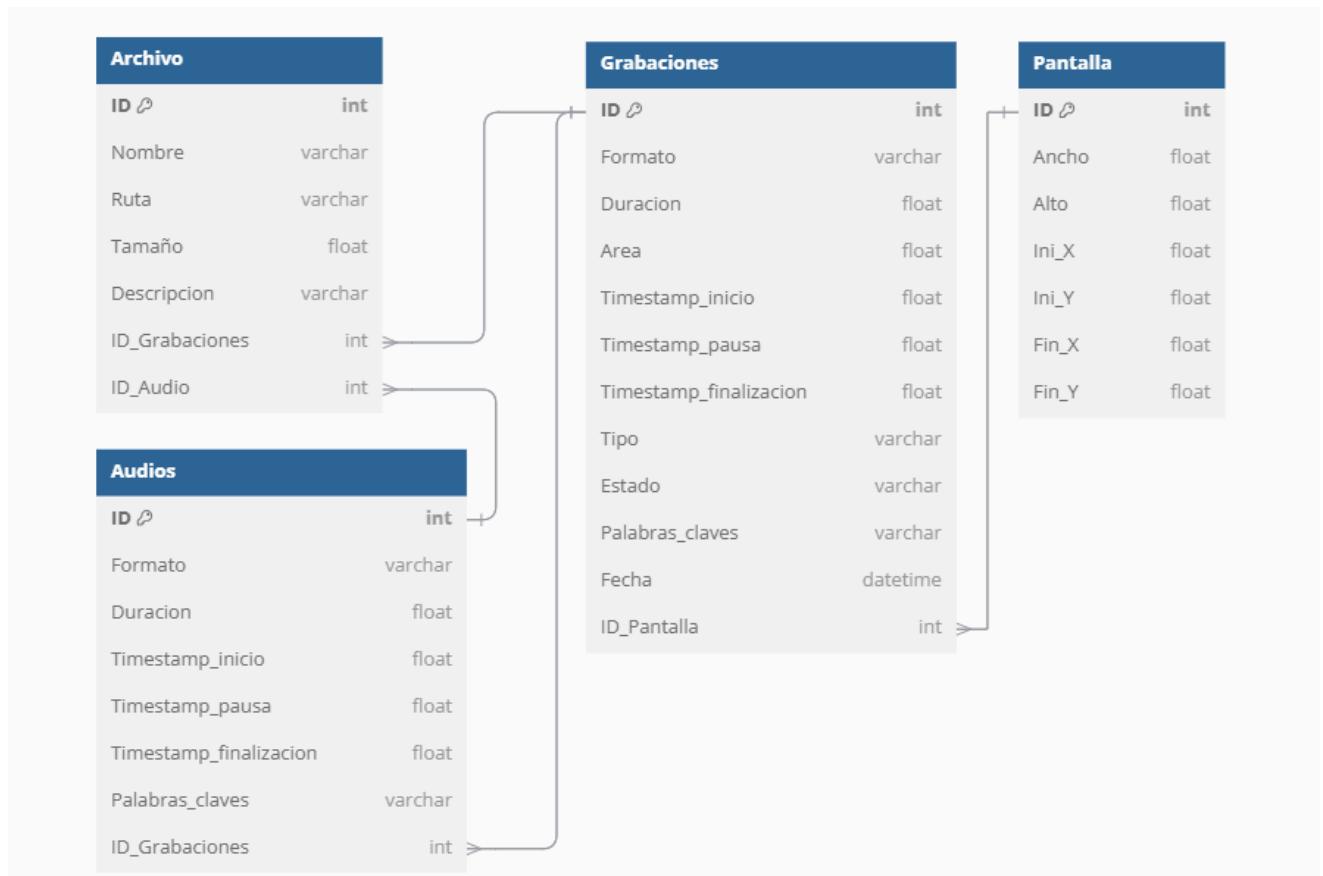


Diagrama relacional



Script del modelo relacional :

```

script.dbml

1 Table Archivo {
2 ID integer [primary key]
3 Nombre varchar
4 Ruta varchar
5 Tamaño integer
6 Descripcion varchar
7 ID_Grabaciones integer [ref: > Grabaciones.ID]
8 ID_Audio integer [ref: > Audios.ID]
9 }
10
11 Table Pantalla {
12 ID integer [primary key]
13 Ancho integer
14 Alto integer
15 Ini_X integer
16 Ini_Y integer
17 Fin_X integer
18 Fin_Y integer
19 }

```

```

script.dbuml

1 Table Audios {
2   ID integer [primary key]
3   Formato varchar
4   Duracion integer
5   Timestamp_inicio integer
6   Timestamp_pausa integer
7   Timestamp_finalizacion integer
8   Palabras_claves varchar
9   ID_Grabaciones integer [ref: > Grabaciones.ID]
10  }
11
12 Table Grabaciones {
13   ID integer [primary key]
14   Formato varchar
15   Duracion integer
16   Area integer
17   Timestamp_inicio integer
18   Timestamp_pausa integer
19   Timestamp_finalizacion integer
20   Tipo varchar
21   Estado varchar
22   Palabras_claves varchar
23   Fecha datetime
24   ID_Pantalla integer [ref: > Pantalla.ID]
25  }
26

```

Descripción de Entidades y Relaciones :

1) Archivo

- a) **Descripción:** Representa los archivos multimedia generados durante las grabaciones, ya sean videos o audios. Incluye información detallada sobre su ubicación, tamaño y relación con otras entidades.
- b) **Atributos principales:**
 - i) **ID:** Identificador único del archivo.
 - ii) **Nombre:** Nombre descriptivo del archivo.
 - iii) **Ruta:** Ubicación donde está almacenado.
 - iv) **Tamaño:** Peso del archivo en bytes.
 - v) **Descripción:** Detalles adicionales sobre el archivo.
 - vi) **ID_Grabaciones:** Relación con la grabación a la que pertenece.
 - vii) **ID_Audio:** Relación con el audio asociado, si corresponde.

2) Pantalla

- a) **Descripción:** Define las configuraciones de captura de pantalla utilizadas durante una grabación. Estas configuraciones determinan el área y la resolución de la pantalla grabada.
- b) **Atributos principales:**
 - i) **ID:** Identificador único de la configuración de pantalla.
 - ii) **Ancho y Alto:** Dimensiones de la pantalla en píxeles.
 - iii) **Ini_X e Ini_Y:** Coordenadas iniciales del área de captura.
 - iv) **Fin_X y Fin_Y:** Coordenadas finales del área de captura.

