**TRIFECTA - Tinajas Rustic**

**(DAS) Documento Arquitectura de Software**

**Versión 1.0**

**Identificación de Documento**

| **Identificación** | Documento DAS |
| --- | --- |
| **Proyecto** | Tinajas Rustic |
| **Versión** | 1.0 |

| **Documento mantenido por** | Vicente Vallejos  Francisca Sepulveda  Matias Gatica |
| --- | --- |
|  |  |
| **Fecha de última revisión** | 02/09/2024 |
| **Fecha de próxima revisión** |  |
| **Fecha de próxima revisión** |  |

| **Documento aprobado por** |  |
| --- | --- |
| **Fecha de última aprobación** |  |

**Historia de cambios**

| **Fecha** | **Versión** | **Descripción** | **Autor** |
| --- | --- | --- | --- |
| 02/09/2024 | 1.0 | Creación e Inicio del Informe | Vicente Vallejos  Francisca Sepulveda  Matias Gatica |
| 03/09/2024 | 1.2 | Modificación de Diagrama de Despliegue y Objetos | Vicente Vallejos  Francisca Sepulveda  Matias Gatica |
| 04/09/2024 | 1.3 | Corrección Diagrama de Componentes | Matias Gatica |
| 06/09/2024 | 1.4 | Finalización del Documento | Vicente Vallejos  Matias Gatica  Francisca Sepulveda |

# Indice

[**Indice 2**](#_heading=h.5r2iqgxqhq0w)

[**Introducción 3**](#_heading=h.30j0zll)

[**Alcance 3**](#_heading=h.1fob9te)

[**Referencias 3**](#_heading=h.3znysh7)

[**Arquitectura de Software 4**](#_heading=h.2et92p0)

[**Objetivos y Restricciones de la Arquitectura 5**](#_heading=h.tyjcwt)

[Objetivos de la arquitectura 5](#_heading=h.pelz0g0ihaa)

[Restricciones de la Arquitectura 6](#_heading=h.1t3h5sf)

[**Arquitectura Lógica 6**](#_heading=h.fsufomm1qe0w)

[Diagrama de Clases 7](#_heading=h.f6e5u3oisi4p)

[Diagrama de Comunicación 8](#_heading=h.ebynjnqxjrkd)

[Diagrama de Secuencia 8](#_heading=h.agxzn92j0a8d)

[**Arquitectura de Procesos 10**](#_heading=h.17dp8vu)

[**Arquitectura de desarrollo 12**](#_heading=h.3rdcrjn)

[Diagrama de paquete: 12](#_heading=h.lqjye86vf6f9)

[Diagrama de Componentes: 13](#_heading=h.b58734u5ael)

[Diagrama de Actividad: 14](#_heading=h.b58734u5ael)

[**Arquitectura física 15**](#_heading=h.26in1rg)

[**Escenarios 16**](#_heading=h.lnxbz9)

[Modelo de Casos de Uso 16](#_heading=h.35nkun2)

[Caso de uso de usuario Tinajas Rustic 17](#_heading=h.vf6mro4q81h0)

[Caso de uso Administrador Tinajas Rustic 18](#_heading=h.ghyzf1qo68xp)

[**Especificación de Casos de Uso Relevantes** 19](#_heading=h.1ksv4uv)

[Especificación de los Escenarios de Calidad Relevantes 19](#_heading=h.44sinio)

[**Tamaño y desempeño 21**](#_heading=h.2jxsxqh)

# Introducción

El proyecto Tinajas Rustic surge con el propósito de desarrollar una solución tecnológica integral para mejorar la gestión de reservas y la experiencia del cliente en una empresa dedicada al arriendo y utilización de tinajas, saunas, y piscinas. En un contexto donde la digitalización y la automatización son esenciales para optimizar los procesos empresariales, este proyecto busca modernizar y simplificar la interacción entre los clientes y la empresa, permitiendo una gestión más eficiente y accesible de los servicios ofrecidos.

El sistema propuesto incluirá funcionalidades clave como la creación de cuentas de usuario, la reserva en línea de servicios, el procesamiento seguro de pagos, y la gestión de comentarios y valoraciones. Además, se implementará un panel administrativo robusto que permitirá a la empresa gestionar de manera eficaz las reservas, supervisar el historial de transacciones, y generar reportes detallados para la toma de decisiones estratégicas.

El enfoque del proyecto no solo está en la eficiencia y seguridad del sistema, sino también en la creación de una experiencia de usuario amigable y accesible, que permita a los clientes realizar sus reservas y consultas de manera sencilla desde cualquier dispositivo, en cualquier momento. Esto será posible mediante el desarrollo de una plataforma web y móvil que responda a las necesidades específicas tanto de los usuarios finales como del equipo administrativo.

# Alcance

El proyecto Tinajas Rustic tiene como objetivo desarrollar una plataforma web y móvil que permita gestionar de manera eficiente las reservas y el arriendo de tinajas, saunas y piscinas ofrecidos por la empresa. Esta plataforma incluirá tanto una interfaz orientada a los clientes como un panel administrativo para la gestión interna. Los componentes clave del alcance del proyecto son los siguientes:

* Gestión de Usuarios: Registro, autenticación (correo y redes sociales), gestión de perfiles, y recuperación de contraseñas.
* Gestión de Reservas: Búsqueda de disponibilidad, confirmación, modificación, cancelación de reservas y notificaciones.
* Procesamiento de Pagos: Integración de pagos en línea y acceso al historial de pagos.
* Valoraciones y Comentarios: Sistema para que los usuarios dejen y visualicen valoraciones, con capacidad de moderación por parte de administradores.
* Panel de Administración: Herramientas para gestionar reservas, generar reportes y manejar la administración de la plataforma.
* Diseño Responsivo y Pruebas: Garantizar la funcionalidad en dispositivos móviles y asegurar la seguridad y usabilidad del sistema.
* Mantenimiento y Soporte: Documentación completa y soporte post-despliegue para asegurar la continuidad operativa del sistema.

