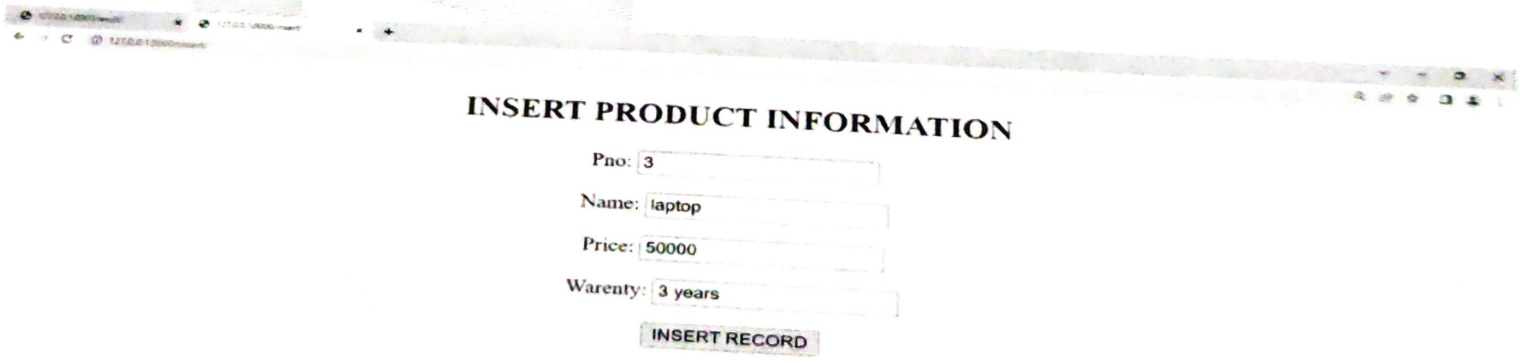
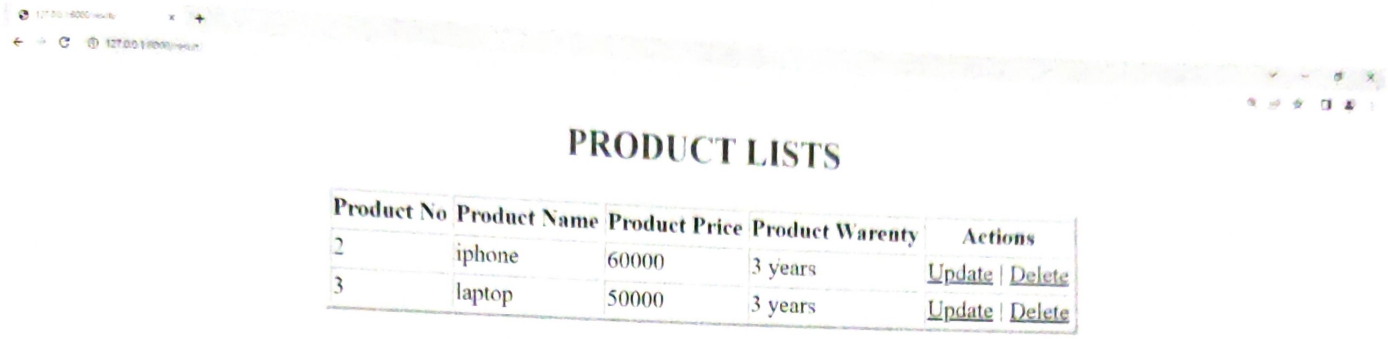
13.Configuring **update** page.

update.html

```
<!DOCTYPE html>
<html>
<center>
<body>
<h2 style="color: red">UPDATE PRODUCT INFORMATION</h2>
<form method="POST" style="color: blue">
    {%csrf_token%}
    PRODUCT NO: <input type="text" name="NO" value="{{p.pno}}"><br>
    PRODUCT NAME: <input type="text" name="name" value="{{p.name}}"><br>
    PRODUCT PRICE: <input type="text" name="price" value="{{p.price}}"><br>
    PRODUCT WARENTY: <input type="text" name="warenty" value="{{p.warenty}}"><br>

    <input type="submit" value="UPDATE RECORD">
</form>
</body>
</center>
</html>
```

