Projet eCOM
INFO5 (ex RICM5)
2019

version provisoire 18/09/2019

Sybille Caffiau

Didier Donsez

Vincent Zurczak*

Université Grenoble Alpes, Polytech Grenoble & UFR IM2AG

*Linagora

Sommaire

- Objectifs fonctionnels
- Objectifs pédagogiques
- Organisation et planning
- Modalités d'évaluation
- Projet 2019-2020 :
  - Gitlab
  - « Lancement » du groupe
  - Architecture globale
  - Technologies

Objectifs fonctionnels du projet

Développement/déploiement agile d'une application web full-stack

- Sujet 2019-2020 : application de réservation de matériel pour le GUC Voile
- Cadre fixé :
  - Equipe
  - Scénarios
  - Contraintes techniques et technologique : architecture, JHipster
  - Contraintes gestion de projet : versionning gitlab, utilisation des issue, agile, livrables (avec dates)
  - Règles de travail : présence obligatoire dans les salles assignées
- Consignes et sujet dans un projet gitlab (fourni)

Objectifs pédagogiques

Intégration de différentes disciplines

- CAR, IHM, GL
- Utilisation des principes et techniques vues dans les différentes formations
- Approfondissement en options
- Suivi / encadrement

  → Aller chercher les informations auprès des enseignants concernés (CAR / IHM / GL)

Manipuler des technologies de pointe dans le développement d'applications ***hautement disponibles et hautement preformantes*** sur Internet

Gestion agile d'un « Gros » projet

- → Scrum, Kanban, Lean, DevOps

Les grandes étapes du projet

Du 17/09 au 08/10 : Pré-étude

- Formation aux technologies : TP encadrés + travail en autonomie
- Analyse des technologies : choix
- Mise en place des procédures

Du 08/10 au 05/11 : Sprint 1

Du 12/11 au 03/12 : Sprint 2

Du 03/12 au 17/12 : Sprint 3 Fin de projet

Du 17/09 au 08/10 : Pré-étude

Soutenance 1 : pré étude/mise en place

- Démo : Version minimale (vers data et dans l'autre sens) déployé dans une VM Azure
- Analyse choix techno
- Démarche d'itération
- Démarche qualité : qu'est ce que vous mettez en place pour garantir la qualité ?
- Mise en place procédure "gestion de projet" pour les jalons (toutes les 3 semaines)
- Analyse scénarios pour définir les US
- Modele de BD
- Modèle de tâches + contraintes contexte

Les grandes étapes du projet

Du 17/09 au 08/10 : Pré-étude

Du 08/10 au 05/11 : Sprint 1

- Formation à la revue de code
- Réalisation d'un premier sprint
- Développement suivant les procédures pour livraison continue

Du 12/11 au 03/12 : Sprint 2

Du 03/12 au 17/12 : Sprint 3 Fin de projet

Du 08/10 au 05/11 : Sprint 1

Audit de code croisé

Audit client 1 (10 min par groupe avec Démo)

Les grandes étapes du projet

Du 17/09 au 08/10 : Pré-étude

Du 08/10 au 05/11 : Sprint 1

Du 12/11 au 03/12 : Sprint 2

Du 03/12 au 17/12 : Sprint 3 Fin de projet

Du 12/11 au 03/12 : Sprint 2

Audit client 2 (10 min par groupe avec Démo)

Les grandes étapes du projet

Du 17/09 au 08/10 : Pré-étude

Du 08/10 au 05/11 : Sprint 1

Du 12/11 au 03/12 : Sprint 2

Du 03/12 au 17/12 : Sprint 3 Fin de projet

Soutenance finale

Réalisation (démo)

Résumé

Prise de recul

Modalités d'évaluation (1)

Cf Grille d'évaluation

Note attribuée à un groupe :

- Présentation du travail (/2)
- Développement (/7)
- Travail en groupe (/3)
- Déploiement (/3)
- Prise en compte de l'utilisateur (/3)
- Démarche de production (/2)

Bonus et malus attribué collectivement ou individuellement

Modalités d'évaluation (2)

Notes qui s'appuient sur :

- Les observations des enseignants
- Les soutenances
- Les analyses
- Le code
- Les démos
- Les outils de gestion de projet (CR, backlog, versionning du code...)

