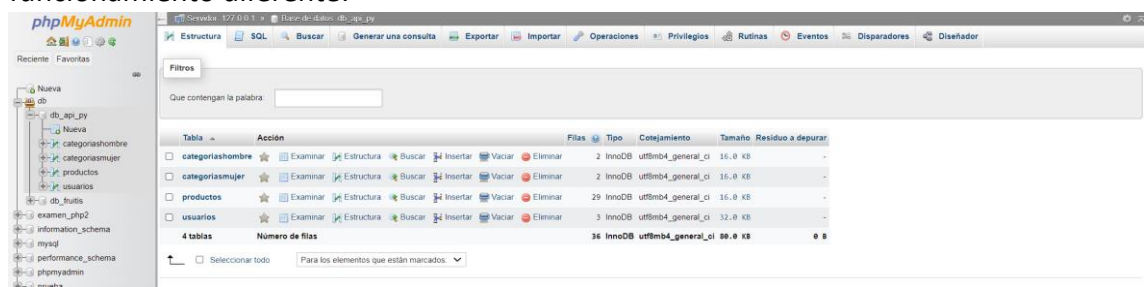


Documentación Api Israel Arribas

- Mi api se basa en una tienda de ropa, en la que he creado métodos get para recoger los productos y usuarios de la base de datos y métodos post, put y delete para modificar y añadir los datos a través de la api.
- El front esta compuesto por muchas paginas html ya que he reciclado y adaptado un proyecto del primer trimestre del año pasado, que hice antes de ni siquiera empezar a estudiar css, todas las fotos y diseños los realice el año pasado para dicho proyecto.
- La base de datos esta creada en localhost de phpmyadmin.

- Base de datos:

Mi base de datos esta compuesta por cuatro tablas, cada una con un funcionamiento diferente:



1. USUARIOS: Almacena los usuarios con todos sus datos:

#	Nombre	Tipo	Cotejamiento	Atributos	Nulo	Predeterminado	Comentarios	Extra	Acción
<input type="checkbox"/>	1	Nombre	varchar(20)	utf8mb4_general_ci	No	Ninguna			Cambiar Eliminar Más
<input type="checkbox"/>	2	Correo	varchar(20)	utf8mb4_general_ci	No	Ninguna			Cambiar Eliminar Más
<input type="checkbox"/>	3	Contraseña	varchar(20)	utf8mb4_general_ci	No	Ninguna			Cambiar Eliminar Más

2. PRODUCTOS: Almacena los productos con todos sus datos:

#	Nombre	Tipo	Cotejamiento	Atributos	Nulo	Predeterminado	Comentarios	Extra	Acción
<input type="checkbox"/>	1	Nombre	varchar(50)	utf8mb4_general_ci	No	Ninguna			Cambiar Eliminar Más
<input type="checkbox"/>	2	Precio	varchar(20)	utf8mb4_general_ci	No	Ninguna			Cambiar Eliminar Más
<input type="checkbox"/>	3	Url	varchar(50)	utf8mb4_general_ci	No	Ninguna			Cambiar Eliminar Más
<input type="checkbox"/>	4	Imagen	varchar(50)	utf8mb4_general_ci	No	Ninguna			Cambiar Eliminar Más
<input type="checkbox"/>	5	Imagen2	varchar(50)	utf8mb4_general_ci	No	Ninguna			Cambiar Eliminar Más
<input type="checkbox"/>	6	Imagen3	varchar(50)	utf8mb4_general_ci	No	Ninguna			Cambiar Eliminar Más
<input type="checkbox"/>	7	Categoría	varchar(20)	utf8mb4_general_ci	No	Ninguna			Cambiar Eliminar Más

3. CATEGORIAS HOMBRE/MUJER: Almacenan las categorías de ropa de hombre y de mujer respectivamente:

#	Nombre	Tipo	Cotejamiento	Atributos	Nulo	Predeterminado	Comentarios	Extra	Acción
<input type="checkbox"/>	1	Categoría	varchar(20)	utf8mb4_general_ci		No	Ninguna		Cambiar Eliminar Más
<input type="checkbox"/>	2	Ruta	varchar(50)	utf8mb4_general_ci		No	Ninguna		Cambiar Eliminar Más

#	Nombre	Tipo	Cotejamiento	Atributos	Nulo	Predeterminado	Comentarios	Extra	Acción
<input type="checkbox"/>	1	Categoría	varchar(20)	utf8mb4_general_ci		No	Ninguna		Cambiar Eliminar Más
<input type="checkbox"/>	2	Ruta	varchar(50)	utf8mb4_general_ci		No	Ninguna		Cambiar Eliminar Más

http://127.0.0.1:5000

- métodos GET
 1. /usuarios/ -> este método lo he creado para extraer los datos de la tabla Usuarios y así poder realizar el login y poder listar los usuarios en la pantalla del admin.


```
#Creamos metodo get para recibir y poder utilizar los usuarios de la base de datos
@application.route("/usuarios/", methods=["GET"])
def getUsuarios():
    if conexion.is_database_connection():
        #Ejecutamos una sencilla query con un select all
        filas = conexion.execute_query("SELECT * FROM usuarios")
        #Devolvemos un diccionario formateado a json mediante la libreria JGVutils
        return Uutils.format_json({"usuarios":filas})
```

20% Discount with the code: Heat2021

Inicio de sesión

```
1 check = document.getElementById("inputCheck");
2
3 function iniciarSesion(correo, contraseña) {
4     if (correo == "" || contraseña == "") {
5         alert("No ha introducido todos los datos");
6     }
7     fetch("http://127.0.0.1:5000/usuarios")
8     .then((ok) => {
9         console.log(ok);
10        return ok.json();
11    })
12    .then((ok1) => {
13        ok1.usuarios.forEach((element) => {
14            if (correo == element.Correo && contraseña == element.Contraseña) {
15                setTimeout(() => {
16                    window.location.href = "../PAGINAADMIN/PaginaPrincipal.html";
17                }, 1500);
18            }
19        });
20    });
21    .catch((err) => {
22        alert("Conexion incorrecta");
23    });
24 }
```

Find the shoebox for an additional 10% discount



Account
[Log out](#)

Shop

Usuarios
Productos

Cambiar
Añadir

Borrar

Nombre	Correo	Contraseña
admin	admin@gmail.com	admin
prueba	njuan@gmail.com	juanin
pruebaModificada	prueba@gmail.com	prueba

```

usuarios = document.getElementById("usuarios");

fetch("http://127.0.0.1:5000/usuarios/")
  .then((ok) => {
    console.log(ok);
    return ok.json();
  })
  .then((ok1) => {
    ok1.usuarios.forEach((element) => {
      usuarios.innerHTML += `
        <tr>
          <td>${element.Nombre}</td>
          <td>${element.Correo}</td>
          <td>${element.Contraseña}</td>
        </tr>`;
    });
  })
  .catch((err) => {
    alert("Conexion incorrecta");
  });

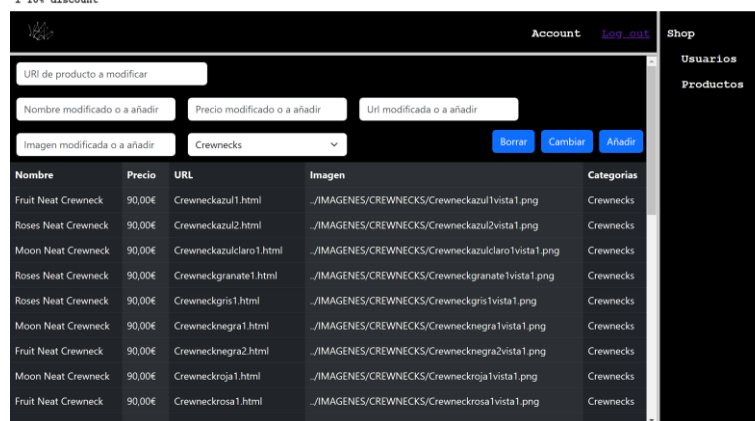
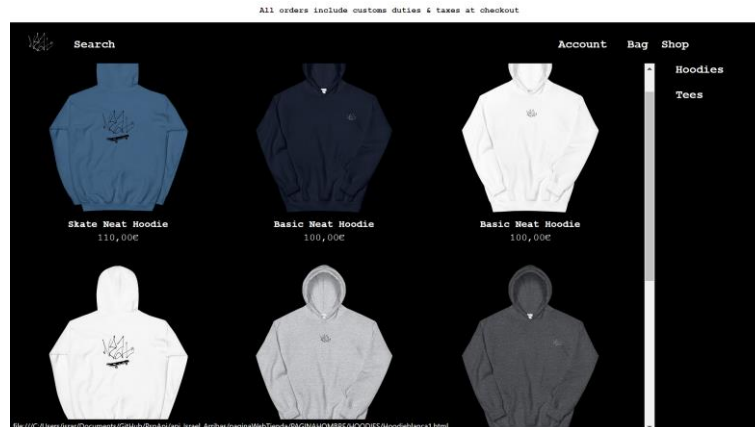
```

