

# Assets Configuration Guide

Complete guide for configuring all assets, images, database entries, and AI prompts in Multilingual Mandi.

## Table of Contents

- [1. Image Assets](#)
- [2. Database Configuration](#)
- [3. AI Prompts](#)
- [4. Environment Variables](#)
- [5. Constants and Configuration](#)

## Image Assets

### Crop Images

**Location:** `frontend/public/images/crops/`

#### Current Images:

- `wheat.jpg`
- `rice.jpg`
- `tomato.jpg`
- `onion.jpg`
- `potato.jpg`
- `cotton.png`
- `maize.jpeg`
- `groundnut.jpg`
- `soybean.jpg`
- `sugarcane.webp`

### How to Add/Change Crop Images

#### Method 1: Manual Addition

- Place image file in `frontend/public/images/crops/`
- Use lowercase crop name as filename (e.g., `wheat.jpg`)
- Supported formats: JPG, PNG, WEBP

#### Method 2: Using Image Configuration File

**File:** `image_link.json`

```
{  
  "wheat": "https://example.com/wheat.jpg",  
}
```

```
"rice": "https://example.com/rice.jpg",
"tomato": "https://example.com/tomato.jpg"
}
```

### Download Images:

```
node download-crop-images.js
```

This script:

- Reads URLs from `image_link.json`
- Downloads images to `frontend/public/images/crops/`
- Names files based on crop name

### Image Path Format

In code, reference images as:

```
const imagePath = `/images/crops/${cropType.toLowerCase()}.jpg`;
```

With fallback:

```
<img
  src={`/${images/crops/${cropType.toLowerCase()}.jpg`}
  onError={(e) => e.target.src = '/images/crops/default.jpg'}
  alt={cropType}
/>
```

## Database Configuration

### Seed Data

**File:** `backend/src/utils/seed.js`

This file contains all initial database entries.

### Configurable Data

#### 1. Users

```
const users = [
  {
```

```
    phoneNumber: '9876543210',
    name: 'Ramesh Kumar',
    role: 'farmer',
    language: 'hi',
    location: 'Punjab'
  },
  // Add more users...
];
```

## 2. Listings

```
const listings = [
  {
    cropType: 'wheat',
    quantity: 1000,
    unit: 'kg',
    pricePerUnit: 25,
    qualityTier: 'premium',
    location: 'Punjab',
    description: 'Fresh wheat from Punjab farms',
    imageUrl: '/images/crops/wheat.jpg'
  },
  // Add more listings...
];
```

## 3. eNAM Prices (Mock Market Data)

```
const enamPrices = [
  {
    cropType: 'wheat',
    marketName: 'Delhi',
    minPrice: 20,
    maxPrice: 30,
    modalPrice: 25,
    date: new Date()
  },
  // Add more prices...
];
```

## How to Modify Seed Data

1. Edit `backend/src/utils/seed.js`
2. Run seed script:

```
cd backend
node src/utils/seed.js
```

Or reset database:

```
rm backend/mandi.db
npm start # Will auto-seed on first run
```

## External Seed Data File (Recommended)

Create `backend/data/seed-data.json`:

```
{
  "users": [
    {
      "phoneNumber": "9876543210",
      "name": "Ramesh Kumar",
      "role": "farmer",
      "language": "hi",
      "location": "Punjab"
    }
  ],
  "listings": [
    {
      "cropType": "wheat",
      "quantity": 1000,
      "unit": "kg",
      "pricePerUnit": 25,
      "qualityTier": "premium",
      "location": "Punjab",
      "description": "Fresh wheat from Punjab farms",
      "imageUrl": "/images/crops/wheat.jpg"
    }
  ],
  "enamPrices": [
    {
      "cropType": "wheat",
      "marketName": "Delhi",
      "minPrice": 20,
      "maxPrice": 30,
      "modalPrice": 25
    }
  ]
}
```

Paras

Then modify `seed.js` to read from this file.

