Software Design Specification

(SDS)

CommunistBadger

Version 1.0

Prepared by

Muhammad Shahzaib Waseem, Raja Hasnain Anwar, Muahmmad Kamran Janjua, and Muhammad Shafay Ijaz

Table of Contents

1. Introduction

1.1. System Purpose

This document is the next step work after the completion of Software Requirements Specification (SRS). The purpose of this document is to describe the detail architecture and design specifications for the project entitled "CommunistBadger: Stock Analysis Tool".

1.2. System Scope

The scope of this tool is to provide stock analysis for the market. This will allow users to be able to anticipate market changes based on news, trends, and market index.

1.3. Definitions, Acronyms, and Abbreviations

Terms	Full forms
PR	Public Relations

2. System Overview

The final product is implemented as a complete working system with a deep neural network based model at the back. Most of the data will be manipulated using this model. Company details are stored in database. There is also a crawler which scraps data off news websites and Twitter to update model and analysis results.

3. Analysis Model

3.1. Interface Objects

The interface objects represent the main interfaces of the system. Given below is the list of different interface objects used in our system.

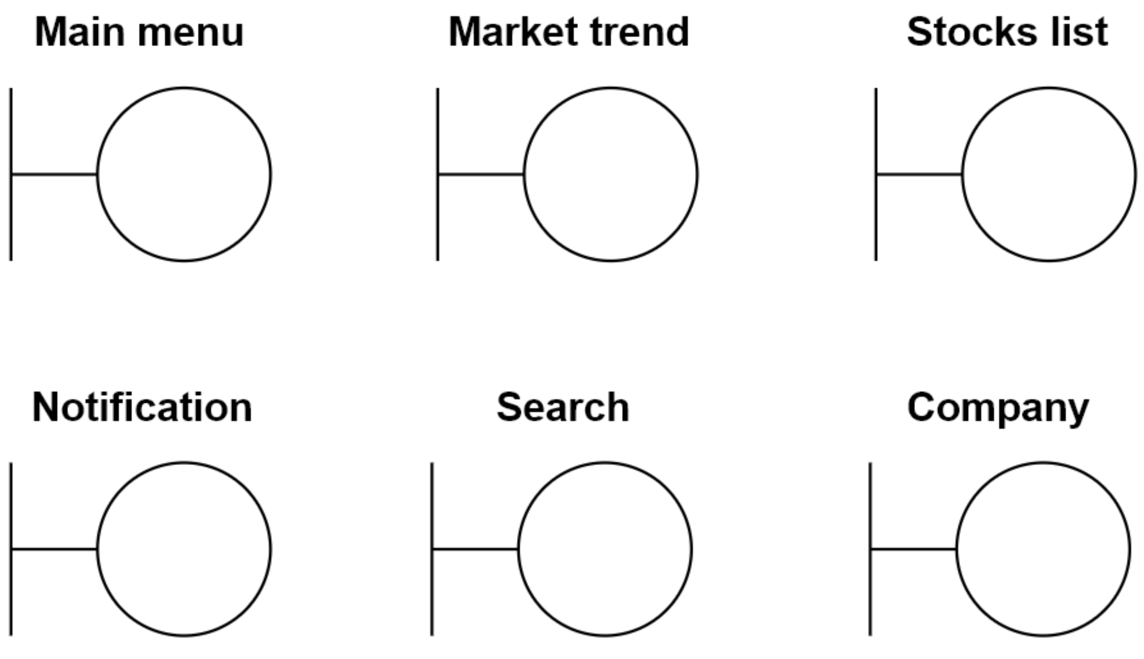


Figure 1: Interface Objects

3.1.1. Acquaintance Association Interface Objects

As all interfaces are unique, there is no need for associations to be mentioned here.

3.1.2. Containership

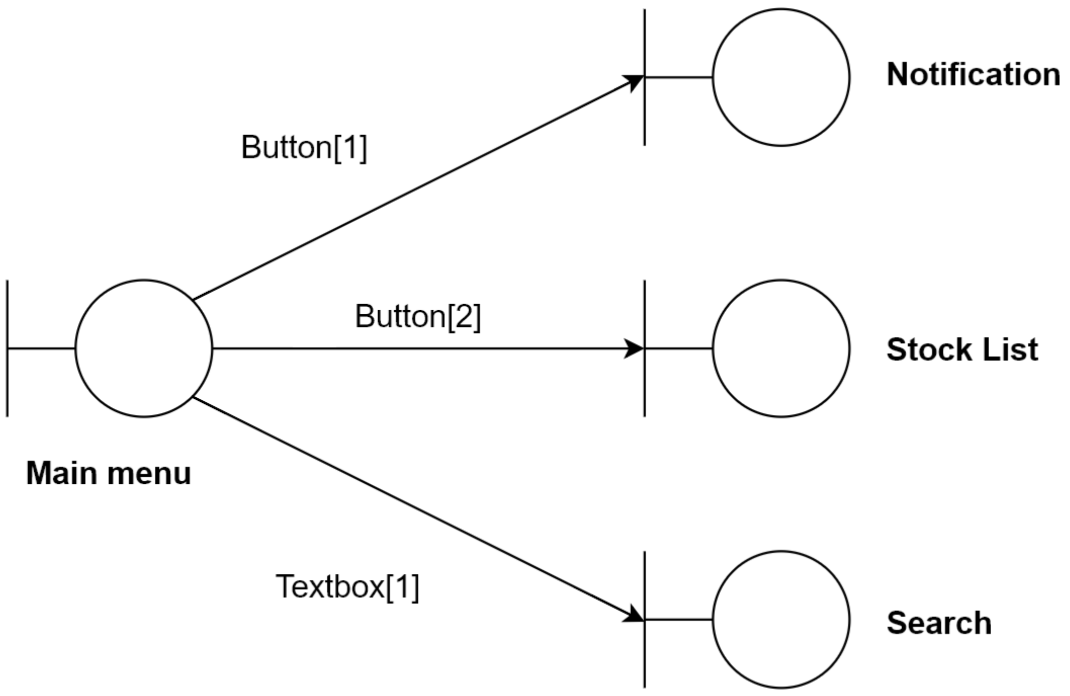


Figure 2: Initial Interface

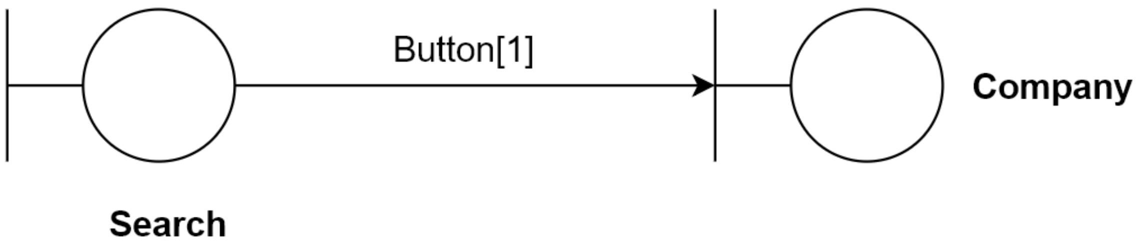


Figure 3: Search "View Graph" Interfacef

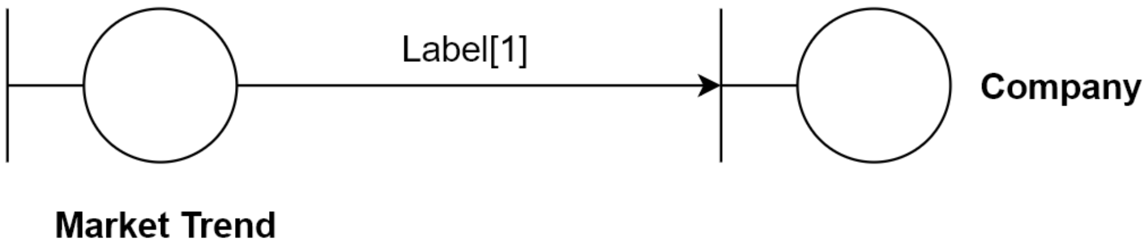


Figure 4: Market "Company Name" Interface

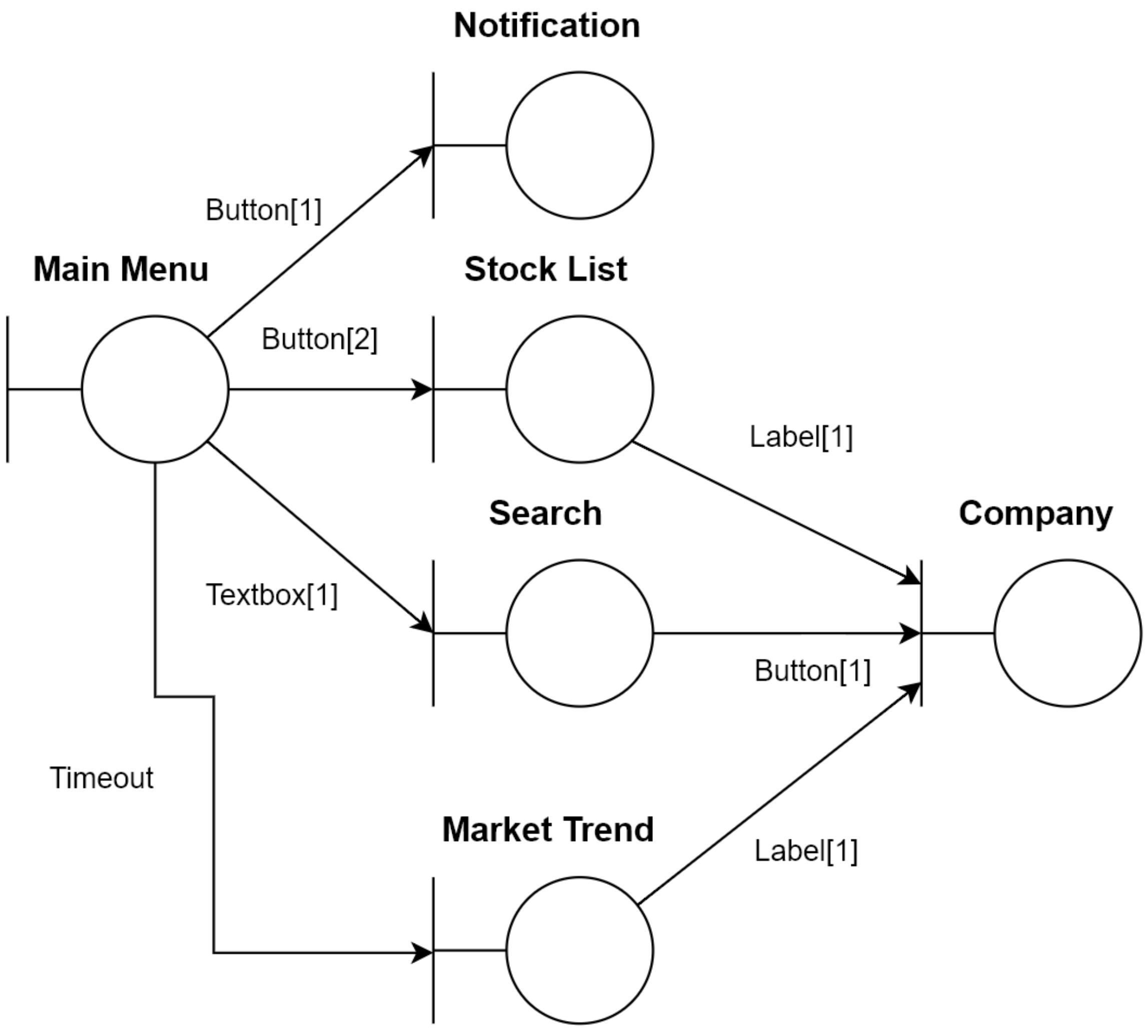
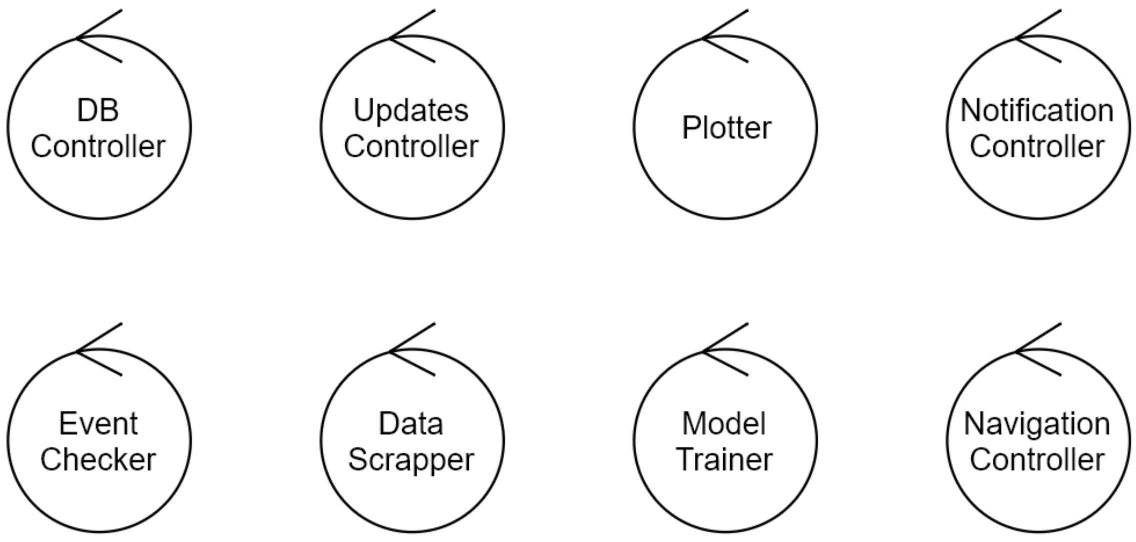


Figure 5: Interface Levels

3.2. Control Objects



3.2.1. Description of Control Objects

DB Controller: Manages database tables, permissions, etc.

Updates Controller: Controls the updates in Database data and retrieval of data to show on user interfaces after updates.

Plotter: Controls the plotting of graphs and other interactive interfaces for user to get more insight of stock analysis.

Notification Controller: Controls display of notifications and messages on various actions to the user.

Event Checker: A controller to check for an event that may have significant impact on market.

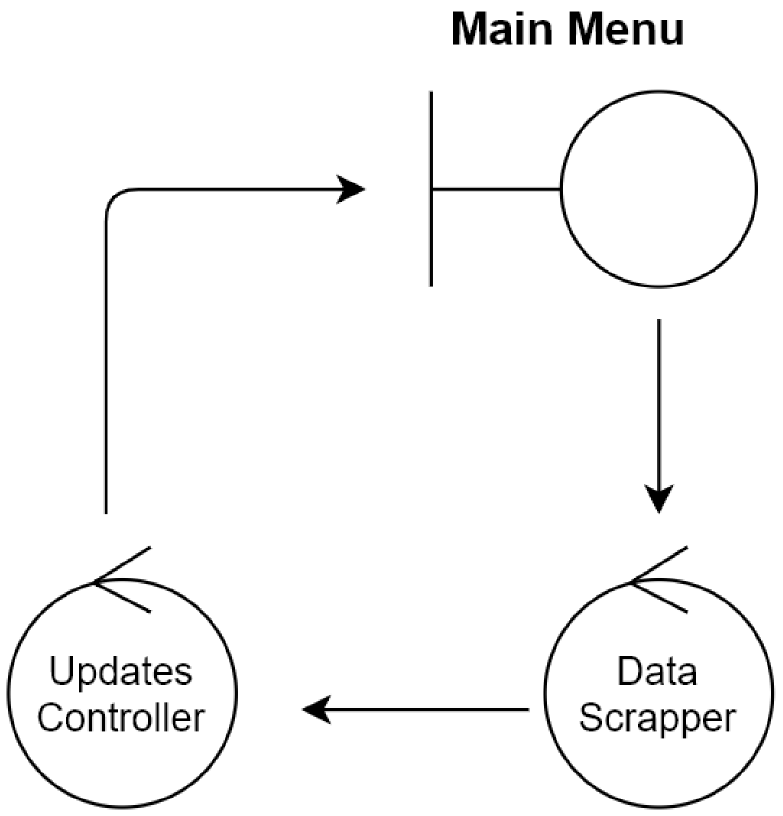
Data Scrapper: Scraps the data off news websites and Twitter to calculate market index.

Model Trainer: Controls updates in neural network model to make more accurate predictions on newly available data.

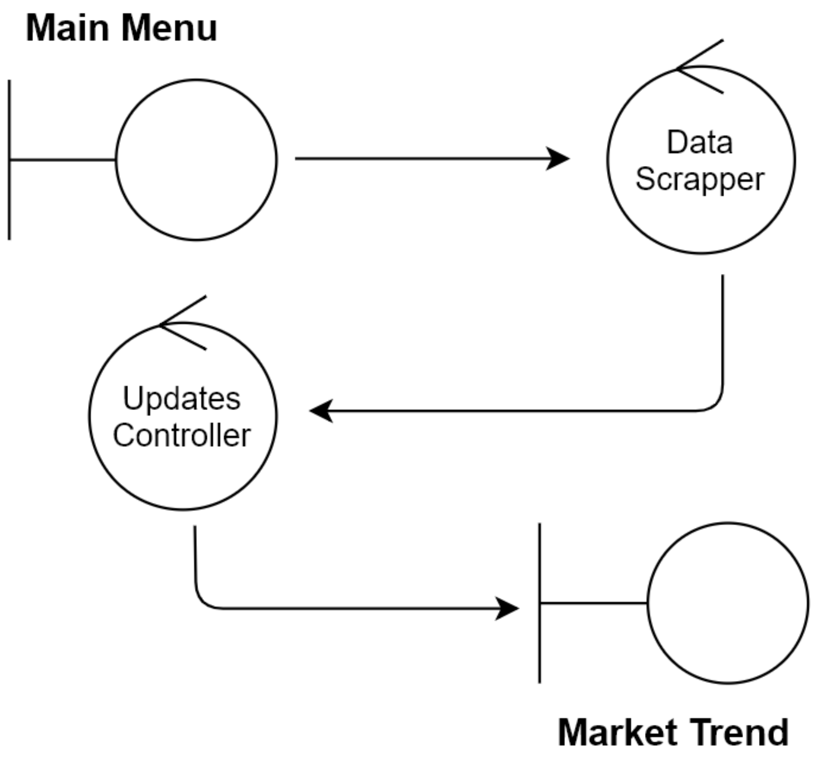
Navigation Controller: Listens to user actions on interfaces and connects relevant sub system to handle the event.

3.3. Subsystem Diagrams

3.3.1. Web Scrapper



3.3.2. Model Training



4. Design Model

4.1. Use Case Diagram

There is only one user type for our system. So, all the operations can be performed by the user. There is no division; so, there is no need for a use case diagram.

