* **Data Structure and Algorithm:**

1. **What is the difference between linear and non-linear data structure?**

**→** The linear and non-linear data structure is the subclassification of the data structure which comes under the Non-primitive data structure. The crucial difference between them is that the linear data structure arranges the data into a sequence and follow some sort of order. Whereas, the non-linear data structure does not organize the data in a sequential manner.

Read more [here](https://techdifferences.com/difference-between-linear-and-non-linear-data-structure.html)

1. **What is primitive and non-primitive data type?**

**→** A primitive data type is one that fits the base architecture of the underlying computer such as int, float, and pointer, and all of the variations, thereof such as char short long unsigned float double and etc, are primitive data type. Primitive data are only single values, they have not special capabilities.

Read more [here](https://www.quora.com/What-is-the-difference-between-primitive-data-structure-and-non-primitive-data-structure)

1. What is recursion?
2. Why recursion?
3. Tell me one real time example of recursion?
4. What is binary tree?
5. **Does Python have a linked list?**

→ It “Depends”. As of Python 3.6 (CPython), doesn’t provide a dedicated linked list data type.

Python does however include the collections.deque class which provides a [double-ended queue](https://dbader.org/blog/queues-in-python) and is implemented as a doubly-linked list internally. Under some specific circumstances you might be able to use it as a “makeshift” linked list. If that’s not an option you’ll need to write your own linked list implementation from scratch.

Ref - [dbader](https://dbader.org/blog/python-linked-list)

1. What is linked list? Print each element of given linked list.
2. **Why doubly link list is useful?**

→ Having a reference to the previous element can speed up some operations, like removing (“unlinking”) an element from a list or traversing the list in reverse order.

* **DB:**

1. Have you handled multiple database in your project?
2. Explain postgres JOINs.
3. What is DB routing?

* **NoSQL DB:**

1. What is NoSQL database?
2. Where you used mongodb? What was the purpose of using mongodb?

* **RESTful Web Services:**

1. What is RESTful web services? Or What is REST API?
2. Why REST API?
3. What is PATCH request in REST API?
4. What is API Authentication?
5. What is session based authentication?
6. What is token based authentication?
7. What is the difference between session based and token based authentication?

* **Django Framework:**

1. What is MVC? Why Django called as MVCT framework?
2. How DB connection happens in Django? Explain?
3. How Django handles multiple DB?
4. How to design frontend in Django?
5. What is ORM? What is django ORM? Why Django ORM needed?
6. How request flows in django? / How Django handle request comes from browser?
7. **What is middleware in django and What are the different middleware in Django? Why we need middleware? What if we don’t want to use middleware? How to create your own middleware(custom middleware)?**

→ [Simpleisbetterthancomplex: Middleware](https://simpleisbetterthancomplex.com/tutorial/2016/07/18/how-to-create-a-custom-django-middleware.html) , [Djangoproject: Middleware](https://docs.djangoproject.com/en/2.1/topics/http/middleware/)

1. What is serializers in django?
2. What is the purpose of using serializers? And How they works?
3. How it works when we make a request/response? I.e Suppose in request on each call I want to add a flag to variable, How middleware works in this case?
4. What is cache? Why need caching needed?
5. Have you used any server caching mechanism or library?
6. Explain how django uses pattern matching to match request URL in urls.py file?
7. How can we hide API url in browser?
8. Suppose in your for certain request, you need to access data from multiple table(model) and It’ll require lot of queries. How this could be done more efficiently in another way?
9. What are the methods related and pre-related in django model?
10. How can we implement RESTful web services without using Django Rest Framework?
11. What are DRF objects and classes you’ve used in API?
12. What is CSRF token in Django?
13. Session Authentication VS Token Authentication in Django?

→ [Stackoverflow: Session vs Token](https://stackoverflow.com/questions/54169145/django-rest-framework-session-auth-vs-token-auth-csrf)

1. What is the purpose of CORS in django? How we can create custom CORS middleware in Django?

→ [Mozilla: CORS](https://developer.mozilla.org/en-US/docs/Web/HTTP/CORS) , [Techiediaries: CORS](https://www.techiediaries.com/django-cors/)

* **Tornado Framework:**

1. What is tornado framework?
2. For what purpose you used tornado framework in your project?

* **Django and Tornado:**

1. What is the difference between django framework and tornado framework?

* **Django vs DRF:**

1. What is the difference between Django Framework and Django REST Framework(DRF)?

* **Server:**

1. Which server you used?
2. How server(Nginx and Gunicorn) route the request?

→ [digitalocean: serve-flask-app-with-gunicorn-and-nginx-on-ubuntu-16-04](https://www.digitalocean.com/community/tutorials/how-to-serve-flask-applications-with-gunicorn-and-nginx-on-ubuntu-16-04)

1. How to setup Gunucorn and Nginx to serve Flask app?

→ [DigitalOcean: Serve Flask Applications with uWSGI and Nginx Ubuntu 18](https://www.digitalocean.com/community/tutorials/how-to-serve-flask-applications-with-uswgi-and-nginx-on-ubuntu-18-04)

1. How to setup uwsgi and Nginx to serve python app in Ubuntu?

→ [DigitalOcean: setup uwsgi and Nginx to serve python app](https://www.digitalocean.com/community/tutorials/how-to-set-up-uwsgi-and-nginx-to-serve-python-apps-on-ubuntu-14-04#definitions-and-concepts)

* **Task Scheduling:**

1. Do you know about any scheduling library?
2. Why you haven't used celory for task scheduling in your project?

* **Distributed commit log technologies or Distributed streaming platform:**

1. What is Kafka? Or Do you know about kafka?

* **Storage technologies and Clustering:**

1. Do you know about storage technologies?
2. What is clustering?

* **Frontend**:

1. How we can create web pages using HTML, CSS and javascript?
2. What is document.read() function in JavaScript?
3. What is AJAX? What is the purpose of using AJAX?
4. What is reactjs? Is it for client side or server side programing?
5. How react manages its state?
6. What is redux? What is the purpose of using redux in a project?

* **Backend + Frontend:**

1. How REST API and react frontend work together?

* **ML | Data Science**

[15 Data sets for ML projects](https://www.datasciencelearner.com/datasets-for-machine-learning-projects-data-scientist/), [Mediam: 50 Datasets for ML projects](https://medium.com/datadriveninvestor/the-50-best-public-datasets-for-machine-learning-d80e9f030279)

* **Project:**

1. Explain your project? Or Tell me about your project? How it is solving problem in real time with example?
2. How and where you scheduled jobs?
3. For what purpose you used tornado framework and django framework?
4. How you provide customised solution to each client?

**## Backend**

1. **# Python**

**# Django/Django Rest API**

**# PostgreSQL**

**# Version Control - GIT | Mercireal**

**# Linux**

**# Celery | Tornado | RabbitMQ**

**# NGINX | Gunicorn**

**# DevOps - GIT | Jenkins | Ansible/Saltstack | Docker | Nagios**

**## Frontend**

**# Javascript**

**# React**

**# Redux**