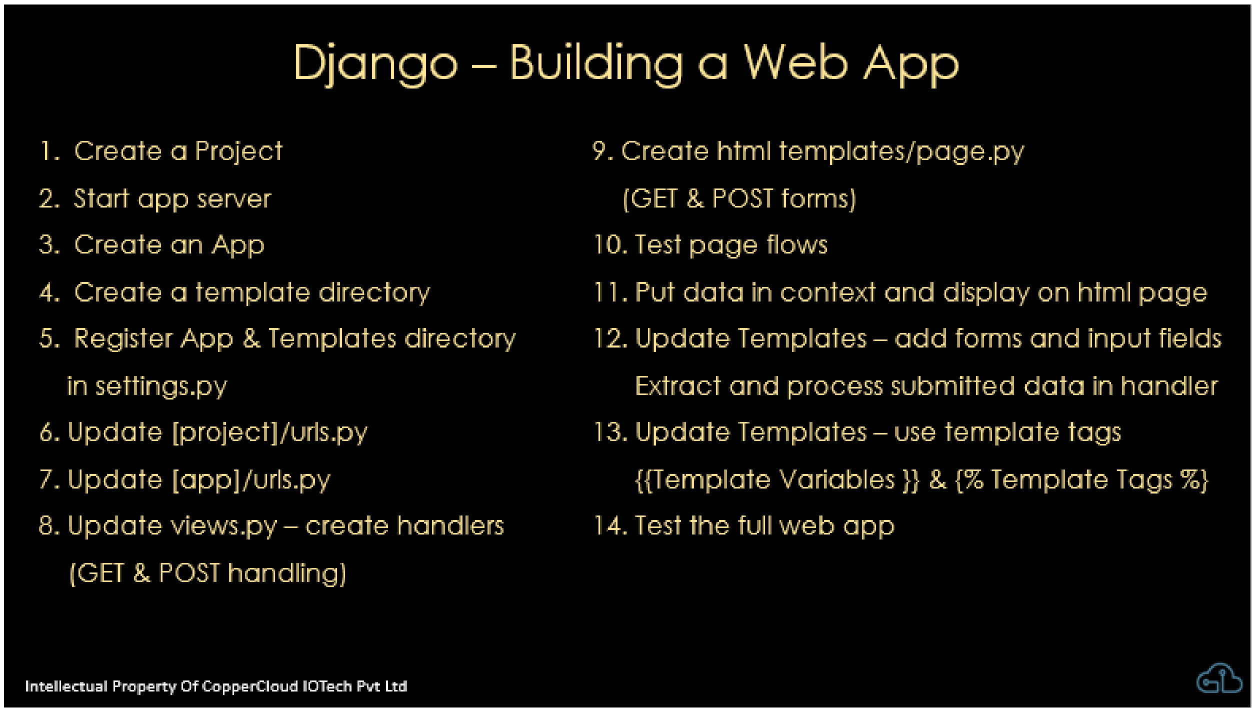
CopperCloud IOTech
Sense | Control | Automate

Cook Book for creating Django Web Apps – All Steps

* ***Note:*** *This document has been created referring to Django set-up and usage on a MicroSoft Windows environment. If you use a different OS, the steps may vary slightly. If so, please contact CopperCloud Support at support@coppercloud.in*.

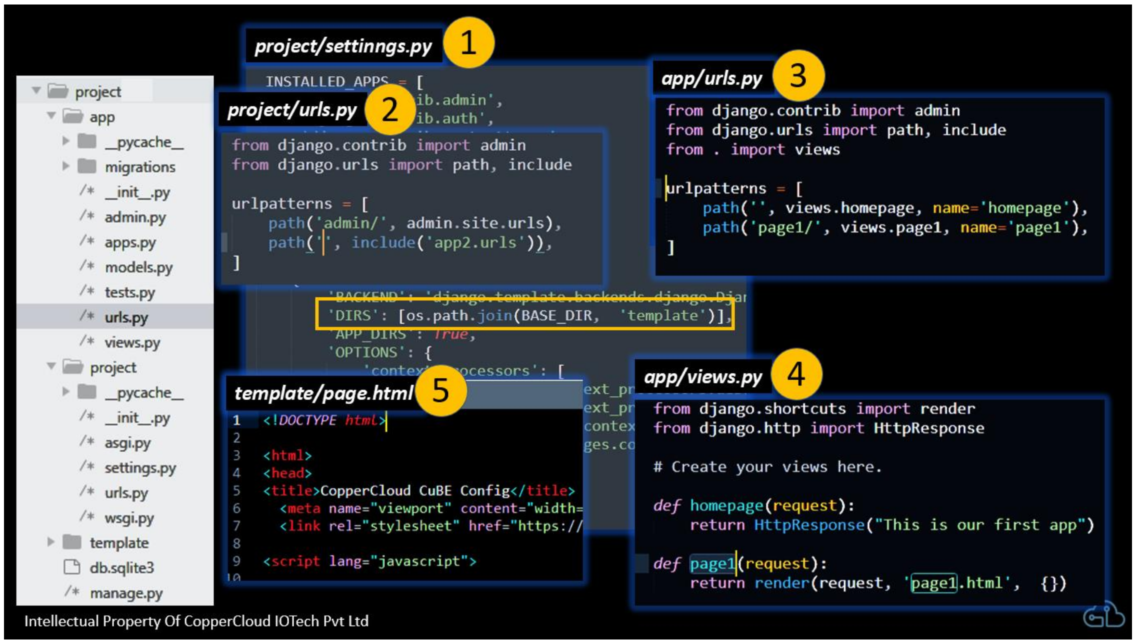


Django – Building a Web App

1. Create a Project
2. Start app server
3. Create an App
4. Create a template directory
5. Register App & Templates directory in settings.py
6. Update [project]/urls.py
7. Update [app]/urls.py
8. Update views.py – create handlers (GET & POST handling)
9. Create html templates/page.py (GET & POST forms)
10. Test page flows
11. Put data in context and display on html page
12. Update Templates – add forms and input fields
    Extract and process submitted data in handler
13. Update Templates – use template tags
    {{Template Variables }} & {% Template Tags %}
14. Test the full web app

Intellectual Property Of CopperCloud IOTech Pvt Ltd

Creating a new Django Web App:



project/settinngs.py 1

```
INSTALLED_APPS = [
    ...ib.admin',
    ...ib.auth',
...
        'DIRS': [os.path.join(BASE_DIR,  'template')],
        'APP_DIRS': True,
        'OPTIONS': {
            'context_processors': [
```

project/urls.py 2

```
from django.contrib import admin
from django.urls import path, include

urlpatterns = [
    path('admin/', admin.site.urls),
    path('', include('app2.urls')),
]
```

app/urls.py 3

```
from django.contrib import admin
from django.urls import path, include
from . import views

urlpatterns = [
    path('', views.homepage, name='homepage'),
    path('page1/', views.page1, name='page1'),
]
```

app/views.py 4

```
from django.shortcuts import render
from django.http import HttpResponse

# Create your views here.

def homepage(request):
    return HttpResponse("This is our first app")

def page1(request):
    return render(request, 'page1.html',  {})
```

template/page.html 5

```
<!DOCTYPE html>

<html>
<head>
<title>CopperCloud CuBE Config</title>
  <meta name="viewport" content="width=
  <link rel="stylesheet" href="https://

<script lang="javascript">
```

Intellectual Property Of CopperCloud IOTech Pvt Ltd

Getting started:

a. Add template directory location to project/settings.py (to be done only once for an app)
b. Add your app starting url (can be “”) to project/urls.py (to be done only once for an app)
c. Add the required flow to app/urls.py (to be done for every new flow/request in the app)
d. Add the view/handler for the new flow/request in app/views.py (to be done for every new flow/request in the app)
e. Create the template/<page>.html file

Detailed Instructions:

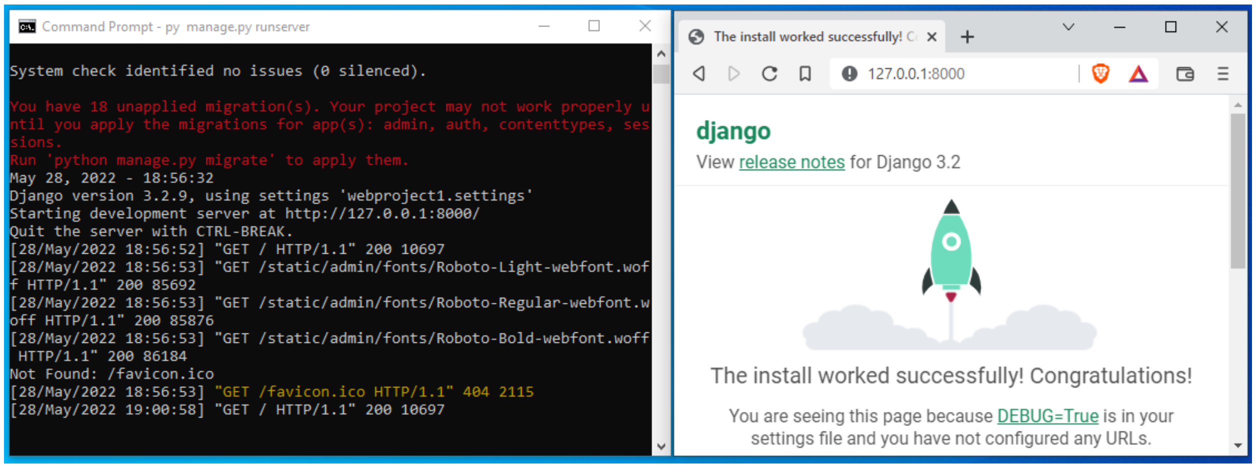
1. Create a Django Project:

```
D:\> mkdir django-projects
D:\> cd django-projects
D:\django-projects> django-admin startproject webproject
D:\django-projects\> cd webproject
D:\django-projects\webproject>
```

2. Start app server:

D:\django-projects\webproject>py manage.py runserver

Test:



```
System check identified no issues (0 silenced).

You have 18 unapplied migration(s). Your project may not work properly u
ntil you apply the migrations for app(s): admin, auth, contenttypes, ses
sions.
Run 'python manage.py migrate' to apply them.
May 28, 2022 - 18:56:32
Django version 3.2.9, using settings 'webproject1.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.
[28/May/2022 18:56:52] "GET / HTTP/1.1" 200 10697
[28/May/2022 18:56:53] "GET /static/admin/fonts/Roboto-Light-webfont.wof
f HTTP/1.1" 200 85692
[28/May/2022 18:56:53] "GET /static/admin/fonts/Roboto-Regular-webfont.w
off HTTP/1.1" 200 85876
[28/May/2022 18:56:53] "GET /static/admin/fonts/Roboto-Bold-webfont.woff
 HTTP/1.1" 200 86184
Not Found: /favicon.ico
[28/May/2022 18:56:53] "GET /favicon.ico HTTP/1.1" 404 2115
[28/May/2022 19:00:58] "GET / HTTP/1.1" 200 10697
```

View release notes for Django 3.2

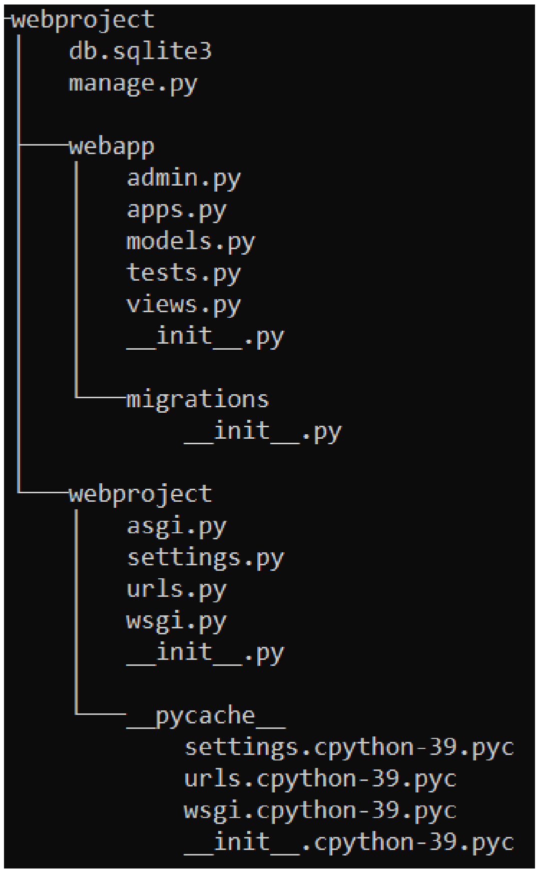
The install worked successfully! Congratulations!

You are seeing this page because DEBUG=True is in your settings file and you have not configured any URLs.

3. **Create an App:**

```
D:\django-projects\webproject>python manage.py startapp webapp
```

Test:

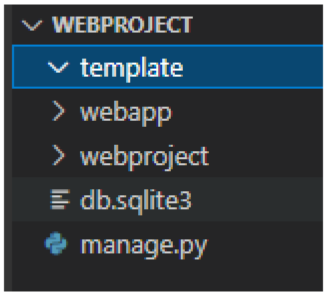


```
webproject
│   db.sqlite3
│   manage.py
│
├───webapp
│   │   admin.py
│   │   apps.py
│   │   models.py
│   │   tests.py
│   │   views.py
│   │   __init__.py
│   │
│   └───migrations
│           __init__.py
│
└───webproject
    │   asgi.py
    │   settings.py
    │   urls.py
    │   wsgi.py
    │   __init__.py
    │
    └───__pycache__
            settings.cpython-39.pyc
            urls.cpython-39.pyc
            wsgi.cpython-39.pyc
            __init__.cpython-39.pyc
```

**** At this point, you may open the project folder in an IDE of your choice. This documents uses VS Code***

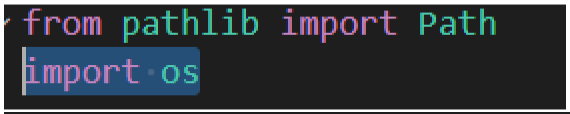
**** It is assumed that VS Code has been set up with Python extensions, and Django has been installed on the Development environment/computer***

4. **Create a 'template' directory for storing html templates/pages:**

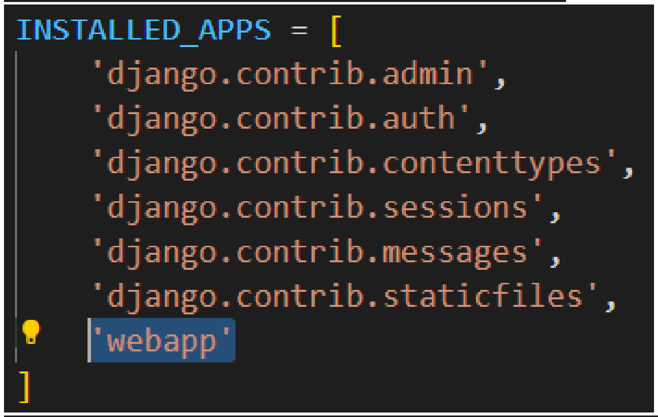


```
WEBPROJECT
  template
  webapp
  webproject
  db.sqlite3
  manage.py
```

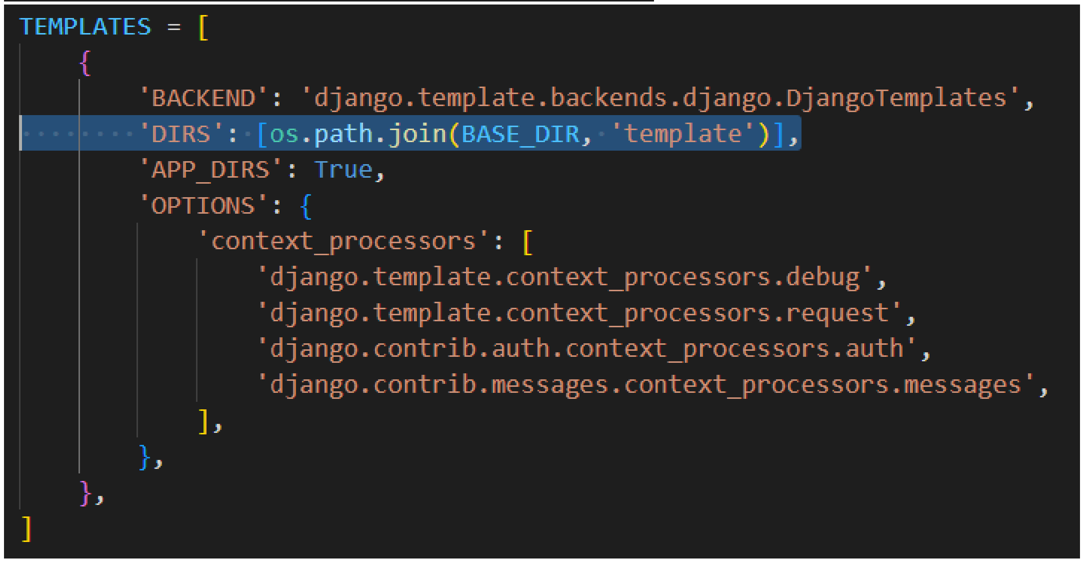
5. **Update webproject/settings.py to register new app and the template directory location:**



```
from pathlib import Path
import os
```

