1. **What is Django ?**

Django is a high-level Python-based free and open-source web framework, which follows the model-view-template architectural pattern.

1. **What is latest and stable version of Django ?**

The latest and stable version  is version 2.2.1.

1. **Why django ? What is advantages of Django ?**

Its free and open source, it makes the web development process very easy and the developer can fully focus on the designing process and boost performance. It is an ideal tool for startups, where web design is the need to bring out the real concept and prospects of the company.

It makes it easier to build web apps more quickly and with less code, It is also ridiculously fast, reassuringly secure and exceedingly scalable.

1. **What are features of Django ?**

Django’s features are: Rapid Development, Secure, Scalable, Fully Loaded, Versatile, Open Source, Vast and Supported Community.

1. **What id CRUD ?**

CRUD means Create, Read, Update and Delete. It provides a memorable framework for reminding developers of how to construct full, usable models when building application programming interfaces (APIs).

1. **What are disadvantages of Django ?**

Django’s disadvantages are that the URL specifying with regular expressions is not an easy task to accomplish, at least for beginners, It’s a strongly opinionated framework, which gives it a monolithic feeling. Also, template errors fail silently by default,.

1. **Explain the architecture of Django?**

Django Architecture is based on Model View Controller (MVC) pattern. It consists of 3 different parts:

* The Model which is the logical data structure behind the complete application & represents by a Database.
* The View which is the user interface that you see in the browser whenever you visit a website.
* The Controller which is the middleman that connects the view & model together. I

1. **Can you explain the working philosophy of Django?**

Django’s working philosophy can be broken into many components:

Models.py file: This file defines your data model by extending your single line of code into full database tables and add a pre-built administration section to manage content.

Urls.py file: It uses a regular expression to capture URL patterns for processing.

Views.py file: It is the main part of Django. The actual processing happens in view.

When a visitor lands on Django page:

1. Django checks the various URLs pattern you have created and uses the information to retrieve the view.
2. The view processes the request, querying your database if necessary.
3. The view passes the requested information on to your template.
4. The template renders the data in a layout you have created and displays the page.

1. **What is MVT and MVC, How it’s related to Django?**

A common way of explaining Django’s architecture in terms of MVC is to describe it as a Model-View-Template (MVT). There’s no difference between MTV and MVT, by the way – they’re two different ways of writing exactly the same thing, which just adds to the confusion.

1. **How to install django ?**

To install Django, we need to download and install Python as required depending on the operating system in use. t’s recommended  to create a virtual environment. Run the command **pip install “django>=2.2,<3”** on the terminal and wait until it’s successfully installed.

1. **How to check the latest version of Django?**

To check the latest version of Django, visit their homepage <https://www.djangoproject.com/> and on the right of the page you’ll see a “Download latest release” button with the latest version on it.

1. **How can you setup Database in Django?**

Create the initial Django project skeleton by using the **django-admin startproject** command.

Open the settings from the **settings.py**file to confirm all settings such as your timezone are suitable for your project, else make changes or add new settings where necessary.

Install MySQL Database Connector.

Create the database.

Add the MySQL Database Connection to your Application.

Test MySQL Connection to Application.

1. **What is the list of backends supported by django ?**

Backends include: IBM DB2, Microsoft SQL Server, Firebird and ODBC.

1. Can i Connect mysql to django ?  
   Yes
2. Can i Connect plsql to django ?  
   Yes
3. How to connect mongodb in Django ?

MongoDB can be connected to Django by adding the DATABASES object with its default object in the **settings.py**file and specifying the ENGINE’s value as **django**and the NAME’s value as the name of the MongoDB database you want to connect to.

1. **How can you set up static files in Django?**

Make sure that **django.contrib.staticfiles**is included in your **INSTALLED\_APPS**.

In your settings file, define the **STATIC\_URL**to the folder path string where your static files will be placed and make sure the folder exists.

In your templates, use the **static**template tag to build the URL, for the given relative path using the configured **STATICFILES\_STORAGE**.

Store your static files in the static folder you specified. In addition to using a static directory inside your app, you can define a list of directories (**STATICFILE\_DIRS**) in your settings file where Django will also look for static files.

1. **What is the use of session framework in Django?**

Session framework lets you store and retrieve arbitrary data on a per-site-visitor basis. It stores data on the server-side and abstracts the sending and receiving of cookies each containing a session ID unless you are using a cookie-based backend.

1. **What is the usage of middlewares in Django?**

Middlewares are used  go to modify the request i.e. HttpRequest object which is sent to the view, to modify the HttpResponse object returned from the view and to perform an operation before the view executes.

1. **What are the roles of receiver and sender in signals?**

* The receiver is the callback function which will be connected to a signal
* The sender specifies a particular sender to receive signals from.

1. **What does Django templates contain ?**

Django templates contains the static parts of the desired HTML output as well as some special syntax describing how dynamic content will be inserted.

1. **How to create super user in django ?**

To create a super user,,

* Create project using the **django-admin startproject** command.
* Move into the project location and run **python manage.py makemigrations && python manage.py migrate && python manage.py createsuperuser**

1. **How to create simple application in django ?**

To create a simple application  use the command **django-admin startproject**followed by the application’s name.

1. **What is ORM ? Advantages of ORM ?**

ORM (Object-relational mapping) is a programming technique for converting data between incompatible type systems using object-oriented programming languages.

