**Documento de Especificación de Arquitectura**

Realizado por: **Daniel Santiago Becerra y Jean Carlo Guzmán**

***HISTORIAL DE REVISIONES***

| **Fecha** | **Versión** | **Autor** | **Descripción** | **Revisado Por** |
| --- | --- | --- | --- | --- |
| <> | <> | <> | <Escriba aquí> | <Escriba aquí> |

**Contenido**

[1.](#_heading=h.3znysh7) Documento de Arquitectura de Software 4

[1.1.](#_heading=h.2et92p0) Introducción 4

[1.2.](#_heading=h.3dy6vkm) Propósito 4

[1.3.](#_heading=h.1t3h5sf) Alcance 4

[1.4.](#_heading=h.4d34og8) Referencias 4

[1.5.](#_heading=h.2s8eyo1) Definiciones acrónimos y abreviaciones 5

[2.](#_heading=h.17dp8vu) Generalidades del Proyecto 5

[2.1.](#_heading=h.3rdcrjn) Problema a Resolver 5

[2.2.](#_heading=h.26in1rg) Descripción General del Sistema a Desarrollar 5

[2.3.](#_heading=h.lnxbz9) Identificación de los Stakeholders y sus responsabilidades 5

[3.](#_heading=h.35nkun2) Vistas de la arquitectura 6

[3.1.](#_heading=h.1ksv4uv) Vista de Casos de Uso 6

[3.2.](#_heading=h.44sinio) Vista de Procesos 6

[3.3.](#_heading=h.2jxsxqh) Vista Lógica 6

[3.4.](#_heading=h.z337ya) Vista de Implementación 6

[3.5.](#_heading=h.3j2qqm3) Vista de Despliegue 6

[4.](#_heading=h.1y810tw) Arquitectura en capas 6

[5.](#_heading=h.4i7ojhp) Vista de Datos 6

[5.1.](#_heading=h.2xcytpi) Modelo Relacional 6

[6.](#_heading=h.1ci93xb) Definición de Interfaces de Usuario 6

[7.](#_heading=h.3whwml4) Características Generales de Calidad 6

[8.](#_heading=h.2bn6wsx) Stack Tecnológico 6

# Documento de Arquitectura de Software

## Introducción

En este documento trataremos temas referentes a la gestión y desarrollo de un proyecto, para ser más exactos, hablaremos del alcance, problema a resolver, como lo implementaremos y nuestra visión a futuro con este proyecto.

## Propósito

El propósito de este proyecto es dar una visión general del cómo funcionará y el cómo lo implementaremos, queremos dejar en evidencia nuestros procesos a seguir para finalizar el proyecto.

## Alcance

Iniciaremos con el problema a resolver, a quien ayudaremos y a quien le interese, es decir, planteamos el problema y buscamos la forma de resolverlo.

## Referencias

1. Documento de Especificación de Requerimientos no funcionales.
2. Documento de Visión del Proyecto.
3. Plan de Proyecto del Sistema

## Definiciones acrónimos y abreviaciones

**ARQUITECTURA DE SOFTWARE:** conjunto de elementos estáticos, propios del diseño intelectual del sistema, que definen y dan forma tanto al código fuente, como al comportamiento del software en tiempo de ejecución. Naturalmente este diseño arquitectónico ha de ajustarse a las necesidades y requisitos del proyecto.

**DESCRIPCIÓN DE ARQUITECTURA**: colección de productos de documentación.

**VISTAS:** es una representación de un área de interés o perspectiva del sistema en alto nivel.

**TIPOS DE VISTAS:** especificación de una convención de cómo construir y usar una vista. Deben satisfacer la capacidad de creación y análisis de una vista.

**STAKEHOLDER:** Individuo, equipo u organización con intereses relativos al sistema.

# Generalidades del Proyecto

## Problema a Resolver

Actualmente el gran problema que tienen los freelancers, son las dificultades para encontrar oportunidades laborales relevantes y confiables, mientras que los clientes pueden tener dificultades para encontrar profesionales calificados y verificados para sus proyectos.

## Descripción General del Sistema a Desarrollar

El sistema a desarrollar será una plataforma web dedicada a la intermediación entre freelancers y clientes. Permitirá a los usuarios registrarse como freelancers o clientes, creando perfiles detallados que muestren sus habilidades, experiencia y requisitos del proyecto. Los clientes podrán publicar proyectos específicos, mientras que los freelancers podrán buscar y postularse a estos proyectos según sus habilidades y disponibilidad.

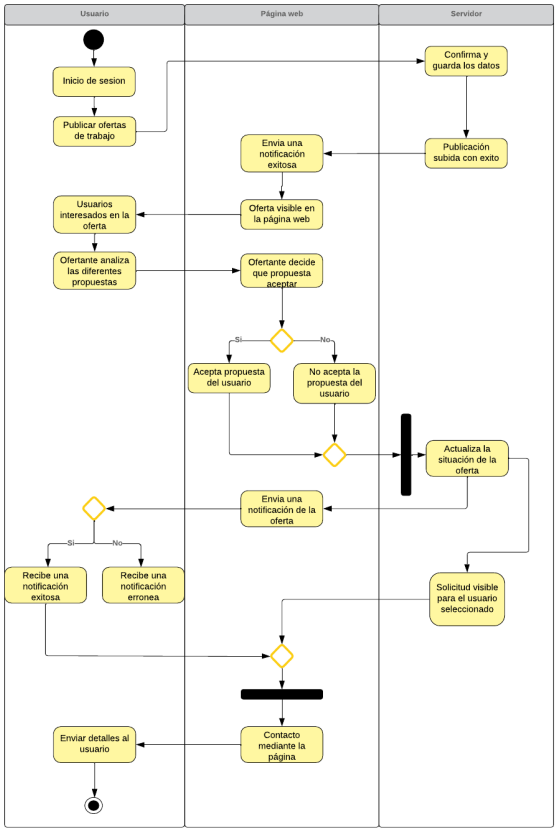
## Identificación de los Stakeholders y sus responsabilidades

| **STAKEHOLDER** | **DESCRIPCIÓN** | **ESCENARIO** | **Caso de Uso** |
| --- | --- | --- | --- |
| Administrador | Responsable de mantener y gestionar la plataforma, asegurando su funcionamiento adecuado y la satisfacción de los usuarios. | Un usuario informa sobre un comportamiento inapropiado o un problema técnico en la plataforma. | El administrador accede al perfil del usuario, revisa su actividad y toma las medidas necesarias, que pueden incluir la suspensión o eliminación de la cuenta del usuario. |
| Cliente | El cliente será aquel usuario que busca contratar servicios profesionales a través de la plataforma. | Un cliente necesita contratar a un diseñador gráfico para un proyecto de diseño de logotipo. | El cliente publica un proyecto detallando sus requisitos, revisa las propuestas recibidas, evalúa los perfiles de los freelancers y selecciona al candidato más adecuado para su proyecto. |
| Freelancer | El freelancer será aquel usuario que ofrece sus servicios profesionales a través de la plataforma. | Un freelancer está interesado en conseguir un proyecto de desarrollo web. | Búsqueda en la plataforma de proyectos, si encuentra un proyecto que se ajuste a sus habilidades y experiencia y contacta con el cliente |

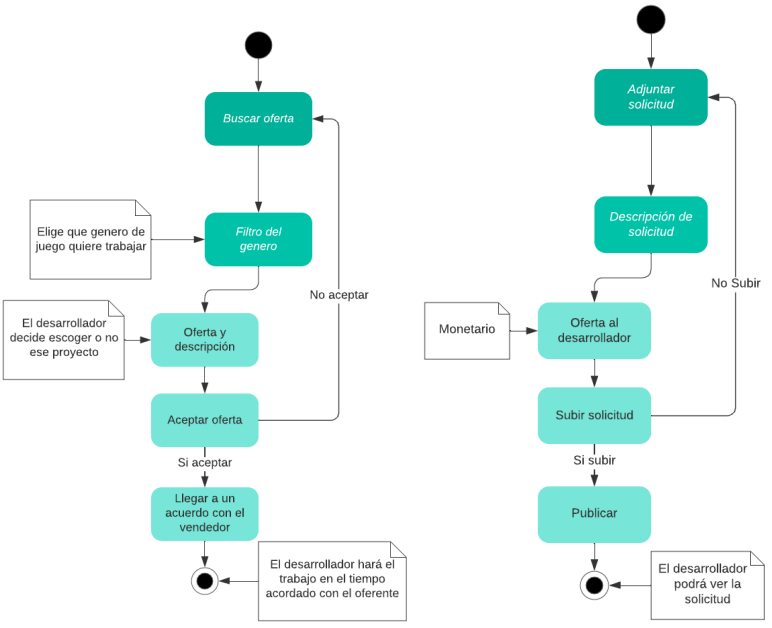
# Vistas de la arquitectura

## Vista de Casos de Uso

## Vista de Procesos

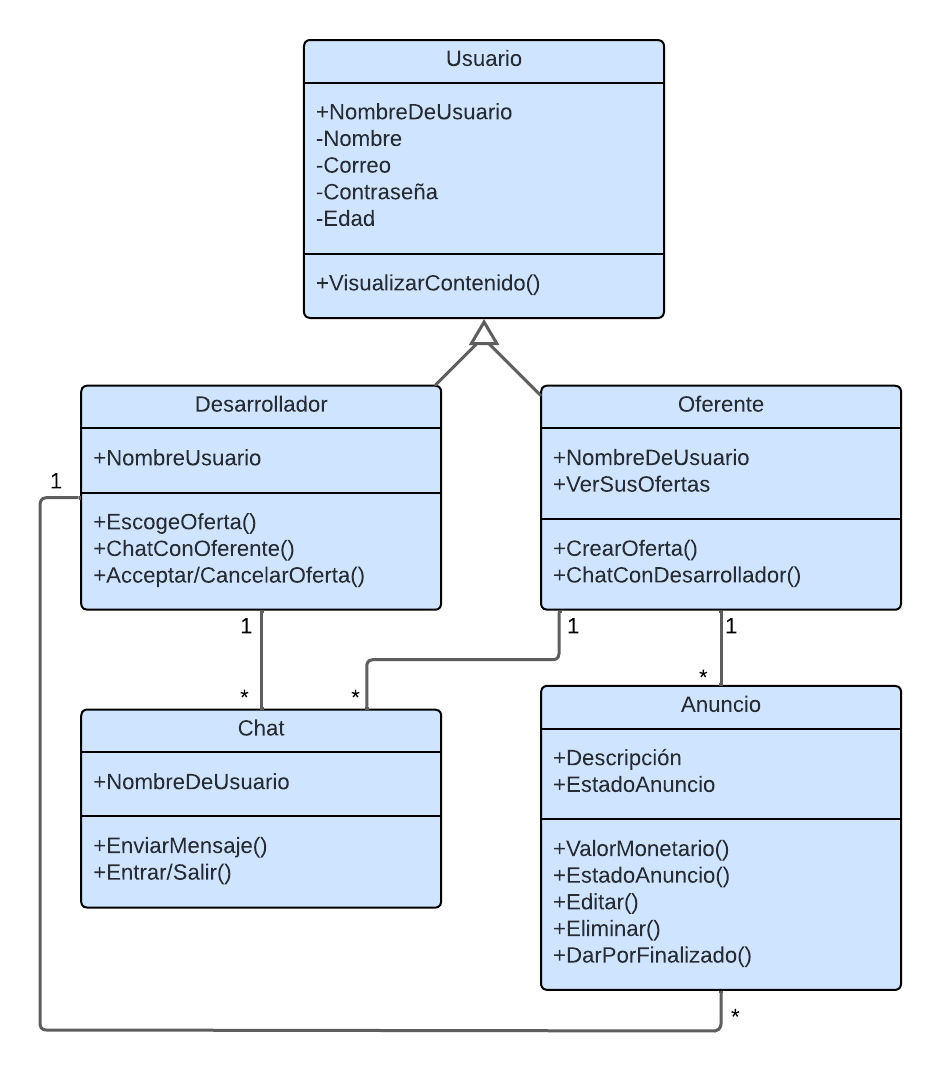


### Diagrama de Actividades



## Vista Lógica

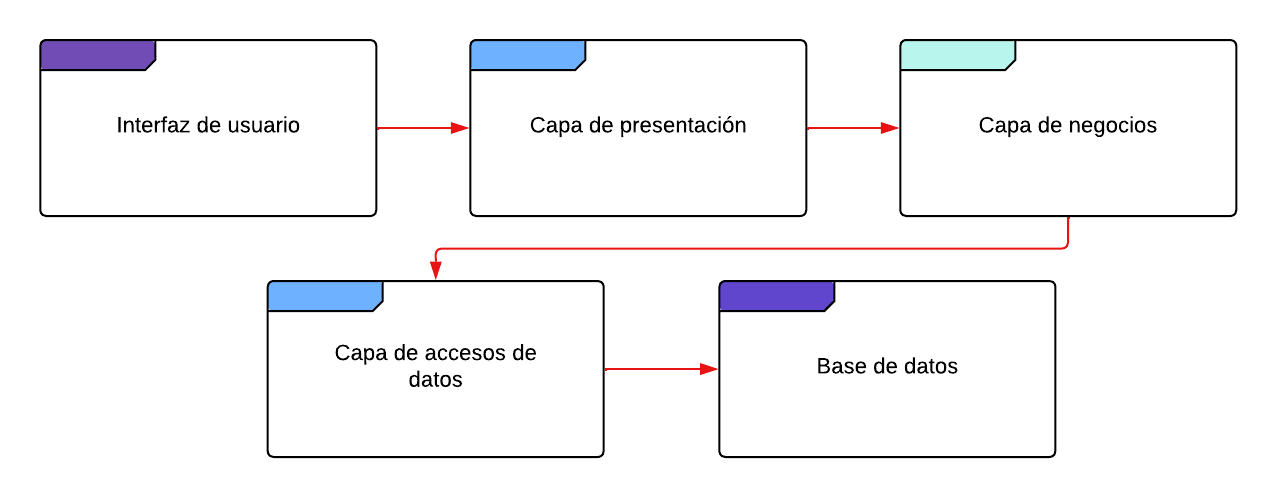
### Diagramas – Clases



## Vista de Implementación

La página web para freelancers se desarrollará utilizando tecnologías modernas y escalables, como HTML5, CSS Se emplea un enfoque modular para facilitar la expansión y mantenimiento del sistema, con integración de bases de datos SQL o según las necesidades específicas del proyecto.

### Diagrama de Paquetes

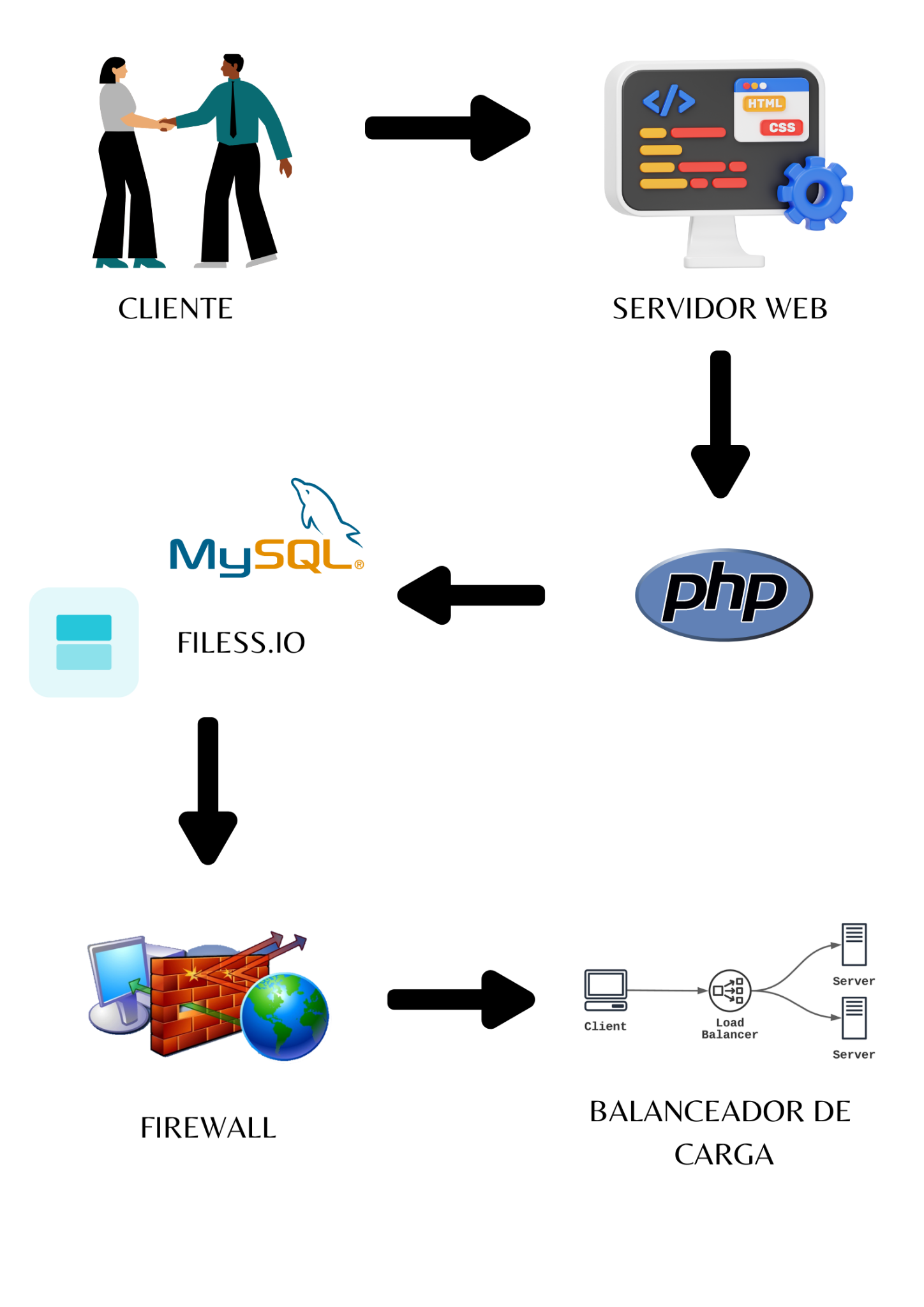


## 

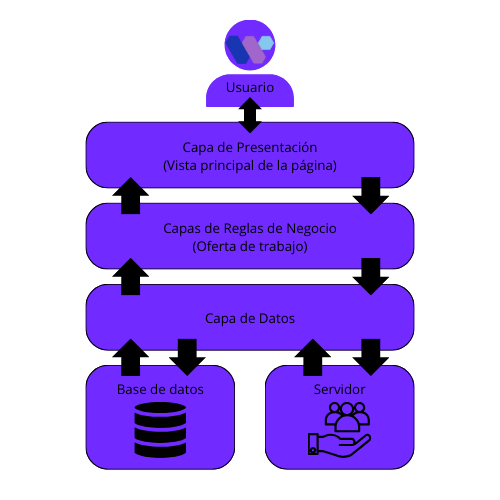
## 3.5 Vista de Despliegue

La página web estará desplegada en un entorno de alojamiento web confiable, utilizando servicios como Apache para el servidor web y PHP para interactuar con la base de datos MySQL. Se establecerán medidas de seguridad robustas, como el cifrado SSL, para proteger la información sensible de los usuarios.