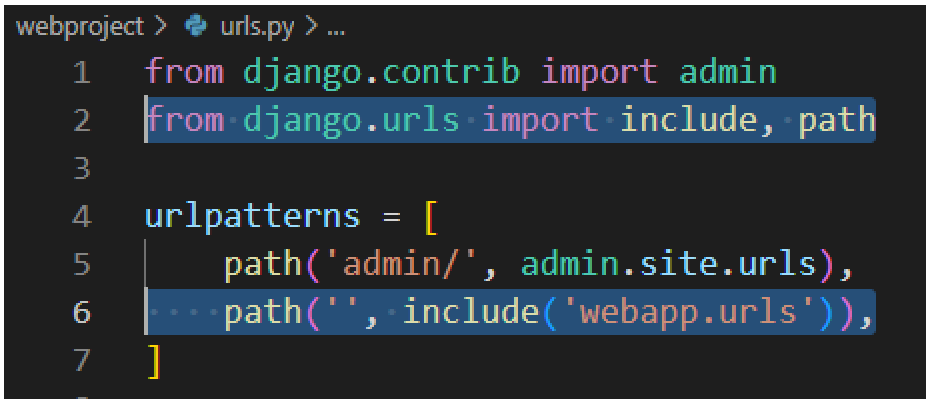


```
INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'webapp'
]
```



```
TEMPLATES = [
    {
        'BACKEND': 'django.template.backends.django.DjangoTemplates',
        'DIRS': [os.path.join(BASE_DIR, 'template')],
        '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',
            ],
        },
    },
]
```

6. **Update webproject/urls.py to add a url to the new app:**

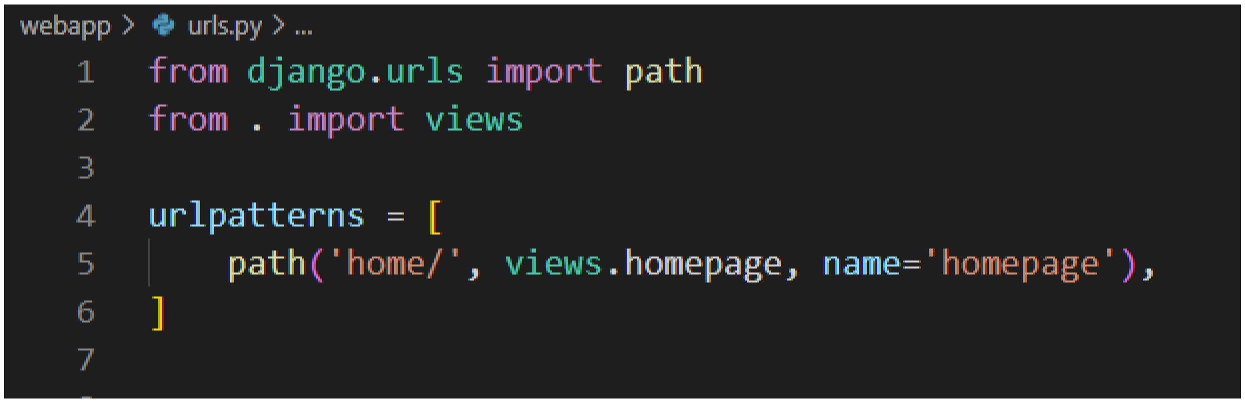


webproject > urls.py > ...

```
from django.contrib import admin
from django.urls import include, path

urlpatterns = [
    path('admin/', admin.site.urls),
    path('', include('webapp.urls')),
]
```

7. **Update webapp/urls.py to add a url to the new app:**

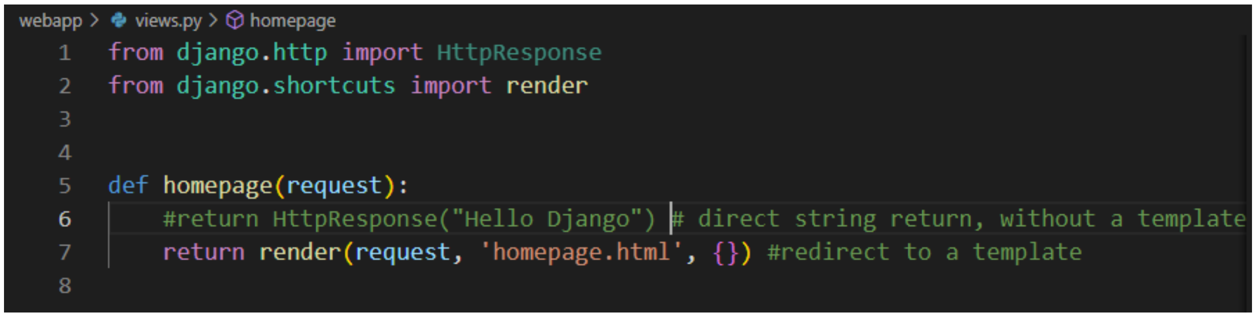
   First, create urls.py under webapp, if it doesn't exist.



webapp > urls.py > ...

```python
from django.urls import path
from . import views

urlpatterns = [
    path('home/', views.homepage, name='homepage'),
]

```

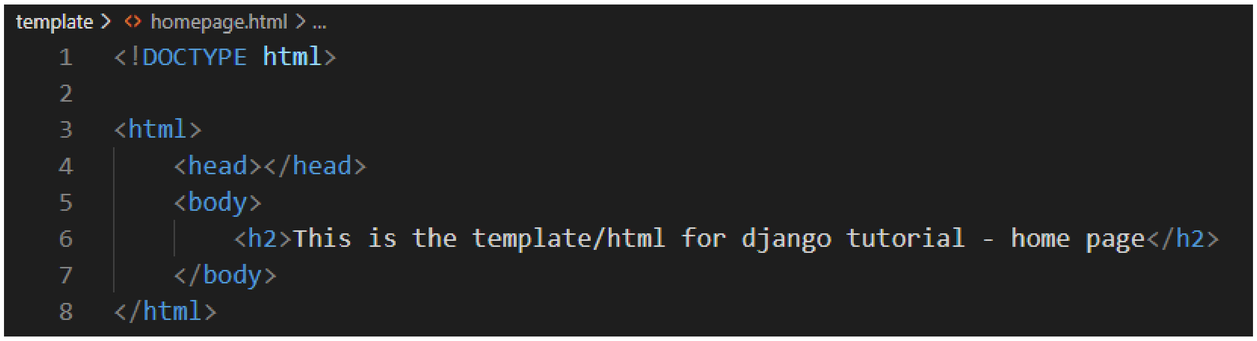
8. **Update webapp/view.py to add a handler function:**



webapp > views.py > homepage

```python
from django.http import HttpResponse
from django.shortcuts import render


def homepage(request):
    #return HttpResponse("Hello Django") # direct string return, without a template
    return render(request, 'homepage.html', {}) #redirect to a template

```

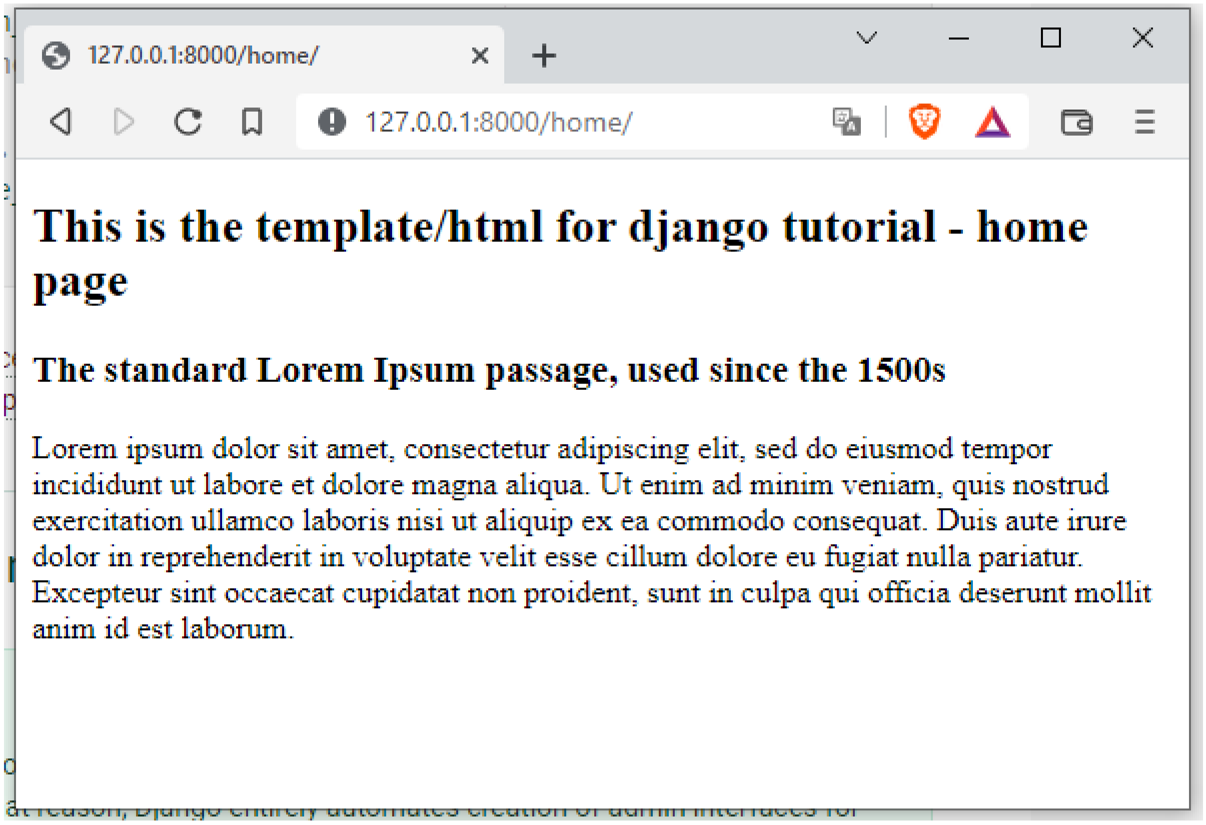
9. **Create template/<page>.html:**



template > homepage.html > ...

```html
<!DOCTYPE html>

<html>
    <head></head>
    <body>
        <h2>This is the template/html for django tutorial - home page</h2>
    </body>
</html>
```

