SUPPLY2U PROJECT WRITE UP.

Problem Statement and Motivation.

Globally and especially in East Africa, agricultural supply chains, small-scale farmers, agro-dealers, and urban retailers face significant challenges in ensuring food products' efficient, sustainable distribution.
Consumers increasingly demand to know the origins of their food, and retailers require precise information to manage inventory and meet consumer demand. Inefficiencies in the agricultural supply chain lead to high post-harvest losses, inconsistent quality and availability of produce, and inflated prices due to multiple layers of intermediaries.
These issues compromise farming's economic viability, restrict consumer access to fresh produce, and hinder fair value distribution. Despite growing demands for transparency and efficiency, existing solutions need more comprehensive integration for all stakeholders, including farmers, agro-dealers, retailers, and logistics providers.
There is a critical need for a transformative platform that seamlessly connects these stakeholders, prioritizing efficiency, sustainability, and profitability throughout the supply chain.

Below, we try to enlist concrete examples to motivate the need for a transformative platform that addresses these inefficiencies;

1. **High Post-Harvest Losses:**

Many farmers in East Africa face significant losses after harvest due to inadequate storage, processing facilities, and poor logistics. This results in a substantial percentage of crops never reaching the market.
As highlighted by a study in the Agriculture & Food Security journal, extensive post-harvest losses are experienced in Ethiopia, where an average of 25.81% of all crops are lost annually, with fruits and vegetables experiencing the highest losses at 33.38%. This reflects a broader regional issue, as a World Bank report notes similar challenges across Sub-Saharan Africa,

2. **Inconsistent Quality and Availability:**

The quality and availability of agricultural products can vary dramatically, which affects pricing and consumer satisfaction. This inconsistency is often due to uncoordinated supply chains and varying standards among small-scale producers.
As highlighted by The World Bank in Uganda and Kenya, up to 40% of tomatoes and other perishable produce are lost due to inefficiencies in transport and inadequate storage, leading to spoilage before reaching consumers. These losses not only signify a gap in the supply chain but also result in fluctuating quality and availability that impacts consumer satisfaction and retailer reliability

A retailer in Nairobi may receive high-quality avocados one week and very poor-quality ones the next, making it difficult to maintain customer loyalty and manage inventory effectively.

3. **Lack of Transparency and traceability**:

Consumers globally and in East Africa are increasingly demanding transparency in the origins and handling of their food. Lack of such information can affect consumer trust and decision-making.
Consumers in urban areas like Kampala are starting to prefer organically grown vegetables but have no way to verify the claims of retailers, leading to skepticism and reduced sales.

SOLUTION ARCHITECTURE.

At Supply2U, we're revolutionizing how agricultural supply chains operate from farm to fork. Our innovative platform seamlessly integrates geolocation data of the farms, real-time analytics, and consumer behavior insights to empower stakeholders at every step of the supply chain. We aim to focus on efficiency, sustainability, and profitability, transforming how stakeholders connect and thrive in a dynamic market through a holistic, data-driven, innovative approach.

The platform will have multiple components, including data collection, processing, analytics, user interface, and integration layers. In the future, the solution will leverage cloud infrastructure for scalability and reliability.

High-Level Architecture

- ➔ **Data Collection Layer**:
  - ➔ ***Geolocation Data Collection**:* Devices/sensors on farms, vehicles, and processing facilities.
  - ➔ ***Consumer Behavior Data Collection**:* Implementing web app analytics will help gather valuable insights into consumer preferences and purchasing patterns, critical for tailoring marketing strategies and improving product offerings.
- ➔ **Data Processing and Storage Layer**:
  - ◆ ***Data Ingestion***: Stream and batch data ingestion pipelines.
  - ◆ ***Data Storage**:* Centralized data warehouse for structured and unstructured data.
- ➔ **Analytics and Insights Layer**:
  - ◆ ***Real-Time Analytics**:* Tools for real-time data processing and analytics.
  - ◆ ***Machine Learning Models**:* Predictive analytics and consumer behavior insights.
- ➔ **User Interface Layer**:
  - ◆ ***Web Platform**:* React.js for the front end.
- ➔ **Integration and APIs Layer**:
  - ◆ ***APIs**:* Django REST Framework for building APIs.
  - ◆ ***Middleware**:* For seamless data exchange between components.

Technology Stack

Data Collection:

- **IoT Devices**: GPS devices, sensors on vehicles and farms.

Data Ingestion:

- **Apache Kafka**: For real-time data streaming.
- **Apache Nifi**: For data flow automation and ETL processes.

Data Storage:

- **Hadoop HDFS**: For distributed file storage.
- **PostgreSQL**: For structured data storage, integrated with Django.
- **Cassandra**: For high-throughput NoSQL storage.
- **Apache Hive**: For data warehousing and analytics.

Data Processing and Analytics:

- **Apache Spark**: For big data processing.
- **Druid**: For real-time analytics.
- **TensorFlow**: For building and deploying machine learning models.

Front-End:

- **React.js**: For building responsive web applications.

Back-End:

- **Django**: For building the server-side web application.
- **Django REST Framework**: For building RESTful APIs.

Integration and Middleware:

- **GraphQL**: For efficient data querying and aggregation.
- **Kong**: For API management.

Security:

- **OAuth2**: For user authentication and authorization.
- **Encryption**: SSL/TLS for data in transit, GPG for data at rest.

