Databases II

Workshop No. 2 — Data System Architecture and Information Retrieval

AgroClima: A Smart Agro-Climatic Decision Support Platform

César Andrés Torres Bernal 20191020147 Juan David Duarte Ruiz 20191020159

Universidad Distrital Francisco José de Caldas
School of Engineering

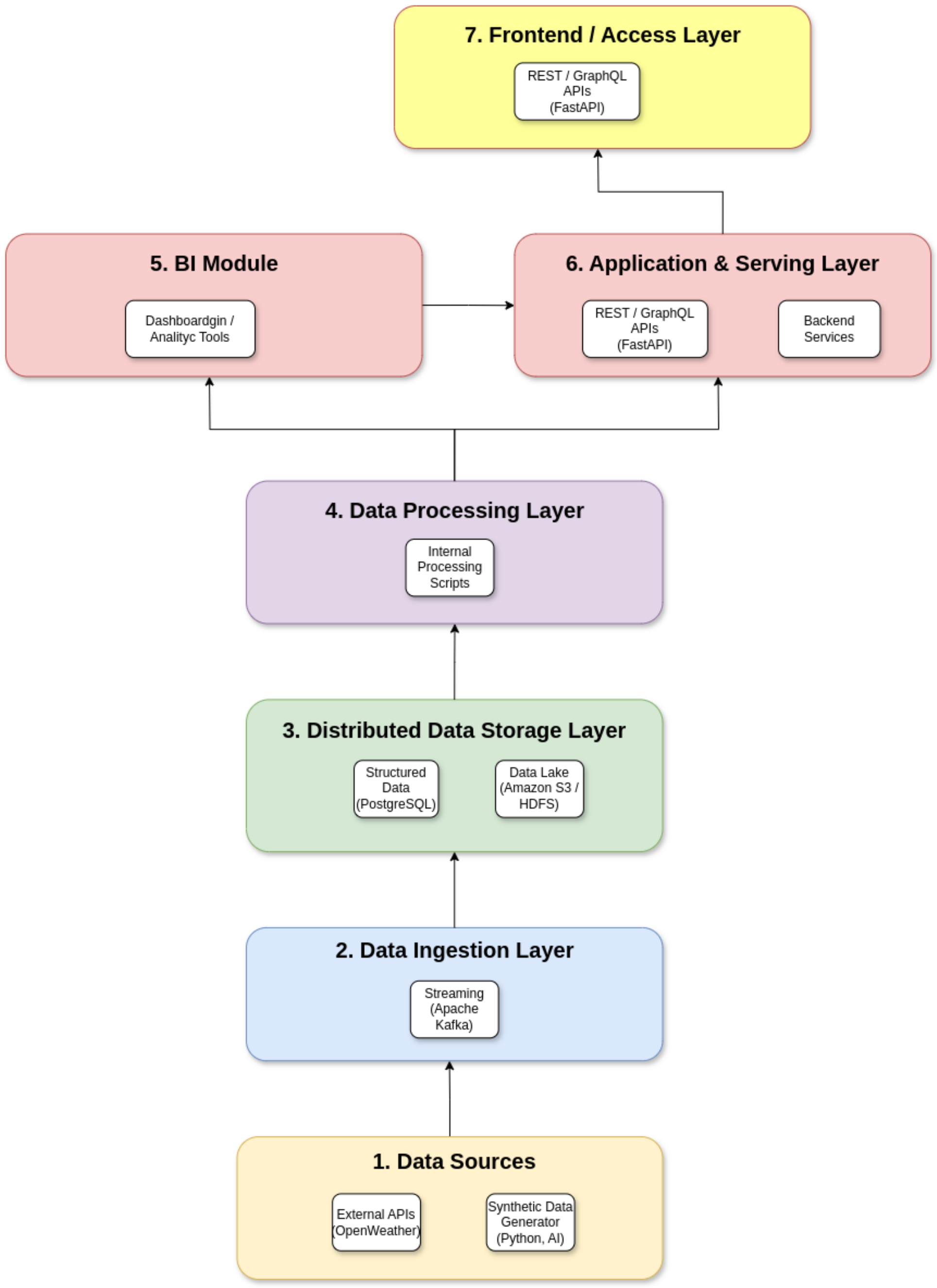
Data System Architecture

The AgroClima platform is designed to operate as a scalable, data-driven system capable of ingesting, processing, storing, and delivering actionable climate intelligence to agricultural stakeholders. Given the system's reliance on large volumes of external data and personalized recommendations, a robust and modular data system architecture is essential.

This architecture follows a layered design pattern that supports continuous data ingestion from external APIs and simulated sources, distributed storage of structured and unstructured data, real-time and batch data processing, and secure delivery of insights through APIs and user interfaces. The architecture also integrates a Business Intelligence (BI) layer to enable data visualization and strategic decision-making. Each component in the system plays a specialized role—from collecting and validating weather data, to executing climate risk models, to generating field-specific recommendations. The architecture ensures high availability, scalability, and modularity to support a wide range of user roles, from farmers in the field to analysts and system administrators.

The following section shows the high-level architecture diagram and describes each architectural layer, its key components, the technologies employed, and how data flows between them to enable real-time decision support and long-term agricultural planning.

High-level Architecture Diagram



7. Frontend / Access Layer
REST / GraphQL APIs (FastAPI)
5. BI Module
Dashboardgin / Analityc Tools
6. Application & Serving Layer
REST / GraphQL APIs (FastAPI)
Backend Services
4. Data Processing Layer
Internal Processing Scripts
3. Distributed Data Storage Layer
Structured Data (PostgreSQL)
Data Lake (Amazon S3 / HDFS)
2. Data Ingestion Layer
Streaming (Apache Kafka)
1. Data Sources
External APIs (OpenWeather)
Synthetic Data Generator (Python, AI)

Figure 1: High-level Architecture Diagram

1. Data Sources Layer

Role: This layer serves as the entry point for all raw data entering the system. It is responsible for retrieving weather, climate, and agricultural data, both from external sources and synthetic generators.

Components & Technologies:

- **External APIs:** `OpenWeather`, `OpenMeteo`, `NOAA` – provide real-time and historical weather and climate data.
- **Synthetic Data Generator:** Python scripts using libraries such as `Faker`, `NumPy`, or `Pandas` – simulate agricultural data in academic or testing scenarios.

Interaction: These sources feed data directly into the ingestion layer through scheduled or real-time calls.

2. Data Ingestion Layer

Role: Responsible for acquiring, normalizing, validating, and routing incoming data from all sources to the system's internal data stores.

Components & Technologies:

- **Apache Kafka**: For high-throughput, real-time stream ingestion where data events are published and consumed asynchronously.

Interaction: Data flows from external APIs or synthetic data generators $\rightarrow$ is collected, processed and is then stored directly in PostgreSQL (for structured data) and MongoDB (for unstructured or semi-structured data). Kafka serves to decouple real-time data streams from downstream processing and storage components.

3. Data Storage Layer

Role: This layer stores both raw and refined data, allowing persistence and long-term access. It includes both relational and `NoSQL` solutions.

Components & Technologies:

- **PostgreSQL:** Stores structured relational data: users, crops, alerts, predictions, sessions. Chosen for its `ACID` compliance, query performance, and relational integrity.
- **Data Lake** (Optional): `Amazon S3` or local `HDFS` bucket for storing raw bulk data (e.g., climate archives) if needed.

Interaction: Data from ingestion flows into `PostgreSQL` for relational entities. These stores are later accessed by the processing and application layers.

4. Data Processing & Intelligence Layer

Role: Transforms raw data into useful insights. Executes `ML` models, risk analysis, climate forecasting, and generates recommendations.

Components & Technologies:

- **Internal Processing Scripts:** Handle logic for alert generation, anomaly detection, and cleaning pipelines.

Interaction: Reads input data from `PostgreSQL` runs models and transformations, then writes back results to `PostgreSQL` (structured outputs like predictions).

5. Business Intelligence (BI) Module

Role: Provides data visualization and decision-making tools for analysts and administrators.

Components & Technologies:

- Dashboarding and analytics tools connected to `PostgreSQL`

Interaction: `BI` tools query `PostgreSQL` for analytics tables or materialized views and query `MongoDB` for custom visualizations or log-based dashboards.

6. Application & Serving Layer

Role: Serves processed data to external applications and end-users through interfaces and `APIs`. Implements business logic and manages user access.

Components & Technologies:

- **FastAPI** (Python): `REST/GraphQL APIs` expose data from both `PostgreSQL` and `MongoDB` securely.
- **Backend Services:** Handle user session management, request routing, access control, and processing `API` calls.

Interaction: `APIs` call `PostgreSQL` (for structured data like user profiles and prediction logs). `Redis` may cache common or urgent responses for performance.

7. Frontend / Access Layer

Role: This is the interface through which users interact with the system – whether via web dashboards or mobile devices.

Components & Technologies:

- **React** Responsive `UIs` for displaying predictions, recommendations, alerts, and reports.

Interaction: The frontend communicates with the `API` layer $\rightarrow$ receives processed data from `PostgreSQL` or `MongoDB` $\rightarrow$ renders information tailored to the user's location, role, and preferences.

Information Requirements

The system must be able to retrieve various types of information to support both business needs and user stories effectively. Below are the key types of information that the system must retrieve:

User Information and Role Management

- **Description:** The system must store and manage detailed information about each user, including personal identification data (name, email, password hash), role (e.g., farmer, analyst, administrator), associated agricultural fields, and user preferences (such as alert preferences or notification channels). The system must also associate users with system access controls to enforce role-based permissions.
- **Business Value:** User information is critical for delivering tailored experiences, restricting access to sensitive data, and segmenting recommendations, alerts, and dashboards. It also enables monitoring of system engagement and effectiveness at the individual or organizational level.
- **Stored In:**
  - **User:** stores primary user profile and role.
  - **Agriculture_Field:** linked to users to define field ownership.
  - **Alert:** linked to users for personalized climate notifications.
  - **Recommendation:** indirectly linked to the user via their associated fields.
  - **Auth_Session** (optional): to store login session and authentication data if implemented.
- **Related User Needs:**
  - Farmers receive field-specific alerts and recommendations.
  - Analysts access aggregated data and reports.
  - Admins manage user access, roles, and system activity.

