Mise à disposition d'un espace Gitlab et Jupyter pour les apprenants du Mooc "Recherche Reproductible"

Benoit Rospars, Laurence Farhi

Inria
LearningLab

Mise à disposition d'un espace Gitlab et Jupyter pour les apprenants du Mooc "Recherche Reproductible"

Benoit Rospars, Laurence Farhi, Inria Learning Lab

Keywords : Mooc, Gitlab, Jupyter, Notebook, Jupyterhub, SSO

Résumé

Ce rapport a pour objectif de décrire l'architecture et le déploiement d'un environement de travail pour éditer et historiser des notebooks dans le cadre du Mooc "Recherche reproductible : principes méthodologiques pour une science transparente". Nous aborderons le sujet avec une vue assez haut-niveau pour avoir un aperçu des contraintes techniques et comment les différents services ont été mis en place. Le principal enjeu étant d'utiliser le protocole LTI pour permettre une authentification unique pour l'apprenant à partir de son compte FUN. Nous décrirons comment nous avons su adapter des solutions open-sources tel que Jupyterhub et Gitlab pour répondre à ces besoins et à ces contraintes.

Table des matières

1. Contexte

Arnaud Legrand (chercheur en informatique, CNRS/Inria/Université de Grenoble), Christophe Pouzat (neurophysiologiste, CNRS/MAP5/Université Paris-Descartes) et Konrad Hinsen (biophysicien, CNRS/Centre de Biophysique Moléculaire à Orléan/Synchrotron SOLEIL) ont conçu le Mooc "Recherche reproductible : principes méthodologiques pour une science transparente" [1] proposé dans la plateforme FUN à partir d'octobre 2018, accompagné du Learning Lab Inria. Ce Mooc propose des principes méthodologiques pour une science transparente. Il aborde de manière pratique des thématiques telles que la prise de notes, le document computationnel, la réplicabilité des analyses.

Dans le cadre de ce Mooc, les participants sont formés à des outils primordiaux pour la recherche reproductible, en particulier :

- **Markdown** pour la prise de notes structurée
- des **Outils d'indexation**
- **Gitlab** pour le suivi de version et le travail collaboratif
- les **Notebooks** (Jupyter, RStudio ou Org-Mode) pour combiner efficacement calcul, représentation et analyse des données

De nombreux exercices pratiques sont proposés dans ce cours en ligne pour permettre aux apprenants d'expérimenter les outils et méthodes proposés. Ils ont le choix entre 3 parcours différents pour réaliser ces exercices :

- Jupyter et le langage Python (ou R) pour les débutants, car aucune installation n'est nécessaire sur son ordinateur
- RStudio et le langage R avec l'installation de RStudio nécessaire
- Emacs/Org-Mode et les langages Python et R avec l'installation d'Emacs, Python, et R nécessaire

Pour ce faire, chaque élève dispose :

- d'un espace **Gitlab** qui lui est personnel pour déposer les résultats des exercices et les partager avec les autres participants du Mooc et les enseignants
- d'un espace **Jupyter** pour réaliser des notebooks Jupyter pour ceux qui ont choisi de travailler avec cet outil. Il est à noter que les notebooks Jupyter peuvent être stockés dans l'espace Gitlab ci-dessus

2. Description du besoin

Un premier travail a été fait avec Arnaud Legrand, un des auteurs du Mooc, pour définir les besoins et les fonctionnalités attendus pour utiliser les notebooks dans le cadre de la recherche reproductible. L'architecture a donc été imaginée pour répondre à ces besoins.

