

PROYECTO FINAL



MADECOFFE

14 DE NOVIEMBRE DEL 2023

Contenido

Introducción.....	4
Lenguajes y tecnologías.....	5
HTML.....	5
CSS.....	5
JavaScript.....	5
Vue.....	6
PrimeVue.....	7
Vite.....	8
JSON.....	9
Bibliotecas y frameworks de javascript.....	10
Axios.....	10
NPM.....	11
Node.js.....	12
Nodemon.....	13
Libreria CORS.....	13
Base de datos y SQL.....	15
SQL.....	15
PostgresSQL.....	16
Procedimientos almacenados.....	17
Vistas en SQL.....	18
API y comunicación.....	19
API REST.....	19
Sockets.....	20
Socket.IO.....	21
Seguridad y hashing.....	22
SHA – 256.....	22
Plataformas y despliegue.....	23
AWS.....	23
Vercel.....	24
Github.....	25
Herramientas de desarrollo.....	26
Visual Studio.....	26
Postman.....	26

Conceptos avanzados	27
Funciones asíncronas en javascript.....	27
Otros.....	27
Procesamiento del lenguaje natural (NLP)	27
Librería0.xlsx	27
Librería moment.....	28
Producto final.....	29
Anexos	32
Conclusión	51
Bibliografías.....	52

Introducción.

La necesidad de este proyecto abarca cumplir con la problemática dada en una empresa de café, debido que tiene ciertas dificultades a la hora de presentar, obtener y recolectar información durante el proceso del café, además de no tener el control de los datos mismos que desea conocer en tiempo real. Debido a esto, se proporcionó un contexto del problema que se planteó en el proyecto además de ciertas consultas indispensables para nuestro usuario o cliente.

Debido a la problemática, se encontró que se podían aplicar diferentes tecnologías de entorno para el desarrollo del software, donde se tomó de manera fundamental para la aplicación del mismo en el proyecto, siempre y cuando mejorando la funcionabilidad, usabilidad del mismo.

Algunas tecnologías que se tomaran en cuenta para el desarrollo de dicha aplicación son Vue.js, PostgreSQL, socket con la librería socket.io para implementar en JavaScript, entre otras tecnologías para realizar tal aplicación de manera web, dando mayor facilidad de acceso al mismo con las tecnologías que aplicaremos a continuación de nuestro proyecto.

Lenguajes y tecnologías

HTML

HTML es el lenguaje de marcado estándar utilizado para crear y diseñar páginas web. Es un acrónimo de "HyperText Markup Language" (Lenguaje de Marcado de Hipertexto). HTML proporciona una estructura básica para los documentos web al utilizar una serie de elementos o etiquetas, cada una con un propósito específico.

- *Estructura*: HTML organiza el contenido de una página web mediante elementos como encabezados, párrafos, listas, imágenes, enlaces, formularios, entre otros. Estos elementos son definidos mediante etiquetas, que rodean el contenido y le dan significado. (Anexo - 1)

CSS

CSS es un lenguaje de estilo que define la presentación y apariencia de un documento HTML. El término "Cascading" se refiere a la forma en que las reglas de estilo se aplican en cascada, permitiendo controlar el diseño y la apariencia de múltiples páginas web desde una sola hoja de estilo.

- *Estilo*: CSS se utiliza para definir estilos como colores, márgenes, fuentes, tamaños de texto, diseño de página, etc. Permite separar la estructura del contenido (definida por HTML) de su presentación visual. (Anexo - 2)
- *Selectores*: Permite seleccionar elementos HTML específicos para aplicar estilos mediante reglas de estilo.
- *Propiedades*: Define propiedades como color, tamaño de fuente, márgenes, etc., para personalizar la apariencia de los elementos seleccionados.
- *Clasificación*: Los estilos pueden ser aplicados directamente en el HTML, en un archivo externo o en línea, y se aplican de manera jerárquica y cascada.

JavaScript

JavaScript es un lenguaje de programación de alto nivel que se utiliza para hacer que las páginas web sean interactivas y dinámicas. Aunque inicialmente se diseñó para ejecutarse en el navegador, hoy en día también se utiliza en entornos de servidor y en el

desarrollo de aplicaciones móviles. JavaScript es el tercer componente de la tecnología web estándar, junto con HTML y CSS. (Anexo - 3)

- **Interactividad:** JavaScript permite manipular el contenido de la página, responder a eventos del usuario, realizar solicitudes a servidores (AJAX), y modificar dinámicamente el contenido y el estilo de una página.
- **Orientado a objetos:** sigue un paradigma orientado a objetos. Los objetos en JS son colecciones de propiedades y métodos, en donde se centra en la manipulación de esos objetos
- **Event-Driven:** Este es eficaz en el manejo de eventos lo que permite crear páginas web interactivas, donde se puede responder a las acciones del usuario, como clics y desplazamientos.

Vue

Vue.js es un marco de JavaScript progresivo utilizado para construir interfaces de usuario interactivas. Fue creado por Evan You y se ha convertido en una herramienta popular en el desarrollo web, especialmente en el ámbito de las aplicaciones de una sola página (SPA, por sus siglas en inglés). A continuación, se detallan las características y conceptos clave de Vue.js (Anexo - 4)

- **Reactividad:** Uno de los pilares fundamentales de Vue.js es su sistema de reactividad. Permite que los cambios en los datos se reflejen automáticamente en la interfaz de usuario, eliminando la necesidad de manipulación manual del DOM.
- **Componentes:** Vue.js se basa en un enfoque de desarrollo de componentes. Los componentes son bloques de construcción reutilizables que encapsulan la lógica y la interfaz de usuario de una parte específica de la aplicación.
- **Directivas:** Vue.js utiliza directivas en el marcado HTML para proporcionar funcionalidades especiales a los elementos del DOM. Algunas directivas comunes incluyen **v-bind** para la vinculación de datos y **v-if** para controlar la visibilidad de elementos.
- **Plantillas:** Vue.js utiliza plantillas declarativas con sintaxis similar a la de HTML para definir la interfaz de usuario. Estas plantillas se vinculan a los datos y se actualizan automáticamente cuando los datos cambian.

- Sistema de Eventos: Facilita la manipulación de eventos y la comunicación entre componentes. Los eventos personalizados pueden ser emitidos y escuchados, permitiendo una interacción eficaz entre componentes.
- Vue Router: Ofrece un enrutador oficial para construir aplicaciones de una sola página, permitiendo la navegación entre diferentes vistas de manera suave y eficiente.
- Vuex: Es un estado de gestión centralizado para aplicaciones Vue.js. Ayuda a manejar el estado de la aplicación de manera predecible y garantiza un flujo de datos unidireccional.

PrimeVue

Es una biblioteca de componentes para Vue.js que proporciona una colección de componentes UI (interfaz de usuario) de alta calidad y listos para usar. Está diseñada para facilitar el desarrollo de interfaces de usuario ricas y atractivas en aplicaciones Vue.js. (Anexo - 5)

Principales Características :

- Componentes de Calidad: PrimeVue ofrece una variedad de componentes de interfaz de usuario de alta calidad y totalmente responsivos. Estos componentes van desde elementos básicos como botones y formularios hasta componentes más complejos como tablas y gráficos.
- Documentación Exhaustiva: La biblioteca cuenta con una documentación completa que incluye ejemplos, guías de uso y detalles sobre cada componente. Esto facilita a los desarrolladores aprender y utilizar eficazmente los componentes de PrimeVue.
- Tema de PrimeVue: La biblioteca viene con un tema propio que puede ser fácilmente personalizado según las necesidades del proyecto. Esto permite una integración coherente de los componentes en la apariencia general de la aplicación.
- Integración con Vue.js: PrimeVue está diseñado específicamente para integrarse sin problemas con aplicaciones Vue.js. Puede utilizarse junto con otras bibliotecas y herramientas de Vue.js para construir interfaces de usuario completas.

- Soporte para Vue 3: PrimeVue es compatible tanto con Vue 2 como con Vue 3, lo que permite a los desarrolladores aprovechar las características más recientes de Vue.js.
- Amplia Variedad de Componentes: Incluye una amplia gama de componentes, como gráficos, tablas, árboles, calendarios, menús, y más. Estos componentes están diseñados para ser versátiles y adaptables a diferentes casos de uso.
- API Rica: PrimeVue proporciona una API rica que facilita la personalización y extensión de los componentes según las necesidades específicas del proyecto.

Vite

Es un entorno de desarrollo para construir aplicaciones web basadas en JavaScript y TypeScript. Es especialmente conocido por ser extremadamente rápido y eficiente durante el proceso de desarrollo. Desarrollado por Evan You, el creador de Vue.js, Vite se centra en ofrecer una experiencia de desarrollo rápida y optimizada al aprovechar la importación de módulos ES (ECMAScript) en el navegador.

Principales Características:

- Desarrollo Rápido: La principal característica de Vite es su capacidad para proporcionar un entorno de desarrollo extremadamente rápido. Logra esto al cargar y procesar módulos ES en tiempo real durante el desarrollo, eliminando la necesidad de transpilación de todo el código antes de la ejecución.
- Importación de Módulos en el Navegador: Vite aprovecha el soporte nativo de los navegadores para importar módulos ES directamente, sin necesidad de bundling (empaquetado) durante el desarrollo. Esto significa que cada módulo se puede cargar individualmente en el navegador según sea necesario, mejorando significativamente los tiempos de desarrollo.
- Multipágina y Monopágina: Vite es versátil y admite tanto aplicaciones multipágina como aplicaciones de una sola página (SPA). Puedes optar por trabajar con rutas tradicionales de varias páginas o aprovechar el enrutamiento de SPA según tus necesidades.

- Soporte para Vue.js y React: Aunque es independiente del framework, Vite se integra de manera nativa con Vue.js y React. Esto significa que puedes desarrollar aplicaciones utilizando Vue.js o React de la misma manera eficiente y rápida.
- Preprocesadores y Plugins: Vite admite preprocesadores de CSS como Sass, Less y Stylus, así como otros recursos como imágenes y fuentes. También es extensible mediante el uso de plugins para agregar funcionalidades adicionales según sea necesario.
- Modo de Producción Eficiente: Al igual que en el desarrollo, el modo de producción de Vite es eficiente. Utiliza la función de importación dinámica para generar un conjunto optimizado de archivos para la producción.
- Integración con TypeScript: Vite es compatible con TypeScript de manera nativa, permitiendo a los desarrolladores aprovechar las ventajas del sistema de tipos estáticos en sus aplicaciones.

JSON

(JavaScript Object Notation) es un formato de intercambio de datos ligero y legible por humanos. Fue derivado originalmente del lenguaje de programación JavaScript, pero se utiliza de forma independiente de cualquier lenguaje y es común en el desarrollo web y en la comunicación entre aplicaciones. (Anexo - 6)

Principales características:

- Formato de Texto: JSON se presenta en formato de texto y es legible por humanos. Está compuesto por pares clave-valor y estructuras de datos anidadas.
- Estructura de Datos: Los datos en JSON se representan como objetos, que son colecciones no ordenadas de pares clave-valor. Además de objetos, JSON admite arreglos, cadenas de texto, números, valores booleanos y valores nulos.
- Pares Clave-Valor: Un objeto JSON consiste en pares clave-valor, donde la clave es siempre una cadena y el valor puede ser un número, una cadena, un , otro objeto JSON, un array, o el valor nulo.
- Legibilidad y Sencillez: La sintaxis de JSON es simple y fácil de leer. Esto facilita tanto la generación como la interpretación de datos por parte de humanos y máquinas.

- Independiente del Lenguaje: JSON es independiente del lenguaje de programación, lo que significa que puede ser utilizado en una variedad de entornos y tecnologías.

Bibliotecas y frameworks de javascript

Axios

Es una biblioteca de JavaScript que se utiliza para realizar solicitudes HTTP desde el navegador o desde Node.js. Su principal objetivo es simplificar y estandarizar la forma en que se realizan solicitudes HTTP, ofreciendo una interfaz fácil de usar y promesas para trabajar con respuestas asíncronas. Axios se ha vuelto especialmente popular en el desarrollo de aplicaciones web y móviles, y es ampliamente utilizado junto con frameworks como Vue.js, React y Angular.

Principales características:

- Solicitud y Respuesta basada en Promesas: Axios utiliza el concepto de promesas para gestionar solicitudes y respuestas. Esto proporciona un enfoque más limpio y fácil de manejar en comparación con el uso de callbacks
- Interceptores de Solicitudes y Respuestas: Axios permite la definición de interceptores que se ejecutan antes o después de enviar una solicitud o recibir una respuesta. Esto es útil para agregar configuraciones globales, manejar errores comunes o realizar operaciones específicas en cada solicitud.
- Soporte para Peticiones Asíncronas: Axios es compatible con el uso de funciones asíncronas y await, facilitando la escritura de código asíncrono y la gestión de múltiples solicitudes de manera más limpia.
- Gestión Automática de JSON: Axios automáticamente convierte los datos de solicitud y respuesta a formato JSON, simplificando la manipulación de datos estructurados en las aplicaciones.
- Envío de Datos en el Cuerpo de la Solicitud: Permite enviar datos en el cuerpo de la solicitud, ya sea en formato JSON, URL-encoded o en otros formatos.