Climate Risk Prediction Data

- **Description:** The system must store and retrieve data related to climate risk predictions, including event type (e.g., flood, drought, frost), location, probability, associated severity level, and prediction timestamp. Each prediction must be traceable to the machine learning model used for generation.
- **Business Value:** This information is core to the system's analytical capacity and value proposition, enabling proactive decision-making for users by anticipating adverse weather conditions.
- **Stored In:**
  - **Prediction** (risk event, probability, timestamp, location, model)
  - **Event_type**, **Severity_level**, and **Model** (supporting metadata)
- **Related User Needs:**
  - Farmers receive alerts and reports based on prediction outcomes.
  - Managers assess the likelihood of risks across regions they oversee.

Personalized Agricultural Recommendations

- **Description:** The system must store actionable recommendations linked to specific climate predictions and tailored to the crop type, field location, and user context. Each recommendation includes a textual description and reference to its originating prediction.
- **Business Value:** These recommendations transform insights into guidance, helping farmers plan irrigation, crop rotation, pest control, and other field-level actions to optimize outcomes.
- **Stored In:**

  - **Recommendation** (linked to Prediction)
  - **Agriculture_Field**, **Crop**, and **User** (to contextualize each recommendation)
- **Related User Needs:**
  - Farmers access field-specific suggestions.
  - Managers oversee aggregated recommendations for strategic coordination.

Real-Time and Historical Weather Data

- **Description:** The system must collect and manage granular weather data (temperature, humidity, wind speed, timestamp) for each weather station and location. This data should be both real-time and historically queryable.
- **Business Value:** Serves as the foundational input for generating accurate predictions and trend analysis. Enables historical reporting and supports scientific credibility of the system's outputs.
- **Stored In:**
  - **Weather_Data** (linked to Weather_Station and Location)
  - **Weather_Data_Source** (to track origin and ingestion timeline)
- **Related User Needs:**
  - Farmers adjust practices based on live updates.
  - Analysts use historical trends to improve forecasting models.

Agricultural Field and Crop Information

- **Description:** The system must maintain detailed records of users' agricultural fields, including crop type, field `ID`, and location coordinates. Each user can be associated with one or more fields.
- **Business Value:** Essential for personalizing predictions and recommendations. Field-specific characteristics help determine risk sensitivity and agricultural practices.
- **Stored In:**
  - **Agriculture_Field**, **Crop**, and **Location** (with foreign key relations to User)
- **Related User Needs:**
  - Farmers see data related to their own plots.
  - Farm managers analyze performance by field and crop type.

Alert and Notification Records

- **Description:** The system must store alert records that include which user received what alert, for which prediction, when it was sent, and the current status (e.g., delivered, read, acknowledged).
- **Business Value:** Ensures timely risk communication, allows users to track and act upon weather-related risks, and provides administrators with traceability for compliance and auditability.
- **Stored In:**
  - **Alert** (linked to Prediction and User)
- **Related User Needs:**
  - Farmers stay informed of imminent threats.
  - Admins verify alert coverage and effectiveness.

Raw API Response Storage

- **Description:** The system must optionally store raw `JSON` payloads from external weather `APIs` (such as `OpenWeather`) for auditing, reprocessing, and debugging purposes. These payloads may vary in structure depending on the source, `API` version, or requested endpoint.
- **Business Value:** Storing raw `API` responses ensures transparency, traceability, and reusability. It allows the system to:
  - Compare predictions based on past input
  - Debug failures in ingestion or modeling logic
- **Stored In:**
  - **MongoDB Collection:** `api_responses`

User Interaction Logs and Feedback

- **Description:** The system must store user interaction data and optional feedback related to recommendations, alerts, and platform usage. These logs include timestamps, device type, interaction type (e.g., viewed recommendation, dismissed alert, submitted feedback), and additional metadata like geolocation, response time, or comments. This information can vary in structure depending on the event or action.
- **Business Value:** Tracking user behavior helps evaluate the effectiveness of alerts and recommendations, identify usability issues, and improve the system over time. Feedback also provides insight into user satisfaction and areas for platform improvement.
- **Stored In:**
  - **MongoDB Collection:** `interaction_logs`

Query Proposals

This section details the key information requirements for the system, along with SQL queries for data retrieval and their business context, all this based on the ER diagrama presented on the initial database architecture:

User Information and Role Management

- **Description:** The system must store and manage detailed information about each user, including personal identification data (name, email, password hash), role (e.g., farmer, analyst, administrator), associated agricultural fields, and user preferences (such as alert preferences or notification channels). The system must also associate users with system access controls to enforce role-based permissions.
- **SQL Query (PostgreSQL):**

```sql
SELECT user_id, user_name, role, email
FROM User
WHERE role = Role;
```

- **Purpose:** Retrieves a list of all user with that role.
- **Insight:** Supports role-based access, user segmentation, and administration.
- **Use Case:** Analytics (for admin dashboards or user management tools).

Climate Risk Prediction Data

- **Description:** The system must store and retrieve data related to climate risk predictions, including event type (e.g., flood, drought, frost), location, probability, associated severity level, and prediction timestamp. Each prediction must be traceable to the machine learning model used for generation.
- **SQL Query (PostgreSQL):**

```sql
SELECT p.prediction_id, e.event_name, l.name AS location, p.probability, s.severity_type, p.timestamp
FROM Prediction p
JOIN Event_type e ON p.event_type_id = e.event_type_id
JOIN Severity_level s ON p.severity_level_type_id = s.severity_level_id
JOIN Location l ON p.location_id = l.location_id
WHERE p.timestamp >= CURRENT_DATE - INTERVAL '7 days';
```

- **Purpose:** Retrieves recent predictions by event type and location.
- **Insight:** Helps users understand imminent or frequent risks.
- **Use Case:** Analytics (historical or weekly reports for managers and dashboards).

Personalized Agricultural Recommendations

- **Description:** The system must store actionable recommendations linked to specific climate predictions and tailored to the crop type, field location, and user context. Each recommendation includes a textual description and reference to its originating prediction.
- **SQL Query (PostgreSQL):**

```sql
SELECT u.user_name, c.crop_name, af.field_id, r.recommen_descrip, p.timestamp
FROM Recommendation r
JOIN Prediction p ON r.prediction_id = p.prediction_id
JOIN Agriculture_Field af ON r.field_id = af.field_id
JOIN Crop c ON af.crop_id = c.crop_id
JOIN User u ON af.user_id = u.user_id
WHERE u.user_id = 1
ORDER BY p.timestamp DESC;
```

- **Purpose:** Retrieves the recommendations for a given user.
- **Insight:** Shows which actions have been recommended based on the current crop and field context.
- **Use Case:** Real-time access for end-users.

Real-Time and Historical Weather Data

- **Description:** The system must collect and manage granular weather data (temperature, humidity, wind speed, timestamp) for each weather station and location. This data should be both real-time and historically queryable.
- **SQL Query (PostgreSQL):**

```sql
SELECT wd.timestamp, wd.temperature, wd.humidity, ws.station_name, l.name AS location
FROM Weather_Data wd
JOIN Weather_Station ws ON wd.station_id = ws.station_id
JOIN Location l ON ws.location_id = l.location_id
WHERE wd.timestamp BETWEEN NOW() - INTERVAL '1 DAY' AND NOW();
```

- **Purpose:** Retrieves real-time weather data for the last 24 hours.
- **Insight:** Enables immediate decision-making for field operations like irrigation.
- **Use Case:** Real-time access for farmers and analytics for analysts.

Agricultural Field and Crop Information

- **Description:** The system must maintain detailed records of users' agricultural fields, including crop type, field ID, and location coordinates. Each user can be associated with one or more fields.
- **SQL Query (PostgreSQL):**

```sql
SELECT af.field_id, af.field_size, l.name AS location, c.crop_name, u.user_name
FROM Agriculture_Field af
JOIN Crop c ON af.crop_id = c.crop_id
JOIN Location l ON af.location_id = l.location_id
JOIN User u ON af.user_id = u.user_id
WHERE u.user_id = 1;
```

- **Purpose:** Retrieves crop and field information for a specific user.
- **Insight:** Used to personalize recommendations and predictions.
- **Use Case:** Analytics for reports and real-time access to visualize fields on the app.

Alert and Notification Records

- **Description:** The system must store alert records that include which user received what alert, for which prediction, when it was sent, and the current status (e.g., delivered, read, acknowledged).
- **SQL Query (PostgreSQL):**

```sql
SELECT u.user_name, e.event_name, a.status, a.sent_at
FROM Alert a
JOIN User u ON a.user_id = u.user_id
JOIN Prediction p ON a.prediction_id = p.prediction_id
JOIN Event_type e ON p.event_type_id = e.event_type_id
WHERE u.user_id = 1
ORDER BY a.sent_at DESC;
```

- **Purpose:** Displays all alerts received by a user, including status and associated risk.
- **Insight:** Helps users and admins track how alerts are being delivered and acknowledged.
- **Use Case:** Real-time access for farmers, analytics for admins.

User Interaction Logs and Feedback

- **Description:** The system must store user interaction data and optional feedback related to recommendations, alerts, and platform usage. These logs include timestamps, device type, interaction type (e.g., viewed recommendation, dismissed alert, submitted feedback), and additional metadata like geolocation, response time, or comments. This information can vary in structure depending on the event or action.
- **NoSQL Query (MongoDB):**

```javascript
db.interaction_logs.find({
  interaction_type: "recommendation_viewed",
  user_id: "u001"
}).sort({ timestamp: -1 }).limit(5);
```

- **Purpose:** Retrieves the 5 most recent interactions where the user with `ID u001` viewed a recommendation.
- **Insight:** Helps monitor user engagement with the recommendation system, understand behavior patterns, and identify which recommendations are being read or ignored.
- **Use Case:** Real-time access for activity monitoring in the user dashboard. Analytics for `UX` evaluation and system improvement.

