



VUE FINAL Hack a Market



Descripción del proyecto

Proyecto '**Hack a Market**'.

Simula un mercado donde ver productos; editar, borrar, ver y crear clientes.



Qué necesitas

Ten a mano la teoría de **VUE: VISTAS Y COMPONENTES**, y **VUE: API + FRONT**, además de los 2 proyectos que hemos hecho en clase comunicando el back y el front (*Hack a notas* y *Hack a clientes*).



Set up del proyecto

Crea tu proyecto Vue, llámalo **hackamarket**.

Instalar todas las dependencias necesarias:

◆ `npm i -D express nodemon`

◆ `npm i --save axios`

- ◆ `npm i --save cors`
- ◆ `npm i --save body-parser`
- ◆ `npm i --save mysql`

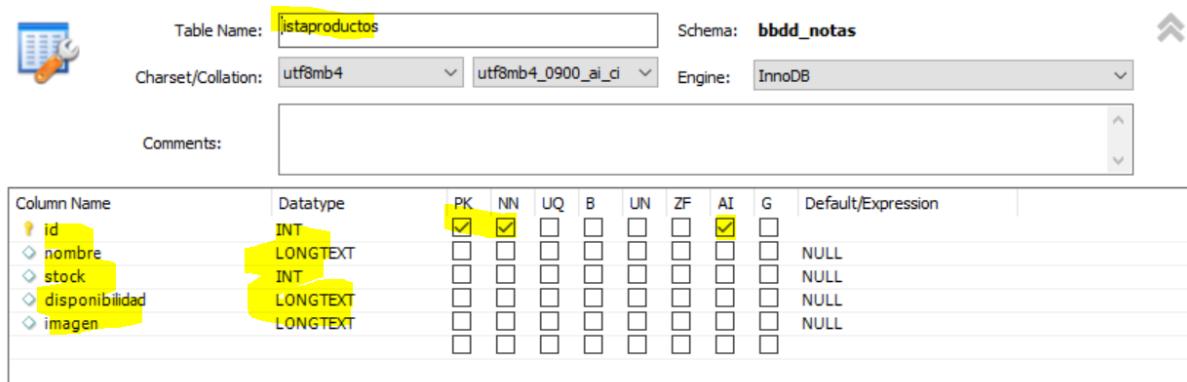
Otras opcionales:

- ◆ `npm i --save vue-headful`
- ◆ `npm i --save sweetalert2`

Recuerda crear una carpeta **api** en **src** y dentro un archivo `api.js` donde crear tu API y tus endpoints.

Debes crear, además, 2 **tablas** en una base de datos existente u otra nueva:

TABLA DE PRODUCTOS



The screenshot shows the MySQL Workbench interface for creating a new table. The table name is set to 'listaproductos'. The schema is 'bbdd_notas'. The engine is set to InnoDB. The table has five columns: 'id' (PK, INT, checked), 'nombre' (LONGTEXT, checked), 'stock' (INT, checked), 'disponibilidad' (LONGTEXT, checked), and 'imagen' (LONGTEXT, checked). All other checkboxes (NN, UQ, B, UN, ZF, AI, G) are unchecked. The 'Default/Expression' column shows 'NULL' for all columns.

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
id	INT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>					
nombre	LONGTEXT	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
stock	INT	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
disponibilidad	LONGTEXT	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
imagen	LONGTEXT	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL



NO CAMBIAS NINGÚN DATO. Recuerda que las tablas deben ser iguales para que funcionen en mi local de la misma manera.

TABLA DE CLIENTES

The screenshot shows the MySQL Workbench interface for creating a new table. The table name is set to 'listadientes'. The schema is 'bbdd_notas', charset is 'utf8mb4', and engine is 'InnoDB'. The table has six columns: 'id', 'nombre', 'usuario', 'password', 'email', and 'foto'. The 'id' column is defined as INT, primary key (PK), not null (NN), auto-increment (AI), and has a default value of NULL. The other five columns are defined as LONGTEXT.

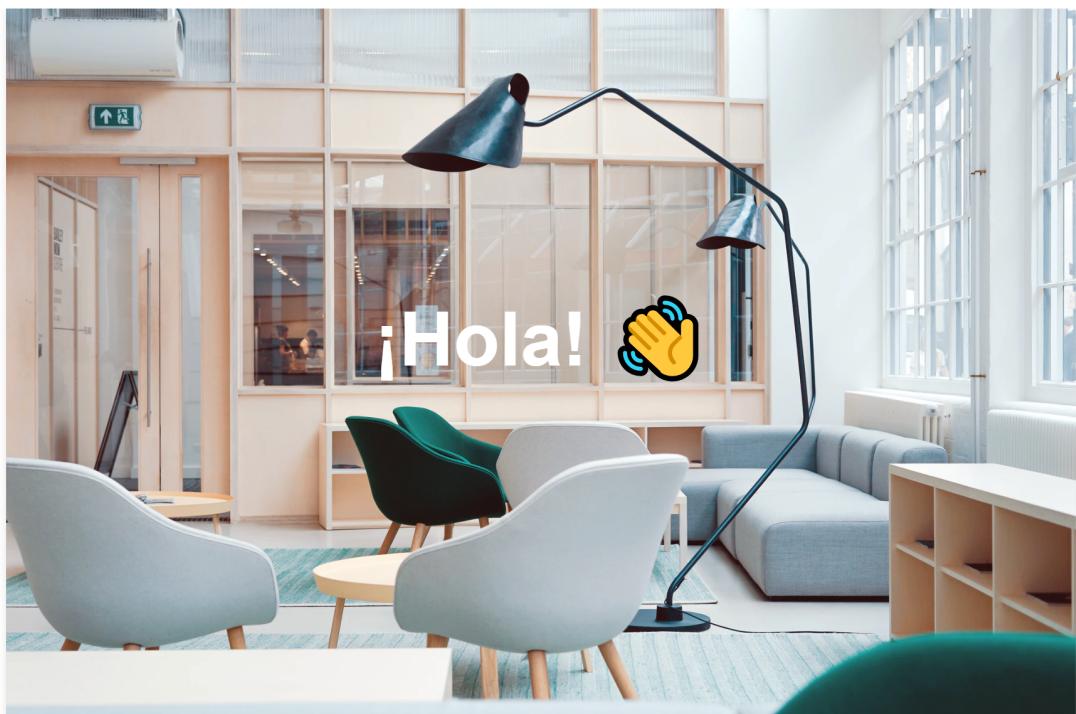
Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
id	INT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	NULL				
nombre	LONGTEXT	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
usuario	LONGTEXT	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
password	LONGTEXT	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
email	LONGTEXT	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
foto	LONGTEXT	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL



NO CAMBIAS NINGÚN DATO. Recuerda que las tablas deben ser iguales para que funcionen en mi local de la misma manera.

📌 Screenshots:

HOME:



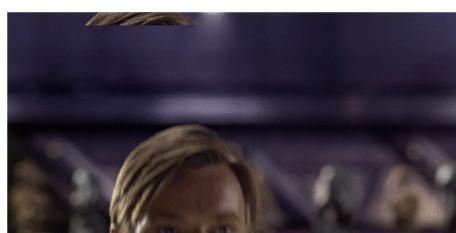
Desiré M Carmona 2020, 2021

En Home hay diseño libre. Pon lo que prefieras.

ABOUT:

Sobre mí

Hello there! 🌟

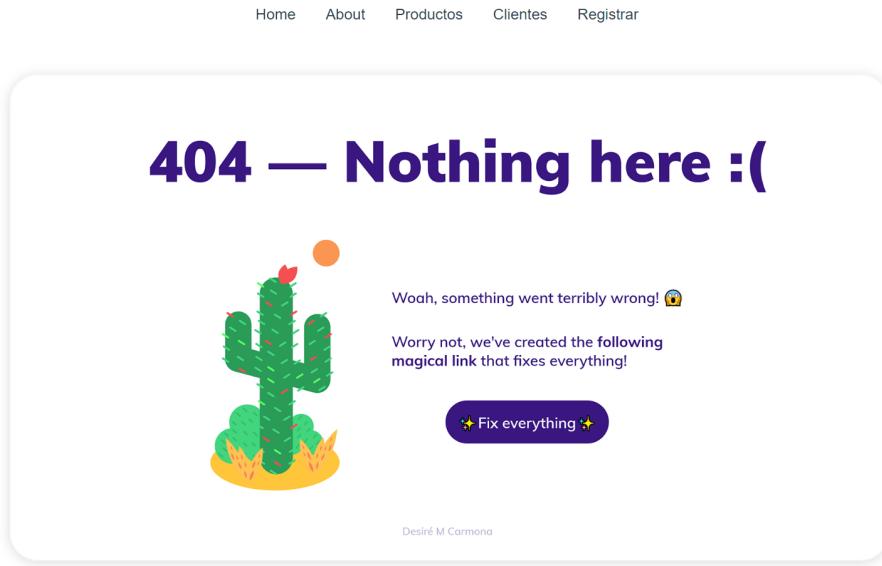


Hay poco que contar sobre mí. Eso es un ejercicio del Bootcamp de HACK A BOSS
que los alumnos/as deben realizar si quieren aprobar mi asignatura. 😊

Desiré M Carmona 2020, 2021

En about debes poner tus redes sociales, o información básica sobre ti y las tecnologías que has utilizado.

ERROR:



Página de error. Pon algo original.

PRODUCTOS:

Listado de productos



Nombre: Sillón verde
Stock: 3
Disponibilidad: Disponible

[Comprar](#) [Reservar](#)



Nombre: Sillón de piel
Stock: 5
Disponibilidad: Disponible

[Comprar](#) [Reservar](#)



Nombre: Sillón blanco pequeño
Stock: 0
Disponibilidad: No disponible

Sin fecha de entrada.



