ICB0006-UF3-UF4
PR01

Nombre del alumno

Practica entregable (Enunciado y entregable)

Guía para el alumno

El alumno debe de entregar la práctica enunciada en este documento antes del cierre programado en el calendario. La nota y corrección de la práctica se publicará en la plataforma en un plazo no superior a 10 días hábiles a contar desde la fecha límite de entrega.

Esta práctica corresponde a las UF3 y UF4, en cada apartado tendréis indicado a qué UF corresponde, si a una, otra o bien a las dos. Para el caso de optar solamente a una UF, seguid estas indicaciones.

Esta práctica se divide en dos ejercicios. Cada uno de estos ejercicios puede ser hecho en dos proyectos Laravel por separado, o bien integrados en uno solo. La base de datos será la misma para ambos.

En esta práctica se valorará (además de las funcionalidades especificadas) que el código esté limpio y correctamente formateado. Juntamente con la calidad del documento de prácticas entregado.

Puntuación: 1,5 UF3 y UF4
(+0,5) Clean code
(+0,5) Calidad del documento y capturas
(+0,5) Github (uso adecuado del Sistema de Control de Versiones)

Entregas:

Los entregables son un fichero zip o rar que contenga:

- Documento de práctica: Este mismo documento convertido a PDF e incluyendo las respuestas solicitadas, los dibujos y las capturas de pantallas indicadas en el mismo:

  - [CSnn]: son capturas de pantalla que demuestran el resultado de una práctica. Para realizar la captura se utilizará la tecla “imp pant” o equivalente del teclado, y luego al final de documento y dentro de la página en blanco habilitada para cada captura, se realiza el “pegado” o “paste”. Si fuera necesario se ajustará el tamaño.

  - Para realizar las capturas de cada una de las pruebas solicitadas, se deberá mostrar toda la información para poder replicar la prueba. Por ello se deberá mostrar:

    - El estado inicial junto con los valores a introducir
    - La respuesta del programa frente a la prueba

Ejemplo de entrega:

ICB0006-UF3-UF4-PR01-“username”.pdf “username” = nombre de usuario del alumno en la plataforma Ejemplo: ICB0006-UF3-UF4-PR01-raulgarciaflores.pdf

- Código fuente de la aplicación con el siguiente nombre:

ICB0006-UF3-UF4-PR01-“username” “username” = nombre de usuario del alumno en la plataforma Ejemplo: ICB0006-UF3-UF4-PR01-raulgarciaflores

- Código de back-up de la base de datos:

DBname_username.sql “username” = nombre de usuario del alumno en la plataforma Ejemplo: products_raulgarciaflores.sql

Ejercicio #1 Aplicación WEB/API con bbdd orientada a documentos

Los objetivos de este ejercicio es que te familiarices con los siguientes conceptos:

UF3:

- Las bases de datos orientadas a documentos
- Gestión de objetos JSON

UF4:

- La programación orientada a componentes
- Uso de componentes no visuales
- Persistencia de componentes

Para resolver este ejercicio se recomienda haber consolidado los conocimientos expuestos en: ICB0006-S06 y ICB0006-S07.

El objetivo de esta práctica es crear una aplicación (versión web y API), que conectará a una base de datos orientada a documentos.

Esto ejercicio se divide en pequeñas tareas, en cada una de ellas viene especificada su puntuación. Ten en cuenta que es imprescindible que las capturas de pantalla [CSnn] relacionadas estén correctamente realizadas para que se pueda tener en cuenta la puntuación del apartado correspondiente. Esto servirá para demostrar la evolución de la práctica.

Ejercicio #1 - Tarea #1 Generación de fichero JSON – **UF3 y UF4**

Esta primera tarea consistirá en crear la base de datos para un e-Commerce. Para ello sigue los siguientes pasos:

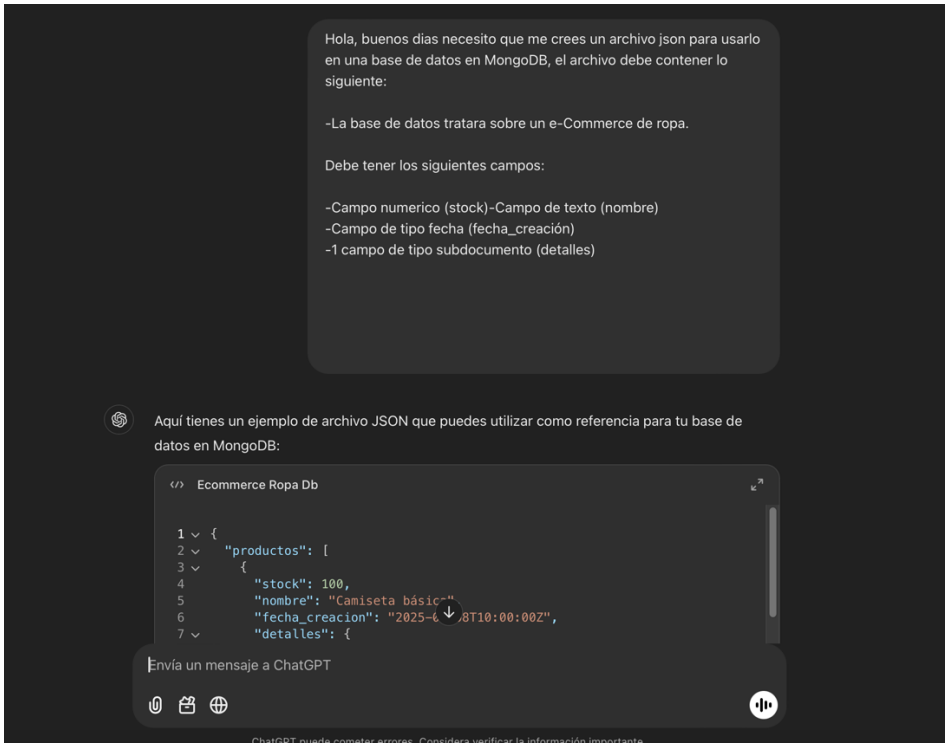
- Elije un tema del comercio de tu interés
- Piensa en varios campos que quieras que tenga. Como mínimo deberás tener:
    - 1 campo numérico (Stock)
    - 1 campo de tipo texto
    - 1 campo de tipo fecha
    - 1 campo de tipo subdocumento (array asociativo)

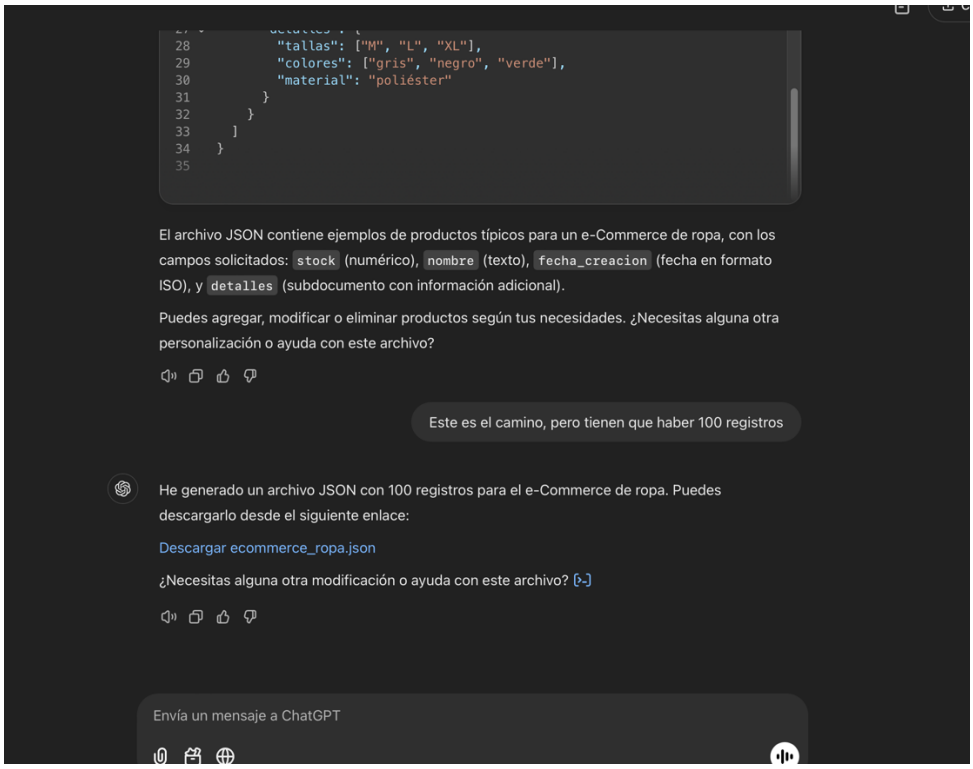
Usa un asistente de Inteligencia Artificial para generar un fichero JSON con 100 registros (también llamados documentos). Ayúdate de estas indicaciones:

- Deja claro al asistente qué quieres conseguir
- Indica al asistente la temática y qué campos quieres que te de
- Pásale un ejemplo
- Puede ser que solamente pueda darte un subgrupo, haz que genere una lista de 100 y pídele que te los dé a grupos.

