Edelsten	Spécifications techniques	Version : 1
	EDELSTEN Application mobile communautaire de référencement de pierre précieuse et semi-précieuse	Mise à jour : 29/11/2019
		Date de création : 14/11/2019

Auteurs	THERY Adrien MAURY Jérome

Table des matières

1. But

Ce document a pour but de préciser les spécifications techniques de l'application mobile Edelsten. Il décrit l'intégralité de l'architecture de l'application Edelsten. Il se décompose en 4 parties, une introduction, une explication des processus de l'application, une explication de l'architecture et enfin une explication de la conception graphique.

Intitulé	Statut	Date de mise à jour
Introduction	Vérifié	22/11/19
Processus d'application	Vérifié	22/11/19
Architecture	Vérifié	29/11/19
Conception graphique	Vérifié	28/11/19

2. Introduction

2.1. Contexte

Edelsten est une application mobile communautaire de référencement de pierre précieuse et semi-précieuse. Elle offre la possibilité à la communauté de pouvoir contribuer à l'élaboration du catalogue, ainsi que de pouvoir interagir via un espace de discussion lié à chaque référencement.

L'application est divisée en deux sections, un espace encyclopédique listant les différentes pierres et leurs caractéristiques, et un espace personnel permettant à l'utilisateur d'éditer ses propres notes et de conserver des références en favoris. L'application a une interface moderne et intuitive permettant à l'utilisateur une manipulation et une navigation aisée de tous les éléments.

Initialement développé par un groupe d'étudiant dans le cadre d'un projet scolaire, cette application a été racheté par MINDSTONE afin d'être mené à terme pour pouvoir introduire toutes les fonctionnalités désirées.

2.2. Objectif

L'objectif de l'application est d'offrir à une communauté désireuse de pouvoir consulter facilement une encyclopédie dédiée aux pierres précieuses et semi-précieuses et de contribuer à celle-ci. Edelsten permet à chaque utilisateur de s'authentifier afin de disposer de nombreuses fonctionnalités telles que l'ajout de pierres dans un registre de favoris personnels, d'interagir avec la communauté par un système d'interaction sur un modèle like/dislike et un espace de commentaire. La communauté la plus active se verra attribuer un rôle important dans la viabilité et la maintenance du contenu par la possibilité de consulter et valider les requêtes de modifications et d'ajout de celui-ci.

L'objectif de la reprise de ce projet est d'intégrer les fonctionnalités suivantes : l'authentification, l'espace personnel, l'espace communautaire comprenant la modification des ressources par un système de requêtes, et la traduction des ressources en plusieurs langues.

2.3. Besoin

L'application répond à un besoin réel, à l'heure actuelle il existe déjà des applications encyclopédiques de pierres précieuse et semi-précieuses et des applications de lithothérapie mais aucune combinant les deux aspects. Il parait essentiel de faire la jonction de ces deux domaines pour pouvoir fournir facilement toutes les informations sur une seule et même base de données. Elle s'adresse à une communauté ne disposant pas d'outil similaire évoluant dans différents domaines : lithothérapie, joaillerie, minéralogie.

2.4. Exigences

2.4.1. Développer une application simple d'utilisation

L'application doit pouvoir être accessible pour le plus grand nombre d'utilisateur ne disposant d'aucunes compétences particulières :

- L'application doit exiger le minimum de documentation.
- L'application ne doit exiger aucun paramétrage.
- L'application ne doit exiger aucune formation.
- Navigation et saisie des données simple et intuitive.
- L'application doit être disponible dans plusieurs langues.

2.4.2. Développer une application sans contraintes techniques

L'application doit pouvoir s'installer et s'exécuter sur le plus grand nombre d'appareil Android et IOS :

- L'application doit utiliser des technologies supportées par plus de 90% du parc des appareils Android.
- L'application doit utiliser des technologies supportées par plus de 90% du parc des appareils IOS.
- L'application doit pouvoir s'installer sur un appareil IOS depuis le magasin d'application distribué par Apple (App Store).
- L'application doit pouvoir s'installer sur un appareil Android depuis le magasin d'application distribué par Google (Google Play).

