Project Report: Binance Futures CLI Bot Author: Ketan Date: October 2, 2025

1. Introduction This report details the design, implementation, and features of a command-line interface (CLI) trading bot for the Binance USDT-M Futures platform. The primary objective was to build a modular, robust, and well-documented application supporting multiple order types, as outlined in the assignment specifications. The project was successfully completed, meeting all mandatory and bonus requirements, including the integration of the "Fear & Greed Index" as an analysis tool.

2. Project Architecture A key decision was to adopt a modular architecture to ensure the project is scalable and maintainable. The code is organized by function rather than being a single monolithic script.

/src/core: This directory acts as the brain of the application. It handles all core services, including the API client for connecting to Binance (client.py), configuration management (config.py), and the structured logger (logger.py).

/src/utils: This directory contains helper modules, such as the validation.py script, which centralizes input validation logic.

Execution Modes: The bot supports two execution modes. Individual order scripts (market_order.py, etc.) allow for direct command-line execution for automation. The main interactive_bot.py script provides a user-friendly menu that leverages the same core logic for a guided experience.

3. Features Implemented Dual Execution Modes: The bot can be operated via a rich, user-friendly interactive menu (interactive_bot.py) or through direct command-line arguments.

Core Orders (Mandatory): Market and Limit orders are fully implemented in both interactive and direct command modes.

Advanced Orders (Bonus): Stop-Limit, OCO, TWAP, and Grid strategies are all fully implemented within the interactive UI in simulation mode.

Bonus Integration (Fear & Greed Index): The bot can read the provided CSV data and display the latest Fear & Greed Index value, providing users with a quick market sentiment indicator.

Structured Logging: A robust logger was configured to record all operations into bot.log. Each entry is timestamped and includes the module of origin for easy debugging.

Input Validation: A centralized validation.py module performs basic checks on user-provided arguments to prevent common errors.

4. Usage Screenshots (IMPORTANT: You need to add your own screenshots where indicated below.)

Example 1: Running the Interactive Menu The bot's primary interface is a user-friendly menu built with the rich library.

Command:

py interactive_bot.py

Example 2: Bonus Feature - Fear & Greed Index The bot can parse and display the latest market sentiment data from the provided CSV file.

Command from menu: fg

Example 3: Placing an Advanced Order (OCO) This screenshot shows the confirmation after placing a simulated OCO order using the interactive menu.

5. Conclusion This project successfully fulfills all requirements of the assignment. The resulting application is a feature-complete, well-documented, and extensible trading bot that offers both a powerful interactive UI for manual use and direct script execution for automation. The project demonstrates strong programming practices and a clear understanding of software architecture.