*1. Prérequis*

Avant de commencer, il est important de créer le répertoire de travail et de télécharger l’ensemble des outils qui seront installés.

• Consul ⇒ Télécharger consul à l’adresse suivante <https://www.consul.io/downloads.html>

• Hazelcast-3.5.5 ⇒ <http://download.hazelcast.com/download.jsp?version=hazelcast-3.5.5&p=>

• Java jdk18 ⇒ <https://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html>

(Java SE Development Kit 8u191 ⇒ Sélectionnez votre architecture Linux)

• Certificat SAP ⇒ cert.zip

• Maven ⇒ <https://maven.apache.org/install.html>

• NodeJs + npm ⇒ <https://nodejs.org/en/download/>

• IntelliJ Comunity et WebStorm⇒ <https://www.jetbrains.com/>

• Xampp ⇒ <https://www.apachefriends.org/xampp-files/7.2.11/xampp-linux-x64-7.2.11-0-installer.run>

Créez-vous un répertoire « B4Co » où vous le souhaitez. Dans ce répertoire nous mettrons tout les fichiers nécessaires à la configuration du projet (les dépôts git, les fichiers de confs maven, etc.)

*2. Git*

**Qu’est-ce Git ?**

Git est un logiciel de gestion de versions décentralisé. C'est un logiciel libre créé par Linus Torvalds, auteur du noyau Linux, et distribué selon les termes de la licence publique générale GNU version 2. En 2016, il s’agit du logiciel de gestion de versions le plus populaire qui est utilisé par plus de douze millions de personnes.

**Documentation (fr)**

La philosophie de Git est très particulière et spécifique à ce système de gestion de version (comparée à d’autres). Ainsi je vous invite en tant que néophyte à vous renseigner sur Internet.

Quelques liens utiles :

- Git sur wikipédia

- OpenClassroom

- Learn Git Branching (Ce site est un tuto ludique sur l’utilisation des commandes git afin de bien comprendre le fonctionnement des branches, il vous aidera grandement dans la compréhension de cet outil très puissant).

**Installation**

https://git-scm.com/download/linux

**Configuration**

Ouvrez le terminal (Ctrl+Alt+T) puis tapez les commandes suivantes (avec vos identifiants) :

$ git config --global user.name "John Doe"

$ git config --global user.email johndoe@example.com

$ git config --global http.proxy http://proxy.boulnet.com:8080

$ git config --global https.proxy http://proxy.boulnet.com:8080

*3. Consul*

**Qu’est-ce que consul ?**

Consul a plusieurs composants, mais dans son ensemble, c'est un outil pour découvrir et configurer des services dans une infrastructure logiciel. Il fournit plusieurs fonctionnalités principales.

**• Service Discovery (catalogue de service)**

◦ Les clients consul peuvent fournir un/des service(s), tel que api ou BDD, et d'autres clients peuvent utiliser Consul pour découvrir les fournisseurs de services de données. En utilisant DNS ou HTTP, les applications peuvent facilement trouver les services dont elles dépendent.

**• Health Checking (Contrôle d’activité et santé du service)**

◦ Les services consul peuvent fournir un nombre quelconque de contrôles de santé, soit associés à un service donné ("le serveur web renvoyant 200 OK"), soit avec le noeud local ("l'utilisation de la mémoire est inférieure à 90%"). Cette information peut être utilisée par un opérateur pour surveiller la santé du cluster et elle est utilisée par les composants de découverte de service pour empêcher le trafic d'être éloigné des hôtes malsains.

**• KV Store (clé/valeur)**

◦ Les applications peuvent utiliser le magasin de clés / valeur hiérarchique de Consul pour un certain nombre d'objectifs, y compris la configuration dynamique, la signalisation des fonctionnalités, la coordination et plus encore. L'API HTTP simple le rend facile à utiliser.

**• Multi Datacenter**

◦ Consul supporte plusieurs datacenters Out of the box. Cela signifie que les utilisateurs de Consul ne doivent pas s'inquiéter de la construction de couches d'abstraction supplémentaires pour se développer vers plusieurs régions.

