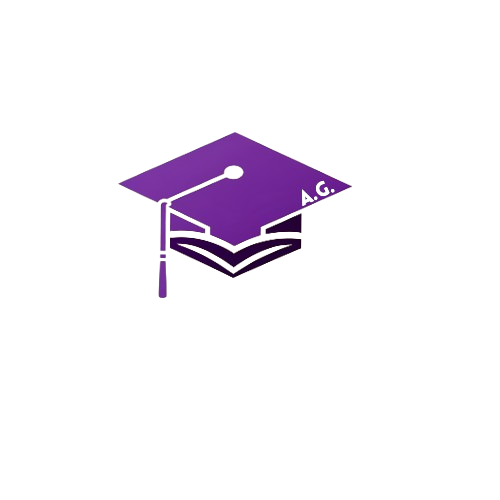
PROYECTO DAM

ACADEMY GESTOR



Fecha de presentación:

2024/25 IZAN FRANCO ANDUAGA - 72153424D

INDICE

* INTRODUCCION
* RESUMEN
* OBJETIVOS
* ANALISIS DE CONTEXTO
* ESTADO DEL ARTE
* INNOVACION
* ANALISIS DE REQUISITOS
* DISEÑO
* IMPLEMENTACION
* PRUEBAS

INTRODUCCION

Para este proyecto realizaremos una aplicación para gestionar el alumnado y las actividades de una escuela local.

La idea nació de la situación de una persona cercana, que en su trabajo como profesora en la escuela de circo de Torrelavega advirtió, junto al resto del equipo, la necesidad de mejorar las capacidades de gestión de la escuela, para agilizar las tareas y poder realizar estas de forma más eficiente, hasta ahora no se había encontrado ninguna alternativa ya desarrollada que se ajuste a sus necesidades o, con demasiadas funciones, de las cuales utilizarían tan solo una parte mínima.

El planteamiento de este proyecto se centra en desarrollar una aplicación más sencilla pero centrada en las funciones esenciales que se van a utilizar, teniendo en cuenta a profesores y alumnos, desde las solicitudes de inscripción hasta el registro de pagos de cada uno.

De esta manera se pretende reducir las numerosas horas trabajo administrativo (hasta ahora todos estos datos se han manejado a mano) que, si se gestionan a través de las herramientas adecuadas se pueden ver notablemente reducidas, dedicando así más horas a otras labores igual de importantes pero quizá menos apremiantes.

En consecuencia, este proyecto tratará de proveer las herramientas para hacer de este trabajo más liviano.

RESUMEN

Academy Gestor

Se plantea como una aplicación para facilitar las tareas administrativas de en este caso una escuela de circo, se proporcionarán una aplicación de escritorio desde la cual se pueda gestionar tanto las inscripciones de alumnos como las resultantes matriculas, los cursos ofertados, el personal de la escuela y tener un control de los recibos generados para cada alumno; y una app móvil con un doble propósito, por un lado y mediante inicio de sesión permitirá a los profesores realizar los controles de asistencia y generación de informes sobre el objetivo de las clases. Por otro lado, será un espacio informativo para las familias donde se publicarán anuncios con respecto a las actividades de la escuela, contenido visual referente a los distintos cursos, información básica, contacto etc…

PALABRAS CLAVE

* Gestor
* Escuela
* Matriculas
* Extraescolares
* Academia
* Cursos
* Talleres
* Gestión académica

OBJETIVOS

* Objetivo principal: Agilizar las labores de gestión de una escuela de extraescolares y facilitar la comunicación con las familias.
* Objetivos específicos:

Obj 1- Automatizar registros de datos de alumnos

Obj 2- Gestión de alumnos (dar de alta/ baja alumnos, modificar sus datos…)

Obj 3- Gestión de matrículas (consulta, modificación, registro…)

Obj 4- Gestión de cursos (registro de cursos disponibles y capacidad para crear, modificar o eliminar los existentes)

Obj 5- Gestión de personal (registro de datos de profesores, pudiendo crear, eliminar y modificar)

Obj 6- Control de recibos (se registrarán automáticamente al inicio del mes, y de cada uno se podrá generar un recibo y/o enviarlo por correo al alumno o tutor)

Obj 7- Publicación de noticias referentes a la escuela

Obj 8- Publicación de contenido visual de los distintos cursos o eventos

Obj 9- Canal de información y contacto para las familias

Obj 10- Control de asistencia e incidencias para los profesores

ANALISIS DE CONTEXTO

El presente proyecto busca modernizar la gestión y comunicación de una escuela de circo mediante la implementación de una base de datos centralizada y el desarrollo de aplicaciones de escritorio y móvil. Para comprender mejor el impacto de esta iniciativa se realiza el siguiente análisis de contexto.

Ahora analizaremos el entorno que rodea a la educación extraescolar en España, considerando factores económicos, tecnológicos, socioculturales y legislativos, con base en datos de los últimos dos años (2023-2024). El objetivo es comprender el estado actual y las proyecciones futuras, fundamentales para el desarrollo de cualquier iniciativa vinculada al sector.

Análisis económico

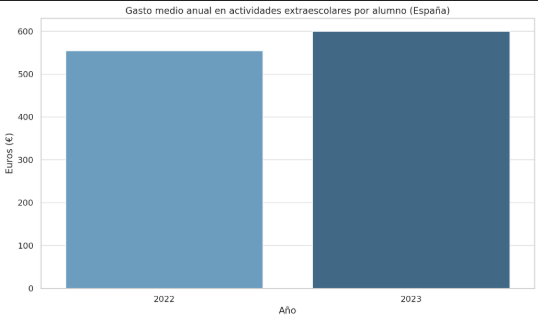
La educación extraescolar, actividades como clases de idiomas, música, refuerzo escolar, etc… se ha visto influenciada por el contexto económico general y por el comportamiento del gasto familiar.

En el ámbito general, en 2023, España mostró signos de estabilización tras la pandemia y el impacto de la guerra en Ucrania. El crecimiento económico fue de en torno al 2,5%.

Aunque la inflación, especialmente en productos y servicios, ha condicionado los presupuestos familiares.

Impacto en la educación extraescolar:

Según datos del INE, el gasto en actividades extraescolares creció un 8% en 2023 respecto al periodo anterior, con una media de más de 600 € anuales por alumno en familias de renta media-alta.



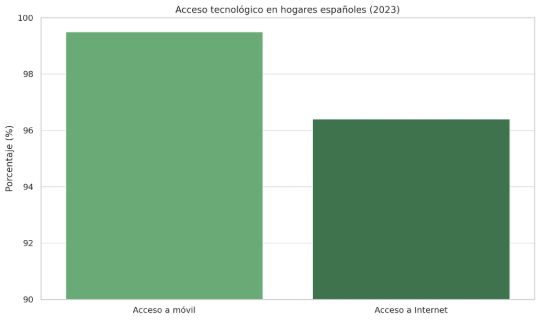
A pesar del aumento general, el acceso sigue siendo desigual. En familias con menores ingresos, las actividades extraescolares siguen siendo un lujo limitado.

