Présentation générale du projet

Ce projet s'inscrit dans le cadre du module de développement web de première année d'ingénierie informatique. L'objectif principal est de concevoir une application web complète en mettant en œuvre des technologies modernes du développement web, côté frontend comme backend.

Le projet est organisé selon une architecture full stack : une partie frontend pour l'interface utilisateur, développée avec React et Vite comme système de transpilation, et une partie backend reposant sur Java avec le framework Spring Boot.

- ➔ Le backend assure la prise en charge de la logique métier, c'est-à-dire l'ensemble des règles et traitements définissant le comportement fonctionnel de l'application, ainsi que la gestion des données et leur interaction avec la base de données. Il expose une API REST, permettant au frontend d'interagir de manière structurée avec les différentes ressources du système (telles que les utilisateurs ou les objets), tout en intégrant des mécanismes de gestion de l'authentification et des droits d'accès. Le backend est écrit en Java 21, structuré via Spring Boot, un framework moderne qui facilite la création d'API REST robustes et sécurisées. L'outil Maven est utilisé pour la gestion des dépendances, la compilation et les tests.

- ➔ Le frontend est développé avec React, une bibliothèque JavaScript permettant de construire des interfaces utilisateurs réactives et modulaires. Cette partie gère l'affichage, la navigation (via React Router), les interactions utilisateur et les appels à l'API REST exposée par le backend. L'interface est stylisée à l'aide de CSS et de composants personnalisés.

Ce découpage clair entre frontend et backend permet une séparation des responsabilités, favorise la maintenabilité et rend possible le déploiement indépendant des deux parties dans un environnement Dockerisé.

L'ensemble du code source est hébergé sur GitHub à l'adresse suivante : https://github.com/fleefie/projet-devweb-ing1

Choix du thème

Le thème retenu pour ce projet est : “Développer une plateforme numérique intelligente destinée aux utilisateurs d’une ville intelligente”.

Objectifs et enjeux :

L’objectif de cette application est de proposer un site web centralisant des fonctionnalités utiles pour différents types d’utilisateurs (simples visiteurs, citoyens connectés, administrateurs), dans le contexte d’une collectivité connectée.

Le site est structuré autour d’un système de gestion des rôles caractérisé par des points:

- ➔ Invité : Un visiteur non connecté peut accéder à certaines fonctionnalités de manière restreinte, notamment rechercher des objets présents sur la plateforme. Il ne gagne aucun points.

- ➔ Utilisateur : Un utilisateur authentifié peut créer un profil, le modifier, effectuer des recherches sur les objets ou sur d’autres utilisateurs enregistrés ou les signaler à un administrateur. L’utilisateur gagne des points en recherchant des informations et une fois un nombre de points atteint il obtient le rôle d’utilisateur complexe et peut modifier et ajouter des objets.

- ➔ Administrateur : En plus des fonctions utilisateurs, un admin a accès à une interface de gestion qui lui permet de :

    - ◆ Authentifier un utilisateur (valider son identité ou son statut),

    - ◆ Consulter les informations détaillées d’un utilisateur,

    - ◆ Créer, rechercher, modifier, ou supprimer des objets ou des utilisateurs.
    - ◆ Consulter les signalements des utilisateurs et les accepter ou non.

Le choix de ce thème permet à la fois :

- ➔ De traiter des fonctionnalités concrètes, orientées utilisateur et gestion des données.

- ➔ D'introduire une hiérarchisation des droits d'accès (rôle-based access control), très répandue dans les applications web modernes.

- ➔ De mettre en œuvre une interface intuitive pour les différents profils d'utilisateurs, en respectant les bonnes pratiques de développement frontend et backend.

Le projet se concentre ainsi sur un ensemble limité mais cohérent de fonctionnalités, bien ciblées, permettant de démontrer la maîtrise des outils techniques tout en construisant une application utile et fonctionnelle.

Outils et technologies choisis :

Technologies Frontend :

Le frontend de l'application a été développé à l'aide des technologies suivantes :

- ➔ React :
  React est une bibliothèque JavaScript populaire pour la construction d'interfaces utilisateur dynamiques. Elle permet de créer des composants modulaires et réutilisables, rendant ainsi l'interface utilisateur réactive et flexible. React facilite également la gestion de l'état de l'application grâce à son système de gestion d'état local et à des outils comme React Context ou Redux pour un état global. La navigation au sein de l'application est gérée avec React Router, permettant de créer une expérience utilisateur fluide sans rechargement de page.

