



DOCUMENTACIÓN PROPUESTA TÉCNICA

GESCOMPH

Jesús David Fierro Rivera
Brayan Santiago Guerrero Mendez

Contenido

1. Resumen Ejecutivo.....	3
1.1. Situación Actual	3
1.2. Problemática identificada.....	3
1.3. Beneficios Esperados	4
2. Introducción	5
2.1. Objetivo del documento	5
2.2. Alcance técnico	5
2.3. Audiencia objetivo.....	6
2.4. Referencias técnicas	6
2.5. Descripción general del sistema.....	6
2.6. Viabilidad técnica	7
2.7. Viabilidad económica	7
2.8. Viabilidad operativa.....	7
3. Descripción General	8
3.1. Propósito del sistema	8
3.2. Visión técnica del producto.....	8
3.3. Tipos de usuarios y dispositivos de acceso	9
3.4. Reglas generales del negocio (a nivel técnico).....	9
4. Arquitectura del sistema	10
4.1. Diagrama general	12
4.3. Tecnologías seleccionadas	13
4.4. Consideraciones de seguridad	13
Autenticación y autorización	13
Protección de datos	14
Servicios externos	14
5. Interfaz de usuario y prototipos	14
5.1. Principios de diseño de la UI	14
5.2. Descripción general de pantallas	15
6. Especificación de interfaces.....	15
6.1. Interfaces internas (API RESTful)	15
1. Web Layer → Business Layer	15
2. Business Layer → Data Layer	16
3. Entity Layer	16
Formato de intercambio interno	16
6.2. Interfaces externas	16

6.3.	Validaciones y estructuras de datos esperadas.....	17
7.	Plan de pruebas	18
7.1.	Objetivo del plan	18
7.2.	Tipos de pruebas consideras	18
7.3.	Criterios de aceptación simulados.....	19
7.4.	Revisión de posibles errores comunes	19
8.	Planificación del Proyecto	19
8.1.	Metodologías de Desarrollo.....	19
8.2.	Cronograma general y por fases	20
8.2.1.	Fase 1	20
8.2.2.	Fase 2: Diseño del sistema	20
8.2.3.	Fase 3: Desarrollo.....	21
8.2.4.	Fase 4: Pruebas y cierre.....	21
9.	Diagramas	21
9.1.	Diagrama de Flujo.....	21
9.2.	Diagrama de Clases	22
9.3.	Diagrama de secuencia	22
9.5.	Diagrama de Casos de Uso	22

1. Resumen Ejecutivo

El proyecto **GESCOMPH (Gestión Comercial Palermo Huila)** tiene como finalidad estructurar una plataforma destinada a la administración de los arriendos de establecimientos comerciales ubicados en plazas públicas del municipio de Palermo, Huila. Este sistema centraliza la información relacionada con plazas, locales, contratos de arrendamiento, disponibilidad de espacios, solicitudes de visitas, obligaciones mensuales y control del estado de ocupación.

La solución está orientada a ofrecer un entorno organizado y accesible para los diferentes perfiles de usuario involucrados. Los administradores pueden gestionar la información institucional y llevar seguimiento de contratos y ocupación de locales; los arrendatarios disponen de un espacio para consultar sus contratos, obligaciones y fechas relevantes; y los ciudadanos interesados pueden explorar la oferta disponible y registrar solicitudes para conocer los locales de manera presencial.

1.1. Situación Actual

La Alcaldía de Palermo – Huila enfrenta dificultades en la gestión administrativa de los arriendos de bienes inmuebles destinados a actividades comerciales en plazas públicas y otros espacios municipales. Los procedimientos actuales presentan limitaciones que afectan la organización de la información, la trazabilidad de los contratos y la eficiencia en la supervisión de los compromisos adquiridos por los arrendatarios.

Actualmente, la administración carece de mecanismos integrados que permitan un control centralizado y actualizado del estado de los locales, contratos y obligaciones mensuales. Esto ha generado las siguientes problemáticas:

- **Ausencia de herramientas de seguimiento y alertas automatizadas**, lo que incrementa el riesgo de vencimientos no atendidos o acumulación de obligaciones atrasadas.
- **Procesos administrativos altamente manuales**, generando duplicidad de tareas, riesgo de errores y tiempos prolongados en la gestión de trámites asociados al arriendo.
- **Carencia de reportes y estadísticas consolidadas**, lo que limita la capacidad de análisis y toma de decisiones relacionadas con la ocupación, regularización contractual y planeación del uso de los espacios públicos.
- **Escasa visibilidad para la ciudadanía respecto a locales disponibles**, reduciendo la transparencia del proceso y dificultando la participación de posibles interesados en acceder a un espacio comercial.

1.2. Problemática identificada

La ausencia de un sistema centralizado para la administración de los arriendos municipales ha generado un conjunto de limitaciones que afectan directa e

indirectamente la eficiencia institucional. Entre los principales problemas identificados se destacan:

- **Dificultad para acceder a información consolidada y actualizada**, ya que los registros se encuentran dispersos o dependen de procedimientos manuales, lo que retrasa la consulta y verificación del estado de los contratos y de los locales.
- **Control insuficiente sobre procesos críticos de seguimiento contractual**, como renovaciones, vencimientos, obligaciones mensuales y estados de pago, lo cual dificulta una supervisión oportuna y confiable.
- **Persistencia de procesos manuales en tareas esenciales**, generando riesgo de errores en la actualización de datos, duplicidad de información y tiempos prolongados en la atención de trámites administrativos.
- **Falta de mecanismos para el seguimiento organizado de obligaciones contractuales**, incluyendo dificultades para identificar moras, pagos pendientes o incumplimientos.
- **Limitada interacción digital con la ciudadanía**, lo que impide que los interesados consulten fácilmente la disponibilidad de locales, la información básica de los espacios o los canales para solicitar citas.
- **Escasa transparencia en la difusión de la oferta de espacios públicos arrendables**, lo que reduce la visibilidad del proceso y afecta la participación de potenciales interesados.

1.3. Beneficios Esperados

El sistema GESCOMPH brindará a la Alcaldía de Palermo – Huila una herramienta orientada a organizar, centralizar y dar seguimiento a la información relacionada con los arriendos de bienes inmuebles municipales. Su utilización permitirá contar con procesos más estructurados, información disponible de manera ordenada y una interacción más clara entre administración, arrendatarios y ciudadanía.

Los beneficios esperados se agrupan en tres dimensiones:

A. Beneficios operativos

- **Automatización de procesos clave**, como el seguimiento de vencimientos, registro de obligaciones y control de pagos asociados a los contratos.
- **Reducción de errores derivados de registros manuales**, mediante formularios estructurados y validaciones que evitan datos inconsistentes.
- **Disponibilidad centralizada de la información**, permitiendo que los funcionarios accedan a datos actualizados en un único punto.
- **Uniformidad en los procedimientos**, al contar con flujos definidos para la gestión de contratos, establecimientos y citas.

B. Beneficios administrativos

- **Acceso organizado a datos sobre contratos, ocupación de locales y estados de pago**, lo que permite una visión clara de la situación contractual municipal.
- **Transparencia interna**, al mantener información sistematizada y documentada sobre los procesos administrativos asociados a los arriendos.

- **Disponibilidad de información consolidada**, facilitando consultas para fines institucionales.

C. Beneficios para la ciudadanía

- **Consulta pública de locales disponibles**, permitiendo a cualquier ciudadano acceder a información general sobre los espacios arrendables del municipio.
- **Solicitud de citas de manera digital**, evitando desplazamientos iniciales y facilitando el acceso al proceso de arriendo.
- **Acceso de arrendatarios a su información contractual**, incluyendo fechas importantes, obligaciones y estado del contrato.

2. Introducción

2.1. Objetivo del documento

El presente documento técnico tiene como propósito describir la propuesta de solución para el sistema GESCOMPH (Gestión Comercial Palermo Huila). Incluye la estructura general del software, la organización arquitectónica, los componentes que conforman la solución, los modelos de datos, los criterios de seguridad y los lineamientos tecnológicos que orientan su desarrollo.

El documento está dirigido a los instructores y miembros del equipo académico encargados del análisis, construcción y validación del sistema. Asimismo, sirve como referencia técnica para comprender la composición interna del proyecto y el funcionamiento de cada uno de sus módulos.

2.2. Alcance técnico

El sistema propuesto corresponde a una plataforma accesible desde navegadores web y dispositivos móviles, orientada a la centralización de procesos relacionados con plazas, establecimientos, contratos, obligaciones mensuales, pagos, solicitudes de citas y consultas ciudadanas.

La arquitectura se estructura bajo un enfoque por capas que separa responsabilidades y organiza los componentes del sistema de la siguiente manera:

- **Capa Entity:**
Define las entidades del dominio y los modelos que representan la información fundamental del sistema (plazas, establecimientos, contratos, usuarios, pagos, obligaciones, citas, etc.).
- **Capa Data:**
Contiene la configuración de la base de datos, la definición de repositorios, consultas y mecanismos de persistencia mediante un ORM.
- **Capa Business:**
Integra las reglas del dominio, validaciones, servicios de operación y procedimientos que regulan el comportamiento funcional del sistema.
- **Capa Web:**
Expone los servicios mediante una API REST, gestiona entradas y salidas de información y opera como punto de comunicación entre el backend y el frontend.

La solución no contempla integración con sistemas contables institucionales externos ni módulos de validación jurídica de contratos. Las funcionalidades de pago que se incluyan al sistema se utilizan únicamente en modo de pruebas, sin manejo de transacciones reales en esta etapa académica.

2.3. Audiencia objetivo

Este documento está dirigido a los siguientes perfiles:

- **Equipo de desarrollo:**
Para consultar la estructura técnica, los componentes del sistema y las directrices de diseño necesarias para la construcción del proyecto.
- **Instructores académicos:**
Para la revisión, validación técnica y seguimiento del avance del proyecto durante el proceso formativo.
- **Representantes institucionales asociados al proceso de arriendo municipal:**
Para comprender el enfoque técnico adoptado y la forma en que la solución propuesta organiza la información relacionada con los arriendos.
- **Integrantes responsables de pruebas y soporte académico:**
Para garantizar la coherencia técnica en las actividades de validación funcional y continuidad del proyecto dentro del contexto académico.

