Product: Restaurant-Depot

Use Case: COGS

Downloads
Database

Expenses

```
    * <b>Restaurant Depot<b>
```

-- Notes
io = status variables (0/1)

Environment (Packages / Libraries)

In [200]:

```
### Import Packages ###

# Data/tables
import numpy as np
import pandas as pd
from pandas import DataFrame as df
import csv

# Get requests from API/URL
import requests

# Automate web navigation
from selenium import webdriver
from selenium.webdriver.chrome.options import Options
from selenium.webdriver.common.by import By
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC
from selenium.common.exceptions import StaleElementReferenceException

# Time
import time
from datetime import datetime

# Files
import glob
import os

# Charts
import matplotlib.pyplot as plt

# RegEx
import re
```

executed in 12ms, finished 12:04:17 2022-09-09

Internal Configuration

In [ ]:
```
# Assets = images / non-source code resources (NOT made by me --> excel downl
# Resource = code files configuration files (MADE BY ME --> code)
```

--> Internal Paths (Assets/Resources)

In [381]:
```
### Directory Paths ###

# Path to: Drive
path_drive = fr'D:'

# Path to: Web-app
path_webapp = fr'{path_drive}\webmtbh'

# Path to: Resources below root folder (codes/config files)
path_resource = fr'{path_webapp}\resource'

# Path to: Assets below root folder (downloads)
path_assets = fr'{path_webapp}\assets'

# Path to: ChromeDriver (assets)
path_chromedriver = fr'{path_assets}\chromedriver.exe'

# Path to clients' files --> All Clients root dir (assets)
path_clients = fr'{path_assets}\clients'


# Path to store receipts (resource)
path_receipts  = ''
path_root = os.getcwd()
```

executed in 18ms, finished 23:09:17 2022-09-09

Client Assets

Client Input

--> Client Credentials

In [ ]:
```
### Restaurant Depot Login Credentials ###
restaurantdepot_client_username = 'yajushsharma1@gmail.com' # <Client Input>
restaurantdepot_client_password = 'Halowars999' # <Client Input>
```

--> Receipt time-range

In [382]:
```
### Restaurant Depot ###
#----------------------#
startdate_receipts = '01-01-2012'   # <Client Input>
startdate_receipts = startdate_receipts   # <Client Input>

enddate_receipts = '09-08-2022'   # <Client Input>
enddate_receipts = enddate_receipts   # <Client Input>
```

Client Directories

--> Client ROOT Directory (assets)

In [389]:
```
### Create Client_i Directory/Folder ###

# Get Client Name (var)
client_i = 'Yaju'  # <Client Input>
client_name = client_i

# List Existing Clients
list_existing_clients = glob.glob(fr'{path_clients}\*')
list_existing_clients = [c.split("\\")[-1] for c in list_existing_clients]

# Set Status == New-client(1) / Existing-client(0)
io_client_new = None

# CREATES new folder for new client
try:
    # Path to Client Directory (assets) (specific client)
    path_client_i = fr'{path_clients}\{client_name}'

    # Make directory for 'client_i'
    os.mkdir(fr'{path_client_i}')
    print('making..')
    io_client_new = 1

# IF folder for that client already exists
except FileExistsError:
    client_exists = 'exists'
    io_client_new = 0
    print(client_exists)
    # ISSUE: if 2 different clients have the same name?
    # SOLUTION: Primary Key
```

executed in 9ms, finished 23:15:30 2022-09-09

```
making..
```

--> Client CHILD Directories

In [398]:
```
# Directory: Expenses #
if io_client_new == 1:
    dir_expenses = fr'{path_client_i}\expenses'
    os.mkdir(dir_expenses)

# Directory: Expenses\Receipts #
if io_client_new == 1:
    dir_receipts = fr'{dir_expenses}\receipts'
    os.mkdir(dir_receipts)
```

executed in 17ms, finished 23:23:44 2022-09-09

Product

Product Variables

--> Product URLS

In [ ]:
```
restaurantdepot_url_signin = 'https://member.restaurantdepot.com'
restaurantdepot_url_receipts = 'https://member.restaurantdepot.com/receipts'
restaurantdepot_url_receipts_timerange = fr'https://member.restaurantdepot.co
```

Product Set-up

In [3]:
```
### Set-up Browser Automation ###

# Define driver download options
chrome_options = webdriver.ChromeOptions()
# prefs = {"savefile.default_directory" : 'D:\\_Yaju\\receipts_restaurant_dep
#          "download.default_directory" : 'D:\\_Yaju\\receipts_restaurant_dep
#          "download.prompt_for_download" : False,
#          "download.directory_upgrade" : True,
#          "safebrowsing.enabled" : False,
#          "safebrowsing.disable_download_protection" : True
#         }
prefs = {"savefile.default_directory" : 'D:\\_Yaju\\receipts_restaurant_depot
         "download.default_directory" : 'D:\\_Yaju\\receipts_restaurant_depot
         "download.prompt_for_download" : False,
         "download.directory_upgrade" : True,
         "safebrowsing.enabled" : False,
         "safebrowsing.disable_download_protection" : True
        }
chrome_options.add_experimental_option('prefs', prefs)

# Create Browser instance
driver = webdriver.Chrome(executable_path=path_chromedriver, options=chrome_c

# Open Browser with URL
driver.get(url_receipts)
driver.implicitly_wait(15)
driver.maximize_window()
```

executed in 24.3s, finished 19:57:37 2022-09-08

```
C:\ProgramData\Anaconda3\lib\site-packages\ipykernel_launcher.py:15: Deprecat
ionWarning: executable_path has been deprecated, please pass in a Service obj
ect
  from ipykernel import kernelapp as app
```

In [4]:
```
# ### Log-in ###

# # Enter username/email
# x_email = '//*[@id="email"]'
# el_email = driver.find_element(By.XPATH, x_email)
# WebDriverWait(driver, 10).until(EC.element_to_be_clickable((By.XPATH, x_emc

# # Enter password
# x_pw = '//*[@id="pass"]'
# el_pw = driver.find_element(By.XPATH, x_pw)
# WebDriverWait(driver, 10).until(EC.element_to_be_clickable((By.XPATH, x_pw)

# # Click Sign-in
# x_signin = '//*[@id="send2"]'
# WebDriverWait(driver, 10).until(EC.element_to_be_clickable((By.XPATH, x_sig

# # Set Start-Date parameter
# driver.implicitly_wait(20)
# driver.get(url_date_start)
```

executed in 4.88s, finished 19:57:53 2022-09-08

In [5]:
```
# ### Receipts ###

# # Locatre Receipts element
# x_receipts_all = '//*[@id="maincontent"]/div[3]/div[1]/div[2]/div[2]/ol/li
# el_receipts_all = driver.find_elements(By.XPATH, x_receipts_all)
```

executed in 51ms, finished 19:58:12 2022-09-08

In [6]:
```
# ### Download receipts ###

# # Define download link element
# x_download = 'div/div[4]/ul/li[2]/a'

# # Loop through all receipt download links
# for r in el_receipts_all:
#     el_download = r.find_element(By.XPATH, x_download)
#     el_download.click()
```

executed in 27m 26s, finished 20:25:40 2022-09-08

In [ ]:

In [ ]:

In [169]:

```
columns = [
    'date',
    'time',

    'description',
    'unit_quantity',
    'case_quantity',
    'price',
    'upc',

    'subtotal',
    'tax',
    'total',
    'amt_paid',
    'balance',
    'payment_name',

    'terminal',
    'invoice_num',
    'loc_rd',
    'loc_mm'
]
```

executed in 5ms, finished 11:17:55 2022-09-09

In [170]:

```
### Read Receipts ###

# Get list of csv files in directory
files = glob.glob(f'{path_receipts}/*')
temp_list_dict_r = []
inum = 1

for file in files:
    print(inum)
    inum = inum+1

    # import csv data
    with open(file, mode='r')  as f:
        reader = csv.reader(f)

        temp_list_reader = []
        for r_row in reader: # reader_row (r_row)
            temp_list_reader.append(r_row)

        dfr = df(temp_list_reader)

        # Terminal, Date, Time
        r_tdt = (dfr.iloc[4,1])
        r_tdt = r_tdt.split(' - ')
        # Terminal
        terminal = r_tdt[0]

        # Convert dtype to DateTime format
        r_dt = r_tdt[1]
        r_dt = datetime.strptime(r_dt, "%m/%d/%Y %I:%M %p")
        # Date
        r_date = (r_dt.date())
        # Time
        r_time = (r_dt.time())

        # Location: Mirch Masala (loc_mm)
        loc_mm = dfr.iloc[2,1]
        # Location: Restaurant Depot (loc_rd)
        loc_rd = dfr.iloc[0,0]
        loc_rd = loc_rd[:-4]

        # Invoice Number
        invoice_num = dfr.iloc[4,0]

        # Sub Total
        r_subtotal = dfr.iloc[-5,4]
        # Tax
        r_tax = dfr.iloc[-4,4]
        # Total
        r_total = dfr.iloc[-3,4]
        # Payment Name
        r_payment = dfr.iloc[-2,1]
        # Amount Paid
        paid = dfr.iloc[-2,4]
        # Balance
        balance = dfr.iloc[-1,4]

        # Inventory
        for i, row in dfr[7:-5].iterrows():

            upc = row[0]
```

```
            desc = row[1]
            unit_quantity = row[2]
            case_quantity = row[3]
            price = row[4]

            dict_r = {
                "date" : r_date,
                "time" : r_time,
        #         "type" : r_type, # purchase, refund

                "upc" : upc,
                "description" : desc,
                "unit_quantity" : unit_quantity,
                "case_quantity" : case_quantity,
                "price" : price,

                "subtotal" : r_subtotal,
                "tax" : r_tax,
                "total" : r_total,
                "payment_name" : r_payment, # card/method of payment
                "amt_paid" : paid,
                "balance" : balance,

                "terminal" : terminal,
                "invoice_num" : invoice_num,
                "loc_rd" : loc_rd, # location of restaurant depot
                "loc_mm" : loc_mm
            }

            temp_list_dict_r.append(dict_r)

    df_receipt = df(temp_list_dict_r, dtype=None, columns=columns)
```

executed in 4.40s, finished 11:18:01 2022-09-09

. . .

In [178]:

```
df_receipt['date'] = pd.to_datetime(df_receipt['date'], infer_datetime_format
df_receipt = df_receipt.sort_values(by=['date'], ascending=False)

# df_receipt.to_excel(f'{path_root}\inventory.xlsx', index=False) # change
df_receipt.to_excel(f'{path_root}\inventory.xlsx', index=False)
```

executed in 11ms, finished 11:24:37 2022-09-09

In [314]:

```
# list(df_receipt['payment_name'].unique())
# display(df_receipt['payment_name'].drop_duplicates(subset='') .value_counts
df_receipt.drop_duplicates(subset='payment_name').payment_name.vaue_counts()
```

executed in 27ms, finished 14:07:30 2022-09-09

. . .

In [304]:

```
start = datetime.strptime("2022/01/01", "%Y/%m/%d")
df_cogs = df_receipt[['date', 'amt_paid', 'invoice_num']][df_receipt['date']>
df_cogs = df_cogs.drop_duplicates(subset='invoice_num')
df_cogs['amt_paid'] = [float(re.findall(p, x)[0].replace(",", "")) for x in c


plt.figure(1, figsize=(20,5))
plt.plot(df_cogs['date'], df_cogs['amt_paid'])

plt.title('COGS')
plt.xlabel('Date', fontsize=25)
plt.ylabel('Spend', fontsize=25)
plt.locator_params(axis='y', nbins=10, tight=False)

params = {
    'xtick.labelsize':12,
    'ytick.labelsize':12
}

plt.rcParams.update(params)

plt.grid(True)
plt.show()
```

executed in 763ms, finished 13:06:37 2022-09-09