

**UNIVERSIDAD PRIVADA DE TACNA**

**FACULTAD DE INGENIERÍA**

**Escuela Profesional de Ingeniería de Sistemas**

**Desarrollo de Aplicativo Móvil “SOS Mascota”**

Curso: Soluciones Móviles I

Docente: Mag. Elard Rodriguez Marca

Integrantes:

* Brian Danilo Chite Quispe (2021070015)
* Christian Dennis Hinojosa Mucho (2019065161)
* Alvaro Javier Contreras Lipa (2021070020)
* Royser Alonsso Villanueva Mamani (2021071090)

**Tacna – Perú**

**2025**

**Desarrollo de Aplicativo Móvil “SOS Mascota”**

**Documento de Arquitectura de Software**

**Versión 1*.0***

| CONTROL DE VERSIONES | | | | | |
| --- | --- | --- | --- | --- | --- |
| Versión | Hecha por | Revisada por | Aprobada por | Fecha | Motivo |
| 1.0 |  |  |  | 06/05/2025 | Versión Original |

**ÍNDICE GENERAL**

[1. Introducción 4](#_heading=h.y1azpy1iwac)

[1.1. Propósito (Diagrama 4+1) 5](#_heading=h.tyrt0o4h0b84)

[1.2. Alcance 7](#_heading=h.ozhpd5iokjj9)

[1.3. Definiciones, Siglas y Abreviaturas 9](#_heading=h.qy1d5tbw4oca)

[1.4. Organización del documento 10](#_heading=h.khm09j472d97)

[2. Objetivos y Restricciones Arquitectónicas 10](#_heading=h.4dnhvdbc4wcd)

[2.1. Priorización de requerimientos 10](#_heading=h.l0776pmpo8oi)

[2.1.1. Requerimientos Funcionales 11](#_heading=h.iakmpief1kj4)

[2.1.2. Requerimientos No Funcionales - Atributos de Calidad 11](#_heading=h.syrk90tpudfu)

[2.2. Restricciones 12](#_heading=h.jhebnz5cel7x)

[3. Representación de la Arquitectura del Sistema 13](#_heading=h.q2gfn9zkv8o)

[3.1. Vista de Caso de Uso 13](#_heading=h.m0smwvu70jia)

[3.1.1. Diagramas de Casos de Uso 13](#_heading=h.wajz2oxbr271)

[3.2. Vista de Lógica 13](#_heading=h.bc3d5869lvu7)

[3.2.1. Diagrama de Subsistemas (paquetes) 13](#_heading=h.qvi59dfrwm6a)

[3.2.2. Diagrama de Secuencia (vista de diseño) 13](#_heading=h.igmapbs95i94)

[3.2.3. Diagrama de Colaboración (vista de diseño) 13](#_heading=h.ptsz8nvmazsb)

[3.2.4. Diagrama de Objetos 13](#_heading=h.m2twzgxa2xaf)

[3.2.5. Diagrama de Clases 13](#_heading=h.mqv7e7ro79w4)

[3.2.6. Diagrama de Base de Datos 13](#_heading=h.mhl4qlizqwrq)

[3.3. Vista de Implementación (vista de desarrollo) 13](#_heading=h.soyzvsq54qkb)

[3.3.1. Diagrama de arquitectura de software (paquetes) 13](#_heading=h.a3p6lzbcqcy3)

[3.3.2. Diagrama de arquitectura del sistema (diagrama de componentes) 13](#_heading=h.h2ozcz4x8mym)

[3.4. Vista de Procesos 13](#_heading=h.ow6mref8olj0)

[3.4.1. Diagrama de procesos del sistema (diagrama de actividad) 13](#_heading=h.6y122kjhtsf1)

[3.5. Vista de Despliegue 13](#_heading=h.c7vflvykzkr0)

[3.5.1. Diagrama de despliegue 13](#_heading=h.11szuehhbpci)

[4. Atributos de Calidad del Software 14](#_heading=h.7bda8ci8jx0v)

[4.1. Escenario de Funcionalidad 14](#_heading=h.p70e43kchg49)

[4.2. Escenario de Usabilidad 14](#_heading=h.msj0y54kbngr)

[4.3. Escenario de Confiabilidad 14](#_heading=h.tdsxr3pwmn6d)

[4.4. Escenario de Rendimiento 15](#_heading=h.9bjflzgode2z)

[4.5. Escenario de Mantenibilidad 15](#_heading=h.qrdijokbgmom)

[4.6. Otros Escenarios 15](#_heading=h.q18h5ycifqs5)

**Documento de Arquitectura de Software**

1. Introducción

## Propósito

El propósito de este documento es definir la arquitectura del sistema **Mascota SOS**, una aplicación móvil diseñada para facilitar la búsqueda y reporte de mascotas perdidas. Este documento proporciona una visión estructurada de alto nivel que permite comprender cómo los diferentes componentes del sistema interactúan entre sí para cumplir con los requerimientos funcionales y no funcionales.

