Đoạn mã trên định nghĩa một controller quản lý các hành động liên quan đến giỏ hàng, bao gồm **thêm sản phẩm**, **hiển thị giỏ hàng**, và **cập nhật sản phẩm**. Hãy phân tích chi tiết từng phần.

**1. Biến controller**

let controller = {};

* Biến **controller** là một đối tượng dùng để chứa các phương thức liên quan đến xử lý logic của giỏ hàng.

**2. Phương thức add**

controller.add = async (req, res) => {

let id = isNaN(req.body.id) ? 0 : parseInt(req.body.id);

let quantity = isNaN(req.body.quantity) ? 0 : parseInt(req.body.quantity);

let product = await models.Product.findByPk(id);

if(product){

req.session.cart.add(product,quantity);

}

return res.json({quantity: req.session.cart.quantity});

}

**Chức năng**

* Xử lý yêu cầu thêm sản phẩm vào giỏ hàng.

**Chi tiết**

1. **Lấy thông tin sản phẩm và số lượng từ request**:
   * **req.body.id**: ID của sản phẩm.
   * **req.body.quantity**: Số lượng sản phẩm cần thêm.
   * Kiểm tra nếu ID hoặc số lượng không hợp lệ (NaN), đặt giá trị mặc định là 0.
2. **Tìm sản phẩm trong cơ sở dữ liệu**:
   * **models.Product.findByPk(id)**: Tìm sản phẩm dựa trên **id**.
   * Trả về **product**, nếu không tìm thấy, giá trị là null.
3. **Thêm sản phẩm vào giỏ hàng**:
   * Nếu **product** tồn tại, gọi **req.session.cart.add(product, quantity)** để thêm sản phẩm vào giỏ hàng hiện tại.
4. **Trả về số lượng sản phẩm trong giỏ hàng**:
   * Kết quả là một JSON với số lượng sản phẩm hiện có: **{quantity: req.session.cart.quantity}**.

**3. Phương thức show**

controller.show = (req, res) => {

res.locals.cart = req.session.cart.getCart();

return res.render('cart');

}

**Chức năng**

* Hiển thị giỏ hàng.

**Chi tiết**

1. **Gán thông tin giỏ hàng vào res.locals**:
   * **res.locals.cart**: Chứa dữ liệu giỏ hàng hiện tại, lấy từ phương thức **getCart** của lớp **Cart**.
2. **Render trang cart**:
   * **res.render('cart')**: Hiển thị giao diện giỏ hàng, sử dụng dữ liệu từ **res.locals.cart**.

**4. Phương thức update**

controller.update = async (req, res) => {

let id = isNaN(req.body.id) ? 0 : parseInt(req.body.id);

let quantity = isNaN(req.body.quantity) ? 0 : parseInt(req.body.quantity);

if(quantity > 0){

let updateItem = req.session.cart.update(id,quantity);

return res.json({

item: updateItem,

quantity: req.session.cart.quantity,

subtotal: req.session.cart.subtotal,

total: req.session.cart.total

});

}

res.sendStatus(204).end();

}

**Chức năng**

* Cập nhật số lượng sản phẩm trong giỏ hàng.

**Chi tiết**

1. **Lấy thông tin sản phẩm và số lượng từ request**:
   * **req.body.id**: ID của sản phẩm.
   * **req.body.quantity**: Số lượng mới của sản phẩm.
   * Nếu thông tin không hợp lệ, đặt giá trị mặc định là 0.
2. **Kiểm tra số lượng**:
   * Nếu số lượng lớn hơn 0:
     + Gọi **req.session.cart.update(id, quantity)** để cập nhật giỏ hàng.
     + Trả về JSON với các thông tin:
       - **item**: Thông tin sản phẩm được cập nhật.
       - **quantity**: Tổng số lượng sản phẩm trong giỏ hàng.
       - **subtotal**: Tổng giá trị các sản phẩm.
       - **total**: Tổng giá trị thanh toán (đã tính phí vận chuyển và giảm giá).
3. **Nếu số lượng không hợp lệ**:
   * Trả về mã trạng thái **204 No Content** và kết thúc yêu cầu.

**5. Module Export**

module.exports = controller;

* Xuất đối tượng **controller** để có thể sử dụng trong các tệp khác (ví dụ: router).

**Tóm tắt**

* **add(req, res)**:
  + Xử lý logic thêm sản phẩm vào giỏ hàng, lưu trữ giỏ hàng trong **session**.
  + Trả về số lượng sản phẩm trong giỏ hàng.
* **show(req, res)**:
  + Hiển thị giỏ hàng hiện tại.
* **update(req, res)**:
  + Cập nhật số lượng sản phẩm.
  + Trả về thông tin cập nhật của giỏ hàng hoặc mã trạng thái **204** nếu không hợp lệ.

**Điểm cần cải thiện**

1. **Xử lý lỗi**:
   * Thêm xử lý nếu **Product.findByPk(id)** không trả về sản phẩm.
   * Trả về lỗi chi tiết hơn khi ID hoặc số lượng không hợp lệ.
2. **Bảo mật**:
   * Xác thực người dùng trước khi cho phép thao tác giỏ hàng.
   * Kiểm tra quyền truy cập với từng yêu cầu.