2. /usuarios/ -> este método lo he creado para extraer los datos de la tabla Productos y así poder poner los productos en la página, y poder listarlos en la pestaña administrador de manera que en el momento en el que introduzca uno nuevo en la base de datos o en la pestaña administrador estaría introducido automáticamente en la página.

```

#Otro metodo get para recibir y poder utilizar los productos, mismo funcionamiento
@application.route("/productos/", methods=["GET"])
def getProductos():
    if conexion.is_database_connection():
        filas = conexion.execute_query("SELECT * FROM productos")
        return Utils.format_json({"productos":filas})

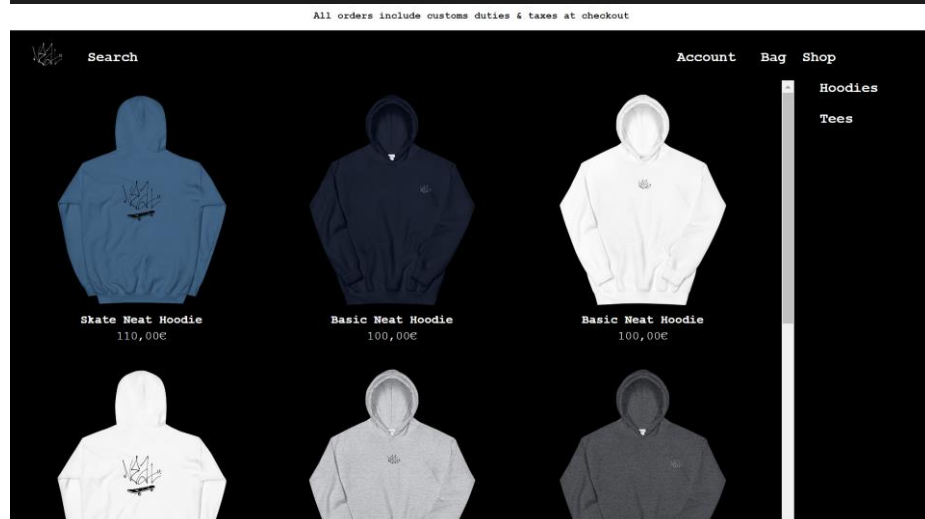
```

