**POSTAL CODE Management System**

**(Django MySQL):**

(NOTE: YOU SHOULD HAVE MYSQL AND PYTCHARM OR ANY OTHER PYTHON IDS INSTALLED IN YOUR COMPUTER)

Step 1: Virtual Environment:-

The first step to start is to setup a virtual environment in the pycharm community. It is suggested to have a dedicated virtual environment for each Django project, and one way to manage a virtual environment is [venv,](https://docs.python.org/3/tutorial/venv.html) which is included in Python.

The name of the virtual envirment is select by the user but in this tutorial we will call it Django\_project

Type the following in the command prompt, remember to navigate to where you want to create your virtual enviroment:

Windows:

py -m venv django\_project

This will set up a virtual environment, and create a folder named "myworld" with subfolders and files, like this:

Django\_project

Include

Lib

Scripts

Puvenc.cfg

Then you have to activate the environment, by typing this command:

|  |
| --- |
| Windows: |
| django\_project\Scripts\activate.bat |

Once the environment is activated, you will see this result in the command prompt:

|  |
| --- |
| Windows: |
| (django\_project) C:\Desktop\SSGC\_Internship\django\_project> |

Step 2: Install Django

Now, that we have created a virtual environment, we are ready to install Django. We have to install Django with in the virtual environment so remember to check that you are using the Django installing command inside it

Django is installed using pip, with this command:

|  |
| --- |
| Windows: |
| (base) C:\Desktop\SSGC Internship\django\_project>pip install django |

Step 3: My First Project

Once you have come up with a suitable name for your Django project, like

I have name my Django project same as the virtual envirment django\_project, navigate to where in the file system you want to store the code (in the virtual environment), I will navigate to the Django\_project folder, and run this command in the command prompt:

django-admin startproject django\_project

Django creates a django\_project folder on my computer, with this content:

django\_project manage.py django\_project/ \_\_init\_\_.py asgi.py settings.py urls.py wsgi.py

In order to run the project, open Python IDE(Pycharm in my case) and navigate to this project.

After the folder is open, open the terminal and enter the following command to run the project.

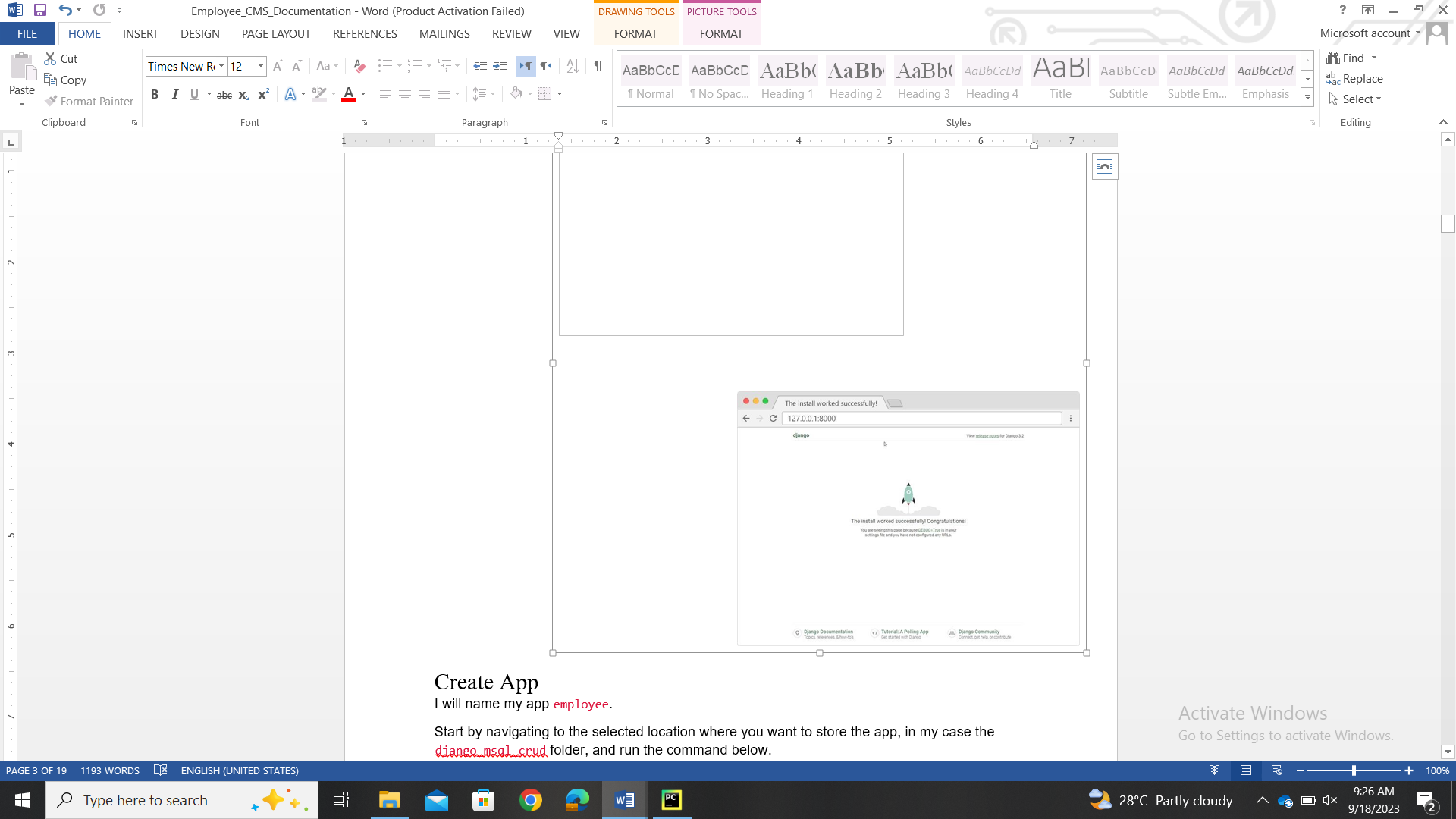
python manage.py makemigrations

python manage.py migrate

python manage.py runserver

Open a new browser window and type [127.0.0.1:8000](http://127.0.0.1:8000/) in the address bar.

The result:



Step 4: Create App:

Now, after creating the project and testing that the project is working them we will niw create the app inside the project.

In this tutorial we will be using app name as postingapp

Now we have to first stop the running of the project by [ctrl][c] and then run the command given below:

python manage.py startapp postingapp

After executing the above command we will now see the following files generated in the project directory.

Django\_project

Manage.py

Django\_project/

Postingapp/

Migrations/

\_\_init\_\_.py

Admin.py

Models.py

Apps.py

Tests.py

Views.py

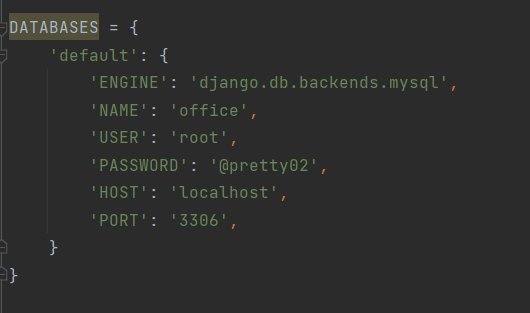
Step 5: Database Setup:

Create a postingapp database in mysql and configure into the **settings.py** file of django project and modify the ENGINE from sqlite3 to mysql. Before configuring thedatabase in the file we first have to run the following commands in the terminal.

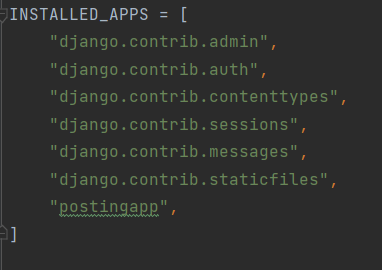
pip install mysqlclient

NOTE: When I was first making the django app and configuring the database as I was using SQL Server Management System (SSMS) so I started to configure SSMS and I tried a lot to configure it because I don’t want to use mysql but after trying many times and surfing the internet I get to know that SSMS is not supported in django so make sure to use Mysql or others database.

Now write the following code in **settings.py** file



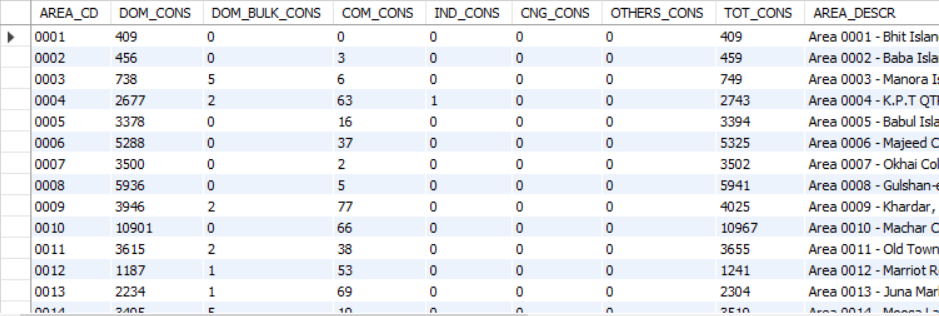
Also add your own app name in the installed apps. Here my app is postingapp so I have added this name.



Also add this set of code in the settings.py file



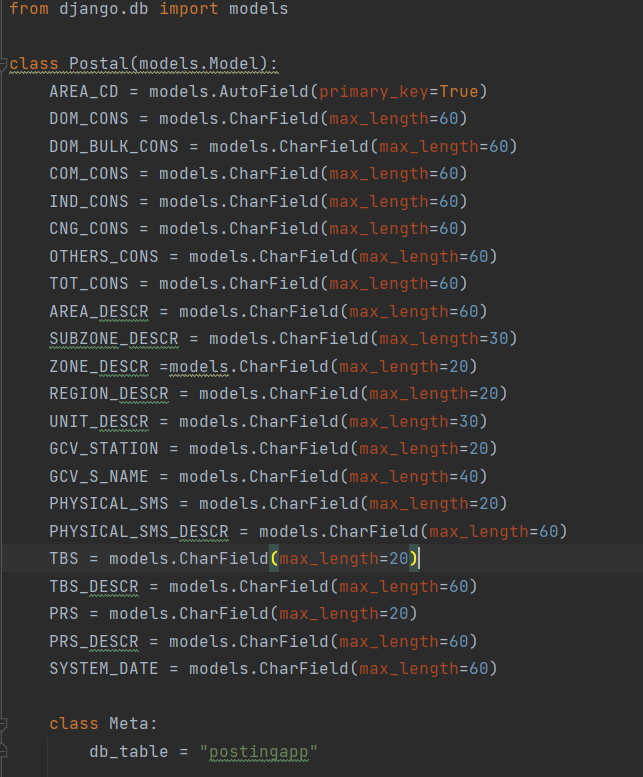
**postingapp table demonstration**



Step 6: Create a Model:

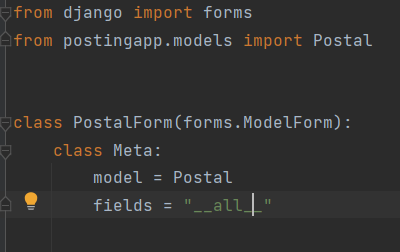
Now we will create a model of the data that is present in the database basically we are defining the database table with the various fields.

Put the following code into **models.py** file.



Step 7: Create a ModelForm:

Here we will create a new file inside the app as **forms.py** and put following code into the file.



Step 8: Apply CRUD operations involving Views, Templates and URLs:

Views:

Views are a fundamental part of the Django web framework, and they play a crucial role in handling user interactions with your web application. Views are responsible for processing data, interacting with the database and rendering templates to generate HTML responses.

Templates:

Django provides a convenient way to generate dynamic HTML pages by using its template system. A template consists of static paths of the desired HTML output as well as some special syntax describing how dynamin content will be inserted

URLs:

Every page on the Internet needs its own URL. This way your application knows what it should show to a user who opens that URL. In Django, we use something called URLconf (URL configuration). URLconf is a set of patterns that Django will try to match the requested URL to find the correct view.