Las ayudas públicas en este ámbito son escasas y varían mucho según la comunidad autónoma o el municipio.

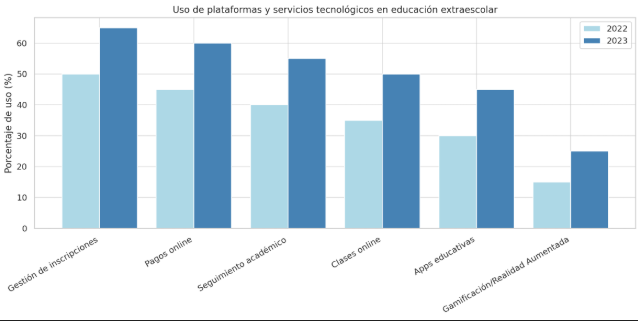
Análisis tecnológico

La digitalización de las actividades extraescolares ha cobrado especial relevancia tras la pandemia y se ha mantenido en ascenso en los últimos años.

En 2023, el 99,5% de los hogares españoles tenía al menos un teléfono móvil, y el 96,4% contaba con acceso a Internet, incluso en zonas rurales.

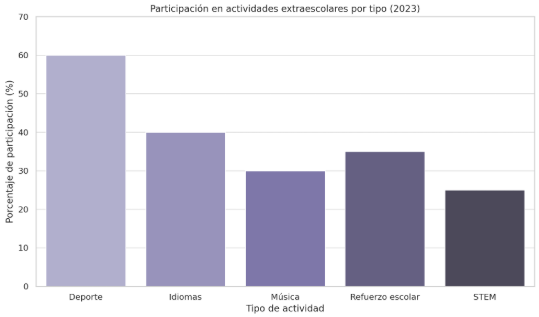


Lo que permitió que muchas academias y centros ofrezcan gestión digital de inscripciones, pagos y seguimiento de actividades extraescolares. Lo que se define como un aumento del uso de plataformas online complementarias a las clases y un mayor uso de aplicaciones para las gestiones académicas de las entidades.



Análisis sociocultural

En 2023, según la Fundación Cotec y datos de Save The Children, alrededor del 70% de los niños y adolescentes entre 6 y 16 años participaron en al menos una actividad extraescolar. También se ha acusado una mayor implicación de las familias en la selección de actividades.



Las actividades más solicitadas siguen siendo deportivas como fútbol, judo o natación, idiomas, especialmente inglés, música y refuerzos escolares. Pero también se ha observado un crecimiento significativo del interés por actividades más enfocadas en el sector tecnológico o científico (STEM Ciencia, Tecnología, Ingeniería y Matemáticas), esto se debe al reconocimiento de las tecnologías como cada vez más importantes en cuanto al futuro profesional de los jóvenes.

También se ha visto incrementada la oferta de programas con enfoque inclusivo para reducir la brecha educativa entre clases sociales.

Análisis legislativo

La regulación de las actividades extraescolares está dispersa y depende en gran medida de competencias autonómicas y municipales.

No existe una ley específica de actividades extraescolares, pero estas están reguladas en parte por la Ley Orgánica 3/2020 (LOMLOE), que reconoce su valor complementario. Las actividades que implican a menores están sujetas a estrictas normativas de protección de datos (LOPDGDD) y prevención de riesgos.

Muchas comunidades autónomas han aprobado planes de impulso a la educación extraescolar, aunque de forma difusa. Algunos municipios ofrecen bonificaciones o subvenciones para actividades extraescolares, de forma que las cuotas sean mucho más asequibles o incluso cubriendo totalmente los gastos de las actividades.

Hay que mencionar especialmente la ley de Protección Integral de la Infancia y la Adolescencia frente a la Violencia (2021), que impone obligaciones de control y prevención para actividades con menores. También la LSSI-CE, aplicable a plataformas o apps que gestionen servicios de este tipo. Así como también, y de carácter obligatorio en algunos casos, se requieren para el personal certificados negativos de delitos sexuales, formación en primeros auxilios, y en algunos casos, titulación específica.

ESTADO DEL ARTE

El presente análisis del estado del arte examina las tecnologías y soluciones actuales utilizadas en la gestión de inscripciones, matrículas y comunicación en escuelas similares, así como las tendencias tecnológicas y los productos existentes en el mercado que abordan problemas similares.

SITUACION ACTUAL

Actualmente, muchas instituciones pequeñas y medianas continúan utilizando herramientas tradicionales como hojas de cálculo (Excel) y comunicaciones manuales (WhatsApp) para la gestión de estudiantes y notificaciones. Sin embargo, la tendencia global se inclina hacia la digitalización y automatización mediante plataformas especializadas que ofrecen bases de datos centralizadas, aplicaciones móviles y soluciones en la nube. Herramientas como Google Classroom o Moodle han ganado popularidad en la gestión educativa, pero están diseñadas principalmente para la educación formal y no se adaptan fácilmente a un entorno de formación más informal y flexible.

En cuanto a la tecnología utilizada, la mayoría de las soluciones actuales emplean bases de datos SQL o documentales para almacenar información, aplicaciones web responsivas para gestión administrativa y aplicaciones móviles para la interacción con los usuarios. La integración de servicios de mensajería automática también es común en plataformas más avanzadas.

COMPETENCIA

El sector de la gestión educativa cuenta con diversas soluciones que pueden considerarse competencia para este proyecto, aunque no todas ofrecen un enfoque específico para la formación artística y circense. Algunas de las soluciones más destacadas incluyen:

* **ClassDojo**: Diseñada para la comunicación entre docentes y familias, permite gestionar asistencia y enviar notificaciones, pero no ofrece una solución completa para la gestión de inscripciones y matrículas.



* **EducaPro**: Una plataforma de pago que facilita la gestión académica y administrativa, pero su costo es elevado para escuelas pequeñas y su interfaz está más orientada a la educación escolar tradicional.

