1862

RĪGAS TEHNISKĀ UNIVERSITĀTE

Riga Technical University

Telecommunications Software (RAE411).

Third Practical Exercise.

29th of May 2023

Zeyad Mohamed Nashaat Abdelghany.

230AMB013.

Spring 2023.

I. Introduction:

This report discusses the tasks needed to be made in third practical exercise. There are four examples which will be done using Django. First example is to create a Django's hello world program to return a webpage with a string in it. The second example is to return an HTML page using Django. Third example discusses the idea of cloud message and how to accept and send message from a cloud to user on the web. Last example examines the 10 different HTML responses by using Django.

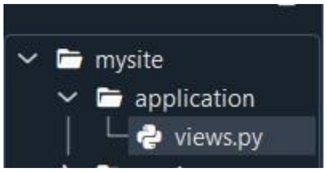
II. Example 1:

In this example we need to return a webpage with a string in it.

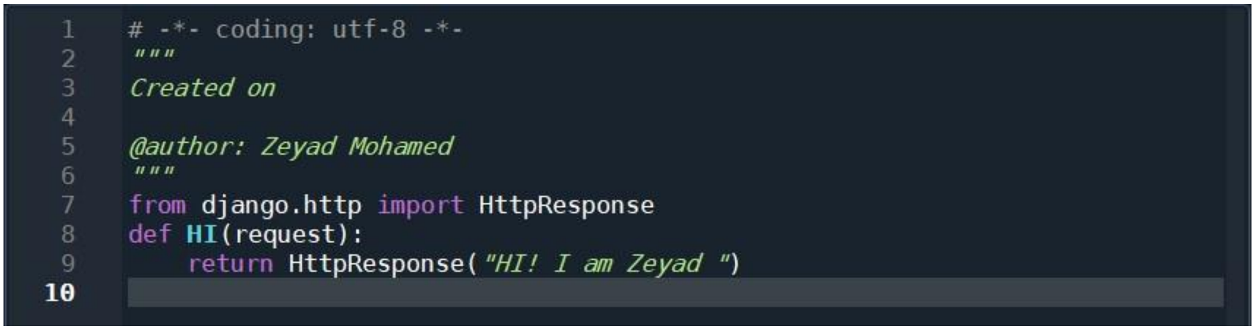
1. Create a new Django project by running this command in the terminal:

```
django-admin startproject <mysite>
```

2. Create a folder for this example (application):

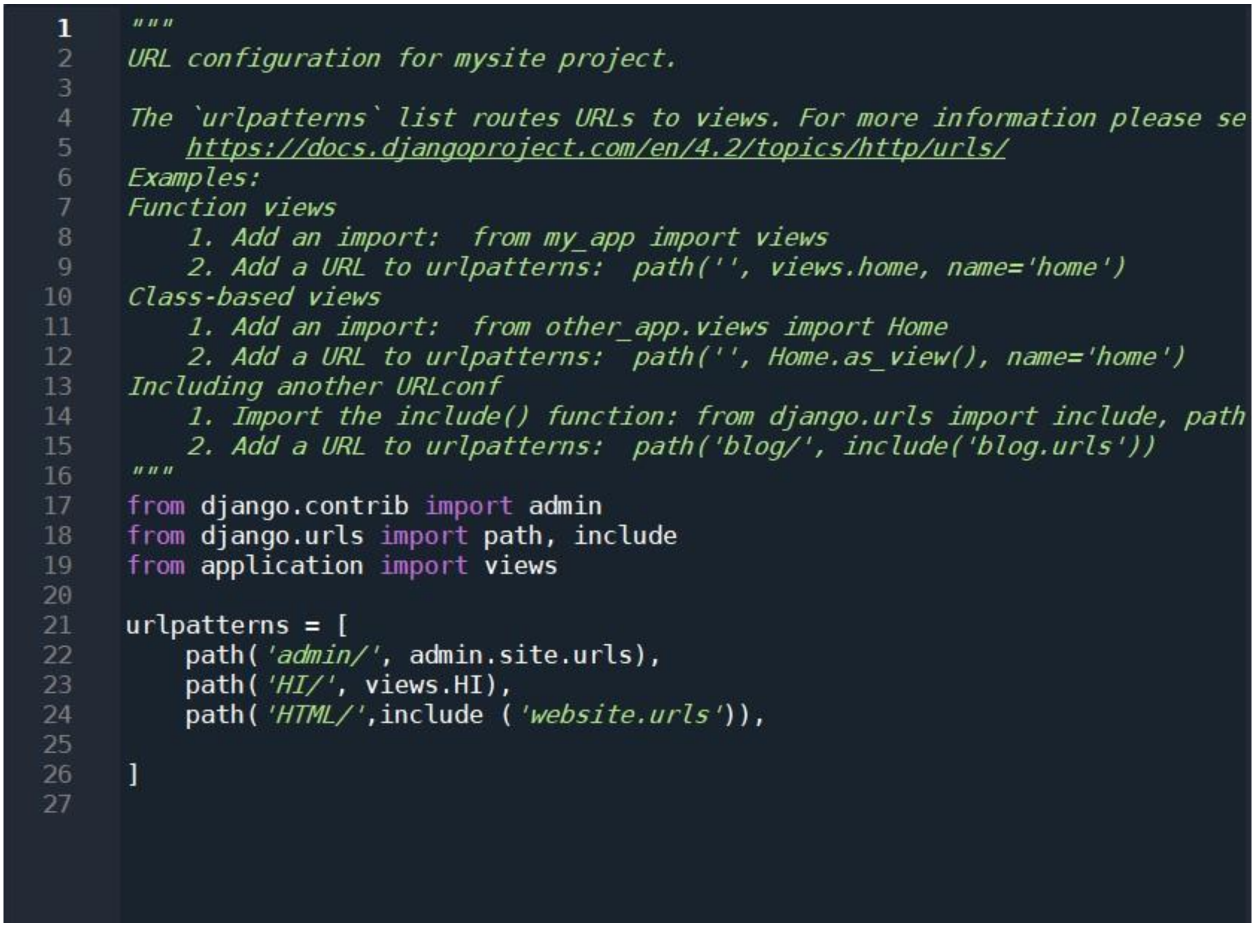


3. Create views.py and put the code in it:



```python
# -*- coding: utf-8 -*-
"""
Created on

@author: Zeyad Mohamed
"""
from django.http import HttpResponse
def HI(request):
    return HttpResponse("HI! I am Zeyad ")

```

4. Add URL for the application in the main file urls.py:

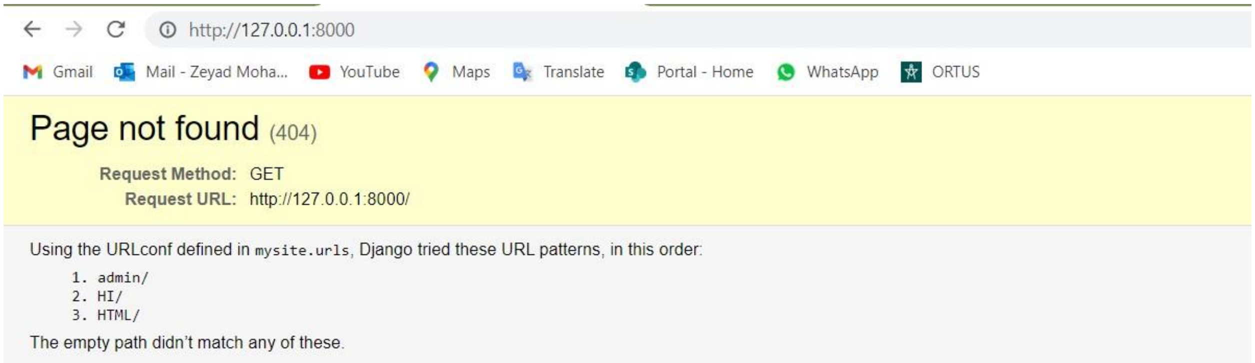


```
"""
URL configuration for mysite project.

The `urlpatterns` list routes URLs to views. For more information please se
    https://docs.djangoproject.com/en/4.2/topics/http/urls/
Examples:
Function views
    1. Add an import:  from my_app import views
    2. Add a URL to urlpatterns:  path('', views.home, name='home')
Class-based views
    1. Add an import:  from other_app.views import Home
    2. Add a URL to urlpatterns:  path('', Home.as_view(), name='home')
Including another URLconf
    1. Import the include() function: from django.urls import include, path
    2. Add a URL to urlpatterns:  path('blog/', include('blog.urls'))
"""
from django.contrib import admin
from django.urls import path, include
from application import views

urlpatterns = [
    path('admin/', admin.site.urls),
    path('HI/', views.HI),
    path('HTML/',include ('website.urls')),

]
```

5. Run the program by writing this line in the terminal:

python manage.py runserver

6. Results should be as follows:



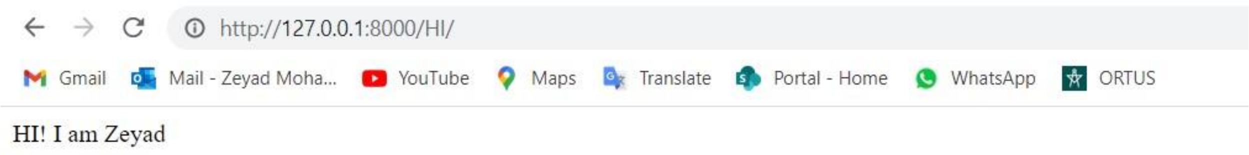
Page not found (404)

Request Method: GET
Request URL: http://127.0.0.1:8000/

Using the URLconf defined in `mysite.urls`, Django tried these URL patterns, in this order:

```
1. admin/
2. HI/
3. HTML/
```

The empty path didn't match any of these.

