Assignment API

Building the api of product and authentication using Nodejs and Express. This api does CRUD operation with products with authentication features using JWT. And created unit test cases using jest and supertest packages.

Prerequisites

Make sure you have the following installed:

1. Nodejs
2. Mongodb

Installation

1. Unzip file in any folder
2. Navigate to the project directory

```
cd your-project-directory
```

3. Install dependencies

```
npm install
```

Database setup

1. Start MongoDB locally or use a remote connection.
2. Open app.js file and replace MongoDB url line number 13

```
connectToMongoDB("mongodb://127.0.0.1:27017/your-database-name")
//(Replace your-database-name with the desired name for your MongoDB database.)
```

Run the Application

1. For development

```
npm run watch
```

2. For production

```
npm start
```

3. Unit testing

```
npm test
```

Note: The application will be available at **http://localhost:8000**

Postman URL to access collection:

https://api.postman.com/collections/3383541-df315133-7806-4cf5-b796-04f06b2dea59?access_key=PMAT-01HJKRRWXV5ZX0B2R0HTK00674

Api Documentation

1. **User**
    a. Endpoint: Post /register

```
Request:
{
    "username":"john123",
    "password":"Abc@123"
}
Response:
{
    "msg": "User registered successfull!",
    "result": {
        "username": "john123",
        "password":
"$2b$10$AonjvnlTPjKVDQwyZ5w/WOSzFCQBjH1r/NfSR0SoGFpKA82OOx6h.",
        "_id": "658b21894e3354c3bfa59995",
        "__v": 0
    }
}
```

    b. Endpoint: POST /login

```
Request:
{
    "username":"john123",
    "password":"Abc@123"
}
Response:
{
    "msg": "Login successful!!"
}
```

2. Product

a. Endpoint: GET /product?orderBy=desc&sortBy=price&size=3&page=2

```
Response
[
    {
        "_id": "658ad0133fe4a5dac97c69dd",
        "name": "product-3",
        "description": "test product 1",
        "price": 200,
        "quantity": 2,
        "__v": 0
    },
    {
        "_id": "658ad045c5ac0e2b841092a8",
        "name": "product-3",
        "description": "test product 1",
        "price": 200,
        "quantity": 2,
        "__v": 0
    }
]
```

b. Endpoint: POST /product

```
Request:
{
  "name": "product-101",
  "description": "test product 101",
  "price": 200,
  "quantity": 2
}
Response:
{
    "name": "product-101",
    "description": "test product 101",
    "price": 200,
    "quantity": 2,
    "_id": "658b23a84e3354c3bfa599a2",
    "__v": 0
}
```

c. Endpoint: Patch /product/:productId

```
Request
{
  "description": "test product 101 - 201",
  "price": 201
}
Response
{
    "msg": "product-4 is updated.",
    "product": {
        "_id": "658ad06ac66faa9f5e35c6af",
        "name": "product-4",
        "description": "test product 101 - 201",
        "price": 201,
        "quantity": 2,
        "__v": 0
    }
}
```

d. Endpoint: GET /api/products/:productId

```
Response:
[
    {
        "_id": "658ad06ac66faa9f5e35c6af",
        "name": "product-4",
        "description": "test product 101 - 201",
        "price": 201,
        "quantity": 2,
        "__v": 0
    }
]
```

e. Endpoint: DELETE /api/products/:productId

```
Response:
{
    "msg": "product-101 deleted from records"
}
```