Documento de evidencias de capacitación para kuantik desktop

Por: Luis Alonso Andrade López

kuantik desktop

kuantik data jump sapi s.a de C. V

Realizado el 05 de diciembre del 2025

Contenido

[Introducción 2](#_Toc209773424)

[Ejercicio 1: Navegación de páginas 5](#_Toc209773425)

[Ejercicio 2: Conexión hacia una base de datos relacional 8](#_Toc209773426)

[Ejercicio 3: Consumo de API con SOAP 16](#_Toc209773427)

[Ejercicio 4: Consumo de API con REST 20](#_Toc209773428)

[Ejercicio 5: Creación de instalador 24](#_Toc209773429)

[Ejercicio 6: Conexión hacia una base de datos no relacional 28](#_Toc209773430)

[Ejercicio 7: Utilización de una base de datos integrada relacional (SQLite), y almacenamiento del archivo de la base de datos en la carpeta de datos predeterminada de las aplicaciones de escritorio 35](#_Toc209773431)

[Ejercicio 8: Utilización de una base de datos integrada no relacional (bbolt), y almacenamiento del archivo de la base de datos en la carpeta de datos predeterminada de las aplicaciones de escritorio 39](#_Toc209773432)

[Ejercicio 9: Implementación de concurrencia con GoRoutines en un apartado de descargas múltiples 45](#_Toc209773433)

# Introducción

La empresa Kuantik Data Jump SAPI C.A de C.V decidió realizar un sistema de escritorio llamado “Kuantik Desktop”, el cual facilitaría a los clientes la descarga de facturas, además la administración de las facturas descargadas, y enviaría las facturas a Kuantik Data Jump. El desarrollo de dicho sistema fue llevado a cabo por el practicante Luis Alonso Andrade López.

Para poder crear el sistema “Kuantik Desktop”, el practicante tuvo que aprender algunos conceptos necesarios para el desarrollo de dicho sistema.

Los conceptos necesarios fueron:

* Las tecnologías de desarrollo seleccionadas (Wails, Go, Vue.js)
* APIs SOAP
* APIs REST
* Formato XML especial para el SAT (CFDI).
* Conceptos de creación de instaladores para aplicaciones de escritorio

Una vez investigados dichos conceptos, se decidió crear un mini sistema de entrenamiento llamado “WailsTraining4KDJ”, el cual, en base a algunos ejercicios propuestos, el practicante crearía ventanas en donde se aplicarían los conceptos investigados.

Los ejercicios necesarios fueron:

* Navegación entre páginas
* Conexión hacia una base de datos
* Consumo de API con SOAP
* Consumo de API con REST
* Creación de instalador

Antes de realizar los ejercicios necesarios, lo primero que se tuvo que hacer fue instalar las tecnologías aprendidas y requeridas:

Texto

El contenido generado por IA puede ser incorrecto.

Ilustración : Verificación de instalación de Wails

Texto

El contenido generado por IA puede ser incorrecto.

Ilustración : Verificación de instalación de Go

Interfaz de usuario gráfica

El contenido generado por IA puede ser incorrecto.

Ilustración : Verificación de instalación de Node.js

Una vez instaladas correctamente las tecnologías, se procedió a crear un proyecto de Wails por defecto para poder crear el sistema:

Texto

El contenido generado por IA puede ser incorrecto.

Ilustración : Creación de un proyecto por defecto de Wails que ocupe Vue.js

Captura de pantalla de computadora

El contenido generado por IA puede ser incorrecto.

Ilustración : Verificación de la creación del proyecto por defecto

Una vez realizados estos pasos, ya se prosiguió con la realización de los ejercicios requeridos. En este documento se mostrarán los pasos que se llevaron a cabo para la realización de dichos ejercicios.

# Ejercicio 1: Navegación de páginas

El primer ejercicio consistió en crear formas para navegar entre pantallas en “WailsTraining4KDJ”. Para esto, se decidió crear un menú principal de navegación, y una pestaña de navegación en la parte superior de todas las ventanas del sistema:

Texto

El contenido generado por IA puede ser incorrecto.

Ilustración : Código del menú principal de navegación

Imagen que contiene Patrón de fondo

El contenido generado por IA puede ser incorrecto.

Ilustración : Código que permite la navegación de ventanas



Ilustración : Código de la pestaña navegadora de ventanas

Una captura de pantalla de un celular

El contenido generado por IA puede ser incorrecto.

Ilustración : Menú principal

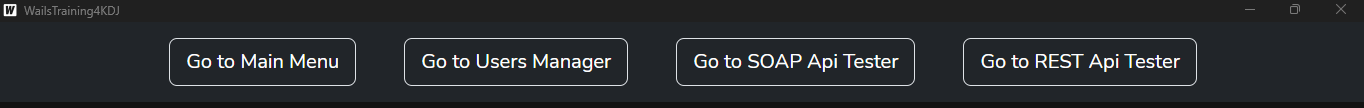


Ilustración : Pestaña navegadora de ventanas

Captura de pantalla de computadora

El contenido generado por IA puede ser incorrecto.

Ilustración : Uso de la pestaña navegadora de ventanas en una ventana del sistema

# Ejercicio 2: Conexión hacia una base de datos relacional

El ejercicio 2 consistió en realizar una conexión entre “WailsTraining4KDJ” y una base de datos, para de esta manera, poder hacer un CRUD sencillo con dicho sistema.

El SGBD usado es MySQL, una tecnología en la que el practicante ya tenía experiencia, así que solamente se mostrará el script de sql para crear lo necesario:

CREATE DATABASE WailsTraining4KDJ;

USE WailsTraining4KDJ;

CREATE TABLE users (

id INT NOT NULL AUTO\_INCREMENT,

name VARCHAR(30),

surnames VARCHAR(30),

birth\_date DATE,

phone\_number VARCHAR(15),

email VARCHAR(100),

PRIMARY KEY (id)

);

Una vez creado el script, se prosiguió con la creación del código del backend de “WailsTraining4KDJ” para realizar la conexión a la base de datos y crear métodos para realizar el CRUD:

Texto

El contenido generado por IA puede ser incorrecto.

Ilustración : Código para crear conexiones a la base de datos

Texto

El contenido generado por IA puede ser incorrecto.

Ilustración : Código del modelo usado para el transporte de datos

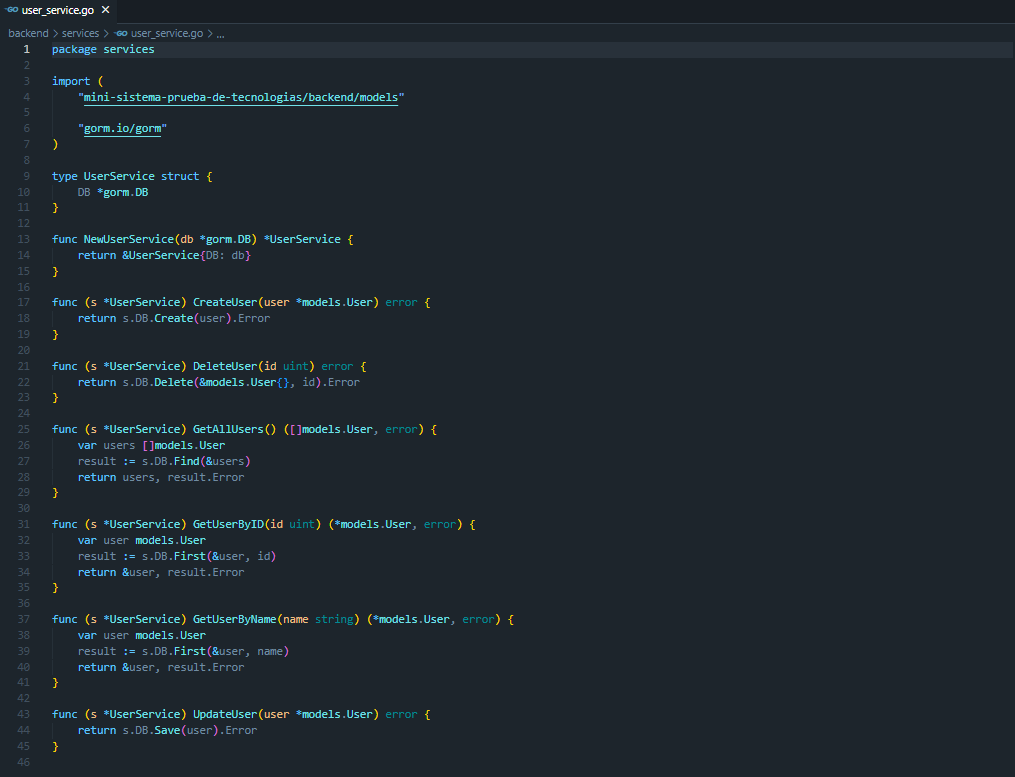


Ilustración : Código de los métodos para hacer CRUD en la base de datos

Después, se realizó una ventana en el frontend con Vue.js, para poder utilizar los métodos de CRUD creados en el backend y realizar CRUD con ellos mediante una interfaz gráfica, junto con unos filtros de búsqueda auxiliares para una mejor administración de los registros de la base de datos:

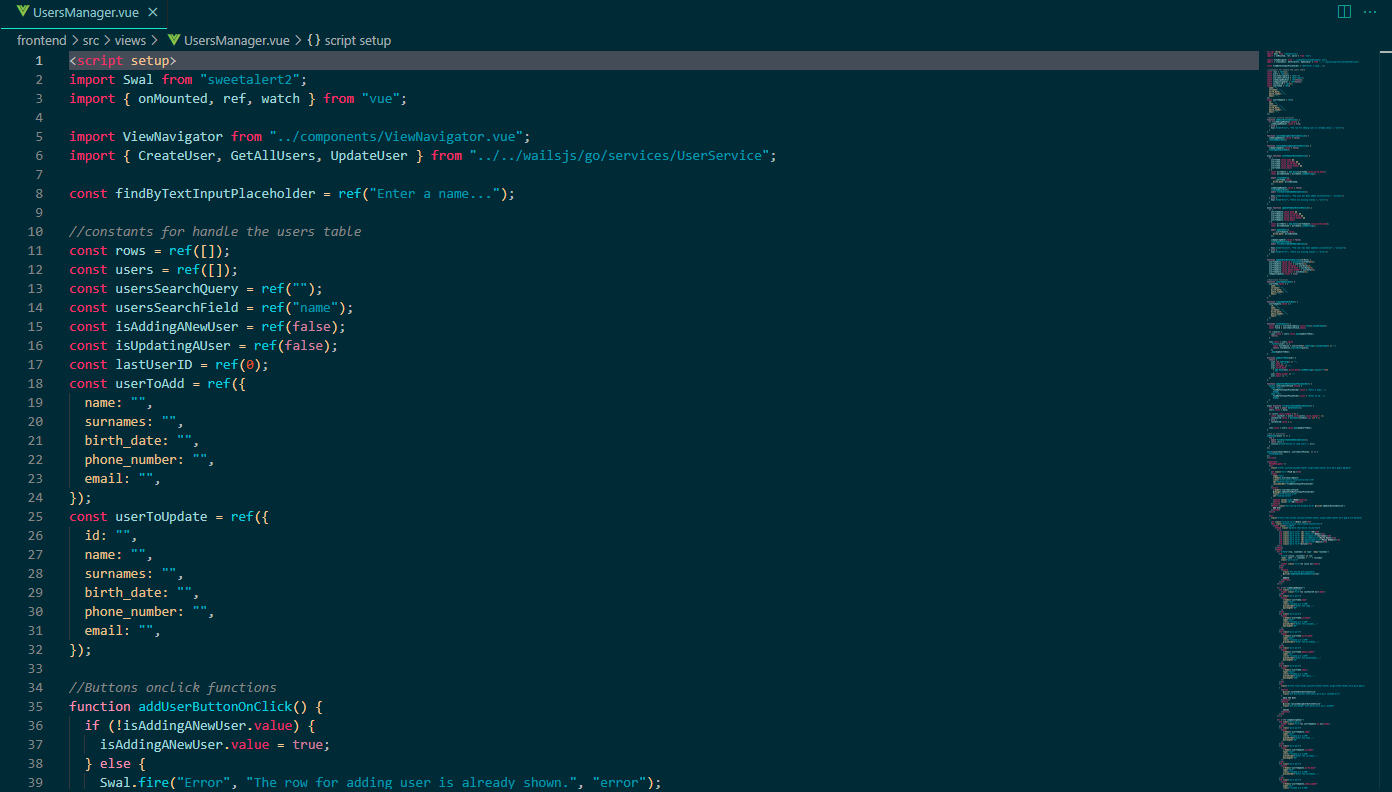


Ilustración : Un extracto del código en Vue.js para la realización de CRUD

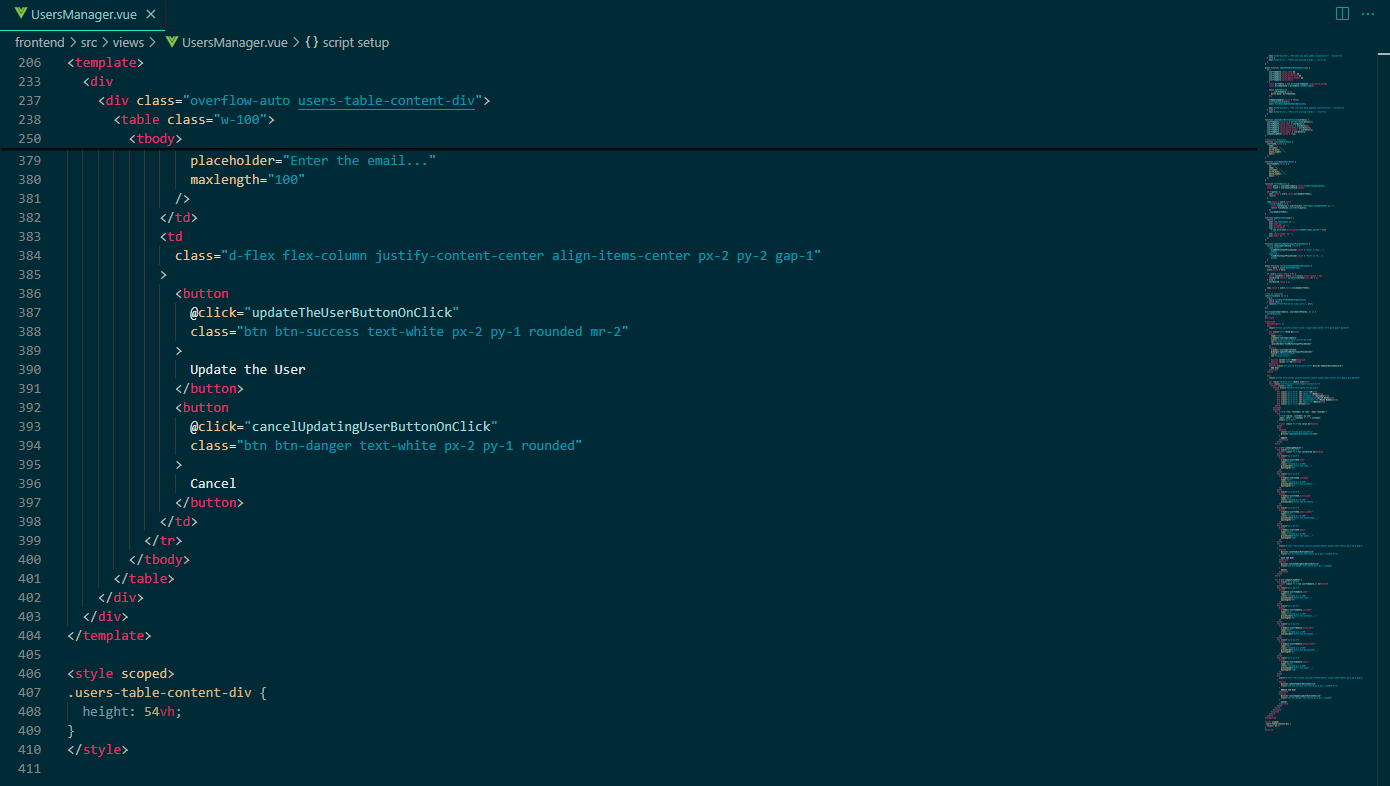


Ilustración : Otro extracto del código en Vue.js para la realización de CRUD

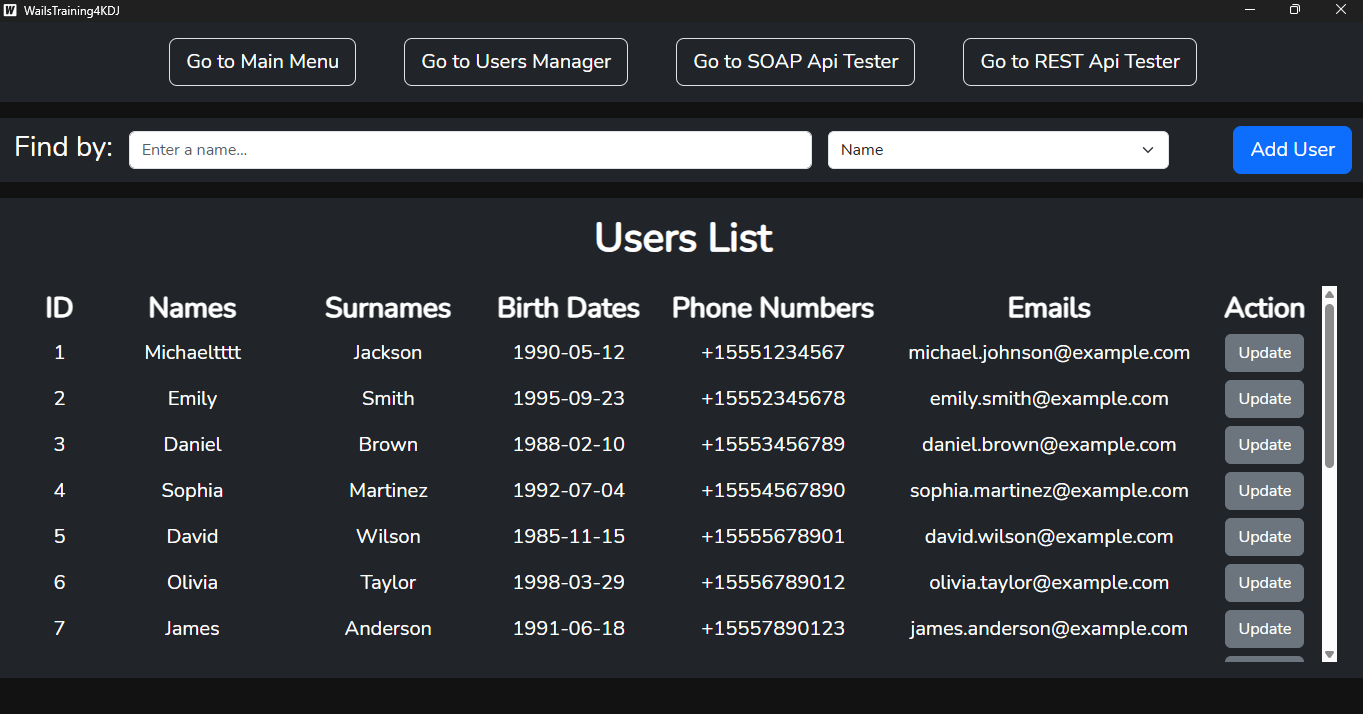


Ilustración : Ventana de CRUD de usuarios

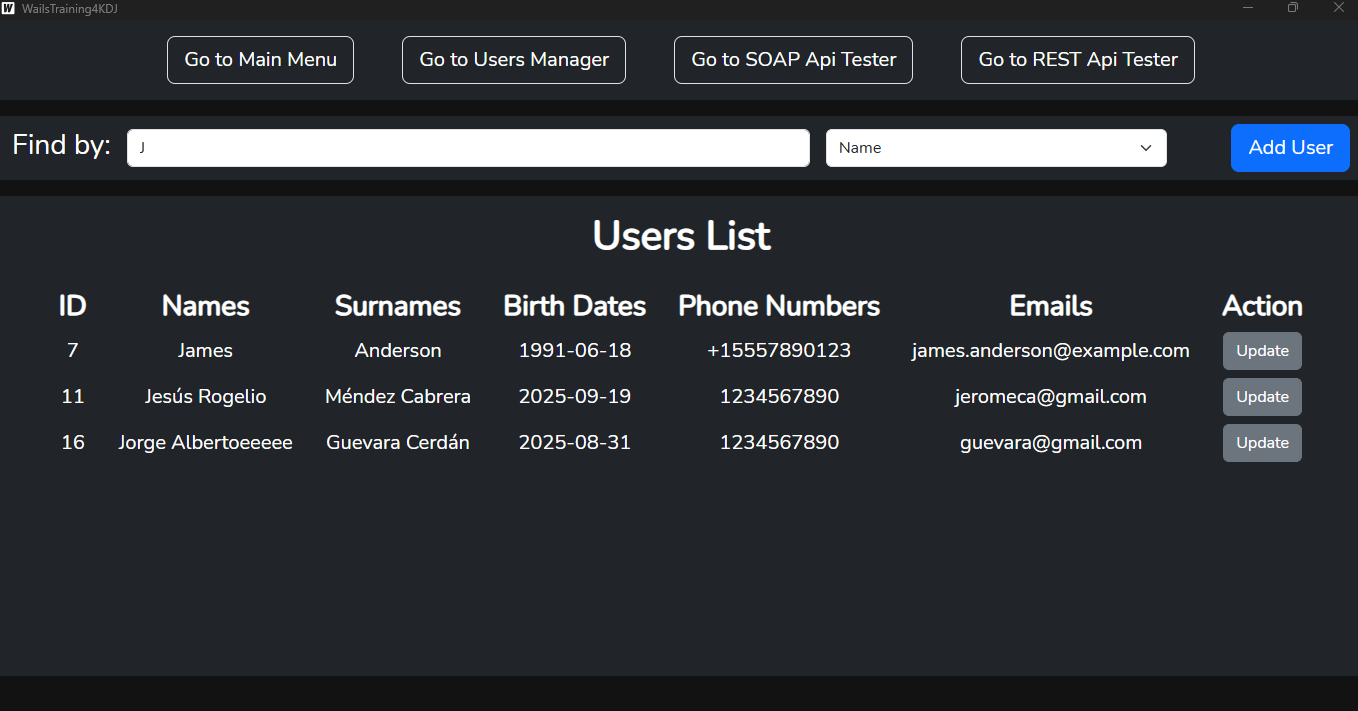


Ilustración : Comprobación del funcionamiento de los filtros de búsqueda

Captura de pantalla de un celular

El contenido generado por IA puede ser incorrecto.

Ilustración : Creación de usuario

Interfaz de usuario gráfica, Sitio web

El contenido generado por IA puede ser incorrecto.

Ilustración : Comprobación de usuario creado

Captura de pantalla de computadora

El contenido generado por IA puede ser incorrecto.

Ilustración : Modificar usuario

Interfaz de usuario gráfica, Sitio web

El contenido generado por IA puede ser incorrecto.

