# Contents

1 Simple is Power 3

2 Terminal Prompt 5

3 Table of Contents 7
  3.1 Writing your first Django app, part 1 7
  3.2 Writing your first Django app, part 2 9
  3.3 Writing your first Django app, part 3 14
  3.4 Writing your first Django app, part 4 22
  3.5 Writing your first Django app, part 5 26
  3.6 Writing your first Django app, part 6 27
  3.7 Want List 29
Hosting:

- Read The Docs, Django2.1 Tutorial Lab

The main purpose of this project is to practice Django 2.1 Getting started effectively.

Like Yoga’s daily practice, I need a routine to awaken my body and mind. Same as programming, I need lab to maintain skills to invoke idea.
CHAPTER 1

Simple is Power

Repeat practice and find out your weakness.
You have your right to be an individual thinker.
Terminal Prompt

Before activate virtual environment:

```
export PS1='$ '  
```

After activate virtual environment:

```
export PS1='(venv)$ '  
```

To determine which python:

```
which python  
```
3.1 Writing your first Django app, part 1

• Django Getting started, part 1
• Read The Docs, part 1

3.1.1 1-1. Virtual Environment

Lab:

```
$ python3.6 -v venv venv
$ source venv/bin/activate
(venv)$ pip install django
(venv)$ pip freeze
```

Note: $ To check installed packages.

Django==2.1.4
pytz==2018.7

3.1.2 1-2. Start Project

Lab:
Note:  To see a rocket!

3.1.3 1-3. Start App

Lab:

(venv)$ python manage.py startapp polls
*** edit mysite/urls.py
*** add polls/urls.py
*** edit polls/views.py
*** add go.py
(venv)$ . go

- mysite/urls.py:

```python
from django.contrib import admin
from django.urls import path,include

urlpatterns = [
    path('admin/', admin.site.urls),
    path('polls/', include('polls.urls')),
]
```
• polls/urls.py:

```python
from django.urls import path
from . import views

urlpatterns = [
    path('', views.index, name='index'),
]
```

• polls/views.py:

```python
from django.http import HttpResponse

def index(request):
    return HttpResponse("Hello, world. You're at the polls index.")
```

• go:

```bash
python manage.py runserver
```

**Note:** To ensure http://127.0.0.1:8000/polls/ is working.

### 127.0.0.1:8000/polls/

Hello, world. You're at the polls index.

**Warning:** Be aware http://127.0.0.1:8000 is damaged!

### 3.2 Writing your first Django app, part 2

• Django Getting started, part 2
• Writing custom django-admin commands
• Read The Docs, part 2

3.2. Writing your first Django app, part 2 9
Page not found (404)

Request Method: GET  
Request URL: http://127.0.0.1:8000/

Using the URLconf defined in mysite.urls, Django tried these URL patterns, in this order:
1. admin/
2. polls/

The empty path didn't match any of these.

You're seeing this error because you have DEBUG = True in your Django settings file. Change that to False to display a standard 404 page.

3.2.1 2-1. Admin

Lab:

(venv)$ python manage.py migrate
(venv)$ python manage.py createsuperuser
(venv)$ . go

Note: http://127.0.0.1:8000/admin/, login to maintain user/group.
3.2. Writing your first Django app, part 2
3.2.2 2-2. Model

Lab:

*** edit mysite/settings.py
*** edit poll/models.py
*** edit poll/admin.py
*** edit go
go

• mysite/settings.py:

```python
INSTALLED_APPS = [
    'polls',
    'django.contrib.admin',
    ...
]
```

• polls/models.py:

```python
from django.db import models

class Question(models.Model):
    question_text = models.CharField(max_length=200)
    pub_date = models.DateTimeField('date published')
    def __str__(self):
        return self.question_text

class Choice(models.Model):
    question = models.ForeignKey(Question, on_delete=models.CASCADE)
    choice_text = models.CharField(max_length=200)
    votes = models.IntegerField(default=0)
    def __str__(self):
        return self.choice_text
```

• polls/admin.py:

```python
from django.contrib import admin
from .models import Question, Choice
admin.site.register(Question)
admin.site.register(Choice)
```

• go.py:

```python
python manage.py makemigrations
python manage.py migrate
python manage.py runserver
```

Note: Able to maintain Question and Choice.

