**UNIVERSIDAD NACIONAL MAYOR DE SAN MARCOS**

**(Universidad del Perú, DECANA DE AMÉRICA)**

**FACULTAD DE INGENIERÍA DE SISTEMAS E INFORMÁTICA**

**ESCUELA PROFESIONAL DE INGENIERÍA DE SOFTWARE**



**“Documento de Arquitectura”**

**Curso: Gestión de la Configuración**

**Presentan los estudiantes:**

Condor Marin Jesus Ernesto

Tome Pando Carlos Alexander

Rique Garcia Marko Jhunior

Hidalgo Cock Joaquin

Yataco Quispe Jorge Armando

Neira Carquin Rogger Kevin

Vasquez Gonzales Pedro Sebastian

**Docente:**

Wong

**LIMA – PERÚ**

**2023-II**

**ÍNDICE**

[**1. Descripción general 3**](#_heading=h.30j0zll)

[**2. Drivers de la Arquitectura 3**](#_heading=h.1fob9te)

[a. Requerimientos funcionales: 3](#_heading=h.3znysh7)

[b. Requerimientos no funcionales: 5](#_heading=h.3dy6vkm)

[c. Restricciones 6](#_heading=h.2s8eyo1)

[d. Preocupaciones / Riesgos. 6](#_heading=h.17dp8vu)

[**3. Contexto del sistema 7**](#_heading=h.3rdcrjn)

[**4. Arquitectura general 7**](#_heading=h.26in1rg)

[**5. Vistas del sistema (en desarrollo) 9**](#_heading=h.lnxbz9)

[**6. Detalles de Implementación 9**](#_heading=h.35nkun2)

[1. Tecnologías y Herramientas: 9](#_heading=h.1ksv4uv)

[2. Arquitectura de software: 11](#_heading=h.44sinio)

[5. Consideraciones de Seguridad: 12](#_heading=h.2jxsxqh)

[6. Pruebas y Depuración: 12](#_heading=h.z337ya)

[7. Escalabilidad y Rendimiento: 12](#_heading=h.3j2qqm3)

[8. Control de Versiones y Gestión de Código Fuente: 13](#_heading=h.1y810tw)

[9. Despliegue: 13](#_heading=h.4i7ojhp)

[10. Análisis de escenarios de atributos 13](#_heading=h.2xcytpi)

# **Descripción general**

La aplicación web Gymbob es una aplicación web orientada a la gestión de un gimnasio y sus clientes. Su utilización requiere de la creación de una cuenta por parte de los clientes, esto les permitirá tener completo acceso a nuestro servicios así como personalizar su propia experiencia mientras utilizan la aplicación.

El proyecto fue creado para satisfacer el sector de los gimnasios, permitiendo a los usuarios reservar sus máquinas en las “horas punta” para poder completar sus rutinas, así como la creación de una dieta de manera virtual con la finalidad de que los usuarios puedan regular su alimentación para obtener los mejores resultados y puedan cumplir su cometido.

# **Drivers de la Arquitectura**

Los drivers del sistema para este módulo se dividen en 4 sub puntos clave:

## **Requerimientos funcionales:** Los requisitos funcionales dentro de este módulo se centran en las operaciones fundamentales relacionadas con el registro de clientes, gestión en línea, etc.