	Hint: Puedes ayudarte de un asistente de Inteligencia Artificial, puedes pasarle un ejemplo del primer registro como: - https://openai.com/blog/chatgpt - https://www.bing.com/ - https://gemini.google.com/

[CS01] Capturas de pantalla del asistente con la conversación





La inteligencia artificial genera directamente un archivo para descargar.

Puntuación:
(+0,5) [CS01] UF3 y UF4

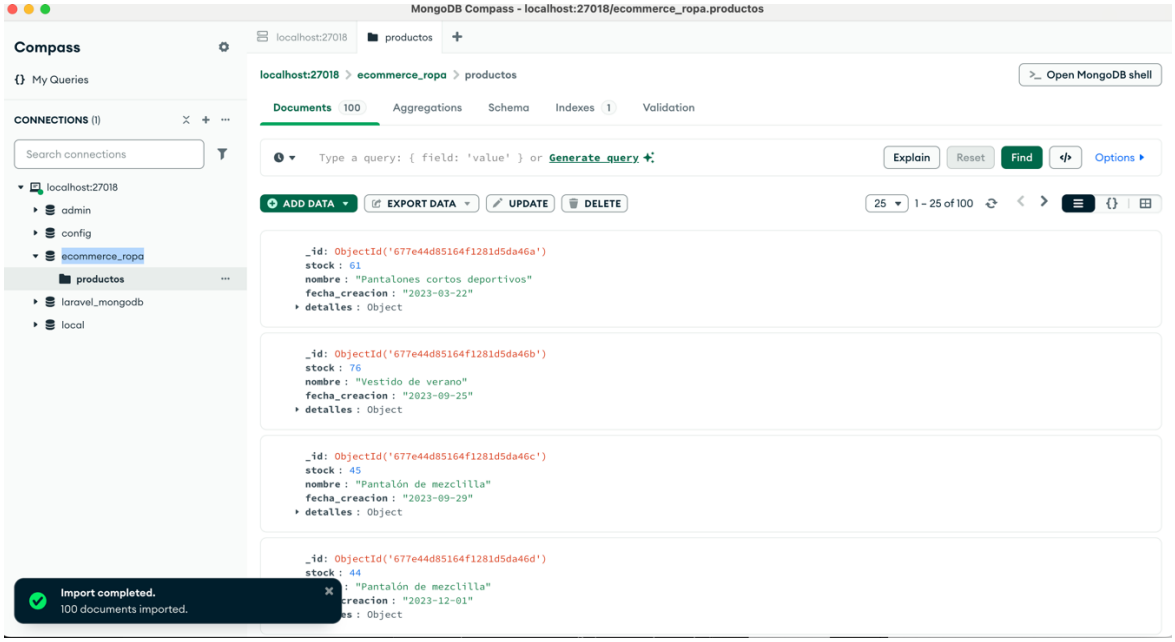
Ejercicio #1 - Tarea #2 Inserción de los datos

Crea una base de datos en MongoDB. Crea una nueva colección e inserta el fichero generado en la Tarea #1.

Dentro del cliente de MongoDB (Shell o bien Mongo Compass) crea las siguientes consultas:

- Filtro por un valor de texto a tu elección
- Filtro por una cantidad y una fecha a tu elección
- Ordenación por fecha y limitar el resultado a 5 registros (documentos)

