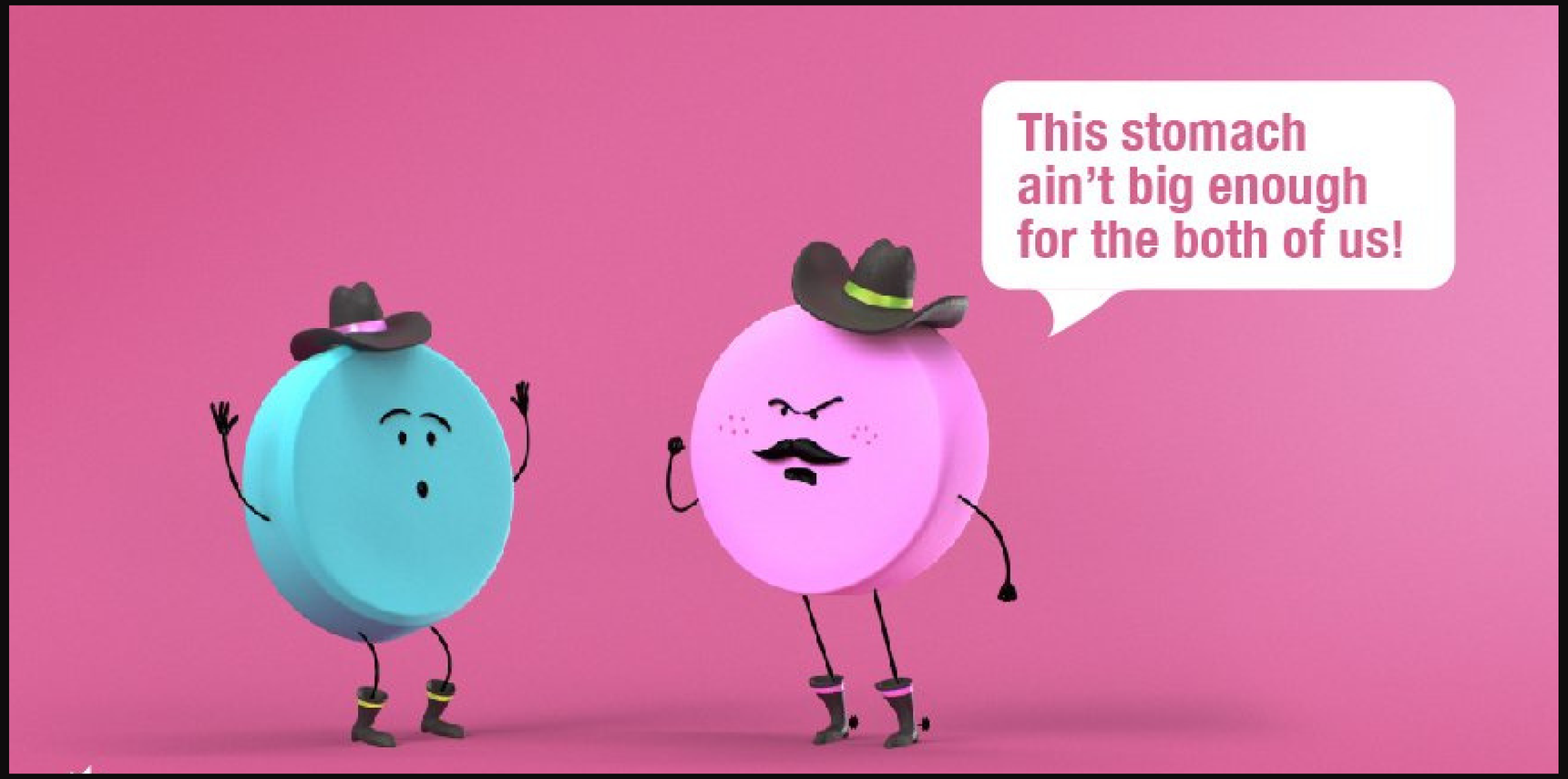


DRUG INTERACTIONS

By: Louise Fear, James Laurence, Gabriela Mkonde, and Chris Whalen

PROJECT RATIONALE

Enabling users to obtain **comprehensive information** about over-the-counter and prescription drugs. Facilitating **informed decision-making** for patients.

Offering a solution to identify potential drug interactions in the future - **enhancing patient safety** by mitigating certain drug interaction risks

Empowering patients to actively participate in their healthcare decisions. Informed patients are **better equipped** to **manage** their medications and **communicate** with their healthcare providers.

ACTIVITIES IN SCOPE

01

Project Charter Documentation

02

Database Deployment

03

Data Optimization and SQL Enhancement

04

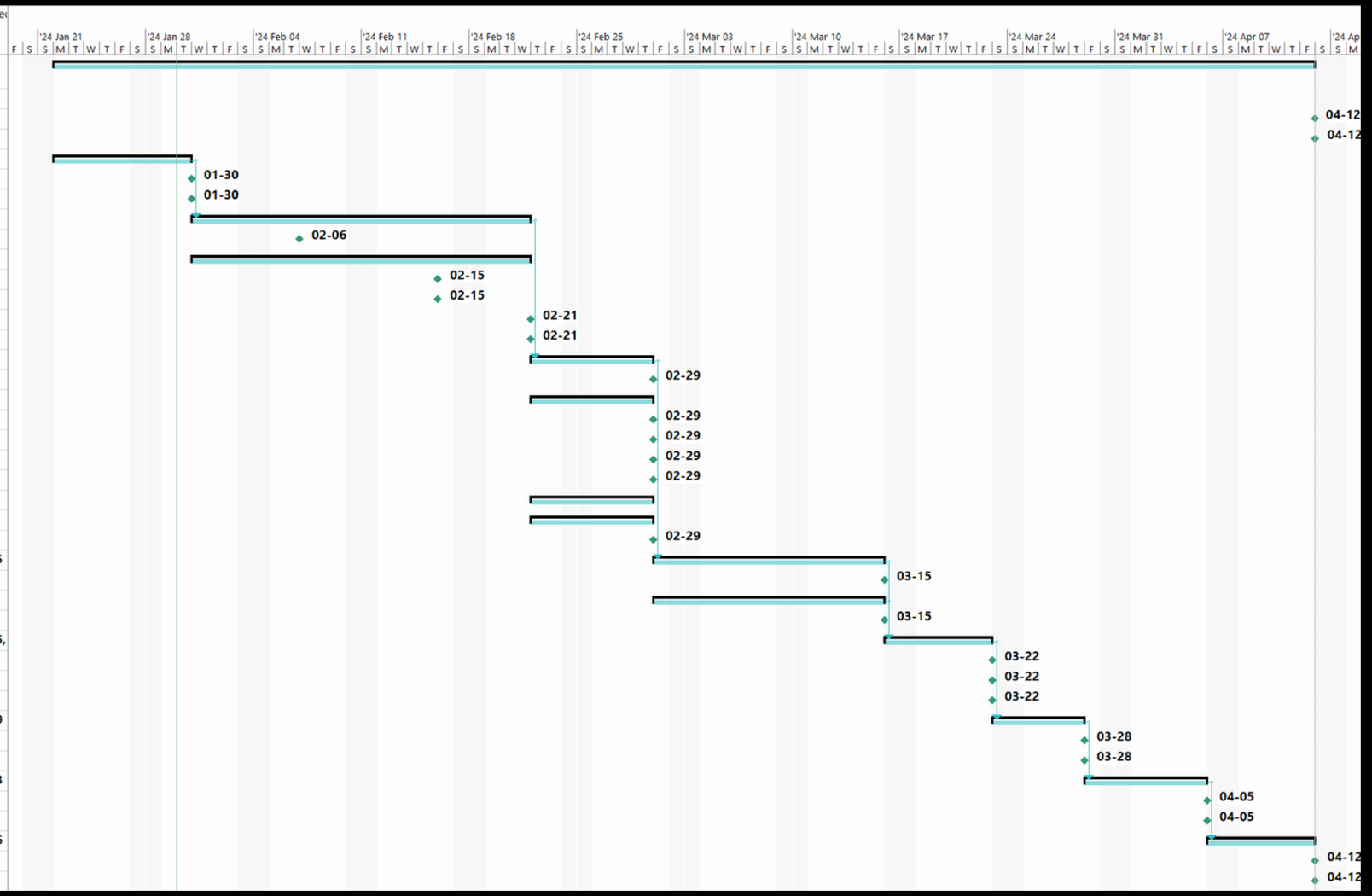
Web Based User Interface

ACTIVITIES OUT OF SCOPE

Automation	Supplemental Documentation	Official User Interface	Drug Interaction Prediction Algorithm
Develop and test a drug interaction prediction algorithm using predictive modelling. Optimize algorithm for effective/efficient interaction predictions (i.e. interactions with clinical trial drugs)	Risk Management Plan and Risk Register, Training Documentation/User Support Materials, and Quality Assurance Report with QA Review Reports	Design, implementation and testing of a User Interface whereby patients can search for specific drugs and pull up side effects, interactions, dosages etc.	Develop and test a predictive algorithm using the project's data that can predict whether a new drug entered in the database could have a interaction with another existing drug.

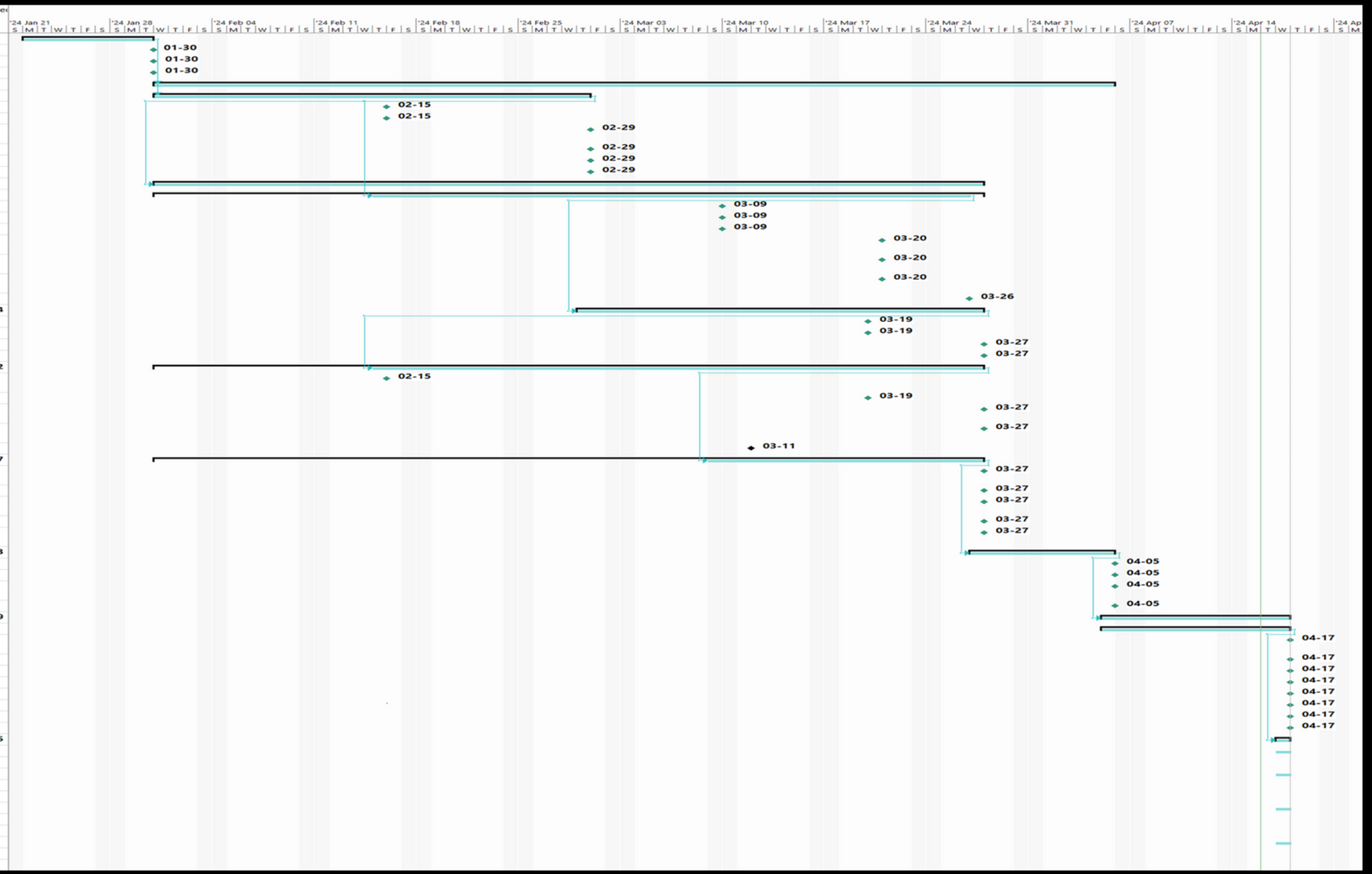
INITIAL PROJECT SCHEDULE

ID	Task Name	Duration	Start	Finish	Pred
1	**Project Charter Updated per Stakeholder Feedback**	**60 days**	**Mon 24-01-22**	**Fri 24-04-12**	
2	**Stakeholder Communication**	**60 days**	**Mon 24-01-22**	**Fri 24-04-12**	
3	Stakeholder Meetings	60 days	Mon 24-01-22	Fri 24-04-12	
4	Stakeholder Feedback	60 days	Mon 24-01-22	Fri 24-04-12	
5	**Project Charter Documentation**	**7 days**	**Mon 24-01-22**	**Tue 24-01-30**	
6	Project Charter Document	7 days	Mon 24-01-22	Tue 24-01-30	
7	Project Schedule	7 days	Mon 24-01-22	Tue 24-01-30	
8	**Database Deployment**	**16 days**	**Wed 24-01-31**	**Wed 24-02-21**	5
9	Requirement Gathering	5 days	Wed 24-01-31	Tue 24-02-06	
10	**Database Documentation**	**16 days**	**Wed 24-01-31**	**Wed 24-02-21**	
11	Entity Relationship Diagram	12 days	Wed 24-01-31	Thu 24-02-15	
12	Data Dictionary	12 days	Wed 24-01-31	Thu 24-02-15	
13	Schema SQL Script	16 days	Wed 24-01-31	Wed 24-02-21	
14	Database Testing	16 days	Wed 24-01-31	Wed 24-02-21	
15	**Data Acquisition, Validation, and Import**	**6 days?**	**Thu 24-02-22**	**Thu 24-02-29**	8
16	Data Acquisition	6 days	Thu 24-02-22	Thu 24-02-29	
17	**Data ETL**	**6 days?**	**Thu 24-02-22**	**Thu 24-02-29**	
18	Data Extraction	6 days?	Thu 24-02-22	Thu 24-02-29	
19	Data Cleaning	6 days?	Thu 24-02-22	Thu 24-02-29	
20	Data Validation	6 days?	Thu 24-02-22	Thu 24-02-29	
21	Data Import	6 days?	Thu 24-02-22	Thu 24-02-29	
22	**Optimized and Accurate SQL Queries**	**6 days?**	**Thu 24-02-22**	**Thu 24-02-29**	
23	**SQL Query Execution**	**6 days?**	**Thu 24-02-22**	**Thu 24-02-29**	
24	Query Optimization	6 days	Thu 24-02-22	Thu 24-02-29	
25	**Drug Interation Predictive Algorithm**	**11 days**	**Fri 24-03-01**	**Fri 24-03-15**	15
26	Drug Interation Algorithm Development	11 days	Fri 24-03-01	Fri 24-03-15	
27	**Optimized Predictive Algorithm**	**11 days?**	**Fri 24-03-01**	**Fri 24-03-15**	
28	Drug Interation Algorithm Testing	11 days	Fri 24-03-01	Fri 24-03-15	
29	**User Interface (UI)**	**6 days**	**Sat 24-03-16**	**Fri 24-03-22**	25,
30	User Interface Design	6 days	Sat 24-03-16	Fri 24-03-22	
31	User Interface Implmentation	6 days	Sat 24-03-16	Fri 24-03-22	
32	User Interface Testing	6 days	Sat 24-03-16	Fri 24-03-22	
33	**Risk Management Plan and Register**	**5 days**	**Sat 24-03-23**	**Thu 24-03-28**	29
34	Risk Assessment	5 days	Sat 24-03-23	Thu 24-03-28	
35	Risk Management Plan	5 days	Sat 24-03-23	Thu 24-03-28	
36	**Quality Assurance Report**	**6 days**	**Fri 24-03-29**	**Fri 24-04-05**	33
37	Quality Assurance Plan	6 days	Fri 24-03-29	Fri 24-04-05	
38	Regular Quality Reviews	6 days	Fri 24-03-29	Fri 24-04-05	
39	**Training and Support Documentation**	**6 days**	**Sat 24-04-06**	**Fri 24-04-12**	36
40	Training Materials	6 days	Sat 24-04-06	Fri 24-04-12	
41	Provision of User Support	6 days	Sat 24-04-06	Fri 24-04-12	

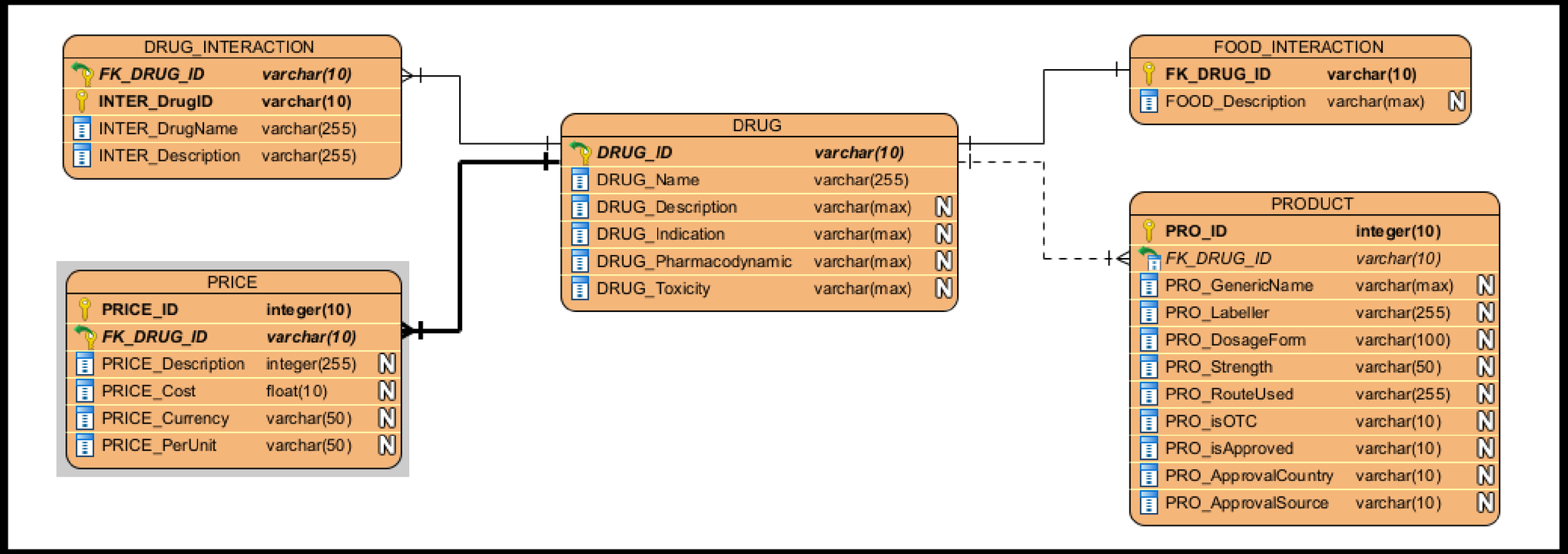


REVISED PROJECT SCHEDULE

Task Name	Duration	Start	Finish	Pred
Project Charter Documentation	**7 days**	**Mon 24-01-22**	**Tue 24-01-30**	
Project Charter Document	7 days	Mon 24-01-22	Tue 24-01-30	
Project Schedule	7 days	Mon 24-01-22	Tue 24-01-30	
Find Data Source	7 days	Mon 24-01-22	Tue 24-01-30	
Back-End Deployment	**48 days?**	**Wed 24-01-31**	**Fri 24-04-05**	1
Data Acquisition/ETL	**22 days?**	**Wed 24-01-31**	**Thu 24-02-29**	1
Pick Important Attributes from XML F	12 days	Wed 24-01-31	Thu 24-02-15	
Approve Important Attributes	12 days	Wed 24-01-31	Thu 24-02-15	
Extract Important Attributes Using Georges Powershell Script	22 days	Wed 24-01-31	Thu 24-02-29	
Convert XML into CSV	22 days?	Wed 24-01-31	Thu 24-02-29	
CSV Data Cleaning	22 days?	Wed 24-01-31	Thu 24-02-29	
Import Data into SQL database	22 days?	Wed 24-01-31	Thu 24-02-29	
Back End Development Documentatio	**41 days?**	**Wed 24-01-31**	**Wed 24-03-27**	6
ERD	**41 days?**	**Wed 24-01-31**	**Wed 24-03-27**	6
Draw a rough sketch of the ERD	18 days?	Thu 24-02-15	Sat 24-03-09	
Create Initial Version of ERD	18 days?	Thu 24-02-15	Sat 24-03-09	
Review the ERD with the team	18 days?	Thu 24-02-15	Sat 24-03-09	
Make necessary changes based on feedback	9 days?	Sat 24-03-09	Wed 24-03-20	
Review updated version of ERD with the team	9 days?	Sat 24-03-09	Wed 24-03-20	
Make necessary changes based on feedback	9 days?	Sat 24-03-09	Wed 24-03-20	
Finalize the ERD	13 days	Sat 24-03-09	Tue 24-03-26	
Data Dictionary	**20 days**	**Thu 24-02-29**	**Wed 24-03-27**	14
List of Data Dictionary Elements	14 days	Thu 24-02-29	Tue 24-03-19	
Review Data Dictionary with Team	14 days	Thu 24-02-29	Tue 24-03-19	
Make Changes based on feedback	7 days	Tue 24-03-19	Wed 24-03-27	
Finalize Data Dictionary	7 days	Tue 24-03-19	Wed 24-03-27	
DFD	**41 days?**	**Wed 24-01-31**	**Wed 24-03-27**	22
Use the ERD and Data Dictionary to create a DFD	1 day?	Thu 24-02-15	Thu 24-02-15	
Review the DFD with the team	8 days?	Sat 24-03-09	Tue 24-03-19	
Make necessary changes based on feedback	7 days	Tue 24-03-19	Wed 24-03-27	
Review the updated DFD with the team	14 days?	Sat 24-03-09	Wed 24-03-27	
Finalize DFD	1 day?	Mon 24-03-11	Mon 24-03-11	
Database Schema Creation	**41 days?**	**Wed 24-01-31**	**Wed 24-03-27**	27
DDL Script to create all tables and relationships	14 days	Sat 24-03-09	Wed 24-03-27	
Import Data into SQL database (TE	7 days?	Tue 24-03-19	Wed 24-03-27	
Update DDL script as required and retest	7 days?	Tue 24-03-19	Wed 24-03-27	
Finalize SQL Script	7 days?	Tue 24-03-19	Wed 24-03-27	
Import Cleaned Data into SQL database	7 days?	Tue 24-03-19	Wed 24-03-27	
SQL Query Testing	**8 days?**	**Wed 24-03-27**	**Fri 24-04-05**	33
SQL Database Query Testing	8 days	Wed 24-03-27	Fri 24-04-05	
SQL Query Validation	8 days?	Wed 24-03-27	Fri 24-04-05	
SQL Query Functional Testing - ACID Properties	8 days?	Wed 24-03-27	Fri 24-04-05	
Finalize SQL Query Testing	8 days?	Wed 24-03-27	Fri 24-04-05	
Front End Development	**9 days?**	**Fri 24-04-05**	**Wed 24-04-17**	39
User Interface (UI)	**9 days?**	Fri 24-04-05	Wed 24-04-17	
Research best practices for drug information UI	9 days?	Fri 24-04-05	Wed 24-04-17	
Sketch initial design ideas	9 days?	Fri 24-04-05	Wed 24-04-17	
Refine sketches based on feedback	9 days?	Fri 24-04-05	Wed 24-04-17	
Create a detailed UI design	9 days?	Fri 24-04-05	Wed 24-04-17	
Review and finalize the design	9 days?	Fri 24-04-05	Wed 24-04-17	
Develop UI Code	9 days?	Fri 24-04-05	Wed 24-04-17	
Develop a UI test plan	9 days?	Fri 24-04-05	Wed 24-04-17	
Execute the test cases	9 days?	Fri 24-04-05	Wed 24-04-17	
TASKS OUT OF SCOPE	**1 day?**	**Wed 24-04-17**	**Wed 24-04-17**	45
~~**Optimized Predictive Algorithm**~~				
~~Drug Interation Algorithm Testing~~				
~~**Risk Management Plan and Register**~~				
~~Risk Assessment~~				
~~Risk Management Plan~~				
~~**Quality Assurance Report**~~				
~~Quality Assurance Plan~~				
~~Regular Quality Reviews~~				
~~**Training and Support Documentatio**~~				
~~Training Materials~~				
~~Provision of User Support~~				



DRUG DATABASE
ENTITY RELATIONSHIP DIAGRAM

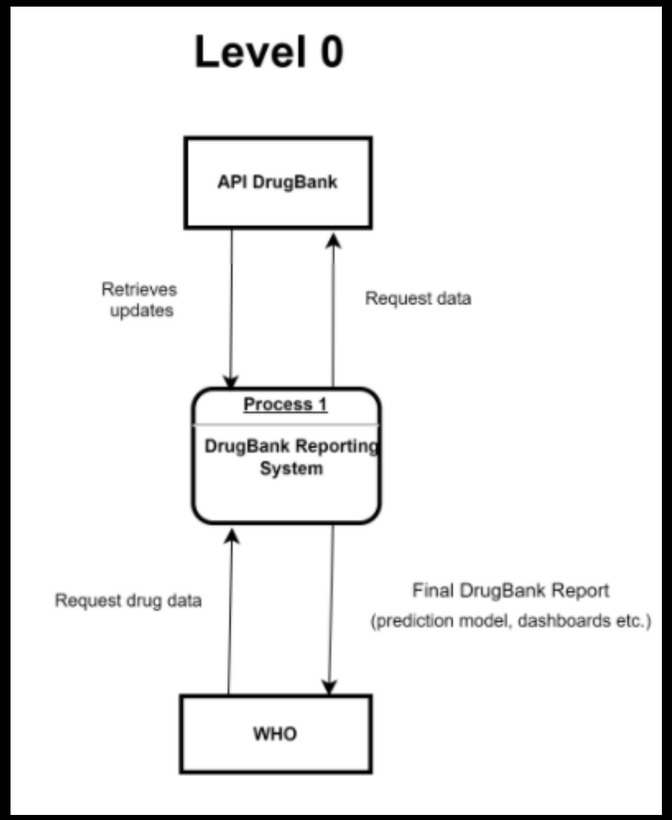


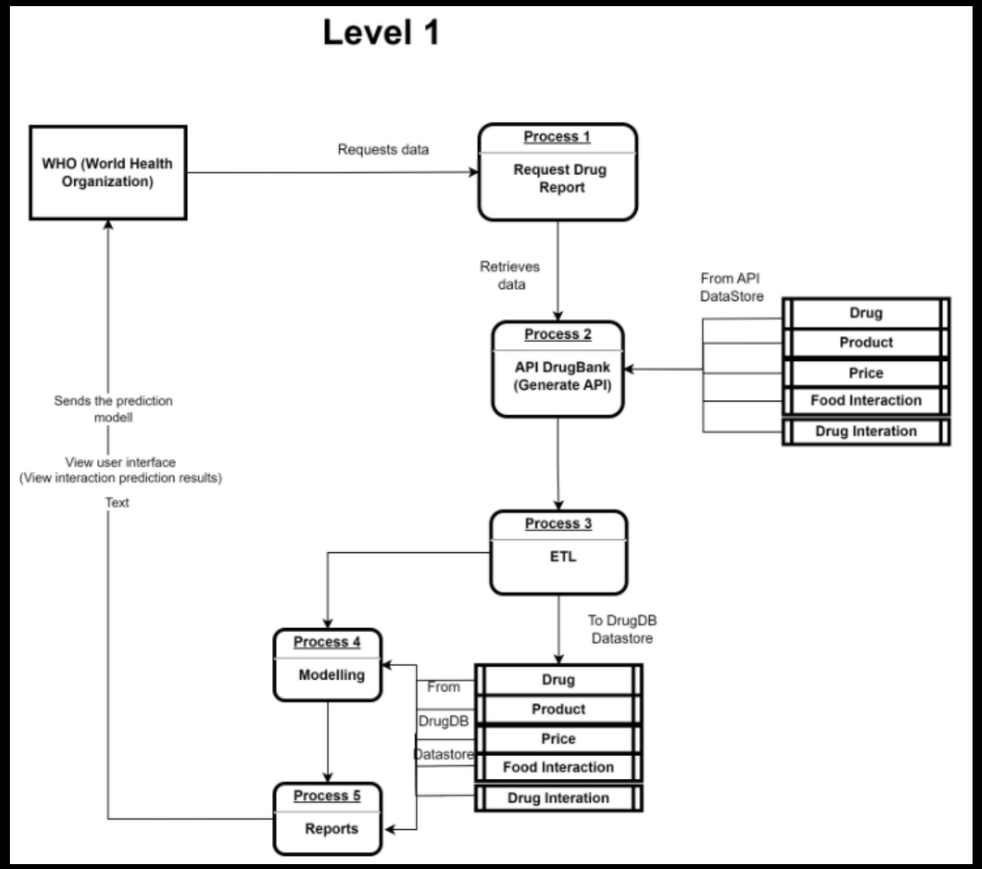
DRUG DATABASE
DATA DICTIONARY

Table name	Attribute name	Description	Type	Format	Constraints	Source
DRUGS	DRUG_ID	Unique numerical identifier assigned to each drug	nvarchar(10)	eg., DB00001	PK, NOT NULL	Drug Bank
	DRUG_Name	The common name of the drug.	nvarchar(255)		NULL	
	DRUG_Description	A brief description of the drug, its use, and possibly its mechanism of action.	nvarchar(max)		NULL	
	DRUG_Indication	Medical conditions or uses for which the drug is intended or prescribed	nvarchar(max)		NULL	
	DRUG_Pharmacodynamic	The biochemical and physiological effects of the drug and its mechanism of action	nvarchar(max)		NULL	
	DRUG_Toxicity	Information on the adverse drug reactions and potential toxic effects of the drug	nvarchar(max)		NULL	
DRUG_INTERACTIONS	FK_DRUG_ID	Unique numerical identifier assigned to each drug	nvarchar(10)		PK, FK (references drug), NOT NULL	Drug Bank
	INTER_DrugID	Unique identifier for each drug interaction entry.	varchar(10)		PK, NOT NULL	
	INTER_Name	The common name of the drug.	varchar(255)		NULL	
	INTER_Description	Textual description of the physiological consequences of the drug interaction.	varchar(255)		NULL	
FOOD_INTERACTIONS	FK_DRUG_ID	Unique numerical identifier assigned to each drug	nvarchar(10)		PK, FK (references drug), NOT NULL	Drug Bank
	FOOD_Description	Description of the interaction between the drug and food components.	varchar(max)		NULL	
PRICE	FK_DRUG_ID	Unique numerical identifier assigned to each drug	nvarchar(10)		PK, FK (references drug), NOT NULL	Drug Bank
	PRICE_Description	Description or context of the pricing information.	varchar(255)		NULL	
	PRICE_Cost	The cost amount for the drug.	float		NULL	
	PRICE_Currency	The currency in which the price is specified.	nvarchar(50)		NULL	
	PRICE_PerUnit	The unit or quantity that the price refers to.	nvarchar(50)		NULL	

PRICE	FK_DRUG_ID	Unique numerical identifier assigned to each drug	nvarchar(10)		PK, FK (references drug), NOT NULL	Drug Bank
	PRICE_Description	Description or context of the pricing information.	varchar(255)		NULL	
	PRICE_Cost	The cost amount for the drug.	float		NULL	
	PRICE_Currency	The currency in which the price is specified.	nvarchar(50)		NULL	
	PRICE_PerUnit	The unit or quantity that the price refers to.	nvarchar(50)		NULL	
PRODUCT	FK_DRUG_ID	Unique numerical identifier assigned to each drug	nvarchar(10)		PK, FK (references drug), NOT NULL	Drug Bank
	PRO_GenericName	The proprietary name(s) provided by the manufacturer for any commercially available products containing this drug.	varchar(max)		NULL	
	PRO_Labeller	Unique identifier assigned to the manufacturer or distributor of the drug.	varchar(255)		NULL	
	PRO_DosageForm	The pharmaceutical formulation by which the drug is introduced into the body.	varchar(100)		NULL	
	PRO_Strength	The concentration of active ingredients in the drug product.	varchar(50)		NULL	
	PRO_RouteUsed	The path by which the drug or product is taken into the body.	varchar(255)	e.g., oral, topical.	NULL	
	PRO_isOTC	Indicates whether the drug is available over the counter without a prescription.	varchar(10)		NULL	
	PRO_isApproved	Indicates whether this drug has been approved by the regulating government.	varchar(10)		NULL	
	PRO_ApprovalCountry	The country where this commercially available drug has been approved.	varchar(10)		NULL	
	PRO_ApprovalSource	The name of the authority or regulatory body that approved the drug.	varchar(10)		NULL	

DRUG DATABASE
DATA FLOW DIAGRAM





TRIAL & ERROR

DATA EXTRACTION

```python
import xml.etree.ElementTree as ET
import csv

# Path to the XML file
xml_path = 'C:/Users/louis/OneDrive - Nova Scotia Community College/Scool/

# Output CSV file
csv_output_path = 'C:/Users/louis/OneDrive - Nova Scotia Community College

# Parse XML
print("Parsing XML...")
tree = ET.parse(xml_path)
print("XML parsing complete.")

# Get the root element
root = tree.getroot()

with open(csv_output_path, 'w', newline='', encoding='utf-8') as csvfile:
    csvwriter = csv.writer(csvfile)

    # Write header
    headers = []
    for element in root.iter():
        headers.append(element.tag)
    csvwriter.writerow(headers)

    # Write data
    print("Writing data to CSV...")
    for row in root.findall('.//'):
        csvwriter.writerow([row.text])
    print("CSV writing complete.")
```

XML to CSV

```python
# File paths
xml_file_path = "C:/Users/louis/OneDrive - Nova Scotia Community College/Scool/Year 2/Semeste
csv_file_path = "C:/Users/louis/OneDrive - Nova Scotia Community College/Scool/Year 2/Semeste

# Check if the XML file exists
if not os.path.exists(xml_file_path):
    print("Error: XML file not found.")
else:
    print("Parsing XML file...")
    # Parse the XML file
    tree = etree.parse(xml_file_path)
    root = tree.getroot()

    # Initialize a set to store unique column headers
    column_headers = set()

    # Iterate through all elements in the XML tree
    for element in root.iter():
        # Get the tag name of the element (column header)
        column_header = element.tag.split("}")[1]  # Extract the tag name without namespace
        # Add the column header to the set
        column_headers.add(column_header)

    # Convert set of column headers to a list and sort it
    column_headers_list = sorted(list(column_headers))

    # Write XML data to CSV file
    print("Writing XML data to CSV file...")
    with open(csv_file_path, "w", newline="", encoding="utf-8") as csvfile:
        writer = csv.DictWriter(csvfile, fieldnames=column_headers_list)
        writer.writeheader()

        # Iterate through all elements in the XML tree
        for element in root.iter():
            # Create a dictionary to store data for the current element
            data_dict = {}
            # Iterate through all column headers
            for header in column_headers_list:
                # If the element has a child with the current column header, extract
                if element.find(header) is not None:
                    data_dict[header] = element.find(header).text
                else:
                    data_dict[header] = ""  # If the element doesn't have a child wi
            # Write the data for the current element to the CSV file
            writer.writerow(data_dict)

    print("CSV file successfully created.")
```

XML to CSV

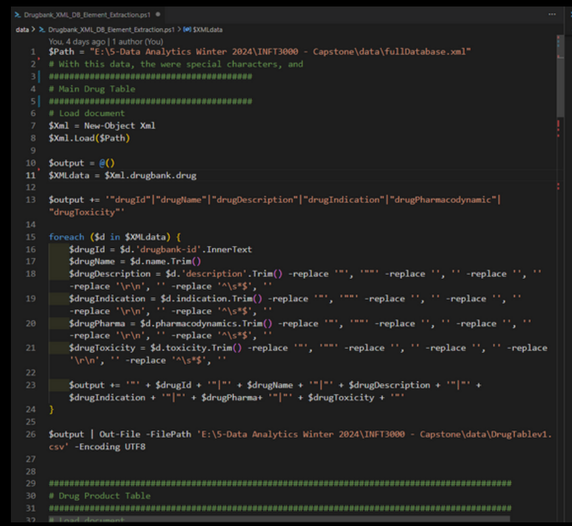
CSV WRITER - FIND COLUMN

```python
import xml.etree.ElementTree as ET
import csv

# Paths
xml_path = 'C:/Users/louis/Downloads/drugbank_all_full_database/full database.xml'
csv_path = 'B:/OneDrive - Nova Scotia Community College/Scool/Year 2/Semester 2/Capstone/output.csv'

# Open XML file for parsing
tree = ET.parse(xml_path)
root = tree.getroot()

# Open CSV file for writing
with open(csv_path, 'w', newline='', encoding='utf-8') as csv_file:
    # Create CSV writer
    csv_writer = csv.writer(csv_file)

    # Extract column names dynamically from the first XML element
    column_names = [child.tag for child in root[0]]  # Assuming all child elements of the root have the same structure

    # Write header row with column names
    csv_writer.writerow(column_names)

    # Iterate over XML elements and write data to CSV
    for element in root:
        row_data = [element.find(column).text if element.find(column) is not None else '' for column in column_names]
        csv_writer.writerow(row_data)

# Done
print("CSV file generated successfully.")
```

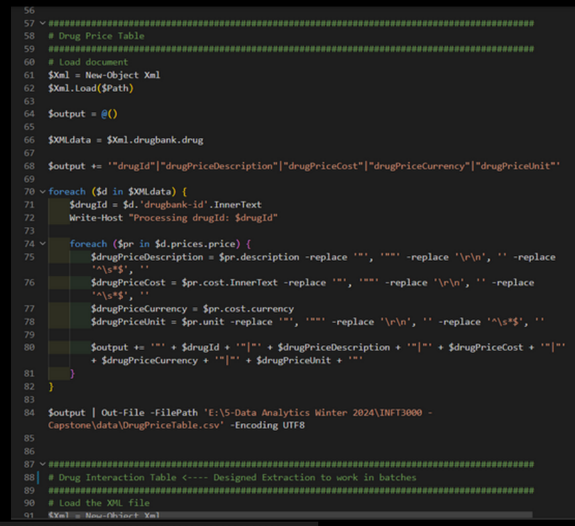
```python
import xmltodict
import csv

# File paths
xml_file_path = "C:/Users/louis/OneDrive - Nova Scotia Community College/Scool/Yea
csv_file_path = "C:/Users/louis/OneDrive - Nova Scotia Community College/Scool/Yea

# Parse the XML file and convert it to a Python dictionary
with open(xml_file_path, 'r', encoding='utf-8') as xml_file:
    xml_data = xml_file.read()
    json_data = xmltodict.parse(xml_data)

# Print the JSON data to understand its structure
print(json_data)
```

XML to JSON

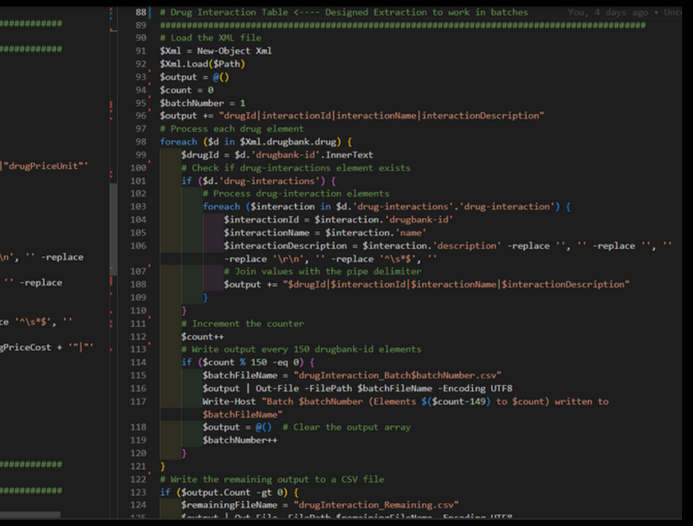
POWERSHELL CODE
XML EXTRACTION



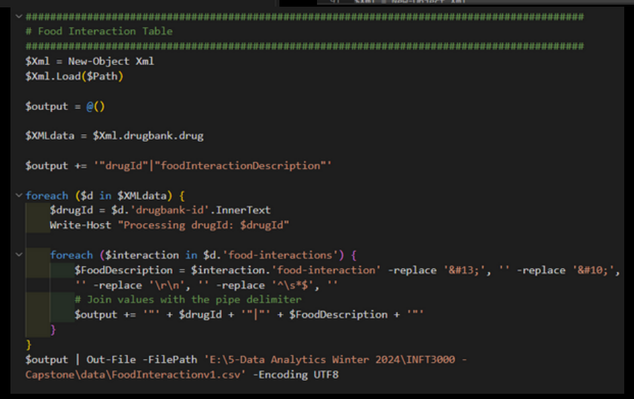
```powershell
# Main Drug Table
##########################################
# Load document
$Xml = New-Object Xml
$Xml.Load($Path)

$output = @()
$XMLdata = $Xml.drugbank.drug

$output += '"drugId"|"drugName"|"drugDescription"|"drugIndication"|"drugPharmacodynamic"|
"drugToxicity"'

foreach ($d in $XMLdata) {
    $drugId = $d.'drugbank-id'.InnerText
    $drugName = $d.name.Trim()
    $drugDescription = $d.'description'.Trim() -replace '"', '""' -replace '', '' -replace '', ''
    -replace '\r\n', '' -replace '^\s*$', ''
    $drugIndication = $d.indication.Trim() -replace '"', '""' -replace '', '' -replace '', ''
    -replace '\r\n', '' -replace '^\s*$', ''
    $drugPharma = $d.pharmacodynamics.Trim() -replace '"', '""' -replace '', '' -replace '', ''
    -replace '\r\n', '' -replace '^\s*$', ''
    $drugToxicity = $d.toxicity.Trim() -replace '"', '""' -replace '', '' -replace '', '' -replace
    '\r\n', '' -replace '^\s*$', ''

    $output += '"' + $drugId + '"|"' + $drugName + '"|"' + $drugDescription + '"|"' +
    $drugIndication + '"|"' + $drugPharma+ '"|"' + $drugToxicity + '"'
}

$output | Out-File -FilePath 'E:\5-Data Analytics Winter 2024\INFT3000 - Capstone\data\DrugTablev1.
csv' -Encoding UTF8


##########################################################################################
# Drug Product Table
##########################################################################################
# Load document
$Xml = New-Object Xml
$Xml.Load($Path)
$output = @()
$XMLdata = $Xml.drugbank.drug
$output += '"drugId"|"drugGenericName"|"drugLabeller"|"drugDosageForm"|"drugStrength"|
"drugRouteUsed"|"drugisOTC"|"drugisApproved"|"drugApprovalCountry"|"drugApprovalSource"'
foreach ($d in $XMLdata) {
    $drugId = $d.'drugbank-id'.InnerText
    Write-Host "Processing drugId: $drugId"
    foreach ($p in $d.products.product) {
        $drugGenericName = $p.name -replace '"', '""' -replace '', '' -replace '', '' -replace
        '\r\n', '' -replace '^\s*$', ''
        $drugLabeller = $p.labeller -replace '"', '""' -replace '', '' -replace '', '' -replace
        '\r\n', '' -replace '^\s*$', ''
        $drugDosageForm = $p.'dosage-form' -replace '"', '""' -replace '', '' -replace '', ''
        -replace '\r\n', '' -replace '^\s*$', ''
        $drugStrength = $p.strength -replace '"', '""' -replace '', '' -replace '', '' -replace
        '\r\n', '' -replace '^\s*$', ''
        $drugRoute = $p.route -replace '"', '""' -replace '', '' -replace '', '' -replace '\r\n',
        '' -replace '^\s*$', ''
        $drugOTC = $p.'over-the-counter' -replace '"', '""' -replace '', '' -replace '', ''
        -replace '\r\n', '' -replace '^\s*$', ''
        $drugApproved = $p.approved -replace '"', '""' -replace '', '' -replace '', '' -replace
        '\r\n', '' -replace '^\s*$', ''
        $drugCountry = $p.country -replace '"', '""' -replace '', '' -replace '', '' -replace
        '\r\n', '' -replace '^\s*$', ''
        $drugSource = $p.source -replace '"', '""' -replace '', '' -replace '', '' -replace
        '\r\n', '' -replace '^\s*$', ''

        $output += '"' + $drugId + '"|"' + $drugGenericName + '"|"' + $drugLabeller + '"|"' +
        $drugDosageForm + '"|"' + $drugStrength + '"|"' + $drugRoute + '"|"' + $drugOTC + '"|"' +
        $drugApproved + '"|"' + $drugCountry + '"|"' + $drugSource + '"'
    }
}
$output | Out-File -FilePath 'E:\5-Data Analytics Winter 2024\INFT3000 -
Capstone\data\DrugProductTable.CSV' -Encoding UTF8
```



```powershell
##########################################################################################
# Drug Price Table
##########################################################################################
# Load document
$Xml = New-Object Xml
$Xml.Load($Path)

$output = @()

$XMLdata = $Xml.drugbank.drug

$output += '"drugId"|"drugPriceDescription"|"drugPriceCost"|"drugPriceCurrency"|"drugPriceUnit"'

foreach ($d in $XMLdata) {
    $drugId = $d.'drugbank-id'.InnerText
    Write-Host "Processing drugId: $drugId"

    foreach ($pr in $d.prices.price) {
        $drugPriceDescription = $pr.description -replace '"', '""' -replace '\r\n', '' -replace
        '^\s*$', ''
        $drugPriceCost = $pr.cost.InnerText -replace '"', '""' -replace '\r\n', '' -replace
        '^\s*$', ''
        $drugPriceCurrency = $pr.cost.currency
        $drugPriceUnit = $pr.unit -replace '"', '""' -replace '\r\n', '' -replace '^\s*$', ''

        $output += '"' + $drugId + '"|"' + $drugPriceDescription + '"|"' + $drugPriceCost + '"|"'
        + $drugPriceCurrency + '"|"' + $drugPriceUnit + '"'
    }
}

$output | Out-File -FilePath 'E:\5-Data Analytics Winter 2024\INFT3000 -
Capstone\data\DrugPriceTable.csv' -Encoding UTF8


##########################################################################################
# Drug Interaction Table <---- Designed Extraction to work in batches
##########################################################################################
# Load the XML file
$Xml = New-Object Xml
```



```powershell
# Drug Interaction Table <---- Designed Extraction to work in batches      You, 4 days ago • Uncc
##########################################################################################
# Load the XML file
$Xml = New-Object Xml
$Xml.Load($Path)
$output = @()
$count = 0
$batchNumber = 1
$output += "drugId|interactionId|interactionName|interactionDescription"
# Process each drug element
foreach ($d in $Xml.drugbank.drug) {
    $drugId = $d.'drugbank-id'.InnerText
    # Check if drug-interactions element exists
    if ($d.'drug-interactions') {
        # Process drug-interaction elements
        foreach ($interaction in $d.'drug-interactions'.'drug-interaction') {
            $interactionId = $interaction.'drugbank-id'
            $interactionName = $interaction.'name'
            $interactionDescription = $interaction.'description' -replace '', '' -replace '', ''
            -replace '\r\n', '' -replace '^\s*$', ''
            # Join values with the pipe delimiter
            $output += "$drugId|$interactionId|$interactionName|$interactionDescription"
        }
    }
    # Increment the counter
    $count++
    # Write output every 150 drugbank-id elements
    if ($count % 150 -eq 0) {
        $batchFileName = "drugInteraction_Batch$batchNumber.csv"
        $output | Out-File -FilePath $batchFileName -Encoding UTF8
        Write-Host "Batch $batchNumber (Elements $($count-149) to $count) written to
        $batchFileName"
        $output = @()  # Clear the output array
        $batchNumber++
    }
}
# Write the remaining output to a CSV file
if ($output.Count -gt 0) {
    $remainingFileName = "drugInteraction_Remaining.csv"
```



```powershell
##########################################################################################
# Food Interaction Table
##########################################################################################
$Xml = New-Object Xml
$Xml.Load($Path)

$output = @()

$XMLdata = $Xml.drugbank.drug

$output += '"drugId"|"foodInteractionDescription"'

foreach ($d in $XMLdata) {
    $drugId = $d.'drugbank-id'.InnerText
    Write-Host "Processing drugId: $drugId"

    foreach ($interaction in $d.'food-interactions') {
        $FoodDescription = $interaction.'food-interaction' -replace '&#13;', '' -replace '&#10;',
        '' -replace '\r\n', '' -replace '^\s*$', ''
        # Join values with the pipe delimiter
        $output += '"' + $drugId + '"|"' + $FoodDescription + '"'
    }
}
$output | Out-File -FilePath 'E:\5-Data Analytics Winter 2024\INFT3000 -
Capstone\data\FoodInteractionv1.csv' -Encoding UTF8
```

PYTHON CODE

DATA CLEANING

```python
import pandas as pd
import os

# Read the CSV file with '|' delimiter
drugInteraction = pd.read_csv("drugInteraction.csv", sep='|')
drugPriceTable = pd.read_csv("DrugPriceTable.csv", sep='|')
drugProductTable = pd.read_csv("DrugProductTable.csv", sep='|')
drugTable = pd.read_csv("DrugTable.csv", sep='|')
foodInteraction = pd.read_csv("FoodInteraction.csv", sep='|')

# Check for duplicates before dropping and print the count
print("Duplicate counts before dropping:")
print("drugInteraction:", len(drugInteraction[drugInteraction.duplicated()]))
print("drugPriceTable:", len(drugPriceTable[drugPriceTable.duplicated()]))
print("drugProductTable:", len(drugProductTable[drugProductTable.duplicated()]))
print("drugTable:", len(drugTable[drugTable.duplicated()]))
print("foodInteraction:", len(foodInteraction[foodInteraction.duplicated()]))

# Drop duplicate rows and null values from each DataFrame
drugInteraction = drugInteraction.drop_duplicates().dropna()
drugPriceTable = drugPriceTable.drop_duplicates().dropna()
drugProductTable = drugProductTable.drop_duplicates().dropna()
drugTable = drugTable.drop_duplicates().dropna()
foodInteraction = foodInteraction.drop_duplicates().dropna()

# Check for duplicates after dropping and print the count
print("\nDuplicate counts after dropping:")
print("drugInteraction:", len(drugInteraction[drugInteraction.duplicated()]))
print("drugPriceTable:", len(drugPriceTable[drugPriceTable.duplicated()]))
print("drugProductTable:", len(drugProductTable[drugProductTable.duplicated()]))
print("drugTable:", len(drugTable[drugTable.duplicated()]))
print("foodInteraction:", len(foodInteraction[foodInteraction.duplicated()]))
```

PYTHON CODE

DATA CLEANING

```python
# ---------- REMOVE COMMAS ----------
# Function to remove commas from a string
def remove_commas(x):
    if isinstance(x, str):
        return x.replace(',', '')
    else:
        return x

# Remove commas from all columns of each DataFrame
drugInteraction = drugInteraction.applymap(remove_commas)
drugPriceTable = drugPriceTable.applymap(remove_commas)
drugProductTable = drugProductTable.applymap(remove_commas)
drugTable = drugTable.applymap(remove_commas)
foodInteraction = foodInteraction.applymap(remove_commas)


# ---------- CHECK COLUMN NAMES ----------
# Print column names for each DataFrame
print("Columns in drugInteraction DataFrame:")
print(drugInteraction.columns)

print("\nColumns in drugPriceTable DataFrame:")
print(drugPriceTable.columns)

print("\nColumns in drugProductTable DataFrame:")
print(drugProductTable.columns)

print("\nColumns in drugTable DataFrame:")
print(drugTable.columns)

print("\nColumns in foodInteraction DataFrame:")
print(foodInteraction.columns)
```

PYTHON CODE

DATA CLEANING

```python
# ---------- ADDING PRIMARY KEY COLUMNS ----------
# Add a new column 'DRIN_ID' with auto-incrementing values to drugInteraction DataFrame
drugInteraction['DRIN_ID'] = range(1, len(drugInteraction) + 1)
# Add a new column 'PRICE_ID' with auto-incrementing values to drugPriceTable DataFrame
drugPriceTable['PRICE_ID'] = range(1, len(drugPriceTable) + 1)
# Add a new column 'FOOD_ID' with auto-incrementing values to foodInteraction DataFrame
foodInteraction['FOOD_ID'] = range(1, len(foodInteraction) + 1)
# Add a new column 'PRO_ID' with auto-incrementing values to drugProductTable DataFrame
drugProductTable['PRO_ID'] = range(1, len(drugProductTable) + 1)


# ---------- RENAMING COLUMNS TO MATCH DDL SCRIPT ----------
# Rename columns for drugInteraction DataFrame
drugInteraction = drugInteraction.rename(columns={
    'drugId': 'FK_DRUG_ID',  # Updated to match DDL script
    'interactionId': 'INTER_DrugID',
    'interactionName': 'INTER_DrugName',
    'interactionDescription': 'INTER_Description'
})

# Rename columns for drugPriceTable DataFrame
drugPriceTable = drugPriceTable.rename(columns={
    'drugId': 'FK_DRUG_ID',  # Updated to match DDL script
    'drugPriceDescription': 'PRICE_Description',
    'drugPriceCost': 'PRICE_Cost',
    'drugPriceCurrency': 'PRICE_Currency',
    'drugPriceUnit': 'PRICE_PerUnit'
})

# Rename columns for drugProductTable DataFrame
drugProductTable = drugProductTable.rename(columns={
    'drugId': 'FK_DRUG_ID',  # Updated to match DDL script
    'drugGenericName': 'PRO_GenericName',
    'drugLabeller': 'PRO_Labeller',
    'drugDosageForm': 'PRO_DosageForm',
    'drugStrength': 'PRO_Strength',
    'drugRouteUsed': 'PRO_RouteUsed',
    'drugisOTC': 'PRO_isOTC',
    'drugisApproved': 'PRO_isApproved',
    'drugApprovalCountry': 'PRO_ApprovalCountry',
    'drugApprovalSource': 'PRO_ApprovalSource'
})
```

PYTHON CODE

DATA CLEANING

```python
# Rename columns for drugTable DataFrame
drugTable = drugTable.rename(columns={
    'drugId': 'DRUG_ID',  # Updated to match DDL script
    'drugName': 'DRUG_Name',  # Removed quotes to match DDL script
    'drugDescription': 'DRUG_Description',  # Removed quotes to match DDL script
    'drugIndication': 'DRUG_Indication',  # Removed quotes to match DDL script
    'drugPharmacodynamic': 'DRUG_Pharmacodynamic',  # Removed quotes to match DDL script
    'drugToxicity': 'DRUG_Toxicity'  # Removed quotes to match DDL script
})

# Drop the 'Unnamed: 0' column
foodInteraction = foodInteraction.drop(columns=['Unnamed: 0'])

# Rename columns for foodInteraction DataFrame
foodInteraction = foodInteraction.rename(columns={
    'drugId': 'FK_DRUG_ID',  # Updated to match DDL script
    'foodInteractionDescription': 'FOOD_Description'
})


# ---------- REORDERING COLUMNS TO HAVE PRIMARY KEY FIRST ----------
# Reorder columns in drugInteraction DataFrame
drugInteraction = drugInteraction[['DRIN_ID', 'FK_DRUG_ID', 'INTER_DrugID', 'INTER_DrugName', 'INTER_Description']]
# Reorder columns in drugPriceTable DataFrame
drugPriceTable = drugPriceTable[['PRICE_ID', 'FK_DRUG_ID', 'PRICE_Description', 'PRICE_Cost', 'PRICE_Currency', 'PRICE_PerUnit']]
# Reorder columns in foodInteraction DataFrame
foodInteraction = foodInteraction[['FOOD_ID', 'FK_DRUG_ID', 'FOOD_Description']]
# Reorder columns in drugProductTable DataFrame
drugProductTable = drugProductTable[['PRO_ID', 'FK_DRUG_ID', 'PRO_GenericName', 'PRO_Labeller', 'PRO_DosageForm', 'PRO_Strength',


# ---------- CHANGING DATATYPES TO MATCH DDL ----------
# Change data types for drugInteraction DataFrame
drugInteraction['FK_DRUG_ID'] = drugInteraction['FK_DRUG_ID'].astype('str')
drugInteraction['INTER_DrugID'] = drugInteraction['INTER_DrugID'].astype('str')
drugInteraction['INTER_DrugName'] = drugInteraction['INTER_DrugName'].astype('str')
drugInteraction['INTER_Description'] = drugInteraction['INTER_Description'].astype('str')

# Change data types for drugPriceTable DataFrame
drugPriceTable['PRICE_Description'] = drugPriceTable['PRICE_Description'].astype('str')
drugPriceTable['PRICE_Cost'] = drugPriceTable['PRICE_Cost'].astype('float')
drugPriceTable['PRICE_Currency'] = drugPriceTable['PRICE_Currency'].astype('str')
drugPriceTable['PRICE_PerUnit'] = drugPriceTable['PRICE_PerUnit'].astype('str')
# Drop rows where 'PRICE_Cost' cannot be converted to float
drugPriceTable = drugPriceTable[pd.to_numeric(drugPriceTable['PRICE_Cost'], errors='coerce').notnull()]
```

PYTHON CODE

DATA CLEANING

```python
# Change data types for drugPriceTable DataFrame
drugPriceTable['PRICE_Description'] = drugPriceTable['PRICE_Description'].astype('str')
drugPriceTable['PRICE_Cost'] = drugPriceTable['PRICE_Cost'].astype('float')
drugPriceTable['PRICE_Currency'] = drugPriceTable['PRICE_Currency'].astype('str')
drugPriceTable['PRICE_PerUnit'] = drugPriceTable['PRICE_PerUnit'].astype('str')
# Drop rows where 'PRICE_Cost' cannot be converted to float
drugPriceTable = drugPriceTable[pd.to_numeric(drugPriceTable['PRICE_Cost'], errors='coerce').notnull()]

# Change data types for drugProductTable DataFrame
drugProductTable['FK_DRUG_ID'] = drugProductTable['FK_DRUG_ID'].astype('str')
drugProductTable['PRO_GenericName'] = drugProductTable['PRO_GenericName'].astype('str')
drugProductTable['PRO_Labeller'] = drugProductTable['PRO_Labeller'].astype('str')
drugProductTable['PRO_DosageForm'] = drugProductTable['PRO_DosageForm'].astype('str')
drugProductTable['PRO_Strength'] = drugProductTable['PRO_Strength'].astype('str')
drugProductTable['PRO_RouteUsed'] = drugProductTable['PRO_RouteUsed'].astype('str')
drugProductTable['PRO_isOTC'] = drugProductTable['PRO_isOTC'].astype('str')
drugProductTable['PRO_isApproved'] = drugProductTable['PRO_isApproved'].astype('str')
drugProductTable['PRO_ApprovalCountry'] = drugProductTable['PRO_ApprovalCountry'].astype('str')
drugProductTable['PRO_ApprovalSource'] = drugProductTable['PRO_ApprovalSource'].astype('str')

# Change data types for drugTable DataFrame
drugTable['DRUG_ID'] = drugTable['DRUG_ID'].astype('str')
drugTable['DRUG_Name'] = drugTable['DRUG_Name'].astype('str')
drugTable['DRUG_Description'] = drugTable['DRUG_Description'].astype('str')
drugTable['DRUG_Indication'] = drugTable['DRUG_Indication'].astype('str')
drugTable['DRUG_Pharmacodynamic'] = drugTable['DRUG_Pharmacodynamic'].astype('str')
drugTable['DRUG_Toxicity'] = drugTable['DRUG_Toxicity'].astype('str')

# Change data types for foodInteraction DataFrame
foodInteraction['FK_DRUG_ID'] = foodInteraction['FK_DRUG_ID'].astype('str')
foodInteraction['FOOD_Description'] = foodInteraction['FOOD_Description'].astype('str')


# ---------- EXPORTING CSVS ----------
# Define the output directory
output_dir = "C:/Users/louis/OneDrive - Nova Scotia Community College/Scool/Year 2/Semester 2/Capstone/cleanData"

# Export DataFrames to CSV files with specified data types
drugInteraction.to_csv(os.path.join(output_dir, "drugInteraction_clean.csv"), index=False, encoding='utf-8', float_format='%.2f')
drugPriceTable.to_csv(os.path.join(output_dir, "drugPriceTable_clean.csv"), index=False, encoding='utf-8', float_format='%.2f')
drugProductTable.to_csv(os.path.join(output_dir, "drugProductTable_clean.csv"), index=False, encoding='utf-8', float_format='%.2f')
drugTable.to_csv(os.path.join(output_dir, "drugTable_clean.csv"), index=False, encoding='utf-8', float_format='%.2f')
foodInteraction.to_csv(os.path.join(output_dir, "foodInteraction_clean.csv"), index=False, encoding='utf-8', float_format='%.2f')
```

SQL CODE

```sql
-- Create Database drugDB
CREATE DATABASE drugDB
GO
-- uses newly created database
USE [drugDB]
GO
-- Creation of the DRUG Table
CREATE TABLE [dbo].[DRUG](
    [DRUG_ID] [varchar](10) NOT NULL,
    [DRUG_Name] [nvarchar](255) NOT NULL,
    [DRUG_Description] [nvarchar](max) NULL,
    [DRUG_Indication] [nvarchar](max) NULL,
    [DRUG_Pharmacodynamic] [nvarchar](max) NULL,
    [DRUG_Toxicity] [nvarchar](max) NULL,
    CONSTRAINT PK_DRUG_ID PRIMARY KEY ([DRUG_ID]) -- Set Primary Key
)
GO
-- Creation of the PRICE table
CREATE TABLE [dbo].[PRICE](
    [PRICE_ID] [int] IDENTITY(1, 1) NOT NULL,
    [FK_DRUG_ID] [varchar](10) NOT NULL,
    [PRICE_Description] [varchar](255) NULL,
    [PRICE_Cost] [float] NULL,
    [PRICE_Currency] [nvarchar](50) NULL,
    [PRICE_PerUnit] [nvarchar](50) NULL,
    CONSTRAINT PK_PRICE_ID PRIMARY KEY ([PRICE_ID]) -- Set primary
    key
)
GO
-- Creation of the DRUG_INTERACTION Table
CREATE TABLE [dbo].[DRUG_INTERACTION](
    [FK_DRUG_ID] [varchar](10) NOT NULL,
    [INTER_DrugID] [varchar](10) NOT NULL,
    [INTER_DrugName] [varchar](255) NOT NULL,
    [INTER_Description] [varchar](255) NOT NULL,
    CONSTRAINT PK_DRUG_ID_INTER_DrugID PRIMARY KEY ([FK_DRUG_ID],
    [INTER_DrugID]) -- Set composite key
)
GO
-- Creation of the FOOD_INTERACTION table
CREATE TABLE [dbo].[FOOD_INTERACTION](
    [FK_DRUG_ID] [varchar](10) NOT NULL,
    [FOOD_Description] [varchar](max) NULL,
    CONSTRAINT PK_FK_DRUG_ID PRIMARY KEY ([FK_DRUG_ID])
    -- Sets composite key
)
GO
-- Creation of the PRODUCT table
CREATE TABLE [dbo].[PRODUCT](
    [PRO_ID] [int] IDENTITY(1, 1) NOT NULL,
    [FK_DRUG_ID] [varchar](10) NOT NULL,
    [PRO_GenericName] [varchar](max) NULL,
    [PRO_Labeller] [varchar](255) NULL,
    [PRO_DosageForm] [varchar](100) NULL,
    [PRO_Strength] [varchar](50) NULL,
    [PRO_RouteUsed] [varchar](255) NULL,
    [PRO_isOTC] [varchar](10) NULL,
    [PRO_isApproved] [varchar](10) NULL,
    [PRO_ApprovalCountry] [varchar](10) NULL,
    [PRO_ApprovalSource] [varchar](10) NULL,
    CONSTRAINT PK_PROD_ID PRIMARY KEY ([PRO_ID]) -- Sets
    primary key
)
GO
-- Setting Foreign Key Constraint for PRICE Table with
Cascade on Update/Delete
ALTER TABLE [dbo].[PRICE]
ADD CONSTRAINT [PRICE_FK_DRUG_ID] FOREIGN KEY
([FK_DRUG_ID])
REFERENCES [dbo].[DRUG] ([DRUG_ID])
ON UPDATE CASCADE
ON DELETE CASCADE
GO


GO
-- Setting Foreign Key Constraint for DRUG_INTERACTION Table
with Cascade on Update/Delete
ALTER TABLE [dbo].[DRUG_INTERACTION]
ADD CONSTRAINT [DRUG_INTER_FK_DRUG_ID] FOREIGN KEY
([FK_DRUG_ID])
REFERENCES [dbo].[DRUG] ([DRUG_ID])
ON UPDATE CASCADE
ON DELETE CASCADE
GO
-- Setting Foreign Key Constraint for FOOD_INTERACTION Table
with Cascade on Update/Delete
ALTER TABLE [dbo].[FOOD_INTERACTION]
ADD CONSTRAINT [FOOD_INTER_FK_DRUG_ID] FOREIGN KEY
([FK_DRUG_ID])
REFERENCES [dbo].[DRUG] ([DRUG_ID])
ON UPDATE CASCADE
ON DELETE CASCADE
GO
-- Setting Foreign Key Constraint for PRODUCT Table with
Cascade on Update/Delete
ALTER TABLE [dbo].[PRODUCT]
ADD CONSTRAINT [PRODUCT_FK_DRUG_ID] FOREIGN KEY([FK_DRUG_ID])
REFERENCES [dbo].[DRUG] ([DRUG_ID])
ON UPDATE CASCADE
ON DELETE CASCADE
GO
```

BULK INSERTS

IMPORT DATA

```
USE drugDB
GO

-- Insert data from 'drugInteraction_clean.csv' into the 'DRUG_INTERACTION' table
BULK INSERT [dbo].[DRUG_INTERACTION]
FROM 'C:/Users/louis/OneDrive - Nova Scotia Community College/Scool/Year 2/Semester 2/Capstone/cleanData/drugInteraction_clean.csv
WITH (
    FIELDTERMINATOR = ',',
    ROWTERMINATOR = '\n',
    FIRSTROW = 2 -- Skip header row if present
);

-- Insert data from 'drugPriceTable_clean.csv' into the 'PRICE' table
BULK INSERT [dbo].[PRICE]
FROM 'C:/Users/louis/OneDrive - Nova Scotia Community College/Scool/Year 2/Semester 2/Capstone/cleanData/drugPriceTable_clean.csv'
WITH (
    FIELDTERMINATOR = ',',
    ROWTERMINATOR = '\n',
    FIRSTROW = 2 -- Skip header row if present
);

-- Insert data from 'drugProductTable_clean.csv' into the 'PRODUCT' table
BULK INSERT [dbo].[PRODUCT]
FROM 'C:/Users/louis/OneDrive - Nova Scotia Community College/Scool/Year 2/Semester 2/Capstone/cleanData/drugProductTable_clean.cs
WITH (
    FIELDTERMINATOR = ',',
    ROWTERMINATOR = '\n',
    FIRSTROW = 2 -- Skip header row if present
);

-- Insert data from 'drugTable_clean.csv' into the 'DRUG' table
BULK INSERT [dbo].[DRUG]
FROM 'C:/Users/louis/OneDrive - Nova Scotia Community College/Scool/Year 2/Semester 2/Capstone/cleanData/drugTable_clean.csv'
WITH (
    FIELDTERMINATOR = ',',
    ROWTERMINATOR = '\n',
    FIRSTROW = 2 -- Skip header row if present
);
```

DATABASE

	DRUG_ID	DRUG_Name	DRUG_Description	DRUG_Indication	DRUG_Pharmacodynamic	DRUG_Toxicity
1	DB00001	Lepirudin	"Lepirudin is a recombinant hirudin formed by 65 am...	L41569] Natural hirudin is an endogenous anticoa...	an immune reaction associated with a high risk of ...	L41539] HIT is caused by the expression of immun...
2	DB00002	Cetuximab	"Cetuximab is a recombinant chimeric human/mous...	as it is often overexpressed in malignant cells [A22...	cetuximab was shown to mediate anti-tumour effe...	cetuximab is used for the treatment of head and ne...
3	DB00003	Dornase alfa	"Dornase alfa is a biosynthetic form of human deoxy...	extracellular DNA	which is an extremely viscous anion	is released by degenerating leukocytes that accum...
4	DB00006	Bivalirudin	"Bivalirudin is a synthetic 20 residue peptide (thromb...	thrombin cannot activate fibrinogen into fibrin	the crucial step in the formation of thrombus. It is ...	it is important to monitor changes in hematocrit, act...
5	DB00007	Leuprolide	"Leuprolide is a synthetic 9-residue peptide analogu...	leuprolide contains a single D-amino acid (D-leucyl...	which helps to increase its circulating half-life from...	leuprolide is capable of binding to the GnRH recept...
6	DB00008	Peginterferon alfa-2a	"Peginterferon alfa-2a is a form of recombinant inter...	an infectious liver disease caused by infection with...	with genotype 1 being the most common in the U...	and affecting 72% of all chronic HCV patients [L85...
7	DB00009	Alteplase	"Alteplase is a recombinant tissue plasminogen acti...	an enzyme involved in the degradation of fibrin clo...	the alteplase-mediated conversion of plasminoge...	thanks to the high affinity between alteplase and fi...
8	DB00013	Urokinase	Urokinase is an endogenous peptide that is cleaved...	"In Canada	urokinase is indicated for lysis of acute massive p...	acute thrombi obstructing coronary arteries, occlusi...
9	DB00014	Goserelin	"Goserelin is a synthetic hormone. In men	it stops the production of the hormone testosterone	which may stimulate the growth of cancer cells. In ...	goserelin decreases the production of the hormone...
10	DB00016	Erythropoietin	"Erythropoietin (EPO) is a growth factor produced in...	followed by other alfa and beta formulations. Epoe...	such as chronic renal failure	antiviral drug therapy, chemotherapy, or a high risk f...
11	DB00017	Salmon calcitonin	"Synthetic peptide	32 residues long formulated as a nasal spray."	"Used in the treatment of symptomatic Paget's dis...	it is used in emergency situations when serum calci...

	PRO_ID	FK_DRUG_ID	PRO_GenericName	PRO_Labeller	PRO_DosageForm	PRO_Strength	PRO_RouteUsed	PRO_isOTC	PRO_isApproved	PRO_ApprovalCountry	PRO_ApprovalSource
1	1	DB00001	Refludan	Bayer	Powder	50 mg/1mL	Intravenous	False	True	US	FDA NDC
2	2	DB00001	Refludan	Bayer	Powder for solution	50 mg / vial	Intravenous	False	True	Canada	DPD
3	3	DB00001	Refludan	Celgene Europe Limited	Injection solution concentrate	50 mg	Intravenous	False	False	EU	EMA
4	4	DB00001	Refludan	Celgene Europe Limited	Injection solution concentrate	20 mg	Intravenous	False	False	EU	EMA
5	5	DB00002	Erbitux	ImClone LLC	Solution	2 mg/1mL	Intravenous	False	True	US	FDA NDC
6	6	DB00002	Erbitux	Imclone Llc	Solution	2 mg / mL	Intravenous	False	True	Canada	DPD
7	7	DB00002	Erbitux	Merck Europe B.V.	Injection solution	5 mg/ml	Intravenous	False	True	EU	EMA
8	8	DB00003	Pulmozyme	Genentech Inc.	Solution	1 mg/1mL	Respiratory (inhalation)	False	True	US	FDA NDC
9	9	DB00003	Pulmozyme 1mg/ml	Hoffmann La Roche	Solution	1 mg / mL	Respiratory (inhalation)	False	True	Canada	DPD
10	10	DB00004	Ontak	Eisai Limited	Injection solution	150 ug/1mL	Intravenous	False	True	US	FDA NDC
11	11	DB00005	Benepali	Samsung Bioepis NI B.V.	Injection solution	50 mg	Subcutaneous	False	True	EU	EMA
12	12	DB00005	Benepali	Samsung Bioepis NI B.V.	Injection solution	25 mg	Subcutaneous	False	True	EU	EMA
13	13	DB00005	Brenzys	Samsung Bioepis Co. Ltd.	Solution	50 mg / mL	Subcutaneous	False	True	Canada	DPD

DATABASE

	FOOD_ID	FK_DRUG_ID	FOOD_Description
1	1	DB00001	"Avoid herbs and supplements with anticoagulant/a...
2	2	DB00006	"Avoid echinacea. Avoid herbs and supplements wi...
3	3	DB00008	Drink plenty of fluids.
4	4	DB00009	"Avoid herbs and supplements with anticoagulant/a...
5	5	DB00011	Avoid alcohol.
6	6	DB00012	"Administer iron supplement. When initiating an ery...
7	7	DB00013	"Avoid herbs and supplements with anticoagulant/a...

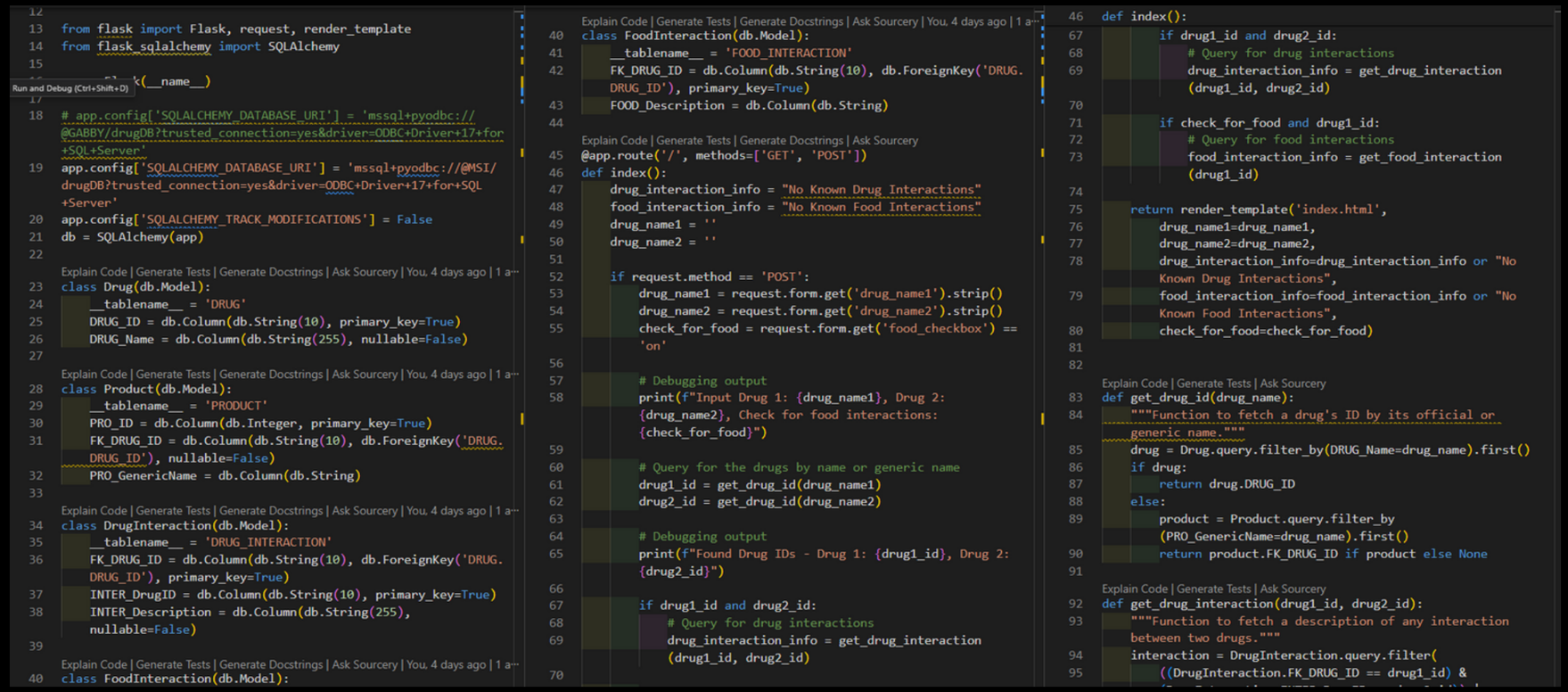
	PRICE_ID	FK_DRUG_ID	PRICE_Description	PRICE_Cost	PRICE_Currency	PRICE_PerUnit
1	1	DB00001	Refludan 50 mg vial	273.19	USD	vial
2	2	DB00003	Lufyllin 200 mg tablet	3.21	USD	tablet
3	3	DB00003	Lufyllin-gg tablet	3.84	USD	tablet
4	4	DB00003	Lufyllin-GG 200-200 mg tablet	3.99	USD	tablet
5	5	DB00003	Lufyllin-400 tablet	4.62	USD	tablet
6	6	DB00003	Lufyllin 400 mg tablet	4.81	USD	tablet
7	7	DB00003	Pulmozyme 1 mg/ml ampul	37.05	USD	ml

DATABASE

	DRIN_ID	FK_DRUG_ID	INTER_DrugID	INTER_DrugName	INTER_Description
1	1	DB00001	DB06605	Apixaban	Apixaban may increase the anticoagulant activitie...
2	2	DB00001	DB06695	Dabigatran etexilate	Dabigatran etexilate may increase the anticoagul...
3	3	DB00001	DB01254	Dasatinib	The risk or severity of bleeding and hemorrhage c...
4	4	DB00001	DB01609	Deferasirox	The risk or severity of gastrointestinal bleeding ca...
5	5	DB00001	DB01586	Ursodeoxycholic acid	The risk or severity of bleeding and bruising can b...
6	6	DB00001	DB02123	Glycochenodeoxycholic Acid	The risk or severity of bleeding and bruising can b...
7	7	DB00001	DB02659	Cholic Acid	The risk or severity of bleeding and bruising can b...

PYTHON CODE

WEB SERVER



```python
from flask import Flask, request, render_template
from flask_sqlalchemy import SQLAlchemy

app = Flask(__name__)

# app.config['SQLALCHEMY_DATABASE_URI'] = 'mssql+pyodbc://@GABBY/drugDB?trusted_connection=yes&driver=ODBC+Driver+17+for+SQL+Server'
app.config['SQLALCHEMY_DATABASE_URI'] = 'mssql+pyodbc://@MSI/drugDB?trusted_connection=yes&driver=ODBC+Driver+17+for+SQL+Server'
app.config['SQLALCHEMY_TRACK_MODIFICATIONS'] = False
db = SQLAlchemy(app)

class Drug(db.Model):
    __tablename__ = 'DRUG'
    DRUG_ID = db.Column(db.String(10), primary_key=True)
    DRUG_Name = db.Column(db.String(255), nullable=False)

class Product(db.Model):
    __tablename__ = 'PRODUCT'
    PRO_ID = db.Column(db.Integer, primary_key=True)
    FK_DRUG_ID = db.Column(db.String(10), db.ForeignKey('DRUG.DRUG_ID'), nullable=False)
    PRO_GenericName = db.Column(db.String)

class DrugInteraction(db.Model):
    __tablename__ = 'DRUG_INTERACTION'
    FK_DRUG_ID = db.Column(db.String(10), db.ForeignKey('DRUG.DRUG_ID'), primary_key=True)
    INTER_DrugID = db.Column(db.String(10), primary_key=True)
    INTER_Description = db.Column(db.String(255), nullable=False)

class FoodInteraction(db.Model):
    __tablename__ = 'FOOD_INTERACTION'
    FK_DRUG_ID = db.Column(db.String(10), db.ForeignKey('DRUG.DRUG_ID'), primary_key=True)
    FOOD_Description = db.Column(db.String)

@app.route('/', methods=['GET', 'POST'])
def index():
    drug_interaction_info = "No Known Drug Interactions"
    food_interaction_info = "No Known Food Interactions"
    drug_name1 = ''
    drug_name2 = ''

    if request.method == 'POST':
        drug_name1 = request.form.get('drug_name1').strip()
        drug_name2 = request.form.get('drug_name2').strip()
        check_for_food = request.form.get('food_checkbox') == 'on'

        # Debugging output
        print(f"Input Drug 1: {drug_name1}, Drug 2: {drug_name2}, Check for food interactions: {check_for_food}")

        # Query for the drugs by name or generic name
        drug1_id = get_drug_id(drug_name1)
        drug2_id = get_drug_id(drug_name2)

        # Debugging output
        print(f"Found Drug IDs - Drug 1: {drug1_id}, Drug 2: {drug2_id}")

        if drug1_id and drug2_id:
            # Query for drug interactions
            drug_interaction_info = get_drug_interaction(drug1_id, drug2_id)

        if check_for_food and drug1_id:
            # Query for food interactions
            food_interaction_info = get_food_interaction(drug1_id)

    return render_template('index.html',
        drug_name1=drug_name1,
        drug_name2=drug_name2,
        drug_interaction_info=drug_interaction_info or "No Known Drug Interactions",
        food_interaction_info=food_interaction_info or "No Known Food Interactions",
        check_for_food=check_for_food)

def get_drug_id(drug_name):
    """Function to fetch a drug's ID by its official or generic name."""
    drug = Drug.query.filter_by(DRUG_Name=drug_name).first()
    if drug:
        return drug.DRUG_ID
    else:
        product = Product.query.filter_by(PRO_GenericName=drug_name).first()
        return product.FK_DRUG_ID if product else None

def get_drug_interaction(drug1_id, drug2_id):
    """Function to fetch a description of any interaction between two drugs."""
    interaction = DrugInteraction.query.filter(
        ((DrugInteraction.FK_DRUG_ID == drug1_id) & (DrugInteraction.INTER_DrugID == drug2_id)) |
        ((DrugInteraction.FK_DRUG_ID == drug2_id) & (DrugInteraction.INTER_DrugID == drug1_id))
    ).first()
    return interaction.INTER_Description if interaction else "No Known Drug Interactions"

def get_food_interaction(drug_id):
    """Function to fetch a description of any food interaction for a drug."""
    food_interaction = FoodInteraction.query.filter_by(FK_DRUG_ID=drug_id).first()
    return food_interaction.FOOD_Description if food_interaction else "No Known Food Interactions"

if __name__ == '__main__':
    app.run(debug=True)
```

HTML CODE

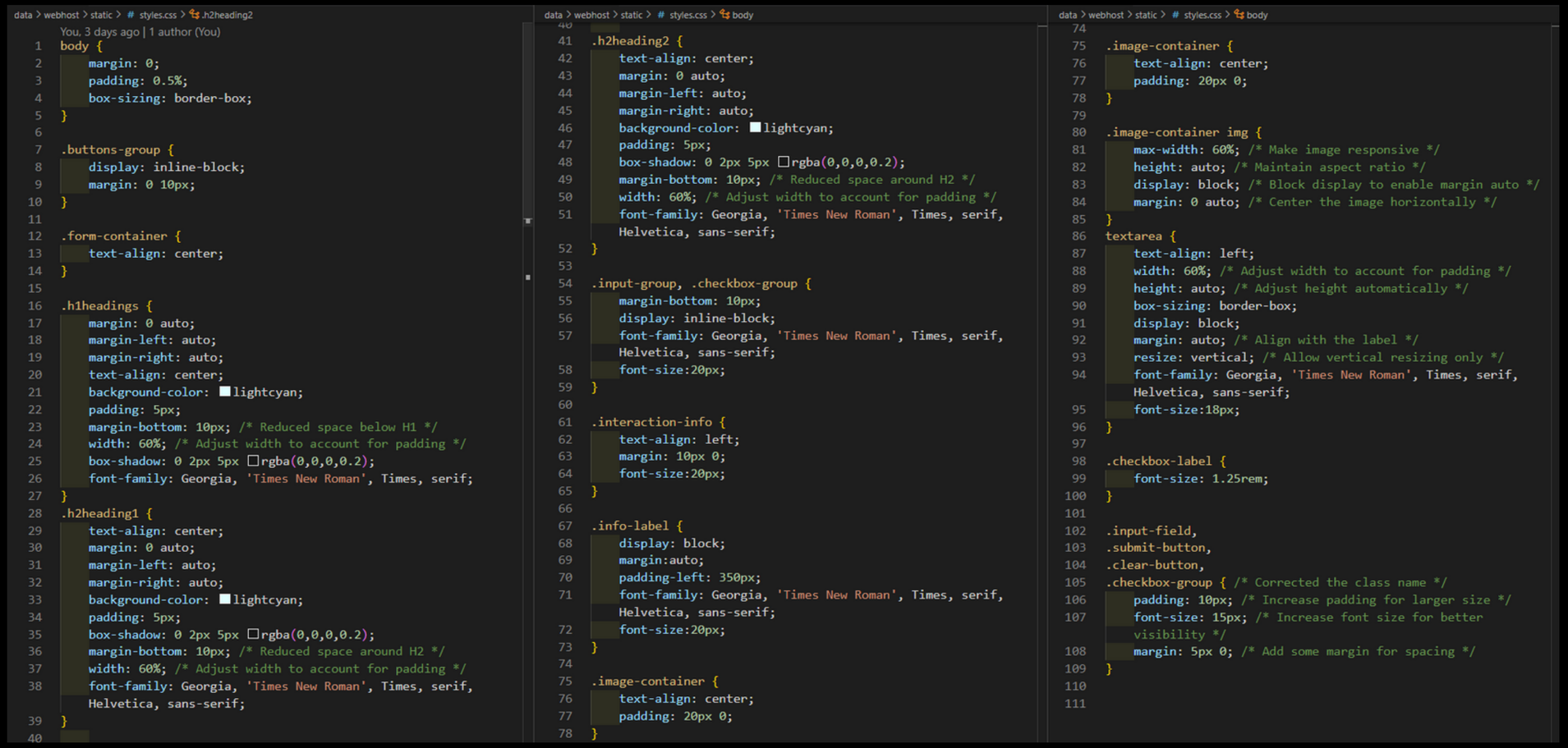
DATABASE UI

```
<!DOCTYPE html>      You, 4 days ago • Your message describing the changes
<html lang="en">
<head>


    <meta charset="UTF-8">
    <title>Database Query</title>
    <link rel="stylesheet" href="{{ url_for('static', filename='styles.css') }}">
    <script>
        function clearForm() {
            document.getElementById("drug_name1").value = '';
            document.getElementById("drug_name2").value = '';
            document.getElementById("drug_interaction_info").value = '';
            document.getElementById("food_interaction_info").value = '';
            document.getElementById("food_checkbox").checked = false;
        }
    </script>
</head>
<body>
    <div class="headings">
        <H1 class="h1headings">Drug Interaction Database Query Engine</H1>
        <H2 class="h2heading1">Which Interactions Are You Interested In?</H2>
    </div>
    <div class="form-container">
        <form method="POST" class="drug-form">
            <div class="input-group">
                <label for="drug_name1" class="input-label">Enter Drug Name 1:</label>
                <input type="text" id="drug_name1" name="drug_name1" class="input-field" value="{{
                drug_name1 }}">  
            </div>

            <div class="input-group">
                <label for="drug_name2" class="input-label">Enter Drug Name 2:</label>
                <input type="text" id="drug_name2" name="drug_name2" class="input-field" value="{{
                drug_name2 }}">
            </div><br>
```

```
            <div class="checkbox-group">
                <input type="checkbox" id="food_checkbox" name="food_checkbox" class="checkbox" {{
                'checked' if check_for_food }}>
                <label for="food_checkbox" class="checkbox-label">Check for Food Interactions</
                label>
            </div>

            <div class="buttons-group">
                <input type="submit" value="Submit" class="submit-button">
                <button type="button" onclick="clearForm()" class="clear-button">Clear</button>
            </div>
        </form>
    </div>
    <div class="headings">
        <H2 class="h2heading2">Interaction Results:</H2>
    </div>
    <div class="interaction-info">
        <label for="drug_interaction_info" class="info-label">Drug Interaction Info:</label>
        <textarea id="drug_interaction_info" name="drug_interaction_info" readonly style= "width:
        60%">
            {{ drug_interaction_info }}
            </textarea>
    </div>

    <div class="interaction-info">
        <label for="food_interaction_info" class="info-label">Food Interaction Info:</label>
        <textarea id="food_interaction_info" name="food_interaction_info" readonly style= "width:
        60%">
            {{ food_interaction_info }}
            </textarea>
    </div>
    <div class="image-container">
        <img src="{{ url_for('static', filename='druginteration.jpg') }}" alt="Drug Interaction
        Image">
    </div>
</body>
</html>
```

CSS CODE

DATABASE UI



data > webhost > static > # styles.css > .h2heading2

You, 3 days ago | 1 author (You)

```css
body {
    margin: 0;
    padding: 0.5%;
    box-sizing: border-box;
}

.buttons-group {
    display: inline-block;
    margin: 0 10px;
}

.form-container {
    text-align: center;
}

.h1headings {
    margin: 0 auto;
    margin-left: auto;
    margin-right: auto;
    text-align: center;
    background-color: lightcyan;
    padding: 5px;
    margin-bottom: 10px; /* Reduced space below H1 */
    width: 60%; /* Adjust width to account for padding */
    box-shadow: 0 2px 5px rgba(0,0,0,0.2);
    font-family: Georgia, 'Times New Roman', Times, serif;
}
.h2heading1 {
    text-align: center;
    margin: 0 auto;
    margin-left: auto;
    margin-right: auto;
    background-color: lightcyan;
    padding: 5px;
    box-shadow: 0 2px 5px rgba(0,0,0,0.2);
    margin-bottom: 10px; /* Reduced space around H2 */
    width: 60%; /* Adjust width to account for padding */
    font-family: Georgia, 'Times New Roman', Times, serif,
    Helvetica, sans-serif;
}
```

data > webhost > static > # styles.css > body

```css
.h2heading2 {
    text-align: center;
    margin: 0 auto;
    margin-left: auto;
    margin-right: auto;
    background-color: lightcyan;
    padding: 5px;
    box-shadow: 0 2px 5px rgba(0,0,0,0.2);
    margin-bottom: 10px; /* Reduced space around H2 */
    width: 60%; /* Adjust width to account for padding */
    font-family: Georgia, 'Times New Roman', Times, serif,
    Helvetica, sans-serif;
}

.input-group, .checkbox-group {
    margin-bottom: 10px;
    display: inline-block;
    font-family: Georgia, 'Times New Roman', Times, serif,
    Helvetica, sans-serif;
    font-size:20px;
}

.interaction-info {
    text-align: left;
    margin: 10px 0;
    font-size:20px;
}

.info-label {
    display: block;
    margin:auto;
    padding-left: 350px;
    font-family: Georgia, 'Times New Roman', Times, serif,
    Helvetica, sans-serif;
    font-size:20px;
}

.image-container {
    text-align: center;
    padding: 20px 0;
}
```

data > webhost > static > # styles.css > body

```css
.image-container {
    text-align: center;
    padding: 20px 0;
}

.image-container img {
    max-width: 60%; /* Make image responsive */
    height: auto; /* Maintain aspect ratio */
    display: block; /* Block display to enable margin auto */
    margin: 0 auto; /* Center the image horizontally */
}
textarea {
    text-align: left;
    width: 60%; /* Adjust width to account for padding */
    height: auto; /* Adjust height automatically */
    box-sizing: border-box;
    display: block;
    margin: auto; /* Align with the label */
    resize: vertical; /* Allow vertical resizing only */
    font-family: Georgia, 'Times New Roman', Times, serif,
    Helvetica, sans-serif;
    font-size:18px;
}

.checkbox-label {
    font-size: 1.25rem;
}

.input-field,
.submit-button,
.clear-button,
.checkbox-group { /* Corrected the class name */
    padding: 10px; /* Increase padding for larger size */
    font-size: 15px; /* Increase font size for better
    visibility */
    margin: 5px 0; /* Add some margin for spacing */
}
```

COMPONENT INTERACTION

- PowerShell Extracted XML Data
- Database Schema based on extracted data
- Insert Extracted XML Data
- Test Query Database
- Generate Python Script for Web Server
- Query Database with Python for Web Based User Interface
- Generate Web Interface that uses python Flask from user prompts

TEST TRIAL RESULTS

```sql
SELECT DRUG_ID, DRUG_Name
FROM DRUG
WHERE DRUG_ID = 'DB00500' OR DRUG_ID = 'DB00054'
SELECT FK_DRUG_ID, PRO_GenericName
FROM PRODUCT
WHERE FK_DRUG_ID = 'DB00500'OR FK_DRUG_ID = 'DB00054'
SELECT FK_DRUG_ID, INTER_DrugID, INTER_Description
FROM DRUG_INTERACTION
WHERE FK_DRUG_ID = 'DB00500' AND INTER_DrugID = 'DB00054'
SELECT FK_DRUG_ID, FOOD_Description
FROM FOOD_INTERACTION
WHERE FK_DRUG_ID = 'DB00500'
```

Results | Messages

	DRUG_ID	DRUG_Name
1	DB00054	Abciximab
2	DB00500	Tolmetin

	FK_DRUG_ID	PRO_GenericName
1	DB00054	Reopro
2	DB00054	Reopro
3	DB00054	Reopro
4	DB00500	Novo-tolmetin Capsules 400mg
5	DB00500	Tolectin 200 Tab 200mg
6	DB00500	Tolectin 400 Cap 400mg
7	DB00500	Tolectin Tab 600mg
8	DB00500	Tolmetin Sodium
9	DB00500	Tolmetin Sodium
10	DB00500	Tolmetin Sodium
11	DB00500	Tolmetin Sodium
12	DB00500	Tolmetin Sodium
13	DB00500	Tolmetin Sodium
14	DB00500	Tolmetin Sodium
15	DB00500	Tolmetin Sodium
16	DB00500	Tolmetin Sodium
17	DB00500	Tolmetin Sodium
18	DB00500	Tolmetin Sodium

	FK_DRUG_ID	INTER_DrugID	INTER_Description
1	DB00500	DB00054	The risk or severity of bleeding and hemorrhage can be increased when Tolmetin is combined with Abciximab.

	FK_DRUG_ID	FOOD_Description
1	DB00500	Avoid alcohol. Ingesting alcohol may increase the risk of developing an ulcer, or gastrointestinal bleed. Take with or without food. Food (or milk) can decrease oral bioavailability by 16%. Tolmetin may be taken with food to reduce

Drug Interaction Database Query Engine

Which Interactions Are You Interested In?

Enter Drug Name 1: Tolmetin Enter Drug Name 2: Abciximab

☐ Check for Food Interactions Submit Clear

Interaction Results:

Drug Interaction Info:

The risk or severity of bleeding and hemorrhage can be increased when Tolmetin is combined with Abciximab.

Food Interaction Info:

Avoid alcohol. Ingesting alcohol may increase the risk of developing an ulcer, or gastrointestinal bleed. Take with or without food. Food (or milk) can decrease oral bioavailability by 16%. Tolmetin may be taken with food to reduce

Drug Interaction Database Query Engine

Which Interactions Are You Interested In?

Enter Drug Name 1: Tolmetin Enter Drug Name 2: Reopro

☑ Check for Food Interactions Submit Clear

Interaction Results:

Drug Interaction Info:

The risk or severity of bleeding and hemorrhage can be increased when Tolmetin is combined with Abciximab.

Food Interaction Info:

Avoid alcohol. Ingesting alcohol may increase the risk of developing an ulcer, or gastrointestinal bleed. Take with or without food. Food (or milk) can decrease oral bioavailability by 16%. Tolmetin may be taken with food to reduce

Drug Interaction Database Query Engine

Which Interactions Are You Interested In?

Enter Drug Name 1: Tolectin Tab 600mg Enter Drug Name 2: Abciximab

☑ Check for Food Interactions Submit Clear

Interaction Results:

Drug Interaction Info:

The risk or severity of bleeding and hemorrhage can be increased when Tolmetin is combined with Abciximab.

Food Interaction Info:

Avoid alcohol. Ingesting alcohol may increase the risk of developing an ulcer, or gastrointestinal bleed. Take with or without food. Food (or milk) can decrease oral bioavailability by 16%. Tolmetin may be taken with food to reduce

DRUG INTERACTION LIVE DEMO

KNOWLEDGE GAINED

- Technology
  - Python Flask
  - Web Server
  - Web Based User Interfaces
- Data Extraction
  - XML data using PowerShell
- Project Collaboration

CHALLENGES FACED

- Technology
  - Web Development
  - PowerShell
  - nginx, jQuery, AJAX
  - Python Flask
- Large Dataset
- Data Extraction
  - XML
    - Data Types
    - Database Attributes
- Illness
- Other Course Demands

FUTURE PROJECT DIRECTIONS

- Predictive Analysis on trial drugs and their potential interactions.
- Improved UI with dynamic features.

THANK YOU!

QUESTIONS