Aplicación móvil para seguimiento de vacunas y procedimientos con lectura de código QR en la identificación de mascotas

Directores:
Dianalin Neme Prada
Ivan Rodrigo Romero Florez

Daniel Camilo Rubiano Rojas
Daniel Armando Hernández Chiquiza
Mayo 2021

Universidad Antonio Nariño.
Ingeniería de Sistemas.
Proyecto de Grado

Tabla de Contenidos

Lista de tablas

Lista de ilustraciones

Capítulo 1

Formulación y Descripción del Problema

Autoridades de salud calculan que hay 903.573 perros en la ciudad de Bogotá y de estos 90.000 son callejeros (Concejo de Bogotá). Además Kantar World Panel dice en un estudio hecho en Colombia, que 3'692.365 de hogares tienen animales de compañía y que de esta cantidad el 60,3% tienen perro, 22,3% gatos y 17,4% tienen ambos, finalmente el ministerio de salud compartió una información en la que calculan a 794.142 entre perros y gatos vacunados en un solo semestre en Bogotá. Estos animales que más que mascotas hacen parte de nuestra familia es necesario llevar un documento donde se haga seguimiento a sus vacunas, ya que es raro que se les lleve siempre a la misma veterinaria o que la vea siempre el mismo veterinario. Al ser este documento físico o en papel, está propenso a que se pierda si no hay precaución con su cuidado, lo que puede ser peligroso para el animal porque no se tiene claro cuál es la vacuna precedente o en qué fecha fue aplicada. Además del hecho del costo que tiene para veterinarias la generación de estos carnets, en donde la cámara de comercio de Bogotá comparte que hay 1.600 personas naturales que ofrecen servicios veterinarios. Con la descripción de esta problemática se plantea lo siguiente para este proyecto:

Determinar una solución mediante infraestructura tecnológica a la dificultad de almacenar cada certificado, carnet de vacunación o procedimientos a una mascota.

Capítulo 2

Objetivos

Objetivo General

Desarrollar una aplicación móvil que permita el registro de datos de las vacunas de mascotas con el fin de facilitar la gestión y seguimiento de los datos, así como la disminución de costos de carnets de vacunación.

Objetivos Específicos

- Disponer un módulo administrativo para la gestión de procedimientos y vacunas aplicadas a una mascota por parte de las veterinarias.

- Proporcionar mecanismos basados en la lectura de codigo de barra QR para la identificación de mascotas, así como el registro de vacunas.

- Proporcionar un sistema de notificaciones push para alertar a los dueños de mascotas sobre los diferentes procedimientos que se realizan sobre su mascota.

- Definir y evaluar escenarios de calidad relacionados con mantenibilidad y usabilidad.

- Crear una estructura de aplicación de alto nivel empleando el paradigma de programación orientado a objetos que implemente en su diseño los principios para la alta cohesión, bajo acoplamiento, polimorfismo, abstracción, encapsulamiento y herencia para satisfacer los requerimientos funcionales y no funcionales que hacen parte del alcance de este proyecto.

Capítulo 3

Marco De Referencia.

Estado del arte

El análisis del estado del arte se realiza mediante una búsqueda de las aplicaciones relacionadas con animales domésticos bien sea para adopción, búsqueda de animales perdidos y/o encontrados, entrenamiento y seguimiento de sus vacunas. Con el fin de tener un conocimiento más profundo de aplicaciones relacionadas a mascotas. Para el desarrollo de este proyecto se presentan investigaciones a nivel nacional e internacional sobre esta temática:

- **Investigaciones Nacionales**

Distrito Appnimal

Distrito Appnimal es una aplicación que permite intercambiar información de animales perdidos y/o encontrados en la ciudad de Bogotá, de igual forma muestra a perros y gatos para adopción. Es una herramienta gratuita lanzada por la alcaldía de Bogotá durante el periodo de Enrique Peñalosa. Funciona con un módulo principal usado para adopciones y muestra características e imágenes de los animales. También podrá postularse como hogar de paso y hacer donaciones en especie. Tiene otro módulo de perdidos/encontrados que es una especie de muro donde hay publicaciones de animales con fotos y lugar de donde se encontró o perdió el animal. Por último tiene un módulo donde se puede encontrar información sobre guarderías, paseadores, clínicas y peluqueros para las mascotas. Es una aplicación bastante

robusta al tener tantas funciones en ella y que toma algún tiempo aprender a usarla correctamente. Pero que funciona muy bien para ayudar a personas en búsqueda de su mascota o bien sea el caso de querer adoptar una.

La Perla

La Perla es creada por la alcaldía de Medellín bajo la administración de Federico Gutierrez que permite la adopción de mascotas, muestra eventos programados para estos animales y permite publicar los extraviados. Es una app que se puede descargar para ios y android muy parecida a Distrito Appnimal pero con la diferencia que tiene menos funcionalidades y sus módulos son en estilo slide que van mostrando uno a uno los animales. De igual forma tiene un módulo de testimonios donde se puede publicar la experiencia de los usuarios al usar la app. Pero el objetivo principal es facilitar la adopción de mascotas.

Laika

En Laika productos y servicios para mascotas se puede encontrar información de donde podemos comprar todo lo relacionado a nuestras mascotas alimentos de todas las marcas, snacks, accesorios y medicina, De igual forma muestra servicios como llamada de emergencia/Denuncias, servicios veterinarios a domicilios, paseadores, seguros, transporte, medicina pre pagada, colegios o guarderías, foto studio, entrenador canino virtual, video consultas, funeraria, previsión exequial, adopción y restaurantes pet Friendly. Esta app solo se especializa en perros y gatos y da muy buenos precios y promociones en productos y servicios, además de ser a domicilio. Una muy buena opción para los usuarios a la hora de buscar nuevas alternativas y más opciones de accesorios para sus animales.

PetMaster Pro

Esta app nos permite documentar las obligaciones de nuestra mascota. Deja crear un perfil para cada animal el cual tiene información vital como nombre, especie, raza, género, día de nacimiento. Información visual en donde se podrá registrar el color del pelaje, color de ojos, peso, tamaño, largo y características especiales. Información del dueño como el día que lo compro, precio y comentarios al respecto. Dentro del perfil del animal podrá ingresar las alergias que tiene, información del seguro, condiciones médicas y necesidades especiales. En esta app podemos registrar dos animales gratis bien sea gatos, perros, hamster, pescado, ratones, pájaros, reptiles y conejos. Para agregar otro debemos pagar una membresía. Nos permite agregar información bastante importante de nuestra mascota pero no nos ofrece ese seguimiento a su vacunación.

- **Investigaciones Internacionales**

Wizapet

Wizapet es creada en España y es una app donde podemos publicar alertas y avisos de un animal (perro, gato, hurón, pájaros, reptiles y otros) perdido, encontrado o que esté en adopción. Ubicando la zona en donde se perdió o se encontró y a través de esta geolocalización las personas recibirán una alerta inmediatamente en su app Wizapet con las características y fotografías de la mascota. Todas las mascotas perdidas se geolocalizan en el mapa y la persona que se haya encontrado su mascota se pondrá en contacto contigo a través del chat interno.

Findster Duo+

Para usar esta aplicación debemos pagar una vez y su uso es ilimitado. Mediante un localizador GPS adherido al collar de nuestro perro podemos salir a pasear con él y soltarlo

sin preocupación de que se pierda, ya que en la app podremos ver su ubicación en tiempo real. Además permite guardar registros de sus caminatas y es posible ver todo esto desde diferentes dispositivos al mismo tiempo, lo que permite mayor control de sus paseos diarios bien sea que tenga un paseador contratado o que nosotros mismos lo saquemos a jugar.

Every Doggy: Dog Training

Every Doggy es una aplicación especializada en entrenamiento personalizados de perros. Se puede registrar a la mascota con su nombre, foto, edad, raza, años y además si conoce ya algunas instrucciones. Tiene más de 230 videos de lecciones donde podemos encontrar desde instrucciones básicas hasta avanzadas. Muestra diferentes juegos que podemos realizar con la mascota. Nos da dos herramienta como el clicker y susurrador que podemos cambiar su frecuencia desde 100Hz hasta 24000Hz y se usa para llamar al animal o para indicar cierta instrucción. De igual forma tiene un módulo en el que podemos preguntar a un entrenador cualquier duda que tengamos o inconveniente que estemos presentando.

11Pets

11Pets es una aplicación con muchas utilidades, diseñada para llevar registros y datos en tiempo real de nuestras mascotas. Entre sus funciones están los avisos en donde ayuda a recordar todo lo referente al cuidado de las mascotas como por ejemplo medicamentos, vacunas y desparasitación. En el módulo de Consultas se puede almacenar resultados, diagnósticos y documentos digitales como rayos x o ecografías. Otra de sus secciones es

“Agenda” en donde se puede planificar peluquería, baños, higiene de uña y oídos de nuestro amigo. Es una aplicación referente para este proyecto ya que tiene algunas de las funciones que se quieren implementar, y dando nuestro componente de innovación se logrará un concepto muy parecido a esta app.

Finalizando en el estado del arte podemos notar que existen muchas aplicaciones para las mascotas domésticas entre ellas; búsqueda de perros perdidos, ubicación del animal, comprar diferentes accesorios y comida, adopción, entre otras. Gracias a esta investigación se dio un enfoque al registro de vacunas y procedimientos realizados, si bien ya existe esta funcionalidad no está especializada, con esto nos referimos a que pertenece a una gama de opciones muy amplia donde el usuario podría perderse. Este proyecto se enfocará en facilitar esta funcionalidad al uso de los usuarios. A continuación se mostrará el impacto que se logra con esta aplicación.

Impacto

El impacto de este proyecto se verá de manera social al permitir el registro de vacunas tanto para los dueños de las mascotas como para veterinarias en una plataforma digital, debido a que es un seguimiento que se puede perder si el animal asiste a diferentes centros de salud. De manera ambiental se notara la disminución del uso de papel o cartón para llevar el seguimiento de vacunas. Un cambio muy importante al medio ambiente que hoy en día se encuentra devastado por la tala de árboles e incendios forestales. Finalmente el impacto económico que ofrece este proyecto se presenta en las situaciones donde estos documentos de vacunación se pierden. Cuando ocurre este tipo de evento es necesario volver a pedir a la veterinaria un nuevo registro que en algunos casos no será de manera gratuita,

adicionalmente, impacta de forma directa a las veterinarias al tener la información de las vacunas centralizada, porque el uso de la tecnología como acompañamiento en estos procesos permitirá que todo sea mucho más rápido, llamando la atención de más usuarios dando a su vez un impacto tecnológico en esta área.

El uso del seguimiento de vacunación tradicional que se lleva por parte de las veterinarias, presenta unos costos que se quieren mostrar para dejar ver el impacto económico que podría causar el implemento de este sistema en una veterinaria.

Descripción del costo	Valor COP
300 impresiones de carnet de vacunación para clientes de una veterinaria.	$400.000 + IVA

Tabla 1/ Costos del seguimiento de vacunas tradicional para las veterinarias. Fuente: Elaboración propia

El impacto económico sólo será percibido por las veterinarias ya que son las que realizan la inversión (Tabla 1) para otorgar los carnets que la aplicación pretende reemplazar. Los duenõs de las mascotas los percibirán mediante la facilidad de llevar el control de sus mascotas desde su smartphone y no en múltiples cartones o papeles.

