Manual Técnico

- ## Roles / integrantes

  Lucas Fernando Olguin Rojas - TEAM LEADER / FULLSTACK DEV
  Joel Alexander Illanes Caballero - FULLSTACK DEV / QA
  Cristhian Flores Herrera - DBMASTER / FULLSTACK DEV
  Kevin Eduardo Alanes Ortega - FULLSTACK DEV / QA
  Kevin Danny Prieto de la Zerda - FULLSTACK DEV
  Victor Alejandro Mollo Huarita - FULLSTACK DEV

- ## Introducción:

  Muchos emprendedores y estudiantes tienen ideas innovadoras, pero carecen del apoyo profesional y las conexiones necesarias para llevarlas a cabo. Las plataformas de crowdfunding tradicionales se centran unicamente en el financiamiento, dejando sin atender estas necesidades criticas de orentacion y networking. IncUVa Lab CROWDFUNDING es una plataforma de apoyo no monetario que conectar proyectos con profesionales dispuestos a ofrecer seguimiento, contactos y colaboracion basada en la identificacion con la vision del proyecto.

- ## Descripción del proyecto:

  IncUVa Lab CROWDFUNDING es una plataforma de apoyo a proyectos innovadores que se diferencia del crowdfunding tradicional. El sistema no utiliza dinero como el crowdfunding tradicional. En su lugar, los proyectos reciben apoyo a traves de seguimiento, contactos y colaboracion de profesionales que se identifican con su vision.

  El sistema permite gestionar proyectos con informacion detallada incluyendo descripciones, imagenes, videos, documentos, categorias y asesorias solicitadas. Los proyectos pueden solicitar diferentes tipos de asesorias, especializadas, tales como asesoria financiera, comercial, en marketing, investigacion de mercados, gestion de equipos, entre otras.

  El objetivo principal es potenciar proyectos mediante networking, ofreciendo a los emprendedores un espacio donde pueden:

  - Crear campañas para sus proyectos
  - Recibir asesorias especializadas en diferentes areas (financiera, comercial, marketing, gestion de equipos, etc.)
  - Conectar con profesionales y colaboradores
  - Obtener seguimiento para el desarrollo de sus iniciativas

- ## Link al Video demostrativo YouTube

  https://www.youtube.com/watch?v=viwSR0zcCAI

- **Listado de los Requisitos Funcionales del Sistema**
    - Registro de cambio y olvido de contraseña
    - Autenticacion con Token y validacion de 2 pasos
    - Compatible con dispositivos movil y desktop
    - Cambios de contrasenas desde el perfil
    - Validaciones entre Docente y Admin
    - Implementacion de historial de llaves y modificaciones
    - Nombre del Docente debajo del aula a prestar llave
    - Descripcion de colores en el mapa de aulas
    - Recoleccion de atributos de las aulas
    - Documentacion oficial de requerimientos
    - Organizar el Excel de Profesores
    - Interfaz de prestamos externos conectada a la BDD
    - Interfaz de usuarios conectada a la BDD
    - Interfaz de aulas conectada a la BDD
    - Implementacion de notificaciones
    - Conexion de la BDD
    - Dashboard de docentes
    - Dashboard del administrador
    - Splash Screen
    - Interfaz del dashboard del administrador
    - Paleta de colores y tipografia
    - Roles y permisos de usuarios

- Objetivos y funcionalidades principales
- Entorno de desarrollo
- Arquitectura de software
- Diseno de la BDD
- Prototipo de interfaz para administracion de llaves
- Funcionalidades principales de la app

- ## Arquitectura del software:

  EEl sistema implementa una arquitectura de tres capas compuesta por un frontend desarrollado en React con Vite, un backend construido con Node.js y Express, y una base de datos MySQL. Toda la aplicación está containerizada mediante Docker Compose, lo que permite el despliegue orquestado de los tres servicios principales: base de datos, backend y frontend.

  Estructura del Proyecto

  Organización de Directorios

  El proyecto se organiza en las siguientes carpetas principales:

  - **src/**: Código fuente del frontend.
  - **server/**: Código del backend.
  - **BaseDeDatos/**: Scripts SQL para inicialización de la base de datos.
  - **public/**: Archivos estáticos públicos.

  Tecnologías Principales

  Frontend:

  - React y React DOM
  - React Router DOM
  - Axios

- Vite

Backend:

- Express
- MySQL2
- Nodemailer
- Bcrypt
- Speakeasy (2FA)
- Multer
- Node-cron

Componentes Principales

Frontend (React)

Estructura de Componentes

El frontend está organizado en:

Layouts:

- MainLayout como contenedor principal de las páginas públicas.

Componentes:

- PrivateRoute para protección de rutas según autenticación y roles.
- Header y Footer para navegación.
- Componentes específicos en subdirectorios.

Vistas:

- Vistas públicas: Home, About, FAQ, Login, Register, ExplorarProyectos, VerProyecto.

- Vistas protegidas: Profile, CrearProyecto, EditarProyectoUsuario.

- Vistas administrativas: UsersAdmin, CrudProyectos, AceptarProyectos, ProyectosRemovidos, CategoriasAdmin, AsesoriasAdmin.

Servicios:

- Servicio centralizado para peticiones HTTP al backend.

Sistema de Ruteo

El routing define:

- Rutas públicas accesibles para todos.

- Rutas privadas que requieren autenticación.

- Rutas administrativas restringidas a roles específicos.

Backend (Express)

Arquitectura MVC

El backend sigue una arquitectura basada en modelos, controladores y rutas.

Modelos:

- Manejan acceso a datos y operaciones CRUD.

Controladores:

- Contienen la lógica de negocio.

Rutas:

- Exponen endpoints de la API organizados por recursos: usuarios, proyectos, categorías, asesorías, autenticación 2FA y relaciones entre usuarios y proyectos.

Configuración del Servidor

El servidor Express incluye:

- Configuración de CORS.
- Middleware para parsing de JSON.
- Servir archivos estáticos.
- Manejo de diversos tipos de contenido como imágenes y documentos.

Conexión a Base de Datos

La conexión a MySQL se gestiona mediante un pool de conexiones con reintentos automáticos.

Sistema de Tareas Programadas

Se ejecuta un cron job diariamente para:

- Revisar proyectos vencidos.
- Marcar proyectos incompletos automáticamente.
- Enviar notificaciones por correo a sus creadores.

Base de Datos (MySQL)

Esquema Relacional

La base de datos contiene las siguientes entidades:

Tablas principales:

- Rol
- Usuario
- Categoria
- Proyecto
- Asesoria

Tablas de relación:

- Proyecto_Asesoria
- Usuario_Proyecto
- Proyecto_Imagen
- Proyecto_Documento
- Proyecto_RedSocial

Tablas de auditoría:

- Usuario_Historial
- Proyecto_Historial

Interacciones Entre Componentes

- Flujo de autenticación.
- Flujo de creación de proyectos.
- Autenticación de dos factores mediante TOTP.
- Comunicación frontend–backend mediante servicio centralizado.
- Gestión de autenticación en frontend con localStorage y eventos personalizados.

Patrones de Diseño Implementados

- Patrón MVC
- Patrón Repository
- Patrón HOC (Higher-Order Component) para protección de rutas
- Patrón Singleton para pool de conexiones
- Patrón Middleware Chain

- Patrón Observer mediante eventos
- Patrón Strategy para almacenamiento de archivos con Multer
- Patrón Facade en el servicio de API

Características Adicionales

Manejo de Archivos Multimedia

Soporta la subida y gestión de imágenes y documentos en múltiples formatos.

