Descripción

Esta PR entrega la funcionalidad completa de **Asignación de Médico a Consultorio/Agenda**, consolidando una nueva funcionalidad que aplica múltiples reglas de negocio y garantiza la integridad de los recursos compartidos (Consultorios).

Resumen del Cambio

Se ha implementado el servicio POST /asignaciones para crear registros de disponibilidad de un médico en un consultorio, siguiendo un enfoque de **Arquitectura Hexagonal (Puertos y Adaptadores)**.

Funcionalidades Clave:

1. **Validación Robusta de Entrada (Zod):** Esquema asignacionesEsquema.ts que valida formato de hora (HH:MM) y la coherencia horaria (inicio < fin).
2. **Verificación de Existencia:** El Caso de Uso verifica que el Médico y el Consultorio referenciados existan antes de la persistencia.
3. **Regla de Negocio Crítica (Solapamiento):** Se aplica la verificación consultorioOcupado para asegurar que la nueva asignación no se solape con ninguna otra reserva existente en el mismo consultorio y fecha, **sin importar el médico asignado**, previniendo la doble reserva del espacio.
4. **Regla de Duplicidad:** Se verifica que no exista una asignación idéntica (mismo médico, mismo consultorio, misma franja) para el mismo profesional.
5. **Manejo de Errores:** Se redefine el error de negocio (HTTP 409) a "Fallo en las Reglas de Negocio" para mayor claridad.

Motivación y Contexto

El objetivo principal fue añadir una funcionalidad que maneja la **disponibilidad y gestión de recursos** (Consultorios) dentro del ecosistema **iuKer**.

La implementación mantiene el principio de **Aislamiento de Lógica** establecido en el Sprint 1:

- El Dominio define el contrato del repositorio (IRepositorioAsignacion) con funciones como existeAsignacion() y consultorioOcupado().
- La Aplicación (AsignacionCasosUso.ts) se encarga de las verificaciones de existencia, solapamiento y duplicidad, independientemente a la base de datos que estemos usando.
- La Infraestructura (AsignacionRepositorio.ts) implementa la lógica SQL y la corrección para la auto-generación de ID por parte de PostgreSQL (eliminando idAsignacion antes del INSERT).

Tipo de cambio

- Nueva funcionalidad (Implementación del servicio de Asignación Médico-Consultorio).

¿Cómo se ha probado esto?

La funcionalidad fue probada y documentada mediante:

1. **Video Demostrativo:** Interacción con el *endpoint* POST /asignaciones desde una herramienta HTTP (Bruno).
2. **Prueba de Cobertura de Errores (400 y 409):**
    - **400 (Formato):** Prueba de envío de horas en formato incorrecto o con hora de inicio posterior a hora de fin.
    - **409 (Negocio):** Pruebas de Médico Inexistente, Consultorio Inexistente, Asignación Duplicada y Conflicto de Solapamiento Horario.

Lista de verificación

- $$x$$
  Mi código sigue las guías de estilo de este proyecto.
- $$x$$
  He realizado una auto-evaluación de mi propio código.
- $$x$$
  He comentado mi código, especialmente en las partes difíciles de entender.
- $$x$$
  He realizado los cambios correspondientes en la documentación (Informe de Implementación y detalles de las validaciones adjuntos).
- $$x$$
  Mis cambios no generan nuevas advertencias.
- $$ $$
  He añadido pruebas que demuestran que mi corrección o funcionalidad funciona correctamente. (Pendiente de añadir pruebas unitarias).
- $$x$$
  Cualquier cambio dependiente ha sido fusionado y publicado en los módulos relacionados (Se reutilizaron los esquemas de Médico y Consultorio del Sprint 1, usando z.shape).