Technical Document Requirements

Summary

The present technical document aims to describe the current state of the IVY assistant, outline the business logic to be implemented, and present a structured proposal for the continuation of its development. It includes functional and non-functional requirements, user stories, and diagrams that clearly identify the system's components, flows, and needs. Finally, it provides a detailed diagnosis of the current system as a starting point for the process of improvement and software scaling.

Introduction

The present technical document aims to detail the current situation and propose the functional and technical evolution of the multichannel conversational assistant named IVY. This system was designed to automate the acquisition of clients interested in the company's products or services through natural interactions via text and voice, supported by technologies such as Next.js, ElevenLabs, N8N, and solid foundations in scalable software engineering principles.

Currently, IVY has an initial implementation that enables lead collection, automatic generation of informal proposals, integration with external services such as CRM and email, and text-to-speech conversion. However, an evolution is required to transform this functional prototype into a robust, modular, and reusable solution, capable of adapting to different conversational flows, integrating with additional channels and real-time audio, and scaling as a marketable product.

To this end, this document presents a technical proposal, which includes the functional and non-functional requirements, user stories, desired structure, conversation flows, and the vision of packaging the system as a configurable library. It also outlines a technical diagnosis of the existing system and provides a detailed description of the components that must be refactored or implemented to ensure its scalability, flexibility, and long-term maintainability.

What you MUST deliver:

- REVIEW TECHNICAL DOCUMENT
- Automations in N8N
    - Filling in data conceived from the conversational CRM into our indicated database (MongoDB)
    - Creation of client folder
    - Currency converter
    - Quote creator
    - Invoice creator
    - Client code generator
    - Product code generator
- Database in MongoDB

    - Client database (Table or client entity)
    - Database of prices, products (Table or entity of product code, prices, products, and description)
    - Database in MongoDB connected to automations
    - Downloadable conversation section, always updated
    - Client classification in the database as a status record for each client: new client, quoted client, client who already paid, client in creative process, finalized client (Subject to modifications)
- APP: next.js
    - Configure it in Typescript, according to new integration in n8n
- Voice and text assistant

What we deliver:

- Updated Ivy database
    - Instructions
    - Operations manual
    - Behavior
    - Habeas Data
    - Quoter/Negotiator
    - Actions
    - Confirmation of actions
    - Security
    - Sales contact
- Our work is in next.js (we need them to deliver the file in next.js so we can adapt it)
- Excel (EXPLANATION on facilitation to adapt and scale this to another internal database where we can do it without using Excel)
    - Excel VMX prices (subject to changes)
    - Excel global approximate prices (subject to changes)
    - Quote generator
    - Client code generator
    - Product code generator
- Automation ideas
- Visual front-end

Table of functional and non-functional requirements

Functional Requirements	
ID	Description
RF01	The assistant must be capable of initiating and maintaining text-based conversations through web interfaces.
RF02	The system must integrate voice responses using ElevenLabs based on the generated text.
RF03	It must support integration with additional channels (e.g., WhatsApp, WebRTC calls, or any alternative that enables real-time calls with the assistant).
RF04	The system must automatically identify the user's intent based on their initial message.
RF05	It must classify the client based on type of person (individual or legal entity), business category (personal brand, SME, etc.), and budget level (A, B, etc.).
RF06	It must not explicitly ask for the sector or type if it can be inferred from the context.
RF07	The system must generate informal proposals and send them via email.
RF08	It must create a unique folder for each client in Google Drive and grant upload permissions.
RF09	The proposals must include a summarized transcript of the conversation.
RF10	The system must query a currency conversion API and convert values to USD. It must support multiple currencies, especially COP, USD, and EUR.
RF11	Values must be stored standardized in USD in the database.
RF12	If a high-value client (types A, AA, AAA) is detected, the system must notify the human team.
RF13	In complex cases or when requested by the user, the system must hand over the conversation to a human agent (via WhatsApp or another channel).
RF14	Every conversation must maintain session context (threadId).
RF15	If the user leaves and returns, it must resume from the previous point (persistent state).
RF16	Webhooks must be implemented to send data to N8N, the CRM, and other systems.
RF17	All data must be validated before being sent.
RF18	Delivery confirmations must be implemented to ensure successful submission

RF19	The conversation logic (flows, questions, validations) must be configurable from: an external JSON file or a configuration endpoint. This enables assistant customization based on context.
RF20	The code must be packageable as an NPM library or exportable module.
RF21	It must accept external configurations for customization without modifying the core logic.
RF22	The system must be prepared to receive and initiate calls (WebSocket/WebRTC).
RF23	During the call or audio stream, the assistant must continue managing the interaction flow (voice mode) using ElevenLabs or a better alternative.
RF24	The system must be scalable to integrate a payment gateway. A Stripe-based payment link solution must be supported. https://www.vyrtiummarketing.com/payment?code=
RF25	The source code must be designed from the beginning as a reusable library (NPM).
RF26	It must be possible to import the assistant module into other projects with minimal external configuration.
RF27	The conversational flow logic must be definable via: local JSON file, remote endpoint returning the flow, or constructor initialization parameters.
RF28	The assistant must be deployable as an NPM module or embedded component in a Next.js frontend.
RF29	If the user responds ambiguously or confusingly, the system must ask clarifying questions or repeat the purpose of the interaction.
RF30	The assistant must allow different branching paths based on the product type or client budget, using configuration files.
RF31	The assistant must generate a downloadable or emailable transcript of the conversation, as a record for both client and sales team.
RF32	After payment confirmation, the system must display a closing message with additional options (e.g., “Would you like anything else?”).

Non-Functional Requirements	
ID	Description
RFN01	The architecture must support multiple simultaneous instances of the assistant.

RFN02	Maximum response time for any interaction: < 500 ms (excluding external services).
RFN03	Proposal generation and delivery: < 2 seconds after conversation ends.
RFN04	The entire system must follow SOLID principles.
RFN05	Code must be decoupled, modular, documented, and reusable.
RFN06	Clear separation between logic, integrations, configuration, and other components is required.
RFN07	If a webhook or external service fails, the system must log the error, retry if applicable, and notify the team by sending an alert via email including the designated email addresses.
RFN08	The user flow must never break due to third-party errors.
RFN09	All information must be transmitted encrypted (HTTPS).
RFN10	JWT authentication must be used for protected endpoints (optional: API Key generation with correspondence).
RFN11	The system must comply with the principle of least privilege.
RFN12	The system must allow language and currency customization per client, while storing data in Spanish in the database.
RFN13	Compatible with Next.js 14 (App Router).
RFN14	Compatible with TypeScript.
RFN15	Supports deployment on platforms like Vercel, Railway, or any Node.js-compatible environment.
RFN16	Full logging of every interaction in MongoDB (new client, quoted client, paying client, creative client, delivered client).
RFN17	Unique identifiers per client and conversation (threadId, userId) according to the Excel specification.

RFN18	Clear technical documentation (README, file structure, endpoints) or Swagger.io.
RFN19	The assistant must fully support multiple languages and currencies — not only currency conversion, but also text, date formats, and regional validations.
RFN20	If an external service fails (e.g., ElevenLabs, N8N), the system must continue the flow from the last stable point and show a friendly fallback.
RFN21	Although web-focused, the conversational experience must remain functional under slow or mid-range mobile connections.
RFN22	Every interaction, conditional decision, error, or exception must be logged in the database for later analysis.
RFN23	All stored data must include timestamps for full traceability.

Arquitectura propuesta

https://excalidraw.com/#json=bcq9y-ieqIPUmxarRexHw,NkyZLpRWZFCTNKFFyWzM_g