Le premier besoin était de déployer notre propre instance Gitlab avec les contraintes suivantes :

- créer automatiquement les comptes, les participants étant déjà authentifiés via la plateforme FUN
- intégrer Gitlab dans la plate-forme FUN (pour pouvoir naviguer dans l'historique et éditer simplement des fichiers markdown)
- initialiser l'espace Gitlab des apprenants avec les fichiers (Markdown, texte, données type .csv, notebooks, ...) nécessaire pour les exercices. Cet espace est créé à partir d'un entrepôt "modèle" la première fois que l'apprenant rentre dans son espace

- pouvoir vérifier la réponse d'un exercice et renvoi d'un "diff" pour des fichiers de type Markdown, Jupyter, Orgmode ou RStudio
- avoir la possibilité de rentrer dans son espace Gitlab sans passer par FUN pour ceux qui le souhaitent

Puis de mettre en place un serveur Jupyter avec ces caractéristiques :

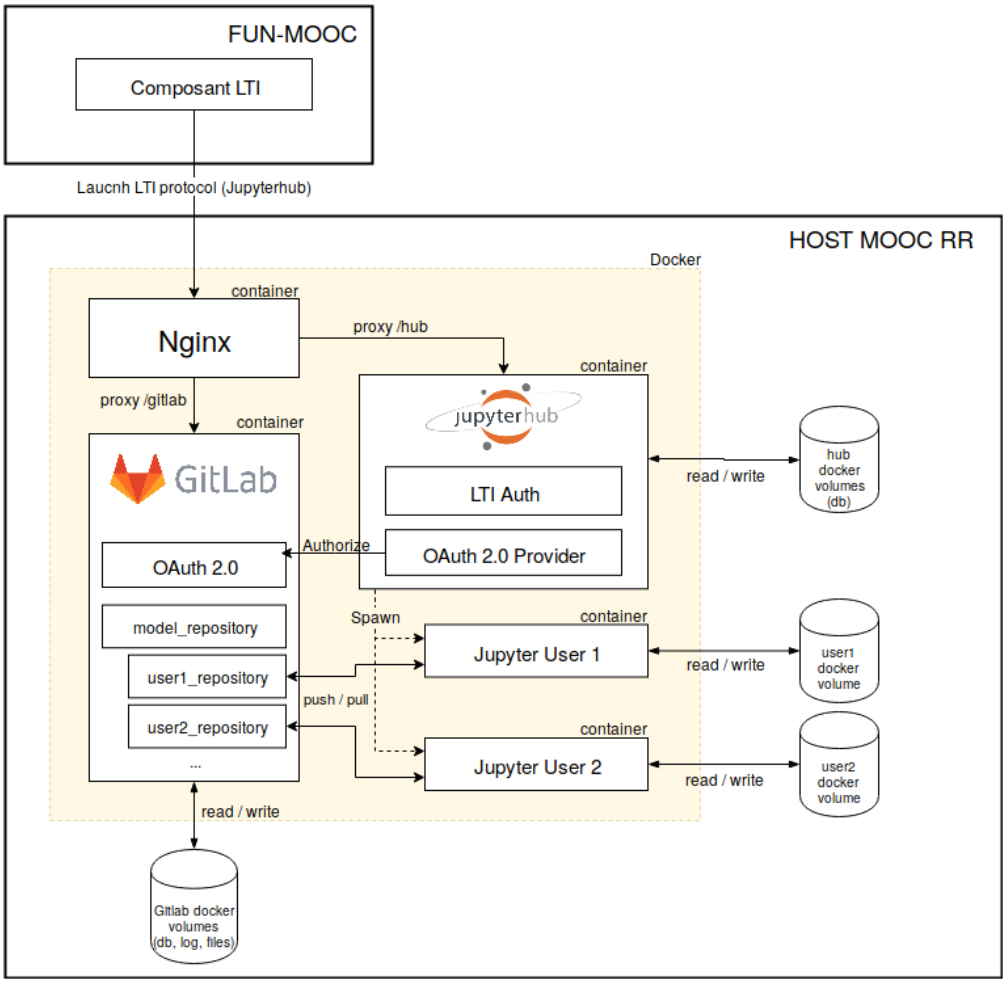
- avoir un mécanisme de synchronisation des notebooks dans Gitlab pour qu'ils soient consultables via Gitlab, mais pas exécutables.
- ajouter des boutons dans la barre des notebooks Jupyter pour pouvoir faire de push / pull depuis/vers Gitlab
- pouvoir écrire et exécuter du code en Python et R

3. Architecture technique

Les services web sont entièrement basés sur la technologie de containersDocker [2]. Cette solution permet d'isoler chacune des applications de façon légère et sécurisée tout en permettant un déploiement rapide et une orchestration pour fonctionner ensemble grâce notamment à Docker Compose [3]. Le tout est hébergé sur un serveur mis à disposition par la DSI Inria.

