《Python全英》

大作业项目总结报告

题目：A题

课程资源推荐平台

组号 30

组员1 刘伟哲 21371421

组员2 龚家年 21371402

组员3 王艺杰 21371429

组员4 王乐 21371401

2023年7月

一、功能简介

1. 爬取课程资源；
2. 导入neo4j知识图谱文件并将其可视化，点击知识图谱中的知识节点显示课程资源；
3. 为知识图谱实现了CRUD（创建、读取、更新、删除）操作和书签，新增节点时还可以爬取相应课程；
4. 实现课程资源根据不同指标的排序，并可根据自定义关键字进行相关度排序；
5. 用户系统实现了注册及登录功能，不同用户可个性化定制自己的知识图谱；
6. 用户信息保存在服务器，支持远程登录。

二、已完成任务

必做任务完成情况（4/4）

1. 为已给出的课程学习网站以外的来源的课程资源开发网络抓取工具；
2. 导入neo4j知识图谱文件并将其可视化。此外，提供删除现有知识图谱节点的方法；
3. 点击知识图谱中的知识节点，应该会显示对应的课程资源，资源应该根据不同的指标进行排序；
4. 实施包括注册、登录和用户仪表板的用户系统。知识图谱资源应根据不同的用户进行区分。

选做任务完成情况（4/4）

1. 对其他课程来源执行网络抓取，包括：

- [必做]部分5个网站：
  - Chinese University MOOC
  - iMOOC
  - iCourse
  - HaoDaXue Online
  - BiliBili
- [选做] 额外3个自选网站：
  - NetEase Open Course
  - Tencent Classroom
  - Mooc China

2. 为知识图谱实现 CRUD（创建、读取、更新、删除）操作和书签；
3. 自定义关键字对课程排序；（自定义）
4. 添加节点时实时抓取课程信息。（自定义）

5. 将用户信息保存在远程服务器。（自定义）

三、总体设计方案

仔细研究后，我们将项目初步解耦
为 爬虫 、 登录系统与MySQL数据库管理 、 知识图谱与Neo4j数据库管理 、 指标排序 、 GUI 五个方面。其中，知识图谱还包括 前端 + 后端 两个部分； GUI 又细分为 初始窗口 、 登录窗口 、 注册窗口 、 知识图谱窗口 。

1. 爬虫

- 具体功能：开发了一个网页工具，可在 8 个课程平台中，根据所选关键词获取所需课程信息，包括课程名、URL 链接、来源院校。
  初始时，根据已有知识节点，爬取相关课程，并将相关的课程存进neo4j数据库；在新增节点时（必须是学科类节点），根据节点名称能爬取相关课程。
- 实现逻辑：由于各大课程平台均为动态网站，使用 JavaScript 等方法实时展示课程搜索信息，我们选择 Selenium 模拟浏览器行为，抓取动态网页并解析获取课程信息。Selenium 通过 WebDriver(浏览器驱动) 和浏览器进程进行通信，向其发送命令和接收响应，从而模拟用户行为。为了提高网页抓取效率，我们使用 multiprocessing 异步多进程加速，创建 8 个进程分别抓取课程网站信息。
  Selenium 抓取动态网页信息主要有以下 3 步：
  - 1. 浏览器加载网页，等待渲染完成；
  - 2. 根据特定网站结构，浏览器定位并获取所需课程信息；
  - 3. 关闭浏览器，释放资源。
- 核心代码：

```
def scraping(keywords: Iterable[str]) -> dict:
    '''网页抓取优化：多进程异步'''
    pool = multiprocessing.Pool()
    manager = multiprocessing.Manager()

    shared_dict = manager.dict()

    for web in web_scraping:
        pool.apply_async(task, (shared_dict, web, keywords))

    pool.close()
    pool.join()
    return shared_dict
```

