



Projet 4A

Développement d'un clone de Twitter

Thibault REY, Frédéric WAGNER



UNIVERSITÉ
DE LORRAINE

LORRAINE
INP

18 janvier 2023



Sommaire

Introduction	3
Présentation des framework	3
1) Spring boot	3
2) Thymeleaf	5
3) Tailwind CSS	5
Architecture du site	6
1) Barres latérales	6
2) Pages de connexion	7
3) Page de profil utilisateur	8
4) Messagerie	10
5) Page d'un twot	12
6) Page d'accueil	13
Conclusion	14



INTRODUCTION

Avant de commencer ce projet de développement web, nous avons cherché un sujet intéressant sur lequel travailler. Nous sommes tous les deux des utilisateurs quotidiens de réseaux sociaux et plus particulièrement de Twitter et nous nous sommes rendu compte que cela représentait un challenge intéressant d'essayer de recréer Twitter de nous-mêmes. Twitter répond aux exigences du projet puisque c'est un site web utilisant beaucoup de base de données. Nous avons donc décidé de garder cette idée et d'en développer une copie que nous avons appelée Twotteur, pour des raisons de droit d'auteur.

PRÉSENTATION DES FRAMEWORK

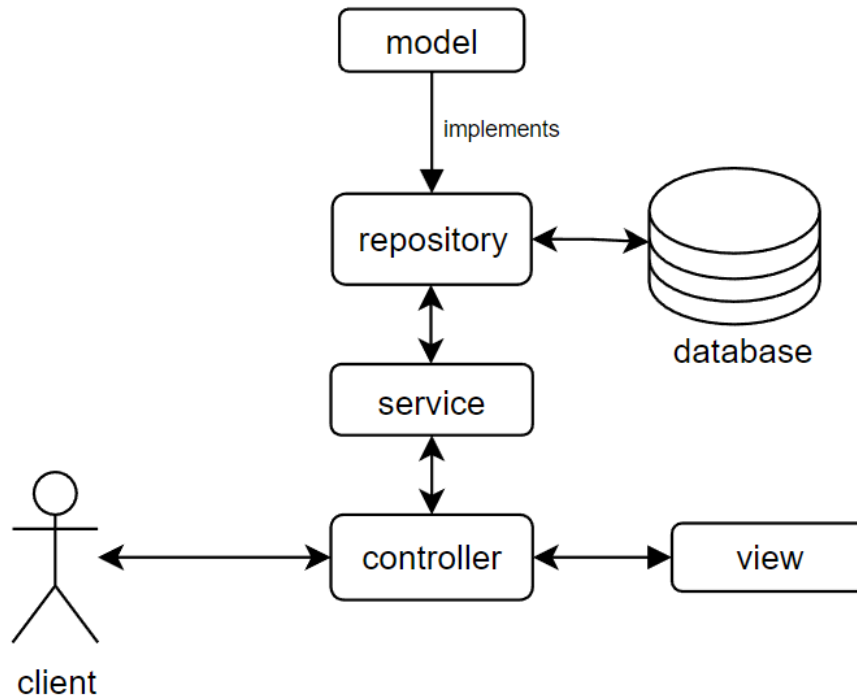
1) Spring boot

Nous avons utilisé Spring Boot pour le backend du site. C'est un framework basé sur le modèle MVC (Modèle-Vue-Contrôleur) très utilisé de nos jours.

L'avantage de Spring Boot comparé à d'autres frameworks MVC comme Laravel par exemple c'est qu'il est basé sur Java, qui est un langage bien plus répandu dans le milieu professionnel que PHP, c'est pourquoi nous avons choisi de l'utiliser.



Pour détailler l'architecture MVC utilisée par Spring Boot, voici un schéma explicatif :



Lorsqu'un utilisateur souhaite accéder à une page, il envoie une requête au serveur. Si l'URL est reconnue par un des contrôleurs (controller), ce dernier va appeler un des services (service). Ce dernier va appeler un des répertoires (repository), qui eux-mêmes effectuent des requêtes à la base de données (database), en MySQL dans notre cas. Chaque répertoire implémente un modèle (model) qui représente une table de la base de données.

Ce qui est important ici est de pouvoir manipuler les tables sous forme d'objets. L'intérêt de faire appel à un service entre le contrôleur et le répertoire est de pouvoir réorganiser les données avant de les transmettre au contrôleur, ce qui permet d'avoir un code plus lisible dans le contrôleur. Ce dernier sert quant à lui plutôt à déterminer quelles données demander aux services selon le niveau de privilège d'un utilisateur par exemple.



Une fois les données récupérées, le contrôleur les inclus dans une vue (view) et renvoie la page souhaitée à l'utilisateur.

Un autre avantage de Spring Boot est la possibilité d'ajouter des dépendances au projet, ce qui facilite énormément l'ajout de fonctionnalités et donc l'évolutivité du projet. Nous avons notamment utilisé la dépendance Spring Websocket qui permet d'utiliser un serveur websocket en parallèle du site.

2) Thymeleaf

Thymeleaf est le ViewResolver de base utilisé par Spring Boot. Il permet notamment d'effectuer des boucles, de tester des conditions, ou encore d'afficher les variables renvoyées par les contrôleurs. C'est un outil indispensable pour afficher des pages dynamiques qui doivent être différentes selon le type d'utilisateur.

Un des inconvénients de Thymeleaf est qu'il ne permet pas d'exécuter du code Java comme le permettent les fichiers en .jsp. Cependant, cela ne nous a pas posé problème puisqu'il est possible d'utiliser du Javascript en parallèle.

3) Tailwind CSS

Nous avons utilisé Tailwind CSS pour le frontend du site. C'est un framework utilisé pour faciliter l'implémentation de style directement dans le code HTML. Il utilise pour cela des classes nommées de façon logique par rapport à du code CSS correspondant. Par exemple, ajouter la classe text-white rendra le texte de l'élément HTML correspondant blanc.

C'est un outil qui permet un gain de temps important puisqu'il permet de ne pas avoir à quitter la page HTML que l'on veut modifier tout en apportant une plus grande efficacité au niveau du code puisque les noms des classes sont abrégés.

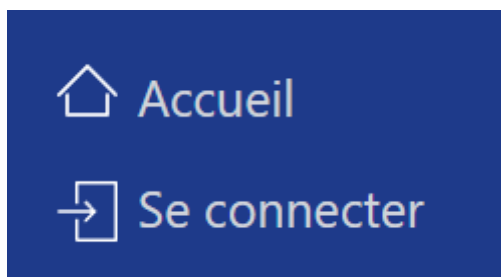
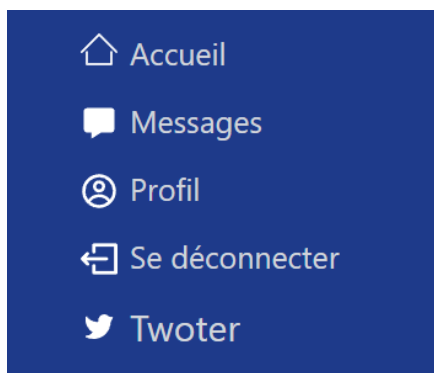


Tailwind possède également des classes qui ne s'appliquent que sur certaines tailles d'écran, ce qui nous a permis de rendre le site responsive assez facilement.

ARCHITECTURE DU SITE

1) Barres latérales

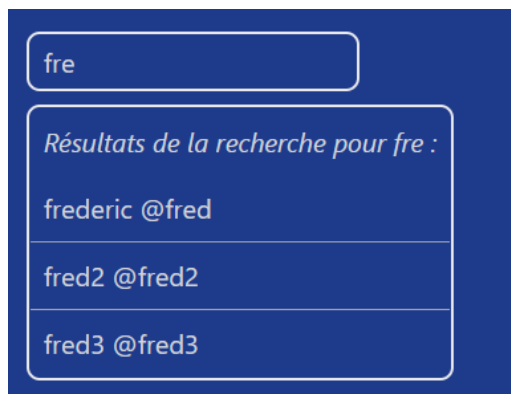
Le site possède une barre latérale gauche et une barre latérale droite qui sont disponibles sur toutes les pages du site. La barre latérale gauche sert à naviguer entre les pages du site, elle se présente sous deux versions, selon si l'utilisateur est connecté ou non (respectivement image de gauche et de droite ci-dessous) :



En mode mobile, cette barre latérale gauche se place en dessous de la page et seules les icônes restent visibles, ce qui permet de gagner en visibilité (cf image ci-dessous) :



La barre latérale droite est une barre de recherche. Elle sert à trouver la page de profil d'une personne avec son nom d'utilisateur (cf image ci-dessous) :

