Rapport de Développement —

Breezy

Équipe

* Jules BESSON
* Hugo CELLIER
* Baptiste TRONEL-PEROZ
* Théo VERSEPUECH

Table des matières

[0. Introduction 1](#_Toc201614637)

[1. Conception 1](#_Toc201614638)

[1.1 Objectifs fonctionnels 1](#_Toc201614639)

[1.2 Cahier des charges technique 2](#_Toc201614640)

[1.3 Réflexion graphique 2](#_Toc201614641)

[2. Workflow 3](#_Toc201614642)

[2.1 Organisation de l’équipe 3](#_Toc201614643)

[2.2 Outils utilisés 3](#_Toc201614644)

[2.3 Méthodologie 4](#_Toc201614645)

[3. Architecture 4](#_Toc201614646)

[3.1 Architecture générale 4](#_Toc201614647)

[3.2 Schéma technique 4](#_Toc201614648)

[3.3 Sécurité 4](#_Toc201614649)

[4. Résultat du développement 4](#_Toc201614650)

[4.1 Fonctionnalités terminées 4](#_Toc201614651)

[4.2 Capture d’écran (à insérer) 4](#_Toc201614652)

[4.3 Limites connues 4](#_Toc201614653)

[4.4 Perspectives d’évolution 4](#_Toc201614654)

# 0. Introduction

Il nous a été demandé, durant les quatre semaines consacrées au développement web avancé, de concevoir une application web de type réseau social, inspirée de Twitter.  
Ce projet comportait plusieurs contraintes : des technologies imposées, un ensemble de fonctionnalités obligatoires et optionnelles, ainsi qu’un travail en groupe de 4 à 5 membres.

À travers ce rapport, nous, groupe 1, allons détailler le déroulement de notre travail et les différentes étapes de l’avancement du projet. Vous y trouverez l’ensemble des informations utiles à la compréhension du fonctionnement de l’application, ainsi que les choix techniques effectués durant son développement.

# 1. Conception

## 1.1 Objectifs fonctionnels

L'application vise à offrir les fonctionnalités classiques d’un réseau social, le tableau ci-dessous résume ainsi les autorisations accordées aux différents profils que l’on pourra retrouvé sur l’application

Une image contenant texte, capture d’écran, nombre, Parallèle

Le contenu généré par l’IA peut être incorrect.

## 1.2 Cahier des charges technique

|  |  |
| --- | --- |
| Front-end | React.js (next.js) |
| Back-end | React.js (Express) |
| Base de données | MangoDB |
| Authentification | JWT |
| Déploiement | Docker |

* Les bibliothèques de composants sont autorisé à partir du moment où elles sont basé sur Tailwind.
* L’utilisation de React.js est demandé mais doit s’effectuer avec comme langage Javascript et non Typescript.

## 1.3 Réflexion graphique

Notre principale source d’inspiration pour ce projet a été Twitter, que nous avons souhaité réinterpréter dans une version la plus minimaliste possible.

L’objectif était de redonner au réseau social sa fonction première : le partage et la discussion.  
Nous avons volontairement écarté les éléments qui, selon nous, ont transformé les applications web actuelles en de véritables murs publicitaires et sources de distraction permanente.

Une image contenant texte, capture d’écran, Téléphone mobile, Police

Le contenu généré par l’IA peut être incorrect.Pour la conception, nous avons adopté une approche mobile first, en commençant par la création de la page de connexion, qui a servi de point de départ à notre réflexion sur l’interface.

Un logo simple, des boutons explicites aux couleurs contrastées pour bien marquer leur fonction primaire, ainsi que des champs de saisie indiquant clairement les informations attendues : tout a été pensé pour concevoir une page claire et accessible.  
L’objectif était de permettre à tout utilisateur, même non familier des réseaux sociaux, de pouvoir l’utiliser sans se poser de questions.

Comme exprimé précédemment, la maquette a été pensée en mobile first, et les éléments ont ensuite été ajustés au cours du développement afin d’assurer une bonne adaptation responsive sur les différents formats d’écran.  
Voici un aperçu du rendu des pages une fois adaptées à un affichage sur écran d’ordinateur :

Une image contenant texte, capture d’écran, logiciel, conception

Le contenu généré par l’IA peut être incorrect.

La maquette a principalement été réalisée pour la suite sur la version mobile. Et nous avons adapté le responsive lors du développement.

