ABHISHYANDH GLOBAL SOLUTIONS
LOCUS OF PERFECTION
Aditya Enclave, Nilgiri Block, 6th Floor, 605A,
Beside Ameerpet Metro Station, Ameerpet- Hyderabad.
040-66858189 8897830910
https://abhishyandh.co.in
info@abhishyandh.co.in

PYTHON FULLSTACK

Python Fullstack Course Contents

Topics	Subtopics
Introduction to Python	- Overview of Python - Use Cases in Development
History of Python	- Evolution of Python - Major Versions and Features
Advantages & Disadvantages of Python	- Benefits of Python - Limitations
Introduction to Interpreter	- What is an Interpreter? - Python Interpreter Basics
Python Installation	- Downloading Python - Setting Up Environment Variables
Python Program Execution Flow	- Compilation vs Interpretation - Execution Process
Python Syntaxes	- Indentation - Comments - Code Blocks
Introduction to IDE (Eclipse, PyCharm)	- Setting Up Eclipse - Setting Up PyCharm - Basic IDE Features
Python Program Development	- Writing First Python Program - Debugging Techniques

Identifiers, Keywords	- Naming Conventions - Python Keywords
Basic Data Types	- int, float, str, bool, etc. - Type Checking
Dynamic Typing	- Dynamic Typing Explained - Examples and Usage
Type Conversion Functions	- Implicit vs Explicit Conversion - Common Conversion Functions
Operators	- Arithmetic, Logical, Comparison Operators - Bitwise Operators
Operator Precedence	- Order of Operations - Associativity
Input/Output	- Using input() - Using print() - File I/O Basics
Reading User Input	- Input Methods - Validating User Input
Displaying Output	- Output Methods - Using print() Function
Formatting Output	- String Formatting - format() Method - f-Strings
Variables, Expressions	- Variable Declaration - Expressions and Evaluations
Selection Control Statements (if, if..else)	- Basic if Statement - if..else Statement - Nested if..else
range Data Type, Indexing, and Slicing	- Using range() Function - List Indexing - Slicing Syntax
Iterative Control Statements (while, for..in)	- while Loop - for..in Loop - Nested Loops
Transfer Control Statements (break & continue)	- Using break Statement - Using continue Statement

Python Strings	- String Operations - String Methods - String Formatting
List	- List Creation - List Operations - List Methods
Tuple	- Tuple Creation - Tuple Operations - Tuple Methods
Set	- Set Creation - Set Operations - Set Methods
Dictionary	- Dictionary Creation - Dictionary Operations - Dictionary Methods
Functions	- Function Definition - Function Call - Return Statement
Definition and Advantages of a Function	- Why Use Functions? - Benefits of Code Reusability
Understanding Scope	- Local vs Global Scope - Scope Resolution
Default Arguments	- Default Values in Functions - Examples
Keyword Arguments	- Using Keyword Arguments - Examples
Variable Length Arguments	- *args and **kwargs - Usage and Examples
Lambda Expressions	- Defining Lambda Functions - Use Cases
Documentation & Annotations	- Writing Docstrings - Function Annotations
Types of Arguments	- Positional, Keyword, Default, Variable-Length Arguments

Scope of Variables	- Understanding Scope - global and nonlocal Keywords
Global Variables	- Declaring Global Variables - Accessing and Modifying Globals
Local Variables	- Defining Local Variables - Scope of Local Variables
Nested Functions	- Defining Functions Inside Functions - Accessing Outer Variables
Lambda Functions	- Understanding Anonymous Functions - Use Cases
Exceptions	- What are Exceptions? - Try, Except Block
Syntax Errors vs Runtime Errors	- Difference Between Syntax and Runtime Errors - Examples
Handling, Raising Exceptions	- Raising Exceptions Manually - Handling Exceptions Gracefully
Built-in Exceptions	- Common Built-in Exceptions (e.g., IOError, ImportError)
MemoryError	- Causes of MemoryError - Handling MemoryError
NameError	- Causes of NameError - Handling NameError
ValueError	- Causes of ValueError - Handling ValueError
TypeError	- Causes of TypeError - Handling TypeError
User-defined Exceptions	- Creating Custom Exceptions - Raising Custom Exceptions
Clean-up Actions	- Using finally Block - Clean-up Actions Post-Exception
Introduction to OOPs	- What is Object-Oriented Programming? - OOP Concepts Overview