### Diagrama de despliegue

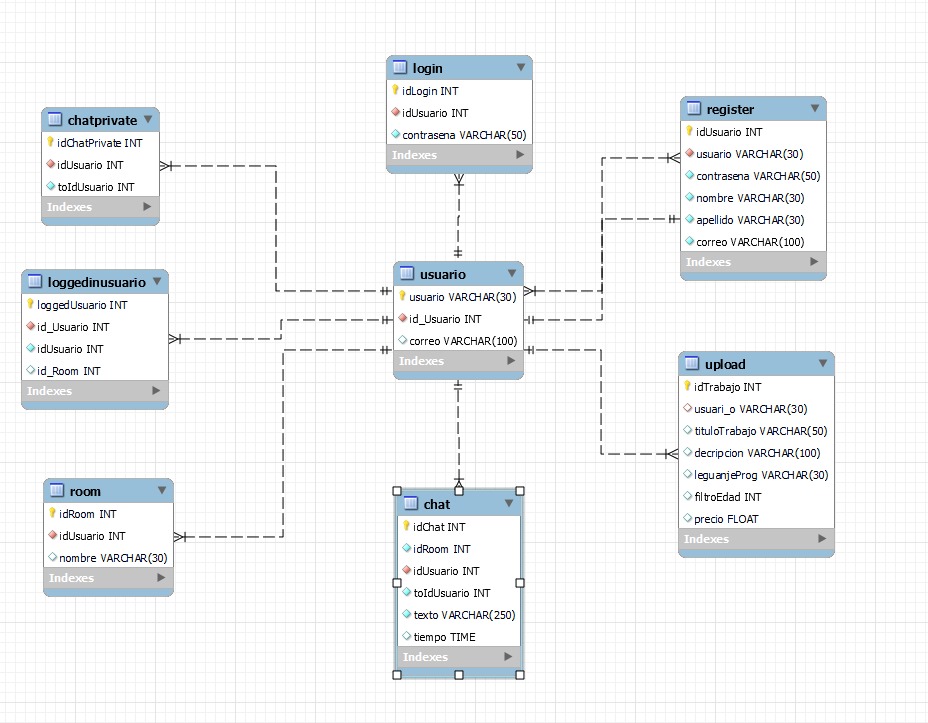


# Arquitectura en capas



# Vista de Datos

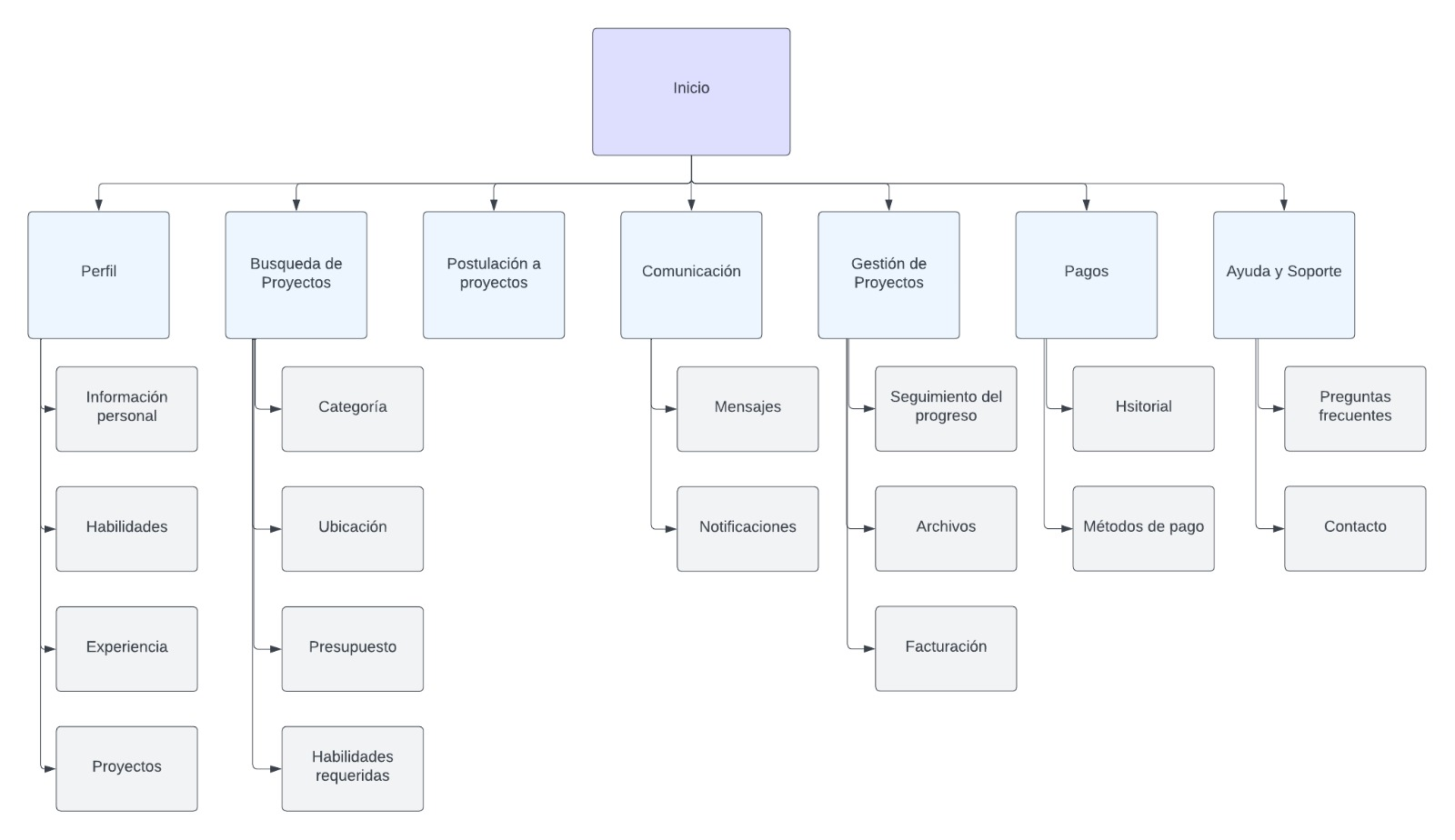
## Modelo Relacional



# Definición de Interfaces de Usuario

Este mapa de navegación define la estructura y organización de las diferentes secciones y funcionalidades de la página web para freelancers. Permite a los usuarios encontrar fácilmente la información y los servicios que buscan.

**-Mapa de navegación. Demostración de las interface**



# Características Generales de Calidad

**1. Funcionalidad:** La página web debe ofrecer todas las funcionalidades que los freelancers necesitan, como búsqueda de proyectos, gestión de clientes, comunicación, pagos, etc., de forma completa, correcta y segura.

**2. Rendimiento:** Debe ser rápida, receptiva, escalable y eficiente para soportar un alto número de usuarios y proyectos sin comprometer la experiencia del usuario.

**3. Usabilidad:** La página web debe ser fácil de usar, intuitiva y accesible para freelancers y clientes, independientemente de su experiencia o habilidades.

**4. Mantenibilidad:** Debe ser fácil de modificar, actualizar y documentar para facilitar su evolución y adaptación a nuevas necesidades.

**5. Portabilidad:** Debe funcionar en diferentes plataformas y navegadores web, y ser fácil de migrar a un nuevo servidor o entorno.

**6. Seguridad:** La información de los usuarios debe ser confidencial, precisa, completa y disponible 24/7, protegiendo la página web contra accesos no autorizados, errores y ataques.

