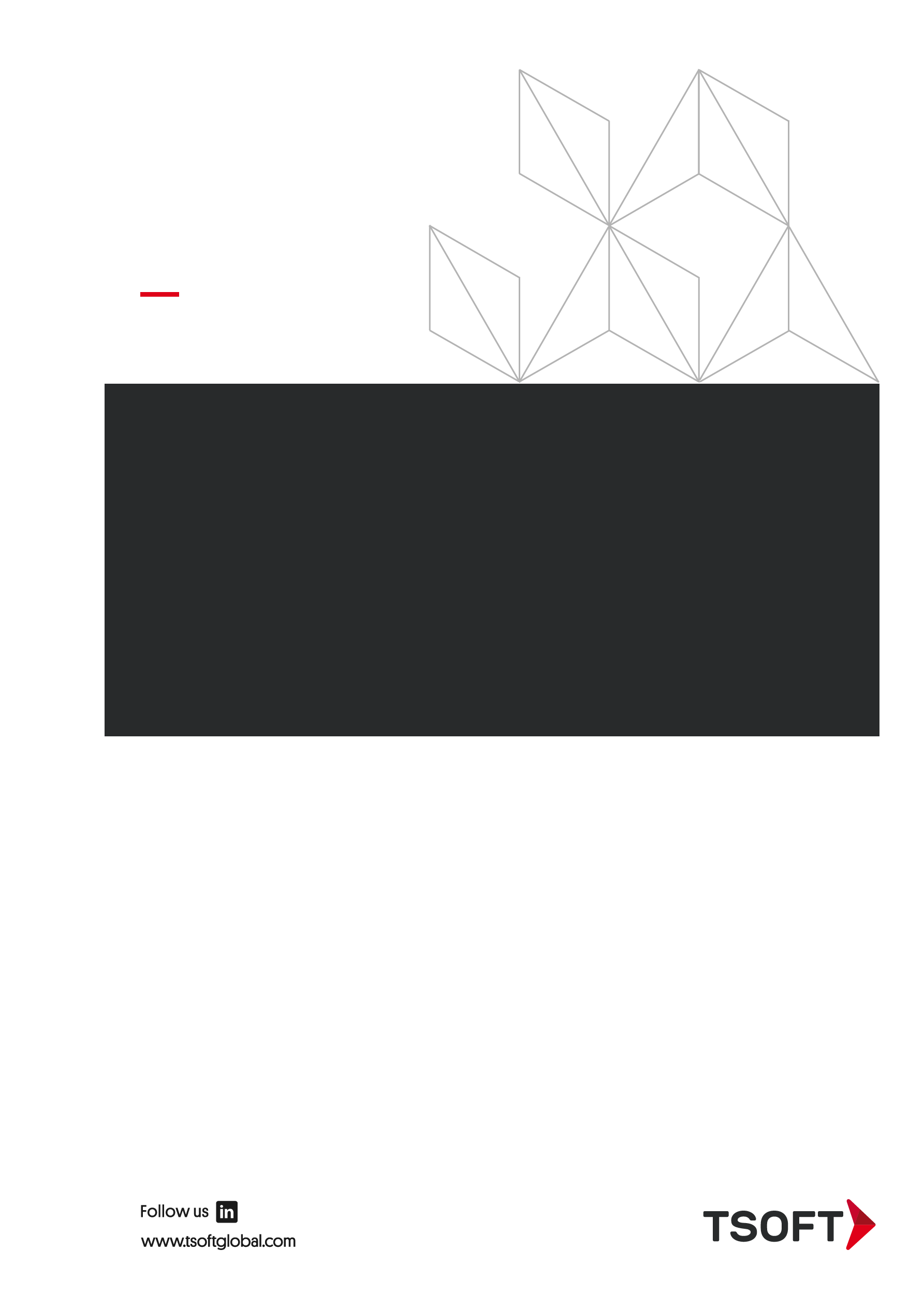
****

**Hubble**

***Administración***

**Diseño**

**Indice**

[1. Introducción 2](#__RefHeading___Toc801_3717703543)

[2. Seguridad 2](#__RefHeading___Toc840_3395260487)

[Autenticación 3](#__RefHeading___Toc180_1185719925)

[Autorización 3](#__RefHeading___Toc182_1185719925)

[3. Prototipos 3](#__RefHeading___Toc803_3717703543)

[4. Colecciones MongoDB 11](#__RefHeading___Toc805_3717703543)

[Aplicaciones 12](#__RefHeading___Toc207_2372516870)

[Proveedores 15](#__RefHeading___Toc209_2372516870)

[ALM 15](#__RefHeading___Toc211_2372516870)

[AppPulse 16](#__RefHeading___Toc213_2372516870)

[BSM 16](#__RefHeading___Toc215_2372516870)

[Jira 16](#__RefHeading___Toc217_2372516870)

[PPM 16](#__RefHeading___Toc219_2372516870)

[SiteScope 17](#__RefHeading___Toc221_2372516870)

[Usuarios 17](#__RefHeading___Toc223_2372516870)

[APIs 17](#__RefHeading___Toc890_2169729150)

# Introducción

Las pantallas de administración se utilizarán para crear y configurar aplicaciones, configurar los *providers*, y administrar los usuarios de Hubble.

Se agregarán pantallas al *front-end*, a las que tendrán acceso los usuarios que tengan el rol Administrador.

Los fuentes de esta aplicación serán incorporados a los repositorios existentes de *front-end* y *back-end*, según corresponda.

El prototipo de las pantallas estará versionado en el repositorio de *back-end*, en el directorio “admin”.

# Seguridad

La seguridad tendrá un esquema simple de usuarios y roles.

Los usuarios deberán autenticarse en el sistema utilizando su dirección de correo electrónico, y su contraseña.

Los usuarios serán almacenados en la colección “userStorage” en MongoDB. La contraseña de cada usuario será almacenada en esa misma colección utilizando el algoritmo Argon2[[1]](#footnote-1). Este elige este algoritmo dado que está recomandado por OWASP[[2]](#footnote-2), existe una librería Java para utilizarlo.

Cada usuario podrá tener uno o más de los siguientes roles:

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

El administrador podrá crear, modificar, y deshabilitar usuarios, así como blanquear las contraseñas.

## Autenticación

La autenticación se hará mediante un servicio REST, realizando un POST al recurso URL http://host:port/users/{email}/auth. El cuerpo del post deberá tener la contraseña.

La respuesta es de tipo JSON, con la siguiente estructura:

{

“token”: “”

“expirationDate”: “2018-06-30T00:00:00.00-3:00”

}

El token se almacena en el documento del usuario, dentro de la colección userStorage.

El token deberá enviarse en todas las invocaciones a servicios.

## Autorización

Para consultar la información del usuario, incluidos sus roles, realizar un GET a la URL http://host:port/users/{email}. La respuesta será de tipo JSON, con la siguiente estructura:

{

"id": "5b1abbb04f6e28305f4eaa7d",

"email": "admin@tsoftlatam.com",

"name": "Administrator",

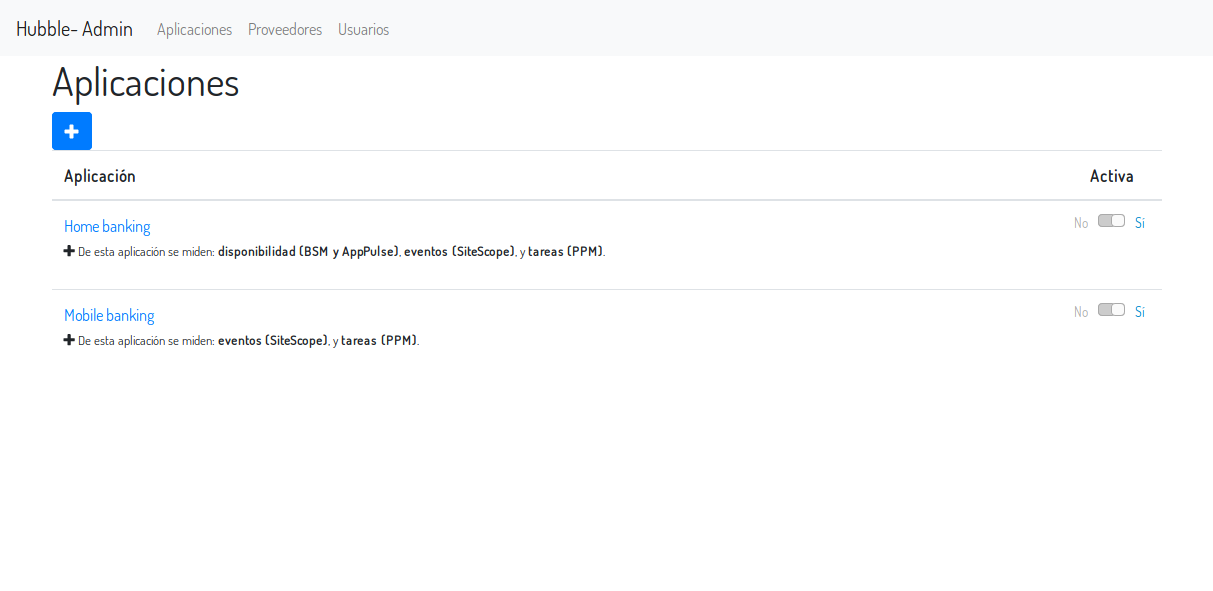
"roles": ["ADMINISTRATOR"],

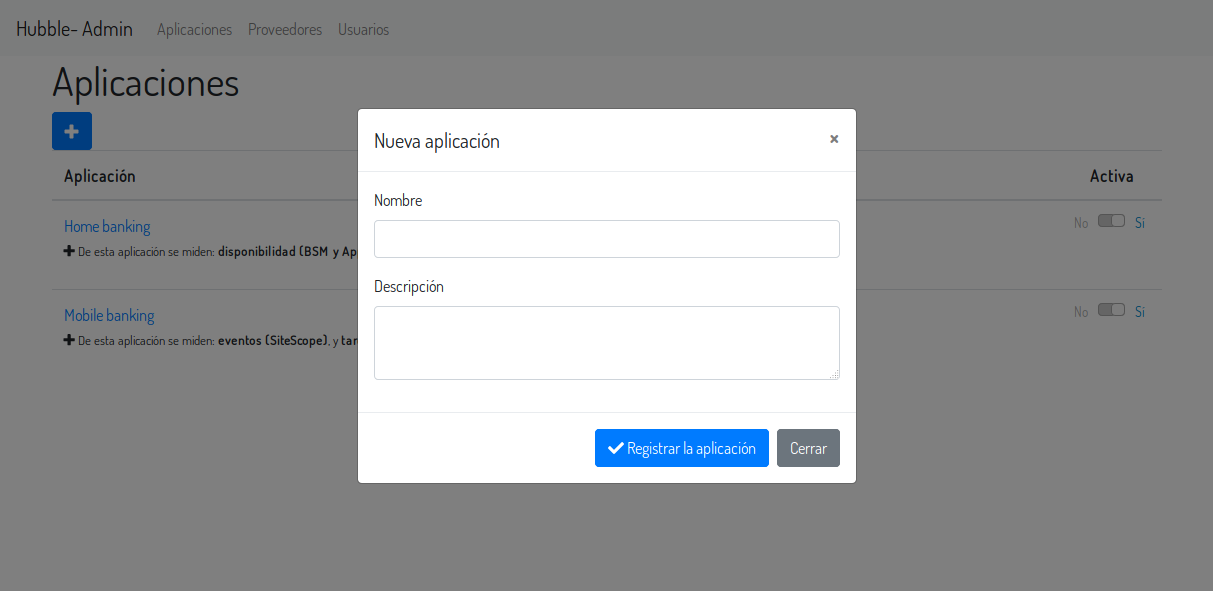
"applications": [“home\_banking”]