Régles :

- Ce qui n'est pas accessible à partir de votre dépôt git n'est pas connu des enseignants (=> 0)
- La livraison continue est une OBLIGATION

Projet eCOM
INFO5 (ex RICM5)
2019

version provisoire 18/09/2019

Le projet 2019-2020

Le dépôt Gitlab

Le projet 2019-2020-ECOM-INFO5-GXX avec les branches

- Ressources : documents fournis par les enseignants (consignes, ressources pédagogiques...)
- Docs : les documents que vous produisez
- Sandbox : la branche pour vos essais de technologie
- Dev : la branche de développement (à vous d'organiser !)

La politique des issues

- Issues utilisées pour la communication interne et externe (vers les enseignants et les « clients »)
- Chaque issue porte au moins 2 tags :
  - le domaine (client, user, ops (déploiement, CI, etc), dev-front...)
  - l'objectif de l'issue (ex : question, absence)

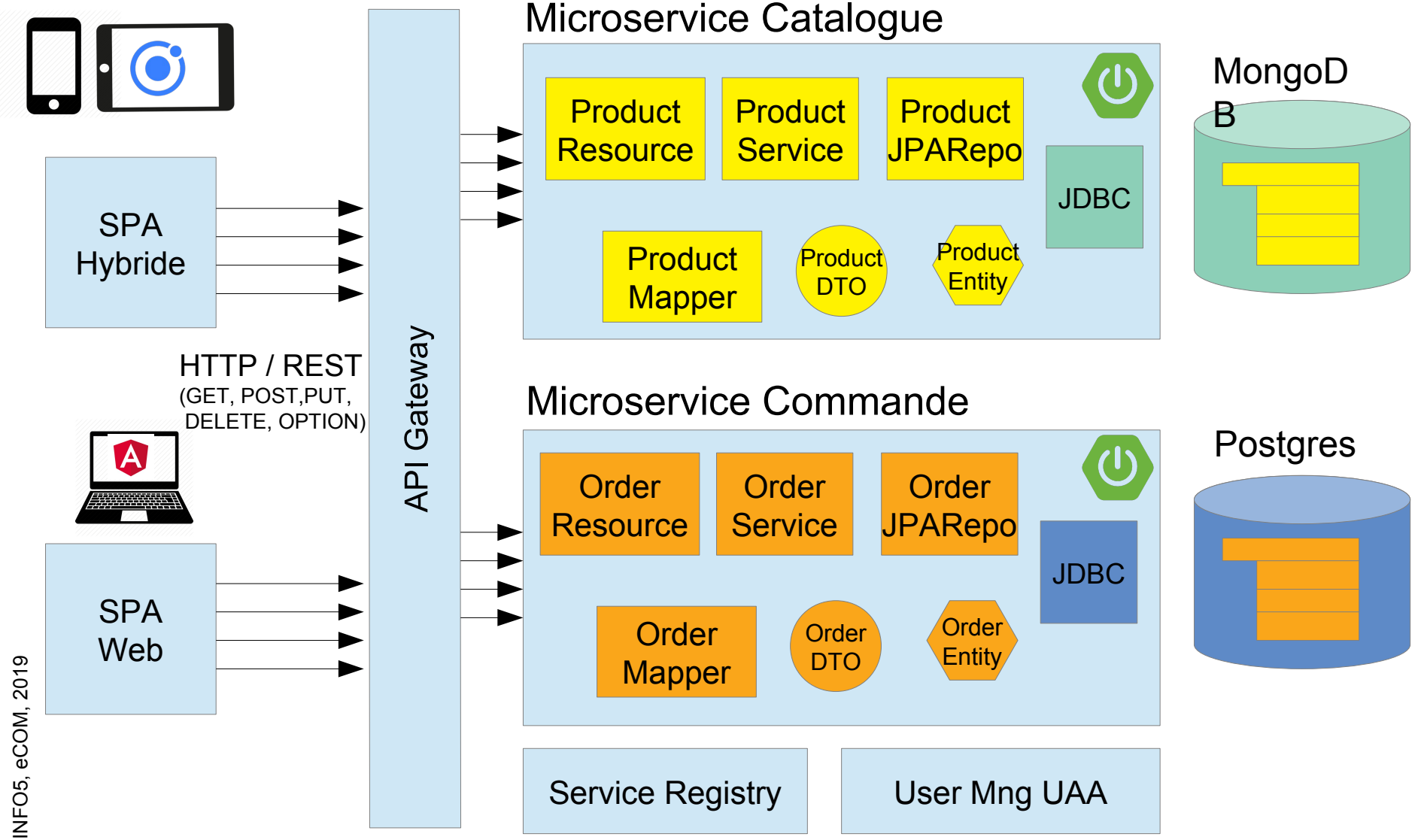
Les premières décisions pour le groupe

Règles de vie du groupe

Organisation

- Elire un chef de projet et un scrum master identifiés du début à la fin
  - Contact avec les enseignants
  - Contact avec le client
- Attribuer des rôles/spécialités

Architecture Microservice
1 uS par (grande) fonction



Environnements/intergiciels Supports (i)

JHipster

- Backend Spring Boot
- Bases de données
  - Dev : H2
  - Prod : Postgres, MySQL, MySQL répliqué
- Frontend SPA et/ou Hybride
- Docker et consors (k8s, ...)
- Hébergement Cloud
  - IaaS :Windows Azure, AWS, Google, Bluemix, Heroku
  - OVH (2,99 euros/mois)
  - Instances « Small » Linux gratuite
  - Déploiement multi-compte (5 à 6comptes)
  - (Ne commitez pas les crédentials sur un dépôt public)

Architecture minimale
Les constituants à **générer** à l'étape 1

