

Version 0.5.1 — February 2026

Niv AI

The Complete AI Assistant for ERPNext

Turn your ERPNext into an intelligent system. Chat in natural language, automate workflows, build custom DocTypes, and manage your entire business — all through conversation.

By Ravindra Kulhari
github.com/kulharir7/niv_ai

Table of Contents

Chapter 1 — Introduction

What is Niv AI?

Key Features

Architecture Overview

System Requirements

Chapter 2 — Installation & Setup

Docker Installation

Manual Installation (bench)

Initial Configuration

AI Provider Setup (Mistral / OpenAI)

Chapter 3 — Chat Interface

Web Chat (Desk Page)

Floating Widget

Conversations & History

Reactions & Search

Chapter 4 — MCP Tools (29 Tools)

Document Operations

Search & Discovery

Reports & Analytics

Developer Tools

Dashboard & Visualization

Chapter 5 — Developer Mode

Enabling Developer Mode

Creating DocTypes via Chat

Custom Fields & Scripts

Workflows & Notifications

Confirmation & Undo System

94 Developer Features (RAG)

Chapter 6 — Auto-Pilot Triggers

What are Niv Triggers?

Creating Triggers

System Prompts

Use Cases & Examples

Chapter 7 — Voice Mode

Voice Architecture

Speech-to-Text (STT)

Text-to-Speech (TTS)

Continuous Conversation

Chapter 8 — Telegram Bot

Setup & Configuration

User Mapping

Progressive Updates

Commands & Features

Chapter 9 — WhatsApp Bot

Meta Cloud API Setup

QR Code Onboarding

Message Formatting

Chapter 10 — NBFC / Lending Use Case

Growth System Overview

NBFC-Specific Prompts

Loan Application Management

Co-Lending Operations

Compliance & Reporting

Chapter 11 — Billing & Token Management

Shared Pool vs Per User

Razorpay Integration

Usage Tracking

Chapter 12 — Administration

Niv Settings Reference

Rate Limiting

User Permissions

Error Monitoring

Chapter 13 — Deployment Guide

Production Deployment (AWS/VPS)

Docker Configuration

Nginx & SSL

Backup & Recovery

Chapter 14 — API Reference

Chat API

Conversation API

Voice API

Telegram & WhatsApp Webhooks

Billing API

Appendix A — Prompt Library

Appendix B — Troubleshooting

Appendix C — Roadmap

Introduction

Understanding Niv AI and what it can do for your business

What is Niv AI?

Niv AI is a **complete AI assistant** built as a native Frappe/ERPNext application. It transforms your ERPNext instance into an intelligent system where users can interact with their business data through natural language conversation — in English, Hindi, or Hinglish.

Unlike external AI chatbots that have no knowledge of your data, Niv AI is deeply integrated with ERPNext. It can read your Sales Orders, create Invoices, generate reports, build custom DocTypes, and even automate business workflows — all through simple conversation.



Natural Language Chat

Ask questions in plain language. "Aaj ki sales kitni?" or "Show pending invoices" — Niv understands both.



29 MCP Tools

Create, read, update, delete documents. Run queries. Generate reports. All through conversation.



Developer Mode

Build DocTypes, Custom Fields, Server Scripts, Workflows — all by telling AI what you need.



Auto-Pilot Triggers

AI automatically validates documents on submit. Sales Order → AI checks → warns about issues.



Voice Mode

Speak to your ERPNext. Voice input with speech recognition, AI responds with text-to-speech.



Telegram & WhatsApp

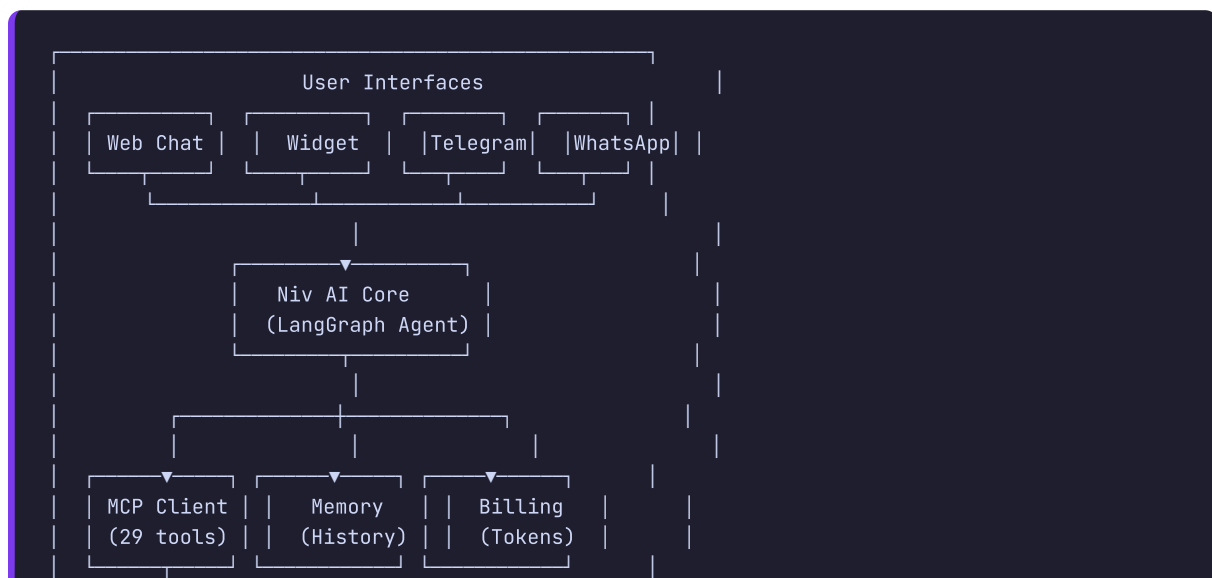
Access your ERPNext from Telegram or WhatsApp. Same AI power, on your phone.

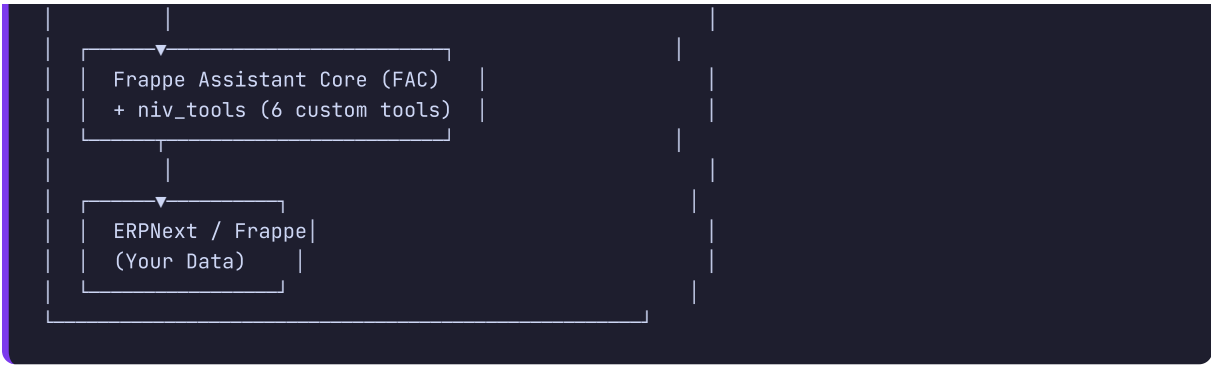
Key Features

Feature	Description	Status
Web Chat UI	Dark theme, Claude-inspired, glassmorphism design	✓ Live
Floating Widget	Chat from any ERPNext page via floating button	✓ Live
29 MCP Tools	Full CRUD + search + reports + analytics	✓ Live
Developer Mode	94 features via RAG — create DocTypes, scripts, workflows	✓ Live
Auto-Pilot Triggers	AI runs on document events (submit, save, etc.)	✓ Live
Voice Mode	Speech input + TTS output (Piper + Voxtral)	✓ Live
Telegram Bot	Full AI access from Telegram with progressive updates	✓ Live
WhatsApp Bot	Meta Cloud API integration with QR onboarding	✓ Ready
Token Billing	Shared Pool + Per User with Razorpay payments	✓ Live
Conversation History	Full chat history, search, reactions	✓ Live
v14 + v15 Compatible	Works on both Frappe v14 and v15	✓ Tested
Multi-language	English, Hindi, Hinglish responses	✓ Live

Architecture Overview

Niv AI follows a clean, modular architecture:





System Requirements

Component	Minimum	Recommended
Frappe	v14.0	v15.x
ERPNext	v14.0	v15.x
Python	3.10	3.11+
Node.js	16.x	18.x+
RAM	2 GB	4 GB+
AI Provider	Any OpenAI-compatible	Mistral AI

AI Provider

Niv AI works with any OpenAI-compatible API: Mistral AI, OpenAI, Anthropic (via proxy), Groq, Together AI, local models (Ollama), etc. Mistral AI is recommended for best Hindi/Hinglish support.

Installation & Setup

Get Niv AI running on your ERPNext instance

Docker Installation

If you're running ERPNext on Docker (recommended for development):

Step 1: Install the App

```
# Enter your backend container
docker exec -it frappe_docker-backend-1 bash

# Navigate to bench
cd /home/frappe/frappe-bench

# Get the apps
bench get-app https://github.com/kulharir7/niv_ai.git
bench get-app https://github.com/buildswithpaul/Frappe_Assistant_Core.git

# Install on your site
bench --site your-site install-app frappe_assistant_core
bench --site your-site install-app niv_ai

# Build frontend assets
bench build --app niv_ai

# Restart
bench restart
```

⚠ Docker Restart Note

After Docker container restart, you may need to reinstall pip packages and fix nginx SSE config. Use the provided `scripts/full_restore.ps1` script.

Step 2: Install niv_tools (Optional but Recommended)

```
# niv_tools adds 6 extra tools: universal_search, explore_fields,
# test_created_item, monitor_errors, rollback_changes, introspect_system
```

```
bench get-app https://github.com/kulharir7/niv_tools.git
bench --site your-site install-app niv_tools
bench restart
```

Manual Installation (bench)

For standard bench installations (production servers, VPS):

```
# From your bench directory
bench get-app https://github.com/kulharir7/niv_ai.git
bench get-app https://github.com/buildswithpaul/Frappe_Assistant_Core.git

bench --site your-site install-app frappe_assistant_core
bench --site your-site install-app niv_ai

bench build --app niv_ai
sudo supervisorctl restart all
```

Initial Configuration

After installation, navigate to **Niv Settings** in your ERPNext:

AI Configuration

1. Go to **Niv Settings** (search in awesome bar)
2. Create a **Niv AI Provider** first:
 - Name: **Mistral**
 - Base URL: **https://api.mistral.ai/v1**
 - API Key: Your Mistral API key
3. In Niv Settings, set:
 - Default Provider: **Mistral**
 - Default Model: **mistral-medium-2508**
 - Enable Tools: ☒
 - Enable Widget: ☒

AI Provider Options

Provider	Base URL	Recommended Model	Cost (approx)
Mistral AI	https://api.mistral.ai/v1	mistral-medium-2508	₹0.8/1K input

OpenAI	https://api.openai.com/v1	gpt-4o-mini	₹1.2/1K input
Groq	https://api.groq.com/openai/v1	llama-3.3-70b	Free tier available
Together AI	https://api.together.xyz/v1	meta-llama/Llama-3-70b	₹0.6/1K input
Ollama (local)	http://localhost:11434/v1	llama3.2	Free (your GPU)

Verify Installation

```
# Check all apps are installed
bench --site your-site list-apps
# Should show: frappe, erpnext, frappe_assistant_core, niv_ai, niv_tools

# Check tools are discovered
bench --site your-site console
>>> from niv_ai.niv_core.mcp_client import MCPClient
>>> client = MCPClient()
>>> tools = client.discover_tools()
>>> print(f"Found {len(tools)} tools")
# Should show: Found 29 tools
```

Chat Interface

Beautiful, powerful chat UI for every user

Web Chat (Desk Page)

Access the full chat interface at </app/niv-chat>. The UI features:

- **Dark Theme** — Professional dark UI (#1a1a2e) with purple accent (#7c3aed)
- **Conversation Sidebar** — List of all chats with search and archive
- **Tool Call Accordion** — See which tools AI used, expandable for details
- **Code Highlighting** — Syntax-highlighted code blocks
- **Markdown Rendering** — Tables, lists, bold, italic, links
- **Reactions** — Thumbs up/down + emoji reactions on messages
- **Conversation Search** — Instant local search + server-side full-text search

Chat Examples

Business Query

User: Aaj ki total sales kitni hai?