**Téléchargement**

Télécharger consul à l’adresse suivante<https://www.consul.io/downloads.html>

• Sélectionnez Linux (attention à bien vérifier si votre distributions est en 32 ou 64 bits)

• Enregistrez l’archive

**Installation**

Dézipper l’archive à la racine de votre projet B4Co

A partir de là, vous pouvez lancer le script directement (./consul)

Si vous voulez pouvoir le lancer en ligne de commande dans un terminal de n’importe où, il vous faudra le rajouter dans votre path :

• Lancez un terminal, placez vous à l’emplacement où se trouve le script consul

• Récupérez le chemin absolu (pwd) et copiez le

• Tapez ensuite export PATH=$PATH:/votre/chemin/absolu

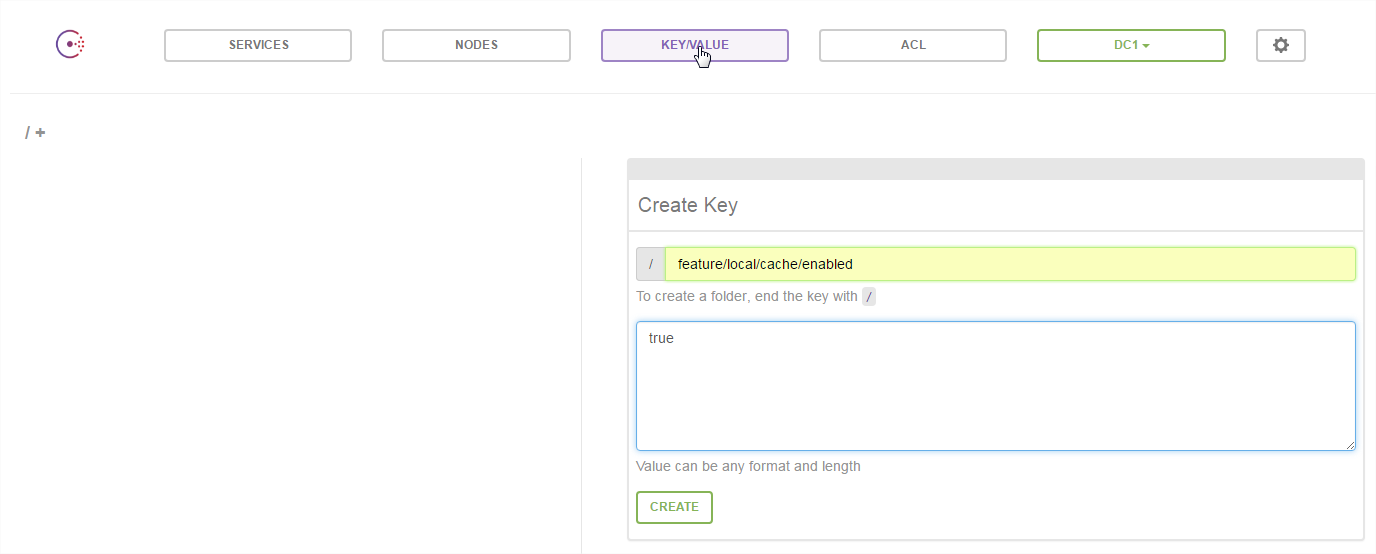
**Démarrage**

Consul se lance via la ligne de commande : consul agent -dev

**Paramétrage pour ‘Previon-3.0**

Une fois Consul démarré via la ligne de commande précédente, ouvrez votre navigateur vers la page : http://localhost:8500

Dans la section : KEY/VALUE ⇒ Créez la clé : feature/local/cache/enabled et entrez la valeur true, puis cliquez sur le bouton ‘Create’



*4. Hazelcast*

**Qu’est-ce que Hazelcast**

Hazelcast est une plate-forme de distribution de cache de données en cluster hautement évolutive.

Avec ses différentes structures de données distribuées, ses capacités de mise en cache distribuée, sa nature ‘élastique’, son support Memcache, son intégration avec Spring et Hibernate et, plus important encore, avec de nombreux utilisateurs heureux, Hazelcast est une solution de grille de données en mémoire riche en fonctionnalités, enterprise-ready et developer-friendly.

