哈爾濱工業大學
HARBIN INSTITUTE OF TECHNOLOGY

Lab 3：MVC编程与云平台部署

实验目标

- 掌握MVC架构下SaaS开发的基本流程；
- 开发环境：Python+Django+SAE

- 在Python编程环境中搭建Django开发环境，开发一个小型SaaS，在Web页面中对数据库中的数据进行CRUD(增加Create、读取Retrieve、更新Update和删除Delete)操作；
- 部署在SAE上并对外发布，模拟用户访问。

- 本实验由个人独立完成

开发要求

- **使用MySQL或SQLite建立一个“图书数据库BookDB”，包含两张表：**
  - Book {ISBN (PK), Title, AuthorID (FK), Publisher, PublishDate, Price}
  - Author {AuthorID (PK), Name, Age, Country}
- **手工输入足够多的若干测试数据；**
- **开发以下功能：**
  - 输入作者名字，查询该作者所著的全部图书的题目；
  - 当用户点击某本图书的题目时，展示图书详细信息和作者详细信息；
  - 当用户点击“删除”按钮时，将对应行的图书从数据表中删除；
  - (opt) 用户可新增一本图书，若该书作者不在库中，还需增加新作者；
  - (opt) 用户可更新一本图书的作者、出版社、出版日期、价格。

Django

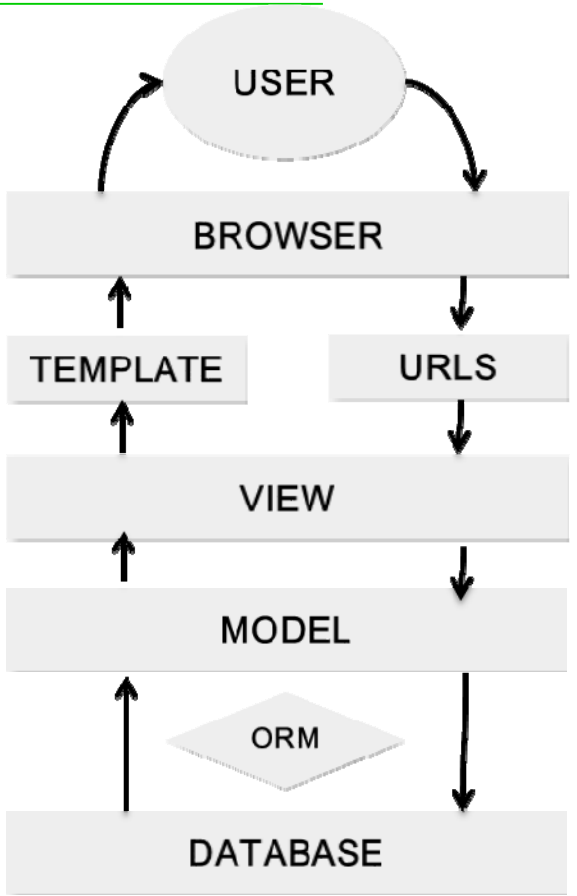
- **Django是一个开放源代码的Web应用框架，由Python写成。**
  https://www.djangoproject.com/ Django官方网站
- **参考文献：**
  - 《The Django Book 2.0》中文译本　CMS网站上
  - 《Django Web开发指南》　CMS网站上

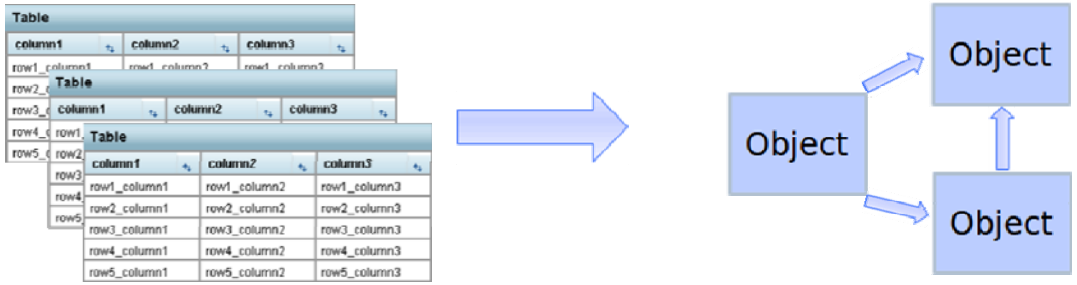
- **Django 安装**
  - 安装Django　“pip install Django==X.X.X”（X.X.X为版本号，如：1.7.10)
  - 验证是否安装成功
    - >>> import django
    - >>> django.VERSION