Le serveur du Mooc Recherche Reproductible contient donc 3 containers d'applications principaux :

- Nginx [4] qui sert de reverse proxy pour servir les différentes applications sur un même port (443) et domain
- Jupyterhub [6] qui permet de mettre à disposition un espace Jupyter à chaque utilisateur (notre version se base sur jupyterhub-deploy-docker [5]
- Gitlab Omnibus [7] la version auto-hébergée et tout-en-un de Gitlab



Infrastructure docker

Les images Docker sont générées et déployées à partir d'un fichier `docker-compose.yml`, d'un `Dockerfile` et d'un `Makefile`. Ce Makefile permet une configuration rapide grâce à des fichiers d'environnement pour définir des variables entre la production et l'environnement de développement.
L'ensemble des fichiers de configuration et de compilation sont disponibles dans le dépôt Git de notre projet [11].

4. Le protocole LTI pour intégrer des modules sur la plateforme FUN

4.1 Le protocole LTI

Le protocole LTI est un standard créé par IMS Global Learning Consortium. Il permet d'interconnecter des systèmes de gestion de l'apprentissage (LMS) tel que Moodle, Edx ou Canvas

avec des outils externes comme Lime Survey, Discourse ou Jupyter de façon sécurisée et transparente pour l'utilisateur.
La terminologie utilisée par le protocole est "LTI tool consumer" qui désigne le LMS utilisant l'outil externe et "LTI tool provider" qui désigne l'outil externe.

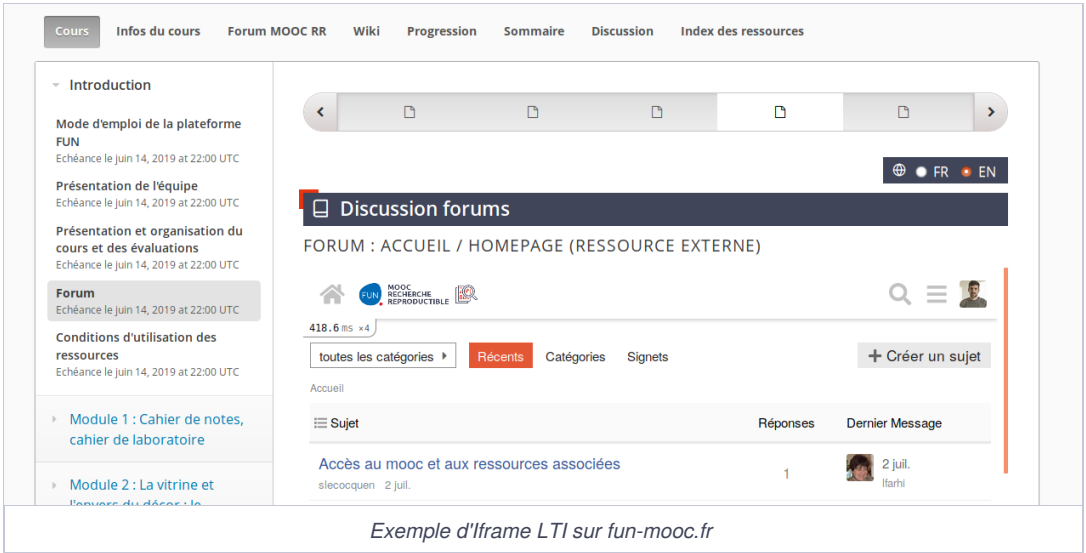
Un des principaux avantages de LTI est de permettre une authentification unique (Single sign-on) entre les services et de pouvoir créer à la volée les comptes des utilisateurs sur l'outil externe. Pour ce faire LTI utilise une paire de clés publique et privée partagée entre les deux applications permettant de valider de façon sécurisée les requêtes entre les deux services. Cet échange est lui-même basé sur le protocole sécurisé OAuth 1.0 [8] pour assurer la validité de la requête.

