Module – 1

Learning Objectives

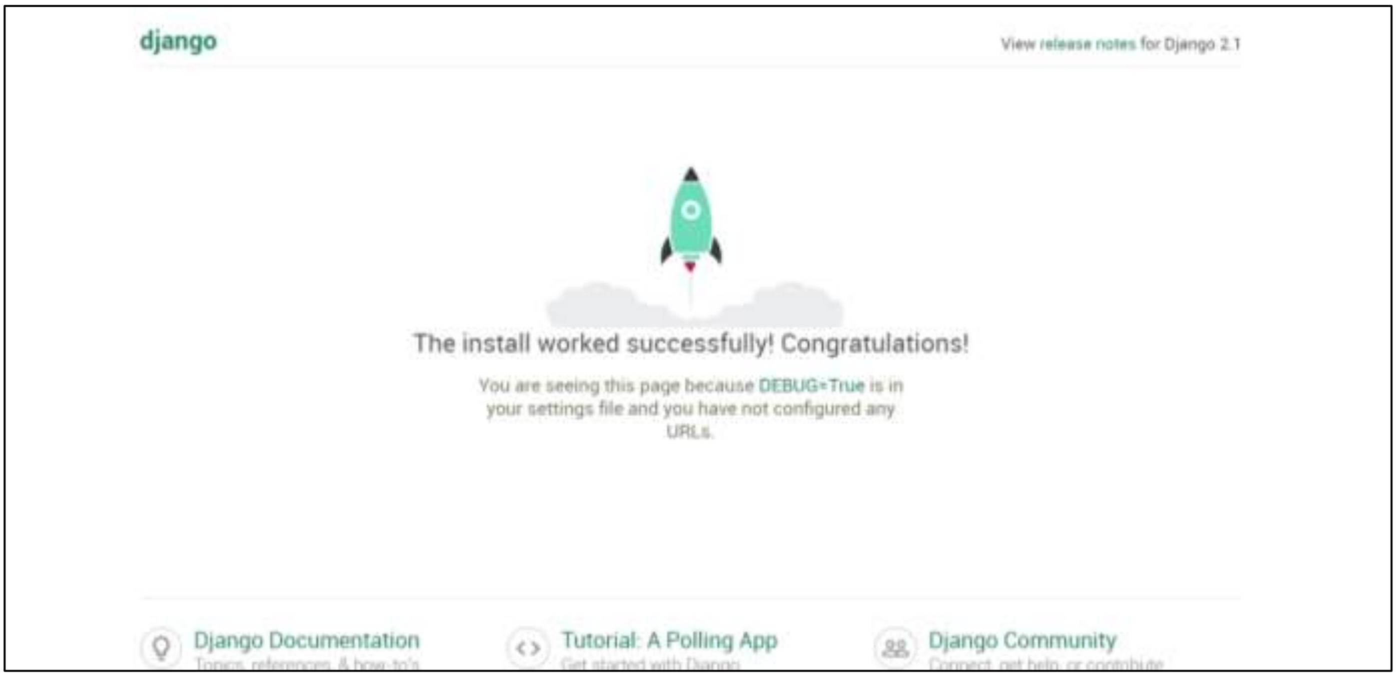
- Installing Python and Django.
- Setting up new project.
- Understanding setting.py file.
- Change default database settings.
- Understanding manage.py file (migrate and runserver commands).

Django – Installation

Steps for Django Installation

```
# Open command Prompt
D:\Django>python -V
Python 3.7.0

D:\Django>python -m pip install --upgrade pip

D:\Django>python -m pip install django

D:\Django>python -m django --version
2.1.1

D:\Django>django-admin.exe startproject MyBlog

D:\Django>cd MyBlog

# This command installs database
D:\Django\MyBlog>python manage.py migrate

D:\Django\MyBlog>python manage.py runserver
Performing system checks...

System check identified no issues (0 silenced).
September 03, 2018 - 10:35:00
Django version 2.1.1, using settings 'MyBlog.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.
```

Open web browser and paste http://127.0.0.1:8000/ in address bar to view running application.


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

Django Installation guide

Creating a project

```
D:\Django>django-admin.exe startproject MyBlog

D:\Django\MyBlog>dir

03-09-2018  10:34    <DIR>          .
03-09-2018  10:34    <DIR>          ..
03-09-2018  10:34           131,072 db.sqlite3
03-09-2018  10:34               553 manage.py
03-09-2018  10:34    <DIR>          MyBlog

D:\Django\MyBlog\MyBlog>dir

03-09-2018  10:34    <DIR>          .
03-09-2018  10:34    <DIR>          ..
03-09-2018  10:34             3,208 settings.py
03-09-2018  10:34               769 urls.py
03-09-2018  10:34               405 wsgi.py
03-09-2018  10:34                 0 __init__.py
03-09-2018  10:35    <DIR>          __pycache__

D:\Django\MyBlog\MyBlog>
```

Changing settings (settings.py)

MyBlog/settings.py

```
# Default Time zone and Language is mentioned as below.

LANGUAGE_CODE = 'en-us'
TIME_ZONE = ‘UTC’

# This can be modified to desired value.

TIME_ZONE = 'Asia/Kolkata'
```

Set up a database (optional)

Django provides built-in support for several types of database backends. With just a few lines in our settings.py file it can support PostgreSQL, MySQL, Oracle, or SQLite. But the simplest–by far–to use is SQLite because it runs off a single file and requires no complex installation. By contrast, the other options require a process to be running in the background and can be quite complex to properly configure. Django uses SQLite by default for this reason and it’s a perfect choice for small projects.

MyBlog/settings.py

```
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.mysql',
        'NAME': 'Django',
        'USER': 'root',
        'PASSWORD': '12345',
        'HOST': 'localhost',
        'PORT': '3306'
# Default database settings

        #'ENGINE': 'django.db.backends.sqlite3',
        #'NAME': os.path.join(BASE_DIR, 'db.sqlite3'),
    }
}
```

Run ***manage.py migrate*** to update the database. This command is executed for the initial installation of database. And then every time models are updated. Technically a db.sqlite3 file is created the first time you run either migrate or runserver. Using runserver configures a database using Django's default settings, however migrate will sync the database with the current state of any database models contained in the project and listed in INSTALLED_APPS. In other words, to make sure the database reflects the current state of your project you'll need to run migrate (and also makemigrations) each time you update a model.