**Real-Time Stock Market Data Fetcher**

* Introduction

This project is a Python script that fetches real-time stock market data from the Alice Blue trading platform and displays it in an Excel sheet. The script establishes a WebSocket connection with the Alice Blue platform, subscribes to a list of stocks, and continuously updates an Excel sheet with the latest market data for those stocks.

* Features
* Fetches real-time stock market data from the Alice Blue trading platform.
* Supports multiple stocks from different exchanges.
* Displays the fetched data in an Excel sheet.
* Continuously updates the Excel sheet with the latest market data.
* Handles WebSocket connection events (open, close, error).
* Stores market data in a dictionary for easy access.
* Technologies Used
* Python
* Alice Blue trading platform API
* WebSocket protocol
* Pandas library for data manipulation
* XlWings library for interacting with Excel
* Code Structure

The code is structured as follows:

* **Imports:** The script imports necessary modules and libraries, including “alice\_credentials” for authentication with the Alice Blue platform, “datetime” for working with dates and times, “json” for handling JSON data, “pandas” for data manipulation, and “xlwings” for interacting with Excel.
* **Authentication and Excel Setup:** The script creates an instance of the Alice Blue object “alice” using the “login()” function from the “alice\_credentials” module. It also opens an Excel workbook named "Stocks\_Scanner.xlsx" and selects the first worksheet ("Sheet1").
* **Instrument List:** The script reads the contents of the first 200 rows from columns A and B in "Sheet1" and stores the exchange and symbol pairs in a list called “instruments”.
* **WebSocket Functions:** The script defines several functions to handle WebSocket events, such as “socket\_open()”, “socket\_close()”, “socket\_error()”, and “feed\_data()”. These functions are used to handle the WebSocket connection, receive real-time market data, and store it in a dictionary “data”.
* **WebSocket Connection:** The script starts the WebSocket connection using “alice.start\_websocket()” and waits for the connection to be established.
* **Data Retrieval and Display:** Once the WebSocket connection is open, the script enters a loop where it:

a. Creates a list “subscribe\_list” containing the instruments to subscribe to.

b. Subscribes to the instruments using “alice.subscribe(subscribe\_list)”.

c. Waits for the “data” dictionary to be populated with market data for all subscribed instruments.

d. Creates a pandas DataFrame “df” from the “data” dictionary.

e. Prints the DataFrame “df” to the console.

f. Writes the DataFrame “df” to the Excel sheet starting from cell C1 using “sht.range('C1').value = df”.

* Usage

1. Install the required Python libraries (`alice\_credentials`, `pandas`, `xlwings`).

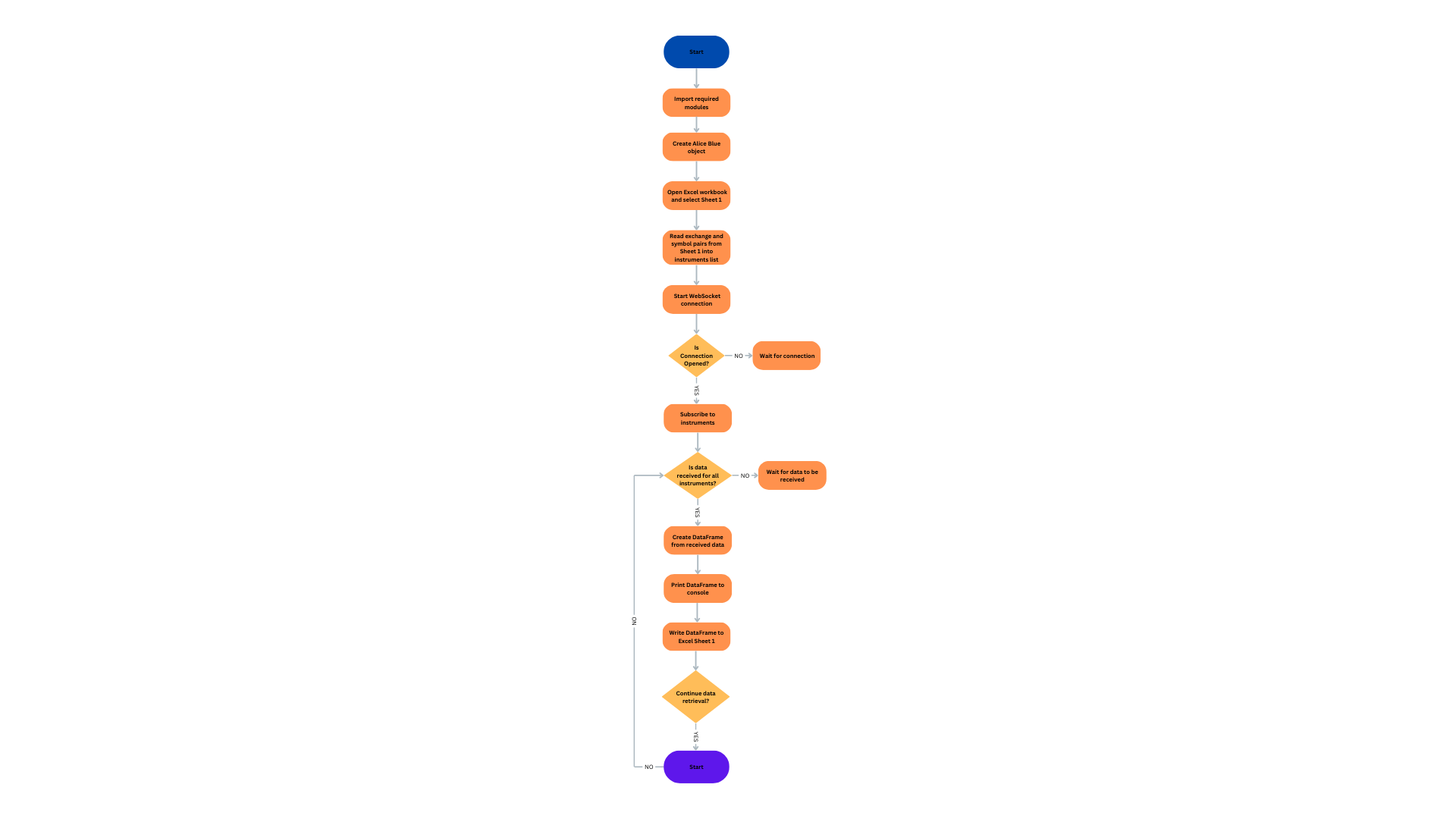
2. Set up the `alice\_credentials` module with your Alice Blue trading platform credentials.

3. Create an Excel workbook named "Stocks\_Scanner.xlsx" with the list of stocks you want to fetch data for (exchange and symbol in columns A and B, respectively).

4. Run the Python script.

5. The script will establish a WebSocket connection with the Alice Blue platform, fetch real-time market data for the specified stocks, and display it in the "Stocks\_Scanner.xlsx" Excel sheet.

* Future Enhancements
* Add error handling and logging mechanisms for better debugging and error reporting.
* Implement a graphical user interface (GUI) for better user experience.
* Add support for user-defined refresh intervals or real-time updates.
* Implement data caching or storage mechanisms for historical data analysis.
* Add support for additional data sources or trading platforms.
* Conclusion
* This project demonstrates the use of Python, WebSocket protocol, and various libraries to fetch real-time stock market data from the Alice Blue trading platform and display it in an Excel sheet. It showcases the integration of different technologies and can be used as a starting point for building more complex stock market analysis tools or trading applications.

FLOWCHART: