## **Q1. What is the difference between Flask and Django?**

|  |  |  |
| --- | --- | --- |
| Comparison Factor | Django | Flask |
| Project Type | Supports large projects | Built for smaller projects |
| Templates, Admin and ORM | Built-in | Requires installation |
| Ease of Learning | Requires more learning and practice | Easy to learn |
| Flexibility | Allows complete web development without the need for third-party tools | More flexible as the user can select any third-party tools according to their choice and requirements |
| Visual Debugging | Does not support Visual Debug | Supports Visual Debug |
| Type of framework | Batteries included | Simple, lightweight |
| Bootstrapping-tool | Built-it | Not available |

## **Q2. What is Django?**

Django is a [**web development framework**](https://www.edureka.co/blog/python-libraries/) 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.

## **Q3. Name some companies that make use of Django?**

Some of the companies that make use of ***Django*** are [Instagram](https://instagram.com/edureka_learning/" \t "_blank), DISCUS, Mozilla Firefox, YouTube, Pinterest, Reddit, etc.

## **Q4. What are the features of Django?**

* SEO Optimized
* Extremely fast
* Fully loaded [framework](https://www.edureka.co/blog/python-frameworks/) that comes along with authentications, content administrations, RSS feeds, etc
* Very secure thereby helping developers avoid common security mistakes such as cross-site request forgery (csrf), clickjacking, cross-site scripting, etc
* It is exceptionally scalable which in turn helps meet the heaviest traffic demands
* Immensely versatile which allows you to develop any kind of websites

## **Q5. How do you check for the version of Django installed on your system?**

To check for the version of[Django installed on your system](https://docs.djangoproject.com/en/3.1/topics/install/), you can open the command prompt and enter the following command:

* python -m django –version

You can also try to import Django and use the get\_version() method as follows:

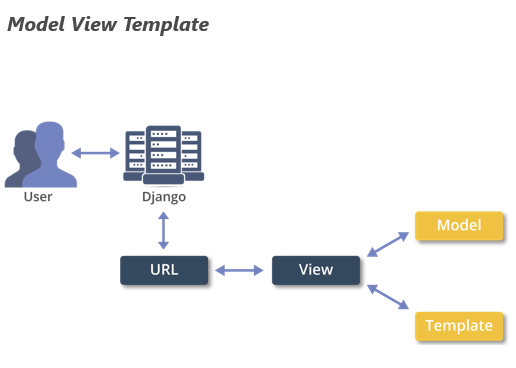
|  |  |
| --- | --- |
| 1  2 | import django  print(django.get\_version()) |

## **Q6. 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
* Behaviors are not implicitly assumed, they are rather explicitly specified
* [SQL statements](https://www.edureka.co/blog/what-is-sql/) are not executed too many times and are optimized internally
* Can easily drop into raw SQL whenever required
* Flexibility while using URL’s

## **Q7. Explain Django architecture.**

Django follows the MVT or [Model View Template architecture](https://www.edureka.co/blog/django-tutorial/#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.

## **Q8. Give a brief about ‘django-admin’.**

django-admin is the command-line utility of Django for administrative tasks. Using the django-admin you can perform a number of tasks some of which are listed out in the following table:

|  |  |
| --- | --- |
| Task | Command |
| To display the usage information and the list of the commands provided by each application | django-admin help |
| To display the list of available commands | django-admin help –command |
| To display the description of a given command and the list of its available options | django-admin help <command> |
| Determining the version of Django | django-admin version |
| Creating new migrations based on the changes made in models | django-admin makemigrations |
| Synchronizing the database state with the current set of models and migrations | django-admin migrate |
| Starting the development server | django-admin runserver |
| Sending a test email in order to confirm the email sending through Django is working | django-admin sendtestemail |
| To start the Python interactive interpreter | django-admin shell |
| To show all the migrations in your project | django-admin showmigrations |

## **Q9. How do you connect your Django project to the database?**

Django comes with a default [database](https://www.edureka.co/blog/category/databases/) which is SQLite. To connect your project to this database, use the following commands:

1. python manage.py migrate (migrate command looks at the INSTALLED\_APPS settings and creates database tables accordingly)
2. python manage.py makemigrations (tells Django you have created/ changed your models)
3. python manage.py sqlmigrate <name of the app followed by the generated id> (sqlmigrate takes the migration names and returns their SQL)

## **Q10. What are the various files that are created when you create a Django Project? Explain briefly.**

When you create a project using the startproject command, the following files will be created:

|  |  |
| --- | --- |
| File Name | Description |
| manage.py | A command-line utility that allows you to interact with your Django project |
| \_\_init\_\_.py | An empty file that tells Python that the current directory should be considered as a Python package |
| settings.py | Consists of the settings for the current project |
| urls.py | Contains the URL’s for the current project |
| wsgi.py | This is an entry-point for the web servers to serve the project you have created |

## **Q11. What are ‘Models’?**

Models are a single and definitive source for information about your data. It consists of all the essential fields and behaviors of the data you have stored. Often, each model will map to a single specific database table.