# Referencias

A continuación, se listan las referencias de la documentación anexa:

| **Nombre** | **Url** |
| --- | --- |
| Acta de Constitución | [Project Charter](https://docs.google.com/document/d/1k7LuqFmH7p4k4WYwaEYPXRDUWO8IwgEt/edit?usp=drive_link&ouid=115511273292806994700&rtpof=true&sd=true) |
| Informe ERS | [ERS](https://docs.google.com/document/d/15vAqL5X363wMpC29RGaktMskmI6BTINJ/edit?usp=drive_link&ouid=115511273292806994700&rtpof=true&sd=true) |
| Planilla de toma de requerimientos | [Historias de Usuario](https://docs.google.com/document/d/1VMd9oA0KIbMSkPwgd_rHftZTA-9M0X22/edit?usp=drive_link&ouid=115511273292806994700&rtpof=true&sd=true) |

# Arquitectura de Software

La arquitectura del sistema Tinajas Rustic está representada siguiendo el enfoque de vistas 4+1, con base en las recomendaciones del proceso unificado. Las vistas incluidas en este documento son:

* Vista de Casos de Uso y Escenarios de Calidad
  + Descripción: Presenta los casos de uso más significativos del sistema, mostrando los actores involucrados y describiendo sus casos de uso asociados. Esta vista también aborda los escenarios de calidad más relevantes para la arquitectura.
  + Función: Relacionar los diferentes elementos del sistema a través de los casos de uso o escenarios que ilustran cómo interactúan todos los componentes. Ejemplos de casos de uso incluyen la gestión de reservas, procesamiento de pagos, y la administración de servicios ofrecidos.
* Vista de Metas y Restricciones
  + Descripción: Detalla las restricciones tecnológicas, normativas, estándares, y otros factores que influyen en las decisiones arquitectónicas y en el proceso de desarrollo.
  + Función: Asegurar que la arquitectura del sistema cumpla con las metas del proyecto y se ajuste a las limitaciones impuestas, como el uso de tecnologías específicas (e.g., Angular para frontend, MongoDB para la base de datos), requisitos de rendimiento, y consideraciones legales.
* Vista Lógica
  + Descripción: Presenta la arquitectura del sistema en varios niveles de refinamiento, mostrando los módulos lógicos principales, sus responsabilidades y sus dependencias.
  + Función: Mostrar cómo los componentes principales del sistema proveen los servicios a los usuarios finales. Esto incluye la estructura de la interfaz web para reservas, el sistema de autenticación de usuarios, y la administración de servicios.
* Vista de Procesos
  + Descripción: Explica los procesos involucrados en el sistema, incluyendo la comunicación y sincronización entre los diferentes módulos durante la ejecución. Hace énfasis en aspectos como la concurrencia, rendimiento, y escalabilidad.
  + Función: Garantizar que la arquitectura soporte los requisitos de rendimiento, concurrencia y escalabilidad, permitiendo una experiencia fluida y eficiente tanto para los administradores como para los clientes.
* Vista de Implementación
  + Descripción: Describe los componentes del sistema que serán desplegados y sus dependencias, además de la arquitectura desde la perspectiva de los desarrolladores.
  + Función: Guiar el proceso de desarrollo, asegurando que todos los elementos se integren de manera coherente y eficiente. Esto incluye la estructura de los componentes de frontend y backend, y su interacción con la base de datos y otros servicios.

# Objetivos y Restricciones de la Arquitectura

A continuación, se revisan las metas y restricciones de la arquitectura.

## Objetivos de la arquitectura

A partir de las reuniones realizadas y del análisis de los requerimientos, se han identificado los principales conductores iniciales de la arquitectura, que corresponden a las metas arquitectónicas fundamentales para el sistema Tinajas Rustic:

* **Desempeño**: Garantizar un alto desempeño en el sistema, asegurando que todas las operaciones, como la gestión de reservas, el procesamiento de pagos, y la generación de reportes, se realicen de manera eficiente y con tiempos de respuesta óptimos para los usuarios, tanto clientes como administradores. Esto incluye la implementación de estándares de calidad en todas las etapas del desarrollo.
* **Tolerancia a fallos**: Diseñar e implementar un sistema que mantenga sus funciones esenciales a pesar de posibles fallos. Esto implica la creación de mecanismos de respaldo y recuperación para asegurar que las reservas, pagos y otros datos críticos no se pierdan ni se vean comprometidos durante fallos inesperados.
* **Seguridad**: Mantener un alto nivel de confidencialidad y protección de datos dentro del sistema, especialmente en lo que respecta a la información personal de los usuarios y datos sensibles relacionados con pagos y reservas. Esto incluye la implementación de prácticas robustas de seguridad, como el cifrado de datos y el control de acceso basado en roles.
* **Modificabilidad/Reuso**: Facilitar la capacidad de realizar cambios o modificaciones en el sistema de manera eficiente y dentro de los límites de los requerimientos acordados. La arquitectura debe permitir la evolución del sistema a lo largo del tiempo, permitiendo la adición de nuevas funcionalidades o la mejora de las existentes sin comprometer la estabilidad del sistema.
* **Operatividad**: Asegurar que todas las actividades de planificación, implementación y mantenimiento del sistema se realicen de manera efectiva y activa. Los encargados del proyecto deben mantenerse proactivos en la gestión y operación del sistema, asegurando que todas las funcionalidades se mantengan alineadas con las necesidades del negocio y de los usuarios. La operatividad también incluye la capacidad del sistema para ser monitoreado y ajustado en tiempo real para garantizar su correcto funcionamiento.

## Restricciones de la Arquitectura

Las principales restricciones de la arquitectura de software son los límites o condiciones que afectan las decisiones arquitectónicas y el diseño de un sistema. Estas restricciones pueden provenir de varios factores, como las necesidades del negocio, la tecnología, el entorno operativo y las expectativas del usuario. Aquí hay algunas de las principales restricciones:

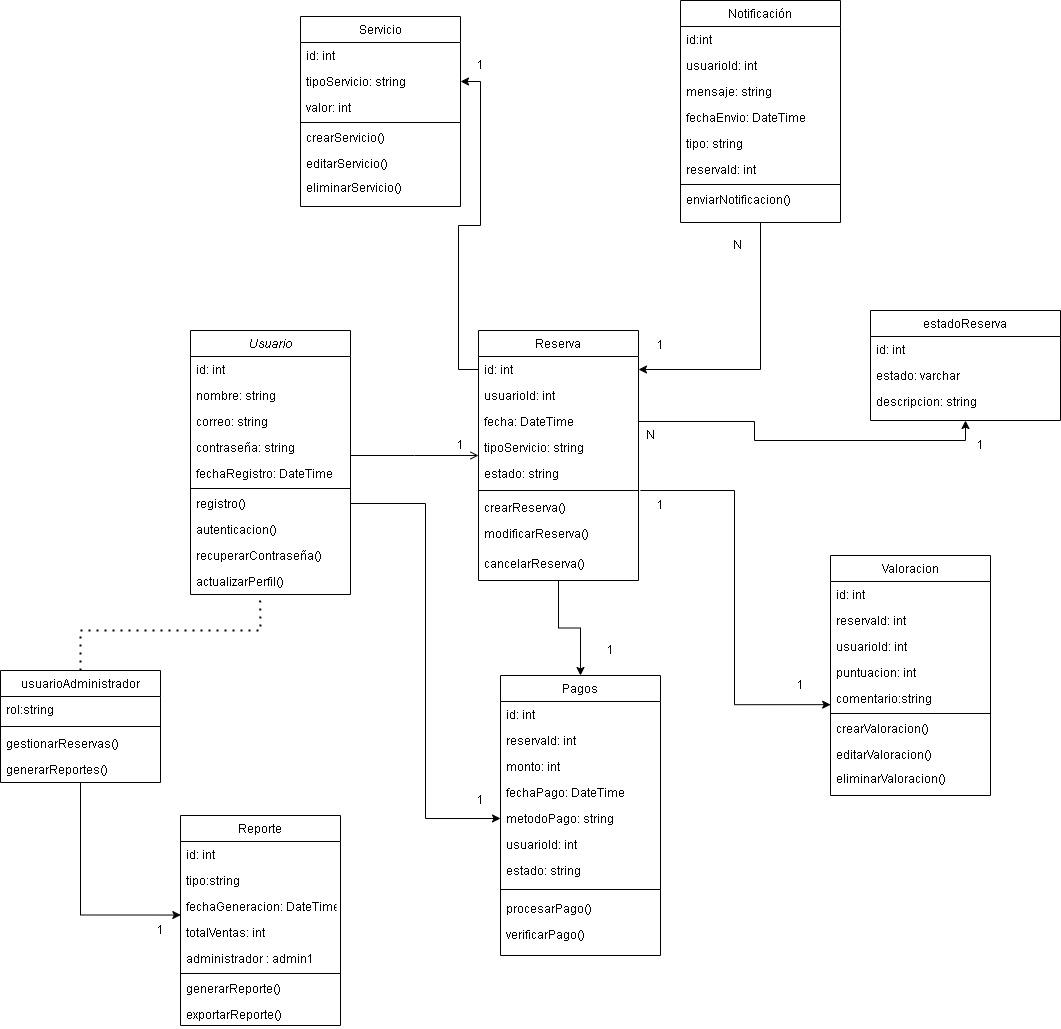
* Requerimientos de Rendimiento: La arquitectura debe cumplir con ciertos requisitos de velocidad, capacidad de respuesta y capacidad para manejar grandes volúmenes de usuarios o datos sin degradación del servicio.
* Escalabilidad: La arquitectura debe poder escalar de manera efectiva con el crecimiento del número de usuarios, transacciones o datos, sin requerir rediseños significativos.
* Seguridad: Se debe diseñar una arquitectura que proteja contra ataques, cumpla con las normativas y ofrezca controles de acceso adecuados para proteger la integridad y confidencialidad de los datos.
* Compatibilidad y Portabilidad: El sistema debe ser compatible con las plataformas y tecnologías existentes, y, en algunos casos, ser portátil para funcionar en múltiples entornos o sistemas operativos.
* Mantenibilidad: La arquitectura debe facilitar el mantenimiento y la evolución del sistema, permitiendo que el código se modifique, actualice o amplíe con el tiempo sin un gran costo.

# Arquitectura Lógica

En esta sección, se muestra una vista lógica de la aplicación expresado en dos diagramas, uno de ellos que presenta la parte estructural o estática de la aplicación (módulos), y la otra vista que representa la parte dinámica (componentes y conectores).

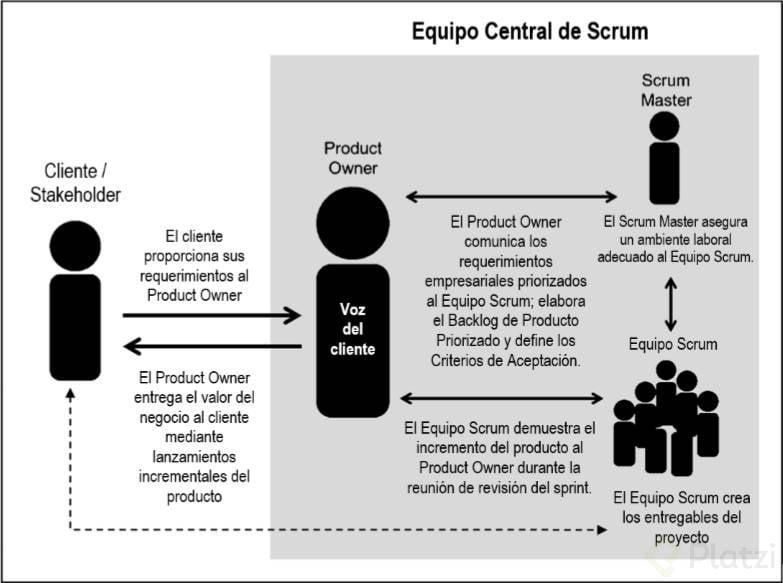
URL:[Diagrama de Clase](https://drive.google.com/file/d/1z-Wd-mRdCPCodqMB3sOx0NS45b9NebeQ/view?usp=drive_link)

## Diagrama de Clases



Representa la estructura estática del sistema, mostrando las clases principales que intervienen, sus atributos, métodos y las relaciones entre ellas. Entre las clases clave se encuentran Usuario, Reserva, Servicio, Pago, y Notificación. Las relaciones entre las clases están definidas principalmente por asociaciones, herencias y composiciones, lo que refleja cómo las diferentes partes del sistema interactúan entre sí.

## Diagrama de Comunicación



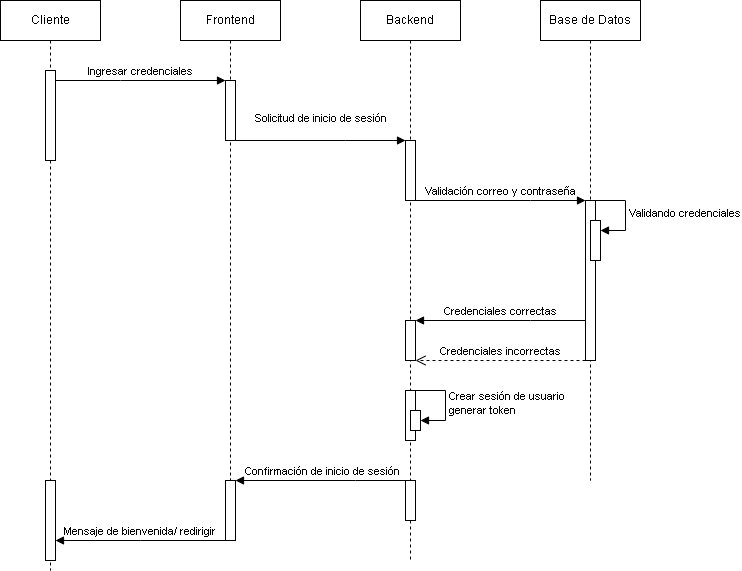
Este diagrama visualiza claramente cómo el Product Owner centraliza la comunicación entre el cliente y el equipo de desarrollo, asegurando que se cumplan las expectativas del cliente, mientras el Scrum Master asegura un ambiente de trabajo óptimo para que el equipo pueda cumplir con sus objetivos.

## Diagrama de Secuencia

El Diagrama de Secuencia de Tinajas Rustic es una representación visual que describe cómo interactúan los distintos componentes del sistema en un flujo temporal para realizar una tarea específica. Este diagrama se enfoca en la secuencia de mensajes intercambiados entre los objetos o actores (como el usuario, el sistema de reservas, el sistema de pagos, etc.) para completar un proceso determinado.

URL:[Carpeta de Diagramas de Secuencia](https://drive.google.com/drive/folders/1K8ycJvHyNK4jfWKOdKqaM953Z5NUD12U?usp=drive_link)

Inicio de Sesión

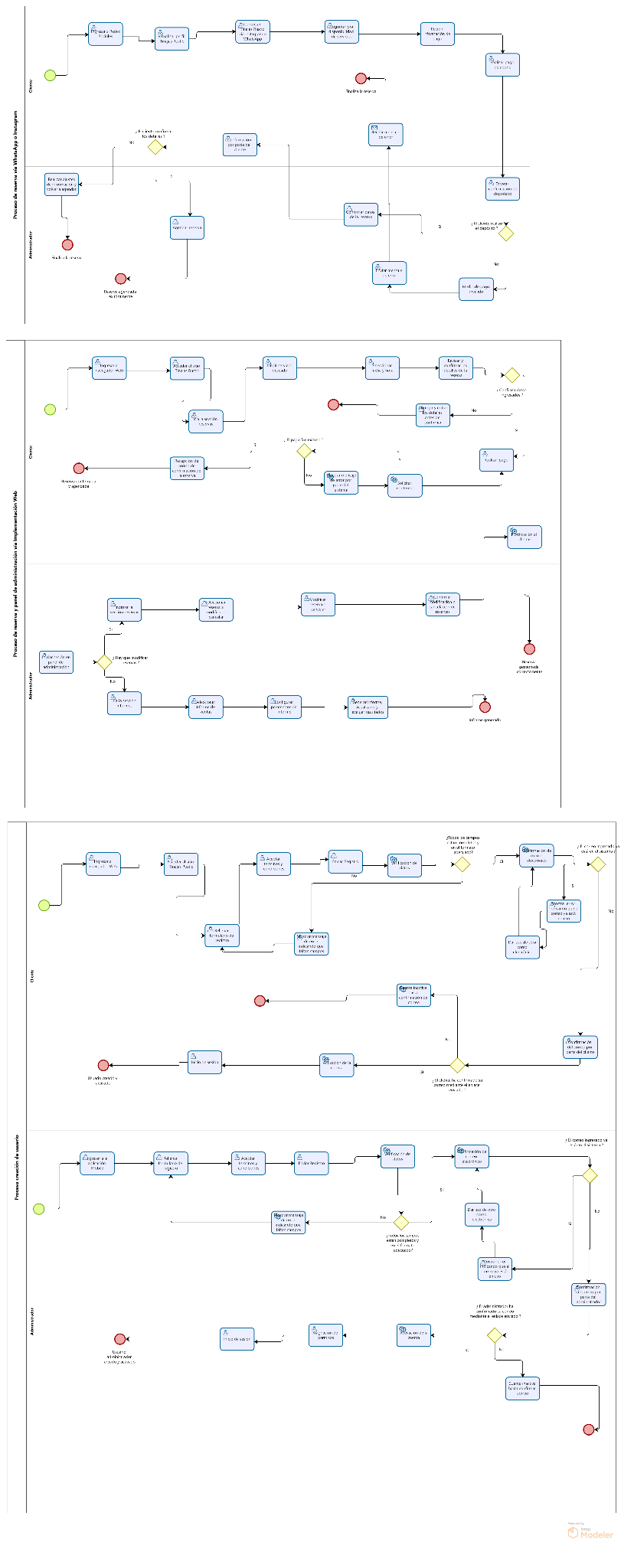


Procesamiento de Pagos

# Arquitectura de Procesos

En esta parte se muestran los procesos que hay en el sistema y la forma en la que se comunican estos procesos, es decir, se representa el flujo de trabajo paso a paso de negocio y operacionales de los componentes que conforman el sistema.

*URL:* [*Bizagi AS IS - TO BE*](https://drive.google.com/file/d/1qJ6-jV_0MBpEUi1y089anQj_6NA-9EEM/view?usp=drive_link)

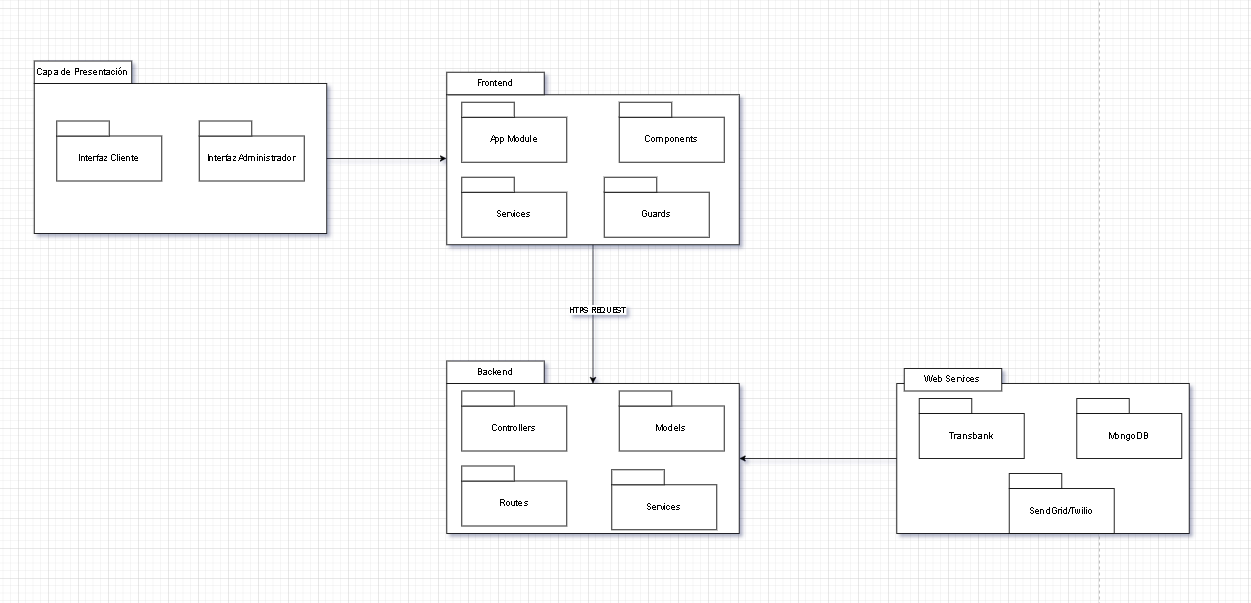


# Arquitectura de desarrollo

Representa cómo un sistema de software es dividido en componentes y demostrando las dependencias entre estos componentes. En el diagrama que se indica en esta sección nos describe los componentes utilizados para constituir las funcionalidades del sistema.

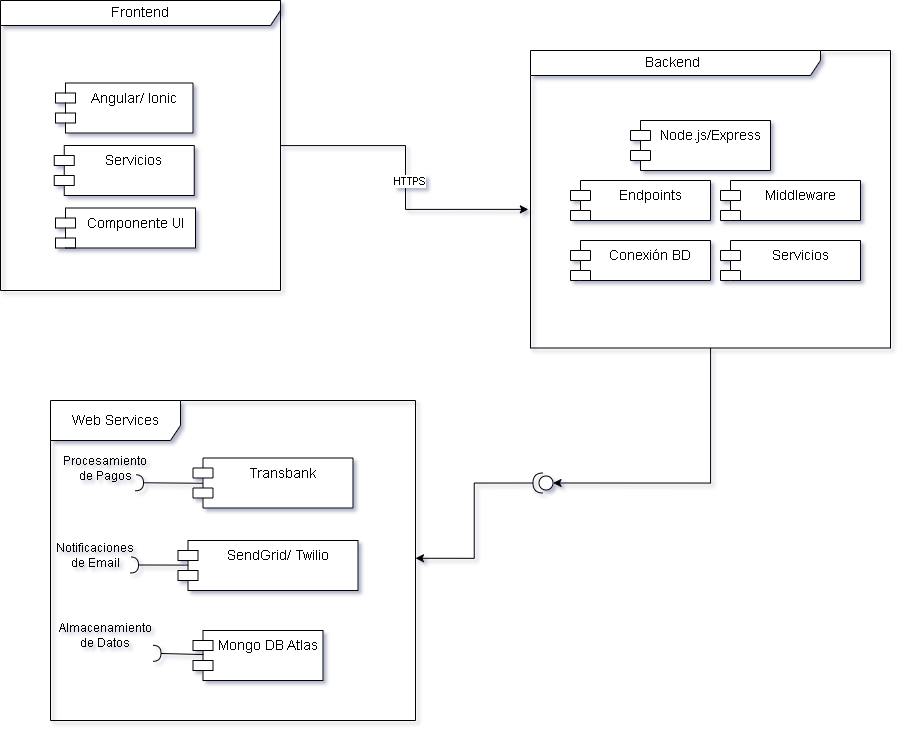
## Diagrama de paquete:

URL: [Diagrama de Paquetes](https://drive.google.com/file/d/1YxkRD9QTofWsaH5IDgNcqEkhXv6aWysy/view?usp=drive_link)

****

## Diagrama de Componentes:

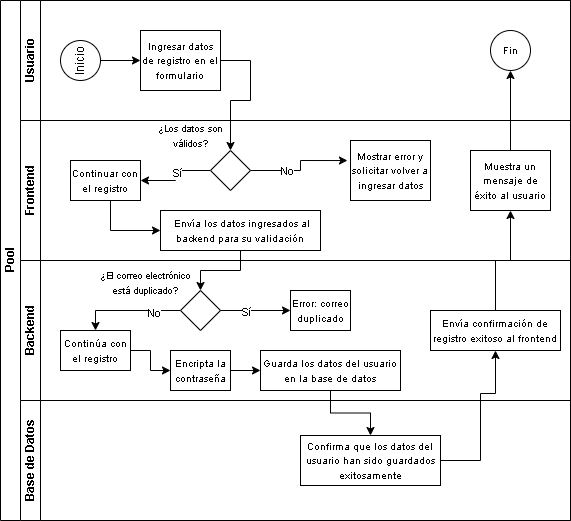
URL: [Diagrama de Componentes](https://drive.google.com/file/d/1Kln3YxlaTMP42AViutoZRGfjKq2Y2O4E/view?usp=drive_link)



## Diagrama de Actividad:

URL: [Carpeta Diagramas de Actividad](https://drive.google.com/drive/folders/1QfiVr11370LuonAXOctA8S1netyXmzML?usp=drive_link)

Registro de Usuario

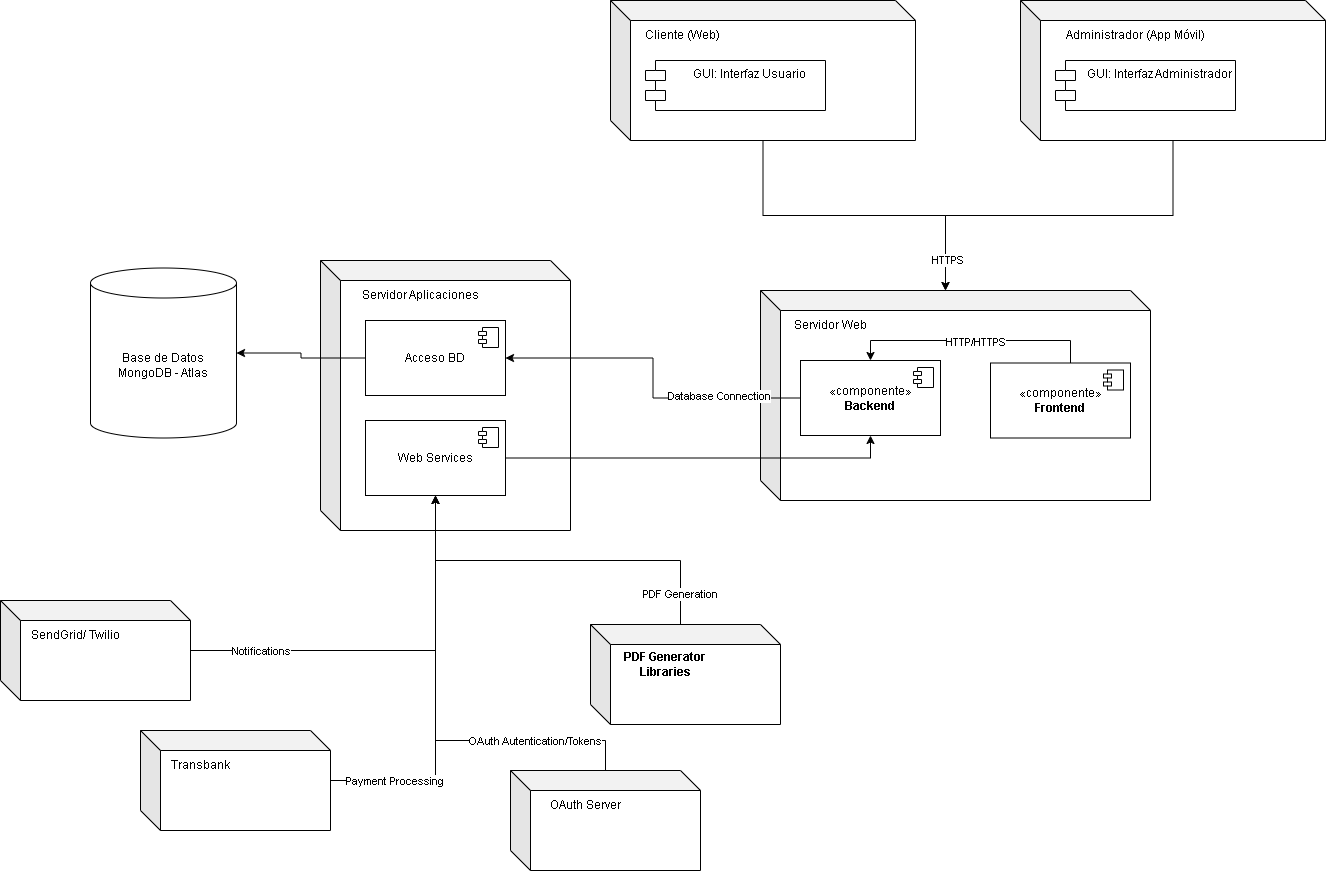


# Arquitectura física

En esta vista se exponen los componentes físicos del sistema así como las conexiones físicas entre estos componentes que conforman la solución (incluyendo los servicios). Dicho lo anterior en este diagrama se muestra como se hace un iniciar conexión y cómo se producen todas las conexiones entre los componentes (nodes).

Diagrama de Despliegue

URL:[Diagrama de Despliegue](https://drive.google.com/file/d/1r4LNAHSnVjhmaiuup_IN01yHD5QGmjQa/view?usp=drive_link)

****

# Escenarios

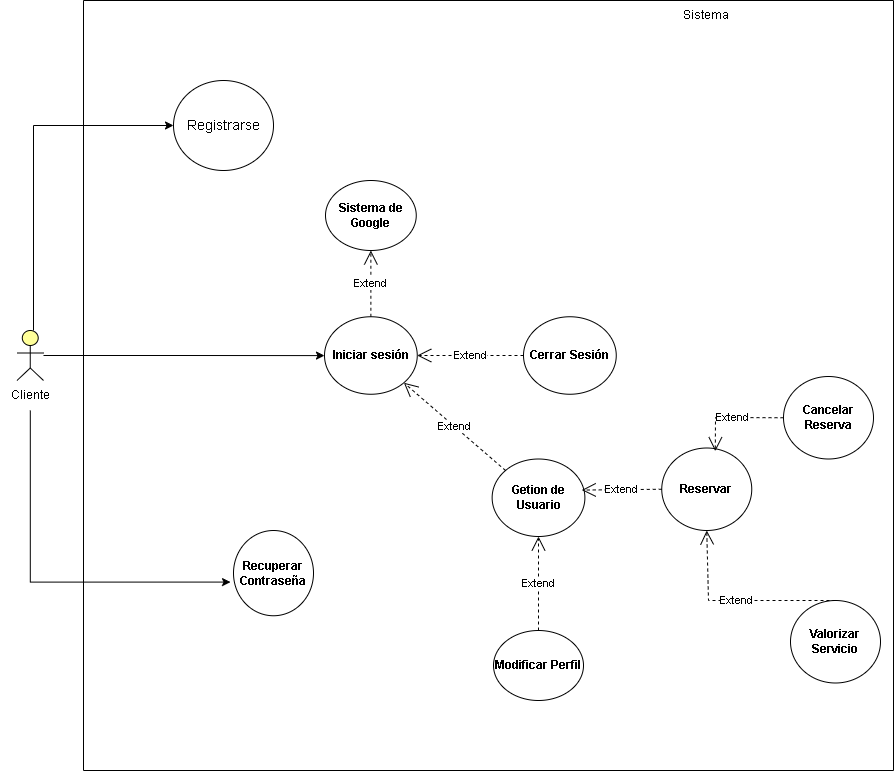
En esta sección se presenta y describe de forma detallada el conjunto de escenarios funcionales y no funcionales que obtuvieron la mayor prioridad en el análisis. Para esto se representa el diagrama de casos de uso y los casos de uso prioritarios, así como los escenarios en que uno o más atributos de calidad se ven involucrados de manera significativa e importante.

## Modelo de Casos de Uso

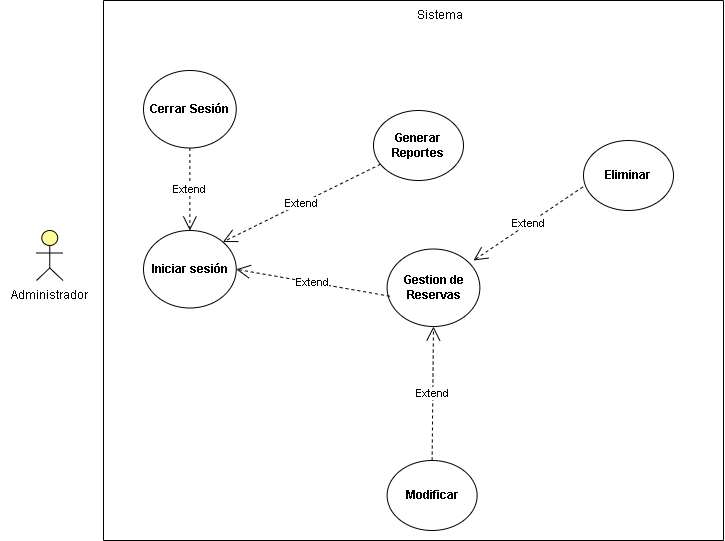
El modelo de casos de uso puede ser encontrado en la documentación denominada “Casos de Uso General”.

URL:[Carpeta Diagramas Caso de Uso](https://drive.google.com/drive/folders/1Rcf2yLKzBoJ-ULjtirJF_8DFNQrsZR2T?usp=drive_link)

### Caso de uso de usuario Tinajas Rustic

**

### Caso de uso Administrador Tinajas Rustic

****

# Especificación de Casos de Uso Relevantes

Los casos de uso más relevantes para el desarrollo de la arquitectura fueron determinados y establecidos en base a los siguientes criterios se muestran a continuación:

* Su implementación implica varios nodos de la vista de despliegue.
* Su implementación es de alto riesgo.
* Incluye muchos conceptos y relaciones del dominio.
* Incluye posibles escenarios críticos de calidad.

