Django

```
py -m venv myvenv

myvenv\Scripts\activate.bat

pip install django

django-admin startproject (projectname)

python manage.py startapp (appname)

pip install psycopg2

# Database setting

DATABASES = {
    "default": {
        "ENGINE": "django.db.backends.postgresql",
        "NAME": "blogs",
        "USER": "db_username",
        "PASSWORD": "password",
        "HOST": "localhost",
        "PORT": "5432",
    }
}
```

```
INSTALLED_APPS = [
    "django.contrib.admin",
    "django.contrib.auth",
    "django.contrib.contenttypes",
    "django.contrib.sessions",
    "django.contrib.messages",
    "django.contrib.staticfiles",
    # Add your apps here
    "blogs",
]
```

notebook

pip install django psycopg2-binary

pip install django-extensions ipython jupyter notebook

pip install ipython==8.25.0 jupyter_server==2.14.1 jupyterlab==4.2.2 jupyterlab_server==2.27.2

pip install notebook==6.5.7

mkdir notebooks

python manage.py shell_plus --notebook

```
INSTALLED_APPS = [
    "django.contrib.admin",
    "django.contrib.auth",
    "django.contrib.contenttypes",
    "django.contrib.sessions",
    "django.contrib.messages",
    "django.contrib.staticfiles",

    "django_extensions",
    "blogs",
]
```

ใส่อันนี้บน note bookบนสุด

```
import os
os.environ["DJANGO_ALLOW_ASYNC_UNSAFE"] = "true"
```

VIEW

```
from django.http import HttpResponse
from django.views import View


class base view
class MyView(View):
    def get(self, request):
        # <view logic>
        return HttpResponse("result")
```

use this in setting

```
import os
SETTINGS_PATH =
os.path.dirname(os.path.dirname(__file__))
```

```
TEMPLATE_DIRS = (
    os.path.join(SETTINGS_PATH, 'templates'),
)
```

VIEW

import ทั้งหมดนี้

```
from django.shortcuts import render
from django.views import View
from .models import *
from django.db.models import *
from django.http import JsonResponse
```

URLLL

import ทั้งหมด

```
from django.contrib import admin
from django.urls import path
from _____ import views
from _____.views import *
```

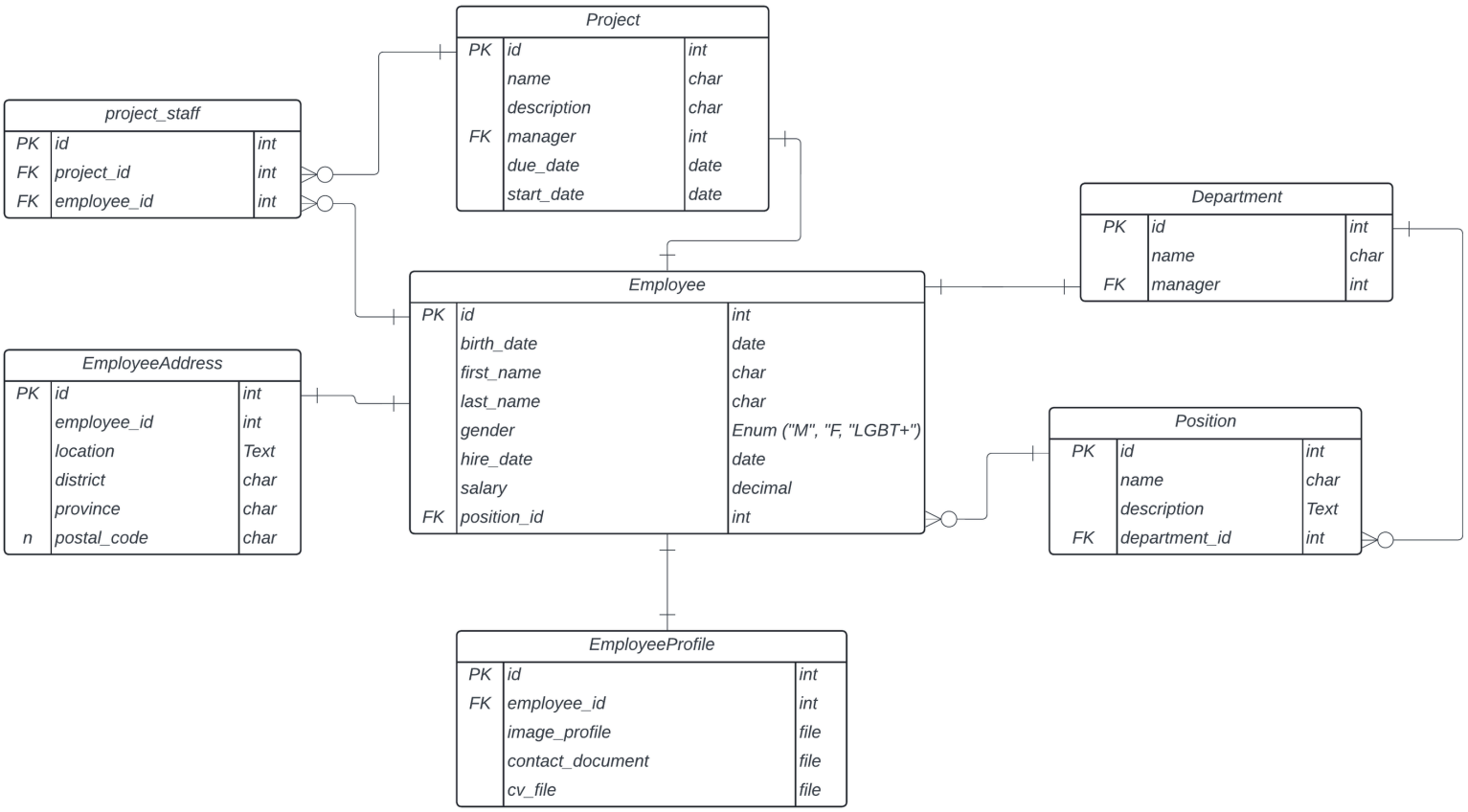
ยกตัวอย่าง ur’

```
urlpatterns = [
```

```
    path('admin/', admin.site.urls),
    path("employee/", MyEmployee.as_view()),
    path("position/", MyPosition.as_view()),
    path("project/", MyProject.as_view()),
    path("project/delete/<int:pro_id>/", MyProject.as_view()),####เอาไว้deleteในหน้าเดียวกับ project
    path("project/project_detail/<int:pro_id>/", MyProjectDetail.as_view()),

path("project/project_detail/<int:pro_id>/<int:emp_id>/", MyProjectDetail.as_view()),
]
```

