2-2

新建虚拟环境：cmd下输入

pip install virtualenv

virtualenv envName

cd envName/Scripts

activate（进入环境）

pip install xxx（在环境中安装新库）

deactivate（退出环境）

3-2

在setting.py中连接数据库：

DATABASES = {  
 **'default'**: {  
 **'ENGINE'**: **'sql\_server.pyodbc'**,  
 **'HOST'**:**'127.0.0.1'**,  
 **'USER'**:**'…'**,  
 **'PASSWORD'**:**'…'**,  
 **'NAME'**: **'OESDB'**,  
 **'OPTIONS'**:{  
 **'host\_is\_server'**:**True**,  
 }  
 }  
}

配置静态文件路径：

STATICFILES\_DIRS = [  
 os.path.join(BASE\_DIR, **'static'**),  
]

配置模版文件路径：

TEMPLATES = [  
 {

…  
 **'DIRS'**: [os.path.join(BASE\_DIR, **'templates'**)],  
 …  
 },  
 },  
]

4-5

自定义Userprofile：

users/models.py:

from django.contrib.auth.models import AbstractUser

class UserProfile(AbstractUser):

…

setting.py:

INSTALLED\_APP = [‘users’]

AUTH\_USER\_MODEL = “users.UserProfile”

4-9

把apps和extra\_apps文件夹标记为源文件目录

在setting.py中添加搜索源路径：

**import** sys

sys.path.insert(0,os.path.join(BASE\_DIR,**'apps'**))  
sys.path.insert(0,os.path.join(BASE\_DIR,**'extra\_apps'**))

5-1

在setting.py中改变语言：

LANGUAGE\_CODE = **'zh-hans'**

改变时区：

TIME\_ZONE = **'Asia/Shanghai'**

USE\_TZ = **False**

（该项是使数据库输入的时间定为本地时间而不是国际时间）

5-2

安装xadmin：

pip install git+git://github.com/sshwsfc/xadmin.git

注册app：

INSTALLED\_APPS = [  
 **…**  
 **'xadmin'**,  
 **'crispy\_forms'**,  
]

安装完成后要执行makemigrations和migrate新建表

5-3

配置xadmin：

在app文件夹下新建adminx.py

**from** .models **import** EmailVerifyRecord

**class** EmailVerifyRecordAdmin(object):  
 list\_display = [**'code'**,**'email'**,**'send\_type'**,**'send\_time'**]  
 search\_fields = [**'code'**,**'email'**,**'send\_type'**,**'send\_time'**]  
 list\_filter = [**'code'**,**'email'**,**'send\_type'**,**'send\_time'**]

5-5

在adminx.py中配置xadmin全局变量：

**from** xadmin **import** views

**class** BaseSetting(object):  
 enable\_themes = **True** use\_bootswatch = **True  
  
class** GlobalSettings(object):  
 site\_title = **'慕学后台管理系统'** site\_footer = **'慕学在线网'** menu\_style = **'accordion'**

xadmin.site.register(views.BaseAdminView, BaseSetting)  
xadmin.site.register(views.CommAdminView, GlobalSettings)

配置app名称：

在每个app下的apps.py输入：

**class** UsersConfig(AppConfig):  
 …verbose\_name = **'用户'**

在\_init\_.py中添加：

default\_app\_config = **'users.apps.UsersConfig'**

6-1

setting.py中添加静态文件路径：

STATICFILES\_DIRS = [  
 os.path.join(BASE\_DIR, **'static'**),  
]

6-2

在views.py中验证与登录用户：

**from** django.contrib.auth **import** authenticate, login

username = request.POST.get(**'username'**, **''**)  
password = request.POST.get(**'password'**, **''**)  
user = authenticate(username=username, password=password)  
**if** user **is not None**:  
 login(request, user)  
 **return** render(request, **'index.html'**)

6-3

自定义用户验证：

在setting.py中添加：

AUTHENTICATION\_BACKENDS = (  
 **'users.views.CustomBackend'**,  
)

在views.py中添加：

**from** django.contrib.auth.backends **import** ModelBackend  
**from** django.db.models **import** Q

**class** CustomBackend(ModelBackend):  
 **def** authenticate(self, request, username=**None**, password=**None**, \*\*kwargs):  
 **try**:  
 user = UserProfile.objects.get(Q(username=username)|Q(email=username))  
 **if** user.check\_password(password):  
 **return** user  
 **except** Exception **as** e:  
 **return None**

6-4

用类的形式写views：

**from** django.views.generic.base **import** View

**class** LoginView(View):  
 **def** get(self,request):  
 **…**  
 **def** post(self,request):  
 …

urls.py中修改：

**from** users **import** views

urlpatterns = [  
 …  
 url(**'^login/$'**, views.LoginView.as\_view(), **'login'**),  
]

