01/12/2023

Alexandre Cubizolle

findByDev

Rapport de Stage

Sommaire

1. Présentation du/des projets
   1. Expression des besoins
   2. Organisation du travail
   3. La stack technique (quelles sont les technologies utilisées ?)
2. Conception du projet (aussi appelé « spécifications fonctionnelles et techniques)

A. Analyse des besoins (diagrammes UML de cas d’utilisation))

B. Architecture du projet

                                                               i.      Conception de BDD

                                                             ii.      Diagrammes de package

                                                           iii.      Diagrammes de Séquence

* 1. Maquettage
  2. Conception des tests unitaires

1. Réalisation (faire un focus sur **une ou deux fonctionnalités** avec des exemples de code pertinents)
   1. Rappel du cas d’utilisation
   2. Contraintes techniques
   3. Développement des fonctionnalités avec description de toutes les couches

                                                               i.      Implémentation de la base de données

                                                             ii.      Composant d’accès aux données (couche DAO ou repository avec configuration de l’ORM)

                                                           iii.      Logique métier (possibilité de mettre du pseudo-code et l’implémentation)

                                                           iv.      Interface graphique

                                                             v.      Tests unitaires

* 1. Résultat

1. Veille sur la sécurité (peut être appliquée au projet)

1. Bilan
   1. Ce qui a été fait (ce qui marche, est-ce déployé ? Que dit le client ?)
   2. Perspective d’évolution (ce qu’il reste à faire)
2. Conclusion sur 1 page (appréciation personnelle du projet, satisfaction
3. **Présentation du Projet**

**Introduction**

Afin de valider les compétences pour la validation du titre de concepteur développeur d’application j’ai développé une application orienté « mobile » qui aura pour vocation d’être aussi accessible en desktop dans le futur.

L’utilisation de plusieurs langages et frameworks appris en formation et des éléments qui n’ont pas été étudiés pour apporter un plus à ce projet.

Le projet "FindByDev" répond à un besoin dans le domaine du développement de logiciels. À l'heure actuelle, il n'existe pas de plateforme dédiée permettant aux développeurs de se rencontrer, de partager leurs expériences et d'établir des relations professionnelles et personnelles. Il s'agit en quelque sorte d'un "Tinder" pour les développeurs. L'objectif principal de ce projet est de créer une application de rencontre gratuite destinée exclusivement aux développeurs adultes (âgés de 18 ans et plus).

L'application aura un style visuel similaire aux applications de rencontre populaires, avec des références liées au monde du développement. Mon objectif principal est de protéger les données personnelles des utilisateurs et de garantir la conformité avec les réglementations en matière de protection des données.

La sécurité tiendra une place importante dans ce projet car dès lors que je manipule des données personnelles, la confiance d’un utilisateur est primordiale pour accepter de donner ces informations.

**A : Expression des besoins**

L'application "FindByDev" vise à répondre à plusieurs besoins essentiels :

* + - Permettre aux utilisateurs de créer des profils.
    - Permettre aux utilisateurs de "liker" d'autres profils et attendre un "like" mutuel pour entamer une conversation.
    - Offrir la possibilité de rechercher d'autres utilisateurs qui partagent des intérêts dans le développement de logiciels.
    - Faciliter la communication grâce à la messagerie instantanée.
    - Mettre l'accent sur la sécurité des données personnelles et assurer la conformité avec les réglementations en matière de protection des données.
    - Incorporer des fonctionnalités adaptées aux développeurs, telles que l'intégration avec Git, la mise en évidence des langages de programmation préférés, l'introduction d'une cote de popularité et l'ajout d'éléments humoristiques.

**B : Organisation du travail**

Dans le cadre de ce projet, j’ai choisi d’utiliser un diagramme de Gantt afin d’obtenir visuellement un planning de gestion du projet. De plus, l'utilisation de Git, comme système de versionning, facilite le partage sur plusieurs postes de travail.