Une fois connecté à son compte (qu’il soit nouvellement créé ou non), l’utilisateur accède à la page d’accueil. Celle-ci reste fidèle à notre ligne directrice : un design minimaliste, ne conservant que l’essentiel pour un réseau social.

À noter : l’icône de cloche est affichée en gris. Elle symbolise la future fonctionnalité de notifications, que nous n’avons pas eu le temps d’implémenter, mais dont nous avons voulu réserver l’emplacement dans l’interface.

Une image contenant texte, capture d’écran, Police, nombre

Le contenu généré par l’IA peut être incorrect. Une image contenant texte, capture d’écran

Le contenu généré par l’IA peut être incorrect.

Enfin, nous avons eu une phase de réflexion sur la page du profil utilisateur :

Une image contenant texte, capture d’écran, menu, Police

Le contenu généré par l’IA peut être incorrect.

Un autre défis affronté a été avec la navbar, mais nous ne l’avons pas théorisé dans la maquette par manque de temps. Et avons réalisé ce changement directement dans le code (4.5).

# 2. Workflow

## 2.1 Organisation de l’équipe

D’un point de vue organisationnelle, notre groupe a créé une conversation discord de groupe. Nous avons adopté des réunions un jour sur deux et une répartition claire des tâches :

* Front-end : Baptiste TRONEL-PEROZ, Hugo CELLIER, Jule BLANCHARD
* Back-end : Théo VERSPUECH, Jules BLANCHARD
* Base de données et intégration : Théo VERSPUECH
* Suivi organisationnelle du projet : Hugo CELLIER, Baptiste TRONEL-PEROZ

Malgré ces rôles, tous les membres ont travaillé sur l’ensemble des parties pour la compréhension globale du projet.

## 2.2 Outils utilisés

Les communications de l’équipe ce sont déroulées sur discord à travers un groupe privé. A coté de cela, le projet à pu se baser sur les différents outils que voici :

* Git/GitHub pour la gestion du code source
* Figma pour la conception des maquettes UI/UX
* Postman pour les tests API

Une image contenant capture d’écran, texte, Logiciel multimédia, logiciel

Le contenu généré par l’IA peut être incorrect. Une image contenant texte, capture d’écran, logiciel, Logiciel multimédia

Le contenu généré par l’IA peut être incorrect.

<figma : https://www.figma.com/design/XSSerPqNnU9MTk86BMFm9b/Untitled?node-id=0-1&m=dev&t=K0SymTVqkBqRdEZ8-1>

github : <https://github.com/BaptisteTP/DAD-Equipe-1.git>

## 2.3 Méthodologie

Nous n’avons pas adopté de méthodologie particulière tel que la méthodologie agile ou scrum. Nous sommes parti sur des objectifs fixés en priorisant le développement de feature comparés à d’autres.  
 Nous sommes parti sur les 10 premiers jours sur la création du répertoire gtihub. Ainsi que la création des premières fonctionnalité back-end.  
 nous voulions que l’API soit en avance comparé au front-end pour permettre de développé les fonctionnalités sur le front-end sans devoir créé le back-end de 0 à chaques fonctionnalités.

Pendant la mise en place de l’API, nous nous sommes penchés sur la création d’une maquette figma. Ce support nous permis par la suite d’estimer une liste de composant, les premières règles tailwinds que nous y appliqueront, et la composition des pages et des liens entre elles. Nous n’avons mis en place qu’une simple maquette de quelques page sous format téléphone. Nous voulions mettre en image certaines idées. Le reste de la conception sera faite en simultané du développement pour s’adapter vraiment aux écrans étant donné qu’aucun membre du groupe n’était expert en figma.

Une fois les idées fixés, nous nous sommes mis à la création d’un front-end statique en créant les composants et les pages que nous avions pu identifier. Cela passer par définir leur fichier, leur mise en forme. Et composé des premières pages statiques avec leurs propriétés tailwind.  
 Une fois le back-end terminé et mis en forme de l’architecture décidé durant la phase de réflexion. Nous avons commencé la mise en place de la communication avec l’API. Nous remplaçions petit à petit les composants statique par des composants récupérant les propriétés nécessaires dans le back-end. Nous nous sommes alors retrouvé face aux erreurs que nous avons décidé de géré petit à petit pour ne pas les accumuler et devoir faire des session de débuggage trop lourde.

