PROYECTO IA MULTI-AGENTE

Arquitectura para ERP Retail

Guia de Diseno e Implementacion

Stack: Laravel + Python + LLM Local

Documento generado con asistencia de Claude AI

INDICE

1. VISION GENERAL DE LA ARQUITECTURA

El sistema multi-agente propuesto se estructura en capas para garantizar escalabilidad, mantenibilidad y separacion de responsabilidades. La arquitectura conecta el ERP existente en Laravel con una capa de agentes inteligentes desarrollada en Python.

Capas del Sistema

Capa	Componentes
Presentacion	ERP Laravel, Dashboard Agentes, API Exte
Gateway/Orquestador	Agent Router (FastAPI), Auth, Rate Limit
Agentes	Asistente, Alertas, Automatizaciones, Pr
Servicios Compartidos	LLM Local (Ollama), Vector Store, Messag
Datos	ERP Database + AI Database

Principios de Diseno

- Modelo local para control total de datos y privacidad
- Comunicacion asincrona via colas de mensajes
- Separacion entre ERP (PHP) y Agentes (Python)
- Base de datos dedicada para IA (embeddings, historial, modelos)

2. AGENTE ASISTENTE

El Agente Asistente es la interfaz conversacional principal del sistema. Permite a los usuarios consultar datos del ERP usando lenguaje natural.

Componentes

- Procesador de Consultas: Interpreta lenguaje natural y clasifica intenciones
- Tools (Herramientas): consultar_producto, generar_reporte, buscar_cliente, ver_inventario, estado_pedido, historial_ventas
- RAG (Retrieval): Documentacion ERP, manuales, FAQs

Casos de Uso

Tipo	Ejemplo de Consulta
Consulta stock	Cuantas unidades tenemos del producto X?
Reportes	Genera un reporte de ventas del mes
Estado pedido	Cual es el estado del pedido #1234?
Deudas	Quienes son los clientes con mayor deuda

3. AGENTE DE ALERTAS

El Agente de Alertas realiza monitoreo proactivo del sistema y genera notificaciones inteligentes basadas en reglas configurables.

Arquitectura

- Scheduler (Cron Jobs): Stock cada 15min, pagos cada hora, anomalias diario
- Motor de Reglas: Condiciones configurables con umbrales
- Analizador LLM: Prioriza, agrupa y sugiere acciones
- Canales de Salida: Email, SMS, Telegram, Dashboard

Tipos de Alertas

Categoria	Trigger	Prioridad
Stock critico	< 10% umbral	Alta
Pago vencido	> 30 dias	Media-Alta
Anomalia ventas	Desviacion > 2 sigma	Media
Pedido retrasado	> fecha estimada	Media

4. AGENTE DE AUTOMATIZACIONES

El Agente de Automatizaciones ejecuta acciones automaticas basadas en eventos o solicitudes, implementando workflows configurables.

Componentes

- Event Listeners: Escucha nuevos pedidos, cambios inventario, triggers usuario
- Workflow Engine: EVENTO -> CONDICION -> ACCION -> NOTIFICACION
- Approval Flow: Acciones criticas requieren aprobacion humana

Acciones Disponibles

Funcion	Descripcion
crear_orden_compra()	Reposicion automatica de inventario
enviar_email()	Comunicaciones automaticas
actualizar_precios()	Ajustes dinamicos de precios
generar_factura()	Creacion automatica de documentos
notificar_proveedor()	Comunicacion cadena suministro
programar_recordatorio()	Follow-ups automaticos

Ejemplo de Workflow

```
WORKFLOW: Reposicion Automatica
TRIGGER: stock(producto) < punto_reorden
IF: proveedor.activo = true
THEN:
  1. calcular_cantidad_optima()
  2. crear_orden_compra()
  3. enviar_email(proveedor)
  4. notificar(compras, "Nueva OC generada")
```

5. AGENTE PREDICTIVO

El Agente Predictivo genera predicciones de demanda, ventas y tendencias utilizando modelos de Machine Learning combinados con interpretacion LLM.

Pipeline de Datos

- ETL desde ERP: Extraccion automatica de datos historicos
- Limpieza y normalizacion: Preparacion de datos para ML
- Feature engineering: Creacion de variables predictivas

Modelos ML

Modelo	Algoritmo	Output	Frecuencia
Prediccion Demanda	Prophet/LSTM	Unidades/semana	Semanal
Segmentacion Clientes	Clustering	Grupos + estrategias	Mensual
Deteccion Anomalias	Isolation Forest	Alertas fraude	Tiempo real
Recomendacion	Collaborative Filter	Cross-sell	Por transaccion

Capa LLM (Interpretacion)

- Explicacion de predicciones en lenguaje natural
- Generacion de insights y recomendaciones
- Respuesta a preguntas sobre tendencias

6. INTEGRACION CON ERP LARAVEL

La integracion entre el ERP Laravel existente y la capa de agentes Python se realiza mediante multiples canales de comunicacion.

Canales de Comunicacion

Canal	Componentes
HTTP/REST	Controllers Laravel <-> FastAPI Gateway
Message Queue	Laravel Events -> RabbitMQ/Redis -> Pyth
Database	Read-Only Replica para acceso desde Pyth

Ejemplo: Servicio Laravel

```
// app/Services/AgentService.php
class AgentService {
    public function askAssistant(string $query): array {
        return Http::post(config('agents.gateway_url')
            . '/assistant', [
            'query' => $query,
            'user_id' => auth()->id(),
            'context' => $this->getERPContext()
        ])->json();
    }
}
```

Ejemplo: API Gateway Python

```
# gateway/main.py
from fastapi import FastAPI
app = FastAPI()

@app.post("/assistant")
async def assistant_endpoint(request: AssistantRequest):
    agent = AssistantAgent(llm=ollama_client)
    return await agent.process(request.query, request.context)
```

7. STACK TECNOLOGICO RECOMENDADO

Modelos Locales Recomendados

Modelo	RAM Min	Caso de Uso	Nota
Mistral 7B	16 GB	Tareas generales	RECOMENDADO
Llama 3 8B	16 GB	Razonamiento	Alternativa
Mixtral 8x7B	48 GB	Alto rendimiento	Si hay recursos
Phi-3	8 GB	Recursos limitados	Ligero

Stack Completo

CAPA EXISTENTE (mantener):

- Laravel 10/11 (PHP 8.2+)
- MySQL/PostgreSQL
- Redis (cache + queues)

CAPA DE AGENTES (nuevo):

- Python 3.11+
- FastAPI (API Gateway)
- LangChain / LlamaIndex
- Ollama (servidor LLM)
- ChromaDB / Qdrant (vector store)

CAPA ML/PREDICTIVA:

- scikit-learn
- Prophet (series temporales)
- PyTorch (deep learning)
- MLflow (tracking)

INFRAESTRUCTURA:

- Docker + Docker Compose
- RabbitMQ (mensajeria)
- Celery (tareas programadas)
- Nginx (reverse proxy)

Hardware Minimo Recomendado

Componente	Especificacion
CPU	8+ cores
RAM	32 GB (64 GB ideal)

