**Proyecto de desarrollo de aplicaciones multiplataforma**

**WhiskerWatch**

CICLO FORMATIVO DE GRADO SUPERIOR

**Desarrollo de Aplicaciones Multiplataforma (IFCS02)**

**Curso 2022-23**

Autor/a/es:

**Rocío Palao Fernández**

Tutor/a:

**José Luis González Sánchez**

Departamento de Informática y Comunicaciones

**I.E.S. Luis Vives**

Contenido

[1. Introducción 3](#_Toc132566039)

[2. Requisitos 4](#_Toc132566040)

[2.1 Requisitos Funcionales 4](#_Toc132566041)

[2.2 Requisitos No Funcionales 4](#_Toc132566042)

[3. Competencia 5](#_Toc132566043)

[4. Elección Tecnológica 6](#_Toc132566044)

[4.1 Creación API Rest 6](#_Toc132566045)

[4.2 Base de datos 7](#_Toc132566046)

[4.3 Front-end 8](#_Toc132566047)

[4.4 Inyección de Dependencias 11](#_Toc132566048)

## Introducción

Este proyecto principalmente se trata de una aplicación de localización de colonias de gatos callejeros.

En ella puede haber asociaciones de adopción los cuales tienen su apartado para difundirse, en el cuál pueden poner una imagen de su asociación con una descripción detallada de lo que realizan y un enlace a su página web, este dato es importante ya que esta aplicación puede ayudarles a rescatar gatos de la calle para poder buscarles una mejor vida y hogar.

Diagrama

Descripción generada automáticamentePor cada punto en el mapa de localización de las colonias tendremos un calendario de ayudas, para poner las tareas que se necesitan cierto día, en el que las personas que quieran ayudar puedan ir.

También para cada punto en el mapa tendremos un foro, por si ocurre algo en esa localización como por ejemplo un desperfecto en los bebederos de los gatos, que esté notificado en algún lugar y que también puedan poner sus opiniones sobre esa localización.

Podremos desde nuestro punto actual trazar una ruta para poder llegar a la localización de la colonia que deseemos.

Notificaciones en tiempo real cuando se realiza una recogida de colonia realizada por una asociación para buscarles un hogar.

Los usuarios se podrán registrar en la aplicación, dependiendo de si son usuarios normales o asociaciones tendrán unas características diferentes.

Todos los usuarios podrán modificar su perfil e incluso eliminar la cuenta si lo desean.

El objetivo de esta aplicación es para la ayuda y mejora de vida de los gatos callejeros.

## Requisitos

### 2.1 Requisitos Funcionales

* Log-in a la aplicación.
* Registro a la aplicación.
* Eliminación de la cuenta.
* Modificación de datos de la cuenta.
* Cambiar imagen asociaciones.
* Señalización de colonias de gatos.
* Agregar localización de colonia de gatos.
* Eliminación de localización de colonia de gatos.
* Muestra de calendario de comidas de una localización concreta.
* Agregar comida al calendario.
* Muestra de asociaciones de adopción.
* Notificación de recogida de colonia por asociación.
* El registro no es el mismo para los usuarios y las asociaciones.
* Los usuarios no pueden notificar recogidas.
* Todos los usuarios deben haberse registrado / iniciado sesión para poder utilizar la aplicación.
* Solamente las asociaciones pueden eliminar del mapa la localización por recogida de colonia.
* Solamente administradores de la aplicación pueden eliminar del mapa la localización.
* Las contraseñas deben tener un mínimo de 6 caracteres.

### 2.2 Requisitos No Funcionales

* La aplicación debe ser multiplataforma.
* Back-end desarrollado en Kotlin.
* La seguridad del back-end con JWT Bearer.
* Notificaciones de recogidas en tiempo real.

## Competencia

Esta aplicación es sobre una especie muy concreta, de competencia he encontrado la siguiente información:

**Vaca de pie sobre superficie terrosa

Descripción generada automáticamente con confianza mediaDigitanimal**

Trata sobre un sistema de GPS para tener el ganado localizado en todo momento, para ello cada animal lleva un dispositivo atado a el cuello con el cual se puede saber en todo momento la localización del animal.

Con un recinto virtual en el que se indica por donde se podrán mover los animales y cuando estos pasan de ese límite el usuario recibe una alerta en la aplicación.

Además cada dispositivo nos muestra la información del animal que lo lleva puesto, nombre, sexo, raza, etc...

<https://digitanimal.com/ganaderia-extensiva/>

Imagen que contiene Interfaz de usuario gráfica

Descripción generada automáticamente

**Animal Tracker**

Esta aplicación es para que los usuarios puedan seguir la pista a la fauna y puedan ayudar a la comunidad científica consiguiendo imágenes de los ejemplares de animales, no son todos los animales ni todas las especies, sino unos animales de ciertas especies a los que se les ha colocado un GPS.