**INSERT OPERATION:**

Views.py

# Create Post  
def add(request):  
 template = loader.get\_template('create.html')  
 return HttpResponse(template.render({}, request))  
  
  
# Store Blog  
def addrecord(request):  
 a = request.POST['AREA\_CD']  
 b = request.POST['DOM\_CONS']  
 c = request.POST['DOM\_BULK\_CONS']  
 d = request.POST['COM\_CONS']  
 e = request.POST['IND\_CONS']  
 f = request.POST['CNG\_CONS']  
 g = request.POST['OTHERS\_CONS']  
 h = request.POST['TOT\_CONS']  
 i = request.POST['AREA\_DESCR']  
 j = request.POST['SUBZONE\_DESCR']  
 k = request.POST['ZONE\_DESCR']  
 l = request.POST['REGION\_DESCR']  
 m = request.POST['UNIT\_DESCR']  
 n = request.POST['GCV\_STATION']  
 o = request.POST['GCV\_S\_NAME']  
 p = request.POST['PHYSICAL\_SMS']  
 q = request.POST['PHYSICAL\_SMS\_DESCR']  
 r = request.POST['TBS']  
 s = request.POST['TBS\_DESCR']  
 t = request.POST['PRS']  
 u = request.POST['PRS\_DESCR']  
 v = request.POST['SYSTEM\_DATE']  
 postingapp = Postal(AREA\_CD=a, DOM\_CONS=b, DOM\_BULK\_CONS=c, COM\_CONS=d, IND\_CONS=e, CNG\_CONS=f, OTHERS\_CONS=g,  
 TOT\_CONS=h,  
 AREA\_DESCR=i, SUBZONE\_DESCR=j, ZONE\_DESCR=k, REGION\_DESCR=l, UNIT\_DESCR=m, GCV\_STATION=n,  
 GCV\_S\_NAME=o, PHYSICAL\_SMS=p,  
 PHYSICAL\_SMS\_DESCR=q, TBS=r, TBS\_DESCR=s, PRS=t, PRS\_DESCR=u, SYSTEM\_DATE=v)  
 postingapp.save()  
 return HttpResponseRedirect(reverse('index'))

Create a folder as templates and create the file as **create.html** in templates.Write the following code in **create.html** file.

<!DOCTYPE html>  
<html>  
<head>  
 <meta charset="utf-8">  
 <meta name="viewport" content="width=device-width, initial-scale=1">  
 <title></title>  
 <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap@4.6.1/dist/css/bootstrap.min.css">  
 <script src="https://cdn.jsdelivr.net/npm/jquery@3.6.0/dist/jquery.slim.min.js"></script>  
 <script src="https://cdn.jsdelivr.net/npm/popper.js@1.16.1/dist/umd/popper.min.js"></script>  
 <script src="https://cdn.jsdelivr.net/npm/bootstrap@4.6.1/dist/js/bootstrap.bundle.min.js"></script>  
</head>  
<body>  
 <div class="container mt-5">  
 <div class="row d-flex justify-content-center">  
 <div class="col-md-6">  
 <div class="card">  
 <div class="card-header">  
 <h4>Create Employee</h4>  
 </div>  
 <div class="card-body">  
 <form method="post" action="addrecord/" enctype="multipart/form-data">  
 {% csrf\_token %}  
 <div class="row">  
 <div class="col-md-12">  
 <div class="form-group">  
 <label for="AREA\_CD"> AREA CODE</label>  
 <input type="text" class="form-control" name="AREA\_CD" id="AREA\_CD" >  
 </div>  
 </div>  
 </div>  
 <div class="row">  
 <div class="col-md-12">  
 <div class="form-group">  
 <label for="DOM\_CONS">DOM\_CONS</label>  
 <textarea class="form-control" name="DOM\_CONS" rows="3" id="DOM\_CONS" required></textarea>  
 </div>  
 </div>  
 </div>  
 <div class="row">  
 <div class="col-md-12">  
 <div class="form-group">  
 <label for="DOM\_BULK\_CONS">DOM\_BULK\_CONS</label>  
 <textarea class="form-control" name="DOM\_BULK\_CONS" rows="3" id="DOM\_BULK\_CONS" ></textarea>  
 </div>  
 </div>  
 </div>  
 <div class="row">  
 <div class="col-md-12">  
 <div class="form-group">  
 <label for="COM\_CONS">COM\_CONS</label>  
 <textarea class="form-control" name="COM\_CONS" rows="3" id="COM\_CONS" ></textarea>  
 </div>  
 </div>  
 </div>  
  
 <div class="row">  
 <div class="col-md-12">  
 <div class="form-group">  
 <label for="IND\_CONS">IND\_CONS</label>  
 <textarea class="form-control" name="IND\_CONS" rows="3" id="IND\_CONS" ></textarea>  
 </div>  
 </div>  
 </div>  
 <div class="row">  
 <div class="col-md-12">  
 <div class="form-group">  
 <label for="CNG\_CONS">CNG\_CONS</label>  
 <textarea class="form-control" name="CNG\_CONS" rows="3" id="CNG\_CONS" ></textarea>  
 </div>  
 </div>  
 </div>  
  
 <div class="row">  
 <div class="col-md-12">  
 <div class="form-group">  
 <label for="OTHERS\_CONS">OTHERS\_CONS</label>  
 <textarea class="form-control" name="OTHERS\_CONS" rows="3" id="OTHERS\_CONS" ></textarea>  
 </div>  
 </div>  
 </div>  
 <div class="row">  
 <div class="col-md-12">  
 <div class="form-group">  
 <label for="TOT\_CONS">TOT\_CONS</label>  
 <textarea class="form-control" name="TOT\_CONS" rows="3" id="TOT\_CONS" ></textarea>  
 </div>  
 </div>  
 </div>  
 <div class="row">  
 <div class="col-md-12">  
 <div class="form-group">  
 <label for="AREA\_DESCR">AREA DESCRIPTION</label>  
 <textarea class="form-control" name="AREA\_DESCR" rows="3" id="AREA\_DESCR" ></textarea>  
 </div>  
 </div>  
 </div>  
 <div class="row">  
 <div class="col-md-12">  
 <div class="form-group">  
 <label for="SUBZONE\_DESCR">SUBZONE DESCRIPTION</label>  
 <textarea class="form-control" name="SUBZONE\_DESCR" rows="3" id="SUBZONE\_DESCR" ></textarea>  
 </div>  
 </div>  
 </div>  
 <div class="row">  
 <div class="col-md-12">  
 <div class="form-group">  
 <label for="ZONE\_DESCR">ZONE DESCRIPTION</label>  
 <textarea class="form-control" name="ZONE\_DESCR" rows="3" id="ZONE\_DESCR" ></textarea>  
 </div>  
 </div>  
 </div>  
 <div class="row">  
 <div class="col-md-12">  
 <div class="form-group">  
 <label for="REGION\_DESCR">REGION DESCRIPTION</label>  
 <textarea class="form-control" name="REGION\_DESCR" rows="3" id="REGION\_DESCR" ></textarea>  
 </div>  
 </div>  
 </div>  
 <div class="row">  
 <div class="col-md-12">  
 <div class="form-group">  
 <label for="UNIT\_DESCR">UNIT DESCRIPTION</label>  
 <textarea class="form-control" name="UNIT\_DESCR" rows="3" id="UNIT\_DESCR" ></textarea>  
 </div>  
 </div>  
 </div>  
 <div class="row">  
 <div class="col-md-12">  
 <div class="form-group">  
 <label for="GCV\_STATION">GCV STATION</label>  
 <textarea class="form-control" name="GCV\_STATION" rows="3" id="GCV\_STATION" ></textarea>  
 </div>  
 </div>  
 </div>  
 <div class="row">  
 <div class="col-md-12">  
 <div class="form-group">  
 <label for="GCV\_S\_NAME">GCV\_S\_NAME</label>  
 <textarea class="form-control" name="GCV\_S\_NAME" rows="3" id="GCV\_S\_NAME" ></textarea>  
 </div>  
 </div>  
 </div>  
 <div class="row">  
 <div class="col-md-12">  
 <div class="form-group">  
 <label for="PHYSICAL\_SMS">PHYSICAL SMS</label>  
 <textarea class="form-control" name="PHYSICAL\_SMS" rows="3" id="PHYSICAL\_SMS" ></textarea>  
 </div>  
 </div>  
 </div>  
 <div class="row">  
 <div class="col-md-12">  
 <div class="form-group">  
 <label for="PHYSICAL\_SMS\_DESCR">PHYSICAL SMS DESCRIPTION</label>  
 <textarea class="form-control" name="PHYSICAL\_SMS\_DESCR" rows="3" id="PHYSICAL\_SMS\_DESCR" ></textarea>  
 </div>  
 </div>  
 </div>  
 <div class="row">  
 <div class="col-md-12">  
 <div class="form-group">  
 <label for="TBS">TBS</label>  
 <textarea class="form-control" name="TBS" rows="3" id="TBS" ></textarea>  
 </div>  
 </div>  
 </div>  
 <div class="row">  
 <div class="col-md-12">  
 <div class="form-group">  
 <label for="TBS\_DESCR">TBS DESCRIPTION</label>  
 <textarea class="form-control" name="TBS\_DESCR" rows="3" id="TBS\_DESCR" ></textarea>  
 </div>  
 </div>  
 </div>  
 <div class="row">  
 <div class="col-md-12">  
 <div class="form-group">  
 <label for="PRS">PRS</label>  
 <textarea class="form-control" name="PRS" rows="3" id="PRS" ></textarea>  
 </div>  
 </div>  
 </div>  
 <div class="row">  
 <div class="col-md-12">  
 <div class="form-group">  
 <label for="PRS\_DESCR">PRS DESCRIPTION</label>  
 <textarea class="form-control" name="PRS\_DESCR" rows="3" id="PRS\_DESCR" ></textarea>  
 </div>  
 </div>  
 </div>  
 <div class="row">  
 <div class="col-md-12">  
 <div class="form-group">  
 <label for="SYSTEM\_DATE">SYSTEM\_DATE</label>  
 <input type="date" class="form-control" name="SYSTEM\_DATE" rows="3" id="SYSTEM\_DATE" >  
 </div>  
 </div>  
 </div>  
 <div class="row">  
 <div class="col-md-12 text-center">  
 <button class="btn btn-success" type="submit">Submit</button>  
 <a class="btn btn-danger" href="{% url 'index' %}" type="button">Cancel</a>  
 </div>  
 </div>  
 </form>  
 </div>  
 </div>  
 </div>  
 </div>  
 </div>  
</body>  
</html>