**C : La Stack Technique**

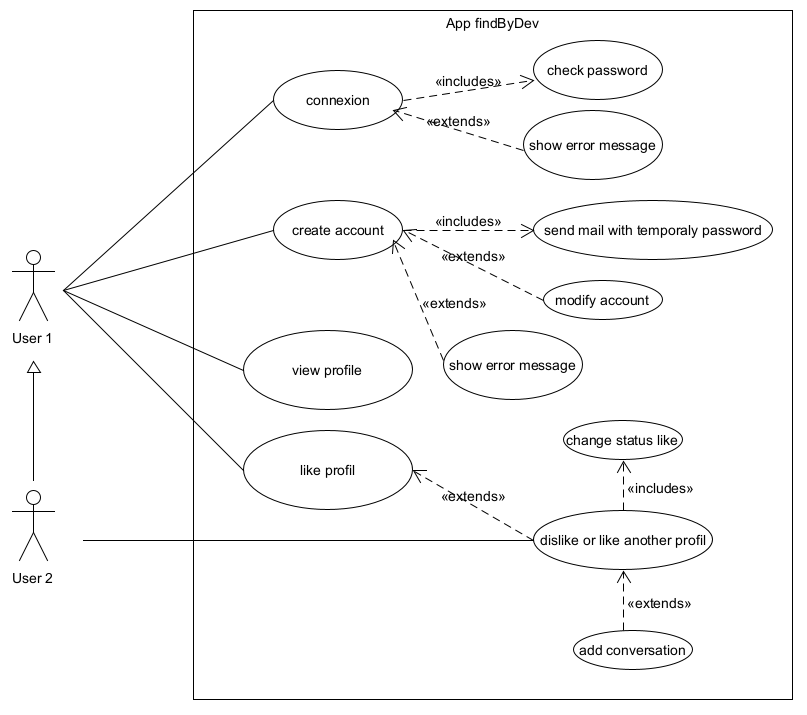
La réalisation de l'application "findByDev" implique l'utilisation d'un ensemble de technologies et d'outils. Cela comprendra la mise en place d'une base de données pour stocker les informations des utilisateurs, un composant d'accès aux données (repository) avec la configuration de l'ORM (Object-Relational Mapping), la logique métier, une interface graphique intuitive, et des tests unitaires pour garantir la fiabilité du code.

* Pour réaliser ce projet j’ai utilisé l’IDE VSCode car il est simple et complet.
* Côté conception c’est UMLet pour les USECases, GranttProject pour la planification, LucidChart pour un diagramme de séquence.
* La conception du MCD/MLD a été faîtes avec Looping, qui est très visuel.
* J’utilise PostGrèsSQL pour la BDD et DBeaver pour sa gestion.
* Côté front développé avec React-pwa adapté au mobile, javascript vanilla.
* SpringBoot pour le back avec un structure MVC.
* Postman pour les tests de requêtes.
* <https://websocketking.com> pour les test de websocket.

Dans la suite de ce rapport, je vais aborder en détail les différentes phases du projet, de la conception à la réalisation, en mettant en évidence les principaux éléments et choix techniques. Nous explorerons également les aspects de sécurité qui sont essentiels pour protéger les données sensibles des utilisateurs. Enfin, nous dresserons un bilan des réalisations à ce stade du projet, ainsi que des perspectives d'évolution pour les prochaines étapes.

1. **Conception du projet**
   1. **Analyse des besoins**

Après avoir défini l’expression de besoins il a fallu déterminer quel serais les possibilités des futurs utilisateurs. Cela permet notamment de visualiser ou mettre de la sécurité aux endroits où ils n’ont pas accès. Etant une application de rencontre, il faut prendre en compte que l’interaction principale se déroulera entre deux protagonistes.

****

**Gestion des profils et des interactions**

**Acteurs :**

* User 1 (utilisateur principal)
* User 2 (utilisateur secondaire)

**Description :** Ce cas d'utilisation couvre les différentes fonctionnalités de l'application "findByDev" liées à la gestion des possibilités des profils, des interactions et des comptes des utilisateurs.

**Scénario principal :**

L'utilisateur User 1 se connecte à l'application. La connexion comprend la vérification du mot de passe et, en cas d'erreur, l'affichage d'un message d'erreur.

Une fois connecté, User 1 peut choisir de créer un compte ou de voir son profil existant.

Si User 1 décide de créer un compte, il doit fournir les informations nécessaires. La création de compte implique également l'envoi d'un e-mail contenant un mot de passe temporaire qu’il devra changer à sa première connexion.