3. Processus de l'application

3.1. Spécifications fonctionnelles

3.1.1. Décomposition fonctionnelle

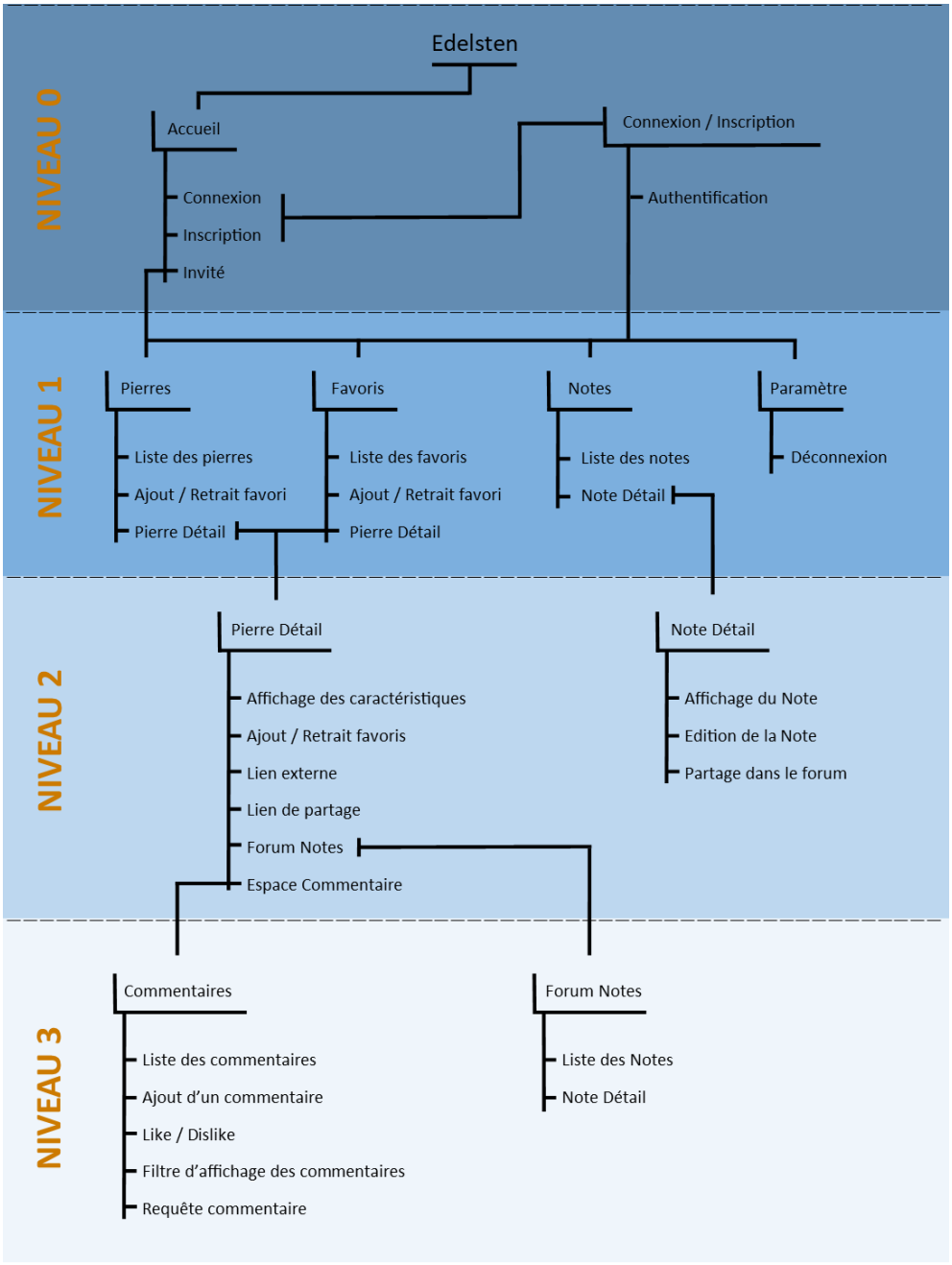
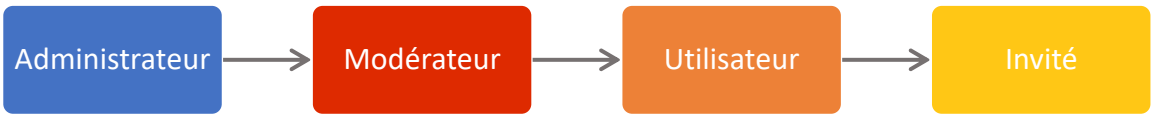


Illustration 1 : Décomposition fonctionnelle des écrans

3.1.2. Organigramme des utilisateurs



Comptes	Suppression	Modification du statut
Admin	✓	✓
Modo	✗	✗
User	✗	✗
Guest	✗	✗

Tableau 1 : Permissions de gestion des comptes

Ressources	Création	Modification	Suppression	Consultation	Proposition
Admin	✓	✓	✓	✓	✓
Modo	✓	✓	✗	✓	✓
User	✗	✗	✗	✓	✓
Guest	✗	✗	✗	✓	✗