---

# AI Prompts

## OpenRouter Prompts

**File:** `backend/src/services/AIService.js`

### 1. Intent Extraction Prompt

**Location:** `AIService.processVoiceQuery()`

```
const systemPrompt = `You are an AI assistant for an agricultural marketplace. Extract intent and parameters from user queries in any Indian language.
```

```
Possible intents:
```

- price\_query: User wants to know crop prices
- create\_listing: User wants to sell/list a product
- make\_offer: User wants to make an offer on a listing
- search\_listings: User wants to search/buy products
- general\_help: General questions

```
Extract these parameters if mentioned:
```

- cropType: Name of the crop (wheat, rice, tomato, onion, potato, cotton, etc.)
- quantity: Amount with unit (e.g., "100 kg")
- price: Price mentioned (number only)
- location: Place name
- qualityTier: premium, standard, or basic

```
Respond ONLY with valid JSON in this exact format:
```

```
{
  "intent": "<intent_type>",
  "cropType": "<crop_name or null>",
  "quantity": "<quantity or null>",
  "price": "<price or null>",
  "location": "<location or null>",
  "qualityTier": "<tier or null>",
  "confidence": "<high/medium/low>"
};
```

### How to Modify:

1. Edit the system prompt in `AIService.processVoiceQuery()`
2. Add new intents or parameters
3. Update response format if needed
4. Restart backend server

### 2. Negotiation Analysis Prompt

**Location:** `AIService.analyzeNegotiation()`

```
const prompt = `Analyze this agricultural product negotiation:
- Listing price: ₹${listingPrice}
- Buyer offer: ₹${offerPrice}
- Market average: ₹${marketAvg}

Provide:
1. A fair counter-offer price (just the number)
2. Brief reasoning (1 sentence)

Format: {"counterOffer": <number>, "reasoning": "<text>"}`;
```

### 3. Listing Description Generation Prompt

**Location:** `AIService.generateListingDescription()`

```
const prompt = `Generate a compelling 2-sentence product description for
${quantity} units of ${cropType} (${qualityTier} quality) from
${location} for an agricultural marketplace. Focus on freshness and
quality.`;
```

### Prompt Configuration File (Recommended)

Create `backend/config/prompts.json`:

```
{
  "intentExtraction": {
    "systemPrompt": "You are an AI assistant...",
    "intents": [
      "price_query",
      "create_listing",
      "make_offer",
      "search_listings",
      "general_help"
    ],
    "parameters": [
      "cropType",
      "quantity",
      "price",
      "location",
      "qualityTier"
    ]
  },
  "negotiation": {
    "template": "Analyze this agricultural product negotiation:\n-
Listing price: ₹{listingPrice}\n- Buyer offer: ₹{offerPrice}\n- Market
```

```

average: ₹{marketAvg}\n\nProvide:\n1. A fair counter-offer price (just
the number)\n2. Brief reasoning (1 sentence)\n\nFormat:
{"counterOffer\: <number>, \"reasoning\: \"<text>\"}
  },
  \"listingDescription\": {
    \"template\": \"Generate a compelling 2-sentence product description
for {quantity} units of {cropType} ({qualityTier} quality) from
{location} for an agricultural marketplace. Focus on freshness and
quality.\"
  }
}

```

Then load in `AIService.js`:

```

const prompts = require('../config/prompts.json');

```

## Environment Variables

### Backend Environment Variables

**File:** `backend/.env`

```

# Server Configuration
NODE_ENV=development
PORT=5000
DATABASE_URL=sqlite:./mandi.db

# JWT Configuration
JWT_SECRET=your-super-secret-jwt-key-change-in-production
JWT_EXPIRES_IN=7d

# SARVAM AI Configuration
SARVAM_API_KEY=sarvam-key
SARVAM_API_URL=https://api.sarvam.ai

# OpenRouter AI Configuration
OPENROUTER_API_KEY=openrouter-key
OPENROUTER_MODEL=qwen/qwen3-vl-32b-instruct
OPENROUTER_API_URL=https://openrouter.ai/api/v1

# Frontend URL
FRONTEND_URL=http://localhost:3000

# CORS Configuration
CORS_ORIGIN=http://localhost:3000

```

## Frontend Environment Variables

**File:** `.env` (root directory)

```
VITE_API_URL=http://localhost:5000
```

## How to Change API Keys

1. Get new API keys from providers:
  - SARVAM: <https://www.sarvam.ai/>
  - OpenRouter: <https://openrouter.ai/>
2. Update `backend/.env`:

```
SARVAM_API_KEY=your-new-key  
OPENROUTER_API_KEY=your-new-key
```