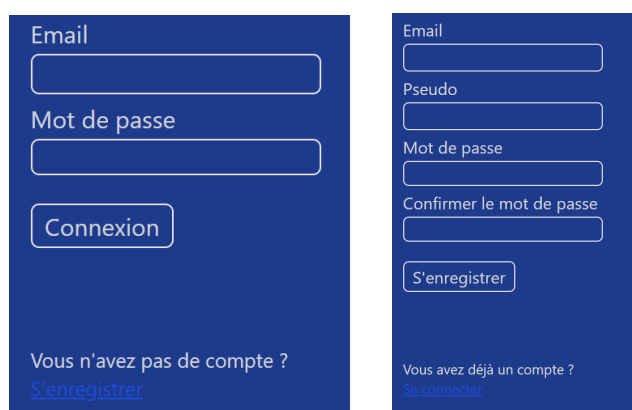


Cette barre de recherche n'est pas visible sur le mode mobile afin d'éviter de saturer la page.

Cette dernière fonctionne en Javascript. Chaque fois que l'utilisateur modifie la recherche d'une lettre, un code Javascript envoie une requête à un rest controller qui répond avec la liste des utilisateurs correspondants et actualise la boîte de résultats en conséquence.

2) Pages de connexion

Le site possède une page de connexion et une page pour enregistrer un compte (respectivement l'image de gauche et de droite ci-dessous) :



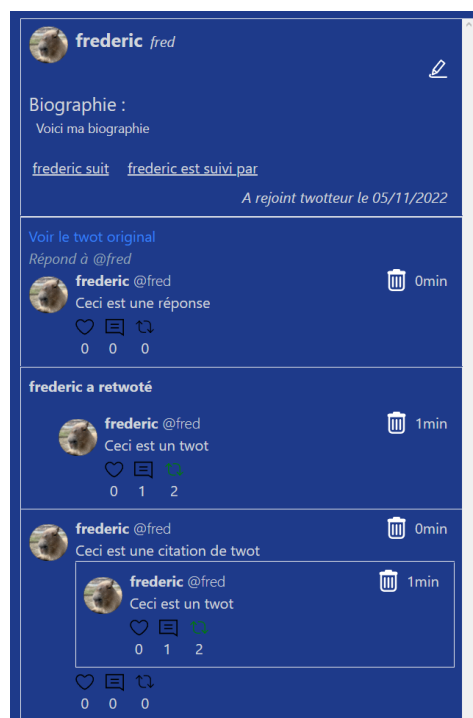
Le service associé à ces fonctionnalités vérifie que l'adresse mail et le pseudo sont uniques avant d'autoriser la création du compte.



A des fins de sécurité, le mot de passe devrait être stocké hashé, ce n'est cependant pas le cas car le site n'a pas pour but d'être hébergé et disponible au public ; cela reste donc plus pratique dans le cadre du développement de laisser les mots de passe en clair dans la base de données.

3) Page de profil utilisateur

Chaque utilisateur a une page de profil, elle se présente comme ceci :




En haut, on retrouve le pseudonyme, le nom d'utilisateur, la photo de profil et la biographie.

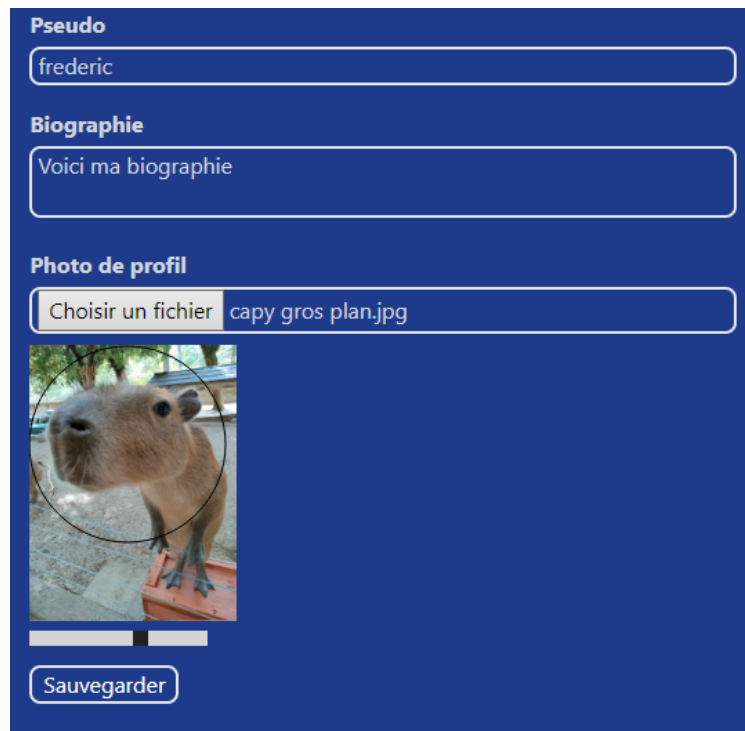
On retrouve ensuite deux liens cliquables qui permettent d'observer la liste des gens suivis par l'utilisateur et la liste des gens qui le suivent. On retrouve aussi la date de création du compte.