Architecture Implementation

Data Collection Layer:

- **Farm Data Collection**: GPS devices installed on farm equipment send data to a local server. Field boundaries and crop areas are mapped using GIS tools.
- **Processing Facilities**: GPS devices track the location of processing facilities. IoT sensors monitor operational metrics.
- **Transportation**: GPS data from transportation vehicles is sent in real-time to track routes, shipping times, and delays.

Data Processing and Storage:

- **Ingestion Pipelines**: Data is ingested through Kafka topics for real-time data and managed by Nifi for ETL processes. Raw data is stored in HDFS. Processed data is moved to PostgreSQL for relational storage and Hive for analytics. Cassandra is used for high-speed data access.

Analytics and Insights:

- **Real-Time Analytics**: Druid and Spark Streaming process data in real-time, providing immediate insights into transport routes and processing facility performance.
- **Machine Learning**: TensorFlow trains models on historical data to predict crop yields, optimize transport routes, and analyze consumer behavior.

User Interface:

- **Web Platform**: Developed with Django (backend) and React.js (frontend), the web platform provides dashboards for distributors. Real-time maps display the current locations of shipments.

Integration:

- **APIs**: RESTful APIs built with Django REST Framework and managed via Kong API Gateway allow third-party services to integrate with the platform.
- **Middleware**: GraphQL server aggregates data from multiple sources, providing a unified API for front-end applications.

Scalability and Performance Considerations

- **Auto-scaling**: Use Kubernetes for container orchestration and scaling.
- **Caching**: Implement caching with Redis to reduce latency.
- **Load Balancing**: Use HAProxy or NGINX for load balancing.

Security Considerations

- **Authentication and Authorization**: Implement OAuth2 with Django's built-in authentication system.
- **Data Encryption**: Encrypt data in transit using SSL/TLS, and at rest using GPG.
- **Compliance**: Ensure compliance with relevant agricultural and data protection regulations.

Monitoring and Maintenance

- **Monitoring**: Use Prometheus to monitor system performance and set up alerts.
- **Logging**: Centralized logging with ELK Stack (Elasticsearch, Logstash, Kibana).
- **Backup and Recovery**: Regular backups using open-source tools like Bacula, with disaster recovery plans in place.

Some of the above technologies and measures are subject to change as we are still in the design phase of the architecture.

SOLUTION ARCHITECTURE EVALUATION.

Example 1: Real-Time Monitoring of Crop Harvesting

Scenario: The distributor needs to monitor crop harvesting areas in real time to ensure timely harvesting and transportation to processing facilities.

Solution:

1. **Data Collection**: GPS devices installed on farm equipment collect real-time location data.
2. **Data Ingestion**: Data is streamed to the platform using Apache Kafka.
3. **Data Processing**: Real-time data is processed using Apache Spark and stored in PostgreSQL for structured queries.
4. **User Interface**: A Django web application with a React.js frontend provides a dashboard displaying real-time maps and metrics.

Evaluation:

- **Experiment**: Set up GPS devices on a few pieces of farm equipment. Stream this data into the system, and visualize it on the dashboard.
- **Outcome**: The system should display the real-time locations of the equipment on the map, showing the progress of the harvesting process. Delays or issues can be quickly identified and addressed.

Example 2: Tracking Transportation and Managing Delays

Scenario: Transportation routes need to be monitored to ensure timely delivery to processing facilities and consumers. Any delays need to be identified and managed efficiently.

Solution:

1. **Data Collection**: GPS data from vehicles is collected in real-time.

2. **Data Ingestion**: Data is ingested via Kafka and managed by Apache Nifi for ETL processes.
3. **Real-Time Analytics**: Druid processes this data to provide real-time analytics on transport routes and delays.
4. **User Interface**: The dashboard shows current vehicle locations, estimated delivery times, and alerts for any delays.

Evaluation:

- **Experiment**: Deploy GPS devices on vehicles, ingest and process the data, and visualize it on the dashboard.
- **Outcome**: The dashboard should display the real-time locations and statuses of the vehicles. If a vehicle is delayed, an alert should be generated, allowing managers to take corrective actions.

Example 3: Consumer Behavior Insights

Scenario: Understanding consumer behavior to tailor marketing strategies and improve product offerings.

Solution:

1. **Data Collection**: User behavior data is collected from the web application.
2. **Data Ingestion**: Data is ingested and processed using Apache Nifi.
3. **Data Analysis**: Machine learning models analyze consumer behavior patterns.
4. **User Interface**: Insights are displayed on a dashboard for marketing teams to access.

Evaluation:

- **Experiment**: Track distributor interactions on the web application over a period. Analyze the data to identify patterns and trends.
- **Outcome**: The dashboard should provide actionable insights, such as popular products, peak usage times, and user demographics. Marketing strategies can be adjusted based on these insights to improve engagement and sales.

This solution leverages open-source technologies to build a comprehensive platform for transforming agricultural supply chains. By using self-hosted infrastructure and open-source tools, the solution ensures flexibility, control, and cost-effectiveness while providing robust data processing, analytics, and integration capabilities.

PROJECT CODE LINKS.

Portfolio website: GITHUB PORTFOLIO WEBSITE CODE.
Frontend GitHub repo: FRONT END GITHUB REPO
Backend GitHub repo: BACK END GITHUB REPO