Au niveau commitologie, nous utilisons une branch develop pour regrouper tous les développements réussi. La création des branches été libre par les membres du groupe selon leurs besoins et envies. Une branche n’était pas forcément liée à une seule feature ou autre. Une fois les fonctionnalités proposées par la branche utilisable, la branche se retrouve merge sur develop, chacun des devs doit alors rebase sa branche sur develop et continuer son développement.

Avec autant de tâches en simultanées, et certains membres qui ne sont pas développeur en alternance. Nous avons eu recours plusieurs fois au paire-programming pour éviter de créer un surplu de conflict dans nos branches.

L’important était surtout de communiquer lors des problèmes pour les résoudre vite et pouvoir avancer ensemble dessus.

# 3. Architecture

## 3.1 Architecture générale

L’architecture globale du projet est décomposé dans 2 grands fichiers pour le back-end et le front-end.

Le back-end repose sur le framework Express.js et est structuré autour de plusieurs microservices permettant d’assurer les différentes fonctionnalités de l’application.  
Il s’appuie sur une base de données MongoDB, choisie pour sa souplesse et sa compatibilité avec les architectures modernes orientées documents.

L’organisation des fichiers respecte les conventions les plus répandues dans l’écosystème Node.js, avec une séparation claire entre les controllers, middlewares, routes, etc., afin de garantir une architecture maintenable et évolutive.

Une image contenant texte, capture d’écran, Police, nombre

Le contenu généré par l’IA peut être incorrect. Une image contenant texte, capture d’écran, Police

Le contenu généré par l’IA peut être incorrect.

Une image contenant texte, capture d’écran, Police, nombre

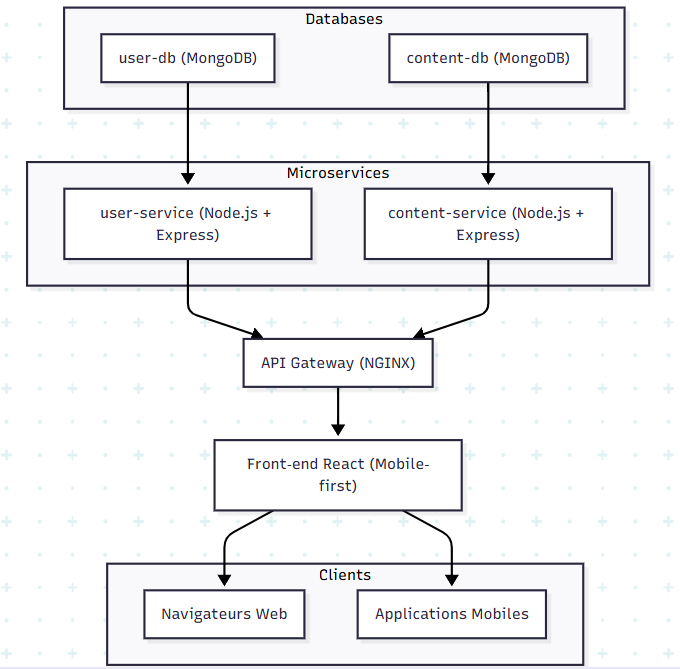
Le contenu généré par l’IA peut être incorrect.Le front-end est basé sur le framework Next.js, qui nous a permis de bénéficier d’un routage optimisé et d’un rendu côté serveur (SSR) facilitant la performance et le référencement.  
La structure du projet se trouve dans un dossier src, contenant notamment :

* le répertoire app, où sont définies les pages de l’application,
* un dossier components, regroupant les composants réutilisables,
* un dossier assets, pour les ressources graphiques et autres fichiers partagés.

Nous utilisons également le Context API de React pour assurer la gestion et la transmission d’état entre les différents composants.  
L’ensemble de l’interface est stylisé à l’aide de Tailwind CSS, complété par un fichier global.css pour les styles généraux et personnalisés.

## 3.2 Schéma technique

Schéma globale des communications entres les composants de l’application



# 4. Résultat du développement

## 4.1 Fonctionnalités terminées

À ce jour, l’application intègre l’ensemble des 11 fonctionnalités obligatoires prévues dans le cahier des charges. Ces fonctionnalités couvrent les bases d’un réseau social : création de compte, connexion, publication de contenu, affichage des messages, interactions, etc.