### **Especificación de los requerimientos funcionales**

| **CÓDIGO** | **DESCRIPCIÓN** | **ACTORES** | **PRIORIDAD** | |
| --- | --- | --- | --- | --- |
| **REQ-FUN001** | **EL SISTEMA PRESENTARÁ UN REGISTRO DE USUARIO**  El sistema debe permitir que el usuario pueda registrarse en la página proporcionando algunos datos esenciales, tales como:   * Nombre * Apellidos * Correo * Contraseña * DNI | Usuario | X | Alta |
|  | Media |
|  | Baja |
| **REQ-FUN002** | **EL SISTEMA PERMITE QUE EL USUARIO INGRESE A SU CUENTA Y CERRAR SESIÓN**  El sistema debe permitir que el usuario inicie sesión en la página mediante un correo electrónico y contraseñas ya registradas. Así como permitir que el usuario cierre sesión en la página mediante un botón salir | Usuario | X | Alta |
|  | Media |
|  | Baja |
| **REQ-FUN003** | **EL SISTEMA DEBE PERMITIR LA BÚSQUEDA DEL LA MÁQUINA**  El sistema debe permitir que el usuario realice una búsqueda de las máquinas que desea usar. Las máquinas presentadas deben presentarse de modo ordenado, de modo que permita al usuario una identificación y diferenciación rápida. | Usuario |  | Alta |
| x | Media |
|  | Baja |
| **REQ-FUN004** | **EL SISTEMA PERMITIRÁ QUE EL USUARIO SELECCIÓN LA MÁQUINA Y MOSTRARÁ INFORMACIÓN DE ESTE**  Tras la búsqueda el usuario podrá escoger la máquina deseada, permitiendo que el usuario pueda visualizar el nombre y otras características del producto. | Usuario | X | Alta |
|  | Media |
|  | Baja |
| **REQ-FUN005** | **EL SISTEMA PERMITIRÁ GESTIONAR LAS MÁQUINAS ELEGIDAS EN EL “GESTOR DE CESTA”**  El sistema debe tener una opción para que el usuario pueda eliminar máquinas del gestor de cesta. El gestor de cesta debe resumir todos los productos seleccionados. | Usuario | X | Alta |
|  | Media |
|  | Baja |
| **REQ-FUN006** | **EL SISTEMA PERMITIRÁ CANJEAR PREMIOS**  El sistema otorgará la opción al usuario de canjear los puntos acumulados, obtenidos por la cantidad de días que asistió al gimnasio, por premios como: descuentos, tarjetas de regalo en tiendas o suplementos vitamínicos. | Usuario | X | Alta |
|  | Media |
|  | Baja |
| **REQ-FUN007** | **EL SISTEMA PERMITIRÁ REALIZAR PAGO POR MEMBRESÍA**  El usuario podrá realizar el pago de la membresía, previo llenado de datos de la tarjeta e ingreso de usuario.  Entre los datos que se pedirán tenemos:   * Nombre de la tarjeta * Número de tarjeta * Fecha de vencimiento   Código Valor de Validación o Verificación (CVV) | Usuario |  | Alta |
| X | Media |
|  | Baja |
| **REQ-FUN008** | **EL SISTEMA OTORGA LA OPCIÓN DE ADMINISTRAR EL PERFIL DE USUARIO**  El usuario podrá observar los datos de su perfil, su nombre, apellido, correo, contraseña así como también DNI. Además, podrá tener la opción de cambiar solo la contraseña ingresada. | Usuario | X | Alta |
|  | Media |
|  | Baja |

## **Requerimientos no funcionales:** Los requisitos no funcionales de este módulo se centran en cómo el sistema debe desempeñarse en temas de calidad, rendimiento, seguridad entre otras características.

