

Office Employees Management System

Prerequisite:

- Python
- Django
- Basic SQL
- Basic HTML
- Basic CSS
- Basic Bootstrap

Steps:

- Create virtual environment , in that environment install Django.
- First install Django 3.2.11 version.

pip install django==3.2.11

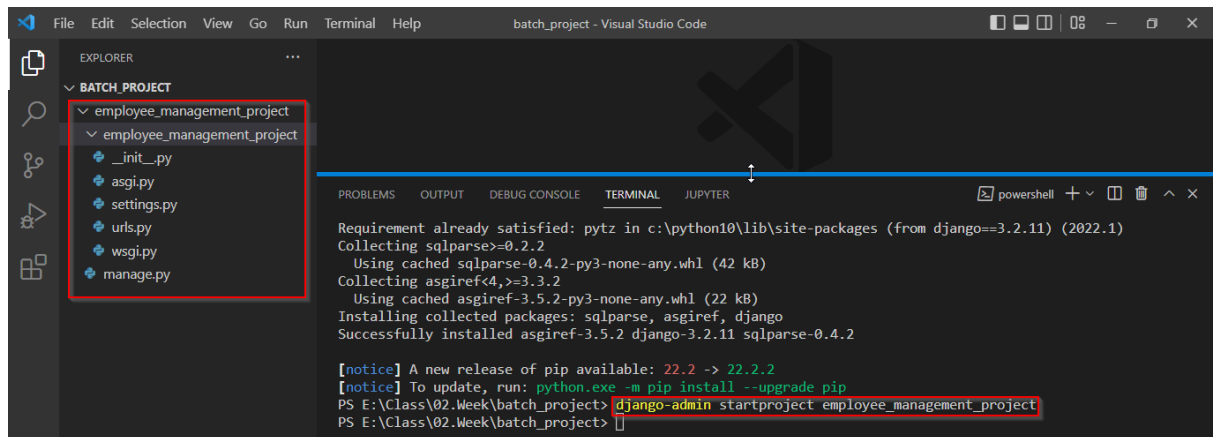
What Is The Difference Between A Project And An App In Django?

In Simple Words:

- ❑ **A Project** is the entire Django application and an **App** is a module inside the project that deals with one specific use case.
For Example: - payment system(app) in the eCommerce app(Project).
 - ❑ **An App** is basically a web Application that is created to perform a specific task.
 - ❑ **A project**, on the other hand, is a collection of these apps.
 - ❑ Therefore, a single project can consist of 'n' number of apps and a single app can be in multiple projects.
- Command To Create A Project:

django-admin startproject project_name

After running this command following folder structure automatically get created.



```
File Edit Selection View Go Run Terminal Help batch_project - Visual Studio Code

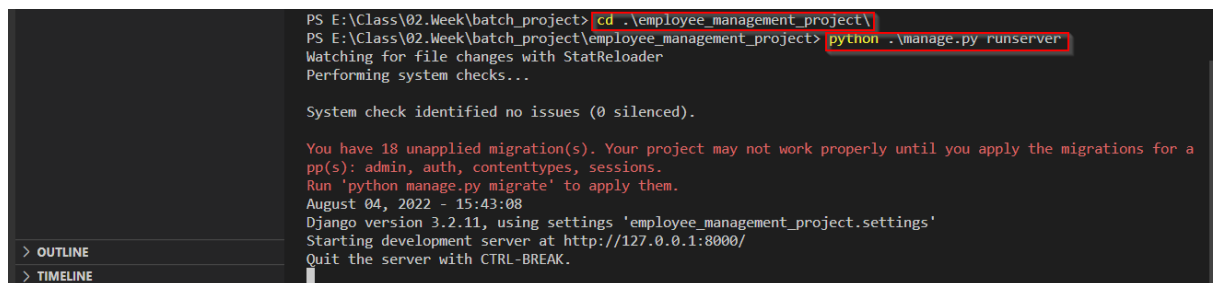
EXPLORER
  BATCH_PROJECT
    employee_management_project
      employee_management_project
        __init__.py
        asgi.py
        settings.py
        urls.py
        wsgi.py
        manage.py

TERMINAL
  Requirement already satisfied: pytz in c:\python10\lib\site-packages (from django==3.2.11) (2022.1)
  Collecting sqlparse==0.2.2
  Using cached sqlparse-0.4.2-py3-none-any.whl (42 kB)
  Collecting asgiref<4,>=3.3.2
  Using cached asgiref-3.5.2-py3-none-any.whl (22 kB)
  Installing collected packages: sqlparse, asgiref, django
  Successfully installed asgiref-3.5.2 django-3.2.11 sqlparse-0.4.2

  [notice] A new release of pip available: 22.2 -> 22.2.2
  [notice] To update, run: python.exe -m pip install --upgrade pip
  PS E:\Class\02.Week\batch_project> django-admin startproject employee_management_project
  PS E:\Class\02.Week\batch_project>
```

Next go in that project folder and run the server using following command, this is the command to run the project.

python .\manage.py runserver

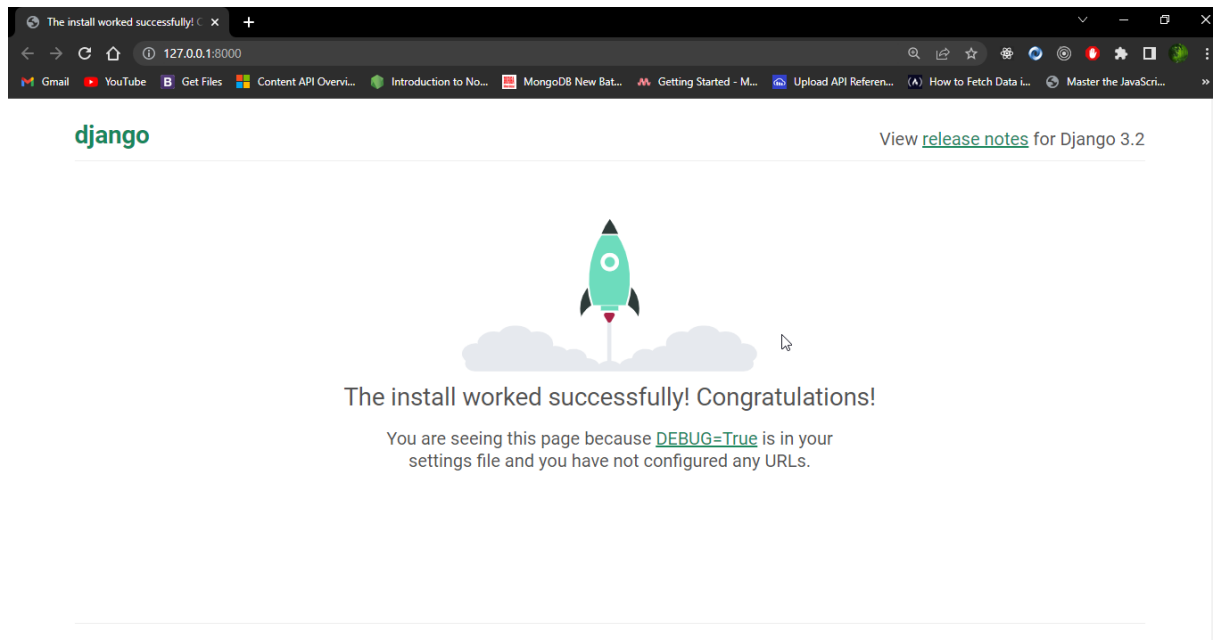


```
PS E:\Class\02.Week\batch_project> cd .\employee_management_project\
PS E:\Class\02.Week\batch_project\employee_management_project> python .\manage.py runserver
Watching for file changes with StatReloader
Performing system checks...

System check identified no issues (0 silenced).

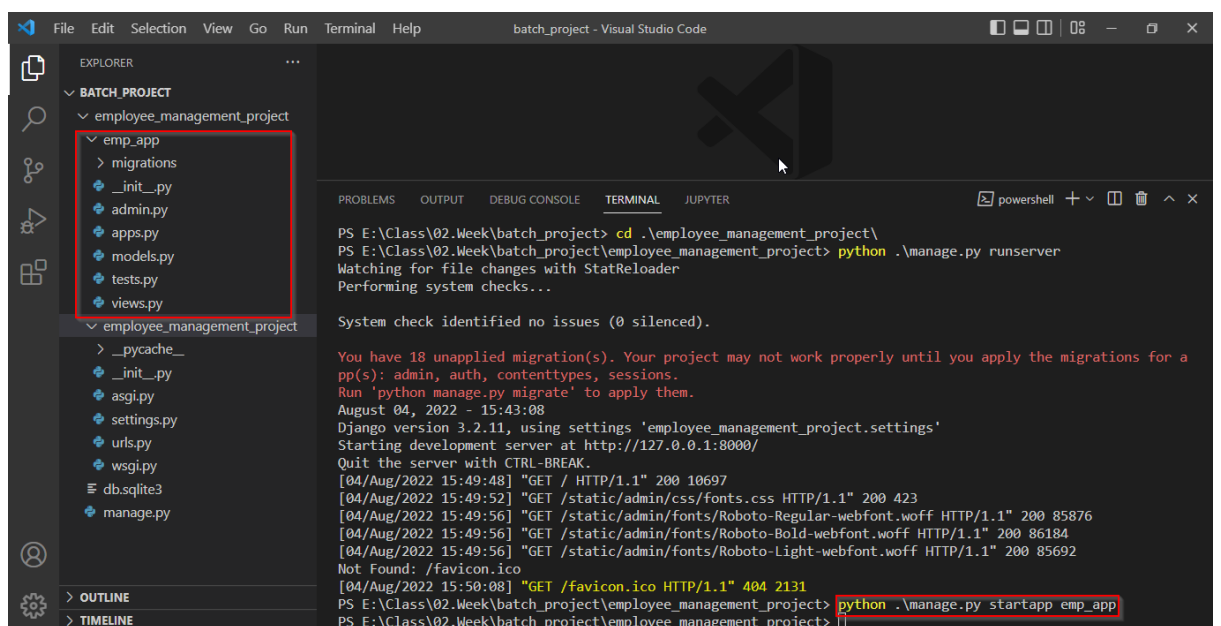
You have 18 unapplied migration(s). Your project may not work properly until you apply the migrations for a
pp(s): admin, auth, contenttypes, sessions.
Run 'python manage.py migrate' to apply them.
August 04, 2022 - 15:43:08
Django version 3.2.11, using settings 'employee_management_project.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.
```

Project server will start running on, localhost port number 8000, this is default port assigned. Your project server starts running you will see the following in localhost.

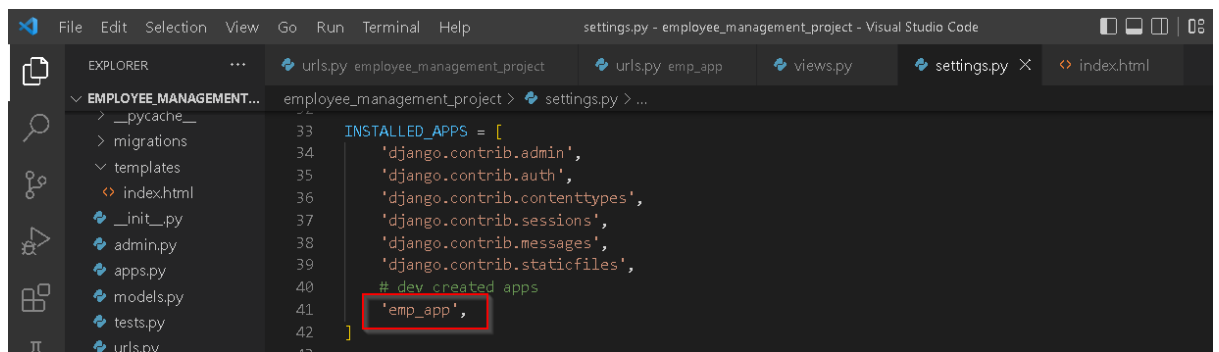


- Command To Create An App:

python manage.py startapp app_name



Add created app in INSTALLED_APPS list in settings.py file



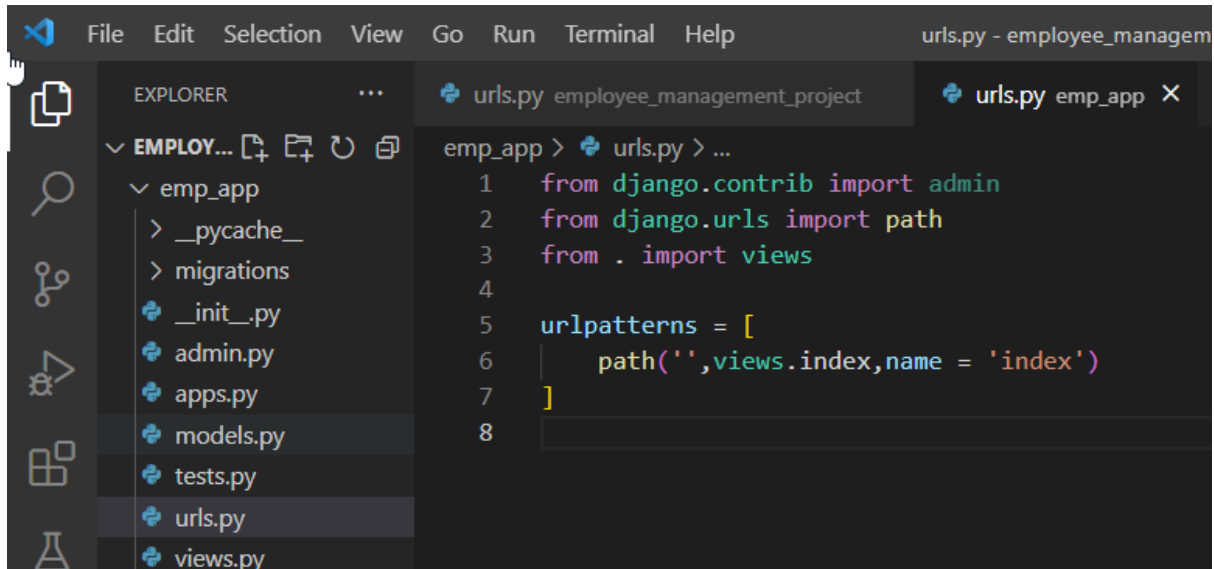
```
33 INSTALLED_APPS = [  
34     'django.contrib.admin',  
35     'django.contrib.auth',  
36     'django.contrib.contenttypes',  
37     'django.contrib.sessions',  
38     'django.contrib.messages',  
39     'django.contrib.staticfiles',  
40     # dev created apps  
41     'emp_app',  
42 ]
```

Which Is The Default Database In Settings File In Django?