Las soluciones existentes destacan por facilitar la comunicación entre estudiantes, familias y administradores, y proporcionan automatización en ciertos aspectos como recordatorios y almacenamiento en la nube. Asimismo, cuentan con interfaces intuitivas y adaptadas para dispositivos móviles. Sin embargo, presentan carencias, como la ausencia de una gestión flexible de inscripciones y matrículas adaptada a clases no tradicionales o la falta de una integración fluida entre bases de datos de inscripciones, asistencia y comunicación.

|  |  |
| --- | --- |
| FORTALEZAS | OPORTUNIDADES |
| Adaptación a las necesidades especificas  Facilidad de uso  Integración de funciones clave | Auge de la digitalización educativa  Posible expansión a otras escuelas |
| DEBILIDADES | AMENAZAS |
| No hay precedentes específicos  Curva de aprendizaje  Algunas opciones de mercado son costosas | Resistencia al cambio en la institución  Pueden surgir soluciones especializadas |

En conclusión, si bien existen plataformas para la gestión educativa y la comunicación entre docentes y familias, ninguna está diseñada específicamente para un entorno extraescolar. Este proyecto representa una oportunidad para ofrecer una solución adaptada que cubra las carencias identificadas en la competencia y permita a la institución modernizar su gestión de manera eficiente y accesible.

INNOVACION

Este proyecto aporta una solución específica que cubre necesidades no resueltas por productos existentes. A diferencia de las plataformas educativas tradicionales, permite gestionar inscripciones y matrículas de manera flexible, adaptándose a clases no convencionales. Además, la automatización de asistencia y la integración de comunicación directa con las familias a través de una aplicación móvil mejoran la eficiencia del sistema actual.

La principal innovación radica en la centralización de funciones clave en una sola plataforma accesible desde dispositivos móviles, eliminando la necesidad de registros en papel y hojas de cálculo manuales.

El cambio de paradigma en este sector se da al reemplazar los métodos tradicionales de gestión manual por una solución digital específica para centros de formación artística. Además, se introduce un sistema de control de asistencia optimizado para dispositivos móviles, lo que permite a los instructores gestionar listas de alumnos en cualquier momento y lugar.

En cuanto a soluciones innovadoras aplicadas, este proyecto utiliza una combinación de metodologías ágiles para el desarrollo de software, bases de datos escalables y una arquitectura modular basada en micro servicios, lo que permite una mejor adaptabilidad y evolución del sistema a futuro.

ANALISIS DE REQUISITOS

**Requisitos Funcionales**

|  |  |
| --- | --- |
|  | |
| Nº Requisito | RF-01 |
| Nombre | Gestión de alumnos |
| Tipo | Requisito |
| Descripción | Permite registrar, modificar y eliminar datos de los alumnos. |
| Prioridad | Alta |
| Objetivo relacionado | Obj.1 |

|  |  |
| --- | --- |
|  | |
| Nº Requisito | RF-02 |
| Nombre | Inscripción y matrículas |
| Tipo | Requisito |
| Descripción | Permite la inscripción de nuevos alumnos y la gestión de matrículas. |
| Prioridad | Alta |
| Objetivo relacionado | Obj.2 / Obj.4 |
|  | |
| Nº Requisito | RF-03 |
| Nombre | Gestión de cursos |
| Tipo | Requisito |
| Descripción | Posibilidad de crear, modificar y eliminar cursos |
| Prioridad | Alta |
| Objetivo relacionado | Obj.3 |

|  |  |
| --- | --- |
|  | |
| Nº Requisito | RF-04 |
| Nombre | Control de asistencia |
| Tipo | Requisito |
| Descripción | Registrar la asistencia de alumnos desde app móvil |
| Prioridad | Media |
| Objetivo relacionado | Obj.10 |

|  |  |
| --- | --- |
|  | |
| Nº Requisito | RF-05 |
| Nombre | Gestión de pagos y recibos |
| Tipo | Requisito |
| Descripción | Registro de pagos y recibos |
| Prioridad | Media |
| Objetivo relacionado | Obj.6 |

|  |  |
| --- | --- |
|  | |
| Nº Requisito | RF-06 |
| Nombre | Publicaciones |
| Tipo | Requisito |
| Descripción | Publicar anuncios y noticias sobre la escuela. |
| Prioridad | Media |
| Objetivo relacionado | Obj.7 |

|  |  |
| --- | --- |
|  | |
| Nº Requisito | RF-07 |
| Nombre | Comunicación con las familias |
| Tipo | Requisito |
| Descripción | Consulta de información o contacto a través de la app móvil |
| Prioridad | Media |
| Objetivo relacionado | Obj.9 |

|  |  |
| --- | --- |
|  | |
| Nº Requisito | RF-08 |
| Nombre | Gestión de personal |
| Tipo | Requisito |
| Descripción | Registro y gestión del personal |
| Prioridad | Alta |
| Objetivo relacionado | Obj.5 |

Requisitos no funcionales

|  |  |
| --- | --- |
|  | |
| Nº Requisito | RNF-01 |
| Nombre | Seguridad |
| Tipo | Requisito |
| Descripción | Protección de datos de alumnos, profesores y tutores |
| Prioridad | Media |

|  |  |
| --- | --- |
|  | |
| Nº Requisito | RNF-02 |
| Nombre | Ley de protección de datos |
| Tipo | Requisito |
| Descripción | Cumplimiento normativa GDRP |
| Prioridad | Alta |

|  |  |
| --- | --- |
|  | |
| Nº Requisito | RNF-03 |
| Nombre | Tiempo de respuesta |
| Tipo | Restricción |
| Descripción | Las operaciones críticas deben ejecutarse en menos de 2 segundos. |
| Prioridad | Alta |

|  |  |
| --- | --- |
|  | |
| Nº Requisito | RNF-04 |
| Nombre | Usabilidad y intuitivita |
| Tipo | Requisito |
| Descripción | Interfaz intuitiva y fácil de usar sin conocimientos técnicos |
| Prioridad | Alta |

|  |  |
| --- | --- |
|  | |
| Nº Requisito | RNF-05 |
| Nombre | Escalabilidad |
| Tipo | Requisito |
| Descripción | Permitir futuras implementaciones de funcionalidades sin afectar al rendimiento |
| Prioridad | Media |

|  |  |
| --- | --- |
|  | |
| Nº Requisito | RNF-07 |
| Nombre | Backups |
| Tipo | Requisito |
| Descripción | Implementación de un sistema de respaldo automático de la base de datos. |
| Prioridad | Baja |

|  |  |
| --- | --- |
|  | |
| Nº Requisito | RNF-08 |
| Nombre | Disponibilidad |
| Tipo | Requisito |
| Descripción | Disponibilidad elevada para evitar interrupciones |
| Prioridad | Media |

DISEÑO

**DIAGRAMA FISICO**

Servidor (VM con Ubuntu Server)

* Servicios instalados:
  + Apache2 (servidor web para alojar y ejecutar la API)
  + MySQL Server (gestiona la base de datos)
  + API alojada (por ejemplo, en Spring Boot ejecutada como servicio o con cron)
  + Almacenamiento de archivos (local en el servidor)

