





-  DRONE FOOD DELIVERY SYSTEM - BACKEND DESIGN
 -  TỔNG QUAN HỆ THỐNG
 - Mô tả dự án
 - Kiến trúc hệ thống
 -  YÊU CẦU CHỨC NĂNG
 - 1. Quản lý người dùng
 - 2. Quản lý cửa hàng
 - 3. Đặt hàng và thanh toán
 - 4. Hệ thống drone
 -  THIẾT KẾ CƠ SỞ DỮ LIỆU
 - 1. Mô hình ERD (Entity Relationship Diagram)
 - 2. Lược đồ cơ sở dữ liệu quan hệ
 - 2.1 Bảng USER
 - 2.2 Bảng SHOP
 - 2.3 Bảng ITEM
 - 2.4 Bảng DRONE
 - 2.5 Bảng ORDER
 - 2.6 Bảng SHOP_ORDER
 - 2.7 Bảng SHOP_ORDER_ITEM
 - 2.8 Bảng PAYMENT
 - 3. Chuẩn hóa dữ liệu
 - 3.1 Dạng chuẩn 1NF (First Normal Form)
 - 3.2 Dạng chuẩn 2NF (Second Normal Form)
 - 3.3 Dạng chuẩn 3NF (Third Normal Form)
 - 4. Ràng buộc toàn vẹn
 - 4.1 Ràng buộc khóa chính
 - 4.2 Ràng buộc khóa ngoại
 - 4.3 Ràng buộc miền giá trị
 - 4.4 Ràng buộc duy nhất
 -  API ENDPOINTS
 - 1. Authentication APIs
 - 2. User Management APIs
 - 3. Shop Management APIs
 - 4. Item Management APIs
 - 5. Order Management APIs
 - 6. Drone Management APIs
 - 7. Payment Management APIs

- 8. Admin Management APIs
-  BẢO MẬT VÀ XÁC THỰC
 - 1. JWT Authentication
 - 2. Password Security
 - 3. Role-Based Access Control (RBAC)
 - 4. Input Validation
-  BUSINESS LOGIC
 - 1. Order Status Flow
 - 2. Drone Assignment Logic
 - 3. Payment Processing
 - 4. Notification System
-  DEPLOYMENT VÀ MONITORING
 - 1. Environment Configuration
 - 2. Database Indexing
 - 3. Error Handling
 - 4. Performance Optimization
-  KẾT LUẬN

DRONE FOOD DELIVERY SYSTEM - BACKEND DESIGN



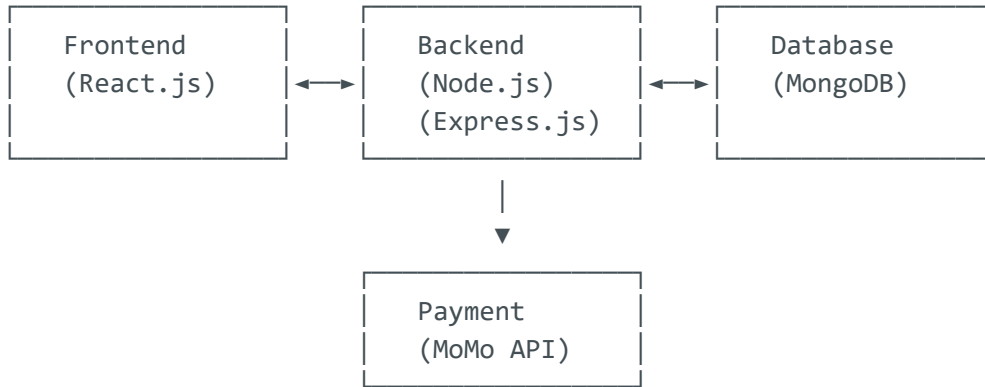
TỔNG QUAN HỆ THỐNG

Mô tả dự án

Hệ thống đặt đồ ăn sử dụng drone để giao hàng, bao gồm:

- **Frontend:** React.js với Redux state management
- **Backend:** Node.js với Express.js và MongoDB
- **Database:** MongoDB với Mongoose ODM
- **Payment:** Tích hợp MoMo Payment Gateway
- **Delivery:** Hệ thống quản lý drone tự động

Kiến trúc hệ thống



YÊU CẦU CHỨC NĂNG

1. Quản lý người dùng

- **Đăng ký/Đăng nhập:** Hỗ trợ 3 loại role (user, owner, admin)
- **Xác thực:** JWT token-based authentication
- **Phân quyền:** Role-based access control (RBAC)

2. Quản lý cửa hàng

- **Tạo/Sửa cửa hàng:** Chủ cửa hàng có thể quản lý thông tin cửa hàng
- **Quản lý sản phẩm:** Thêm/sửa/xóa món ăn
- **Quản lý drone:** Mỗi cửa hàng có 5 drone cố định

3. Đặt hàng và thanh toán

- **Giỏ hàng:** Quản lý sản phẩm trong giỏ hàng
- **Đặt hàng:** Tạo đơn hàng với nhiều cửa hàng
- **Thanh toán:** Hỗ trợ COD và MoMo
- **Theo dõi đơn hàng:** Real-time status updates

4. Hệ thống drone

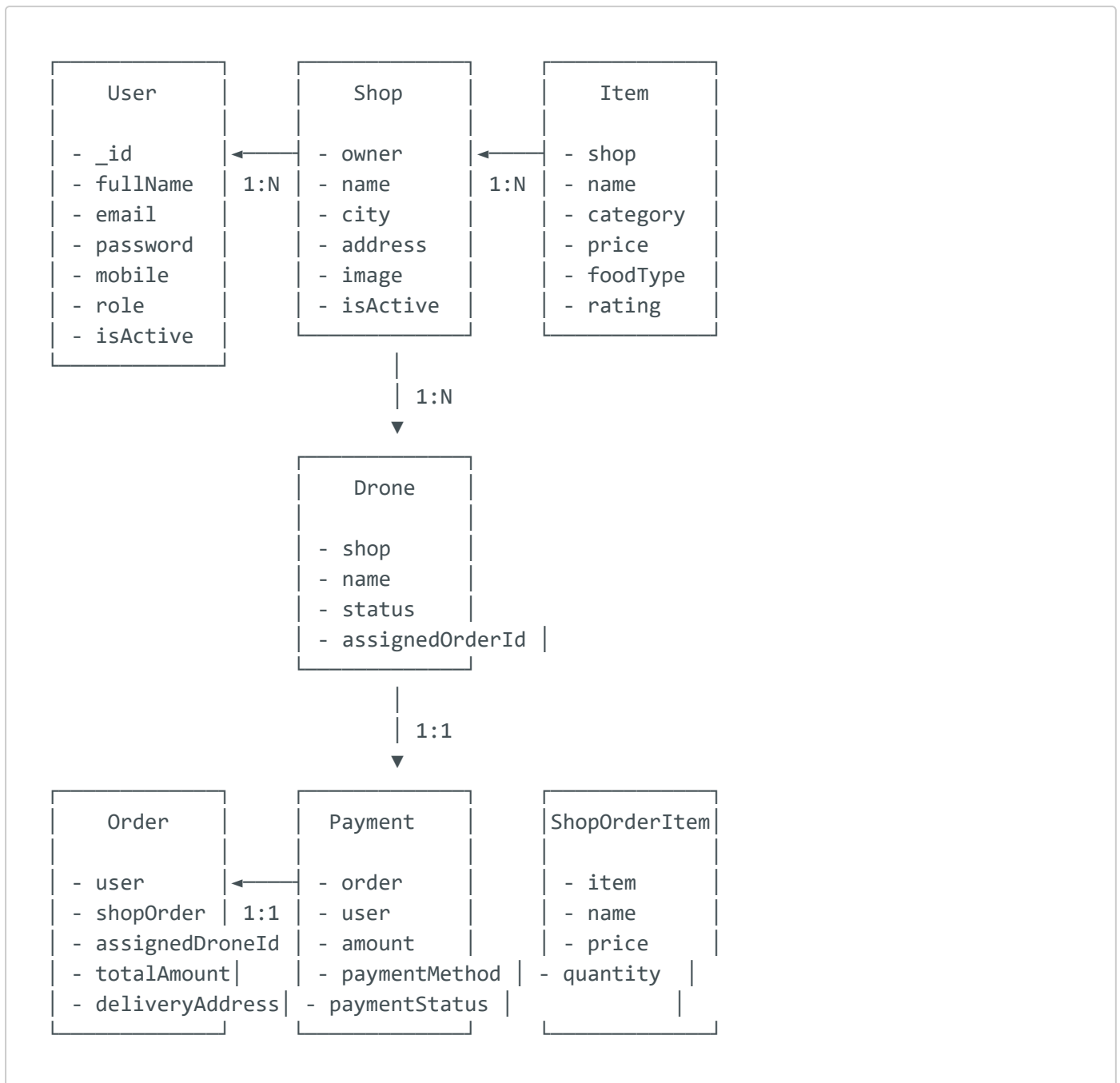
- **Quản lý drone:** 5 drone/cửa hàng với trạng thái khác nhau
- **Phân công drone:** Tự động phân công drone cho đơn hàng