3. Restart backend server

## How to Change AI Model

Edit `backend/.env`:

```
# Available models on OpenRouter:  
# - qwen/qwen3-vl-32b-instruct (current)  
# - qwen/qwen-2.5-72b-instruct  
# - meta-llama/llama-3.1-70b-instruct  
# - anthropic/claude-3-sonnet  
  
OPENROUTER_MODEL=qwen/qwen3-vl-32b-instruct
```

---

Paras

---

## Constants and Configuration

### Crop Types

**File:** `backend/src/utils/validators.js` or create `backend/config/crops.json`

```
{  
  "crops": [  
    {  
      "name": "wheat",  
      "displayName": "Wheat",  
      "hindiName": "गेहूँ",  
    },  
  ],  
}
```



```

        "category": "grain",
        "defaultUnit": "kg",
        "imageUrl": "/images/crops/wheat.jpg"
    },
    {
        "name": "rice",
        "displayName": "Rice",
        "hindiName": "चावल",
        "category": "grain",
        "defaultUnit": "kg",
        "imageUrl": "/images/crops/rice.jpg"
    },
    {
        "name": "tomato",
        "displayName": "Tomato",
        "hindiName": "टमाटर",
        "category": "vegetable",
        "defaultUnit": "kg",
        "imageUrl": "/images/crops/tomato.jpg"
    }
]
}

```

## Quality Tiers

**File:** backend/src/services/PricingService.js

```
const QUALITY_MULTIPLIERS = {
  premium: 1.2, // 20% above base price
  standard: 1.0, // Base price
  basic: 0.8 // 20% below base price
};
```

Paras

### How to Modify:

```
const QUALITY_MULTIPLIERS = {
  premium: 1.3,    // 30% above base price
  standard: 1.0,   // Base price
  basic: 0.7       // 30% below base price
};
```

## Supported Languages

**File:** backend/src/services/TranslationService.js

```
static SUPPORTED_LANGUAGES = {  
  hi: 'Hindi',  
  mr: 'Marathi',  
  ta: 'Tamil',  
  te: 'Telugu',  
  kn: 'Kannada',  
  pa: 'Punjabi',  
  gu: 'Gujarati',  
  ml: 'Malayalam',  
  or: 'Odia',  
  bn: 'Bengali',  
  en: 'English'  
  // Add more languages...  
};
```

## Trust Score Thresholds

**File:** `backend/src/services/TrustService.js`

```
const TRUST_THRESHOLDS = {  
  excellent: 80,  
  good: 60,  
  fair: 40,  
  poor: 20  
};
```

---

## Configuration Best Practices

### 1. Use Environment Variables for Secrets

- Never commit API keys to git
- Use `.env` files (already in `.gitignore`)
- Use `.env.example` as template

### 2. Use JSON Files for Data

- Easier to edit without code knowledge
- Can be updated without restarting server
- Can be version controlled separately

### 3. Centralize Configuration

Create `backend/config/` directory:

```
backend/config/
├─ crops.json           # Crop definitions
├─ languages.json       # Language configurations
├─ prompts.json         # AI prompts
├─ constants.json       # App constants
└─ features.json        # Feature flags
```

## 4. Document Changes

- Update this file when adding new configurations
- Add comments in configuration files
- Keep a changelog

---

## Quick Reference

### Change Crop Images

1. Add image to `frontend/public/images/crops/`
2. Name it `{cropname}.jpg`
3. Or update `image_link.json` and run `node download-crop-images.js`

### Change Database Data

1. Edit `backend/src/utils/seed.js`
2. Delete `backend/mandi.db`
3. Restart server

### Change AI Prompts

1. Edit `backend/src/services/AIService.js`
2. Find the relevant method
3. Update the prompt string
4. Restart server

### Change API Keys

1. Edit `backend/.env`
2. Update `SARVAM_API_KEY` or `OPENROUTER_API_KEY`
3. Restart server

### Change AI Model

1. Edit `backend/.env`
  2. Update `OPENROUTER_MODEL`
  3. Restart server
-

# Support

For questions or issues with configuration:

1. Check this documentation
2. Check `.env.example` for required variables
3. Check service files for available options
4. Refer to API provider documentation