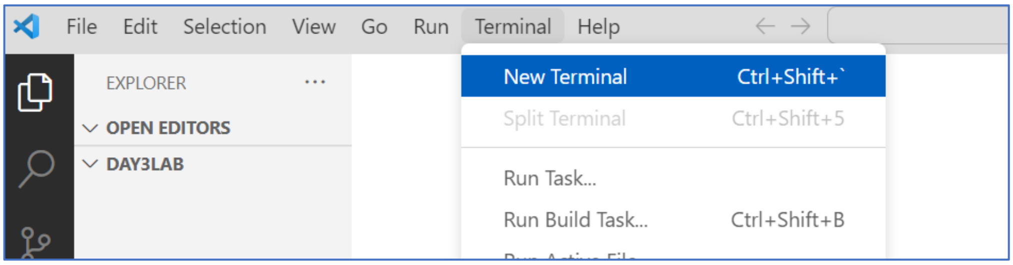
Build a Django CRUD App by Using Class-Based Views

Yes, it is that easy and fast with Django

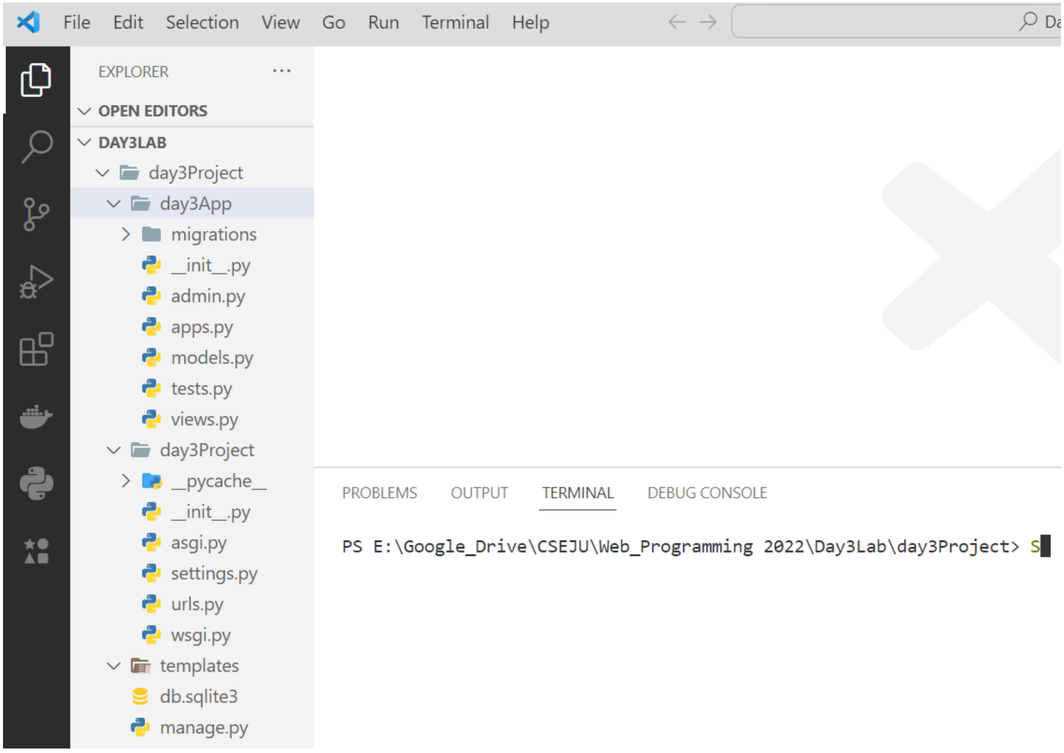
1. Create a folder named "`Day3Lab`"
2. Open the folder from vs code
3. Go to "*Terminal*" and click "`New Terminal`"