- 具体实现图（下图为初始图的课程信息 excel 表格部分截图。）：

理论计算机科学	bilibili,【北京大学】理论计算机科学基础（全70讲）,https://	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	netease,	netease,	netease,	netease,	netease,	netease,	netea
数据结构	icourses,计算机软件基础（数据结构）,https://www.icourse163	icourses,	icourses,	icourses,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	netease,	netease,	netea
算法	icourses,算法设计与问题求解,https://www.icourse163.org/cou	icourses,	icourses,	icourses,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	netease,	netease,	netea
自动机理论	bilibili,【斯坦福大学】CS154 自动机理论· 2020年春（完结·	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	netease,	netease,	netease,	netease,	netease,	netease,	netea
形式化方法	bilibili,逻辑与形式化方法-龙星计划,https://www.bilibili.co	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	netease,	netease,	netease,	netease,	netease,	netease,	netea
信息论	bilibili,国防科技大学-信息论与编码基础（国家级精品课）,htt	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	netease,	netease,	netease,	netease,	netease,	netease,	netea
编码理论	bilibili,西安电子科技大学刘刚教授讲授《信息论与编码理论》,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	netease,	netease,	netease,	netease,	netease,	netease,	netea
计算复杂性理论	bilibili,【卡内基梅隆大学】复杂性理论 ｜ 理论计算机科学 ｜	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	netease,	netease,	netease,	netease,	netease,	netease,	netea
计算学习理论	bilibili,计算学习理论,https://www.bilibili.com/video/BV1fJ	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	netease,	netease,	netease,	netease,	netease,	netease,	netea
密码学	icourses,区块链中的密码学,https://www.icourse163.org/cours	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	netease,	netease,	netease,	netease,	netease,	netea
分布式计算	bilibili,【算法姬】通俗易懂的算法教程：了解分布式计算,http	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	netease,	netease,	netease,	netease,	netease,	netease,	netea
机器学习	icourses,机器学习与模式识别,https://www.icourse163.org/cou	icourses,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	netease,	netease,	netease,	netease,	netea
编程语言理论和程序语义	bilibili,我该学哪个编程语言？对比10+主流语言优劣，入门编程	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	netease,	netease,	netease,	netease,	netease,	netease,	netea
量子计算	bilibili,【完结】大一新生也能懂的量子计算,https://www.bili	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	netease,	netease,	netease,	netease,	netease,	netease,	netea
计算机系统	bilibili,爆肝上传！清华大佬终于把困扰我大学四年的【计算机	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	netease,	netease,	netease,	netease,	netease,	netease,	netea
计算机体系结构	bilibili,计算机体系结构 - 国防科技大学(35讲),https://www.b	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	netease,	netease,	netease,	netease,	netease,	netease,	netea
计算机组成原理	icourses,计算机组成原理,https://www.icourse163.org/course/	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	netease,	netease,	netease,	netease,	netease,	netea
并行计算	bilibili,【卡耐基梅隆大学】15-418（CS149） 并行计算结构与	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	netease,	netease,	netease,	netease,	netease,	netease,	netea
操作系统	icourses,计算机操作系统,https://www.icourse163.org/course/	icourses,	icourses,	icourses,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	netease,	netease,	netea
嵌入式系统	icourses,嵌入式系统,https://www.icourse163.org/course/NWPU	icourses,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	netease,	netease,	netease,	netease,	netea
分布式系统	bilibili,【麻省理工学院—中文字幕】也只有MIT大佬才能把分布	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	netease,	netease,	netease,	netease,	netease,	netease,	netea
计算机网络	icourses,计算机网络,https://www.icourse163.org/course/USTC	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	netease,	netease,	netease,	netease,	netease,	netea
计算机安全	bilibili,【麻省理工公共课】网络安全和计算机安全(已更新中文	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	netease,	netease,	netease,	netease,	netease,	netease,	netea
计算机应用技术	bilibili,计算机应用技术专业（专科篇）,https://www.bilibili	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	netease,	netease,	netease,	netease,	netease,	netease,	netea
数据库系统	icourses,数据库系统,https://www.icourse163.org/course/CQU-	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	netease,	netease,	netease,	netease,	netease,	netea
计算机视觉	bilibili,计算机视觉与深度学习 北京邮电大学 鲁鹏 清晰版合集	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	netease,	netease,	netease,	netease,	netease,	netease,	netea
图像处理	icourses,图形图像处理技术,https://www.icourse163.org/cours	icourses,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	netease,	netease,	netease,	netease,	netea
虚拟现实和增强现实	bilibili,增强现实、虚拟现实和混合现实什么区别？不要被误导	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	netease,	netease,	netease,	netease,	netease,	netease,	netea
多媒体技术	bilibili,【公开课】 多媒体技术及应用,https://www.bilibili.	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	netease,	netease,	netease,	netease,	netease,	netease,	netea
人工智能	icourses,人工智能基础,https://www.icourse163.org/course/NU	icourses,	icourses,	icourses,	icourses,	icourses,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	netea
软件工程	bilibili,我来告诉你【软件工程】会学些什么！,https://www.bi	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	netease,	netease,	netease,	netease,	netease,	netease,	netea
科学计算	bilibili,MATLAB科学计算（基础部分共10章，已完结）,https://	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	netease,	netease,	netease,	netease,	netease,	netease,	netea
计算机辅助设计	icourses,计算机辅助设计,https://www.icourse163.org/course/	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	bilibili,	netease,	netease,	netease,	netease,	netease,	netea

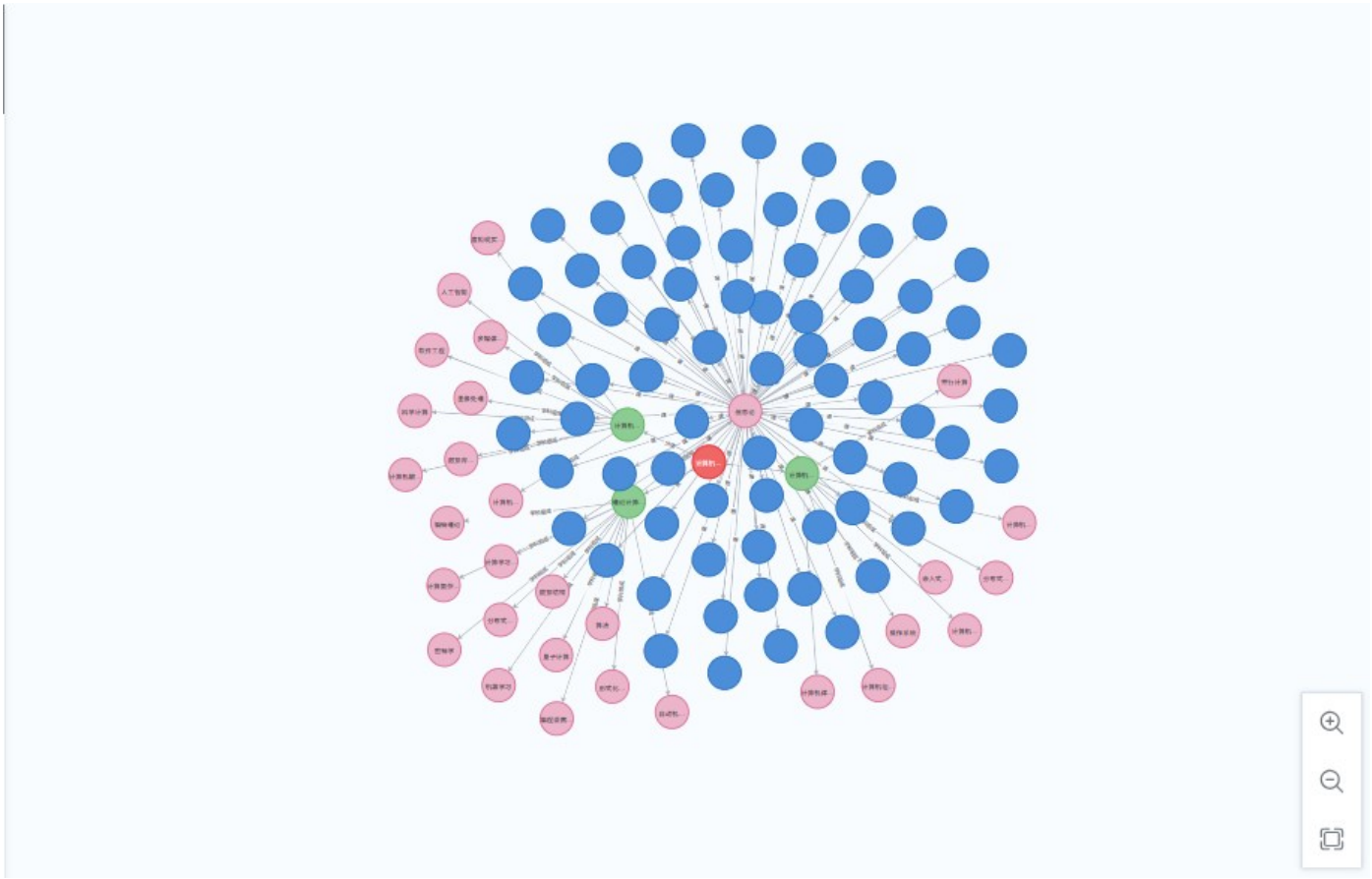
2. 知识图谱与Neo4j数据库管理

- 具体功能：实现带有各种交互功能的可供用户定制的知识图谱，提供包括CRUD、书签、点击节点显示相关课程的交互功能。同时管理 `Neo4j` 数据库。
- 实现逻辑：为实现交互功能，我们采用了 `Echarts + html + js + css` 编写前端， `Flask` 作为后端，最后嵌入GUI之中。

