DELODUR POS System - API Documentation

Table of Contents

Overview

The DELODUR POS System API provides RESTful endpoints for managing inventory, sales, users, and system operations. All endpoints return JSON responses and require proper authentication except for login.

API Version: v1.0

Base URL: `http://localhost:5000/api`

Authentication

JWT Token Authentication

All API endpoints (except `/auth/login`) require a valid JWT token in the Authorization header.

```
Authorization: Bearer <your_jwt_token>
```

Token Format

- **Algorithm**: HS256
- **Expiration**: 24 hours
- **Refresh**: Automatic on valid requests

Base URL

Development

```
http://localhost:5000/api
```

Production

```
https://your-domain.com/api
```

Error Handling

Standard Error Response Format

```
{
  "error": true,
  "message": "Error description",
  "code": "ERROR_CODE",
  "details": {}
}
```

HTTP Status Codes

Code	Description
200	Success
201	Created
400	Bad Request

401 Unauthorized
403 Forbidden
404 Not Found
500 Internal Server Error

Common Error Codes

Code	Description
`INVALID_CREDENTIALS`	Wrong username/password
`TOKEN_EXPIRED`	JWT token has expired
`INVALID_TOKEN`	Malformed or invalid token
`PERMISSION_DENIED`	User lacks required permissions
`VALIDATION_ERROR`	Request data validation failed
`NOT_FOUND`	Resource not found
`DUPLICATE_ENTRY`	Resource already exists

Endpoints

Authentication

POST /auth/login

Authenticate user and return JWT token

Request Body:

```
{
  "username": "admin",
  "password": "password"
}
```

Response:

```
{
  "error": false,
  "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...",
  "user": {
    "id": 1,
    "username": "admin",
    "role": "admin",
    "created_at": "2024-01-15T10:30:00Z"
  }
}
```

Error Response:

```
{
  "error": true,
  "message": "Invalid credentials",
  "code": "INVALID_CREDENTIALS"
}
```

Products

GET /products

Get all products with pagination and filtering

Query Parameters:

- `page` (optional): Page number (default: 1)
- `limit` (optional): Items per page (default: 10)
- `search` (optional): Search term for product name/brand
- `brand` (optional): Filter by brand
- `sort` (optional): Sort field (name, brand, created_at)
- `order` (optional): Sort order (asc, desc)

Response:


```
{
  "error": false,
  "data": [
    {
      "id": 1,
      "barcode": "123456789",
      "benz_number": "BN001",
      "brand": "Toyota",
      "alt_number": "ALT001",
      "description": "Oil Filter",
      "application": "Engine Oil Filtration",
      "unit": "piece",
      "reorder_point": 10,
      "location": "Warehouse A",
      "created_at": "2024-01-15T10:30:00Z",
      "updated_at": "2024-01-15T10:30:00Z"
    }
  ],
  "pagination": {
    "page": 1,
    "limit": 10,
    "total": 150,
    "pages": 15
  }
}
```


POST /products

Create new product

Request Body:


```
{
  "barcode": "123456789",
  "benz_number": "BN001",
  "brand": "Toyota",
  "alt_number": "ALT001",
  "description": "Oil Filter",
  "application": "Engine Oil Filtration",
  "unit": "piece",
  "reorder_point": 10,
  "location": "Warehouse A"
}
```


Response:


```
{
  "error": false,
  "message": "Product created successfully",
  "data": {
    "id": 1,
    "barcode": "123456789",
    "brand": "Toyota",
    "description": "Oil Filter",
    "created_at": "2024-01-15T10:30:00Z"
  }
}
```


GET /products/:id

Get product by ID

Response:

```
{
  "error": false,
  "data": {
    "id": 1,
    "barcode": "123456789",
    "benz_number": "BN001",
    "brand": "Toyota",
    "alt_number": "ALT001",
    "description": "Oil Filter",
    "application": "Engine Oil Filtration",
    "unit": "piece",
    "reorder_point": 10,
    "location": "Warehouse A",
    "created_at": "2024-01-15T10:30:00Z",
    "updated_at": "2024-01-15T10:30:00Z",
    "stock_items": [
      {
        "id": 1,
        "color_code": "BLK",
        "remarks": "Black variant",
        "cost": 80.00,
        "selling_price": 100.00,
        "quantity": 50,
        "currency": "USD"
      }
    ]
  }
}
```


PUT /products/:id

Update product

Request Body:


```
{
  "brand": "Toyota Updated",
  "description": "Updated Oil Filter",
  "reorder_point": 15
}
```


Response:


```
{
  "error": false,
  "message": "Product updated successfully",
  "data": {
    "id": 1,
    "brand": "Toyota Updated",
    "description": "Updated Oil Filter",
    "updated_at": "2024-01-15T11:30:00Z"
  }
}
```


