NOTE: Root of Django project is where manage.py is.

To Create Django Project manually from terminal:

- https://www.youtube.com/watch?v=V-GNjychIK0&list=PLEsfXFp6DpzTD1BD1aWNxS2Ep06vIkaeW&index=4

// Follows Udemy Tutorial
To create Virtual Env:

- virtualenv -p python3 <name of environment)
- virtualenv -p python3 . --- With dot

To activate Virtual Env:

- source <name of environment>/bin/activate
- If used With dot, only require: source bin/activate

To deactivate a Virtual Env:

- deactivate

To create a new website directory:

- django-admin startproject <name of website>

To create an app within website (Current working directory should be name of website):

- python manage.py startapp <name of app>
- After this, add the name of your app to the settings.py file of the main website in the INSTALLED_APPS list.

To run your app:

- python manage.py runserver

To make migrations. Migrations are to make our models and database remain in-sync. Run both of these commands together whenever you make changes to the models.py file:

- python manage.py makemigrations <name of app>
- python manage.py migrate

To populate models through the shell:

- python manage.py shell
- In the shell:
    - Import django
    - django.setup()
    - from django.utils import timezone
    - from <name of app>.models import Question, Choice (name of models)
    - q = Question(question_text = ‘What\’s your name’, published_date = timezone.now()
    - q.save()
    - ####### To add choices
    - q = Question.objects.get(pk=1)
    - q.choice_set.create(choice_text = ‘Bob’, votes = 0)
    - q.choice_set.create(choice_text = ‘Rachel’, votes = 0)
    - q.choice_set.create(choice_text = ‘Fred’, votes = 0)
    - q.save()

To access **admin tool**, we need a login or a superuser. This user has access to the admin tool and is not regular user. To create superuser:

- python manage.py createsuperuser
- Add in the details prompted
- Then go to: 127.0.0.1/admin and type-in username and password.

To add models to admin tool:

- Go to admin.py of your app
- Add the following code:
    - From .models import Question, Choice (your models)
    - admin.site.register(Question)
    - admin.site.register(Choice)

To link URLs in MySite to Polls:

- Go to urls.py in MySite directory.
- Add: from django.urls import include
- In the urlpatterns list, add the following code:
    - path(‘polls/’, include(‘polls.urls’)),
- Create a urls.py file in the polls directory.
- Add all code from Section 4, Playing Around With URLs video beginning from 2:17.

To link a view and display it:

- Section 4, Playing Around With URLs from 2:17

To work with templates and HTML:

- Section 4, T for Templates!

## From Code For Entrepreneurs Tutorial

To work with Shell:

- python manage.py shell
- To import your model(s):
    - From products.models import Product
    - From <app name>.models import <name of model(s)>
- To see all objects of a model:
    - Product.objects.all()
    - <model name>.objects.all()
- To add an object:
    - Product.objects.create(title=’’, description=’’, price=’’, summary=’’)
    - <model name>.objects.create(attribute1=value, attribute2=value)

For arguments blank and null in our models:

- blank has to do with how the field is rendered i.e. whether it is required or not.
    - blank=false means it’s not required
- null has to do with whether it can be null in our database or not.
    - null=false means it cannot be null in our database.

## Precreated Django Files

Settings.py: Configuration center of the Django project. Contains information on where Django is located, apps that are part of the website, databases, static files etc.

INSTALLED_APPS: Has all the built-in components like the Admin Tool and this is also where we add the apps that we create for Django to be able to run them.

## From Net Ninja Tutorial

1. Created a url for article detail that operated on <slug:slug>
2. When you click on an article heading in the article-list view, backend code executes:
    a. href of anchor tag receives a DTL Tag called {% url %}.
    b. This tag's first property is the **URL name** that needs to be fired.
    c. The second property is the parameter we want to send with it.
    d. When you click on the heading, the URL is updated with the parameter we want to send back.
    e. The **URL name** specifies which view will be activated and the parameters will get sent to that view.

- forms.ModelForm: The Meta class specifies how we want to output our form, from which model we want to inherit our fields into our form and which fields we want to output.
- form.save(commit=False): commit=False means to wait for a second and not save the form just yet. Rather give us the instance of the form that you're about to save.

Link:
https://www.youtube.com/watch?v=OuKLvoHd5oE&list=PL4cUxeGkcC9ib4HsrXEYpQnTOTZE1x0uc&index=16

## From Pluralsight's Tutorial For Django:

- To see all pending migrations:
    - python manage.py showmigrations
- To open sqllite3 shell:
    - python manage.py dbshell

## From Pluralsight's tutorial on Python PEP8

To make PyCharm treat Type Hint warnings as errors:
- Go to **Improving Your Code With Type Checking**.

- Open **Demo: Type Hints** video.
- Go to 1:50.

## Comments

- When you delete a database, you’ve to create superuser again.
- When you mark an attribute in the model as blank=True and null=true, you don’t need to send it in the terminal as an argument when creating an object.
- To check whether a user is either logged-in or logged-out in our templates to display them different content, we need to basically check if a user is authenticated. That means that the user is logged-in.
- Inheriting a model class from models.Model tells the database to map this class to a database model.

## Add Repositories To GitHub

https://help.github.com/en/github/importing-your-projects-to-github/adding-an-existing-project-to-github-using-the-command-line