基于BKVision图表平台实现用户画像与行为分析

目录

实战开发需求

●**课题目标：**

1、巩固SaaS应用的软件开发&设计能力

2、了解蓝鲸BKVision图表平台的产品功能与使用方法

3、掌握基本Django中间件的开发技能与数据采集

4、掌握蓝鲸图表平台的嵌入方式与SDK使用

5、提升SaaS开发技能，巩固基础数据分析能力与数据采集技能

6、提升SaaS开发技能，进一步熟悉开发框架与后台建模

●**课题内容：**

在此前两期SaaS开发作业的基础上，借助蓝鲸BKVision图表平台实现用户行为可视化分析与前端嵌入，通过设计并开发Django中间件，实现用户行为数据埋点采集并存储至数据库，通过BKVision实现仪表盘嵌入

●**样例展示：**https://apps.ce.bktencent.com/stag--frontend--bk-class4/DashBoard

需求分析-功能点

- **功能一：** 设计实现用户行为存储Model
- **功能二：** 设计自定义中间件，实现用户行为埋点记录
- **功能三：** 在BKVision平台添加并配置数据源
- **功能四：** 创建并配置仪表盘
- **功能五：** 将仪表盘嵌入到自己的前端应用

目录

实战步骤

- **功能点：实现用户数据采集并存储记录到数据库中**
  - 步骤一：实现用户行为数据存储Model，并执行数据库的迁移
  - 步骤二：设计自定义中间件，实现用户行为埋点记录
  - 步骤三：推送后端代码并部署上线

- **功能点：在BKVision平台实现用户画像仪表盘**
  - 步骤四：在BKVision平台添加并配置数据源
  - 步骤五：创建并配置仪表盘

- **功能点：将仪表盘嵌入到前端应用**
  - 步骤六：使用iFrame方式将仪表盘嵌入到前端应用
  - 步骤七：部署展示

步骤一：实现用户行为数据存储Model

- 需要实现存储Model，存储两类数据
  - 请求的API类别（如CMDB，JOB），命名为api_category
  - 请求的API名称（如biz-list、set-list），命名为api_name
  - 在home_application/models.py中实现，以api_category 和 api_name为联合索引，确保每条API只存在一条记录
  - 每个API的访问次数命名为request_count

- Meta类的作用
  - Meta类用于提供关于模型的元数据。元数据是描述数据的数据，它不会直接存储在数据库中，而是用于配置模型的行为和属性。
  - unique_together: 指定模型字段的组合必须是唯一的。