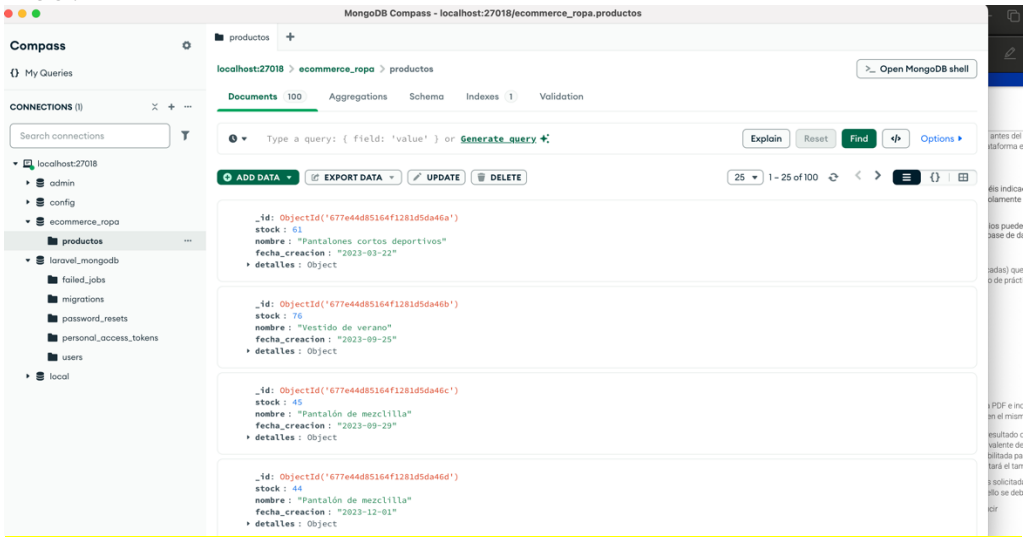
[CS02] Capturas de pantalla de los datos cargados.



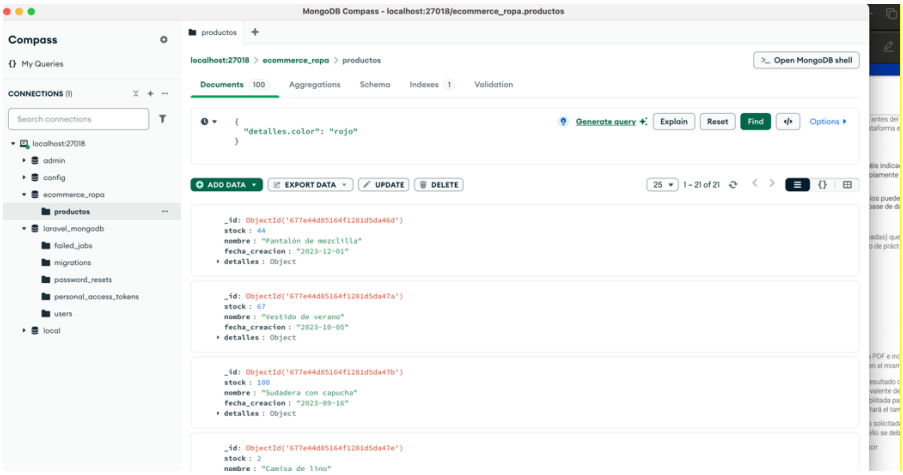
[CS03] Capturas de pantalla de las consultas en tu cliente MongoDB.

Buscar todos los productos cuyo color sea rojo:

Inicial:

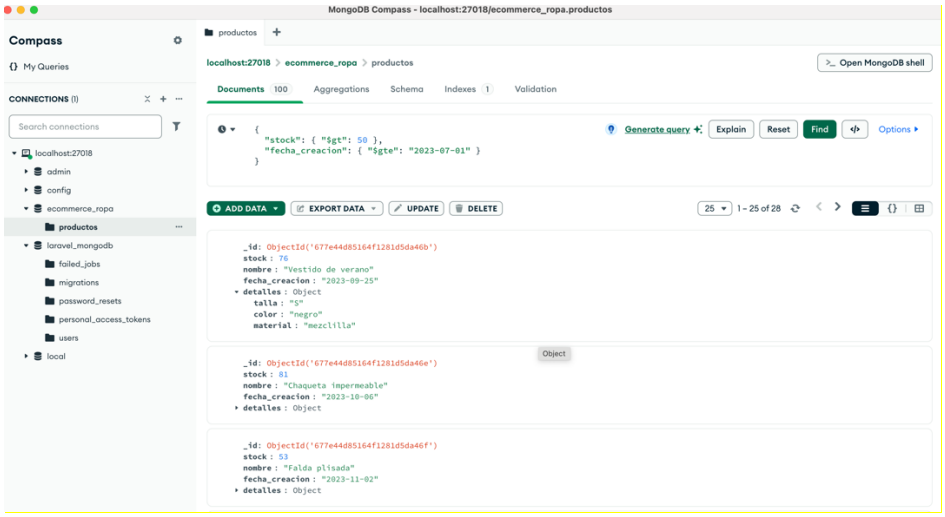


Después de la consulta:

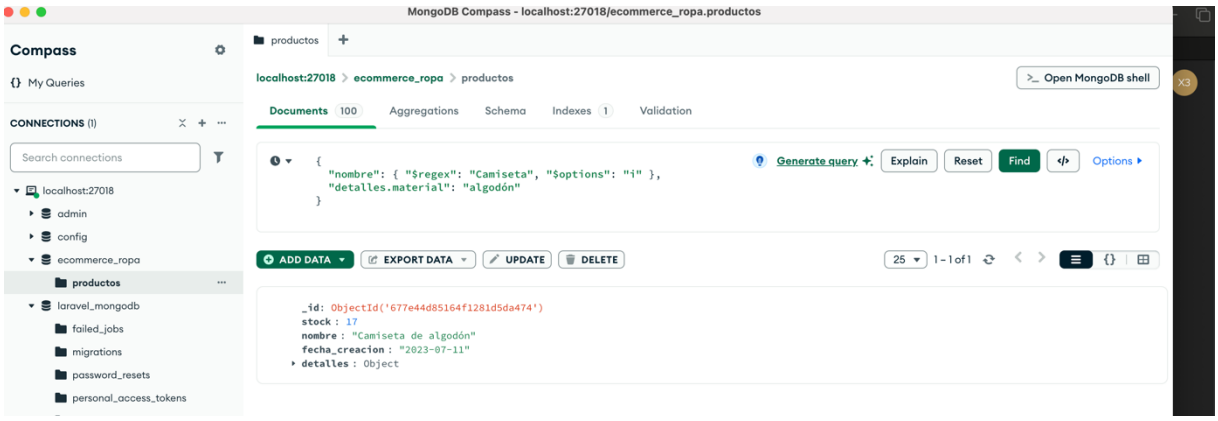


Filtro por cantidad stock > 50 y fecha fecha_creacion >= "2023-07-01":

(Utilizamos la captura anterior como inicial)



Busca productos cuyo nombre contenga la palabra camiseta y cuyo material en el campo detalles sea algodón.
(Utilizamos la captura anterior como inicial)



Puntuación:
(+1) [CS02] UF3 y UF4
(+1,5) [CS03] UF3

Ejercicio #1 - Tarea #3 Creación de la aplicación CRUD

De forma similar a lo que hiciste en la UF2, ahora vamos a realizar una aplicación CRUD que acceda a estos datos. Para simplificarlo **vamos a limitar las funcionalidades**. Simplemente se pedirá que el usuario pueda realizar las siguientes acciones:

- **Visualizar** todos los datos
- **Filtrar los datos** (elije el campo que prefieras para filtrar, preferiblemente alguna categoría o período de tiempo)
- Poder **augmentar o reducir el stock**

Crea lo siguiente:

Vista:
Crea una única vista que permita estas acciones. Deberá contener una tabla en html para ver los datos. Un formulario con un filtro, y un par de botones sobre cada registro para añadir o reducir el stock.

Controladores:
Crea los controladores necesarios para este ejercicio. Cómo mínimo vas a necesitar un controlador con 3 métodos:

- Devolver todos los datos
- Filtrar datos por un campo
- Modificar datos (suma / resta)

Añade el código necesario de Eloquent para realizar estas acciones.

Rutas:
Utiliza las rutas que creas necesarias.

[CS04] Capturas de pantalla con las pruebas de cada funcionalidad.

(en las capturas solo sacare la parte superior de la pagina ya que si debería hacer 3-4 capturas para sacar la pagina al completo):

ESTADO INICIAL:

Gestión de Productos

Buscar por nombre Filtrar

Nombre	Stock	Fecha Creación	Detalles	Acciones
Pantalones cortos deportivos	62	2023-03-22	Talla: XL, Color: azul, Material: poliéster	Aumentar Reducir
Vestido de verano	76	2023-09-25	Talla: S, Color: negro, Material: mezclilla	Aumentar Reducir
Pantalón de mezclilla	45	2023-09-29	Talla: S, Color: verde, Material: mezclilla	Aumentar Reducir
Pantalón de mezclilla	44	2023-12-01	Talla: XL, Color: rojo, Material: mezclilla	Aumentar Reducir
Chaqueta impermeable	82	2023-10-06	Talla: L, Color: gris, Material: poliéster	Aumentar Reducir
Falda plisada	53	2023-11-02	Talla: S, Color: azul, Material: poliéster	Aumentar Reducir
Camiseta de algodón	47	2023-11-08	Talla: L, Color: amarillo, Material: mezclilla	Aumentar Reducir

Filtrar por Vestido:

Gestión de Productos

vestido Filtrar

Nombre	Stock	Fecha Creación	Detalles	Acciones
Vestido de verano	76	2023-09-25	Talla: S, Color: negro, Material: mezclilla	Aumentar Reducir
Vestido de verano	27	2023-07-05	Talla: L, Color: amarillo, Material: algodón	Aumentar Reducir
Vestido de verano	5	2023-12-16	Talla: L, Color: blanco, Material: lana	Aumentar Reducir
Vestido de verano	76	2023-03-26	Talla: S, Color: gris, Material: lana	Aumentar Reducir
Vestido de verano	57	2023-11-01	Talla: XXL, Color: blanco, Material: algodón	Aumentar Reducir
Vestido de verano	67	2023-10-05	Talla: S, Color: rojo, Material: algodón	Aumentar Reducir
Vestido de verano	15	2023-04-08	Talla: XXL, Color: azul, Material: lino	Aumentar Reducir

Filtrar por pantalón:

Gestión de Productos

Buscar por nombre Filtrar

Nombre	Stock	Fecha Creación	Detalles	Acciones
Pantalones cortos deportivos	62	2023-03-22	Talla: XL, Color: azul, Material: poliéster	Aumentar Reducir
Pantalones cortos deportivos	2	2023-07-17	Talla: S, Color: blanco, Material: mezclilla	Aumentar Reducir
Pantalones cortos deportivos	48	2023-09-08	Talla: S, Color: gris, Material: algodón	Aumentar Reducir
Pantalones cortos deportivos	53	2023-05-28	Talla: XL, Color: negro, Material: algodón	Aumentar Reducir
Pantalones cortos deportivos	96	2023-06-25	Talla: S, Color: blanco, Material: mezclilla	Aumentar Reducir

Sumar stock: al clicar en aumentar (estaba en 61)

Gestión de Productos

Stock actualizado correctamente.

Buscar por nombre Filtrar

Nombre	Stock	Fecha Creación	Detalles	Acciones
Pantalones cortos deportivos	62	2023-03-22	Talla: XL, Color: azul, Material: poliéster	Aumentar Reducir

Reducir stock:

Gestión de Productos

Stock actualizado correctamente.

Buscar por nombre Filtrar

Nombre	Stock	Fecha Creación	Detalles	Acciones
Pantalones cortos deportivos	61	2023-03-22	Talla: XL, Color: azul, Material: poliéster	Aumentar Reducir

Puntuación:
(+1) Todos los datos [CS04] UF3
(+2) Datos filtrados [CS04] UF3
(+2,5) Editar datos [CS04] UF3

Ejercicio #1 - Tarea #4 Creación de la API-REST

Para este ejercicio, vamos a crear la API-REST que accederá a nuestra base de datos. Ayúdate de la demo realizada en los vídeos del curso y otra documentación propuesta.
Sigue los siguientes pasos y devuelve un fichero JSON con la información de la base de datos:

Controladores:
Crea los controladores necesarios para este ejercicio. Cómo mínimo vas a necesitar un controlador con 3 métodos:

- Devolver todos los datos
- Filtrar datos por un campo
- Edición de datos (suma / resta)

Añade el código necesario de Eloquent para realizar estas acciones.

Rutas:
Utiliza las rutas que creas necesarias. Recuerda en usar el fichero api.php en lugar del web.php.

Devuelve también un código de petición, y un mensaje en un JSON si no se han encontrado los datos.

[CS05] Capturas de pantalla con las pruebas de cada funcionalidad.

Puntuación:
(+1) Todos los datos [CS05] UF4
(+1) Datos filtrados [CS05] UF4

Ejercicio #2 Creación de eventos **UF4**

Los objetivos de este ejercicio es que te familiarices con:

- Creación de eventos
- Asociación de acciones a eventos

Para resolver este ejercicio se recomienda haber consolidado los conocimientos expuestos en: ICB0006-S07. Se recomienda haber terminado el Ejercicio #1 (parte UF4).

