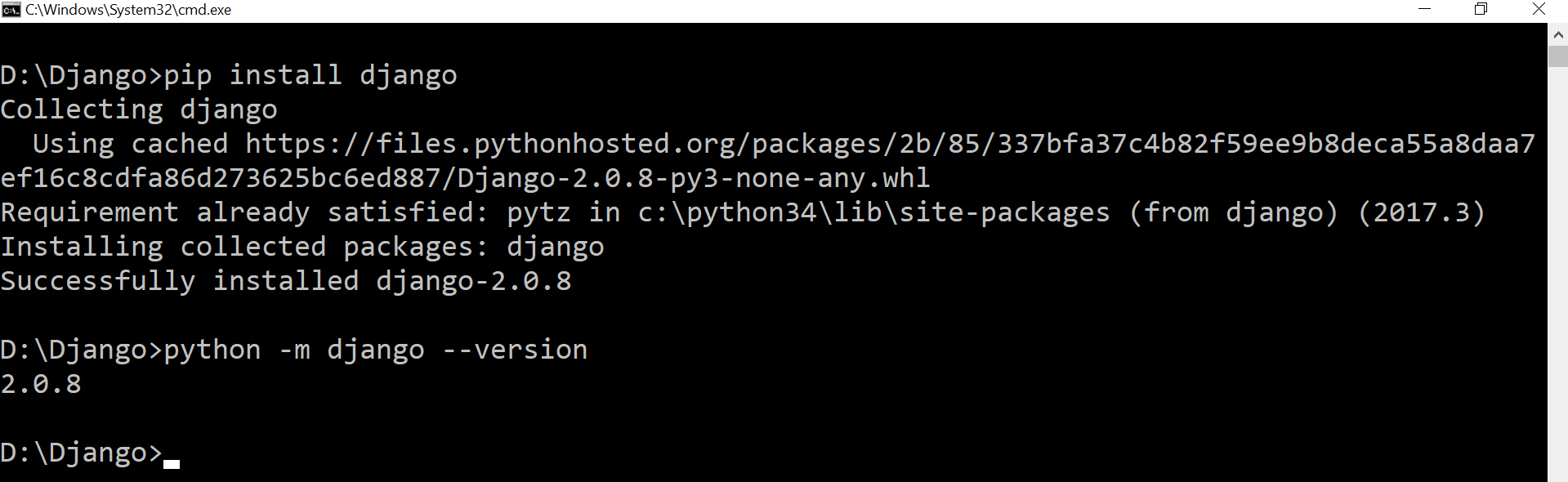
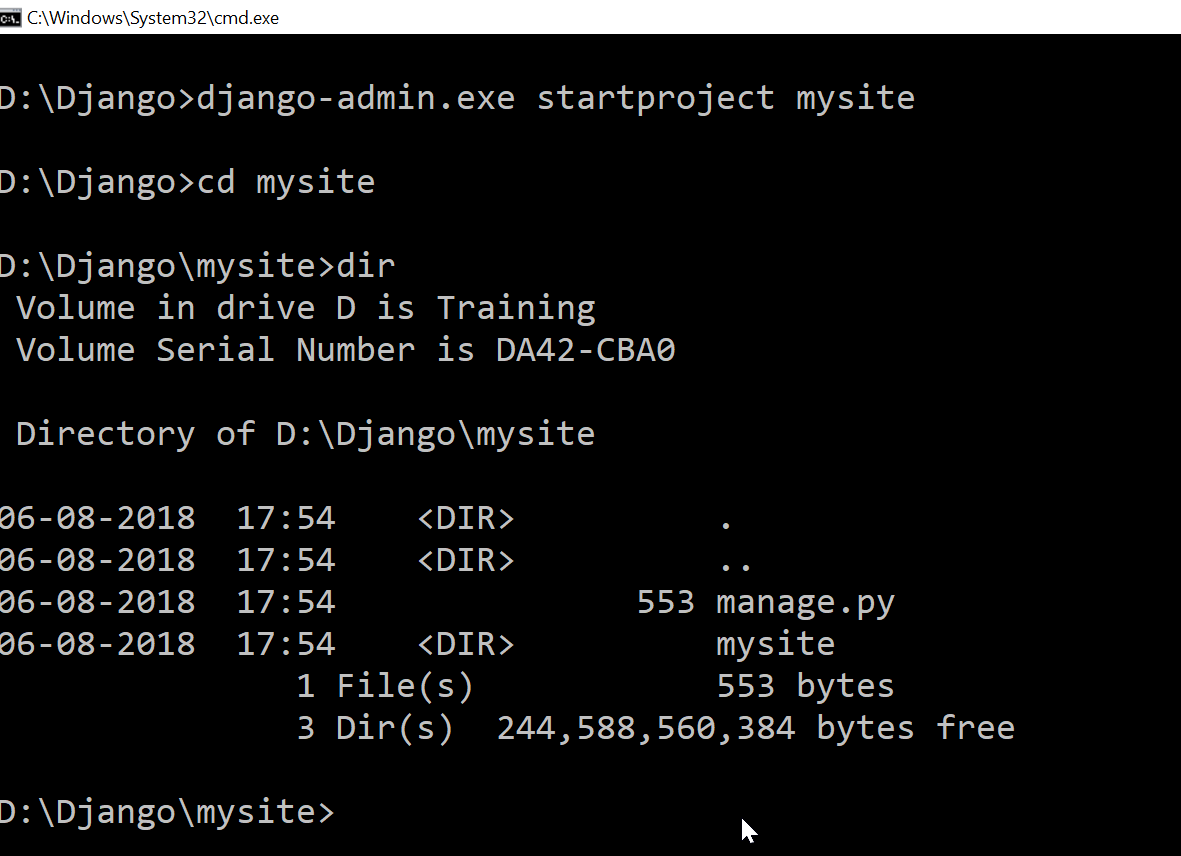
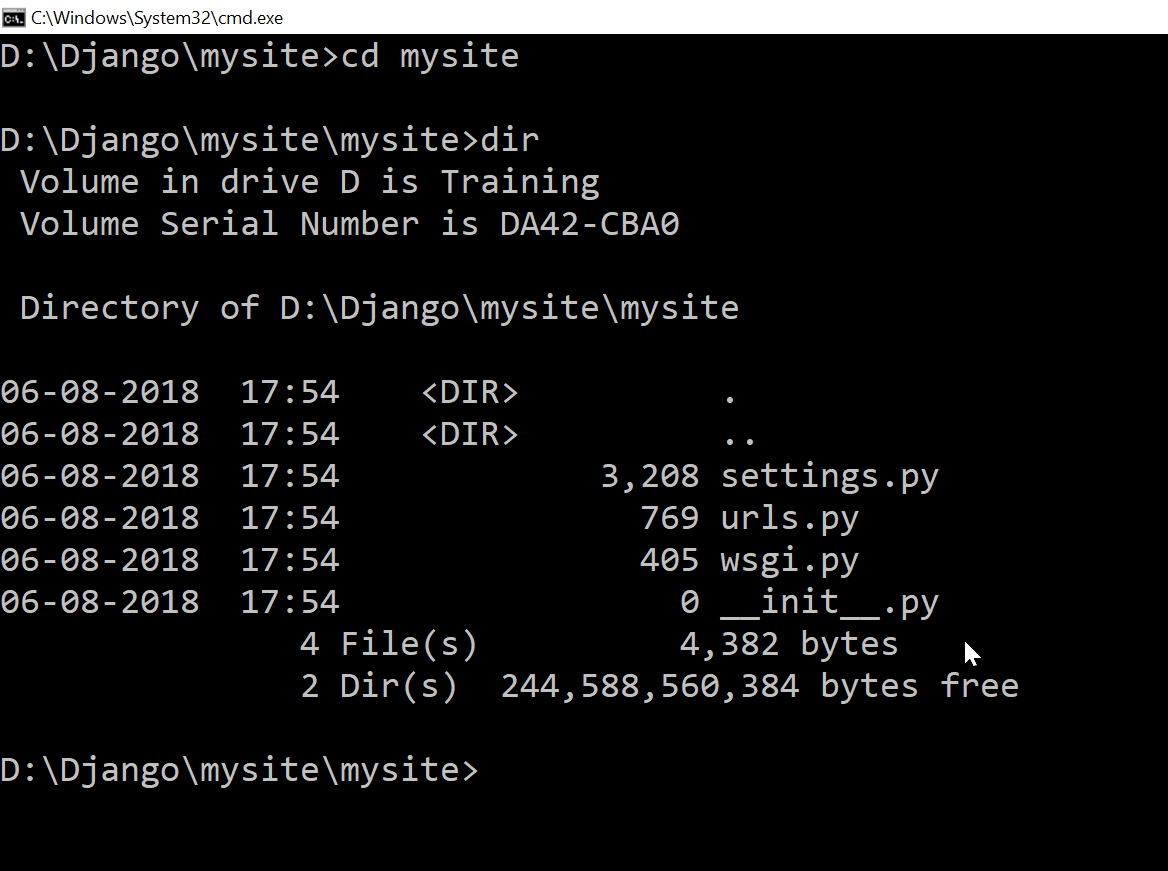
**Django Installation guide**

# **Step 1. Installing Django**



# **Step 2. Creating a project**

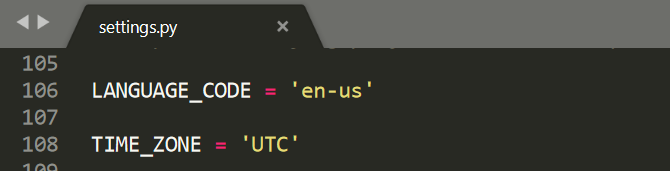
 

This will create a **mysite** directory in current directory. The files automatically created inside the directory are:

* The outer **mysite/** root directory is just a container for your project. Its name doesn’t matter to Django; you can rename it to anything you like.
* **manage.py**: A command-line utility that lets you interact with this Django project in various ways. **django-admin** is Django’s command-line utility for administrative tasks. In addition, **manage.py** is automatically created in each Django project. **manage.py** does the same thing as **django-admin** but takes care of a few things for you:
* It puts your project’s package on **sys.path**.
* It sets the [**DJANGO\_SETTINGS\_MODULE**](https://docs.djangoproject.com/en/2.0/topics/settings/#envvar-DJANGO_SETTINGS_MODULE) environment variable so that it points to your project’s **settings.py** file.
* The inner **mysite/** directory is the actual Python package for your project.
* **mysite/\_\_init\_\_.py**: An empty file that tells Python that this directory should be considered a Python package.
* **mysite/settings.py**: A Django settings file contains all the configuration of your Django installation. A settings file is just a Python module with module-level variables.
* **mysite/urls.py**: The URL declarations for this Django project; a “table of contents” of your Django-powered site.
* **mysite/wsgi.py**: An entry-point for WSGI-compatible web servers to serve your project.

# **Step 3. Changing settings (settings.py)**

Default Time zone and Language is mentioned as below. This can be modified to desired value.

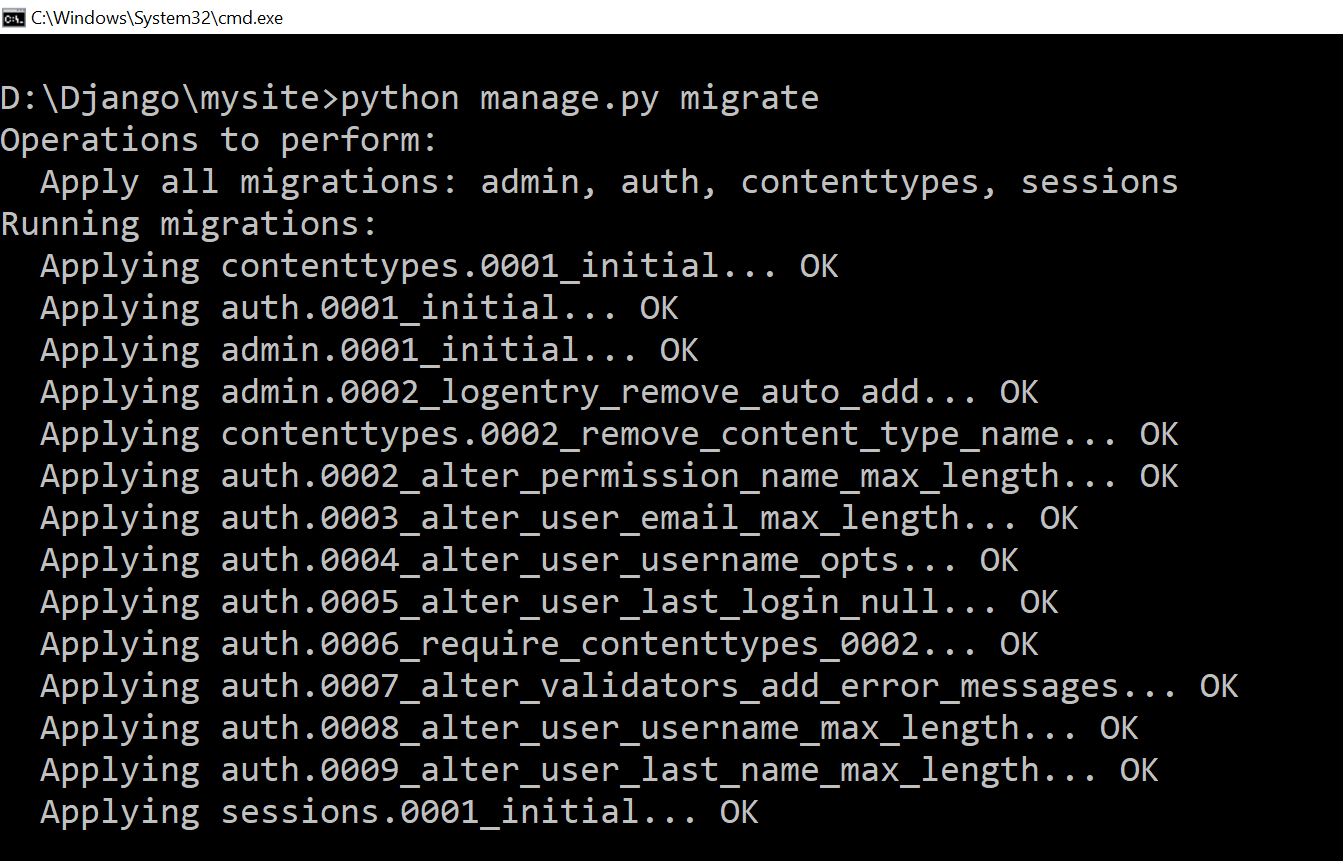


# **Step 4. Set up a database (optional)**

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.

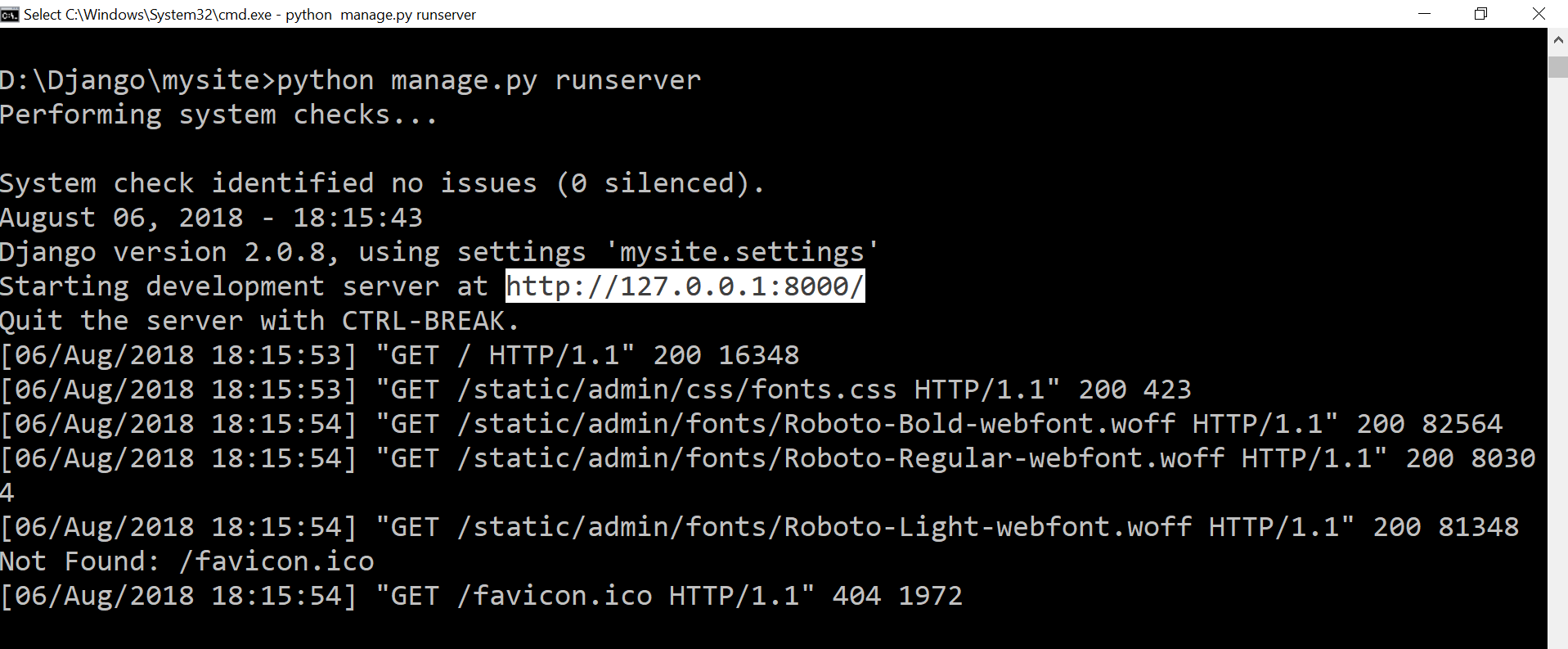
## 

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.

# **Step 5. Starting the server**



Copy paste the highlighted address in the browser and you can see your django project up and running!!!!