In Django, models serve as the abstraction layer that is used for structuring and manipulating your data. Django models are a subclass of the django.db.models.Model class and the attributes in the models represent database fields.

## **Q12. What are ‘views’?**

Django views serve the purpose of encapsulation. They encapsulate the logic liable for processing a user’s request and for returning  
the response back to the user. Views in Django either return an HttpResponse or raise an exception such as Http404. HttpResponse contains the objects that consist of the content that is to be rendered to the user. Views can also be used to perform tasks such as read records from the database, delegate to the templates, generate a PDF file, etc.

## **Q13. What are ‘templates’?**

Django’s template layer renders the information to be presented to the user in a designer-friendly format. Using templates, you can generate [HTML](https://www.edureka.co/blog/what-is-html/) dynamically. The HTML consists of both static as well as dynamic parts of the content. You can have any number of templates depending on the requirement of your project. It is also fine to have none of them.

[[](https://www.edureka.co/python-django)](https://www.edureka.co/python-django" \t "_blank)

### [Python Django Certification Training Course](https://www.edureka.co/python-django" \t "_blank)

[Explore Curriculum](https://www.edureka.co/python-django" \t "_blank)

Django has its own template system called the Django template language (DTL). Regardless of the backend, you can also load and render templates using Django’s standard admin.

## **Q14. What is the difference between a Project and an App?**

An app is basically a Web Application that is created to do something for example, a database of employee records. A project, on the other hand, is a collection of apps of some particular website. Therefore, a single project can consist of ‘n’ number of apps and a single app can be in multiple projects.

## **Q15. What are the different inheritance styles in Django?**

Django has three possible inheritance styles:

|  |  |
| --- | --- |
| Inheritance style | Description |
| Abstract base classes | Used when you want to use the parent class to hold information that you don’t want to type for each child model. Here, the parent class is never used in solitude |
| Multi-table inheritance | Used when you have to subclass an existing model and want each  model to have its own database table |
| Proxy models | Used if you only want to modify the Python-level behavior of a model, without changing the ‘models’ fields in any way |

## **Q16. What are static files?**

Static files in Django are those files that serve the purpose of additional files such as the [CSS](https://www.edureka.co/blog/what-is-css/), images or [JavaScript](https://www.edureka.co/blog/what-is-javascript/) files. These files are managed by django.contrib.staticfiles. These files are created within the project app directory by creating a subdirectory named as static.

## **Q17. What are ‘signals’?**

Django consists of a signal dispatcher that helps allow decoupled applications to get notified when actions occur elsewhere in the framework. Django provides a set of built-in signals that basically allow senders to notify a set of receivers when some action is executed. Some of the signals are as follows:

|  |  |
| --- | --- |
| Signal | Description |
| django.db.models.signals.pre\_save  django.db.models.signals.post\_save | Sent before or after a model’s save() method is called |
| django.db.models.signals.pre\_delete  django.db.models.signals.post\_delete | Sent before or after a model’s delete() method or queryset’s delete() method is called |
| django.db.models.signals.m2m\_changed | Sent when Django starts or finishes an HTTP request |

## **Q18. Briefly explain Django Field Class.**

‘Field’ is basically an abstract class that actually represents a column in the database table. The Field class, is in turn, a subclass of  RegisterLookupMixin. In Django, these fields are used to create database tables (db\_type()) which are used to map Python types to the database using get\_prep\_value() and vice versa using from\_db\_value() method. Therefore, fields are fundamental pieces in different Django APIs such as models and querysets.

## **Q19. How to do you create a Django project?**

To create a Django project, cd into the directory where you would like to create your project and type the following command:

* django-admin startproject xyz

**NOTE:**Here, xyz is the name of the project. You can give any name that you desire.

## **Q20. What is mixin?**

Mixin is a type of multiple inheritance wherein you can combine behaviors and attributes of more than one parent class. Mixins provide an excellent way to reuse code from multiple classes. For example, generic class-based views consist of a mixin called TemplateResponseMixin whose purpose is to define render\_to\_response() method. When this is combined with a class present in the View, the result will be a TemplateView class.

One drawback of using these mixins is that it becomes difficult to analyze what a child class is doing and which methods to override in case of its code being too scattered between multiple classes.

## **Q21. What are ‘sessions’?**

Sessions are fully supported in Django. Using the session framework, you can easily store and retrieve arbitrary data based on the per-site-visitors. This framework basically stores data on the server-side and takes care of sending and receiving cookies. These cookies consist of a session ID but not the actual data itself unless you explicitly use a cookie-based backend.

## **Q22. What do you mean by context?**

Context in Django is a dictionary mapping template variable name given to [Python objects](https://www.edureka.co/blog/python-class/#Objects). This is the conventional name, but you can give any other name of your choice if you wish to do it.

## **Q23. When can you use iterators in Django ORM?**

Iterators in Python are basically containers that consist of a countable number of elements. Any object that is an iterator implements two methods which are, the \_\_init\_\_() and the \_\_next\_\_()  methods. When you are making use of iterators in Django, the best situation to do it is when you have to process results that will require a large amount of memory space. To do this, you can make use of the iterator() method which basically evaluates a QuerySet and returns the corresponding iterator over the results.