También te muestra información del animal y la ruta que toma a lo largo de un año.

Además también puedes marcar animales como favoritos y luego ver tu lista de favoritos.

<https://www.greenappsandweb.com/android/pista-a-la-fauna-animal-tracker/#:~:text=Animal%20Tracker%20es%20una%20aplicaci%C3%B3n,momento%20est%C3%A1n%20incluidas%2017%20especies>).

## Elección Tecnológica

### Creación API Rest

Para la creación de la API Rest tenemos dos opciones:

#### 4.1.1 Spring

Es el framework más utilizado en Java.

* Ventajas:

- Utiliza el patrón de inyección de dependencias.

- Está dividido en módulos que nos ayudan a gestionar diferentes necesidades.

- Permite el desarrollo de aplicaciones flexibles y escalables, por lo que el mantenimiento del programa siempre será más fácil.

- Está basado en la programación orientada a interfaces.

* Icono

  Descripción generada automáticamenteDesventajas:

- Cuando hay actualizaciones del framework cuesta mucho trabajo migrarlo a la siguiente versión.

- Es complejo para poder utilizarlo.

- Si no tienes experiencia, cuesta un poco aprenderlo.

- Utiliza muchos XML.

#### 4.1.2 Ktor

Framework para crear servidores y clientes asíncronos creado por JetBrains.

* Ventajas:

- Código abierto.

- Fácil de empezar con Ktor.

- Utiliza DSL.

- Ligero.

- Flexible.

- Uso de corrutinas.

- Buena documentación.

* Desventajas:

- Solamente se puede utilizar el lenguaje Kotlin.

- Muy nuevo.

Para la creación de la API Rest he escogido ktor.

Porque es ligero, flexible y también es multiplataforma, usa el lenguaje kotlin y es fácil de empezar a utilizarlo.

Además es un framework que he utilizado anteriormente y me gustaría poder mejorar y descubrir más sobre él.

### Base de datos

#### MySQL

Es una base de datos relacional, actualmente uno de los más utilizados y de código abierto.

* Ventajas:

- Es multiplataforma

- Requiere de pocos recursos (Hardware)

-Tiene un buen rendimiento en volúmenes de datos medianos o pequeños.

Imagen que contiene dibujo, alimentos

Descripción generada automáticamente- Realizar las consultas rápidamente.

- Baja probabilidad de corromper los datos.

- Esquema fijo.

- Seguro.

- Soporte de transacciones.

* Desventajas:

- En las modificaciones de las estructuras de la BDD pueden aparecer pequeños errores.

- Mal rendimiento con cantidades de datos muy grandes.

- No es intuitivo, se necesita un estudio previo para poder entenderlo.

#### MariaDB

Es una versión mejorada de MySQL

* Imagen que contiene Forma

  Descripción generada automáticamente Ventajas:

- Mejor rendimiento con cantidades de datos más grandes.

- Mas seguridad de MySQL.

- Código abierto.

- Es más fácil de usar que MySQL.

* Desventajas:

- No es compatible con todas las aplicaciones.

- Poca ayuda de soporte.

#### Postgresql

* Logotipo, nombre de la empresa

  Descripción generada automáticamenteVentajas:

- Gratuito.

- Sencillo de utilizar.

- Multiplataforma.

- Gran rendimiento.

- Es un sistema robusto y fiable.

- Escalable.

* Desventajas:

- Rendimiento lento para bases de datos pequeñas.

- No tiene buen soporte técnico.

- Necesario tener una buena base de el lenguaje SQL.

#### MongoDB

Es una base de datos NoSQL ( no sólo SQL )

* Ventajas:

- Necesita pocos recursos.

- Código abierto.

- Buena documentación.

Logotipo, nombre de la empresa

Descripción generada automáticamente- Esquema no fijo.

- Escalable.

- Flexible por su buena adaptación a evoluciones.

- Optimización en las consultas con grandes cantidades de datos.

- Basado en JavaScript

* Desventajas:

- No tiene joins para las consultas.

- No admite todas las consultas SQL.

- No es correcta para las transacciones complejas.

Para el proyecto al final he elegido MongoDB

Ya que es escalable por lo que cuando aumenten los datos de localización no habrá tantos problemas, son más rápidas, son flexibles ya que las bases de datos NoSQL no siguen ningún esquema y tiene una alta disponibilidad por lo que cuando haya muchos accesos a la base de datos el rendimiento no se verá tan afectado como podría estarlo con otras bases de datos.

### Front-end

#### Flutter

* Ventajas:

- Rápido desarrollo.