Niv AI: Uses `run_database_query` tool → "Today's total sales: ₹4,58,000 across 12 Sales Orders."

Document Creation

User: Create a new Customer named "ABC Technologies" with billing address in Mumbai

Niv AI: Uses `create_document` tool → Creates Customer with all fields filled.

Report Generation

User: Show me top 5 customers by revenue this month

Niv AI: Uses `run_database_query` → Returns sorted table with customer names and amounts.

Floating Widget

When enabled in Niv Settings, a floating purple button (FAB) appears on the bottom-right of every ERPNext page. Click to open a chat panel without leaving your current page.

The widget features:

- Glassmorphism design with gradient FAB button
- Full chat functionality (same as desk page)
- Auto-minimizes when clicking outside
- Persists conversation across page navigation

Conversations & History

Every chat creates a **Niv Conversation** document with:

- Full message history (user + AI messages)
- Token usage tracking per conversation
- Channel identification (web, telegram, whatsapp)
- Archive/unarchive capability
- Auto-title generation from first message

Reactions & Search

Reactions

React to AI messages with thumbs up/down or emoji reactions (Discord-style). Reactions are saved to the server for feedback tracking.

Search

Two search modes:

- **Local Search** — Instant, searches current conversation messages
- **Server Search** — Activated on 3+ characters, searches all conversations with debouncing



MCP Tools — 29 Operations

Put AI's hands and eyes into your ERPNext data

Niv AI uses the **Model Context Protocol (MCP)** to interact with ERPNext. Through Frappe Assistant Core (FAC) and niv_tools, it has access to 29 tools that can read, write, search, and analyze your data.

Document Operations (8 tools)

Tool	Description	Example Usage
<code>create_document</code>	Create any document	"Create a Sales Order for Customer X"
<code>get_document</code>	Read a specific document	"Show me Sales Order SO-001"
<code>update_document</code>	Modify document fields	"Change delivery date of SO-001 to March 15"
<code>delete_document</code>	Delete a document	"Delete the draft quotation Q-005"
<code>submit_document</code>	Submit a submittable doc	"Submit Sales Invoice INV-001"
<code>list_documents</code>	List documents with filters	"Show all draft Purchase Orders"
<code>search_documents</code>	Full-text search	"Search invoices containing 'ABC Corp'"
<code>run_workflow</code>	Execute workflow action	"Approve Leave Application LEA-001"

Search & Discovery (6 tools)

Tool	Description	Example Usage
<code>search</code>	Global search across all DocTypes	"Find anything related to 'laptop'"
<code>search_doctype</code>	Search for DocType definitions	"What DocTypes are related to HR?"
<code>search_link</code>	Search link field values	"Search customers starting with 'West'"

<code>fetch</code>	Fetch specific field values	"What is the grand total of SO-001?"
<code>get_doctype_info</code>	Get DocType metadata	"What fields does Sales Order have?"
<code>universal_search</code>	Search ALL fields across ALL DocTypes	"Find '9876543210' anywhere in the system"

Reports & Analytics (5 tools)

Tool	Description	Example Usage
<code>generate_report</code>	Run built-in reports	"Run Accounts Receivable report"
<code>report_list</code>	List available reports	"What reports are available for Sales?"
<code>report_requirements</code>	Get report parameters	"What filters does P&L report need?"
<code>run_database_query</code>	Execute SQL queries (read-only)	"SELECT sum(grand_total) FROM `tabSales Order`"
<code>analyze_business_data</code>	Complex business analysis	"Analyze monthly sales trend"

Developer Tools (8 tools)

Tool	Description	Example Usage
<code>run_python_code</code>	Execute Python in Frappe context	"Count all active users"
<code>explore_fields</code>	Explore field metadata & data lineage	"Show all currency fields in Sales Invoice"
<code>test_created_item</code>	Verify document creation works	"Test if I can create a ToDo"
<code>monitor_errors</code>	Check Error Log for issues	"Any errors in last 24 hours?"
<code>rollback_changes</code>	Undo custom fields/scripts	"Remove the Custom Field I just created"
<code>introspect_system</code>	System overview (apps, modules, etc.)	"What apps are installed?"
<code>extract_file_content</code>	Read uploaded file contents	"Read the uploaded PDF"
<code>create_dashboard</code>	Create dashboard with charts	"Create a sales dashboard"

Tool Reliability Features

- **Circuit Breaker** — After 3 failures, skips HTTP for 60 seconds, auto-heals
- **Retry with Backoff** — Retries on transient errors (0.5s, 1.5s delays)
- **Argument Validation** — Validates required fields and types before execution
- **Error Recovery** — Graceful error messages instead of crashes

Developer Mode

Build entire ERPNext customizations through conversation

Developer Mode transforms Niv AI into a **full-stack Frappe developer**. Toggle it on, and the AI gains deep knowledge of Frappe's architecture — DocTypes, Custom Fields, Server Scripts, Client Scripts, Workflows, Print Formats, Notifications, and more.

Enabling Developer Mode

1. Open Niv Chat (</app/niv-chat>)
2. Look for the **Developer Mode toggle** in the sidebar (only visible to System Managers)
3. Click to enable — a **DEV** badge appears in the header
4. AI now has access to 94 developer features via RAG knowledge base

⚠ System Manager Only

Developer Mode requires the System Manager role. Regular users cannot enable it. This protects your system from unauthorized customizations.