OUTPUT:





PRODUCT LISTS

Product No	Product Name	Product Price	Product Warenty	Actions
2	iphone	60000	3 years	Update | Delete
3	laptop	50000	3 years	Update | Delete

UPDATE PRODUCT INFORMATION

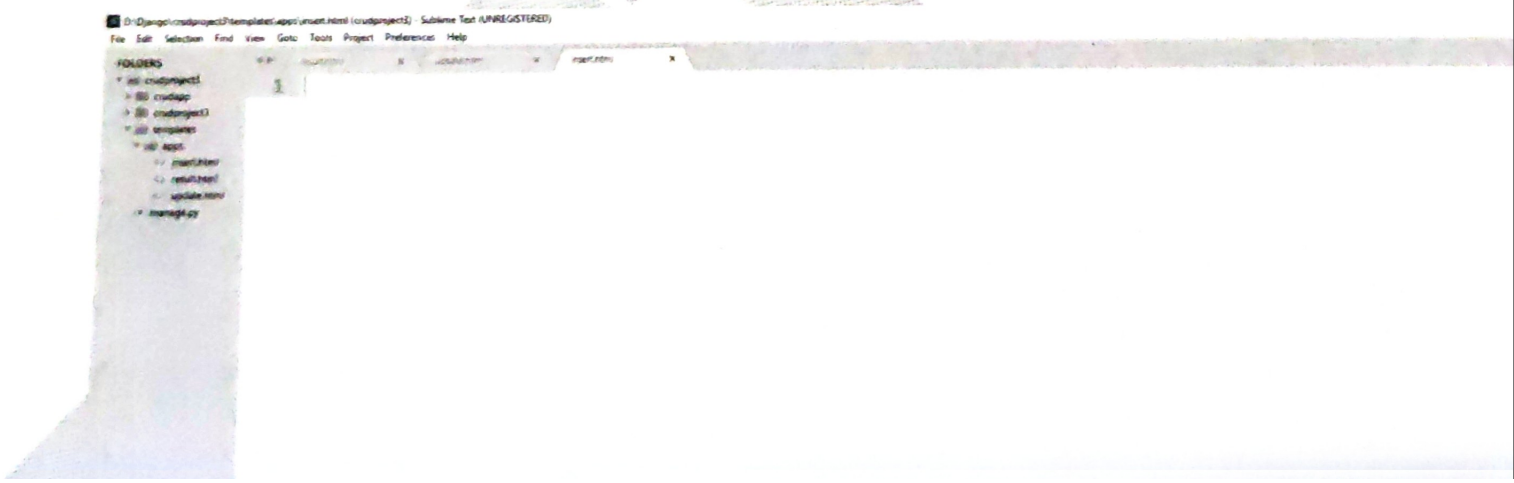
PRODUCT NO: 2
PRODUCT NAME: iphone
PRODUCT PRICE: 60000
PRODUCT WARENTY: 3 years
UPDATE RECORD

3. PROJECT : CREATE A BANK WEB APPLICATION

1. Create **Project and Application**

```
C:\Windows\system32\cmd.exe
Microsoft Windows [Version 10.0.16299.1087]
(c) 2017 Microsoft Corporation. All rights reserved.

C:\Users\systech>d:

D:\>cd Django

D:\Django>django-admin startproject crudproject3

D:\Django>
D:\Django>cd crudproject3

D:\Django\crudproject3>python manage.py startapp crudapp

D:\Django\crudproject3>
```

2. Create **Templates, apps folder and create three html page.**



3. Add **app name and template directory** in Settings file

settings.py

```
from pathlib import Path
import os
# Build paths inside the project like this: BASE_DIR / 'subdir'.
BASE_DIR = Path(__file__).resolve().parent.parent
T_dir=os.path.join(BASE_DIR,'templates')


# Quick-start development settings - unsuitable for production
# See https://docs.djangoproject.com/en/4.0/howto/deployment/checklist/

# SECURITY WARNING: keep the secret key used in production secret!
SECRET_KEY = 'django-insecure-fa_ogrw*j2qn)8duwf)0aeth%bg&5ek*2s(*93's-z61$68yw4'

# 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',
    'crudapp'
]


TEMPLATES = [
    {
        'BACKEND': 'django.template.backends.django.DjangoTemplates',
        'DIRS': [T_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',
            ],
        },
    },
]
```