- ➔ CSS :
  Pour la mise en forme, l'application utilise CSS pour définir l'apparence et la disposition des composants. Des feuilles de style modulaires sont appliquées, et des composants CSS personnalisés sont créés pour s'adapter aux besoins spécifiques de l'interface utilisateur. Des techniques comme les Flexbox et CSS

Grid sont utilisées pour une disposition fluide et adaptable sur différentes tailles d'écran.

➔ Node + Vite:

- Node est l'outil de construction pour le frontend. Il gère les dépendances et la construction. En soi, Node est simplement une implémentation de l'interpréteur JS. Cependant, le framework Vite implémente des outils avec Node pour pouvoir facilement démarrer, compiler et tester un projet JS. React est alors utilisé au travers d'un plugin Vite pour facilement l'ajouter à un projet. Vite permet aussi de très facilement utiliser React avec le framework TypeScript, ainsi que d'autres frameworks tels que Tailwind. Ces deux ont étés utilisés au début, mais TS est trop complexe pour un premier projet de développement web, et Tailwind rend le code beaucoup moins lisible en ne donnant aucun avantage autre qu'un code avec du CSS de partout.

Technologies Backend :

Le backend repose sur les technologies suivantes :

➔ Java 21 :

Le backend est écrit en Java 21, un langage de programmation puissant, fiable et largement utilisé pour le développement de systèmes complexes. La version 21 de Java offre des fonctionnalités modernes et des améliorations de performance, ce qui le rend particulièrement adapté pour les applications à grande échelle.

➔ Spring Boot :

Spring Boot est un framework Java permettant de créer des applications backend modulaires et évolutives. Il simplifie la création d'API RESTful grâce à une configuration automatique et des outils de gestion de sécurité intégrés. Spring Boot facilite également l'intégration avec des bases de données et des outils externes, tout en offrant une grande flexibilité pour la gestion des mappages d'URL, des contrôleurs et des services.

➔ Maven :

Maven est un outil de gestion de projet pour Java. Il est utilisé pour la gestion des dépendances, la compilation, les tests et la construction de l'application. Maven permet de simplifier l'intégration des librairies et des plugins nécessaires au projet, tout en garantissant une gestion cohérente des versions de

chaque composant. Un plugin a notamment permis l'utilisation de Node par le biais de Maven, centralisant la construction du projet.

➔ Spring Security :
Pour la gestion de la sécurité et de l'authentification, le projet utilise Spring Security. Ce framework permet de gérer de manière robuste et flexible les connexions des utilisateurs, la gestion des rôles et la protection des API via le mécanisme de JWT (JSON Web Tokens).

➔ SQLite:
- ◆ La base de données est créée autour de SQLite. Elle est utilisée par Spring par l'API de persistance Java (JPA). Cette API est implémentée par Hibernate. Ce lien a permis de transformer des objets Java en des entrées dans la base de données, sans devoir passer par des méthodes de traitement inappropriées. Le framework de migration Flyway a permis de facilement mettre en place et mettre à jour nos versions individuelles de la base de données, utilisée lors de nos tests.

➔ Jackson:
- ◆ Jackson est un puissant outil de sérialisation. Il a permis de grandement faciliter nos controlleurs REST. Cette bibliothèque permet de transformer une classe Java en un objet JSON, qui est alors utilisé par le frontend. De même, les données envoyées par le frontend ont été très faciles à désérialiser.

Outils de développement :

➔ Docker :
L'application utilise Docker pour la contenerisation. Cela permet de créer un environnement de développement reproductible, simplifiant le déploiement de l'application dans différents environnements (local, production, etc.). Docker permet de s'assurer que l'application fonctionnera de manière identique, quelle que soit la machine ou l'environnement.

➔ Postman :
Postman est utilisé pour tester les API REST du backend. Il permet de simuler les requêtes HTTP, de vérifier les réponses de l'API et de s'assurer que les endpoints sont correctement configurés et fonctionnent comme attendu.

Outils de collaboration :

➔ Git / GitHub :
Le projet utilise Git comme système de contrôle de version. Le code source est hébergé sur GitHub, une plateforme permettant une collaboration en temps réel entre les membres de l'équipe, le suivi des évolutions du code et la gestion des versions via des branches et des pull requests.

➔ Discord : Discord permet de créer un groupe, d'envoyer des messages et des fichiers mais également de s'appeler pour communiquer à propos du projet. Notre équipe a essentiellement échangé sur Discord pour tout ce qui concerne l'application à développer, les problèmes rencontrés, les tâches à effectuer…

Répartition des tâches :

Dans notre équipe, il y a des étudiants qui ont déjà travaillé sur des projets pendant leurs années de PRE-ING à CY Tech. Ayant davantage d'expérience, ils se sont occupés de la partie plus compliquée et délicate qu'est le développement du backend, c'est-à-dire les fondations du projet.

