MODUSER\_WEB

**Manual Tecnico de Desarrollo – Telecel S.A.**

Versión 2.0

MICROSOLUTION

MICRIUM, Soluciones Tecnológicas

Contenido

[Sobre este manual 3](#_Toc37944265)

[Introducción 3](#_Toc37944266)

[Alcance y público 3](#_Toc37944267)

[Documentos relacionados 3](#_Toc37944268)

[Historia de cambios 3](#_Toc37944269)

[1. Vista de La Arquitectura. 4](#_Toc37944270)

[ Vista de Desarrollo 4](#_Toc37944271)

[2. Configuración Acceso a Datos 5](#_Toc37944272)

[3. Incorporar Nuevo Parámetro al Sistema 6](#_Toc37944273)

[4. Autenticación con Active Directory (caso de estudio Tigo) 8](#_Toc37944274)

[5. Backend 8](#_Toc37944275)

[1) bo.com.micrium.modulobase.ldap 9](#_Toc37944276)

[2) bo.com.micrium.modulobase.model 9](#_Toc37944277)

[3) bo.com.micrium.modulobase.repository 9](#_Toc37944278)

[4) bo.com.micrium.modulobase.resources 10](#_Toc37944279)

[5) bo.com.micrium.modulobase.resources.dto y bo.com.micrium.modulobase.resources.dto.jwt 10](#_Toc37944280)

[6) bo.com.micrium.modulobase.security.jwt 11](#_Toc37944281)

[7) bo.com.micrium.modulobase.service 11](#_Toc37944282)

[8) bo.com.micrium.modulobase.util 11](#_Toc37944283)

[9) bo.com.micrium.modulobase.util.abm 12](#_Toc37944284)

[10) bo.com.micrium.modulobase.validator 12](#_Toc37944285)

[6. Frontend 13](#_Toc37944286)

**Sobre este manual**

# Introducción

Este manual describe la parte técnica de desarrollo que se deberá considerar al momento de iniciar un nuevo proyecto web, tomando como proyecto base el que se describa aquí, es necesario que el lector tenga conceptos básicos tecnologías web en particular de servidores de aplicaciones web java como Jboss, tomcat, weblogic.

# Alcance y público

Orientado al personal del área de desarrollo de software, al revisar este manual podrá entender y comenzar a trabajar en un nuevo proyecto web.

# Documentos relacionados

Manual de Implementación

Manual de Operaciones

# Historia de cambios

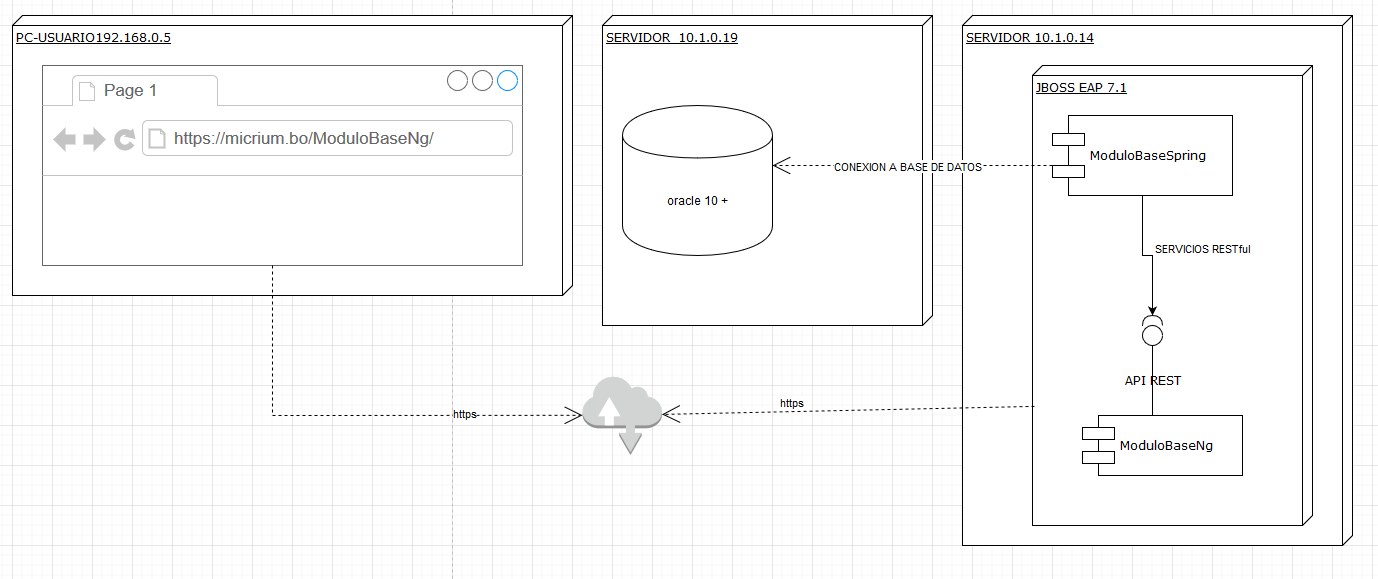
|  |  |  |  |
| --- | --- | --- | --- |
| **Rev.** | **Fecha** | **Autor** | **Descripción** |
| 1.0 | 16/04/2020 | Alex J. Paco | Creación del Documento |
|  |  |  |  |
|  |  |  |  |

## Vista de La Arquitectura.

Las aplicaciones Web son aplicaciones distribuidas. Las aplicaciones web integran un conjunto de tecnologías y representar la arquitectura se vuelve algo complejo y no es el objetivo de este manual, sabemos que existen varios tipos de Vistas como Vista Lógica, de Procesos, Física y la Vista de Desarrollo. En esta sección veremos la vista de Desarrollo que es la que nos interesa.

### Diagrama de despliegue

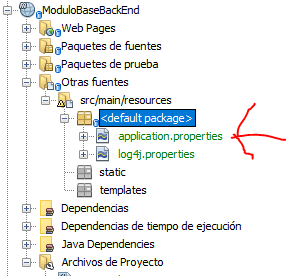
Esta vista muestra los distintos artefactos y su relación entre ellos.



## Configuración Acceso a Datos

El proyecto Base usa Spring framework para el acceso a base de datos, para el manejo del pool de conexiones usa HikariCP, se debe tomar en cuenta las siguientes configuraciones:

* Configuración del Archivo *aplication.properties* lo podemos encontrarjunto con el código fuente



* Establecer las propiedades según el tipo de BD:

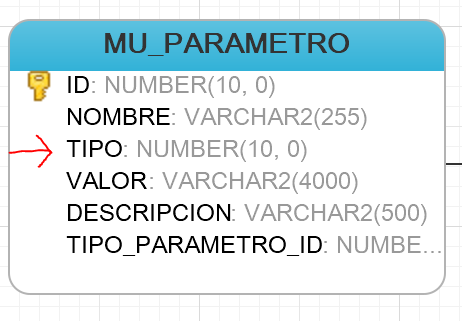


## Incorporar Nuevo Parámetro al Sistema

El objetivo de esta funcionalidad es evitar manejar archivo de properties en el sistema, dado que para modificar un parámetro es necesario el reinicio de la apliacion web cuando se usa archivos de properties, para evitar esto se creó una Tabla de Parametros en BD que guardará los parámetros del sistema.

A continuación mostraremos los pasos para incorporar un parámetro y su uso en tiempo de diseño dentro el proyecto.

* Inserte el parámetro en la tabla “MU\_PARAMETRO”, debe tomar en cuenta el tipo de dato que manejara el parámetro dentro el sistema.



*Tabla de Parámetros*

El campo tipo define el tipo de dato que el parámetro será

**Tipo**=1. Es una Cadena

**Tipo**=2. Es una Fecha

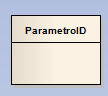
**Tipo**=3. Es una Numérico, puede ser un número entero o un número real.

**Tipo**=4. Es una Booleano

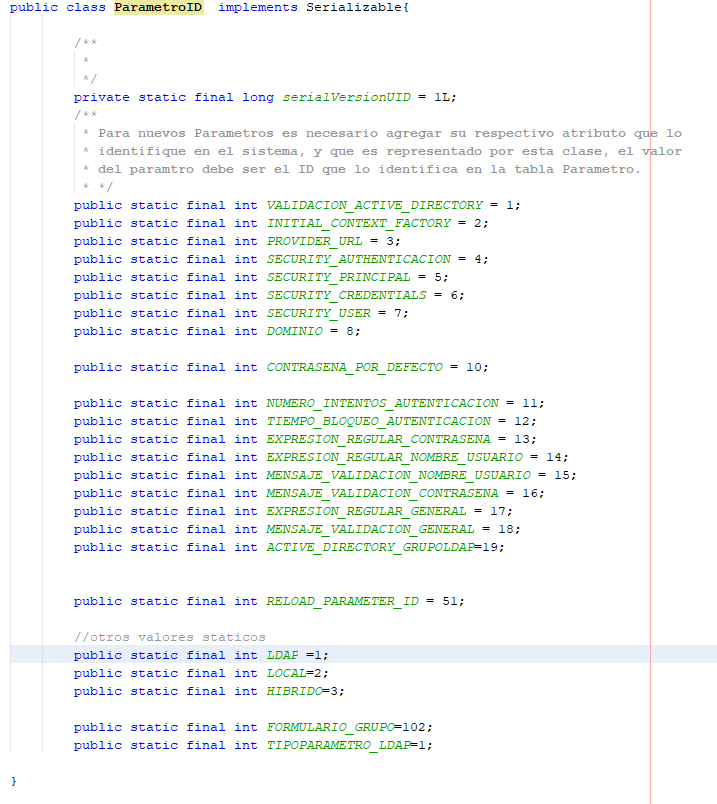
**Tipo**=5. Es una Color

**Tipo**=6. Es una lista de cadenas.

* Agregue el id del parámetro como atributo nuevo de la clase “ParametroID”, asigne un alias al parámetro y que por recomendación sea la misma que está en la BD ósea el nombre del parámetro, también por recomendación trate de definir un formato del nombre para los parámetros nuevos como por ejemplo vaya de lo más general a lo más específico y que trate de mantener una categoría



*Clase java Parámetros*



*Vista de Parámetros del sistema*

## Autenticación con Active Directory (caso de estudio Tigo)

Las empresas suelen tener sus propios sistemas de autenticación u otros sistemas que hacen esta validación, de tal modo que los nuevos sistemas se deben integrar mediantes interfaces de entradas y salidas.

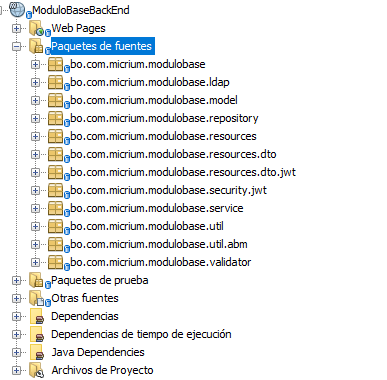
En el caso de estudio Tigo se tiene un sistema de Active Directory al cual el sistema debe integrarse y dejar que el sistema de Tigo LDAP haga la autenticación de las credenciales del usuario final y dependiendo su respuesta se le dara acceso al sistema o no.

## Backend

El backend contiene toda la lógica de negocio del sistema. Donde este expone servicios RESTful para ser consumidos por el Frontend u otros sistemas externos.

Entre estos servicios existe un servicio de autenticación en base a usuario y contraseña, si las credenciales son válidas este retorna un token de autorización para el consumo de los otros servicios. El método de autorización de los servicios está basado en tokens.

El backend está construido usando el framework Spring. El patrón de diseño está basado en la documentación oficial de Spring.



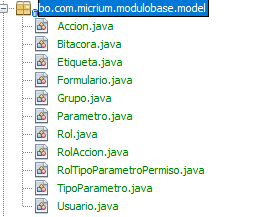
## bo.com.micrium.modulobase.ldap

Este paquete contiene las clases para la autenticación con el LDAP de Tigo.



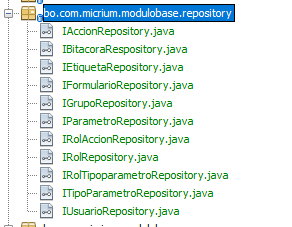
## bo.com.micrium.modulobase.model

Este paquete contiene las entidades o pojos, que son la representación en java de las tablas de base de datos



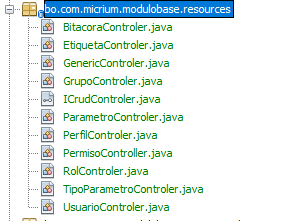
## bo.com.micrium.modulobase.repository

Este paquete contiene los repositorios, que son las interfaces para el acceso datos. Mediante los estándares de Spring se realizan consultas de manera sencilla y rápida.



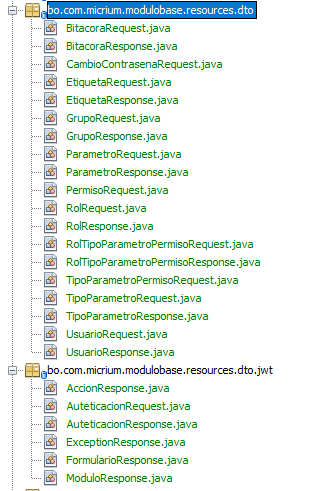
## bo.com.micrium.modulobase.resources

Este paquete contiene los controladores o servicios RESTful, que son las interfaces de salida y entrada del sistema con otros sistemas externos como el Frontend.



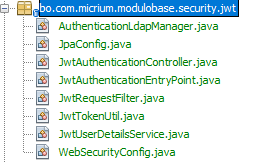
## bo.com.micrium.modulobase.resources.dto y bo.com.micrium.modulobase.resources.dto.jwt

Este paquete contiene los dtos de los servicios RESTful, que son los objetos que represetan los json de entrada y salida. (dto = data transfer object)



## bo.com.micrium.modulobase.security.jwt

Este paquete contiene las clases que proporcionan la seguridad del sistema, la autenticación y la autorizacion, la autenticación está basada en un usuario y contraseña y la autorización en tokens.



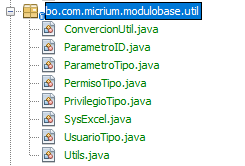
## bo.com.micrium.modulobase.service

Este paquete contiene las clases con lógica de negocio complejos o con características especiales tanto para el acceso a datos o consumo de servicios externos.



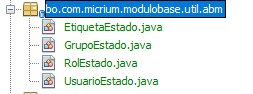
## bo.com.micrium.modulobase.util

Este paquete contiene las de utilidad general en todo el sistema.



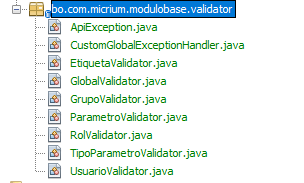
## bo.com.micrium.modulobase.util.abm

Este paquete contiene las de utilidad de los ABMS’s del sistema en general.

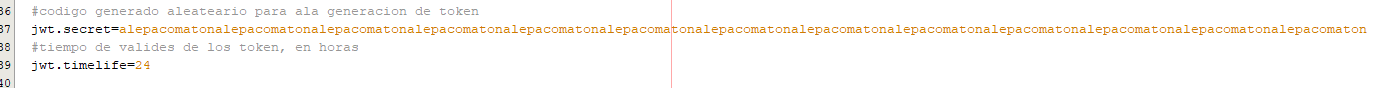


## bo.com.micrium.modulobase.validator

Este paquete contiene las clases validadores de las entradas y salidas de los servicios RESTful y el manejo de errores.



En el archivo de aplication.properties existen algunos parámetros importantes a tener consideración.



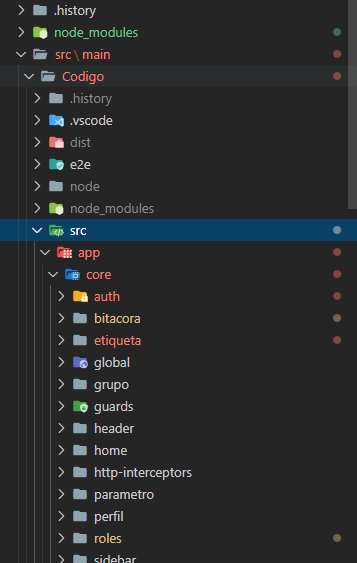
jwt.secret = se recomienda cambiar a una cadena aleatoria de solo caracteres como letras y que tenga la misma longitud de la cadena actual.

Jwt.timelife = tiempo en horas de la valides de los tokens de autorización.

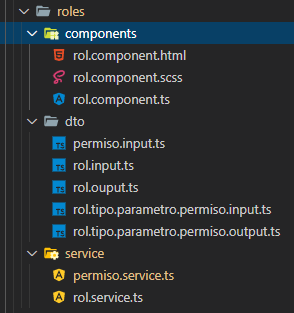
Nota.- Cualquier duda o dificultad se recomienda revisar la documentación oficial de Spring.

## Frontend

El frontend está basado en el framework Angular 8 y PrimeNg. Es una aplicación SPA (Simple Page Application) que se alimenta de datos a travez de un API REST.

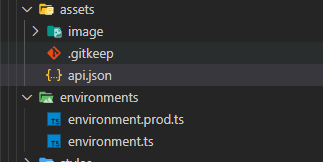


El proyecto está basado en componentes.

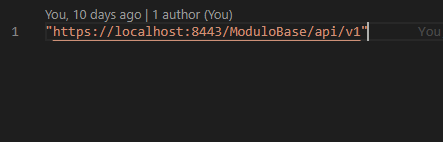


* components.- para el manejo visual de los datos
* dto.- clases typescript que son los mapeos json de las entradas y salidas de la API REST
* service.- llamada a los servicios de la API REST

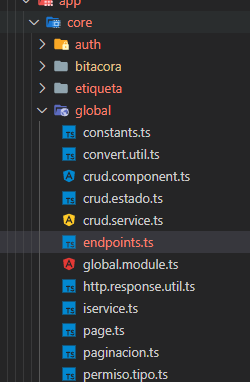
Configuración de la dirección de los servicios RESTful

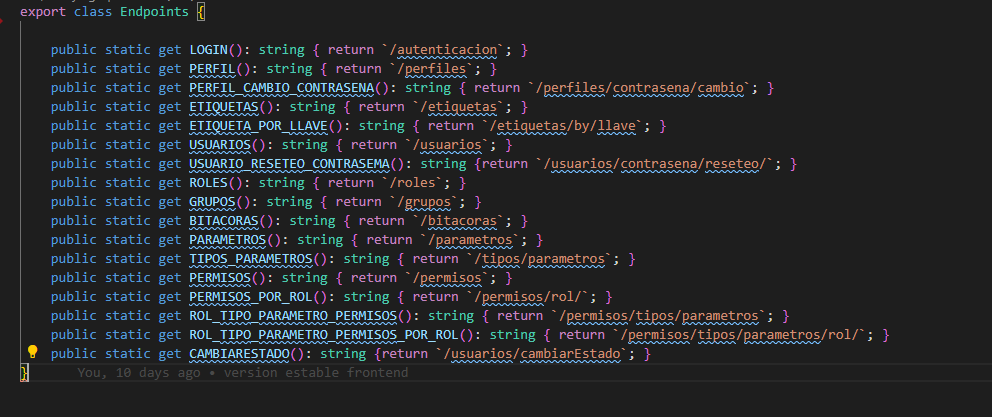


Abrir el archivo api.json



Las direcciones de los servicios REST se encuentran el archivo endpoints.ts

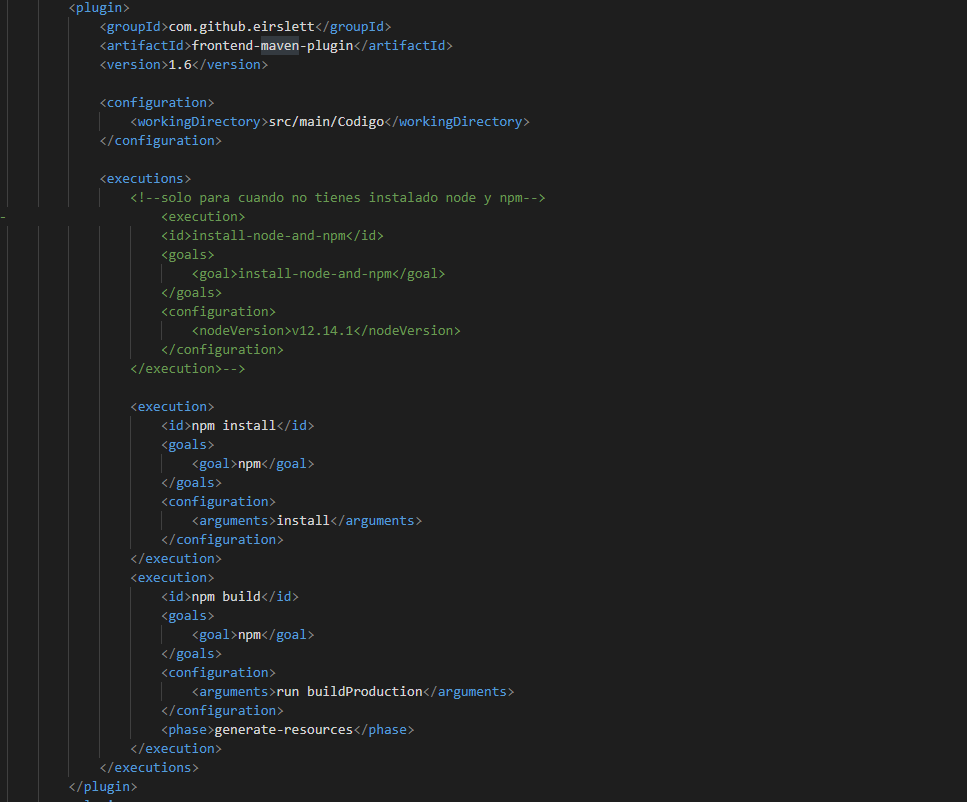




Para el desarrollo se puede levantar el frontend de la forma en que angular usa

ng s

para ambiente de producción hay que contruir el war para su despliegue, ir al pom.xml del proyecto



El plugin frontend-maven-plugin genera los archivos de angular para producción e instala en el war para luego ser desplgado en un servidor de aplicaciones.

La parte que esta comentado, es la instalación de node js en el caso que no se lo tenga instalado, solo se debería ejecutar una sola vez. En el caso de tenerlo instalado no es necesario y se lo tiene comentado. Los otros comandos son para la construcción de los archivos angular para producción.