- Recarga de los cambios en caliente.

Logotipo, nombre de la empresa

Descripción generada automáticamente- Multiplataforma.

- Utiliza librerías de widgets.

- Rendimiento nativo.

- Open Source.

* Desventajas:

- Lenguaje de programación Dart.

- Pesa más que otros frameworks o aplicaciones nativas.

- Puede haber dificultad para integrar en el proyecto librerías nativas de terceros.

- Limitación de integración con hardware.

#### Jetpack Compose

* Ventajas:

- Acelera el desarrollo de Interfaz de Usuario.

Icono

Descripción generada automáticamente- Adaptabilidad del dispositivo.

- Se usa menos código.

- Liberia independiente.

- Compatible con código XML.

- Compatible con todo el código existente.

* Desventajas:

- Problemas con el tiempo de compilación en la interoperabilidad.

- Problemas con el tamaño de aplicación en la interoperabilidad.

- Al ver la vista previa siempre tiene que renderizar.

- Algunos componentes no son compatibles.

- Más lento que XML para representar un cambio.

#### Angular

* Una señal de alto

  Descripción generada automáticamente con confianza bajaVentajas:

- Utiliza TypeScript.

- Desarrollo rápido.

- Tiene compatibilidad para el desarrollo de aplicaciones móviles o de escritorio.

- Desarrollo web más rápido.

- Soporte de Google.

- Pruebas sencillas de escribir.

- Productivo.

- Más opciones de lenguajes para escribir el código.

* Desventajas:

- Puede llegar a ser complejo.

- Puede tener un aprendizaje un poco difícil.

- Aplicaciones relativamente grandes.

#### VueJS

* Ventajas:

- Pesa poco.

- División en componentes individuales.

- Fácil de aprender.Logotipo, nombre de la empresa

Descripción generada automáticamente

- Fácil de utilizar.

- Reactividad.

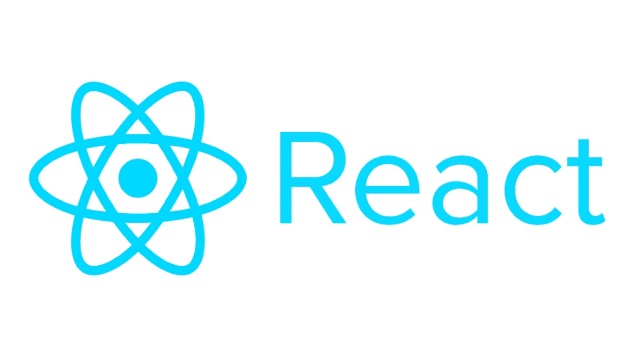
* Desventajas:

- Necesario combinarlo con Electron para hacer aplicaciones de escritorio y móviles.

#### React

* Ventajas:

- Acelera el desarrollo de las aplicaciones.

- JavaScript.

- Código abierto.

- Recarga de cambios en caliente.

- Buen rendimiento.

- Apariencia nativa.

* Desventajas:

- Problemas para depurar.

- Necesario desarrolladores nativos.

- Depende de su desarrollador Facebook.

- No muy buena gestión de la memoria.

- Actualización lenta de funciones recientes.

- Mal rendimiento con funciones complejas.

#### Ionic

* Ventajas:

- Desarrollo y despliegue híbrido.

- Diseño limpio, sencillo y funcional.

Icono

Descripción generada automáticamente- Utiliza Capacitor.

- Se puede usar con frameworks web populares.

- Fácil de aprender y utilizar.

- Muy productivo.

- Diseño de interfaz sencillo y rápido.

- Buena documentación.

* Desventajas:

- Peor rendimiento que una aplicación nativa.

- Pesa más que las aplicaciones nativas

/////////

### 4.4 Inyección de Dependencias

#### Koin VS Dagger

Dagger es más eficiente que koin ya que genera el código necesario para la inyección de dependencias en tiempo de compilación mientras que koin lo genera en tiempo de ejecución. Por lo que las aplicaciones con koin tardarán más en ejecutarse.

Dagger es más complejo de aprender ya que es un inyector de dependencias y necesita mucha configuración antes de empezar a usarlo, mientras que koin es un localizador de servicios y es mucho más fácil de aprender.

En un localizador de servicios cuando tenemos nuevas dependencias, tendremos que añadir manualmente como se debe generar la dependencia, mientras que en un inyector de dependencias al principio es muy complejo pero el resto del tiempo no tendremos que tocar casi nada de las clases que inyectan las dependencias.

He elegido Koin con Anotaciones porque ayuda a crear lo necesario para la inyección de dependencias muy fácilmente, se entiende, es muy rápido e intuitvo.

## Diagrama de Clases

## Diagrama de Casos de Uso