



**Guillermo Algárate del Arco**  
**Gabriel Sánchez Vallejo Millán**

**Proyecto intermodular**

**2º DAW**

2025 - 2026

**Desarrollo de Aplicaciones Web**

**CENTRO SAN VALERO**  
**ZARAGOZA**

## Resumen

**ReservArte** es una aplicación web que permite la gestión completa de citas, clientes y pagos en negocios locales que funcionan a través de reservas tales como centros de estética, clínicas y otros negocios. Su objetivo es optimizar y unificar en un mismo entorno los procesos de administración de citas, cobros, mejorar la comunicación con los clientes y mejorar el tiempo de gestión de empleados y servicios.

La solución incluye un panel de administración (backoffice) donde los propietarios pueden gestionar empleados, servicios, clientes, pagos y recordatorios, así como una parte pública desde la que los clientes pueden reservar citas y gestionar su información personal.

ReservArte se ha desarrollado utilizando tecnologías modernas como ASP.NET Core 8.0 para el backend, Vue.js con Nuxt.js para el frontend, PostgreSQL como base de datos y servicios en AWS para la infraestructura. *Además, se integra con Redsys u otra plataforma para pagos y notificaciones para recordatorios automatizados.*

## Abstract

## 2. Objetivo del Proyecto

La clienta manifiesta su necesidad de mejorar el sistema de gestión de citas de su negocio. Al no existir en el mercado una aplicación que permita satisfacer las necesidades particulares de su negocio, la clienta, aprovechando su relación marital con uno de los desarrolladores, plantea la posibilidad de crear un ecosistema que le permita implementar una solución holística de su negocio y que se adapte al 100% a su casuística, desarrollando un proyecto multiplataforma a través del cual gestionar el core de su negocio (estudio de diseño de cejas).

Su marido, que no es tonto del todo y que es incapaz de decirle que no a su mujer, ha convencido a su compañero de clase para implementar esta solución bajo la premisa de poder implementar una solución configurable que sea capaz de adaptarse tanto a las necesidades de su mujer, como a otros potenciales clientes con necesidades más genéricas, distribuyendo esta solución bajo la fórmula SaaS.

A través de esta solución que se desarrollará en varias fases, alguna de las cuales quedará fuera del alcance de este proyecto, se pretende dar solución a los distintos problemas de gestión de este y otros negocios.

## 3. Solución Propuesta

Se propone la creación de un ecosistema digital compuesto por los siguientes elementos:

- Aplicación web compuesta de:
  - Infraestructura cloud

Albergará tanto la capa de datos conteniendo la base de datos, como el servidor web y donde se ejecutarán las aplicaciones de servidor.

- Front-End

Será una capa de presentación y forma de acceso del usuario, así como para la gestión del backoffice.

- Back-End/API

Constituirá la capa lógica de negocio y nutrirá de contenido a las capas de presentación.

- Aplicaciones web nativas (fuera del alcance de este proyecto)

A petición de la clienta, se implementarán como otra capa de presentación en una fase posterior.

Esta solución nace para cumplir los siguientes objetivos:

- **Objetivos de Negocio**

1. Digitalizar completamente la gestión del centro de diseño de cejas
2. Reducir ausencias y cancelaciones mediante sistema de pre-autorización de pagos
3. Mejorar la comunicación con clientes mediante recordatorios automatizados
4. Aumentar la eficiencia operativa con gestión automatizada de citas y empleados
5. Crear una base para monetización futura mediante modelo SaaS multi-tenant.

- **Objetivos Técnicos**

1. Alta disponibilidad: 99.9% uptime objetivo
2. Escalabilidad: Soportar desde 1 hasta 100+ organizaciones
3. Rendimiento: Tiempos de respuesta < 200ms para operaciones críticas
4. Seguridad: Cumplimiento estricto RGPD y PCI-DSS mediante Redsys InSite
5. Mantenibilidad: Código limpio, documentado y testable.

- **Objetivos de Usuario**

- Cliente Final:
  - i. Reservar citas de forma intuitiva en < 2 minutos
  - ii. Recibir recordatorios oportunos y relevantes
  - iii. Gestionar sus citas y perfil fácilmente
  - iv. Guardar tarjetas de forma segura para pagos rápidos
- Personal del Centro:
  - i. Gestionar agenda de forma eficiente
  - ii. Visualizar información de clientes rápidamente
  - iii. Gestionar servicios y disponibilidad
- Administrador:
  - i. Control total de configuración

- ii. Acceso a estadísticas y reportes
- iii. Gestión de empleados y permisos

### 3.1 Funcionalidades principales

#### 3.1.1 Módulos del Sistema

##### 3.1.1.1 Gestión de Organizaciones (Multi-Tenant)

Prioridad: **MUST-HAVE**

Funcionalidades:

- Registro y alta de nuevas organizaciones
- Configuración por organización:
- Datos fiscales y comerciales
- Horarios de operación
- Políticas de cancelación personalizadas
- Branding (logo, colores, dominio personalizado)
- Panel de administración de organizaciones
- Aislamiento total de datos entre organizaciones
- Gestión de suscripciones y facturación (para modelo SaaS)

Casos de uso:

- Dueño de centro registra su negocio en la plataforma
- Administrador configura horarios y servicios del centro
- Sistema crea base de datos aislada o schema para la organización

##### 3.1.1.2 Gestión de Empleados

Prioridad: **MUST-HAVE**

Funcionalidades:

- CRUD completo de empleados
- Asignación de roles y permisos:
  - Administrador
  - Especialista/Técnico
  - Recepcionista

- Visualizador (solo lectura)
- Gestión de horarios y disponibilidad:
  - Horarios semanales recurrentes
  - Excepciones (vacaciones, bajas, eventos)
  - Bloques de tiempo no disponibles
- Especialización en servicios:
  - Asignar qué servicios puede realizar cada empleado
  - Niveles de experiencia (junior, senior, experto)
- Comisiones y objetivos de ventas
- Historial de servicios realizados
- Evaluaciones y comentarios de clientes

### **Entidades de base de datos:**

#### Employee

- Id (Guid)
- OrganizationId (Guid)
- FirstName (string)
- LastName (string)
- Email (string)
- Phone (string)
- RoleId (Guid)
- IsActive (bool)
- HireDate (DateTime)
- ProfileImageUrl (string)

#### EmployeeAvailability

- Id (Guid)
- EmployeeId (Guid)
- DayOfWeek (int)
- StartTime (TimeSpan)
- EndTime (TimeSpan)
- IsRecurring (bool)

#### EmployeeException

- Id (Guid)
- EmployeeId (Guid)

- StartDateTime (DateTime)
- EndDateTime (DateTime)
- Reason (string)
- Type (Vacation/Sick/Other)

#### EmployeeService

- EmployeeId (Guid)
- ServiceId (Guid)
- ProficiencyLevel (int)

### 3.1.1.3 Gestión de Clientes

Prioridad: **MUST-HAVE**

Funcionalidades:

- Registro y perfiles de clientes:
  - Datos personales (nombre, email, teléfono)
  - Preferencias de contacto
  - Historial médico/alergias relevantes
  - Fotografías antes/después (con consentimiento)
- Gestión de tarjetas de crédito guardadas:
  - Guardar múltiples tarjetas por cliente mediante tokenización de Redsys
  - Visualizar últimos 4 dígitos y marca de tarjeta
  - Marcar tarjeta por defecto
  - Eliminar tarjetas guardadas
  - Renovación automática de tarjetas caducadas
- Sistema de categorización de clientes:
  - VIP (clientes frecuentes)
  - Regular
  - Nuevo
  - Bloqueado (por no-shows reiterados)
- Control de acceso a reservas:
  - Lista blanca: solo clientes aprobados pueden reservar
  - Restricción por categoría
  - Requerir aprobación manual para nuevos clientes
- Historial completo:
  - Servicios recibidos
  - Empleados que le atendieron

- Pagos realizados
  - Cancelaciones y no-shows
- Programa de fidelización:
  - Puntos por servicios
  - Descuentos personalizados
  - Cupones y promociones
- Notas internas (solo visibles para el personal)
- Consentimientos y autorizaciones (RGPD)

### Entidades de base de datos:

#### Customer

- Id (Guid)
- OrganizationId (Guid)
- FirstName (string)
- LastName (string)
- Email (string)
- Phone (string)
- BirthDate (DateTime?)
- CategoryId (Guid)
- LoyaltyPoints (int)
- IsBlocked (bool)
- BlockedReason (string)
- PreferredContactMethod (Email/WhatsApp/SMS)
- MarketingConsent (bool)
- CreatedAt (DateTime)

#### CustomerNote

- Id (Guid)
- CustomerId (Guid)
- EmployeeId (Guid)
- Note (string)
- CreatedAt (DateTime)

#### CustomerAllergy

- Id (Guid)

- CustomerId (Guid)
- AllergyDescription (string)
- Severity (Low/Medium/High)

#### CustomerConsent

- Id (Guid)
- CustomerId (Guid)
- ConsentType (DataProcessing/Marketing/Photos/SavedCards)
- IsGranted (bool)
- GrantedAt (DateTime)
- RevokedAt (DateTime?)

#### CustomerPaymentMethod

- Id (Guid)
- CustomerId (Guid)
- RedsysToken (string) // Token de Redsys para la tarjeta
- RedsysCofTxnid (string) // ID de transacción original COF
- CardLast4 (string) // Últimos 4 dígitos
- CardBrand (string) // Visa, Mastercard, etc.
- CardExpiry (string) // AAMM
- IsDefault (bool)
- CreatedAt (DateTime)
- UpdatedAt (DateTime)

### 3.1.1.4 Catálogo de Servicios

Prioridad: **MUST-HAVE**

Funcionalidades:

- CRUD de servicios:
  - Diseño de cejas
  - Tinte de cejas
- Configuración por servicio:

- Nombre y descripción
- Duración estimada
- Precio base
- Precio para empleados junior/senior/experto
- Categoría del servicio
- Imagen representativa
- Productos utilizados
- Requisitos previos (ej: prueba de alergia 48h antes)
- Paquetes y combos:
  - Agrupar múltiples servicios con descuento
  - Servicios secuenciales
- Servicios con variaciones:
  - Tamaño (pequeño/mediano/grande)
  - Técnica (manual/con máquina)
- Disponibilidad temporal:
  - Servicios de temporada
  - Promociones con fecha de inicio/fin

#### Entidades de base de datos:

##### Service

- Id (Guid)
- OrganizationId (Guid)
- Name (string)
- Description (string)
- DurationMinutes (int)
- BasePrice (decimal)
- CategoryId (Guid)
- ImageUrl (string)
- IsActive (bool)
- RequiresAllergyTest (bool)
- AllergyTestHoursBefore (int)

##### ServiceVariation

- Id (Guid)
- ServiceId (Guid)
- Name (string)
- PriceModifier (decimal)

- DurationModifier (int)

#### ServicePackage

- Id (Guid)
- OrganizationId (Guid)
- Name (string)
- Description (string)
- TotalPrice (decimal)
- DiscountPercentage (decimal)

#### ServicePackageItem

- ServicePackageId (Guid)
- ServiceId (Guid)
- Order (int)

### 3.1.1.5 Sistema de Agenda y Citas

Prioridad: **MUST-HAVE (núcleo del sistema)**

Funcionalidades principales:

#### A. Visualización de Agenda

- Vista diaria, semanal y mensual
- Vista por empleado individual o todos
- Vista por sala/estación de trabajo
- Código de colores por:
  - Tipo de servicio
  - Estado de la cita (confirmada/pendiente/completada)
  - Cliente VIP
- Drag & drop para reorganizar citas
- Leyenda visual clara

#### B. Creación de Citas

- Dos modos según configuración:
  - Modo Público: Clientes pueden reservar directamente

- Modo Privado: Solo personal puede crear citas
- Wizard de reserva paso a paso:
  - Selección de servicio(s)
  - Selección de empleado (o automático según disponibilidad)
  - Selección de fecha y hora
  - Datos del cliente (o login si ya existe)
  - Selección de método de pago (tarjeta guardada o nueva)
  - Confirmación y pago
- Validaciones automáticas:
  - Disponibilidad del empleado
  - Tiempo suficiente para el servicio
  - No solapamiento de citas
  - Restricciones del cliente
  - Horarios de operación
- Sugerencias inteligentes:
  - Próximos slots disponibles
  - Empleados alternativos
  - Servicios complementarios

### C. Gestión de Citas

- Estados de cita:
  - Pendiente de confirmación
  - Confirmada
  - En progreso
  - Completada
  - Cancelada por cliente
  - Cancelada por negocio
  - No presentado (no-show)
- Acciones disponibles:
  - Confirmar/Rechazar
  - Reagendar (automático con notificación)
  - Cancelar (con o sin penalización)
  - Marcar como completada
  - Añadir notas internas
  - Registrar pago
- Notificaciones automáticas en cada cambio de estado

### D. Políticas de Cancelación

- Configuración por organización:
  - Tiempo mínimo de anticipación para cancelar/modificar sin penalización
  - Porcentaje de penalización (0-100%)
  - Número máximo de no-shows antes de bloqueo

- Sistema de pre-autorización con Redsys:
  - Pre-autorización del importe en tarjeta al reservar
  - Liberación si se cancela con tiempo suficiente
  - Captura del porcentaje penalizado si cancela tarde
  - Captura total en caso de no-show
- Excepciones:
  - Clientes VIP pueden tener políticas diferentes
  - Motivos justificados (emergencias, con evidencia)

## E. Lista de Espera

- Clientes pueden apuntarse a lista de espera para:
  - Fecha/hora específica si está ocupada
  - Cualquier hueco en un rango de fechas
  - Empleado específico
- Notificación automática cuando se libera un hueco
- Prioridad según:
  - Orden de registro en lista
  - Categoría del cliente (VIP primero)
  - Número de servicios contratados

## Entidades de base de datos:

### Appointment

- Id (Guid)
- OrganizationId (Guid)
- CustomerId (Guid)
- EmployeeId (Guid)
- AppointmentDate (DateTime)
- StartTime (TimeSpan)
- EndTime (TimeSpan)
- (Pending/Confirmed/InProgress/Completed/Cancelled/NoShow) Status
- TotalPrice (decimal)
- DepositAmount (decimal)
- RedsysOrderNumber (string) // Número de pedido Redsys
- RedsysPreAuthToken (string) // Token de pre-autorización
- PaymentMethodId (Guid?) // Tarjeta usada si es guardada
- CancellationReason (string)
- CancelledAt (DateTime?)

- CancelledBy (CustomerId/EmployeeId)
- Notes (string)
- CreatedAt (DateTime)

#### AppointmentService

- Id (Guid)
- AppointmentId (Guid)
- ServiceId (Guid)
- ServiceVariationId (Guid?)
- Price (decimal)
- DurationMinutes (int)
- Order (int)

#### WaitingList

- Id (Guid)
- OrganizationId (Guid)
- CustomerId (Guid)
- ServiceId (Guid)
- PreferredEmployeeId (Guid?)
- PreferredDate (DateTime?)
- DateRangeStart (DateTime)
- DateRangeEnd (DateTime)
- Priority (int)
- CreatedAt (DateTime)
- NotifiedAt (DateTime?)

### **3.1.1.6 Sistema de Pagos**

Prioridad: **MUST-HAVE (crítico)**

Funcionalidades:

A. Métodos de Pago Aceptados

- Tarjeta de crédito/débito (a través de Redsys)
- Bizum (integración con Redsys)
- Efectivo (registro manual por el personal)
- Transferencia bancaria
- TPV físico (registro en sistema)

## B. Pre-autorización de Pagos con Redsys

- Al crear la cita online:
  - Cliente introduce datos de tarjeta (o selecciona tarjeta guardada)
  - Sistema hace pre-autorización (tipo transacción "1") del 100% o monto configurado
  - El dinero se bloquea pero NO se cobra
  - Pre-autorización válida por 7 días
- Escenarios:
  - Asiste a la cita: Se confirma (tipo "2") el pago completo al finalizar servicio
  - Asiste a la cita y paga en efectivo en el local: Se confirma (tipo "2") el pago completo al finalizar servicio
  - Cancela con > 24h (o tiempo configurado): Se cancela (tipo "9") la pre-autorización completa
  - Cancela con < 24h: Se confirma el % de penalización configurado
  - No se presenta (no-show): Se confirma el 100% del importe

## C. Procesamiento de Pagos con Redsys

- Integración InSite (Principal):
  - Campos de pago en iframes hospedados por Redsys
  - Datos de tarjeta nunca tocan servidor del comercio
  - PCI-DSS SAQ A-EP (requisitos mínimos)
  - SDK JavaScript de Redsys: <https://sis.redsys.es/sis/NC/redsysV3.js>
  - Flujo: Captura en frontend → idOper → Confirmación REST desde backend
- Integración REST (Alternativa):
  - Para casos que requieran mayor control
  - Endpoint: <https://sis.redsys.es/sis/rest/trataPeticonREST>
  - PCI-DSS SAQ D (compliance completo requerido)
  - Manejo directo de datos de tarjeta
- Tokenización de tarjetas:
  - Parámetro DS\_MERCHANT\_IDENTIFIER: "REQUIRED" en primera transacción
  - DS\_MERCHANT\_COF\_INI: "S" para indicar credential-on-file
  - DS\_MERCHANT\_COF\_TYPE: "R" para recurrente

- Almacenar token devuelto (Ds\_Merchant\_Identifier)
  - Usar token en transacciones subsiguientes sin re-introducir tarjeta
- Webhooks/Notificaciones asíncronas:
  - URL de notificación en DS\_MERCHANT\_MERCHANTURL
  - Validación de firma HMAC SHA-256
  - Actualización de estado en base de datos
- Registro de todas las transacciones:
  - Importe, fecha, método
  - Estado (pending/authorized/captured/failed/refunded)
  - Logs de eventos de pasarela
  - Respuestas de Redsys

#### D. Facturación - **FUNCIONALIDAD MARCADA COMO FUTURO**

- Generación automática de facturas (FASE FUTURA)
- Formato PDF con diseño personalizado (FASE FUTURA)
- Numeración secuencial por organización (FASE FUTURA)
- Envío automático por email (FASE FUTURA)
- Descarga desde perfil de cliente (FASE FUTURA)
- Integración con contabilidad (FASE FUTURA)

#### E. Gestión Financiera - **FUNCIONALIDAD MARCADA COMO FUTURO**

- Dashboard con métricas (FASE FUTURA)
- Reportes exportables (Excel/PDF) (FASE FUTURA)
- Gestión de descuentos y cupones (FASE FUTURA)

#### Entidades de base de datos:

##### Payment

- Id (Guid)
- OrganizationId (Guid)
- AppointmentId (Guid?)
- CustomerId (Guid)
- Amount (decimal)
- Currency (string) // EUR
- PaymentMethod (Card/Cash/Transfer/Bizum)
- Status (Pending/Authorized/Captured/Failed/Refunded/PartiallyRefunded)
- RedsysOrderNumber (string) // Número de pedido único
- RedsysAuthCode (string) // Código de autorización

```

- RedsysResponse (string) // Código de respuesta (0000-0099)
- RedsysTransactionType (string) // 0=Auth, 1=PreAuth, 2=Confirm,
- RedsysCardNumber (string) // PAN enmascarado
- PaymentMethodId (Guid?) // Referencia a tarjeta guardada si
- ProcessedAt (DateTime?)
- RefundedAmount (decimal)
- RefundedAt (DateTime?)
- CreatedAt (DateTime)
- Metadata (jsonb) // Datos adicionales de Redsys

```

### 3.1.1.7 Sistema de Recordatorios

Prioridad: **MUST-HAVE** (crítico para reducir no-shows)

Funcionalidades:

#### A. Configuración de Recordatorios

- Por organización, configurar:
  - Cantidad de recordatorios (ej: 2 recordatorios)
  - Tiempos antes de la cita (ej: 24h y 2h antes)
  - Canales de comunicación (Email, WhatsApp, ambos)
  - Contenido del mensaje (plantillas personalizables)
  - Horarios de envío (no enviar de noche)
- Plantillas con variables dinámicas:
  - Nombre del cliente
  - Fecha y hora de cita
  - Nombre del servicio
  - Nombre del empleado
  - Dirección del local
  - Botón de confirmación/cancelación
  - Instrucciones previas (ej: no usar cremas 24h antes)

#### B. Recordatorios por Email

- Servicio de email transaccional: Amazon SES
- Características:
  - HTML responsive
  - Plain text alternativo

- Branding personalizado
  - Links de confirmación/cancelación con un click
  - Añadir al calendario (iCal/Google Calendar)
- Tracking:
  - Enviado exitosamente
  - Entregado
  - Abierto (open rate)
  - Click en links
  - Rebotado (bounce)

#### C. Recordatorios por WhatsApp (Se implementará en función de la aprobación de los costes operativos)

- Integración con WhatsApp Business API
- Proveedor: Meta (Facebook) + BSP (Business Solution Provider)
- Consideraciones:
  - Cliente debe haber dado opt-in (consentimiento)
  - Usar plantillas aprobadas por Meta
  - Categoría: "Utilidad" (más económica, ~ 0.01 € por mensaje en España)
  - Respuesta dentro de 24h es gratuita para el negocio
- Contenido del mensaje:
  - Texto con variables (nombre, fecha, hora)
  - Botones de acción (confirmar/cancelar/reagendar)
  - Link a WhatsApp Web para conversación directa
- Costos:
  - ~ 0.01 € por recordatorio en España (categoría Utilidad)
  - Estimado para 100 citas/mes con 2 recordatorios = 2 €/mes

#### D. Confirmación de Asistencia

- Cliente puede confirmar asistencia desde:
  - Link en email
  - Botón en WhatsApp
  - Aplicación móvil
- Al confirmar:
  - Actualiza estado de cita
  - Notifica al personal
  - Reduce prioridad en lista de recordatorios

#### E. Gestión de Opt-In/Opt-Out

- Cliente puede elegir:
  - Recibir o no recordatorios

- Canal preferido (email/whatsapp)
- Desactivar temporalmente
- Gestión de lista de supresión (bounces, quejas)
- Cumplimiento RGPD: consentimiento explícito

### Entidades de base de datos:

#### ReminderConfiguration

- Id (Guid)
- OrganizationId (Guid)
- ReminderOrder (int) // 1º, 2º, 3º recordatorio
- HoursBeforeAppointment (int)
- Channel (Email/WhatsApp/Both)
- IsActive (bool)
- MessageTemplateId (Guid)

#### MessageTemplate

- Id (Guid)
- OrganizationId (Guid)
- Name (string)
- Type (EmailReminder/WhatsAppReminder/Confirmation)
- Subject (string) // solo email
- Body (string) // con variables {{customerName}}, {{appointmentDate}},
- Language (string) // es-ES

#### ReminderLog

- Id (Guid)
- AppointmentId (Guid)
- ReminderConfigurationId (Guid)
- Channel (Email/WhatsApp)
- SentAt (DateTime)
- Status (Sent/Delivered/Failed/Opened/Clicked)
- ExternalMessageId (string) // ID de SES o WhatsApp
- ErrorMessage (string?)

### 3.1.1.8 Historial y Fotografías

Prioridad: **SHOULD-HAVE**

Funcionalidades:

- Subida de fotografías antes/después del servicio
- Almacenamiento en Amazon S3
- Organización por cliente y fecha
- Comparación lado a lado
- Galería privada (solo cliente y personal)
- Opción de compartir en redes (con consentimiento)
- Marca de agua con logo del negocio
- Expiración automática según RGPD (ej: 2 años)

Entidades de base de datos:

ServicePhoto

- Id (Guid)
- AppointmentId (Guid)
- Type (Before/After)
- S3Key (string)
- S3Bucket (string)
- UploadedBy (Guid) // EmployeeId
- UploadedAt (DateTime)
- IsPublic (bool)
- ExpiresAt (DateTime)

### 3.1.1.9 Gestión de Productos

FUNCIONALIDAD COMPLETA MARCADA COMO FUTURO

Prioridad: **NICE-TO-HAVE (FASE FUTURA)**

Funcionalidades (FUTURO):

- Catálogo de productos de venta (FASE FUTURA)
- Inventario (FASE FUTURA)

- Venta en el local (FASE FUTURA)
- E-commerce básico (FASE FUTURA)

#### Entidades de base de datos:

Product (FUTURO)

- Id (Guid)
- OrganizationId (Guid)
- Name (string)
- Description (string)
- SKU (string)
- Price (decimal)
- Stock (int)
- MinStockAlert (int)
- ImageUrl (string)
- IsActive (bool)

InventoryMovement (FUTURO)

- Id (Guid)
- ProductId (Guid)
- Quantity (int)
- MovementType (Purchase/Sale/Adjustment/Waste)
- Notes (string)
- CreatedAt (DateTime)

#### 3.1.10 Reportes y Analíticas

⚠ FUNCIONALIDAD COMPLETA MARCADA COMO FUTURO

Prioridad: **SHOULD-HAVE (FASE FUTURA)**

Funcionalidades (FUTURO):

- Dashboard ejecutivo (FASE FUTURA)
- Reportes operativos (FASE FUTURA)

- Reportes financieros (FASE FUTURA)
- Reportes de marketing (FASE FUTURA)
- Exportación a Excel/PDF (FASE FUTURA)

### 3.1.1.11 Aplicación Móvil

Prioridad: MUST-HAVE (Fase 2)

Funcionalidades para Clientes:

- Login/registro
- Buscar centros cercanos (si multi-tenant público)
- Ver catálogo de servicios
- Reservar citas
- Ver historial de citas
- Gestionar perfil y preferencias
- Gestionar tarjetas de crédito guardadas
- Ver fotografías antes/después
- Recibir notificaciones push
- Programa de fidelización
- Valorar servicios recibidos

Funcionalidades para Personal:

- Login con credenciales de empleado
- Ver agenda del día
- Recibir notificaciones de nuevas citas
- Confirmar/cancelar citas
- Registrar llegada del cliente (check-in)
- Marcar servicio como completado
- Ver perfil de cliente
- Registrar pagos en efectivo

Tecnología:

- React Native para iOS y Android
- Single codebase
- Notificaciones push: Firebase Cloud Messaging
- Sincronización en tiempo real con backend

## 3.2 Stack tecnológico

### 3.2.1 Backend

- Framework: ASP.NET Core 8.0 (LTS)
- Lenguaje: C# 8
- Patrón arquitectónico: Clean Architecture / Onion Architecture
- API: RESTful con ASP.NET Core Web API
- ORM: Entity Framework Core 8.0
- Autenticación: ASP.NET Core Identity + JWT
- Validación: FluentValidation
- Logging: Serilog con sinks a AWS CloudWatch
- Testing: xUnit, Moq, FluentAssertions

#### Librerías principales:

```
<PackageReference Include="Microsoft.EntityFrameworkCore" Version="8.0" />
<PackageReference Include="Npgsql.EntityFrameworkCore.PostgreSQL" Version="8.0" />
<PackageReference Include="Microsoft.AspNetCore.Authentication.JwtBearer" Version="8.0" />
<PackageReference Include="RedsysTPV.NetStandard" Version="3.1.0" />
<PackageReference Include="AWSSDK.S3" Version="3.7" />
<PackageReference Include="AWSSDK.SimpleEmail" Version="3.7" />
<PackageReference Include="Serilog.Sinks.AWSCloudWatch" Version="5.0" />
<PackageReference Include="FluentValidation.AspNetCore" Version="11.3" />
<PackageReference Include="Swashbuckle.AspNetCore" Version="6.5" />
<PackageReference Include="Hangfire.AspNetCore" Version="1.8" />
<PackageReference Include="MediatR" Version="12.0" />
```

#### Estructura del proyecto:

```
src/
├── ReservArte.API/           # API Controllers, Middleware
├── ReservArte.Application/   # Use Cases, DTOs, Interfaces
├── ReservArte.Domain/        # Entities, Value Objects, Aggregates
├── ReservArte.Infrastructure/ # Data Access, External Services
└── ReservArte.Shared/        # Common utilities, Constants

tests/
├── ReservArte.UnitTests/
```

```
|— ReservArte.IntegrationTests/  
|— ReservArte.E2ETests/
```

### 3.2.2 Frontend Web

#### Framework: React 18.2 + Vite

- Lenguaje: TypeScript 5.3
- Build Tool: Vite 5.0 (Hot Module Replacement ultra-rápido)
- Gestión de estado: Zustand o Redux Toolkit
- UI Framework: Tailwind CSS + shadcn/ui
- Formularios: React Hook Form + Zod
- Peticiones HTTP: Axios o TanStack Query
- Calendario: FullCalendar o react-big-calendar
- Gestión de fechas: date-fns o Day.js
- Autenticación: Custom JWT hooks

#### Razones para elegir Vite sobre Next.js:

- Rendimiento desarrollo: HMR instantáneo, arranque en milisegundos
- Simplicidad: Sin convenciones de routing complejas, ideal para SPA
- Flexibilidad: Control total sobre bundling y optimización
- Tamaño bundle: Tree-shaking más eficiente para aplicación SPA
- Costo: Sin vendor lock-in, completamente open-source
- DevEx: Experiencia de desarrollo superior para SPAs

#### Configuración de Vite ([vite.config.ts](https://vite.config.ts)):

```
import { defineConfig } from 'vite'  
  
import react from '@vitejs/plugin-react'  
  
import path from 'path'  
  
export default defineConfig({  
  plugins: [react()],  
  resolve: {  
    alias: {  
      '@': path.resolve(__dirname, './src'),  
    },  
  },  
})
```

```
server: {  
  port: 3000,  
  proxy: {  
    '/api': {  
      target: 'http://localhost:5000',  
      changeOrigin: true,  
    },  
  },  
},  
build: {  
  outDir: 'dist',  
  sourcemap: true,  
  rollupOptions: {  
    output: {  
      manualChunks: {  
        vendor: ['react', 'react-dom', 'react-router-dom'],  
        ui: ['@radix-ui/react-dialog', '@radix-ui/react-dropdown-menu'],  
      },  
    },  
  },  
},  
})
```

### Estructura del proyecto:

```
frontend-web/  
├─ src/  
|   ├─ components/  
|   |   ├─ ui/                # Componentes básicos  
|   |   └─ features/         # Componentes por feature  
|   |   └─ layouts/          # Layouts  
|   └─ lib/  
|       ├─ api/              # Cliente API  
|       └─ hooks/            # Custom hooks  
|       └─ utils/            # Utilidades
```

```
|   └─ stores/                # Estado global (Zustand/Redux)
|   └─ types/                 # TypeScript types
|   └─ App.tsx                # Componente raíz
|   └─ main.tsx               # Entry point
└─ public/                   # Assets estáticos
└─ index.html                 # HTML template
└─ vite.config.ts             # Configuración Vite
└─ package.json
```

### Script de inicio:

```
{
  "scripts": {
    "dev": "vite",
    "build": "tsc && vite build",
    "preview": "vite preview",
    "lint": "eslint . --ext ts,tsx --report-unused-disable-directives --max-warnings 0"
  }
}
```

### 3.2.3 Frontend Móvil

#### Framework: React Native 0.73

- Lenguaje: TypeScript 5.3
- Navegación: React Navigation 6
- Gestión de estado: Zustand
- UI Framework: React Native Paper o NativeBase
- Notificaciones: React Native Firebase
- Gestión de fechas: date-fns
- HTTP: Axios
- Almacenamiento local: AsyncStorage o MMKV

### 3.2.4 Base de Datos

#### RDBMS: PostgreSQL 15

- Hosting: Amazon RDS PostgreSQL
- Características utilizadas:
  - JSONB para datos flexibles
  - Row Level Security para multi-tenancy
  - Índices GiST para búsquedas fulltext
  - Particionamiento por organizationId
  - Point-in-Time Recovery (backups automáticos)

#### Schema Multi-Tenant:

- Enfoque inicial: Shared Database + Shared Schema con OrganizationId en todas las tablas
- Ventajas:
  - Menor costo inicial
  - Más fácil de gestionar
  - Ideal para pequeña/mediana escala
- Escalabilidad futura: Migrar a Database-per-Tenant para clientes Enterprise

## 3.3 Arquitectura técnica

### 3.3.1 Infraestructura AWS

#### 3.3.1.1 Servicios AWS Utilizados

##### Compute:

- AWS Elastic Beanstalk: Deployment simplificado de ASP.NET Core
  - O alternativamente: Amazon ECS Fargate para containers
- AWS Lambda: Funciones serverless para tareas asíncronas (envío de emails, procesamiento de imágenes)

##### Storage:

- Amazon RDS PostgreSQL: Base de datos principal
  - Instancia: db.t3.medium (2 vCPU, 4 GB RAM) para empezar
  - Multi-AZ para alta disponibilidad (producción)
- Amazon S3: Almacenamiento de archivos

- Fotografías de clientes
- Logos de organizaciones
- Backups

#### Networking:

- Application Load Balancer (ALB): Distribución de tráfico
- Amazon CloudFront: CDN para contenido estático de Vite
- Amazon Route 53: DNS y dominios personalizados

#### Seguridad:

- AWS Secrets Manager: Almacenamiento de secrets (API keys, DB credentials, Redsys keys)
- AWS Certificate Manager (ACM): Certificados SSL/TLS gratuitos
- AWS WAF: Firewall de aplicaciones web

#### Monitoring:

- Amazon CloudWatch: Logs, métricas y alarmas
- AWS X-Ray: Tracing distribuido

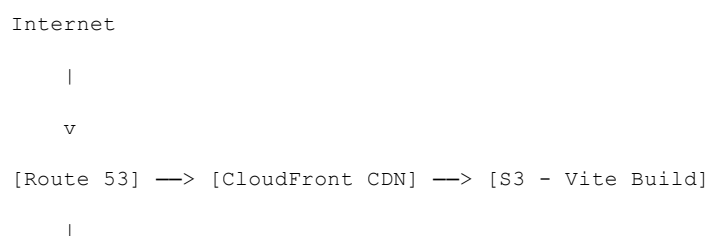
#### Email:

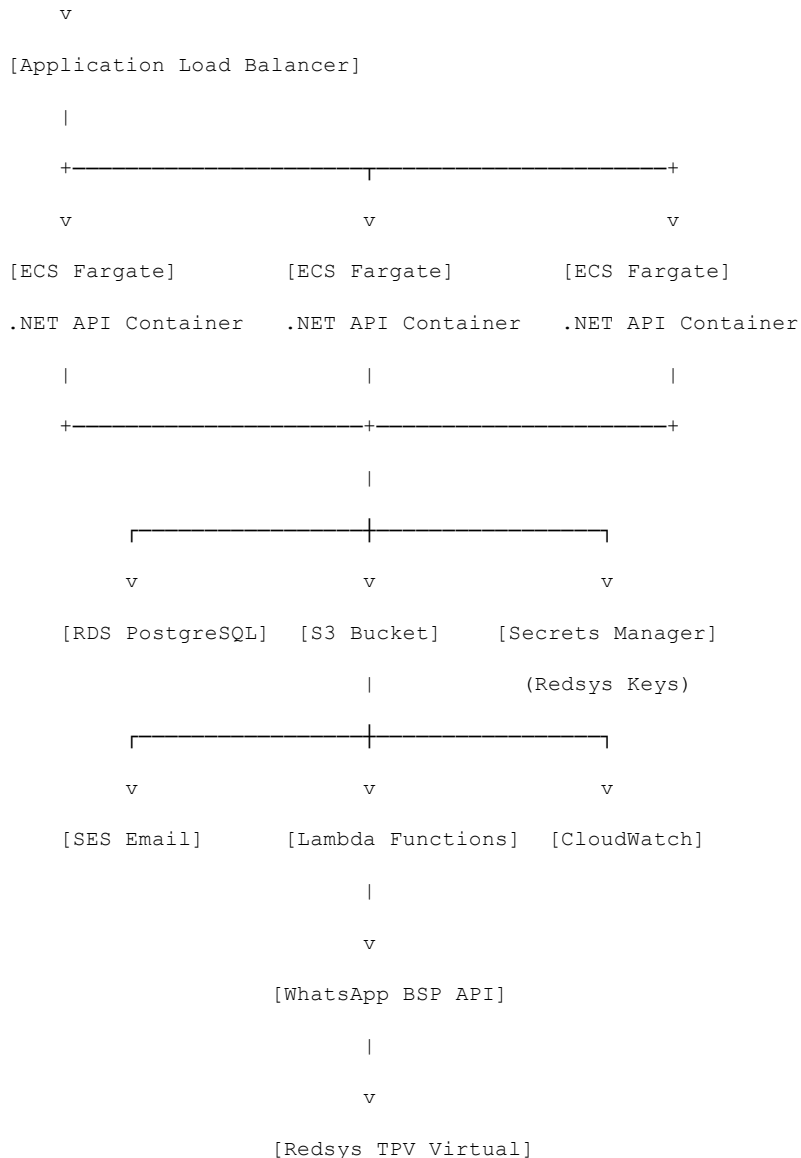
- Amazon SES: Envío de emails transaccionales
  - Costo: \$0.10 por 1,000 emails
  - Alta deliverability

#### Messaging:

- Amazon SNS: Notificaciones y pub/sub
- Amazon SQS: Colas para procesamiento asíncrono

### 3.3.1.2 Diagrama de Arquitectura AWS





### 3.3.2 Multi-Tenant Architecture

#### 3.3.2.1 Patrón Seleccionado: Shared Database + Shared Schema

Implementación:

##### 1. Aislamiento de datos:

- Todas las tablas incluyen columna `OrganizationId`
- Filtros globales en Entity Framework:

```
protected override void OnModelCreating(ModelBuilder modelBuilder)
```

```
{  
  
    modelBuilder.Entity<Appointment>()  
  
        .HasQueryFilter(a => a.OrganizationId == _currentOrganizationId);  
  
}
```

## 2. Identificación de tenant:

- Desde subdomain: `organizacion.reservarte.com`
- O desde header HTTP: `X-Organization-Id`
- O desde JWT claim: `organization_id`

## 3. Middleware de resolución de tenant:

```
public class TenantMiddleware  
{  
  
    public async Task InvokeAsync(HttpContext context)  
  
    {  
  
        var subdomain = ExtractSubdomain(context.Request.Host);  
  
        var organizationId = await _resolver.ResolveOrganizationId(subdomain);  
  
        _tenantService.SetCurrentTenant(organizationId);  
  
        await _next(context);  
  
    }  
  
}
```

## 4. Índices compuestos:

```
CREATE INDEX idx_appointments_org_date  
  
ON appointments (organization_id, appointment_date);
```

### 3.3.2.2 Escalabilidad: Plan de Migración Futuro

Para organizaciones grandes (>5000 citas/mes):

- Migrar a Database-per-Tenant
- Sharding horizontal por región geográfica
- Leer réplicas para reportes

### 3.3.3 Seguridad

#### 3.3.3.1 Autenticación y Autorización

Flujo de autenticación:

1. Usuario envía credenciales (email/password)
2. Backend valida contra ASP.NET Core Identity
3. Si válido, genera JWT token con claims:
  - user\_id
  - organization\_id
  - role
  - permissions
4. Frontend guarda token (httpOnly cookie o localStorage)
5. Token se envía en header `Authorization: Bearer <token>` en cada request

Autorización por roles:

- Admin: acceso total
- Manager: gestión de empleados, servicios, configuración
- Employee: solo sus citas y clientes
- Customer: solo sus datos

Implementación:

```
[Authorize(Roles = "Admin,Manager")]

public async Task<IActionResult> GetOrganizationSettings() { ... }

[Authorize(Policy = "CanManageAppointments")]

public async Task<IActionResult> CancelAppointment() { ... }
```

#### 3.3.3.2 Cifrado y Protección de Dato

En tránsito:

- TLS 1.3 obligatorio
- HSTS (HTTP Strict Transport Security)
- Certificate Pinning en apps móviles

En reposo:

- RDS encryption at rest (AES-256)
- S3 encryption (SSE-S3 o SSE-KMS)
- Secrets Manager para API keys y Redsys credentials

Datos sensibles:

- Passwords: bcrypt con salt
- Datos de pago: Tokenizados por Redsys, nunca almacenados directamente
- PII: tokenización cuando sea posible

### 3.3.3.3 Cifrado y Protección de Dato

SQL Injection:

- Entity Framework con queries parametrizadas
- Validación estricta de inputs

XSS (Cross-Site Scripting):

- React escapa automáticamente
- Content Security Policy headers

CSRF (Cross-Site Request Forgery):

- SameSite cookies
- Anti-forgery tokens en formularios

DDoS:

- AWS WAF con rate limiting
- CloudFront con Shield Standard

Brute Force:

- Rate limiting en login (10 intentos / hora)
- CAPTCHA después de 3 intentos fallidos
- Bloqueo temporal de cuenta

## 3.4 Integraciones técnicas destacadas

### 3.4.1 Integración con Pasarelas de Pago - Redsys







#### 3.4.1.1 Redsys InSite (Método Principal Recomendado)

Por qué InSite como método principal:

- PCI-DSS simplificado: SAQ A-EP vs SAQ D completo
- Seguridad mejorada: Datos de tarjeta nunca tocan el servidor del comercio
- Experiencia de usuario: Pago sin redirección externa
- Cumplimiento automático: Redsys maneja 3D Secure 2.x y SCA
- Costos reducidos: Auditorías PCI más económicas (~ 2k-10k € vs 20k-100k €+ anuales)

[Flujo InSite para ReservArte:](#)

Ventajas de InSite para ReservArte:

-  PCI-DSS SAQ A-EP (requisitos mínimos)
-  Datos de tarjeta en iframes de Redsys
-  Experiencia sin redirección
-  Tokenización integrada para guardar tarjetas
-  Costo de compliance reducido
-  Implementación más rápida (~1-2 semanas)

#### 3.4.1.2 Redsys REST API (Método Alternativo)

Cuándo usar REST en lugar de InSite:

- Necesidad de control total del flujo de pago
- Diseño completamente personalizado imposible con CSS
- Ya existe infraestructura PCI-DSS certificada

[Flujo REST para pre-autorización:](#)

Desventajas de REST para ReservArte:

- ~~❌~~ PCI-DSS SAQ D (compliance completo obligatorio)
- ~~❌~~ Costos de auditoría elevados (20k-100k €/año)
- ~~❌~~ Mayor responsabilidad de seguridad
- ~~❌~~ Implementación más compleja (~2-4 semanas)
- ~~❌~~ Mantenimiento continuo de seguridad

### 3.4.1.3 Costos de Redsys en España

Estructura de costos:

- Redsys es un procesador, no banco adquirente
- Los comercios contratan a través de su banco
- Costos varían por banco y volumen

Costos típicos 2025:

- Comisión por transacción: 1.0% - 1.5% (tarjetas ES/UE)
- Tarjetas internacionales: 2% - 3%
- Cuota mensual: 0 € - 50 € (según banco)
- Bizum: ~ 0.50 € por transacción
- Sin costos de setup de Redsys (puede haber del banco)

Comparativa:

Redsys es competitivo especialmente para comercios en España con alto volumen de transacciones nacionales.

Pasarela	Comisión UE	Comisión No-UE	Cuota Mensual
Redsys	1.0-1.5%	2-3%	0-50 €
Stripe	1.4% + 0.25 €	2.9% + 0.25 €	0 €
PayPal	2.9% + 0.35 €	3.4% + 0.35 €	0 €

### **3.5 Guía de estilos**

## **4. Metodología de trabajo**

### **4.1 Enfoque ágil (Scrum)**

- Sprints de 2 semanas
- Daily standups (15 minutos)
- Sprint review y retrospective
- Continuous Integration/Continuous Deployment (CI/CD)

### **4.2 Herramientas utilizadas**

- Gestión de proyecto: Jira o Linear
- Comunicación: Slack
- Control de versiones: GitHub
- CI/CD: GitHub Actions
- Documentación: Notion o Confluence

### **4.3 Roles**

- Product Owner: Cliente (centro de cejas)
- Scrum Master: Líder técnico del equipo
- Development Team: Desarrolladores Full-Stack
- QA Engineer: Testing y calidad

## 5. Estudio de mercado

### 5.1 DAFO

El análisis **DAFO** es una herramienta estratégica que permite evaluar la situación actual del proyecto **ReservArte** desde una perspectiva integral. Mediante esta técnica se analizan los factores **internos** - Fortalezas y Debilidades - y los **externos** - Oportunidades y Amenazas - que pueden influir en su desarrollo, implantación y viabilidad futura.

Este diagnóstico ofrece una visión global del entorno técnico, económico y social del proyecto, permitiendo identificar ventajas competitivas, riesgos potenciales y áreas de mejora. A partir de este análisis, se podrán definir estrategias adecuadas para potenciar el crecimiento, la sostenibilidad y la escalabilidad del sistema en el mercado digital de gestión de citas.

### ANÁLISIS INTERNO

#### Fortalezas

1. **Competencia técnica del equipo**

Los desarrolladores poseen una sólida formación en programación, bases de datos, diseño web y despliegue en la nube, adquirida en el Grado Superior en Desarrollo de Aplicaciones Web. Esto garantiza un desarrollo estructurado, seguro y mantenible.

2. **Uso de tecnologías modernas y escalables**

El stack tecnológico (ASP.NET Core, Vue.js/Nuxt.js, PostgreSQL y AWS) se alinea con los estándares de la industria actual. Esto no solo mejora el rendimiento y la seguridad, sino que también facilita futuras ampliaciones, como la conversión del sistema a modelo SaaS.

3. **Arquitectura modular y bien estructurada**

La arquitectura de tres capas favorece la separación de responsabilidades, la reutilización de código y la facilidad para implementar nuevas funcionalidades sin comprometer la estabilidad del sistema.

4. **Orientación al usuario y experiencia de uso**

El diseño de la interfaz prioriza la claridad y la accesibilidad, buscando reducir la curva de aprendizaje tanto para los administradores del negocio como para los clientes que reservan servicios.

5. **Metodología ágil de trabajo**

La aplicación de Scrum con reuniones semanales y planificación por sprints ha permitido un desarrollo iterativo, con entregas funcionales y

revisiones continuas, adaptándose a los cambios y garantizando un producto siempre operativo.

## Debilidades

### 1. Recursos y tiempo limitados

El desarrollo se ha llevado a cabo con un equipo reducido y una planificación ajustada. Esto ha implicado priorizar funcionalidades esenciales, dejando algunas ampliaciones para fases futuras.

### 2. Dependencia de integraciones externas

El uso de APIs de terceros como **Redsys** implica riesgos de mantenimiento, ya que cualquier cambio en estas plataformas puede afectar la operatividad del sistema.

### 3. Limitaciones de testing automatizado

Dado el enfoque educativo y los plazos, las pruebas unitarias y de integración no se han implementado en su totalidad, lo que puede dejar margen para errores no detectados en entornos productivos.

### 4. Ausencia de aplicación móvil en esta fase

Aunque está prevista en el roadmap, la app móvil no se incluye en esta versión inicial, limitando el acceso a usuarios que prefieran dispositivos móviles.

## ANÁLISIS EXTERNO

### Oportunidades

#### 1. Digitalización creciente de pequeños negocios

Muchos comercios locales están en proceso de transformación digital, lo que abre una amplia oportunidad de mercado para herramientas como ReservArte, que automatizan procesos sin requerir grandes inversiones.

#### 2. Demanda de soluciones personalizables y asequibles

A diferencia de competidores internacionales, ReservArte puede adaptarse a las necesidades específicas de cada cliente y ofrecer precios más ajustados, lo que refuerza su atractivo entre microempresas.

#### 3. Expansión mediante modelo SaaS (Software as a Service)

La estructura multi-tenant planificada permitirá ofrecer ReservArte como servicio en la nube, generando ingresos recurrentes y facilitando su adopción por parte de múltiples negocios sin necesidad de instalación local.

#### 4. Integración con canales digitales de comunicación

La automatización de recordatorios a través de WhatsApp o correo

electrónico mejora la fidelización del cliente y reduce ausencias, aportando un valor añadido que muchos competidores aún no ofrecen.

#### 5. **Potencial de crecimiento en sectores diversos**

Aunque nace del ámbito de la estética, el sistema es adaptable a cualquier negocio que funcione con citas o reservas (clínicas, gimnasios, talleres, academias, etc.), lo que multiplica su alcance potencial.

### **Amenazas**

#### 1. **Alta competencia en el sector de software de reservas**

Existen soluciones consolidadas como **Booksy**, **Fresha** o **Calendly**, que cuentan con reconocimiento de marca y grandes presupuestos de marketing. Esto exige que ReservArte destaque por su personalización y soporte cercano.

#### 2. **Evolución tecnológica rápida**

Las plataformas y frameworks cambian con frecuencia; mantener el software actualizado y compatible puede suponer una carga de mantenimiento constante.

#### 3. **Dependencia de servicios externos**

Problemas o cambios en APIs de Redsys podrían afectar el funcionamiento de funcionalidades críticas como los pagos o los recordatorios.

#### 4. **Riesgos de seguridad y protección de datos**

Al gestionar información sensible (datos personales, citas, pagos), la aplicación debe cumplir estrictamente con la normativa RGPD y aplicar medidas robustas de seguridad y cifrado.

#### 5. **Barreras de entrada en el mercado SaaS**

La transición a un modelo SaaS requerirá inversión en infraestructura, soporte técnico y marketing digital, lo que podría suponer un reto financiero en etapas iniciales.

## **5.2 Plan de negocio**

El modelo de negocio propuesto es SaaS (Software as a Service), donde cada negocio paga una cuota mensual por el uso de la plataforma.

Posibles líneas de ingresos:

- Suscripción mensual básica/premium.
- Servicios adicionales (analíticas, personalización visual, dominio propio).

### **5.3 Marketing y ODS**

## **6. Conclusiones**

Esta memoria técnica presenta un plan completo, detallado y viable para el desarrollo de ReservArte, una aplicación multi-tenant de gestión para centros de diseño de cejas en España.

### **Puntos Clave del Proyecto**

#### **Tecnología Moderna y Robusta:**

- Backend: ASP.NET Core 8.0
- Frontend Web: React 18 + Vite (HMR ultra-rápido)
- Frontend Móvil: React Native
- Base de Datos: PostgreSQL en AWS RDS
- Infraestructura: AWS con alta disponibilidad

#### **Cumplimiento Legal Estricto:**

- RGPD y LOPD compliant desde el diseño
- PCI-DSS SAQ A-EP con Redsys InSite
- Políticas de privacidad, cookies y términos
- EIPD para datos sensibles
- Datos permanecen en España/UE

#### **Sistema de Pagos Robusto:**

- Redsys InSite como método principal (PCI simplificado)
- Pre-autorizaciones para reducir no-shows
- Tokenización nativa para guardar tarjetas
- Penalizaciones automáticas por cancelaciones tardías
- Soporte para Bizum

### **Arquitectura Escalable:**

- Multi-tenant desde el inicio
- Escalado horizontal posible
- Optimización de costos por etapas
- Preparado para 100+ organizaciones

### **Notificaciones Multi-Canal:**

- Email con Amazon SES
- WhatsApp Business API (Fase 3+)
- Push notifications en apps móviles
- Recordatorios configurables
- Viabilidad Económica

### **Inversión:**

- MVP (4 meses): 92,560 €
- Proyecto completo (9 meses): 215,739 €

### **Costos operativos:**

- Inicio (1 org): ~200 €/mes
- Escalado (50 orgs): ~1,800 €/mes

### **Ingresos potenciales (SaaS):**

- Mes 12: 2,500 €/mes (30 clientes)
- Mes 24: 10,000 €/mes (100 clientes)

### **ROI esperado:**

- 18-30 meses

### **Riesgos Mitigados**

- Seguridad de pagos garantizada por Redsys
- Cumplimiento legal desde el diseño
- Arquitectura probada y escalable
- Stack tecnológico maduro y bien soportado
- Roadmap realista con hitos claros

## Próximos Pasos Inmediatos

- Aprobación del cliente y firma de contrato
- Setup de infraestructura AWS y repositorios
- Inicio del Sprint 1 de desarrollo
- Contacto con banco para credenciales Redsys
- Contratación de asesor legal RGPD

El proyecto ReservArte está técnicamente bien fundamentado, es viable económicamente, cumple con toda la normativa legal española y europea, y tiene un camino claro hacia la rentabilidad como plataforma SaaS.

## 7. Bibliografía

### Documentación Técnica

#### Frameworks y Librerías:

- ASP.NET Core: <https://docs.microsoft.com/aspnet/core>
- React: <https://react.dev>
- Vite: <https://vitejs.dev>
- React Native: <https://reactnative.dev>
- Entity Framework Core: <https://docs.microsoft.com/ef/core>
- Tailwind CSS: <https://tailwindcss.com/docs>
- shadcn/ui: <https://ui.shadcn.com>

#### Infraestructura:

- AWS Documentation: <https://docs.aws.amazon.com>
- AWS RDS PostgreSQL: <https://docs.aws.amazon.com/rds/>
- Amazon SES: <https://docs.aws.amazon.com/ses/>
- Amazon S3: <https://docs.aws.amazon.com/s3/>

#### Redsys:

- Portal desarrolladores Redsys: <https://pagosonline.redsys.es>
- Documentación técnica: <https://pagosonline.redsys.es/desarrolladores.html>
- Manual de integración InSite: Solicitar a banco adquirente
- Códigos de respuesta: Consultar documentación oficial

## RGPD y Legal

### Recursos oficiales:

- AEPD (Agencia Española de Protección de Datos): <https://www.aepd.es>
- RGPD Texto completo: <https://gdpr.eu>
- Guía de Cookies AEPD: <https://www.aepd.es/guias/guia-cookies.pdf>
- Guía de Análisis de Riesgos: <https://www.aepd.es/sites/default/files/2019-09/guia-analisis-de-riesgos.pdf>

## WhatsApp Business

### Documentación oficial:

- WhatsApp Business API: <https://business.whatsapp.com>
- Meta for Developers: <https://developers.facebook.com/docs/whatsapp>
- Precios WhatsApp: <https://business.whatsapp.com/products/platform-pricing>
- 360dialog Docs: <https://docs.360dialog.com>

### Categorías de mensajes:

- Utility: Recordatorios, confirmaciones
- Marketing: Promociones, ofertas
- Authentication: Códigos OTP
- Service: Respuestas a consultas

## Herramientas y Servicios

### Desarrollo:

- GitHub: <https://github.com>
- Vite documentation: <https://vitejs.dev/guide/>
- Figma: <https://www.figma.com>
- Postman: <https://www.postman.com>

### Monitoreo:

- Sentry: <https://sentry.io>
- AWS CloudWatch: <https://aws.amazon.com/cloudwatch/>
- Google Analytics: <https://analytics.google.com>

### Testing:

- TestFlight (iOS): <https://testflight.apple.com>
- Google Play Console (Android): <https://play.google.com/console>

### Calculadoras:

- AWS Pricing Calculator: <https://calculator.aws>
- Redsys Simulator: Solicitar acceso a banco

## 8. Anexos

### Anexo A: Glosario de Términos

#### A-E:

- API: Application Programming Interface
- AWS: Amazon Web Services
- BSP: Business Solution Provider (WhatsApp)
- CAC: Customer Acquisition Cost
- CAPTCHA: Completely Automated Public Turing test
- CDN: Content Delivery Network
- CI/CD: Continuous Integration / Continuous Deployment
- COF: Credential On File (Redsys)
- CRUD: Create, Read, Update, Delete
- DPA: Data Processing Agreement
- DPO: Data Protection Officer (Delegado de Protección de Datos)
- EIPD: Evaluación de Impacto en Protección de Datos

#### F-M:

- FUC: Número de comercio en Redsys
- HMAC: Hash-based Message Authentication Code
- HMR: Hot Module Replacement (Vite)
- JWT: JSON Web Token
- KPI: Key Performance Indicator
- LOPD: Ley Orgánica de Protección de Datos
- LSSI-CE: Ley de Servicios de la Sociedad de la Información
- MRR: Monthly Recurring Revenue

#### N-Z:

- MVP: Minimum Viable Product
- NPS: Net Promoter Score
- ORM: Object-Relational Mapping

- PAN: Primary Account Number (número de tarjeta)
- PCI-DSS: Payment Card Industry Data Security Standard
- RGPD: Reglamento General de Protección de Datos (GDPR en inglés)
- ROI: Return On Investment
- SaaS: Software as a Service
- SAQ: Self-Assessment Questionnaire (PCI-DSS)
- SCA: Strong Customer Authentication
- SDK: Software Development Kit
- SES: Simple Email Service (AWS)
- TLS: Transport Layer Security
- TPV: Terminal Punto de Venta
- VPC: Virtual Private Cloud

## **Anexo B: Contactos Recomendados**

### **Proveedores de Servicios**

#### **Redsys:**

- Contratar a través de tu banco comercial
- Bancos recomendados con Redsys:
- BBVA
- Santander
- CaixaBank
- Banco Sabadell

#### **WhatsApp BSP:**

- 360dialog: <https://www.360dialog.com>
- Twilio: <https://www.twilio.com/whatsapp>
- Vonage (ex-Nexmo): <https://www.vonage.com>

#### **Asesoría Legal RGPD:**

- Buscar despacho local especializado en RGPD y tech
- Verificar experiencia con startups SaaS
- Solicitar referencias

#### **Hosting Alternativo:**

- DigitalOcean: <https://www.digitalocean.com>
- Vultr: <https://www.vultr.com>
- Hetzner Cloud: <https://www.hetzner.com/cloud>