Manuel d’installation technique Plateforme d’accompagnement

02/12/2024 – VERSION 2

Sommaire

[Lexique 3](#_Toc107504243)

[Avant de commencer 4](#_Toc107504244)

[Configuration minimale requise 4](#_Toc107504245)

[Prérequis 4](#_Toc107504246)

[1. Récupération des sources 5](#_Toc107504247)

[1.1. Github 5](#_Toc107504248)

[1.2. Docker 5](#_Toc107504249)

[2. Configuration des services 6](#_Toc107504250)

[2.1. Lancement de l’installation 6](#_Toc107504251)

[2.2. Les différents services 6](#_Toc107504252)

[2.2.1. BDD PostgreSQL pour la factory 6](#_Toc107504253)

[2.2.2. BDD PostgreSQL pour KEYCLOAK 6](#_Toc107504254)

[2.2.3. BDD Influxdb 6](#_Toc107504255)

[2.2.4. Analytica 6](#_Toc107504256)

[2.2.5. Keycloak 7](#_Toc107504257)

[2.2.6. Le designer 7](#_Toc107504258)

[2.2.7. Le(s) runner(s) 7](#_Toc107504259)

[2.2.8. Rasa 8](#_Toc107504260)

[2.3. Configuration de Keycloak 8](#_Toc107504261)

[3. Annexes 10](#_Toc107504262)

# Lexique

Ci-dessous une liste de termes spécifiques au projet qui peuvent être utilisés dans la suite du manuel d’installation.

* **Factory (ou Designer)** : il s’agit de l’interface d’administration, permettant la création et l’administrations des différentes plateformes déployées.
* **Runner (ou Executor)** : il s’agit d’une instance de plateforme avec une configuration et une interface qui lui sont propre.

# Avant de commencer

## Configuration minimale requise

L’installation de la plateforme d’accompagnement nécessite une configuration suivante au minimum :

* **16 Go de RAM** (8 Runner, 4 Factory, 4 BDD)
* **12 vCPU** (4 Runner, 4 Factory, 4 BDD)
* **80 Go d’espace disque** (20 Runner, 20 Factory, 40 BDD)

## Prérequis

Pour procéder à l’installation, les éléments suivants doivent être disponibles sur la machine :

* **Système d’exploitation** : Distribution Linux (Debian 10 x64 par exemple)
* **Processeur** prenant en charge les instructions AVX, nécessaires au fonctionnement du moteur de NLP (Rasa).
* **Docker** installé et prêt à l’emploi
* **Un accès internet** sur la machine pour le pull des containers docker depuis le repository public.

## Variantes de configuration

En plus du socle de base de la plateforme d’accompagnement, il est possible de réaliser l’installation associée de connecteurs complémentaires :

* Connecteur JIRA : permet de pouvoir créer des tickets JIRA à partir de demandes faites via la plateforme.
* Connecteur Confluence : permet de chercher et d’afficher des pages Confluence à partir de demandes faites via la plateforme.

Ces connecteurs ne sont pas obligatoires à installer, et ne font actuellement pas l’objet de facturation spécifique par rapport au socle de base de la plateforme.

# Récupération des sources

## Github

Lien du projet Github : <https://github.com/vertigo-io/vertigo-bot-factory>

Pour compiler et packager l’application en local, il est nécessaire d’avoir cloné et packagé le repository suivant au préalable : <https://github.com/vertigo-io/vertigo-ai>

## Docker

Pour chaque nouvelle version, plusieurs nouvelles images sont publiées sur le repository docker de Klee.

Pour la factory : <https://registry.kleegroup.com/harbor/projects/23/repositories/bot-factory-designer>

Exemple de commande de pull :

**docker pull registry.kleegroup.com/chatbot-factory/bot-factory-designer:2.3.5**

Pour le runner, l’URL diffère en fonction des extensions choisies :

|  |  |
| --- | --- |
| Configuration | URL |
| Runner seul | <https://registry.kleegroup.com/harbor/projects/23/repositories/bot-factory-runner> |
| Runner + JIRA | <https://registry.kleegroup.com/harbor/projects/23/repositories/bot-factory-runner-jira> |
| Runner + CONFLUENCE | <https://registry.kleegroup.com/harbor/projects/23/repositories/bot-factory-runner-confluence> |
| Runner + JIRA + CONFLUENCE | <https://registry.kleegroup.com/harbor/projects/23/repositories/bot-runner-all-included> |

Il est recommandé de ne jamais utiliser les images avec le tag « latest » qui sont des versions en cours de développement. Les versions stables sont identifiées pas des tags du type « 2.3.5 » etc…

# Configuration des services

## Lancement de l’installation

Pour plus de facilité, il est recommandé de lancer l’installation avec docker compose. Un exemple de fichier docker-compose.yml est fourni avec cette documentation. Le champ « $version » est à remplacer par le numéro de version de la factory et des différents runners à installer.

