Backend - Buy Buy Store

Table of contents

Usuarios del sistema

- Registrar usuarios (los usuarios se registran ellos mismos)
- Filtrar productos
- Comprar productos

Administrador

- Crear productos desde administrador
- Editar productos

- Los administradores deben ser creados desde base de datos o comando desde el server

Se deberá de hacer los endpoints requeridos para que se haga el registro de usuarios, control de ventas y otros endpoints requeridos para que el frontend funcione correctamente siguiendo el protocolo REST API.

Modelo de datos

Los datos a almacenar en este sistema será

user_buyer

- **id:** primary Key con un ID de su elección
- **username:** Usuario escrito con solo letras y números sin espacios y sin simbolos en todo minúscula.
- **email:** correo electrónico opcional (Blanco o Nullable o ambos)
- **password:** Contraseña del usuario

products

- **id:** primary Key con un ID de su elección
- **name:** nombre del producto
- **price:** valor de dinero del producto
- **currency:** moneda en la que se encuentra el producto en el ISO 4217 en Alpha-3 (es decir de 3 letras) valores obligatorios “USD” (Dolares), “PEN” (Sol peruano), “GBP” (Libra esterlina)
- **status:** estado actual del producto, indicará si está en venta, vendido o estará pronto a la venta, los estados obligatorios: “for_sale”, “sold”, “soon”
- **purchased_by:** Usuario Buyer que lo compró, valor opcional (Un producto no empieza comprado, Blanco, nullable o ambos)

- **purchased_at:** fecha en que fue comprado (recordar que los productos no empiezan comprados, considerar null o blank)
- **for_sale_at**: Fecha de cuando estará a la venta (este valor será colocado cuando esté en el estado soon, solo es obligatorio si llega al estado soon por ende deberia ser nullable o blank)
- **created_at**: Fecha de cuando fue creado el dato (desde que existe en base de datos, haciendo referencia a cuando fue publicado)

user_admin

- **id:** primary Key con un ID de su elección
- **username:** Usuario escrito con solo letras y números sin espacios y sin simbolos en todo minuscula. Este valor es único
- **email:** correo electronico
- **password:** Contraseña del usuario

Los tipos de datos, nombres de tablas, columnas e incluso si utilizar más variables son bajo su criterio propio. Hay otras tablas que el ORM generará de forma adicional asi como el manejador de sesiones y migraciones los cuales son validados.

Tener presente que el manejo de sesión es de su elección si usar JWT token, Token aleatorio o alguna otra estrategia.

Endpoints

Los endpoints a desarrollar deben de resolver los siguientes puntos:

Sin sesión

- **Registro de usuarios sin sesión iniciada** (es decir un usuario externo llega al sitio y se registra) con únicamente usuario y contraseña. Tras registrarse NO se generará token de sesión, deberá el usuario luego realizar el login. **Datos requeridos** [usuario, contraseña]
- **Login de usuario buyer**. Si el usuario está mal escrito (es decir no es letras y números) deberá arrojar bad request e indicar que el usuario debe contener letras sin simbolos y números. Si el usuario no existe indicar que el usuario no existe, si el usuario existe y la contraseña es incorrecta indicar que la contraseña es incorrecta. **Datos requeridos** [usuario, contraseña]
- **Login de usuario Administrador**. Si el usuario está mal escrito (es decir no es letras y números) deberá arrojar bad request e indicar que el usuario debe contener letras sin simbolos y números. Si el usuario no existe indicar que el usuario no existe, si el usuario existe y la contraseña es incorrecta indicar que la contraseña es incorrecta. **Datos requeridos** [usuario, contraseña]

Sesión de usuario Admin

- **Crear producto en estado soon (pronto)**: el endpoint recibirá los valores de name, price, currency. El valor purchased by es colocado tras la compra de producto (es decir otro endpoint lo va a realizar), el valor de created at es colocado con la fecha del momento que es creado el dato, la fecha del día que sale a la venta (**for_sale_at**) es obligatorio en este caso, el status es colocado como **“soon”**. **Datos requeridos** [name, price, currency, for_sale_at]. La respuesta puede no tener datos
- **Crear producto en estado for_sale (en venta)**: con los valores de name, price, currency, status. El valor purchased by es colocado tras la compra de producto, el valor de created at es colocado con la fecha del momento que es creado el dato, la fecha del día que sale a la venta (**for_sale_at**) **NO** podrá ser recibido en este caso, el dato de status será colocado como “for_sale”. **Datos requeridos** [name, price, currency]. La respuesta puede no tener datos
- **Listar los usuarios**: Entrega todos los usuarios buyer con los valores username, email y id. **Datos a entregar** [username, email, id]

