

Task: Develop API Endpoint for Authentication and Product Module

Objective: Create an API endpoint in a Laravel application that retrieves detailed information about a product and formats the response according to the given data structure.

Requirements:

1. Setup:

- Ensure you have a Laravel project setup.
- Use the provided data structure to create the necessary database migrations, models, and relationships.

2. Database Schema: Create migrations for the following tables based on the provided data structure -

- User
- Product
- Brand
- Category
- Unit
- Tax
- Warehous
- ProductQuantity
- Attachment

3. Authentication:

- Develop a authentication with [Laravel/Passport](#).

4. Models and Relationships:

- Define Eloquent models for each entity.
- Establish appropriate relationships (e.g., a product belongs to a brand, category, unit, tax, and has many product quantities and polymorphic relationship with attachments).

5. API Endpoint:

- Create a controller named `ProductController`.
- Implement `CRUD` in `ProductController`.
- The endpoint should be `base_url/api/inventory/product/*`.

6. Response Structure:

- All the API response should match the provided structure.
- Ensure that nested relationships (e.g., product quantities, categories, brands, units, taxes, attachments) are included in the response.
- Handle response for success and error both.

7. Relational Module:

- The relational modules (`product_quantities`, `warehouses`, `categories`, `brands`, `units`, `taxes`) do not need to develop `CRUD`. It has to be generated by using database factory.

8. Sample Response: The response should look like this- JSON

```

1 {
2   "success": true,
3   "status": 200,
4   "message": "Product showing successfully!",
5   "data": {
6     "id": 1,
7     "name": "teer ata",
8     "sku": "TA088362",
9     "symbology": "code 128",
10    "brand_id": 2,
11    "category_id": 3,
12    "unit_id": 3,
13    "price": 100,
14    "qty": 200,
15    "alert_qty": 30,
16    "tax_method": "Inclusive",
17    "tax_id": 2,
18    "has_stock": 1,
19    "has_expired_date": 0,
20    "expired_date": null,
21    "details": "<p>Product Description</p>",
22    "is_active": 1,
23    "created_at": "2024-06-23T08:25:25.000000Z",
24    "updated_at": "2024-06-23T08:26:06.000000Z",
25    "deleted_at": null,
26    "productqties": [
27      {
28        "id": 15,
29        "product_id": 14,
30        "warehouse_id": 6,
31        "qty": 200,
32        "warehouse": {
33          "id": 7,
34          "name": "Ata",
35          "is_active": 1
36        }
37      }
38    ],
39    "categories": {
40      "id": 77,
41      "name": "Ata",
42      "parent_id": null,
43      "is_active": 1
44    },
45    "brands": {
46      "id": 27,
47      "name": "Teer",
48      "is_active": 1
49    },
50    "units": {
51      "id": 3,
52      "name": "piece",
53      "code": "pc",
54      "for": "product-unit",
55      "base_unit": "piece",
56      "operator": "*",
57      "operation_value": "1",
58      "is_active": 1
59    },
60    "taxes": {
61      "id": 2,
62      "name": "@10",
63      "rate": 10,
64      "is_active": 1
65    },
66    "attachments": [
67      {
68        "id": 159,
69        "uploaded_user_id": null,
70        "attachmentable_id": 14,
71        "url": "/storage/app/public/upload/product/attachment/240623082525-8249.jpg",
72        "state": "Product",
73        "label": "attachments",
74        "deleted_at": null,
75        "created_at": "2024-06-23T08:25:26.000000Z",
76        "updated_at": "2024-06-23T08:25:26.000000Z",
77        "file": "240623082525-8249.jpg",
78        "content_type": "jpg",
79        "user": "Undefined User"
80      }
81    ]
82  },
83   "errors": null
84 }

```

Bonus:

- Write reusable code or class base code.
- Implement error handling for cases where the data does not exist.


Submission:


- Provide the GitHub repository link with the implemented feature.
- Include a README.md file with instructions on how to set up and run the project.
- Ensure the code is well-documented and follows Laravel best practices.
- Submission link : <https://forms.gle/Bt2KBiM8KUJFX4mbA>

Duration:

- You must complete your task with in 24 Hours.
- Our total task duration is 72 Hours for short-listed our selected candidates.

This task will test your understanding of Laravel's Eloquent ORM, relationships, API development, and best practices. Good luck!

+880 1620-841623 

vitasoftware@gmail.com 

Corporate office: 677, Brothers Tower, east
Dholaipar, kadamtoli, dhaka-1236 