- Theo dõi giao hàng: Customer confirmation system



THIẾT KẾ CƠ SỞ DỮ LIỆU

1. Mô hình ERD (Entity Relationship Diagram)



2. Lược đồ cơ sở dữ liệu quan hệ

Dựa trên nguyên tắc chuyển đổi từ ERD sang mô hình quan hệ, ta có các bảng sau:

2.1 Bảng USER

```

USER (
  _id VARCHAR(24) PRIMARY KEY,
  fullName VARCHAR(255) NOT NULL,
  email VARCHAR(255) UNIQUE NOT NULL,
  password VARCHAR(255) NOT NULL,
  mobile VARCHAR(20) NOT NULL,
  role ENUM('user', 'owner', 'admin') NOT NULL,
  isActive BOOLEAN DEFAULT TRUE,
  createdAt TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
  updatedAt TIMESTAMP DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP
)

```

2.2 Bảng SHOP

```

SHOP (
  _id VARCHAR(24) PRIMARY KEY,
  name VARCHAR(255) NOT NULL,
  image VARCHAR(500) NOT NULL,
  owner VARCHAR(24) NOT NULL,
  city VARCHAR(100) NOT NULL,
  state VARCHAR(100) NOT NULL,
  address VARCHAR(500) NOT NULL,
  isActive BOOLEAN DEFAULT TRUE,
  createdAt TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
  updatedAt TIMESTAMP DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP,
  FOREIGN KEY (owner) REFERENCES USER(_id) ON DELETE CASCADE
)

```

2.3 Bảng ITEM

```

ITEM (
  _id VARCHAR(24) PRIMARY KEY,
  name VARCHAR(255) NOT NULL,
  image VARCHAR(500) NOT NULL,
  shop VARCHAR(24),
  category ENUM('Snack', 'Main course', 'Dessert', 'Beverage', 'Salad', 'Others')
  NOT NULL,
  price DECIMAL(10,2) NOT NULL CHECK (price >= 0),
  foodType ENUM('veg', 'non-veg') NOT NULL,
  rating_average DECIMAL(3,2) DEFAULT 0.00,
  rating_count INT DEFAULT 0,
  createdAt TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
  updatedAt TIMESTAMP DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP,
  FOREIGN KEY (shop) REFERENCES SHOP(_id) ON DELETE SET NULL
)

