**PET HERO**

TECNICATURA UNIVERSITARIA EN PROGRAMACIÓN

METODOLOGÍA DE SISTEMAS I

**Docentes: Fernando Castañeda**

**SCIACCO DANIEL ESTEBAN**

**JUAN JOSÉ DECOUD**

**PABLO OLIVERA**

**GitHub:** [**https://github.com/PabloKaz/TP-FINAL-2022**](https://github.com/PabloKaz/TP-FINAL-2022)

**Fecha de Entrega: Miércoles 23 de Noviembre de 2022**

**CONTENIDO**

1. **INTRODUCCIÓN**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_3

1.1 Propósito \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_3

1.2 Alcance\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_3

1.3 Definiciones, acrónimos y abreviaturas\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_3

1.4 Referencias\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_4

1. **DESCRIPCIÓN GENERAL DEL SISTEMA**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_5

2.1 Perspectiva de la aplicación\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_5

2.2 Funciones del sistema\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_5

2.3 Objetivos\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_5

2.4 Características de los usuarios\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_6

1. **DEFINICIÓN DE REQUISITOS DEL SISTEMA**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_7

3.1 Definición de Requisitos Funcionales\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_7

3.2 Definición de Requisitos No Funcionales\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_7

1. **CASOS DE USO**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_8

4.1 Especificación de Casos de Uso: Realizar una reserva\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_8

4.2 Diagrama de Casos de Uso\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_9

1. **DIAGRAMA DE CLASES CON LA ARQUITECTURA IMPLEMENTADA**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_10

Controladores\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_10

Config\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_11

DAO\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_11

DAOSQL\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_12

Modelos\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_13

1. **DIAGRAMA DE ENTIDAD – RELACION**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_14
2. **INTRODUCCION**

En este documento se presentan las especificaciones de requisitos del software “Pet Hero” donde harán reservas para cuidado de mascotas entre un dueño de mascota y un cuidador.

* 1. **PROPOSITO**

El propósito de este proyecto es conectar a los diferentes dueños de mascotas (Owner) con los diferentes cuidadores (Keeper) que cumplan con los requisitos para cuidar a su mascota durante un tiempo definido. El documento está dirigido tanto a los que tomen el lugar de Owner como a los que tomen el lugar de Keeper para enseñarles la composición y alcance de esta aplicación.

* 1. **ALCANCE**

El sistema se denomina **“Pet Hero”**

Para gestionar el servicio de reserva se deberá gestionar tanto la mascota del Owner y sus características, así como el tiempo durante el cual se desea que cuiden a su mascota. De esta forma se presentará una lista de Keepers donde se podrá seleccionar aquellos que cumplan con esos requisitos. También se cuenta con una simulación de pago según el tiempo definido y lo estimado por el keeper.

* 1. **DEFINICIONES, ACRÓNIMOS Y ABREVIATURAS**

**Definiciones**:

* Tipo de Usuario Owner: Son los dueños de las mascotas que están registradas en la aplicación. Es el mismo Owner el que registra a su mascota
* Tipo de Usuario Keeper: Son los cuidadores a los que se les encarga el cuidado de las mascotas.
* Aplicación Web: Tipo de software que se codifica en un lenguaje soportado por los navegadores web y cuya ejecución es llevada a cabo por el navegador en Internet o en una intranet.
* Requisito Funcional: Condición que describe qué debe hacer el sistema respecto a su entorno. Especifica los servicios que se espera que el sistema proveerá, sus entradas y salidas, excepciones, etc.
* Requisito No Funcional: Propiedades emergentes del sistema.

