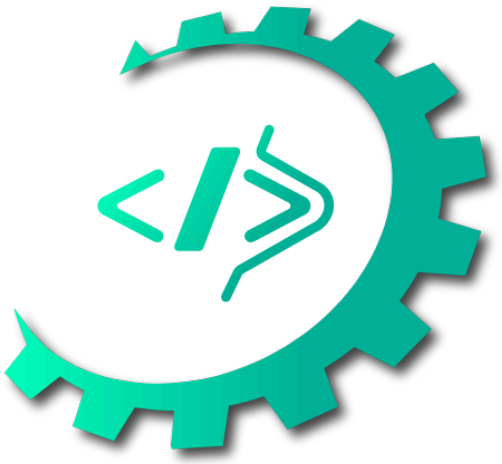
CUESTIONARIO DJANGO

Desarrollo de Aplicaciones Web



Nombre: Leonardo Borgo
Curso: 4°AO
Profesor: Ignacio García
Fecha de entrega: 10/07/2025

Índice

 # CUESTIONARIO DJANGO

1. ¿Qué es Django y por qué lo usaríamos?

Django es un framework web escrito en Python que nos permite crear aplicaciones web de forma rápida y eficiente. Fue creado originalmente en un ambiente de redacción de noticias, por lo que está diseñado para hacer que las tareas comunes de desarrollo web sean rápidas y fáciles.

Lo usaríamos porque nos ofrece rapidez de desarrollo ya que viene con muchas funcionalidades ya listas para usar, como autenticación de usuarios, panel de administración y manejo de formularios. También incluye protecciones de seguridad contra ataques comunes como SQL injection y cross-site scripting. Además, puede manejar aplicaciones desde pequeñas hasta muy grandes, tiene una gran comunidad y documentación excelente, y sirve para crear desde blogs simples hasta aplicaciones complejas como Instagram o Pinterest.

2. ¿Qué es el patrón MTV (Model-Template-View) en Django? Compará MTV con MVC

MTV (Model-Template-View) es la arquitectura que usa Django para organizar el código. El Model se encarga de los datos y la lógica de la base de datos, el Template define cómo se presenta la información al usuario (el HTML), y la View maneja la lógica de la aplicación y conecta el modelo con la plantilla.

Comparando MTV con MVC, ambos conceptos de Modelo son iguales ya que manejan datos. Sin embargo, en MTV la Vista maneja la lógica de negocio, mientras que en MVC la Vista se encarga de la presentación. Por otro lado, en MTV el Template maneja la presentación, mientras que en MVC el Controlador maneja la lógica.

3. ¿Qué entendemos por app en Django?

Una app en Django es un módulo independiente que se encarga de una funcionalidad específica dentro de tu proyecto web. Es como una pieza del rompecabezas que hace una tarea particular.

Por ejemplo, podemos tener una app para manejar usuarios (registro, login, perfiles), una app para un blog (artículos, comentarios), una app para una tienda online (productos, carrito de compras), o una app para contacto (formularios, mensajes).

Las apps son reutilizables, ya que puedes usar la misma app en diferentes proyectos. También son modulares porque cada una tiene su propósito específico, pueden comunicarse entre ellas y facilitan el mantenimiento del código.

4. ¿Qué es el flujo request-response en Django?

El flujo request-response es el proceso que sigue Django cuando un usuario visita una página web. Es como una conversación entre el navegador y el servidor.

El proceso comienza cuando el usuario hace clic en un enlace o escribe una URL, y el navegador envía una solicitud HTTP al servidor Django. Django recibe la solicitud y crea un objeto HttpRequest con toda la información como URL, datos enviados y headers. Luego Django revisa la URL y busca en los archivos de rutas qué vista debe manejar esa solicitud.

