Daily Progress Report

Date: November 13, 2025

Summary of Work

Today's work focused on completing the **Authentication Module** of the Eyeora AI-CCTV Backend system. The goal was to establish a secure user management system integrated with MongoDB, allowing both admin and user login functionalities using JWT authentication and password hashing. The authentication system now provides the foundation for secure role-based access and user data management.

Detailed Progress

- Successfully configured the backend connection to MongoDB using a secure `.env` file and validated the connection via the `settings.py` and `connection.py` scripts.
- Created a modular backend structure for **security**, **database**, and **authentication routes**, each organized into their own folders for scalability.
- Implemented secure password hashing using the `bcrypt` library and integrated JWT-based authentication for user login and token generation.
- Developed and tested the following endpoints through Swagger UI:
  - `/auth/register` – Registers new users (admin or user) and stores hashed passwords in MongoDB.
  - `/auth/login` – Allows login through form-data authentication compliant with OAuth2 standards.
  - `/auth/login_json` – Supports login through JSON payload for API and frontend integration.
  - `/auth/me` – Retrieves user details based on JWT token validation.
- Verified data insertion and retrieval from MongoDB Compass ensuring secure storage of user credentials and roles.

Challenges Faced

Several technical issues were encountered during implementation:

- The initial connection to MongoDB failed due to missing environment variables in the `.env` file. This was resolved by adding all required parameters including `MONGO_URI` and `MONGO_DB`.
- The `email-validator` dependency was missing, causing schema validation errors in Pydantic. This was fixed by installing the package via `pip install pydantic[email]`.
- A major error occurred with the `bcrypt` version mismatch, producing an *AttributeError*. This was resolved by reinstalling the library using `pip install bcrypt --upgrade`.
- Minor import path issues (e.g., `ModuleNotFoundError: repositories`) were corrected by restructuring the project imports and ensuring all submodules contained `__init__.py` files.

Outcome

After successful debugging, the authentication module is now fully functional. The backend securely connects to MongoDB Atlas, supports JWT token-based authentication, and correctly validates password credentials. The system can now handle new user registrations and logins seamlessly, forming the basis for the upcoming modules.

Next Steps

The next phase of development will focus on the **Camera Management Module**. This will include:

- Creating a structured database for camera information including model name, UID, and associated metadata.
- Enabling users to link their cameras by UID, with backend validation against the existing camera inventory.
- Designing an admin-level control system (future Role-Based Access Control) to manage cameras, monitor active users, and perform administrative operations.

Conclusion

In summary, today's session successfully completed the backend authentication infrastructure, integrated secure MongoDB connectivity, and established a reliable login-register workflow. The development environment and codebase are now stable, paving the way for upcoming camera integration and user management enhancements.