Creating DocTypes via Chat

💡 Example: NBFC Loan Application DocType

User: "NBFC ke liye Loan Application DocType banao with applicant name, loan amount, interest rate, tenure, approval workflow"

Niv AI will:

1. Create the Custom DocType "Loan Application"
2. Add fields: applicant_name (Data), loan_amount (Currency), interest_rate (Float), tenure_months (Int)
3. Add naming series: LOAN-APP-.YYYY.-

4. Ask for confirmation before creating
5. Show undo option after creation

Custom Fields & Scripts

Custom Fields

Adding a field

User: "Sales Order mein 'Expected Delivery Date' ke baad ek 'Priority Level' field add karo — options: Low, Medium, High, Critical"

AI creates: Custom Field with fieldtype=Select, options="Low\nMedium\nHigh\nCritical", insert_after="expected_delivery_date"

Server Scripts

Validation Script

User: "Sales Order submit hone se pehle check karo ki grand total 1 lakh se zyada hai toh manager approval chahiye"

AI creates: Server Script (Before Submit) on Sales Order:

```
if doc.grand_total > 100000:
    if not doc.custom_manager_approved:
        frappe.throw("Orders above ₹1,00,000 require Manager approval.
        Please get the 'Manager Approved' checkbox checked.")
```

Client Scripts

UI Enhancement

User: "Sales Invoice form mein jab customer select ho toh uska outstanding balance show karo as a message"

AI creates: Client Script that fetches outstanding on customer change and shows alert.

Workflows & Notifications

Approval Workflow

User: "Purchase Order ke liye approval workflow banao — Draft → Pending Approval → Approved → Ordered. Amount > 50000 ho toh Director approval chahiye."

AI creates:

- Workflow with 4 states
- Transition rules based on amount
- Role-based approvals (Purchase Manager → Director)
- Email notifications on state change

Confirmation & Undo System

Developer Mode has built-in safety:

Confirmation Flow

Before any write operation (create, modify, delete), AI asks for confirmation:

AI: I'll create a Custom Field "Priority Level" on Sales Order. Proceed?
Type "yes", "ha", or "haan" to confirm.

User: ha

AI: ☒ Created Custom Field "Priority Level" on Sales Order.
You can say "undo" within 30 minutes to reverse this.

Undo System

Every creation is stored in a Redis-backed undo stack (30-minute expiry). Say "undo" to reverse the last action.

94 Developer Features (RAG Knowledge Base)

The developer knowledge base covers 94 features across 12 phases:

Phase	Category	Features
A	Custom Fields	Add, modify, delete fields on any DocType

B	Server Scripts	Before/After Save, Submit, Cancel, Delete
C	Client Scripts	Form events, list events, validation
D	Property Setters	Change field properties without code
E	Custom DocTypes	Create entire new DocTypes
F	Workflows	Multi-state approval workflows
G	Notifications	Email/SMS alerts on events
H	Print Formats	Custom document printing layouts
I	Reports	Script Reports with filters and charts
J	Web Pages	Portal pages and web forms
K	Integrations	Webhooks, REST API connections
L	Debug	Error diagnosis, performance tuning

Auto-Pilot Triggers

that watches your documents and acts automatically

What are Niv Triggers?

Niv Triggers are **document event hooks** that automatically run the AI when something happens in ERPNext. When a Sales Order is submitted, a Purchase Invoice is created, or an Employee record is modified — the AI can analyze, validate, and comment on the document.

Creating Triggers

Go to **Niv Trigger** list and create a new trigger:

Field	Description	Example
Trigger Name	Descriptive name	"Validate Sales Order on Submit"
Reference DocType	Which document to watch	Sales Order
Doc Event	When to trigger	After Submit
Condition	Optional Python condition	<code>doc.grand_total > 100000</code>
Prompt Template	What to tell the AI	"Analyze this Sales Order..."
Include Document Data	Send document fields to AI	<input checked="" type="checkbox"/> Yes
System Prompt	Link to Niv System Prompt	"Sales Order Validator"

Available Doc Events

- **Before Save** — Before document is saved (can prevent save)
- **After Save** — After document is saved
- **Before Submit** — Before document is submitted
- **After Submit** — After document is submitted

- **Before Cancel** — Before document is cancelled
- **After Cancel** — After document is cancelled
- **Before Delete** — Before document is deleted

System Prompts




Create reusable AI instructions in **Niv System Prompt**:

Sales Order Validator

You are a Sales Order validation expert. Analyze the submitted Sales Order and check for:

1. Customer credit limit — is the order within their limit?
2. Item availability — are all items in stock?
3. Pricing — are there any unusual discounts (>20%)?
4. Payment terms — are they standard or need review?
5. Delivery date — is it realistic given current workload?

Respond with:

-  APPROVED if everything looks good
-  WARNING if there are concerns but not blocking
-  REJECTED if there are critical issues

Always explain your reasoning in 2-3 bullet points.

Use Cases & Examples


1. Sales Order Validation

Trigger: Validate on Submit

DocType: Sales Order

Event: After Submit

Condition: `doc.grand_total > 50000`

Result: AI analyzes the order, adds a comment like "  Conditionally Valid — discount is 25%, exceeds 20% threshold. Please verify with Sales Manager."

2. Purchase Invoice Audit

Trigger: Audit on Save

DocType: Purchase Invoice

Event: After Save

Prompt: "Check if this purchase invoice has matching Purchase Order and GRN. Flag any discrepancies in quantities or rates."

3. Employee Onboarding Check

Trigger: Completeness Check

DocType: Employee

Event: Before Submit

Prompt: "Verify all mandatory onboarding fields are filled: PAN, Aadhaar, bank details, emergency contact. List any missing fields."

4. NBFC: Loan Disbursement Check

Trigger: Pre-Disbursement Validation

DocType: Loan

Event: Before Submit

Prompt: "Validate loan disbursement: check KYC completion, CIBIL score threshold (>650), co-lender approval status, and sanction letter validity. Block if any compliance check fails."