3) Audios

- a) **Descripción:** Representa las pistas de audio generadas o asociadas a las grabaciones. Proporciona información sobre su duración, palabras clave y relación con las grabaciones.
- b) **Atributos principales:**
 - i) **ID:** Identificador único del audio.
 - ii) **Formato:** Tipo de archivo de audio (MP3, WAV, etc.).
 - iii) **Duración:** Tiempo total del audio en segundos.
 - iv) **Timestamps:** Marcas temporales de inicio, pausa y finalización.
 - v) **Palabras clave:** Etiquetas descriptivas para el contenido del audio.
 - vi) **ID_Grabaciones:** Relación con la grabación a la que pertenece.

4) Grabaciones

- a) **Descripción:** Entidad central que agrupa la información de las sesiones de grabación, incluyendo las configuraciones de pantalla, archivos asociados y estado de la grabación.
- b) **Atributos principales:**
 - i) **ID:** Identificador único de la grabación.
 - ii) **Formato:** Tipo de grabación (por ejemplo, video MP4).
 - iii) **Duración:** Tiempo total de la grabación.
 - iv) **Área:** Tamaño del área capturada en píxeles.
 - v) **Timestamps:** Marcas temporales de inicio, pausa y finalización.
 - vi) **Tipo:** Tipo de grabación (pantalla completa, área seleccionada, etc.).
 - vii) **Estado:** Estado actual de la grabación (iniciada, pausada, finalizada).
 - viii) **Palabras clave:** Etiquetas para facilitar su búsqueda.

- ix) **Fecha:** Fecha y hora en la que se realizó la grabación.
- x) **ID_Pantalla:** Relación con la configuración de pantalla utilizada.

Relaciones

1. Archivo - Grabaciones

- a. **Relación:** Un archivo pertenece a una única grabación, pero una grabación puede tener múltiples archivos asociados (video y audio).
- b. **Cardinalidad:**
 - i. Un archivo (1) → Una grabación (1).
 - ii. Una grabación (1) → Múltiples archivos (N).
- c. **Clave foránea:** **ID_Grabaciones** en **Archivo**.

2. Archivo - Audios

- a. **Relación:** Un archivo puede tener un audio asociado. Este vínculo no es obligatorio, ya que algunas grabaciones pueden no incluir audio.
- b. **Cardinalidad:**
 - i. Un archivo (1) → Un audio (0 o 1).
- c. **Clave foránea:** **ID_Audio** en **Archivo**.

3. Grabaciones - Pantalla

- a. **Relación:** Una grabación utiliza una configuración de pantalla específica. Cada configuración puede ser reutilizada en varias grabaciones.
- b. **Cardinalidad:**
 - i. Una grabación (1) → Una pantalla (1).
 - ii. Una pantalla (1) → Múltiples grabaciones (N).
- c. **Clave foránea:** **ID_Pantalla** en **Grabaciones**.

4. Grabaciones - Audios

- a. **Relación:** Una grabación puede generar un audio asociado. Un audio solo puede pertenecer a una grabación.
- b. **Cardinalidad:**
 - i. Una grabación (1) → Un audio (0 o 1).
- c. **Clave foránea:** **ID_Grabaciones** en **Audios**.

Diagrama Relacional Simplificado

1. **Archivo - Grabaciones:** Relación 1:N para asociar múltiples archivos a una grabación.
2. **Archivo - Audios:** Relación 1:1 para vincular un archivo de video con su pista de audio (opcional).
3. **Grabaciones - Pantalla:** Relación 1:N para vincular una configuración de pantalla a varias grabaciones.
4. **Grabaciones - Audios:** Relación 1:1 para gestionar la pista de audio de una grabación específica.

Colecciones No SQL

```
{-->} grabaciones.json  
1 [  
2   {  
3     "id": 101,  
4     "formato": "mp4",  
5     "duracion": 3600,  
6     "area": 500,  
7     "timestampInicio": 1620303030,  
8     "timestampPausa": 1620303200,  
9     "timestampFinalizacion": 1620306630,  
10    "tipo": "clase",  
11    "estado": "completa",  
12    "palabrasClaves": ["clase", "curso", "video"],  
13    "fecha": {  
14      "dia": 3,  
15      "mes": 5,  
16      "anio": 2021,  
17      "hora": 10,  
18      "minuto": 15,  
19      "segundo": 0  
20    },  
21    "pantallaId": 301  
22  }  
23 ]
```

```
{-->} pantallas.json
```

```
1 [  
2   {  
3     "id": 301,  
4     "ancho": 1920,  
5     "alto": 1080,  
6     "iniX": 0,  
7     "iniY": 0,  
8     "finX": 1920,  
9     "finY": 1080  
10    }  
11 ]
```

```
{-->} audios.json
```

```
1 [  
2   {  
3     "id": 201,  
4     "formato": "mp3",  
5     "duracion": 180,  
6     "timestampInicio": 1620303030,  
7     "timestampPausa": 1620303200,  
8     "timestampFinalizacion": 1620303330,  
9     "palabrasClaves": ["audio", "grabacion", "clase"],  
10    "grabacionId": 101  
11    }  
12 ]
```

```
{-->} archivos.json
```

```
1 [  
2   {  
3     "id": 1,  
4     "nombre": "nombreArchivo1",  
5     "ruta": "/ruta/archivo1",  
6     "tamano": 12345,  
7     "descripcion": "Descripción del archivo 1",  
8     "grabacionId": 101,  
9     "audioId": 201  
10    }  
11 ]
```

Etapa 2: Persistencia de Datos con Backend

Introducción:

En esta etapa del proyecto "**Video Recording**", se establecerá la infraestructura de persistencia de datos en el backend. Este componente es esencial para almacenar y gestionar de manera estructurada las grabaciones y la información asociada, permitiendo que los datos puedan guardarse, recuperarse y organizarse de forma eficaz. Esto asegura una integración robusta entre el componente de grabación de pantalla y el backend.

Durante el semestre anterior, se trabajó en la documentación inicial del proyecto, y se espera que el desarrollo del frontend sea abordado en el próximo semestre, permitiendo una experiencia completa de gestión de grabaciones para los usuarios finales.

Propósito de la Etapa:

El objetivo de esta etapa es garantizar la persistencia y administración confiable de los datos generados por el sistema. Esto incluye el almacenamiento de las grabaciones y otros datos necesarios para su administración, facilitando la integración entre el frontend y el backend y permitiendo que los usuarios accedan y gestionen sus archivos de manera eficiente.

Alcance de la Etapa:

En la Etapa 2 se diseñará la base de datos para almacenar y organizar las grabaciones de manera estructurada y eficiente, implementando una API para permitir la futura comunicación entre el frontend y el backend, lo cual se desarrollará en una etapa posterior. Además, se gestionarán los metadatos necesarios para facilitar la administración de los archivos, junto con la persistencia de las opciones de edición, como cortes y eliminaciones de segmentos.

