

**O0500 - SOFTWARE ARCHITECTURE**



| **Version:** | **1.0** |
| --- | --- |
| **Status:** | Plan |
| **Approver:** |  |
| **Author:** |  |



**Document history**

| Version | Date | Author | Status | Remarks |
| --- | --- | --- | --- | --- |
| 1.0 | 24/12/2021 | Nguyễn Bảo Nguyên  Quách Hoàng Minh  Ngô Gia Hân  Nguyễn Vũ Anh Thư | Draft |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

**References**

| Reference | Title | Author | Version |
| --- | --- | --- | --- |
|  |  |  |  |

Table of contents

[**Introduction**](#_heading=h.1fob9te) **3**

[**Definitions and abbreviations**](#_heading=h.lts9tn815a5h) **3**

[**Purpose**](#_heading=h.ot2k188tre1w) **3**

[**Scope**](#_heading=h.2et92p0) **3**

[**Tools**](#_heading=h.3dy6vkm) **3**

[**Software architecture document aim**](#_heading=h.1t3h5sf) **3**

[**Contents of O0500**](#_heading=h.4d34og8) **4**

[**Architectural aims and framework**](#_heading=h.2s8eyo1) **4**

[**Architectural model**](#_heading=h.17dp8vu) **4**

[3.2.1. Component diagram](#_heading=h.dz12hkln1gq) **4**

[3.2.2. Use case view](#_heading=h.3rdcrjn) 5

[3.2.3. Logical data model](#_heading=h.26in1rg) 6

[3.2.4. Interface](#_heading=h.35nkun2) 9

# Introduction

## Definitions and abbreviations

| Designation | Description |
| --- | --- |
| TVP | Trading Vision Project |
|  |  |
|  |  |

## Purpose

The purpose of this document is to describe the software architecture i.e. which components are included in the solution, and how the components that have been developed as part of the Trading Vision Project (TVP) are designed and developed. The document also describes essential architectural principles for e.g. logging, fetching data, etc.

## Scope

The software architecture interacts with the following deliverables and should create an overview of these:

* A0140 - Use Case
* D0130 - Logical Data Model

The software architecture document also provides input for detailed design:

* DD130 - Detailed Design

## 

## Tools

Use of a UML model in LucidChart, where the software architecture can be maintained, is recommended.

