Django

Getting Python in the web

Prerequisites

Objectives

Outline

- Introduction
- Project vs. App structure in Django
- Views and Templates
- Models and Databases
- Forms and User Input
- Admin
- Authentication and User Management
- Advanced Concepts

Introduction

Introduction

- Django is a high-level open-source web framework written in Python. It follows the Model-View-Controller (MVC) architectural pattern, commonly referred to as Model-View-Template (MVT) in Django. Django's primary goal is to simplify the development of complex web applications by providing a robust set of tools and functionalities.

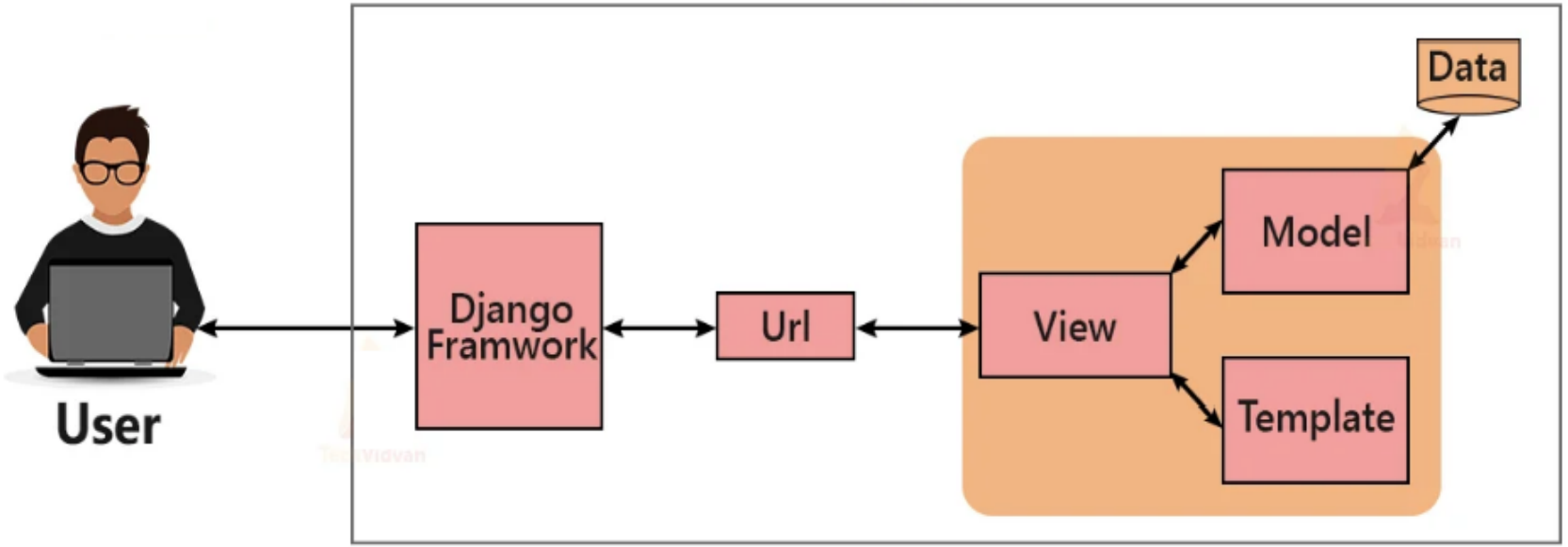




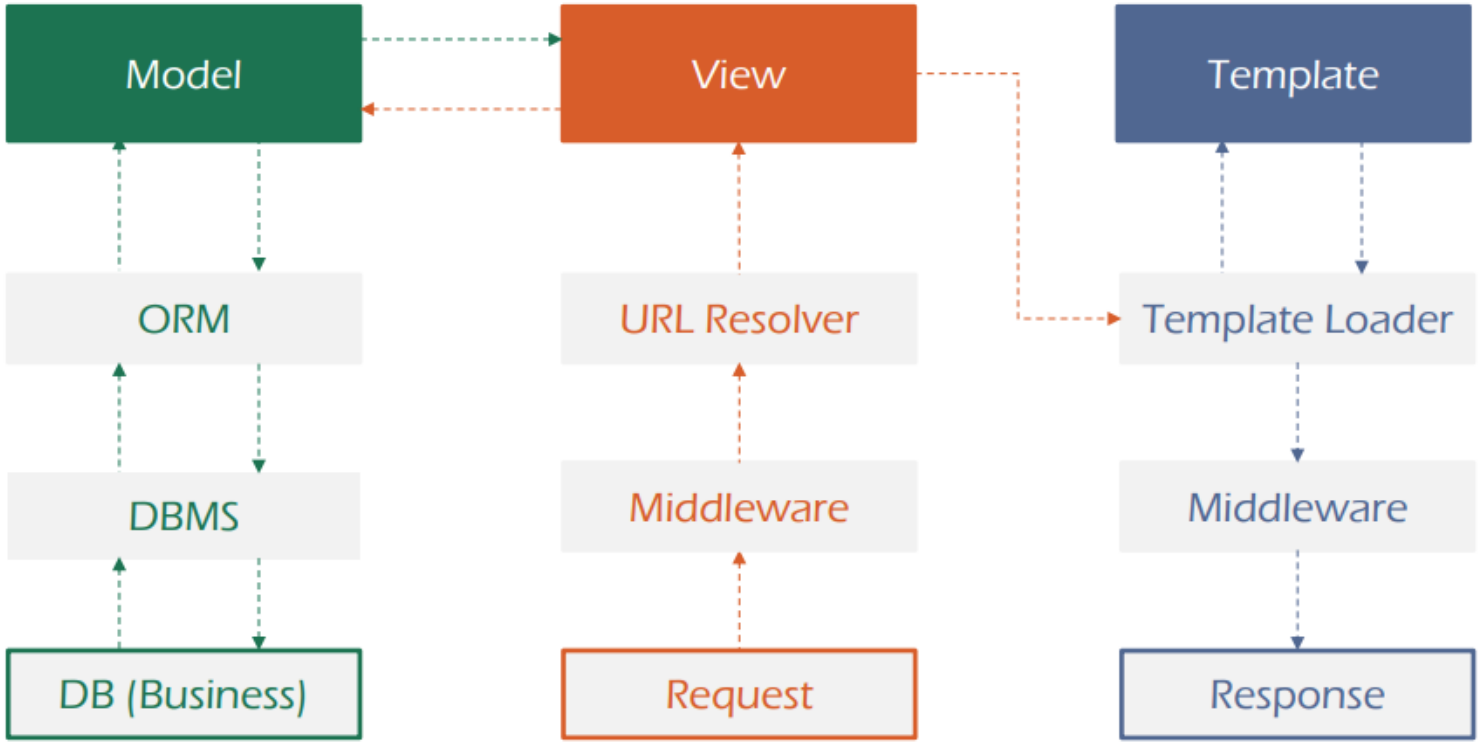
Key features

- Object-Relational Mapping (ORM)
- URL routing
- Template system
- Forms handling
- Authentication and authorization.
- Admin interface
- Security features
- Scalability and extensibility