Definiciones y Acrónimos

Definiciones

1. **MongoDB:** Sistema de gestión de bases de datos NoSQL, orientado a documentos, que utiliza un formato de almacenamiento basado en BSON (Binary JSON). MongoDB es conocido por su escalabilidad, flexibilidad y facilidad para manejar grandes volúmenes de datos no estructurados.
2. **Mongo Atlas:** Servicio administrado de bases de datos MongoDB en la nube, proporcionado por MongoDB Inc. Permite a los desarrolladores implementar, gestionar y escalar bases de datos MongoDB sin tener que preocuparse por la infraestructura o el mantenimiento.
3. **Node.js:** Entorno de ejecución para JavaScript que permite ejecutar código JavaScript en el servidor. Basado en el motor V8 de Google Chrome, es conocido por su alta eficiencia y su modelo de programación asíncrona y orientado a eventos.
4. **Backend:** Parte del desarrollo de una aplicación que se encarga de la lógica de negocio, almacenamiento de datos, y la comunicación con otras aplicaciones o servicios. Se ejecuta en un servidor y no es accesible directamente por los usuarios finales, pero es esencial para el funcionamiento de la aplicación.
5. **Capa de datos:** Capa en una arquitectura de software que se encarga de gestionar el acceso a la base de datos y de garantizar que los datos se almacenen, recuperen y

manipulen de forma eficiente. A menudo, se maneja mediante consultas y comandos SQL (en bases de datos relacionales) o comandos específicos de bases de datos NoSQL.

6. **Service:** Componente de software que contiene la lógica de negocio y ofrece funcionalidades específicas a otras partes del sistema, como la interacción con la base de datos, el procesamiento de datos, o la ejecución de tareas relacionadas. Los servicios en una arquitectura de software suelen estar desacoplados de la interfaz de usuario.
7. **Controller:** Componente dentro de un patrón de diseño MVC (Modelo-Vista-Controlador) que maneja las solicitudes de los usuarios, coordina la ejecución de las operaciones del servicio y gestiona la respuesta que se envía al usuario. Los controladores gestionan la interacción entre la capa de presentación y la capa de datos.
8. **Repository:** Patrón de diseño que proporciona una capa de abstracción entre la capa de acceso a los datos y el resto de la aplicación. Los repositorios gestionan la lógica de consulta y persistencia de datos, y permiten realizar operaciones CRUD (Crear, Leer, Actualizar, Eliminar) de forma más estructurada.

Acrónimos

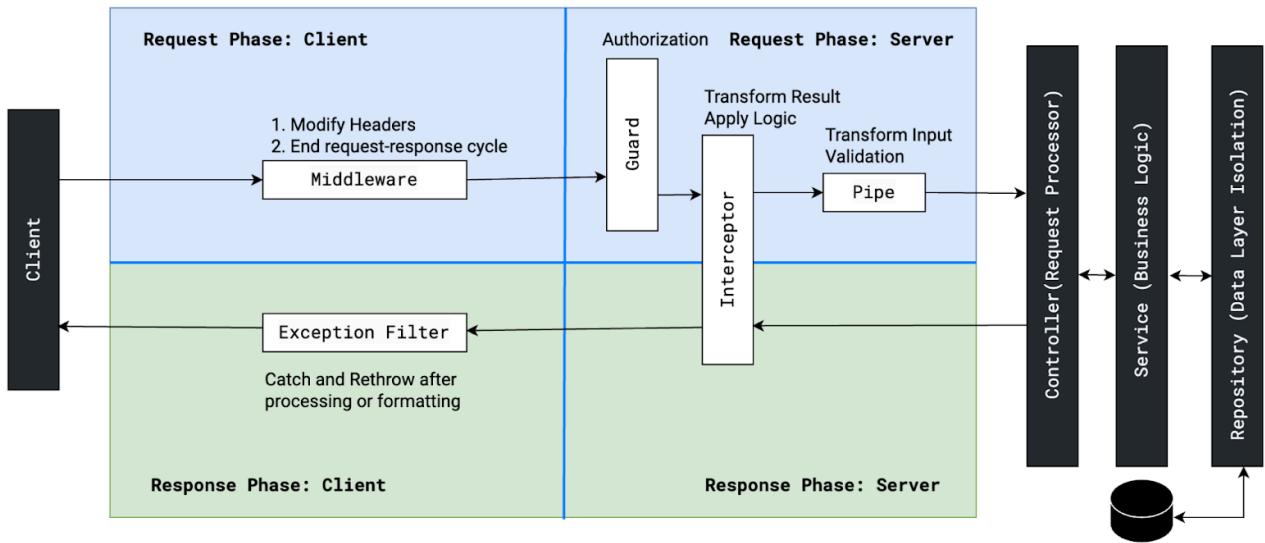
1. **MongoDB:** NoSQL DB, orientado a documentos, basado en BSON (Binary JSON).
2. **API** (Application Programming Interface): Conjunto de funciones y protocolos que permiten que las aplicaciones se comuniquen entre sí.
3. **CRUD (Create, Read, Update, Delete):** Operaciones básicas realizadas en bases de datos para gestionar los datos.
4. **MVC** (Model-View-Controller): Patrón de diseño de software que separa la lógica de negocio, la interfaz de usuario y el control de la aplicación en tres componentes distintos.
5. **SQL (Structured Query Language):** Lenguaje estándar para gestionar y manipular bases de datos relacionales.
6. **NoSQL (Not Only SQL):** Tipo de bases de datos que no utilizan un modelo relacional. MongoDB es un ejemplo de base de datos NoSQL.
7. **JSON (JavaScript Object Notation):** Formato de intercambio de datos basado en texto que es fácil de leer y escribir para los humanos, y fácil de analizar y generar por las máquinas. Se usa comúnmente para el intercambio de datos entre clientes y servidores.

Diseño de la Arquitectura de Backend

Descripción de la Arquitectura Propuesta:

La arquitectura backend del componente "Video Recording" utiliza **Nest.js** en **Node.js**, organizada en módulos para manejar la grabación, almacenamiento y recuperación de vídeos. Incluye un controlador para gestionar las solicitudes, un servicio para la lógica de grabación, y una base de datos para almacenar metadatos de las grabaciones. También cuenta con un sistema de autenticación para asegurar el acceso y, opcionalmente, una cola de procesamiento para tareas intensivas. Todo esto está expuesto mediante una API REST que permite la integración con otros servicios o un frontend, garantizando un sistema seguro y escalable.