Notificaciones por Correo

Usa Nodemailer para recuperación de contraseñas y alertas de proyectos incompletos.

Gestión de Estados de Proyecto

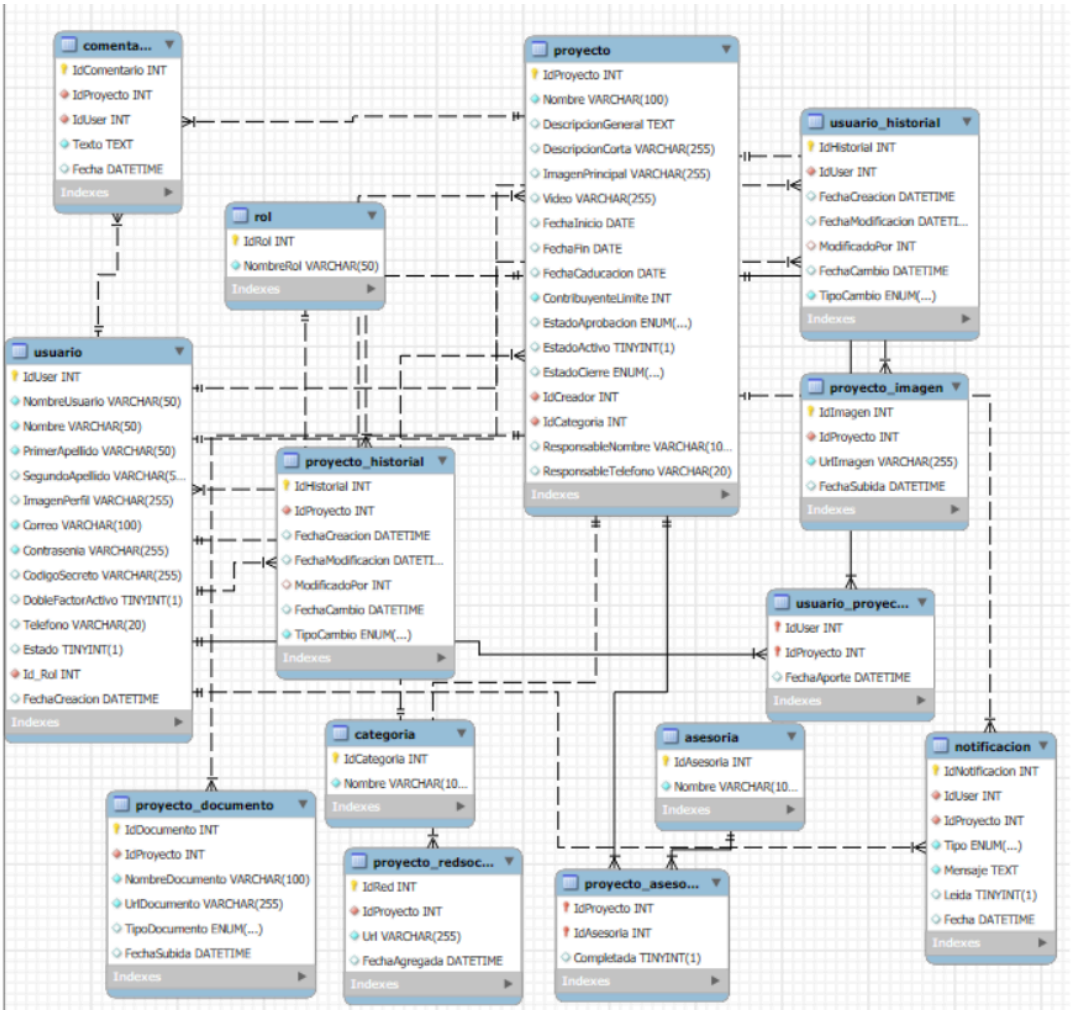
Incluye estados de aprobación, activación y cierre de proyectos.

Control de Acceso Basado en Roles (RBAC)

Define dos roles principales:

- **Admin:** acceso completo a funcionalidades administrativas.
- **Usuario:** acceso a funcionalidades estándar.

- **Base de datos**
  - **Diagrama completo y actual**



- **En el GIT una carpeta con la base de datos con script de generación e inserción de datos de ejemplo utilizados**

Tabla: Roles

```
CREATE TABLE Rol (
    IdRol INT AUTO_INCREMENT PRIMARY KEY,
    NombreRol VARCHAR(50) NOT NULL UNIQUE
);

INSERT INTO Rol (NombreRol) VALUES
('Admin'),
('Usuario');
```

Tabla: Usuarios

```
CREATE TABLE Usuario (
    IdUser INT AUTO_INCREMENT PRIMARY KEY,
    NombreUsuario VARCHAR(50) NOT NULL UNIQUE,
    Nombre VARCHAR(50) NOT NULL,
    PrimerApellido VARCHAR(50) NOT NULL,
    SegundoApellido VARCHAR(50),
    ImagenPerfil VARCHAR(255),
    Correo VARCHAR(100) NOT NULL UNIQUE,
    Contrasenia VARCHAR(255) NOT NULL,
    CodigoSecreto VARCHAR(255),
    DobleFactorActivo BOOLEAN DEFAULT FALSE,
    Telefono VARCHAR(20),
    Estado BOOLEAN DEFAULT TRUE,
    Id_Rol INT NOT NULL,
    FechaCreacion DATETIME DEFAULT
CURRENT_TIMESTAMP,
    FOREIGN KEY (Id_Rol) REFERENCES Rol(IdRol)
);

INSERT INTO Usuario (
    NombreUsuario, Nombre, PrimerApellido,
SegundoApellido,
    ImagenPerfil, Correo, Contrasenia, Id_Rol
) VALUES
(
    'admin', 'Administrador', 'General', '',
NULL,
    'admin@example.com',
    'admin123',
    1
),
(
    'jrojas', 'Jorge', 'Rojas', 'Mendoza', NULL,
    'jorge.rojas@example.com',
    'usuario123',
    2
);
```

Tabla: Categorias

```
CREATE TABLE Categoria (
    IdCategoria INT AUTO_INCREMENT PRIMARY KEY,
    Nombre VARCHAR(100) NOT NULL UNIQUE
);

INSERT INTO Categoria (Nombre) VALUES
('Tecnologia'),('Innovacion'),('Educacion'),('Cie
ncia'),('Ingenieria'),
('Salud'),('Medio Ambiente'),('Arte y
Cultura'),('Emprendimiento'),
('Software'),('Robotica'),('Inteligencia
Artificial'),('Electronica'),
('Social'),('Biotecnologia');
```

Tabla: Proyectos

```
CREATE TABLE Proyecto (
    IdProyecto INT AUTO_INCREMENT PRIMARY KEY,
    Nombre VARCHAR(100) NOT NULL,
    DescripcionGeneral TEXT,
    DescripcionCorta VARCHAR(255),
    ImagenPrincipal VARCHAR(255),
    Video VARCHAR(255),
    FechaInicio DATE,
    FechaFin DATE,
    FechaCaducacion DATE,
    ContribuyenteLimite INT NOT NULL,
    EstadoAprobacion
ENUM('Pendiente','Activo','Cancelado') DEFAULT
'Pendiente',
    EstadoActivo BOOLEAN DEFAULT TRUE,
    EstadoCierre
ENUM('Activo','FinalizadoExitoso','FinalizadoInco
mpleto') DEFAULT 'Activo',
    IdCreador INT NOT NULL,
    IdCategoria INT NOT NULL,
    ResponsableNombre VARCHAR(100),
    ResponsableTelefono VARCHAR(20),
```

```
    FOREIGN KEY (IdCreador) REFERENCES
Usuario(IdUser) ON DELETE CASCADE,
    FOREIGN KEY (IdCategoria) REFERENCES
Categoria(IdCategoria)
);
```

