Django – Part IV

2015-05-27

SPARCS 11 undead

Greatly Inspired by SPARCS 10 hodduc

Previously on Django Seminar (Part III)

- The MVC Design Pattern

- Models
  - Handling Structured Data
  - Using Database

Today’s Topic

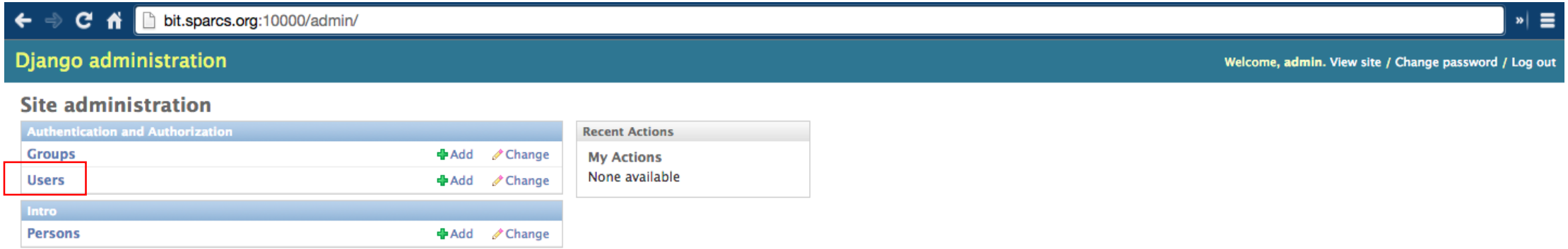
- Users

- Forms and AJAX calls

Users

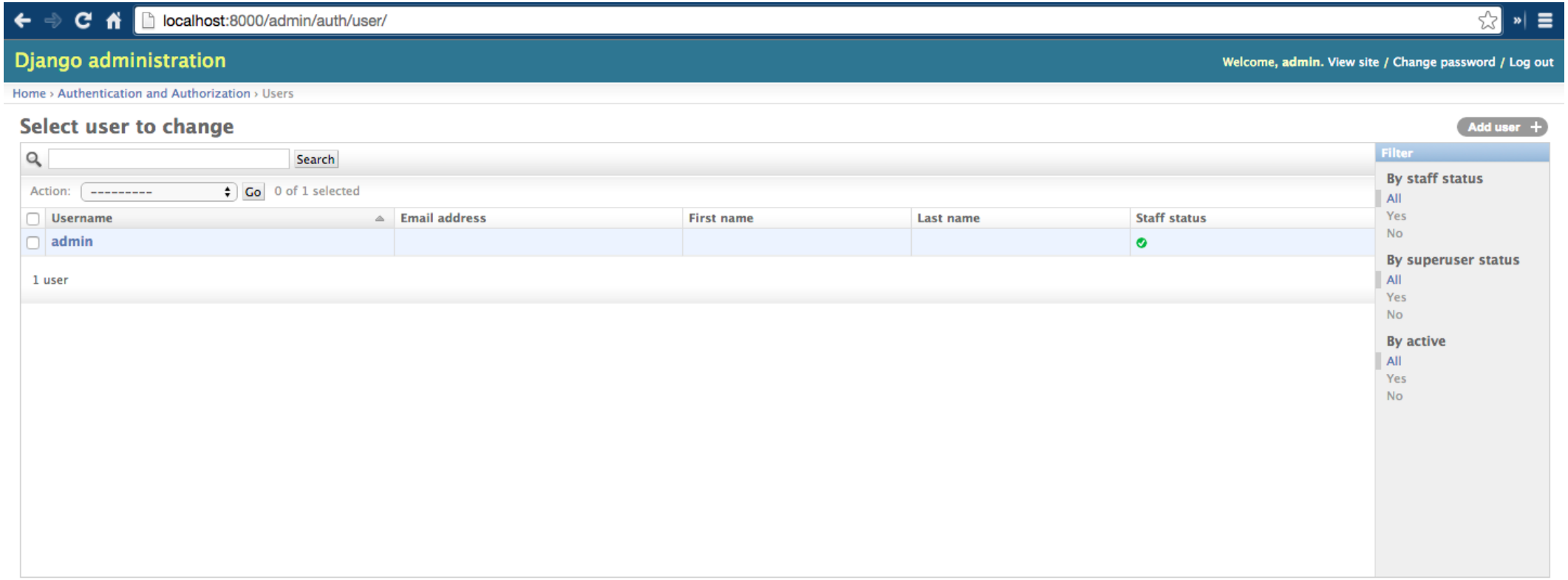
Django's default User model

Django's User Model



Django Admin

```
(env) ~/tutorial $ python manage.py createsuperuser
(env) ~/tutorial $ python manage.py runserver 0.0.0.0:10000
```