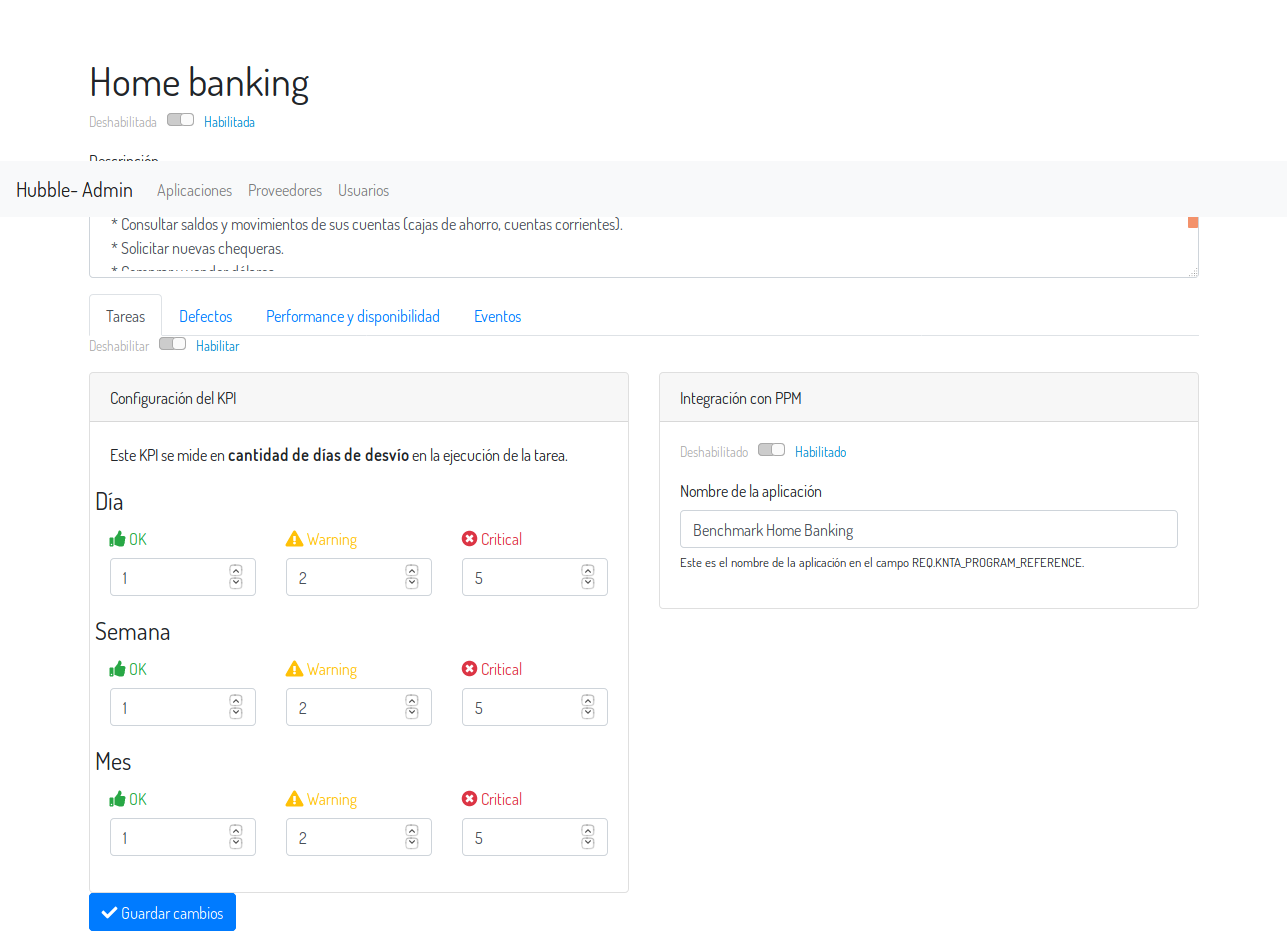
}

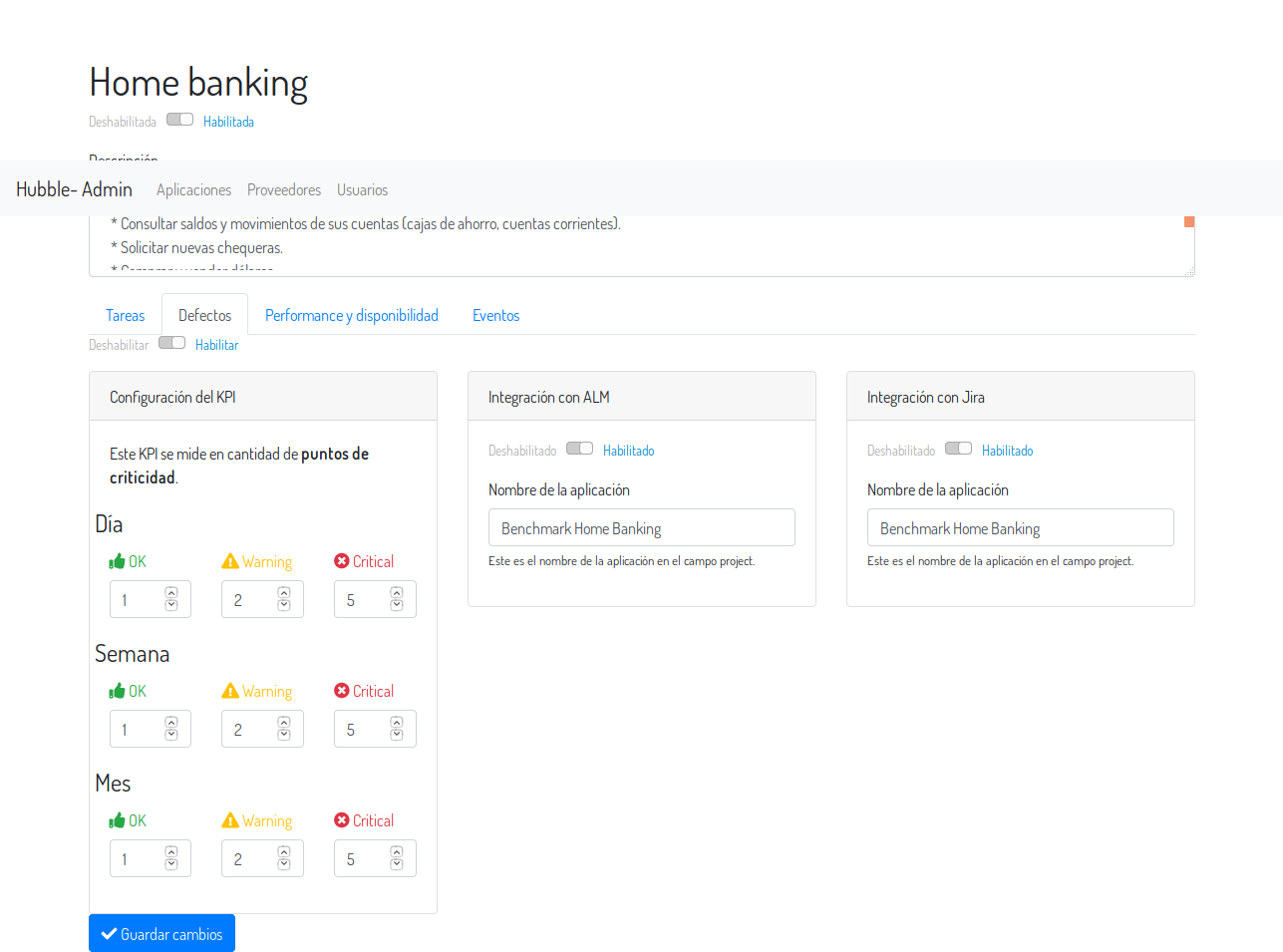
# Prototipos

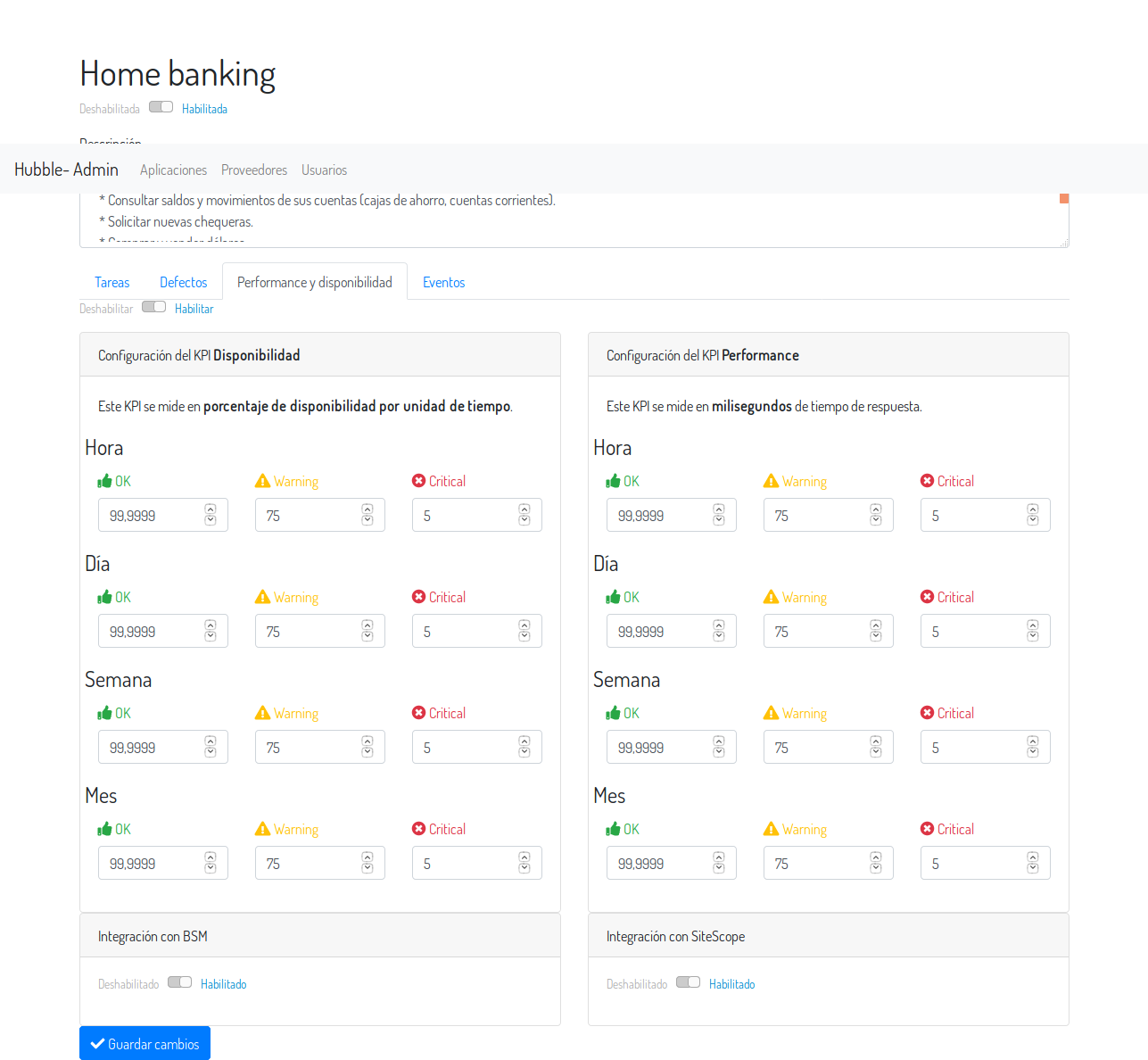
A continuación se incluyen los prototipos de las pantallas de administración.