Componentes de innovación

La posibilidad de contar con una plataforma en donde poder registrar y consultar los procedimientos y vacunas realizados a una mascota, ofrece una base de datos centralizada sobre las mismas, la cual facilita y mejora el tratamiento de estas. Además de notificar al usuario cuando una vacuna se haya registrado con éxito en el perfil de su mascota por parte de la veterinaria, será un diferenciador importante a la hora de hacer un ‘benchmarking’ con

otras aplicaciones móviles. Dicho esto, se presenta a continuación una serie de definiciones de las tecnologías que se implementan en este proyecto, junto al por que se van a usar.

Marco teórico

Para el correcto desarrollo y entendimiento de este proyecto es necesario tener un conjunto de conocimientos asociado al software que se explican a continuación. Además entraremos a exponer las tecnología a implementar y por que el uso de ellas.

Software

El software es un conjunto de programas de cómputo, procedimientos, reglas, documentaciones y datos que hace parte de las operaciones de un sistema de computación. Son diseñados para cumplir unas tareas dentro de un sistema (IEEE 720, 1983).

Arquitectura de Software

La arquitectura de software se puede definir como un conjunto de patrones a través de los cuales se define el código fuente. Marca unas pautas, objetivos y restricciones basándose en los riesgos que puedan presentarse durante el desarrollo y sus posibles soluciones. Determina cómo se quiere que interactúen las partes del sistema. La arquitectura de software trae muchos beneficios como el tener una base sólida y escalable para el proyecto, aumenta el rendimiento, reduce costos, reduce tiempos, optimiza la evolución del código entre otros.

El arquitecto de software es el responsable de conciliar las solicitudes de los involucrados en el desarrollo. Esta persona debe tener un papel de liderazgo, de mucha iniciativa y capaz de aprender constantemente. Sus actividades constan desde la concepción del proyecto, en una

fase de requerimientos, en el diseño, la construcción y pruebas del sistema y finalmente la liberación.

Códigos QR

De acuerdo con la definición dada por la ISO/IEC 18004:2006 el código QR es una representación bidimensional (ancho*largo) que contiene datos guardados como formas geométricas en un espacio fijo.

Un código QR (Quick Response) o de respuesta rápida, es un módulo que se usa para almacenar información en una matriz de puntos o en un código de barras bidimensional. Este código se lee desde un dispositivo móvil a través de un lector específico llamado lector QR y de forma inmediata nos muestra bien sea una página web, un mapa de localización, un correo u otra información almacenada en él.

Aplicaciones Móviles

Es un software pensado para dispositivos móviles y tabletas. Se puede decir que una app móvil es para un dispositivo móvil lo que los programas son para un ordenador. Hoy en día podemos encontrar todo tipo de aplicación para mejorar nuestra productividad, las más básicas son las alarmas, calculadoras, calendarios entre otras. Cuando Iphone ingreso al mercado se generaron nuevos modelos de negocio lo que provocó que estas aplicaciones fueran rentables para los desarrolladores y de igual manera las herramientas para el desarrollo de estas evolucionaron. Entre los mercados de app móviles podemos encontrar a App Store, Google Play y Windows Phone Store. Existen distintas maneras de desarrollar una app móvil cada una de ellas tienen sus ventajas y limitaciones. Estas aplicaciones no deben confundirse

con las aplicaciones web que se adaptan a dispositivos móviles o web responsive (debitoor, 2020).

El uso de una aplicación de este tipo para el desarrollo del proyecto es pensado gracias a la accesibilidad y rapidez con la que el usuario podrá llegar a ella. Además el uso del código de barras QR que se escaneara a través de la cámara del dispositivo.

Aplicaciones Nativas

Las aplicaciones nativas son llamadas a las desarrolladas por el lenguaje de programación específico de cada equipo, estos son llamados los Software Development Kit o SDK y tenemos uno diferente para Android, IOS y Windows Phone. Las principales ventajas de estas aplicaciones es que se adaptan perfectamente al dispositivo y puede usar todos los recursos que ofrece cada equipo móvil como por ejemplo acelerómetro, giroscopio, GPS, cámara entre otros.

Las aplicaciones para Android son desarrolladas en Java, las de IOS en Objective-C o Swift y las de Windows Phone en C#. Algunas de sus desventajas se presentan cuando queremos que nuestra app corra en todos los dispositivos móviles ya que para esto se debe tener conocimiento de todos los lenguajes mencionados o tener desarrolladores especializados en código nativo, esto genera mayor costos y toma mayor tiempo de desarrollo (José Vittone, 2013-2017).

Aplicaciones Híbridas

Las aplicaciones híbridas son desarrolladas con los lenguajes más comunes de las aplicaciones web como los son HTML, CSS y Javascript. Son esencialmente aplicaciones nativas que pueden acceder a los mismos recursos nativos como si fueran creadas con un SDK. Las principales ventajas que tienen este tipo de aplicaciones son: La velocidad, porque

al programar en un único código hace que se pueda cumplir con el objetivo de un app móvil en múltiples dispositivos 2 o 3 veces más rápido que una app nativa. El soporte es mucho más fácil para este tipo de aplicaciones ya que no es necesario tener personal especializado en código nativo. Además estas aplicaciones tienen la capacidad de correr en cualquier plataforma que corra la web como desktop, pwa, carros, móvil, etc (Yeeply, 2019).

IONIC

Ionic es un framework que ofrece un kit de herramientas para crear una interfaz de usuario móvil. Fue desarrollado sobre AngularJS y Cordova con la intención de poder crear aplicaciones híbridas. Ha sido actualizada en múltiples ocasiones en donde en su más reciente anuncio informan la incorporación de Capacitor un Native API Container que entra a reemplazar a Cordova (IonicFramework, 2020).

El uso de este framework para el proyecto se debe a la necesidad de crear una app que corra en diferentes dispositivos móviles tanto en Android como en IOS. Además de esto es un framework que ofrece gracias al marco de desarrollo móvil ahora con Capacitor, todos los plugins disponibles en Hardware, Software, Imágenes, Textos, Códigos QR, etc. Al ser el desarrollo de este proyecto por capas esta sería la capa de más alto nivel.

NodeJS

Como dice Desarolloweb(2020), NodeJS se puede definir a una tecnología que funciona como una plataforma de ejecución de javascript o ‘runtime’ sobre el cual se puede ejecutar cualquier tipo de programa. En otras palabras, nodeJS es el lenguaje javascript fuera del navegador. En esta tecnología se pueden construir páginas web, API Rest, aplicaciones de escritorio y programas de consola.

En este proyecto el uso de NodeJS se implementa para conectar la capa de más alto nivel en IONIC y la capa de bajo nivel hecho en Spring Boot. De esta manera aplicando esta arquitectura se tendrá menor acoplamiento y mayor cohesión en el código. Delegando una responsabilidad única a cada capa.

Spring Boot

Spring Boot es una infraestructura ligera que elimina la mayor parte del trabajo de configurar las aplicaciones basadas en Spring. En este proyecto se usará esta tecnología para el desarrollo del backend que funcionará como una capa más, la cual se encargará de llamar la información a la base de datos Firebase. (Schuurman, 2017).

Firebase

Firebase es una plataforma creada por Google para el desarrollo de aplicaciones web y móviles. Es una base de datos muy parecida a las noSQL o no relacionales que funciona como un servicio en la nube. Trabaja como un Realtime Database esto quiere decir que firebase nos envía nueva información cada vez que un cliente guarda un cambio y este cambio se recibe de manera casi instantánea. Permite guardar cualquier tipo de archivos y tiene un sistema de autenticación con email y contraseña, Google, Facebook o GitHub(Esplin, 2016).

Esta plataforma se implementará para centralizar la información recibida (datos de usuario, de mascotas e información relacionadas a las vacunas). La capa encargada de enviar y recibir esta información será desarrollada a través del framework Spring Boot el cual tendrá que enviar estos datos a NodeJS que finalmente irán a la capa de presentación en IONIC para mostrarlos en la interfaz de usuario.

Capítulo 4

Metodología

Se emplea la metodología ágil “Scrum” para la elaboración del proyecto. Se realizarán diferentes iteraciones para conseguir de manera gradual y progresiva la totalidad del proyecto consiguiendo en cada iteración propuesta de valor.

Como menciona Abellán(2020), Scrum es una metodología o framework empleado dentro de equipos que ejecutan proyectos desde básicos a complejos. Dicho de otra forma, es una metodología de trabajo ágil cuya finalidad o meta es la entrega de valor en lapsos cortos de tiempo, para poder cumplir esto se basa en tres pilares: la transparencia, inspección y adaptación. Lo anterior le permite al cliente, junto con su equipo comercial, introducir el producto en el mercado de forma rápida y empezar a obtener ventas o ganancias.

En Scrum se manejan diferentes roles para todo el proceso de desarrollo, documentación, implementación y demás. En esta metodología destacan tres roles fundamentales, imprescindibles para la correcta ejecución de la misma, Project Owner, Scrum Master y Developer Team. En este proyecto, los roles serán ejecutados de la siguiente manera:

Project Owner: Veterinario(Por definir)

Scrum Master: Daniel Rubiano.

Developer Team: Daniel Hernández y Daniel Rubiano.

Daily: Lunes a Viernes de 5:45 pm a 6:00 pm

Al ejecutarse por iteraciones, el trabajo se divide en “sprints”, partiendo de un sprint principal donde se plantea el análisis de las historias de usuario desarrolladas, la priorización de las

mismas, división de éstas en diferentes sprints para dar paso al trabajo de desarrollo incremental. En posteriores Sprint se iniciará con el desarrollo y seguimiento de la aplicación.

Sprints

Sprint 1:

- Creación de proyectos en Spring Boot, NodeJS y IONIC.
- Creación de proyecto en Firebase.

Sprint 2:

- Creación de repositorio en GitHub
- Creación de colecciones y documentos necesarios para Login y Registro en Firebase.
- Creación de pruebas de colecciones y documentos de la base de datos en Firebase.

Sprint 3:

- Creación de EndPoints necesarios para Login y Registro en Spring Boot.
- Creación de EndPoints y conexión a Spring Boot desde NodeJs.
- Implementación de interfaz gráfica en IONIC.

Sprint 4:

- Creación de servicios en spring y node para mascotas
- Creación de pantalla de listado de mascotas en ionic.

Sprint 5:.

- Creación de servicios de QR para spring y node
- Desarrollo de pantalla de agregado adicional de mascotas en ionic.

Sprint 6:

- Creación de servicios para vacuna en spring y node y pantalla de leída de QR en ionic.

Sprint 7:

- Creación de servicios para búsquedas por dueño, mascota y qr en spring y node y creación de pantalla de listado de vacunas.

Sprint 8:

- Creación de pantalla de agregar vacuna y pantalla de perfil

Sprint 9:

- Pruebas

Capítulo 5

Proceso de Software

En esta sección del documento se mostrará cómo está construido el sistema o software. Partiendo de los requerimientos tanto funcionales como no funcionales, mostrando diagramas UML para entrar a explicar las diferentes vistas, entre ellas, la lógica, de escenarios, de desarrollo, física y de procesos. Por último habrá un diagrama de arquitectura de alto nivel, para finalmente de esta manera darle un conocimiento detallado al lector de cómo este software se encuentra desarrollando.

Definiendo los requerimientos como una condición o necesidad que debe exhibir o tener un sistema para satisfacer un contrato, estándar, especificaciones u otra documentación formalmente establecida (IEEE) o como lo menciona Jacobson-Booch-Rumbaugh que es una condición o capacidad que debe cumplir un sistema ya que es el hilo conductor de todo desarrollo de software. Para la construcción de requerimientos se decidió tomar en cuenta una serie de buenas prácticas como tomar una primera aproximación amplia, esto ayuda a tener una visión más grande del problema que queremos solucionar y cuáles son los mejores caminos a tomar. Tratar los requerimientos como una pila priorizada, de esta forma podemos ir solucionando paso a paso nuestros requerimientos. A continuación se presentan los requerimientos funcionales que son los servicios que presta el sistema y los requerimientos no funcionales que no nos habla de lo que hace si no de cómo (Notas de clase, 2020).