## Les différents services

### BDD PostgreSQL pour la factory

Il s’agit d’une base de données PostgreSQL pour la factory.

Le script d’initialisation de la BDD est fourni et doit être monté dans un volume.

* POSTGRES\_USER : chatbot, ne pas modifier
* POSTGRES\_PASSWORD : chatbot, peut être modifié
* POSTGRES\_DB : chatbot, ne pas modifier

### BDD PostgreSQL pour KEYCLOAK

Il s’agit d’une base de données PostgreSQL pour KeyCloak, le composant mis en œuvre pour servir de gestionnaire d’authentification à la plateforme.

Les paramètres suivants peuvent être mis à jour :

* POSTGRES\_USER
* POSTGRES\_PASSWORD
* POSTGRES\_DB

### BDD InfluxDB

Il s’agit d’une base de données InfluxDB pour la collecte des différentes statistiques.

### Analytica

Ce service permet d’envoyer les données de statistique depuis le runner vers la base InfluxDB.

* INFLUXDB\_URL : URL de base InfluxDB (ex : <http://influxdb:8086>)

### KeyCloak

Il s’agit d’un service qui permet la gestion des utilisateurs ayant accès à la factory. Les 3 paramètres suivant peuvent être mis à jour et doivent correspondre aux paramètres définis dans la base de données PostgreSQL pour KeyCloak (cf. section 2.2)

* KEYCLOAK\_DATABASE\_NAME
* KEYCLOAK\_DATABASE\_USER
* KEYCLOAK\_DATABASE\_PASSWORD

Les paramètres suivants sont à ajuster avec le chemin sur lequel le KeyCloak est exposé :

* KC\_HOSTNAME\_PATH: Chemin relatif (ex : /sso)
* KC\_HOSTNAME\_URL: URL complete (ex: https://localhost:8200/sso)
* KC\_HOSTNAME\_ADMIN\_URL: URL complete (ex: https://localhost:8200/sso)

### Le designer

Un service “*designer*” pour la factory

* DB\_URL : URL de la BDD PostgreSQL (ex : //postgres:5432/chatbot)
* ANALYTICA\_HOST : Host pour le service Analytica
* ANALYTICA\_PORT : Port du service Analytica
* ANALYTICA\_DBNAME : Nom de la base de données
* INFLUXDB\_URL : URL de la base InfluxDB (ex : <http://influxdb:8086>)
* DEPLOY\_PATH : *application context*, chemin de déploiement après le host (ex : designer 🡪 <http://designer:8080/designer>)
* CLAMAV\_HOST : Host pour le service d’antivirus (ex : ClamAV)
* ENCRYPTION\_PASSWORD : Chaine de caractère aléatoire pour le chiffrement des mots de passe
* EXTENSIONS\_WHITELIST : Liste d’extensions autorisées pour l’upload de fichiers (ex : png,jpg,jpeg,pdf,csv,js)
* MAIL\_NAME : Nom du serveur mail (ex : KleeMail)
* MAIL\_PROTOCOL : Protocole du serveur mail (ex : smtp)
* MAIL\_HOST : Host du serveur mail (ex : localdelivery.klee.lan.net)
* MAIL\_PORT : Port du serveur mail (ex : 25)
* KEYCLOAK\_URL : URL vers le realm KeyCloak à utiliser (ex: <https://localhost:8200/sso/realms/chatbot>)

### Le(s) runner(s)

Un ou plusieurs services « *runner* » :

* DESIGNER\_URL : URL du service « designer » (ex : http://designer:8080/designer/)
* ANALYTICA\_HOST : Host pour le service Analytica
* ANALYTICA\_PORT : Port du service Analytica
* ANALYTICA\_DBNAME : Nom de la base de données
* API\_KEY : Une chaine de caractère arbitraire qui permettra d’ajouter le runner au bot dans la factory
* RASA\_URL : URL du service RASA (<http://rasa:5005>)
* MAIL\_NAME : Nom du serveur mail
* MAIL\_PROTOCOL : Protocol du serveur de mail
* MAIL\_HOST : Host du serveur de mail
* MAIL\_PORT : Port du serveur de mail
* FRAME\_ANCESTORS : Liste des sites autorisés à intégrer la plateforme en mode Iframe (\* pour tout autoriser)
* ENCRYPTION\_PASSWORD : Chaine de caractère aléatoire pour le chiffrement des mots de passe (attention, la chaine de caractère doit être identique à celle définie dans la factory)
* CLAMAV\_HOST : Host pour le service d’antivirus (ex : ClamAV)

### Rasa

Un service « *Rasa* » par runner. Rasa est un moteur de NLP (*Natural Language Processing*) qui permet d’extraire des intentions hors de phrase utilisateurs. Des échanges sont faits entre le runner et Rasa pour l’entrainement et la détections des intentions.

## Configuration de KeyCloak

Une fois les services démarrés, une phrase de configuration de KeyCloak est nécessaire avant de pouvoir accéder à la factory.

Pour cela, les actions suivantes sont à réaliser (l’interface de KeyCloak peut varier en fonction des versions) :

1. Ouvrir la console d’administration KeyCloak et se logger avec les identifiants suivants (identifiants par défaut)
   * Login : admin
   * Mot de passe : changeme
2. Créer un « *realm* » nommé « *chatbot* »
3. Dans ce realm, créer un client « *chatbot* » avec comme Root URL le chemin de déploiement du designer (ex : <http://localhost:8080/designer>), cf image 1 ci-dessous. (l’Admin URL se crée automatiquement)
4. Créer également un user « *admin* ». Aller ensuite dans *Credentials* et créer un mot de passe par défaut qui pourra être changé lors de la première connexion, cf. image 2 ci-dessous.
5. Il est maintenant possible de se logger à la factory avec l’utilisateur « *admin* » et le mot de passe défini dans KeyCloak. Cet utilisateur est un « *super* *admin* » qui possède les privilèges les plus élevés dans la factory.

Il existe deux autres types de profils, les « visiteur » et les « contributeur » qui on des droits adaptés à leur statut dans la factory

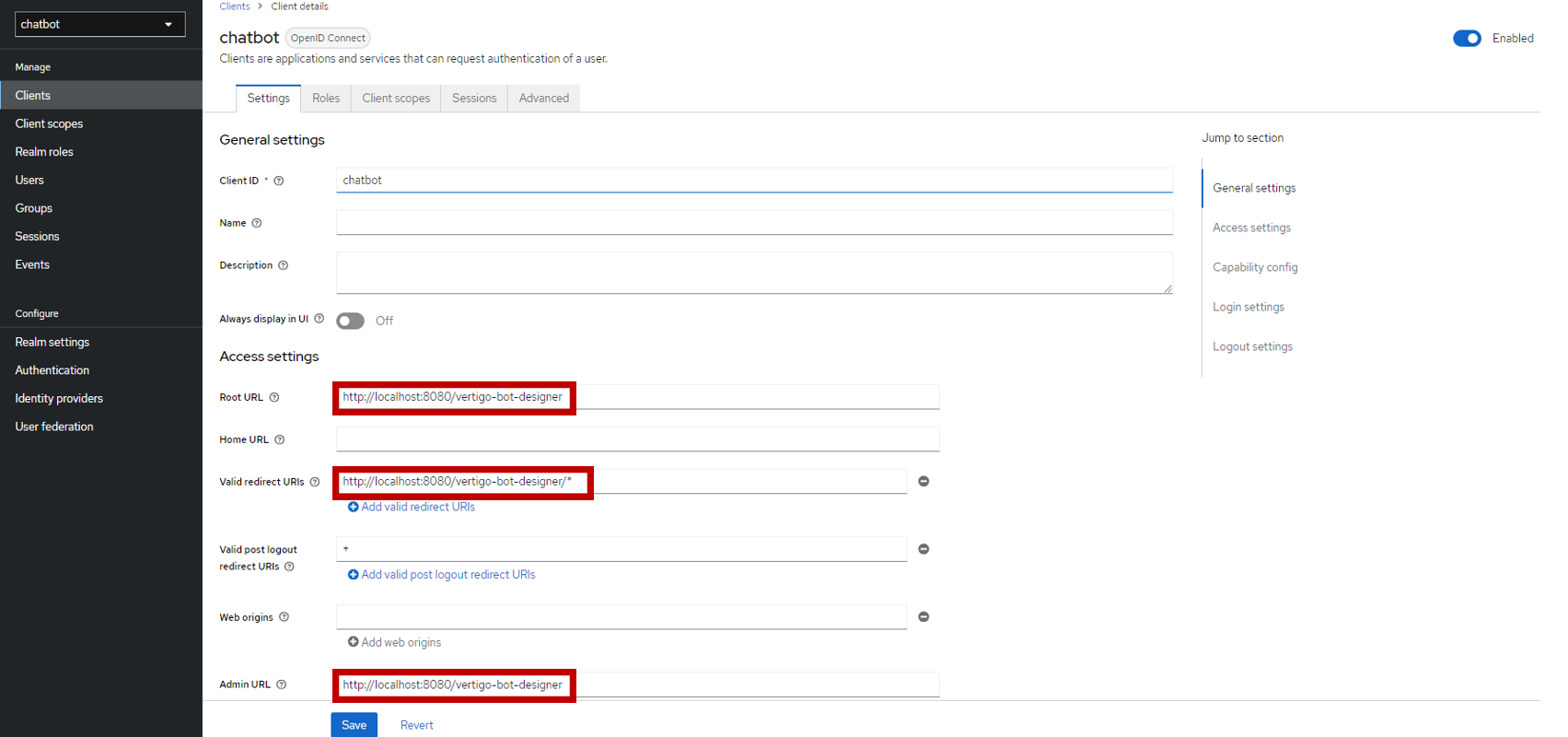
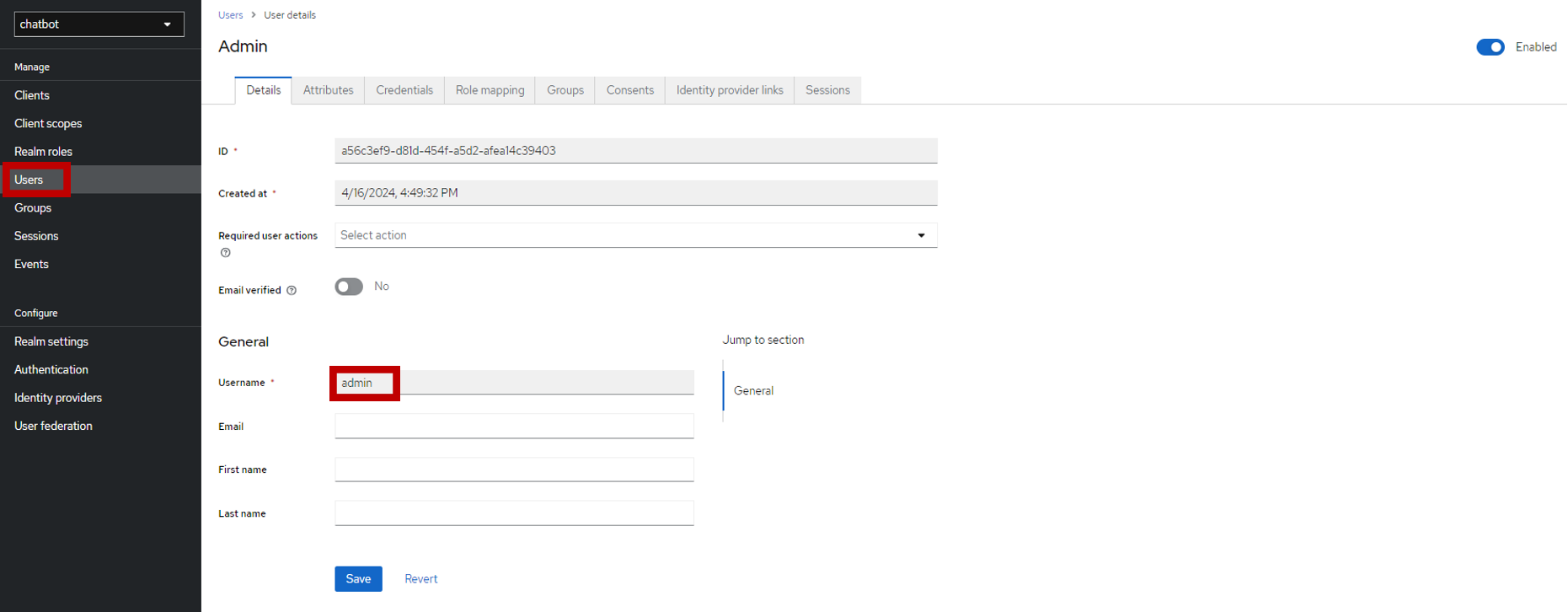


