Documentación Completa – Sistema de Gestión de Citas para Spa

2025-10-24 23:59

Este documento incluye Roadmap, Casos de uso, Diagramas UML, ERD final y Script SQL (DDL). Los márgenes y anchos fueron ajustados para evitar desbordes.

Roadmap de Desarrollo

F1 – MVP: Usuarios/Admin, Servicios, Citas, Pagos básicos, Inventario light.
F2 – Operativa: Descuento de inventario al cerrar, Puntos (ganado/usado), Días inactivos, Dashboard.
F3 – Finanzas/Promos: Anticipos, Promociones (vigencia/estado), Comisiones y tickets de pago.
F4 – Admin/Auditoría: RBAC, Historial de actividades, Reportes, Notificaciones/recordatorios.
F5 – Escalabilidad: Multi-sucursal, portal de clientes, API pública, exportables, CI/CD, monitoreo.

Casos de Uso de Negocio

CU-01 Agendar Cita

Objetivo: Crear una cita válida.
Actores: Recepcionista, Admin
Precondiciones: Cliente/servicio existentes; empleado activo y disponible.
Flujo principal:
1) Seleccionar cliente y servicio.
2) Elegir fecha/hora y validar.
3) (Opcional) Promo/anticipo.
4) Guardar 'pendiente'.
5) Notificar a empleado.
Flujos alternos:
A1) Sin disponibilidad → proponer horarios.
Postcondiciones: Cita en 'pendiente'; bitácora creada; notificación enviada.

CU-02 Cerrar Cita

Objetivo: Completar servicio y registrar pago/consumos.
Actores: Recepcionista, Empleado, Admin
Precondiciones: Cita 'pendiente' o 'en curso'.
Flujo principal:
1) Registrar pago (efectivo/tarjeta/puntos).
2) Descontar inventario.
3) Registrar puntos 'ganado'.
4) Calcular comisión.
5) Cambiar a 'cerrada'.
Flujos alternos:
A1) Pago insuficiente → saldo.
A2) No asistencia → cancelar.
Postcondiciones: Cita cerrada; puntos otorgados; comisiones listas.

CU-03 Usar Puntos

Objetivo: Aplicar puntos como descuento.
Actores: Recepcionista, Cliente
Precondiciones: Cliente con saldo suficiente.
Flujo principal:
1) Ingresar puntos a usar.
2) Validar saldo.
3) Registrar 'usado'.
4) Recalcular total.
Flujos alternos:
A1) Saldo insuficiente → bloquear.
Postcondiciones: Pago con puntos aplicado; historial actualizado.

CU-04 Notificaciones

Objetivo: Recordatorios a clientes y avisos a empleados.
Actores: Sistema (cron), Recepcionista, Empleado, Cliente
Precondiciones: Citas futuras registradas.
Flujo principal:
1) Programar recordatorios X horas antes.
2) Enviar a cliente y empleado.
3) Panel recepcionista con próximas citas.
Flujos alternos:
A1) Falla de proveedor → reintentos/fallback.

Postcondiciones: Notificaciones enviadas; estados actualizados.

Diagramas UML – Casos de Uso

Diagrama 1: Citas y Pagos (Admin/Recepcionista)

