IT Alumni Database Milestone 2 Report IT 4983

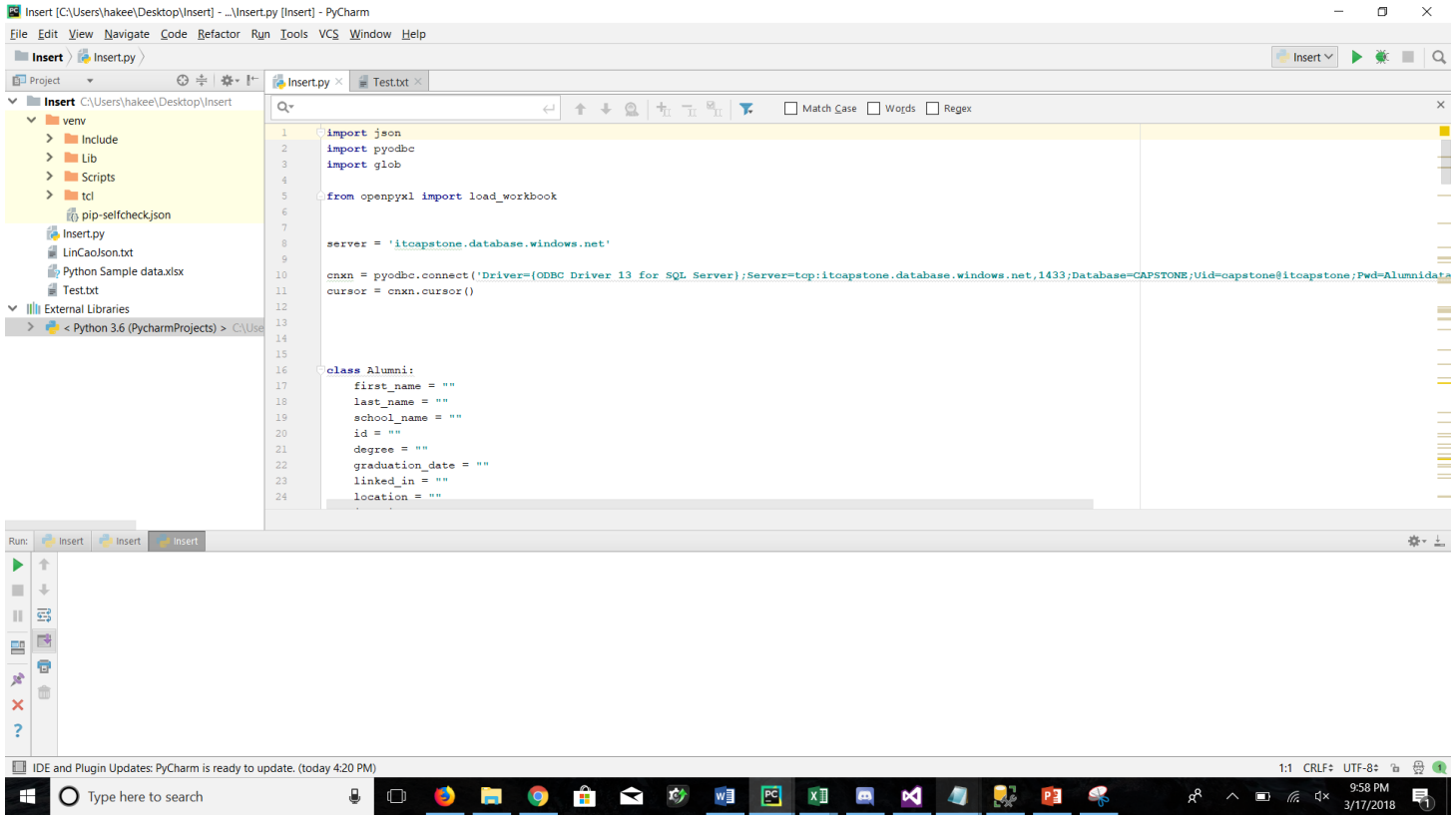
Ricky Parks

Vy Duong

Zack Downing
Desiree Smokes

Summary of Milestone 2 tasks

Task	Status
Connect to database using Python	Completed
Collect and parse data	Completed
Create logic to check if record exists	In progress
Work on logic to insert data	In progress
Testing the insertion logic	Completed

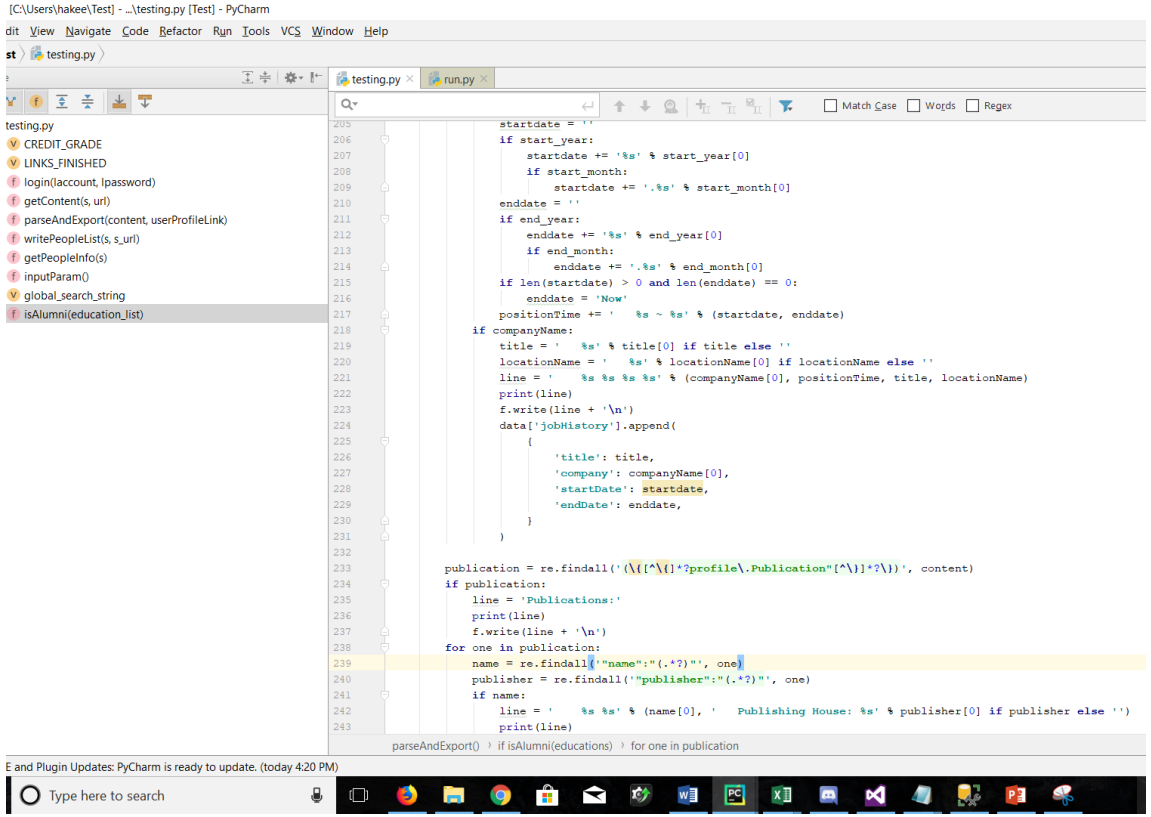


Connecting to database from Python Script

Collecting and parsing data

- Original Web crawler outputs data into .txt files
- Had to be modified to use JSON
- We were originally going to use Excel, but we found JSON made it easier to read the data
- Since Python naturally understands Dictionary objects, this was much simpler and faster

Collecting and parsing data (continued)



- Highlighted section shows the web crawler modified to use JSON

Checking if record exists before insert

- Currently the primary key attributes in each table prevents duplicate data
- It’s still being determined as to whether or not we need this functionality

Inserting data into database from web crawler files

Insert [C:\Users\hakee\Desktop\Insert] - ...\Insert.py [Insert] - PyCharm

File Edit View Navigate Code Refactor Run Tools VCS Window Help

Insert 〉 Insert.py 〉

Insert.py | Test.txt

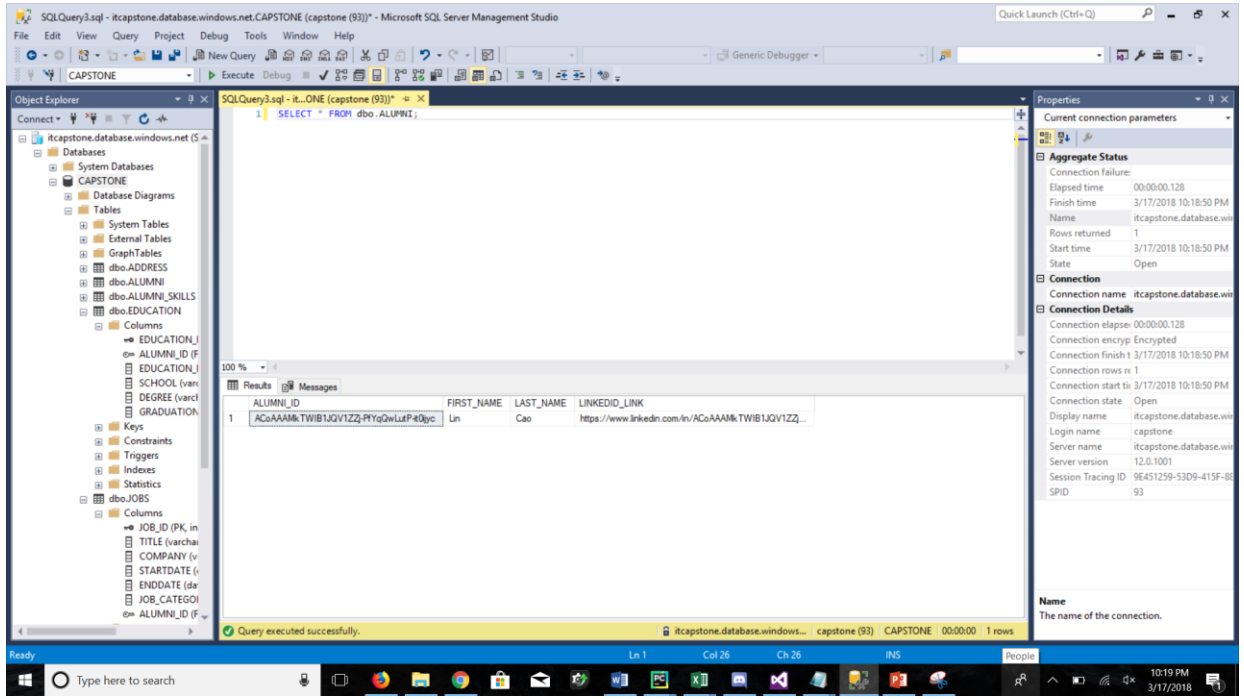
```
AlumniList = []
#Get alumni information for every file
for filename in glob.glob('*json.txt'):
    education_list = []
    job_list = []
    with open(filename.title()) as json_file:
        data = json.load(json_file)
        for fName in data['nameUrlId']:
            first_name = fName['firstName']
        for lName in data['nameUrlId']:
            last_name = lName['lastName']
        for alumniId in data['nameUrlId']:
            id = alumniId['id']
        for url in data['nameUrlId']:
            linked_in = url['linkedInUrl']
        for education in data['education']:
            degree = education['degree']
            field = education['field']
            schoolName = education['schoolName']
            graduation_date = education['endDate']
            education_object = Education(degree,field,schoolName,graduation_date)
            education_list.append(education_object)
        for position in data['jobHistory']:
            title = position['title']
            company = position['company']
            startDate = position['startDate']
            endDate = position['endDate']
            job_object = Job(title,company,startDate,endDate)
            job_list.append(job_object)
        json_file.close()
    AlumniList.append(Alumni(first_name,last_name,id,linked_in,education_list,job_list))


for alumni in AlumniList:
    print(alumni.first_name)
    cursor.execute("Insert Into dbo.Alumni(alumni_id,first_name,last_name,linkedid_link) " +
                   "Values(" + "\'" + alumni.id + "\'" + ","
                             "\'" + alumni.first_name + "\'" + ","
                             "\'" + alumni.last_name + "\'" + ","
```

IDE and Plugin Updates: PyCharm is ready to update. (today 4:20 PM)

1:1 CRLF UTF-8

Type here to search

10:14 PM 3/17/2018



Data from Alumni table after being inserted from Python

Lessons learned

- We learned a lot about the Python language and libraries
- Got more familiar with parsing using JSON
- Learned that we have to come up with an optimal rate for running the web crawler to avoid sending out too many page requests
- We may have to use a premium business LinkedIn account to be granted a higher number of monthly searches on the LinkedIn

Plan going Forward

- Research the optimal number of page requests to send to LinkedIn
- Determine the best way to update the database using the LinkedIn urls
- Begin documenting all of the scripts involved
- Analyze the scripts to see if they can be made to run faster