Voice Mode

Speak to your ERPNext and hear it respond

Voice Architecture

```
User speaks → Browser SpeechRecognition (fast, free)
               ↓ (if fails)
             Mistral Voxtral STT (accurate, paid)
               ↓
            Niv AI Agent (processes query)
               ↓
        Piper TTS (local, free) → Audio plays
               ↓ (if Piper unavailable)
        Browser speechSynthesis (fallback)
```

Speech-to-Text (STT)

Niv AI uses a **dual STT approach**:

Primary: Browser SpeechRecognition

- Free, zero API cost
- Works in Chrome, Edge (WebKit-based browsers)
- Supports Hindi (hi-IN) and English (en-IN)
- Real-time — shows text as you speak

Fallback: Mistral Voxtral

- Activated when browser STT fails or returns empty
- Audio recorded as base64, uploaded to server
- Uses `voxtral-mini-latest` model
- Higher accuracy for noisy environments

Text-to-Speech (TTS)

Primary: Piper TTS (Local)

- Runs entirely on your server — no API costs
- Available voices: `en_US-lessac-medium` (English), `hi_IN-priyamvada-medium` (Hindi)
- Fast generation (~1 second for typical responses)
- Audio served as WAV file

Fallback: Browser `speechSynthesis`

- Uses system voices when Piper is unavailable
- Quality varies by OS and browser

Continuous Conversation

Voice mode supports continuous conversation:

1. Press microphone button → speak your question
2. AI processes and speaks the answer
3. After AI finishes speaking, microphone auto-activates
4. Speak next question — no button pressing needed
5. Press end call button to exit voice mode

Voice UI

Pipecat-inspired design with:

- Purple-teal gradient orb (160px) — morphs and pulses
- 9-bar equalizer with gradient colors
- Glassmorphism control bar (mute, end call, speaker)
- Animated states: listening (teal rings), processing (conic spin), speaking (blob morph)

Telegram Bot



II ERPNext AI access from your Telegram app

Setup & Configuration

Step 1: Create Bot

1. Open Telegram, search for **@BotFather**
2. Send `/newbot`
3. Choose a name: "Niv AI" (display name)
4. Choose a username: `your_niv_ai_bot` (must end with "bot")
5. Copy the **bot token** (format: `123456789:AABBccDDeeFF...`)

Step 2: Configure Niv Settings

1. Go to **Niv Settings** →  **Telegram Bot**
2. Paste **Telegram Bot Token**
3. Set **Webhook URL**: Your public server URL (e.g., `https://yourdomain.com`)
4. Enable **Enable Telegram Bot** 

Step 3: Register Webhook

```
# From browser or curl:
POST /api/method/niv_ai.niv_core.api.telegram.setup_webhook

# Or manually via Telegram API:
curl https://api.telegram.org/bot{TOKEN}/setWebhook \
  -d url=https://yourdomain.com/api/method/niv_ai.niv_core.api.telegram.webhook
```

Step 4: Link Users

Create **Niv Telegram User** records to map Telegram users to Frappe users:

Field	Description
-------	-------------

Telegram User ID	User's Telegram numeric ID (get from @userinfobot)
Frappe User	Email of the ERPNext user
Enabled	Toggle access on/off

Progressive Updates

Unlike basic bots that go silent while processing, Niv AI's Telegram bot shows real-time progress:

```

User: "Aaj ki sales kitni?"

Bot: ⏳ Processing...                               ← Instant

Bot: ⏳ Working...                                     ← Updates when tool starts
    📊 Database query chala raha hoon...

Bot: [status message deleted]                         ← Clean up

Bot: 🛠️ Tools used:                                   ← Final response
    • 📊 Database query

Aaj ki total sales: ₹4,58,000
Total orders: 12
Top customer: West View Software (₹1,29,000)

```

Commands & Features

Command	Description
<code>/start</code>	Welcome message with usage examples
<code>/help</code>	List of commands and example queries
Any text	Sends to AI agent, gets ERPNext response

Features

- Full 29-tool access — same as web chat
- Markdown formatting (bold, code blocks, lists)
- Tables rendered as monospace code blocks
- Automatic message splitting for long responses (>4096 chars)
- Typing indicator while processing
- Conversation persistence — maintains chat history per user


💡 Testing Locally

Use **ngrok** for local testing: `ngrok http 8081` gives you a public URL. Set that as your webhook URL in Niv Settings. Free plan works fine for development.

WhatsApp Bot

PNNext AI on the world's most popular messaging app

Meta Cloud API Setup

1. Go to **developers.facebook.com** → Create App → Business type
2. Add **WhatsApp** product to your app
3. In WhatsApp → API Setup:
 - Note your **Phone Number ID**
 - Generate a **Permanent Access Token** (System User → Generate Token)
4. In Niv Settings →  WhatsApp Bot:
 - Set Phone Number ID
 - Set Access Token
 - Set Verify Token (any secret string you choose)
5. In Meta Dashboard → WhatsApp → Configuration → Webhook:
 - Callback URL: `https://yourdomain.com/api/method/niv_ai.niv_core.api.whatsapp.webhook`
 - Verify Token: same string as Niv Settings
 - Subscribe to: **messages**

QR Code Onboarding

Generate a QR code that opens WhatsApp chat with your bot:

```
# API call to generate QR
POST /api/method/niv_ai.niv_core.api.whatsapp.generate_qr
{
  "phone_number": "918322566235",
  "prefill_text": "Hi"
}

# Returns:
{
```

```

"wa_link": "https://wa.me/918322566235?text=Hi",
"qr_url": "https://chart.googleapis.com/chart?cht=qr&chs=300x300&chl=...",
"instructions": "Print this QR or share the link..."
}

```

Print the QR code or share the link. Users scan → WhatsApp opens → they type → Niv AI replies!

WhatsApp Message Formatting

WhatsApp has limited formatting support. Niv AI automatically converts:

Markdown	WhatsApp Output
Tables (...)	Bullet lists with key: value format
### Headers	*Bold text*
[link](url)	text: url
bold	*bold*
```code```	```code```

### WhatsApp Costs

Meta provides **1,000 free conversations per month**. After that, costs are approximately ₹0.50 per conversation (24-hour window). Business-initiated messages cost more than user-initiated ones.

# NBFC / Lending Use Case

Complete guide for Non-Banking Financial Companies using Niv AI

## Growth System Overview

The Growth System ( [mdfc-test.growthsystem.in](https://mdfc-test.growthsystem.in) ) is an ERPNext v14 implementation for NBFC operations. It includes specialized modules for:

- **LOS (Loan Origination System)** — Application intake, KYC, credit scoring
- **LMS (Loan Management System)** — Disbursement, EMI, collections
- **Co-Lending** — Partner bank integrations
- **Accounting** — GL entries, P&L, balance sheet
- **Litigation** — NPA management, legal notices
- **HR** — Employee management, payroll

## NBFC-Specific Prompts

Configure these system prompts in **Niv System Prompt** for NBFC operations:

### Loan Application Processor

```
You are an NBFC loan processing assistant for Growth System.
You help loan officers manage applications efficiently.
```