4. Create **Database** in model file

models.py

```
from django.db import models

class Bank(models.Model):
    acno=models.IntegerField()
    name=models.CharField(max_length=64)
    depamnt=models.IntegerField()
    bal=models.IntegerField()
```

5. Type command : **python manage.py makemigrations**

Python manage.py migrate

```
D:\Django\crudproject3>python manage.py makemigrations
Migrations for 'crudapp':
  crudapp\migrations\0001_initial.py
    - Create model Bank

D:\Django\crudproject3>python manage.py migrate
Operations to perform:
  Apply all migrations: admin, auth, contenttypes, crudapp, 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 crudapp.0001_initial... OK
  Applying sessions.0001_initial... OK

D:\Django\crudproject3>
```

6. Type command : **python manage.py createsuperuser.**

```
C:\Windows\system32\cmd.exe
D:\Django\crudproject3>python manage.py createsuperuser
Username (leave blank to use 'systech'): 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.

D:\Django\crudproject3>
```

7. Configuring **admin** file.

admin.py

```
from django.contrib import admin
from crudapp.models import Bank
class BankAdmin(admin.ModelAdmin):
    list_display=['acno','name','depamnt','bal']
admin.site.register(Bank,BankAdmin)
```

8. Creating **forms.py**

forms.py

```
from django import forms
from crudapp.models import Bank

class BankForm(forms.ModelForm):
    class Meta:
        model=Bank
        fields='__all__'
```

9. Configuring **url** file

urls.py

```
from django.contrib import admin
from django.urls import path
from crudapp import views


urlpatterns = [
    path('admin/', admin.site.urls),
    path('result/', views.read),
    path('insert/', views.insert),
    path('delete/<int:id>', views.delete),
    path('update/<int:id>', views.update)
]
```

10.Configuring **View** file

views.py

```
from django.shortcuts import render,redirect
from crudapp.models import Bank
from crudapp.forms import BankForm
def read(request):
    bank=Bank.objects.all()
    return render(request,'apps/result.html',{'b':bank})
def insert(request):
    form=BankForm()
    if request.method=="POST":
        form=BankForm(request.POST)
        if form.is_valid():
            form.save()
    return render(request,'apps/insert.html',{'form':form})
def delete(request,id):
    b=Bank.objects.get(id=id)
    b.delete()
    return redirect('/result')
def update(request,id):
    bank=Bank.objects.get(id=id)
    if request.method=="POST":
        form=BankForm(request.POST, instance=bank)
        if form.is_valid():
            form.save()
        return redirect('/result')
    return render(request,'apps/update.html',{'b':bank})
```

11.Creating **Inserting html file.**

```
<!DOCTYPE html>
<html>
<head><h1 style="text-align: center;color: blue;font-size: x-large;">SYSTECH GROUP</h1></head
<style>
    table{
        border-color: white; border-spacing: 10px;}
    th{color: blue;border-color: white;}</style><center>
<body><h2 style="text-align: center;color: red">PRODUCT LISTS</h2>
<table border='2'>
<thead>
    <th>ACCOUNT NO</th>
    <th>HOLDER NAME</th>
    <th>DEPOSIT AMOUNT</th>
    <th>AVAILABLE AMOUNT</th>
    <th>UPDATE</th>
    <th>DELETE</th>
</thead>
{%for i in b%}
<tr>
    <td>{{i.acno}}</td>
    <td>{{i.name}}</td>
    <td>{{i.depamnt}}</td>
    <td>{{i.bal}}</td>
    <td><a href="/update/{{ i.id }}">Click</a> </td>
    <td><a href="/delete/{{ i.id }}">Click</a></td>
</tr>
{%endfor%}
</table></body></center></html>
```

