

BlueWhale Terminal - README Summary

BlueWhale Terminal

Modern Investment Research Platform for JSE Small-Cap Insights

Frontend: React + TailwindCSS | Backend: Node.js + PostgreSQL | AI-ready architecture

Project Overview

BlueWhale Terminal is a financial research dashboard inspired by Bloomberg and TradingView, designed to help analysts, investors, and financial institutions analyze small-cap markets in South Africa. It delivers a terminal-like experience with a clean UI, modular architecture, and future-ready integrations (AI, live data APIs, payment systems).

Tech Stack

Frontend: React 18 + TypeScript

Styling: TailwindCSS

State Management: React Query / Redux Toolkit

Routing: React Router v6

Backend: Node.js (Express)

Database: PostgreSQL

ORM: Prisma

Scraping: Puppeteer / Cheerio

AI Layer (future): OpenAI API

Payments (future): Stripe / Paystack

Repository Structure (simplified)

client/ - React Frontend (components, layouts, pages, hooks, services, store, types, utils)

server/ - Node.js Backend (routes, controllers, services, config, middleware, prisma, tests)

Setup & Installation

1. Prerequisites: Node.js ≥ 18 , npm, PostgreSQL ≥ 14 , Git
2. Clone repository: `git clone https://github.com//bluewhale-terminal.git`
3. Create .env files for client and server
4. Install dependencies: `npm install` (in client/ and server/)
5. Run database migrations: `npx prisma migrate dev`
6. Start servers: `npm run dev` (backend) and `npm run dev` (frontend)

Features (Phase 1 MVP)

Login Page – secure branded login (mock auth)

Dashboard Layout – persistent sidebar, header, content

Screener – sortable stock data table

Watchlist – track user tickers

Reports Section – mock downloadable reports

AI Hub – sentiment analysis, summary, DCF calculator placeholders

Subscription Page – pricing tiers and upgrades

Data Scraper – Node.js scraper for financial data

PostgreSQL Integration – persistent backend storage

Testing

Unit & Integration tests using Jest, 80% coverage recommended.

Deployment

Frontend: Netlify or Vercel

Backend: Render or Railway

Database: PostgreSQL managed instance

Definition of Done

Responsive UI, PostgreSQL connection, API endpoints functional, scraper prototype operational, code linted, and documented.

License

Private license © 2025 BlueWhale Technologies.