- **Manejo de Errores Mejorado:** Axios maneja automáticamente errores HTTP y permite la definición de interceptores de errores para gestionar situaciones específicas.
- **Cancelación de Solicitudes:** Proporciona funcionalidades para cancelar solicitudes, lo cual es útil en escenarios donde se realizan múltiples solicitudes y se necesita detener alguna de ellas.
- **Compatibilidad con el Navegador y Node.js:** Puede ser utilizado tanto en el navegador como en entornos de servidor basados en Node.js, lo que hace que Axios sea versátil y aplicable en una variedad de contextos de desarrollo.

NPM

(Node Package Manager), es el sistema de gestión de paquetes para Node.js. Es una herramienta esencial en el ecosistema de Node.js que permite a los desarrolladores instalar, compartir y gestionar las dependencias de sus proyectos de manera eficiente.

Principales características:

- **Instalación de paquetes:** npm permite a los desarrolladores instalar paquetes de software (bibliotecas, frameworks, herramientas, etc.) de manera sencilla. Los paquetes están disponibles a través del registro público de npm.
- **Gestión de dependencias:** npm facilita la gestión de dependencias en proyectos de Node.js. Al incluir un archivo package.json en el proyecto, los desarrolladores pueden especificar las dependencias y versiones exactas requeridas.
- **Scripts de tareas:** npm permite a los desarrolladores definir scripts de tareas en el archivo package.json. Estos scripts pueden ejecutar diversas operaciones, como iniciar el servidor, ejecutar pruebas, o construir el proyecto.
- **Versionamiento temático:** npm utiliza el concepto de versionamiento semántico para gestionar las versiones de los paquetes. Esto facilita la especificación de reglas para las actualizaciones de dependencias.
- **Registro público:** npm alberga un registro público que contiene una amplia variedad de paquetes de software. Los desarrolladores pueden compartir sus propios paquetes y contribuir al ecosistema de código abierto.

- Instalación global y local: Los paquetes pueden instalarse globalmente para ser utilizados en todo el sistema o localmente para un proyecto específico. Esto proporciona flexibilidad en la gestión de dependencias.
- Publicación de paquetes propios: Los desarrolladores pueden publicar sus propios paquetes en el registro público de npm, permitiendo a otros utilizar y contribuir a su código.
- Auditorías de seguridad: npm proporciona herramientas para auditar la seguridad de las dependencias de un proyecto y ayuda a los desarrolladores a identificar y corregir posibles problemas de seguridad.

Node.js

Node.js es un entorno de ejecución de JavaScript del lado del servidor que permite a los desarrolladores construir aplicaciones web escalables y de alto rendimiento. A diferencia de otros entornos de ejecución de JavaScript, como el navegador, Node.js está diseñado para ejecutarse en el servidor, lo que significa que puede realizar operaciones en el sistema de archivos, interactuar con bases de datos y gestionar solicitudes HTTP.Express.

Principales características:

- Motor V8 de Google: Node.js utiliza el motor V8 de Google, que es el mismo motor de JavaScript que impulsa el navegador Google Chrome. Esto proporciona un rendimiento rápido y eficiente en la ejecución de código JavaScript.
- Event-Driven y No Bloqueante: La arquitectura de Node.js es event-driven y no bloqueante, lo que significa que utiliza un modelo de I/O no bloqueante y maneja las operaciones de manera asíncrona. Esto permite manejar múltiples solicitudes simultáneamente sin bloquear el hilo de ejecución.
- Módulos y NPM: Node.js tiene un sistema de módulos incorporado que permite a los desarrolladores organizar su código en archivos modulares. Además, utiliza npm (Node Package Manager) para gestionar dependencias y paquetes de terceros.
- Librería Estándar: Node.js incluye una amplia librería estándar que proporciona funcionalidades para el manejo de archivos, eventos, redes, HTTP y más. Esto

facilita el desarrollo de aplicaciones sin depender en exceso de bibliotecas externas.

- Desarrollo Rápido: Node.js facilita el desarrollo rápido de aplicaciones mediante la reutilización de código

Nodemon

Es una herramienta que ayuda en desarrollo de aplicaciones Node.js reiniciando automáticamente la aplicación cuando detecta cambios en los archivos del proyecto. Esto significa que no necesitas reiniciar manualmente tu servidor cada vez que realizas modificaciones en el código fuente, lo que agiliza el proceso de desarrollo.

Principales características:

- Reinicio automático: Nodemon monitoriza los archivos de tu proyecto y reinicia automáticamente la aplicación cada vez que detecta cambios. Esto es especialmente útil durante el desarrollo, ya que ahorra tiempo y evita la necesidad de reiniciar manualmente el servidor.
- Compatibilidad con Node.js: Nodemon es compatible con proyectos Node.js y se puede utilizar con aplicaciones web, servidores API, aplicaciones de línea de comandos, y otros tipos de proyectos basados en Node.js.
- Fácil de usar: Nodemon se instala como una dependencia de desarrollo en tu proyecto a través de npm o yarn, y luego se ejecuta desde la línea de comandos. Puedes usarlo en lugar del comando **node** para ejecutar tu aplicación.
- Configuración avanzada: Nodemon permite la configuración personalizada a través de un archivo **nodemon.json** o mediante opciones de línea de comandos. Puedes especificar directorios para monitorizar, ignorar archivos y carpetas, y ajustar diversos aspectos del comportamiento de Nodemon.
- Soporte para proyectos con TypeScript: Nodemon es compatible con proyectos escritos en TypeScript. Puede observar y reiniciar automáticamente proyectos TypeScript cuando se realizan cambios en los archivos fuente.

Librería CORS

Es una biblioteca que en sí misma es una biblioteca específica en el sentido de tener una implementación única o centralizada. CORS es una especificación implementada en los

navegadores web para controlar el acceso a recursos en un dominio diferente (origen) al de la página web que está haciendo la solicitud.

Sin embargo, en el contexto de Node.js y la creación de servidores web, a menudo se hace referencia a la biblioteca "**cors**", que es una middleware que se puede utilizar con frameworks web como Express para facilitar la implementación de CORS en el lado del servidor.

Principales características:

- Control de acceso a recursos: CORS es necesario cuando una aplicación web en un dominio intenta realizar solicitudes a un servidor en otro dominio. CORS establece políticas que permiten o restringen el acceso a recursos en diferentes dominios.
- Cabeceras HTTP específicas: CORS utiliza cabeceras HTTP específicas, como Origin, Access-Control-Allow-Origin, Access-Control-Allow-Methods, y otras, para facilitar la comunicación entre el cliente y el servidor y definir qué solicitudes cruzadas están permitidas.
- Middleware para Node.js: La biblioteca **cors** para Node.js es una middleware que se integra con frameworks web como Express. Facilita la configuración de las cabeceras CORS necesarias para permitir o restringir el acceso desde diferentes orígenes.
- Configuración personalizada: La middleware **cors** permite una configuración personalizada para adaptarse a los requisitos específicos de la aplicación. Puedes especificar los orígenes permitidos, los métodos de solicitud permitidos, las cabeceras personalizadas, entre otras opciones.
- Manejo de credenciales: CORS admite la posibilidad de enviar y recibir credenciales (como cookies y encabezados de autorización) entre dominios si se configura correctamente tanto en el servidor como en el cliente.

Base de datos y SQL

SQL

Este es un lenguaje de programación diseñado para gestionar y manipular bases de datos relacionales. SQL es un acrónimo de (Structured Query Language), en la cual es un estándar en el mundo de la gestión de datos y se utiliza para realizar diversas operaciones en bases de datos, como la creación, consulta, actualización y eliminación de datos.

Principales características:

- **Lenguaje declarativo:** SQL es un lenguaje declarativo, lo que significa que describes qué resultados quieres obtener, pero no especificas cómo obtenerlos. Esto contrasta con los lenguajes imperativos, donde defines paso a paso cómo se deben realizar las operaciones.
- **Operaciones CRUD:** SQL se utiliza para realizar las operaciones fundamentales de manipulación de datos, conocidas como operaciones CRUD:
 - **Crear (Create):** INSERT - Para agregar nuevos registros a una tabla.
 - **Leer (Read):** SELECT - Para consultar datos de una tabla.
 - **Actualizar (Update):** UPDATE - Para modificar datos existentes en una tabla.
 - **Eliminar (Delete):** DELETE - Para eliminar registros de una tabla.
- **Gestión de bases de datos relacionales:** SQL se utiliza principalmente en sistemas de gestión de bases de datos relacionales (RDBMS), como MySQL, PostgreSQL, SQLite, Microsoft SQL Server y Oracle. Estas bases de datos siguen el modelo relacional, que organiza los datos en tablas con relaciones.
- **Tipos de datos:** SQL define varios tipos de datos que pueden ser utilizados para almacenar información en una base de datos. Algunos ejemplos comunes incluyen INTEGER (enteros), VARCHAR (cadenas de texto), DATE (fechas) y BOOLEAN (valores lógicos).
- **Restricciones (Constraints):** SQL permite aplicar restricciones a las tablas para garantizar la integridad de los datos. Ejemplos de restricciones incluyen PRIMARY KEY (clave primaria), FOREIGN KEY (clave externa), UNIQUE (valores únicos), y NOT NULL (no nulo).

PostgresSQL

Es un sistema de gestión de bases de datos relacionales de código abierto, en la que se utiliza para almacenar, manipular, y recuperar datos. PostgreSQL es conocido por su fiabilidad, flexibilidad soporte de estándares técnicos abiertos. A diferencia de otros sistemas de gestión de bases de datos relacionales y no relacionales. Es conocido por su robustez, capacidad para manejar grandes cantidades de datos y soporte para características avanzadas. PostgreSQL sigue los principios del modelo de base de datos relacional y se adhiere al estándar SQL. (Anexo – 7)

Principales características:

- Bases de datos relacional: PostgreSQL sigue el modelo relacional, organizando los datos en tablas con relaciones entre ellas. Esto facilita la gestión de datos estructurados y la realización de consultas complejas.
- Estándar SQL: PostgreSQL sigue los estándares SQL y, además, agrega extensiones propias. Esto significa que es compatible con la mayoría de las aplicaciones que utilizan SQL estándar.
- Soporte para tipos de datos avanzados: PostgreSQL ofrece una amplia variedad de tipos de datos, incluyendo tipos numéricos, de texto, de fecha, geométricos, JSON, y muchos más. También permite la creación de tipos de datos personalizados.
- Extensiones y contribuciones: PostgreSQL permite la creación y uso de extensiones, lo que facilita la adición de nuevas funcionalidades al sistema. También tiene un sistema de contribuciones que incluye módulos adicionales desarrollados por la comunidad.
- Soporte para transacciones ACID: PostgreSQL garantiza la consistencia de las transacciones utilizando el estándar ACID (Atomicidad, Consistencia, Aislamiento y Durabilidad). Esto asegura que las transacciones se realicen de manera segura y confiable.
- Replicación y alta disponibilidad: Gracias a la gestión eficiente de la información, podemos decir que es esencial para el funcionamiento exitoso en la era digital debido que surge la alta disponibilidad constante así como la integridad de datos

siendo factores críticos que tiene un gran impacto en toma de decisiones o operaciones debido a esto, se considera de alta disponibilidad.

Procedimientos almacenados

Los procedimientos almacenados son un conjunto de comandos SQL que se almacenan en una base de datos y pueden ejecutarse llamándolos desde un programa o de una aplicación. Los procedimientos almacenados se utilizan para realizar diversas tareas, como recuperar datos en una base de datos, insertar dos en una base de datos o modificar una base de datos. (Anexo – 8)

Vistas de sql.

Principales características:

- Encapsulación de lógica de negocios: Los procedimientos almacenados encapsulan la lógica de negocio y las operaciones complejas en un conjunto de instrucciones SQL. Esto facilita la gestión y el mantenimiento del código, ya que la lógica está centralizada en un único lugar.
- Reutilización de código: Al almacenar la lógica en un procedimiento almacenado, puedes reutilizar el código en varias partes de una aplicación o en diferentes aplicaciones. Esto promueve la consistencia y evita la duplicación de código.
- Optimización de rendimiento: Los procedimientos almacenados pueden ayudar a mejorar el rendimiento de la base de datos al reducir la cantidad de datos transferidos entre la base de datos y la aplicación. Al ejecutar la lógica del lado del servidor, se minimiza la latencia de red.
- Transacciones y control de concurrencia: os procedimientos almacenados pueden ejecutarse dentro de transacciones, permitiendo que un conjunto de instrucciones se complete de manera exitosa o se revierta si ocurre un error. Esto contribuye al control de la concurrencia y garantiza la integridad de los datos.
- Seguridad: Los procedimientos almacenados pueden tener permisos de ejecución asignados a roles específicos, lo que permite controlar quién puede ejecutar o

modificar la lógica encapsulada. Esto contribuye a la seguridad y al control de acceso.