GPU	NVIDIA 8+ GB VRAM (RTX 3060+) - opcional
Almacenamiento	500 GB SSD

8. ROADMAP DE IMPLEMENTACION

Fase 1: Fundamentos

- Configurar Ollama + Mistral 7B en servidor
- Crear API Gateway con FastAPI
- Integrar llamadas basicas desde Laravel
- Implementar Agente Asistente (MVP)

Fase 2: Monitoreo

- Sistema de alertas con reglas basicas
- Configurar cola de mensajes (Redis/RabbitMQ)
- Implementar dashboard de alertas

Fase 3: Automatizacion

- Desarrollar motor de workflows
- Implementar acciones automaticas criticas
- Crear sistema de aprobaciones

Fase 4: Predictivo

- Construir pipeline de datos
- Entrenar modelos de prediccion de demanda
- Integrar con decisiones de compra

9. TIPOS DE AGENTES IA (GUIA GENERAL)

9.1 Por Funcion Principal

Tipo	Uso
Asistente Conversacional	Soporte, consultas
Alertas/Monitoreo	Stock, anomalias, SLAs
Automatizacion	RPA, workflows
Predictivo	Demanda, churn, ventas
Analisis	BI conversacional
Busqueda/RAG	Knowledge base
Clasificacion	Tickets, emails
Generacion	Contenido, documentos
Validacion/QA	Tests, compliance
Recomendacion	Cross-sell, next action

9.2 Por Nivel de Autonomia

Nivel	Caracteristica	Ejemplo
Reactivo	Solo responde a inputs	Chatbot basico
Proactivo	Inicia acciones por triggers	Alertas
Semi-autonomo	Propone, humano aprueba	Ordenes compra
Autonomo	Decide y ejecuta solo	Trading, escalado

9.3 Por Arquitectura Tecnica

Tipo	Descripcion	Complejidad
Single Agent	Un agente, una tarea	Baja
Multi-Agent	Varios colaborando	Media-Alta
Hierarchical	Supervisor + workers	Alta
Swarm	Distribuido sin control central	Muy Alta

9.4 Por Area de Negocio

Area	Tipos de Agentes
Ventas	Lead scoring, propuestas, CRM assistant
Marketing	Contenido, segmentacion, campanas
Operaciones	Inventario, logistica, scheduling
Finanzas	Conciliacion, fraude, forecasting
RRHH	Screening CV, onboarding, consultas
IT	Helpdesk, monitoreo, incidents
Legal	Contratos, auditoria, regulaciones
Atencion Cliente	Soporte L1, escalamiento
Produccion	Mantenimiento predictivo, calidad
Compras	Proveedores, negociacion, ordenes

9.5 Por Capacidad Tecnica

Capacidad	Descripcion
Tool-Using	Usa APIs, bases de datos, calculadoras
Coding	Genera y ejecuta codigo
Browsing	Navega web, extrae informacion
Memory	Mantiene contexto largo plazo
Planning	Descompone tareas complejas
Reflection	Auto-evalua y mejora respuestas
Multimodal	Procesa texto, imagen, audio, video

10. MATRIZ DE RECOMENDACIONES

Evaluacion por Tipo de Agente

Tipo	Impacto	Dificultad	Tiempo	ROI
Asistente/Chat	Medio	Baja	Rapido	Medio
Alertas	Alto	Baja	Rapido	Alto
Automatizacion	Muy Alto	Media	Medio	Muy Alto
Predictivo	Muy Alto	Alta	Largo	Muy Alto
Generacion	Medio	Baja	Rapido	Medio
Analisis/BI	Alto	Media	Medio	Alto
Recomendacion	Alto	Media	Medio	Alto

Recomendacion por Madurez de IA

NIVEL 1 - Inicio:

- Chatbot FAQ
- Alertas simples
- Busqueda documentos
- Clasificacion basica

NIVEL 2 - Crecimiento:

- Automatizacion workflows
- Analisis predictivo
- Recomendaciones
- Generacion contenido

NIVEL 3 - Avanzado:

- Agentes autonomos
- Sistemas multi-agente
- Agentes de decision
- Optimizacion continua

11. CUADRO: USUARIOS, FRONTEND Y BACKEND

Por Tipo de Usuario

Usuario	Rol	Frontend	Backend	BD
Pescadero	Operario	Blade + Vue.js	Laravel	MySQL
Logistica	Operario	Blade + Vue.js	Laravel	MySQL
Administrativo	Operario	Blade + Vue.js	Laravel	MySQL
Gerente	Direccion	Vue.js completo	Laravel+FastAPI	PostgreSQL
Director	Direccion	Vue.js completo	Laravel+FastAPI	PostgreSQL

Funciones por Usuario

Usuario	Funciones Disponibles
Pescadero	CRUD + Alertas + Chat IA
Logistica	CRUD + Alertas + Chat IA
Administrativo	CRUD + Alertas + Chat IA
Gerente	Dashboards + KPIs + Alertas + Chat IA +
Director	Dashboards + KPIs + Alertas + Chat IA +

Frontend: Que se Mantiene y Que se Anade

FRONTEND ACTUAL (se mantiene):

- Formularios CRUD
- Tablas con filtros
- Calendario
- Reportes basicos

WIDGETS NUEVOS (se anaden con Vue.js):

- Widget Alertas (campanita con notificaciones)
- Widget Chat IA (burbuja de conversacion)

12. CUADRO: EVENTOS, ACCIONES Y BASES DE DATOS

Por Tipo de Evento/Accion

Evento	Usuario	Frontend	Backend	BD Lee	BD Escribe
Registrar venta	Pescadero	Blade	Laravel	MySQL	MySQL
Ver stock	Todos	Blade	Laravel	MySQL	-
Crear pedido	Logistica	Blade	Laravel	MySQL	MySQL
Facturar	Administrativo	Blade	Laravel	MySQL	MySQL
Ver dashboard	Gerencia	Vue.js	FastAPI	PostgreSQL	-
Chat con IA	Todos	Vue.js	FastAPI+Ollama	PostgreSQL	-
Recibir alerta	Todos	Vue.js	Laravel Echo	Redis	-
Config workflow	Gerencia	Vue.js	FastAPI	PostgreSQL	PostgreSQL
Ver predicciones	Gerencia	Vue.js	FastAPI+ML	PostgreSQL	-

Por Tipo de Alerta

Alerta	Quien Crea	Destinatarios	Canal
Stock bajo	Agente IA	Pescadero+Logistica+Gerencia	Push+Banner
Pago vencido	Agente IA	Administrativo+Gerencia	Push+Email
Pedido retrasado	Agente IA	Logistica+Gerencia	Push
Anomalia ventas	Agente IA	Gerencia	Push+Dashboard
Error datos	Agente IA	Responsable del dato	Push+Banner
Tarea asignada	Gerencia	Trabajador asignado	Push+Banner

13. CUADRO: AGENTES IA Y TIPOS DE BUSQUEDA

Por Agente IA

Agente	Funcion	Backend	BD Principal	BD Secund.
Asistente	Responde preguntas	FastAPI+Ollama	PostgreSQL	ChromaDB
Alertas	Detecta anomalias	FastAPI+Celery	PostgreSQL	Redis
Automatizacion	Ejecuta workflows	FastAPI+Celery	PostgreSQL	MySQL
Documentos	Busca en docs	FastAPI+Ollama	ChromaDB	-
Predictivo	Genera pronosticos	FastAPI+ML	PostgreSQL	-