Si User 1 décide de voir son profil existant, il peut afficher les détails de son profil et en modifier les éléments.

User 1 a la possibilité de "liker" un profil d'un autre utilisateur. Cette action peut être étendue pour "disliker" le profil de User 2. La gestion de cette interaction inclut la modification de l'état de "like" qui au départ est à « EN\_ATTENTE » et peut entraîner l'ajout d'une conversation si le like est réciproque.

L'utilisateur User 2, qui hérite des fonctionnalités de User 1, peut "disliker" ou "liker" un profil d'un autre utilisateur qui l’aurai liké. L'interaction est similaire à celle de User 1.

Lors de l'interaction de "like" ou "dislike", l'application gère un changement de statut de "like" et permettre l'ajout d'une conversation entre les utilisateurs si le profil est également liké.

Ce cas d'utilisation "Gestion des profils et des interactions" couvre les principales fonctionnalités l’application "findByDev", y compris la création de comptes, la visualisation de profils, les interactions "like" et "dislike", la gestion des conversations et la gestion d'erreurs lors de la connexion ou de la création de compte. Il permet aux utilisateurs de s'engager avec d'autres profils d'utilisateurs et de gérer leurs propres informations de compte.

B. **Architecture du projet**

1. **Conception de BDD**

Avant la création d’un BDD et afin d’éviter trop de modifications une fois en place, j’ai créé un dictionnaire des données pour identifier les principales interactions.

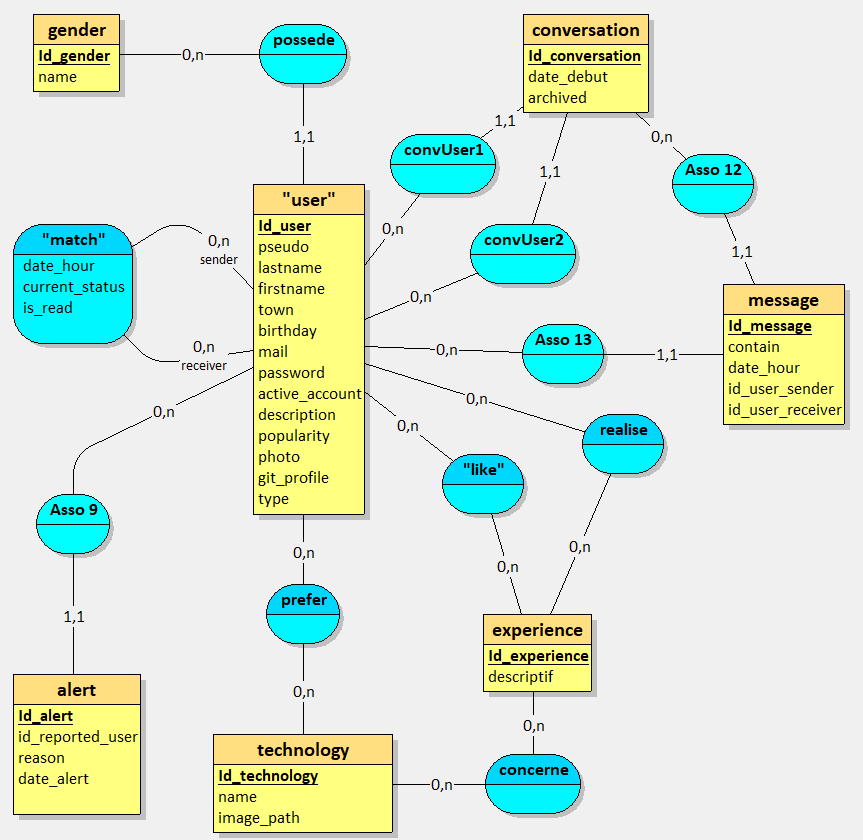
****Il va me servir à organiser et décrire toutes les données pertinentes qui seront utilisées dans l’application. Cela documente le projet et permet une meilleure compréhension pour un œil extérieur. Un simple fichier Excel suffit à mettre à plat ses idées.

C’est un échantillon de ce que sera la BDD, avec les futurs noms de colonne qui, au cours du développement, pourront être adaptés selon les besoins encore non-identifiés. Cela m’a aussi permis d’identifier les futures relations qui seront mise en place.

