AuthController API Documentation

Base Route: `/api/auth`

---

1. Register a New User

Endpoint: `POST /api/auth/register` **Description:** Registers a new user.

Request Body:

```
{
    "username": "string",
    "firstName": "string",
    "lastName": "string",
    "email": "string",
    "phoneNumber": "string",
    "country": "string",
    "city": "string",
    "address": "string",
    "password": "string"
}
```

Response:

- `201 Created`: User registered successfully.
- `400 Bad Request`: Username or email is already registered.
- `500 Internal Server Error`: Unexpected error while registering user.

---

2. Login a User

Endpoint: `POST /api/auth/login` **Description:** Authenticates a user and sets authentication cookies.

Request Body:

{

"username": "string",

"password": "string"

}

Response:

- `200 OK`: Login successful.
- `401 Unauthorized`: Invalid username or password.

Notes:

- This endpoint sends `AccessToken` and `RefreshToken` in the header. But they must be assigned to the browser in the frontend.

---

3. Logout a User

Endpoint: `POST /api/auth/logout` **Description:** Logs out the user by clearing authentication cookies.

Response:

- `204 No Content`: Logout successful.

Notes:

- Cookies must be deleted from the frontend manually.

---

4. Refresh Access Token

Endpoint: `POST /api/auth/refresh-token` **Description:** Generates a new access token using a valid refresh token.

Request:

- Requires `RefreshToken` cookie.

Response:

- `200 OK`: Token refreshed successfully.
- `401 Unauthorized`: Missing or invalid refresh token.

---

5. Check Authentication Status

Endpoint: `GET /api/auth/authorize` **Description:** Verifies if the user is authenticated.

Response:

- `200 OK`: User is authenticated.
- `401 Unauthorized`: User is not authenticated.

---

6. Admin-Only Access

Endpoint: `GET /api/auth/admin-only` **Description:** Checks if the user is an admin.

Response:

- `200 OK`: User is an admin.
- `403 Forbidden`: User does not have admin rights.

Notes:

- This endpoint requires the user to have an `Admin` role.

Authentication Mechanism

- Uses JWT-based authentication.
- Access token is stored in an HTTP-only cookie.
- Refresh token allows users to obtain a new access token.

CategoryController API Documentation

Base Route: `/api/categories`

---

1. Get All Categories

Endpoint: `GET /api/categories` **Description:** Retrieves a list of all categories.

Response:

- `200 OK`: Returns a list of category DTOs.

---

2. Get Category by ID

Endpoint: `GET /api/categories/{id}` **Description:** Retrieves a category by its ID.

Response:

- `200 OK`: Returns the requested category.
- `404 Not Found`: Category not found.

---

3. Add a New Category *(Admin Only)*

Endpoint: `POST /api/categories` **Description:** Adds a new category.

Request Body:

{

"name": "string"

}

Response:

- `201 Created`: Category successfully created.
- `400 Bad Request`: Invalid category data.
- `500 Internal Server Error`: Unexpected error while adding category

Notes:

- Requires `Admin` role authorization.

---

4. Update an Existing Category *(Admin Only)*

Endpoint: `PUT /api/categories/{id}` **Description:** Updates an existing category.

Request Body:

{

"name": "string"

}

Response:

- `200 OK`: Category successfully updated.
- `400 Bad Request`: Invalid category data.
- `404 Not Found`: Category not found.

Notes:

- Requires `Admin` role authorization.

---

5. Delete a Category *(Admin Only)*

Endpoint: `DELETE /api/categories/{id}` **Description:** Deletes a category by its ID.

Response:

- `204 No Content`: Category successfully deleted.
- `404 Not Found`: Category not found.

Notes:

- Requires `Admin` role authorization.

OrderController API Documentation

Base Route: `/api/orders`

---

1. Purchase an Item

Endpoint: `POST /api/orders/purchase` **Description:** Places an order for an item.

Request Body:

```
{
  "productId": "int",
  "quantity": "int"
}
```

Response:

- `201 Created`: Order created successfully.
- `400 Bad Request`: Invalid order request.

---

2. Get All Orders

Endpoint: `GET /api/orders` **Description:** Retrieves all orders.

Response:

- `200 OK`: Returns a list of orders.

ProductController API Documentation

Base Route: `/api/products`

---

1. Get All Products

Endpoint: `GET /api/products` **Description:** Retrieves all available products.

Response:

- `200 OK`: Returns a list of products.

---

2. Get Product by ID

Endpoint: `GET /api/products/{id}` **Description:** Retrieves a product by ID.

Response:

- `200 OK`: Returns the requested product.
- `404 Not Found`: Product not found.

---

3. Add a New Product *(Admin Only)*

Endpoint: `POST /api/products` **Description:** Adds a new product.

Response:

- `201 Created`: Product successfully created.
- `400 Bad Request`: Invalid product data.

---

4. Update a Product *(Admin Only)*

Endpoint: `PUT /api/products/{id}` **Description:** Updates an existing product.

Response:

- `200 OK`: Product successfully updated.
- `404 Not Found`: Product not found.

---

5. Delete a Product *(Admin Only)*

Endpoint: `DELETE /api/products/{id}` **Description:** Deletes a product by ID.

Response:

- `204 No Content`: Product successfully deleted.
- `404 Not Found`: Product not found.

UserController API Documentation

Base Route: `/api/user`

---

1. Get All Users *(Admin Only)*

Endpoint: `GET /api/user` **Description:** Retrieves all registered users.

Response:

- `200 OK`: Returns a list users.

---

2. Get User Profile

Endpoint: `GET /api/user/get-profile` **Description:** Gets the logged in user.

Response:

- `200 OK`: Returns DTO of user.
- `401 Unauthorized`: User not signed in.
- `404 Not Found`: User profile not found.

---

3. Update User Profile

Endpoint: `PUT /api/user/update-profile` **Description:** Updates profile information of signed in user.

Response:

- `200 OK`: Product successfully updated.
- `401 Unauthorized`: User not signed in.
- `400 Bad Request`: Altered email or username is already registered, or entered password is incorrect. (see error message for specifics)
- `500 Internal Server Error`: Unexpected error while altering user information.
