StockMonitor

Project Objective: To be able to monitor its stock prices of companies

Dependencies:

A. Java SDK 1.8
B. Maven framework
C. IntelliJ IDE
D. Apache Tomcat

Assumptions:

A. The application depends on YAHOO API as data source.
B. Stocks are monitored every 2 minutes.

Tasks:

The user can:
A. Add a new company to be monitored (addCompany)
B. View list of companies and their last read stock prices (listCompanies)
C. Delete a company (deleteCompany)
D. View company history (getHistory)

The system on its own updates the stock values for companies in the database.

System Functioning:

<u>Standalone Application - Fetch Stock</u>
A. This is a simple multithread application which acts as the monitor.
B. It fetches stock prices for all added companies, spawning new thread requests for each.
C. After the thread executes, the results received are saved into the database.

<u>RESTful API</u>

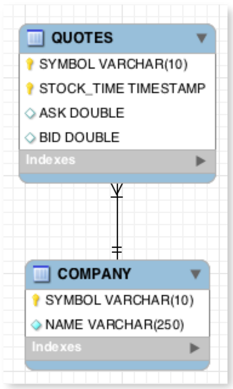
A. It is a Java Spring based application providing RESTful services for the tasks as listed below:

API Feature	URL	URL Params	Response
Add Company	/addCompany?symbol=YHOO	**Required** symbol=[valid ticker symbol]	**Success** {“message":"SUCCESS","reason":"Yahoo has been added to the database and will be monitored","suggestions":null} **Failure** {“message":"FAILURE","reason":"Symbol could not be found","suggestions":null} **Duplicate Entry** {"message":"FAILURE","reason":"YHOO already exists in the database","suggestions":null}
Delete Company	/deleteCompany?symbol=YHOO	**Required** symbol=[valid ticker symbol]	**Success** {“message”:"SUCCESS","reason":"Deleted successfully"} **Failure** {“message”:”FAILURE","reason":"Company Symbol could not be found"}
List Companies	/listCompanies	No Param Required	**Success** {"companyList":[{“symbol”:”...","name":"...","timestamp":1...,"ask": … ,”bid”: …},}],”count":10,"message":"SUCCESS","reason":"Companies found from DB”} **Failure** {“companyList": null,”count":0,"message":"FAILURE","reason":"No Company found in the database. Add company to view list.”}
Company History	/getHistory?symbol=YHOO	**Required** symbol=[valid ticker symbol]	**Success** {"companyList":[{“symbol”:”...","name":"...","timestamp":1...,"ask": … ,”bid”: …},}],”count":10,"message":"SUCCESS","reason":"Company history found”} **Failure** {“companyList": null,”count":0,"message":"FAILURE","reason":"Company symbol empty.”}

B. It is built using SpringBoot for rapid sprint based REST API development.

Database Model

A. The database has two tables - Company and Quotes
B. The company table stores Company name and symbol, Quotes table stores company symbol with ask price, bid price and timestamp of data.



C. Company has a one-to-many relationship with Quotes table.

Initial Setup:

API Code Setup

A. Unzip the code from Stock.zip
B. Import the project to an IDE (IntelliJ preferred)
C. Run the pom.xml file to resolve all dependencies.

Client Code Setup

D. Unzip the folder CompanyClient.zip
E. Import the project to an IDE (IntelliJ preferred)

Database Setup

F. The stocks.sql script has the DDL commands to create db and the required tables.
```
mysql -u username -p password database_name < stocks.sql
```

Execution Steps:

A. From the IDE, run “Fetchstock.java”.
`File path: stocks/complete/src/main/java/com/chetana/stockapp/standalone/Fetchstock.java`
B. Run “Application.java”.
`File path: stocks/complete/src/main/java/hello/Application.java`
C. (Optional) Run the test file “CompanyControllerTest.java”.
`File path: stocks/complete/src/main/java/test/CompanyControllerTest.java`
D. Run the index.html file on tomcat server to emulate client.

Sample Inputs:

Adding Companies

A. addCompany/symbol=GOOG

B. addCompany/symbol=YHOO
C. addCompany/symbol=MSFT
D. addCompany/symbol=GOOG
E. addCompany/symbol=TGT

List of Companies

F. listCompanies

Obtaining History

G. getHistory/symbol=GOOG

Deleting Companies

H. deleteCompany/symbol=GOOG