3.2.3 2-3. Command

Lab:
Django administration

WELCOME, ADMIN. VIEW SITE / CHANGE PASSWORD / LOG OUT

Site administration

AUTHENTICATION AND AUTHORIZATION

<table>
<thead>
<tr>
<th>Groups</th>
<th>Add</th>
<th>Change</th>
</tr>
</thead>
<tbody>
<tr>
<td>Users</td>
<td>Add</td>
<td>Change</td>
</tr>
</tbody>
</table>

POLLS

<table>
<thead>
<tr>
<th>Choices</th>
<th>Add</th>
<th>Change</th>
</tr>
</thead>
<tbody>
<tr>
<td>Questions</td>
<td>Add</td>
<td>Change</td>
</tr>
</tbody>
</table>
### 3.3 Writing your first Django app, part 3

References:
- Django Getting started, part 3
- Django Girls Tutorial, STATIC_ROOT
- Django, Django import/export

Hosting:
- Read The Docs, part 3
(venv)$ python manage.py help

Type 'manage.py help <subcommand>' for help on a specific subcommand.

Available subcommands:

[auth]
    changepassword
createsuperuser

[contenttypes]
    remove_stale_contenttypes

[django]
    check
compilemessages
createcachetable
dbsshell
diffsettings
dumpdata
flush
inspectdb
loaddata
makemessages
makemigrations
migrate
sendtestemail
shell
showmigrations
sqlflush
sqlmigrate
sqlsequencereset
squashmigrations
startapp
startproject
test
testserver

[polls]
    initpolls
3.3.1 3-1. Polls Index

Lab:

*** edit polls/views.py
*** add polls/templates/polls/index.html

• polls/views.py:

```python
from django.shortcuts import render
from .models import Question

def index(request):
    latest_question_list = Question.objects.order_by('-pub_date')[:5]
    context = {'latest_question_list': latest_question_list}
    return render(request, 'polls/index.html', context)
```

• polls/templates/polls/index.html:

```html
{% if latest_question_list %}
<ul>
{% for question in latest_question_list %}
    <li><a href="/polls/{{ question.id }}/">{{ question.question_text }}</a></li>
{% endfor %}
</ul>
{% else %}
<p>No polls are available.</p>
{% endif %}
```

Note: Show questions on our polls page.

3.3.2 3-2. Polls Detail

Lab:
Select question to change

<table>
<thead>
<tr>
<th>Action:</th>
<th>Question #12</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Question #11</td>
</tr>
<tr>
<td></td>
<td>Question #10</td>
</tr>
<tr>
<td></td>
<td>Question #9</td>
</tr>
<tr>
<td></td>
<td>Question #8</td>
</tr>
<tr>
<td></td>
<td>Question #7</td>
</tr>
<tr>
<td></td>
<td>Question #6</td>
</tr>
<tr>
<td></td>
<td>Question #5</td>
</tr>
<tr>
<td></td>
<td>Question #4</td>
</tr>
<tr>
<td></td>
<td>Question #3</td>
</tr>
<tr>
<td></td>
<td>Question #2</td>
</tr>
<tr>
<td></td>
<td><strong>Question #1</strong></td>
</tr>
</tbody>
</table>

12 questions
Select choice to change

Action:  

- [ ] CHOICE
- [ ] Choice C for Question #12
- [ ] Choice B for Question #12
- [ ] Choice A for Question #12
- [ ] Choice C for Question #11
- [ ] Choice B for Question #11
- [ ] Choice A for Question #11
- [ ] Choice C for Question #10
- [ ] Choice B for Question #10
- [ ] Choice A for Question #10
- [ ] Choice C for Question #9
- [ ] Choice B for Question #9
- [ ] Choice A for Question #9
• **Question #12**
• **Question #11**
• **Question #10**
• **Question #9**
• **Question #8**

*** edit polls/urls.py
*** edit polls/models.py
*** edit polls/views.py
*** edit polls/templates/polls/index.html
*** add polls/templates/polls/detail.html