Tableau 2 : Permission de la gestion des ressources

Commentaires	Création	Modification	Suppression	Consultation
Admin	✓	✓	✓	✓
Modo	✓	✓	✓	✓
User	✓	✗	✗	✓
Guest	✗	✗	✗	✓

Tableau 3: Permissions de gestion des commentaires

3.1.3. Périmètre fonctionnel

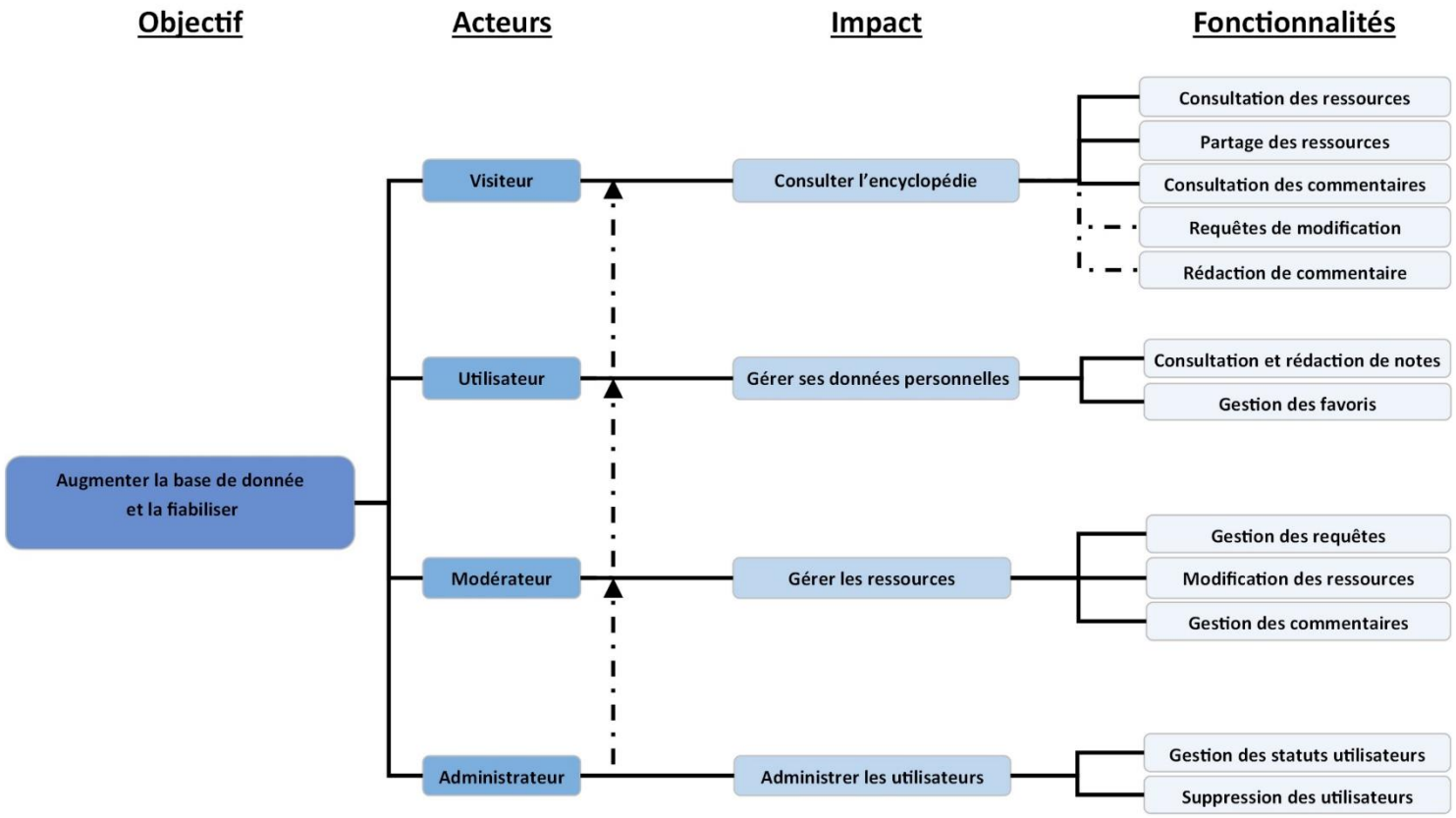


Illustration 2 : Impact mapping de l'application Edelsten

3.2. Spécifications techniques

3.2.1. Technologies utilisées

L'application développée en Dart et utilise le framework Flutter, Firebase est utilisé pour l'authentification des utilisateurs et comme base de données.