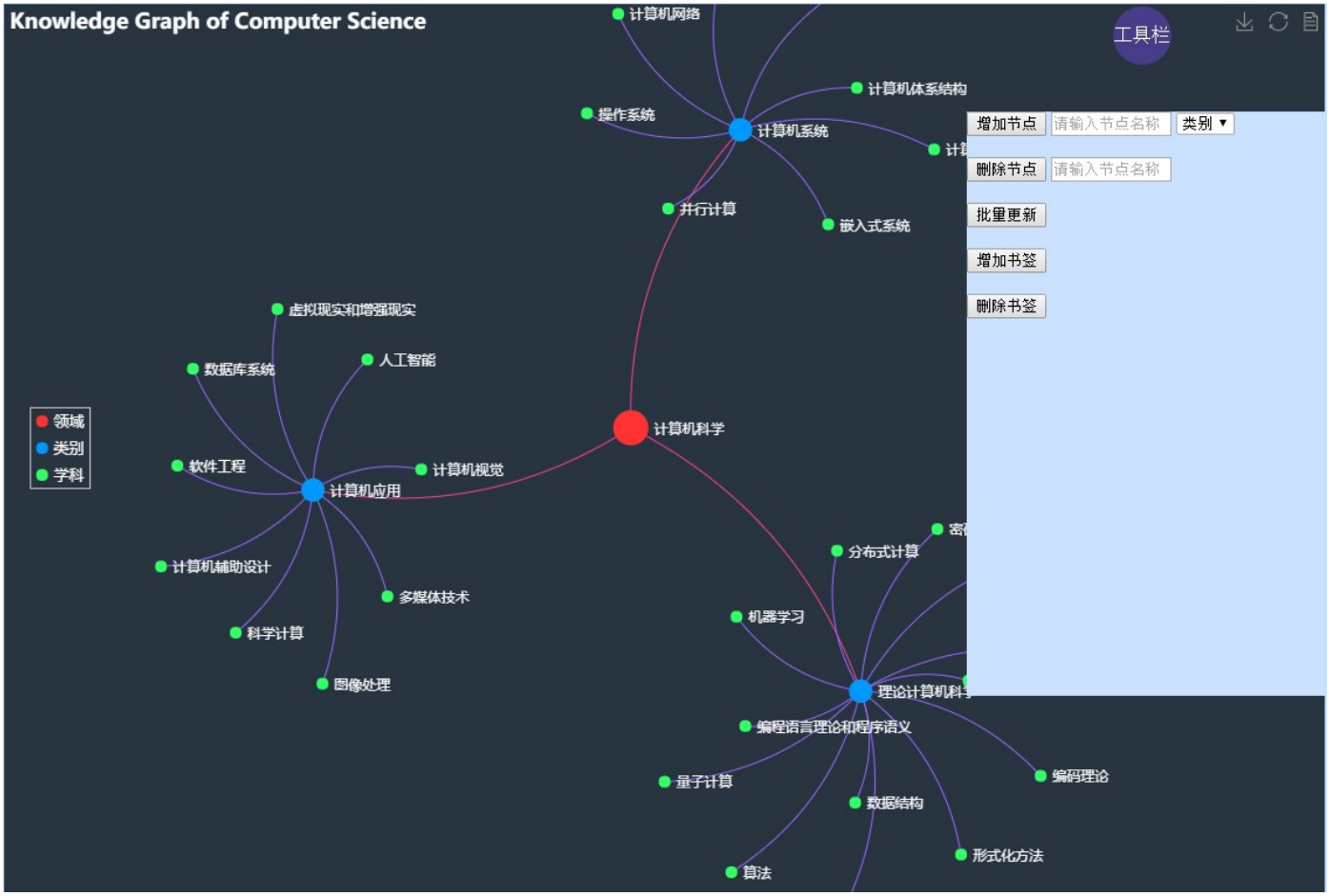
  对于 `Neo4j` 数据库的管理，分为初始化和后续动态管理。初始化部分，提供初始知识图谱与课程信息，编写为init_Neo4j.py；后续动态管理，主要体现在增删节点以及增加节点时爬取课程信息对数据库的扩充更新，编写为manage_Neo4j.py。同时，用以为后端提供接口。
- 核心代码（此处仅展示部分Neo4j的动态管理核心代码，分别为增加节点并爬取课程、点击节点取出课程列表）：

```
# 此为管理Neo4j的动态管理实现
from py2neo import Graph, Node, Relationship, NodeMatcher
...

KG = Graph("neo4j://localhost:7687", auth=("neo4j", "buaayyds"))

# 增加节点
def add_single(data):
    label = data['category']
    name = data['name']
    father = data['classify']
    ... # 属性相关处理

    # 增加节点
    node = Node(label, name=name, fixedName=name, classify=father)
    KG.create(node)
    ... # 边相关处理
    relationship = Relationship(father_node, rtype, node)
    KG.create(relationship)

    # 爬取课程
    new_list = Spiders.scraping(name)
    ... # 加入数据库

# 批量增加
def add_in_neo4j(data):
    for item in data:
        add_single(item)

...

# 为实现点击展示课程，故须从Neo4j中取出数据
def fetch_data(name):
    relationship_matcher = RelationshipMatcher(KG)
    # 查找指定名称的父节点
    parent_node = KG.nodes.match(name=name).first()
    ...# 提取属性等处理

    return utils.sorter.sort_by_name(result_list[1:])

...
```

- 具体实现图：
  Neo4j数据库（仅展示部分）：



初始知识图谱（仅展示部分）：



3. 登录系统与MySQL数据库管理

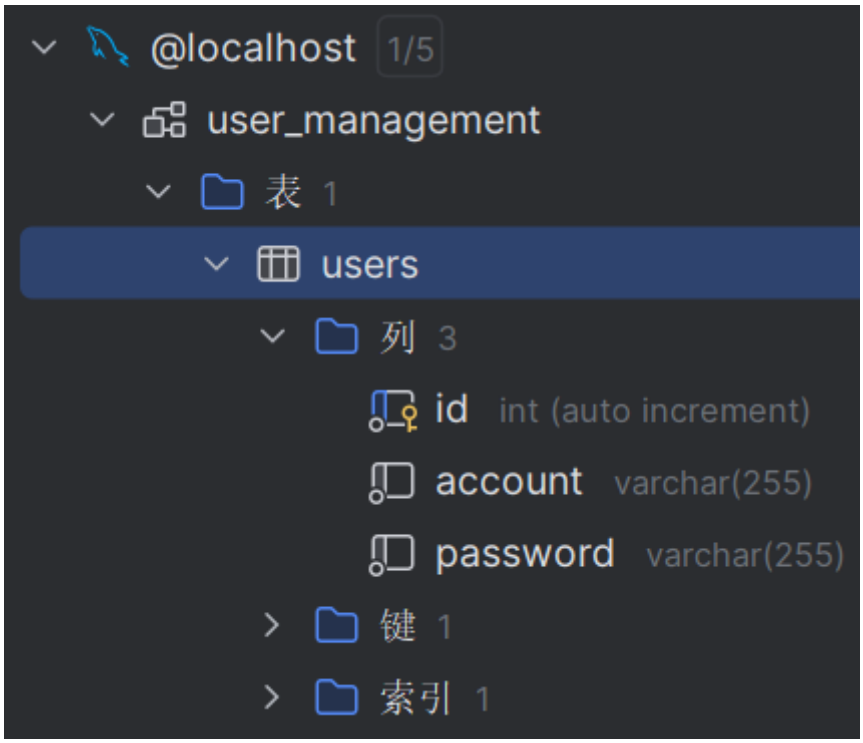
- 具体功能：存储用户账号信息以及知识图谱的节点与关系资源，从而做到对不同用户，可以实现个性化定制知识图谱。将用户账号密码信息保存在 MySQL 数据库中，相较于保存在明文文件中，提高了用户信息安全性。
- 实现逻辑：使用 pymysql 库与本地 MySQL 连接。在连接到数据库后，会尝试执行 MySQL 初始化代码，创建 user_management 数据库， users 数据表，在其中保存用户账号及密码信息。用户登录会读取 MySQL 信息，并反馈登录情况。注册会先检查账户名是否已存在，若不存在则成功注册，并写入 MySQL，否则注册失败。
  在本机登录的用户，其知识图谱节点与边的信息会以用户名区分，以json格式保存在本地（在知识图谱的后端中具体实现）。登陆后，会根据用户名调用对应的数据，进行知识图谱的渲染，从而实现不同用户对各自知识图谱的个性化定制。
- 核心代码：

```
# 连接数据库
...
db = pymysql.connect(host='47.93.46.112',
                     user='root',
                     password='123456')
...
```

```
// 使用JavaScript编写，使用Ajax机制进行前后端交互
// 保存节点数据
...
function saveNodes() {
    var dataToSend = option_KG.series[0].data;

    fetch('/api/saveNodes', {
        method: 'POST',
        headers: {
            'Content-Type': 'application/json'
        },
        body: JSON.stringify(dataToSend)
    })
    .then(response => response.json())
    .then(data => {
        console.log('后端返回的响应数据:', data);
    })
    .catch(error => {
        console.error('发送请求失败:', error);
    });
}
... // 保存边相关数据
```