. go

• polls/urls.py:

```python
from django.urls import path
from . import views
app_name = 'polls'

urlpatterns = [
    path('', views.index, name='index'),
    path('<int:question_id>/', views.detail, name='detail')
]
```

• polls/models.py:

```python
from django.db import models

class Question(models.Model):
    question_text = models.CharField(max_length=200)
    pub_date = models.DateTimeField('date published')
    def __str__(self):
        return self.question_text
```

(continues on next page)
class Choice(models.Model):
    question = models.ForeignKey(Question, on_delete=models.CASCADE)
    choice_text = models.CharField(max_length=200)
    votes = models.IntegerField(default=0)
    def __str__(self):
        return self.question.question_text + self.choice_text

• polls/views.py:

```python
from django.shortcuts import render, get_object_or_404
from .models import Question

def index(request):
    latest_question_list = Question.objects.order_by('-pub_date')[:5]
    context = {'latest_question_list': latest_question_list}
    return render(request, 'polls/index.html', context)

def detail(request, question_id):
    question = get_object_or_404(Question, pk=question_id)
    return render(request, 'polls/detail.html', {'question': question})
```

• polls/templates/polls/index.html:

```html
{% if latest_question_list %}
<ul>
{% for question in latest_question_list %}
    <li><a href="{{ url 'polls:detail' question.id }}">{{ question.question_text }}</a></li>
{% endfor %}
</ul>
{% else %}
<p>No polls are available.</p>
{% endif %}
```

• polls/templates/polls/detail.html:

```html
<h1>{{ question.question_text }}</h1>
<ul>
{% for choice in question.choice_set.all %}
    <li>{{ choice.choice_text }}</li>
{% endfor %}
</ul>
```

### 3.3.3 3-3. Django import / export

Lab:

```
pip install django-import-export
pip freeze
*** edit mysite/setting.py
*** edit polls/admin.py
```

* mysite/setting.py:
Question #12

- Choice A for Question #12
- Choice B for Question #12
- Choice C for Question #12

INSTALLED_APPS = (
    'import_export',
    'polls',
    ...

... 
STATIC_URL = '/static/'
# https://tutorial.djangogirls.org/en/django_start_project/
STATIC_ROOT = os.path.join(BASE_DIR, 'static')

- polls/admin.py:

```python
from django.contrib import admin
from import_export import resources
from import_export.admin import ImportExportModelAdmin
from .models import Question, Choice

class QuestionResource(resources.ModelResource):
    class Meta:
        model = Question

class QuestionAdmin(ImportExportModelAdmin):
    resource_class = QuestionResource

admin.site.register(Question, QuestionAdmin)
```

(continues on next page)
3.4 Writing your first Django app, part 4

- Django Getting started, part 4
- Read The Docs, part 4
3.4.1 4-1. Forms

Lab:

*** edit polls/urls.py
*** edit polls/views.py
*** edit polls/templates/polls/detail.html
*** add polls/templates/polls/vote.html
*** add polls/templates/polls/result.html
  go

• polls/urls.py:

```python
from django.urls import path
from . import views
app_name = 'polls'
urlpatterns = [
    path('', views.index, name='index'),
    path('<int:question_id>/', views.detail, name='detail'),
    path('<int:question_id>/vote/', views.vote, name='vote'),
    path('<int:question_id>/results/', views.results, name='results'),
]
```

• polls/views.py:

```python
from django.http import HttpResponse, HttpResponseRedirect
from django.shortcuts import get_object_or_404, render
from django.urls import reverse
from .models import Choice, Question

def index(request):
    latest_question_list = Question.objects.order_by('-pub_date')[:5]
    context = {'latest_question_list': latest_question_list}
    (continues on next page)```
```
return render(request, 'polls/index.html', context)

def detail(request, question_id):
    question = get_object_or_404(Question, pk=question_id)
    return render(request, 'polls/detail.html', {'question': question})

