



Facultad de Estudios Estadísticos
Universidad Complutense de Madrid

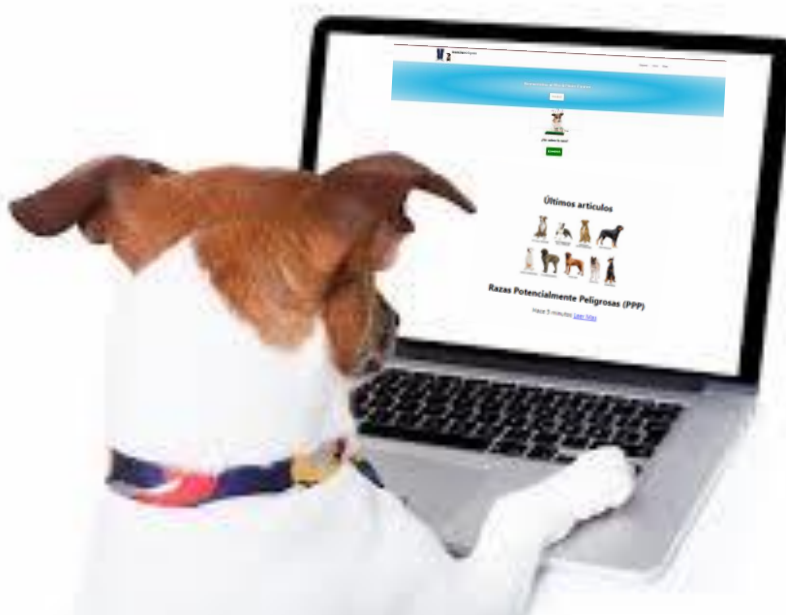


VERSION
FINAL

15-9-2020

Blockchain & BigData Canino

ANEXO IV: Api-Rest y Web



Autores:

Cristina Rodríguez Chamorro
Daniel Lanzas Pellico
Helena García Fernández
José Bennani Pareja
Juan José Lucas de la Fuente
Unai Ares Icaran

Tutor:

Sergio Torres Palomino

Documentación del TFM

Máster Blockchain y Big Data. Curso 2019-2020

ÍNDICE

Tecnologías utilizadas.....	2
MariaDB	2
Apache Tomcat	2
Java	3
AWS	3
Maven	4
HTML	4
Bootstrap	5
CSS	6
JavaScript y JQuery	6
Servidor.....	7
Representación de los diferentes recursos.....	8
Login	8
Vacuna	9
Chips	9
Perro	9
Afijos	10
Diseño y definición de URIs.....	10
Diseño de la base de datos	11
Ejemplo de llamada	12
Página web.....	14
Próximas mejoras.....	17
Actualización de JSON.....	17
Actualización de las URIs.....	18
Actualización de la base de datos	19
Bibliografía.....	20

Tecnologías utilizadas

MariaDB

MariaDB es uno de los sistemas de gestión de bases de datos relacionales existentes actualmente. Fue lanzado el 29 de mayo de 2009 desarrollado por el fundador de MySQL Michale Widenius, la fundación MariDB y la comunidad de código abierto. [1]

La base de datos ha sido alojada en Amazon Web Services utilizando el servicio RDS con una versión de MariaDB 10.4.8.



Apache Tomcat

Apache Tomcat es un contenedor de Servlets desarrollado por la empresa Apache Software Foundation, cuyo lanzamiento fue el año 1999. Actualmente, su última versión data del 13 de mayo de 2019 cuyo número es la 9.0.20. Está desarrollado en el lenguaje de programación Java, es multiplataforma y funciona sobre una JVM (Java Virtual Machine). [2]



En cuanto a los diferentes entornos del proyecto, en el entorno de pruebas, el servicio REST fue probado en tres versiones diferentes de Tomcat, la 7, 8 y 8.5. La elección final fue escoger la versión 8.5, por lo que tenemos el servidor que gestiona el servicio en AWS con una versión de Apache Tomcat de 8.5, esta versión es la única que se nos ofrece en esta plataforma.

Java

Es un lenguaje de programación orientado a objetos que desarrollado con la intención de permitir a los desarrolladores escribir una vez el código del programa y que lo ejecuten en cualquier dispositivo. [3]

Su desarrollo empezó por James Gosling que trabajaba en la empresa Sun Microsystems, en 1996 fue publicado el primer JDK 1.0 (23 de enero de 1996). Si continuamos hasta actualidad, podemos avanzar hasta Java SE 14. A lo largo de este camino Java ha ido evolucionando en casi todos los aspectos, como por ejemplo, la incorporación de los interfaces de Runnable y Callable haciendo más fácil la programación concurrente en el lenguaje, aportando soporte para las conexiones a bases de datos con JDBC (Java Database Connectivity) en el JDK 1.1... [3]



Además, Java cuenta con un recolector de basura, es decir, Java runtime es el encargado de gestionar el ciclo de vida de los diferentes objetos que se van creando en la ejecución de un programa. [3] En este proyecto, Java es el encargado de la programación del servicio REST completo, es decir, todo el servicio está programado en Java 8.0, gracias a las diferentes librerías utilizadas como, JAX-RS una librería para el soporte de servicios web o el conector con las bases de datos MySQL mysql-connector-java. [4]

AWS

Amazon Web Services es una plataforma de computación en la nube creada por Amazon en el año 2006. Esta plataforma nos ofrece un total de más de 160 servicios en ella, entre ellas podemos destacar, el servicio de informática que nos permite lanzar máquinas, generar servicios web, el más importante de estos servicios es el EC2. El servicio de almacenamiento nos permite almacenar diferentes tipos de archivos en la nube, cabe destacar el servicio S3. Servicios de machine learning, análisis, bases de datos, robótica, seguridad, servicios multimedia... [5]

Actualmente, Amazon Web Services cuenta con con 21 regiones como París, Tokio, Ohio, Canadá, etc. Además, de tener tres de ellas en proceso de lanzamiento en España, Osaka y Yakarta. [5]



Por otro lado, es la plataforma líder en servicios en la web por delante de Google, Microsoft, IBM. También, cuenta con grandes empresas como clientes como la Fórmula 1, Netflix, Vodafone o Adobe entre muchas otras. [5]

Para nuestro proyecto, AWS ha sido utilizado en el entorno de producción dando uso a los servicios EC2 para lanzar una máquina en la nube, el servicio Elastic beanstalk para lanzar un servidor Apache Tomcat y el servicio RDS para crear una base de datos MariaDB.

Maven

Maven ha sido el software de gestión de paquetes utilizado en Eclipse para desarrollar el servicio REST. Fue desarrollado en el año 2002 por Jason Van Zyl. Actualmente, es un proyecto de Apache Software Foundation. [6]

Este utiliza un archivo llamado POM (Project Object Model) que es el encargado de gestionar las dependencias de los diferentes módulos y encargada de construir la aplicación. Gracias a este archivo podemos añadir todas las librerías que necesitemos en un instante, solo deberemos añadir la etiqueta <dependencies> y dentro de ella añadir las dependencias que queramos añadir con el siguiente formato:

```
<dependency>
  <groupId> </groupId>
  <artifactId> </artifactId>
  <version></version>
</dependency>
```

Una vez añadido, Maven se encargará de descargar las librerías necesarias y añadirlas al proyecto. Además, cuenta con su propia página web donde encontrar las dependencias rápidamente (<https://mvnrepository.com/>).

HTML

HTML (HyperText Markup Language) es un lenguaje de programación orientado a la creación de páginas web. Es un estándar del consorcio World Wide Web (WWW). [7]

Este fue lanzando en el año 1993, sin embargo, su desarrollo comenzó en el año 1991 donde Tim Berners-Lee describió casi 20 elementos en el diseño inicial de HTML, actualmente 13 de estos casi 20 elementos que describió



siguen estando en el lenguaje. [7]

HTML es un lenguaje formado por etiquetas donde tenemos etiquetas de apertura `<p>` y etiquetas que nos indican el cierre de dicha etiqueta `</p>`. Actualmente, se encuentra en la versión conocida como HTML5 lanzada el 28 de octubre de 2014. [7]

En HTML5 encontramos una gran variedad de etiquetas para la definición de la página web divididas en diferentes secciones las cuales son, elemento raíz, metadatos del documento, scripting, secciones, agrupaciones de contenido, semántica a nivel de texto, ediciones, contenido incrustado, datos tabulares, formularios y elementos interactivos.

Las dos etiquetas más importantes y que no deben faltar en ninguna página web son las etiquetas: `<!doctype html>` y `<html>`. La primera define que el documento está bajo el estándar de HTML5 y la segunda representa el elemento raíz de un documento HTML. Todos los elementos que compondrán la página web deben estar dentro de las etiquetas de apertura y de cierre de `<html>`.

Bootstrap

Bootstrap es una librería de código abierto desarrollada por la empresa Twitter cuyo objetivo es proveer diseños con diferentes características a diferentes componentes de HTML para la creación de páginas web. Esto es posible gracias a la ayuda de los lenguajes de programación web CSS y JavaScript. [28]

Esta herramienta fue lanzada el 19 de agosto de 2011 y actualmente se encuentra en la versión 4.2.1 que fue lanzada el 21 de diciembre de 2018. [8]

Además del propio Twitter, este framework es utilizado por grandes empresas como la NASA o MSNBC.

Para poder utilizar este framework, solo necesitaremos añadir tres etiquetas script y una etiqueta link a nuestra página web HTML, se indican a continuación: [9]



```
<link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css"
integrity="sha384-
ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoRxT2MZw1T"
crossorigin="anonymous">
```

```
<script src="https://code.jquery.com/jquery-3.3.1.slim.min.js" integrity="sha384-q8i/X+965Dz00rT7abK41JStQIAqVgRVzpbzo5smXKp4YfRvH+8abtTE1Pi6jizo"
crossorigin="anonymous">

</script> <script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.7/umd/popper.min.js"
integrity="sha384-
U02eT0CpHqdSJQ6hJty5KVphtPhzWj9WO1c1HTMga3JDZwrnQq4sF86dIHNDz0W1"
crossorigin="anonymous">

</script> <script
src="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.min.js"
integrity="sha384-
JjSmVgyd0p3pXB1rRibZUAYoIIy60rQ6VrjIEaFf/njGzIxFDs4x0xIM+B07jRM"
crossorigin="anonymous"></script>
```

CSS

CSS o Hojas de estilo en cascada (Cascading Style Sheets en inglés) es un lenguaje que nos permite definir y añadir un diseño gráfico a diferentes lenguajes, en nuestro caso, al lenguaje HTML. [10]

Lanzado el 17 de diciembre de 1996 por la empresa CSS Working Group y actualmente en la versión CSS3. Junto a HTML y JavaScript son los lenguajes más utilizados a la hora de crear páginas webs.[10]

CSS es una herramienta muy potente que nos permite crear una gran variedad de estilos e incluso animaciones solo con su propio código.

JavaScript y JQuery

JavaScript apareció en el año 1995 desarrollado por Brendan Eich. Este es un lenguaje interpretado y utilizado en el lado cliente. Además, nos permite interactuar con el DOM (Document Object Model), es decir, nos permite interactuar con los diferentes elementos definidos en el archivo HTML de la página web. [11]

En cambio, en este proyecto JavaScript toma un plano secundario y la librería JQuery toma el papel principal.

