

TRABAJO INTEGRADOR FINAL

Parte 3: Tutorial Creando un CRUD con HTML, CSS, Bootstrap, JavaScript y Vue

¡Bienvenidos al Tutorial de Front-End para el proyecto en Flask! En esta guía, aprenderemos cómo crear una interfaz de usuario funcional que complementará nuestro Back-End desarrollado en Flask. El Front-End es la cara visible de nuestra aplicación, y es aquí donde los usuarios interactuarán con nuestro sistema.

Durante este tutorial, veremos cómo utilizar HTML y CSS para estructurar y estilizar nuestros elementos, así como también aprovecharemos la potencia de JavaScript y Vue.js para agregar interactividad y dinamismo.

Siguiendo los pasos y ejemplos proporcionados, podrás crear una interfaz atractiva y amigable para tu aplicación Flask.

En este proyecto en particular, crearemos un sistema CRUD de productos utilizando las siguientes tecnologías: HTML5, CSS3, Bootstrap 5, JavaScript, Vue.js, MySQL, Python 3.9.x y Flask. Seguiremos los pasos detallados y revisaremos los ejemplos de código para cada etapa del desarrollo.

Recuerda que este tutorial es solo el comienzo. ¡Las posibilidades de personalización y mejora de tu Front-End son infinitas! Te animamos a experimentar y explorar nuevas ideas para llevar tu proyecto en Flask al siguiente nivel.

¡Manos a la obra y disfruta del proceso de creación de tu propio sistema gestor de productos!

Sin más preámbulos, ¡comencemos con el Front-End para nuestro proyecto en Flask!

Material del FrontEnd para crear el CRUD

Aquí te proporcionamos el material necesario para crear el FrontEnd de tu CRUD. Utiliza los siguientes recursos:



1. Repositorio en Drive (FrontEnd):
 - Ingresa al repositorio haciendo clic en el enlace: [Repositorio del FrontEnd](#)
 - Aquí encontrarás los archivos necesarios para desarrollar el FrontEnd de tu CRUD.

Recuerda que estos recursos son fundamentales para comprender y llevar a cabo la creación del FrontEnd de tu proyecto.

Comenzando con el Front-End del proyecto

En esta etapa, nos enfocaremos en crear el Front-End del proyecto de manera independiente, antes de subirlo a servicios de hosting como Netlify o GitHub Pages. Trabajaremos en el entorno de desarrollo VSCode para construir y probar nuestra interfaz.

Para empezar, crearemos un archivo HTML llamado "productos.html". Este archivo contendrá una tabla que mostrará los registros de la base de datos. Utilizaremos tecnologías como Vue o JavaScript para obtener los datos desde el endpoint previamente creado en el Back-End. Además, agregaremos botones de "Nuevo", "Modificar" y "Eliminar" para facilitar la interacción con los registros.

Productos					
<button>Nuevo</button>					
Id	Nombre	Precio	Stock	Imagen	Accion
1	mouse XS	7000	25		<button>Editar</button> <button>Eliminar</button>
2	mouse	5000	15		<button>Editar</button> <button>Eliminar</button>
3	mouse XXX	8000	5		<button>Editar</button> <button>Eliminar</button>

Es importante destacar que, una vez que hayamos completado el desarrollo y realizado las pruebas localmente, podremos subir el proyecto a servicios de hosting como Netlify o GitHub Pages para hacerlo accesible en línea.

Código para la página de productos con Bootstrap y Vue.js

A continuación, te proporcionaremos el código correspondiente al archivo "productos.html".

Este código HTML representa la estructura básica de una página web que muestra una lista de productos. Utiliza la biblioteca Bootstrap para estilos y Vue.js para la manipulación dinámica de datos. En la página, se encuentra un encabezado con un menú (que completaremos más adelante), seguido de una tabla que muestra los productos con sus respectivos atributos como nombre, precio, stock e imagen. También se incluyen botones para editar y eliminar los productos. Al final del documento se importan los archivos de JavaScript necesarios para la interactividad y funcionalidad de la página.

