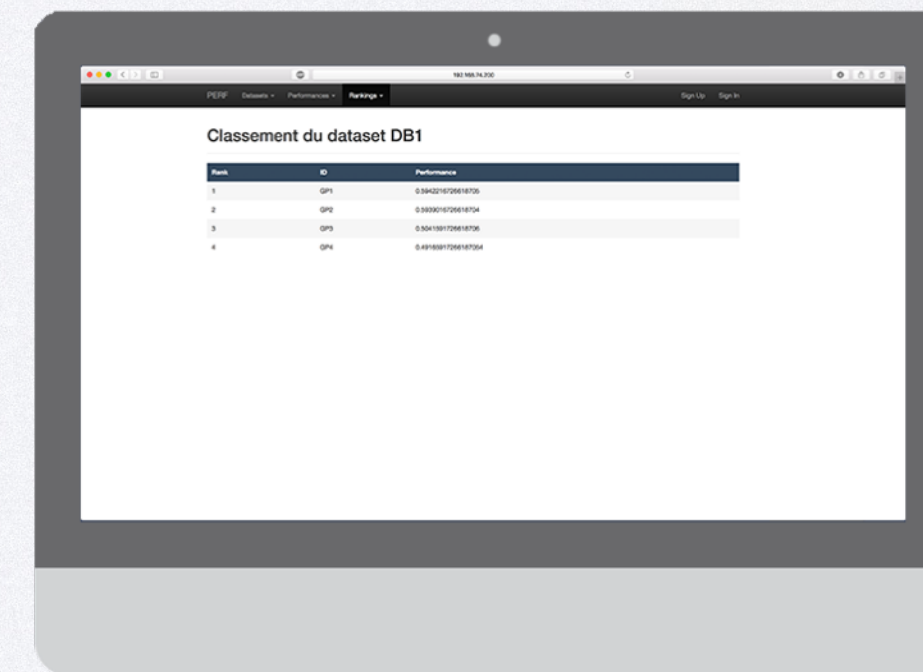
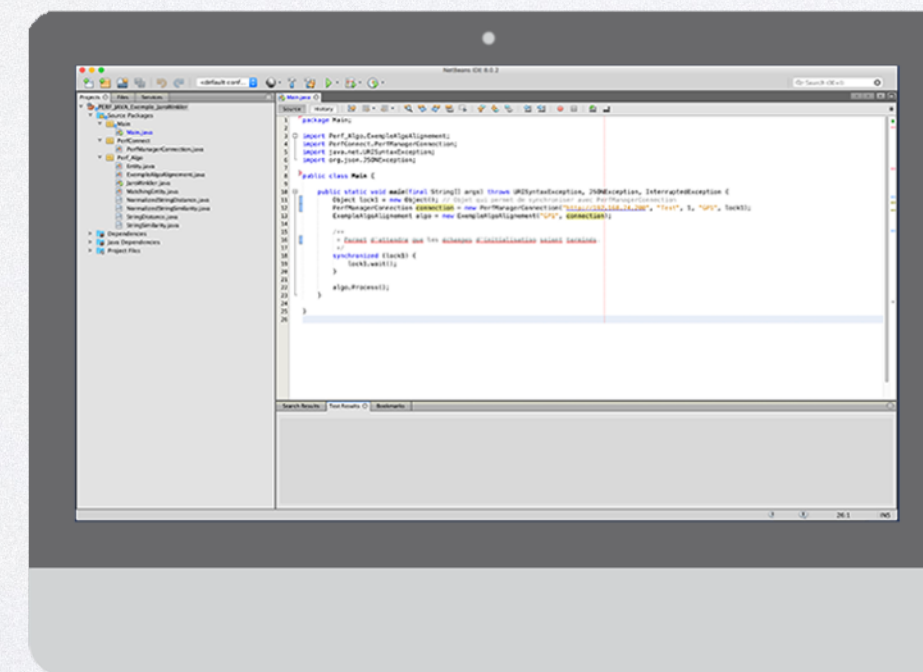
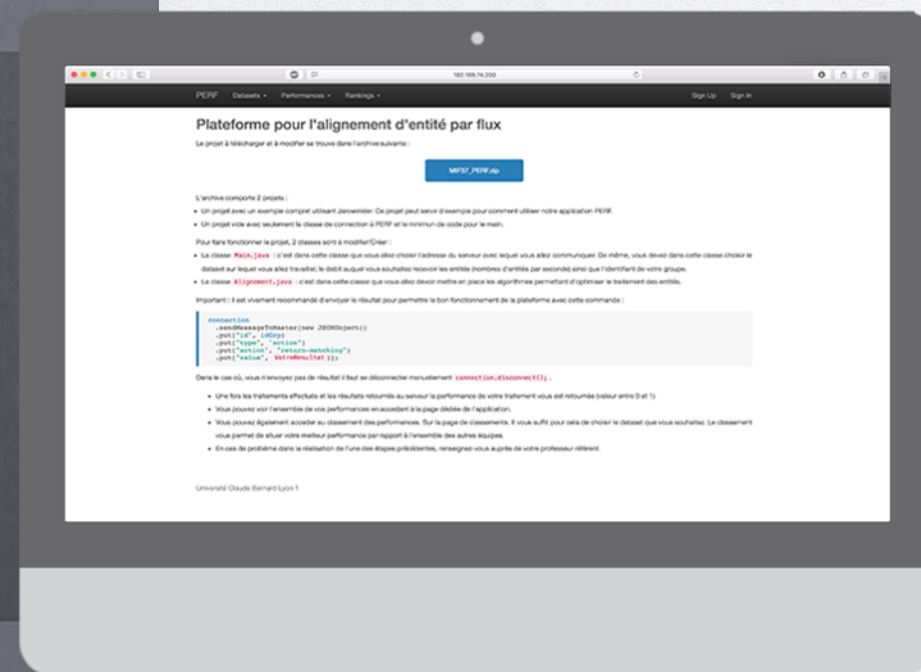