Está dirigido a los desarrolladores, diseñadores, analistas, testers y otros interesados, con el objetivo de asegurar que todos los miembros del equipo compartan una comprensión común de la estructura, comportamiento y restricciones del sistema.

Objetivos de Diseño:

* Escalabilidad: La arquitectura debe permitir la inclusión de nuevas funcionalidades (como adopción, alertas vecinales, etc.) sin afectar negativamente al sistema.
* Modularidad: El sistema debe estar dividido en componentes independientes que permitan una fácil implementación, mantenimiento y prueba.
* Reutilización: Promover el uso de componentes reutilizables como controladores y servicios comunes.
* Mantenibilidad: Facilitar la identificación y solución de errores mediante una estructura clara y bien documentada.
* Seguridad: Asegurar los datos sensibles del usuario, especialmente en el proceso de autenticación y gestión de la información personal.
* Usabilidad: Garantizar una interfaz clara y fluida para todos los perfiles de usuario (ciudadano y veterinario).

Decisiones y Prioridades Arquitectónicas:

* La aplicación se desarrollará con Flutter como framework para el cliente móvil.
* Se estará usando una base de datos para el aplicativo móvil.
* Se realiza con el patrón de arquitectura MVC (Modelo-Vista-Controlador).
* Se prioriza el cumplimiento de los requerimientos funcionales clave: autenticación, registro de reportes de mascotas, localización en mapa y comunicación entre usuarios.

Diagrama 4+1:

* Vista de casos de uso

Define los requerimientos funcionales a través de escenarios interactivos entre los actores y el sistema. Se centra en lo que el sistema debe hacer.

* Vista Lógica

Representa los componentes funcionales del sistema, como clases, objetos, paquetes y sus relaciones. Muestra la organización del software desde una perspectiva estructural.

* Vista de Procesos

Describe el comportamiento dinámico del sistema. Incluye concurrencia, comunicación y sincronización mediante diagramas de secuencia y actividad.

* Vista de desarrollo

Representa la organización del código fuente y los módulos del sistema. Esta vista es relevante para los desarrolladores e integradores.

* Vista de despliegue

Describe el entorno físico donde el sistema será ejecutado. Se modela mediante un diagrama de nodos que incluye dispositivos móviles, servidores web y bases de datos.

## Alcance

a) Inclusiones

* Registro de reportes de mascotas perdidas o encontradas en base a los datos: Nombre de la mascota, tipo de mascota, descripción de la mascota, imagen referencial, datos de contacto del usuario, ubicación aproximada.
* Consulta de reportes mediante filtros por mascota, orden cronológico, orden geográfico.
* Visualización de la ubicación referencial del reporte mediante una API de mapas.
* Almacenamiento local de los reportes en el dispositivo móvil utilizando SQLite, para permitir acceso sin conexión.
* Interfaz amigable e intuitiva desarrollada con Flutter, orientada a dispositivos Android.

b) Exclusiones

* El sistema estará disponible solo en idioma español.
* El desarrollo estará enfocado únicamente en dispositivos Android.
* No se contempla una plataforma web para administrar o validar reportes.

## Definiciones, Siglas y Abreviaturas

| **Término / Sigla** | **Definición** |
| --- | --- |
| Flutter | Framework de desarrollo de interfaces móviles multiplataforma creado por Google. |
| FireBase | Firebase es una plataforma para el desarrollo de aplicaciones web y aplicaciones móviles. |
| CRUD | Create, Read, Update, Delete. Operaciones básicas sobre datos persistentes. |
| Modelo | Representa los datos y la lógica de la aplicación. Maneja la interacción con la base de datos, la manipulación de datos y la ejecución de la lógica empresarial. |
| RF | Requerimiento Funcional. Describe una función específica que el sistema debe cumplir. |
| RNF | Requerimiento No Funcional. Requisitos relacionados con la calidad, rendimiento, seguridad y otras características del sistema. |
| MVC | Patrón de diseño arquitectónico que divide una aplicación en tres componentes interconectados: el modelo, la vista y el controlador |
| Usuario | Persona que utiliza la aplicación. Puede ser un ciudadano que reporta o encuentra una mascota. |
| Veterinario | Usuario con rol especializado que puede validar o confirmar reportes de mascotas. |
| Mascota | Mascota que es reportada o necesita ayuda, se guardará el nombre, descripción, tipo y dueño. |
| Reporte | Información registrada en el sistema sobre una mascota perdida o encontrada, incluyendo nombre, tipo, foto, descripción y ubicación. |

## Organización del documento

Este documento de arquitectura de software está estructurado en secciones que describen de manera detallada la visión técnica del sistema **Mascota SOS**, cubriendo los aspectos clave del diseño, estructura y comportamiento del sistema desde múltiples perspectivas. A continuación, se detalla la organización del contenido:

* **Sección 1: Introducción** Proporciona una visión general del sistema, su propósito, alcance, términos clave y estructura del documento. Presenta el enfoque arquitectónico basado en el modelo 4+1 vistas.
* **Sección 2: Objetivos y Restricciones Arquitectónicas** Describe los requerimientos funcionales y no funcionales priorizados, así como las restricciones tecnológicas, organizativas y de diseño que condicionan la arquitectura del sistema.
* **Sección 3: Representación de la Arquitectura del Sistema** Presenta las distintas vistas arquitectónicas de acuerdo con el modelo 4+1: casos de uso, lógica, implementación, procesos y despliegue. Cada vista se acompaña de los diagramas UML correspondientes, como casos de uso, clases, objetos, paquetes, secuencia, colaboración, componentes, actividad y despliegue.
* **Sección 4: Atributos de Calidad del Software** Analiza cómo la arquitectura propuesta cumple con atributos de calidad como funcionalidad, usabilidad, confiabilidad, rendimiento y mantenibilidad. Se presentan escenarios específicos para cada atributo.

1. Objetivos y Restricciones Arquitectónicas

## Priorización de requerimientos

### Requerimientos Funcionales

| **Código** | **Requerimiento** | **Descripción** |
| --- | --- | --- |
| RF-01 | Autenticación de usuario | La aplicación debe permitir que los usuarios inicien sesión utilizando su correo electrónico y contraseña, verificando sus credenciales para acceder al sistema. |
| RF-02 | Registro de Usuario | La aplicación debe permitir a los nuevos usuarios crear una cuenta ingresando sus datos personales como nombre, correo electrónico, contraseña y número de contacto. |
| RF-03 | Registrar veterinario | La aplicación debe permitir que se registren veterinarios autorizados, asociándolos a un perfil especial con permisos definidos. |
| RF-04 | Registrar reporte de mascota | La aplicación debe permitir a los usuarios registrar un reporte de mascota, incluyendo nombre, tipo, descripción, imagen, contacto y ubicación. |
| RF-05 | Ver mascotas reportadas | La aplicación debe mostrar un listado con todos los reportes de mascotas que necesiten ayuda, ordenados por fecha o cercanía geográfica. |
| RF-06 | Chat entre usuarios | La aplicación debe permitir que los usuarios se comuniquen mediante un chat interno para coordinar el reencuentro de mascotas. |
| RF-07 | Panel de Usuario | La aplicación debe contar con un panel donde el usuario pueda gestionar su información personal, ver sus reportes registrados, historial de chats, y acceder a funcionalidades principales. |
| RF-09 | Panel de Veterinario | La aplicación debe ofrecer un panel exclusivo para veterinarios donde puedan consultar reportes relevantes, validar información y apoyar en la verificación de mascotas. |

## 

### Requerimientos No Funcionales - Atributos de Calidad

| **Código** | **Requerimiento** | **Descripción** |
| --- | --- | --- |
| RNF-01 | Rendimiento | La aplicación debe responder en menos de 3 segundos en la mayoría de las operaciones, como listar reportes o cargar la ubicación en el mapa. |
| RNF-02 | Disponibilidad | La aplicación debe estar disponible para el usuario las 24 horas del día, excepto durante mantenimientos programados. |
| RNF-03 | Usabilidad | La interfaz debe ser intuitiva y amigable, permitiendo que usuarios sin experiencia técnica puedan utilizar todas las funcionalidades sin dificultad. |
| RNF-04 | Portabilidad | La aplicación debe ser compatible con dispositivos Android desde la versión 8.0 en adelante. |
| RNF-05 | Seguridad de datos | La información del usuario y de los reportes debe almacenarse de forma segura, utilizando encriptación para contraseñas y restricciones de acceso. |
| RNF-06 | Escalabilidad | La arquitectura del sistema debe permitir añadir nuevas funcionalidades (como notificaciones o integración con redes sociales) sin rediseños mayores. |
| RNF-07 | Mantenibilidad | El código fuente debe estar estructurado y documentado para facilitar futuras actualizaciones y mantenimiento del sistema. |
| RNF-08 | Integración con servicios externos | La aplicación debe integrarse correctamente con APIs externas (como mapas) garantizando estabilidad y tiempo. |

## Restricciones

1. Restricciones de Plataforma

* El sistema debe funcionar en dispositivos móviles con Android 8.0 o superior e iOS 13 o superior.
* No se contempla, por ahora, una versión web o de escritorio.