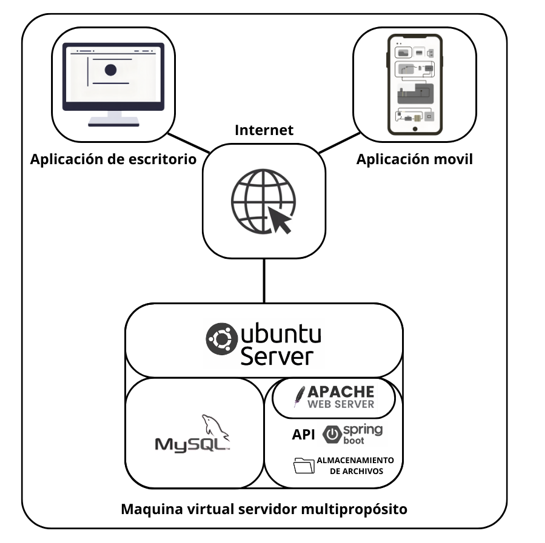
Clientes

* Aplicación de escritorio
  + Accede a la API vía HTTP usando JPA (con REST API)
* Aplicación móvil
  + Accede a la API vía HTTP usando Retrofit

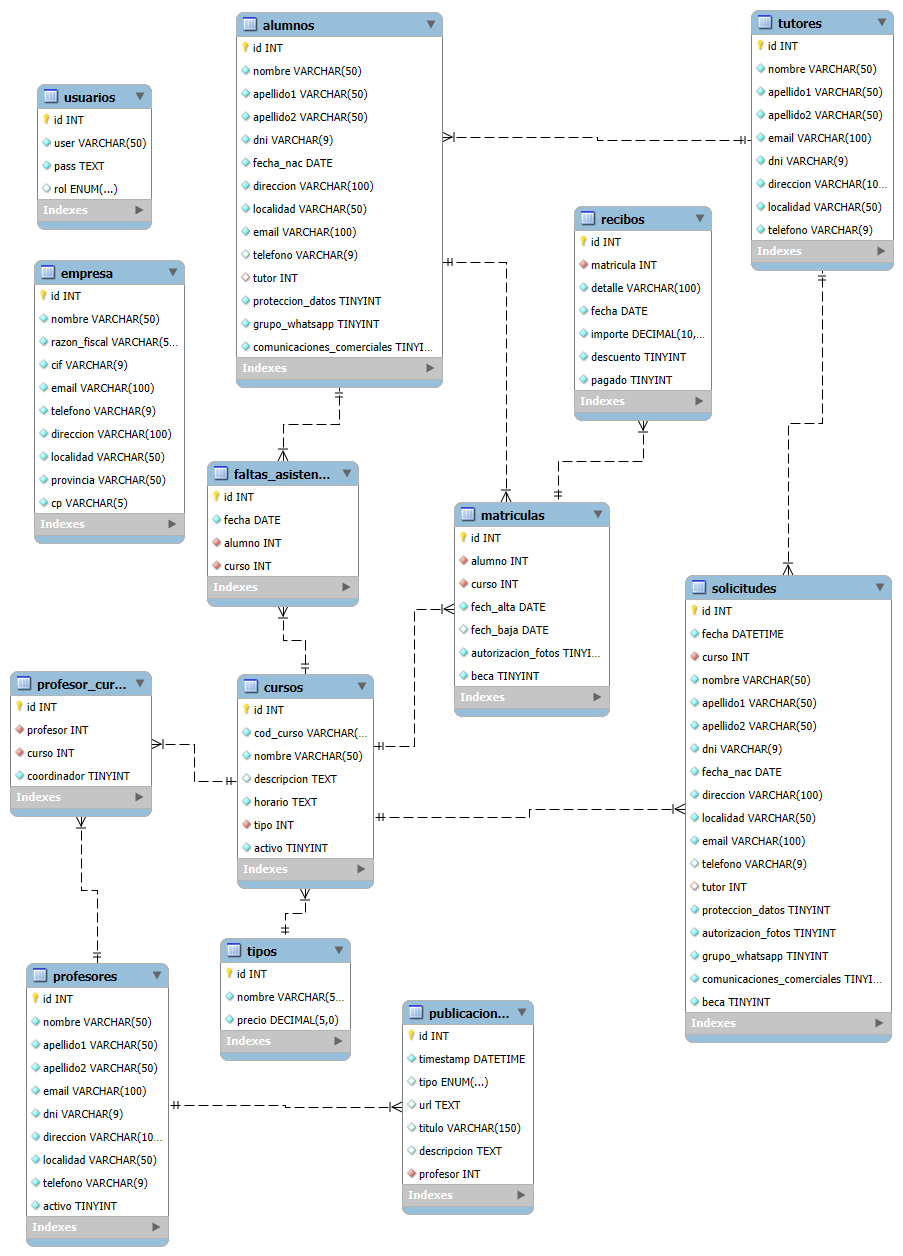
Red

Comunicación entre clientes y servidor vía HTTP,

Acceso restringido al puerto 3306 (MySQL) solo localmente (la API es quien consulta la base de datos)

****

**DIAGRAMA RELACIONAL**

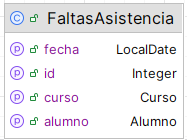
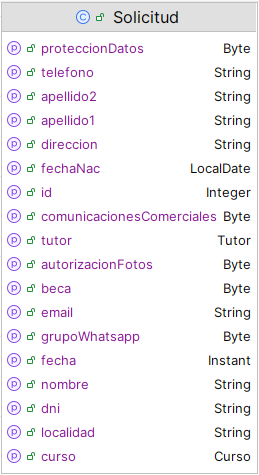
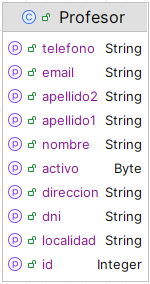
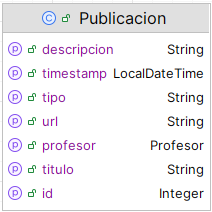
****

**EXPLICACION:**

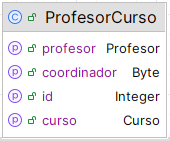
**La base de datos recoge los registros de la escuela de circo, reflejados en las siguientes tablas:**

