

CM3035 Advanced Web Development

Lesson 1

Created by Ben Gay

Module Requirements to Pass

Minimum 35% in each element of summative assessment

Overall weighted average of 40%

Assessment

Coursework 1 - 50%

Coursework 2 - 50%

Django

What is Django?

Django is a back-end server side web framework

Django makes it easier to build web pages using Python

Django emphasizes reusability of components - DRY
(Don't Repeat Yourself)

How does Django Work?

Django follows the MVT design pattern
(Model View Template).

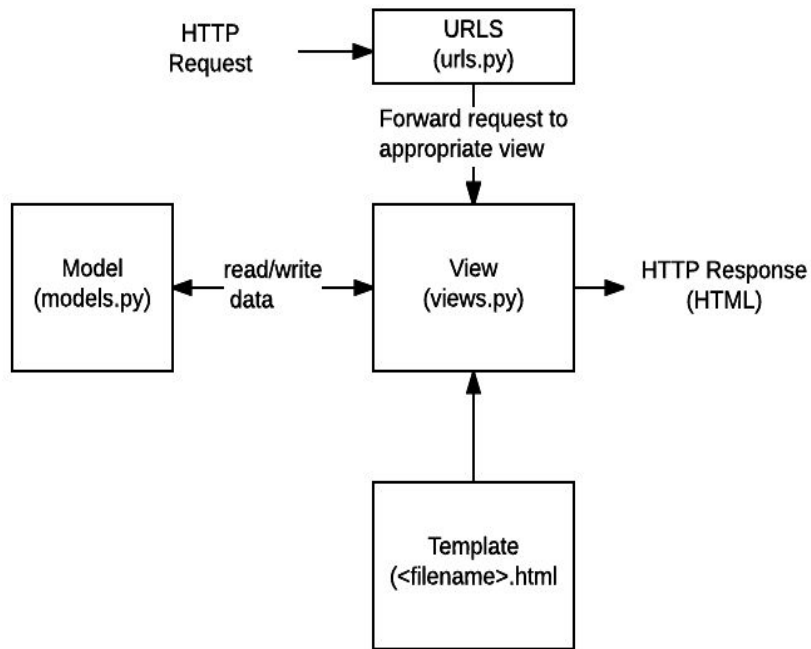
Model View Template(MVT)

Model - The data you want to present, usually data from a database.

View - A request handler that returns the relevant template and content - based on the request from the user.

Template - A text file (like an HTML file) containing the layout of the web page, with logic on how to display the data.

Django Framework?



Model

Model:

The model provides data from the database

Data is delivered as an Object Relational Mapping (ORM)

Database is SQL

The models are usually located in a file called models.py

Views

Views:

A view is a function or method that takes http requests as arguments, imports the relevant model(s), and finds out what data to send to the template, and returns the final result.

The views are usually located in a file called `views.py`

Template

Template:

A template is a file where you describe how the result should be represented.

Templates are often .html files, with HTML code.

Example:

```
<h1>My Homepage</h1>
```

```
<p>My name is {{ firstname }}.</p>
```

URLs

URLs:

Django also provide a way to navigate around the different pages in a website.

When a user requests a URL, Django decides which view it will send it to.

This is done in a file called `urls.py`.

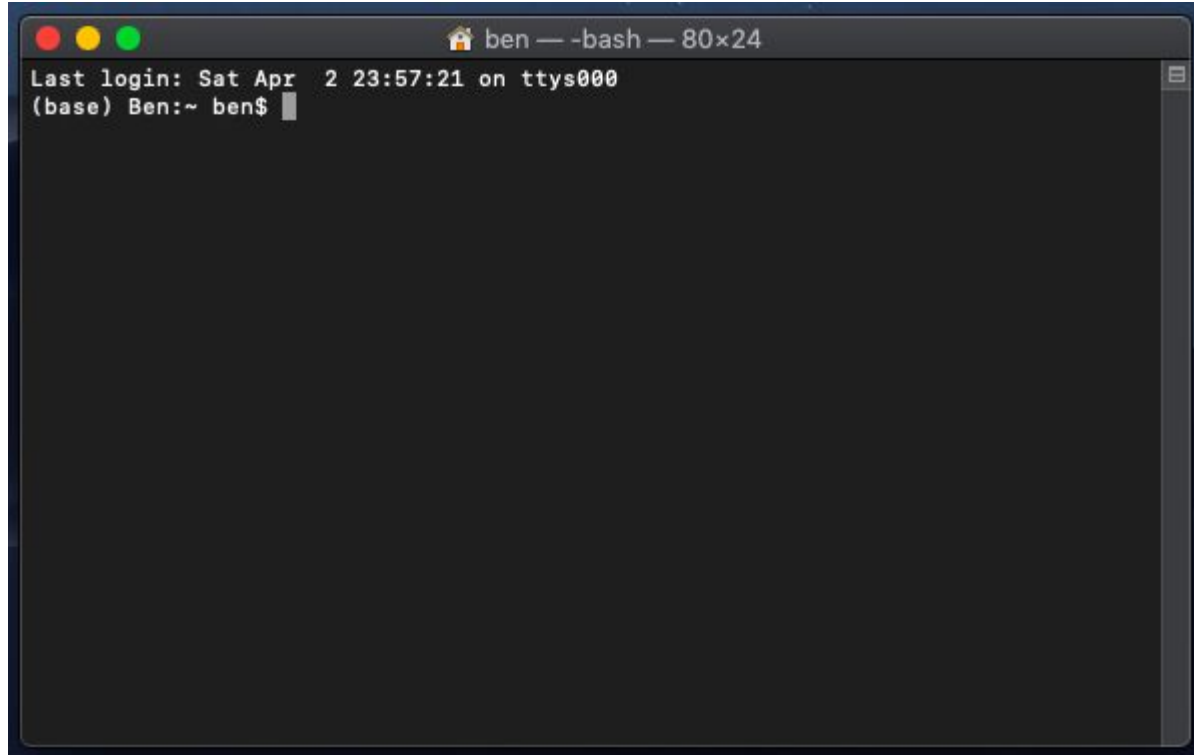
Process:

After you have created your first Django web application, and the browser requests the URL, this is basically what happens:

1. Django receives the URL, checks the `urls.py` file, and calls the view that matches the URL.
2. The view, located in `views.py`, checks for relevant models.
3. The models are imported from the `models.py` file.
4. The view then sends the data to a specified template in the template folder.
5. The template contains HTML and Django tags, and with the data it returns finished HTML content back to the browser.

Let's Begin Coding

Open terminal



Test your Global Environment

Let's test the application for python3 and pip3 in the Global Environment

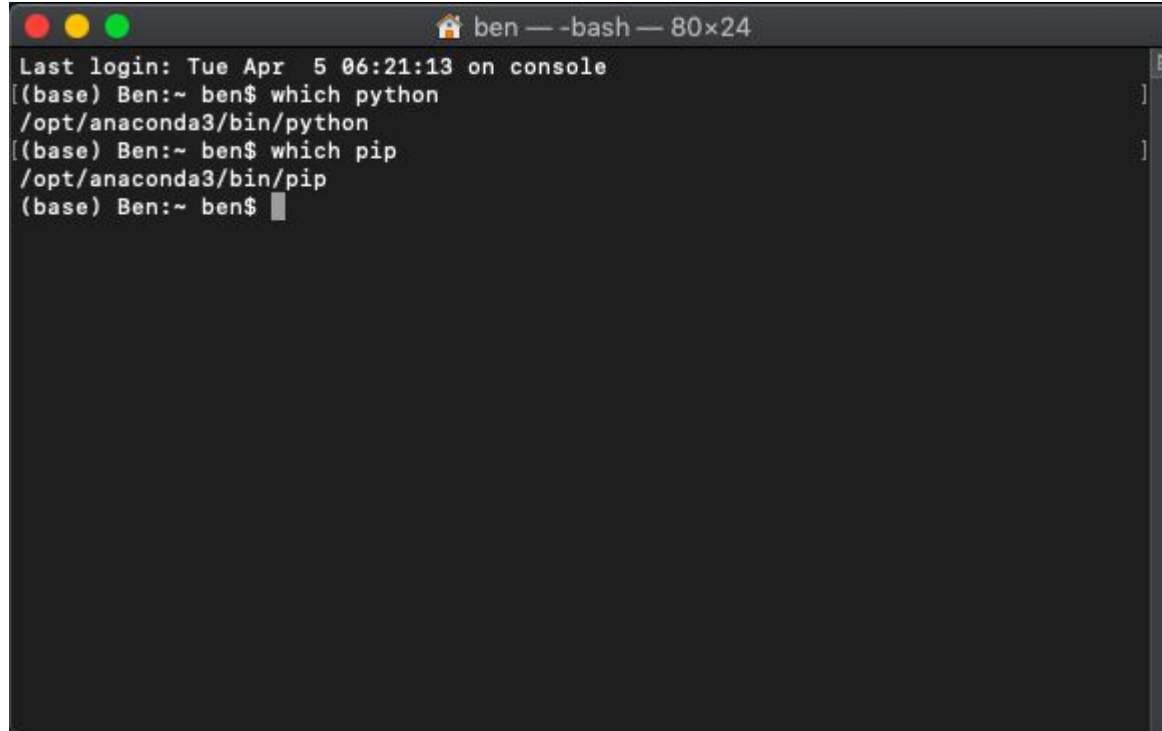
Check where python3 is installed `>>> which python3`

(Optional) `>>> python --version`

Check where pip3 is installed `>>> which pip3`

Use Virtual Environment instead because you want to achieve full control.

Test your Global Environment

A terminal window with a dark background and light gray text. The window title bar shows three colored circles (red, yellow, green) on the left, a house icon followed by 'ben' and '— -bash — 80x24' on the right. The terminal content shows a login message and two 'which' commands being executed in a conda environment.

```
ben — -bash — 80x24
Last login: Tue Apr  5 06:21:13 on console
[(base) Ben:~ ben$ which python
/opt/anaconda3/bin/python
[(base) Ben:~ ben$ which pip
/opt/anaconda3/bin/pip
(base) Ben:~ ben$
```


Create Virtual environment

Create a New folder on desktop >>>mkdir **djangoVenv**

(Optional)To Delete folder >>>rmdir

Go to folder djangoVenv >>>cd **djangoVenv**

Look at content inside **djangoVenv** folder >>> ls

(Optional) go back to parent folder >>>cd ..

(Optional) install tree for mac user >>> brew install tree

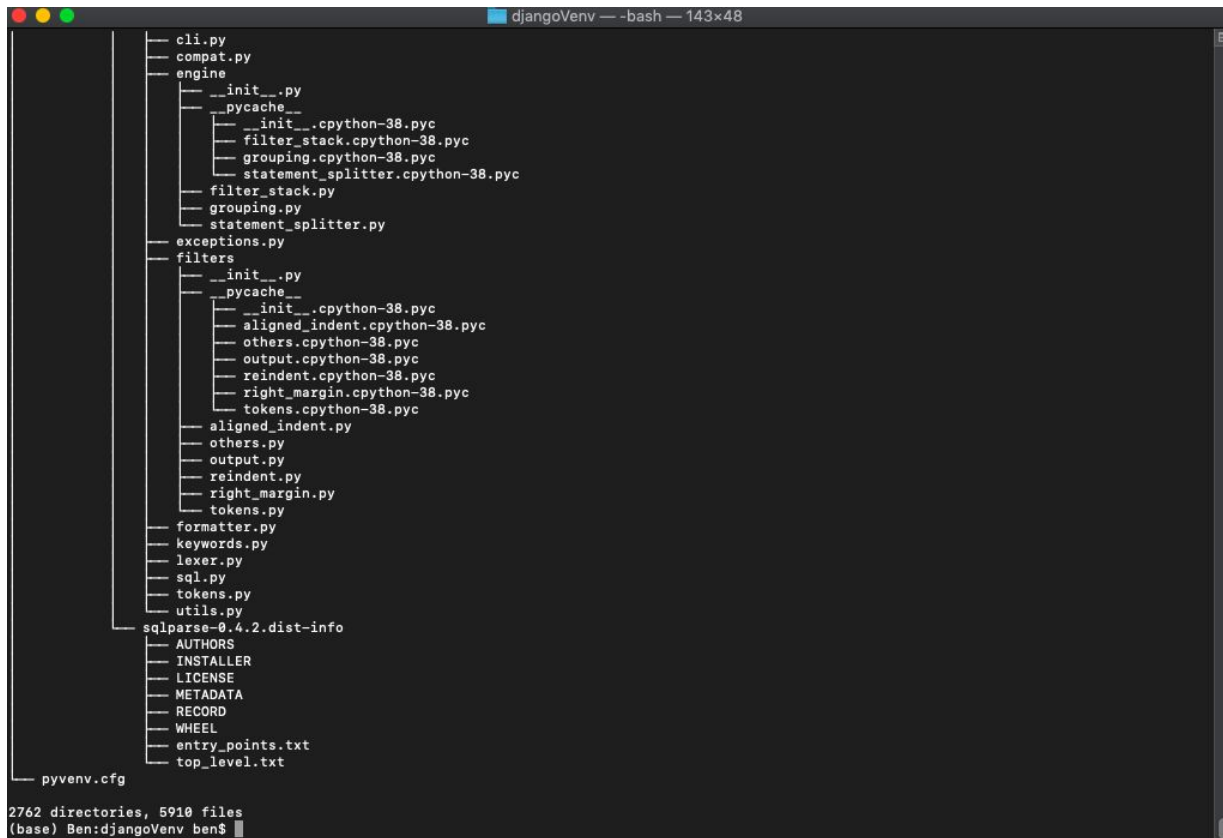
Create virtual environment >>>python3 -m venv **myvenv**

