

# Hubble

*Módulo - Administración*

**Documento de arquitectura**

# Índice

Índice.....	2
1. Objetivos del documento.....	3
2. Introducción.....	3
3. Flujo básico del módulo de administración.....	3
4. Estructura alto nivel.....	4
5. Seguridad .....	4
Autenticación / Autorización .....	4
6. Backend .....	6
Conexión con la base de datos. ....	8
7. FrontEnd .....	8
Componentes relevantes .....	8
Ubicación de los componentes .....	9
Interceptor .....	9
Estilos .....	9
8. Prototipos de pantallas.....	10
9. Colecciones MongoDB.....	14
ApplicationStorage .....	14
ProviderStorage.....	17
UserStorage.....	18



# 1. Objetivos del documento

Documentar el diseño del módulo de administración desde el punto de vista arquitectónico, abarcando tanto el backend como el frontend y las colecciones MongoDB resultantes de la persistencia del modelo.

## 2. Introducción

El módulo de administración forma parte del propio Hubble, compartiendo la base de datos y la estructura de paquetes que se venía utilizando para el resto de las entidades. Los fuentes serán incorporados a los repositorios existentes de frontend y backend, según corresponda.

Desde este módulo se podrán crear y configurar aplicaciones, configurar los proveedores, además de administrar los usuarios y los umbrales que serán utilizados en los tableros.

La comunicación entre el backend y el frontend será a través de una API REST que expondrá todas las operaciones necesarias y se encargará de interceptar todas las invocaciones para garantizar la autenticación/autorización en el acceso a la información que se maneja en dicho módulo.

## 3. Flujo básico del módulo de administración

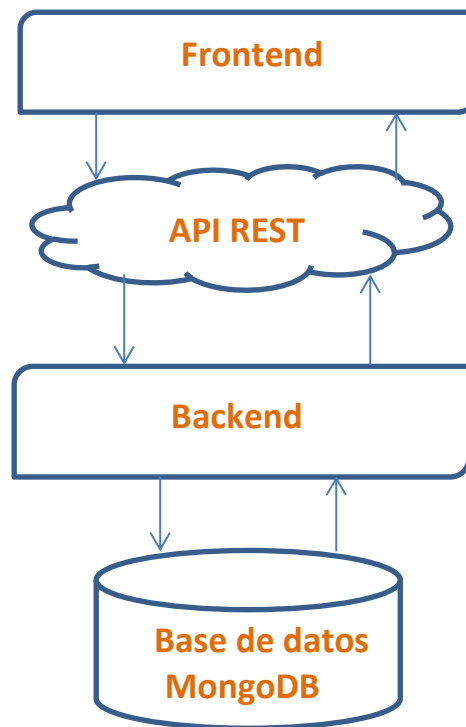


Una vez realizada todas las configuraciones los usuarios creados podrán autenticarse en el sistema, haciendo uso de sus credenciales y comenzar con la explotación del sistema.



## 4. Estructura alto nivel

Hubble está separada en dos grandes componentes **Backend** y **Frontend**, ambos comunicados a través de una capa API Rest. Para el resguardo de los datos se utiliza Mongo DB.



## 5. Seguridad

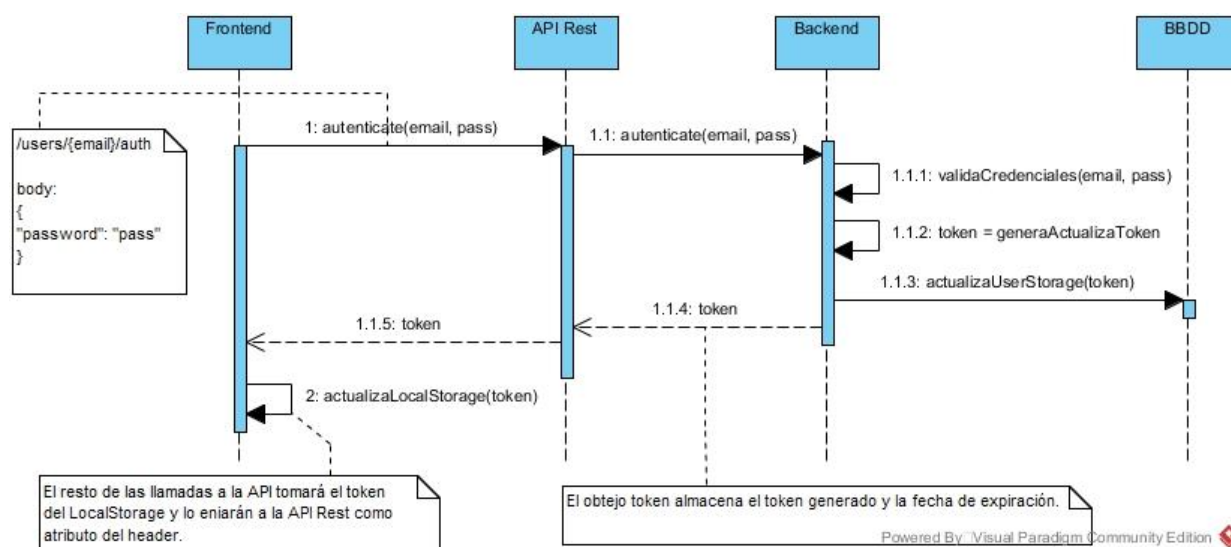
### Autenticación / Autorización

Los usuarios deberán autenticarse en el sistema utilizando su dirección de correo electrónico y su contraseña. En caso de contar con credenciales válidas el backend responderá con un token que será almacenado en el LocalStorage del navegador y acompañará a todas las peticiones en el header, de forma tal que se pueda validar la identidad de quién está pretendiendo acceder a las operaciones.

La contraseña de cada usuario será almacenada utilizando el algoritmo Argon21, recomendado por OWASP2.

