



FURY - JAVA

SETUP & DEPLOY

Índice



01

Crear Aplicación
Fury

02

Importar proyecto
IntelliJ

03

Crear Base de Datos
Fury-DataBase

04

Configurar Base de Datos
IntelliJ

05

Levantar cambios
Fury-GIT

06

Crear versión
Fury-GIT

07

Crear SCOPE
Fury-WEB y Deploy

08

Trabajar con la
aplicación Fury

IT BOARDING

BOOTCAMP

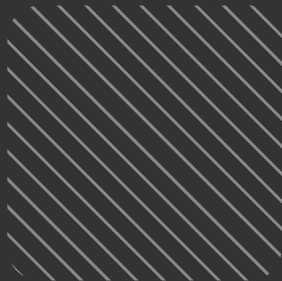


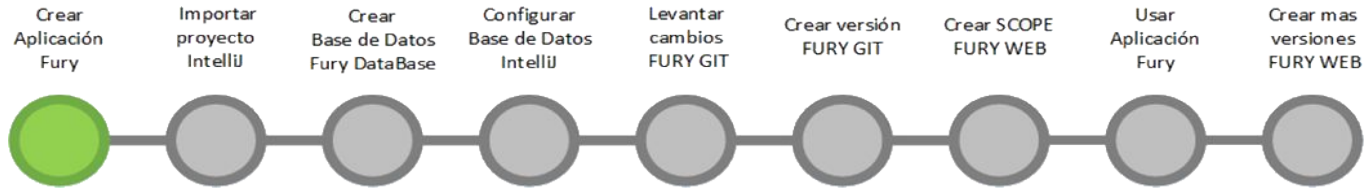
1

//Crear Aplicación Fury

IT BOARDING

BOOTCAMP






1 - Ingresar a la url de Fury y loguearse con el usuario provisto por Mercado Libre.

Fury

web.furycloud.io/login

Apps gmail Fury Mercadolibre S... Mi unidad - Go...

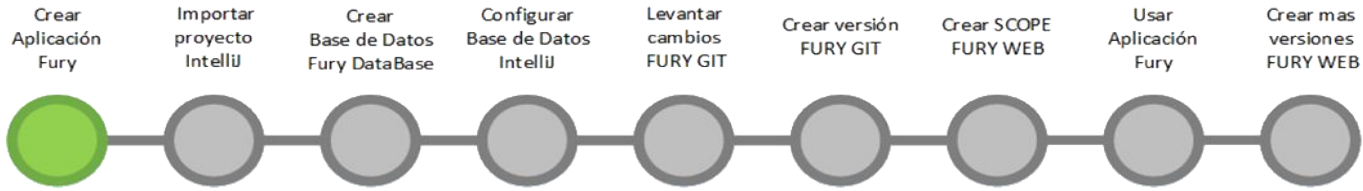
 **Fury**

Username

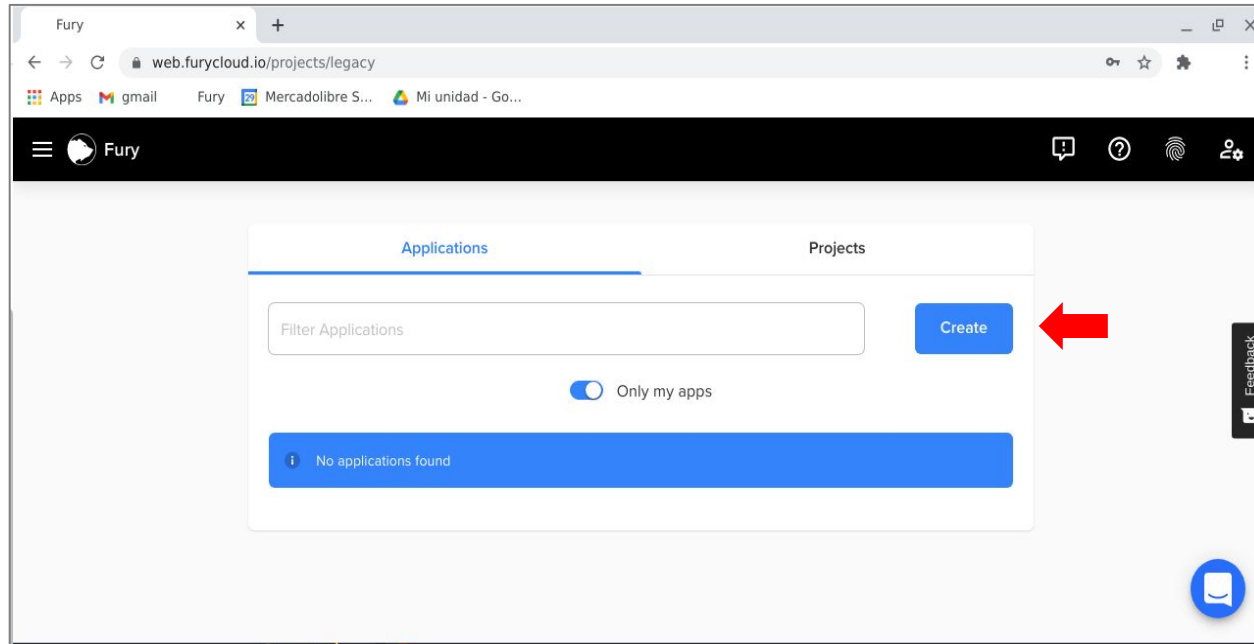
Password

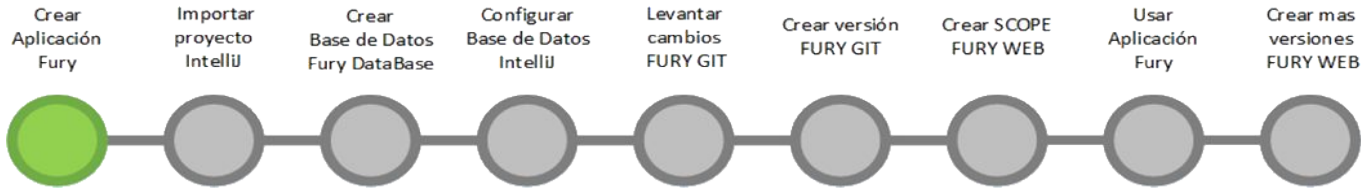
Log in

Feedback



2 - Presionar botón “Create”.