Componentes del Backend



Este proyecto está diseñado para gestionar una plataforma de grabaciones, audios, archivos y pantallas utilizando **NestJS** como marco principal. Su propósito es proporcionar una API robusta y escalable que permita a los usuarios interactuar eficientemente con estos recursos, garantizando modularidad y mantenibilidad.

Componentes principales del backend

1. Cliente

El cliente representa cualquier sistema que consume la API desarrollada. Puede ser una interfaz web, una aplicación móvil o un script automatizado. Los clientes interactúan con el backend para subir archivos, acceder a grabaciones, consultar audios o gestionar pantallas. Realizan solicitudes HTTP a la API mediante endpoints específicos definidos en los controladores.

Por ejemplo:

1. Un cliente podría realizar un **POST /archivos** para subir un archivo al sistema.
2. Otro cliente podría solicitar una lista de grabaciones con **GET /grabaciones**.

2. Controlador

Los controladores en este proyecto actúan como intermediarios entre el cliente y los servicios del backend. Cada módulo (archivos, grabaciones, audios y pantallas) tiene su propio controlador que gestiona las rutas y define cómo se manejan las solicitudes entrantes.

En el caso del módulo **archivos**, el controlador podría:

1. Recibir una solicitud para subir un archivo (**POST /archivos**).
2. Validar que el archivo tenga el formato y los datos requeridos.
3. Pasar la solicitud al servicio correspondiente para que este realice el procesamiento necesario.

El controlador asegura que cada solicitud se gestione de manera estructurada y que el cliente reciba una respuesta adecuada, ya sea confirmando una operación exitosa o indicando un error.

3. Servicio

El servicio es el núcleo del backend, donde se define la lógica de negocio para cada módulo. En este proyecto, cada módulo tiene su propio servicio encargado de procesar las solicitudes del controlador y aplicar las reglas necesarias.

Por ejemplo:

- El servicio del módulo **grabaciones** podría procesar los metadatos de una grabación (como duración y calidad) antes de almacenarla en la base de datos.
- En el módulo **audios**, el servicio podría verificar que el archivo de audio cumple con los requisitos antes de relacionarlo con una grabación específica.

El servicio interactúa directamente con el repositorio para realizar las operaciones sobre la base de datos y asegura que las reglas del negocio se cumplan antes de devolver una respuesta al controlador.

4. Repositorio

El repositorio es el componente encargado de interactuar directamente con la base de datos. Este proyecto utiliza **Mongoose** para gestionar las colecciones de MongoDB, como **archivos**, **grabaciones**, **audios** y **pantallas**.

Cada módulo tiene su propio repositorio, que se encarga de:

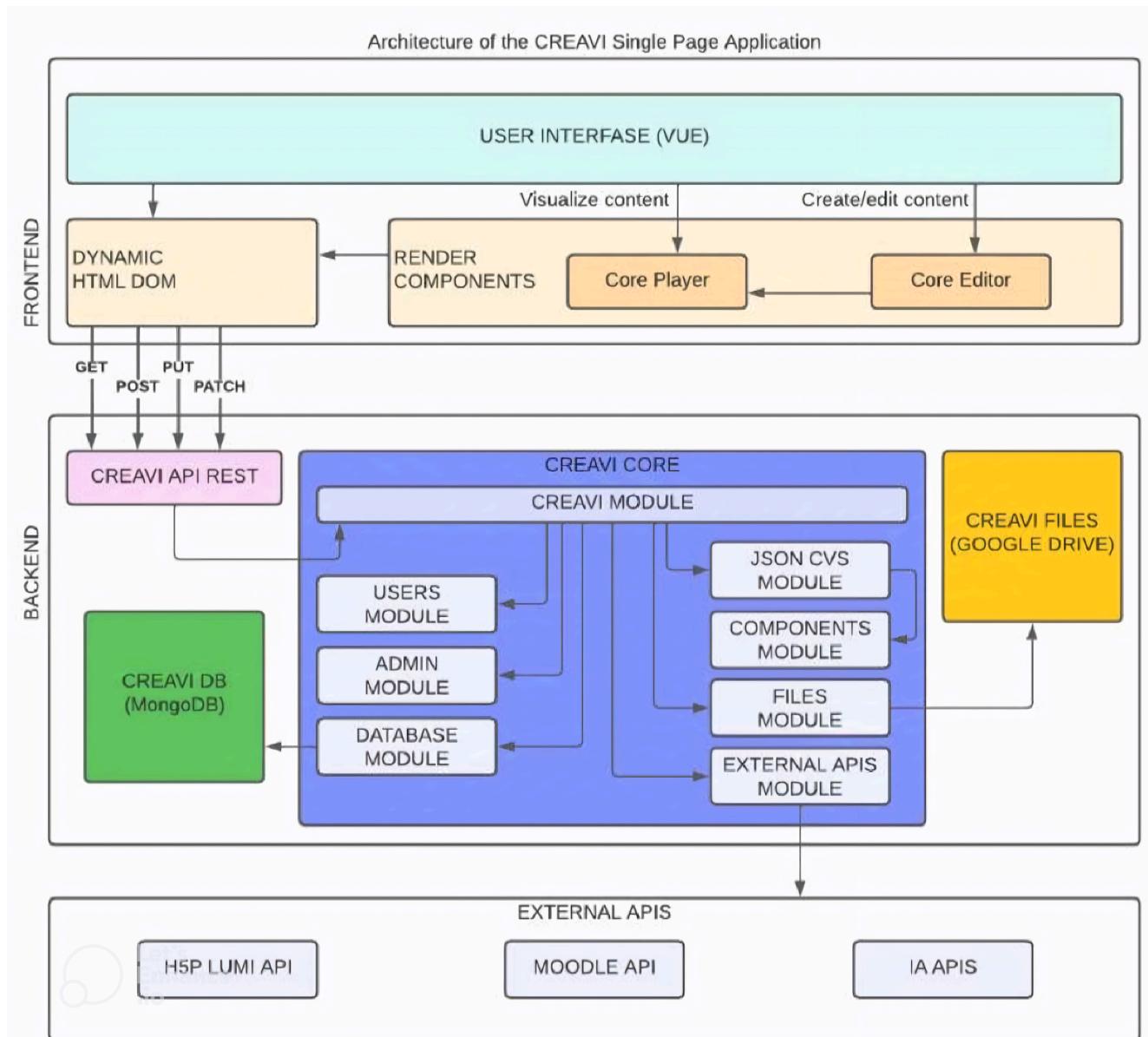
- Almacenar datos estructurados según los esquemas definidos.
- Realizar consultas eficientes, como búsquedas por filtros o paginación.
- Actualizar o eliminar datos cuando sea necesario.