Database Name: **SQLite**

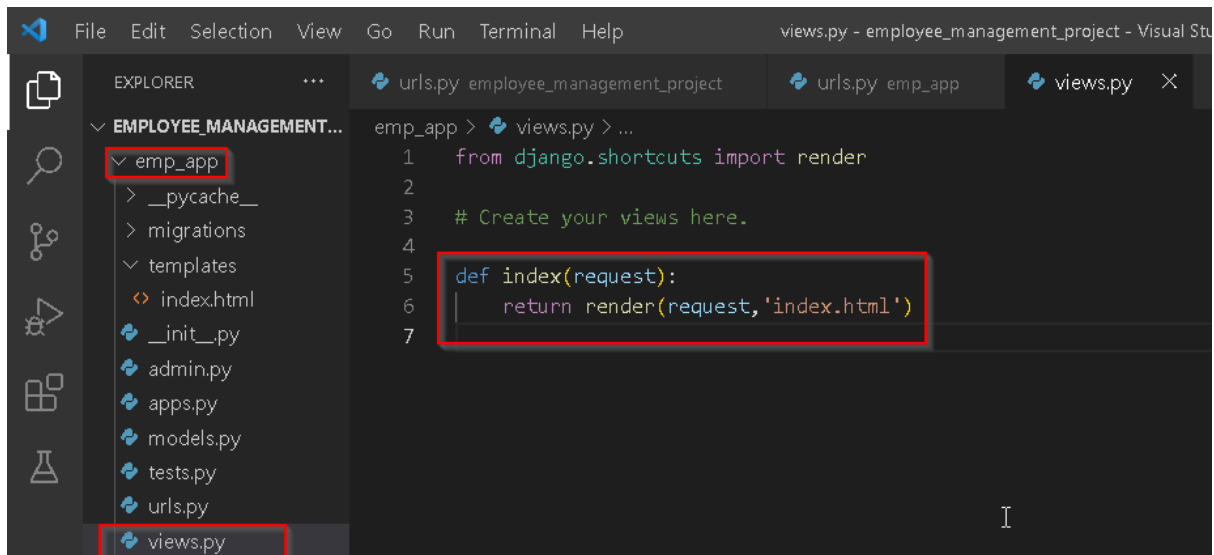
Add paths in urls.py file , that you want to route your app whenever request is coming from client.

Create app level urls.py file, add path to index function in views.py



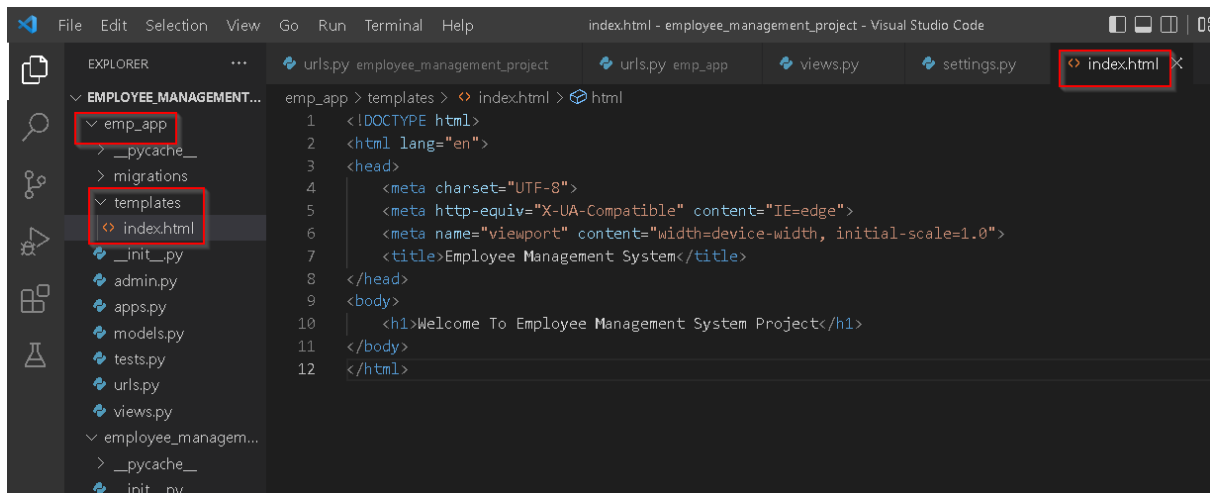
```
File Edit Selection View Go Run Terminal Help urls.py - employee_managem...  
EXPLORER  
EMPLOY...  
  emp_app  
    > __pycache__  
    > migrations  
    urls.py  
    views.py  
    __init__.py  
    admin.py  
    apps.py  
    models.py  
    tests.py  
urls.py emp_app X  
emp_app > urls.py > ...  
1 from django.contrib import admin  
2 from django.urls import path  
3 from . import views  
4  
5 urlpatterns = [  
6     path('', views.index, name = 'index')  
7 ]  
8
```

We have to create this index function in views.py file, in this function we have rendered index.html file.



```
File Edit Selection View Go Run Terminal Help views.py - employee_management_project - Visual Stu...  
EXPLORER  
EMPLOYEE_MANAGEMENT...  
  emp_app  
    > __pycache__  
    > migrations  
    templates  
      index.html  
    __init__.py  
    admin.py  
    apps.py  
    models.py  
    tests.py  
    urls.py  
    views.py  
urls.py emp_app X  
views.py X  
emp_app > views.py > ...  
1 from django.shortcuts import render  
2  
3 # Create your views here.  
4  
5 def index(request):  
6     return render(request, 'index.html')  
7
```

In index function we have to render a template, for that we are going to create templates folder in emp_app, in that we will create index.html file.



After all, above changes, run your project server, you will see following on localhost.

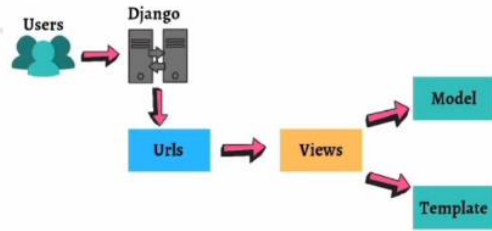


Welcome To Employee Management System Project

Whenever we run the project, first control goes to manage.py file, the request processing is shown below.

Explain How A Request Is Processed In Django?

- ❑ Here, a user requests for a resource to the Django, Django works as a controller and check to the available resource in URL.
- ❑ When Django server is started, the manage.py file searches for settings.py file, which contains information of all the applications installed in the project, middleware used, database connections and path to the main urls config.
- ❑ `Manage.py >> Setting.py >> urls.py >> views.py >> models.py >> templates`
- ❑ Django first determines which root URLconf or URL configuration module is to be used
- ❑ Then, that particular Python module urls is loaded and then Django looks for the variable urlpatterns
- ❑ Then check each URL patterns in urls.py file, and it stops at the first match of the requested URL
- ❑ Once that is done, the Django then imports and calls the given view.
- ❑ In case none of the URLs match the requested URL, Django invokes an error-handling view
- ❑ If URL maps, a view is called that interact with model and template, it renders a template.
- ❑ Django responds back to the user and sends a template as a response.



When the function runs from views.py file whatever in html file it will get rendered on client side i.e frontend.

Up to this, is the basic understanding of Django project. This is basic flow of Django project; this is how request is processed in Django. Next, we will start actual project.

-Create Models

- Here we specify what things you need in database.
- Defining models means you are telling which tables you want in database.
- We are defining Employee model, in Employee model we can multiple departments, so creating separate table for department and connect that with foreign key with Employee table.

What Is The Command For Migrations In Django?

Command to create a migration file inside the migration folder:

```
python manage.py makemigrations
```

After creating a migration, to reflect changes in the database permanently execute migrate command:

```
python manage.py migrate
```

To see raw SQL query executing behind applied migration execute the command:

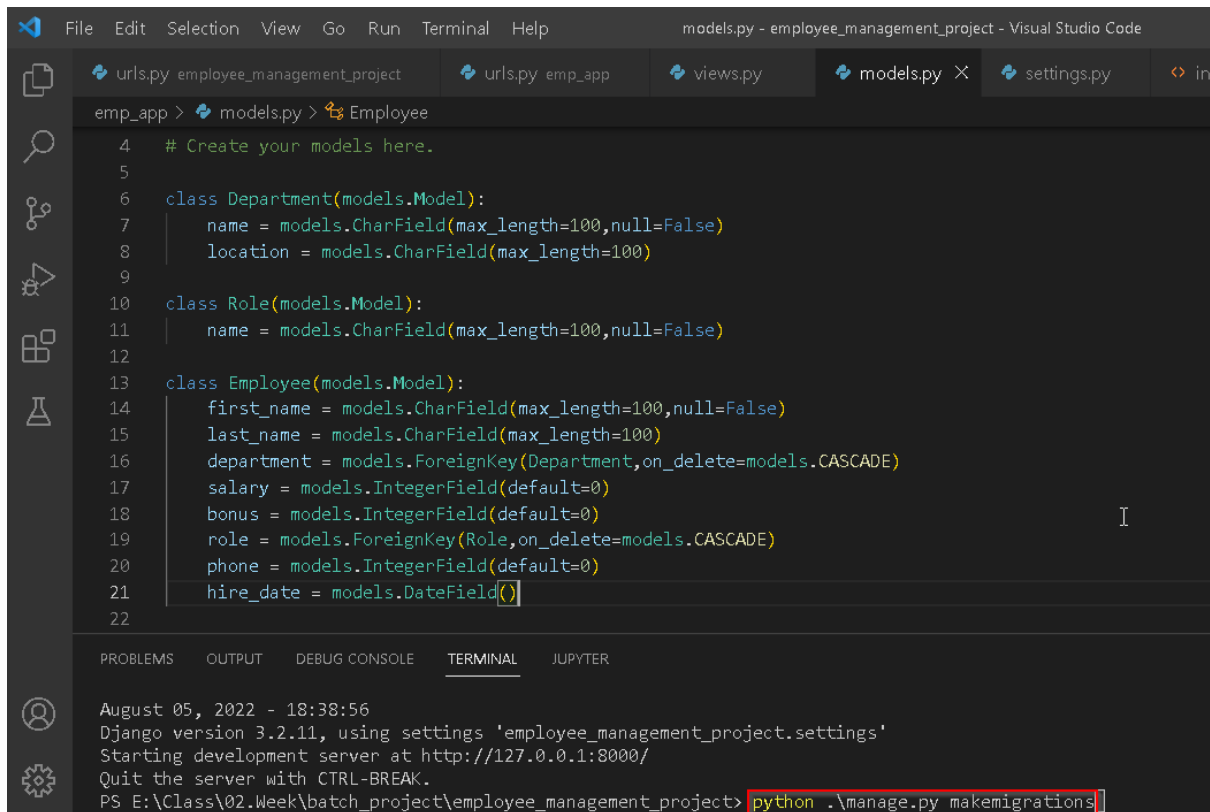
```
python manage.py sqlmigrate app_name migration_name  
python manage.py sqlmigrate app 0001
```

To see all migrations, execute the command:

```
python manage.py showmigrations
```

To see app-specific migrations by specifying app-name, execute the command:

```
python manage.py showmigrations app
```

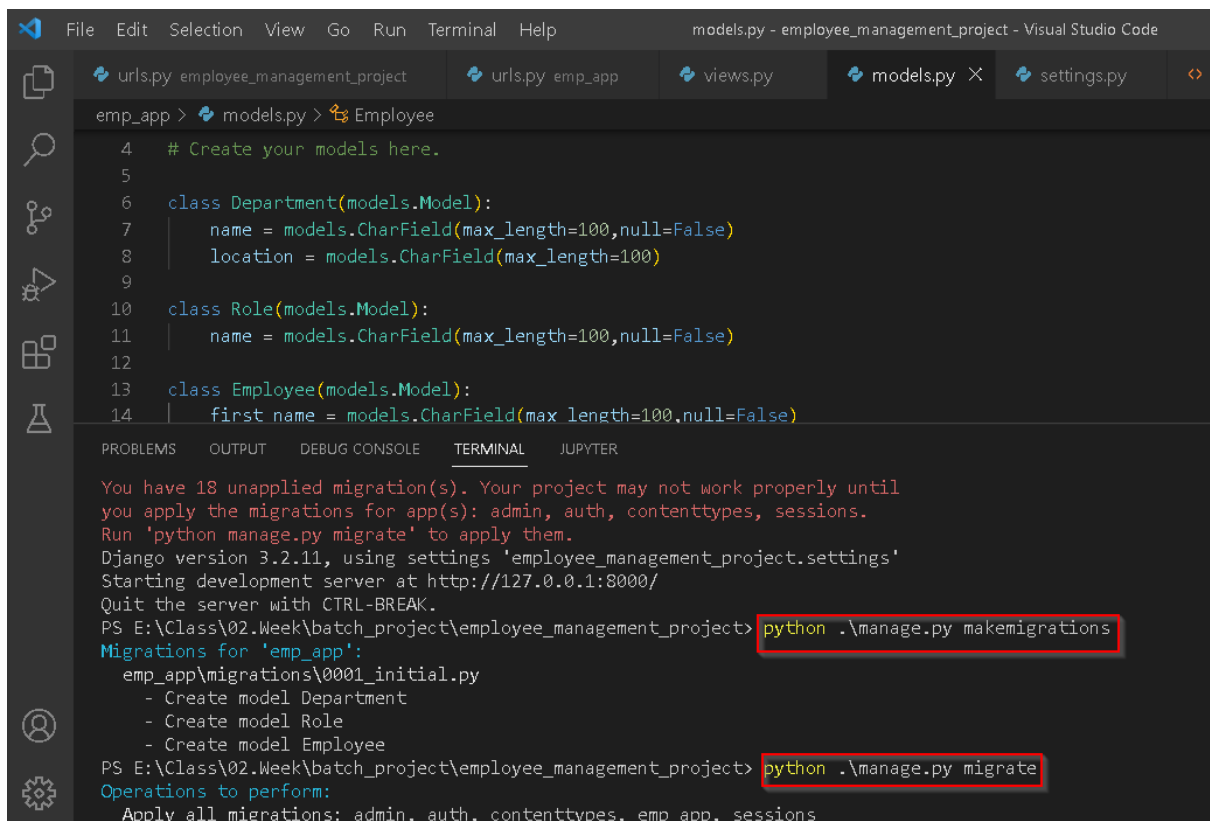


The screenshot shows the Visual Studio Code interface with the Django models file open. The models are defined as follows:

```
4 # Create your models here.
5
6 class Department(models.Model):
7     name = models.CharField(max_length=100,null=False)
8     location = models.CharField(max_length=100)
9
10 class Role(models.Model):
11     name = models.CharField(max_length=100,null=False)
12
13 class Employee(models.Model):
14     first_name = models.CharField(max_length=100,null=False)
15     last_name = models.CharField(max_length=100)
16     department = models.ForeignKey(Department,on_delete=models.CASCADE)
17     salary = models.IntegerField(default=0)
18     bonus = models.IntegerField(default=0)
19     role = models.ForeignKey(Role,on_delete=models.CASCADE)
20     phone = models.IntegerField(default=0)
21     hire_date = models.DateField()
22
```