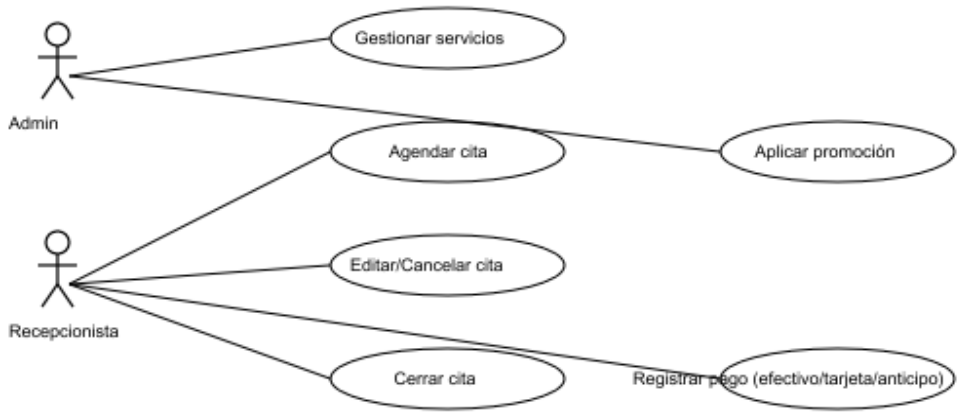
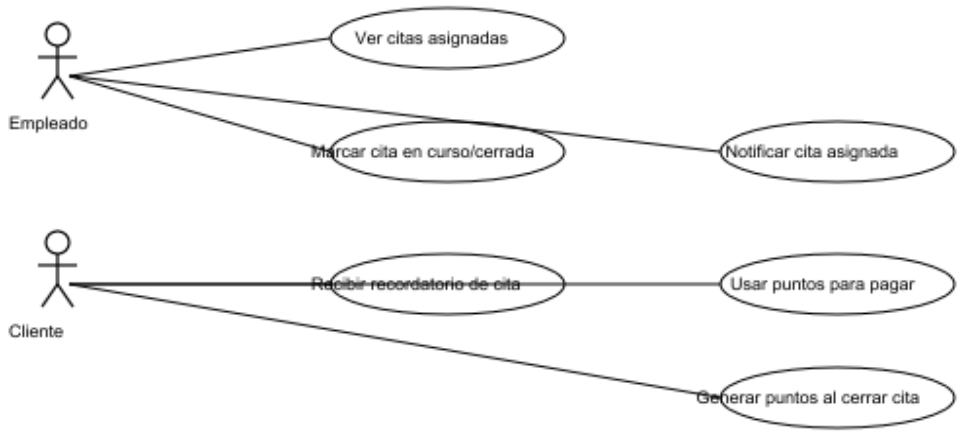
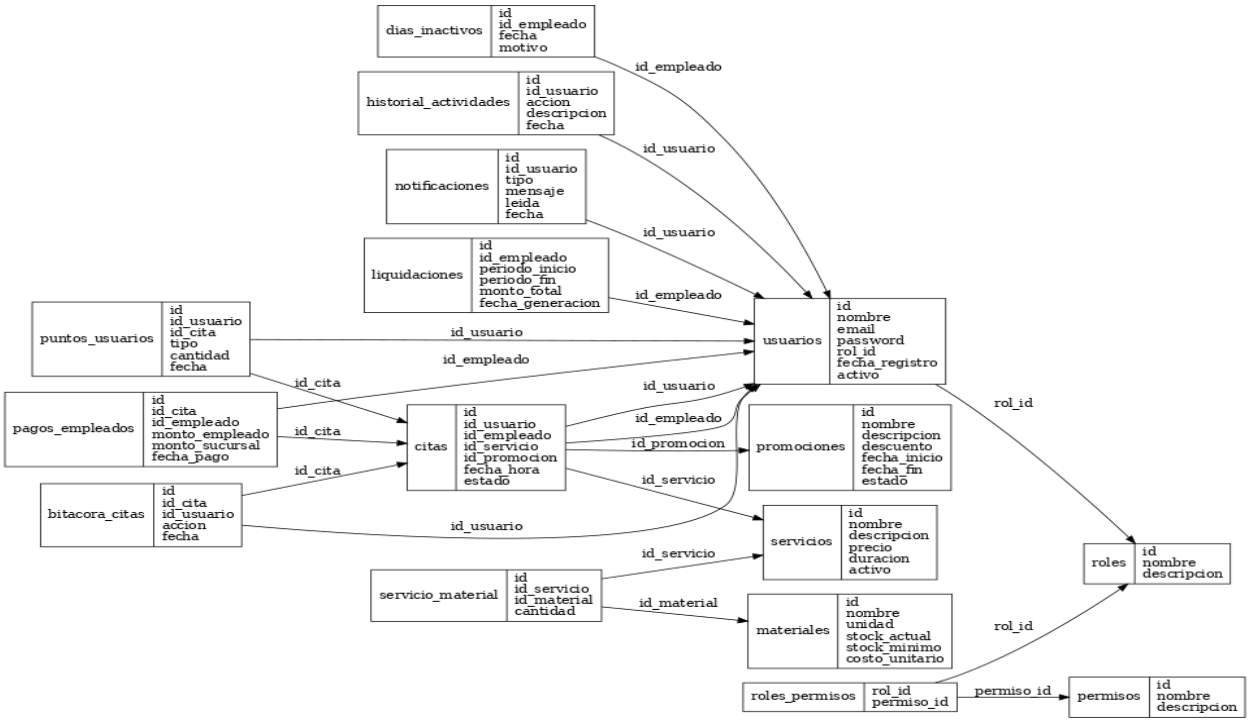