- 
- 1 <https://en.wikipedia.org/wiki/Argon2> (NO SE SI PONDRIA ESTO JEJEJE DEJARÍA SOLO EL SEGUNDO Y LISTO TOTAL)
  - 2 [https://www.owasp.org/index.php/Password\\_Storage\\_Cheat\\_Sheet#Argon2\\_usage\\_proposal\\_in\\_Java](https://www.owasp.org/index.php/Password_Storage_Cheat_Sheet#Argon2_usage_proposal_in_Java)





Hubble tendrá un esquema simple de usuarios y roles, donde cada usuario podrá tener uno o más de los siguientes roles:

- ✓ **Administrador:** podrá acceder a todas las pantallas del módulo de administración de la aplicación.
- ✓ **Usuario:** podrá acceder a los dashboards del frontend.

La autenticación/autorización del usuario a las distintas funcionalidades tiene lugar tanto en el frontend como en el backend, partiendo de la premisa que no se puede acceder a ninguna funcionalidad ni operación de la API sin haberse autenticado antes y una vez autenticado, el usuario sólo debe visualizar lo que le corresponda según su rol.

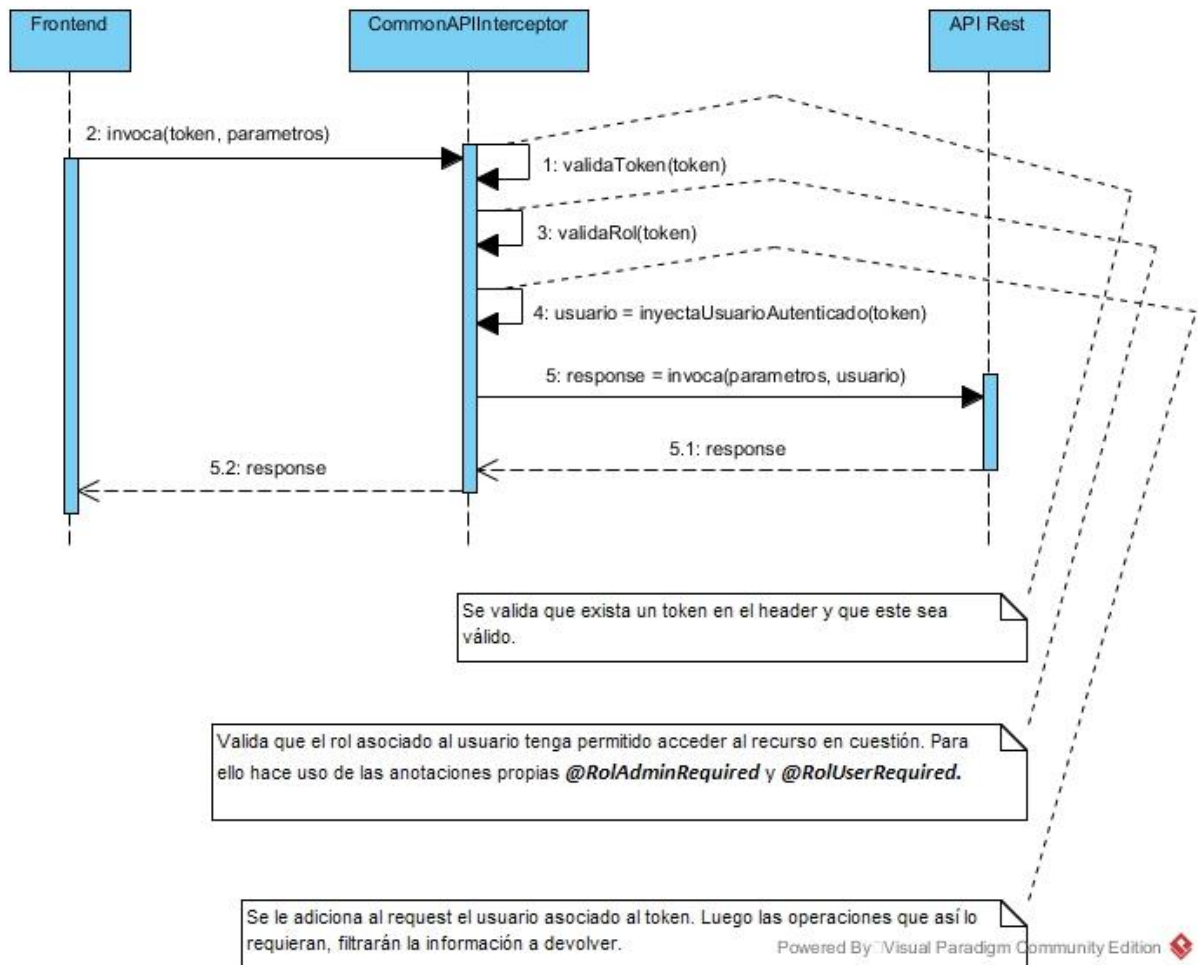
En el frontend se maneja la visualización de componentes en función de los roles del usuario autenticado y desde el backend se cuenta con un componente llamado **CommonAPIInterceptor** que extiende su comportamiento de **HandlerInterceptorAdapter** que se encarga de interceptar todas las llamadas a la API Rest y desde acá valida la veracidad del token suministrado así como el acceso al recurso solicitado dependiendo del rol que tiene asignado el usuario asociado al token.

Para la identificación de las operaciones de las APIs accesibles por uno u otro rol, se crearon anotaciones propias, **@RolAdminRequired** y **@RolUserRequired**, con las cuales es posible anotar las operaciones que así lo requieran, facilitando la identificación de estas durante la autorización.

Que una operación no tenga ninguna de las dos anotaciones, indica que no se requiere validación del rol para acceder a ella.

El componente **CommonAPIInterceptor** se encarga además de facilitar los datos del usuario autenticado a partir del token, de esta forma operaciones como las de aplicaciones restringen la información a devolver en función del usuario que está consultando el recurso.