Raw API Response Storage

- **Description:** The system must optionally store raw `JSON` payloads from external weather `APIs` (such as `OpenWeather`) for auditing, reprocessing, and debugging purposes. These payloads may vary in structure depending on the source, `API` version, or requested endpoint.
- **NoSQL Query (MongoDB):**

```javascript
db.api_responses.find({
  source: "OpenWeather",
  "location.name": "Cali",
  timestamp: { $gte: ISODate("2025-05-27T00:00:00Z") }
}).sort({ timestamp: -1 }).limit(10);
```

- **Purpose:** Retrieves the last 10 raw weather `API` responses from the `OpenWeather` source for the city of Cali.
- **Insight:** Supports auditability of predictions, debugging of data pipelines, and retraining of climate models using original `API` inputs.
- **Use Case:** Analytics for historical comparison and model validation. Data engineering use for reprocessing or feature extraction.

References

[1] Stonebraker, M., & Çetintemel, U. (2005). *"One size fits all": An idea whose time has come and gone*. Proceedings of the 21st International Conference on Data Engineering (ICDE), 2–11. IEEE.

[2] Marz, N., & Warren, J. (2015). *Big Data: Principles and best practices of scalable real-time data systems.* Manning Publications.

Databases II

Workshop No. 1 — Project Definition and Database Modeling

AgroClima: A Smart Agro-Climatic Decision Support Platform

César Andrés Torres Bernal
20191020147

Juan David Duarte Ruiz
20191020159

Universidad Distrital Francisco José de Caldas
School of Engineering

Introduction to the Business Model

AgroClima is a smart agro-climatic decision support platform designed to provide advanced climate-based solutions for farmers and agricultural organizations. The core idea of this project is to leverage climate data and artificial intelligence tools to deliver accurate and timely recommendations that enhance agricultural decision-making in the face of adverse weather conditions.

The main problem addressed by *AgroClima* is the high vulnerability of the agricultural sector to extreme weather events such as droughts, frosts, and heavy rainfall, which result in substantial crop losses, reduced productivity, and threats to food security. Currently, many farmers lack effective and accessible tools that would allow them to anticipate and adapt to these climate-related challenges.

The scope of the project includes the development of a final product: a user-friendly web platform specifically adapted for rural areas with limited internet connectivity. This platform will deliver personalized climate alerts, practical recommendations for agricultural management, and optimization strategies for resource use such as water and fertilizers. Furthermore, it will provide access to key regional climate risk information for public and private institutions. The overall goal of *AgroClima* is to empower small-scale farmers and relevant stakeholders in making informed strategic and operational decisions.

Key Partners

- Weather data providers (OpenWeather, Copernicus, IDEAM).

- Manufacturers and distributors of IoT sensors for agricultural use.

- Ministries of Agriculture, government agencies and NGOs that promote sustainable practices.

- Universities and climate and agricultural research institutes, which provide knowledge and scientific validation.

- Payment platforms and rural banking entities, which facilitate payments and subsidies.

- Agricultural input distributors that could partner for integrated offers.

Key Activities

- Design and management of weather forecasting and advisory services, adapted to different agricultural profiles.
- Execution of awareness and technology adoption campaigns in rural areas.
- Management of strategic relationships with governments, cooperatives, NGOs and agricultural associations.
- Technical support: development and maintenance of the digital system to offer the service.

Key Resources

- Network of strategic alliances with actors in the agricultural and climate sector.
- Access to reliable databases and satellite coverage.
- Intelligent predictive analytics and machine learning system to generate accurate and timely recommendations.
- Technological infrastructure for the efficient management of large volumes of climate and agricultural data in real time.

Value Propositions

For farmers:
- Accurate and actionable recommendations based on local climate predictions.
- Reduction of losses due to extreme events such as drought, frost or excessive rainfall.
- Optimization of the use of resources such as water and fertilizers.
- Ease of use, even without technical training.

For institutions and companies:
- Access to regional data and reports for agro-climatic decision making.
- Tools for planning subsidies, agricultural insurance and technical support.
- Improved public policies and agroclimatic disaster mitigation.

Customer Relationships

- Multichannel technical assistance (chat, telephone, mail) for rural users with limited connectivity.
- Training and support in the adoption of the platform, especially in areas with low digitalization.
- Automated and personalized alerts, without the need for constant interaction.

Channels

- Web platform accessible in rural areas with low connectivity.
- API access for institutions wishing to connect their agricultural management systems.
- Institutional partners (agricultural agencies, cooperatives) as implementation and support channels.

Customer Segments

Individual users:

- Small and medium farmers interested in protecting their crops.
- Technified producers who integrate IoT in their agricultural management.

Institutional customers:

- Regional and national governments that manage climate risks in the agricultural sector.
- Agribusiness companies seeking to optimize production and climate logistics.

Cost Structure

- Operation of alliances, training and customer service in the territory.

- Access and processing of climate and satellite data.

- Marketing, campaigns and community training.

- Investment in continuous improvement of the regional recommendation and adaptation model.

- Technical support: maintenance and technological operation costs servers, cloud, support.

Revenue Streams

- Monthly or annual subscription of farmers, with staggered plans according to needs.

- Institutional licenses for local or national governments.

- Additional services (consulting, premium data, risk maps).

- Freemium model with free basic functionalities and advanced paid plans.

- Alliances with insurance companies or input companies, which sponsor access to the service.

Requirements

1. Functional Requirements

ID	Requirement	Description	Priority
FR1	User registration	Allow farmers, technicians, and administrators to create an account using email or third-party login providers.	High
FR2	Authentication and access control	Provide secure login with differentiated access levels based on user roles.	High
FR3	Continuous data ingestion	Incorporate weather data in near real time from open or authorized sources, ensuring constant availability.	High
FR4	Distributed data storage	Store and access climate and agricultural data efficiently from any location with low latency.	High
FR5	Weather data queries	Enable users to consult and visualize current and historical weather data relevant to their agricultural zone.	High
FR6	Climate risk prediction	Offer reliable predictions about droughts, frosts, or heavy rainfall to support preventive decision-making.	High
FR7	Agricultural recommendations	Generate personalized recommendations for planting, irrigation, and fertilization based on local microclimates.	Medium
FR8	Report generation	Allow users to create customized reports for farmers and institutions to support strategic decisions.	Medium
FR9	Administrative panel	Provide administrative tools to manage users, roles, and global system statistics.	High

2. Non-Functional Requirements

ID	Requirement	Description	Priority
NFR1	Performance	The system must respond to queries and predictions within acceptable times for the expected data volumes.	High
NFR2	Horizontal scalability	Must support efficient scaling as the number of users or data ingestion volume increases.	High
NFR3	High availability	Must minimize downtime through automatic recovery mechanisms, ensuring operational continuity.	High
NFR4	Multi-region access	Must provide reasonable load times to geographically distributed users, regardless of physical location.	Medium
NFR5	Interoperability	Must integrate with external weather and agricultural data sources using standardized protocols.	Medium
NFR6	Usability	The interface should be intuitive, responsive, and usable on various devices, even with limited connectivity.	Medium
NFR7	Maintainability	The system should have a modular and well-documented architecture to support updates and scaling.	Medium

3. Prioritization Strategy — Applied Criteria

- **Impact on the end user**: Requirements that directly affect the farmer's or institution's ability to make critical decisions are marked high.
- **Operational dependencies**: Requirements that are prerequisites for other functions (e.g., authentication before access) are rated high.
- **Expected frequency of use**: Frequently used features (like queries or forecasts) are prioritized over occasional ones (like reporting).
- **Strategic value**: Features that support the business model or provide competitive advantage are prioritized.
- **Incremental benefit vs. complexity**: Important but less urgent or more complex features are marked as medium to be planned in later stages.

Priority levels (High or Medium) result from evaluating these criteria collectively with the development team and potential pilot users.

4. Performance and Capacity Analysis

System Dimensioning Assumptions

- Registered users: 2,000 in year one.
- Peak concurrent users: 500 (25%).
- Data sources: 1,200 sensors / 1 reading per minute → 1.7M records/day.
- Peak usage windows: 5:00–8:00 AM and 6:00–9:00 PM.
- Rural limitations: 2–5 Mbps connections, unstable latencies.

Target Metrics

Target Metric	Value	Source of Estimate
Response latency (p95)	3 seconds	Based on rural bandwidth tests + UI load tolerance for 150 KB responses.
Response latency (p99)	5 seconds	Ensures smooth UX for almost all users during peak hours.
Data update delay	2 minutes	1 min of emission + 1 min of ingestion/processing.
Ingestion throughput	2,000 rec/s sustained (6,000 peak)	Derived from 1.7M daily recs × peak load factor × 3.
Availability	99.5% (3.6 h downtime/month)	Balanced cost-benefit for mid-tier infrastructure.
Payload size	150 KB	Suitable for 3G/4G connections to ensure ¡ 0.5s load.
Scalability	Double capacity in ¡ 1 hour	Tested via auto-provisioning in pilot environment.

Source of Numbers

- Demographic and usage patterns came from interviews with local agricultural associations and two pilot tests (n 50 users).
- Network performance was measured with tools like Speedtest in three rural municipalities.
- Peak values include safety factors ×3–5 over observed averages to cover climate emergency scenarios and system growth.
- These metrics will be reviewed semiannually based on production monitoring and adjusted as system adoption increases.

Sizing Methodology

- **Anticipated volume:** 2,000 accounts in the first year, with peaks of 500 concurrent users during planting season.

- **Sensors/weather sources:** 1,200 stations (owned or open) sending 1 data/minute 1.7 million records/day.
- **Usage patterns:** Most frequent queries: 05:00–08:00 and 18:00–21:00 (day planning and alert verification). On average, a user executes 3 queries per session.
- **Limitations:** Rural connectivity: 2–5 Mbps links and unstable latencies.