The terminal output shows the Django version and the start of the development server:

```
August 05, 2022 - 18:38:56
Django version 3.2.11, using settings 'employee_management_project.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.
PS E:\Class\02.Week\batch_project\employee_management_project> python .\manage.py makemigrations
```



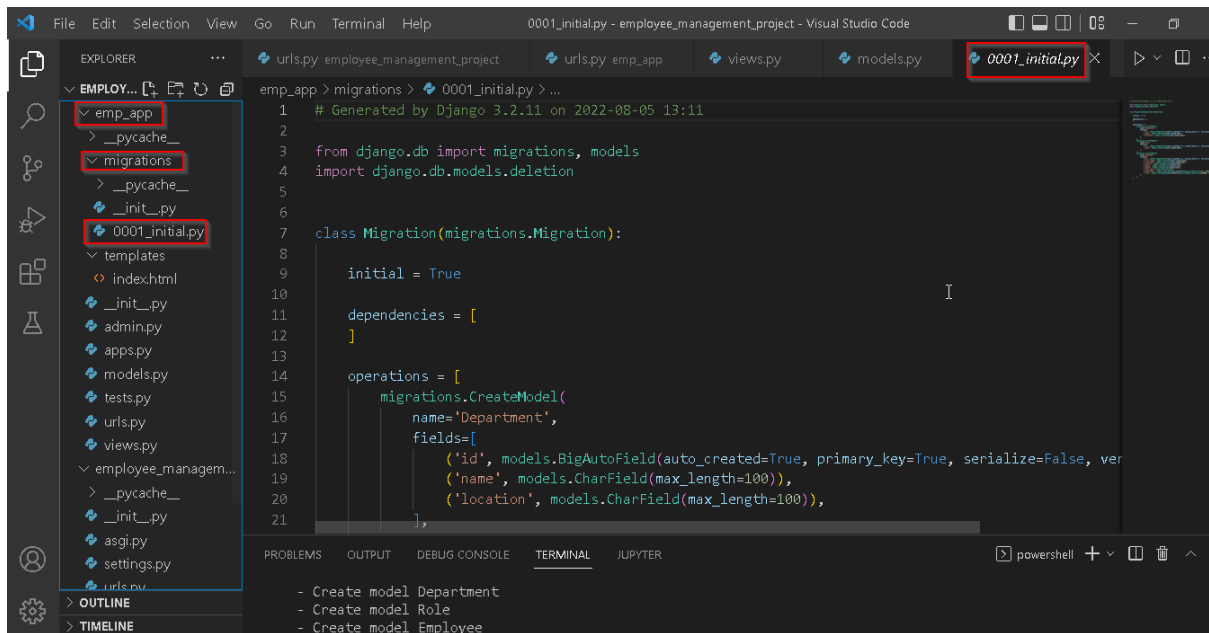
The screenshot shows the Visual Studio Code interface with the Django migrations file open. The migrations are defined as follows:

```
14 first name = models.CharField(max_length=100,null=False)
```

The terminal output shows the Django version and the start of the development server:

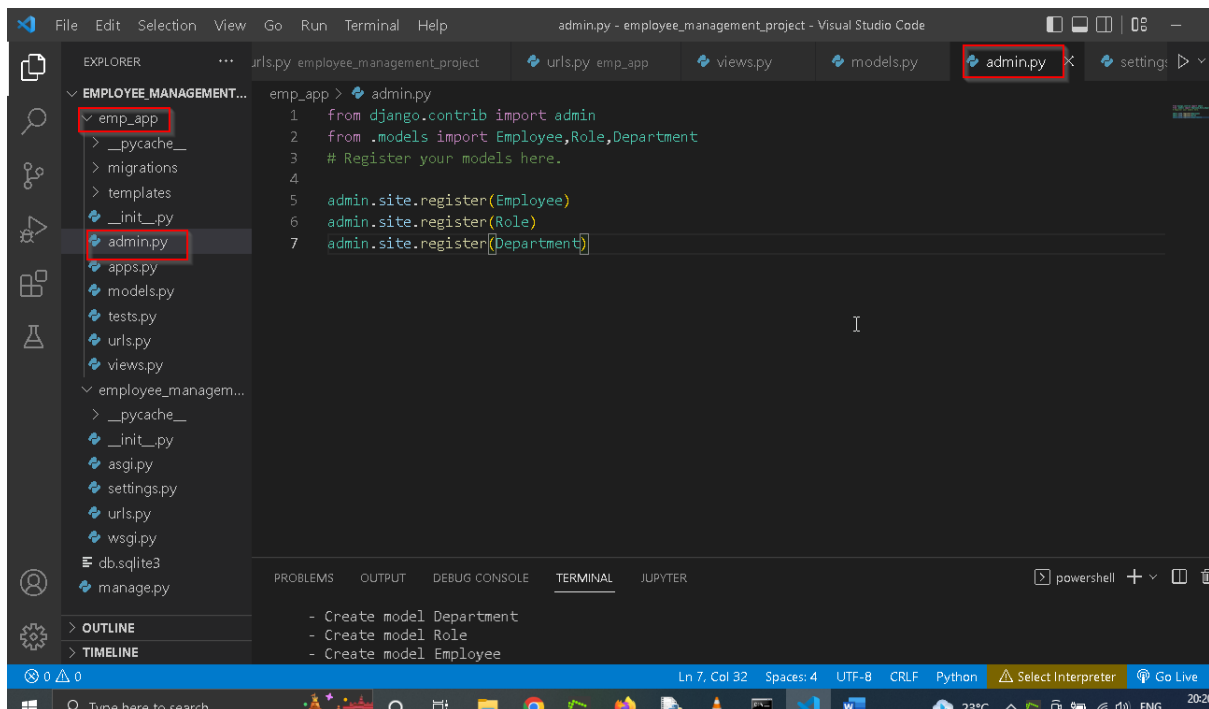
```
You have 18 unapplied migration(s). Your project may not work properly until
you apply the migrations for app(s): admin, auth, contenttypes, sessions.
Run 'python manage.py migrate' to apply them.
Django version 3.2.11, using settings 'employee_management_project.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.
PS E:\Class\02.Week\batch_project\employee_management_project> python .\manage.py makemigrations
Migrations for 'emp_app':
  emp_app\migrations\0001_initial.py
    - Create model Department
    - Create model Role
    - Create model Employee
PS E:\Class\02.Week\batch_project\employee_management_project> python .\manage.py migrate
Operations to perform:
  Apply all migrations: admin, auth, contenttypes, emp_app, sessions
```

After these commands, in migrations folder we can see migration files as follows.



Using admin interface we can put data in database, for that purpose you have to register your model in admin interface, for this we have admin.py file using **admin.site.register(model_name)**, we will do that as follows.

First, we need to import models in admin.py from models file,



Now our models are registered in admin. When we want to access anything on admin interface we need a super user, we need to create that superuser.

What Is The Command To Create A Superuser In Django?

Command To Create A SuperUser:

```
python manage.py createsuperuser
```

Enter your desired username and press enter.

```
Username: admin
```

You will then be prompted for your desired email address:

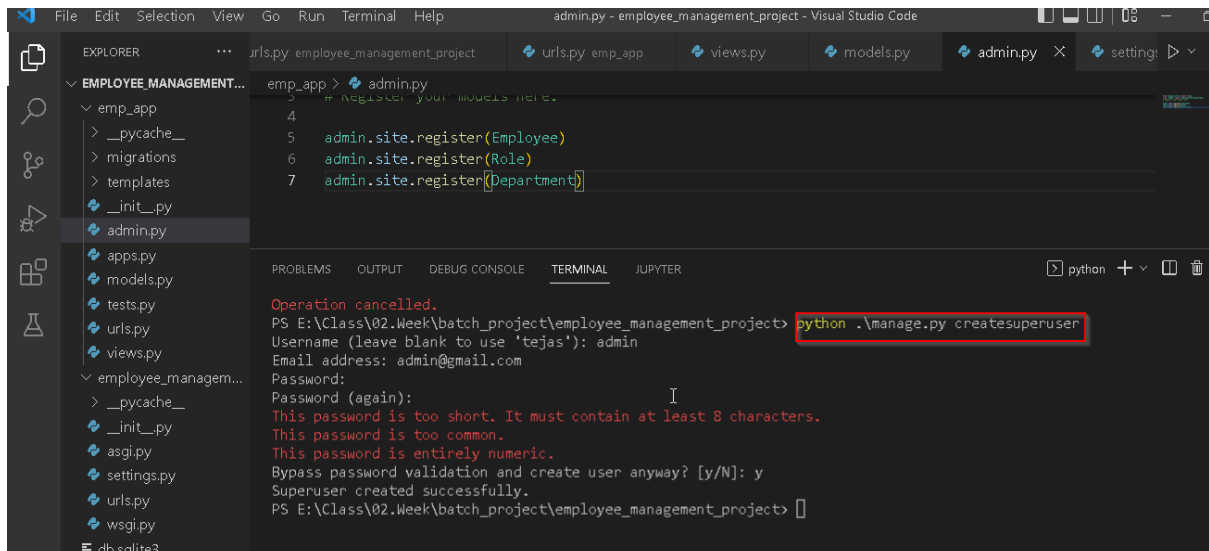
```
Email address: admin@example.com
```

The final step is to enter your password twice,
the second time as a confirmation of the first.

```
Password: *****
```

```
Password (again): *****
```

```
Superuser created successfully.
```

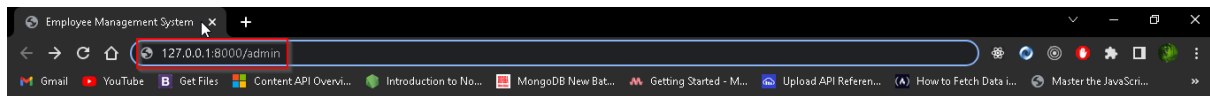


```
admin.py - employee_management_project - Visual Studio Code
EXPLORER
  EMPLOYEE_MANAGEMENT...
    emp_app
      __pycache__
      migrations
      templates
      __init__.py
      admin.py
    apps.py
    models.py
    tests.py
    urls.py
    views.py
  employee_managem...
    __pycache__
    __init__.py
    asgi.py
    settings.py
    urls.py
    wsgi.py
  db.sqlite3

emp_app > admin.py
3 # Register your models here.
4
5 admin.site.register(Employee)
6 admin.site.register(Role)
7 admin.site.register(Department)

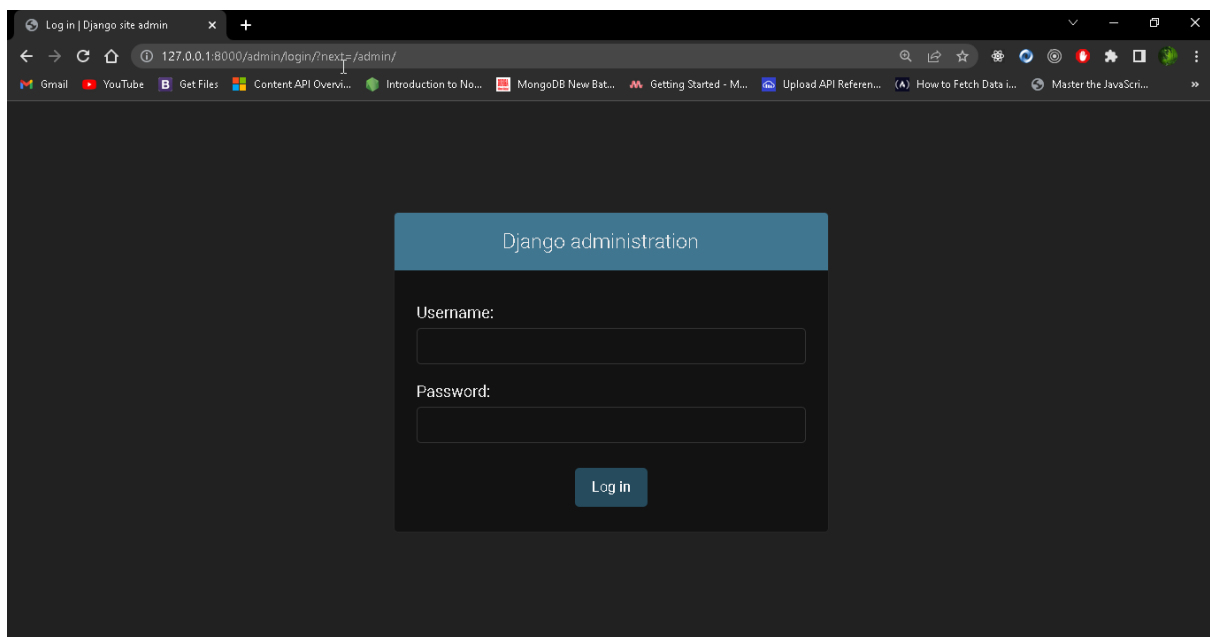
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL JUPYTER
Operation cancelled.
PS E:\Class\02.Week\batch_project\employee_management_project> python .\manage.py createsuperuser
Username (leave blank to use 'tejas'): admin
Email address: admin@gmail.com
Password:
Password (again):
This password is too short. It must contain at least 8 characters.
This password is too common.
This password is entirely numeric.
Bypass password validation and create user anyway? [y/N]: y
Superuser created successfully.
PS E:\Class\02.Week\batch_project\employee_management_project>
```

After this run the server.