Le modèle de données

- base de données relationnelle générée
  - 3 Tables Category, Product, *User*
  - + Tables optionnelles : ExtraUser, Customer, Order, OrderLine, Payment

Le Backend Spring: Monolithe RESTFul (générés avec JHipster)

- 3 Entities + Relationship + JPARepository
- 3 DTOs + mappers
- 3 Resources+ 3 Services
- + services additionnels (MailService, ...)
- API OpenAPI (Swagger)

2 Frontends (générés avec JHipster)

- 1 SPA Web : AngularX, React, VueJS au choix
- Mobile hybride : Ionic + plugins Cordova (plateformes iOS et Android)

Un client CLI basé sur cURL

- Généré avec Swagger Codegen

Architecture globale
Les constituants à développer à l'étape 2

Implémenter la logique métier dans les services

- Par rapport aux ROLE des jhi_user

IHM « responsive » ou IHM mobile first

- en fonction des besoins clients
- Canevas SPA (Angular X, …), Ionic + Plugins

Taches périodiques

- Campagne de mailing
- Envoi de rappel par mail

Génération de PDF (voir pdfbox.apache.org)

- Factures, Dossards, ...

Architecture globale
Les constituants à développer à l'étape 2

Optimisation des transferts des LOB / Collections

- SPA ← → Backend
- Backend ← → Base de données
- LocalStorageService

Single-Sign On

- keycloak

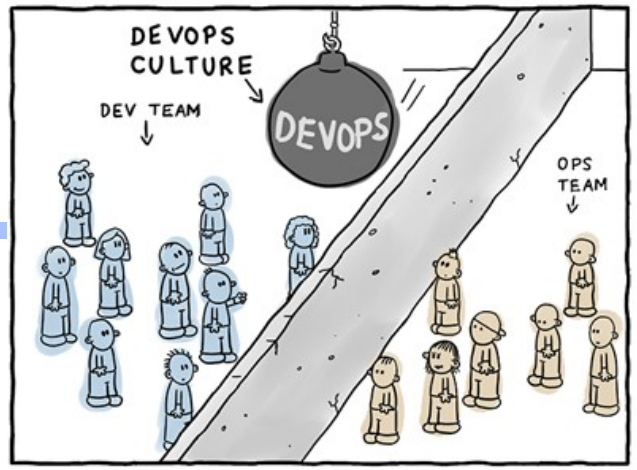
Services externes

- Stripe.com (API de paiement) : utiliser le mode « test »
- Gravatar, Social logins

Séparation en 2 « microservices »

- API Gateway
- 1 seule base de données (répliquée)

DevOps



Dev(elopment) – Op(eration)s

Objectifs

- TTM, MVP, Rapid Production Deployment

Moyens

- Continuous Integration (CI)
- Continuous Delivery (CD)
- Apllication Performance Monitoring (APM)

Environnements/intergiciels Supports (ii)