T15 - PROJET PERF



PLAN

Présentation

Équipe

Organisation et travail en équipe

Conception

Qualité technique

Démonstration

Difficultés rencontrées

Bilan

PRÉSENTATION

CONTEXTE

Munir les enseignants de base de données d'un outil permettant de **distribuer des flux d'entités sémantique** à destination d'une promotion d'étudiants et d'**évaluer la qualité du travail effectué** par ces derniers.

PRÉSENTATION

OBJECTIFS

- Concevoir et réaliser une **plateforme** :
 - **facilement déployable** et configurable à petite ou grande échelle
 - **administrable** par les enseignants (édition de datasets)
 - facilitant la communication **client-serveur** pour les étudiants
 - **évaluant les résultats** des étudiants et **proposant un classement**

PRÉSENTATION

FONCTIONNALITÉS

- Administration des datasets
- Répartition des clients sur les workers
- Distribution de flux d'entités
- Calcul des performances et établissement d'un classement
- Mise à disposition de l'interface Java et de l'énoncé du TP

ÉQUIPE

	Murat Halat	Antoine Hintzy	Corentin Lonjarret	Felician Matinya	Alexandre Millot
Chef de projet / Correspondant encadrants					X
Scrum Master			X		
Concepteur base de données			X		X
Expert technique Web		X			
Designer		X			
Développeur	X	X	X	X	X
Concepteur	X			X	

Porteurs du projet : **Fabien Duchateau** et **Nicolas Lumineau**, enseignants chercheurs au sein du LIRIS, pôle Data Science, équipe Base de Données