Tabla: Asesorias

```
CREATE TABLE Asesoria (
    IdAsesoria INT AUTO_INCREMENT PRIMARY KEY,
    Nombre VARCHAR(100) NOT NULL UNIQUE
);

INSERT INTO Asesoria (Nombre) VALUES
('Asesoria financiera'),('Asesoria
comercial'),('Asesoria en marketing'),
('Apoyo en valoracion de empresas'),('Asesoria en
gestion impositiva'),
('Asesoria en desarrollo de marca'),('Apoyo en
desarrollo de packaging'),
('Apoyo en procesos de innovacion'),('Asesoria en
investigacion de mercados'),
('Asesoria en procesos de
exportacion'),('Asesoria en procesos de
importacion'),
('Asesoria en apertura de mercados
internacionales'),('Asesoria en gestion de
equipos'),
('Asesoria en elevator pitch'),('Asesoria en
costos y estructura contable');
```

Tabla: Proyecto_Asesoria

```
CREATE TABLE Proyecto_Asesoria (
    IdProyecto INT NOT NULL,
    IdAsesoria INT NOT NULL,
    Completada BOOLEAN DEFAULT FALSE,
    PRIMARY KEY (IdProyecto, IdAsesoria),
```

```
    FOREIGN KEY (IdProyecto) REFERENCES
Proyecto(IdProyecto) ON DELETE CASCADE,
    FOREIGN KEY (IdAsesoria) REFERENCES
Asesoria(IdAsesoria) ON DELETE CASCADE
);
```

Tabla: Proyecto_Imagen

```
CREATE TABLE Proyecto_Imagen (
    IdImagen INT AUTO_INCREMENT PRIMARY KEY,
    IdProyecto INT NOT NULL,
    UrlImagen VARCHAR(255) NOT NULL,
    FechaSubida DATETIME DEFAULT
CURRENT_TIMESTAMP,
    FOREIGN KEY (IdProyecto) REFERENCES
Proyecto(IdProyecto) ON DELETE CASCADE
);
```

Tabla: Proyecto_Documento

```
CREATE TABLE Proyecto_Documento (
    IdDocumento INT AUTO_INCREMENT PRIMARY KEY,
    IdProyecto INT NOT NULL,
    NombreDocumento VARCHAR(100) NOT NULL,
    UrlDocumento VARCHAR(255) NOT NULL,
    TipoDocumento
ENUM('PDF','DOCX','XLSX','Otro') DEFAULT 'PDF',
    FechaSubida DATETIME DEFAULT
CURRENT_TIMESTAMP,
    FOREIGN KEY (IdProyecto) REFERENCES
Proyecto(IdProyecto) ON DELETE CASCADE
);
```

Tabla: Usuario_Proyecto

```
CREATE TABLE Usuario_Proyecto (
    IdUser INT NOT NULL,
    IdProyecto INT NOT NULL,
```

```
    FechaAporte DATETIME DEFAULT
CURRENT_TIMESTAMP,
    PRIMARY KEY (IdUser, IdProyecto),
    FOREIGN KEY (IdUser) REFERENCES
Usuario(IdUser) ON DELETE CASCADE,
    FOREIGN KEY (IdProyecto) REFERENCES
Proyecto(IdProyecto) ON DELETE CASCADE
);
```

Tabla: Comentario

```
CREATE TABLE Comentario (
    IdComentario INT AUTO_INCREMENT PRIMARY KEY,
    IdProyecto INT NOT NULL,
    IdUser INT NOT NULL,
    Texto TEXT NOT NULL,
    Fecha DATETIME DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (IdProyecto) REFERENCES
Proyecto(IdProyecto) ON DELETE CASCADE,
    FOREIGN KEY (IdUser) REFERENCES
Usuario(IdUser) ON DELETE CASCADE
);
```

Tabla: Notificacion

```
CREATE TABLE Notificacion (
    IdNotificacion INT AUTO_INCREMENT PRIMARY
KEY,
    IdUser INT NOT NULL,
    IdProyecto INT NOT NULL,
    Tipo
ENUM('Aporte','Comentario','MetaAlcanzada','Proye
ctoCerrado') NOT NULL,
    Mensaje TEXT NOT NULL,
    Leida BOOLEAN DEFAULT FALSE,
    Fecha DATETIME DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (IdUser) REFERENCES
Usuario(IdUser) ON DELETE CASCADE,
```

```
    FOREIGN KEY (IdProyecto) REFERENCES
Proyecto(IdProyecto) ON DELETE CASCADE
);
```

Tabla: Usuario_Historial

```
CREATE TABLE Usuario_Historial (
    IdHistorial INT AUTO_INCREMENT PRIMARY KEY,
    IdUser INT NOT NULL,
    FechaCreacion DATETIME,
    FechaModificacion DATETIME,
    ModificadoPor INT,
    FechaCambio DATETIME DEFAULT
CURRENT_TIMESTAMP,
    TipoCambio ENUM('Insert','Update','Delete')
NOT NULL,
    FOREIGN KEY (IdUser) REFERENCES
Usuario(IdUser) ON DELETE CASCADE,
    FOREIGN KEY (ModificadoPor) REFERENCES
Usuario(IdUser) ON DELETE SET NULL
);
```

Tabla: Proyecto_Historial

```
CREATE TABLE Proyecto_Historial (
    IdHistorial INT AUTO_INCREMENT PRIMARY KEY,
    IdProyecto INT NOT NULL,
    FechaCreacion DATETIME,
    FechaModificacion DATETIME,
    ModificadoPor INT,
    FechaCambio DATETIME DEFAULT
CURRENT_TIMESTAMP,
    TipoCambio ENUM('Insert','Update','Delete')
NOT NULL,
    FOREIGN KEY (IdProyecto) REFERENCES
Proyecto(IdProyecto) ON DELETE CASCADE,
    FOREIGN KEY (ModificadoPor) REFERENCES
Usuario(IdUser) ON DELETE SET NULL
);
```

Tabla: Proyecto_RedSocial

```
CREATE TABLE Proyecto_RedSocial (
    IdRed INT AUTO_INCREMENT PRIMARY KEY,
    IdProyecto INT NOT NULL,
    Url VARCHAR(255) NOT NULL,
    FechaAgregada DATETIME DEFAULT
CURRENT_TIMESTAMP,
    FOREIGN KEY (IdProyecto) REFERENCES
Proyecto(IdProyecto) ON DELETE CASCADE
);
```

- **Listado de Roles más sus credenciales de todos los Admin / Users del sistema**

  Administrador = admin@example.com Contraseña = admin123
  Usuario = jorge.rojas@example.com Contraseña = usuario123

- **Requisitos del sistema:**

  Requerimientos de Hardware (minimo) – Cliente

  - Procesador de doble nucleo a 2.0 GHz o superior
  - Memoria RAM de 4 GB (8 GB recomendado)
  - Almacenamiento de 500 MB libres
  - Conexion a internet de banda ancha (5 Mbps minimo)
  - Pantalla con resolucion minima de 1024x768
  - Navegador web moderno compatible con ES6+ y JavaScript moderno