Architecture



Architecture



Installatoin

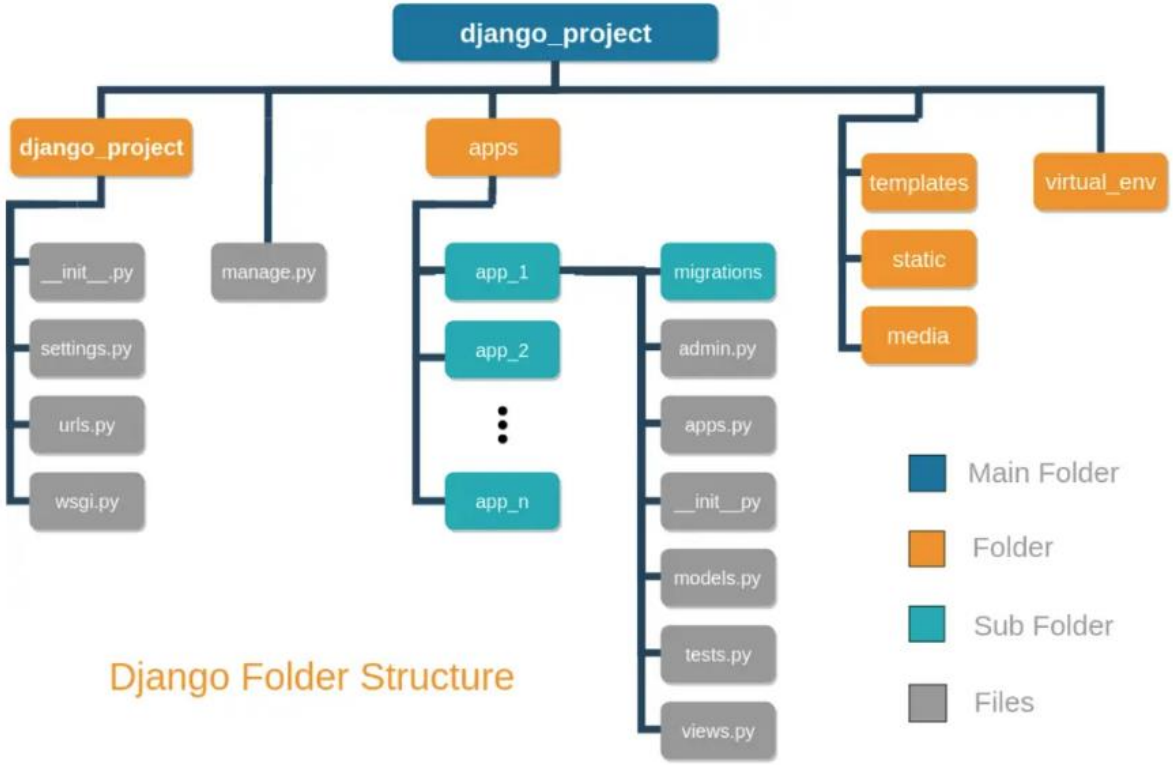
Create a virtual environment

- python -m venv myenv
- myenv\Scripts\activate

Install django & start project

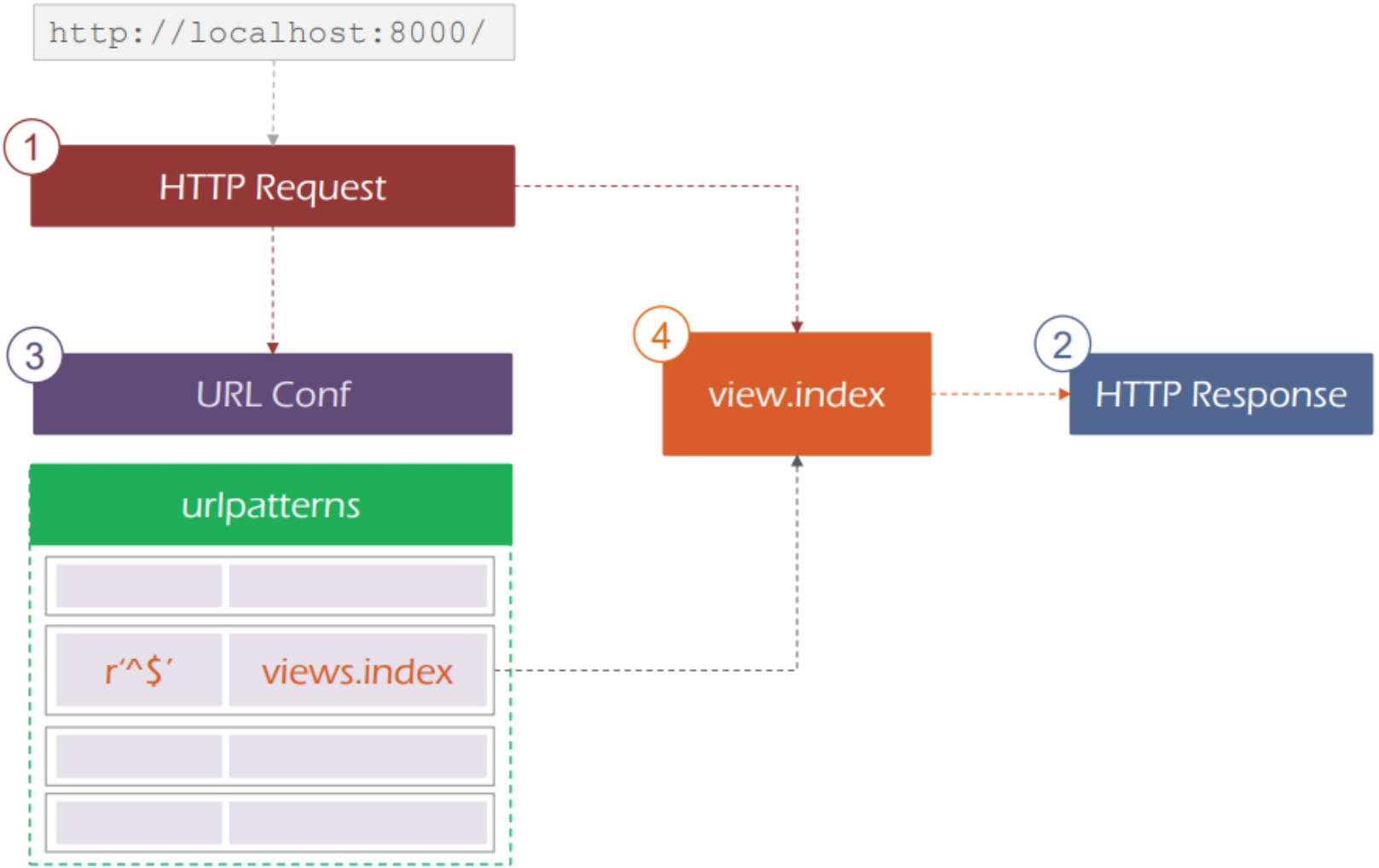
- pip install Django
- django-admin startproject projectname
- python manage.py startapp applicationname

Project & Application Structure



Django Folder Structure

Routes


http://localhost:8000/
1
HTTP Request
3
URL Conf
urlpatterns
r'^$'
views.index
4
view.index
2
HTTP Response

Urls.py

```
path('',Books,name=Bookslist),

path('insert',Bookinsert),

path('Update/<id>/',Update,name='updateBook'),

path('Delete/<id>/',Delete,name='deleteBook'),
```

View

Views.py

```
def Books(request):

def Bookinsert(request):

def Update(request,id):

def Delete(request,id):
```

HttpRequest Attributes

`body`	An attribute that contain the request body
`method`	An attribute that contain the request method
`path`	An attribute that contain the request path
`GET`	An attribute that contain the GET request parameters
`POST`	An attribute that contain the POST request parameters
`COOKIES`	An attribute that contain the cookies
`FILES`	An attribute that contain the request File objects
`META`	An attribute that contain the request headers.