ORGANISATION ET TRAVAIL EN ÉQUIPE

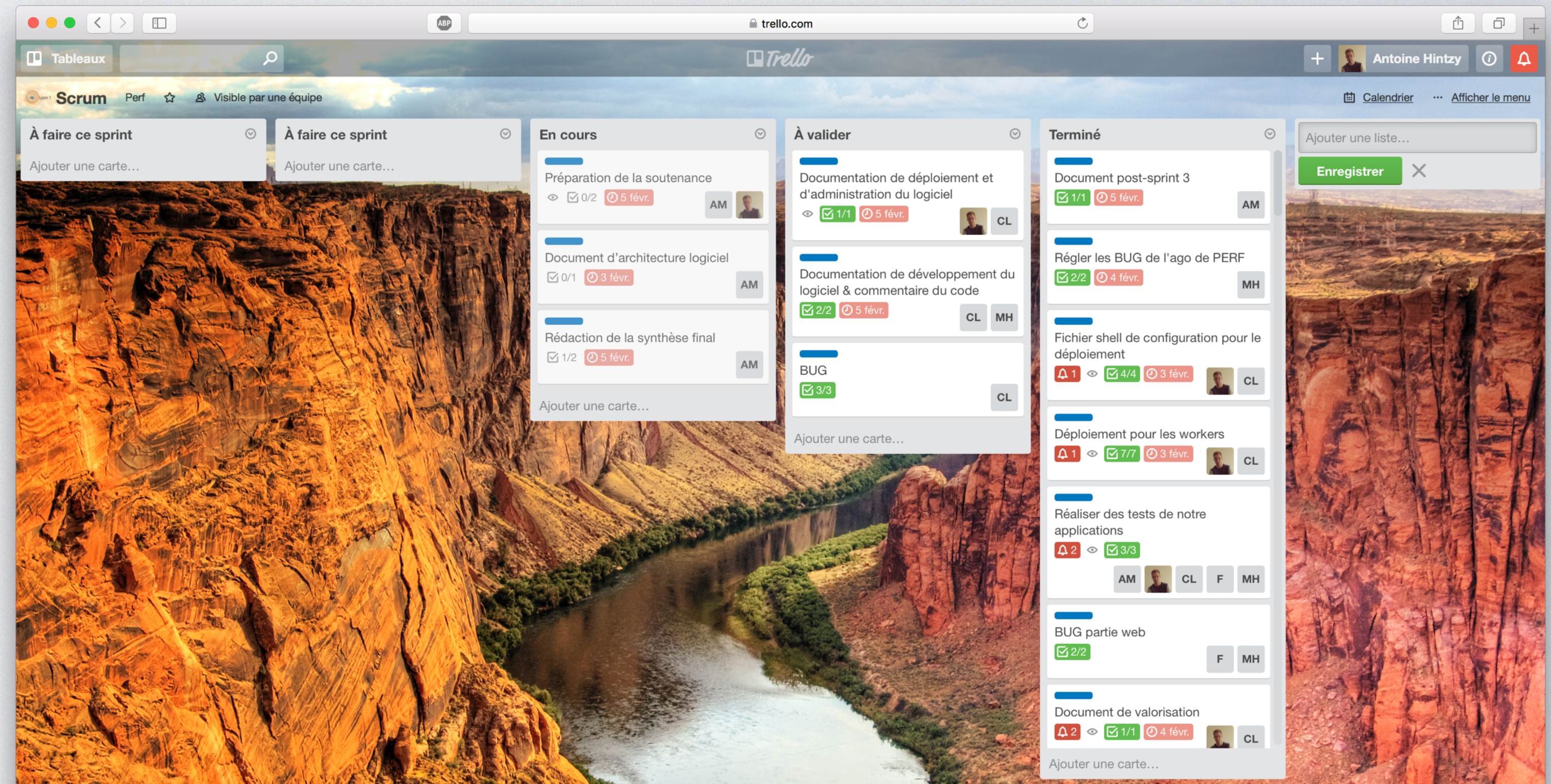
GESTION PRÉ-SPRINT ET POST-SPRINT

- Tâches à effectuer avant le sprint (configuration, installation, ...)
 - éviter la perte de temps au début du sprint
- La plateforme est dans un état stable à la fin de chaque sprint

ORGANISATION ET TRAVAIL EN ÉQUIPE

MÉTHODE AGILE (SCRUM)