Ilustración : Comprobación de usuario modificado

Interfaz de usuario gráfica, Texto, Aplicación

El contenido generado por IA puede ser incorrecto.

Ilustración : Comprobación de existencia de base de datos

# Ejercicio 3: Consumo de API con SOAP

El ejercicio 3 consistió en hacer que “WailsTraining4KDJ” consumiera una API que utilizara el protocolo SOAP, y hacer algunas operaciones que la API ofreciera. En este caso, se decidió hacer una ventana de conversión de números utilizando la API pública: <https://www.dataaccess.com/webservicesserver/numberconversion.wso>

Primero lo que se tuvo que hacer, es crear un archivo en el backend para poder conectarse a la API y realizar algunas operaciones que la API permite hacer:

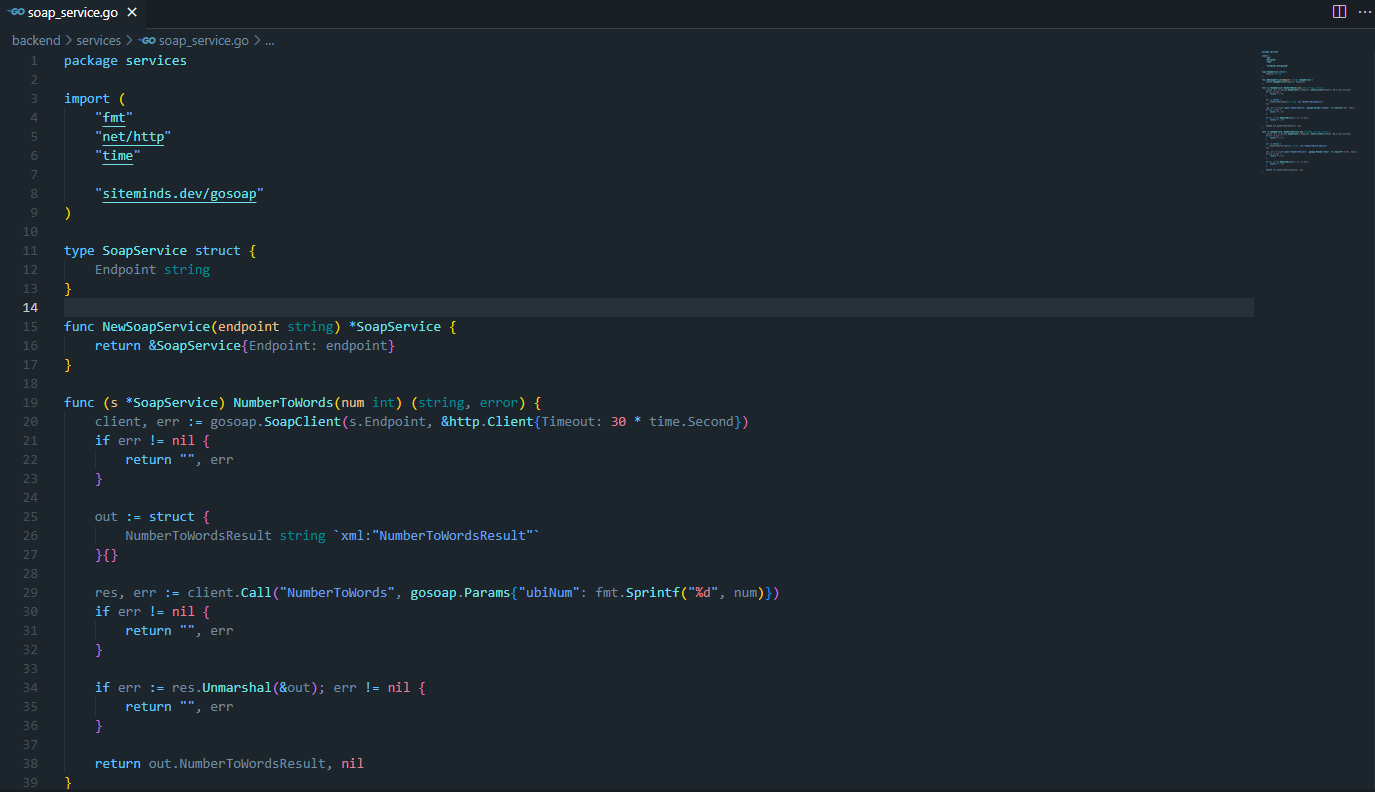


Ilustración : Un fragmento del código que permite conectarse y realizar operaciones con la API

Texto

El contenido generado por IA puede ser incorrecto.