Flutter fournit les éléments essentiels permettant la réalisation de l'application :

- Flutter permet la portabilité de l'application. Il permet de pouvoir l'exécuter sous Android et IOS. Il facilite la maintenance et le développement car il n'y a qu'une seule application pour tous les supports.
- Flutter est un kit de développement open-source créé par Google, il est soutenu par une large communauté et très largement poussé par Google.
- Flutter propose la possibilité de l'utiliser depuis Android Studio, Visual Studio Code ou n'importe quel IDE

Parmi les nombreux éléments que propose ce framework, certains sont essentiels par rapport à l'architecture de l'application mise en place. Le set de plugin FlutterFire dispose de tous les plugins nécessaires à la liaison entre l'application et Firebase. Firebase est un ensemble de services d'hébergements pour applications. Racheté par Google, Firebase propose d'héberger en NoSQL et en temps réel des bases de données, du contenu, de l'authentification des notifications et des services de communications. Les différents plugins utilisés sont :

- Firebase_core : Un plugin Flutter pour utiliser l'API Firebase Core, il permet l'utilisation et la connexion de tous les services Firebase.
- Firebase_auth : Un plugin Flutter pour utiliser l'API Firebase Authentification, il permet l'utilisation des services d'authentification de Firebase.
- Cloud_firestore : Un plugin Flutter pour utiliser l'API Firestore, il permet d'utiliser le service de base de données NoSQL de Firebase, simplifiant le stockage, la synchronisation et l'interrogation des données par l'application.
- Firebase_storage : Un plugin Flutter pour utiliser l'API Firebase Cloud Storage, il permet d'utiliser le service de stockage en ligne de Firebase proposé par Google.
- Firestore_mock : Un plugin utilisé pour les tests unitaires.
- Rxdart : Un package qui est l'implémentation en Dart de l'API ReactiveX, qui étend les API Dart relatives au Streams originales pour se conforter aux normes ReactiveX

