**Python Hiring Topics**

1. **Virtual Env in Python**

A virtual environment, often referred to as a virtualenv, is a tool in Python that allows you to create isolated environments for your Python projects. Each virtual environment has its own Python interpreter and installed packages, independent of the system-wide Python installation. This enables you to manage project-specific dependencies and avoid conflicts between different projects.

Ex:

mkidr myproject

cd myproject

python –m venv myenv

myenv\Scripts\activate

pip install requests

1. **Can Python 2 & 3 be installed on same system**

Yes, it is possible to have both Python 2 and Python 3 installed on the same system. Python 2 and Python 3 are designed to coexist and can be installed and used independently.

1. **Code for creating Custom packages in Python**

mkdir mypackage

cd mypackage

touch \_\_int\_\_.py

def greet():

print("Hello from module1!")

setup.py:

from setuptools import setup, find\_packages

setup(

name='mypackage',

version='1.0',

packages=find\_packages(),

# other metadata like author, description, dependencies, etc.

)

pip install .

import mypackage.module1

mypackage.module1.greet()

python main.py

1. **Sample code for Decorator**

def uppercase\_decorator(func):

def wrapper():

original\_result = func()

modified\_result = original\_result.upper()

return modified\_result

return wrapper

@uppercase\_decorator

def greet():

return "Hello, World!"

# Calling the decorated function

print(greet())

**Django topics:**

1. **Django Request & Response life cycle**

The Django request and response lifecycle involves the following steps:

1. A user makes a request to a Django web application.

2. The URL dispatcher matches the URL to a corresponding view function or class.

3. The view function or class processes the request and performs necessary operations.

4. The view returns an HTTP response.

5. The response is sent back to the user's browser to be rendered and displayed.

**2. Difference between Authentication & Authorization**

Authentication is the process of verifying the identity of a user, confirming that they are who they claim to be.

Authorization, on the other hand, is the process of granting or denying access to specific resources or functionalities based on the authenticated user's permissions and privileges.

**3. How does the Communication between API or implemented**

In simple terms, the communication between APIs or implemented systems follows these steps:

1. The client sends a request to the API or system.
2. The server processes the request and performs the necessary operations.
3. The server generates a response with the requested data.
4. The server sends the response back to the client.
5. The client receives the response and processes it as needed.

This cycle allows the client and server to exchange information and functionality, enabling the client to access the desired data or perform specific actions through the API or implemented system.

1. **What are Routers**

In Django, routers are components that help manage how URLs are mapped to views in an application. They automate the process of URL routing and simplify the handling of URLs and views. Routers are commonly used in Django REST Framework for building APIs.

1. **What is Serializes in django**

In Django, serializes are classes that help convert complex data types, such as model instances, into a format that can be easily transmitted or stored, such as JSON. Serializes handle the process of serialization and deserialization, allowing data to be converted to a desired format and back into Python objects when needed. They are commonly used in Django projects, especially when working with APIs.