**7. Confiabilidad:** La página web debe ser tolerante a fallos, errores y ataques, con mecanismos de recuperación para minimizar el impacto en los usuarios.

**8. Eficiencia:** Debe utilizar los recursos del sistema de forma eficiente, minimizando costes y optimizando el rendimiento.

**9. Escalabilidad:** Debe ser capaz de crecer y adaptarse a nuevas necesidades, con flexibilidad para cambios en el entorno y la demanda.

**10. Interoperabilidad:** Debe ser compatible con otros sistemas, como plataformas de pago o sistemas de gestión de proyectos, para facilitar la integración y el intercambio de datos.

# Stack Tecnológico

Para el **Front-end**, hay lenguajes de programación como el HTML5, CSS3 y JavaScript. Para Frameworks: ReactJS o VueJS para una interfaz de usuario moderna y dinámica, y para las librerías, Bootstrap para una experiencia de usuario consistente y atractiva.

Para el **Back-end,** hay lenguajes de programación como Python o NodeJS para un desarrollo rápido y escalable, para las Bases de datos, 2 muy recomendadas y usadas son PostgreSQL o MySQL para el almacenamiento seguro y eficiente de datos, y para un Alojamiento seguro, esta AWS, Google Cloud o Heroku para un entorno de producción confiable y escalable.

**Herramientas adicionales:**

- Git: Para el control de versiones del código fuente.

- Herramientas de testing: Jest o Mocha para asegurar la calidad del código.

- Herramientas de automatización: Jenkins o Travis CI para automatizar tareas repetitivas.

**REFERENCIAS:**

<https://kinsta.com/blog/vue-vs-react/>

<https://www.temok.com/blog/aws-vs-google-cloud-vs-heroku/>

<https://git-scm.com/about>