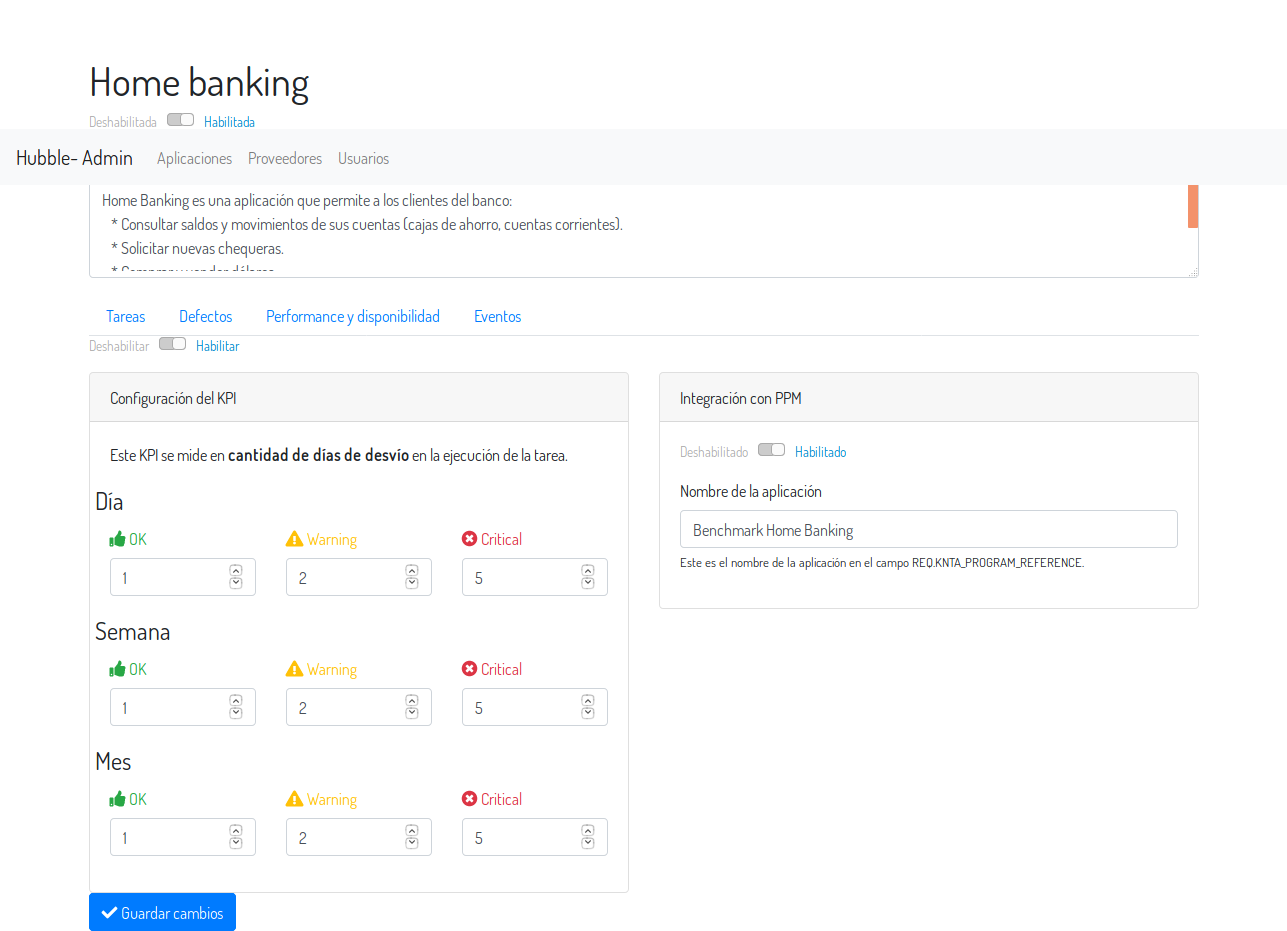


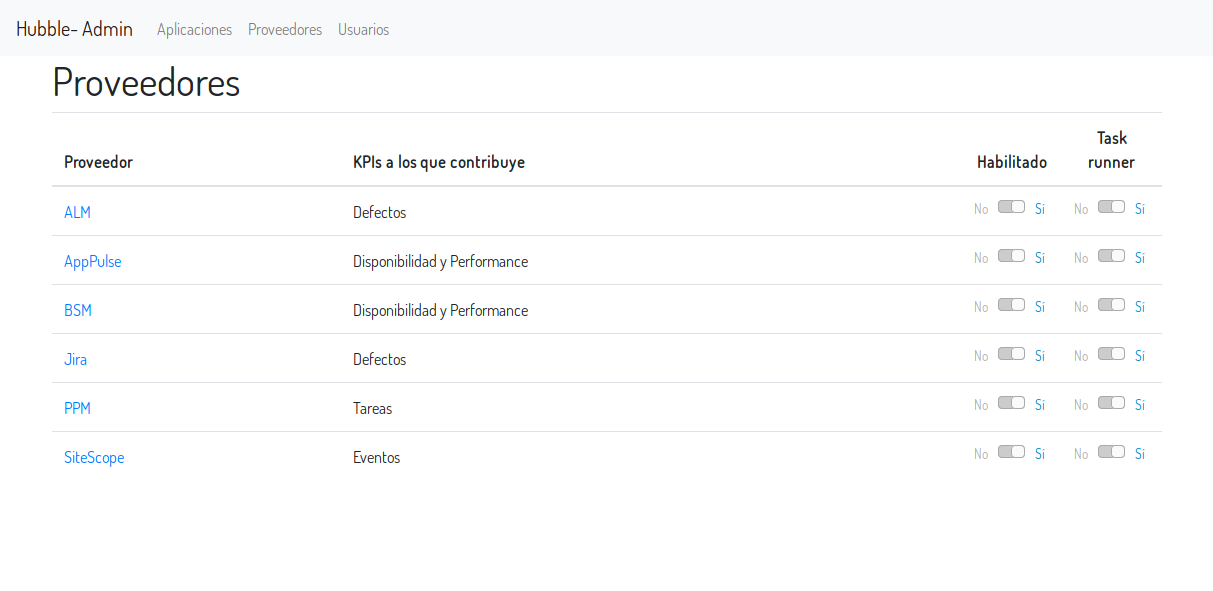


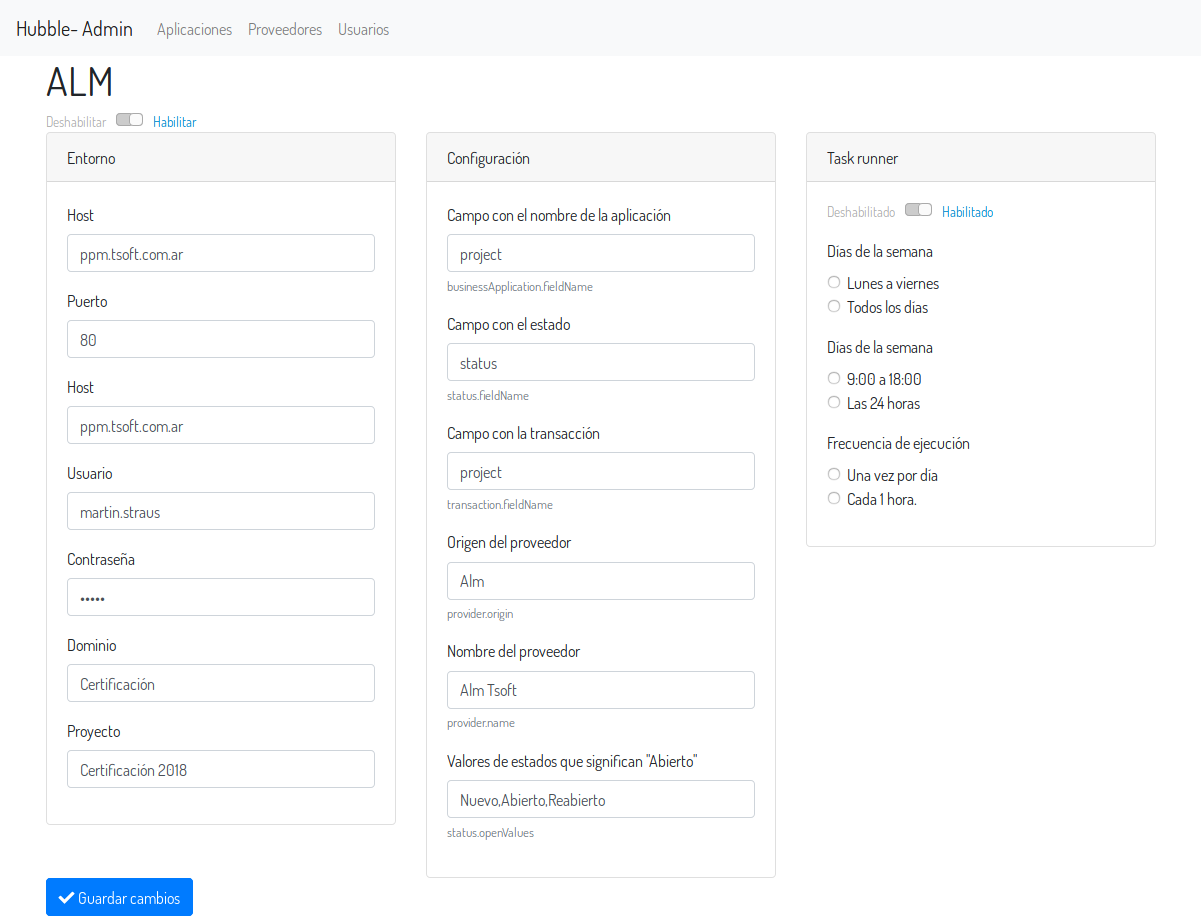


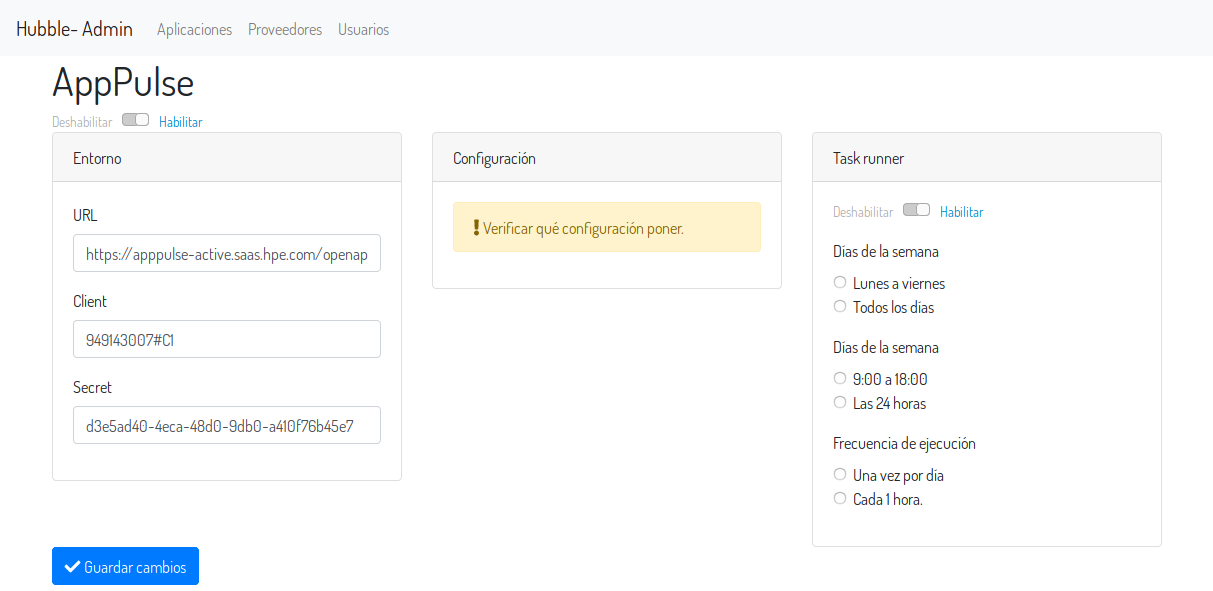