view content in myvenv folder >>> tree **myvenv**

Create Virtual environment

```
djangoVenv — -bash — 143x48
Last login: Tue Apr  5 06:54:45 on ttys000
(base) Ben:~ ben$ cd desktop/djangoVenv
(base) Ben:djangoVenv ben$ python3 -m venv myvenv
(base) Ben:djangoVenv ben$ tree myvenv
myvenv
├── bin
│   ├── Activate.ps1
│   ├── activate
│   ├── activate.csh
│   ├── activate.fish
│   ├── django-admin
│   ├── easy_install
│   ├── easy_install-3.8
│   ├── pip
│   ├── pip3
│   ├── pip3.8
│   ├── python -> python3
│   ├── python3 -> /opt/anaconda3/bin/python3
│   └── sqlformat
├── include
├── lib
│   └── python3.8
│       └── site-packages
│           ├── Django-4.0.3.dist-info
│           │   ├── AUTHORS
│           │   ├── INSTALLER
│           │   ├── LICENSE
│           │   ├── LICENSE.python
│           │   ├── METADATA
│           │   ├── RECORD
│           │   ├── REQUESTED
│           │   ├── WHEEL
│           │   ├── entry_points.txt
│           │   └── top_level.txt
│           └── __pycache__
```

Create Virtual environment



The screenshot shows a terminal window titled "djangoVenv -- bash -- 143x48". It displays a tree-like structure of files and directories for a virtual environment. The files listed include:

- cli.py
- compat.py
- engine
 - __init__.py
 - __pycache__
 - __init__.cpython-38.pyc
 - filter_stack.cpython-38.pyc
 - grouping.cpython-38.pyc
 - statement_splitter.cpython-38.pyc
 - filter_stack.py
 - grouping.py
 - statement_splitter.py
- exceptions.py
- filters
 - __init__.py
 - __pycache__
 - __init__.cpython-38.pyc
 - aligned_indent.cpython-38.pyc
 - others.cpython-38.pyc
 - output.cpython-38.pyc
 - reindent.cpython-38.pyc
 - right_margin.cpython-38.pyc
 - tokens.cpython-38.pyc
 - aligned_indent.py
 - others.py
 - output.py
 - reindent.py
 - right_margin.py
 - tokens.py
- formatter.py
- keywords.py
- lexer.py
- sql.py
- tokens.py
- utils.py
- sqlparse-0.4.2.dist-info
 - AUTHORS
 - INSTALLER
 - LICENSE
 - METADATA
 - RECORD
 - WHEEL
 - entry_points.txt
 - top_level.txt

At the bottom of the tree is the file "pyvenv.cfg". Below the file list, the terminal shows the command prompt and the output of the 'ls' command:

```
2762 directories, 5910 files
(base) Ben:djangoVenv ben$
```

Activate Virtual Environment

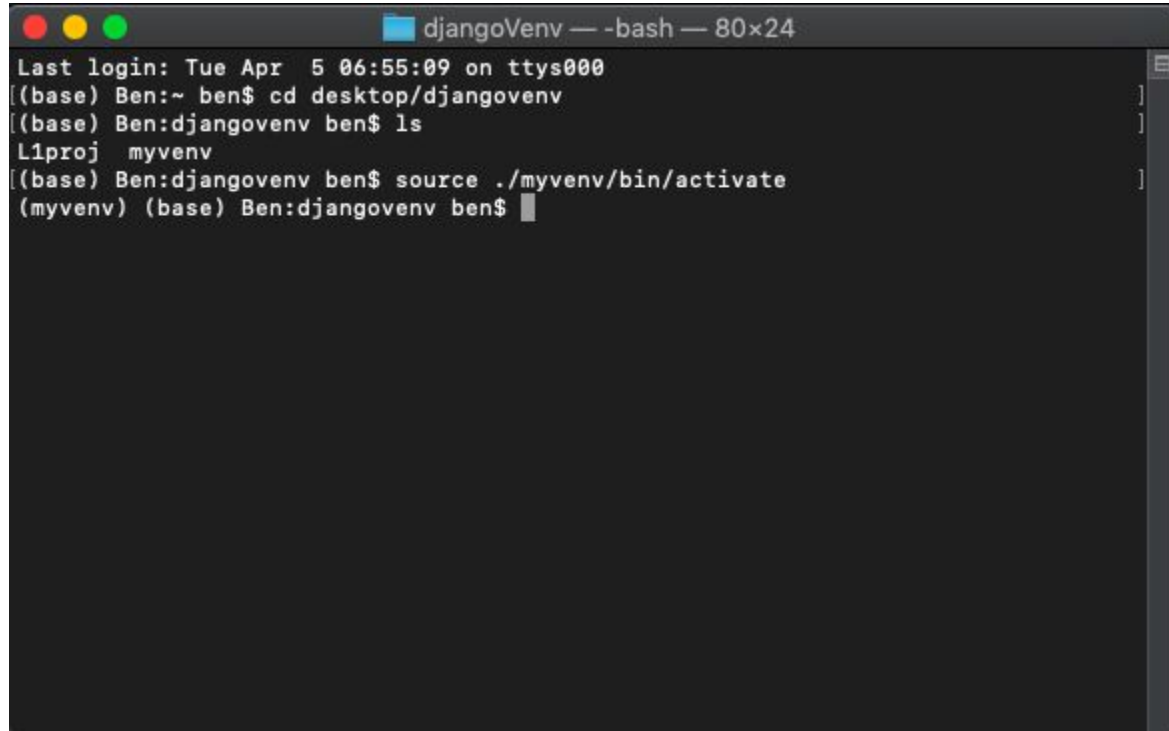
To **activate** virtual environment >>>source **./myvenv/bin/activate**

Notice (**myvenv**) appears, means virtual environment is running

Check pip running in virtual environment >>>which pip

if pip folder is found inside **myvenv** subfolder means virtual environment is set up successfully!!!