Advantages include:

* Concurrency support
* Cache management

1. **How to create a model in django ?**

Add the model object in the **models.py**file, updated settings for the newly created app by adding it to the **INSTALLED\_APPS**section in **settings.py**, make migrations, and verify the database schema.

1. **What is migration in django ?**

Migrations are a way of propagating changes made in the model into the database schema (adding a field, deleting a model, etc.)

1. **How to do migrations in django ?**

To do migrations , create or update a model and in the app directory, run the command **./manage.py makemigrations <app name> && ./manage.py migrate <app name>**

1. **How to clear cache in django ?**

To clear cache, run the **clear()**method from **django.core.cache**in a python script.

1. **What is Rest API ?**

A REST API is an application program interface that uses HTTP requests to GET, PUT, POST and DELETE data.

1. **How to Create APIs in Django ?**

Create a project directory,  create python virtual environment,   and activate it, install Django and **djangorestframework** using the **pip install command**. In the same project directory, create  project using the command **django-admin.py startproject api.**Start the app. Add the **rest\_framework** and the Djano app to INSTALLED\_APPS to settings. Open the **api/urls.py**and add urls for the Django app. We can then create models and make migrations, create serializers, and finally wiring up the views.

1. **What is DRF of Django Rest Frame work ?**

Django Rest Framework (DRF) is a powerful module for building web APIs. It’s very easy to build model-backed APIs that have authentication policies and are browsable.

1. **How to Fetch data from apis using Django ?**

We use the Fetch API and **SessionAuthentication** by adding it to the **settings.py**file on the server and on the client, include  the **getCookie**method. Finally, use the fetch method to call your endpoint.

1. **How to update the data from apis ?**

We update data by sending **PUT**requests. Add a new path in the data model **urlpatterns**from which the update will be sent to. We then add an update method to the serializer that will do the update.

1. **What is Authentication ?**

Authentication is the process or action of verifying the identity of a user or process.

1. **Types of Authentication in REST API ?**

Token based authentication and Session based authentication.

1. **What is token based authentication system ?**

A token based authentication system is a security system that authenticates the users who attempt to log in to a server, a network, or some other secure **system**, using a security **token** provided by the server

1. **Can i use django apis in mobile application development ?**Yes
2. **Explain Mixins in Django ?**

A mixin is a special kind of multiple inheritance. There are two main situations where mixins are used: to provide a lot of optional features for a class and to use one particular feature in a lot of different classes

1. **Different types caching strategies in django ?**

Different types of  caching strategies  include Filesystem caching, in-memory caching, using memcached and database caching.

1. **How a request is process in Django ?**

When the user makes a request of your application, a WSGI handler is instantiated, which:

* imports your settings.py file and Django’s exception classes.
* loads all the middleware classes it finds in the MIDDLEWARE\_CLASSES or MIDDLEWARES(depending on Django version) tuple located in settings.py
* builds four lists of methods which handle processing of request, view, response, and exception.
* loops through the request methods, running them in order
* resolves the requested URL
* loops through each of the view processing methods
* calls the view function (usually rendering a template)
* processes any exception methods
* loops through each of the response methods, (from the inside out, reverse order from request middlewares)
* finally builds a return value and calls the callback function to the web server

1. **When to use iterator in Django ORM ?**

The iterator is used when  processing results that take up a large amount of available memory (lots of small objects or fewer large objects).

1. **What are signals in Django ?**Signals allow certain senders to notify a set of receivers that some action has taken place. They’re especially useful when many pieces of code may be interested in the same events.
2. **How to implement social login authentication in Django ?**Run the development server to make sure all is in order. The install python-social-auth using the pip install command. Update settings.py to include/register the library in the project  Update the database by  making migrations. Update the Project’s urlpatterns in urls.py to include the main auth URLs. Create a new app <https://apps.twitter.com/app/new> and make sure to use the callback url <http://127.0.0.1:8000/complete/twitter>. In the project directory, add a config.py file and grab the consumer key and consumer secret and add them to the config file. Finally add urls to the config file to specify the login and redirect urls. Do a sanity check and add friendly views.
3. **Where to store static files in django ?**

Static files are stored in the folder called **static**in the Django app.

1. **Types of Session handling methods in Django**

Session handling types  include database-backed sessions, Using cached sessions, Using file-based sessions, and Using cookie-based sessions.

1. **What does Of Django Field Class types do?**

* The database column type
* The widget to use ’s forms and admin site,.
* he minimal validation requirements, which are used ’s admin interface and by forms.

1. **How to use file based sessions?**

To use file-based sessions, set the [**SESSION\_ENGINE**](https://docs.djangoproject.com/en/2.2/ref/settings/#std:setting-SESSION_ENGINE) setting to **“django.contrib.sessions.backends.file”** or the [**SESSION\_FILE\_PATH**](https://docs.djangoproject.com/en/2.2/ref/settings/#std:setting-SESSION_FILE_PATH) setting (which defaults to output from **tempfile.gettempdir()**, most likely **/tmp**) to control where Django stores session files. Be sure to check that your Web server has permissions to read and write to this location.

1. **Is Django Open source ?**

Yes

1. **How to set and unset session in django ?**

To set a session , set a key and a value in the session object. To unset a session Django, use the in-built Python **del** keyword to delete the session by specifying the key.