El objetivo de esta práctica es aplicar una serie de eventos a una aplicación de construida en Laravel. Esta aplicación será un **control de seguridad en un login** para un usuario presente en la base de datos. También se añadirá un **control sobre el stock disponible** en el e-commerce del ejercicio #1.

Este ejercicio se divide en 4 tareas, en cada una de ellas viene especificada su puntuación. Ten en cuenta que es imprescindible que las capturas de pantalla [CSnn] relacionadas estén correctamente realizadas para que se pueda tener en cuenta la puntuación del apartado correspondiente. Esto servirá para demostrar la evolución de la práctica.

Ejercicio #2 - Tarea #1 Preparación de la aplicación

En esta primera tarea partiremos de una aplicación de Laravel.

Crea un formulario en la vista "welcome" que sirva para un login (usuario y password), el botón del login deberá apuntar a un controleador llamado "loginController".

Ejercicio #2 - Tarea #2 Preparación de los datos

Usando Laravel, crea y rellena una **tabla de usuarios** para nuestro e-Commerce. Incluye los campos que consideres y como mínimo 1000 registros. Como mínimo inserta usuario y password.

Para ello tendrás que generar:

- Migraciones: para generar la estructura de la tabla
- El modelo Cliente
- Una factory para generar el estilo de datos (usa faker explicado en las videoconferencias)
- Un seeder para fijar en 1000 registros.

Aplica una modificación sobre la **tabla de productos** (del ejercicio anterior) y añade un campo booleano para indicar si está o no disponible. Haz los cambios que consideres.

[CS06] Captura de pantalla con algunos datos y los ficheros de migraciones, modelo, factory y seeder

Puntuación:

(+1,5) [CS06]

Ejercicio #2 - Tarea #2 Flujo

Una vez el usuario haya introducido unas credenciales, tendremos que verificar si son correctas. Para ello vamos a trabajar dentro del controlador "loginController" implementando la lógica del programa.

Lógica del login

Añade la lógica correspondiente en "loginController" para que verifique si las credenciales introducidas son correctas.

Tanto si el login ha sido exitoso como si no, muéstralo por pantalla.

Ya dentro de la home

Una vez dentro, el usuario debería de poder ver los productos del ejercicio 1. Aquí deberíamos de filtrar solamente los que estén disponibles y poder aumentar o reducir su stock.

Si no tienes la UF3, puedes poner un enlace a la API para un producto concreto.

[CS07] Captura de pantalla con las pruebas del login

[CS08] Captura de pantalla con la vista del resultado

Puntuación:

(+1) [CS07]

(+0,25) [CS08]

Ejercicio #2 - Tarea #3 Creación del evento y oyentes

Mediante los comandos Artisan crea los siguientes eventos:

- failedLoginEvent
- outOfStockEvent

Añade tres funciones listener: "failedLoginListener" y "OutOfStockListener". Las funcionalidades de estos listeners se detallan en la Tarea #4.
Haz los cambios que consideres pertinentes en el fichero "EventServiceProvider.php".

[CS09] Captura de pantalla con los nuevos ficheros

Puntuación:
(+0,25) [CS09]

Ejercicio #2 - Tarea #4 Funcionalidad de los oyentes

De los eventos creados en el ejercicio anterior, vamos a darles un objetivo.

- Fallo de login:
  El evento "failedLoginEvent" deberá llamarse siempre que un usuario entre una contraseña errónea.
  El listener "failedLoginListener" deberá registrar en el fichero log de Laravel un mensaje informando de la fecha y hora en la que se ha introducido el password erróneo.

- Fuera de stock
  El evento "outOfStockEvent" deberá llamarse siempre y cuando el stock de un producto sea = 0 o bien <= 0.
  El listener "OutOfStockListener" deberá registrar en el fichero log de Laravel un mensaje informando el producto que se ha terminado. También simulará un mensaje para el cliente (del modo que prefiráis). Por ejemplo, un mensaje por pantalla así:

  "Se envía email al cliente XXX informando que no hay stock del producto YYY"

[CS10] Captura de pantalla con las pruebas del listener "failedLoginListener"
[CS11] Captura de pantalla con las pruebas del listener "outofStockListener"

Puntuación:
(+1) [CS10]
(+1) [CS11]