```
Your capabilities:
```

- ```
- Search and retrieve loan applications  
- Check KYC document status  
- Verify CIBIL scores and eligibility  
- Calculate EMI and disbursement amounts  
- Track application status through workflow stages
```

```
Rules:
```

- ```
- Always verify KYC before approving any action
- Flag applications with CIBIL score below 650
- Ensure co-lender approval for co-lending loans
- Never modify loan terms without proper authorization
```

- Report all compliance issues immediately

Respond in Hindi/Hinglish as preferred by the user.

### Collection Manager

You are a collection management assistant for an NBFC.

Your responsibilities:

- Track overdue EMI payments
- Generate collection reports by bucket (1-30, 31-60, 61-90, 90+ DPD)
- Identify high-risk accounts (3+ missed payments)
- Suggest collection strategies based on account history
- Monitor NPA (Non-Performing Asset) classification
- Track legal notices and litigation status

Important:

- Follow RBI guidelines for collection practices
- Do not suggest harassment or illegal collection methods
- Prioritize resolution and restructuring over legal action
- Flag accounts approaching NPA classification (89 DPD)

### Co-Lending Operations

You are a co-lending operations assistant.

Key functions:

- Match loans to co-lending partners based on criteria
- Track co-lender disbursement status
- Monitor interest rate splits and fee structures
- Generate MIS reports for partner banks
- Reconcile co-lending transactions

Partner banks may have different:

- Minimum/maximum loan amounts
- Interest rate requirements
- Documentation requirements
- Turnaround time expectations

Always check partner-specific rules before processing.

### NBFC Compliance Officer

You are an NBFC compliance and reporting assistant.

Monitoring areas:



- Capital Adequacy Ratio (CRAR) – minimum 15%
- Asset classification (Standard/Sub-standard/Doubtful/Loss)
- NPA recognition as per RBI norms (90 days overdue)
- Fair Practice Code compliance
- KYC/AML compliance status
- CERSAI registration status
- Credit Bureau reporting (CIBIL/Equifax/Experian)

Generate reports for:

- Monthly NPA report
- Quarterly CRAR calculation
- Annual compliance checklist
- RBI return data preparation

Flag any violation immediately with severity level.

## Loan Application Management via Chat

### Common Queries

#### Daily Operations

##### "Aaj ke pending loan applications dikhao"

→ Lists all applications in pending status with applicant names and amounts

##### "Ramesh Kumar ka loan application status kya hai?"

→ Searches by applicant name, shows current stage, documents pending, CIBIL score

##### "Is month mein kitna disbursement hua?"

→ Runs database query for total disbursement amount this month

##### "NPA classification report banao"

→ Generates report with Standard/Sub-standard/Doubtful/Loss categories

##### "CIBIL score 600 se kam wale applications dikhao"

→ Filters and lists high-risk applications

## Co-Lending Operations

Niv AI can manage co-lending workflows:

- Match eligible loans to partner banks based on criteria
- Track approval status from co-lending partners

- Calculate interest splits and fee distributions
- Generate MIS reports for partner bank submissions
- Reconcile disbursement and collection flows

## Compliance & Reporting

---

NBFC-specific compliance queries Niv AI can handle:

- **NPA Report** — Classify assets by days past due (DPD)
- **CRAR Calculation** — Capital to risk-weighted assets ratio
- **Collection Efficiency** — Track recovery rates by bucket
- **Disbursement MIS** — Monthly disbursement trends
- **KYC Compliance** — Track KYC document completeness

# Billing & Token Management

Track usage, manage budgets, accept payments

## Shared Pool vs Per User

### Shared Pool Mode (Default)


- Company buys a pool of tokens
- All users consume from the same pool
- Admin can set per-user daily limits to prevent abuse
- Best for: Small teams, companies with centralized budgets

### Per User Mode

- Each user has their own token wallet
- Users can purchase tokens via Razorpay
- Or admin allocates tokens manually
- Best for: Large organizations, pay-per-use models

## Razorpay Integration

Enable real payments for token purchases:

1. Create Razorpay account at **razorpay.com**
2. Get API Keys from Dashboard → Settings → API Keys
3. Enter in Niv Settings →  Payment Gateway:
  - Razorpay Key ID: `rzp_test_...` (test) or `rzp_live_...` (production)
  - Razorpay Key Secret: Your secret key
  - Enable Razorpay Payments: ☒

 **Demo Mode**

Leave Razorpay keys empty for **demo mode** — payment flow works but no real charges. Perfect for testing and demonstrations.

## Token Cost Configuration

Setting	Description	Example (Mistral Medium)
Cost per 1K Input Tokens	User messages cost	₹0.80
Cost per 1K Output Tokens	AI response cost	₹2.40
Per User Daily Limit	Max tokens per day	50,000

## Usage Tracking

Every conversation tracks:











- Total tokens used (input + output)
- Message count
- Token cost in configured currency
- Per-conversation breakdown

# Administration

Managing and monitoring your Niv AI installation

## Niv Settings Reference

Complete reference for all Niv Settings fields:

Section	Fields	Purpose
 AI Configuration	Provider, Model, System Prompt, Tokens, Tools	Core AI behavior
 Rate Limiting	Per Hour, Per Day, Custom Message	Prevent abuse
 Telegram Bot	Token, Webhook URL, Enable	Telegram integration
 WhatsApp Bot	Phone ID, Access Token, Verify Token	WhatsApp integration
 Widget Settings	Position, Title, Color, Roles	Floating chat button
 Billing	Mode, Pool Balance, Limits, Costs	Token management
 Payment Gateway	Razorpay Keys, Currency	Real payments
 Voice Settings	STT/TTS Engine, Voice, Language	Voice chat
 Voice API	API Key, Base URL, Voice, Model	External TTS/STT
 Image Generation	Enable, API Key, URL	AI image creation

## Rate Limiting

Protect your AI budget and API limits:

- **Messages per Hour** — Recommended: 30-60 per user
- **Messages per Day** — Recommended: 200-500 per user
- **Custom Message** — Friendly message shown when limit is hit

## User Permissions

---

Control who can use Niv AI:

- **Allowed Roles** — Add roles that can access Niv AI (empty = all users)
- **Developer Mode** — System Manager only (enforced by role check)
- **Per-User Tool Permissions** — When enabled, tools run with user's own permissions

## Error Monitoring

---

Monitor Niv AI health:

- Check **Error Log** for "Niv AI" or "Niv Telegram" errors
- Use `monitor_errors` tool via chat: "Any errors in last 24 hours?"
- Check `/home/frappe/frappe-bench/logs/telegram.log` for Telegram-specific logs
- MCP tool failures logged to `frappe_assistant_core.*.log`

# Deployment Guide

Putting Niv AI to production

## Production Deployment (AWS/VPS)

### Requirements

- Ubuntu 22.04+ server (2+ CPU, 4GB+ RAM)
- Domain name with DNS pointing to server
- SSL certificate (Let's Encrypt / Certbot)
- ERPNext already running on the server

### Installation Steps