**Installation**

Dézippez l’archive à la racine de votre projet B4Co

**Démarrage**

Le démarrage se fait en ligne de commande en lançant le script « server.sh » =

Placez vous dans hazelcast-3.5.5/bin et tapez ./server.sh

La configuration par défaut est suffisante pour les développements, et ne demande aucune modification.

*5. Apache Maven*

**Qu’est-ce que Maven**

Apache Maven est un outil pour la gestion et l'automatisation de production des projets logiciels Java en général et Java EE en particulier. L'objectif recherché est comparable au système make sous Unix : produire un logiciel à partir de ses sources, en optimisant les tâches réalisées à cette fin et en garantissant le bon ordre de fabrication.

Il est semblable à l'outil Ant, mais fournit des moyens de configuration plus simples, eux aussi basés sur le format XML. Maven est géré par l'organisation Apache Software Foundation. Précédemment Maven était une branche de l'organisation Jakarta Project.

Maven utilise un paradigme connu sous le nom de Project Object Model (POM) afin de décrire un projet logiciel, ses dépendances avec des modules externes et l'ordre à suivre pour sa production. Il est livré avec un grand nombre de tâches pré-définies, comme la compilation de code Java ou encore sa modularisation.

Un élément clé et relativement spécifique de Maven est son aptitude à fonctionner en réseau. Une des motivations historiques de cet outil est de fournir un moyen de synchroniser des projets indépendants : publication standardisée d'information, distribution automatique de modules jar. Ainsi en version de base, Maven peut dynamiquement télécharger du matériel sur des dépôts logiciels connus. Il propose ainsi la synchronisation transparente de modules nécessaires.

**Installation**

Récupérez Maven :<http://wwwftp.ciril.fr/pub/apache/maven/maven-3/3.5.4/binaries/apache-maven-3.5.4-bin.tar.gz>

Décompressez l’archive où vous le souhaitez

Il vous faudra ensuite rajouter le dossier bin à votre path

export PATH=$PATH:votre/emplacement/apache-maven-3.5.4-bin/apache-maven-3.5.4/bin

Relancez votre shell et testez la version de maven avec la commande : mvn -v

**Configuration**

Remplacez le fichier ‘apache-maven-3.5.4-bin/apache-maven-3.5.4/conf/settings.xml’ par le settings.xml qui se trouve dans le wiki

*6. Node et npm*

**Qu’est-ce que Node.Js**

Node.js est une plateforme logicielle libre et événementielle en JavaScript orientée vers les applications réseau qui doivent pouvoir monter en charge.

Elle utilise la machine virtuelle V8 et implémente sous licence MIT les spécifications CommonJS.

Node.js contient une bibliothèque de serveur HTTP intégrée, ce qui rend possible de faire tourner un serveur web sans avoir besoin d'un logiciel externe comme Apache ou lighttpd, et permettant de mieux contrôler la façon dont le serveur web fonctionne.

Concrètement, node.js est un environnement d'assez bas niveau permettant d'exécuter du JavaScript non plus dans le navigateur web mais sur le serveur.

Il a été créé par Ryan Lienhart Dahl en 2009. Son développement et sa maintenance sont effectués par l'entreprise Joyent. Dahl a eu l'idée de créer Node.js après avoir observé la barre de progression d'un chargement de fichier sous Flickr : le navigateur ne savait pas quel pourcentage du fichier était chargé et devait adresser une requête au serveur web. Dahl voulait développer une méthode plus simple. Le serveur web Mongrel de Ruby a été l'autre source d'inspiration pour Dahl. Dahl avait échoué dans plusieurs projets en C, Lua et Haskell, mais, quand le moteur V8 fut diffusé, il commença à s'intéresser à JavaScript. Même si son idée d'origine était d'utiliser des entrées et sorties non bloquantes, il en reviendra un peu dans son module système car cela provoquait des problèmes de chargement de bibliothèques externes.

**Qu’est-ce que NPM**

npm (pour Node Package Manager) est le gestionnaire de paquets officiel de Node.js. Depuis la version 0.6.3 de Node.js, npm fait partie de l'environnement et est donc automatiquement installé par défaut. npm fonctionne avec un terminal et gère les dépendances pour une application. Il permet également d'installer des applications Node.js disponibles sur le dépôt npm.