### **Especificación de los requerimientos no funcionales**

| **CÓDIGO** | **DESCRIPCIÓN** | **CLASIFICACIÓN** | **PRIORIDAD** | |
| --- | --- | --- | --- | --- |
| **REQ-NFU001** | **PROPORCIONAR SEGURIDAD AL ACCESO**  Se debe garantizar la seguridad del sistema mediante la autenticación de usuarios. Cada usuario deberá proporcionar un código de usuario y una contraseña asignados previamente en la página principal. El sistema validará esta información y otorgará acceso a la cuenta correspondiente. El acceso concedido estará restringido únicamente a las opciones y funcionalidades que se ajusten al perfil definido para cada usuario | SEGURIDAD | X | Alta |
|  | Media |
|  | Baja |
| **REQ-NFU002** | **SEGURIDAD DEL USUARIO**  La aplicación debe garantizar la seguridad de los usuarios, protegiendo de manera adecuada los datos personales y financieros que se manejan. Esto incluye la implementación de protocolos de seguridad robustos y prácticas de gestión de acceso adecuadas. Al asegurar la confidencialidad, integridad y disponibilidad de los datos, la aplicación generará confianza en los clientes y fortalecerá la relación de confidencialidad y seguridad con ellos. | SEGURIDAD | X | Alta |
|  | Media |
|  | Baja |
| **REQ-NFU003** | **FACILIDAD DE USO DEL SISTEMA**  El sistema debe poseer una interfaz gráfica estandarizada, esta incluirá pantallas, menús y opciones, catálogo de búsqueda, tamaño de las pantallas, color, tipo de letra y configuración de los campos de entrada.  El diseño debe realizarse guiado por las características generales. En cuanto a colores y disposición de contenidos son los encontrados en el sitio web de la organización. | DISEÑAR INTERFAZ DE USUARIO AMIGABLES | X | Alta |
|  | Media |
|  | Baja |
| **REQ-NFU004** | **DISPONIBILIDAD DEL SISTEMA**  El sistema debe estar disponible de manera continua, sin interrupciones o caídas en el servicio. Para lograr esto, se implementarán medidas de monitoreo y mantenimiento preventivo. Estas acciones garantizarán que el sistema se encuentre en pleno funcionamiento y disponible en todo momento para los usuarios. Asimismo, se llevará a cabo un seguimiento constante para detectar cualquier anomalía y tomar las medidas necesarias de manera proactiva, asegurando así una alta disponibilidad del sistema | DISPONIBILIDAD | X | Alta |
|  | Media |
|  | Baja |
| **REQ-NFU005** | **RENDIMIENTO DEL SISTEMA**  La aplicación debe ofrecer un rendimiento óptimo con tiempos de carga rápidos. Esto garantizará que los usuarios puedan utilizar la página de manera ágil y fluida, brindándoles comodidad en su experiencia de uso. Para lograrlo, se realizan ajustes y optimizaciones en el programa, asegurando que los tiempos de carga sean mínimos y la respuesta del sistema sea rápida y eficiente. La mejora en el rendimiento del programa contribuirá a una interacción más satisfactoria con la aplicación, mejorando la experiencia global del usuario. | TIEMPO DE RESPUESTA | X | Alta |
|  | Media |
|  | Baja |
| **REQ-NFU006** | **ESCALABILIDAD DEL SISTEMA**  La aplicación debe ser capaz de manejar de manera eficiente un número considerable de usuarios y procesos simultáneos. Esto implica una capacidad de escalabilidad que permita adaptarse y responder adecuadamente a un aumento en la demanda del sistema. Para lograrlo, se implementarán medidas y arquitecturas escalables que permitan distribuir la carga de trabajo y garantizar un rendimiento óptimo en todo momento. La escalabilidad del sistema asegurará su capacidad para soportar un crecimiento sostenido de usuarios y procesos, manteniendo la calidad y la velocidad de respuesta esperadas | TIEMPO DE RESPUESTA | X | Alta |
|  | Media |
|  | Baja |
| **REQ-NFU007** | **MANTENIBILIDAD DEL SISTEMA**  La aplicación debe ser diseñada y desarrollada de manera que sea fácil de mantener, actualizar y modificar. Esto permitirá una rápida respuesta a las necesidades cambiantes de la empresa y los usuarios. Para lograrlo, se seguirán prácticas de programación y diseño de software que fomenten la modularidad, la claridad del código y la documentación adecuada. La mantenibilidad del sistema garantizará una gestión eficiente y ágil, lo que resultará en una mejora continua y una mayor capacidad de adaptación. | MANTENIMIENTO | X | Alta |
|  | Media |
|  | Baja |