  Requerimientos de Software – Cliente

  - React 19.1.1 o superior
  - Vite 7.1.7 como herramienta de desarrollo
  - Node.js con soporte para ES Modules
  - axios para solicitudes HTTP
  - react-router-dom para enrutamiento
  - react-icons para iconografia
  - Navegador moderno como Chrome, Firefox, Edge, Safari u Opera
  - Font Awesome 6.4.0 (via CDN)
  - Node.js 18 o superior para desarrollo local

● npm o yarn para gestion de paquetes
● Ejecucion por defecto en puerto 5173

Requerimientos de Hardware – Servidor / Hosting / Base de Datos

● Procesador de dos nucleos a 2.4 GHz o superior (4 nucleos recomendado)
● Memoria RAM minima de 2 GB (4 GB recomendado para produccion)
● Contenedores sugeridos (si se usa Docker):
  – Frontend: 512 MB
  – Backend: 512 MB
  – MySQL: 1 GB
● Almacenamiento minimo de 10 GB para sistema y aplicaciones
● Espacio adicional para:
  – Archivos subidos (imagenes y documentos)
  – Logs del sistema
  – Backups de base de datos
● Conexion a internet estable para atender solicitudes y servicios de correo
● Puertos necesarios: 5173 (frontend), 4000 (backend), 3306/3307 (MySQL)

Requerimientos de Software – Servidor / Hosting / Base de Datos

● Node.js 18 LTS o superior
● npm v9 o superior
● Express.js 5.1.0 como servidor web
● cors para CORS
● dotenv para variables de entorno
● mysql2 como driver de base de datos
● axios como cliente HTTP
● nodemailer para envio de correos
● node-cron para tareas programadas
● Docker y Docker Compose (opcional pero recomendado para despliegue)
● MySQL 8.0 o superior con:
  – Charset utf8mb4
  – Collation utf8mb4_unicode_ci
  – Pool de hasta 10 conexiones concurrentes
● Estructura de BD con tablas relacionales para usuarios, proyectos, categorias, asesorias, archivos y registros historicos
● Red de contenedores, health checks y volumenes persistentes (si se usa Docker)

- ## Instalación y configuración:

Requerimientos Previos

Software necesario

- Node.js
- MySQL 8.0
- Docker y Docker Compose (opcional)
- Git

Hardware recomendado

- 4GB de RAM
- 2GB de espacio en disco
- Conexion a internet

Configuracion de la Base de Datos

Credenciales con Docker

- Usuario root
- Contraseña root
- Base de datos incuvalab
- Usuario de aplicacion user
- Contraseña incuvalab
- Puerto externo 3307
- Charset utf8mb4
- Collation utf8mb4_unicode_ci

Credenciales para desarrollo local

- Host localhost
- Usuario root
- Contraseña STARKILLER1475369
- Base de datos incuvalab
- Puerto 3306

Configuracion adicional

- Pool de conexiones habilitado

Estructura de la Base de Datos

Tablas principales

- Rol
- Usuario
- Categoria
- Proyecto
- Asesoria

Campos principales del usuario

- Correo
- Contraseña
- Autenticacion 2FA
- Nombre y apellidos
- Imagen
- Telefono
- Estado

- Rol
- Fecha de creacion

Usuario administrador por defecto

- Usuario admin
- Correo landeschris10@gmail.com
- Contraseña hasheada

Campos principales del proyecto

- Estados: Pendiente, Activo, Cancelado
- Estados de cierre: Activo, FinalizadoExitoso, FinalizadoIncompleto
- Fechas de inicio, fin y caducacion
- Limite de contribuyentes

Tablas relacionales

- Proyecto_Asesoria
- Proyecto_Imagen
- Proyecto_Documento
- Usuario_Proyecto
- Comentario
- Notificacion
- Proyecto_RedSocial

Configuracion del Servidor Backend

Ajustes principales

- Puerto por defecto 4000
- CORS habilitado
- Parser JSON con limite de 10MB
- Archivos estaticos en /uploads

Configuracion de correo

- Servicio Gmail
- Usuario landesl502@gmail.com
- Contraseña de aplicacion protegida

Tarea programada

- Revisa proyectos incompletos a diario
- Envia notificaciones automaticas

Rutas principales cargadas

- Usuarios
- Proyectos
- Categorias
- Asesorias
- Autenticacion 2FA

Endpoints del Servidor

Usuarios

- Registrar

- Login
- Verificar 2FA
- Enviar token de recuperacion
- Validar token
- Restablecer contraseña
- Editar perfil

Gestion de usuarios

- Listar
- Obtener usuario
- Crear
- Actualizar
- Eliminar logico
- Restaurar

Proyectos

- Listar todos
- Listar pendientes
- Listar removidos
- Listar destacados
- Proyectos creados por usuario
- Proyectos donados por usuario
- Detalle del proyecto
- Busqueda por nombre o categoria

CRUD de proyectos

- Crear
- Editar
- Eliminar logico
- Restaurar
- Eliminar permanente

Aprobacion

- Aprobar proyecto
- Cancelar proyecto

Interaccion

- Me gusta
- Quitar me gusta

Categorias

- Listar
- Crear
- Editar
- Eliminar

Asesorias

- Listar
- Crear
- Editar
- Eliminar

Proyectos para usuarios

- Obtener proyecto con asesorias
- Editar proyecto como usuario

Healthcheck

- Verificar API
- Verificar base de datos

2FA

- Generar secreto y QR
- Verificar codigo
- Desactivar 2FA
- Consultar estado

Configuracion del Frontend

General

- React con Vite
- Puerto de desarrollo 5173
- Hot Module Replacement
- Variables de entorno para API
- Scripts de desarrollo, build, preview y lint

Dependencias Principales

Backend

- express
- mysql2

- cors
- dotenv
- nodemailer
- axios

Frontend

- react
- react-dom
- react-router-dom
- react-icons

Desarrollo

- vite
- plugin de React
- eslint
- Typescript types

Detalles Adicionales y Recomendaciones

Docker Compose

- MySQL con healthcheck, carga automatica y puerto 3307
- Backend en puerto 4000, con nodemon y volumenes
- Frontend en puerto 5173 con HMR
- Servicios conectados y dependencias correctas

Validaciones

- Email obligatorio
- Contraseña entre 8 y 20 caracteres
- Telefono boliviano +591
- Hash de contraseñas con bcrypt
- Autenticacion de dos factores con secreto y QR

Limites de archivos

- Imagen principal: 1
- Imagenes adicionales: 10
- Documentos: 5
- Tipos aceptados: PDF, DOCX, XLSX, PNG, JPG, JPEG, GIF

Restricciones de fechas

- Maximo de 30 dias para la fecha de finalizacion

Usuarios de prueba

- juanperez@incuvalab.com
- maria.rodriguez@incuvalab.com
- carlos.garcia@incuvalab.com
- ana.fernandez@incuvalab.com
- luis.ramirez@incuvalab.com

Proyectos precargados

- 5 proyectos distribuidos en Tecnologia, Educacion, Salud, Arte y Cultura, Robotica

Recomendaciones finales

- Crear archivo .env
- Cambiar contraseñas en produccion
- Configurar CORS
- Mantener utf8mb4
- Configurar backups
- Revisar logs regularmente

- **GIT :**

  Versión final entregada del proyecto.

  https://github.com/C4BALLERO/PS-II---IncubaLab

  Rama: dev

- **Dockerizado**

  Base de Datos (MySQL)

  - Motor: MySQL 8.0
  - Contenedor: **mysql_db**
  - Puerto: **3307 (host) → 3306 (contenedor)**
  - Inicialización automática con los scripts SQL ubicados en BaseDeDatos/
  - Charset configurado en **utf8mb4** para manejo correcto de caracteres especiales y emojis

  Backend (Express/Node.js)

  - Contenedor: **express_api_Incuvalab**
  - Puerto expuesto: **4000**
  - El servicio espera a que MySQL esté "saludable" antes de iniciarse

- Utiliza **Nodemon** en desarrollo para recarga automática
- Volúmenes montados para código fuente y archivos subidos

Frontend (React/Vite)

- Contenedor: **react_frontend**
- Puerto expuesto: **5173**
- Depende del backend para funcionar correctamente
- Construido con **Vite** para desarrollo y producción

Proceso de Dockerizado y Configuración

Red de Contenedores

- Todos los servicios comparten una red personalizada llamada **app_network**, permitiendo comunicación interna sin exponer servicios innecesarios al exterior.

