

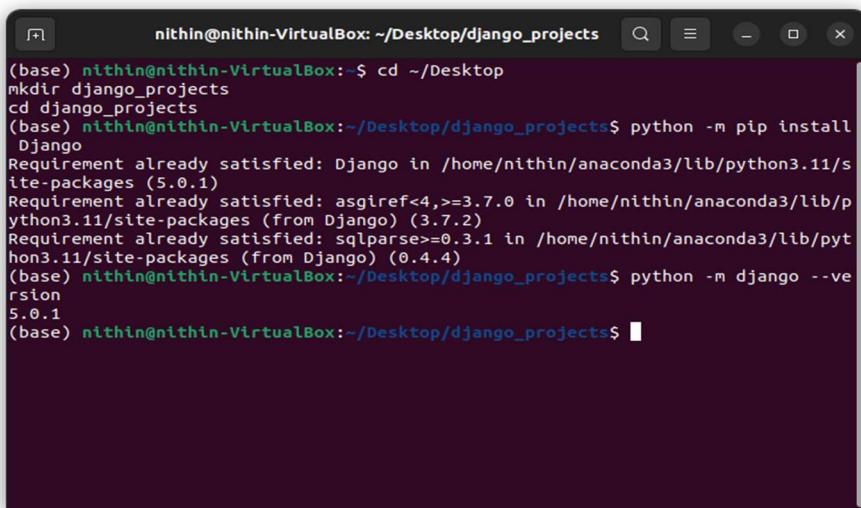
RIGA TECHNICAL UNIVERSITY  
TELECOMMUNICATIONS SOFTWARE

THIRD PRACTICAL EXERCISE

Nithin Anil  
231AEM008

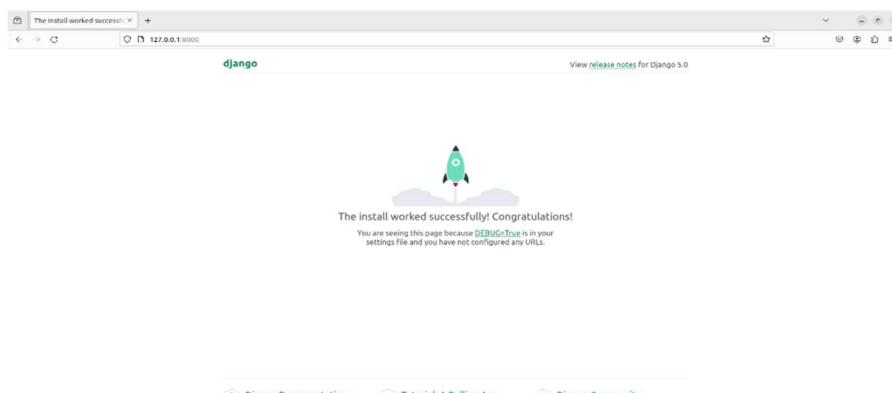
# Practical Exercise 3 (Python cloud full stack development)

1. Example1: Django's Hello World program (Tips: • This application inspection of a specific function, please return a string-related web page that can combine with CSS and JS to design your style. )
  - ❖ First created a folder “django\_project” in the desktop.
  - ❖ Installed Django.
  - ❖ Confirmed the version.



```
nithin@nithin-VirtualBox: ~/Desktop/django_projects
(base) nithin@nithin-VirtualBox:~$ cd ~/Desktop
mkdir django_projects
cd django_projects
(base) nithin@nithin-VirtualBox:~/Desktop/django_projects$ python -m pip install
Django
Requirement already satisfied: Django in /home/nithin/anaconda3/lib/python3.11/s
ite-packages (5.0.1)
Requirement already satisfied: asgiref<4,>=3.7.0 in /home/nithin/anaconda3/lib/p
ython3.11/site-packages (from Django) (3.7.2)
Requirement already satisfied: sqlparse>=0.3.1 in /home/nithin/anaconda3/lib/pyt
hon3.11/site-packages (from Django) (0.4.4)
(base) nithin@nithin-VirtualBox:~/Desktop/django_projects$ python -m django --ve
rsion
5.0.1
(base) nithin@nithin-VirtualBox:~/Desktop/django_projects$
```

- ❖ Django is successfully installed.



- ❖ Created a project “mysite”.

The screenshot shows the Spyder IDE interface. The top bar displays the path: ~/Desktop/django\_projects/mysite - Spyder (Python 3.11). The menu bar includes File, Edit, Search, Source, Run, Debug, Consoles, Projects, Tools, View, and Help. The toolbar has icons for file operations like Open, Save, and Run. The left sidebar shows the project structure:

```
mysite
├── mysite
│   ├── __init__.py
│   ├── asgi.py
│   ├── settings.py
│   ├── urls.py
│   └── wsgi.py
└── manage.py
```

The main editor window contains a temporary script named temp.py with the following code:

```
# -*- coding: utf-8 -*-
"""
Spyder Editor

This is a temporary script file.
"""


```

- ❖ Created an app “helloapp” in mysite.

```
nithin@nithin-VirtualBox: ~/Desktop/django_projects/mysite
(base) nithin@nithin-VirtualBox:~$ cd ~/Desktop
mkdir django_projects
cd django_projects
(base) nithin@nithin-VirtualBox:~/Desktop/django_projects$ python -m pip install
Django
Requirement already satisfied: Django in /home/nithin/anaconda3/lib/python3.11/s
ite-packages (5.0.1)
Requirement already satisfied: asgiref<4,>=3.7.0 in /home/nithin/anaconda3/lib/p
ython3.11/site-packages (from Django) (3.7.2)
Requirement already satisfied: sqlparse>=0.3.1 in /home/nithin/anaconda3/lib/pyt
hon3.11/site-packages (from Django) (0.4.4)
(base) nithin@nithin-VirtualBox:~/Desktop/django_projects$ python -m django --ve
rsion
5.0.1
(base) nithin@nithin-VirtualBox:~/Desktop/django_projects$ django-admin startpro
ject mysite
(base) nithin@nithin-VirtualBox:~/Desktop/django_projects$ cd mysite
(base) nithin@nithin-VirtualBox:~/Desktop/django_projects/mysite$ python manage.
py startapp helloapp
(base) nithin@nithin-VirtualBox:~/Desktop/django_projects/mysite$
```

- ❖ Entered the code to view the string, “Hello world, Django” in the views.py file of helloapp.