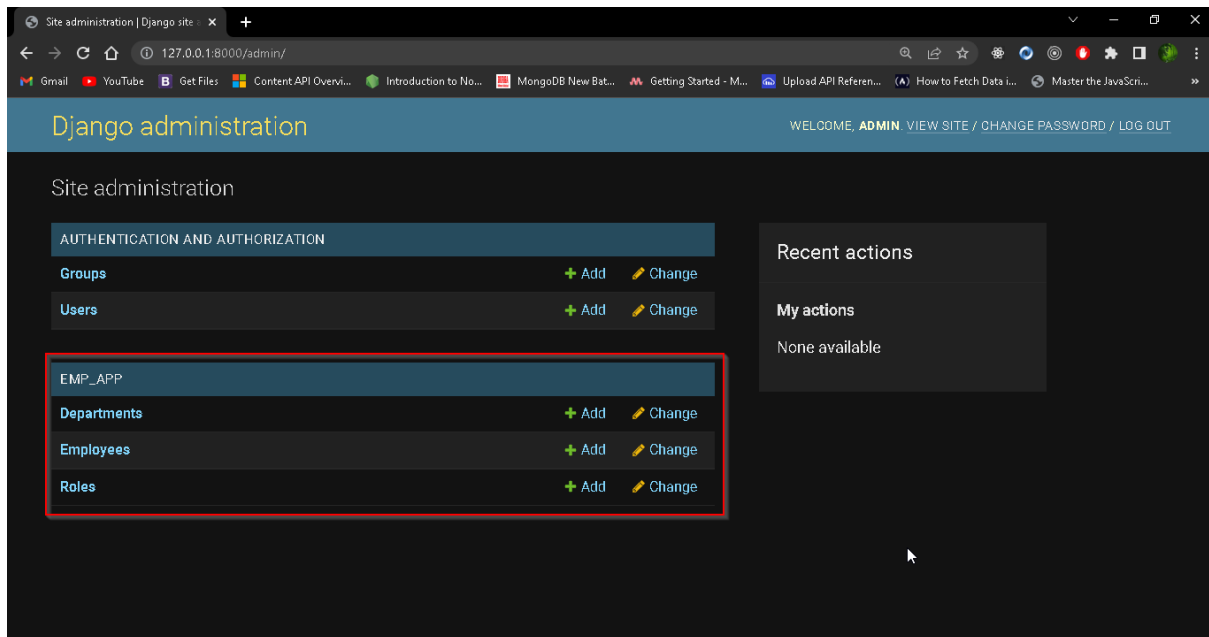


Welcome To Employee Management System Project

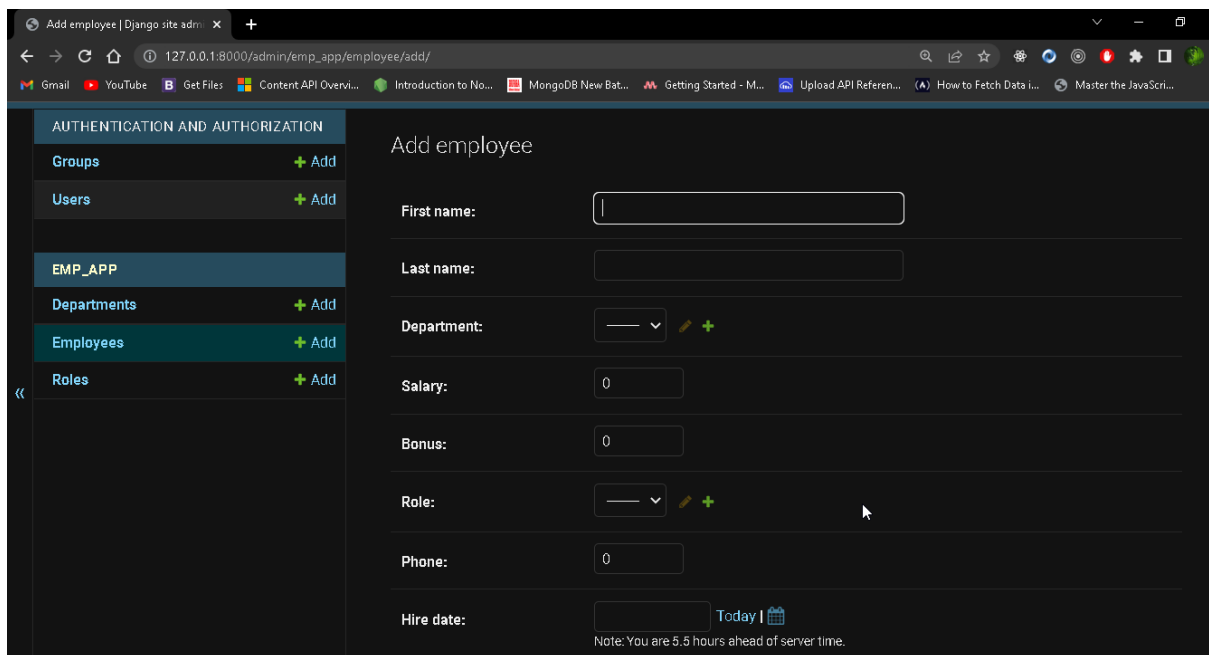
Enter username and password , you have created for superuser



Then it will show all your project's emp_app application's models in admin interface,

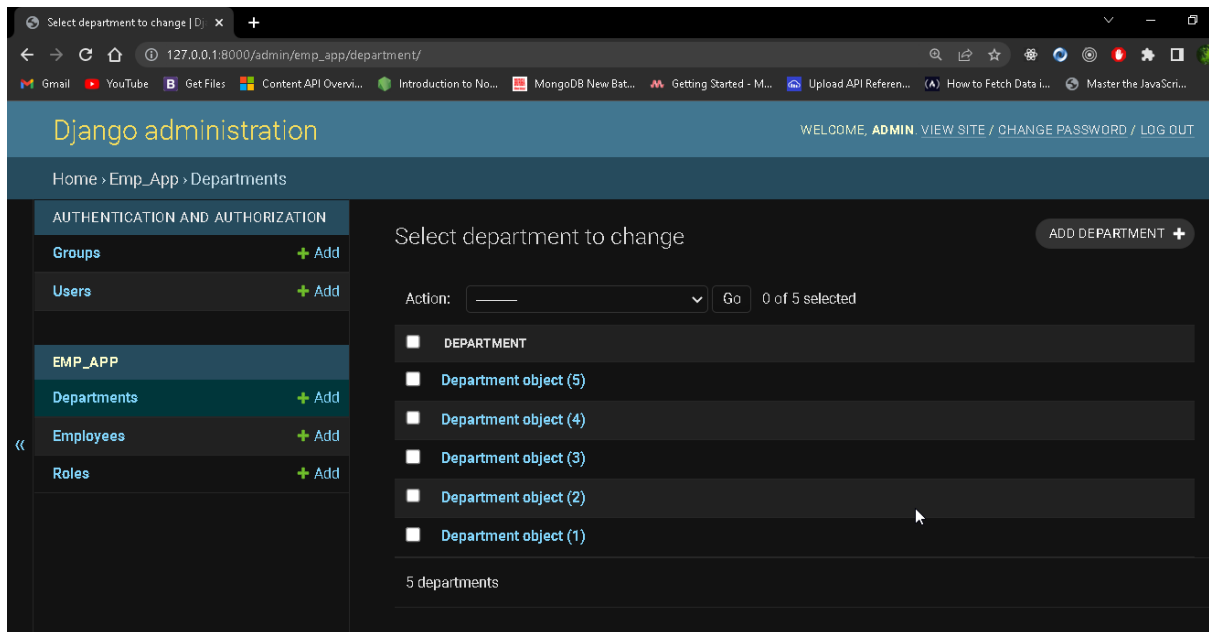
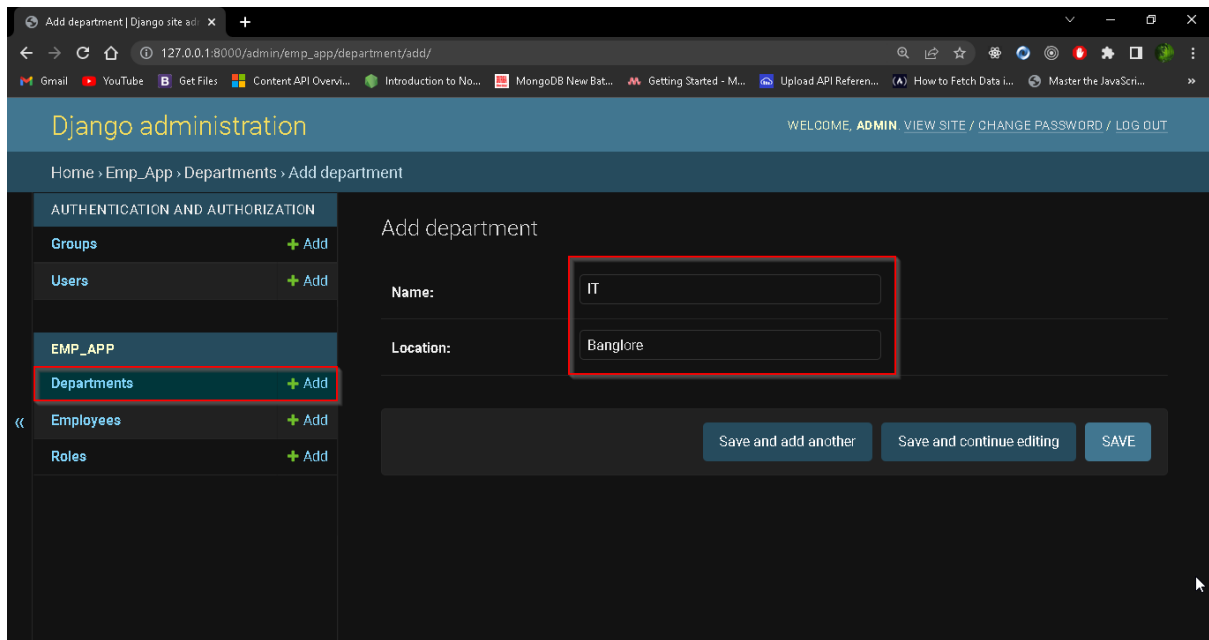


Click on Add button to add employees,

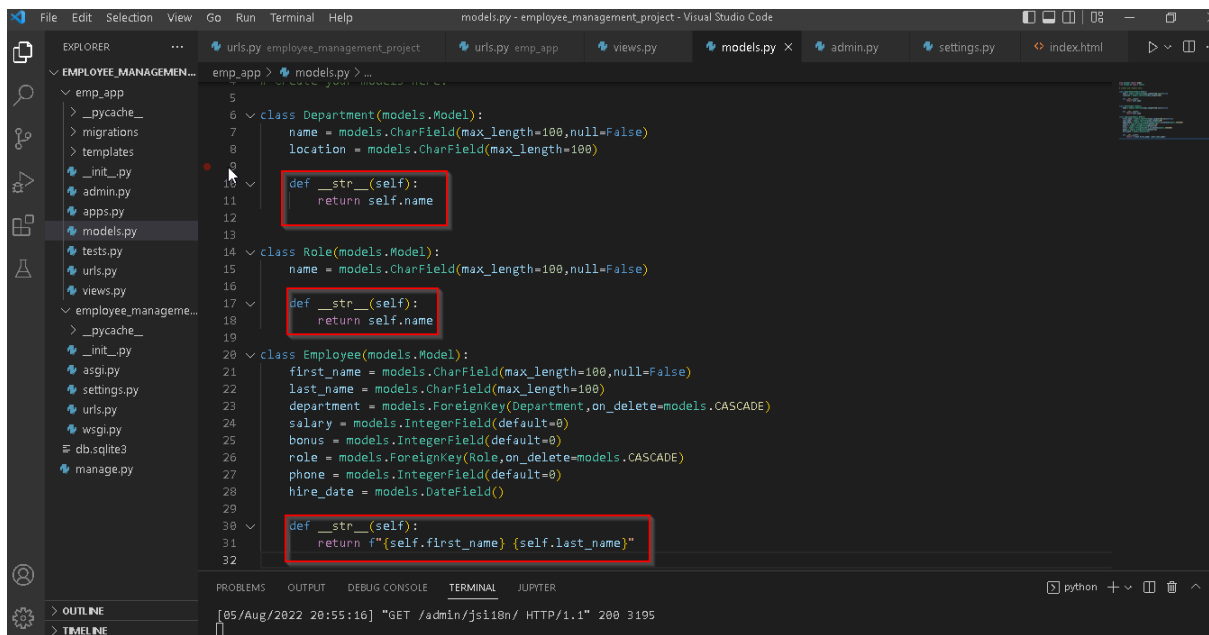


Here before creating employee we need to create Department and Role , because its foreign keys we have given in Employee model.

So first create Departments, you have to save and create as many as you want.

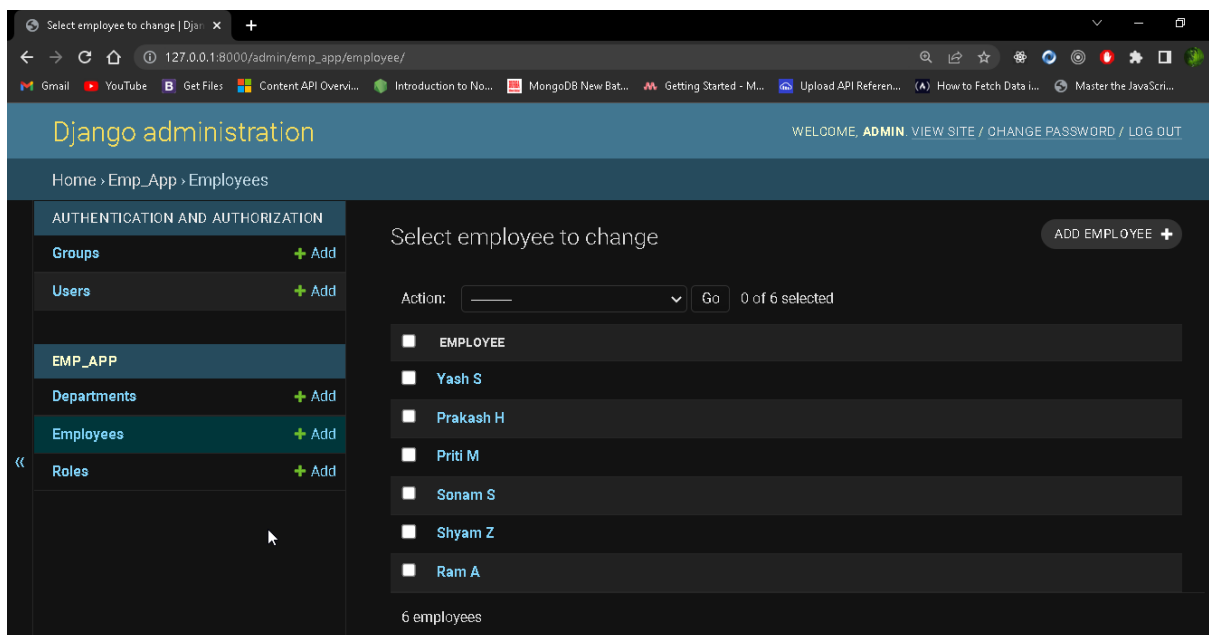


Add data in all tables. In above image for added object it is showing default name which is, Department object, we can customize that by overriding `__str__` method in each class, as follows



```
5 class Department(models.Model):
6     name = models.CharField(max_length=100,null=False)
7     location = models.CharField(max_length=100)
8
9     def __str__(self):
10         return self.name
11
12 class Role(models.Model):
13     name = models.CharField(max_length=100,null=False)
14
15     def __str__(self):
16         return self.name
17
18 class Employee(models.Model):
19     first_name = models.CharField(max_length=100,null=False)
20     last_name = models.CharField(max_length=100)
21     department = models.ForeignKey(Department,on_delete=models.CASCADE)
22     salary = models.IntegerField(default=0)
23     bonus = models.IntegerField(default=0)
24     role = models.ForeignKey(Role,on_delete=models.CASCADE)
25     phone = models.IntegerField(default=0)
26     hire_date = models.DateField()
27
28     def __str__(self):
29         return f'{self.first_name} {self.last_name}'
```