- 具体实现图：

  `user_management` 数据库：



4. 指标排序

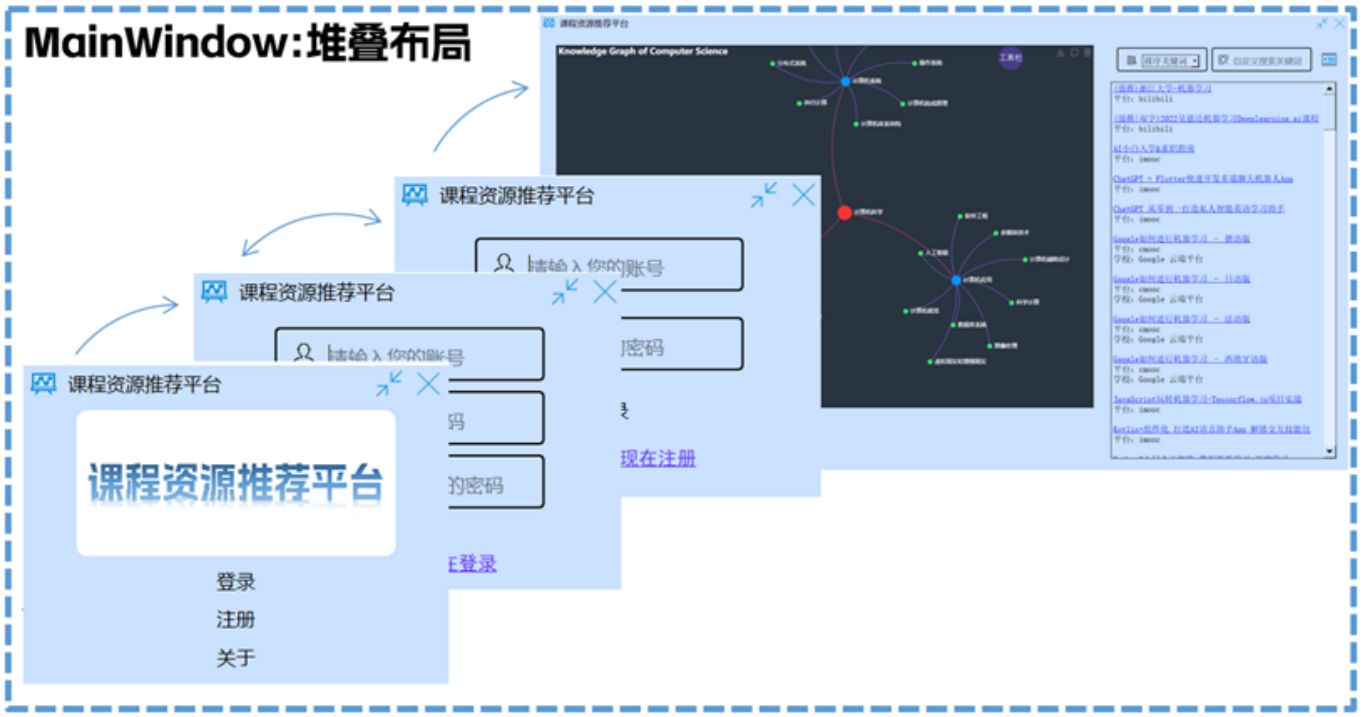
- 具体功能：不仅可以按照“课程名称”、“课程平台来源”、以及“课程所属院校”的字典序对课程列表排序，还可以根据输入关键字与课程信息的相关度对课程列表排序，为用户提供个性化的课程检索服务。
- 实现逻辑：我们围绕知识图谱的排序功能专门设计了sorter模块，其中共包括四个排序函数，分别对应了“输入关键字”、“课程名称”、“学校名称”以及“平台名称”四类排序指标。
  - 基于关键字的相关性排序通过调用Python中的 difflib 包实现。difflib中的SequenceMatcher类可求解“源字符串”与“目标字符串”之间的最长公共子序列，为字符串之间的相关度评估分数。sorter模块综合 “课程名称”、“课程平台来源”、“课程所属院校”三类信息作为“源字符串”、将用户输入关键字作为“目标字符串”计算相关度评估分数，并使用Python内置的排序函数 sorted( ) 根据相关性评分从高到低的顺序对课程列表进行排序。
  - 基于“课程名称”、“学校名称””、“平台名称”这三类指标的排序处理方式相似，这里以“学校名称”为例阐述实现逻辑。在排序前，知识图谱会使用Python内置的过滤函数filter( ) 筛除“学校名称”对应属性值为空字符串的课程信息，再使用排序函数sorted( ) 根据“学校名称”信息的字典序对课程列表进行排序。
- 核心代码：

```
import difflib

def get_similarity(s1, s2):
    return difflib.SequenceMatcher(None, s1, s2).ratio()

# 按关键字相关度排序
def sort_by_search_key(course_list, search_key):
    return sorted(course_list, key=lambda info: get_similarity(search_key, info[0] + info[1] + i

# 按课程名称排序
def sort_by_name(course_list):
    filtered_course_list = list(filter(lambda info: info[1] != '', course_list))
    return sorted(filtered_course_list, key=lambda info: info[1])

...
```