1. **Looping**

MCD :

Ma BDD contient 7 tables principales, cela reste très visuel mais ça nous permet d’identifier ces fameuses relations notamment la relation « Match » qui va se transformer en table récessive avec une gestion back légèrement différente aux autres tables.



Mise à part vers la table « gender », les autres relations qui partent de « user » sont toutes en 0, n car un utilisateur peut très bien être inactif et ne pas avoir de conversation, de match ou encore un signalement qui aura été fait à son encontre.

Il sera cependant bien inscrit et son profil sera visible des autres. Il a été pensé une note de popularité attribué à un utilisateur et incrémenté selon deux critères : le nombre de fois que sa fiche a été visité et le nombre de matchs reçus. Le but sera de mettre en avant les profils actifs le plus possible et incité les utilisateurs à interagir. Cette note n’est visible que de l’utilisateur.

**MLD (Annexe 1)**

En MLD (Modèle Logique de Données), on identifie un total de 5 associations qui sont transformées en tables d'association, composées des deux foreign keys, afin de représenter les relations entre les différentes entités de manière structurée et normalisée. Celles-ci aussi auront un traitement un peu particulier en Back sur SpringBoot.

**Diagramme de Séquence**

**(Annexe 2)**

Le processus d’inscription répond à plusieurs exigences, notamment à celle de répondre à la problématique première de ce site, la vérification de l’âge de l’utilisateur. Il n’est bien sur pas possible de demander une pièce d’identité aux utilisateurs à l’inscription, alors je vérifie en front/back et BDD si la date d’anniversaire entrée est bien supérieure à 18 ans.

La deuxième problématique et celle de la responsabilité de chaque information demandée et les regroupé selon un schéma bien précis. L’utilisateur est guidé tout au long de son processus d’inscription.

Je vérifie que l’email entré n’existe pas déjà, pour rester cohérent cette vérification se fait aussi à trois niveaux.

Je ne demande pas de mot de passe sur le formulaire car il sera géré après, lors de l’inscription un email est envoyé à utilisateur (cela permet aussi de vérifier que cet email est valide). Cet email contient un mot de passe temporaire qui répond à tous les critères de sécurité demandé. Celui-ci sera demande à la première connexion et devra être immédiatement changer.

Concernant les critères de sécurité demandé j’ai suivi les recommandations de la CNIL qui préconise un minimum de 11 caractères avec une majuscule, une minuscule, un chiffre et un caractère spécial. Selon leur étude il faudrait 3 ans à un hacker pour trouver ce mot de passe par force brut.

Une fois que l’utilisateur change sont mot de passe (vérification faîte qu’il n’utilise pas le mot de passe temporaire comme nouveau mot de passe) il arrive sur l’upload de la photo de profil qui lui sera associé. Une fois fait s’ouvre le processus d’enregistrement du profil Git.

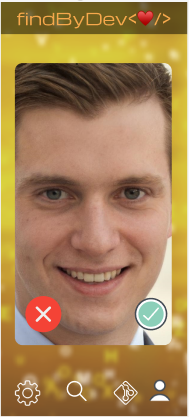
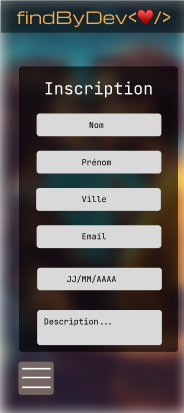
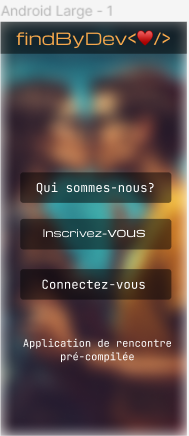
Pour récupérer les principales informations publiques d’un user, nul besoin d’avoir une authentification envers le endpoint Git. Il suffit d’y entrer l’id de l’utilisateur et nous récupérons ces informations. Cependant par soucis de vérifications, j’ai créé une application sur Gitlab afin de m’assurer que mon utilisateur passera par une authentification git qui lui est propre (il doit entrer son email et son mot de passe Git) je m’assure donc que c’est bien son Git que l’utilisateur utilise pour son profil.