Ilustración : Otro fragmento del código que permite conectarse y realizar operaciones con la API

Texto

El contenido generado por IA puede ser incorrecto.

Ilustración : Fragmento del código para referenciar la URL de la API

Después, para poder mostrar el funcionamiento de la API de mejor manera, se creó una ventana en el frontend con Vue.js en donde se utilizan los métodos del backend que consumen la API:

Texto

El contenido generado por IA puede ser incorrecto.

Ilustración : Un fragmento del código de la interfaz gráfica que utiliza los métodos del backend consumidores de la API

Texto

El contenido generado por IA puede ser incorrecto.

Ilustración : Otro fragmento del código de la interfaz gráfica que utiliza los métodos del backend consumidores de la API

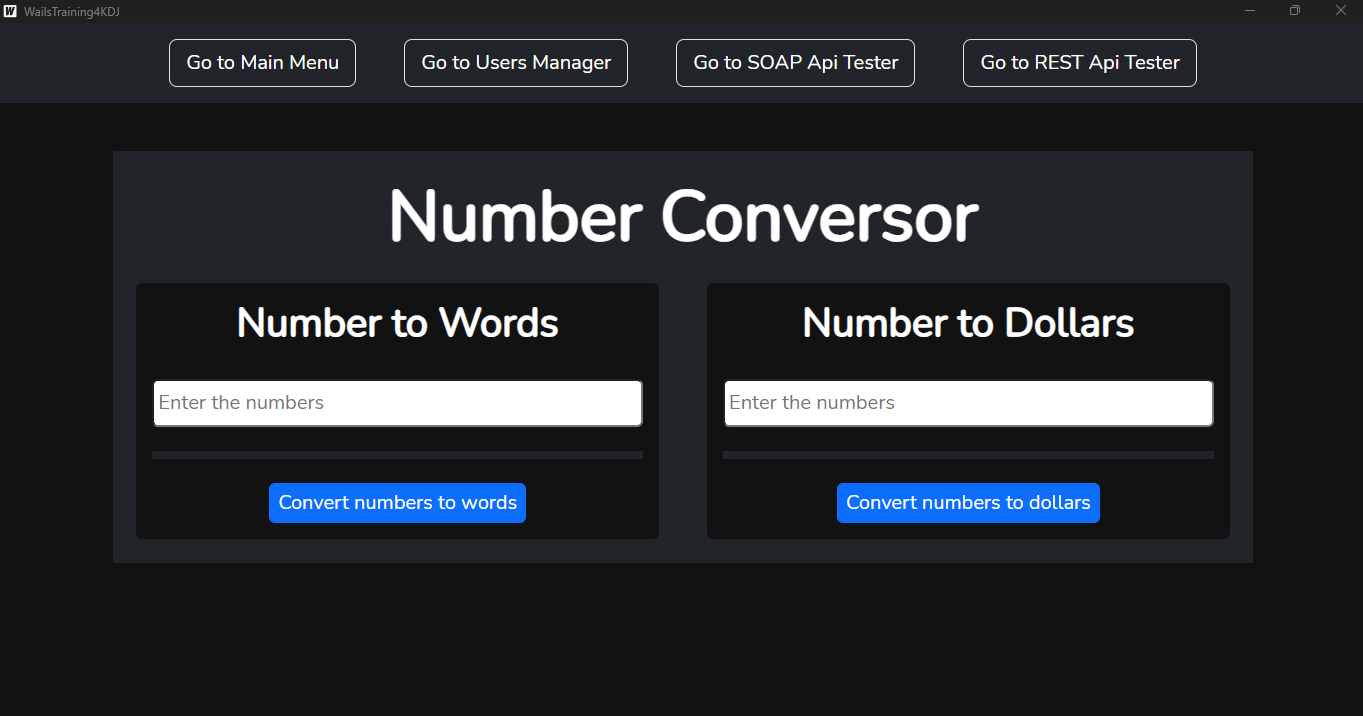


Ilustración : Ventana que utiliza los métodos del backend consumidores de la API

Interfaz de usuario gráfica, Aplicación, Sitio web

El contenido generado por IA puede ser incorrecto.

Ilustración : Verificación del funcionamiento del método para convertir números a palabras (método de la API)

Interfaz de usuario gráfica, Sitio web

El contenido generado por IA puede ser incorrecto.

Ilustración : Verificación del funcionamiento del método para convertir números a dólares (método de la API)

# Ejercicio 4: Consumo de API con REST

Este ejercicio consistió en hacer algo similar que el ejercicio 3, pero esta vez con una API REST (la utilizada es esta: <https://jsonplaceholder.typicode.com>). Primero, lo que se hizo fue crear en el backend un archivo que hiciera uso de los 5 métodos HTTP (GET, POST, PUT, PATCH, DELETE) con la API:

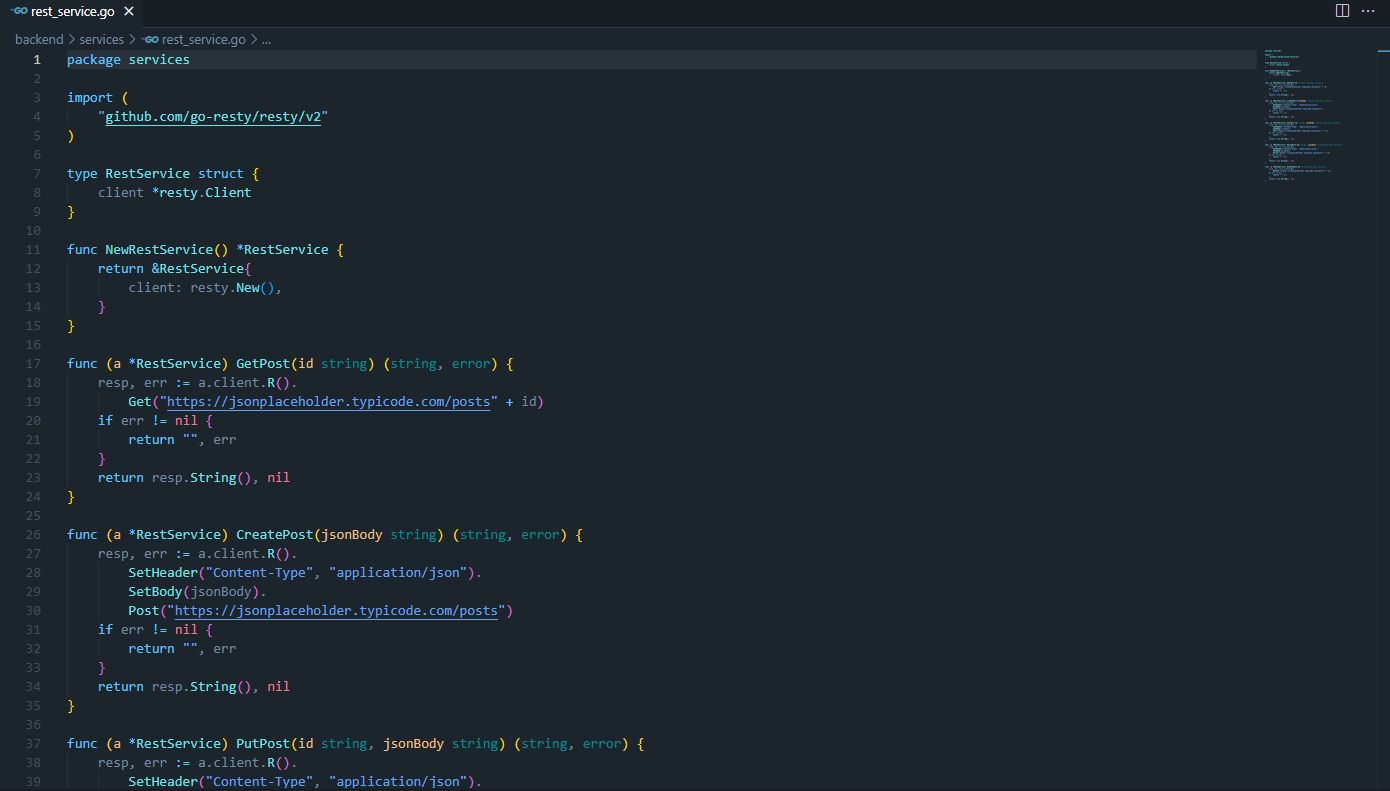


Ilustración : Un fragmento del código para consumir el API REST

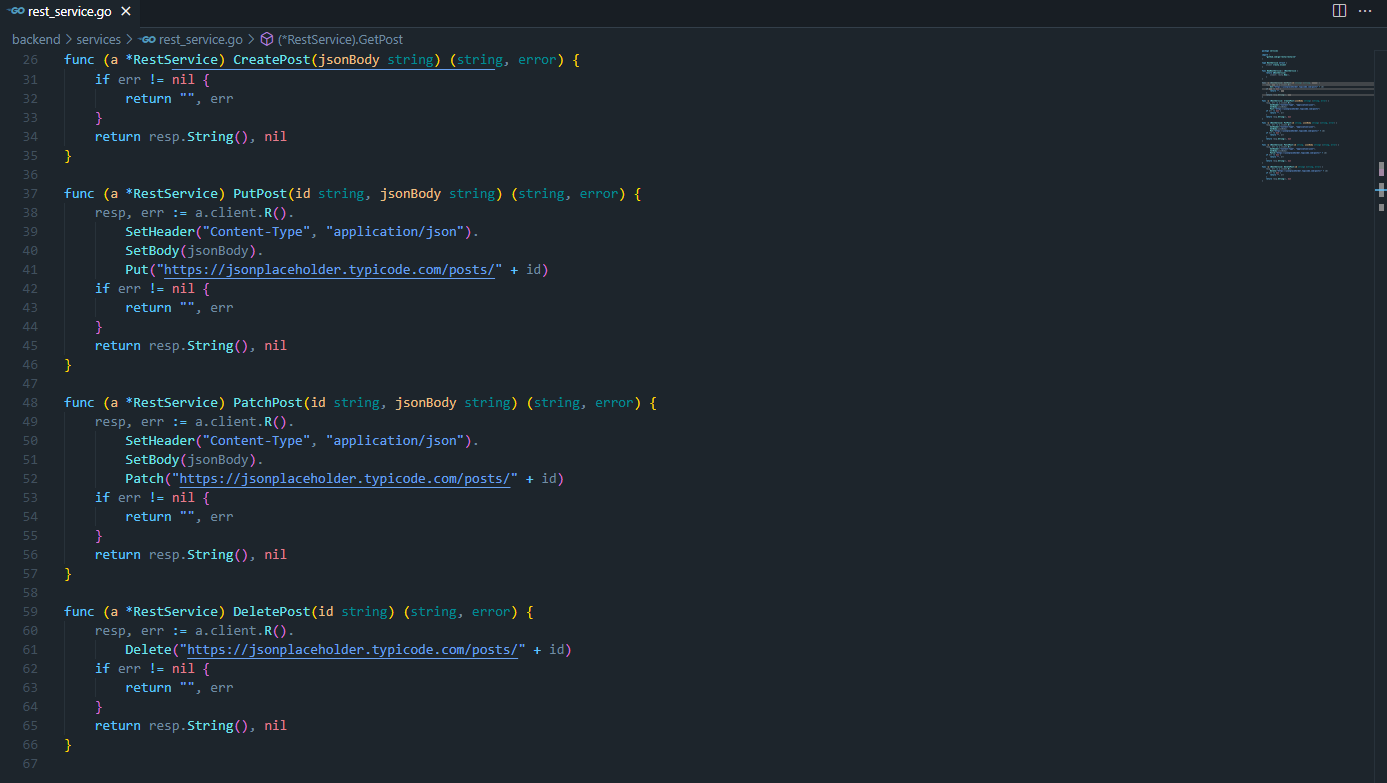


Ilustración : Otro fragmento del código para consumir el API REST

Una vez hecho esto, se tuvo que crear la ventana con Vue.js en el frontend para ejecutar los métodos del backend que hacen uso de la API:



Ilustración : Un fragmento del código de la ventana que utiliza los métodos del backend consumidores del API REST

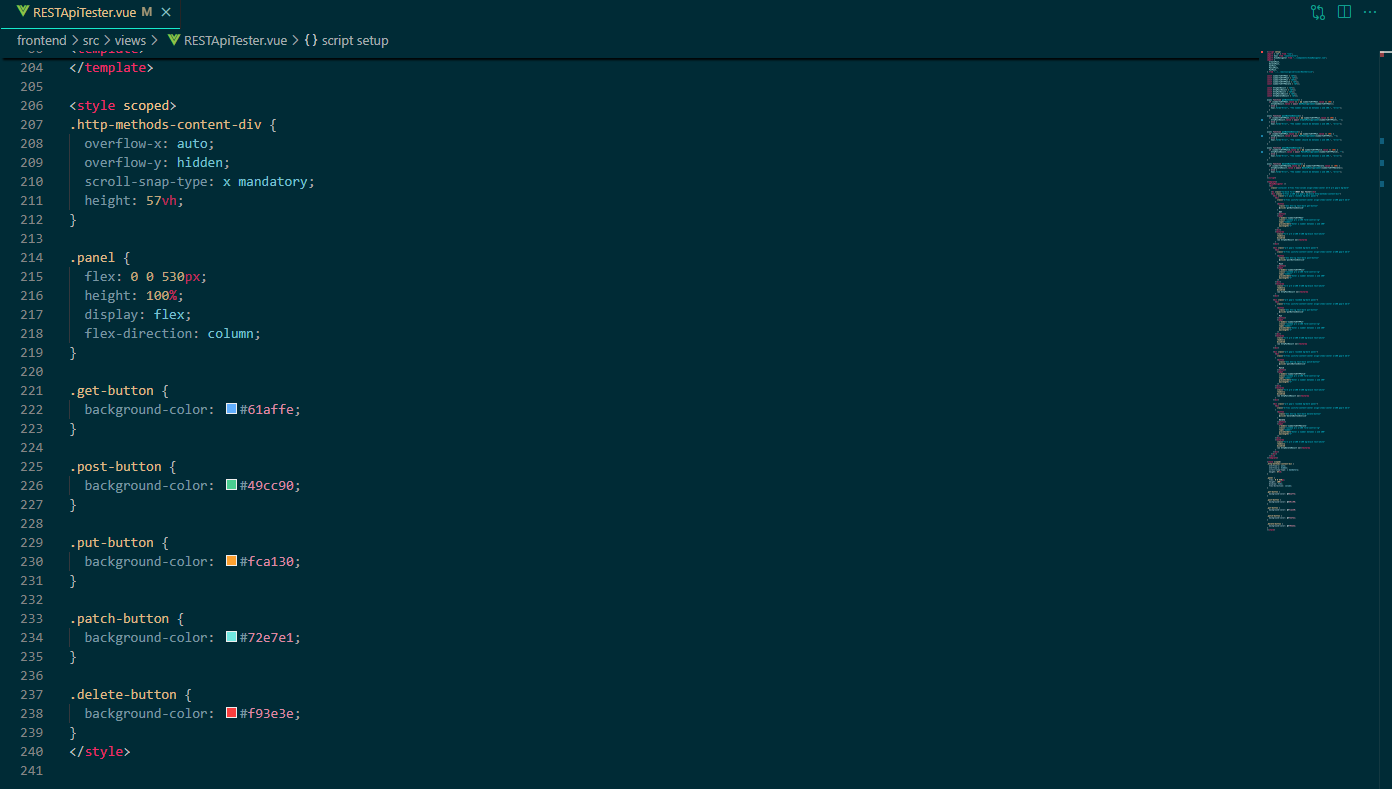


Ilustración : Otro fragmento del código de la ventana que utiliza los métodos del backend consumidores del API REST