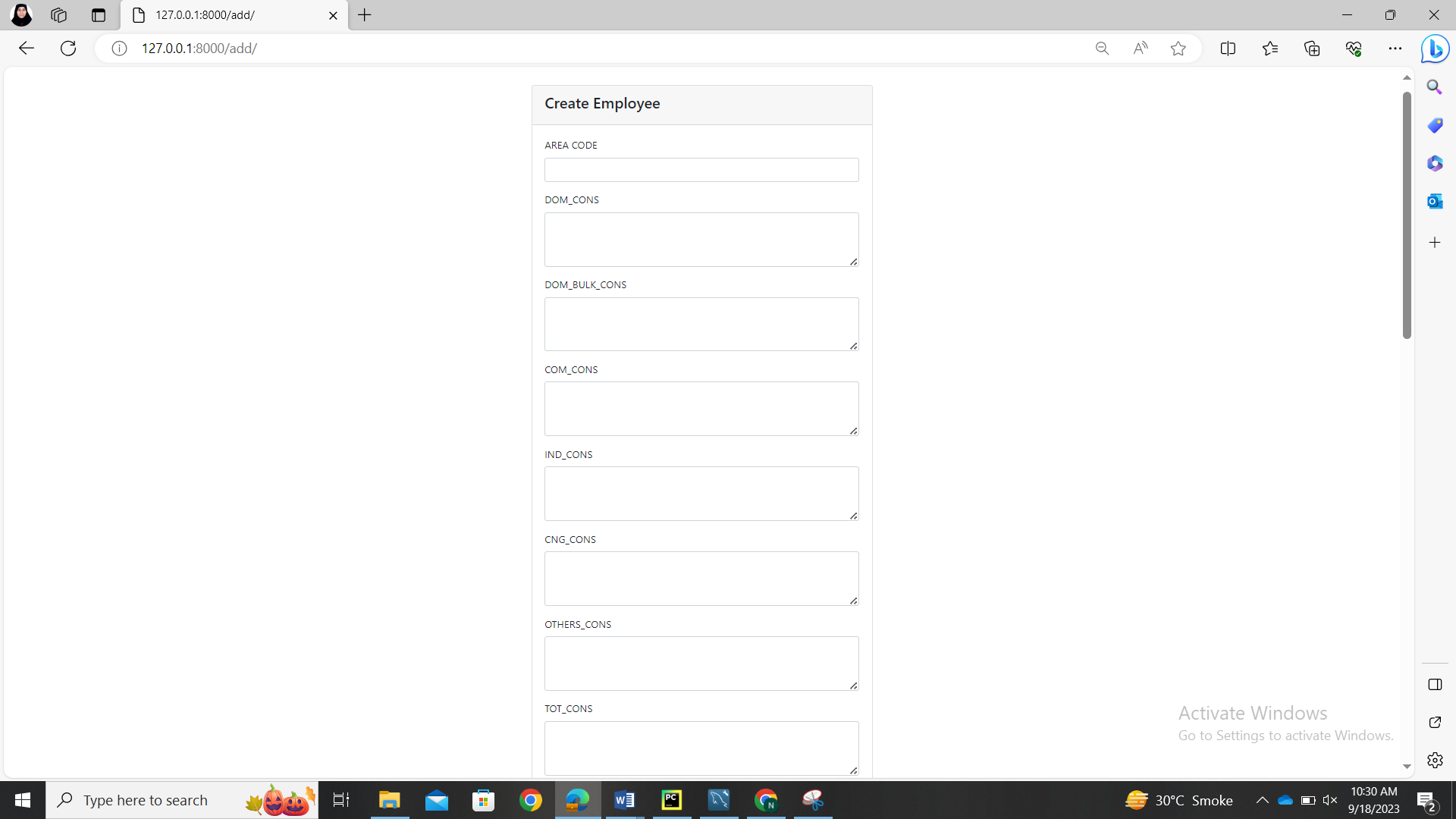
Create the file as **index.html** in templates. Write the following code in **index.html** file.

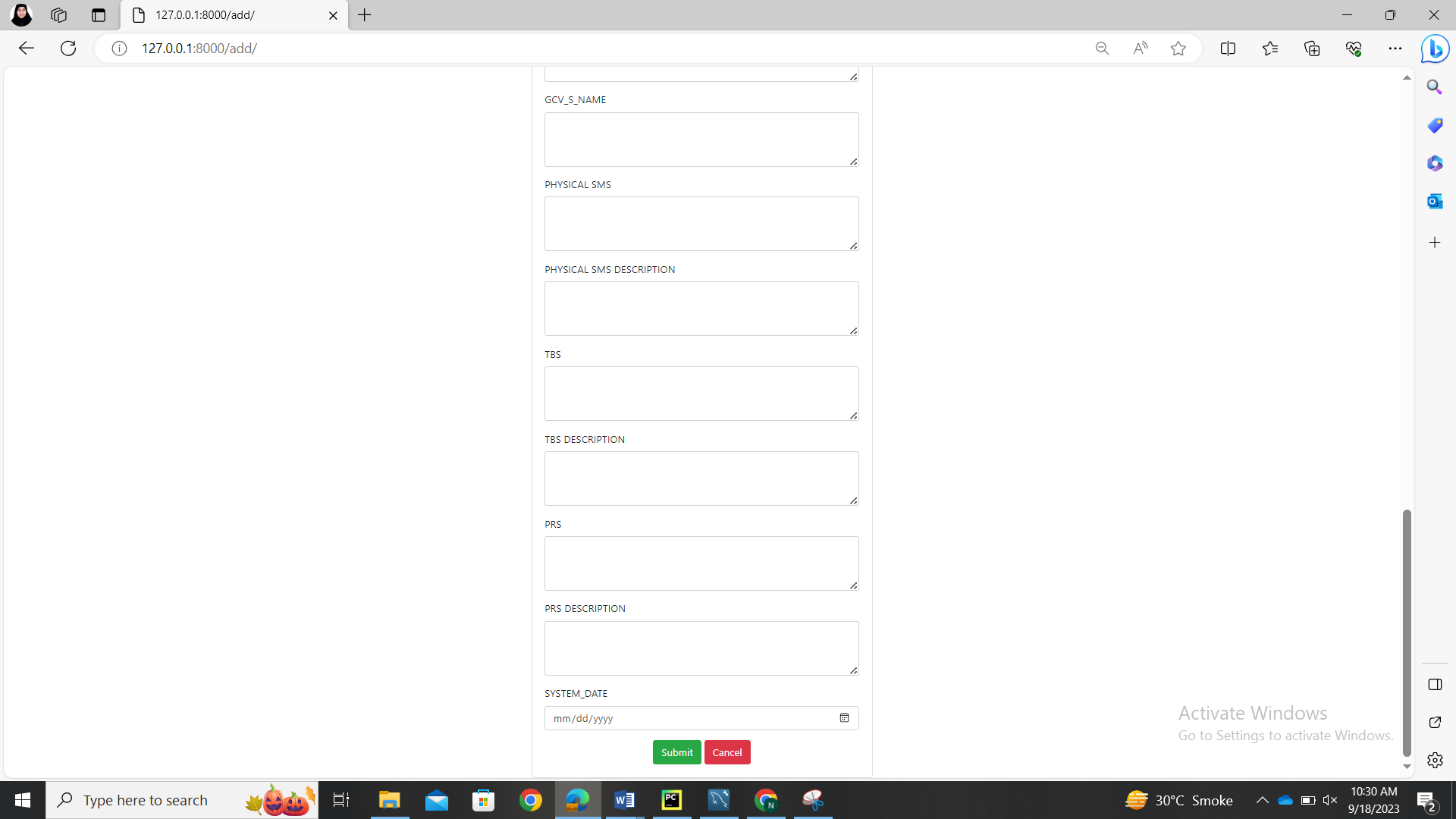
<!DOCTYPE html>  
<html>  
<head>  
 <meta charset="utf-8">  
 <meta name="viewport" content="width=device-width, initial-scale=1">  
 <title></title>  
 <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap@4.6.1/dist/css/bootstrap.min.css">  
 <script src="https://cdn.jsdelivr.net/npm/jquery@3.6.0/dist/jquery.slim.min.js"></script>  
 <script src="https://cdn.jsdelivr.net/npm/popper.js@1.16.1/dist/umd/popper.min.js"></script>  
 <script src="https://cdn.jsdelivr.net/npm/bootstrap@4.6.1/dist/js/bootstrap.bundle.min.js"></script>  
 <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css" />  
</head>  
<body>  
  
  
 <div class="container mt-5">  
 <div class="row">  
 <div class="col-md-12">  
 <div class="card">  
 <div class="card-header">  
 <div class="row">  
 <div class="col-md-9">  
 <form >  
 <h3 >Search for Employee: </h3>  
 <input type="text" name="AREA\_CD" placeholder="Enter Employee Full Name" class="m-2" size="30"/>  
 <input type="text" name="PHYSICAL\_SMS" placeholder="Enter Employee Contact" class="m-2" size="30"/>  
 <button type="submit" class="btn btn-primary mb-2 mx-2">Search</button>  
 </form>  
  
 <h3>Employee List Hello </h3>  
 </div>  
 <div class="col-md-3 text-right">  
 <a href="add/" class="btn btn-success btn-sm"> <i class="fa fa-plus"></i></a>  
 </div>  
 </div>  
 </div>  
 <div class="card-body">  
 <table class="table table-striped table-bordered table-sm" border="2">  
 <thead>  
 <tr>  
 <th width="150">AREA</th>  
 <th width="200">AREA\_DESCR</th>  
 <th width="80">PHYSICAL\_SMS</th>  
 <th width="80">PHYSICAL\_SMS\_DESCR</th>  
 <th width="80">TBS</th>  
 <th width="80">TBS\_DESCR</th>  
  
 <th width="80">Action</th>  
 </tr>  
 </thead>  
 <tbody>  
 {% for employee in postingapp %}  
 <tr>  
 <td>{{ employee.AREA\_CD }}</td>  
 <td>{{ employee.AREA\_DESCR }}</td>  
 <td>{{ employee.PHYSICAL\_SMS }}</td>  
 <td>{{ employee.PHYSICAL\_SMS\_DESCR }}</td>  
 <td>{{ employee.TBS }}</td>  
 <td>{{ employee.TBS\_DESCR }}</td>  
 <td>  
 <a href="show/{{ employee.AREA\_CD }}" class="btn btn-info btn-sm"> <i class="fa fa-eye"></i></a>  
 <a href="update/{{ employee.AREA\_CD }}" class="btn btn-primary btn-sm"> <i class="fa fa-pencil"></i></a>  
 <a href="delete/{{ employee.AREA\_CD }}" onclick="return confirm('Are you sure you want to delete this?')" class="btn btn-danger btn-sm"> <i class="fa fa-trash"></i></a>  
 </td>  
 </tr>  
 {% empty %}  
 <tr class="text-center">  
 <td colspan="4">There are no Record Found!</td>  
 </tr>  
 {% endfor %}  
 </tbody>  
 </table>  
 <nav aria-label="Page navigation example">  
 <ul class="pagination justify-content-end">  
 {% if postingapp.has\_previous %}  
 <li class="page-item">  
 <a class="page-link" href="?page={{ postingapp.previous\_page\_number }}">Previous</a>  
 </li>  
 {% else %}  
 <li class="page-item disabled">  
 <a class="page-link" href="#" tabindex="-1" aria-disabled="True">Previous</a>  
 </li>  
 {% endif %}  
 {% for i in postingapp.paginator.page\_range %}  
 {% if postingapp.number == i %}  
 <li class="page-item active" aria-current="page">  
 <span class="page-link">  
 {{ i }}  
 <span class="sr-only">(current)</span>  
 </span>  
 </li>  
 {% else %}  
 <li class="page-item"><a class="page-link" href="?page={{ i }}">{{ i }}</a></li>  
 {% endif %}  
 {% endfor %}  
 {% if postingapp.has\_next %}  
 <li class="page-item">  
 <a class="page-link" href="?page={{ postingapp.next\_page\_number }}">Next</a>  
 </li>  
 {% else %}  
 <li class="page-item disabled">  
 <a class="page-link" href="#" tabindex="-1" aria-disabled="True">Next</a>  
 </li>  
 {% endif %}  
 </ul>  
 </nav>  
 </div>  
 </div>  
 </div>  
 </div>  
 </div>  
</body>  
</html>

Write the following code into **urls.py** file in project folder.

from django.contrib import admin  
from django.urls import path  
  
from postingapp import views  
  
urlpatterns = [  
 path("admin/", admin.site.urls),  
 path('', views.index, name='index'),  
 path('add/', views.add),  
 path('add/addrecord/', views.addrecord),

**OUTPUT:**





**DISPLAY OPERATION:**

Views.py

def index(request):  
 postingapp = Postal.objects.all()  
 page = request.GET.get('page', 1)  
  
 paginator = Paginator(postingapp, 1000)  
 try:  
 postingapp = paginator.page(page)  
 except PageNotAnInteger:  
 postingapp = paginator.page(1)  
 except EmptyPage:  
 postingapp = paginator.page(paginator.num\_pages)  
  
 return render(request, 'show.html', {'postingapp': postingapp})  
  
  
def show(request):  
 postingapp = Postal.objects.all()  
 template = loader.get\_template('show.html')  
 context = {  
 'postingapp': postingapp,  
 }  
 return HttpResponse(template.render(context, request))

The index function here is used to divide the data in different pages because the data I am using have 22 colums and 4000 above rows.

