Options Chain Tool

TEAM E-D1LLV1Z

Presented by: Swarangi Kule, Akash Shetgar,Aakash Wagle,Dwayne Vaz

TABLE OF CONTENT

PROBLEM STATEMENT

TO BUILD AN OPTIONS CHAIN TOOL:

Develop a real-time options chain screen as a webpage that processes market data received over TCP/IP and calculates implied volatility (IV). The webpage should also display expiry date, highlighting "in the money" and "out of money" options. It should be updated in real-time as the market data changes, without requiring a browser reload. The system should also consider a risk-free interest rate of 5% for IV calculations and handle accurate time-to-maturity (TTM) by assuming the expiry time as 15:30 IST on the expiry day.

GOALS AND OBJECTIVES

The main goal of the project is to create a dynamic webpage that displays an options chain screen similar to platforms like NSE India's option chain. The options chain should show data for various underlying assets , highlighting "in the money" and "out of money" options.

The system should receive market data over TCP/IP and process it to extract relevant information for options pricing.
Using the Black Scholes formula, the system should calculate implied volatility (IV) for the options based on their prices and other parameters.

OUR APPROACH TOWARDS THE PROBLEM STATEMENT

01 Establishing TCP/IP Connection: We Set up a TCP/IP connection to receive real-time market data. This was achieved by running the provided "feed-play" server and configuring the system to listen to the specified port. We Used the '**net**' module in Node.js to establish the TCP connection and receive data streams.

02 Parse Market Data: After Receiving the market data packets over the TCP/IP connection we extracted relevant information. Analyzed the packet structure described in the problem statement and use appropriate techniques, such as bitwise operations and string manipulation, to extract the required fields and converted the extracted data into JSON objects.

Market data was processed in 130-byte chunks with error handling of the buffer memory giving the appropriate relevant extracted data transformed into JSON.

03 We then calculated the Implied Volatility (IV for the options contracts using the Black-Scholes formula using mathematical functions and formulas keeping all the assumptions mentioned in the problem statement.

OUR APPROACH TOWARDS THE PROBLEM STATEMENT

04

We created a frontend application using React and material UI libraries to present the data in a tabular format . We used libraries like Socket.IO to establish a bidirectional connection between the frontend and backend servers to enable real-time data transmission from the backend to the frontend.

05

Real-time Updates: Utilized Socket.IO to push the processed data from the backend server to the frontend in real-time. Emit events from the backend server whenever new market data is received. Update the React state with the received data, triggering automatic re-rendering of the options chain screen.

OUR APPROACH TOWARDS THE PROBLEM STATEMENT

Tech Stack used-

Node js

Node.js was used as the server-side runtime environment to handle the back-end operations. It provided a scalable and event-driven architecture, allowing for efficient handling of network connections, data processing, and communication with external data sources.

JavaScript

JavaScript was used for both the front-end and back-end development. It enabled dynamic and interactive functionality on the client-side and facilitated server-side scripting with Node.js.

Socket Programming

Socket programming, specifically using Socket.IO, enabled efficient bidirectional communication between the server and client. This facilitated real-time transmission of market data to the React frontend.

React

React, a JavaScript library, was used to build the front-end user interface. It facilitated efficient seamless updates of the options chain display through its component-based architecture

Additional Tools and Libraries:

- Git: A version control system for tracking code changes and collaboration.
- npm (Node Package Manager): A package manager for installing and managing project dependencies.
- IDE (Integrated Development Environment): A code editor such as Visual Studio Code or Atom for writing and managing the project code.

This tech stack combines the power of Node.js for server-side development, React for building the frontend, and various supporting libraries and tools to handle real-time data transmission, state management, and code organization.

Why our application/solution should be considered for the final round

Key Points-

1. Our solution addresses all the key requirements and objectives outlined in the problem statement. We have implemented a robust system for handling real-time market data, calculating implied volatility, and displaying an options chain.

2. Efficient Data Processing: We have optimized the data processing workflow by handling data in small, manageable chunks. This approach ensures efficient processing and minimizes the risk of errors or performance issues.

3. Error Handling: We have implemented robust error-handling mechanisms to handle potential issues while accessing the TCP buffer. This ensures the stability and reliability of the system, allowing for smooth data transmission and processing. We have also used try-catch blocks at appropriate places in our code which safeguards the application from crashing

Why our application/solution should be considered for the final round

4. Real-Time Updates: By utilizing socket programming and the Socket.IO library we enables real-time updates of the market data on the front-end React application, providing users with the latest information.

5. User-Friendly Interface: Our solution uses React, a popular JavaScript library, for building a user-friendly interface. React's component-based architecture enables efficient updates of the options chain display. The interface is easy to navigate, enhancing the user experience.

6. Scalability: Our solution has been designed with scalability in mind. The architecture and technologies used allow for easy scalability to handle larger data volumes .

Why our application/solution should be considered for the final round

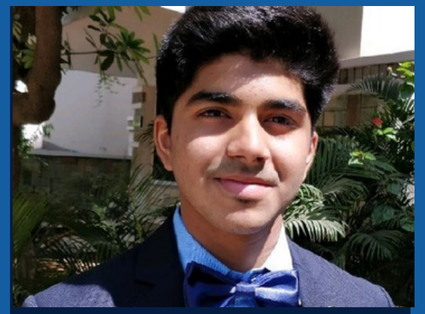
8. Documentation and Support: We provide comprehensive documentation that outlines the implementation details, architecture, and usage instructions of our solution. Additionally, our team is available to provide support and address any questions or issues that may arise during the implementation process.The necessary contact details of our team has been mentioned on our webpage.

9.Considering these factors, our application/solution offers a robust, efficient, and user-friendly solution for the given problem statement. We believe it deserves serious consideration for the final round based on its technical merits and the value it can bring to the intended users or stakeholders.

OUR TEAM

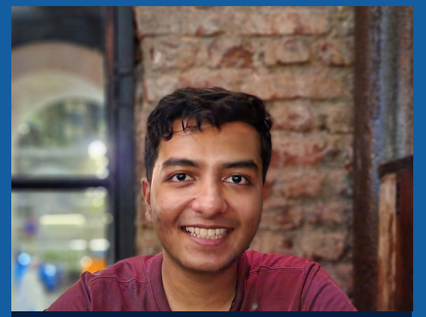
Aakash Wagle

Developer/Network Engineer



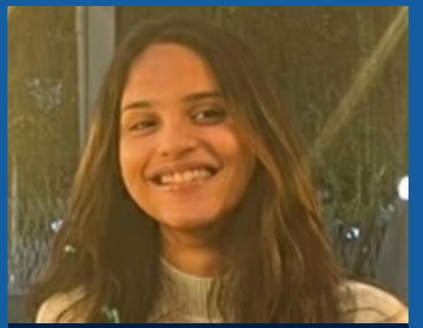
Akash Shetgar

Developer/Network Engineer



Dwayne Vaz

Developer/Analyst



Swarangi Kule

Developer/Analyst