1. **How to create views in django ?**

To create views , create and activate a virtual environment.  Create a file called **views.py** for example and create view functions to render views using the **HttpResponse** object in this file and  map URLs to views by adding the path to **urlpatterns**in the **urls.py**file.

1. **What is url mapping and how to do it in django ?**

To map an URL,  open the **urls.py**file and add a path entry to the **urlpatterns** array specifying the path string, the view function, and the name of the view function.

1. **What are population websites built on django ?**

Sites include Disqus, Instagram, Knight Foundation, MacArthur Foundation.

1. **How to create custom sql query in django ?**

To create custom sql query ,use the database connection, call **connection.cursor()** to get a cursor object. Then, call **cursor.execute(sql, [params])** to execute the SQL.

1. **Types of Model relationships in django ?**

Types of model relationships  include: One to many, Many to many,  One to one model relationships.

1. **Types of inheritances in django ?**

Types of inheritances  include: Single Table Inheritance , Multi Table Inheritance and ContentTypes

1. **What is context in django ?**

Context is a dictionary with variable names as the **key** and their values as the **value**. A Context object is the *context*in which the template is being rendered.

1. **How to fetch data through objects in django ?**

Use  the model to access the database and Django will execute the SQL.

1. **What is QuerySet in django ?**

**queryset** is a collection of (sql) queries, in your example above print(b.**query**) will show you the sql **query** generated from your **django** filter calls.

1. **How to make Foreign Key relation in django ?**

To make a foreign key relation , create a many-to-one relationship by calling the **ForeignKey()**function on the model with a value of the key’s name.

1. **How to setup django and apache in server ?**

It is done by using mod\_wsgi. Make sure you have Apache and mod\_wsgi installed.  Let’s say we want to share our project (myproject) with Apache.Set Apache to access our folder. Assume we put our myproject folder in the default “/var/www/html”. At this stage, accessing the project will be done via 127.0.0.1/myproject. Configure Apache in httpd.conf. By adding a <Files wsgi.py>  file configuration in it.

**Q1).Explain what is Django?**  
**Ans:** [Django](https://www.mytectra.com/python-django-training-in-bangalore.html) is a web framework in python to develop a web application in python.  
Django is a free and open source web application framework, written in Python.

**Q2).Mention what are the features available in Django?**  
**Ans:** Features available in Django are

* Admin Interface (CRUD)
* Templating
* Form handling
* Internationalization
* Session, user management, role-based permissions
* Object-relational mapping (ORM)
* Testing Framework
* Fantastic Documentation

**Q3).Mention the architecture of Django architecture?**  
**Ans:** Django architecture consists of

* **Models:** It describes your database schema and your data structure
* **Views:** It controls what a user sees, the view retrieves data from appropriate models and execute any calculation made to the data and pass it to the template
* **Templates:** It determines how the user sees it. It describes how the data received from the views should be changed or formatted for display on the page
* **Controller:** The [Django](https://www.mytectra.com/python-django-training-in-bangalore.html) framework and URL parsing

**Q4).Why Django should be used for web-development?**  
**Ans:**

* It allows you to divide code modules into logical groups to make it flexible to change
* To ease the website administration, it provides auto-generated web admin
* It provides pre-packaged API for common user tasks
* It gives you template system to define HTML template for your web page to avoid code duplication
* It enables you to define what URL be for a given function
* It enables you to separate business logic from the HTML
* Everything is in python

**Q5). Explain how you can create a project in Django?**  
**Ans:**To start a project in Django, you use command $ django-admin.py and then use the command  
**Project**  
**\_init\_.py**  
**manage.py**  
**settings.py**  
**urls.py**

**Q6). Explain how you can set up the Database in**[**Django**](https://www.mytectra.com/python-django-training-in-bangalore.html)**?**  
**Ans:**You can use the command edit **mysite/setting.py**, it is a normal python module with module level representing Django settings.  
Django uses SQLite by default; it is easy for Django users as such it won’t require any other type of installation. In the case your database choice is different that you have to the following keys in the **DATABASE ‘default’**item to match your database connection settings

* **Engines:** you can change database by using ‘django.db.backends.sqlite3’ , ‘django.db.backeneds.mysql’, ‘django.db.backends.postgresql\_psycopg2’, ‘django.db.backends.oracle’ and so on
* **Name:**The name of your database. In the case if you are using SQLite as your database, in that case database will be a file on your computer, Name should be a full absolute path, including file name of that file.

If you are not choosing SQLite as your database then setting like Password, Host, User, etc. must be added.

**Q7). Give an example how you can write a VIEW in**[**Django**](https://www.mytectra.com/python-django-training-in-bangalore.html)**?**  
**Ans:**Views are Django functions that take a request and return a response.  To write a view in Django we take a simple example of “Guru99\_home” which uses the template Guru99\_home.html and uses the date-time module to tell us what the time is whenever the page is refreshed.  The file we required to edit is called view.py, and it will be inside mysite/myapp/

Copy the below code into it and save the file  
**from datatime import datetime**  
**from django.shortcuts import render**  
**def home (request):**  
**return render(request, ‘Guru99\_home.html’, {‘right\_now’: datetime.utcnow()})**  
Once you have determined the VIEW, you can uncomment this line in urls.py  
**# url ( r ‘^$’ , ‘mysite.myapp.views.home’ , name ‘Guru99’),**  
The last step will reload your web app so that the changes are noticed by the web server.

**Q8).Explain how you can setup static files in Django?**  
**Ans:**There are three main things required to set up static files in Django

* Set STATIC\_ROOT in settings.py
* run manage.py collectsatic
* set up a Static Files entry on the **PythonAnywhere** web tab

**Q9).Mention what does the Django templates consists of?**

**Ans:**The template is a simple text file.  It can create any text-based format like XML, CSV, HTML, etc.  A template contains variables that get replaced with values when the template is evaluated and tags (% tag %) that controls the logic of the template.

**Q10). Explain the use of session framework in Django?**  
**Ans:** In Django, the session framework enables you to store and retrieve arbitrary data on a per-site-visitor basis.  It stores data on the server side and abstracts the receiving and sending of cookies.  Session can be implemented through a piece of middleware.

**Q11).Explain how you can use file based sessions?**  
**Ans:**To use file based session you have to set the **SESSION\_ENGINE**settings to “django.contrib.sessions.backends.file”

**Q12).Explain the migration in Django and how you can do in SQL?**  
**Ans:**Migration in Django is to make changes to your models like deleting a model, adding a field, etc. into your database schema.  There are several commands you use to interact with migrations.

* Migrate
* Makemigrations
* Sqlmigrate

To do the migration in SQL, you have to print the SQL statement for resetting sequences for a given app name.  
**django-admin.py sqlsequencreset**  
Use this command to generate SQL that will fix cases where a sequence is out sync with its automatically incremented field data.

**Q13).Mention what command line can be used to load data into Django?**  
**Ans:**To load data into Django you have to use the command line**Django-admin.py loaddata.**The command line will searches the data and loads the contents of the named fixtures into the database.

**Q14).Explain what does django-admin.py makemessages command is used for?**  
**Ans:**This command line executes over the entire source tree of the current directory and abstracts all the strings marked for translation.  It makes a message file in the locale directory.

**Q15). List out the inheritance styles in**[**Django**](https://www.mytectra.com/python-django-training-in-bangalore.html)**?**  
**Ans:**In Django, there is three possible inheritance styles

* **Abstract base classes:** This style is used when you only wants parent’s class to hold information that you don’t want to type out for each child model
* **Multi-table Inheritance:** This style is used If you are sub-classing an existing model and need each model to have its own database table
* **Proxy models:** You can use this model, If you only want to modify the Python level behavior of the model, without changing the model’s fields

**Q16).Mention what does the Django field class types?**  
**Ans:**Field class types determines

* The database column type
* The default HTML widget to avail while rendering a form field
* The minimal validation requirements used in Django admin and in automatically generated forms

**Q17).What constitutes  Django templates ?**  
**Ans**: Template can create formats like XML,HTML and CSV(which are text based formats). In general terms template is a simple text file. It is made up of variables that will later be replaced by values after the template is evaluated and has tags which will control template’s logic.

**Q18).List some typical usage of middlewares in Django.**  
**Ans:**Some of the typical usage of middlewares in Django are: Session management, user authentication, cross-site request forgery protection, content Gzipping, etc.

**Q19).How do you use views in Django?**  
**Ans:**Views will take request to return response.  Let’s write a view in Django :  “example” using template example.html , using  the date-time module to tell us exact time of reloading the page.  Let’s edit a file called view.py, and it will be inside randomsite/randomapp/  
**To do this save and copy following into a file:**  
Default

from datatime import datetime

from django.shortcuts import render

def home (request):

return render(request, ‘Guru99\_home.html’, {‘right\_now’: datetime.utcnow()})  
Default

You have to determine the  VIEW first, and then uncomment this line located in file *urls.py*

**# url ( r ‘^$’ , ‘randomsite.randomapp.views.home’ , name ‘example’),**

**Q20).How do you make a Django app that is test driven and will display Fibonacci’s sequence?**This will reload the site making changes obvious.

**Ans20:**Keep in mind that it should take an index number and output the sequence. Additionally, there should be a page that shows the most recent generated sequences.

Following is one of the solution for generating fibonacci series:  
Default

def fib(n):

“Complexity: O(log(n))”

if n <= 0:

return 0

i = n – 1

(a, b) = (1, 0)

(c, d) = (0, 1)

while i > 0:

if i % 2:

(a, b) = (d \* b + c \* a,  d \* (b + a) + c \* b)

(c, d) = (c \* c + d \* d, d \* (2 \* c + d))

i = i / 2

return a + b  
Default

Below is a model that would keep track of latest numbers:

from django.db import models

class Fibonacci(models.Model):

parameter = models.IntegerField(primary\_key=True)

result = models.CharField(max\_length=200)

time = models.DateTimeField()  
DefaultFor view, you can simply use the following code:

from models import Fibonacci

def index(request):

result = None

if request.method==”POST”:

try:

n=int(request.POST.get(‘n’))

except:

return Http404

try:

result = Fibonacci.objects.get(pk=n)

result.time = datetime.now()

except DoesNotExist:

result = str(fib(n))

result = Fibonacci(n, result, datetime.now())

result.save()

return direct\_to\_template(request, ‘base.html’, {‘result’:result.result})  
You could use models to get last ‘n’ entities.

**Q21).What makes up Django architecture?**  
**Ans:**Django runs on MVC architecture. Following are the components that make up django architecture:

* **Models:** Models elaborate back-end stuffs like database schema.(relationships)
* **Views:**Views control what is to be shown to end-user.
* **Templates:** Templates deal with formatting of view.
* **Controller:** Takes entire control of Models.A MVC framework can be compared to a Cable TV with remote. A Television set is View(that interacts with end user), cable provider is model(that works in back-end) and Controller is remote that controls which channel to select and display it through view.

**Q22). What does session framework do in django framework ?**  
**Ans:** Session framework in django will store data on server side and interact with end-users. Session is generally used with a middle-ware. It also helps in receiving and sending cookies for authentication of a user.

**Q23).Can you create singleton object in python?If yes, how do you do it?**  
**Ans:**Yes, you can create singleton object. Here’s how you do it :

Default

|  |  |
| --- | --- |
| 1  2  3  4  5 | class Singleton(object):  def \_\_new\_\_(cls,\*args,\*\*kwargs):  if not hasattr(cls,’\_inst’):  cls.\_inst = super(Singleton,cls).\_\_new\_\_(cls,\*args,\*\*kwargs)  return cls.\_inst |

**Q24).Mention caching strategies that you know in Django!**  
**Ans:** Few caching strategies that are available in Django are as follows:

* File sytem caching
* In-memory caching
* Using Memcached
* Database caching

**Q25).What are inheritance type in Django?**  
**Ans:**There are 3 inheritance types in Django

* Abstract base classes
* Multi-table Inheritance
* Proxy models

**Q26).What do you think are limitation of**[**Django**](https://www.mytectra.com/python-django-training-in-bangalore.html)**Object relation mapping(ORM) ?**

**Ans:**If the data is complex and consists of multiple joins using the SQL  will be clearer.

If Performance is a concern for your, ORM aren’t your choice. Genrally. Object-relation-mapping are considered good option to construct an optimized query, SQL has an upper hand when compared to ORM.

**Q27):How to Start Django project with ‘Hello World!’? Just say hello world in django project.**  
**Ans:** **There are 7 steps ahead to start Django project.**

**Step 1:** Create project in terminal/shell

f2finterview:~$ django-admin.py startproject sampleproject

**Step 2:** Create application

f2finterview:~$ cd sampleproject/

f2finterview:~/sampleproject$ python manage.py startapp sampleapp

**Step 3:** Make template directory and index.html file

f2finterview:~/sampleproject$ mkdir templates

f2finterview:~/sampleproject$ cd templates/

f2finterview:~/sampleproject/templates$ touch index.html

**Step 4:** Configure initial configuration in settings.py

Add PROJECT\_PATH and PROJECT\_NAME

import os

PROJECT\_PATH = os.path.dirname(os.path.abspath(\_\_file\_\_))

PROJECT\_NAME = ‘sampleproject’

Add Template directories path

TEMPLATE\_DIRS = (

os.path.join(PROJECT\_PATH, ‘templates’),

)

Add Your app to INSTALLED\_APPS

INSTALLED\_APPS = (

‘sampleapp’,

)

**Step 5:** Urls configuration in urls.py

from django.conf.urls.defaults import patterns, include, url

urlpatterns = patterns(”,

url(r’^$’, ‘sampleproject.sampleapp.views.index’, name=’index’),

)

**Step 6:** Add index method in views.py

from django.shortcuts import render\_to\_response, get\_object\_or\_404

from django.template import RequestContext

def index(request):

welcome\_msg = ‘Hello World’

return render\_to\_response(‘index.html’,locals(),context\_instance=RequestContext(request))

**Step7:** Add welcome\_msg in index.html

<!DOCTYPE html>

<html>

<body>

<h1>My First Heading For Say…</h1>

<p>{{welcome\_msg}}</p>

</body>

</html>

**Q28).How to login with email instead of username in Django?**  
**Ans:**Use bellow sample method to login with email or username.

from django.conf import settings  
from django.contrib.auth import authenticate, login, REDIRECT\_FIELD\_NAME  
from django.shortcuts import render\_to\_response  
from django.contrib.sites.models import Site  
from django.template import Context, RequestContext  
from django.views.decorators.cache import never\_cache  
from django.views.decorators.csrf import csrf\_protect  
@csrf\_protect  
@never\_cache  
def signin(request,redirect\_field\_name=REDIRECT\_FIELD\_NAME,authentication\_form=LoginForm):  
redirect\_to = request.REQUEST.get(redirect\_field\_name, settings.LOGIN\_REDIRECT\_URL)  
form = authentication\_form()  
current\_site = Site.objects.get\_current()  
if request.method == “POST”:  
pDict =request.POST.copy()  
form = authentication\_form(data=request.POST)  
if form.is\_valid():  
username = form.cleaned\_data[‘username’]  
password = form.cleaned\_data[‘password’]  
try:  
user = User.objects.get(email=username)  
username = user.username  
except User.DoesNotExist:  
username = username  
user = authenticate(username=username, password=password)  
# Log the user in.  
login(request, user)  
return HttpResponseRedirect(redirect\_to)  
else:  
form = authentication\_form()  
request.session.set\_test\_cookie()  
if Site.\_meta.installed:  
current\_site = Site.objects.get\_current()  
else:  
current\_site = RequestSite(request)  
return render\_to\_response(‘login.html’,locals(), context\_instance=RequestContext(request))

**Q29).How Django processes a request?**  
**Ans:** When a user requests a page from your Django-powered site, this is the algorithm the system follows to determine which Python code to execute:  
Django determines the root URLconf module to use. Ordinarily, this is the value of the ROOT\_URLCONF setting, but if the incoming HttpRequest object has an attribute called urlconf (set by middleware request processing), its value will be used in place of the ROOT\_URLCONF setting.  
Django loads that Python module and looks for the variable urlpatterns. This should be a Python list, in the format returned by the function django.conf.urls.patterns()  
Django runs through each URL pattern, in order, and stops at the first one that matches the requested URL.

Once one of the regexes matches, Django imports and calls the given view, which is a simple Python function (or a class based view). The view gets passed an HttpRequest as its first argument and any values captured in the regex as remaining arguments.

If no regex matches, or if an exception is raised during any point in this process, Django invokes an appropriate error-handling view.

**Q30).How to filter latest record by date in Django?**  
**Ans: Messages(models.Model):**  
    message\_from = models.ForeignKey(User,related\_name=”%(class)s\_from”)  
message\_to = models.ForeignKey(User,related\_name=”%(class)s\_to”)  
message=models.CharField(max\_length=140,help\_text=”Your message”)  
created\_on = models.DateTimeField(auto\_now\_add=True)  
class Meta:  
db\_table = ‘messages’

**Query**:messages = Messages.objects.filter(message\_to = user).order\_by(‘-created\_on’)[0]

**Output:**

message\_from | message\_to  | message                 | created\_on

——————|—————–|——————–|——————–

Stephen        | Anto              | Hi, How are you? | 2012-10-09 14:27:48

**Q31).How to filter data from Django models using python datetime?**  
**Ans:** Assume Bellow model for storing messages with timelines  
class Message(models.Model):  
from = models.ForeignKey(User,related\_name = “%(class)s\_from”)  
to = models.ForeignKey(User, related\_name = “%(class)s\_to”)  
msg = models.CharField(max\_length=255)  
rating = models.IntegerField(blank=’True’,default=0)  
created\_on = models.DateTimeField(auto\_now\_add=True)  
updated\_on = models.DateTimeField(auto\_now=True)  
Filter messages with specified Date and Time  
today = date.today().strftime(‘%Y-%m-%d’)

yesterday = date.today() – timedelta(days=1)  
yesterday = yesterday.strftime(‘%Y-%m-%d’)

this\_month = date.today().strftime(‘%m’)  
last\_month = date.today() – timedelta(days=32)  
last\_month = last\_month.strftime(‘%m’)  
this\_year = date.today().strftime(‘%Y’)

last\_year = date.today() – timedelta(days=367)  
last\_year = last\_year.strftime(‘%Y’)

today\_msgs = Message.objects.filter(created\_on\_\_gte=today).count()  
yesterday\_msgs = Message.objects.filter(created\_on\_\_gte=yesterday).count()  
this\_month\_msgs = Message.objects.filter(created\_on\_\_month=this\_month,created\_on\_\_year=this\_year).count()  
last\_month\_msgs = Message.objects.filter(created\_on\_\_month=last\_month,created\_on\_\_year=this\_year).count()  
this\_year\_msgs = Message.objects.filter(created\_on\_\_year=this\_year).count()  
last\_year\_msgs = Message.objects.filter(created\_on\_\_year=last\_year).count()

**Q32). What does Django mean?**  
**Ans:** Django is named after Django Reinhardt, a gypsy jazz guitarist from the 1930s to early 1950s who is known as one of the best guitarists of all time.

**Q33). Which architectural pattern does Django Follow?**  
**Ans:** Django follows Model-View Controller (MVC) architectural pattern.

**Q34). Is Django a high level web framework or low level framework?**  
**Ans:** Django is a high level Python’s web framework which was designed for rapid development and clean realistic design.

**Q35). How would you pronounce Django?**  
Django is pronounced JANG-oh. Here D is silent.

**Q36). How does Django work?**  
**Ans:** Django can be broken into many components:  
**Models.py file:** This file defines your data model by extending your single line of code into full database tables and add a pre-built administration section to manage content.  
**Urls.py file:** It uses regular expression to capture URL patterns for processing.  
**Views.py file:** It is the main part of Django. The actual processing happens in view.  
When a visitor lands on Django page, first Django checks the URLs pattern you have created and uses information to retrieve the view. After that view processes the request, querying your database if necessary, and passes the requested information to template.  
After that the template renders the data in a layout you have created and displays the page.

**Q37).Which foundation manages Django web framework?**  
**Ans:** Django web framework is managed and maintained by an independent and non-profit organization named Django Software Foundation (DSF).

**Q38).Is Django stable?**  
**Ans:** Yes, Django is quite stable. Many companies like Disqus, Instagram, Pinterest, and Mozilla have been using Django for many years.

**Q39)What are the features available in Django web framework?**  
**Ans:** Features available in [Django](https://www.mytectra.com/python-django-training-in-bangalore.html)web framework are:

* Admin Interface (CRUD)
* Templating
* Form handling
* Internationalization
* Session, user management, role-based permissions
* Object-relational mapping (ORM)
* Testing Framework
* Fantastic Documentation

**Q40). What are the advantages of using Django for web development?**  
**Ans:**

* It facilitates you to divide code modules into logical groups to make it flexible to change.
* It provides auto-generated web admin to make website administration easy.
* It provides pre-packaged API for common user tasks.
* It provides template system to define HTML template for your web page to avoid code duplication.
* It enables you to define what URL is for a given function.
* It enables you to separate business logic from the HTML.