The screenshot shows the Spyder IDE interface with the same top bar and menu as before. The left sidebar shows the project structure:

```
mysite
├── mysite
│   ├── __init__.py
│   ├── asgi.py
│   ├── settings.py
│   ├── urls.py
│   └── wsgi.py
└── helloapp
    ├── migrations
    │   ├── __init__.py
    │   ├── admin.py
    │   ├── apps.py
    │   ├── models.py
    │   ├── tests.py
    │   └── urls.py
    └── views.py
```

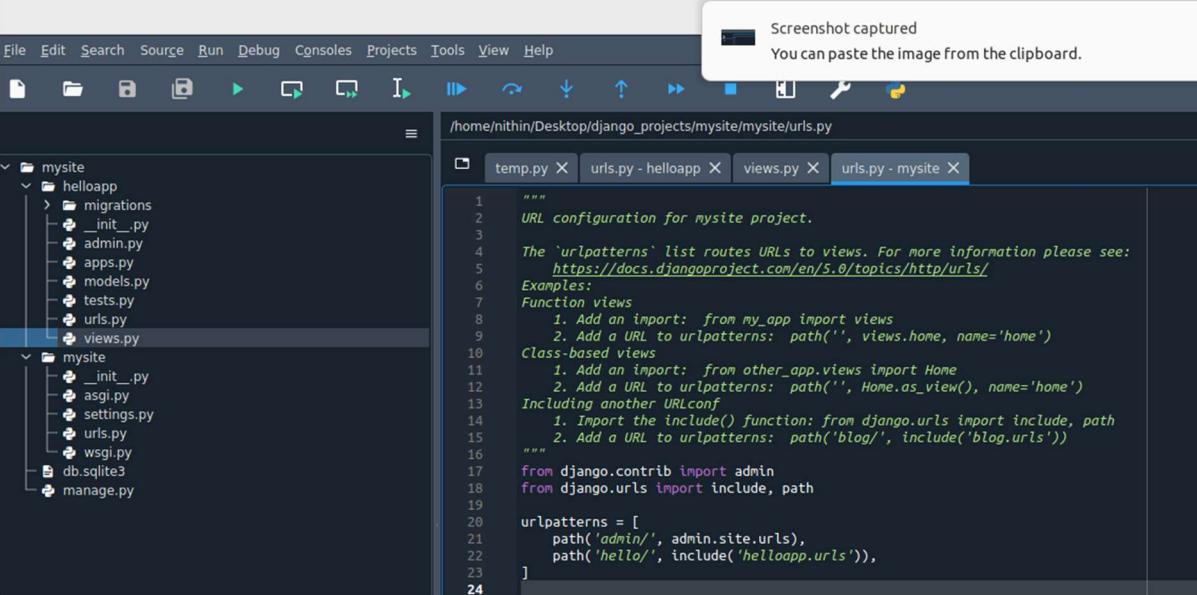
The main editor window shows the views.py file with the following code:

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

def hello(request):
    return HttpResponse("Hello world, Django!")

# Create your views here.
```

- ❖ Entered the URLs code to view the string in the urls.py file of the app and shared the path with the urls.py file of mysite folder.



The screenshot shows a code editor with the following details:

- File Explorer:** Shows the project structure under `mysite`, including `helloapp` and `mysite` apps, along with their respective `migrations`, `admin.py`, `apps.py`, `models.py`, `tests.py`, `urls.py`, and `views.py` files.
- Code Editor:** Displays the content of `urls.py` for the `mysite` app. The code is as follows:

```

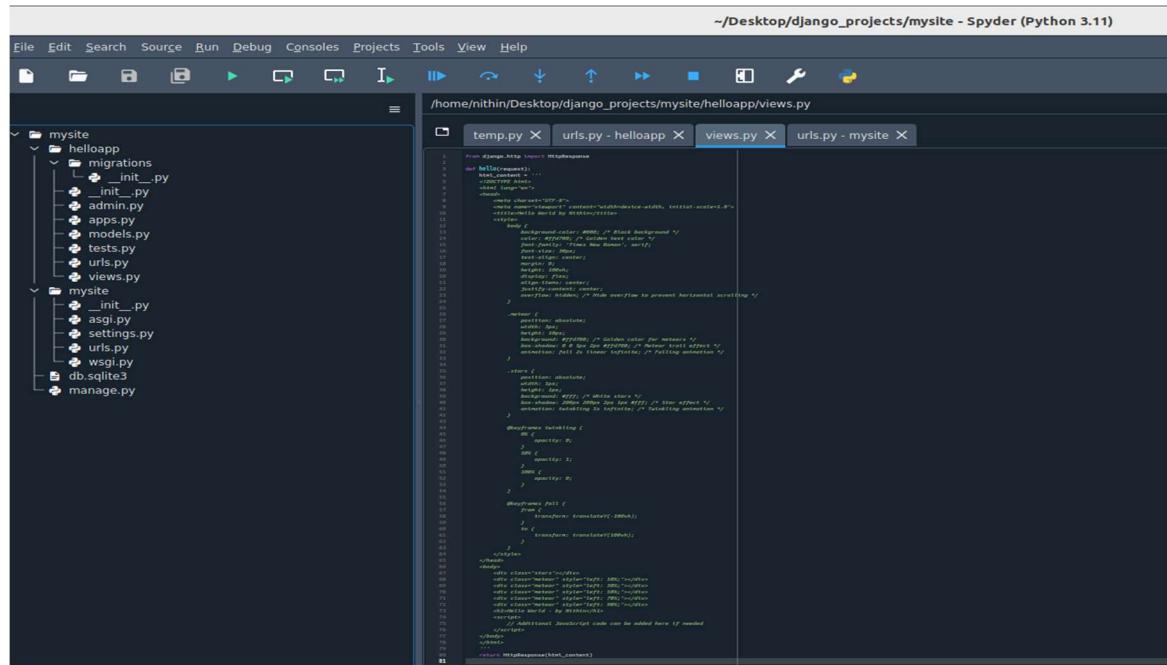
1  """
2   URL configuration for mysite project.
3
4   The `urlpatterns` list routes URLs to views. For more information please see:
5       https://docs.djangoproject.com/en/5.0/topics/http/urls/
6   Examples:
7     Function views
8         1. Add an import: from my_app import views
9             2. Add a URL to urlpatterns: path('', views.home, name='home')
10    Class-based views
11        1. Add an import: from other_app.views import Home
12            2. Add a URL to urlpatterns: path('', Home.as_view(), name='home')
13    Including another URLconf
14        1. Import the include() function: from django.urls import include, path
15            2. Add a URL to urlpatterns: path('blog/', include('blog.urls'))
16
17 from django.contrib import admin
18 from django.urls import include, path
19
20 urlpatterns = [
21     path('admin/', admin.site.urls),
22     path('hello/', include('helloapp.urls')),
23 ]

```

- ❖ Then in the project folder run the following command to run the server  
`python manage.py runserver` and copied the url <http://127.0.0.1:8000/hello/>



- ❖ Now made some changes in the views.py file to customize the web page.



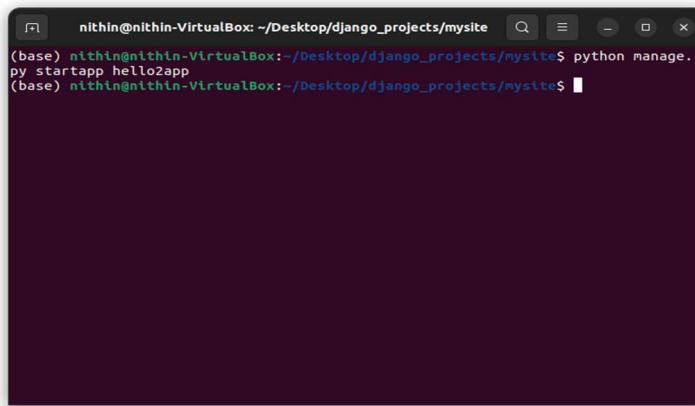
The screenshot shows the Spyder IDE interface with the following details:

- File Explorer:** Shows the project structure:
  - mysite**:
    - helloapp**:
      - migrations
      - \_\_init\_\_.py
      - admin.py
      - apps.py
      - models.py
      - tests.py
      - urls.py
      - views.py
    - mysite
    - db.sqlite3
    - manage.py
- Code Editor:** The current file is `views.py`, containing the following code:

```
from django.http import JsonResponse
def hello(request):
    return JsonResponse({
        "name": "Nithin",
        "age": 22,
        "city": "Mysore"
    })
```
- Terminal:** The terminal tab shows the command: `python manage.py runserver`.
- Browser Preview:** A screenshot of a browser window titled "Hello World by Nithin" showing the output of the code: "Hello World - by Nithin".

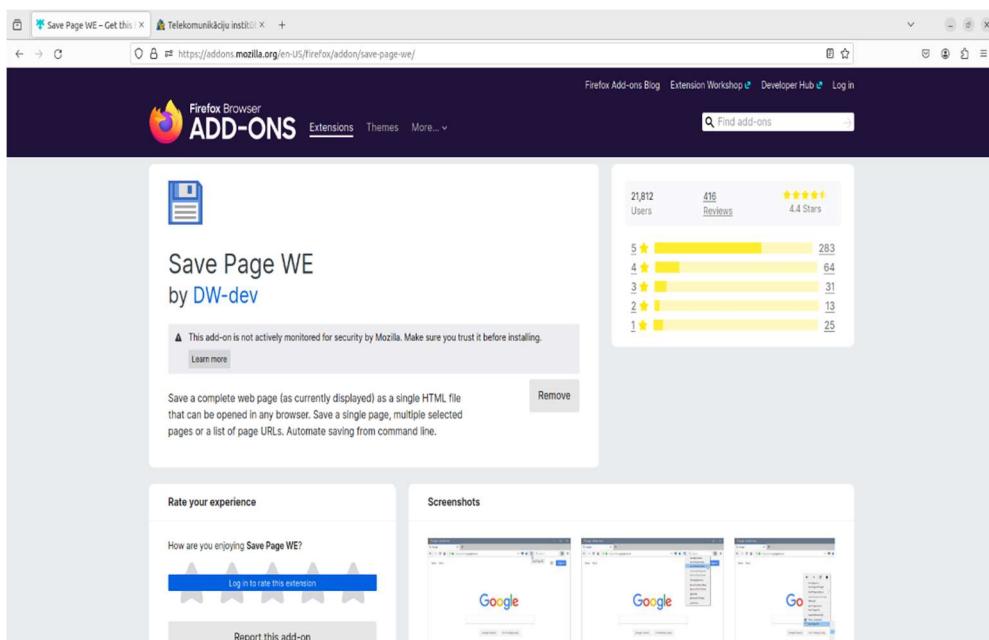
2. Example 2: Improve example 1 and return an HTML page, not a string; please use MVT (Model View Template) to design the pattern. You are welcome to use multiple decorations on your HTML page.

- ❖ Created a new app “hello2ap” in the already created project of example1.

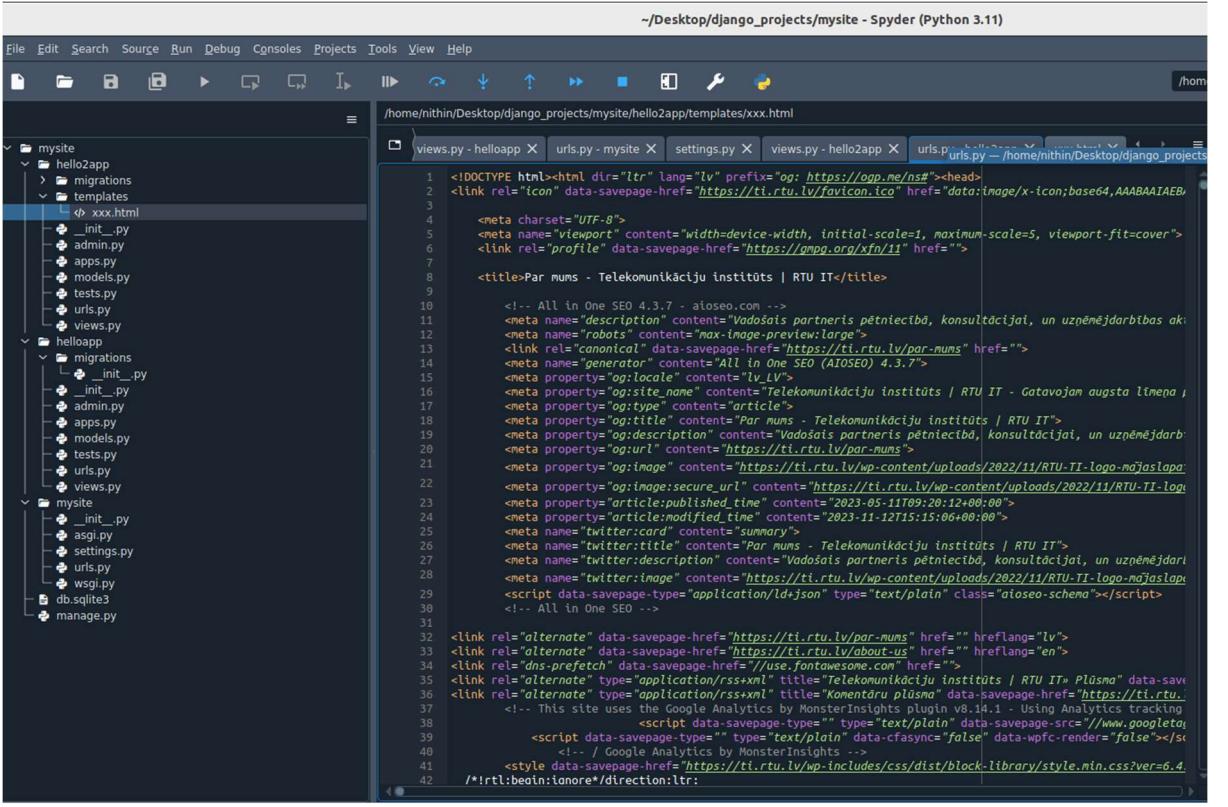


