



# UNIVERSIDAD NACIONAL AUTONOMA DE HONDURAS



**FACULTAD DE INGENIERIA**

**DEPARTAMENTO DE INGENIERIA EN SISTEMAS**

**PROGRAMACION ORIENTADA A OBJETOS-(IS410)**

**TEMA:**

**PRESENTACION DE PROYECTO**

**DOCENTE:**

**ING. ERICK VLADIMIR REYES MARIN**

**TRABAJO GRUPAL**

**NOMBRE DE LOS INTEGRANTES:**

JOHAN CALEB PAZ IZAGUIRRE.....	20231003056
JESUS EDUARDO RAMIREZ AGUILAR.....	20231002862
JUSTIN DAVID GONZALEZ PALMA.....	20231031599
ANGEL JOSUÉ PACHECO ERAZO.....	20151003925

**III PAC 2025**

**CIUDAD UNIVERSITARIA(C.U.)**

**27 DE NOVIEMBRE 2025**

# INDICE

INTRODUCCION.....	1
-------------------	---

## MARCO TEORICO:

### 1.1 HERRAMIENTAS USADAS:

#### ➤ FRONTEND:

○ LENGUAJES.....	2
○ LIBRERÍAS.....	2

#### ➤ BACKEND:

○ APLICACIONES.....	3
○ LENGUAJES.....	3
○ FRAMEWORKS.....	3

#### ➤ BASE DE DATOS:

○ BASE DE DATOS.....	4
----------------------	---

### 1.2 PANTALLAS DISEÑADAS

❖ PAGINA DE INICIO/LADING.....	5
❖ INICIO .....	5
❖ REGISTRO.....	6
❖ DESARROLLO.....	6
❖ DATOS DE USUARIO .....	7
❖ ADMINISTRACION DE PERFILES.....	7
❖ CARPETA DE PROYECTOS.....	8
❖ PLANES DISPONIBLES.....	8
❖ PREFERENCIAS DE USUARIO.....	9

1.3 DIAGRAMA DE BASE DE DATOS.....	10
------------------------------------	----

1.4 FUNCIONALIDADES/TECNOLOGÍAS INVESTIGADAS. ....	12
--	----

1.5 GLOSARIO .....	14
--------------------	----

# INTRODUCCION

En el siguiente informe estaremos recapitulando y viendo el proceso que hicimos para crear el proyecto asignado. La plataforma permite a los usuarios registrarse, iniciar sesión, administrar su perfil, organizar sus archivos en carpetas lógicas y crear proyectos de desarrollo web que integran HTML, CSS y JavaScript en una misma interfaz.

El uso de un contenedor dinámico permite observar cada proyecto en tiempo real, lo que hace posible evaluar los resultados inmediatamente sin depender de instrumentos ajenos. Con el propósito de replicar un ambiente auténtico de servicios en la nube, la app también incorpora un sistema de planes, que abarcan desde opciones sin costo hasta paquetes con una mayor capacidad para almacenar.

Además de los proyectos completos, el sistema también permite la creación de snippets o fragmentos de código en diversos lenguajes de programación. Estos snippets pueden organizarse dentro de carpetas lógicas y compartirse con otros usuarios, promoviendo así la colaboración y el intercambio de recursos entre desarrolladores.

Desde el punto de vista técnico, la plataforma se construye utilizando tecnologías modernas como Node.js con Express para el backend, MongoDB como un gestor de base de datos, y el frontend basado en Bootstrap o Tailwind además de CSS para asegurar una correcta visualización en todo tipo de dispositivos. Todas las interacciones entre el cliente y el servidor se realizan mediante peticiones FETCH en formato JSON, garantizando fluidez y dinamismo puro en la experiencia de usuario que al final es lo que buscamos.

El objetivo principal de este proyecto es demostrar la capacidad de integrar múltiples tecnologías para construir una aplicación web funcional, modular y escalable, orientada a la colaboración y a la gestión eficiente de proyectos de desarrollo web.

