

Preguntas abp2 M7

Repositorio del modulo 7 : https://github.com/el-javier/m7_individuales

¿Qué es PostgreSQL y cuáles son sus ventajas en comparación con otras bases de datos relacionales?

PostgreSQL es un sistema de gestión de bases de datos relacional de código abierto y uno de los sistemas más populares en su categoría. Se centra en el cumplimiento y la extensibilidad de SQL y ofrece una amplia gama de características y funcionalidades que lo convierten en una opción sólida para aplicaciones y proyectos de diferentes tamaños y complejidades.

Algunas ventajas de postgres son:

Código abierto: PostgreSQL lo que significa que es gratuito y se puede modificar y utilizar en proyectos comerciales y no comerciales sin restricciones.

Cumplimiento de estándares: PostgreSQL cumple con los estándares ANSI SQL y ofrece una amplia gama de tipos de datos, funciones y operadores, lo que facilita la portabilidad de aplicaciones entre diferentes sistemas de bases de datos.

Escalabilidad y rendimiento: PostgreSQL puede manejar grandes volúmenes de datos y ofrece opciones de escalabilidad vertical y horizontal. Además, cuenta con herramientas de optimización de consultas y la capacidad de indexar eficientemente grandes conjuntos de datos, lo que contribuye a un rendimiento rápido y eficiente.

Características avanzadas: PostgreSQL ofrece funcionalidades avanzadas como soporte para tipos de datos personalizados, arrays, estructuras JSON y geoespaciales. También es compatible con transacciones ACID para garantizar la integridad de los datos.

Extensibilidad y flexibilidad: PostgreSQL permite a los usuarios crear funciones y tipos de datos personalizados, lo que brinda flexibilidad y posibilidades de personalización. También es compatible con una amplia gama de lenguajes de programación, facilitando la integración con diferentes entornos y frameworks.

Comunidad activa y soporte: PostgreSQL cuenta con una gran comunidad de desarrolladores y usuarios que contribuyen constantemente con mejoras y nuevas características. Esto garantiza un sólido soporte, documentación extensa y una amplia gama de recursos disponibles.

- ¿Cuál es la diferencia entre una llave primaria simple y una llave primaria compuesta en Django?

La diferencia principal entre una llave primaria simple y una llave primaria compuesta en Django radica en la cantidad de campos involucrados. Una llave primaria simple utiliza un solo campo como identificador único, mientras que una llave primaria compuesta involucra múltiples campos que trabajan en conjunto para crear un identificador único.

- ¿Cuál es el propósito de las operaciones CRUD en el desarrollo de aplicaciones web y cómo se implementan en Django?

Las operaciones CRUD (Create, Read, Update, Delete) se implementan utilizando el ORM de Django. El ORM mapea los modelos de Django a tablas en la base de datos y proporciona métodos y consultas para realizar operaciones CRUD de manera sencilla y segura. Los modelos de Django definen los campos y comportamientos de los objetos y actúan como una interfaz para interactuar con la base de datos.

el proposito de cda una es :

Create (Crear): Esta operación implica la creación de nuevos registros o elementos en la base de datos. En Django, se implementa utilizando el método create() del modelo correspondiente.

Read (Leer): Esta operación implica la lectura y recuperación de datos de la base de datos. En Django, se implementa utilizando consultas a través del ORM (Object-Relational Mapping). El ORM de Django proporciona métodos como all(), filter(), get(), entre otros, que permiten recuperar objetos del modelo según los criterios definidos.

Update (Actualizar): Esta operación implica la modificación de registros existentes en la base de datos. En Django, se implementa recuperando el objeto que se desea actualizar, modificando sus atributos y luego llamando al método save() para guardar los cambios en la base de datos.

Delete (Eliminar): Esta operación implica la eliminación de registros de la base de datos. En Django, se implementa llamando al método delete() en el objeto que se desea eliminar. Esto eliminará el registro correspondiente de la base de datos.

- ¿Qué herramientas o componentes adicionales de Django utilizarías para mejorar el rendimiento y la seguridad de una aplicación web?