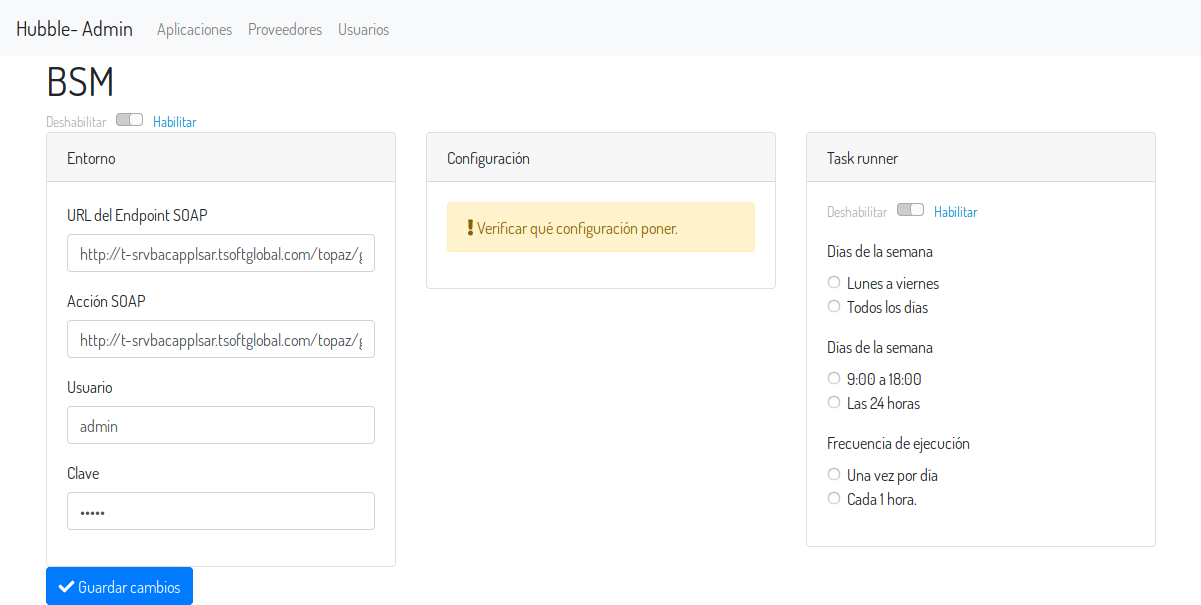


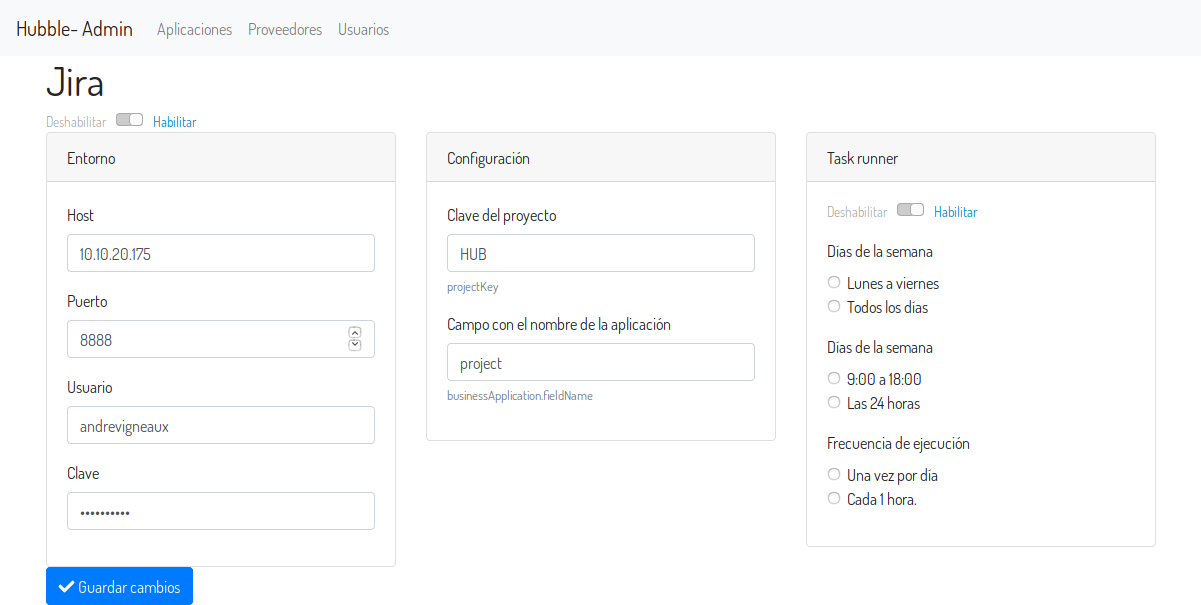


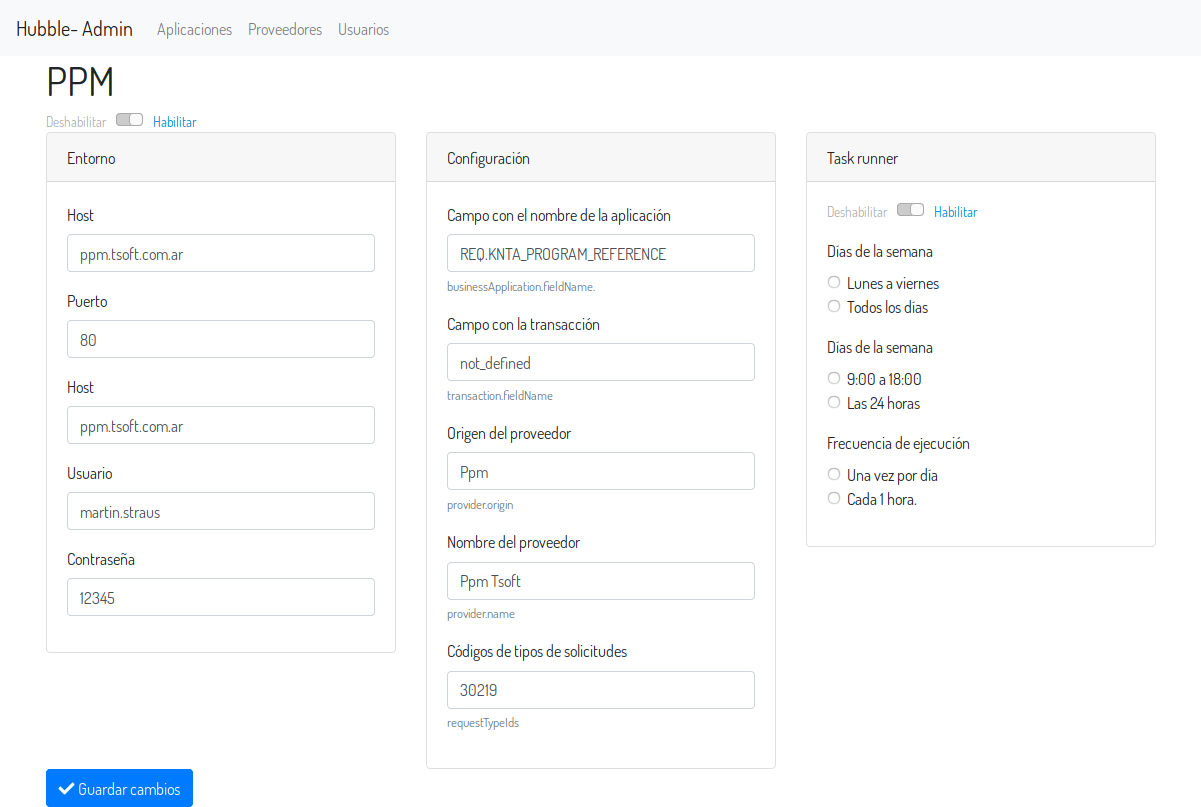


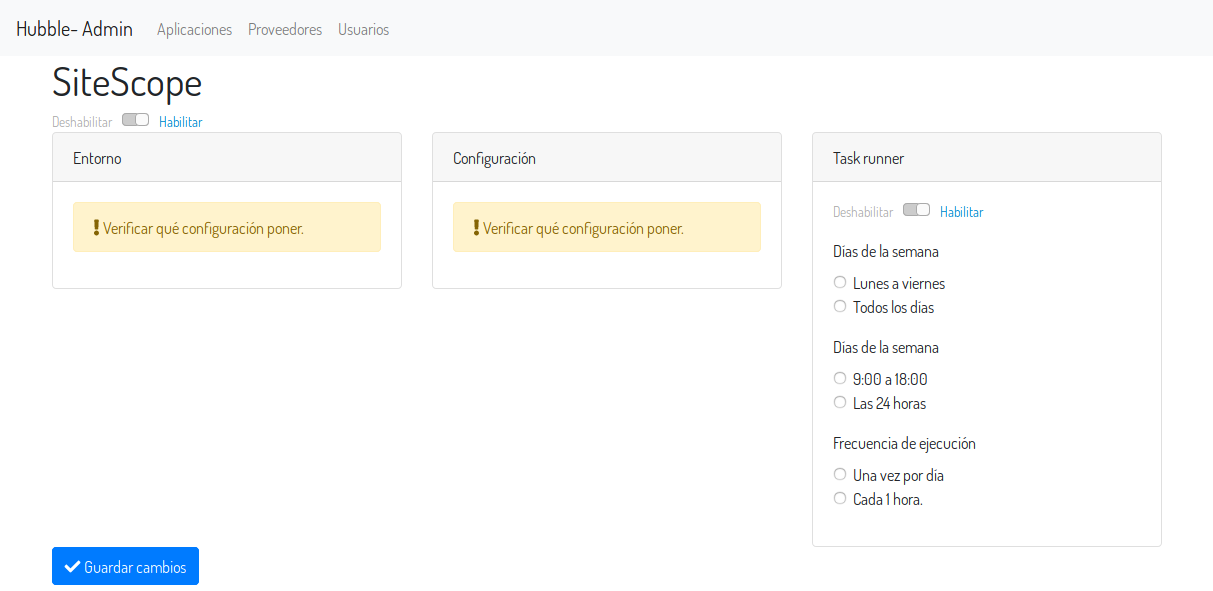