- À faire
- En cours
- À valider
- Terminé

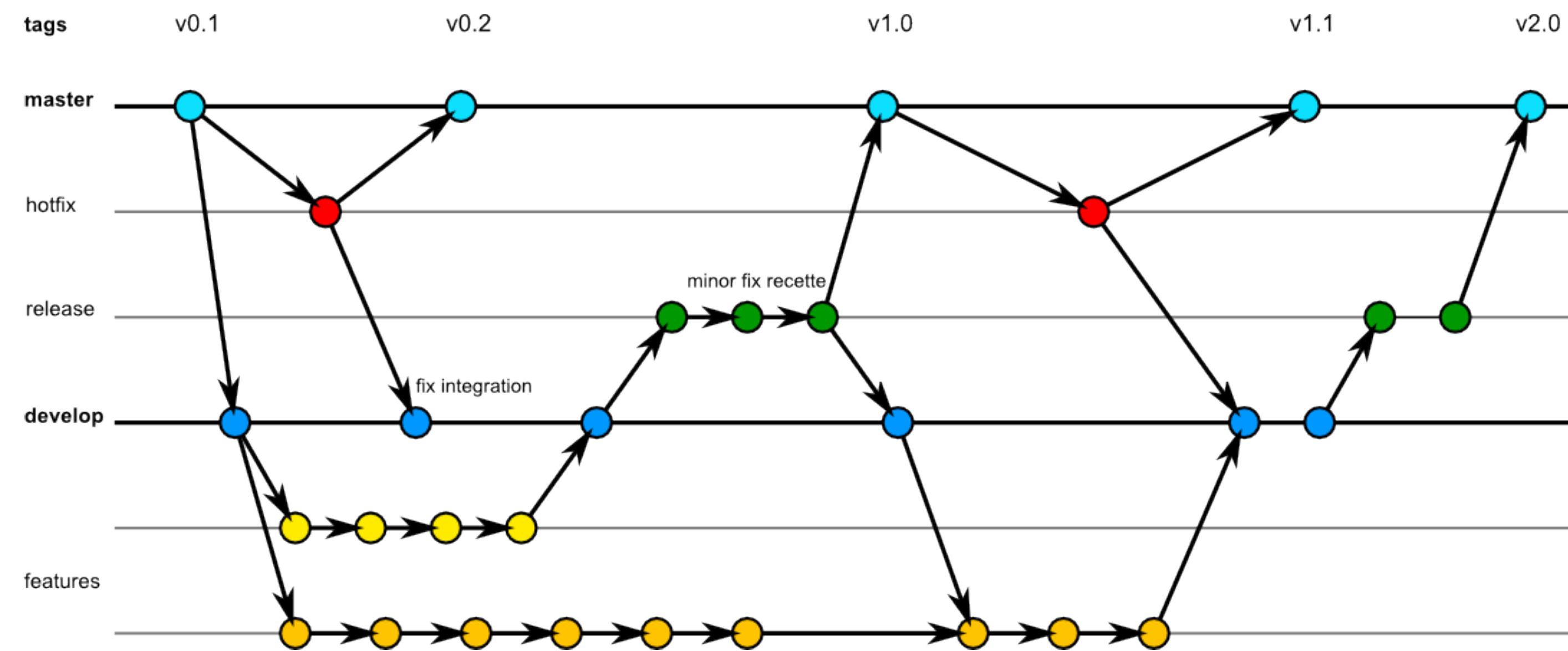


ORGANISATION ET TRAVAIL EN ÉQUIPE

MÉTHODES DE GESTION DE PROJET



- Sauvegarde et versionnage du code source avec **Git**
- Méthode **Git Flow** :



