# Requirements

## Application Phase 1

Initially I want my application to have basic features:

1. Ability to store candlestick data in a continuous manner for daily timeframe for a limited number of Stocks(F&O).
2. Ability to implement atleast one simple strategy(Decide on which one)
3. Ability to send me messages about potential trade opportunities on a realtime basis.
4. Ability to run independently on servers(Preferably free servers for now)

## Application Phase 2

Later on I want my application to have more advanced features such as multiple strategies and ability to test a strategy on previous data and ability to paper trade and show the potential of each strategy.

1. Ability to run multiple strategies( Any number of strategies based on user input)
2. Ability to test any particular strategy for efficiency by paper trading.
3. Ability to back up the data in the database in a continuous manner.

# Application TimeLine:

PHASE 1

31st March – 20th April , 2018

PHASE 2

30th June – 31st July , 2018

# Application Architecture:

## High Level Design

**Web Service API Layer**

(Data Vendor Specific)

**Data Vendor**

Zerodha / Global DataFeeds / Alpha Vantage

**Data Storage Logic**

Logic to query from API Layer on periodic basis(Data Vendor Specific)

Logic to store the data in the database(Data Vendor Independent)

**Database**

My Sql Database with Hourly and Daily Candle information.

Database design needs to be generic , independent of the Data Vendor.

**Strategy Implementation**

Separate logic for separate strategies.

**Notification**

Email / Text Notifications

**Strategy Result Analysis**

Separate logic for testing each strategy and recording the results in database

# Data Vendors:

|  |  |  |
| --- | --- | --- |
| **Name** | **Cost** | **Comments** |
| ***Zerodha*** | 2K to 4K per month |  |
| ***Globaldatafeeds*** | 5K per month for 50 symbols across all exchanges. |  |
| ***Quandl*** | Free but with one day’s delay or Paid with approximately 3K cost. |  |
| ***AlphaVantage*** | Free EOD and other time frames but not very reliable data. |  |
| ***NSE Bhav Copy*** | Free EOD data in csv format for all EQ | Can start with this set of free data.  Need to check how soon it is available on nse website. |

# Database Design:

### watch\_list Table:

List of Stocks and Markets which need to be kept track of:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Column Name** | Symbol | Market | VendorId | Active |
| **Column Type** | Varchar | Varchar | Varchar | Boolean |
| **Constraints / Relations** | PK | |  |  |

### trading\_holidays table:

List of dates for which stocks need to be kept track of. This table contains a list of the last 100 trading dates for the stock market excluding holidays. This table should contain only the last 100 trading dates and remove older dates.

|  |  |  |
| --- | --- | --- |
| **Column Name** | Trading\_Date | Market |
| **Column Type** | Date | NSE |
| **Constraints / Relations** | PK | |

### daily\_candlesticks\_foTable:

Common table for all symbols. Stores 100 candles per symbol for 200 Symbols. So maximum 20000 records.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Column Name** | Symbol | Market | Date | Open | High | Low | Close | Volume | Change | Change(%) |
| **Column Type** | Varchar | Varchar | Date | Double | Double | Double | Double | Big Int | Double | Double |
| **Constraints / Relations** | PK | | |  |  |  |  |  |  |  |

### daily\_candlesticks\_fo\_calculations Table:

Common table for all symbols. All these values will be calculate on a daily basis and updated in this table either via code or via Procedural SQL.

This table will contain upto 20 candles( 1month) of dates for a particular symbol hence 20 records per symbol, because of performance reasons.

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Column Name** | Symbol | Market | Date | Change | Change(%) | EMA 20 | EMA 50 | RSI | Avg Vol | Avg Candle Body | Avg  Candle  Height |
| **Column Type** | Varchar | Varchar | Date |  |  | Double | Double | Big Int |  |  |  |
| **Constraints / Relations** | PK + FK from above table | | |  |  |  |  |  |  |  |  |

# Algorithm Logic:

### Strategy 1:

Strategy to find and identify breakout patterns on high volume and big candles:

1. Data needed :
2. To identify the range of the stock.
3. To identify a candle with a big body.
4. To identify a spurt in volume.

The below query runs everyday:

***Select*** *Symbol, Date from* ***daily\_candlestick\_calculation*** *where* ***{******(****current candle > 1.5\*Avg Candle****)******+*** ***(****current volume > 1.5\*Avg Volume****) + (****Stock’s close has extended the range of last 10 / 20 / 30/ 50 / 100 candles****) } and*** *date* ***=today’s date***

***Insert*** *into* ***daily\_candlestick\_calculation*** *column* ***signal****= Symbol abc formed pattern dce on such date.*

# Initial Setup:

## GIT

git init

git add README.md

git commit -m "first commit"

git remote add origin https://github.com/play-area/stock-api.git

git push --set-upstream origin master

username : [kulbir2408@gmail.com](mailto:kulbir2408@gmail.com)

password : ks9556462141

## ECLIPSE SETUP:

1. Add an apache server to Eclipse:
2. Import Project as Maven project into eclipse.
3. Navigate to Project’s Home Directory where pom.xml file is kept in command prompt and run the mvn clean install command.
4. Add the following settings to Eclipse for a Maven Dependencies.
5. You need to add the "Maven Dependency" in the Deployment Assembly

* right click on your project and choose properties.
* click on Deployment Assembly.
* click add
* click on "Java Build Path Entries"
* select Maven Dependencies"
* click Finish.
* Rebuild and deploy again

# NSE BHAV COPY:

## URL:

<https://www.nseindia.com/content/historical/EQUITIES/2018/JAN/cm05JAN2018bhav.csv.zip>

## File Structure :

The file is a CSV file containing information about all symbols in NSE for a particular day:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **SYMBOL** | **SERIES** | **OPEN** | | **HIGH** | **LOW** | **CLOSE** | **LAST** | **PREVCLOSE** | **TOTTRDQTY** | **TOTTRDVAL** | **TIMESTAMP** | **TOTALTRADES** | **ISIN** |
| 3MINDIA | EQ | 19440 | 19600.05 | | 19255.35 | 19367.5 | 19350 | 19449.8 | 867 | 16855986.45 | 05-Jan-18 | 407 | INE470A01017 | |

# KITE CONNECT API:

Unless stated otherwise, every request requires the following arameters.

#### api\_key

Every request before and after authentication requires the api\_key

#### access\_token

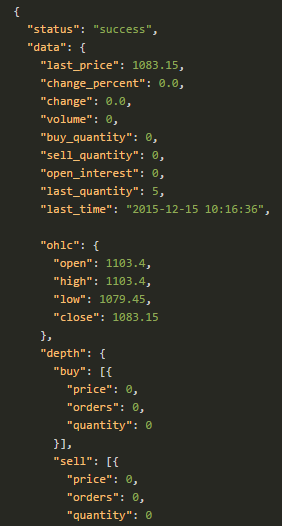
Every request after authentication requires th eaccess\_token along with api\_key

## LIVE QUOTES :

### Request:

"https://api.kite.trade/instruments/NSE/INFY?api\_key=xxx&access\_token=yyy

### Response:



## HISTORICAL QUOTES :

### Request:

https://api.kite.trade/instruments/historical/5633/minute?from=2015-12-28&to=2016-01-01&api\_key=xxx&access\_token=yyy

| **parameter** |  |
| --- | --- |
| :instrument\_token | Identifier for the instrument whose historical records you want to fetch. This is obtained with the[instrument list](https://kite.trade/docs/connect/v1/#retrieving-full-instrument-list) API. |
| :interval | The candle record interval. Possible values are:  · minute · day · 3minute · 5minute · 10minute · 15minute · 30minute · 60minute |

### Response:

The response is an array of records, where each record in turn is an array of the following values — [timestamp, open, high, low, close, volume].

