**Creating API using Flask and Mongo DB**

Requirement Summary:

As per the Client I need to create the Flask based API, which contain below list of action.

Description: As per the assignment i need to create the 5 API endpoints as mentioned below

1. Registration API
2. Login API
3. Product Upload API(CSV file uploads)
4. Product Review API
5. Product view Pagination API

System Requirement:

1. Any windows system or linux system enable virtual Environment.
2. Clone the assignment repository
3. Run the requirement.txt (which contain required package for the current Project)
4. Install Mongo DB and make service active.
5. Once all above done export FLASK\_APP=app.py
6. when run the application execute >> flask run

Project Code Setup Details:

I have Created list of files for normal project setup

1. .env file : This file contains os environment file for the project to set static variables like DBHost, upload file location..etc
2. app.py : This file contains route related info like when ever calls API end points this will route to specific class and functions.
3. main.py : this file contains multiple classes which required for the torget from each route
4. utils.py : This file contain function which helps for main.py
5. Uploads: This folders used to uploding csv files from product uploads

Registration API:

As per the requirement: /users/register [POST] (anonymus access) This Endpoint will be used to create an user with client role receives a post request with name, email, password, gender, phonenumber, role (admin or client) create an user in the users collection with client access.

Solution: Created all given fields, also added some additional fields Token, Created time, last logged in, Delete flag normally created when creating project still not important for current requirement. When we call the Url like <http://localhost:5000/users/register> from postman, this calls app.py app.py route that in the 'Register' Class in main.py file. This will collect the post data change Male to 'M' Female to 'F' and convert the password to encrypted (for security purpose) other fields will be created and updated in the mongo DB More

Find the dev testing logs in below document.

Dev Testing Logs:

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Login API:

As per the requirement : /users/login [POST] (anonymus access) This Endpoint will be used to login a user. Response should contain token and expiration for the token.

Solution: when the user post the username and password, blow action will be performed

URL : <http://localhost:5000/users/login>

1. get username and password query to DB send username and get encrypted password
2. Then de-crypt the password and compare with the input password
3. once match then create Jwt token based on below jwt.encode({"user\_id": username,"exp": datetime.utcnow() + timedelta(minutes=5000)}, app.config['SECRET\_KEY'], algorithm="HS256")
4. And save the token in the user table and also send success response with Token.
5. This token will be saved in the APP for the future API call.

Dev Testing Logs:

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Product Upload API:

As per the requirement : Following endpoints only accessible by logged in users. /products [POST] (admin access) This Endpoint will be used by admin to import products using file upload (refer to attached sample csv file) name: string, barcode: number, brand: string, description: string, price: number, available: Boolean

Solution: User can get the product csv and upload to the below API URL and below list of action will be happen

URL : <http://localhost:5000/products>

1. When user upload files with Token, then only the URL will accept
2. JWT token will be decrypt and get the user name from the bearer token
3. Find the Roles in the user collection for the user name if the role is Admin then only the admin user can upload the product CSV

Dev Testing Logs:

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Table

Description automatically generated

Product Review API:

As per the requirement : /products/review [POST] (client access) This Endpoint will be used to review the product by the client recieives a post request with (barcode, review) create the reviews in the review collection. each review should have (userId, barcode, review, createdAt: Timestamp)

Solution: When user want to give review for the product they can use the below url and below list of action will be happen

URL : <http://localhost:5000/products/review>

1. When user give review with Token, then only the URL will accept
2. JWT token will be decrypt and get the user name from the bearer token
3. Find the Roles in the user collection for the user name if the role is User(instead of client i have given user) then only the client user can update review
4. More Ref : Find the dev testing logs below document .

Dev Testing Logs:

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, application

Description automatically generated

Product View Pagination API:

As per the requirement : This endpoint will be used by client to search for the products recieves a post request with (searchText) and the page in the query response should contain totalCount of the products and products should have maximum of 10 products.

Return products which are available accordingly. response should also contain the latest two reviews of the product. each review should also contain the name of the user who reviewed the product.

Example Response:

{ totalCount: 567, products: [ { name: "Priya Rice Bag (20 kg)" barcode: 34898998, brand: "Priya" description: "Priya Rice Bag (20 kg)" price: 200 available: true, reviews: [{ name: "Adam", review: "Good Product....." }, { name: "Eve", review: "Worth of money...." }] } ] }

Solution:

When user want to view the product by pagination they can use the below url and below list of action will be happen

URL : <http://localhost:5000/products/search?page=1>

1. When user view the product with Token, then only the URL will accept
2. JWT token will be decrypt and get the user name from the bearer token
3. Find the Roles in the user collection for the user name if the role is User can view
4. Code contains to get Total product count, based on search param for page number search field in the POST
5. Get the product list based on search param, in the .env file can define how much product need to display per page
6. Also can define how many review can view in the each product also this will latest review will show first.
7. If we pass page 0 or 1 will show first 10 product based on the .env file 2nd page onward will show next list product, based on our off set value we provide
8. More Ref : Find the dev testing logs below document.

Dev Testing Logs:

Text

Description automatically generated with medium confidence

Timeline

Description automatically generated with low confidence

**Conclusion:**

All 5 Requirement has been completed apart from the coding document related I try to

fulfil as much as possible with in my limited time.