HttpRequest Methods

`get_host`	Return the Host name of the request
`is_ajax`	Return True if the request was made by XMLHttpRequest
`is_secure`	Return True if the request was made by https
`get_signed_cookie(key, salt='')`	Get the value of the signed(with salt) cookie

HttpResponse-Content

views.py

```
from django.http import HttpResponse

def index(request):

        res = HttpResponse("Hello World")

        res.write('<p> Hello Open Source </p>')

        return res
```

Hello World

Hello Open Source

HttpResponse-Headers

views.py

```
from django.http import HttpResponse

def index(request):

        res = HttpResponse("Hello World")

        res.write('<p> Hello Open Source </p>')

        res['content-type'] = 'text/plain'

        return res
```

Hello World Hello Open Source

HttpResponse-Cookies

views.py

```
from django.http import HttpResponse

def index(request):

        res = HttpResponse("Hello World")

        res.write('<p> Hello Open Source </p>')

        res['content-type'] = 'text/plain'

        res.set_cookie('name', 'Ahmed')

        return res
```

HttpResponse-JsonResponse

views.py

```
from django.http import JsonResponse

def index(request):

        return JsonResponse({'name': 'Ahmed'})
```

HttpResponse-HTTPResponseRedirect

views.py

```
from django.http import HttpResponseRedirect, HttpResponse

def index(request): #view for /

        return HttpResponseRedirect("/posts")


def get_posts(request): #view for /posts

        return HttpResponse ("Here are your posts")
```