Toutefois, la gestion des permissions n’a pas encore été implémentée de manière complète. Les utilisateurs ne disposent pas de rôles attribués (tel qu’« utilisateur », « modérateur », ou « administrateur »), ce qui empêche l’activation de certaines restrictions d’accès ou d’actions. Cette amélioration reste une piste pour les évolutions futures de l’application.

Nous avons également intégré une fonctionnalité de changement de thème (mode clair / mode sombre), permettant à l’utilisateur de personnaliser l’apparence de l’interface. Cependant, cette préférence n’est pour l’instant pas liée au profil utilisateur, ce qui signifie qu’un changement de thème est conservé localement (via le stockage navigateur), même après une déconnexion ou un changement de compte.

Enfin, une étape de déploiement est en cours afin de rendre l’application accessible publiquement. Cela nous permettra de faciliter les démonstrations, les tests multi-utilisateurs, et de simuler des conditions plus proches d’une utilisation réelle.

## 4.2 Limites connues

Malgré les fonctionnalités déjà opérationnelles, certaines limitations subsistent dans la version actuelle de l’application. Elles sont recensées ci-dessous :

Les publications (posts) et les commentaires ne peuvent actuellement contenir que du texte. Il n’est pas encore possible d’y ajouter des images, vidéos ou tout autre type de fichier joint. Cette fonctionnalité nécessiterait la mise en place d’un système de stockage (local ou cloud) ainsi que d’une interface de téléchargement sécurisée.

Un bouton représentant une cloche a été intégré à l’interface afin de réserver un emplacement pour le futur système de notifications. Cependant, aucun menu ou logique fonctionnelle ne lui est actuellement associé. Le bouton est donc purement décoratif à ce stade.

Un menu de sélection de la langue est présent dans l’interface utilisateur, mais il ne déclenche aucun changement dans le contenu affiché. L’internationalisation (i18n) n’a pas été mise en œuvre. Cela nécessiterait l’intégration d’un système de gestion de traduction, tel que next-i18next, et l’adaptation de tous les textes de l’application à un format multilingue.

À l’heure actuelle, tous les utilisateurs disposent des mêmes droits et accès. Le système de rôles (ex. : utilisateur, modérateur, administrateur) n’est pas encore en place. Cela limite la possibilité d’introduire des restrictions ou des règles spécifiques, comme la modération de contenus ou la gestion des utilisateurs. L’implémentation de ce système passerait par une structure de rôles dans la base de données et une logique de filtrage côté serveur selon les privilèges attribués.

## 4.3 Capture d’écran

Une image contenant capture d’écran, texte, logiciel, Logiciel multimédia

Le contenu généré par l’IA peut être incorrect.Accueil d’un utilisateur connecté : Profile d’un utilisateur connecté :

Une image contenant capture d’écran, texte, logiciel, Logiciel multimédia

Le contenu généré par l’IA peut être incorrect.

Comme il est montré sur ces screens, nous avons mis en place pour le format ordinateur une navbar sur le coté de l’écran avec l’ensemble des menus utiles. Ce menu est aussi présent sur smartphone mais de manière ouvrable, et il s’ajoute au footer.

Nous voulions au maximum faciliter la navigation de l’utilisateur sur notre application. Et pour cela les menus sont tous clairement notés et accessibles.

## 4.4 Perspectives d’évolution

Les objectifs seraient à présent de mettre en place l’entièreté des fonctionnalités optionelles. Nous voudrions aussi faire une revisite du graphique de l’application pour réussir à la rendre plus cohérente sur elle-même (harmonisation de l’ensemble des affichages).

Nous pourrions également envisager une conversion des fichiers javascript en typescript pour solidifier le typage des variables employés.

Ou encore envisager l’ajout d’un framework de component pour optimisé la partie front-end de l’application.

## 4.5 Les grands choix pris

Pour l’application, malgré la conception de la maquette, il nous a fallu à plusieurs reprise faire des choix durant le développement :

* Les boutons ne sont plus en noir sur le thème blanc : choix graphique pour la simplicité
* Nous étions parti sur un back groupé en un seul micro-service, puis suite à la réflexion architecturale. Nous l’avons divisé en 2 micro-service différents ce qui rendais l’application plus optimisée.
* Nous avons également niveau front l’icon de listes des utilisateurs qui est devenu un icon de recherche. Nous trouvions cela plus approprié pour rechercher un utilisateur.