Requerimientos funcionales

Como se mencionó anteriormente los requerimientos funcionales son declaraciones de los servicios que presta el sistema o la forma en que debe reaccionar a ciertos insumos. También se debe tener en cuenta que en algunos casos establecen lo que el sistema no debe hacer. Definiendo esto se presentan estos requerimientos:

Nombre	Descripción	Entradas	Salidas	Criterio de Aceptación
Registro de veterinarias y usuarios	Se debe permitir el registro tanto para veterinarias como para usuarios a la aplicación.	Nombre Correo Contraseña Tipo	Registro exitoso Registro fallido	Se debe validar el formato del correo. La contraseña debe ser mayor a 8 caracteres y permitir solo numeros y caracteres.
Ingreso de datos personales del usuario	Se debe solicitar la información básica de contacto y foto de perfil al usuario.	Ciudad Departamento Municipio Direccion de residencia/veterinaria Numero de contacto Foto de perfil	Guardado exitoso Guardado fallido	Se deben ingresar todos los datos para habilitar el guardado. Se debe validar que el número de contacto sea solo números.
Registro de mascotas	Permitir el registro de mascotas a los usuarios dueños de mascotas.	Nombre Sexo Raza Color Peso Tamaño Edad Foto	Registro exitoso Registro fallido	Solo se podrá guardar la información cuando todos los campos estén llenos.
Generar código de barras QR	Al registrar una mascota, debe generarse un código QR único para ella.	Datos de la mascota registrados previamente en la aplicación.	Código de barras QR	Se debe poder visualizar el código de barras QR de la mascota.
Registro de vacunas	El usuario debe poder registrar	Nombre Fecha	Registro exitoso Registro fallido	Visualización del registro

relacionadas a una mascota	vacunas que correspondan a una mascota previamente registrada en el sistema.	Foto etiqueta Mascota Usuario veterinaria		exitoso en el seguimiento de la mascota.
Notificación push de nuevo registro de vacuna a una mascota.	Debe llegar al usuario dueño de mascota, una notificación push indicando un nuevo registro.	Registro exitoso de vacuna.	Notificación push al usuario dueño.	Se debe visualizar la notificación push en el dispositivo del usuario dueño.
Lectura de código de barras QR	Se debe permitir que tanto un usuario dueño de mascota como veterinaria escaneen el código QR..	Código de barras QR.	Información básica de la mascota. Información de las vacunas de la mascota. Mensaje informando que no se encontró información.	Se debe visualizar toda la información de la mascota y sus vacunas.
Permitir registro de vacunas al escanear código de barras QR.	Cuando un usuario veterinaria escanea un código QR de una mascota. Además de mostrarle la información de la misma y sus vacunas, debe permitirle registrar nuevas vacunas.	Nombre Fecha Foto etiqueta Mascota Usuario veterinaria	Registro exitoso Registro fallido	Visualización del registro exitoso en el seguimiento de la mascota.

Tabla 2 / Requerimientos funcionales del sistema. Fuente: Elaboración propia

Requerimientos no funcionales

Este tipo de requerimiento trata de requisitos que no hacen referencia a las funciones específicas del sistema si no a sus propiedades y se originan de la necesidad del usuario,

políticas organizacionales, la necesidad de interoperabilidad con otros software o hardware o algún otro factor externo (Notas de clase, 2020). Teniendo en cuenta esto se presentan los requerimientos no funcionales de este proyecto:

Nombre	**Descripción**	**Entradas**	**Salidas**	**Criterio de Aceptación**
Transacciones por segundo	La aplicación debe ser capaz de procesar “n” transacciones por segundo.	Múltiples transacciones	Transacciones exitosas	Atender varias transacciones por segundo de forma exitosa
Intuitivo	El sistema debe ser de un uso fácil e intuitivo para los usuarios.	Uso de la aplicación	uso completo de una funcionalidad	Poder usar la app sin necesidad de mucha instrucción
Disponibilidad	Debe estar disponible el 99% de las veces que un usuario intente acceder.	Ingreso de usuario	Ingreso exitoso	El usuario debe poder ingresar a la aplicación el 99% de las veces que lo intenta.
Tiempo de respuesta	Toda funcionalidad y transacción de negocio de la aplicación debe responder al usuario en un tiempo inferior a 5s.	Uso de cualquier funcionalidad	Respuesta exitosa	Debe responder en un tiempo inferior a 5 seg.
Aplicación móvil Híbrida	Debe ser una aplicación móvil que funcione tanto para IOS como Android.	Único código	Compilación en los sistemas operativos requeridos.	Compilación exitosa en iOS y Android.
SCRUM	Se debe implementar la	Proyecto	Backlog y Ceremonias	Uso correcto de la metodología

	metodología ágil SCRUM para el desarrollo del proyecto.			
Tiempo de encendido o apagado	El tiempo para iniciar o apagar los servicios no debe ser mayor a 1 minuto	Encendido o apagado de servicios	Inicio o apagado exitoso	Duración del proceso de un minuto o menor.
Pruebas unitarias	Las pruebas unitarias del código deberán ser hechas en JEST (Javascript Testing).	Código sin pruebas	Pruebas unitarias	Coverage superior al 65%

Tabla 3 / Requerimientos No funcionales del sistema. Fuente: Elaboración propia

Diseño y arquitectura

A continuación se mostraran las diferentes vistas de la aplicación, desarrolladas en diagramas UML, con esto se pretende dar un entendimiento claro de cómo la aplicación se está ejecutando, exponiendo la vista de escenarios (casos de uso), lógica (diagrama de clases), procesos (diagrama de secuencia), física (diagrama de despliegue) finalmente se presenta una ilustración en la cual se podrá ver en un alto nivel la arquitectura adaptada para el desarrollo de este proyecto.

Diagrama de despliegue

Mediante este diagrama podemos visualizar la disposición física de los artefactos software de la aplicación. Se observa a groso modo un artefacto de datos(Firebase), uno de negocio(NodeJS), una interfaz de comunicación entre negocio y datos (Springboot) y uno de presentación dispuesto en dos escenarios(Ionic).

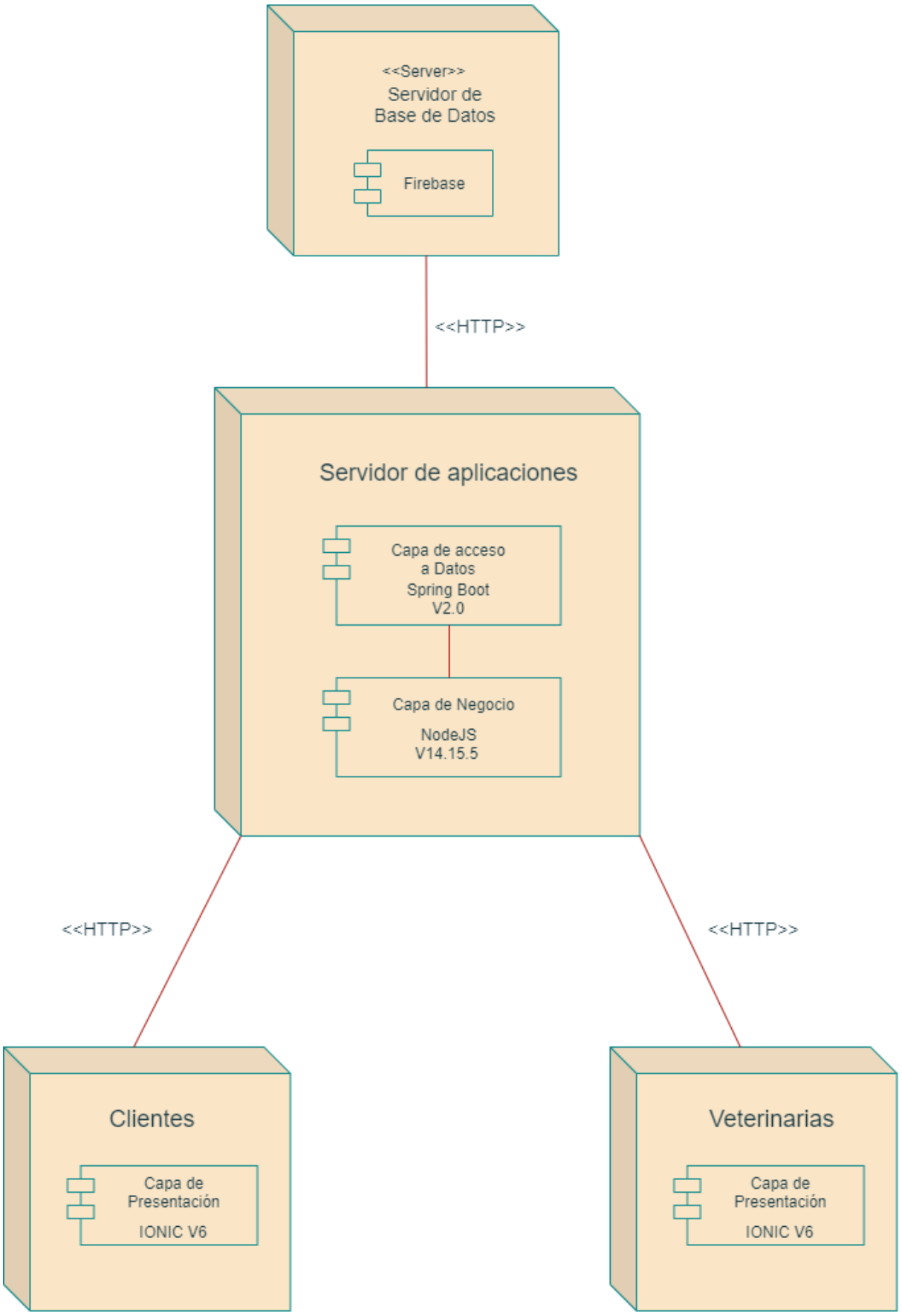

<<Server>>
Servidor de
Base de Datos
Firebase
<<HTTP>>
Servidor de aplicaciones
Capa de acceso
a Datos
Spring Boot
V2.0
Capa de Negocio
NodeJS
V14.15.5
<<HTTP>>
<<HTTP>>
Clientes
Capa de
Presentación
IONIC V6
Veterinarias
Capa de
Presentación
IONIC V6

Ilustración 1/ Vista Física, UML: diagrama de Despliegue. Fuente: Elaboración propia

Caso de uso Arquitecturalmente relevante

En este diagrama que corresponde a un modelo de requerimientos se puede ver la interacción que el usuarios y las veterinarias tendrán con la aplicación. Mostrando cómo se comunican éstos con la aplicación y como esta última se comporta internamente se quiere mostrar los diferentes escenarios que habrán en el sistema.