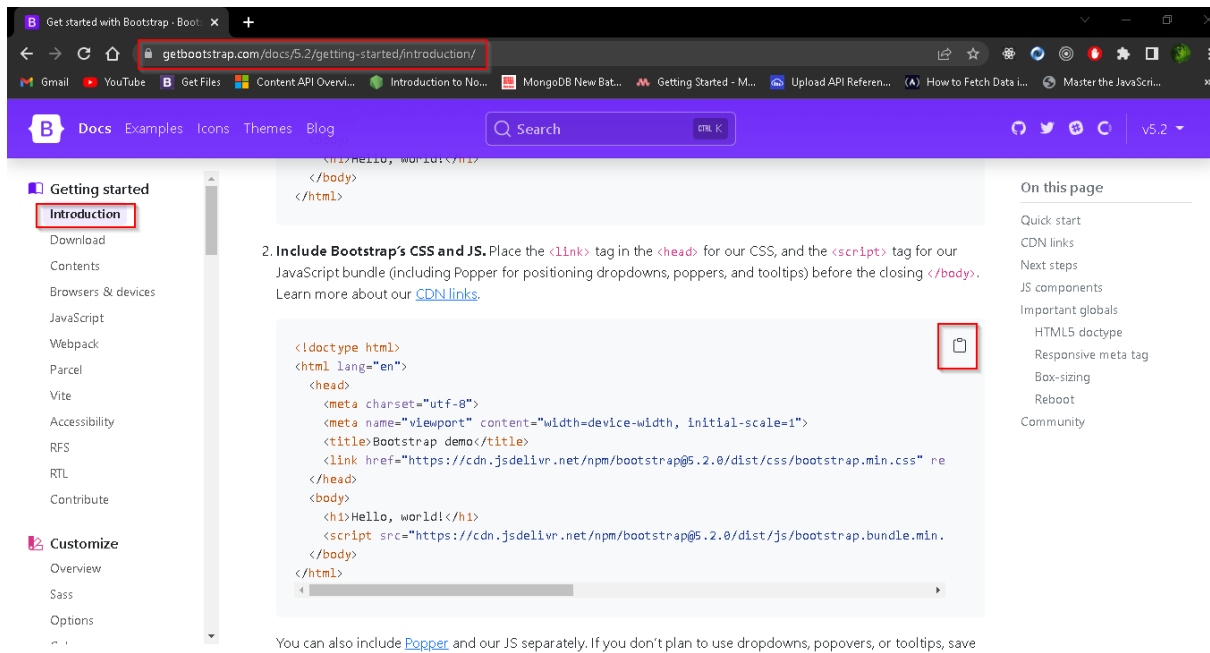
After this we get interface as follows,



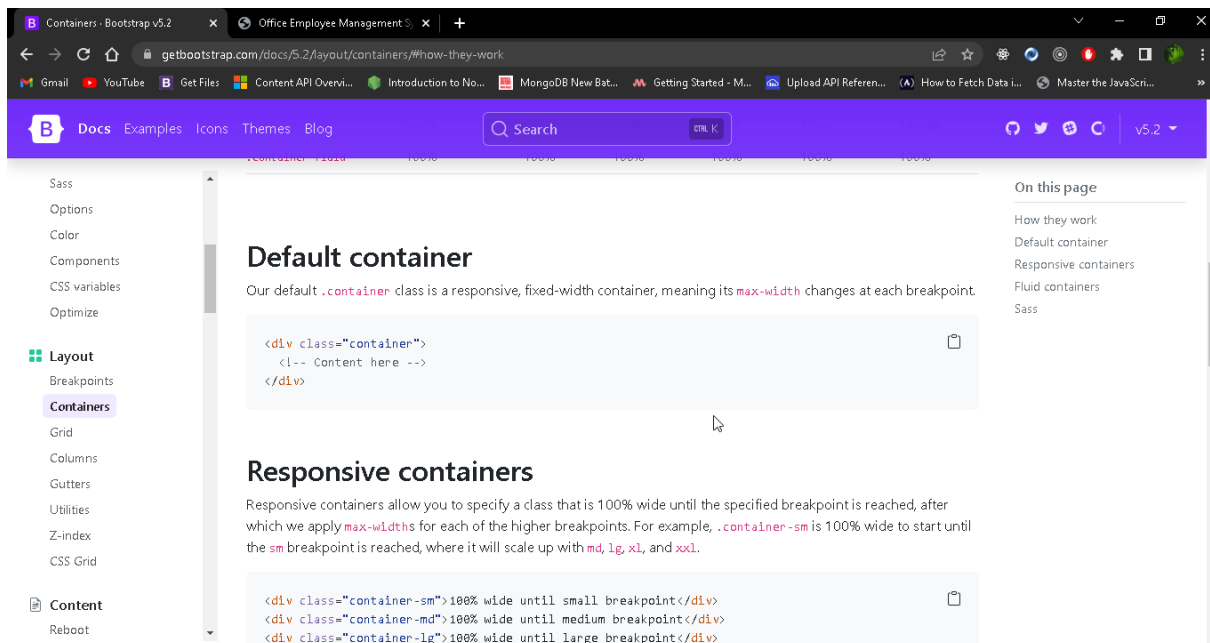
Next we will display html content.

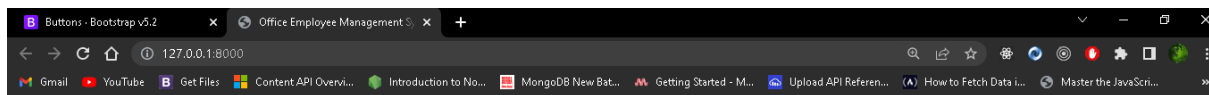
Go to , <https://getbootstrap.com/>

From this website copy the starter template as follows and paste it in index.html.



Copy default container.





Welcome to Employee Management System

Next, we want buttons for our project, take those from get those from bootstrap website.

Buttons - Bootstrap v5.2

Office Employee Management

getbootstrap.com/docs/5.2/components/buttons/#examples

Docs Examples Icons Themes Blog

Search

Buttons

Button tags

The `.btn` classes are designed to be used with the `<button>` element. However, you can also use these classes on `<a>` or `<input>` elements (though some browsers may apply a slightly different rendering).

When using button classes on `<a>` elements that are used to trigger in-page functionality (like collapsing content), rather than linking to new pages or sections within the current page, these links should be given a `role="button"` to appropriately convey their purpose to assistive technologies such as screen readers.

Link Button Input Submit Reset

HTML

```
<a class="btn btn-primary" href="#" role="button">Link</a>
<button class="btn btn-primary" type="submit">Button</button>
<input class="btn btn-primary" type="button" value="Input">
<input class="btn btn-primary" type="submit" value="Submit">
<input class="btn btn-primary" type="reset" value="Reset">
```

On this page

- Examples
- Disable text wrapping
- Button tags
- Outline buttons
- Sizes
- Disabled state
- Link functionality caveat
- Block buttons
- Button plugin
- Toggle states
- Methods
- CSS
- Variables
- Sass variables
- Sass mixins
- Sass loops

File Edit Selection View Go Run Terminal Help

index.html - employee_management_project - Visual Studio Code

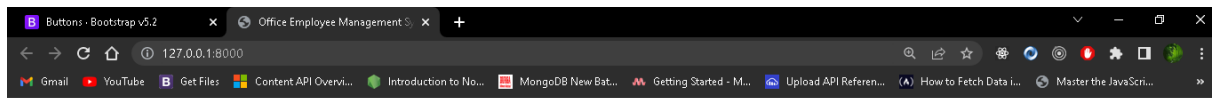
EMPLOYEE_MANAGEMENT_PROJECT

emp_app > templates > index.html

```
1 <!doctype html>
2 <html lang="en">
3
4 <head>
5 <meta charset="utf-8">
6 <meta name="viewport" content="width=device-width, initial-scale=1">
7 <title>Office Employee Management System</title>
8 <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.0/dist/css/bootstrap.min.css" rel="stylesheet" integrity="s
9 </head>
10 <body>
11 <div class="container">
12 <h1>Welcome to Employee Management System</h1>
13 <a class="btn btn-primary" href="/" role="button">View All Employee</a>
14 </div>
15 <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.0/dist/js/bootstrap.bundle.min.js" integrity="sha384-A3rJ0
16 </body>
17 </html>
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL JUPITER

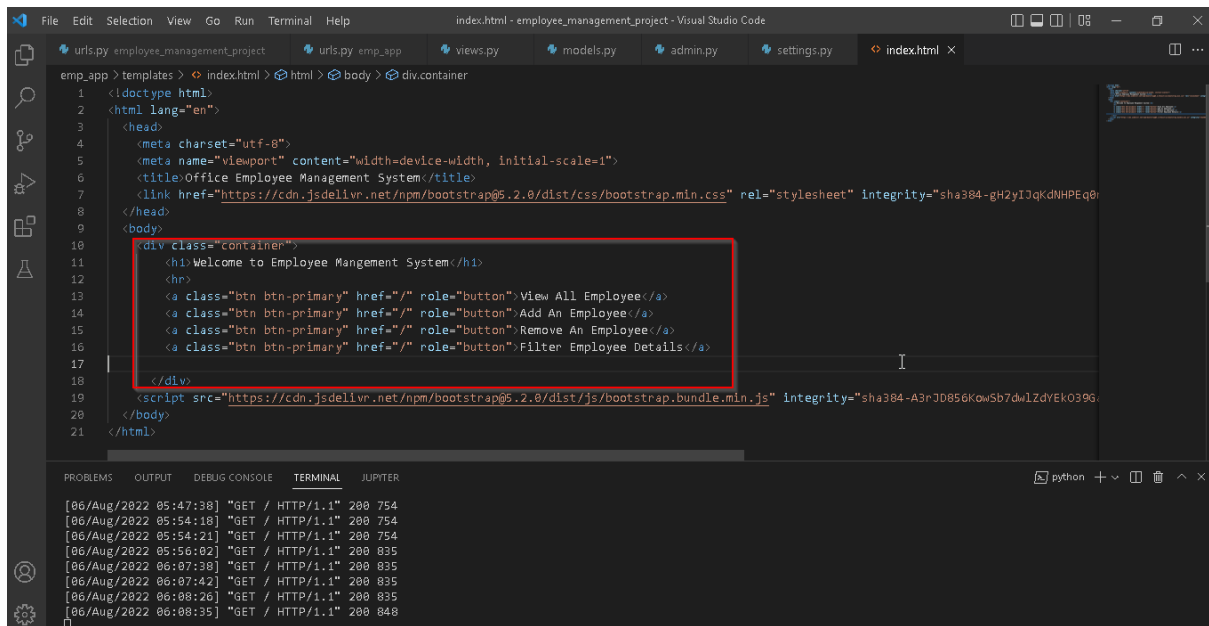
```
[06/Aug/2022 05:42:29] "GET / HTTP/1.1" 200 667
Not Found: /favicon.ico
[06/Aug/2022 05:42:30] "GET /favicon.ico HTTP/1.1" 404 2292
[06/Aug/2022 05:43:08] "GET / HTTP/1.1" 200 679
[06/Aug/2022 05:43:43] "GET / HTTP/1.1" 200 686
[06/Aug/2022 05:47:38] "GET / HTTP/1.1" 200 754
[06/Aug/2022 05:54:18] "GET / HTTP/1.1" 200 754
[06/Aug/2022 05:54:21] "GET / HTTP/1.1" 200 754
```



Welcome to Employee Mangement System

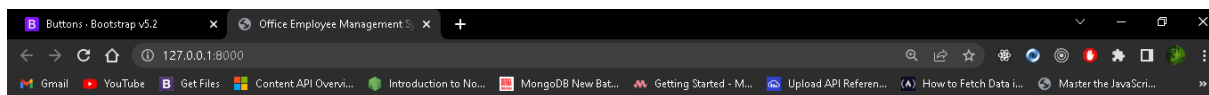
[View All Employee](#)

Add more buttons,



```
emp_app > templates > index.html > html > body > div.container
1 <!doctype html>
2 <html lang="en">
3 <head>
4 <meta charset="utf-8">
5 <meta name="viewport" content="width=device-width, initial-scale=1">
6 <title>Office Employee Management System</title>
7 <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.0/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-gH2yIqKdNHPEq8"
8 </head>
9 <body>
10 <div class="container">
11 <h1>Welcome to Employee Management System</h1>
12 <hr>
13 <a class="btn btn-primary" href="/" role="button">View All Employee</a>
14 <a class="btn btn-primary" href="/" role="button">Add An Employee</a>
15 <a class="btn btn-primary" href="/" role="button">Remove An Employee</a>
16 <a class="btn btn-primary" href="/" role="button">Filter Employee Details</a>
17 </div>
18 <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.0/dist/js/bootstrap.bundle.min.js" integrity="sha384-A3n3D856KowSb7dw1ZdYEk039G"
19 </body>
20 </html>
21
```

After this , you will see as follows.



Welcome to Employee Management System

[View All Employee](#) [Add An Employee](#) [Remove An Employee](#) [Filter Employee Details](#)

```
emp_app > urls.py > ...
1 from django.contrib import admin
2 from django.urls import path
3 from . import views
4
5 urlpatterns = [
6     path('', views.index, name='index'),
7     path('all_emp', views.all_emp, name='all_emp'),
8     path('add_emp', views.add_emp, name='add_emp'),
9     path('remove_emp', views.remove_emp, name='remove_emp'),
10    path('filter_emp', views.filter_emp, name='filter_emp'),
11]
12
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL JUPYTER

System check identified no issues (0 silenced).
August 06, 2022 - 06:30:28
Django version 3.2.11, using settings 'employee_management_project.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.

```
emp_app > views.py > ...
1 from django.shortcuts import render
2
3 # Create your views here.
4
5 def index(request):
6     return render(request, 'index.html')
7
8 def all_emp(request):
9     return render(request, 'view_all_emp.html')
10
11 def add_emp(request):
12     return render(request, 'add_emp.html')
13
14 def remove_emp(request):
15     return render(request, 'remove_emp.html')
16
17 def filter_emp(request):
18     return render(request, 'filter_emp.html')
19
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL JUPYTER

System check identified no issues (0 silenced).
August 06, 2022 - 06:30:28
Django version 3.2.11, using settings 'employee_management_project.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.