## **Q24. Explain the caching strategies of Django?**

Caching basically means to save the output of an expensive calculation in order to avoid performing the same calculation again. [Django](https://www.edureka.co/blog/django-tutorial/" \t "_blank) provides a robust cache system which in turn helps you save dynamic web pages so that they don’t have to be evaluated over and over again for each request. Some of the caching strategies of Django are listed down in the following table:

|  |  |
| --- | --- |
| Strategy | Description |
| Memcached | Memory-based cache server which is the fastest and most efficient |
| Filesystem caching | Cache values are stored as separate files in a serialized order |
| Local-memory caching | This is actually the default cache in case you have not specified any other. This type of cache is per-process and thread-safe as well |
| Database caching | Cache data will be stored in the database and works very well if you have a fast and well-indexed database server |

## **Q25. Explain the use of Middlewares in Django.**

You may come across numerous Django Interview Questions, where you will find this question. Middleware is a framework that is light and low-level plugin system for altering Django’s input and output globally. It is basically a framework of hooks into the request/ response processing of Django. Each component in middleware has some particular task. For example, the AuthenticationMiddleware is used to associate users with requests using sessions. Django provides many other middlewares such as cache middleware to enable site-wide cache, common middleware that performs many tasks such as forbidding access to user agents, URL rewriting, etc, GZip middleware which is used to compress the content for browsers, etc.

## **Q26. What is the significance of manage.py file in Django?**

The manage.py file is automatically generated whenever you create a project. This is basically a command-line utility that helps you to interact with your Django project in various ways. It does the same things as django-admin but along with that, it also sets the DJANGO\_SETTINGS\_MODULE environment variable in order to point to your project’s settings. Usually, it is better to make use of manage.py rather than the django-admin in case you are working on a single project.

## **Q27. Explain the use of ‘migrate’ command in Django?**

In Django, migrations are used to propagate changes made to the models. The migrate command is basically used to apply or unapply migrations changes made to the models. This command basically synchronizes the current set of models and migrations with the database state. You can use this command with or without parameters. In case you do not specify any parameter, all apps will have all their migrations running.

## **Q28. How to view and filter items from the database?**

In order to view all the items from your database, you can make use of the ‘all()’ function in your interactive shell as follows:

* XYZ.objects.all()     where XYZ is some class that you have created in your models

To filter out some element from your database, you either use the get() method or the filter method as follows:

* XYZ.objects.filter(pk=1)
* XYZ.objects.get(id=1)

## **Q29. Explain how a request is processed in Django?**

In case some user requests a page from some Django powered site, the system follows an algorithm that determines which Python code needs to be executed. Here are the steps that sum up the algorithm:

1. Django first determines which root URLconf or URL configuration module is to be used
2. Then, that particular Python module is loaded and then Django looks for the variable urlpatterns
3. These URL patterns are then run by Django, and it stops at the first match of the requested URL
4. Once that is done, the Django then imports and calls the given view
5. In case none of the URLs match the requested URL, Django invokes an error-handling view

## **Q30. How did Django come into existence?**

Django basically grew from a very practical need. World Online developers namely Adrian Holovaty and Simon Willison started using [Python](https://www.edureka.co/blog/python-basics/) to develop its websites. As they went on building intensive, richly interactive sites, they began to pull out a generic Web development framework that allowed them to build Web applications more and more quickly. In summer 2005, World Online decided to open-source the resulting software, which is, Django.

## **Q31. How to use file-based sessions?**

In order to make use of file-based sessions, you will need to set the SESSION\_ENGINE setting to “django.contrib.sessions.backends.  
file”.

## **Q32. Explain the Django URLs in brief?**

Django allows you to design your own URLs however you like. The aim is to maintain a clean URL scheme without any framework limitations. In order to create URLs for your app, you will need to create a Python module informally called the URLconf or URL configuration which is pure Python code and is also a mapping between the URL path expressions to the Python methods. The length of this mapping can be as long or short as required and can also reference other mappings. When processing a request, the requested URL is matched with the URLs present in the urls.py file and the corresponding view is retrieved. For more details about this, you can refer to the answer to Q29.

## **Q33. Give the exception classes present in Django.**

Django uses its own exceptions as well as those present in Python. Django core exceptions are present in django.core.exceptions class some of which are mentioned in the table below:

|  |  |
| --- | --- |
| Exception | Description |
| AppRegistryNotReady | Raised when you try to use your models before the app loading process (initializes the ORM) is completed. |
| ObjectDoesNotExist | This is the base class for DoesNotExist exceptions |
| EmptyResultSet | This exception may be raised if a query won’t return any result |
| FieldDoesNotExist | This exception is raised by a model’s \_meta.get\_field() function in case the requested field does not exist |
| MultipleObjectsReturned | This is raised by a query if multiple objects are returned and only one object was expected |

## **Q34. Is Django stable?**

Yes, Django is quite stable. Many companies like [Instagram](https://instagram.com/edureka_learning/" \t "_blank), Discus, Pinterest, and Mozilla have been using Django for a duration of many years now. Not just this, Websites that are built using Django have weathered trafﬁc spikes of over 50 thousand hits per second.