Sécurité

- Nginx en SSL Terminaison + Certificat Let's Encrypt
  - Requiert un nom de domaine pour la/les VMs
- IpTables, UFW
- Fail2Ban
- Pen testing

Monitoring des instances de VM

- Prometheus, Grafana

OAuth2

- Keycloak, OKTA
- Social Login

Exemple de rançon pour une base Mongo non protégée

$ db.PLEASE_READ_ME.find()

{ "_id" : ObjectId("58a7287db7dc324adb249fdf"), "info" : "Don't panic. Your DB is in safety and backed up (check logs). To restore send 0.1 BTC and email with your server ip or domain name. Each 48 hours we erase all the data.", "amount" : "0.1 BTC", "data_we_have" : { "local" : ["startup_log"], "first_database" : ["users", "preferences"], "MyAppXXX" : ["emails"] }, "Bitcoin Address" : "1NSz9TRBGKHKFdjdjH2Gme3LwDi5", "email" : "xxxxx@xxxx.org" }

Environnements/intergiciels Supports (ii)

DevOps

- Docker, k8s (rolling update)

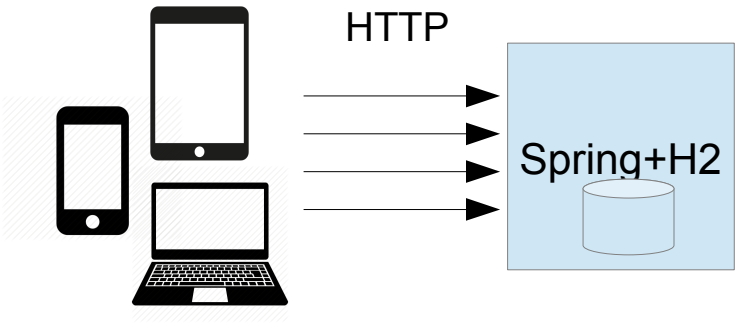
Haute disponibilité

- Nginx en Load Balancer
- Spring en cluster
- Postgres ou MySQL en replication
- Backup périodique de la base

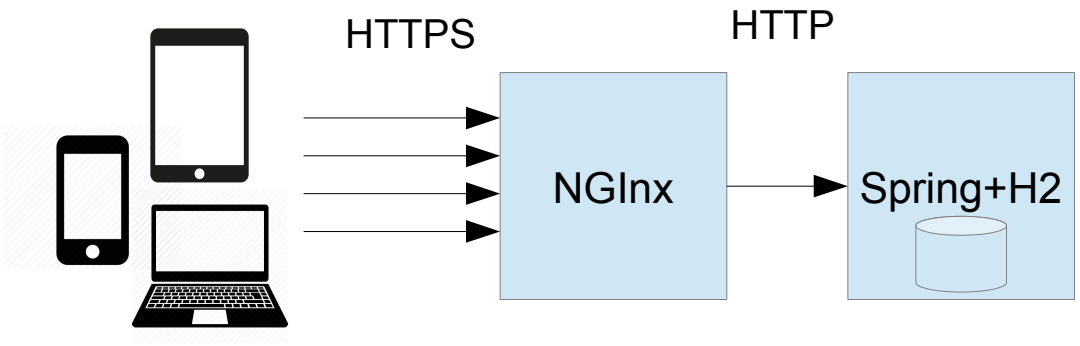
Haute performance

- Elasticité horizontale k8s (scale up/down)

