PDS Assignment

Jithendra Pavuluri

16343746

Question – 1

A)

```
#A)
#Accesing Data in JSON Format
import requests
url = "https://coinranking1.p.rapidapi.com/coins"
querystring = {"referenceCurrencyUuid":"yhjMzLPhuIDl","timePeriod":"24h","tiers[0]":"1","orderBy":"marketCap","orderDirection":"desc","limit":"50","o
headers = {
  "X-RapidAPI-Key": "aa650591dfmshe9f295a7128aafep176eddjsn1ea120362b22",
  "X-RapidAPI-Host": "coinranking1.p.rapidapi.com"
}
response = requests.get(url, headers=headers, params=querystring)
print(response.json())
```

```
{'status': 'success', 'data': {'stats': {'total': 783, 'totalCoins': 37182, 'totalMarkets': 43680, 'totalExchanges': 178, 'totalMarketCap': '239188986
```

 # A) Fetching Data:

1. It utilizes the requests library to send an HTTP GET request to the Coinranking API endpoint.

2. Query parameters are provided to specify details of the request, such as the reference currency, time period, tiers, order, limit, and offset. These parameters help filter and sort the data to retrieve the top 10 coins by market capitalization.

3. The response received from the API is in JSON format, containing information about the requested coins.

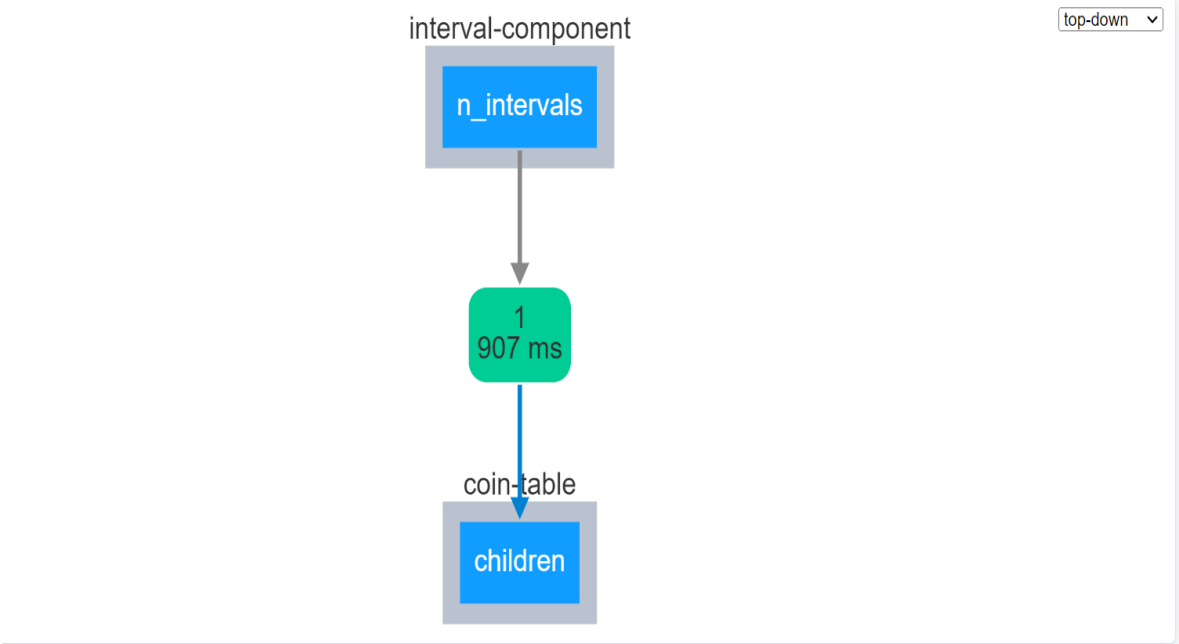
B)

```
#B)
#Displaying the data using Dash
import dash
from dash import html
from dash import dcc
from dash.dependencies import Input, Output
import requests

# Initialize Dash app
```

```
app = dash.Dash(__name__)

# Function to fetch coin data from the Coinranking API
def fetch_coin_data():
    url = "https://coinranking1.p.rapidapi.com/coins"
    querystring = {
        "referenceCurrencyUuid": "yhjMzLPhuIDl",
        "timePeriod": "24h",
        "tiers[0]": "1",
        "orderBy": "marketCap",
        "orderDirection": "desc",
        "limit": "10",  # Fetching top 10 coins
        "offset": "0"
    }
    headers = {
        "X-RapidAPI-Key":
"aa650591dfmshe9f295a7128aafep176eddjsn1ea120362b22",
        "X-RapidAPI-Host": "coinranking1.p.rapidapi.com"
    }
    response = requests.get(url, headers=headers, params=querystring)
    return response.json()

# Define app layout
app.layout = html.Div([
    html.H1("Top 10 Coins by Market Cap"),
    html.Div(id='coin-table'),
    dcc.Interval(
        id='interval-component',
        interval=60*1000,  # Update every minute
        n_intervals=0
    )
])

# Callback to update coin data every minute
@app.callback(
    Output('coin-table', 'children'),
    [Input('interval-component', 'n_intervals')]
)
def update_coins_table(n_intervals):
    coin_data = fetch_coin_data()
    if coin_data and 'data' in coin_data:
        coins = coin_data['data']['coins']
        rows = [html.Tr([html.Td(coin['name']), html.Td(coin['symbol']),
html.Td(coin['marketCap'])]) for coin in coins]
        return html.Table([
```

```
            html.Thead([
                html.Tr([html.Th('Name'), html.Th('Symbol'),
html.Th('Market Cap')])
            ]),
            html.Tbody(rows)
        ])
    else:
        return "Failed to fetch data"

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

Output:



 # B) Displaying Data Using Dash:

1. The Dash app is initialized with **dash.Dash(__name__)**.

2. There's a function **fetch_coin_data()** defined to fetch coin data from the Coinranking API. It constructs the API request using the specified parameters and retrieves the JSON response.

3. The layout of the Dash app is defined within an **html.Div** element. It includes an H1 header indicating the purpose of the displayed data and a div with an id of 'coin-table'. Additionally, a **dcc.Interval** component is included to trigger updates at regular intervals (every 60 seconds in this case).

4. A callback function update_coins_table() is defined using the @app.callback decorator. This function is triggered by changes in the n_intervals property of the dcc.Interval component. It fetches fresh coin data using the fetch_coin_data() function and updates the content of the 'coin-table' div element with an HTML table containing information about the top 10 coins.

5. Finally, the Dash app is started with app.run_server(debug=True).

When you run this code, it launches a Dash server and provides a web page with a table displaying information about the top 10 cryptocurrencies by market capitalization. The table content is updated automatically every minute due to the **dcc.Interval** component triggering the callback function at regular intervals. If there's an issue in fetching data or rendering the page, it will display "Failed to fetch data".