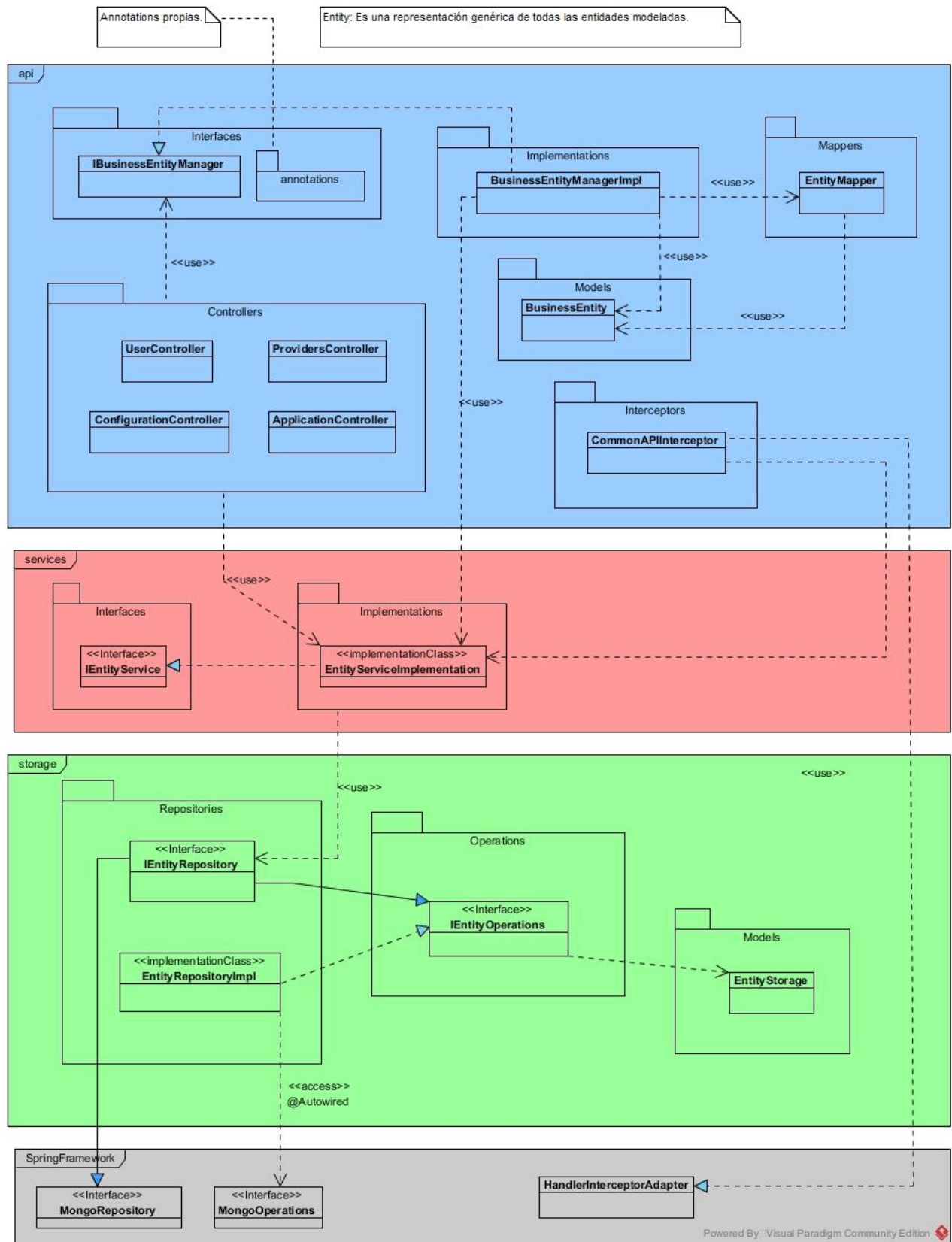




## 6. Backend

Para una mejor comprensión de la arquitectura del backend en lo que al módulo de administración respecta, se muestra una imagen dividida por artefactos y la interrelación entre ellos para procesar las peticiones que llegan a través de la API Rest, haciendo uso de componentes propios, de Java y Spring.





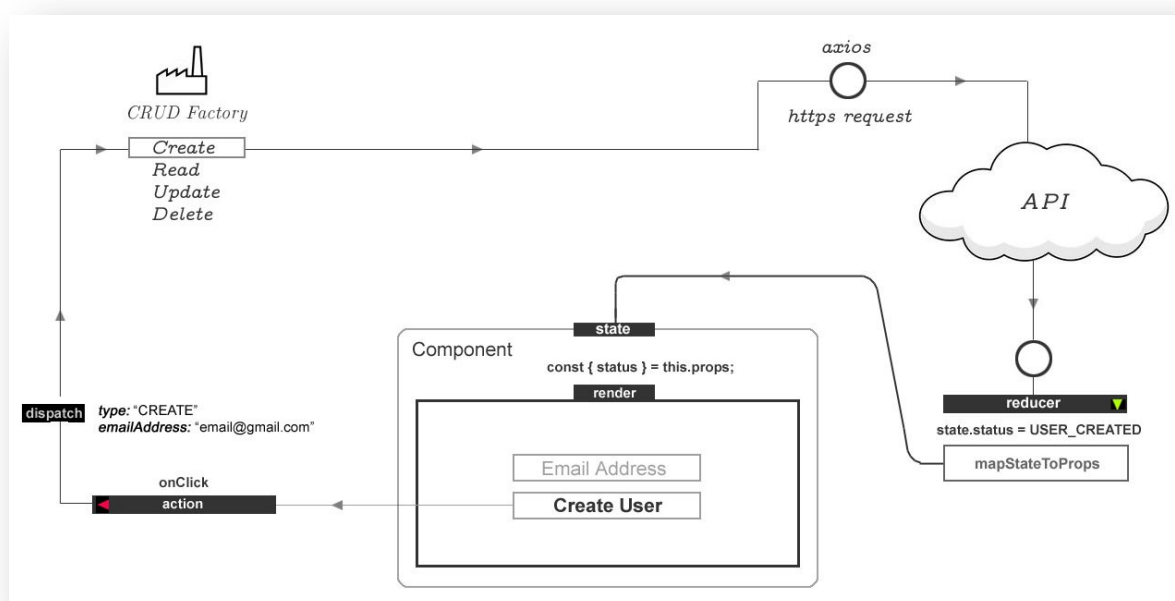
## Conexión con la base de datos.

Las propiedades utilizadas para la conexión a la base de datos se definen en el fichero **application.properties** ubicado en el artefacto Storage.

El host donde se encuentra la base de datos se define haciendo uso del nombre DNS **hubble.bd.host**, el cual debe estar configurado en fichero **hosts** en el equipo donde esté corriendo el backend apuntando al servidor que corresponda.

