Django Lab Exercises with Solutions

Lab 1: Django Project Setup

Exercise:

1. Install Django in a virtual environment.
2. Create a project called `mysite`.
3. Run the development server.

Solution:

```
# Step 1: Create virtual environment
python -m venv venv
source venv/bin/activate  # on Windows: venv\Scripts\activate

# Step 2: Install Django
pip install django

# Step 3: Create project
django-admin startproject mysite

cd mysite

# Step 4: Run server
python manage.py runserver
```

Lab 2: Create a Django App

Exercise:

1. Inside `mysite`, create an app named `blog`.
2. Register the app in `settings.py`.

3. Create a view that returns "Hello, Blog!".

Solution:

python manage.py startapp blog

In `mysite/settings.py`:

```
INSTALLED_APPS = [
    ...,
    'blog',
]
```

In `blog/views.py`:

```
from django.http import HttpResponse

def home(request):
    return HttpResponse("Hello, Blog!")
```

In `mysite/urls.py`:

```
from django.contrib import admin
from django.urls import path
from blog import views

urlpatterns = [
    path('admin/', admin.site.urls),
    path('blog/', views.home),
]
```

Lab 3: Django Templates

Exercise:

1. Create a template `home.html` inside `blog/templates/blog/`.

2. Render the template instead of returning plain text.

Solution:
`blog/views.py`:

```
from django.shortcuts import render

def home(request):
    return render(request, 'blog/home.html')
```

`blog/templates/blog/home.html`:

```
<!DOCTYPE html>
<html>
<head><title>Blog Home</title></head>
<body>
  <h1>Welcome to My Blog</h1>
</body>
</html>
```

Lab 4: Django Models and ORM

Exercise:

1. Create a model `Post` with fields: `title`, `content`, and `date_created`.
2. Apply migrations.
3. Add a few posts using the Django shell.
4. Fetch all posts.

Solution:
`blog/models.py`:

```
from django.db import models

class Post(models.Model):
    title = models.CharField(max_length=100)
    content = models.TextField()
    date_created = models.DateTimeField(auto_now_add=True)

    def __str__(self):
        return self.title
```

Run migrations:

```
python manage.py makemigrations
```

python manage.py migrate

Django shell:

python manage.py shell

```
from blog.models import Post
Post.objects.create(title="First Post", content="Hello Django!")
Post.objects.all()
```

Lab 5: Display Data in Templates

Exercise:

1. Fetch all posts from the database.
2. Display them in `home.html`.

Solution:
`blog/views.py`:

```
from .models import Post

def home(request):
    posts = Post.objects.all()
    return render(request, 'blog/home.html', {'posts': posts})
```

`blog/templates/blog/home.html`:

```
<h1>My Blog</h1>
<ul>
  {% for post in posts %}
    <li><b>{{ post.title }}</b>: {{ post.content }}</li>
  {% endfor %}
</ul>
```

Lab 6: Django Forms

Exercise:

1. Create a form to add a new blog post.

2. Save the post to the database.

Solution:

`blog/forms.py`:

```
from django import forms
from .models import Post

class PostForm(forms.ModelForm):
    class Meta:
        model = Post
        fields = ['title', 'content']
```

`blog/views.py`:

```
def add_post(request):
    if request.method == 'POST':
        form = PostForm(request.POST)
        if form.is_valid():
            form.save()
            return redirect('home')
    else:
        form = PostForm()
    return render(request, 'blog/add_post.html', {'form': form})
```

`blog/templates/blog/add_post.html`:

```
<h2>Add Post</h2>
<form method="POST">
  {% csrf_token %}
  {{ form.as_p }}
  <button type="submit">Save</button>
</form>
```

`mysite/urls.py`:

```
path('add/', views.add_post, name='add_post'),
```

Lab 7: User Authentication

Exercise:

1. Create a user registration page.
2. Allow login/logout.

Solution (simple):

python manage.py createsuperuser

In `mysite/urls.py`:

```
from django.contrib.auth import views as auth_views

urlpatterns += [
    path('login/', auth_views.LoginView.as_view(template_name='blog/login.html'),
name='login'),
    path('logout/', auth_views.LogoutView.as_view(next_page='home'), name='logout'),
]
```

`blog/templates/blog/login.html`:

```
<h2>Login</h2>
<form method="post">
  {% csrf_token %}
  {{ form.as_p }}
  <button type="submit">Login</button>
</form>
```

Lab 8: Admin Customization

Exercise:

1. Register the `Post` model in Django admin.
2. Customize the admin to display `title` and `date_created`.

Solution:

`blog/admin.py`:

```
from django.contrib import admin
from .models import Post
```

```
class PostAdmin(admin.ModelAdmin):
    list_display = ('title', 'date_created')

admin.site.register(Post, PostAdmin)
```