```

2.4 Bảng DRONE

```
DRONE (  
  _id VARCHAR(24) PRIMARY KEY,  
  shop VARCHAR(24) NOT NULL,  
  name ENUM('Drone-1', 'Drone-2', 'Drone-3', 'Drone-4', 'Drone-5') NOT NULL,  
  status ENUM('Under Maintenance', 'Busy', 'Available') DEFAULT 'Available',  
  assignedOrderId VARCHAR(24),  
  lastAssignedAt TIMESTAMP NULL,  
  maintenanceReason VARCHAR(500),  
  createdAt TIMESTAMP DEFAULT CURRENT_TIMESTAMP,  
  updatedAt TIMESTAMP DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP,  
  FOREIGN KEY (shop) REFERENCES SHOP(_id) ON DELETE CASCADE,  
  FOREIGN KEY (assignedOrderId) REFERENCES ORDER(_id) ON DELETE SET NULL,  
  UNIQUE KEY unique_shop_drone (shop, name)  
)
```

2.5 Bảng ORDER

```
ORDER (  
  _id VARCHAR(24) PRIMARY KEY,  
  user VARCHAR(24),  
  paymentMethod ENUM('cod', 'momo') NOT NULL,  
  paymentStatus ENUM('pending', 'success', 'failed', 'cancelled') DEFAULT  
'pending',  
  deliveryAddress_text VARCHAR(500),  
  deliveryAddress_latitude DECIMAL(10,8),  
  deliveryAddress_longitude DECIMAL(11,8),  
  totalAmount DECIMAL(10,2),  
  sessionId VARCHAR(100),  
  assignedDroneId VARCHAR(24),  
  droneAssignedAt TIMESTAMP NULL,  
  createdAt TIMESTAMP DEFAULT CURRENT_TIMESTAMP,  
  updatedAt TIMESTAMP DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP,  
  FOREIGN KEY (user) REFERENCES USER(_id) ON DELETE SET NULL,  
  FOREIGN KEY (assignedDroneId) REFERENCES DRONE(_id) ON DELETE SET NULL  
)
```

2.6 Bảng SHOP_ORDER

```
SHOP_ORDER (  
  _id VARCHAR(24) PRIMARY KEY,  
  order_id VARCHAR(24) NOT NULL,  
  shop VARCHAR(24),  
  owner VARCHAR(24),  
  subtotal DECIMAL(10,2),  
  status ENUM('pending', 'accepted', 'preparing', 'prepared', 'handed over to  
drone', 'delivering', 'delivered', 'cancelled') DEFAULT 'pending',
```

```

cancelReason VARCHAR(500),
cancelledAt TIMESTAMP NULL,
createdAt TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
updatedAt TIMESTAMP DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP,
FOREIGN KEY (order_id) REFERENCES ORDER(_id) ON DELETE CASCADE,
FOREIGN KEY (shop) REFERENCES SHOP(_id) ON DELETE SET NULL,
FOREIGN KEY (owner) REFERENCES USER(_id) ON DELETE SET NULL
)

```

2.7 Bảng SHOP_ORDER_ITEM

```

SHOP_ORDER_ITEM (
  _id VARCHAR(24) PRIMARY KEY,
  shopOrder_id VARCHAR(24) NOT NULL,
  item VARCHAR(24) NOT NULL,
  name VARCHAR(255),
  price DECIMAL(10,2),
  quantity INT NOT NULL CHECK (quantity > 0),
  createdAt TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
  updatedAt TIMESTAMP DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP,
  FOREIGN KEY (shopOrder_id) REFERENCES SHOP_ORDER(_id) ON DELETE CASCADE,
  FOREIGN KEY (item) REFERENCES ITEM(_id) ON DELETE CASCADE
)