12. Configuring **Result page.**

insert.html

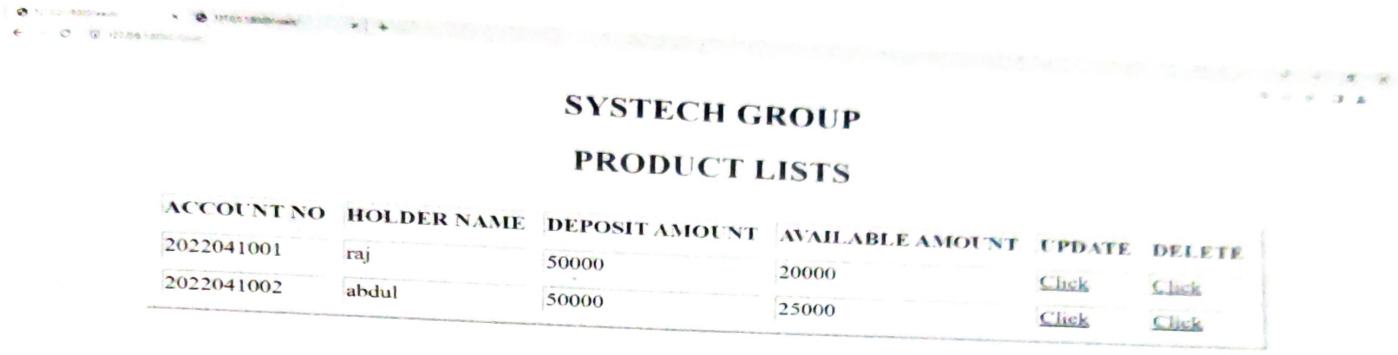
```
<!DOCTYPE html>
<html>
<head><h1 style="text-align: center;color: blue;font-size: x-large;">SYSTECH GROUP</h1></head>
<style>
    table{
        border-color: white; border-spacing: 10px;}
    th{
        color: blue;
        border-color: white;}</style>
 <center>
 <body>
 <h2 style="color: red;text-align: center;">INSERT RECORD</h2>
 <form method="POST" style="color: blue;text-align: justify;">
    {{form.as_p}}
    {%csrf_token%}
    <input type="submit" value="SUBMIT">
 </form>
 </body>
 </center>
 </html>
```

13. Configuring **update page.**

update.html

```
<!DOCTYPE html>
<html>
<head><h1 style="text-align: center;color: blue;font-size: x-large;">SYSTECH GROUP</h1></
<style>
    table{
        border-color: white; border-spacing: 10px;}
    th{
        color: blue;
        border-color: white;}</style>
<center>
<body>
<h2 style="color: red">UPDATE RECORDS</h2>
<form method="POST" style="color: blue">
    {%csrf_token%}
    ACCOUNT NO: <input type="text" name="acno" value="{{b.acno}}"><br>
    HOLDER NAME: <input type="text" name="name" value="{{b.name}}"><br>
    DEPOSIT AMOUNT: <input type="text" name="depamnt" value="{{b.depamnt}}"><br>
    AVAILABLE AMOUNT: <input type="text" name="bal" value="{{b.bal}}"><br>

    <input type="submit" value="UPDATE RECORD">
</form>
</body>
</center>
</html>
```

OUTPUT:



SYSTECH GROUP

PRODUCT LISTS

ACCOUNT NO	HOLDER NAME	DEPOSIT AMOUNT	AVAILABLE AMOUNT	UPDATE	DELETE
2022041001	raj	50000	20000	Click	Click
2022041002	abdul	50000	25000	Click	Click

SYSTECH GROUP

INSERT RECORD

Acno: 2022041002

Name: abdul

Depamnt: 50000

Bal: 25000

SUBMIT

4. PROJECT : CREATE AN EMPLOYEE WEB APPLICATION

1. Create Project and Application

C:\Windows\system32\cmd.exe

```
D:\Django>django-admin startproject crudproject4

D:\Django>cd crudproject4

D:\Django\crudproject4>python manage.py startapp crudapp

D:\Django\crudproject4>
```

