

|  |
| --- |
|  |

Especificación de requisitos de software

Proyecto: VetTrack

**PRIMERA REVISION**

|  |  |  |
| --- | --- | --- |
|  |  | Octubre |

Ficha del documento

|  |  |  |  |
| --- | --- | --- | --- |
| **Fecha** | **Revisión** | **Autor** | **Verificado dep. calidad.** |
| [Fecha] | v0.5 | [Descripción] | [Firma o sello] |

Documento validado por las partes en fecha: [Fecha]

|  |  |
| --- | --- |
| Por el cliente | Por la empresa suministradora |
|  |  |
| Fdo. D./ Dña [Nombre] | Fdo. D./Dña [Nombre] |

Contenido

[Ficha del documento 3](#_Toc33411057)

[Contenido 4](#_Toc33411058)

[1 Introducción 6](#_Toc33411059)

[1.1 Propósito 6](#_Toc33411060)

[1.2 Alcance 6](#_Toc33411061)

[1.3 Personal involucrado 6](#_Toc33411062)

[1.4 Definiciones, acrónimos y abreviaturas 7](#_Toc33411063)

[1.5 Referencias 7](#_Toc33411064)

[1.6 Resumen 7](#_Toc33411065)

[2 Descripción general 7](#_Toc33411066)

[2.1 Perspectiva del producto 7](#_Toc33411067)

[2.2 Funcionalidad del producto 7](#_Toc33411068)

[2.3 Características de los usuarios 8](#_Toc33411069)

[2.4 Restricciones 8](#_Toc33411070)

[2.5 Suposiciones y dependencias 8](#_Toc33411071)

[2.6 Evolución previsible del sistema 8](#_Toc33411072)

[3 Requisitos específicos 8](#_Toc33411073)

[3.1 Requisitos comunes de los interfaces 9](#_Toc33411074)

[3.1.1 Interfaces de usuario 9](#_Toc33411075)

[3.1.2 Interfaces de hardware 9](#_Toc33411076)

[3.1.3 Interfaces de software 9](#_Toc33411077)

[3.1.4 Interfaces de comunicación 9](#_Toc33411078)

[3.2 Requisitos funcionales 9](#_Toc33411079)

[3.2.1 Requisito funcional 1 10](#_Toc33411080)

[3.2.2 Requisito funcional 2 10](#_Toc33411081)

[3.2.3 Requisito funcional 3 10](#_Toc33411082)

[3.2.4 Requisito funcional n 10](#_Toc33411083)

[3.3 Requisitos no funcionales 10](#_Toc33411084)

[3.3.1 Requisitos de rendimiento 10](#_Toc33411085)

[3.3.2 Seguridad 10](#_Toc33411086)

[3.3.3 Fiabilidad 10](#_Toc33411087)

[3.3.4 Disponibilidad 10](#_Toc33411088)

[3.3.5 Mantenibilidad 10](#_Toc33411089)

[3.3.6 Portabilidad 11](#_Toc33411090)

[3.4 Otros requisitos 11](#_Toc33411091)

[4 Apéndices 11](#_Toc33411092)

# Introducción

Se necesita un programa para gestionar una clínica localizada en León que cuenta con (por ahora) una plantilla de 3 personas que son las encargadas de dirigir y gestionar la clínica. Se nos ha pedido que podamos realizar una aplicación de escritorio para poder hacer esto mucho más rápido.

Para ello implementaremos 3 interfaces distintas, una para iniciar sesión, otra para usuarios y otra para los empleados de la aplicación en la que gestionaremos no solo las mascotas de cada persona sino también implementaremos otras funciones como por ejemplo el manejo del inventario, un historial de cada persona en la que aparece todas las acciones en la aplicación y un horario.

## Propósito

Poder gestionar tanto a las personas encargadas de la clínica como a los clientes para poder crear una mayor fluidez y organización del personal, inventario y de las instalaciones.

El propósito de este escrito seria poder dejar todo claro acerca de la aplicación para generar un único camino que tenemos que seguir a la hora de poder programar y entregar la aplicación a la fecha señalizada. Va dirigido a las personas encargadas de programar la aplicación como para los dueños de la clínica. Si cualquier cosa no es de agrado de los dueños siempre se podrá modificar.

## Alcance

Hemos decidido usar el nombre VetTrack por ser un nombre corto y describe la actividad que tiene como objetivo nuestra aplicación. Esto hace que sea más fácilmente reconocible y por tanto que llegue a más audiencia.

El objetivo de esta aplicación es facilitar la organización y el trabajo del personal de la clínica veterinaria

## Personal involucrado

|  |  |
| --- | --- |
| Nombre | Víctor Acevedo Hernández |
| Rol | Administrador/Desarrollador |
| Categoría profesional | Ingeniero Informático y desarrollador Web |
| Responsabilidades | Encargado del back-end |
| Información de contacto | vacevh00@estudiantes.unileon.es |
| Aprobación |  |

|  |  |
| --- | --- |
| Nombre | Mateo González Alonso |
| Rol | Administrador/Desarrollador |
| Categoría profesional | Ingeniero Informático y desarrollador Web |
| Responsabilidades | Encargado del front-end |
| Información de contacto | mgonza40@estudiantes.unileon.es |
| Aprobación |  |

.

## Definiciones, acrónimos y abreviaturas

Algunos términos que manejamos podrían ser:

-back-end: La parte del servidor, no puede ser visto por el usuario normal

-front-end: La parte que pueden ver el usuario, interfaces, botones, cuadros de texto…

-interfaz: Parte del programa con la que el usuario puede interactuar

-backup: copia de seguridad para no perder progreso

## Referencias

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Referencia** | **Titulo** | **Ruta** | **Fecha** | **Autor** |
| 9ª edición | Software engineering | [Ruta](https://engineering.futureuniversity.com/BOOKS%20FOR%20IT/Software-Engineering-9th-Edition-by-Ian-Sommerville.pdf) | 2011 | Ian Sommerville |
|  |  |  |  |  |

Relación completa de todos los documentos relacionados en la especificación de requisitos de software, identificando de cada documento el titulo, referencia (si procede), fecha y organización que lo proporciona.

## Resumen

El contenido del resto del documento recoge las características de la aplicación VetTrack, tanto las características de implementación como la descripción, requisitos y funcionalidades del mismo.

# Descripción general

## Perspectiva del producto

El producto es un producto independiente, no seria la expansión de uno mayor ya que estaríamos gestionando la aplicación de la clínica entera. A partir de este producto se podrían ir añadiendo otras modificaciones en el futuro que complementen a este. Si tuviésemos que posicionarlo en un punto seria la base ya que hay funciones como la gestión de usuarios que es básico para el funcionamiento de este

## Funcionalidad del producto

Las principales funcionalidades del producto como mencionamos en el anterior apartado serian el correcto funcionamiento de la gestión de usuarios, una correcta implementación de un historial de actividades de esa cuenta y la posible (pero aun por decidir) implementación de una gestión de fechas para poder pedir citas y que un administrador pueda aceptarlas y en caso de rechazarlas poder modificar para cuando podría ser la siguiente. Entre otras funcionalidades mas pequeñas podemos ver la implementación de un sistema de inicio de sesión, una interfaz para los administradores que sea distinta de los usuarios normales (clientes) y una funcionalidad de búsqueda en la que los administradores pueden buscar usuarios no tan solo por su nombre sino también por nombre de sus mascotas. La aplicación a priori podemos ver que manejamos animales de todo tipo (aun por decidir) y un sistema de implementación de inventario en el que aparecen productos con su respectivo precio, en la que no manejamos el patrimonio de cada cuenta, cada usuario tiene que pagar en físico

## Características de los usuarios

|  |  |
| --- | --- |
| Tipo de usuario | Administrador |
| Formación | Profesional veterinario con conocimiento del programa |
| Habilidades | Conocimiento de artículos de veterinaria, uso básico de un ordenador |
| Actividades | Capacidad para gestionar ventas, crear usuarios para los nuevos clientes, crear nuevos perfiles para las mascotas, editar las características de la mascota, generar nuevas citas |

|  |  |
| --- | --- |
| Tipo de usuario | Usuario |
| Formación | No es necesaria formación |
| Habilidades | Uso básico de un ordenador |
| Actividades | Ver sus compras, el perfil de su mascota y las citas asociadas a ella |

## Restricciones

No se manejaran saldos aun teniendo productos con precio, implementaremos todo en el sistema de lenguaje de java con (aun por decidir) una base de datos. Tendremos varios archivos con distintas clases entre las que se encuentran administrador, cliente, inicio de sesión (main) y llamadas a la base de datos

## Suposiciones y dependencias

Se implementa en Windows, si este se ejecutase en Linux se ejecutaría de una manera completamente distinta

## Evolución previsible del sistema

Un historial de búsqueda de productos, un historial de fechas venideras y pasadas

# Requisitos específicos

Esta es la sección más extensa y más importante del documento.

Debe contener una lista detallada y completa de los requisitos que debe cumplir el sistema a desarrollar. El nivel de detalle de los requisitos debe ser el suficiente para que el equipo de desarrollo pueda diseñar un sistema que satisfaga los requisitos y los encargados de las pruebas puedan determinar si éstos se satisfacen.