## 1.1 Herramientas Utilizadas:

### FRONTEND:

#### 1.Lenguajes:

- **HTML:** Creamos la estructura principal de la página, así como todos sus componentes
- **CSS:** Estilos y vida a nuestra página, además de ajustar todo lo que vemos
- **JAVASCRIPT(JS):** Interactividad dentro de la página, además de hacer la conexión entre backend y frontend

#### 2.Librerías y Frameworks:

- **Bootstrap:** Usado con clases CSS importadas
- **FontAwesome:** Iconos en forma de texto
- **CodeMirror:** Editor web de código con syntax highlighting
- **FontsGoogle:** Utilizado para importar letras y familias de fuentes.

```
<script src="https://kit.fontawesome.com/e4c6f78817.js" crossorigin="anonymous"></script>

<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.7/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-LN+7fdVzj6u52u3"
<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.7/dist/js/bootstrap.bundle.min.js" integrity="sha384-ndDqU0Gzau9qJ11fw4pNL1hNTk

<script src="https://cdnjs.cloudflare.com/ajax/libs/html2canvas/1.4.1/html2canvas.min.js"></script>
<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/codemirror/5.65.7/codemirror.min.css" />
<script src="https://cdnjs.cloudflare.com/ajax/libs/codemirror/5.65.7/codemirror.min.js"></script>
<script src="https://cdnjs.cloudflare.com/ajax/libs/codemirror/5.65.7/mode/xml/xml.min.js"></script>
<script src="https://cdnjs.cloudflare.com/ajax/libs/codemirror/5.65.7/mode/javascript/javascript.min.js"></script>
<script src="https://cdnjs.cloudflare.com/ajax/libs/codemirror/5.65.7/mode/css/css.min.js"></script>
<script src="https://cdnjs.cloudflare.com/ajax/libs/codemirror/5.65.7/mode/htmlmixed/htmlmixed.min.js"></script>

<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/codemirror/5.65.7/addon/hint/show-hint.min.css">

<script src="https://cdnjs.cloudflare.com/ajax/libs/codemirror/5.65.7/addon/hint/show-hint.min.js"></script>
<script src="https://cdnjs.cloudflare.com/ajax/libs/codemirror/5.65.7/addon/hint/html-hint.min.js"></script>
<script src="https://cdnjs.cloudflare.com/ajax/libs/codemirror/5.65.7/addon/hint/css-hint.min.js"></script>
<script src="https://cdnjs.cloudflare.com/ajax/libs/codemirror/5.65.7/addon/hint/javascript-hint.min.js"></script>
<script src="https://cdnjs.cloudflare.com/ajax/libs/codemirror/5.65.7/addon/hint/anyword-hint.min.js"></script>
<script src="https://cdnjs.cloudflare.com/ajax/libs/codemirror/5.65.7/addon/hint/xml-hint.min.js"></script>
```

Dependencias, herramientas y librerías llamándose para su correcto uso

## BACKEND:

### 1.Aplicaciones:

- **POSTMAN:** Usado para verificar el correcto trabajo de los endpoint que fueron creados
- **MongoDB:** Utilizada en desktop para ir visualizando el funcionamiento de la base de datos conectada en el backend

### 2.Lenguajes y Entornos:

- **Node.js:** Entorno usado para poder ejecutar el código JavaScript del lado del servidor.
- **TypeScript:** Superset JavaScript para darle funcionalidad adicional y escalable.

### 3.Frameworks:

- **Express: Framework** para crear rutas, manejar peticiones HTTP y middleware.
- **Cors:** Middleware para habilitar peticiones cross-origin (AJAX desde otros dominios, Fetch en nuestro caso).
- **Bcryptjs:** Librería para encriptar contraseñas de usuarios

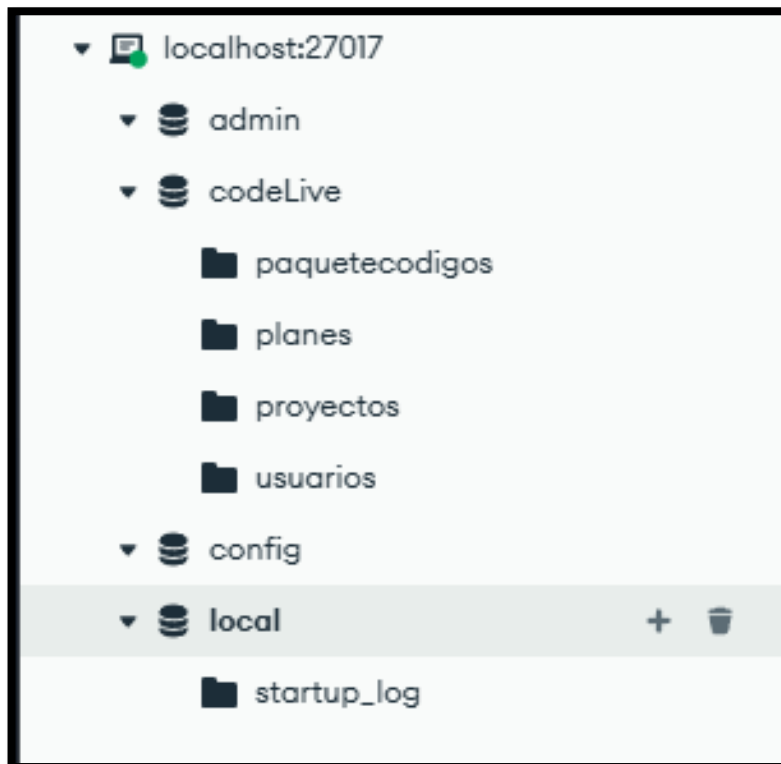
```
19     "devDependencies": {
20       "@types/express": "^5.0.5",
21       "@types/node": "^24.10.1",
22       "dotenv": "^17.2.3",
23       "express": "^5.1.0",
24       "install": "^0.13.0",
25       "nodemon": "^3.1.11",
26       "npm": "^11.6.4",
27       "ts-node": "^10.9.2",
28       "typescript": "^5.9.3"
29     }
30   }
31 }
```