User Stories

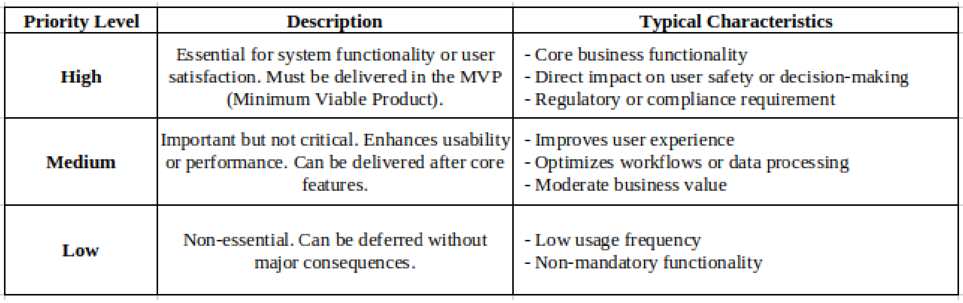
This section outlines the user stories that guide the functional development of the AgroClima Climate Risk Prediction System. User stories are structured descriptions of system features from the perspective of end users, helping the development team stay aligned with user needs. To ensure consistency, clarity, and prioritization in planning and execution, this section also includes:

- **Priority Rubric:** A framework for classifying stories as High, Medium, or Low priority based on business impact, user value, and technical dependencies.
- **Estimation Metrics:** A story point-based system used to estimate the relative effort required for implementing each user story.

Each user story is presented in a standardized format and includes clearly defined acceptance criteria to support testing and validation. The stories are grouped by user role (Client, Analyst, Administrator) to reflect the different system interactions.

Priority Rubrics

The priority rubric defines how each user story is categorized based on its importance to the system's core functionality, user impact, and time sensitivity. This classification helps the team focus on delivering the most critical features first, ensuring that essential needs are met in early development phases. It should be noted that the priority levels assigned to each user story are based on the defined scope of the project. This does not imply that some stories are inherently more important than others, but rather that certain functionalities are more critical to meet the objectives of the current delivery stage of the system.

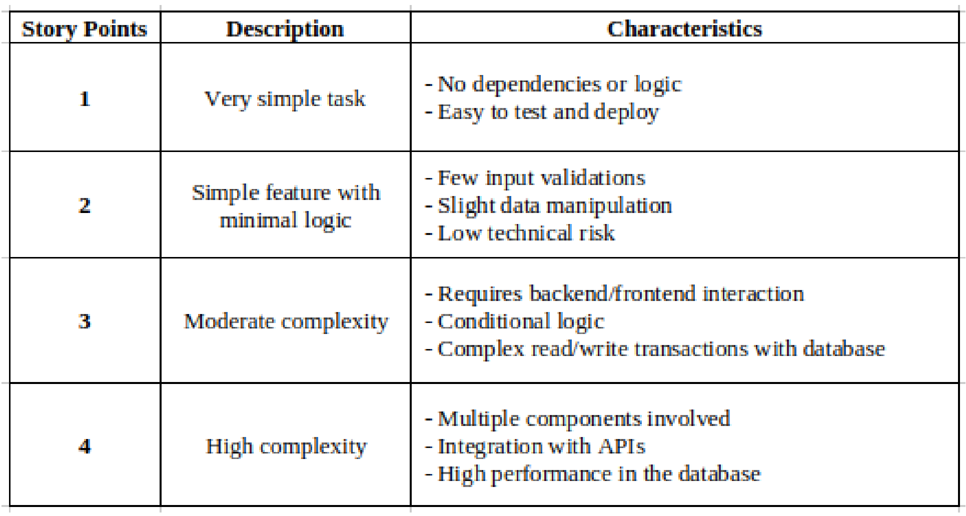


Priority Level	Description	Typical Characteristics
High	Essential for system functionality or user satisfaction. Must be delivered in the MVP (Minimum Viable Product).	- Core business functionality - Direct impact on user safety or decision-making - Regulatory or compliance requirement
Medium	Important but not critical. Enhances usability or performance. Can be delivered after core features.	- Improves user experience - Optimizes workflows or data processing - Moderate business value
Low	Non-essential. Can be deferred without major consequences.	- Low usage frequency - Non-mandatory functionality

Figure 1: Priority Rubrics

Estimation Rubrics

Estimation metrics provide to evaluate the relative effort required to implement each user story. These metrics will help the team to plan sprints more accurately, balance workloads, and assess development complexity without relying on exact time measurements. Points are assigned based on factors such as technical complexity, risk, dependencies, and required resources.



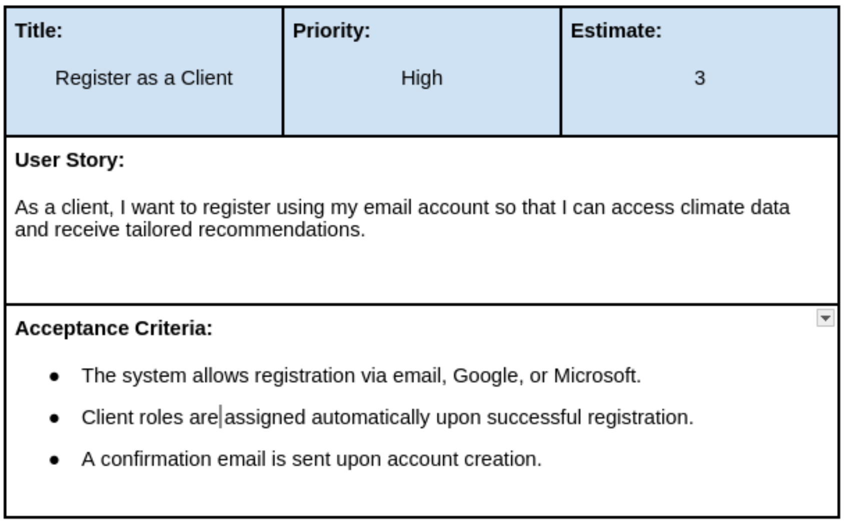
Story Points	Description	Characteristics
1	Very simple task	- No dependencies or logic - Easy to test and deploy
2	Simple feature with minimal logic	- Few input validations - Slight data manipulation - Low technical risk
3	Moderate complexity	- Requires backend/frontend interaction - Conditional logic - Complex read/write transactions with database
4	High complexity	- Multiple components involved - Integration with APIs - High performance in the database

Figure 2: Estimation metrics

User Stories

Based on the priority rubrics and estimation metrics defined earlier, we developed a set of user stories for the AgroClima system to identify and address the primary needs of its users. The user stories are organized by user roles within the system: the Client, typically a farmer or crop owner; the Analyst, who processes, analyzes, and visualizes environmental data; and the Administrator, who manages user access and ensures proper and secure system operation.

User Stories (Client)



Title:	Priority:	Estimate:
Register as a Client	High	3

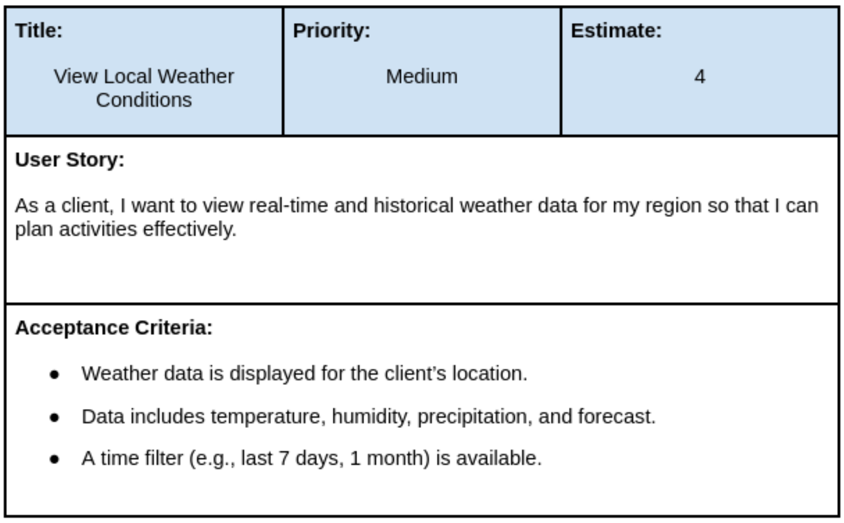
User Story:

As a client, I want to register using my email account so that I can access climate data and receive tailored recommendations.

Acceptance Criteria:

- The system allows registration via email, Google, or Microsoft.
- Client roles are assigned automatically upon successful registration.
- A confirmation email is sent upon account creation.

Figure 3: User Story 1



Title: View Local Weather Conditions	Priority: Medium	Estimate: 4

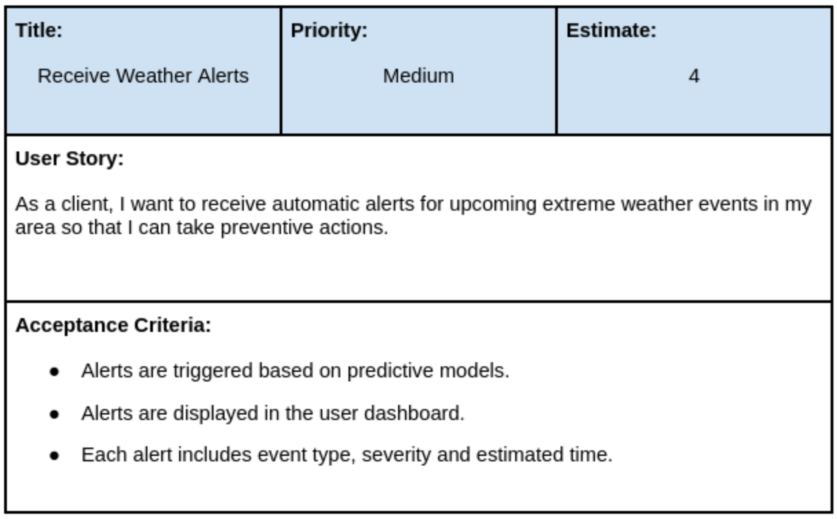
User Story:

As a client, I want to view real-time and historical weather data for my region so that I can plan activities effectively.

Acceptance Criteria:

- Weather data is displayed for the client's location.
- Data includes temperature, humidity, precipitation, and forecast.
- A time filter (e.g., last 7 days, 1 month) is available.