File
Edit
Selection
View
Go
Run
Terminal
Help
EXPLORER
OPEN EDITORS
DAY3LAB
New Terminal Ctrl+Shift+`
Split Terminal Ctrl+Shift+5
Run Task...
Run Build Task... Ctrl+Shift+B

4. In the command line, type

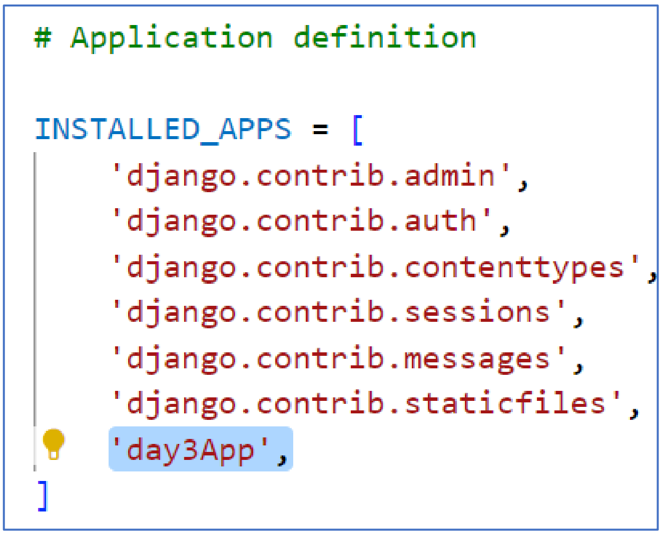
```
pip install django
python -m pip install --upgrade pip
```

5. Create a project by typing "`django-admin startproject day3Project`". This will create Django project with folder name "`day3Project`"
6. Go to the `day3Project` folder by typing `cd .\day3Project\`
7. To check that your project has been successfully created by typing "`python .\manage.py runserver`"
8. Create an app named "`day3App`". Type: "`python .\manage.py day3app`"
9. Create a template folder named "templates" by typing "`mkdir templates`"

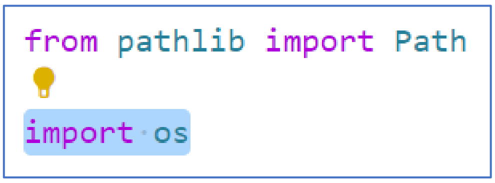


```
PS E:\Google_Drive\CSEJU\Web_Programming 2022\Day3Lab\day3Project> S
```