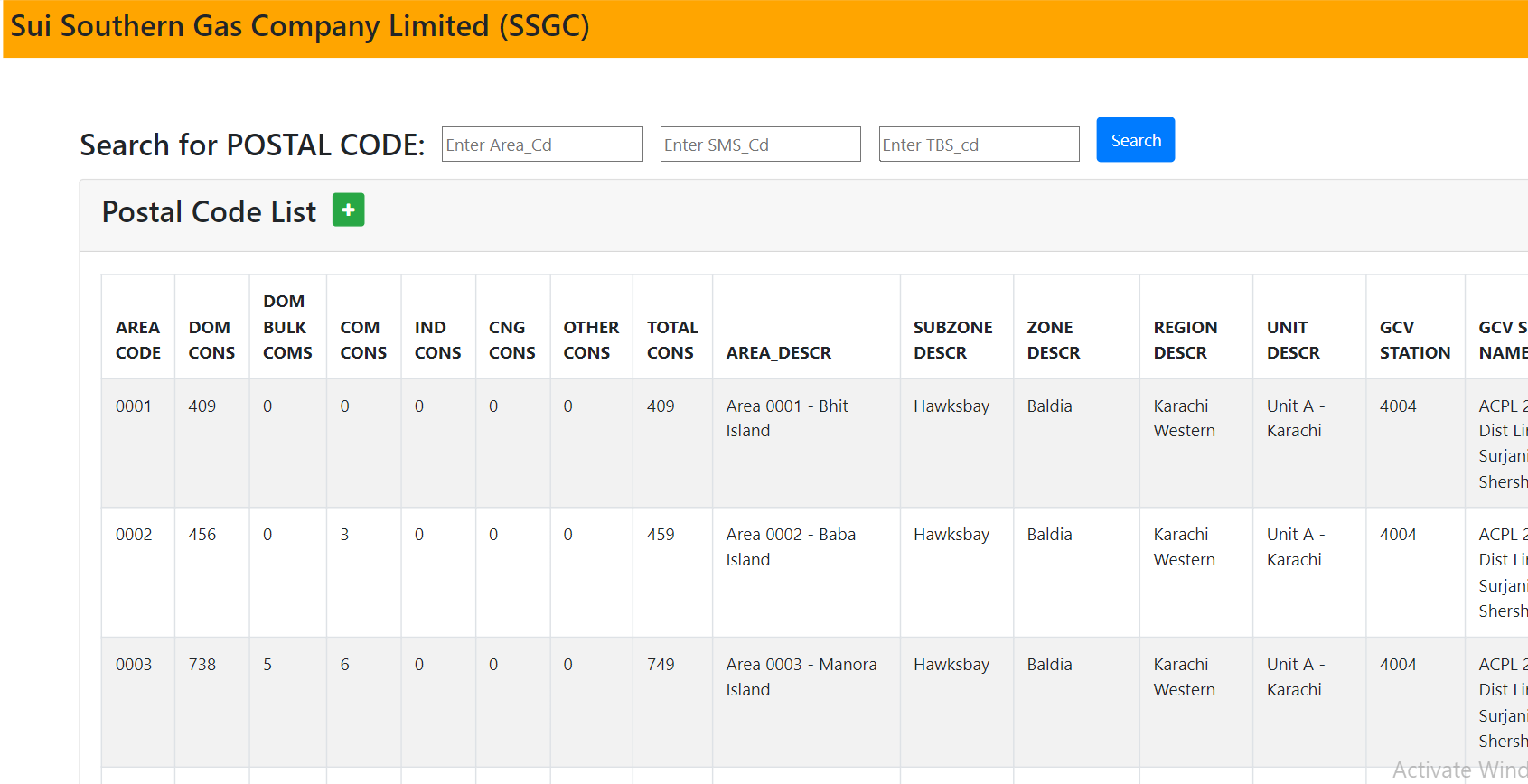
Create **show.html** file in templates folder

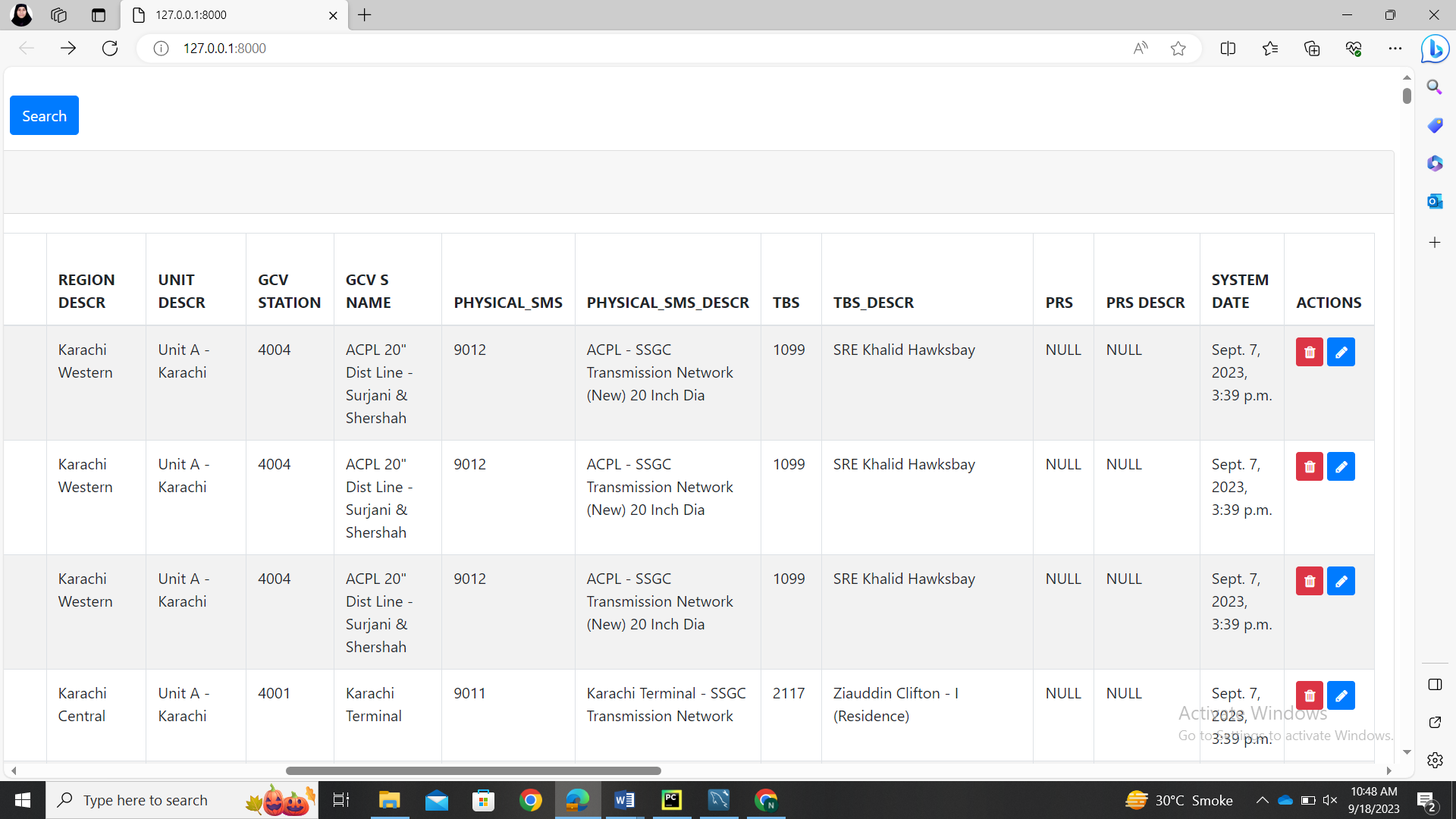
<!DOCTYPE html>  
<html>  
<head>  
 <meta charset="utf-8">  
 <meta name="viewport" content="width=device-width, initial-scale=1">  
  
 <meta name="viewport" content="width=device-width, initial-scale=1">  
 <title></title>  
 <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap@4.6.1/dist/css/bootstrap.min.css">  
 <script src="https://cdn.jsdelivr.net/npm/jquery@3.6.0/dist/jquery.slim.min.js"></script>  
 <script src="https://cdn.jsdelivr.net/npm/popper.js@1.16.1/dist/umd/popper.min.js"></script>  
 <script src="https://cdn.jsdelivr.net/npm/bootstrap@4.6.1/dist/js/bootstrap.bundle.min.js"></script>  
 <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css" integrity="sha512-SfTiTlX6kk+qitfevl/7LibUOeJWlt9rbyDn92a1DqWOw9vWG2MFoays0sgObmWazO5BQPiFucnnEAjpAB+/Sw==" crossorigin="anonymous" referrerpolicy="no-referrer" />  
 <style>  
 .row {  
 width: 220%;  
 margin:Auto;  
 }  
 bg-image {  
 background-image: url("ssgc.jpg");  
  
  
 filter: blur(8px);  
 -webkit-filter: blur(8px);  
 height: 100%;  
  
 /\* Center and scale the image nicely \*/  
 background-position: center;  
 background-repeat: no-repeat;  
 background-size: cover;}  
  
</style>  
</head>  
  
<body >  
<div class="bg-image">  
<div style= "margin-left:10px; padding:6px; background-color:orange;">  
<h3> Sui Southern Gas Company Limited (SSGC) </h3>  
</div>  
 <div class="container mt-5" style="margin-left:50px" >  
  
 <form class="d-flex p-2 mx-auto" action="search/" >  
 <h3 class="m-2">Search for POSTAL CODE: </h3>  
 <input type="text" name="AREA\_CD" placeholder="Enter Area\_Cd" class="m-2" size="20"/>  
  
 <input type="text" name="PHYSICAL\_SMS" placeholder="Enter SMS\_Cd" class="m-2" size="20"/>  
  
 <input type="text" name="TBS" placeholder="Enter TBS\_cd " class="m-2" size="20"/>  
  
 <button type="submit" class="btn btn-primary mb-2 mx-2">Search</button>  
</form>  
  
 <div class="row" >  
 <div class="col-md-12">  
 <div class="card">  
 <div class="card-header">  
 <div class="row">  
 <h3>Postal Code List</h3>  
  
 <div class="col-md-9">  
 <a href="add/" class="btn btn-success btn-sm"> <i class="fa fa-plus"></i></a>  
  
 </div>  
 <div class="col-md-3 text-right">  
 </div>  
 <div class="col-md-9">  
  
 </div>  
  
 </div>  
 </div>  
  
 <div class="card-body">  
 <table class="table table-bordered table-striped">  
 <thead>  
 <th width="200px">AREA CODE</th>  
 <th width="200px">DOM CONS</th>  
 <th width="200px">DOM BULK COMS</th>  
 <th width="200px">COM CONS</th>  
 <th width="200px">IND CONS</th>  
 <th width="200px">CNG CONS</th>  
 <th width="200px">OTHER CONS</th>  
 <th width="200px">TOTAL CONS</th>  
 <th width="200px">AREA\_DESCR</th>  
 <th width="200px">SUBZONE DESCR</th>  
 <th width="200px">ZONE DESCR</th>  
 <th width="200px">REGION DESCR</th>  
 <th width="200px">UNIT DESCR</th>  
 <th width="200px">GCV STATION</th>  
 <th width="500px">GCV S NAME</th>  
 <th width="200px">PHYSICAL\_SMS</th>  
 <th width="200px">PHYSICAL\_SMS\_DESCR</th>  
 <th width="200px">TBS</th>  
 <th width="200px">TBS\_DESCR</th>  
 <th width="200px">PRS</th>  
 <th width="200px">PRS DESCR</th>  
 <th width="500px">SYSTEM DATE</th>  
  