创建forms.py

**from** django **import** forms  
**class** LoginForm(forms.Form):  
 username = forms.CharField(required=**True**)  
 password = forms.CharField(required=**True**, min\_length=6, max\_length=12)

修改views.py：

**from** .forms **import** LoginForm

**class** LoginView(View):  
 **def** get(self,request):  
 …  
 **def** post(self,request):  
 login\_form = LoginForm(request.POST)  
 **if** login\_form.is\_valid():  
 …

**else**:  
 **return** render(request, **'index.html'**,{**'login\_form'**: login\_form})

6-5

通过form传递错误提示：

template中：

{% **for key**,**error in login\_form**.**errors**.**items** %}

{{ **error** }}

{% **endfor** %}

6-7

添加验证码：

pip install  django-simple-captcha

把’captcha’添加到INSTALLED\_APPS中

在urls.py中添加 url(**'^captcha/'**, include(**'captcha.urls'**)),

通过makemigrations和migrate生成表

在表单中添加验证码项：

**from** captcha.fields **import** CaptchaField  
**class** RegisterForm(forms.Form):  
 …  
 captcha = CaptchaField()

6-8

在网页上添加验证码图片：

在views.py中添加：

**from** .forms **import** RegisterForm

**class** RegisterView(View):  
 **def** get(self, request):  
 register\_form = RegisterForm()  
 **return** render(request, **'UserPage/registPage1.html'**, {**'register\_form'**:register\_form,})

在模版中调用：

{{ **register\_form**.**captcha** }}

从网页中获取form表单：

**class** RegisterView(View):  
 **…**  
 **def** post(self, request):  
 register\_form = RegisterForm(request.POST)  
 **if** register\_form.is\_valid():  
 **…**

创建保存新数据：

**from** django.contrib.auth.hashers **import** make\_password

**class** RegisterView(View):  
 **…**  
 **def** post(self, request):  
 …  
 **if** register\_form.is\_valid():  
 username = request.POST.get(**'username'**, **''**)  
 password = request.POST.get(**'password'**, **''**)  
 user\_profile = UserProfile()  
 user\_profile.username = username  
 user\_profile.email = username  
 user\_profile.password = make\_password(password=password)  
 user\_profile.save()

6-8

在apps下新建python package：utils

在utils下新建email\_send.py

生成随机激活码：

**from** random **import** Random

**def** random\_str(randomlength = 8):  
 str = **''** chars = **'AaBbCcDdEeFfGgHhIiJjKkLlMmNnOoPpQqRrSsTtUuVvWwXxYyZz0123456789'** length = len(chars)-1  
 random = Random()  
 **for** i **in** range(randomlength):  
 str += chars[random.randint(0, length)]  
 **return** str

在setting.py中配置邮箱发送：

EMAIL\_HOST = **'smtp.qq.com'**EMAIL\_PORT = 25  
EMAIL\_HOST\_USER = **'xxx@qq.com'**EMAIL\_HOST\_PASSWORD = **'admin123'**EMAIL\_USE\_TLS = **False**EMAIL\_FROM = **'xxx@qq.com'**

继续在email\_send.py中编写发送邮箱代码：

**from** django.core.mail **import** send\_mail  
**from** users.models **import** EmailVerifyRecord  
**from** OnlineEducationSite.settings **import** EMAIL\_FROM

**def** send\_register\_email(email, send\_type=**'register'**):  
 email\_record = EmailVerifyRecord()  
 code = random\_str(16)  
 email\_record.code = code  
 email\_record.email = email  
 email\_record.send\_type = send\_type  
 email\_record.save()  
  
 email\_title = **''** email\_body = **''  
  
 if** send\_type == **'register'**:  
 email\_title = **'在线教育网站注册用户激活链接'** email\_body = **'请点击下面链接激活用户：http://127.0.0.1:8000/active/{0}'**.format(code)  
  
 send\_status = send\_mail(email\_title, email\_body, EMAIL\_FROM, [email])  
 **if** send\_status:  
 **pass**

6-10

创建用户时修改用户为未激活：

**if** register\_form.is\_valid():  
 …  
 user\_profile.is\_active = **False** user\_profile.save()

编写链接的view：

**class** ActiveUserView(View):  
 **def** get(self,request,active\_code):  
 **pass**

配置点击的链接：

urlpatterns = [  
…

url(**'^active/(?P<active\_code>.\*)/$'**,userViews.ActiveUserView.as\_view(), name=**'user\_active'**),  
…  
]

修改登陆时对激活的判断：

**if** user.is\_active:  
 login(request, user)  
 **…**  
**else**:  
 **return** render(request, **'UserPage/loginPage.html'**, {**'msg'**: **'用户未激活！'**})

6-11

添加错误链接时的页面activeFailPage.html

添加找回密码页面，找回密码form与登陆form类似，view和发邮件也相似：

url(**'^forget/$'**, userViews.ForgetPwdView.as\_view(), name=**'forget'**),

**class** ForgetForm(forms.Form):  
 username = forms.CharField(required=**True**,max\_length=10)  
 email = forms.EmailField(required=**True**)  
 captcha = CaptchaField(error\_messages={**'invalid'**:**'验证码错误！'**})

**class** ForgetPwdView(View):  
 **def** get(self,request):  
 forget\_form = ForgetForm()  
 **return** render(request, **'UserPage/forgetPwdPage.html'**, {**'forget\_form'**:forget\_form,})  
 **def** post(self,request):  
 forget\_form = ForgetForm(request.POST)  
 **if** forget\_form.is\_valid():  
 email = request.POST.get(**'email'**, **''**)  
 send\_user\_email(email, **'forget'**)  
 **return** render(request, **'UserPage/sendSuccessPage.html'**)  
 **else**:  
 **return** render(request, **'UserPage/forgetPwdPage.html'**, {**'forget\_form'**: forget\_form, })

**if** send\_type == **'register'**:  
 … **elif** send\_type == **'forget'**:  
 email\_title = **'在线教育网站重置密码链接'** email\_body = **'请点击下面链接重置密码：http://127.0.0.1:8000/reset/{0}'**.format(code)  
  
 send\_status = send\_mail(email\_title, email\_body, EMAIL\_FROM, [email])  
 **if** send\_status:  
 **pass**

7-1

模版继承：

{% block blockName %} {% endblock %}

{% extends ‘master\_page.html’ %}

{% block blockName %} xxx {% endblock %}

7-3

设置上传图片路径：

在根目录创建media文件夹

在setting.py中添加：

MEDIA\_URL = **'/media/'**MEDIA\_ROOT = os.path.join(BASE\_DIR, **'media'**)

在urls.py中配置：

**from** django.views.static **import** serve

**from** OnlineEducationSite.settings **import** MEDIA\_ROOT

urlpatterns = [  
 …  
 url(**r'^media/(?P<path>.\*)$'**, serve, {**"document\_root"**:MEDIA\_ROOT}),  
 …  
]

7-5

分页 github - pure pagination

获取网址上的参数：

www.baidu.com/?para1=value1&para2=value2

value = request.Get.get(‘paraName’, ‘default’)

重看！！！！

7-6

对比：

{% ifequal a b %}{% endifequal %}

类型转换：

int|stringformat:’I’

重看！！！！

7-7

**class** UserForm(forms.ModelForm):  
 my\_filed = forms.CharField()  
 **class** Meta:  
 model = UserAsk  
 field = [**'name'**, **'phone'**, **'course\_name'**]

urls分发：

*# urls.py*url(**'^org/'**, include(**'organization.urls'**, namespace=**'org'**)),

*# organization/urls.py*urlpatterns = [  
 url(**'^list/$'**, orgViews.OrgView.as\_view(), name=**'list'**),  
 url(**'^add\_ask/$'**, orgViews.AddUserAskView.as\_view(), name=**'add\_ask'**),  
]

7-8

异步请求：

**from** django.http **import** HttpResponse

**return** HttpResponse(**"{'status':'success'}"**,content\_type=**'application/json'**)

自定义form检测：

**def** clean\_phone(self):  
 phone = self.cleaned\_data[**'phone'**]  
 REGEX\_MOBILE = **"^1[358]\d{9}$|^147\d{8}$|^176\d{8}$"** p = re.compile(REGEX\_MOBILE)  
 **if** p.match(phone):  
 **return** phone  
 **else**:  
 **raise** forms.ValidationError(**'手机号码非法！'**, code=**'phone\_invalid'**)

7-12

判断当前是否有用户登录：

**if** request.user.is\_authenticated():

ajax代码

8-1

在模版中显示model中choise类型的展示项：

degree = models.CharField(choices=((**'cj'**,**'初级'**),(**'zj'**,**'中级'**),(**'gj'**,**'高级'**))…)

{{ **get\_degree\_display** }}

8-7

数据库数据获取：

user\_ids = [user\_course.user.id for user\_course in user\_courses]

all\_user\_courses = UserCourse.objects.filter(user\_id\_\_in=user\_ids)

需要预先登陆的网页：

先在utils下新建mixin\_utils.py

**from** django.contrib.auth.decorators **import** login\_required  
**from** django.utils.decorators **import** method\_decorator  
  
**class** LoginRequiredMixin(object):  
 @method\_decorator(login\_required(login\_url=**'/login/'**))  
 **def** dispatch(self, request, \*args, \*\*kwargs):  
 **return** super(LoginRequiredMixin, self).dispatch(request, \*args, \*\*kwargs)

在写View时继承该类。

8-8

视频播放器：video-js

视频上传网站：七牛云

10-1

模版字符串分割：string|slice:’5’

10-2

07:08 JS代码

10-4

图片上传保存。

<**form method="post" action=".." enctype="multipart/form-data"**>

<**input name="image" type="file" value="上传头像"**>

</**form**>

class UploadImageView(LoginRequiredMixin, View):

def post(self, request):

image\_form = UploadImageForm(request.POST, request.FILES, instance = request.user)

if image\_form.is\_valid():

image\_form.save()

class UploadImageForm(forms.ModelForm):

class Meta:

model = UserProfile

fields = [‘image’,]

11-1

登出：

**from** django.contrib.auth **import** logout

**from** django.core.urlresolvers **import** reverse

**class** LogoutView(View):  
 **def** get(self,request):  
 logout(request)   
 **return** HttpResponseRedirect(reverse(**'index'**))

11-4

定义404和500页面：

*# urls.py*

handler404 = **'users.views.page\_not\_found'**

handler500 = **'users.views.page\_error'**

*# views.py***def** page\_not\_found(request):  
 **from** django.shortcuts **import** render\_to\_response  
 response = render\_to\_response(**'404.html'**,{})  
 response.status\_code = 404  
 **return** response

**def** page\_error(request):  
 **from** django.shortcuts **import** render\_to\_response  
 response = render\_to\_response(**'500.html'**,{})  
 response.status\_code = 500  
 **return** response

13-3

xadmin的icon：font-awesome

xadmin显示顺序、只读、隐藏：

ordering = [**'-code'**]  
readonly\_fields = [**'code'**]  
exclude = [**'email'**]

外键搜索：

relfield\_style = **'fk-ajax'**

外键新建：

**class** LessonInline(object):  
 model = Lesson  
 extra = 0

**class** CourseAdmin(object):

inlines = [LessonInline]

同一个model注册两个admin：

先新建一个代理model：

**class** BannerCourse(Course):  
 **class** Meta:  
 verbose\_name = **'轮播图课程'** verbose\_name\_plural = verbose\_name  
 proxy = **True**

用新admin去绑定注册新model：

**class** BannerCourseAdmin(object):  
 list\_display = [**…**]  
 search\_fields = [**…**]  
 list\_filter = [**…**]  
  
 **def** queryset(self):  
 qs = super(BannerCourseAdmin, self).queryset()  
 qs = qs.filter(is\_banner=**True**)  
 **return** qs

xadmin.site.register(BannerCourse, BannerCourseAdmin)

13-5

在列表中修改：

list\_editable = [**'degree'**]

13-7

ueditor

<https://github.com/zhangfisher/DjangoUeditor>

**from** DjangoUeditor.models **import** UEditorField

detail = UEditorField(verbose\_name=**'课程详情'**,width=600, height=300, imagePath=**"courses/ueditor/"**, filePath=**"courses/ueditor/"**,default=**''**)

*# adminx.py*

style\_fields = {**'detail'**: **'ueditor'**}

xadmin中ManyToManyField字段的显示：

在adminx.py中添加：

style\_fields = {**'A'**: **'m2m\_transfer'**,}

其中A是ManyToManyField字段。

13-8

excel插件

在extra\_apps/xadmin/plugins中添加excel.py:

**import** xadmin

**from** xadmin.views **import** BaseAdminPlugin, ListAdminView

**from** django.template **import** loader

**from** xadmin.plugins.utils **import** get\_context\_dict

*#excel 导入*

**class** ListImportExcelPlugin(BaseAdminPlugin):

import\_excel = **False**

**def** init\_request(self, \*args, \*\*kwargs):

**return** bool(self.import\_excel)

**def** block\_top\_toolbar(self, context, nodes):

nodes.append(loader.render\_to\_string(**'xadmin/excel/model\_list.top\_toolbar.import.html'**, context=get\_context\_dict(context)))

xadmin.site.register\_plugin(ListImportExcelPlugin, ListAdminView)

在\_\_init\_\_.py中添加’excel’

在需要excel批量上传的表对应app/adminx.py中对应admin类添加：

import\_excel = **True**

**def** post(self, request, \*args, \*\*kwargs):

**if 'excel' in** request.FILES:

data = xlrd.open\_workbook(file\_contents=request.FILES.get(**'excel'**).read())

table = data.sheets()[0]

nrows = table.nrows

**for** i **in** range(1,nrows):

…

**return** super(QuestionAdmin, self).post(request, args, kwargs)