Vistas en SQL

Una vista en SQL es una tabla virtual que se define mediante una consulta. A diferencia de las tablas físicas, las vistas no contienen datos, si no que se construyen a partir de los datos almacenados en una o mas tablas. Las vistas se utilizan para simplificar las consultas complejas y para proporcionar una capa adicional de seguridad al ocultar los detalles de la estructura de la base de datos. Las vistas pueden ser creadas utilizando la sintaxis *Create View*

Seguida de una consulta que define la vista. Una vez creada, la vista se puede utilizar como cualquier otra tabla en la base de datos, las vistas se pueden actualizar, lo que permite modificar los datos que se muestran en la vista. (Anexo - 9)

Principales características:

- Representación lógica de datos: Una vista proporciona una representación lógica de los datos almacenados en una o más tablas. Esta representación abstracta puede ocultar la complejidad de las relaciones y consultas subyacentes.
- Encapsulación de logica: Al crear una vista, puedes encapsular la lógica de una consulta compleja en un objeto que puede ser referenciado fácilmente en otras consultas. Esto ayuda a simplificar el código y promueve la reutilización.
- Seguridad y control de accesos: Las vistas pueden utilizarse para controlar el acceso a los datos. Puedes limitar las columnas visibles o las filas accesibles en una vista para garantizar que solo ciertos usuarios vean la información necesaria.
- Simplificación de consultas: Al utilizar vistas, puedes simplificar consultas complejas al proporcionar una capa de abstracción sobre las tablas subyacentes. Esto es particularmente útil cuando hay consultas frecuentes o complejas que se repiten en varias partes de una aplicación.
- Mejora de rendimiento: En algunos casos, las vistas pueden mejorar el rendimiento al precalcular resultados o al almacenar en caché resultados

comunes. Sin embargo, esto depende del sistema de gestión de bases de datos específico y de la complejidad de la vista.

- Actualización de datos: En general, las vistas no almacenan datos físicamente, sino que son consultas que se ejecutan en tiempo real. Al realizar operaciones de inserción, actualización o eliminación en una vista, estas operaciones se traducen en operaciones en las tablas subyacentes.
- Independencia de esquemas Las vistas pueden proporcionar una capa de independencia de esquema al ocultar la estructura real de la base de datos. Esto permite realizar cambios en las tablas subyacentes sin afectar directamente a las aplicaciones que utilizan las vistas.

API y comunicación

API REST

Es un estilo arquitectónico para diseñar servicios web que utiliza un conjunto de restricciones y principios específicos para lograr una comunicación eficiente y escalable entre sistemas distribuidos. Las APIs REST se basan en el protocolo HTTP (Hypertext Transfer Protocol) y se centran en la representación de recursos y las operaciones estándar sobre estos recursos. (Anexo – 10)

Principales características:

- Arquitectura basada en recursos: En una API REST, los recursos son entidades identificables, como objetos o servicios, que pueden ser manipulados mediante las operaciones estándar de HTTP (GET, POST, PUT, DELETE). Cada recurso tiene una URL única que actúa como su identificador.
- Transferencia de representación: Las representaciones de recursos se transfieren entre el cliente y el servidor en formato de datos estándar, como JSON o XML. El cliente puede solicitar una representación específica del recurso mediante la especificación de encabezados en la solicitud HTTP.
- Operaciones estándar del HTTP: las operaciones CRUD (Crear, Leer, Actualizar, Eliminar) se mapean a las operaciones estándar de HTTP:
 - **GET:** Obtener un recurso.
 - **POST:** Crear un nuevo recurso.

- **PUT o PATCH:** Actualizar un recurso existente.
- **DELETE:** Eliminar un recurso.
- Estado representacional: La respuesta a una solicitud REST debe contener toda la información necesaria para entender y modificar el estado del recurso. Esto se logra mediante la transferencia de una representación del recurso solicitado.
- Sin estado (Stateless): Cada solicitud del cliente al servidor en una API REST contiene toda la información necesaria para comprender y procesar la solicitud. El servidor no mantiene ningún estado sobre el cliente entre solicitudes, lo que mejora la escalabilidad y la confiabilidad.
- Interfaz uniforme: Una interfaz uniforme simplifica y normaliza la interacción entre clientes y servidores. Esto incluye la identificación de recursos, la manipulación de recursos a través de representaciones y la navegación entre recursos mediante hipervínculos.
- Sin acoplamiento (Loose coupling): La arquitectura REST promueve la independencia entre el cliente y el servidor, lo que significa que ambos pueden evolucionar de manera independiente sin afectar la operación del otro.

Sockets

Los sockets en el contexto de sistemas operativos (SO) y programación se refieren a un mecanismo de comunicación entre procesos, tanto en la misma máquina (comunicación interproceso o IPC) como entre máquinas distintas a través de una red. Un socket proporciona un punto de conexión entre procesos para el intercambio de datos.

Principales características:

- Comunicación entre procesos: Los sockets permiten la comunicación bidireccional entre procesos, lo que significa que ambos procesos pueden enviar y recibir datos. Esta comunicación puede ocurrir tanto en la misma máquina (comunicación interproceso local) como a través de una red (comunicación interproceso remota).
- Protocolo de comunicación: Los sockets están asociados con un protocolo de red específico, como TCP (Transmission Control Protocol) o UDP (User Datagram Protocol). TCP proporciona una comunicación orientada a la conexión y confiable,

mientras que UDP es más liviano y ofrece una comunicación no orientada a la conexión y sin garantía de entrega.

- Dirección IP y puerto: En el caso de la comunicación a través de una red, los sockets se identifican por una dirección IP y un número de puerto. La dirección IP indica la ubicación del proceso en la red, y el número de puerto identifica un servicio específico en ese proceso.
- Creación y conexión: En el proceso de comunicación, un socket se crea y se vincula a una dirección y un puerto específicos. Luego, puede escuchar conexiones entrantes (en el caso de TCP) o enviar y recibir datos directamente (en el caso de TCP o UDP).
- Modelo cliente servidor: El uso común de los sockets es en el modelo cliente-servidor, donde un proceso actúa como el servidor esperando conexiones entrantes, y otros procesos actúan como clientes que se conectan al servidor para intercambiar datos.
- Comunicación sin conexión (UDP): Los sockets UDP no mantienen una conexión continua entre los extremos. En cambio, los mensajes se envían de manera independiente, y no hay garantía de entrega ni de orden de llegada. Esto hace que UDP sea más adecuado para aplicaciones que pueden tolerar cierta pérdida de datos, como transmisiones de video en tiempo real.

Socket.IO

Es una biblioteca de JavaScript que permite la comunicación en tiempo real bidireccional y basada en eventos entre el servidor y el cliente. Socket.IO se utiliza comúnmente en el desarrollo web para construir aplicaciones en tiempo real, como chats en línea, juegos en línea, actualizaciones en tiempo real y otras aplicaciones colaborativas.

(Anexos 11 – 17).

Principales características:

- Comunicación en tiempo real: Socket.IO proporciona una capa de abstracción sobre los WebSockets y otros mecanismos de transporte para facilitar la comunicación en tiempo real entre el cliente y el servidor.
- Detección automática de transporte: Socket.IO utiliza una variedad de transportes para la comunicación, incluyendo WebSockets, AJAX, y otros. Selecciona automáticamente el mejor transporte disponible según las capacidades del cliente y del servidor.
- Emisión y escucha de eventos: La comunicación en Socket.IO se basa en el principio de eventos. Tanto el cliente como el servidor pueden emitir eventos y escuchar eventos emitidos por la otra parte. Esto facilita la interacción bidireccional y la transmisión de datos estructurados.

Seguridad y hashing

SHA – 256

Conocido como Secure Hash Algorithm 256-bit, es una función hash de tipo criptográfica, la cual pertenece a la familia de funciones conocida como hash SHA – 2. Donde nos proporciona un valor hash de tamaño fijo de 256 bits. Fue diseñada por la Agencia de Seguridad Nacional de los Estados Unidos, donde la función hash toma aquellos datos de entrada, conocidos normalmente como mensajes, el cual produce una cadena de caracteres tipo alfanuméricos de longitud fija, normalmente como se menciono es de 256 bits.

La importancia de SHA – 256 tiene varias aplicaciones de seguridad y criptografía, de los cuales son:

- Almacenamiento seguro de contraseñas: Se puede almacenar contraseñas como hashes en vez de usar un texto plano, dando una mayor y alta seguridad a estas contraseñas.
- Firmas digitales: Se garantiza mayor seguridad en cuanto a la autenticidad e integridad de mensajes y archivos mediante la generación y verificación de firmas digitales necesario para conocimiento del mismo.

- Autenticidad: Como se menciona anteriormente, se verifica la autenticidad, tanto de datos como mensajes mediante comparación para la generación de claves seguras y únicas.
- Generar claves criptográficas: Son elementos esenciales en sistemas de seguridad y criptografía, siendo usadas para cifrar y descifrar información, garantizando la autenticidad e integridad de los datos.

Gracias a esto, se considera SHA -256 por la resistencia en cuanto a colisiones y siendo considerado como seguro para una variedad de aplicaciones criptográficas, siendo un conjunto de funciones que tienen un gran valor en cuanto seguridad informática, así como protección de datos para mayor autenticidad.

Plataformas y despliegue

AWS

(Amazon Web Services) es una plataforma de servicios en la nube ofrecida por Amazon.com. Se lanzó en 2006 y ha crecido para convertirse en uno de los proveedores líderes de servicios en la nube, proporcionando una amplia gama de servicios y soluciones para empresas y desarrolladores. AWS ofrece servicios en la nube en áreas como cómputo, almacenamiento, bases de datos, inteligencia artificial, Internet de las cosas (IoT), análisis, seguridad, y más.

Principales características:

- Computo en la nube: AWS ofrece servicios de cómputo escalables y flexibles que incluyen instancias de máquinas virtuales (EC2), contenedores (Elastic Container Service - ECS, Kubernetes - EKS), y funciones sin servidor (Lambda). Estos servicios permiten a los usuarios ejecutar aplicaciones y cargas de trabajo en la nube de manera eficiente.
- Almacenamiento y bases de datos: AWS proporciona una variedad de opciones de almacenamiento, como Amazon S3 para almacenamiento de objetos, Amazon EBS para almacenamiento de bloques, y Amazon RDS para bases de datos

relacionales. También ofrece soluciones de bases de datos NoSQL, como Amazon DynamoDB.

- Redes y contenido distribuido: AWS permite la creación de redes virtuales personalizadas con Amazon VPC. También ofrece servicios de entrega de contenido (CDN) como Amazon CloudFront para acelerar la entrega de contenido a usuarios finales
- Inteligencia artificial y aprendizaje automático: Con servicios como Amazon SageMaker, AWS proporciona herramientas para desarrollar, entrenar e implementar modelos de aprendizaje automático. También ofrece servicios preentrenados para tareas específicas, como reconocimiento de imágenes y procesamiento del lenguaje natural.
- Seguridad y identidad: AWS proporciona herramientas y servicios para garantizar la seguridad de las aplicaciones y los datos, como AWS Identity and Access Management (IAM) para gestionar el acceso a recursos, y AWS Key Management Service (KMS) para gestionar claves de cifrado.

Vercel

Es una plataforma en la nube que se especializa en la implementación (deploy) y alojamiento de aplicaciones web. La plataforma está diseñada para simplificar el proceso de desarrollo, implementación y entrega continua de sitios web y aplicaciones front-end. Vercel es conocido por su integración estrecha con marcos de desarrollo populares y su enfoque en proporcionar una experiencia sin configuración para los desarrolladores.

Principales características:

- Implementación continua: Vercel facilita la implementación continua al proporcionar integración nativa con repositorios de Git (GitHub, GitLab, Bitbucket) y permitir la implementación automática cada vez que se realiza un cambio en el código base.
- Sin configuración: Vercel sigue el principio de "cero configuración" para la mayoría de las aplicaciones. Esto significa que muchos proyectos se pueden implementar sin necesidad de configuración adicional, lo que simplifica el proceso de implementación.