Por Tipo de Busqueda

Tipo Busqueda	Agentes	Tecnologia	BD	Velocidad
SQL directo	Asistente, Alertas, Predictivo	SQLAlchemy	PostgreSQL	< 100ms
Tiempo real	Alertas	Laravel Echo	Redis	< 50ms
Semantica RAG	Documentos, Asistente	LangChain	ChromaDB	< 300ms
Series tiempo	Predictivo	Prophet/LSTM	PostgreSQL	< 500ms
Trigger/Evento	Automatizacion	Laravel Events	MySQL->PostgreSQL	< 200ms

Resumen de Bases de Datos

Base de Datos	Que Guarda	Agentes que la Usan
MySQL	Datos operativos ERP	Automatizacion (escribir)
PostgreSQL	Analitica, historicos, metricas	Todos (leer)
ChromaDB	Embeddings de documentos	Documentos, Asistente
Redis	Cache, umbrales, sesiones	Alertas, Asistente

14. EJEMPLOS PRACTICOS DE ALERTAS

Ejemplo 1: Alerta de Error (Precio = 0)

El agente de verificacion detecta automaticamente errores en los datos y crea alertas que persisten hasta que el error se corrija.

Paso	Quien	Accion
1	Agente IA	Detecta producto con precio = 0.00
2	Agente IA	Crea alerta asignada a responsable
3	Maria (Admin)	Ve alerta en su panel
4	Maria (Admin)	Corrige precio en ERP: 7.50 euros
5	Agente IA	Verifica precio > 0, cierra alerta

Caracteristicas de este tipo de alerta:

- Creacion: Automatica por el Agente IA
- Asignacion: Al responsable del dato (ej: administrativo de productos)
- Persistencia: No desaparece hasta que el dato se corrija
- Cierre: Automatico cuando la condicion se cumple (precio > 0)

Ejemplo 2: Alerta de Progreso (Llamadas CV)

Gerencia asigna tareas con seguimiento de progreso. El sistema actualiza automaticamente el estado X/Y.

Paso	Quien	Accion
1	Gerencia	Selecciona 20 CVs en dashboard
2	Gerencia	Asigna tarea a Laura (RRHH)
3	Laura	Ve alerta: 0/20 llamadas pendientes
4	Laura	Llama candidato, confirma cita, registra
5	Sistema	Actualiza automaticamente: 1/20
6	Laura	Completa todas las llamadas
7	Sistema	20/20 completadas, cierra alerta

Caracteristicas de este tipo de alerta:

- Creacion: Manual por Gerencia desde dashboard
- Asignacion: A trabajador especifico (ej: RRHH)
- Progreso: Muestra X/Y tareas completadas
- Actualizacion: Automatica cuando trabajador registra accion
- Cierre: Automatico cuando progreso = 100%

Comparativa de Tipos de Alerta

Tipo	Quien Crea	Categoria	Progreso	Cierre
Error (Precio=0)	Agente IA	Error/Anomalia	No	Auto (dato OK)
Progreso (CVs)	Gerencia	Seguimiento	Si (X/Y)	Auto (100%)
Vencimiento	Agente IA	Tiempo limite	No	Auto (pagado)
Manual	Usuario	Recordatorio	No	Manual

15. FRONTENDS ESPECIFICOS POR DEPARTAMENTO

Se crearan 11 frontends especificos con Vue.js, cada uno optimizado para las necesidades de su departamento. El ERP base (Blade + jQuery) se mantiene para funcionalidades que no tengan frontend especifico.

15.1 Lista de Frontends a Desarrollar

Frontend	Ruta	Funciones Especificas
1. RRHH	/rrhh/*	CVs, entrevistas, nominas, alertas perso
2. Administrativo	/admin/*	Facturas, pagos, cobros, alertas contabl
3. Tiendas	/tiendas/*	Ventas, stock, pedidos rapidos, alertas
4. Logistica	/logistica/*	Rutas, entregas, tracking, alertas envio
5. Compras	/compras/*	Proveedores, ordenes, precios, alertas s
6. Comercial	/comercial/*	Clientes, ofertas, visitas, alertas vent
7. Calidad	/calidad/*	Controles, incidencias, trazabilidad
8. Almacen	/almacen/*	Entradas, salidas, inventario, ubicacion
9. Contabilidad	/contabilidad/*	Asientos, balances, impuestos, cierres
10. Produccion	/produccion/*	Elaborados, recetas, costes, mermas
11. Gerencia	/gerencia/*	Dashboards, KPIs, predicciones, vision g

15.2 Caracteristicas Comunes de Todos los Frontends

- Desarrollados con Vue.js 3 + Inertia.js
- Diseno responsive con Tailwind CSS
- Widget de alertas integrado (campanita)
- Widget de chat IA integrado
- Conexion a API Laravel compartida
- Autenticacion y permisos por rol

15.3 Funciones Especificas por Frontend

RRHH:

- Gestion de candidatos y CVs
- Seguimiento de entrevistas (X/Y completadas)
- Control de nominas
- Alertas de personal (ausencias, vencimientos)

ADMINISTRATIVO:

- Facturacion y cobros
- Control de pagos a proveedores
- Alertas de vencimientos
- Conciliacion bancaria

TIENDAS:

- Punto de venta optimizado
- Consulta de stock en tiempo real
- Pedidos rapidos al almacen
- Alertas de productos (precio 0, stock bajo)

LOGISTICA:

- Gestion de rutas de reparto
- Tracking de entregas
- Confirmacion de recepciones
- Alertas de pedidos retrasados

COMPRAS:

- Gestion de proveedores
- Ordenes de compra automaticas
- Comparativa de precios
- Alertas de stock minimo

COMERCIAL:

- Cartera de clientes
- Seguimiento de ofertas
- Registro de visitas
- Alertas de oportunidades

CALIDAD:

- Controles de calidad
- Registro de incidencias
- Trazabilidad de productos
- Alertas de no conformidades

ALMACEN:

- Entradas y salidas de mercancia
- Control de ubicaciones
- Inventarios y recuentos
- Alertas de caducidades

CONTABILIDAD:

- Asientos contables
- Balances y cuentas
- Gestion de impuestos
- Cierres mensuales/anuales

PRODUCCION:

- Gestion de elaborados
- Recetas y escandallos
- Control de costes
- Alertas de mermas

GERENCIA:

- Dashboards con KPIs en tiempo real
- Graficos y predicciones IA
- Vision global de todos los departamentos
- Asignacion de tareas con seguimiento
- Chat IA avanzado con acceso a todos los datos

15.4 Orden de Desarrollo Sugerido

Fase	Frontend	Justificacion
Fase 1	Gerencia	Toma decisiones, necesita KPIs primero
Fase 2	Tiendas	Mayor volumen de usuarios
Fase 3	Administrativo	Gestiona alertas de pagos/cobros
Fase 4	RRHH	Gestion de personal y entrevistas
Fase 5	Logistica	Control de entregas y rutas
Fase 6	Compras	Ordenes automaticas y proveedores
Fase 7	Almacen	Control de stock y ubicaciones
Fase 8	Comercial	Seguimiento de clientes
Fase 9	Contabilidad	Gestion contable
Fase 10	Calidad	Controles e incidencias
Fase 11	Produccion	Elaborados y costes