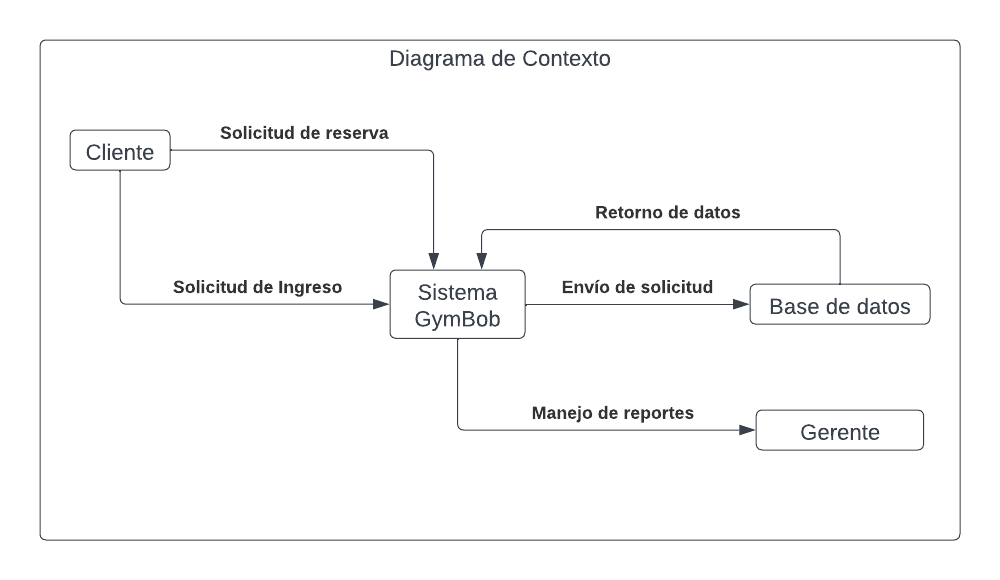
## **Restricciones**

* + 1. **Versiones Compatibles.** Asegurarse de que todas las versiones de NodeJS, React y MySQL utilizadas en el proyecto sean compatibles entre sí y se mantengan actualizadas para evitar problemas de incompatibilidad.
    2. **Rendimiento de ReactJS:** Optimizar el rendimiento de las interfaces de usuario desarrolladas en ReactJS para garantizar una experiencia de usuario fluida.
    3. **Seguridad en Node.js:** Implementar buenas prácticas de seguridad en Node.js, como la protección contra ataques de inyección de SQL y la gestión adecuada de sesiones y autenticación de usuarios.
    4. **Seguridad de MySQL:** Configurar y mantener la base de datos en MySQL de manera segura, aplicando prácticas recomendadas de seguridad de bases de datos, como la limitación de permisos.
    5. **Resiliencia y Tolerancia a Fallos:** Implementar estrategias de resiliencia y tolerancia a fallos en la aplicación para mitigar el impacto de posibles interrupciones en los servicios de NodeJS y MySQL.

## **Preocupaciones / Riesgos.** A continuación, explicaremos algunos de los riesgos y preocupaciones se deben tener en cuenta al gestionar el sistema

* + 1. **Falta de integración.** Si los sistemas utilizados para gestionar la atención no están bien integrados entre sí, puede haber problemas de inconsistencia en los datos del cliente.
    2. **Problemas de redundancia de datos.** La redundancia excesiva de datos o la falta de un sistema centralizado pueden dificultar la gestión eficiente de la información del cliente.
    3. **Problemas de escalabilidad.** El sistema debe adaptarse a un aumento en la carga de trabajo, como durante picos de demanda, para mantener la calidad del servicio.
    4. **Falla en la seguridad de datos.** Las brechas de seguridad o la falta de medidas adecuadas pueden exponer información confidencial, lo que podría resultar en problemas legales.
    5. **Accesos no autorizados.** Existe el riesgo de que personas no autorizadas accedan a la información del cliente si los controles de acceso no son adecuados.
    6. **Dependencias o versiones antiguas.** Si el sistema depende de tecnologías o versiones de software obsoletas, podría haber vulnerabilidades de seguridad no parcheadas, lo que aumentaría el riesgo de ataques cibernéticos y la exposición de datos sensibles.
    7. **Deficiencia en la usabilidad.** La falta de un diseño de interfaz de usuario intuitivo y amigable puede llevar a una usabilidad deficiente. Esto puede resultar en errores por parte de los agentes al ingresar datos o gestionar líneas de servicio.

# **Contexto del sistema**

El sistema GymBob permite a los usuarios realizar todas sus actividades a través de nuestro portal web, dichas actividades se mandan como una solicitud a la base de datos la cual valida los datos y envía una repuestos de regreso al sistema, todas las operaciones quedarán registradas como datos y podrán ser visualizadas por el gerente en caso de quejas o reclamos.

# **Arquitectura general**

La arquitectura del sistema GymBob está diseñada para proporcionar a los usuarios una experiencia de usuario excepcional, permitiendo un acceso eficiente y seguro a las funcionalidades del sistema.