```

2.8 Bảng PAYMENT

```

PAYMENT (
  _id VARCHAR(24) PRIMARY KEY,
  order VARCHAR(24) NOT NULL,
  sessionId VARCHAR(100),
  user VARCHAR(24) NOT NULL,
  paymentMethod ENUM('cod', 'momo') NOT NULL,
  amount DECIMAL(10,2) NOT NULL,
  paymentStatus ENUM('pending', 'success', 'failed', 'cancelled') DEFAULT
'pending',
  momoOrderId VARCHAR(100),
  momoTransactionId VARCHAR(100),
  momoPayUrl VARCHAR(500),
  momoResponse JSON,
  paidAt TIMESTAMP NULL,
  failedAt TIMESTAMP NULL,
  createdAt TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
  updatedAt TIMESTAMP DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP,
  FOREIGN KEY (order) REFERENCES ORDER(_id) ON DELETE CASCADE,
  FOREIGN KEY (user) REFERENCES USER(_id) ON DELETE CASCADE
)

```

3. Chuẩn hóa dữ liệu

3.1 Dạng chuẩn 1NF (First Normal Form)

- ☒ Tất cả thuộc tính đều là nguyên tử (atomic)
- ☒ Không có nhóm lặp lại
- ☒ Mỗi bảng có khóa chính duy nhất

3.2 Dạng chuẩn 2NF (Second Normal Form)

- ☒ Đã đạt 1NF
- ☒ Tất cả thuộc tính không khóa đều phụ thuộc đầy đủ vào khóa chính
- ☒ Loại bỏ phụ thuộc hàm từng phần

3.3 Dạng chuẩn 3NF (Third Normal Form)

- ☒ Đã đạt 2NF
- ☒ Không có phụ thuộc bắc cầu
- ☒ Tất cả thuộc tính không khóa đều phụ thuộc trực tiếp vào khóa chính

4. Ràng buộc toàn vẹn

4.1 Ràng buộc khóa chính

- Mỗi bảng có khóa chính duy nhất
- Khóa chính không được NULL

4.2 Ràng buộc khóa ngoại

- `SHOP.owner` → `USER._id`
- `ITEM.shop` → `SHOP._id`
- `DRONE.shop` → `SHOP._id`
- `DRONE.assignedOrderId` → `ORDER._id`
- `ORDER.user` → `USER._id`
- `ORDER.assignedDroneId` → `DRONE._id`
- `SHOP_ORDER.order_id` → `ORDER._id`
- `SHOP_ORDER.shop` → `SHOP._id`
- `SHOP_ORDER.owner` → `USER._id`

- `SHOP_ORDER_ITEM.shopOrder_id` → `SHOP_ORDER._id`
- `SHOP_ORDER_ITEM.item` → `ITEM._id`
- `PAYMENT.order` → `ORDER._id`
- `PAYMENT.user` → `USER._id`

4.3 Ràng buộc miền giá trị

- `USER.role` ∈ {'user', 'owner', 'admin'}
- `ITEM.category` ∈ {'Snack', 'Main course', 'Dessert', 'Beverage', 'Salad', 'Others'}
- `ITEM.foodType` ∈ {'veg', 'non-veg'}
- `ITEM.price` ≥ 0
- `DRONE.name` ∈ {'Drone-1', 'Drone-2', 'Drone-3', 'Drone-4', 'Drone-5'}
- `DRONE.status` ∈ {'Under Maintenance', 'Busy', 'Available'}
- `ORDER.paymentMethod` ∈ {'cod', 'momo'}
- `ORDER.paymentStatus` ∈ {'pending', 'success', 'failed', 'cancelled'}
- `SHOP_ORDER.status` ∈ {'pending', 'accepted', 'preparing', 'prepared', 'handed over to drone', 'delivering', 'delivered', 'cancelled'}
- `PAYMENT.paymentMethod` ∈ {'cod', 'momo'}
- `PAYMENT.paymentStatus` ∈ {'pending', 'success', 'failed', 'cancelled'}
- `SHOP_ORDER_ITEM.quantity` > 0

4.4 Ràng buộc duy nhất

- `USER.email` UNIQUE
- `DRONE(shop, name)` UNIQUE (mỗi cửa hàng chỉ có 1 drone với tên cụ thể)



API ENDPOINTS

1. Authentication APIs

<code>POST /api/auth/signup</code>	- Đăng ký người dùng
<code>POST /api/auth/signin</code>	- Đăng nhập
<code>POST /api/auth/forgot-password</code>	- Quên mật khẩu
<code>POST /api/auth/reset-password</code>	- Đặt lại mật khẩu

