

1. Create project and application

```
>django-admin startproject CRUD

>cd CRUD

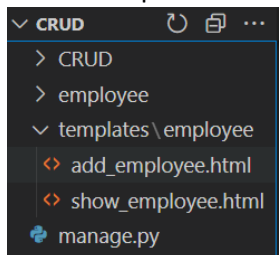
\CRUD>code .

\CRUD>python manage.py startapp employee
```

2. Register application in settings.py

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

3. Create templates folder and register it in settings.py



```
TEMPLATES = [
    {
        'BACKEND': 'django.template.backends.django.DjangoTemplates',
        'DIRS': [BASE_DIR/'templates'],
        'APP_DIRS': True,
```

4. Create database “crud” in workbench and register it in settings.py

```
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.mysql',
        'NAME': 'crud',
        'USER': 'root',
        'PASSWORD': 'root',
        'HOST': 'localhost',
        'PORT': '3306',
    }
}
```

5. Create model EmployeeTable in models.py and register it in admin.py

```
models.py ×

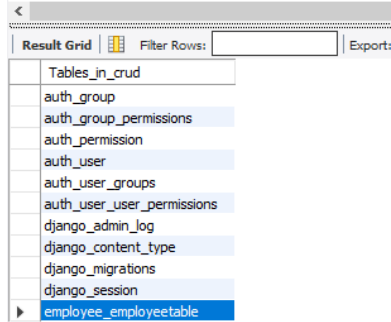
employee > models.py > EmployeeTable
1  from django.db import models
2
3  # Create your models here.
4  class EmployeeTable(models.Model):
5      id=models.AutoField(primary_key=True)
6      name=models.CharField(max_length=50)
7      salary=models.FloatField()
8      DESIGNATION = (('hr','HR'),
9                      ('manager','MANAGER'),
10                     ('trainer','TRAINER'))
11     designation=models.CharField(choices=DESIGNATION,max_length=30)
12

admin.py ×

employee > admin.py
1  from django.contrib import admin
2  from employee.models import EmployeeTable
3
4  # Register your models here.
5  admin.site.register(EmployeeTable)
6
```

6. Do makemigrations and migrate and check table in database

```
1 • create database CRUD;
2 • use CRUD;
3 • show tables;
4
```

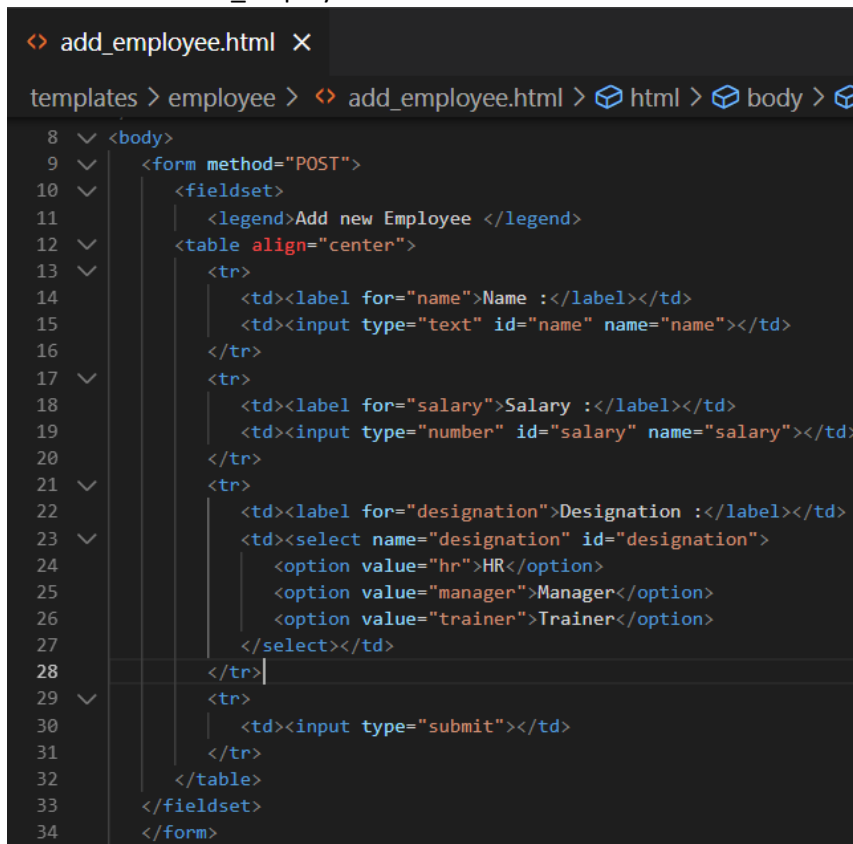


Result Grid | Filter Rows: | Export:

Tables_in_crud
auth_group
auth_group_permissions
auth_permission
auth_user
auth_user_groups
auth_user_user_permissions
django_admin_log
django_content_type
django_migrations
django_session
employee_employeetable

7. Create form and display

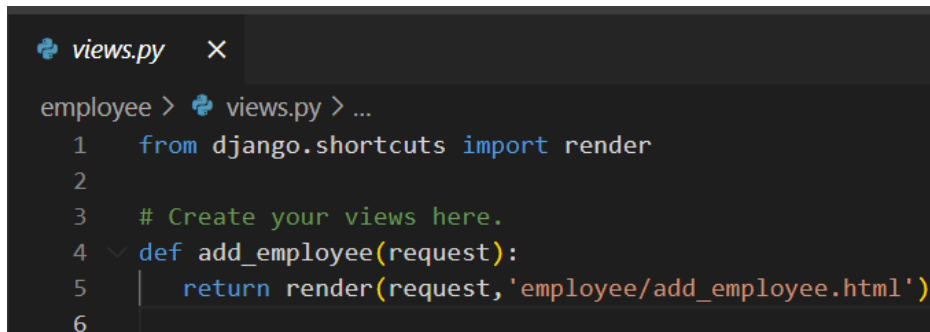
a. Create form in add_employee.html



```
<> add_employee.html X
templates > employee > <> add_employee.html > html > body >
8 <body>
9 <form method="POST">
10 <fieldset>
11 <legend>Add new Employee </legend>
12 <table align="center">
13 <tr>
14 <td><label for="name">Name :</label></td>
15 <td><input type="text" id="name" name="name"></td>
16 </tr>
17 <tr>
18 <td><label for="salary">Salary :</label></td>
19 <td><input type="number" id="salary" name="salary"></td>
20 </tr>
21 <tr>
22 <td><label for="designation">Designation :</label></td>
23 <td><select name="designation" id="designation">
24 <option value="hr">HR</option>
25 <option value="manager">Manager</option>
26 <option value="trainer">Trainer</option>
27 </select></td>
28 </tr>
29 <tr>
30 <td><input type="submit"></td>
31 </tr>
32 </table>
33 </fieldset>
34 </form>
```

{% csrf_token %} add after line 9

b. Create view to show above form



```
views.py X
employee > views.py > ...
1 from django.shortcuts import render
2
3 # Create your views here.
4 def add_employee(request):
5     return render(request, 'employee/add_employee.html')
6
```

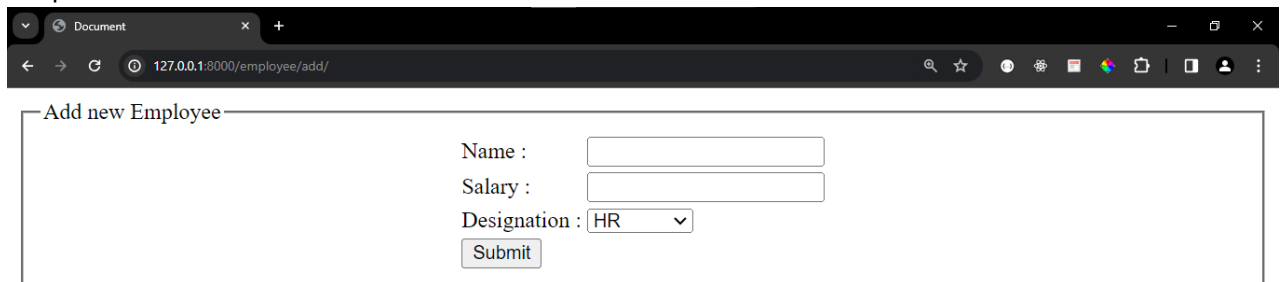
- c. Create employee_urls.py file (application level url)

```
employee_urls.py X
employee > employee_urls.py > ...
1  from django.urls import path
2  from employee import views
3
4  urlpatterns = [
5      path('add/', views.add_employee),
6  ]
```

- d. Register above employee_urls.py in urls.py

```
urls.py X
CRUD > urls.py > ...
15  """ 2. Add a URL to urlpatterns: path('blog/', include('bl
16  """
17  from django.contrib import admin
18  from django.urls import path,include
19
20  urlpatterns = [
21      path('admin/', admin.site.urls),
22      path('employee/', include('employee.employee_urls')),
23  ]
```

- e. Output :



The screenshot shows a web browser window with the address bar displaying '127.0.0.1:8000/employee/add/'. The page content is a form titled 'Add new Employee'. The form contains three input fields: 'Name :', 'Salary :', and 'Designation :'. The 'Designation' field is a dropdown menu with 'HR' selected. Below the fields is a 'Submit' button.

8. Collect data from form and add it to the table and after inserting , redirect to show that data
- a. Create fbv(function based view) to insert and get employee data (views.py)

```
views.py X

employee > views.py > ...
1  from django.shortcuts import render,redirect
2  from employee.models import EmployeeTable
3  # Create your views here.
4  def show_employee(request):
5      data={}
6      all_employee=EmployeeTable.objects.all()
7      data['employees']=all_employee
8      return render(request,'employee/show_employee.html',context=data)
9
10
11 def add_employee(request):
12     if request.method=='POST':
13         emp_name=request.POST['name']
14         emp_salary=request.POST['salary']
15         emp_designation=request.POST['designation']
16
17         employee=EmployeeTable.objects.create(name=emp_name,salary=emp_salary,designation=emp_designation)
18
19         employee.save()
20
21         return redirect('/employee/show')
22     return render(request,'employee/add_employee.html')
23
```