15.5 Arquitectura Tecnica

Todos los frontends comparten:

- Backend unico: Laravel 11 con API REST
- Base de datos: MySQL (operativo) + PostgreSQL (analitica)
- Agentes IA: FastAPI + Ollama (compartidos)
- Alertas: Laravel Echo + WebSockets (compartido)
- Autenticacion: Laravel Sanctum con permisos por rol

15.6 ERP Base (Se Mantiene)

El ERP actual con Blade + jQuery se mantiene para:

- Funcionalidades sin frontend especifico asignado
- Usuarios con multiples roles
- Acceso de emergencia si falla un frontend especifico
- Migracion gradual (hasta que todos los frontends esten listos)

16. COMPATIBILIDAD MOVIL (PWA)

Todos los frontends Vue.js seran compatibles con dispositivos moviles mediante PWA (Progressive Web App), permitiendo acceso desde smartphones y tablets sin necesidad de publicar en App Store o Google Play.

16.1 Opciones de Acceso Movil

Opcion	Tecnologia	Resultado	Estado
Web Responsive	Vue.js + Tailwind	Navegador movil	Inmediato
PWA	Service Worker + Manifest	Instalable como app	Recomendado
App Hibrida	Ionic + Vue o Capacitor	App nativa	Opcional
App Nativa	Flutter / React Native	Stores (iOS/Android)	Futuro

16.2 PWA - Recomendado

PWA (Progressive Web App) es la opcion recomendada porque:

- Se instala desde el navegador (sin App Store)
- Funciona offline (con datos en cache)
- Recibe notificaciones push (alertas)
- Acceso rapido desde icono en pantalla
- Actualizaciones automaticas (sin reinstalar)
- Un solo desarrollo para web y movil
- Sin costes de publicacion en stores

16.3 Funciones por Dispositivo

Frontend	Dispositivo Principal	Funciones Movil
Tiendas	Movil/Tablet	Ventas rapidas, consulta stock, alertas
Logistica	Movil	Rutas, entregas, confirmaciones, GPS
Comercial	Movil/Tablet	Visitas clientes, pedidos, fotos
Almacen	Tablet	Entradas/salidas, escaneo, ubicaciones
RRHH	Escritorio	Gestion CVs, entrevistas (menos movil)
Gerencia	Tablet/Escritorio	Dashboards, KPIs, decisiones
Administrativo	Escritorio	Facturacion, contabilidad

16.4 Caracteristicas PWA por Frontend

TIENDAS (Alta prioridad movil):

- Punto de venta tactil optimizado
- Escaneo de codigos de barras con camara
- Consulta de precios y stock en tiempo real
- Alertas push de productos
- Funciona offline (sincroniza al conectar)

LOGISTICA (Alta prioridad movil):

- Lista de entregas del dia
- Navegacion GPS integrada
- Confirmacion de entrega con firma
- Foto de entrega como comprobante
- Alertas de pedidos urgentes

COMERCIAL (Media prioridad movil):

- Ficha de cliente con historial
- Registro de visitas con geolocalizacion
- Creacion de pedidos en campo
- Catalogo de productos con fotos
- Alertas de oportunidades

ALMACEN (Tablet recomendado):

- Escaneo de codigos para entradas/salidas
- Consulta de ubicaciones
- Inventario con contador tactil
- Alertas de caducidades proximas

16.5 Notificaciones Push

Las alertas del sistema llegaran como notificaciones push al movil:

- Stock bajo: Notificacion a tienda y almacen
- Pedido urgente: Notificacion a logistica
- Pago vencido: Notificacion a administrativo
- Tarea asignada: Notificacion al responsable
- Error detectado: Notificacion al encargado del dato

16.6 Requisitos Tecnicos PWA

Requisito	Funcion	Estado
HTTPS	Obligatorio para PWA	Ya disponible
Service Worker	Cache y offline	Configurar
Web Manifest	Icono y nombre app	Configurar
Push API	Notificaciones	Laravel + Firebase/Pusher
Responsive	Adaptacion pantalla	Tailwind CSS

16.7 Orden de Implementacion Movil

Fase	Accion	Objetivo
Fase 1	Todos los frontends responsive	Base para movil
Fase 2	PWA en Tiendas y Logistica	Mayor uso movil
Fase 3	PWA en Comercial y Almacen	Uso en campo
Fase 4	Notificaciones push	Alertas en tiempo real
Fase 5	Funciones offline	Sincronizacion diferida

17. FLUJOS DE TRABAJO Y COMUNICACION

Esta seccion detalla los flujos de comunicacion entre todos los componentes del sistema: usuarios, frontends, backends, agentes IA y bases de datos.

17.1 Flujo de Arquitectura General

Diagrama de capas del sistema completo:

Capa	Componente	Descripcion
Capa 1	USUARIOS	Gerencia, Operarios, Tiendas, etc.
Capa 2	FRONTENDS	Vue.js (11 apps) + ERP Base (Blade)
Capa 3	API GATEWAY	Laravel (REST) + FastAPI (IA)
Capa 4	SERVICIOS	Agentes IA, ML, Notificaciones
Capa 5	DATOS	MySQL + PostgreSQL + ChromaDB + Redis

Flujo de comunicacion entre capas:

Origen		Destino	Proposito
Usuario	->	Frontend	Acciones del usuario
Frontend	->	Laravel API	Peticiones HTTP/REST
Laravel API	->	MySQL	Operaciones CRUD
Laravel API	->	FastAPI	Consultas a agentes IA
FastAPI	->	PostgreSQL	Consultas analiticas
FastAPI	->	Ollama	Procesamiento LLM
Agentes IA	->	Redis	Cache y estado alertas
Laravel Echo	->	Frontend	Notificaciones push

17.2 Flujo de Frontend por Tipo de Usuario

Flujo para OPERARIOS (Tiendas, Logistica, Almacen):

Paso	Accion
1	Usuario abre app (PWA o web)
2	Frontend Vue.js carga datos desde Larave
3	Usuario realiza accion (venta, entrada,
4	Frontend envia POST a Laravel API
5	Laravel guarda en MySQL
6	Laravel dispara evento (si aplica)
7	Evento sincroniza a PostgreSQL (ETL)
8	Si hay alerta, Laravel Echo notifica en

Flujo para GERENCIA (Dashboard, KPIs, Chat IA):

Paso	Accion
1	Gerente abre dashboard Vue.js
2	Frontend solicita KPIs a FastAPI
3	FastAPI consulta PostgreSQL (datos agreg
4	Graficos ApexCharts renderizan datos
5	Gerente usa chat IA: pregunta algo
6	Frontend envia pregunta a FastAPI
7	FastAPI consulta PostgreSQL + ChromaDB +
8	Respuesta IA se muestra en chat

17.3 Flujo de Comunicacion de Agentes IA

Como se comunican los agentes entre si y con los usuarios:

Agente	Trigger	Backend	BD que Usa
Asistente	Recibe pregunta	FastAPI	PostgreSQL + ChromaDB
Alertas	Detecta anomalia	Celery (cron)	PostgreSQL + Redis
Automatizacion	Ejecuta workflow	Celery (evento)	PostgreSQL + MySQL
Predictivo	Genera pronostico	FastAPI (manual)	PostgreSQL
Documentos	Busca en docs	FastAPI	ChromaDB

Flujo de una ALERTA (desde deteccion hasta usuario):

Paso	Componente	Accion
1	Agente Alertas	Celery ejecuta job cada 15 min
2	Agente Alertas	Consulta PostgreSQL: SELECT productos WH
3	Agente Alertas	Encuentra anomalia: Lubina precio 0.00
4	Agente Alertas	Crea registro en tabla 'alertas' (Postgr
5	Agente Alertas	Publica evento en Redis (canal alertas)
6	Laravel Echo	Escucha Redis, detecta nueva alerta
7	Laravel Echo	Envia push via WebSocket al frontend del
8	Frontend Vue	Muestra notificacion en campanita del us

17.4 Flujo de Base de Datos (Sincronizacion)

Como se mantienen sincronizadas MySQL y PostgreSQL:

Paso	Componente	Accion
1	Operario	Registra venta en ERP
2	Laravel	INSERT en MySQL (tabla ventas)
3	Laravel	Dispara evento VentaCreada
4	Listener	Escucha evento, transforma datos
5	Listener	INSERT en PostgreSQL (fact_ventas)
6	PostgreSQL	Dato disponible para agentes IA
7	Agente	Consulta PostgreSQL para analisis

Estrategia de sincronizacion por tipo de dato:

Dato	Frecuencia	Metodo	Latencia
Ventas	Tiempo real	Evento Laravel	< 1 segundo
Stock	Tiempo real	Evento Laravel	< 1 segundo
Clientes	Cada 15 min	Job ETL	15 minutos
Productos	Cada 15 min	Job ETL	15 minutos
Historicos	Cada noche	Job nocturno	1 vez/dia

17.5 Flujo de Comunicacion Eficiente

Reglas para evitar cuellos de botella:

- MySQL solo para CRUD operativo (escrituras frecuentes)
- PostgreSQL solo para LECTURAS analiticas (nunca escribir desde ERP)
- Redis como intermediario de mensajes (no consultas directas)
- ChromaDB solo para busquedas semanticas (documentos)
- Agentes NUNCA escriben en MySQL directamente
- Alertas se cachean en Redis para no reconsultar

Flujo de escritura vs lectura:

Accion	Flujo	Caracteristica
Operario escribe	Frontend -> Laravel -> MySQL	Rapido, transaccional
Sync a analitica	MySQL -> Evento -> PostgreSQL	Asincrono, no bloquea
Gerencia lee	Frontend -> FastAPI -> PostgreSQL	Optimizado para lectura
Agente consulta	Agente -> PostgreSQL	Indices y vistas materializadas
Agente alerta	Agente -> Redis -> Laravel Echo	Tiempo real, sin BD

17.6 Flujo Completo: Ejemplo Practico

Escenario: Pescadero vende producto con precio 0, se detecta y corrige.

Paso	Quien/Que	Accion
1	Pescadero	Registra venta de Lubina en tienda
2	Laravel	Guarda venta en MySQL
3	Evento	VentaCreada -> sincroniza a PostgreSQL
4	Agente Alertas	Job cada 15 min: detecta precio = 0
5	Agente Alertas	Crea alerta en PostgreSQL
6	Redis	Publica mensaje: nueva alerta
7	Laravel Echo	Detecta mensaje, busca responsable
8	WebSocket	Envia push a Maria (admin productos)
9	Maria	Ve notificacion en su pantalla
10	Maria	Abre producto, corrige precio a 7.50
11	Laravel	UPDATE en MySQL (precio = 7.50)
12	Evento	ProductoActualizado -> sincroniza Postgr
13	Agente Alertas	Siguiente job: verifica condicion
14	Agente Alertas	Precio > 0 = TRUE, cierra alerta
15	Maria	Ve que la alerta desaparecio automaticam

17.7 Diagrama de Conexiones entre Componentes

Resumen de todas las conexiones del sistema:

Origen	Destino	Protocolo	Uso
Frontend Vue	Laravel API	HTTP/REST	Peticiones usuario
Frontend Vue	Laravel Echo	WebSocket	Alertas tiempo real
Laravel API	MySQL	TCP/SQL	CRUD operativo
Laravel API	FastAPI	HTTP/REST	Consultas IA
Laravel API	Redis	TCP	Cache y colas
FastAPI	PostgreSQL	TCP/SQL	Consultas analiticas
FastAPI	Ollama	HTTP	Procesamiento LLM
FastAPI	ChromaDB	HTTP	Busqueda semantica
Celery	Redis	TCP	Cola de tareas
Celery	PostgreSQL	TCP/SQL	Jobs de agentes
Laravel Echo	Redis	Pub/Sub	Mensajes tiempo real

17.8 Resumen de Puertos y Servicios

Servicio	Puerto	Protocolo	Funcion
Laravel API	8000	HTTP	API principal
FastAPI	8001	HTTP	Gateway IA
MySQL	3306	TCP	BD operativa
PostgreSQL	5432	TCP	BD analitica
Redis	6379	TCP	Cache/mensajes
Ollama	11434	HTTP	LLM local
ChromaDB	8002	HTTP	Vector store
WebSocket	6001	WS	Notificaciones

18. ANALISIS FLUJO RRHH - SISTEMA DE SELECCION

Esta seccion mapea el flujo de trabajo de seleccion de personal (RRHH) al sistema multi-agente, detallando que componentes intervienen en cada paso.

18.1 Vision General del Proceso

El proceso de seleccion de personal se divide en 6 fases principales, cada una con diferentes niveles de automatizacion e intervencion humana.

Fase	Tipo	Automatizacion	Componentes IA
1. Entrada CVs	Automatico	Alta	Agente Documentos + Automatizacion
2. Primera Criba	Automatico	Alta	Agente Alertas + Asistente
3. Segunda Criba	Semi-auto	Media	Frontend RRHH + Asistente
4. Llamadas	Manual	Baja	Frontend RRHH + Alertas
5. Entrevistas	Manual	Baja	Frontend RRHH
6. Reactivacion	Automatico	Alta	Agente Alertas

18.2 Fase 1: Entrada de CVs

Los CVs llegan por multiples canales y se procesan automaticamente.

Canal Entrada	Componente	Accion
Email (cv@empresa.com)	Agente Automatizacion	Detecta adjuntos PDF/DOC
Portal web	Laravel API	Formulario de candidatos
LinkedIn	Agente Automatizacion	Scraping autorizado
Agencias	Email/API	Integracion con ETTs

Stack tecnologico de esta fase:

Componente	Tecnologia	Funcion
Frontend	-	No hay interaccion de usuario
Backend	Laravel + FastAPI	Recepcion y procesamiento
Agente	Documentos + Automatizacion	Extraccion de texto del CV
BD Escribe	PostgreSQL	Tabla candidatos + embeddings
BD Lee	ChromaDB	Indexacion semantica del CV

Flujo detallado:

Paso	Accion
1	CV llega por email/web/API
2	Agente Automatizacion detecta nuevo docu
3	Agente Documentos extrae texto (OCR si e
4	LLM (Ollama) parsea: nombre, email, expe
5	Datos estructurados se guardan en Postgr
6	Embedding del CV se guarda en ChromaDB
7	Candidato queda listo para la primera cr

18.3 Fase 2: Primera Criba (Automatica)

El sistema filtra automaticamente CVs que no cumplen requisitos minimos.