Ainsi, Waally BOUHADOU s'est chargé de mettre en place toute la partie API, interface, BDD et serveur, à l'aide de Java 21 et Spring Boot, ainsi que la structure générale du projet, comme la mise en place des outils de constructions Maven et NPM, ainsi que les choix des technologies utilisés.

Yanis LAID, quant à lui, a mis en place le lien entre les deux parties, en créant la majorité des fonctions du frontend qui communiquent avec l'API exposée par le backend. Il s'est chargé de l'implémentation et du prototypage du frontend.

Alice PARENT, Paul GUIDERDONI et Baptiste LAMBERT ont été chargés de finaliser le frontend. À partir du prototype proposé par Yanis, ils ont créé le CSS et l'agencement, permettant d'implémenter les fonctions d'utilisation de l'API de Yanis, tout en y associant une expérience utilisateur finalisée.

Étapes réalisées :

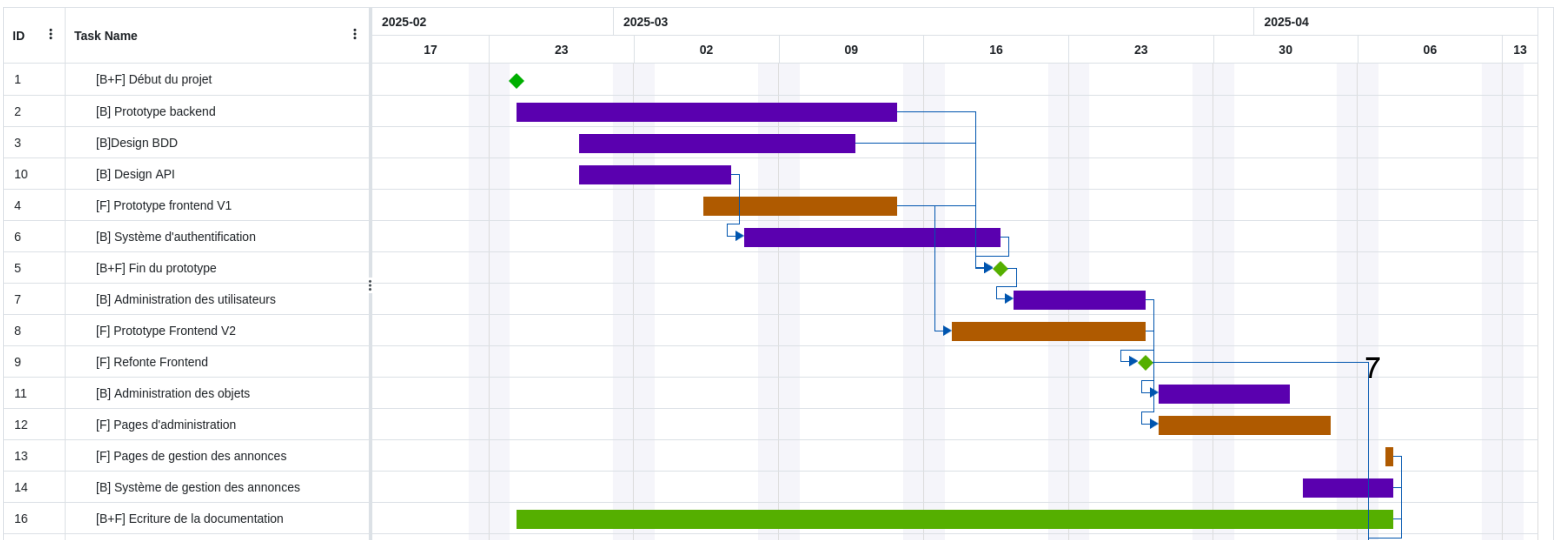
Planification :

Dans un premier temps, il a fallu décider du thème que l'on souhaitait traiter ; ville, bâtiment ou établissement intelligent(e). Après concertation, il est ressorti que la ville semblait permettre d'implémenter des fonctionnalités plus nombreuses et plus intéressantes. À partir de ce thème, il a ensuite fallu choisir quels outils allaient être utilisés pour développer l'application, notamment les frameworks. Comme il n'y avait simplement pas assez de temps pour une planification Agile, nous avons utilisé une planification en cascade. Plus spécifiquement, on travaillait en trois temps. D'abord, une fonctionnalité était implémentée au backend, et son API exposée, avant que l'équipe backend passe à la prochaine fonctionnalité. Une fois une API exposée, l'équipe front end commençait par l'implémenter. Une fois l'implémentation terminée, l'équipe CSS et style se chargeait de l'expérience utilisateur. Ce développement en parallèle nous a permis de tout de même sortir le projet dans les temps, malgré le peu de temps donné pour un projet de cette ampleur.