 <th width="200px"> ACTIONS </th>  
 </thead>  
 {% for employee in postingapp %}  
 <tr>  
 <td>{{ employee.AREA\_CD }}</td>  
 <td>{{ employee.DOM\_CONS }}</td>  
 <td>{{ employee.DOM\_BULK\_CONS }}</td>  
 <td>{{ employee.COM\_CONS }}</td>  
 <td>{{ employee.IND\_CONS }}</td>  
 <td>{{ employee.CNG\_CONS }}</td>  
 <td>{{ employee.OTHERS\_CONS }}</td>  
 <td>{{ employee.TOT\_CONS }}</td>  
 <td>{{ employee.AREA\_DESCR }}</td>  
 <td>{{ employee.SUBZONE\_DESCR }}</td>  
 <td>{{ employee.ZONE\_DESCR }}</td>  
 <td>{{ employee.REGION\_DESCR }}</td>  
 <td>{{ employee.UNIT\_DESCR }}</td>  
 <td>{{ employee.GCV\_STATION }}</td>  
 <td>{{ employee.GCV\_S\_NAME }}</td>  
 <td>{{ employee.PHYSICAL\_SMS }}</td>  
 <td>{{ employee.PHYSICAL\_SMS\_DESCR }}</td>  
 <td>{{ employee.TBS }}</td>  
 <td>{{ employee.TBS\_DESCR }}</td>  
 <td>{{ employee.PRS }}</td>  
 <td>{{ employee.PRS\_DESCR }}</td>  
 <td>{{ employee.SYSTEM\_DATE }}</td>  
 <td>  
  
 <a href="delete/{{ employee.AREA\_CD }}" onclick="return confirm('Are you sure you want to delete this?')" class="btn btn-danger btn-sm"> <i class="fa fa-trash"></i></a>  
 <!-- <a href="#" class="btn btn-warning btn-sm"> Edit </a> -->  
 <a href="update/{{ employee.AREA\_CD }}" class="btn btn-primary btn-sm"> <i class="fa fa-pencil"></i></a>  
  
  
 </td>  
 </tr>  
 {% endfor %}  
  
 </thead>  
 </table>  
 <nav aria-label="Page navigation example">  
 <ul class="pagination justify-content-end">  
 {% if postingapp.has\_previous %}  
 <li class="page-item">  
 <a class="page-link" href="?page={{ postingapp.previous\_page\_number }}">Previous</a>  
 </li>  
 {% else %}  
 <li class="page-item disabled">  
 <a class="page-link" href="#" tabindex="-1" aria-disabled="True">Previous</a>  
 </li>  
 {% endif %}  
 {% for i in postingapp.paginator.page\_range %}  
 {% if postingapp.number == i %}  
 <li class="page-item active" aria-current="page">  
 <span class="page-link">  
 {{ i }}  
 <span class="sr-only">(current)</span>  
 </span>  
 </li>  
 {% else %}  
 <li class="page-item"><a class="page-link" href="?page={{ i }}">{{ i }}</a></li>  
 {% endif %}  
 {% endfor %}  
 {% if postingapp.has\_next %}  
 <li class="page-item">  
 <a class="page-link" href="?page={{ postingapp.next\_page\_number }}">Next</a>  
 </li>  
 {% else %}  
 <li class="page-item disabled">  
 <a class="page-link" href="#" tabindex="-1" aria-disabled="True">Next</a>  
 </li>  
 {% endif %}  
 </ul>  
 </nav>  
 </div>  
 </div>  
 </div>  
 </div>  
 </div>  
 </div>  
</body>  
</html>

Add the following code in **urls.py** file

path('show/', views.show, name='show'),

**OUTPUT:**





**UPDATE OPERATION:**

Views.py

def updaterecord(request, AREA\_CD):  
 # a = request.POST['AREA\_CD']  
 b = request.POST['DOM\_CONS']  
 c = request.POST['DOM\_BULK\_CONS']  
 d = request.POST['COM\_CONS']  
 e = request.POST['IND\_CONS']  
 f = request.POST['CNG\_CONS']  
 g = request.POST['OTHERS\_CONS']  
 h = request.POST['TOT\_CONS']  
 i = request.POST['AREA\_DESCR']  
 j = request.POST['SUBZONE\_DESCR']  
 k = request.POST['ZONE\_DESCR']  
 l = request.POST['REGION\_DESCR']  
 m = request.POST['UNIT\_DESCR']  
 n = request.POST['GCV\_STATION']  
 o = request.POST['GCV\_S\_NAME']  
 p = request.POST['PHYSICAL\_SMS']  
 q = request.POST['PHYSICAL\_SMS\_DESCR']  
 r = request.POST['TBS']  
 s = request.POST['TBS\_DESCR']  
 t = request.POST['PRS']  
 u = request.POST['PRS\_DESCR']  
 v = request.POST['SYSTEM\_DATE']  
 postingapp = Postal.objects.get(AREA\_CD=AREA\_CD)  
 postingapp.DOM\_CONS = b  
 postingapp.DOM\_BULK\_CONS = c  
 postingapp.COM\_CONS = d  
 postingapp.IND\_CONS = e  
 postingapp.CNG\_CONS = f  
 postingapp.OTHERS\_CONS = g  
 postingapp.TOT\_CONS = h  
 postingapp.AREA\_DESCR = i  
 postingapp.SUBZONE\_DESCR = j  
 postingapp.ZONE\_DESCR = k  
 postingapp.REGION\_DESCR = l  
 postingapp.UNIT\_DESCR = m  
 postingapp.GCV\_STATION = n  
 postingapp.GCV\_S\_NAME = o  
 postingapp.PHYSICAL\_SMS = p  
 postingapp.PHYSICAL\_SMS\_DESCR = q  
 postingapp.TBS = r  
 postingapp.TBS\_DESCR = s  
 postingapp.PRS = t  
 postingapp.PRS\_DESCR = u  
 postingapp.SYSTEM\_DATE = v  
 postingapp.save()  
 return HttpResponseRedirect(reverse('index'))  
  
  
# employee.ename = ename  
# employee.eemail = eemail  
# employee.econtact = econtact  
# employee.save()  
  
  
def update(request, AREA\_CD):  
 postingapp = Postal.objects.get(AREA\_CD=AREA\_CD)  
 if request.method == 'POST':  
 form = PostalForm(request.POST, instance=postingapp)  
 if form.is\_valid():  
 form.save()  
 return redirect('/show')  
 else:  
 form = PostalForm(instance=postingapp)  
  
 return render(request, 'edit.html', {'form': form, 'postingapp': postingapp})

Create **edit.html** file in templates folder

<!DOCTYPE html>  
<html>  
<head>  
 <meta charset="utf-8">  
 <meta name="viewport" content="width=device-width, initial-scale=1">  
 <title></title>  
 <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap@4.6.1/dist/css/bootstrap.min.css">  
 <script src="https://cdn.jsdelivr.net/npm/jquery@3.6.0/dist/jquery.slim.min.js"></script>  
 <script src="https://cdn.jsdelivr.net/npm/popper.js@1.16.1/dist/umd/popper.min.js"></script>  
 <script src="https://cdn.jsdelivr.net/npm/bootstrap@4.6.1/dist/js/bootstrap.bundle.min.js"></script>  
 <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css" integrity="sha512-SfTiTlX6kk+qitfevl/7LibUOeJWlt9rbyDn92a1DqWOw9vWG2MFoays0sgObmWazO5BQPiFucnnEAjpAB+/Sw==" crossorigin="anonymous" referrerpolicy="no-referrer" />  
</head>  
<body>  
 <div class="container mt-5">  
 <div class="row d-flex justify-content-center">  
 <div class="col-md-6">  
 <div class="card">  
 <div class="card-header">  
 <div class="row">  
 <div class="col-md-12">  
 <h3>Update Employee</h3>  
 </div>  
 </div>  
 </div>  
 <div class="card-body">  
 <form action="updaterecord/{{ postingapp.AREA\_CD }}" method="post">  
 {% csrf\_token %}  
  
 <div class="row">  
 <div class="col-md-12">  
 <div class="form-group">  
 <label for="AREA\_CD"><B> AREA CODE </B></label>  
 <input type="text" class="form-control" name="AREA\_CD" id="AREA\_CD" value="{{ postingapp.AREA\_CD }}">  
 </div>  
 </div>  
 </div>  
 <div class="row">  
 <div class="col-md-12">  
 <div class="form-group">  
 <label for="DOM\_CONS">DOM\_CONS</label>  
 <input type="text" class="form-control" name="DOM\_CONS" id="DOM\_CONS" value="{{ postingapp.DOM\_CONS }}" />  
 </div>  
 </div>  
 </div>  
 <div class="row">  
 <div class="col-md-12">  
 <div class="form-group">  
 <label for="DOM\_BULK\_CONS">DOM\_BULK\_CONS</label>  
 <input type="text" class="form-control" name="DOM\_BULK\_CONS" rows="3" id="DOM\_BULK\_CONS" value="{{ postingapp.DOM\_BULK\_CONS }}">  
 </div>  
 </div>  
 </div>  
 <div class="row">  
 <div class="col-md-12">  
 <div class="form-group">  
 <label for="COM\_CONS">COM\_CONS</label>  
 <input type="text" class="form-control" name="COM\_CONS" rows="3" id="COM\_CONS" value="{{ postingapp.COM\_CONS }}">  
 </div>  
 </div>  
 </div>  
  
 <div class="row">  
 <div class="col-md-12">  
 <div class="form-group">  
 <label for="IND\_CONS">IND\_CONS</label>  
 <input type="text" class="form-control" name="IND\_CONS" rows="3" id="IND\_CONS" value="{{ postingapp.IND\_CONS }}">  
 </div>  
 </div>  
 </div>  
 <div class="row">  
 <div class="col-md-12">  
 <div class="form-group">  
 <label for="CNG\_CONS">CNG\_CONS</label>  
 <input type="text" class="form-control" name="CNG\_CONS" rows="3" id="CNG\_CONS" value="{{ postingapp.CNG\_CONS }}">  
 </div>  
 </div>  
 </div>  
  