**Acrónimos**:

* API: *Application Programming* *Interface*, interfaz de programación de aplicaciones
* DNI: Documento Nacional de Identidad
* HTML: *HyperText Markup Language*, lenguaje de marcas de hipertexto
* PHP: *Hypertext Pre-Processor*, pre-procesador de hipertexto
* RF: Requisito Funcional
* RNF: Requisito No Funcional
* URL: *Uniform Resource Locator*, localizador de recursos uniforme
* UTN: Universidad Tecnológica Nacional

**Abreviaturas**:

* ID: *Identification*, Identificación
  1. **REFERENCIAS**

Para la elaboración de este escrito se han seguido los lineamientos presentados en el siguiente documento:

* Méndez, G. (2008). Especificación de Requisitos según el estándar de IEEE 830. *Facultad de Informática, Universidad Complutense de Madrid.*

1. **DESCRIPCIÓN GENERAL DEL SISTEMA**

En esta sección se describen todos aquellos factores que afectan a la aplicación y sus requisitos.

* 1. **PERSPECTIVA DE LA APLICACIÓN**

Esta aplicación es una aplicación independiente que no está vinculada con ninguna API ni fuente de registros aparte de su propia base de datos que contiene las características de cada usuario ya sea Owner o Keeper, al igual que las características que definen a cada uno, es decir Mascotas y Tarjetas de los Owner y las fechas Disponibles de los Keeper.

* 1. **FUNCIONES DEL SISTEMA**

Las funciones del sistema son:

Para los que están ingresando a la página:

* Login: Nombre de Usuario y Contraseña
* Sign Up: el sistema da la opción de elegir Owner o Keeper antes de mostrar el formulario de registro

Para los Owner:

* Registrarse con los datos pedidos en el formulario
* Agregar Mascotas
* Ver mascotas
* Ver reservas pendientes (el keeper debe aceptarla)
* Ver reservas en curso
* Realizar reserva:
* Seleccionar una mascota para el servicio
* Seleccionar un keeper de los que aparecen en la lista que cumplen los requisitos
* Realizar la mitad del pago

Para los Keeper:

* Registrarse con los datos pedidos en el formulario
* Ver fechas disponibles registradas
* Agregar/Eliminar fechas disponibles
* Ver reservas pendientes
* Aceptar/Rechazar reservas
* Ver reservas en curso
* Finalizar Reservas
  1. **OBJETIVOS**

El objetivo de la aplicación web es generar un espacio para que los Owner puedan hacer una reserva con un Keeper por el cuidado de su mascota a cambio de una remuneración.

* 1. **CARACTERISTICAS DE LOS USUARIOS**

Los usuarios de la aplicación son:

* Owner: dueños de las mascotas que desean registrar en la aplicación.
* Keeper: cuidadores que seleccionan los Owner para el cuidado de sus mascotas.

1. **DEFINICIÓN DE REQUISITOS DEL SISTEMA**

En esta sección se presentan los requisitos funcionales y no funcionales que deberán ser cumplidos por el sistema.

