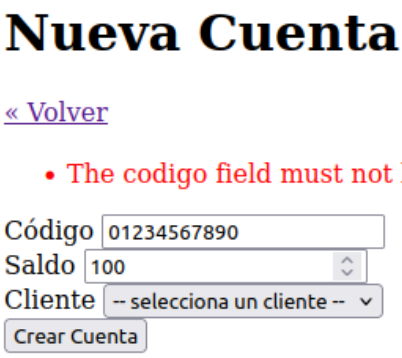
UD9. Pt3 – Modelo Vista Controlador. Uso framework (3)

- El objectivo de la práctica es añadir al proyecto desarrollado en las prácticas anteriores validaciones en el servidor.
  - Antes de crear o editar un registro, la aplicación comprobarà que los datos introducidos cumplan unos requisitos.
  - En caso contrario, se mostrará un mensaje de validación y no se persistirán los cambios.
- Se pide implementar las validaciones siguientes:
- **Cuenta**
  - Los campos código y saldo son obligatorios.
  - El campo código debe tener una longitud máxima de 10 caracteres.
  - El campo código es único (no puede haber 2 cuentas con el mismo código).



- **Cliente**
  - Los campos dni, nombre y apellidos son obligatorios.
  - El campo dni debe tener una longitud exactamente de 9 caracteres.
  - El campo dni es único (no puede haber 2 clientes con el mismo DNI).
  - La fecha de nacimiento debe ser como muy tarde el día de hoy.
- Usaremos las validaciones integradas en el framework Laravel:

https://laravel.com/docs/11.x/validation#quick-writing-the-validation-logic

- Haz una copia del proyecto de la UD9-Pt2, lo llamamos por ejemplo ud9_pt3
- Pasos a seguir:

1) Validaciones de la cuenta.
   - Añade al CuentaController la instrucción `$request->validate()` que vemos en la documentación oficial.
     - Tienes que hacerlo tanto en el método *new* como en el método *edit*.
     - Se tiene que hacer la validación sólo si accedemos por método HTTP POST, no por GET.

- Aplica las validaciones indicadas anteriormente para los campos *codigo* y *saldo*.
  - Tienes ejemplos de cómo hacerlo en la documentación de Laravel.
  - En el caso de la validación **unique** que tenemos que aplicar al campo *codigo*, hay que tener cuidado de cómo lo hacemos, para que no nos dé problemas en el formulario de edición.

Si no lo hacemos bien, nos obligará a cambiar el código de la cuenta cada vez que la editemos.

https://laravel.com/docs/11.x/validation#rule-unique

- El siguiente paso es añadir los mensajes de validación a los templates (formularios de creación y edición).
  - Hazlo tal y como indica la documentación del framework.
  - Es conveniente que los mensajes se vean de color rojo. Como de momento no estamos aplicando Bootstrap, puedes sencillamente añadir un CSS inline en el
- Haz una prueba de funcionamiento para comprobar que se aplican las validaciones requeridas y se muestran los mensajes.

2) Validaciones del cliente.

- Añade las validaciones del cliente indicadas anteriormente para los campos *dni*, *nombre*, *apellidos* y *fechaN*.
  - Se tienen que aplicar las validaciones tanto al crear como al editar un cliente.
- En la documentación oficial, hay una sección donde se indican todas las validaciones disponibles:

https://laravel.com/docs/11.x/validation#available-validation-rules

- Vemos, entre otras, las validaciones de fechas que ahora necesitamos.
- Para hacer la comparación con la fecha de hoy, tienes que concatenar código PHP en la string donde se define la validación.
  - Este código PHP nos permite obtener la fecha de hoy. Si creamos una instancia de la clase DateTime() de PHP, obtenemos un objeto que representa la fecha y hora actual.
  - Habrá que formatearlo como string, si no no nos dejará concatenar.

3) Personalizar los mensajes de validación.

- Se pide traducir los mensajes de validación, que originalmente están en inglés.
- En la documentación oficial nos dan información de cómo hacerlo:

https://laravel.com/docs/11.x/validation#specifying-custom-messages-in-language-files

- Es posible traducir los mensajes del fichero `lang/en/validation.php` directamente a nuestro idioma. Sin embargo, se propone hacer otra cosa:
  - Hacer editables los ficheros de configuración de idioma. Hay que ejecutar el comando:

```
php artisan lang:publish
```

  - Haz una copia de la carpeta **en** que encontramos en el directorio **lang**. Llama a la carpeta copiada **es**
  - En esta carpeta copiada podemos traducir al español los mensajes de validación que encontramos en el fichero **validation.php**
  - Se pide traducir los mensajes de todas las validaciones que hemos empleado en esta

práctica.

9. Por último, tendremos que configurar el español como idioma por defecto de la aplicación.
   - Hay que modificar la opción *locale* del fichero de configuración **config/app.php**

O bien (mejor) podemos cambiar la variable de entorno del fichero **.env** de donde se está tomando la configuración local.

- Puedes comprobar que ahora los mensajes de validación se visualizan traducidos al español.

Entrega de la práctica: carpeta comprimida con los ficheros de código fuente SIN LA CARPETA VENDOR.