Activate Virtual Environment

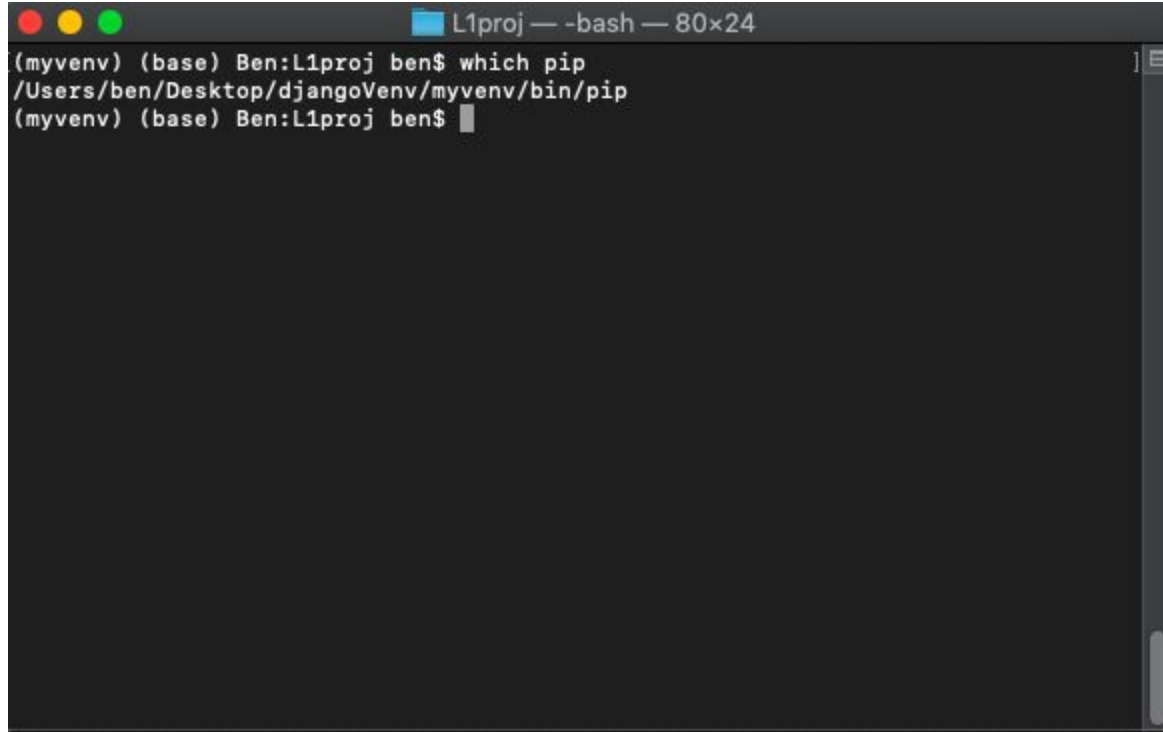


```
djangoVenv — -bash — 80x24
Last login: Tue Apr  5 06:55:09 on ttys000
(base) Ben:~ ben$ cd desktop/djangovenv
(base) Ben:djangovenv ben$ ls
L1proj  myvenv
(base) Ben:djangovenv ben$ source ./myvenv/bin/activate
(myvenv) (base) Ben:djangovenv ben$
```

The image shows a terminal window titled "djangoVenv — -bash — 80x24". The terminal output shows the following sequence of commands and their results:

- Initial prompt: `(base) Ben:~ ben$`
- Command: `cd desktop/djangovenv`
- Result: `(base) Ben:djangovenv ben$`
- Command: `ls`
- Result: `L1proj myvenv`
- Command: `source ./myvenv/bin/activate`
- Result: `(myvenv) (base) Ben:djangovenv ben$`

Test Activated Virtual Environment



```
L1proj — -bash — 80x24
(myvenv) (base) Ben:L1proj ben$ which pip
/Users/ben/Desktop/djangoVenv/myvenv/bin/pip
(myvenv) (base) Ben:L1proj ben$
```

A terminal window titled "L1proj — -bash — 80x24" with standard macOS window controls (red, yellow, green buttons). The prompt is "(myvenv) (base) Ben:L1proj ben\$". The user enters "which pip", and the terminal outputs the path "/Users/ben/Desktop/djangoVenv/myvenv/bin/pip". The prompt returns to "(myvenv) (base) Ben:L1proj ben\$".