exxxxxx



ตัวอย่าง ex7แบบ กุเข้าใจ

1. แสดงผลข้อมูลใน Template

> สำหรับแบบฝึกหัดนี้ให้สร้าง View แบบ "class-based view"

1.1 ให้นักศึกษาสร้าง View และกำหนด URL ให้แสดงข้อมูลของพนักงานทั้งหมดในฐานข้อมูลในไฟล์ employee.html ตามภาพ (0.25 คะแนน)

URL

```
 path("employee/", MyEmployee.as_view()),
```

View

```
class MyEmployee(View):#มีViewด้วย
    def get(self,request):
```

```
    empall = Employee.objects.all().order_by('id')
    empcount = empall.count()
    popo = {'emp':empall,'count':empcount}
    #return render (request ต้องมี,"ชื่อไฟลhtmlที่จะใช้",ตัวแปรที่จะส่งข้อมูลไปเป็นdict)
    return render (request,"employee.html",popo)
```

อธืบาย

class ... คือสร้าง class base view
ซึ่งdef คือ method ที่ได้มา ต้องมีself request ตลอด

emp all คือ คิวรีข้อมูลทุกตัวของ employee เรียงตาม id

emp count คือนับ จำนวนตัวในempall

popoคือ dict [’ชื่อค้า’:ตัวที่เราหมายถึง]

```
 return render (request,"employee.html",popo)
```