CONCEPTION TECHNOLOGIES

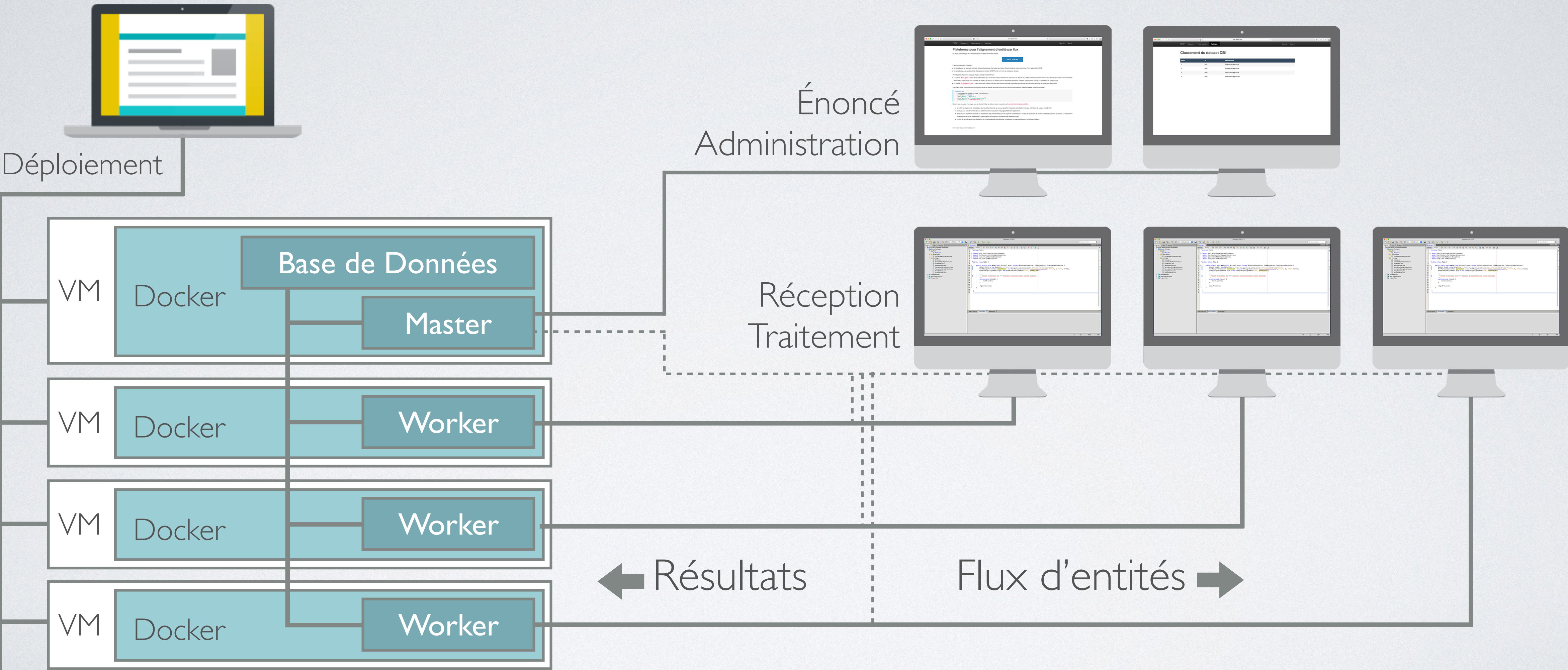


Déploiement



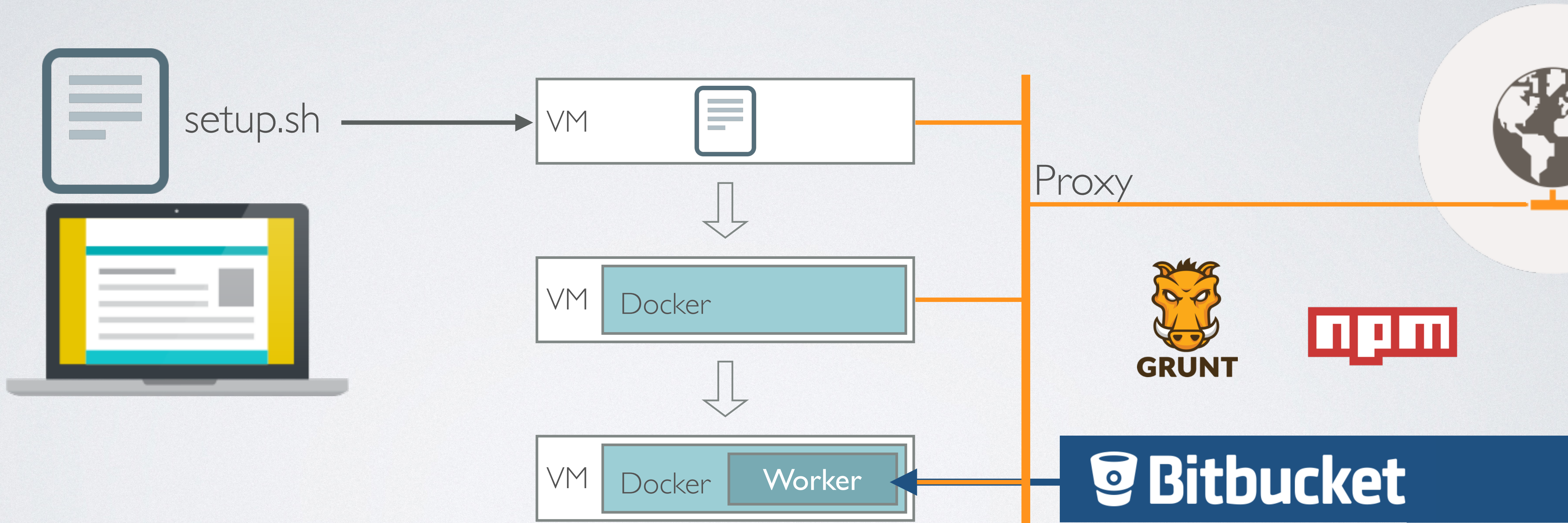
Serveur

CONCEPTION ARCHITECTURE



1 master
n workers

QUALITÉ TECHNIQUE DÉPLOIEMENT



QUALITÉ TECHNIQUE

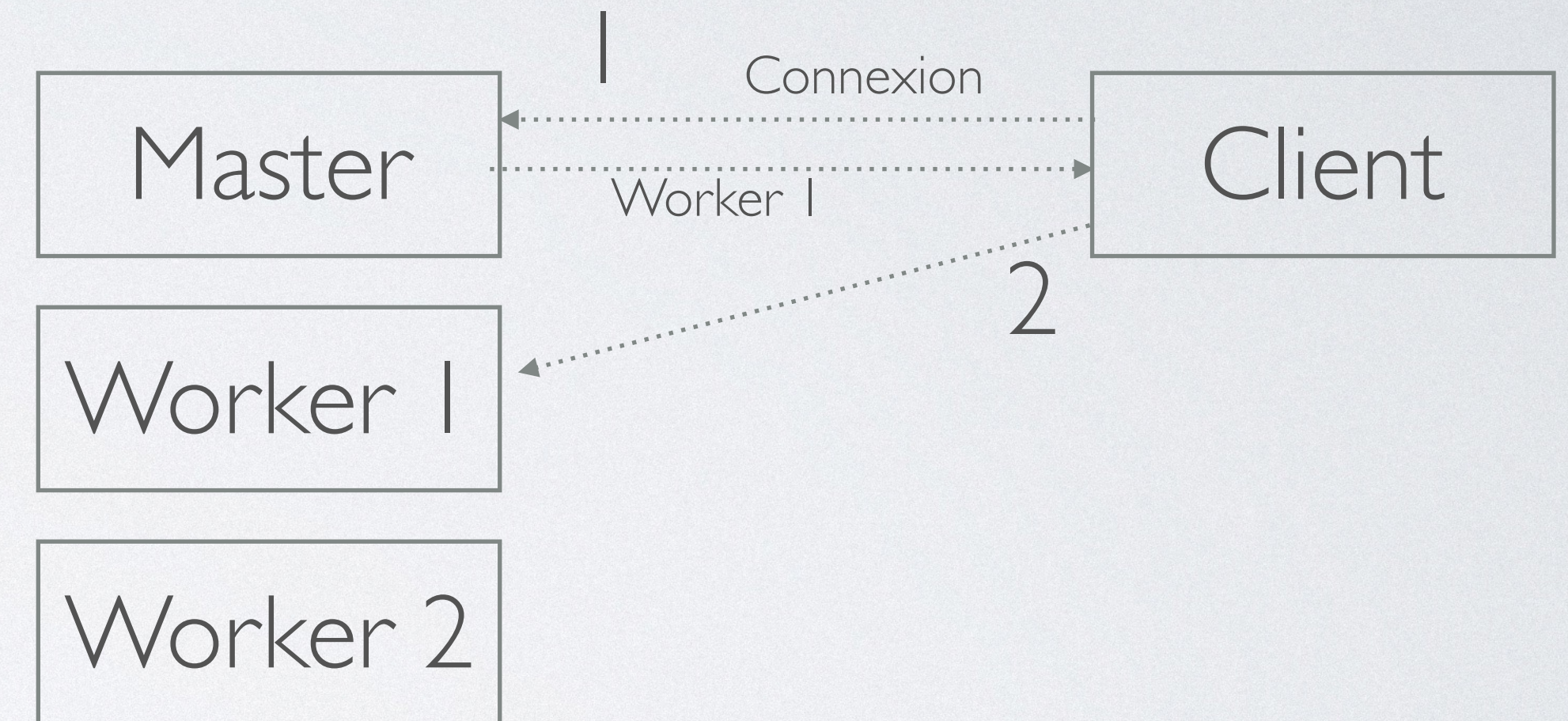
APPLICATION WEB

- 2 types d'utilisateur
 - admin, authentification requise (professeur)
 - client (étudiant)