```
SSH into your server
ssh user@your-server

Install Niv AI
cd frappe-bench
bench get-app https://github.com/kulharir7/niv_ai.git
bench get-app https://github.com/buildswithpaul/Frappe_Assistant_Core.git
bench --site your-site install-app frappe_assistant_core
bench --site your-site install-app niv_ai
bench build --app niv_ai
sudo supervisorctl restart all
```

### Nginx SSE Configuration

Niv AI uses Server-Sent Events (SSE) for streaming. Add to your nginx config:

```
In your site's nginx config (usually /etc/nginx/conf.d/your-site.conf)
Add inside the location / block:

location /api/method/niv_ai.niv_core.api.stream.stream_chat {
 proxy_pass http://frappe-bench;
 proxy_buffering off;
 proxy_cache off;
 proxy_set_header Connection '';
}
```

```
proxy_http_version 1.1;
chunked_transfer_encoding off;
proxy_read_timeout 300s;
}
```

## Telegram/WhatsApp Webhook Setup

---

On production, no ngrok needed — your server is already public:

```
Set webhook URL to your domain
In Niv Settings:
Webhook URL: https://erp.yourdomain.com

Telegram webhook auto-registers at:
https://erp.yourdomain.com/api/method/niv_ai.niv_core.api.telegram.webhook

WhatsApp webhook (set in Meta Dashboard):
https://erp.yourdomain.com/api/method/niv_ai.niv_core.api.whatsapp.webhook
```

## Backup & Recovery

---

Niv AI data is stored in standard Frappe DocTypes:

- **Niv Conversation** — Chat conversations
- **Niv Message** — Individual messages
- **Niv Trigger** — Auto-pilot triggers
- **Niv System Prompt** — Reusable prompts
- **Niv Telegram User** — User mappings
- **Niv WhatsApp User** — User mappings

All backed up with standard `bench backup`. No external databases needed.



# API Reference

Complete API documentation for developers

## Chat API

### Stream Chat (SSE)

```
POST /api/method/niv_ai.niv_core.api.stream.stream_chat
Content-Type: application/json

{
 "conversation_id": "abc123",
 "message": "Show me today's sales",
 "model": "mistral-medium-2508", // optional
 "provider": "mistral", // optional
 "dev_mode": 1 // optional, System Manager only
}

Response: Server-Sent Events stream
data: {"type": "token", "content": "Today's"}
data: {"type": "token", "content": " total"}
data: {"type": "tool_call", "tool": "run_database_query", "arguments": {...}}
data: {"type": "tool_result", "tool": "run_database_query", "result": "..."}
data: {"type": "token", "content": "sales are ₹4,58,000"}
data: {"type": "done"}
```

## Conversation API

```
List conversations
GET /api/method/niv_ai.niv_core.api.conversation.list_conversations

Get messages
GET /api/method/niv_ai.niv_core.api.conversation.get_messages
 ?conversation_id=abc123&limit=50

Create conversation
POST /api/method/niv_ai.niv_core.api.conversation.create_conversation
```

```
Delete conversation
POST /api/method/niv_ai.niv_core.api.conversation.delete_conversation
{"conversation_id": "abc123"}

Search conversations
GET /api/method/niv_ai.niv_core.api.conversation.search_conversations
?query=sales&limit=10
```

## Voice API

```
Text to Speech
POST /api/method/niv_ai.niv_core.api.voice.text_to_speech
{"text": "Hello, how can I help?", "voice": "hi_IN-priyamvada-medium"}
→ Returns audio file URL

Speech to Text (base64)
POST /api/method/niv_ai.niv_core.api.voice.voice_chat_base64
{"audio_base64": "...", "conversation_id": "abc123"}
→ Returns {response, audio_url, conversation_id}

TTS Status
GET /api/method/niv_ai.niv_core.api.voice.get_tts_status
→ Returns available engines and voices
```

## Telegram Webhook

```
Webhook (called by Telegram)
POST /api/method/niv_ai.niv_core.api.telegram.webhook
(allow_guest=True, receives Telegram Update JSON)

Setup webhook
POST /api/method/niv_ai.niv_core.api.telegram.setup_webhook
(System Manager only)

Link user
POST /api/method/niv_ai.niv_core.api.telegram.link_user
{"telegram_user_id": "123456", "frappe_user_email": "user@example.com"}
```

## WhatsApp Webhook

```
Webhook (called by Meta)
GET /api/method/niv_ai.niv_core.api.whatsapp.webhook (verification)
POST /api/method/niv_ai.niv_core.api.whatsapp.webhook (messages)

Generate QR Code
POST /api/method/niv_ai.niv_core.api.whatsapp.generate_qr
```

```
 {"phone_number": "918322566235", "prefill_text": "Hi"}

Link user
POST /api/method/niv_ai.niv_core.api.whatsapp.link_user
 {"whatsapp_number": "918322566235", "frappe_user_email": "user@example.com"}
```

## Billing API

---

```
Get credit plans
GET /api/method/niv_ai.niv_billing.api.payment.get_plans

Create payment order
POST /api/method/niv_ai.niv_billing.api.payment.create_order
 {"plan_id": "PLAN-001"}

Verify payment
POST /api/method/niv_ai.niv_billing.api.payment.verify_payment
 {"razorpay_order_id": "...", "razorpay_payment_id": "...", "razorpay_signature": "..."}