```
nithin@nithin-VirtualBox: ~/Desktop/django_projects/mysite
(base) nithin@nithin-VirtualBox:~/Desktop/django_projects/mysite$ python manage.
py startapp hello2app
(base) nithin@nithin-VirtualBox:~/Desktop/django_projects/mysite$
```

- ❖ Saved the HTML page as xxx.html using the Firefox plugin
- ❖ Open Firefox and go to check ti.rtu.lv.
- ❖ Use the Save Page WE plugin (<https://addons.mozilla.org/en-US/firefox/addon/save-page-we/>) to save the HTML page as xxx.html.



- ❖ Save the HTML file inside a new folder named "templates" within the "hello2app" directory.

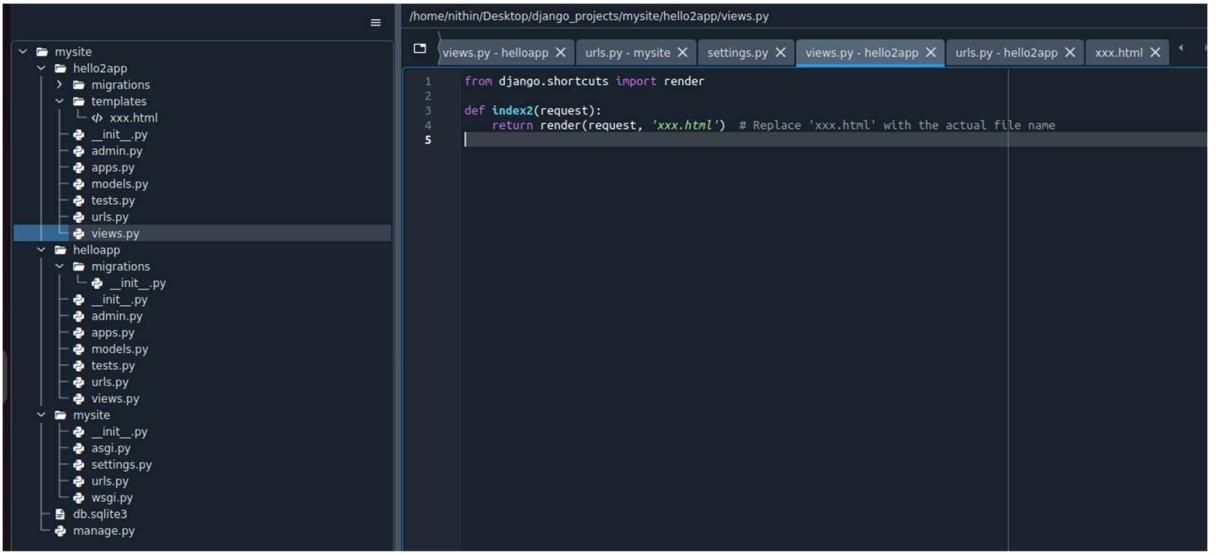


The screenshot shows the Spyder IDE interface with the following details:

- File Explorer:** Shows the project structure under "mysite". The "hello2app" directory contains "migrations", "templates" (which contains "xxx.html"), and "views.py". The "helloapp" directory contains "migrations", "views.py", and "urls.py". The "mysite" directory contains "db.sqlite3" and "manage.py".
- Code Editor:** Displays the content of "xxx.html". The code is an HTML document with meta tags for SEO, including descriptions, robots, canonical URLs, and generator information. It also includes links for alternate pages, DNS prefetching, and Google Analytics tracking.