10. **Test Page Flows:**



127.0.0.1:8000/home/

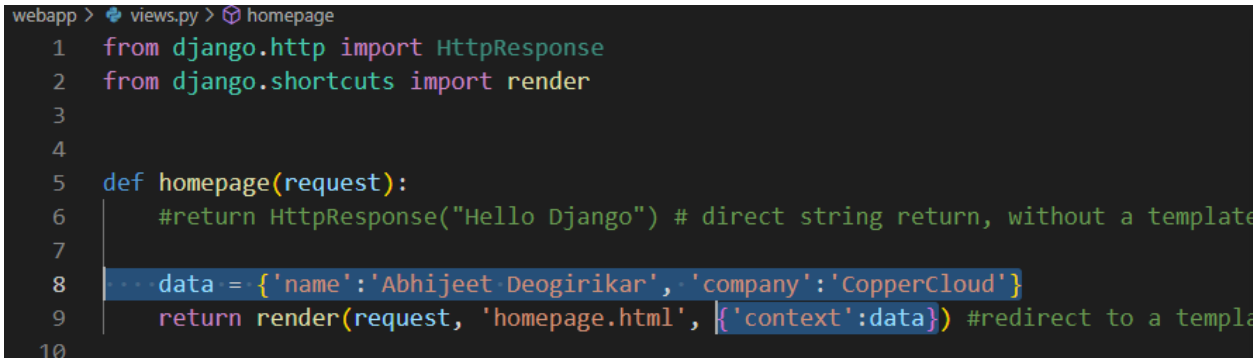
This is the template/html for django tutorial - home page

The standard Lorem Ipsum passage, used since the 1500s

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

11. **In handler, put data in context, and display on template:**

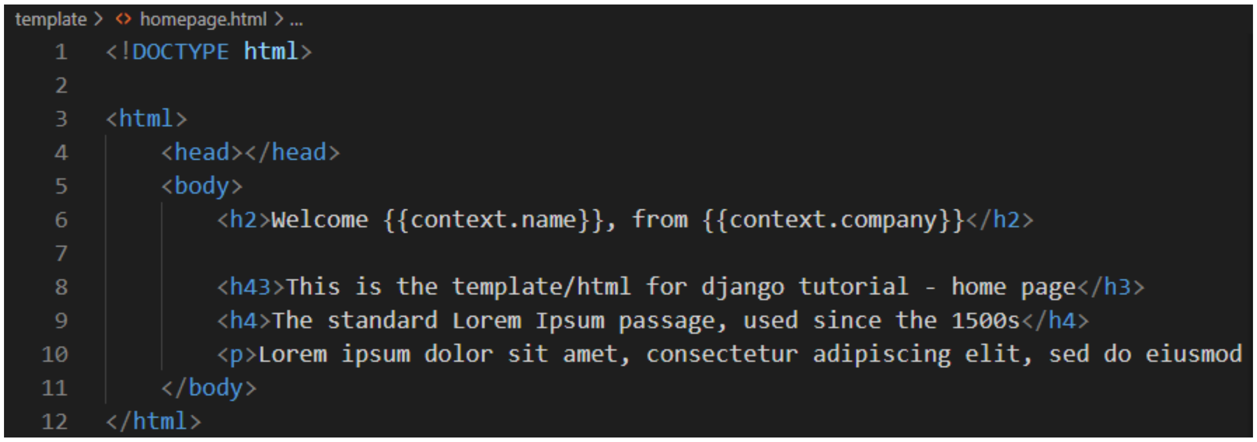
Update View:



webapp > views.py > homepage

```
from django.http import HttpResponse
from django.shortcuts import render


def homepage(request):
    #return HttpResponse("Hello Django") # direct string return, without a template

    data = {'name':'Abhijeet Deogirikar', 'company':'CopperCloud'}
    return render(request, 'homepage.html', {'context':data}) #redirect to a templa
```

Update Template:

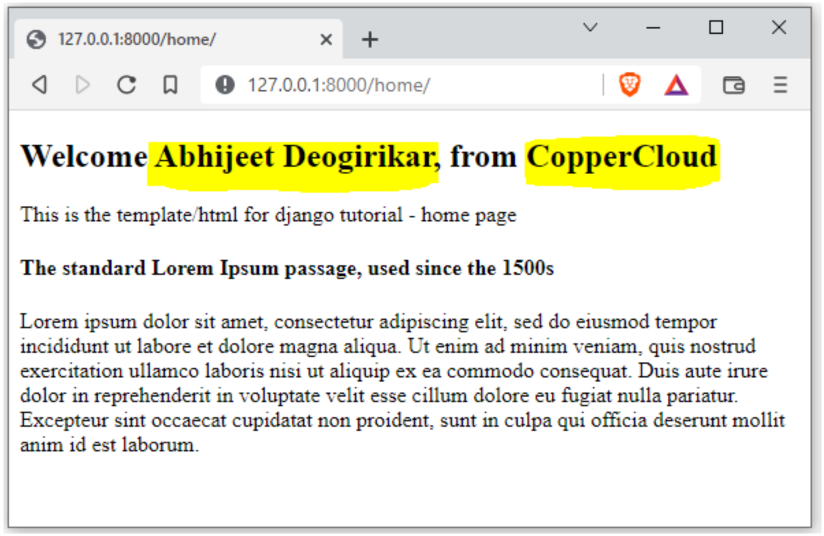


template > homepage.html > ...

```
<!DOCTYPE html>

<html>
    <head></head>
    <body>
        <h2>Welcome {{context.name}}, from {{context.company}}</h2>

        <h43>This is the template/html for django tutorial - home page</h3>
        <h4>The standard Lorem Ipsum passage, used since the 1500s</h4>
        <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod
    </body>
</html>
```

Test:

(Restart Django server if needed – due to some errors during configuration the server may have stopped)



127.0.0.1:8000/home/

Welcome Abhijeet Deogirikar, from CopperCloud

This is the template/html for django tutorial - home page

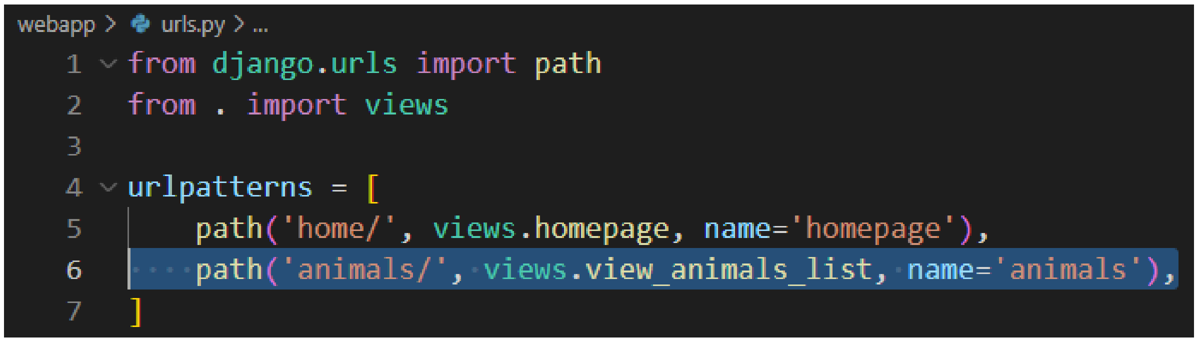
The standard Lorem Ipsum passage, used since the 1500s

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Further testing – create more flows, and show data in different ways in templates:

- ➔ Create a new flow: animals
- ➔ Add webapp/urls.py entry (for new url)
- ➔ Add webapp/views.py entry (for new handler)
- ➔ Add animals.html with template tags to iterate through animals placed in context by handler (view)
- ➔ Test

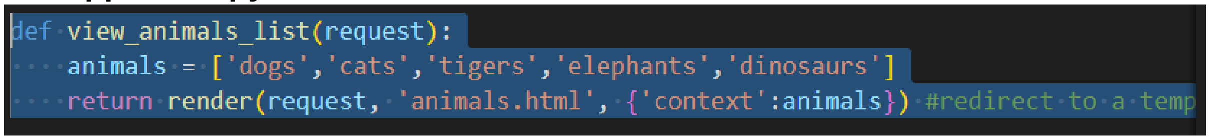
webapp/urls.py:



webapp > urls.py > ...

