**FUNCTION BASED VIEW CRUD EXAMPLE**

**MIGRATION**

PS C:\Users\kamal\django\_projects\fbvEmployees> py -m manage makemigrations

Migrations for 'fbvEmpApp':

fbvEmpApp\migrations\0001\_initial.py

- Create model Employee

PS C:\Users\kamal\django\_projects\fbvEmployees> py -m manage migrate

Operations to perform:

Apply all migrations: admin, auth, contenttypes, fbvEmpApp, sessions

Running migrations:

Applying contenttypes.0001\_initial... OK

Applying auth.0001\_initial... OK

Applying admin.0001\_initial... OK

Applying admin.0002\_logentry\_remove\_auto\_add... OK

Applying admin.0003\_logentry\_add\_action\_flag\_choices... OK

Applying contenttypes.0002\_remove\_content\_type\_name... OK

Applying auth.0002\_alter\_permission\_name\_max\_length... OK

Applying auth.0003\_alter\_user\_email\_max\_length... OK

Applying auth.0004\_alter\_user\_username\_opts... OK

Applying auth.0005\_alter\_user\_last\_login\_null... OK

Applying auth.0006\_require\_contenttypes\_0002... OK

Applying auth.0007\_alter\_validators\_add\_error\_messages... OK

Applying auth.0008\_alter\_user\_username\_max\_length... OK

Applying auth.0009\_alter\_user\_last\_name\_max\_length... OK

Applying auth.0010\_alter\_group\_name\_max\_length... OK

Applying auth.0011\_update\_proxy\_permissions... OK

Applying auth.0012\_alter\_user\_first\_name\_max\_length... OK

Applying fbvEmpApp.0001\_initial... OK

Applying sessions.0001\_initial... OK

PS C:\Users\kamal\django\_projects\fbvEmployees>

**models.py**

from django.db import models  
  
*# Create your models here.*class Employee(models.Model):  
 DEPARTMENT=[('Developer','DEVELOPER'),('Tester','TESTER'),('DBA','DBA'),('DevOps','DEVOPS'),('Cloud','Cloud'),  
 ('Security','SECURITY'),('Networking','NETWORKING'),('Support','SUPPORT')]  
 GENDER=[('Male','MALE'),('Female','FEMALE'),('Others','OTHERS')]  
 firstName=models.CharField(max\_length=20)  
 lastName=models.CharField(max\_length=20)  
 age=models.IntegerField(default=24)  
 gender = models.CharField(max\_length=10, choices=GENDER)  
 mobile=models.CharField(max\_length=25)  
 email=models.EmailField(max\_length=50)  
 department=models.CharField(max\_length=15,choices=DEPARTMENT)  
 salary=models.FloatField()  
 dateOfJoining=models.DateField(default='2024/01/01')

**forms.py**

from django import forms  
from fbvEmpApp.models import Employee  
  
  
class EmployeeForm(forms.ModelForm):  
 class Meta:  
 model=Employee  
 fields='\_\_all\_\_'

**views.py**

from django.shortcuts import render, redirect  
from fbvEmpApp.models import Employee  
from fbvEmpApp.forms import EmployeeForm  
  
  
*# Create your views here.*def getEmployees(request):  
 employeesList=Employee.objects.all()  
 print(employeesList.count())  
 return render(request, 'fbvEmpApp/index.html',{'employees':employeesList})  
  
def createEmployees(request):  
 form=EmployeeForm()  
 if request.method == 'POST':  
 form=EmployeeForm(request.POST)  
 if form.is\_valid():  
 form.save()  
 return redirect('/')  
 return render(request, 'fbvEmpApp/create.html', {'form':form})  
  
  
def deleteEmployees(request, id):  
 employee=Employee.objects.get(id=id)  
 employee.delete()  
 return redirect('/')  
  