Creating Users

```
~/tutorial $ python manage.py shell
>>> from django.contrib.auth.models import User
>>> user = User.objects.create_user('undead',
'undead@sparcs.org', 'shavakan')
>>> user.save()
```

Authenticating Users

```
~/tutorial $ python manage.py shell
>>> from django.contrib.auth import authenticate
>>> authenticate(username='undead', password='undead')
>>> authenticate(username='undead', password='shavakan')
<User: undead>
```

Practice: Membership

Make memgership feature for our practice website

Create Session App

```
(env) ~/tutorial $ python manage.py startapp session
(env) ~/tutorial $ cp tutorial/urls.py session/
(env) ~/tutorial $ vi tutorial/urls.py
```

Create Session App

```
(env) ~/tutorial $ python manage.py startapp session
(env) ~/tutorial $ vi tutorial/urls.py
```

```
urlpatterns = [
    …
    url(r’^session/’, include(‘session.urls’)),
]
```

Create Session App

```
(env) ~/tutorial $ vi templates/login.html
```

```
{% extends “base.html” %}
{% block content %}
<div class=“container”>
    <form method=“POST”>
        {% csrf_token %}
        <div class=“form-group”>
            <label>USERNAME></label>
            <input type=“username” class=“form-control” name=“username” placeholder=“Username”>
        </div>
        …
```

Create Session App

```
(env) ~/tutorial $ vi templates/login.html
```

```
        …
        <div class="form-group">
            <label>PASSWORD</label>
            <input type="password" class="form-control" name="password" placeholder="Password">
        </div>
        <button type="submit" class="btn btn-default">Login</button>
    </form>
    {{ error }}
</div>
{% endblock %}
```

Create Session App

```
(env) ~/tutorial $ vi session/views.py
```

```
def login(request):
    return render(request, ‘login.html’)
```

Create Session App

```
(env) ~/tutorial $ cp tutorial/urls.py session/
(env) ~/tutorial $ vi session/urls.py
```

```
urlpatterns = [
    url(r'^admin/', include(admin.site.urls)),
    …
    url(r'^session/', include('session.urls')),
    url(r'^login/', 'session.views.user_login'),
]
```

Create Session App

Create Login Function

```
(env) ~/tutorial $ vi session/views.py
```

```
from django.shortcuts import render, redirect
from django.contrib.auth import authenticate, login

def user_login(request):
    if request.method == ‘POST’:
        username = request.POST[‘username’]
        password = request.POST[‘password’]
        user = authenticate(username=username, password=password)
    …
```

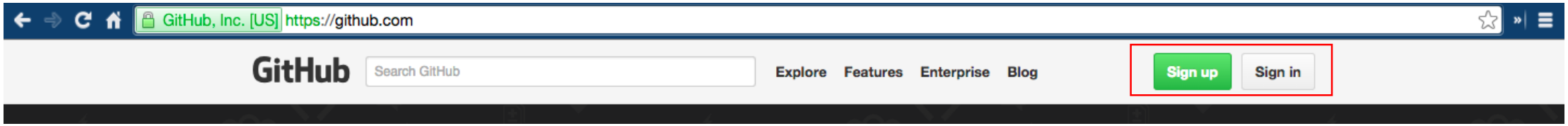
Create Login Function

```
(env) ~/tutorial $ vi session/views.py
```

```
        …
        if user is not None and user.is_active:
            login(request, user)
            return redirect('/')
        else:
            error = "Invalid login"
        return render(request, 'login.html', {'error': error})
    return render(request, 'login.html')
```