- 执行数据库迁移
  - python manage.py makemigrations
  - python manage.py migrate

步骤二：设计自定义中间件，实现用户埋点记录

- 什么是中间件
  - Django中间件文档：Django中间件原理及示例

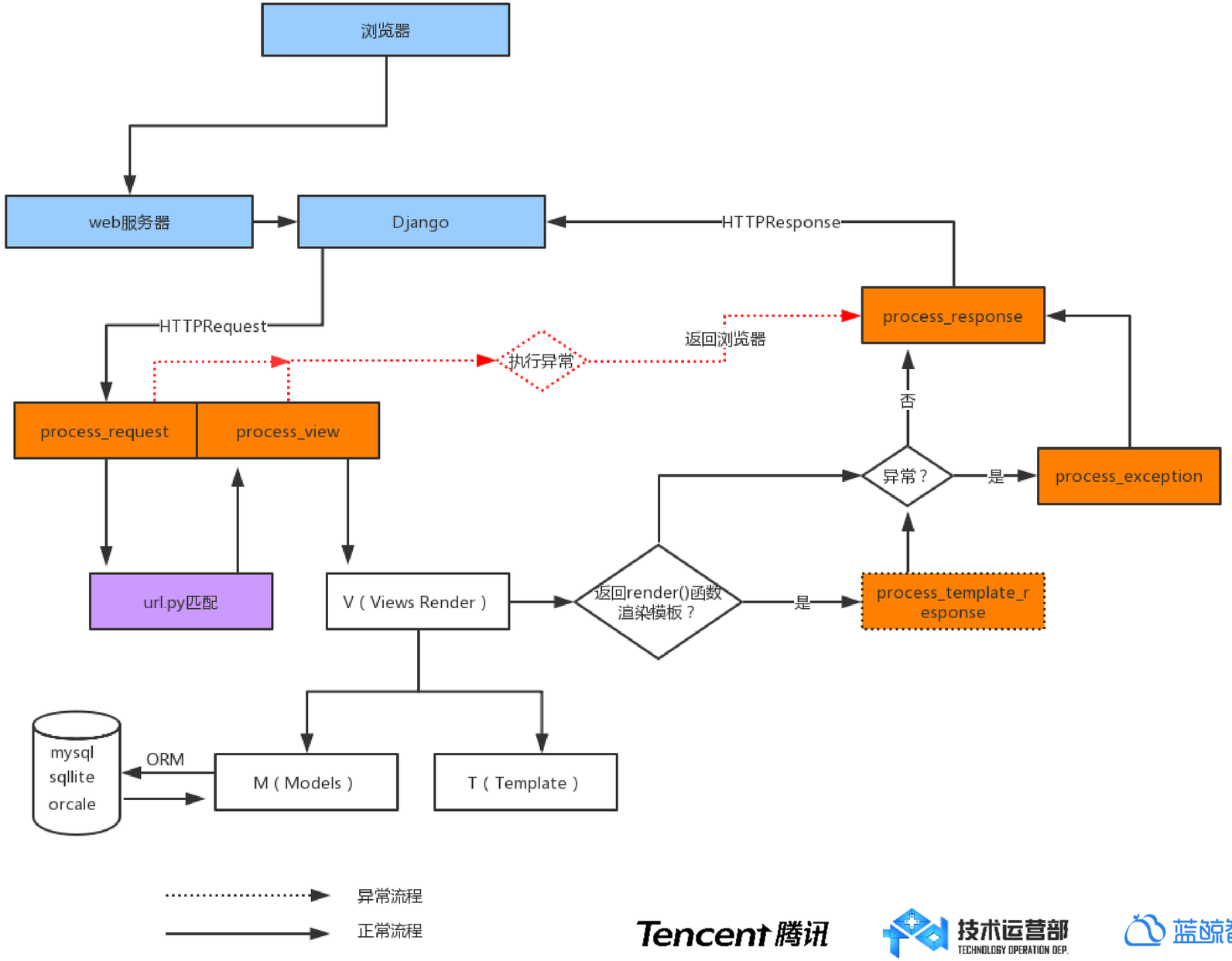
  - 中间件是一个轻量级、底层级别的插件系统，可以介入Django生命周期中的请求和响应的处理过程，控 制Django程序的输入和输出

  - 使用：在Django配置文件中的 MIDDLEWARE 中注册即可

  - Django中的中间件最多可以定义五个方法：
    - process_request(self,request)
    - process_view(self, request, view_func, view_args, view_kwargs)
    - process_template_response(self,request,response)
    - process_exception(self, request, exception)
    - process_response(self, request, response)

步骤二：设计自定义中间件，实现用户埋点记录

- Django中间件的执行流程



步骤二：设计自定义中间件，实现用户埋点记录

- 实现用户埋点记录
  - 重写process_request 方法，实现在用户请求时，拦截并埋点记录
  - 从 request.user.username 中获取到用户名
  - **注意：这里需要注意中间件的顺序，如果自定义中间件放在较前的位置的话，会拿不到用户名，因为用户名的处理是在蓝鲸开发框架内置的`blueapps.middleware.request_provider.RequestProvider`中实现的，所以我们的自定义中间件的执行顺序必须在其后**
  - 从 request.path 中获取到请求的接口名称，并进行split分割处理，只取最后的接口名称部分，忽略此 前的前缀
  - 根据映射表关系，判断当前接口所属的类别：CMDB/JOB
  - 存储行为数据到数据库中，使用`get_or_create`进行，因为每一个API只需要一个记录，我们需要做的是递增其`count`值
  - 添加自定义中间件到config/default.py 中的对应 MIDDLEWARE 中