Django的MTV

- **M = Model**
  - 定义数据结构和行为，负责数据库管理
  - 使用ORM技术
  - models.py文件

```
class Issue(models.Model):
    dms_id = models.CharField(max_length=200)
    title = models.CharField(max_length=500)
    submitter = models.CharField(max_length=100)
    rft_status = models.CharField(max_length=100, blank=True, null=True, choices=choices.MODEL_CHOICES_STATUS,
    rft_owner = models.CharField(max_length=200, blank=True, null=True)
    rft_result = models.CharField(max_length=100, blank=True, null=True, choices=choices.MODEL_CHOICES_RESULT,

    class Meta:
        app_label = "rftissue"
        abstract = True
        # ordering = ['dms_id']

    def __unicode__(self):
        return self.dms_id

    def get_basic_status(self):

    def get_specified_status(self):
```

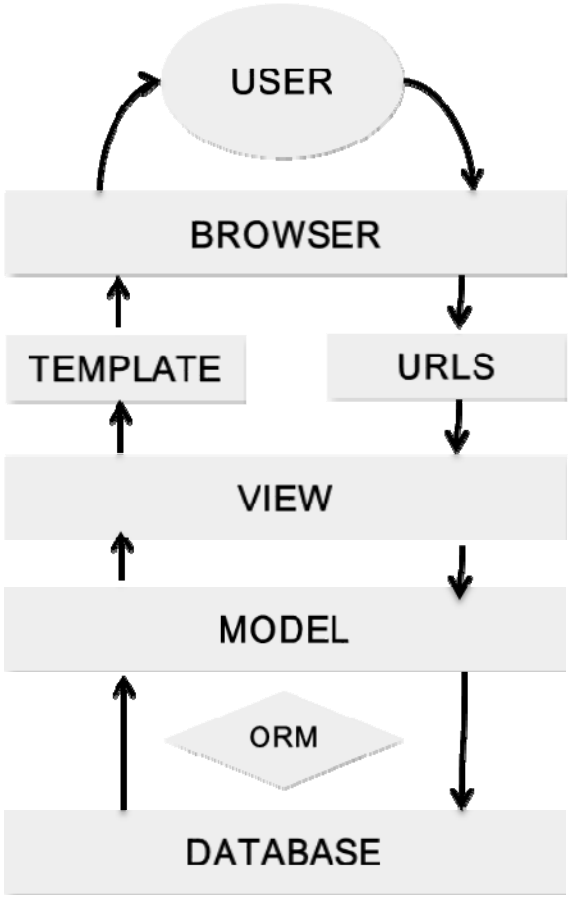




Django的MTV

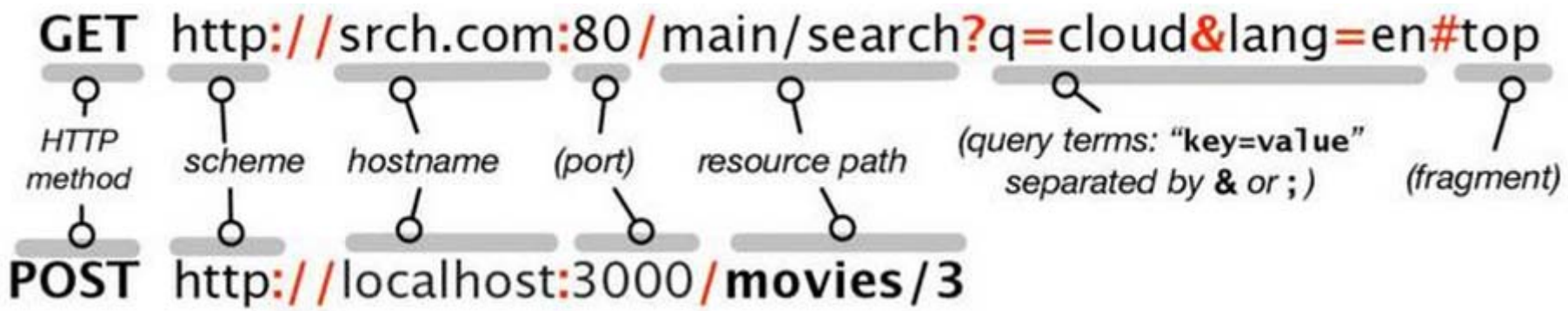
- **V = View**
  - 用于控制要显示什么数据
  - 返回HTTP 响应
  - views.py文件

```
@ensure_csrf_cookie
@login_required(login_url='/accounts/signin/')
def home(request):
    '''
    Display home page.
    '''
    # User must be logged, otherwise the user will redirected to signin page
    user = "Guest"
    if request.user.is_authenticated():
        user = request.user.username
    return render(request, 'rft/home.html', {'user': user})


@login_required(login_url='/accounts/signin/')
def display_acm_rft(request, export='none'):


@login_required(login_url='/accounts/signin/')
def display_sw_rft(request, export='none'):
```