```
File Edit Selection View Go Run Terminal Help
index.html - employee_management_project - Visual Studio Code

emp_app > templates > index.html > html > body > div.container > a.btn.btn-primary
1 <doctype html>
2 <html lang="en">
3 <head>
4   <meta charset="utf-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1">
6   <title>Office Employee Management System</title>
7   <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.0/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-gH2YIJqKdNHPEqB8"
8 </head>
9
10 <div class="container">
11   <h1>Welcome to Employee Mangement System</h1>
12   <hr>
13   <a class="btn btn-primary" href="/all_emp" role="button">View All Employee</a>
14   <a class="btn btn-primary" href="/add_emp" role="button">Add An Employee</a>
15   <a class="btn btn-primary" href="/remove_emp" role="button">Remove An Employee</a>
16   <a class="btn btn-primary" href="/filter_emp" role="button">Filter Employee Details</a>
17 </div>
18 <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.0/dist/js/bootstrap.bundle.min.js" integrity="sha384-A3rJD856KowSb7dw1ZdYEkO3Pg"
19 </body>
20 </html>
21

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL JUPYTER

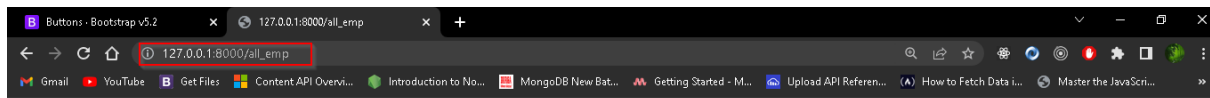
System check identified no issues (0 silenced).
August 06, 2022 - 06:30:28
Django version 3.2.11, using settings 'employee_management_project.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.
```

```
File Edit Selection View Go Run Terminal Help
index.html - employee_management_project - Visual Studio Code

EMPLOYEE_MANAGEMENT... emp_app > templates > index.html > html
1 <doctype html>
2 <html lang="en">
3 <head>
4   <meta charset="utf-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1">
6   <title>Office Employee Management System</title>
7   <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.0/dist/css/bootstrap.min.css" rel="stylesheet" integrity="s
8 </head>
9
10 <body>
11   <div class="container">
12     <h1>Welcome to Employee Mangement System</h1>
13     <hr>
14     <a class="btn btn-primary" href="/all_emp" role="button">View All Employee</a>
15     <a class="btn btn-primary" href="/add_emp" role="button">Add An Employee</a>
16     <a class="btn btn-primary" href="/remove_emp" role="button">Remove An Employee</a>
17     <a class="btn btn-primary" href="/filter_emp" role="button">Filter Employee Details</a>
18   </div>
19   <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.0/dist/js/bootstrap.bundle.min.js" integrity="sha384-A3rJD
20 </body>
21 </html>

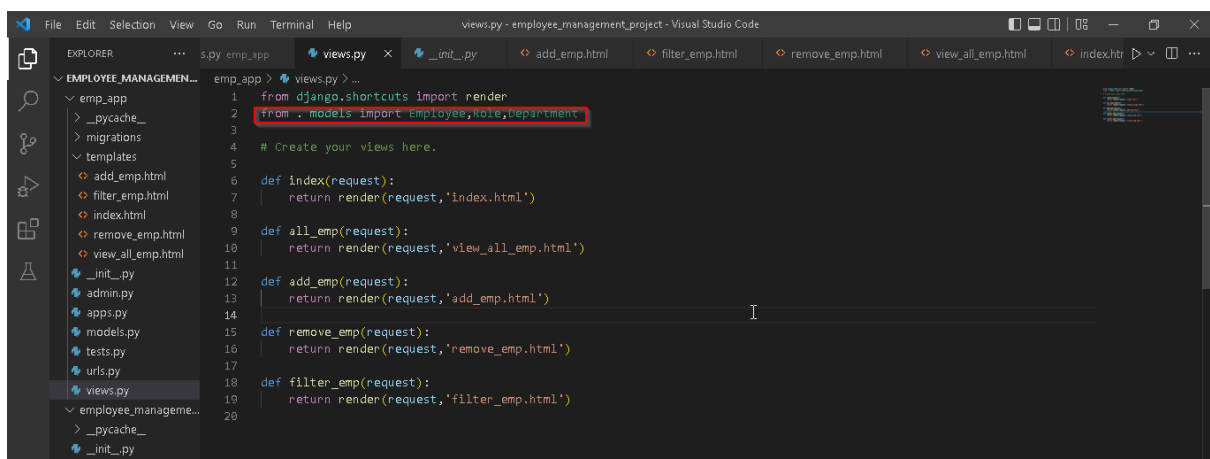
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL JUPYTER

System check identified no issues (0 silenced).
August 06, 2022 - 06:30:28
Django version 3.2.11, using settings 'employee_management_project.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.
```



Now first we will see logic to see all employees.

First import all models from models.py file into views.py file.



First we will implement , logic to view all employees, in all_emp function.

How To View All Items In The Model Using Django QuerySet?

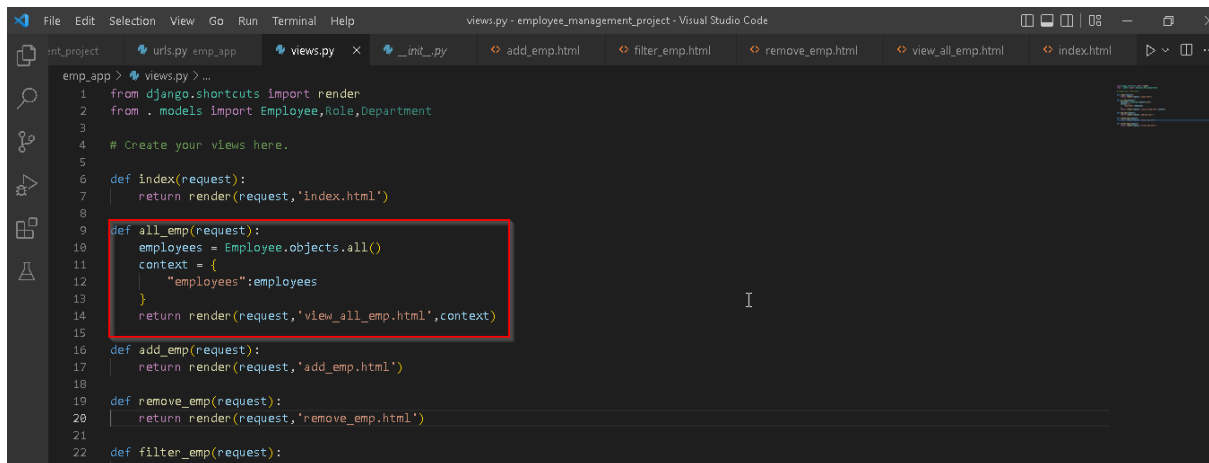
Django Command To View All Items In A Model:

Users.objects.all()

where "User" is a model name.

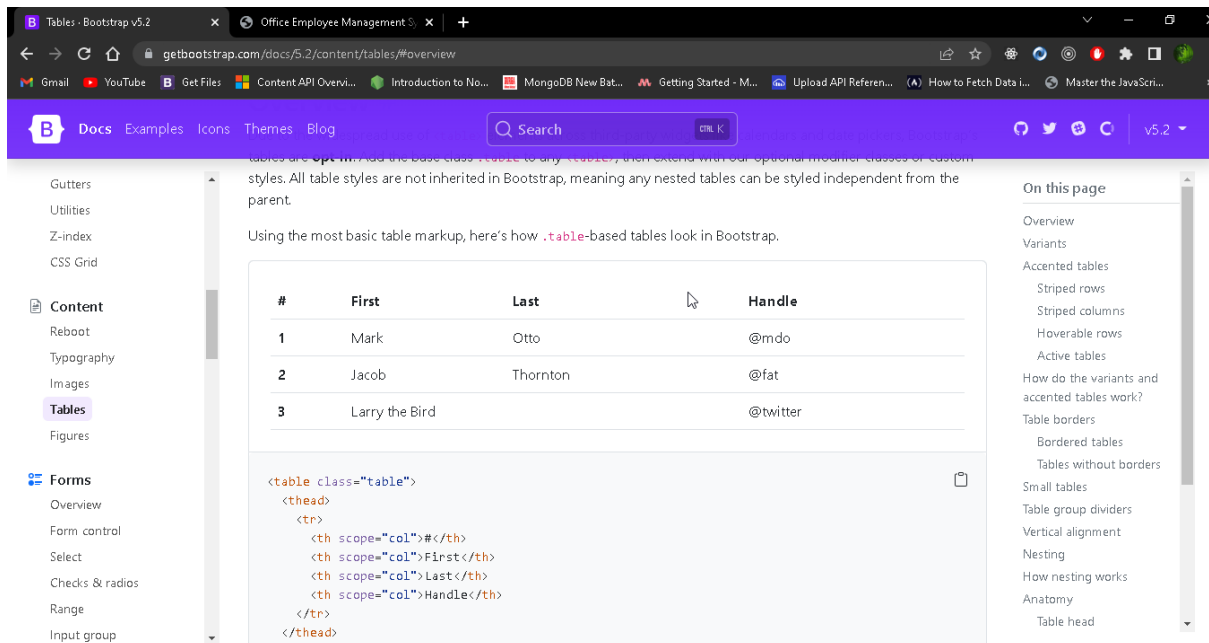
What Is Context In Django?

- ❑ A context in Django is a dictionary, in which keys represent variable names and values represent their values. This dictionary (context) is passed to the template which then uses the variables to output the dynamic content.
- ❑ A context is a variable name -> variable value mapping that is passed to a template.
- ❑ Context processors let you specify a number of variables that get set in each context automatically – without you having to specify the variables in each render() call.



```
1 from django.shortcuts import render
2 from .models import Employee, Role, Department
3
4 # Create your views here.
5
6 def index(request):
7     return render(request, 'index.html')
8
9 def all_emp(request):
10     employees = Employee.objects.all()
11     context = {
12         "employees": employees
13     }
14     return render(request, 'view_all_emp.html', context)
15
16 def add_emp(request):
17     return render(request, 'add_emp.html')
18
19 def remove_emp(request):
20     return render(request, 'remove_emp.html')
21
22 def filter_emp(request):
```

Now we want to see all employees in html page, so we will show that in table format, for this we will copy table from bootstrap.com



Tables are **opt-in**. Add the `table` class to any `<table>`, then extend with our optional modifier classes or custom styles. All table styles are not inherited in Bootstrap, meaning any nested tables can be styled independent from the parent.

Using the most basic table markup, here's how `.table`-based tables look in Bootstrap.

#	First	Last	Handle
1	Mark	Otto	@mdo
2	Jacob	Thornton	@fat
3	Larry the Bird		@twitter

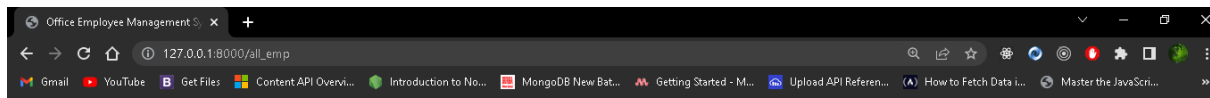
```
<table class="table">
  <thead>
    <tr>
      <th scope="col">#</th>
      <th scope="col">First</th>
      <th scope="col">Last</th>
      <th scope="col">Handle</th>
    </tr>
  </thead>
  <tbody>
```

```
4 <meta charset="utf-8">
5 <meta name="viewport" content="width=device-width, initial-scale=1">
6 <title>Office Employee Management System</title>
7 <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.0/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-gH2yIJqKdNHPEqQb"
8 </head>
9 <body>
10 <div class="container">
11 <h1>View All Employee Employee</h1>
12 <table class="table">
13 <thead>
14 <tr>
15 <th scope="col">#</th>
16 <th scope="col">First Name</th>
17 <th scope="col">Last Name</th>
18 <th scope="col">Salary</th>
19 <th scope="col">Bonus</th>
20 <th scope="col">Phone Number</th>
21 <th scope="col">Role</th>
22 <th scope="col">Department</th>
23 <th scope="col">Location</th>
24 <th scope="col">Hire Date</th>
25 </tr>
26 </thead>
27 <tbody>
28 <tr>
29 <th scope="row">1</th>
30 <td>Mark</td>
```

#	First Name	Last Name	Salary	Bonus	Phone Number	Role	Department	Location	Hire Date
1	Mark	Otto	@mdo						
2	Jacob	Thornton	@fat						
3	Larry the Bird		@twitter						

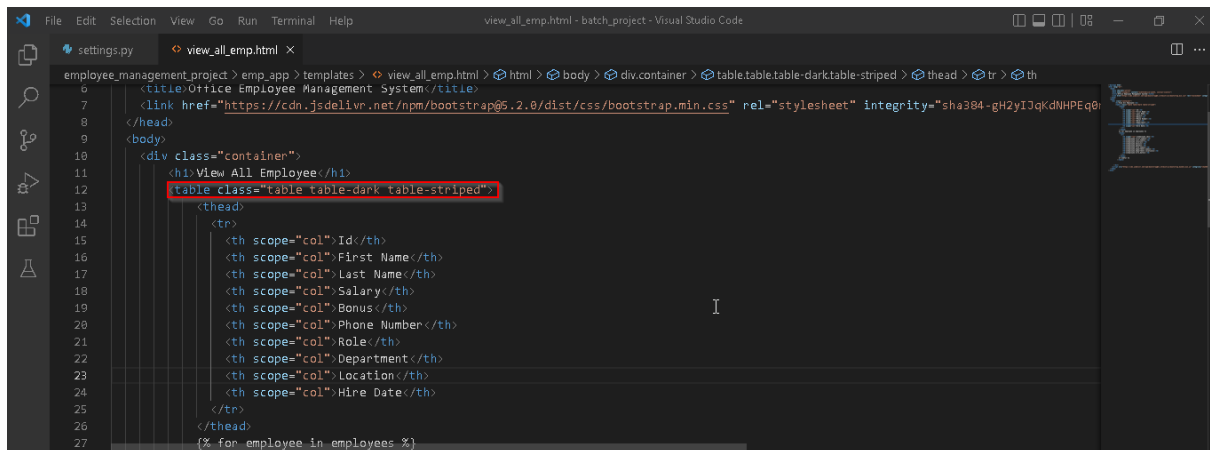
Using templating language we can implement logic in html file, eg. in following code, we created for loop to access each employee details.