**Installation**

En ligne de commande :

npm install -g npm@3.10.10

Vérifiez que les versions sont corrects :

En ligne de commande : ‘node --version’ ⇒ v6.10.2; ‘npm --version’ ⇒ 3.10.10 (ou supérieur).

Si vos versions sont plus récentes que les versions présentées ici, il vous faudra revenir à ces versions antérieures. Pour cela :

• Pour npm : npm install -g npm@3.10.10

• Pour node : Il vous faudra télécharger l’utilitaire « n » qui permettra de changer de versions

◦ npm install -g n

◦ n 6.10.2

◦ relancer le terminal pour prendre en compte les modifications (et vérifier que les versions sont correctes => node -v / npm -v)

*7. Java*

**Qu’est-ce que Java**

La plate-forme Java (the Java Platform en anglais, anciennement plateforme Java 2) est un standard de facto de plate-forme logicielle, produit par Sun Microsystems, puis Oracle Corporation, permettant de développer et d'exécuter des programmes écrits en langage Java indépendants de tout processeur et de tout système d'exploitation, conformément à la technologie Java.

Toute plate-forme Java se compose principalement d'un moteur d'exécution (appelé une machine virtuelle Java, ou 'JVM') et d'un compilateur fourni avec un ensemble de bibliothèques standards dont il existe plusieurs implémentations pour divers matériels et systèmes d'exploitation, de façon que les programmes Java puissent s'exécuter de façon identique sur chacun d'entre eux.

Chaque version de plate-forme Java répond à des spécifications normalisées dans les Java Specification Requests ou (JSR). Plusieurs versions de cette plateforme cohabitent mi-2015,

avec des JVM propres, Oracle implémentant et proposant notamment ses propres implémentations : HotSpot pour Java SE, serveur Glassfish pour Java EE.

À l'instar de la plateforme Microsoft .NET ou de l'environnement Eclipse qui ciblent plusieurs langages, la plateforme Java vise à supporter d'autres langages de programmation que son langage natif Java.

**Installation**

En ligne de commande : dnf install java-1.8.0-openjdk.x86\_64

En ligne de commande vérifiez la version ‘java -version’.

Dans le répertoire ‘jre/lib/security’ de votre installation java, déposer le contenu du fichier cert.zip

depuis une invite de commande faire :

cd <Répertoire JAVA>/jre/lib/security

puis pour chaque fichier faire :

..\..\bin\keytool.exe -import -trustcacerts -file <Nom du fichier> -keystore cacerts -storepass changeit -noprompt -alias <Nom du fichier>

**Clonage des dépôts b4co-back et b4co-front**

Pour clôner les dépots (back et front), vous pouvez le réaliser en ligne de commande sans passer par IntelliJ ou WebStorm :

• Lancez un terminal et connectez vous en super user (sudo su)

• git clone git@gitlab.boulnet.com:b4co/b4co-front.git

• git clone git@gitlab.boulnet.com:b4co/b4co-back.git

Il se peut que malgré cela, vous obteniez une erreur d’authentification =

« fatal: Impossible de lire le dépôt distant.

Veuillez vérifier que vous avez les droits d'accès

et que le dépôt existe. »

Pour contourner cela vous aurez besoin d’une clé ssh afin de récupérer les dépôts sur le gitlab de Boulanger :

• Dans un terminal : ssh-keygen

• appuyez sur entrée 3 fois. Votre clé à été générée dans le fichier /.ssh/id\_rsa.pub

• cat ~/.ssh/id\_rsa.pub

• Récupérez la clé publique dans ce fichier « rsa... @domaine.localhost»

• Allez sur le gitlab de boulanger :

◦ En haut à droite, cliquez sur votre avatar => settings

◦ A gauche, sélectionnez « SSH Key s »

◦ Dans le champ « Key », collez la clé puis cliquez sur « Add key »

Vous avez maintenant une clé associée à votre compte root. Reclonez ensuite les dépôts dans votre workspace.

Ces dossiers ne seront accessibles que par root, si vous voulez donner les droits à votre profile => placez vous dans votre workspace => sudo chown R user:user b4co-\* (user = votre nom de profil sous linux)

Cela appliquera les droits à votre user sur les dépôts de manière récursive (sur tout les dossiers et fichiers enfants)

**Configuration IntelliJ et Webstorm**

IntelliJ et WebStorm sont des IDEs développés par JetBrains

Vous pouvez les télécharger directement sur le site de JetBrains

Une fois le fichier téléchargé, décompressez l’archive où vous le souhaitez et lancez le script à l’emplacement suivant /ideaIU-2018.2.2/idea-IU-182.4129.33/bin

En ligne de commande : ./idea.sh

Une fois IntelliJ ouvert, acceptez les valeurs par défaut.

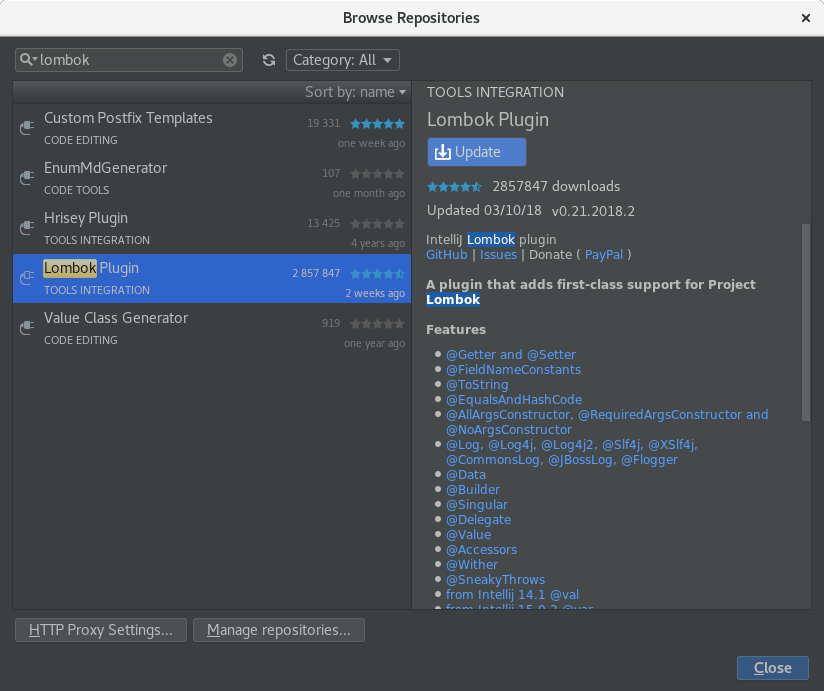
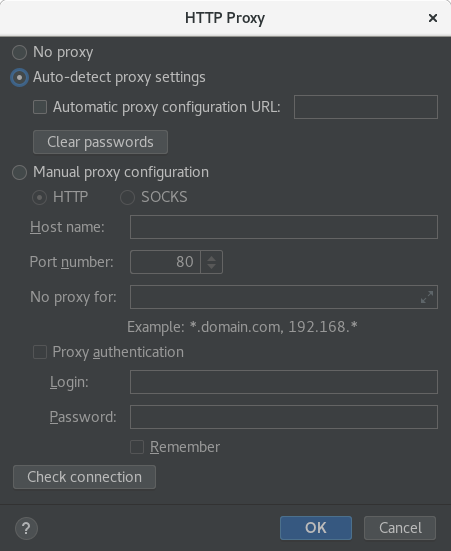
Vous arriverez sur un écran projet : cliquez sur configure puis settings



Dans la section Plugin, cliquez sur ‘Browse repositories’

recherche le projet Lombok puis cliquez sur le bouton ‘install’.