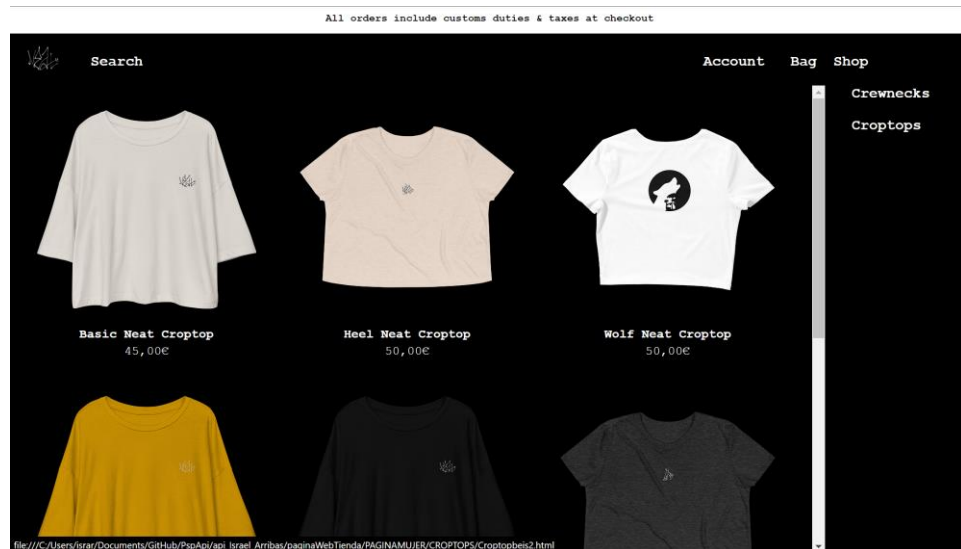


3. /categorias/mujer/ & /categorias/hombre/-> este método lo he creado para traer las categorías de ambos sexos respectivamente

```
#Otro metodo get para recibir y poder utilizar las categorias, mismo funcionamiento
@application.route("/categorias/hombre/", methods=["GET"])
def getCategoriasHombre():
    if conexion.is_database_connection():
        filas = conexion.execute_query("SELECT * FROM categoriasHombre")
        return Utils.format_json({"categorias":filas})

#Otro metodo get para recibir y poder utilizar las categorias, mismo funcionamiento
@application.route("/categorias/mujer/", methods=["GET"])
def getCategoriasMujer():
    if conexion.is_database_connection():
        filas = conexion.execute_query("SELECT * FROM categoriasMujer")
        return Utils.format_json({"categorias":filas})
```





- métodos POST

1. /registro/ -> este método lo he creado para introducir datos en la tabla Usuarios y así poder realizar el registro y poder añadir usuarios en la pantalla del admin.

```
#Creamos metodo post para poder enviar datos a la tabla usuarios de la base de datos
@application.route("/registro/", methods=["POST"])
def postUsuarios():
    #Requiero como "parametro" un json en el body
    datos = request.get_json()
    if conexion.is_database_connection():
        #Ejecutamos otra sentencia sql pero esta vez un Insert
        #indicando que los valores del nuevo elemento son los recogidos del json del body
        conexion.execute_query("INSERT INTO usuarios (Nombre, Correo, Contraseña) VALUES (%s, %s, %s)", [datos["Nombre"], datos["Correo"], datos["Contraseña"]])
    return datos
```

20% Discount with the code: Weat2021

Registro

Nombre

Correo

Contraseña

Regístrate

```
function registro(nombre, correo, contraseña) {
    if (nombre == "" || correo == "" || contraseña == "") {
        alert("No ha introducido todos los datos");
    } else {
        fetch("http://127.0.0.1:5000/registro/", {
            method: "POST",
            headers: {
                "Content-Type": "application/json",
            },
            body: JSON.stringify({
                Nombre: nombre,
                Correo: correo,
                Contraseña: contraseña,
            }),
        })
        .then((response) => {
            return response.json();
        })
        .then((data) => console.log(data));
        setTimeout(() => {
            window.location.href = "inicio_sesion.html";
        }, 1500);
    }
}
```

20% Discount with the code: Neat2021

Account [Log out](#) Shop

Nombre de usuario a modificar

Nombre modificado o a añadir Email modificado o a añadir Contraseña modificada o a añadir [Cambiar](#) [Añadir](#) [Borrar](#)

Nombre	Correo	Contraseña
admin	admin@gmail.com	admin
prueba	prueba@gmail.com	prueba
pruebaModificada	prueba@gmail.com	prueba

Usuarios
Productos

```
function registro(nombre, correo, contraseña) {
  if (nombre == "" || correo == "" || contraseña == "") {
    alert("No ha introducido todos los datos");
  } else {
    fetch("http://127.0.0.1:5000/registro/", {
      method: "POST",
      headers: {
        "Content-Type": "application/json",
      },
      body: JSON.stringify({
        Nombre: nombre,
        Correo: correo,
        Contraseña: contraseña,
      }),
    })
      .then((response) => {
        return response.json();
      })
      .then((data) => console.log(data));
  }
  setTimeout(() => {
    location.reload();
  }, 4000);
}
```

2. /registro/productos/ -> este método lo he creado para introducir datos en la tabla Productos y así poder introducir nuevos productos desde la pantalla admin.

```

Otro metodo get para poder enviar datos a la tabla productos, mismo funcionamiento
application.route("/registro/productos/", methods=["POST"])
def postProductos():
    datos = request.get_json()
    if conexion.is_database_connection():
        conexion.execute_query("INSERT INTO productos (Nombre, Precio, Url, Imagen, Categoria) VALUES (%s, %s, %s, %s, %s)", [datos["Nombre"], datos["Precio"], datos["Url"], datos["Imagen"], datos["Categoria"]])
    return datos