**Q41) How to create a project in Django?**  
**Ans:** To start a project in Django, use the command **$django-admin.py** and then use the following command:  
**Project**  
**\_init\_.py**  
**manage.py**  
**settings.py**  
**urls.py**

**Q42) What are the inheritance styles in Django?**  
**Ans: There are three possible inheritance styles in Django:**  
**1) Abstract base classes:** This style is used when you only want parent’s class to hold information that you don’t want to type out for each child model.  
**2) Multi-table Inheritance:** This style is used if you are sub-classing an existing model and need each model to have its own database table.  
**3) Proxy models:** This style is used, if you only want to modify the Python level behavior of the model, without changing the model’s fields.

**How can you set up the database in Djanago**?  
**A:** To set up a database in Django, you can use the command edit mysite/setting.py , it is a normal python module with module level representing Django settings.  
By default, Django uses SQLite database. It is easy for Django users because it doesn’t require any other type of installation. In the case of other database you have to the following keys in the DATABASE ‘default’ item to match your database connection settings.

**Engines:** you can change database by using ‘django.db.backends.sqlite3’ , ‘django.db.backeneds.mysql’, ‘django.db.backends.postgresql\_psycopg2’, ‘django.db.backends.oracle’ and so on

**Name:** The name of your database. In the case if you are using SQLite as your database, in that case database will be a file on your computer, Name should be a full absolute path, including file name of that file.  
**Note:** You have to add setting likes setting like Password, Host, User, etc. in your database, if you are not choosing SQLite as your database.

**Q43).What does the Django templates contain?**  
**Ans:** A template is a simple text file. It can create any text-based format like XML, CSV, HTML, etc. A template contains variables that get replaced with values when the template is evaluated and tags (%tag%) that controls the logic of the template.

**Q44).Is Django a content management system (CMS)?**  
**Ans:** No, Django is not a CMS. Instead, it is a Web framework and a programming tool that makes you able to build websites.

**Q45).What is the use of session framework in Django?**  
**Ans**: The session framework facilitates you to store and retrieve arbitrary data on a per-site visitor basis. It stores data on the server side and abstracts the receiving and sending of cookies. Session can be implemented through a piece of middleware.

**Q46).How can you set up static files in Django?**  
**Ans:** There are three main things required to set up static files in Django:  
1) Set STATIC\_ROOT in settings.py  
2) run manage.py collectsatic  
3) set up a Static Files entry on the PythonAnywhere web tab

**Q47). How to use file based sessions?**  
**Ans:**You have to set the SESSION\_ENGINE settings to “django.contrib.sessions.backends.file” to use file based session.

**Q48).What is some typical usage of middlewares in Django?**  
**Ans:** Some usage of middlewares in Django is:

* Session management,
* Use authentication
* Cross-site request forgery protection
* Content Gzipping, etc.

**Q49).What does of Django field class types do?**  
**Ans:** The Django field class types specify:

* The database column type.
* The default HTML widget to avail while rendering a form field.
* The minimal validation requirements used in Django admin.
* Automatic generated forms.

**Q50).What is the usage of Django-admin.py and manage.py?**  
**Ans: Django-admin.py:** It is a Django’s command line utility for administrative tasks.  
**Manage.py:** It is an automatically created file in each Django project. It is a thin wrapper around the Django-admin.py. It has the following usage:

* It puts your project’s package on sys.path.
* It sets the DJANGO\_SETTING\_MODULE environment variable to points to your project’s setting.py file.

**Q51).What are signals in Django?**  
**Ans:** Signals are pieces of code which contain information about what is happening. Dispatcher is used to send the signals and listen for those signals.

**Q52).What are the two important parameters in signals?**  
**Ans:** Two important parameters in signals are:

* **Receiver:** It specifies the callback function which will be connected to the signal.
* **Sender:** It specifies a particular sender to receive signal from.

**Q53).What command line is used to load data into Django?**  
**Ans:** The command line “Django-admin.py loaddata” is used to load data into Django. The command line will searches the data and loads the contents of the named fixtures into the database.

**What kind of calling model does Python use?**

* **What most people say:**“Python uses call-by-reference,” or, “Python uses call-by-value.”
* **What you should say:**“Actually, Python uses call-by-object.”
* **Why you should say it:**This question separates the veterans from the rookies, according to Wendt. Only developers with considerable hands-on experience typically know about Python’s unique calling model.

**What is Unicode, what is UTF-8 and how do they relate?**

* **What most people say:**“Unicode has something to do with special characters, right?”
* **What you should say:**“Unicode is an international encoding standard that works with different languages and scripts. It consists of letters, digits or symbols representing characters from across the world. UTF-8 is a type of encoding, a way of storing the code points of Unicode in a byte form, so you can send Unicode strings over the network or store them in files.”
* **Why you should say it:**Since developers must create code for a world comprised of many cultures and languages, you should know how to develop an application that will be used by non-English speakers. Knowing about Unicode/UTF-8 shows that you understand the importance of global development protocols.

