**KAIZEN APP**



*“Payments only with your voice”*

By Kaizen Code Team

Forma

El contenido generado por IA puede ser incorrecto.

Team:

* Denise Jacobo Garcia
* Alexandra Lizeth Ruiz Martinez
* Julio Andres Reyes Garcia
* Axell Jonathan Mendez Perez

**1. What is the problem?**

Users require a faster and more intuitive way to make transfers, without relying on complex menus or lengthy forms. Currently, many mobile financial applications present accessibility barriers, making them difficult to use for people with visual or motor disabilities, as well as for users who prefer to interact through voice commands. This limitation generates digital exclusion and restricts the autonomy of these groups. Therefore, it is essential to design an application that prioritizes social inclusion, ensuring that anyone, regardless of their abilities, can perform financial operations in a safe, accessible, and simple way.

**2. What technologies were used?**

* Frontend: Web/Mobile.
* Voice recognition: Browser Web Speech API (SpeechRecognition).
* Backend: Node.js with Express.
* Payments: Interledger Open Payments API.
* Communication: REST API (/api/transfer).

**3. What is the solution?**

KaizenApp removes accessibility barriers in financial transfers by integrating voice control with a secure and efficient architecture. Unlike traditional applications that rely on menus and lengthy forms, KaizenApp allows users, including people with visual or motor disabilities, to easily perform operations through spoken commands. The application not only translates voice into actionable instructions but also provides clear and accessible confirmations at each step, reducing errors and increasing user autonomy. Thanks to its integration with the Interledger API, transfers are executed quickly, securely, and compatibly with different financial systems, positioning KaizenApp as an innovative solution that combines social inclusion, accessibility, and cutting-edge technology.

**4. What are the benefits?**

* Speed: transfers with a single voice command.
* Accessibility: useful for people with hearing or visual difficulties.
* Intuitive experience: natural flow, without menus or unnecessary steps.
* Security: Using the Interledger Open Payments API for secure transactions.

**5. Architecture / Simple Stack**

1. Voice → Text: Speech recognition API.
2. Text → Analysis: Command parser (amount, currency, recipient).
3. Text → API: Request to backend (/api/transfer).
4. Backend Node.js: Integration with Interledger Open Payments.
5. Response → Screen: Result in the app (toast + text box).

**6. Essential Functions**

* Microphone button: to start voice recognition.
* Real-time voice-to-text conversion.
* Command detection (e.g., transfers).
* REST API to send request to backend.

**7. Data Dictionary**

**7.1 Entity: User**

|  |  |  |  |
| --- | --- | --- | --- |
| Attribute | Data Type | Description | Constraints |
| id\_usuario | INT (PK) | Unique user identifier | Auto-incremental |
| nombre | VARCHAR(100) | Full name of the user | Required |
| correo | VARCHAR(100) | User’s email address | Unique |
| moneda\_base | VARCHAR(10) | Main currency of the user (e.g., MXN, USD) | ISO 4217 |

**7.2 Entity: Transfer**

|  |  |  |  |
| --- | --- | --- | --- |
| Attribute | Data Type | Description | Constraints |
| id\_transferencia | INT (PK) | Unique transfer identifier | Auto-incremental |
| monto | DECIMAL(10,2) | Amount of money to transfer | Greater than 0 |
| moneda | VARCHAR(10) | Currency used in the transaction | ISO 4217 |
| destinatario | VARCHAR(100) | Name of the recipient | Required |
| estado | VARCHAR(20) | Current transaction status (pending, success, failed) | Limited values |
| fecha | DATETIME | Date and time of transfer creation | Automatic |

**7.3 Entity: Log\_Processing**

|  |  |  |  |
| --- | --- | --- | --- |
| Attribute | Data Type | Description | Constraints |
| id\_log | INT (PK) | Unique log identifier | Auto-incremental |
| texto\_comando | TEXT | Voice command converted into text | Required |
| resultado\_nlp | JSON | Object with extracted parameters (amount, currency, recipient) | Structured |
| respuesta\_api | JSON | Response obtained from Interledger Open Payments | Structured |

**8. Security and Compliance**

KaizenApp’s security is based on the use of the Interledger API, which incorporates multiple mechanisms to guarantee confidentiality, integrity, and interoperability of transactions.

**8.1 NAP Protocol**

Implements a secure NAP protocol, designed to connect disconnected financial systems.

It ensures clarity, control, and protection for the user in every transfer.

**8.2 RESTful Model**

The API is organized under a RESTful model, which facilitates resource and method management in a standardized way.

It enables the creation of payment functionalities without third-party processors, reducing the attack surface and improving operational efficiency.

**8.3 Interoperability**

Provides compatibility between stablecoins and traditional fiat currencies.

Reduces transaction costs and processing times, ensuring faster and safer operations across different payment networks.

**8.4 Connectors**

Connectors act as intermediaries, capable of receiving a money packet in one currency and forwarding it in another.

They ensure the transfer of value independently of the ledger, guaranteeing integrity in heterogeneous systems.

**8.5 Conditional Transfers**

Cryptographic protocols are employed that establish strict conditions for releasing funds.

Payments are only executed if these conditions are met, providing a robust anti-fraud mechanism.

**9. UML Diagrams**

**9.1 Use Case Diagram**

**Diagrama

El contenido generado por IA puede ser incorrecto.**

Actors:

* User
* Frontend System
* Backend Node.js
* Interledger API

**9.2 Class Diagram**

**Diagrama

El contenido generado por IA puede ser incorrecto.**

Main Classes:

1. User

Attributes: id, name, email, base\_currency

Methods: startCommand(), checkHistory()

1. Transfer

Attributes: id, amount, currency, recipient, status, date

Methods: validate(), execute(), updateStatus()

1. VoiceProcessor

Methods: convertVoiceToText(), cleanText()

1. CommandAnalyzer

Methods: extractAmount(), extractCurrency(), extractRecipient()

1. InterledgerAPI

Methods: createGrant(), createQuote(), executePayment()

**10. Design File**

**10.1 Interface Design**

Una captura de pantalla de un celular

El contenido generado por IA puede ser incorrecto.

**Pantalla de celular con una luz azul

El contenido generado por IA puede ser incorrecto.**

**10.2 Logos**

**Imagen que contiene Forma

El contenido generado por IA puede ser incorrecto.**

**11. Conclusion**

In the medium and long term, KaizenApp could evolve into a comprehensive platform for inclusive financial services. Beyond voice-based transfers, it envisions the incorporation of advanced technologies such as intelligent virtual assistants and biometric authentication, allowing users with visual or motor disabilities to interact more naturally and securely. Likewise, the application could integrate multilingual support, expanding its reach and adaptability.

On the social side, KaizenApp aspires to become a benchmark in inclusive banking, promoting the financial autonomy of sectors historically excluded from the digital ecosystem. In the future, the platform could expand to include accessible international payments, voice-based personal finance management, and integration with government services and social programs, thus consolidating a truly accessible, secure, and universal financial ecosystem.