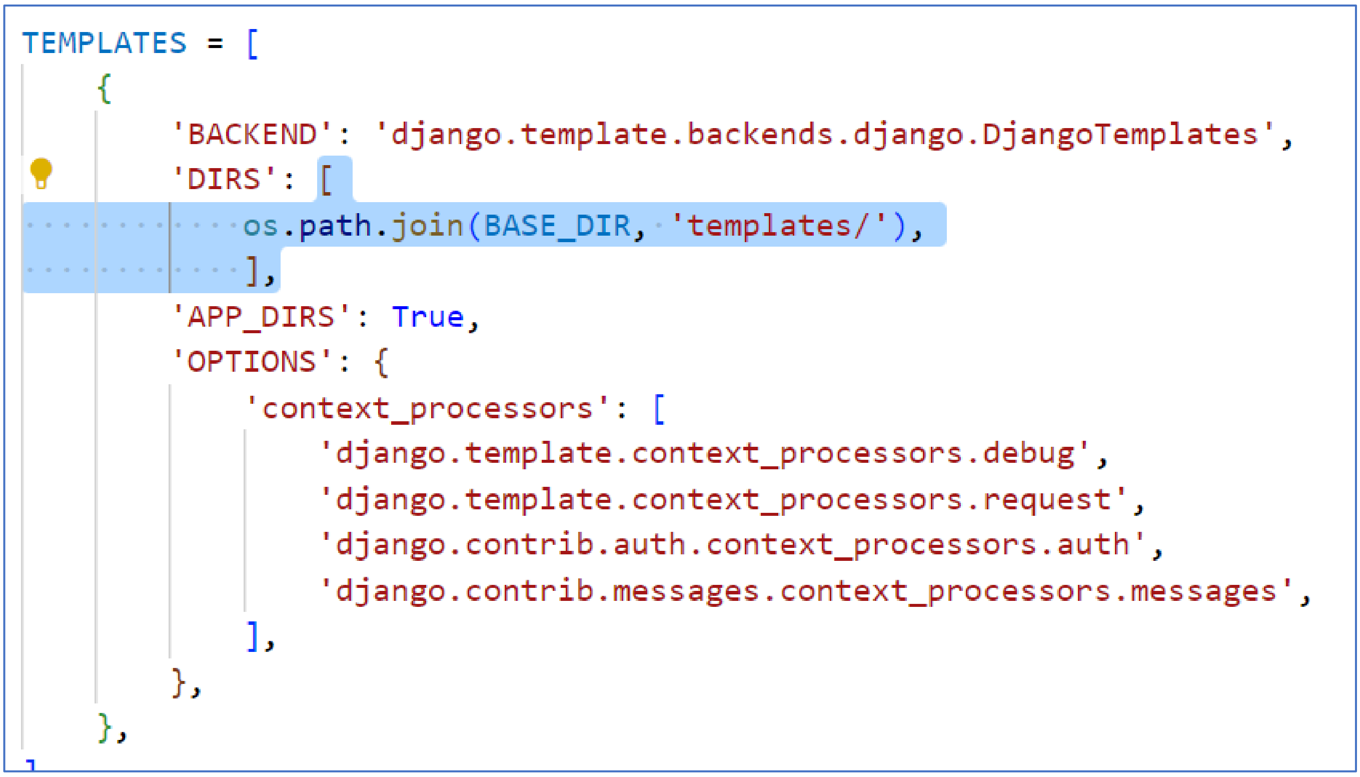
10. Modify `setting.py` file in the following manner



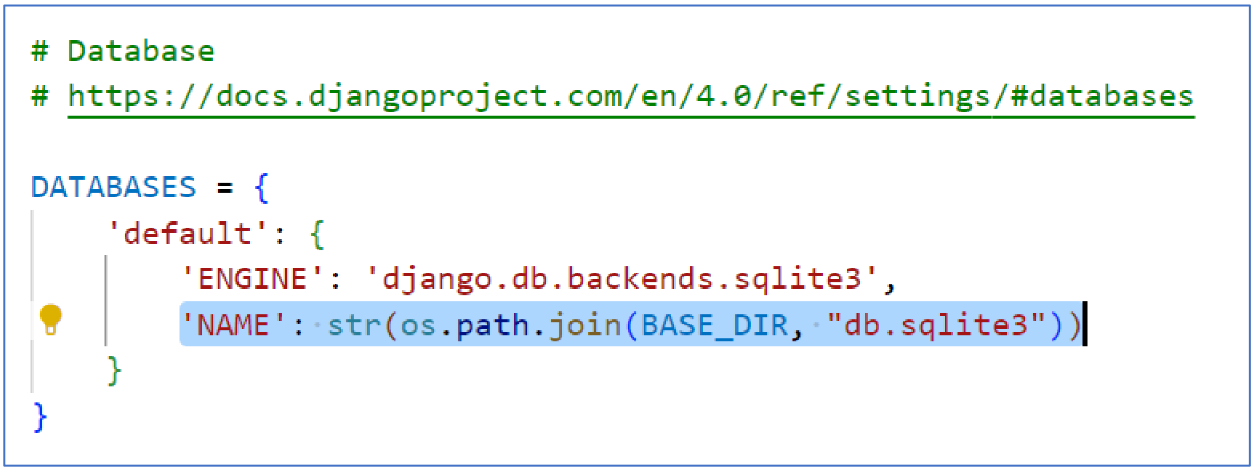
```
# Application definition

INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'day3App',
]
```



```
from pathlib import Path

import os
```

