DJANGO

1. What is Django?

Django is a Full-stack web development framework that facilitates the creation and maintenance of high-quality Dynamic pages while also encouraging rapid development and a clean, pragmatic style. Django makes it easier to automate repeated operations, resulting in a more efficient development process with fewer lines of code.

2.Name some Companies that use Django.

Some companies that use the Django framework are Instagram, Mozilla, Disqus, Bitbucket, Nextdoor, and Clubhouse.

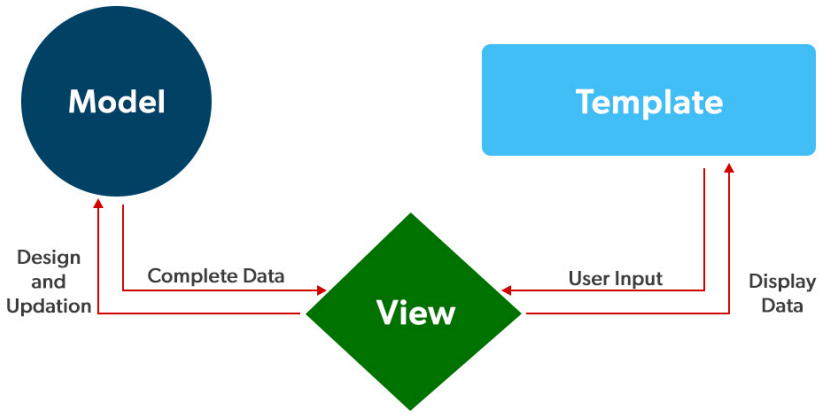
3.What is the difference between a Project and an App?

The main difference between a project and an app is that a project is defined as the entire application whereas, an app is part of the project that is self-sufficient to perform any task.

4. Explain Django's architecture.

The Model-View-Template also known as MVT architecture is used by Django. It is a software design pattern for developing a web application. The Django MVT Structure is made up of three parts:

- The model will serve as an interface for your data. It is in charge of data management. A database represents the logical data structure that supports the entire application such as MySql and Postgres.
- The View is the user interface, that renders a website page in your browser. HTML/CSS/Javascript and Jinja files are used to represent it.
- A template is made up of both static sections of the desired HTML output and specific syntax that describes how dynamic content will be included.



6. What are the Features of using Django?

- Flexible server arrangement,
- Model relation database
- Provide object-relational mapper
- Web templating system

- **Middleware class support**
- **Regex-based URL Dispatcher**
- **Unit testing framework**
- **Admin Interface**
- **Django is SEO optimized.**
- **In-build mitigation**
- **Easy inheritance**

7.Explain the Django project directory structure.

When you first start a Django project, it comes with some basic files like manage.py and view.py.

1. **init.py – It's an empty Python file. It is called when the package or one of its modules is imported. This file tells the Python interpreter that this directory is a package and that the presence of the __init.py_ file makes it a Python project.**
2. **manage.py – This file is used to interact with your project from the command line utility. with the help of this command, we can manage several commands such as:**
    1. **manage.py runserver**
    2. **manage.py makemigration**
    3. **manage.py migrate' etc**
3. **setting.py – It is the most important file in Django projects. It holds all the configuration values that your web app needs to**

work, i.e. pre-installed, apps, middleware, default database, API keys, and a bunch of other stuff.

4. views.py – The View shows the user the model's data. The view knows how to get to the data in the model, but it has no idea what that data represents or what the user may do with it.
5. urls.py – It is a universal resource locator which contains all the endpoints, we store all links of the project and functions to call it.
6. models.py – The Model represents the models of web applications in the form of classes, it contains no logic that describes how to present the data to a user.
7. wsgi.py – WSGI stands for Web Server Gateway Interface, This file is used for deploying the project in WSGI. It helps communication between your Django application and the web server. more...
8. admin.py – It is used to create a superuser Registering model, login, and use the web application.
9. app.py – It is a file that helps the user to include the application configuration for their app.

8. How do you create a Django project?

We can create a Django project with the help of the following command

```
django-admin startproject projectname
```

9. How do you create a Django app?

We can create a Django app with the help of the following command

python manage.py startapp appname

10. How do we start our development server?

We can start our development server with the help of the following command

python manage.py runserver

11. Importance of virtual environment setup for Django.

A virtual environment allows you to establish separate dependencies of the different projects by creating an isolated environment that isn't related to each other and can be quickly enabled and deactivated when you're done.

It is not necessary to use a virtual environment without a virtual environment we can work with Django projects. However, using virtualenv is thought to be the ideal practice. Because it eliminates dependencies and conflicts.

12.Give a brief about the Django admin interface.