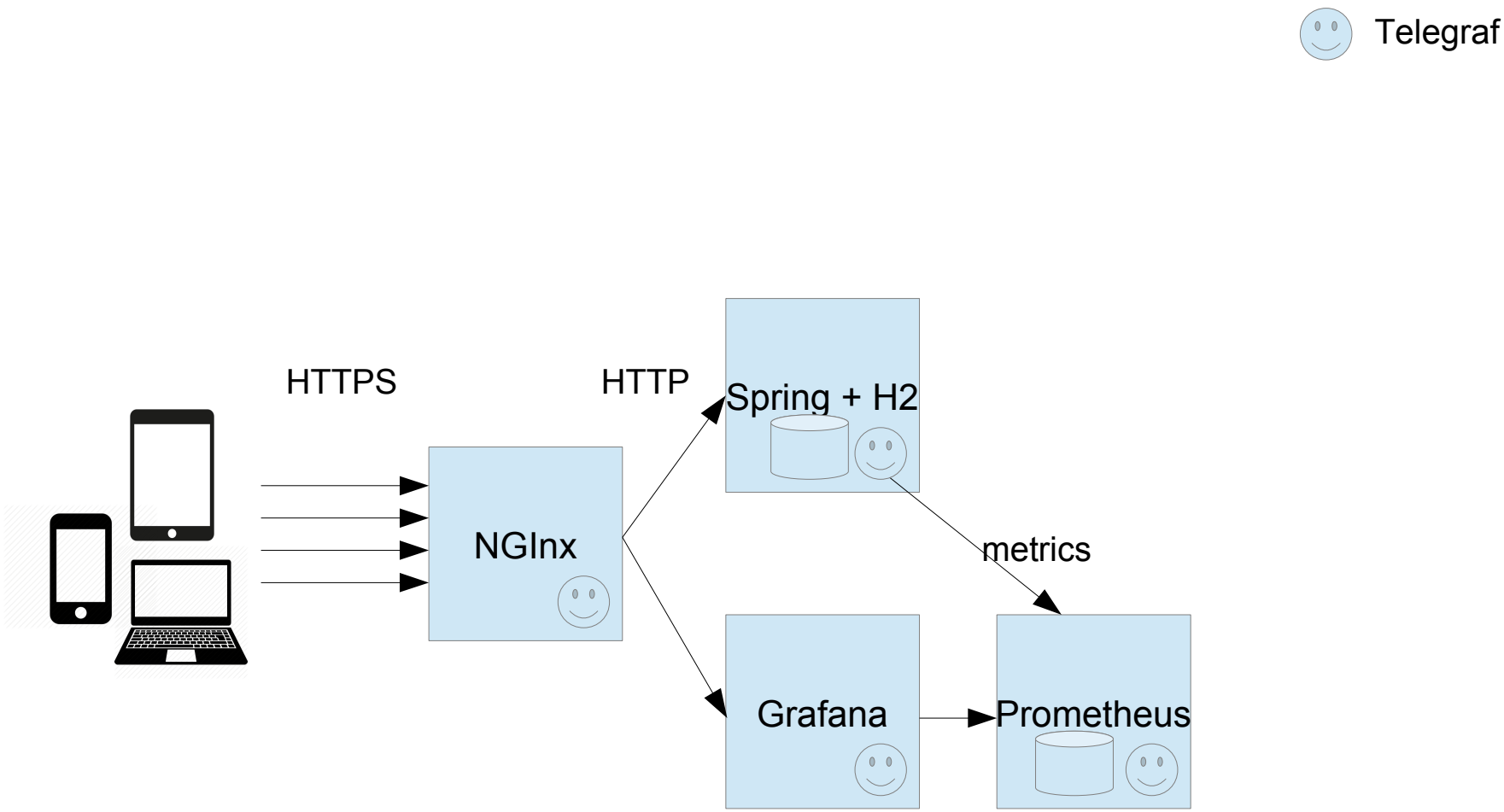
Etapes de développement (i)