Lors du chargement initial de la page, uniquement les informations ci-dessus sont chargées. En dessous, on peut voir la liste des twots postés par l'utilisateur. Ces twots ne sont initialement pas chargés, c'est un script client qui effectue les requêtes pour les charger par bloc de 10.

L'objectif de cela est de réduire le temps de chargement de la page. En effet, si un utilisateur a posté 1000 twots et qu'on tente de les charger d'un coup, la page risque de prendre au moins une minute à charger. Au contraire, en chargeant les twots 10 par 10, on obtient un temps de rendu bien inférieur et on évite de charger du contenu inutile.

Si l'utilisateur visite sa propre page, il peut cliquer sur le bouton d'édition  ce qui le ramène vers la page d'édition suivante :



The screenshot shows a user profile editing interface on a dark blue background. It contains the following elements:

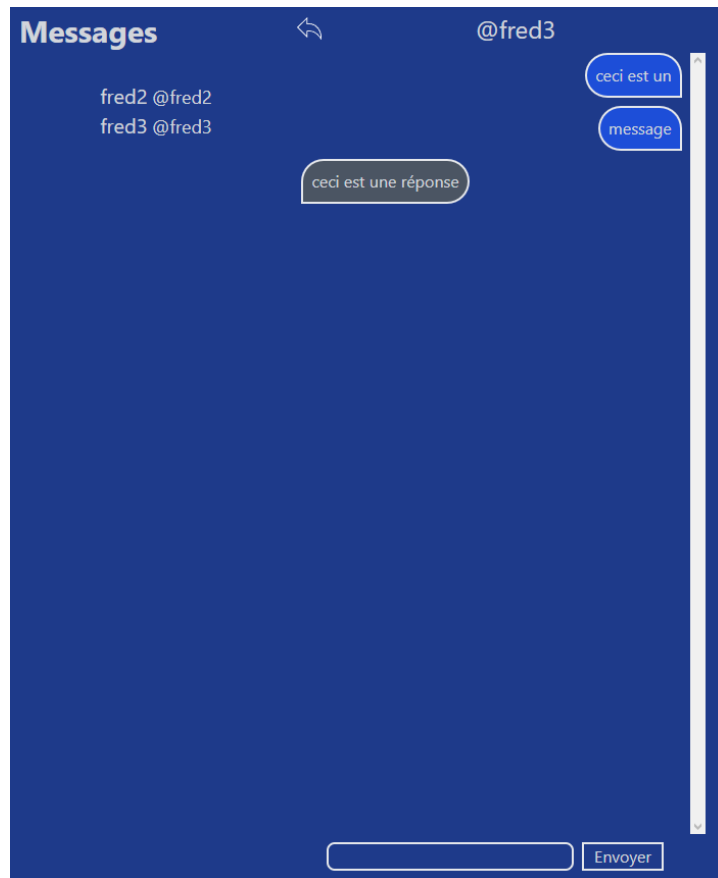
- Pseudo**: A text input field containing the name "frederic".
- Biographie**: A text area containing the text "Voici ma biographie".
- Photo de profil**: A section with a file selection button labeled "Choisir un fichier" and a file name "capy gros plan.jpg". Below this is a preview of a capybara's face with a black circular crop overlay. A small horizontal slider is positioned below the image.
- Sauvegarder**: A button at the bottom of the form.

L'utilisateur a la possibilité de modifier son pseudonyme ou sa biographie ainsi qu'uploader une photo de profil et la recadrer (cercle noir sur l'image).



4) Messagerie

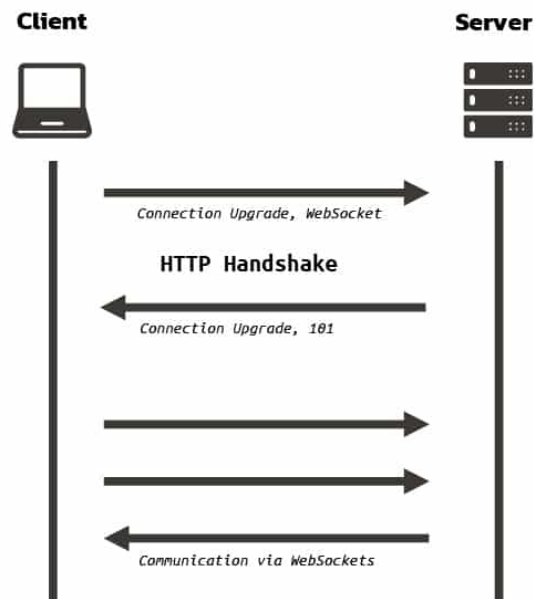
Le site permet à deux utilisateurs qui se suivent mutuellement de s'échanger des messages en temps réel. L'interface se présente comme ceci :