## **Django Interview Questions**

## **Q35. Does the Django framework scale?**

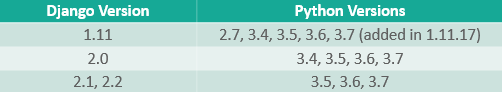
Yes. Hardware is much cheaper when compared to the development time and this is why Django is designed to make full use of any amount of hardware that you can provide it. Django makes use of a “shared-nothing” architecture meaning you can add hardware at any level i.e database servers, caching servers or Web/ application servers.

## **Q36. Is Django a CMS?**

Django is not a CMS (content-management-system) . It is just a Web framework, a tool that allows you to build websites.

## **Q37. What Python version should be used with Django?**

The following table gives you the details of the versions of Python that you can use for Django:



Python 3 is actually the most recommended because it is fast, has more features and is better supported. In the case of Python 2.7, Django 1.1 can be used along with it but only till the year 2020.

## **Q38. Does Django support NoSQL?**

[NoSQL](https://www.edureka.co/blog/sql-vs-nosql-db/#What%20is%20NoSQL?) basically stands for “not only SQL”. This is considered as an alternative to the traditional RDBMS or the relational Databases.  Officially, Django does not support NoSQL databases. However, there are third-party projects, such as Django non-rel, that allow NoSQL functionality in Django. Currently, you can use MongoDB and Google App Engine.

## **Q39. How can you customize the functionality of the Django admin interface?**

There are a number of ways to do this. You can piggyback on top of an add/ change form that is automatically generated by Django, you can add JavaScript modules using the js parameter. This parameter is basically a list of URLs that point to the JavaScript modules that are to be included in your project within a <script> tag. In case you want to do more rather than just playing around with from, you can exclusively write views for the admin.

## **Q40. Is Django better than Flask?**

Django is a framework that allows you to build large projects. On the other hand, [Flask](https://www.youtube.com/watch?v=lj4I_CvBnt0) is used to build smaller websites but flask is much easier to learn and use compared to Django. Django is a full-fledged framework and no third-party packages are required. Flask is more of a lightweight framework that allows you to install third-party tools as and how you like. So, the answer to this question basically depends on the user’s need and in case the need is very heavy, the answer is definitely, Django.

## **Q41. Give an example of a Django view.**

A view in Django either returns an HttpResponse or raises an exception such as Http404. HttpResponse contains the objects that consist of the content that is to be rendered to the user.

**EXAMPLE:**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | from django.http import HttpResponse  def hello\_world(request):      html = "  <h1>Hello World!</h1>    "      return HttpResponse(html) |

**Q42. What should be done in case you get a message saying “Please enter the correct username and password” even after entering the right details to log in to the admin section?**

In case you have entered the right details and still not able to login to the admin site, cross verify if the user account has is\_active and is\_staff attributes set to True. The admin site allows only those users for whom these values are set to True.

## **Q43. What should be done in case you are not able to log in even after entering the right details and you get no error message?**

In this case, the login cookie is not being set rightly. This happens if the domain of the cookie sent out by Django does not match the domain in your browser. For this, you must change the SESSION\_COOKIE\_DOMAIN setting to match that of your browser.

## **Q44. How can you limit admin access so that the objects can only be edited by those users who have created them?**

Django’s ModelAdmin class provides customization hooks using which, you can control the visibility and editability of objects in the admin. To do this, you can use the get\_queryset() and has\_change\_permission().

## **Q45. What to do when you don’t see all objects appearing on the admin site?**

Inconsistent row counts are a result of missing Foreign Key values or if the Foreign Key field is set to null=False. If the ForeignKey points to a record that does not exist and if that foreign is present in the list\_display method, the record will not be shown the admin changelist.

[[](https://www.edureka.co/python-django)](https://www.edureka.co/python-django" \t "_blank)

### [Python Django Certification Training Course](https://www.edureka.co/python-django" \t "_blank)

[Weekday / Weekend BatchesSee Batch Details](https://www.edureka.co/python-django" \t "_blank)

## **Q46. What do you mean by the csrf\_token?**

The csrf\_token is used for protection against Cross-Site Request Forgeries. This kind of attack takes place when a malicious website consists of a link, some [JavaScript](https://www.edureka.co/blog/javascript-tutorial/) or a form whose aim is to perform some action on your website by using the login credentials of a genuine user.

## **Q47. Does Django support multiple-column Primary Keys?**

No. Django only supports single-column Primary Keys.

## **Q48. How can you see the raw SQL queries that Django is running?**

First, make sure that your DEBUG setting is set to True. Then, type the following commands:

|  |  |
| --- | --- |
| 1  2 | from django.db import connection  connection.queries |

## **Q49. Is it mandatory to use the model/ database layer?**

No. The model/ database layer is actually decoupled from the rest of the framework.

