Introduction Document

Quantitative Development – Spring 2025

Trading at Georgia Tech Ayush Gundawar

1 Introduction

1.1 Document Goals

The purpose of this document is to introduce you to our tech stack and development process through a minimal project. The project should take a few hours at most. Once your project is accepted, you'll be given access to Trading at Georgia Tech repositories.

1.2 Stack Overview

Our primary programming language of choice is **Rust**. The reasoning is simple:

1. It lends itself to being used to write high-performance systems.
2. Unlike C++ and other C flavors, it encourages memory and thread safety by default.
3. Though subjective, it's pretty similar to C++.

In addition to Rust, we use **Python** for research and scripting, such as quoter deployment utilities. For research, we analyze and visualize market data using Jupyter notebooks.

1.3 Definitions

- **Quoter**: An instance of a trading engine. A quoter may manage a few, related strategies. Quoters are generally composed of listeners, strategies, and order gateways.
- **Listener**: A connection handler that manages a market data feed from an exchange. Listeners can be asynchronous or synchronous.
- **Strategy**: The algorithm that decides what, when, and how to trade based on the information fed to it by the dispatcher and pricing engine. Generally, it can be thought of as a black box that takes market data and theoretical prices as inputs and outputs a set of trading decisions based on some rules.
- **Order Gateway**: API for sending orders to a specific exchange based on that exchange's entry protocol.

1.4 Required Readings

We require all new members to read selected sections of the Rust Book and Tokio documentation so you can understand existing code.

To be blunt, it'll be pretty obvious if you're not familiar with the content covered in the assigned sections. Don't be the person tripping up on basic language errors – we have more important problems to solve! These are core concepts used in all of our systems.

1.4.1 Rust Book Sections

The Rust book can be found here: `https://rust-book.cs.brown.edu/`

- 1. Getting Started
- 3. Common Programming Concepts
- 4. Understanding Ownership
- 6. Enums and Pattern Matching
- 7. Managing Growing Projects with Packages, Crates, and Modules
- 9. Error Handling
- 10. Generic Types, Traits, and Lifetimes
- 11. Writing Automated Tests
- 15. Smart Pointers
- 16. Fearless Concurrency
- 17. Object-Oriented Programming Features of Rust
- 19. Advanced Features

1.4.2 Tokio Documentation Sections

- Tokio Overview
- Spawning Tasks
- Shared State
- Channels Overview
- Deep Async
- Streams Overview

2 Warm-up Project

The **due date** for this warm-up project is **EOD on February 20th, 2025**.

2.1 Description

You are working on a cryptocurrency trading desk. A researcher wants to visualize the interesting price levels for ETH-USDT futures.

Your task is to write an order book that:

1. Ingests level 2 (i.e., 5 best bid/ask orders) `ETH-USDT` futures data from the KuCoin WebSocket.
2. Prints the state of the order book in columnar format after every market data update.

2.2 Requirements

- Written in Rust, using the Tungstenite crate to connect to the KuCoin WebSocket server.
- Has a clean public API and is reasonably efficient.
- Includes well-written documentation for all public methods and associated functions.

2.3 Submission

Publish the final version on GitHub, and email the repository link to `agundawar3@gatech.edu` by **EOD on February 20th, 2025**.