Fresh Roots Backend API Documentation

🌱 Complete API Reference for Frontend Development

Base URL (Local): `http://localhost:3000`
Base URL (Production): Will be provided after deployment
API Prefix: All endpoints are prefixed with `/api`

📋 Table of Contents

🔐 Authentication

Register New User

```
POST /api/auth/register
Content-Type: application/json

{
  "name": "John Doe",
  "email": "john@example.com",
  "password": "SecurePass123",
  "phone": "+23054321234"
}

Response 201:
{
  "success": true,
  "message": "User registered successfully",
  "data": {
    "user": {
      "id": "uuid",
      "name": "John Doe",
      "email": "john@example.com",
      "role": "user"
    },
    "tokens": {
      "access_token": "jwt...",
      "refresh_token": "jwt..."
    }
  }
}
```

Login

```
POST /api/auth/login
Content-Type: application/json

{
  "email": "john@example.com",
  "password": "SecurePass123"
}

Response 200:
{
  "success": true,
  "data": {
    "user": { "id": "uuid", "name": "John Doe", "email": "john@example.com", "role": "user" },
    "tokens": {
      "access_token": "jwt...",
      "refresh_token": "jwt..."
    }
  }
}
```

Refresh Token

```
POST /api/auth/refresh
Content-Type: application/json

{
  "refresh_token": "jwt..."
}

Response 200:
{
  "success": true,
  "data": {
    "access_token": "new_jwt...",
    "refresh_token": "new_refresh_jwt..."
  }
}
```

Get Current User

```
GET /api/auth/me
Authorization: Bearer {access_token}

Response 200:
{
  "success": true,
  "data": {
    "user": {
      "id": "uuid",
      "name": "John Doe",
      "email": "john@example.com",
      "phone": "+23054321234",
      "role": "user",
      "created_at": "2026-01-23T..."
    }
  }
}
```

👤 User Management

All authenticated endpoints require:

```
Authorization: Bearer {access_token}
```

📂 Categories

Get All Categories

```
GET /api/categories

Response 200:
{
  "success": true,
  "data": [
    {
      "id": "uuid",
      "name": "Légumes Frais",
      "slug": "legumes-frais",
      "description": "Fresh vegetables",
      "icon": "🥬",
      "created_at": "2026-01-23T..."
    }
  ]
}
```

Create Category (Admin Only)

```
POST /api/categories
Authorization: Bearer {admin_access_token}
Content-Type: application/json

{
  "name": "Fruits",
  "description": "Fresh fruits from Mauritius",
  "icon": "🍎"
}

Response 201: Category created
```

🛒 Listings (Products)

Browse Listings

```
GET /api/listings?page=1&limit=20&search=bredes&category=legumes-frais&sortBy=price&sortOrder=asc

Query Parameters:
- page (optional, default: 1)
- limit (optional, default: 20, max: 100)
- search (optional): Search in title/description
- category (optional): Filter by category slug
- sortBy (optional): price | created_at | title
- sortOrder (optional): asc | desc

Response 200:
{
  "success": true,
  "data": {
    "listings": [
      {
        "id": "uuid",
        "title": "Brèdes Cresson",
        "description": "Fresh watercress from local farms",
        "price": 45,
        "unit": "bunch",
        "stock": 50,
        "category": { "name": "Légumes Frais", "slug": "legumes-frais" },
        "location": "Curepipe",
        "tags": ["organic", "fresh", "local"],
        "images": [
          { "image_url": "https://www.caneyforkfarms.com/cdn/shop/files/10_c5e4f5ac-dcd8-4b05-a792-fa19d9769a73.png?v=1743650920&width=1445", "is_primary": true }
        ],
        "is_active": true,
        "created_at": "2026-01-23T..."
      }
    ],
    "pagination": {
      "total": 15,
      "page": 1,
      "limit": 20,
      "totalPages": 1
    }
  }
}
```

Get Single Listing

```
GET /api/listings/{id}

Response 200:
{
  "success": true,
  "data": {
    "listing": {
      "id": "uuid",
      "title": "Brèdes Cresson",
      "description": "Fresh watercress...",
      "price": 45,
      "unit": "bunch",
      "stock": 50,
      "category": { "name": "Légumes Frais", "slug": "legumes-frais" },
      "images": [...],
      "admin": { "name": "Yash Beeharry", "email": "yashbeeharry363@gmail.com" }
    }
  }
}
```