## **Q50. How to make a variable available to all the templates?**

You can make use of the RequestContext in case all your templates require the same objects, such as, in the case of menus. This method takes an HttpRequest as its first parameter and it automatically populates the context with a few variables, according to the engine’s  
context\_processors configuration option.

Below are the top Django interview questions and answers that will surely boost your confidence in interview preparation.

**Q.1 What is Django? Elaborate some technical features of Django.**

**Ans.** Django is a high-level web application framework based on Python.

This framework is one of the best in the industry for rapid development, pragmatic design without compromising on features.

Some of the technical features of Django include:

* Admin Interface
* Code Reusability
* CDN Integration
* 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.

***Check out other unique***[***features of Django***](https://data-flair.training/blogs/django-features/)***in detail***

**Q.2 What is Django Admin Interface?**

**Ans.** Django Admin is the preloaded interface made to fulfill the need of web developers as they won’t need to make another admin panel which is time-consuming and expensive.

Django Admin is application imported from django.contrib packages.

It is meant to be operated by the organization itself and therefore doesn’t need the extensive frontend.

Django’s Admin interface has its own user authentication and most of the general features.

It also offers lots of advanced features like authorization access, managing different models, CMS (Content Management System), etc.

**Q.3 How is Django’s code reusability feature different from other frameworks?**

**Ans.** Django framework offers more code-reusability than other frameworks out there.

As Django Project is a collection of different applications like login application, signup application.

These applications can be just copied from one directory to another with some tweaks to settings.py file and you won’t need to write new signup application from scratch.

That’s why Django is a rapid development framework and this level of code reusability is not there in other frameworks.

**Q.4 Explain the file structure of a typical Django project?**

**Ans.**A Django project is a collection of web-applications that coordinate together to serve the request of the user.

These applications have one assigned feature and shall do only that.

A ***typical Django project*** consists of these four files:

* manage.py
* settings.py
* \_\_init\_\_.py
* urls.py
* wsgi.py

The last four files are inside a directory, which is at the same level of manage.py.

Here the structure is very logical, and the names of these files and their purpose should remain intact.

**manage.py** is the command-line utility of your Django project and this file is used to control your Django project on the server or even to begin one.

When Django server is started, the manage.py file searches for **settings.py file**, which contains information of all the applications installed in the project, middleware used, database connections and path to the main urls config.

The**urls.py**file is like a map of your whole web-project, this file examines URL and calls the right view function or transports the URL to another application-specific urls-config file.

This is like the main URL linker and any app installed in the settings.py which you wish to be searched by the URL should have a link here.

The **\_\_init\_\_.py** file is an empty file which is there to make the python interpreter understand that the directory consisting settings.py is a module/ package.

The **wsgi.py** file is for the server format WSGI, which Django supports natively. We can customize that for other server formats.

**Note –**This one is the most popular Django interview question.

**Q.5 Django is an MVC based framework, how this framework implements MVC?**

**Ans.**Django is based on MTV architecture which is a variant of MVC architecture.

MVC is an acronym for Model, View, and Controller.

There are different parts of a website so that they can develop and execute in different machines to achieve faster and more responsive websites.

Django implements MTV architecture by having 3 different components and they are all handled by Django itself.

Models are the part which is models.py file in a Django application, which defines the data structure of the particular application.

View are the mediators between models and templates, they receive the data from the Model and make it a dictionary and return the same as a response to a request to the Template.

The Template is the component with which user interacts, and it generates both statically and dynamically in the Django server.

That’s how the Django implements 3 components and work in coordination with each other.

**Q.6 What happens when a typical Django website gets a request? Explain.**

**Ans.**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.

**Q.7 What is the Controller in the MVC framework of Django?**

**Ans.**As Django implements in MTV framework, these three components communicate with each other via the controller and that controller is actually Django framework.

Django framework does the controlling part itself.

**Q.8 Is Django’s Admin Interface customizable? If yes, then How?**

**Ans.**Django’s Admin is just one of the applications and very customizable, also you can download a different third-party application and install it for a totally different view.

You can make your own Admin Application if you want complete control over your Admin.

Although you can customize the Django Admin site like changing the properties of admin.site object.

We can also make some changes in particular models and apply them in our Django Admin for particular apps like we can add a search bar for particular applications.

The **Django Admin Interface** is fully customizable to the lowest level, but instead of customizing that much, we can rather create a new Admin Interface.

So those who don’t like Django Admin Interface, prefer making a new one from scratch then editing the previous one.

**Q.9 Why is Django called a loosely coupled framework?**

**Ans.**Django is called a loosely coupled framework because of the MTV architecture it’s based on.

Django’s architecture is a variant of MVC architecture and MTV is useful because it completely separates server code from the client’s machine.

Django’s Models and Views are present on the client machine and only templates return to the client, which are essentially HTML, CSS code and contains the required data from the models.

These components are totally different from each other and therefore, front-end developers and backend developers can work simultaneously on the project as these two parts changing will have little to no effect on each other when changed.

Therefore, Django is a loosely coupled framework.

