**Enhancing the Django Expense Counter Project**

**1. User Registration and Expense Ownership**

* **Implement Django authentication for user registration and login.**
* **Modify the Expense model to include a ForeignKey to the User model.**
* **Filter expenses by the logged-in user.**

**2. Creating a Separate Category Model**

* **Create a Category model and connect it to Expense via a ForeignKey.**
* **Allow users to add custom categories.**
* **Modify expense creation to use the new category model.**

**3. Implementing Dynamic Filtering**

* **Add filtering options for expenses by date range and category.**
* **Implement filtering using Django ORM and Django Filters.**
* **Create a user-friendly UI to filter expenses dynamically.**

**4. Adding Group Expenses Feature**

* **Create a GroupExpense model to manage shared expenses.**
* **Allow multiple users to be linked to a single expense.**
* **Implement logic to split expenses automatically among users.**
* **Display expense shares for each user.**

**Enhancing the Django Expense Counter Project**

**1. User Registration and Expense Ownership**

**1.1 Install Django Authentication**

Django provides built-in authentication, so add it to your project:

pip install django-allauth

**1.2 Update settings.py**

Enable authentication apps:

INSTALLED\_APPS = [

'django.contrib.auth',

'django.contrib.contenttypes',

'django.contrib.sessions',

'django.contrib.sites',

'allauth',

'allauth.account',

'allauth.socialaccount',

'your\_app',

]

**1.3 Modify Expense Model**

from django.contrib.auth.models import User

from django.db import models

class Expense(models.Model):

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

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

description = models.TextField()

date = models.DateField()

def \_\_str\_\_(self):

return f"{self.user.username} - {self.amount}"

**1.4 Restrict Expenses to Authenticated Users**

Modify your views to filter expenses by user:

from django.contrib.auth.decorators import login\_required

@login\_required

def expense\_list(request):

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

return render(request, 'expense\_list.html', {'expenses': expenses})

**2. Creating a Separate Category Model**

**2.1 Define Category Model**

class Category(models.Model):

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

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

def \_\_str\_\_(self):

return self.name

**2.2 Modify Expense Model to Use Category**

class Expense(models.Model):

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

category = models.ForeignKey(Category, on\_delete=models.SET\_NULL, null=True, blank=True)

**2.3 Allow Users to Add Custom Categories**

Modify views to handle user-created categories:

@login\_required

def add\_category(request):

if request.method == 'POST':

name = request.POST['name']

Category.objects.create(name=name, user=request.user)

return redirect('category\_list')

**3. Implementing Dynamic Filtering**

**3.1 Install django-filter**

pip install django-filter

**3.2 Update Views with Filtering**

import django\_filters

class ExpenseFilter(django\_filters.FilterSet):

date = django\_filters.DateFromToRangeFilter()

category = django\_filters.ModelChoiceFilter(queryset=Category.objects.all())

class Meta:

model = Expense

fields = ['date', 'category']

**3.3 Apply Filtering in Views**

@login\_required

def expense\_list(request):

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

expense\_filter = ExpenseFilter(request.GET, queryset=expenses)

return render(request, 'expense\_list.html', {'filter': expense\_filter})

**4. Adding Group Expenses Feature**

**4.1 Create GroupExpense Model**

class GroupExpense(models.Model):

name = models.CharField(max\_length=255)

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

date = models.DateField()

users = models.ManyToManyField(User)

def split\_expense(self):

return self.amount / self.users.count()

**4.2 View to Add Group Expense**

@login\_required

def add\_group\_expense(request):

if request.method == 'POST':

name = request.POST['name']

amount = request.POST['amount']

users = request.POST.getlist('users')

group\_expense = GroupExpense.objects.create(name=name, amount=amount, date=timezone.now())

group\_expense.users.set(users)

return redirect('group\_expense\_list')

**4.3 Display Expense Shares**

@login\_required

def group\_expense\_list(request):

expenses = GroupExpense.objects.all()

return render(request, 'group\_expense\_list.html', {'expenses': expenses})