- 具体实现图：

5. GUI

- 总体界面设计：
  GUI界面基于PyQt5设计。整个程序的入口为MainWindow.py，主要采用堆叠布局(QStackedLayout)，以实现初始窗口(InitWindow.py)、登录窗口(LoginWindow.py)、注册窗口(Register.py)、知识图谱窗口(GraphWindow.py)之间的切换。如图所示：



定义主窗口的核心代码如下：

```
class MainWindow(MyTitleWidget):
    def __init__(self):
        super().__init__()
        # 用到的窗口
        self.init_win = InitWindow(caller=self)
        self.log_win = LoginWindow(caller=self)
        self.reg_win = RegisterWindow(caller=self)
        self.graph_win = GraphWindow(caller=self)

        # 创建堆叠布局器并将可切换的窗口加入
        subWidget = QWidget()
        ...

    def shift_to_login(self):
        self.s_layout.setCurrentIndex(1)

    ...

    def shift_to_graph(self, name):
        ...
        self.graph_win.init_html(name)
        ...
        self.s_layout.setCurrentIndex(3)
        ...
```

- 知识图谱功能的实现：
  知识图谱窗口(GraphWindow.py)的图界面使用QWebEngineView来渲染html格式文件，由app.py程

序提供域名和交互支持。切换到该窗口时，由主进程使用Qt封装的QProcess类创建app.py子进程提供支持，并通过将其stdout和stderr连接到主进程的槽函数以实现进程间通信。再设置QWebEngineView的域名，渲染知识图谱界面。如图所示：

QWebEngineView组件渲染的html页面，由app.py子进程提供域名和主进程交互支持

其初始化核心代码如图所示(name为用户名参数)：

```
def init_html(self, name):
    args = []
    script_path = os.path.join(self.app_path, "app.py")
    args.append(script_path)
    args.append(name)
    self.flask_process = QProcess()
    self.flask_process.setWorkingDirectory(self.app_path)
self.flask_process.readyReadStandardOutput.connect(self.handle_stdout)
    self.flask_process.readyReadStandardError.connect(self.handle_stderr)
    self.flask_process.start('python', args)
    self.webview.setUrl(QUrl('http://127.0.0.1:5000'))
```

- 交互功能实现：
  采用将鼠标/键盘事件连接到槽函数的方法实现各种互动功能。例如，知识图谱窗口侧边栏的搜索框按回车实现排序的代码如下：

```
self.imgLineEdit.line_edit.returnPressed.connect(lambda:caller.searchActivated(self.imgLineEdit.text()))
```