```
<!DOCTYPE html><html dir="ltr" lang="lv" prefix="og: https://ogp.me/ns#><head>
<link rel="icon" data-savename="f" href="https://ti.rtu.lv/favicon.ico" href="data:image/x-icon;base64,AAABAAIAEB"/>
<meta name="viewport" content="width=device-width, initial-scale=1, maximum-scale=5, viewport-fit=cover">
<link rel="profile" data-savename="f" href="https://amp.org/xfn/11" href="">
<title>Par mums - Telekomunikāciju institūts | RTU IT</title>
<!-- All in One SEO 4.3.7 - aioseo.com -->
<meta name="description" content="Vadošais partneris pētniecībā, konsultācijai, un uzaņēmējdarbības akadēmiskajās iestādēs."/>
<meta name="robots" content="max-image-preview:large">
<link rel="canonical" data-savename="f" href="https://ti.rtu.lv/par-mums" href="">
<meta name="generator" content="All in One SEO (AOSEO) 4.3.7">
<meta property="og:locale" content="lv_LV">
<meta property="og:site_name" content="Telekomunikāciju institūts / RTU IT - Gatavojan augsta līmena izglītību un zinātnisko darbību."/>
<meta property="og:type" content="article">
<meta property="og:title" content="Par mums - Telekomunikāciju institūts / RTU IT">
<meta property="og:description" content="Vadošais partneris pētniecībā, konsultācijai, un uzaņēmējdarbības akadēmiskajās iestādēs."/>
<meta property="og:url" content="https://ti.rtu.lv/par-mums" href="">
<meta property="og:image" content="https://ti.rtu.lv/wp-content/uploads/2022/11/RTU-TI-logo-mājaslapa.jpg" href="">
<meta property="og:image:secure_url" content="https://ti.rtu.lv/wp-content/uploads/2022/11/RTU-TI-logo-mājaslapa.jpg" href="">
<meta property="article:published_time" content="2023-05-11T09:20:12+00:00">
<meta property="article:modified_time" content="2023-11-12T15:15:06+00:00">
<meta name="twitter:card" content="summary">
<meta name="twitter:title" content="Par mums - Telekomunikāciju institūts / RTU IT">
<meta name="twitter:description" content="Vadošais partneris pētniecībā, konsultācijai, un uzaņēmējdarbības akadēmiskajās iestādēs."/>
<meta name="twitter:image" content="https://ti.rtu.lv/wp-content/uploads/2022/11/RTU-TI-logo-mājaslapa.jpg" href="">
<script data-savename="f" type="application/json" href="https://ti.rtu.lv/wp-content/themes/rtu/rtu/applications/all-in-one-seo-schema.js" type="text/plain" class="aioseo-schema"></script>
<!-- All in One SEO -->
<link rel="alternate" data-savename="f" href="https://ti.rtu.lv/par-mums" hreflang="lv">
<link rel="alternate" data-savename="f" href="https://ti.rtu.lv/about-us" href="" hreflang="en">
<link rel="dns-prefetch" data-savename="f" href="use.fontawesome.com" href="">
<link rel="alternate" type="application/rss+xml" title="Telekomunikāciju institūts / RTU IT Plūsmas" data-savename="f" href="https://ti.rtu.lv/feed/">
<link rel="alternate" type="application/rss+xml" title="Komentaru plūsmas" data-savename="f" href="https://ti.rtu.lv/comments/">
<!-- This site uses the Google Analytics by MonsterInsights plugin v8.14.1 - Using Analytics tracking -->
<script data-savename="f" type="text/plain" href="https://ti.rtu.lv/wp-content/plugins/monsterinsights/js/min/Analytics.min.js" data-savename="f" data-wpfc-render="false"></script>
<!-- / Google Analytics by MonsterInsights -->
<style data-savename="f" href="https://ti.rtu.lv/wp-includes/css/dist/block-library/style.min.css?ver=6.4" href=""></style>
/*!rtl:begin*!ignore*/direction:rtl;
```

- ❖ Modified the views.py file with the following code:



The screenshot shows the Spyder IDE interface with the following details:

- File Explorer:** Shows the project structure under "mysite". The "hello2app" directory contains "migrations", "templates" (which contains "xxx.html"), and "views.py". The "helloapp" directory contains "migrations", "views.py", and "urls.py". The "mysite" directory contains "db.sqlite3" and "manage.py".
- Code Editor:** Displays the content of "views.py". It contains a single function "index2" that imports "render" from "django.shortcuts" and returns the rendered template "xxx.html".

