WEB APPLICATION DEVELOPMENT WITH

DJANGO

To get started:

+ Install Python 2.7+
+ Install Django

Presentation!

http://pramodk.net/django/

Presentation flow:

```
mkdir django_short_urls
cd django_short_urls
git init

# Create a new Django project and app
django-admin startproject shortlink
cd shortlink
python manage.py startapp shorturls

# Open shortlink folder in Atom
# Show that shortlink/shortlink is an app within the project "shortlink" for project settings

git add --all
git commit -m "Initial commit"

python manage.py runserver
# Chrome: localhost:8000

# Explain shortlink/urls.py file and how Django matches HTTP request URL to regex pattern
16:  from django.conf.urls import url, include
+21: url(r'', include(r'$', shorturls.urls')),
```

```
# Create shorturls/urls.py:
from django.conf.urls import url
From . import views

urlpatterns = [
     url(r'shorten/', views.home),
]


# In shorturls/views.py
+2: from django.http import HttpResponse

def home(request):
     return HttpResponse("Welcome home!")

# Show how the console has been updating the entire time

# MODELS
# Now we need to create database models to represent a ShortUrl object
# In shorturls/models.py:
class ShortLink(models.Model):     # like Java's public class ShortLink extends Model ...
     long_url = models.CharField(max_length=256, blank=False)
     short_id = models.CharField(max_length=4, blank=False)

     def __str__(self):
        return self.short_id.__str__() + " --> " + self.long_url.__str__()

# Terminal:
Ctrl-c
# Explain what a migration is
python manage.py makemigrations
python manage.py migrate
# Explain what migrate does
# Explain the concept of abstracting communications with relational databases (any DB works
with Django)
```

```
git status
git add --all        # mention to not include .pyc files
git commit -m "Added db model, views, urls"

# shortlink/shorturls/views.py
# Explain what the views.py file should do
+4: from .models import ShortLink

# Create: shortlink/shorturls/forms.py
# Explain why Django forms are awesome: Very DRY, derives HTML input types from model,
near-perfect validation system, etc.
from django import forms
from django.forms import ModelForm

from .models import ShortLink

class ShortlinkForm(ModelForm):
     class Meta:
          model = ShortLink
          fields = '__all__'

# discuss how my thoughts instantly become actions with little in-between thought!

# In shorturls/views.py
+5: from .forms import ShortlinkForm
+12:
def create(request):
     if request.method == 'POST':
          pass
     else:
          form = ShortlinkForm()
     return render(request, 'create.html', {'form': form, 'form_action_uri': ''})
+3:
from django.views.decorators.http import require_http_methods
+12:
\n
@require_http_methods(['GET', 'POST'])
```

```
# Create templates/create.html
<h1>Create short links!</h1>

<form id="main-form" action="{{ form_action_uri }}" method='POST' class="form-horizontal">
     {% csrf_token %}
    {{ form }}
    {{ form.errors }}
     <input class="" type="submit" value="Submit!" />
</form>

# In shorturls/urls.py
# Point r‘shorten/’ to views.create

git add --all
git commit -m “Created initial shorten path”

# Explain where we are at and where we are going!
# Update forms.py so fields = (‘long_url’,) only
# We need to handle the form from the client!
@require_http_methods(['GET', 'POST'])
def create(request):
    if request.method == 'POST':
        form = ShortlinkForm(request.POST)
        if form.is_valid():
            long_url = form.cleaned_data['long_url']
            short_id = get_md5(long_url, 4)
            new_shortlink = ShortLink(long_url=long_url, short_id=short_id)
            new_shortlink.save()
            return render(request, 'create.html', {'new_shortlink': new_shortlink})
    else:
        form = ShortlinkForm()
    return render(request, 'create.html', {'form': form})

def get_md5(string, length):
    encoded_string = string.encode(encoding='UTF-8', errors='strict')
    hashed = hashlib.md5(encoded_string).hexdigest()
```

```
    return hashed[0:length]


# Update create.html
<h1>Create short links!</h1>

{% if form %}
<form id="main-form" action="{{ form_action_uri }}" method='POST' class="form-horizontal">
{% csrf_token %}
    {{ form }}
    {{ form.errors }}
     <input class="" type="submit" value="Submit!" />
</form>
{% endif %}

{% if new_shortlink %}
    {{ new_shortlink }}
{% endif %}


git add --all
git commit -m "Initial shortlink form"


# Let's create a way for me to redirect you to your site given the pattern I give you
# Append to shorturls/urls.py

    url(r'(?P<pattern>[0-9a-zA-Z]{1,4})$', views.match),

# Add to shorturls/views.py
from django.http import HttpResponse, HttpResponseRedirect

@require_http_methods(['GET'])
def match(request, pattern):
    try:
        shortlink = ShortLink.objects.filter(short_id=pattern.lower())[0]
```

```
    except ShortLink.DoesNotExist:
        return HttpResponse("Cannot find pattern")
    return HttpResponseRedirect(shortlink.long_url)

git add --all
git commit -m “Added matching service”

# Admin page:
from django.contrib import admin
from .models import ShortLink

class LinkAdmin(admin.ModelAdmin):
    pass
admin.site.register(ShortLink, LinkAdmin)

python manage.py createsuperuser

# Initial deployment!
# Add to shortlink/settings.py
STATIC_ROOT = 'staticcontent/'

# Add Procfile to root
web: gunicorn shortlink.wsgi --log-file -

# Add requirements.txt
dj-database-url==0.3.0
Django==1.9.1
gunicorn==19.4.5
psycopg2==2.6.1
whitenoise==2.0.6
requests==2.9.1

git add --all
git commit -m “Added deployment settings”

k-pramod : GitHubIOCOM@)1%
```

More topics:
* python manage.py shell
* for loop through all links
* create a more comprehensive model and show validation and admin page:
    * Student: email, graduation year, name, description