Criterio	Valor Requerido	Tipo	Accion
Experiencia minima	3 anios en el sector	Automatico	Descarta < 3 anios
Ubicacion	Provincia de Malaga	Automatico	Descarta fuera de zona
Idiomas	Espanol nativo	Automatico	Verifica idioma CV
Palabras clave	Pescaderia, alimentacion	Automatico	Busqueda semantica
Disponibilidad	Inmediata/1 mes	Semi-auto	Si indica en CV

Stack tecnologico de esta fase:

Componente	Tecnologia	Funcion
Frontend	-	Proceso automatico en background
Backend	FastAPI + Celery	Jobs programados
Agente	Alertas + Asistente	Evaluacion y clasificacion
BD Lee	PostgreSQL + ChromaDB	Datos candidato + similitud
BD Escribe	PostgreSQL	Estado: aprobado/descartado

Resultado de la primera criba:

Estado	Color	Siguiente Paso
Aprobado	Verde	Pasa a segunda criba
Descartado	Rojo	Se archiva con motivo
Revision	Amarillo	RRHH decide manualmente

18.4 Fase 3: Segunda Criba (RRHH)

El responsable de RRHH revisa los candidatos preseleccionados con ayuda de IA.

Stack tecnologico de esta fase:

Componente	Tecnologia	Funcion
Frontend	Vue.js (RRHH)	Dashboard de candidatos
Backend	Laravel + FastAPI	API + consultas IA
Agente	Asistente	Responde preguntas sobre CVs
BD Lee	PostgreSQL + ChromaDB	Datos + busqueda semantica
BD Escribe	PostgreSQL	Notas y decisiones RRHH

Funciones del frontend RRHH en esta fase:

- Lista de candidatos aprobados en primera criba
- Vista detallada de cada CV con datos parseados
- Chat IA: 'Comparame estos 3 candidatos'
- Chat IA: 'Quien tiene mas experiencia en X?'
- Boton: Aprobar para llamada / Descartar
- Campo de notas internas por candidato

Interaccion con Agente Asistente:

Actor	Accion/Respuesta
RRHH pregunta	Quien tiene experiencia en pescaderia?
Asistente	Busca en ChromaDB por similitud semantic
Asistente	Responde: Juan (5 anios), Maria (3 anios
RRHH pregunta	Compara a Juan y Maria
Asistente	Genera tabla comparativa de skills y exp

18.5 Fase 4: Llamadas a Candidatos

RRHH contacta a los candidatos seleccionados para agendar entrevistas.

Stack tecnologico de esta fase:

Componente	Tecnologia	Funcion
Frontend	Vue.js (RRHH)	Gestor de llamadas
Backend	Laravel	CRUD de contactos
Agente	Alertas	Seguimiento progreso X/Y
BD Lee	PostgreSQL	Datos de contacto
BD Escribe	PostgreSQL	Resultado llamada + cita

Flujo de trabajo con alertas de progreso:

Paso	Quien	Accion
1	Gerencia/RRHH	Selecciona 20 candidatos para llamar
2	Sistema	Crea alerta de progreso: 0/20 llamadas
3	RRHH	Ve alerta en su panel con lista de pendi
4	RRHH	Llama a candidato, registra resultado
5	Sistema	Actualiza automaticamente: 1/20
6	RRHH	Candidato confirma, agenda fecha entrevi
7	Sistema	Guarda cita en calendario
8	Sistema	20/20 completado, cierra alerta

Resultados posibles de cada llamada:

Resultado	Accion	Siguiente
Cita confirmada	Se agenda entrevista	Pasa a Fase 5
No contesta	Se reintenta 2 veces	Queda pendiente
Rechaza oferta	Se descarta	Se archiva
Pide mas info	Se envia email	Queda pendiente

18.6 Fase 5: Entrevistas

Se realizan las entrevistas presenciales o por videollamada.

Stack tecnologico de esta fase:

Componente	Tecnologia	Funcion
Frontend	Vue.js (RRHH)	Agenda y evaluacion
Backend	Laravel	Calendario + notas
Agente	-	Sin IA en entrevista
BD Lee	PostgreSQL	Historial candidato
BD Escribe	PostgreSQL	Evaluacion entrevista

Funciones del frontend RRHH en esta fase:

- Calendario con citas programadas
- Ficha del candidato con todo su historial
- Formulario de evaluacion post-entrevista
- Puntuacion por competencias (1-5)
- Decision: Contratar / Segunda entrevista / Descartar
- Generacion de oferta laboral (si aplica)

Estados post-entrevista:

Estado	Accion	Nota
Contratado	Pasa a onboarding	Proceso finalizado
Segunda entrevista	Se agenda nueva cita	Vuelve a Fase 5
En espera	Pool de reserva	Posible reactivacion
Descartado	No apto para el puesto	Se archiva

18.7 Fase 6: Reactivacion de Candidatos

El sistema puede reactivar automaticamente candidatos del pool de reserva.

Stack tecnologico de esta fase:

Componente	Tecnologia	Funcion
Frontend	-	Proceso automatico
Backend	FastAPI + Celery	Jobs programados
Agente	Alertas	Detecta vacantes + matching
BD Lee	PostgreSQL + ChromaDB	Pool + similitud
BD Escribe	PostgreSQL	Reactivacion candidato

Triggers de reactivacion:

Trigger	Condicion	Accion
Nueva vacante	Se abre puesto similar	Busca en pool
Tiempo transcurrido	Candidato en espera > 3 meses	Pregunta si sigue interesado
Cambio requisitos	Se flexibilizan criterios	Reevalua descartados
Solicitud RRHH	Manual desde dashboard	Recupera candidato especifico

Flujo de reactivacion automatica:

Paso	Componente	Accion
1	Agente Alertas	Detecta nueva vacante en el sistema
2	Agente Alertas	Busca candidatos en pool con skills simi
3	ChromaDB	Devuelve top 10 por similitud semantica
4	Agente Alertas	Crea alerta para RRHH: 10 candidatos pot
5	RRHH	Revisa lista y decide quien recontactar
6	Sistema	Candidatos seleccionados vuelven a Fase

18.8 Integracion MySQL (ERP) y PostgreSQL (IA)

El flujo RRHH conecta datos existentes del ERP (MySQL) con el nuevo sistema de seleccion (PostgreSQL):

Tabla MySQL	Contenido	Uso en RRHH
trabajadores	Empleados actuales	Verificar si candidato ya trabaja
trabajadores_antiguos	Ex-empleados	Verificar si candidato ya trabajo antes
registro_horario	Fichajes	Consultar horarios al contratar
nominas	Datos salariales	Preparar oferta economica
vacantes	Puestos abiertos	Disparar busqueda de candidatos
departamentos	Estructura empresa	Asignar departamento al contratar