**Note –**This one is the favorite Django interview question of many interviewees.

**Q.10 What is Django Rest Framework (DRF)?**

**Ans.**Django REST is a framework which lets you create RESTful APIs rapidly.

This framework has got funding by many big organizations and is popular because of its features over Django frameworks like Serialisation, Authentication policies and Web-browsable API.

RESTful APIs are perfect for web applications since they use low bandwidth and are designed such that they work well with communications over the Internet like GET, POST, PUT, etc.

**Q.11 Explain the importance of settings.py file and what data/ settings it contains.**

**Ans.**When Django server starts, it first looks for settings.py. As the name settings, it is the main settings file of your web application.

Everything inside your Django project like databases, backend engines, middlewares, installed applications, main URL configurations, static file addresses, templating engines, allowed hosts and servers and security key stores in this file as a list or dictionary.

So, when your Django server starts it executes settings.py file and then loads particular engines and databases so that when a request is given it can serve the same quickly.

**Q.12 Why Django uses regular expressions to define URLs? Is it necessary to use them?**

**Ans.**Django uses a very powerful format for storing URLs, that is regular expressions.

RegEx or regular expression is the format for sophisticated string searching algorithms. It makes the searching process faster.

Although it’s not necessary to use RegEx when defining URLs.

They can be defined as normal string also, Django server should still be able to match them, but when you need to pass some data from the user via URL, then RegEx is used.

The RegEx also makes much cleaner URLs then other formats.

**Q.13 What is Django ORM?**

**Ans.**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.

Django ORM is the abstraction between models (web application data-structure) and the database where the data is stored.

It makes possible to retrieve, save, delete and perform other operations over the database without ever writing any SQL code.

It also covers many loopholes and takes all the field attributes and gives you more control over your code in Python rather than any database language.

**Q.14 What is a Model in Django and what is the Model class?**

**Ans.**A Model in Django is a python class which derives from Model class that imports from the django.db.models library.

The ***concept of Django Models*** is to create objects that can store data from the user in a user-defined format.

Therefore, python class is used for the process and that class is generally defined in models.py file of the particular application.

The model class is a pre-defined class of Django framework and every class which is a model must derive from the same.

The model class has lots of benefits like you can define the field with specific attributes as you would do in SQL, but now the same can be achieved in Python.

Django Model class is parsed by the Django ORM or backend engine and you won’t need to do anything related to the database, like creating tables and defining fields afterward mapping the fields with the attribute of the class.

**Q.15 How does Django Templating Work?**

**Ans.**Django Templates are Django’s answer to generate dynamic web pages. Templates, in general, are the HTML or the formats which can return as an Http response.

Django templating engine handles templating in the Django framework.

There are some template syntaxes which declares variables, control logic, filters, and comments.

After putting these inside the HTML structure, when the web page is requested and called upon by the view function, the Django Template engine gets two things, the HTML structure with variables in place and the data to replace with those variables.

It replaces the variables with data while also executing the control logic and generating filters.

It renders the required HTML and sends it to the browser when all the work gets complete.

**Q.16 What are View functions? Can we directly import a function in URL?**

**Ans.** The View is the middle component in Django that receives data from the Django models and pass the same to the Templates.

Every application in Django comes with views.py file, this file contains the View functions.

The View functions are functions which receive an argument and they return a browser-renderable format or a redirect.

Django Views function can import directly in the urls file.

For that, we have to first import the view function in the urls.py file and then add the path/ URL which browser should request to call that View function.

Here as you can see that we imported all the functions from our View module which is in the same folder.

We added the URL in the urlpatterns list (red box). When the ‘dataflair/’ gets searched in the yellow box, we have called a function named index.

**Q.17 What is django.shortcuts.render function?**

**Ans.** When a View function returns a webpage as HttpResponse rather than a simple string, we use render().

Render function is a shortcut function which lets the developer to easily pass the data dictionary with the template.

This function then combines the template with data dictionary via templating engine.

Finally, this render() returns an HttpResponse with the rendered text, which is the data returned by the models.

Thus, Django render() bypasses lots of work for the developer and lets him use different templating engines.

It is because this function provides the same functionality with other templating systems.

The basic render Syntax:

render(request, template\_name, context=None, content\_type=None, status=None, using=None)

The request is the parameter which generates the response, the template\_name containing the value where the template is stored.

The template name and other parameters are for passing the dictionary.

If you want more control, you can specify the content type, status of the data you passed and the render you are returning. That is the render().

**Note –** This one is a bit difficult Django interview question. Make sure to prepare it thoroughly.

**Q.18 List some ways by which we can add our View functions to urls.py file?**

**Ans.** We can add our view to the main urls config in two ways:

**1. Adding a function View**

In this method, we import our view as function.

We import the function itself from the particular view and then, add the particular URL to the urlpatterns list.

**2. Adding a Class-based view**

The class-based view is a more object-oriented approach.

To begin, import the class from the views.py and then add the URL to the urlpatterns. This time we will need an inbuilt method to call the class as a view.

