**Contributions of Individuals:**

Faizan = Airflow DAGs, Kafka, Spark

Salman = Scripts orchestrations, ML Model, ML pipeline

Ahsan = ML pipeline, ML model, Deploying on Hugging Face

Zafir = S3, Postgres setup (DB).

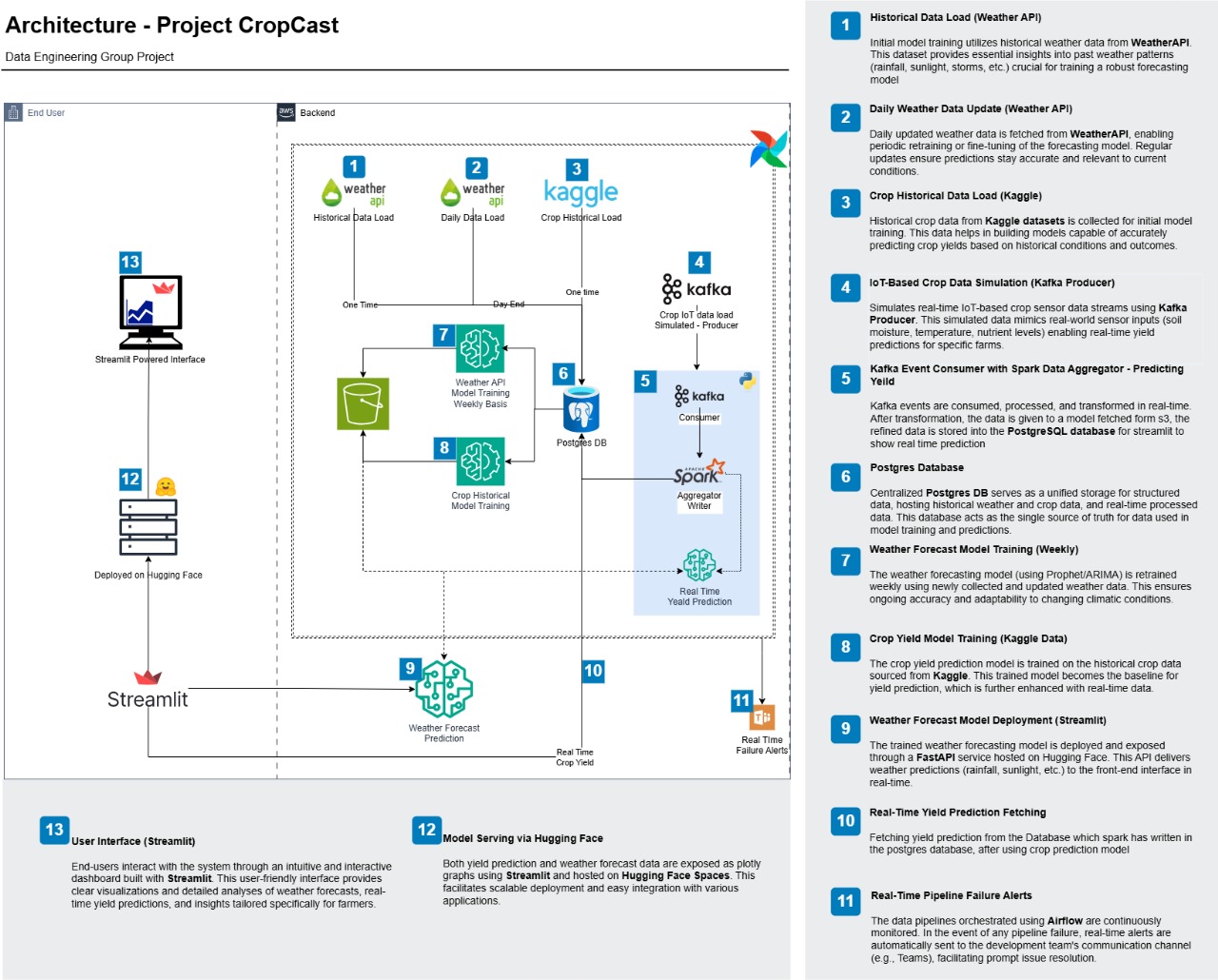
**Brief description of your project title, goal, dataset, and domain/theme. 1-2 paragraphs.**

**SmartCrop: Weather Forecasting and Yield Prediction System**

This project aims to build a complete pipeline that supports farmers by predicting future weather conditions and estimating crop yield based on multiple input factors. Using historical weather data and time series models like Prophet, the system forecasts 7-day rainfall, temperature, and wind speed. In parallel, a Kafka producer streams real-time feature values such as sunlight, rainfall, soil type, farm area, and fertilizer amount. These features are consumed and processed using Apache Spark and then fed into a yield prediction model trained on a Kaggle dataset.