What's in Virtual Environment

View applications installed in virtual Environment `>>>pip3 list`

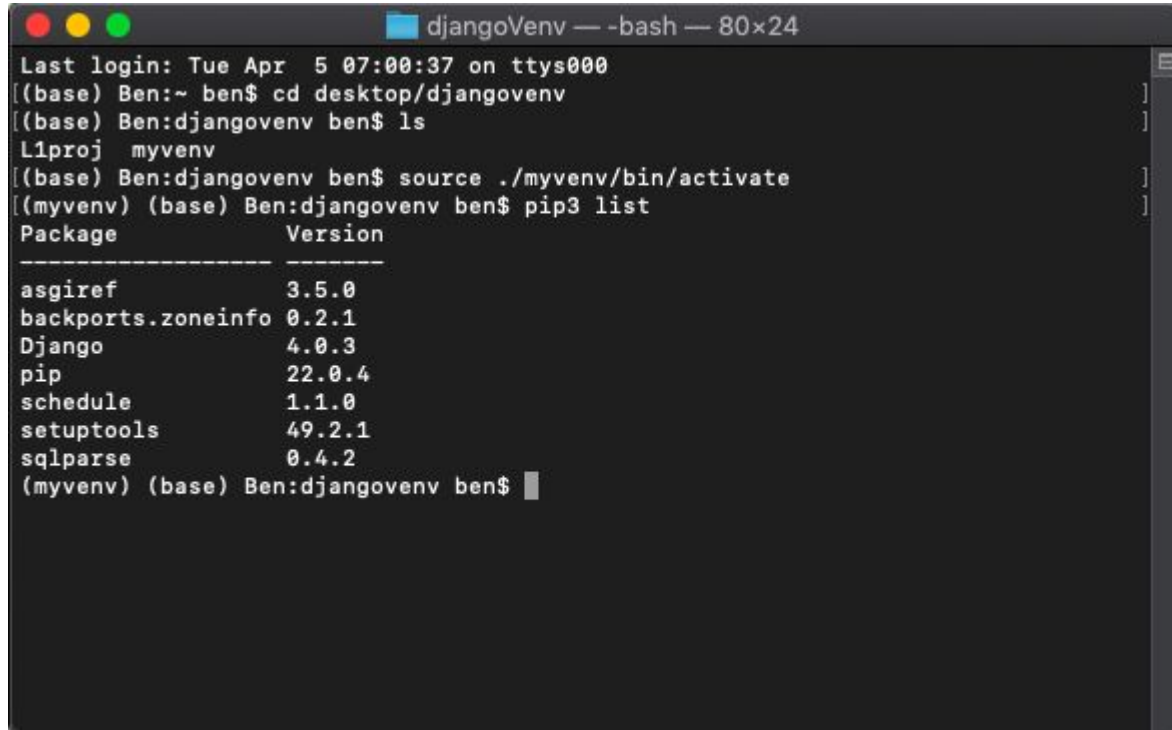
Try install a new application `>>> pip3 install schedule`

To View newly installed application `>>> pip3 list`

(Optional) upgrade pip3 `>>> python3 -m pip install --upgrade pip`

Install django `>>> pip3 install django` 'or' `python3 -m pip install
django`

What's in Virtual Environment



A terminal window titled "djangoVenv — -bash — 80x24" showing the steps to activate a virtual environment and list installed packages. The terminal output includes the last login time, directory navigation, file listing, activation command, and a list of installed packages with their versions.

```
Last login: Tue Apr 5 07:00:37 on ttys000
[(base) Ben:~ ben$ cd desktop/djangovenv
[(base) Ben:djangovenv ben$ ls
L1proj  myenv
[(base) Ben:djangovenv ben$ source ./myenv/bin/activate
[(myenv) (base) Ben:djangovenv ben$ pip3 list
```

Package	Version
asgiref	3.5.0
backports.zoneinfo	0.2.1
Django	4.0.3
pip	22.0.4
schedule	1.1.0
setuptools	49.2.1
sqlparse	0.4.2

```
(myenv) (base) Ben:djangovenv ben$
```


Deactivate Virtual Environment

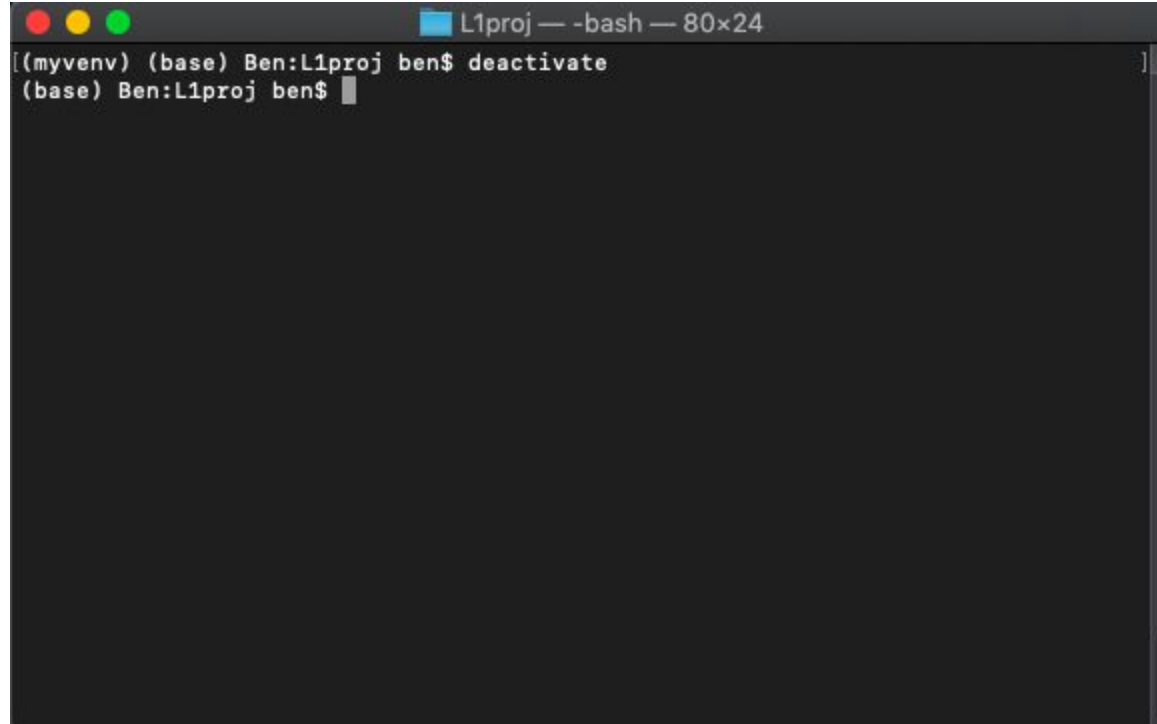
To **deactivate** virtual environment >>> deactivate
(myvenv) **disappears**

Let's Activate Virtual Environment Again!!!

Let'sActivate Virtual Environment Again!!!

Let'sActivate Virtual Environment Again!!!

Deactivate Virtual Environment



```
L1proj — -bash — 80x24
(myvenv) (base) Ben:L1proj ben$ deactivate
(base) Ben:L1proj ben$
```

A terminal window titled "L1proj — -bash — 80x24" with standard macOS window controls (red, yellow, green buttons). The terminal shows the command `deactivate` being executed, which switches the prompt from `(myvenv) (base)` to `(base)`. The cursor is positioned at the end of the second prompt line.