## 7. FrontEnd

Para una mejor comprensión del funcionamiento del frontend basado en las tecnologías utilizadas, se muestra el siguiente diagrama<sup>3</sup>:



El módulo de administración, al igual que el resto de las funcionalidades de Hubble sigue este flujo de trabajo. Para más información, consultar bibliografía especializada de las tecnologías utilizadas.

### Componentes relevantes

Vale la pena resaltar algunos componentes que juegan un papel importante en la articulación del frontend que si bien no son utilizados únicamente por el módulo de administración, juegan un papel fundamental en su funcionamiento.

**Common.js:** Es el encargado de centralizar todas aquellas acciones que por su uso dentro de la aplicación son reutilizables. Dígase:

<sup>3</sup> Tomado de [https://medium.com/@js\\_tut/the-saga-continues-magic-in-react-44da8d134285](https://medium.com/@js_tut/the-saga-continues-magic-in-react-44da8d134285)





- Setear, obtener y eliminar el token del LocalStorage.
- Conformar el header que luego será utilizado en los request.
- Verificar el estado del usuario respecto a la autenticación (autenticado o no)
- Llamada genérica a las APIs con parámetros por defecto, que son redefinidos según la necesidad de cada invocación.
- Gestionar los roles del usuario autenticado en el LocalStorage.

**.env:** Define las propiedades de conexión con la API y el TaskRunner, así como el host y puerto donde estará disponible Hubble una vez desplegado. En el repositorio de fuentes existe un fichero **.env.copy** con las propiedades que son necesarias configurar y la idea es que este sirva de guía para crear el archivo .env local que será el utilizado por la aplicación.

**webpack.config.js:** Entre otras cosas define los valores por defecto para todas las propiedades que conforman el .env en caso que este no haya sido configurado o no exista.

**app.jsx:** Siendo este el punto de entrada a la aplicación, se define toda la lógica de validación de los roles de usuario y la visualización de los componentes según corresponda. Por otro lado también maneja el componente **AppRoutes** encargado de las reglas de navegación dentro del sitio.

## Ubicación de los componentes

Los componentes que conforman las páginas del módulo de administración se encuentran todos bajo el directorio `pages (src/pages)`, con la siguiente distribución:

**Autenticación:** `/login`

**Usuarios:** `/users`

**Proveedores:** `/providers`

**Aplicaciones:** `/applications`

**Umbrales HealthIndex:** `/thresholds`

## Interceptor

Al igual que en el backend, el frontend también cuenta con un interceptor, en este caso se hace uso de los interceptors de **Axios js**. Si bien esta librería provee interceptores tanto para el request como para el response, dentro de Hubble se utiliza sólo para el response. Interceptando aquellos responses con `HttpStatus 401` o `405`.

**HTTP\_401** => Es devuelto por la API ante una autenticación fallida.

**HTTP\_405** => Es devuelto por la API ante una autorización fallida.

Ante un **HTTP\_401**, el interceptor redirecciona a la pantalla de autenticación, mientras que ante un **HTTP\_405** se redirecciona a una página que indica al usuario que está accediendo a un recurso no autorizado.

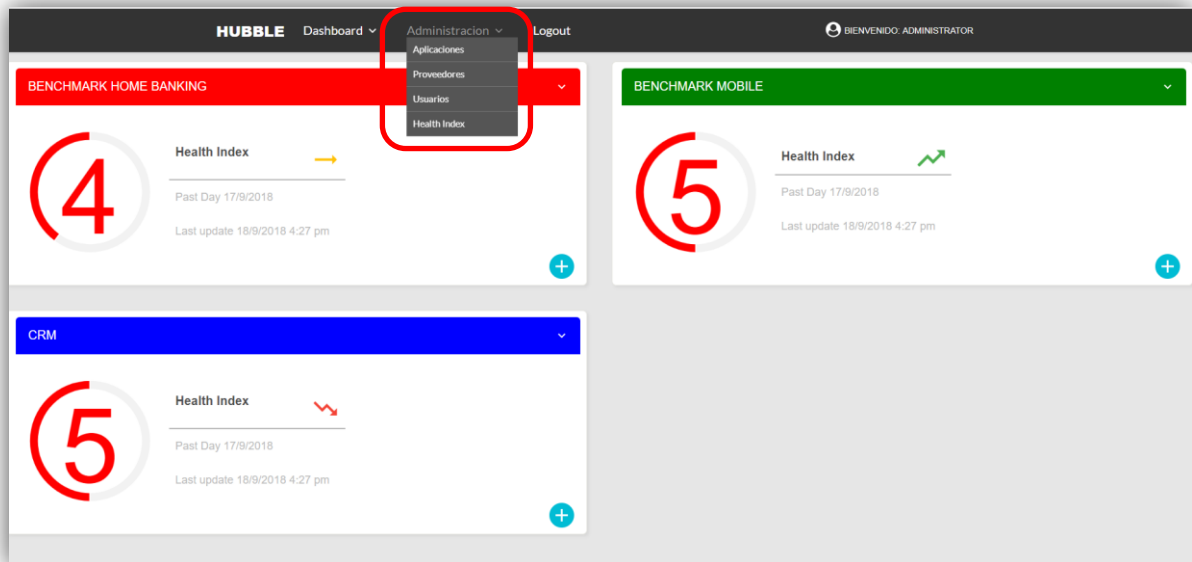
## Estilos

Los estilos utilizados para los formularios del módulo de administración se encuentran en el archivo **admin.css** ubicado en `src/styles/admin.css`.



## 8. Prototipos de pantallas

A continuación se incluyen los prototipos de las pantallas que conforman el módulo de administración. Para conocer su funcionamiento, consulte el Manual de Usuario confeccionado a tales fines.



Aplicaciones			
Aplicación	KPIs activos	Habilitado ?	Task Runner ?
<b>Home Banking</b> Descripción de Benchmark Home Banking	DEFECTS AVAILABILITY TASKS PERFORMANCE EVENTS	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
<b>Mobile Banking</b> Descripción de Benchmark Mobile	DEFECTS AVAILABILITY TASKS PERFORMANCE EVENTS	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
<b>CRM</b> Descripción de CRM	DEFECTS AVAILABILITY TASKS PERFORMANCE EVENTS	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>



Home Banking

Nombre \*

Home Banking

Descripción

Descripción

Guardar

✕ Cerrar

Home Banking

Nombre \*

Home Banking

Habilitado ?