- Soporte para frameworks front-end: Vercel es compatible con una variedad de marcos y bibliotecas front-end populares, como React, Angular, Vue.js, Svelte y otros. Puede detectar automáticamente el tipo de proyecto y proporcionar una configuración adecuada.
- Rapida distribución de contenido: Vercel utiliza una red global de entrega de contenido (CDN) para distribuir el contenido de manera rápida y eficiente a nivel mundial. Esto garantiza una baja latencia y un rendimiento óptimo para los usuarios finales.
- Previsualización automática (Preview): Vercel ofrece funcionalidades de previsualización automática para cada rama de un repositorio Git. Esto permite a los desarrolladores visualizar cambios antes de fusionarlos con la rama principal.
- Escalabilidad automática: Vercel maneja automáticamente la escalabilidad de las aplicaciones, asegurando que las aplicaciones puedan manejar cargas de tráfico variables y se adapten a las demandas del usuario.
- Funciones sin servidor: Vercel permite la creación y implementación de funciones sin servidor que se pueden utilizar para manejar lógica del lado del servidor en respuesta a eventos específicos. Estas funciones son escalables y se ejecutan de forma automática.

Github

Es una plataforma de desarrollo colaborativo que utiliza el sistema de control de versiones Git. Fue creada para facilitar la colaboración entre desarrolladores en proyectos de software, pero su uso se ha extendido a otras áreas. GitHub proporciona una interfaz web y servicios para repositorios de código, seguimiento de problemas (issue tracking), integración continua, y otras funciones relacionadas con el desarrollo de software.

Principales características:

- Repositorios: GitHub almacena proyectos de software en lo que se llama "repositorios". Un repositorio es un espacio de almacenamiento para el código fuente, archivos de configuración y otros recursos relacionados con un proyecto
- Control de versiones con git: Git es un sistema de control de versiones distribuido, y GitHub utiliza Git para gestionar las versiones de código. Esto permite a los desarrolladores trabajar en paralelo en diferentes funciones y fusionar sus cambios de manera eficiente.
- Colaboraciones y contribuciones: itHub facilita la colaboración entre desarrolladores permitiendo la contribución de código mediante solicitudes de extracción (pull requests). Los desarrolladores pueden colaborar en proyectos abiertos o privados.
- Gestión de problemas y tareas: GitHub proporciona un sistema de seguimiento de problemas (issue tracking) que permite a los equipos registrar problemas, realizar un seguimiento de tareas y discutir cambios propuestos.

Herramientas de desarrollo

Visual Studio

Visual Studio es conocido como aquel entorno de desarrollo integrado (IDE), siendo desarrollado por Microsoft, nos proporciona una variedad y conjunto de herramientas como servicios que nos facilitan el desarrollo de software en los diversos lenguajes de programación y aplicaciones.

Gracias a esto nos soporta múltiples lenguajes como java, c++, Python, SQL, entre una variedad sin fin de lenguajes. Además nos proporciona herramientas de depuración para detectar y corregir errores de una manera eficiente. Algo muy indispensable del Visual Studio es que podemos agregar extensiones, así como complementos para adaptar el entorno a nuestras necesidades requeridas.

Postman

Postman es una plataforma que ofrece herramientas para el desarrollo de APIs (Interfaces de Programación de Aplicaciones). Es ampliamente utilizado por

desarrolladores y equipos de desarrollo para simplificar el proceso de prueba, desarrollo y documentación de APIs. Postman proporciona una interfaz gráfica de usuario que permite a los usuarios realizar solicitudes HTTP, organizar y automatizar pruebas, y colaborar en el desarrollo de APIs.

Conceptos avanzados

Funciones asíncronas en javascript

Las funciones asíncronas en JavaScript son una característica que permite escribir código asíncrono de manera más legible y estructurada. Estas funciones son una forma de trabajar con operaciones asíncronas de una manera más fácil y comprensible, utilizando la palabra clave `async` antes de la palabra clave `function`. Las funciones asíncronas trabajan junto con el concepto de Promesas (Promise), que es otra característica de JavaScript utilizada para gestionar operaciones asíncronas. Al usar funciones asíncronas, se simplifica la sintaxis para trabajar con promesas y se proporciona una forma más clara de estructurar el código asíncrono.

Otros

Procesamiento del lenguaje natural (NLP)

Es una rama de la inteligencia artificial que se ocupa de la interacción entre las computadoras y el lenguaje humano. Su objetivo principal es permitir que las máquinas comprendan, interpreten y generen texto de manera similar a cómo lo hacen los humanos. El NLP combina conocimientos de lingüística, informática y aprendizaje automático para lograr este propósito.

Librería0 xlsx

La referencia a "la librería xlsx" podría hacer referencia a varias bibliotecas o herramientas que trabajan con archivos XLSX, que es el formato de archivo asociado a

Microsoft Excel a partir de la versión 2007. Aquí, te proporcionaré información sobre una de las bibliotecas comunes llamada **SheetJS/js-xlsx**.

SheetJS/js-xlsx:

Esta es una biblioteca JavaScript que permite leer y escribir archivos en formato Excel (.xlsx) en el lado del cliente o del servidor. Es especialmente útil para trabajar con hojas de cálculo en aplicaciones web y para manipular datos en formato Excel de manera programática.

Principales características:

- Lectura y escritura de archivos excel:
- Compatibilidad con diferentes plataformas:
- Manipulación de datos:
- Soporte de formulas

Libreria moment

La biblioteca **Moment.js** es una librería de manipulación de fechas y horas en JavaScript. Facilita la manipulación, formateo y análisis de fechas en una variedad de formatos. Moment.js ha sido ampliamente utilizada en proyectos JavaScript y se ha vuelto muy popular gracias a su facilidad de uso y su capacidad para gestionar operaciones complejas con fechas.

Principales características:

- Manipulación de fechas y horas: moment.js permite la creación y manipulación sencilla de objetos de fecha y hora. Puedes realizar operaciones como sumar o restar días, meses, años, horas, minutos, etc.
- Formateo y análisis: Moment.js facilita el formateo y análisis de fechas en diferentes formatos. Puedes especificar patrones para mostrar fechas de manera personalizada.
- Comparación de fechas: Puedes comparar fechas para determinar cuál es anterior o posterior. Moment.js proporciona métodos para realizar comparaciones y determinar la diferencia entre dos fechas.
- Manejo de zonas horarias: Moment.js permite trabajar con zonas horarias, lo que es útil al tratar con fechas en diferentes regiones del mundo.

- Cálculos relativos: oment.js facilita el cálculo de fechas relativas, como "hace 2 horas" o "dentro de 3 días".

Producto final

Como producto final, una vez expuesto toda tecnología realizada damos por terminado nuestro proyecto aplicado todo lo mencionado, teniendo como producto final las siguientes representaciones gráficas, cumpliendo con éxito el desarrollo de tal aplicación, una vez resolviendo la problemática dada.

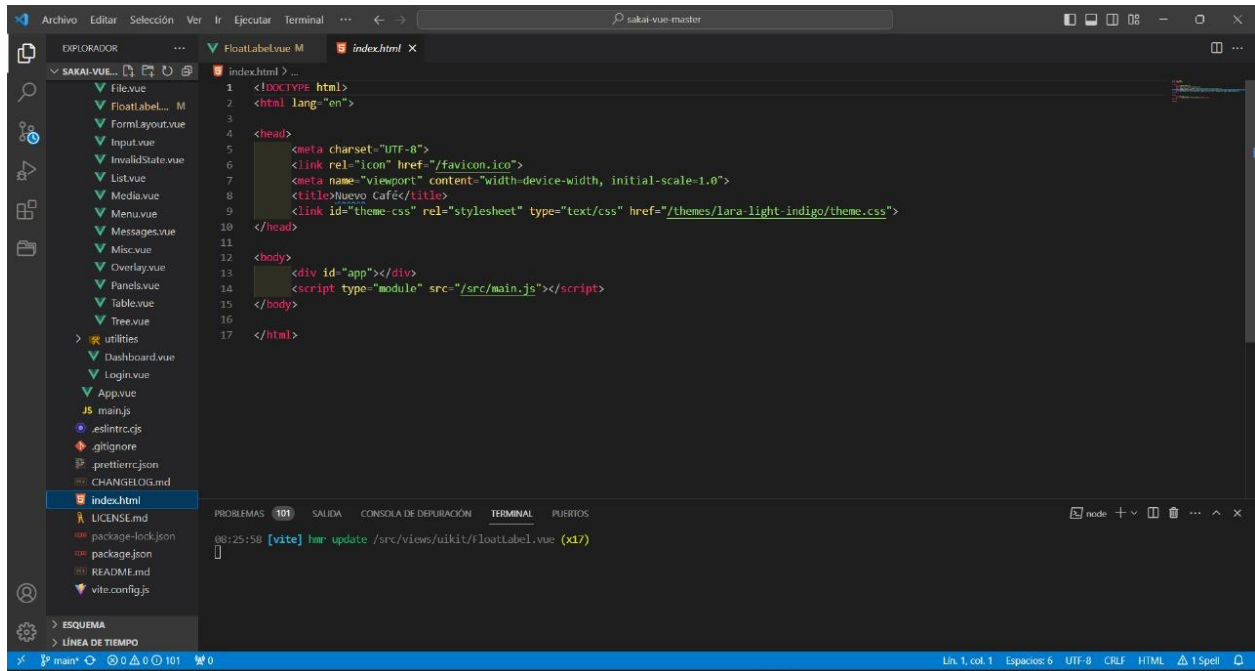
- Representación del inicio de sesión para el usuario: se da a ingresar el inicio de sesión a nuestro sistema, ingresando el email además de la contraseña para iniciar sesión y continuar con la aplicación. (Anexo – 18)
- Representación del menú principal: se presenta el inicio principal de nuestra aplicación, mostrando visualmente el chat bot realizado y aplicado para dar un breve conocimiento al usuario acerca del proceso del café. (Anexo - 19)
- Representación de terrenos y lotes: se muestra visual, así como acciones tanto editar como eliminar los terrenos y lotes que desee el usuario para mayor facilidad, además de agregar la opción de agregar un nuevo artículo deseado. (Anexo - 20)
- Representación de proceso de café recolección: se muestra visual para cargar los lotes en caso de a ver agregados nuevos y realizar la recolección, donde si al cargar uno nuevo, se añade el valor de los campos de tipo de café que desee agregar, una vez agregado se actualiza la fecha y total de cantidad de café, dando como finalizado tal proceso para pasar al siguiente. (Anexo - 21)
- Representación de proceso de café despulpado: se muestra visual para cargar la recolección siempre y cuando estén finalizados en tal proceso anterior, donde

se permite finalizar el mismo y dando paso al siguiente proceso de café o en su caso contrario, eliminar tal proceso. (Anexo - 22)

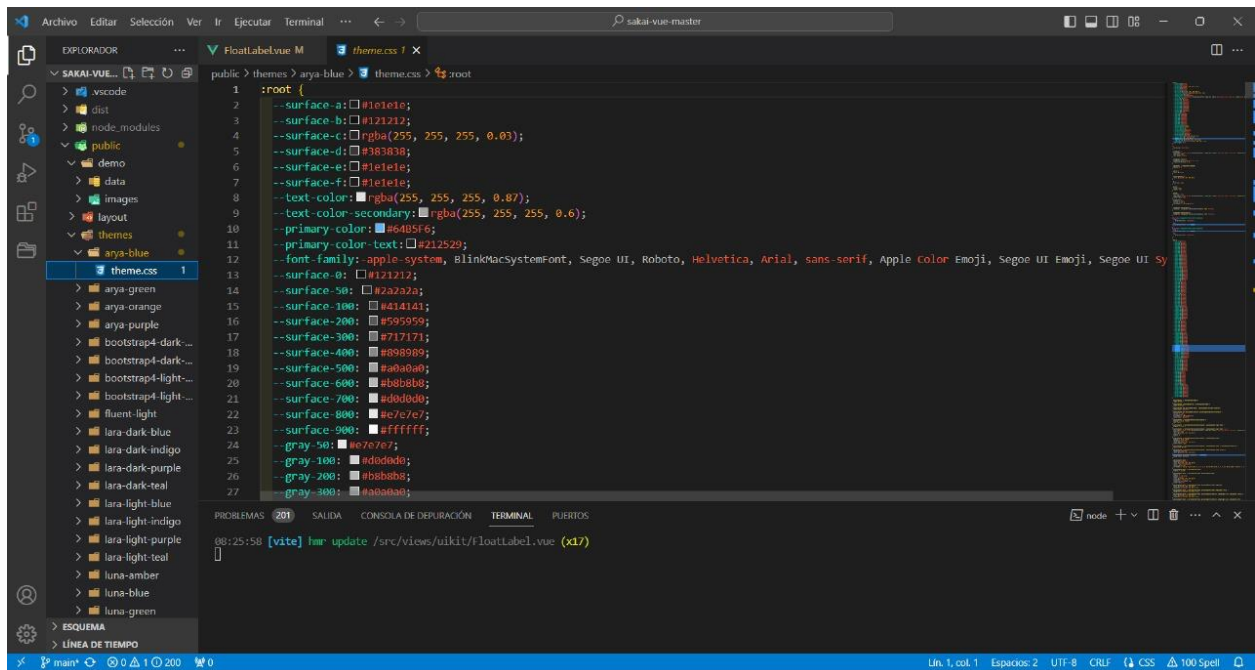
- Representación de proceso de café fermentación: se muestra visual para agregar un tanque además de mostrar los tanques ingresados anteriormente, o eliminar en caso contrario. También se muestran aquellos lotes de café que pasaron el proceso de despulpado, mostrando solo aquellos lotes que terminaron tal proceso, además de elegir la temperatura mínima como máxima y asignar el tanque, una vez realizado esto, se bloquea el tanque dando a conocer como ocupado y se inicia la fermentación hasta que el usuario de finalizado tal proceso de café. (Anexo - 23)
- Representación de proceso de café lavado: se muestra una visual para el lavado, donde se cargan aquellos lotes de café que solamente estén finalizados en la fermentación, donde se da a conocer los mismos, dando continuación de finalizado para permitir pasar al siguiente proceso de café. (Anexo - 24)
- Representación de proceso secado: se muestra visual para el secado, dando a conocer dos tablas para agregar cuartos de secado, donde puedes realizar edición o eliminación del mismo, además de conocer sus respectivos valores, por otra parte, en secado, se tienen los lotes a secar además de listarlos, se mostraran aquellos lotes que ya pasaron por el proceso de lavado y solamente se mostraran aquellos que finalizaron tal proceso, se elige cuando inicia y se genera automáticamente la hora y fecha de inicio, de la misma manera una vez finalizado se puede generar la finalización del mismo. (Anexo - 25)
- Representación de reportes: se muestra todos los reportes necesarios para la consulta del usuario, donde solamente el primero se ingresa el valor, ya que se desea saber específicamente el lote para la consulta del mismo, estos lotes son:

- Mostrar el tiempo que dura el proceso de un determinado lote de café, desde la recolección hasta el secado. (Anexo - 26)
- Mostrar los datos de la fase en la que se encuentra cada lote de café. (Anexo - 27)
- Mostrar los tanques y cuartos disponibles en cualquier momento. (Anexo - 28)
- Mostrar cual es el mes en el que se ha recolectado más cantidad de café (Anexo -29)
- Mostrar cual es el tipo de café que se ha recolectado más en un año (verde, maduro, seco o sobre maduro). (Anexo – 30)
- Mostrar el promedio del tiempo al mes del secado de los lotes de café. (Anexo – 31)

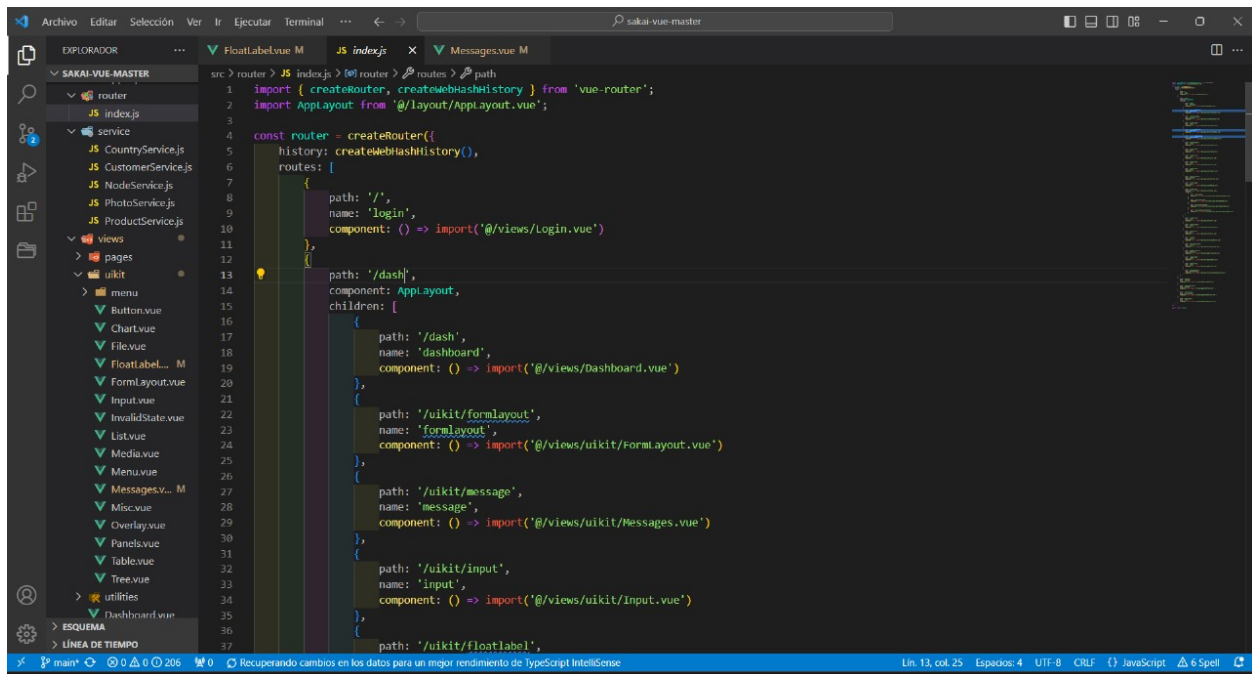
Anexos



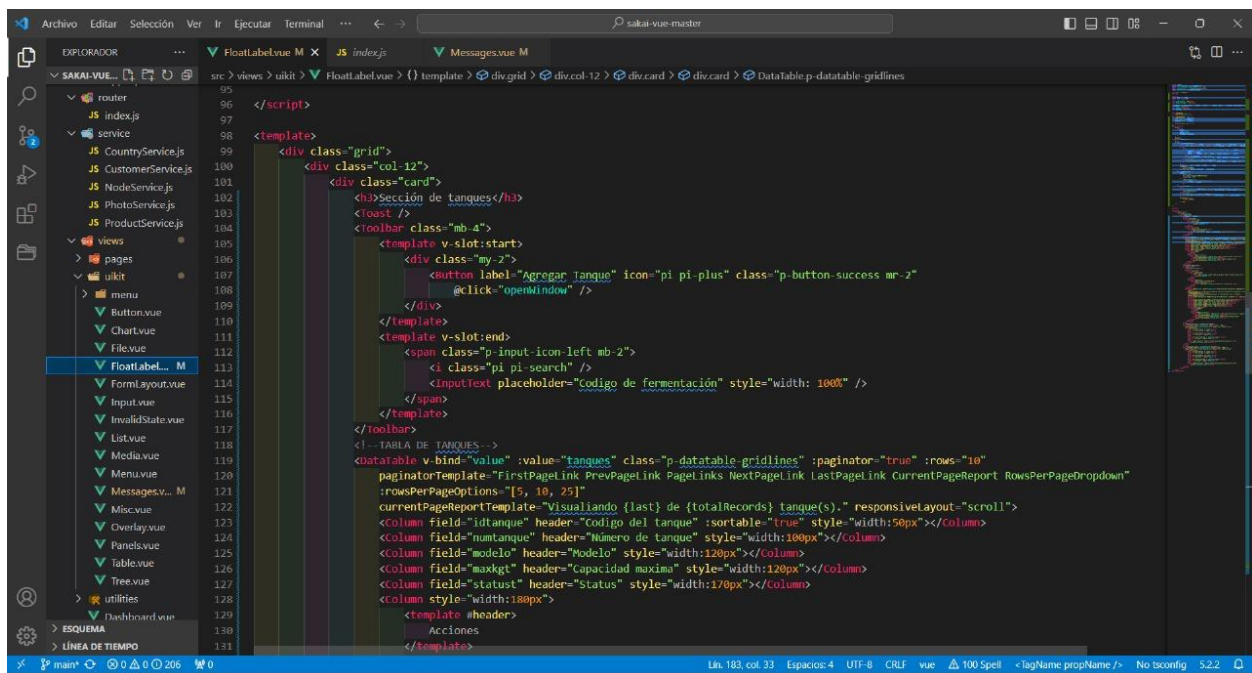
Anexo 1



Anexo 2



Anexo 3



Anexo 4

The Most Complete UI Suite for Vue.js

Elevate your web applications with PrimeVue's comprehensive suite of customizable, feature-rich UI components. With PrimeVue, turning your development vision into reality has never been easier.

[Get Started →](#)[Give a Star ★](#)

Features

PrimeVue is the most complete solution for your UI requirements.



80+ UI Components

The ultimate set of UI Components to assist you with 80+ impressive Vue Components.

Anexo 5

```
src > controllers > JS controllers.js > [0] updateTanques
195 const updateTanques = async (req, res) => {
196   const idtanque = req.params.id;
197   const { numtanque } = req.body;
198   const { modelo } = req.body;

PROBLEMAS (362) SALIDA CONSOLA DE DEPURACIÓN TERMINAL PUERTOS
node + v ...

hora: null,
statusd: 'PENDIENTE'
},
{
  iddespulpado: 18,
  idrecoleccion: 764,
  fecha: null,
  hora: null,
  statusd: 'PENDIENTE'
},
{
  iddespulpado: 19,
  idrecoleccion: 762,
  fecha: null,
  hora: null,
  statusd: 'PENDIENTE'
},
{
  iddespulpado: 20,
  idrecoleccion: 763,
  fecha: null,
  hora: null,
  statusd: 'PENDIENTE'
},
{
  iddespulpado: 21,
  idrecoleccion: 764,
  fecha: 2023-11-13T06:00:00.000Z,
  hora: 2023-11-13T19:17:50.000Z,
  statusd: 'FINALIZADO'
},
{
  iddespulpado: 10,
  idrecoleccion: 762,
  fecha: 2023-11-13T06:00:00.000Z,
  hora: 2023-11-13T19:17:38.000Z,
```

Anexo – 6

```
C: > Users > alexg > Downloads > fincaActPrimaria (1).sql
1 create database FincaOriginal;
2
3 --\c finca
4
5
6
7 create table terreno(
8   numTerreno integer NOT NULL GENERATED ALWAYS AS IDENTITY,
9   nombreT varchar(50) NOT NULL,
10  primary key(numTerreno)
11 );
12 +
13 create table loteCafe(
14   numLote integer NOT NULL GENERATED ALWAYS AS IDENTITY,
15   tipoGrano varchar(25) NOT NULL,
16   numTerreno integer NOT NULL,
17   primary key(numLote),
18   CONSTRAINT numTerreno foreign key (numTerreno)
19   References terreno(numTerreno) MATCH SIMPLE ON UPDATE NO ACTION ON DELETE CASCADE
20 );
21
22 create table Recoleccion(
23   idRecoleccion integer NOT NULL GENERATED ALWAYS AS IDENTITY,
24   numLote integer NOT NULL,
25   cantTotalCafe numeric(10, 2) DEFAULT NULL constraint cantTotalCafeInv check(cantTotalCafe > 0),
26   fecha date,
27   statusR varchar(25) DEFAULT 'PENDIENTE' NOT NULL,
28   primary key(idRecoleccion),
29   CONSTRAINT numLoteFKKey foreign key (numLote)
30   References loteCafe(numLote) MATCH SIMPLE ON UPDATE NO ACTION ON DELETE CASCADE
31 );
32
33 create table fincaCafe(
```

Anexo 7

```
351 -----
352 -----PLs para las tablas - FERMENTACION
353 -----
354
355 CREATE OR REPLACE PROCEDURE crearFermentacion()
356 AS $$
357 DECLARE
358
359
360     curredespul CURSOR FOR SELECT idDespulpado AS idDes FROM despulpado WHERE statusD = 'FINALIZADO' ORDER BY idDespulpado;
361     vcursordespu RECORD;
362     vIdDespulpado INTEGER;
363
364 BEGIN
365
366     OPEN curredespul;
367
368     LOOP
369         FETCH curredespul INTO vcursordespu;
370         EXIT WHEN NOT FOUND;
371
372         -- Verificar si el idDespulpado ya existe en la tabla Fermentacion
373         SELECT idDespulpado INTO vIdDespulpado FROM Fermentacion WHERE idDespulpado = vcursordespu.idDes;
374
375         IF NOT FOUND THEN
376             -- Si no existe, realizar la inserción
377             INSERT INTO Fermentacion (idDespulpado) VALUES (vcursordespu.idDes);
378         END IF;
379
380     END LOOP;
381
382     CLOSE curredespul;
383 END;
384 $$ LANGUAGE plpgsql;
```