4.2. Sequence Diagram

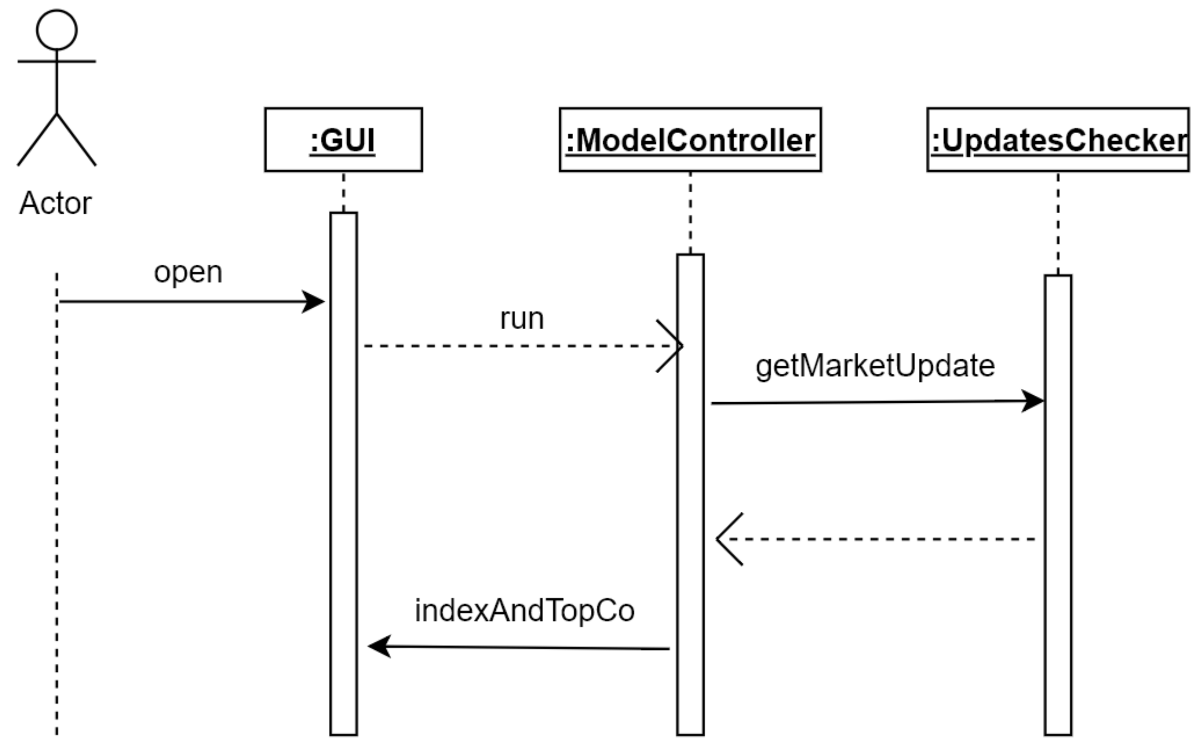


Figure 6: On application launch

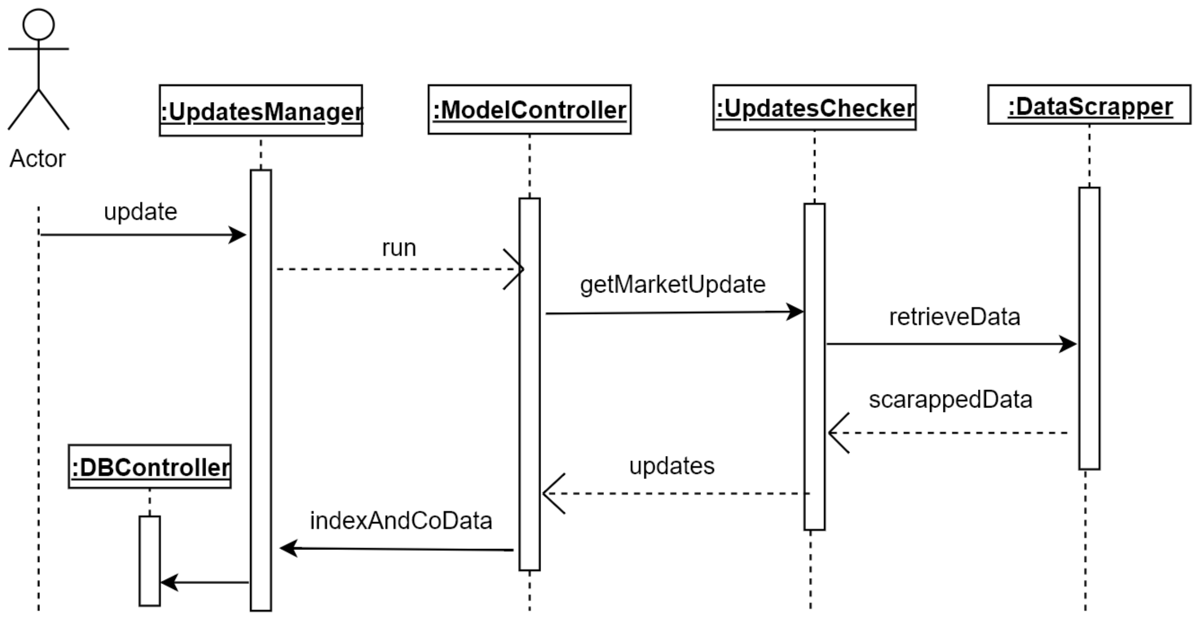


Figure 7: Update Button

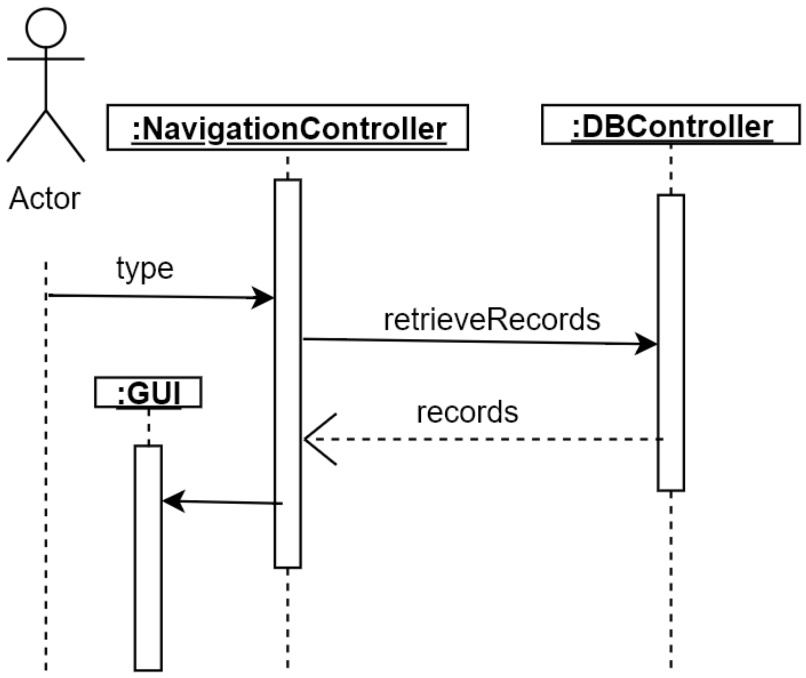


Figure 8: Search

4.3. Communication Diagram

All the communication is depicted in section 4.2. The inter-modular communication will be carried in same fashion as we do not have many classes to show.