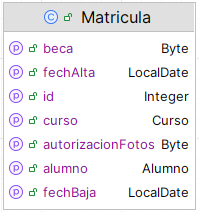
* **Empresa: se recogen los datos de la empresa, id (PK), nombre, razón fiscal, CIF, email, teléfono, dirección, localidad, provincia y código postal. No guarda relación con ninguna otra tabla.**
* **Usuario: datos de los usuarios que pueden acceder a las aplicaciones, id (PK), usuario, contraseña y rol (admin para quienes gestionaran todos los registros de la escuela desde la aplicación de escritorio, user destinado a los profesores a la hora de usar la aplicación móvil para pasar lista, subir publicaciones etc…). No guarda relación con otras tablas.**
* **Solicitudes: recoge los datos de alumnos y tutores legales para solicitar nuevas inscripciones, que una vez aceptadas generaran registros en las tablas relacionadas. id (PK), fecha de la solicitud, curso que se solicita(FK de la tabla curso),(datos del alumno) nombre, apellido1, apellido2, DNI, fecha de nacimiento, dirección, localidad, email, teléfono, tutor (FK de la tabla tutores), protección de datos, autorización de fotos grupo de WhatsApp, comunicaciones comerciales y si tiene o no beca.**
* **Alumnos: todos los datos de los alumnos, id (PK), nombre, apellido1, apellido2, DNI, fecha de nacimiento, dirección, localidad, email, teléfono, tutor (FK de la tabla tutores), protección de datos, grupo de WhatsApp y comunicaciones comerciales, estas tres últimas son booleanos que indican si se aceptan o no dichas condiciones.**
* **Tutores: datos de los tutores legales de los alumnos menores de edad, id (PK), nombre, apellido1, apellido2, email, DNI, dirección, localidad, teléfono.**
* **Faltas de asistencia: registros de faltas de alumnos, id (PK), fecha, alumno (FK tabla alumnos), curso (FK tabla cursos).**
* **Matriculas: registros para cada curso que realiza un alumno, id (PK), alumno (FK tabla alumnos), curso (FK tablas cursos), fechas de alta y de baja la cual puede ser nula, si se tiene autorización para fotos y beca.**
* **Recibos: registros mensuales para cada matricula para control de pagos, id (PK), matricula (FK tabla matriculas), detalle, fecha, importe, si tiene descuento por beca, si esta pagado.**
* **Cursos: datos de cada curso ofrecido, id (PK), código, nombre, descripción, horario, tipo (FK tabla tipos), y si esta o no activo.**
* **Tipos: tipos de cursos impartidos, id (PK), nombre y precio.**
* **Profesores: datos de los profesores, id (PK), nombre, apellido1, apellido2, email, DNI, dirección, localidad, teléfono y si esta activo.**
* **Profesores\_Curso: tabla intermedia entre cursos y profesores que recoge los profesores participantes en cada curso y si son o no coordinadores, id (PK), profesor (FK tabla profesores), curso (FK tabla cursos), y si es coordinador (booleano).**
* **Publicaciones: datos de las publicaciones subidas por cada profesor.**

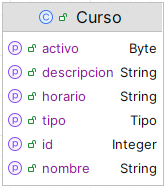
**DIAGRAMA DE CLASES**

****

****

****

****

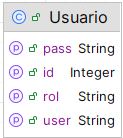
****

****

****

****

****

****

DESCRIPCIÓN DE CLASES DEL SISTEMA

* Solicitud
  + Atributos:  
    id, nombre, apellido1, apellido2, email, telefono, dni, fechaNac, direccion, localidad, grupoWhatsapp, proteccionDatos, autorizacionFotos, comunicacionesComerciales, fecha, curso, tutor
  + Métodos: getters y setters para cada atributo.
  + Relaciones:

curso - referencia a Curso

tutor - referencia a Tutor

* Curso
  + Atributos:  
    id, nombre, descripcion, horario, activo, tipo
  + Métodos: getters y setters.
  + Relaciones:

tipo - referencia a Tipo

Relación con Alumno, ProfesorCurso, FaltasAsistencia, Matricula

* Tipo
  + Atributos:  
    id, nombre, precio
  + Métodos: getters y setters.
  + Relaciones:

Referenciado por Curso

* Tutor
  + Atributos:  
    id, nombre, apellido1, apellido2, email, telefono, dni, direccion, localidad
  + Métodos: getters y setters.
  + Relaciones:

Relacionado con Alumno (como padre/tutor)

Relacionado con Solicitud (si el alumno es menor de edad)

* Profesor
  + Atributos:  
    id, nombre, apellido1, apellido2, email, telefono, dni, direccion, localidad
  + Métodos: getters y setters.
  + Relaciones:

Relacionado con ProfesorCurso

* Publicacion
  + Atributos:  
    id, titulo, descripcion, profesor, uri, timestamp, email
  + Métodos: getters y setters.
  + Relaciones:

No se relaciona directamente con otras entidades

Profesor – referencia a Profesor

* ProfesorCurso
  + Atributos:  
    id, curso, profesor, coordinador
  + Métodos: getters y setters.
  + Relaciones:

curso - referencia a Curso

profesor - referencia a Profesor

Un solo profesor puede ser coordinador por curso

* Alumno
  + Atributos:  
    id, nombre, apellido1, apellido2, dni, fechaNac, direccion, localidad, grupoWhatsapp, proteccionDatos, comunicacionesComerciales, tutor
  + Métodos: getters y setters.
  + Relaciones:

tutor - referencia a Tutor

Asociado a Matricula, FaltasAsistencia

* FaltasAsistencia
  + Atributos:  
    id, fecha, curso, alumno
  + Métodos: getters y setters.
  + Relaciones:

curso - referencia a Curso

alumno - referencia a Alumno

* Matricula
  + Atributos:  
    id, fechaAlta, fechaBaja, beca, autorizacionFotos, curso, alumno
  + Métodos: getters y setters.
  + Relaciones:

curso - referencia a Curso

alumno - referencia a Alumno

* Recibo
  + Atributos:  
    id, detalle, descuento, pagado, importe, fecha, matricula
  + Métodos: getters y setters.
  + Relaciones:

matricula - referencia a Matricula

* Usuario
  + Atributos:  
    id, user, pass, rol
  + Métodos: getters y setters.
  + Relaciones:

No tiene relaciones directas en el modelo. Usado para autenticación del sistema.

* Empresa
  + Atributos:  
    id, nombre, razonFiscal, cif, direccion, localidad, provincia, telefono, email
  + Métodos: getters y setters.
  + Relaciones:

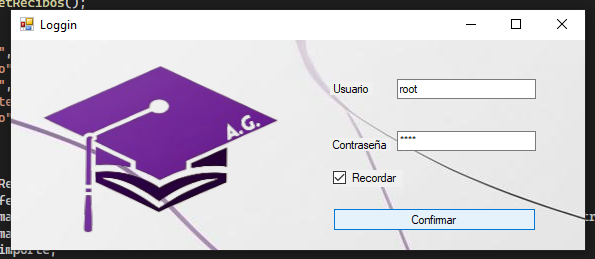
No se relaciona directamente con otras clases en el diagrama