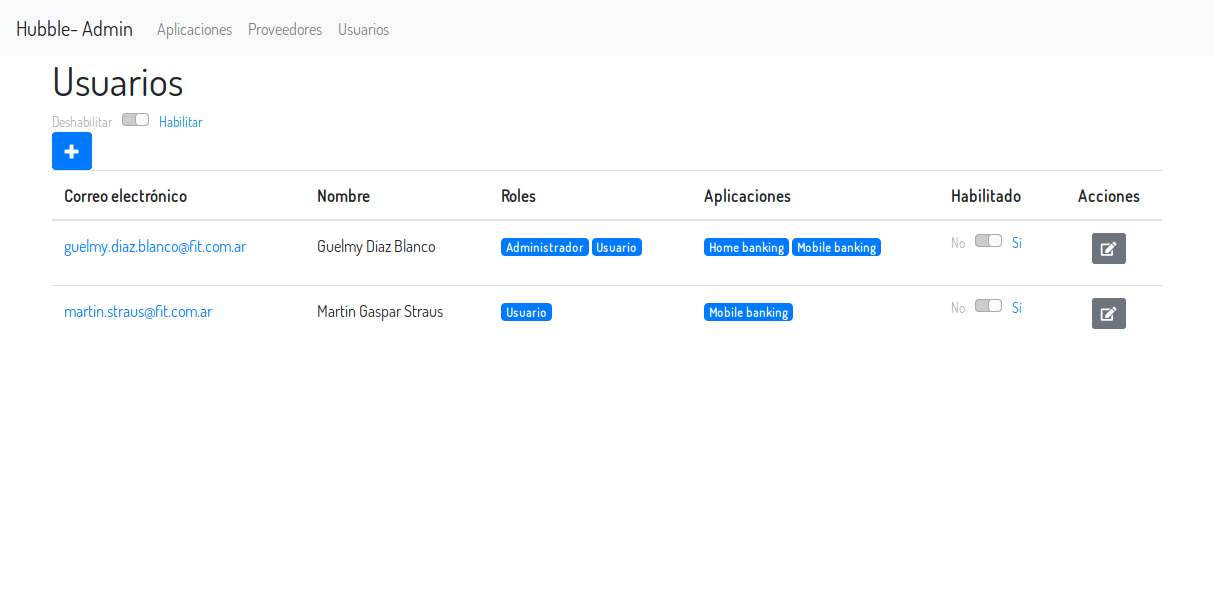


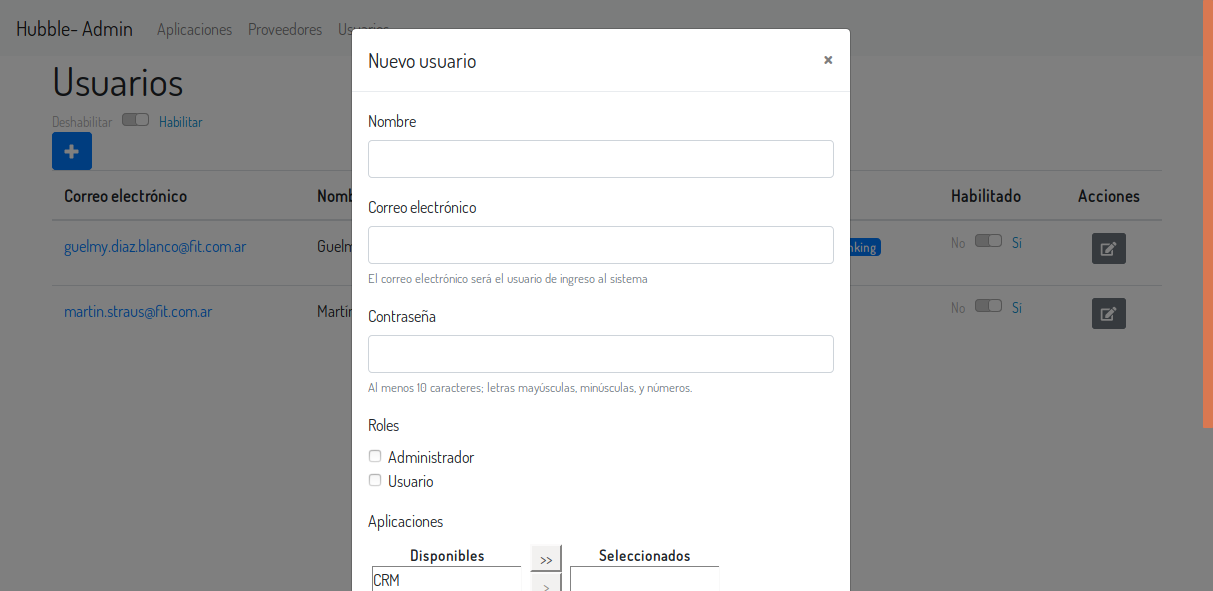


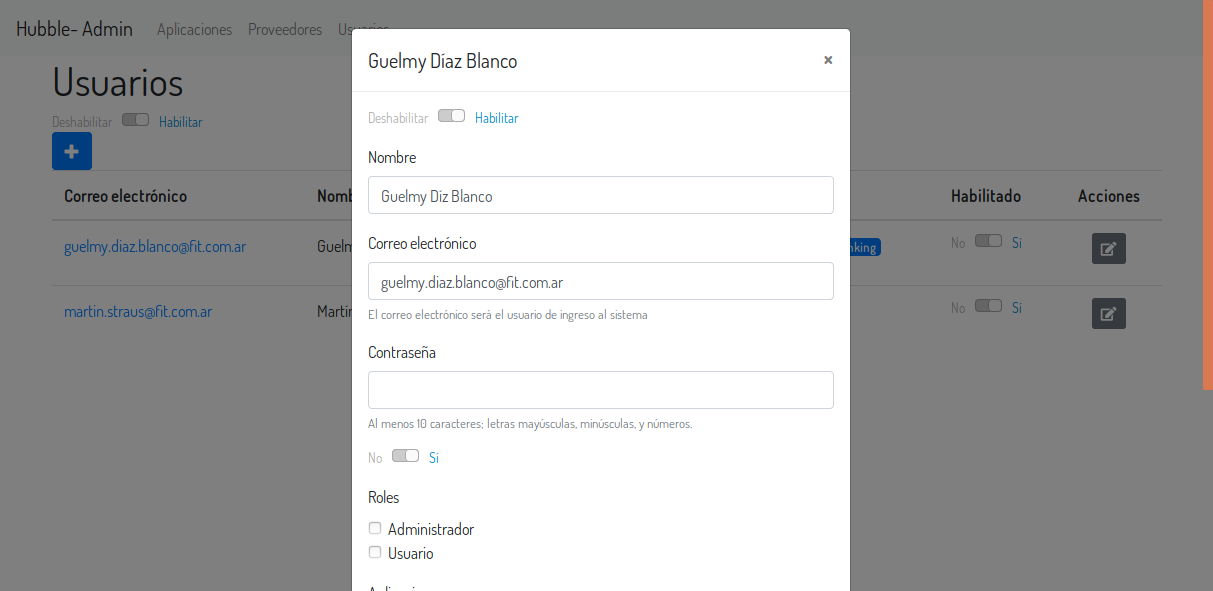






