**Short Notes: Django Project Creation and Development**

**1. Installing Django**

* **Why**: To set up the environment for developing the Django project.
* **Command**:

bash

Copy code

pip install django

**2. Starting the Django Project**

* **What**: Create a Django project as a starting point.
* **Command**:

bash

Copy code

django-admin startproject financial\_planner

cd financial\_planner

**3. Running the Development Server**

* **Command**:

bash

Copy code

python manage.py runserver

* **Purpose**: Verify the setup at http://127.0.0.1:8000/.

**4. Setting Up the Database (MySQL)**

* **Purpose**: Configure MySQL as the database backend.
* **Where**: In settings.py:

python

Copy code

DATABASES = {

'default': {

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

'NAME': 'financial\_planner\_db',

'USER': 'root',

'PASSWORD': 'Shyamlal@2004',

'HOST': 'localhost',

'PORT': '3306',

}

}

* **Commands**:

bash

Copy code

python manage.py makemigrations

python manage.py migrate

**5. Creating a Django App**

* **Command**:

bash

Copy code

python manage.py startapp expenses

**6. User Authentication (Register/Login/Logout)**

**Registration**

* **Why**: Allow new users to sign up.
* **View Code**:

python

Copy code

from django.contrib.auth.forms import UserCreationForm

from django.shortcuts import render, redirect

def register(request):

if request.method == 'POST':

form = UserCreationForm(request.POST)

if form.is\_valid():

form.save()

return redirect('login')

else:

form = UserCreationForm()

return render(request, 'registration/register.html', {'form': form})

* **URL Configuration**:

python

Copy code

from django.urls import path

from . import views

urlpatterns = [

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

]

* **Template (register.html)**:

html

Copy code

<h2>Register</h2>

<form method="post">

{% csrf\_token %}

{{ form.as\_p }}

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

</form>

**Login/Logout**

* **Setup**: Use Django's built-in authentication views.
* **URLs**:

python

Copy code

from django.contrib.auth import views as auth\_views

urlpatterns = [

path('login/', auth\_views.LoginView.as\_view(), name='login'),

path('logout/', auth\_views.LogoutView.as\_view(), name='logout'),

]

* **Templates**: Create registration/login.html and registration/logged\_out.html.

**7. Creating the Home Page**

* **What**: A dashboard showing links to major features.
* **Template (home.html)**:

html

Copy code

<h1>Welcome to Financial Planner</h1>

<ul>

<li><a href="{% url 'add\_expense' %}">Add Expense</a></li>

<li><a href="{% url 'view\_expenses' %}">View Expenses</a></li>

<li><a href="{% url 'financial\_report' %}">Financial Report</a></li>

</ul>

**8. Add, Edit, Delete Expenses**

**Model**

* **Code**:

python

Copy code

from django.db import models

from django.contrib.auth.models import User

class Expense(models.Model):

user = models.ForeignKey(User, on\_delete=models.CASCADE)

category = models.CharField(max\_length=50)

amount = models.DecimalField(max\_digits=10, decimal\_places=2)

date = models.DateField()

**Add Expense**

* **View**:

python

Copy code

from django.shortcuts import render, redirect

from .models import Expense

def add\_expense(request):

if request.method == 'POST':

category = request.POST['category']

amount = request.POST['amount']

date = request.POST['date']

Expense.objects.create(user=request.user, category=category, amount=amount, date=date)

return redirect('view\_expenses')

return render(request, 'expenses/add\_expense.html')

* **Template (add\_expense.html)**:

html

Copy code

<h2>Add Expense</h2>

<form method="post">

{% csrf\_token %}

<input type="text" name="category" placeholder="Category">

<input type="number" name="amount" placeholder="Amount">

<input type="date" name="date">

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

</form>

**View/Edit/Delete Expense**

* **View**:

python

Copy code

def view\_expenses(request):

expenses = Expense.objects.filter(user=request.user)

return render(request, 'expenses/view\_expenses.html', {'expenses': expenses})

* **Template (view\_expenses.html)**:

html

Copy code

<h2>Your Expenses</h2>

<ul>

{% for expense in expenses %}

<li>

{{ expense.category }}: ₹{{ expense.amount }}

<a href="{% url 'edit\_expense' expense.id %}">Edit</a>

<a href="{% url 'delete\_expense' expense.id %}">Delete</a>

</li>

{% endfor %}

</ul>

**9. Financial Reports and Graphs**

* **Why**: To summarize financial data and visualize it using a bar chart.

**Financial Report View**

python

Copy code

from django.db.models import Sum

def financial\_reports(request):

expenses = Expense.objects.filter(user=request.user)

total\_expense = expenses.aggregate(Sum('amount'))['amount\_\_sum'] or 0

unique\_dates = expenses.values('date').distinct().count()

avg\_daily\_expense = total\_expense / unique\_dates if unique\_dates > 0 else 0

categories = expenses.values\_list('category', flat=True)

amounts = expenses.values\_list('amount', flat=True)

return render(request, 'expenses/financial\_reports.html', {

'total\_expense': total\_expense,

'avg\_daily\_expense': avg\_daily\_expense,

'categories': list(categories),

'amounts': list(amounts),

})

**Financial Report Template**

* Include a graph using Chart.js:

html

Copy code

<h2>Financial Report</h2>

<p>Total Expense: ₹{{ total\_expense }}</p>

<p>Average Daily Expense: ₹{{ avg\_daily\_expense }}</p>

<canvas id="expenseChart"></canvas>

<script>

const categories = JSON.parse('{{ categories|safe }}');

const amounts = JSON.parse('{{ amounts|safe }}');

const ctx = document.getElementById('expenseChart').getContext('2d');

new Chart(ctx, {

type: 'bar',

data: {

labels: categories,

datasets: [{ label: 'Amount', data: amounts }]

}

});

</script>

**Summary**

| **Feature** | **Description** | **Command/Code** |
| --- | --- | --- |
| **Project Setup** | Initialize Django project and app | startproject, startapp |
| **Authentication** | User registration, login, logout | UserCreationForm, built-in views |
| **Database Setup** | Use MySQL for persistent data storage | Configured in settings.py |
| **CRUD Operations** | Add, view, edit, delete expenses | Views and templates for CRUD |
| **Reports & Graphs** | Summarize and visualize financial data | Chart.js for graph generation |
| **PDF Download** | Export financial reports as PDF | jsPDF and html2canvas integration |

This guide consolidates **all commands, features, and concepts** from your project. Let me know if you need further assistance!