前端请求的URI（Uniform Resource Identifier）

- 用户通过浏览器发出的请求，通过**HTTP**协议传递到服务器端。
- 具体形式即为**URI**



- **HTTP**是无状态协议，通过浏览器端的**cookies**保持同一用户的多次请求，通过服务器端的**session**保持与用户的连接。

Django的MTV

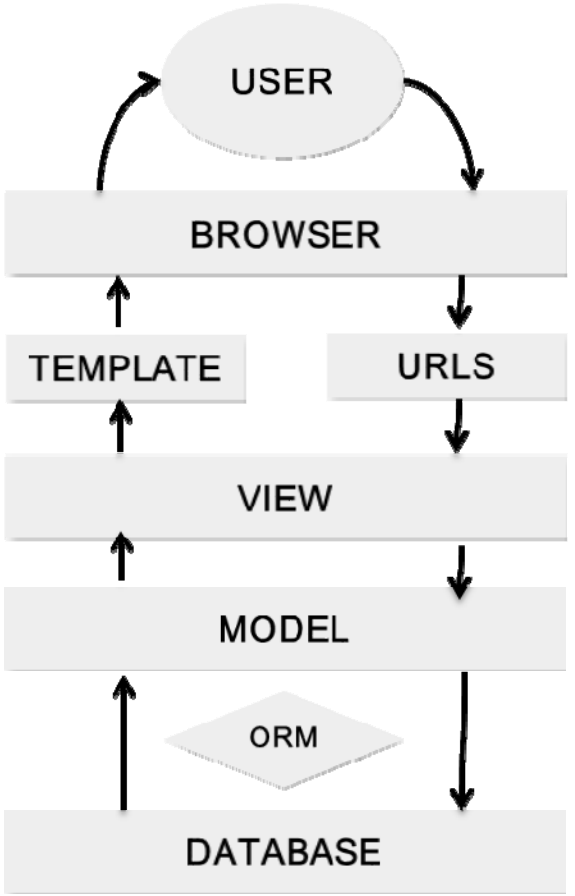
- **urls.py is a mapping between URL patterns to Python functions (Views)**

```
urlpatterns = patterns('',
                       url(r'^$', views.home),
                       url(r'^acm/$', views.display_acm_rft),
                       url(r'^acm/csv$', views.display_acm_rft, {'export': 'csv'}),
                       url(r'^sw/$', views.display_sw_rft),
                       url(r'^sw/csv$', views.display_sw_rft, {'export': 'csv'}),
                       url(r'^acm/(?P<pk>\d+)/$', views.edit_issue, {'issue_type': 'acm'}),
                       url(r'^sw/(?P<pk>\d+)/$', views.edit_issue, {'issue_type': 'sw'}),
                       url(r'^export/csv/$', views.export_to_csv),
                       url(r'^about/$', views.about),
                       url(r'^doc/$', views.doc),
                       url(r'^doc/guide/$', views.user_guide),
                       url(r'^send_email_notification/$', views.send_email_notification),
                       url(r'^api/acm/$', views.ACMList.as_view()),
                       url(r'^api/acm/(?P<pk>[0-9]+)/$', views.ACMDetail.as_view()),
                       url(r'^api/sw/$', views.SWList.as_view()),
                       url(r'^api/sw/(?P<pk>[0-9]+)/$', views.SWDetail.as_view()),
)
```

