

JanSaarthi – India’s AI Bridge to Governance

Problem Statement

A large portion of rural India remains digitally underserved due to literacy barriers, language diversity, and limited connectivity. Existing platforms often exclude non-tech-savvy users. **JanSaarthi** solves this with a Gen-AI-powered assistant—accessible via mobile app, WhatsApp, and toll-free IVR—offering voice, text, and image support in regional languages to ensure inclusive, real-time access to government services.

Target Audience

JanSaarthi is built for **India’s rural and underserved citizens** who face challenges like low digital literacy, language diversity, and **limited smartphone or internet access**. These users often struggle with existing government platforms. JanSaarthi empowers them with a simple, multilingual, AI-powered assistant accessible via voice, text, and image—bridging the gap between citizens and public services.

Relevance of Problem




Millions in rural India are excluded from accessing government services due to language barriers, poor digital literacy, and lack of internet. Existing platforms fail to serve these users effectively. **JanSaarthi** addresses this gap with a multilingual, voice-first AI solution—making governance truly inclusive, accessible, and aligned with the vision of Digital India.

Use of Gen-AI

Generative AI forms the backbone of **JanSaarthi’s intelligent, multilingual civic interface**. The system employs **Bhashini ASR** and **Whisper** for robust, real-time transcription of regional voice inputs. **Bhashini NLP** powers deep cross-lingual semantic parsing, entity recognition, and transliteration. **Tesseract OCR** enables multimodal input by extracting structured data from images and scanned documents. **GPT-4 APIs** drive advanced intent classification and dynamic content generation, while **fine-tuned LLMs** ensure contextual, human-readable responses across diverse queries—government schemes, legal FAQs, and grievances. These are further synthesized into natural-sounding voice outputs using **neural Text-to-Speech (TTS)** models. This Gen-AI stack facilitates real-time, hyper-personalized, and voice-first service delivery—making JanSaarthi inclusive, adaptive, and accessible for users regardless of digital literacy or connectivity constraints.

Solution Framework: How JanSaarthi Works

JanSaarthi runs on a powerful **shared backend system** that supports **three smart user interfaces**:

-  A user-friendly **Mobile App**
-  A voice-first **Toll-Free IVR system**
-  An accessible **WhatsApp Chatbot**

Together, they ensure **multilingual, seamless access to government services**—no matter the user’s literacy level or device.

Multiple Ways to Ask, One Intelligent Brain to Understand




Users can interact in the way most natural to them:

- **Speak** in their regional language (processed via **Bhashini ASR** and **Whisper**)
- **Type** text queries
- **Send images or scanned documents** (read using **Tesseract OCR**)

All inputs are passed through **Bhashini NLP**, which performs smart **language detection, semantic analysis, and normalization**, enabling the system to truly understand regional and non-standard expressions.

Intelligent Intent Classification

Once a query is processed, it flows through a **GPT-based intent classifier**, which identifies the user’s need:

-  Searching for **government schemes**
-  Filing or tracking a **grievance**
-  Seeking **legal assistance**

Seamless Integration with Government Systems

Based on user intent, JanSaarthi connects to:

- **UMANG/API Setu** for real-time scheme information
- **CPGRAMS** APIs to register and track grievances
- **Firebase Firestore** for legal FAQs and citizen rights resources

Smart, Human-Centered Responses

A **fine-tuned GPT-4 model** generates clear, contextual, and easy-to-understand responses. These are delivered in formats tailored to the user:

- 🗣️ **Voice replies** (via **Bhashini TTS**) for IVR users
- 💬 **WhatsApp or SMS** messages
- 📄 **PDF documents** for official records

💡 **Inclusive Features Across All Platforms**

- 📱 **Mobile App**: Offers voice/text/image inputs, user dashboard, and grievance tracking
- 📞 **Toll-Free IVR**: Enables low-literacy users to talk directly to an AI voice agent using **Asterisk**
- 💬 **WhatsApp Chatbot**: Mirrors app capabilities and connects rural users easily

🔄 **Built to Learn & Scale**

JanSaarthi is always evolving. With continuous feedback loops and **AI-driven self-improvement**, the system:

- Learns from real-world usage
- Ensures **real-time synchronization** across platforms
- Delivers consistent and reliable performance—even in **low-connectivity rural areas**

Feasibility

JanSaarthi is built for national-scale deployment with a modular, cloud-native architecture. It seamlessly integrates government-backed platforms like **Bhashini (multilingual ASR/NLP)**, **API Setu (UMANG)**, and **CPGRAMS**, ensuring **regulatory compliance** and smooth interoperability. A **shared backend** powers mobile, WhatsApp, and IVR interfaces uniformly, minimizing engineering effort. Optimized for **low-bandwidth rural areas** via **BharatNet** and **CSC networks**, JanSaarthi is both **inclusive** and **cost-effective**.

Conclusion / MLP & Business Potential

JanSaarthi is more than a chatbot—it's a transformative civic-tech initiative.

Designed to bridge the gap between governance and citizens, it delivers multilingual, voice-first public service access powered by Generative AI. Whether it's legal guidance or access to welfare schemes, JanSaarthi is launched as a **Minimum Lovable Product (MLP)** with clear scalability. As a **SaaS solution for government bodies**, it enables API licensing, intelligent IVR support, and region-specific administrative dashboards—positioning itself as a replicable and sustainable **public-private model** aligned with the vision of Digital India.

Implementation in the hackathon

0–4 Hours: Setup & Planning

- Finalize tech stack and APIs (Bhashini, CPGRAMS, UMANG, GPT-4).
 - Assign roles for backend, frontend, integrations.
 - Initialize GitHub repo, Firebase, and secure API credentials.
-

4–10 Hours: Core Backend Development

- Set up Node.js + Express backend to manage all user requests.
 - Integrate GPT-4 API for intent classification and response generation.
 - Create Firebase Firestore structure for user sessions and grievance tracking.
 - Mock UMANG and CPGRAMS API endpoints for early integration.
-

10–16 Hours: Multimodal Input Handling

- Integrate Bhashini ASR & Whisper for real-time voice-to-text processing.
 - Add Bhashini NLP for semantic understanding and translation.
 - Implement Tesseract OCR to support image-based input.
-

16–22 Hours: Mobile App Development

- Develop a lightweight React Native app UI.
 - Add modules for text, voice, and image input.
 - Connect frontend with backend and display structured outputs.
-

22–28 Hours: IVR System with Asterisk

- Simulate toll-free functionality using local Asterisk setup.
- Build voice pipeline (Voice → ASR → GPT → TTS → Voice Output).

- Enable voice-based grievance filing and information retrieval.
-

28–34 Hours: WhatsApp Bot Integration

- Build a prototype using WhatsApp Business Sandbox (or mock interface).
 - Implement message flows and connect to backend services.
 - Return structured outputs for all three use cases.
-

34–40 Hours: Testing & Feedback

- Conduct functional testing across app, IVR, and WhatsApp.
 - Validate end-to-end query flows: scheme info, grievance, legal FAQs.
 - Add error handling, fallback prompts, and refine chatbot accuracy.
-

40–44 Hours: UI/UX Final Touches

- Polish app design for clarity and accessibility.
 - Build grievance tracking dashboard in the mobile app.
 - Improve GPT output formatting, enhance PDF generation and delivery.
-

44–48 Hours: Final Feature Integration & Optimization

- Refine API integrations (UMANG, CPGRAMS, GPT pipelines).
- Add analytics for usage monitoring (Firebase/Google Analytics).
- Implement fallback handling for unsupported queries.
- Conduct UX testing for multilingual and low-literacy users.
- Final bug fixes and performance optimization across all channels.