**Module 9) Python DB and Framework**

**Lab Question**

**1. HTML in Python**

* Write a Python program to render an HTML file using Django’s template system.

**Answer:**  
Steps:

**1.** **Create a new Django project:**

django-admin startproject doctorfinder

cd doctorfinder

python manage.py startapp home

**2.In doctorfinder/settings.py, add 'home' to INSTALLED\_APPS.**

**3.Create home/views.py:**

from django.shortcuts import render

def index(request):

return render(request, "index.html")

**4.Create home/templates/index.html:**

<!DOCTYPE html>

<html>

<head>

<title>Doctor Finder</title>

</head>

<body>

<h1>Welcome to Doctor Finder</h1>

</body>

</html>

**5.Add URL in doctorfinder/urls.py:**

from django.contrib import admin

from django.urls import path

from home.views import index

urlpatterns = [

path('admin/', admin.site.urls),

path('', index, name='index'),

]

**Output:**  
When you run python manage.py runserver, the home page shows:  
 *Welcome to Doctor Finder*.

**2. CSS in Python**

**Lab Question:**  
Create a CSS file to style a basic HTML template in Django.

**Answer:**  
Steps:

**1.Inside home/static/css/style.css:**

body {

background-color: #f5f5f5;

font-family: Arial, sans-serif;

text-align: center;

}

h1 {

color: #2c3e50;

}

**2.Modify index.html:**

<!DOCTYPE html>

<html>

<head>

<title>Doctor Profile</title>

{% load static %}

<link rel="stylesheet" href="{% static 'css/style.css' %}">

</head>

<body>

<h1>Doctor Profile</h1>

<p>Dr. John Doe - Cardiologist</p>

</body>

</html>

**Output:**  
Styled doctor profile page using custom CSS.

**3. JavaScript with Python**

**Lab Question:**  
Create a Django project with JavaScript-enabled form validation.

**Answer:**

**1.Create templates/register.html:**

<!DOCTYPE html>

<html>

<head>

<title>Patient Registration</title>

<script>

function validateForm() {

let name = document.forms["regForm"]["name"].value;

let email = document.forms["regForm"]["email"].value;

if (name === "" || email === "") {

alert("All fields must be filled!");

return false;

}

return true;

}

</script>

</head>

<body>

<h2>Patient Registration</h2>

<form name="regForm" onsubmit="return validateForm()">

<input type="text" name="name" placeholder="Name"><br>

<input type="email" name="email" placeholder="Email"><br>

<button type="submit">Register</button>

</form>

</body>

</html>

**Output:**  
Form will not submit if fields are empty.

**4. Django Introduction**

**Lab Question:**  
Write a short project using Django’s built-in tools to render a simple webpage.

**Answer:**

* **After project setup, create a views.py:**

from django.http import HttpResponse

def hello(request):

return HttpResponse("Hello, this is Django Framework!")

Add in urls.py:

path('hello/', hello)

**Output:**  
Visiting /hello/ shows *Hello, this is Django Framework!*.

**5. Virtual Environment**

**Lab Question:**  
Set up a virtual environment for a Django project.

**Answer:**

* **Create virtual environment**

python -m venv venv

# Activate it

venv\Scripts\activate # (Windows)

source venv/bin/activate # (Linux/Mac)

# Install Django

pip install django

**Output:**  
Isolated environment created with Django installed.

**6. Project and App Creation**

**Lab Question:**  
Create a Django project with an app to manage doctor profiles.

**Answer:**

django-admin startproject doctorproject

cd doctorproject

python manage.py startapp doctor

Add 'doctor' in settings.py > INSTALLED\_APPS.