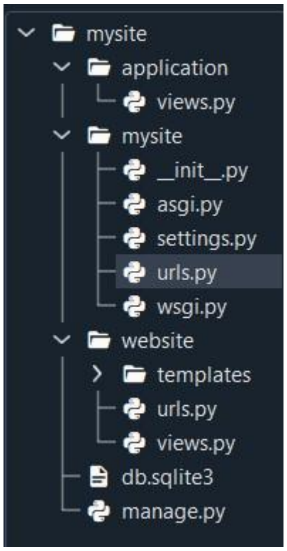


HI! I am Zeyad

III. Example 2:

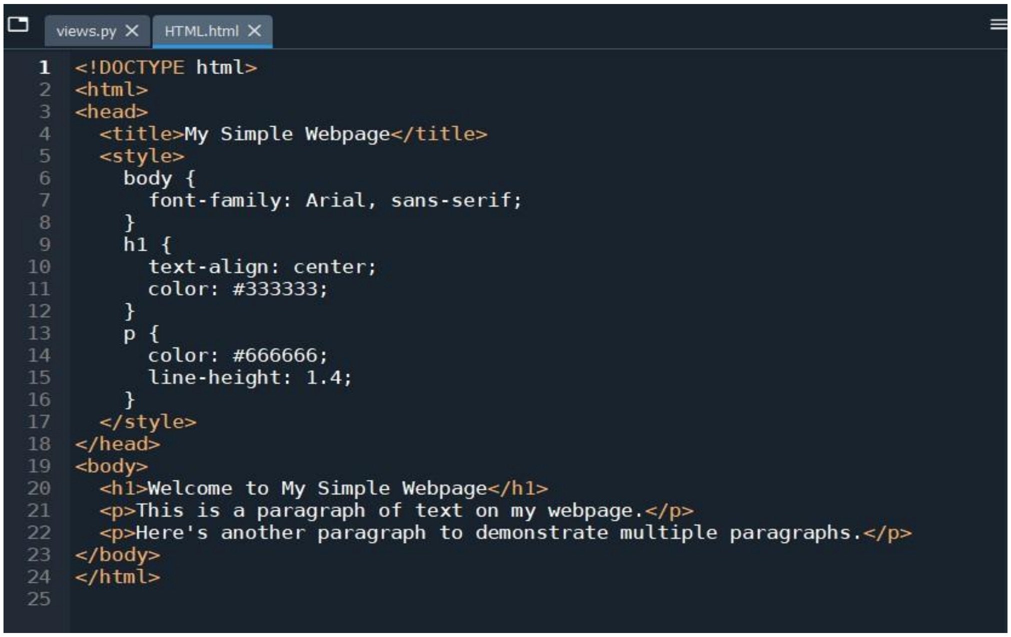
In this example, we will update the previous one to be able to return HTML page instead of a string.

1. Create a folder for this example (website):



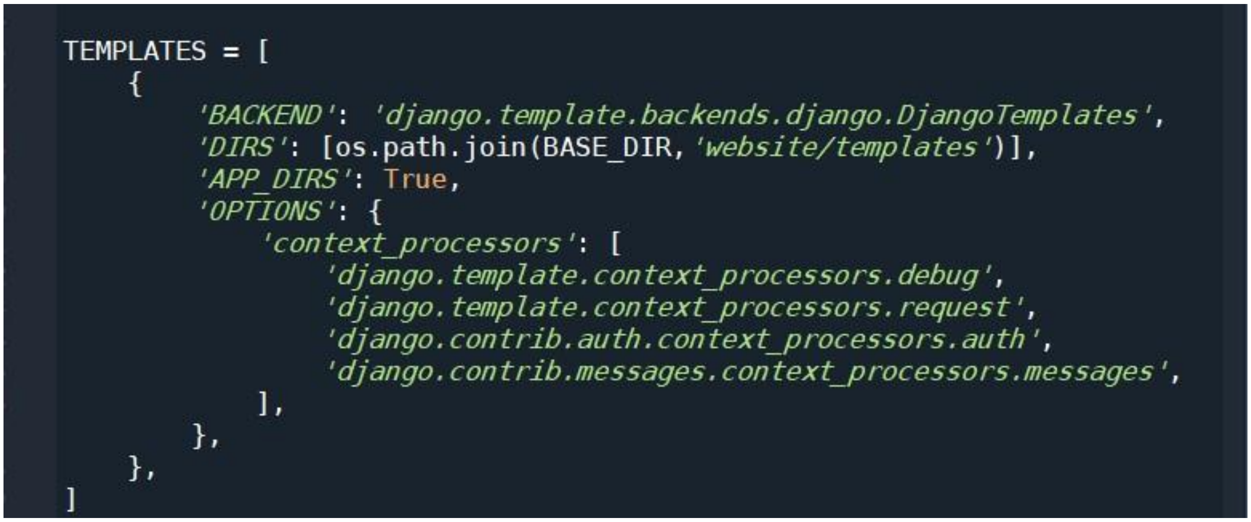
```
mysite
    application
        views.py
    mysite
        __init__.py
        asgi.py
        settings.py
        urls.py
        wsgi.py
    website
        templates
        urls.py
        views.py
    db.sqlite3
    manage.py
```

2. Add a template folder to hold the HTML code for the page:



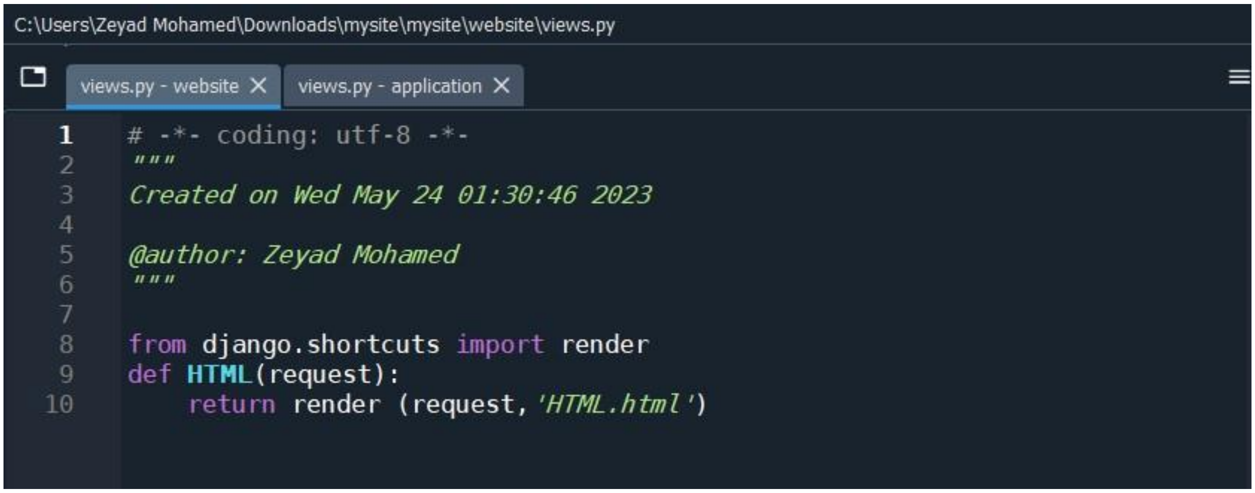
```html
<!DOCTYPE html>
<html>
<head>
  <title>My Simple Webpage</title>
  <style>
    body {
      font-family: Arial, sans-serif;
    }
    h1 {
      text-align: center;
      color: #333333;
    }
    p {
      color: #666666;
      line-height: 1.4;
    }
  </style>
</head>
<body>
  <h1>Welcome to My Simple Webpage</h1>
  <p>This is a paragraph of text on my webpage.</p>
  <p>Here's another paragraph to demonstrate multiple paragraphs.</p>
</body>
</html>

```

3. Edit the settings.py in the main project to hold the direction of the added template:



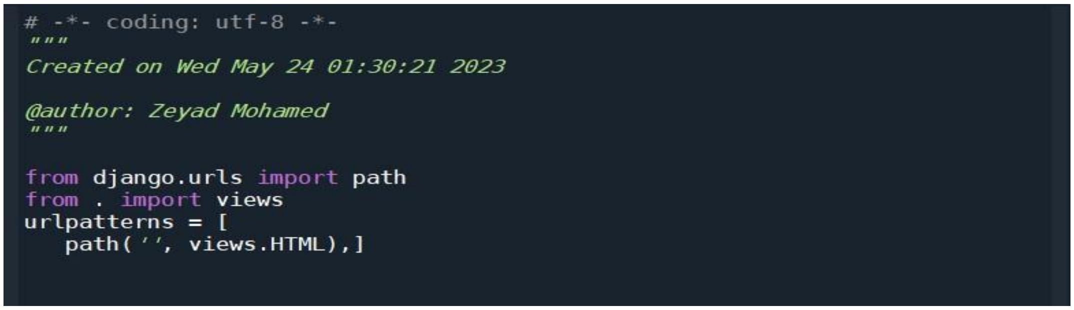
```
TEMPLATES = [
    {
        'BACKEND': 'django.template.backends.django.DjangoTemplates',
        'DIRS': [os.path.join(BASE_DIR, 'website/templates')],
        'APP_DIRS': True,
        'OPTIONS': {
            'context_processors': [
                'django.template.context_processors.debug',
                'django.template.context_processors.request',
                'django.contrib.auth.context_processors.auth',
                'django.contrib.messages.context_processors.messages',
            ],
        },
    },
]
```