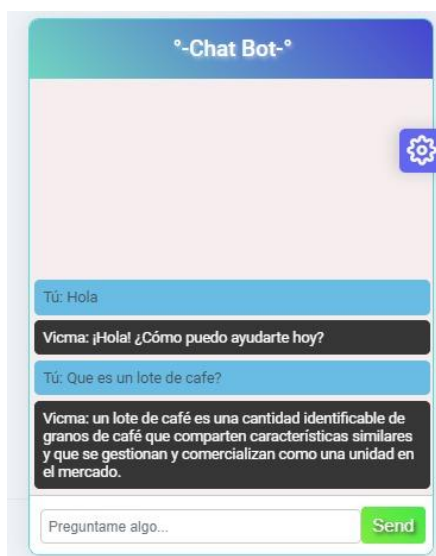
Anexo – 8

```
793
794 CREATE OR REPLACE VIEW TanquesCuartosDisponibles AS
795 SELECT
796     t.idTanque AS idRecurso,
797     t.statusT AS estadoRecurso,
798     NULL::INTEGER AS idCuarto,
799     NULL::VARCHAR(30) AS estadoCuarto
800 FROM
801     Tanques t
802 WHERE
803     t.statusT = 'DISPONIBLE'
804
805 UNION ALL
806
807 SELECT
808     NULL::INTEGER AS idRecurso,
809     NULL::VARCHAR(30) AS estadoRecurso,
810     c.idCuarto AS idCuarto,
811     c.estado AS estadoCuarto
812 FROM
813     Cuartos c
814 WHERE
815     c.estado = 'DISPONIBLE';
816
817
818 SELECT * FROM TanquesCuartosDisponibles;
819
```

Anexo – 9

```
src > controllers > JS controllers.js > ...
1 //conexion a la -BD
2 const { Pool } = require('pg');
3 const moment = require('moment');
4 const crypto = require('crypto');
5
6
7 const pool = new Pool({
8   host: 'database-cafe.ccwublznyky.us-east-1.rds.amazonaws.com',
9   user: 'root',
10  password: 'admin2023',
11  database: 'fincacafe',
12  port: '5432',
13  ssl: {
14    rejectUnauthorized: false, // En entornos de producción, deberías configurar esto correctamente
15  },
16 });
17
18 const client = pool.connect();
19 console.log("respuesta cliente: ", client);
20 //tabla terrenos
21
22 const getTerreno = async (req, res) => {
23   const response = await pool.query('SELECT * FROM terreno ORDER BY numterreno ASC');
24   res.status(200).json(response.rows);
25   console.log(response.rows);
26 };
27
28 const getTerrenoById = async (req, res) => {
29   const numterreno = req.params.id;
30   const response = await pool.query('SELECT * FROM terreno WHERE numterreno = $1', [numterreno]);
31   res.json(response.rows);
32 };
33
```

Anexo – 10



Anexo - 11


```
sakai-vue-master > JS server.js > ...
1  const express = require('express');
2  const http = require('http');
3  const socketIO = require('socket.io');
4
5  const app = express();
6  const server = http.createServer(app);
7  const io = socketIO(server);
8
9  io.on('connection', (socket) => {
10     console.log('Usuario conectado');
11
12     socket.on('disconnect', () => {
13         console.log('Usuario desconectado');
14     });
15
16     socket.on('message', (message) => {
17         console.log(`Mensaje recibido: ${message}`);
18         io.emit('message', message);
19     });
20 });
21
22 const PORT = process.env.PORT || 3000;
23 server.listen(PORT, () => {
24     console.log(`Servidor Socket.io escuchando en el puerto ${PORT}`);
25 });
26
```

Anexo - 12 (Crear servidor)

```

1  <div class="chatbox-container">
2    <div class="chatbox-wrapper">
3      <div class="container">
4        <div class="chatbot-name">
5          <h1>º-Chat Bot-º</h1>
6        </div>
7        <div class="messages-container" ref="messagesContainer">
8          <div class="messageBox" ref="messageBox">
9            <template v-for="(message, index) in messages" :key="index">
10              <div :class="message.from === 'user' ? 'messageFromUser' : 'messageFromChatGpt'">
11                <div :class="message.from === 'user' ? 'userMessageWrapper' : 'chatGptMessageWrapper'">
12                  <div :class="message.from === 'user' ? 'userMessageContent' : 'chatGptMessageContent'">
13                    {{ message.data }}
14                  </div>
15                </div>
16              </div>
17            </template>
18          </div>
19        </div>
20        <div class="inputContainer">
21          <input v-model="currentMessage" type="text" class="messageInput" placeholder="Preguntame algo..." />
22          <button @click="$event => sendMessage(currentMessage)" class="askButton">Send</button>
23        </div>
24      </div>
25    </div>
26  </div>
27 </template>

```

Anexo - 13 (Vista del chat)

```

<script>
import io from 'socket.io-client';

export default {
  data() {
    return {
      socket: null,
      messages: [],
      currentMessage: '',
      maxMessages: 70, // cantidad máxima de mensajes que se desea mostrar
      predefinedResponses: {
        'Hola': '¡Hola! ¿Cómo puedo ayudarte hoy?',
        'hola': '¡Hola! ¿Cómo puedo ayudarte hoy?',
        'holaa': '¡Hola! ¿Cómo puedo ayudarte hoy?',
        'ola': '¡Hola! ¿Cómo puedo ayudarte hoy?',
        '¿Cómo estás?': 'Estoy bien, gracias. ¿Y tú?',

        'Que es un lote de cafe?': 'un lote de café es una cantidad identificable de granos de café que comparten características similares',
        'Cuál es la definición de un lote de café?': 'un lote de café es una cantidad identificable de granos de café que comparten características similares',
        'Explicame qué significa un lote de café.': 'un lote de café es una cantidad identificable de granos de café que comparten características similares',
        'Define el concepto de lote de café.': 'un lote de café es una cantidad identificable de granos de café que comparten características similares',
        'Puedes describir lo que implica un lote de café?': 'un lote de café es una cantidad identificable de granos de café que comparten características similares',
        'A qué nos referimos con la expresión "lote de café"?': 'un lote de café es una cantidad identificable de granos de café que comparten características similares',
        'Explica el término "lote de café"': 'un lote de café es una cantidad identificable de granos de café que comparten características similares',

        'Cuales son los procesos de los lotes de cafe?': 'Los procesos de los lotes de cafe son: Recolección, Despulpado, \n Fermentación',

        'Cual es el primer proceso de los lotes del cafe?': 'El primer proceso de los lotes de cafe es: Recolección.',

        'Cual es el segundo proceso de los lotes del cafe?': 'El segundo proceso de los lotes de cafe es: Despulpado.',

        'Cual es el tercer proceso de los lotes del cafe?': 'El tercer proceso de los lotes de cafe es: Fermentación'.
      }
    }
  }
}

```

```

'Cuales son los procesos de los lotes de cafe?': 'Los procesos de los lotes de cafe son: Recolección, Despulpado, \n Fermentación',

'Cual es el primer proceso de los lotes del cafe?': 'El primer proceso de los lotes de cafe es: Recolección.',

'Cual es el segundo proceso de los lotes del cafe?': 'El segundo proceso de los lotes de cafe es: Despulpado.',

'Cual es el tercer proceso de los lotes del cafe?': 'El tercer proceso de los lotes de cafe es: Fermentación',

'Cual es el cuarto proceso de los lotes del cafe?': 'El cuarto proceso de los lotes de cafe es la Lavado',

'Cual es el quinto proceso de los lotes del cafe?': 'El quinto proceso de los lotes de cafe es la Secado',

'Donde se fermenta los lotes del cafe?': 'El cafe se fermenta en tanques de fermentación',
'Donde se fermenta los lotes del cafe?': 'El cafe se fermenta en tanques de fermentación',
'Donde se fermenta los lotes del cafe?': 'El cafe se fermenta en tanques de fermentación',
'Donde se fermenta los lotes del cafe?': 'El cafe se fermenta en tanques de fermentación',
'Donde se fermenta los lotes del cafe?': 'El cafe se fermenta en tanques de fermentación',
'Donde se fermenta los lotes del cafe?': 'El cafe se fermenta en tanques de fermentación',

'Donde se secan los lotes el cafe?': 'Los lotes de cafe se secan en cuartos de secado.',
'Cuantos tipos de cafe existen?': 'Existen dos tipos de cafe, los cuales son: Café arábica, Café robusta.',
'Cual es el proposito de este sistema?': 'El control de los procesos de los lotes de cafe, desde su recolección \n hasta el lavado',
'Cual es el tipo de cafe mas utilizado en México?': 'El tipo de cafe mas utilizado en México es.',
'Cual es tu nombre?': 'No tengo nombre, pro puedes llamarme Vicma',
'Como te llamas?': 'No tengo nombre, pro puedes llamarme Vicma',
'Tienes': 'No tengo nombre, pro puedes llamarme Vicma',

'': '¡Estas mandando un mensaje vacío!, porfavor escribe una pregunta.',
'Adios': '¡Hasta luego!, que tengas un gran día.',
'adios': '¡Hasta luego!, que tengas un gran día.',
'bay': '¡Hasta luego!, que tengas un gran día.',
'hasta luego': '¡Hasta luego!, que tengas un gran día.',
'bye': '¡Hasta luego!, que tengas un gran día.',

```

```

    'Cual es el quinto proceso de los lotes del cafe?': 'El quinto proceso de los lotes de cafe es la Secado',
    'Donde se fermenta los lotes del cafe?': 'El cafe se fermenta en tanques de fermentación',
    'Donde se fermenta los lotes del cafe?': 'El cafe se fermenta en tanques de fermentación',
    'Donde se fermenta los lotes del cafe?': 'El cafe se fermenta en tanques de fermentación',
    'Donde se fermenta los lotes del cafe?': 'El cafe se fermenta en tanques de fermentación',
    'Donde se fermenta los lotes del cafe?': 'El cafe se fermenta en tanques de fermentación',
    'Donde se secan los lotes el cafe?': 'Los lotes de cafe se secan en cuartos de secado.',
    'Cuantos tipos de cafe existen?': 'Existen dos tipos de cafe, los cuales son: Café arábica, Café robusta.',
    'Cual es el proposito de este sistema?': 'El control de los procesos de los lotes de cafe, desde su recolección \n hasta el lavado',
    'Cual es el tipo de cafe mas utilizado en México?': 'El tipo de cafe mas utilizado en México es.',
    'Cual es tu nombre?': 'No tengo nombre, pro puedes llamarme Vicma',
    'Como te llamas?': 'No tengo nombre, pro puedes llamarme Vicma',
    'Tienes': 'No tengo nombre, pro puedes llamarme Vicma',

    '': '¡Estas mandando un mensaje vacío!, porfavor escribe una pregunta.',
    'Adios': '¡Hasta luego!, que tengas un gran día.',
    'adios': '¡Hasta luego!, que tengas un gran día.',
    'bay': '¡Hasta luego!, que tengas un gran día.',
    'hasta luego': '¡Hasta luego!, que tengas un gran día.',
    'bye': '¡Hasta luego!, que tengas un gran día.',
    'nos vemos': '¡Hasta luego!, que tengas un gran día.',
    'chao': '¡Hasta luego!, que tengas un gran día.',
  },
  defaultMessage: 'No entendí la pregunta. ¿Puedes reformularla?'
},
},

```

Anexo – 14 (socket cliente y configuración de preguntas respuestas)

```

mounted() {
  this.socket = io('http://localhost:3000');
  this.socket.on('message', (message) => {
    this.handleMessage(message);
    this.scrollToBottom();
  });
},
methods: {
  sendMessage(message) { //mandamos la pregunta al chat
    this.messages.push({ from: 'user', data: `Tú: ${message}` });
    this.socket.emit('message', message);
    this.handleMessage(message);
    this.currentMessage = '';
    this.scrollToBottom();
  },
}

```

Anexo – 15 (mandar pregunta al chat por un socket)


```

handleMessage(message) { //el chat manda la respuesta en un socket
  const response = this.predefinedResponses[message] || this.defaultMessage;
  this.messages.push({ from: 'chatGpt', data: `Vicma: ${response}` });
  this.socket.emit('message', `Chatbot: ${response}`);

  // Limitar la cantidad de mensajes
  if (this.messages.length > this.maxMessages) {
    this.messages.shift();
  }
},
scrollToBottom() {
  this.$nextTick(() => {
    const container = this.$refs.messagesContainer;
    container.scrollTop = container.scrollHeight;
  });
},
};

```

Anexo – 16 (chat contesta por medio de un socket al usuario)

```

<style scoped>
@import url('https://fonts.googleapis.com/css2?family=Roboto:wght@400;500&display=swap');

@import url('https://fonts.googleapis.com/css2?family=Roboto:wght@400;500&display=swap');

1 reference
.chatbot-name {
  text-align: center;
  padding: 15px;
  background: linear-gradient(45deg, #71d6c2, #4643d1);
  /* Efecto de color neón */
}

1 reference
.chatbot-name h1 {
  margin: 0;
  font-size: 1.5rem;
  color: white;
  text-shadow: 2px 2px 4px rgba(255, 255, 255, 0.5);
  /* Sombra de texto */
}

1 reference
.chatbox-container {
  position: fixed;
  bottom: 30px;
  right: 30px;
  z-index: 100;
}

1 reference
.chatbox-wrapper {
  border: 1.5px solid rgb(80, 219, 215);