Crear una carpeta llamada "js" y dentro de ella, crear un archivo llamado "productos.js": **Enlace a productos.js**

Este código utiliza Vue.js para crear una aplicación que interactúa con nuestra API para mostrar y gestionar una lista de productos. A continuación se explica el código paso a paso:

1. Se importa la función **createApp** de Vue.
2. Se utiliza la función **createApp** para crear una nueva instancia de la aplicación Vue.
3. En el objeto de configuración de la aplicación, se define una sección **data** que devuelve un objeto con diversas propiedades. Estas propiedades representan el estado de la aplicación, como la lista de productos, la URL de la API, indicadores de error y carga, y los atributos para guardar los valores del formulario.
4. En la sección **methods**, se definen los métodos que realizarán acciones en la aplicación. **fetchData** se encarga de obtener los datos de la API utilizando la función **fetch**. Los datos obtenidos se asignan a la propiedad **productos** y se actualiza el indicador **cargando** cuando la operación se completa. En caso de error, se activa el indicador **error**.
5. El método **eliminar** se utiliza para eliminar un producto de la lista. Realiza una petición **DELETE** a la API utilizando la URL correspondiente al producto. Después de eliminar el producto, se recarga la página para reflejar los cambios.
6. El método **grabar** se utiliza para guardar un nuevo producto en la lista. Crea un objeto **producto** con los valores del formulario y realiza una petición **POST** a la API. Si la petición es exitosa, se muestra una alerta y se redirige a la página "productos.html". En caso de error, se muestra una alerta de error.
7. Dentro de la sección **created**, se llama al método **fetchData** al iniciar la aplicación para obtener los datos iniciales de la API.
8. Finalmente, se utiliza el método **mount** para montar la aplicación Vue en el elemento con el id "app" del documento HTML. Esto conecta la instancia de la aplicación Vue con la interfaz de usuario.

En resumen, este código crea una aplicación Vue que muestra una lista de productos obtenida de nuestra API, permite eliminar productos de la lista y agregar nuevos productos mediante un formulario.

Generar un archivo denominado "main.js" dentro del directorio "js": **Enlace a main.js**

Este código se encarga de modificar el contenido HTML de un elemento con el id "header" utilizando JavaScript.

1. **document.getElementById("header")** busca y selecciona el elemento del documento HTML con el id "header".

2. Se asigna a **.innerHTML** una cadena de texto que representa el código HTML para construir una barra de navegación (navbar) utilizando la biblioteca Bootstrap.
3. La barra de navegación contiene un logotipo, un botón de navegación desplegable, enlaces a diferentes páginas (Home, Link) y un menú desplegable "CRUD" que contiene un enlace a "Productos".
4. Al asignar la cadena de texto a **.innerHTML** del elemento con el id "header", se reemplaza su contenido existente con la barra de navegación generada.

En resumen, este código inserta dinámicamente una barra de navegación en el elemento con el id "header" del documento HTML, utilizando una cadena de texto que representa el código HTML correspondiente. Esto permite agregar y personalizar la barra de navegación mediante JavaScript.

Ya podés abrir el archivo "productos.html" en tu navegador para probar su funcionalidad. Ahora podrás probar el botón de eliminar y verificar cómo se comporta.

