MACS

-

Manuale Tecnico

**QUESTA PAGINA È LASCIATA**

**INTENZIONALMENTE BIANCA**

|  |
| --- |
| STATO DEL DOCUMENTO |

|  |  |  |  |
| --- | --- | --- | --- |
| TITOLO DEL DOCUMENTO | | | |
| MACS Manuale Tecnico | | | |
| CLASSIFICAZIONE DEL DOCUMENTO | | | |
| USO INTERNO | | | |
| EDIZ. | REV. | DATA | AGGIORNAMENTO |
| 1 | 0 | 13/03/2023 | Prima stesura |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| STATO DI AGGIORNAMENTO | | | |
| PAR | EDIZ. | REVISIONE | MOTIVO DELL'AGGIORNAMENTO |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

|  |  |
| --- | --- |
| NUMERO TOTALE PAGINE: | 12 |

|  |
| --- |
| LISTA DI DISTRIBUZIONE |

INTERNA A:

ESTERNA A:

**Sommario**

[1. Scopo del documento 6](#_Toc129677781)

[2. Architettura 7](#_Toc129677782)

[3. Moduli 8](#_Toc129677783)

[3.1 MACS 8](#_Toc129677784)

[3.2 Notificatore 8](#_Toc129677785)

[3.3 Profilatore 8](#_Toc129677786)

[3.4 Portale 8](#_Toc129677787)

[3.5 Maggioli Prenotazioni 9](#_Toc129677788)

[4. Installazione 9](#_Toc129677789)

[4.1 Servizi MACS 9](#_Toc129677790)

[4.2 Databases 10](#_Toc129677791)

[5. Configurazione proprietà 10](#_Toc129677792)

[6. Comandi utili 12](#_Toc129677793)

[6.1 Riavvio containers 12](#_Toc129677794)

[6.2 Logs 12](#_Toc129677795)

[6.3 Rimozione Logs 12](#_Toc129677796)

# Scopo del documento

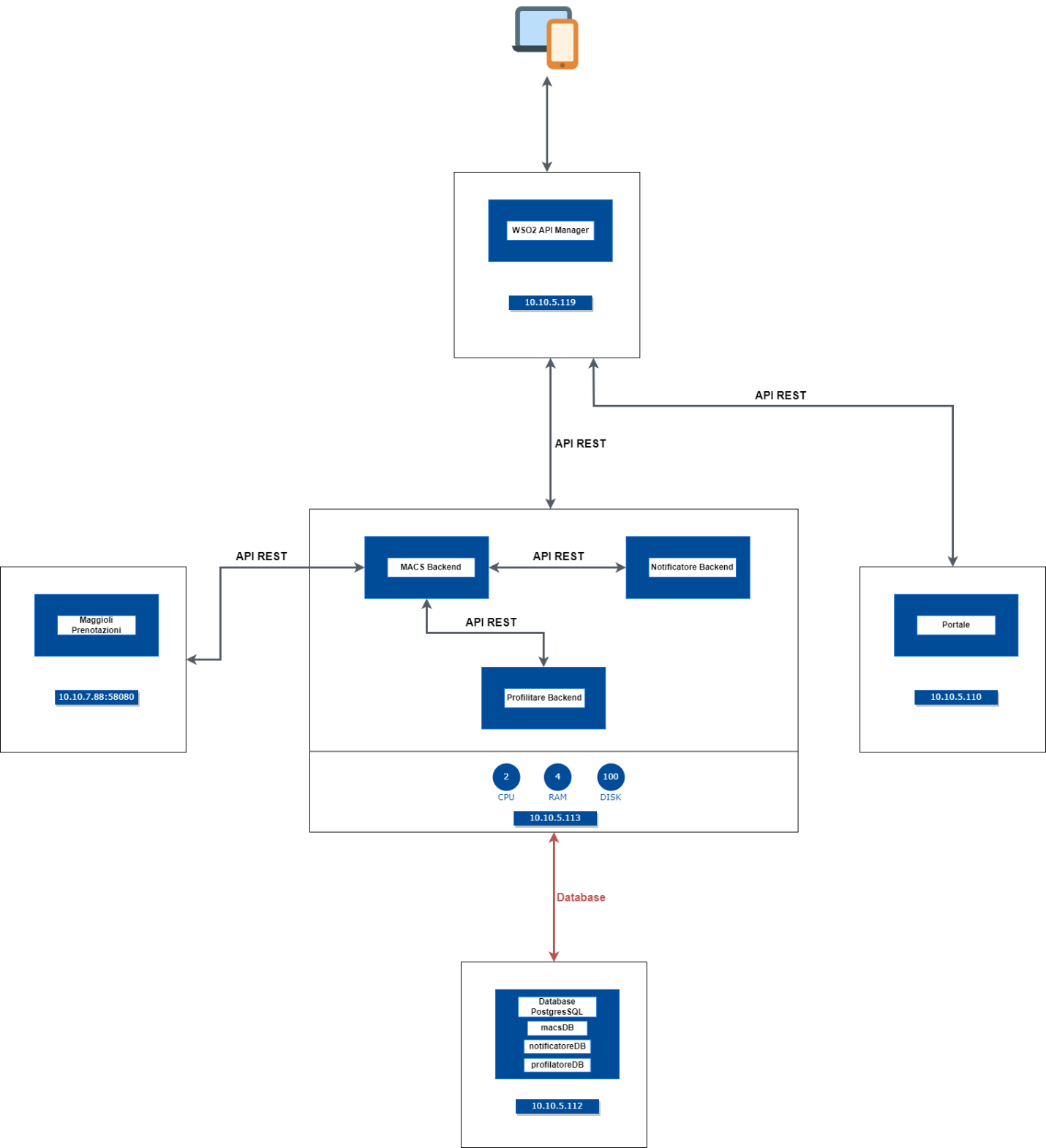
Il documento ha lo scopo di illustrare il manuale tecnico dell’applicazione mobile commissionata dal Comune di Salerno.

Le componenti che vedremo nel paragrafo “Architettura” sono installate attraverso la tecnologia della virtualizzazione, in particolare con l’utilizzo di Docker.

**Docker** è una piattaforma software che permette di creare, testare e distribuire applicazioni con la massima rapidità. Docker raccoglie il software in unità standardizzate chiamate container che offrono tutto il necessario per la loro corretta esecuzione, incluse librerie, strumenti di sistema, codice e runtime. Con Docker, è possibile distribuire e ricalibrare le risorse per un'applicazione in qualsiasi ambiente, tenendo sempre sotto controllo il codice eseguito.

# Architettura

Nello schema sottostante è rappresentata in grandi linee la comunicazione tra i vari componenti.



|  |
| --- |
| [*macs\_architettura\_collaudo.png*](macs_architettura_collaudo.png) |

# Moduli

L’applicazione Mobile è rivolta ai cittadini della città di Salerno.

L’applicazione MACS per il corretto funzionamento utilizza una serie di servizi offerti da altri moduli: **MACS Backend**, **Notificatore**, **Profilatore**, **Portale**, **Maggioli Prenotazioni**

## 3.1 MACS

Il modulo MACS gestisce la logica applicativa della mobile App e le varie integrazioni verso altri moduli.

MACS si integra con:

* **Notificatore**: gestisce le notifiche PUSH
* **Profilatore**: gestisce la profilazione delle utenze SPID in termini di funzioni e funzionalità per il ruolo “**CITTADINO**”
* **Maggioli Prenotazione**: gestisce le risorse comunali e le prenotazioni di quest’ultime.

**Servizi**

I servizi esposti dalla mobile app sono i seguenti:

|  |
| --- |
|  |

## 3.2 Notificatore

Il modulo notificatore gestisce le notifiche PUSH in termini di configurazione e di invio.

**Servizi**

|  |
| --- |
|  |

## 3.3 Profilatore

Il modulo profilatore gestisce i ruoli, funzioni e funzionalità associate alle utenze SPID.

**Servizi**

|  |
| --- |
|  |

## 3.4 Portale

Il modulo espone i servizi per le news così da mostrarle anche sulla Mobile App.

**Servizi**

|  |
| --- |
|  |

## 3.5 Maggioli Prenotazioni

Maggioli è un’azienda che offre servizi per la pubblica amministrazione.

Nel nostro caso ci vengono forniti i servizi per le prenotazioni delle risorse comunali.

**Servizi**

|  |
| --- |
|  |

# Installazione

## 4.1 Servizi MACS

Entrare nella cartella “**docker-csgateway**” ed eseguire i seguenti comandi:

* docker-compose build
* docker-compose up -d

Dopo l’esecuzione dei comandi accedere al Tomcat ed effettuare l’upload dei files presenti nella cartella “*wars*”

**Tomcat**

Indirizzo: **http://10.10.5.113:8080/**

Username: **admin**

Password: **csgateway**

Nella cartella “*shared*” ci sono i file di configurazione, nel dettaglio:

* *properties MACS*
  + docker-csgateway/shared/configuration/configurationFile/macs/ application.properties
* *properties Profilatore*:
  + docker-csgateway/shared/configuration/modules/profilatore/ application.properties
* *properties Notificatore*:
  + docker-csgateway/shared/configuration/modules/notificatore/ application.properties

All’interno della medesima cartella è presente la cartella “*root\_macs*”, dove è presente il file per configurare i filtri per i POI (Point of interest) della mappa sull’applicazione Mobile.

Nel dettaglio:

* *poi-configuration*
  + docker-csgateway/shared/root\_macs/Macs/categorie-poi.json

## 4.2 Databases

**MACS Database**

Accedere al server “*10.10.5.112*”, copiare nella cartella principale il file “*db\_macs.sql*” presente nella cartella “*database\_macs*”.

Dopo di che eseguire seguire i seguenti step:

1. Eseguire il comando
   1. psql -U postgres
2. inserire la password di postres
3. Eseguire i comandi
   1. create database macsdb;
   2. create user iside with encrypted password 'password\_macs';
   3. grant all privileges on database macsdb to macsdb;
4. Eseguire il comando
   1. psql -U macsdb < db\_macs.sql
   2. password\_macs

**Profilatore Database**

Accedere al server “*10.10.5.112*”, copiare nella cartella principale il file “*db\_profilatore.sql*” presente nella cartella “*database\_profilatore*”.

Dopo di che eseguire seguire i seguenti step:

1. Eseguire il comando
   1. psql -U postgres
2. inserire la password di postres
3. Eseguire i comandi
   1. create database profilatore;
   2. create user iside with encrypted password 'password\_profilatore';
   3. grant all privileges on database profilatore to profilatore;
4. Eseguire il comando
   1. psql -U profilatore < db\_profilatore.sql
   2. password\_ profilatore

**Notificatore Database**

Accedere al server “*10.10.5.112*”, copiare nella cartella principale il file “*db\_notificatore.sql*” presente nella cartella “*database\_notificatore*”.

Dopo di che eseguire seguire i seguenti step:

1. Eseguire il comando
   1. psql -U postgres
2. inserire la password di postres
3. Eseguire i comandi
   1. create database notificatore;
   2. create user iside with encrypted password 'password\_notificatore';
   3. grant all privileges on database notificatore to notificatore;
4. Eseguire il comando
   1. psql -U profilatore < db\_notificatore.sql
   2. password\_notificatore

# Configurazione proprietà

**MACS**

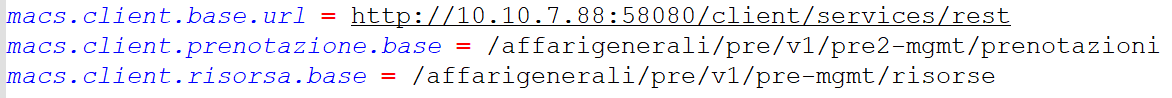
Accedere al file “*docker-csgateway/shared/configuration/configurationFile/macs/ application.properties*”

**Configurazione del database**

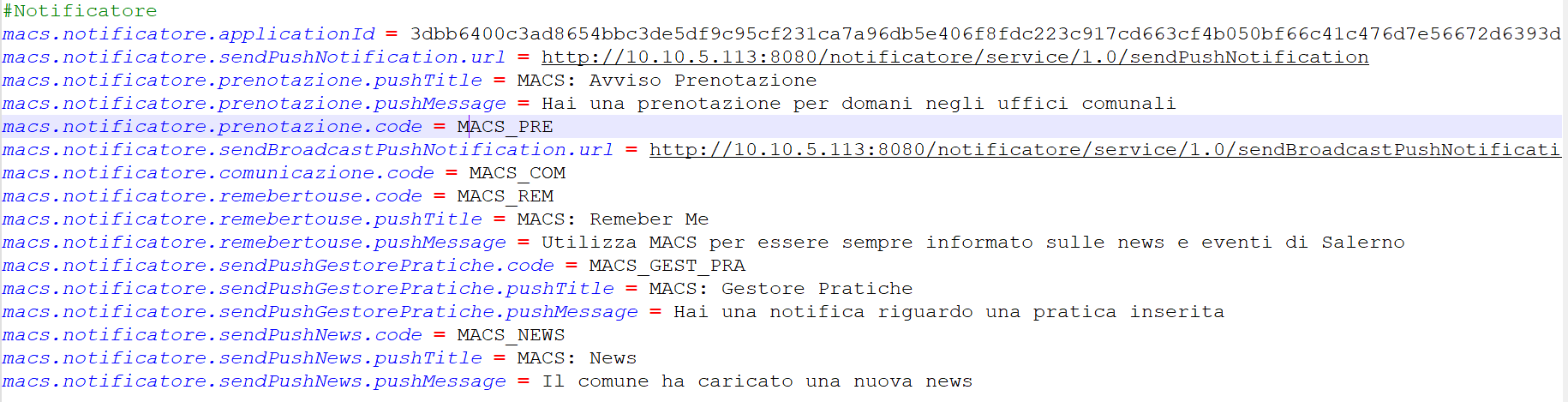
Immagine che contiene testo

Descrizione generata automaticamente

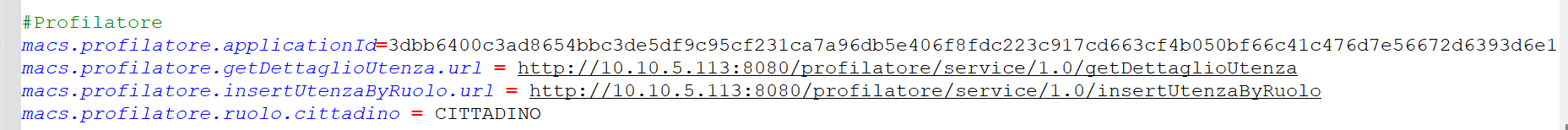
**Configurazione dell’integrazione con i servizi delle prenotazioni**



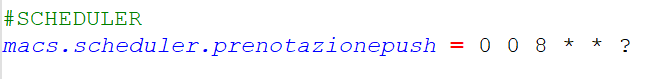
**Configurazione dell’integrazione il modulo “Notificatore”**



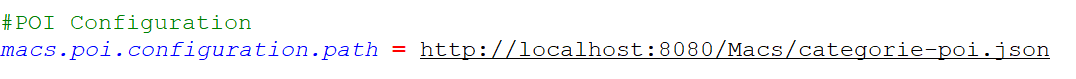
**Configurazione dell’integrazione il modulo “Profilatore”**



**Configurazione dello schedule per l’invio della PUSH per le prenotazioni**



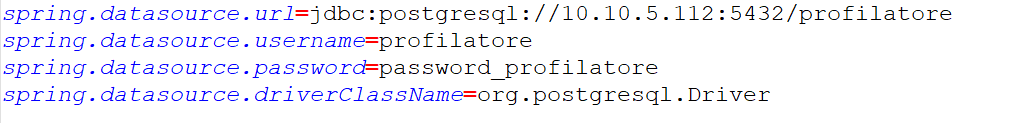
**Configurazione della path delle categorie dei poi**



**Profilatore**

Accedere al file “*docker-csgateway/shared/configuration/modules/profilatore/ application.properties*”

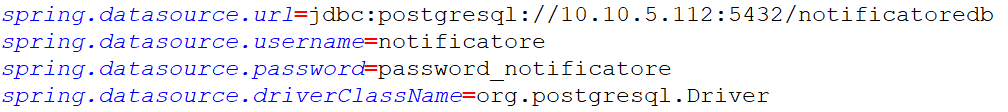
**Configurazione del database**

****

**Notificatore**

Accedere al file “*docker-csgateway/shared/configuration/modules/notificatore/ application.properties*”

**Configurazione del database**

****

# Comandi utili

## 6.1 Riavvio containers

Il seguente i comandi sono utili per riavviare il container:

* docker restart csgateway-tomcat

## 6.2 Logs

Il seguente comando è utile per visionare il log in real time del container:

* docker logs csgateway --tail 500 -f

## 6.3 Rimozione Logs

I seguenti i comandi sono utili per rimuovere i file di log di un qualsiasi container installato sulla macchina:

* Eseguire i seguenti comandi
  + sudo su
  + docker ps
    - recuperare l’id del container di interesse (prima colonna)
  + cd /var/lib/docker/containers
  + ls
    - trovare il container le cui 12 cifre sono uguale all’id del container di interesse
  + cd xxxxxxxxxxxxyyyyyyyyy…y
  + rm xxxxxxxxxxxxyyyyyyyyy…y -json.log
  + docker restart **idcontainer**