Dependencias instaladas en el backend, para evitar errores en la compilación y depuración

## BASE DE DATOS:

### 1.Base de Datos:

- **MONGODB:** Es una base de datos que no usa tablas como MySQL, si no que utiliza documentos que están en formato JSON



Carpetas creadas de acuerdo con las páginas que se fueron usando, para una mejor organización

localhost:27017				<a href="#">Open MongoDB shell</a>	<a href="#">+ Create database</a>	<a href="#">Refresh</a>
Database name	Storage size	Collections	Indexes			
admin	20.48 kB	0	1			
codeLive	65.54 kB	4	9			
config	12.29 kB	0	2			
local	20.48 kB	1	1			

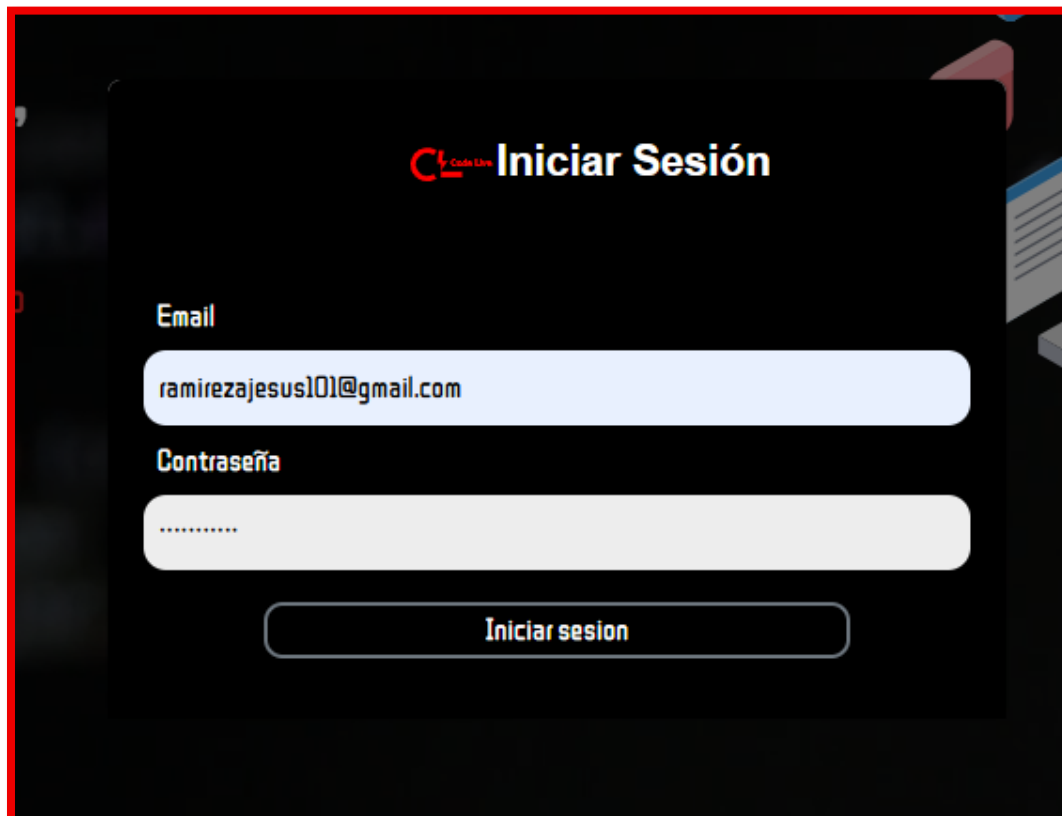
Localhost dándonos respuesta correcta, con la colección de bases dentro del servidor