A continuación, se presentan los casos de uso relevantes, los cuales fueron listados y ordenados, además estos pueden ser encontrados con su especificación detallada en el documento “Casos de Uso General ”.

| **Código** | **Nombre** | **Actores** | **Prioridad** |
| --- | --- | --- | --- |
| CU-01 | Registrarse | Cliente | Muy Alta |
| CU-02 | Iniciar Sesión | Cliente | Muy Alta |
| CU-03 | Recuperar Contraseña | Cliente | Muy Alta |
| CU-08 | Gestionar Perfil | Cliente | Muy Alta |
| CU-05 | Reservar | Cliente | Muy Alta |
| CU-06 | Cancelar Reserva | Cliente | Muy Alta |
| CU-12 | Valorización del Servicio | Cliente | Muy Alta |

## Especificación de los Escenarios de Calidad Relevantes

Después de un análisis en conjunto entre los dos proyectos, se llegó a la conclusión que los escenarios que se son los más relevantes para estos proyectos son:

| Prioridad Arquitectónica | Aplicación |
| --- | --- |
| Muy alta | Web |

| Código | Nombre | Sinopsis | Entorno y  Cambio en el Entorno | | Comportamiento Esperado | Medida |
| --- | --- | --- | --- | --- | --- | --- |
| CU-01 | Registrarse | Debe permitir a los nuevos usuarios registrarse de manera efectiva y sin errores. | Sistema operativo con internet,aumentó en usuarios registrándose simultáneamente. | | Registro en menos de 5 segundos. | Menos de 5 segundos. |
| CU-02 | Iniciar Sesión | Debe permitir a los usuarios iniciar sesión con credenciales válidas. | Sistema operativo con internet,aumentó en el número de intentos de inicio de sesión simultáneos | | Autenticación en menos de 3 segundos. | Menos de 3 segundos. |
| CU-03 | Recuperar Contraseña | Debe permitir a los usuarios recuperar su contraseña de forma segura y eficiente. | Sistema operativo con internet,aumentó el número de solicitudes de recuperación de contraseña. | | Enlace de recuperación en menos de 10 minutos. | Menos de 10 minutos. |
| CU-08 | Gestionar Perfil | Debe permitir a los usuarios gestionar su perfil, incluyendo la actualización de información personal. | Sistema operativo con internet, aumentó en el número de usuarios realizando cambios simultáneamente en sus perfiles. | | Cambios en menos de 5 segundos. | Menos de 5 segundos |
| CU-05 | Reservar | Debe permitir a los usuarios realizar una reserva de manera eficiente. | Sistema operativo con internet,aumentó el número de usuarios realizando reservas simultáneamente. | | Confirmación en menos de 10 segundos. | Menos de 10 segundos. |
| CU-06 | Cancelar Reserva | Debe permitir a los usuarios cancelar reservas de manera eficiente. | Sistema operativo con internet,aumentó en el número de usuarios intentando cancelar reservas simultáneamente | | Cancelación en menos de 10 segundos. | Menos de 10 segundos. |
| CU-12 | Valorización del Servicio | Debe permitir a los usuarios valorar y dejar comentarios sobre el servicio recibido. | Sistema operativo con internet, aumentó en el número de usuarios dejando valoraciones y comentarios simultáneamente. | | Valoraciones en menos de 5 segundos. | Menos de 5 segundos. |

# Tamaño y desempeño

El tamaño del sistema desarrollado para Tinajas Rustic se encuentra enfocado en la eficiencia y la modularidad de los componentes. La arquitectura adoptada está diseñada para ser escalable, permitiendo la adición de nuevas funcionalidades sin comprometer el rendimiento del sistema. El sistema incluye múltiples módulos como la Gestión de Reservas, Procesamiento de Pagos, Notificaciones y Recordatorios, y Valoraciones, los cuales están distribuidos de forma que se pueda paralelizar su construcción y realizar pruebas unitarias sobre ellos, disminuyendo los riesgos de fallos y facilitando el mantenimiento.

Cada uno de estos módulos ha sido diseñado de manera desacoplada, siguiendo un enfoque modular que permite el desarrollo independiente de los mismos, lo cual ayuda a reducir los tiempos de implementación. Además, se ha priorizado una estructura ligera en los componentes móviles para asegurar que el sistema pueda ser utilizado sin problemas en una amplia gama de dispositivos.

Se ha tenido en cuenta que el sistema debe ser capaz de gestionar un gran volumen de usuarios simultáneamente, especialmente en las áreas críticas como la gestión de reservas y el procesamiento de pagos. Para garantizar un alto nivel de desempeño, se han considerado los siguientes puntos:

* Optimización de consultas a la base de datos: Se utilizarán índices eficientes y optimizaciones de consultas SQL/MongoDB para minimizar los tiempos de respuesta en las operaciones de lectura y escritura.
* Gestión eficiente de la concurrencia: El sistema estará diseñado para soportar múltiples usuarios simultáneamente. Esto implica que los procesos de validación, registro de usuarios, reservas y procesamiento de pagos podrán ejecutarse de manera concurrente sin que se produzcan bloqueos o degradación del servicio.
* Bajo tiempo de respuesta: El sistema está diseñado para asegurar tiempos de respuesta óptimos en cada interacción del usuario, con un enfoque particular en el rendimiento del frontend y la minimización de latencias en las comunicaciones entre el backend y la base de datos. Se espera que operaciones críticas, como el registro de usuario o la confirmación de una reserva, se completan en menos de 5 segundos.
* Manejo de grandes volúmenes de datos: El sistema será capaz de manejar un número considerable de registros de usuarios, reservas y transacciones sin que ello afecte su rendimiento, gracias a la infraestructura y a los servicios escalables empleados.
* Uso de servicios externos: Para mantener una alta disponibilidad y eficiencia, algunos de los servicios, como los de notificaciones y procesamiento de pagos, se delegarán a proveedores externos como SendGrid, Twilio, y Transbank, lo que permitirá disminuir la carga de procesamiento interno.