Healthcheck de Base de Datos

- El contenedor MySQL ejecuta verificaciones de salud cada 5 segundos para garantizar que la BD esté lista antes de aceptar conexiones del backend.

Inicialización Automática

- Durante la primera ejecución, cualquier archivo .sql en el directorio BaseDeDatos/ se ejecuta automáticamente para crear tablas, insertar datos o configurar parámetros iniciales.

Cómo Ejecutar el Proyecto

Comando de Ejecución Principal

```
docker-compose up
```

Variables de Entorno del Backend

- Se configuran mediante el archivo .env del directorio server/
- Incluyen host, usuario, contraseña y nombre de la base de datos

Conexión a la Base de Datos

- El backend tiene valores por defecto, pero pueden sobrescribirse mediante .env
- Incluye un sistema de reconexión automática con **hasta 10 intentos**, útil cuando la base de datos tarda en levantarse

Acceso y Credenciales

Credenciales del Entorno Docker

- Usuario root: **root**
- Contraseña root: **root**
- Usuario de aplicación: **user**
- Contraseña de aplicación: **incuvalab**
- Base de datos principal: **incuvalab**

Credenciales de Usuarios del Sistema

Usuario Administrador

- Usuario: **admin**
- Correo: **admin@example.com**
- Contraseña: **admin123**
- Rol asignado: **Administrador (Id_Rol = 1)**

Usuario de Ejemplo

- Usuario: **jrojas**

- Correo: **jorge.rojas@example.com**
- Contraseña: **usuario123**
- Rol asignado: **Usuario (Id_Rol = 2)**

Roles del Sistema

Roles Disponibles

- **Administrador (IdRol = 1):** Acceso completo al sistema
- **Usuario (IdRol = 2):** Acceso estándar y limitado

Estructura del Usuario

El registro de usuarios incluye:

- Autenticación con contraseña
- Segundo factor opcional
- Estado activo o inactivo
- Asociación obligatoria a un rol
- Campos de perfil e información personal

Base de Datos con Datos Válidos

Categorías de Proyectos (15)

Incluyen:

Tecnologia, Innovacion, Educacion, Ciencia, Ingenieria, Salud, Medio Ambiente, Arte y Cultura, Emprendimiento, Software, Robotica, Inteligencia Artificial, Electronica, Social, Biotecnologia.

Asesorías Disponibles (15)

Ejemplos:

Asesoría financiera, marketing, investigacion de mercados, exportacion/importacion, desarrollo de marca, innovacion, apertura de mercados internacionales, gestion de equipos, elevator pitch, costos y estructuras contables, entre otras.

Proyectos de Ejemplo (5)

- **EcoTech Innovacion** – Soluciones tecnológicas sostenibles
- **Aprendizaje Inteligente** – Plataforma educativa con IA
- **BioSalud Avanzada** – Aplicaciones biotecnológicas
- **Arte y Tecnologia** – Cultura digital e interactividad
- **Robotica Educativa** – Robots educativos

Cada proyecto incluye descripciones completas, imágenes, videos, documentos, redes sociales y asesorías relacionadas.

- ## Personalización y configuración:

Variables de Entorno de Base de Datos

- Conexion configurada mediante variables de entorno.
- DB_HOST define la direccion del servidor.
- DB_USER define el usuario de la base.
- DB_PASSWORD define la contraseña.
- DB_NAME define la base de datos usada.
- DB_PORT define el puerto de conexion.
- Conexion con limite de 10 conexiones simultaneas.
- Configuracion de caracteres UTF8MB4.

Configuracion de Docker Compose

Base de Datos (MySQL):

- Creacion automatica de la base con charset utf8mb4.

- Variables para usuario, contraseña y permisos.
- Puerto externo asignado.
- Volumen para persistencia de datos.

Backend (API Express):

- Puerto definido por variable o valor predeterminado.
- Archivo .env para configuraciones internas.
- Volumen montado para el codigo.
- Volumen para carpeta de archivos subidos.
- Dependencia directa del servicio de la base de datos.

Frontend (React):

- Puerto expuesto para el entorno de desarrollo.
- Dependencia del servicio backend.
- Volumen para actualizacion inmediata del codigo.

Configuracion del Servidor (Backend)

CORS:

- Origen permitido definido mediante variable.
- Metodos habilitados GET, POST, PUT, PATCH y DELETE.

Middlewares:

- Manejo de JSON con limite de 10mb.
- Carpeta publica para archivos subidos.
- Tipos de contenido configurados para documentos e imagenes.

Nodemailer:

- Servicio configurado con Gmail.
- Credenciales desde variables de entorno.
- Envio de correos para notificaciones y recuperacion.

Cron Job:

- Ejecucion diaria a medianoche.
- Verificacion de proyectos incompletos.

Puerto del Servidor:

- Definido por variable PORT o valor predeterminado.

Configuracion del Frontend

- URL base del API definida con variable de entorno.
- Vite configurado para conexiones desde Docker.
- Sistema de recarga en vivo.
- Deteccion de cambios mediante polling.

Autenticacion de Doble Factor (2FA)

- Codigo secreto de 20 caracteres.
- Nombre de la aplicacion incluye el ID del usuario.
- Margen de verificacion en caso de desfase de tiempo.
- Generacion y verificacion mediante libreria dedicada.

Subida de Archivos (Multer)

- Carpeta de destino: “uploads/”.

- Tipos permitidos: PDF, DOCX, XLSX, PNG, JPG, JPEG, GIF.
- Rechazo automatico de tipos no validos.
- Nombre de archivo gestionado para evitar conflictos.

Validaciones del Sistema

Contraseñas:

- Longitud entre 8 y 20 caracteres.

Telefono (Bolivia):

- Puede iniciar con +591 o sin prefijo.
- Debe comenzar con 6 o 7.
- Cantidad de digitos consistente con numeracion movil.
- No se permiten letras o simbolos.

Correo Electronico:

- Debe seguir el formato general de correos.
- Solo un arroba permitido.
- Dominio valido obligatorio.

Recuperacion de Contraseña

- Generacion de codigo corto aleatorio.
- Duracion limitada a 15 minutos.
- Almacen temporal para comparacion.
- Envio mediante correo configurado.

Base de Datos (Estructura y Parametros)

- Charset UTF8MB4 para textos ampliados.
- Collation que admite ordenamiento correcto para español.
- Tablas principales: usuarios, proyectos, categorias, asesorias, notificaciones y otras.
- Estados definidos en español.
- Relaciones con claves foraneas.
- Manejo de fechas con marcas de creacion y actualizacion.