La deuxième fonctionnalité de LTI est de fournir un contexte à l'outil externe comme la provenance de l'utilisateur, son identité ou encore une URL de redirection pour rediriger l'apprenant vers une ressource spécifique de l'application externe. Ces informations sont transmises avec les données oauth par le moyen de paramètres cachés de formulaire.

4.2 LTI sur fun-mooc.fr

La plateforme FUN est basée surOpen Edx [9] : la version open source du système de gestion de l'apprentissage Edx, développé par le MIT.
FUN met à disposition un composant LTI permettant d'insérer au sein d'un cours un outil externe et de le configurer pour s'ouvrir soit dans une nouvelle fenêtre ou onglet, soit sous forme d'iframe directement incluse dans la page du cours. Cette dernière possibilité étant soumise à l'acceptation des iframes sur l'application tierce. Il faut ensuite paramétrer les clés partagées entre les deux applications et les données que l'on souhaite envoyer au service externe.

Une fois le composant correctement configuré, un bouton ou une iframe (selon la configuration) s'affiche dans la page du cours. Le compte de l'apprenant s'est créé automatiquement, s'il n'existait pas déjà et il s'est authentifié sur le service externe. Par exemple, il est possible d'utiliser un composant LTI pour afficher un forum de discussion Discourse, l'utilisateur étant automatiquement authentifié, comme nous pouvons le voir ci-dessous.



Exemple d'Iframe LTI sur fun-mooc.fr

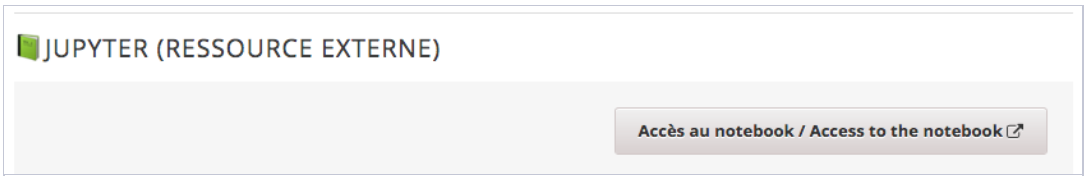
5. Intégration en LTI

5.1 Intégration de Jupyter en LTI

L'intégration d'une application tierce en LTI nécessite que celle-ci implémente le protocole comme un moyen d'authentification valide. Or, par défaut, Jupyterhub ne fournit pas ce moyen d'authentification. En revanche, il permet l'utilisation d'extensions pour modifier son comportement et une extension implémentant le protocole LTI existe.