OOPs Principles	- Encapsulation - Abstraction - Inheritance - Polymorphism
Classes and Objects	- Defining Classes - Creating Objects - Accessing Class Members
Instance Variables	- Defining Instance Variables - Instance vs Class Variables
Instance Methods	- Defining Instance Methods - Using self Keyword
Class Variables	- Understanding Class Variables - Accessing Class Variables
Class Methods	- Defining Class Methods - Using @classmethod Decorator
Constructors	- **init**() Method - Constructor Overloading
Inheritance	- Single, Multiple, Multilevel Inheritance - Method Overriding
Abstract Classes	- Defining Abstract Classes - Using @abstractmethod Decorator
Inner Classes	- Defining Inner Classes - Accessing Outer Class Members
Exception Handling	- Exception Handling in OOP Context - Best Practices
Regular Expressions	- Introduction to Regex - Using re Module - Pattern Matching
File Handling	- Opening and Closing Files - Reading and Writing Files - File Modes
Accessing (Reading/Writing) Files	- Reading File Content - Writing to a File - Working with Binary Files

Serialization	- What is Serialization? - Using pickle Module - JSON Serialization
Multithreading	- Understanding Threads - Creating and Managing Threads - Synchronization
Introduction to Database	- Overview of Databases - Types of Databases (SQL vs NoSQL)
Python Database Connectivity	- Connecting to Databases with Python - Using sqlite3 Module
CRUD Operations with Database	- Create, Read, Update, Delete Operations - Executing SQL Queries
Unit Testing	- Introduction to Unit Testing - Using unittest Module - Writing Test Cases
Mini Projects Development	- Planning and Developing Mini Projects - Application of Learned Concepts
Debugging	- Debugging Techniques - Using Debugging Tools
Introduction to Web Development	- Basics of Web Development - Understanding Client-Server Architecture
Introduction to Django Framework	- What is Django? - Django's Features - MVC vs MVT Architecture
Features of Django	- Django's Key Features - Why Django is Popular?
Django Installation	- Installing Django - Setting Up Django Environment
MVC Model	- Understanding MVC Architecture - Django's MVT Architecture
HTTP Concepts	- Basics of HTTP - Understanding Request-Response Cycle
Views	- Creating Views in Django - Function-Based vs Class-Based Views

URL Mapping	- Configuring URLs in Django - Using urlpatterns
Templates	- Creating Templates - Rendering Templates
Django Template Language	- Understanding Django's Template Language - Tags, Filters, Variables
Utilities of Templates	- Template Utilities - Context Processors
Creating Template Objects	- Template Object Creation - Rendering Context
Tags, Variables, and Filters	- Using Tags - Using Variables - Using Filters in Templates
Rendering Templates	- Rendering Views - Passing Context Data
Template Inheritance	- Block Tags - Extending Templates - Reusability of Templates
Form Handling	- Form Classes - Form Validations and Error Messages
Form Display	- Rendering Forms in Templates - Customizing Form Appearance
Capturing Form Data	- Retrieving Form Data - Handling POST Data
Advanced Form Processing	- Handling File Uploads - Multi-Page Forms
Django Models	- Django Models - Model Fields - Model Inheritance
Model Inheritance	- Abstract Base Classes - Multi-Table Inheritance - Proxy Models
Primary Keys and the Model	- Setting Primary Keys - Auto-Field vs Custom Primary Keys

Dynamic Webpages	- Dynamic Content Generation - Contextual Data Rendering
Toggle Hidden Content	- Using JavaScript to Toggle Content - Integrating Toggle Functionality
JQuery and AJAX Integration	- Introduction to JQuery - Asynchronous Calls with AJAX
Serialization and Deserialization	- Understanding Serialization - JSON Serialization and Deserialization
Django REST Framework	- Introduction to Django REST Framework - Setting Up REST API
REST Principles	- REST Architecture - Designing RESTful APIs
Serializer Class	- Creating Serializers - Using Serializer Classes
Model Serializers	- Model Serializer vs Regular Serializer - Integrating with Django Models
REST APIs	- Building REST APIs - CRUD Operations with REST
JSON, Parsing Object to JSON and Back	- JSON Basics - Serialization and Deserialization with JSON
Swagger	- Introduction to Swagger - API Documentation with Swagger
POSTMAN	- Using POSTMAN for API Testing - Automating API Tests with POSTMAN
REST Client Development	- Building REST Clients - Consuming REST APIs
REST Security	- Securing REST APIs - Implementing OAuth2, JWT
Page Redirection	- Redirecting Pages - Handling Redirects in Django
Sending Emails	- Sending Emails with Django - Email Configuration

Deploying Django Framework	- Deployment Best Practices - Deploying on AWS/Heroku
Generic Views	- Using Django's Generic Views - Customizing Generic Views
File Uploading	- Handling File Uploads - Storing and Serving Files
Cookie Handling	- Working with Cookies - Setting and Retrieving Cookies
Session Management	- Managing User Sessions - Using Django's Session Framework
Mini Projects Development	- Planning and Developing Mini Projects - Applying Django Concepts
GitHub	- Version Control with Git - Using GitHub for Collaboration
Jenkins	- Continuous Integration with Jenkins - Automating Builds and Deployments