Figure 4: User Story 2



Title: Receive Weather Alerts	Priority: Medium	Estimate: 4

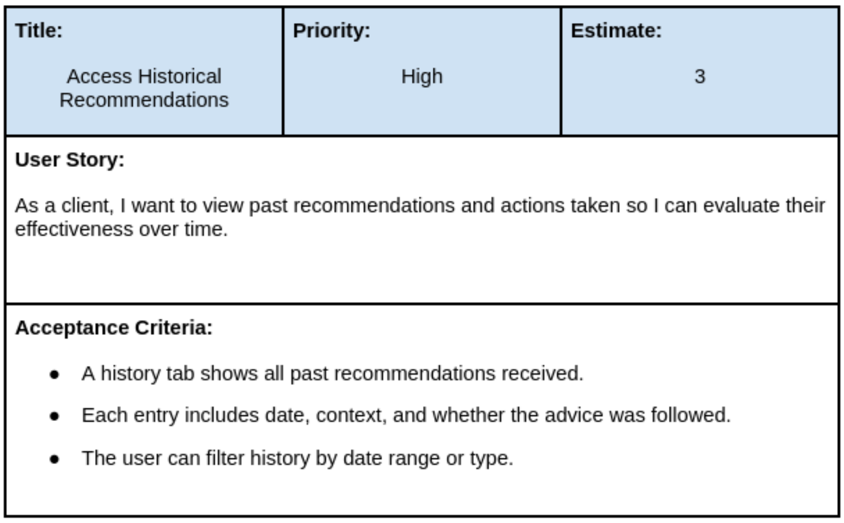
User Story:

As a client, I want to receive automatic alerts for upcoming extreme weather events in my area so that I can take preventive actions.

Acceptance Criteria:

- Alerts are triggered based on predictive models.
- Alerts are displayed in the user dashboard.
- Each alert includes event type, severity and estimated time.

Figure 5: User Story 3



Title:	Priority:	Estimate:
Access Historical Recommendations	High	3

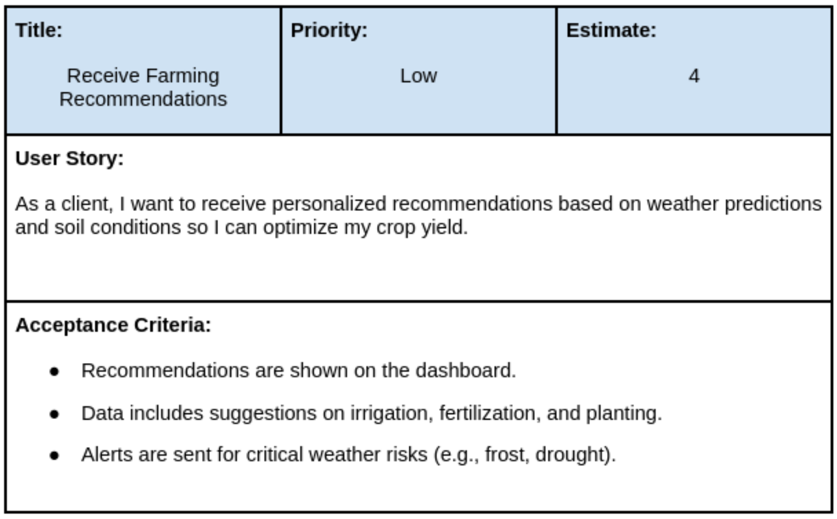
User Story:

As a client, I want to view past recommendations and actions taken so I can evaluate their effectiveness over time.

Acceptance Criteria:

- A history tab shows all past recommendations received.
- Each entry includes date, context, and whether the advice was followed.
- The user can filter history by date range or type.

Figure 6: User Story 4



Title:	Priority:	Estimate:
Receive Farming Recommendations	Low	4

User Story:

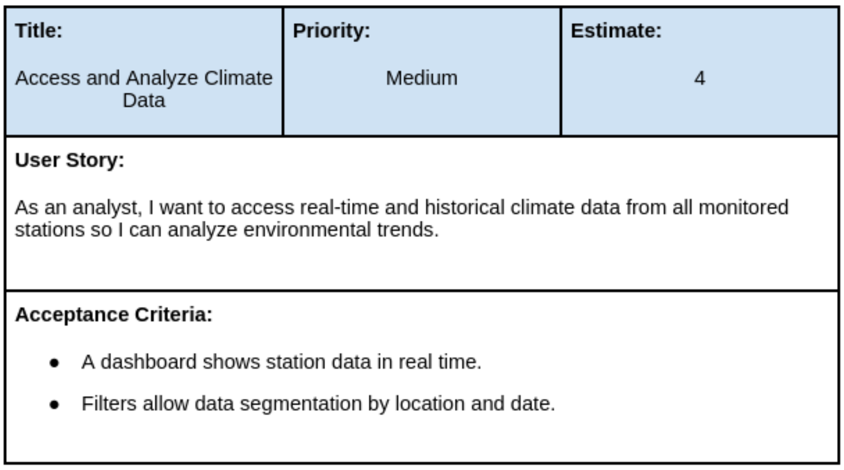
As a client, I want to receive personalized recommendations based on weather predictions and soil conditions so I can optimize my crop yield.

Acceptance Criteria:

- Recommendations are shown on the dashboard.
- Data includes suggestions on irrigation, fertilization, and planting.
- Alerts are sent for critical weather risks (e.g., frost, drought).

Figure 7: User Story 5

User Stories (Analyst)



Title:	**Priority:**	**Estimate:**
Access and Analyze Climate Data	Medium	4

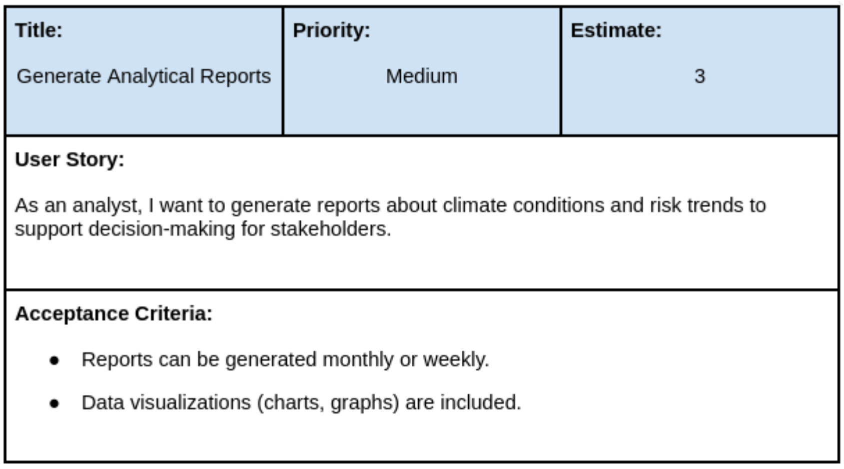
User Story:

As an analyst, I want to access real-time and historical climate data from all monitored stations so I can analyze environmental trends.

Acceptance Criteria:

- A dashboard shows station data in real time.
- Filters allow data segmentation by location and date.

Figure 8: User Story 6



Title:	**Priority:**	**Estimate:**
Generate Analytical Reports	Medium	3

User Story:

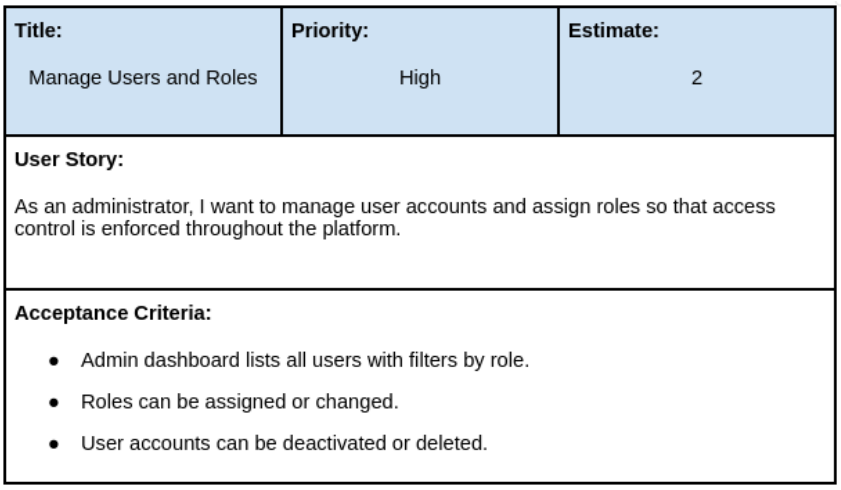
As an analyst, I want to generate reports about climate conditions and risk trends to support decision-making for stakeholders.

Acceptance Criteria:

- Reports can be generated monthly or weekly.
- Data visualizations (charts, graphs) are included.

Figure 9: User Story 7

User Stories (Administrator)



Title:	**Priority:**	**Estimate:**
Manage Users and Roles	High	2

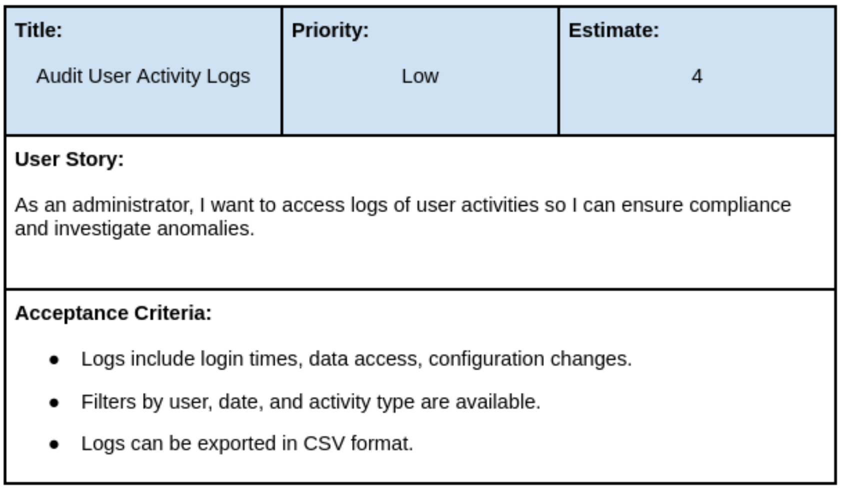
User Story:

As an administrator, I want to manage user accounts and assign roles so that access control is enforced throughout the platform.

