Projet SNCF

Laurent Morin

Janvier 2020

1 Cahier des charges

Nous souhaitons réaliser un site web pour afficher la liste des TGV disponible classés par ordre croissant de date de départ.

Un TGV est identifié par un numéro (par exemple 6440) et possède une ville de départ et une ville d'arrivée, un horaire de départ et une durée (et par conséquent une date d'arrivée).

Ce projet devra être réalisé avec la technologie php/mysql. Le serveur web devra proposer $php > 5.6$ et la base de données devra être idéalement a la version mysql 5.7.

Nous nous situons plutôt dans le contexte des devops ou nous devons proposer la plateforme suivante :

1.1 Sur poste windows :

installer easyphp ou wamp ainsi que visual studio code (ou tout IDE qui vous convient). A chaque fois ou on juge que notre travail est "stable", nous devons faire un push sur la branche git dev de notre repository git.

Notre pipeline est composé de deux stages : dev et prod

1.2 Dans le stage dev :

nous nous limitons à un serveur linux contenant un httpd/php et mysql. Ce serveur doit toujours proposer toujours la version la plus à jours sur la branche dev.

1.3 Dans le stage prod :

- Dans ce stage, nous devons proposer une architecture hautement disponible qui doit être composée d'un minimum de deux clones des composants informatiques pour assurer la tolérance aux pannes
- Dans ce stage, nous devons déployer la dernière version de la branche master de git sur la plateforme du stage prod.
- Prévoir une solution pour que votre infrastructure soit extensible, i.e, dans le cas ou on se rend compte qu'il y aurait une surchage des serveurs, nous devons mettre à échelle (scaling) notre architecture en un temps relativement court.

1.4 Votre mission :

- Concevoir le diagramme UML de l'application
- Concevoir et implémenter la base de données
- Insérer un jeu de données des tests
- exporter le schéma (et uniquement le schéma) dans un fichier sql
- développer un fichier html statique qui affiche un message de bienvenu
- Créer un repository git et faites un "push" initial des fichiers html et sql
- La suite des tâches dépendent de votre conception de la solution conçue