Une fois l’autorisation donné à mon application je récupère ces informations grâce à l’acces\_token généré par Gilab. Il me suffit d’utiliser les endpoints de Gitlab et d’accéder à l’id de l’utilisateur ainsi qu’a ses derniers projets et son dernier « push ».

C. **Maquettage**

L’application a été développé en mobile first et a pour objectif d’être aussi viable sur desktop, c’est pour cela que j’ai fait deux visuels. Les backgrounds ont été générés par IA.

Vue Mobile



Pour la version mobile j’ai fait trois fenêtres qui représente l’accueil, le formulaire d’inscription et la vue « Card Swippable » accessible après s’être inscrit. Sur cette vue on y voit le design global qui reprend les codes classique d’applications de rencontre avec la photo d’un utilisateur que l’on peut liké ou pas au centre et les options d’interactions disponible en bas en fixed.

Les Icones choisis au départ ont un peu changé car j’ai recentré les besoins primaires sur ce type d’applications qui sont pouvoir :

* Accéder à son compte
* Retrouver ses conversations
* Voir les notifications
* Faire une recherche
* Revenir sur la vue générale

Sur la photo de profil j’ai également ajouté une icône retour en cas d’erreur de choix de l’utilisateur et le pseudo en haut de la photo. **(Annexe 3)**

Vue Desktop (Annexe 4)

Les couleurs utilisées restent les même avec une utilisation maximale de la place notamment pour la partie principale du site ou j’ai développé une navbar type « Vscode » qui se déploie pour afficher les options associées. Dans le texte de présentation on peut y retrouver le RGPD qui sera finir au moment du déploiement (nom de domaine, propriétaire du site, contact pour réclamation, …).

Le header est une navbar dynamique qui, à la connexion, affichera la photo le pseudo et la note de popularité de l’utilisateur. Un accès à son compte est possible en cliquant sur ce bloc profil.

Cette vue desktop est visible sur des écrans avec une largeur minimale de 928px, en dessous, c’est la vue mobile qui prend le relai.

1. **Test unitaire**

Afin de repérer rapidement les éventuels défaut de code, j’ai mis en place des tests unitaires sur diffèrent module clé de mon application.

(Faire les tests)

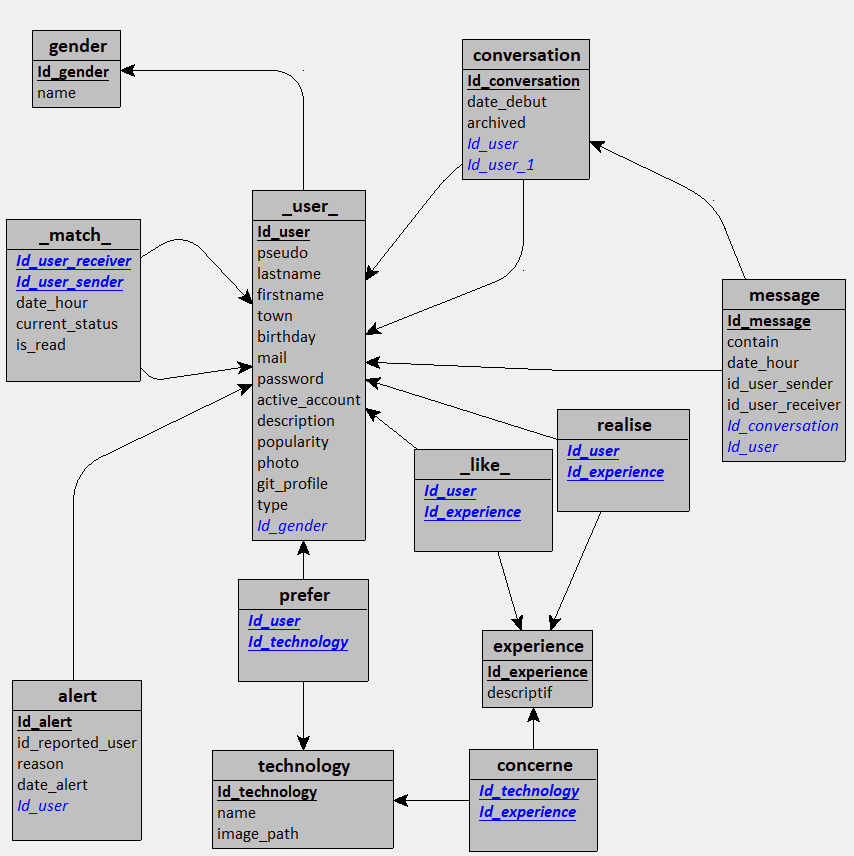
1. **Réalisations**

Cas d’utilisation :