Datos nuevos en PostgreSQL (sistema de seleccion):

Tabla PostgreSQL	Contenido	Detalle
candidatos	Datos parseados del CV	Nombre, email, skills, experiencia
cvs_documentos	PDF original + embeddings	Busqueda semantica
proceso_seleccion	Estado de cada candidato	Fase actual, fechas, notas
llamadas	Registro de contactos	Resultado, fecha cita
entrevistas	Evaluaciones	Puntuacion, decision
alertas_rrhh	Notificaciones	Progreso, tareas pendientes

Momentos de integracion MySQL <-> PostgreSQL:

Momento	Accion	Detalle
Entrada CV	Lee MySQL	Verifica si email ya es empleado/ex-empl
Primera Criba	Lee MySQL	Descarta si ya trabajo y salio mal
Segunda Criba	Lee MySQL	Muestra historial si fue empleado antes
Entrevista OK	Lee MySQL	Consulta salarios del puesto para oferta
Contratacion	Escribe MySQL	INSERT en trabajadores + nominas
Nueva vacante	Lee MySQL	Trigger desde tabla vacantes del ERP

18.9 Resumen: Stack Completo por Fase

Tabla resumen incluyendo ambas bases de datos:

Fase	Frontend	Backend	Agentes	BD Lee	BD Escr
1. Entrada CVs	-	Laravel+FastAPI	Docs+Auto	MySQL+PG	PG
2. Primera Criba	-	FastAPI+Celery	Alertas+Asist	MySQL+PG	PG
3. Segunda Criba	Vue RRHH	Laravel+FastAPI	Asistente	MySQL+PG	PG
4. Llamadas	Vue RRHH	Laravel	Alertas	PG	PG
5. Entrevistas	Vue RRHH	Laravel	-	MySQL+PG	PG+MySQL
6. Reactivacion	-	FastAPI+Celery	Alertas	MySQL+PG	PG

Leyenda:

- MySQL = Base de datos ERP (trabajadores, nominas, vacantes)
- PG = PostgreSQL (candidatos, proceso seleccion, alertas)
- Chroma = ChromaDB (embeddings de CVs)
- Docs = Agente Documentos
- Auto = Agente Automatizacion
- Asist = Agente Asistente

Flujo de sincronizacion al CONTRATAR:

Paso	Accion	BD	Detalle
1	RRHH marca 'Contratar' en frontend	PostgreSQL	estado = contratado
2	Laravel detecta cambio de estado	-	Evento ContratacionAprobada
3	Listener crea registro empleado	MySQL	INSERT trabajadores
4	Listener crea registro nomina	MySQL	INSERT nominas
5	Sistema notifica a Gerencia	Redis	Push de confirmacion

18.10 Metricas y KPIs del Proceso

El sistema puede calcular automaticamente estos KPIs de RRHH:

KPI	Calculo	Fuente
CVs recibidos/mes	COUNT candidatos por mes	PostgreSQL
Tasa primera criba	% aprobados vs total	PostgreSQL
Tasa segunda criba	% aprobados vs primera criba	PostgreSQL
Tasa contacto	% llamadas exitosas	PostgreSQL
Tasa entrevista	% citas confirmadas	PostgreSQL
Tasa contratacion	% contratados vs entrevistados	PostgreSQL
Tiempo medio proceso	Dias desde CV hasta contrato	PostgreSQL
Coste por contratacion	Recursos usados / contratados	PostgreSQL

18.11 Pantallas del Frontend RRHH por Fase

Detalle de las vistas y pantallas especificas del frontend Vue.js RRHH:

Fase	Ruta	Pantalla
Fase 1	-	Sin frontend (automatico)
Fase 2	-	Sin frontend (automatico)
Fase 3	/rrhh/candidatos	Lista candidatos preseleccionados
Fase 3	/rrhh/candidatos/:id	Ficha completa del candidato
Fase 3	/rrhh/comparar	Comparador de candidatos
Fase 4	/rrhh/llamadas	Gestor de llamadas pendientes
Fase 4	/rrhh/llamadas/:id	Registro de llamada
Fase 5	/rrhh/entrevistas	Calendario de entrevistas
Fase 5	/rrhh/entrevistas/:id	Evaluacion post-entrevista
Fase 6	/rrhh/pool	Pool de candidatos en reserva

18.12 Detalle de Cada Pantalla

PANTALLA: /rrhh/candidatos (Lista de Candidatos)

Elemento	Descripcion	Detalle
Tabla	Lista de candidatos con filtros	Nombre, email, estado, fecha
Filtros	Por estado, fecha, skills	Dropdown + busqueda
Acciones	Ver ficha, aprobar, descartar	Botones por fila
Widget	Chat IA flotante	Preguntas sobre candidatos
Widget	Campanita alertas	Nuevos CVs, tareas pendientes

PANTALLA: /rrhh/candidatos/:id (Ficha del Candidato)

Elemento	Descripcion	Detalle
Cabecera	Foto + datos basicos	Nombre, email, telefono, ubicacion
Seccion CV	Datos extraidos por IA	Experiencia, formacion, skills
Seccion Doc	PDF original del CV	Visor embebido
Seccion Notas	Notas internas RRHH	Editor de texto
Seccion Hist	Historial de acciones	Timeline de estados
Botones	Aprobar / Descartar / Llamar	Acciones principales
Chat IA	Preguntas sobre este CV	'Resume este CV', 'Skills clave?'

PANTALLA: /rrhh/llamadas (Gestor de Llamadas)

Elemento	Descripcion	Detalle
Alerta	Progreso X/Y	Barra: 5/20 llamadas completadas
Lista	Candidatos a llamar	Ordenados por prioridad
Por fila	Nombre + telefono + boton	Click para registrar llamada
Modal	Registrar resultado	Contesta/No contesta/Rechaza
Modal	Agendar cita	Selector de fecha y hora
Resumen	Estadisticas del dia	Llamadas hechas, citas, rechazos

PANTALLA: /rrhh/entrevistas (Calendario)

Elemento	Descripcion	Detalle
Vista	Calendario mensual/semanal	Citas programadas
Evento	Click en cita	Abre ficha del candidato
Color	Por estado	Pendiente/Realizada/Cancelada
Accion	Arrastrar para mover	Reagendar entrevista
Panel	Entrevistas de hoy	Lista lateral con accesos rapidos

PANTALLA: /rrhh/entrevistas/:id (Evaluacion)

Elemento	Descripcion	Detalle
Cabecera	Datos del candidato	Nombre + puesto solicitado
Formulario	Puntuacion competencias	1-5 estrellas por area
Areas	Tecnica, comunicacion, actitud	Sliders o estrellas
Textarea	Observaciones	Notas de la entrevista
Decision	Contratar/2a entrev/Pool/Descartar	Radio buttons
Boton	Guardar evaluacion	Guarda y notifica

PANTALLA: /rrhh/pool (Candidatos en Reserva)