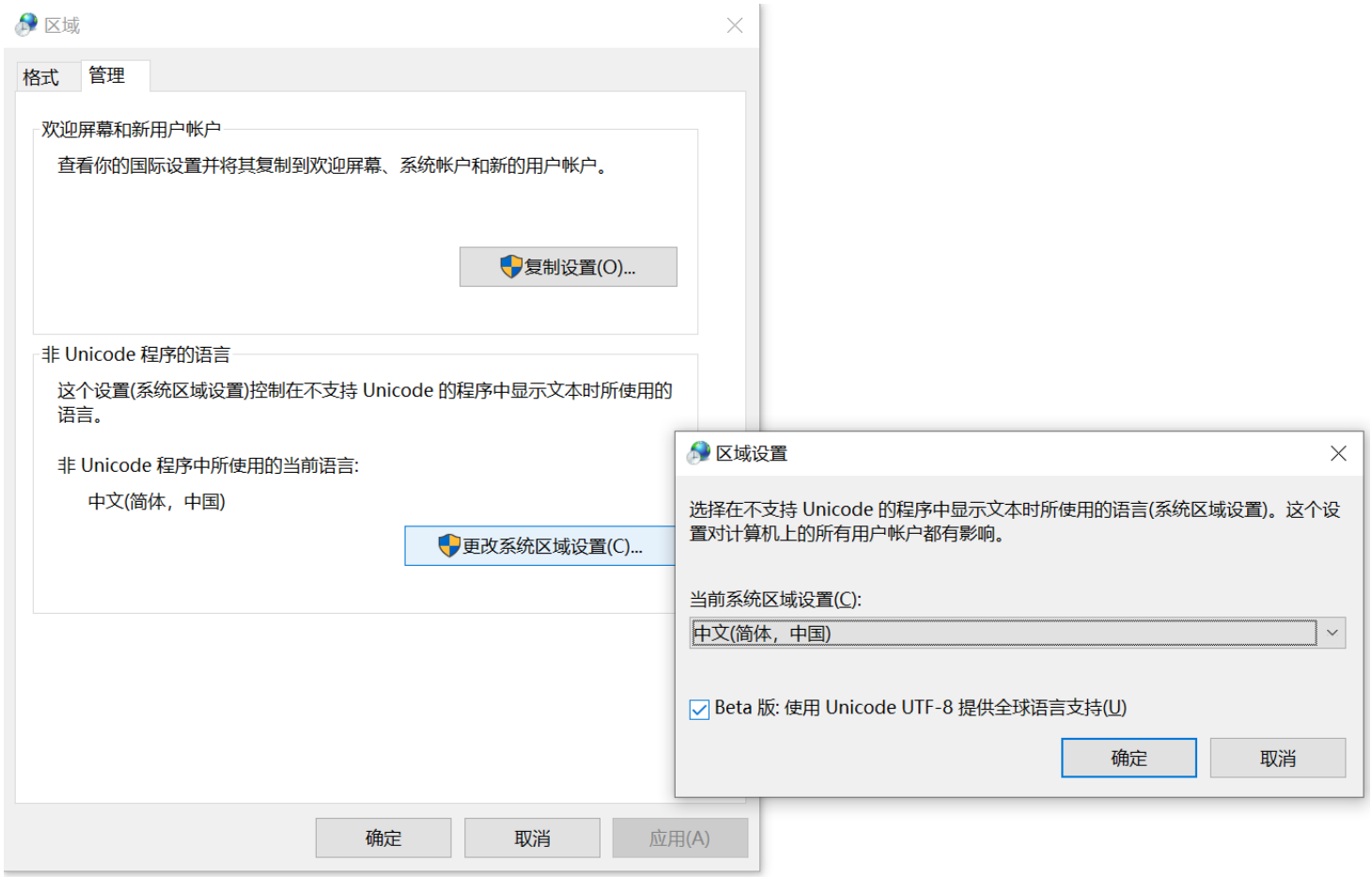
- 自定义图的实现：
  考虑到读取数据库所需要的时间受网络、服务器等因素影响很大，不如直接在本地读取效率高。因此包含大量数据的知识图谱节点、边信息以json文件保存于本地，命名以用户名区分，以此提高动态渲染知识图谱的效率。
  用户在自定义或修改图节点时，其节点信息会被子进程保存在两个暂存的json文件(nodes_data.json, links_data.json) 中，待主进程退出时会将该暂存文件的内容写至该用户自己的json文件(username_nodes.json, username_links_json)中。该用户下次登录时就会用到这两个自己的文件，以复现自定义内容。主进程在结束时的操作是通过重写closeEvent()方法实现的：

```
def closeEvent(self, event):
    app_path = os.path.abspath(os.path.dirname(__file__))
    if self.name:
        name_nodes_path = os.path.join(app_path, 'templates', self.name + '_nodes.json')
        name_links_path = os.path.join(app_path, 'templates', self.name + '_links.json')
        with open('templates/nodes_data.json', encoding='utf-8') as file:
            nodes_code = file.read()
        ...
```

四、项目运行过程

需要安装的库：py2neo、openpyxl、PyQt5、QtWebEngineWidgets、selenium、pymysql

运行 start.ps1 即可(若中文出现乱码，请修改编码格式为 UTF-8，win10操作见下)。



五、项目总结

1. 团队开发

为了进行团队共同开发，我们将项目部署在Github上进行管理：Steel-Shadow/Summer-Python-Course-Final-Project: Course Resource Platform Recommendation (github.com)。小组分别在本地进行开发，使用 git 进行版本管理，在 Github 上进行进度同步。此外，我们进行了相对明确的分工，以便各自完成所需要提供的接口，能够很好地对接各自的任务，大大提高了开发效率。

2. 前后端设计

为了完成知识图谱的交互功能，我们选择了前后端框架的模式制作知识图谱。由于第一次接触前后端，所以预先进行了Echarts、Html、css、JavaScript、Flask的学习。在前端方面，由于时间限制，学习的内容并不充分，因此制作出来的前端只能说差强人意。学习相关知识后，我们选择以力导向图的形式呈现知识图谱，可以避免节点之间的重叠，并遵循任务要求依次完成了点击课程显示节点、CRUD、书签三大功能，并对界面进行了一定的美化；在后端方面，我们选择了轻量级的Flask框架，对接包括管理数据库、爬取课程、呈现课程等功能接口。

前后端知识的学习+框架的制作是一个很艰难的过程，使用Echarts绘制图谱是最艰难的过程（前期竟然不知道用模板，完全手搓），除此还需要写css设计元素，使用JavaScript写相应的脚本实现前后端交

互。总的来说，虽然最后的结果差强人意，但是上手设计，确实学到了不少知识。

3. 绘制 GUI

GUI的设计直接决定界面是否简洁、对用户友好。我们的GUI界面基于PyQt5设计。整个程序的入口为 `MainWindow.py` ，主要采用堆叠布局(QStackedLayout)，以实现初始窗口、登录窗口、注册窗口、知识图谱窗口之间的切换。由于要与前后端框架对接，经过相当长时间的摸索才实现了与app.py子进程的通信，完成了任务要求。但整个界面谈不上美观，只能说差强人意。