JQuery es una librería de JavaScript creada por John Resig y que fue lanzada en enero de 2006. Esta nos permite interactuar con los elementos de los archivos HTML de una manera más sencilla y cómoda para el desarrollador que el propio lenguaje JavaScript. [12]

Para poder utilizar JQuery necesitaremos añadir la siguiente línea a nuestro archivo HTML con la definición de la página web:


```
<script src="https://code.jquery.com/jquery-3.4.1.js" integrity="sha256-WpOohJOqMqqyKL9FccASB900KwACQJpFTUBLTYOVvVU=" crossorigin="anonymous"></script>
```

En nuestro caso, la versión utilizada ha sido la 3.4.1, una de las últimas versiones lanzadas de la librería.

Para entender el acceso más sencillo a los elementos del DOM que nos permite esta librería vamos a ver algunos ejemplos.

Si queremos acceder a un elemento del DOM con un id establecido, en JavaScript necesitaremos utilizar la siguiente línea: `document.getElementById('valorId');`, en cambio con JQuery solo necesitaremos añadir la siguiente línea: `$("#id")`.

En el caso de querer acceder a un elemento con el atributo class, en JQuery solo utilizaremos `$(".class")`, por el otro lado, para JavaScript necesitaríamos utilizar, `document.getElementsByClassName("class");`

Pero la función más importante proporcionada por JQuery, es `$(document).ready(function(){})`, esta identifica si la página está preparada para interactuar con el DOM.

Por último, JQuery contiene AJAX (Asynchronous JavaScript and XML) que nos permite realizar llamadas a servicios REST.

Servidor


Para una mayor disponibilidad del servicio, este servicio se ha introducido en Amazon Web Services (AWS), utilizando el servicio Elastic Beanstalk.

AWS nos permite elegir diferentes regiones alrededor de la Tierra, entre los que se encuentran diferentes países de Asia, diferentes estados de Estados Unidos o diferentes países de Europa. En nuestro caso, hemos elegido la región de París para obtener un mejor rendimiento en el apartado de latencia de la red.

Al utilizar el servicio Elastic Beanstalk, este asocia una instancia de EC2 donde se ejecutará nuestro servicio REST. Además, nos permite elegir varios tipos de servidores distintos para diferentes lenguajes de programación. En nuestro caso, hemos elegido la plataforma Tomcat 8.5 with Java 8, debido a que hemos desarrollado este sistema en Java.

Netcan-env-2

i-020ffaff1f49c43c7

 En ejecución

t2.micro

Representación de los diferentes recursos

Para la representación de los diferentes objetos se ha utilizado JSON por motivo de comodidad a la hora de programación.

Seguidamente se muestran los esquemas de todos los JSON definidos para trabajar en el servicio REST:

Login

Cuenta con 5 parámetros fijos para el registro del usuario en el sistema:

- Email
- Nickname
- Tipo
- Password
- Teléfono

```
{  
  "email": "",  
  "nickname": "",  
  "tipo": "",  
  "password": "",  
  "telefono": "",  
  "propietario": {},  
  "veterinario": {},  
  "federacion": {}  
}
```

Además de estos 5 parámetros, este JSON contiene también la información que vamos a introducir en el sistema según el tipo de usuario que se vaya a introducir. Contamos con 3 tipos de usuarios, veterinario (representado por una V), propietario (representado por una P) y federación (representado por una F).

El propietario está representado por el siguiente JSON:

```
"propietario": {  
  "documento": "",  
  "pais": "",  
  "tipoDoc": "",  
  "paisEmisorDoc": "",  
  "nombre": "",  
  "apellido1": "",  
  "apellido2": "",  
  "direccion": "",  
  "ciudad": ""  
}
```

El veterinario está representado por el siguiente JSON:

```
"veterinario":{
  "colegiado":,
  "nombre": "",
  "apellido1": "",
  "apellido2": "",
  "clinicaVeterinaria": "",
  "direccion": "",
  "pais": "",
  "ciudad": "",
  "fechaBaja": ""
}
```

Por último, la federación está representada por el JSON:

```
"federacion":{
  "nombre": "",
  "pais": ""
}
```

Vacuna

```
{
  "idPerro":,
  "idVeterinario":,
  "codVacuna":,
  "fechaBaja": "",
  "idProteccion":,
  "fechaBajaProteccion": ""
}
```

Chips

```
{
  "idVeterinario":,
  "codMicrochip":,
  "idPerro":
}
```

Perro

```
{
  "nombre": "",
  "sexo":,
  "idPadre":,
  "idMadre":,
  "fechaNacimiento": ""
}
```

Afijos

```
{  
  "nombre": "",  
  "idPersona1": "",  
  "idPersona2": "",  
  "idPersona3": ""  
}
```

Diseño y definición de URIs

La URI principal comenzará por el nombre principal del servicio, en este caso, se llamará como el nombre que hemos establecido al sistema, “/netcan”, seguido de “/api” para identificar la API con los diferentes métodos. Para una mayor escalabilidad del sistema, la URL principal a las funciones del sistema termina con “/v1”, ya que nos encontramos en la versión 1 del sistema REST. Por todo esto, la URI principal del sistema REST es el siguiente:

- */netcan/api/v1/afijos*
- */netcan/api/v1/afijos/baja*
- */netcan/api/v1/bigdata/image*
- */netcan/api/v1/chips*
- */netcan/api/v1/login*
- */netcan/api/v1/perros*
- */netcan/api/v1/perros/baja*
- */netcan/api/v1/vacunas*