2. Create templates, apps folder and create three html page.
3. Add **app name and template** directory in Settings file

```
from pathlib import Path
import os
# Build paths inside the project like this: BASE_DIR / 'subdir'.
BASE_DIR = Path(__file__).resolve().parent.parent
T_dir=os.path.join(BASE_DIR,'templates')

# Quick-start development settings - unsuitable for production
# See https://docs.djangoproject.com/en/4.0/howto/deployment/checklist/

# SECURITY WARNING: keep the secret key used in production secret!
SECRET_KEY = 'django-insecure-k+*x7^!1(qlhuxw8b+n&ehylzm!e^0153nl&190*&*fifddubx'

# 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',
    'crudapp'
]
```

```
TEMPLATES = [
    {
        'BACKEND': 'django.template.backends.django.DjangoTemplates',
        'DIRS': [T_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',
            ],
        },
    },
]
```

4. Create **Database in model** file

models.py

```
from django.db import models
class Employee(models.Model):
    no=models.IntegerField()
    name=models.CharField(max_length=64)
    sal=models.IntegerField()
    exp=models.CharField(max_length=100)
```

5. Type command : **python manage.py makemigrations**

Python manage.py migrate

C:\Windows\system32\cmd.exe

```
D:\Django\crudproject4>python manage.py makemigrations
Migrations for 'crudapp':
  crudapp\migrations\0001_initial.py
    - Create model Employee

D:\Django\crudproject4>python manage.py migrate
Operations to perform:
  Apply all migrations: admin, auth, contenttypes, crudapp, 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 crudapp.0001_initial... OK
  Applying sessions.0001_initial... OK

D:\Django\crudproject4>
```

6. Type command : **python manage.py createsuperuser.**

```
C:\Windows\system32\cmd.exe
D:\Django\crudproject4>python manage.py createsuperuser
Username (leave blank to use 'systech'): 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.

D:\Django\crudproject4>
```

7. Configuring **admin** file.

admin.py

```
from django.contrib import admin
from crudapp.models import Employee
class EmployeeAdmin(admin.ModelAdmin):
    list_display=['no','name','sal','exp']
admin.site.register(Employee,EmployeeAdmin)
```

8. Creating **forms.py**

forms.py

```
from django import forms
from crudapp.models import Employee
class EmployeeForm(forms.ModelForm):
    class Meta:
        model=Employee
        fields='__all__'
```

9. Configuring **url** file

urls.py

```
from django.contrib import admin
from django.urls import path
from crudapp import views
urlpatterns = [
    path('admin/', admin.site.urls),
    path('result/', views.read),
    path('insert/', views.insert),
    path('delete/<int:id>', views.delete),
    path('update/<int:id>', views.update)
]
```

10.Configuring **View** file

views.py

```
from django.shortcuts import render,redirect
from crudapp.models import Employee
from crudapp.forms import EmployeeForm
def read(request):
    employee=Employee.objects.all()
    return render(request,'apps/result.html',{'e':employee})
def insert(request):
    form=EmployeeForm()
    if request.method=="POST":
        form=EmployeeForm(request.POST)
        if form.is_valid():
            form.save()
    return render(request,'apps/insert.html',{'form':form})
def delete(request,id):
    e=Employee.objects.get(id=id)
    e.delete()
    return redirect('/result')
def update(request,id):
    employee=Employee.objects.get(id=id)
    if request.method=="POST":
        form=EmployeeForm(request.POST, instance=employee)
        if form.is_valid():
            form.save()
        return redirect('/result')
    return render(request,'apps/update.html',{'e':employee})
```

11. Creating **Inserting html** file.

```
<!DOCTYPE html>
<html>
<style>
    table{
        border-color: white;}
    th{
        color: blue;
        border-color: white;}</style>
<center><body>
<h2 style="text-align: center;color: red">Employee Details</h2>
<table border='2'>
<thead>
    <th>Employee IDNo</th>
    <th>Employee Name</th>
    <th>Salary</th>
    <th>Experience</th>
    <th>Actions</th>
</thead>
{%for i in e%}
<tr>
    <td>{{i.no}}</td>
    <td>{{i.name}}</td>
    <td>{{i.sal}}</td>
    <td>{{i.exp}}</td>
    <td><a href="/update/{{ i.id }}">Update</a> | <a href="/delete/{{ i.id }}">Delete</a></td>
</tr>
{%endfor%}
</table></body></center></html>
```

12.Configuring **Result** page.