Nombre: Sillón marrón
Stock: 2
Disponibilidad: Disponible

[Comprar](#) [Reservar](#)



Nombre: Escultura plátano
Stock: 0
Disponibilidad: No disponible

Sin fecha de entrada.



Nombre: Un aguacate
Stock: 0
Disponibilidad: No disponible

Sin fecha de entrada.

Desiré M Carmona 2020, 2021

En esta vista hay una función principal: **UN GET DE PRODUCTOS A LA API A TRAVÉS DE AXIOS**. Estos productos se muestran a través de un **componente importado en la vista**.

Además hay otras varias funcionalidades.

Obligatorias: los botones de "**comprar**" y "**reservar**" hacen saltar un sweetalert simple que ponga "¡Gracias por comprar el producto" o "¡Gracias por reservar el producto!" respectivamente.

Opcionales: 1. Que se muestren o se escondan los botones dependiendo de la disponibilidad del producto. 2. Que la disponibilidad se vea de un color u otro según si está disponible o no disponible.

CLIENTES:

Home About Productos **Clientes** Registrar

Listado de clientes



Nombre: María
Usuario: mary_2364
Email: mary2@gmail.com
Contraseña: 1234524

[Editar](#) [Borrar](#)



Nombre: Ernesto
Usuario: nestor_62
Email: nestor_23@gmail.com
Contraseña: nestorete381

[Editar](#) [Borrar](#)



Nombre: Rebeca
Usuario: bequi
Email: rebecaBC@gmail.com
Contraseña: rebccccca1

[Editar](#) [Borrar](#)



Nombre: Andrea
Usuario: drea224
Email: andrea@gmail.com
Contraseña: andreh231

[Editar](#) [Borrar](#)



Nombre: Kevin
Usuario: kvin17
Email: kevvin@gmail.com
Contraseña: kevinalaster

[Editar](#) [Borrar](#)



Nombre: Laura
Usuario: lalala3
Email: laura_234@gmail.com
Contraseña: lauritaaa5

[Editar](#) [Borrar](#)

Desiré M Carmona 2020, 2021

En esta vista se podrá:

1. **Ver** los clientes. (GET)
2. **Editar** los clientes a través de un modal. (PUT) Esta función es disparada por un evento emitido desde el componente.
3. **Borrar** los clientes. (DELETE) Esta función es disparada por un evento emitido desde el componente.

Ten en cuenta que los clientes han de **verse desde un componente importado en la vista**. Este componente ha de lanzar los **eventos** respectivos para lanzar la **edición** de clientes o **borrar** un cliente **a través de un modal que crees en esta propia vista**.

Recuerda que todo esto ya lo hemos hecho en el proyecto de Hack a Clientes.

REGISTRO DE CLIENTES:

Home About Productos Clientes **Registrar**

Registrando nuevo cliente

Nombre del cliente
Usuario del cliente
Email del cliente
Contraseña del cliente
Fotografía (en URL) del cliente

Crear cliente

Desiré M Carmona 2020, 2021

Una vista que permite añadir clientes a través de un **POST**.



Recomendaciones para el proyecto

1. Primero **crea el proyecto e instala las dependencias necesarias**.
2. **Crea las tablas/bases de datos** que necesites.
3. **Crea la API y los endpoints** y **asegúrate de que tu conexión funciona** en ambas consolas.
4. Crea las **vistas menos importantes**: Home, About y Error.

5. Crea la vista del **Registro de clientes** y crea la **lógica** necesaria para **añadir clientes (EL POST)** a la base de datos.
6. En la vista de **Productos**, empieza creando la lógica necesaria para recuperar los productos desde la API (EL GET) y luego consigue que se muestren a través de un **componente**.
7. En la vista de **Clientes**, empieza creando la lógica necesaria para recuperar los clientes desde la API (EL GET) y luego consigue que se muestren a través de un **componente**.
8. En el **componente** desde donde muestras los clientes, crea los **eventos** necesarios para disparar las funciones de **EDITAR (PUT)** y **BORRAR (DELETE)** que has creado en la vista.

A continuación tienes **screenshots de las vistas** y **una explicación de lo que se debe hacer en cada una**:

Requisitos obligatorios



Además de los componentes necesarios para cada vista, crea un componente para **menú** y otro para el **footer**. El menú debe contener **links dinámicos**, no **estáticos**.



Es obligatorio presentar un **CSS original**. (Original quiere decir **propio**, personalizado).



Todas las llamadas y datos deben **funcionar y mostrarse correctamente en el front**.



Uso de **Sweetalert** funcional allá donde se pida.



No puede haber ningún **error** ni en **consola** ni en el **navegador** a la hora de entregar este proyecto.

📌 Requisitosopcionales



Uso de CSS Spinner.



Uso de Vue-Headful.



Que la aplicación sea responsive.



Añadir las opciones de borrar, o añadir, o editar productos. (No hacen falta las 3).