FarmPower API Documentation

Generated on: 2025-06-17 00:07:09

FarmPower API Documentation

Table of Contents

Authentication

Base URL

```
http://localhost:8000/api
```

Authentication Headers

```
Authorization: Bearer <jwt_token>
```

User Management

Endpoints

Register User

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

{
    "email": "string",
    "password": "string",
    "full name": "string",
    "role": "farmer" | "dealer" | "service_provider" | "admin"
}
```

Login

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

{
    "email": "string",
    "password": "string"
}
```

Get User Profile

```
GET /api/users/me
```

Update User Profile

```
PUT /api/users/me
Content-Type: application/json

{
    "full name": "string",
    "email": "string"
}
```

Verify Email

```
POST /api/users/verify-email
Content-Type: application/json

{
    "email": "string",
    "otp": "string"
}
```

Equipment Marketplace

Endpoints

List Tractors

```
 GET /api/tractors
Query Parameters:
- page: int
- limit: int
- brand: string
- model: string
- price min: float
- price max: float
- location: string
```

Create Tractor Listing

```
 POST /api/tractors
Content-Type: application/json

{
    "name": "string",
    "brand": "string",
    "model": "string",
    "year": int,
    "price": float,
    "description": "string",
    "location": "string",
    "images": ["string"]  // Base64 encoded images
}
```

Get Tractor Details

```
 GET /api/tractors/{tractor_id}
```

Update Tractor Listing

```
 PUT /api/tractors/{tractor id}
Content-Type: application/json

{
    "name": "string",
    "price": float,
    "description": "string"
}
```

Delete Tractor Listing

```
DELETE /api/tractors/{tractor_id}
```

Field Management

Endpoints

List Fields

```
GET /api/fields
Query Parameters:
- page: int
- limit: int
```

Create Field

```
POST /api/fields
Content-Type: application/json

{
    "name": "string",
    "area": float,
    "crop_type": "string",
    "soil type": "string",
    "coordinates": {
        "type": "Feature",
        "geometry": {
            "type": "Polygon",
            "coordinates": [[[float, float], ...]]
        }
    }
}
```

Get Field Details

```
GET /api/fields/{field_id}
```

Update Field

```
 PUT /api/fields/{field id}
Content-Type: application/json

{
    "name": "string",
    "crop type": "string",
    "soil_type": "string"
}
```

Delete Field

```
 DELETE /api/fields/{field_id}
```

Crop Management

Endpoints

List Crops

```
 GET /api/crops
Query Parameters:
- page: int
- limit: int
- field_id: int
```

Create Crop Entry

```
 POST /api/crops
Content-Type: application/json

{
    "name": "string",
    "field id": int,
    "planting date": "date",
    "expected harvest date": "date",
    "expected yield": float,
    "costs": {
        "seeds": float,
        "fertilizer": float,
        "pesticides": float,
        "labor": float,
        "equipment": float
    }
}
```

Get Crop Details

```
 GET /api/crops/{crop_id}
```

Update Crop Entry

```
 PUT /api/crops/{crop id}
Content-Type: application/json

{
    "name": "string",
    "expected harvest date": "date",
    "expected_yield": float
}
```

Delete Crop Entry

```
 DELETE /api/crops/{crop_id}
```

Service Booking

Endpoints

List Service Bookings

```
 GET /api/services
Query Parameters:
- page: int
- limit: int
- status: string
```

Create Service Booking

```
 POST /api/services
Content-Type: application/json

{
    "tractor id": int,
    "service type": "string",
    "scheduled_date": "datetime",
    "description": "string"
}
```

Get Service Booking Details

```
GET /api/services/{booking_id}
```

Update Service Booking

```
PUT /api/services/{booking id}
Content-Type: application/json

{
    "scheduled date": "datetime",
    "status": "string"
}
```

Cancel Service Booking

```
DELETE /api/services/{booking_id}
```

Parts Marketplace

Endpoints

List Parts

```
GET /api/parts
Query Parameters:
- page: int
- limit: int
- category: string
- tractor brand: string
- condition: string
- price min: float
- price_max: float
```

Create Part Listing

```
POST /api/parts
Content-Type: application/json

{
    "name": "string",
    "category": "string",
    "tractor brand": "string",
    "condition": "string",
    "price": float,
    "quantity": int,
    "description": "string",
    "images": ["string"]  // Base64 encoded images
}
```

Get Part Details

```
GET /api/parts/{part_id}
```

Update Part Listing

```
PUT /api/parts/{part id}
Content-Type: application/json

{
    "price": float,
    "quantity": int,
    "description": "string"
}
```