DELETE /products/:id

Delete product

Response:


```
{
  "error": false,
  "message": "Product deleted successfully"
}
```

Stock Items

GET /stock

Get stock items with filtering

Query Parameters:

- `product_id` (optional): Filter by product ID
- `low_stock` (optional): Show only low stock items (boolean)
- `search` (optional): Search term
- `page` (optional): Page number
- `limit` (optional): Items per page

Response:


```
{
  "error": false,
  "data": [
    {
      "id": 1,
      "product_id": 1,
      "product_name": "Oil Filter",
      "product_brand": "Toyota",
      "color_code": "BLK",
      "remarks": "Black variant",
      "cost": 80.00,
      "selling_price": 100.00,
      "quantity": 50,
      "currency": "USD",
      "fc_cost": 80.00,
      "conversion_rate": 1.0000,
      "created_at": "2024-01-15T10:30:00Z"
    }
  ],
  "pagination": {
    "page": 1,
    "limit": 10,
    "total": 75,
    "pages": 8
  }
}
```


POST /stock

Create new stock item

Request Body:


```
{
  "product_id": 1,
  "color_code": "BLK",
  "remarks": "Black variant",
  "cost": 80.00,
  "selling_price": 100.00,
  "quantity": 50,
  "currency": "USD",
  "fc_cost": 80.00,
  "conversion_rate": 1.0000
}
```


Response:


```
{
  "error": false,
  "message": "Stock item created successfully",
  "data": {
    "id": 1,
    "product_id": 1,
    "color_code": "BLK",
    "quantity": 50,
    "created_at": "2024-01-15T10:30:00Z"
  }
}
```

PUT /stock/:id

Update stock item

Request Body:

```
{
  "quantity": 45,
  "selling_price": 110.00
}
```

Response:

```
{
  "error": false,
  "message": "Stock item updated successfully",
  "data": {
    "id": 1,
    "quantity": 45,
    "selling_price": 110.00,
    "updated_at": "2024-01-15T11:30:00Z"
  }
}
```

Sales

GET /sales

Get sales history with filtering

Query Parameters:

- `start_date` (optional): Start date (YYYY-MM-DD)
- `end_date` (optional): End date (YYYY-MM-DD)
- `customer` (optional): Filter by customer name
- `page` (optional): Page number
- `limit` (optional): Items per page

Response:

```
{
  "error": false,
  "data": [
    {
      "id": 1,
      "date": "2024-01-15",
      "receipt_number": "RCP-2024-001",
      "customer": "John Doe",
      "product_name": "Oil Filter",
      "product_brand": "Toyota",
      "quantity": 2,
      "selling_price": 100.00,
      "total_amount": 200.00,
      "tax_amount": 24.00,
      "created_at": "2024-01-15T10:30:00Z"
    }
  ],
  "pagination": {
    "page": 1,
    "limit": 10,
    "total": 250,
    "pages": 25
  }
}
```

POST /sales

Create new sale transaction

Request Body:

```
{
  "customer": "John Doe",
  "items": [
    {
      "product_id": 1,
      "stock_item_id": 1,
      "quantity": 2,
      "selling_price": 100.00
    }
  ]
}
```

Response:

```
{
  "error": false,
  "message": "Sale completed successfully",
  "data": {
    "receipt_number": "RCP-2024-001",
    "total_amount": 200.00,
    "tax_amount": 24.00,
    "grand_total": 224.00,
    "items": [
      {
        "product_name": "Oil Filter",
        "quantity": 2,
        "selling_price": 100.00,
        "subtotal": 200.00
      }
    ]
  }
}
```

GET /sales/:receipt_number

Get sale by receipt number

Response:

```
{
  "error": false,
  "data": {
    "receipt_number": "RCP-2024-001",
    "date": "2024-01-15",
    "customer": "John Doe",
    "total_amount": 200.00,
    "tax_amount": 24.00,
    "grand_total": 224.00,
    "created_at": "2024-01-15T10:30:00Z",
    "items": [
      {
        "product_name": "Oil Filter",
        "product_brand": "Toyota",
        "quantity": 2,
        "selling_price": 100.00,
        "subtotal": 200.00
      }
    ]
  }
}
```

Suppliers

GET /suppliers

Get all suppliers

Query Parameters:

- `search` (optional): Search term
- `page` (optional): Page number
- `limit` (optional): Items per page

Response:

```
{
  "error": false,
  "data": [
    {
      "id": 1,
      "code": "SUP001",
      "name": "Toyota Parts Supplier",
      "address": "123 Supplier St, Manila",
      "phone": "+63 912 345 6789",
      "email": "contact@toyotaparts.com",
      "created_at": "2024-01-15T10:30:00Z"
    }
  ],
  "pagination": {
    "page": 1,
    "limit": 10,
    "total": 25,
    "pages": 3
  }
}
```

POST /suppliers

Create new supplier

Request Body:

```
{
  "code": "SUP001",
  "name": "Toyota Parts Supplier",
  "address": "123 Supplier St, Manila",
  "phone": "+63 912 345 6789",
  "email": "contact@toyotaparts.com"
}
```

Response:

```
{
  "error": false,
  "message": "Supplier created successfully",
  "data": {
    "id": 1,
    "code": "SUP001",
    "name": "Toyota Parts Supplier",
    "created_at": "2024-01-15T10:30:00Z"
  }
}
```

Incoming Stock

GET /stock/incoming

Get incoming stock records

Query Parameters:

- `start_date` (optional): Start date (YYYY-MM-DD)
- `end_date` (optional): End date (YYYY-MM-DD)
- `supplier_id` (optional): Filter by supplier
- `status` (optional): Filter by status (new, posted)
- `page` (optional): Page number
- `limit` (optional): Items per page

Response:

```
{
  "error": false,
  "data": [
    {
      "id": 1,
      "date": "2024-01-15",
      "reference": "PO-001",
      "supplier_name": "Toyota Parts Supplier",
      "product_name": "Oil Filter",
      "quantity": 50,
      "cost": 80.00,
      "selling_price": 100.00,
      "status": "posted",
      "created_at": "2024-01-15T10:30:00Z"
    }
  ],
  "pagination": {
    "page": 1,
    "limit": 10,
    "total": 45,
    "pages": 5
  }
}
```


POST /stock/incoming

Add incoming stock

Request Body:


```
{
  "date": "2024-01-15",
  "reference": "PO-001",
  "supplier_id": 1,
  "items": [
    {
      "product_id": 1,
      "stock_item_id": 1,
      "quantity": 50,
      "cost": 80.00,
      "selling_price": 100.00,
      "currency": "USD",
      "fc_cost": 80.00,
      "conversion_rate": 1.0000,
      "document_ref": "INV-001"
    }
  ]
}
```


Response:


```
{
  "error": false,
  "message": "Incoming stock added successfully",
  "data": {
    "id": 1,
    "reference": "PO-001",
    "total_items": 1,
    "total_quantity": 50,
    "created_at": "2024-01-15T10:30:00Z"
  }
}
```

Dashboard

GET /dashboard

Get dashboard statistics

Response:


```
{
  "error": false,
  "data": {
    "total_products": 150,
    "total_stock_items": 450,
    "total_sales_today": 25,
    "inventory_value": 150000.00,
    "low_stock_items": 12,
    "recent_sales": [
      {
        "receipt_number": "RCP-2024-001",
        "customer": "John Doe",
        "total_amount": 224.00,
        "created_at": "2024-01-15T10:30:00Z"
      }
    ],
    "top_products": [
      {
        "product_name": "Oil Filter",
        "brand": "Toyota",
        "total_sales": 150,
        "revenue": 15000.00
      }
    ]
  }
}
```

Reports

GET /reports/sales

Generate sales report

Query Parameters:

- `start_date` (required): Start date (YYYY-MM-DD)
- `end_date` (required): End date (YYYY-MM-DD)
- `group_by` (optional): Group by (day, week, month, product)

Response:

```
{
  "error": false,
  "data": {
    "period": {
      "start_date": "2024-01-01",
      "end_date": "2024-01-31"
    },
    "summary": {
      "total_sales": 1250,
      "total_revenue": 125000.00,
      "total_tax": 15000.00,
      "average_order_value": 100.00
    },
    "daily_data": [
      {
        "date": "2024-01-01",
        "sales_count": 45,
        "revenue": 4500.00,
        "tax": 540.00
      }
    ],
    "top_products": [
      {
        "product_name": "Oil Filter",
        "brand": "Toyota",
        "quantity_sold": 150,
        "revenue": 15000.00
      }
    ]
  }
}
```


GET /reports/inventory

Generate inventory report

Response:


```
{
  "error": false,
  "data": {
    "total_items": 450,
    "total_value": 150000.00,
    "low_stock_items": 12,
    "out_of_stock_items": 3,
    "stock_by_location": [
      {
        "location": "Warehouse A",
        "item_count": 250,
        "total_value": 85000.00
      }
    ],
    "stock_by_brand": [
      {
        "brand": "Toyota",
        "item_count": 200,
        "total_value": 75000.00
      }
    ]
  }
}
```

Users

GET /users

Get all users (Admin only)

Response:

```
{
  "error": false,
  "data": [
    {
      "id": 1,
      "username": "admin",
      "role": "admin",
      "created_at": "2024-01-15T10:30:00Z"
    }
  ]
}
```