Django provides us with a default admin interface that is very helpful for creating, reading, updating, and deleting model objects that allow the user to do administrative tasks. It reads a set of data that explains and provides information about data from the model

in order to create a quick interface where the user can alter the application's contents. This is an in-built module.

13. What are Django URLs?

The routing of a website is determined by its URLs. In the program, we build a python module or a file called urls.py. The navigation of your website is determined by this file. When a user visits a given URL route in the browser, the URLs in the urls.py file are compared. The user then receives the response for the requested URL after retrieving a corresponding view method.

14. What are the views of Django?

Django views are an important feature of the MVT Structure. This is a function or class that takes a web request and delivers a Web response, according to the Django script. This response could be an HTML template, a content of a Web page, an XML document, a PDF, or images. the view is a part of the user interface that renders the HTML/CSS/Javascript in your Template files into what you see in your browser when we render a web page.

15. What are the models in Django?

Model is a built-in feature of Django that contain a definitive source of information such as behavior and essential fields about the data we are storing. It allows us to build tables, fields, and constraints and organize tables into models. Generally, each model maps to a single database table. To use Django Models, you'll need a project and an app to work with. Also, Django makes use of SQL to access the database. SQL is a complex language with many

queries for generating, removing, updating, and other database-related tasks.

16. What do the following commands do?

- **python manage.py makemigrations**
- **python manage.py migrate**

The makemigration command scans the model in your application and generates a new set of migrations based on the model file modifications. This command generates the SQL statement, and when we run it, we obtain a new migration file. After running this command, no tables will be created in the database.
Running migrate command helps us to make these modifications to our database. The migrate command runs the SQL instructions (produced by makemigrations) and applies the database changes. After running this command, tables will be created.

17. Define static files and explain their uses.

Static files are used to save files such as CSS, JavaScript, pictures, and other types of static files. We keep them in distinct folders, for as the js folder, which has all of the JavaScript files, and the images folder, which contains all of the images. These files are kept in the static subfolder of the project app. Django provides django.contrib.staticfiles which helps us to manage static files.
There are different uses for static files:

- **It clarifies the use of file methods and attributes.**
- **It is platform-independent in many ways, whereas generic files are not.**
- **Subclasses can be used to extend it.**

18.What are templates in the Django language?

A Django template is a text document that is used to give a front and a layout for our website. It is the third and most significant aspect of Django's MVT Structure. In Django, a template is an HTML file that contains HTML, CSS, and Javascript. The Django framework efficiently manages and generates dynamically generated HTML web pages for end-user viewing. Django is mostly a backend framework, thus we use templates to give a layout for our website. There are two ways to incorporate the template into our website.

- We can utilize a single template directory that will be distributed throughout the project.
- We can make a separate template directory for each app in our project.

19.What does the settings.py file do?

settings.py is a core file in Django projects. It holds all the configuration values that your web app needs to work; database settings, logging configuration, where to find static files, API keys if you work with external APIs, and a bunch of other stuff.

21. 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 the View and Template components in MVT.
- 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.

22.What is Django ORM?

ORM which is also known as the object relation model enables us to interact with our database. It allows us to add, delete, change, and query objects (Object Relational Mapper). Django uses a database abstraction API called ORM to interface Viewed with its database models, to use Django object relation Models, we must first have a project and an app running. Models can be created in app/models.py after an app has been started. The Django ORM may be accessed by running the following command in our project directory.

```
python manage.py shell
```

This opens a Python console where we may add objects, retrieve objects, modify existing items, and delete objects.

23.What is Superuser?

superuser is the most powerful user with permission to create, read, delete, and update on the admin page which includes model records and another user. Users of Django have access to an Admin Panel. Before using this feature, you must have to migrate your project; otherwise, the superuser database will not be created. To create a superuser, first, reach the same directory, and run the following command

```
python manage.py createsuperuser
```

24.What is Jinja templating?

Jinja is also known as jinja2, which is the most recent version. It's a template engine that allows you to make HTML, XML, and other markup types. Jinja2 is valuable since it features a templating tag syntax and because the project has been extracted as a standalone open-source project that may be utilized as a dependency by other code libraries. Some of its features are:

- HTML Escaping – It provides automatic HTML Escaping as <, >, & characters have special values in templates and if using a regular text, these symbols can lead to XSS Attacks which Jinja deals with automatically.
- Sandbox Execution – This is a framework for automating the testing process in a sandbox (or protected) environment.
- Template Inheritance
- Produces HTML templates far more quickly than the default engine.
- When compared to the default engine, it is easier to debug.

25.What do you mean by the csrf_token?