 <div class="row">  
 <div class="col-md-12">  
 <div class="form-group">  
 <label for="OTHERS\_CONS">OTHERS\_CONS</label>  
 <input type="text" class="form-control" name="OTHERS\_CONS" rows="3" id="OTHERS\_CONS" value="{{ postingapp.OTHERS\_CONS }}">  
 </div>  
 </div>  
 </div>  
 <div class="row">  
 <div class="col-md-12">  
 <div class="form-group">  
 <label for="TOT\_CONS">TOT\_CONS</label>  
 <input type="text" class="form-control" name="TOT\_CONS" rows="3" id="TOT\_CONS" value="{{ postingapp.TOT\_CONS }}">  
 </div>  
 </div>  
 </div>  
 <div class="row">  
 <div class="col-md-12">  
 <div class="form-group">  
 <label for="AREA\_DESCR">AREA DESCRIPTION</label>  
 <input type="text" class="form-control" name="AREA\_DESCR" rows="3" id="AREA\_DESCR" value="{{ postingapp.AREA\_DESCR }}">  
 </div>  
 </div>  
 </div>  
 <div class="row">  
 <div class="col-md-12">  
 <div class="form-group">  
 <label for="SUBZONE\_DESCR">SUBZONE DESCRIPTION</label>  
 <input type="text" class="form-control" name="SUBZONE\_DESCR" rows="3" id="SUBZONE\_DESCR" value="{{ postingapp.SUBZONE\_DESCR }}">  
 </div>  
 </div>  
 </div>  
 <div class="row">  
 <div class="col-md-12">  
 <div class="form-group">  
 <label for="ZONE\_DESCR">ZONE DESCRIPTION</label>  
 <input type="text" class="form-control" name="ZONE\_DESCR" rows="3" id="ZONE\_DESCR" value="{{ postingapp.ZONE\_DESCR }}">  
 </div>  
 </div>  
 </div>  
 <div class="row">  
 <div class="col-md-12">  
 <div class="form-group">  
 <label for="REGION\_DESCR">REGION DESCRIPTION</label>  
 <input type="text" class="form-control" name="REGION\_DESCR" rows="3" id="REGION\_DESCR" value="{{ postingapp.REGION\_DESCR }}">  
 </div>  
 </div>  
 </div>  
 <div class="row">  
 <div class="col-md-12">  
 <div class="form-group">  
 <label for="UNIT\_DESCR">UNIT DESCRIPTION</label>  
 <input type="text" class="form-control" name="UNIT\_DESCR" rows="3" id="UNIT\_DESCR" value="{{ postingapp.UNIT\_DESCR }}">  
 </div>  
 </div>  
 </div>  
 <div class="row">  
 <div class="col-md-12">  
 <div class="form-group">  
 <label for="GCV\_STATION">GCV STATION</label>  
 <input type="text" class="form-control" name="GCV\_STATION" rows="3" id="GCV\_STATION" value="{{ postingapp.GCV\_STATION }}">  
 </div>  
 </div>  
 </div>  
 <div class="row">  
 <div class="col-md-12">  
 <div class="form-group">  
 <label for="GCV\_S\_NAME">GCV\_S\_NAME</label>  
 <input type="text" class="form-control" name="GCV\_S\_NAME" rows="3" id="GCV\_S\_NAME" value="{{ postingapp.GCV\_S\_NAME }}">  
 </div>  
 </div>  
 </div>  
 <div class="row">  
 <div class="col-md-12">  
 <div class="form-group">  
 <label for="PHYSICAL\_SMS">PHYSICAL SMS</label>  
 <input type="text" class="form-control" name="PHYSICAL\_SMS" rows="3" id="PHYSICAL\_SMS" value="{{ postingapp.PHYSICAL\_SMS }}">  
 </div>  
 </div>  
 </div>  
 <div class="row">  
 <div class="col-md-12">  
 <div class="form-group">  
 <label for="PHYSICAL\_SMS\_DESCR">PHYSICAL SMS DESCRIPTION</label>  
 <input type="text" class="form-control" name="PHYSICAL\_SMS\_DESCR" rows="3" id="PHYSICAL\_SMS\_DESCR" value="{{ postingapp.PHYSICAL\_SMS\_DESCR }}">  
 </div>  
 </div>  
 </div>  
 <div class="row">  
 <div class="col-md-12">  
 <div class="form-group">  
 <label for="TBS">TBS</label>  
 <input type="text" class="form-control" name="TBS" rows="3" id="TBS" value="{{ postingapp.TBS }}">  
 </div>  
 </div>  
 </div>  
 <div class="row">  
 <div class="col-md-12">  
 <div class="form-group">  
 <label for="TBS\_DESCR">TBS DESCRIPTION</label>  
 <input type="text" class="form-control" name="TBS\_DESCR" rows="3" id="TBS\_DESCR" value="{{ postingapp.TBS\_DESCR }}">  
 </div>  
 </div>  
 </div>  
 <div class="row">  
 <div class="col-md-12">  
 <div class="form-group">  
 <label for="PRS">PRS</label>  
 <input type="text" class="form-control" name="PRS" rows="3" id="PRS" value="{{ postingapp.PRS }}">  
 </div>  
 </div>  
 </div>  
 <div class="row">  
 <div class="col-md-12">  
 <div class="form-group">  
 <label for="PRS\_DESCR">PRS DESCRIPTION</label>  
 <input type="text" class="form-control" name="PRS\_DESCR" rows="3" id="PRS\_DESCR" value="{{ postingapp.PRS\_DESCR }}">  
 </div>  
 </div>  
 </div>  
 <div class="row">  
 <div class="col-md-12">  
 <div class="form-group">  
 <label for="SYSTEM\_DATE">SYSTEM\_DATE</label>  
 <input type="date" class="form-control" name="SYSTEM\_DATE" rows="3" id="SYSTEM\_DATE" value="{{ postingapp.SYSTEM\_DATE }}">  
 </div>  
 </div>  
 </div>  
 <div class="row">  
 <div class="col-md-12 text-center">  
 <button class="btn btn-success" type="submit">Update</button>  
 <a class="btn btn-danger" href="{% url 'index' %}" type="button">Cancel</a>  
 </div>  
 </div>  
 </form>  
 </div>  
  
 </div>  
 </div>  
 </div>  
 </div>  
</body>  
</html>

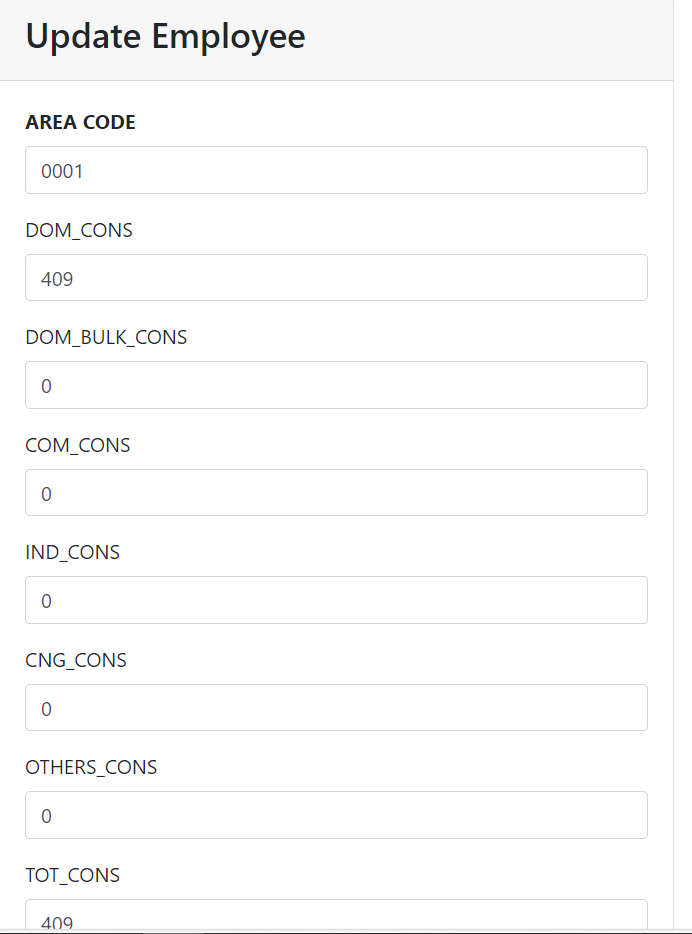
urls.py

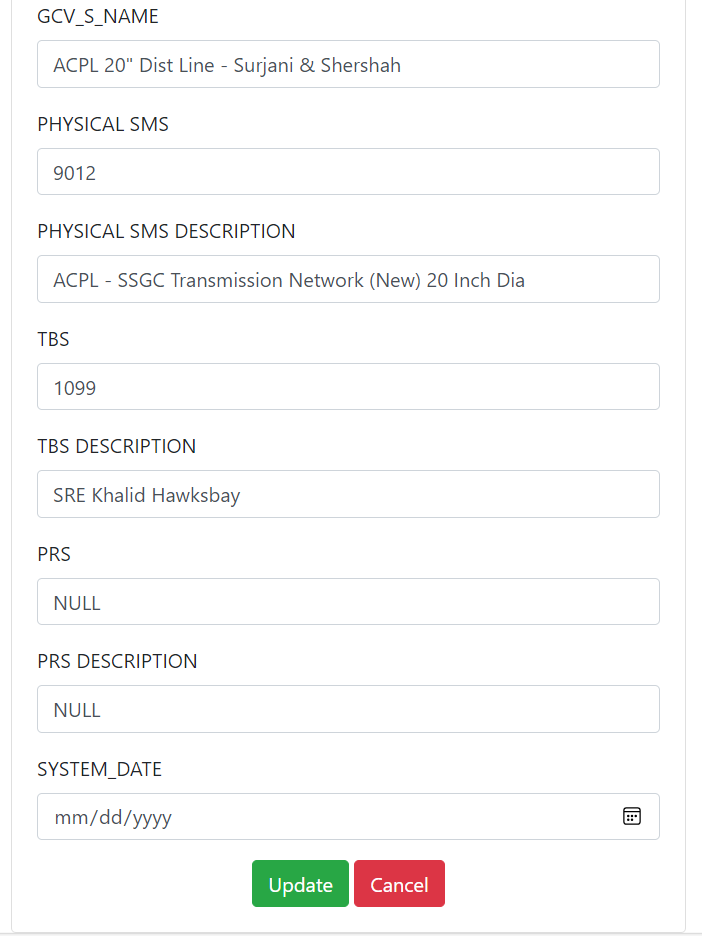
path('update/<int:AREA\_CD>', views.update, name='update'),  
path('update/updaterecord/<int:AREA\_CD>', views.updaterecord,

name='updaterecord'),

**OUTPUT:**

After clicking the pencil button in each row it will give something like below





**DELETE OPERATION:**

**Views.html**

def delete(request, AREA\_CD):  
 postingapp = Postal.objects.get(AREA\_CD=AREA\_CD)  
 postingapp.delete()  
 return HttpResponseRedirect(reverse('index'))

**urls.py**

path('delete/<int:AREA\_CD>', views.delete, name='delete'),

**OUTPUT**

Now after clicking the trash can button present in every colum you it will simply delete the corresponding colum in the data after showing the confirmation message.

**SEARCH OPERATION:**

**views.html**

def search(request):  
 AREA\_CD = request.GET.get('AREA\_CD')  
 PHYSICAL\_SMS = request.GET.get('PHYSICAL\_SMS')  
 TBS = request.GET.get('TBS')  
 if AREA\_CD:  
 postingapp = Postal.objects.filter(AREA\_CD\_\_iexact=AREA\_CD)  
 elif PHYSICAL\_SMS:  
 postingapp = Postal.objects.filter(PHYSICAL\_SMS\_\_iexact=PHYSICAL\_SMS)  
 elif TBS:  
 postingapp = Postal.objects.filter(TBS\_\_iexact=TBS)  
 elif AREA\_CD and PHYSICAL\_SMS:  
 postingapp = Postal.objects.filter(AREA\_CD\_\_iexact=AREA\_CD) & Postal.objects.filter(  
 PHYSICAL\_SMS\_\_iexact=PHYSICAL\_SMS)  
 elif AREA\_CD and TBS:  
 postingapp = Postal.objects.filter(AREA\_CD\_\_iexact=AREA\_CD) & Postal.objects.filter(  
 TBS\_\_iexact=TBS)  
 elif TBS and PHYSICAL\_SMS:  
 postingapp = Postal.objects.filter(TBS\_\_iexact=TBS) & Postal.objects.filter(  
 PHYSICAL\_SMS\_\_iexact=PHYSICAL\_SMS)  
 elif AREA\_CD and PHYSICAL\_SMS and TBS:  
 postingapp = Postal.objects.filter(AREA\_CD\_\_iexact=AREA\_CD) & Postal.objects.filter(  
 PHYSICAL\_SMS\_\_iexact=PHYSICAL\_SMS) & Postal.objects.filter(TBS\_\_iexact=TBS)  
 else:  
 postingapp = Postal.objects.none()  
 return render(request, 'search.html', {'postingapp': postingapp})

