# Capstone Project

AllWinRx
Steven Lo (Data Engineer) , Yonathan Amare (Software Engineer), Adarsh Reddy (Project Manager)

## Elevator Pitch

We are helping small independent pharmacies calculate drug price comparisons by developing a comprehensive database to compare drug prices and using analytical tools to calculate potential discounts and profits from joining the All Win Rx GPO.

***ISBA Subfields***

**Storage and Data**

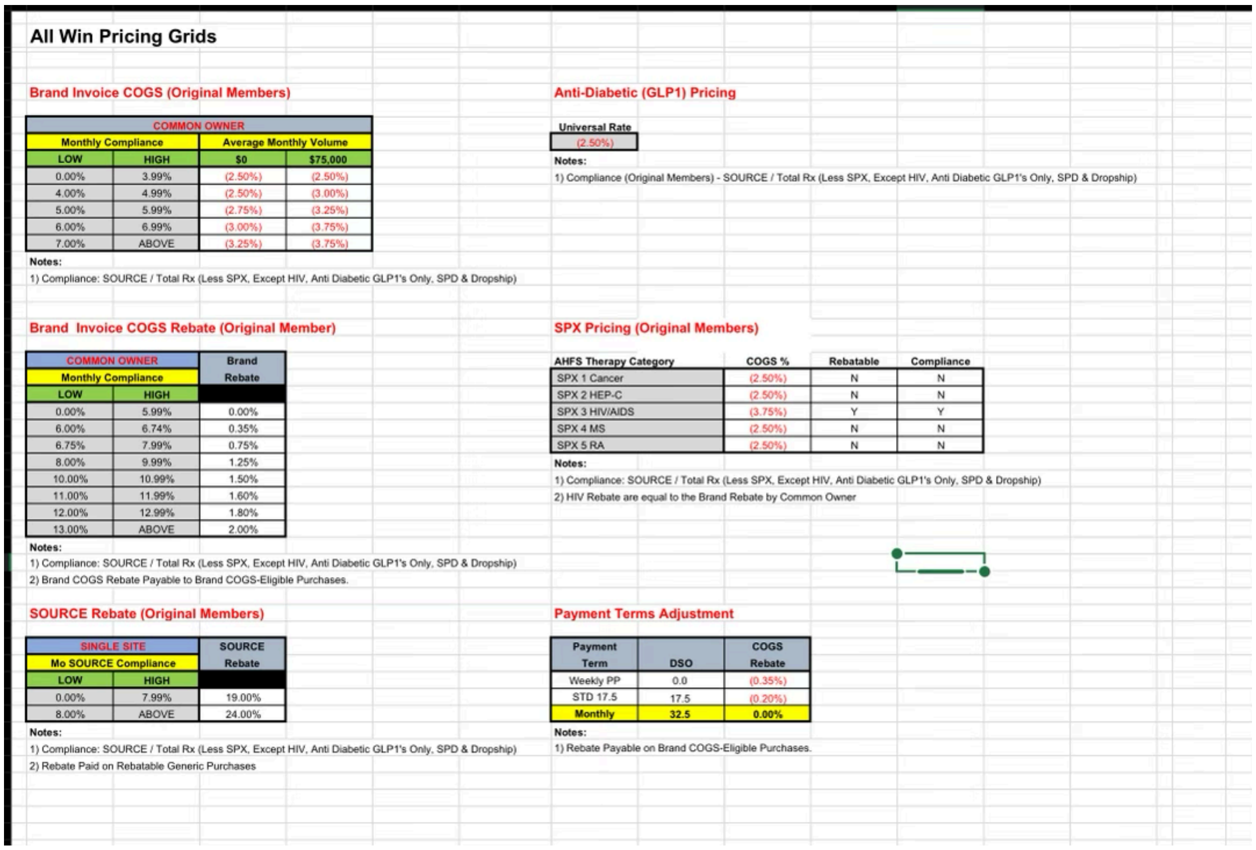
Storage and Data were a large part of our project. We had 2 data sheets, one was an invoice sheet from the Group Purchasing Organization (GP), and one was from the pharmacy that we were trying to create a cost analysis for. One of the issues we had was the data was not all there (Ex. Drugs pharmacy was not in invoice details doc.), so we had to gather more data from the procurement website. After the data was gathered we stored all said data and associated headers in a LMU Build SQL database that is hosted on the cloud.

**Software Development**

Software development was a large part of this project. When the data was missing, we realized that we had to go to the distributor's website and extract data from there. We had over 1000 missing data entries rather than individually, searching every single entry and scraping data, we would create a selenium driver. The selenium driver would extract things like National Drug Code (NDC),  Corporate  Identification Number (CIN, quantity price, and much more. We used python to create the driver, and then we had the driver dump all the information back into the database.

**Services and Infrastructure**

For services and infrastructure, we were given a pricing methodology from the GPO for us to incorporate in our database. This was given to us on Excel and we were supposed to interpret it and add this logic into our SQL database.



# All Win Pricing Grids

## Brand Invoice COGS (Original Members)

| COMMON OWNER | | | |
|---|---|---|---|
| Monthly Compliance | | Average Monthly Volume | |
| LOW | HIGH | $0 | $75,000 |
| 0.00% | 3.99% | (2.50%) | (2.50%) |
| 4.00% | 4.99% | (2.50%) | (3.00%) |
| 5.00% | 5.99% | (2.75%) | (3.25%) |
| 6.00% | 6.99% | (3.00%) | (3.75%) |
| 7.00% | ABOVE | (3.25%) | (3.75%) |

**Notes:**

1) Compliance: SOURCE / Total Rx (Less SPX, Except HIV, Anti Diabetic GLP1's Only, SPD & Dropship)

## Anti-Diabetic (GLP1) Pricing

| Universal Rate |
|---|
| (2.50%) |

**Notes:**

1) Compliance (Original Members) - SOURCE / Total Rx (Less SPX, Except HIV, Anti Diabetic GLP1's Only, SPD & Dropship)

## Brand Invoice COGS Rebate (Original Member)

| COMMON OWNER | | Brand |
|---|---|---|
| Monthly Compliance | | Rebate |
| LOW | HIGH | |
| 0.00% | 5.99% | 0.00% |
| 6.00% | 6.74% | 0.35% |
| 6.75% | 7.99% | 0.75% |
| 8.00% | 9.99% | 1.25% |
| 10.00% | 10.99% | 1.50% |
| 11.00% | 11.99% | 1.60% |
| 12.00% | 12.99% | 1.80% |
| 13.00% | ABOVE | 2.00% |

**Notes:**

1) Compliance: SOURCE / Total Rx (Less SPX, Except HIV, Anti Diabetic GLP1's Only, SPD & Dropship)
2) Brand COGS Rebate Payable to Brand COGS-Eligible Purchases.

## SPX Pricing (Original Members)

| AHFS Therapy Category | COGS % | Rebatable | Compliance |
|---|---|---|---|
| SPX 1 Cancer | (2.50%) | N | N |
| SPX 2 HEP-C | (2.50%) | N | N |
| SPX 3 HIV/AIDS | (3.75%) | Y | Y |
| SPX 4 MS | (2.50%) | N | N |
| SPX 5 RA | (2.50%) | N | N |

**Notes:**

1) Compliance: SOURCE / Total Rx (Less SPX, Except HIV, Anti Diabetic GLP1's Only, SPD & Dropship)
2) HIV Rebate are equal to the Brand Rebate by Common Owner

## SOURCE Rebate (Original Members)

| SINGLE SITE | | SOURCE |
|---|---|---|
| Mo SOURCE Compliance | | Rebate |
| LOW | HIGH | |
| 0.00% | 7.99% | 19.00% |
| 8.00% | ABOVE | 24.00% |

**Notes:**

1) Compliance: SOURCE / Total Rx (Less SPX, Except HIV, Anti Diabetic GLP1's Only, SPD & Dropship)
2) Rebate Paid on Rebatable Generic Purchases

## Payment Terms Adjustment

| Payment Term | DSO | COGS Rebate |
|---|---|---|
| Weekly PP | 0.0 | (0.35%) |
| STD 17.5 | 17.5 | (0.20%) |
| Monthly | 32.5 | 0.00% |

**Notes:**

1) Rebate Payable on Brand COGS-Eligible Purchases.

***Problem***

Our project addresses a pressing problem faced by small or independent pharmacies: the high cost of medicines due to a lack of bargaining power and discount opportunities. To address this issue, we worked with pharmacies to help them quantify the potential savings from joining a Win Group Purchasing Organization (GPO).

Our core solution is to calculate the discounts pharmacies can obtain through GPOs. After joining the All Win GPO, there will be different rebates. For example, purchasing a higher percentage of a generic drug will result in a larger discount. With our help, they can accurately calculate the costs they can save, and more importantly, they can have cases and information to convince more members or pharmacies to join.

Before we got involved, these pharmacies had no specific way to evaluate discounts on specific drugs; their only tool was Excel, which they used to track the discounted prices they received. This basic approach cannot effectively analyze or predict future savings.

The current financial resources of the entire project are free because we store the data on the lmubuild, but if they want to move to other platforms in the future, there will be different prices depending on the size of the data.The project took approximately 80 hours to develop and implement. It's a challenging but rewarding process that greatly enhances small pharmacies' profitability by providing them with the necessary tools and data to make strategic purchasing decisions.This project addresses two critical issues for small or independent pharmacies. The first challenge is creating a reliable database to store the prices and details of all drugs, essential for future growth and development. The second challenge is calculating the remaining funds or profits after drug purchases and discounts. This financial assessment is crucial as it directly impacts the pharmacies' operational efficiency.

## ***Solution***

The challenge we tackled was to assist pharmacies in assessing the potential discounts they could receive after joining the All Win GPO. This multifaceted project was methodically divided into several steps to ensure a comprehensive solution. The initial phase involved thoroughly understanding the stakeholders' needs.

To achieve this, we organized several meetings with them, where we not only listened and gathered their requirements but also actively engaged in discussions to clarify their expectations. Based on these interactions, we decided to construct a robust database that would serve as the backbone of our solution, storing essential data such as drug prices and other pertinent information.

The second is to build our database. There are five different tables in our database. The MasterSheet stores the prices and data of all medicines. Customer_Data records the customers' purchase history over the past three months. The remaining three tables manage rebates, which represent the discounts.

The third step involved building a web scraper because we lacked information on all drugs and their prices. We needed to visit the Capitol Drug website to collect their pricing data and store it in our database. This process was split into two parts: data extraction and loading the data into the database.

The final step was to use SQL queries to compare the data and calculate the potential profit from the All Win GPO rebates based on the drugs listed in Customer_Data.

1. **Programming Languages**
    a. Python, SQL, HTML
    b. Key packages: mysql.connector, selenium, time, pandas
2. **Data Storage**
    a. Database: MySQL,
    b. Object storage: lmu build
3. **Software/tools**
    a. Open-source software
        i. DBeaver, Visual Studio Code
    b. Paid software
        i. Excel, Zoom
1. **Hosting**
    a. Cloud
        i. Lmu.build free for students and faculty
2. **Data**
    a. From company website and Invoice
    b. Source
        i. Websites scraped
        ii. Client provided

**Solution in Technical Terms**

Since this was mainly a data analytics capstone project, the methodology we chose to use was the CRISP-DM method which stands for “Cross-Industry Standard Process for Data Mining.” The process consists of six stages that we will go over. We first had to understand the problem and create a cost-analysis solution for pharmacies to compare their current Cost of Goods Sold versus what we can provide. Understand the data, understand what the different data meant and what we needed to be successful. Next, we had to model our data by creating a MySQL database through LMU Build. Once we had that, we tested and evaluated it by using Selenium webdriver to extract data with missing values from the pharmacy website, and update the database with those new values. Lastly, we deployed our code through GitHub and utilized terminal commands to have the most recent and past versions of our code. We also utilized task management tools such as Jira and Confluence to track progress and keep our stakeholders and professors up-to-date.

This ultimately led us to the first step in our solution which was using Chromedriver and Selenium to extract the missing drug information from the pharmacy website and fill it into the database.I will now run you through a technical overview of the webdriver and how we implemented it. Once we run the python code, our stakeholder would be automatically logged in to the pharmacy website. Then, different CIN’s (or Corporate Identification Numbers) would be autofilled into the search bar, running through each drug and displaying their information. Based on the missing columns in the database, the webdriver would grab that information from each page and fill it back into the database. The way we implemented this function was mainly through web scraping; grabbing the specific ids of each part of the website and connecting that to SQL based on a specific condition.

**Next Steps/Future Improvements (½ page minimum)**

The data we scraped was from a website called Cardinal Health (Supplier). The Cardinal Health website is effective, but it does not give us a complete solution. The complete solution would be to either scrape off the AllWinRx(Group Purchasing Organization) website, kinray.com. Although the GPO uses Cardinal Health as a supplier, they have their own specific website. On this website there's a handful of other drug substitutes and we're supposed to be able to choose the cheapest one from here. The prices that we were given are actually from Capital Drugs (Pharmacy). The issue here is that Capital Drugs already has a negotiated price which is listed on Cardinal so if we want to continue to use these prices, the we need to add the negotiated rate (4.5%), and add it to the price we are seeing to get as close as we can to a non-negotiated price. The reason we used Cardinal Health instead of Kinray was because there was a multi-factor authenticator on the Kinray website. Everytime we wanted to access the date we would need to message the stakeholder, who had other priorities. The stakeholder was pleased with our progress, and shared that they will add me to the Kinray authenticator now to get the most accurate prices. The code we had is perfect, and the HTML elements will need to be changed if we go this route.

**Learning Resources (½ page minimum)**

We mainly utilized ChatGPT and our capstone advisor Professor Mukherjee to work on this project. We also utilized a few lessons from DataCamp's "Web Scraping In Python" course to implement XPath instead of CSS locator to extract element ids from Cardinal Health's website. We used these IDs to connect different parts of the website that we wanted to the mySQL database based on specific conditions from the columns with missing values.

# How to Automate a Manual Task Using Web Driver

This article shows you how to create a web driver, using selenium, to automate the process of clicking through different pages and bars.

## Instructions

1. **Install Selenium:** First, ensure you have Selenium installed. You can do this using
   a. pip: pip install selenium
2. **Download ChromeDriver:** You'll need to download ChromeDriver separately. Visit the ChromeDriver download page and download the appropriate version for your operating system. Extract the downloaded file.
3. **Import Required Libraries**: Import the necessary libraries at the beginning of your Python script:
   a. `from selenium import webdriver`
   b. `from selenium.webdriver.common.by import By`
   c. `from selenium.webdriver.support.ui import WebDriverWait`
   d. `from selenium.webdriver.support import expected_conditions as EC`
   `import time`
4. **Set Up Chrome WebDriver**: Set the path to your ChromeDriver executable. Replace `'/path/to/chromedriver'` with the actual path where you extracted ChromeDriver.
   a. `chromedriver_path = '/path/to/chromedriver'`
   b. `driver = webdriver.Chrome(executable_path=chromedriver_path)`
5. **Navigate to a Website**: Use `driver.get()` to open a website:
   a. `driver.get('` Example Domain `')`
6. **Interact with Web Elements**: You can interact with elements on the webpage using various methods

6. **Interact with Web Elements**: You can interact with elements on the webpage using various methods like `find_element_by_id()`, `send_keys()`, and `click()`:
    a. *Wait for the username element to be visible*
        i. `wait = WebDriverWait(driver, 30)`
        ii. `username_element = wait.until(EC.visibility_of_element_located((By.ID, 'okta-signin-username')))`
        iii. username_element.send_keys('username')
    b. *Find and Input Passwords*
        i. `password_element = driver.find_element(By.ID, 'okta-signin-password')`
        ii. `password_element.send_keys('password')`
        iii. `submit_button = driver.find_element(By.ID, 'okta-signin-submit')`
        iv. `submit_button = driver.find_element(By.ID, 'okta-signin-submit')`
        v. `submit_button.click()`
7. **Pause Execution (Optional)**: If you need to wait for certain actions to complete, you can add pauses using `time.sleep()`:
    a. `time.sleep(5) # Pauses for 5 seconds`
8. **Close the Browser Window**: Always close the browser window when done to free up system resources:
    a. `driver.quit()`

**Bibliography**

**-** Title: How Cyber Attacks Work

- Author: Techquickie

- URL:

[https://youtu.be/TdwwPP9oW4c?feature=shared](https://youtu.be/TdwwPP9oW4c?feature=shared)

- Publication Date: October 6, 2017

- Publisher: Linus Tech Tips

- Duration: 5 minutes 16 seconds

**ChatGPT:**

- Title: ChatGPT - AI Language Model

- Author: OpenAI

- Publication Date: 2021

- URL: [https://openai.com](https://openai.com)

- Description: ChatGPT is an AI language model developed by OpenAI, capable of engaging in conversation, providing information, and assisting users in various tasks.

## Challenges

During this project, we faced four major challenges: understanding the data, cleaning the data, developing a web scraper, and scheduling meetings with our team and stakeholders.
The first hurdle was understanding the data, which was heavily related to pharmacy operations and included many specific terms, such as CIN (Cardinal Item Number) and NDC (National Drug Code). Initially, the data was overwhelming, and we struggled to make sense of it. However, through several meetings, we gradually grasped the details and began to devise appropriate strategies.The second challenge was cleaning the data. A significant issue here was the incomplete drug pricing information; often, we only had the product name and number, making price comparisons impossible. This gap significantly hindered our analysis and required innovative solutions to address.The development of the web scraper was our most time-consuming task. None of us had prior experience with web scraping, and the website's requirement for a login, coupled with our need to avoid detection as bots, added layers of complexity to the task. Finally, scheduling meetings with the team and stakeholders proved exceptionally challenging. The stakeholders had limited availability, and our academic schedules often conflicted with potential meeting times. Early in the project, frequent meetings were essential to understand the data fully, demanding substantial time and effort from everyone involved.

These challenges were significant, but they provided valuable learning experiences and shaped our approach to project management and problem-solving.

## Lessons Learned

There were many lessons learned during this project, but we have narrowed them down to three. Firstly, we realized that fully understanding the business problem before we started the project would have helped us immensely. Capitol Drugs gave us Excel sheets with different pricing information and general drug information, and we immediately started to analyze and clean the data sheets without understanding what exactly the stakeholders wanted. This leads into our second lesson learned, which is to get on the same page with the stakeholders as soon as possible. The longer we waited and kept assuming what needed to be done, the more confusion would start to pile on us. And this confusion leads to the last lesson we learned: keep asking questions. Once we got in contact with Capitol Drugs, we learned that the owner wanted us to narrow down the amount of drugs in the database, fill in or delete missing information (this is where the webscraper plays its role), and implement rebate and discount logic. The only way we got to this conclusion was by continuously asking the pharmacy owner endless questions. There is no such thing as a bad question as long as you genuinely don’t know the answer to it, as it will only lead you closer to the solution. This project is still a work in progress, as we are trying to connect to another pharmacy website and extract information from there.

## Recommendations to Improve the Capstone Project Experience

Our recommendation would be to do weekly sprints. The reason we say weekly sprints is because although we were meeting weekly, our goals were changing almost on a weekly basis. We were trying to understand the project requirements, what the data meant, and what the stakeholders were requesting to meet larger deadlines and adjust accordingly on a weekly basis. So much was changing, but meeting with the professor to assess where we were at and what our goals were before the next meeting worked effectively for us. It was difficult to follow the Sprint schedule required by the project, because our Sprints were actually weekly. We did not think about what was going to be done in a month but what was going to be done next week.