Por ejemplo, el repositorio del módulo **pantallas** podría guardar información sobre pantallas grabadas o devolver un listado de estas según un rango de fechas especificado.

Resumen del flujo del sistema

1. El cliente realiza una solicitud a través de un endpoint específico.
2. El controlador recibe la solicitud, la valida y la envía al servicio correspondiente.
3. El servicio procesa la solicitud aplicando las reglas de negocio y solicita al repositorio que interactúe con la base de datos.
4. El repositorio ejecuta la operación en la base de datos (como almacenar, recuperar o eliminar información).
5. Finalmente, la respuesta fluye de regreso al cliente, indicando el resultado de la operación.

Diagramas de Arquitectura

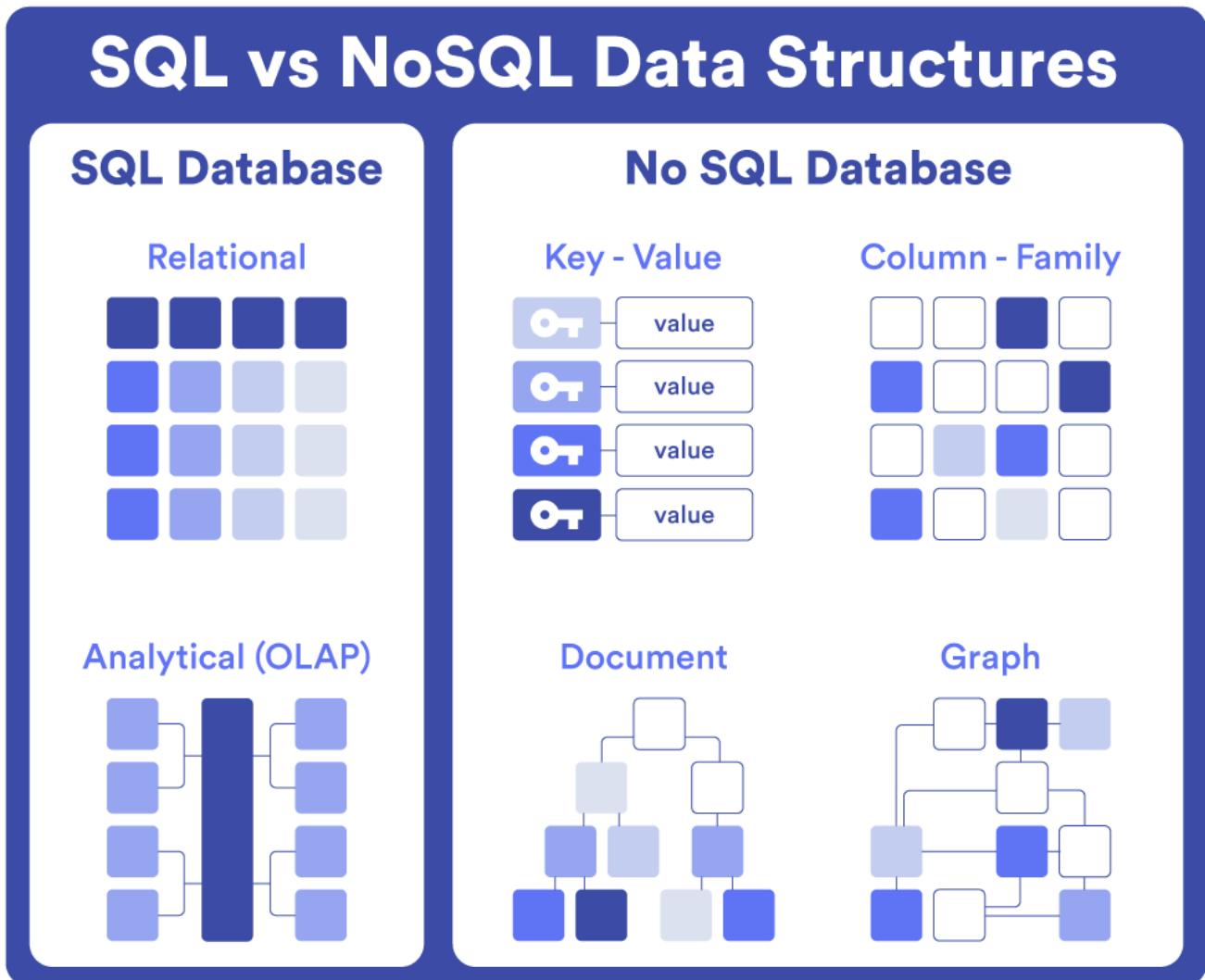


El diagrama anterior muestra la arquitectura de la aplicación de una sola página (SPA) de CREAVI, compuesta por una interfaz de usuario que incluye componentes para la visualización y edición de contenido, un backend que maneja la lógica de la aplicación mediante módulos, y una base de datos MongoDB. En esta arquitectura, se integrará el componente "Video Recording", un grabador de pantalla que será muy útil para satisfacer las necesidades de los usuarios dentro de la plataforma grabando la pantalla completamente o por secciones, con audio o sin audio según la necesidad.

En el frontend, se incorporarán componentes para la captura de video y retroalimentación en los módulos de visualización y edición de las grabaciones. En el backend, se añadirá un nuevo módulo dentro del núcleo de la plataforma, con endpoints en la API REST para manejar solicitudes de comandos relacionados con la grabación de video, lógica para interpretar y ejecutar estos comandos (como iniciar, pausar o detener la grabación), y generación de respuestas visuales o de estado. Esto mejorará significativamente la accesibilidad y usabilidad de la plataforma, permitiendo a los usuarios crear, editar y gestionar grabaciones de pantalla de manera eficiente.

