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.

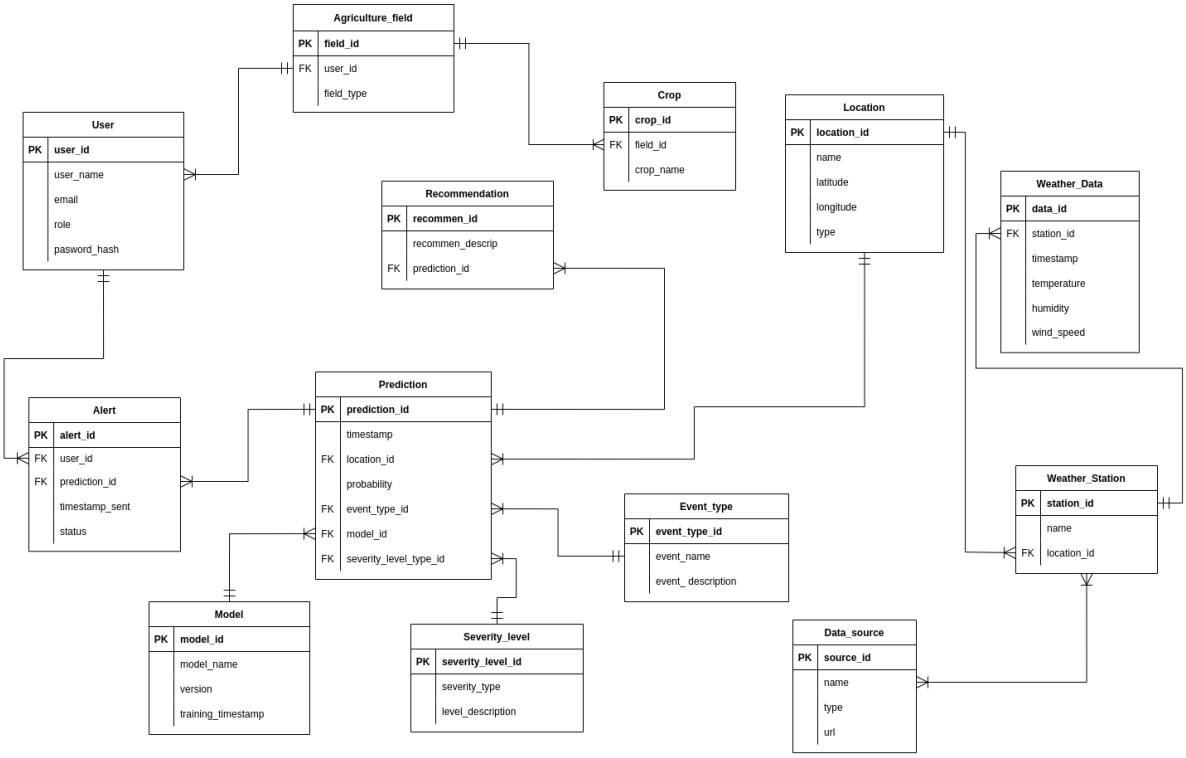

Figure 1: First ER Model Draw

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)

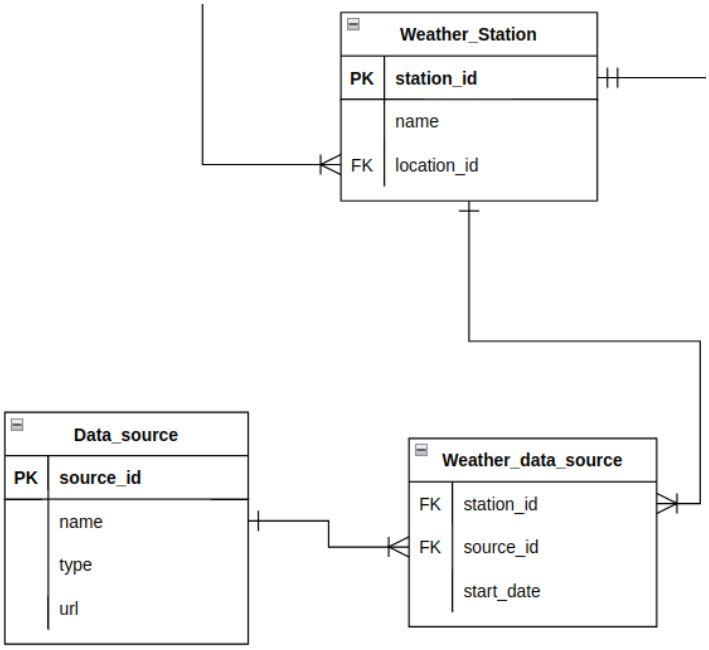

Figure 2: Many-to-Many relationship fix

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.

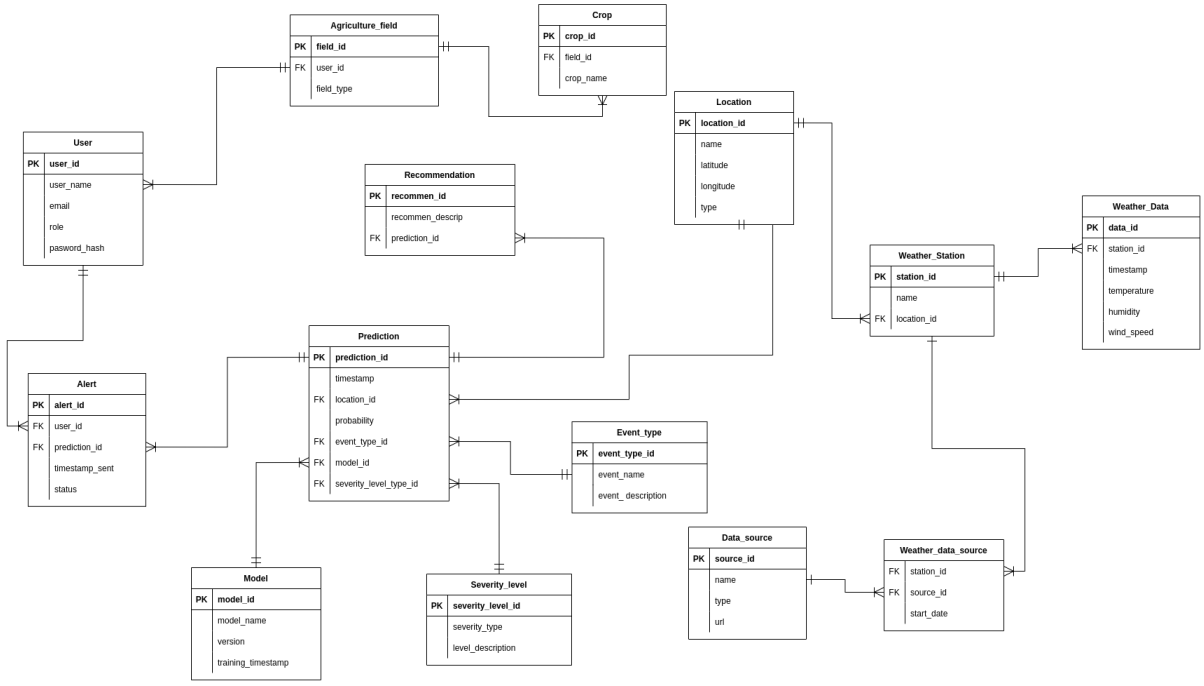

Figure 3: Second ER Model Draw

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.

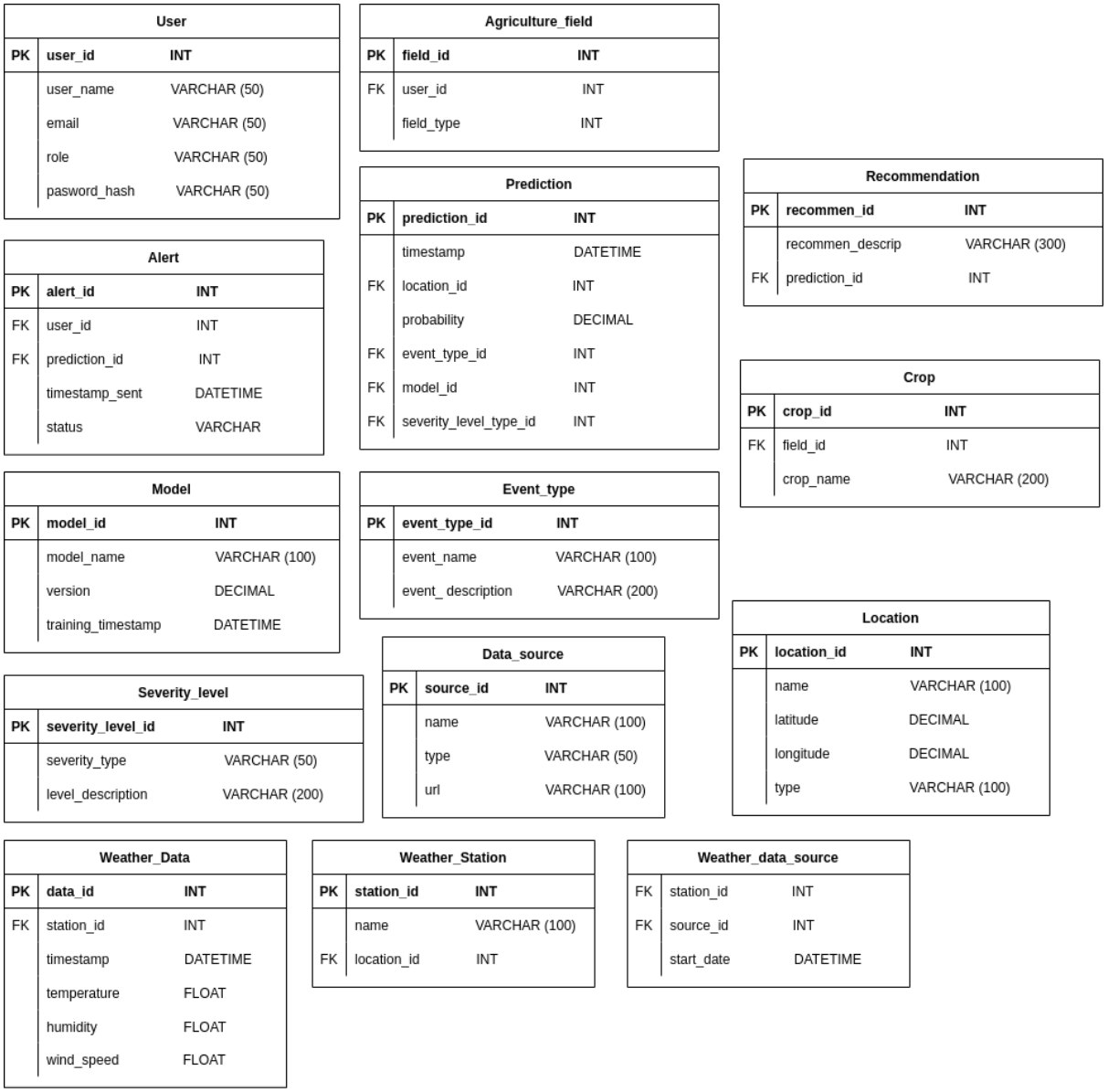

Figure 4: Entities Data-structure

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.

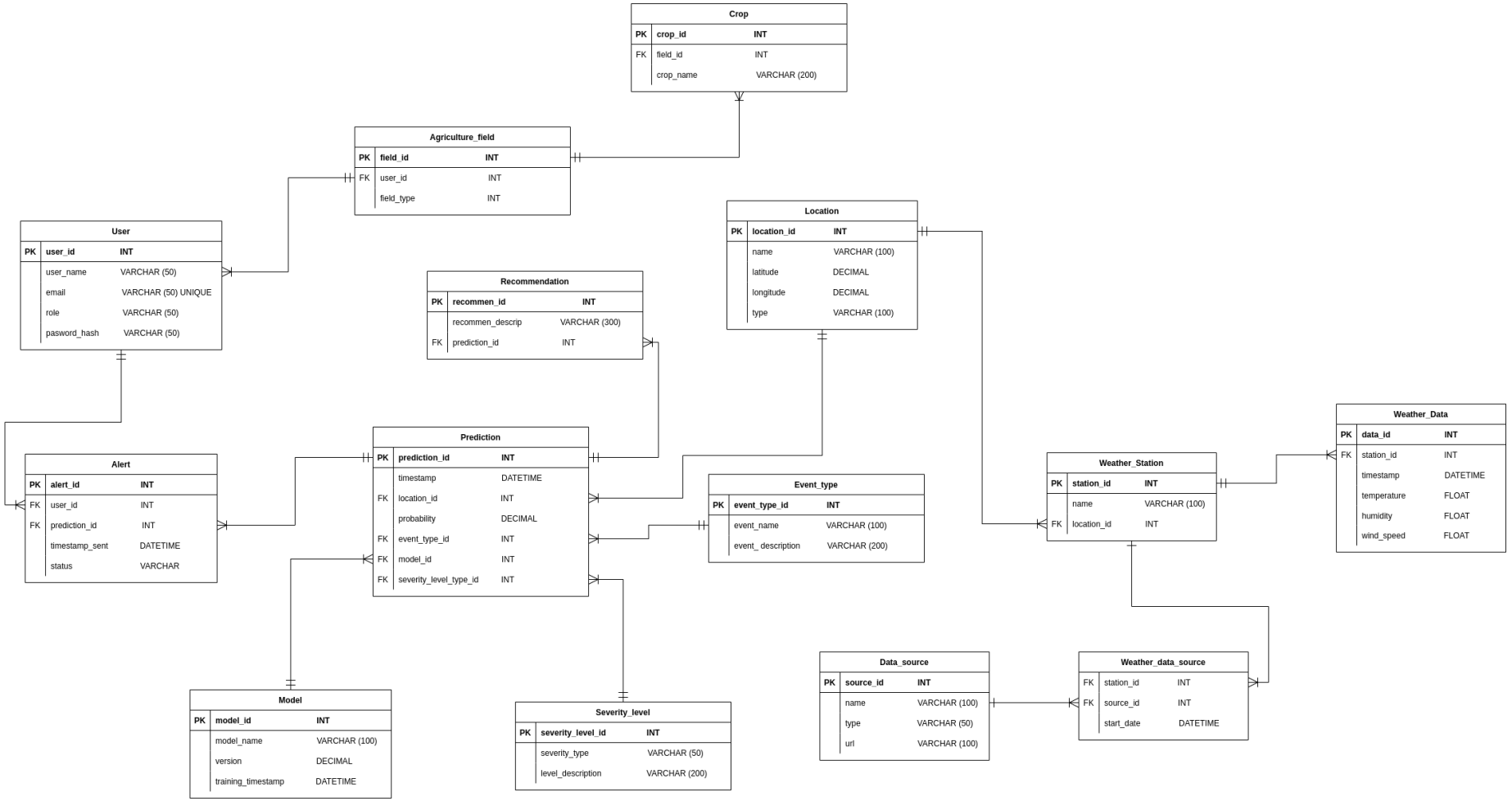

Figure 5: Initial ER Model Draw

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.