Project Proposal

Course: INF1002 Programming Fundamentals
Team ID: P1-8

1. Project Title: Stalking Stocks

2. Team List

Name	Student ID	Email
Mohammed Aamir	2500933	2500933@sit.singaporetech.edu.sg
Muhammad Hasif Bin Mohd Faisal	2500619	2500619@sit.singaporetech.edu.sg
Timothy Chia Kai Lun	2501530	2501530@sit.singaporetech.edu.sg
Low Gin Lim	2501267	2501267@sit.singaporetech.edu.sg
Dalton Chng Cheng Hao	2504003	2504003@sit.singaporetech.edu.sg

3. Project Description

Our project is a comprehensive stock-analysis web application, leveraging pre-loaded financial data from multiple sectors (technology, finance, healthcare, consumer goods, and energy) for the past 2-3 years.

4. Problem Statement

Novice investors often find the stock market overwhelming due to the need to navigate sheer volumes of data and complex financial instruments. This makes understanding the current financial landscape a tedious task, leaving beginners confused or discouraged. Our project, **Stalking Stocks**, aims to address this gap by simplifying financial data through intuitive and interactive visuals, as well as data-driven insights to empower the user to learn and make informed decisions confidently in a risk-free environment.

5. Objectives

1. Implement efficient, **modular Python** functions for technical indicators and performance metrics using sliding-window and dictionary-based algorithms.
2. Develop a **streamlined web interface** Streamlit) that delivers interactive charts and dashboards with minimal latency.
3. Validate analytical outputs against trusted sources (e.g., pandas' .rolling().mean()) with at least five test cases per core function.
4. Demonstrate **clear logic**, maintainable code structure, and effective team coordination throughout all project phases.

6. Features

- **Browse** stocks by company or sector
- **View** interactive price charts with technical indicators (Simple Moving Averages)
- **Analyse** performance metrics, including daily returns, trend runs, and maximum profit calculations
- **Access** a market-overview dashboard showcasing top/worst performers, volatility rankings, and longest trend streaks
- **Compare** up to four stocks simultaneously
- **Review** sector performance comparisons across different time periods
- **Explore** "Market Moments" highlighting stock reactions to significant events Fed announcements, earnings seasons, market corrections)

All data will be pre-processed and embedded in the application, ensuring instant analysis without external uploads.

7. Initial Task Allocation

Phase	Team Member	Purpose	Key Activities	Deliverables
1	All Members	Project Architecture & Methodology	- Define modular structure - Set up project repository file/folder layout - Plan core functions (inputs/outputs) - Choose a programming paradigm - Agree on design philosophy - Outline documentation practices	- Project folder/file structure - Design plan (module/function list) - Coding conventions/style guide - Initial README
2	Aamir, Dalton	Data Acquisition, Preprocessing, Transformation & Validation	- Retrieve historical price data via yfinance API - Remove non-trading dates, weekends, and duplicates - Handle missing values and standardize date formats - Transform data into analysis-ready structures (pivot tables, normalized series) - Identify and flag price outliers - Generate data-quality summary reports	- Analysis functions library - Algorithm documentation - Test results summary

3	Hasif, Gin	Core Algorithm Development	- Implement sliding-windows SMAs - Develop trend-run detection for consecutive up/down streaks - Calculate daily returns and volatility metrics - Build max-profit algorithm (multiple transactions) - Write unit tests for each function - Identify, review, and handle edge cases	- Analysis functions library - Algorithm documentation - Test results summary
4	Timothy, Gin, Dalton	Full-Stack Interface & Data Services (Streamlit)	- Load and process CSV data on demand - Design Streamlit dashboard layout - Integrate Plotly charts for prices, SMAs, returns, and trend highlights - Build interactive selectors (tickers, sectors, date ranges) - Implement multi-stock comparison view - Ensure responsive design and UX polish - Ensure a modular service layer for future features	- Streamlit app interface - Interactive chart components - Defined service layer modules - UI design notes - API documentation
5	All Members	Integration, Testing & Deployment	- Integrate backend and frontend components - Conduct end-to-end functional testing - Validate outputs vs. pandas/manual calculations (≥5 scenarios) - Debug performance and rendering issues - Prepare deployment package and user guide	- Deployed application package - Final test report - Setup and usage documentation

8. References

Flask Documentation. (n.d.). *Flask documentation*. Pallets Projects. Retrieved September 6, 2025, from https://flask.palletsprojects.com

LeetCode. (n.d.). *Best time to buy and sell stock II*. LeetCode. Retrieved September 6, 2025, from https://leetcode.com

Pandas Development Team. (2023). *pandas.DataFrame.rolling API reference*. pandas. Retrieved September 6, 2025, from https://pandas.pydata.org

Python Software Foundation. (2001). *PEP 8 — Style guide for Python code*. Python Software Foundation. Retrieved September 6, 2025, from https://peps.python.org/pep-0008/

Plotly Technologies Inc. (2023). *Plotly Python graphing library*. Plotly. Retrieved September 6, 2025, from https://plotly.com/python/

Streamlit Inc. (2024). *Streamlit documentation*. Streamlit. Retrieved September 6, 2025, from https://streamlit.io

YFinance Documentation. (n.d.). *yfinance*. PyPI. Retrieved September 6, 2025, from https://pypi.org/project/yfinance/

Google Developers. (2025). *Responsive design principles*. Google. Retrieved September 6, 2025, from https://developers.google.com/web/fundamentals/design-and-ux/responsive

IEEE. (2019). *IEEE standard for software and system test documentation (IEEE 829-2019)*. IEEE.