4. Add the function in views.py to call the template:


C:\Users\Zeyad Mohamed\Downloads\mysite\mysite\website\views.py
views.py - website
views.py - application

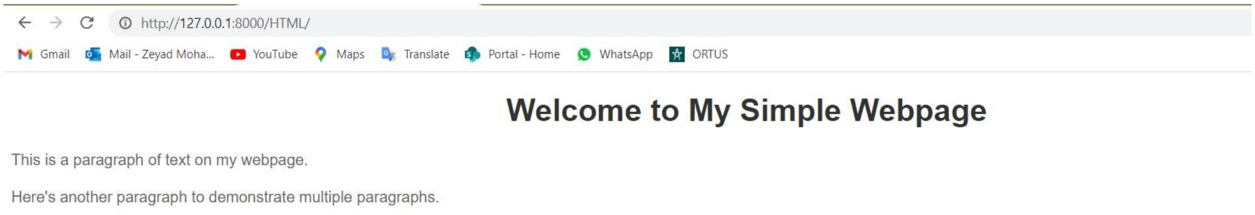
```
# -*- coding: utf-8 -*-
"""
Created on Wed May 24 01:30:46 2023

@author: Zeyad Mohamed
"""

from django.shortcuts import render
def HTML(request):
    return render (request,'HTML.html')
```

5. Add the URL for urls.py:



```
# -*- coding: utf-8 -*-
"""
Created on Wed May 24 01:30:21 2023

@author: Zeyad Mohamed
"""

from django.urls import path
from . import views
urlpatterns = [
   path('', views.HTML),]
```

6. The results should be:



Welcome to My Simple Webpage

This is a paragraph of text on my webpage.

Here's another paragraph to demonstrate multiple paragraphs.

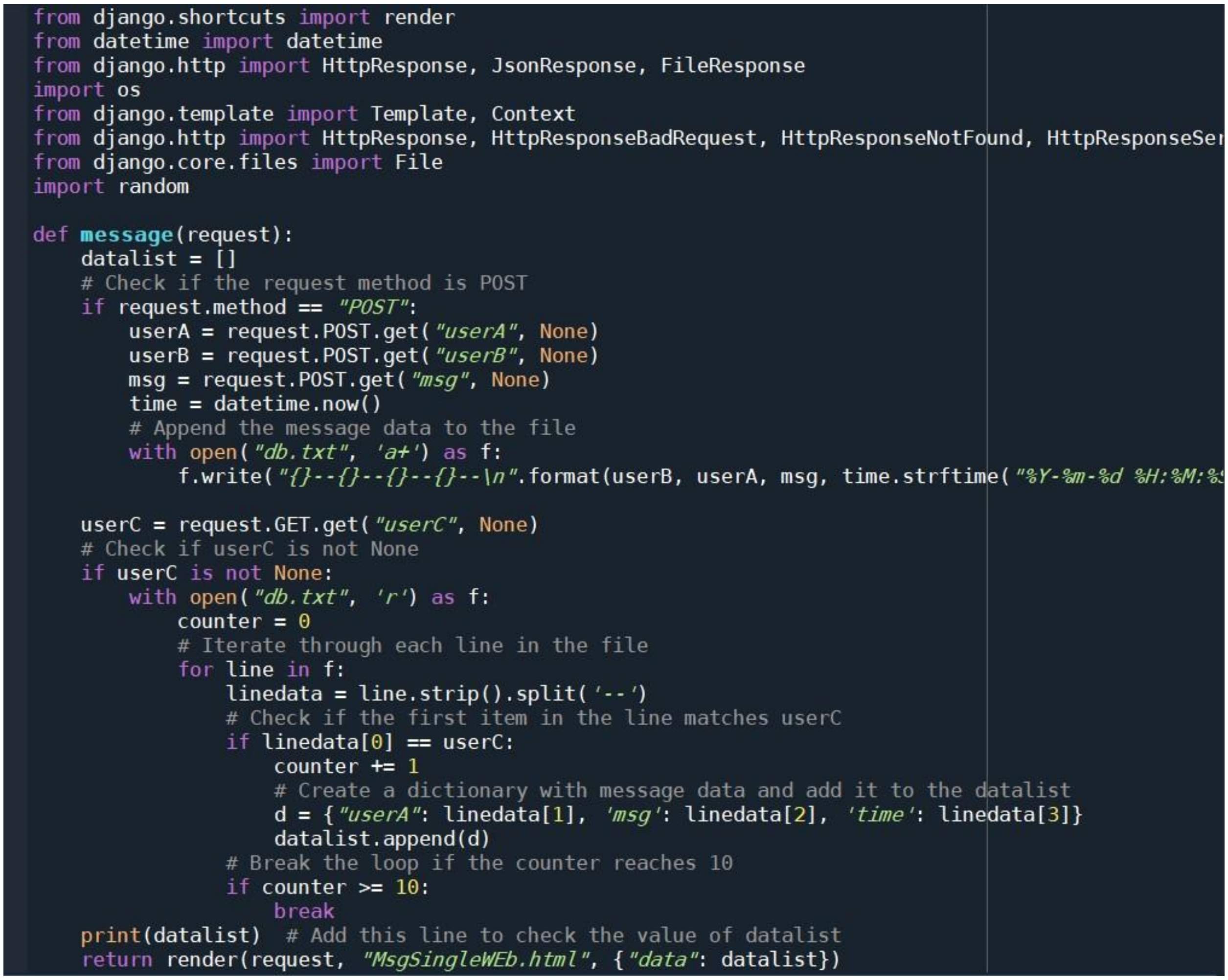
IV. Example 3:

This example is to make a message system where you can submit a message with your name to a certain receiver. Also, the receiver could see up to 20 messages sent to him by showing the message with the name of the sender as well. The steps to create such project:

1. Create a new Django project through the terminal:

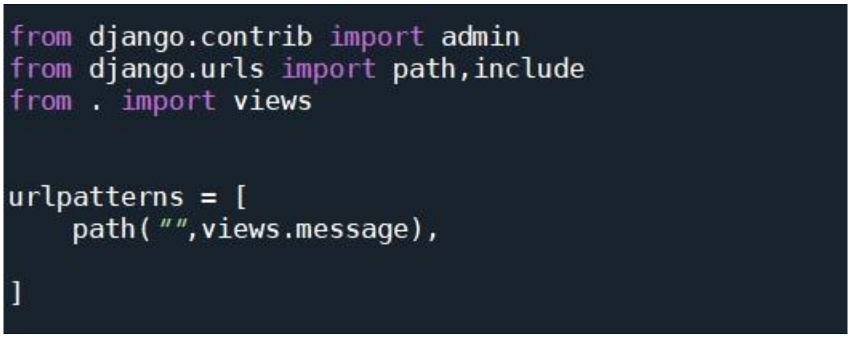
django-admin startproject <C_Message>

2. Add functions to views.py:



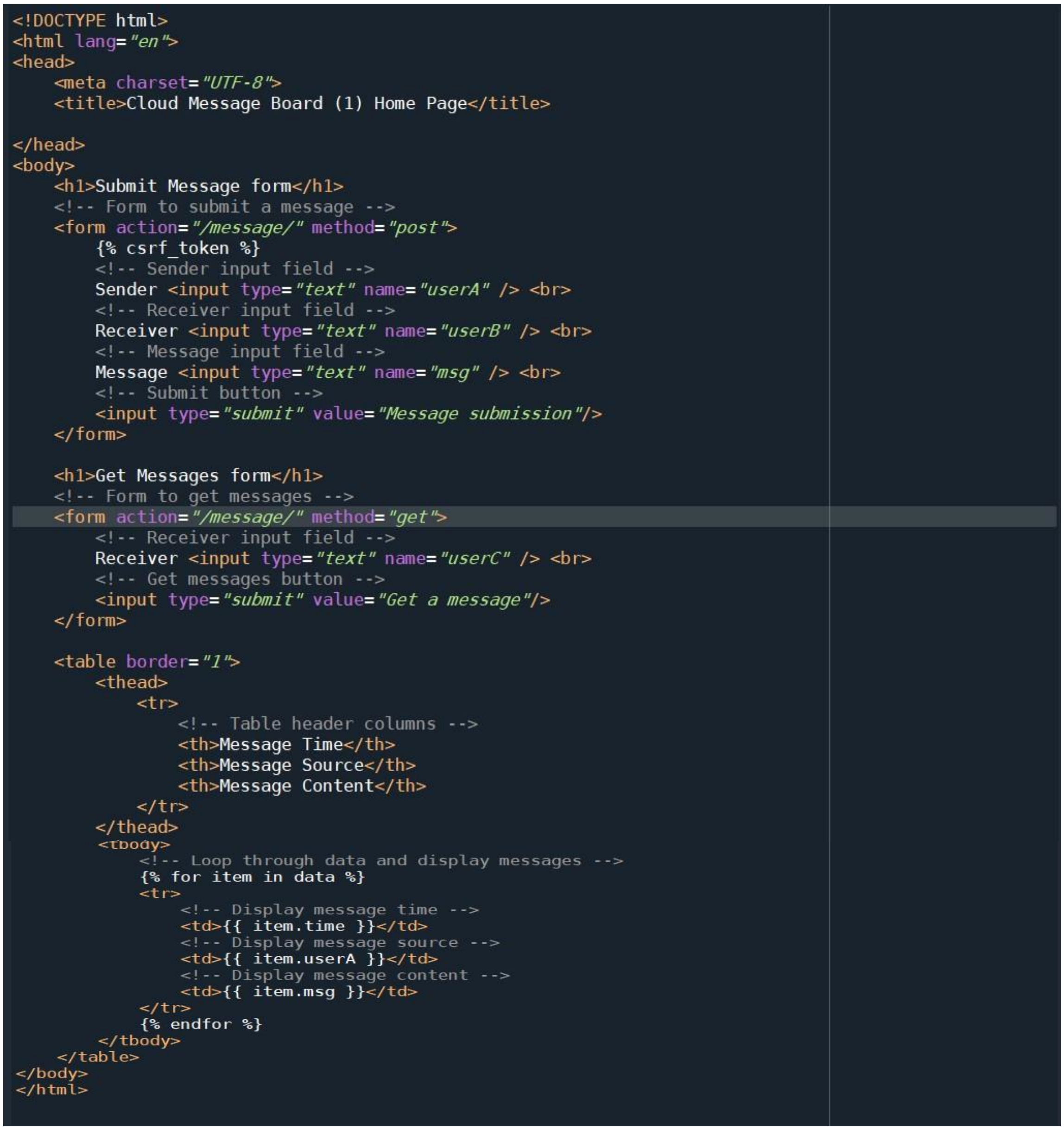
```
from django.shortcuts import render
from datetime import datetime
from django.http import HttpResponse, JsonResponse, FileResponse
import os
from django.template import Template, Context
from django.http import HttpResponse, HttpResponseBadRequest, HttpResponseNotFound, HttpResponseSer
from django.core.files import File
import random

def message(request):
    datalist = []
    # Check if the request method is POST
    if request.method == "POST":
        userA = request.POST.get("userA", None)
        userB = request.POST.get("userB", None)
        msg = request.POST.get("msg", None)
        time = datetime.now()
        # Append the message data to the file
        with open("db.txt", 'a+') as f:
            f.write("{}--{}--{}--{}--\n".format(userB, userA, msg, time.strftime("%Y-%m-%d %H:%M:%

    userC = request.GET.get("userC", None)
    # Check if userC is not None
    if userC is not None:
        with open("db.txt", 'r') as f:
            counter = 0
            # Iterate through each line in the file
            for line in f:
                linedata = line.strip().split('--')
                # Check if the first item in the line matches userC
                if linedata[0] == userC:
                    counter += 1
                    # Create a dictionary with message data and add it to the datalist
                    d = {"userA": linedata[1], 'msg': linedata[2], 'time': linedata[3]}
                    datalist.append(d)
                # Break the loop if the counter reaches 10
                if counter >= 10:
                    break
    print(datalist)  # Add this line to check the value of datalist
    return render(request, "MsgSingleWEb.html", {"data": datalist})
```

3. Add the needed URLs to urls.py:



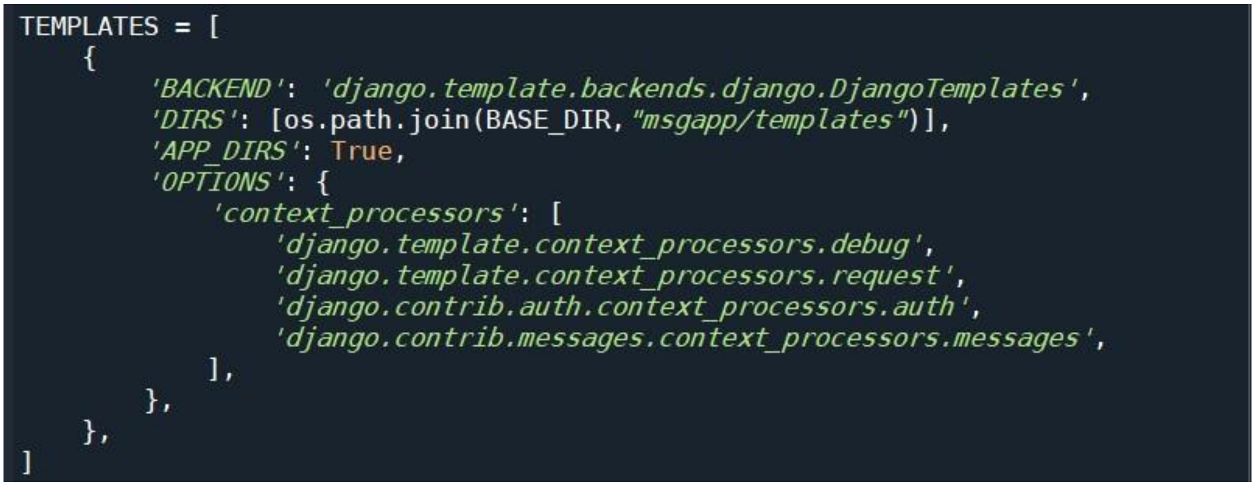
```
from django.contrib import admin
from django.urls import path,include
from . import views


urlpatterns = [
    path("",views.message),

]
```

4. Create an HTML page to be the view for the user to hold the messages sending and receiving actions:



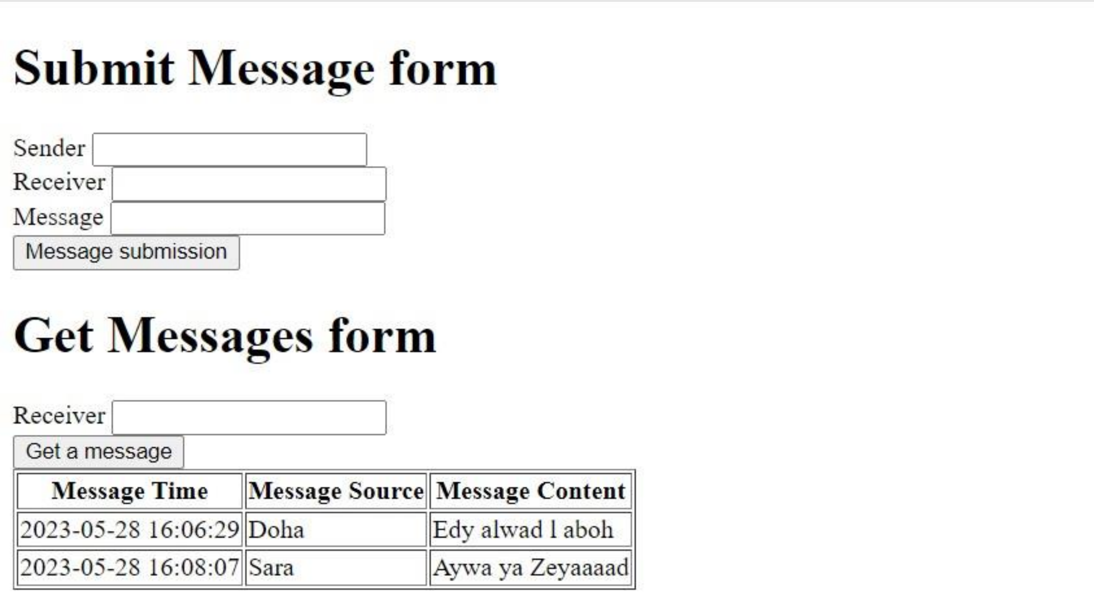
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Cloud Message Board (1) Home Page</title>

</head>
<body>
    <h1>Submit Message form</h1>
    <!-- Form to submit a message -->
    <form action="/message/" method="post">
        {% csrf_token %}
        <!-- Sender input field -->
        Sender <input type="text" name="userA" /> <br>
        <!-- Receiver input field -->
        Receiver <input type="text" name="userB" /> <br>
        <!-- Message input field -->
        Message <input type="text" name="msg" /> <br>
        <!-- Submit button -->
        <input type="submit" value="Message submission"/>
    </form>

    <h1>Get Messages form</h1>
    <!-- Form to get messages -->
    <form action="/message/" method="get">
        <!-- Receiver input field -->
        Receiver <input type="text" name="userC" /> <br>
        <!-- Get messages button -->
        <input type="submit" value="Get a message"/>
    </form>

    <table border="1">
        <thead>
            <tr>
                <!-- Table header columns -->
                <th>Message Time</th>
                <th>Message Source</th>
                <th>Message Content</th>
            </tr>
        </thead>
        <tbody>
            <!-- Loop through data and display messages -->
            {% for item in data %}
            <tr>
                <!-- Display message time -->
                <td>{{ item.time }}</td>
                <!-- Display message source -->
                <td>{{ item.userA }}</td>
                <!-- Display message content -->
                <td>{{ item.msg }}</td>
            </tr>
            {% endfor %}
        </tbody>
    </table>
</body>
</html>
```

5. Adjust templates part in settings.py:



```
TEMPLATES = [
    {
        'BACKEND': 'django.template.backends.django.DjangoTemplates',
        'DIRS': [os.path.join(BASE_DIR,"msgapp/templates")],
        '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. For the working results:



Submit Message form

Sender

Receiver

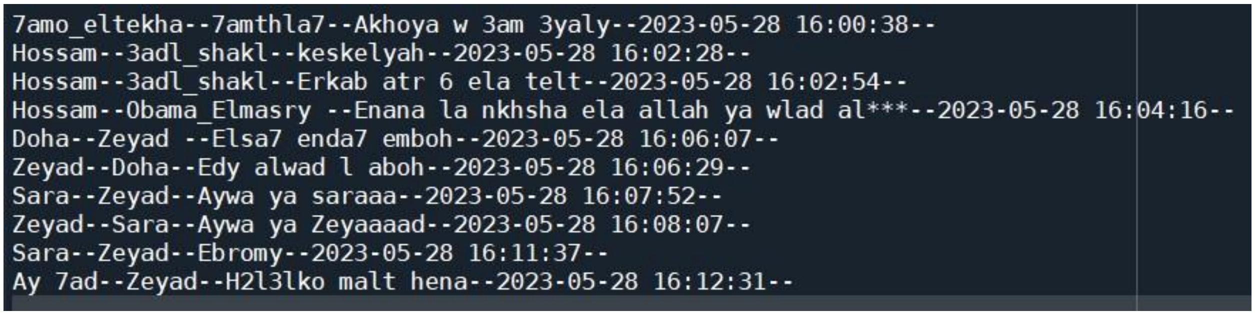
Message

Get Messages form

Receiver

Message Time	Message Source	Message Content
2023-05-28 16:06:29	Doha	Edy alwad l aboh
2023-05-28 16:08:07	Sara	Aywa ya Zeyaaaad

7. For the data base of the messages:



```
7amo_eltekha--7amthla7--Akhoya w 3am 3yaly--2023-05-28 16:00:38--
Hossam--3adl_shakl--keskelyah--2023-05-28 16:02:28--
Hossam--3adl_shakl--Erkab atr 6 ela telt--2023-05-28 16:02:54--
Hossam--Obama_Elmasry --Enana la nkhsha ela allah ya wlad al***--2023-05-28 16:04:16--
Doha--Zeyad --Elsa7 enda7 emboh--2023-05-28 16:06:07--
Zeyad--Doha--Edy alwad l aboh--2023-05-28 16:06:29--
Sara--Zeyad--Aywa ya saraaa--2023-05-28 16:07:52--
Zeyad--Sara--Aywa ya Zeyaaaad--2023-05-28 16:08:07--
Sara--Zeyad--Ebromy--2023-05-28 16:11:37--
Ay 7ad--Zeyad--H2l3lko malt hena--2023-05-28 16:12:31--
```

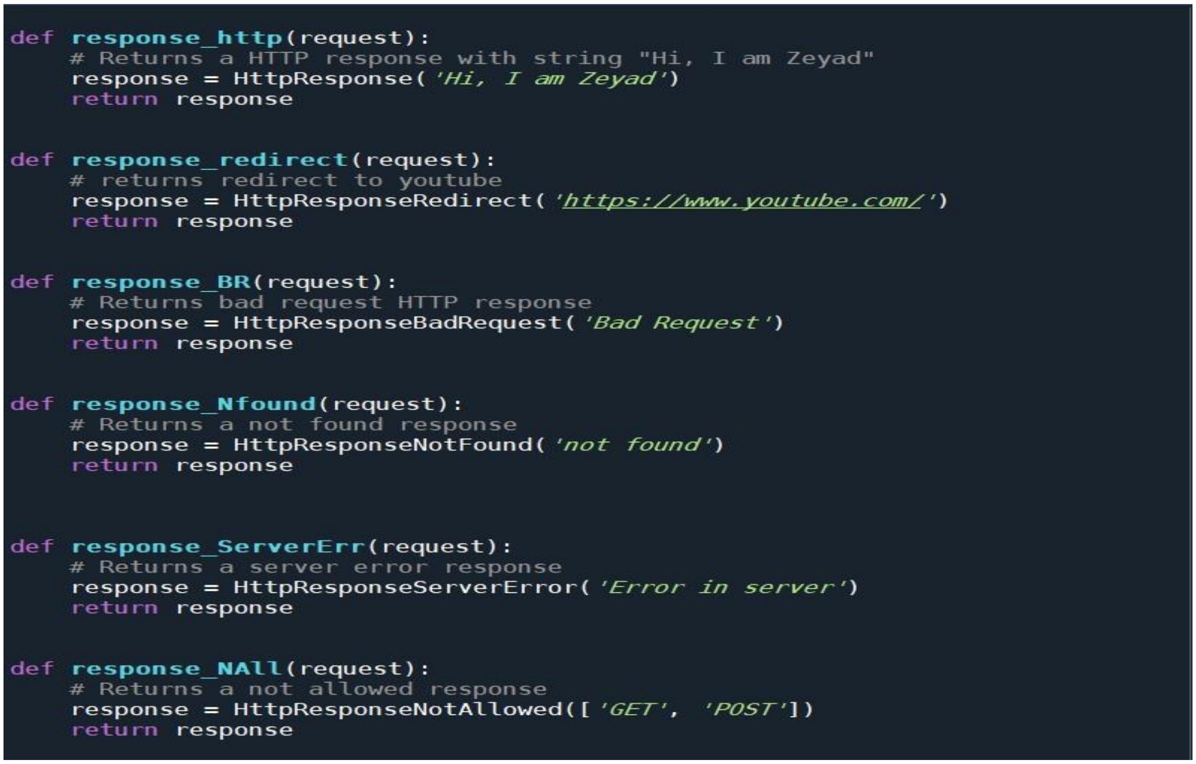
V. Example 4:

In this section, 10 responses for Django are tried. The 10 responses will be explained with given screenshots and discussion.

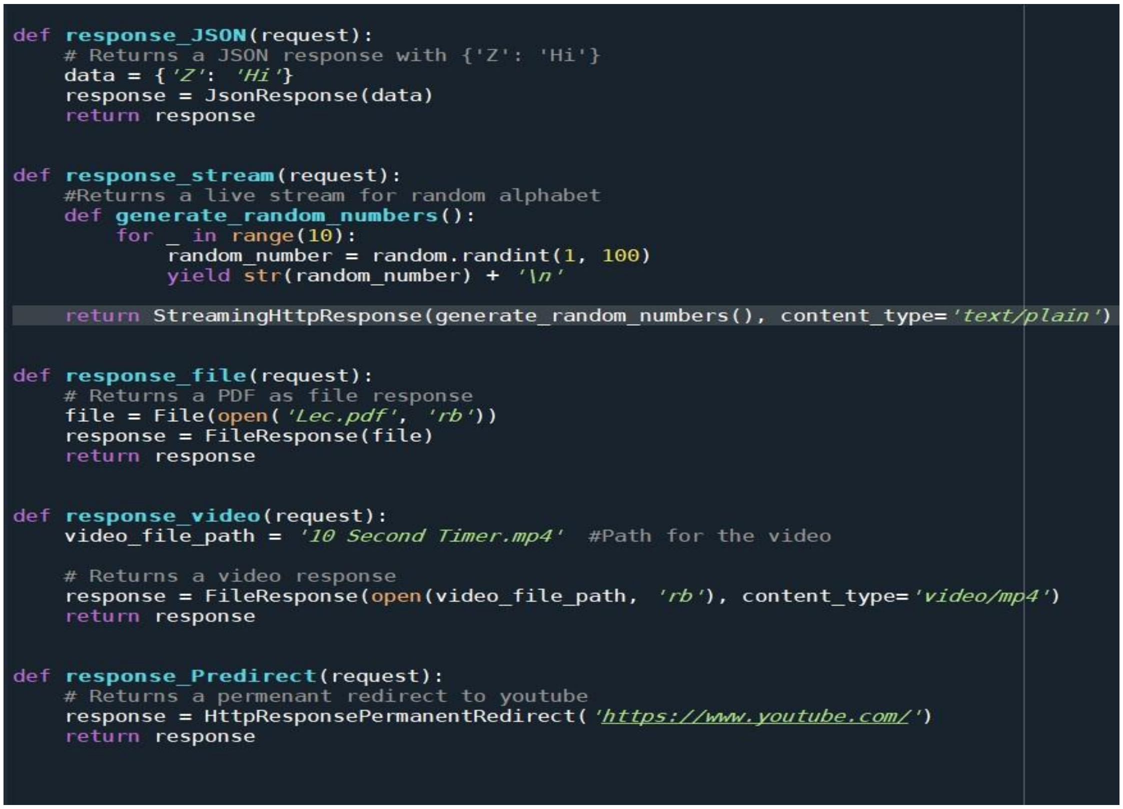
a. **Http response:** this response returns a simple Http page with a string in it “Hi, I am Zeyad”.
b. **Redirect Response:** this response returns a redirect page to any website, for example: YouTube.
c. **Bad Request response:** this response returns an HTTP page with Bad request.
d. **Not Found response:** this response returns Not found HTTP response.
e. **Server Error response:** this response returns server error HTTP response.
f. **Not Allowed response:** this response returns not allowed HTTP response.
g. **JSON response:** this response returns a JSON file with data “{‘Z’:’ Hi’}”.
h. **Stream response:** this response returns a stream HTTP response. For example, it can return a stream of 10 random alphabets.
i. **File response:** this response returns a file as HTTP response. For example, you can return pdf file as response.
j. **Video response:** this response returns a video as HTTP response.
k. **Permanent redirect response:** this response returns a permanent redirect link to YouTube.