Código para agregar productos en el sistema

A continuación deberás crear un nuevo archivo llamado "producto-create.html". Este archivo te permitirá agregar y registrar nuevos productos en tu sistema. Al crear este archivo, podrás acceder a una interfaz donde podrás ingresar los detalles del nuevo producto, como su nombre, precio, stock e imagen. De esta manera, podrás ampliar tu base de datos de productos de manera sencilla y eficiente. El código es el siguiente:

Enlace a producto-create.html

Este código crea una página llamada "Producto Nuevo". Aquí se presenta un formulario que permite agregar y registrar nuevos productos en el sistema. A continuación, se explica el propósito y las características clave del código:

- La página utiliza la biblioteca Bootstrap para estilos.
- El formulario se encuentra dentro de un contenedor con la clase "container" para mantener el diseño centrado en la página.
- El formulario consta de campos para ingresar los detalles del nuevo producto, como nombre, precio, stock e imagen. Cada campo tiene una etiqueta asociada y un campo de entrada donde los usuarios pueden ingresar los valores correspondientes.
- Los campos de entrada utilizan la directiva "v-model" de Vue.js para enlazar los valores ingresados por el usuario a propiedades específicas del objeto "data" en el componente Vue.
- Después de los campos de entrada, hay un botón "Grabar" que está asociado a un método llamado "grabar()" en el componente Vue. Cuando se hace clic en

el botón, se ejecuta el método "grabar()", que realiza la lógica de guardar los datos ingresados del nuevo producto.

- Al final del código HTML, se importan los archivos de JavaScript necesarios, así como los archivos "main.js" y "productos.js" que contienen la lógica adicional y las interacciones con Vue.js.

En resumen, este código crea una página web que muestra un formulario para agregar nuevos productos, utilizando Vue.js para el enlace de datos y Bootstrap para los estilos.

Ahora puedes verificar el funcionamiento de la función de agregar productos a tu sistema.

Simplemente abre la página "producto-nuevo.html" en tu navegador y encontrarás un formulario intuitivo donde podrás ingresar los detalles del nuevo producto. Completa los campos solicitados, como nombre, precio, stock e imagen, y luego haz clic en el botón "Grabar". Esto activará la función correspondiente que se encargará de guardar los datos ingresados.

Código para modificar productos en el sistema

Es hora de crear un archivo para actualizar productos existentes en tu sistema. Para ello, sigue los pasos a continuación:

1. Abre tu editor de código y crea un nuevo archivo.
2. Guarda el archivo con el nombre "producto-update.html".
3. Copia y pega el siguiente código en el archivo: Enlace a **producto-update.html**

El código proporciona una plantilla HTML para una página de "Actualizar Producto" con un formulario para editar los detalles del producto. Utiliza Bootstrap para dar estilo a los elementos y Vue.js para la interactividad.

Para editar productos existentes en tu sistema, necesitarás crear un archivo JavaScript específico. A continuación, te explico cómo crear el archivo "producto-edit.js":

1. Abre tu editor de código y ve a la carpeta donde se encuentra el directorio "js" de tu proyecto.
2. Dentro de la carpeta "js", crea un nuevo archivo y nómbralo "producto-edit.js".
3. Abre el archivo "producto-edit.js" en tu editor de código y edítalo con el siguiente código: **Enlace a producto-edit.js**

El código contiene un script que utiliza Vue.js para crear una aplicación que permite actualizar un producto en una base de datos.

- La línea **console.log(location.search)** imprime en la consola los argumentos pasados al formulario.
- La variable **id** se obtiene a partir de **location.search.substr(4)** y representa el identificador del producto.
- La aplicación Vue.js se crea usando **createApp()** y define un objeto de datos con propiedades como **id**, **nombre**, **imagen**, **stock** y **precio**.
- El método **fetchData(url)** se encarga de obtener los datos del producto desde la URL especificada usando la función **fetch()**. Luego, actualiza las propiedades del objeto de datos con los valores obtenidos.
- El método **modificar()** se activa cuando se hace clic en el botón de modificar. Envía una solicitud PUT al servidor con los datos actualizados del producto.
- Después de la actualización exitosa, se muestra una alerta y se redirige al usuario a la página "productos.html".
- En el bloque **created()**, se llama al método **fetchData(this.url)** para obtener los datos iniciales del producto al cargar la página.
- Por último, se monta la aplicación Vue en el elemento con el id "app" del documento HTML.

Ahora puedes probar la función de actualizar un registro.