- b. Code for show_employee.html to show employees

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <center>
        <h1>Employee Data</h1>
        <button>Add new Employee</button>
    </center>
    <br>
    <table border="1" align="center", cellpadding="10" cellspacing="0">
        <tr>
            <th>ID</th>
            <th>NAME</th>
            <th>SALARY</th>
            <th>DESIGNATION</th>
            <th colspan="2">ACTION</th>
        </tr>
        {% for employee in employees %}
        <tr>
            <td>{{employee.id}}</td>
            <td>{{employee.name}}</td>
            <td>{{employee.salary}}</td>
            <td>{{employee.designation}}</td>
            <td><button>Update</button></td>
            <td><button>Delete</button></td>
        </tr>
        {% endfor %}
    </table>
</body>
</html>
```

- c. Create URL in employee_urls.py

```
employee_urls.py X
employee > employee_urls.py > ...
1 from django.urls import path
2 from employee import views
3
4 urlpatterns = []
5     path('add/', views.add_employee),
6     path('show/', views.show_employee),
7 
```

- d. OUTPUT :

The screenshot shows a web browser at 127.0.0.1:8000/employee/add/. The form has fields for Name (prasad), Salary (25000), and Designation (Manager), with a Submit button. Below the form, a SQL query is shown: `select * from employee_employeetable;`. The result is a table with one row: id 1, name prasad, salary 25000, designation manager.

id	name	salary	designation
1	prasad	25000	manager

Employee Data

Add new Employee

ID	NAME	SALARY	DESIGNATION	ACTION	
1	prasad	25000.0	manager	<button>Update</button>	<button>Delete</button>

9. Provide link on add new employee button to open add_employee.html

```
show_employee.html X
templates > employee > show_employee.html > html > body > center > button
9 <center>
10 <h1>Employee Data</h1>
11 <button><a href="/employee/add">Add new Employee</a></button>
12 </center>
```

10. Provide button in add_employee.html to show all employees

```
add_employee.html X
templates > employee > add_employee.html
11 <fieldset>
12 <legend>Add new Employee
13 <button><a href="/employee/show">Show all employees</a></button>
14 </legend>
15 <table align="center">
```

11. Delete functionality

- Create fbv and write logic to delete

```
views.py x
employee > views.py > ...
24 def delete_employee(request,empid):
25     employee=EmployeeTable.objects.get(id=empid)
26     employee.delete()
27     return redirect('/employee/show')
28
```

- Create URL for view

```
employee_urls.py x
employee > employee_urls.py > ...
1 from django.urls import path
2 from employee import views
3
4 urlpatterns = [
5     path('add/', views.add_employee),
6     path('show/', views.show_employee),
7     path('delete/<empid>', views.delete_employee),
8 ]
```

- Use above url on delete button in show_employees.html

```
show_employee.html x
templates > employee > show_employee.html
28 <td><button>Update</button></td>
29 <td><button><a href="/employee/delete/{employee.id}">Delete</a></button></td>
30 </tr>
```

12. Update Logic

- Create view

```
def update_employee(request,empid):
    data={}
    fetched_employee=EmployeeTable.objects.get(id=empid)
    print(fetched_employee.name)
    data['employee']=fetched_employee
    if request.method=='POST':
        emp_name=request.POST['name']
        emp_salary=request.POST['salary']
        emp_designation=request.POST['designation']
        employee=EmployeeTable.objects.filter(pk=empid)
        employee.update(name=emp_name,salary=emp_salary,designation=emp_designation)
        return redirect('/employee/show')
    return render(request,'employee/update_employee.html',context=data)
```

- b. Create update_employee.html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Document</title>
</head>
<body>
  <form method="POST">
    {% csrf_token %}
    <fieldset>
      <legend>Update Employee</legend>
      <table align="center">
        <tr>
          <td><label for="name">Name :</label></td>
          <td><input type="text" id="name" name="name" value="{{employee.name}}"></td>
        </tr>
        <tr>
          <td><label for="salary">Salary :</label></td>
          <td><input type="number" id="salary" name="salary" value="{{employee.salary}}"></td>
        </tr>
        <tr>
          <td><label for="designation">Designation :</label></td>
          <td><select name="designation" id="designation">
            <option value="hr">HR</option>
            <option value="manager">Manager</option>
            <option value="trainer">Trainer</option>
          </select></td>
        </tr>
        <tr>
          <td colspan="2"><input type="submit"></td>
        </tr>
      </table>
    </fieldset>
  </form>
</body>
</html>
```

- c. Create application level url

```
urlpatterns = [
    path('add/', views.add_employee),
    path('show/', views.show_employee),
    path('delete/<empid>', views.delete_employee),
    path('update/<empid>', views.update_employee),
]
```

- d. Use above url on button

```
<td>{{employee.designation}}</td>
<td><button><a href="/employee/update/{{employee.id}}">Update</a></button></td>
<td><button><a href="/employee/delete/{{employee.id}}">Delete</a></button></td>
</tr>
```

- e.

13. DONE !!!!!!!!!!!!!!!!!!!!!!!

Done