Si pas trouvé cliquez sur HTTP Proxy Settings et cochez ‘Auto-detect proxy settings’ pour relancer la recherche



Lancer le install puis accepter le redémarrage de Idea

*8. Xampp*

**Qu’est-ce Xampp**

XAMPP est un ensemble de logiciels permettant de mettre en place facilement un serveur Web local, un serveur FTP et un serveur de messagerie électronique. Il s'agit d'une distribution de logiciels libres (X (cross) Apache MariaDB Perl PHP) offrant une bonne souplesse d'utilisation, réputée pour son installation simple et rapide. Ainsi, il est à la portée d'un grand nombre de personnes puisqu'il ne requiert pas de connaissances particulières et fonctionne, de plus, sur les systèmes d'exploitation les plus répandus.

Il est distribué avec différentes bibliothèques logicielles qui élargissent la palette des services de façon notable : OpenSSL, Expat (parseur XML), PNG, SQLite, zlib… ainsi que différents modules Perl et Tomcat. Nombre de ces extensions étant inutiles aux débutants, une version allégée — version lite — est en conséquence aussi proposée.

**Installation**

Lancez le script xampp-linux-x64-7.2.9-0-installer.run

**Configuration**

Ouvrez le fichier /opt/lampp/apache2/conf/httpd.conf (avec nano ou vi)

Recherchez et décommenter (supprimer le ‘#’ de début de ligne) les lignes suivantes :

• LoadModule proxy\_module modules/mod\_proxy.so

• LoadModule proxy\_ajp\_module modules/mod\_proxy\_aLoadModule proxy\_http\_module modules/mod\_proxy\_http.sojp.so

Ajouter les lignes suivantes :

• ProxyPass /rest/ http://localhost:8095/

• ProxyPassReverse /rest/ http://localhost:8095/

Vous trouverez le fichier complet httpd.conf sur le wiki.

*9. Datasync.properties*

Après avoir récupéré l’archive zip datasync.properties.zip, décompressez la dans le répertoire de votre dépôt local

Il y aura alors quelques manipulations à effectuer sur les différents fichiers du répertoire :

• modifier le fichier datasync.properties et datasync.properties\_dev : Changer votre username et mdp (normalement User : dev et Password : dev-pass)

aux lignes « #Cache Cluster name » et « #Cache Cluster password »

• modifier les deux fichiers jetty-env.xml et jetty-env.xml\_dev :

à la ligne :

<Arg type="java.lang.String">file:///C:/Users/EX1001/Documents/Worskpace/Configuration/b4co-back/datasync-properties/</Arg>

⇒ Changer le chemin indiqué et le remplacer par le chemin absolu de votre répertoire datasync.properties (après le « file:/// ». Ici le dossier datasync.properties a été placé dans C:/Users/EX1001/Documents/Worskpace/Configuration/b4co-back/datasync-properties/). Faire cela pour les deux fichiers