The entire pipeline is orchestrated using Apache Airflow, which automates scheduled tasks and monitors failures. In case of any pipeline breakdown, Airflow DAGs are configured to send failure alerts via Microsoft Teams. The final outputs, including predictions and recommendations, are displayed through an interactive Streamlit dashboard deployed on Hugging Face Spaces. The project belongs to the agriculture and climate-tech domain, offering a scalable solution to help farmers, especially newcomers—make informed and data-driven decisions.

**Architecture Diagram (you can use draw.io for architecture and schema diagrams)**

****

**Schema Diagram**

**A screenshot of a computer

AI-generated content may be incorrect.**

**Brief overview of each stage of the data engineering lifecycle and rationale explaining your decision to use a particular tool. Please explicitly cover data collection/ingestion, data cleaning/transformation/quality, storage/modeling, pipeline orchestration, deployment & frontend.**

**Data Collection/Ingestion:**  
Data was collected through the Open-Meteo API for both model training and future predictions. The API was chosen due to its free access and lack of usage restrictions, making it ideal for academic experimentation. During ingestion, the data was validated for completeness, and missing values in partially empty records were imputed using mean values to ensure minimal data loss—especially important for training accurate machine learning models.

**Data Storage/Modeling:**  
PostgreSQL was used as the central storage solution to enable all team members to access the data remotely. Its cloud-based accessibility, security, and compatibility with Apache Spark made it a strong choice for handling large, structured datasets.

**Machine Learning Models:**  
The machine learning models were trained and stored on AWS S3. Lightweight models such as Facebook Prophet (for time series weather forecasting) and a two-layer dense neural network (for yield prediction) were selected to minimize latency and ensure the system could provide near real-time responses without performance bottlenecks.

**Streaming and Real-Time Ingestion:**  
Simulated sensor data from a farm was streamed using Apache Kafka. Kafka was selected for its high-throughput, fault-tolerant design, and ability to handle real-time data from multiple sensors. Although we simulated one farm’s data, the architecture is scalable to accommodate real-world, multi-farm sensor networks.

**Pipeline Orchestration:**  
Apache Airflow was employed to orchestrate and automate the pipeline, ensuring that different modules (e.g., data ingestion, transformation, model prediction) run at scheduled intervals without manual intervention. Airflow also monitors the health of the pipeline and sends alerts via Microsoft Teams in case of failures, ensuring reliability and responsiveness.

**Deployment & Frontend:**  
Visualizations and user interactions were implemented using Streamlit and deployed on Hugging Face Spaces, making the system easily accessible from any device with a web browser. This cloud-based deployment minimizes the need for local computational resources while providing a lightweight and intuitive interface—particularly beneficial for farmers and other stakeholders. The application features two main tabs: one displays forecasted weather variables such as rainfall, wind and temperature, while the other predicts crop yield based on simulated sensor data (including sunlight, soil type, farm area, and fertilizer usage) streamed through Kafka.

**Link to deployed demo/interface**

https://huggingface.co/spaces/AI71/DE2

**GitHub link to your repo/folder.**

<https://github.com/FaizanSh/cropcast-de-gp>

**Use of AI:**

AI, specifically Large Language Models (LLMs), were actively utilized throughout the development of this project. During coding, we encountered challenges across multiple platforms—including Airflow, Kafka, PostgreSQL, and Hugging Face Spaces—for which we consulted LLMs to receive guidance, debug errors, and understand best practices. This support significantly streamlined our development process, especially on unfamiliar tools.

Furthermore, the documentation—including what you are currently reading—was initially drafted by the team and then paraphrased and refined using an LLM to enhance clarity, coherence, and readability. All AI-generated content was reviewed and validated by the team to ensure technical accuracy and project alignment. This approach allowed us to maintain high-quality documentation while accelerating our workflow.