PYTHON FULL STACK DEVELOPER

Duration: 180 Hours

HTML

- HTML Introduction
- Foundation & Structure
- Body Attributes & Break Tags
- Heading Paragraph and Horizontal Rules
- Fonts:Bold & Italics
- Adding Images
- Links Emails & Book Marks
- Creating Lists
- Creating Iframes
- Creating Tables
- Frames & Frameset.
- Forms:Text/Password/Textarea
- Forms:Checkbox/Radio
- Forms:File Upload/Selection List
- Forms:Reset/Submit

CSS

- Introduction to CSS
- External style sheet
- Font Properties
- CSS Comments
- Box Model,Padding.
- Box Model,Border
- Box Model,Margin
- Div & Span
- Working with Colors
- Background Images
- Classes
- ID's

- Child selectors
- Pseudo Classes
- Nth Child Pseudo Classes
- Styling Links
- Styling Lists
- Rounded Corners
- Linear Gradient
- Float & Clear
- New tags for the Layout
- Creating a Website Layout
- Styling Tables
- Styling Form Elements

Java Script

- Introduction
- Statements and Comments
- Finding Errors
- Variables
- Data Types
- Operators
- Strings
- Events in JS
- Functions in JS
- Functions with parameters
- Functions with multiple parameters
- Variable scope
- Arrays in JS
- IF Statement
- IF ELSE Statement
- Nested IF Statement
- Complex Condition
- Switch Statements
- While loops

- Do While loops
- For loops
- Objects,Methods and Properties
- Referencing elements in JS
- Changing an image sources
- Using a timer
- Forms
- Form Values
- Form Submission
- Form Validation TextBoxes and passwords
- Radio button
- Check boxes
- Changing CSS

Jquery

- Introduction
- Syntax
- Selectors
- Event Method
- Chaining
- Show/Hide
- Animate
- Jquery CSS

MYSQL

- MYSQL Basics
- Django DB Connectivity
- MYSQL-UDF,Views,Indexes
- Stored Procedure
- MYSQL - Advanced

Python

- GETTING STARTED
- History & need of Python
- Application of Python

- Advantages of Python
- Disadvantages of Python
- Installing Python
- Program structure
- Interactive Shell
- Executable or script files.
- User Interface or IDE

PYTHON FUNDAMENTALS

- Working with Interactive mode
- Working with Script mode
- Python Character Set
- Python Tokens, Keywords, Identifiers, Literals, Operators Variables and Assignments
- Input and Output in Python

DATA HANDLING

- Data Types
- Numbers
- Strings
- Lists
- Tuples
- Dictionary
- Set
- Frozenset
- Bool
- Mutable and Immutable

STRING MANIPULATION

- Introduction to Python String
- Accessing Individual Elements
- String Operators
- String Slices
- String Functions and Methods

LIST MANIPULATION

- Introduction to Python List
- Creating List
- Accessing List
- Joining List
- Replicating List
- List Slicing

TUPLES

- Introduction to Tuple
- Creating Tuples
- Accessing Tuples
- Joining Tuples
- Replicating Tuples
- Tuple Slicing

DICTIONARIES

- Introduction to Dictionary
- Accessing values in dictionaries
- working with dictionaries
- Properties

OPERATORS

- Arithmetic Operators
- Relational Operators
- Logical Operators
- Membership Operators
- Identity Operators
- Bitwise Operators
- Assignment Operators
- Operators Precedence
- Evaluating Expression
- Type Casting

PROGRAM CONTROL FLOW

- Conditional Statements
- The if Statement
- The if-else Statement
- The if-elif Statement
- Nested if Statements
- Python Indentation
- Looping and Iteration
- The For Loop
- The While Loop
- Loop else Statement
- Nested Loops
- Break and Continue
- The Range Function
- Introduction to range()
- Types of range() function
- Use of range() function

INTRODUCTION TO FUNCTIONS

- Built-In Functions
- Introduction to Functions
- Using a Functions
- Python Function Types
- Structure of Python Functions E.g. - map, zip, reduce, filter, any, chr, ord, sorted, globals,
- locals, all, etc.
- User Defined Functions
- Structure of a Python Program w.r.t. UDF
- Types of Functions
- Invoking UDF
- Flow of Execution
- Arguments and Parameters
- Default Arguments, Named Arguments
- Scope of Variables

- Lambda function
- Recursion Function
- Use of recursion function

Django FrameWork

Django Basics

- Introduction
- Why Django?
- Who's Using Django?
- Installation
- Installing Virtualenv
- Starting a New Project
- Django Apps
- Use Case Diagram
- Class Diagram
- Wireframes
- Models
- Migrating the Models
- Experimenting with the Models API
- Summary of Model's Operations
- Views, Templates, and Static Files
- Django Template Engine Setup
- Testing the Homepage
- Static Files Setup
- Introduction to Django Admin
- URLs
- Basic URLs
- Advanced URLs
- Using the URLs API
- List of Useful URL Patterns

Handling HTTP requests

- URL dispatcher
- Writing views

- View decorators
- File Uploads
- Django shortcut functions
- Generic views
- Middleware
- How to use sessions

Working with forms

- HTML forms
- Django’s role in forms
- Forms in Django
- Building a form
- More about Django **Form** classes
- Working with form templates

Django Templates

- Django Templates
- Template Filters
- Template Tags
- variables
- Boolean Operators
- for loop
- if – Django Templates
- Template Inheritance
- Reusable Templates