```

```

1 reference
.inputContainer {
  display: flex;
  align-items: flex-end;
  /* Posiciona el contenido al final del eje vertical */
}

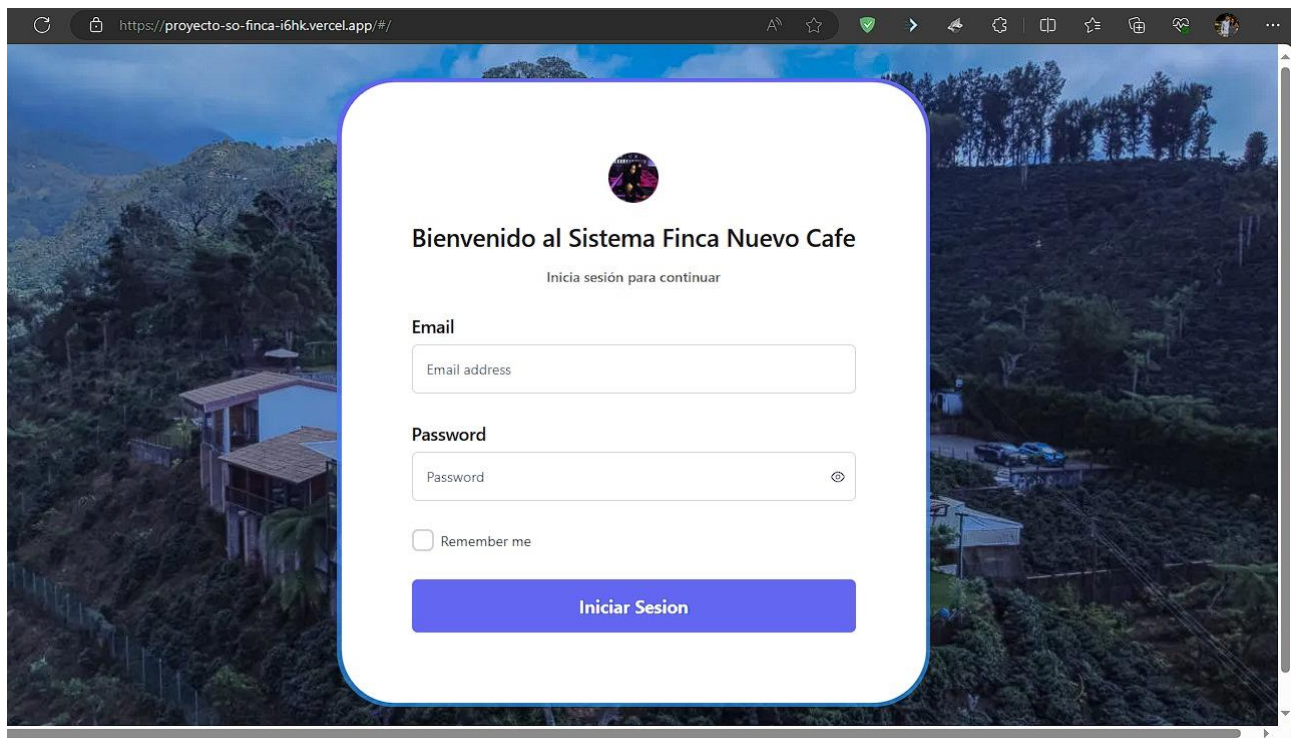
1 reference
.messageInput {
  flex: 1;
  padding: 7px;
  border: 1px solid #ccc;
  border-radius: 5px;
}

1 reference
.askButton {
  font-size: 1.2rem;
  text-shadow: 2px 2px 4px rgba(36, 31, 31, 0.911);
  padding: 8px 10px;
  background: linear-gradient(45deg, #80eb42, #47e947);
  color: #fff;
  border: none;
  border-radius: 4px;
  cursor: pointer;
}

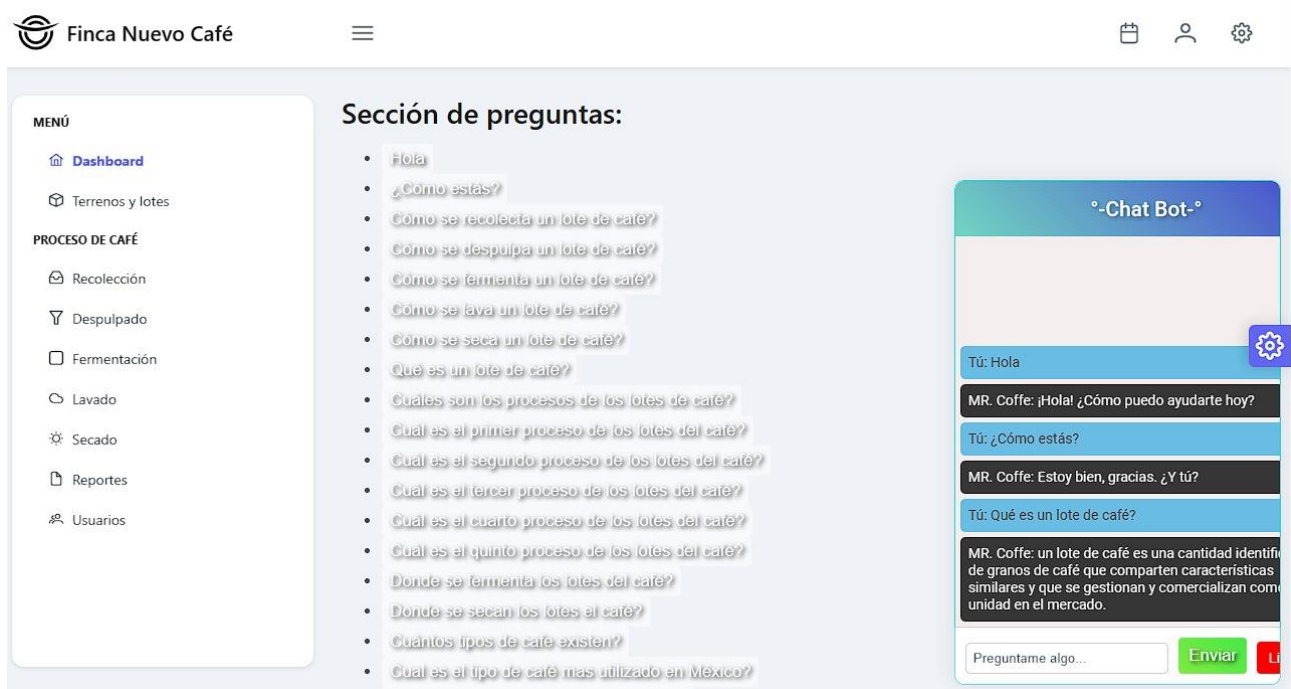
1 reference
.askButton:hover {
  background: linear-gradient(45deg, #22ddb8, #0066ff);
}
</style>

```

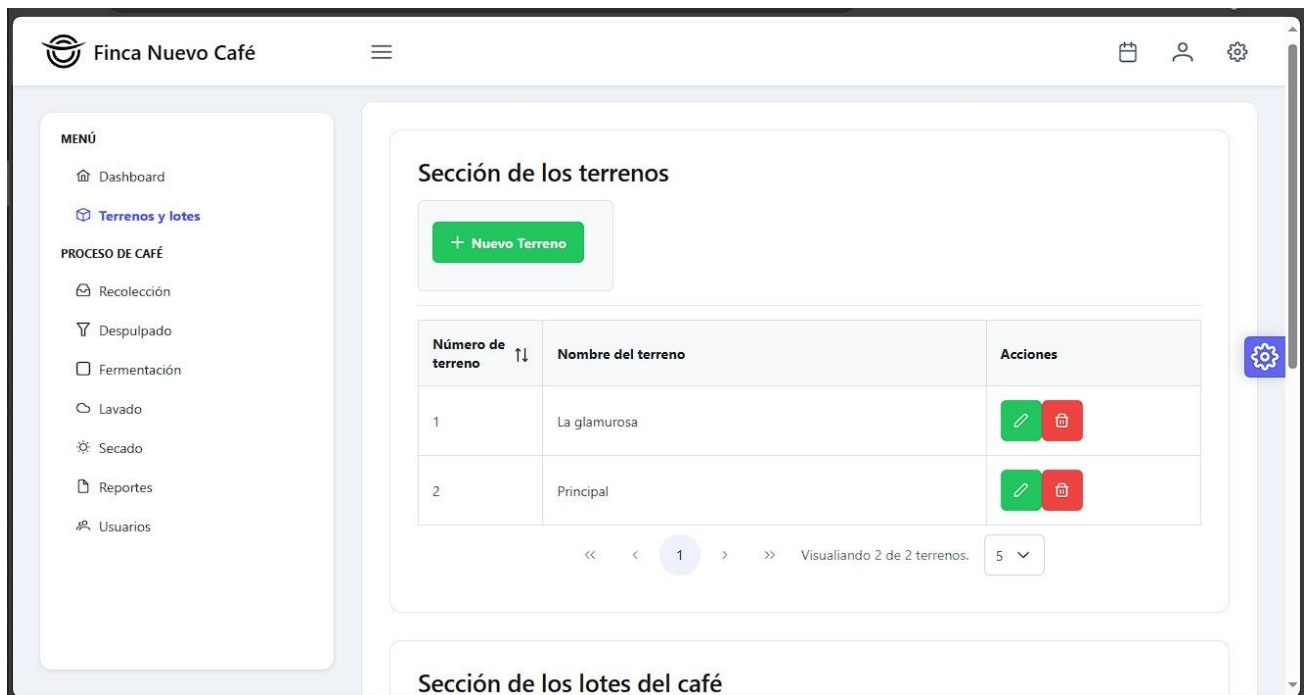
Anexo – 17 (Estilos para el chat)



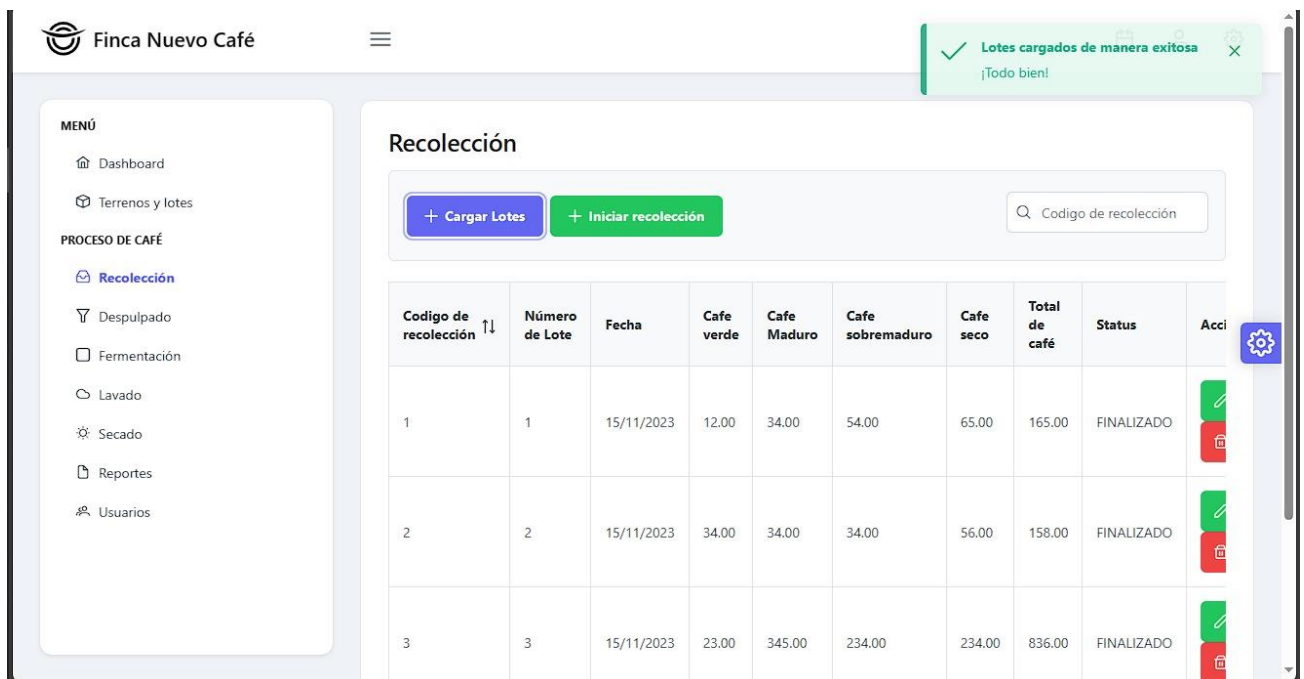
Anexo – 18



Anexo – 19



Anexo – 20



Anexo – 21

Finca Nuevo Café

Lotes disponibles para despulpar

¡Todo bien!