Elementos Funcionales Adicionales

- Registro y autenticacion con manejo de sesiones.
- Roles y permisos definidos en la base de datos.
- Sistema de notificaciones internas.
- Control de estados para proyectos y usuarios.
- Almacenamiento de imagenes de perfil.
- Activacion y desactivacion de cuentas.
- Token de seguridad para acciones especiales.
- Prevencion de duplicados en correos y usuarios.
- Proteccion contra solicitudes no autorizadas.
- Compatibilidad con despliegue en hosting o servidores locales.
- Arquitectura preparada para escalamiento con contenedores.

Notas Generales

- Uso de archivos .env como configuracion privada.
- Comentarios del codigo en español.
- Estructura orientada a contenedores.

- Seguridad mediante hashing de contraseñas.
- Integracion opcional de doble factor.

- **Seguridad**:

Mecanismos de Autenticacion

Encriptacion de Contraseñas

- Se usa bcrypt para proteger contraseñas.
- Las contraseñas se codifican con 10 rondas de salt.
- El hash se aplica tanto en el registro como en el restablecimiento.

Validacion de Contraseñas

- Las contraseñas deben tener entre 8 y 20 caracteres.
- No se permiten contraseñas vacias o nulas.

Autenticacion de Doble Factor (2FA)

- Se usa speakeasy para generar codigos TOTP.
- Cada usuario obtiene un secreto unico al activar el 2FA.
- Se ofrece un codigo QR para registrar el 2FA en apps de autenticacion.
- Los codigos se validan con un margen de tiempo para evitar fallas por desfase.
- Si el usuario tiene 2FA activo, el sistema exige un codigo adicional al iniciar sesion.

Verificacion en el Inicio de Sesion

- Las credenciales se comparan con bcrypt.
- Se verifica si la cuenta esta activa o desactivada.

- Solo despues de validar todos los pasos se otorga acceso.

Permisos y Control de Acceso

Sistema de Roles

- Los roles definen el nivel de acceso dentro del sistema.
- Roles existentes:
    - Admin
    - Usuario
- El rol se asigna desde la base de datos y controla la experiencia del usuario.

Control de Rutas Protegidas

- Se verifica si el usuario esta autenticado.
- Se valida si el rol tiene permiso para acceder a la ruta.
- Si el usuario no cumple los requisitos, es redirigido automaticamente.

Estado de la Cuenta

- Las cuentas pueden desactivarse desde la base de datos.
- Un usuario desactivado no puede iniciar sesion aunque su contraseña sea correcta.

Validaciones de Seguridad

Validacion de Datos de Entrada

- Los correos se validan con una expresion regular.
- Los telefonos usan validacion especifica para numeracion boliviana.

- Se revisa que correos y nombres de usuario no esten repetidos.

Recuperacion de Contraseña

- Se generan codigos temporales de 6 caracteres hexadecimales.
- Los codigos expiran despues de 15 minutos.
- El cambio de contraseña solo se permite si el token es valido y vigente.

Seguridad del Servidor

CORS

- Solo se permite el acceso desde el origen configurado.
- Se limitan los metodos HTTP aceptados.
- Tambien se controlan headers autorizados.

Gestion de Sesiones

- El frontend almacena credenciales en localStorage.
- Se sincroniza el estado mediante eventos propios.
- El sistema puede borrar toda la informacion de sesion al cerrar sesion.

Proteccion de Archivos

- Aunque se admite la subida de archivos, solo se aceptan tipos especificos.
- No se permite la subida de codigo ejecutable.

Seguridad de la Base de Datos

Configuracion UTF-8

- Toda la base esta en utf8mb4 para evitar errores con caracteres especiales.

- Esto tambien protege contra ciertas vulnerabilidades relacionadas con encoding.

Restricciones e Integridad

- Se usan claves foraneas con reglas para eliminar datos dependientes.
- Esto mantiene consistencia en toda la base.

Extras Necesarios (Recomendados para un Sistema Completo)

Rate Limiting

- Limitar intentos de inicio de sesion para evitar ataques de fuerza bruta.
- Evitar spam en endpoints sensibles como recuperar contraseña.

JWT para Autenticacion Moderna

- Reemplazar localStorage simple por tokens firmados digitalmente.
- Permite expiracion automatica y mayor seguridad.
- Posibilidad de tokens de refresco para sesiones largas.

Validacion de Peso y Tamano de Archivos

- Establecer limites de peso para los archivos subidos.
- Evitar subidas gigantes o corruptas.

HTTPS en Produccion

- Uso obligatorio de HTTPS para proteger datos en transito.
- Redireccion automatica cuando el usuario accede por HTTP.

Sanitizacion de Datos

- Limpiar los datos de entrada para prevenir inyecciones.
- Evitar caracteres peligrosos en campos como nombres, descripciones, etc.

Proteccion CSRF

- Implementar tokens anti-CSRF cuando el sistema maneja sesiones persistentes.
- Evita que un tercero pueda ejecutar acciones desde otra pagina.

Logs de Seguridad

- Registrar intentos fallidos de acceso.
- Registrar cambios de configuracion o asignacion de roles.
- Registrar eventos de 2FA.

Politica de Contraseñas Opcional

- Requerir numeros, mayusculas o simbolos si se quiere mayor seguridad.
- Bloquear contraseñas demasiado comunes.

Expiracion de Sesiones

- Cerrar sesiones automaticamente despues de cierto tiempo sin uso.

Proteccion contra XSS

- Escapar contenido que se muestre en el frontend.
- Evitar que inputs con scripts puedan ejecutarse.

- **Depuración y solución de problemas:**

Errores de Validacion de Proyectos

Validacion de archivos

- Solo se permiten documentos PDF, DOCX o XLSX
- Solo se permiten imagenes PNG, JPG, JPEG o GIF

Campos obligatorios

- Nombre obligatorio
- Categoria obligatoria
- Descripcion general obligatoria

Errores del formulario frontend

- El nombre es obligatorio
- Seleccione una categoria
- La descripcion corta es obligatoria
- La descripcion general es obligatoria

Validacion de fechas

- La fecha de inicio no puede ser anterior a hoy
- La fecha de fin no puede ser menor a la fecha de inicio

Validacion de telefono

- Ingrese un numero valido de Bolivia (8 digitos, comenzando con 6 o 7)

Validacion de video

- Ingrese una URL valida de YouTube

Limites de archivos

- Maximo 10 imagenes

- Maximo 5 documentos

Errores de Autenticacion y Usuario

Login

- Usuario no encontrado
- Contrasena incorrecta
- Cuenta desactivada

Registro

- Todos los campos obligatorios deben ser completados
- Correo invalido
- La contrasena debe tener entre 8 y 20 caracteres
- Numero de telefono invalido

Duplicados

- Correo ya registrado
- Nombre de usuario ya registrado

Mensajes del frontend de login

- Tu cuenta esta desactivada o eliminada
- Usuario o contrasena incorrectos
- Esta cuenta ha sido desactivada. Contacta al administrador

Errores de Autenticacion de Dos Factores (2FA)

- Faltan datos obligatorios
- Usuario no tiene 2FA activo

- Codigo incorrecto

Errores de Recuperacion de Contrasena

- Token no generado
- Token invalido
- Token expirado
- Token no validado aun
- Las contrasenas no coinciden

Errores de Base de Datos

- Proyecto no encontrado
- Ya diste me gusta a este proyecto
- Datos incompletos

Instrucciones de Depuracion

Reintentos de conexion a la base de datos
El sistema reintenta hasta 10 veces con un retraso de 2 ` segundos entre intentos.

