Commodity Price Prediction

Kaavish Project SRS Document
by

Syed Hamza	hs07141
Aiza Imran	ai07597
Sidra Aamir	sa07316
Owais Aijaz	oa07610

In partial fulfillment of the requirements for
Bachelor of Science
Computer Science

Dhanani School of Science and Engineering

Habib University
Fall 2024

1 Introduction

The purpose of this document is to present a comprehensive Software Requirements Specification (SRS) for the "Commodity Price Prediction" project for Pakistan Agriculture Research (PAR). This document defines the scope, requirements, and key architectural components essential for developing a reliable and user-friendly prediction web application. The SRS outlines functional requirements—including accurate price prediction using machine learning techniques, user-friendly interfaces for traders, the ability to visualize pricing trends through intuitive dashboards, etc—and non-functional requirements such as performance, scalability, security, usability, and more. Additionally, it includes use cases demonstrating various user interactions with the system, helping stakeholders understand the software's primary functions and expected outcomes, along with a system diagram and prototype to provide a visual representation of the web application. This document serves as a foundational reference for the design, development, and implementation phases, ensuring that the web application meets the specific needs of Pakistan's agricultural sector by providing data-driven insights to effectively manage commodity price volatility.

2 Project Plan

2.1 Project Objectives

- **Develop an Accurate Commodity Price Prediction Model:** Use advanced Machine Learning techniques to predict agricultural commodity prices based on historical data, weather patterns, and economic indicators (external factors).
- **Build a User-Friendly Web Application and Dashboard:** Create a web app and dashboard with interactive data visualization and customizable features for traders and stakeholders
- **Integrate Real-Time External Data:** Incorporate streaming data (e.g., weather, market news, government policies, etc) and build pipelines to update the prediction model continuously.
- **Enhance Agricultural Decision-Making:** Empower stakeholders by providing timely insights to reduce financial risks and optimize crop production strategies.

2.2 Tasks

- **Data Cleaning and Formatting**

  - **Data Preprocessing:** Identify and handle missing or inconsistent data.
  - **Data Formatting:** Standardize data formats to meet model requirements for each commodity.
  - **Feature Engineering:** Create new features or transform data to enhance model performance.
  - **Commodity-Specific Data Split:** Organize data by commodity.

- **External Factors Research**
  - **Identify Relevant Factors per Commodity:** Research factors (e.g., weather, demand trends) affecting each commodity.
  - **Experiment with Factor Incorporation:** Test combinations of factors to determine significance for each commodity's price.
  - **Define Factor Parameters:** Decide which factors to include in each commodity's model.
  - **Document Factor Influence:** Compile a reference document on each factor's impact.

- **NLP Data Extraction**
  - **NLP Model Selection:** Choose suitable NLP models (e.g., BERT, GPT) for data extraction for specific news and weather reports.
  - **Data Preprocessing for NLP:** Clean and preprocess text data (e.g., tokenization, stop-word removal).
  - **Train NLP Models:** Set up models to extract relevant trends or events impacting commodities.
  - **Organize Extracted Data:** Store data by commodity for easy model integration.

- **Model Development (Commodity-Specific)**
  - **Model Experimentation:** Test models (LSTMs, Classical ML, RNNs) on each commodity to capture price trends.
  - **External Factor Analysis:** Assess impact of factors (e.g., weather, economy) on predictions. Adjust factors as needed.
  - **Hyperparameter Tuning:** Optimize model parameters per commodity.

  - **Performance Evaluation:** Assess model accuracy and reliability by commodity, noting the best factors.
  - **Model Selection:** Choose and document the best model for each commodity.

- **App and Dashboard Development**
  - **Wireframe Design:** Design UI/UX wireframes.
  - **Frontend Framework Selection:** Select a UI framework (e.g., React, Tailwind CSS, Redux).
  - **Homepage and Dashboard Layout:** Create a user-friendly homepage and interactive dashboard.
  - **Data Visualization Integration:** Add charts, graphs, and tables for model outputs.
  - **User Interaction Features:** Implement search filters, alerts, and comparison tools.
  - **Multilingual Support:** Add language options for Urdu and English.

- **Real-Time Data Pipeline Setup**
  - **Identify Data Sources:** Select APIs (e.g., weather, market prices) for real-time data.
  - **Data Integration Pipeline:** Build pipelines to integrate data into models.
  - **API Integration:** Connect to APIs for real-time data fetching.
  - **Data Processing Automation:** Automate data cleaning and formatting.
  - **Database Update Mechanism:** Implement a system to update the database with new data.
  - **Error Handling:** Develop methods to manage potential data source failures.

- **Backend Development**
  - **Database Design:** Design schema for data, model outputs, and user info.

  - **Backend Framework Selection:** Choose a framework (e.g., Django, Flask).
  - **API Development:** Develop APIs for data requests.
  - **Authentication Setup:** Implement user authentication.
  - **Cloud Integration:** Set up cloud-based infrastructure (e.g., AWS, Google Cloud).
  - **Model Integration:** Link models to backend for real-time predictions.
  - **Scalability Optimization:** Ensure backend can handle increased traffic.

- **Testing and Validation**
  - **Unit Testing:** Test individual components (e.g., model, API, UI).
  - **Integration Testing:** Test component interactions (e.g., frontend-backend).
  - **System Testing:** Conduct end-to-end application testing.
  - **Model Validation:** Validate models with new data.
  - **Usability Testing:** Collect user feedback on app interface and features.
  - **Bug Fixing:** Identify and resolve any issues.
  - **Final Validation:** Perform final testing for reliability.

- **Documentation and User Guide Creation**
  - **Technical Documentation:** Document architecture, models, and backend.
  - **User Guide:** Create a manual on app features and navigation.
  - **FAQ Section:** List common questions and answers.
  - **System Requirements:** Outline hardware/software requirements.
  - **Setup Instructions:** Provide setup instructions for local/server use.
  - **Update Documentation:** Detail processes for updates and maintenance.

2.3 Timeline

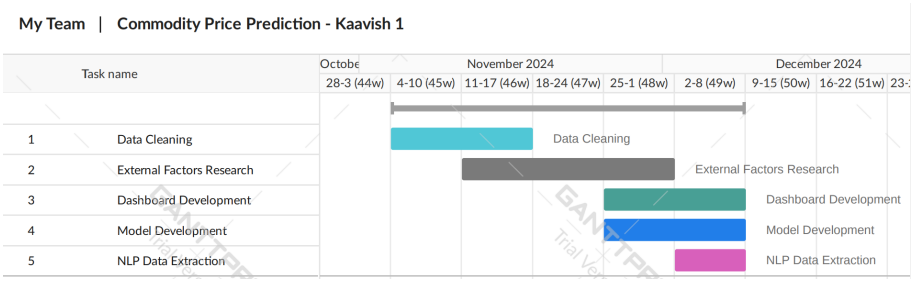

Figure 1: Project Plan for Kaavish 1

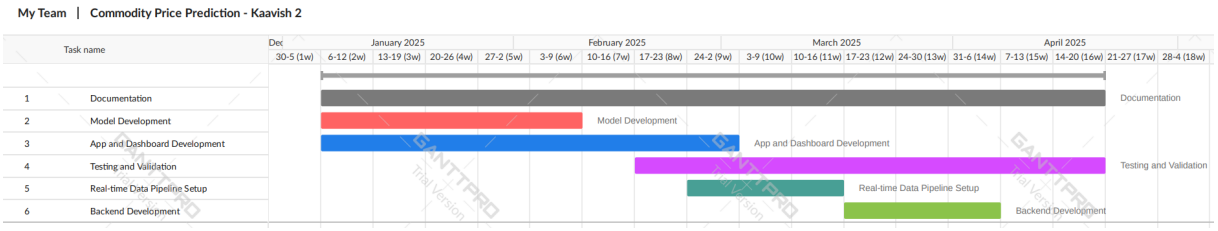

Figure 2: Project Plan for Kaavish 2

2.4 Resources

- **Compute Resources**
  - **Cloud Computing Services:** Platforms like AWS or Google Cloud will be necessary for scalable storage and processing of large datasets. This infrastructure will support real-time data integration and model training.

  Access: Cloud services will be selected based on budget and computational power requirements for model training and deployment.

- **Hardware**
  - **Development Machines:** Team members will need personal computers with adequate processing power (preferably with GPUs) to handle data processing and model training tasks effectively.

- **Testing Devices:** Various devices, including laptops, desktops, and tablets, will be required for testing the web app across different browsers, platforms, and screen sizes.

Access: Team members will utilize their own devices.

- **Software**
  - **Machine Learning Libraries:** Python libraries such as Pandas, NumPy, Scikit-learn, TensorFlow/Keras will be essential for data manipulation, machine learning model development, and neural network implementation.
  - **Data Visualization Tools:** Software like PowerBI or Tableau will be required for creating interactive visualizations that present both historical data and future predictions.
  - **Frontend Development:** React JS, Tailwind CSS, and Redux will be used to develop a user-friendly interface that works seamlessly on the web.
  - **NLP Tools:** Natural Language Processing libraries (e.g., NLTK or SpaCy) will be necessary for analyzing unstructured data sources to enhance prediction capabilities.

  Access: Most software libraries are open-source or available through institutional licenses, making them accessible for the project team.

- **Human Resources:** Our team combines diverse skills and experiences for flexible collaboration. Each member contributes uniquely while working towards common goals.
  - **Owais Aijaz:** Has a strong background in AI, data science, and machine learning, bringing both academic knowledge and practical experience. He plays a key role in experimenting with and optimizing the machine learning model while managing project timelines and refining his frontend skills.
  - **Syed Hamza:** Focuses on developing machine learning models and data analysis, using his knowledge of probability, statistics, and neural networks to ensure accurate predictions. He manages dataset cleaning and feature selection while also looking to improve his frontend skills.
  - **Aiza Imran:** Specializes in Web Development, Data Science, OOP, DSA, and DBMS. She designs and develops the user interface for a smooth

experience and is eager to expand into backend prediction systems to enhance her skills and contribute to the machine learning model through current courses such as Deep Learning and Natural Language Processing.

- **Sidra Aamir:** Handles design, research, and management, ensuring the final product is functional and user-friendly. Her expertise in UI/UX design, deep learning, and natural language processing strengthens the project, and her project management skills facilitate clear communication and coordination among team members.

2.5 Risks

Following are some potential risks that our team may face when developing this project:

- **External Factors Complexity:** Commodity prices are influenced by numerous unpredictable and highly non-linear external factors, such as sudden weather changes, international trade policies, and economic shifts.
  - **Challenge:** The model may struggle to capture the effects of all relevant variables, especially during unexpected events, which could lead to less accurate predictions for certain commodities.
  - **Mitigation:** Utilize advanced machine learning techniques that can adapt to non-linear relationships and continuously monitor external factors, updating the model to reflect real-time changes when possible.
- **Data Availability and Quality:** Inconsistent or missing data could significantly hinder model accuracy, especially if key historical or real-time data points are unreliable or unavailable.
  - **Challenge:** Limited or low-quality data may lead to gaps in training, making predictions less robust across different conditions.
  - **Mitigation:** Secure reliable data sources, implement data validation checks, and use data imputation techniques to handle missing values. Regularly review and update data sources to ensure quality and consistency.
- **Overfitting:** There is a risk of the model becoming too tailored to training data, capturing noise rather than underlying patterns.
  - **Challenge:** Overfitting can lead to reduced accuracy when the model is exposed to new or unseen data, impacting its reliability for practical use.

  - **Mitigation:** Employ techniques such as cross-validation, regularization, and pruning to ensure the model generalizes well. Regularly evaluate the model's performance on out-of-sample data.

- **Real-Time Data Integration Challenges:** Incorporating real-time data, such as weather updates or market trends, may present technical challenges and could impact prediction speed.
  - **Challenge:** Real-time data streams may have variable quality or latency, which could affect the responsiveness of the model's predictions.
  - **Mitigation:** Develop efficient data pipelines that can handle real-time integration smoothly, with buffer systems to manage latency issues. Use API rate limits and caching mechanisms to reduce delays.

- **User Adoption:** Stakeholders, including traders and policymakers, might be hesitant to adopt new technology due to lack of familiarity or trust.
  - **Challenge:** Limited user adoption could reduce the model's impact and limit data-driven decision-making across the sector.
  - **Mitigation:** Conduct user training sessions, simplify the user interface, and provide clear documentation. Gather feedback to improve usability, build trust, and highlight the benefits of data-driven insights for decision-making.

3 Functional Requirements

1. **User Authentication & Management**
   - The system should allow users to securely sign up and log in to access the web application, with the option to change/remove the given information.
   - The system should authenticate users and validate the information entered.
2. **Guided Tour or Walkthrough**
   - Upon first login, the system should provide a guided tour to help users understand core functions, including price comparison, crop selection, and setting up alerts. This is helpful for the kind of audience we are targeting.

3. **Search Filters**

    - Users should be able to select specific regions and crops, which will filter predictions and historical data based on their choices.
    - The system should organize crops by season (Rabi/Kharif), allowing users to view relevant data and predictions specific to each season.

4. **Market Insight Comparison**

    - The system should allow users to compare prices for selected crops across multiple regions, helping them identify the best market opportunities for buying or selling.

5. **Daily Price Alerts and Push Notifications**

    - Users should be able to set up customized daily alerts and push notifications to alert users of significant price changes or predictions, enabling quick, informed decisions.

6. **Interactive Dashboard**

    - The system should feature an interactive dashboard where users can explore predictions, trends, and comparisons through clickable data points and filters.

7. **Market Commentary and External Factors**

    - The system should provide real-time commentary on market trends, weather impacts, and other external factors influencing prices, sourced from relevant data feeds in a sliding banner fashion.
    - The system should integrate context from local weather conditions, economic factors, and international news to help users understand broader market impacts on crop prices.

8. **Text-Based Summaries**

    - The system should generate reports that summarize trend insights and key data points from the dashboard.
    - Users should have the option to view these reports in both Urdu and English to accommodate language preferences.

9. **Multilingual Support**

    - The system should provide the whole interface, reports, and FAQs in both Urdu and English, making the web app accessible to non-English-speaking users.

10. **FAQ Section**

    - The system should include a multilingual FAQ section addressing common questions about navigation, setting alerts, and understanding predictions, to aid user onboarding and troubleshooting.

11. **User Feedback Mechanism**

    - The system should incorporate a feedback form to allow users to submit suggestions or report issues, supporting ongoing improvements.

3.1 Use Case Diagram and User Interfaces

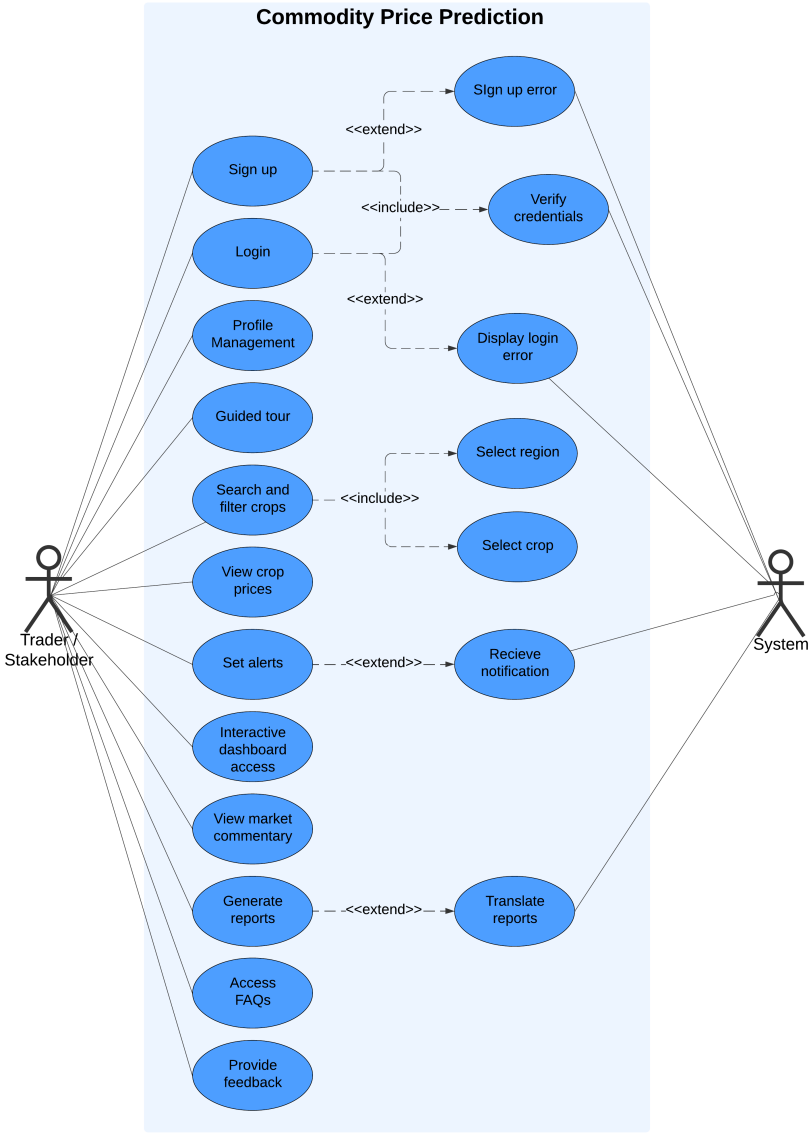

Figure 3: Use Case Diagram

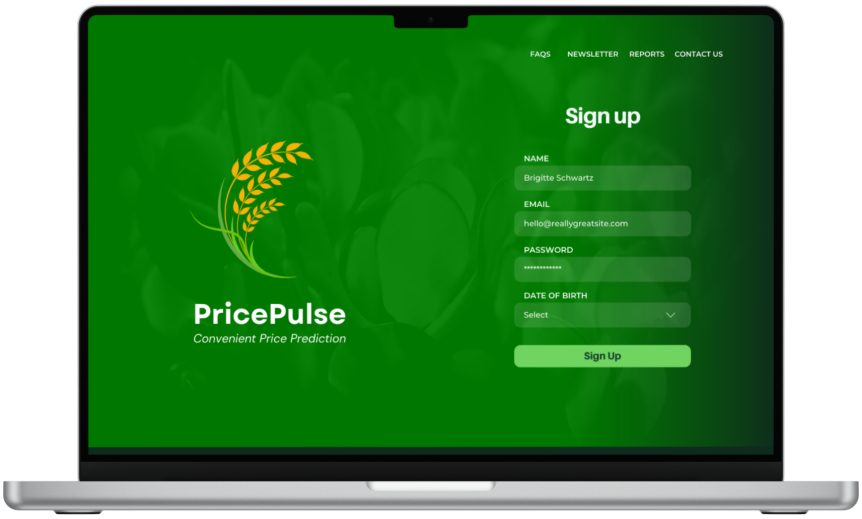

Figure 4: User Sign Up Screen

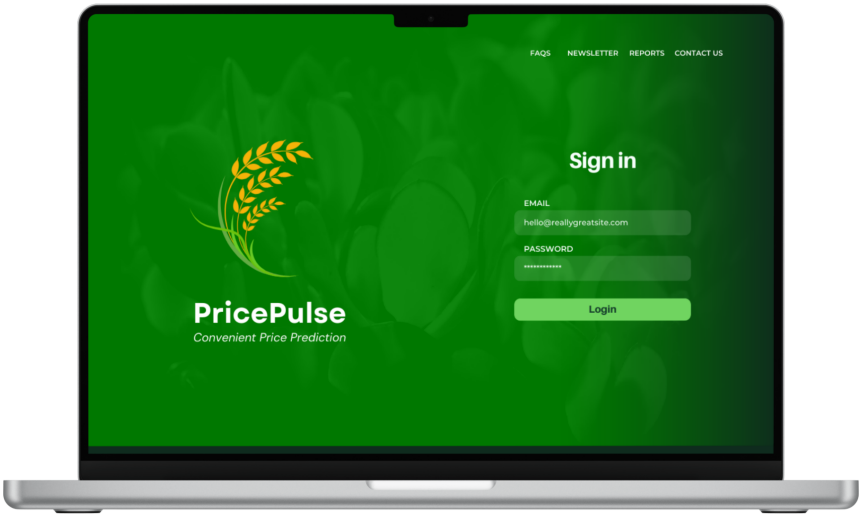

Figure 5: User Sign In Screen

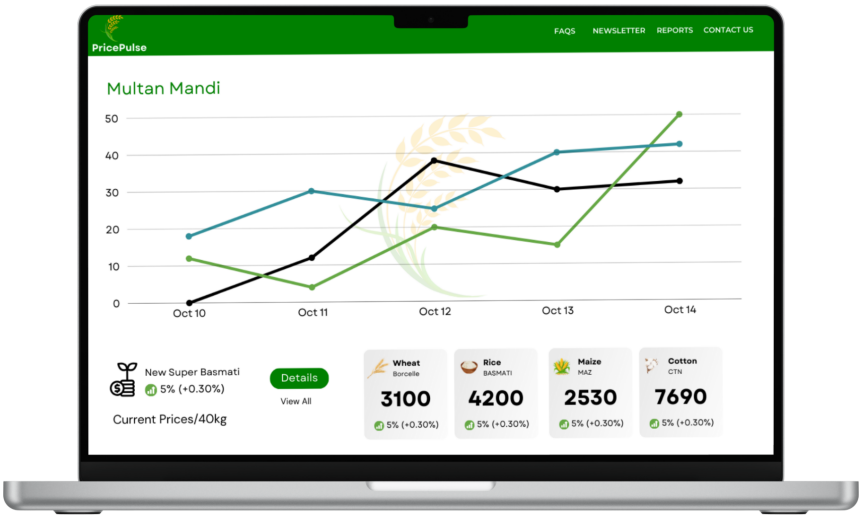

Figure 6: View Dashboard Screen 1

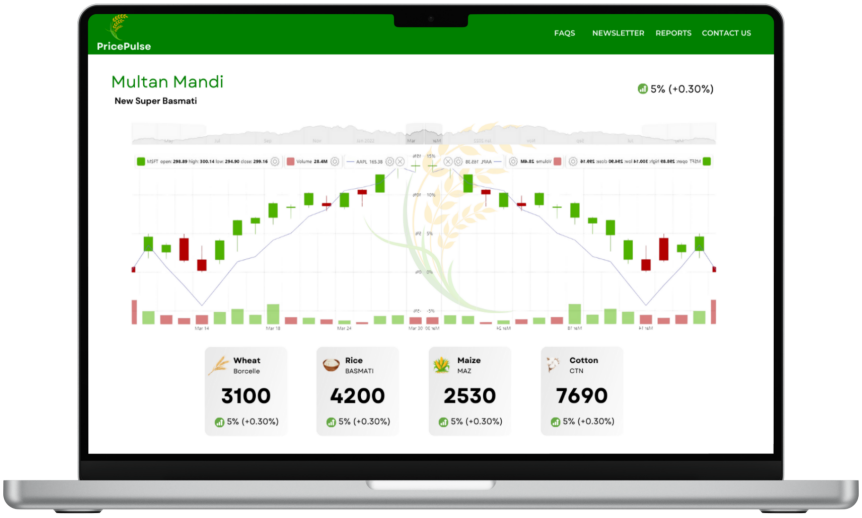

Figure 7: View Dashboard Screen 2

4 Non-functional Requirements

1. **Performance**

   - Efficient Data Processing: The system should handle large datasets efficiently, optimizing data retrieval and prediction time for real-time or near real-time analysis.
   - Responsive Interface: Ensure minimal latency in data display and interaction, particularly for dashboard refreshes and data insights.
   - Caching Mechanisms: Implement caching for frequently accessed data to reduce load times and improve user experience.

2. **Security**

   - Data Encryption: Use end-to-end encryption (such as AES or RSA) for any sensitive data, ensuring secure storage and transfer of user and agricultural data.
   - User Authentication & Access Control: Implement secure login protocols (e.g., OAuth) and role-based access control, limiting data access based on user roles.
   - Regular Vulnerability Testing: Conduct periodic security assessments and penetration testing to proactively address potential vulnerabilities.
   - Data Privacy Compliance: Ensure compliance with local and GDPR regulations by implementing user consent mechanisms for data collection and storage.

3. **Reliability**

   - High Availability: Ensure system uptime and availability through server redundancy and failover mechanisms, so users can rely on uninterrupted access.
   - Data Backup and Recovery: Establish regular data backups and recovery procedures to prevent data loss in case of failures.
   - Error Monitoring and Alerts: Implement automated monitoring with alerts for system errors and potential failures, enabling quick response to issues.
   - Consistent Data Refresh: Ensure timely updates of historical data and weather/economic inputs to maintain prediction accuracy and reliability.

4. **Scalability**

    - Cloud-Based Scalability: Leverage cloud infrastructure for scalability, allowing the system to dynamically adjust resources in response to demand.
    - Load Distribution: Implement load balancing to distribute traffic efficiently across servers, ensuring smooth operation even during peak loads.
    - Database Optimization: Utilize indexing and partitioning strategies to manage the storage and retrieval of large datasets efficiently as usage grows.

5. **Usability**

    - User-Friendly Interface: Design a clean, intuitive interface with clear navigation and easy access to prediction insights, charts, and data filtering options.
    - Multilingual Support: Offer the platform in both Urdu and English to accommodate diverse user preferences and increase accessibility in Pakistan.
    - Responsive Design: Ensure the web app interface is fully responsive and optimized for various screen sizes, delivering a seamless experience across different devices.
    - Interactive Tutorials and Help Section: Include a help section with FAQs, user guides, and step-by-step tutorials, along with a guided tour on first-time login to improve user onboarding.
    - Feedback System: Provide an in-app feedback mechanism to gather user input, allowing them to report issues and suggest improvements.

6. **Compliance**

    - GDPR and Local Data Protection Laws: Adhere to GDPR and local regulations, ensuring user data is stored with consent.
    - PAR Data Compliance: Comply with licensing requirements by PAR for agricultural data distribution and display.
    - Intellectual Property: Use licensed or open-source tools, APIs, and data sources with proper attribution.
    - Data Encryption: Implement end-to-end encryption for sensitive user data.

- Regular Security Audits: Conduct periodic security audits to identify vulnerabilities.
- Privacy Policy: Display a clear, concise privacy policy explaining data collection and usage.

5 Model Specifications

5.1 Feature Selection

- **Weather:** Includes temperature, humidity, air quality, and water content, essential for crop growth and price fluctuation analysis.
- **Energy Costs:** Key energy inputs such as electricity and fuel costs that impact production and transportation expenses.
- **Supply & Demand:** Measures demand trends and supply levels, reflecting scarcity or abundance and influencing price volatility.
- **Crude Oil Prices:** Includes historical and current crude oil prices, affecting transportation and input costs.
- **Economic Conditions:** Encompasses market trends, tariffs, inflation rates, and trade policies, providing context for macroeconomic influences on prices.

5.2 Algorithm Selection

- **Machine Learning Approaches:**
  - **Traditional Forecasting Methods:** Use established techniques like linear regression and polynomial regression for baseline predictions.
  - **Intelligent Forecasting:** Leverages AI-driven methods (ANNs, SVMs, and LSTMs) for improved accuracy in dynamic environments.
  - **Hybrid Forecasting:** Combines traditional and AI approaches, optimizing performance through ensemble methods and model stacking.
- **Statistical and Machine Learning Techniques:**
  - **Regression Analysis:** Provides a basic statistical foundation for identifying relationships between factors and predicting outcomes.

- **Time Series Analysis:** Moving Average and Exponential Smoothing for trend identification and short-term forecasting.
- **Artificial Neural Networks (ANNs):** Effective for non-linear, complex relationships and patterns within the dataset.
- **Support Vector Machines (SVMs):** Useful for classification and regression in high-dimensional feature spaces.
- **Recurrent Neural Networks (RNNs):** Employ Time-Delay Neural Networks (TDNN), Long Short-Term Memory (LSTM) models, and RNNs for temporal dependencies, especially for multi-seasonal and sequence-based data.
- **Ensemble Learning:** Use methods like Random Forests or Gradient Boosted Trees to combine predictions and reduce model variance, enhancing robustness.

6 Data Requirements

6.1 Historical Data

- **Weather Data:** Collect historical records of temperature, humidity, air quality index, and precipitation from local and global sources.
- **Energy Cost Data:** Data on fuel and electricity rates from relevant authorities or energy departments over time.
- **Supply & Demand Metrics:** Historical supply-demand figures, including seasonal trends, crop yield, and consumption statistics.
- **Crude Oil Price Data:** Historical oil price trends from economic data repositories, showing global or regional impact.
- **Economic Indicators:** Macro indicators like inflation rates, GDP, tariffs, and market indices to assess economic influences on price dynamics.

6.2 External Data Sources

- **Real-Time Weather Feeds:** Integrate weather APIs (e.g., OpenWeatherMap or regional meteorological data) for up-to-date environmental data.

- **Commodity Market Data:** Current and historical data from local markets and government trade bodies.
- **News and Reports:** Utilize NLP to gather economic insights from news sources or financial reports, providing a qualitative layer to the model.
- **Pakistan Meteorological Department Reports:** Utilize NLP to extract key indicators' data from the PMD.
- **Exchange Rates:** Get real-time currency exchange rates using free APIs such as ExchangeRate, to include as a feature to the model.

7 System Diagram

- **Front-end UI:** Built with React JS, styled using Tailwind CSS, and managed with Redux. Handles user interactions and sends API requests to the backend.
- **Interface Layer:** Manages API requests, user authentication, and data handling (GET and POST requests), ensuring smooth communication between the front end and backend services.
- **Prediction Layer:** Validates and processes data through real-time pipelines (AWS Kinesis, Google Pub/Sub), running machine learning models (LSTMs, Classical ML, RNNs) for predictions.
- **Data Layer:** Gathers real-time and historical data (exchange rates, weather, market data) and preprocesses it using NLP modules. Integrates historical crop price datasets.
- **Database:** Stores unstructured data in MongoDB, structured data in PostgreSQL, and trained models in a model registry.
- **Operating Environment:** Hosted on cloud platforms (AWS/Google Cloud) with dedicated server options and third-party API integration.

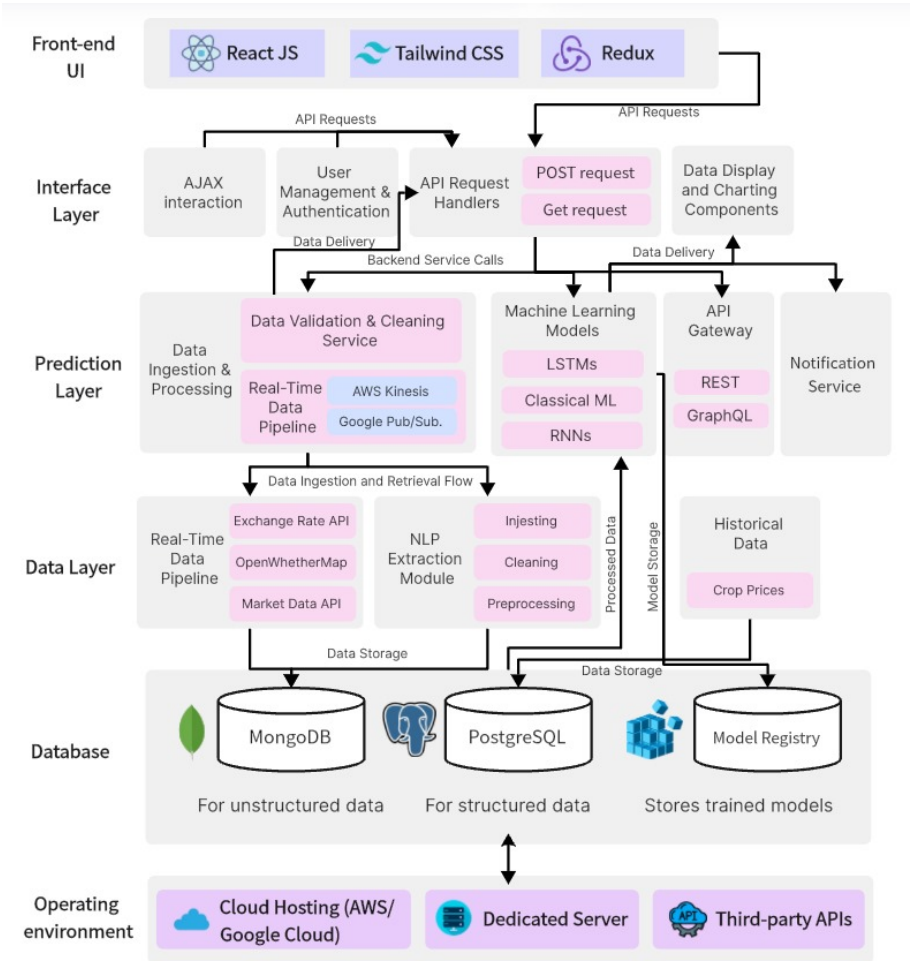

Figure 8: System Block Diagram

8 Conclusion

In conclusion, this Software Requirements Specification (SRS) document outlines a comprehensive approach for developing a commodity price prediction system aimed at improving decision-making and financial planning for stakeholders in Pakistan's agricultural sector. By integrating a wide range of relevant features—such as weather conditions, economic indicators, and supply-demand dynamics—and leveraging machine learning and statistical models, the system aims to provide accurate and timely price forecasts. This SRS serves as a foundational blueprint to guide the design, development, and deployment of an efficient, reliable, and accessible prediction system.