La vista correspondiente se ejecuta y puede consultar la base de datos, hacer cálculos o cualquier lógica necesaria. Durante el camino, pueden intervenir middlewares que agreguen funcionalidades como autenticación o logs. Finalmente, la vista crea un objeto HttpResponse con el contenido y lo envía de vuelta, para que el navegador reciba la respuesta y la muestre al usuario.

5. ¿Qué es el concepto de ORM (Object-Relational Mapping)?

El ORM es como un traductor entre tu código Python y la base de datos. Te permite trabajar con datos de la base de datos como si fueran objetos Python normales, sin necesidad de escribir SQL.

Funciona convirtiendo las tablas de la base de datos en clases Python, las filas de datos en objetos Python, y traduce tus operaciones en Python a consultas SQL automáticamente.

Las ventajas del ORM incluyen que es más fácil de usar porque no necesitas ser experto en SQL, es más seguro ya que previene ataques de SQL injection automáticamente, ofrece portabilidad porque puedes cambiar de base de datos sin cambiar tu código, y es más limpio porque el código es más fácil de leer y mantener.

6. ¿Qué son los templates en Django?

Los templates son archivos HTML que definen cómo se ve tu página web. Es como una plantilla que Django llena con datos reales.

Están escritos en HTML con etiquetas especiales de Django, permiten mostrar datos dinámicos, incluyen lógica básica, y pueden heredar de otros templates para no repetir código.

Las ventajas de los templates son que separas el diseño de la lógica, los diseñadores pueden trabajar en HTML sin tocar Python, son reutilizables y permiten crear sitios web dinámicos fácilmente.

7. Django at a Glance - Puntos Clave

Después de revisar la documentación oficial, los puntos más importantes son:

Django incluye un ORM que permite describir la estructura de tu base de datos usando código Python sin necesidad de escribir SQL. Una vez que defines tus modelos, Django automáticamente crea una API para interactuar con tus datos.

Django genera automáticamente un panel de administración profesional para gestionar tus datos. Permite crear URLs limpias y amigables sin extensiones como .php o .asp. Incluye un sistema de plantillas muy bueno, con herencia de templates para evitar repetir código.

La filosofía de Django es estar diseñado para ser rápido de desarrollar, mantenible y escalable.

8. ¿Cómo se instala Django?

Para instalar Django necesitas tener Python instalado, que puedes descargar de python.org. Primero tenes que verificar que Python esté instalado ejecutando python --version en la terminal.

Los pasos de instalación son: instalar Python desde la página oficial, verificar la instalación ejecutando python --version, instalar Django con pip install Django, y verificar la instalación con python -c "import django; print(django.get_version())".

Tenes tres opciones de instalación: la versión oficial que es la más estable y recomendada, la versión del sistema operativo a través del gestor de paquetes, o la versión de desarrollo para quienes quieren las últimas funcionalidades.

Para configuración de base de datos, Django incluye SQLite por defecto para proyectos simples, pero para proyectos más grandes puedes configurar PostgreSQL, MySQL u otras bases de datos.

Referencias:

Amazon Web Services. (2025). **¿Qué es Django?**
https://aws.amazon.com/es/what-is/django/

Gentile, A. (2025). **Basics of Django Model View Template (MVT) Architecture.**
https://angelogentileiii.medium.com/basics-of-django-model-view-template-mvt-architecture-8585aecffbf6

Python Plain English. (2025). **Understanding MTV vs MVC Design Patterns: A Guide for Developers.**
https://python.plainenglish.io/understanding-mtv-vs-mvc-design-patterns-a-guide-for-developers-28999bfd3109

Chirilova, A. (2025). **Apps in Django Concept.** CodeMentor.
https://www.codementor.io/@chirilovadrian360/apps-in-django-concept-free-samples-294vudyim5

Django Software Foundation. (2025). **Django at a glance.** Django Documentation.
https://docs.djangoproject.com/en/5.2/intro/overview/

Django Software Foundation. (2025). **Quick install guide**. Django Documentation.
https://docs.djangoproject.com/en/5.2/intro/install/