Create Django Project

Go to folder djangoVenv and **create** a new **django project**

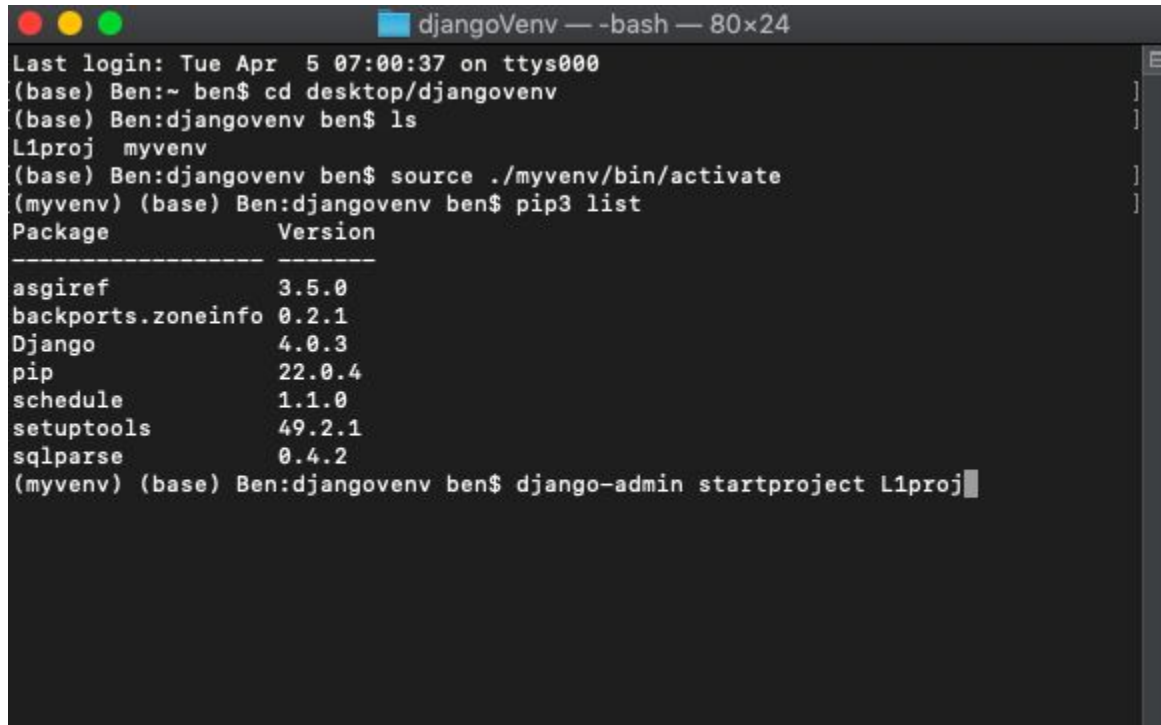
```
>>>django-admin startproject L1proj
```

Run the server >>> python3 manage.py runserver

To test if server is working, open browser to view **website** created
by django >>> <http://localhost:8000/>

To **stop** server --> CONTROL C

Create Django Project



```
djangoVenv — -bash — 80x24
Last login: Tue Apr  5 07:00:37 on ttys000
(base) Ben:~ ben$ cd desktop/djangovenv
(base) Ben:djangovenv ben$ ls
L1proj  myenv
(base) Ben:djangovenv ben$ source ./myenv/bin/activate
(myvenv) (base) Ben:djangovenv ben$ pip3 list
Package            Version
-----
asgiref             3.5.0
backports.zoneinfo  0.2.1
Django              4.0.3
pip                 22.0.4
schedule            1.1.0
setuptools           49.2.1
sqlparse            0.4.2
(myvenv) (base) Ben:djangovenv ben$ django-admin startproject L1proj
```

Create Django Project

```
L1proj — python3 • python3 manage.py runserver — 80x24
(myvenv) (base) Ben:desktop ben$ cd.djangovenv
(myvenv) (base) Ben:djangovenv ben$ ls
L1proj myvenv
(myvenv) (base) Ben:djangovenv ben$ cd L1proj
(myvenv) (base) Ben:L1proj ben$ ls
L1app      L1proj      db.sqlite3  manage.py
(myvenv) (base) Ben:L1proj ben$ python3 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.
April 04, 2022 - 23:11:23
Django version 4.0.3, using settings 'L1proj.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CONTROL-C.
```

Create Django Project

django

[View release notes](#) for Django 2.1



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.



Django Documentation
Topics, references, & how-to's



Tutorial: A Polling App
Get started with Django



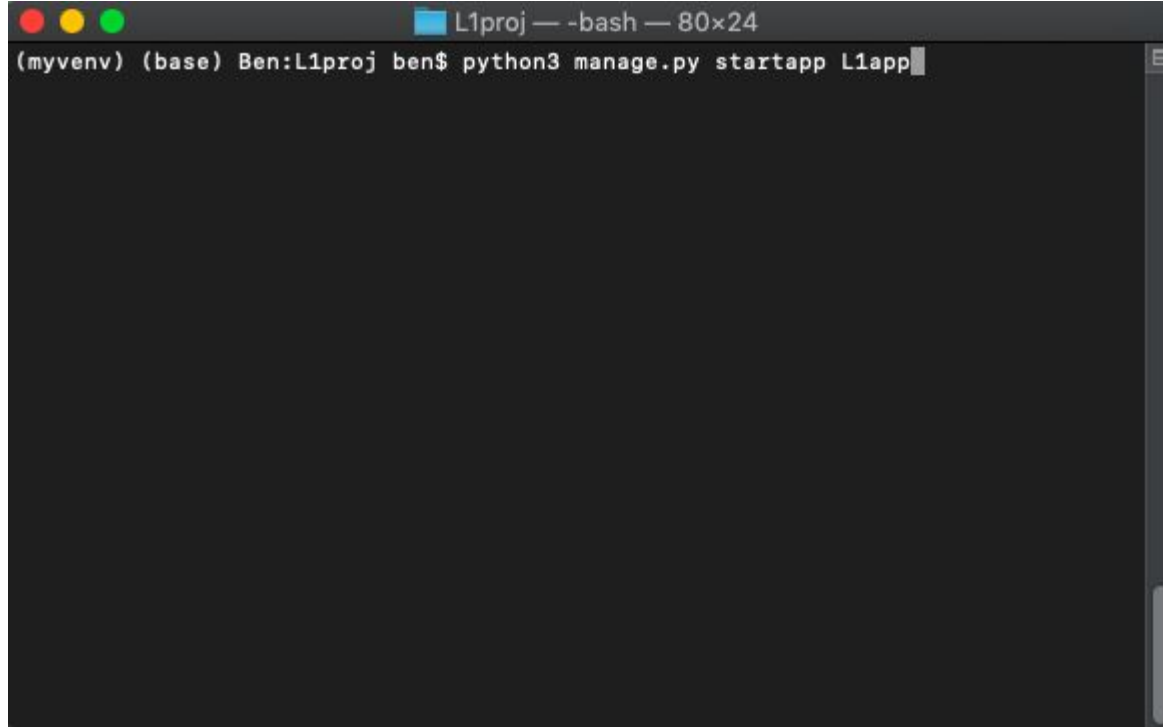
Django Community
Connect, get help, or contribute