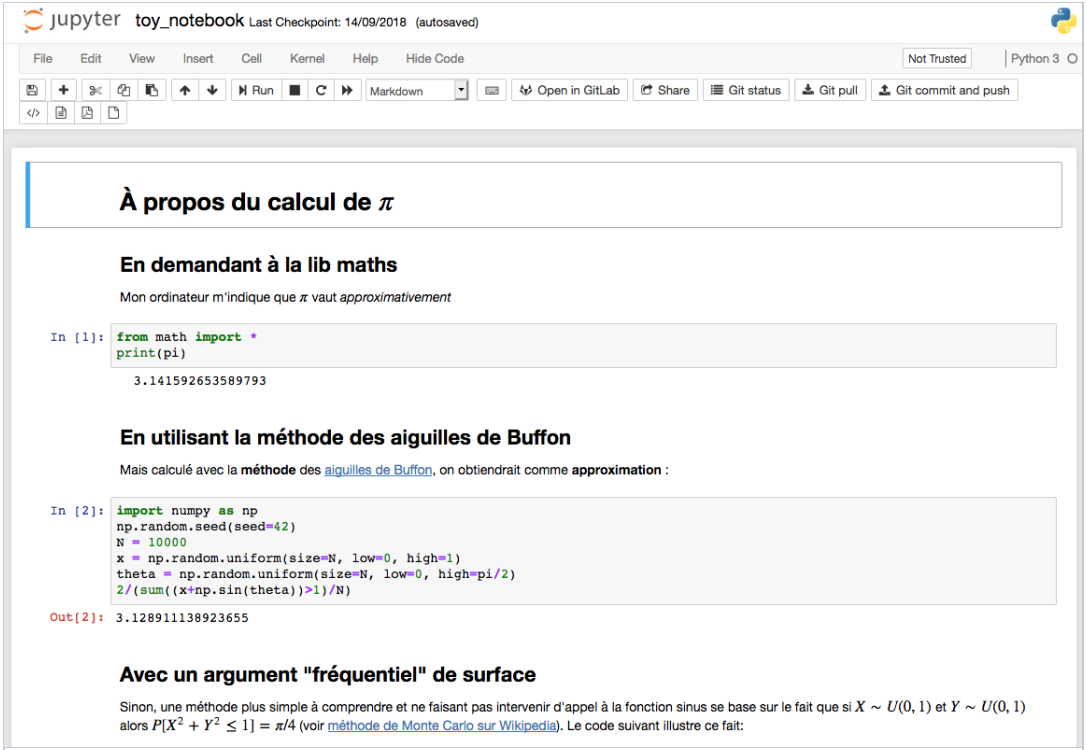
L'extension Jupyterhub LTI Authenticator [10] convenait tout à fait à notre besoin à une exception prêt. Il ne fonctionnait pas derrière un proxy inverse (serveur permettant l'accès à différents services sur le même port. Jupyter et Gitlab dans notre cas). Nous avons donc dû modifier l'extension pour que celle-ci fonctionne dans notre architecture d'application. Ces modifications ont ensuite été intégrées dans le dépôt officiel de l'extension.

Une fois cette extension configurée à l'aide des clés partagées avec la plateforme FUN son intégration se fait simplement via le composant LTI de FUN.



Bouton LTI vers un notebook du cours

Lors du clic sur le bouton LTI qui permet d'accéder à Jupyter, nous sommes redirigés dans un nouvel onglet vers le notebook jupyter configuré dans l'URL de redirection du composant LTI.



Interface des notebooks Jupyter

Le diagramme de séquence ci-dessous détaille la procédure du protocole LTI pour authentifier et rediriger l'utilisateur vers son propre espace Jupyter.

