Proyecto 2 - Paola & SoonWoo

**Supermercado “SUPA”**

BEST BUY API / MERCADONA API

models

seeds.js

routes

views

index.html

layout.html

ROLES

-Administrador

-clientes/socios

MVP

- Git branches and PRs

- Readme with endpoints table for Monday

- First Heroku deployment the latest by Tuesday

- Seeds file

- .env file

- Login / signup

- Use at least one third party API

- At least 2 mongoose models

- Create a readme.md with the endpoints table and the instructions to start your app.

- Implement UI framework

**@canal** obligatorio para el lunes:

* TABLA DE ENDPOINTS
* APIS TESTEADAS, COMPROBAD QUE FUNCIONAN OK
* MODELOS YA PICADOS
* SI LO ANTERIOR ESTÁ RESUELTO, COMENZAD CON EL SEED

Referencia endpoints

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Id** | **Method** | **Path** | **Description** | **permisos** |
| 1 | get | /market/newlist | Muestra el formulario para crear una lista de productos | Admin, dev, |
| 2 | post | /market/newlist | Guarda en la BBDD un producto | Admin, dev, |
| 3 | get | /product/list | Muetra la lsita de todos los productos | Admin, dev, client, guest |
| 4 | post | /product/cart | Guarda en la BBDD el carrito/cesta | Admin, dev, client, guest |
| 5 | get | /product/:id | Muestra los detalles de un producto | Admin, dev, client, guest |
| 6 | get | /product/delete?id=xxx | Elimina de la BBDD un producto | Admin, dev |
| 7 | get | /product/edit?id=xxx | Muestra el formulario para editar un producto | Admin, dev, |
| 8 | post | /product/edit?id=xxx | Edita en la BBDD el producto | Admin, dev, |
| 9 | Bcrypt | /register | Muestra el formulario para registrar al guest | Admin, dev, , client, guest |
| 10 | bcrypt | /login | Muestra el formulario para entrar a la cuenta | Admin, dev, , client, guest |
| 11 | bcrypt | /logout | Logout de cliente | Admin, dev, , client, guest |
| 12 | get | /profile | Muestra el perfil del usuario | Admin, client |
| 13 | get | /profile/edit?id=xxx | Muestra el formulario para editar el perfil | Admin, client |
| 14 | post | /profile/edit?id=xxx | Edita en la BBDD el perfil | Admin, client |
| 15 | get | /profiles | Muestra todos los perfiles | Admin |
| 16 | post | /profiles/edit?id=xxx | Edita en la BBDD cualquier perfil | Admin |
|  |  |  |  |  |
|  |  |  |  |  |

Diseño:

Index

* imagen principal del Supa con botones login, register, ver productos
* carousel: imagen dentro del supermercado, comida, servicios, etc….
* footer: derechos, localización(maps) y RRSS

layout

* nav bar

product list

* menu en la izq con las categorías de producto desplegable
* vista principal con boton de busqueda y vista de todos los productos filtrados

register / login

* formulario con los campos requeridos

profile

- vista de todos los campos del usuario

extras

- cloudinary

Iteraciones

0.- Comienza a trabajar [sobre este repo](https://github.com/GermanAlvarez-IH/repaso-crud/)

1.- Crea un modelo Park con el esquema:

name: Str

description: Str

active: Bool

2.- Crea un modelo Coaster con el esquema:

name: Str

description: Str

inversions: Num

length: Num

active: Bool

park: ObjectId (referenciando al modelo Park)

3.- Utiliza los modelos para popular la BBDD mediante el seed.js alojado en /bin

4.- Crea la ruta 1 en el servidor y muestra la vista de parks/new-park.hbs

5.- Crea y configura la ruta 2 en el servidor para guardar parques en la BBDD mediante el modelo Park, testea mediante Postman su validez. Renderiza a continuación en el cliente la vista new-park.hbs (ruta 1) cuyo formulario te permitirá crear nuevos parques. No olvides modificar los atributos action y method de su formulario, así como el atributo name de cada campo para conectarlo adecuadamente con el controlador.

6.- Testea en la BBDD que el formulario crea parques sin incidencia.

7.- Crea la ruta 3 en el servidor y muestra en el cliente la vista coasters/new-coaster.hbs

8.- En en formulario, las opciones del desplegable deben contener los nombres de cada uno de los parques de la BBDD (debes enviarlos desde el controlador que genera la vista haciendo uso del modelo Park). Añade a cada option el atributo value con el \_id de ese parque. Ejemplo:

<select name="park" class="form-control" id="park-list">

<option>Seleccionar</option>

<option value="5d408c5eff558796d4cf681e">Port Aventura</option>

<option value="5d408c5eff558796d4cf6894">Isla Mágica</option>

</select>

9.- Crea y configura la ruta 4 en el servidor para guardar montañas rusas en la BBDD mediante el modelo Coaster, testea mediante Postman su validez. Renderiza a continuación en el cliente la vista new-coaster.hbs (ruta 3) cuyo formulario te permitirá crear nuevas montañas. No olvides modificar los atributos action y method del formulario, así como el atributo name de cada campo para conectarlo adecuadamente con el controlador. El ID del parque al que pertenece llegará al controlador identificado con el valor name de la lista de selección.

10.- Testea en la BBDD que el formulario crea montañas rusas sin incidencia.

11.- Crea la ruta 5 en el cliente y muestra la vista de coasters/index-coasters.hbs

12.- Adecúa la vista para que muestre todas las montañas rusas, y para que cada botón enlace con la URL /coasters/id\_de\_la\_montaña\_rusa. Ejemplo:

http://localhost:3000/coasters/5d408c5eff558796d4cf681e

13.- Crea la ruta 6 y en el cliente muestra la vista de coasters/coaster-details.hbs

14.- Adecúa la vista para que muestre toda la información de la montaña rusa, incluyendo el nombre y descripción del parque al que pertenece.

15.- Incluye en la vista un botón adicional que permita eliminar la montaña rusa, tal y como se solicita en la ruta 7. Recuerda que necesitarás un *endpoint* para ello.

16.- Incluye en la vista un botón adicional que permita editar la montaña rusa, tal y como se solicita en las rutas 8 y 9. Recuerda que necesitarás dos *endpoints* para ello, y recuerda que en la vista de edición (que no se incluye en el *repo*, créala) que tendrás que renderizar la lista de selección donde el usuario pueda seleccionar cualquiera de los parques existentes en el sistema.

¡A por ello!

Bonus I

Crea las vistas restantes del CRUD (índice de parques, detalles de parque...) y las acciones restantes del CRUD (editar parque, eliminar parque...)

Bonus II

Desarrolla **validación de formularios**: en caso de detectar una edición de montaña rusa con cualquier campo vacío, o si en la lista desplegable de parques está la primera opción “Seleccionar” activa, evita que se pueda crear el item y muestra un mensaje de error personalizado en la vista.