步骤四：在BKVision平台添加并配置数据源

- 在BKVision平台添加并配置数据源
  - 访问BKVision平台，在空间【24华南理工实训课程】中点击【配置数据源】添加配置自己的SaaS应用的数据源
  - 选择【MySQL】数据源
  - 从开发者中心，获取自己的SaaS应用的数据库连接信息
  - 将数据库配置信息回填到BKVision中，并测试能否成功联通

步骤五：创建并配置仪表盘

- 创建并配置仪表盘
  - 在此前的【24华南理工实训课程】空间下，新增仪表盘
  - 编辑仪表盘->选择左侧的样式，拖拽到右边->选择此前配置好的数据源，选择需要的数据表
  - 这里可以选择任何喜欢的样式来展示
  - 本次课程选用【柱状图】来展示各类别API调用次数统计图，选用【仪表盘】展示API调用总次数，选用【饼图】来统计用户备份文件次数，选用【折线图】来统计API调用次数明细图

注意创建完成之后要保存

步骤六：将仪表盘嵌入到前端应用

- 将仪表盘嵌入到前端应用
  - 首先，在BKVision平台->嵌入管理(24华南理工实训课程空间下)->新增
  - 选择需要嵌入的仪表盘及嵌入目标应用的应用ID
  - 选择嵌入方式，此处选择iFrame进行嵌入，这里的一些教程链接无法打开，嵌入代码比较简单，参考本实验手册即可
  - 选择嵌入方式后，系统会自动生成一段iFrame代码，后续复制到自己的前端应用中
  - 在自己的前端应用中新增页面
    - 在前端代码的`router/index.js`中新增`DashBoard`页面的配置
    - 在`src/App.vue`中新增`DashBoard`相关配置
    - 在`src/views`目录下新增文件夹`DashBoard`，在`src/views/DashBoard`目录下新增Vue组件文件`index.vue`（刚刚复制的iFrame代码就粘贴到这里）
  - 发布仪表盘并推送代码到远程仓库，部署上线

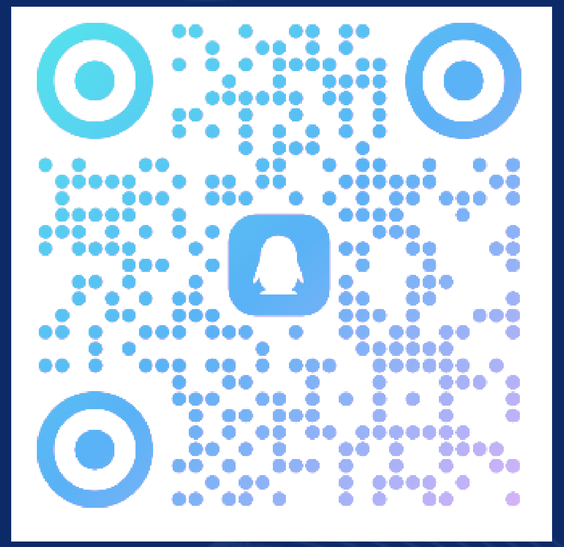
目录

实验评分标准

整体要求	采用迭代方式进行需求分析、面向对象设计和编程实现，实训课报告中需包含相应的需求规约、设计规约，项目开发说明
考点一	在此前两期课程SaaS作业的基础上，通过Django中间件实现用户行为采集并存储到SaaS数据库，比如：登录行为、查询业务列表行为、执行作业行为等
考点二	在BKVision图表平台创建空间，接入对应的SaaS数据库，并对采集的数据进行仪表盘配置（仪表盘样式不限，鼓励大家自由发挥），并发布仪表盘
考点三	设计通过iFrame或BKVision-SDK方式，实现仪表盘发布并嵌入到对应的SaaS前端界面中
其他评分项	1.Python代码符合PEP8规范，可酌情加分
	2.系统边界考虑完善，系统性能优良，可酌情加分
	3. Django中间件实现出色，采集覆盖大部分接口场景，可酌情加分
	4. Django中间件在实现数据存储时，能够通过Celery异步任务实现，可酌情加分
	5.前端界面优美，用户交互体验良好，可酌情加分
	6.后端代码能够实现单元测试以及日志、异常处理等，可酌情加分

蓝鲸智云

官方微信公众号



蓝鲸高校培训群