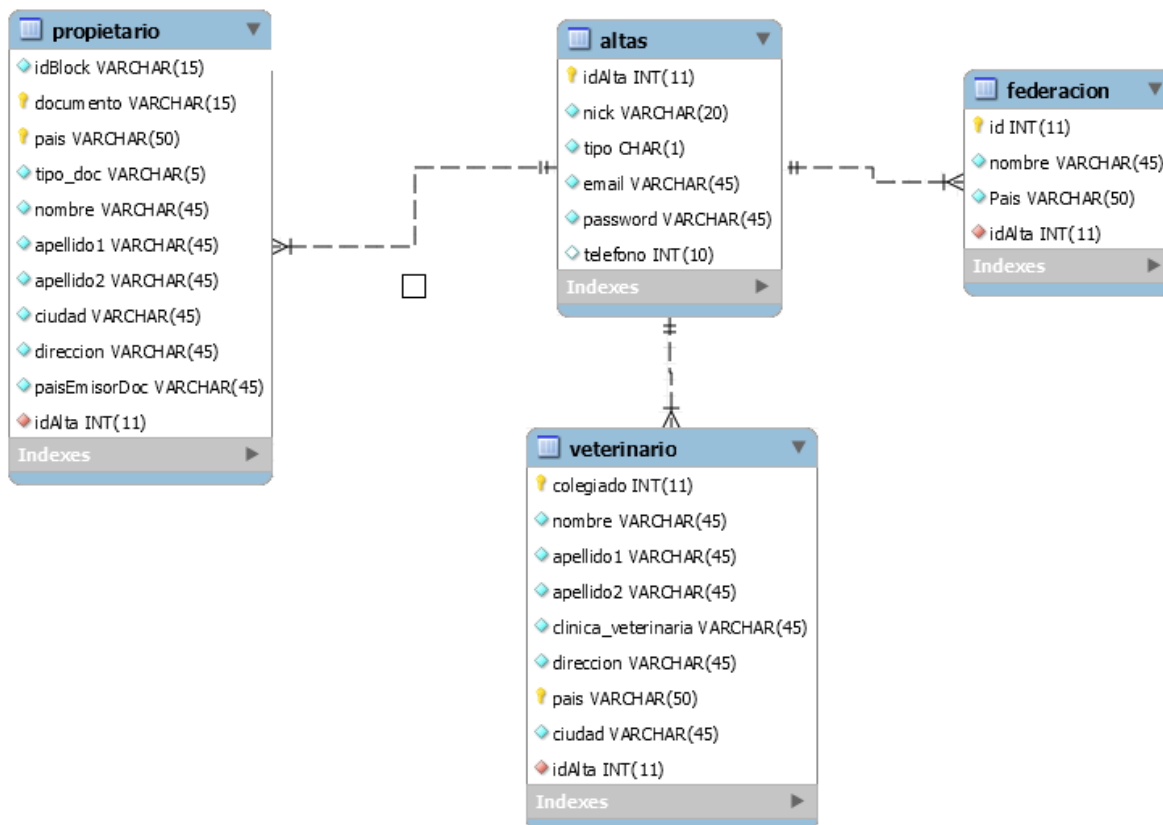
Por el otro lado, tenemos las URIs definidas para el api de Big Data:

- */bigData/api/v1/modelos/preddicion*

Diseño de la base de datos

En la versión actual del sistema, la base de datos MariaDB es utilizada para la gestión del acceso al sistema, es decir, su único objetivo es el manejo del registro y el inicio de sesión en la aplicación.

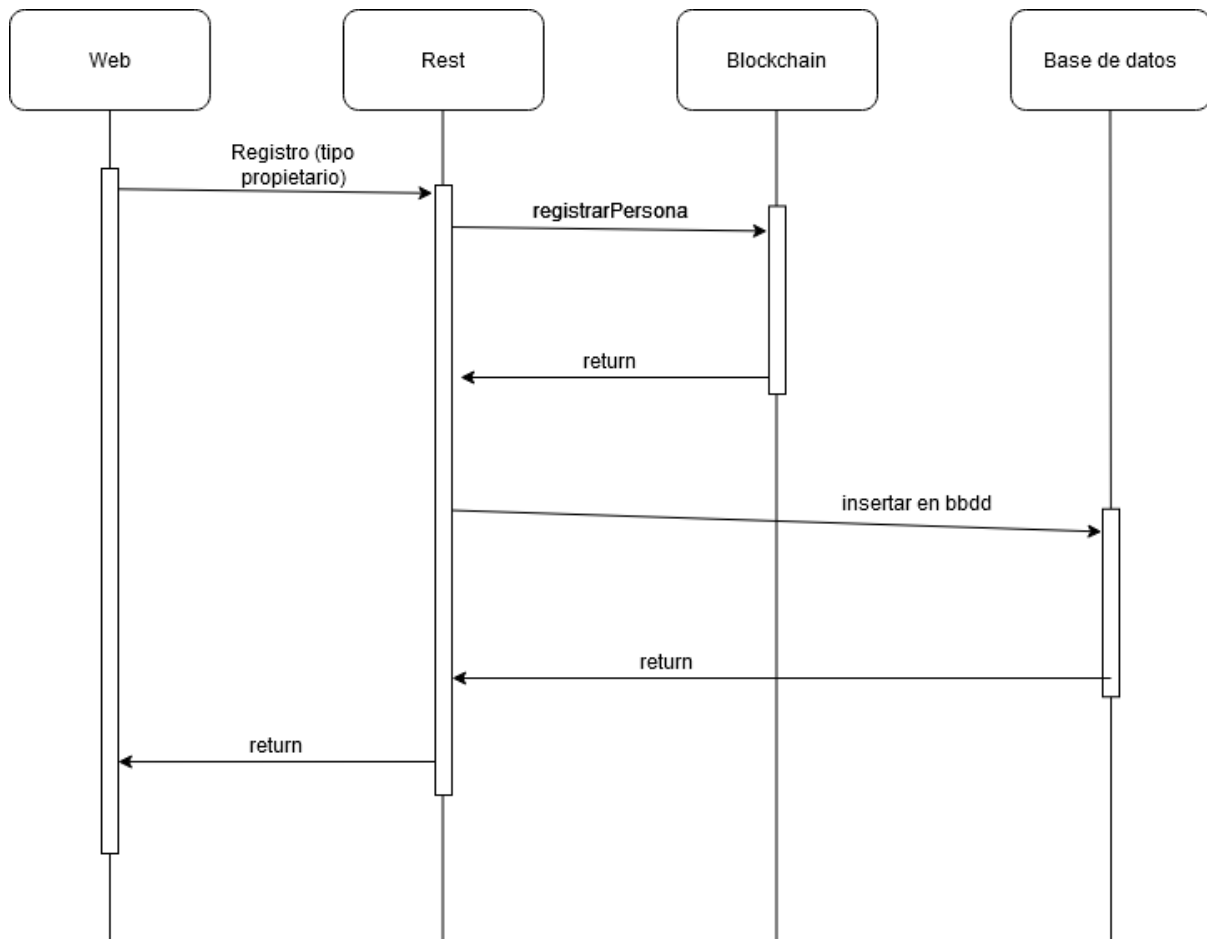
Para ello, se define una tabla “altas” que contiene varios campos obligatorios para el registro de los diferentes tipos de usuarios al sistema. Una vez registrado en la tabla “altas”, según el tipo de usuarios guardamos los datos correspondientes:



Ejemplo de llamada

En este apartado se mostrará dos ejemplos que muestran como interactúa los diferentes servicios con el servidor.

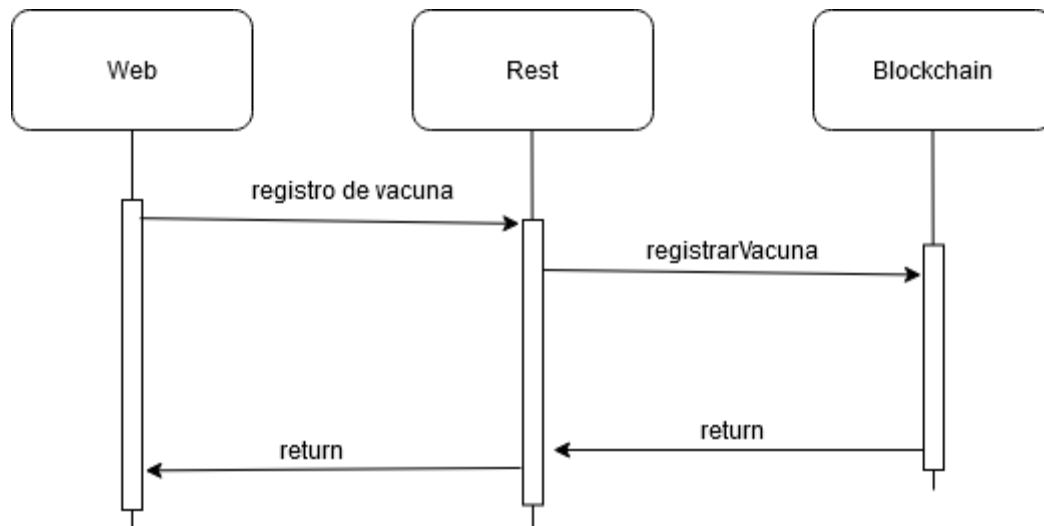
El primer caso se mostrará el sistema de alta en el sistema:



Observamos como el usuario hace el registro desde la página web contactando con el servicio rest y este es el encargado de insertarlo en la Blockchain y seguidamente, si todo ha ido correctamente en la Blockchain, se inserta en la base de datos.

Una vez insertado en los dos sistemas, el registro del usuario ha terminado exitosamente.

Por otro lado, tenemos las llamadas que solo atacan a la red Blockchain:

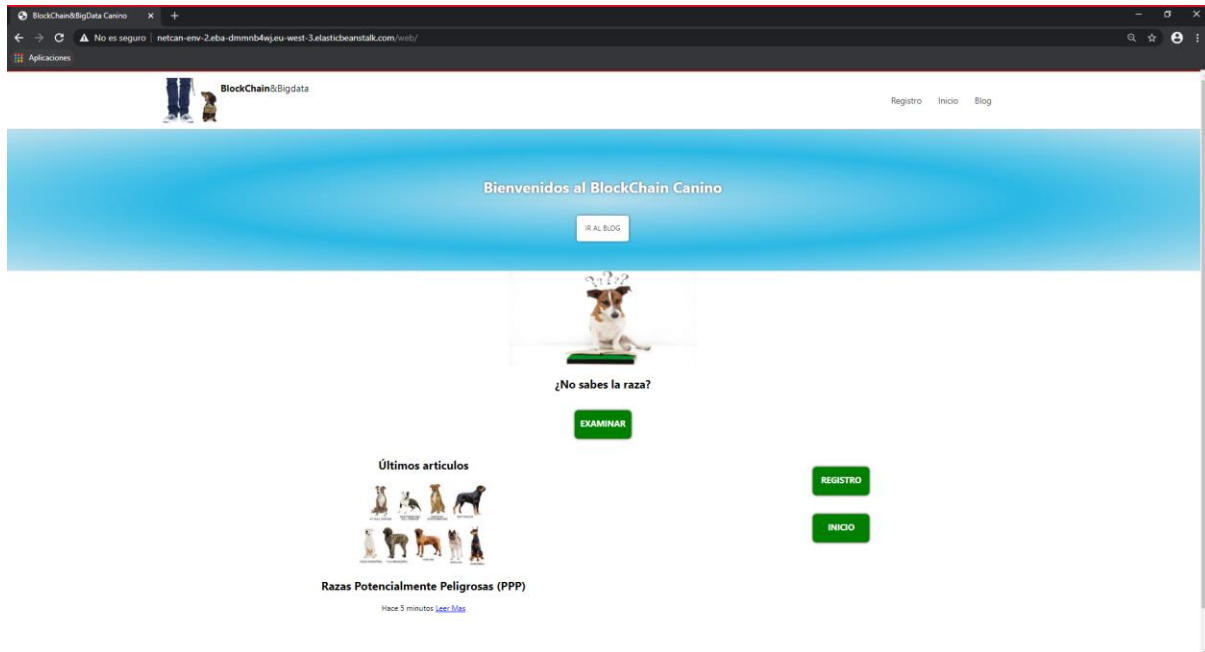


Página web

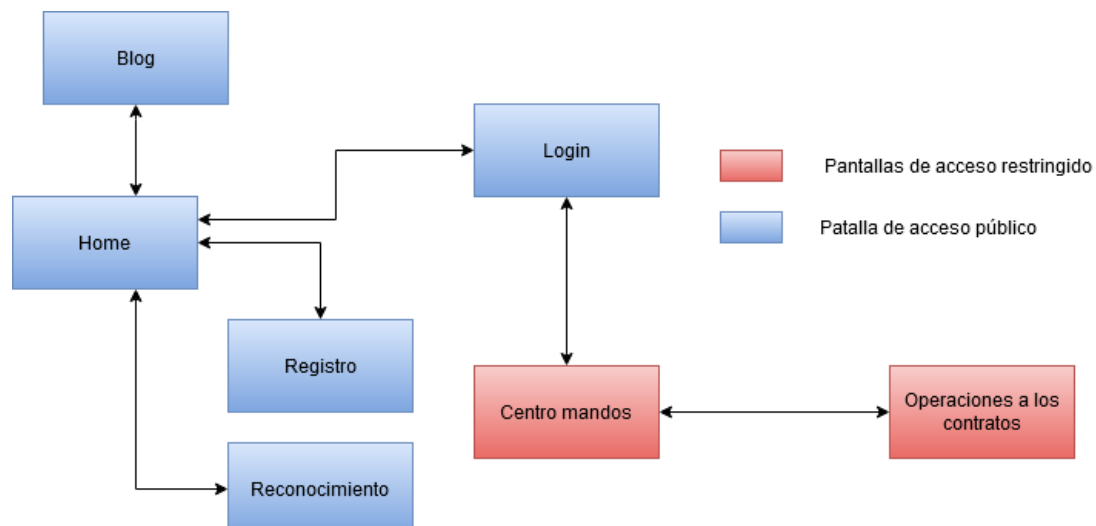
El portal web del aplicativo está alojado en el servidor de rest de entrada al sistema. Para acceder a dicho portal, utilizaremos la siguiente URL:

<http://netcan-env-2.eba-dmmnb4wj.eu-west-3.elasticbeanstalk.com/web/>

Una vez dentro encontraremos la página Home del portal web que tendrá la siguiente apariencia:

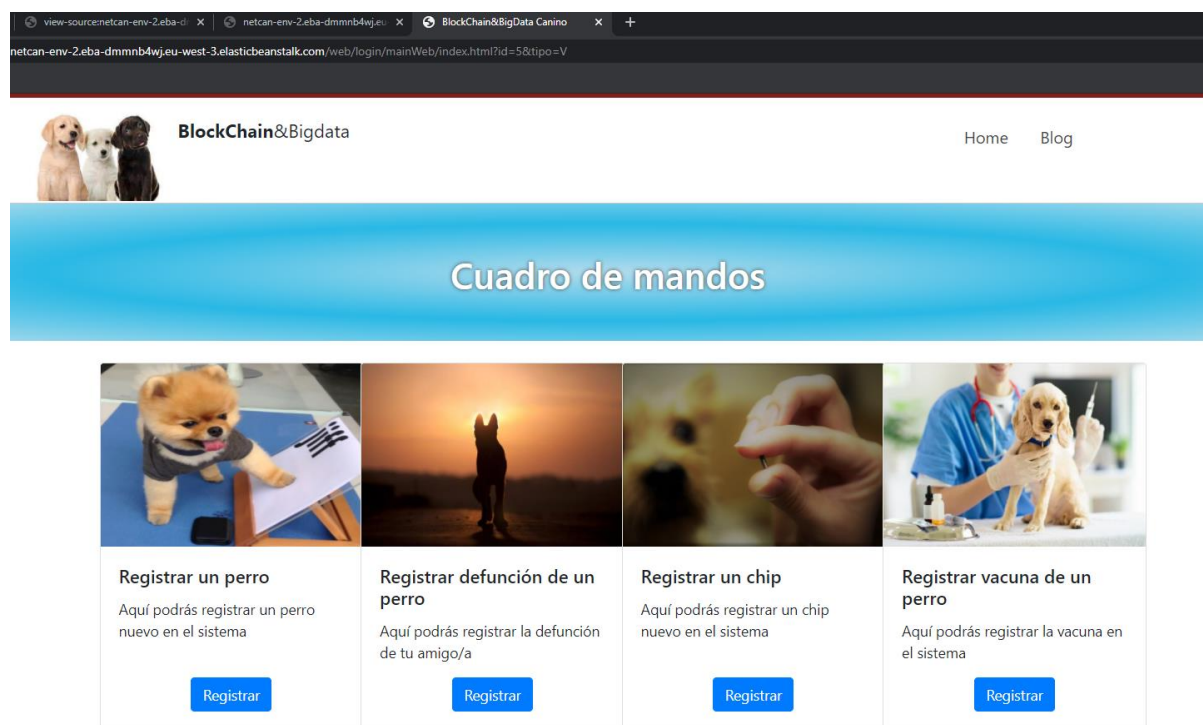


Desde la página Home, tendremos acceso a diferentes servicios dependiendo de si contamos con una cuenta o no en el aplicativo. Como se muestra en siguiente estructura, todo lo relativo a operaciones sobre Blockchain quedará restringido a navegaciones con usuario en el aplicativo. Por el contrario, acciones como el Blog o el Reconocimiento (reconocimiento de raza según una imagen) quedarán abiertos al público general:

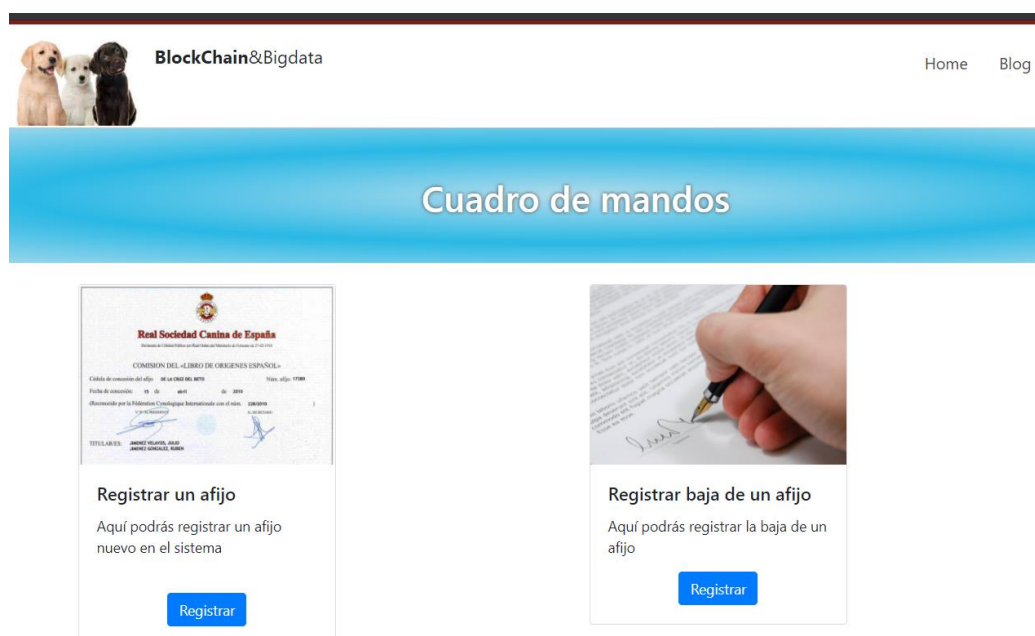


Como se indicó anteriormente en el presente documento, contamos con 3 tipos de usuarios. Estos tres tipos contarán con diferentes opciones de operaciones sobre la BlockChain en la pantalla Centro de mandos.

En el caso de ser un usuario de tipo veterinario, encontraremos la siguiente apariencia en nuestro Centro de mandos:



Se observa como este tipo de usuarios cuentan con 4 operaciones, registrar un perro, registrar la defunción de un perro, registrar el chip y registrar una vacuna de un perro. Sin embargo, para un perfil de tipo federación, el Cuadro de mandos tiene la siguiente apariencia:



Las federaciones cuentan con dos operaciones relacionadas con los afijos, registrar afijo y registrar baja de un afijo.

Para el caso de un usuario de tipo propietario, tendremos las operaciones registrar perro y registrar defunción de un perro.

Al seleccionar cualquiera de las operaciones que se han indicado anteriormente, accederos a diferentes formularios que hablarán con el Blockchain. Por ejemplo, se muestra el registro de una vacuna:



view-source:netcan-env-2.eba-d... X netcan-env-2.eba-dmmb4wj.eu... X BlockChain&BigData Canino X +

netcan-env-2.eba-dmmb4wj.eu-west-3.elasticbeanstalk.com/web/login/mainWeb/RegistroVacuna/registroVacuna.html?id=5&tipo=V

BlockChain&Bigdata Cuadro de mandos

Registro Vacunas

Formulario Alta

Identificador Perro

Identificador Veterinario

Codigo Vacuna

Fecha de Administracion

Duracion Dosis

Descripcion Vacuna

Enviar

Próximas mejoras

Actualización de JSON

Se añadirán nuevos JSON para una mejor experiencia de usuario de la página web para todos los tipos de usuarios.

En el caso de un usuario de tipo veterinario, la página web tendrá la información del veterinario, los chips que ha insertado y las vacunaciones realizadas por dicho veterinario. Para ello, se define el siguiente JSON:

Para el tipo de usuarios de federación, se define un JSON con la información de dicha federación y los concursos que organizan:

```
{
  "federacion": {
    "id": ,
    "nombre": "",
    "pais": "",
    "idAlta":
  },
  "concursos": {
    "concurso": [
      {
        "idConcurso":,
        "nombre": "",
        "patrocinadores": "",
        "pais": ""
      },
      ...
      {
        "idConcurso":,
        "nombre": "",
        "patrocinadores": "",
        "pais": ""
      }
    ]
  }
}
```

```
{
  "veterinario": {
    "colegiado": ,
    "nombre": "",
    "apellido1": "",
    "apellido2": "",
    "clinicaVeterinaria": "",
    "direccion": "",
    "pais": "",
    "ciudad": "",
    "idAlta":
  },
  "listaChips": {
    "chip": [
      {
        "codigo": ,
        "marca": "",
        "colegiado": ,
        "pais":
      },
      ...
      {
        "codigo": ,
        "marca": "",
        "colegiado": ,
        "pais":
      }
    ]
  },
  "vacunaciones": {
    "vacunacion": [
      {
        "perroNombre": "",
        "afijo": ,
        "vacuna": ,
        "fecha": ""
      },
      ...
      {
        "perroNombre": "",
        "afijo": ,
        "vacuna": ,
        "fecha": ""
      }
    ]
  }
}
```

Finalmente, para el tipo de usuario propietario, se define un nuevo JSON con la información del propietario y toda la información de los perros de dicho propietario:

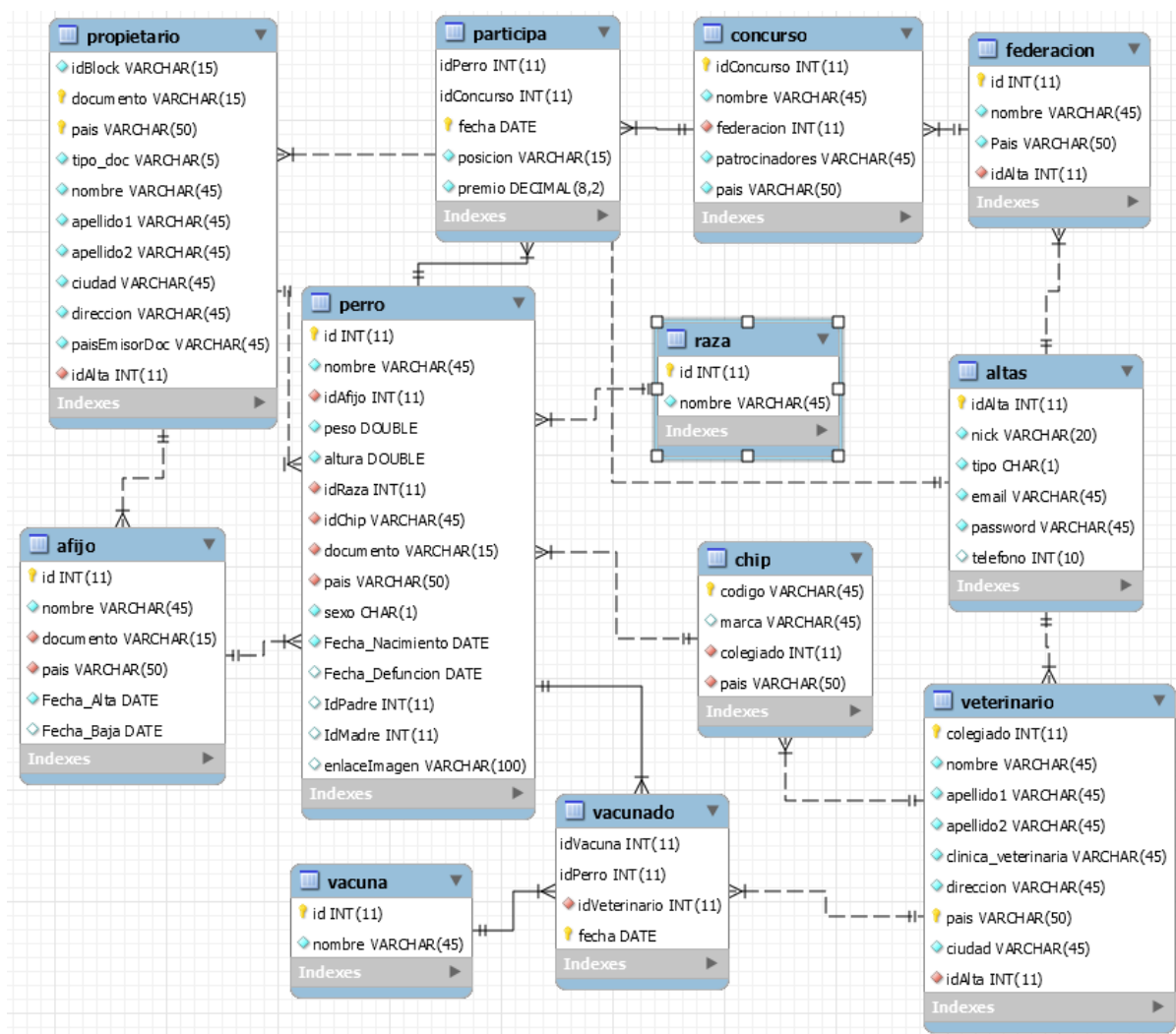
```
{
  "propietario": {
    "documento": "",
    "pais": "",
    "tipoDoc": "",
    "paisEmisorDoc": "",
    "nombre": "",
    "apellido1": "",
    "apellido2": "",
    "direccion": "",
    "ciudad": ""
  },
  "perros": {
    "perro": [
      {
        "nombre": "",
        "afijo": "",
        "peso": "",
        "altura": "",
        "raza": "",
        "sexo": "",
        "Fecha nacimiento": "",
        "Fecha defuncion": "",
        "nombre padre": "",
        "nombre madre": "",
        "enlace imagen": ""
      },
      ...
      {
        "nombre": "",
        "afijo": "",
        "peso": "",
        "altura": "",
        "raza": "",
        "sexo": "",
        "Fecha nacimiento": "",
        "Fecha defuncion": "",
        "nombre padre": "",
        "nombre madre": "",
        "enlace imagen": ""
      }
    ]
  }
}
```

Actualización de las URIs

Al añadir diferentes recursos para la actualización del servicio a la versión 2.0, se define la siguiente URI para mejorar la experiencia de usuario en el front con los nuevos JSON definidos:

- /netcan/api/v1/web

Actualización de la base de datos



Bibliografía

- [1] Wikipedia; MariaDB: <https://es.wikipedia.org/wiki/MariaDB>
- [2] Apache; Apache Tomcat: <http://tomcat.apache.org/>
- [3] Wikipedia; Java: [https://es.wikipedia.org/wiki/Java_\(lenguaje_de_programaci%C3%B3n\)](https://es.wikipedia.org/wiki/Java_(lenguaje_de_programaci%C3%B3n))
- [4] Wikipedia; JAX-RX: <https://es.wikipedia.org/wiki/JAX-RS>
- [5] Amazon; ¿Qué es AWS?: <https://aws.amazon.com/es/what-is-aws/>
- [6] Wikipedia; Maven: <https://es.wikipedia.org/wiki/Maven>
- [7] Wikipedia; HTML: <https://es.wikipedia.org/wiki/HTML>
- [8] Wikipedia; Bootstrap: [https://es.wikipedia.org/wiki/Bootstrap_\(framework\)#cite_note-3](https://es.wikipedia.org/wiki/Bootstrap_(framework)#cite_note-3)
- [9] Bootstrap; Introduction: <https://getbootstrap.com/docs/4.3/getting-started/introduction/>
- [10] Wikipedia; Hoja de estilos en cascada: https://es.wikipedia.org/wiki/Hoja_de_estilos_en_cascada
- [11] Wikipedia; JavaScript: <https://es.wikipedia.org/wiki/JavaScript>
- [12] Wikipedia; JQuery: <https://es.wikipedia.org/wiki/JQuery>