Create Login Function

Create Login Button

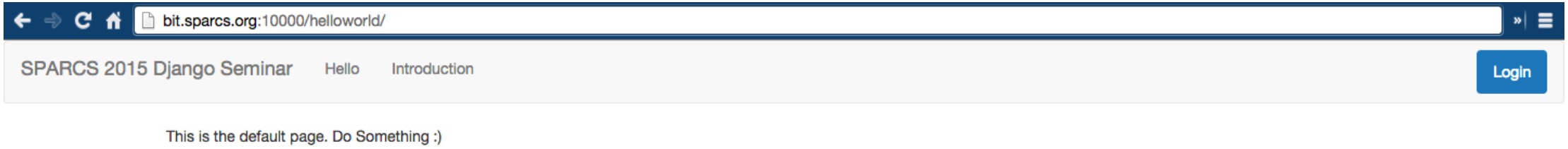


Create Login Button

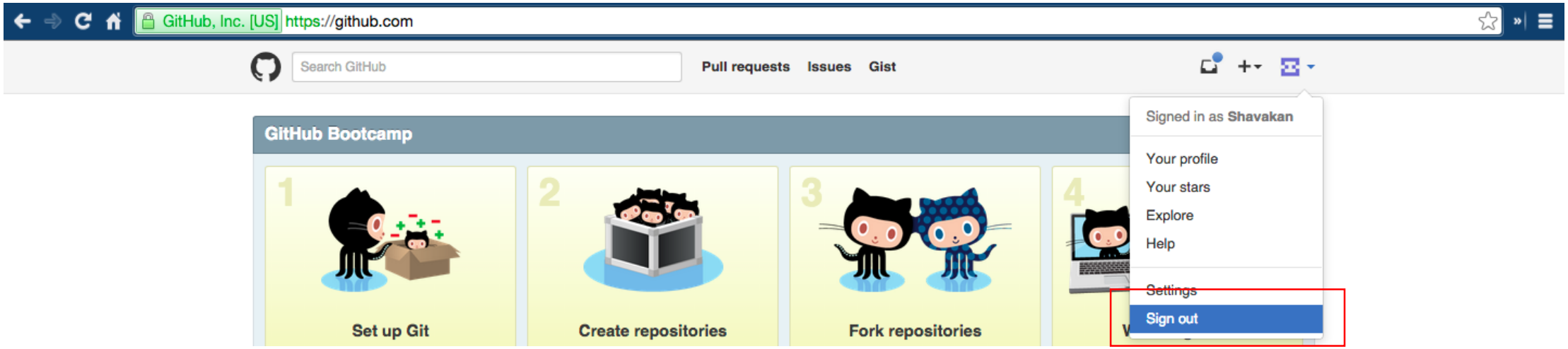
```
(env) ~/tutorial $ vi tutorial/base.html
```

```
…
<ul class=“nav navbar-nav”>
    …
</ul>
<ul class=“nav navbar-right”>
    <li><a href=“/session/login/” class=“navbar-form btn btn-primary”>Login</a>/li>
</ul>
…
```

Create Login Button



Create Logout Button



Create Logout Function

```
(env) ~/tutorial $ vi session/views.py
```

```
from django.contrib.auth import authenticate, login, logout


def user_logout(request):
    if request.user.is_authenticated():
        logout(request)
    return redirect('/')
```

Create Logout Function

```
(env) ~/tutorial $ vi session/urls.py
```

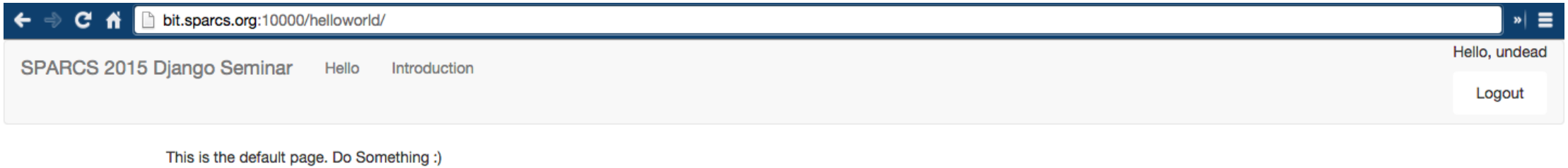
```
    …
    url(r’^login/’, ‘session.views.user_login’),
    url(r’^logout/’, ‘session.views.user_logout’),
]
```