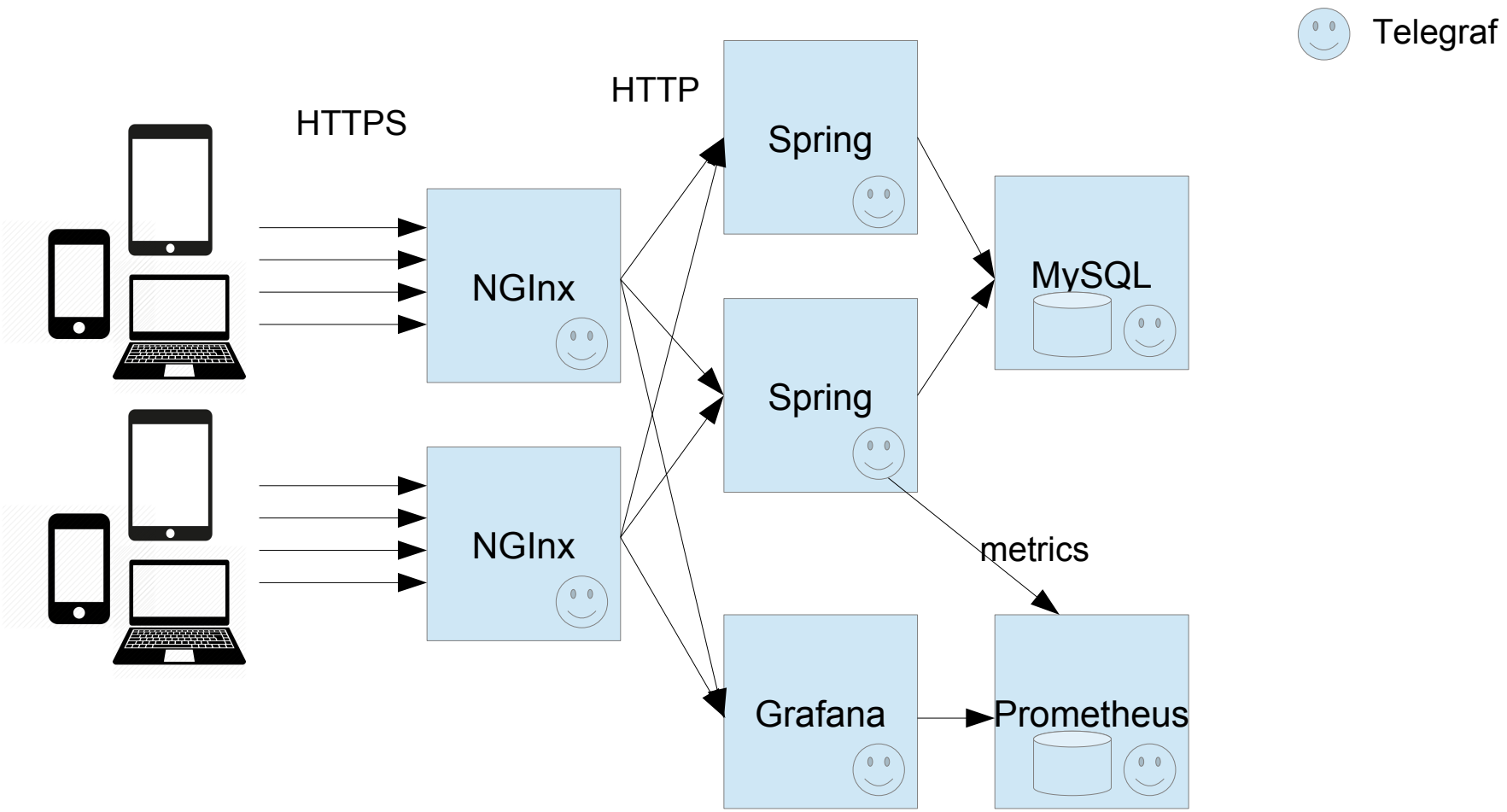
Etapes de développement (ii)



