**Design Document for Web Scraping and Data Processing**

**1. Overview**

This document outlines the design and functionality of a Python script that performs web scraping using Selenium to gather bus route data and save it to an Excel file. The script also calls a data cleaning and insertion function after saving the data.

**2. Importing Libraries**

import os

import time

import pandas as pd

from datetime import datetime, timedelta

from selenium import webdriver

from selenium.webdriver.common.by import By

from selenium.webdriver.support.ui import WebDriverWait

from selenium.webdriver.common.action\_chains import ActionChains

from selenium.webdriver.support import expected\_conditions as EC

from selenium.webdriver.chrome.service import Service

from webdriver\_manager.chrome import ChromeDriverManager

from DataClean\_DB\_Insert import datacleandbinsert

* **os**: For interacting with the operating system (e.g., file handling).
* **time**: For managing time-related tasks (e.g., delays).
* **pandas**: For data manipulation and analysis.
* **datetime**: For handling date and time.
* **selenium**: For automating web browsing tasks.
* **webdriver\_manager**: For automatically managing browser drivers.
* **datacleandbinsert**: Custom module for data cleaning and database insertion.

**3. Function: scrabdata**

**3.1 Purpose**

The scrabdata function scrapes bus route data from the Redbus website based on a unique key, processes the data, and saves it to an Excel file.

**3.2 Parameters**

* unique\_key: A string representing the state-specific identifier to fetch the relevant data.

**3.3 Code Breakdown**

def scrabdata(unique\_key):

* **Purpose**: Define the function to start the scraping process.

state\_map = {

"Chandigarh\_CTU": ("chandigarh-transport-undertaking-ctu", "Chandigarh"),

...

}

* **Purpose**: Maps unique keys to state-specific route details.

stateroute, statename = state\_map.get(unique\_key, ("pepsu", "Punjab"))

* **Purpose**: Retrieves the state route and name based on the unique\_key.

driver = webdriver.Chrome(service=Service(ChromeDriverManager().install()))

driver.get(f'https://www.redbus.in/online-booking/{stateroute}/?utm\_source=rtchometile')

driver.maximize\_window()

wait = WebDriverWait(driver, 10) # Increase wait time for stability

* **Purpose**: Initializes the Selenium WebDriver and opens the specified URL.

all\_route\_names = []

all\_route\_links = []

j = 1

while True:

...

print(f"Page {j}")

...

try:

next\_page\_button = wait.until(EC.element\_to\_be\_clickable((By.XPATH, f'//div[contains(@class, "DC\_117\_pageTabs") and text()="{j + 1}"]')))

...

except Exception as e:

print(f"Could not move to the next page: {e}")

break

* **Purpose**: Collects route names and links from multiple pages, handling pagination.

routes = list(zip(all\_route\_names, all\_route\_links))

print(f"Total routes found: {len(routes)}")

* **Purpose**: Combines the collected route names and links into a list of tuples.

bus\_data = []

seen\_buses = set() # Set to track unique bus records

* **Purpose**: Initializes lists to store bus data and track unique records.

for route\_name, url in routes:

try:

driver.get(url)

...

last\_height = driver.execute\_script("return document.body.scrollHeight")

while True:

...

driver.execute\_script("window.scrollTo(0, document.body.scrollHeight);")

...

* **Purpose**: Iterates over each route link, scrapes bus details, and handles dynamic content loading.

df = pd.DataFrame(bus\_data)

* **Purpose**: Converts the collected data into a Pandas DataFrame.

now = datetime.now()

current\_date = now.strftime("%Y-%m-%d")

current\_time = now.strftime("%H-%M-%S")

folder\_name = "D:/Project"

if not os.path.exists(folder\_name):

os.makedirs(folder\_name)

file\_path = os.path.join(folder\_name, f"{statename}\_{current\_date}\_{current\_time}.xlsx")

* **Purpose**: Sets up the file path for saving the Excel file, including creating directories if they don't exist.

file\_created\_within\_10s = False

for file in os.listdir(folder\_name):

...

if file\_created\_within\_10s:

print("File already present from the last 10 seconds. Please try again later.")

else:

df.to\_excel(file\_path, index=False)

print(f"Data saved to {file\_path}")

* **Purpose**: Checks for recent files to avoid overwriting, then saves the DataFrame to an Excel file.

driver.quit()

datacleandbinsert(statename)

* **Purpose**: Closes the WebDriver and calls a function to clean and insert the data into the database.

**4. Conclusion**

This script automates the process of collecting bus route data from the Redbus website, processing it, and storing it in a structured format for further analysis. It handles dynamic content and pagination effectively and ensures data integrity by avoiding duplicate records and recent file overwriting.

**5. Appendix**

**5.1 Example Output**

Total routes found: 100

Data saved to D:/Project/Chandigarh\_2024-08-27\_12-30-45.xlsx