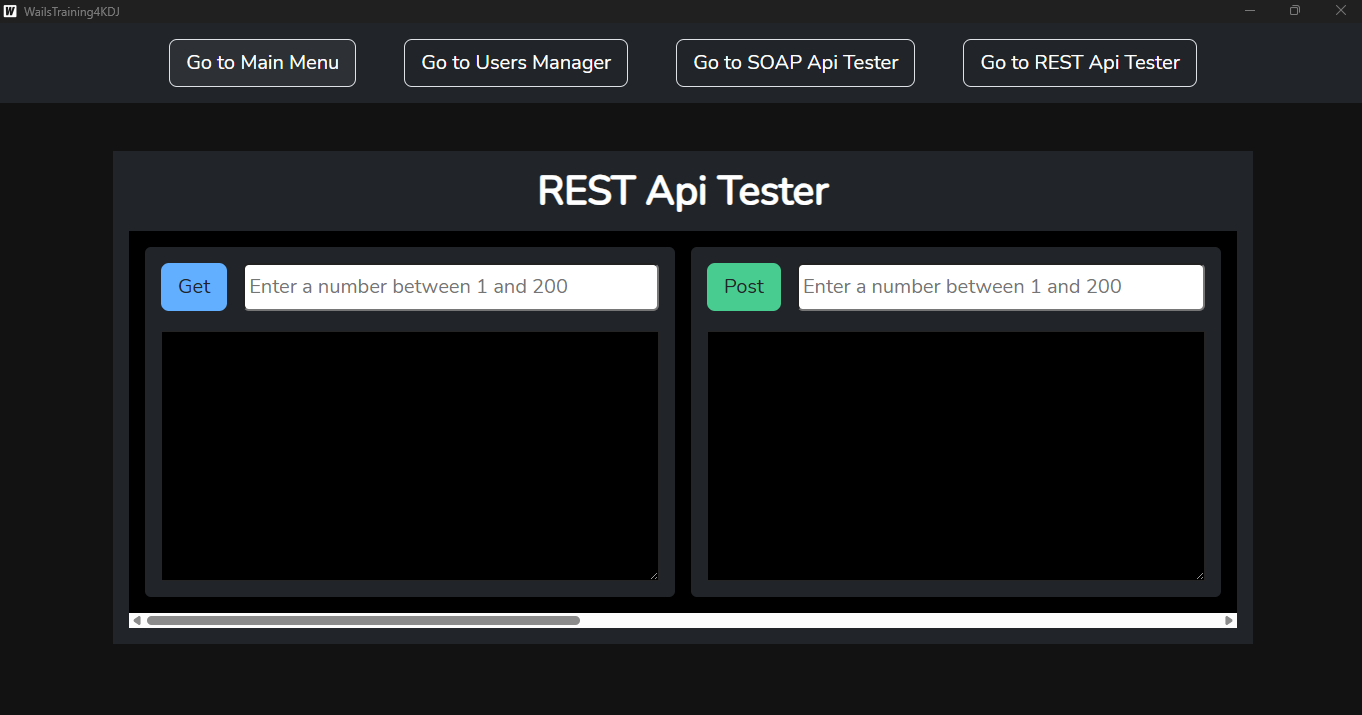


Ilustración : Ventana para probar los métodos

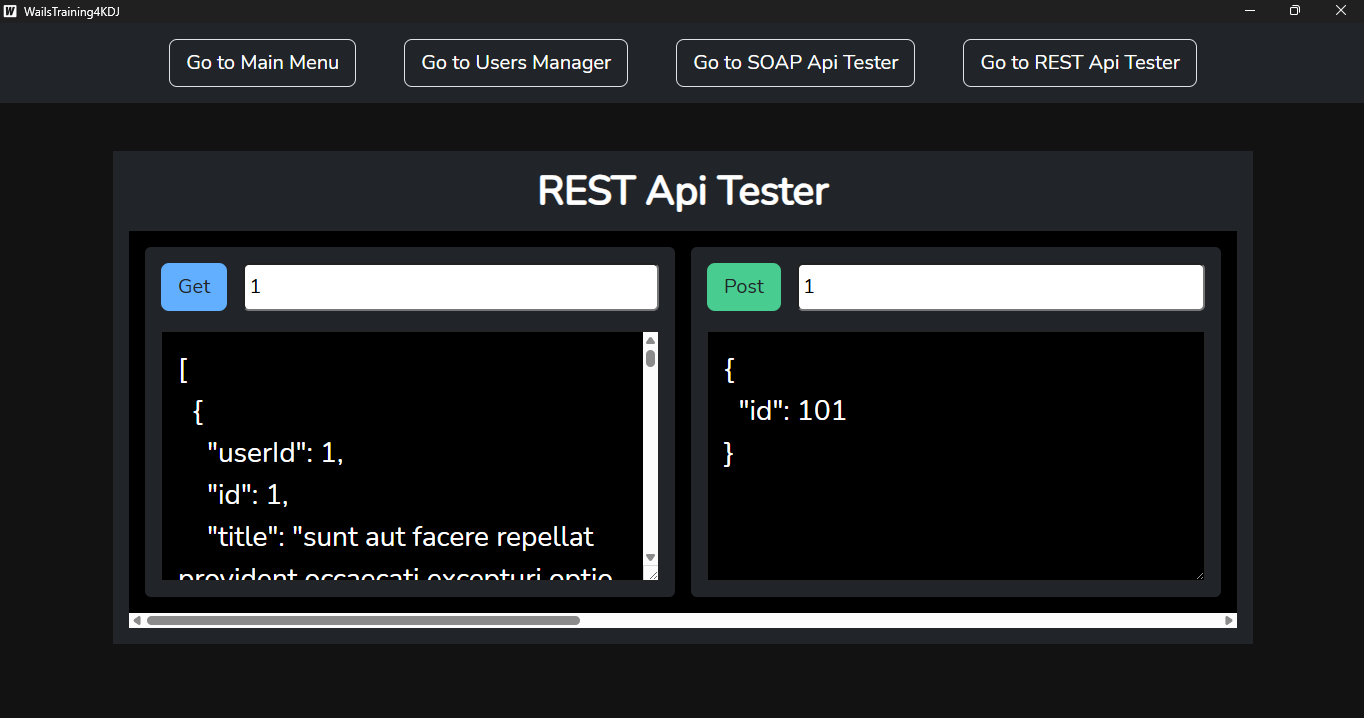


Ilustración : Verificación del funcionamiento de los métodos GET y POST

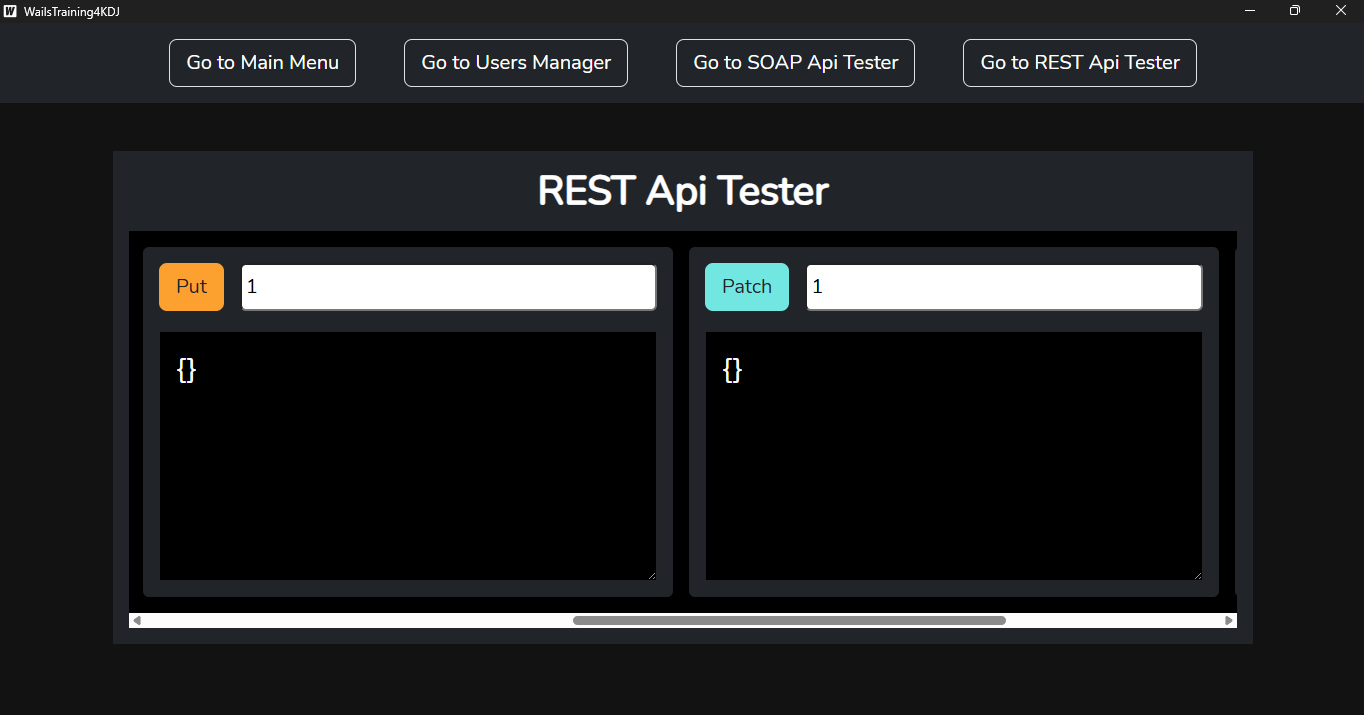


Ilustración : Verificación del funcionamiento de los métodos PUT y PATCH

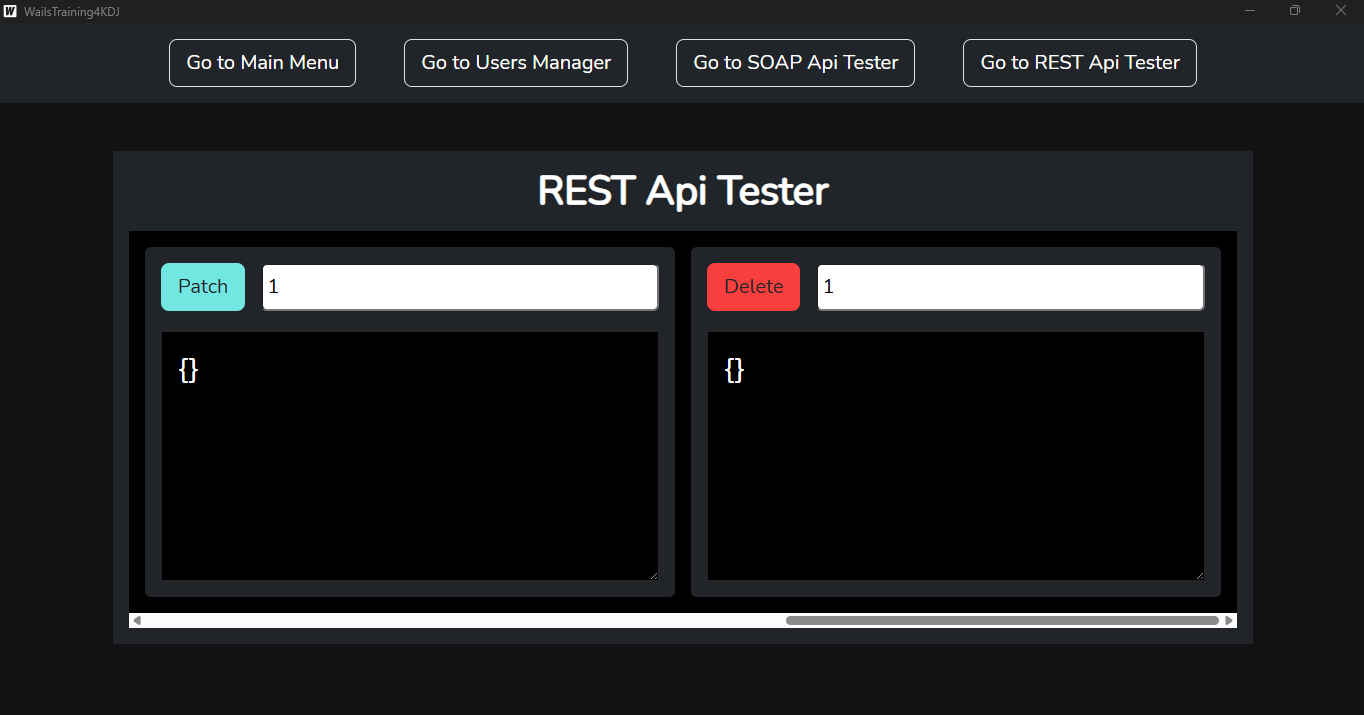


Ilustración : Verificación del método DELETE

# Ejercicio 5: Creación de instalador

Este ejercicio consistió en la realización de un instalador tradicional para aplicaciones de escritorio, el cual iba a contener las siguientes ventanas:

* Ventana de inicio
* Ventana para mostrar el eula
* Ventana para cambiar la dirección de instalación de la aplicación (por defecto está en “C:\Program Files”)
* Ventana para mostrar el proceso de instalación
* Ventana de éxito o fracaso en la instalación

Wails ya tiene funcionalidades que permiten crear un instalador de la aplicación que se esté desarrollando, solamente hay que modificar el archivo “/build/Windows/installer/project.nsi/” a conveniencia para agregar/modificar/eliminar funcionalidades deseadas al instalador.

Texto

El contenido generado por IA puede ser incorrecto.

Ilustración : Extracto del archivo "project.nsi"

Este archivo fue modificado para agregar al instalador las ventanas requeridas. A continuación, se mostrarán las ventanas requeridas del instalador:

Interfaz de usuario gráfica, Aplicación, Word

El contenido generado por IA puede ser incorrecto.

Ilustración : Ventana de inicio del instalador

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

El contenido generado por IA puede ser incorrecto.

Ilustración : Ventana para mostrar el eula en el instalador

Interfaz de usuario gráfica, Texto, Aplicación, Chat o mensaje de texto

El contenido generado por IA puede ser incorrecto.

Ilustración : Ventana de directorio de instalación y ventana del cambio de directorio

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

El contenido generado por IA puede ser incorrecto.

Ilustración : Ventana del instalador sobre el proceso de instalación de la aplicación

Interfaz de usuario gráfica, Aplicación, Word

El contenido generado por IA puede ser incorrecto.

Ilustración : Ventana del instalador sobre el éxito o fracaso de la instalación de la aplicación

# Ejercicio 6: Conexión hacia una base de datos no relacional

Este ejercicio consistió en realizar algo similar al ejercicio 2, pero utilizando una base de datos no relacional, en este caso MongoDB, y el CRUD sería de libros en lugar de usuarios.

Los elementos que contendría cada libro serían los siguientes:

{

"title": String,

"author": String,

"published\_at": Date,