* 1. **DEFINICIÓN DE REQUISITOS FUNCIONALES:** *Tabla 1 Requisitos Funcionales*

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **NOMBRE** | **DESCRIPCION** | **USUARIO** |
| **RF-001** | Ingresar Nuevo Owner | El sistema deberá permitir que un nuevo Owner se registre ingresando: nombre, apellido, DNI, teléfono, nombre de usuario, contraseña. | Owner |
| **RF-002** | Ingresar Nueva Mascota | El Owner deberá ingresar: nombre, raza, tamaño, foto, plan de vacunación, video y observación de la mascota a registrar | Owner |
| **RF-003** | Consultar Listado de Mascotas | En el menú de Owner aparece una opción “Mascotas” para consultar el listado de mascotas | Owner |
| **RF-004** | Ingresar Nuevo Keeper | El sistema deberá permitir que un nuevo Owner se registre ingresando: nombre, apellido, DNI, teléfono, nombre de usuario, contraseña, remuneración solicitada y tamaño de mascota. | Keeper |
| **RF-005** | Indicar Disponibilidad de Estadías | En el menú de Keeper aparece la opción para ver, agregar y eliminar la disponibilidad de estadías del Keeper | Keeper |
| **RF-006** | Consultar Listado de Keepers como Owner | Esta unido al RF-007. Para consultar los keeper hay que seguir esos pasos. | Owner |
| **RF-007** | Consultar Disponibilidad de Keepers en un rango de fechas | Cuando se selecciona una mascota y fecha para una reserva el sistema muestra una lista de Keepers que cumplan con los requisitos de “disponibilidad horaria” | Owner |
| **RF-008** | Generar nueva reserva desde un Owner a un Keeper | Una vez cumplido el RF-007 se selecciona uno de los keeper de la lista y se envía la solicitud de reserva. Si la acepta se envía la mitad del pago de la reserva. | Owner |
| **RF-009** | Consultar mis reservas programadas e históricas como Keeper | El sistema deberá permitir ver al usuario sus reservas programadas y consultar las históricas | Keeper |
| **RF-010** | Confirmar reserva como Keeper | En el menú de keeper aparece la opción de “Reservas Pendientes” donde muestra las reservas que fueron enviadas al keeper para confirmar. Para aceptar la reserva pulsar “Aceptar”, sino, “Rechazar” | Keeper |
| **RF-011** | Generar nuevo cupón de Pago para un Owner | El sistema deberá generar un cupón de pago por el 50% del total de la estadía. Una vez realizado el pago el sistema deberá generar un nuevo cupón de pago por el valor restante de la estadía que solo permitirá abonar al final de la misma. | Keeper |
| **RF-012** | Simulación de pago de cupón (confirmar reserva) | El sistema deberá confirmar la estadía luego de que el usuario realice el pago del cupón por el 50% del valor total | Owner |

* 1. **DEFINICIÓN DE REQUISITOS NO FUNCIONALES:** *Tabla 2: Requisitos No Funcionales*

|  |  |
| --- | --- |
| **ID** | **DESCRIPCIÓN** |
| **RNF-001** | Programación en capas de la aplicación respetando la arquitectura de 3 capas lógicas vista durante la cursada. Esto implica el desarrollo de las clases que representen las entidades del modelo y controladoras de los casos de uso, las vistas y la capa de acceso a datos. |

1. **CASOS DE USO**
   1. **ESPECIFICACIÓN DE CASO DE USO: Realizar una reserva.**

A continuación veremos una Especificación de Caso de Uso referido a un Owner que realiza una reserva:

*Tabla 3: Plantilla Caso de Uso*

|  |  |
| --- | --- |
| **Reserva de Cuidado de Mascota** | |
| **Identificador** |  |
| **Nombre** | **Reserva de Cuidado de Mascota** |
| **Fecha** | **16/11/2022** |
| **Analistas** | **Daniel Sciacco – Juan José Decoud – Pablo Olivera** |
| **Actores** | **Owner** |
| **Objetivos** | **A través de este proceso se podrán ingresar nuevas reservas** |
| **Precondiciones** |  |

*Tabla 4: Escenario de Éxito*

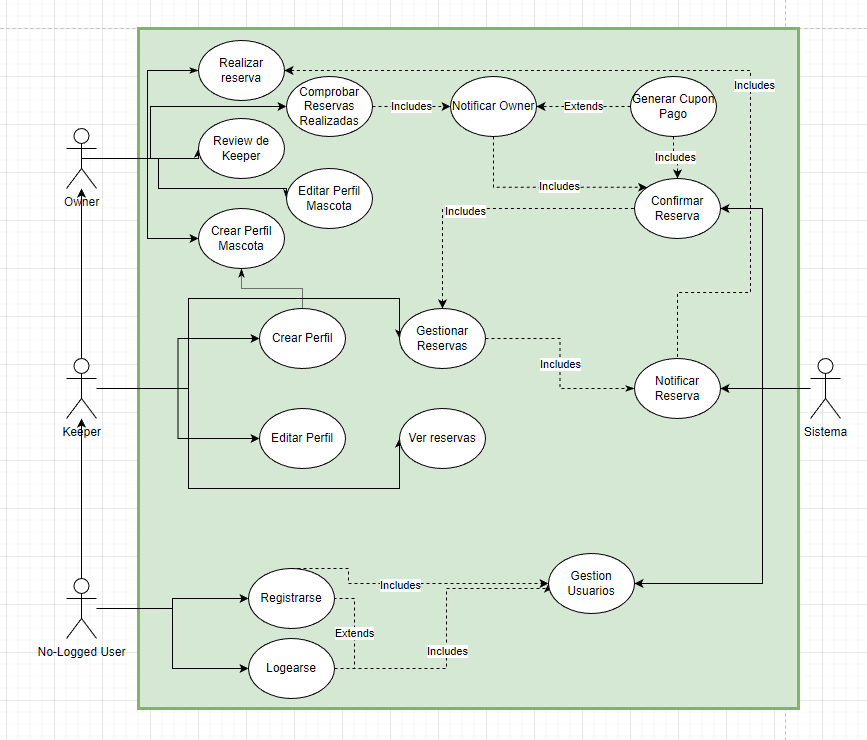
|  |  |
| --- | --- |
| **ACTOR** | **SISTEMA** |
| El Owner Selecciona una de sus mascotas y solicita la reserva con el botón correspondiente | El sistema verifica :  El tamaño de la mascota concuerde con el que acepta el Keeper.  Que la mascota no tenga ninguna reserva en proceso para ningún Keeper. |
| El Owner Selecciona las fechas que quiere que cuiden a su mascota. | El sistema genera una reserva en estado pendiente reservando esas fechas para que no puedan ser solicitadas por otros clientes y le envía dicha solicitud al Keeper |
| El Keeper acepta la reserva | Cambia el estado de la reserva a “Aceptada” y le envía al Owner un cupón de pago |
| El Owner paga el 50% de la reserva | Cambia el estado de la reserva a “Confirmada” |
| El Owner Selecciona una de sus mascotas y solicita la reserva con el botón correspondiente | El sistema verifica :  El tamaño de la mascota concuerde con el que acepta el Keeper.  Que la mascota no tenga ninguna reserva en proceso para ningún Keeper. |

*Tabla 5: Alternativas o excepciones*

|  |  |
| --- | --- |
| El Owner ya posee una estadía “Confirmada” o “Pendiente” para la mascota seleccionada. | El sistema comprueba si hay reservas en estado pendiente o confirmada para dicha mascota, en caso de ser así, le avisa al Owner que no puede realizar una nueva ya que todavía tiene reservas en proceso. |
| El tamaño de la mascota no es permitido por el Keeper | El sistema le avisa al Owner que el tamaño de la mascota no está permitido por el Keeper |
| El Keeper rechaza la reserva | El Sistema elimina la reserva y establece nuevamente como disponibles las fechas que habían sido seleccionadas por el Owner. |

* 1. **DIAGRAMA DE CASOS DE USO.**

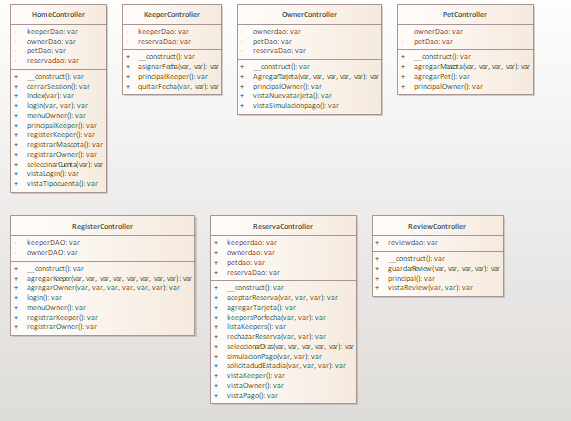
*Figura 1: Diagrama de Casos de Uso*



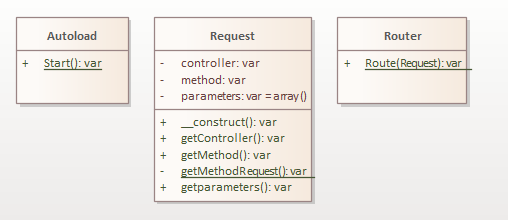
1. **DIAGRAMA DE CLASES**

En este punto se presentan los diagramas de clases con la arquitectura implementada.

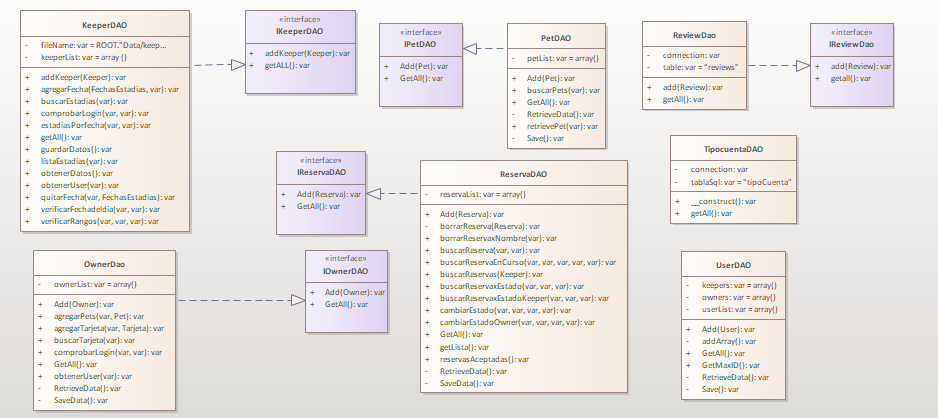
*Figura 2: Controladores*

****

*Figura 3: Config*

****

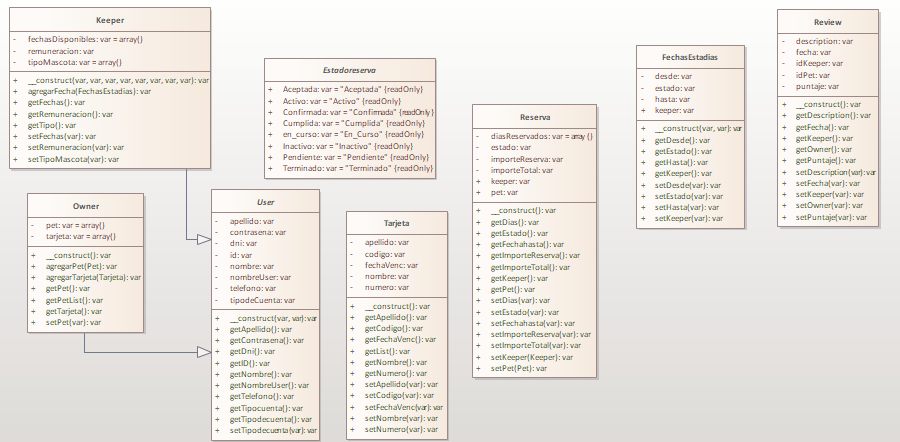
*Figura 4: DAO*

****

*Figura 5: DAOSQL*

****

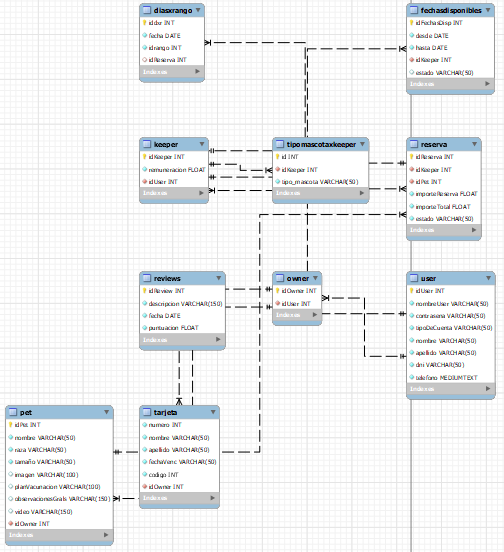
*Figura 6: Modelos*

****

1. **DIAGRAMA DE ENTIDAD – RELACION**

En este punto se presenta el Diagrama de Entidad – Relación de la base de datos del proyecto

*Figura 7: Diagrama de Entidad – Relación*

****