# Software architecture document aim

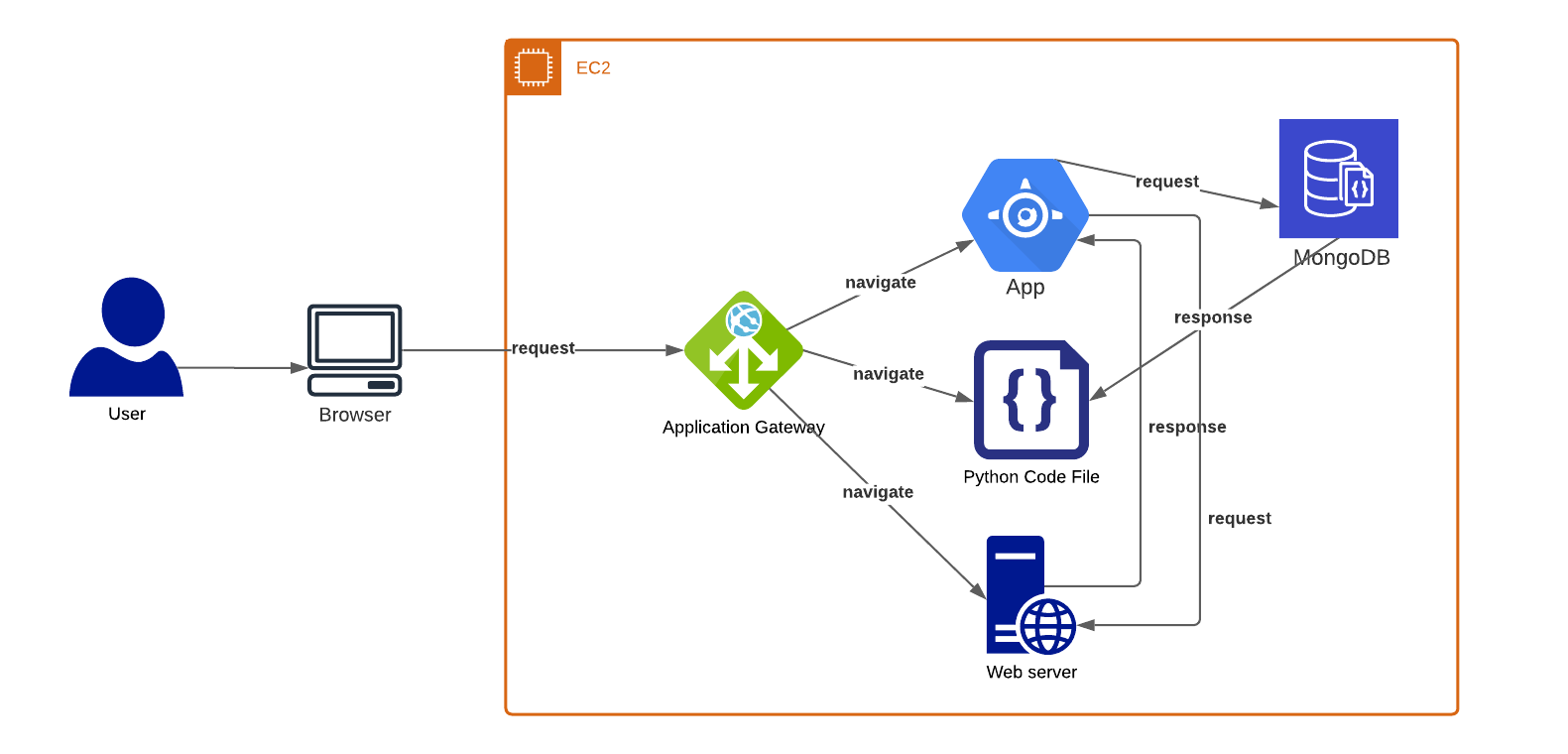
This deliverable is essential to the success of the project. The software architecture is the first document a new project participant should read in order to gain an overview of the system.

The deliverable must be started as soon as possible in the project. Introduction of a software architecture as early as the actual tendering process is recommended.

This section describes the principles used in the development of software architecture. Later sections describe how essential principles are realized in the design of the Data Analysis project.

# Contents of O0500

## Architectural aims and framework



Architect framework for Analysis Project

## Architectural model

### 3.2.1. Component diagram

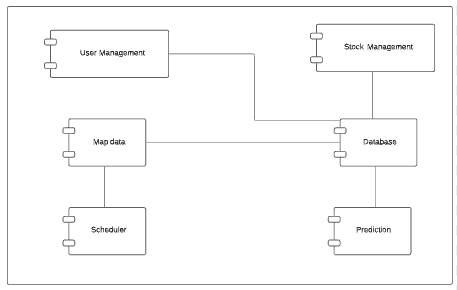
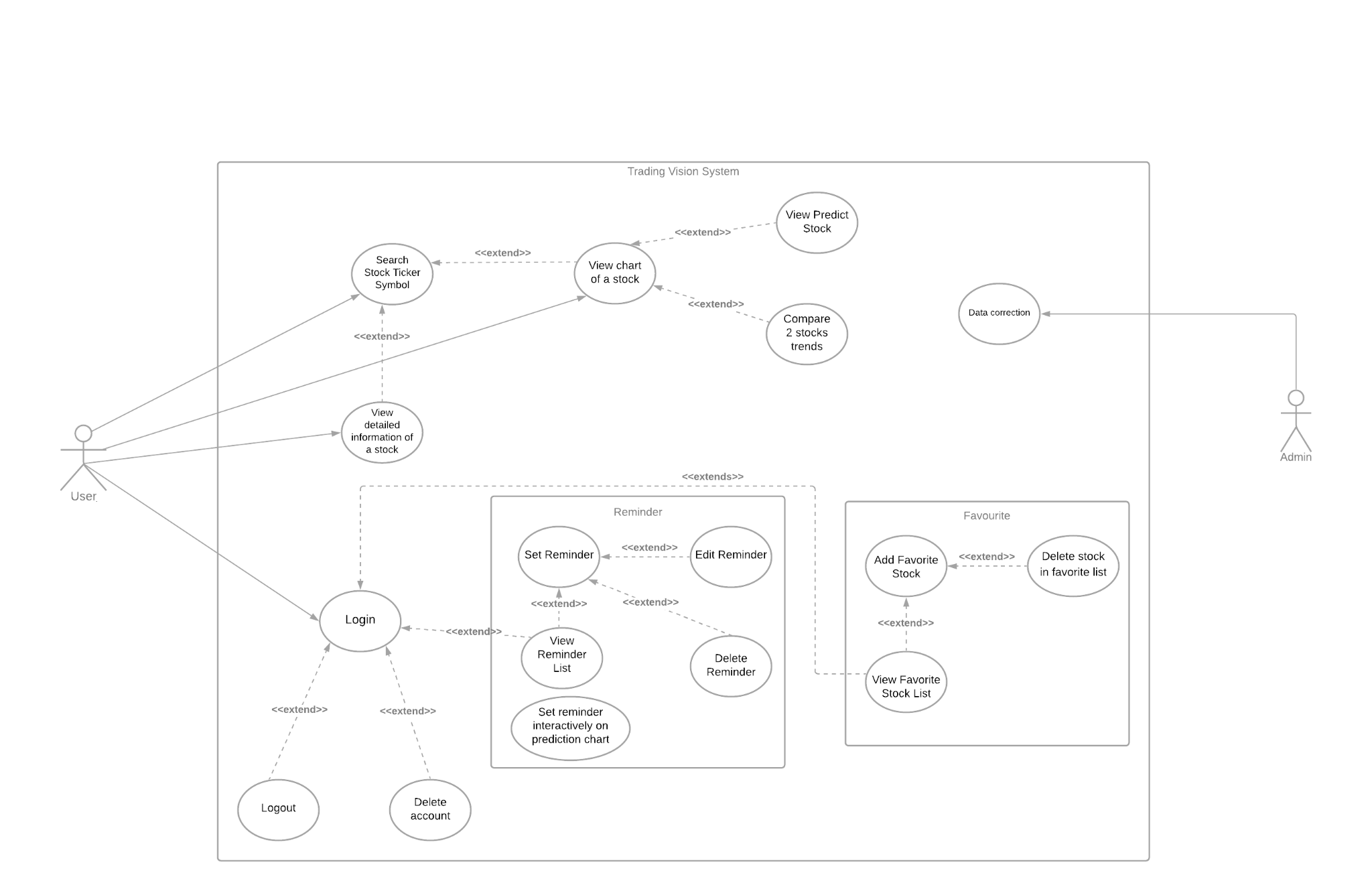


Figure 1: Component Diagram

The component diagram above shows the high-level view of the technical architecture of our system. Five components play their role in the system respectively below:

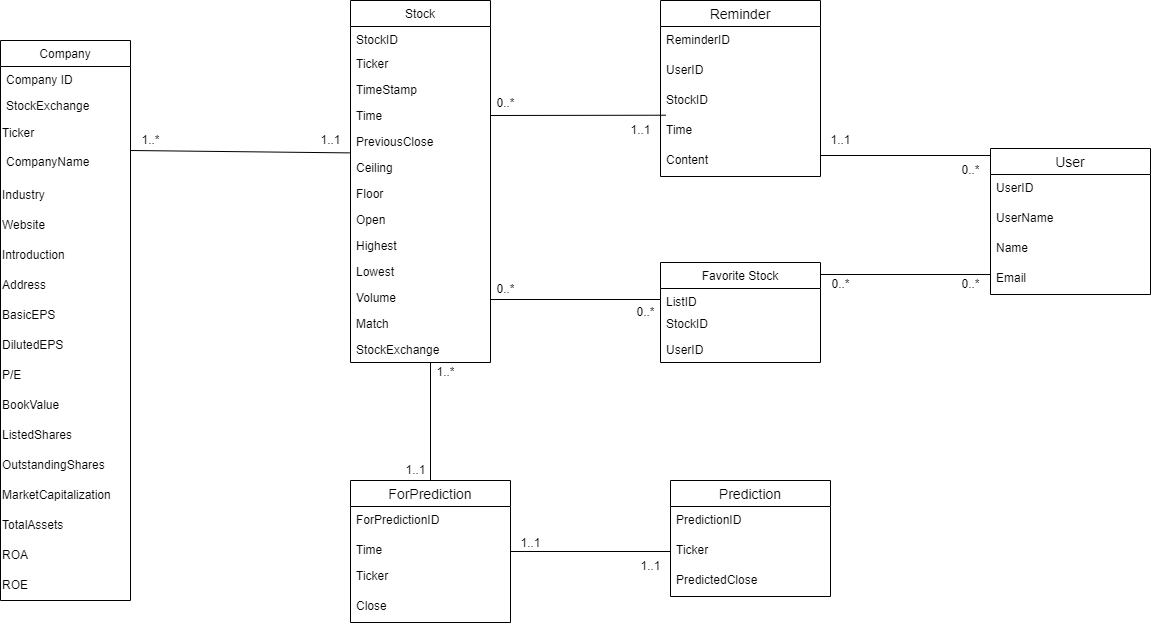
* **Scheduler** fetches data from other third parties (VCBS - Vietcombank Securities, CafeF). The data includes stock information (High, Low, Ceiling, Floor, Match, Volume, Time and Stock Ticker) and company information (Full company name, field, address, website, brief information, EPS, P/E, book value, shares outstanding, market capitalization, total assets, ROA, ROE). Data will be collected every one minute.
* **Map Data** maps data from other sources to system databases appropriately.
* **Database** is the place storing all system data, and response when requested.
* **User Management** manages all users’ necessary information, reminder list, favorite list.
* **Stock Management** manages all stocks and their company’s information, generates charts to display for users.
* **Prediction** predicts stock price for upcoming week based on a selected algorithm. In order to forecast the close price, we will use the algorithm LSTM stated in A0100 for training and predicting.

### 3.2.2. Use case view



Use Case Diagram

### 3.2.3. Logical data model

**

🔷 **Company**

| **Attribute** | **Description** |
| --- | --- |
| CompanyID: Object ID | ID of a company |
| StockExchange: string | Name of stock exchange |
| Ticker: string | Ticker label |
| CompanyName: string | Full name of company |
| Industry: string | The company's field of business |
| Website: string | Website of the company |
| Introduction: string | Short description of the company |
| Address: string | Address of the company |
| BasicEPS: double | Basic earnings per share is a rough measurement of the amount of a company's profit that can be allocated to one share of its common stock |
| DilutedEPS: double | Diluted earnings per share (diluted EPS) calculates a company’s earnings per share if all convertible securities were converted |
| P/E: double | The price-to-earnings ratio (P/E ratio) is the ratio for valuing a company that measures its current share price relative to its [earnings per share](https://www.investopedia.com/terms/e/eps.asp) (EPS) |
| BookValue: double | Book value is equal to the cost of carrying an asset on a company's balance sheet. |
| ListedShares: double | Listed shares (also referred to as quoted shares) include all shares with prices listed on a recognized stock exchange. |
| OutstandingShares: double | Shares outstanding refer to a company's stock currently held by all its shareholders. |
| MarketCapitalization: double | The total value of all a company's shares of stock |
| TotalAssets: double | Total assets are the representation of the worth of everything company after considering all assets and liabilities |
| ROA: double | Return on assets is a metric that indicates a company's profitability in relation to its total assets. |
| ROE: double | Return on equity (ROE) is the measure of a company's net income divided by its shareholders' equity. |

🔷 **Stock**

| **Attribute** | **Description** |
| --- | --- |
| StockID: Object ID | ID of the stock |
| Ticker: string | Name of the stock |
| TimeStamp: int | Time of getting the information (unix timestamp) |
| Time: int | Original time in the source (unix timestamp) |
| PreviousClose: double | Close price of previous day |
| Ceiling: double | Ceiling price |
| Floor: double | Floor price |
| Open: double | Open price |
| Highest: double | Highest price |
| Lowest: double | Lowest price |
| Volume: double | Volume of the stock |
| Match: double | Matching price |
| StockExchange: double | Name of the stock exchange |

🔷 **Reminder**

| **Attribute** | **Description** |
| --- | --- |
| RemiderID: Object ID | ID of reminder |
| UserID | ID of user |
| StockID | Id of Stock |
| Time: Date | The time that reminder will send notify |
| Content: string | The content of the remider |

🔷 **Favorite Stock**

| **Attribute** | **Description** |
| --- | --- |
| ListID: Object ID | the ID of the favorite list |
| StockID | ID of Stock |
| User ID | ID of user |

🔷 **ForPrediction**

| **Attribute** | **Description** |
| --- | --- |
| ForPredictionID: Object ID | ID of the prediction list |
| Time: string | The time of the data save for predict |
| Ticker: string | The predicted ticket ‘s label |
| Close: string | The closed price of the predicted stock |

🔷 **Prediction**

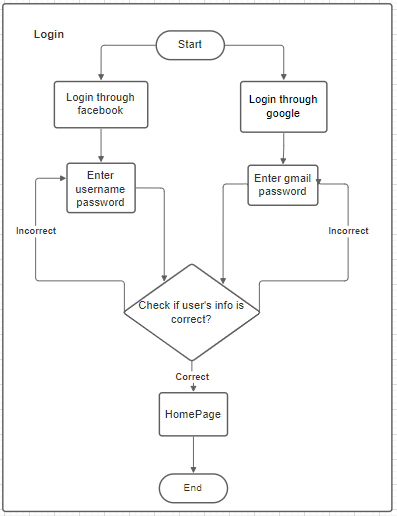
| **Attribute** | **Description** |
| --- | --- |
| PredictionID: Object ID | ID of the prediction list |
| Ticket: string | The label of the stock |
| Closed: string | The stock closed price |

🔷 **User**

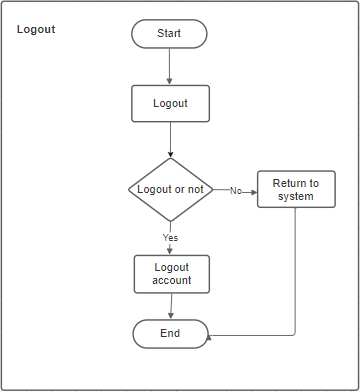
| **Attribute** | **Description** |
| --- | --- |
| UserID: Object ID | ID of user |
| UserName: string | The username that third-parties provide |
| Name: string | Full name of user provided by third-parties |
| Email: string | The user’s email provided by third-parties |

### 3.2.4. Interface

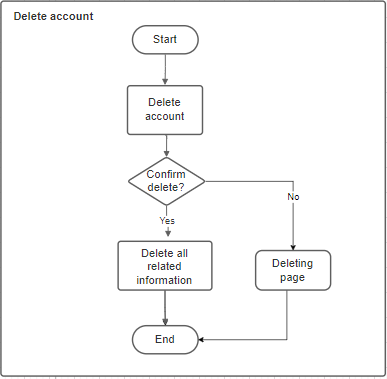
All functions except the Login function, users must login successfully.



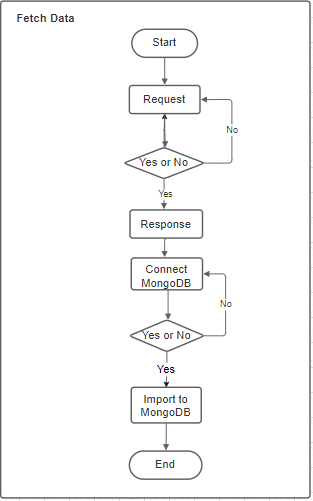
Login function



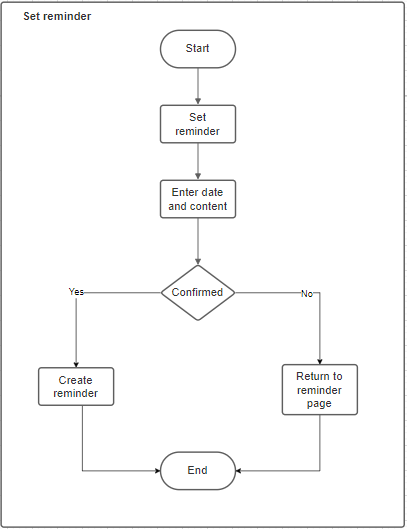
Logout function



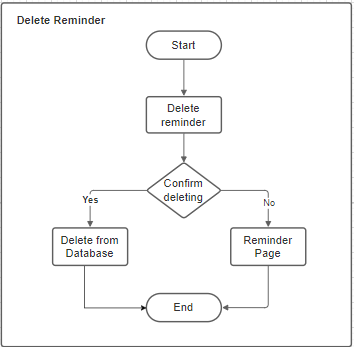
Delete account



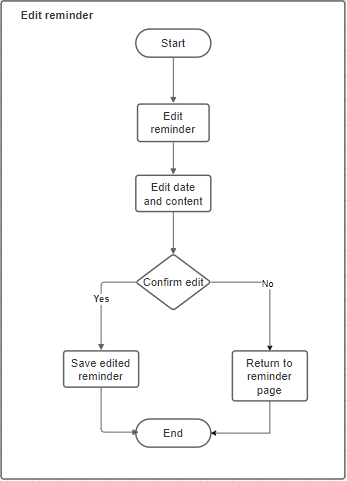
Fetch Data



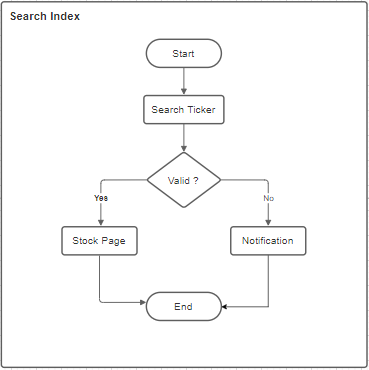
Set Reminder function



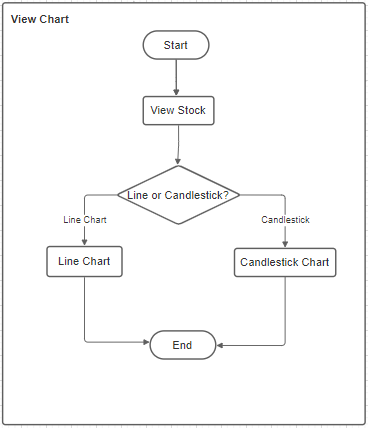
Delete Reminder function



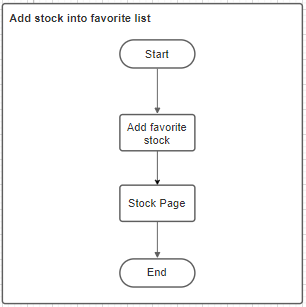
Edit Reminder list



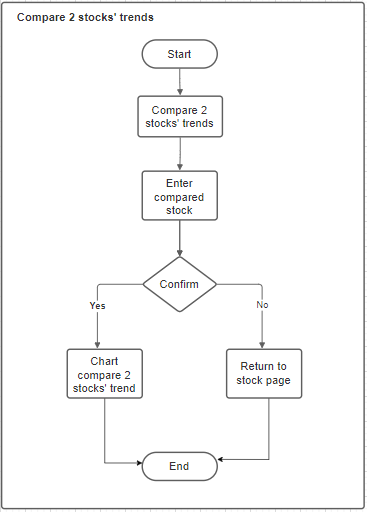
Search Index Function



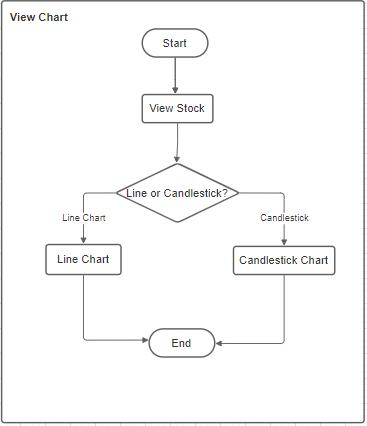
View Chart Function



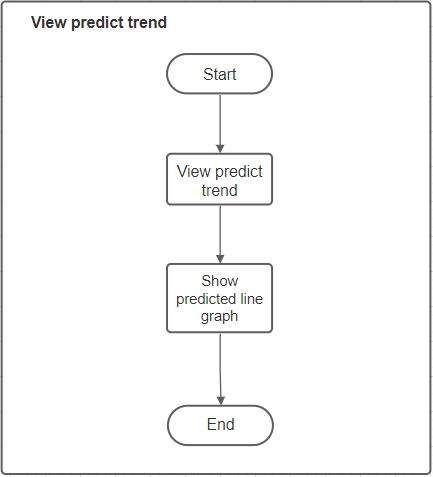
Add stock into favorite list



Compare two stocks’ trends

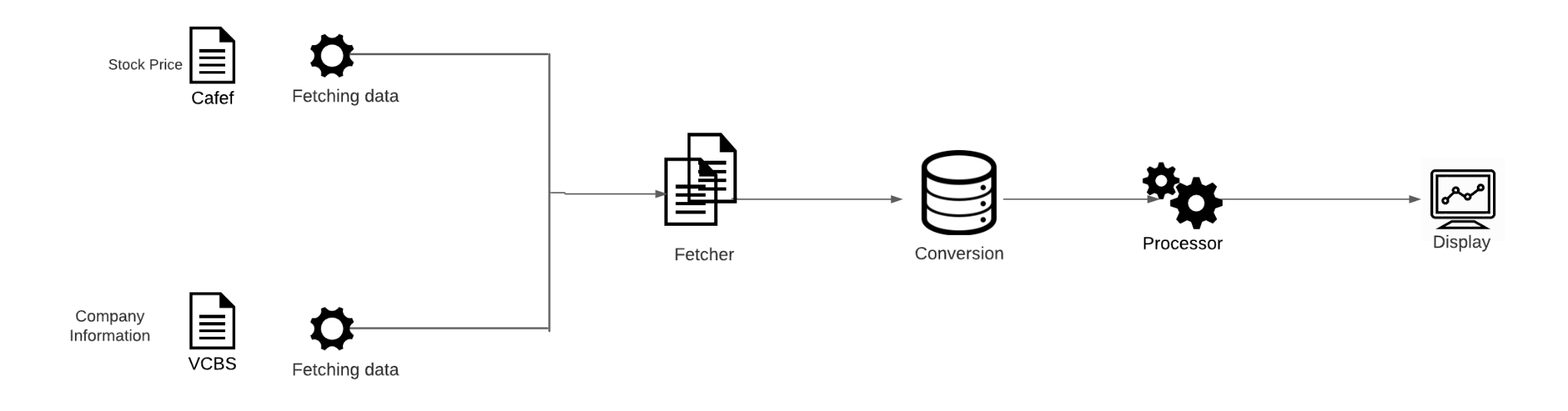


View Chart



View Predict chart

**3.2.5. Data View**



System gets data from two sources: CafeF for stock price and VCBS for company information. The retrieved data will be mapped to the database of system MongoDB in suitable schema. After that, those data will be preprocessed to prepare for the prediction phase afterwards. Finally, on the monitor of the users shows the chart according to the output of the prediction.

🔷 **Stock Price Data**

* With the HNX and Upcom stock exchanges, we used crontab to collect data from Monday to Friday from 8:59 a.m. to 11:30 a.m. and 12:59 p.m. to 15:05 p.m. The HOSE stock exchange, on the other hand, retrieved data from 9:14 AM to 11:30 AM and 12:59 PM to 14:45 PM.
* The columns stored in the database are ‘TimeStamp’, ‘Time’, ‘StockExchange’, ‘Ticker’, ‘PreviousClose’, ‘Ceiling’, ‘Floor’, ‘Highest’, ‘Lowest’, ‘Volume’, and ‘Match’ respectively.

🔷 **Company Information Data**

* The company information is automatically updated at 04:00 PM everyday from Monday to Friday by using crontab.
* The columns stored in the database are ‘StockExchange’, ‘Ticker’, ‘CompanyName’, ‘Address’, ‘Website’, ‘Industry’, ‘Info’, ‘BasicEPS’, ‘DilutedEPS’, ‘P/E’, ‘BookValue’, ‘ListedShares’, ‘OutstandingShares’, ‘MarketCapitalization’, ‘TotalAssets’, ‘ROA’ and ‘ROE’ respectively. If there is no value, fill in the empty string.

🔷 **Data for prediction**

* We wrote a python script to query stock closing prices, and we've set up crontab to run data queries every day at 3:30 p.m.
* The columns stored in the database are ‘Time’, ‘StockExchange’, ‘Ticker’ and ‘Close’ respectively.