



StockOverflow:

Simulate Market Trends

Students: Brandon Rodriguez, Lorenzo Zaidowicz, Gabriel Guzman, Diego Peralta, Alex Gomez

Instructor/Faculty: Masoud Sadjadi *Florida International University*

Introduction

- Generates simulated stock price movements based on historical data, economic indicators, and market conditions.
- Adjustable macro-variables like interest rates, inflation, and sector performance.
- Visualizes trends in real time with charts, heat maps, and performance tracker.
- Back-testing and forecasting
- Automated alerts and signal detection flags potential trend reversals, breakout opportunities, and risk conditions.





Introduction cont.

Functional Requirements:

- [Chart.js](#)
- huggingface
- ZeroShot Classifier (huggingface)

Non-Functional Requirements:

- Node v16.16.0 (LTS)
- React 18.x
- [Next.js](#)
- GitHub
- Vercel (Hosting)

Deployment/Implementation

- Next.js framework powers the application.
- React front end fetches the JSON generated by the simulation and renders it as an interactive chart with [Chart.js](#).
- Persistent data is stored in a GitHub Repository.
- Majority of the project is coded in JavaScript.



System Design

- StockOverflow's display of market stocks are modeled after Brownian Motion.
- Our system's simulation engine implements a GMB equation that takes drift and volatility into account when simulating stock market trends:

$$dS(t) = \mu S(t)dt + \sigma S(t)dW(t)$$

μ - drift σ - volatility

- System takes input of real-world stock ticker symbols (e.g. AMZN (Amazon), MSFT (Microsoft)) and auto inputs volatility and drift parameters of the specific symbol.

System Design cont.

StockOverflow

$$dS_t = \mu S_t dt + \sigma S_t dW_t$$

S_t : Stock price at time t (your Initial Price and generated data)

μ (Drift): Expected return rate of the stock. Can be interpreted as the slope of the stock. (Drift % input)

σ (Volatility): Random fluctuations or variability. Higher volatility = greater fluctuations (Volatility % input)

dW_t : Random Brownian motion shock (simulated by random \pm volatility changes)

Our simulation currently models changes based on the brownian motion equation, which depends on user inputs!

Stock Symbol

AMZN

Fetch

Initial Price

100

Volatility (%)

2.15

Drift (%)

0.1

Days (Max Points to Show)

100

Stop



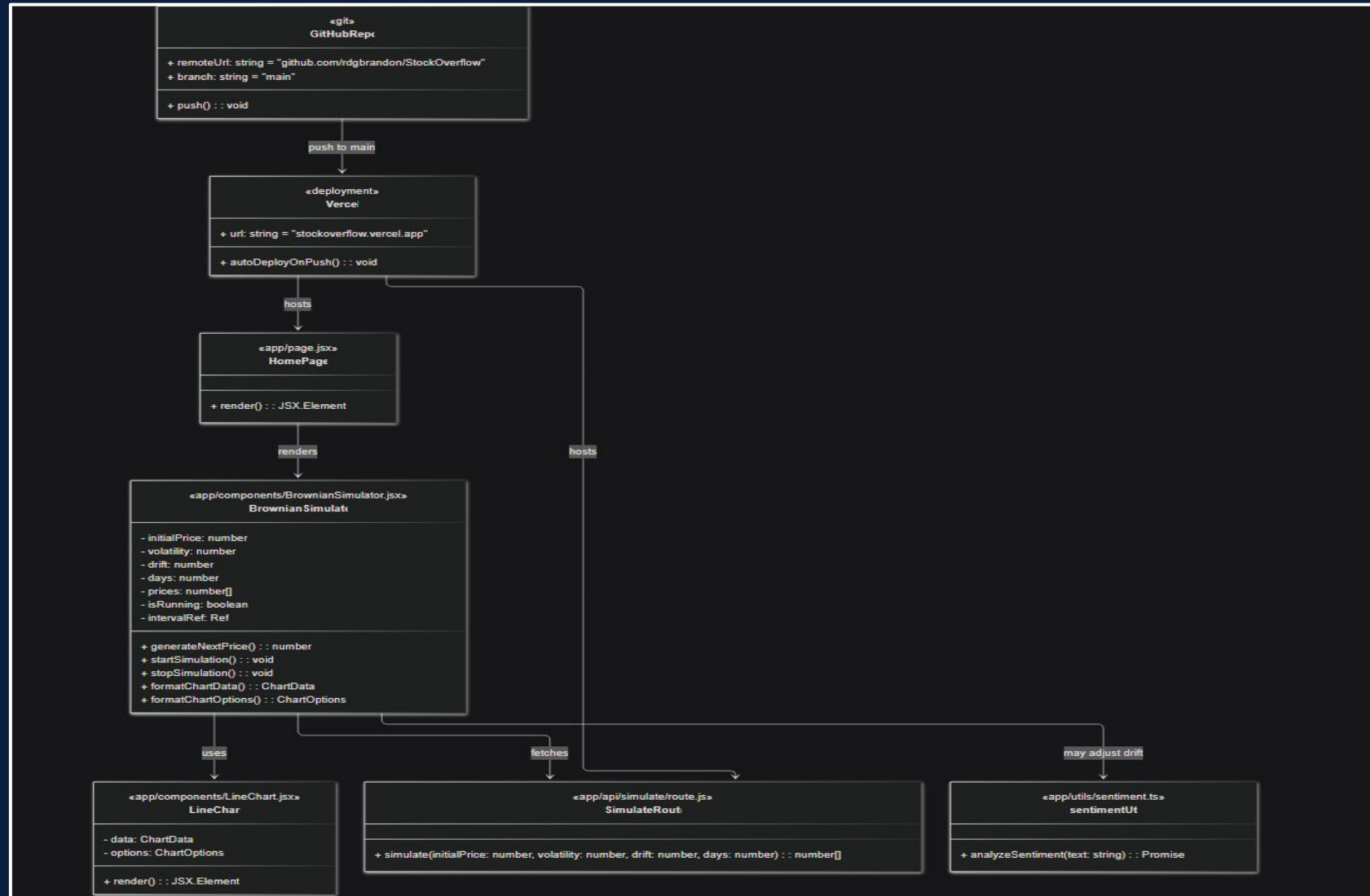


Detailed Design

- Uses Four Tier Architecture
- Maintained and updated on GitHub Repository
- Client-side hosted by Vercel
- Brownian Simulator:
 - Uses LineChart and renders JSX element
 - Fetches route JS component to simulate route
 - Utilizes real market drift and volatility parameters.



Detailed Design cont.





Future Implementations

- Option for user to extract graph data into a file.
- Pause/Resume feature for the simulation.
- Implement an event-based sentiment analysis.
 - System takes String input of an “event” and uses sentiment analysis to classify “event” as positive or negative for the market. (e.g. Company releases new, successful product.)
 - Positive = (+)drift = increasing stock slope.
 - Negative = (-)drift = decreasing stock slope.
- Keyword -> Stock Symbol feature





Questions?