## 1.2 Pantallas Diseñadas:

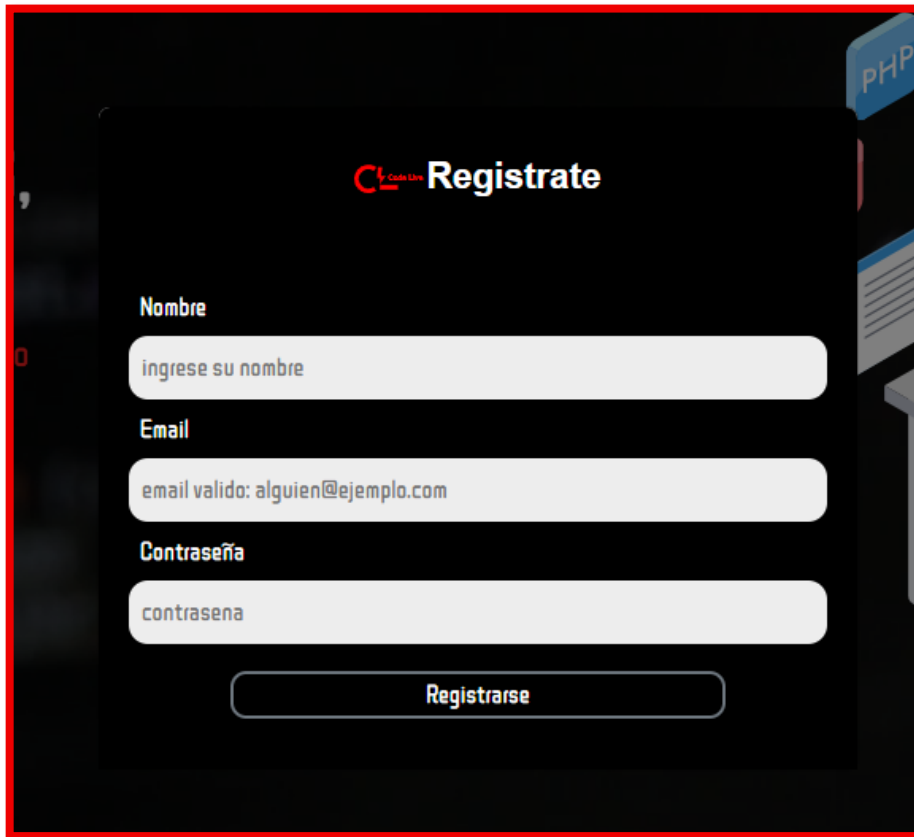
### 1. Página de Inicio/Landing Page



### 2. Login/Inicio de Sesión

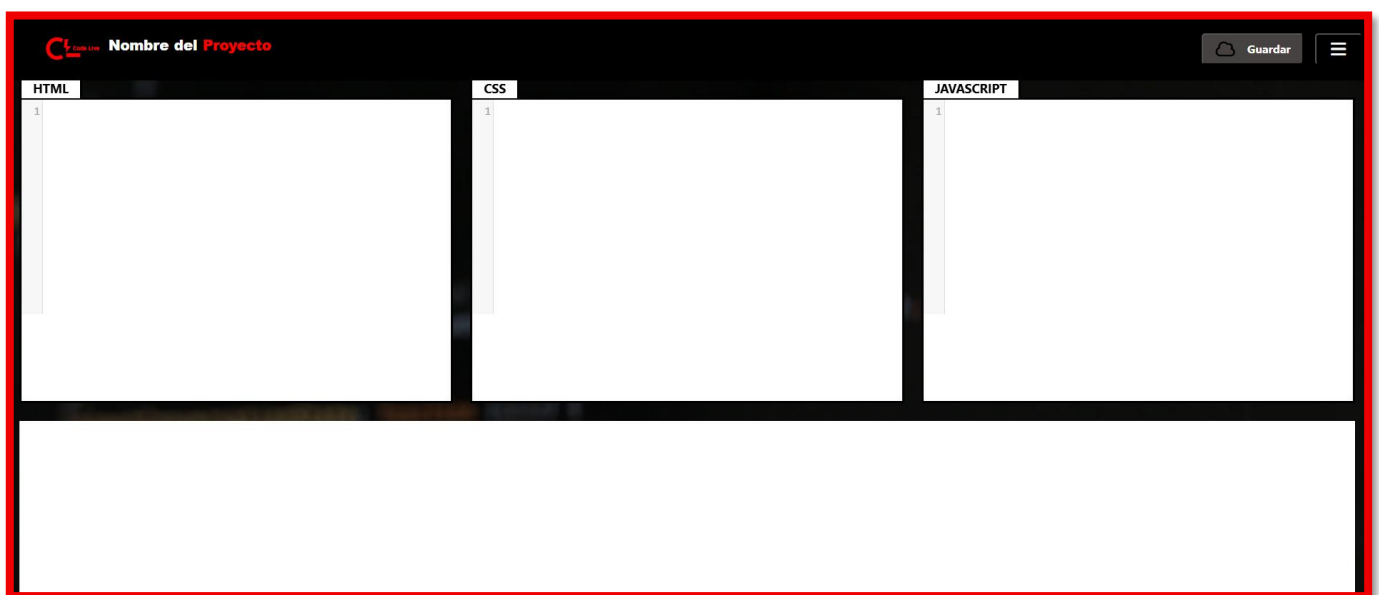


### 3. Registro



The screenshot shows a registration form titled "Registrate" with the CodeLive logo. It features three input fields: "Nombre" (Name) with placeholder text "ingrese su nombre", "Email" with placeholder text "email valido: alguien@ejemplo.com", and "Contraseña" (Password) with placeholder text "contrasena". A "Registrarse" button is located at the bottom of the form. The form is set against a dark background with a red border.

### 4.Desarrollo de Proyecto





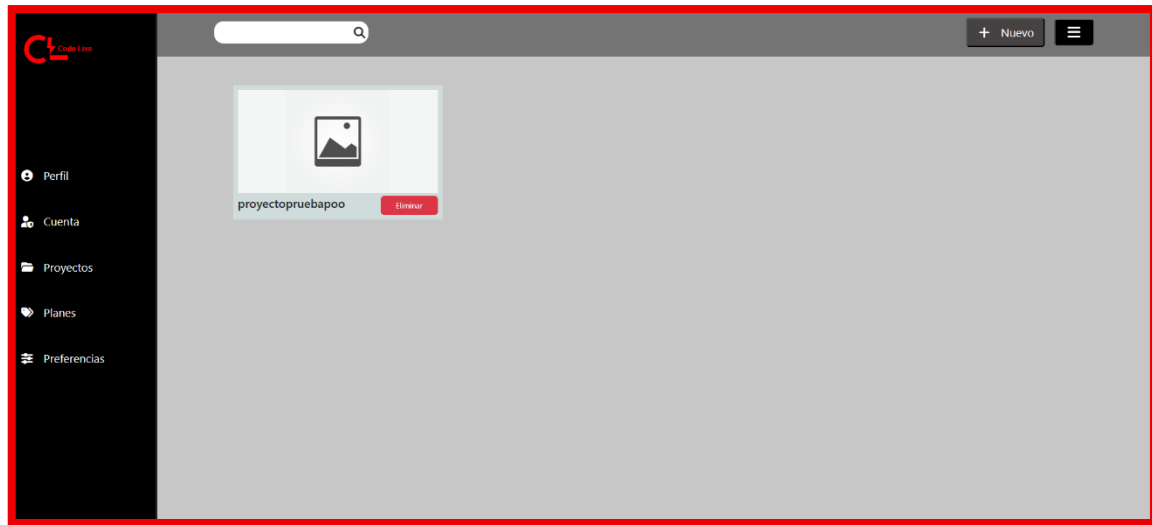
## 5.Datos de Usuario

The screenshot shows the 'Datos de Usuario' (User Data) form. On the left is a dark sidebar with the 'Code Line' logo and a menu with options: Perfil, Cuenta, Proyectos, Planes, and Preferencias. The top header contains a search bar and a '+ Nuevo' button. The main content area is divided into two columns. The left column has labels 'imagen perfil' and 'acerca de ti'. The right column contains a profile picture placeholder (a square with a person icon) and two input fields labeled 'Nombre:' and 'Correo:'.

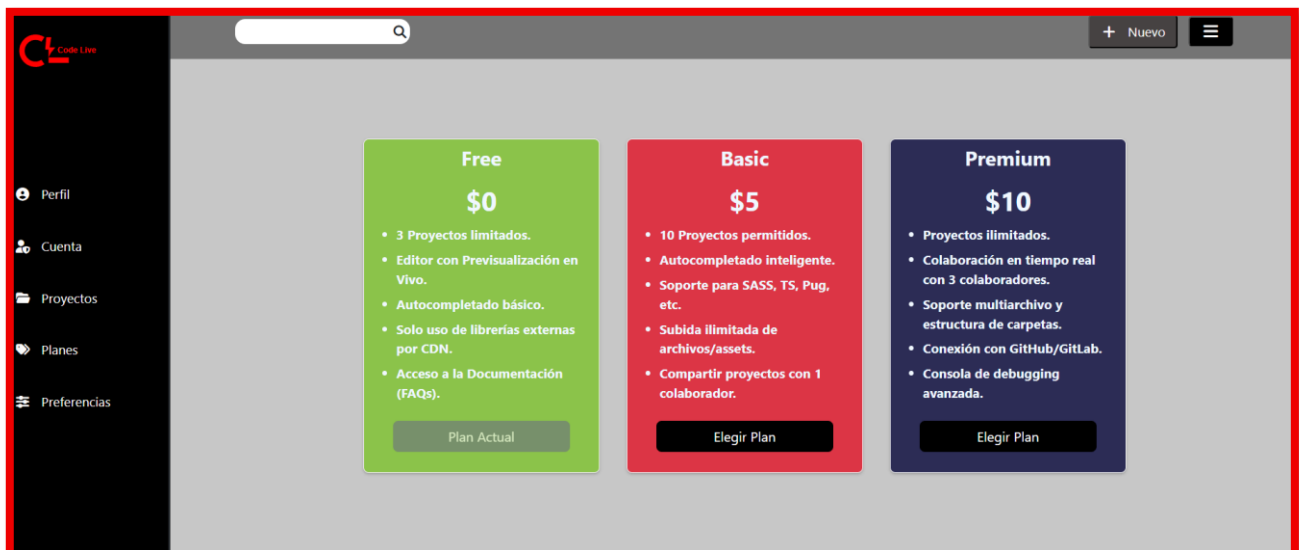
## 6.Administracion de Perfiles

The screenshot shows the 'Administracion de Perfiles' (Profile Management) form. It features the same sidebar and header as the previous form. The main content area has three sections for profile management. The first section, 'Nombre de usuario', includes a 'Nuevo nombre' input field and a 'Cambiar' button. The second section, 'Contraseña', includes 'Contraseña actual' and 'Contraseña nueva' input fields, followed by a 'Cambiar' button. The third section, 'Correo', includes a 'Cambiar Correo' input field and a 'Cambiar' button.

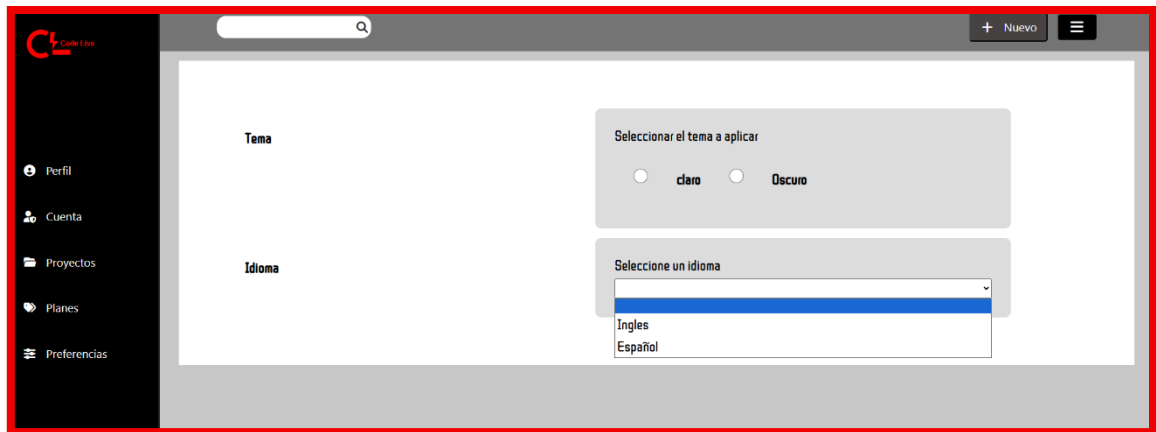
## 6. Visual de Proyectos



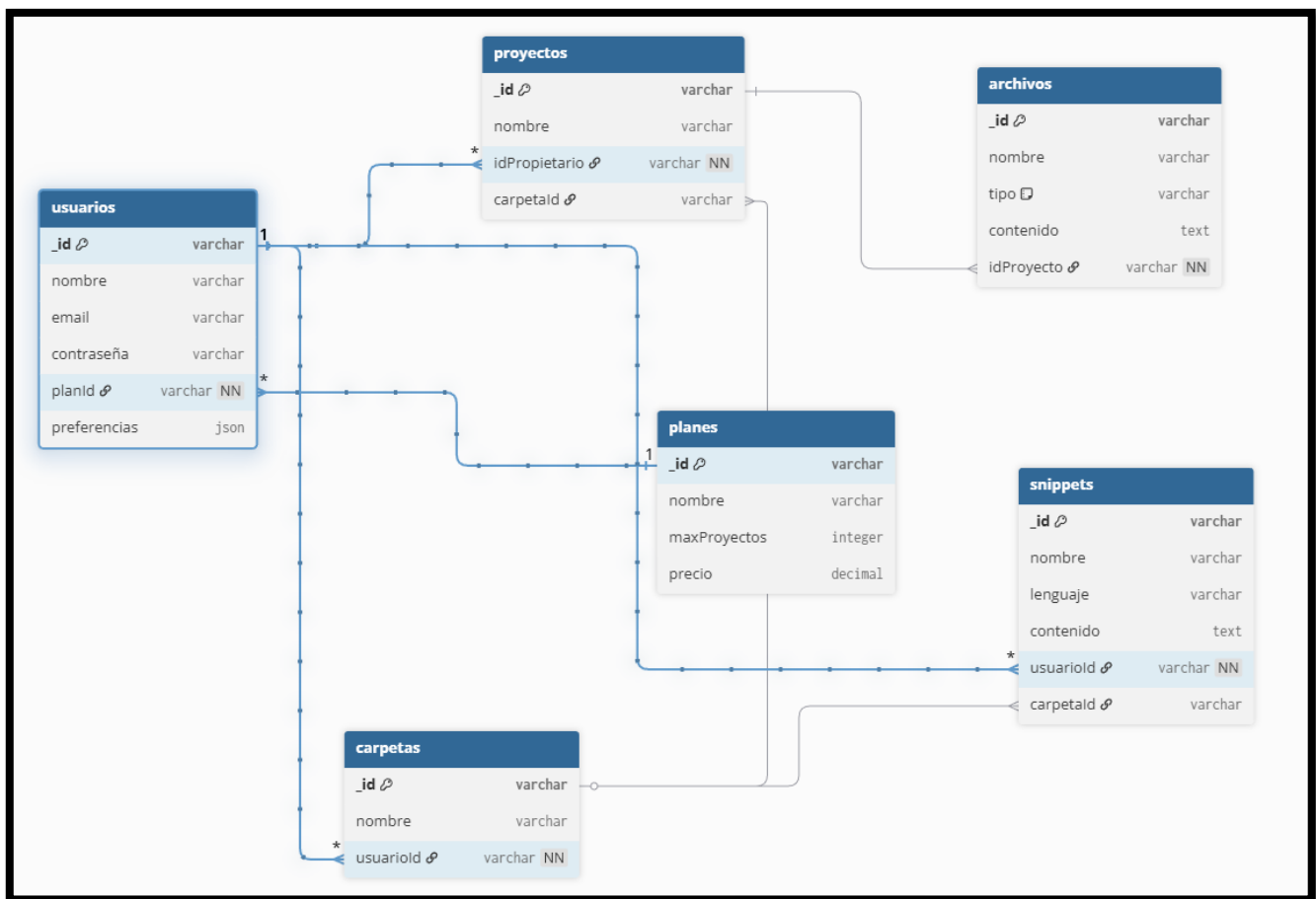
## 7. Planes



## 8.Preferencias del Usuario/Ajustes



### 1.3 DIAGRAMA BASE DE DATOS



Estamos utilizando MongoDB como gestor de base de datos. Esta base almacena la información de los usuarios, los planes, proyectos y carpetas. Con esto prácticamente definimos un modelo de datos claro, para administrar de manera eficiente la gestión y almacenamiento de datos.

Este modelo está orientado a documentos, para poder crear colecciones que se manejen con facilidad en las relaciones que hay entre los proyectos y el usuario

- ✓ **Planes:** nombre, precio, límite de proyectos.
- ✓ **Usuarios:** nombre, email, contraseña, plan asignado y preferencias.

- ✓ **Proyectos:** nombre, usuario propietario, carpeta asociada.
- ✓ **Archivos:** nombre, tipo (HTML, CSS, JS, snippet), contenido, proyecto asociado.
- ✓ **Carpetas:** nombre, usuario propietario, organiza proyectos o snippets.
- ✓ **Snippets:** nombre, lenguaje, contenido, usuario y carpeta asociada

#### **Relaciones entre entidades:**

- ◆ Un usuario tiene un plan asignado(**planID**)
- ◆ Un proyecto pertenece a un usuario (**IDpropietario**)
- ◆ Cada archivo pertenece a un proyecto (**IDproyecto**)
- ◆ Cada carpeta le pertenece a un usuario y puede contener snippets o proyectos
- ◆ Cada snippet pertenece a un usuario

Las relaciones antes mencionadas se crearon con la finalidad de crear una base de datos coherentes y que no existiera ningún tipo de colisión de datos, para que se mantenga bien organizada y que evite confusiones.