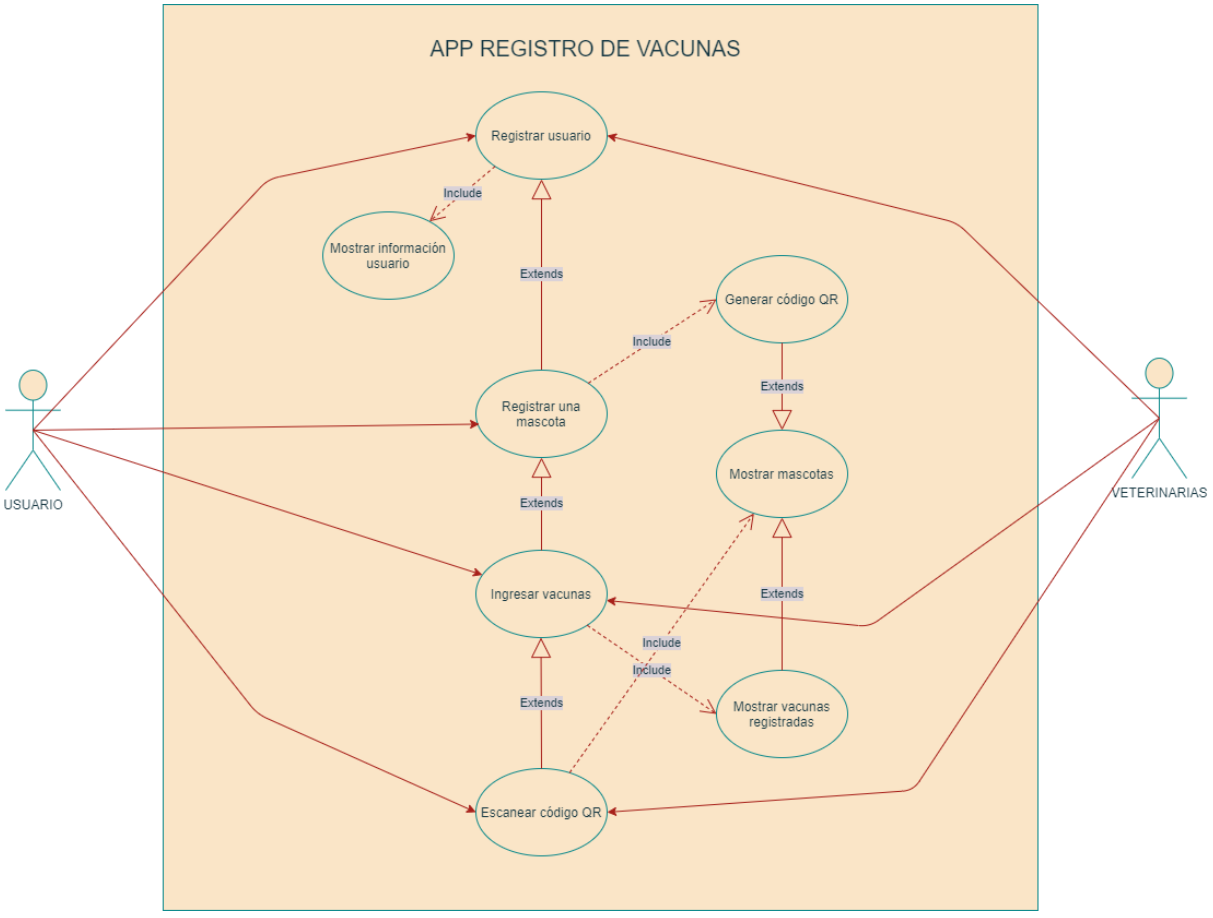


Ilustración 2 / Vista de Escenarios, UML: casos de uso y Modelo: requerimientos. Fuente: Elaboración propia

Diagrama de secuencia

El diagrama que nos permite ver la vista de los procesos que habrá en la aplicación se

muestran a continuación de la siguiente manera.

Teniendo en cuenta que existirán dos roles uno de “Usuarios” y otro de “Veterinarias” los procesos entre estas dos son muy similares exceptuando que al momento de realizar el escaneo del código de barras QR creado por la app, el rol “Veterinaria” tendrá unos privilegios que el “Usuario” no. Dicho esto se muestran la secuencia primero de los procesos que comparten estos dos roles y finalmente el proceso de escanear código QR para cada Rol por separado, dando así un mejor entendimiento al lector de este documento sobre esta vista.

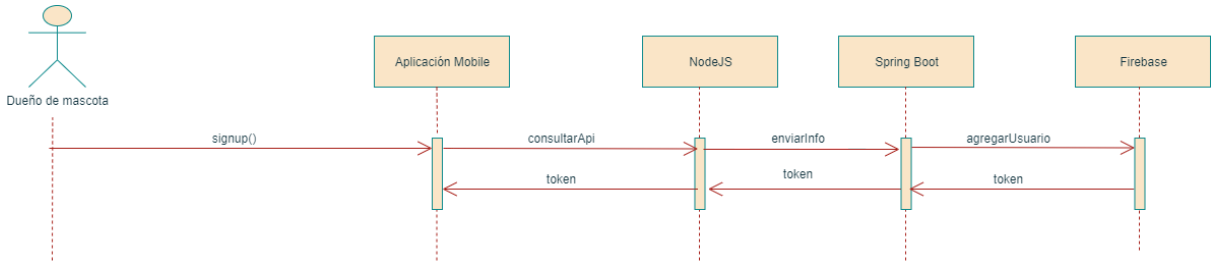


Ilustración 3 / Vista de Procesos, UML: diagrama de Secuencia 1. Fuente: Elaboración propia

La ilustración 4 nos muestra el primer proceso que deberán hacer todos los usuarios para ingresar a la aplicación. El registro de usuarios se hará cuando el usuario ingrese los datos a la app, está consultará y enviará la información a la api (NodeJS) la api se encargará de transmitir la información a spring boot que finalmente la almacenará en Firebase. Este último devolverá un token, que se enviará al spring boot y finalmente a la api para que responda a la aplicación y permita al usuario dar acceso a la misma.

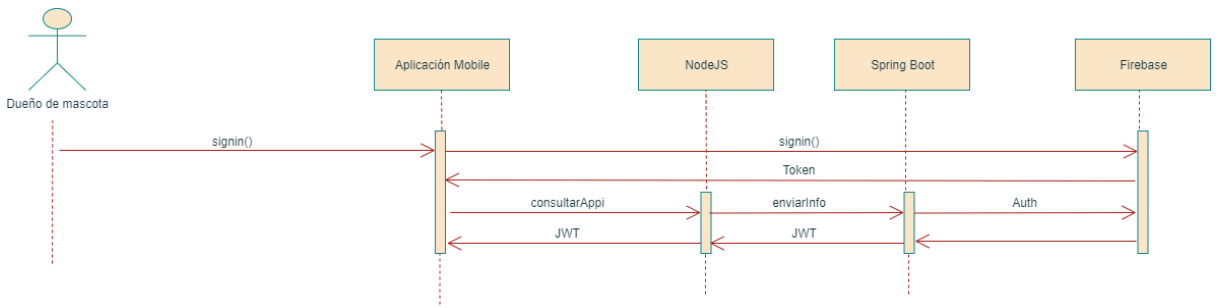


Ilustración 4 / Vista de Procesos, UML: diagrama de Secuencia 2. Fuente: Elaboración propia

El proceso que se muestra en la ilustración 5 es el login o signin que es el ingreso a la aplicación a usuarios ya registrados. Este proceso consta de unos pasos más a diferencia del registro, ya que el usuario debe pasar su información en la aplicación, este la envía a firebase que debe responder con un token y lo enviará directamente al frontend que enviará esta información a la api de igual manera se enviará a spring boot y lo que hará es enviar este token a firebase que verificará si es un token correcto lo que responderá si es o no al springboost que generará nuestro token personalizado a la api y finalmente al frontend permitiendo el acceso del usuario a la aplicación.

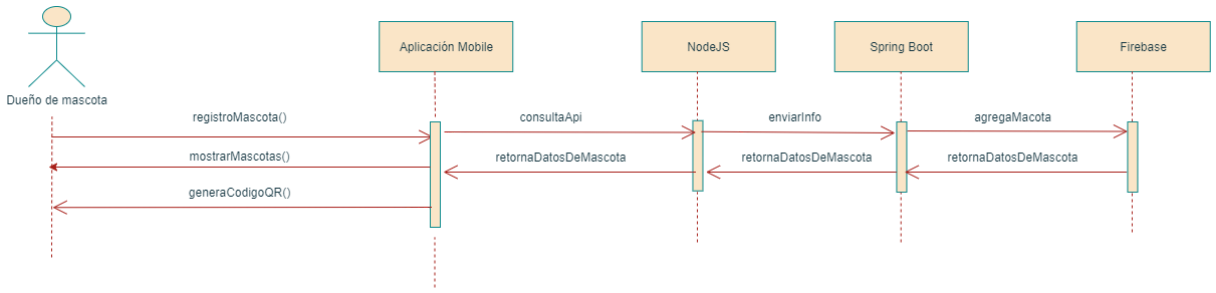


Ilustración 5/ Vista de Procesos, UML: diagrama de Secuencia 3. Fuente: Elaboración propia

En la ilustración 6 se puede apreciar el proceso de registro de mascotas. Este proceso consta de los siguientes pasos. El usuario deberá ingresar la información en la aplicación que consultara y enviará la información a la api, esta hará lo mismo al spring boot que finalmente lo guardará en firebase. Firebase dará una respuesta que enviaremos a través de todas las capas al frontend para que pueda ser usada y muestre el listado de mascotas registradas lo que me generará un código QR.

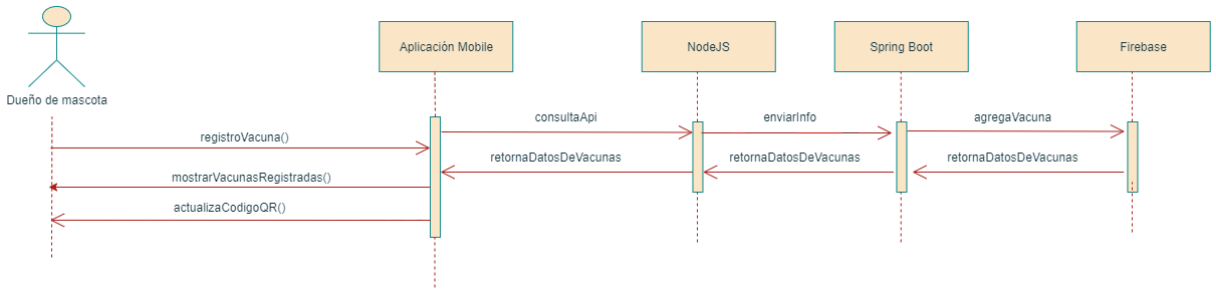


Ilustración 6/ Vista de Procesos, UML: diagrama de Secuencia 4. Fuente: Elaboración propia

De la misma manera como se hizo el proceso anterior la ilustración 7 muestra el registro de vacunas. Esta información será enviada a través de las capas a firebase para que se almacene, lo que debe retornar al frontend una respuesta que usará para mostrar al usuario el registro de vacunas que existen y actualizará el código QR que pertenece a la mascota.