3.2.2. Modèles des données

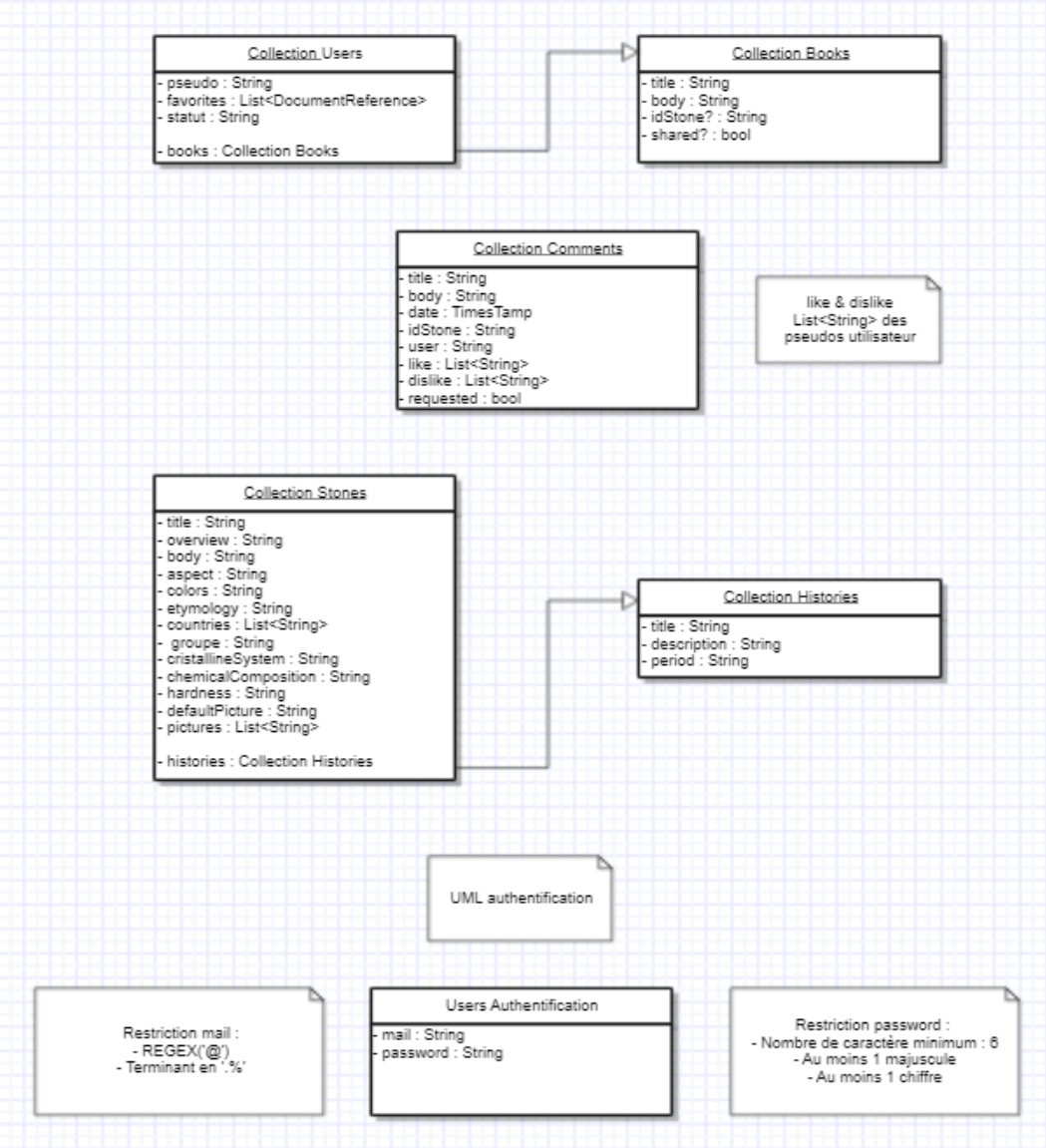


Illustration 3: Diagramme UML des données

Sur ce diagramme on retrouve une table « Collection Users » dans laquelle sont stockées les informations relatives à l'utilisateur (pseudo, favoris, statut, books).
Une table « Collection Stones » qui elle contient toutes les informations relatives aux pierres précieuses et semi-précieuses de la base de données (titre, aspect, couleur, étymologie, pays, composition chimique, dureté…).
Une table « Collection Books » relative à la précédente contenant les informations de la section ''Notes'' de l'espace personnel de l'utilisateur (titre, corps du texte, ID de la pierre et l'état du partage)
Une table « Collection Histories » liée à la précédente contenant les informations relatives à l'histoire des pierres (titre, description, période).
Une table « Collection Comments » qui contient les informations relatives aux commentaires laissés sur l'application (titre, corps du commentaire, date, ID de la pierre, id de l'utilisateur, like/dislike et l'état de la requête).

