CHAPTER 4

SYSTEM ARCHITECTURE

The term system architecture is used to describe the overall design and structure of a computer network or system. As information technology has expanded to include a wide range of physical devices, a method is required to organize and connect these items together in a cohesive manner. The term is also used to describe complex computer software tools that include multiple modules.

There are four main components to any system architecture: processing power, storage, connectivity, and user experience. The complexity of the system varies widely and is dependent upon user needs, business requirements, funding, and resource availability. It is important to note that system architecture must be flexible and able to meet changing needs quickly. A structure that is too rigid will not be able to accommodate new software or hardware.

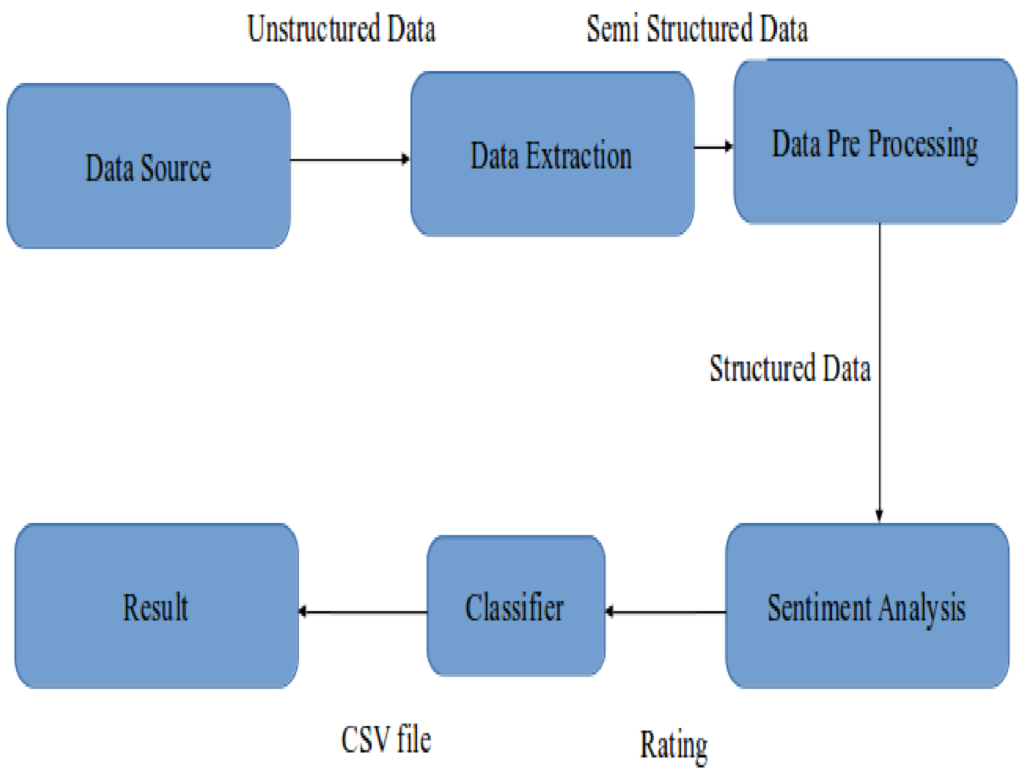

Figure 4.1: Architecture Diagram for proposed system

The prime aim of the proposed system is to fetch live server data by using Python programming language, which can be used for performing sentiment analysis on the extracted datasets from online news portal. In this context, first python is installed on the Ubuntu 14.04 LTS host machine, after that required software such as Selenium is installed using Ubuntu terminal command prompt environment. For Debian or Ubuntu platform Selnium software can be installed with system package manager. Selenium use Python library for fetching live data, and this tool helps to pull contents from desired webpage then save the required information. Selenium supports HTML parser which is included in Python's standard library. For illustration purpose we fetch the live investors reviews that can be used further for sentiment analysis. The steps needed for sentiment analysis using python scripting language are given in the Block digram.

4.1 Modules

4.1.1 Module 1: Web Data Extraction

This module is used to extract the data from web.

- The url of the company is already has to to stored in csv file.
- The program must uses these data to extract the investors reviews from web through selenium tool.

4.1.2 Module2: Data pre-processing

- The extracted web data is to be pre-processed to form semi-structured data.
- The raw data from web has to processed to remove the reviews which may be one day old.
- Tokenization has to be done on the semi-structured data to make tokens of sentences.
- Lemmas of the keyword should be produced on the keywords. These keywords are only means to identify the correct stock from the application.

4.1.3 Module 3:Sentimental Analyzer

- Only those comments should be selected if it is related to stock. These sentences are identified by predefined keyword.
- Sentimental score is calculated for the selected sentences.
- For each predefined keywords sentiment score is calculated.
- Ratings are given to the keywords by how many times they occur at sentences.
- All ratings of keywords are calculated.

4.1.4 Module 4: Classifier

- The classifier will classify these sentiment scores as buy, sell, hold.
- The company has more positive rating then the stock can be brought.
- If company has more negative rating then the stocks of that company can be sold.
- If not hold the stocks.

System Design

Systems design is the process of defining the architecture, modules, interfaces, and data for a system to satisfy specified requirements. Systems design could be seen as the application of systems theory to product development. There is some overlap with the disciplines of systems analysis, systems architecture and systems engineering. If the broader topic of product development "blends the perspective of marketing, design, and manufacturing into a single approach to product development," then design is the act of taking the marketing information and creating the design of the product to be manufactured. Systems design is therefore the process of defining and developing systems to satisfy specified requirements of the user.

Data flow diagram

A data flow diagram (DFD) is a graphical representation of the "flow" of data through in information system, modelling its process aspects. A DFD is often used as a preliminary step to create an overview of the system without going into great detail, which can later be elaborated. DFDs can also be used for the visualization of data processing (structured design).It's easy to understand the flow of data through systems with the right data flow diagram software. This guide provides everything you need to know about data flow diagrams, including definitions, history, and symbols and notations. You'll learn the different levels of a DFD, the difference between a logical and a physical DFD and tips for making a DFD. A data flow diagram can dive into progressively more detail by using levels and layers, zeroing in on a particular piece. DFD levels are numbered 0, 1 or 2, and occasionally go to even Level 3 or beyond. The necessary level of detail depends on the scope of what you are trying to accomplish.

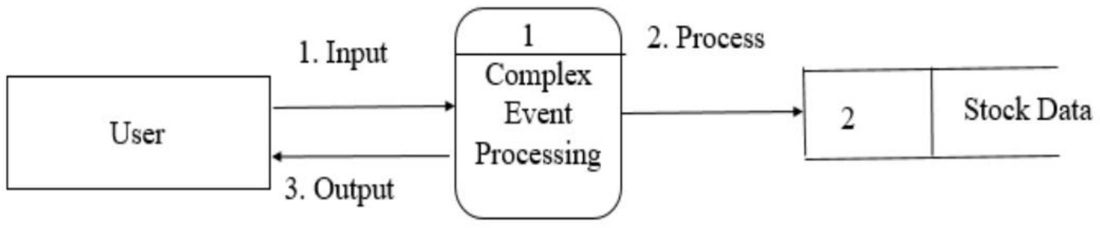

Figure 4.2 DFD Level 0

- The diagram gives us the general idea of the entire system. It also specifies the entities involved. In the above specified diagram level 0, user enters the stock symbol, process the stock data of the corresponding stock symbol, and produces output upon analysis of the stock data. Data is obtained from a .csv file or from internet sources.

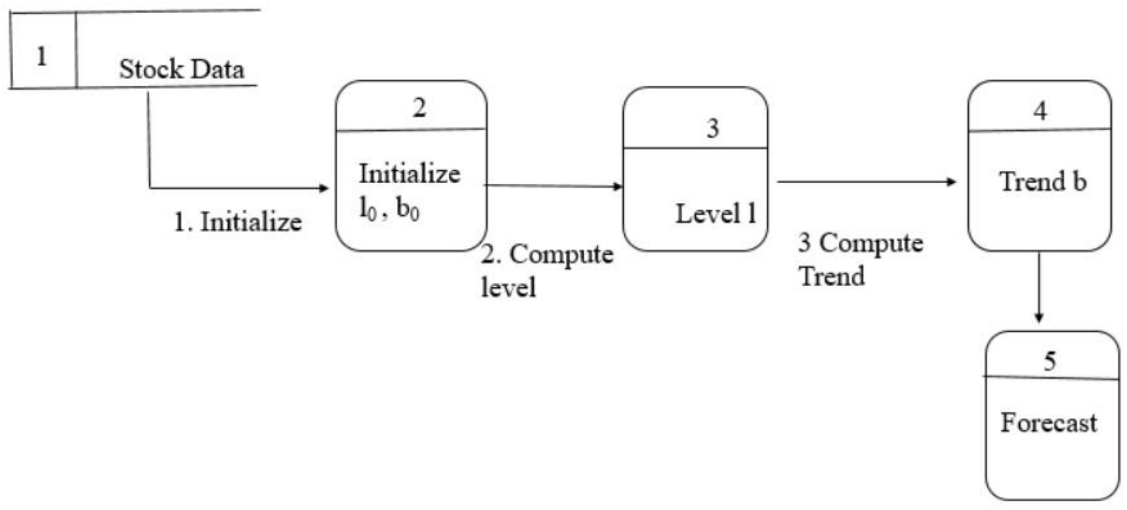

Figure 4.3 DFD Level 1

- The next stage is to create the Level 1 Data Flow Diagram.In the above specified diagram level 1, upon receiving the investors reviews, predicting is done on the stock trends(buy, sell, hold).

Sequence Diagram

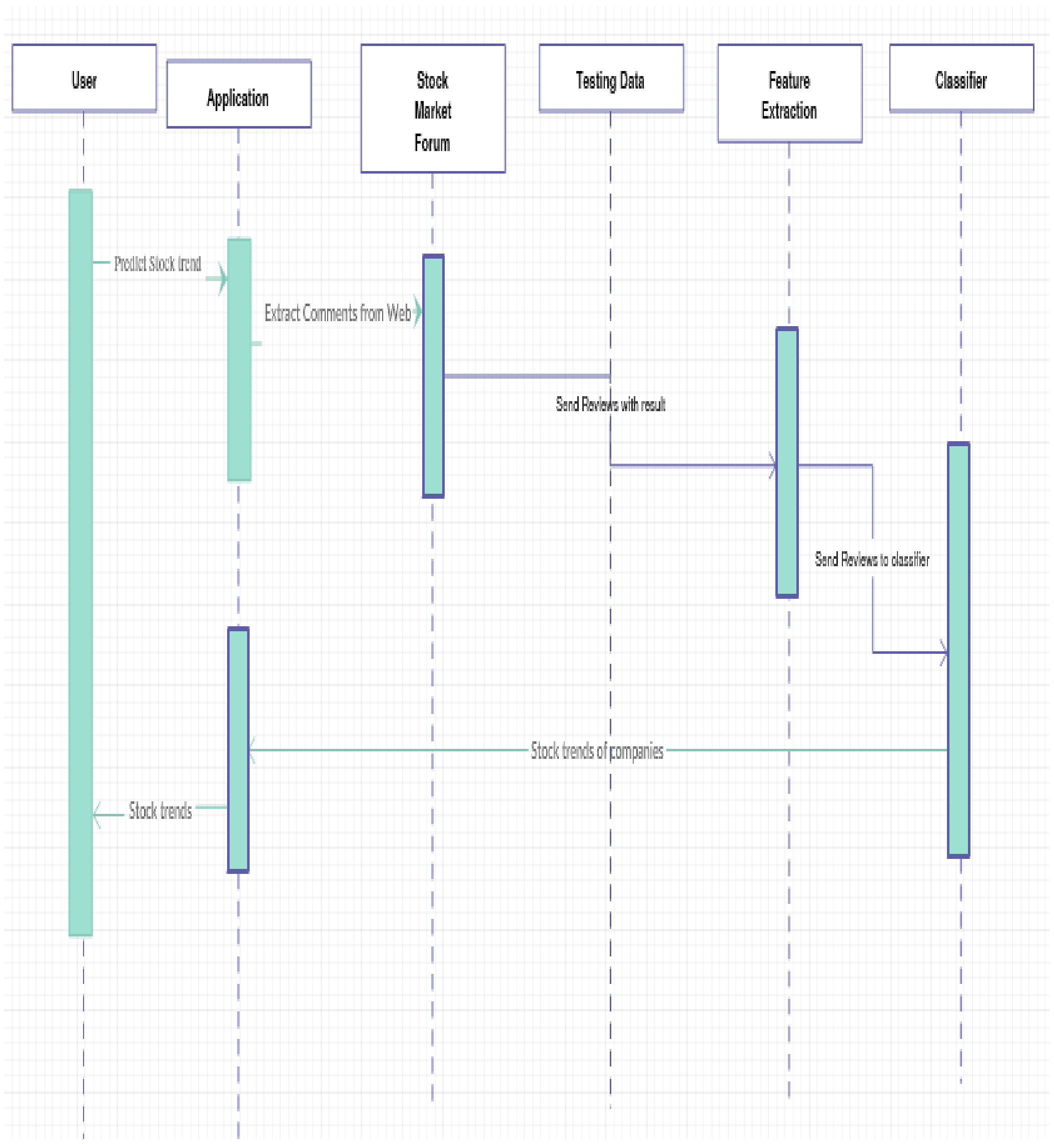

Figure 4.4: Sequence diagram

A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. Sequence diagrams are typically associated with use case realizations in the Logical View of the system under development. Sequence diagrams are sometimes called event diagrams or event scenarios.

Messages, written with horizontal arrows with the message name written above them, display interaction. Solid arrow heads represent synchronous calls, open arrow heads represent asynchronous messages, and dashed lines represent reply messages.[1] If a caller sends a synchronous message, it must wait until the message is done, such as invoking a subroutine. If a caller sends an asynchronous message, it can continue processing and doesn’t have to wait for a response. Asynchronous calls are present in multithreaded applications, event-driven applications and in message-oriented middleware. Activation boxes, or method-call boxes, are opaque rectangles drawn on top of lifelines to represent that processes are being performed in response to the message (ExecutionSpecifications in UML).

Objects calling methods on themselves use messages and add new activation boxes on top of any others to indicate a further level of processing. If an object is destroyed (removed from memory), an X is drawn on bottom of the lifeline, and the dashed line ceases to be drawn below it. It should be the result of a message, either from the object itself, or another.

Sequence diagrams are a popular dynamic modeling solution in UML because they specifically focus on *lifelines*, or the processes and objects that live simultaneously, and the messages exchanged between them to perform a function before the lifeline ends. Along with our UML diagramming tool, use this guide to learn everything there is to know about sequence diagrams in UML.

A sequence diagram is a type of interaction diagram because it describes how—and in what order—a group of objects works together. These diagrams are used by software developers and business professionals to understand requirements for a new system or to document an existing process. Sequence diagrams are sometimes known as event diagrams or event scenarios.