Algunas herramientas podrían ser

Caché de página y caché de consultas: Caché de página y la caché de consultas de Django para almacenar en memoria resultados de vistas y consultas frecuentes, lo que reduce la carga en la base de datos y mejora el rendimiento.

Servidores de caché externos: Uso de servidores de caché externos como Memcached o Redis para almacenar en caché resultados a nivel de aplicación y mejorar aún más el rendimiento.

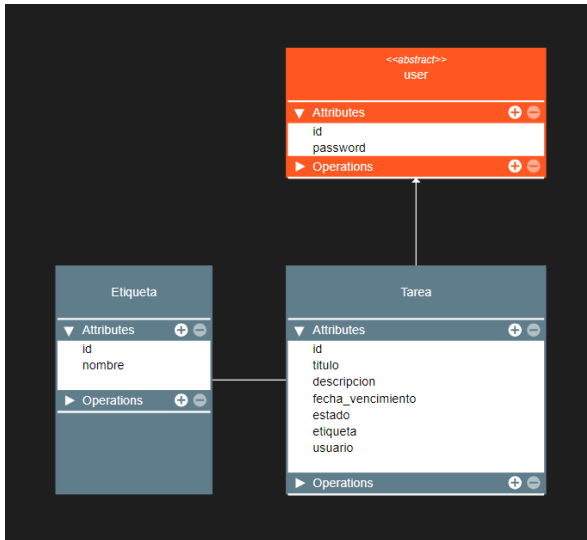
Indexación y optimización de consultas: Asegurar que las tablas estén debidamente indexadas y utiliza el ORM de Django para optimizar las consultas y reducir la cantidad de consultas realizadas.

Middleware de seguridad: Habilita el middleware de seguridad de Django para proteger la aplicación contra ataques de CSRF, configurar encabezados de seguridad y prevenir clics falsos.

Autenticación y autorización: Para asegurar que solo los usuarios autorizados puedan acceder a partes específicas de la aplicación. Considera la implementación de autenticación de dos factores y bibliotecas adicionales para fortalecer la seguridad de la autenticación.

SSL/TLS: Para cifrar las comunicaciones entre el servidor y los clientes y proteger la confidencialidad de los datos transmitidos.

Realiza modelo de datos en el que se puedan almacenar los datos gestionará en su aplicación.



Codifica todos los modelos necesarios para la construcción de su aplicación, considerando la definición de las llaves primarias y las relaciones entre las entidades.

En modelo incluye 2 clases , etiqueta y tarea

en la etiqueta está:

id y nombre

y en las tareas todo el registro de estados relacionados con la clase etiqueta :

id: identificador de la tarea.

título: título de la tarea.

descripción: descripción de la tarea.

fecha_vencimiento: fecha de vencimiento de la tarea.

estado: estado actual de la tarea.

etiqueta: relación con la entidad

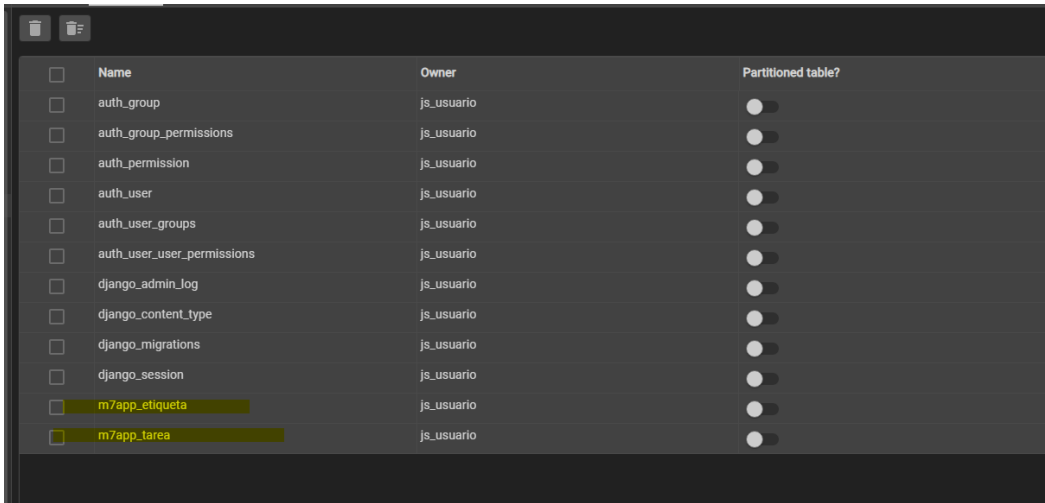
usuario: relación con la entidad User de Django. Una tarea está asociada con un usuario.