POST /users

Create new user (Admin only)

Request Body:

```
{
  "username": "cashier1",
  "password": "secure_password",
  "role": "user"
}
```

Response:

```
{
  "error": false,
  "message": "User created successfully",
  "data": {
    "id": 2,
    "username": "cashier1",
    "role": "user",
    "created_at": "2024-01-15T10:30:00Z"
  }
}
```

Data Models

Product Model

```
{
  "id": "integer (auto-increment)",
  "barcode": "string (unique)",
  "benz_number": "string",
  "brand": "string (required)",
  "alt_number": "string",
  "description": "text",
  "application": "text",
  "unit": "string",
  "reorder_point": "integer (default: 0)",
  "location": "string",
  "created_at": "timestamp",
  "updated_at": "timestamp"
}
```

Stock Item Model

```
{
  "id": "integer (auto-increment)",
  "product_id": "integer (foreign key)",
  "color_code": "string",
  "remarks": "string",
  "cost": "decimal(10,2) (default: 0)",
  "selling_price": "decimal(10,2) (default: 0)",
  "quantity": "integer (default: 0)",
  "currency": "string (default: 'USD')",
  "fc_cost": "decimal(10,2) (default: 0)",
  "conversion_rate": "decimal(10,4) (default: 1)",
  "created_at": "timestamp"
}
```

Sales History Model

```
{
  "id": "integer (auto-increment)",
  "date": "date (required)",
  "receipt_number": "string (required)",
  "customer": "string",
  "product_id": "integer (foreign key)",
  "stock_item_id": "integer (foreign key)",
  "quantity": "integer (required)",
  "selling_price": "decimal(10,2)",
  "total_amount": "decimal(10,2)",
  "tax_amount": "decimal(10,2)",
  "created_at": "timestamp"
}
```

Supplier Model

```
{
  "id": "integer (auto-increment)",
  "code": "string (unique, required)",
  "name": "string (required)",
  "address": "text",
  "phone": "string",
  "email": "string",
  "created_at": "timestamp"
}
```

Examples

Complete Sales Flow

1. Login

```
curl -X POST http://localhost:5000/api/auth/login \
  -H "Content-Type: application/json" \
  -d '{
    "username": "admin",
    "password": "password"
  }'
```

2. Search Products

```
curl -X GET "http://localhost:5000/api/products?search=oil&limit=5" \
  -H "Authorization: Bearer YOUR_JWT_TOKEN"
```

3. Create Sale

```
curl -X POST http://localhost:5000/api/sales \
  -H "Authorization: Bearer YOUR_JWT_TOKEN" \
  -H "Content-Type: application/json" \
  -d '{
    "customer": "John Doe",
    "items": [
      {
        "product_id": 1,
        "stock_item_id": 1,
        "quantity": 2,
        "selling_price": 100.00
      }
    ]
  }'
```

4. Get Sales Report

```
curl -X GET "http://localhost:5000/api/reports/sales?start_date=2024-01-01&end_date=2024-01-31" \
  -H "Authorization: Bearer YOUR_JWT_TOKEN"
```

Error Handling Example

Invalid Token

```
curl -X GET http://localhost:5000/api/products \
  -H "Authorization: Bearer INVALID_TOKEN"
```

Response:

```
{
  "error": true,
  "message": "Invalid token",
  "code": "INVALID_TOKEN"
}
```

Validation Error

```
curl -X POST http://localhost:5000/api/products \
  -H "Authorization: Bearer YOUR_JWT_TOKEN" \
  -H "Content-Type: application/json" \
  -d '{
    "barcode": "123456789"
  }'
```

Response:

```
{
  "error": true,
  "message": "Brand is required",
  "code": "VALIDATION_ERROR",
  "details": {
    "brand": "Brand field is required"
  }
}
```

🔧 Rate Limiting

The API implements rate limiting to prevent abuse:

- **Rate Limit**: 1000 requests per 15 minutes per IP
- **Headers**: Rate limit information included in response headers
- **Exceeded Response**: 429 Too Many Requests with retry-after header

Rate Limit Headers

```
X-RateLimit-Limit: 1000
X-RateLimit-Remaining: 999
X-RateLimit-Reset: 1642234567
```

Security

Authentication

- JWT tokens with 24-hour expiration
- Secure password hashing with bcrypt
- Role-based access control

API Security

- Rate limiting to prevent abuse
- Input validation and sanitization
- SQL injection protection
- CORS configuration
- Security headers (Helmet)

Best Practices

1. Always use HTTPS in production
2. Store JWT tokens securely
3. Implement proper error handling
4. Validate all input data
5. Use environment variables for sensitive data

This API documentation is maintained by the development team and should be updated with each system release.