- Permet d'administrer la plateforme
 - création de dataset
 - ajout de worker
- Pages d'aide à la prise en main du système
- Visualisation des classements

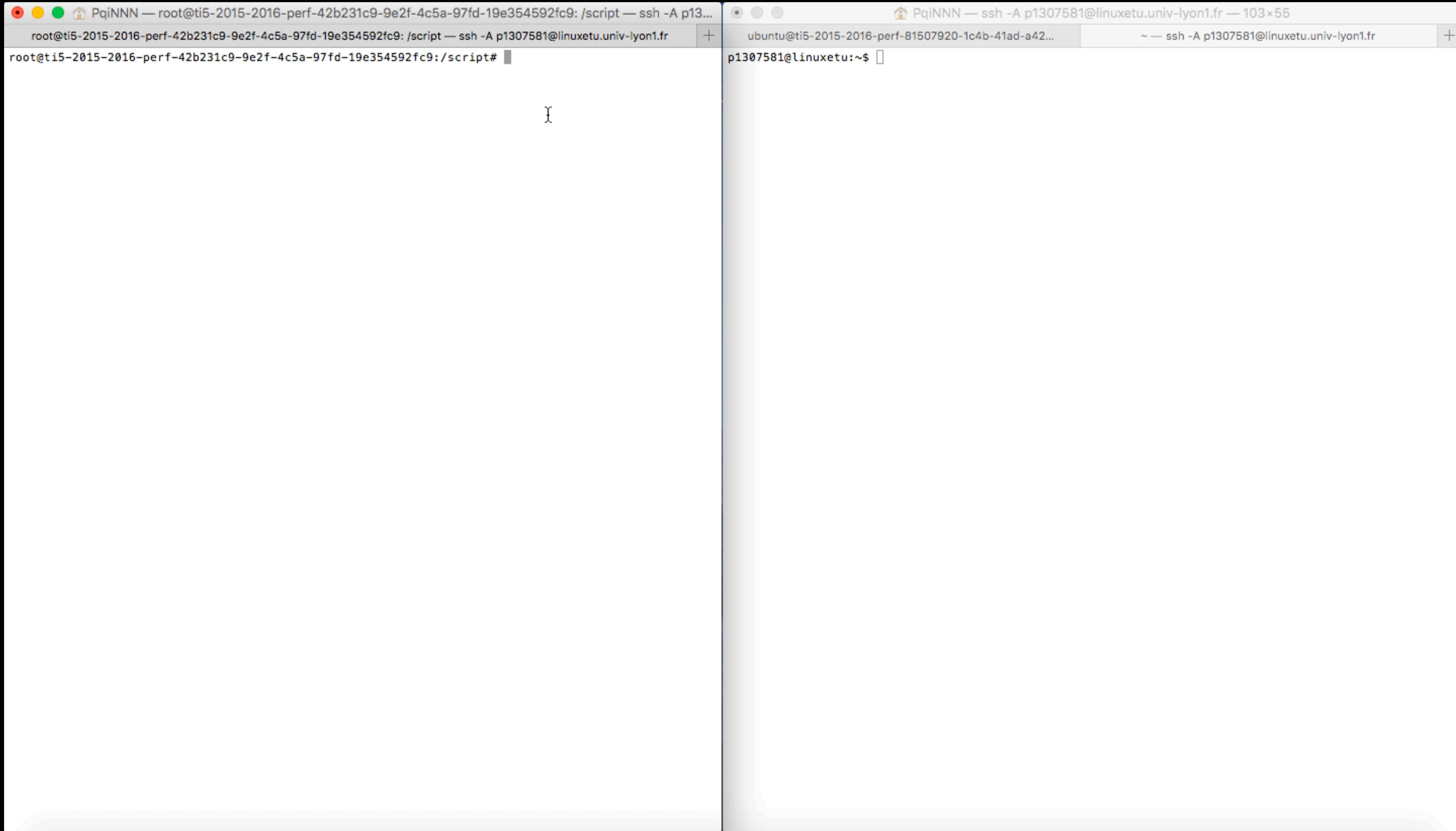
QUALITÉ TECHNIQUE

ÉCHANGES CLIENT-SERVEUR

- 2 sockets de connexion client-serveur (client-master et client-worker)
- Echange de messages au format JSON (entités)
- Calcul de performances



