This guide aims to install GeoKey on a cloud server running on a **Ubuntu** machine, specifically version 18.04 that will be supported until 2028.

It explains how to

- install pre-requisites & Python packages,
- setup the database,
- set up virtual environment,
- set up GeoKey and
- configure the system files and make necessary modifications to make GeoKey run smoothly.

This documentation aims to simplify the processes as much as possible to make it suitable for everyone including non-technical people not familiar with Python / Linux environment but keen on running GeoKey on their own cloud servers.

A. Installing GeoKey on Ubuntu

1. Installing Pre-Requisites

1.0 Connect remote server through ssh

```
ssh root@XXX.XX.XXX.XX (IP adress)
```

1.1 Update your system:

```
sudo apt-get update && sudo apt-get upgrade
```

1.2 Install Postgres and PostGIS

```
wget --quiet -O - https://www.postgresql.org/media/keys/ACCC4CF8.asc | sudo apt-key add -

echo "deb http://apt.postgresql.org/pub/repos/apt/ `lsb_release -cs`-pgdg main" |sudo tee  /etc/apt/sources.list.d/pgdg.list

sudo apt update

sudo apt -y install postgresql-12 postgresql-client-12

sudo apt install postgis postgresql-12-postgis-2.5
```

1.3 Setup all other dependencies

```
sudo apt-get install python-pip python-virtualenv python-dev libjpeg-dev libmagickcore-dev libmagickwand-dev imagemagick binutils libproj-dev gdal-bin python-gdal

sudo apt-get install libgdal-dev

sudo apt-get install ffmpeg
```

```
wget http://launchpadlibrarian.net/348889634/libav-tools_3.4.1-1_all.deb

sudo dpkg -i libav-tools_3.4.1-1_all.deb
```

1.4 Change ImageMagick permissions for it to be able to write:

```
sed -i 's/\(<policy domain="coder" rights=\)"none" \(pattern="PDF" \/>\)/\
1"read|write"\2/g' /etc/ImageMagick-6/policy.xml
```

2. Setting Up The Database

2.1 Connect to PostgreSQL

```
su – postgres
psql
```

2.2 Now that you're connected to PostgreSQL, you can add the user and create a new database, and to make it simple, we add ownership of that database to the user

```
CREATE USER django WITH PASSWORD 'somepass';
```

2.3 Create the database and close the connection

```
CREATE DATABASE geokey OWNER django;
\q
```

2.4 Install the PostGIS extension on your database

```
psql -d geokey -c 'create extension postgis;'
psql -d geokey -c 'create extension hstore;'
```

2.5 Logout

```
logout
```

3. Installing Apache

3.1 Installing Apache and mod-wsgi

```
sudo apt-get install apache2 libapache2-mod-wsgi
```

4.Setting Up GeoKey

4.1 Clone an existing Git repository ready for work. Assuming you are installing a production environment on a cloud server:

```
cd /var/www
git clone https://github.com/ExCiteS/geokey.git
```

4.2 Change the owner and group of files, directories.

```
chown -R www-data:www-data geokey
```

4.3 Setting Up Virtual Environment

We will install GeoKey in a virtual enviroment. Choose a directory where you have write access and create an empty virtual environment

```
cd geokey
virtualenv env
```

4.4. Activate the env

```
source env/bin/activate
```

4.5 Upgrade pip so the correct python dependencies will be installed later on

```
pip install --upgrade pip
```

4.6 GDAL is likely to fail specifying the include directories when instaling it via pip on Debian, so follow the procedure:

Install Python binding for GDAL

```
export CPLUS_INCLUDE_PATH=/usr/include/gdal
export C_INCLUDE_PATH=/usr/include/gdal
```

4.7 To be sure that you install the correct version and avoid any problem, retrieve the version

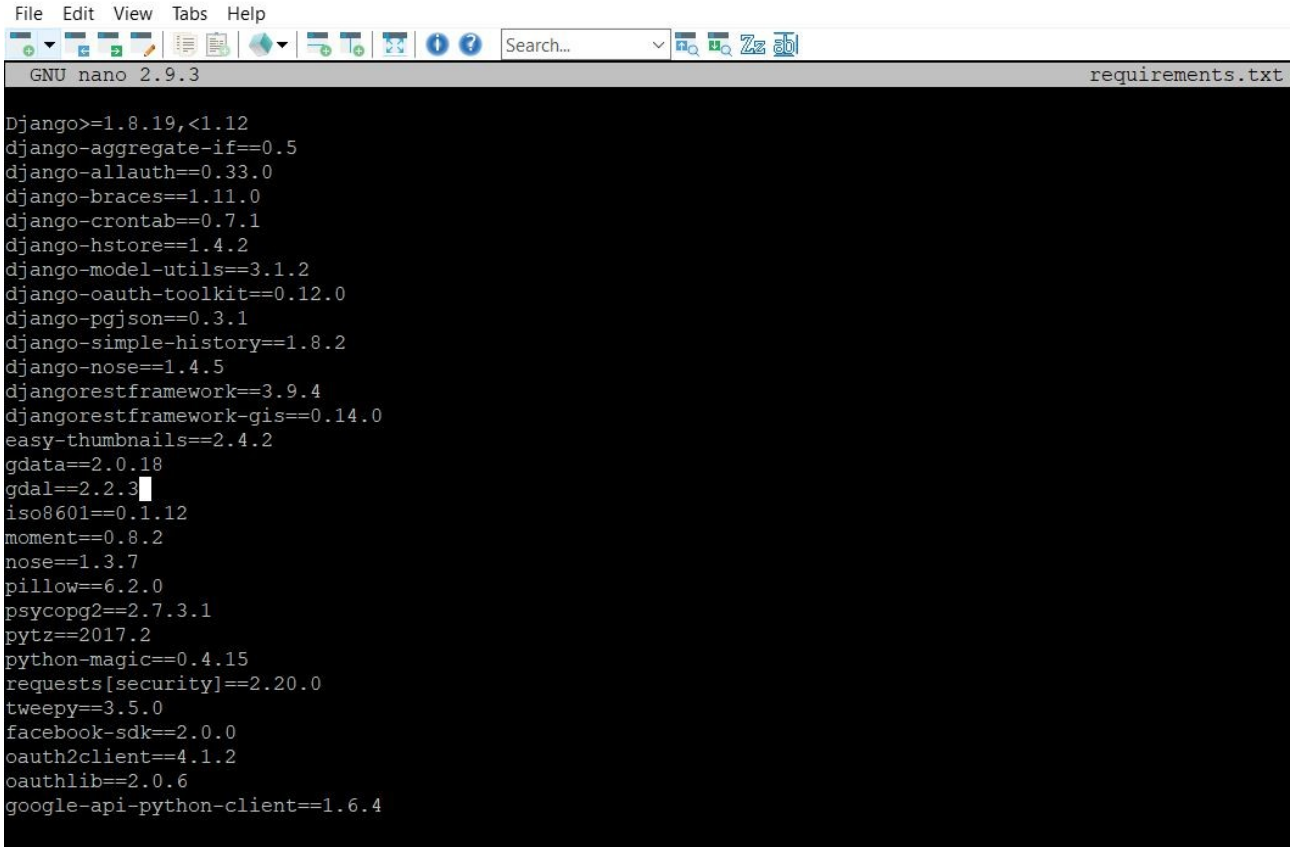
```
gdalinfo –version
```

4.8 Use the same version (x.x.x) to install GDAL as a Python package:

```
pip install GDAL==x.x.x
```

4.9 To avoid version conflicts, we need to update pip installation requirements document: (Update gdal==1.10.0 to x.x.x)

```
nano requirements.txt
```



4.10 Updating the development requirements file, continue with pip command to install GeoKey setup packages and development requirements.

```
pip install -r requirements.txt
```

4.11 Copy the directory local_settings.example to local_settings

```
cp -r local_settings.example local_settings
```

4.12 Inside the local_settings open ***settings.py*** in a text editor and

```
nano local_settings/settings.py
```

4.12.1 Add your database settings:

```
DATABASES = {
    'default': {
        'ENGINE': 'django.contrib.gis.db.backends.postgis',
        'NAME': 'geokey',
        'USER': 'django',
        'PASSWORD': 'xxxxxxxxx',
        'HOST': os.environ.get('DJANGO_DATABASE_HOST','localhost'),
        'PORT': '',
    }
}
```

4.12.2 Set the secret key

```
SECRET_KEY = 'xxxxxxxxxxxxxxx'
```

4.12.3 Set the static root directory

```
STATIC_ROOT = '/var/www/geokey/static'
```

4.13 Inside the local_settings open ***wsgi.py*** in a text editor and configure as follows for the production environment

```
nano local_settings/wsgi.py
```

4.13.1 Configure as follows for the production environment

```
"""WSGI configuration."""

import os
import sys

from django.core.wsgi import get_wsgi_application
sys.path.append('/var/www/geokey')
sys.path.append('/var/www/geokey/env/lib/python2.7/site-packages')

try:
 settings_module = 'local_settings.settings'
except ImportError:
 settings_module = 'geokey.core.settings.project'

os.environ.setdefault('DJANGO_SETTINGS_MODULE', settings_module)
application = get_wsgi_application()
```

4.14 Migrate the database

```
python manage.py migrate
```

4.15 Add yourself as a superuser (you can use the same email and password to log into the system later)

```
python manage.py createsuperuser
```

4.15 Run the collectstatic management command to copy new and updated static files

```
python manage.py collectstatic
```

4.16 Configure Apache

```
nano /etc/apache2/sites-available/000-default.conf
```

4.16.1 Modify as follows:

```
<VirtualHost *:80>
ServerName localhost
 DocumentRoot /var/www/geokey
 WSGIScriptAlias / /var/www/geokey/local_settings/wsgi.py

WSGIDaemonProcess localhost processes=2 threads=15 python-home=/var/www/geokey/env/
WSGIProcessGroup localhost
WSGIApplicationGroup %{GLOBAL}
 WSGIProcessGroup localhost

 <directory /var/www/geokey/>
   AllowOverride all
   Require all granted
   Options FollowSymlinks
 </directory>

 Alias /static/ /var/www/geokey/static/

 <Directory /var/www/geokey/static>
  Require all granted
 </Directory>
</VirtualHost>
```

4.17 Restart Apache

```
service Apache2 restart
```

B. Installing GeoKey Extensions

1. Installing geokey-sapelli extension

Extension for GeoKey to add support for Sapelli. This extension enables users to upload Sapelli project files (resulting in corresponding GeoKey projects), and Sapelli-produced data via the admin interface.

geokey-sapelli requires min:

- Python version 2.7
- Java (JRE or JDK) version 7 or up
- GeoKey version 1.6 or greater

1.1 Connect remote server through ssh

```
ssh root@XXX.XX.XXX.XX (IP adress)
```

1.2 Installing Java

The easiest option for installing Java is to use the version packaged with Ubuntu. By default, Ubuntu 18.04 includes Open JDK, which is an open-source variant of the JRE and JDK.

To install this version, first update the package index:

```
sudo apt update
```

Next check if Java already installed

```
java -version
```

If Java is not currently installed, you'll see the following output:

```
Command 'java' not found, but can be installed with:

apt install default-jre
apt install openjdk-11-jre-headless
apt install openjdk-8-jre-headless
apt install openjdk-9-jre-headless
```

Execute the following command to install OpenJDK:

```
sudo apt install default-jre
```

Verify the installation with:

```
java -version
```

You'll see the following output:

```
openjdk version "11.0.6" 2020-01-04
OpenJDK Runtime Environment (build 11.0.6+10-post-Ubuntu-1ubuntu118.04.1)
OpenJDK 64-Bit Server VM (build 11.0.6+10-post-Ubuntu-1ubuntu118.04.1, mixed
mode,sharing)
```

Finally modify the following installation document checking configuration. We mainly specify Java version and update the condition as follows:

```
cd /var/www/geokey/env/lib/python2.7/site-packages/geokey_sapelli/helper/

nano install_checks.py
```

Make sure not to leave unnecessary spaces while modifying the file as they may lead to compile failures in Python. For easiness, you may copy code blocks from a Gist , simple way to share snippets and pastes with others.

https://gist.github.com/gulbalasalamov/3ee8a086260ef2b8b47482ebbfbcf449

```
import commands
import re

from django.conf import settings

from geokey.applications.models import Application

from .sapelli_exceptions import SapelliException
from .sapelli_loader import get_sapelli_dir_path, get_sapelli_jar_path

MINIMAL_JAVA_VERSION = '1.7.0'

def check_extension():
    # Check if SAPELLI_CLIENT_ID value is set in settings.py:
    try:
        client_id = settings.SAPELLI_CLIENT_ID
    except AttributeError:
        raise SapelliException('no SAPELLI_CLIENT_ID value set in geokey
settings.py.')
```

```
    # Check if an application is registered with this client_id:
    try:
        Application.objects.get(client_id=client_id,
authorization_grant_type='password')
    except Application.DoesNotExist:
        raise SapelliException('geokey_sapelli is not registered as an
application (with password authorisation) on the server.')
    # Check if java 1.7.0 or more recent is installed:
    try:
        status_output = commands.getstatusoutput('java -version')
        if(status_output[0] != 0):
            raise SapelliException('java not installed, please install JRE v7 or
later.')
        java_version = re.match(r'java version "(?P<java_version>[0-9]+\.[0-
9]+\.[0-9]+)_.*', status_output[1])
        if java_version is None:
            java_version_res = '11.0.6'
        else:
            java_version_res = java_version.group('java_version')
        if(java_version_res < MINIMAL_JAVA_VERSION):
            raise SapelliException('installed version of java is too old
(installed: %s, minimum required: %s).' % (java_version, MINIMAL_JAVA_VERSION))
    except BaseException, e:
        raise SapelliException('could not run java command (%s).' % str(e))
    # Check if there is a sapelli working directory:
    get_sapelli_dir_path()  # raises SapelliException
    # Check if we have the sapelli JAR:
    get_sapelli_jar_path()  # raises SapelliException
```

1.3 Change directory of the installation of GeoKey

```
cd /var/www/geokey
```

1.4 Activate virtual environment

```
source env/bin/activate
```

1.5 Install the geokey-sapelli from PyPI

```
pip install geokey-sapelli
```

1.6 Add the package to installed apps using the text editor

```
nano local_settings/settings.py
```

```
INSTALLED_APPS += (
      …
      ‘geokey_sapelli’,
      )
```

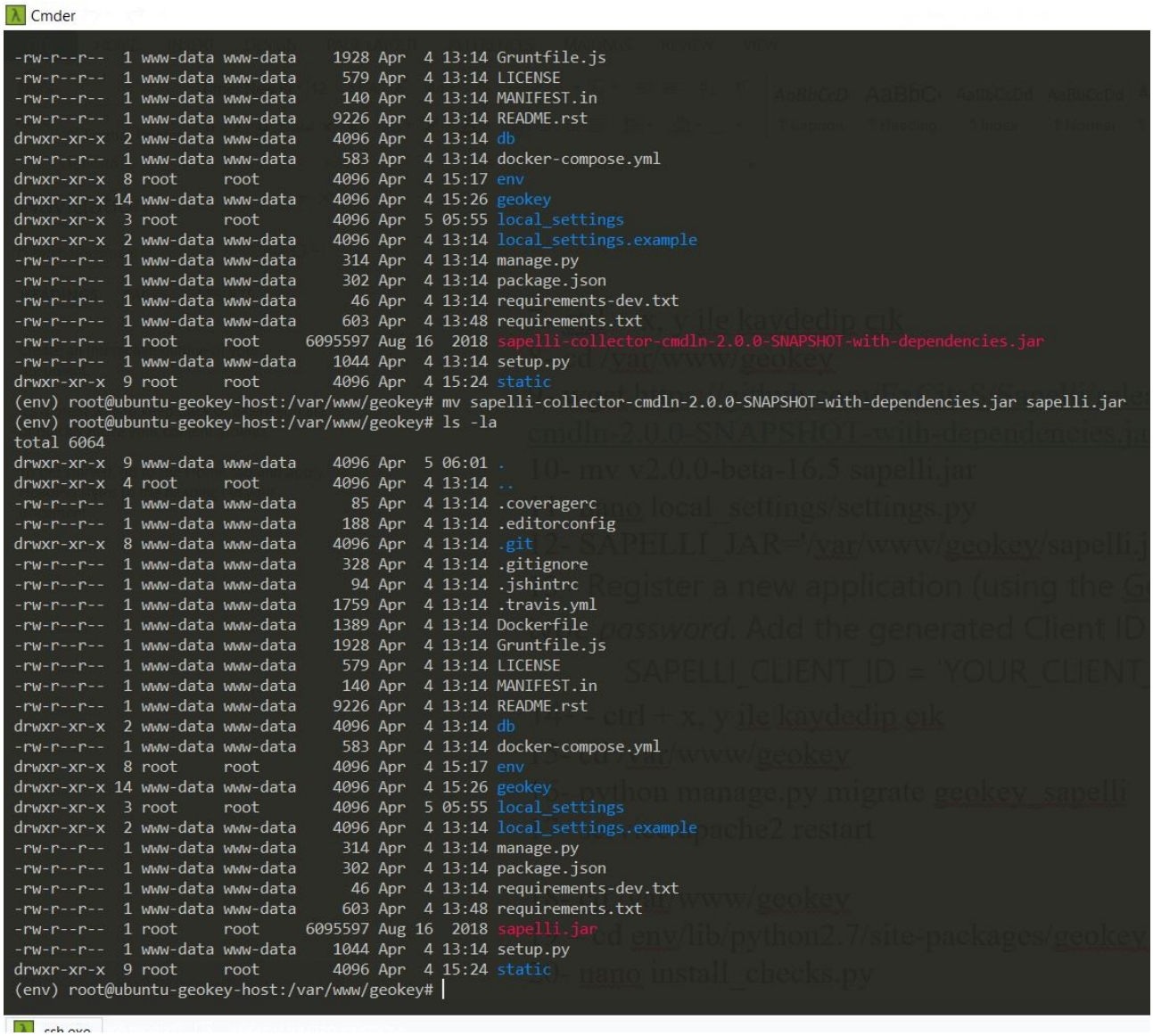
Download the latest available version of the Sapelli Collector CmdLn front-end (the file you need looks like *sapelli-collector-cmdln-X.X.X-XXXXXX-with-dependencies.jar*). Place the file in a folder of your choice (you can rename it as well if you want)

```
wget https://github.com/ExCiteS/Sapelli/releases/download/2.0.0-beta-28/sapelli-collector-cmdln-2.0.0-SNAPSHOT-with-dependencies.jar
```

Rename it to call easier

```
mv sapelli-collector-cmdln-2.0.0-SNAPSHOT-with-dependencies.jar sapelli.jar
```

You can use ls -la to check the file has been renamed.



Edit the settings.py file to add the absolute path to the bottom of file:

```
nano local_settings/settings.py
```

```
SAPELLI_JAR='/var/www/geokey/sapelli.jar'
```

Register a new application (using the GeoKey admin interface) with authorisation type password. Add the generated Client ID to your settings.py:

```
SAPELLI_CLIENT_ID='YOUR_CLIENT_ID'
```

Migrate the models into the database:

```
python manage.py migrate geokey_sapelli
```

Restart Apache to take change affect

```
service apache2 restart
```

2. Installing geokey-epicollect extension

Use EpiCollect's phone app as a mobile client to collect data and store it with GeoKey.

geokey-epicollect requires:

- Python version 2.7
- GeoKey version 1.6 or greater

2.1 Connect remote server through ssh

```
ssh root@XXX.XX.XXX.XX (IP adress)
```

2.2 Change directory of the installation of GeoKey

```
cd /var/www/geokey
```

2.3 Activate virtual environment

```
source env/bin/activate
```

2.4 Make sure dependencies are installed

```
sudo apt-get install libxslt1-dev libxml2-dev
```

2.5 Install the geokey-epicollect by cloning repository

```
git clone https://github.com/ExCoteS/geokey-epicollect
```

2.6 Modify the initialization configuration. Put the stament in try catch block

```
cd geokey-epicollect

nano geokey_epicollect/__init__.py
```

You may copy paste from following Gist.

https://gist.github.com/gulbalasalamov/1abadbb972897e87f14b9dac9676a8a8

2.7 Install the packaged

```
pip install -e .
```

2.8 Add the package to installed apps using the text editor

```
cd ..

nano local_settings/settings.py
```

```
INSTALLED_APPS += (
      …
      'geokey_epicollect',
      )
```

2.9 Migrate the models into the database:

```
python manage.py migrate geokey_epicollect
```

2.10 Restart Apache to take change affect

```
service apache2 restart
```

3. Installing geokey-cartodb extension

Provide API endpoints that can be used to import GeoKey data into CartoDB.

geokey-epicollect requires:

- Python version 2.7
- GeoKey version 1.6 or greater

3.1 Connect remote server through ssh

```
ssh root@XXX.XX.XXX.XX (IP adress)
```

3.2 Change directory of the installation of GeoKey

```
cd /var/www/geokey
```

3.3 Activate virtual environment

```
source env/bin/activate
```

3.4 Install the geokey-cartodb by cloning repository

```
git clone https://github.com/ExCiteS/geokey-cartodb
```

3.5 Modify the initialization configuration. Put the stament in try catch block

```
cd geokey-cartodb

nano geokey_cartodb/__init__.py
```

You may copy paste from following Gist.

https://gist.github.com/gulbalasalamov/ecbbd1bd0d71b3736c7d10e4b4dd6fb5

3.6 Install the packaged

```
pip install -e .
```

3.7 Add the package to installed apps using the text editor

```
cd ..

nano local_settings/settings.py
```

```
INSTALLED_APPS += (
      …
      ‘geokey_cartodb’,
      )
```

3.8 Migrate the models into the database:

```
python manage.py migrate geokey_cartodb
```

3.9 Restart Apache to take change affect

```
service apache2 restart
```

4. Installing geokey-export extension

Export data from GeoKey into various formats.

Currently supported formats:

- KML
- JSON
- CSV

geokey-export requires:

- Python version 2.7
- GeoKey version 1.6 or higher

4.1 Connect remote server through ssh

```
ssh root@XXX.XX.XXX.XX (IP adress)
```

4.2 Change directory of the installation of GeoKey

```
cd /var/www/geokey
```

4.3 Activate virtual environment

```
source env/bin/activate
```

4.4 Install the geokey-export by cloning repository

```
git clone https://github.com/ExCiteS/geokey-export
```

4.5 Modify the initialization configuration. Put the stament in try catch block

```
cd geokey-export

nano geokey_export/__init__.py
```

You may copy paste from following Gist.

https://gist.github.com/gulbalasalamov/943cbfc7b295de58404a5519e9abb9e9

4.6 Install the packaged

```
pip install -e .
```

4.7 Add the package to installed apps using the text editor

```
cd ..

nano local_settings/settings.py
```

```
INSTALLED_APPS += (
      …
      ‘geokey_export’,
      )
```

4.8 Migrate the models into the database:

```
python manage.py migrate geokey_cartodb
```

4.9 Copy static files:

```
python manage.py collectstatic
```

4.10 Restart Apache to take change affect

```
service apache2 restart
```

5. Installing geokey-dataimports extension

Import data from various formats (GeoJSON, KML, CSV) into GeoKey.

geokey-dataimport requires:

- Python version 2.7
- GeoKey version 1.6 or higher

5.1 Connect remote server through ssh

```
ssh root@XXX.XX.XXX.XX (IP adress)
```

5.2 Change directory of the installation of GeoKey

```
cd /var/www/geokey
```

5.3 Activate virtual environment

```
source env/bin/activate
```

5.4 Install the geokey-export by cloning repository

```
git clone https://github.com/ExCiteS/geokey-dataimports
```

5.5 Install the packaged

```
cd geokey-dataimports

pip install -e .
```

5.6 Add the package to installed apps using the text editor

```
cd ..

nano local_settings/settings.py
```

```
INSTALLED_APPS += (
      …
      ‘geokey_dataimports’,
      )
```

5.7 Migrate the models into the database:

```
python manage.py migrate geokey_dataimports
```

5.8 Copy static files:

```
python manage.py collectstatic
```

5.9 Restart Apache to take change affect

```
service apache2 restart
```

---

6. Installing geokey-webresources extension

Extend GeoKey projects by adding web resources: GeoJSON and KML.

geokey-webresources requires:

- Python version 2.7
- GeoKey version 1.6 or higher

6.1 Connect remote server through ssh

```
ssh root@XXX.XX.XXX.XX (IP adress)
```

6.2 Change directory of the installation of GeoKey

```
cd /var/www/geokey
```

6.3 Activate virtual environment

```
source env/bin/activate
```

6.4 Install the geokey-export by cloning repository

```
git clone https://github.com/ExCiteS/geokey-webresources
```

6.5 Install the packaged

```
cd geokey-webresources

pip install -e .
```

6.6 Add the package to installed apps using the text editor

```
cd ..
```

```
nano local_settings/settings.py
```

```
INSTALLED_APPS += (
      …
      ‘geokey_webresources’,
      )
```

6.7 Migrate the models into the database:

```
python manage.py migrate geokey_cartodb
```

6.8 Restart Apache to take change affect

```
service apache2 restart
```