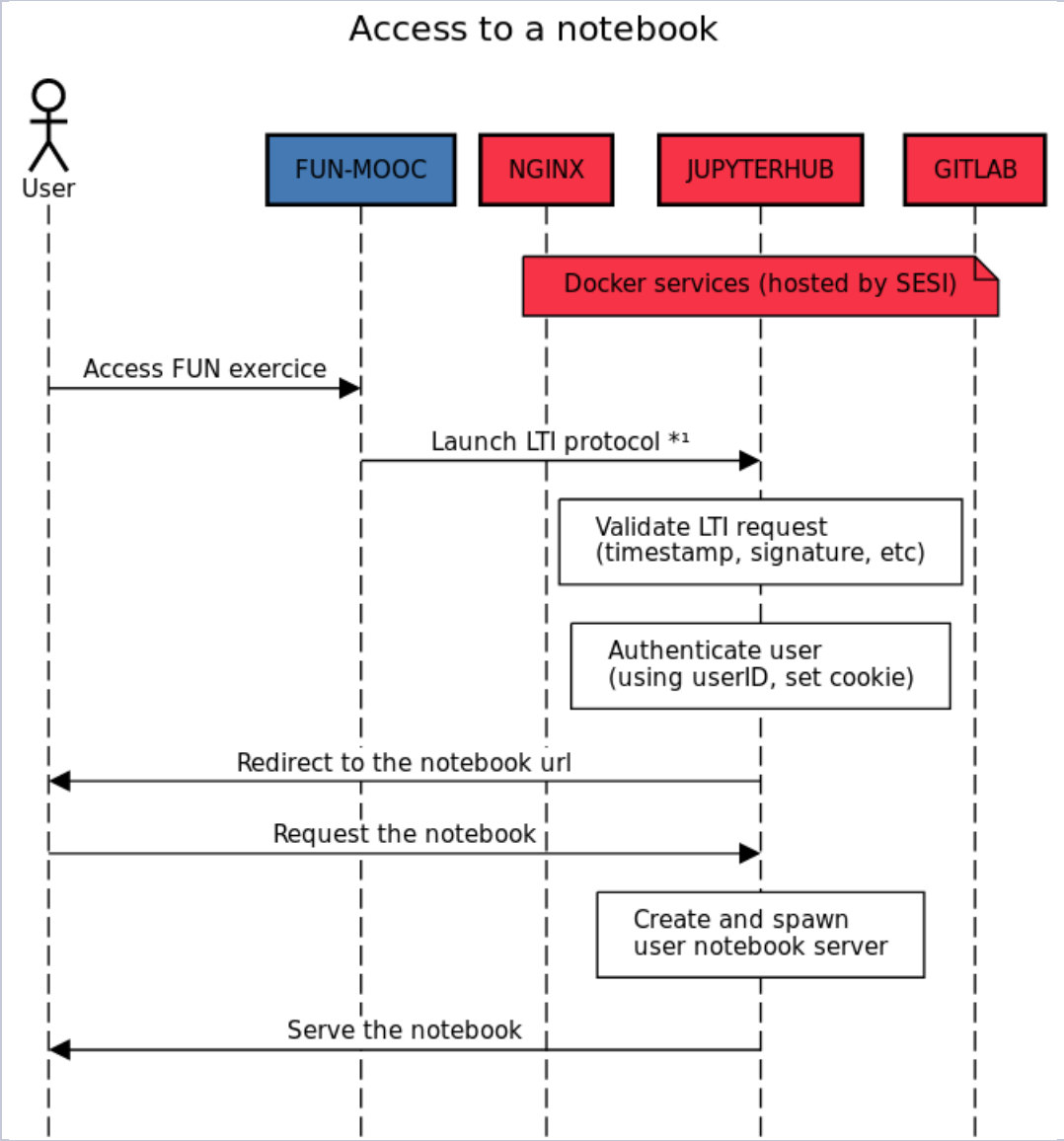


Diagramme de séquence LTI Jupyter

5.2 Intégration de Gitlab en LTI et OAuth 2.0

Notre volonté était de fournir le service Gitlab sans que l'apprenant n'ait à créer son compte afin d'éviter de multiplier les comptes tierces, maximiser le nombre d'apprenants qui utilisent le service et minimiser les problèmes techniques. Cependant, Gitlab ne supporte pas LTI comme méthode d'authentification, son intégration dans le cours a donc été plus complexe.

En revanche, Gitlab permet l'authentification par des services externes en utilisant le protocole OAuth 2.0. Et Jupyterhub peut être fournisseur d'authentification. Nous avons donc opté pour cette solution avec Jupyterhub comme serveur relais entre la plateforme FUN et Gitlab.

En résumé :

1. L'utilisateur accède à une page de la plateforme FUN avec un composant LTI pointant vers Gitlab
2. Il est dirigé et authentifié sur Jupyterhub grâce au protocole LTI
3. Puis il accède à la page de connexion de Gitlab
4. Clic du bouton de connexion Gitlab
5. Gitlab demande à Jupyterhub d'authentifier l'utilisateur
6. L'utilisateur accède à son compte

Tout ceci se fait de manière totalement transparente pour l'utilisateur et ne requiert qu'un simple clic à l'apprenant pour accéder à son compte.

