

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

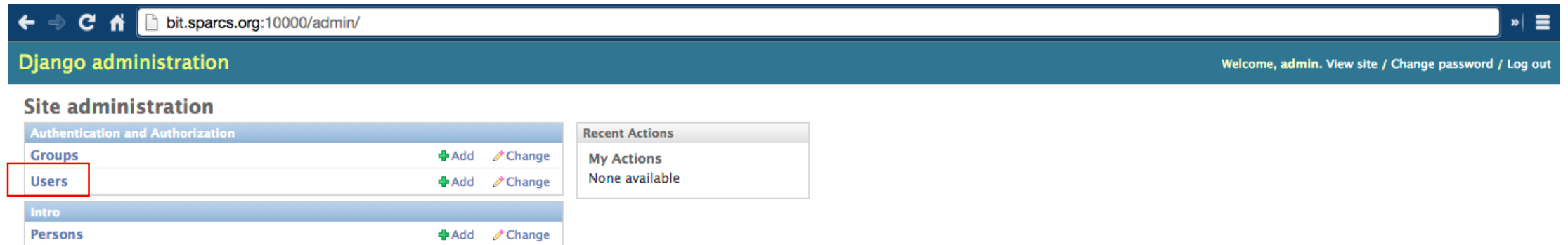
Today's Topic

- Users
- Forms and AJAX calls

Users

Django's default User model

Django's User Model



The screenshot shows the Django administration interface. At the top, the browser address bar displays `bit.sparcs.org:10000/admin/`. Below the address bar, the page title is "Django administration" and the user is logged in as "admin". The main content area is titled "Site administration" and contains a list of links for managing the site. The "Users" link is highlighted with a red box. To the right of the "Users" link, there are "Add" and "Change" buttons. Below the "Users" link, there is an "Intro" section with a "Persons" link and "Add" and "Change" buttons. To the right of the "Users" link, there is a "Recent Actions" section with a "My Actions" link and the text "None available".

← → ↻ 🏠 bit.sparcs.org:10000/admin/ » ☰

Django administration Welcome, admin. View site / Change password / Log out

Site administration

Authentication and Authorization

Groups	+ Add ✎ Change
Users	+ Add ✎ Change

Intro

Persons	+ Add ✎ Change
---------	--

Recent Actions

My Actions
None available

Django Admin

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

localhost:8000/admin/auth/user/

Django administration

Welcome, admin. View site / Change password / Log out

Home > Authentication and Authorization > Users

Select user to change Add user +

Search

Action: ----- Go 0 of 1 selected

<input type="checkbox"/>	Username	Email address	First name	Last name	Staff status
<input type="checkbox"/>	admin				✓

1 user

Filter

- By staff status
 - All
 - Yes
 - No
- By superuser status
 - All
 - Yes
 - No
- By active
 - All
 - Yes
 - No

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 membership 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>
    ...
  </form>
</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



The screenshot shows a web browser window with the address bar containing the URL `bit.sparcs.org:10000/session/login/`. The browser's navigation bar includes back, forward, refresh, and home icons. Below the address bar, a breadcrumb trail shows the path: `SPARCS 2015 Django Seminar > Hello > Introduction`. The main content area features a login form with the following elements:

- A label **USERNAME** above a text input field containing the placeholder text "Username".
- A label **PASSWORD** above a text input field containing the placeholder text "Password".
- A "Login" button located below the password field.

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



The screenshot shows a web browser window with the address bar containing `bit.sparcs.org:10000/session/login/`. The page title is "SPARCS 2015 Django Seminar" and the navigation menu includes "Hello" and "Introduction". The login form consists of two input fields: "USERNAME" with the value "undead" and "PASSWORD" with masked characters. A "Login" button is present, and below it, the text "Invalid login" is displayed, indicating a failed authentication attempt.