El código permite actualizar un producto en la base de datos. Al cargar la página, se obtienen los datos del producto que se va a modificar. Luego, puedes editar los campos. Cuando haces clic en el botón de modificar, se envía una solicitud al servidor para actualizar el registro con los nuevos valores. Si la modificación es exitosa, se muestra una alerta y se redirige al usuario a la página "productos.html". En caso de error, se muestra una alerta de error.

Puedes probar esta funcionalidad ingresando los nuevos valores en los campos correspondientes y haciendo clic en el botón de modificar.

Redirección a la página de Productos

A continuación el código para un archivo **index.html** que redirecciona a **productos.html**: **Enlace a index.html**

Este código utiliza la etiqueta **<meta http-equiv="refresh">** para redireccionar automáticamente a **productos.html** después de un tiempo de espera de 0 segundos. Si la redirección automática no funciona, se muestra un enlace para hacer clic y acceder directamente a **productos.html**.

Recuerda asegurarte de tener un archivo **productos.html** en la misma ubicación que **index.html** para que la redirección funcione correctamente.

Despliegue del Frontend: Netlify y GitHub Pages

Para subir el frontend a Netlify o GitHub Pages, sigue los siguientes pasos:

Netlify:

1. Crea una cuenta en Netlify si aún no tienes una.
2. Accede al panel de control de Netlify.
3. Haz clic en el botón "New site from Git" o "New site".
4. Conecta tu repositorio de GitHub.
5. Selecciona el repositorio que contiene tu frontend.
6. Configura las opciones de compilación y configuración del sitio.
7. Haz clic en "Deploy site" o "Deploy".

GitHub Pages:

1. Abre tu cuenta de GitHub.
2. Crea un nuevo repositorio para tu frontend si aún no tienes uno.
3. Sube los archivos del frontend a tu repositorio de GitHub.
4. Asegúrate de que el archivo principal del frontend se llame "index.html".
5. Accede a la configuración de tu repositorio de GitHub.
6. Desplázate hacia abajo hasta la sección "GitHub Pages".
7. En la opción "Source" o "Branch", selecciona la rama o carpeta que contiene tu frontend.
8. Haz clic en "Save" o "Apply".

Una vez completados estos pasos, Netlify o GitHub Pages comenzarán a compilar y desplegar tu frontend. Puedes acceder a la URL proporcionada por la plataforma para ver tu frontend en línea.

Recuerda que debes tener los archivos del frontend correctamente configurados y funcionando correctamente antes de realizar el despliegue. Verifica que todos los enlaces a recursos, como archivos CSS y JavaScript, estén configurados correctamente en tu archivo HTML principal.

¡Ahora tu frontend estará disponible en Netlify o GitHub Pages para que puedas compartirlo con otros y acceder a él en cualquier momento!

Continúa explorando y mejorando tu Front-end

¡Felicidades por completar el Tutorial del Front-end! Has aprendido los fundamentos necesarios para crear una interfaz de usuario funcional. Sin embargo, esto es solo el comienzo de tu viaje en el desarrollo web.

Te animamos a que sigas explorando y practicando con tu Front. Puedes agregar nuevas características, mejorar el diseño, experimentar con diferentes bibliotecas y frameworks, y hacer que tu aplicación sea más impresionante.

Además, te invitamos a que consideres expandir tus habilidades y conocimientos explorando la parte del Back. Puedes comenzar por modificar la API en Flask que hemos utilizado en este tutorial. Personaliza la API según tus necesidades y añade nuevas funcionalidades a tu aplicación. Esto te permitirá tener un control total sobre los datos y la lógica del lado del servidor.

Recuerda que la programación es un proceso de aprendizaje continuo. Cuanto más practiques y experimentes, más habilidades desarrollarás. ¡No tengas miedo de probar cosas nuevas y desafiarte a vos mismo!

¡Sigue adelante y disfruta del mundo del desarrollo web!