Le diagramme de séquence ci-dessous détaille davantage l'intégralité du protocole d'accès à Gitlab.

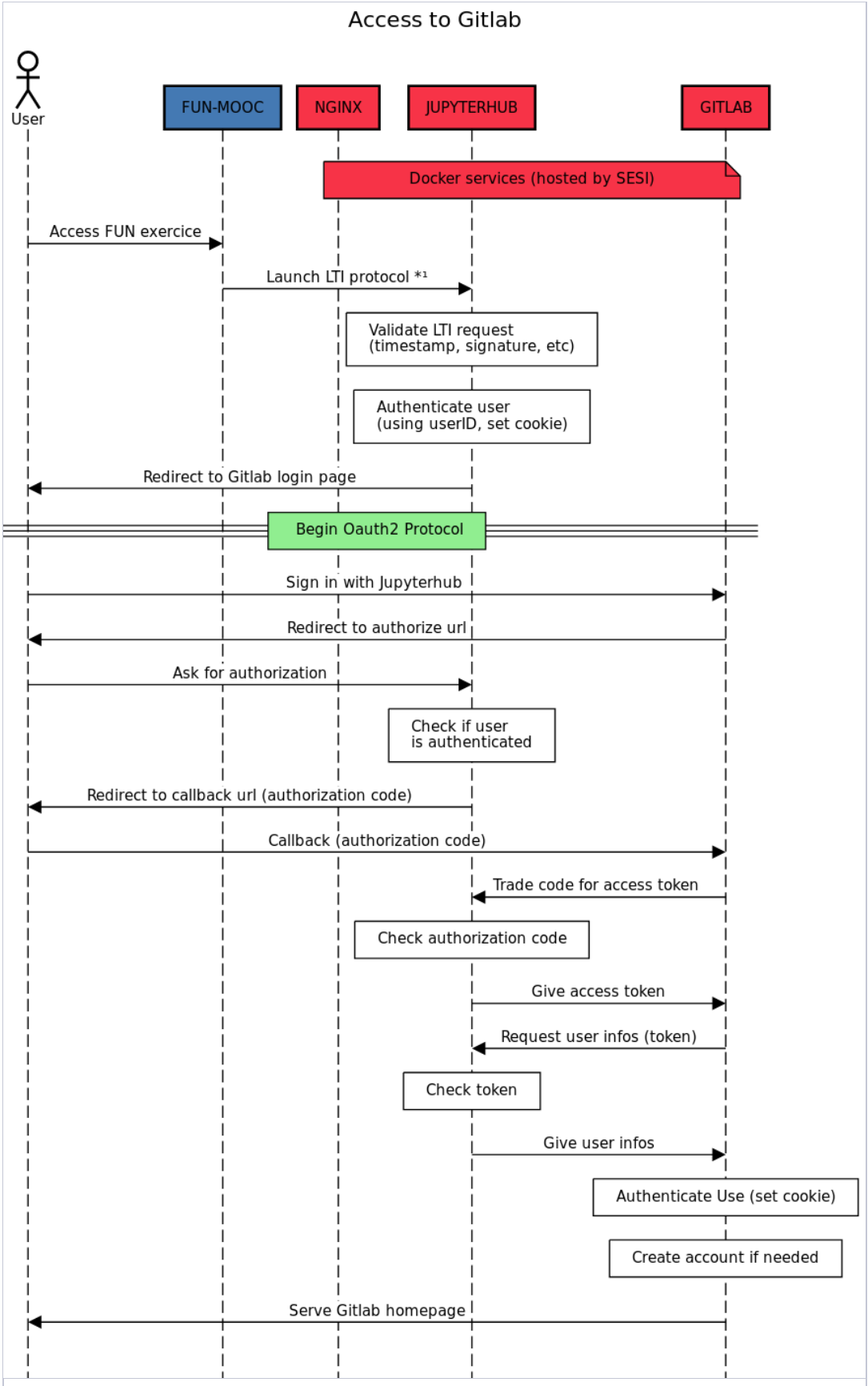


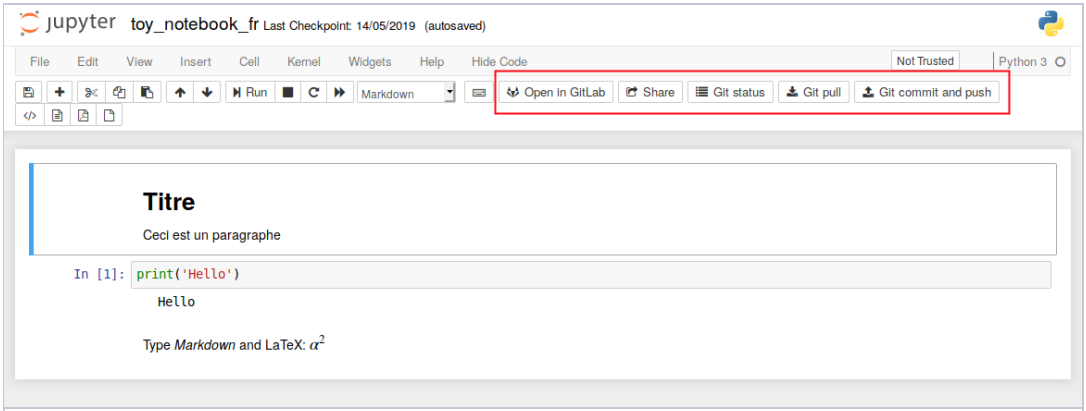
Diagramme de séquence LTI OAuth Gitlab

6. Extension Git pour Jupyter

Le développement de cette extension pour Jupyter avait pour but d'initier les utilisateurs à la gestion de version de leurs documents. Cependant, l'objectif pédagogique du cours n'était pas d'apprendre toutes les subtilités de Git, mais plutôt une version graphique et simplifiée avec simplement les commandes de base :

- **Git status** : pour visualiser les changements effectués depuis le dernier commit
- **Git commit** and push : pour sauvegarder et synchroniser avec Gitlab les derniers changements
- **Git pull** : pour mettre à jour les fichiers locaux depuis Gitlab
- Lien pour visualiser le fichier dans Gitlab

L'extension est développée en javascript pour l'affichage dans le navigateur et en python pour l'exécution des commandes git suivant la documentation des extensions Jupyter. Les boutons sont ajoutés dans la barre d'outils de l'interface de Jupyter et permettent les fonctions décrites plus haut.



Extension Git pour Jupyter

Conclusion