Acceptance Criteria:

- Admin dashboard lists all users with filters by role.
- Roles can be assigned or changed.
- User accounts can be deactivated or deleted.

Figure 10: User Story 8



Title: Audit User Activity Logs	Priority: Low	Estimate: 4

User Story:

As an administrator, I want to access logs of user activities so I can ensure compliance and investigate anomalies.

Acceptance Criteria:

- Logs include login times, data access, configuration changes.
- Filters by user, date, and activity type are available.
- Logs can be exported in CSV format.

Figure 11: User Story 9

Initial Database Architecture

This section presents a conceptual design of the initial database architecture proposed for the AgroClima system. The system is intended to collect, process, store, and provide access to large volumes of data related to agricultural environments, including weather conditions and user activity. The proposed architecture is designed to be scalable, modular, and capable of supporting predictive analytics and real-time decision-making, while maintaining a clear separation of concerns across its components.

High-Level Architecture Proposal

The architecture of the AgroClima platform is designed to meet the specific needs of a system focused on predicting climate-related agricultural risks. This requires the integration of diverse environmental data sources, the continuous processing of high-volume geospatial and time-series data, and the delivery of accurate and timely recommendations to a heterogeneous group of users, including farmers, analysts, and administrators.

This High-level architectural proposal is structured not around specific technologies, but around solving the core functional and non-functional requirements of the system. These include scalability, data integrity, low-latency access to forecasts, support for historical data analysis, and secure multi-user access.

1. Data Sources and Acquisition

The system must gather data from multiple heterogeneous sources, which vary in structure, frequency, and origin. These sources include:

- Real-time environmental measurements, such as temperature, humidity, and soil moisture from in-field sensors or public meteorological networks.
- Historical climate datasets, used for training predictive models and evaluating long-term patterns.
- External services, such as weather forecast APIs, satellite data repositories, or governmental platforms.

The data acquisition layer must support both continuous ingestion of real-time data and batch ingestion of historical or bulk data, ensuring flexibility in handling structured, semi-structured, and unstructured formats.

2. Data Ingestion and Preprocessing

To ensure consistency and reliability in downstream processes, all incoming data must be validated, cleaned, and transformed. The ingestion layer should support:

- The decoupling of data producers from processors, allowing asynchronous and scalable data flows.
- The detection and correction of anomalies or missing values before data enters the core system.

This layer serves as a critical buffer between raw data collection and structured data storage, facilitating quality control and enabling fault tolerance.

3. Data Storage Architecture

Given the volume, velocity, and variety of the data, the storage architecture must accommodate multiple data types and access patterns. The storage design is composed of two main components:

- A raw data repository that temporarily stores ingested data in its original format for traceability, reprocessing, or auditing purposes.
- A processed data store that holds structured information, supporting efficient querying.

The processed store should enable time-efficient access to both historical records and the latest readings.

4. Data Processing and Analytics Layer

Once the data is curated and structured, it must be processed to extract actionable insights. This includes:

- Feature extraction and data enrichment, such as calculating vegetation indices, aggregating precipitation levels, or linking sensor readings to crop stages.
- Execution of predictive models, which estimate the likelihood and severity of climate events based on historical and current inputs.
- Triggering of alerts and recommendations, based on the outputs of the predictive algorithms.

This layer must support both real-time processing (for alerts and urgent decisions) and batch processing (for retrospective analysis and reporting).

5. Application and Serving Layer

This layer is responsible for providing personalized, timely, and reliable access to the system's outputs. It serves:

- Visual dashboards for analysts, with access to aggregated data, prediction outputs, and historical comparisons.
- Advisory interfaces for farmers, focused on immediate recommendations, alert notifications, and field-specific information.
- Administrative tools for managing user roles, permissions, and system configurations.

The application layer must enforce role-based access control, ensure data security, and offer modular interfaces that adapt to different user profiles and usage scenarios.

6. Access and Integration Interfaces

To allow external systems, applications, and users to interact with AgroClima, a set of structured interfaces is exposed through secure APIs. These interfaces must:

- Support modular access to data and services, depending on the user's role and needs.
- Enable integration with third-party systems, such as government platforms, mobile apps, or agricultural decision-support tools.

7. Data Flow Overview

The overall flow of data through the system proceeds as follows:

- Acquisition: Data is collected from external APIs, sensors, and user uploads.
- Landing and Validation: Incoming data is stored temporarily and validated for structure and quality.
- Processing: Data is transformed, enriched, and passed through prediction and analysis pipelines.
- Storage: Structured outputs are stored in databases optimized for querying and access.
- Interaction: End users access insights via dashboards, alerts, or API endpoints, depending on their profile.

This flow ensures traceability, adaptability, and responsiveness throughout the system.

ER Diagram (Initial Version)

The following section presents an initial version of the Entity-Relationship (ER) diagram for the AgroClima system. This diagram has been developed by applying a structured 10-step design methodology, which facilitates a systematic understanding of the domain and its key components. The ER model serves as a conceptual representation of the system's data structure, capturing the main entities, their attributes, and the relationships between them. This model provides a foundation for the logical and physical design of the database and ensures alignment between system requirements and data organization.

1. Components Definition

The first step consists of identifying the core components involved in the domain of the system. AgroClima focuses on collecting environmental data, processing it through prediction models, and generating actionable recommendations for agricultural users. Thus, the essential components are:

- System users
- Agricultural fields and cultivated crops
- Meteorological data
- Predictive models and generated risk events
- Alerts and recommendations for decision-making
- Geographical locations and external data sources

These components provide the foundation for defining the data entities required to represent and support the system's operations.

2. Entities Definition

Entities represent real-world objects or concepts that need to be stored and managed in the database. From the components above, we identify the following entities:

- ***User***: Represents the individuals registered in the AgroClima platform. Users may have different roles such as farmers, analysts, or administrators and are the recipients of climate alerts and predictions.
- ***Agriculture_field***: Represents a physical agricultural field owned or managed by a user. Each field is geolocated and can be associated with one or more crops. This entity allows AgroClima to tailor predictions and recommendations to specific land plots.
- ***Crop***: Stores the crops cultivated in each agricultural field. This allows the platform to provide crop-specific recommendations and monitor weather impacts on individual crop types.
- ***Prediction***: Stores the results generated by machine learning models that predict extreme weather events that may affect specific locations.
- ***Alert***: Stores records of alerts sent to users when a relevant climate event is predicted. Alerts are based on predictions generated by the system.
- ***Recommendation***: Stores system-generated recommendations based on predictions and field conditions. These are delivered to farmers to support agricultural decisions such as irrigation, fertilization, or harvesting.
- ***Weather_Station***: Represents the weather stations (physical or virtual) that collect meteorological data used by the system to feed predictions.
- ***Weather_Data***: Stores individual weather readings collected by a weather station at a specific timestamp. These readings are key inputs for generating predictions.
- ***Location***: Represents geographic locations associated with weather stations or prediction areas. Enables spatial organization of environmental data.

- ***Event_type***: Defines the types of weather events that the system can predict.
- ***Severity_level***: Categorizes the severity level of a predicted event, helping to prioritize alerts and responses.
- ***Model***: Contains metadata about the predictive models used by AgroClima, including versioning and training information.
- ***Data_source***: Defines the external or internal sources of meteorological data used in the system. This allows tracking of data provenance and managing integration with third-party APIs.

Each entity will later be expanded with attributes and relationships.

3. Define Attibutes per Entity

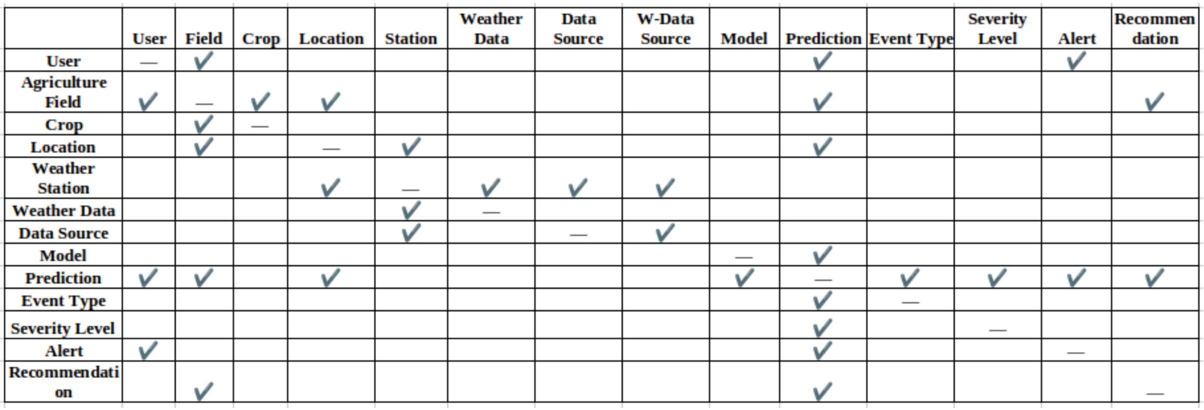
In this step, each entity is described by a set of attributes that define its properties:

- ***User***: user_id, user_name, email, role, password_hash
- ***Agriculture_field***: field_id, field_type
- ***Crop***: crop_id, crop_name
- ***Prediction***: prediction_id, timestamp, probability
- ***Alert***: alert_id, timestamp_sent, status
- ***Recommendation***: recommend_id, recommend_description
- ***Weather_Station***: station_id, name
- ***Weather_Data***: data_id timestamp, temperature, humidity, wind_speed
- ***Location***: location_id, name, latitude, longitude, type
- ***Event_type***: event_type_id, event_name, event_description
- ***Severity_level***: severity_level_id, severity_type, level_description
- ***Model***: model_id, model_name, version, training_timestamp
- ***Data_source***: source_id, name, type, url

Each attribute is assigned a data type and constraints in later stages.

4. Relationships Definition

The following table presents the relationship matrix among the identified entities in the AgroClima system. This matrix facilitates the systematic identification of associations between pairs of entities, helping to visualize how data elements interact within the system. A check mark indicates that there is a direct relationship between the corresponding entities, typically supported by a foreign key in the database schema.