**DIAGRAMA DE INTERFACES**

Aplicación de escritorio

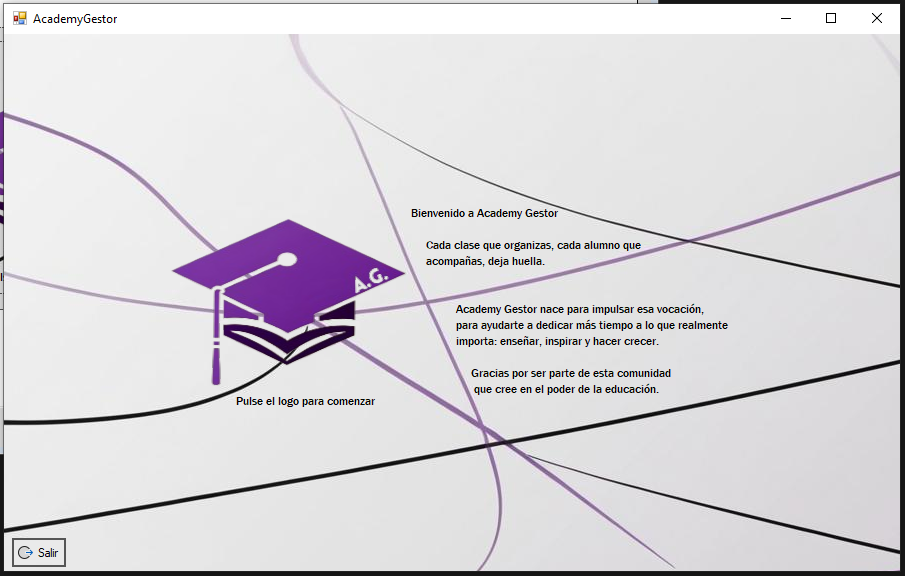
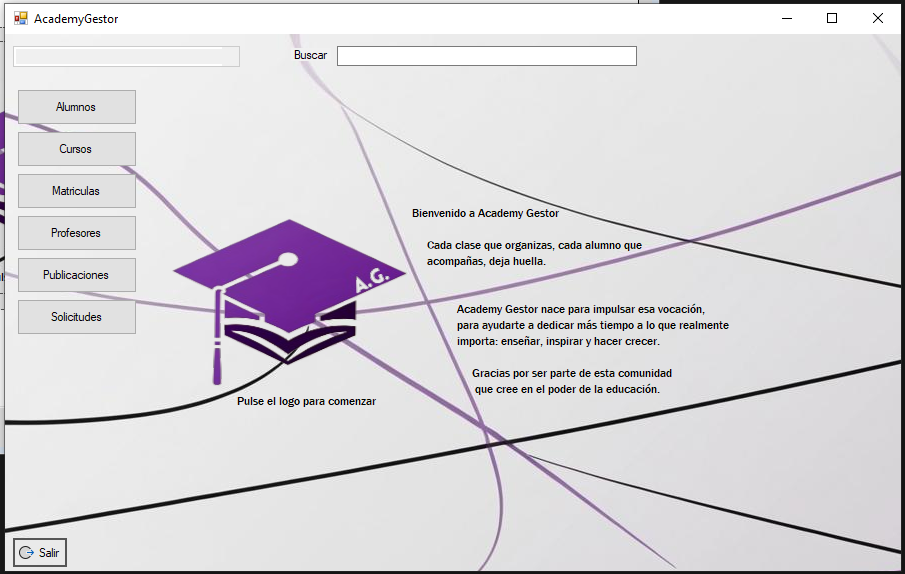
Loggin

Al confirmar se accede a la ventana principal y donde se 333333333333333desarrollará la mayoría de la actividad de la app.



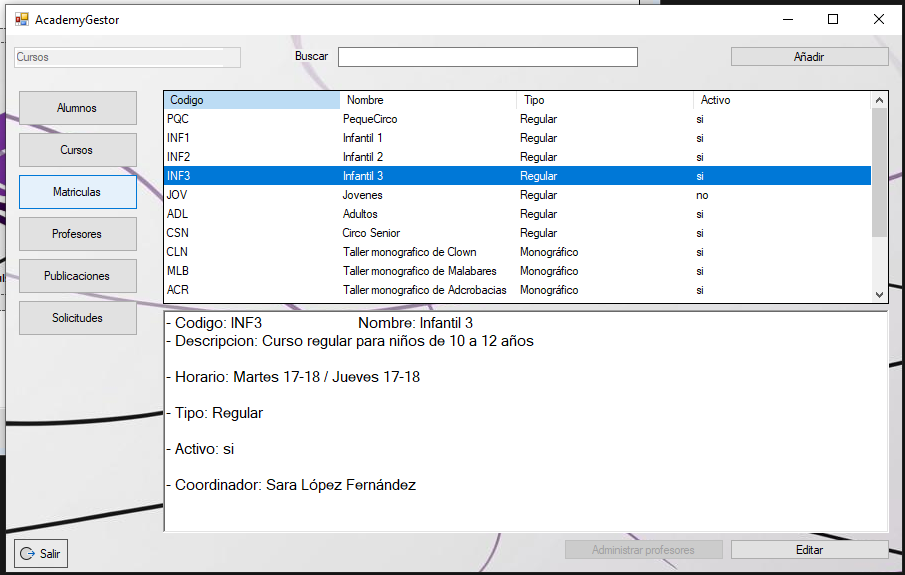
Inicio / Pantalla de bienvenida

Al pulsar sobre el logo se mostrarán las opciones iniciales que iniciarán la actividad.

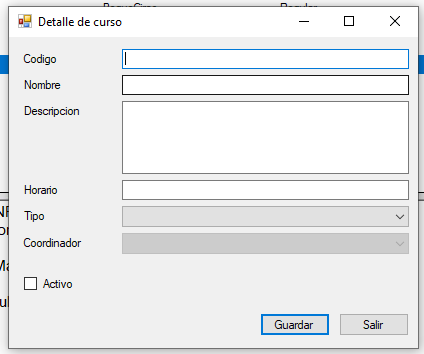
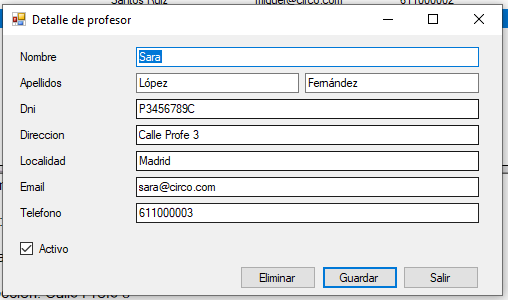
Inicio / Vista Principal

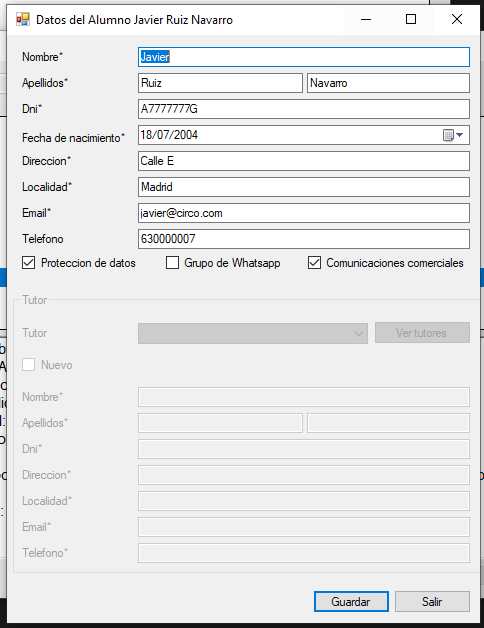
Al pulsar sobre cualquiera de las opciones se nos mostraran los datos oportunos y al seleccionar uno se mostrara en el cuadro inferior los datos del mismo.



Añadir/Editar

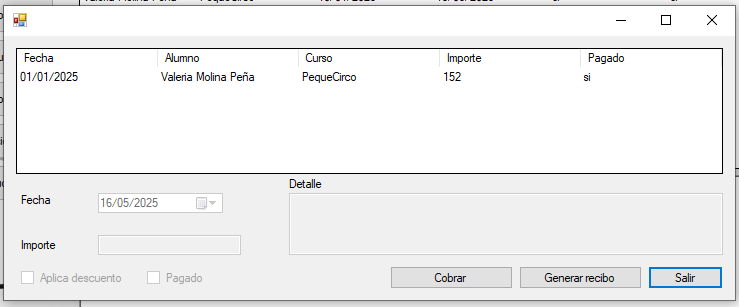
Al pulsar en la ventana principal los botones de adición o edición se abrirán las ventanas correspondientes para esos fines.

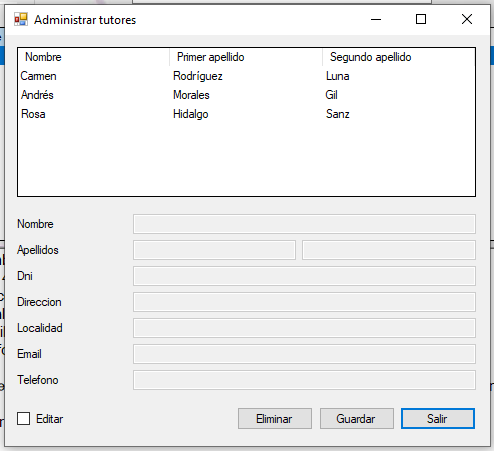
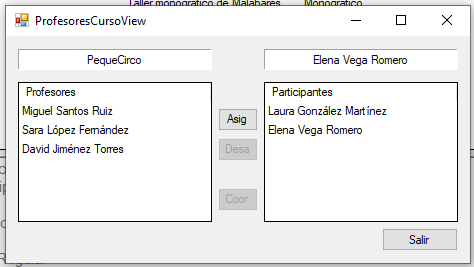
 

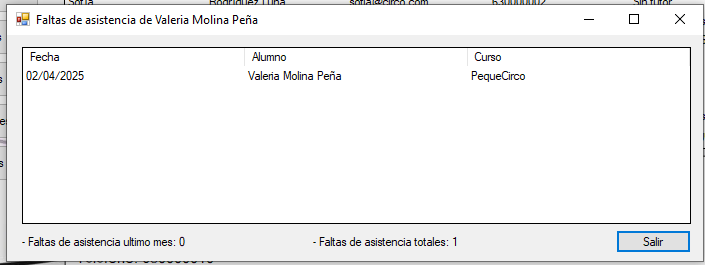


Gestión

En algunos casos tendremos opciones adicionales al seleccionar un elemento para gestionar datos relacionados con dichos elementos.



