

# Centro de Resiliencia Económica Argentina (Resi)

Proyecto + Actualizaciones — Septiembre 2025

---

## 1. Objetivo General

Crear un espacio digital gratuito, interactivo y visual, que empodere al usuario promedio argentino a mejorar su economía, producir alimentos sostenibles, ahorrar inteligentemente y planificar la vida familiar. Todo esto integrando inteligencia artificial práctica y personalizada, gamificación y comunidad.

---

## 2. Arquitectura Técnica Detallada

La plataforma está construida sobre una arquitectura desacoplada con un frontend moderno y un backend modular, preparada para escalar.

### 2.1 Backend (FastAPI)

La API está diseñada para ser robusta, escalable y fácil de mantener.

#### **main.py — Orquestador Principal**

- Inicializa FastAPI, configura middlewares (CORS), incluye routers de módulos y define endpoints globales.
- Contiene el “prompt de sistema” de la IA, que reúne contexto del usuario y del mundo real.

#### **database.py — Estructura de Datos**

- Define el esquema de la base de datos con SQLAlchemy ORM.

#### **routers/**

- `finance.py`, `family.py`, `cultivation.py` → lógica de negocio.
- `market_data.py` (nuevo) → conexión a APIs externas para datos económicos en tiempo real.

#### **dependencies.py — Funciones auxiliares**

#### **schemas.py — Validación de datos**

---

## 2.2 Frontend (Next.js & TypeScript)

La interfaz es una SPA rápida, reactiva y adaptable.

- `src/app/page.tsx` → página de inicio, organiza módulos y estados globales.
  - `OnboardingFlow.tsx` → guía para recolectar información del usuario.
  - `FamilyPlannerModule.tsx` → genera planes familiares con IA.
  - `ChatWindow.tsx` → chat flotante con texto y voz.
  - `useResiVoice.ts` → reconocimiento de voz del navegador.
  - `apiClient.ts` → cliente Axios centralizado.
- 

## 3. Flujo de Datos Crítico










Ejemplo: el usuario consulta “¿conviene comprar dólares hoy?”.

1. El frontend envía la pregunta al backend (`/chat`).
  2. El backend recopila:
    - Cotización del dólar (API externa).
    - Perfil de riesgo, metas y plan familiar del usuario.
    - Historial de chat.
  3. Se consolida el contexto y se envía a la IA (Gemini).
  4. La respuesta se guarda y se devuelve al chat en el frontend.
- 

## 4. Esquema de la Base de Datos

- **users**: email, onboarding completo, perfil de riesgo, metas.
  - **expenses**: id, descripción, monto, categoría, fecha, usuario.
  - **budget\_items**: id, categoría, monto asignado, usuario.
  - **saving\_goals**: id, nombre, meta, progreso, usuario.
  - **chat\_messages**: id, emisor, mensaje, timestamp, usuario.
  - **family\_plans**: id, plan alimenticio, presupuesto, sugerencias, fecha, usuario.
-

## 5. Roadmap de Módulos

1. Finanzas 
  2. Cultivo 
  3. Planificación Familiar 
  4. Chat con IA (datos en tiempo real) 
  5. Comunidad y Gamificación 
  6. Mercado Resiliente 
  7. Educación Financiera 
  8. Energías Renovables 
  9. Suscripción Premium 
- 

## 6. Historial de Actualizaciones

### Septiembre 2025 — Ruta 2: Conexión con el mundo real

- Nuevo módulo `market_data.py` para consumir APIs externas.
- La IA ahora integra datos económicos en tiempo real.
- Endpoint `/chat` actualizado para fusionar contexto del usuario y mundo real.

### Septiembre 2025 — Ruta 1: Hiper-personalización

- Onboarding ampliado con perfil de riesgo y metas a largo plazo.
- Planificador Familiar con más opciones y guardado persistente.
- Base de datos reestructurada para nuevo contexto.
- Historial de chat por usuario.
- IA actualizada para usar todo el contexto disponible.