"pages": Int32

}

Y las funcionalidades de la ventana de administración de libros serían los mismos que la de administración de usuarios. A continuación, se mostrarán extractos de código del frontend y backend para el funcionamiento de la ventana, y capturas de la ventana funcionando:

Texto

El contenido generado por IA puede ser incorrecto.

Ilustración : Código para crear conexiones a la base de datos

Texto

El contenido generado por IA puede ser incorrecto.

Ilustración : Código del modelo usado para el transporte de datos

Texto

El contenido generado por IA puede ser incorrecto.

Ilustración : Un extracto del código de los métodos para hacer CRUD en la base de datos

Texto

El contenido generado por IA puede ser incorrecto.

Ilustración : Otro extracto del código de los métodos para hacer CRUD en la base de datos

Texto

El contenido generado por IA puede ser incorrecto.

Ilustración : Un extracto del código de la ventana de CRUD de libros

Imagen que contiene Texto

El contenido generado por IA puede ser incorrecto.

Ilustración : Otro extracto del código de la ventana de CRUD de libros

Interfaz de usuario gráfica

El contenido generado por IA puede ser incorrecto.

Ilustración : Ventana de CRUD de libros

Interfaz de usuario gráfica, Aplicación

El contenido generado por IA puede ser incorrecto.

Ilustración : Comprobación de filtros de búsqueda

Captura de pantalla de un celular

El contenido generado por IA puede ser incorrecto.

Ilustración : Modificar libro

Interfaz de usuario gráfica, Aplicación, Sitio web

El contenido generado por IA puede ser incorrecto.

Ilustración : Comprobación de modificación de libro

Captura de pantalla de un celular

El contenido generado por IA puede ser incorrecto.

Ilustración : Agregar libro

Interfaz de usuario gráfica, Aplicación, Sitio web

El contenido generado por IA puede ser incorrecto.

Ilustración : Comprobación de agregación de libro

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

El contenido generado por IA puede ser incorrecto.

Ilustración : Comprobación de existencia de datos en la base de datos

# Ejercicio 7: Utilización de una base de datos integrada relacional (SQLite), y almacenamiento del archivo de la base de datos en la carpeta de datos predeterminada de las aplicaciones de escritorio

Este ejercicio consistió en comprender lo que son las bases de datos integradas (en este caso SQLite) y algunas ventajas que proporciona su utilización en aplicaciones de escritorio. Para esto, se decidió hacer un ejercicio que consistiera en 2 partes: CRUD con una base de datos de SQLite, y almacenar el archivo de SQLite de la base de datos en una carpeta local.