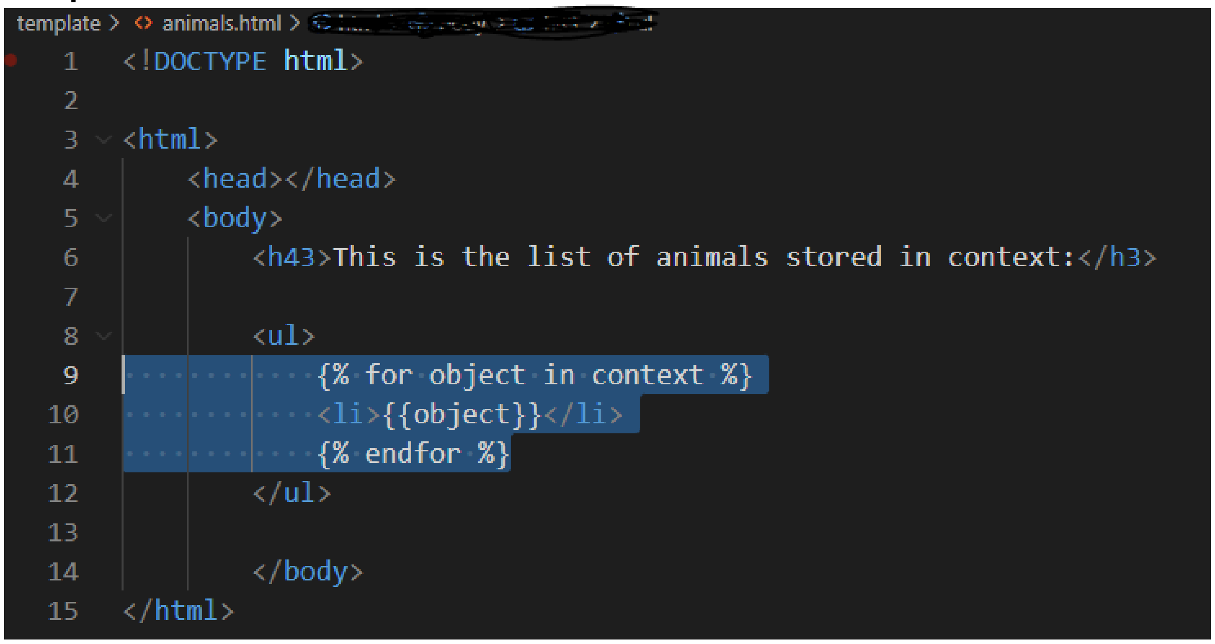
```python
from django.urls import path
from . import views

urlpatterns = [
    path('home/', views.homepage, name='homepage'),
    path('animals/', views.view_animals_list, name='animals'),
]
```

webapp/views.py:



```python
def view_animals_list(request):
    animals = ['dogs','cats','tigers','elephants','dinosaurs']
    return render(request, 'animals.html', {'context':animals}) #redirect to a temp
```

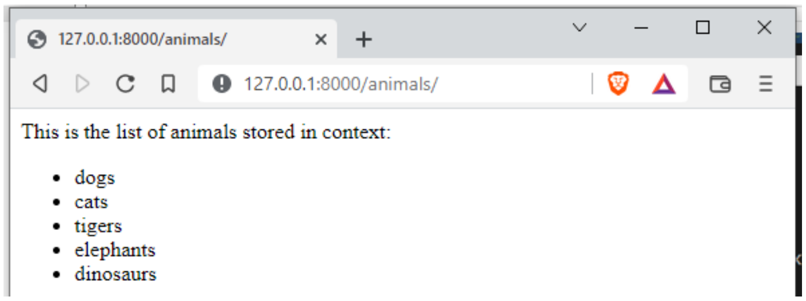
template/animals.html:



template > animals.html > [redacted]

```html
<!DOCTYPE html>

<html>
    <head></head>
    <body>
        <h43>This is the list of animals stored in context:</h3>

        <ul>
            {% for object in context %}
            <li>{{object}}</li>
            {% endfor %}
        </ul>

        </body>
</html>
```

Test:



127.0.0.1:8000/animals/

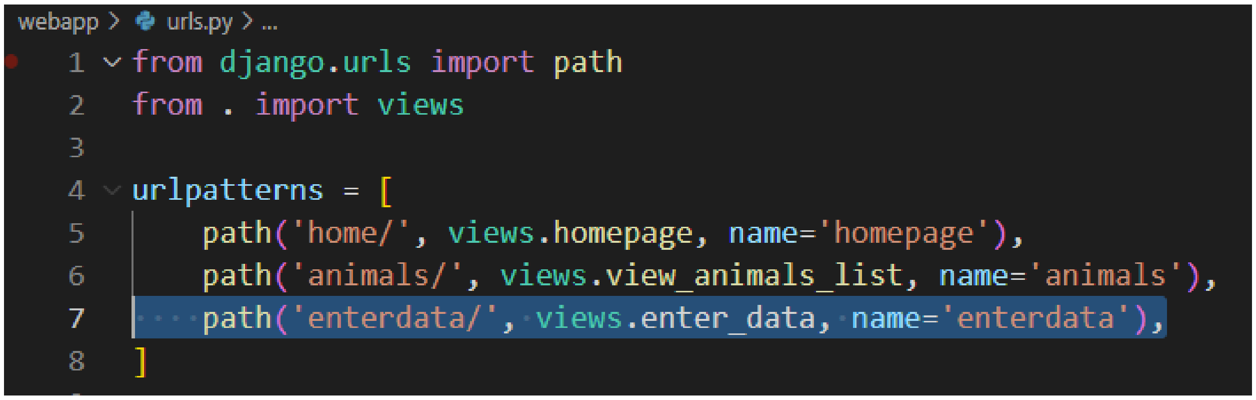
127.0.0.1:8000/animals/

This is the list of animals stored in context:

- dogs
- cats
- tigers
- elephants
- dinosaurs

12. (also 13 & 14) Send form data to server using input fields and data in the request object:

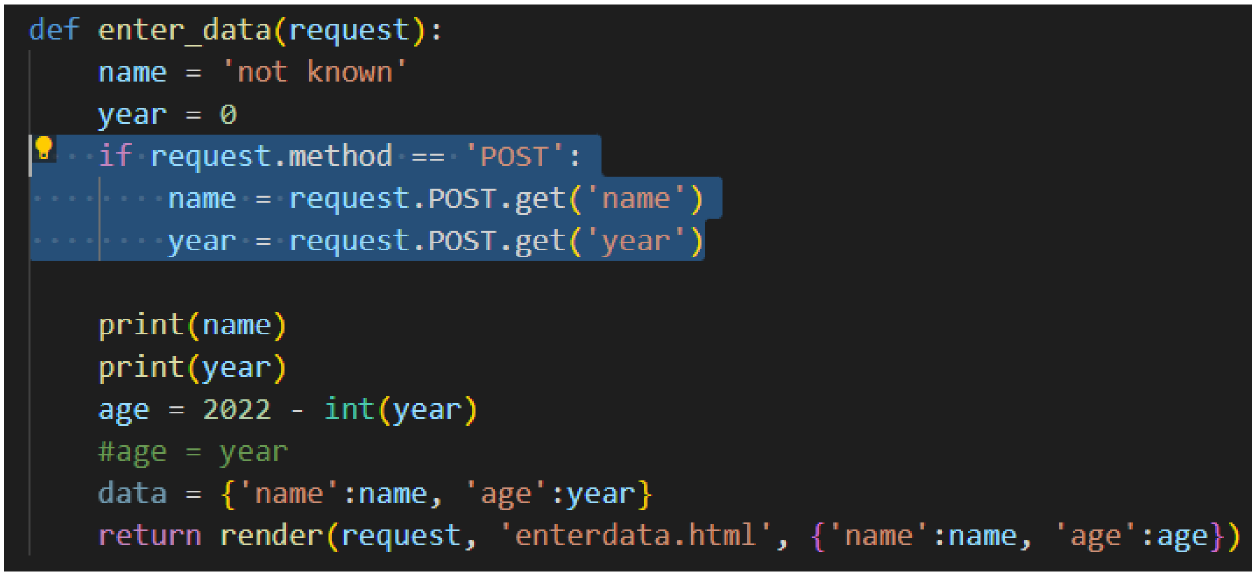
- ➔ Create a new flow: enterdata
- ➔ Add webapp/urls.py entry (for new url)
- ➔ Add webapp/views.py entry (for new handler)
- ➔ Add enterdata.html with form to submit entered data (using HTTP POST method)
- ➔ Test



