**Project Scope – Stock Prediction Using Machine Learning Web Application**

**By Jonathon Mitchell**

**Project Objectives**

* Develop a Web Application for Stock Price Prediction:
  + Create a user-friendly web application that allows users to input historical stock market data and receive predictions regarding whether a specific stock will go up or down on the next trading day.
* Train and Implement a Machine Learning Model:
  + Build and train a robust machine learning model that leverages historical stock market data for accurate predictions. This model should be capable of handling various stocks and historical data sets.
* Real-Time Data Integration (Optional):
  + Consider integrating real-time stock market data feeds (if available) to provide up-to-the-minute predictions and enhance user experience. This could be an advanced feature if time permits.
* Interactive User Interface:
  + Design an intuitive and visually appealing user interface that provides users with a seamless experience. Include features like data visualization, charting, and easy data input.
* Performance Optimization:
  + Ensure the web application's performance is optimized for efficient predictions and responsiveness. This may involve optimizing code, caching, and server scaling strategies.
* Documentation and User Instructions:
  + Create comprehensive documentation for your web application, including clear instructions for users on how to use the application effectively.
* Deployment on Hosting Platform:
  + Deploy the web application to a reliable hosting platform (e.g., Heroku, AWS, or Azure) to make it accessible to users on the internet.
* Monitoring and Maintenance Plan:
  + Develop a plan for monitoring the application's performance and resolving issues promptly. Consider setting up logging and alerting mechanisms.

**Stakeholder Identification:**

* Project Owner:
  + Description: The primary stakeholder is the project owner, Jonathon Mitchell, who is undertaking this personal project to develop and showcase skills on a CV.
  + Interest: The project owner's main interest is self-improvement, skill development, and enhancing their employability.
* Potential Employers or Hiring Managers (Indirect Stakeholders):
  + Description: Potential employers or hiring managers are indirect stakeholders who may review the project on the project owner's CV.
  + Interest: Their interest lies in evaluating the project owner's skills, knowledge, and practical experience demonstrated by the project.
* Fellow Developers or Peers:
  + Description: Fellow developers and peers in the tech community may interact with and provide feedback on the project.
  + Interest: They have an interest in knowledge sharing and learning from the project.
* End Users (if publicly deployed):
  + Description: If the web application is made publicly accessible, end users are stakeholders who may use the application for stock predictions.
  + Interest: Their interest centres around the usability and effectiveness of the application for stock market predictions.

**Project Deliverables:**

* Functioning Web Application:
  + A fully operational web application accessible via a web browser, where users can input historical stock data and receive predictions on whether a specific stock will go up or down on the next trading day.
* Trained Machine Learning Model:
  + A machine learning model trained on historical stock market data that can make accurate stock price predictions. This includes the model file and any necessary pre-processing or feature engineering scripts.
* User Interface (UI) Design:
  + The design components and assets for the user interface, including layout, graphics, and user interaction elements, to create an intuitive and visually appealing user experience.
* Documentation:
  + Comprehensive documentation, including user guides, installation instructions, and an explanation of how the machine learning model works. This documentation should be clear and user-friendly.
* Source Code:
  + The complete source code for both the frontend (React) and backend (Node.js with Express.js), including any relevant scripts, configurations, and dependencies.
* Deployment Package:
  + A package containing all necessary files and instructions for deploying the web application on a hosting platform, such as Heroku, AWS, or another provider.
* Security Measures Implementation:
  + Documentation outlining the security measures implemented in the application to protect user data and ensure data integrity.
* Performance Optimization Documentation:
  + Documentation highlighting the performance optimization strategies employed in the application, such as code optimizations and caching mechanisms.
* Version Control Repository:
  + A version control repository (e.g., on GitHub) where the project's source code and related files are hosted for version tracking, collaboration, and community engagement.
* Monitoring and Maintenance Plan:
  + A plan outlining how the application will be monitored, maintained, and updated to ensure its reliability and sustainability.

**Functional Requirements:**

* Users will be able to input a company’s name, and if it is in the data that the model has been trained on, the model will output whether the stock will go up or down the next day with a certain level of accuracy.

**Non-Functional Requirements:**

* Performance Requirements:
  + Response Time: The web application should respond to user interactions within 2 seconds.
  + Scalability: The application should be designed to gracefully handle increased user traffic without experiencing significant performance degradation.
  + Load Testing: Load testing should be conducted to validate the application's responsiveness under various user loads.
* Reliability Requirements:
  + Availability: The application should maintain a high level of availability, with a target uptime of 99% to ensure consistent user access.
  + Fault Tolerance: The application should implement measures to gracefully handle unexpected failures, minimising downtime, and service disruption.
* Usability Requirements:
  + User Interface Consistency: The application must maintain a consistent and user-friendly interface throughout all its pages and features.
  + Accessibility: The application should be accessible to users with disabilities, complying with web accessibility standards such as WCAG.
* Compatibility Requirements:
  + Cross-Browser Compatibility: The application must function correctly on popular web browsers like Chrome, Firefox, and Safari.
  + Mobile Responsiveness: The application should be responsive and user-friendly on various devices, including smartphones and tablets.
* Documentation Requirements:
  + User Documentation: The application should provide user-friendly documentation to assist users in effectively utilising its features.
  + Code Documentation: Maintain clear and comprehensive code documentation to facilitate future development and maintenance efforts.

**Constraints:**

* Time Constraint: There is no time constraint for this project. It should be completed within 3 months.
* Budget Constraint: The budget for this project will be £0.
* Technology Stack Constraint: This project is restricted to using only libraries and technologies that are accessible for free.
* Data Availability Constraint: Since this project relies on external data sources, there may be constraints in terms of data availability, API access, data update frequency and data source availability.
* Resource Constraint: The project will be developed on a 2020 Intel-based MacBook Pro laptop. Due to budget constraints, only free and open-source software, libraries, and development tools will be utilised throughout the project's lifecycle.
* Legal or Compliance Constraints: The project exclusively accesses publicly available stock data, which is not subject to data privacy or compliance regulations typically associated with personal or sensitive information.

**Assumptions:**

* Data Availability Assumption: It is assumed that historical stock data will be readily accessible for model training through publicly available data sources or APIs. Any potential limitations or constraints related to data availability will be addressed as necessary.
* User Assumptions: It is assumed that users of the application have a basic understanding of stock market concepts and terminology, enabling them to interpret the predictions effectively. Users are assumed to have access to standard web browsers and the internet, allowing them to use the web application.
* ­Development Environment Assumption: It is assumed that the development environment will include a 2020 Intel-based MacBook Pro laptop as the primary development machine. Development tools and software used will be freely available, open-source options to align with budget constraints. Access to a reliable internet connection for research, development, and deployment is assumed.
* Scope Assumption: The project scope is initially focused on implementing stock price prediction based on historical data. Future enhancements or additional features may be considered as separate project phases.
* Project Team Assumption: The project will be developed as an individual effort, with roles and responsibilities managed by a single developer. Availability for the project's development will be primarily during personal time outside of other commitments.
* External Dependencies Assumption: It is assumed that necessary external dependencies, such as machine learning libraries and APIs for stock data, will be available and reliable throughout the project. Contingency plans will be in place for addressing any disruptions.
* Performance Assumption: Initial performance expectations for the machine learning model assume a reasonable level of accuracy based on standard machine learning practices. Performance will be evaluated and optimised as the project progresses.
* Regulatory Compliance Assumption: Given that the project primarily deals with publicly available stock data and doesn't involve sensitive user information, it is assumed that the project will not have specific regulatory compliance requirements. Any legal or compliance changes will be addressed as necessary if they arise.

**Scope Boundaries:**

* This project is limited to the development of a web application that predicts stock price movements based on historical data. It does not encompass the following:
  + Financial Advice: The web application will not provide financial or investment advice to users. All predictions and insights generated by the application are for informational purposes only and should not be considered as financial recommendations.
  + User Account Functionality: User account management and related features (e.g., user profiles, authentication) are outside the scope of this project. The application will not require users to create accounts or store personal information.
  + Advanced Analytics: Complex financial analysis, advanced technical indicators, or other specialised analytics beyond basic stock price prediction are beyond the scope of this project.
  + Mobile Applications: The project scope does not include the development of mobile applications for iOS or Android platforms. It focuses on web-based functionality accessible via standard web browsers.
  + Extended Historical Data: The application will use publicly available historical stock data, but it will not include extensive historical data retrieval and storage beyond what is necessary for prediction modelling.
  + Algorithm Customisation: While the project will implement a stock prediction algorithm, it will not provide user customisation or tuning of the underlying algorithm parameters.

**Project Timeline:**

* Project Initiation (Week 1)
  + Define project scope, objectives, and constraints.
  + Set up the development environment.
  + Research and select the machine learning algorithm for stock prediction.
* Data Collection and Pre-processing (Week 2)
  + Gather stock data.
  + Pre-process and clean the data for model training.
  + Prepare the dataset for training and testing.
* Machine Learning Model Development (Week 3)
  + Learn and implement the selected machine learning algorithm.
  + Train the initial version of the model.
  + Evaluate and refine the model's accuracy.
* Web Application Development (Week 4)
  + Learn and implement the React and Node.js frameworks.
  + Develop the user interface for the web application.
  + Integrate the machine learning model with the application.
* Testing and Quality Assurance (Week 5)
  + Perform testing, including unit testing and integration testing.
  + Identify and resolve any bugs or issues.
  + Ensure the application meets performance requirements.
* Documentation and Deployment (Week 6)
  + Create user documentation and code documentation.
  + Prepare the application for deployment to a hosting platform.
  + Deploy the web application to make it accessible to users.
* Project Completion and Review (Week 7)
  + Finalise and review the project.
  + Ensure all project objectives and requirements have been met.
  + Consider future enhancements or additions to the project.

**Potential Risks and Mitigation Plans:**

* Data Availability and Quality:
  + Risk: The availability and quality of historical stock data may vary, affecting the accuracy of the machine learning model.
  + Mitigation: Use multiple data sources or APIs to cross-verify data. Implement data validation and pre-processing techniques to address data quality issues. Develop contingency plans to handle data source outages.
* Learning Curve:
  + Risk: Learning new technologies like React, Node.js, and machine learning algorithms may take longer than anticipated, causing project delays.
  + Mitigation: Allocate sufficient time for learning and experimentation with new technologies before the development phase. Consider seeking online tutorials, courses, or mentorship for faster skill acquisition.
* Algorithm Performance:
  + Risk: The machine learning algorithm's initial performance may not meet expectations.
  + Mitigation: Continuously monitor and evaluate the model's performance during development. Implement optimisation techniques and fine-tune parameters as needed to improve accuracy.
* Resource Constraints:
  + Risk: Limited access to hardware or software resources may lead to performance bottlenecks or compatibility issues.
  + Mitigation: Prioritise resource-efficient development practices. Optimise code and minimise resource usage. Consider cloud-based solutions for scalability.
* Time Constraints:
  + Risk: Unexpected delays or interruptions in your schedule may affect the project timeline.
  + Mitigation: Create a detailed project plan with buffer time for unforeseen issues. Regularly track progress to ensure timely completion. Adjust timelines as needed.
* Deployment and Hosting Issues:
  + Risk: Deployment to a hosting platform may encounter technical difficulties or compatibility issues.
  + Mitigation: Conduct thorough testing in a staging environment before deployment. Plan for rollbacks in case of deployment failures. Seek assistance from hosting platform support if necessary.
* External Service Reliability:
  + Risk: Reliance on third-party services or APIs may lead to service disruptions or changes in service availability.
  + Mitigation: Have backup plans in place for service interruptions. Monitor third-party service status and adapt your application to accommodate changes in their offerings.