Logging para debugging
El servidor registra errores completos para facilitar la depuracion.

Endpoint de verificacion de base de datos
Existe un endpoint dedicado para verificar el estado de la conexion.

Sistema de Notificaciones Automatizado

Un job cron revisa diariamente los proyectos incompletos y envia notificaciones por correo.

Conflictos Potenciales con Otros Sistemas

Conflictos de puerto
MySQL usa el puerto 3307 en lugar del estandar 3306, lo que puede generar choques con otras instancias.

Configuracion CORS
Solo se permite el origen http://localhost:5173, lo que puede causar problemas si el frontend corre en otro puerto o dominio.

Codificacion de caracteres
El sistema usa UTF8MB4 para soportar caracteres especiales; una mala configuracion puede generar errores.

Restricciones de tipos de archivo
El sistema solo acepta ciertos formatos, lo que puede generar fallos si el usuario intenta subir archivos no soportados.

Dependencias entre servicios en Docker
El backend depende de que MySQL este saludable antes de iniciar, y el frontend depende del backend. Si falla el healthcheck de MySQL, toda la aplicacion puede fallar.

- **Glosario de términos:**

Termino	Definicion
API RESTful	Interfaz que sigue los principios REST y permite la comunicacion entre frontend y backend mediante peticiones HTTP estandar como GET, POST, PUT y DELETE.
Backend	Capa del servidor que maneja logica de negocio, procesos, autenticacion y comunicacion con la base de datos. No es visible para el usuario final.
bcrypt	Biblioteca de JavaScript usada para encriptar contrasenas mediante hashing con salt, protegiendo contra ataques de fuerza bruta.
CORS	Mecanismo del navegador que controla si un dominio puede realizar peticiones a otro dominio, permitiendo o restringiendo el acceso a recursos.
Cron Job	Tarea programada que se ejecuta automaticamente en intervalos definidos, comun en sistemas Unix/Linux.

CRUD	Conjunto de operaciones basicas sobre datos: Crear, Leer, Actualizar y Eliminar.
Docker	Plataforma que permite ejecutar aplicaciones en contenedores aislados con todas sus dependencias.
Docker Compose	Herramienta que define y ejecuta aplicaciones Docker con multiples servicios mediante un archivo YAML.
Express	Framework para Node.js que facilita la creacion de servidores web y APIs mediante rutas y middleware.
Frontend	Parte visual de la aplicacion con la que el usuario interactua directamente en el navegador.
Git	Sistema de control de versiones distribuido para rastrear cambios, colaborar y mantener historiales de codigo.
Healthcheck	Mecanismo que verifica periodicamente si un servicio funciona correctamente y permite reiniciarlo si falla.
Multer	Middleware de Node.js para gestionar cargas de archivos mediante multipart/form-data.
MVC	Patron de diseño que separa la aplicacion en Modelo (datos), Vista (interfaz) y Controlador (logica de peticiones).
MySQL	Sistema de gestion de bases de datos relacional que utiliza SQL para almacenar y consultar datos estructurados.
Node-cron	Biblioteca para programar tareas usando sintaxis de cron dentro de aplicaciones Node.js.
Nodemailer	Herramienta de Node.js para enviar correos electronicos mediante diferentes servicios de transporte.
npm	Gestor de paquetes de Node.js que permite instalar y manejar dependencias de proyectos JavaScript.
React	Biblioteca de JavaScript para construir interfaces interactivas con componentes reutilizables.
Speakeasy	Biblioteca para generar y verificar codigos TOTP usados en autenticacion de dos factores (2FA).

2FA	Metodo de seguridad que requiere dos formas de verificacion: contrasena y un codigo temporal adicional.
UTF8MB4	Codificacion de MySQL que permite almacenar todos los caracteres Unicode, incluidos emojis.
Vite	Herramienta moderna de desarrollo y build para proyectos frontend, con servidor rapido y HMR.

- **Referencias y recursos adicionales:**

Recurso	**Descripcion**	**Enlace**
Documentacion oficial de React	Guia completa para aprender y consultar React.	https://es.react.dev/
Tutorial interactivo de React	Practica guiada paso a paso.	https://es.react.dev/learn
Comunidad React en Stack Overflow	Foro con dudas y soluciones tecnicas.	https://stackoverflow.com/questions/tagged/reactjs
Documentacion Express.js	Guia oficial del framework.	https://expressjs.com/es/
Guia de inicio Express	Introduccion basica a Express.	https://expressjs.com/es/starter/installing.html
Comunidad Node.js	Foro oficial para preguntas tecnicas.	https://github.com/nodejs/help
Documentacion Node.js	Guia oficial del entorno backend.	https://nodejs.org/es/docs/
Documentacion MySQL	Manual y guias oficiales.	https://dev.mysql.com/doc/

Tutorial MySQL	Guia de aprendizaje oficial.	https://dev.mysql.com/doc/mysql-tutorial-excerpt/8.0/en/
Foros MySQL Community	Espacio oficial de discusion tecnica.	https://forums.mysql.com/
Documentacion Docker	Guia de referencia oficial.	https://docs.docker.com/
Documentacion Docker Compose	Configuracion de aplicaciones multicontenedor.	https://docs.docker.com/compose/
NPM bcrypt	Pagina oficial del paquete.	https://www.npmjs.com/package/bcrypt
NPM speakeasy	Implementacion de 2FA.	https://www.npmjs.com/package/speakeasy
GitHub speakeasy	Repositorio oficial.	https://github.com/speakeasyjs/speakeasy
NPM multer	Manejo de archivos en Express.	https://www.npmjs.com/package/multer
GitHub multer	Repositorio oficial.	https://github.com/expressjs/multer
Documentacion Nodemailer	Guia para enviar correos.	https://nodemailer.com/
NPM nodemailer	Paquete en npm.	https://www.npmjs.com/package/nodemailer
NPM node-cron	Programacion de tareas cron en Node.	https://www.npmjs.com/package/node-cron
GitHub node-cron	Repositorio oficial.	https://github.com/node-cron/node-cron

MDN Web Docs	Documentacion de JavaScript y web en español.	https://developer.mozilla.org/es/
freeCodeCamp Español	Cursos gratuitos en español.	https://www.freecodecamp.org/espanol/
Platzi	Cursos en español sobre desarrollo.	https://platzi.com/
Stack Overflow en Español	Foro de programacion en español.	https://es.stackoverflow.com/

Herramienta	Descripcion	Enlace
React	Biblioteca para construir interfaces de usuario.	https://es.react.dev/
Vite	Herramienta de desarrollo y build para frontend.	https://vitejs.dev/
Express.js	Framework minimalista para crear APIs y servidores en Node.js.	https://expressjs.com/es/
Node.js	Entorno de ejecucion para JavaScript en backend.	https://nodejs.org/es/docs/

- **Herramientas de Implementación:**

  Lenguajes de programacion

  - **JavaScript (ES6+ con ESM)**
  - **JSX (para componentes React)**
  - **SQL (para consultas y esquemas MySQL)**

Frameworks

- **Frontend**
    - **React 19**
    - **React DOM**
    - **React Router DOM**
    - **React Icons**
    - **Vite**
    - **@vitejs/plugin-react**
- **Backend**
    - **Express**
    - **Node.js**

Base de Datos

- **MySQL 8.0**
- **mysql2 (driver con soporte de promesas)**

APIs y Librerias de Terceros

- **Nodemailer (envio de correos)**
- **Gmail SMTP (servicio de correo)**
- **Axios (peticiones HTTP)**
- **Font Awesome (iconos via CDN)**
- **WhatsApp (enlaces de contacto)**
- **YouTube (videos incrustados)**
- **Redes sociales (Facebook, Instagram, X, TikTok)**

Middleware y Seguridad

- **CORS**
- **bcrypt**
- **speakeasy (2FA)**
- **qrcode (generacion de QR)**
- **dotenv (variables de entorno)**
- **ESLint (calidad de codigo)**
- **eslint-plugin-react-hooks**
- **eslint-plugin-react-refresh**

Procesamiento de Archivos

- **multer (subida de archivos)**

Tareas Programadas

- **node-cron (jobs automaticos)**

Herramientas de Desarrollo

- **Nodemon**
- **Vite (desarrollo y build)**
- **TypeScript Types:**
    - **@types/react**
    - **@types/react-dom**

Contenedorizacion

- **Docker**

- **Docker Compose (orquestacion de MySQL, backend y frontend)**

Bibliografía

Documentación Técnica

Node.js

- Node.js Foundation. *Node.js Documentation*. nodejs.org/docs/. Accedido el 2 Dic. 2025.
- Node.js Foundation. *Node.js Learn Portal*. nodejs.org/en/learn/getting-started/introduction-to-nodejs. Accedido el 2 Dic. 2025.
- Node.js Foundation. *Node.js API Reference*. nodejs.org/api/. Accedido el 2 Dic. 2025.

React

- Meta Open Source. *React Documentation*. react.dev/. Accedido el 2 Dic. 2025.
- Meta Open Source. *React Learn Guide*. react.dev/learn. Accedido el 2 Dic. 2025.
- Meta Open Source. *React API Reference*. react.dev/reference/react. Accedido el 2 Dic. 2025.

Vite

- Vite Contributors. *Vite Documentation*. vitejs.dev/. Accedido el 2 Dic. 2025.
- Vite Contributors. *Vite Config Guide*. vitejs.dev/config/. Accedido el 2 Dic. 2025.
- Vite Contributors. *Vite Plugins Directory*. vitejs.dev/plugins/. Accedido el 2 Dic. 2025.

React Router DOM

- Remix Software. *React Router Documentation*. reactrouter.com/. Accedido el 2 Dic. 2025.

- Remix Software. *React Router Tutorial*. reactrouter.com/en/main/start/tutorial. Accedido el 2 Dic. 2025.

Express.js

- OpenJS Foundation. *Express Documentation*. expressjs.com/. Accedido el 2 Dic. 2025.

- OpenJS Foundation. *Express Starter Guide*. expressjs.com/en/starter/installing.html. Accedido el 2 Dic. 2025.

- OpenJS Foundation. *Express 4.x API Reference*. expressjs.com/en/4x/api.html. Accedido el 2 Dic. 2025.

MySQL

- Oracle Corporation. *MySQL Documentation*. dev.mysql.com/doc/. Accedido el 2 Dic. 2025.

- Oracle Corporation. *MySQL SQL Reference*. dev.mysql.com/doc/refman/8.0/en/. Accedido el 2 Dic. 2025.

- Oracle Corporation. *MySQL Tutorial*. dev.mysql.com/doc/mysql-tutorial-excerpt/8.0/en/. Accedido el 2 Dic. 2025.

mysql2

- Sidorares. *mysql2 Documentation*. github.com/sidorares/node-mysql2. Accedido el 2 Dic. 2025.

- Sidorares. *mysql2 API Reference*. github.com/sidorares/node-mysql2#readme. Accedido el 2 Dic. 2025.

- npm. *mysql2 Package Page*. npmjs.com/package/mysql2. Accedido el 2 Dic. 2025.

bcrypt

- Kelektiv. *bcrypt Documentation*. github.com/kelektiv/node.bcrypt.js. Accedido el 2 Dic. 2025.

- npm. *bcrypt Package Page*. npmjs.com/package/bcrypt. Accedido el 2 Dic. 2025.

speakeasy

- SpeakeasyJS. *speakeasy Documentation*. github.com/speakeasyjs/speakeasy. Accedido el 2 Dic. 2025.

- npm. *speakeasy Package Page*. npmjs.com/package/speakeasy. Accedido el 2 Dic. 2025.

qrcode

- soldair. *qrcode Documentation*. github.com/soldair/node-qrcode. Accedido el 2 Dic. 2025.

- npm. *qrcode Package Page*. npmjs.com/package/qrcode. Accedido el 2 Dic. 2025.

Multer

- Express Community. *Multer Documentation*. github.com/expressjs/multer. Accedido el 2 Dic. 2025.

- npm. *multer Package Page*. npmjs.com/package/multer. Accedido el 2 Dic. 2025.

Nodemailer

- Nodemailer Project. *Nodemailer Documentation*. nodemailer.com/. Accedido el 2 Dic. 2025.

- Nodemailer Project. *Nodemailer About Page*. nodemailer.com/about/. Accedido el 2 Dic. 2025.

- npm. *Nodemailer Package Page*. npmjs.com/package/nodemailer. Accedido el 2 Dic. 2025.

node-cron

- Node Cron Team. *node-cron Documentation*. github.com/node-cron/node-cron. Accedido el 2 Dic. 2025.

- npm. *node-cron Package Page*. npmjs.com/package/node-cron. Accedido el 2 Dic. 2025.

CORS

- Express Community. *CORS Middleware Documentation*. github.com/expressjs/cors. Accedido el 2 Dic. 2025.

- npm. *cors Package Page*. npmjs.com/package/cors. Accedido el 2 Dic. 2025.

dotenv

- motdotla. *dotenv Documentation*. github.com/motdotla/dotenv. Accedido el 2 Dic. 2025.

- npm. *dotenv Package Page*. npmjs.com/package/dotenv. Accedido el 2 Dic. 2025.

Axios

- Axios Team. *Axios Documentation*. axios-http.com/. Accedido el 2 Dic. 2025.

- npm. *axios Package Page*. npmjs.com/package/axios. Accedido el 2 Dic. 2025.

ESLint

- OpenJS Foundation. *ESLint Documentation*. eslint.org/. Accedido el 2 Dic. 2025.

- OpenJS Foundation. *ESLint Configuration Guide*. eslint.org/docs/latest/use/configure/. Accedido el 2 Dic. 2025.

- OpenJS Foundation. *ESLint Rules*. eslint.org/docs/latest/rules/. Accedido el 2 Dic. 2025.

Nodemon

- Nodemon Team. *Nodemon Documentation*. nodemon.io/. Accedido el 2 Dic. 2025.

- Nodemon Team. *Nodemon GitHub Repository*. github.com/remy/nodemon. Accedido el 2 Dic. 2025.

- npm. *nodemon Package Page*. npmjs.com/package/nodemon. Accedido el 2 Dic. 2025.

Docker Compose

- Docker Inc. *Docker Compose Documentation*. docs.docker.com/compose/. Accedido el 2 Dic. 2025.

- Docker Inc. *Compose File Reference*. docs.docker.com/compose/compose-file/. Accedido el 2 Dic. 2025.

- Docker Inc. *Docker Compose Getting Started*. docs.docker.com/compose/gettingstarted/. Accedido el 2 Dic. 2025.