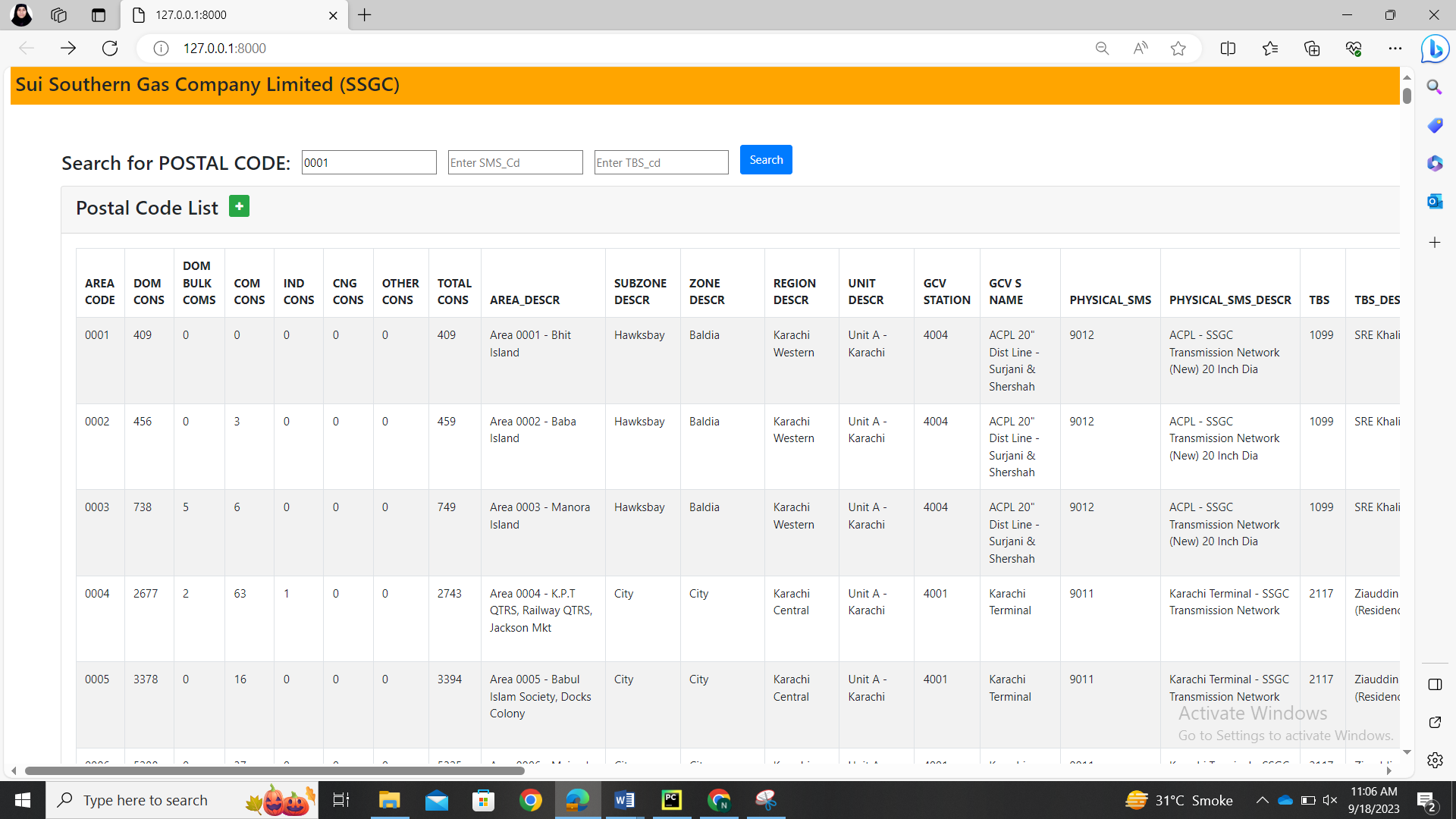
Create **search.html** file in the templates folder

<!DOCTYPE html>  
<html lang="en">  
<head>  
 <meta charset="UTF-8">  
 <title>Postal Area</title>  
   
</head>  
<body>  
<h1 class="m-5">Postal Area </h1>  
<table class="table table-striped table-stripped table-sm" border="2">  
 <thead>  
 <tr>  
 <th>AREA CODE</th>  
 <th> DOM CONS</th>  
 <th> DOM BULK CONS</th>  
 <th> COM CONS</th>  
 <th> IND CONS</th>  
 <th> CNG CONS</th>  
 <th> OTHERS CONS</th>  
 <th> TOTAL CONS</th>  
 <th>AREA DESCRIPTION</th>  
 <th>SUBZONE DESCRIPTION</th>  
 <th>ZONE DESCRIPTION</th>  
 <th>REGION DESCRIPTION</th>  
 <th>UNIT DESCRIPTION</th>  
 <th>GCV STATION</th>  
 <th>GCV S NAME</th>  
 <th>SMS CODE</th>  
 <th>SMS DESCRIPTION</th>  
 <th>TBS CODE</th>  
 <th>TBS DESCRIPTION</th>  
 <th>PRS</th>  
 <th>PRS DESCRIPTION</th>  
 <th>SYSTEM DATE</th>  
  
  
 <th> ACTIONS</th>  
 </tr>  
 </thead>  
 <tbody>  
 {% for a in postingapp %}  
 <tr>  
 <td>{{a.AREA\_CD}}</td>  
 <td> {{a.DOM\_CONS}}</td>  
 <td> {{a.DOM\_BULK\_CONS}}</td>  
 <td> {{a.COM\_CONS}}</td>  
 <td> {{a.IND\_CONS}}</td>  
 <td> {{a.CNG\_CONS}}</td>  
 <td> {{a.OTHERS\_CONS}}</td>  
 <td> {{a.TOT\_CONS}}</td>  
 <td>{{a.AREA\_DESCR}}</td>  
 <td>{{a.SUBZONE\_DESCR}}</td>  
 <td>{{a.ZONE\_DESCR}}</td>  
 <td>{{a.REGION\_DESCR}}</td>  
 <td>{{a.UNIT\_DESCR}}</td>  
 <td>{{a.GCV\_STATION}}</td>  
 <td>{{a.GCV\_S\_NAME}}</td>  
 <td>{{a.PHYSICAL\_SMS}}</td>  
 <td>{{a.PHYSICAL\_SMS\_DESCR}}</td>  
 <td>{{a.TBS}}</td>  
 <td>{{a.TBS\_DESCR}}</td>  
 <td>{{a.PRS}}</td>  
 <td>{{a.PRS\_DESCR}}</td>  
 <td>{{a.SYSTEM\_DATE}}</td>  
 </tbody>  
  
</table>  
 <div class="col-md-12 text-center">  
 <a class="btn btn-danger" href="{% url 'index' %}" type="button">Cancel</a>  
 </div>  
</body>  
</html>

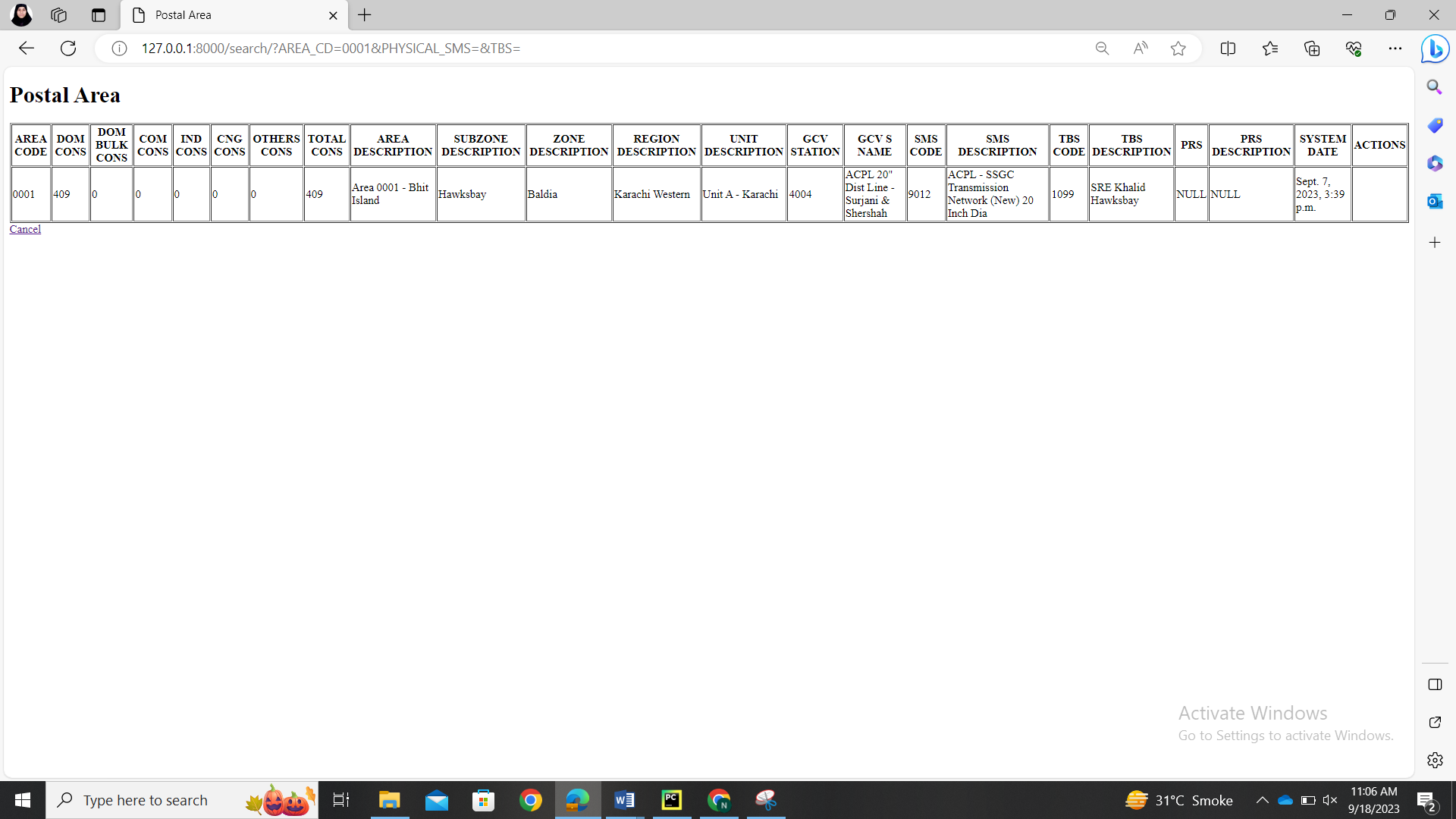
**Urls.py**

path('search/', views.search, name='search'),

**OUTPUT:**



After entering the desired input and clicking the search button it will provide the following output.



Now, the thing is that I have a lot of data which I have to maintain so I am able to search the data with search function but if I have to edit the row at 3000 row so i have to go all the way down to the 3000 row and them I will delete or edit that data which is not good so I decided to put the edit and delete buttons in each row in search.html page so the person can easily perform action after search the desired output so we have to perform some change in two file ( **urls.py and search.html** )

**Search.html:**

Add the below code in the head of html file

<link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap@4.6.1/dist/css/bootstrap.min.css">  
<script src="https://cdn.jsdelivr.net/npm/jquery@3.6.0/dist/jquery.slim.min.js"></script>  
<script src="https://cdn.jsdelivr.net/npm/popper.js@1.16.1/dist/umd/popper.min.js"></script>  
<script src="https://cdn.jsdelivr.net/npm/bootstrap@4.6.1/dist/js/bootstrap.bundle.min.js"></script>  
<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css" integrity="sha512-SfTiTlX6kk+qitfevl/7LibUOeJWlt9rbyDn92a1DqWOw9vWG2MFoays0sgObmWazO5BQPiFucnnEAjpAB+/Sw==" crossorigin="anonymous" referrerpolicy="no-referrer" />