4. Architecture

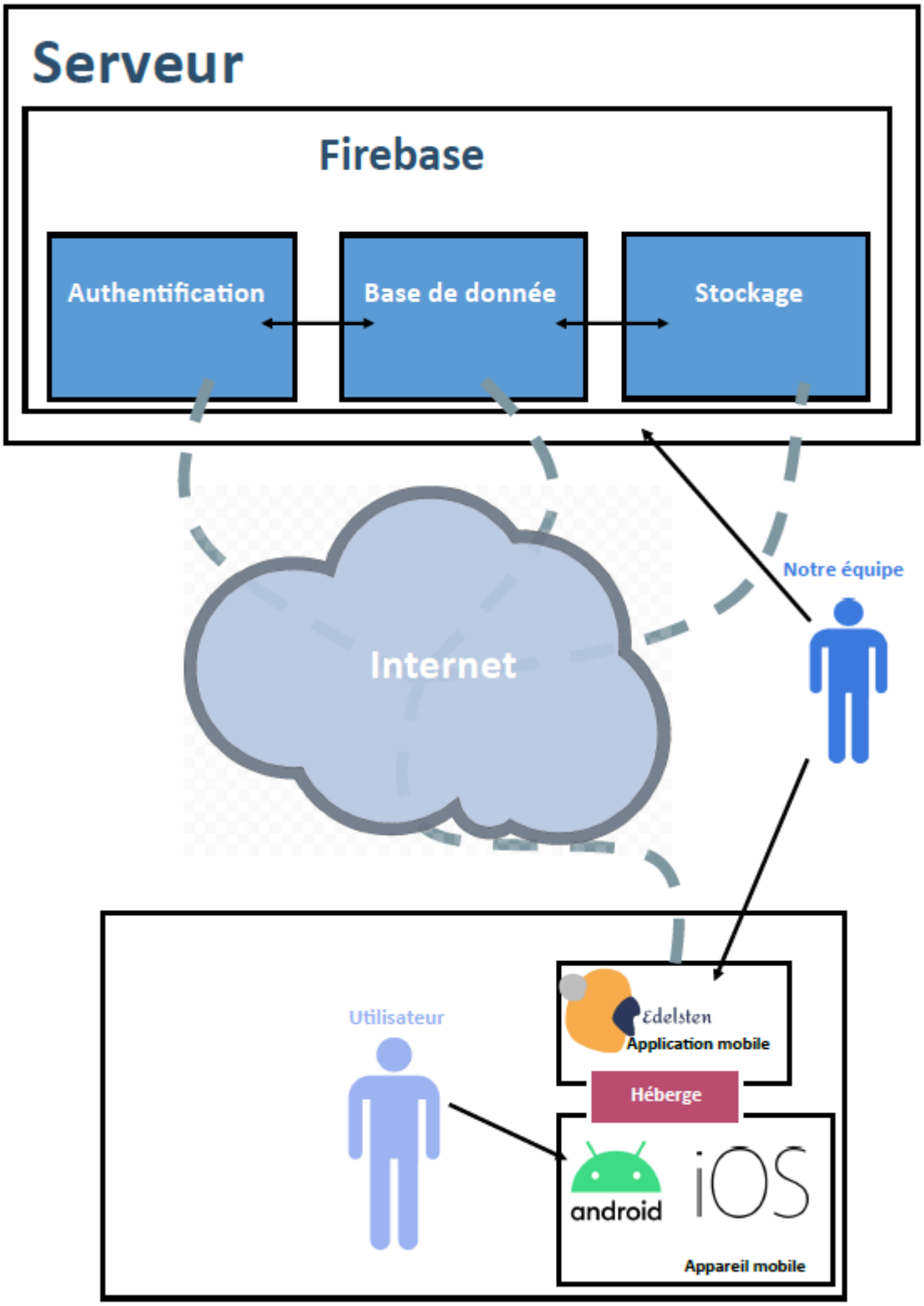


Illustration 4: Architecture globale de Edelsten

4.1. Choix techniques Back-end

Initialement le projet Edelsten fonctionnait sur une base de stockage local pour répondre au besoin de disponibilité des données hors réseaux. Ce système de stockage local était pertinent pour les fonctionnalités de base de l'application comme la disponibilité d'une encyclopédie de pierre précieuse et semi-précieuse. Cependant dans le but d'une évolution communautaire de l'application, il se doit d'exister un système d'authentification et les fonctions lié à l'action d'un utilisateur qui met en déroute les choix d'architectures initiaux.

Pour procéder à l'évolution de l'application, le besoin a évolué et évoluera dans le futur. Il est nécessaire que le système de données soit en ligne, rapide et accessible sans interruption. Les différents besoins actuels sont une base de données, une authentification et un espace de stockage pour les images en ligne.

Plusieurs choix se proposent pour répondre à ces besoins, la simplification de l'accès des données en ligne a été le moteur de décision pour l'architecture Back-End.

La gestion de l'architecture Back-End s'appuie sur plusieurs points : une base de données noSQL, un cloud de stockage et une API permettant l'obtention des datas souhaitées et la gestion des Tokens d'authentification. Ce qui inclus l'utilisation d'un hébergeur et le développement complet d'une API. Or un service développé par Google, FireBase réuni toutes les conditions de notre besoin. C'est une solution clé en main se divisant en plusieurs sous services. La gestion de l'authentification dans un premier service, la disposition d'une base de données noSQL et la possibilité de stocker les images dans un store.

De plus, cette solution permet de faire le suivi de l'application, comme un suivi de performance réunissant toutes les données des utilisateurs afin d'optimiser et corriger les éventuelles parties lourdes de l'application. Elle propose aussi Google Analytics qui permet d'obtenir des statistiques sur l'utilisation de l'application afin d'observer l'activité des utilisateurs et de pouvoir envisager de futures améliorations.

Enfin, FireBase propose des extensions supplémentaires qui pourront servir à l'évolution et l'amélioration de l'application. Il propose un service pour redimensionner les images de la base de données ce qui permettrait de gérer tout type de format et de définition d'image proposée à l'upload par un utilisateur, ou encore un service de traduction de texte basé sur Google Translate pour pouvoir déployer l'application dans plusieurs langues.

4.1.1. Authentification

Dans cette partie, le besoin de l'application est de faciliter l'utilisation de l'application notamment en proposant un système d'authentification facile et rapide. Pour ce faire FireBase propose une authentification depuis plusieurs plateformes comme Google, Facebook, Twitter, GitHub ou encore par inscription mail classique et numéro de téléphone.

De plus afin de garantir la sécurité des comptes utilisateurs, FireBase répond à cette problématique sérieuse. Le service s'appuie sur l'expertise interne de Google grâce à ses propres solutions de sécurité par ses équipes responsables de Google Sign-in, Smart Lock et Chrome Password Manager.

4.1.2. Cloud firestore

Cloud Firestore est le système de base de données de Firebase. C'est base de données orienté documents NoSQL permettant de stocker, synchroniser et interroger facilement et rapidement les datas. Cloud Firestore permet d'organiser les datas avec des collections et documents et propose une API d'accès pour interroger la base de données efficacement. Cette API est une API brute en temps réel capable de gérer des « Streams » sur des datas afin de toujours utiliser un jeu de donnée à jour à tout moment. De plus Firestore met à disposition un kit de développement (Cloud Functions) permettant d'exécuter du code Back-End répondant aux modifications souhaitées dans la base de données.