def vote(request, question_id):
    question = get_object_or_404(Question, pk=question_id)
    try:
        selected_choice = question.choice_set.get(pk=request.POST['choice'])
    except (KeyError, Choice.DoesNotExist):
        # Redisplay the question voting form.
        return render(request, 'polls/detail.html', {
            'question': question,
            'error_message': "You didn't select a choice."
        })
    else:
        selected_choice.votes += 1
        selected_choice.save()
        # Always return an HttpResponseRedirect after successfully dealing
        # with POST data. This prevents data from being posted twice if a
        # user hits the Back button.
        return HttpResponseRedirect(reverse('polls:results', args=(question.id,)))

def results(request, question_id):
    question = get_object_or_404(Question, pk=question_id)
    return render(request, 'polls/results.html', {'question': question})
```

- **polls/templates/polls/detail.html**:

```html
<h1>{{ question.question_text }}</h1>

{% if error_message %}<p><strong>{{ error_message }}</strong></p>{% endif %}

<form action="{% url 'polls:vote' question.id %}" method="post">
{% csrf_token %}
{% for choice in question.choice_set.all %}
   <input type="radio" name="choice" id="choice{{ forloop.counter }}" value="{{ choice.id }}">
   <label for="choice{{ forloop.counter }}">{{ choice.choice_text }}</label><br>
{% endfor %}
<input type="submit" value="Vote">
</form>
```

- **polls/templates/polls/results.html**:

```html
<h1>{{ question.question_text }}</h1>

<ul>
{% for choice in question.choice_set.all %}
   <li>{{ choice.choice_text }} -- {{ choice.votes|pluralize }}</li>
{% endfor %}
</ul>

<a href="{% url 'polls:detail' question.id %}">Vote again?</a>
```
Question #12

- Choice A for Question #12 -- 1 vote
- Choice B for Question #12 -- 0 votes
- Choice C for Question #12 -- 0 votes

Vote again?
3.5 Writing your first Django app, part 5

- Django Getting started, part 5
- Read The Docs, part 5

3.5.1 5-1. Test

Lab:

```python
*** edit mysite/urls.py
*** edit polls/tests.py
*** add polls/templates/index.html
(venv)$ python manage.py test
```

- polls/urls.py:

```python
from django.contrib import admin
from django.urls import path, include
from django.views.generic import TemplateView

urlpatterns = [
    path('admin/', admin.site.urls),
    path('polls/', include('polls.urls')),
    path('', TemplateView.as_view(template_name='index.html'), name='index'),
]
```

- polls/tests.py:

```python
from django.test import TestCase
from django.utils import timezone
from django.urls import reverse
import datetime
from .models import Question

class QuestionModelTests(TestCase):
    def test_was_published_recently_with_future_question(self):
        """
        was_published_recently() returns False for questions whose pub_date
        is in the future.
        """
        time = timezone.now() + datetime.timedelta(days=30)
        future_question = Question(pub_date=time)
        self.assertIs(future_question.was_published_recently(), False)

    def test_hello_world(self):
        response = self.client.get(reverse('index'))
        self.assertEqual(response.status_code, 200)
        self.assertContains(response, "Hello World!")
```
• polls/templates/index.html:

Hello World!

Creating test database for alias 'default'...
System check identified no issues (0 silenced).
..
Ran 2 tests in 0.015s
OK
Destroying test database for alias 'default'...

Note: Fix / with ‘Hello World!’ using CBV.

3.6 Writing your first Django app, part 6

• Django Getting started, part 6
• Django Girls Tutorials, Template extending
• Boostrap v4.2, Starter template
• Read The Docs, part 6

3.6.1 6-1. Template Extending

Lab:

```bash
*** edit polls/static/polls/style.css
*** add polls/templates/base.html
*** edit polls/templates/index.html
*** edit polls/templates/detail.html
*** edit polls/templates/results.html
```
• polls/static/polls/style.css:

```css
body{
  margin-top: 12px;
}
li a {
  color: black;
}
```

• polls/templates/base.html:

```html
{% load static %
<!doctype html>
<html lang="en">
<head>
<!-- Required meta tags -->
<meta charset="utf-8">
<meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
<!-- Bootstrap CSS -->
<link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.2.1/css/bootstrap.min.css" integrity="sha384-GJzZqFGwb1QTTN6wy59ffF1BuGJpLSa9DkKMp0DgiMDm4iYMj70gZWKYbI706tWS" crossorigin="anonymous">
<link rel="stylesheet" type="text/css" href="{% static 'polls/style.css' %}">
<title>Lab</title>
</head>
<body>
<div class="container">
  <h1><a href="/polls/">Django2.1 Tutorial Lab</a></h1>
  <div class="row">
    <div class="col-md-8">
      {% block content %}
      {% endblock %}
    </div>
  </div>
</div>
<!-- Optional JavaScript -->
<!-- jQuery first, then Popper.js, then Bootstrap JS -->
<script src="https://code.jquery.com/jquery-3.3.1.slim.min.js" integrity="sha384-q8i/X+965DzO0rT7abK41JStQIAqVgRVzpbzo5smXKp4YfRvH+8abtTE1Pi6jizo" crossorigin="anonymous"></script>
<script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.6/umd/popper.min.js" integrity="sha384-wHaIkR+jN6CkvvICOB2joaf5I4l3gm9GU6Hc1og6Ls7i6U/mkkaduKaBhlAXv9k" crossorigin="anonymous"></script>
<script src="https://stackpath.bootstrapcdn.com/bootstrap/4.2.1/js/bootstrap.min.js" integrity="sha384-B0UglyR+jN6CkvvICOB2joaf5I4l3gm9GU6Hc1og6Ls7i6U/amkDm5ga4Sd" crossorigin="anonymous"></script>
</body>
</html>
```

• polls/templates/index.html:

```html
{% extends 'polls/base.html' %}
{% block content %}
(continues on next page)
```
{%- if latest_question_list %}
<h3><ul>
{%- for question in latest_question_list %}
<li><a href="{{ url 'polls:detail' question.id }}">{{ question.question_text }}</a></li>
{%- endfor %}
</ul></h3>
{%- else %}
<p>No polls are available.</p>
{%- endif %}
{%- endblock %}

• polls/templates/detail.html:

{%- extends 'polls/base.html' %}
{% block content %}
<h3>{{ question.question_text }}</h3>
{% if error_message %}<p><strong>{{ error_message }}</strong></p>{%- endif %}
<form action="{{ url 'polls:vote' question.id }}" method="post">
{% csrf_token %}
{%- for choice in question.choice_set.all %}
    <input type="radio" name="choice" id="choice{{ forloop.counter }}" value="{{ choice.id }}">
    <label for="choice{{ forloop.counter }}">{{ choice.choice_text }}</label><br>
{%- endfor %}
<input class='btn btn-success' type="submit" value="Vote">
</form>
{%- endblock %}

• polls/templates/results.html:

{%- extends 'polls/base.html' %}
{% block content %}
<h2>{{ question.question_text }}</h2>
<ul>
{%- for choice in question.choice_set.all %}
    <li>{{ choice.choice_text }} -- {{ choice.votes|pluralize }}</li>
{%- endfor %}
</ul>
<a class='btn btn-success' href="{{ url 'polls:detail' question.id }}">Vote again?</a>
{%- endblock %}

**Warning:** You might need to ‘Clear Browsing Data’ to let css working during development.

### 3.7 Want List

Hosting:
Django2.1 Tutorial Lab

- Do you have your New Year plan?
- Did you fulfill your plan of Year 2018?
3.7.1 Why?

From Django tutorial, we learned

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

Later on we still need to create urls.py for route. And, to be production ready, we also need to have templates/polls/base.html, as well as static/polls/style.css. Why not to provide a much ready templates for startapp?

**Learn lower level python**

Something we need to guess to assume first. Then to verify and to adjust.

**My desired startapp**

References:

- Django ref, django-admin and manage.py

```python
python manage.py startapp
urls.py, templates/{{app_name}}/base.html,
```

**What's standard startapp doing?**

111

create the folder
startapp

django-admin startapp name [directory]

Creates a Django app directory structure for the given app name in the current directory or the given destination.

By default the directory created contains a `models.py` file and other app template files. (See the source for more details.) If only the app name is given, the app directory will be created in the current working directory.

If the optional destination is provided, Django will use that existing directory rather than creating a new one. You can use `'.'` to denote the current working directory.

For example:

```
django-admin startapp myapp /Users/jezdez/Code/myapp
```
copy files

replace variable with app name

To understand Django

Prepare environment to study Django:

```bash
$ pwd
/Users/pinglingchen/ksndjs/django001/django
$ ls
django
$ cd django
$ git remote -v
origin https://github.com/twoutlook/django.git (fetch)
origin https://github.com/twoutlook/django.git (push)
$ cd ..
$ ls
django
$ python3.6 -m venv venv
$ . venv/bin/activate
(venv) $ pip freeze
(venv) $ pip install -e django
Obtaining file:///Users/pinglingchen/ksndjs/django001/django/django
Collecting pytz (from Django==2.2.dev20190102231945)
  Using cached https://files.pythonhosted.org/packages/f8/0e/2365ddc010afb3d79147f1dd544e5ee24bf4e6368d502b2f464ede2982e724828628e6e8368628e724827c32157f#egg=Django
Collecting sqlparse (from Django==2.2.dev20190102231945)
  Using cached https://files.pythonhosted.org/packages/65/85/20bddd72f4537cf2c4d5d005368d502b2f464ede22982e724a828628e6e8368628e724827c32157f#egg=Django
Installing collected packages: pytz, sqlparse, Django
Running setup.py develop for Django
Successfully installed Django pytz-2018.7 sqlparse-0.2.4
You are using pip version 10.0.1, however version 18.1 is available.
You should consider upgrading via the 'pip install --upgrade pip' command.
(venv) $ pip freeze
-e git+https://github.com/twoutlook/django.
--git@b5fe97a34ea527d4254b58c2e828450e7c32157f#egg=Django
pytz==2018.7
sqlparse==0.2.4
You are using pip version 10.0.1, however version 18.1 is available.
You should consider upgrading via the 'pip install --upgrade pip' command.
(venv) $
```

python manage.py startapp having urls.py

3.7. Want List
$ ls
    django
$ cd django
$ git remote -v
   origin https://github.com/twoutlook/django.git (fetch)
   origin https://github.com/twoutlook/django.git (push)
$ cd ..
$ ls
django
$ python3.6 -m venv venv
$ . venv/bin/activate
(venv) $ pip freeze
(venv) $ pip install -e django
Obtaining file:///Users/pinglingchen/ksndjs/django001/django/django
Collecting pytz (from Django==2.2.dev20190102231945)
   Using cached https://files.pythonhosted.org/packages/f8/0e/2365ddc010af3d79147f1d
d544e5ee24bf4ece58ab99b16fbb465ce6d0/pytest-2018.7-py2.py3-none-any.whl
Collecting sqlparse (from Django==2.2.dev20190102231945)
   Using cached https://files.pythonhosted.org/packages/65/85/20bdd72f4537cf2c4d5d005
368d502b2f464ed6e22982e724a82c86268eda/sqlparse-0.2.4-py2.py3-none-any.whl
Installing collected packages: pytz, sqlparse, Django
   Running setup.py develop for Django
Successfully installed Django pytz-2018.7 sqlparse-0.2.4
You are using pip version 10.0.1, however version 18.1 is available.
You should consider upgrading via the 'pip install --upgrade pip' command.
(venv) $ pip freeze
   -e git+https://github.com/twoutlook/django.git@b5fe97a34ea527d4254b58c2e82845b7c321
57f#egg=Django
   pytz==2018.7
   sqlparse==0.2.4
You are using pip version 10.0.1, however version 18.1 is available.
You should consider upgrading via the 'pip install --upgrade pip' command.
(venv) $
(venv) $ python manage.py startapp02 app005
(venv) $ tree app005
app005
   __init__.py
   admin.py
   apps.py
   migrations
      __init__.py
   models.py
   tests.py
   urls.py
   views.py

1 directory, 8 files
(venv) $ cat app005/urls.py

# ... by Chenpingling, 2019-01-04
(venv) $