3 - Completar el formulario y presionar el botón “Save”.

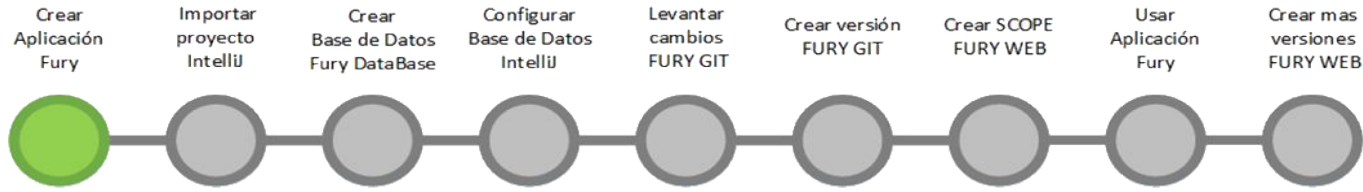
Indicar nombre de la App → **demo-app**

Indicar que es una Demo App → **yes**

Indicar tipo de App “Web” → **web**

Indicar Tecnología “Java” → **java**

Save **Cancel**



4 - Comprobar que el procedimiento termine de la siguiente manera.

The screenshot shows the Fury web interface for 'demo-app'. The interface includes a sidebar with navigation options and a main content area. The main content area displays 'demo-app' with a 'Created' date of 29/3/2021. It includes sections for 'Last activities' (showing no activity in the last 3 days), 'Services' (showing no services to display), and 'Policies' (listing several authorization policies). A green checkmark icon is overlaid on the right side of the screenshot, indicating successful completion.



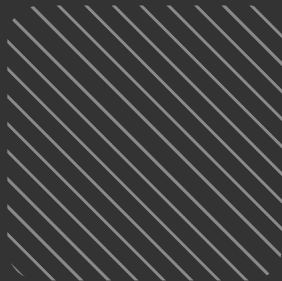
No son errores.
Son políticas.
Acciones no permitidas.

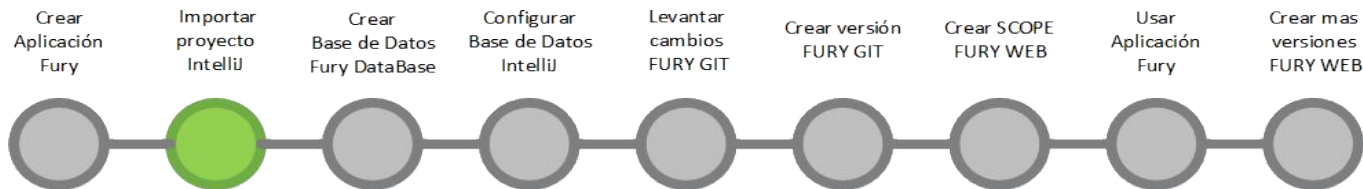
2

//Importar Proyecto en IntelliJ

IT BOARDING

BOOTCAMP





1 - Chequear que repositorio de GitHub este vacío (en Dashboard de Fury, clickear logo de Git).

2 - Abrir cualquier Consola/Terminal (IntelliJ, Mac, Fury).

3 - Ubicarse en la carpeta donde esten los proyectos locales de IntelliJ.

AR0C02DT4S6ML85:IdeaProjects [usuario]\$

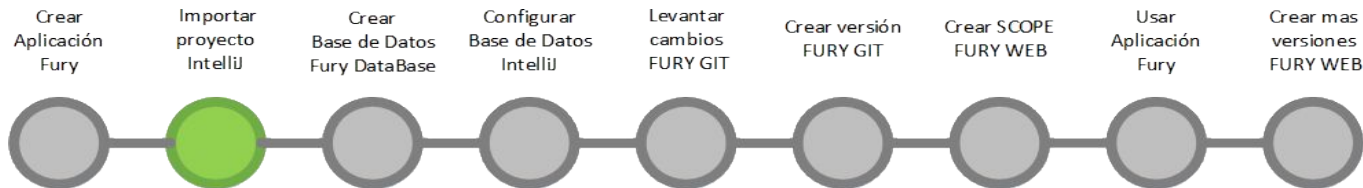
4 - Ejecutar el comando:

fury get [nombre de tu aplicación]

✖ ✖ ✖

✖ ✖ ✖

✖ ✖ ✖

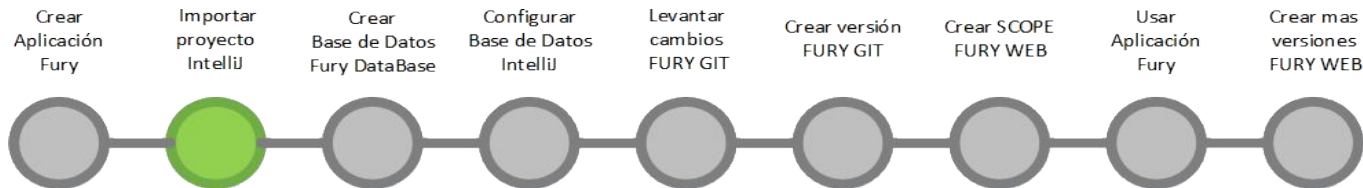


5 - Cuando pregunte seleccionar la opción “3. web-maven-spring”.

```
[AR0C02DT4UPML85:Fury mpromet$ fury get msp-demo-app
Validating if the application msp-demo-app exists
Cloning repository git@github.com:mercadolibre/fury_msp-demo-app.git
Repository cloned successfully
Configuration file created in /Users/mpromet/Bootcamp IT/Fury/msp-demo-app/.fury
Scaffolding project...
WARNING! If you interrupt this your project will be deleted
Cambiado a nueva rama 'feature/initial-scaffolding'
No template has been specified. Please select one from the list below:
1. web-gradle-spring
2. web-maven-spark
3. web-maven-spring [default]
Template to use (press ENTER for default) [3]:
```

6 - Importar proyecto a IntelliJ, esperar que Maven resuelva todas las dependencias.





TROUBLESHOOTING

ERROR → File "/Users/sgalvan/Library/Python/3.7/bin/fury", line 6, in <module> from furycli.cli import main ModuleNotFoundError: No module named 'furycli'

SOLUCIÓN → ver [furycli - installation](#)

ERROR → something went wrong: can only concatenate str (not "dict") to str

SOLUCIÓN →

- Actualizar la versión de fury con el comando "fury upgrade",
- Eliminar la carpeta de la app (si ya se creó)
- volver a ejecutar el comando "fury get [app_name]"

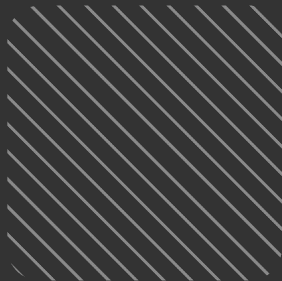


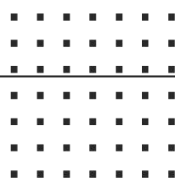
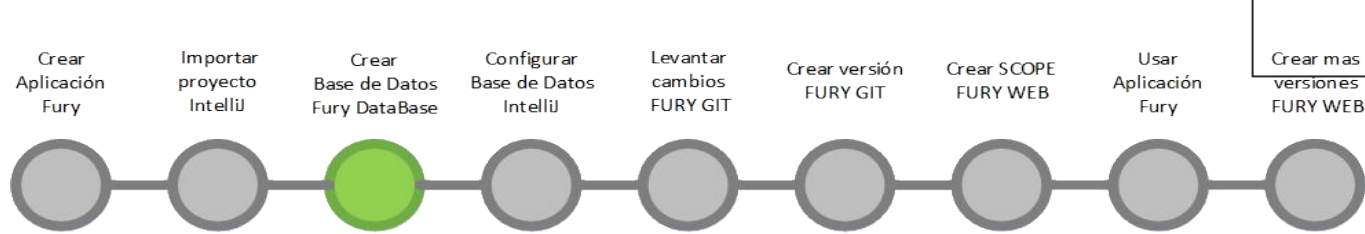
3

//Crear Base de Datos en Fury

IT BOARDING

BOOTCAMP





1 - En el dashboard de Fury (menú de la izquierda) clicar en Database.

The screenshot shows the Fury dashboard interface. The left sidebar contains a menu with the following items: Ops, Service Map, Infrastructure, Doctor, Logs, Traffic Catalog, Summary, Endpoints, Consume, Services, Audits, Bazooka, BigQ Consumer, BigQ Topics, Configurations, Cache, **Database** (highlighted with a red arrow), Documentation, Document Search, and FDA. The main content area displays the 'Database' section, which includes a 'Comparison between weeks' bar chart and a table of services.

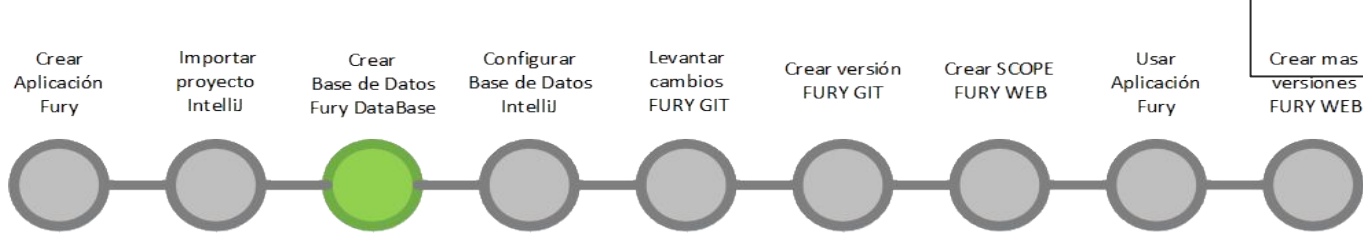
Comparison between weeks

Week	Tue 23/03	Wed 24/03	Thu 25/03	Fri 26/03	Sat 27/03	Sun 28/03	Mon 29/03
production	0.82	0.81	0.8	0.84	0.85	0.85	0.81
test	0	0.81	0.83	0.85	0.82	0.82	0.81

Services

Services	Last 21 days in USD	Week 1	Week 2	Week 3
Web Scope	15.04	5.13	5.78 (▲ 12.67%)	5.76 (▼ 0.35%)
TOTALS	15.04	5.13	5.78 (▲ 12.67%)	5.76 (▼ 0.35%)





2 - Completar el formulario y presionar el botón “Create”.

Fury - msp-demo-app

Create schema

Name	mspdemodb	→ Indicar el nombre de la App.
Description	DB for demo app	→ Indicar la descripción de la App (obligatorio).
Availability	Low: Application which is not required to be productive 7x24, downtime doesn't affect business or operation	→ Indicar en Disponibilidad “low”.
Integrity	Low: Alterable information	→ Indicar en Integridad “low”.
Confidentiality	Low: Non critical information that could be displayed without inconvenience	→ Indicar en Confidencialidad “low”.
Test	Yes	→ Indicar Test “yes”. <input type="checkbox"/>
You will not be able to change this after creation		
Core-metric	not-apply	
Core-metric-site	not-apply	

Advanced configurations

Cluster

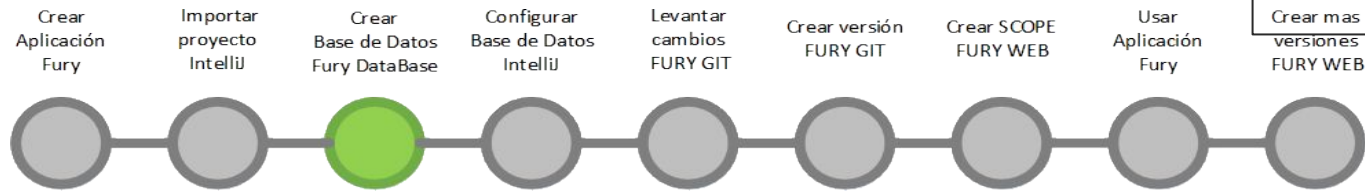
desaenv03

Perfect for: Project's internal, non critical information (e.g. Staging)

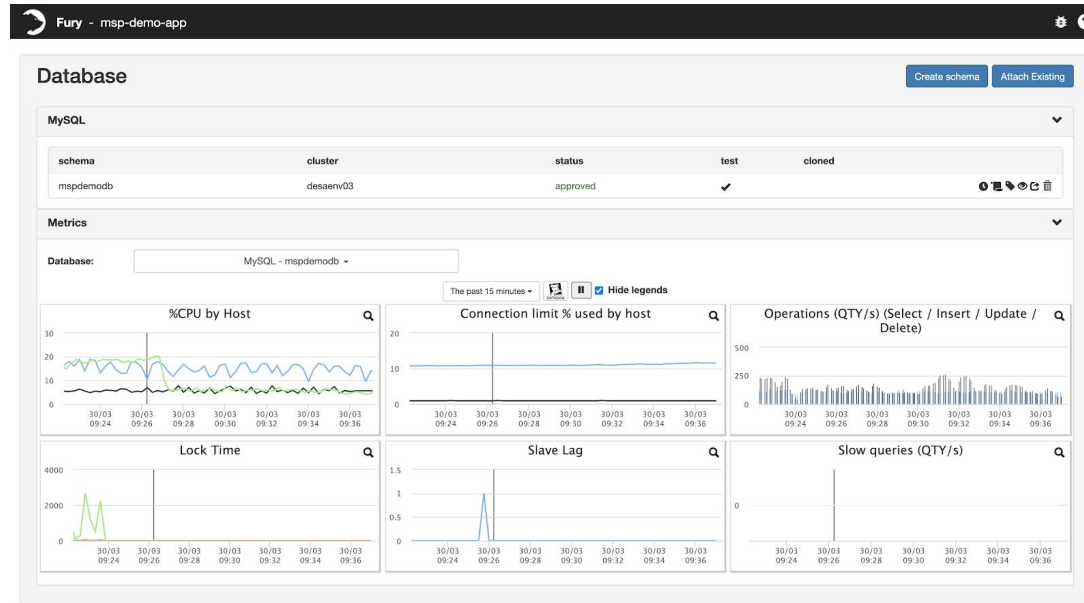
Cancel Create

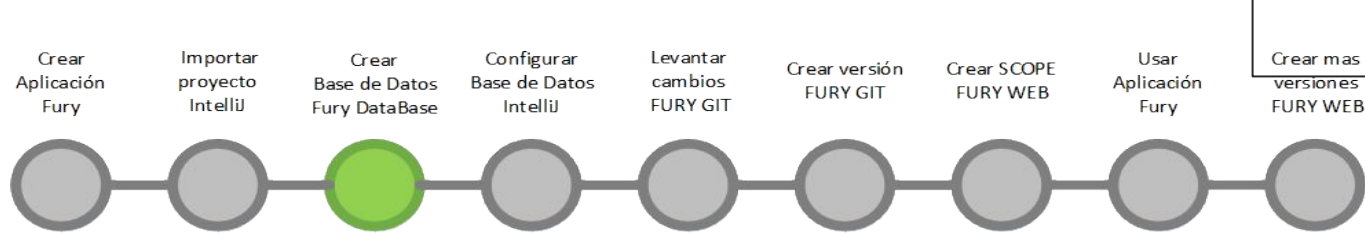
→ Verificar en Advanced configurations
desaem03 cluster compartido

←



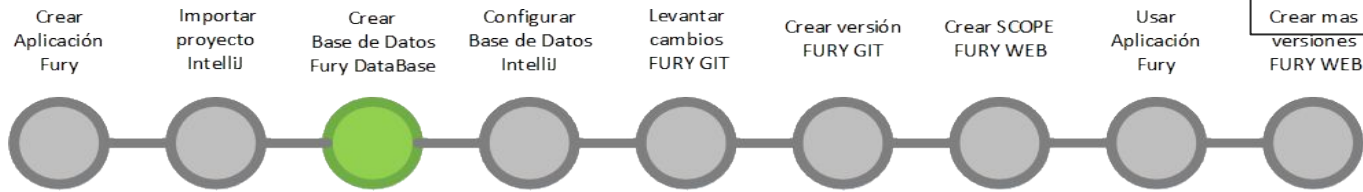
3 - Chequear que el proceso nos derive exitosamente en la pantalla de la BD.





4 - Verificar la recepción de un e-mail con usuario y contraseña.





OPCIONAL - Probar la conexión a la base de datos con herramientas (MySQL Workbench u otro manager)

DATOS PARA LA CONEXIÓN BASE DE DATOS

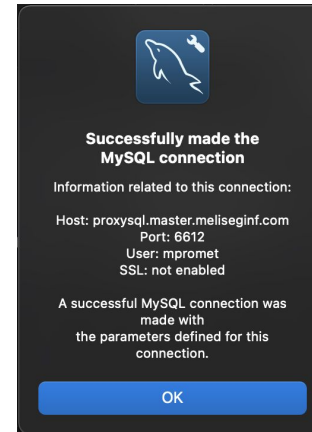
Hostname: proxysql.master.meliseginf.com

Port: 6612

Username: [usuario_meli]

Password: [la contraseña que llegó por e-mail]

Default Schema: [nombre de la base creada]



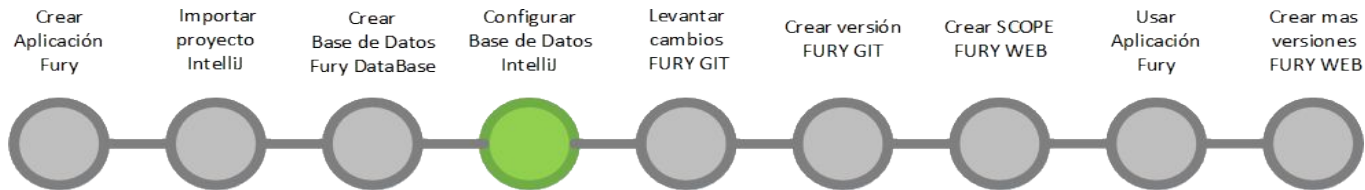
4

//Configurar Base de Datos en el proyecto

IT BOARDING

BOOTCAMP





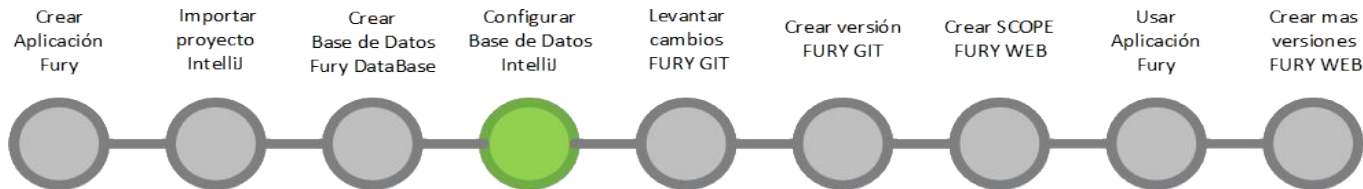
1 - Agregar/chequear dependencias de BD del proyecto en el pom.xml.

```
<!-- JPA -->
<dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-data-jpa</artifactId>
</dependency>

<!-- MySQL -->
<dependency>
  <groupId>mysql</groupId>
  <artifactId>mysql-connector-java</artifactId>
  <scope>runtime</scope>
</dependency>

<!-- Hibernate -->
<dependency>
  <groupId>com.h2database</groupId>
  <artifactId>h2</artifactId>
  <scope>runtime</scope>
</dependency>

<!-- Meli spring boot -->
<dependency>
  <groupId>com.mercadolibre</groupId>
  <artifactId>spring-boot-starter</artifactId>
  <version>0.0.4</version>
</dependency>
```



2 - Ingresar en el manager “dbaccess.meliseginf.com” para gestionar “Mis Schemas - MySQL”.
(tarda aprox. 15 min. en aparecer, luego de crear el schema en Fury)

3 - Configurar en el archivo “application.yml” las propiedades:

Se podrán configurar varios entornos dsitintos: application-local.yml / application-test.yml / application-prod.yml)

spring:

datasource:

url: jdbc:mysql://proxysql.master.meliseginf.com:6612/[nombre_schema]?useSSL=true&serverTimezone=UTC&characterEncoding=UTF-8

username: \${LOCAL_DB_USER}

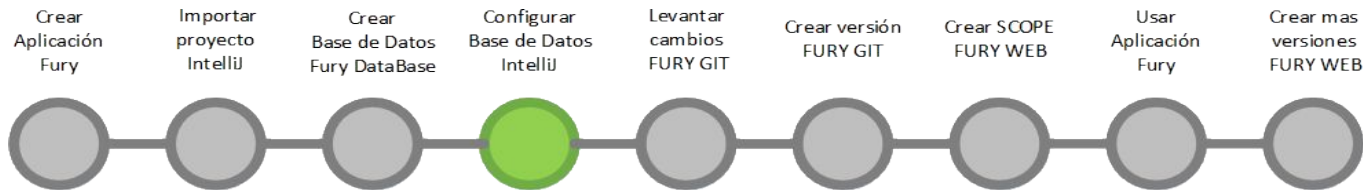
password: \${LOCAL_DB_PASS}

jpa:

show-sql: false

hibernate:

ddl-auto: none



4 - Editar variables de entorno y setear:

`LOCAL_DB_USER = [usuario]`

`LOCAL_DB_PASS = [clave]`

5 - En el dashboard de Fury, Bases de datos, solapa “Snippets”, copiar los snippets de conexión a la BD e incorporarlos al proyecto.

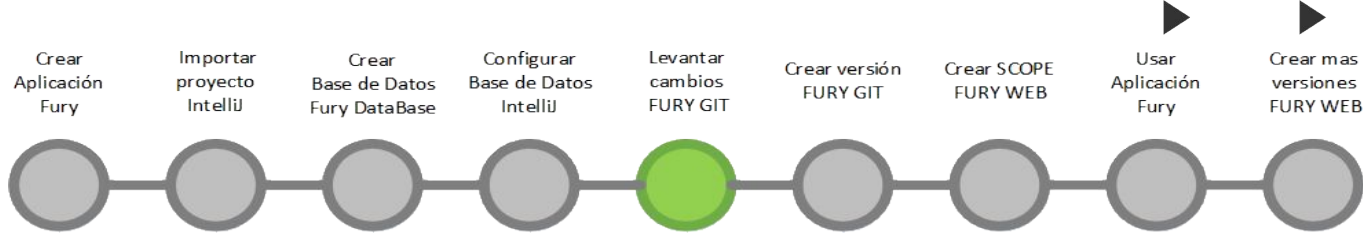
5

//Actualizar cambios en Fury-GitHub

IT BOARDING

BOOTCAMP





1 - Sobre la carpeta principal del proyecto Ejecutar:

`git status`

2 - Crear la rama principal:

`git branch -M main`

3 - Agregar los archivos al stage:

`git add .`

4 - Establecer el commit inicial:

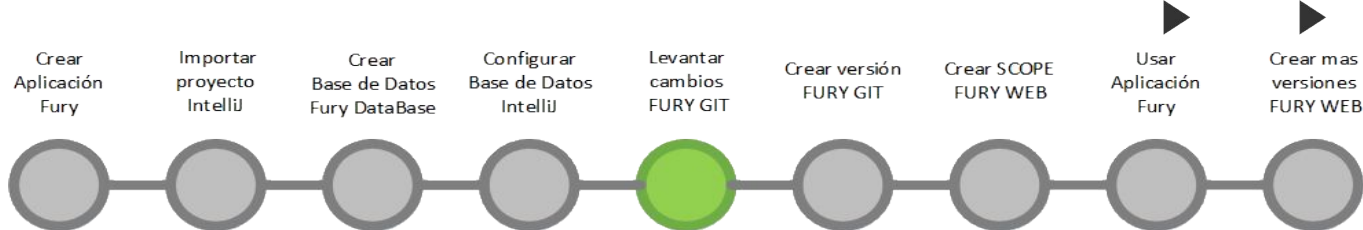
`git commit -am "Commit initial"`

5 - Subir el commit al repositorio remoto:

`git push -u origin main`

6 - Entrar a fury y verificar la aplicación

[github.com/mercadolibre/\[nombre_app\]](https://github.com/mercadolibre/[nombre_app])



TROUBLESHOOTING

ERROR → `java.lang.NoClassDefFoundError: com/mercadolibre/restclient/http/Headers`

SOLUCIÓN →

- Dentro del archivo "pom.xml" buscar el tag de la dependency **com.mercadolibre.restclient** (aprox linea 296) y borrar el tag scope tanto de esta dependencia asi como también los de las siguientes dos.
- Volver a hacer un build del proyecto y correr la aplicación.

ERROR → `Exception in thread "main" java.lang.IllegalArgumentException: [Assertion Fail] SCOPE must be set to startup the application.`

SOLUCIÓN →

- Al correr dentro de IntelliJ debemos setear la variable **SCOPE** con el nombre del scope que estamos usando. Al ser una prueba dentro de nuestra máquina podemos usar **local** y alcanza para que corra. Igualmente si tenemos distintas definiciones para cada scope y queremos probarlas, es cuestión de cambiar el valor de esta variable, por ejemplo **test**.

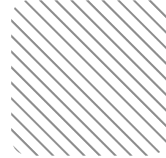
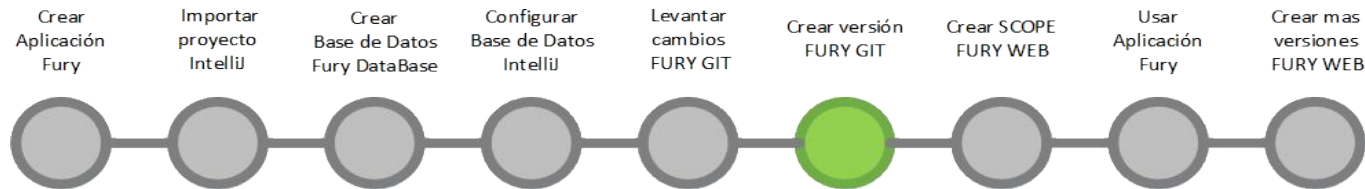
6

//Crear versión en Fury-Git

IT BOARDING

BOOTCAMP





1 - Sobre la carpeta del repositorio, ejecutar el comando

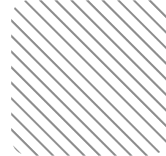
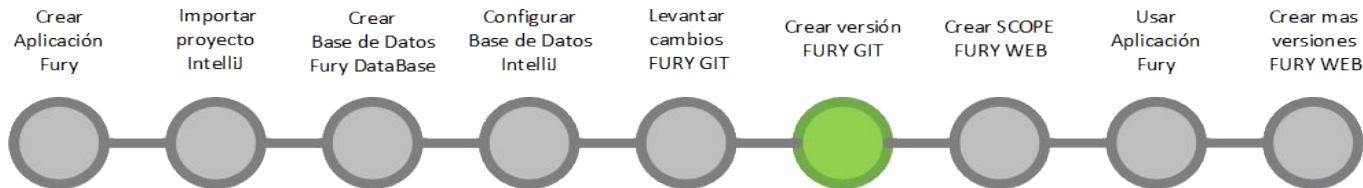
fury create-version 0.0.1 [--no-tests]

```
AR0C02D14UPML85:msp-demo-app mpromet$ fury create-version 0.0.1
Validating if the repository has changes pending to commit
Version 0.0.1 is being created with commit afcda92f35bc3a7abf8a9cdf46e202fc45c862e6 for application msp-demo-app on branch main
Fetching build information for version '0.0.1'...
Job status is pending. I'm going to wait until it starts...
You can abort the wait at any time with Ctrl-C
Build URL: http://rp-builds.furyccloud.io/blue/organizations/jenkins/msp-demo-app/detail/msp-demo-app/1/pipeline
```



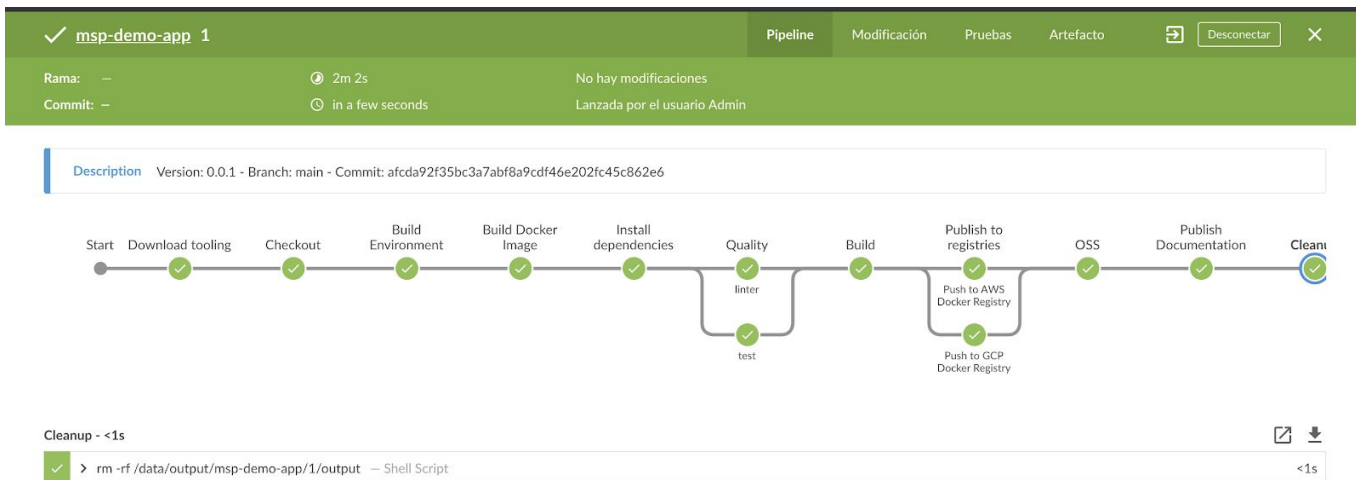
URL de Jenkins

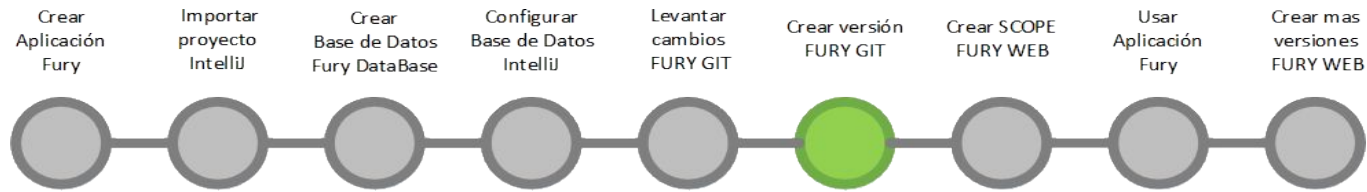




2 - Verificar el pipeline de creación de la versión en la URL de Jenkins.

- opción 1: copiar URL que devuelve la consola en la creación de la versión (slide anterior)
- opción 2: Desde el dashboard de fury ---> versions ---> actions





3 - Chequear en el dashboard de Fury la creación de la nueva versión.

Fury | msp-demo-app

Search

Name	Type	Branch	Status	Username	Created	Actions
0.0.1	RELEASE	main	FINISHED	mpromet	8 minutes ago	View logs ▾

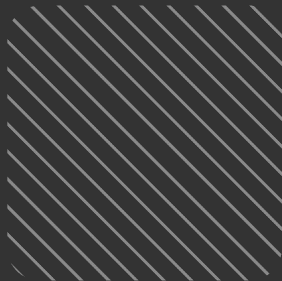


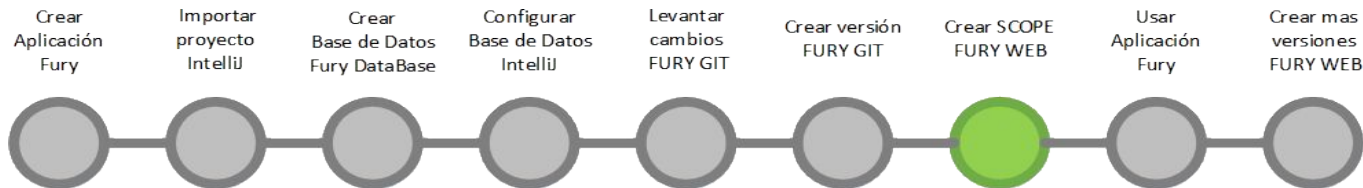
7

//Crear scope Fury Web y efectuar el Deploy

IT BOARDING

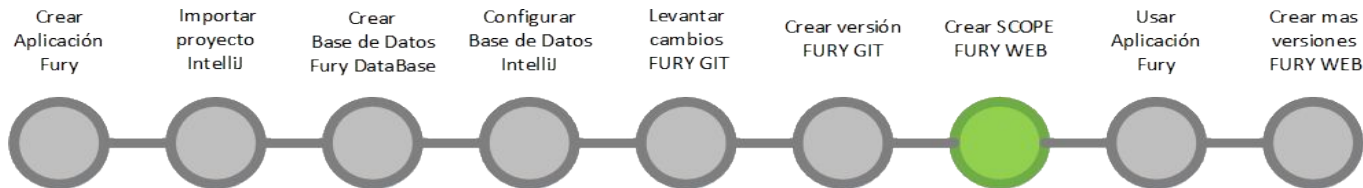
BOOTCAMP





1 - En el dashboard de Fury clicar en el menú “Web”.





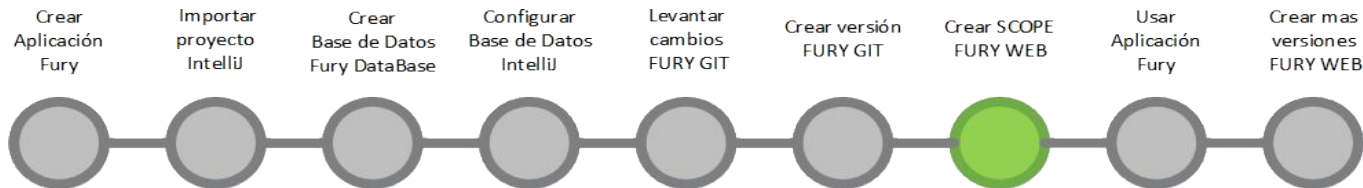
2 - Completar el formulario y clicar “Create”.

Fury - msp-demo-app

Create scope

Application	msp-demo-app		
Name	test		
Test	Yes	Indicar nombre del Scope	
You will not be able to change this after creation			
Version	0.0.1	Indicar que es un Scope de “Test”	
Core-metric	not-apply	Seleccionar una versión	
Core-metric-site	not-apply		
Traffic-type	backend	Indicar Tipo de tráfico “backend”	
User-affectation	not-apply		
Secrets	Select secret		
Advanced configurations >			

Cancel Create



3 - Verificar la creación correcta del Scope.

Fury - msp-demo-app

Scopes

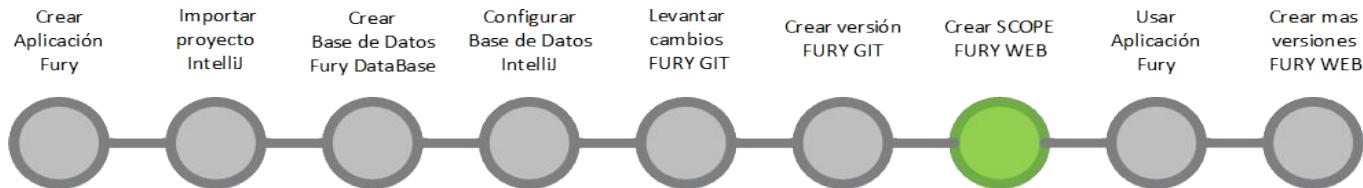
Search:

Productive

name	region	app version	branch	cname prod access	cname dev access	productive	test	mesh
test	us-east-4-gcp	0.0.1		test.msp-demo...	test_msp-demo-a...		✓	✓

Deploy

4 - Desplegar la App. Clickear el botón deploy.



5 - Comprobar el estado del despliegue.

Fury | msp-demo-app

Deployment 2678694

Created 50 seconds ago
Fury User

test 0.0.1
Version Tag

[BOOT APPLICATION](#) Initial [Rollback](#)

Steps

- ✓ Start
- ✓ Create Infrastructure
- > **Boot Application** IN PROGRESS

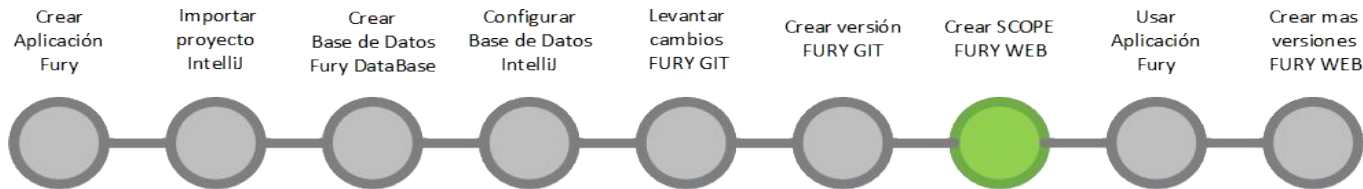
The version is being booted in the instances and waiting to response "/>

Instances Booted - 1/1 Containers Started - 0/1 Application Up - 0/1

Effective

[DATADOG](#) [New Relic](#) [Kibana](#)

Feedback



6 - Verificar el resultado exitoso del despliegue.

Deployment 2678694[®]

Created 2 minutes ago
Fury User

test 0.0.1
Version Tag

FINISHED
Initial

Steps

- ✓ Start
- ✓ Create Infrastructure
- ✓ Boot Application
- ✓ Effective

Finished successfully.

Feedback

DATADOG New Relic Kibana

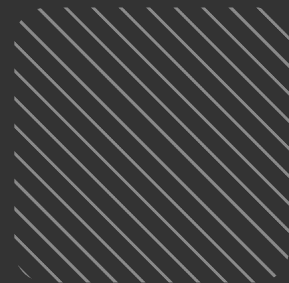


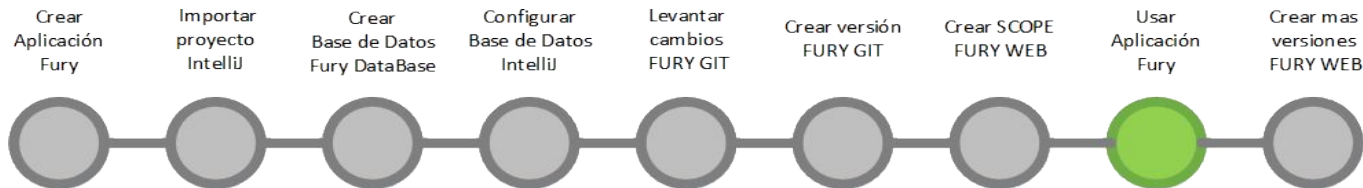
8

//Trabajar con la App

IT BOARDING

BOOTCAMP





Correr la App en forma local

Sobre la carpeta base del proyecto ejecutar el comando:

fury run

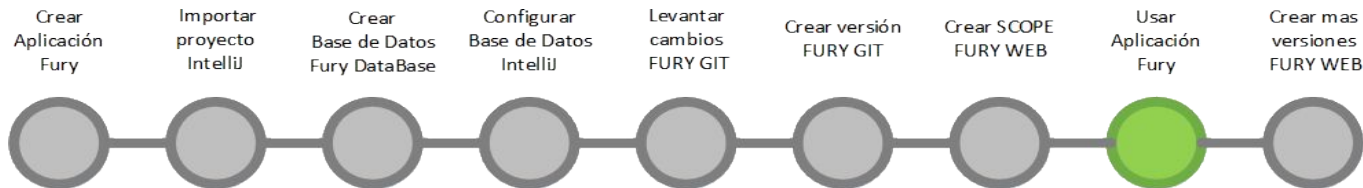
Eso levanta la app para probar en el scope de Testing. Configurar Scopes en Docker.
Si la App levanta (compila) en forma local, debería hacerlo luego al subirla.

✖ ✖

✖ ✖

✖ ✖



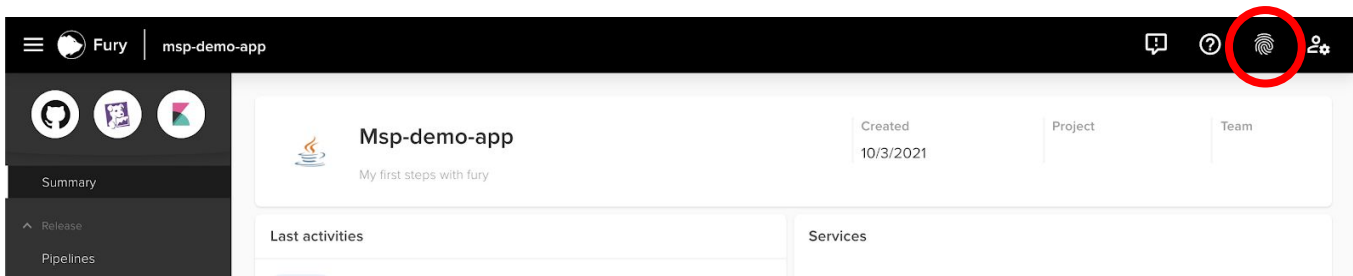


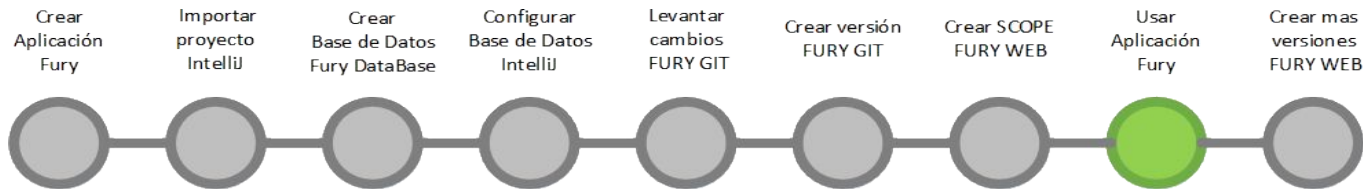
Obtener token de acceso

1a - Por consola se puede ejecutar el comando para obtener el Token: **fury get-token**

```
AR0C02DT4UPML85:mSP-demo-app mpromet$ fury get-token  
d8051f0f0e50a40a677e33977f5439336fddada6cd33214e5f1ceae338f84c35c
```

1b - También se puede obtener el token desde el dashboard de Fury, clickeando en el ícono de la huella que se encuentra en el extremo superior derecho.





Utilizar token de acceso

Al llamar a nuestra aplicación desde el browser o cualquier otra herramienta (PostMan) necesitamos autenticar el request enviando en el encabezado (header) la llave (key) “X-Auth-Token” con el token como valor (value).

The image shows two screenshots from the Postman application. The top screenshot displays the 'test' environment variables table:

	VARIABLE	INITIAL VALUE	CURRENT VALUE		Persist All	Reset All
<input checked="" type="checkbox"/>	baseUrl	https://test_msp-demo-app.furyap...	https://test_msp-demo-app.furyapps.io/			
<input checked="" type="checkbox"/>	fury-token	cc1ce7e3df5fe11d76727180163173...	cc1ce7e3df5	:3262ffb5		
	Add a new variable					

The bottom screenshot shows a REST client request in Postman. The URL is `msp-demo-app / ping` with the method `GET`. The request body is `{{baseUrl}}/ping`. The 'Headers' tab is selected, showing a table with one header:

KEY	VALUE	DESCRIPTION		Bulk Edit	Presets
<input checked="" type="checkbox"/> x-auth-token	{{fury-token}}				
Key	Value	Description			



Gracias.

IT BOARDING

BOOTCAMP