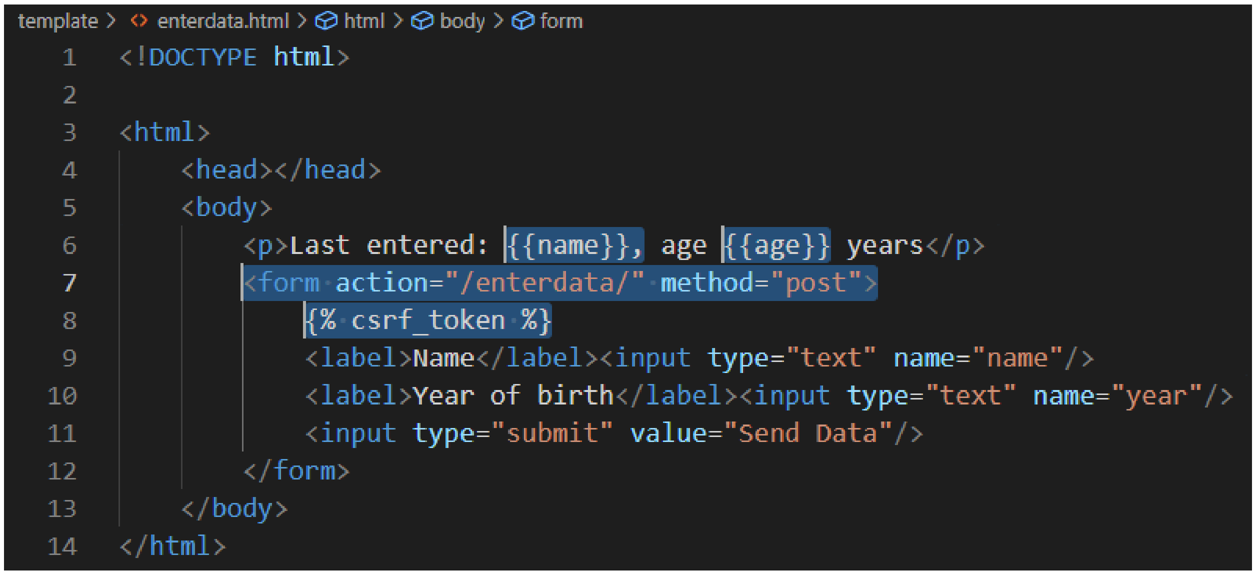
webapp > urls.py > ...

```python
from django.urls import path
from . import views

urlpatterns = [
    path('home/', views.homepage, name='homepage'),
    path('animals/', views.view_animals_list, name='animals'),
    path('enterdata/', views.enter_data, name='enterdata'),
]
```

webapp/views.py:



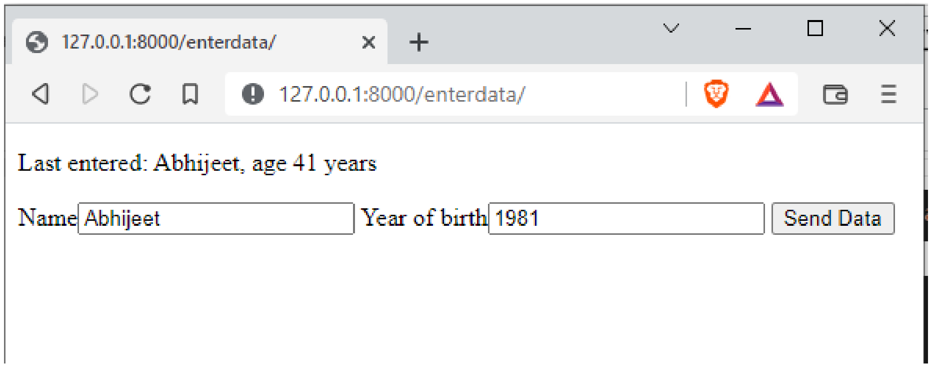
```python
def enter_data(request):
    name = 'not known'
    year = 0
    if request.method == 'POST':
        name = request.POST.get('name')
        year = request.POST.get('year')

    print(name)
    print(year)
    age = 2022 - int(year)
    #age = year
    data = {'name':name, 'age':year}
    return render(request, 'enterdata.html', {'name':name, 'age':age})
```



template > enterdata.html > html > body > form

```
<!DOCTYPE html>

<html>
    <head></head>
    <body>
        <p>Last entered: {{name}}, age {{age}} years</p>
        <form action="/enterdata/" method="post">
            {% csrf_token %}
            <label>Name</label><input type="text" name="name"/>
            <label>Year of birth</label><input type="text" name="year"/>
            <input type="submit" value="Send Data"/>
        </form>
    </body>
</html>
```

Test:



127.0.0.1:8000/enterdata/

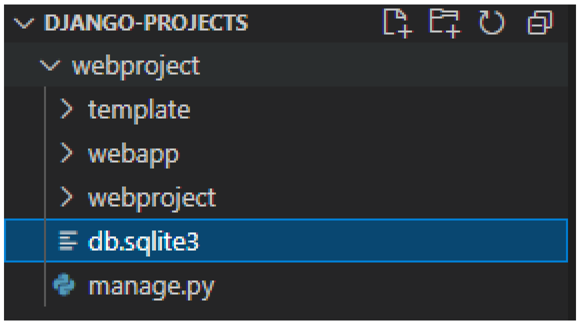
Last entered: Abhijeet, age 41 years

Name Abhijeet Year of birth 1981 Send Data

---

15. Database Integration:

The database file already exists when you create the Django Project:

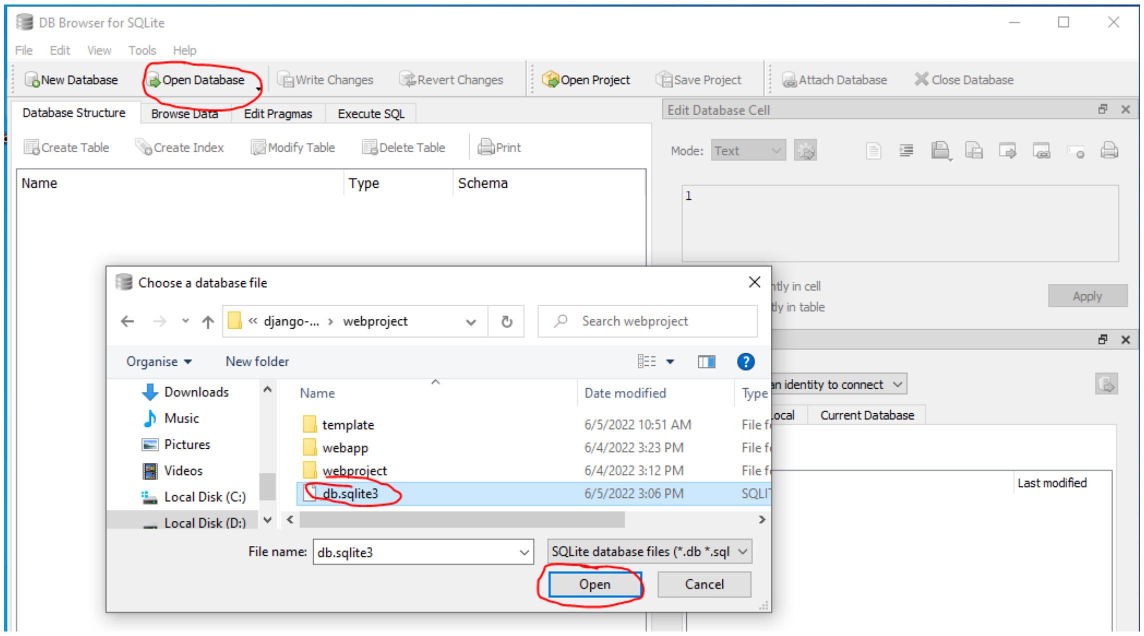


```
DJANGO-PROJECTS
  webproject
    > template
    > webapp
    > webproject
    db.sqlite3
    manage.py
```

➔ Open the SQLite DB Browser and update the db.sqlite3 database by adding a few entries.