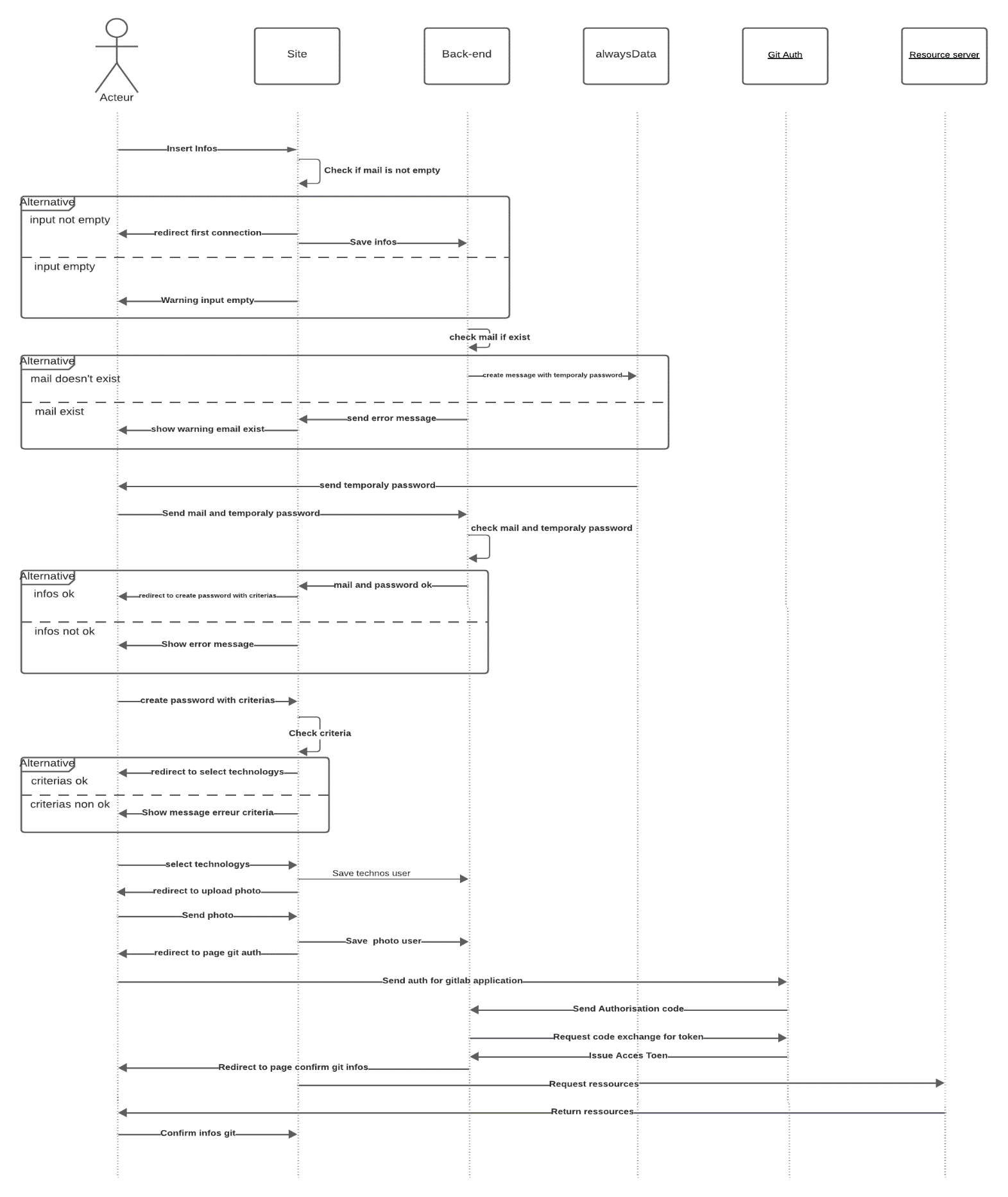
Deux utilisateurs doivent pouvoir communiquer entre eux et passe par l’étape du Like validé pour cela.

