

Mode opératoire

Utilisation GitLab

Table des matières

1. INTRODUCTION.....	3
1.1. PRESENTATION DE GITLAB.....	3
1.2. OBJECTIFS DU DOCUMENT :	3
2. PRE-REQUIS	3
2.1. OUTILS NECESSAIRES.....	3
2.2. CREATION D'UN COMPTE GITLAB :	3
2.3. CONFIGURATION DE GIT SUR VOTRE POSTE LOCAL :	4
2.3.1. Téléchargement de Git :	4
2.3.2. Création d'un token pour accéder au projet JAVA :	4
3. GESTION DES BRANCHES.....	5
3.1. CREATION, MODIFICATION, ET SUPPRESSION DE BRANCHES.....	5
3.2. STRATEGIES DE BRANCHING :	5
4. COLLABORATION SUR GITLAB	6
5. CI/CD AVEC GITLAB	6
6. HISTORIQUE DES MODIFICATIONS	8
7. POINTS OUVERTS	8

1. INTRODUCTION

1.1. PRESENTATION DE GITLAB

GitLab est une plateforme DevOps offrant une suite d'outils intégrés pour gérer tout le cycle de vie du développement logiciel. Elle combine plusieurs fonctionnalités, notamment la gestion du code source, la gestion des versions, l'intégration continue (CI), la livraison continue (CD), la gestion des projets, et bien plus encore. Avec GitLab, les équipes peuvent collaborer efficacement pour concevoir, tester, et déployer des applications.

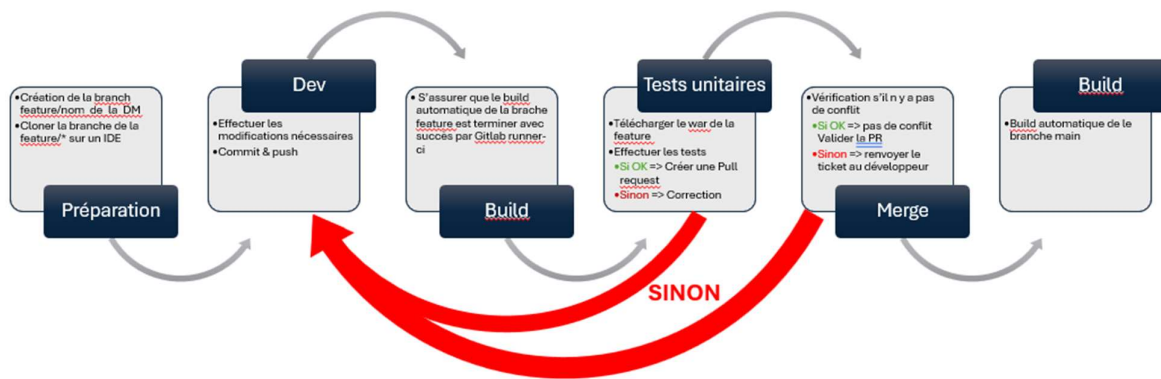
1.2. OBJECTIFS DU DOCUMENT :

Dans ce document, nous proposons de décrire l'utilisation de GitLab dans le cadre du projet SIHAM. Deux utilisations principales :

- Gestion des projets JAVA de bout en bout : Gestion des dépôts de codes sources, gestion collaborative et Intégration continue
- Gestion des projets hors JAVA : Gestion des dépôts de codes sources et gestion collaborative

Dans ce document, nous allons décrire la partie JAVA.

Le schéma global idéal se présente comme suit :



2. PRE-REQUIS

2.1. OUTILS NECESSAIRES

- Accès Gitlab : Nous accédons à Gitlab via l'URL <https://git.siham.amue.fr/>. Le code source peut être récupéré à partir d'un IDE.
- IDE Java (Eclipse, IntelliJ IDEA, etc.)

2.2. CREATION D'UN COMPTE GITLAB :

Les comptes Gitlab sont gérés par l'équipe TEC AMUE.

2.3. CONFIGURATION DE GIT SUR VOTRE POSTE LOCAL :

2.3.1. TELECHARGEMENT DE GIT :

Il est possible de télécharger en suivant ce lien :

<https://git-scm.com/download/win>

2.3.2. CREATION D'UN TOKEN POUR ACCEDER AU PROJET JAVA :

Afin de pouvoir récupérer les codes sources, il faudra créer un

Création d'un nouveau token d'accès pour le projet bdp en suivant les étapes suivantes

- Aller dans User Settings -> Access Tokens
- Saisir « bdp » dans le champ Token Name
- Cocher toutes les cases de la section « Select scopes »
- Cliquer sur le bouton « Create personal Access Token » → le token se génère dans le champ « Your new personal access token »

The image displays two screenshots of the GitLab web interface, specifically the 'User Settings' > 'Access Tokens' section.

Top Screenshot: Creating a new token

- Token name:** bdp
- Expiration date:** YYYY-MM-DD (calendar icon)
- Select scopes:** All scopes are checked:
 - ☒ api: Grants complete read/write access to the API, including all groups and projects, the container registry.
 - ☒ read_api: Grants read access to the API, including all groups and projects, the container registry, and
 - ☒ read_user: Grants read-only access to the authenticated user's profile through the /user API endpoint, public email, and full name. Also grants access to read-only API endpoints under /users.
 - ☒ read_repository: Grants read-only access to repositories on private projects using Git-over-HTTP or the Rep
 - ☒ write_repository: Grants read-write access to repositories on private projects using Git-over-HTTP (not using
- Buttons:** « Collapse sidebar », « Create personal access token »

Bottom Screenshot: Token created

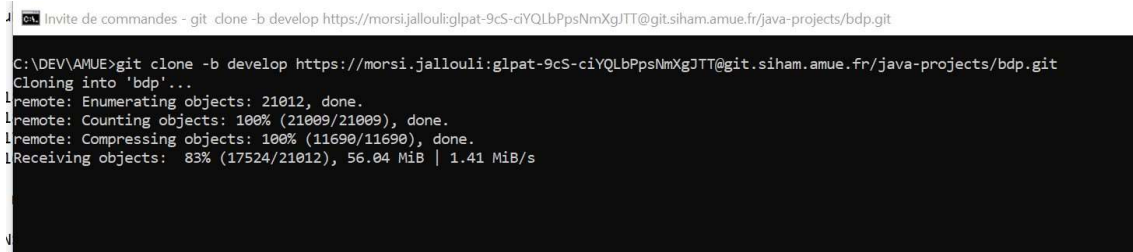
- Message:** Your new personal access token has been created.
- Your new personal access token:** glp6t-9c5-c1YQLbPpsNmXgJTT
- Warning:** Make sure you save it - you won't be able to access it again.
- Form below:** The 'Add a personal access token' form is visible again, with 'Token name' and 'Expiration date' fields.

3. GESTION DES BRANCHES

3.1. CREATION, MODIFICATION, ET SUPPRESSION DE BRANCHES

Pour la création d'une branche, il faut lancer la commande suivante avec l'invite de commande :

`git clone -b <nom_branche> https://prenom.nom:token@git.siham.amue.fr/java-projects/bdp.git`



```
invite de commandes - git clone -b develop https://morsi.jallouli:glpat-9cS-ciYQLbPpsNmXgJTT@git.siham.amue.fr/java-projects/bdp.git
C:\DEV\AMUE>git clone -b develop https://morsi.jallouli:glpat-9cS-ciYQLbPpsNmXgJTT@git.siham.amue.fr/java-projects/bdp.git
Cloning into 'bdp'...
remote: Enumerating objects: 21012, done.
remote: Counting objects: 100% (21009/21009), done.
remote: Compressing objects: 100% (11690/11690), done.
Receiving objects: 83% (17524/21012), 56.04 MiB | 1.41 MiB/s
```

La modification se fait sur l'IDE JAVA. Pour vérifier la liste des fichiers modifiés, il est possible d'utiliser la commande « `git status` »

Le développeur réalise les modifications nécessaires et effectue les tests unitaires (TU) en local pour vérifier que les changements n'introduisent pas de bugs.

Pour envoyer les modifications locales apportées à la branche en question ((exemple temp.txt) :

- `git add temp.txt`

Une fois le développement terminé et les TU validés, le développeur effectue un commit local (`git commit`) pour enregistrer les changements. Ensuite, il envoie les changements vers le serveur distant (`git push [nom-branche]`) :

Il faut préciser qu'à ce stade, seule la branche de travail du développeur est mise à jour sur le serveur GitLab, la branche principale n'est pas impactée.

- `git commit -m "Description du commit"`
- `git push`

Remarque : les commits doivent décrire le travail effectué. La description doit contenir au moins le numéro de la DM + un descriptif de la correction.

3.2. STRATEGIES DE BRANCHING :

Dans le cadre du projet SIHAM, il a été décidé de créer une branche par DM (ou version dans le cadre de la BDP). Si plusieurs DM sont traitées sur le même code source, le développeur met le numéro d'une seule DM et marque les autres dans la description de son commit.

La branche est créée exclusivement à partir de la branche main. Une commande permet d'être sûr que la branche principale est à jour : `git pull origin main`.

La création de la nouvelle branche peut se faire en ligne de commande :

`git checkout -b [Numéro_DM]`

Il est interdit de travailler directement sur la branche principale (branche main).

Les branches sont supprimées à la dernière étape, à la validation finale pour ajouter les modifications dans la branche principale. C'est le valideur qui choisit cette option :

8✓ **Approve** Approval is optional ?

✓ **Ready to merge!**

☒ **Delete source branch**
☐ Squash commits ?
☐ Edit commit message

1 commit and 1 merge commit will be added to main.

Merge

4. COLLABORATION SUR GITLAB

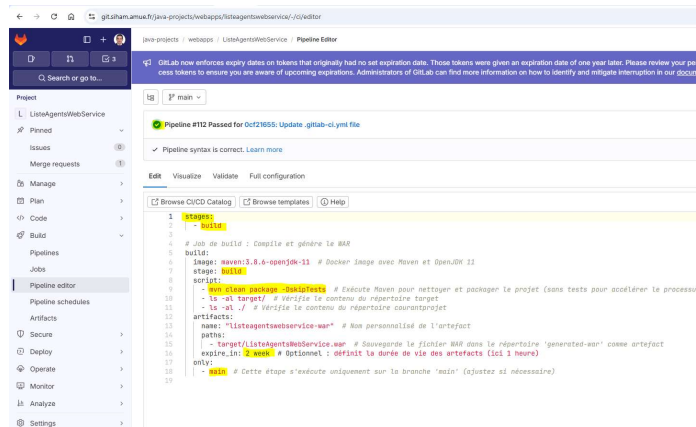
Lorsque la DM est validée en recette, une demande de « merge request » est envoyée par le développeur pour ajouter son code source sur la branche principale (la demande inclut une description des changements effectués et le lien avec la DM associée). Le valideur (ou le responsable du domaine) vérifie la cohérence du code par rapport à la liste des DM livrées et décide de valider ou non l'ajout des modifications sur la branche principale. Deux possibilités se présentent :

- Code conforme et cohérent : Le valideur effectue un merge pour ajouter le code dans la branche principale
- Code incohérent (présence de conflit) : Le valideur refuse le merge et un message est envoyé au développeur pour corriger son code.

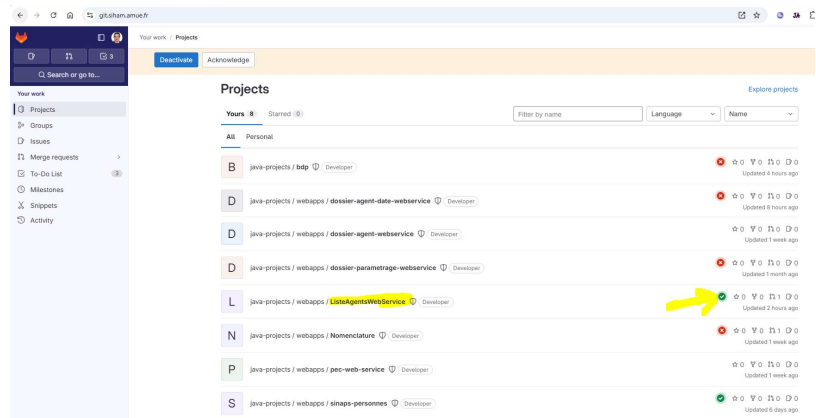
5. CI/CD AVEC GITLAB

Afin d'éviter toute erreur sur le code utilisé, la compilation du code source ne se fait pas en local et la génération du fichier .war (ou .jar) est effectuée directement dans Gitlab. Les étapes suivantes permettent de le faire :

- Création du fichier **gitlab-ci.yml** dans la racine du projet.



- Ajout d'un job de build dans le pipeline.



6. HISTORIQUE DES MODIFICATIONS

N° version	Etape	Date	Auteur
V1.00	Initialisation du document	10/12/2024	Bassem AKROUTI

7. POINTS OUVERTS

Identifiant	Commentaire
P.001	La compilation en ligne est bloquée à cause de l'absence de 2 bibliothèques HRA sur le serveur Nexus. Action : Ajouter les deux lib suivantes : - hr-openhr-api-7.17.04009.1022.jar - hr-kernel-7.17.04009.1022.jar