|  |  |
| --- | --- |
| CREATE VIRTUAL ENVIRONMENT |  |
|  |  |
|  | Mkdir djangogirls |
|  | Cd djangogirls |
| Create virtual environment called myvenv | python3 -m venv myvenv |
| Start the myvenv | source myvenv/bin/activate |
|  |  |
| INSTALL DJANGO |  |
| Check that we have correct version of pip | python -m pip install --upgrade pip |
| **Installing Packages with requirements** | Create requirements.txt in djangogirls directory and add this text  Django~=3.2.10 |
| Install Django | pip install -r requirements.txt |
| INSTALL GIT |  |
| Download GIT | [git-scm.com](https://git-scm.com/)  You should change the ownership of these directories to your user.  sudo chown -R $(whoami) /usr/local/bin  And make sure that your user has write permission.  chmod u+w /usr/local/bin  Install homebrew if not already done  brew install git  Install git-gui  brew install git-gui |
| Create a GitHub Account | https://github.com/ |
| CREATE PythonAnywhere |  |
| Create a PythonAnywhere account for running Python code on servers "in the cloud". We'll use it for hosting our site, live and on the Internet. | Pythonanywhere.com |
| Create Pythonanywhere API |  |
|  |  |
| CREATE DJANGO PROJECT |  |
| Create Django project called mysite | django-admin startproject mysite . |
| Change settings.py |  |
| Create a dtaabase as defined in manage.py and settings.py | python manage.py migrate |
| Start web server | python manage.py runserver |
| Now you need to check that your website is running. Open your browser (Firefox, Chrome, Safari, Internet Explorer or whatever you use) and enter this address:  http://127.0.0.1:8000/ |  |
| To type additional commands while the web server is running, open a new terminal window and activate your virtualenv |  |
| CREATE APPLICATION INSIDE DJANGO PROJECT |  |
|  |  |
| Create application | python manage.py startapp blog |
| After creating an application, we also need to tell Django that the project *mysite* should use it. | Add ‘blog’ in mysite/settings.py |
| Create the blog post model | Models.py  from django.conf import settings  from django.db import models  from django.utils import timezone  class Post(models.Model):  author = models.ForeignKey(settings.AUTH\_USER\_MODEL, on\_delete=models.CASCADE)  title = models.CharField(max\_length=200)  text = models.TextField()  created\_date = models.DateTimeField(default=timezone.now)  published\_date = models.DateTimeField(blank=True, null=True)  def publish(self):  self.published\_date = timezone.now()  self.save()  def \_\_str\_\_(self):  return self.title |
| Create table for model in the database | This adds the new model to our database  python manage.py makemigrations blog  Now Django prepared a migration file for us that we now have to apply to our database. |
| Apply migration File to databse | python manage.py migrate blog |
|  |  |