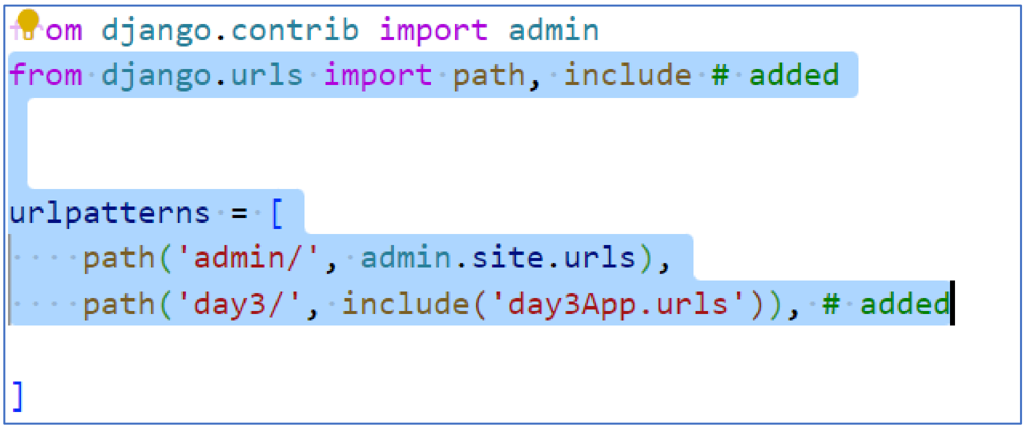


```
TEMPLATES = [
    {
        'BACKEND': 'django.template.backends.django.DjangoTemplates',
        'DIRS': [
            os.path.join(BASE_DIR, 'templates/'),
        ],
        'APP_DIRS': True,
        'OPTIONS': {
            'context_processors': [
                'django.template.context_processors.debug',
                'django.template.context_processors.request',
                'django.contrib.auth.context_processors.auth',
                'django.contrib.messages.context_processors.messages',
            ],
        },
    },
]
```



```
# Database
# https://docs.djangoproject.com/en/4.0/ref/settings/#databases

DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.sqlite3',
        'NAME': str(os.path.join(BASE_DIR, "db.sqlite3"))
    }
}
```

11. Create `urls.py` file in the day3App folder and modify `day3Project/urls.py` file.



```
from django.contrib import admin
from django.urls import path, include # added


urlpatterns = [
    path('admin/', admin.site.urls),
    path('day3/', include('day3App.urls')), # added

]
```

12. In `day3Apps/models.py`, enter the following code:

```
from django.db import models

# Create your models here.
from django.core.validators import MinValueValidator, MaxValueValidator

class Department(models.Model):
    name = models.CharField(max_length=100)

    def __str__(self):
        return self.name

class Employee(models.Model):
    name = models.CharField(max_length=200,)
    designation = models.CharField(max_length=200,)
    email=models.EmailField(blank=True)
    joining_date = models.DateField(auto_now_add=True)
    salary = models.FloatField(blank=True, null=True,validators=[MinValueValidator(20000),
MaxValueValidator(10000000)])
    department = models.ForeignKey(Department, blank=True, null=True,on_delete=models.CASCADE)

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

13. In `day3Apps/views.py`, enter the following code:

```python
from django.shortcuts import render

# Create your views here.
from django.views import View
from django.views.generic.list import ListView
from django.views.generic.detail import DetailView
from django.views.generic.edit import CreateView, UpdateView, DeleteView
from django.urls import reverse_lazy

from .models import Employee

class EmployeeBaseView(View):
    model = Employee
    fields = '__all__'
    success_url = reverse_lazy('day3App:all')

class EmployeeListView(EmployeeBaseView, ListView):
    """_summary_
    """
class EmployeeDetailView(EmployeeBaseView, DetailView):
    """_summary_
    """

class EmployeeCreateView(EmployeeBaseView, CreateView):
    """_summary_
    """
class EmployeeUpdateView(EmployeeBaseView, UpdateView):
    """View to update a Day3"""

class EmployeeDeleteView(EmployeeBaseView, DeleteView):
    """View to delete a Day3"""
```

14. In `day3Apps/urls.py`, enter the following code:

```python
from django.urls import path

from . import views

app_name = 'day3App'

urlpatterns = [
    path('', views.EmployeeListView.as_view(), name='all'),
    path('<int:pk>/detail', views.EmployeeDetailView.as_view(), name='employee_detail'),
    path('create/', views.EmployeeCreateView.as_view(), name='employee_create'),
    path('<int:pk>/update/', views.EmployeeUpdateView.as_view(), name='employee_update'),
    path('<int:pk>/delete/', views.EmployeeDeleteView.as_view(), name='employee_delete'),
]
```

15. Adding the Templates

After defining the CRUD views, you next need to add the template for each of your views. Each view expects a template with a specific name in the templates folder of your application.

a) Inside the templates folder, create a templates/day3App/ folder and start by adding the employee_list.html file with the following content:

```
<h1>Employee List</h1>

<hr>

<table>
    <thead>
        <tr>

                <th>Name</th>
                <th>Designation</th>
                <th>Email</th>
                <th>Joining Date</th>
                <th>Salary</th>
                <th>Department</th>

                <th></th>
                <th></th>
            </tr>
        </thead>
        <tbody>

        {% for employee in employee_list %}
        <tr>

          <td><a href="{% url 'day3App:employee_detail' employee.id %}">{{employee.name}}</a></td>
          <td>{{employee.designation}}</td>
          <td>{{employee.email}}</td>
          <td>{{employee.joining_date}}</td>
          <td>{{employee.salary | floatformat:2}}</td>
          <td>{{employee.department}}</td>

          <td><a href="{% url 'day3App:employee_update' employee.id %} ">Update</a></td>
          <td><a href="{% url 'day3App:employee_delete' employee.id %}">Delete</a></td>
       </tr>
        {% endfor %}

        </tbody>
</table>
<hr>
<a href="{% url 'day3App:employee_create' %}" >Add New Employee</a>
```

b) Add the employee_form.html file with the following content:

```
<form action="" method="post">
    {% csrf_token %}
    <table class="table">
    {{form.as_table}}

    <input type="submit" value="Submit">
    <input type="submit"" onclick="window.location='{% url 'day3App:all' %}'; return 
false;" value="Cancel">

