**--Django—**

**1) What is Django Rest Framework?**

The Django Rest Framework (DRF) is a framework that helps you quickly create RESTful APIs. They are ideal for web applications due to low bandwidth utilization.

**2) What do you use middleware for in Django?**

You use middleware for four different functions:

* Content Zipping
* Cross-site request forgery protection
* Session management
* Use authentication

**3) What does a URLs-config file contain?**

The URLs-config file in Django contains a list of URLs and mappings created to view those URLs' functions. The URLs can map to view functions, class-based views, and the URLs-config of other applications.

**4) What Does Django support multiple-column primary keys?**

No, Django supports only single-column primary keys.

**5) How can you see raw SQL queries running in Django?**

To begin, make sure that the DEBUG setting is set to True. If the setting is squared away, then type the following commands:

* from Django.db import connection
* connection.queries

**6) List several caching strategies supported by Django.**

Django supports these caching strategies:

* Database caching
* In-memory caching
* File System Caching
* Memory Cached

**7) What is a Query Set in the context of Django?**

Query Set is a collection of SQL queries. The command print(b.query) shows you the SQL query created from the Django filter call.

**8) What do you use django.test.Client class for?**

The Client class acts like a dummy web browser, enabling users to test views and interact with Django-powered applications programmatically. This is especially useful when performing integration testing.

**9) How to use file-based sessions?**

To use a file-based session, you must set the SESSION\_ENGINE settings to "Djangoo.contrib.sessions.backends.file".

**10) What is mixin?**

In Django, a mixin is a Python class that is inherited by another class to carry out extra functions. Classes that can be reused and scaled are mixins. A unique form of multiple inheritances is a mixin. Mixins are typically employed in two contexts:

* You wish to give a class several optional features.
* You wish to apply a specific feature to numerous classes.

**11) What is Difference between Django OneToOneField and ForeignKey Field?**

Both are among the most frequent sorts of fields in Django. The sole difference between these two is that the ForeignKey field includes an on\_delete option in addition to a model's class because it is used for many-to-one relationships, whilst the OneToOneField only handles one-to-one relationships and requires only the model's class.

**12) What are the views in Django?**

Django views are Python functions similar to HTML documents and accept HTTP requests and return HTTP responses.

**13) What is Django ORM?**

Django ORM is a database abstraction API using which we can interact with its database models i.e., perform actions like add, delete, modify and query objects.

**14) Define static files and explain their uses.**

The word "static files" refers to files in a web app that do not change, such as CSS, JavaScript, or pictures. They remain still. Static files are served up by the local Django web server for local development, and minimal configuration is required.

**15) What are Django URLs?**

In Django, URLs serve as the front door to your online application. In urls.py, you can configure how Django routing works.

**16) What's the use of a session framework?**

The session structure enables per-site-visitor storage and retrieval of any type of data. It abstracts the sending and receiving of cookies and keeps data on the server side.

**17) Difference between MVC and MVT design patterns?**

* Model and View are both driven by the controller in MVC, whereas Views in MVT are used to receive HTTP requests and return HTTP responses.
* We must write all of the control-specific code in MVC whereas, we must write all of the control-specific code in MVC.
* MVC is Highly coupled whereas, MVT is loosely coupled.
* In MVC, it is difficult to modify whereas, Modification is easy in MVT.
* MVC is suitable for large applications, but MVT is suitable for both small and large applications.
* MVC does not involve any URL mapping, whereas in MVT URL pattern mapping takes place.
* Flow is clear and easy to understand, whereas MVT is sometimes harder to understand.