MENÚ

Dashboard

Terrenos y lotes

PROCESO DE CAFÉ

Recolección

Despulpado

Fermentación

Lavado

Secado

Reportes

Usuarios

Despulpado

+ Cargar Lotes

+ Ver lotes a despulpar

Q

Codigo de despulpado

Codigo de despulpado	Codigo de recolección	Fecha	Hora	Status	Acciones
1	1	15/11/2023	08:15:18	FINALIZADO	<div>+ </div> <div> </div>
3	2			PENDIENTE	<div>+ </div> <div> </div>
4	3			PENDIENTE	<div>+ </div> <div> </div>
2	1	15/11/2023	09:45:51	FINALIZADO	<div>+ </div> <div> </div>
5	1			PENDIENTE	<div>+ </div> <div> </div>

Anexo – 22

Finca Nuevo Café

MENÚ

Dashboard

Terrenos y lotes

PROCESO DE CAFÉ

Recolección

Despulpado

Fermentación

Lavado

Secado

Reportes

Usuarios

Sección de tanques

+ Agregar Tanque

Q

Codigo de fermentación

Codigo del tanque	Cantidad de café	Status	Acciones
1		DISPONIBLE	<div> </div> <div> </div>

<<

<

1

>

>>

Visualizando 1 de 1 tanque(s).

10

Sección de Fermentacion

+ Cargar Lotes

+ Listar lotes a fermentar

Q

Codigo de fermentación

Finca Nuevo Café

MENÚ

- Dashboard
- Terrenos y lotes
- PROCESO DE CAFÉ
- Recolección
- Despulpado
- Fermentación**
- Lavado
- Secado
- Reportes
- Usuarios

Sección de Fermentacion

+ Cargar Lotes + Listar lotes a fermentar

Código de fermentación

Codigo de fermentación	Codigo de despulpado	Codigo de tanque	Tem. minima	Tem. maxima	Fecha Inicial	Hora Inicial	Fecha Final
1	1	1	33	13	2023-11-15T06:00:00.000Z	08:16:17	2023-11-15T06:00:00.000Z
2	2	1	30	20	2023-11-15T06:00:00.000Z	09:46:35	2023-11-15T06:00:00.000Z

Anexo – 23

Finca Nuevo Café

Lotes disponibles para lavar
¡Todo bien!





MENÚ

- Dashboard
- Terrenos y lotes
- PROCESO DE CAFÉ
- Recolección
- Despulpado
- Fermentación
- Lavado**
- Secado
- Reportes
- Usuarios

Lavado

+ Cargar Lotes a Lavar + Ver lotes a Lavar

Código de lavado

Codigo de Lavado	Codigo de Fermentacion	Kg a Lavar	Fecha	Hora	Status	Acciones
1	1	165.00	2023-11-15T06:00:00.000Z	08:16:39	FINALIZADO	 
2	2	165.00	2023-11-15T06:00:00.000Z	09:48:19	FINALIZADO	 

by Finca "Nuevo Café"

Anexo – 24

Finca Nuevo Café

MENÚ

Dashboard

Terrenos y lotes

PROCESO DE CAFÉ

Recolección

Despulpado

Fermentación

Lavado

Secado

Reportes

Usuarios

Sección de Cuartos de Secado

+ Agregar Un Nuevo Cuarto

Q

Codigo de cuarto

Codigo del Cuarto ↑↓	Cantidad de Cafe	Status	Acciones
1		DISPONIBLE	<div><div></div><div></div></div>
2		DISPONIBLE	<div><div></div><div></div></div>
3		DISPONIBLE	<div><div></div><div></div></div>

<<

<

1

>

>>

Visualiando 3 de 3 Cuartos.

10

▼

Finca Nuevo Café

MENÚ

Dashboard

Terrenos y lotes

PROCESO DE CAFÉ

Recolección

Despulpado

Fermentación

Lavado

Secado

Reportes

Usuarios

Sección de Secado

+ Cargar Lotes a Secar

+ Listar lotes a Secar

Q

Codigo de Secado

Codigo de Secado ↑↓	Codigo de Lavado	Codigo de Cuarto	Tipo de Secado	Kg. Secados	Fecha Inicial	Hora Inicial	Fecha Final	Ho Fin
1	1	1	Maquina	165.00	2023-11-15T06:00:00.000Z	08:17:09	2023-11-15T06:00:00.000Z	08:
2	2	1	Maquina	165.00	2023-11-15T06:00:00.000Z	09:49:42	2023-11-15T06:00:00.000Z	09:

<<

<

1

>

>>

Visualiando 2 de 2 lotes a secar.

10

▼

MENÚ

- 🏠 Dashboard
- 📁 Terrenos y lotes

PROCESO DE CAFÉ

- 📁 Recolección
- 🔍 Despulpado
- 📁 Fermentación
- 📁 Lavado
- ⚙️ Secado
- 📄 Reportes
- 👤 Usuarios

PRIMER REPORTE

Mostrar el tiempo que dura el proceso de un determinado lote de café, desde la recolección hasta el secado.

2 ▾

Consultar

Exportar Excel

Número de lote ↑↓	Fecha de recolección	Fecha de despulpado	Fecha inicio fermentacion	Fecha fin de fermentacion	Fecha inicio de secado	Fecha fin de secado
2	2023-11-15T06:00:00.000Z					
2	2023-11-15T06:00:00.000Z					

<< < 1 > >>

Visualizando 2 de 2 registros.

5 ▾



Anexo – 26

SEGUNDO REPORTE

Mostrar los datos de la fase en la que se encuentra cada lote de café.

Exportar Excel

Número de lote ↑↓	Estado de la recolección	Estado del despulpado	Estado del lavado	Estado del secado
1	FINALIZADO	PENDIENTE	FINALIZADO	FINALIZADO
2	FINALIZADO	PENDIENTE	FINALIZADO	FINALIZADO
3	FINALIZADO	PENDIENTE		

<< < 1 > >> Visualizando 3 de 3 registros.

5 ▾

Anexo – 27

TERCER REPORTE

Mostrar los tanques y cuarto disponibles en cualquier momento.

Exportar Excel

ID del tanque ↑↓	Estado	ID del cuarto	Estado
1	DISPONIBLE		
		1	DISPONIBLE
		2	DISPONIBLE
		3	DISPONIBLE

<< < 1 > >> Visualiando 4 de 4 registros. 5 ▾

Anexo – 28

☰

📅 👤 ⚙️

CUARTO REPORTE

Mostrar cual es el mes en el que má se ha recolectado más cantidad de café.

Exportar Excel

Mes ↑↓	Cantidad recolectada
noviembre	1159.00

<< < 1 > >> Visualiando 1 de 1 registros. 5 ▾

Anexo – 29

QUINTO REPORTE

Mostrar cual es el tipo de café que se ha recolectado más en un año (verde, maduro, sobremaduro y seco).

Exportar Excel

Año ↑↓	Tipo de café	Veces recolectado	Cantidad recolectada
2023	MADURO	3	413.00

<< < 1 > >> Visualiando 1 de 1 registros.

5 ▾

Anexo – 30

SEXTO REPORTE

Mostrar el promedio del tiempo al mes del secado de los lotes de café.

Exportar Excel

Mes ↑↓	Promedio Tiempo de Secado
November	00.00

<< < 1 > >> Visualiando 1 de 1 registros.

5 ▾

Anexo – 31

Conclusión

Por lo expuesto anteriormente, comprendemos que desarrollamos la problemática de nuestro problema planteado, dando a conocer la resolución del mismo, cuidando aquellos datos que se mencionaban en el contexto además de necesarios para realizar las consultas requeridas.

Gracias a este producto se abordaron diferentes lenguajes de programación, cada uno tiene su propia importancia, aunque uno difiere del otro en totalidad, pueden ser enlazados para trabajar de manera cooperativa, gracias a esto podemos aplicar de forma más fácil estos conocimientos abordados en algún proyecto futuro. En nuestro caso, abordamos el chat Bot que realiza acciones conforme al usuario. Así como crear tus propias Apis para que las consultas tú mismo y aplicar en el script del vue.js.

Se considera que, gracias a este proyecto, se comprendieron nuevos conceptos de programación en SQL, socket.io para implementar y utilizarla, además de conocer nuevos componentes de primevue.js o socket.io para implementar sockets con tal librería.

Bibliografías

Desconocido. (n.d.-a). *Empezando*. Empezando | Axios Docs. <https://axios-http.com/es/docs/intro>

Desconocido. (n.d.-b). *HTML: Lenguaje de etiquetas de hipertexto: MDN*. MDN Web Docs. <https://developer.mozilla.org/es/docs/Web/HTML>

Desconocido. (n.d.-c). *¿Qué es el CSS? - aprende desarrollo web: MDN*. MDN Web Docs. https://developer.mozilla.org/es/docs/Learn/CSS/First_steps/What_is_CSS

Desconocido. (n.d.-d). *¿Qué es javascript? - aprende desarrollo web: MDN*. MDN Web Docs. https://developer.mozilla.org/es/docs/Learn/JavaScript/First_steps/What_is_JavaScript

Desconocido. (n.d.-e). *Vue*. Prime Vue: Empezar con un proyecto desde cero_vue.js_weixin_0010034-Vue. <https://devpress.csdn.net/vue/62f0cf467e66823466183293.html#:~:text=Prime%20Vue%20es%20una%20librería,con%20una%20documentación%20bastante%20buena.>

Hernandez, M. (2021, February 8). *¿Qué es npm?*. freeCodeCamp.org. <https://www.freecodecamp.org/espanol/news/que-es-npm/>

Oracle. (n.d.). *¿Qué es el json?. ¿Qué es JSON?* | Oracle México. <https://www.oracle.com/mx/database/what-is-json/>

Sim, C. (2021, July 27). *¿Qué es Node.js, y para qué sirve?*. Blog ITDO - Agencia de desarrollo Web, APPs y Marketing en Barcelona. <https://www.itdo.com/blog/que-es-node-js-y-para-que-sirve/>

Vite. (n.d.). *Vite*. <https://es.vitejs.dev/guide/>

Vue. (n.d.). *Introducción - Vue.js*. - Vue.js. <https://es.vuejs.org/v2/guide/>

Desconocido. (2023, October 9). *PostgreSQL*. Wikipedia.

<https://es.wikipedia.org/wiki/PostgreSQL>

Desconocido. (n.d.-a). *Programación de sockets*. Programación de Sockets.

<https://www.ibm.com/docs/es/i/7.5?topic=communications-socket-programming>

Desconocido. (n.d.-b). *Socket.io*. SocketIO RSS. <https://socket.io/>

Desconocido. (n.d.-c). *Vercel: Desarrollar, Previsualizar, enviar*. Aplyca Tecnología

SAS. <https://www.aplyca.com/blog/blog-que-es-vercel-desarrollar-previsualizar-enviar>

Emaz, A., & Pérez-Ruiz, U. (1998a). *AWS*. Amazon. [https://aws.amazon.com/es/what-is/cross-origin-resource-](https://aws.amazon.com/es/what-is/cross-origin-resource-sharing/#:~:text=CORS%20permite%20que%20el%20navegador,realizar%20cu)

[sharing/#:~:text=CORS%20permite%20que%20el%20navegador,realizar%20cu](https://aws.amazon.com/es/what-is/cross-origin-resource-sharing/#:~:text=CORS%20permite%20que%20el%20navegador,realizar%20cu)
[alquier%20transferencia%20de%20datos.](https://aws.amazon.com/es/what-is/cross-origin-resource-sharing/#:~:text=CORS%20permite%20que%20el%20navegador,realizar%20cu)

Emaz, A., & Pérez-Ruiz, U. (1998b). *que es SQL*. Amazon.

<https://aws.amazon.com/es/what-is/sql/>

IBM. (n.d.). *¿Qué es una api rest?* <https://www.ibm.com/mx-es/topics/rest-apis>

WilliamDAssafMSFT. (n.d.-a). *Procedimientos Almacenados (motor de base de datos) -*

SQL server. SQL Server | Microsoft Learn. [https://learn.microsoft.com/es-](https://learn.microsoft.com/es-es/sql/relational-databases/stored-procedures/stored-procedures-database-engine?view=sql-server-ver16)
[es/sql/relational-databases/stored-procedures/stored-procedures-database-](https://learn.microsoft.com/es-es/sql/relational-databases/stored-procedures/stored-procedures-database-engine?view=sql-server-ver16)
[engine?view=sql-server-ver16](https://learn.microsoft.com/es-es/sql/relational-databases/stored-procedures/stored-procedures-database-engine?view=sql-server-ver16)

WilliamDAssafMSFT. (n.d.-b). *Vistas - SQL Server*. SQL Server | Microsoft Learn.

[https://learn.microsoft.com/es-es/sql/relational-databases/views/views?view=sql-](https://learn.microsoft.com/es-es/sql/relational-databases/views/views?view=sql-server-ver16)
[server-ver16](https://learn.microsoft.com/es-es/sql/relational-databases/views/views?view=sql-server-ver16)