4.1.3. Firestorage

Ce dernier point propose un système de stockage de photos performant dans un espace dédié. Firestorage permet de mettre du contenu lourd rapidement à la disposition des utilisateurs. Afin de prévoir une potentielle utilisation par une grande communauté de l'application, ce système permet de gérer une très grande quantité de données en même temps. Enfin, Firestorage met en place un cloud robuste agissant directement sur les transferts de données de l'utilisateur en mettant en pause en fonction de la connectivité de l'utilisateur.

4.2. Choix techniques Front-end

Edelsten était initialement basé sur la technologie ReactNative utilisant Redux pour partager des données aux travers de toute l'application. L'application est maintenant basée sur Flutter utilisant les Providers pour réaliser le même travail.

Flutter facilite le développement, déployé par Google, ce framework propose une nouvelle façon de développer. Par des widgets il simplifie et propose un développement en block, en widget. Il existe des widgets pour tout type d'interface à réaliser, des icones, des photos, des animations, des transitions, Flutter propose une multitude de widget fluide et efficace.

ReactNative quant à lui est basé sur des packages développé par la communauté et supporté par elle-même. Basé sur un développement open-source, il est compliqué de pouvoir s'appuyer entièrement sur cette technologie pour maintenir l'application dans le temps et proposer une base stable et fiable. Flutter dans sa conception remédie à cela par une licence BSD et non MIT, bien que le framework reste libre le développement est exclusivement réalisé et maintenu par les équipes de Google.

Google utilise Flutter dans ses propres projets comme le Google Assistant et de grands groupes comme Ebay ou Tencent utilise aussi cette technologie. Le support et la durée de vie de Flutter sont assurés et bénéficient de la plus grande attention. Flutter est donc un choix à la fois de confort, de facilité de mise en place, et de stabilité. Ce framework permet enfin l'exploitation d'une nouvelle architecture Flutter en vogue, les providers.

4.2.1. Architecture Provider

Les providers sont un type d'architecture Flutter redéfinissant l'architecture en BloC classique de Flutter. L'architecture provider est basé sur le modèle MVVM (Model, View, ViewModel).
Cette architecture est composée de 3 éléments :

- « Views » représente l'interface utilisateur et possède très peu de logique métier. Chaque View représente un écran avec un ensemble de fonctionnalité. Chacune des View possède également une gestion d'état géré et modifier par un deuxième élément, le ViewModel.

- « ViewModel » est un contrôleur, il gère la logique métier d'une View en plus de son état. Ce système de « Data Binding » améliore l'interface, la dynamise et la rend réactive.

- « Model » a pour rôle de faire le lien entre la base de données et les ViewModels, les Models sont les repositories.

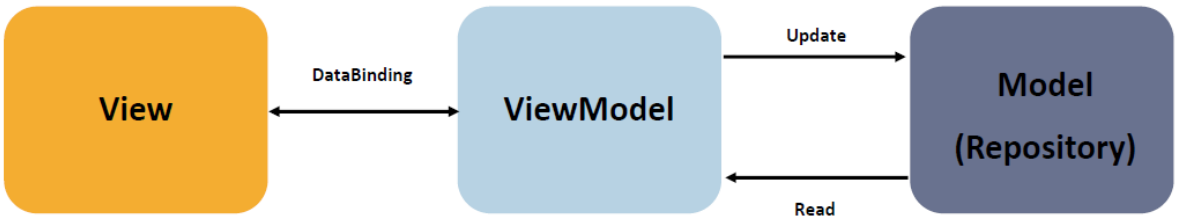


Illustration 5 : Architecture MVVM

Cette architecture nous fournit donc autant de Repository qu'il y a de table dans la base de données, autant de ViewModels et de Views qu'il y a de groupes de fonctionnalités.