In the name of the function on the previous method, write

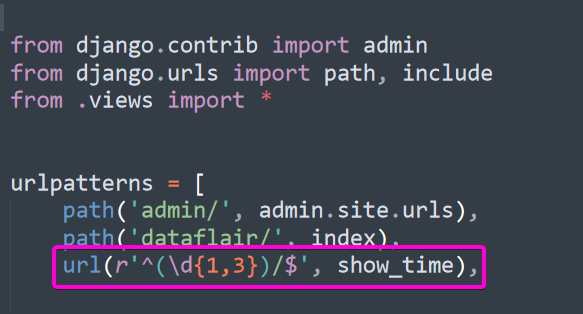
class\_name.as\_view()

This will pass your view class as view function.

Both class-based views and function-based views have their own pros and cons and we can use them in the appropriate situations to get the right results.

**Q.19 How can we extract the data from the request/ URL and pass the same to the View function?**

**Ans.** We can very easily take some input via URL request and generate something dynamically via the same input.

[](https://data-flair.training/blogs/wp-content/uploads/sites/2/2019/04/dataflair-index-input.png)

Here focus on the pink box, where a regular expression as URL is present. It will perform an important task that is checking the URL which has 3 or fewer digits.

[](https://data-flair.training/blogs/wp-content/uploads/sites/2/2019/04/Welcome-to-dataflair-command.png)

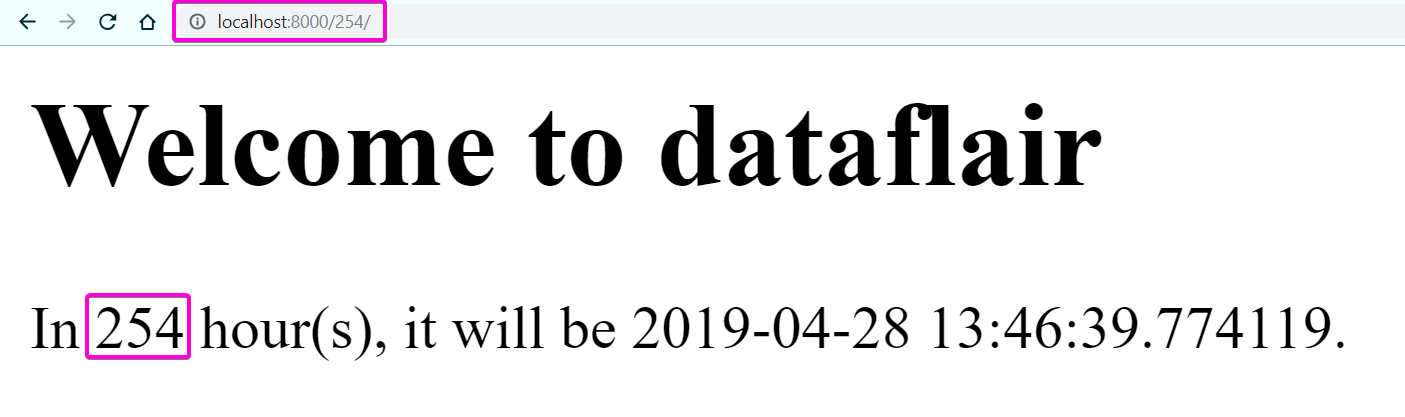
That code will call the show\_time() in the Views file and here as you can see we are passing two values in the request.

The other one is offset, which is a variable containing the numerical value which we entered after the URL.

So when the Django server gets the URL, it snips off the rest part and compares it with regex by automatically passing the numerical part or the part which is random in RegEx.

After that, it transfers to the function, where we are converting the same in integer type as, by default, python takes the URL requests as strings.

Then we execute regular python statements and we get a rendered application.

[](https://data-flair.training/blogs/wp-content/uploads/sites/2/2019/04/welcome-to-dataflair-url-input.png)

**Q.20 What does urls-config file contain?**

**Ans.**The URLs-config in Django contains the list of urls and the mappings to view functions for those urls.

The urls can map to view functions, Class-based views and urls-config of another application. All these methods have their use-cases.

**For example –** If we want to keep all the URLs of our application sorted, we will use the URL-config mapping. Inside urls file, we will use view function mapping and class-based views if we require some data from the user.

**Q.21 How does Django compare to other popular frameworks like Laravel?**

**Ans.**Django has many unique features, the top-line is that Django allows for rapid development of applications without any security loopholes or performance lacking.

Some of the features which Laravel doesn’t have but Django implies are:

* Laravel is PHP based while Django is Python-based, which is clearly more powerful and robust then PHP.
* The performance of Django is better than Laravel because of the different programming languages it uses.
* Django is based on MTV architecture, a more robust and loosely coupled architecture while Laravel is strictly based on MVC architecture.
* Django can use RegEx (as used in the previous answer), while Laravel doesn’t provide you with that functionality.

**Q.22 Django is too monolithic. Explain this statement.**

**Ans.**Django framework is too monolithic, which is true to some extent.

Django is MTV architecture-based framework and since Django is the controller of the architecture, it requires some rules that the developer should follow so that the framework can find and execute appropriate files at the right time.

Therefore, Django is one of the frameworks where file structure is as important as its architecture.

In Django, you get great customizability with the implementations.