CopperCloud IOTech
Sense | Control | Automate

➔ ***DO NOT FORGET TO "Write Changes" to the database after manually running the SQL Queries, otherwise the changes will not be saved after the SQLite Browser closes.***



Name	Date modified	Type
template	6/5/2022 10:51 AM	File f
webapp	6/4/2022 3:23 PM	File f
webproject	6/4/2022 3:12 PM	File f
db.sqlite3	6/5/2022 3:06 PM	SQLI

File name: db.sqlite3
SQLite database files (*.db *.sql
Open
Cancel

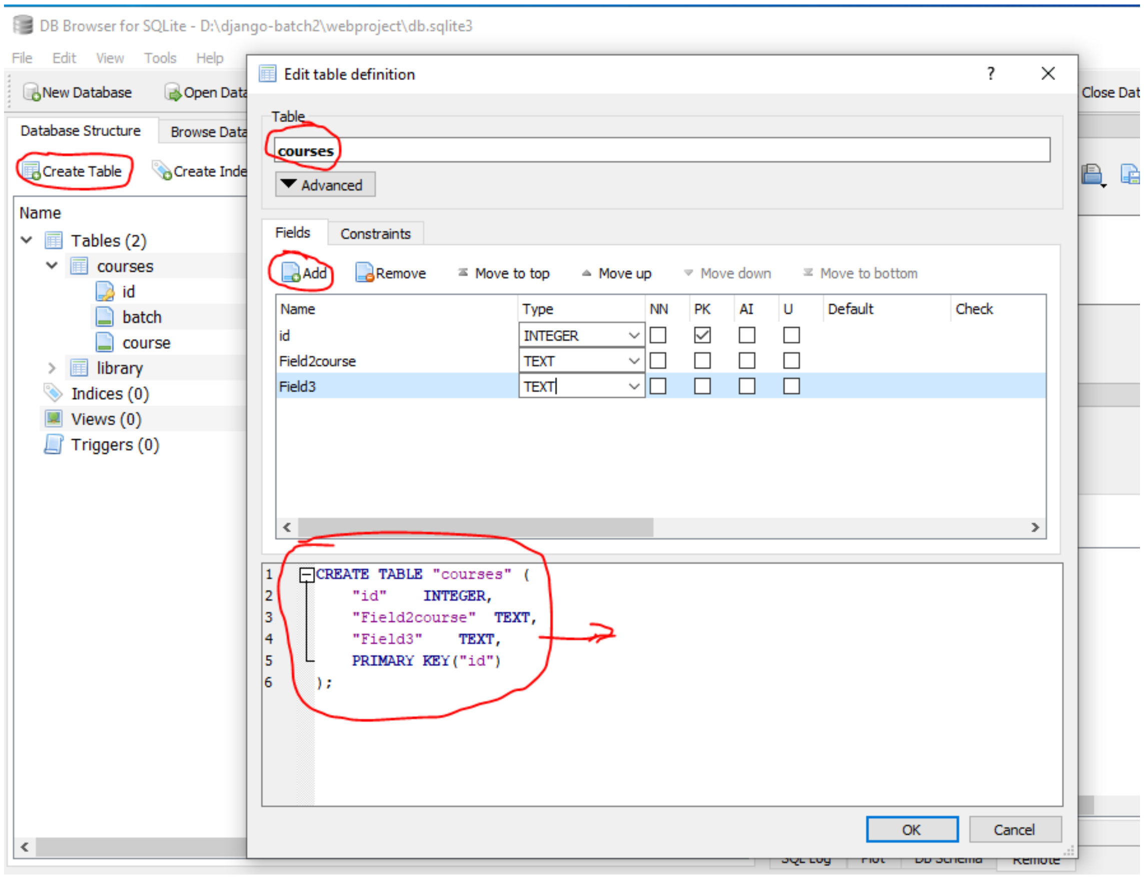
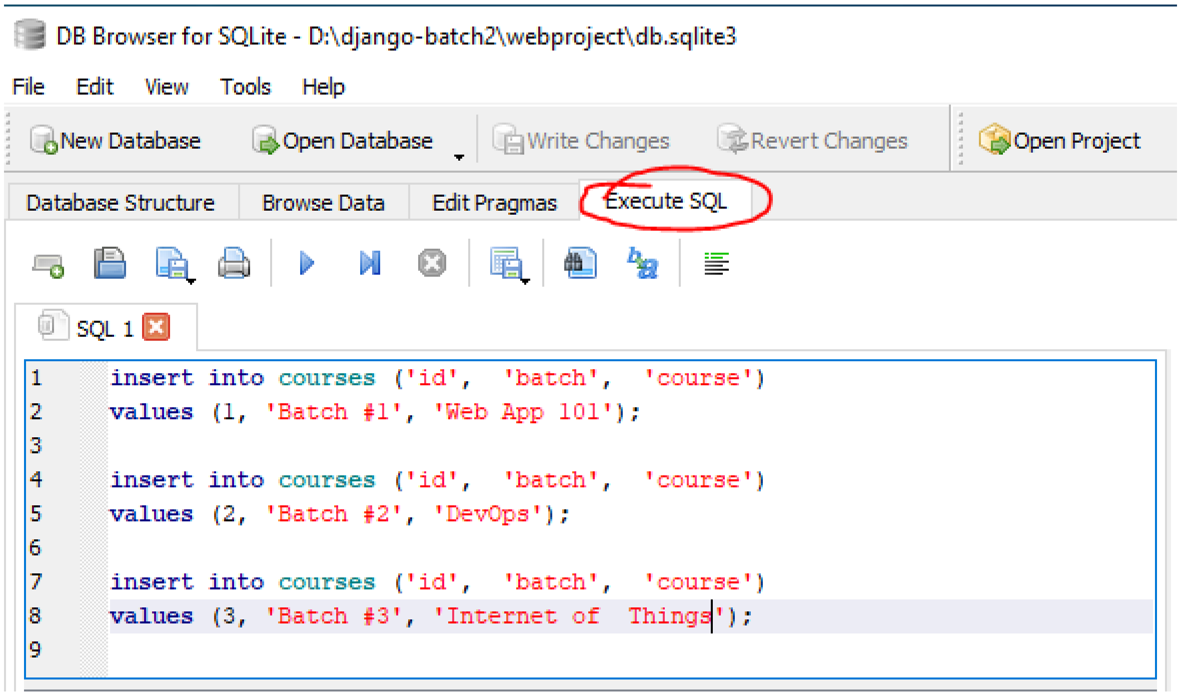


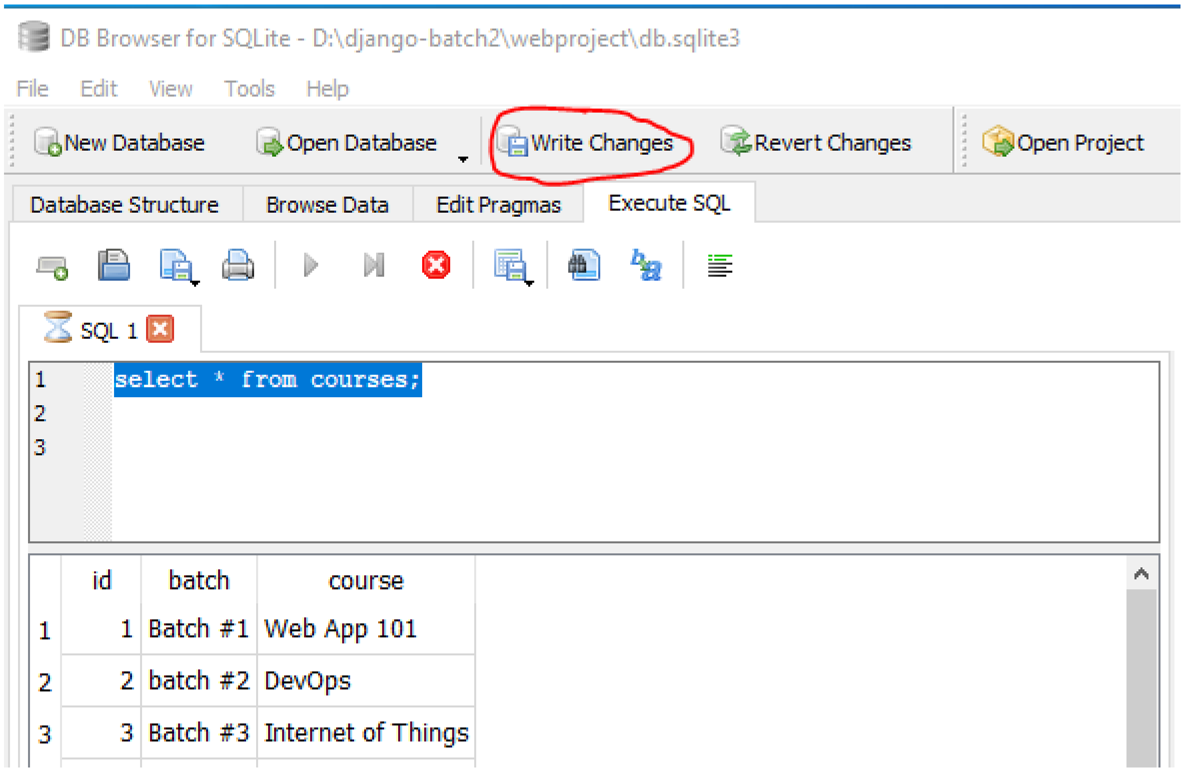
Table: courses

Name	Type	NN	PK	AI	U	Default	Check
id	INTEGER	☐	☑	☐	☐		
Field2course	TEXT	☐	☐	☐	☐		
Field3	TEXT	☐	☐	☐	☐		

```
CREATE TABLE "courses" (
    "id"    INTEGER,
    "Field2course"  TEXT,
    "Field3"    TEXT,
    PRIMARY KEY("id")
);
```