```

20% Discount with the code: Neat2021

Account [Log out](#) Shop

URI de producto a modificar

Nombre modificado o a añadir Precio modificado o a añadir Url modificada o a añadir

Imagen modificada o a añadir Hoodies [Borrar](#) [Cambiar](#) [Añadir](#)

Nombre	Precio	URL	Imagen	Categorias
Fruit Neat Crewneck	90.00€	Crewneckazul1.html	./IMAGENES/CREWNECKS/Crewneckazul1vista1.png	Crewnecks
Roses Neat Crewneck	90.00€	Crewneckazul2.html	./IMAGENES/CREWNECKS/Crewneckazul2vista1.png	Crewnecks
Moon Neat Crewneck	90.00€	Crewneckazulclaro1.html	./IMAGENES/CREWNECKS/Crewneckazulclaro1vista1.png	Crewnecks
Roses Neat Crewneck	90.00€	Crewneckgranate1.html	./IMAGENES/CREWNECKS/Crewneckgranate1vista1.png	Crewnecks
Roses Neat Crewneck	90.00€	Crewneckgris1.html	./IMAGENES/CREWNECKS/Crewneckgris1vista1.png	Crewnecks
Moon Neat Crewneck	90.00€	Crewnecknegra1.html	./IMAGENES/CREWNECKS/Crewnecknegra1vista1.png	Crewnecks
Fruit Neat Crewneck	90.00€	Crewnecknegra2.html	./IMAGENES/CREWNECKS/Crewnecknegra2vista1.png	Crewnecks
Moon Neat Crewneck	90.00€	Crewneckroja1.html	./IMAGENES/CREWNECKS/Crewneckroja1vista1.png	Crewnecks
Fruit Neat Crewneck	90.00€	Crewneckrosa1.html	./IMAGENES/CREWNECKS/Crewneckrosa1vista1.png	Crewnecks

Usuarios
Productos


```
function registro(nombre, precio, url, imagen, categoria) {
  if (
    nombre == "" ||
    precio == "" ||
    url == "" ||
    imagen == "" ||
    categoria == ""
  ) {
    alert("No ha introducido todos los datos");
  } else {
    fetch("http://127.0.0.1:5000/registro/productos/", {
      method: "POST",
      headers: {
        "Content-Type": "application/json",
      },
      body: JSON.stringify({
        Nombre: nombre,
        Precio: precio,
        Url: url,
        Imagen: imagen,
        Categoria: categoria,
      }),
    })
      .then((response) => {
        return response.json();
      })
      .then((data) => console.log(data));
  }
  setTimeout(() => {
    location.reload();
  }, 4000);
}
```

- métodos PUT

1. /modificar/ -> este método lo he creado para modificar datos en la tabla Usuarios y así poder modificar usuarios en la pantalla del admin.

```
#Creamos metodo put para poder modificar datos de la tabla usuarios de la base de datos
@application.route("/modificar/", methods=["PUT"])
def modificarUsuarios():
    #Requiero como "parametro" un json en el body
    datos = request.get_json()
    if conexion.is_database_connection():
        #Ejecutamos otra sentencia sql pero esta vez un Update
        #Indicando que los valores del nuevo elemento son los recogidos del json del body
        conexion.execute_query("UPDATE usuarios SET Nombre = %s, Correo = %s, Contraseña = %s WHERE Nombre = %s", [datos["Nombre"], datos["Correo"], datos["Contraseña"], datos["NombreAntiguo"]])
    return datos
```

20% Discount With the code: Heat2021



Account
[Log out](#)

[shop](#)

[Usuarios](#)
[Productos](#)