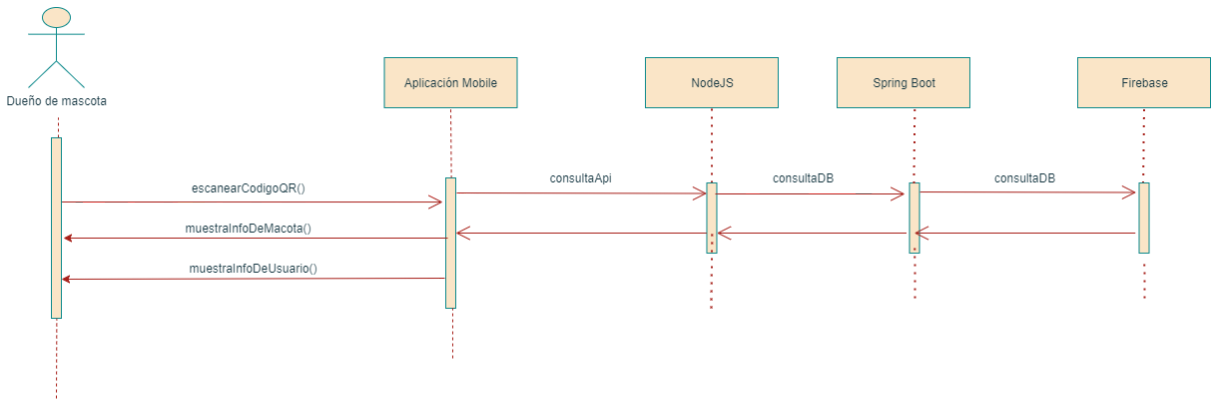


Ilustración 7/ Vista de Procesos, UML: diagrama de Secuencia 5.1. Fuente: Elaboración propia

Finalmente se explicará en las ilustraciones 9 y 10 el proceso que corresponde al escaneo del código QR. Para la primera ilustración mencionada anteriormente, vemos el proceso que tiene para el dueño de una mascota o usuario con rol “Usuario”, esta persona escaneara el código en la aplicación que enviará esta información por todas sus capas hasta consultar la base de datos en firebase. Esta devolverá la información que se retornará al FrontEnd teniendo en cuenta su rol. Si fuera rol “Veterinaria” se debe mostrar información del usuario,

información de las vacunas e información de la mascota. Y de ser rol “Usuario” dejará ver la información del usuario junto a la de la mascota.

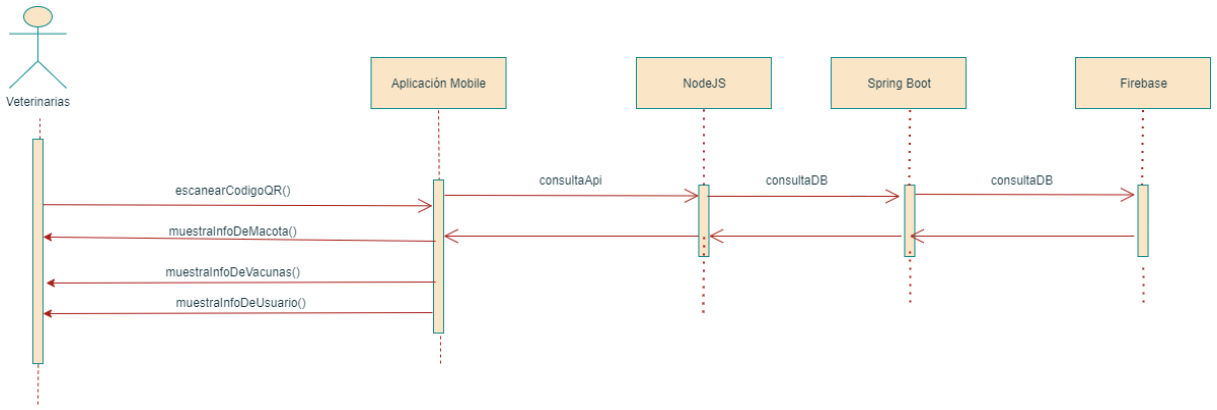


Ilustración 8 / Vista de Procesos, UML: diagrama de Secuencia 5.2. Fuente: Elaboración propia

Diagrama de clases

El diagrama de clases nos permite ver la vista lógica del sistema. Que es de tipo estructura estática y muestra las diferentes clases del sistema, atributos, operaciones y relaciones que existen entre ellas.

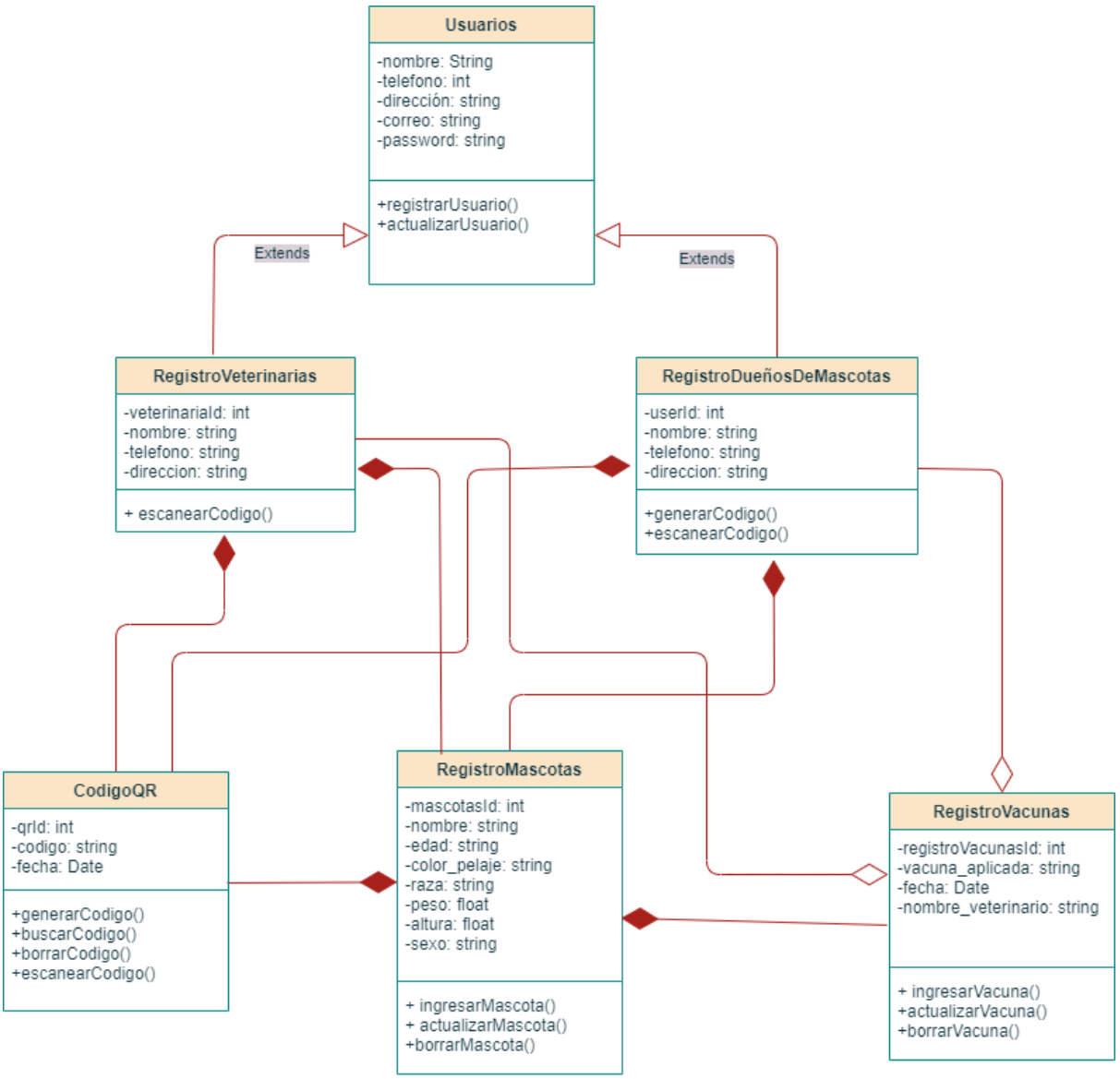


Ilustración 9 / Vista lógica, UML: diagrama de clases y Modelo: estructura. Fuente: Elaboración propia

Arquitectura de alto nivel

A continuación en la ilustración 10 se muestra un diagrama en el que se pretende exponer la arquitectura del sistema en un nivel alto. Basado en el modelo C4 para diagramar de manera efectiva, a su nivel 0.

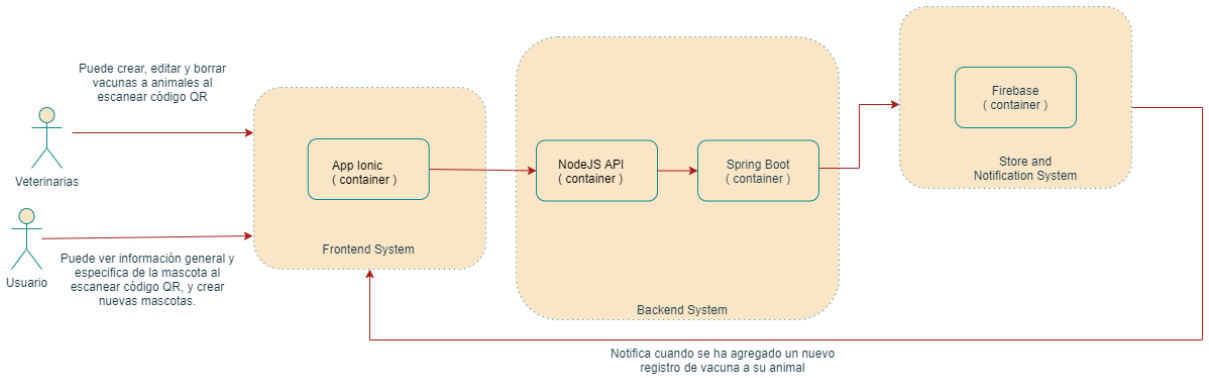


Ilustración 10 / Arquitectura de Alto Nivel. Fuente: Elaboración propia

En la ilustración anterior se puede ver los diferentes sistemas que tendrá la aplicación y los diferentes tipos de usuarios que tendrán acceso a ella. Primero se puede ver la existencia de dos tipos de usuarios que para la operación de registrar vacunas, tienen una un diferenciador. Acceden al sistema de frontend en donde se encuentra el contenedor App Ionic. Este contender se comunicará a NodeJS que finalmente hará lo mismo a Spring Boot, estos últimos pertenecientes al sistema del backend, que se comunicará a Firebase el cual notificará al sistema de frontend cualquier novedad.