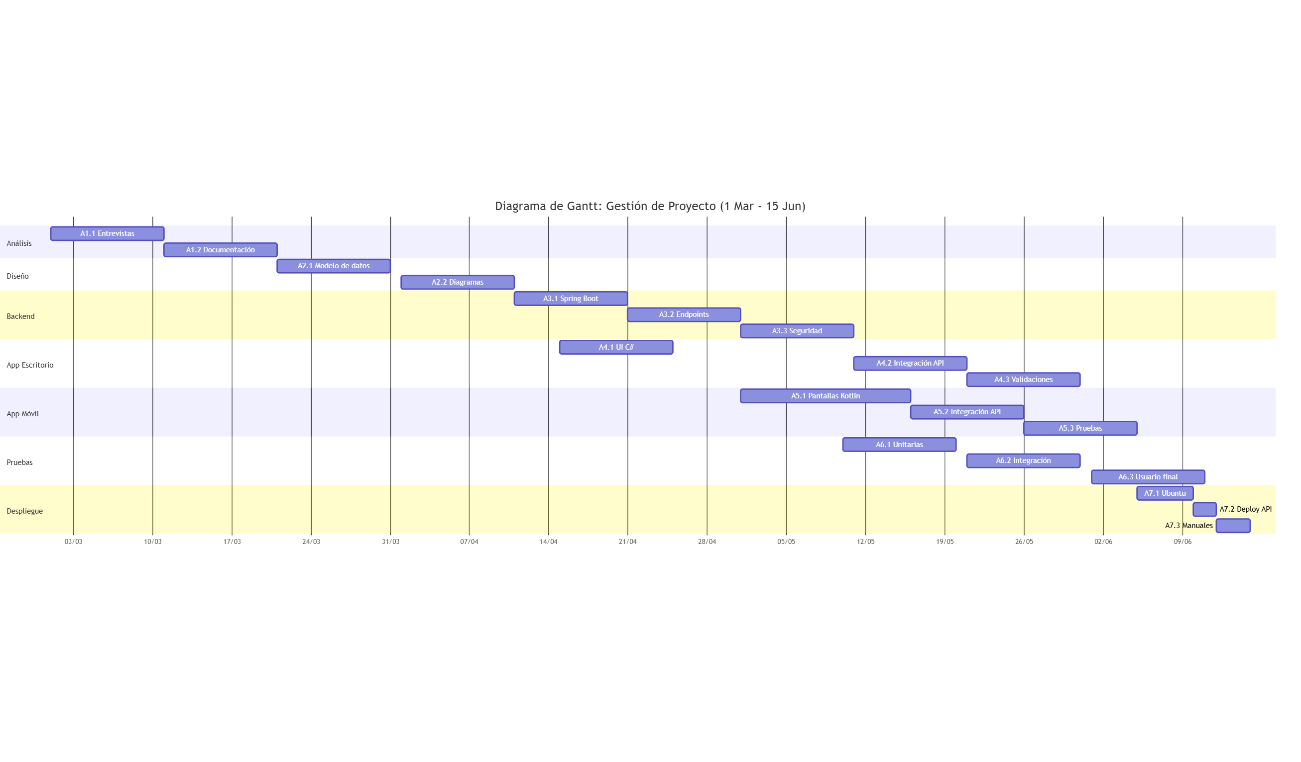


Aplicación móvil

PLANIFICACION

DEFINICIÓN DE ACTIVIDADES Y SUBTAREAS

|  |  |  |
| --- | --- | --- |
| ID | Actividad Principal | Tareas |
| A1 | Análisis y Recolección de Requisitos | A1.1 Entrevistas  A1.2 Documentación funcional |
| A2 | Diseño del Sistema | A2.1 Modelo de datos  A2.2 Diagramas de clases y arquitectura |
| A3 | Desarrollo Backend (API) | A3.1 Estructura Spring Boot  A3.2 Endpoints  A3.3 Seguridad |
| A4 | Desarrollo Aplicación de Escritorio | A4.1 UI C#  A4.2 Integración API  A4.3 Validaciones |
| A5 | Desarrollo App Móvil | A5.1 Pantallas Kotlin  A5.2 Integración API  A5.3 Pruebas |
| A6 | Pruebas del Sistema | A6.1 Unitarias  A6.2 Integración  A6.3 Usuario final |
| A7 | Despliegue y Documentación | A7.1 Instalación Ubuntu  A7.2 Deploy API  A7.3 Manuales |



IMPLEMENTACION / DESARROLLO

IDES UTILIZADOS

IntelliJ IDEA

* IDE especializado para desarrollos en Java entre otros, utilizado por compatibilidad nativa con Java y SpringBoot, ofrece herramientas avanzadas como autocompletado inteligente, análisis de código, depuración integrada, gestión de dependencias…

Visual Studio 2022

* IDE integrado de Microsoft ideal para C# y .Net, permitiendo utilizarlos para desarrollo de aplicaciones de escritorio y API Rest de forma eficiente, además de ser intuitivo para el desarrollo de interfaces.

Android Studio

* IDE oficial para el desarrollo de aplicaciones Android, basado en IntelliJ, empleado para desarrollo de aplicaciones móviles con Kotlin, además de capacidad de poder testear la aplicación en todo momento a través de emuladores y con soporte para todo el ciclo de vida de aplicaciones Android.

PLATAFORMAS DE IMPLEMENTACIÓN

Máquina virtual con Ubuntu Server

* Servidor simulado de S.O. Ubuntu Server de Linux, permite la instalación para pruebas y desarrollo de servicios como Apache2 y MySql Server, facilitando pruebas de automatización de servicios y simulación de comportamiento de servidor real.



LENGUAJES DE PROGRAMACIÓN

****

Java

* Lenguaje POO, multiplataforma, ampliamente utilizado en entornos empresariales.  
  Integrándolo con Spring Boot permite desarrollar servicios backend fiables, seguros y escalables.

C#

* Lenguaje desarrollado por Microsoft, POO y muy tipado, y perfecto para .Net. Muy utilizado para desarrollo de APIs y aplicaciones de escritorio.

Kotlin

* Lenguaje Moderno y directo, compatible con Java, utilizado para desarrollos Android, seguro contra errores comunes y alta compatibilidad con Android Studio.

FRAMEWORKS



Spring Boot

* Framework Java para desarrollos de API REST, facilita la configuración y despliegue de servicios backend, con seguridad, persistencia de datos y pruebas automáticas.

.NET

* Framework Microsoft para el desarrollo de aplicaciones multiplataforma, permite crear proyectos robustos utilizando C#, integrar eficientemente con bases de datos, interfaces gráficas y servicios web.

Gradle

* Herramienta de automatización de compilación y gestión de dependencias, especialmente usada en proyectos Java y Android, facilita la gestión de librerías, compilación de codigoy empaquetado de aplicaciones, en especial con Kotlin.



BASE DE DATOS

MySQL

* Sistema de gestión de bases de datos relacional, gran rendimiento y facilidad de uso, utilizado para almacenar y gestionar datos de forma estructurada. Su integración con Java utilizando JDBC y .NET con Entity Framework o ADO.Net lo convierte en una opción versátil para múltiples tecnologías.

CONTROL DE VERSIONES

Git

* Sistema de control de versiones distribuido, utilizado para el seguimiento de cambios en el código fuente, permite desarrollos y trabajo colaborativo, seguimiento detallado de versiones, historial de cambios y trabajo en ramas para trabajo seguro y corrección de errores, imprescindible para un trabajo organizado.

PRUEBAS

Estrategias:

* Pruebas unitarias:  
  Se desarrollarán para todas las partes lógicas clave, especialmente en la API y en la lógica de negocio de las aplicaciones de escritorio y móvil. Se usarán frameworks como JUnit y NUnit.
* Pruebas de integración:  
  Se comprobará la correcta interacción entre componentes del. Se automatizarán en parte con herramientas como Postman o RestAssured para la API.
* Pruebas de sistema:  
  Se probará el sistema completo desde el punto de vista del usuario, asegurando que los flujos funcionales se ejecutan correctamente.
* Pruebas de rendimiento:  
  Principalmente aplicadas a la API, para verificar tiempos de respuesta, carga y concurrencia.
* Pruebas de seguridad:  
  Se verificarán aspectos como autenticación, autorización y protección contra inyecciones en la API.
* Pruebas de interfaz de usuario (UI):  
  Para la aplicación de escritorio y móvil, se realizarán pruebas funcionales en la interfaz para validar la correcta disposición, comportamiento e interacción de los elementos.

Diseño y ejecución de pruebas

* Diseño de las pruebas:  
  Las pruebas se diseñan en paralelo con el desarrollo, a medida que se finaliza cada módulo o funcionalidad.

Ejecución de las pruebas:

* Las pruebas unitarias se ejecutan automáticamente con cada nueva compilación.
* Las pruebas de integración y de sistema se ejecutan al finalizar cada sprint o bloque de desarrollo.
* Las pruebas por parte de los usuarios se ejecutan al final del desarrollo y durante la fase de entrega.

Responsables de las pruebas

* Desarrollador

Evaluación por parte del cliente

Objetivos de esta evaluación son:

* Confirmar que todas las funcionalidades cumplen los requisitos iniciales.
* Detectar posibles problemas de usabilidad o errores funcionales menores.
* Recoger observaciones finales para mejoras post-entrega.

Aspectos relevantes

* Se llevará un control de errores detectados, su prioridad, y su estado (pendiente / resuelto).
* Se aplicará control de versiones para garantizar la trazabilidad de las pruebas respecto a los cambios realizados.
* El sistema no se considera “apto para entrega” hasta que no supere el 100% de los casos de prueba críticos.