L'utilisateur sélectionne le contact avec qui il souhaite échanger à gauche (dans la colonne de gauche, ici il y a 2 possibilités fred2 ou fred3) et peut voir l'historique de ses messages à droite.



Pour permettre une communication en temps réel, on utilise un serveur websocket, hébergé par Spring Boot grâce à une dépendance. Un websocket est un protocole de communication qui permet d'établir un canal de communication entre un hôte et le serveur. Ce canal est établi par un échange HTTP initial, comme expliqué dans le schéma suivant :

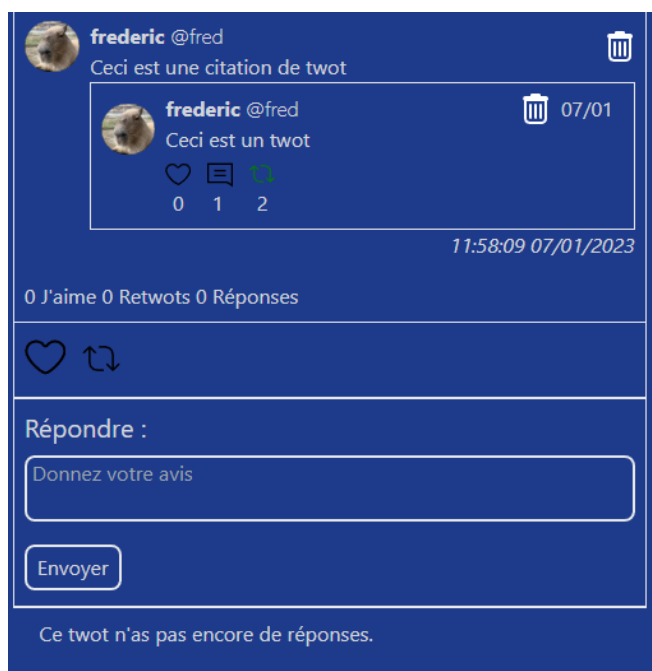



En parallèle de la communication en websocket, les messages sont stockés dans la base de données pour pouvoir afficher l'historique de la conversation.



5) Page d'un twot

Chaque twot publié par un utilisateur a une page qui lui est dédiée. Elle se présente comme ceci :



Cette page permet de consulter les réponses au twot initial, sa date de publication, son nombre de “j’aime” et de retwots. L'utilisateur qui a publié le twot peut également le supprimer en cliquant sur l'icône .



6) Page d'accueil

Pour finir cette liste de fonctionnalités, voici la page d'accueil du site, lorsque l'utilisateur est connecté :



L'utilisateur peut ici voir tous les twots et les retwots des comptes qu'il suit.

Il a également la possibilité de publier des twots depuis le formulaire en haut de la page.

Les twots sont classés par ordre chronologique et chargés par bloc de 10 via du code javascript pour permettre un rendu plus rapide.



CONCLUSION

Pour conclure ce projet, nous sommes assez fiers de l'application qui est relativement ressemblante à Twitter. En effet, la majorité des possibilités que propose Twitter sont présentes également dans notre projet et l'application fonctionne rapidement et sans bugs apparents.

Il y a cependant quelques points que nous n'avons pas eu le temps d'implémenter et qui sont manquants pour obtenir une application encore plus ressemblante à Twitter. Voici une liste des points non exhaustive de ce qu'il manque à l'application et donc des potentiels axes d'améliorations :

- Ajouter une fonction recherche sur mobile en la mettant éventuellement sur une autre page pour ne pas surcharger la page
- Ajouter la possibilité de twotter des images aussi bien sur pc que sur mobile
- Améliorer l'algorithme de recommandation de la page d'accueil pour qu'il propose des twots potentiellement intéressants pour l'utilisateur et notamment des twots d'autres utilisateurs que ceux qu'il suit
- Ajouter un bouton pour supprimer son compte
- Hasher les mots de passe pour augmenter la sécurité de l'application