2. User Management APIs

```
GET /api/user/getCurrentUser - Lấy thông tin user hiện tại
PUT /api/user/updateProfile - Cập nhật thông tin cá nhân
GET /api/user/getAllUsers - Lấy danh sách tất cả users (admin)
PUT /api/user/updateUserStatus - Cập nhật trạng thái user (admin)
```

3. Shop Management APIs

```
POST /api/shop/createShop - Tạo cửa hàng mới
GET /api/shop/getMyShop - Lấy thông tin cửa hàng của owner
PUT /api/shop/updateShop - Cập nhật thông tin cửa hàng
GET /api/shop/getShopByCity - Lấy danh sách cửa hàng theo thành phố
GET /api/shop/getAllShops - Lấy tất cả cửa hàng (admin)
```

4. Item Management APIs

```
POST /api/item/addItem - Thêm món ăn mới
GET /api/item/getItemsByCity - Lấy món ăn theo thành phố
PUT /api/item/updateItem - Cập nhật thông tin món ăn
DELETE /api/item/deleteItem - Xóa món ăn
GET /api/item/getItemById - Lấy thông tin món ăn theo ID
```

5. Order Management APIs

```
POST /api/order/createOrder - Tạo đơn hàng mới
GET /api/order/getUserOrders - Lấy đơn hàng của user
GET /api/order/getOwnerOrders - Lấy đơn hàng của owner
PUT /api/order/updateOrderStatus - Cập nhật trạng thái đơn hàng
PUT /api/order/assignDrone - Phân công drone cho đơn hàng
PUT /api/order/confirmReceived - Xác nhận nhận hàng
```

6. Drone Management APIs

```
GET /api/drone/getShopDrones - Lấy danh sách drone của cửa hàng
PUT /api/drone/updateStatus - Cập nhật trạng thái drone
```

PUT /api/drone/assignToOrder - Phân công drone cho đơn hàng
PUT /api/drone/releaseDrone - Giải phóng drone sau khi giao hàng

7. Payment Management APIs

POST /api/payment/createPayment - Tạo thanh toán
POST /api/payment/momoPayment - Thanh toán qua MoMo
GET /api/payment/checkPaymentStatus - Kiểm tra trạng thái thanh toán
PUT /api/payment/updatePaymentStatus - Cập nhật trạng thái thanh toán

8. Admin Management APIs

GET /api/admin/getDashboard - Lấy thống kê tổng quan
GET /api/admin/getAllOrders - Lấy tất cả đơn hàng
GET /api/admin/getAllUsers - Lấy tất cả người dùng
PUT /api/admin/updateUserStatus - Cập nhật trạng thái user



BẢO MẬT VÀ XÁC THỰC

1. JWT Authentication

- **Access Token:** Thời hạn 15 phút
- **Refresh Token:** Thời hạn 7 ngày
- **Secret Key:** Sử dụng biến môi trường

2. Password Security

- **Hashing:** Sử dụng bcrypt với salt rounds = 12
- **Validation:** Minimum 8 characters, có chữ hoa, chữ thường, số

3. Role-Based Access Control (RBAC)

- **User:** Chỉ có thể xem và quản lý đơn hàng của mình

- **Owner:** Quản lý cửa hàng, sản phẩm, drone và đơn hàng của cửa hàng
- **Admin:** Toàn quyền truy cập hệ thống

4. Input Validation

- **Sanitization:** Loại bỏ HTML tags và script
- **Validation:** Kiểm tra kiểu dữ liệu và độ dài
- **Rate Limiting:** Giới hạn số lượng request



BUSINESS LOGIC

1. Order Status Flow

Pending → Accepted → Preparing → Prepared → Handed over to drone → Delivering → Delivered
↓
Cancelled

2. Drone Assignment Logic

- Chỉ drone có status "Available" mới được phân công
- Mỗi drone chỉ phục vụ 1 đơn hàng tại một thời điểm
- Tự động chuyển status drone từ "Available" → "Busy" khi được phân công
- Tự động chuyển status drone từ "Busy" → "Available" khi giao hàng hoàn tất

3. Payment Processing

- **COD:** Thanh toán khi nhận hàng
- **MoMo:** Thanh toán trực tuyến qua MoMo API
- **Multi-order Payment:** Hỗ trợ thanh toán nhiều đơn hàng cùng lúc

4. Notification System

- **Real-time Updates:** Sử dụng WebSocket hoặc Server-Sent Events

- **Email Notifications:** Gửi email cho các sự kiện quan trọng
- **Push Notifications:** Thông báo đẩy cho mobile app



DEPLOYMENT VÀ MONITORING

1. Environment Configuration

```
NODE_ENV=production
PORT=5000
MONGODB_URI=mongodb://localhost:27017/drone-food-delivery
JWT_SECRET=your-secret-key
JWT_REFRESH_SECRET=your-refresh-secret-key
CLOUDINARY_CLOUD_NAME=your-cloudinary-name
CLOUDINARY_API_KEY=your-cloudinary-api-key
CLOUDINARY_API_SECRET=your-cloudinary-api-secret
MOMO_PARTNER_CODE=your-momo-partner-code
MOMO_ACCESS_KEY=your-momo-access-key
MOMO_SECRET_KEY=your-momo-secret-key
MOMO_ENDPOINT=https://test-payment.momo.vn/v2/gateway/api/create
```

2. Database Indexing

```
// User collection indexes
db.users.createIndex({ "email": 1 }, { unique: true })
db.users.createIndex({ "role": 1 })

// Shop collection indexes
db.shops.createIndex({ "owner": 1 })
db.shops.createIndex({ "city": 1 })
db.shops.createIndex({ "isActive": 1 })

// Item collection indexes
db.items.createIndex({ "shop": 1 })
db.items.createIndex({ "category": 1 })
db.items.createIndex({ "foodType": 1 })

// Drone collection indexes
db.drones.createIndex({ "shop": 1, "name": 1 }, { unique: true })
db.drones.createIndex({ "shop": 1, "status": 1 })
db.drones.createIndex({ "assignedOrderId": 1 })

// Order collection indexes
db.orders.createIndex({ "user": 1 })
db.orders.createIndex({ "assignedDroneId": 1 })
db.orders.createIndex({ "paymentStatus": 1 })
```

```
db.orders.createIndex({ "createdAt": -1 })

// Payment collection indexes
db.payments.createIndex({ "order": 1 })
db.payments.createIndex({ "user": 1 })
db.payments.createIndex({ "paymentStatus": 1 })
db.payments.createIndex({ "momoOrderId": 1 })
```

3. Error Handling

- **Global Error Handler:** Xử lý lỗi tập trung
- **Custom Error Classes:** Tạo các loại lỗi cụ thể
- **Logging:** Sử dụng Winston cho logging
- **Error Tracking:** Tích hợp Sentry cho error tracking






4. Performance Optimization

- **Database Connection Pooling:** Sử dụng connection pool
- **Caching:** Redis cho caching dữ liệu thường xuyên truy cập
- **Compression:** Gzip compression cho response
- **Rate Limiting:** Giới hạn số lượng request



KẾT LUẬN

Thiết kế backend này đảm bảo:

-  **Scalability:** Có thể mở rộng dễ dàng
-  **Security:** Bảo mật cao với JWT và RBAC
-  **Performance:** Tối ưu với indexing và caching
-  **Maintainability:** Code structure rõ ràng và dễ bảo trì
-  **Reliability:** Error handling và logging đầy đủ

Hệ thống được thiết kế theo chuẩn RESTful API và tuân thủ các nguyên tắc thiết kế database quan hệ, đảm bảo tính toàn vẹn dữ liệu và hiệu suất cao.