2.4. Referencias técnicas

Las siguientes fuentes técnicas fueron consultadas para la elaboración de la propuesta tecnológica:

- **Angular 20 – Documentación oficial:**
<https://angular.io>
- **ASP.NET Core / .NET 8 – Documentación oficial:**
<https://learn.microsoft.com/dotnet>
- **Entity Framework Core – ORM y persistencia:**
<https://learn.microsoft.com/ef/core>
- **SQL Server – Motor de base de datos:**
<https://learn.microsoft.com/sql>

2.5. Descripción general del sistema

GESCOMP (Gestión Comercial Palermo Huila) es un sistema de información orientado a la administración digital de los arriendos de establecimientos comerciales ubicados en plazas públicas del municipio de Palermo, Huila. La plataforma centraliza información sobre plazas, establecimientos, contratos, obligaciones mensuales, pagos en ambiente de pruebas, solicitudes ciudadanas y estado de ocupación de los locales.

El sistema contempla tres perfiles principales:

- **Administradores:**
Acceden a los módulos de plazas, establecimientos, contratos, usuarios,

parámetros del sistema, obligaciones mensuales, estado de pagos y solicitudes de citas. Pueden visualizar información consolidada sobre ocupación, vencimientos y actividades asociadas a los arriendos.

- **Arrendatarios:**

Consultan sus contratos vigentes, obligaciones mensuales, fechas relevantes, estado de cumplimiento y notificaciones asociadas. Tienen acceso a un calendario informativo y a su información contractual.

- **Ciudadanos no autenticados:**

Consultan la disponibilidad de locales aplicando filtros, visualizan información general de los establecimientos y pueden registrar solicitudes de citas para conocer espacios disponibles.

El sistema permite consultas públicas e interacción digital básica, y está diseñado para funcionar de forma accesible desde navegadores web en distintos dispositivos, mediante una interfaz adaptable y organizada.

2.6. Viabilidad técnica

El desarrollo del sistema GESCOMPH es técnicamente viable dentro del contexto académico, dado que se basa en herramientas ampliamente conocidas y de uso gratuito, como .NET 8, Angular 20, SQL Server Express y Visual Studio Community. Estas tecnologías permiten implementar las funcionalidades previstas para la gestión de plazas, establecimientos, contratos, obligaciones mensuales, pagos en ambiente de pruebas y solicitudes ciudadanas.

2.7. Viabilidad económica

El proyecto presenta viabilidad económica en el contexto académico, ya que todas las herramientas empleadas para su desarrollo son de acceso gratuito y no requieren licencias comerciales. El uso de entornos como Visual Studio Community, Angular CLI, SQL Server Express y repositorios de control de versiones gratuitos evita costos directos asociados al software.

2.8. Viabilidad operativa

La propuesta del sistema se ajusta a los flujos que se analizan en el proceso de arriendo de espacios municipales. Las funcionalidades planteadas permiten organizar información sobre contratos, locales, ocupación, pagos y solicitudes ciudadanas, de forma coherente con los procesos estudiados.

El uso de interfaces diferenciadas para administradores, arrendatarios y ciudadanos facilita la comprensión de las tareas previstas para cada perfil dentro del proyecto académico. El sistema se plantea como una herramienta accesible desde navegadores web, lo que permite su uso sin requerimientos adicionales de instalación o capacitación especializada.

3. Descripción General

GESCOMP (Gestión Comercial Palermo Huila) es un sistema de información orientado a la administración organizada de los procesos asociados al arriendo de establecimientos comerciales ubicados en plazas públicas del municipio de Palermo, Huila. La plataforma centraliza datos relacionados con plazas, establecimientos, contratos, obligaciones mensuales, pagos en ambiente de pruebas, solicitudes de citas y estados de ocupación. El sistema contempla tres perfiles principales de usuario:

- **Administradores:**
Acceden a los módulos de gestión de plazas, establecimientos, contratos, usuarios, parámetros del sistema, obligaciones mensuales, estados de pago y solicitudes de citas. Pueden consultar información sobre ocupación y eventos contractuales registrados.
- **Arrendatarios:**
Consultan sus contratos vigentes, obligaciones generadas, fechas relevantes, historial de pagos, estado de cumplimiento y notificaciones internas o por correo relacionadas con su relación contractual.
- **Ciudadanos no autenticados:**
Consultan locales disponibles mediante filtros y registran solicitudes de citas para conocer establecimientos libres, sujetas a validación por parte del administrador.

El sistema permite la gestión de información de manera centralizada e incluye funciones como generación de obligaciones mensuales, cálculo de mora, validación de pagos mediante pasarela en ambiente de pruebas, registro de solicitudes ciudadanas y administración de accesos mediante roles y permisos.

La plataforma es accesible desde navegadores web en dispositivos de escritorio y móviles, utilizando una interfaz adaptable a distintos tamaños de pantalla.

3.1. Propósito del sistema

El propósito de GESCOMP es ofrecer una plataforma informática que centralice la información relacionada con los procesos de arriendo de establecimientos comerciales administrados por el municipio de Palermo, Huila. El sistema organiza datos sobre plazas, establecimientos, contratos, obligaciones mensuales, pagos en ambiente de pruebas, solicitudes de citas y estados de ocupación, permitiendo que estas operaciones se gestionen de manera estructurada dentro de un entorno digital.

El sistema permite que los administradores gestionen la información institucional asociada a los arriendos; que los arrendatarios consulten su información contractual, obligaciones y estados registrados; y que los ciudadanos accedan a información pública sobre locales disponibles y puedan registrar solicitudes de visita.

3.2. Visión técnica del producto

GESCOMP se concibe como un sistema estructurado bajo un enfoque por capas, que separa responsabilidades y organiza la solución de la siguiente manera:

- **Capa Entity:**
Define las entidades del dominio y los modelos utilizados para representar contratos, plazas, establecimientos, usuarios, obligaciones mensuales, pagos, citas y demás elementos del sistema.
- **Capa Data:**
Contiene la configuración del acceso a datos mediante Entity Framework Core, el contexto de base de datos, los repositorios y las definiciones de mapeo hacia la base de datos relacional.
- **Capa Business:**
Implementa las reglas del dominio, validaciones, generación de obligaciones mensuales, cálculo de mora, administración del estado de ocupación y servicios asociados a los procesos operativos.
- **Capa Web:**
Expone los servicios mediante una API REST construida en .NET 8, encargada de recibir solicitudes, procesarlas y comunicarse con las capas internas del sistema.
- **Frontend:**
Desarrollado de forma independiente utilizando Angular 20, consume los servicios expuestos por la API, administra la interacción con los usuarios y presenta la interfaz en navegadores web.

La solución contempla el uso de pagos en ambiente de pruebas mediante una pasarela externa y no incluye integraciones con sistemas contables ni módulos jurídicos.

3.3. Tipos de usuarios y dispositivos de acceso

El sistema contempla tres tipos de usuario:

- **Administrador:**
Accede a los módulos de plazas, establecimientos, contratos, usuarios, parámetros del sistema, obligaciones mensuales, estado de pagos y solicitudes de citas.
- **Arrendatario:**
Consulta sus contratos, obligaciones generadas, estados asociados, fechas relevantes y notificaciones del sistema.
- **Ciudadano:**
Consulta locales disponibles y registra solicitudes de citas sin necesidad de autenticación.

3.4. Reglas generales del negocio (a nivel técnico)

- El sistema registra obligaciones mensuales asociadas a cada contrato y conserva su historial.
- Un local solo puede tener un contrato activo a la vez y mantiene un estado que indica si se encuentra disponible u ocupado.
- Cada contrato se asocia únicamente a un arrendatario y a un establecimiento específico.

- No se permiten contratos superpuestos en fechas ni asignados a locales no disponibles.
- Los pagos se gestionan en ambiente de pruebas mediante integración con una pasarela externa, y los cambios de estado se registran mediante notificaciones automáticas.
- Las solicitudes de citas realizadas por ciudadanos se registran en el sistema y quedan pendientes de aprobación o rechazo por parte del administrador.
- Los arrendatarios solo pueden ver información vinculada a su propio contrato y no pueden acceder a datos de otros usuarios.
- El sistema funciona de forma independiente y no depende de plataformas contables externas.

4. Arquitectura del sistema

La arquitectura del sistema GESCOMP se estructura bajo un enfoque por capas con separación clara de responsabilidades. Esto permite organizar el proyecto de manera ordenada y mantener una estructura coherente entre los distintos módulos funcionales.

Las capas que conforman la solución son las siguientes:

- **Entity Layer (Dominio)**

Incluye los modelos que representan las entidades centrales del sistema, tales como:

- Plaza
- Establecimiento
- Contrato
- Usuario
- Obligación mensual
- Pago
- Cita

En esta capa se definen únicamente las estructuras del dominio y las reglas invariantes que forman parte de la naturaleza de cada entidad. No contiene dependencias hacia otras capas ni componentes externos.

- **Data Layer (Persistencia)**

Encapsula todo lo relacionado con la administración de datos:

- Configuración de Entity Framework Core
- DbContext
- Configuraciones de entidades
- Consultas específicas requeridas por los módulos del sistema}

La capa Data interactúa con la base de datos relacional y expone las operaciones necesarias para almacenar, consultar y actualizar información.

- **Business Layer (Aplicación)**

Contiene los servicios encargados de gestionar las operaciones del sistema.
Incluye:

- Lógica asociada a la creación, actualización y finalización de contratos
- Reglas para la generación de obligaciones mensuales
- Validación de estados relacionados con ocupación de establecimientos
- Cálculo de mora (cuando corresponda)
- Procesamiento de solicitudes de citas
- Servicios de seguridad (roles, permisos, autenticación mediante tokens)
- Validaciones con FluentValidation
- Mapeo entre entidades y DTOs mediante Mapster

Esta capa funciona como punto intermedio entre la API y la persistencia.

- **Web Layer (API):** Expone los servicios a través de una API REST construida en ASP.NET Core.

Sus responsabilidades incluyen:

- Recepción de solicitudes del frontend
- Enrutamiento mediante controladores
- Autenticación y autorización mediante JWT y roles
- Validación de datos de entrada
- Manejo de respuestas en formato JSON
- Gestión de errores y excepciones
- Versionado de endpoints cuando sea necesario

La API interactúa únicamente con la capa Business y no contiene lógica operativa interna.

- **Frontend (Aplicación Angular) :** El frontend se desarrolla de forma independiente utilizando Angular, consumiendo la API en formato REST.

Sus responsabilidades incluyen:

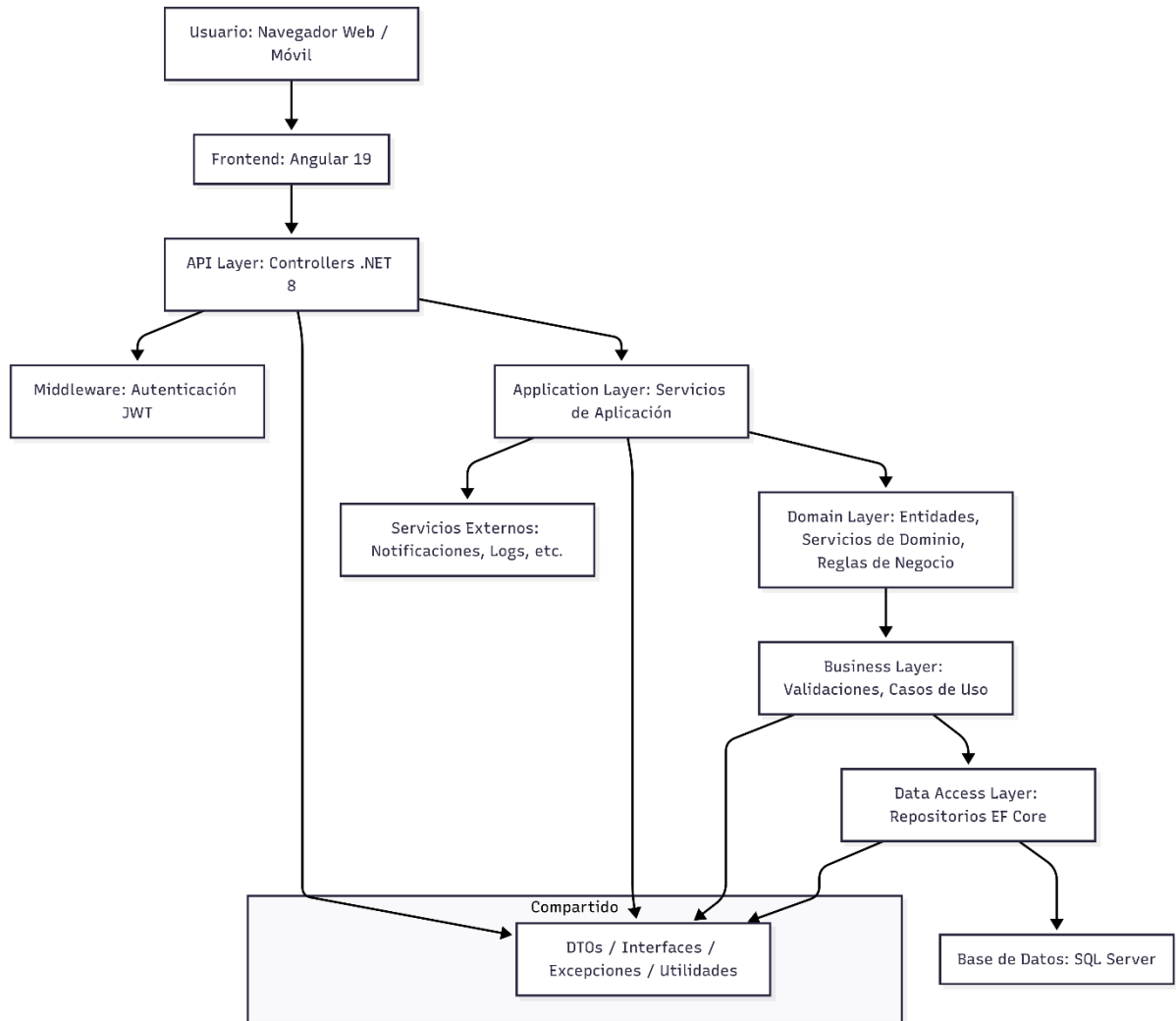
- Presentar la interfaz para administradores, arrendatarios y ciudadanos
- Gestionar formularios, tablas, filtros e interacción general
- Administrar el estado local de cada vista
- Consumir servicios mediante HTTP
- Gestionar autenticación basada en tokens
- Adaptarse a dispositivos móviles y de escritorio mediante diseño responsivo

- **Servicios externos integrados :**El sistema utiliza servicios externos para funcionalidades específicas:

- **Pasarela de pagos** en ambiente de pruebas (notificaciones y validación de transacciones).

- **Correo electrónico** para notificaciones a arrendatarios y ciudadanos.
 - **Servicio de almacenamiento de imágenes** para plazas y establecimientos.
- Estas integraciones se consumen desde la capa Business mediante adaptadores o servicios específicos.

4.1. Diagrama general



4.2. Componentes y responsabilidades

Capa	Componentes principales	Responsabilidad Clave
Entity	Clases: Usuario, contrato, establecimiento. DbContext, Migraciones	Modelo de dominio
Data	Repositorios	Persistencia de datos
Business	Servicios, validaciones, lógica de negocios	Procesos internos

Web	Servicios HTTP, Rutas	Comunicación con el cliente
-----	-----------------------	-----------------------------

4.3. Tecnologías seleccionadas

El sistema GESCOMPH utiliza un conjunto de tecnologías de uso académico y libre acceso que permiten estructurar la solución de manera organizada y consistente. Estas tecnologías se clasifican según su función dentro del sistema:

Frontend

- Angular 20 – Framework para construcción de aplicaciones web SPA.
- Angular Material – Componentes de interfaz bajo estándares de diseño.
- TypeScript – Lenguaje base para la implementación del frontend.

Backend

- .NET 8 / ASP.NET Core – Implementación de la API REST.
- C# – Lenguaje de programación principal.
- Entity Framework Core 8 – ORM para acceso y persistencia de datos.
- Mapster – Mapeo entre entidades, DTOs y modelos de transporte.
- FluentValidation – Validación estructurada de entradas.

Base de Datos

- SQL Server

Servicios Externos

- Mercado Pago (sandbox) – Registro y validación de pagos mediante checkout y webhooks.
- SMTP / correo electrónico – Envío de notificaciones a arrendatarios y ciudadanos.
- Servicio de almacenamiento de imágenes (Cloudinary u otro).

DevOps y herramientas de apoyo

- Git y GitHub – Control de versiones.
- Trello – Gestión de tareas académicas.
- Postman / Thunder Client – Pruebas de API.
- Docker (si aplica) – Entorno de despliegue y contenedores.
- Visual Studio / VS Code – Entornos de desarrollo.

4.4. Consideraciones de seguridad

El sistema incorpora mecanismos básicos orientados al control de acceso y protección de la información gestionada en el proyecto académico. Entre las medidas adoptadas se encuentran:

Autenticación y autorización

- **Autenticación basada en JWT**, con generación de access token y refresh token.
- **Control de acceso por roles**, diferenciando permisos para administradores, arrendatarios y ciudadanos.
- Restricción de endpoints según nivel de privilegio y reglas del sistema.

Protección de datos

- **Cifrado de contraseñas** mediante algoritmos de hashing seguros.
- **Validación estricta de entradas** usando FluentValidation para prevenir datos inconsistentes.
- Eliminación lógica de registros sensibles para preservar historial cuando corresponda.

Servicios externos

- Validación de **webhooks** provenientes de la pasarela de pagos en ambiente de pruebas.
- Manejo seguro de claves privadas utilizadas para pagos, correo o almacenamiento de imágenes.

5. Interfaz de usuario y prototipos

5.1. Principios de diseño de la UI

El diseño de la interfaz de usuario se basa en los siguientes principios:

Consistencia visual

- Uso de la paleta institucional definida para el proyecto.
- Componentes reutilizables para formularios, tablas y tarjetas.
- Diseño modular apoyado en guías como Material Design.

Diseño responsivo

- Adaptación automática de la interfaz a dispositivos de escritorio, tablet y móvil.
- Reestructuración de columnas, tarjetas y elementos según el ancho de pantalla.

Navegación clara

- Menú lateral organizado por módulos.
- Flujos estructurados y consistentes.
- Mensajes informativos cuando se ejecutan acciones poco frecuentes.

Retroalimentación del sistema

- Indicadores visuales para operaciones en curso (spinners).
- Mensajes de validación en formularios.
- Banners y alertas en estados relevantes.
- Tooltips para aclarar funciones específicas.

5.2. Descripción general de pantallas

Pantallas para Ciudadano (no autenticado)

- **Pantalla de Inicio:** Presenta información general y accesos para consulta de locales disponibles.
- **Pantalla de Consulta de Locales:** Listado filtrable con vista detallada de cada establecimiento.
- **Pantalla de Solicitud de Cita:** Formulario para registrar solicitudes de visita a locales disponibles.

Pantallas para Arrendatario

- **Login:** Acceso mediante correo y contraseña.
- **Panel inicial:** Resumen de contrato, obligaciones y fechas relevantes.
- **Mis Contratos:** Consulta detallada de contratos asociados al usuario, vista de fechas de pago, obligaciones registradas y estado asociado.
- **Configuración de usuario:** Edición de información básica y cambio de contraseña.

Pantallas para Administrador

- **Dashboard del Administrador:** Resumen de datos sobre ocupación, contratos activos, locales disponibles y citas registradas.
- **Gestión de Plazas:** Operaciones de registro, edición y consulta.
- **Gestión de Establecimientos:** Administración del inventario de locales y su estado actual.
- **Gestión de Contratos:** Registro y consulta de contratos, incluyendo obligaciones mensuales y su estado.
- **Gestión de Usuarios:** Administración de cuentas, asignación de roles y control de acceso.
- **Gestión de Citas:** Revisión, aprobación o rechazo de solicitudes ciudadanas.
- **Parámetros del sistema:** Configuración de datos como IVA, UVT y tasas aplicables.

6. Especificación de interfaces

6.1. Interfaces internas (API RESTful)

Las capas internas del sistema se comunican mediante contratos definidos que separan responsabilidades y evitan dependencias directas entre componentes. La interacción interna se establece del siguiente modo:

1. Web Layer → Business Layer

La capa Web expone controladores que reciben solicitudes HTTP, y a través de inyección de dependencias consumen servicios definidos en la capa Business.

- La Web Layer no implementa lógica de dominio.

- Los controladores procesan solicitudes, validan entradas, gestionan autorizaciones y delegan la operación al servicio correspondiente.

2. Business Layer → Data Layer

La capa Business utiliza interfaces que representan operaciones de acceso a datos. Estas interfaces son implementadas en la capa Data.

- El acceso a la base de datos se realiza mediante Entity Framework Core.
- La Business Layer nunca accede directamente al contexto ni a entidades persistentes sin pasar por las abstracciones definidas.

3. Entity Layer

Define las entidades que representan el modelo de dominio. No depende de ninguna otra capa.

Formato de intercambio interno

- DTOs para entrada y salida de datos.
- Entidades para persistencia.
- Mapeos realizados con Mapster.

6.2. Interfaces externas

El sistema interactúa con servicios y dispositivos externos mediante mecanismos definidos:

1. Frontend Web (Angular)

- El navegador consume la API mediante solicitudes **HTTP**.
- Navegadores compatibles: Chrome, Edge, Firefox, Opera.
- Flujo de autenticación basado en **JWT**.
- Las peticiones se realizan mediante HttpClient en Angular.

2. Pasarela de pagos (sandbox)

El sistema utiliza una pasarela de pago en ambiente de pruebas para validar transacciones:

- Envío de preferencia de pago desde la API hacia el servicio externo.
- Recepción de **webhooks** de confirmación o cambio de estado.
- Manejo de tokens y claves privadas desde configuración.

3. Servicio de correo electrónico

- Envío de correos para confirmación de cita o recuperación de acceso.
- Uso de un servidor SMTP o proveedor equivalente.
- Mensajes generados desde la capa Business.

4. Servicio de almacenamiento de imágenes

- Carga de fotografías de establecimientos a un servicio externo.
- Los enlaces resultantes se almacenan en la base de datos.
- Las imágenes se consultan desde el frontend directamente.

6.3. Validaciones y estructuras de datos esperadas

El sistema aplica validaciones tanto en frontend como en backend para garantizar consistencia y evitar datos incompletos.

Validaciones en Frontend (Angular)

- Campos requeridos.
- Formatos de correo electrónico.
- Restricción de valores numéricos (área, montos).
- Fechas válidas y orden cronológico (inicio/fin).
- Estados seleccionados desde listas controladas (select).

Validaciones en Backend (.NET 8)

Realizadas mediante **FluentValidation**:

- Longitud mínima y máxima.
- Unicidad donde aplique (correo, documento, nombre de plazas o módulos).
- Validación de estados del dominio (disponibilidad del establecimiento, fechas de contrato, obligaciones mensuales).
- Validación de existencia de entidades relacionadas.

Estructuras de datos (DTOs)

ContratoDto

Incluye información representativa:

- Id
- Arrendatario
- Establecimiento
- FechaInicio
- FechaFin
- Estado
- ValorMensual

EstablecimientoDto

- Id
- Nombre
- Plaza
- Área
- Tipo
- Estado
- Imágenes asociadas

PagoDto

- Id
- ContratId
- Mes
- Valor
- EstadoPago
- FechaRegistro

CitaDto

- Id
- LocalId
- Información del ciudadano
- Fecha solicitada
- Estado

Manejo de errores

La API responde con:

- Formato **JSON**
- Estructura estándar:
 - {
 - "message": "Descripción del error",
 - "errors": ["detalle 1", "detalle 2"],
 - "status": 400
 - }
- Códigos HTTP definidos:
 - 400: Error de validación
 - 401: No autenticado
 - 403: Acceso no autorizado
 - 404: No encontrado
 - 409: Conflicto (ej. contrato duplicado o local ocupado)
 - 500: Error interno

7. Plan de pruebas

7.1. Objetivo del plan

El objetivo de este plan es establecer los criterios de verificación funcional y técnica del sistema GESCOMPH bajo un entorno simulado, asegurando que los módulos definidos cumplan con los requerimientos estipulados y que el comportamiento general del sistema sea coherente, estable y predecible, a pesar de que no se realizará un despliegue real.

7.2. Tipos de pruebas consideradas

Tipo de prueba	Descripción
Pruebas unitarias	Validación de métodos individuales de lógica de negocio (por ejemplo: cálculo de mora, validación de contrato, etc.).
Pruebas de validación	Simulación de entrada de datos en formularios para asegurar el cumplimiento de reglas definidas.

Pruebas de interfaz (UI)	Revisión de la disposición y comportamiento visual esperado en cada pantalla.
--------------------------	---

7.3. Criterios de aceptación simulados

Un módulo o funcionalidad se considerará aceptado si cumple con los siguientes criterios:

Cumple con todos los requisitos funcionales documentados.

Las validaciones de datos funcionan correctamente en frontend y backend.

El flujo de navegación permite completar las tareas sin errores.

El sistema responde con mensajes claros ante errores o acciones inválidas.

Las restricciones por rol están correctamente implementadas y no hay fugas de acceso.

El diseño UI es visualmente consistente, responsivo y usable.

7.4. Revisión de posibles errores comunes

Durante la validación conceptual del sistema, se identificaron posibles errores comunes que se deben considerar en una futura implementación:

Errores de validación mal manejados: como permitir guardar un contrato con campos vacíos o sin validaciones cruzadas.