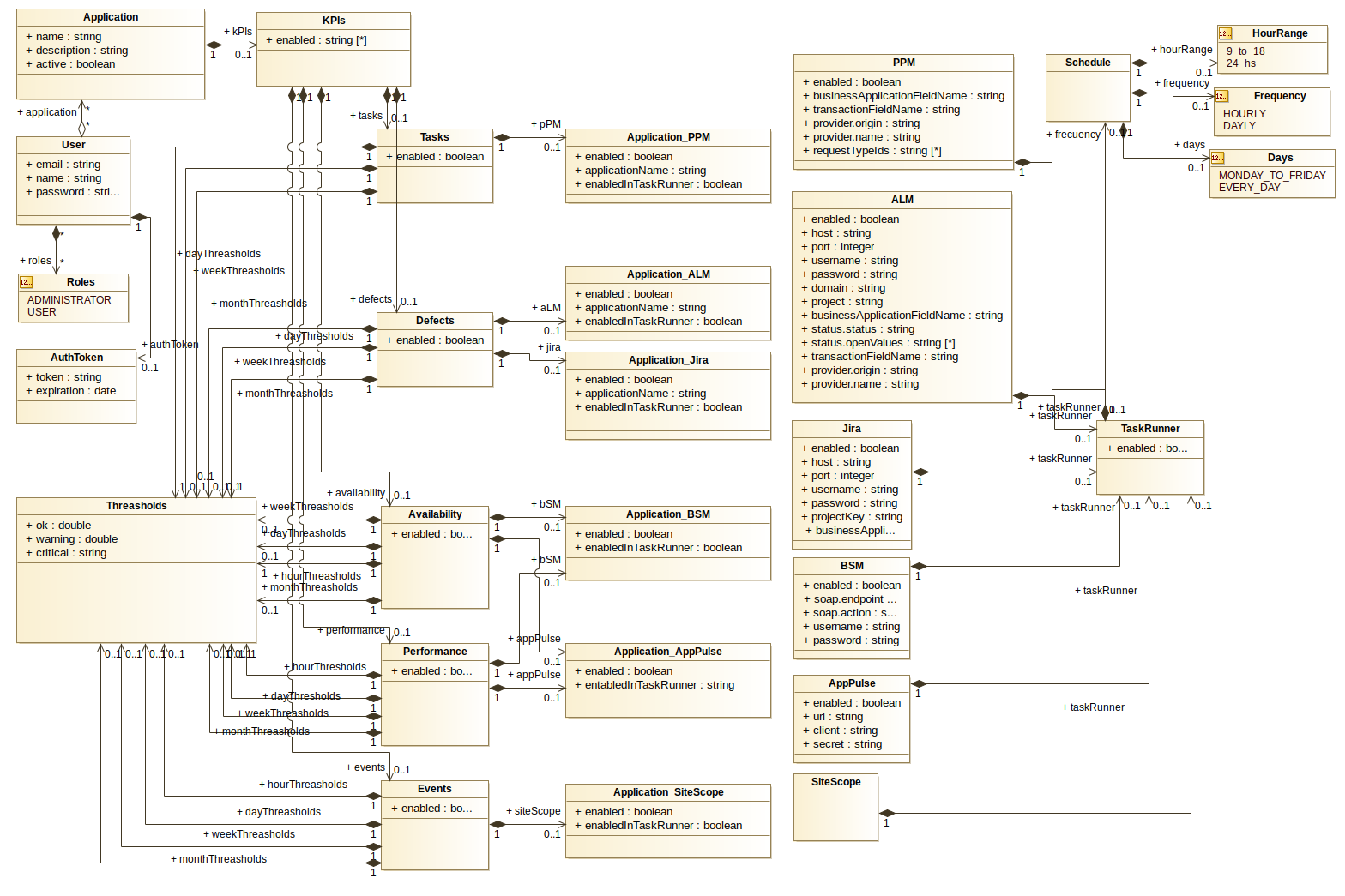






# Colecciones MongoDB

El siguiente diagrama representa todas las entidades agregadas o modificadas para las funciones de administración.