**ANNEXES :**

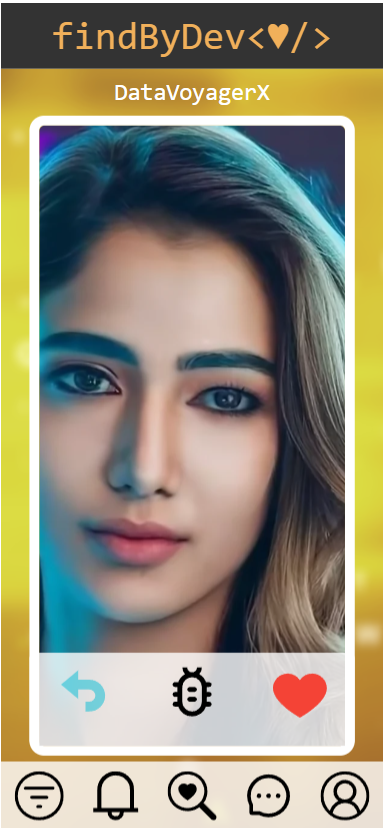
**1.MLD**



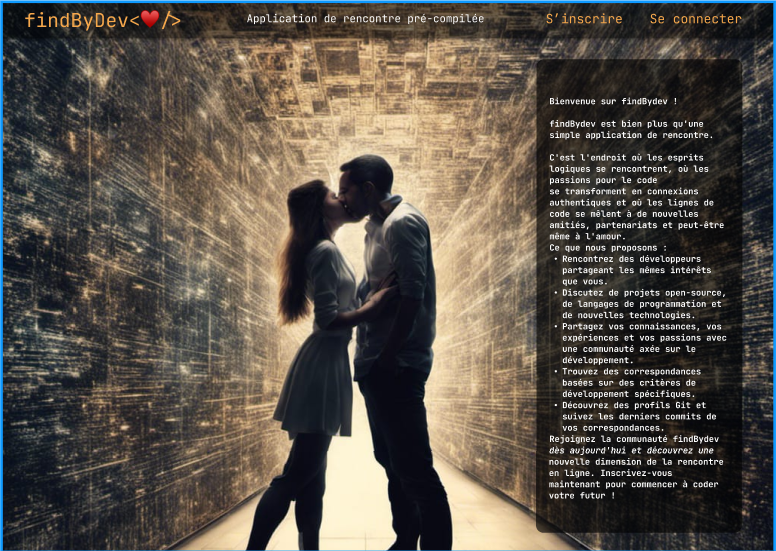
1. **Diagramme de séquence** (processus d’inscription)



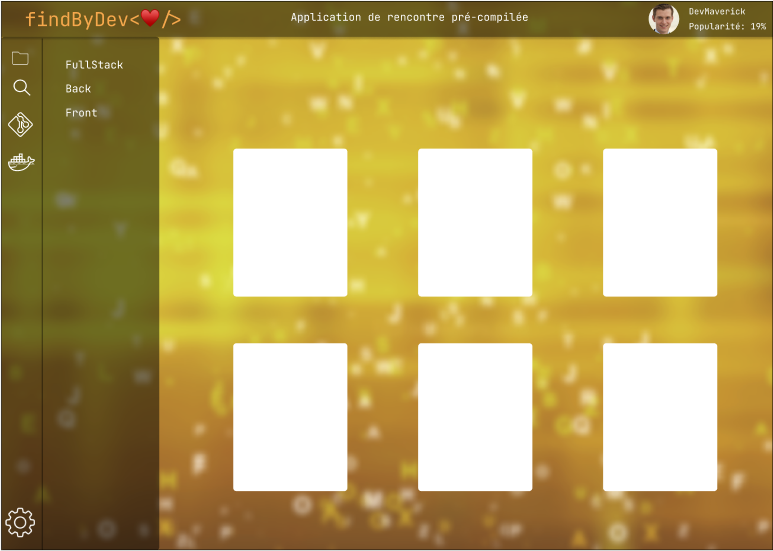
**3.Vue Mobile Card Réel**

****

1. **Vues Desktop Maquette**



****

****