Para la utilización, se decidió crear una base de datos con las mismas características que la base de datos creada para el ejercicio 2, y utilizarla por el mismo apartado de la aplicación creado en el mismo ejercicio:

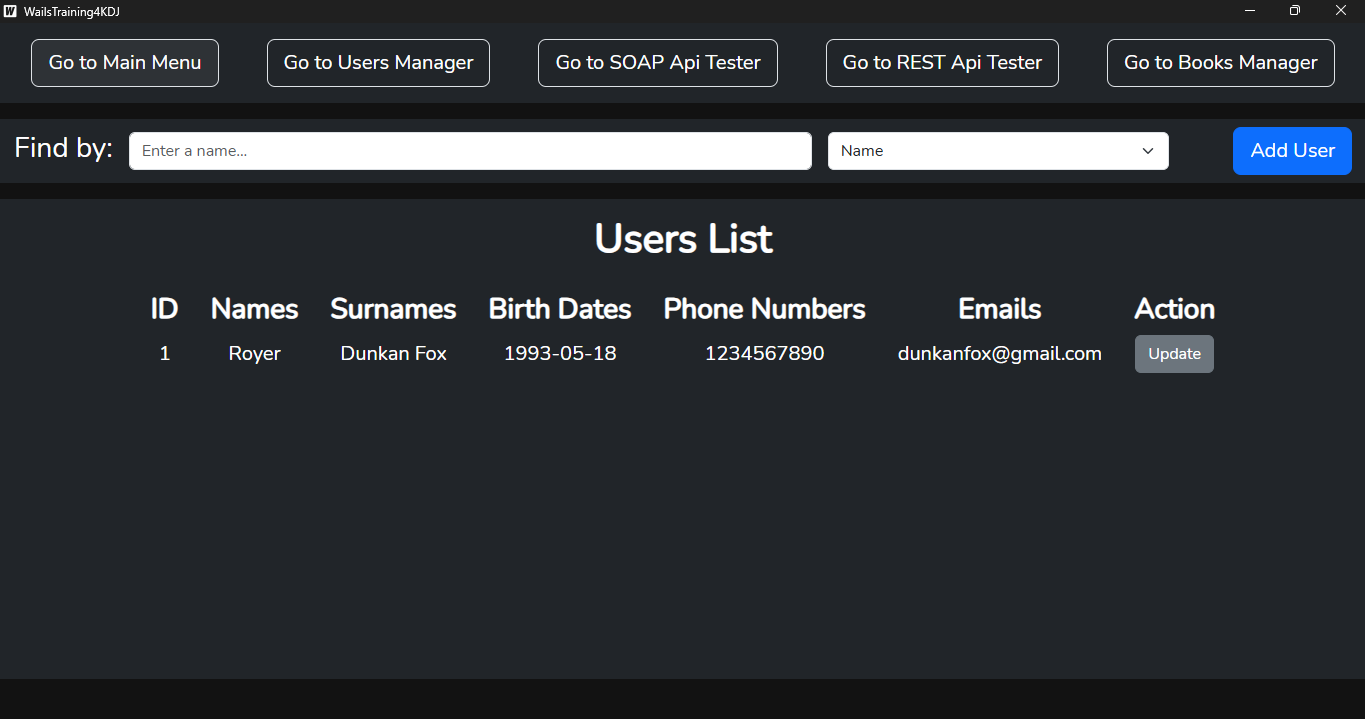


Ilustración : Comprobación de funcionamiento de la base de datos

Captura de pantalla de un celular

El contenido generado por IA puede ser incorrecto.

Ilustración : Agregación de usuario

Interfaz de usuario gráfica, Sitio web

El contenido generado por IA puede ser incorrecto.

Ilustración : Comprobación de agregación de usuario

Captura de pantalla de un celular

El contenido generado por IA puede ser incorrecto.

Ilustración : Modificación de usuario

Interfaz de usuario gráfica, Sitio web

El contenido generado por IA puede ser incorrecto.

Ilustración : Comprobación de modificación de usuario

Y, para el almacenamiento de la base de datos, en el código se agregó la funcionalidad de que el archivo de la base de datos fuera almacenado en la carpeta por defecto que utilizan las aplicaciones de escritorio para almacenar sus datos (“C:\Users\NombreDeUsuario\AppData\Roaming”):

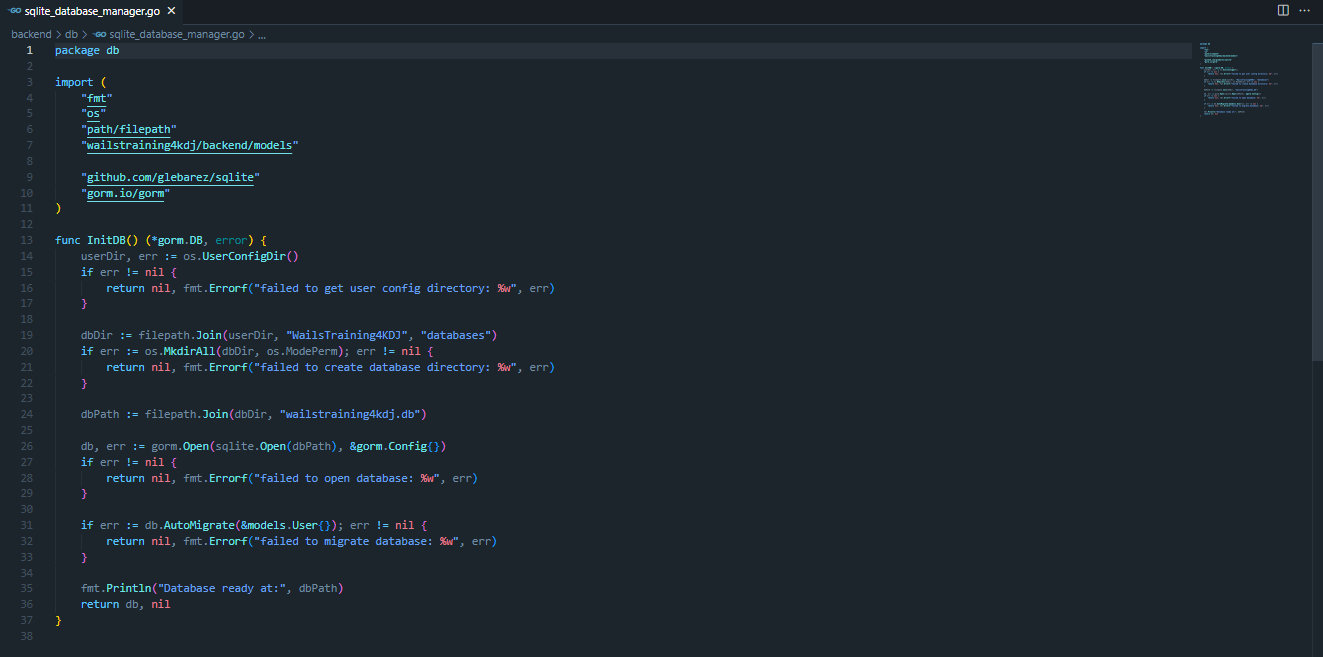


Ilustración : Código para la creación, almacenamiento y utilización de la base de datos SQLite

Captura de pantalla de un celular

El contenido generado por IA puede ser incorrecto.

Ilustración : Comprobación de creación del archivo de la base de datos SQLite

# Ejercicio 8: Utilización de una base de datos integrada no relacional (bbolt), y almacenamiento del archivo de la base de datos en la carpeta de datos predeterminada de las aplicaciones de escritorio

Este ejercicio consistió en realizar algo similar a lo realizado en el ejercicio 7, pero aquí la diferencia era usar una base de datos integrada no relacional (en este caso bbolt).

Para los datos de los libros y su almacenamiento, el modelo de libros utilizado es el mismo que el del ejercicio 6, y para el uso de la base de datos, se decidió utilizar el apartado de administración de libros creado también en el ejercicio 6:

Texto

El contenido generado por IA puede ser incorrecto.

Ilustración : Código para la creación, almacenamiento y utilización de la base de datos bbolt

Para poder cambiar entre bases de datos creadas en diferentes sistemas de gestión de bases de datos no relacionales, se tuvo que crear archivos llamados “repositorios” para colocar la lógica de la realización de CRUD, y modificar los archivos “book\_services.go” y “app.go” para que los repositorios funcionen:

Texto

El contenido generado por IA puede ser incorrecto.

Ilustración : Extracto de código del repositorio para bbolt

Texto

El contenido generado por IA puede ser incorrecto.

Ilustración : Extracto de código del repositorio para MongoDB

Texto

El contenido generado por IA puede ser incorrecto.

Ilustración : Extracto de código de "book\_service.go" modificado

Captura de pantalla de computadora

El contenido generado por IA puede ser incorrecto.

Ilustración : Extracto del código de “app.go” modificado

Nota: en el código de “app.go” hay código comentado, en este archivo se debe comentar y descomentar el código requerido para hacer funcionar unos sistemas de gestión de bases de datos u otros.

Interfaz de usuario gráfica, Aplicación

El contenido generado por IA puede ser incorrecto.

Ilustración : Comprobación del funcionamiento del apartado de administración de libros con la base de datos en bbolt

Captura de pantalla de un celular

El contenido generado por IA puede ser incorrecto.

Ilustración : Creación de libro nuevo

Interfaz de usuario gráfica, Aplicación, Teams

El contenido generado por IA puede ser incorrecto.

Ilustración : Comprobación de libro nuevo

Captura de pantalla de un celular

El contenido generado por IA puede ser incorrecto.

Ilustración : Modificación de libro

Interfaz de usuario gráfica, Aplicación, Teams

El contenido generado por IA puede ser incorrecto.

Ilustración : Comprobación de modificación de libro

Captura de pantalla de un celular

El contenido generado por IA puede ser incorrecto.

Ilustración : Comprobación de existencia de las bases de datos "SQLite" y "bbolt"

# Ejercicio 9: Implementación de concurrencia con GoRoutines en un apartado de descargas múltiples

Este ejercicio consistió en la realización de un instalador tradicional para aplicaciones de escritorio, el cual iba a contener las siguientes ventanas: