YOUR FIRST DJANGO PROJECT

06016322 - Web Programming

Creating a project

- From the command line, cd into a directory where you'd like to store your code, then run the following command:

```
$ django-admin startproject mysite
```

```
mysite/
    manage.py
    mysite/
        __init__.py
        settings.py
        urls.py
        wsgi.py
```

FOLDER STRUCTURE

Let’s look at
what startproject created:

Let’s start your development server

- Let’s verify your Django project works. Change into the outer mysite directory and run the following commands:

```
$ python manage.py runserver
```

```
Performing system checks...

System check identified no issues (0 silenced).

You have unapplied migrations; your app may not work properly until they are applied.
Run 'python manage.py migrate' to apply them.

February 28, 2019 - 15:50:53
Django version 2.1, using settings 'mysite.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CONTROL-C.
```

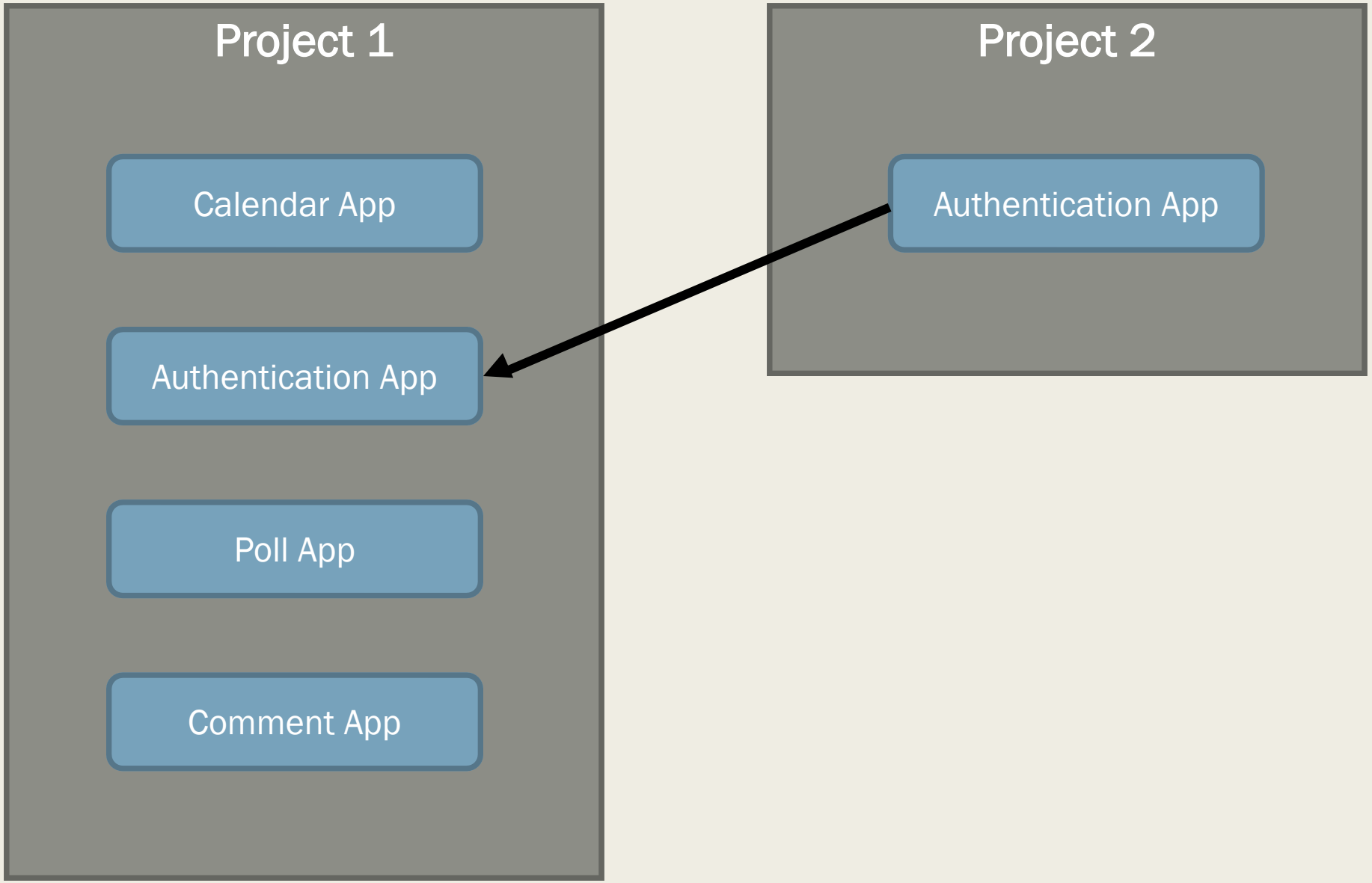
Creating the Polls app

- Django comes with a utility that automatically generates the basic directory structure of an app, so you can focus on writing code rather than creating directories.

Project VS. App

- To create your app, make sure you're in the same directory as manage.py and type this command:

```
$ python manage.py startapp polls
```


Project 1
Calendar App
Authentication App
Poll App
Comment App
Project 2
Authentication App

```
polls/
    __init__.py
    admin.py
    apps.py
    migrations/
        __init__.py
    models.py
    tests.py
    views.py
```

IN YOUR POLLS APP

The startapp command will create a directory polls, which is laid out like this:

Let’s create your first view

- Open the file polls/views.py and start writing some code
- To call the view, we need to map it to a URL - and for this we need a URLconf.
- To create a URLconf in the polls directory, create a file called urls.py.

```
polls/
    __init__.py
    admin.py
    apps.py
    migrations/
        __init__.py
    models.py
    tests.py
    urls.py
    views.py
```

Path() function

- The path() function is passed four arguments, two required: route and view, and two optional: kwargs, and name.
- **Argument: route**
  - *route* is a string that contains a URL pattern.
- **Argument: view**
  - When Django finds a matching pattern, it calls the specified view function with an HttpRequest object as the first argument and any “captured” values from the route as keyword arguments.
- **Argument: kwargs**
  - Arbitrary keyword arguments can be passed in a dictionary to the target view.
- **Argument: name**
  - Naming your URL lets you refer to it unambiguously from elsewhere in Django, especially from within templates.

Let’s create some more views that actually do something

- Create some views in views.py
  - *Poll homepage – displays the list of all polls.*
  - *Poll “detail” page*
  - *Poll “result” page*

- Add path() in urls.py

Poll homepage - index

Welcome to my poll page

Popular polls

- เพื่อน ๆ คิดว่าวิชาเวบโปรสอนรู้เรื่องไหม? vote result
- เพื่อน ๆ คิดว่า อ. บัณฑิต อายุเท่าไหร่? vote result
- เพื่อน ๆ จะ drop วิชาเวบโปรกันไหม vote result

Frequently used templatetags

- For

```
<ul>
{% for athlete in athlete_list %}
    <li>{{ athlete.name }}</li>
{% endfor %}
</ul>
```

- If ... else

```
{% if athlete_list %}
    Number of athletes: {{ athlete_list|length }}
{% elif athlete_in_locker_room_list %}
    Athletes should be out of the locker room soon!
{% else %}
    No athletes.
{% endif %}
```

- Include

```
{% include "foo/bar.html" %}
```

- **now** - Displays the current date and/or time, using a format according to the given string.

```
It is {% now "jS F Y H:i" %}
```

- url

```
{% url 'some-url-name' v1 v2 %}
```

https://docs.djangoproject.com/en/2.1/ref/templates/builtins/

Path converters

- The following path converters are available by default:
  - *str - Matches any non-empty string, excluding the path separator, '/'. This is the default if a converter isn't included in the expression.*
  - *int - Matches zero or any positive integer. Returns an int.*
  - *slug - Matches any slug string consisting of ASCII letters or numbers, plus the hyphen and underscore characters. For example, building-your-1st-django-site.*

Path converter examples

```
urlpatterns = [
    path('articles/2003/', views.special_case_2003),
    path('articles/<int:year>/', views.year_archive),
    path('articles/<int:year>/<int:month>/', views.month_archive),
    path('articles/<int:year>/<int:month>/<slug:slug>/', views.article_detail),
]
```

- /articles/2003 -> no match
- /articles/2005/03/ -> views.month_archive(request, year=2005, month=3)
- /articles/2003/03/building-a-django-site/ -> views.article_detail(request, year=2003, month=3, slug="building-a-django-site")

Create some templates for those views

- Create 3 html files in templates/polls folder
  - *index.html*
  - *detail.html*
  - *results.html*

Now! Let’s style out polls website

- We’ve built a tested Web-poll application, and we’ll now add a stylesheet and an image.

- Aside from the HTML generated by the server, web applications generally need to serve additional files — such as images, JavaScript, or CSS — necessary to render the complete web page. In Django, we refer to these files as “static files”.

Customize your *app's* look and feel

- First, create a directory called **"static"** in your polls directory. Django will look for static files there.
- Within the static directory you have just created, create another directory called **polls** and within that create a file called **style.css**.
- Next, add "{% **load** static %}" at the top of your html file:

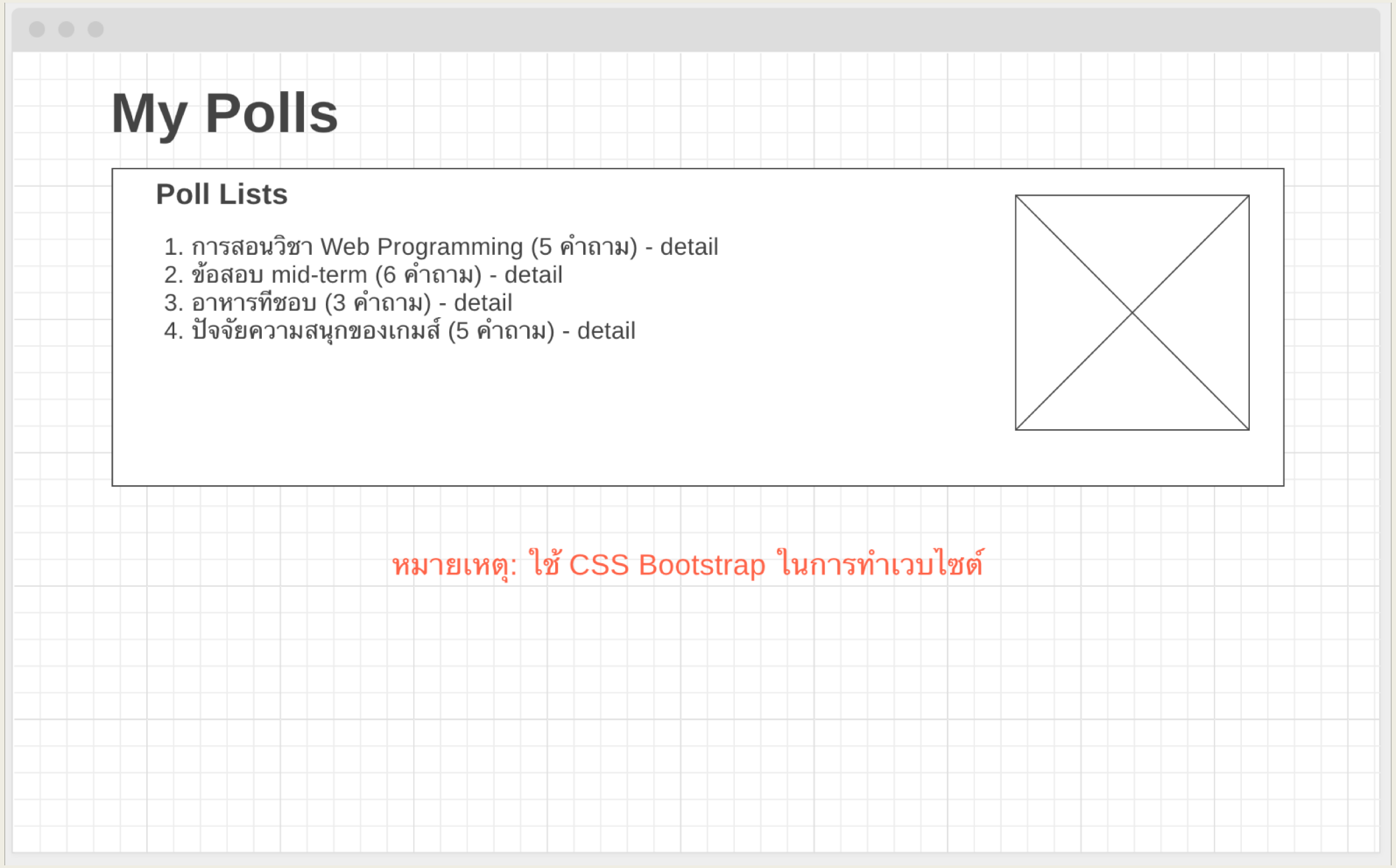
```
<link rel="stylesheet" type="text/css" href="{% static 'polls/style.css' %}">
```

EXERCISES

Index Page

คำสั่ง

- ปรับปรุงเพิ่มเติมหน้า polls/index.html โดยใช้ Bootstrap
- เพิ่มรูปอะไรก็ได้เข้าไปที่หน้าเวบ (ดังในรูป)
- เมื่อกดปุ่ม detail ให้แสดงหน้ารายละเอียดของ poll นั้น ๆ
- ใช้ข้อมูลในไฟล์ data.txt



Detail Page

คำสั่ง

- แสดงรายการคำถาม พร้อมคำตอบ
- คำตอบเป็น radio button
- ถ้ากด “กลับหน้าแรก” กลับไปหน้า poll index
- ถ้ากด “บันทึก” ให้ส่งค่า form ไปที่ view page detail และทำการ print ค่าที่ post ออกมาใน console
- ใช้ข้อมูลในไฟล์ data.txt