Las entidades serán almacenadas en las siguientes colecciones de MongoDB:

* applicationStorage
* providerStorage
* userStorage

## Aplicaciones

La colección se llama \_\_applicationStorage\_\_. Registra las aplicaciones administradas en Hubble.

applications {

name: "string",

description: "string",

active: "boolean",

kpis: {

enabled: "string[]",

tasks: {

thresholds: {

normal: "integer",

warning: "integer",

critical: "integer"

},

providers: {

ppm: {

enabled: "boolean",

applicationName: "string",

enabledInTaskRunner: "boolean"

}

}

},

defects: {

thresholds: {

normal: "integer",

warning: "integer",

critical: "integer"

},

providers: {

alm: {

enabled: "boolean",

applicationName: "string",

enabledInTaskRunner: "boolean"

},

jira: {

enabled: "boolean",

applicationName: "string",

enabledInTaskRunner: "boolean"

}

}

},

availability: {

thresholds: {

normal: "double",

warning: "double",

critical: "double"

},

providers: {

bsm: {

enabled: "boolean",

enabledInTaskRunner: "boolean"

}

}

},

performance: {

thresholds: {

normal: "double",

warning: "double",

critical: "double"

},

providers: {

appPulse: {

enabled: "boolean",

enabledInTaskRunner: "boolean"

}

}

},

events: {

thresholds: {

normal: "integer",

warning: "integer",

critical: "integer"

},

providers: {

siteScope: {

enabled: "boolean",

enabledInTaskRunner: "boolean"

}

}

}

}

}

Ejemplo

{

\_id: 1

name: "Home Banking",

description: "Home Banking es una aplicación que permite....",

active: true,

kpis: {

enabled: ["tasks", "defects", "availability", "performance", "events"],

tasks: {

thresholds: {

normal: 1,

warning: 2,

critical: 5

},

providers: {

ppm: {

enabled: true,

applicationName: "Benchmark Home Banking",

enabledInTaskRunner: true

}

}

},

defects: {

thresholds: {

normal: 1,

warning: 2,

critical: 5

},

providers: {

alm: {

enabled: true,

applicationName: "Benchmark Home Banking",

enabledInTaskRunner: true

},

jira: {

enabled: true,

applicationName: "Benchmark Home Banking",

enabledInTaskRunner: true

},

}

},

availability: {

thresholds: {

normal: 99.9999,

warning: 75,

critical: 50

},

providers: {

bsm: {

enabled: true,

enabledInTaskRunner: true

}

}

},

performance: {

thresholds: {

normal: 1.0,

warning: 10.0,

critical: 100.0

},

providers: {

appPulse: {

enabled: true,

enabledInTaskRunner: true

}

}

},

events: {

thresholds: {

normal: 1,

warning: 2,

critical: 5

},

providers: {

siteScope: {

enabled: true,

enabledInTaskRunner: true

}

}

}

}

}

## Proveedores

La colección de proveedores tiene pocos atributos comunes.

{

name: "string",

enabled: "boolean",

environment: "document",

configuration: "document",

taskRunner: {

enabled: "boolean",

cronExpression: "string"

}

}

Los atributos puntuales de los atributos \_environment\_ y \_configuration\_ dependerán de cada proveedor.

### ALM

{

name: "ALM",

enabled: true,

environment: {

host: "ppm.tsoft.com.ar",

port: 80,

username: "martin.straus",

password: "",

domain: "TSAR\_SOFTWAREFACTORY",

project: "HUBBLE"

},

configuration: {

businessApplicationFieldName: "project",

status: {

status: "status",

openValues: ["Nuevo", "Abierto", "Reabierto"]

},

transactionFieldName: "project"

provider: {

origin: "Alm",

name: "Alm Tsoft",

}

},

taskRunner: {

enabled: true,

cronExpression: "0 0 0 \* \* \*"

}

}

### AppPulse

{

name: "AppPulse",

enabled: true,

environment: {

url: "https://apppulse-active.saas.hpe.com/openapi/rest/v1/949143007/",

client: "949143007#C1",

secret: "d3e5ad40-4eca-48d0-9db0-a410f76b45e7"

},

configuration: {},

taskRunner: {

enabled: true,

cronExpression: "0 0 0 \* \* \*"

}

}

### BSM

{

name: "BSM",

enabled: true,

environment: {

soap: {

endpoint: "https://apppulse-active.saas.hpe.com/openapi/rest/v1/949143007/",

action: "http://t-srvbacapplsar.tsoftglobal.com/topaz/gdeopenapi/services/GdeWsOpenAPI",

},

username: "admin",

password: "admin"

},

configuration: {},

taskRunner: {

enabled: true,

cronExpression: "0 0 0 \* \* \*"

}

}

### Jira

{

name: "Jira",

enabled: true,

environment: {

host: "10.10.20.175",

port: 8888,

username: "andrevigneaux",

password: ""

},

configuration: {

projectKey: "HUB",

businessApplicationFieldName: "project"

},

taskRunner: {

enabled: true,

cronExpression: "0 0 0 \* \* \*"

}

}

### PPM

{

name: "PPM",

enabled: true,

environment: {

host: "ppm.tsoft.com.ar",

port: 80,

username: "admin",

password: "ppm931"

},

configuration: {

businessApplicationFieldName: "project"

transactionFieldName: "not\_defined",

provider: {

origin: "Ppm",

name: "Ppm Tsoft",

},

requestTypeIds: [30219]

},

taskRunner: {

enabled: true,

cronExpression: "0 0 0 \* \* \*"

}

}

### SiteScope

{

name: "SiteScope",

enabled: true,

environment: {},

configuration: {},

taskRunner: {

enabled: true,

cronExpression: "0 0 0 \* \* \*"

}

}

## Usuarios

Los usuarios se almacenarán en una colección llamada "users".

{

email: "string",

name: "string",

password: "string",

roles: "string[]",

applications: "integer[]"

}

Ejemplo

{

email: "martin.straus@fit.com.ar",

name: "Martín Gaspar Straus",

password: "",

roles: ["administrator"", "user"],

applications: [1, 2, 3]

}

## APIs

Las APIs están documentadas siguiendo el estándar OpenAPI[[3]](#footnote-3) y haciendo uso de SpringFox, una librería que ofrece tanto el documento de especificación Swagger como una interfaz web que permite describir e interactuar con el API REST.

Con el backend ejecutándose se podrá acceder a través de las siguientes urls:

Documento de especificación swagger:

* <http://localhost:8080/v2/api-docs?group=Applications>
* <http://localhost:8080/v2/api-docs?group=Providers>
* <http://localhost:8080/v2/api-docs?group=Users>

Interfaz interactiva:

* <http://localhost:8080/swagger-ui.html>

Desde acá será posible alternar entre los diferentes grupos de APIs, pudiendo incluso interactuar con los diferentes operaciones que exponen.

1. <https://en.wikipedia.org/wiki/Argon2> [↑](#footnote-ref-1)
2. <https://www.owasp.org/index.php/Password_Storage_Cheat_Sheet#Argon2_usage_proposal_in_Java> [↑](#footnote-ref-2)
3. <https://github.com/OAI/OpenAPI-Specification> [↑](#footnote-ref-3)