For the steps to make such responses:

1. Add the returning functions to views.py:

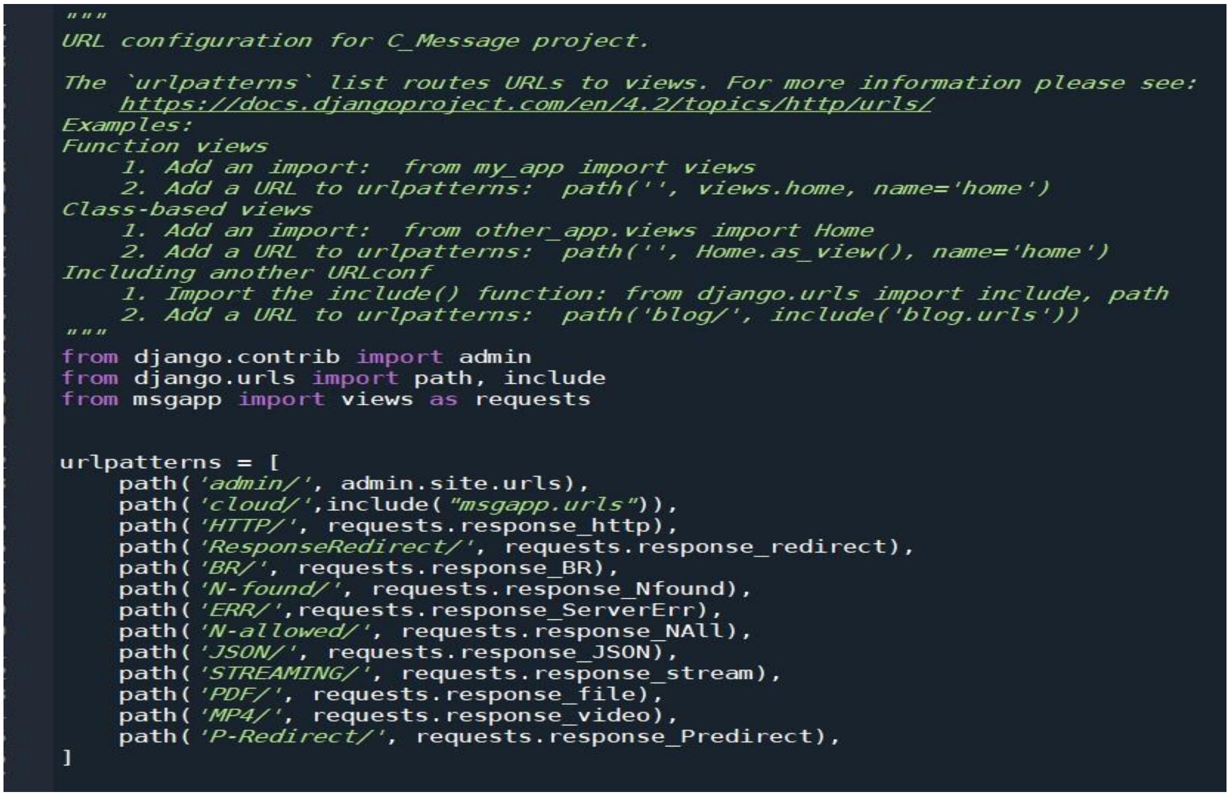


```
def response_http(request):
    # Returns a HTTP response with string "Hi, I am Zeyad"
    response = HttpResponse('Hi, I am Zeyad')
    return response


def response_redirect(request):
    # returns redirect to youtube
    response = HttpResponseRedirect('https://www.youtube.com/')
    return response


def response_BR(request):
    # Returns bad request HTTP response
    response = HttpResponseBadRequest('Bad Request')
    return response


def response_Nfound(request):
    # Returns a not found response
    response = HttpResponseNotFound('not found')
    return response



def response_ServerErr(request):
    # Returns a server error response
    response = HttpResponseServerError('Error in server')
    return response


def response_NAll(request):
    # Returns a not allowed response
    response = HttpResponseNotAllowed(['GET', 'POST'])
    return response
```



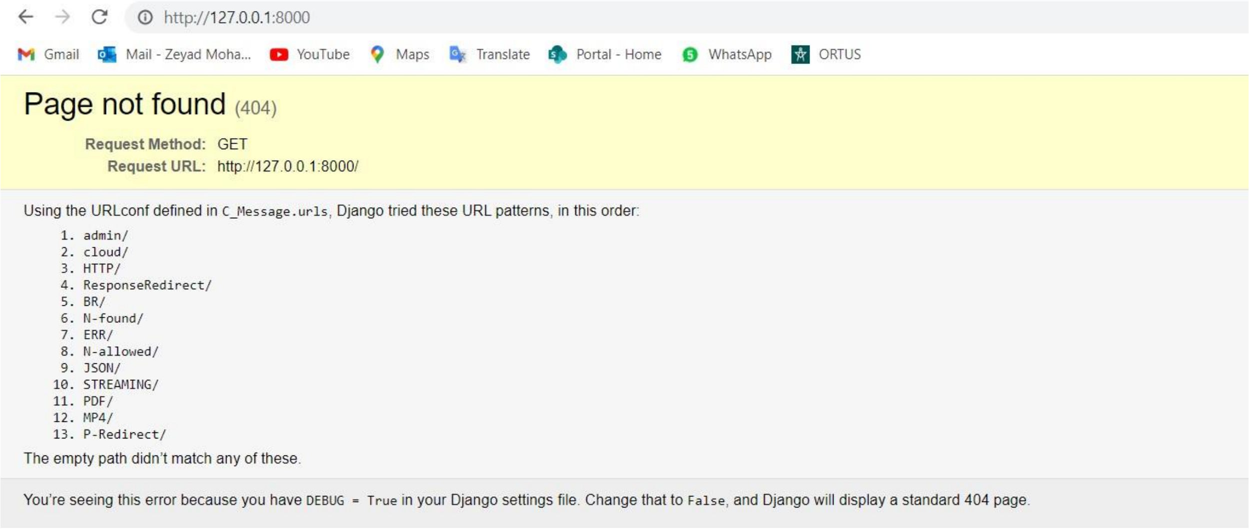
```
def response_JSON(request):
    # Returns a JSON response with {'Z': 'Hi'}
    data = {'Z': 'Hi'}
    response = JsonResponse(data)
    return response


def response_stream(request):
    #Returns a live stream for random alphabet
    def generate_random_numbers():
        for _ in range(10):
            random_number = random.randint(1, 100)
            yield str(random_number) + '\n'

    return StreamingHttpResponse(generate_random_numbers(), content_type='text/plain')


def response_file(request):
    # Returns a PDF as file response
    file = File(open('Lec.pdf', 'rb'))
    response = FileResponse(file)
    return response


def response_video(request):
    video_file_path = '10 Second Timer.mp4'  #Path for the video

    # Returns a video response
    response = FileResponse(open(video_file_path, 'rb'), content_type='video/mp4')
    return response


def response_Predirect(request):
    # Returns a permenant redirect to youtube
    response = HttpResponsePermanentRedirect('https://www.youtube.com/')
    return response
```

2. Adjust urls.py file:



```
"""
URL configuration for C_Message project.

The `urlpatterns` list routes URLs to views. For more information please see:
    https://docs.djangoproject.com/en/4.2/topics/http/urls/
Examples:
Function views
    1. Add an import:  from my_app import views
    2. Add a URL to urlpatterns:  path('', views.home, name='home')
Class-based views
    1. Add an import:  from other_app.views import Home
    2. Add a URL to urlpatterns:  path('', Home.as_view(), name='home')
Including another URLconf
    1. Import the include() function: from django.urls import include, path
    2. Add a URL to urlpatterns:  path('blog/', include('blog.urls'))
"""
from django.contrib import admin
from django.urls import path, include
from msgapp import views as requests


urlpatterns = [
    path('admin/', admin.site.urls),
    path('cloud/',include("msgapp.urls")),
    path('HTTP/', requests.response_http),
    path('ResponseRedirect/', requests.response_redirect),
    path('BR/', requests.response_BR),
    path('N-found/', requests.response_Nfound),
    path('ERR/',requests.response_ServerErr),
    path('N-allowed/', requests.response_NAll),
    path('JSON/', requests.response_JSON),
    path('STREAMING/', requests.response_stream),
    path('PDF/', requests.response_file),
    path('MP4/', requests.response_video),
    path('P-Redirect/', requests.response_Predirect),
]
```

3. Results:



Page not found (404)

Request Method: GET
Request URL: http://127.0.0.1:8000/

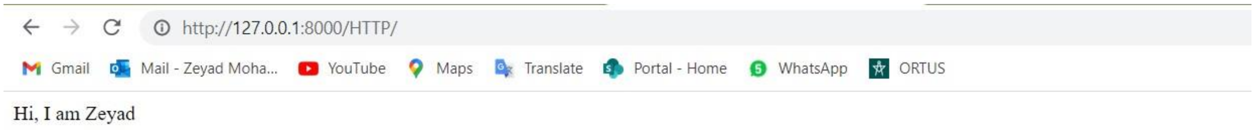
Using the URLconf defined in `C_Message.urls`, Django tried these URL patterns, in this order:

```
 1. admin/
 2. cloud/
 3. HTTP/
 4. ResponseRedirect/
 5. BR/
 6. N-found/
 7. ERR/
 8. N-allowed/
 9. JSON/
10. STREAMING/
11. PDF/
12. MP4/
13. P-Redirect/
```

The empty path didn't match any of these.

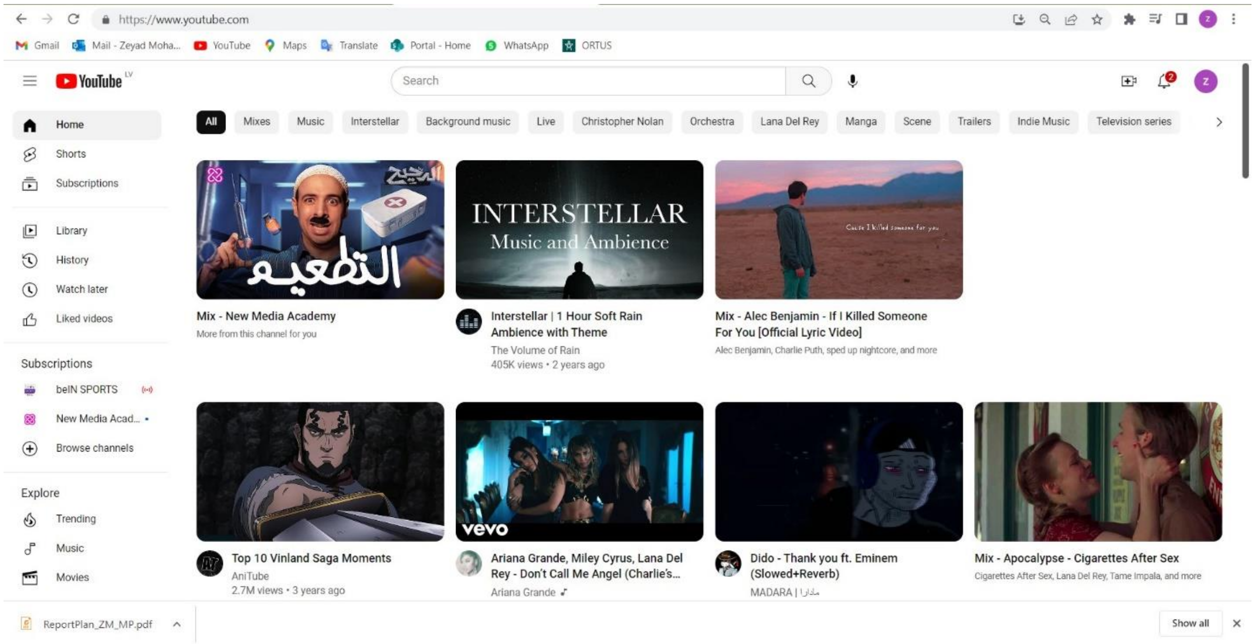
You're seeing this error because you have `DEBUG = True` in your Django settings file. Change that to `False`, and Django will display a standard 404 page.

a. HTTP response:

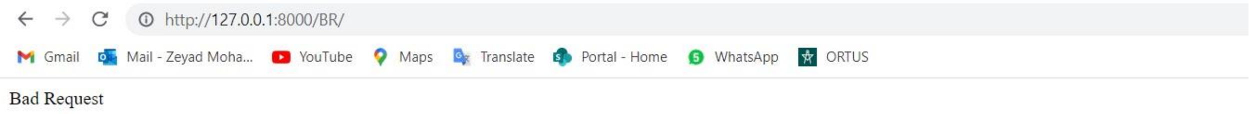


Hi, I am Zeyad

b. Redirect Response:

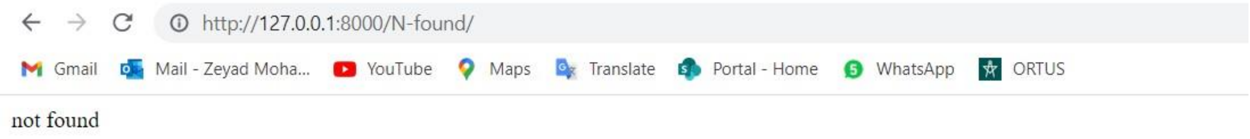


c. Bad Request response:



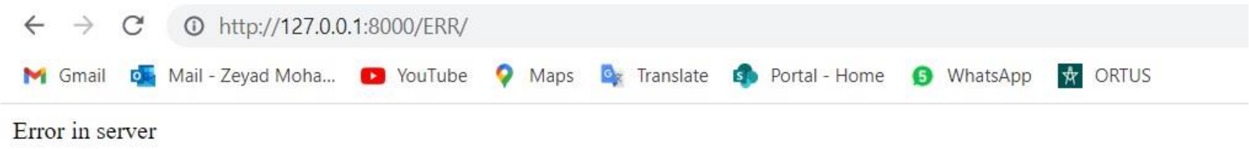
Bad Request

d. Not Found response:



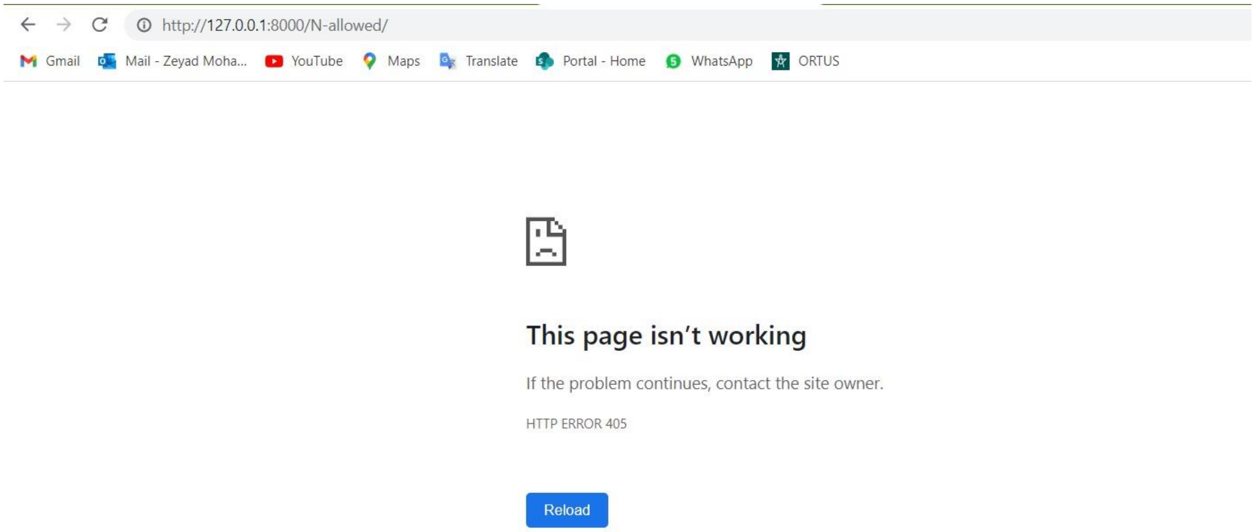
not found

e. Server Error response:



Error in server

f. Not Allowed response:

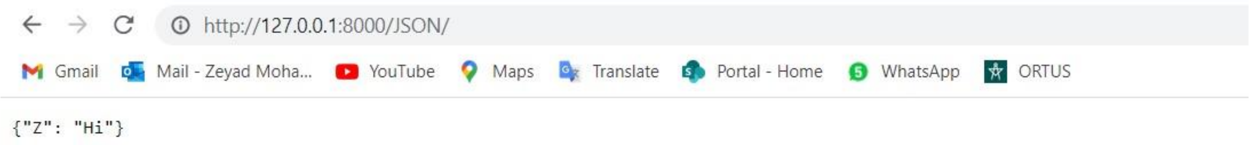


This page isn’t working

If the problem continues, contact the site owner.

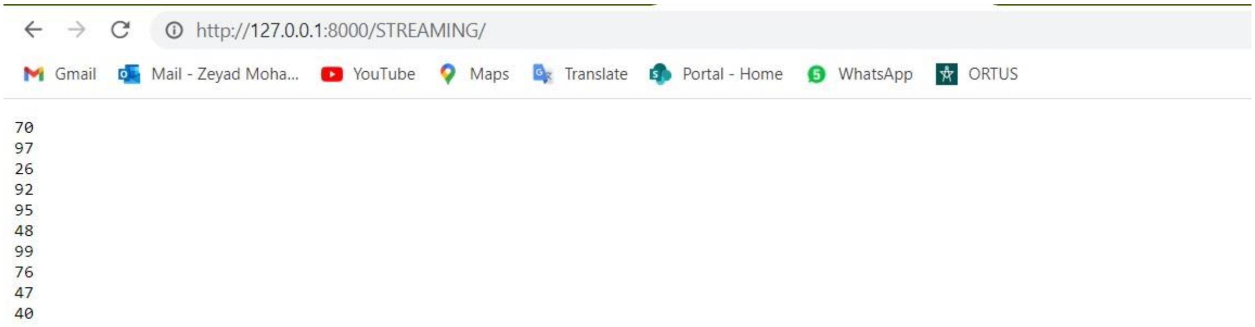
HTTP ERROR 405

g. JSON response:



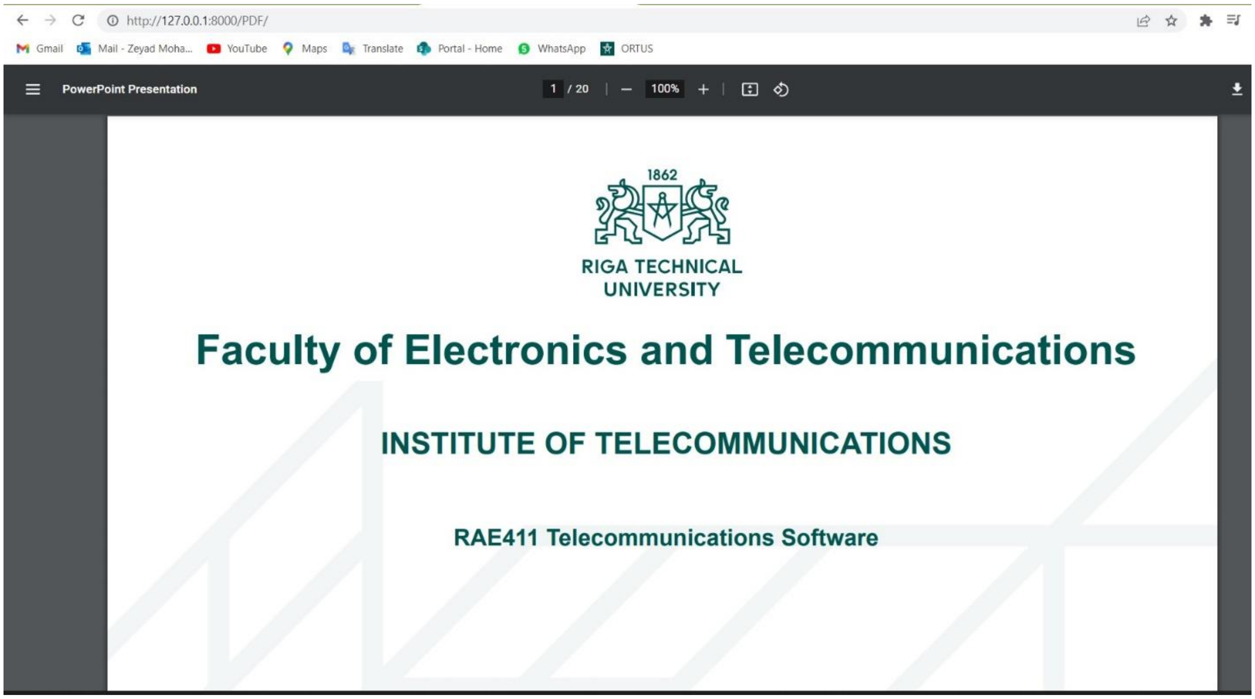
```
{"Z": "Hi"}
```

h. Stream response:



```
70
97
26
92
95
48
99
76
47
40
```

i. File response:



1862

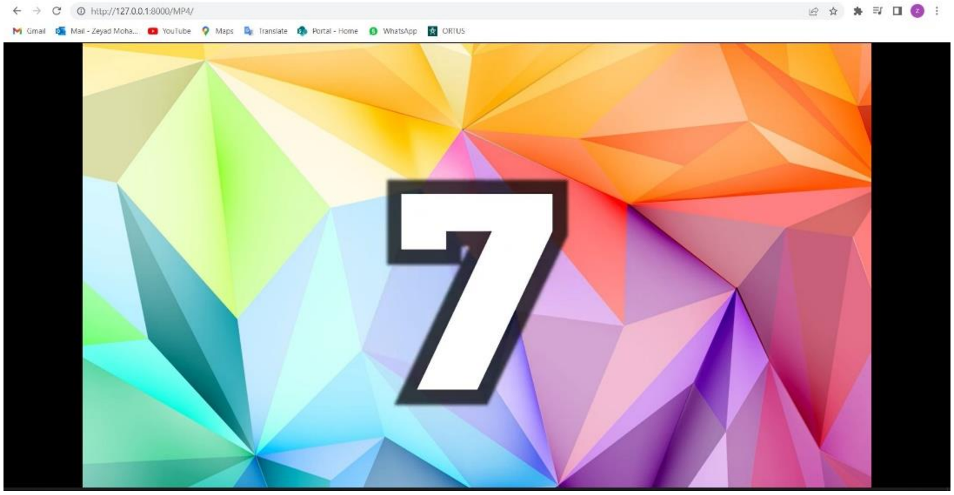
RIGA TECHNICAL
UNIVERSITY

Faculty of Electronics and Telecommunications

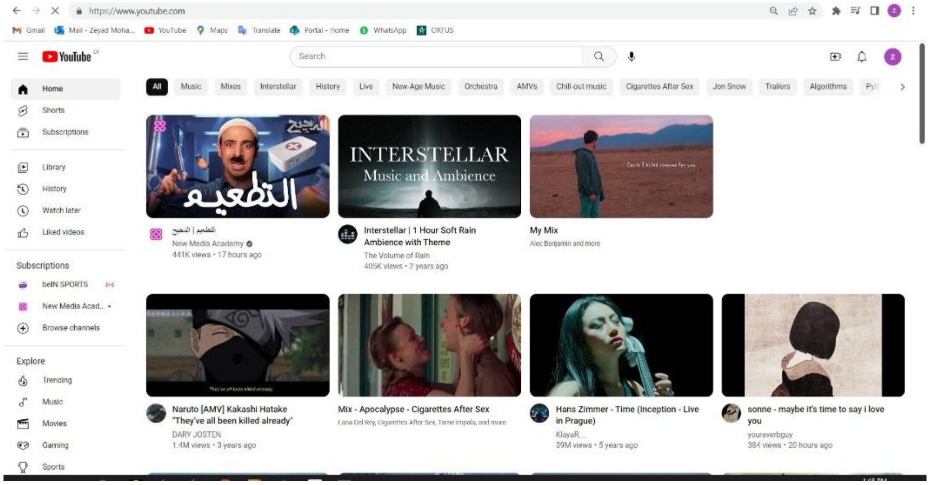
INSTITUTE OF TELECOMMUNICATIONS

RAE411 Telecommunications Software

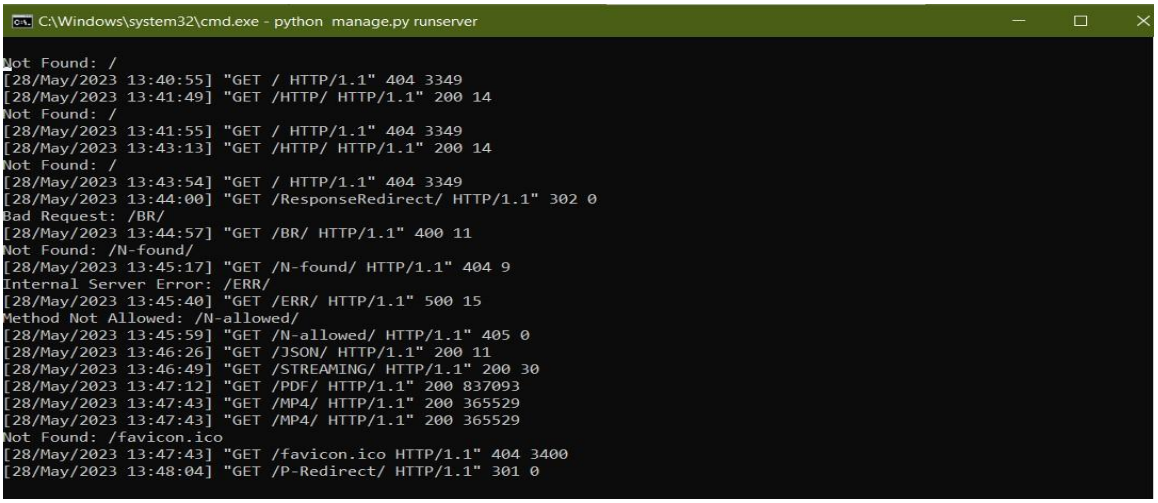
j. Video response:



k. Permanent redirect response:



For all the responses in the terminal:



```
Not Found: /
[28/May/2023 13:40:55] "GET / HTTP/1.1" 404 3349
[28/May/2023 13:41:49] "GET /HTTP/ HTTP/1.1" 200 14
Not Found: /
[28/May/2023 13:41:55] "GET / HTTP/1.1" 404 3349
[28/May/2023 13:43:13] "GET /HTTP/ HTTP/1.1" 200 14
Not Found: /
[28/May/2023 13:43:54] "GET / HTTP/1.1" 404 3349
[28/May/2023 13:44:00] "GET /ResponseRedirect/ HTTP/1.1" 302 0
Bad Request: /BR/
[28/May/2023 13:44:57] "GET /BR/ HTTP/1.1" 400 11
Not Found: /N-found/
[28/May/2023 13:45:17] "GET /N-found/ HTTP/1.1" 404 9
Internal Server Error: /ERR/
[28/May/2023 13:45:40] "GET /ERR/ HTTP/1.1" 500 15
Method Not Allowed: /N-allowed/
[28/May/2023 13:45:59] "GET /N-allowed/ HTTP/1.1" 405 0
[28/May/2023 13:46:26] "GET /JSON/ HTTP/1.1" 200 11
[28/May/2023 13:46:49] "GET /STREAMING/ HTTP/1.1" 200 30
[28/May/2023 13:47:12] "GET /PDF/ HTTP/1.1" 200 837093
[28/May/2023 13:47:43] "GET /MP4/ HTTP/1.1" 200 365529
[28/May/2023 13:47:43] "GET /MP4/ HTTP/1.1" 200 365529
Not Found: /favicon.ico
[28/May/2023 13:47:43] "GET /favicon.ico HTTP/1.1" 404 3400
[28/May/2023 13:48:04] "GET /P-Redirect/ HTTP/1.1" 301 0
```

VI. GitHub link:

https://github.com/ZeyadNashaat/Telecommunications-Software-RAE411-/tree/main/Third_Practical