# Inventory and Order Management API

A Ruby on Rails API for managing products and orders with business and user authentication.

## Setup Instructions

1. Install Ruby 3.0.0
2. Install PostgreSQL
3. Clone the repository
4. Run bundle install
5. Create and setup the database:

rails db:create

rails db:migrate

1. Start the server:

rails server

## Authentication

### Business Authentication

* POST /api/v1/businesses/signup - Register a new business

{

"business": {

"name": "Business Name",

"email": "business@example.com",

"password": "password123",

"business\_type": "retail"

}

}

* POST /api/v1/businesses/login - Login as a business

{

"email": "business@example.com",

"password": "password123"

}

### User Authentication

* POST /api/v1/users/signup - Register a new user

{

"user": {

"name": "User Name",

"email": "user@example.com",

"password": "password123"

}

}

* POST /api/v1/users/login - Login as a user

{

"email": "user@example.com",

"password": "password123"

}

## API Endpoints

### Products (Business Authentication Required)

* GET /api/v1/products - List all products (paginated)
  + Query parameters:
    - page: Page number (default: 1)
    - per\_page: Items per page (default: 10)
    - query: Search term for name or SKU
* GET /api/v1/products/:id - Get product details
* POST /api/v1/products - Create a new product

{

"product": {

"name": "Product Name",

"sku": "PROD123",

"price": 99.99,

"quantity": 100

}

}

* PUT /api/v1/products/:id - Update a product

{

"product": {

"name": "Updated Name",

"price": 89.99,

"quantity": 50

}

}

* DELETE /api/v1/products/:id - Soft delete a product

### Orders (User Authentication Required)

* GET /api/v1/orders - List user's orders (paginated)
  + Query parameters:
    - page: Page number (default: 1)
    - per\_page: Items per page (default: 10)
* GET /api/v1/orders/:id - Get order details
* POST /api/v1/orders - Create a new order

{

"order": {

"order\_items\_attributes": [

{

"product\_id": 1,

"quantity": 2

},

{

"product\_id": 2,

"quantity": 1

}

]

}

}

## Authentication Headers

After successful login, include the following header in all authenticated requests:

Authorization: Bearer <your\_auth\_token>

## Bug Hunting Task

Review the order management system code and identify potential bugs. Focus on data consistency, business logic, and security. Document each bug with its impact and suggested fix.

### Example Bug

In `OrderService#process\_order`, what happens when two users try to order the last item in stock at the same time? Can you spot the issue in the code below?

```ruby

def process\_order

@order.order\_items.each do |item|

product = item.product

product.update!(quantity: product.quantity - item.quantity)

end

@order.update!(status: 'processing')

end

```

Try to:

1. Identify the problem

2. Explain why it occurs

3. Suggest a solution

4. Consider other similar issues in the codebase