Create First Django App

Location : Root folder > virtual environment folder > Django
project folder > application

To create new application >>> python3 manage.py startapp **L1app**

Create First Django App

A terminal window titled "L1proj — -bash — 80x24" with standard macOS window controls (red, yellow, green buttons). The terminal shows the command `(myvenv) (base) Ben:L1proj ben$ python3 manage.py startapp L1app` entered at the prompt. The rest of the terminal is empty.

```
(myvenv) (base) Ben:L1proj ben$ python3 manage.py startapp L1app
```

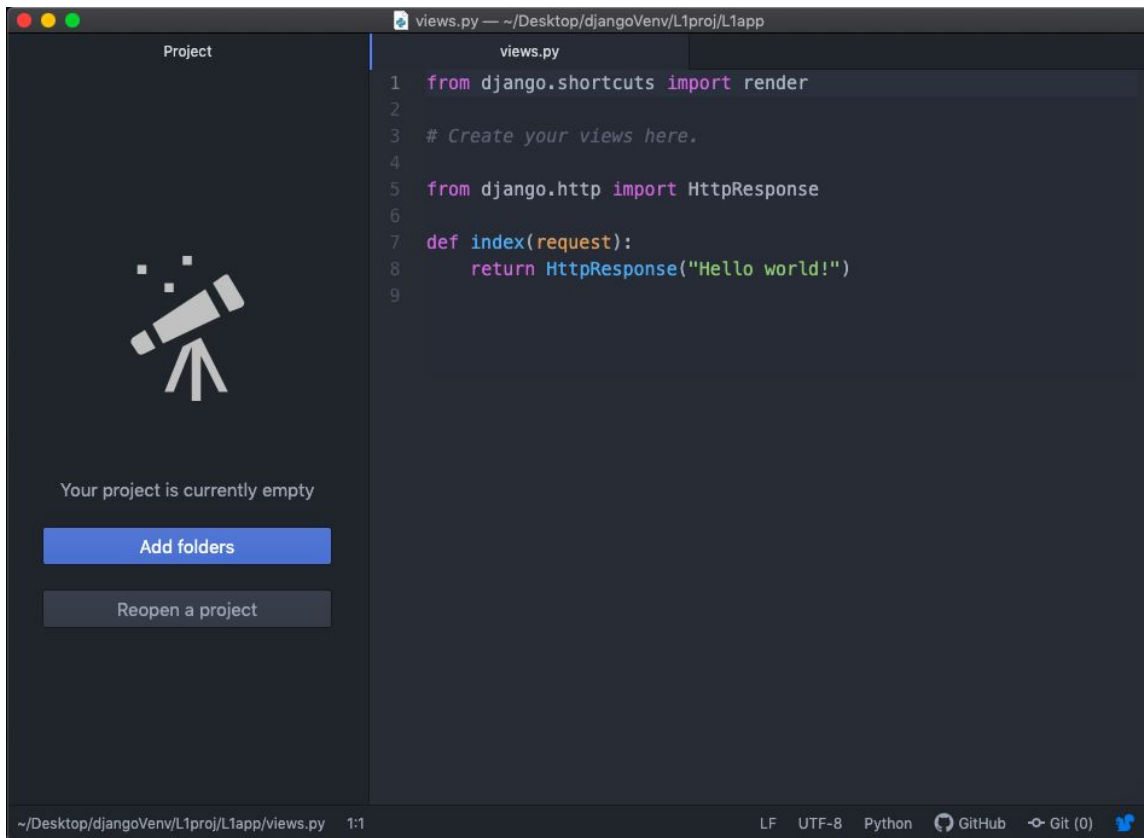

Display Hello world on browser

1. Respond Hello World to browser

Location : **L1app/views.py**

```
from django.shortcuts import render  
  
from django.http import HttpResponse  
  
def index(request):  
    return HttpResponse("Hello world!")
```

Display Hello world on browser



Establish URL for index function

2. Create a file named **urls.py**

Location : **L1app/urls.py**

```
from django.urls import path
```

```
from . import views
```

```
urlpatterns = [  
    path("", views.index, name='index'),  
]
```

Establish URL in project folder

3. Set url in project folder to include L1app urls

Location : **L1proj/urls.py**

```
from django.contrib import admin
```

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

```
urlpatterns = [  
    path('L1app/', include('L1app.urls')),  
    path('admin/', admin.site.urls),  
]
```