← → ↻ 🏠 bit.sparcs.org:10000/session/login/

SPARCS 2015 Django Seminar Hello Introduction

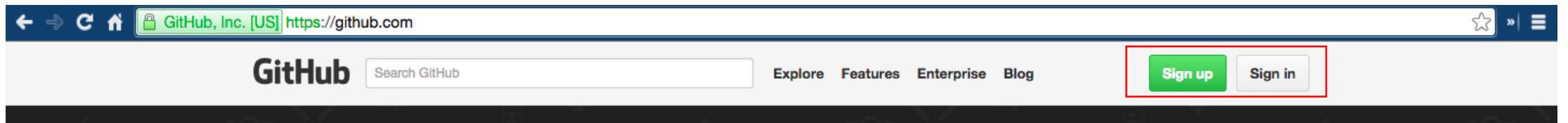
USERNAME

PASSWORD

Login

Invalid login

Create Login Button

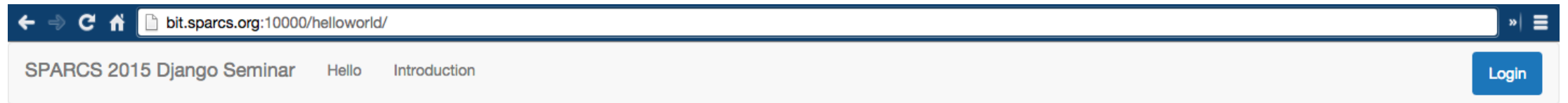


Create Login Button

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

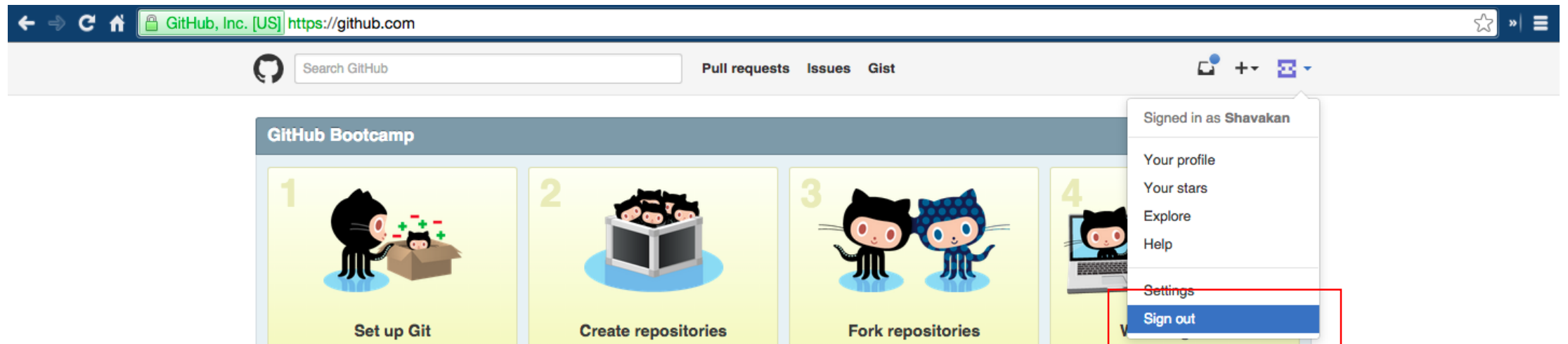
```
...  
<ul class="nav navbar-nav">  
    ...  
</ul>  
<ul class="nav navbar-right">  
    <li><a href="/session/login/" class="navbar-form btn btn-  
primary">Login</a>/li>  
</ul>  
...
```

Create Login Button



This is the default page. Do Something :)

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
```

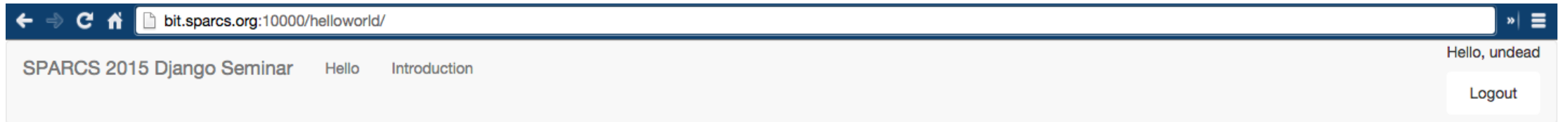
```
...  
url(r'^login/', 'session.views.user_login'),  
url(r'^logout/', 'session.views.user_logout'),  
]
```

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



This is the default page. Do Something :)

Login Required Pages

KAIST WebARA LKIN OTL FTPKAIST more...

ara 모아보기 KAIST TALK SHARE HOBBY ARA 도움말 English

Log In

Username
chivah

Password

[로그인](#) | [회원가입](#)

에러가 발생했습니다
You are not logged in!

Copyright © 2010 SPARCS. All rights reserved.
[About us](#) | [Contact us](#)

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