</form>
```

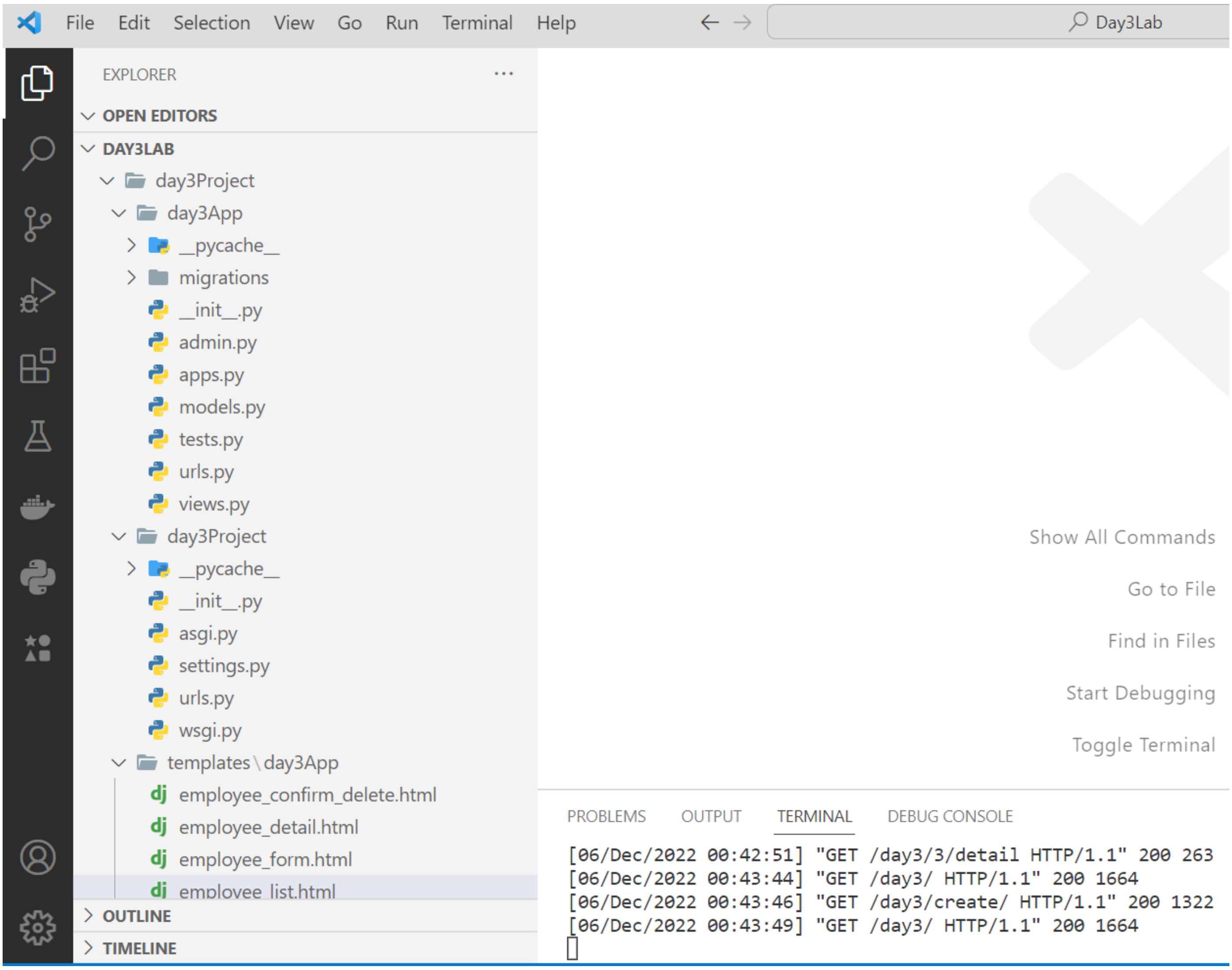
c) Add the employee_detail.html file with the following content:

```
<h1>Employee Details</h1>
<hr>
<p>Name: {{employee.designation}}</p>
<p>Designation: {{employee.designation}}</p>
<p>Email Address: {{employee.email}}</p>
<p>Joining Date: {{employee.joining_date}}</p>
<p>Salary: {{employee.salary | floatformat:2}}</p>
<p>Department: {{employee.department}}</p>
<hr>


<a href="{% url 'day3App:all' %}" >BACK</a>
```

d) Add the employee_confirm_delete.html file with the following content:

```
<h1 class="mb-5">Delete Employee</h1>
<hr>

<p class="h5">Are you sure you want to delete this record:</p>

<h3>{{ employee.name }}</h3>

<form action="" method="POST">
    {% csrf_token %}

    <input type="submit" value="Confirm">
    <input type="submit" onclick="window.location='{{list_all}}'; return false;" value="Cancel">
</form>
```

At a glance



```
[06/Dec/2022 00:42:51] "GET /day3/3/detail HTTP/1.1" 200 263
[06/Dec/2022 00:43:44] "GET /day3/ HTTP/1.1" 200 1664
[06/Dec/2022 00:43:46] "GET /day3/create/ HTTP/1.1" 200 1322
[06/Dec/2022 00:43:49] "GET /day3/ HTTP/1.1" 200 1664
```

ENJOY YOUR TIME