insert.html x

```
<!DOCTYPE html>
<html>


<center>
<body>
<h2 style="color: red">INSERT EMPLOYEE INFORMATION</h2>
<form method="POST" style="color: blue">
    {{form.as_p}}
    {%csrf_token%}
    <input type="submit" value="SUBMIT">
</form>
</body>
</center>
</html>
```

13.Configuring **update** page.

update.html

```
<!DOCTYPE html>
<html>
<center>
<body>
<h2 style="color: red">UPDATE INFORMATION</h2>
<form method="POST" style="color: blue">
    {%csrf_token%}
    EMPLOYEE NO: <input type="text" name="no" value="{{e.no}}"><br>
    EMPLOYEE NAME: <input type="text" name="name" value="{{e.name}}"><
    SALARY: <input type="text" name="sal" value="{{e.sal}}"><br>
    EXPERIENCE: <input type="text" name="exp" value="{{e.exp}}"><br>

    <input type="submit" value="UPDATE">
</form>
</body>
</center>
</html>
```

OUTPUT:

Employee Details

Employee IDNo	Employee Name	Salary	Experience	Actions
2	raj	30000	4 years	Update Delete
103	arjun	50000	10 years	Update Delete

INSERT EMPLOYEE INFORMATION

No:

Name:

Sal:

Exp:

SUBMIT

UPDATE INFORMATION

EMPLOYEE NO: 103

EMPLOYEE NAME: arjun

SALARY: 50000

EXPERIENCE: 10 years

UPDATE

5. PROJECT : CREATE A LIBRARY WEB APPLICATION

1. Create Project and Application

Administrator: Command Prompt

```
C:\Windows\system32>d:

D:\>cd Django

D:\Django>django-admin startproject crudproject5

D:\Django>cd crudproject5

D:\Django\crudproject5>python manage.py startapp crudapp

D:\Django\crudproject5>
```

2. Create templates, apps folder and create three html page.
3. Add **app name and template directory** in Settings file

settings.py

```
from pathlib import Path
import os
# Build paths inside the project like this: BASE_DIR / 'subdir'.
BASE_DIR = Path(__file__).resolve().parent.parent
T_dir=os.path.join(BASE_DIR,'templates')

# Quick-start development settings - unsuitable for production
# See https://docs.djangoproject.com/en/4.0/howto/deployment/checklist/

# SECURITY WARNING: keep the secret key used in production secret!
SECRET_KEY = 'django-insecure-+ryc=h9)g#h9+(qc9gng70-k=57flpq9givmuu1r05_i@k@&29'

# 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',
    'crudapp'
]
```

```
TEMPLATES = [
    {
        'BACKEND': 'django.template.backends.django.DjangoTemplates',
        'DIRS': [T_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',
            ],
        },
    },
]
```

4. Create **Database in model file**

models.py

```
from django.db import models
class Book(models.Model):
    no=models.IntegerField()
    name=models.CharField(max_length=64)
    year=models.IntegerField()
    price=models.IntegerField()
```

5. Type command : **python manage.py makemigrations**

Python manage.py migrate

Administrator: Command Prompt

```
D:\Django\crudproject5>python manage.py makemigrations
Migrations for 'crudapp':
  crudapp\migrations\0001_initial.py
    - Create model Book

D:\Django\crudproject5>python manage.py migrate
Operations to perform:
  Apply all migrations: admin, auth, contenttypes, crudapp, 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 crudapp.0001_initial... OK
  Applying sessions.0001_initial... OK

D:\Django\crudproject5>
```

6. Type command : **python manage.py createsuperuser.**

Administrator: Command Prompt

```
D:\Django\crudproject5>python manage.py createsuperuser
Username (leave blank to use 'psipc2'): 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.

D:\Django\crudproject5>
```

7. Configuring **admin** file.

admin.py

```
from django.contrib import admin
from crudapp.models import Book
class BookAdmin(admin.ModelAdmin):
    list_display=['no','name','year','price']
admin.site.register(Book,BookAdmin)
```

8. Creating **forms.py**

forms.py

```
from django import forms
from crudapp.models import Book

class BookForm(forms.ModelForm):
    class Meta:
        model=Book
        fields='__all__'
```