**Rajouter les certificats dans la jvm**

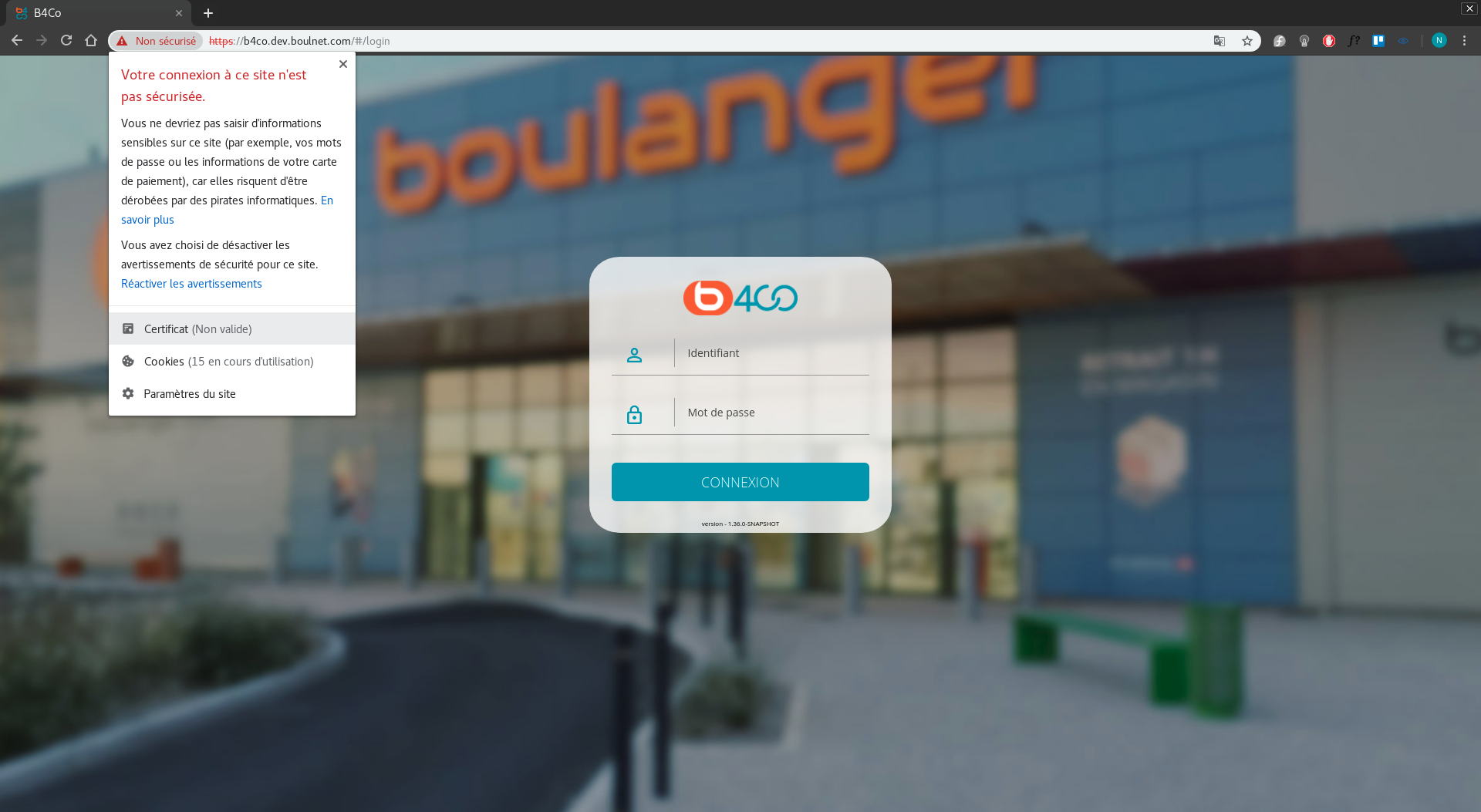
Afin que le back puisse accéder aux sites de B4Co il vous faudra récupérer les certificats d’autorités de ces sites et les rajouter à votre navigateur.

Pour ce faire, accédez à ces deux urls dans le navigateur de votre choix (ici j’utiliserais google chrome mais la manipulation reste la même pour tous les navigateurs) :

* <https://b4co.dev.boulnet.com/>
* <https://b4co.qua.boulnet.com/>



Cliquez sur et sélectionnez « Certificat »



Une page s’ouvre vous indiquant le certificat utilisé pour ce site. Accédez à l’onglet détail et cliquez sur en bas.

Enregistrez le certificat où vous le souhaitez. Faites de même pour le deuxième site.

Il vous faudra ensuite utiliser la commande « keytool » afin de rajouter les certificats directement dans les fichiers de la jvm.

Tout d’abord rajouter dans votre path le lien vers la commande keytool :

export PATH=$PATH :/usr/lib/jvm/jre-1.8.0-openjdk/bin

Testez ensuite la commande keytool -list

Vous devriez normalement avoir ce message :

« erreur keytool : java.lang.Exception: Le fichier de clés n'existe pas : /nathan/.keystore »

Lancez ensuite la commande : keytool -list -keystore /usr/lib/jvm/jre-1.8.0-openjdk/lib/security/cacerts

Entrez le mot de passe « changeit » et une page va s’afficher avec la list edes certificats déjà présents.

Nous allons ensuite importer les deux certificats par la commande :

keytool -import -trustcacerts -file <certificat.pem> -alias monAlias -keystore /usr/lib/jvm/jre-1.8.0-openjdk/lib/security/cacerts

Faite de même pour le deuxième certificat.

NB : Remplacez <certificat.pem> par le nom du certificat que vous avez téléchargé