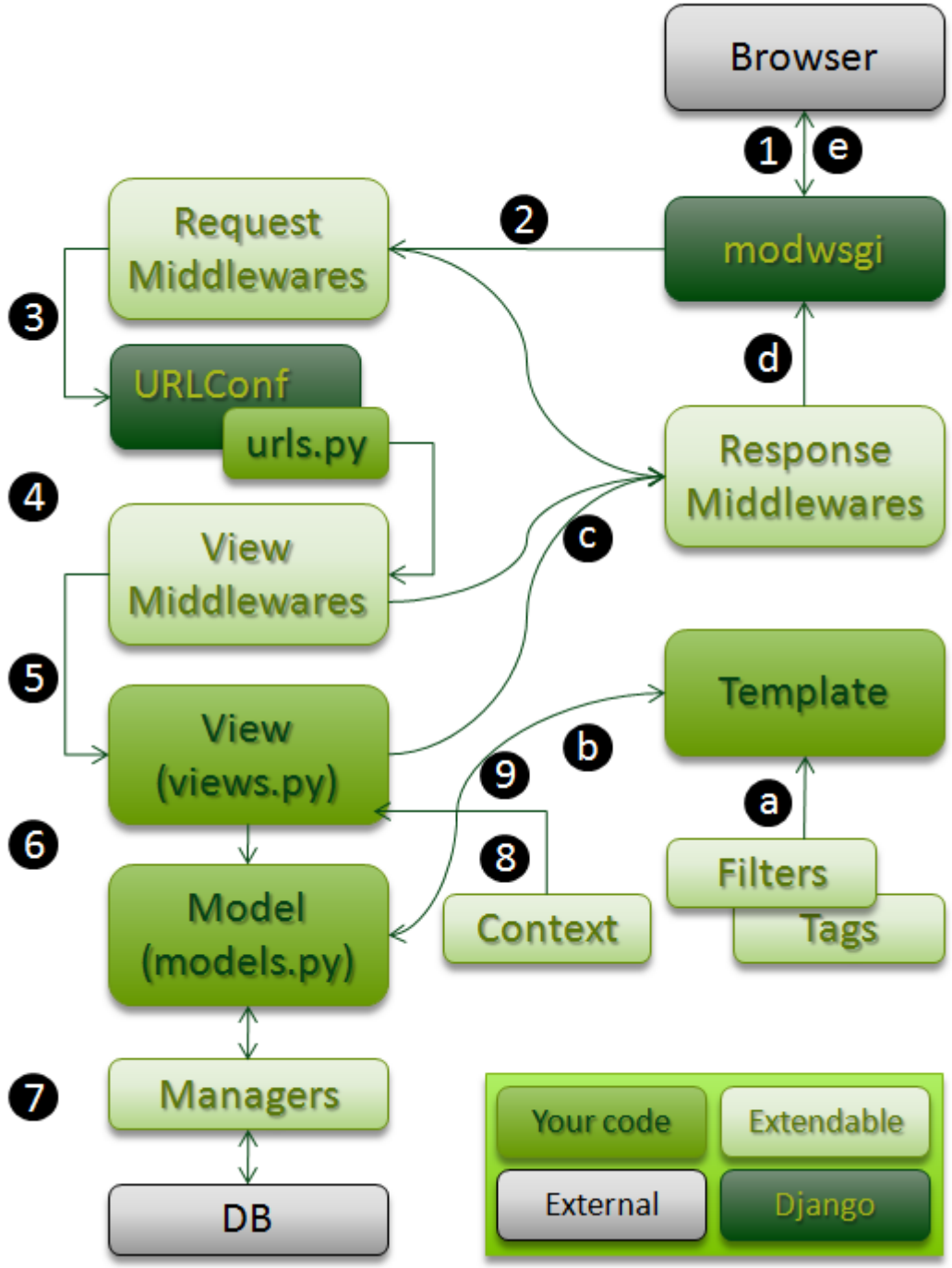
Django的MTV

- **T = Template**
  - 负责怎样显示数据
  - 利用格式化的html文件，使数据按照要求显示
  - 强大而可扩展的模板语言

```
{% block title %}{% trans "Signin" %}{% endblock %}

{% block content %}
    <form action="" method="post">
        {% csrf_token %}
        <fieldset>
            <legend><strong>{% trans "Welcome to CV RFT Issues Management Site, please log in:" %}</strong></legend>
            {{ form.non_field_errors }}
            {% for field in form %}
                {{ field.errors }}
                {% comment %} Displaying checkboxes differently {% endcomment %}
                {% if field.name == 'remember_me' %}
                    <p class="checkbox">
                        <label for="id_{{ field.name }}">{{ field }} {{ field.label }}</label>
                    </p>
                {% else %}
                    <p>
                        {{ field.label_tag }}
                        {{ field }}
                    </p>
                {% endif %}
            {% endfor %}
        </fieldset>

        <input type="submit" value="{% trans "Signin" %}"/>

        <p class="forgot-password"><a href="{% url 'userena_password_reset' %}"
                                      title="{% trans 'Forgot your password?' %}">{% trans "Forgot your password?" %}</a>
        </p>
        {% if next %}<input type="hidden" name="next" value="{{ next }}"/>{% endif %}
    </form>
{% endblock %}
```



Django调用顺序


Browser
1
e
Request Middlewares
2
modwsgi
3
URLConf
urls.py
d
Response Middlewares
4
View Middlewares
c
5
View (views.py)
Template
b
9
a
8
Filters
Context
Tags
6
Model (models.py)
7
Managers
DB
Your code
Extendable
External
Django

MTV vs MVC

传统MVC	Django的MVC（MTV）
M：封装业务逻辑/数据；将数据变化通知视图	M：数据存取
V： 负责把数据格式化后呈现给用户	V：Django将MVC中的视图进一步分解为 Django视图V和 Django模板T两个部分，分别决定 “展现哪些数据”和 “如何展现”，使得Django的模板可以根据需要随时替换，而不仅仅限制于内置的模板
C：将用户行为映射到模型的更新上；选择需要显示的视图	C：由Django框架的URLconf来实现

