

# Capstone

---

## Capstone

---

### Intitulé

« Logiciel de gestion de rendus étudiants sur Gitlab »

### Description en une image

[capstone.pdf](#)

### Description détaillée

#### Contexte

Au département informatique, il est courant de demander aux étudiants de produire des logiciels et de les rendre à l'enseignant responsable pour retour ou évaluation. En parallèle, le département recommande aux enseignements et aux étudiants d'utiliser l'instance Gitlab de l'université pour stocker et versionner le code source des projets logiciels. Pour faciliter le rendu des projets, il apparaît donc naturel de reposer sur l'instance Gitlab, ainsi chaque projet peut être rendu en communiquant simplement une URL à l'enseignant.

#### Problème

Malheureusement, Gitlab est une forge logicielle avant tout dédiée au développement collaboratif de code, et non au rendu de projets étudiants. Ainsi un certain nombre de choses compliquent la vie d'un enseignant, comme par exemple :

- Pour donner accès à un projet à l'enseignant, un étudiant peut soit rendre son projet public, soit ajouter l'enseignant comme membre du projet. Comme rendre le projet public est hors de question, puisque cela pourrait encourager le plagiat, la seule solution est d'ajouter l'enseignant. Malheureusement, cette dernière solution va massivement "polluer" l'espace utilisateur (y compris sa page d'accueil) de l'enseignant avec possiblement une très très grande quantité de projets.
- L'enseignant n'a aucune manière simple de rechercher ou de regrouper tous les projets Gitlabs dédiés à une matière ou à un exercice donné, ou à une période donnée, il est donc rapidement difficile de s'y retrouver.
- Pour analyser, explorer, tester les projets étudiants, l'enseignant a besoin de les télécharger sur sa machine personnelle. Cela nécessite d'exécuter un (trop) grand nombre de fois la commande `git clone` manuellement. En outre, une fois téléchargé, le répertoire obtenu ne permet pas de facilement identifier le groupe d'étudiants ayant produit le rendu.
- Il n'y a pas de manière simple pour l'enseignant de faire un retour aux étudiants sur leur projet. Une possibilité est d'utiliser le système de ticket de Gitlab, mais le faire pour un grand nombre de projets est laborieux.

## Idée(s)

Pour répondre à tous ces problèmes, l'idée proposée ici est de mettre au point un outil simple – au moins dans un premier temps – pour permettre à un enseignant de recevoir des rendus via Gitlab, de travailler dessus, et éventuellement d'effectuer un retour auprès des étudiants.

Fort heureusement, Gitlab expose non pas une mais deux APIs permettant d'interagir programmatiquement avec la plateforme :

- une API REST classique : <https://gitlab.univ-nantes.fr/help/api/README.md>
- une API GraphQL : <https://gitlab.univ-nantes.fr/help/api/graphql/index.md>

Ces APIs permettent à un programme de récupérer des informations sur les projets et utilisateurs de l'instance Gitlab, ce qui permet ainsi éventuellement de gérer un téléchargement automatique des dits projets.

Pour pouvoir récupérer des projets privés tout en évitant que l'enseignant soit ajouté comme membre de chaque projet étudiant, une idée est de reposer sur un compte Gitlab "marionnette" créé juste pour être utilisé avant tout au travers de l'outil proposé ici. Ce compte pourra être ajouté comme "membre" des projets rendus à l'équipe enseignante, permettant à cette dernière d'y accéder sans utiliser leurs propres comptes.

En terme de fonctionnalités, on peut envisager les idées suivantes :

- Récupérer les informations sur tous les projets qui satisfont un ensemble de critères : période, mot-clé, membres, etc.
- Télécharger les dits projets dans des répertoires bien nommés.
- Pour chaque projet, enregistrer l'ensemble des informations importantes (membres, activité, etc.)
- Permettre à un enseignant de stocker sous la forme d'un fichier (eg. Markdown) un compte-rendu dans le répertoire d'un projet, et faire que l'outil puisse lire ce contenu et ouvrir un ticket dans le projet étudiant afin de lui faire un retour.

## Contraintes

- Le logiciel doit avant tout fonctionner sous Linux et éventuellement MacOS.
- Le logiciel doit reposer sur des technologies familières aux membres de l'équipe enseignante qui devront en faire la maintenance une fois le semestre terminé.
- Le logiciel doit être utilisable à minima en ligne de commande, et exposer toutes ses options sous la forme de paramètres à donner directement en appelant l'exécutable. Cela n'exclut pas la réalisation d'une UI additionnelle (par exemple web) si le temps le permet.
- Le logiciel doit être conçu afin de faciliter le développement d'un second outil permettant lui de faciliter l'évaluation automatique des projets étudiants. Par exemple, il serait intéressant d'enregistrer dans chaque projet télécharger l'ensemble des métadonnées du projet, cela afin de permettre à un second outil d'accéder facilement aux informations sur le projet.

## Possibilités d'aller plus loin

Si tous les objectifs sont atteints, deux axes sont explorables pour aller plus loin :

- Intégrer à l'outil une fonctionnalité pour exécuter l'**évaluation automatique** des projets étudiants. Une évaluation pourrait être effectuée par un outil externe qui serait appelé par l'outil – pourquoi pas via docker pour une parfaite réplication de l'évaluation.
- Étendre l'outil pour le rendre **utilisable sur le web de manière collaborative** par tout un groupe d'enseignant. Cette plateforme web servirait par exemple à afficher tous les projets dans une grande liste/tableau, afin de (1) monitorer les étudiants (qui a rendu ? qui a fait des commits ?), (2) répartir les projets entre les profs, (3) ajouter des commentaires à un projet, (3) noter les projets.

## Tentative existante

Une bibliothèque logicielle TypeScript a été conçue afin de faciliter l'évaluation des projets TypeScript rendus par des groupes de L1 via Gitlab. La bibliothèque est présente ici : <https://gitlab.univ-nantes.fr/naomod/eval-lib/>

Elle contient entre autre :

- des fonctions utilitaires pour interagir avec l'API REST de Gitlab : <https://gitlab.univ-nantes.fr/naomod/eval-lib/-/blob/master/src/gitlabUtil.ts>.  
On y trouve entre autre de quoi récupérer des projets, récupérer des commits, récupérer des auteurs, ajouter des issues, etc.
- des fonctions pour récupérer auprès de Gitlab toutes les informations sur tout un lot de projets, et les stocker au sein d'un fichier JSON : <https://gitlab.univ-nantes.fr/naomod/eval-lib/-/blob/master/src/extractAllGitlabData.ts>
- des fonctions pour télécharger tous les projets Gitlab obtenus dans un fichier JSON (cf point précédent) : <https://gitlab.univ-nantes.fr/naomod/eval-lib/-/blob/master/src/downloadProjects.ts>

Cette bibliothèque a été utilisée avec succès pendant deux années, et a reposé sur l'utilisation d'un compte Gitlab dédié "Naobot" que chaque étudiant devait ajouter comme membre de son projet à rendre : <https://gitlab.univ-nantes.fr/Naobot>. Ce compte Naobot était partagé entre les différents enseignants impliqués.

Cette première tentative a permis notamment d'observer que les interactions avec Gitlab peuvent être lentes, et qu'il est donc intéressant de d'abord extraire toutes les données d'importance (en gros la liste des projets), et cela avant de procéder au téléchargement des dits projets.