Nombre	Correo	Contraseña
admin	admin@gmail.com	admin
prueba	njuan@gmail.com	juamin
pruebaModificada	prueba@gmail.com	prueba

```
function cambiar(nombre, correo, contraseña, nombreAntiguo) {
  if (nombre == "" || correo == "" || contraseña == "" || nombreAntiguo == "") {
    alert("No ha introducido todos los datos");
  } else {
    fetch("http://127.0.0.1:5000/modificar/", {
      method: "PUT",
      headers: {
        "Content-Type": "application/json",
      },
      body: JSON.stringify({
        Nombre: nombre,
        Correo: correo,
        Contraseña: contraseña,
        NombreAntiguo: nombreAntiguo,
      }),
    })
      .then((response) => {
        return response.json();
      })
      .then((data) => console.log(data));
  }
  setTimeout(() => {
    location.reload();
  }, 4000);
}
```

2. /registro/productos/ -> este método lo he creado para modificar datos en la tabla Productos y así poder modificar productos desde la pantalla admin.

```

//creo metodo get para poder modificar datos a la tabla productos, mismo funcionamiento
@application.route("/modificar/productos/", methods=["PUT"])
def modificarProductos():
    datos = request.get_json()
    if conexion.is_database_connection():
        conexion.execute_query("UPDATE productos SET Nombre = %s, Precio = %s, Url = %s, Imagen=%s, Imagen2=%s, Categoria=%s WHERE Url = %s", [datos["Nombre"], datos["Precio"], datos["Url"], datos["Imagen"], datos["Imagen2"], datos["Categoria"], datos["UrlAntigua"]])
    return datos

```

20% Discount with the code: Neat2021

Account [Log out](#) Shop

Usuarios
Productos

URI de producto a modificar

Nombre modificado o a añadir Precio modificado o a añadir Url modificada o a añadir

Imagen modificada o a añadir Hoodies [Borrar](#) [Cambiar](#) [Añadir](#)

Nombre	Precio	URL	Imagen	Categorias
Fruit Neat Crewneck	90.00€	Crewneckazul1.html	../IMAGENES/CREWNECKS/Crewneckazul1vista1.png	Crewnecks
Roses Neat Crewneck	90.00€	Crewneckazul2.html	../IMAGENES/CREWNECKS/Crewneckazul2vista1.png	Crewnecks
Moon Neat Crewneck	90.00€	Crewneckazulclaro1.html	../IMAGENES/CREWNECKS/Crewneckazulclaro1vista1.png	Crewnecks
Roses Neat Crewneck	90.00€	Crewneckgranate1.html	../IMAGENES/CREWNECKS/Crewneckgranate1vista1.png	Crewnecks
Roses Neat Crewneck	90.00€	Crewneckgris1.html	../IMAGENES/CREWNECKS/Crewneckgris1vista1.png	Crewnecks
Moon Neat Crewneck	90.00€	Crewnecknegra1.html	../IMAGENES/CREWNECKS/Crewnecknegra1vista1.png	Crewnecks
Fruit Neat Crewneck	90.00€	Crewnecknegra2.html	../IMAGENES/CREWNECKS/Crewnecknegra2vista1.png	Crewnecks
Moon Neat Crewneck	90.00€	Crewneckroja1.html	../IMAGENES/CREWNECKS/Crewneckroja1vista1.png	Crewnecks
Fruit Neat Crewneck	90.00€	Crewneckrosa1.html	../IMAGENES/CREWNECKS/Crewneckrosa1vista1.png	Crewnecks

```

function cambiar(nombre, precio, url, imagen, categoria, UrlAntigua) {
    if (
        nombre == "" ||
        precio == "" ||
        url == "" ||
        imagen == "" ||
        categoria == "" ||
        UrlAntigua == ""
    ) {
        alert("No ha introducido todos los datos");
    } else {
        fetch("http://127.0.0.1:5000/modificar/productos/", {
            method: "PUT",
            headers: {
                "Content-Type": "application/json",
            },
            body: JSON.stringify({
                Nombre: nombre,
                Precio: precio,
                Url: url,
                Imagen: imagen,
                Categoria: categoria,
                UrlAntigua: UrlAntigua,
            })
        })
        .then((response) => {
            return response.json();
        })
        .then((data) => console.log(data));
    }
    setTimeout(() => {
        location.reload();
    }, 4000);
}