Diagrama 2: Notificaciones y Puntos (Empleado/Cliente)



ERD – Blueprint Final Corregido



Definición de Tablas

Tabla	Descripción
usuarios	Registra a clientes, empleados, recepcionistas y administradores del sistema.
roles	Define roles (admin, recepcionista, empleado, cliente).
permisos	Acciones que se pueden otorgar a un rol.
roles_permisos	Relación N:M entre roles y permisos.
servicios	Catálogo de servicios con costo y duración.
materiales	Inventario de materiales/insumos con stock y umbrales.
servicio_material	Asociación de materiales a servicios con cantidades.
promociones	Promociones con vigencia/estado aplicables a citas.
citas	Agenda de citas (cliente/empleado/servicio/promoción).
dias_inactivos	Días libres/permiso de empleados.
puntos_usuarios	Movimientos de puntos ligados a una cita.
historial_actividades	Auditoría de acciones por usuario.
bitacora_citas	Historial específico de una cita.
notificaciones	Mensajes a clientes/empleados.
pagos_empleados	Comisiones por cita para empleados.
liquidaciones	Resumen de pagos por periodo a empleados.

Script SQL (DDL en PostgreSQL) – Completo

```
-- ========================================
-- Script DDL PostgreSQL - Sistema de Citas Spa (Completo y Corregido)
-- ========================================

-- Roles
CREATE TABLE IF NOT EXISTS roles (
    id SERIAL PRIMARY KEY,
    nombre VARCHAR(50) UNIQUE NOT NULL,
    descripcion TEXT
);

-- Permisos
CREATE TABLE IF NOT EXISTS permisos (
    id SERIAL PRIMARY KEY,
    nombre VARCHAR(80) UNIQUE NOT NULL,
    descripcion TEXT
);

-- Relación Roles - Permisos (N:M)
CREATE TABLE IF NOT EXISTS roles_permisos (
    rol_id INT REFERENCES roles(id) ON DELETE CASCADE,
    permiso_id INT REFERENCES permisos(id) ON DELETE CASCADE,
    PRIMARY KEY (rol_id, permiso_id)
);

-- Usuarios (Admin, Recepcionista, Empleados, Clientes)
CREATE TABLE IF NOT EXISTS usuarios (
    id SERIAL PRIMARY KEY,
    nombre VARCHAR(100) NOT NULL,
    email VARCHAR(100) UNIQUE NOT NULL,
    password TEXT NOT NULL,
    rol_id INT REFERENCES roles(id),
    fecha_registro TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    activo BOOLEAN DEFAULT TRUE
);

CREATE INDEX IF NOT EXISTS idx_usuarios_email ON usuarios(email);

-- Servicios
CREATE TABLE IF NOT EXISTS servicios (
    id SERIAL PRIMARY KEY,
    nombre VARCHAR(100) NOT NULL,
    descripcion TEXT,
    precio DECIMAL(10,2) NOT NULL,
    duracion INTERVAL NOT NULL,
    activo BOOLEAN DEFAULT TRUE
);

-- Materiales (inventario)
CREATE TABLE IF NOT EXISTS materiales (
    id SERIAL PRIMARY KEY,
    nombre VARCHAR(100) NOT NULL,
    unidad VARCHAR(20) NOT NULL,
    stock_actual NUMERIC(12,2) DEFAULT 0,
    stock_minimo NUMERIC(12,2) DEFAULT 0,
    costo_unitario NUMERIC(12,2) DEFAULT 0
);

-- Servicio-Material (N:M)
CREATE TABLE IF NOT EXISTS servicio_material (
    id SERIAL PRIMARY KEY,
    id_servicio INT REFERENCES servicios(id) ON DELETE CASCADE,
    id_material INT REFERENCES materiales(id) ON DELETE RESTRICT,
    cantidad NUMERIC(12,2) NOT NULL CHECK (cantidad >= 0)
);

-- Promociones
CREATE TABLE IF NOT EXISTS promociones (
    id SERIAL PRIMARY KEY,
    nombre VARCHAR(100) NOT NULL,
    descripcion TEXT,
    descuento DECIMAL(5,2) NOT NULL CHECK (descuento >= 0 AND descuento <= 100),
    fecha_inicio DATE NOT NULL,
    fecha_fin DATE NOT NULL,