Delete Part Listing

```
DELETE /api/parts/{part_id}
```

Messaging System

Endpoints

List Conversations

```
GET /api/messages/conversations
```

Get Conversation Messages

```
GET /api/messages/conversations/{conversation_id}
Query Parameters:
```

```
- page: int
- limit: int
```

Send Message

```
POST /api/messages
Content-Type: application/json

{
    "recipient id": int,
    "content": "string"
}
```

Mark Messages as Read

```
PUT /api/messages/conversations/{conversation_id}/read
```

Notifications

Endpoints

List Notifications

```
GET /api/notifications
Query Parameters:
- page: int
- limit: int
- unread_only: boolean
```

Mark Notification as Read

```
PUT /api/notifications/{notification_id}/read
```

Mark All Notifications as Read

```
PUT /api/notifications/read-all
```

Admin Operations

Endpoints

List Users

```
 GET /api/admin/users
Query Parameters:
- page: int
- limit: int
- role: string
```

Ban User

```
 PUT /api/admin/users/{user_id}/ban
```

Unban User

```
 PUT /api/admin/users/{user_id}/unban
```

Delete User

```
 DELETE /api/admin/users/{user_id}
```

Crop Calculator

Endpoints

Calculate Profitability

```
 POST /api/crop-profit/calculate
Content-Type: application/json

{
    "crop type": "string",
    "area": float,
    "expected yield": float,
    "market price": float,
    "costs": {
        "seeds": float,
        "fertilizer": float,
        "pesticides": float,
        "labor": float,
        "equipment": float
    }
}
```

Get Calculation History

```
GET /api/crop-profit/history
Query Parameters:
- page: int
- limit: int
```

Data Models

User

```
class User:
    id: int
    email: str
    hashed password: str
    full name: str
    role: UserRole
    is active: bool
    is banned: bool
    is verified: bool
    created at: datetime
    updated_at: datetime
```

Tractor

```
class Tractor:
    id: int
    name: str
    brand: str
    model: str
    year: int
    price: float
    description: str
    location: str
    owner_id: int
    created_at: datetime
    updated_at: datetime
```

Field

```
class Field:
    id: int
    name: str
    area: float
    crop type: str
    soil type: str
    coordinates: GeoJSON
    owner id: int
    created at: datetime
    updated_at: datetime
```

Crop

```
class Crop:
    id: int
    name: str
    field id: int
    planting date: date
    expected harvest date: date
    expected yield: float
    costs: JSON
    owner id: int
    created at: datetime
    updated_at: datetime
```

ServiceBooking

```
class ServiceBooking:
    id: int
    tractor id: int
    service type: str
    scheduled date: datetime
    status: str
    description: str
    user_id: int
    service provider id: int
    created at: datetime
    updated_at: datetime
```

Part

```
class Part:
    id: int
    name: str
    category: str
    tractor brand: str
    condition: str
    price: float
    quantity: int
    description: str
    seller id: int
    created at: datetime
    updated_at: datetime
```

Message

```
class Message:
    id: int
    sender id: int
    recipient id: int
    content: str
    is read: bool
    created_at: datetime
```

Notification

```
class Notification:
    id: int
    recipient id: int
    title: str
    content: str
    type: str
    is read: bool
    created_at: datetime
```

Error Responses

All endpoints may return the following error responses:

400 Bad Request

```
{
    "detail": "Invalid input data"
}
```

401 Unauthorized

```
{
    "detail": "Not authenticated"
}
```

403 Forbidden

```
{
    "detail": "Not authorized"
}
```

404 Not Found

```
{
    "detail": "Resource not found"
}
```

500 Internal Server Error

```
{
    "detail": "Internal server error"
}
```

Rate Limiting

API endpoints are rate-limited to prevent abuse: - 100 requests per minute for authenticated users - 20 requests per minute for unauthenticated users

Pagination

List endpoints support pagination with the following query parameters: - `page`: Page number (default: 1) - `limit`: Items per page (default: 10, max: 100)

Response format:

```
{
    "items": [],
    "total": 0,
    "page": 1,
    "limit": 10,
    "pages": 1
}
```

File Upload

For image uploads, use multipart/form-data with the following constraints: - Maximum file size: 5MB - Allowed formats: JPEG, PNG - Maximum dimensions: 1920x1080

WebSocket Events

The API supports real-time updates via WebSocket connections:

Connection

```
ws://localhost:8000/ws
```

Events

- `message received` : New message notification
- `notification` : New notification
- `service status update` : Service booking status change
- `tractor_status_update` : Tractor status change

Security

- All endpoints require HTTPS
- JWT tokens expire after 24 hours
- Passwords are hashed using bcrypt
- API keys are required for external services
- CORS is enabled for specified origins
- Rate limiting is implemented
- Input validation is enforced
- SQL injection protection is in place
- XSS protection is implemented