Ce Mooc fut un succès avec, pour la première session, plus de 600 comptes Gitlab crées et 230 utilisateurs de Jupyter. Bien que ce dernier soit gourmand en ressource très peu de problèmes techniques sont survenus.

L'utilisation de Docker comme base des applications Web permet une agilité de configuration et un déploiement rapide de nombreux services voire à dupliquer un service en quelques minutes. Grâce à cela, cette infrastructure sera réutilisée pour de prochains Moocs proposants l'utilisation de notebook Jupyter ou de Gitlab.

Bibliographie

- [1] Mooc Recherche Reproductible :https://www.fun-mooc.fr/courses/course-v1:inria+41016+session01bis/about
- [2] Docker :https://www.docker.com/resources/what-container
- [3] Docker Compose :https://docs.docker.com/compose/
- [4] Docker nginx :https://hub.docker.com/_/nginx
- [5] Jupyterhub :https://jupyterhub.readthedocs.io/en/stable/
- [6] Jupyterhub deploy Docker :https://github.com/jupyterhub/jupyterhub-deploy-docker
- [7] Gitlab Omnibus :https://docs.gitlab.com/omnibus/
- [8] Oauth :https://oauth.net/core/1.0/
- [9] Open Edx :https://open.edx.org/
- [10] LTI authenticator :https://github.com/jupyterhub/ltiauthenticator
- [11] Dépôt du projet :https://gitlab.inria.fr/learninglab/mooc-rr/mooc-rr-apps