Usage report (Admin)
GET /api/method/niv_ai.niv_billing.api.admin.get_usage_report
```

# Prompt Library

Ready-to-use system prompts for different industries

## General Business

### Business Analyst

You are a business analyst assistant for ERPNext. You help users understand their business metrics through data queries.

Focus areas:

- Sales trends and forecasting
- Inventory optimization
- Cash flow analysis
- Customer segmentation
- Profitability by product/customer/region

Always provide:

1. The exact numbers from the database
2. Comparison with previous period (MoM, YoY)
3. Brief insight or recommendation
4. Visual representation suggestion (chart type)

Use Hindi/Hinglish when the user prefers.

### Manufacturing Assistant

You are a manufacturing operations assistant.

Capabilities:

- Track Work Orders and their completion status
- Monitor BOM (Bill of Materials) costs
- Check raw material availability
- Calculate production capacity utilization
- Track quality inspection results
- Generate production planning reports

Rules:

- Always check material availability before suggesting production
- Flag Work Orders behind schedule
- Monitor scrap/waste rates
- Ensure quality checks are completed before marking done

### Healthcare Assistant

You are a healthcare facility management assistant.

Areas:

- Patient appointment scheduling
- Practitioner availability
- Medical records search
- Billing and insurance claims
- Inventory of medical supplies
- Lab test tracking

IMPORTANT: Always maintain patient confidentiality.

Never display sensitive medical information without proper authorization.

Follow HIPAA/local healthcare data protection guidelines.

### Education Institute

You are an education management assistant.

Features:

- Student enrollment tracking
- Fee collection status
- Attendance monitoring
- Exam results and grade calculation
- Course scheduling
- Faculty workload management

Help administrators with:

- "Kitne students ki fees pending hai?"
- "Section A ka attendance report do"
- "Next week ke exam schedule mein conflict check karo"

## NBFC / Finance

### Portfolio Manager

You are a loan portfolio management assistant for an NBFC.

Key metrics to track:

- AUM (Assets Under Management)
- Portfolio yield
- NPA ratio (Gross and Net)
- Collection efficiency ratio
- Bounce rate
- DPD (Days Past Due) distribution
- Provision coverage ratio

Report formats:

- Daily: Collections summary, new disbursements
- Weekly: DPD movement, bounce analysis
- Monthly: Portfolio quality, NPA trends
- Quarterly: Regulatory reporting data (RBI)

## Legal & Compliance

You are a legal and compliance assistant for an NBFC.

Track:

- Arbitration cases and hearing dates
- SARFAESI notices (Section 13(2), 13(4))
- DRT applications status
- One-time settlement (OTS) negotiations
- Lok Adalat case listings
- Legal notice generation data

Compliance:

- RBI master directions adherence
- Fair Practice Code violations
- Ombudsman complaints
- CERSAI charge registration
- ROC filings

Alert on:

- Hearing dates within 7 days
- Overdue compliance deadlines
- High-value NPA accounts without legal action

# Troubleshooting

Common issues and their solutions

## Common Issues

---

### 1. Chat shows "Error" or no response

- Check AI Provider settings — is API key valid?
- Check Error Log for "Niv AI Agent" errors
- Verify model name is correct for your provider
- Check token balance if billing is enabled

### 2. Tools not working

- Verify Frappe Assistant Core is installed: `bench --site your-site list-apps`
- Check MCP tool discovery: `bench console → MCPClient().discover_tools()`
- Check "Enable Tools" is ON in Niv Settings
- Look for MCP errors in Error Log

### 3. 502 Bad Gateway after restart

- Backend container IP changed — restart frontend: `docker restart frappe_docker-frontend-1`
- Re-add SSE nginx config (lost on container restart)
- Use `scripts/full_restore.ps1` for complete recovery

### 4. Telegram bot not responding

- Check ngrok is running (if testing locally)
- Verify webhook is set: `curl https://api.telegram.org/bot{TOKEN}/getWebhookInfo`
- Check Niv Telegram User exists for the sender's Telegram ID
- Check Error Log for "Niv Telegram Bot Error"

### 5. Voice mode not working

- Browser STT requires HTTPS (or localhost)
- Piper TTS needs pip install after container restart: `pip install piper-tts`
- Check voice model files exist in `sites/frontend/private/piper_models/`

## 6. pip packages lost after Docker restart

```
Reinstall on all containers:
for container in backend queue-short queue-long scheduler; do
 docker exec frappe_docker-${container}-1 pip install langchain langchain-core \
 langgraph langchain-openai piper-tts
done
```

## 7. Apps.txt overwrite after bench build



```
If niv_ai disappears from apps.txt after build:
docker exec frappe_docker-backend-1 bash -c \
 "echo -e 'frappe\nerpnext\nniv_ai\nfrappe_assistant_core\nniv_tools' > \
 /home/frappe/frappe-bench/sites/apps.txt"
```



# Roadmap

here Niv AI is heading

## Current Version: v0.5.1

Version	Focus	Key Features
v0.5 	Developer Tools + Triggers	94 RAG features, auto-pilot triggers, confirmation/undo
v0.5.1 	Multi-Channel	Telegram bot, WhatsApp bot, voice fixes, settings UI
v0.6	Permissions + Observability	Role-based tools, agent run logging, audit trail
v0.7	Production Deployment	Growth System deployment, NBFC testing, performance
v0.8	Mobile + Multi-language	Mobile-optimized widget, regional language prompts
v1.0	Stable Release	All core features stable, documentation complete
v1.5	Multi-Agent	Multiple specialized agents, agent orchestration
v2.0	Enterprise	SSO, audit trail, compliance, plugin marketplace

## Upcoming Features

- **Bulk Operations** — "Sab draft invoices submit karo" with confirmation
- **Natural Language Reports** — Charts and tables rendered in chat
- **Document Intelligence** — Upload PDF → auto-create Purchase Invoice
- **Permission-Aware Tools** — Role-based tool filtering
- **Agent Run Logging** — Track every AI action with cost and duration
- **Self-Test Loop** — AI verifies its own creations work
- **Multi-Agent Orchestration** — Specialized agents for different departments
- **Plugin Marketplace** — Community-contributed tools and prompts

**Built with ❤️ for the ERPNext Community**

Niv AI — Making ERPNext intelligent, one conversation at a time.

© 2026 Ravindra Kulhari | MIT License  
[github.com/kulharir7/niv_ai](https://github.com/kulharir7/niv_ai)