
# Installation Guide

---

**Team Members:**
Adam Burich, Bright Hsu, Kayla Savage,
Janice Tran, William Puppo, Sandalu Widyalankara

May 31st, 2022
**Version 1.0.0**

**Table of Contents**

# 1 Introduction

## *1.1 Purpose*

The purpose of this document is to specify the software requirements and steps for installation for DAITrader. It is to clarify and explain the requirements needed for the application to function for both programmers and users.

DAITrader provides insight about stocks, particularly, the companies that own them, their trends, and examinations of their projections. This application will feature a GUI, in which the user will be able to interact with and see various simulations.

> **WARNING:** An inadequate, incomplete, or inexistent install procedure may lead to the refusal of the delivery.

## *1.2 Scope*

The application DAITrader offers users insight into the simulation of stocks. Furthermore, it allows for one to create their own account and keep track of stock projections provided by the system.

The main purpose is to feature scenarios of stocks and sectors to the user to provide an experience of various stock projections. This is done with a GUI, database, evaluation system and backend API.

## *1.3 Design Overview*

DAITrader is an investment scenario simulation of the stock market. Using machine learning/AI, DAITrader generates predictions of various companies' value. DAITrader will also aim to offer users a finance related educational perspective; it is our hope that users will be able to use DAITrader to learn more about the stock market.

## 2 Installation Manual

The main components that should be completely and correctly described are the following:

- Prerequisites
- Install procedure
- Tests

### *2.1 Prerequisites*

In this section is a list of install prerequisites that must be fulfilled before the install can begin.

Prerequisites are of the order of

- Operating System
- System Requirements – Docker
- Docker Image - Software Components used by the application (e.g., Database, Application Software, and Configuration Files)

### *2.2 Pre-installation Tasks*

Before installation of DAITrader, please ensure that Docker is installed. Docker is an open-source software designed to facilitate and simplify application development. Ensure that DAITrader's Docker image is downloaded. This file contains immutable source code, dependencies, database, and libraries necessary for DAITrader to run successfully.

**NOTE:** DAITrader Docker image cannot be installed on Windows.

### *2.3 Installation Procedure*

Once all prerequisite checks and controls are positively finished, the install process can start.

#### 2.3.1 Systems

The delivery may include tasks to be operated by the systems team (UNIX, MAC OS).
Most systems installation tasks include, but are not limited to, the following:

- file system setup;
- disk space, memory upgrade;
- installation done by an administrator;

### *2.4 Tasks Description*

- Step: ordinal numbering of the steps, in chronological order
- User: integrator, actor who is executing the installation, or executing the command.
- Actions to execute: unambiguous command the integrator has to execute.

| Step | User | Action to execute |
|---|---|---|
| 1. | User | Open Docker and accept terms of service |
| 2. | User | Download DAITrader Docker Image (file name: daitrader.tar) using this link<br>Link: https://drive.google.com/file/d/1O6qUXlzXQqHY0t70EXltz8EZOL0mW_Px/view?usp=sharing |
| 3. | User | Move File “daitrader.tar” to intended location |
| 4. | User | Open Terminal and change the terminal directory to where you moved daitrader.tar to and run command “docker image load -i daitrader.tar” |
| 5. | User | Run command “docker run -p 8080:8080 -it daitrader” |
| 6. | User | Run command “service postgresql start” |
| 7. | User | Run command “cd ~/Daitrader” |
| 8. | User | Run command “python3.10 server.py: |
| 9. | User | Copy and Paste the first link provided in the Docker container in address bar in browser. |
| 10. | User | Generate Stock and Sector Simulations |

### *2.5 Technical Tests*

System testing is conducted on the Graphical User Interface or GUI, API and Database. Because this document is for the user, below is testing conducted for the only user facing component, the GUI. More information on testing of the API and Database can be found in the *System Test Plan.*

The testing done for the UI component of the system is primarily aimed at making sure the user can not cause the system to send queries to the API that are not valid. Testing should also result in system behavior that tells the user they're doing something wrong.

Zero, One, Many:

- All possible fields tested with zero input, minimum input, maximum or over maximum input
- Test for all empty fields

Checking for Valid Input:

- Testing for the UI component has to ensure that in every place a user can place input the system correctly handles bad input.
- For this reason there are multiple tests run on each input field trying to stress the boundaries of what's allowed in the field
- Testing Success of Action Buttons:
- At any point during use of the application the user may attempt to use the exit button; we need to ensure this actually closes the application at every stage.
- The simulation display that is outputted after generating a simulation includes a save button, it's important to test that this allows a user to save a simulation correctly.

Resizing Application:

- One of the key visual aspects that are important to the GUI are the ability to be able to resize the application while the elements continue to simultaneously display properly. This is tested by analyzing the element sizes as the user readjusts the size of the application for their window.
- Furthermore, resizing must not hinder the deployment of the system. It must not break the application and must continue to run smoothly for the user.

Application Interaction:

- Testing the GUI also requires that each element that was created, such as buttons, scales, and input boxes, must be placed properly and must not glitch.
- The elements should be stagnant with the exception of actions that occur when the user hovers on the button, manipulates the scale, or inputs a value for the simulation.

| **Test Case: UI-1 (Priority: 1)** | |
|---|---|
| **ID** | UI-1 |
| **Functionality** | Test<br>simulation generation for valid single stock |
| **Description** | Given valid<br>parameters in all slots this test establishes that for a valid single stock a simulation is correctly generated. |
| **Pre-condition** | User has completed input form, all parameters including test param are valid |
| **Action(s)** | 1. Input valid single stock<br>2. Input valid historical context<br>3. Input valid dates for simulation<br>4. Generate simulation |
| **Post-conditions** | Valid simulation is generated and displayed |
| | |
| **Test Case: UI-2 (Priority: 1)** | |
| **ID** | UI-2 |
| **Functionality** | Test simulation generation for valid list of stocks |

| | |
|---|---|
| **Description** | Given valid parameters in all slots this test establishes that for a valid multi stock input the correct multi simulation output is generated. |
| **Pre-condition** | User has completed input form, all parameters including test param are valid |
| **Action(s)** | 1. Input valid list of stocks<br>2. Input valid historical context<br>3. Input valid dates for simulation<br>4. Generate simulation |
| **Post-conditions** | Valid simulation is generated and displayed |
| | |
| **Test Case: UI-3 (Priority: 1)** | |
| **ID** | UI-3 |
| **Functionality** | Test simulation generation for sector selection |
| **Description** | Given only a sector input and valid input in other slots this test establishes that a valid sector simulation is generated. |
| **Pre-condition** | User has completed input form, all parameters including test param are valid |
| **Action(s)** | 1. Input valid sector<br>2. Input valid historical context<br>3. Input valid dates for simulation<br>4. Generate simulation |
| **Post-conditions** | Valid<br>simulation is generated and displayed |

| | |
|---|---|
| **Test Case: UI-4 (Priority: 1)** | |
| **ID** | UI-4 |
| **Functionality** | Test input rejection for fake stock IDs |
| **Description** | This test will verify that when we give a stock ID that doesn't exist to the system it doesn't try to generate a simulation but rejects the input. |
| **Pre-condition** | User has completed input form, all parameters excluding test param are valid |
| **Action(s)** | 1. Input invalid stock ID<br>2. Input valid historical context<br>3. Input valid dates for simulation<br>4. Attempt to generate simulation |
| **Post-conditions** | GUI rejects input and does not attempt simulation generation |
| | |
| **Test Case: UI-5 (Priority: 2)** | |
| **ID** | UI-5 |
| **Functionality** | Test input rejection for mostly valid list containing 1 fake stock ID |
| **Description** | This test verifies that the GUI can recognize multi stock input containing only a single incorrectly identified stock to be entirely invalid |
| **Pre-condition** | User has completed input form, all parameters excluding test param are valid |

| | |
|---|---|
| **Action(s)** | 1. Input stock list with valids and single **invalid** stock<br>2. Input valid historical context<br>3. Input valid dates for simulation<br>4. Attempt to generate simulation |
| **Post-conditions** | GUI rejects input and does not attempt simulation generation |
| | |
| **Test Case: UI-6 (Priority: 1)** | |
| **ID** | UI-6 |
| **Functionality** | Test input rejection for invalid simulation date start/finish |
| **Description** | This test verifies that given a start date before data exists or an end date too far in the future the GUI will reject the input as invalid. |
| **Pre-condition** | User has completed input form, all parameters excluding test param are valid |
| **Action(s)** | 1. Input valid stock or list of stocks<br>2. Input valid historical context<br>3. Input **invalid** dates for simulation<br>4. Attempt to generate simulation |
| **Post-conditions** | GUI rejects input and does not attempt simulation generation |
| | |
| **Test Case: UI-7 (Priority: 1)** | |
| **ID** | UI-7 |
| **Functionality** | Test input rejection for all empty fields |

| | |
|---|---|
| **Description** | This test verifies that when an empty form is submitted the GUI rejects the input. |
| **Pre-condition** | None |
| **Action(s)** | 1. Attempt to generate simulation with 0 user input on the input form |
| **Post-conditions** | GUI rejects input and does not attempt simulation generation |
| | |
| **Test Case: UI-8 (Priority: 1)** | |
| **ID** | UI-8 |
| **Functionality** | Test input rejection for only empty stock |
| **Description** | This test verifies that with mostly filled input and an empty stock field the GUI will reject the input |
| **Pre-condition** | User has completed input form, all parameters excluding test param are valid |
| **Action(s)** | 1. Input nothing in stock field<br>2. Input valid historical context<br>3. Input valid date constraints for simulation<br>4. Attempt to generate simulation |
| **Post-conditions** | GUI rejects input and does not attempt simulation generation |
| | |
| **Test Case: UI-9 (Priority: 1)** | |
| **ID** | UI-9 |

| | |
|---|---|
| **Functionality** | Test input rejection for only empty dates |
| **Description** | This test verifies that with mostly filled input and empty date selection(s) the GUI will reject the input |
| **Pre-condition** | User has completed input form, all parameters excluding test param are valid |
| **Action(s)** | 1. Input valid stock or list of stocks<br>2. Input valid historical context<br>3. Do not input anything in date selection for simulation<br>4. Attempt to generate simulation |
| **Post-conditions** | GUI rejects input and does not attempt simulation generation |
| | |
| **Test Case: UI-10 (Priority: 1)** | |
| **ID** | UI-10 |
| **Functionality** | Test multi simulation output for sector + n stocks |
| **Description** | This test verifies that for input with stocks and sectors the correct multi simulation output is generated and displayed. |
| **Pre-condition** | User has completed input form, all parameters including test param are valid |
| **Action(s)** | 1. Input valid stock or list of stocks<br>2. Select a sector also<br>3. Input valid historical context<br>4. Input valid date constraints for simulation<br>5. Generate simulation |
| **Post-conditions** | Valid<br><br>simulation is generated and displayed |

| | |
|---|---|
| | |
| **Test Case: UI-11 (Priority: 1)** | |
| **ID** | UI-11 |
| **Functionality** | Test input rejection for gibberish/meaningless input as stock |
| **Description** | Test verifies that given garbage input for stock field the GUI rejects input entirely |
| **Pre-condition** | User has completed input form, all parameters excluding test param are valid |
| **Action(s)** | 1. Input garbage for stock field<br>2. Input valid historical context<br>3. Input valid date constraints for simulation<br>4. Attempt to generate simulation |
| **Post-conditions** | GUI rejects input and does not attempt simulation generation |
| | |
| **Test Case: UI-12 (Priority: 2)** | |
| **ID** | UI-12 |
| **Functionality** | Test simulation generation for maximum historical context |
| **Description** | Test verifies that with maximum historical context selected a correctly generated simulation is displayed |
| **Pre-condition** | User has completed input form, all parameters including test param are valid |
| **Action(s)** | 1. Input valid stock entry<br>2. Input maximum historical context<br>3. Input valid date constraints for simulation<br>4. Attempt to generate simulation |

| | |
|---|---|
| **Post-conditions** | Valid simulation is generated and displayed |
| | |
| **Test Case: UI-13 (Priority: 2)** | |
| **ID** | UI-13 |
| **Functionality** | Test simulation generation for minimum historical context |
| **Description** | Test verifies that with minimum historical context selected a correctly generated simulation is displayed |
| **Pre-condition** | User has completed input form, all parameters including test param are valid |
| **Action(s)** | 1. Input valid stock entry<br>2. Input minimum historical context<br>3. Input valid date constraints for simulation<br>4. Attempt to generate simulation |
| **Post-conditions** | Valid simulation is generated and displayed |
| | |
| **Test Case: UI-14 (Priority: 2)** | |
| **ID** | UI-14 |
| **Functionality** | Test simulation generation for max simulation duration |
| **Description** | Test verifies that with maximum simulation duration selected a correctly generated simulation is displayed |
| **Pre-condition** | User has completed input form, all parameters including test param |

| | |
|---|---|
| | are valid |
| **Action(s)** | 1. Input valid stock entry<br>2. Input valid historical context<br>3. Input date constraints with maximum simulation length<br>4. Attempt to generate simulation |
| **Post-conditions** | Valid simulation is generated and displayed |
| | |
| **Test Case: UI-15 (Priority: 3)** | |
| **ID** | UI-15 |
| **Functionality** | Test input rejection for above max duration simulation input |
| **Description** | Test verifies that for input requiring a longer simulation than is allowed the GUI rejects input |
| **Pre-condition** | User has completed input form, all parameters excluding test param are valid |
| **Action(s)** | 1. Input valid stock entry<br>2. Input valid historical context<br>3. Input date constraints with above maximum simulation length<br>4. Attempt to generate simulation |
| **Post-conditions** | GUI rejects input and does not attempt simulation generation |
| | |
| **Test Case: UI-16 (Priority: 3)** | |
| **ID** | UI-16 |
| **Functionality** | Test single simulation generated for list of same stock |

| | |
|---|---|
| **Description** | This test verifies that for a list of the same stock (ie, AAPL, AAPL, AAPL) only one simulation is generated and displayed |
| **Pre-condition** | User has completed input form, all parameters including test param are valid |
| **Action(s)** | 1. Input list of same stock ID (AAPL, AAPL, AAPL)<br>2. Input valid historical context<br>3. Input valid date constraints for simulation<br>4. Attempt to generate simulation |
| **Post-conditions** | Valid simulation is generated and displayed |
| | |
| **Test Case: UI-17 (Priority: 3)** | |
| **ID** | UI-17 |
| **Functionality** | Test input rejection for stock input in excess of maximum |
| **Description** | Test verifies that for a list of stocks that exceeds the maximum the GUI rejects the input and explains why |
| **Pre-condition** | User has completed input form, all parameters excluding test param are valid |
| **Action(s)** | 1. Input list larger than maximum size of stock input<br>2. Input valid historical context<br>3. Input valid date constraints for simulation<br>4. Attempt to generate simulation |
| **Post-conditions** | GUI rejects input and does not attempt simulation generation |
| | |
| **Test Case: UI-18 (Priority: 3)** | |

| | |
|---|---|
| **ID** | UI-18 |
| **Functionality** | Test maximum simulations generated with maximum size input stock list |
| **Description** | This test verifies that for a maximally sized list of inputs the corresponding simulations are correctly generated and returned. |
| **Pre-condition** | User has completed input form, all parameters including test param are valid |
| **Action(s)** | 1. Input list equal to maximum size of stock input<br>2. Input valid historical context<br>3. Input valid date constraints for simulation<br>4. Attempt to generate simulation |
| **Post-conditions** | Valid simulation is generated and displayed |
| | |
| **Test Case: UI-19 (Priority: 3)** | |
| **ID** | UI-19 |
| **Functionality** | Test simulation generation for valid stock input padded with spacing on either side |
| **Description** | This test verifies that for stocks input in weird ways (ie, "  AAPL  ") the GUI can still correctly recognize input and display a generated simulation |
| **Pre-condition** | User has completed input form, all parameters including test param are valid |
| **Action(s)** | 1. Input stock padded randomly by spaces ("  AAPL  ")<br>2. Input valid historical context<br>3. Input valid date constraints for simulation<br>4. Attempt to generate simulation |

| | |
|---|---|
| **Post-conditions** | Valid simulation is generated and displayed |
| | |
| **Test Case: UI-20 (Priority: 1)** | |
| **ID** | UI-20 |
| **Functionality** | Test saving simulation saves simulation |
| **Description** | This test verifies that the save functionality on the simulation display functions as it should |
| **Pre-condition** | Valid simulation has been generated and is being displayed |
| **Action(s)** | 1. Fill input with valid inputs for a simulation to be generated<br>2. Generate simulation<br>3. Attempt to save the simulation |
| **Post-conditions** | Application raises a window to save files locally and files successfully save locally |
| **Test Case: UI-21 (Priority: 1)** | |
| **ID** | UI-21 |
| **Functionality** | Test exit closes application at every stage |
| **Description** | This test verifies that exit does its job at every stage. During input & output, during output this is uniquely through the exit button below the chart. |
| **Pre-condition** | None, or valid simulation has been generated and is being displayed |

| | |
|---|---|
| **Action(s)** | 1. Exit on the input form works<br>2. Exit on the simulation output also works |
| **Post-conditions** | Application closes |

| **Test Case: UI-22 (Priority: 2)** | |
|---|---|
| **ID** | UI-22 |
| **Functionality** | Application resizes properly |
| **Description** | This test verifies that the application can resize without any glitches or system breaks. Each element should be consistent with its original size. |
| **Pre-condition** | Application is open |
| **Action(s)** | 1. Open application<br>2. Resize window |
| **Post-conditions** | Application closes |

| **Test Case: UI-23 (Priority: 1)** | |
|---|---|
| **ID** | UI-23 |
| **Functionality** | Elements created appear properly and interact accordingly with user |
| **Description** | This test verifies that each element created (buttons, scales, input boxes) are displayed properly and the user can interact with these elements as intended |
| **Pre-condition** | None, or valid simulation has been generated and is being displayed |

| | |
|---|---|
| **Action(s)** | 1. Open the application<br>2. Click a button, manipulate a scale, or input a value |
| **Post-conditions** | Application closes |

## 3 Additional Information

### *3.1 Acronyms, Abbreviations*

| **Value** | **Definition** |
|---|---|
| DAITrader | Name of the application |
| User | User identification of the user installing and running the application |
| SRS | Software Requirements. |
| Schema | A schema is the organization or structure of a database. The activity of data modeling leads to a schema. |
| Specification Connects | Links this requirement with another |
| Includes | Has the appropriate constraint in it |
| Extends | Shows or cancels a constraint effect if the conditions are met. |
| ToS | Terms of Service agreement defining terms that customers must agree to before use of application. |
| RSA | Encryption algorithm based on ‘the factoring problem’ that utilizes a public and private key to maintain data obfuscation. |
| User Stories | Presents a user description based on a potential use of a certain software requirement within the application. |
| Domain Class Diagram | A diagram featuring the domain requirements for the trajectory of the application procedures. |
| API | Application programming interface |
| GUI | Graphical User Interface |
| OS | Operating System |

| | |
|---|---|
| PAAS | Platform as a service |

### *3.2 Reference Documents*

| **Title** | **Description** |
|---|---|
| System Test Plan | The purpose of this document is to specify the system test plan logistics and implementation for DAITrader. It is to clarify and explain the actions required for testing DAITrader's infrastructure. |
| Software Requirements Specification | The purpose of this document is to specify the software requirements for DAITrader. It is to clarify and explain the requirements needed in order for the application to function for both programmers and users. |
| Software Design Document | The purpose of this document is to specify the software requirements for DAITrader. It is to clarify and explain the requirements needed in order for the application to function for both programmers and users. |