Elección de la Base de Datos

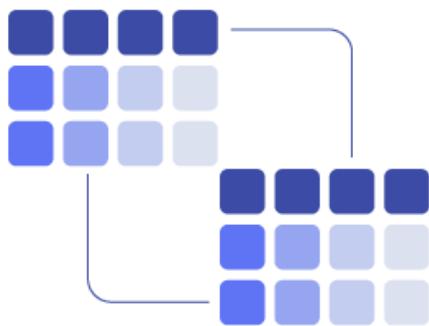
Evaluación de Opciones (SQL o NoSQL)



SQL

NoSQL

Structured



VS

Unstructured

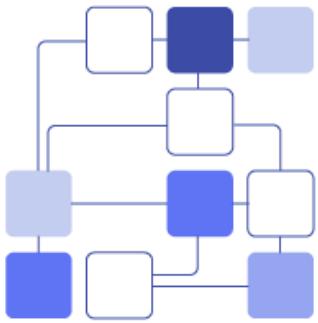


Table with fixed rows
and columns
(Static schema)

Relational model

Vertically scalable

Not suited for
hierarchical data storage

SQL Server, MySQL,
PostgresSQL Oracle

Dynamic schema

Non relational model

Horizontally scalable

Suited for
hierarchical data storage

MongoDB, RavenDB,
LiteDB

Justificación de la Elección

En el proyecto “**video recording**”, una base de datos NoSQL es la opción más adecuada debido a la flexibilidad y escalabilidad que ofrece. Al manejar grandes volúmenes de datos no estructurados, como los metadatos de las grabaciones y los registros de actividad, NoSQL permite un almacenamiento más eficiente y rápido. Además, su capacidad para escalar horizontalmente facilita el manejo de crecientes cantidades de datos a medida que el sistema crece, sin necesidad de reestructurar la base de datos. Esto optimiza tanto el rendimiento como la disponibilidad de la plataforma

Diseño de esquema de bases de datos

Las colecciones del componente **Video Recording** en la plataforma **Creavi** gestionan diferentes aspectos de las grabaciones de pantalla. La colección **Grabación** almacena información sobre cada grabación, incluyendo detalles como formato, duración, área grabada, timestamps de inicio, pausa y finalización, tipo, estado, palabras clave y la fecha de grabación. La colección **Pantalla** define las propiedades de las pantallas o áreas grabadas, como dimensiones y coordenadas. **Audio** contiene los datos de los archivos de audio asociados, como formato, duración y timestamps, junto con las palabras clave. Por último, la colección **Archivo** gestiona los archivos generados durante la grabación, incluyendo información sobre el nombre, ruta, tamaño, descripción y referencias a las grabaciones y audios correspondientes. Juntas, estas colecciones permiten un control completo sobre las grabaciones, su almacenamiento y gestión dentro de la plataforma Creavi.

Implementación del Backend

Elección del Lenguaje de Programación

El componente **Video Recording** se implementa utilizando **Node.js** y **NestJS**, un framework modular y escalable para crear aplicaciones backend. **Node.js** proporciona un entorno de ejecución eficiente para manejar múltiples solicitudes concurrentes, mientras que **NestJS** facilita la estructura del proyecto y la gestión de dependencias, permitiendo un desarrollo más organizado y mantenable. En este caso, se utilizan las colecciones de **MongoDB** para almacenar información sobre las grabaciones de pantalla, los archivos multimedia asociados (vídeo y audio), y las pantallas grabadas. **Mongoose** se utiliza para interactuar con la base de datos, y los esquemas están definidos en NestJS para reflejar las colecciones como **Grabación**, **Audio**, **Archivo** y **Pantalla**. La arquitectura se basa en servicios que gestionan la lógica de negocio, y controladores que manejan las solicitudes HTTP para interactuar con los datos. Esto permite un manejo eficiente de las grabaciones de video y audio, así como un almacenamiento organizado de los archivos generados durante el proceso de grabación

Creación de la Lógica de Negocio

Estructura del Servicio en NestJS

Un servicio en NestJS es responsable de la lógica de negocio, gestionando las operaciones con las bases de datos y exponiendo métodos que pueden ser llamados por los controladores. Para el componente de **Video Recording**, el servicio manejará la creación de grabaciones, la vinculación de audios y archivos, la gestión de pantallas grabadas, y más. Este servicio es el

lugar donde se implementa la lógica de negocio principal, como la creación de grabaciones, la adición de audios y archivos relacionados, y la gestión de las pantallas grabadas.

Componentes de la Estructura del Servicio

A continuación, se detalla la estructura básica del servicio en el contexto de este proyecto:

a. Decorador `@Injectable()`

El servicio se define como una clase decorada con `@Injectable()`, lo que le permite ser inyectado en otros componentes de la aplicación, como los controladores y otros servicios. Este decorador habilita a NestJS para gestionar automáticamente la instancia del servicio y sus dependencias a través de la inyección de dependencias. Por ejemplo:

```
archivos.service.ts
1  @Injectable() // Decorador que convierte esta clase en un servicio inyectable.
2  export class ArchivosService {
3      // Constructor para injectar el modelo de Mongoose asociado a 'Archivo'.
4      constructor(
5          @InjectModel(Archivo.name) private archivoModel: Model<Archivo>, // 'archivoModel' se usa para interactuar con la colección 'archivos'.
6      ) {}
7
8      // Método para crear un nuevo archivo en la base de datos.
9      create(createArchivoDto: CreateArchivoDto) {
10         const createdArchivo = new this.archivoModel(createArchivoDto); // Crear una nueva instancia del modelo con los datos proporcionados.
11         return createdArchivo.save(); // Guardar el nuevo documento en la base de datos.
12     }
13
14     // Método para obtener todos los documentos de la colección 'archivos'.
15     async findAll() {
16     }
17
18     // Método para buscar un archivo específico por su ID.
19     async findOne(id: string) {
20     }
21
22     // Método para actualizar un archivo específico por su ID.
23     async update(id: string, updateArchivoDto: UpdateArchivoDto) {
24     }
25
26     // Método para eliminar un archivo específico por su ID.
27     async remove(id: string) {
28 }
29
```

Métodos del Servicio

Los métodos del servicio contienen la **lógica de negocio**. En este caso, el servicio tendría métodos para realizar operaciones CRUD (Crear, Leer, Actualizar, Eliminar) sobre las grabaciones de video. Además, puede incluir lógica adicional para asociar otros recursos, como audios o pantallas, a las grabaciones de video.

Algunos de los métodos que podrían estar presentes en el servicio son:

- **create()**: Crea una nueva grabación de video en la base de datos.
- **findAll()**: Obtiene todas las grabaciones de vídeo almacenadas.
- **findOne()**: Recupera una grabación de video específica por su ID.
- **update()**: Actualiza los datos de una grabación de video existente.
- **remove()**: Elimina una grabación de video de la base de datos..

Desarrollo de Endpoints

Colección	Método	Descripción	Endpoint
Archivos	POST	Crear un nuevo archivo y asociarlo a una grabación.	/archivos
	GET	Obtener todos los archivos almacenados.	/archivos
	GET	Obtener un archivo específico por su ID.	/archivos/:id
	PUT	Actualizar un archivo existente.	/archivos/:id
	DELETE	Eliminar un archivo por su ID.	/archivos/:id
Grabaciones	POST	Crear una nueva grabación de video y asociarla con pantallas y audios.	/grabaciones
	GET	Obtener todas las grabaciones de vídeo almacenadas.	/grabaciones
	GET	Obtener una grabación específica por su ID.	/grabaciones/:id
	PUT	Actualizar los detalles de una grabación existente.	/grabaciones/:id
	DELETE	Eliminar una grabación específica por su ID.	/grabaciones/:id
Audios	POST	Subir un nuevo archivo de audio y asociarlo a una grabación.	/audios
	GET	Obtener todos los audios almacenados.	/audios
	GET	Obtener un audio específico por su ID.	/audios/:id
	PUT	Actualizar los detalles de un audio existente.	/audios/:id
	DELETE	Eliminar un audio por su ID.	/audios/:id
Pantallas	POST	Crear una nueva configuración de pantalla asociada a una grabación.	/pantallas
	GET	Obtener todas las configuraciones de pantallas almacenadas.	/pantallas

	GET	Obtener una configuración de pantalla específica por su ID.	<code>/pantallas/:id</code>
	PUT	Actualizar una configuración de pantalla existente.	<code>/pantallas/:id</code>
	DELETE	Eliminar una configuración de pantalla por su ID.	<code>/pantallas/:id</code>

Conexión a la Base de Datos

1. Instalación de Dependencias

Antes de realizar la conexión, se deben instalar las dependencias necesarias en el proyecto de **NestJS**. Esto incluye tanto **Mongoose** como la **integración de NestJS con Mongoose**:

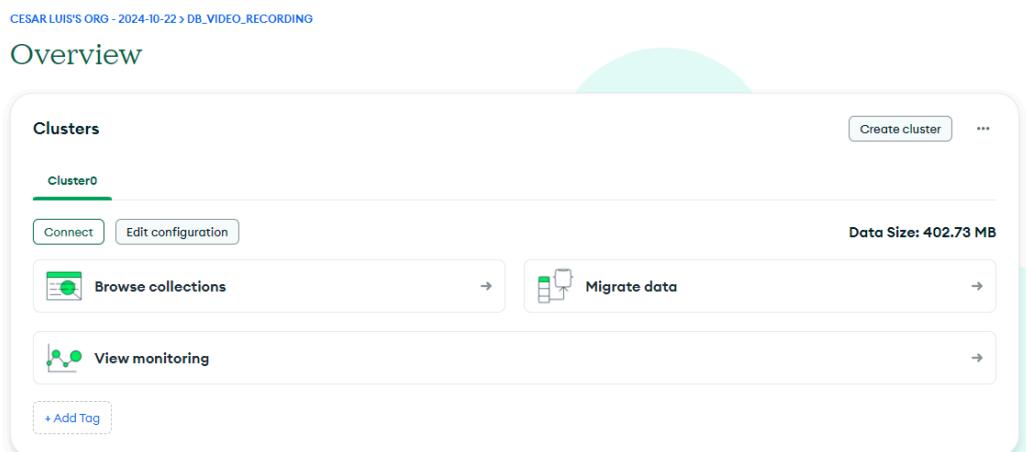
```
npm install @nestjs/mongoose mongoose
```

- **@nestjs/mongoose:** Proporciona la integración entre **NestJS** y **Mongoose**.
- **mongoose:** Es el ORM (Object-Relational Mapper) que facilita la comunicación con **MongoDB**.

2. Configuración de MongoDB Atlas

MongoDB Atlas es un servicio de base de datos en la nube para **MongoDB**, que permite gestionar bases de datos de manera remota. Aquí están los pasos para configurarlo:

1. **Crear una cuenta en MongoDB Atlas:**
 - Visita [MongoDB Atlas](#) y crea una cuenta.
 - Una vez iniciada la sesión, crea un nuevo **Proyecto**.
2. **Crear un Cluster en MongoDB Atlas:**
 - En el panel de Atlas, crea un nuevo **cluster** seleccionando la configuración gratuita ("M0").
 - Una vez creado el cluster, se genera una URL de conexión, que se utilizará para conectar la aplicación con la base de datos.



3. **Configurar IP y Usuarios de la Base de Datos:**

- En el menú de seguridad de MongoDB Atlas, debes **añadir tu IP** a la lista de IPs permitidas para acceder al cluster (puedes permitir el acceso desde cualquier IP durante el desarrollo, usando **0.0.0.0/0**).
- Además, crea un **usuario de base de datos** con un nombre de usuario y contraseña. Este usuario tendrá acceso a la base de datos que deseas conectar desde la aplicación.

4. Obtener la URL de Conexión:

- MongoDB Atlas proporciona una URL para la conexión. Este es el formato típico:

```
app.module.ts
1 @Module({
2   imports: [
3     MongooseModule.forRoot(
4       'mongodb+srv://<usuario>:<contraseña>@cluster0.avxwc.mongodb.net/db_video_recording?retryWrites=true&w=majority&appName=Cluster0',
5     ),
6     ArchivosModule,
7     AudiosModule,
8     PantallasModule,
9     GrabacionesModule,
10   ],
11   controllers: [AppController],
12   providers: [AppService],
13 })
```

3. Configuración de la Conexión en NestJS

Una vez que tienes las dependencias instaladas y la URL de conexión, debes configurarlo dentro del archivo de configuración de NestJS. Para ello, se utilizará el módulo **MongooseModule** de NestJS para establecer la conexión a la base de datos de **MongoDB Atlas**.

```
app.module.ts
1 import { Module } from '@nestjs/common';
2 import { AppController } from './app.controller';
3 import { AppService } from './app.service';
4 import { ArchivosModule } from './archivos/archivos.module';
5 import { AudiosModule } from './audios/audios.module';
6 import { PantallasModule } from './pantallas/pantallas.module';
7 import { GrabacionesModule } from './grabaciones/grabaciones.module'
8 import { MongooseModule } from '@nestjs/mongoose';
```

Configuración de los Esquemas (Schemas)

Con la conexión ya establecida, el siguiente paso es definir los esquemas de las colecciones en MongoDB utilizando Mongoose. Esto se hace para mapear los documentos de las colecciones a las clases de JavaScript de NestJS. Ejemplo:

```

archivos.schemas.ts

1 import { Prop, Schema, SchemaFactory } from '@nestjs/mongoose';
2 import { Document } from 'mongoose';
3
4 @Schema()
5 export class Archivo extends Document {
6   @Prop({ required: true })
7   id: number;
8
9   @Prop({ required: true })
10  nombre: string;
11
12  @Prop({ required: true })
13  ruta: string;
14
15  @Prop()
16  tamano: number;
17
18  @Prop()
19  descripcion: string;
20
21  @Prop({ required: true })
22  grabacionId: number;
23
24  @Prop({ required: true })
25  audioId: number;
26 }
27
28 export const ArchivoSchema = SchemaFactory.createForClass(Archivo);