9. Configuring **url** file

urls.py

```
from django.contrib import admin
from django.urls import path
from crudapp import views
urlpatterns = [
    path('admin/', admin.site.urls),
    path('result/', views.read),
    path('insert/', views.insert),
    path('delete/<int:id>', views.delete),
    path('update/<int:id>', views.update)
]
```

10. Configuring **View** file

views.py

```
from django.shortcuts import render,redirect
from crudapp.models import Book
from crudapp.forms import BookForm
def read(request):
    book=Book.objects.all()
    return render(request,'apps/result.html',{'b':book})
def insert(request):
    form=BookForm()
    if request.method=="POST":
        form=BookForm(request.POST)
        if form.is_valid():
            form.save()
    return render(request,'apps/insert.html',{'form':form})
def delete(request,id):
    b=Book.objects.get(id=id)
    b.delete()
    return redirect('/result')
def update(request,id):
    book=Book.objects.get(id=id)
    if request.method=="POST":
        form=BookForm(request.POST, instance=book)
        if form.is_valid():
            form.save()
        return redirect('/result')
    return render(request,'apps/update.html',{'b':book})
```

11.Creating **Inserting html file.**

```
<!DOCTYPE html>
<html>
<head><h1 style="text-align: center;color: blue;font-size: x-large;">SYSTECH GROUP</h1></head>
<style>
    table{
        border-color: white; border-spacing: 10px;}
    th{
        color: blue;
        border-color: white;}</style>
<center>
<body>
<h2 style="color: red;text-align: center;">INSERT RECORD</h2>
<form method="POST" style="color: blue;text-align: justify;">
    {{form.as_p}}
    {%csrf_token%}
    <input type="submit" value="SUBMIT">
</form>
</body>
</center>
</html>
```

12.Configuring **Result** page.

```
<!DOCTYPE html>
<html>
<head><h1 style="text-align: center;color: blue;font-size: x-large;">SYSTECH GROUP</h1></head>
<style>
    table{
        border-color: white; border-spacing: 10px;}
    th{color: blue;border-color: white;}</style><center>
<body><h2 style="text-align: center;color: red">BOOKS LISTS</h2>
<table border='2'>
<thead>
    <th>BOOK NO</th>
    <th>BOOK NAME</th>
    <th>YEAR OF PUBLICATION</th>
    <th>PRICE</th>
    <th>UPDATE</th>
    <th>DELETE</th>
</thead>
{%for i in b%}
<tr>
    <td>{{i.no}}</td>
    <td>{{i.name}}</td>
    <td>{{i.year}}</td>
    <td>{{i.price}}</td>
    <td><a href="/update/{{ i.id }}">Click</a> </td>
    <td><a href="/delete/{{ i.id }}">Click</a></td>
</tr>
{%endfor%}
</table></body></center></html>
```

13.Configuring **update** page.

```
<!DOCTYPE html>
<html>
<head><h1 style="text-align: center;color: blue;font-size: x-large;">SYSTECH GROUP</h1></head>
<style>
    table{
        border-color: white; border-spacing: 10px;}
    th{
        color: blue;
        border-color: white;}</style>
<center>
<body>
<h2 style="color: red">UPDATE RECORDS</h2>
<form method="POST" style="color: blue">
    {%csrf_token%}
    BOOK NO: <input type="text" name="no" value="{{b.no}}"><br>
    BOOK NAME: <input type="text" name="name" value="{{b.name}}"><br>
    YEAR: <input type="text" name="year" value="{{b.year}}"><br>
    PRICE: <input type="text" name="price" value="{{b.price}}"><br>

    <input type="submit" value="UPDATE RECORD">
</form>
</body>
</center>
</html>
```

OUTPUT:

SYSTECH GROUP

INSERT RECORD

No: 106

Name: c++

Year: 2015

Price: 900

SUBMIT

SYSTECH GROUP

BOOKS LISTS

BOOK NO	BOOK NAME	YEAR OF PUBLICATION	PRICE	UPDATE	DELETE
101	python	2020	500	Click	Click
103	Django Framework	2021	560	Click	Click
105	Oops programming	2010	600	Click	Click

SYSTECH GROUP

UPDATE RECORDS

BOOK NO: 106
BOOK NAME: c++
YEAR: 2015
PRICE: 500
UPDATE RECORD