4. 任务对接

- 接口设计
  我们共设计了如下端口：

函数方法	功能
`Spiders.scraping`	爬虫，用于爬取课程
`add_in_neo4j`	数据库端增加节点
`update_in_neo4j`	数据库端批量更新节点
`fetch_data`	数据库端读取节点课程信息
`sort_by_search_key`	关键字排序
`sort_by_name`	课程名称排序
`sort_by_platform`	平台名称排序
`sort_by_college`	院校名称排序

- 前后端框架与GUI的对接
  最初，我们各自完成的页面各有自己的窗体结构及设计风格，如何将其融合进我们的主 GUI 界面成为一大难题。反思之后认为，最开始设计的时候，应该直接使用前后端框架设计大部分内容，而不是使用前后端设计知识图谱，GUI展示课程节点。
  为了将前后端框架嵌入GUI中，我们也费了很大的功夫。不仅需要解决html的渲染问题，还需要在GUI中嵌入后端程序域名和交互支持，由于渲染html和后端程序挂到服务器两个进程的不确定性，还需要解决父子进程的问题。最后我们使用QWebEngineView来渲染html格式文件，由app.py程序提供域名和交互支持。切换到该窗口时，由主进程使用Qt封装的QProcess类创建app.py子进程提供支持，并通过将其stdout和stderr连接到主进程的槽函数以实现进程间通信。再设置QWebEngineView的域名，渲染知识图谱界面。

六、课程学习总结

1、课程收获和难点分析

- 理论课与日常作业
  在本门课程中，我们通过理论课与日常作业得到了许多收获。课程作业的难度逐步增加，让我们逐渐熟悉了Python的基本语法和一些常用函数。前几次的代码作业涉及字符串处理等编程题，帮助我们快速掌握Python的基本知识。然而，后面的作业逐渐多出一些算法题，但这让我们更深入地了解了动态规划、深度优先搜索等基本算法，并从中受益匪浅。
- 大作业
  本次大作业对我们小组内所有人都是一个挑战。此前，我们并没有接触过爬虫、数据库、前后端框架、GUI设计；也没有做过面向用户的大工程。所以我们都是一边学习、一边设计，完成下来，收获不小。
  - Python语言理解加深：在项目中，我们更深入地学习了Python，了解了它作为解释型语言、胶水语言和面向对象语言的特点。除了使用Python的基本库外，我们还学习了与Python配套的其他服务库，如PyQt5和py2neo，这进一步展现了Python的强大兼容性，使得它可以在复杂项目中连接各个模块。
  - 加强面向对象意识：正好结束OO课不久，这次大作业又是对我们面向对象意识的磨砺。本次大作业是我们第一次团队开发（相对）大型项目。在过去的两年中，最大的代码工程是OO中千行左右的单元作业。而在本次大作业中，我们团队开发了近万行规模的代码，极大地训练了面向对象的工程能力与项目复杂度控制能力。通过大量运用面向对象的知识和技能，我们不仅实现了团队协同开发，达到了低耦合高内聚的目标，提高了项目的鲁棒性和可扩展性；还进一步加强了面向对象的意识。
  - 知识拓宽：由于任务需要，我们主动地去学习了前端、后端、数据库知识。在完成任务的过程中，又进一步加强了掌握。
- 难点：本次大作业最大的难点在于我们基础薄弱，缺乏相关知识面，框架设计方面有所不及。由于来不及学习vue，因此也很难找到适合的框架，容易被网络上大量重复且无用的“相关信息”耽搁时间。首先是前端的设计，需要利用有效时间学习相关知识，去绘制能够交互的知识图谱且实现与后端的交互。遇到的最大的问题也与框架设计有关，就是在GUI和前后端的对接时，由于没有使用专业的设计前后端的工具，所以相关资料甚少，在解决交互问题时出现了大量奇怪bug，非常耗费精力。

2、教师授课评价

老师上课过程中讲解思路清晰，鼓励引导学生思考，讲授内容兼具广度与深度，并通过课上代码演示更好地帮助同学们巩固课程知识点，比如在讲授Python GUI设计时，老师曾带领同学现场编写GUI中的各种组件，帮助同学们加深了对图形用户界面设计的理解。此外，老师非常关心同学们对授课效果的反馈，每节课结束时老师都会询问同学们讲解是否清楚。总体而言，老师的授课质量非常高，如果可以将重点从Python基础语法更多地转移到Python进阶技能上就更好了。

3、助教评价

助教们工作态度认真负责，能够非常耐心地解答同学们在课堂作业、课程考核等方面的问题，帮助同学们更高效地完成课程任务，不过在前几次课堂练习与作业中，存在少数题目描述不够清楚的问题，比如加不加空格或分号等。建议助教们将先前题目中存在的问题收集起来，完善题目要求与作答规范。

4、当前课程教授内容评价与课程进一步改进建议

本学期的Python程序设计课主要涵盖了Python基本语法、面向对象编程以及图形用户界面设计等知识点，授课内容全面、授课形式丰富，通过课堂讲解、随堂练习、课后作业，课程大作业等多个环节，引导同学们构建了从“夯实基础”到“实战应用”的学习桥梁。

下面是三条改进意见：

- 拓展课程深度，适当引入更多有关Web开发，数据库等开发技能
- 增加作业或考试次数，帮助同学们端正学习态度、巩固所学知识
- 在课件中推荐相关知识点的教材或其它课程资料，引导学生自主学习更多知识

七、主要参考资料

1. Selenium with Python
2. Qt
3. PyQt文档
4. IT项目网-PyQt5快速上手课程
5. Neo4j教程
6. Echarts使用手册
7. 尚硅谷前端html+css零基础教程