2. Restricciones Organizativas

* El proyecto se desarrolla como parte de una actividad académica, lo que establece plazos concretos y un alcance controlado.

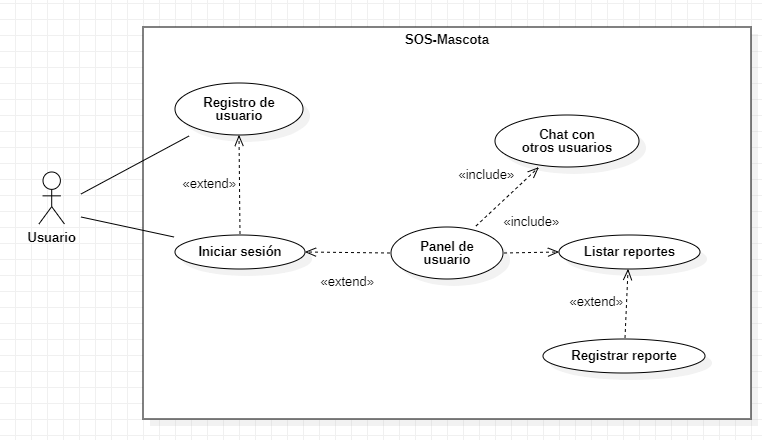
3. Restricciones de Seguridad

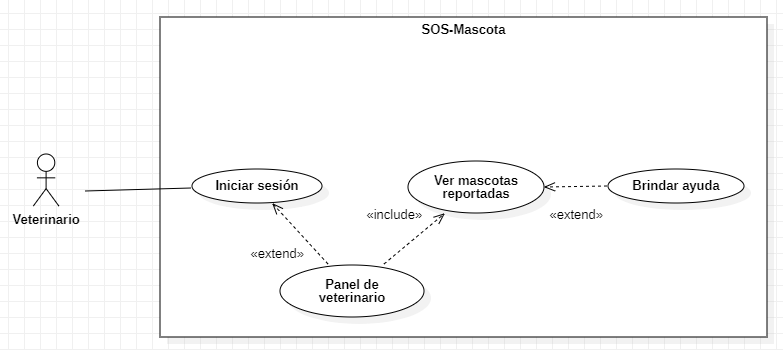
* La aplicación debe cumplir con prácticas básicas de seguridad como el hash de contraseñas, validación de datos de entrada y control de sesiones.

1. Representación de la Arquitectura del Sistema

## Vista de Caso de Uso

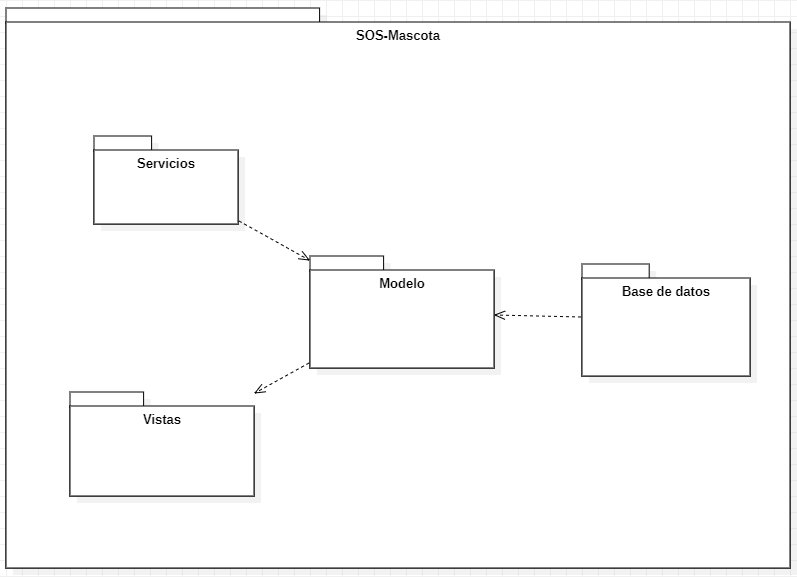
### Diagramas de Casos de Uso





## Vista de Lógica

### Diagrama de Subsistemas (paquetes)

**

### 

### 

### Diagrama de Clases

.