DÉMONSTRATION



Déploiement de l'application


```
root@ti5-2015-2016-perf-71797760-10bc-4c82-9538-7b6e9d5fac07: /docker/ti5-perf/PERF_MEAN_Master -- ssh -A p13...
master_prod | { worker1:
master_prod |   { id: 'worker1',
master_prod |     url: 'http://192.168.74.201',
master_prod |     active: false,
master_prod |     delai: [] },
master_prod |   worker2:
master_prod |     { id: 'worker2',
master_prod |       url: 'http://192.168.74.202',
master_prod |       active: false,
master_prod |       delai: [] } }
db | 2016-02-12T18:34:05.907+0000 I NETWORK [initandlisten] connection accepted from 172.17.
0.3:33986 #6 (6 connections now open)
db | 2016-02-12T18:34:05.908+0000 I NETWORK [initandlisten] connection accepted from 172.17.
0.3:33987 #7 (7 connections now open)
db | 2016-02-12T18:34:05.908+0000 I NETWORK [initandlisten] connection accepted from 172.17.
0.3:33988 #8 (8 connections now open)
db | 2016-02-12T18:34:05.909+0000 I NETWORK [initandlisten] connection accepted from 172.17.
0.3:33989 #9 (9 connections now open)
db | 2016-02-12T18:34:05.910+0000 I NETWORK [initandlisten] connection accepted from 172.17.
0.3:33990 #10 (10 connections now open)
master_prod | --
master_prod | MEAN.JS
master_prod | Environment:          production
master_prod | Port:                 8443
master_prod | Database:             mongodb://db/mean
master_prod | App version:          0.4.1
master_prod | MEAN.JS version:     0.4.1
master_prod | --
master_prod | Initialisation de Tab_Dataset -->
master_prod | { DB1: 4910, DB2: 4910, DB3: 4910, DB_Test: 4910 }
master_prod | Client #[#mmdYnn0HvGBvhZ05AAAA] is now connected.
db | 2016-02-12T18:34:08.782+0000 I NETWORK [initandlisten] connection accepted from 192.168
.74.202:44845 #11 (11 connections now open)
db | 2016-02-12T18:34:08.783+0000 I NETWORK [initandlisten] connection accepted from 192.168
.74.202:44846 #12 (12 connections now open)
db | 2016-02-12T18:34:08.784+0000 I NETWORK [initandlisten] connection accepted from 192.168
.74.202:44847 #13 (13 connections now open)
db | 2016-02-12T18:34:08.784+0000 I NETWORK [initandlisten] connection accepted from 192.168
.74.202:44848 #14 (14 connections now open)
db | 2016-02-12T18:34:08.785+0000 I NETWORK [initandlisten] connection accepted from 192.168
.74.202:44849 #15 (15 connections now open)
master_prod | Client #[#TbqoPW8JfZI514UAAAB] is now connected.
db | 2016-02-12T18:34:11.986+0000 I NETWORK [initandlisten] connection accepted from 192.168
.74.203:52459 #16 (16 connections now open)
db | 2016-02-12T18:34:11.987+0000 I NETWORK [initandlisten] connection accepted from 192.168
.74.203:52460 #17 (17 connections now open)
db | 2016-02-12T18:34:11.987+0000 I NETWORK [initandlisten] connection accepted from 192.168
.74.203:52461 #18 (18 connections now open)
db | 2016-02-12T18:34:11.988+0000 I NETWORK [initandlisten] connection accepted from 192.168
.74.203:52462 #19 (19 connections now open)
db | 2016-02-12T18:34:11.988+0000 I NETWORK [initandlisten] connection accepted from 192.168
.74.203:52463 #20 (20 connections now open)
master_prod | Client #[#XwALSjhrBHLnLhsuAAAC] is now connected.
[]

root@ti5-2015-2016-perf-281de...
worker_prod | Running "sass:dist" (sass) task
worker_prod | Running "less:dist" (less) task
worker_prod | Running "ishint:all" (ishint) task
worker_prod | >> 92 files lint free.
worker_prod | Running "csslint:all" (csslint) task
worker_prod | >> 2 files lint free.
worker_prod | Running "ngAnnotate:production" (ngAnnotate) task
worker_prod | >> 1 file successfully generated.
worker_prod | Running "uglify:production" (uglify) task
worker_prod | >> 1 file created.
worker_prod | Running "cssmin:combine" (cssmin) task
worker_prod | >> 1 file created. 1.22 kB -> 996 B
worker_prod | Running "env:prod" (env) task
worker_prod | Running "mkdir:upload" task
worker_prod | Running "copy:localConfig" (copy) task
worker_prod | Running "concurrent:default" (concurrent) task
worker_prod | Running "nodemon:dev" (nodemon) task
worker_prod | [nodemon] 1.8.1
worker_prod | [nodemon] to restart at any time, enter `rs`
worker_prod | [nodemon] watching: server.js config/**/*.js modules/*/server/**/*.js
worker_prod | [nodemon] starting `node --debug=5859 server.js`
worker_prod | debugger listening on port 5859
worker_prod | Running "watch" task
worker_prod | Waiting...
worker_prod |
worker_prod | + WARNING: It is strongly recommended that you change sessionSecret config while r
unning in production!
worker_prod | Please add `sessionSecret: process.env.SESSION_SECRET || 'super amazing secret'`
to
worker_prod | `config/env/production.js` or `config/env/local.js`
worker_prod | --
worker_prod | MEAN.JS
worker_prod | Environment:          production
worker_prod | Port:                 8443
worker_prod | Database:             mongodb://192.168.74.200/mean
worker_prod | App version:          0.4.1
worker_prod | MEAN.JS version:     0.4.1
worker_prod | --
worker_prod | Client #[#Y4XRBIq6mnIQ7_M9AAAA] is now connected.
worker_prod | Name set to worker1 no active
[]
```