**How do you decide when to reuse code and when to start from scratch?**

* **What most people say:**“I typically search GitHub, Bitbucket and PyPI (Python Package Index). If I find something I like, I’ll copy the code. If I don’t find a viable solution online, I’ll create fresh code.”
* **What you should say:**“Creating brand new code is a last resort. I’ll research code libraries, using several criteria to decide whether I should integrate existing code into my project. For instance, I’ll consider the quality of the code, the reputation and activity of the developer as well as the efficacy and size of the coding community. I want to know whether the developer is generating timely updates and notes, how quickly bugs are being fixed, and whether the code has received recent updates.”
* **Why you should say it:**A good engineer’s goal is to write and maintain the least possible amount of code so that they can focus on other things, like making their product unique. However, the decision to incorporate existing code requires careful consideration. If you’re unable to find a quality solution online, sometimes it’s better to rethink the problem.

**How would you scale an existing application when starting a new project?**

* **What most people say:**“I’d use a NoSQL data store to store my data. Also, I’d cache a good portion of the data in Redis or Memcached through Django’s caching facilities. Perhaps I’d put a reverse proxy like Varnish in front of it.”
* **What you should say:**“I see performance and scaling as two separate things. Performance is how fast a user is served and scaling refers to the number of users that can be served by an app at the same time. Usually, time is best spent developing during the early stages of a project. When scale does become an issue, usually business is good and there are sufficient funds to optimize the application.”
* **Why you should say it:**First of all, as an engineer it’s always good to take a step back and analyze the question instead of jumping to conclusions. The best way to answer this one is by explaining what scaling is and to question whether it’s required in the early stages of a project. A good engineer asks the right questions before rendering a decision. These are the type of characteristics employers are looking for.

**Are there situations where you wouldn’t use Python/Django**?

* **What most people say:**“Not really. Python is a very good, generic programming language and Django has so many options, that it works for any Web application.”
* **What you should say:**“Sure. For example, if a project involves some kind of reasoning it might be better to use Prolog and have Python interface with it. Of course, I would be mindful about adding more complexity to the stack by introducing a new language.”
* **Why you should say it:**Every programming language or framework has its strengths and weaknesses. Good engineers don’t become emotionally attached to a language, and weigh several options before making a decision. However, they also realize that adding a new programming language increases complexity. So, they constantly ask themselves if adding another language is really worth it.

Django along with [Python](https://www.edureka.co/blog/python-programming-language) is one of the most in-demand skills and surely amongst some of the trickiest ones. So if you want to prepare yourself to perform the best in the upcoming [Django](https://www.edureka.co/blog/django-tutorial/) interview, here are the top 50 commonly asked Django Interview Questions and Answers.

Before moving on, let’s take a look at the top 10 Django Interview Questions:

[Q1. What is the difference between Flask and Django?](https://www.edureka.co/blog/interview-questions/django-interview-questions/#flaskvsdjango)  
[Q2. What is Django?](https://www.edureka.co/blog/interview-questions/django-interview-questions/#django)  
[Q3. Do you know any companies that use Django?](https://www.edureka.co/blog/interview-questions/django-interview-questions/#companies)  
[Q4. What are the features of Django?](https://www.edureka.co/blog/interview-questions/django-interview-questions/#features)  
[Q5. How do you check for the version of Django installed on your system?](https://www.edureka.co/blog/interview-questions/django-interview-questions/#version)  
[Q6. What are the advantages of using Django?](https://www.edureka.co/blog/interview-questions/django-interview-questions/#advantages)  
[Q7. Explain Django’s architecture.](https://www.edureka.co/blog/interview-questions/django-interview-questions/#architecture)  
[Q8. Give a brief about the Django admin.](https://www.edureka.co/blog/interview-questions/django-interview-questions/#admin)  
[Q9. How do you connect your Django Project to the database?](https://www.edureka.co/blog/interview-questions/django-interview-questions/#databaseconnection)  
[Q10. What are the various files that are created when you create a Django Project? Explain briefly.](https://www.edureka.co/blog/interview-questions/django-interview-questions/#files)

**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/), 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://www.edureka.co/blog/django-tutorial/#installation), 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:

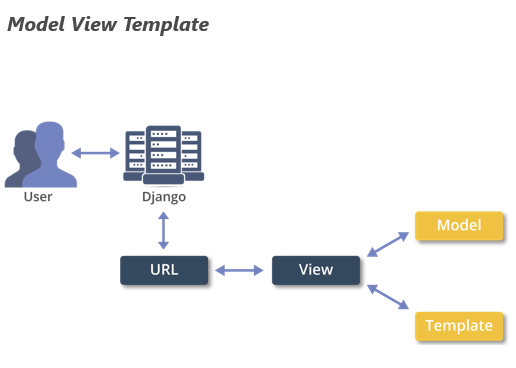
|  |  |
| --- | --- |
| 1  2 | import django  print(django.get\_version()) |

**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.

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/) 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.**

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/), 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.

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

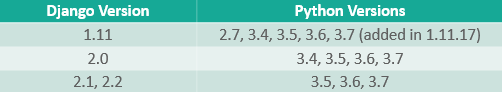
Yes. Hardware is much cheaper when compared to the development time and this is why Django is designed to make full use of any amount of hardware that you can provide it. Django makes use of a “shared-nothing” architecture meaning you can add hardware at any level i.e database servers, caching servers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

**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.

With this, we have reached the end of this article on Django Interview Questions. I hope you are clear with all that has been shared with you in this article. ***Make sure you practice as much as possible and revert your experience.***