Ejercicio 04 - Node.js, Express, Mongoose

Este ejercicio hace referencia a los siguientes ejemplos:

04_node_mongoose_y_async (https://github.com/albertsgrc/curso-mean-jedi-2016Q2/tree/master/ejemplos/04_node_mongoose_y_async)

05_express (https://github.com/albertsgrc/curso-mean-jedi-2016Q2/tree/master/ejemplos/05_express)

06_express_routers (https://github.com/albertsgrc/curso-mean-jedi-2016Q2/tree/master/ejemplos/06_express_routers)

07_express_con_mongoose (https://github.com/albertsgrc/curso-mean-jedi-2016Q2/tree/master/ejemplos/07_express_con_mongoose)

Introducción

Los propietarios de Pccomponentes.com han contactado con nosotros para que les montemos una API para su tienda, con el fin de tener control sobre todo lo que ocurre en ella.

Ejercicio

Parte 1

####Esquemas:
Primero de todo definid los esquemas. De momentos tendremos dos esquemas: el de los productos y el de los usuarios.

El esquema de los productos es el siguiente:

```
var productosSchema = new Schema({
  modelo: {type: String, required: true},
  descripcion: {type: String},
  precio: {type: Number, default: 0},
  stock: {type: Number, default: 0}
});
```

El esquema de los usuarios es el siguiente:

```
var usuariosSchema = new Schema({
  nombre: {type: String, required: true},
  lista_de_la_compra: [{idProducto: String, cantidad: Number}]
});
```

####Rutas:
Vamos a tener 3 rutas en nuestra API.
Una pública para todo el mundo que entre inicialmente en nuestra página `localhost:8080/`

Una para usuarios que será `localhost:8080/usuario`

Y una para el administrador que será `localhost:8080/admin`

En la ruta pública deberemos:

- Ver todos los productos. GET: `localhost:8080/productos`
- Ver un producto. GET: `localhost:8080/producto/<id_del_producto>`
- Registrar un usuario. POST: `localhost:8080/registrarUsuario` (Pasaremos la información del usuario por el Body)

En la ruta para el usuario deberemos:

- Ver la información del propio usuario. GET: `localhost:8080/usuario/<id_del_usuario>/info`

En la ruta para el administrador debemos:

- Ver todos los usuarios. GET: `localhost:8080/admin/listaUsuarios`
- Añadir un producto. POST: `localhost:8080/admin/anadirProducto/` (Pasaremos la información del producto por el Body)
- Borrar un producto. DELETE: `localhost:8080/admin/borrarProducto/<id_del_producto>`
- Actualizar un producto. PATCH: `localhost:8080/admin/actualizarProducto/<id_del_producto>` (Pasaremos la información del producto por el Body)

Cada vez que añadamos o borremos tanto usuarios como productos, debemos indicar la acción que acabamos de hacer.
Por ejemplo: `res.status(200).send('Se ha borrado el producto');`

Parte 2

En esta segunda parte del ejercicio, vamos a implementar el sistema de compra para los usuarios.

####Rutas:
En la ruta para el usuario debemos:

- Comprar un producto. POST: `localhost:8080/usuario/<id_del_usuario>/compra/<id_del_producto>`.

Al realizar una compra, el stock del producto debe decrementar en 1 y se debe añadir a la lista de la compra del usuario.

Si el producto no tiene stock, debe salir un mensaje tal que `res.status(200).send('No hay stock del producto');` y se debe abortar la compra.

Si el producto no existe, debe salir un mensaje tal que `res.status(200).send('El producto que intentas comprar no existe');`.

Si ya ha comprado este producto anteriormente y lo vuelve a comprar, en la lista de la compra el atributo `cantidad` del producto debe incrementar en uno.

Extra

Si os sobra tiempo, podéis implementar un sistema de cupones de descuento que genere el administrador y los usuarios puedan utilizar al comprar sus productos. La implementación de éste sistema es totalmente libre.