Communication client-serveur

PERF Datasets Performances Rankings S'enregistrer S'authentifier

L'archive comporte 2 projets :

- Un projet avec un exemple complet utilisant Jarowinkler. Ce projet peut servir d'exemple pour la façon d'utiliser notre application PERF.
- Un projet vide avec seulement la classe de connexion à PERF et le minimum de code pour le main.

Pour faire fonctionner le projet, 2 classes sont à modifier/créer :

- La classe `Main.java` : c'est dans cette classe que vous allez choisir l'adresse du serveur avec lequel vous allez communiquer. De même, vous devez dans cette classe choisir le dataset sur lequel vous allez travailler, le débit auquel vous souhaitez recevoir les entités (nombre d'entités par seconde) ainsi que l'identifiant de votre groupe.
- La classe `Alignement.java` : c'est dans cette classe que vous allez devoir mettre en place les algorithmes permettant d'optimiser le traitement des entités.

Important : il est vivement recommandé d'envoyer le résultat pour permettre le bon fonctionnement de la plateforme avec cette commande :

```
connexion
.sendMessageToMaster(new JSONObject()
.put("id", idGrp)
.put("type", "action")
.put("action", "return-matching")
.put("value", VotreResultat));
```

Dans le cas où vous n'envoyez pas de résultat, il faut se déconnecter manuellement `connection.disconnect();`.

- Une fois les traitements effectués et les résultats retournés au serveur, la performance de votre traitement vous est retournée (valeur entre 0 et 1)
- Vous pouvez voir l'ensemble de vos performances en accédant à la page dédiée de l'application.
- Vous pouvez également accéder au classement des performances sur la page de classements. Il vous suffit pour cela de choisir le dataset que vous souhaitez. Le classement vous permet de situer votre meilleure performance par rapport à l'ensemble des autres équipes.
- En cas de problème dans la réalisation de l'une des étapes précédentes, renseignez-vous auprès de votre professeur référent.

Université Claude Bernard Lyon 1

Interface client - administration

DIFFICULTÉS RENCONTRÉES

- Déploiement
 - proxy
 - connexion à MongoDB
 - mise en production (modification des variables d'environnement)
- Interopérabilité Node.js/Java
 - sockets de Node.js trop sécurisées (nécessité de socket ID)
- Calcul de performances sur le master (blocage des autres fonctionnalités)

BILAN

Projet fonctionnel

Déployable à différentes échelles, mais un worker par client conseillé
pour de meilleurs résultats

Administrable par les enseignants via une interface graphique

Interface Java toute faite pour la communication avec le serveur

Propose un classement des résultats obtenus

POURSUITE DE PROJET

Gestion de nouveaux types de sources de données

Mise en place de nouveaux modes : projet, test, ...

Gestion des utilisateurs (étudiants) via LDAP

Algorithme de recherche du meilleur worker

Faire un Sharding et de la réplication de la base MongoDB

IMPRESSIONS PERSONNELLES

Projet complet, proche d'un travail en entreprise

Acquisition de compétences en déploiement d'application, en interopérabilité et dans l'utilisation de MEAN

REMERCIEMENTS

Nous tenons à remercier :

Fabien Rico et Emmanuel Coquery

pour leur aide précieuse

Nicolas Lumineau, Fabien Duchateau et Romuald Thion

pour l'encadrement de ce projet