Django Views

- Django Views
- Function Based Views
- ○ Create View
- ○ List View
- ○ Detail View
- ○ Update View
- ○ Delete View
- Class Based Generic Views Django

- Introduction to class-based views
- Built-in class-based generic views
- Form handling with class-based views
- Using mixins with class-based views
- Basic examples
- Usage in your URLconf
- Subclassing generic views

Django Models

- Django Models
- ORM – Inserting, Updating & Deleting Data
- Basic App Model – Makemigrations and Migrate
- model data types and fields list
- Add the slug field inside Django Model
- Intermediate fields in Django
- Uploading images in Django
- Render Model in Django Admin Interface
- Change Object Display Name using __str__ function – Django Models
- Built-in Field Validations – Django Models
- Custom Field Validations in Django Models
- How to use Django Field Choices ?
- Overriding the save method – Django Models

Migrations

- The Commands
- Backend Support
- Workflow
- Transactions
- Dependencies
- Migration files
- Adding migrations to apps
- Reversing migrations
- Historical models
- Considerations when removing model fields

- Data Migrations
- Squashing migrations
- Serializing values
- supporting multiple Django versions

Managing files

- Using files in models
- The **File** object
- File storage

Testing in Django

- Writing and running tests
- Testing tools
- Advanced testing topics

User authentication in Django

- Overview
- Installation
- Usage

Django’s cache framework

- Setting up the cache
- The per-site cache
- The per-view cache
- Template fragment caching
- The low-level cache API
- Downstream caches
- Using **Vary** headers
- Controlling cache: Using other headers
- Order of **MIDDLEWARE**

Conditional View Processing

- The **condition** decorator
- Shortcuts for only computing one value
- Using the decorators with other HTTP methods
- Comparison with middleware conditional processing

Cryptographic signing

- Protecting the **SECRET_KEY**
- Using the low-level API

Sending email

- Quick example
- send_mail()
- send_mass_mail()
- mail_admins()
- mail_managers()
- Examples
- Preventing header injection
- The EmailMessage class
- Email backends
- Configuring email for development

Internationalization and localization

- Overview
- Definitions

Logging

- A quick logging primer
- Using logging
- Configuring logging
- Django’s logging extensions
- Django’s default logging configuration

Pagination

- The **Paginator** class
- Example
- Paginating a **ListView**
- Using **Paginator** in a view function

Security in Django

- Cross site scripting (XSS) protection
- Cross site request forgery (CSRF) protection

- SQL injection protection
- Clickjacking protection
- SSL/HTTPS
- Host header validation
- Referrer policy
- Session security
- User-uploaded content
- Additional security topics

Performance and optimization

- Introduction
- General approaches
- Caching
- Understanding laziness
- Databases
- HTTP performance
- Template performance
- Using different versions of available software

Serializing Django objects

- Serializing data
- Deserializing data
- Serialization formats
- Natural keys

Django settings

- The basics
- Designating the settings
- Default settings
- Using settings in Python code
- Altering settings at runtime
- Security
- Available settings
- Creating your own settings
- Using settings without setting **DJANGO_SETTINGS_MODULE**

Signals

● listening to signals

● Defining and sending signals

● Disconnecting signals

System check framework

● writing your own checks

External packages

● Localflavor

● Comments

● Formtools

Asynchronous support

● Async views

● Async safety

● Async adapter functions

“How-to” guides

○ Authentication using **REMOTE_USER**

■ Configuration

■ Using **REMOTE_USER** on login pages only

○ Writing custom **django-admin** commands

■ Accepting optional arguments

■ Management commands and locales

■ Testing

■ Overriding commands

■ Command objects

○ Writing custom model fields

■ Introduction

■ Background theory

■ Writing a field subclass

■ Writing a **FileField** subclass

○ Custom Lookups

■ A lookup example

■ A transformer example

■ Writing an efficient **abs__lt** lookup

■ A bilateral transformer example

■ Writing alternative implementations for existing lookups

■ How Django determines the lookups and transforms which are used

○ Custom template backend

■ Custom backends

■ Debug integration for custom engines

○ Custom template tags and filters

■ Code layout

■ Writing custom template filters

■ Writing custom template tags

○ Writing a custom storage system

○ Deploying Django

■ How to deploy with WSGI

■ How to deploy with ASGI

■ Deployment checklist

○ Upgrading Django to a newer version

■ Required Reading

■ Dependencies

■ Resolving deprecation warnings

■ Installation

■ Testing

■ Deployment

○ Error reporting

■ Email reports

■ Filtering error reports

○ Providing initial data for models

■ Providing initial data with migrations

■ Providing data with fixtures

○ Integrating Django with a legacy database

■ Give Django your database parameters

■ Auto-generate the models

■ Install the core Django tables

■ Test and tweak

○ Outputting CSV with Django

■ Using the Python CSV library

■ Using the template system

■ Other text-based formats

○ Outputting PDFs with Django

■ Install ReportLab

■ Write your view

■ Other formats

○ Overriding templates

■ Overriding from the project’s templates directory

■ Overriding from an app’s template directory

■ Extending an overridden template

○ Managing static files (e.g. images, JavaScript, CSS)

■ Configuring static files

■ Serving static files during development

■ Serving files uploaded by a user during development

■ Testing

■ Deployment

■ Learn more

○ Deploying static files

■ Serving static files in production

○ Writing database migrations

■ Data migrations and multiple databases

■ Migrations that add unique fields

■ Controlling the order of migrations

■ Migrating data between third-party apps

■ Changing a **ManyToManyField** to use a **through** model

■ Changing an unmanaged model to managed