Elemento	Descripcion	Detalle
Tabla	Candidatos en espera	Nombre, fecha, motivo, skills
Filtros	Por skills, fecha, puntuacion	Busqueda avanzada
Accion	Reactivar candidato	Vuelve al proceso activo
Alerta IA	Sugerencias automaticas	'3 candidatos coinciden con vacante X'
Busqueda	Semantica por skills	'Buscar expertos en logistica'

18.13 Widgets Comunes en Todas las Pantallas RRHH

Todos los frontends RRHH incluyen estos widgets:

Widget	Posicion	Funcion
Campanita	Esquina superior derecha	Alertas y notificaciones
Chat IA	Flotante inferior derecha	Asistente conversacional
Breadcrumb	Cabecera	Navegacion: RRHH > Candidatos > Juan
Usuario	Esquina superior derecha	Nombre + rol + logout
Menu lateral	Izquierda	Navegacion entre pantallas

Estructura del menu lateral RRHH:

Menu	Ruta	Descripcion
Dashboard	/rrhh	Resumen y KPIs
Candidatos	/rrhh/candidatos	Gestion de CVs
Llamadas	/rrhh/llamadas	Gestor de contactos
Entrevistas	/rrhh/entrevistas	Calendario y evaluaciones
Pool	/rrhh/pool	Candidatos en reserva
Vacantes	/rrhh/vacantes	Puestos abiertos
Reportes	/rrhh/reportes	Estadisticas y KPIs
Configuracion	/rrhh/config	Criterios de criba, plantillas

18.14 Actividades por Trabajador en el Flujo RRHH

Detalle de cada actividad, quien la realiza y que componentes utiliza:

FASE 1: ENTRADA DE CVs

Quien	Actividad	Frontend	Backend	BD	Agente
Candidato	Envia CV por email/web	-	Laravel	-	-
Sistema	Verifica si ya es empleado	-	FastAPI	MySQL	Auto
Sistema	Extrae texto del PDF	-	FastAPI	-	Docs
Sistema	Parsea datos con LLM	-	FastAPI+Ollama	-	Docs
Sistema	Guarda candidato	-	FastAPI	PG	Docs
Sistema	Guarda embeddings CV	-	FastAPI	Chroma	Docs

FASE 2: PRIMERA CRIBA (AUTOMATICA)

Quien	Actividad	Frontend	Backend	BD	Agente
Sistema	Ejecuta job de criba	-	Celery	-	Alertas
Sistema	Consulta historial empleado	-	FastAPI	MySQL	Asist
Sistema	Evalua criterios minimos	-	FastAPI	PG	Asist
Sistema	Clasifica candidato	-	FastAPI	PG	Asist
Sistema	Notifica a RRHH	-	Laravel Echo	Redis	Alertas

FASE 3: SEGUNDA CRIBA (MANUAL)

Quien	Actividad	Frontend	Backend	BD	Agente
RRHH	Abre lista candidatos	Vue /candidatos	Laravel	PG	-
RRHH	Ve ficha candidato	Vue /candidatos/:id	Laravel	PG	-
RRHH	Ve si fue empleado antes	Vue /candidatos/:id	Laravel	MySQL	-
RRHH	Pregunta al chat IA	Vue (widget)	FastAPI	PG+Chroma	Asist
RRHH	Compara candidatos	Vue /comparar	FastAPI	PG+Chroma	Asist
RRHH	Aprueba o descarta	Vue /candidatos/:id	Laravel	PG	-
RRHH	Escribe notas	Vue /candidatos/:id	Laravel	PG	-

FASE 4: LLAMADAS A CANDIDATOS

Quien	Actividad	Frontend	Backend	BD	Agente
Gerencia	Selecciona candidatos	Vue /candidatos	Laravel	PG	-
Gerencia	Asigna tarea a RRHH	Vue /candidatos	Laravel	PG	Alertas
Sistema	Crea alerta 0/X	-	Laravel	PG+Redis	Alertas
RRHH	Ve alerta en panel	Vue (campanita)	Echo	Redis	Alertas
RRHH	Abre gestor llamadas	Vue /llamadas	Laravel	PG	-
RRHH	Llama al candidato	Vue /llamadas/:id	-	-	-
RRHH	Registra resultado	Vue /llamadas/:id	Laravel	PG	-
RRHH	Agenda cita	Vue /llamadas/:id	Laravel	PG	-
Sistema	Actualiza X/Y	-	Laravel	PG	Alertas
Sistema	Cierra alerta	-	Laravel	PG	Alertas

FASE 5: ENTREVISTAS Y CONTRATACION

Quien	Actividad	Frontend	Backend	BD	Agente
RRHH	Consulta calendario	Vue /entrevistas	Laravel	PG	-
RRHH	Ve citas del dia	Vue /entrevistas	Laravel	PG	-
RRHH	Realiza entrevista	-	-	-	-
RRHH	Evalua competencias	Vue /entrev/:id	Laravel	PG	-
RRHH	Escribe observaciones	Vue /entrev/:id	Laravel	PG	-
RRHH	Consulta salario puesto	Vue /entrev/:id	Laravel	MySQL	-
RRHH	Decide contratar	Vue /entrev/:id	Laravel	PG	-
Sistema	Crea empleado en ERP	-	Laravel	MySQL	Auto
Sistema	Crea registro nomina	-	Laravel	MySQL	Auto
Sistema	Notifica a gerencia	-	Echo	Redis	Alertas

FASE 6: REACTIVACION DE CANDIDATOS

Quien	Actividad	Frontend	Backend	BD	Agente
Sistema	Detecta nueva vacante	-	Celery	MySQL	Alertas
Sistema	Busca en pool por skills	-	FastAPI	Chroma	Docs
Sistema	Filtra por historial	-	FastAPI	MySQL	Asist
Sistema	Crea alerta sugerencias	-	Laravel	PG	Alertas
RRHH	Ve alerta reactivacion	Vue (campanita)	Echo	Redis	Alertas
RRHH	Abre pool candidatos	Vue /pool	Laravel	PG	-
RRHH	Revisa sugeridos	Vue /pool	FastAPI	PG+Chroma	Asist
RRHH	Reactiva candidato	Vue /pool	Laravel	PG	-
Sistema	Mueve a Fase 4	-	Laravel	PG	-

18.15 Resumen de Actividades por Rol

Cuantas actividades realiza cada trabajador en el proceso completo:

Rol	Actividades	Tipo	Principales Tareas
Sistema/Agentes	23	Automaticas	Deteccion, criba, alertas, sync
RRHH	22	Manuales	Revision, llamadas, entrevistas, pool
Gerencia	2	Supervision	Asignar tareas, recibir notificaciones
Candidato	1	Externa	Enviar CV

Distribucion de trabajo humano vs automatico:

Fase	Humano	Automatico	Descripcion
Fase 1	0%	100%	Totalmente automatizada
Fase 2	0%	100%	Totalmente automatizada
Fase 3	100%	0%	RRHH revisa con ayuda IA
Fase 4	80%	20%	RRHH llama, sistema trackea
Fase 5	100%	0%	RRHH entrevista y evalua
Fase 6	30%	70%	Sistema sugiere, RRHH decide

Este documento sirve como guia de referencia para el diseno e implementacion del sistema multi-agente. Se recomienda adaptar la arquitectura segun las necesidades especificas del negocio y los recursos disponibles.