    estado BOOLEAN DEFAULT TRUE
);
```

```
-- Citas
CREATE TABLE IF NOT EXISTS citas (
    id SERIAL PRIMARY KEY,
    id_usuario INT REFERENCES usuarios(id),      -- cliente
    id_empleado INT REFERENCES usuarios(id),     -- empleado que atiende
    id_servicio INT REFERENCES servicios(id),
    id_promocion INT REFERENCES promociones(id),
    fecha_hora TIMESTAMP NOT NULL,
    estado VARCHAR(20) CHECK (estado IN ('pendiente','cancelada','cerrada')) NOT NULL,
    CONSTRAINT ck_fecha_futura CHECK (fecha_hora > '2000-01-01')
);

CREATE INDEX IF NOT EXISTS idx_citas_fecha ON citas(fecha_hora);
CREATE INDEX IF NOT EXISTS idx_citas_estado ON citas(estado);

-- Días inactivos del empleado
CREATE TABLE IF NOT EXISTS dias_inactivos (
    id SERIAL PRIMARY KEY,
    id_empleado INT REFERENCES usuarios(id) ON DELETE CASCADE,
    fecha DATE NOT NULL,
    motivo TEXT
);

-- Puntos de usuarios (siempre ligados a una cita)
CREATE TABLE IF NOT EXISTS puntos_usuarios (
    id SERIAL PRIMARY KEY,
    id_usuario INT REFERENCES usuarios(id) ON DELETE CASCADE,
    id_cita INT REFERENCES citas(id) ON DELETE CASCADE,
    tipo VARCHAR(10) CHECK (tipo IN ('ganado','usado')) NOT NULL,
    cantidad INT NOT NULL CHECK (cantidad > 0),
    fecha TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);

CREATE INDEX IF NOT EXISTS idx_puntos_usuario ON puntos_usuarios(id_usuario);

-- Historial de actividades (auditoría)
CREATE TABLE IF NOT EXISTS historial_actividades (
    id SERIAL PRIMARY KEY,
    id_usuario INT REFERENCES usuarios(id) ON DELETE SET NULL,
    accion VARCHAR(100) NOT NULL,
    descripcion TEXT,
    fecha TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);

-- Bitácora de citas
CREATE TABLE IF NOT EXISTS bitacora_citas (
    id SERIAL PRIMARY KEY,
    id_cita INT REFERENCES citas(id) ON DELETE CASCADE,
    id_usuario INT REFERENCES usuarios(id) ON DELETE SET NULL,
    accion VARCHAR(100) NOT NULL,
    fecha TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);

-- Notificaciones
CREATE TABLE IF NOT EXISTS notificaciones (
    id SERIAL PRIMARY KEY,
    id_usuario INT REFERENCES usuarios(id) ON DELETE CASCADE,
    tipo VARCHAR(50) NOT NULL,
    mensaje TEXT NOT NULL,
    leida BOOLEAN DEFAULT FALSE,
    fecha TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);

-- Pagos empleados (comisiones por cita)
CREATE TABLE IF NOT EXISTS pagos_empleados (
    id SERIAL PRIMARY KEY,
    id_cita INT REFERENCES citas(id) ON DELETE CASCADE,
    id_empleado INT REFERENCES usuarios(id) ON DELETE CASCADE,
    monto_empleado NUMERIC(12,2) NOT NULL DEFAULT 0,
    monto_sucursal NUMERIC(12,2) NOT NULL DEFAULT 0,
    fecha_pago TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);

-- Liquidaciones (resumen por periodo)
CREATE TABLE IF NOT EXISTS liquidaciones (
    id SERIAL PRIMARY KEY,
    id_empleado INT REFERENCES usuarios(id) ON DELETE CASCADE,
    periodo_inicio DATE NOT NULL,
    periodo_fin DATE NOT NULL,
```

```
    monto_total NUMERIC(12,2) NOT NULL DEFAULT 0,
    fecha_generacion TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);

-- Índices útiles
CREATE INDEX IF NOT EXISTS idx_notif_usuario ON notificaciones(id_usuario, leida);
CREATE INDEX IF NOT EXISTS idx_bitacora_cita ON bitacora_citas(id_cita);
CREATE INDEX IF NOT EXISTS idx_pagos_empleado ON pagos_empleados(id_empleado);
```