**IMPLEMENTACIÓN DE ARQUITECTURA DDD PARA EL PROYECTO DE MICROSERVICIOS DE GESTIÓN DE CITAS MÉDICAS**

### **Introducción**

El proyecto consta de tres microservicios independientes pero interconectados para la gestión de citas médicas y recetas asociadas. Cada microservicio manejará su propia base de datos de manera aislada, asegurando una arquitectura desacoplada y escalable.

### **Arquitectura DDD**

Cada microservicio seguirá el enfoque de **Domain-Driven Design (DDD)** con las siguientes capas:

1. **Capa de Dominio**: Define las entidades, agregados y lógica de negocio.
2. **Capa de Aplicación**: Contiene los casos de uso y servicios de aplicación.
3. **Capa de Infraestructura**: Implementa repositorios, integración con bases de datos y mensajería.
4. **Capa de Presentación**: Expone la API a través de controladores REST.

### **Diseño de Bases de Datos por Microservicio**

### **1. Microservicio de Personas**

* **Entidad: Persona**
  + Id (PK, GUID)
  + Nombre (VARCHAR)
  + Apellido (VARCHAR)
  + IdTipoPersona (FK)
  + Email (VARCHAR)
  + Teléfono (VARCHAR)
  + FechaNacimiento (DATE)
  + Especialidad (VARCHAR)
* **Entidad: TipoPersona**
  + Id (PK, GUID)
  + Descriptor (VARCHAR)

### **2. Microservicio de Citas**

* **Entidad: Cita**
  + Id (PK, GUID)
  + PacienteId (GUID, externo)
  + MedicoId (GUID, externo)
  + FechaHora (DATETIME)
  + Lugar (VARCHAR)
  + Estado (ENUM: Pendiente/En proceso/Finalizada)

### **3. Microservicio de Recetas**

* **Entidad: Receta**
  + Id (PK, GUID)
  + CitaId (GUID, externo)
  + PacienteId (GUID, externo)
  + Paciente (VARCHAR)
  + IdMedico (GUID externo)
  + Medico (VARCHAR)
  + Fecha\_Emision (datatime)
  + Descriptor
* **Entidad:Medicamento**
  + Id (PK, GUID)
  + IdReceta (FK)
  + NombreMedicamento (VARCHAR)
  + Dosis (VARCHAR)
  + Frecuencia(VARCHAR)

**Comunicación entre Microservicios**

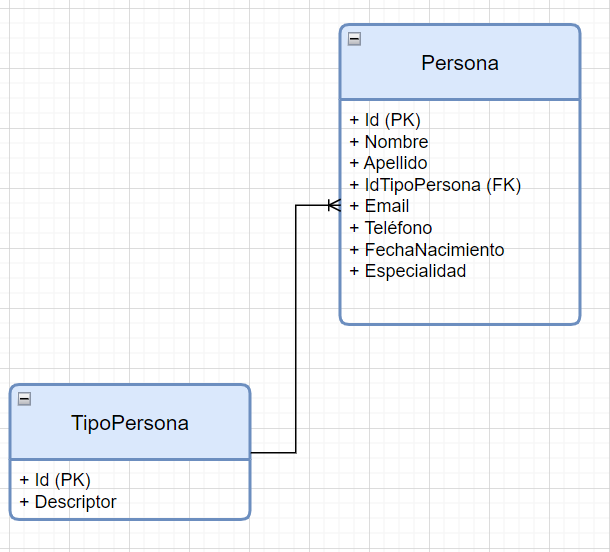
**Comunicación Síncrona (API REST)**

* El microservicio de **Citas** consulta a **Personas** para obtener información de pacientes y médicos.
* El microservicio de **Recetas** consulta a **Citas** para verificar información de la cita asociada.

**Comunicación Asíncrona (RabbitMQ)**

**Citas → Recetas**: Cuando una cita cambia a "Finalizada", se envía un mensaje a RabbitMQ para que **Recetas** cree una nueva receta médica.

**DIAGRAMA ENTIDAD RELACIÓN DE CADA BASE DE DATOS, RESPECTIVAMENTE:**

****

****

**Estructura general del proyecto en DDD:**

**📂 GestionCitasMedicas**

**├── 📂 Personas**

**│ ├── 📂 Application**

**│ │ ├── DTOs**

**│ │ ├── Services**

**│ │ └── Commands & Queries**

**│ ├── 📂 Domain**

**│ │ ├── Entities**

**│ │ └── Interfaces**

**│ ├── 📂 Infrastructure**

**│ │ ├── Persistence (EF Core / Dapper)**

**│ │ ├── Repositories**

**│ │ ├── Configurations**

**│ │ └── Messaging (RabbitMQ)**

**│ └── 📂 Mgration**

**│ ├── 📂 Presentation**

**│ │ ├── Controllers**

**│ └── 📂 Tests**

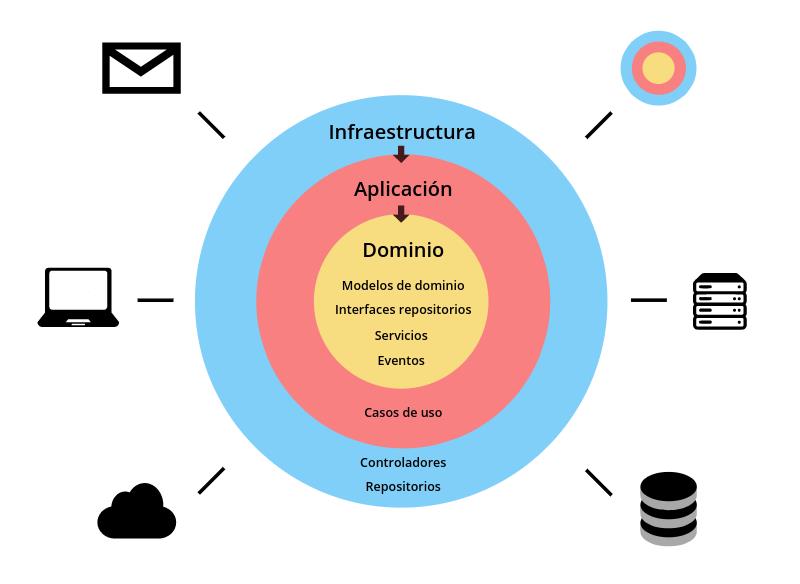
**├── 📂 Citas**

**├── 📂 Recetas**

**├── 📂 SharedKernel (Componentes compartidos)**

**├── 📂 API Gateway (Opcional)**

**└── 📂 Docs (Modelos y especificaciones)**

****