Settings.py

```
INSTALLED_APPS = (
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
)

MIDDLEWARE_CLASSES = (
    'django.contrib.sessions.middleware.SessionMiddleware',
    'django.middleware.common.CommonMiddleware',
    'django.middleware.csrf.CsrfViewMiddleware',
    'django.contrib.auth.middleware.AuthenticationMiddleware',
    'django.contrib.messages.middleware.MessageMiddleware',
    'django.middleware.clickjacking.XFrameOptionsMiddleware',
)

ROOT_URLCONF = 'demoproject.urls'

WSGI_APPLICATION = 'demoproject.wsgi.application'


# Database
# https://docs.djangoproject.com/en/1.6/ref/settings/#databases

DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.sqlite3',
        'NAME': os.path.join(BASE_DIR, 'db.sqlite3'),
    }
}
```

- **INSTALLED_APPS: Enable the Django applications. App names MUST be unique.**
- **MIDDLEWARE_CLASSES: Enable the middleware components. The order in MIDDLEWARE_CLASSES matters.**
- **ROOT_URLCONF: Represent the full Python path to root URLconf.**
- **WSGI_APPLICATION: The full Python path of WSGI app object that built-in application will use.**
- **DATABASES: Settings for all databases to be used with Django.**

环境的安装配置

- **Step1：安装Python环境，注意版本和32/64位问题；本机编程时可根据本机OS类型选择，SAE只支持32位Python。**
- **Step2：安装Django，“pip install Django==X.X.X”(X.X.X为版本号，如：1.7.10）**
  - 验证是否安装成功
    - >>> import django
    - >>> django.VERSION
- **Step3：安装MySQL数据库(注意32/64位问题，同本机OS一致）；如使用SQLite，Python内置，无需安装**

安装配置

- **Step4：安装数据库驱动，以MySQL为例，MySQL数据库目前存在两种驱动，安装其一即可：**
  - mysql-connector-python：是MySQL官方的纯Python驱动，http://dev.mysql.com/downloads/file.php?id=458915 下载时注意Python版本和32/64位的问题
  - MySQL-python：是封装了MySQL C驱动的Python驱动，https://pypi.python.org/pypi/MySQL-python
  - SQLite驱动Python内置，无需安装.

- **Step5：生成Django项目 “Django-admin startproject XXX”（XXX项目名字）**
- **Step6： 启动内置服务器 “python manage.py runserver”，**

  浏览器运行：http://127.0.0.1:8000/ XXX/YYY

关于数据库

- 方法1：数据库操作可以使用Django的ORM映射机制，即使用Model层，适合对SQL语言不熟悉的同学，详见《The Django Book 2.0》中的说明。