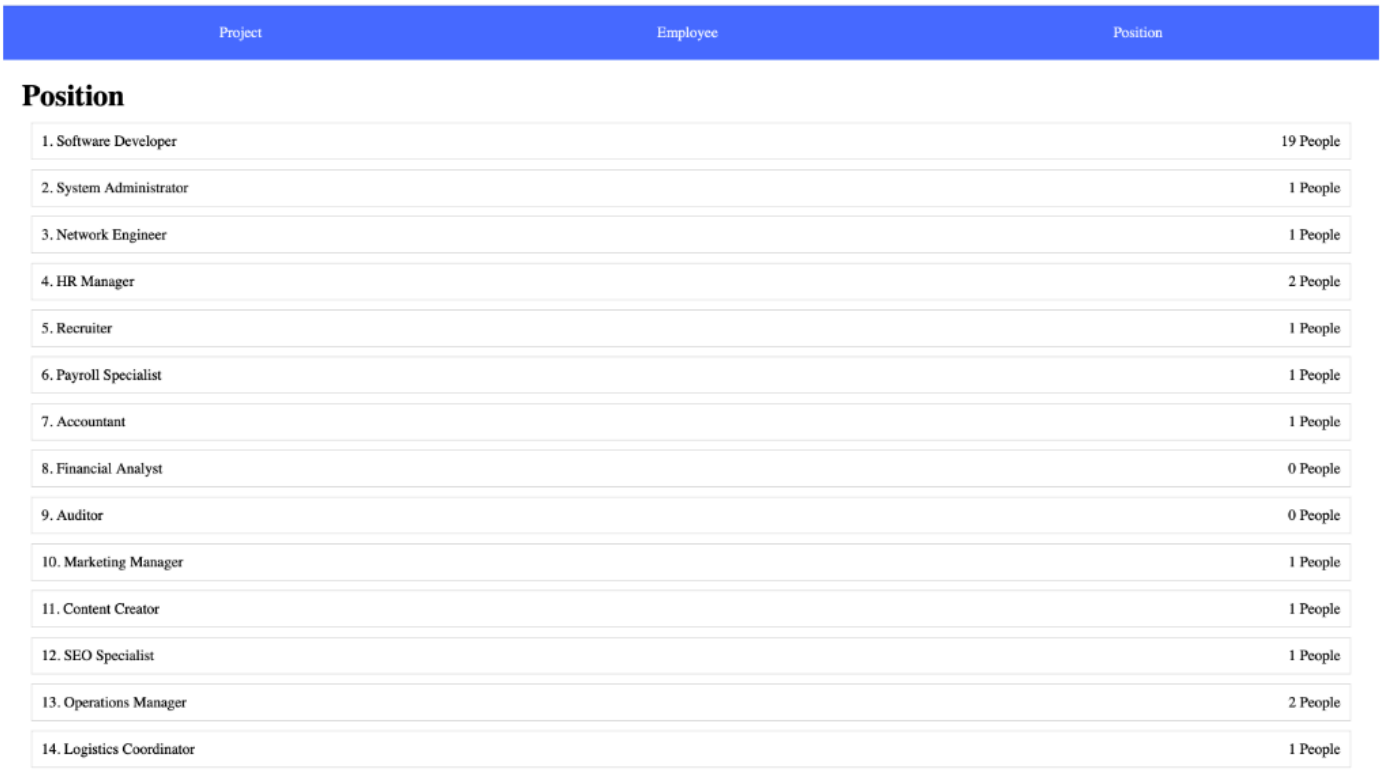
ต้องreturnตลอด request ไฟล ตัวแปรที่อยากให้ไปถึงหน้านั้น

```
 <tbody>
            {% for i in emp %}
            <tr>
               <!-- employee id  ปริ้นตัวแปรใช้  ใช้ สองอัน-->
               <td>{{i.id}}</td>
```

```
    <!-- ชื่อ นามสกุล -->
    <td>{{i.first_name}} {{i.last_name}}</td>
    <!-- เพศ -->
    <td>{{i.gender}}</td>
    <!-- วันเกิด -->
    <td>{{i.birth_date}}</td>
    <!-- วันเริ่มงาน -->
    <td>{{i.hire_date}}</td>
    <!-- เงินเดือน -->
    <td>{{i.salary}}</td>
    <!-- แผนก -->
    <td>{{i.position.department.name}}</td>
    <!-- ตำแหน่งงาน -->
    <td>{{i.position.name}}</td>

</tr>
{% endfor %}
```

1.2 ให้นักศึกษาสร้าง View และกำหนด URL ให้แสดงข้อมูลของตำแหน่งงาน และแสดงจำนวนของพนักงานในไฟล์ position.html ตามภาพ (0.25 คะแนน)



URL

```
path("position/", MyPosition.as_view()),
```

VIEW

```
class MyPosition(View):#มีViewด้วย
    def get(self,request):
        posall = Position.objects.annotate(countpos=Count('employee')).order_by('id')
        pos = {'pos':posall}
```

```
    #return render (request ต้องมี,"ชื่อไฟลhtmlที่จะใช้",ตัวแปรที่จะส่งข้อมูลไปเป็นdict)
    return render (request,"position.html",pos)

template
 <main>
    <div class="head">
      <h1>Position</h1>
    </div>

    <!-- กำหนด if และ for ให้ถูกต้อง -->
    <div class="itemGroup">
      {%for i in pos%}
      <div class="item">
        <!-- position id และ ชื่อตำแหน่ง -->
        <div>{{i.id}}. {{i.name}}</div>

        <!-- จำนวนพนักงานทั้งหมด -->
        <div class="action">{{i.countpos}} People</div>
      </div>
```

{%endfor%}

</main>

1.3ก้พอๆกัน

1.4ก้งาสยๆ

Project

Employee

Position

ใช้แบบนี้นะ1.4

ทำ ตัว แปรใน view กับ url ต้องเหมือนกัน แต่ใน js เราต้องหาตัวแปรอะไรก้ได้มาทำให้สมบูน

ex8

ใส่ใน setting

STATIC_URL = "static/"

STATICFILES_DIRS = [

    BASE_DIR / "static",

]

เพิ่ม static ใน installed app

'django.contrib.staticfiles',

ทุกครั้งที่ใช้ ไฟล static อย่างลืม{% load static %}

{% extends "layout.html"%}คือเอาหน้า baseมาทั้งหน้า แล้วถ้าอยากเปลี่ยนตรงblockก้แค่สร้างblockแล้วเปลี่ยนข้างใน ถ้าไม่เปลี่ยนไรไม่ต้องเขียนblockนั่น

HUMANICE

1. pip install humanize

'django.contrib.humanize' ใน setting

django.contrib.humanize¶

A set of Django template filters useful for adding a “human touch” to data.

To activate these filters, add 'django.contrib.humanize' to your INSTALLED_APPS setting. Once you’ve done that, use {% load humanize %} in a template, and you’ll have access to the following filters.

apnumber¶

For numbers 1-9, returns the number spelled out. Otherwise, returns the number. This follows Associated Press style.

Examples:

- 1 becomes one.
- 2 becomes two.
- 10 becomes 10.

You can pass in either an integer or a string representation of an integer.

intcomma¶

Converts an integer or float (or a string representation of either) to a string containing commas every three digits.

Examples:

- 4500 becomes 4,500.
- 4500.2 becomes 4,500.2.
- 45000 becomes 45,000.
- 450000 becomes 450,000.
- 4500000 becomes 4,500,000.

Format localization will be respected if enabled, e.g. with the 'de' language:

- 45000 becomes '45.000'.
- 450000 becomes '450.000'.

intword¶

Converts a large integer (or a string representation of an integer) to a friendly text representation. Translates 1.0 as a singular phrase and all other numeric values as plural, this may be incorrect for some languages. Works best for numbers over 1 million.

Examples:

- 1000000 becomes 1.0 million.
- 1200000 becomes 1.2 million.
- 1200000000 becomes 1.2 billion.
- -1200000000 becomes -1.2 billion.

Values up to 10^100 (Googol) are supported.

Format localization will be respected if enabled, e.g. with the 'de' language:

- 1000000 becomes '1,0 Million'.
- 1200000 becomes '1,2 Millionen'.
- 1200000000 becomes '1,2 Milliarden'.
- -1200000000 becomes '-1,2 Milliarden'.

naturalday¶

For dates that are the current day or within one day, return “today”, “tomorrow” or “yesterday”, as appropriate. Otherwise, format the date using the passed in format string.

Argument: Date formatting string as described in the date tag.

Examples (when ‘today’ is 17 Feb 2007):

- 16 Feb 2007 becomes yesterday.
- 17 Feb 2007 becomes today.
- 18 Feb 2007 becomes tomorrow.
- Any other day is formatted according to given argument or the DATE_FORMAT setting if no argument is given.

naturaltime¶

For datetime values, returns a string representing how many seconds, minutes or hours ago it was – falling back to the timesince format if the value is more than a day old. In case the datetime value is in the future the return value will automatically use an appropriate phrase.

Examples (when ‘now’ is 17 Feb 2007 16:30:00):

- 17 Feb 2007 16:30:00 becomes now.
- 17 Feb 2007 16:29:31 becomes 29 seconds ago.
- 17 Feb 2007 16:29:00 becomes a minute ago.
- 17 Feb 2007 16:25:35 becomes 4 minutes ago.
- 17 Feb 2007 15:30:29 becomes 59 minutes ago.
- 17 Feb 2007 15:30:01 becomes 59 minutes ago.
- 17 Feb 2007 15:30:00 becomes an hour ago.
- 17 Feb 2007 13:31:29 becomes 2 hours ago.
- 16 Feb 2007 13:31:29 becomes 1 day, 2 hours ago.
- 16 Feb 2007 13:30:01 becomes 1 day, 2 hours ago.
- 16 Feb 2007 13:30:00 becomes 1 day, 3 hours ago.
- 17 Feb 2007 16:30:30 becomes 30 seconds from now.
- 17 Feb 2007 16:30:29 becomes 29 seconds from now.
- 17 Feb 2007 16:31:00 becomes a minute from now.
- 17 Feb 2007 16:34:35 becomes 4 minutes from now.
- 17 Feb 2007 17:30:29 becomes an hour from now.
- 17 Feb 2007 18:31:29 becomes 2 hours from now.
- 18 Feb 2007 16:31:29 becomes 1 day from now.
- 26 Feb 2007 18:31:29 becomes 1 week, 2 days from now.

ordinal¶

Converts an integer to its ordinal as a string.

Examples:

- 1 becomes 1st.
- 2 becomes 2nd.
- 3 becomes 3rd.

You can pass in either an integer or a string representation of an integer. Negative integers are returned unchanged.