```
28 <th scope="col">Phone Number</th>
29 <th scope="col">Role</th>
30 <th scope="col">Department</th>
31 <th scope="col">Location</th>
32 <th scope="col">Hire Date</th>
33 </tr>
34 </thead>
35 <tbody>
36 <tr>
37 <th scope="row">{{employee.id}}</th>
38 <td>{{employee.first_name}}</td>
39 <td>{{employee.last_name}}</td>
40 <td>{{employee.salary}}</td>
41 <td>{{employee.bonus}}</td>
42 <td>{{employee.phone}}</td>
43 <td>{{employee.role.name}}</td>
44 <td>{{employee.department.name}}</td>
45 <td>{{employee.department.location}}</td>
46 <td>{{employee.hire_date}}</td>
47 </tr>
48 </tbody>
49 </table>
50 </div>
```

View All Employee

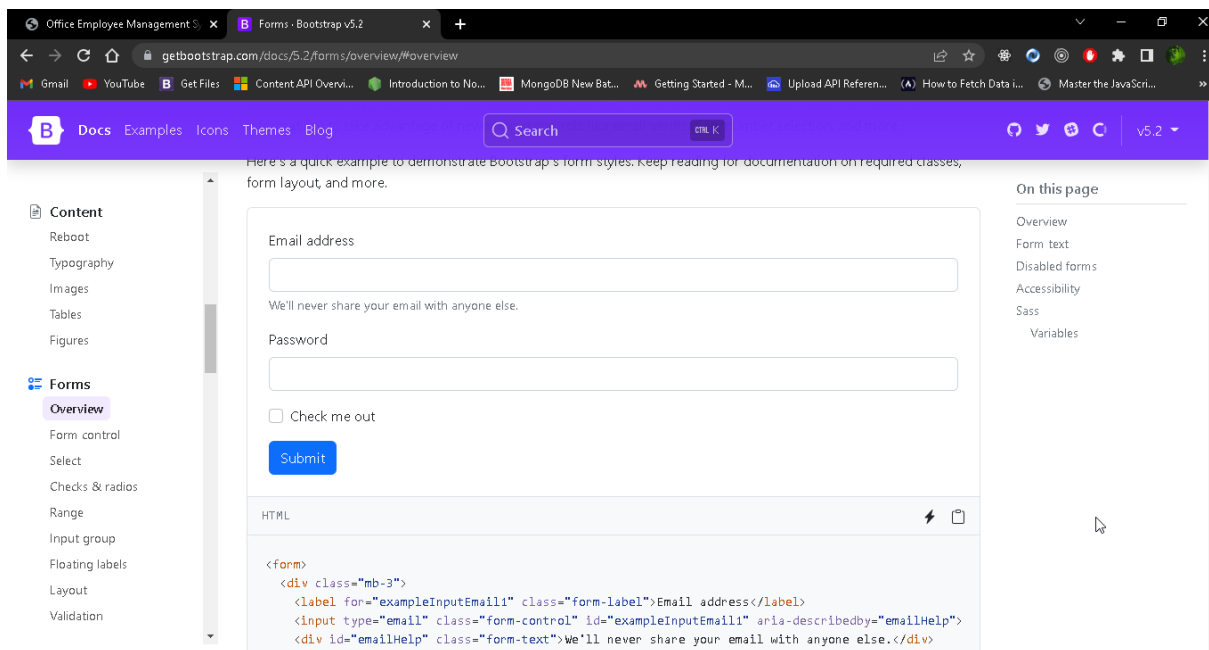
#	First Name	Last Name	Salary	Bonus	Phone Number	Role	Department	Location	Hire Date
1	Ram	A	30000	2000	1234567	Python Developer	IT	Banglore	Aug. 1, 2022
2	Shyam	Z	20000	1000	12345678	Marketing Manager	Accounting	Hydrabaad	Aug. 2, 2022
3	Sonam	S	30000	2000	1234567	Sales Manager	Marketing	Mumbai	Aug. 3, 2022
4	Priti	M	40000	3000	1234567	Mechanical Engineer	Sales	Delhi	Aug. 4, 2022
5	Prakash	H	70000	40000	1234567	Mechanical Engineer	Sales	Delhi	Aug. 5, 2022
6	Yash	S	90000	5000	123456	Chartered Accountant	Manufacturing	Pune	Aug. 5, 2022



View All Employee

	First Name	Last Name	Salary	Bonus	Phone Number	Role	Department	Location	Hire Date
1	Ram	A	30000	2000	1234567	Python Developer	IT	Banglore	Aug. 1, 2022
2	Shyam	Z	20000	1000	12345678	Marketing Manager	Accounting	Hydrabaad	Aug. 2, 2022
3	Sonam	S	30000	2000	1234567	Sales Manager	Marketing	Mumbai	Aug. 3, 2022
4	Priti	M	40000	3000	1234567	Mechanical Engineer	Sales	Delhi	Aug. 4, 2022
5	Prakash	H	70000	40000	1234567	Mechanical Engineer	Sales	Delhi	Aug. 5, 2022
6	Yash	S	90000	5000	123456	Chartered Accountant	Manufacturing	Pune	Aug. 5, 2022

Add an Employee



Office Employee Management

Forms - Bootstrap v5.2

getbootstrap.com/docs/5.2/forms/overview/#overview

Docs Examples Icons Themes Blog

Search

v5.2

Content

- Reboot
- Typography
- Images
- Tables
- Figures

Forms

- Overview
- Form control
- Select
- Checks & radios
- Range
- Input group
- Floating labels
- Layout
- Validation

Here's a quick example to demonstrate Bootstrap's form styles. Keep reading for documentation on required classes, form layout, and more.

Email address

We'll never share your email with anyone else.

Password

☐ Check me out

Submit

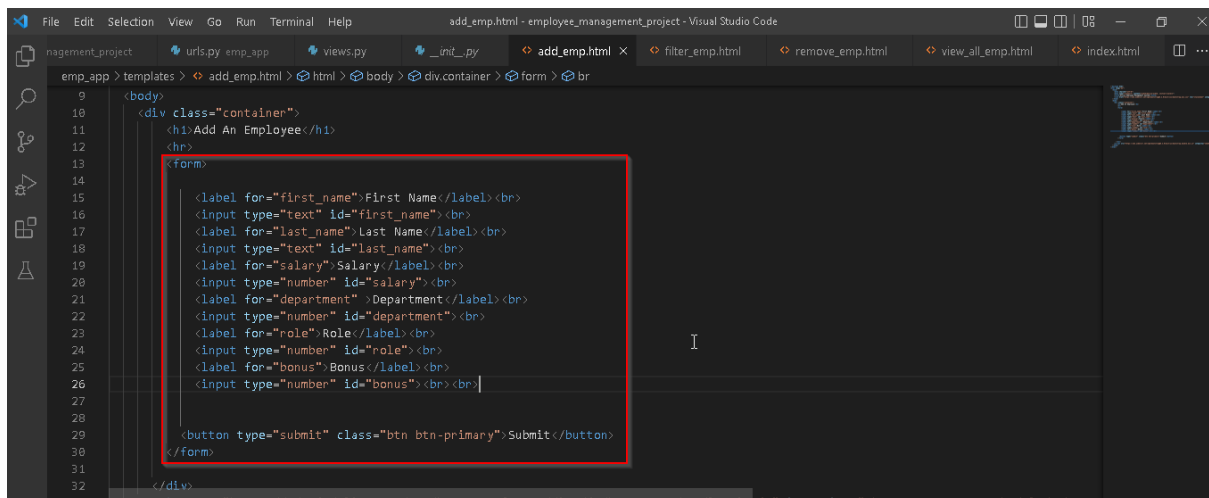
HTML

```
<form>
  <div class="mb-3">
    <label for="exampleInputEmail1" class="form-label">Email address</label>
    <input type="email" class="form-control" id="exampleInputEmail1" aria-describedby="emailHelp">
    <div id="emailHelp" class="form-text">We'll never share your email with anyone else.</div>
  ...
```

On this page

- Overview
- Form text
- Disabled forms
- Accessibility
- Sass
- Variables

Paste this form in add an employee html page, customize it as per your requirements.



File Edit Selection View Go Run Terminal Help

add_emp.html - employee_management_project - Visual Studio Code

emp_app > templates > add_emp.html > html > body > div.container > form > br

```
9 <body>
10 <div class="container">
11 <h1>Add An Employee</h1>
12 <hr>
13 <form>
14
15 <label for="first_name">First Name</label> <br>
16 <input type="text" id="first_name"> <br>
17 <label for="last_name">Last Name</label> <br>
18 <input type="text" id="last_name"> <br>
19 <label for="salary">Salary</label> <br>
20 <input type="number" id="salary"> <br>
21 <label for="department">Department</label> <br>
22 <input type="number" id="department"> <br>
23 <label for="role">Role</label> <br>
24 <input type="number" id="role"> <br>
25 <label for="bonus">Bonus</label> <br>
26 <input type="number" id="bonus"> <br> <br>
27
28 <button type="submit" class="btn btn-primary">Submit</button>
29 </form>
30
31 </div>
32
```

Added action and method,

```
<body>
  <div class="container">
    <h1>Add An Employee</h1>
    <hr>
    <form action="/add_emp" method="post">
      {% csrf_token %}
      <label for="first_name">First Name</label><br>
      <input type="text" id="first_name"><br>
      <label for="last_name">Last Name</label><br>
      <input type="text" id="last_name"><br>
      <label for="salary">Salary</label><br>
      <input type="number" id="salary"><br>
      <label for="department">Department</label><br>
      <input type="number" id="department"><br>
      <label for="role">Role</label><br>
      <input type="number" id="role"><br>
      <label for="bonus">Bonus</label><br>
      <input type="number" id="bonus"><br><br>
      <button type="submit" class="btn btn-primary">Submit</button>
    </form>
  </div>
</body>
```

Added value attribute to each input tag,

```
File Edit Selection View Go Run Terminal Help
add_emp.html - employee_management_project - Visual Studio Code

emp_app > templates > add_emp.html > html > body > div.container > form > br
9      <body>
10     <div class="container">
11       <h1>Add An Employee</h1>
12       <hr>
13       <form action="/add_emp" method="post">
14         {% csrf_token %}
15         <label for="first_name">First Name</label><br>
16         <input type="text" id="first_name" value="{{first_name}}"><br>
17         <label for="last_name">Last Name</label><br>
18         <input type="text" id="last_name" value="{{last_name}}"><br>
19         <label for="salary">Salary</label><br>
20         <input type="number" id="salary" value="{{salary}}"><br>
21         <label for="department">Department</label><br>
22         <input type="number" id="department" value="{{department}}"><br>
23         <label for="role">Role</label><br>
24         <input type="number" id="role" value="{{role}}"><br>
25         <label for="bonus">Bonus</label><br>
26         <input type="number" id="bonus" value="{{bonus}}"><br><br>
27         <button type="submit" class="btn btn-primary">Submit</button>
28       </form>
29     </div>
30   </body>
31   <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.0/dist/js/bootstrap.bundle.min.js" integrity="sha384-A3r3D856KowSb7dw1ZdYEkO39G"
32   </script>
```

Next we will write logic of adding employee in views.py file. Following code is based on this logic.

Queryset To Insert/Add An Object:

```
new_object = ModelClass(var1=value1,var2=value2)
```

```
new_object.save()
```

```
def add_emp(request):
    if request.method == "POST":
        first_name = request.POST["first_name"]
        last_name = request.POST["last_name"]
        salary = int(request.POST["salary"])
        bonus = int(request.POST["bonus"])
        phone = int(request.POST["phone"])
        department = int(request.POST["department"])
        role = int(request.POST["role"])
        new_employee = Employee(first_name=first_name,
                                last_name=last_name,
                                salary=salary,
                                bonus=bonus,
                                phone=phone,
                                department_id=department,
                                role_id=role,
                                hire_date = datetime.now())
        new_employee.save()
        return HttpResponseRedirect("Employee added Successfully")
    elif request.method=="GET":
        return render(request,'add_emp.html')
    else:
        return HttpResponseRedirect("An Exception Occured! Employee cant be added")
```

Getting following error, because in html file, I have not specified name attribute for input fields.

C:\python10\lib\site-packages\django\core\handlers\base.py, line 181, in _get_response

```
181.         response = wrapped_callback(request, *callback_args, **callback_kwargs)
```

► Local vars

E:\Class\02.Week\batch_project\employee_management_project\emp_app\views.py, line 19, in add_emp

```
19.         first_name = request.POST["first_name"]
```

▼ Local vars

Variable	Value
request	<WSGIRequest: POST '/add_emp'>

C:\python10\lib\site-packages\django\utils\datastructures.py, line 78, in __getitem__

```
78.         raise MultiValueDictKeyError(key)
```

▼ Local vars

Variable	Value
__class__	<class 'django.utils.datastructures.MultiValueDict'>
key	'first_name'
self	<QueryDict: {'csrfmiddlewaretoken': ['11RY5xKm6dP60tUoNcccd1ZStMjiVLG2WrF0LudXosSTgagDTWg2uCEuJ25Ur2Np']}>

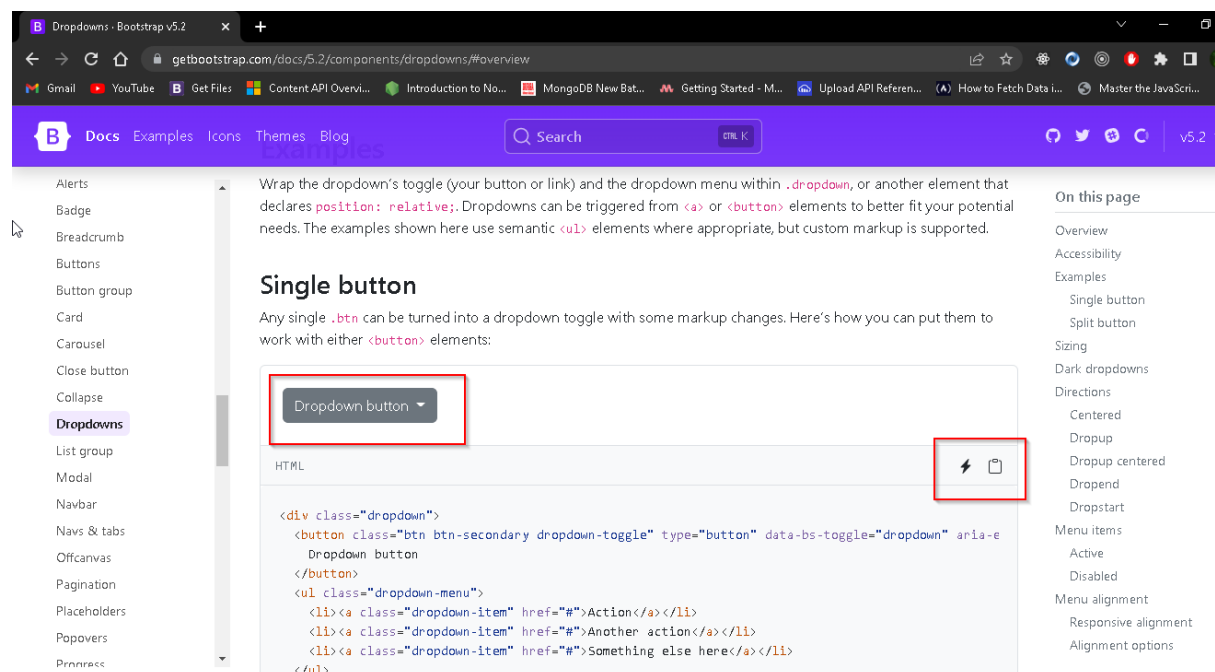
Added name attribute for each input field.