Conception :

Avant de commencer à développer le site, il était nécessaire de configurer et orchestrer l'application : configurer l'environnement, les paramètres de sécurité, la base de données, gérer les dépendances, les tokens, les requêtes, ...
Une fois que cette étape était réglée, on a pu commencer à réfléchir à l'architecture du site en lui-même : page d'accueil, interfaces de création de compte / de connexion, panel administrateur, profil d'utilisateur, pistes artistiques pour le CSS, …

Développement du projet (diagramme de Gantt du projet)



Fonctionnalités implémentés au backend/API:

- Utilisateurs:
  - Authentification
  - Création, suppression, modification, validation
  - Recherche (selon le nom. selon l'E-Mail, selon les rôles)
  - Envoi d'une photo de profil (+photo de base)
  - Signalement
- Objets:
  - Création, suppression, modification (Seulement pour les utilisateurs avancés)
  - Recherche (selon le nom, selon les propriétés)
  - Signalement
- Annonces / posts:
  - Création, suppression, modification
  - Gestion d'accès selon les rôles
  - Recherche selon le titre, le contenu ou les tags
  - Signalement
- Administration:
  - Modification d'utilisateurs
  - Recherche de signalements
  - Suppression de signalements

Fonctionnalités implémentés au frontend/site:

- Utilisateurs:
  - Authentification
  - Création, suppression, modification, validation
  - Recherche (selon le nom. selon l'E-Mail, selon les rôles)
  - Envoi d'une photo de profil (+photo de base)
- Objets:
  - Création, suppression, modification (Seulement pour les utilisateurs avancés)
  - Recherche (selon le nom, selon les propriétés)
  - Créateur de propriétés JSON visuel
- Annonces / posts:
  - Annonces statiques sur la page d'accueil
- Administration:
  - Modification des utilisateurs

Conclusion et perspectives :

Le projet de développement de la plateforme numérique intelligente pour les utilisateurs d'une ville intelligente a été un véritable défi technique, collaboratif et organisationnel.

Compte tenu du temps limité imparti, nous avons dû faire preuve de réactivité, d'organisation et de communication pour mener à bien ce projet ambitieux. Travailler en groupe a ajouté une dimension supplémentaire de complexité, en particulier pour une partie des membres qui ont dû faire face à la première expérience de travail sur un projet collaboratif via Git. Pour eux, gérer les versions de code et assurer une communication fluide via cette plateforme étaient des défis nouveaux. Cependant, cette expérience de collaboration a été enrichissante et nous a permis de mieux structurer notre travail collectif.

L'application, développée selon une architecture full-stack, avec un backend en Java et Spring Boot, et un frontend en React, a permis de répondre aux besoins définis en début de projet. La gestion des rôles utilisateurs, la création de profil, la recherche d'objets, ainsi que l'administration des données utilisateurs ont été intégrées avec succès. Le backend gère la logique métier et la communication avec la base de données, tandis que le frontend propose une interface fluide et réactive, facilitant l'interaction avec les utilisateurs.

Malgré les difficultés liées au temps restreint et au fait de travailler pour la première fois en groupe, nous avons su surmonter les obstacles grâce à une planification intelligente et une répartition efficace des tâches. Les résultats obtenus montrent que l'application répond aux exigences initiales, avec une gestion fluide des utilisateurs et des objets et une sécurisation des données.

Bien que le projet soit aujourd'hui terminé, plusieurs pistes d'amélioration existent. Nous pourrions ajouter des fonctionnalités avancées comme des tests automatisés pour renforcer la fiabilité du système. De plus, l'intégration d'outils de monitoring et de gestion des performances seraient des évolutions intéressantes pour améliorer la scalabilité (= affichage sur différents supports) et la résilience de l'application. Enfin, plusieurs fonctionnalités de l'API n'ont pas pu être implémentées au niveau de l'application web, bien que fonctionnelles au niveau de l'API, à l'aide d'un outil comme Postman ou encore Curl.

En conclusion, bien que le temps court et le défi de travailler en groupe, notamment pour ceux découvrant GitHub, aient représenté des obstacles, ce projet a été une expérience enrichissante, permettant de développer des compétences techniques et de collaboration. Il constitue une base solide pour de futurs projets, et nous a permis de démontrer qu'une plateforme numérique fonctionnelle pour une ville intelligente est non seulement réalisable, mais aussi un excellent terrain d'apprentissage.