Create Logout Button

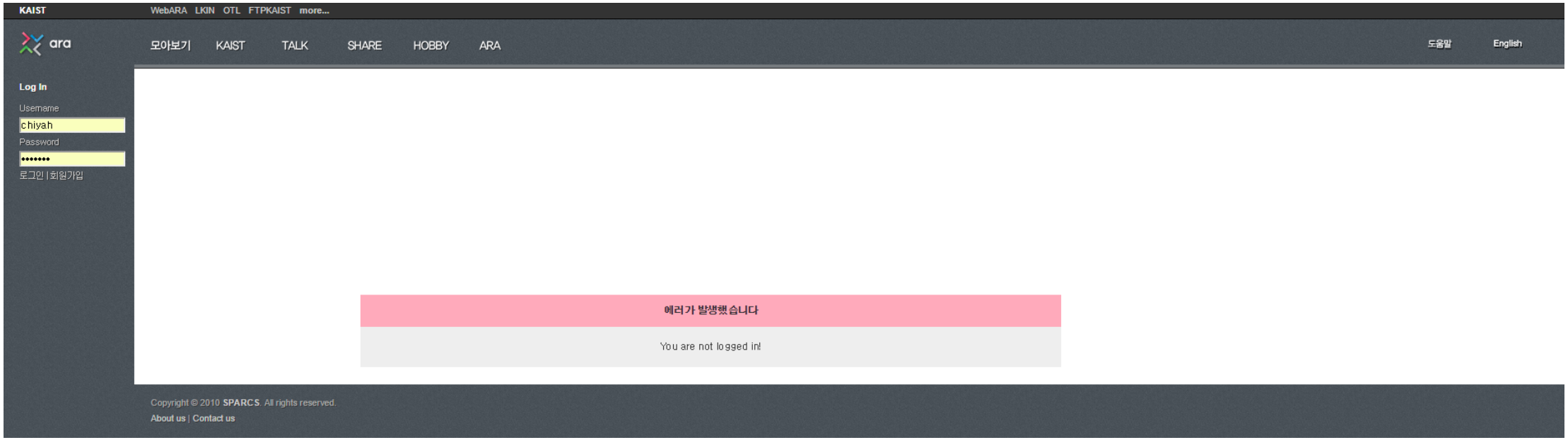
```
(env) ~/tutorial $ vi tutorial/base.html
```

```
<ul class=“nav navbar-right”>
    {% if user.is_authenticated %}
    <li>Hello, {{ user }}</span></li>
    <li><a href=“/session/logout/” class=“navbar-form btn btn-default”>Logout</a></li>
    {% else %}
    <li><a href=“/session/login/” class=“navbar-form btn btn-primary”>Login</a>/li>
    {% endif %}
</ul>
…
```

Create Logout Button



Login Required Pages



Login Required Pages

```
(env) ~/tutorial $ vi helloworld/views.py
```

```
from django.contrib.auth.decorators import login_required

@login_required(login_url='/session/login/')
def helloworld(request):
    …
@login_required(login_url='/session/login/')
def introduce(request):
    …
@login_required(…)
def me(request):
    …
```

Login Required Pages

```
(env) ~/tutorial $ vi helloworld/views.py
```

```
def user_login(request):
    …
            return redirect(‘/’request.POST[‘next’])
    …
    return render(request, ‘login.html’,
                  {‘next’: request.GET.get(‘next’, ‘/’)})
```

Login Required Pages

```
(env) ~/tutorial $ vi templates/login.html
```

```
{% extends “base.html” %}
{% block content %}
<div class=“container”>
    <form method=“POST”>
        {% csrf_token %}
        <input type=“hidden” name=“next” value=“{{ next }}”>
        <div class=“form-group”>
            <label>USERNAME></label>
            <input type=“username” class=“form-control” name=“username” placeholder=“Username”>
        </div>
        …
```

Today’s Practice

Sign Up function

Today’s Practice: Sign Up function

- 회원가입 기능을 만들어보자.
  - 로그인하지 않았을 때 Login 버튼과 Register 버튼이 동시에 보이도록
  - 로그인하였을 경우 Register 버튼은 보이지 않도록

- 회원가입을 위한 register.html, views.user_register를 작성하자
  - Username (views.py에서 중복확인)
  - Password (password field)
  - Password check (password field, Password와 같은지 확인)
  - Email (email field)
  - First Name
  - Last Name