Se ha optado por la arquitectura Monolítica por varias razones clave:

1. **Monolito único:** En una arquitectura monolítica, todo el código y la lógica de la aplicación se agrupan en una sola base de código y se despliegan como una única aplicación.
2. **Interconexión directa:** Los módulos y componentes de la aplicación se comunican entre sí de manera directa, a menudo a través de llamadas de función o invocación de métodos.
3. **Despliegue integral:** Cuando se realiza una actualización o corrección, se debe implementar toda la aplicación. Esto significa que se detiene y reemplaza toda la aplicación en el servidor cuando se realiza un cambio.
4. **Escalabilidad vertical:** Para manejar un mayor volumen de usuarios o carga, generalmente se escala verticalmente la infraestructura, lo que implica aumentar la potencia de procesamiento y los recursos de la máquina en la que se ejecuta la aplicación.
5. **Fácil desarrollo inicial:** Para aplicaciones pequeñas o simples, una arquitectura monolítica puede ser más rápida y fácil de desarrollar y desplegar en comparación con arquitecturas más complejas.

La elección de la arquitectura Monolítica se basa en que es adecuada para aplicaciones pequeñas o simples, pero puede presentar desafíos a medida que la aplicación crece en tamaño y complejidad.

# **Detalles de Implementación**

## **1. Tecnologías y Herramientas:**

**Tecnologías utilizadas:**

1. **Javascript:** Herramienta accesible y compatible con distintos frameworks para el desarrollo web, también cuenta con una gran versatilidad para ejecutarse en múltiples navegadores.
2. **ReactJS**: Biblioteca muy utilizada para construir interfaces de usuario interactivas y dinámicas en aplicaciones web.
3. **MySQL:** Rendimiento sólido, accesible para todos y con gran popularidad lo cual conlleva a una gran cantidad de recursos disponibles para su utilización.
4. **NodeJS:** Permite manejar muchas conexiones simultáneas de manera eficaz, esto resulta particularmente útil para aplicaciones en tiempo real y de alta concurrencia.
5. **TestLink:** Herramienta de gestión de pruebas de software de fácil uso y amigable con el usuario, permite tener un control para organizar, planificar y ejecutar pruebas.

## **2. Arquitectura de software**

* **Registrar usuario:** Permite a los empleados de la empresa registrar un nuevo cliente a través de la plataforma web. Algunos de los datos que se toman son: DNI, nombre, apellido completo, domicilio, correo electrónico, etc. Así como también asociar a un familiar o conocido del cliente para darle una línea de la cuál se harán responsable el titular.
* **Logear y Cerrar Sesión:** Permite a los usuarios previamente registrados acceder al sistema o salir del mismo, esto con la finalidad de obtener una experiencia personalizada.
* **Buscar y seleccionar máquinas:** Proporciona a los usuarios la capacidad de ver las máquinas disponibles, y permite reservar un horario para su próxima utilización.
* **Planear alimentación:** Esta función permitiría a los clientes solicitar un plan alimenticio según sus necesidades para complementar sus rutinas de ejercicios.
* **Gestionar cesta:** Permite a los usuarios gestionar sus compras: añadir, modificar cantidad o eliminar de la cesta aquellos productos solicitados.
* **Canjear premios:** Los usuarios que cumplan con su cometido recibirán vales de descuento que podrán digitar en el carrito para obtener un descuento final en el precio total.
* **Realizar pago:** El usuario podrá seleccionar su medio de pago y realizar la acción correspondiente.

## **5. Consideraciones de Seguridad:**

La base de datos estará resguardada con los permisos correspondientes.

## **6. Pruebas y Depuración:**

Se utilizará el gestor Test Link, con la finalidad de mantener el sistema con estándares de calidad, estas pruebas y sus fechas se encuentran detallados en el cronograma previamente entregado.

## **7. Escalabilidad y Rendimiento:**

El proyecto será realizado mediante diferentes módulos, esto permitirá que se agreguen futuras actualizaciones y funcionalidades al sistemas. Así mismo esto resultará más sencillo de manejar para el mantenimiento de las mismas.