Image 1 : création du client « *chatbot* »



*Image 2 : Création de l’utilisateur « admin »*

# Annexes

En annexe de ce manuel se trouve une archive contenant les différents fichiers de configuration :

* A la racine, un fichier exemple « docker-compose.yml ». Ainsi qu’un script « start.sh » qui permet de lancer le docker compose avec en paramètre le numéro de version de la factory et des runner (ex : ./start.sh -- version 2.3.5)
* Le dossier « *initSql* » contient le script d’initialisation de la BDD PostgreSQL pour la factory.
* Le dossier « *integration* » contient les fichiers JS et CSS permettant d’intégrer la plateforme sur un site client. Le fichier chatbot.js doit être mis à jour avec l’url du runner, le nom du bot ainsi que le chemin vers l’avatar (ces paramètres se situent en fin de fichier), cf. image 3 ci-dessous.

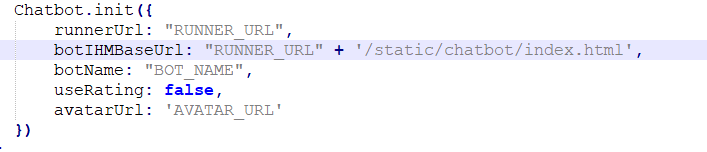


Image 3 : Exemple de complétion de la fonction d’initialisation du chatbot dans le fichier chatbot.js du dossier « *integration* »