""""  
def updateEmployees(request, id):  
 upemployee=Employee.objects.get(id=id)  
 if request.method == 'POST':  
 form=EmployeeForm(request.POST, instance=upemployee)  
 if form.is\_valid():  
 form.save()  
 return redirect('/')  
 return render(request, 'fbvEmpApp/update.html', {'employee':upemployee})  
"""  
def updateEmployees(request, id):  
 upemployee=Employee.objects.get(id=id)  
 form = EmployeeForm(instance=upemployee)  
 if request.method == 'POST':  
 form=EmployeeForm(request.POST, instance=upemployee)  
 if form.is\_valid():  
 form.save()  
 return redirect('/')  
 return render(request, 'fbvEmpApp/update.html', {'form':form})

**settings.py**

*"""  
Django settings for fbvEmployees project.  
  
Generated by 'django-admin startproject' using Django 5.0.4.  
  
For more information on this file, see  
https://docs.djangoproject.com/en/5.0/topics/settings/  
  
For the full list of settings and their values, see  
https://docs.djangoproject.com/en/5.0/ref/settings/  
"""*from pathlib import Path  
  
*# Build paths inside the project like this: BASE\_DIR / 'subdir'.*BASE\_DIR = Path(\_\_file\_\_).resolve().parent.parent  
TEMPLATES\_DIR = Path(BASE\_DIR/'templates')  
STATIC\_DIR = Path(BASE\_DIR/'static')  
  
*# Quick-start development settings - unsuitable for production  
# See https://docs.djangoproject.com/en/5.0/howto/deployment/checklist/  
  
# SECURITY WARNING: keep the secret key used in production secret!*SECRET\_KEY = 'django-insecure-@&y^2b58nk!a4ks5v%^(8c6pl3wk=du\*c811!r=xwbe7\_c589m'  
  
*# SECURITY WARNING: don't run with debug turned on in production!*DEBUG = True  
  
ALLOWED\_HOSTS = []  
  
  
*# Application definition*INSTALLED\_APPS = [  
 'django.contrib.admin',  
 'django.contrib.auth',  
 'django.contrib.contenttypes',  
 'django.contrib.sessions',  
 'django.contrib.messages',  
 'django.contrib.staticfiles',  
 'fbvEmpApp'  
]  
  
MIDDLEWARE = [  
 'django.middleware.security.SecurityMiddleware',  
 'django.contrib.sessions.middleware.SessionMiddleware',  
 'django.middleware.common.CommonMiddleware',  
 'django.middleware.csrf.CsrfViewMiddleware',  
 'django.contrib.auth.middleware.AuthenticationMiddleware',  
 'django.contrib.messages.middleware.MessageMiddleware',  
 'django.middleware.clickjacking.XFrameOptionsMiddleware',  
]  
  
ROOT\_URLCONF = 'fbvEmployees.urls'  
  
TEMPLATES = [  
 {  
 'BACKEND': 'django.template.backends.django.DjangoTemplates',  
 'DIRS': [TEMPLATES\_DIR,],  
 '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',  
 ],  
 },  
 },  
]  
  
WSGI\_APPLICATION = 'fbvEmployees.wsgi.application'  
  
  
*# Database  
# https://docs.djangoproject.com/en/5.0/ref/settings/#databases*DATABASES = {  
 'default': {  
 'ENGINE': 'django.db.backends.mysql',  
 'NAME': 'employeeDb',  
 'USER': 'root',  
 'PASSWORD': 'Jerrick2911',  
 'HOST': 'localhost',  
 'PORT': '3306'  
 }  
}  
  
  
*# Password validation  
# https://docs.djangoproject.com/en/5.0/ref/settings/#auth-password-validators*AUTH\_PASSWORD\_VALIDATORS = [  
 {  
 'NAME': 'django.contrib.auth.password\_validation.UserAttributeSimilarityValidator',  
 },  
 {  
 'NAME': 'django.contrib.auth.password\_validation.MinimumLengthValidator',  
 },  
 {  
 'NAME': 'django.contrib.auth.password\_validation.CommonPasswordValidator',  
 },  
 {  
 'NAME': 'django.contrib.auth.password\_validation.NumericPasswordValidator',  
 },  
]  
  