```

Registrar el esquema en el módulo correspondiente:

Una vez creado el esquema, es necesario registrararlo en el módulo para que NestJS pueda gestionarlo adecuadamente. Ejemplo:

```

archivos.module.ts

1 import { Module } from '@nestjs/common';
2 import { ArchivosService } from './archivos.service';
3 import { ArchivosController } from './archivos.controller';
4 import { MongooseModule } from '@nestjs/mongoose';
5 import { Archivo, ArchivoSchema } from './schemas/archivos.schemas';
6
7 @Module({
8   imports: [
9     MongooseModule.forFeature([{ name: Archivo.name, schema: ArchivoSchema }])
10    ],
11    controllers: [ArchivosController],
12    providers: [ArchivosService],
13  })
14
15 export class ArchivosModule {}
16

```

Pruebas del Backend

En las pruebas del backend utilizando Postman, se validaron las rutas de la API, asegurando que los endpoints funcionen correctamente y devuelvan las respuestas esperadas. Este proceso se realizó para verificar que las operaciones CRUD (Crear, Leer, Actualizar y Eliminar) de las colecciones (como Archivos, Grabaciones, Audios, y Pantallas) se ejecuten correctamente en el backend, conectado a MongoDB Atlas. A continuación, se explica detalladamente qué se hizo en las pruebas del backend con Postman:

1. Configuración Inicial de Postman

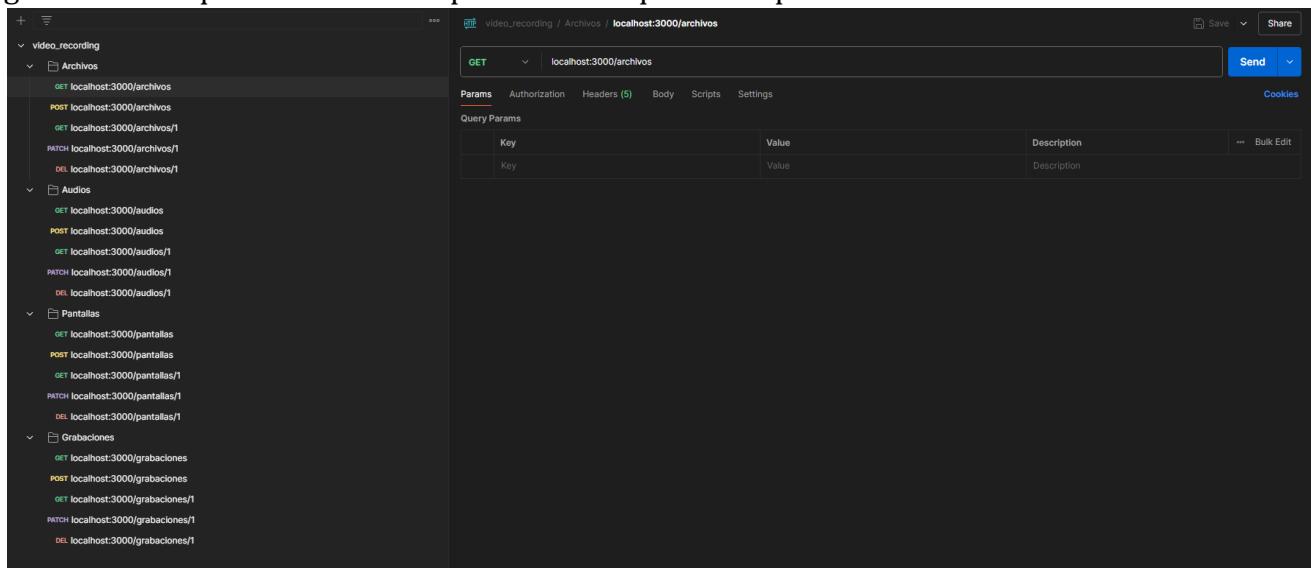
Antes de realizar las pruebas, se configuró **Postman** para hacer solicitudes HTTP a los endpoints del backend:

- **URL base de la API:** Se definió la URL base del backend en Postman (por ejemplo, `http://localhost:3000/` para desarrollo local o la URL correspondiente en producción).
- **Colección de pruebas:** Se organizó una **colección de pruebas** en Postman para agrupar todas las solicitudes relacionadas con las distintas colecciones (por ejemplo, **Archivos, Grabaciones, Audios y Pantallas**).
- **Autenticación (si es necesaria):** Si el backend está protegido por autenticación (JWT, API Key, etc.), se configuraron los headers de autenticación en Postman para cada solicitud.

2. Pruebas de las Operaciones CRUD (Create, Read, Update, Delete)

Las pruebas se enfocaron en verificar que cada operación de las colecciones en el backend funcione correctamente. Cada tipo de solicitud (POST, GET, PUT, DELETE) se probó según el caso de uso para cada colección.

Las pruebas realizadas en **Postman** ayudaron a asegurar que todas las funcionalidades principales del backend estén operativas. Además, permitieron verificar que las rutas, la gestión de recursos y las interacciones con la base de datos se realizan correctamente, garantizando que el sistema cumpla con los requisitos esperados.



Etapa 3: Consumo de Datos y Desarrollo Frontend Introducción

Propósito de la Etapa

Alcance de la Etapa

Definiciones y Acrónimos

Creación de la Interfaz de Usuario (UI)

Diseño de la Interfaz de Usuario (UI) con HTML y CSS

Consideraciones de Usabilidad

Maquetación Responsiva

Programación Frontend con JavaScript (JS)

Desarrollo de la Lógica del Frontend

Manejo de Eventos y Comportamientos Dinámicos

Uso de Bibliotecas y Frameworks (si aplicable)

Consumo de Datos desde el Backend

Configuración de Conexiones al Backend

Obtención y Presentación de Datos

Actualización en Tiempo Real (si aplicable)

Interacción Usuario-Interfaz

Manejo de Formularios y Validación de Datos

Implementación de Funcionalidades Interactivas

Mejoras en la Experiencia del Usuario

Pruebas y Depuración del Frontend

Diseño de Casos de Prueba de Frontend

Pruebas de Usabilidad

Depuración de Errores y Optimización del Código

Implementación de la Lógica de Negocio en el Frontend

Migración de la Lógica de Negocio desde el Backend (si necesario)

Validación de Datos y Reglas de Negocio en el Frontend

Integración con el Backend

Verificación de la Comunicación Efectiva con el Backend

Pruebas de Integración Frontend-Backend