- **Listar todos los productos**: Debe de mostrar todos los datos de los productos y la información del usuario username y email del que lo compró, este recurso tendrá el filtro de status del producto, este es un parametro opcional el cual si no va mostrará TODOS los productos. **Datos a entregar** [id_product, name, price, currency, status, purchased_by, purchased_at, for_sale_at, created_at, username, email, id_user], **Datos de entrega opcionales** [status]

Los endpoints de usuario admin **deben de validar** que el usuario admin está **logeado**

Sesión de usuario Buyer

- **Listar productos para comprar y que estarán pronto a la venta**: los productos listará son aquellos que no están comprados (estado "**for_sale**") y aquellos que estarán a la venta (estado "soon"). **Datos a entregar** [id, name, price, currency, status, for_sale_at, created_at]
- **Listar productos comprados**: Los productos que el usuario de la sesión tiene comprado. **Datos a entregar** [id, name, price, currency, status, purchased_at, for_sale_at, created_at]
- **Comprar producto:** Dado un id de producto el usuario podrá comprar el producto si está en el estado "for_sale" tras la compra el status cambiará a "**sold**" y el **purchased_by** será el usuario de esa sesión **Datos requeridos** [id]

Los endpoints de usuario buyer **deben de validar** que el usuario Buyer está **logeado**

Elabore los endpoints que considere necesarios para suplir esas necesidades.

NOTA: Si los datos ingresados están en el formato incorrecto o no es valido deberá arrojar un Bad request y el mensaje del por que del error

Consideraciones

Finalidad del Backend

Recordar que este es un sistema endpoints, es decir, que va funcionar tipo API que ofrece recursos para que **su frontend en Angular los consuma**. En este proyecto no

se debe de configurar nada de templates ya que el tema de renderizado será para el sistema en el Frontend.

Almacenamiento del repositorio con git

El código debe de ser subido en un repositorio de Git con cualquier manejador de git (preferiblemente con GitHub), el manejo de los commits y del repositorio en sí mismo está bajo su propio criterio.

Gestión de la Base de datos

La base de datos es suficiente con que lo maneje local, para facilidad de la prueba puede hacerlo con SQL-Lite, sin embargo, el motor de base de datos a utilizar en el Proyecto de será de su preferencia, por lo cual deberá de hacer la configuración requerida para que funcione con el motor que haya elegido. **EL PROYECTO TIENE QUE FUNCIONAR LOCALMENTE** por ejemplo, si usa algún otro motor como Posgres, mysql, mongo una alternativza para funcionamiento local es usar **Docker.**

Endpoints

En la web los endpoints arrojan diferentes estados a razón de los datos que ingresan y de la operación algunos procedimientos en específicos puede generar diferentes status.

En estos endpoints debería considerarse también los casos error (Puntos extra), por ejemplo si se llama el endpoint sin parte de los datos requeridos (Los obligatorios) indique los campos y los errores.

Manejo del proyecto en el backend

Cada framework permite múltiples formas para gestionarse, sin embargo, tenga presente que esta prueba calificará puntos como el orden, manejo de estándares, performance del aplicativo (basado en el uso de variables y operaciones) que dependen mucho en como sea configurado el proyecto y de cómo maneje los endpoints y querysets.

Criterio de evaluación

El proyecto que va a desarrollar será evaluado los siguientes puntos

Puntos principales

- Manejo de Git
- Conocimiento en bases de datos
- Uso de estándares de programación (programación limpia, estructurada, documentación en código)
- Cumplir con el mínimo requerimiento de los endpoints

Puntos extra

- Manejo de excepciones (manejar los errores a los que está sujeto el endpoint)
- Eficacia del desarrollo
- Uso de estructuras de datos (enfocado a mejora de rendimiento)
- Validaciones
- Estándares web (HTTP Status siguiendo un estándar para los escenarios)
- Puede usar el Backend de su elección sin embargo, tendrá **puntos extra** si es con Nest JS o Express JS
- Información sobre How to Compile o ejecutar el proyecto (README)