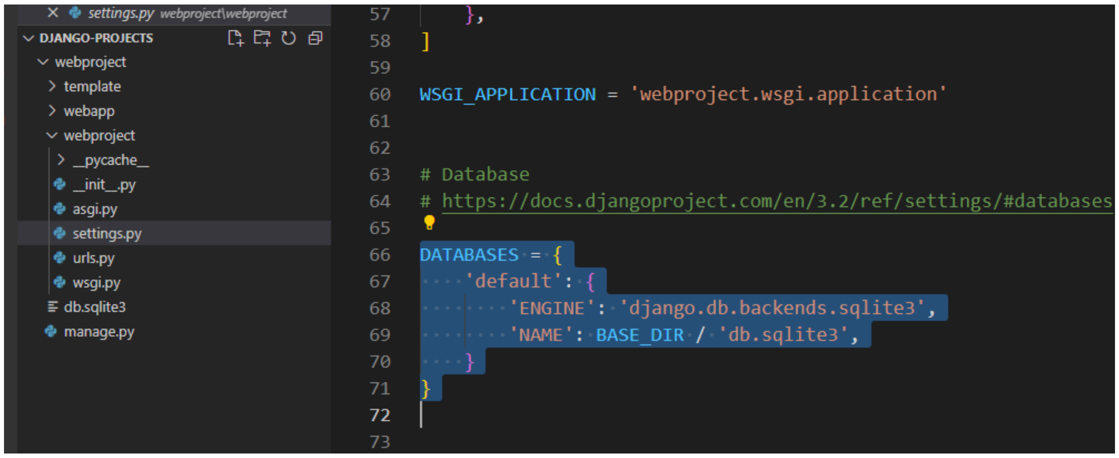
```
insert into courses ('id',  'batch',  'course')
values (1, 'Batch #1', 'Web App 101');

insert into courses ('id',  'batch',  'course')
values (2, 'Batch #2', 'DevOps');

insert into courses ('id',  'batch',  'course')
values (3, 'Batch #3', 'Internet of  Things');
```



```
select * from courses;
```

	id	batch	course
1	1	Batch #1	Web App 101
2	2	batch #2	DevOps
3	3	Batch #3	Internet of Things

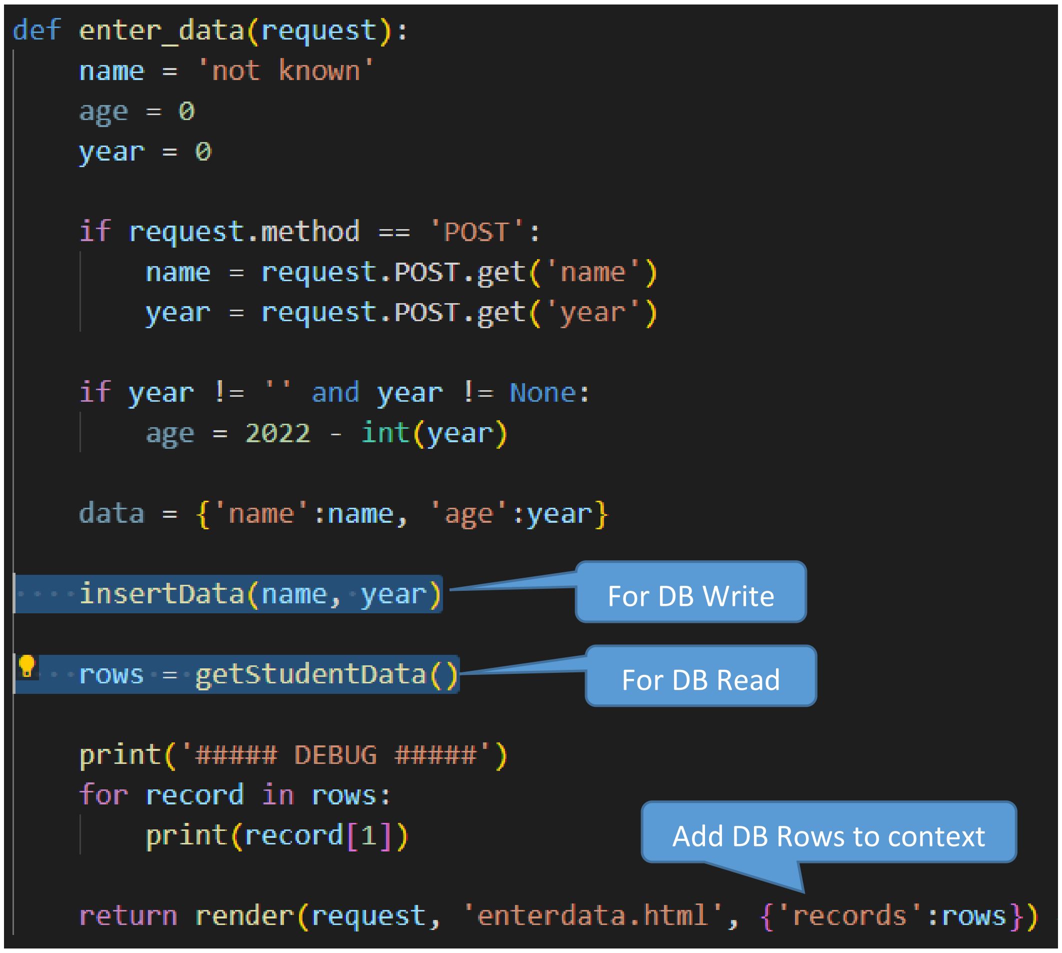
➔ **Check that project/settings.py has the sqlite database driver added (it should be added by default, and no change is required from your side):**



```
    },
]

WSGI_APPLICATION = 'webproject.wsgi.application'


# Database
# https://docs.djangoproject.com/en/3.2/ref/settings/#databases

DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.sqlite3',
        'NAME': BASE_DIR / 'db.sqlite3',
    }
}
```

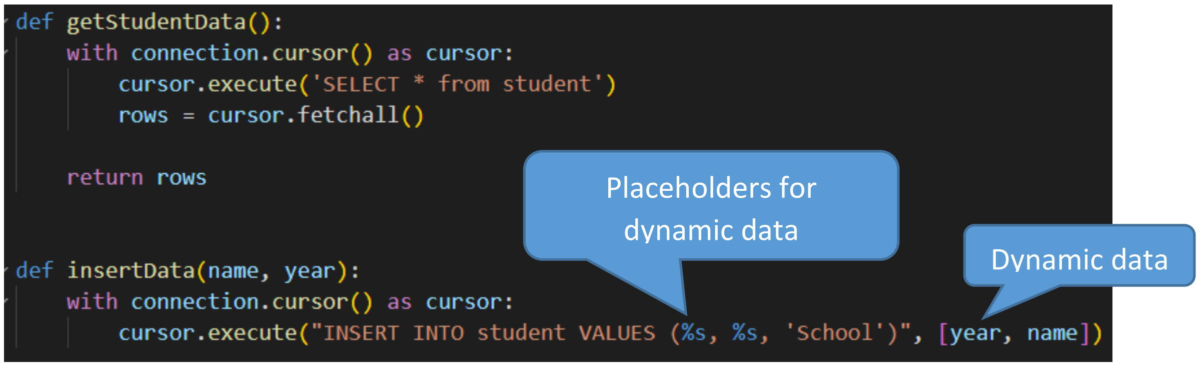
➔ **To read & write from the database using raw SQL in Django Handler (app/views.py):**

Call DB read/write functions in handler:



```
def enter_data(request):
    name = 'not known'
    age = 0
    year = 0

    if request.method == 'POST':
        name = request.POST.get('name')
        year = request.POST.get('year')

    if year != '' and year != None:
        age = 2022 - int(year)

    data = {'name':name, 'age':year}

    insertData(name, year)

    rows = getStudentData()

    print('##### DEBUG #####')
    for record in rows:
        print(record[1])

    return render(request, 'enterdata.html', {'records':rows})
```

Define the DB read/write functions:



```
def getStudentData():
    with connection.cursor() as cursor:
        cursor.execute('SELECT * from student')
        rows = cursor.fetchall()

    return rows


def insertData(name, year):
    with connection.cursor() as cursor:
        cursor.execute("INSERT INTO student VALUES (%s, %s, 'School')", [year, name])
```