Falta de retroalimentación visual: ausencia de confirmaciones o errores que confundan al usuario.

Desalineación entre estados: ejemplo, un pago registrado no refleja correctamente el estado del contrato.

Problemas de navegación: botones sin funcionalidad o rutas incorrectas entre vistas.

Errores de permisos: usuarios accediendo a módulos restringidos por error en la lógica de roles.

Estos puntos se documentan con el fin de preverlos durante el desarrollo, incluso si el sistema no será desplegado en esta etapa académica.

8. Planificación del Proyecto

8.1. Metodologías de Desarrollo

El desarrollo del sistema GESCOMPH se realizará utilizando la metodología **Scrum**, un marco ágil de trabajo orientado a la entrega incremental de valor funcional, mediante iteraciones cortas, enfocadas en la colaboración con el cliente y en la mejora continua del producto.

Scrum permite adaptar el proceso de desarrollo a los cambios razonables que surjan durante el ciclo del proyecto, manteniendo visibilidad del progreso y alineando las prioridades del equipo técnico con las necesidades reales de la Alcaldía de Palermo – Huila.

Principales características de la metodología aplicada:

- **Iteraciones cortas (sprints):** el proyecto se dividirá en ciclos de trabajo regulares (normalmente entre 2 y 4 semanas), al final de los cuales se entregarán incrementos funcionales del sistema.
- **Backlog priorizado:** los requerimientos funcionales y no funcionales se organizan en un backlog gestionado por el Product Owner, con revisiones frecuentes de prioridades.
- **Revisión y demostración continua:** al final de cada sprint se presentarán avances funcionales para revisión por parte de los usuarios clave, facilitando ajustes tempranos.
- **Adaptación continua:** tras cada entrega se realiza una retrospectiva del sprint para identificar oportunidades de mejora en el proceso de desarrollo.
- **Colaboración activa con el cliente:** los representantes institucionales participarán como *stakeholders activos*, revisando entregables, validando funcionalidades y aportando retroalimentación de valor.

El desarrollo teórico del sistema GESCOMP se organiza en **cuatro fases principales**, siguiendo una metodología estructurada que permite simular el ciclo completo de vida del software. A continuación, se describe cada fase, sus actividades y entregables claves

8.2. Cronograma general y por fases

8.2.1. Fase 1

Objetivo: Definir claramente los requisitos funcionales, no funcionales y las condiciones técnicas y operativas del sistema.

Actividades clave:

- Levantamiento de requerimientos con el cliente.
- Definición de alcance funcional y técnico.
- Redacción del Documento de Requisitos del Software (SRS).
- Análisis de viabilidad técnica, económica y operativa.
- Revisión de tecnologías y arquitectura seleccionada.

Entregables:

- Documento SRS completo.
- Análisis de viabilidad.
- Cronograma base del proyecto.
- Estructura general del sistema y actores.

8.2.2. Fase 2: Diseño del sistema

Objetivo: Establecer la arquitectura técnica, los flujos lógicos y las interfaces de usuario del sistema.

Actividades clave:

- Diseño de la arquitectura en N capas (Entity, Data, Business, Web).
- Modelado de base de datos y relaciones entre entidades.
- Elaboración de diagramas (casos de uso, flujo, clases, secuencia).
- Creación de prototipos visuales y pantallas clave.

- Definición de interfaces internas (API REST) y validaciones.

Entregables:

- Documento de arquitectura técnica.
- Diagramas estructurales y de comportamiento.
- Prototipos visuales de la UI.
- Especificación de interfaces.

8.2.3. Fase 3: Desarrollo

Objetivo: Simular el desarrollo funcional del sistema según la arquitectura definida, sin llegar a desplegar el producto.

Actividades clave:

- Construcción de estructura base en Visual Studio y Angular.
- Implementación de modelos, controladores y servicios (estructura).
- Simulación de interacciones API (con ejemplos de endpoints).
- Documentación de flujos de negocio (crear contrato, registrar pagos, etc.).
- Validación interna del modelo técnico.

Entregables:

- Estructura funcional de carpetas y módulos (no ejecutable).
- Código estructurado de componentes clave (simulado).
- Registro de funcionalidades diseñadas.

8.2.4. Fase 4: Pruebas y cierre

Objetivo: Simular la validación del sistema a través de casos de prueba conceptuales y documentar los resultados esperados.

Actividades clave:

- Diseño de plan de pruebas teóricas por módulo.
- Revisión de criterios de aceptación y control de calidad.
- Simulación de errores comunes y respuestas esperadas.
- Documentación final técnica.

Entregables:

- Plan de pruebas documentado.
- Casos de prueba representativos por rol.
- Documento final de análisis técnico.
- Presentación de resultados (académica).

9. Diagramas

9.1. Diagrama de Flujo

9.1.1. Diagrama de Flujo General Administrador

9.1.2. Diagrama de Flujo del Usuario

9.1.3. Diagrama de Flujo General del Arrendatario

9.2. Diagrama de Clases

9.3. Diagrama de secuencia

9.3.1. Diagrama General de Secuencia del Administrador

9.3.2. Diagrama General de Secuencias del Usuario

9.3.3. Diagrama General de Secuencia del Arrendatario

9.4. Diagrama General de Procesos

9.5. Diagrama de Casos de Uso