Cryptocurrency Price Tracker –Project Report

Author: Vemulapalli Sravya(Team 1),
Project Type: Individual Project

EXECUTIVE SUMMARY:

The Cryptocurrency Price Tracker project demonstrates the implementation of an automated real-time cryptocurrency data extraction system using Python and Selenium WebDriver. The tool scrapes live market data from CoinMarketCap and presents it in a structured tabular format in the console, with optional data export to CSV files for historical analysis. This system eliminates the dependency on manual checking of cryptocurrency prices by providing an automated and reliable mechanism for monitoring price movements, trends, and market capitalization. The application supports headless browser automation and filtering mechanisms for customized market analysis.

PROJECT OVERVIEW:

Objective:

To build an automated tool that collects real-time cryptocurrency market data from CoinMarketCap and stores it in a structured format for analysis and decision-making**.**

Scope:

- Real-time cryptocurrency data scraping
- Top 10 coin monitoring
- Price, market cap, and 24-hour change extraction
- CSV-based historical logging
- Filtering by price and percentage change
- Headless browser support

Key Features:

- **Live Scraping** – Fetches dynamic market data
- **Headless Mode** – Runs in background without UI
- **Data Export** – Stores pricing data in CSV format
- **Filtering** – Filter coins by price and 24h change
- **Top Gainers & Losers** – Identifies market movements
- **Timestamp Logging** – Enables trend analysis over time

TECHNICAL IMPLEMENTATION:

Architecture:

The system follows an automated scraping pipeline:

1. User Input Collection (headless / filters / coin limit)
2. WebDriver Initialization
3. Page Rendering via Selenium
4. DOM Parsing and Data Extraction
5. Data Processing using Pandas
6. CSV Export and Console Display
7. Termination of WebDriver Session

Technology Stack:

- Python 3.11 - Core programming language
- Selenium - Browser automation and scraping
- WebDriver Manager -Automatic driver handling
- Pandas - Data manipulation & CSV export
- CSV - Persistent storage
- Chrome WebDriver - Rendering JavaScript-based data

Performance Metrices:

- **Load Time:** 5–10 seconds
- **Data Capture Rate:** 10 coins per execution
- **Reliability:** High (JS-rendered content supported)
- **Storage Format:** CSV with timestamps

SOURCE CODE IMPLEMENTATION (Screenshot):

crypto_price_tracker_v2.py > ...

```python
import os
import time
from datetime import datetime
import pandas as pd

from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.chrome.service import Service
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC

from webdriver_manager.chrome import ChromeDriverManager


URL = "https://coinmarketcap.com/"
CSV_FILE = "crypto_prices_v2.csv"


def start_browser(headless=True):
    options = webdriver.ChromeOptions()
    if headless:
        options.add_argument("--headless=new")

    options.add_argument("--window-size=1920,1080")
    service = Service(ChromeDriverManager().install())
    return webdriver.Chrome(service=service, options=options)


def clean_number(text):
    return (
        text.replace("$", "")
            .replace(",", "")
            .replace("%", "")
            .replace("\n", "")
            .strip()
    )

```

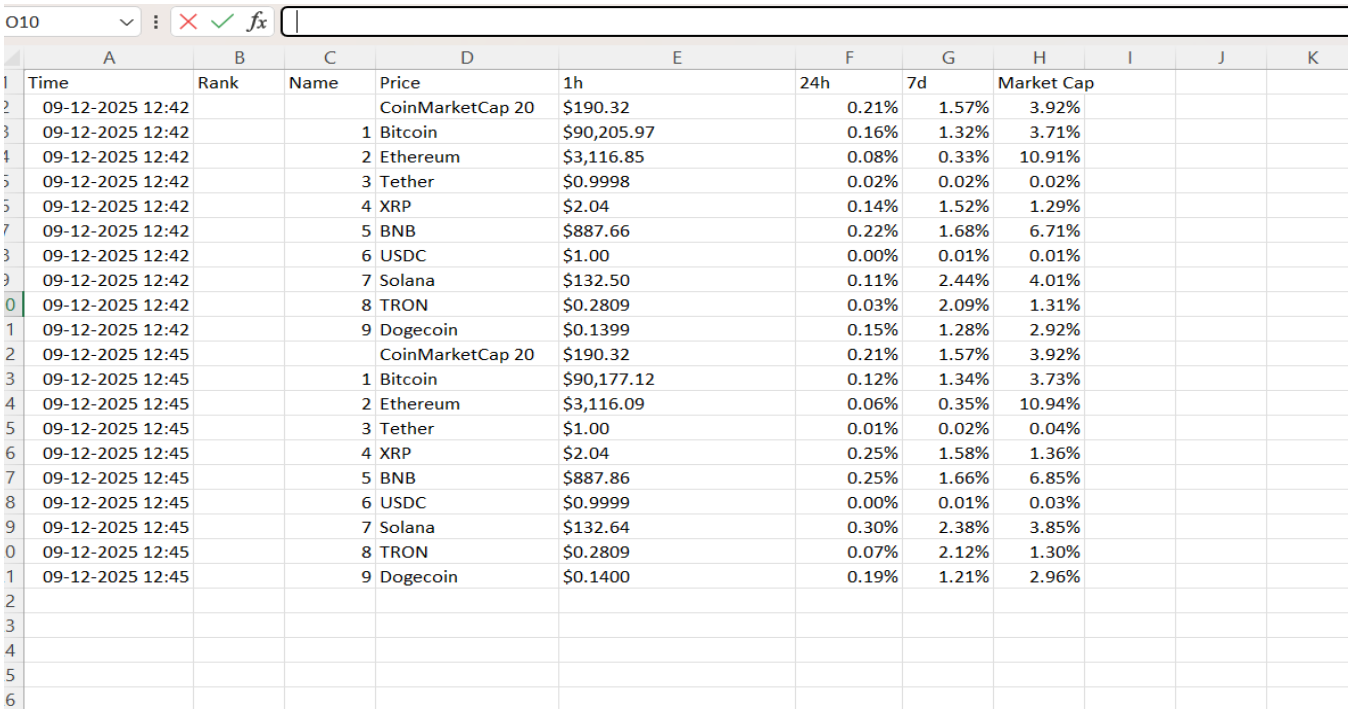
```python
def fetch_crypto_data(driver):
    driver.get(URL)
    wait = WebDriverWait(driver, 25)

    wait.until(EC.presence_of_all_elements_located((By.CSS_SELECTOR, "tbody tr")))
    rows = driver.find_elements(By.CSS_SELECTOR, "tbody tr")[:10]

    data = []
    timestamp = datetime.now().strftime("%Y-%m-%d %H:%M:%S")

    for row in rows:
        cols = row.find_elements(By.TAG_NAME, "td")
        if len(cols) < 7:
            continue

        try:
            coin = {
                "Time": timestamp,
                "Rank": cols[0].text,
                "Name": cols[1].text.split("\n")[0],
                "Price": cols[2].text,
                "1h": cols[3].text,
                "24h": cols[4].text,
                "7d": cols[5].text,
                "Market Cap": cols[6].text
            }
            data.append(coin)
        except:
            continue

    return data


def show_data(data):
    df = pd.DataFrame(data)
    print("\n========== LIVE CRYPTO DATA ==========")
    print(df.to_string(index=False))
```

```python
    return df


def save_to_csv(df):
    if os.path.exists(CSV_FILE):
        df.to_csv(CSV_FILE, mode="a", header=False, index=False)
    else:
        df.to_csv(CSV_FILE, index=False)

    print(f"\nSaved to {CSV_FILE}")


def find_gainers_and_losers(df):
    df["24h_num"] = df["24h"].apply(lambda x: float(clean_number(x)) if "%" in x else 0)

    gainers = df.sort_values("24h_num", ascending=False).head(3)
    losers = df.sort_values("24h_num").head(3)

    print("\n========== TOP 3 GAINERS (24H) ==========")
    print(gainers[["Name", "24h"]].to_string(index=False))

    print("\n========== TOP 3 LOSERS (24H) ==========")
    print(losers[["Name", "24h"]].to_string(index=False))


def filter_data(df):
    choice = input("\nApply filter? (y/n): ").lower()
    if choice != "y":
        return df

    try:
        min_price = input("Minimum Price ($): ")
        min_change = input("Minimum 24h change (%) : ")

        if min_price:
            df["Price_num"] = df["Price"].apply(lambda x: float(clean_number(x)))
```

```
            df["Price_num"] = df["Price"].apply(lambda x: float(clean_number(x)))
            df = df[df["Price_num"] >= float(min_price)]

        if min_change:
            df["24h_num"] = df["24h"].apply(lambda x: float(clean_number(x)))
            df = df[df["24h_num"] >= float(min_change)]

    except:
        print("Filter error. Showing full data.")
        return df

    return df


def main():
    print("=== Crypto Price Tracker V2 ===")

    headless = input("Headless mode (y/n): ").lower() == "y"

    driver = start_browser(headless)

    print("\nFetching data...")
    data = fetch_crypto_data(driver)
    driver.quit()

    if not data:
        print("No data fetched. Site layout may have changed.")
        return

    df = show_data(data)

    df = filter_data(df)

    if df.empty:
        print("\nNo data matched your filter.")
    else:
```

```
    else:
        print("\n========= FILTERED DATA =========")
        print(df.to_string(index=False))
        save_to_csv(df)
        find_gainers_and_losers(df)

    print("\nDONE ✅")


if __name__ == "__main__":
    main()
```

Extracted Job Result In Excel:



Time	Rank	Name	Price	1h	24h	7d	Market Cap
09-12-2025 12:42			CoinMarketCap 20	$190.32	0.21%	1.57%	3.92%
09-12-2025 12:42		1	Bitcoin	$90,205.97	0.16%	1.32%	3.71%
09-12-2025 12:42		2	Ethereum	$3,116.85	0.08%	0.33%	10.91%
09-12-2025 12:42		3	Tether	$0.9998	0.02%	0.02%	0.02%
09-12-2025 12:42		4	XRP	$2.04	0.14%	1.52%	1.29%
09-12-2025 12:42		5	BNB	$887.66	0.22%	1.68%	6.71%
09-12-2025 12:42		6	USDC	$1.00	0.00%	0.01%	0.01%
09-12-2025 12:42		7	Solana	$132.50	0.11%	2.44%	4.01%
09-12-2025 12:42		8	TRON	$0.2809	0.03%	2.09%	1.31%
09-12-2025 12:42		9	Dogecoin	$0.1399	0.15%	1.28%	2.92%
09-12-2025 12:45			CoinMarketCap 20	$190.32	0.21%	1.57%	3.92%
09-12-2025 12:45		1	Bitcoin	$90,177.12	0.12%	1.34%	3.73%
09-12-2025 12:45		2	Ethereum	$3,116.09	0.06%	0.35%	10.94%
09-12-2025 12:45		3	Tether	$1.00	0.01%	0.02%	0.04%
09-12-2025 12:45		4	XRP	$2.04	0.25%	1.58%	1.36%
09-12-2025 12:45		5	BNB	$887.86	0.25%	1.66%	6.85%
09-12-2025 12:45		6	USDC	$0.9999	0.00%	0.01%	0.03%
09-12-2025 12:45		7	Solana	$132.64	0.30%	2.38%	3.85%
09-12-2025 12:45		8	TRON	$0.2809	0.07%	2.12%	1.30%
09-12-2025 12:45		9	Dogecoin	$0.1400	0.19%	1.21%	2.96%

KEY ACHIEVEMENTS:

Technical Accomplishments:

- Successful Selenium integration with CoinMarketCap
- Dynamic JavaScript scraping capability
- Effective browser automation in both UI and headless mode
- Accurate historical logging using CSV
- Automatic ChromeDriver management
- Advanced filter logic for price analysis

Process Automation Benefits:

- Saves manual effort
- Provides accurate real-time data
- Enables offline data analysis
- Reusable for data science and visualization projects

LIMITATIONS AND CONSTRAINTS:

Current Limitations:

- Limited to CoinMarketCap only
- No real-time dashboard
- Internet required
- Static file export only (CSV)

Resource Constraints:

- Dependent on website layout
- Scraping speed dependent on network
- No cloud database support

PROJECT IMPACT AND VALUE:

- **Immediate Benefits:**
  - Reduces effort in tracking crypto prices
  - Automates repetitive market checking
  - Allows structured data retention
- **Long-Term Potential:**
  - Dashboard integration
  - Machine learning trend prediction
  - Portfolio analytics
  - Alert systems
  - Database storage

CONCLUSION:

The Cryptocurrency Price Tracker project successfully demonstrates real-time data automation using Selenium and Python. It provides an efficient tool for cryptocurrency monitoring while enabling historical data analysis through CSV logging. The project showcases strong fundamentals in automation, scraping, exception handling, and data processing. With further enhancements such as dashboards and AI-based predictions, this tool can be expanded into a full-fledged crypto analytics platform.