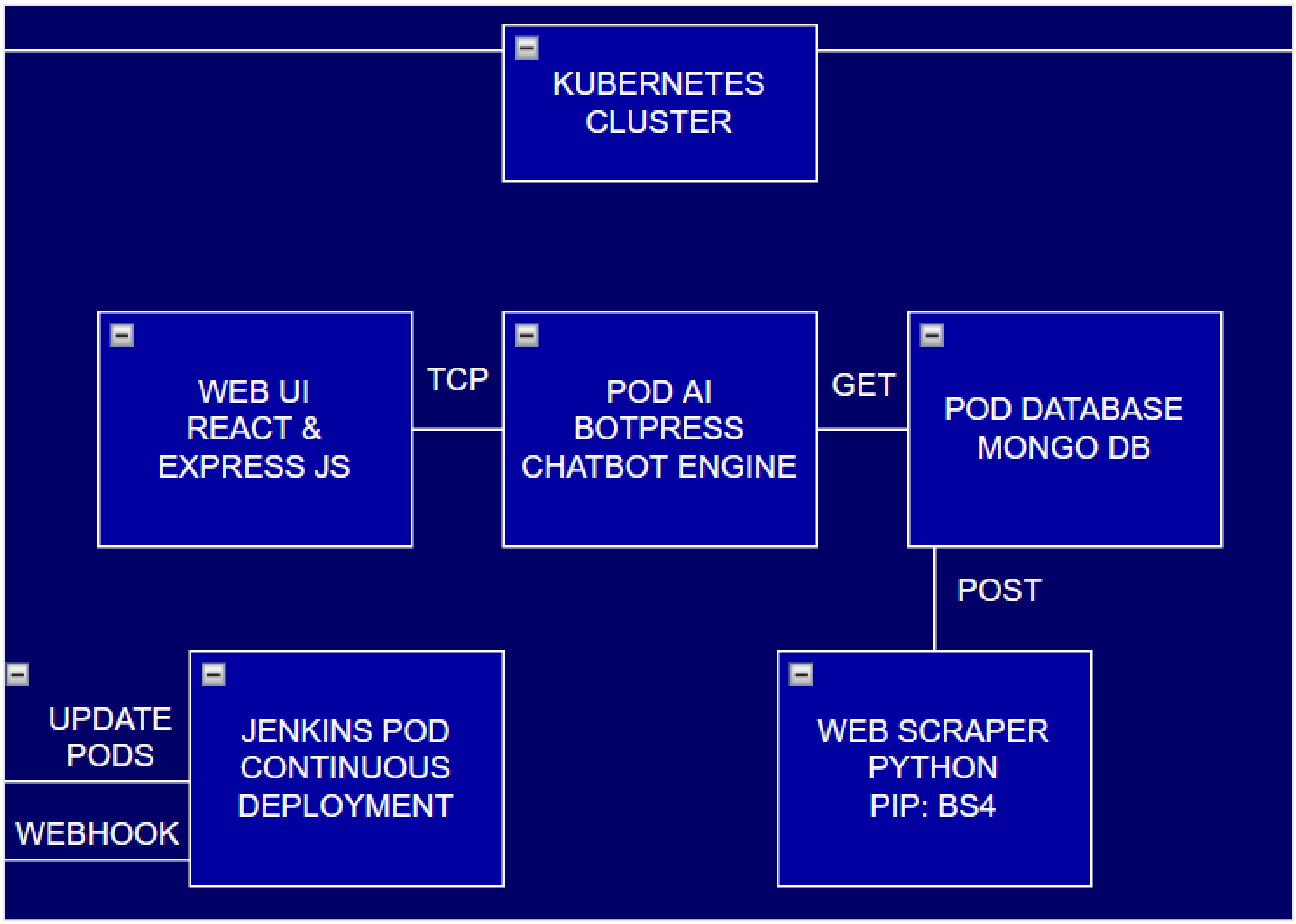
Team Name: CloudMen

CSC 468-01

WCU RAMBOT

Members: Kane Deiley, Daniel Muth, Calvin Miner, Peter McClelland, Ridha AlHamadani

Chapter 1


KUBERNETES CLUSTER
WEB UI REACT & EXPRESS JS
TCP
POD AI BOTPRESS CHATBOT ENGINE
GET
POD DATABASE MONGO DB
POST
UPDATE PODS
WEBHOOK
JENKINS POD CONTINUOUS DEPLOYMENT
WEB SCRAPER PYTHON PIP: BS4

Chapter 2

This project intends to allow for West Chester students to use a Q&A bot to answer any questions that they might have about the university.

The WCU Q&A Bot will be bit using a Kubernetes cluster which will provide the service with resource allocation and load balancing. To provide the cluster with continuous integration and deployment we will intend to use a Jenkins server.

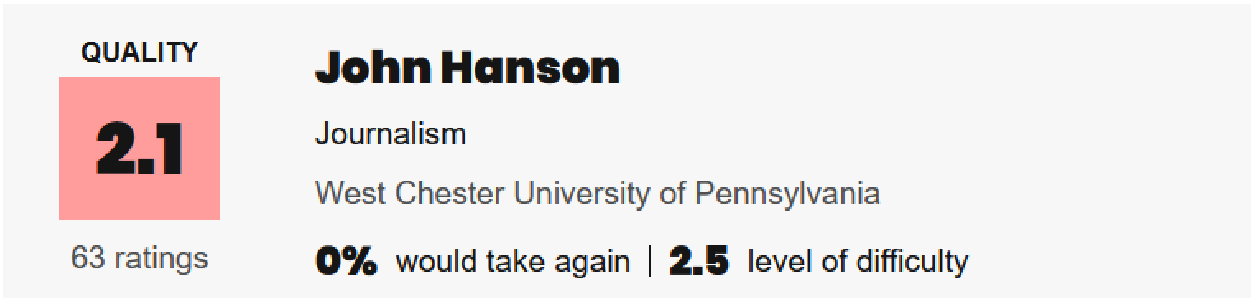
The Project will utilize Node.JS along with the Web App frontend server React and the backend server react to provide the user with a full web user interface. We will use and open source AI Bot called BotPress to provide our user with a fully developed but that aids in providing responses to user questions.

The Project will utilize a Mongo DB database to store our information about the university, to which we will have http post and gets coming from the Node JS frontend and backends. The project utilizes and open source web scrapper to retain this information about West Chester.

Chapter 3:

We worked on creating a web UI for our chatbot application using React. We started by setting up a new React project and designing the UI using CSS. The frontend displays text input box where the user can type into a chat window including conversation history as well as a "clear chat" feature, where the user can type "/clear" to clear the chat history. The WebUI is containerized using Docker, there are separate Dockerfiles for the frontend and backend, where the frontend image runs on port 3000 and communicates with the backend running on port 5005. Integration of our Rasa chatbot into the web UI by setting up a webhook that connects to our Rasa server. This will allow our web UI to send user queries to the Rasa server and receive responses back in real-time. We will implement this by adding a new endpoint to our backend that sends user queries to the Rasa server using the Rasa API, and then returns the response back to the frontend for display in the chat window.

Our database is Container has been constructed using MySQL in Docker. In regards to setting up the database with a dockerfile, the file is still basic and needs to include more information to fully support our goals regarding the webscraper, but there are yaml files which help with setting up the volume and persistent storage via Kubernetes. Given which data we intend to pull from the Rate My Professor website which includes all teacher data on the following screenshot:


QUALITY

2.1

63 ratings

John Hanson

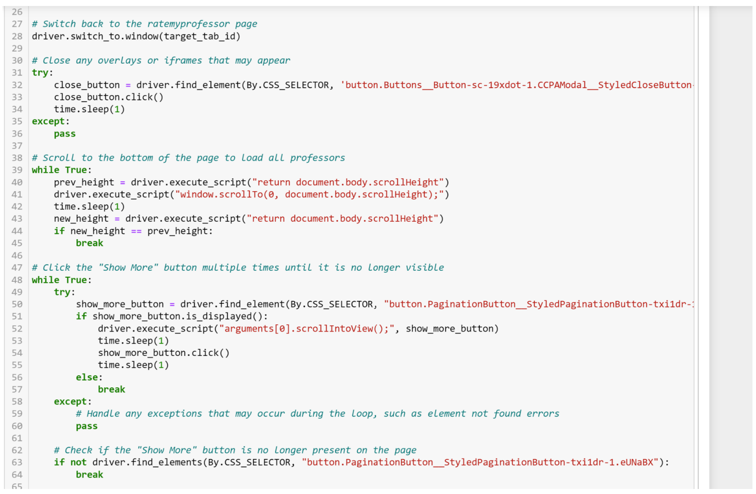
Journalism

West Chester University of Pennsylvania

0% would take again | 2.5 level of difficulty

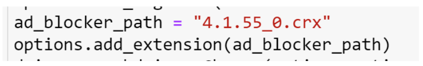
This schema has then been implemented to create test DB files to test when training our Rasa Chatbot. The port used to connect to the database container is 3306. Connection with the Webscraper container has been tested via docker by pinging the image created during our testing as well as basic docker compose files.

The webscraper was originally designed to be a continuously running program which gets new HTML data from Ratemyprofessor.com each time the program is ran within the excitable webscraper.py program and the proceeding Dockerfile when deployed as an image. We hit multiple walls within this course of action. The first problem that occurred was that Ratemyprofessor’s webpage was a dynamically designed webpage, meaning the HTML data of all professors from West Chester University would not be in the source code when the website was first visited. This roadblock forced us to implement Selenium, a python import, which creates an automated web driver to open up all professor while using certain CSS classes to click and navigate through the webpage shown below:



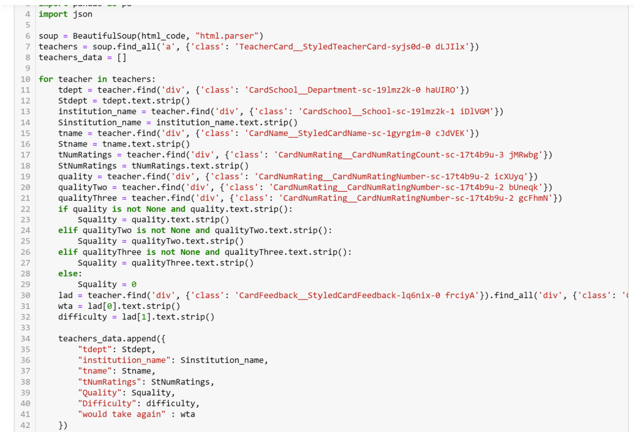
```python

# Switch back to the ratemyprofessor page
driver.switch_to.window(target_tab_id)

# Close any overlays or iframes that may appear
try:
    close_button = driver.find_element(By.CSS_SELECTOR, 'button.Buttons__Button-sc-19xdot-1.CCPAModal__StyledCloseButton
    close_button.click()
    time.sleep(1)
except:
    pass

# Scroll to the bottom of the page to load all professors
while True:
    prev_height = driver.execute_script("return document.body.scrollHeight")
    driver.execute_script("window.scrollTo(0, document.body.scrollHeight);")
    time.sleep(1)
    new_height = driver.execute_script("return document.body.scrollHeight")
    if new_height == prev_height:
        break

# Click the "Show More" button multiple times until it is no longer visible
while True:
    try:
        show_more_button = driver.find_element(By.CSS_SELECTOR, "button.PaginationButton__StyledPaginationButton-txi1dr-
        if show_more_button.is_displayed():
            driver.execute_script("arguments[0].scrollIntoView();", show_more_button)
            time.sleep(1)
            show_more_button.click()
            time.sleep(1)
        else:
            break
    except:
        # Handle any exceptions that may occur during the loop, such as element not found errors
        pass

    # Check if the "Show More" button is no longer present on the page
    if not driver.find_elements(By.CSS_SELECTOR, "button.PaginationButton__StyledPaginationButton-txi1dr-1.eUNaBX"):
        break
```

This code for the python-based webscraper would work, yet due to certain pop-up ads and display messages blocking given CSS buttons we realized that there was a requirement to add an adblocker into the webdriver if we wanted to guarantee success. This was a complicated task due to troubles of getting required .pem and .crx files for the webdriver, implementing them within the python code, and accessibilty/privacy problems while trying the use the files. We did get a working version with the an blocker and it worked smoothly getting the HTML code for us everytime:

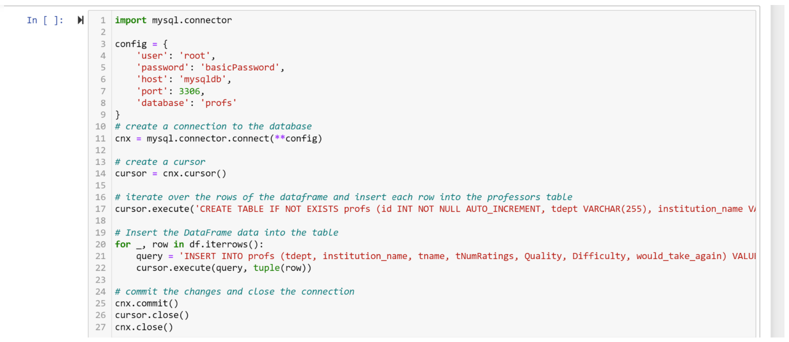


```python
ad_blocker_path = "4.1.55_0.crx"
options.add_extension(ad_blocker_path)
```

This solution worked and we believed we had completed the first step within completing the webscraper, but over the course of a few weeks after, the webscraper stopped working. This is when we realized that Ratemyprofessor changed the give CSS button class names periodically on what seems to be a week to week basis, so we were forced to move away from the idea of a consistent webscraper. We decided to get the information using the same code shown above, and save the HTML code as a file, which was then implemented into GitHub. This meant that our webscraper would just be searching through a given HTML file, getting the necessary data, formatting it, and sending it to MySQL:



```python
import json

soup = BeautifulSoup(html_code, "html.parser")
teachers = soup.find_all('a', {'class': 'TeacherCard__StyledTeacherCard-syjs0d-0 dLJIlx'})
teachers_data = []

for teacher in teachers:
    tdept = teacher.find('div', {'class': 'CardSchool__Department-sc-19lmz2k-0 haUIRO'})
    Stdept = tdept.text.strip()
    institution_name = teacher.find('div', {'class': 'CardSchool__School-sc-19lmz2k-1 iDlVGM'})
    Sinstitution_name = institution_name.text.strip()
    tname = teacher.find('div', {'class': 'CardName__StyledCardName-sc-1gyrgim-0 cJdVEK'})
    Stname = tname.text.strip()
    tNumRatings = teacher.find('div', {'class': 'CardNumRating__CardNumRatingCount-sc-17t4b9u-3 jMRwbg'})
    StNumRatings = tNumRatings.text.strip()
    quality = teacher.find('div', {'class': 'CardNumRating__CardNumRatingNumber-sc-17t4b9u-2 icXUyq'})
    qualityTwo = teacher.find('div', {'class': 'CardNumRating__CardNumRatingNumber-sc-17t4b9u-2 bUneqk'})
    qualityThree = teacher.find('div', {'class': 'CardNumRating__CardNumRatingNumber-sc-17t4b9u-2 gcFhmN'})
    if quality is not None and quality.text.strip():
        Squality = quality.text.strip()
    elif qualityTwo is not None and qualityTwo.text.strip():
        Squality = qualityTwo.text.strip()
    elif qualityThree is not None and qualityThree.text.strip():
        Squality = qualityThree.text.strip()
    else:
        Squality = 0
    lad = teacher.find('div', {'class': 'CardFeedback__StyledCardFeedback-lq6nix-0 frciyA'}).find_all('div', {'class': '
    wta = lad[0].text.strip()
    difficulty = lad[1].text.strip()

    teachers_data.append({
        "tdept": Stdept,
        "institutiion_name": Sinstitution_name,
        "tname": Stname,
        "tNumRatings": StNumRatings,
        "Quality": Squality,
        "Difficulty": difficulty,
        "would take again" : wta
    })
```



```python
In [ ]:
import mysql.connector

config = {
    'user': 'root',
    'password': 'basicPassword',
    'host': 'mysqldb',
    'port': 3306,
    'database': 'profs'
}
# create a connection to the database
cnx = mysql.connector.connect(**config)

# create a cursor
cursor = cnx.cursor()

# iterate over the rows of the dataframe and insert each row into the professors table
cursor.execute('CREATE TABLE IF NOT EXISTS profs (id INT NOT NULL AUTO_INCREMENT, tdept VARCHAR(255), institution_name V

# Insert the DataFrame data into the table
for _, row in df.iterrows():
    query = 'INSERT INTO profs (tdept, institution_name, tname, tNumRatings, Quality, Difficulty, would_take_again) VALU
    cursor.execute(query, tuple(row))

# commit the changes and close the connection
cnx.commit()
cursor.close()
cnx.close()
```

On the Rasa front, things have gone well. The Rasa chatbot was initially constructed via the command 'rasa init' and subsequently trained on the default dataset. This default chatbot was then dockerized via a Dockerfile in the actions folder and a Dockerfile and a Docker Compose file in the main folder. After that, a 'chatbot.yml' file was added for Kubernetes support. That done, we are now working on modifying the dataset so that it will actually give responses based on our desired data. This is in the extremely early stages at the moment, but the plan moving forward is to add a list of base questions to the NLU file, a matching set of stories to the stories file, and a set of actions that will allow the chatbot to directly query our SQL database for information.

Chapter 4:

Using Docker Network we were able to connect all test containers. This has been done with containers for MySQL, Python, Rasa and our WebUI. However we have not gotten around to testing this with our current project version. We figure our preliminary connection will allow us to retrofit this to the project rather easily. The container communication that we feel will be vital to our success are between the Webscraper Container and the Database Container, then the Database Container to the ChatBot, and finally a persistent and reliable connection between our ChatBot and our WebUI.

Because of the troubles found within the webscraper process, multiple attempts were made to create a working container for the webscraper and connect it to the given MySQL server. The first attempt was using a Selenium image, which we found running the webscraper as a python image would work best. We attempted to connected the webscraper while still in the process of using the continuous webscraping design, when we decided to change to a preset HTML file, the need for the Selenium image, .crx, and .pem was relieved. This made the process of creating the webscraper into a Job on Kubernetes much easier.

For the chatbot initial progress was promising, the intents and stories portion of the data set was quickly assembled however when it came time to make the actions that would call the sql database a scope problem arose. With the audible actions the various data points could not be directly linked to each other as a group, an anchor data point had to be chosen, which dramatically limited the variety of questions the chatbot would recognize. Surpassing this limit would require either a dramatically more complex set of training data and a staggering number of actions, or a new data paradigm. After a dive into rasa's document an extremely promising option was found, knowledge base actions. With a KB all the data points of each professor could be linked together in a referenceable group dramatically increasing the flexibility of the chatbot's viable questions and answers, if we could get it to work. Unfortunately despite getting a viable model and local server running the knowledge base remained non-functional with no error messages and limited resources online debugging proved fruitless and the KB was eventually abandoned and the original sql Query based actions where uncommented, the teacher's names were chosen as the anchor point and after a test to ensure it built correctly as both a chatbot and docker image the chatbot was uploaded to github so that the sql queries could be properly integrated to the database

Chapter 5:

Final status of the project - partially complete. Unfortunately our team was unable to finish our project to our intended standards. However, most of original vision was attained by our final product. Despite running into issues dynamically scraping our professor information, we were able to provide an adequate solution by scrapping our selenium method simply returning to beautiful soup.

With this it became a simple solution when storing our professor information in a single database, and in a single table which retained the same schema as above. The database and web scraping components were the first two that were successfully deployed on kubernetes.

The next issue became our familiarity with the chatbot that rasa provides. We underestimated how intricate a knowledgebase solution would be, especially pulling this knowledge directly from the database. This pitfall will be versioned and worked on further, our test code for this version falls in the knowledgebasebot directory of our app directory. When then reverted to the chatbot directory which is capable of completing simple actions. These actions query our database for each of the schema mentioned above. The chatbot became out third component which was added to our kubernetes deployment.

Our Web-UI utilized react js which initially was a pain. None of our members had experience with react, at least until member Ridha AlHamadani joined. Ridha quickly built a simple receive and send interface which is displayed by our poster in our Read Me. However, we were unable to accurately deploy the WebUI such that it was capable of sending and receiving responses from the shell of our rasa chatbot.

Although we consider our WebUI implementation a failure, we are however able to deploy the bot and access it's shell which at a minimum is capable of providing us some receive and response actions. We feel this is a justifiable proof of concept which would allow for us to continue our efforts and make the application much more functional.

Team Resumes:

Kane Deiley

(267) 374-6525 | | Kpatrickdeiley@gmail.com | 22 Hollow Horn Road, Erwinna, PA 18920

EDUCATION

West Chester University of Pennsylvania, West Chester PA

- Bachelor of Science in Computer Science - May 2023
- Minor in Applied Statistics
- GPA: 3.89
- Student President of Upsilon Pi Epsilon Honor Society

RELEVANT COURSEWORK

Project: Developed a small web application using GitHub, JavaScript, HTML, CSS and Mongoose
Coursework: Data Structures | Software Engineering |Software Testing | Applied Statistics | Computer Systems |Computer Security and Ethics | Data Communications and Networking

WORK EXPERIENCE

St. Luke's University Health Network, Allentown, PA
Student IT Intern – May 2022 – Present

- Excelled within a Project Management Office helping strengthen and automate procedures
- Utilized applications such as Power BI and Project to relay pertinent information to a team
- Handled vague and complex requirements in reporting and reduced them into a desired product

Giant Food Stores, Plumsteadville, PA
Front End Associate - April 2021 - August 2021

- Initiative shown through resolving customer issues which manifested in various circumstances
- Promotion gained within two months of work, due to exceptional customer service

VOLUNTEERING & LEADERSHIP

Upsilon Pi Epsilon Honor Society – May 2021 - Present

- As Student President of the society, organized meetings and helped with outreach

Athletes Helping Athletes – August 2017 – June 2019

- Helped the less fortunate participate in school sports
- Presented AHA award to those chosen as the winner of the subsequent years

SKILLS

- Proficiency in Object-Oriented Languages such as Java, Python, C
- Experience with Power BI and other languages which include R, Haskell and DAX
- Minor Web Development experience using Node.js, JavaScript, HTML and CSS

Daniel Muth

As a soon to be Computer Science major, dedicated and eager to explore more of the field, sharing my knowledge, and gaining even more valuable knowledge, I am looking forward to joining a team and contributing to the company.

Work Experience

Giant Food Stores – Front End Lead

April 2017 – Current — Plumsteadville, PA and West Chester, PA

- 30-40 hours a week
- Lead team of over 15-20 people per day directing cashiers, cash maintenance, and sales
- Experience in all departments of company including grocery, dairy, front end, and produce
- Leadership, time management, and communication skills required and implemented every day

A&S Performance – Landscaper

June – August, 2018-2019 — Ottsville, PA

- 40 house a week
- Contributor to a small team of 5 people, with direct communication to CEO of company
- Teamwork and good communication skills showed

Projects

Poker Simulation – Creator

- Developed java poker game, which simulates outcomes of randomized poker games, where the user players against a simulated and randomized opponent.

Cloud WCU Student Chatbot – Contributor

- Ongoing project, which helps students gain easy access to information about professors for West Chester University, through a chatbot created through the cloud.

Education

West Chester University – B.S., Computer Science

2019 – Current

- Applied Statistics Minor
- Business Analytics Minor
- Computer Security Certificate (
- Current GPA: 3.4

Skills

- Advanced: Java; Python; MS Word; MS Excel

- Familiar: SQL; Haskell; C; C++; MySQL; MS Access; HTML;

Peter McClelland

PeterMcClelland0@gmail.com | (484) 364-1893 | 135 Kaiser Drive, Downingtown, PA 19335

Education

West Chester University of Pennsylvania, West Chester PA

Bachelor of Science in Computer Science and Minor in Applied Statistics, May 2023

- Computer Security Certification
- Current cumulative GPA: 3.4

Relevant Coursework

- Computer Security & Ethics, Data Structures & Algorithms, Computer Systems, Applied Statistics, Intro Statistical Computing, Big Data Engineering, Software Security, Software Engineering, Topics in Complex Systems, Computer Security, Modern Malware Analysis, and Introduction to Cloud Computing

Skills

- Experienced in Java, C, assembly/machine language, SAS, R, Python, and Jupyter Notebook, Pyspark

Extracurricular Activities

Tau Kappa Epsilon, West Chester PA

- Hypophetes (Chaplain) (May 2021-May 2022)
- Audited over 70 students to insure high academic standards and maintained it through the year
- Planned and created quality events with multiple organizations
- Help plan and fundraise $18,300 this year so far for multiple philanthropies like St. Jude

Employment

West Bradford Township, Downingtown PA (June 2019- 2022)

- Coordinated projects with manager to get them done in an efficient and timely manner
- Worked with team members to assign duties and tasks for projects
- Took citizens inquiry to resolve their concerns
- Monitored facilities and machinery for problems

Calvin M Miner

1 Crumley Ave Malvern, PA, 19355 | 623-640-8875 | calvin.miner.2017@gmail.com

Education

Delaware County Community College
Enrolled - November 2019 Graduated – May 2021

Associate in Science - Computer Science – Honors

West Chester University
Enrolled – June 2021 Expected to Graduate – May Spring 2023 Computer Science BS

Employment History

Giant Food, West Chester, Pennsylvania

Service Associate: December 2017 – June 2020

Starbucks, Malvern Pennsylvania

Barista: June 2020 - Present

TestOut Network pro certification

- Certified 12/13/2018.
  ○ Cables and connectors ○ Wired networking
○ Wireless networking ○ Network connection
configuration ○ Network services ○ Network security ○
Network troubleshooting School work:

https://github.com/CalvinMMiner/Dccc_school_work.git
- A repository holding all the miscellaneous projects I made while at Delaware county community college.

https://github.com/CalvinMMiner/WCUPA_School_Work.git
- A repository holding a significant chunk of my java projects from WCUPA.

Notable Classes or activities:
- CSC 496: Topics in Complex Systems (Edge Computing)
- CSC 471: Modern Malware Analysis
- CSC 321: Data Base Mgt Syst
- Member of the WCU Computer Science club
- Phi Theta Kappa Honor Society

Coding Languages

- Capable of writing code with java, python, C++, and C
- Currently learning Haskell

Ridha AlHamadani

484-655-3646 | ridhazaki25@gmail.com | Coatesville, PA

Education

West Chester University Jan 2018 – Dec 2023 (expected)

Bachelor of Computer Science Major GPA 3.83

- Member of Phi Sigma Pi Honor Fraternity

Skills

Languages: Java, Python, HTML/CSS, C, Swift
Technologies: Docker, Linux, Git, Excel
Developer Tools: Visual Studio Code, Xcode

Certificates

Java	Codecademy
HTML/CSS	FreeCodeCamp

Work History

Marine Corps Reserves
Baltimore, MD

Team Leader Feb 2019 - Current

- Holds security clearance.
- Lead, trained, and mentored a team of Marines to achieve mission success.
- Established and maintained a positive command climate that fosters teamwork, accountability, and respect.
- Planned and executed training events to improve individual and team proficiency.
- Managed team resources, including personnel, equipment, and supplies.

Best Buy
Downingtown, PA

Warehouse/Inventory Worker Aug 2018 – Jan 2019

- Received and processed incoming merchandise in a timely and accurate manner.
- Assisted with inventory planning and forecasting to ensure adequate stock levels.
- Ensured the accuracy of inventory by performing regular cycle counts and audits.
- Organized and maintained inventory levels to ensure the availability of products for sale.
- Operated material handlings equipment, such as forklifts and pallet jacks, to move merchandise within the warehouse.