📄 Software Requirements Specification (SRS)

Module: AI & Smart Assistant Features

Product: Gati CRM

Version: 1.0

Author: Phoenix360 Solutions Pvt. Ltd.

Date: [2025-07-08]

---

1. Introduction

1.1 Purpose

This document defines the software requirements for integrating AI-driven assistant features into Gati CRM. The goal is to enhance user productivity, assist with CRM navigation, simplify report generation, and enable smart automation using natural language understanding and machine learning.

1.2 Scope

The AI module will include:

- An interactive **AI chatbot assistant** for CRM navigation and help.
- A **report generation engine** powered by natural language processing (NLP).
- **Lead scoring algorithms** for prioritization based on historical behavior.
- **Action execution via chat** (Chat-to-Action).
- **AI-powered summary generation** of lead interactions.

This module is a part of Gati CRM and will be tenant-aware, role-based, and API-driven.

---

2. Overall Description

2.1 Product Perspective

This AI module will be a microservice integrated into the Gati CRM frontend and backend systems. It will consume internal APIs for CRUD operations, reporting, and contextual user data.

2.2 User Classes and Characteristics

User Role	Characteristics
Counselor	Needs quick access to leads, reports, and reminders
Admin/Manager	Needs deep reports, monitoring tools, and configuration
Marketing/Sales	Uses reports and insights for strategy and follow-up

2.3 Assumptions and Dependencies

- The CRM already exposes secure APIs for lead, task, and report data.
- A chat interface is available on all major CRM screens.
- NLP models are trained or integrated via OpenAI or similar LLM provider.
- Role and tenant-based permission systems are already implemented.

---

3. Functional Requirements

🔹 3.1 AI Chatbot Assistant

FR-1.1: The system shall provide a chat widget accessible across all CRM views.
FR-1.2: The chatbot shall respond to common queries like:

- "How to add a lead?"
- "Where is my today's follow-up list?"
  FR-1.3: The chatbot shall respond differently based on user role (e.g., admin vs counselor).
  FR-1.4: The chatbot shall provide clickable suggestions and links to CRM pages.
  FR-1.5: The chatbot shall allow command-based actions like:

- “Add a lead named Rohit interested in MBA”
- “Show all leads from today”

🔹 3.2 AI-Driven Report Generator

FR-2.1: The system shall accept natural language report queries like:

- "Show conversion rate last month"
- "Generate source-wise lead report"
  FR-2.2: An NLP parser shall convert the query into a predefined SQL/report structure.
  FR-2.3: The system shall display the result as a table, chart, or downloadable format (CSV, PDF).
  FR-2.4: User access control will filter the data based on tenant and role.
  FR-2.5: Report outputs shall be exportable via frontend options.
  FR-2.6: A report history log shall be maintained per user for audit and quick reuse.

🔹 3.3 Predictive Lead Scoring

FR-3.1: The system shall calculate a score for each lead based on:

- Source, number of counselor interactions
- Email/SMS open rates
- Region, time to respond, etc.
  FR-3.2: The scoring logic shall be customizable per tenant.
  FR-3.3: The lead list view shall support filtering and sorting by score.
  FR-3.4: Counselor dashboard shall highlight high-priority leads first.
  FR-3.5: The scoring engine shall periodically retrain on new data trends.

🔹 3.4 Chat-to-Action Execution

FR-4.1: The chatbot shall allow users to request action execution like:

- “Book a follow-up call with Ayesha for tomorrow at 3PM”

- “Send email to all leads from Kolkata”
  FR-4.2: These actions shall be executed by securely calling internal APIs.
  FR-4.3: A confirmation message shall appear after execution.
  FR-4.4: A rollback or undo option will be shown for 15 seconds post-action.

🔹 3.5 Auto Summary Generation

FR-5.1: On user command or dashboard view, the system shall generate a summary like:

> “Lead Rohit from Pune interacted 3 times, interested in BBA, status: Counselled”
> **FR-5.2**: Summary shall be generated using lead interaction logs and chat data.
> **FR-5.3**: Summaries will be stored for fast future access.
> **FR-5.4**: Summary visibility shall be restricted to relevant roles (counselor, manager).

---

4. Non-Functional Requirements

Requirement Type	Description
Performance	90% of chatbot and report queries should respond within 2 seconds.
Scalability	Should handle 1000+ concurrent users across tenants.
Security	Data access restricted by user role and tenant. Token-authenticated API access only.
Availability	99.9% uptime for AI features.
Maintainability	Modular components, separate model training pipeline.
Compatibility	Must work on both web and mobile CRM portals.

---

5. System Architecture Overview

- **Frontend**: Chatbot widget in Vue/React embedded into CRM layout
- **Backend**: Node.js/Go microservice for NLP → CRM API handling
- **NLP**: OpenAI / RAG-based model for report query understanding

- **Storage**: PostgreSQL or Athena for lead/report data; Redis for recent queries
- **Auth**: Existing auth token for session validation
- **Logs**: All AI interactions stored for monitoring and model retraining

---

6. Future Enhancements

- Voice-to-text integration in chatbot
- Proactive chatbot tips based on screen context
- Slack/WhatsApp chatbot support
- AI-generated follow-up recommendations
- Feedback rating for chatbot responses (to improve relevance)