- No olvide realizar la migración antes de continuar.

```
(venv) PS C:\Users\W520\0 - CURS0\0 FULLSTACK PYTHON\E individual\M6\GITHUB\m7_individuales\ind_m7> python manage.py makemigrations
• Migrations for 'm7app':
  m7app\migrations\0002_etiqueta_tarea_delete_post_delete_user.py
    - Create model Etiqueta
    - Create model Tarea
    - Delete model Post
    - Delete model User
(venv) PS C:\Users\W520\0 - CURS0\0 FULLSTACK PYTHON\E individual\M6\GITHUB\m7_individuales\ind_m7> python manage.py migrate
• Operations to perform:
  Apply all migrations: admin, auth, contenttypes, m7app, sessions
Running migrations:
  Applying m7app.0002_etiqueta_tarea_delete_post_delete_user... OK
(venv) PS C:\Users\W520\0 - CURS0\0 FULLSTACK PYTHON\E individual\M6\GITHUB\m7_individuales\ind_m7> python manage.py migrate[]
```

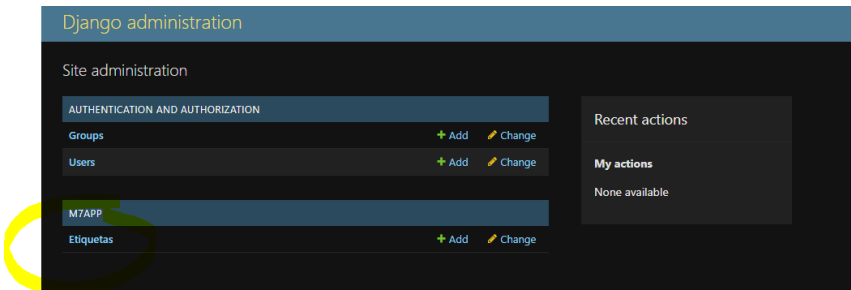
había hecho otro modelo antes por eso la migración incluye borrar lo anterior

En postgres



<input type="checkbox"/>	Name	Owner	Partitioned table?
<input type="checkbox"/>	auth_group	js_usuario	<input type="checkbox"/>
<input type="checkbox"/>	auth_group_permissions	js_usuario	<input type="checkbox"/>
<input type="checkbox"/>	auth_permission	js_usuario	<input type="checkbox"/>
<input type="checkbox"/>	auth_user	js_usuario	<input type="checkbox"/>
<input type="checkbox"/>	auth_user_groups	js_usuario	<input type="checkbox"/>
<input type="checkbox"/>	auth_user_user_permissions	js_usuario	<input type="checkbox"/>
<input type="checkbox"/>	django_admin_log	js_usuario	<input type="checkbox"/>
<input type="checkbox"/>	django_content_type	js_usuario	<input type="checkbox"/>
<input type="checkbox"/>	django_migrations	js_usuario	<input type="checkbox"/>
<input type="checkbox"/>	django_session	js_usuario	<input type="checkbox"/>
<input type="checkbox"/>	m7app_etiqueta	js_usuario	<input type="checkbox"/>
<input type="checkbox"/>	m7app_tarea	js_usuario	<input type="checkbox"/>

- Habilitar la gestión del modelo Etiqueta para que pueda ser gestionada desde la administración de Django.



con las claves de acceso

usuario : admin

pass : 1z2x3c4v.