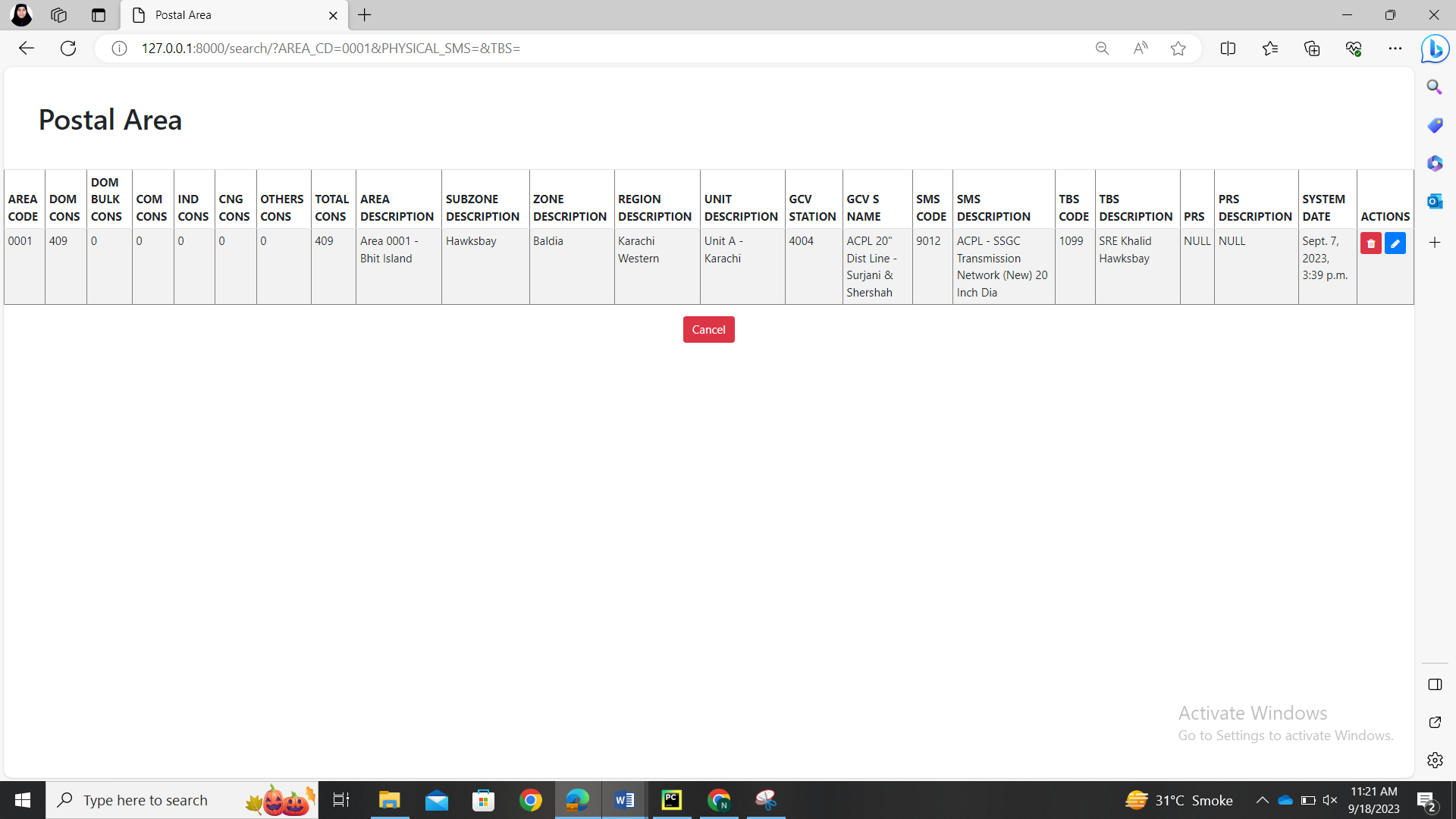
Add the below code in body of the table code:The first line in the code for you to reference that where you have to put it.

<td>{{ employee.SYSTEM\_DATE }}</td>   
<td>  
  
 <a href="delete/{{ employee.AREA\_CD }}" onclick="return confirm('Are you sure you want to delete this?')" class="btn btn-danger btn-sm"> <i class="fa fa-trash"></i></a>  
 <a href="update/{{ employee.AREA\_CD }}" class="btn btn-primary btn-sm"> <i class="fa fa-pencil"></i></a>  
  
 </td>

**Urls.py**

path('search/delete/<int:AREA\_CD>', views.delete, name='delete'),  
path('search/update/<int:AREA\_CD>', views.update, name='update'),  
path('search/update/updaterecord/<int:AREA\_CD>', views.updaterecord, name='updaterecord'),

OUTPUT:



Now I am able to edit my rows by searching them.

## **Errors Encountered and their solutions:**

Error:

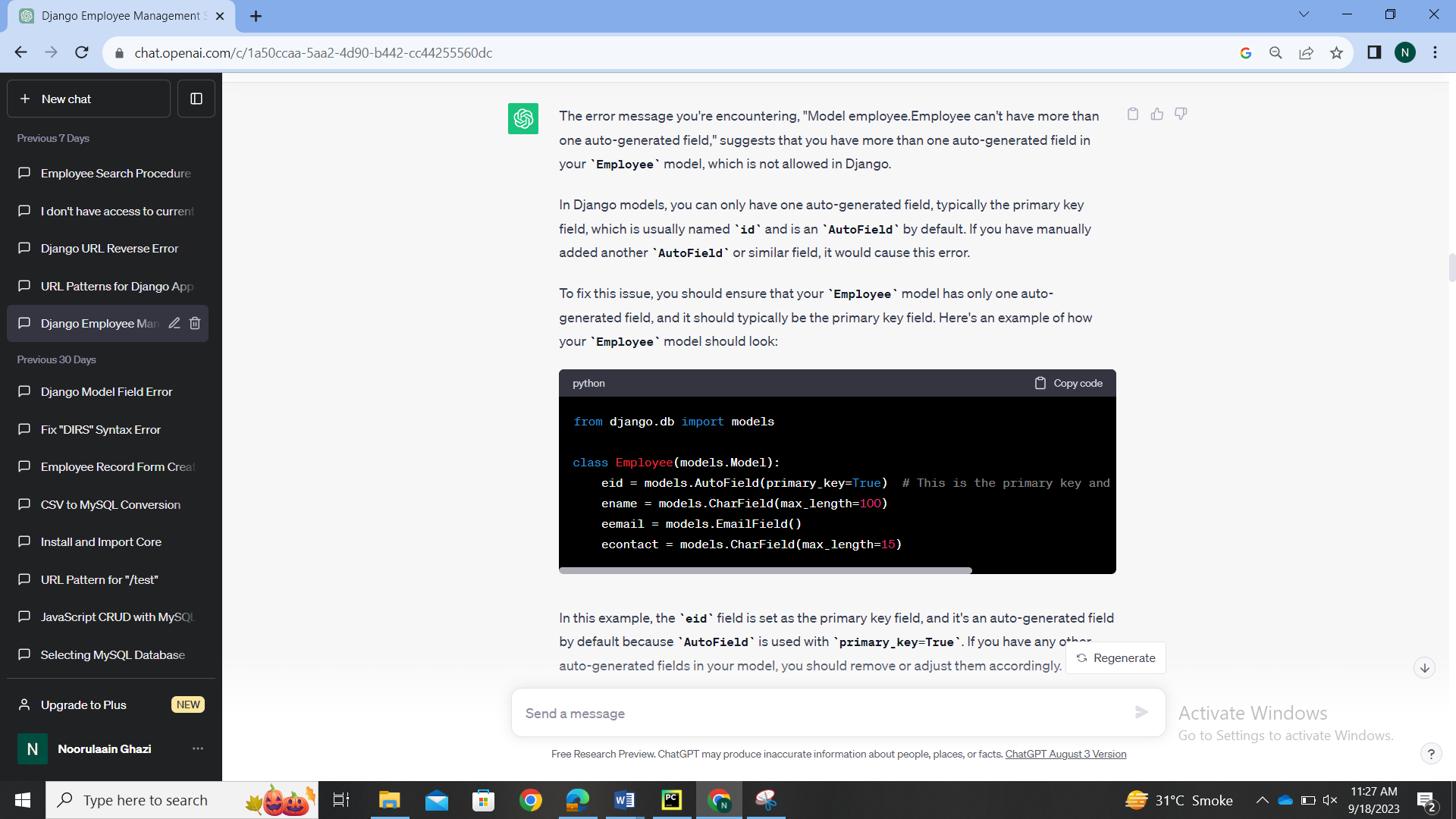
raise ValueError( ValueError: Model employee.Employee can't have more than one auto-generated field.

Solution:

The error message you're encountering, "Model employee.Employee can't have more than one auto-generated field," suggests that you have more than one auto-generated field in your **Employee** model, which is not allowed in Django.

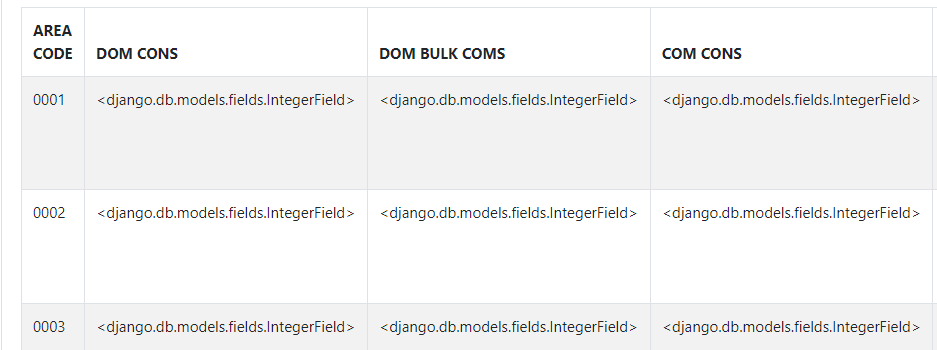
In Django models, you can only have one auto-generated field, typically the primary key field, which is usually named **id** and is an **AutoField** by default. If you have manually added another **AutoField** or similar field, it would cause this error.

To fix this issue, you should ensure that your **Employee** model has only one auto-generated field, and it should typically be the primary key field. Here's an example of how your **Employee** model should look:



Error:

The most important error I faced was that my Colum are of two datatype integer and string. So I defined all my columns in models with the same datatypes as the as I defined them in my database table. Here I encountered the problem when I put Integer field where the value in my database is integer the compiler don’t give me any error nor the web page but when I see the column It is not giving me the Colum value in the database. Let me show you what it was showing me instead of the correct value.



Solution:

I that point I came to know that python takes integer values as the char. Here I change All my integer fields to the char value while entering through the webpage.

Error:

We will not called this problem error. Now my whole code was correct it does not have any error but I was facing the error on webpage that when I was clicking any update or delete button it directs me to the error page and when I write in the the url block The row number then it was performing the action on the entered row and it was wrong because Lets say that I am clicking the button at row 1 and then it directs me to the empty error page and them I write the 10 row in the url then it is giving me the form to delete or update 10 that was incorrect because I want that when ever I click the update button it will directly take me to the form of that row and I don’t need to reenter the row number.

Solution:

So for this I do a slight change in the html file of the show and index and it give me what I want. Lets see the code.

UNCORRECTED CODE

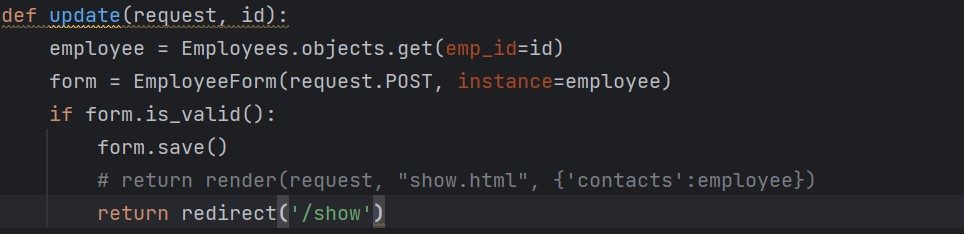
<a href="delete/{{ }}" onclick="return confirm('Are you sure you want to delete this?')" class="btn btn-danger btn-sm"> <i class="fa fa-trash"></i></a>  
 <a href="update/{{}}" class="btn btn-primary btn-sm"> <i class="fa fa-pencil"></i></a>

CORRECTED CODE

<a href="delete/{{ employee.AREA\_CD }}" onclick="return confirm('Are you sure you want to delete this?')" class="btn btn-danger btn-sm"> <i class="fa fa-trash"></i></a>  
 <a href="update/{{ employee.AREA\_CD }}" class="btn btn-primary btn-sm"> <i class="fa fa-pencil"></i></a>

Same changes are made in the index file.

Error:



Solution:

The error you're encountering is because the update view in your views.py does not have a return statement for all possible code paths. In Django, views are expected to return an HttpResponse object, but in your update view, you're not returning anything explicitly in the case where the form is not valid.

To fix this issue, you should ensure that your update view returns an appropriate response for both valid and invalid form submissions. Here's how you can do it:

