PYTHON1: DESARROLLO BACKEND CON PYTHON, POO Y LA ERA DE LAS APLICACIONES WEB

7. FORMULARIOS Y MODELOS

FEDISAL | FORMÁS
CENTRO DE FORMACIÓN AVANZADA PARA PROFESIONALES

INCAF
Instituto Nacional de Capacitación y Formación

CONTENIDOS

1. Operaciones CRUD
2. Validaciones

FEDISAL | FORMÁS
CENTRO DE FORMACIÓN AVANZADA PARA PROFESIONALES

INCAF
Instituto Nacional de Capacitación y Formación

OPERACIONES CRUD

Para trabajar una aplicación Flask con bases de datos relacionales requiere instalar las siguientes librerías:

1. Flask.
2. SQLAlchemy.
3. Flask-SQLAlchemy.

OPERACIONES CRUD

Desarrollo de proyecto de ejemplo por instructor

Para ello es posible hacer uso del Patron MVC

QUE ES MVC?

Es un patrón de arquitectura de software que separa los datos y la lógica de negocio de una aplicación de la interfaz de usuario y el módulo encargado de gestionar los eventos y las comunicaciones. Para ello MVC propone la construcción de tres componentes distintos que son el **modelo**, la **vista** y el **controlador**.

Lo que es lo mismo, *Model, Views & Controllers*, si se prefiere en inglés

El principio fundamental del patrón MVC es definir una arquitectura con claras responsabilidades para diferenciar componentes.

- Wikipedia

¿POR QUÉ UTILIZAR EN MVC EN EL WEB?

Es una solución que va muy bien con los sitios *web*.

Cada *petición* en una página es la interacción de un usuario (entrada) con el sistema (el servidor *web* procesando los *scripts*). Asumiendo que se necesita mantener persistencia y que se necesita presentar la información en una variedad de maneras, el patrón MVC es una buena solución.

EL MODELO (MODEL)

El modelo es la porción que implementa la “Lógica del Negocio”.

Un modelo representa los datos o reglas de negocio así como el estado de la aplicación

Contiene toda la lógica que es específica de la aplicación. Además, incluye a las entidades de dominio. No posee conocimiento de los controladores o vistas que pueden utilizarlo.

Un modelo, por ejemplo, puede representar a un "usuario" en el sistema y manejar todas las operaciones que puede llevar a cabo un usuario.

LA VISTA (VIEW)

Presenta el 'modelo' (información y lógica de negocio) en un formato adecuado para interactuar (usualmente la interfaz de usuario) por tanto requiere de dicho 'modelo' la información que debe representar como salida.

La salida más común para aplicaciones web es el HTML. Podrían ser otras.

Esta representa los datos al usuario en un formato especificado como ya se dijo

EL CONTROLADOR (CONTROLLER)

Responde a eventos (usualmente acciones del usuario) e invoca peticiones al 'modelo' cuando se hace alguna solicitud sobre la información (por ejemplo, editar un registro en una base de datos). También puede enviar comandos a su 'vista' asociada si se solicita un cambio en la forma en que se presenta de 'modelo‘, por tanto se podría decir que el 'controlador' hace de intermediario entre la 'vista' y el 'modelo‘.

El controlador generalmente crea instancias de los modelos y utiliza métodos de esos modelos para conseguir los datos que se presentan a los usuarios, enviándolos a la vista correspondiente.

UN PATRÓN DE APLICACIONES WEB

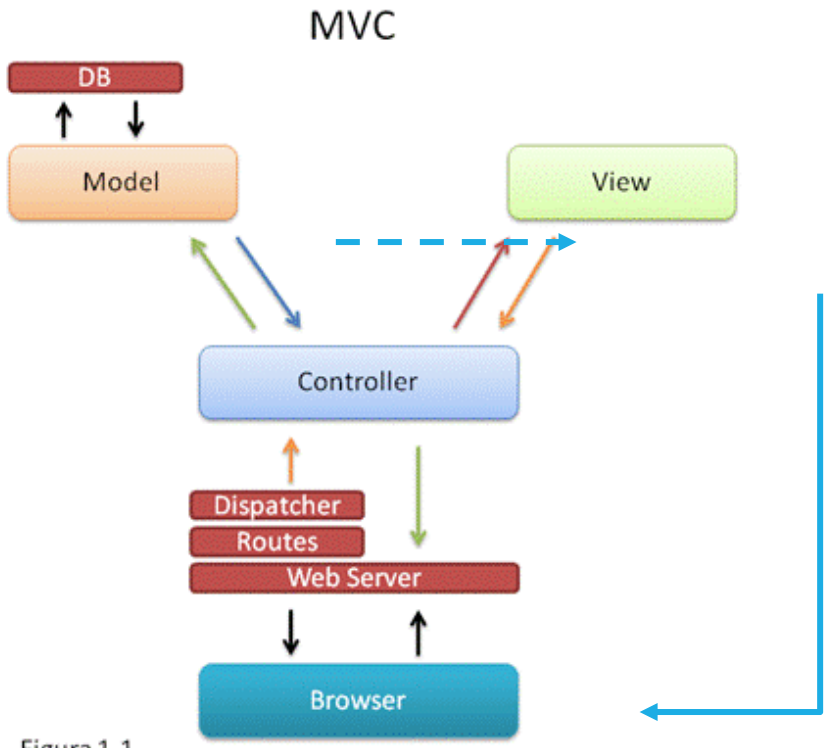

Figura 1-1

VALIDACIONES

Un validador simplemente toma una entrada, verifica que cumpla con algún criterio, como una longitud máxima para una cadena y devuelve. O, si la validación falla, genera un ValidationError. Este sistema es muy simple y flexible y le permite encadenar cualquier número de validadores en los campos.

```
if form.validate_on_submit():
        return redirect('/success')
```

```
{% if form.name.errors %}
    <ul class="errors">
    {% for error in form.name %}
        <li>{{ error }}</li>
    {% endfor %}
    </ul>
{% endif %}
```

MENSAJES FLASH

Flask proporciona una forma realmente sencilla de dar retroalimentación a un usuario con el sistema de flasheo. El sistema de flasheo básicamente permite grabar un mensaje al final de una solicitud y acceder a la siguiente solicitud y solo a la siguiente solicitud. Esto generalmente se combina con una plantilla de diseño que hace esto.

flash('You were successfully logged in')

```
<!doctype html>
<title>My Application</title>
{% with messages = get_flashed_messages() %}
  {% if messages %}
    <ul class=flashes>
    {% for message in messages %}
      <li>{{ message }}</li>
    {% endfor %}
    </ul>
  {% endif %}
{% endwith %}
{% block body %}{% endblock %}
```

PREGUNTAS?

- Gracias por su atención!!!

FEDISAL | FORMÁS CENTRO DE FORMACIÓN AVANZADA PARA PROFESIONALES

INCAF Instituto Nacional de Capacitación y Formación


thank you
merci
gracias
danke
謝謝
ngiyabonga
teşekkür ederim
tapadh leat
go raibh maith agat
obrigado
dziękuję
bedankt
спасибо
mauruuru
hvala
sukriya
kop khun krap
terima kasih
dakujem
arigatô
grazie
takk
dank je
mochchakkeram
merci
рахмат
Баярлалаа
faafetai lava
kiitos
dankie
dhanyavad
nanni
nandri
bayarlalaa
gràçie
enkosi
köszönöm
vinaka
спасибі
blagodaram
mersi
kia ora
barka
welalin
tack
spas
misaotra
matondo
paldies
grazzi
mahalo
djiere dieuf
tau
akun
dankon
ačiū
chnorakaloutioun
gratias ago
gràcies
sulpáy
taiku
sagolun
sobodi
děkuji
mèsi
didi madloba
najis tuke
ありがとう
tanemirt
rahmet
rahmat
kam sah hamnida
তোমাকে ধন্যবাদ
감사합니다
xièxie
ευχαριστώ
diolch
dhanyavadagalu
shukriya
mercé
мерси
trugarez
дякую
mamnun
chokrane
murakoze
хвала
asante
manana
obrigada
tenki