暑期学校实验项目：高考志愿填报助手

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 小组名称 | 第十组：Django开发 | | | | | | |
| 姓 名 | 陈耿 | 专业 | 人工智能 | 班级 | 人工智能2班 | 学号 | 09118212 |
| 实验时间 | 2020.8.31-2020.9.23 | | 指导教师 | 孔祥龙 | | 成绩 |  |
| 1. **实验背景和目的**   **背景：**放假在家，邻居孩子跑来问你，我这成绩能考上东南大学么？如果考不上东大，我能考上哪？  **目的：**爬虫自动获取不同梯队的学校信息，构建院校、专业、各省分数的知识图谱，训练模型以实现分数同目标学校专业的适配度的精准评估。 | | | | | | | |
| 1. **小组任务和个人任务**   **小组任务：**通过对接其他小组的数据和算法，通过Django框架下的ORM操作来搭建一个包含大学、专业、排名等类的数据库。  **个人任务：**创建登陆界面、创建数据表格、对接第七组、上传Majors数据至数据库、数据修改、数据库四次更新。 | | | | | | | |
| **三、个人任务需求分析**  **1）创建登陆界面**  使用PyCharm创建Django项目，创建一个由用户名、密码、验证码输入框和提交按钮构成的登陆界面。   1. **创建数据表格**   使用Django框架进行ORM操作，设定好models.py后，通过命令行操作实现数据表的创建。   1. **对接第七组**   和第七组：AI算法应用B组进行对接。   1. **上传Majors数据至数据库**   读取高校分数线整合等csv格式文件并上传至数据库。   1. **数据修改**   协助第一组进行数据清洗，保证数据的一致性。  **6）数据库更新**  进行数据库的四次更新， | | | | | | | |
| **四、实验过程（需附上关键代码及相关说明）**  **1）创建登陆界面**  POST方式登录页html源码。  <html lang="en">  <head>  <meta charset="UTF-8">  <title>Title</title>  </head>  <body>  <form action=" /seu/login/check/" method="post">  <p>  <label>用户名:</label><input type="text" name="uname">  </p>  <p>  <label>密&emsp;码:</label><input type="password"name="pwd">  </p>  <p>  <label>验证码:</label><input type="text"name="yzm">  </p>  <p>  &emsp;&emsp;&emsp;<input type="submit" name="login">  </p>  </form>  </body>  </html>   1. **创建数据表格**   定义大学类，并创建表格。  class Colleges(models.Model): # 大学类  collegeID = models.IntegerField(primary\_key=True) # 大学id（主键）  collegeName = models.TextField() # 大学名称  provinceID = models.ForeignKey('Provinces',  on\_delete=models.DO\_NOTHING,  null=True) #省份id（外键->Provinces类）  project985 = models.BooleanField(default=False) #是否为985大学  project211 = models.BooleanField()#是否为211大学  top = models.BooleanField() #是否为一流学校  class Meta:#设置元信息，使用db\_table自定义表的名字  db\_table = 'Colleges'  verbose\_name = '大学列表'  通过命令行运行python manage.py makemigrations，在app内的migrations包内将类转换为数据表结构。  再运行python manage.py migrate，运行操作文件，创建数据表。   1. **对接第七组**   和第七组：AI算法应用B组进行对接，提供数据库代码及访问方式。   1. **上传Majors数据至数据库**   创建getMajors函数，读取高校录取分数线整合文件中的数据并上传至数据库，  其中包含了部分异常数据处理操作。  def getMajors(request):  df=pd.read\_csv('./extFiles/高校录取分数线整合（省份名字统一）.csv')  num=df.shape[0]  for i in range(num):  if df.at[i,'category']=='不分文理':  df.at[i,'category']='综合'  if df.at[i, 'College']=='华东理科大学':  df.at[i, 'College']='华东理工大学'  if df.at[i, 'College'] == '北京理科大学':  df.at[i, 'College'] = '北京理工大学'  if df.at[i, 'College'] == '\ufeff北京航空航天大学':  df.at[i, 'College'] = '北京航空航天大学'  c1=models.Majors(majorName=df.at[i,'Major'],  year=df.at[i,'Year'],  id=i,  categoryID=models.Category.objects.get(categoryname=df.at[i,'category']),  minScore=df.at[i,'score'],  collegeID=models.Colleges.objects.get(collegeName=df.at[i,'College']),  provinceID=models.Provinces.objects.get(provinceName = df.at[i,'Province']))  c1.save()  return HttpResponse("Majors in!")    **5）数据修改**  协助第一组进行数据清洗，保证数据库中数据的一致性。修改不一致的省份名称，如“江苏”、“江苏省”；删除非本科一批录取的数据；修改csv文件中的异常数据，如‘\ufeff北京航空航天大学’。  **6）数据库更新**  第一次： 检查核对分数线数据名称及格式，对比任务分配表检查贡献者与大学名称是否对应。  第二次：检查核对2018上海all数据、2018山西文科和理科数据、2018黑龙江文科和理科数据、2018江苏文科和理科选测BC数据。  第三次：检查核对json文件中福建省数据。  第四次：创建缺失的一分一段表json文件。 | | | | | | | |
| 1. **实验结果与分析**   构建了如下图所示的数据库。数据库包含六个实体对象，分别是“大学”、“专业”、“一级学科”、“省份”、“分数排名”、“科类”。大学类中包含六个属性，其中“大学ID”为主键，“省份ID”为省份类的外键。专业类中包含九个属性，其中“大学ID”为大学类的外键，“省份ID”为省份类的外键，以此类推。  其中，专业类的属性“一级学科IDs”表示该专业对应的一级学科，可以没有也可以有多个，所以这里的一级学科是一个文本数据，多个ID由特定字符分开。因此，这里的一级学科IDs与一级学科类中的一级学科ID不同，也不是外键。  其他小组只能查询数据库，不能增、删和修改数据表内容。    数据库ER图（对应代码参数） | | | | | | | |
| **六、实验总结与心得体会**  本次实验中，我初步接触了多人通过码云在线协作完成一个项目的过程。  通过本次实验，我有很多收获。首先，我初步学会了在PyCharm上创建一个Django项目，  并学会使用Django制作简单的网页。其次，通过分组完成任务，我认识到了多人协作的  重要性，学会了更好的合作完成任务。最后，我在与其他组对接的过程中体会到了协作的  难度，提升了对接听取需求的能力。  本次实验中还存在几点不足：数据来源不同导致数据不具有一致性，在导入数据库之前花  废了大量时间进行处理；组间协作不足，导致降低效率；不同组工作量差异较大。 | | | | | | | |

2020年9月制