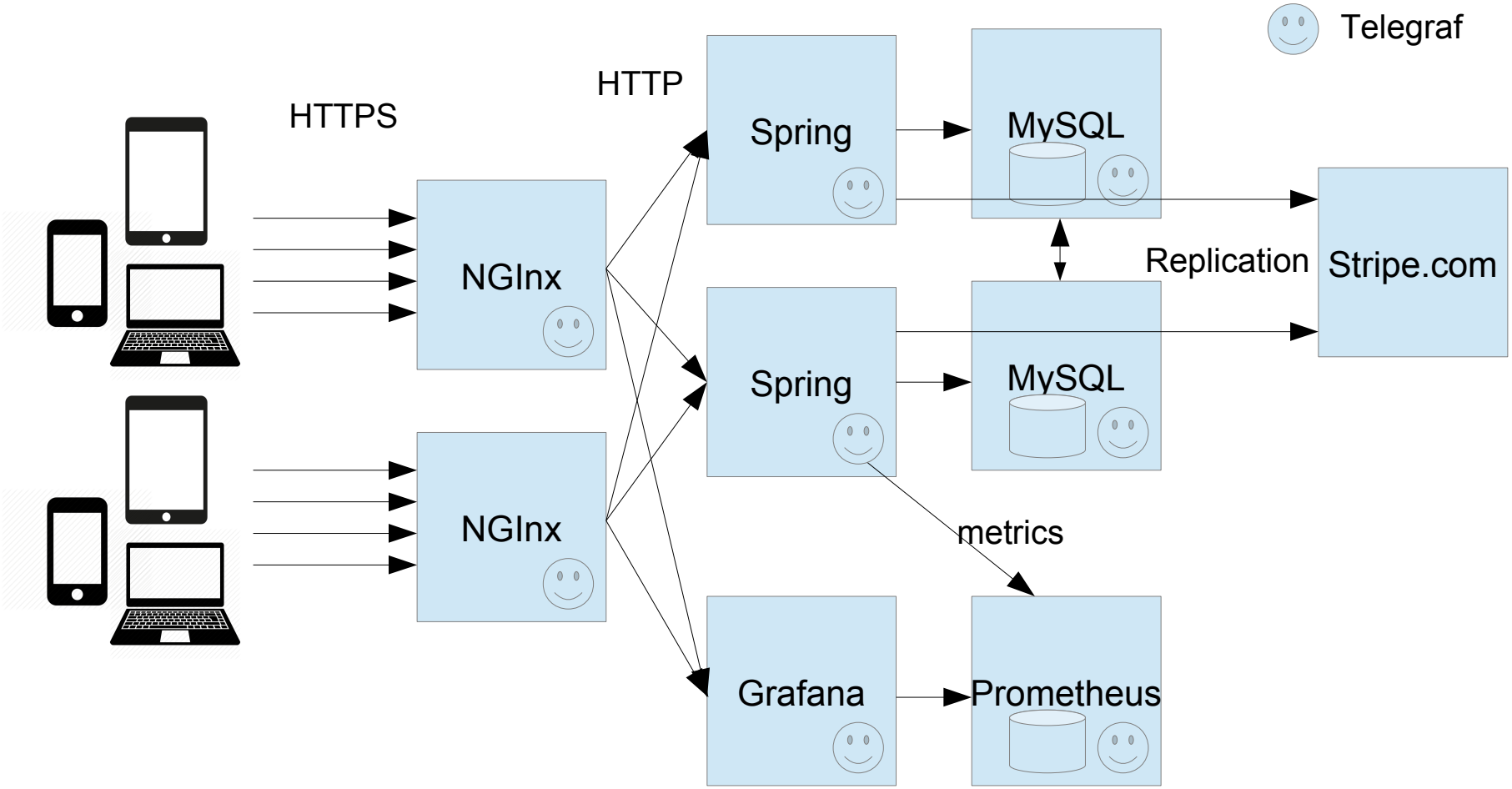
Etapes de développement (iii)



Etapes de développement (iv)



Etapes de développement (v)



CD-CI
Intégration en Continue

Test unitaire

Test d'intégration

Test e2e

Test de charge

- Peupler la base via des CSV conséquents*
  - src/main/resource/config/liquibase/fake-data/*.csv

Plusieurs milliers d'entrées ayant du sens

Test de résilience

- Panne d'une machine jusqu'à datacenter outage

Test de pénétration

Software Quality Assurance

Automatisation

- Jenkins, Travis CI, Circle CI ...

CD-CI
Déploiement

Profil DEV

- Sur votre localhost

Profil PROD

- Sur une machine virtuelles (Azure, AWS, …) gratuites

Profil PROD HA

- Des machines virtuelles Linux (multi comptes)

CD-CI
Livraison en Continue

Principe « Minimal Viable Product »

Service immédiatement en production

- Après un staging « court »

Blue – Green Deployment

- Rolling Update
- Fast rollback

En option : Elasticité horizontale (scaling)

- Ajout et Retrait de VM en cours d'exécution

→ scripts avec awscli, ssh, curl (Swagger), git, Docker, Docker Compose, mini Kube ...

Qualité du logiciel produit

- IDE
  - Eclipse JEE ou NetBeans JEE ou JetBrains
  - Plugins Azure, AWS, ...
- Builder
  - maven 3, npm, yarn, webpack ...
- Forge (Gitlab ...)
  - Intègre Git, Jenkins, Sonar …
- Dépôt d'artéfacts (pom, docker images, js lib, ...)
  - Artifactory CE ...
- Software Quality Assurance
  - Rapport SonarQube (voir conteneur Docker généré)
  - Code review

Resources

JhipsterConf 2019

- https://www.youtube.com/playlist?list=PL6IFaLdAcgE3aSg
  https://www.youtube.com/watch?v=LfpV34seu10&list=PL6I