Assignment:

FastAPI Assignment Description:
As part of your assignment, you will be required to build a FastAPI application that performs CRUD (Create, Read, Update, Delete) operations for two entities: **Items** and **User Clock-In Records**. You will create a total of **8 APIs** using FastAPI, with specific requirements for GET APIs (including filter options and one MongoDB aggregation). Follow all FastAPI standards for API development, documentation, and error handling.

You will be expected to push your code to **GitHub** and host the application on a **free hosting service** like Koyeb. The **GitHub repository link** and the **hosted Swagger documentation URL** must be provided as part of your submission.

Assignment Requirements:

1. Create a MongoDB Database
- Set up a MongoDB database using either MongoDB Atlas or any local instance.
- Use appropriate collections for the two entities: **Items** and **User Clock-In Records**.

2. CRUD Operations (10 APIs)

A. **Items API**
Create APIs to perform the following operations on **Items**:
- **POST /items**: Create a new item.
- Input: Name, Email, Item Name, Quantity, Expiry Date (YYYY-MM-DD).
- The Insert Date should be automatically inserted when the item is created.
- **GET /items/<id>**: Retrieve an item by ID.
- **GET /items/filter**: Filter items based on:
- Email (exact match).
- Expiry Date (filter items expiring after the provided date).
- Insert Date (filter items inserted after the provided date).
- Quantity (items with quantity greater than or equal to the provided number using a `gte` filter).
- **MongoDB Aggregation**: Aggregate data to return the count of items for each email (grouped by email).
- **DELETE /items/<id>**: Delete an item based on its ID.
- **PUT /items/<id>**: Update an item's details by ID (excluding the Insert Date).

B. **Clock-In Records API**
Create APIs to perform the following operations on **User Clock-In Records**:
- **POST /clock-in**: Create a new clock-in entry.
- Input: Email, Location.
- The Insert DateTime should be automatically added during clock-in.

- **GET /clock-in/<id>**: Retrieve a clock-in record by ID.
- **GET /clock-in/filter**: Filter clock-in records based on:
- Email (exact match).
- Location (exact match).
- Insert DateTime (clock-ins after the provided date).
- **DELETE /clock-in/<id>**: Delete a clock-in record based on its ID.
- **PUT /clock-in/<id>**: Update a clock-in record by ID (excluding Insert DateTime).

Deliverables:
1. **GitHub Repository**: Push the entire project with proper folder structure, code, and readme file to GitHub. The README should include:
- How to set up and run the project locally.
- Endpoints explanation.
2. **Free Hosting**: Host the application on a free service like **Koyeb**, **Heroku**, or similar, and share the hosted application's Swagger URL.
3. **Code Quality**: Follow FastAPI standards, use Pydantic models for request/response bodies, and ensure proper validation.
4. **Swagger Documentation**: Ensure that all APIs are documented automatically using FastAPI's integrated Swagger UI.

Evaluation Criteria:
1. Proper setup and use of MongoDB.
2. Implementation of all CRUD operations.
3. Correct implementation of filters and MongoDB aggregation.
4. Code structure and readability.
5. Hosting on a free platform and providing both GitHub and Swagger links.
6. API documentation and error handling using FastAPI standards.