**Installing and working with GeoDjango**

**Step 1: Download and Install Python**

Visit the Python website and download the Python version suitable for your operating system. For example, if you're using Windows, download the executable file for Windows 64-bit from here.

**Step 2: Set up Environment Variables**

Navigate to Environment Variables > System Variables > Path and add the path to your Python installation.

Step 3: Create a Virtual Environment

If you are working with a virtual environment, you do not need to set up environment variables separately. Open a command prompt and navigate to the directory where you want to create a virtual environment.

Run the command: ***python -m venv <env\_name>***

**Step 4: Activate the Virtual Environment**

Activate the virtual environment by running the command: ***<env\_name>\Scripts\activate***.

**Step 5: Install Django**

Install Django using pip by running: ***pip install django.***

Check the installed Django version by running: ***python -m django --version***.

**Step 6: Install GeoDjango Dependencies**

Download the GDAL wheel file from here (https://www.lfd.uci.edu/~gohlke/pythonlibs/#gdal), ensuring it matches your Python version.

Place the GDAL file in your directory and install it using pip: ***python -m pip install <gdal\_file\_name>***

Verify GeoDjango installation by opening Python and ***import django.contrib.gis***

**Step 7: Configure GDAL Paths**

Update Django settings by adding GDAL path. If you're on Windows, you may need to set the environment variables as described here.

Go to Django settings and add GDAL path[[1]](#footnote-1)

if os.name == 'nt':

VENV\_BASE = os.environ['VIRTUAL\_ENV']

os.environ['PATH'] = os.path.join(VENV\_BASE, 'Lib\\site-packages\\osgeo') + ';' + os.environ['PATH']

os.environ['PROJ\_LIB'] = os.path.join(VENV\_BASE, 'Lib\\site-packages\\osgeo\\data\\proj') + ';' + os.environ['PATH']

**Step 8: Create a Django Project**

Create a new Django project using the command: ***django-admin startproject my\_project\_name***

**Step 9: Create a Django App**

Create a Django app within the project by running: ***python manage.py startapp my\_app\_name***

**Step 10: Install PostgreSQL/PostGIS**

Install and set up PostgreSQL/PostGIS as per your requirements.

**Step 11: Configure Database Settings**

In the project's settings.py, modify the DATABASES dictionary to use the PostGIS database backend as

***DATABASES = {***

***'default': {***

***'ENGINE': 'django.contrib.gis.db.backends.postgis',***

***'NAME': 'your\_database\_name',***

***'USER': 'your\_database\_user',***

***'PASSWORD': 'your\_database\_password',***

***'HOST': 'localhost',***

***'PORT': '5432',***

***}***

***}***

**Step 12: Create Django components**

Create basic components of django model, view, URLs, template, static files etc.

For django: <https://docs.djangoproject.com/en/5.0/intro/tutorial01/>

For Geodjango: <https://docs.djangoproject.com/en/5.0/ref/contrib/gis/tutorial/>

Use Generative AI (ChatGPT, Copilot, Gemini, etc.) for learning and don’t forget to use proper prompting rules to have answer you wish to.

If you want o use Google Earth Engine, follow the link to setup

<https://developers.google.com/earth-engine/guides/python_install>

1. reference from: https://stackoverflow.com/questions/49139044/geodjango-on-windows-could-not-find-the-gdal-library-oserror-winerror-12 [↑](#footnote-ref-1)