Entities and Attributes

1. **Users**
    - `user_id` (PK)
    - `username`
    - `email`
    - `password_hash`
    - `salt`
    - `created_at`
    - `updated_at`
2. **Admins**
    - `admin_id` (PK)
    - `username`
    - `email`
    - `password_hash`
    - `salt`
    - `role`
    - `created_at`
    - `updated_at`
3. **Images**
    - `image_id` (PK)
    - `user_id` (FK)
    - `image_url`
    - `uploaded_at`
    - `is_processed`
4. **ProcessingResults**
    - `result_id` (PK)
    - `image_id` (FK)
    - `processed_data`
    - `processed_at`
5. **HelpTickets**
    - `ticket_id` (PK)
    - `user_id` (FK)
    - `admin_id` (FK)
    - `subject`
    - `description`
    - `status`
    - `created_at`
    - `updated_at`
6. **Sessions**

- `session_id` (PK)
- `user_id` (FK)
- `login_at`
- `logout_at`

7. **ErrorLogs**
    - `log_id` (PK)
    - `user_id` (FK)
    - `image_id` (FK, nullable)
    - `error_message`
    - `logged_at`

8. AI Models

- **model_id** (PK) — Unique identifier for the AI model.
- **admin_id** (FK) — Foreign key referencing the **Admins** table. The admin managing the model.
- **model_name** — Name of the AI model.
- **model_type** — Type of AI model (e.g., classification, regression).
- **model_version** — Version of the AI model.
- **model_path** — Path or location where the model is stored.
- **created_at** — Timestamp of when the model was created.
- **updated_at** — Timestamp of the last update made to the model.

Relationships:

1. **Users ↔ Images**
    - **Relationship**: "Uploads"
    - **Type**: One-to-Many (1:N)
    - **Description**: A user can upload multiple images, but each image is uploaded by exactly one user.
    - **Cardinality**: 1 : N (One User ↔ Many Images)
2. **Images ↔ ProcessingResults**
    - **Relationship**: "Generates Results"
    - **Type**: One-to-One (1:1)
    - **Description**: Each image generates a single processing result.
    - **Cardinality**: 1 : 1 (One Image ↔ One Processing Result)
3. **Users ↔ HelpTickets**
    - **Relationship**: "Creates"
    - **Type**: One-to-Many (1:N)

- **Description**: A user can create multiple help tickets, but each ticket is created by exactly one user.
- **Cardinality**: 1 : N (One User ↔ Many Help Tickets)

4. **Admins ↔ HelpTickets**
    - **Relationship**: "Manages"
    - **Type**: One-to-Many (1:N)
    - **Description**: An admin can manage multiple help tickets, but each ticket can be assigned to only one admin (optional).
    - **Cardinality**: 1 : N (One Admin ↔ Many Help Tickets)
5. **Users ↔ Sessions**
    - **Relationship**: "Has"
    - **Type**: One-to-Many (1:N)
    - **Description**: A user can have multiple login sessions, but each session belongs to exactly one user.
    - **Cardinality**: 1 : N (One User ↔ Many Sessions)
6. **Users ↔ ErrorLogs**
    - **Relationship**: "Has Error Logs"
    - **Type**: One-to-Many (1:N)
    - **Description**: A user can have multiple error logs, but each error log is associated with only one user.
    - **Cardinality**: 1 : N (One User ↔ Many Error Logs)
7. **Images ↔ ErrorLogs**
    - **Relationship**: "Can Cause Errors"
    - **Type**: One-to-Many (1:N)
    - **Description**: An image can cause multiple errors during processing, but each error log can optionally reference one image.
    - **Cardinality**: 1 : N (One Image ↔ Many Error Logs)
8. **Admins ↔ AI Models**
    - **Relationship**: "Manages"
    - **Type**: One-to-Many (1:N)
    - **Description**: An admin can manage multiple AI models, but each AI model is managed by exactly one admin.
    - **Cardinality**: 1 : N (One Admin ↔ Many AI Models)
9. **AI Models ↔ ProcessingResults**
    - **Relationship**: "Generates Results"
    - **Type**: One-to-Many (1:N)
    - **Description**: An AI model can generate multiple processing results, but each processing result is generated by a single AI model.
    - **Cardinality**: 1 : N (One AI Model ↔ Many Processing Results)

10. Images ↔ AI Models

- **Type**: Many-to-Many (N:M)
- **Description**: An image can be processed by multiple AI models, and an AI model can process multiple images. This may require a junction table or foreign keys to represent the relationship.

Chen ERD Diagram Description

- **Entities**: Represented by rectangles with the name of the entity inside. Each entity contains its attributes.
- **Relationships**: Represented by diamonds that connect entities. The cardinalities (1:N, 1:1) are indicated next to the relationship lines.
- **Primary Key (PK)**: Indicated for each entity.
- **Foreign Key (FK)**: Indicated to show relationships with other entities.

Cardinality Type Summary:

1. **Users** ↔ **Images**: One-to-Many (1:N)
2. **Images** ↔ **ProcessingResults**: One-to-One (1:1)
3. **Users** ↔ **HelpTickets**: One-to-Many (1:N)
4. **Admins** ↔ **HelpTickets**: One-to-Many (1:N) 2
5. **Users** ↔ **Sessions**: One-to-Many (1:N)
6. **Users** ↔ **ErrorLogs**: One-to-Many (1:N)
7. **Images** ↔ **ErrorLogs**: One-to-Many (1:N)
8. **Admins** ↔ **AI Models**: One-to-Many (1:N)
9. **AI Models** ↔ **ProcessingResults**: One-to-Many (1:N)
10. **Images** ↔ **AI Models**: Many-to-Many (N:M)