  
*# Internationalization  
# https://docs.djangoproject.com/en/5.0/topics/i18n/*LANGUAGE\_CODE = 'en-us'  
  
TIME\_ZONE = 'UTC'  
  
USE\_I18N = True  
  
USE\_TZ = True  
  
  
*# Static files (CSS, JavaScript, Images)  
# https://docs.djangoproject.com/en/5.0/howto/static-files/*STATIC\_URL = 'static/'  
STATICFILES\_DIRS=[  
 STATIC\_DIR,  
]  
  
  
*# Default primary key field type  
# https://docs.djangoproject.com/en/5.0/ref/settings/#default-auto-field*DEFAULT\_AUTO\_FIELD = 'django.db.models.BigAutoField'

**index.html**

<!DOCTYPE html>  
{% load static %}  
<html lang="en">  
<head>  
 <meta charset="UTF-8">  
 <title>Employee</title>  
 <link rel="stylesheet" href="{% static 'css/index.css' %}" />  
</head>  
<body>  
 <div class="myDiv">  
 <h1> EMPLOYEES INFORMATION</h1>  
 <div>  
 <img src="{%static 'images/image2.jpg'%}" />  
 </div>  
 </div>  
 <div class="mycontainer">  
 {% if employees %}  
 <table>  
 <thead>  
 <tr>  
 <th>Emp Id</th>  
 <th>First Name</th>  
 <th>Last Name</th>  
 <th>Age</th>  
 <th>Gender</th>  
 <th>Mobile</th>  
 <th>Email</th>  
 <th>Department</th>  
 <th>Salary</th>  
 <th>Date of Joining</th>  
 <th>Options</th>  
 <th>Options</th>  
 </tr>  
 </thead>  
 {% for emp in employees %}  
 <tr>  
 <td>{{emp.id}}</td>  
 <td>{{emp.firstName}}</td>  
 <td>{{emp.lastName}}</td>  
 <td>{{emp.age}}</td>  
 <td>{{emp.gender}}</td>  
 <td>{{emp.mobile}}</td>  
 <td>{{emp.email}}</td>  
 <td>{{emp.department}}</td>  
 <td>{{emp.salary}}</td>  
 <td>{{emp.dateOfJoining}}</td>  
 <td>  
 <button type="button" class="button button1" onclick="window.location.href='/delete/{{emp.id}}';">  
 DELETE  
 </button>  
 </td>  
 <td>  
 <button type="button" class="button button1" onclick="window.location.href='/update/{{emp.id}}';">  
 UPDATE  
 </button>  
 </td>  
 </tr>  
 {% endfor %}  
 </table>  
 {% else %}  
 <p>  
 Employee details not available... Shortly updated....  
 {% endif %}  
 </div>  
 <div class="myDiv">  
 <button type="button" class="button button1" onclick="window.location.href='/create';">  
 ADD EMPLOYEE  
 </button>  
 </div>  
</body>  
</html>

**create.html**

<!DOCTYPE html>  
{% load static %}  
<html lang="en">  
<head>  
 <meta charset="UTF-8">  
 <title>Employee</title>  
 <link rel="stylesheet" href="{% static 'css/index.css' %}" />  
</head>  
<body>  
 <div class="myDiv">  
 <h1> EMPLOYEES </h1>  
 <div>  
 <img src="{%static 'images/image2.jpg'%}" />  
 </div>  
 </div>  
 <div class="myDiv">  
 <h2> CREATE EMPOYEES </h2>  
 </div>  
 <div class="mycontainer">  
 <form method="post">  
 <table>  
 {{form.as\_table}}  
 {% csrf\_token %}  
 </table>  
 <br>  
 <input type="submit" value="Add Employee">  
 <br> <br>  
 </form>  
 </div>  
</body>  
</html>