There is just one condition that you cannot change the file names, the pre-defined lists, and variable names.

You can create new ones but you can’t change the predefined variables for which people say that they always have to follow a certain pattern while working on Django.

Django’s file structure is one of the most logical workflows.

The monolithic behavior is actually helping the developers to easily understand the project.

Even, when the company changes, the project layout remains the same.

Therefore, the developer would take less time to understand every aspect, will be able to perform more work productively.

**Q.23 What is the latest release of Django framework? What are its features?**

**Ans.**Django’s latest version is Django 3.0. Some new features of Django 3.0 are:

* ASGI Support for Async Programming
* It supports MariaDB 10.1 and higher
* Custom enumeration types TextChoices, IntegerChoices, and Choices are now available as a way to define model field choices
* Exclusion constraints on PostgreSQL
* Filter expressions

**Q.24 What is Jinja Templating?**

**Ans.**Django supports many popular templating engines and by default, it comes with one very powerful templating engine.

Jinja Templating is a very popular templating engine for Python, the latest version in the market is Jinja 2.

There are some features of Jinja templating that make it a better option than the default template system in Django.

* **Sandbox Execution –**This is like a sandbox or a protected framework for automating the testing process.
* **HTML Escaping –**Jinja 2 provides automatic HTML Escaping, as <, >, & characters have special values in templates and if used as regular text, these symbols can lead to XSS Attacks which Jinja deals with automatically.
* **Template Inheritance**
* **Generates HTML templates much faster than default engine**
* **Easier to debug, compared to default engine.**

**Q.25 Why permanent redirecting is not a good option?**

**Ans.**Permanent redirecting is not a good option because the browser caches the response generated by the permanent redirect.

This is the difference between permanent and temporary redirect. It causes all sorts of issues when you change that redirect to something different.

Since the browser has cached the redirect before, this time it won’t look on the server for the changed redirection and will load the previously saved redirect.

So, even though the developer might have redirected the user to a different page, it will still load the same page.

It is browser/ client-side operation, therefore, the user can’t even do anything about the same.

Because of this reason, permanent redirecting is not a good option as informing the users to clear their internal caching data is not good for any website.

**Q.26 Why is Django better than Flask?**

**Ans.**Django has its own unique qualities over Flask which is also a Python Framework. The key differences between them are:

* Django is a high-level Python framework while Flask is a low-level Python Framework providing you with the minimum functionality, a server would require.
* Django comes with lots of built-in functionality like **Django ORM, Admin Panel, Web-templating System, Inheritance, serialization** while Flask comes with a **development server, NoSQL support, support for unit testing**, which are already there in Django.
* Flask is more customizable than Django as Flask comes with no pre-defined structure or scaffold while Django’s file structure is fixed.
* Flask settings are user made and can be altered completely by the user. Django settings are not customizable to that degree, it has variables where only values are modifiable.
* Flask has more speed than Django when it comes to processing requests but that comes without any APIs or functionality which Django gives you in-built.
* Flask is for the developers who want more flexibility on their website and don’t need lots of built-in extra functions, while Django is for developers who want rapid development of their applications that can sustain dynamic changes to its environment.

**Note –** Comparison questions are very popular in the Django interview. Prepare this one nicely.

**Q.27 Explain user authentication in Django?**

**Ans.**Django comes with a built-in user authentication system, which handles objects like users, groups, user-permissions, and some cookie-based user sessions.

Django’s User authentication not only authenticates (verifying the user identity) the user but also authorizes him (determines what permissions the user have).

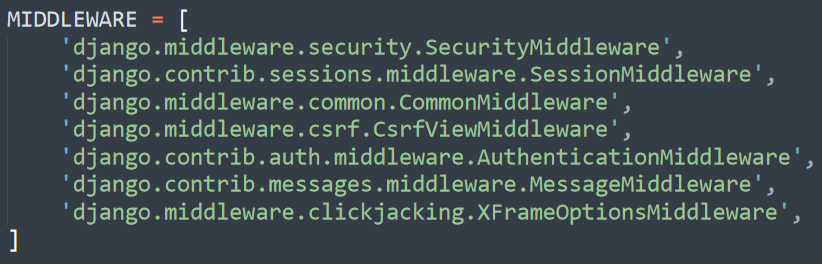
The system consists and operates on these objects:

* users
* Permissions
* Groups
* Password Hashing System
* Forms Validation
* A pluggable backend system

There are many third-party web applications that we can use in place of the default system as they provide much more control over user authentication with more features.

**Q.28 Middleware in Django is useful for which purpose?**

**Ans.**Middleware in the Django framework is the component that operates on request and transfers it to the view and before passing it to the template engine, it starts operating on a response.

[](https://data-flair.training/blogs/wp-content/uploads/sites/2/2019/04/Django-middleware-list.png)

This is the list of middleware that installs by default in your Django framework.

It serves many different purposes like session management, user authentication, etc.

**Q.29 What is the use of the djangopackages.org website?**

**Ans.**Django packages website is the place where all the third-party applications upload.

After that, you can install them in your system or in the Django project very easily.