Los requisitos se dispondrán en forma de listas numeradas para su identificación, seguimiento, trazabilidad y validación (ej. RF 10, RF 10.1, RF 10.2,...).

Para cada requisito debe completarse la siguiente tabla:

|  |  |  |  |
| --- | --- | --- | --- |
| Número de requisito | RF 1 | | |
| Nombre de requisito | Las mascotas deben estar asociadas a un dueño | | |
| Tipo | Requisito | Restricción | |
| Fuente del requisito | TO-DO | | |
| Prioridad del requisito | Alta/Esencial | Media/Deseado | Baja/ Opcional |

|  |  |  |  |
| --- | --- | --- | --- |
| Número de requisito | RF 2 | | |
| Nombre de requisito |  | | |
| Tipo | Requisito | Restricción | |
| Fuente del requisito | TO-DO | | |
| Prioridad del requisito | Alta/Esencial | Media/Deseado | Baja/ Opcional |

|  |  |  |  |
| --- | --- | --- | --- |
| Número de requisito | RF 1 | | |
| Nombre de requisito | Las mascotas deben estar asociadas a un dueño | | |
| Tipo | Requisito | Restricción | |
| Fuente del requisito | TO-DO | | |
| Prioridad del requisito | Alta/Esencial | Media/Deseado | Baja/ Opcional |

|  |  |  |  |
| --- | --- | --- | --- |
| Número de requisito | RF 1 | | |
| Nombre de requisito | Las mascotas deben estar asociadas a un dueño | | |
| Tipo | Requisito | Restricción | |
| Fuente del requisito | TO-DO | | |
| Prioridad del requisito | Alta/Esencial | Media/Deseado | Baja/ Opcional |

|  |  |  |  |
| --- | --- | --- | --- |
| Número de requisito |  | | |
| Nombre de requisito |  | | |
| Tipo | Requisito | Restricción | |
| Fuente del requisito |  | | |
| Prioridad del requisito | Alta/Esencial | Media/Deseado | Baja/ Opcional |

y realizar la descripción del requisito

La distribución de los párrafos que forman este punto puede diferir del propuesto en esta plantilla, si las características del sistema aconsejan otra distribución para ofrecer mayor claridad en la exposición.

## Requisitos comunes de los interfaces

Descripción detallada de todas las entradas y salidas del sistema de software.

### Interfaces de usuario

El usuario va a tener un total de 3 posibles interfaces.

-Una para administrador en el que tendrá botones a mayor de la interfaz de usuario con algunas funcionalidades a mayores para poder modificar/crear/borrar posibles eventos

-Una interfaz de usuario en la que tendrá un par de funcionalidades en la que podrá buscar información, solicitar eventos y (OPCIONAL) adquirir productos

-Una interfaz común de inicio de sesión que tendremos 2 cuadros de texto (uno para contraseñas y otro para el nombre de usuario), no manejaremos el cambio de contraseña por si se le ha olvidado al usuario ya que por ahora esto podría generar un par de problemas con la seguridad. (OPCIONAL) Creación de usuarios (nombre + username + contraseña + animales que tiene + información del animal)

### Interfaces de hardware

No se necesita un hardware específico

### Interfaces de software

Indicar si hay que integrar el producto con otros productos de software.

Para cada producto de software debe especificarse lo siguiente:

* Descripción del producto software utilizado
* Propósito del interfaz
* Definición del interfaz: contiendo y formato

### Interfaces de comunicación

Describir los requisitos del interfaces de comunicación si hay comunicaciones con otros sistemas y cuales son las protocolos de comunicación.

## Requisitos funcionales

Definición de acciones fundamentales que debe realizar el software al recibir información, procesarla y producir resultados.

En ellas se incluye:

* Comprobación de validez de las entradas
* Secuencia exacta de operaciones
* Respuesta a situaciones anormales (desbordamientos, comunicaciones, recuperación de errores)
* Parámetros
* Generación de salidas
* Relaciones entre entradas y salidas (secuencias de entradas y salidas, formulas para la conversión de información)
* Especificación de los requisitos lógicos para la información que será almacenada en base de datos (tipo de información, requerido)

Las requisitos funcionales pueden ser divididos en sub-secciones.

### Requisito funcional 1

### Requisito funcional 2

### Requisito funcional 3

### Requisito funcional n

## Requisitos no funcionales

### Requisitos de rendimiento

El sistema tendrá que soportar un número considerable de usuarios. La aplicación al ser de escritorio tendrá como límite las capacidades del propio ordenador. Ya que por aplicación iniciada se hará 1 usuario, si se abren 3 aplicaciones tendrá 3 usuarios. Nosotros al no manejar dinero (AUN POR DECIDIR) no tendremos ningún tipo de opción de transacción. Todo tipo de compras se pueden generar desde la aplicación, pero se tendrá que ir a entregar el dinero y recoger el producto desde la propia tienda

### Seguridad

Tendremos en la primera interfaz (la interfaz de login) el saneamiento de los cuadros para meter las credenciales (no se permitirán “, ‘, paréntesis…) en que primero lo pasaremos el texto a texto plano y haremos un replace de todos estos caracteres

Hashearemos las contraseñas en la base de datos con un “salt” que funcionara de la siguiente manera: “username\_password” y todo ira a un md5.

Tendremos un archivo en el que se registraran todas las actividades que se hagan en la aplicación

La aplicación al ser un .class no se podrá modificar el código por lo que no podremos cambiar el código fuente

Tendremos cuidado con el BufferOverflow y los caracteres nulos para evitar posibles ataques de este

(OPCIONAL) Para poder entrar en Administrador tendremos que poner un numero/cadena de caracteres a parte de la contraseña

### Fiabilidad

A la hora de tener en cuenta los posibles fallos de aplicación o ataques, de encontrar alguno se avisaría a los desarrolladores para poder solucionarlo lo antes posible, para errores/fallos de la base de datos se crean un backup de usuarios en los que cada semana se haría un backup.

Siempre se va a tener una copia de la aplicación de backup que se ira modificando al mismo tiempo que la aplicación que se usaría de cara al publico para tener un “respaldo” por si se llegase a realizar un ataque de deformación de la aplicación.

### Disponibilidad

La aplicación estará disponible la gran mayoría del tiempo, excepto 1 hora entre las 4-5am para poder reiniciar el dispositivo que contiene la aplicación, para que podamos realizar las actualizaciones (si hubiese alguna del sistema) o posibles modificaciones de alguna parte de la aplicación. Exceptuando cuando se tengan que efectuar cambios en el sistema un poco mas grandes (arreglo de fallos de seguridad, que se quitaría la aplicación de cara al publico para no generar mas problemas a la empresa) y se trabajaría lo antes posible para poder arreglar estos fallos.Si son fallos de bajo y algunos de medio riesgo estos solo se arreglarían en las horas antes mencionadas (4-5 al dia) y si son algunos medios y graves se usaría el plan antes mencionado

### Mantenibilidad

Las tareas de mantenimiento o ampliación de la aplicación deben ser llevadas a cabo por un desarrollador, ya que habría que modificar el código fuente de la aplicación.

Lo que si puede hacer el administrador es añadir nuevos clientes a la base de datos ya que se añadirá una función a la interfaz para facilitar este servicio.

A priori, no se preeveen mantenimientos totalmente necesarios para la aplicación

### Portabilidad

La portabilidad sería en cierto modo compleja ya que habría que desarrollar una nueva interfaz para dispositivos móviles, lo que conllevaría un nuevo desarrollo como mínimo de la interfaz, siendo probable que se tuvieran que usar otras herramientas de desarrollo.

El lenguaje que hemos decidido utilizar es Java, que facilita en cierta manera la portabilidad y además dispone de numerosas librerías y recursos para facilitar la implementación

El programa lo desarrollaremos orientado a Windows, aunque el cambio a Linux tampoco supondría demasiados problemas

## Otros requisitos

Los requisitos podrían variar dependiendo de las leyes de la zona. Podrían estar influenciados a la hora de poner datos, por ejemplo el añadir un campo de datos a la hora de crear un usuario con los animales que sea por ejemplo un seguro especifico para el animal, o si se ve influenciado por una cuota a la hora de tener un animal en especifico. Otro ejemplo podría ser que si se aprobase una ley por ejemplo para que los perros rescatados de la calle se adoptan estos tendrían un descuento a la hora de comprar comida, poner vacunas…

# Apéndices

Para empezar pensamos que el usuario no siempre seguirá las instrucciones indicadas o intentara crear problemas para el servidor, por lo que para empezar nos centraremos en crear una interfaz de inicio de sesión segura, en la que no se podrá saltar el inicio de sesión.

Utilizaremos un total de 2 hosts, uno en la nube en la que tendremos la base de datos con todos los usuarios (con contraseña pasada a hash con un salt para que no pongamos la contraseña en peligro en caso de ataque) y otro en la que residirá la aplicación de escritorio. Tendremos que tener conexión de internet para poder utilizar la aplicación.

La aplicación la estaremos modificando con la ayuda de aplicaciones de edición de código y luego la subiremos a aplicaciones de GitHub para poder descargar la aplicación y tener una especie de backup para no perder progreso (vacevh00 y mgonza40)