**7. MVT Pattern Architecture**

**Lab Question:**  
Build a simple Django app showcasing how the MVT architecture works.

**Answer:**

**Model (models.py)**:

from django.db import models

class Doctor(models.Model):

name = models.CharField(max\_length=100)

specialty = models.CharField(max\_length=100)

**View (views.py)**:

from django.shortcuts import render

from .models import Doctor

def doctor\_list(request):

doctors = Doctor.objects.all()

return render(request, "doctors.html", {"doctors": doctors})

**Template (doctors.html)**:

<h2>Doctor List</h2>

<ul>

{% for doc in doctors %}

<li>{{ doc.name }} - {{ doc.specialty }}</li>

{% endfor %}

</ul>

**8. Django Admin Panel**

**Lab Question:**  
Set up and customize the Django admin panel.

**Answer:**

In doctor/admin.py:

from django.contrib import admin

from .models import Doctor

class DoctorAdmin(admin.ModelAdmin):

list\_display = ('name', 'specialty')

admin.site.register(Doctor, DoctorAdmin)

**9. URL Patterns and Template Integration**

**Lab Question:**  
Create a Django project with URL patterns, views, and templates.

**Answer:**

In urls.py:

from django.urls import path

from . import views

urlpatterns = [

path('', views.home, name='home'),

path('profile/', views.profile, name='profile'),

path('contact/', views.contact, name='contact'),

]

Templates: home.html, profile.html, contact.html.

**10. Form Validation using JavaScript**

**Lab Question:**  
Implement JavaScript form validation for a user registration form.

**Answer:**

<form onsubmit="return validateForm()">

<input type="email" id="email" placeholder="Email"><br>

<input type="text" id="phone" placeholder="Phone"><br>

<button type="submit">Submit</button>

</form>

<script>

function validateForm() {

let email = document.getElementById("email").value;

let phone = document.getElementById("phone").value;

let emailPattern = /^[^ ]+@[^ ]+\.[a-z]{2,3}$/;

if (!email.match(emailPattern)) {

alert("Invalid Email!");

return false;

}

if (phone.length < 10) {

alert("Phone must be at least 10 digits!");

return false;

}

return true;

}

</script>

**11. Django Database Connectivity (MySQL or SQLite)**

**Lab Question:**  
Set up database connectivity for a Django project.

**Answer:**

1. Open settings.py and configure the database:

**SQLite (default):**

DATABASES = {

'default': {

'ENGINE': 'django.db.backends.sqlite3',

'NAME': BASE\_DIR / "db.sqlite3",

}

}

**MySQL (requires pip install mysqlclient):**

DATABASES = {

'default': {

'ENGINE': 'django.db.backends.mysql',

'NAME': 'doctor\_db',

'USER': 'root',

'PASSWORD': 'password',

'HOST': 'localhost',

'PORT': '3306',

}

}

1. Run:
2. python manage.py makemigrations
3. python manage.py migrate

**12. ORM and QuerySets**

**Lab Question:**  
Perform CRUD operations using Django ORM.

**Answer:**

from doctor.models import Doctor

# Create

doc = Doctor.objects.create(name="Dr. John", specialty="Cardiology")

# Read

all\_docs = Doctor.objects.all()

# Update

doc = Doctor.objects.get(id=1)

doc.specialty = "Neurology"

doc.save()

# Delete

doc.delete()

**13. Django Forms and Authentication**

**Lab Question:**  
Create a Django project for user registration and login functionality.

**Answer:**

1. Add django.contrib.auth to INSTALLED\_APPS.
2. Create forms.py:
3. from django import forms
4. from django.contrib.auth.models import User
5. from django.contrib.auth.forms import UserCreationForm
6. class SignUpForm(UserCreationForm):
7. email = forms.EmailField(required=True)
8. class Meta:
9. model = User
10. fields = ['username', 'email', 'password1', 'password2']
11. Views in views.py:
12. from django.shortcuts import render, redirect
13. from .forms import SignUpForm
14. def signup(request):
15. if request.method == "POST":
16. form = SignUpForm(request.POST)
17. if form.is\_valid():
18. form.save()
19. return redirect("login")
20. else:
21. form = SignUpForm()
22. return render(request, "signup.html", {"form": form})

**14. CRUD Operations using AJAX**

**Lab Question:**  
Implement AJAX in a Django project for performing CRUD operations.

**Answer:**

* Example: Add doctor profile without page reload.