	User	Field	Crop	Location	Station	Weather Data	Data Source	W-Data Source	Model	Prediction	Event Type	Severity Level	Alert	Recommendation
User	—	✓								✓			✓	
Agriculture Field	✓	—	✓	✓						✓				✓
Crop		✓	—											
Location		✓		—	✓					✓				
Weather Station				✓	—	✓	✓	✓						
Weather Data					✓	—								
Data Source					✓		—	✓						
Model									—	✓				
Prediction	✓	✓		✓					✓	—	✓	✓	✓	✓
Event Type										✓	—			
Severity Level										✓		—		
Alert	✓									✓			—	
Recommendation		✓								✓				—

Table 1: Relationship Matrix

5. Relationships Types Definition

Each identified relationship is classified by cardinality:

- **One-to-Many:**
  - A user → many fields
  - A user → many alerts
  - A field → one crop
  - A prediction → many alerts
  - A prediction → many recommendations
  - A station → many weather data entries
- **Many-to-One:**
  - A weather station → one location
  - A prediction → one location, one event type, one severity level, one model
- **Many-to-Many:**
  - A weather station ↔ multiple data sources

This classification supports correct implementation of referential integrity and constraint rules.

6. First Entity-Relationship Model Draw

At this stage, an initial version of the Entity-Relationship (ER) model is constructed based on the previously defined entities, attributes, and relationships. This first diagram provides a conceptual visualization of the system's data structure, capturing the logical connections between entities. Although preliminary, it serves as a foundation for further refinement and normalization in subsequent steps.

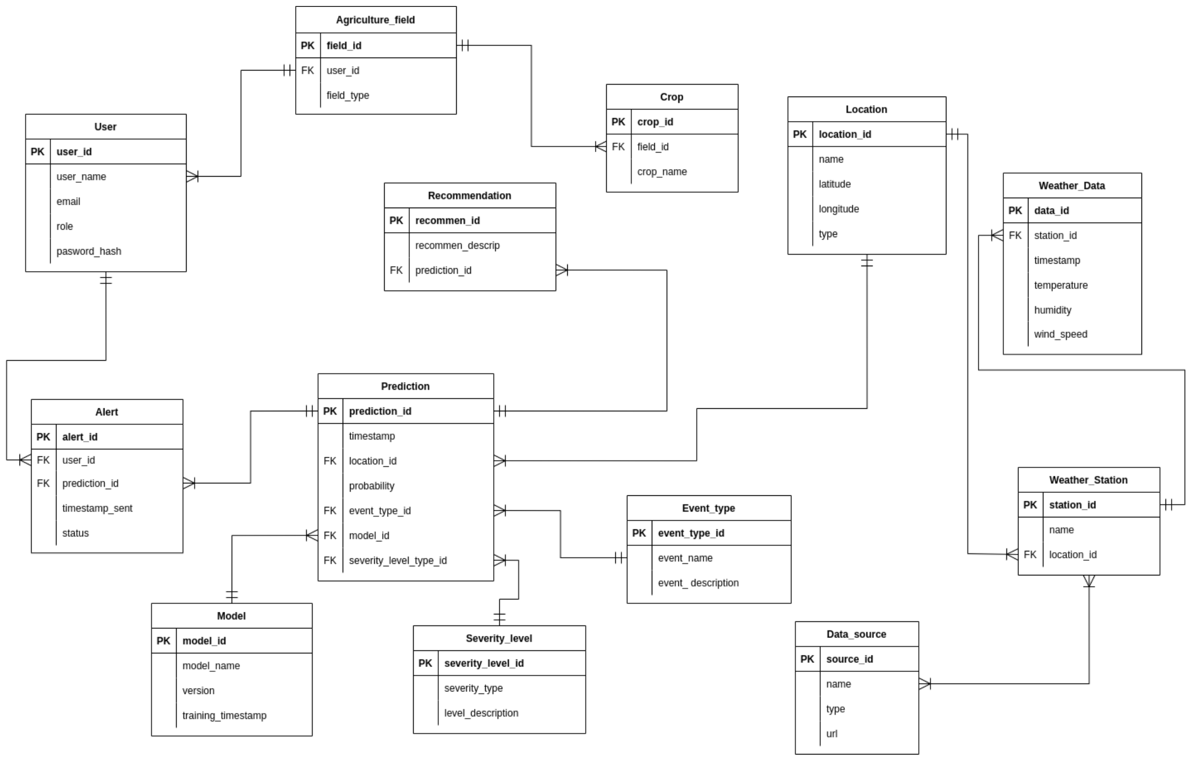


Figure 1: First ER Model Draw

7. Split Many-toMany Relationships

To comply with best practices in relational modeling, any many-to-many relationships are broken down. Specifically:
The relationship between weather stations and data sources is transformed into an associative entity, *Weather_data_source*, which includes:

- *station_id* (FK)
- *source_id* (FK)
- *start_date* (attribute)

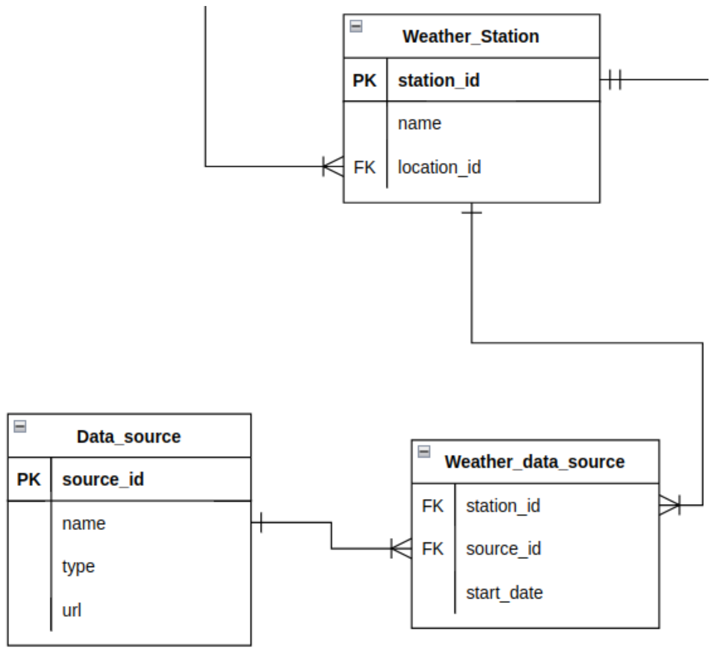


Figure 2: Many-to-Many relationship fix

This allows tracking the history of data provider associations for each station.

8. Second Entity-Relationship Model Draw

This step presents the refined version of the Entity-Relationship (ER) model, incorporating adjustments derived from the analysis of relationship types and attribute distribution. The updated diagram resolves any previously identified many-to-many relationships through associative entities and improves overall clarity and consistency.

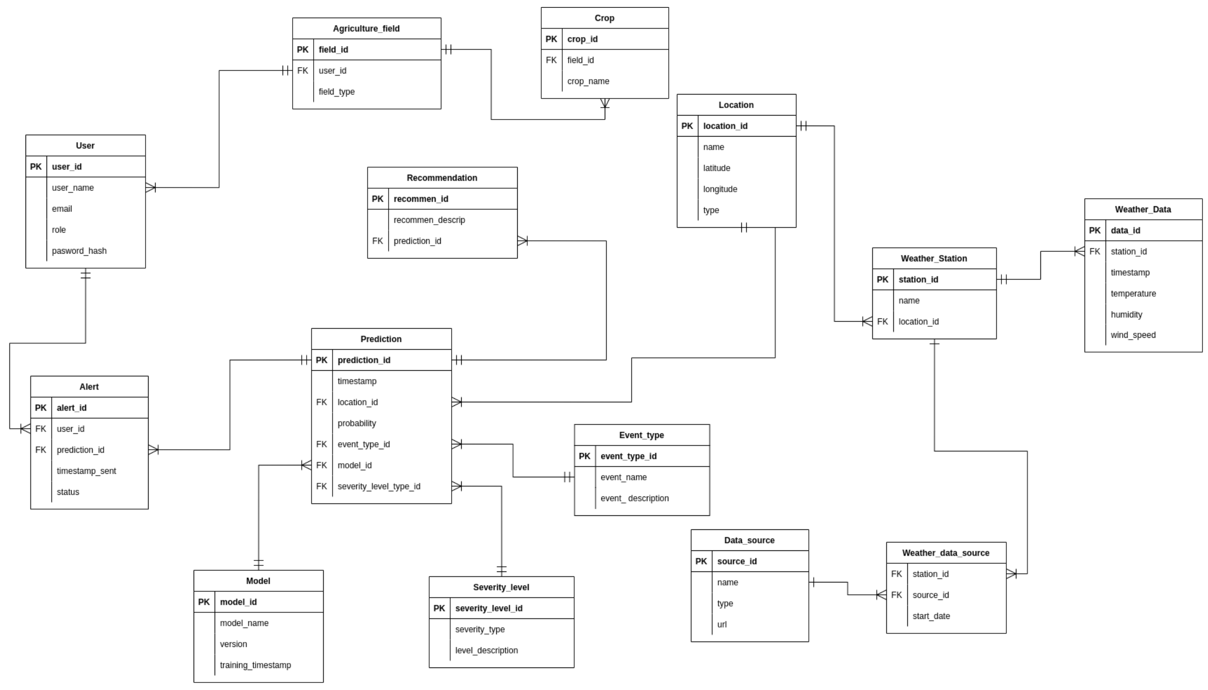
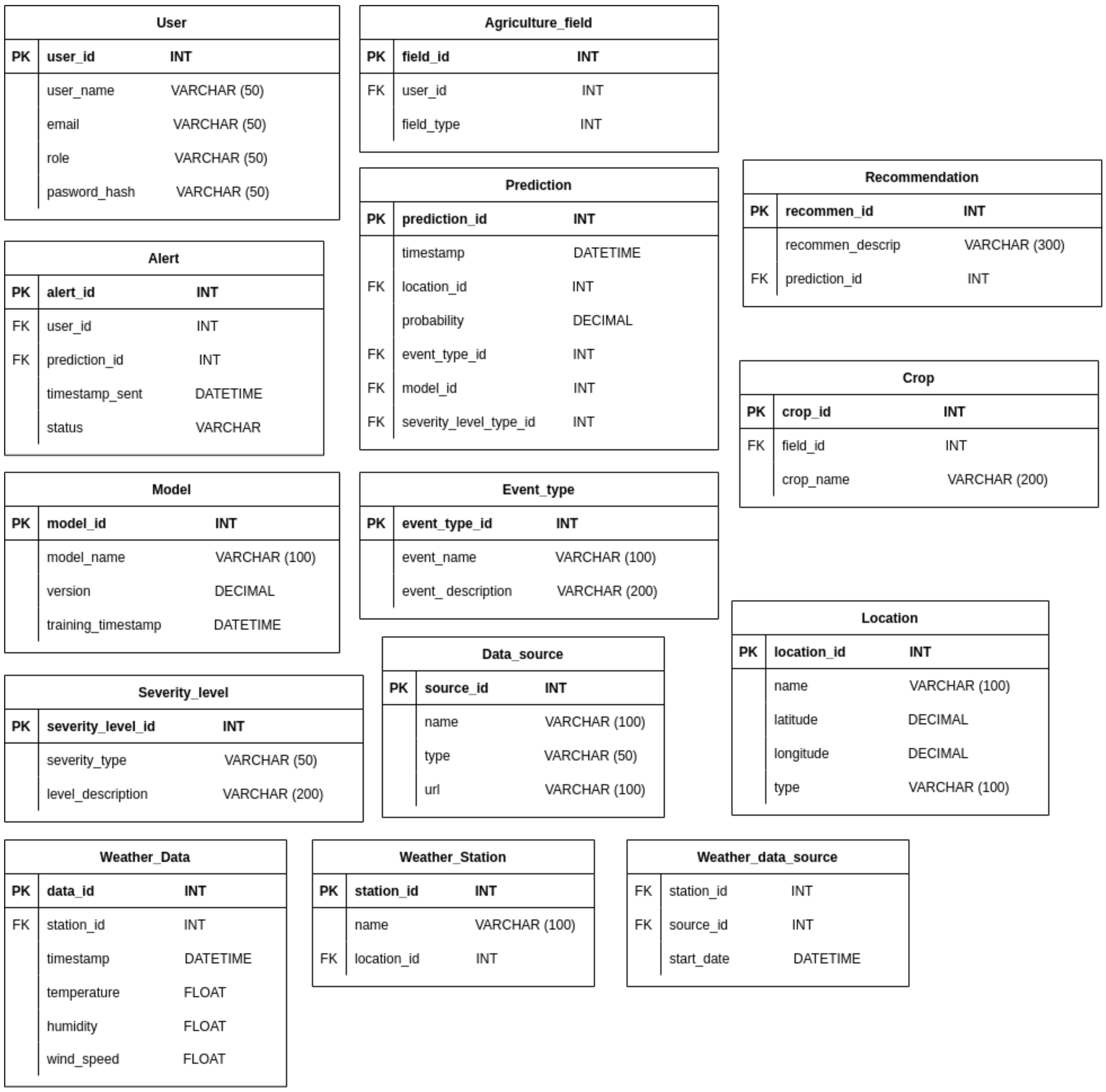