Capítulo 6 Construcción

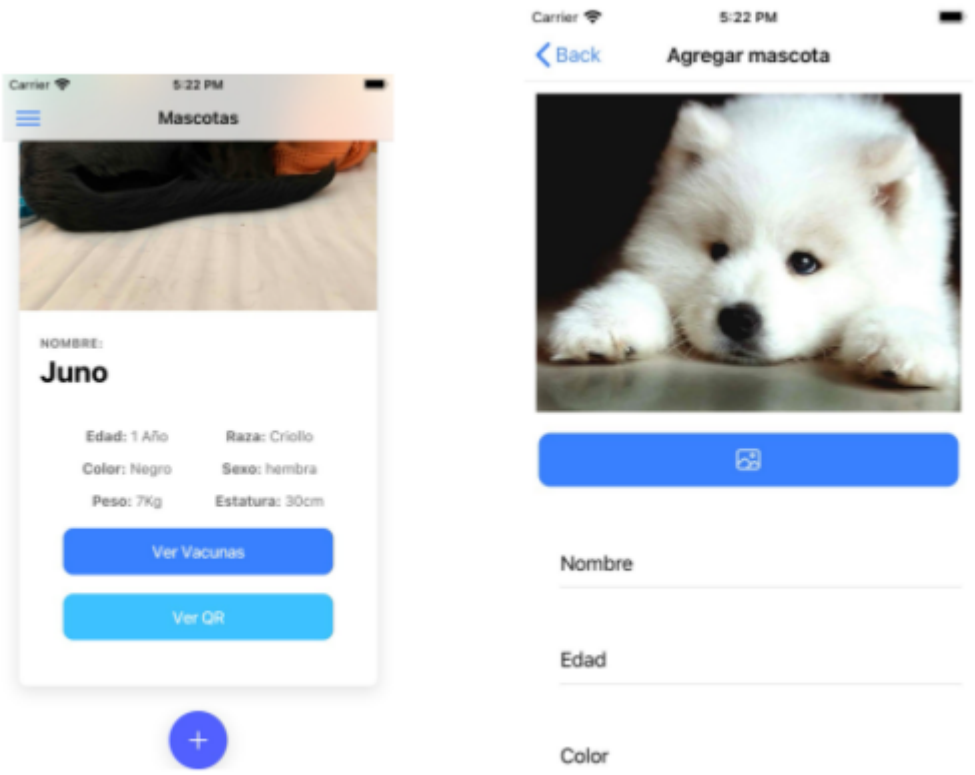


Ilustración 11 / Perfil y formulario de registro de mascota

Habiendo explicado la arquitectura que tiene el sistema. se entra a explicar el proceso de construcción. Si miramos la ilustración 1 que hace referencia al diagrama de despliegue, el proyecto se inició desarrollando el servidor de aplicaciones, en el cual encontramos: Spring Boot y NodeJS. Principalmente se creó la conexión a la base de datos (Firebase) desde Spring Boot. Se crearon los "Endpoints" para que el cliente pudiera consultarlos y obtener la información deseada, esto con la intención de hacer unas pruebas iniciales de la conexión de Spring Boot con NodeJS. Una vez teniendo esto, comienza la elaboración de la API (Application Programming Interface) desarrollada en NodeJS. Haciendo la implementación de módulos que permitieron crear el servidor al cual nuestro cliente consulta para obtener la

información.

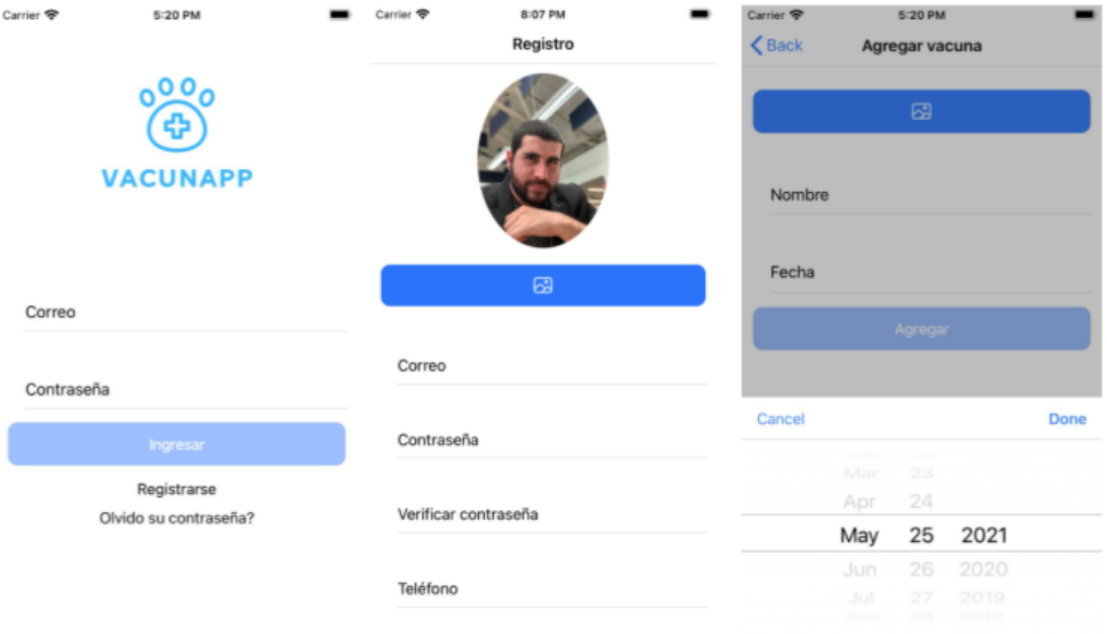


Ilustración 12 / Pantalla de login, perfil de usuario y formulario de registro de vacunas.

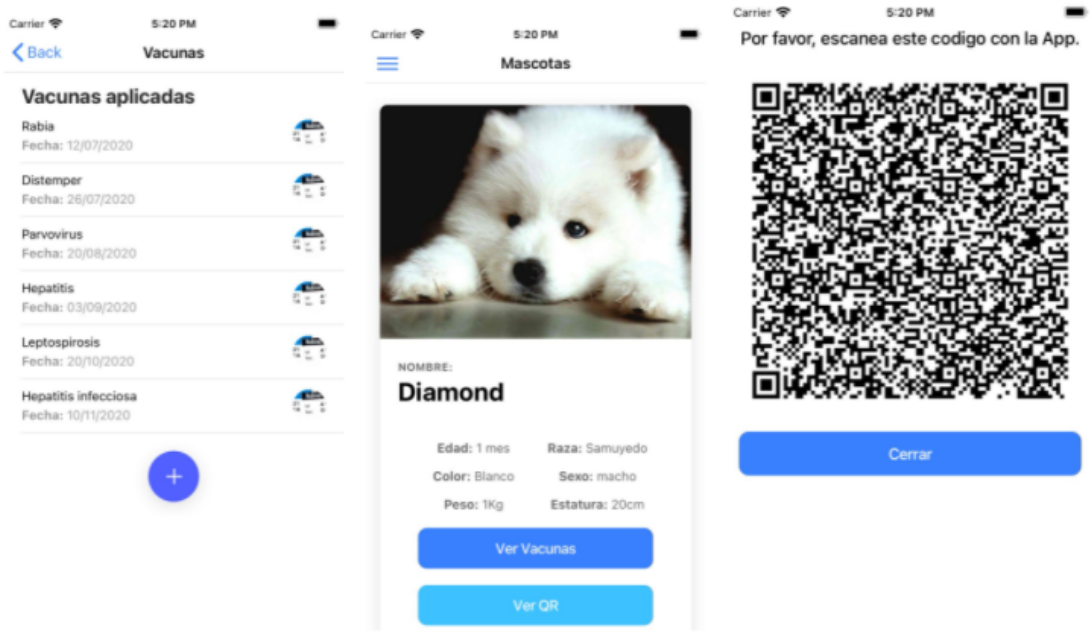
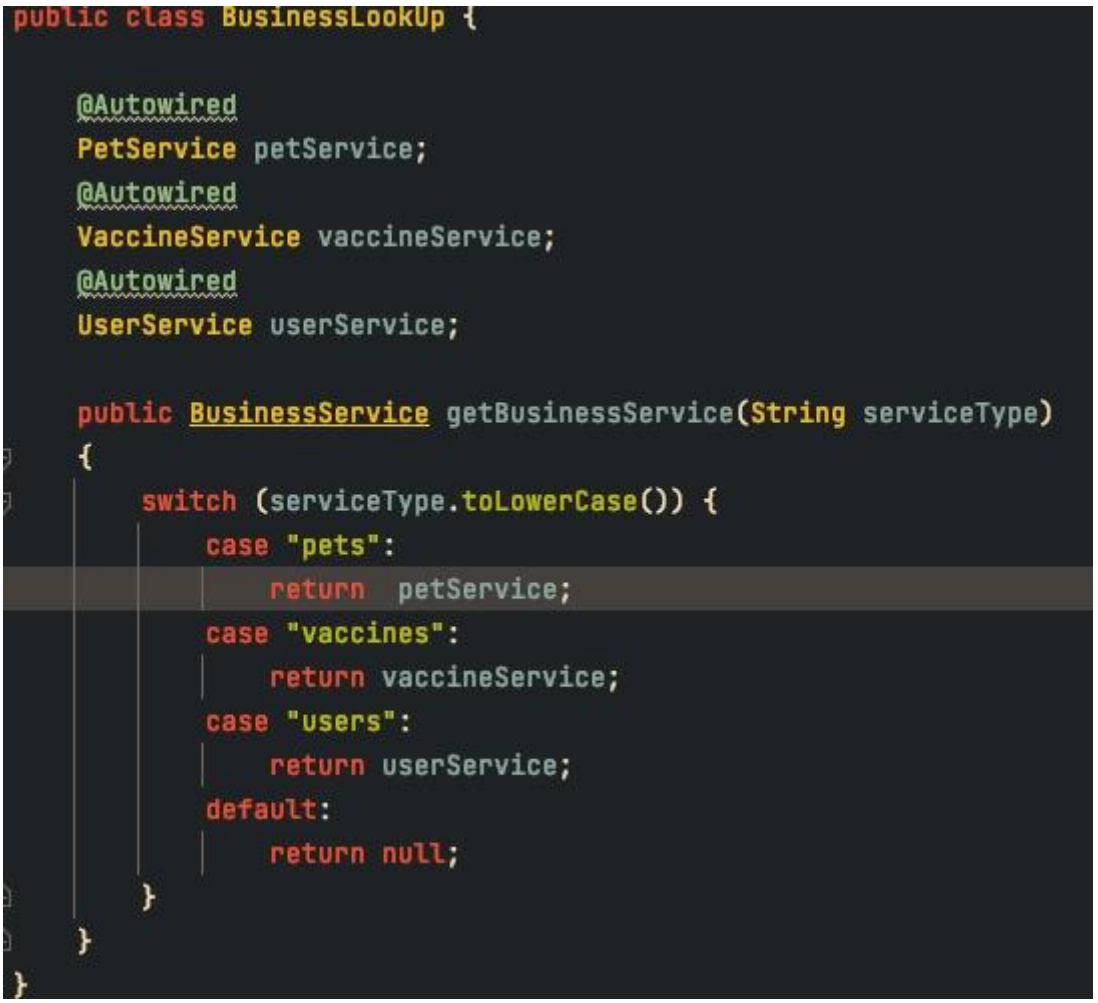


Ilustración 13 / Lista de vacunas aplicadas, perfil y código de barra QR único para esa mascota.

Backend en Spring Boot

El backend en Spring Boot busca gestionar la autenticación de la aplicación mediante JWT(oAuth 2.0) y FireBase. Adicionalmente, es el encargado en gestionar los datos de forma directa según el modelo de datos, contando con los siguientes servicios, cada uno con su CRUD respectivo.

- **Usuarios**
- **Roles**
- **Mascotas**
- **Vacunas**
- **Código QR**

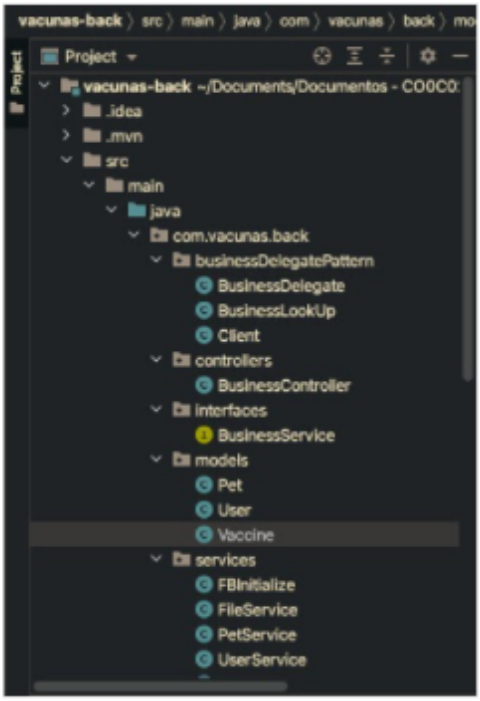


```
public class BusinessLookUp {

    @Autowired
    PetService petService;
    @Autowired
    VaccineService vaccineService;
    @Autowired
    UserService userService;

    public BusinessService getBusinessService(String serviceType)
    {
        switch (serviceType.toLowerCase()) {
            case "pets":
                return  petService;
            case "vaccines":
                return vaccineService;
            case "users":
                return userService;
            default:
                return null;
        }
    }
}
```

Ilustración 14 / Business Delegate Pattern.



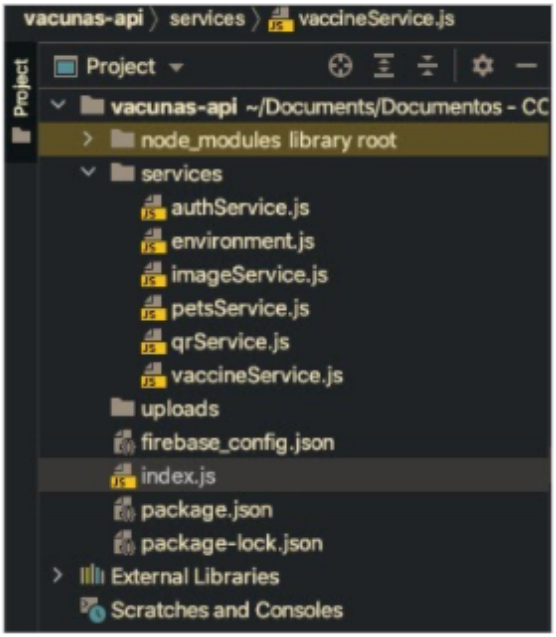


Ilustración 15 / Business Delegate Pattern.

Rest Api en Node

La Rest Api en Node(Express) se encargará de ser puente entre las consultas de la aplicación y el backend, tendrá endpoints o servicios similares al backend, más funciones especiales de búsqueda por Ids, mascotas, dueños entre otros. Adicionalmente, dará formato a la data que recibirá la App para que esta esté lista para mostrar. Ahorrando tiempo de procesamiento a nivel del dispositivo.

- **Usuarios**

- **Roles**
- **RolesUsuario**
- **Mascotas**
- **MascotasUsuario**
- **Vacunas**
- **Vacunas Mascota**
- **Código QR**
- **Código QR Mascota**

App en Ionic

La aplicación es el medio por el cual los usuarios interactúan, gestionan, modifican y ven los datos de las mascotas y sus vacunas. Contará con los siguientes módulos. Se usará un patrón Facade para implementar todos los accesos al Store de la aplicación y manejar de forma reactiva y central toda la persistencia en el lado del dispositivo.

- **Login**
- **Registro**
- **Usuario**
- **Mascota**
- **Vacunas**

Ilustración 16 / Login de la aplicación. Fuente: Elaboración propia

En la ilustración 11 se aprecia el login con el cual se dará acceso a la aplicación. En él podemos ver, en la parte superior, el logo en el cual menciona el nombre de la app (VACUNAPP). Debe ingresar el correo electrónico que fue registrado con anterioridad y su clave correspondiente.

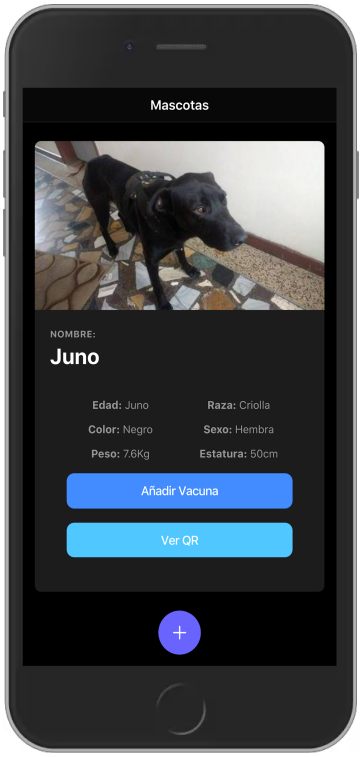


Ilustración 17 / Pantalla de mascotas del usuario dueño de mascota. Fuente: Elaboración propia

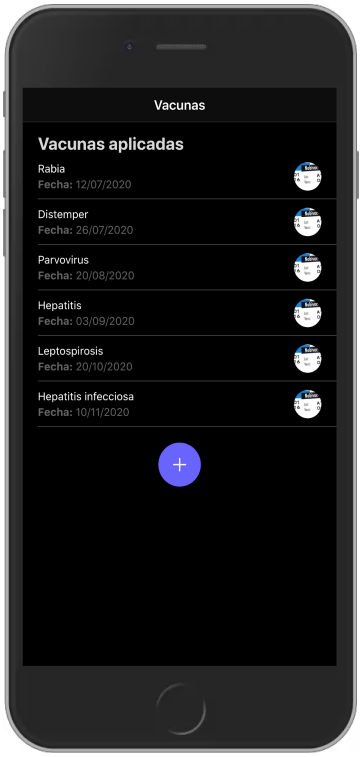


Ilustración 18 / Listado de mascotas registradas como usuario. Fuente: Elaboración propia

Metodología

La metodología implementada para el desarrollo de este aplicativo fue SCRUM en el cual se llevó el daily en un horario de 5:45 pm a 6:00 pm. Los sprints fueron programados para hacer seguimiento cada mes (Capitulo 4. Metodología. Sprints). De esta manera se llevó un control del desarrollo del proyecto durante todo el tiempo para el cual fue programado.

Capítulo 7

Calidad y Pruebas

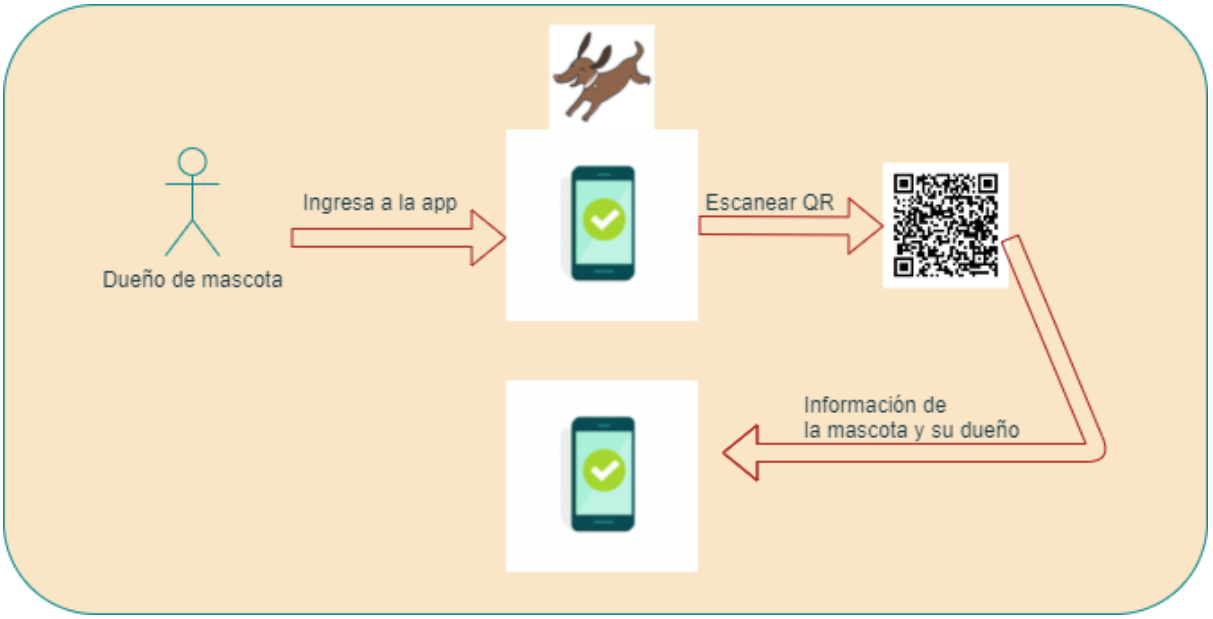


Ilustración 19 /Escenario de usabilidad al escanear código de barras QR para dueños de mascotas.

Ilustración 20 /Escenario de usabilidad al escanear código de barras QR para veterinarias.

Para la medición de la calidad del software tales como bugs, vulnerabilidades, code Smells, cobertura y la duplicación de código, se usó la herramienta Sonarcloud conectada al repositorio de GitHub en este caso del backend desarrollado en Spring Boot. Para poder dejar el código con nuestro escenario de calidad de mantenibilidad con sus propiedades analizabilidad y capacidad de ser probado.

Se creó una organización en GitHub en donde se estableció el repositorio que queremos evaluar. Una vez hay, registramos la organización y el repositorio de github en Sonarcloud. Dirigiéndonos a la página de sonar cloud evaluamos el código que hubo que realizar ciertas modificaciones como un archivo .yml y una dependencia nueva en el pom, para poderse evaluar.

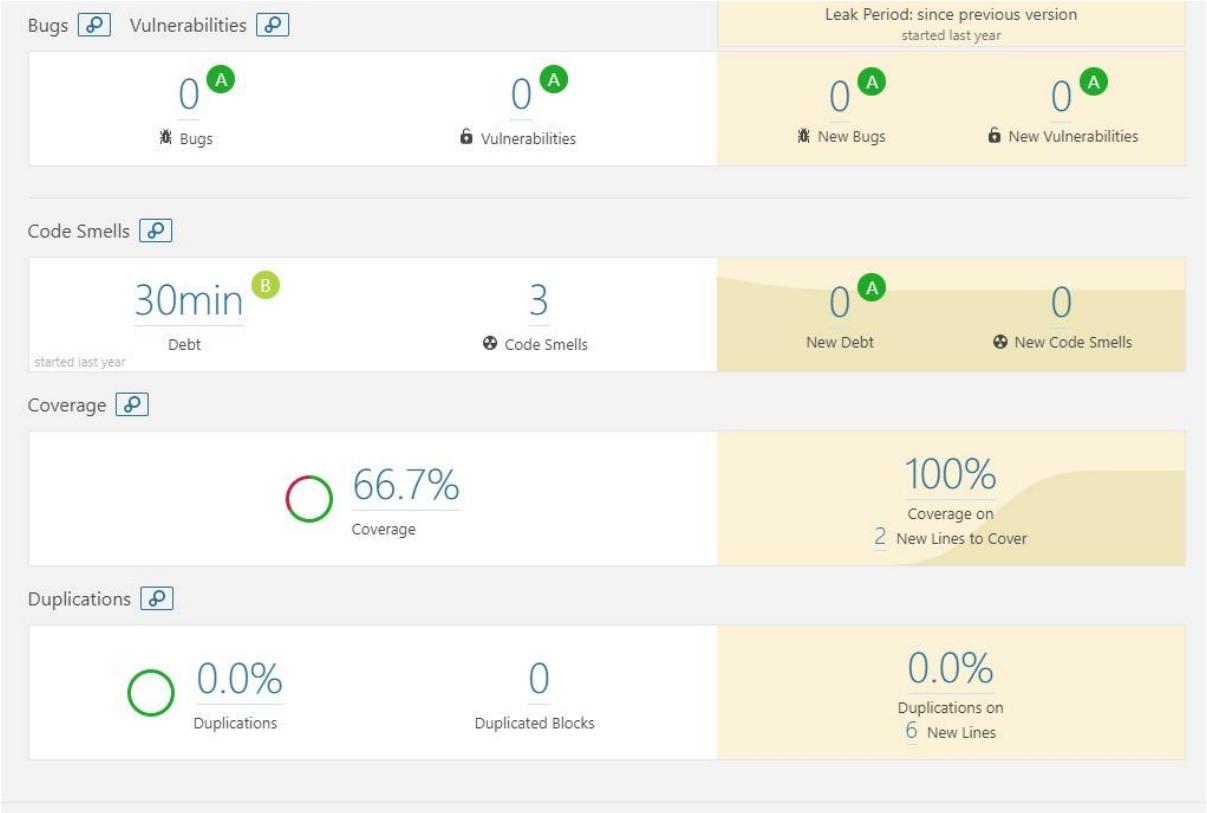


Ilustración 21 /Pruebas de calidad en Backend - Spring Boot. Fuente: Sonarcloud

Analizando la ilustración 11 vemos que en lo que respecta a bugs y vulnerabilidades se encuentra muy bien, existe una cobertura del 66.7% lo cual entra en nuestros requerimientos no funcionales. Pero de igual forma hay 30 min de deuda técnica y 3 code smells. Teniendo en cuenta esta información podemos determinar que el código es reusable ya que al no tener duplicidad será muy fácil reusar código que tengamos ya desarrollado para nuevas funcionalidades.