4.4. Activity Diagram

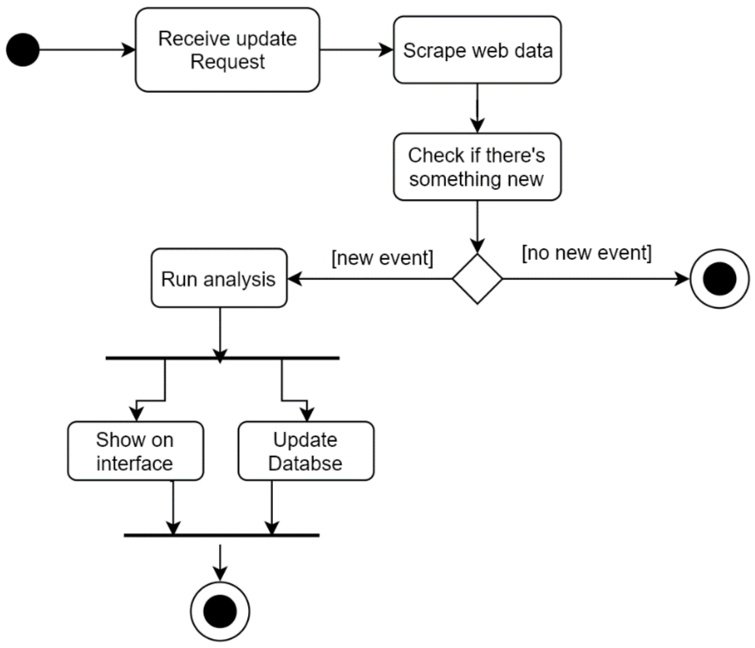


Figure 9: Update Activity

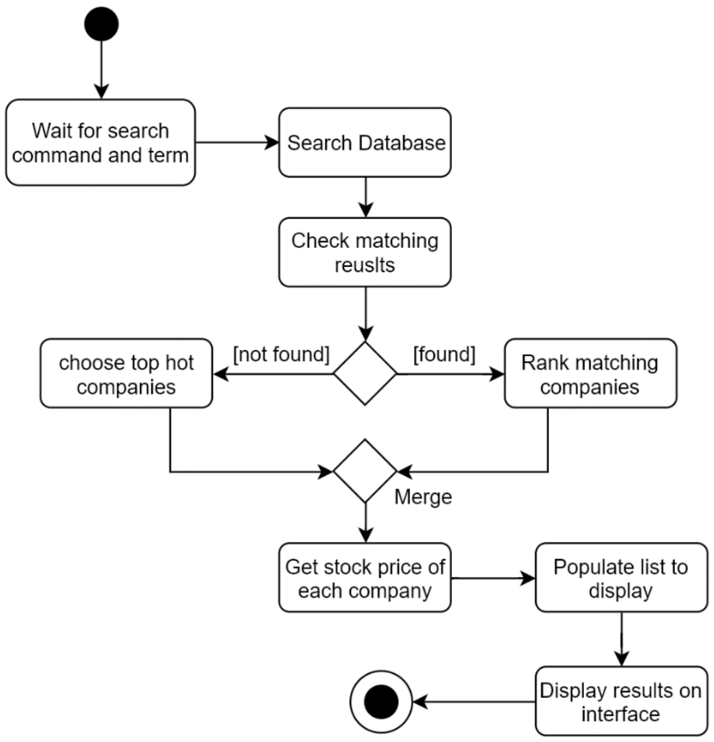


Figure 10: Searching Activity

4.5. Deployment Diagram

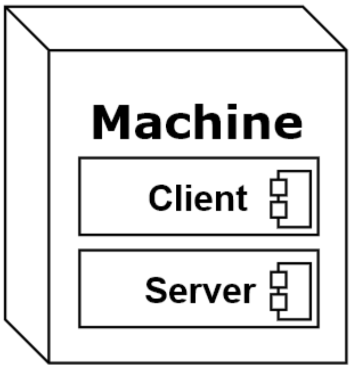
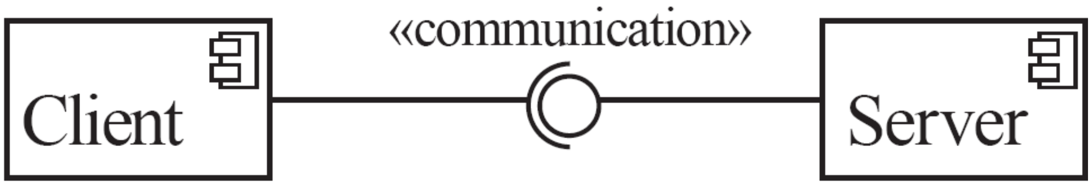


Figure 11: Client-Server deployment

The system is based on and implemented as a client server architecture. But due to limited scope, it is implemented on a single machine for easy management and communication. This reduces are overhead of network communication and the problems that come with it.

4.6. Component Diagram



4.7. Package Diagram

The project is coded using Python which is not strictly object-oriented and does not have any concept of packages. Its modules are basically files with .py extension. All main components described under section 3 are in separate files ensuring lessen coupling.