```
from django.shortcuts import render
def index2(request):
    return render(request, 'xxx.html') # Replace 'xxx.html' with the actual file name
```

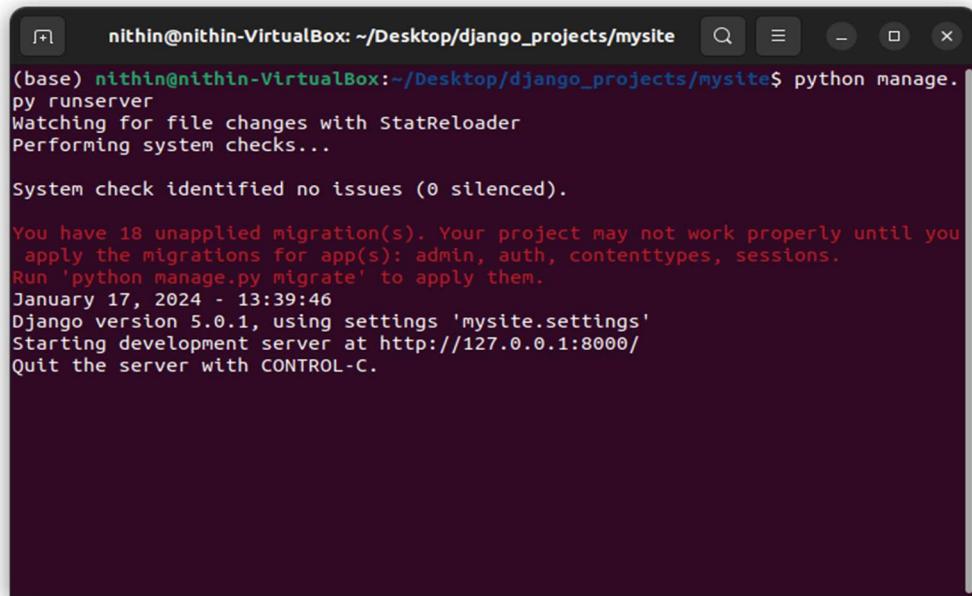
- ❖ Then changed the urls.py file for both mysite and hello2app.

The image shows two separate Spyder IDE windows side-by-side. Both windows have a dark theme and are displaying Python code in their main panes. The left window's title bar says "File Edit Search Source Run Debug Consoles Projects Tools View Help" and its status bar says "~/Desktop/django\_projects/mysite - Spyder (Python 3.11)". The right window has a similar interface with its own title bar and status bar. Both windows show the directory structure of a Django project with apps named "mysite" and "hello2app". The left window displays the contents of "mysite/urls.py", which includes code for including "admin.site.urls" and "hello2app.urls". The right window displays the contents of "hello2app/urls.py", which includes a single URL pattern for "xxx/" pointing to "views.index2".

- ❖ Set the template path in settings.py

This screenshot shows a single Spyder IDE window with a dark theme. The title bar reads "File Edit Search Source Run Debug Consoles Projects Tools View Help" and the status bar says "~/Desktop/django\_projects/mysite - Spyder (Python 3.11)". The main pane contains the "settings.py" file from the "mysite" app. The "TEMPLATES" section is highlighted with a blue selection bar. The code in the "TEMPLATES" section includes the "BACKEND", "DIRS", "APP\_DIRS", and "OPTIONS" parameters, with the "OPTIONS" part being specifically highlighted.

- ❖ Now run the following command



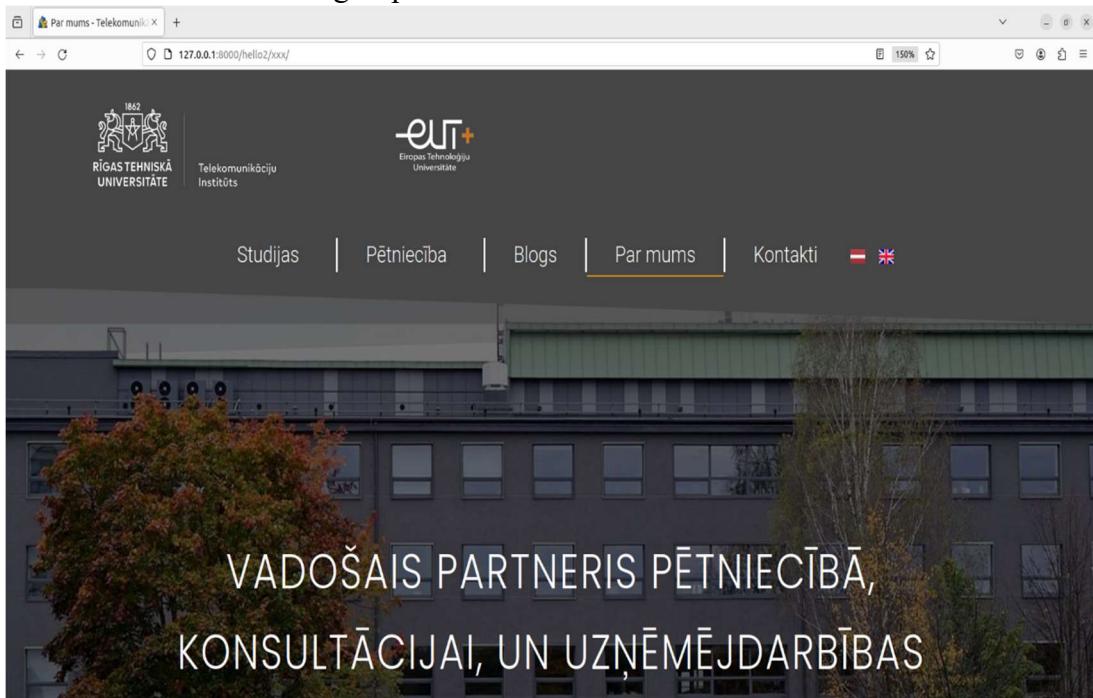
```
nithin@nithin-VirtualBox: ~/Desktop/django_projects/mysite$ python manage.py runserver
Watching for file changes with StatReloader
Performing system checks...

System check identified no issues (0 silenced).

You have 18 unapplied migration(s). Your project may not work properly until you
apply the migrations for app(s): admin, auth, contenttypes, sessions.
Run 'python manage.py migrate' to apply them.

January 17, 2024 - 13:39:46
Django version 5.0.1, using settings 'mysite.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CONTROL-C.
```

- ❖ And browse the following <http://127.0.0.1:8000/hello2/xxx/>



- ❖ I also tried to create some other webpage html file and return it.



### 3. Example 3: Cloud message board.

- Create another new Django project

```
nithin@nithin-VirtualBox:~/Desktop/cloudproject/cloudm$ python manage.py runserver
Watching for file changes with StatReloader
Performing system checks...

System check identified no issues (0 silenced).
January 21, 2024 - 14:20:18
Django version 5.0.1, using settings 'cloudm.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CONTROL-C.

Not Found: /hello/
[21/Jan/2024 14:20:38] "GET /hello/ HTTP/1.1" 404 2203
Not Found: /favicon.ico
[21/Jan/2024 14:20:52] "GET /msgapp/ HTTP/1.1" 200 1422
[21/Jan/2024 14:21:08] "POST /msgapp/submit_message/ HTTP/1.1" 200 21
[21/Jan/2024 14:21:19] "POST /msgapp/get_messages/ HTTP/1.1" 200 327
```

- Creating an HTML file for the message board display. Let's name it **message\_board.html**. Save it inside a new folder named "templates" within the "msgapp" directory.

The screenshot shows the Spyder IDE interface with the following details:

- File Explorer:** Shows the project structure under `/home/nithin/Desktop/cloudproject/cloudms`. It includes files like `__init__.py`, `asgi.py`, `settings.py`, `urls.py`, `wsgi.py`, and `message_board.html` within the `msgapp` directory.
- Code Editor:** Displays the content of `message_board.html`. The code is an HTML template for a message board. It includes sections for submitting messages, getting messages, and displaying messages. It uses Django's template language (e.g., `{% csrf_token %}`) and HTML forms.
- Toolbar:** Standard Spyder toolbar with icons for file operations, run, debug, and help.
- Status Bar:** Shows the path `~/Desktop/cloudproject/cloudms - Spyder (Python 3.11)`.

- Enter the codes for `views.py` to view our web page and also enter the `urls.py` codes in `msgapp` and combine it with the main `cloudms`.

The screenshot shows the Spyder IDE interface with the following details:

- File Explorer:** Shows the project structure under `/home/nithin/Desktop/cloudproject/cloudms`. It includes files like `__init__.py`, `asgi.py`, `settings.py`, `urls.py`, `wsgi.py`, and `message_board.html` within the `msgapp` directory.
- Code Editor:** Displays the content of `message_board.html`, which is identical to the one shown in the previous screenshot.
- Toolbar:** Standard Spyder toolbar with icons for file operations, run, debug, and help.
- Status Bar:** Shows the path `~/Desktop/cloudproject/cloudms - Spyder (Python 3.11)`.

~/Desktop/cloudproject/cloudms - Spyder (Python 3.11)

File Edit Search Source Run Debug Consoles Projects Tools View Help

/home/nithin/Desktop/cloudproject/cloudms/msgapp/models.py

temp.py urls.py - msgapp urls.py - cloudms views.py models.py settings.py

```
1 # msgapp/models.py
2
3 from django.db import models
4
5 class Message(models.Model):
6     sender = models.CharField(max_length=50)
7     receiver = models.CharField(max_length=50)
8     content = models.TextField()
9     timestamp = models.DateTimeField(auto_now_add=True)
10
11     def __str__(self):
12         return f'{self.sender} to {self.receiver} - {self.timestamp}'
```

~/Desktop/cloudproject/cloudms - Spyder (Python 3.11)

File Edit Search Source Run Debug Consoles Projects Tools View Help

/home/nithin/Desktop/cloudproject/cloudms/msgapp/urls.py

temp.py urls.py - msgapp urls.py - cloudms views.py models.py settings.py

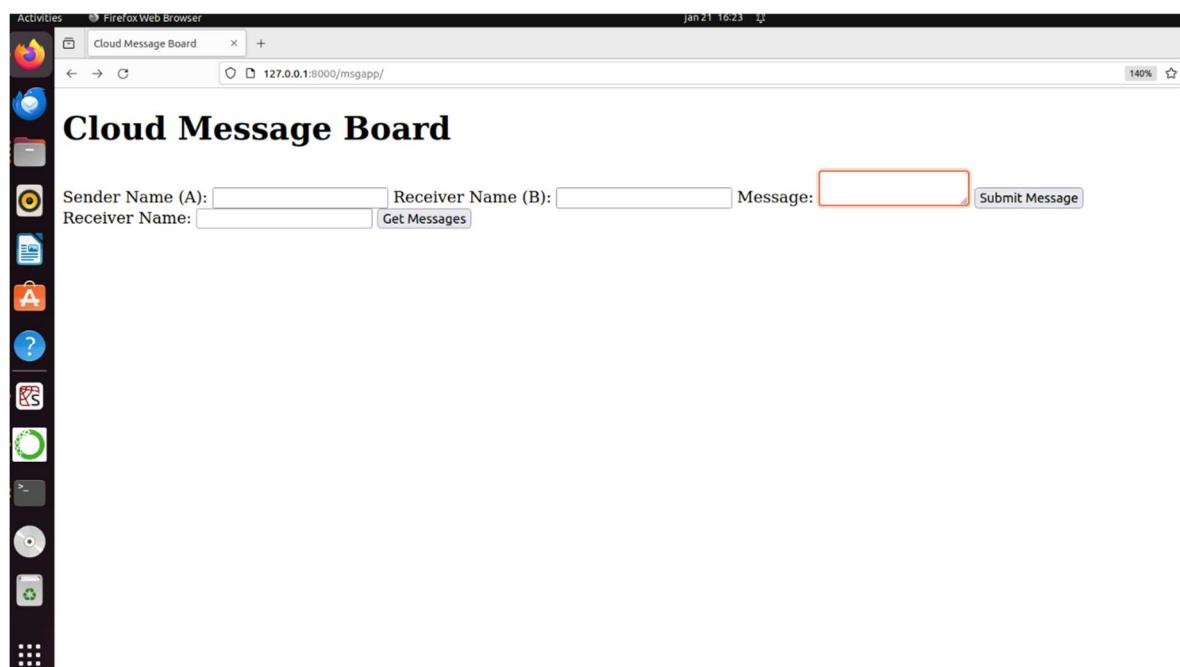
```
1 #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3 """
4 Created on Sun Jan 21 15:02:26 2024
5
6 @author: nithin
7 """
8
9 # msgapp/urls.py
10
11 from django.urls import path
12 from . import views
13
14 urlpatterns = [
15     path('', views.message_board, name='message_board'),
16     path('submit_message/', views.submit_message, name='submit_message'),
17     path('get_messages/', views.get_messages, name='get_messages'),
18 ]
19
```

```

1 # msgapp/views.py
2
3 from django.shortcuts import render
4 from django.http import JsonResponse
5 from .models import Message
6
7 def message_board(request):
8     return render(request, 'msgapp/message_board.html')
9
10 def submit_message(request):
11     # Logic to save the submitted message to the database (you need to create the Model)
12     sender = request.POST.get('sender')
13     receiver = request.POST.get('receiver')
14     message_content = request.POST.get('message')
15
16     # Save the message to the database
17     message = Message.objects.create(sender=sender, receiver=receiver, content=message_content)
18
19     return JsonResponse({'status': 'success'})
20
21 def get_messages(request):
22     # Logic to retrieve the latest 20 messages for a specific receiver from the database
23     receiver_name = request.GET.get('get_receiver')
24     messages = Message.objects.filter(receiver=receiver_name).order_by('-timestamp')[0:20]
25
26     # Return the messages as JSON
27     messages_data = [{"source": msg.sender, "time": msg.timestamp, "content": msg.content} for msg in messages]
28
29     return JsonResponse({'messages': messages_data})

```

- Now run the server and use this url to enter the web site.  
<http://127.0.0.1:8000/msgapp/>



The screenshot shows a desktop environment with a docked application menu on the left. Two Firefox browser windows are open:

- Top Window:** Title bar says "Cloud Message Board". Address bar shows "127.0.0.1:8000/msgapp/". Content area displays a form:
  - Sender Name (A): Megha
  - Receiver Name (B): Nithin
  - Message: How are you?
  - Submit Message button

Below the form, it says "Receiver Name: Nithin" and "Get Messages".
- Bottom Window:** Title bar shows "127.0.0.1:8000/msgapp/get\_messages/". Address bar shows "127.0.0.1:8000/msgapp/get\_messages/". Content area displays a JSON response:

```
JSON Raw Data Headers
Save Copy Collapse All Expand All Filter JSON
{
  "messages": [
    {
      "source": "Megha",
      "time": "2024-01-21T14:24:38.081Z",
      "content": "How are you?"
    }
  ]
}
```

The image shows a terminal window with two separate JSON responses. Both responses are for the URL `127.0.0.1:8000/msgapp/get_messages/`.

**Response 1:**

```
127.0.0.1:8000/msgapp/get_messages/
{
  "messages": [
    {
      "0": {
        "source": "albin",
        "time": "2024-01-21T14:26:06.486Z",
        "content": "helloooooo"
      },
      "1": {
        "source": "Megha",
        "time": "2024-01-21T14:24:38.081Z",
        "content": "How are you?"
      }
    ]
}
```

**Response 2:**

```
127.0.0.1:8000/msgapp/get_messages/
{
  "messages": [
    {
      "0": {
        "source": "Nithin",
        "time": "2024-01-21T13:59:53.073Z",
        "content": "hi hello\r\n"
      },
      "1": {
        "source": "Nithin",
        "time": "2024-01-21T13:59:07.333Z",
        "content": "hi"
      }
    }
}
```

4. Example 4: Please try to test Django's different response types, including `HttpResponse` class and subclasses (10 in total), `JsonResponse` class, `StreamingHttpResponse`, and `FileResponse` class. Please show those response files, JSON, or video on your screenshot.

❖ Create a New Django Project and App

```

nithin@nithin-VirtualBox: ~/Desktop/testingdjango/response...
(base) nithin@nithin-VirtualBox:~/Desktop$ mkdir testingdjango
(base) nithin@nithin-VirtualBox:~/Desktop$ cd testingdjango
(base) nithin@nithin-VirtualBox:~/Desktop/testingdjango$ django-admin startproject response_test
cd response_test
python manage.py startapp testapp
(base) nithin@nithin-VirtualBox:~/Desktop/testingdjango/response_test$ python manage.py runserver
Watching for file changes with StatReloader
Performing system checks...

System check identified no issues (0 silenced).

You have 18 unapplied migration(s). Your project may not work properly until you
apply the migrations for app(s): admin, auth, contenttypes, sessions.
Run 'python manage.py migrate' to apply them.
Error: That port is already in use.
(base) nithin@nithin-VirtualBox:~/Desktop/testingdjango/response_test$ python manage.py runserver
Watching for file changes with StatReloader
Performing system checks...

System check identified no issues (0 silenced).

```

❖ Define View Functions

```

# /home/nithin/Desktop/testingdjango/response_test/testapp/views.py
temp.py views.py urls.py - testapp urls.py - response_test admin.py
1 # testapp/views.py
2 from django.http import HttpResponse, HttpResponseRedirect, JsonResponse, PermanentRedirect, Http404
3 from django.http import Http404, HttpResponseRedirect, HttpResponseRedirect, HttpResponseRedirect, JsonResponse, StreamingHttpResponse, FileResponse
4
5 def normal_response(request):
6     return HttpResponse("This is a normal HTTP response.")
7
8 def redirect_response(request):
9     return HttpResponseRedirect('/response_test/testapp/normal_response/')
10
11 def permanent_redirect_response(request):
12     return HttpResponseRedirectPermanentRedirect('/response_test/testapp/normal_response/')
13
14 def not_modified_response(request):
15     return HttpResponseNotModified()
16
17 def bad_request_response(request):
18     return HttpResponseBadRequest("Bad Request")
19
20 def forbidden_response(request):
21     return HttpResponseForbidden("Forbidden")
22
23 def not_allowed_response(request):
24     return HttpResponseNotAllowed([ 'GET', 'POST' ], "Method Not Allowed")
25
26 def gone_response(request):
27     return HttpResponseGone("Gone")
28
29 def server_error_response(request):
30     return HttpResponseServerError("Internal Server Error")
31
32 def not_found_response(request):
33     return HttpResponseNotFound("Page Not Found")
34
35 def json_response(request):
36     data = { 'message': 'This is a JSON response.' }
37     return JsonResponse(data)
38
39 def streaming_response(request):
40     def streaming_content():
41         yield 'Part 1 of the content.'
42         yield b'Part 2 of the content.'
43
44     return StreamingHttpResponse(streaming_content(), content_type="text/plain")
45
46 def file_response(request):
47     file_path = 'path/to/your/file.txt' # Replace with the actual file path
48     return FileResponse(open(file_path, 'rb'))
49

```

❖ Configure URLs and Include App URLs in Project URLs

```

# testapp/urls.py
from django.urls import path
from .views import *

urlpatterns = [
    path('normal_response/', normal_response),
    path('redirect_response/', redirect_response),
    path('permanent_redirect_response/', permanent_redirect_response),
    path('not_modified_response/', not_modified_response),
    path('bad_request_response/', bad_request_response),
    path('forbidden_response/', forbidden_response),
    path('not_allowed_response/', not_allowed_response),
    path('gone_response/', gone_response),
    path('server_error_response/', server_error_response),
    path('not_found_response/', not_found_response),
    path('json_response/', json_response),
    path('streaming_response/', streaming_response),
    path('file_response/', file_response),
]

```

❖ Run the Server

```

nithin@nithin-VirtualBox: ~/Desktop/testingdjango/response...
^C(base) nithin@nithin-VirtualBox:~/Desktop/testingdjango/response... python manage.py runserver
Watching for file changes with StatReloader
Performing system checks...
System check identified no issues (0 silenced).

You have 18 unapplied migration(s). Your project may not work properly until you
apply the migrations for app(s): admin, auth, contenttypes, sessions.
Run 'python manage.py migrate' to apply them.

January 17, 2024 - 22:53:36
Django version 5.0.1, using settings 'response_test.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CONTROL-C.

[17/Jan/2024 22:53:55] "GET /response_test/testapp/normal_response/ HTTP/1.1" 200 31
[17/Jan/2024 22:55:33] "GET /response_test/testapp/redirect_response/ HTTP/1.1" 302 0
[17/Jan/2024 22:55:33] "GET /response_test/testapp/normal_response/ HTTP/1.1" 200 31
Bad Request: /response_test/testapp/bad_request_response/
[17/Jan/2024 22:56:38] "GET /response_test/testapp/bad_request_response/ HTTP/1.1" 400 11

```

- ❖ Outputs for all the responses
- **HttpResponse** and its subclasses (10 in total):
    - **HttpResponse**
    - **HttpResponseRedirect**
    - **HttpResponsePermanentRedirect**
    - **HttpResponseNotModified**
    - **HttpResponseBadRequest**
    - **HttpResponseForbidden**
    - **HttpResponseNotAllowed**
    - **HttpResponseGone**
    - **HttpResponseServerError**
    - **HttpResponseNotFound**
    - **JsonResponse** class.
    - **StreamingHttpResponse** class.
    - **FileResponse** class.