Output (Adding Records)

A blue and green rectangle with black lines

Description automatically generated

A screenshot of a computer

Description automatically generated

**Update (Method 1) – update.html**

<!DOCTYPE html>  
{% load static %}  
<html lang="en">  
<head>  
 <meta charset="UTF-8">  
 <title>Employee</title>  
 <link rel="stylesheet" href="{% static 'css/index.css' %}" />  
</head>  
<body>  
 <div class="myDiv">  
 <h1> EMPLOYEES </h1>  
 <div>  
 <img src="{%static 'images/image2.jpg'%}" />  
 </div>  
 </div>  
 <div class="myDiv">  
 <h2> UPDATE EMPOYEES </h2>  
 </div>  
 <div class="mycontainer">  
 <form method="post">  
 {% csrf\_token %}  
 Employee First Name: <input type="text" name="firstName" value="{{employee.firstName}}">  
 <br>  
 Employee Last Name: <input type="text" name="lastName" value="{{employee.lastName}}">  
 <br>  
 Employee Age: <input type="text" name="age" value="{{employee.age}}">  
 <br>  
 Employee Gender: <input type="text" name="gender" value="{{employee.gender}}">  
 <br>  
 Employee Mobile: <input type="text" name="mobile" value="{{employee.mobile}}">  
 <br>  
 Employee Email: <input type="email" name="email" value="{{employee.email}}">  
 <br>  
 Employee Department: <input type="text" name="department" value="{{employee.department}}">  
 <br>  
 Employee Salary: <input type="number" name="salary" value="{{employee.salary}}">  
 <br>  
 Employee Date of Joining: <input type="text" name="dateOfJoining" value="{{employee.dateOfJoining}}" pattern="\d{4}-\d{2}-\d{2}">  
 <br>  
 <br>  
 <input type="submit" value="Update Employee">  
 <br> <br>  
 </form>  
 </div>  
</body>  
</html>

**views.py**

def updateEmployees(request, id):  
 upemployee=Employee.objects.get(id=id)  
 if request.method == 'POST':  
 form=EmployeeForm(request.POST, instance=upemployee)  
 if form.is\_valid():  
 form.save()  
 return redirect('/')  
 return render(request, 'fbvEmpApp/update.html', {'employee':upemployee})

**Output**

A screenshot of a computer

Description automatically generated

Change the date of joining as 01-07-2023

A screenshot of a computer

Description automatically generated

Now the record is

A screenshot of a computer

Description automatically generated

**Update (Method 2) – update.html**

<!DOCTYPE html>  
{% load static %}  
<html lang="en">  
<head>  
 <meta charset="UTF-8">  
 <title>Employee</title>  
 <link rel="stylesheet" href="{% static 'css/index.css' %}" />  
</head>  
<body>  
 <div class="myDiv">  
 <h1> EMPLOYEES </h1>  
 <div>  
 <img src="{%static 'images/image2.jpg'%}" />  
 </div>  
 </div>  
 <div class="myDiv">  
 <h2> UPDATE EMPOYEES </h2>  
 </div>  
 <div class="mycontainer">  
 <form method="post">  
 {% csrf\_token %}  
 <table>  
 {{form.as\_table}}  
 </table>  
 <br>  
 <input type="submit" value="Update Employee">  
 <br> <br>  
 </form>  
 </div>  
</body>  
</html>

views.py

def updateEmployees(request, id):  
 upemployee=Employee.objects.get(id=id)  
 form = EmployeeForm(instance=upemployee)  
 if request.method == 'POST':  
 form=EmployeeForm(request.POST, instance=upemployee)  
 if form.is\_valid():  
 form.save()  
 return redirect('/')  
 return render(request, 'fbvEmpApp/update.html', {'form':form})

Output:

A screenshot of a computer

Description automatically generated

Changed the Last Name and Department

A screenshot of a computer

Description automatically generated