[Interactive Trader] User Guide

|  |  |  |
| --- | --- | --- |
| Date | Version Number | Document Changes |
| 4/7/2020 | 1.0 | Initial Draft |

Table of Contents

1 Introduction 3

1.1 Scope and Purpose 3

1.2 Process/Workflows 3

2 Technologies 6

3 Install Steps 7

4 Index 9

4.1 Style Sheet Information 10

5 Heading 1 10

5.1 Heading 2 10

5.1.1 Heading 3 10

# Introduction

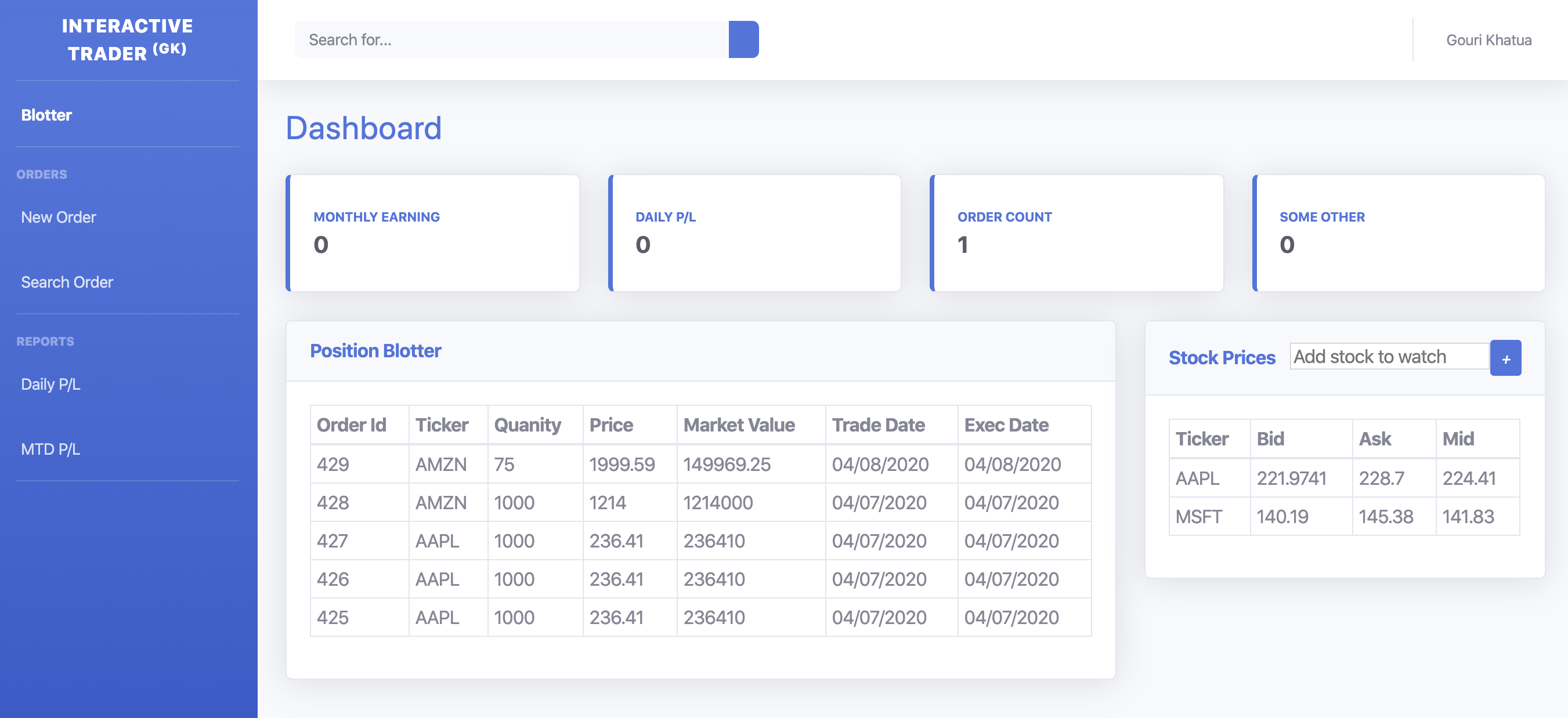
## Scope and Purpose

This Document provides an overview of the application designed and installation steps required to view it.

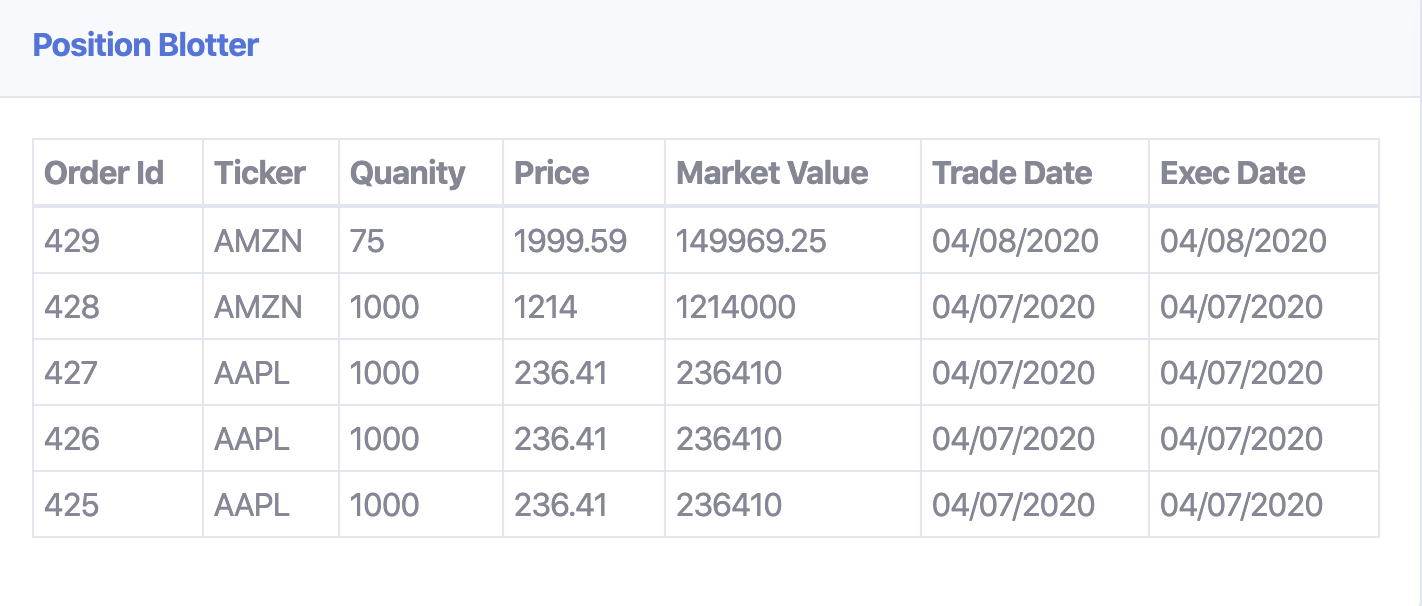
## Process/Workflows

The Application as requested is Named as “Interactive Trader”. It’s a web based application that provides following workflows.

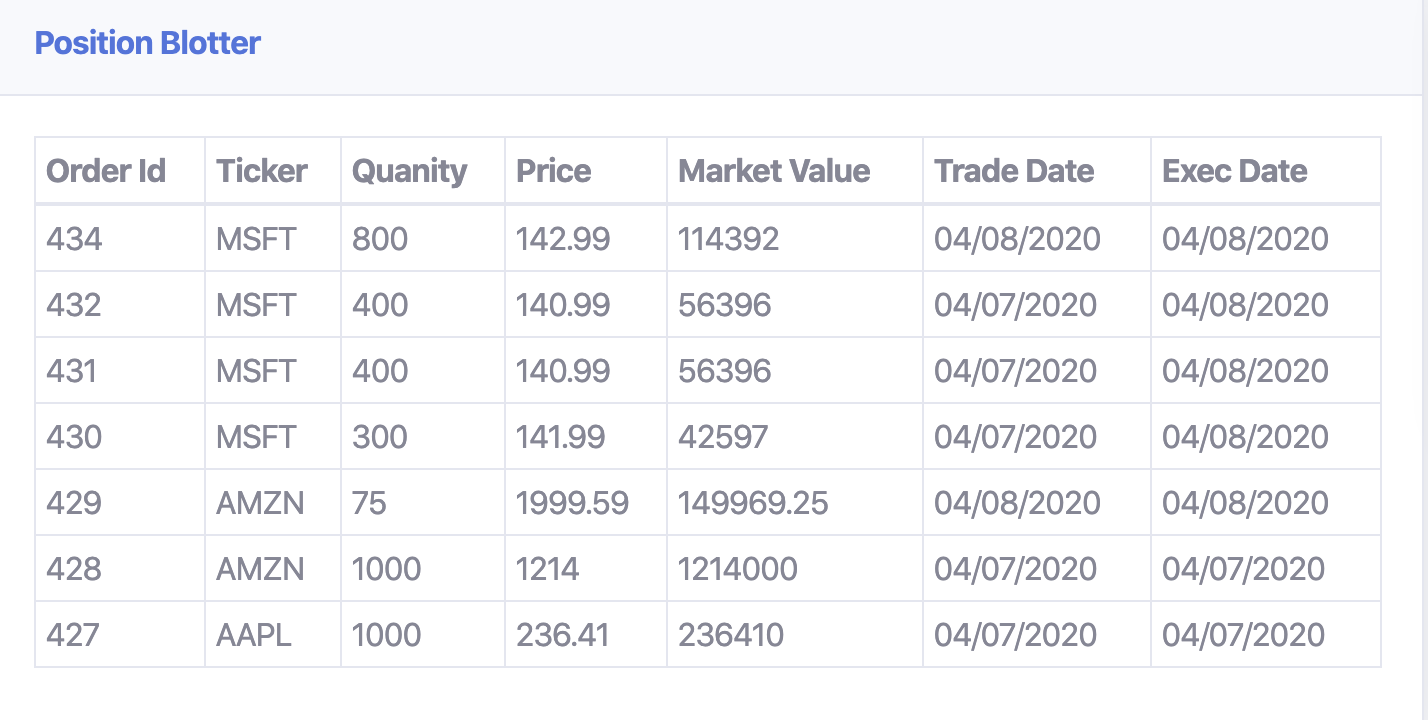
1. A Dashboard summary of Trader’s Trade Count, PL and Trade Details.



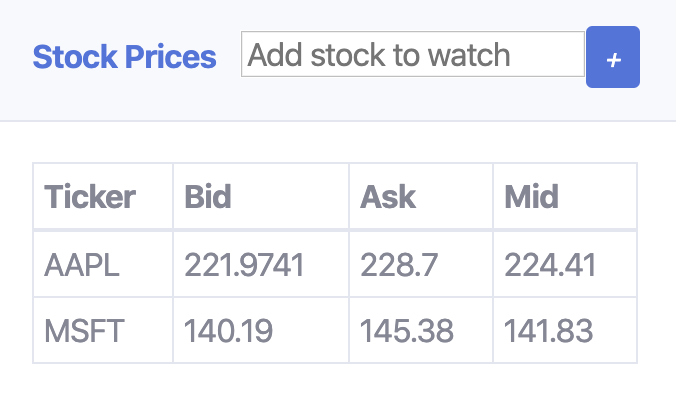
1. A Trading blotter which shows the orders (Last 5 by default).



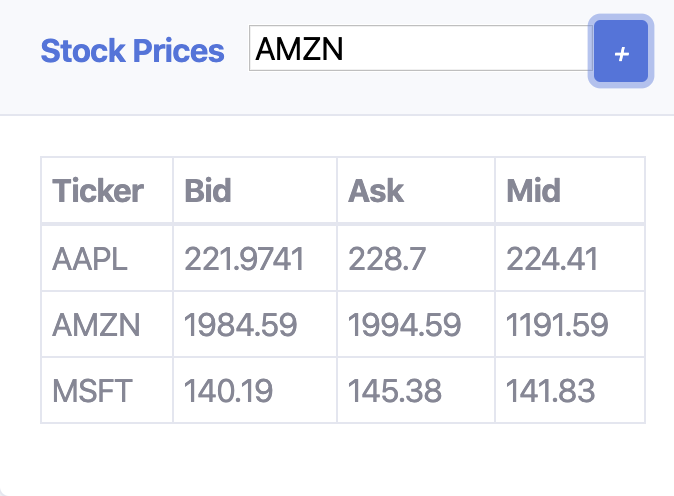
1. The blotter does auto refresh as the trades are created.(e.g. if trades are booked by brokers. Broker screen is not completed yet)



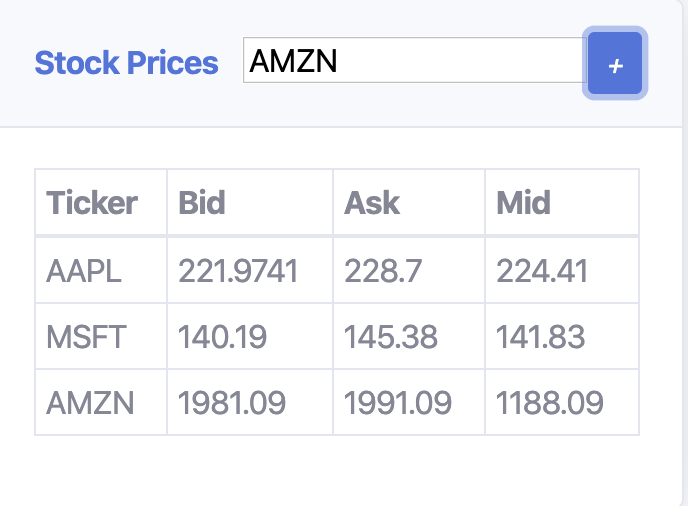
1. An Instrument blotter that shows the pricing information of the tickers the trader is interested.

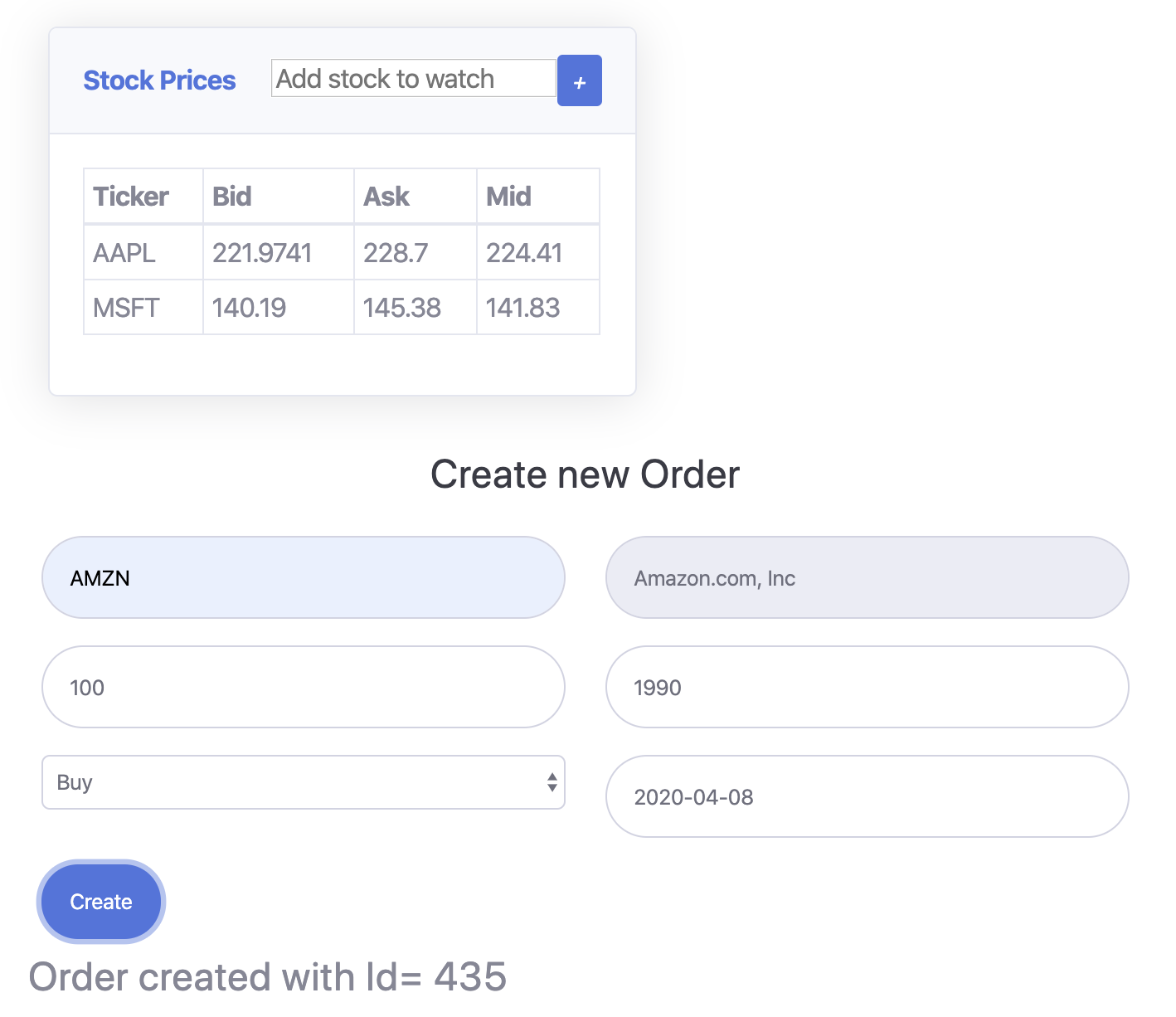


1. The Instrument Blotter allows adding new tickers to monitor price change.



1. The instrument blotter auto refresh when prices are published from backend. (AMZN changed below)



1. A basic order create screen to input new order and it does show tickers with current prices. 

# Technologies

Technologies Used are as Below.

|  |  |  |
| --- | --- | --- |
| Type | Name | Notes |
| Server Side | Spring Boot | Spring REST, spring web socket, |
| Language | Java | 11 |
| Persistence | Spring Data JPA |  |
| Database | H2 | A simple db file has been created and uploaded to repository. Test data already created |
| Front End | ReactJS | React JS/Node JS based web UI. |
| Build Tool | Gradle | Java Project |
| Build Tool | Yarn | Front-end |

# Install Steps

The project has been uploaded to git-hub open repository.

[interactive-trader-service]

URL: <https://github.com/gourisk/interactive-trader-service>

Clone URL: <https://github.com/gourisk/interactive-trader-service.git>

1. Go to interactive trader service

cd interactive-trader-service

1. Copy test data file to home dir

cp ./data/tradingdb.mv.db ~/

If gradle is already installed

1. To Directly run dev build

gradle --info bootRun

1. To build a production build.

gradle bootJar

1. Run the Jar file created.

java -jar ./build/libs/interactive-trader-service-0.0.1-SNAPSHOT.jar

If gradle is not installed

1. To Directly run dev build

./gradlew --info bootRun

1. To build a production build.

./gradlew bootJar

1. Run the Jar file created.

java -jar ./build/libs/interactive-trader-service-0.0.1-SNAPSHOT.jar

[interactive-trader-ui]

URL: <https://github.com/gourisk/interactive-trader-ui>

Clone URL: <https://github.com/gourisk/interactive-trader-ui.git>

1. Go to interactive trader ui

cd interactive-trader-ui

Install Yarn (go to step 4 if already done) [Assume NodeJs is already installed)

1. Install Yarn

npm install -g yarn

1. Install Server.

Npm install -g serve

Build the Project

1. Init the project

yarn install

1. To run a dev build.

Yarn start

1. To build a production build.

Yarn build

1. Run the production build

Serve -s build

Thank you