Se aplicará aplicar las normas ISO 9001 para que el sistema funcione correctamente y de manera fluida, permitiendo a los usuarios su fácil y los programadores un sencillo manejo del código para su entendimiento con fines de mejora u optimización.

## **8. Control de Versiones y Gestión de Código Fuente:**

El desarrollo del sisetma estará desplegado en un repositorio de github, esto se hará con la finalidad de tener un seguimiento y control del avance del proyecto el cual buscará cumplir con el cronograma de entrega por módulos, el enlace del repositorio se anexará a continuación: https://github.com/JesusCondor16/ProyectoGimnasio.git

## **9. Despliegue:**

Para el correcto despliegue del aplicativo se utilizarán los siguientes:

* **ReactJS:** Se basa en un enfoque de desarrollo por componentes, esto permitirá dividir la interfaz del usuario y desarrollar cada componente de forma independiente. Además de contar con muchas librerías las cuales facilitan el desarrollo de aplicaciones completas y ofrecer un alto rendimiento para desarrollo web.
* **NodeJS:** Permite manejar aplicaciones que manejan un gran número de conexiones simultáneas, debido a su facilidad de uso y disponibilidad de numerosos módulos es muy frecuente elegida para realizar MVP. También es multiplataforma y facilita la portabilidad entre diferentes sistemas operativos.

## **10. Análisis de escenarios de atributos**

1. Atributos de calidad y Escenarios
   1. Rendimiento:
      1. Escenario: El cliente solicita datos de la base de datos en tiempo real.
      2. Requisito de Rendimiento: El Sistema debe proporcionar una respuesta casi instantánea al realizar búsquedas de clientes.
   2. Disponibilidad.
      1. Escenario: Los representantes de atención al cliente dependen del módulo de clientes para ayudar a los clientes en cualquier momento.
      2. Requisito de Disponibilidad: El sistema debe estar disponible las 24 horas del día, los 7 días de la semana, con un tiempo de inactividad mínimo planificado para mantenimiento.
   3. Modificabilidad
      1. Escenario: La empresa necesita agregar un nuevo campo personalizado para registrar información adicional de los clientes.
      2. Requisito de Modificabilidad: El sistema debe ser fácil de modificar para agregar nuevos campos o personalizar la información que se almacena sin afectar la funcionalidad existente.
   4. Seguridad
      1. Escenario: El sistema almacena información sensible sobre los clientes, como números de tarjetas de crédito.
      2. Requisito de Seguridad: El sistema debe implementar medidas de seguridad robustas, como cifrado de datos y control de acceso, para proteger la información confidencial de los clientes.
   5. Facilidad para Pruebas:
      1. Escenario: El equipo de desarrollo necesita realizar pruebas exhaustivas en el módulo de clientes.
      2. Requisito de Facilidad para Pruebas: El sistema debe ser diseñado de manera que las pruebas unitarias y de integración puedan realizarse de manera eficiente, y los resultados de las pruebas deben ser fácilmente interpretados por el equipo de pruebas.
   6. Usabilidad:
      1. Escenario: La interfaz del sistema debe ser entendible para el rápido aprendizaje de los clientes.
      2. Requisito de Usabilidad: La interfaz de usuario debe ser intuitiva y amigable, con una curva de aprendizaje mínima.
   7. Mantenimiento
      1. Escenario: La empresa necesita aplicar actualizaciones periódicas al sistema para cumplir con los cambios en las regulaciones de protección de datos.
      2. Requisito de Mantenimiento: El sistema debe ser fácil de mantener y actualizar para incorporar cambios regulatorios y correcciones de errores sin interrupciones significativas.
   8. Escabilidad
      1. Escenario: La empresa planea expandirse y espera un aumento en el número de clientes.
      2. Requisito de Escalabilidad: El sistema debe poder manejar un crecimiento significativo en la cantidad de datos y usuarios sin pérdida de rendimiento.
   9. Interoperabilidad:
      1. Escenario: El sistema necesita comunicarse eficazmente con la base de datos.
      2. Requisito de Interoperabilidad: El sistema debe seguir estándares de comunicación y formatos de datos establecidos para asegurar una integración sin problemas.