Here are your posts

Redirect

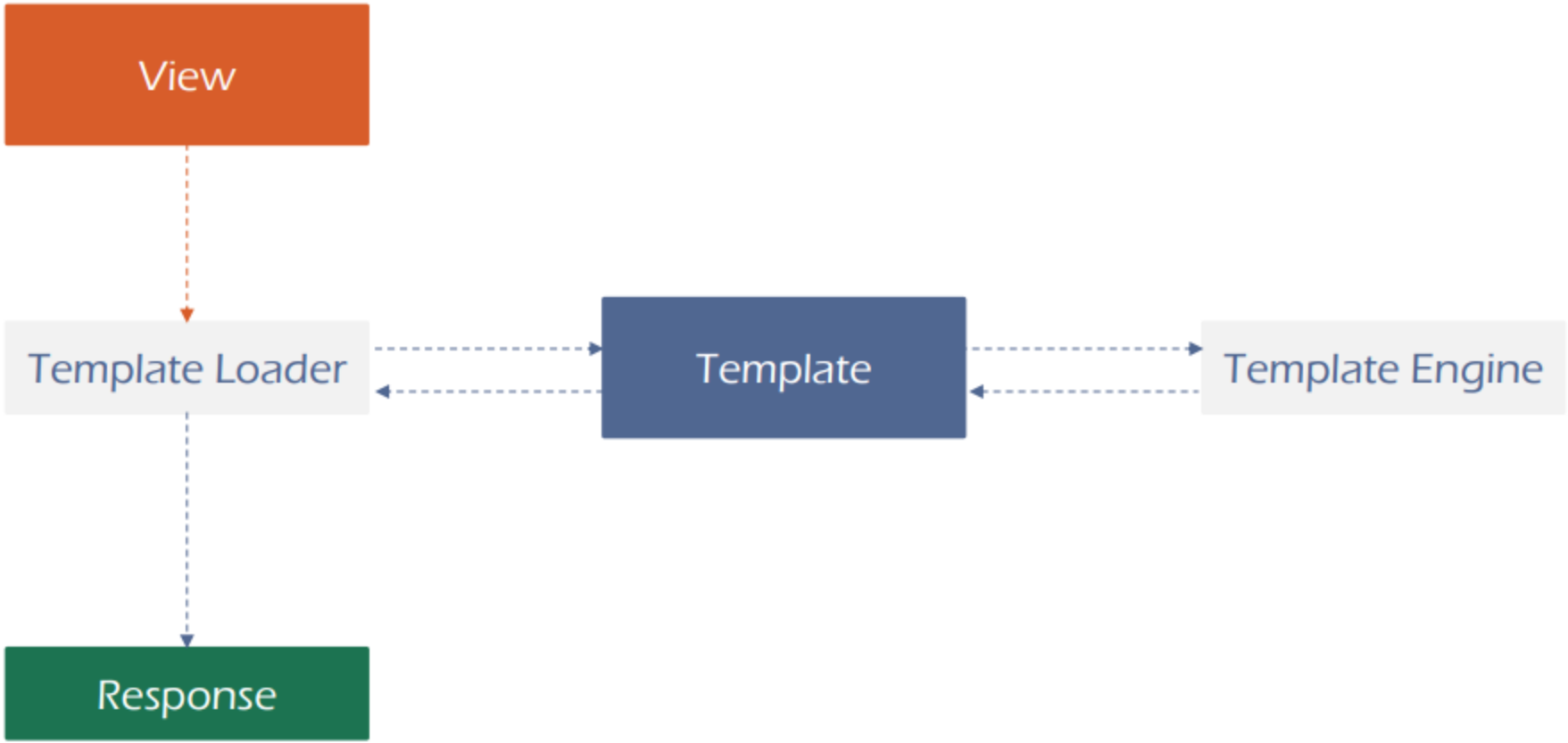
```
redirect(to, *args, **kargs)
```

views.py

```
from django.shortcuts import redirect

# view.py
def index(request):
    # Your View Body and Actions
    return redirect('/persons', {'name': 'Ahmed'})
```

Templates


View
Template Loader
Template
Template Engine
Response

Settings

mysite/settings.py

```
TEMPLATES = [
    {
        #Define the Template Engine for the following templates
        'BACKEND':'django.template.backends.django.DjangoTemplates',
        #Define the directories that that the loader will search in
         'DIRS': [],
          #Define if the loader search in installed apps or not.
        'APP_DIRS': True,
    },
]
```

Static Files Management

settings.py

```
INSTALLED_APPS = [
        # Other Apps,
        "django.contrib.staticfiles"
]
STATIC_URL = '/static/'
STATICFILES_DIRS = [
    os.path.join(BASE_DIR, "static"),
]
```

Static Files Management

In your app create a directory called `static` and inside it put your app related static files

library/index.html

```
{% load static %}

<img src="{% static "library/example.jpg" %}" alt="My image"/>
```

Template Language-Variables

```
{{ variable_name }}
```

library/index.html

```
<html>
  <head> ... </head>
  <body>
      <p> Hello, {{ name }} </p>
  </body>
</html>
```

```
Hello, Ahmed
```

library/views.py

```
from django.shortcuts import render
def index(request):
    return render(request, 'library/index.html', {'name': 'Ahmed'})
```

Template Language-Tags

```
{% for .. in ..%} {% empty %} {% endfor %}
```

```
{% if %} {% elif %} {% else %} {% endif %}
```

Template Language-Tags

```
<a href={{ url 'posts' }} >Posts</a>
```

```
{% csrf_token %}
```

```
{{ value | filter : options }}
```

Template Language-Tags

Filter	value	Example	Output
add	value =3	{{ value | add : 3 }}	6
first	value = [1,2,4]	{{ value | first }}	1
join	value =['a','b','c']	{{ value | join: ':' }}	a:b:c
linebreaks	value ="Hi\nOS"	{{ value | linebreaks }}	Hi OS
pluralize	v = 4	{{ v }} value{{ v | pluralize}}	4 values

Template Language-Comment

```
{# write your comment #}
```

Template Inheritance

```
{% block %} {% endblock %} .... {% extends %}
```

student.html

```
{% extends "base.html" %}

{% block title %}Student Page{% end %}

{% block student %} Ahmed{% end %}
```

Template -Include

templates/footer.html :

```
<p>You have reached the bottom of this page, thank you for your time.</p>
```

templates/template.html :

```
<h1>Hello</h1>

<p>This page contains a footer in a template.</p>

{% include 'footer.html' %}
```

Lab

template inhertance