Create Listing (Admin Only)

```
POST /api/listings
Authorization: Bearer {admin_access_token}
Content-Type: application/json

{
  "title": "Fresh Tomatoes",
  "description": "Ripe red tomatoes from Moka",
  "price": 80,
  "unit": "kg",
  "stock": 100,
  "category_id": "uuid",
  "location": "Moka",
  "tags": ["organic", "fresh"],
  "images": [
    { "image_url": "http://mokaorigins.com/cdn/shop/articles/5th-year-collage.jpg?
v=1641841930", "is_primary": true }
  ]
}

Response 201: Listing created
```

Update Listing (Admin Only)

```
PATCH /api/listings/{id}
Authorization: Bearer {admin_access_token}
Content-Type: application/json

{
  "price": 75,
  "stock": 80
}

Response 200: Listing updated
```

Delete Listing (Admin Only)

```
DELETE /api/listings/{id}
Authorization: Bearer {admin_access_token}

Response 200: Listing deleted
```

💚 Interest Expressions

Express Interest in a Product

```
POST /api/interest
Authorization: Bearer {access_token}
Content-Type: application/json

{
  "listing_id": "uuid",
  "message": "Ki pri pou 5kg bredes? I want bulk order."
}

Response 201:
{
  "success": true,
  "message": "Interest expressed successfully. Admin will contact you soon.",
  "data": {
    "interest": {
      "id": "uuid",
      "listing_id": "uuid",
      "message": "Ki pri pou 5kg bredes?",
      "status": "pending",
      "created_at": "2026-01-23T..."
    }
  }
}
```

Get My Interest Expressions

```
GET /api/interest/my-interests?page=1&limit=20
Authorization: Bearer {access_token}

Response 200:
{
  "success": true,
  "data": {
    "interests": [
      {
        "id": "uuid",
        "listing": { "title": "Brèdes Cresson", "price": 45 },
        "message": "Ki pri pou 5kg?",
        "status": "contacted",
        "created_at": "2026-01-23T..."
      }
    ],
    "pagination": {...}
  }
}
```

Get All Interest Expressions (Admin Only)

```
GET /api/interest/admin/all?page=1&limit=20&status=pending
Authorization: Bearer {admin_access_token}

Response 200: List of all interest expressions with user details
```

Update Interest Status (Admin Only)

```
PATCH /api/interest/{id}
Authorization: Bearer {admin_access_token}
Content-Type: application/json

{
  "status": "contacted"
}

Response 200: Interest status updated
```

🛍️ Orders

Create Order

```
POST /api/orders
Authorization: Bearer {access_token}
Content-Type: application/json

{
  "items": [
    {
      "listing_id": "uuid",
      "quantity": 2
    },
    {
      "listing_id": "uuid2",
      "quantity": 5
    }
  ],
  "payment_method": "cash_on_delivery"
}

Response 201:
{
  "success": true,
  "message": "Order created successfully",
  "data": {
    "order": {
      "id": "uuid",
      "order_number": "ORD-20260123-001",
      "total_amount": 450,
      "payment_method": "cash_on_delivery",
      "payment_status": "pending",
      "order_status": "pending",
      "items": [
        {
          "listing": { "title": "Brèdes Cresson" },
          "quantity": 2,
          "unit_price": 45,
          "subtotal": 90
        }
      ],
      "created_at": "2026-01-23T..."
    }
  }
}
```

Get My Orders

```
GET /api/orders/my-orders?page=1&limit=20
Authorization: Bearer {access_token}

Response 200:
{
  "success": true,
  "data": {
    "orders": [...],
    "pagination": {...}
  }
}
```

Get Single Order

```
GET /api/orders/{id}
Authorization: Bearer {access_token}

Response 200: Full order details with items
```

Get All Orders (Admin Only)

```
GET /api/orders/admin/all?page=1&limit=20&status=pending
Authorization: Bearer {admin_access_token}

Query Parameters:
- page, limit
- status: pending | approved | completed | cancelled
- payment_status: pending | completed | failed

Response 200: List of all orders
```

Update Order Status (Admin Only)

```
PATCH /api/orders/{id}/status
Authorization: Bearer {admin_access_token}
Content-Type: application/json

{
  "order_status": "approved",
  "notes": "Order approved. Will deliver tomorrow."
}

Response 200: Order status updated
```

💳 Payments

Initiate Payment

```
POST /api/payments/initiate
Authorization: Bearer {access_token}
Content-Type: application/json

{
  "orderId": 1,
  "amount": 450,
  "currency": "MUR",
  "customerEmail": "john@example.com",
  "customerPhone": "+23054321234",
  "returnUrl": "myapp://payment/success",
  "cancelUrl": "myapp://payment/cancel"
}

Response 200:
{
  "success": true,
  "transaction_id": "SIM-1234567890",
  "message": "Development mode: Payment simulation. In production, customer would be redirected to Juice app."
}

Note: Currently in dev mode. With MIPS configured, will return paymentUrl for Juice payment.
```

Get Payment Status

```
GET /api/payments/status/{transactionId}
Authorization: Bearer {access_token}

Response 200:
{
  "success": true,
  "data": {
    "transaction_id": "SIM-1234567890",
    "status": "completed",
    "amount": 450
  }
}
```

📊 Admin Dashboard

Get Dashboard Overview

```
GET /api/admin/dashboard/overview
Authorization: Bearer {admin_access_token}

Response 200:
{
  "success": true,
  "data": {
    "metrics": {
      "totalUsers": 150,
      "totalOrders": 89,
      "totalRevenue": 12450,
      "totalListings": 15,
      "pendingOrders": 5,
      "lowStockItems": 3
    },
    "recentOrders": [
      {
        "id": "uuid",
        "order_number": "ORD-20260123-001",
        "customer": "John Doe",
        "customer_email": "john@example.com",
        "total_amount": 450,
        "payment_status": "pending",
        "order_status": "pending",
        "items_count": 3,
        "created_at": "2026-01-23T..."
      }
    ],
    "popularProducts": [
      {
        "listing_id": "uuid",
        "title": "Brèdes Cresson",
        "total_sold": 150,
        "order_count": 45,
        "image": "https://upload.wikimedia.org/wikipedia/commons/d/dd/Water-
cress_%282%29.JPG"
      }
    ]
  }
}
```

Get Statistics

```
GET /api/admin/dashboard/stats?period=30d
Authorization: Bearer {admin_access_token}

Query Parameters:
- period: 7d | 30d | 90d | all

Response 200:
{
  "success": true,
  "data": {
    "period": "30d",
    "periodDays": 30,
    "stats": {
      "newUsers": 25,
      "newOrders": 45,
      "totalRevenue": 8500,
      "averageOrderValue": 188.89,
      "interestExpressions": 32
    },
    "growth": {
      "orders": "15.5",
      "revenue": "22.3"
    }
  }
}
```

📈 Admin Analytics

Get Sales Analytics

```
GET /api/admin/analytics/sales?period=30d
Authorization: Bearer {admin_access_token}

Response 200:
{
  "success": true,
  "data": {
    "period": "30d",
    "chartData": [
      { "date": "2026-01-01", "revenue": 450, "orders": 5 },
      { "date": "2026-01-02", "revenue": 680, "orders": 8 }
    ],
    "summary": {
      "totalRevenue": 8500,
      "totalOrders": 45,
      "averageDaily": 283.33
    }
  }
}
```

Get Product Analytics

```
GET /api/admin/analytics/products
Authorization: Bearer {admin_access_token}

Response 200:
{
  "success": true,
  "data": {
    "topSelling": [
      {
        "listing_id": "uuid",
        "title": "Brèdes Cresson",
        "price": 45,
        "total_sold": 150,
        "revenue": 6750,
        "orders": 45,
        "image": "https://i.ytimg.com/vi/oGd-HvvLPOA/hq720.jpg?sqp=-oaymwEhCK4-
FEIIDSFryq4qpAxMIARUAAAAAGAElAADIQj0AgKJD&rs=AOn4CLClnfD9mP1uC4yjcI7R1D1reNHKUg"
      }
    ],
    "lowStock": [
      { "id": "uuid", "title": "Lalo", "stock": 8, "price": 35 }
    ],
    "outOfStock": 2,
    "revenueByCategory": [
      { "category": "Légumes Frais", "revenue": 5400 }
    ]
  }
}
```

👥 Admin User Management

Get All Users

```
GET /api/admin/users?page=1&limit=20&search=john&role=user
Authorization: Bearer {admin_access_token}

Response 200:
{
  "success": true,
  "data": {
    "users": [
      {
        "id": "uuid",
        "name": "John Doe",
        "email": "john@example.com",
        "phone": "+23054321234",
        "role": "user",
        "created_at": "2026-01-20T...",
        "orders_count": 5,
        "interests_count": 2
      }
    ],
    "pagination": {...}
  }
}
```

Get User Details

```
GET /api/admin/users/{id}
Authorization: Bearer {admin_access_token}

Response 200:
{
  "success": true,
  "data": {
    "user": {...},
    "statistics": {
      "total_orders": 5,
      "total_spent": 1200,
      "interest_expressions": 2
    },
    "recentOrders": [...],
    "recentInterests": [...]
  }
}
```

Update User Role

```
PATCH /api/admin/users/{id}/role
Authorization: Bearer {admin_access_token}
Content-Type: application/json

{
  "role": "admin"
}

Response 200: User role updated
```

Delete User

```
DELETE /api/admin/users/{id}
Authorization: Bearer {admin_access_token}

Response 200: User deleted
```

📋 Admin Reports

Get Revenue Report

```
GET /api/admin/reports/revenue?startDate=2026-01-01&endDate=2026-01-31
Authorization: Bearer {admin_access_token}

Response 200:
{
  "success": true,
  "data": {
    "summary": {
      "totalRevenue": 12450,
      "totalOrders": 89,
      "averageOrderValue": 139.89
    },
    "revenueByPaymentMethod": [
      { "method": "cash_on_delivery", "revenue": 12450, "orders": 89 }
    ],
    "orders": [...]  // Last 50 orders
  }
}
```

Get Inventory Report

```
GET /api/admin/reports/inventory
Authorization: Bearer {admin_access_token}

Response 200:
{
  "success": true,
  "data": {
    "summary": {
      "activeListings": 15,
      "inStock": 12,
      "lowStock": 3,
      "outOfStock": 0,
      "totalStockUnits": 850
    },
    "listings": [
      {
        "id": "uuid",
        "title": "Brèdes Cresson",
        "stock": 50,
        "price": 45,
        "status": "in_stock",
        "category": "Légumes Frais"
      }
    ]
  }
}
```

Get Activity Log

```
GET /api/admin/activity-log?page=1&limit=50
Authorization: Bearer {admin_access_token}

Response 200:
{
  "success": true,
  "data": {
    "activities": [
      {
        "id": "uuid",
        "action": "order_approved",
        "admin": "Yash Beeharry",
        "admin_email": "yashbeeharry363@gmail.com",
        "order_number": "ORD-20260123-001",
        "notes": "Order approved. Will deliver tomorrow.",
        "timestamp": "2026-01-23T..."
      }
    ],
    "pagination": {...}
  }
}
```

🛠️ Utilities

Health Check

```
GET /api/health

Response 200:
{
  "success": true,
  "data": {
    "status": "healthy",
    "timestamp": "2026-01-23T...",
    "service": "Fresh Roots API",
    "version": "1.0.0"
  }
}
```

Test Email (Development Only)

```
GET /api/test-email?to=test@example.com

Response 200:
{
  "success": true,
  "message": "Test email sent successfully to test@example.com. Check your inbox!"
}
```

🔒 Authentication Flow

For Mobile/Web App:

1. **Register/Login** → Receive `access_token` and `refresh_token`
2. **Store tokens securely** (Keychain/Secure Storage)
3. **Include in all requests**: `Authorization: Bearer {access_token}`
4. **Token expires in 15 minutes** → Use refresh token to get new access token
5. **Refresh token expires in 30 days** → User must login again

Token Refresh Flow:

```
// When access token expires (401 error)
if (response.status === 401) {
  // Call refresh endpoint
  const newTokens = await fetch('/api/auth/refresh', {
    method: 'POST',
    headers: { 'Content-Type': 'application/json' },
    body: JSON.stringify({ refresh_token: stored_refresh_token })
  });

  // Update stored tokens
  updateTokens(newTokens);

  // Retry original request with new access token
  retryOriginalRequest();
}
```

🌍 Swagger Documentation

Interactive API Documentation: `http://localhost:3000/api-docs`

- Test all endpoints
- See request/response schemas
- Try with authentication
- Download OpenAPI specification

⚠️ Error Handling

All API errors follow this format:

```
{
  "success": false,
  "error": {
    "message": "Human-readable error message",
    "statusCode": 400,
    "details": {}  // Optional additional details
  }
}
```

Common Status Codes:
- `200` OK - Success
- `201` Created - Resource created
- `400` Bad Request - Invalid input
- `401` Unauthorized - Invalid/expired token
- `403` Forbidden - Insufficient permissions
- `404` Not Found - Resource not found
- `500` Internal Server Error - Server error

📱 Mobile App Integration Notes

React Native / Expo:

1. **API Client Setup:**

```
import axios from 'axios';
import AsyncStorage from '@react-native-async-storage/async-storage';

const API_URL = 'http://localhost:3000/api';  // Change for production

const api = axios.create({
  baseURL: API_URL,
  timeout: 10000,
});

// Add auth token to every request
api.interceptors.request.use(async (config) => {
  const token = await AsyncStorage.getItem('access_token');
  if (token) {
    config.headers.Authorization = `Bearer ${token}`;
  }
  return config;
});

// Handle token refresh on 401
api.interceptors.response.use(
  (response) => response,
  async (error) => {
    if (error.response?.status === 401) {
      const refreshToken = await AsyncStorage.getItem('refresh_token');
      if (refreshToken) {
        try {
          const { data } = await axios.post(`${API_URL}/auth/refresh`, {
            refresh_token: refreshToken,
          });
          await AsyncStorage.setItem('access_token', data.data.access_token);
          await AsyncStorage.setItem('refresh_token', data.data.refresh_token);
          // Retry original request
          error.config.headers.Authorization = `Bearer ${data.data.access_token}`;
          return axios.request(error.config);
        } catch (refreshError) {
          // Redirect to login
          await AsyncStorage.clear();
          // Navigate to login screen
        }
      }
    }
    return Promise.reject(error);
  }
);

export default api;
```

1. **Usage Examples:**

```
// Register
const register = async (name, email, password, phone) => {
  const { data } = await api.post('/auth/register', {
    name, email, password, phone
  });
  await AsyncStorage.setItem('access_token', data.data.tokens.access_token);
  await AsyncStorage.setItem('refresh_token', data.data.tokens.refresh_token);
  return data.data.user;
};

// Get listings
const getListings = async (page = 1, search = '') => {
  const { data } = await api.get('/listings', {
    params: { page, limit: 20, search }
  });
  return data.data;
};

// Create order
const createOrder = async (items, paymentMethod) => {
  const { data } = await api.post('/orders', {
    items,
    payment_method: paymentMethod
  });
  return data.data.order;
};
```

1. **PostHog Analytics Integration:**

```
import PostHog from 'posthog-react-native';

const posthog = new PostHog(
  'phc_bXtr0QqRHJPBwM8ImrULI7CYO678sq5DZf2Eqma5vnB',
  { host: 'https://us.i.posthog.com' }
);

// Track events
posthog.capture('product_viewed', {
  product_id: listing.id,
  product_name: listing.title,
  price: listing.price
});

posthog.capture('order_placed', {
  order_id: order.id,
  total_amount: order.total_amount
});
```

🎨 UI/UX Recommendations

Key Screens Needed:

1. **Splash Screen** - “Frais ek Kalite” 🌱🇲🇺
2. **Auth Screens** - Login, Register
3. **Home/Browse** - Product listings with categories

4. **Product Details** - Images, description, stock, price
5. **Cart** - Review items before order
6. **Checkout** - Payment method selection
7. **My Orders** - Order history and status
8. **Interest** - Express interest in products
9. **Profile** - User settings
10. **Admin Dashboard** (if admin) - Statistics, orders, users

Mauritian Touch:

- Use Creole greetings: “Bonzour”, “Mersi”, “Ki fer?”
- Green color theme (#2d8659)
- Local product images
- Mauritius flag emoji 🇲🇺

📞 Support & Questions

Admin Email: yashbeeharry363@gmail.com
PostHog Dashboard: https://app.posthog.com/
GitHub: (Add repository link)

Last Updated: January 23, 2026
API Version: 1.0.0
Backend Framework: NestJS + PostgreSQL + Prisma