Assignment: Flask Application for CRUD operations on MongoDB

You need to develop a Flask application that performs CRUD (Create, Read, Update, Delete) operations on a MongoDB database for a User resource using a REST API. The REST API endpoints should be accessible via HTTP requests and tested using Postman.

Requirements

- The application should be developed using Flask and the PyMongo library for MongoDB.
- The application should provide REST API endpoints for CRUD operations on a User resource.
- The User resource should have the following fields:
    - id (a unique identifier for the user)
    - name (the name of the user)
    - email (the email address of the user)
    - password (the password of the user)
- The application should connect to a MongoDB database.
- The application should provide the following REST API endpoints:
    - GET /users - Returns a list of all users.
    - GET /users/<id> - Returns the user with the specified ID.
    - POST /users - Creates a new user with the specified data.
    - PUT /users/<id> - Updates the user with the specified ID with the new data.
    - DELETE /users/<id> - Deletes the user with the specified ID.

Setup

1. Create a new Python virtual environment and activate it.
2. Install Flask and PyMongo libraries using pip.
3. Install Postman for testing the REST API endpoints.
4. Create a new MongoDB database and collection for the application.

Implementation

1. Open the app.py file in your code editor.
2. Import the necessary libraries: Flask, PyMongo, and jsonify.
3. Create a new Flask application instance.
4. Set the MongoDB URI and database name in the Flask application configuration.
5. Create a new PyMongo client and database instance.
6. Create the necessary routes and functions for the REST API endpoints.
7. Run the Flask application using the flask run command.
8. **Using Docker will be highly appreciated**

Testing

1. Open Postman and create a new HTTP request for each of the REST API endpoints.

2. Send requests to the endpoints to test the CRUD operations on the User resource.
3. Verify that the responses are correct and the database is being updated correctly.

Submission

1. Submit the complete code for the Flask application and any additional files needed.
2. Include a README file with instructions on how to set up and run the application.
3. Include screenshots or videos showing the testing process and results.
4. You may send the code in a zip or share the link to github repo.

Good luck with your assignment! Let me know if you have any questions.