2 Conception UML

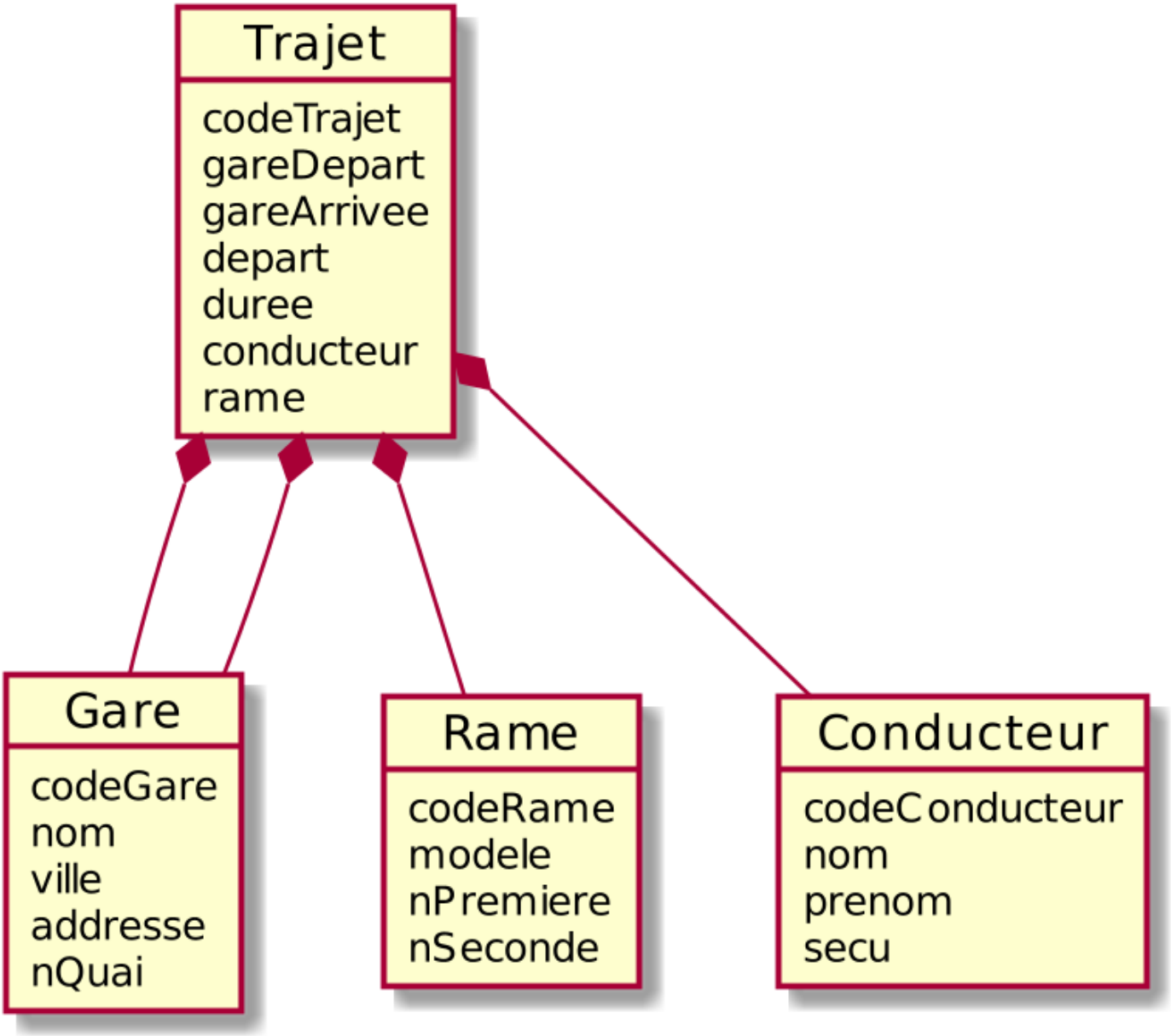
2.1 Liste des objets

- Gare : code gare, composé d'un nom, d'une addresse, d'une ville, d'un nombre de quai
- Rame : composé d'un numéro,d'une modèle, d'un nombre de place 1er, d'un nombre de place 2ème
- Conducteur : composé d'un Nom, prénom, No secu, No d'identifiant
- Trajet : composé d'un numero, d'une gare de départ, d'une gare d'arrivée, d'un train, d'un conducteur, date de depart, d'une duréé de trajets

2.2 Relation

- Un trajet est composé d'une Rame conduit par un conducteur, depuis une gare de depart à l'heure de départ vers une gare d'arrivé pendant une durée de trajet

2.3 Graph uml



3 Conception SQL

3.1 Script SQL

```
CREATE TABLE IF NOT EXISTS gare (
    codegare int NOT NULL AUTO_INCREMENT PRIMARY KEY ,
    nom varchar(50) NOT NULL,
    ville varchar(50) NOT NULL,
    addresse varchar(250),
    nquai int
);

CREATE TABLE IF NOT EXISTS rame(
    coderame int NOT NULL AUTO_INCREMENT PRIMARY KEY ,
    modele VARCHAR(10) NOT NULL,
    npremiere int,
    nseconde int
);

CREATE TABLE IF NOT EXISTS conducteur(
    codeconducteur int NOT NULL AUTO_INCREMENT PRIMARY KEY ,
    nom varchar(50) NOT NULL,
    prenom varchar(50),
    secu varchar(13)
);


CREATE TABLE IF NOT EXISTS trajet(
    codetrajet int NOT NULL AUTO_INCREMENT PRIMARY KEY ,
    garedepart int NOT NULL FOREIGN KEY references gare(codegare),
    garearrivee int NOT NULL FOREIGN KEY references gare(codegare),
    conducteur int NOT NULL FOREIGN KEY references conducteur(codeconducteur),
    rame int NOT NULL FOREIGN KEY references rame(coderame),
    depart date NOT NULL,
    duree TIME
);
```

4 Conception Devops

4.1 Gestion de versions

Tous les ficheirs sources et de configuration sont versionné avec git en utilisant github :
`https://github.com/RaoulChartreuse/exo_SNCF`

4.2 Machine de production

Compte tenue des besoins de disponibilité et de mise à l'echelle (scalling), la conteneurisation retenue est *kubernetes* qui permet justement de géner le nombre de réplication et de garentir celui-ci.

Compte tenue de l'environement du projet, nous utiliseront *minikube* et simulerons un ensemble de serveur sur une machine virtuelle de dévellopement.

Liste des pods :

- web : un pod de serveurs appache/php
- db : un pod de serveurs mysql

4.3 Machine de Dévellopement

Idéalement l'environement de test doit identique à m'environement de production. Ici en l'abscence de machine de test et avec des resources limité à un seul ordinateur, nous utiliseront pour le dévellopement des serveurs dockers obtenue grace à un docker compose.

Liste des serveurs :

- web : un docker appache/php
- db : un docker mysql

4.4 Deploymenent et integration