☒

Task Runner ?

☒

Descripción

Descripción de Benchmark Home Banking

Tareas

Defectos

Performance y Disponibilidad

Eventos

Habilitado ?

☒

Configuración del KPI

Este KPI se mide en cantidad de días de desvío en la ejecución de la tarea.

Día

0 < OK ≤ 5 < Warning ≤ 100 < Critical ≤ Infinity

Semana

0 < OK ≤ 5 < Warning ≤ 100 < Critical ≤ Infinity

Mes

0 < OK ≤ 5 < Warning ≤ 100 < Critical ≤ Infinity

Integración PPM

Activo ?

☒

Nombre de la Aplicación \*

Banca por Internet

Este es el nombre del campo REQKNTA, PROGRAM\_REFERENCE

Guardar

✕ Cerrar

Proveedores		
Proveedor	Habilitado ?	Task runner ?
ALM	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
AppPulse	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
BSM	<input type="checkbox"/>	<input type="checkbox"/>
Jira	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
PPM	<input checked="" type="checkbox"/>	<input type="checkbox"/>
SiteScope	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>



### Editar Proveedor ALM

**Entorno**

Host  
10.10.20.170

Puerto  
8080

Usuario  
matias.lapalma

Constaseña  
Constaseña

Dominio  
TSAR\_SOFTWAREFACTORY

Proyecto  
HUBBLE

**Configuración**

Campo con el nombre de la aplicación  
project

Campo con el estado  
status

Valores de estados que significan "Abierto"  
Nuevo.Abierto.Reabierto

Campo con la transacción  
project

Origen del proveedor  
Alm

Nombre del proveedor  
Alm Tsoft

**Task Runner**

Habilitado ?  
☒

Días de la semana  
☐ Lunes a viernes  
☒ Todos los días

Rango horario  
☒ 9:00 a 18:00  
☐ Las 24 horas

Frecuencia de ejecución  
☒ Una vez por día  
☐ Cada 1 hora  
☐ Cada 30 min  
☐ Cada 15 min  
☐ Cada 5 min

Guardar

Cerrar

Usuarios				
Correo electrónico	Nombre	Roles	Aplicaciones	Habilitado <span>?</span>
admin@tsoftlatam.com	Administrator	ADMINISTRATOR USER	Benchmark Home Banking Benchmark Mobile CRM	
juan.alfonso@fit.com.ar	Juan	ADMINISTRATOR	CRM Benchmark Home Banking	<input checked="" type="checkbox"/>

### Rodrigo Patricio

Nombre <sup>\*</sup>  
Rodrigo Patricio

Correo Electrónico <sup>\*</sup>  
rodrigo@tsoftlatam.com  
El correo electrónico será el usuario de ingreso al sistema

Contraseña <sup>\*</sup>  
.....  
Al menos 10 caracteres: letras mayúsculas, minúsculas y números


Roles  
☒ Administrador  
☒ Usuario

Home Banking ☐ Mobile Banking ☐ CRM ☐

Guardar

Cerrar



 Juan

Nombre \*

Juan

Correo Electrónico \*

juan.alfonso@fit.com.ar

El correo electrónico será el usuario de ingreso al sistema

☒ Modificar Contraseña

Contraseña \*

.....