Con esto logramos que se administren bien los recursos, y que debido a su orden permita que en una futura actualización no se comprometa nada del rendimiento

## **1.4 FUNCIONALIDADES Y TECNOLOGÍAS INVESTIGADAS.**

### **1. Gestión de usuarios**

La gestión de usuarios permite que los usuarios que están utilizando el sistema sean registrados correctamente y que al iniciar sesión sean autenticados de forma correcta. Incluye funciones como crear cuentas, iniciar sesión, codificar contraseñas y asignar roles o privilegios. Esta función garantiza que cada usuario solo tenga acceso a las opciones que le son propias, lo que asegura la organización y la seguridad dentro de la aplicación.

### **2. Sistema de planes y suscripciones**

El sistema de planes permite definir diferentes niveles de uso dentro de la plataforma de acuerdo con la inversión de cada usuario se le presentaran varios planes, cada uno con sus respectivos límites y características. Por ejemplo, un plan premium brinda beneficios adicionales, mientras que uno gratuito puede presentar limitaciones en cuanto a proyectos. Esto facilita la monetización del sistema e, igualmente, ordena los recursos a los que tiene acceso cada usuario.

### **3. Gestión de proyectos**

El manejo de proyectos permite que el usuario cree, modifique y gestione sus diversas tareas en la plataforma. Cada proyecto cuenta con su propia información y está vinculado directamente con un usuario que lo posee. Esta función es esencial en nuestra Rest API, para hacer un seguimiento y control del contenido o código que el usuario produce en cada proyecto.

#### **4. NodeJS**

Node.js es el ambiente de ejecución que habilita la ejecución de JavaScript, en el lado del servidor. Su arquitectura orientada a eventos lo hace rápido, ligero y eficaz; contiene módulos que facilitaron el trabajo de desarrollo y prueba del código. Es perfecto para desarrollar aplicaciones y APIs en tiempo real, ya que brinda un ecosistema extenso de herramientas y librerías.

#### **5. Express**

Express es un marco de trabajo de Node que permite la generación de controladores, middleware y rutas para aplicaciones web. Facilita la organización del backend de un modo modular, claro y escalable. Su simplicidad y flexibilidad lo convierten en una de las tecnologías más utilizadas para desarrollar APIs REST.

#### **6. MongoDB**

MongoDB es una base de datos que se enfoca en los documentos y almacena la información en formato JSON extendido. Es adaptable a estructuras cambiantes, se ajusta bien y permite una sencilla escalabilidad. Su modelo, que se basa en colecciones y documentos, facilita la gestión de datos sin un esquema rígido.

#### **7. Mongoose**

Mongoose es una librería para Node que posibilita el uso de MongoDB empleando modelos y esquemas. Permite manejar documentos, validar datos y establecer relaciones con mayor control.

## 1.6 GLOSARIO

**AJAX:** Por sus siglas (Asynchronous JavaScript and XML) es una forma asíncrona de ejecutar el programa, permite que una pagina web se comunice con el servidor sin recargar la pagina completa, para hacer su actualización y captura de datos de forma más rápida y dinámica

**API REST:** Es el servicio que se encarga de comunicar los sistemas utilizando los métodos HTTP estándares, entre ellos GET, POST, DELETE y PUT, Es la estructura que nos permite actualizar y crear una escalabilidad organizada en nuestro proyecto.

**FETCH** Es un API de navegador moderna, que permite el ingreso de solicitudes HTTP basada en promesas, este remplaza tecnologías como XMLHttpRequest para estandarizar y facilitar la comunicación entre el servidor y backend con el frontend.

**JSON:** por sus siglas (JavaScript Object Notation) es un formato ligero de datos, que busca un intercambio de manera eficaz entre servidor y cliente. Al ser ligero su formato es fácil de leer y escribir, por lo que puede ser interpretado de manera sencilla.

**MIDDLEWARE:** Es una función que ejecuta una solicitud del usuario y la respuesta del servidor, en palabras sencillas sirve como “Mediador”. Es usado para validar, modificar y procesar los datos para que lleguen al controlador final.

**SUPERSET:** Es el conjunto que incluye todos los elementos de otros conjuntos, así como elementos que se adicionan, se usa para describir los lenguajes y sus capacidades, como ejemplo tenemos el TypeScript que es el superset de JavaScript.