View Hello World on Browser

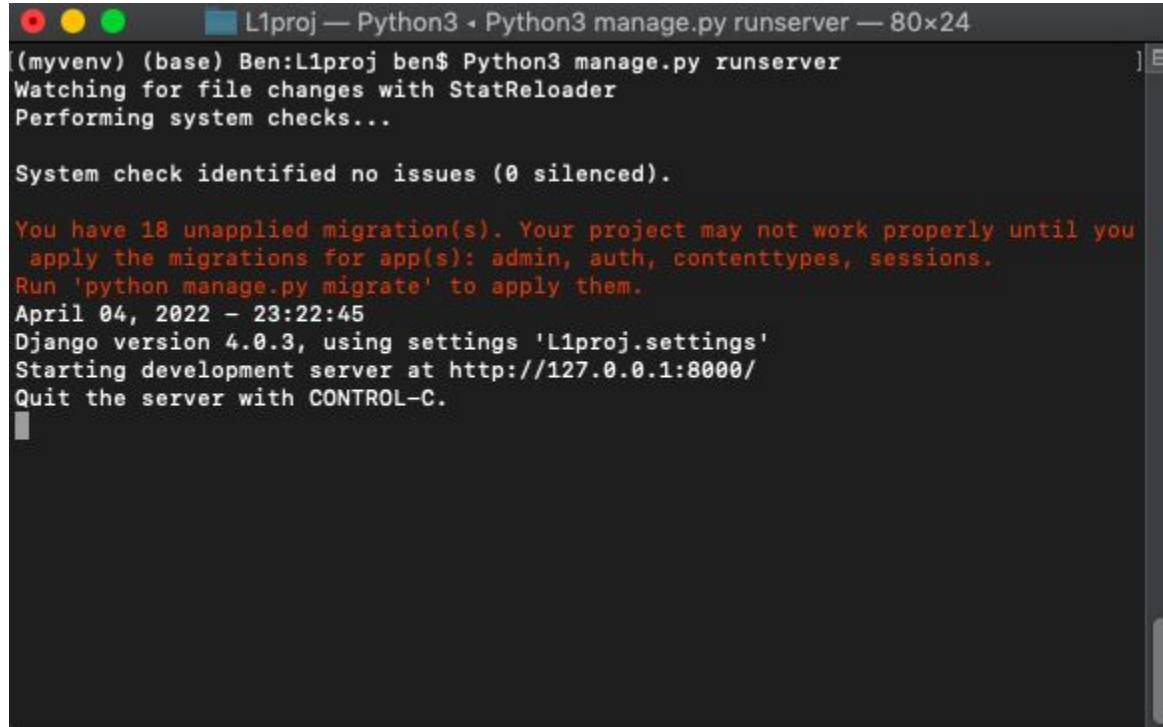
4. Python3 manage.py runserver

Open browser

`http://localhost:8000/L1app`

You should see **Hello World!** First Django App **successful!**

View Hello World on Browser

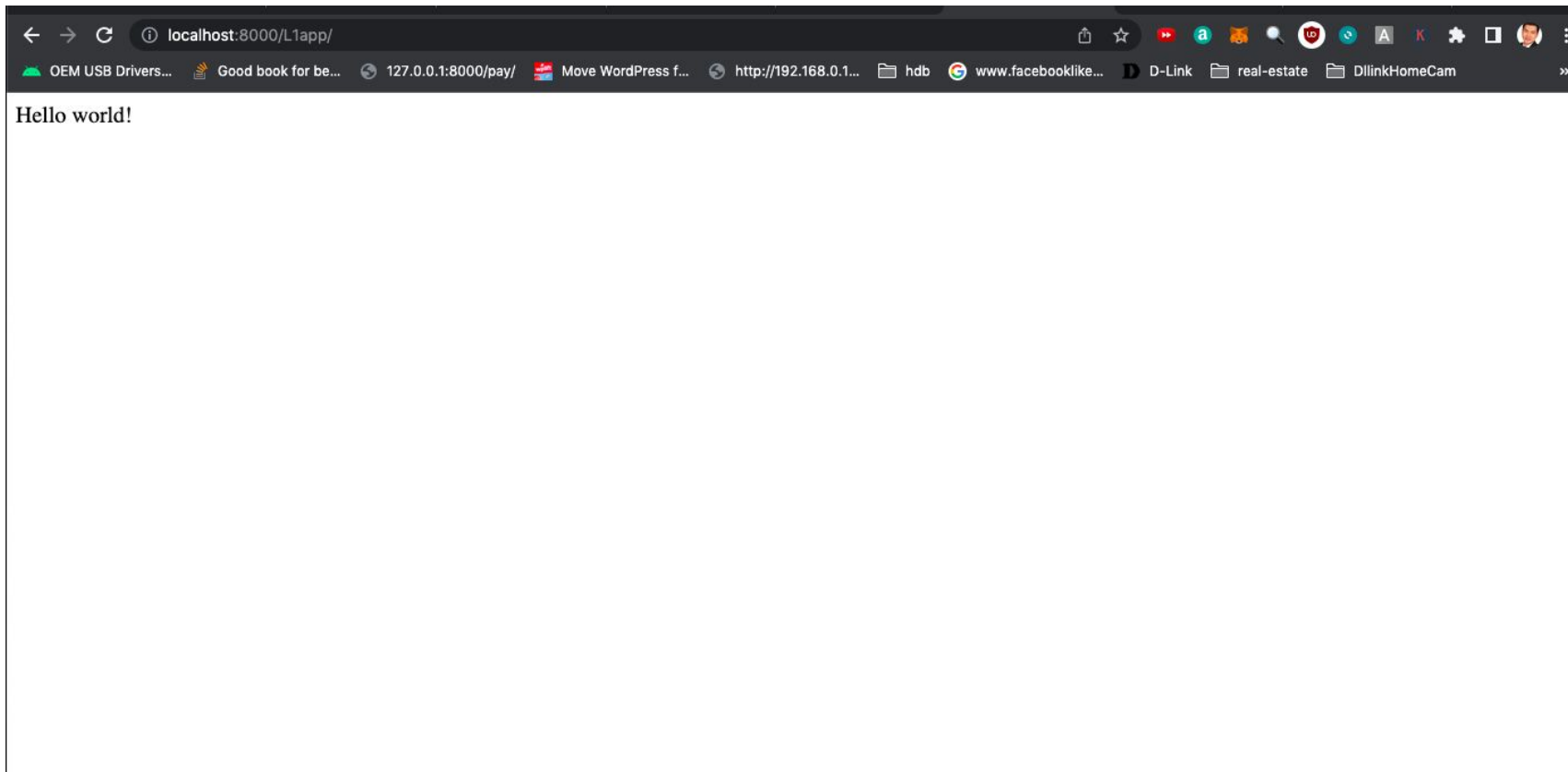
A terminal window titled 'L1proj — Python3 · Python3 manage.py runserver — 80x24'. The window shows the output of running 'python manage.py runserver'. It includes system checks, a warning about 18 unapplied migrations, and the server starting at http://127.0.0.1:8000/.

```
(myvenv) (base) Ben:L1proj ben$ Python3 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.
April 04, 2022 - 23:22:45
Django version 4.0.3, using settings 'L1proj.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CONTROL-C.
```

View Hello World on Browser



End of Lesson 1