Al menos 10 caracteres: letras mayúsculas, minúsculas y números

Roles


☒ Administrador

☐ Usuario


Guardar

Cerrar


Configuración umbrales health index

10 <  OK ≤

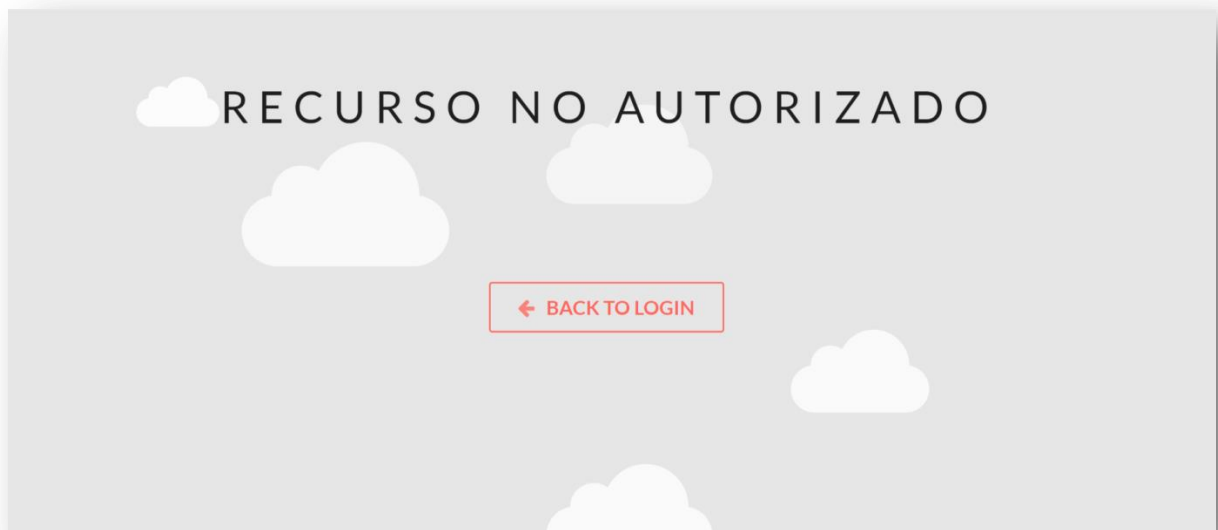
8

<  Warning ≤

6

<  Critical ≤ 1

Guardar



## 9. Colecciones MongoDB

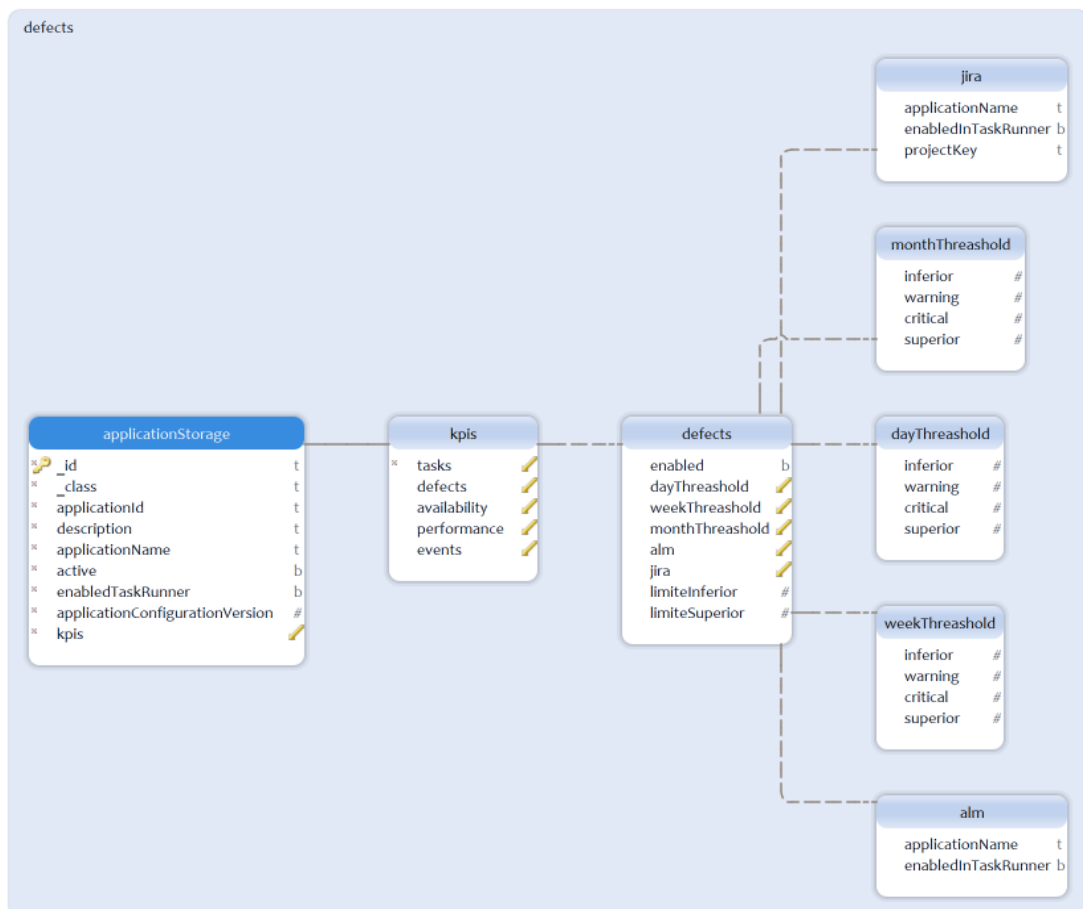
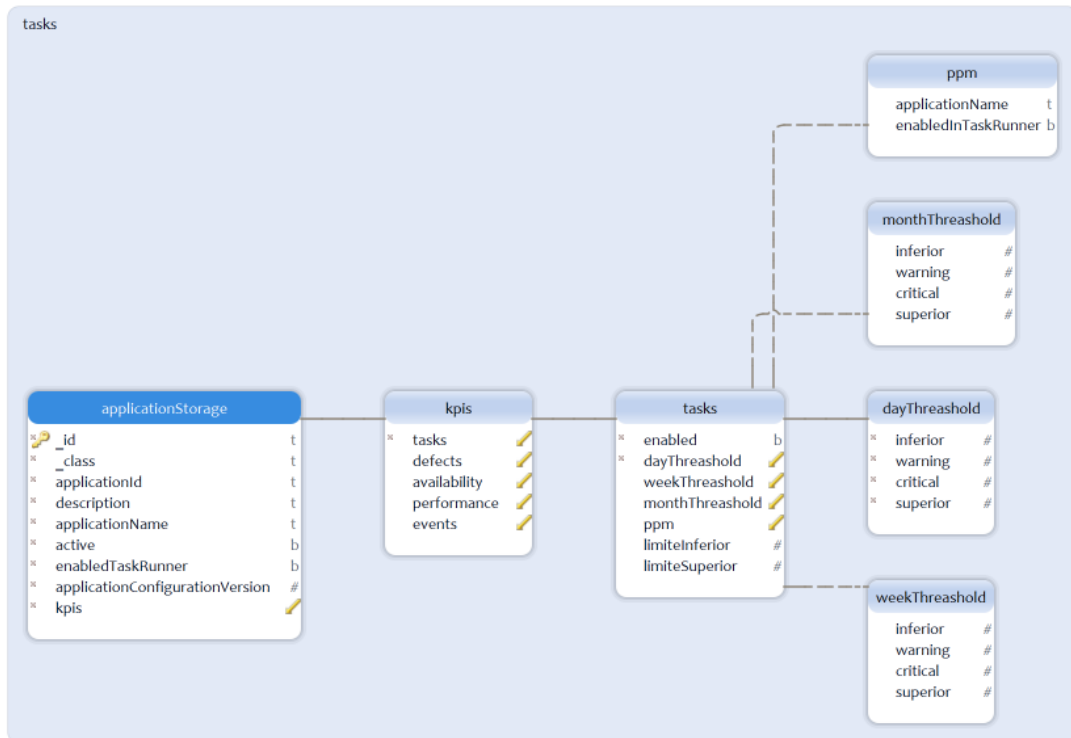
Las entidades serán almacenadas en MongoDB siguiendo el esquema:

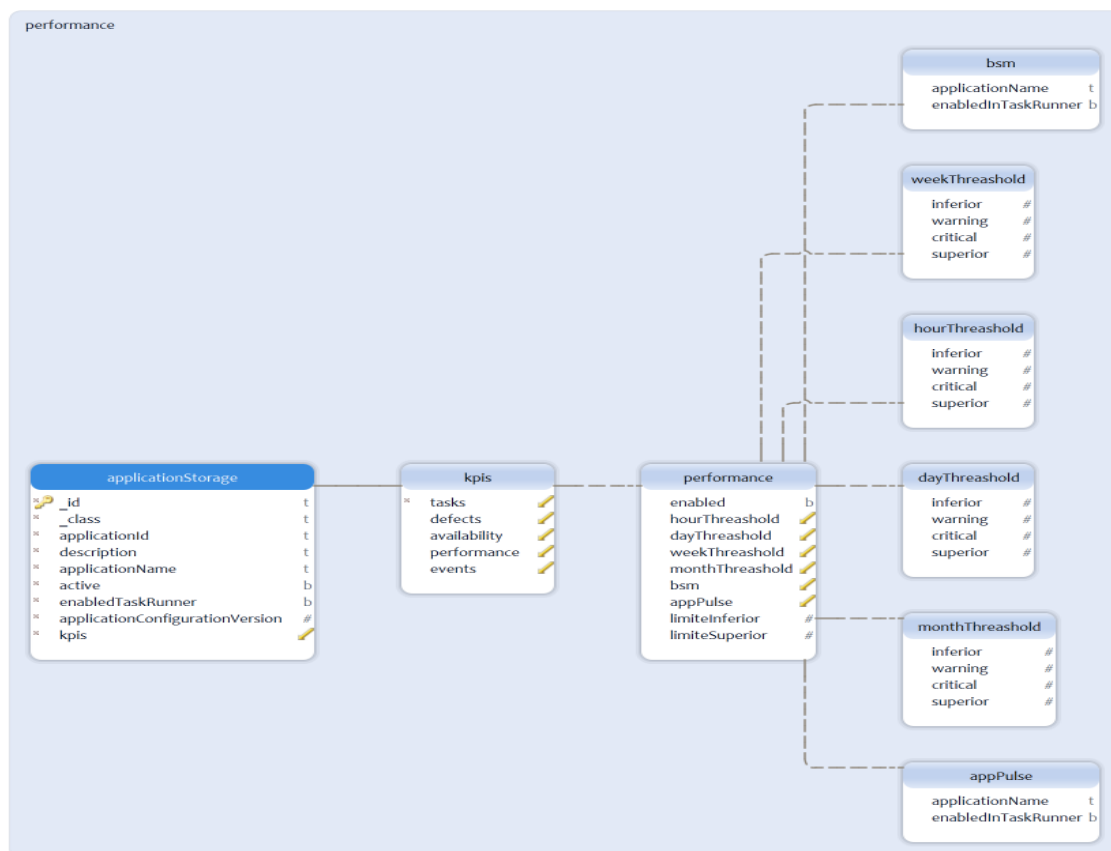
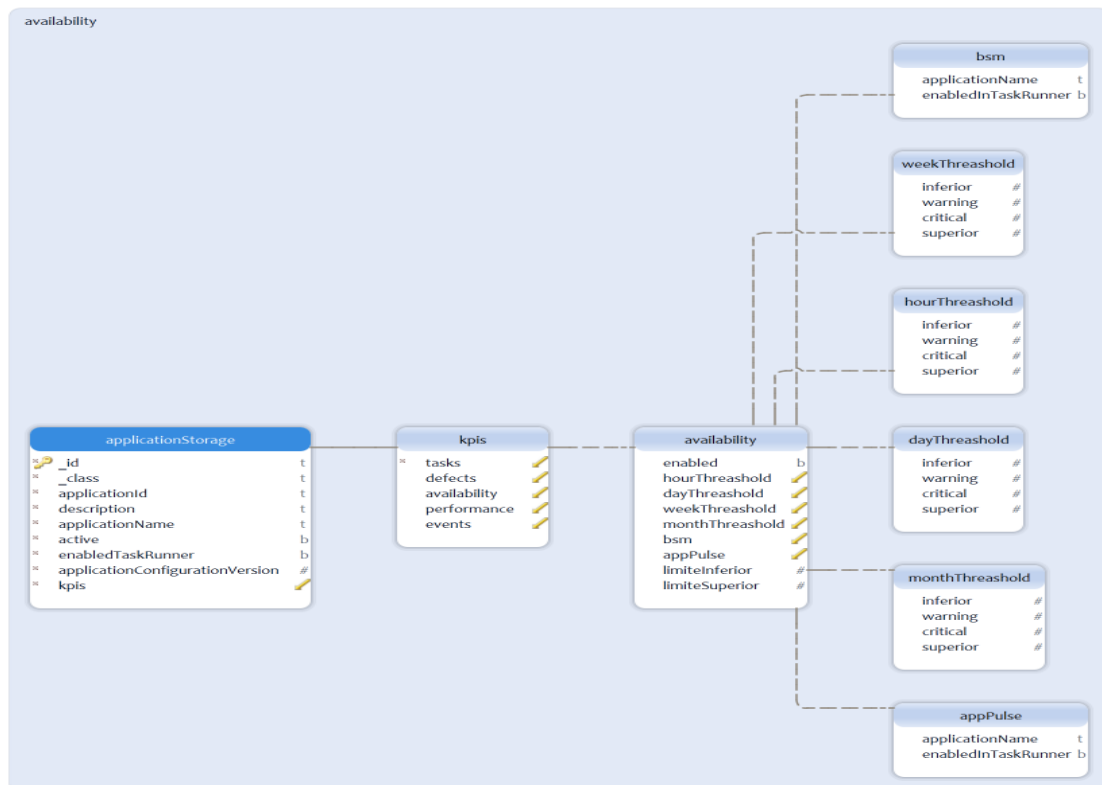
### ApplicationStorage