Cross-Site Request Forgery (CSRF) is one of the most serious vulnerabilities, and it can be used to do everything from changing a user's information without their knowledge to obtaining full control of their account. To prevent malicious attacks, Django provides a per cent token per cent tag {% csrf_token %} that is implemented within the form. When generating the page on the server, it generates a token and ensures that any requests coming back in are cross-checked against this token. The token is not included in the inbound requests, thus they are not executed.

26.Explain the use of Middlewares in Django.

Middleware is a lightweight plugin in Django that is used to keep the application secure during request and response processing. The application's middleware is utilized to complete a task. Security, session, CSRF protection, and authentication are responsible for doing some specific functions. The application's security is maintained by the usage of the middleware component, AuthenticationMiddleware which is associated with user requests using sessions.

27.What are 'signals'?

Signals are used to take action in response to the modification or creation of a database entry. Its utilities help us to connect events with their action. we can create methods that will run a signal when it is called. For example, as soon as a new user instance is generated in the Database, one might want to create a profile instance. Generally, There are 3 types of signals.

1. pre_delete/post_delete: This signal is thrown before the remove() method is used to delete a model's instance.

2. pre_init/post_init: This signal is thrown before/after instantiating a model (__init__() method)
3. pre_save/post_save: This signal works before/after the method save().

28.What is Media Root?

Media root is used to upload user-generated content. We can serve user-uploaded media files locally, using the MEDIA_ROOT and MEDIA_URL settings. User-upload these files are referred to as Media or Media Files in Django. The first step is to include the code below in the settings.py file.

MEDIA_ROOT = os.path.join(BASE_DIR, 'media')
MEDIA_URL = '/media/'

MEDIA_ROOT: It is for the server path to store files in the computer.
MEDIA_URL: It is the referring URL for the browser to access the files over HTTP.

29.How you can include and inherit files in your application?

Using the extends tag in Templates, we can inherit our files in Django, The extends tag is used to inherit these templates. The syntax for inheriting these templates is given as:
{% extends 'template_name.html' %}
This syntax helps us to add all the elements of an HTML file into another without copy-pasting the entire code. Django templates not only allow us to pass variables from view to template, but they also provide some programming capabilities like loops, comments, and extensions.

30.How do you connect your Django Project to the database?

We need to configure our database in the settings.py file. By default, SQLite is mentioned there, and we need to change this setting accordingly like Postgres, MongoDB, and MySql.

31.Explain the caching strategies of Django. ?

Django has its own inbuilt caching system that allows us to store our dynamic pages. So that we don't have to request them again when we need them. The advantage of the Django Cache framework is that it allows us to cache data such as templates or the entire site. Django provides four different types of caching options, they are:

- per-site cache – It is the most straightforward to set up and caches your entire website.
- per-view cache – Individual views can be cached using the per-view cache.
- Template fragment caching allows you to cache only a portion of a template.
- low-level cache API – It can manually set, retrieve, and maintain particular objects in the cache using the low-level cache API.

32.Give the exception classes present in Django.

An exception is a rare occurrence that causes a program to fail. Django has its own exception classes to cope with this circumstance, and it also supports all fundamental Python exceptions. some of the exception classes are listed below:

- MultipleObjectsReturned – If just one item is anticipated but many objects are returned, this error is thrown by the query.

- **ViewDoesNotExist – When a requested view does not exist, Django.URLs raise this exception.**
- **PermissionDenied – It's triggered when a user doesn't have the necessary permissions to perform the requested activity.**
- **SuspiciousOperation – The query throws this error if only one item is expected but several things are returned.**
- **ValidationError – It's triggered when data validation fails on a form or a model field.**
- **FieldDoesNotExist – It raises when the requested field does not exist.**
- **ObjectDoesNotExist – The base class for DoesNotExist exceptions.**
- **AppRegistryNotReady – It is raised when attempting to use models before the app loading process.**
- **EmptyResultSet – If a query does not return any result, this exception is raised.**

33.What is No SQL and Does Django support NoSQL?

NoSql is also known as a non-relational database that stores data in a form of non-tabular, instead it stores data in the form of a storage model that is optimized for specific requirements of the type of data being stored. The types of NoSQL databases include pure documented databases, graph databases, wide column databases, and a key-value store.
No, Django does not officially support no-SQL databases such as CouchDB, Redis, Neo4j, MongoDB, etc.

34.What are the different model inheritance styles in Django?

Django supports 3 types of inheritance. They are

- **Abstract base classes**

- **Multi-table Inheritance**
- **Proxy models**

35.What databases are supported by Django?

Databases that are supported by Django are SQLite(Inbuild), Oracle, PostgreSQL, and MySQL. Django also uses some third-party packages to handle databases including Microsoft SQL Server, IBM DB2, SAP SQL Anywhere, and Firebird. Django does not officially support no-SQL databases such as CouchDB, Redis, Neo4j, MongoDB, etc.