<form id="doctorForm">

<input type="text" id="name" placeholder="Doctor Name">

<input type="text" id="specialty" placeholder="Specialty">

<button type="submit">Save</button>

</form>

<script>

document.getElementById("doctorForm").onsubmit = function(e) {

e.preventDefault();

fetch("/add\_doctor/", {

method: "POST",

headers: {"X-CSRFToken": "{{ csrf\_token }}"},

body: new FormData(this)

}).then(res => res.json())

.then(data => alert("Doctor Added: " + data.name));

};

</script>

**15. Customizing the Django Admin Panel**

**Lab Question:**  
Customize the Django admin panel for better management of records.

**Answer:**

from django.contrib import admin

from .models import Doctor

class DoctorAdmin(admin.ModelAdmin):

list\_display = ('name', 'specialty', 'availability')

search\_fields = ('name', 'specialty')

list\_filter = ('specialty',)

admin.site.register(Doctor, DoctorAdmin)

**16. Payment Integration Using Paytm**

**Lab Question:**  
Implement Paytm payment gateway in a Django project.

**Answer (simplified):**

1. Install Paytm checksum library:
2. pip install paytmchecksum
3. Create a payment view in views.py:
4. from django.shortcuts import render
5. def initiate\_payment(request):
6. return render(request, "paytm\_payment.html")
7. Use Paytm’s sandbox credentials in settings.

**17. GitHub Project Deployment**

**Lab Question:**  
Deploy a Django project to GitHub.

**Answer:**

1. Initialize Git:
2. git init
3. git add .
4. git commit -m "Initial Commit"
5. Create repository on GitHub.
6. Push project:
7. git remote add origin https://github.com/username/doctorfinder.git
8. git branch -M main
9. git push -u origin main

**18. Live Project Deployment (PythonAnywhere)**

**Lab Question:**  
Deploy a Django project to PythonAnywhere.

**Answer:**

1. Create account on pythonanywhere.com.
2. Upload project or clone from GitHub.
3. Configure **Web tab → Add new web app → Manual configuration → Django**.
4. Set WSGI configuration file and static file paths.
5. Reload web app → project goes live.

**19. Social Authentication**

**Lab Question:**  
Implement Google and Facebook login for the Django project.

**Answer:**

1. Install:
2. pip install django-allauth
3. Add to INSTALLED\_APPS:
4. 'django.contrib.sites',
5. 'allauth',
6. 'allauth.account',
7. 'allauth.socialaccount',
8. 'allauth.socialaccount.providers.google',
9. 'allauth.socialaccount.providers.facebook',
10. Configure in settings.py:
11. SITE\_ID = 1
12. LOGIN\_REDIRECT\_URL = '/'
13. Apply migrations and configure Google/Facebook developer credentials.

**20. Google Maps API**

**Lab Question:**  
Use Google Maps API to display doctor locations in the "Doctor Finder" project.

**Answer:**

1. Get an API key from Google Cloud.
2. Add this in templates/map.html:
3. <!DOCTYPE html>
4. <html>
5. <head>
6. <title>Doctor Locations</title>
7. <script src="https://maps.googleapis.com/maps/api/js?key=YOUR\_API\_KEY"></script>
8. <script>
9. function initMap() {
10. var location = {lat: 21.1702, lng: 72.8311}; // Example: Surat
11. var map = new google.maps.Map(document.getElementById('map'), {
12. zoom: 12,
13. center: location
14. });
15. new google.maps.Marker({position: location, map: map});
16. }
17. </script>
18. </head>
19. <body onload="initMap()">
20. <h2>Doctor Locations</h2>
21. <div id="map" style="height:400px;width:100%;"></div>
22. </body>
23. </html>

Top of Form

Bottom of Form