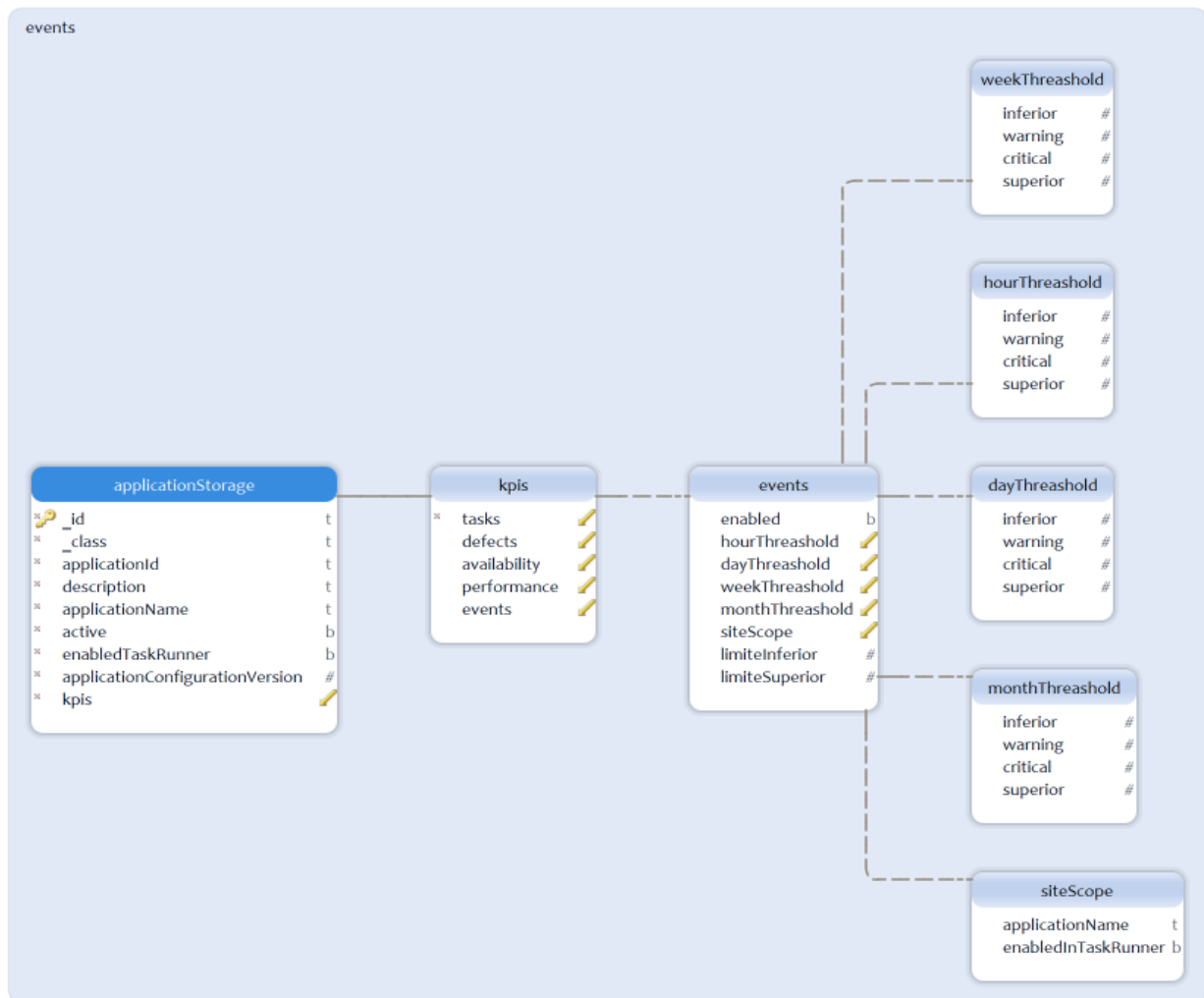
Para un mejor entendimiento del diagrama cada KPI se representa en diagramas independientes.



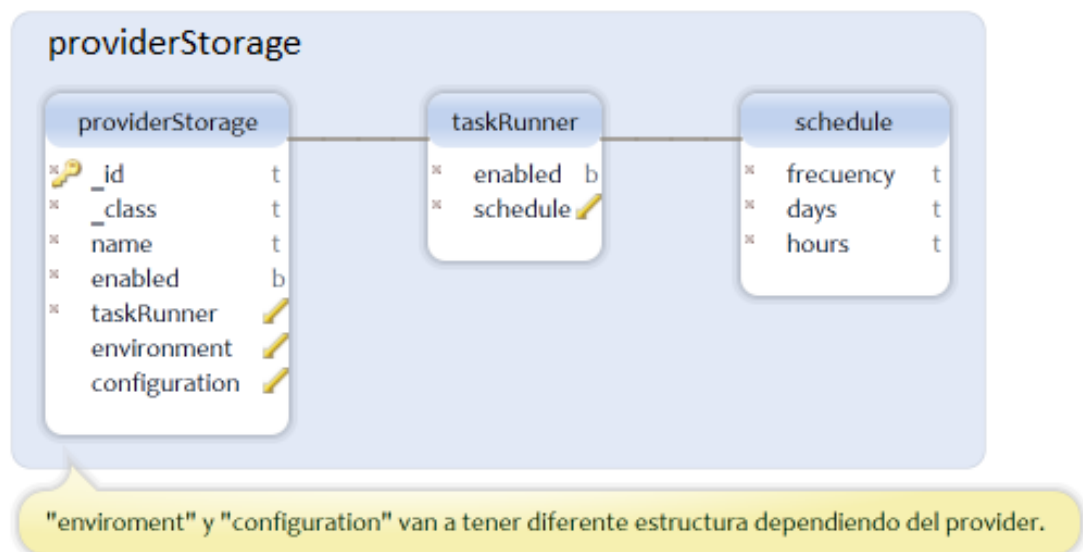




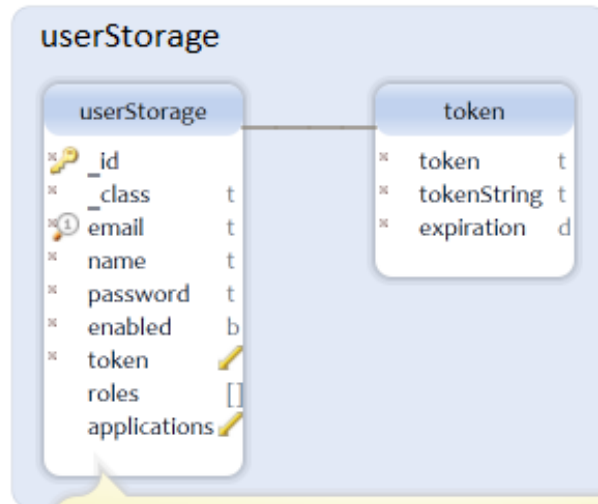




## ProviderStorage



## UserStorage



Tiene una referencia a las aplicaciones asociadas en caso que tenga el rol de "Usuario".