Figure 3: Second ER Model Draw

9. Get Data-Structure Entity-Relationship Model

In this step, the conceptual design of the system is translated into a structured data model suitable for implementation in a relational database management system. This data-structure ER model includes detailed definitions of data types and normalization refinements. It serves as a bridge between the abstract representation of the system and its physical realization, ensuring data consistency, integrity, and scalability.



User

PK	**user_id**	**INT**
	user_name	VARCHAR (50)
	email	VARCHAR (50)
	role	VARCHAR (50)
	pasword_hash	VARCHAR (50)

Agriculture_field

PK	**field_id**	**INT**
FK	user_id	INT
	field_type	INT

Prediction

PK	**prediction_id**	**INT**
	timestamp	DATETIME
FK	location_id	INT
	probability	DECIMAL
FK	event_type_id	INT
FK	model_id	INT
FK	severity_level_type_id	INT

Recommendation

PK	**recommen_id**	**INT**
	recommen_descrip	VARCHAR (300)
FK	prediction_id	INT

Alert

PK	**alert_id**	**INT**
FK	user_id	INT
FK	prediction_id	INT
	timestamp_sent	DATETIME
	status	VARCHAR

Crop

PK	**crop_id**	**INT**
FK	field_id	INT
	crop_name	VARCHAR (200)

Model

PK	**model_id**	**INT**
	model_name	VARCHAR (100)
	version	DECIMAL
	training_timestamp	DATETIME

Event_type

PK	**event_type_id**	**INT**
	event_name	VARCHAR (100)
	event_ description	VARCHAR (200)

Location

PK	**location_id**	**INT**
	name	VARCHAR (100)
	latitude	DECIMAL
	longitude	DECIMAL
	type	VARCHAR (100)

Data_source

PK	**source_id**	**INT**
	name	VARCHAR (100)
	type	VARCHAR (50)
	url	VARCHAR (100)

Severity_level

PK	**severity_level_id**	**INT**
	severity_type	VARCHAR (50)
	level_description	VARCHAR (200)

Weather_Data

PK	**data_id**	**INT**
FK	station_id	INT
	timestamp	DATETIME
	temperature	FLOAT
	humidity	FLOAT
	wind_speed	FLOAT

Weather_Station

PK	**station_id**	**INT**
	name	VARCHAR (100)
FK	location_id	INT

Weather_data_source

FK	station_id	INT
FK	source_id	INT
	start_date	DATETIME

Figure 4: Entities Data-structure

10. Final Initial ER Model Draw

The final step in the entity-relationship modeling process involves defining the constraints and structural properties that govern how data is stored, validated, and maintained within the database system.

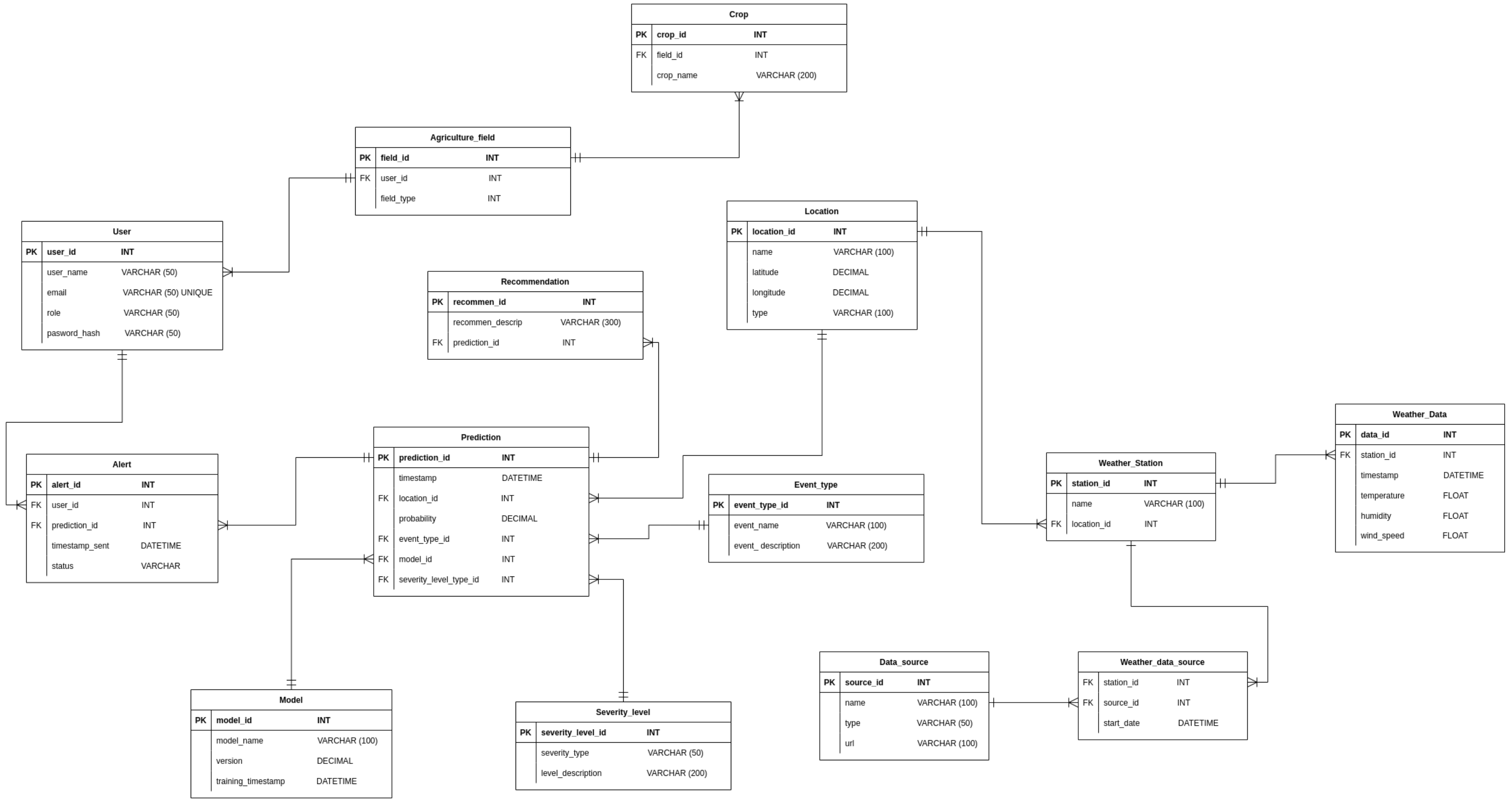


Figure 5: Initial ER Model Draw

References

1 R. Elmasri and S. B. Navathe, *Fundamentals of Database Systems*, 7th ed. Boston, MA, USA: Pearson, 2016.

2 C. A. Sierra, ”INTRODUCTION TO DATABASES / Database Foundations,” Lecture, Universidad Distrital Francisco José de Caldas, Bogotá, Colombia, 2024.