- 方法2：直接通过SQL语句操作，适合可熟练使用SQL语言的同学，SQL语句的简单说明见CMS课程网站上的《SQL语句简单说明》，代码示例：

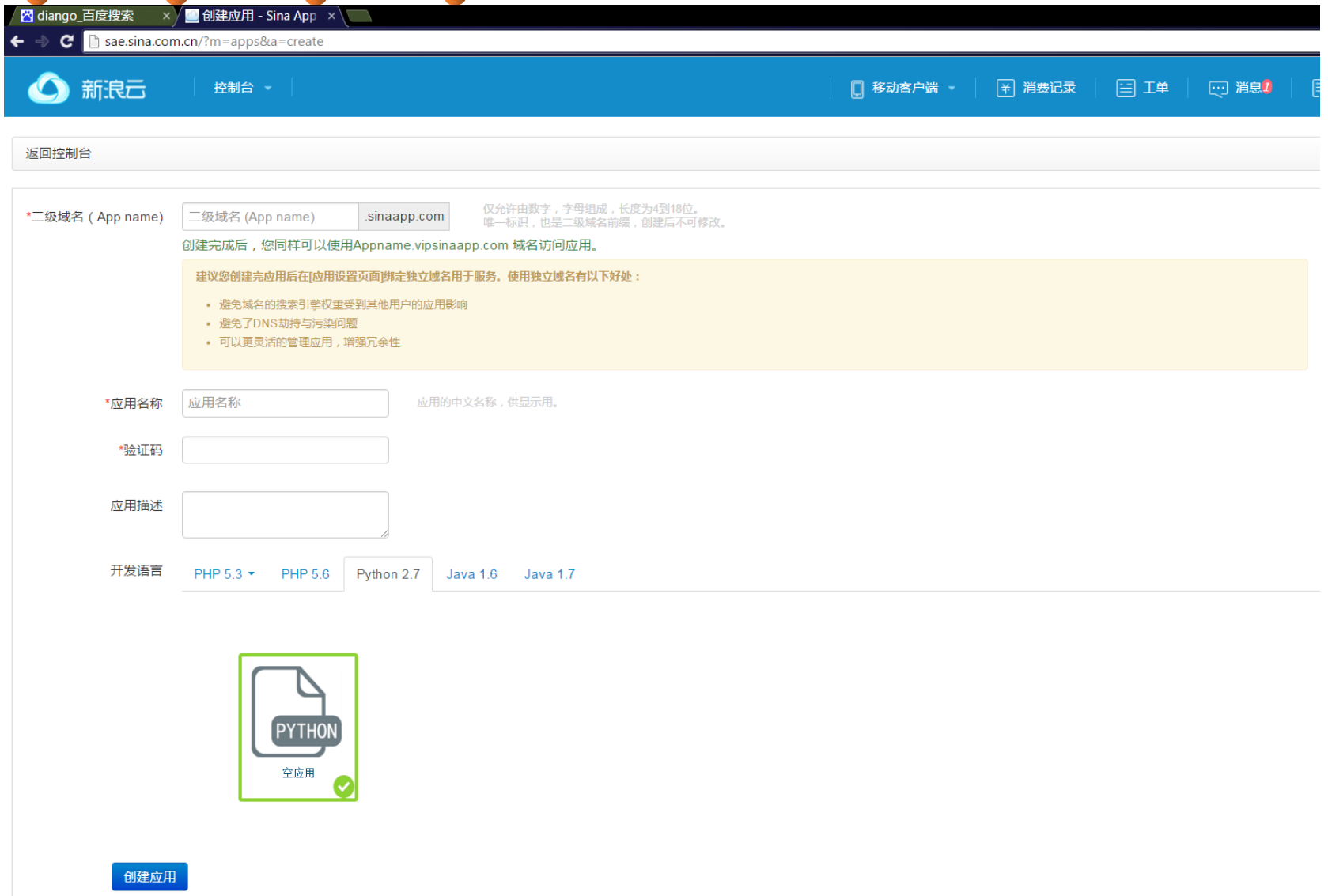
  – Views.py中： import mysql.connector

```
def search(request):

    if 'q' in request.GET and request.GET['q']:
        q = request.GET['q']

        conn = mysql.connector.connect(host='localhost', user='root',passwd='123456',db='bookdb',charset='utf8')
        cur = conn.cursor()
        cur.execute("select * from Book where title='%s'"%q)
        books = cur.fetchall()
        conn.close()

        return render_to_response('search_results.html',{'books': books, 'query': q})


    else:
        return HttpResponse('Please submit a search term.')
```

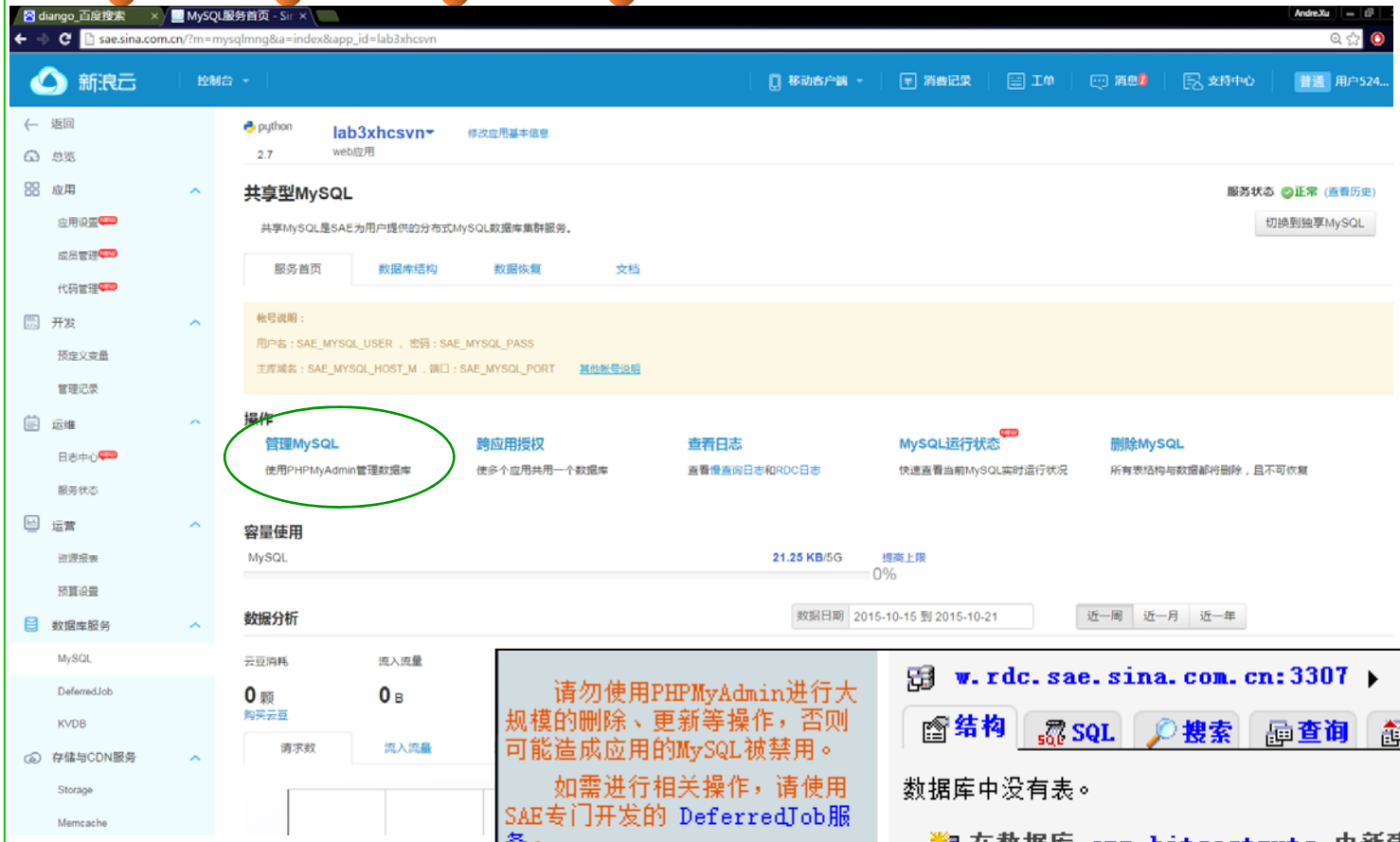
在SAE上建立帐号

- **SAE：PaaS云平台，相当于将本地的SaaS执行环境(Tomcat、MySQL、Python等)迁移到了新浪所提供的云平台上。**

在SAE上创建应用

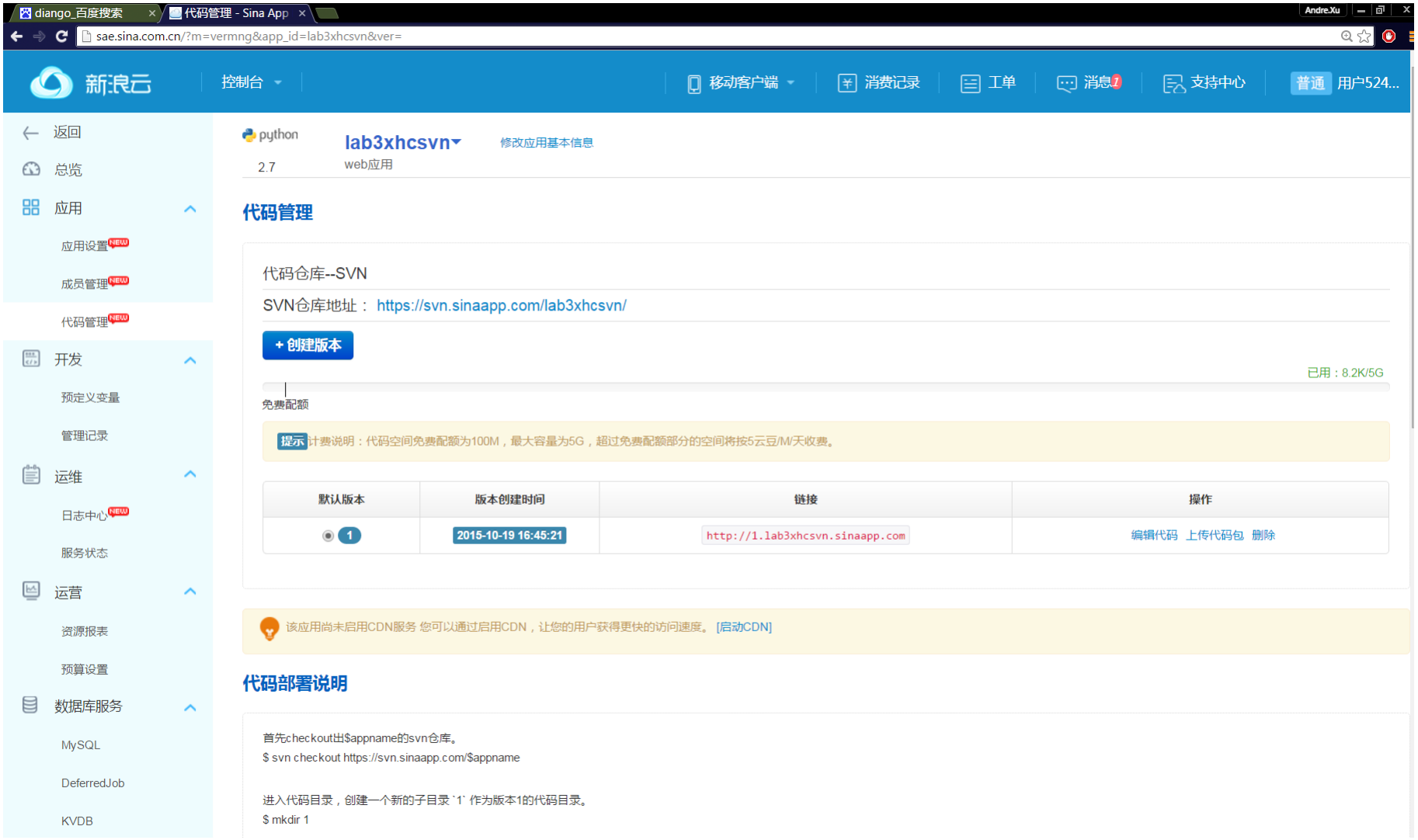

diango_百度搜索
创建应用 - Sina App
sae.sina.com.cn/?m=apps&a=create
新浪云
控制台
移动客户端
消费记录
工单
消息
返回控制台
*二级域名（App name）
二级域名 (App name)
.sinaapp.com
仅允许由数字，字母组成，长度为4到18位。
唯一标识，也是二级域名前缀，创建后不可修改。
创建完成后，您同样可以使用Appname.vipsinaapp.com 域名访问应用。
建议您创建完应用后在[应用设置页面]绑定独立域名用于服务。使用独立域名有以下好处：
避免域名的搜索引擎权重受到其他用户的应用影响
避免了DNS劫持与污染问题
可以更灵活的管理应用，增强冗余性
*应用名称
应用名称
应用的中文名称，供显示用。
*验证码
应用描述
开发语言
PHP 5.3
PHP 5.6
Python 2.7
Java 1.6
Java 1.7
PYTHON
空应用
创建应用

配置相关的服务(Database等)



注：
1.点击创建的应用后进入左图的配置界面；
2.点击“管理MySQL”进行数据库设置；
3.可将本地DB中的所有表结构和数据以SQL语言的形式导出，然后在SAE中导入。

将程序部署至SAE


diango_百度搜索
代码管理 - Sina App
sae.sina.com.cn/?m=vermng&app_id=lab3xhcsvn&ver=
新浪云
控制台
移动客户端
消费记录
工单
消息
支持中心
普通 用户524...
返回
总览
应用
应用设置
成员管理
代码管理
开发
预定义变量
管理记录
运维
日志中心
服务状态
运营
资源报表
预算设置
数据库服务
MySQL
DeferredJob
KVDB
python
2.7
lab3xhcsvn
修改应用基本信息
web应用
代码管理
代码仓库--SVN
SVN仓库地址： https://svn.sinaapp.com/lab3xhcsvn/
+ 创建版本
已用：8.2K/5G
免费配额
提示 计费说明：代码空间免费配额为100M，最大容量为5G，超过免费配额部分的空间将按5云豆/M/天收费。
默认版本
版本创建时间
链接
操作
1
2015-10-19 16:45:21
http://1.lab3xhcsvn.sinaapp.com
编辑代码 上传代码包 删除
该应用尚未启用CDN服务 您可以通过启用CDN，让您的用户获得更快的访问速度。 [启动CDN]
代码部署说明
首先checkout出$appname的svn仓库。
$ svn checkout https://svn.sinaapp.com/$appname
进入代码目录，创建一个新的子目录 `1` 作为版本1的代码目录。
$ mkdir 1

SAE部署

- **Sae.sina.com.cn 注册账号（微博账号）**
- **采用SVN或git进行代码的版本控制（注意用户名和密码是安全邮箱和安全密码，不是登陆用的微博账号和密码）http://www.sinacloud.com/doc/sae/tutorial/code-deploy.html**
- **由于SAE目前暂不支持最新版的Django，如果采用旧版Django（1.5以下），可参考官方说明 http://www.sinacloud.com/doc/sae/python/index.html**
- **如果采用较新版的Django，需要将Django包同代码一起上传，可参考:http://blog.csdn.net/a359680405/article/details/43113039、http://www.cnblogs.com/weberypf/p/4274199.html 等文章或类似的网站。**
- **数据库可在本地导出SQL语言（表结构和格式），然后在SAE中导入**

评判标准

- 是否完成图书SaaS中必选功能的开发；
- 是否顺利部署到SAE并能对外提供访问；
- 所开发的图书SaaS的质量；

- 对Python中开发Django程序的流程是否理解；
- 对Django的机理是否了解清楚；

提交方式

- 请遵循实验报告模板撰写。
- 不能修改数据库/表的结构和名字，不能变更规定的功能，但可扩展新功能。

- **提交日期：2015年11月5日 23:55**
- **提交两个文件到CMS：**
  - 实验报告：命名规则“学号-Lab3-report.doc”
  - SaaS工程所有文件打包：包含Python代码、网页代码、配置文件等，命名规则“学号-Lab3-code.rar/zip”

哈爾濱工業大學
HARBIN INSTITUTE OF TECHNOLOGY

结束