Capítulo 8

Instalación y Configuración

Spring Boot

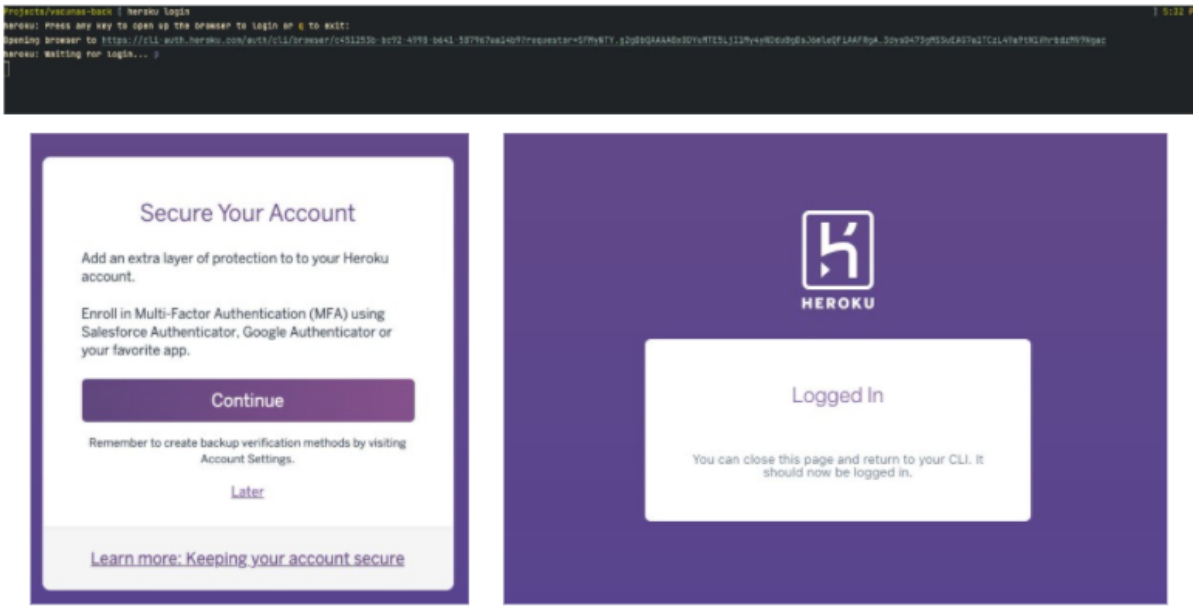


Ilustración 22 /Despliegue de Spring Boot en Heroku.

Ilustración 23 /Despliegue de Spring Boot en Heroku.

Firebase Api NodeJS

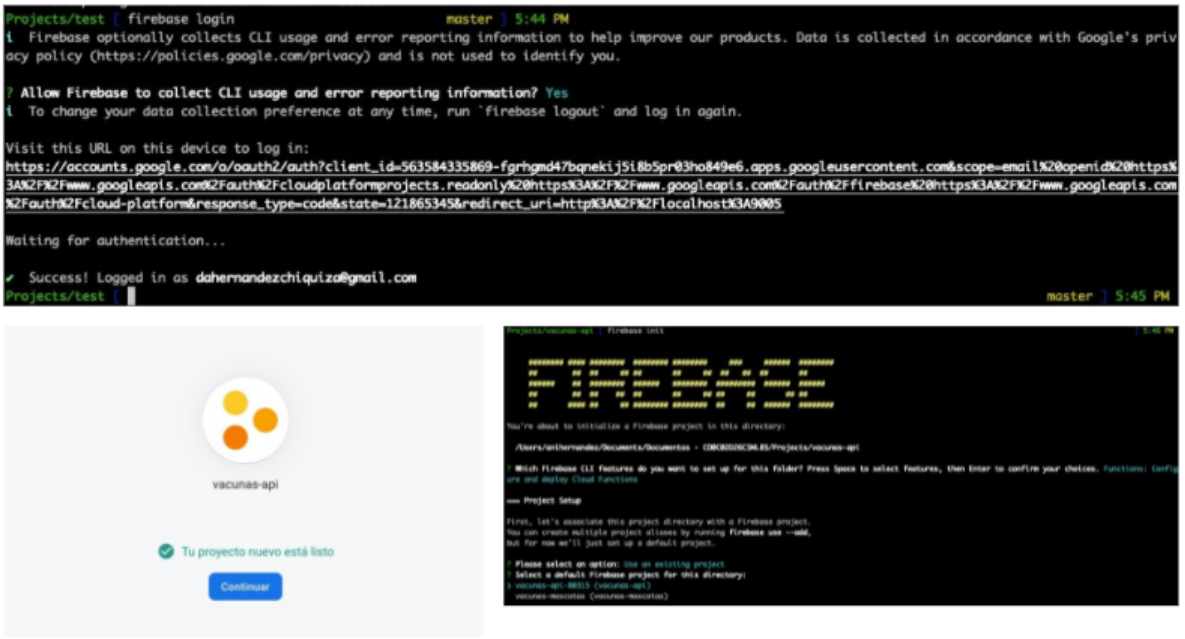


Ilustración 24 /Despliegue de NodeJS en Firebase

Ilustración 25 /Despliegue de NodeJS en Firebase

Para la publicación de la aplicación al ser híbrida tuvo dos diferentes procesos uno para Android y otro para IOS. Se pretende mostrar los dos procesos de configuración que se hicieron en IONIC. Primero debimos generar el build del proyecto. ionic build.

Para Android debimos pasar el comando ionic integrations enable capacitor, npx cap add android por consola en el directorio del proyecto. Esto me genera un archivo config.xml, que es el que nos permite hacer la configuración necesaria para hacer pruebas y ver como se muestra en las diferentes versiones de Android. Una vez hechas las configuraciones necesarias vamos a la Google Play Store Developer Console y cargamos el proyecto en Android.

Para IOS fue necesario primero conectar xcode con la cuenta de desarrollador que estábamos manejando para el proyecto. Se configura el proyecto pasando los comandos npx cap add ios, npx cap open ios.Una vez hecho esto abrimos el xcode y subimos el proyecto, realizando configuraciones que nos permitieran hacerlo.

Usuario final

Para poder usar la aplicación se debe tener un celular bien sea android o IOS. Accedemos al store según la plataforma que estemos, app store o google play store, se descarga la aplicación y una vez lista se tendrá libre acceso para usarla. El usuario deberá registrarse como requisito inicial, una vez registrado podrá realizar un login de acceso a la app. Ya dentro de la aplicación podrá registrar mascotas y a ellas sus vacunas

Capítulo 9

Conclusiones

- Ciertamente las veterinarias podrán agregar vacunas a través de un módulo administrativo que se abre al escanear un código QR, un impacto grande que tendrán las veterinarias ya que modifica todo el proceso del que se lleva actualmente de cómo registrar vacunas a una mascota, y al permitir que suba una imagen que haga

referencia a esa vacuna le dará toda la credibilidad a las veterinarias que quieran acceder a la vacunación de una mascota.

- Está claro que los usuarios al generar un código QR no solo con la información básica de la mascota si no con la de sus vacunas, dará acceso a este información de una manera rápida, fácil de llevar para estos usuarios dueños de mascotas y sobre todo no tendrán que preocuparse por algún daño o pérdida de esta información.
- Las push notification son un recurso útil cuando se trata de brindar información oportuna al usuario final, para este proyecto nos sirvió para hacer saber al dueño de una mascota cuando una vacuna es registrada a la misma, permitiéndole estar informado sobre este proceso.
- Pese a las diferentes dificultades de salud que se presentaron durante el desarrollo, se pudo demostrar que el trabajo en equipo, las habilidades de comunicación, la adaptabilidad, resolución de problemas son muy importantes para un equipo de desarrollo, ya que estas nos permiten moldearnos a diferentes situaciones y lograr que el impacto sobre el proyecto sea el menor posible.
- Finalmente a nivel profesional el desarrollo de este proyecto nos aporta conocimiento nuevo de mejores prácticas de desarrollo, testing, metodologías ágiles que nos ayudaron a crecer y dar ese paso hacia el camino de ser arquitectos de software. Las infinidades de herramientas vistas en clases durante el transcurso de la especialización favorecieron para el desarrollo de este proyecto tanto para habilidades fuertes como blandas.

Capítulo 10

Referencias

P. C. Len Bass, *Software Architecture in Practica*, 3rd ed.

Robert C. Martin, SOLID.

G. B. Raul, *El Libro Negro del Programador*

G. B. Raul, El libro práctico del programador ágil.

J. Narayanan, Elegant Software Design principles.

M. Michael, Data Structures and Algorithms with Javascript

IEEE 720, (1983). *IEEE Standard Glossary of Software Engineering Terminology*.

Debitoor, (2020). *App móvil - ¿Qué es una app móvil?*. URL https://debitoor.es/glosario/app-movil

José Vittone, (2017). *Las aplicaciones*. URL https://appdesignbook.com/es/contenidos/las-aplicaciones/

Yeeply, (2019). *¿Qué son las Aplicaciones Nativas, Web e Híbridas?*. URL https://www.yeeply.com/blog/tipos-de-app-y-para-que-sirven/

IonicTeam, (2020). *One codebase. Any platform.* URL https://ionicframework.com/

QUALITY DEVS, (2019). *Qué es Ionic y por qué te interesa conocerlo si eres desarrollador web.* URL https://www.qualitydevs.com/2019/05/31/que-es-ionic-desarrollador-web/

Schuurman Steven, (2017). *Spring Boot*. URL https://spring.io/blog

Splin Chris, (2016). *What is Firebase?*. URL https://howtofirebase.com/what-is-firebase-fcb8614ba442

Abellán Encarna. (2020). *Metodología Scrum: qué es y cómo funciona*. URL https://www.wearemarketing.com/es/blog/metodologia-scrum-que-es-y-como-funciona.html

L. Can, *Análisis y estudio del código QR y su aplicación en centros de información*

Universidad de Salamanca - Facultad de traducción y Documentación, 2015.

What is a QR Code, Unitag QR. https://www.unitag.io.

L. Can, *Análisis y estudio del código QR y su aplicación en centros de información*, Universidad de Salamanca - Facultad de traducción y Documentación, 2015.

Los códigos QR conllevan importantes riesgos para empresas y usuarios, según un estudio de Modilelron, CIBERSEGURIDAD. https://cuadernosdeseguridad.com/.

Capítulo 11

Anexos

Documento de Arquitectura de Software (SAD).