## **What is Django? Elaborate some technical features.**

Django is a web development framework that was developed in a fast-paced newsroom. It is a free and open-source framework that was named after Django Reinhardt who was a jazz guitarist from the 1930s. Django is maintained by a non-profit organization called the Django Software Foundation. The main goal of Django is to enable Web Development quickly and with ease.

Some of the technical features of Django include:

* Admin Interface
* Code Reusability
* Security Features
* ORM
* A huge number of third-party applications

There are many features which Django community has been developing over the years and therefore it’s called “Batteries-Included” framework, as it has lots of features built-in which otherwise would be time-consuming and expensive to make.

## **What are the advantages of using Django?**

* Django’s stack is loosely coupled with tight cohesion
* The Django apps make use of very less code
* Allows quick development of websites
* Follows the DRY or the Don’t Repeat Yourself Principle which means, one concept or a piece of data should live in just one place
* Consistent at low as well as high levels
* SQL statemants are not executed too many times and are optimized internally
* Easy database migration
* Can easily drop into raw SQL whenever required
* Is written in Python, one of the most popular programming languages available today

## **Explain Django architecture.**

Django follows the MVT or Model View Template architecture whcih is based on the MVC or Model View Controller architecture. The main difference between these two is that Django itself takes care of the controller part.

According to Django, the ‘view’ basically describes the data presented to the user. It *does not deal with how* *the data looks* but rather *what the data* *actually is*. Views are basically callback functions for the specified URL’s and these callback functions describe which data is presented.

The ‘templates’ on the other hand deal with the presentation of data, thereby, separating the content from its presentation. In Django, views delegate to the templates to present the data.

The ‘controller’ here is Django itself which sends the request to the appropriate view in accordance with the specified URL. This is why Django is referred to as MTV rather than MVC architecture.

## **What is Django Admin Interface?**

Django comes with a fully customizable in-built admin interface, which lets us see and make changes to all the data in the database of registered apps and models. To use a database table with the admin interface, we need to register the model in the admin.py file

## **In Django, what’s the difference between a project and an app?**

The project covers the entire application, while an app is a module or application within the project that deals with one dedicated requirement. So, a project consists of several apps, while an app features in multiple projects.

## **What happens when a typical Django website gets a request? Explain.**

When a user enters a URL in the browser the same request is received by the Django Server. The server then looks for the match of the requested URL in its URL-config and if the URL matches, it returns the corresponding view function. It will then request the data from the Model of that application, if any data is required and pass it to the corresponding template which is then rendered in the browser, otherwise, a 404 error is returned.

## **What is a Model in Django and what is the Model class?**

A Model in Django is a python class which derives from Model class that imports from the django.db.models library. A model is used in Django to represent a table in a database.

## **Explain Django’s Request/Response Cycle.**

In the Request/Response Cycle, first, a request is received by the Django server. The server then looks for a matching URL in the URL patterns defined for the project. If the server can’t find a matching URL, it produces a 404-status code. If the URL matches, it executes the corresponding code in the view file associated with the URL and sends a response.

## **What is Django ORM?**

Django ORM is one of the special feature-rich tools in Django. ORM is an acronym for Object-Relational Mapper. This ORM enables a developer to interact with a database in a pythonic way.

## **What are migrations in Django?**

Migrations are Django’s way of propagating changes you make to your models (adding a field, deleting a model, etc.) into your database schema. They’re designed to be mostly automatic, but you’ll need to know when to make migrations, when to run them, and the common problems you might run into.

There are several commands which you will use to interact with migrations and Django’s handling of database schema:

* **migrate**, which is responsible for applying and unapplying migrations.
* **makemigrations,** which is responsible for creating new migrations based on the changes you have made to your models.
* **sqlmigrate**, which displays the SQL statements for a migration.
* **showmigrations**, which lists a project’s migrations and their status.

## **What are Django's templates?**

Django templates render information in a designer-friendly format to present to the user. Using the Django Template Language (DTL), a user can generate HTML dynamically. Django templates consist of simple text files that can create any text-based format such as XML, CSV, and HTML.

## **What is a middleware in Django?**

A middleware is a layer in Django’s Request/Response processing pipeline. Each middleware is responsible for performing some specific functions on the request and/or response, such as caching, gzipping, etc

## **What is Django REST Framework?**

Django REST Framework (DRF) is a Django app and a framework that lets us create RESTful APIs rapidly. DRF is especially useful if we have an existing Django web application and we wish to quickly generate an API for it.