WEB APPLICATIONS

Salas figueroa Jesus Nahataen

Universidad Tecnológica de Tijuana.
TSU. Tecnologías de la información.
Entornos virtuales y negocios digitales.
Docente: DR. Parra Galviz Ray Burnett.
Email: 0322103855@ut-tijuana.edu.mx

I. INTRODUCCIÓN

This document will cover a series of activities on the managers

Managers

A Manager is the interface through which database query operations are provided to Django models. At least one Manager exists for every model in a Django application.

The way Manager classes work is documented in Making queries; this document specifically touches on model options that customize Manager behavior. is an essential part of the Object-Relational Mapping (ORM) system. Managers are Python classes used to interact with a database table. They provide a high-level API for querying and manipulating database records. [1]

Managers name

By default, Django adds a Manager with the name objects to every Django model class. However, if you want to use objects as a field name, or if you want to use a name other than objects for the Manager, you can rename it on a per-model basis. To rename the Manager for a given class, define a class attribute of type models.Manager() on that model [1]

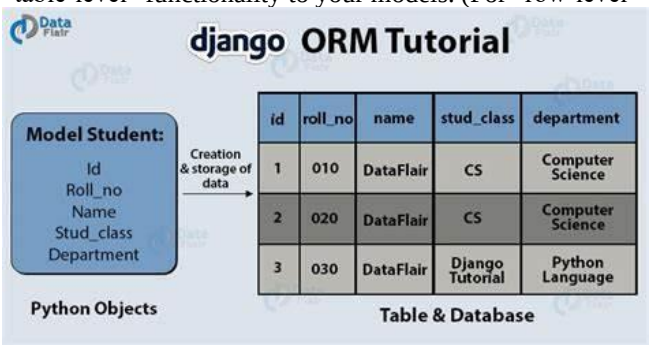
Custom managers

You can use a custom Manager in a particular model by extending the base Manager class and instantiating your custom Manager in your model. [1]

There are two reasons you might want to customize a Manager: to add extra Manager methods, and/or to modify the initial QuerySet the Manager returns. [1]

Adding extra manager methods

Adding extra Manager methods is the preferred way to add "table-level" functionality to your models. (For "row-level"



functionality – i.e., functions that act on a single instance of a model object – use Model methods, not custom Manager methods.) [1]

Conclusion

In conclusion, this document covers web architecture, its advantages, types, the history of the web and its versions, and provides a plan for developing web applications. Additionally, it discusses the concept of Managers in Django, which are essential classes for interacting with databases in models. Managers in Django are used for querying and database operations and can be customized to suit project needs, allowing the addition of custom methods and modification of initial query sets.

[1] Django, "Django project," [Online]. Available: Django. (s. f.). Django Project. Recuperado 5 de noviembre de 2023, de https://docs.djangoproject.com/en/4.2/topics/templates/. [Accessed 05 11 2023].