```
<form action="/add_emp" method="post">
  {% csrf_token %}
  <label for="first_name">First Name</label><br>
  <input type="text" id="first_name" name="first_name" value="{{ first_name }}"><br>
  <label for="last_name">Last Name</label><br>
  <input type="text" id="last_name" name="last_name" value="{{ last_name }}"><br>
  <label for="salary">Salary</label><br>
  <input type="number" id="salary" name="salary" value="{{ salary }}"><br>
  <label for="department">Department</label><br>
  <input type="number" id="department" name="department" value="{{ department }}"><br>
  <label for="role">Role</label><br>
  <input type="number" id="role" name="role" value="{{ role }}"><br>
  <label for="bonus">Bonus</label><br>
  <input type="number" id="bonus" name="bonus" value="{{ bonus }}"><br>
  <label for="phone">Phone</label><br>
  <input type="number" id="phone" name="phone" value="{{ phone }}"><br><br>
  <button type="submit" class="btn btn-primary">Submit</button>
</form>
```

Remove an employee., based on Id we will delete that employee.

We will display its ids , we will keep one drop down, in those, these ids will be visible.

Go to get bootstrap.com



Dropdowns - Bootstrap v5.2

getbootstrap.com/docs/5.2/components/dropdowns/#overview

Docs Examples Icons Themes Blog

Search

Alerts
Badge
Breadcrumb
Buttons
Button group
Card
Carousel
Close button
Collapse
Dropdowns
List group
Modal
Navbar
Navs & tabs
Offcanvas
Pagination
Placeholders
Popovers
Progress

Wrap the dropdown's toggle (your button or link) and the dropdown menu within `.dropdown`, or another element that declares `position: relative`. Dropdowns can be triggered from `<a>` or `<button>` elements to better fit your potential needs. The examples shown here use semantic `` elements where appropriate, but custom markup is supported.

Single button

Any single `.btn` can be turned into a dropdown toggle with some markup changes. Here's how you can put them to work with either `<button>` elements:

Dropdown button

HTML

```
<div class="dropdown">
  <button class="btn btn-secondary dropdown-toggle" type="button" data-bs-toggle="dropdown" aria-expanded="false">
    Dropdown button
  </button>
  <ul class="dropdown-menu">
    <li><a class="dropdown-item" href="#">Action</a></li>
    <li><a class="dropdown-item" href="#">Another action</a></li>
    <li><a class="dropdown-item" href="#">Something else here</a></li>
  </ul>
</div>
```

On this page

- Overview
- Accessibility
- Examples
 - Single button
 - Split button
- Sizing
- Dark dropdowns
- Directions
 - Centered
- Dropup
- Dropup centered
- Dropend
- Dropstart
- Menu items
 - Active
 - Disabled
- Menu alignment
- Responsive alignment
- Alignment options

We will write logic of removing employee, in views.py file, get employee from employees table where its id is matching.

```
File Edit Selection View Go Run Terminal Help
views.py - employee_management_project - Visual Studio Code
uris.py employee_management_project uris.py emp_app views.py x add_emp.html filter_emp.html remove_emp.html view_all_emp.html index.htm
emp_app > views.py > ...
39         return HttpResponse("An Exception Occurred! Employee cant be added")
40
41     def remove_emp(request, employee_id=None):
42         if employee_id:
43             try:
44                 delete_employee = Employee.objects.get(id=employee_id)
45                 delete_employee.delete()
46                 return HttpResponse("Employee Removed Successfully")
47             except:
48                 return HttpResponse("Please Enter Valid Employee Id")
49
50         employees = Employee.objects.all()
51         context = {
52             "employees": employees
53         }
54         return render(request, 'remove_emp.html', context)
55
56     def filter_emp(request):
57         return render(request, 'filter_emp.html')
58
```

Remove_emp.html

```
<body>
<div class="container">
    <h1>Remove An Employee</h1>
    <hr>
    <div class="dropdown">
        <button class="btn btn-primary dropdown-toggle" type="button" data-bs-toggle="dropdown" aria-expanded="false">
            Choose Employee To Be Removed
        </button>
        <ul class="dropdown-menu">
            {% for employee in employees %}
            <li><a class="dropdown-item" href="/remove_emp/{{employee.id}}"/>{{ employee.first_name}} {{ employee.last_name}}</a></li>
            {% endfor %}
        </ul>
    </div>
</div>
<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.0/dist/js/bootstrap.bundle.min.js" integrity="sha384-A3rJD856KowSb7dwlZdYEk039G<
</body>
```

Filter employees based on the fields specified.

Explain Q objects in Django ORM?

Q object `django.db.models.Q` is an object to encapsulate a collection of keyword arguments specified as **FIELD LOOKUPS**.

Q objects are used to write complex queries, as in `filter()` functions just "AND" the conditions while if you want to "OR" the conditions you can use Q objects.

Let's see an example:

```
from django.db import models
from django.db.models import Q

Models.objects.get(Q(question__startswith='When'), Q(answer__startswith='On') | Q(answer__startswith='At'))
```

[Q Objects can be combined with the help of the | and & operators to get a new Q Object]

This is equivalent to the following SQL WHERE Clause:

```
SELECT * FROM Model WHERE question LIKE 'When%' And (answer="On" OR answer="At%")
```

views.py file contains filter_emp function to handle this request. iconatains specifies search should be case insensitive.

```

def filter_emp(request):
    if request.method == "POST":
        name = request.POST['name']
        department = request.POST['department']
        role = request.POST['role']
        employees = Employee.objects.all()
        if name:
            employees = employees.filter(Q(first_name__icontains=name)|Q(last_name__icontains=
        if department:
            employees = employees.filter(department__name__icontains=department)
        if role:
            employees = employees.filter(role__name__icontains=role)
        context = {
            'employees':employees
        }
        return render(request,"view_all_emp.html",context)
    elif request.method == 'GET':
        return render(request,'filter_emp.html')
    else:
        return HttpResponseRedirect("An Exception Occurred")

```

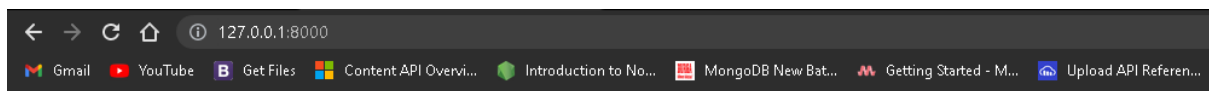
filter_emp.html

```

<body>
<div class="container">
    <h1>Filter Employee Details </h1>
    <hr>
    <form action="/filter_emp" method="post">
        {% csrf_token %}
        <label for="name">Employee First/Last Name</label><br>
        <input type="text" id="name" name="name" value="{{ name }}"><br>
        <label for="department">Department</label><br>
        <input type="text" id="department" name="department" value="{{ department }}"><br>
        <label for="role">Role</label><br>
        <input type="text" id="role" name="role" value="{{ role }}"><br>
        <hr>
        <button type="submit" class="btn btn-primary">Submit</button>
    </form>
</div>

```

Our final project's frontend looks as follows.



Welcome to Employee Mangement System

[View All Employee](#)[Add An Employee](#)[Remove An Employee](#)[Filter Employee Details](#)

Add an employee.

Add An Employee

First Name

Last Name

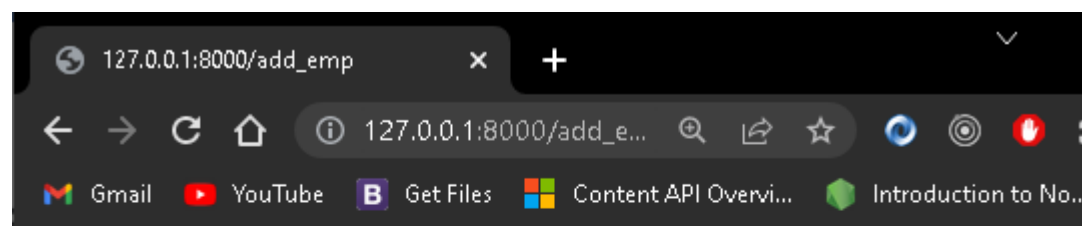
Salary

Department

Role

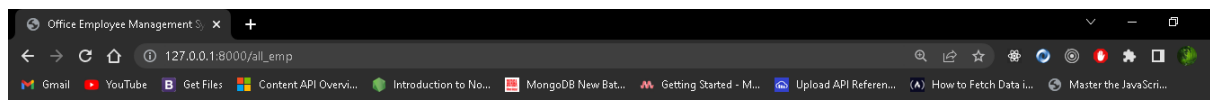
Bonus

Phone Number



Employee added Successfully

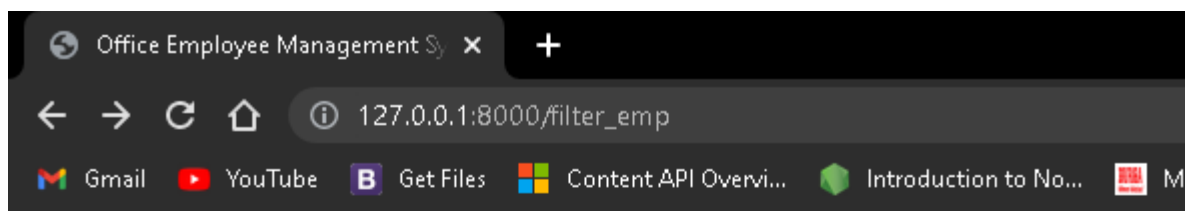
View all employees.



View All Employee

Id	First Name	Last Name	Salary	Bonus	Phone Number	Role	Department	Location	Hire Date
12	Swati	Patil	30000	2000	123456	Zonal Manager	Marketing	Mumbai	Aug. 14, 2022
13	Priti	M	30000	1000	12345678	Marketing Manager	Accounting	Hydrabaad	Aug. 14, 2022
14	Aditi	B	1000	1500	123456	Python Developer	IT	Banglore	Aug. 14, 2022

Filter Employee.



Filter Employee Details

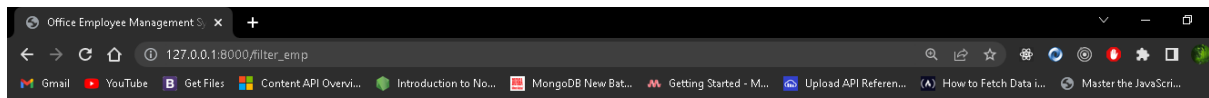
Employee First/Last Name

Department

Role

Submit

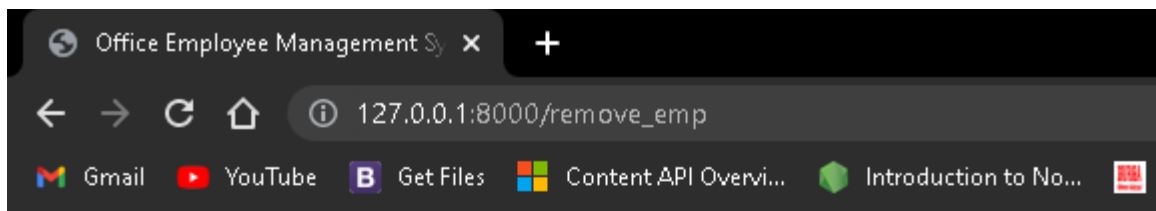
After filtration based on search name search criteria of employees



View All Employee

Id	First Name	Last Name	Salary	Bonus	Phone Number	Role	Department	Location	Hire Date
12	Swati	Patil	30000	2000	123456	Zonal Manager	Marketing	Mumbai	Aug. 14, 2022
14	Aditi	B	1000	1500	123456	Python Developer	IT	Banglore	Aug. 14, 2022

Remove an employee, select employee to be deleted.



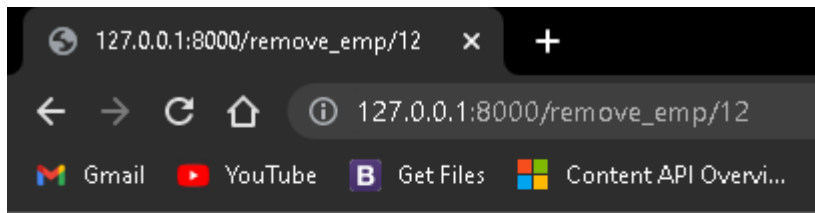
Remove An Employee

Choose Employee To Be Removed ▾

Swati Patil

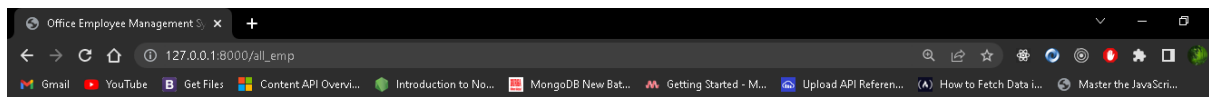
Priti M

Aditi B



Employee Removed Successfully

Employee with Id 12 is deleted, in previous API

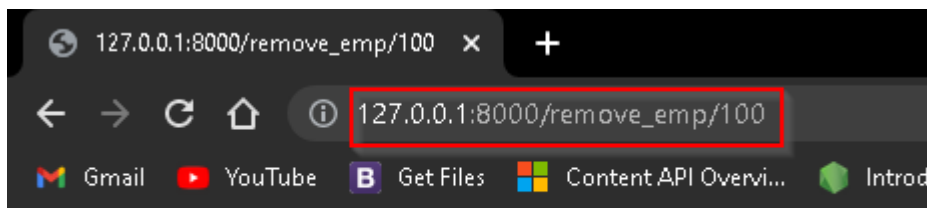


View All Employee

Id	First Name	Last Name	Salary	Bonus	Phone Number	Role	Department	Location	Hire Date
13	Priti	M	30000	1000	12345678	Marketing Manager	Accounting	Hydrabaad	Aug. 14, 2022
14	Aditi	B	1000	1500	123456	Python Developer	IT	Banglore	Aug. 14, 2022



If you give invalid Id to remove an employee, as Id 100 is not present in our database.



Please Enter Valid Employee Id