```

- métodos DELETE

1. /eliminar/ -> este método lo he creado para eliminar datos en la tabla Usuarios y así poder eliminar usuarios en la pantalla del admin.

```

#Creamos metodo delete para poder eliminar datos de la tabla usuarios de la base de datos
@application.route("/eliminar/", methods=["DELETE"])
def deleteUsuarios():
    #Requiero como "parametro" un json en el body
    datos = request.get_json()
    if conexion.is_database_connection():
        #Ejecutamos otra sentencia sql pero esta vez un delete
        #indicando que los valores del nuevo elemento son los recogidos del json del body
        conexion.execute_query("DELETE FROM usuarios WHERE Nombre=%s", [datos["Nombre"]])
    return datos

```


20% Discount with the code: Neat2021

Account [Log_out](#)

Shop

Usuarios

Productos

Nombre de usuario a modificar

Nombre modificado o a añadir Email modificado o a añadir Contraseña modificada o a añadir [Cambiar](#) [Añadir](#) [Borrar](#)

Nombre	Correo	Contraseña
admin	admin@gmail.com	admin
prueba	njuan@gmail.com	juanin
pruebaModificada	prueba@gmail.com	prueba

```
function eliminar(nombre) {
  if (nombre == "") {
    alert("No ha introducido todos los datos");
  } else {
    fetch("http://127.0.0.1:5000/eliminar/", {
      method: "DELETE",
      headers: {
        "Content-Type": "application/json",
      },
      body: JSON.stringify({
        Nombre: nombre,
      }),
    })
      .then((response) => {
        return response.json();
      })
      .then((data) => console.log(data));
  }
  setTimeout(() => {
    location.reload();
  }, 4000);
}
```

2. /eliminar/productos/ -> este método lo he creado para eliminar datos en la tabla productos y así poder eliminar productos en la pantalla del admin.

```
#Otro metodo delete para poder eliminar datos a la tabla productos, mismo funcionamiento
@application.route("/eliminar/productos/", methods=["DELETE"])
def deleteProductos():
    datos = request.get_json()
    if conexion.is_database_connection():
        conexion.execute_query("DELETE FROM productos WHERE Url=%s", [datos["Url"]])
    return datos
```

20% Discount with the code: Neat2021

Account [Log_out](#)

Shop

Usuarios

Productos

URI de producto a modificar

Nombre modificado o a añadir Precio modificado o a añadir Url modificada o a añadir

Imagen modificada o a añadir Hoodies [Borrar](#) [Cambiar](#) [Añadir](#)

Nombre	Precio	URL	Imagen	Categorias
Fruit Neat Crewneck	90,00€	Crewneckazu1.html	./IMAGENES/CREWNECKS/Crewneckazu1vista1.png	Crewnecks
Roses Neat Crewneck	90,00€	Crewneckazu2.html	./IMAGENES/CREWNECKS/Crewneckazu2vista1.png	Crewnecks
Moon Neat Crewneck	90,00€	Crewneckazuclaro1.html	./IMAGENES/CREWNECKS/Crewneckazuclaro1vista1.png	Crewnecks
Roses Neat Crewneck	90,00€	Crewneckgranate1.html	./IMAGENES/CREWNECKS/Crewneckgranate1vista1.png	Crewnecks
Roses Neat Crewneck	90,00€	Crewneckgris1.html	./IMAGENES/CREWNECKS/Crewneckgris1vista1.png	Crewnecks
Moon Neat Crewneck	90,00€	Crewnecknegra1.html	./IMAGENES/CREWNECKS/Crewnecknegra1vista1.png	Crewnecks
Fruit Neat Crewneck	90,00€	Crewnecknegra2.html	./IMAGENES/CREWNECKS/Crewnecknegra2vista1.png	Crewnecks
Moon Neat Crewneck	90,00€	Crewneckroja1.html	./IMAGENES/CREWNECKS/Crewneckroja1vista1.png	Crewnecks
Fruit Neat Crewneck	90,00€	Crewneckrosa1.html	./IMAGENES/CREWNECKS/Crewneckrosa1vista1.png	Crewnecks

```
function eliminar(url) {
  if (url == "") {
    alert("No ha introducido todos los datos");
  } else {
    fetch("http://127.0.0.1:5000/eliminar/productos/", {
      method: "DELETE",
      headers: {
        "Content-Type": "application/json",
      },
      body: JSON.stringify({
        Url: url,
      }),
    })
      .then((response) => {
        return response.json();
      })
      .then((data) => console.log(data));
  }
  setTimeout(() => {
    location.reload();
  }, 4000);
}
```