L'architecture Provider a besoin de plusieurs éléments pour être mit en place. Elle a besoin d'un fichier **« locator.dart »** afin de créer une instance du plugin **GetIt**. Ce fichier permet d'obtenir une instance partagée des différents Service/Repository/ViewModel. Celui-ci permettra de partager les ressources dans toute l'application et ne pas avoir besoin de faire plusieurs appels d'authentification. Les données de l'authentification seront disponible avec un simple appel d'une des Instances qu'héberge **« locator.dart ».**

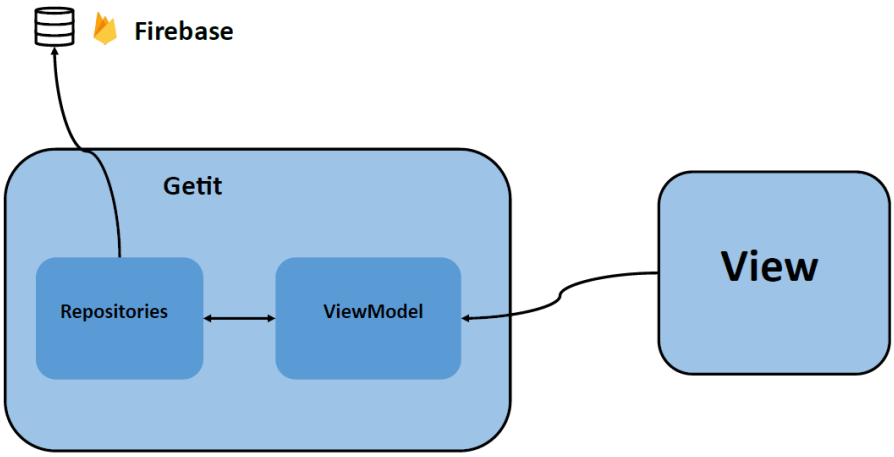


Illustration 6 : Architecture Provider

Voici un exemple plus détaillé du fonctionnement de l'architecture Provider et la façon dont sont accessible les Datas des différentes instances appelées dans les Views et ViewModels. Lors de la redirection vers une route pour obtenir un nouvel écran comme la liste des pierres, la View « ListStone » est appelé. Cette dernière fait appel au ModelView lui correspondant via GetIt. Puis enfin, le ViewModel fait appel aux différents repositories dont il a besoin. De ces repositories des données peuvent être stocké dans un service comme les datas de l'utilisateur dans le service AuthentificationService qui se trouve également dans l'instance de **GetIt**.

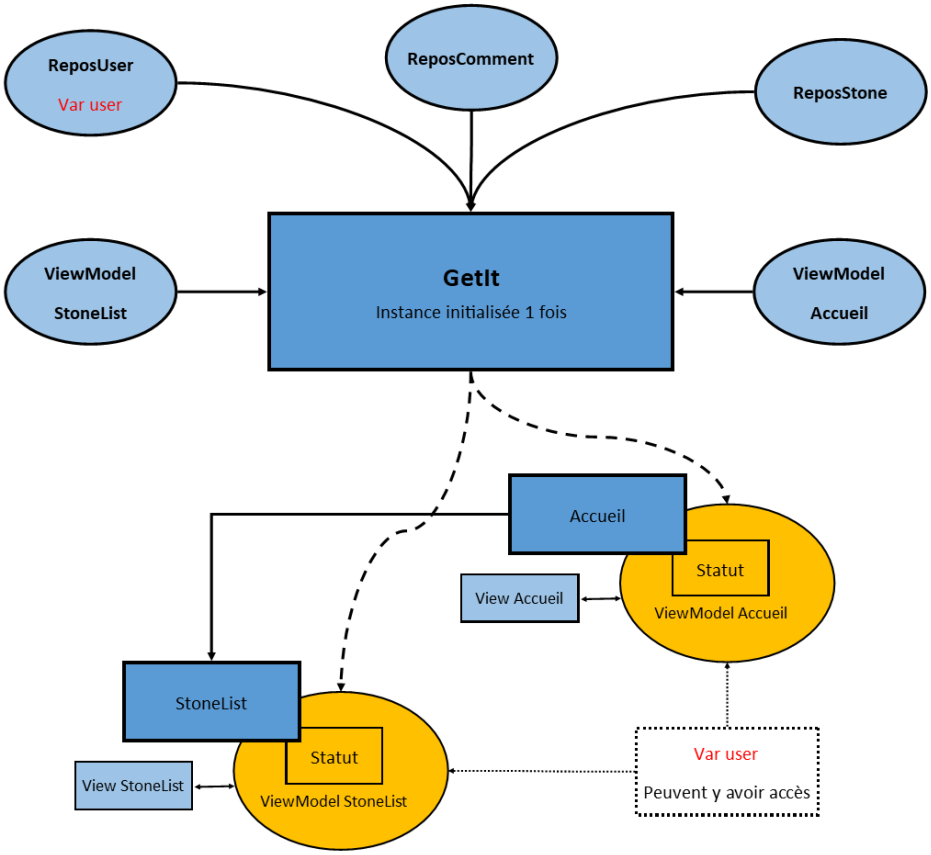


Illustration 7 : Architecture Provider Détaillé

Notre architecture Provider rendra l'arborescence suivante :

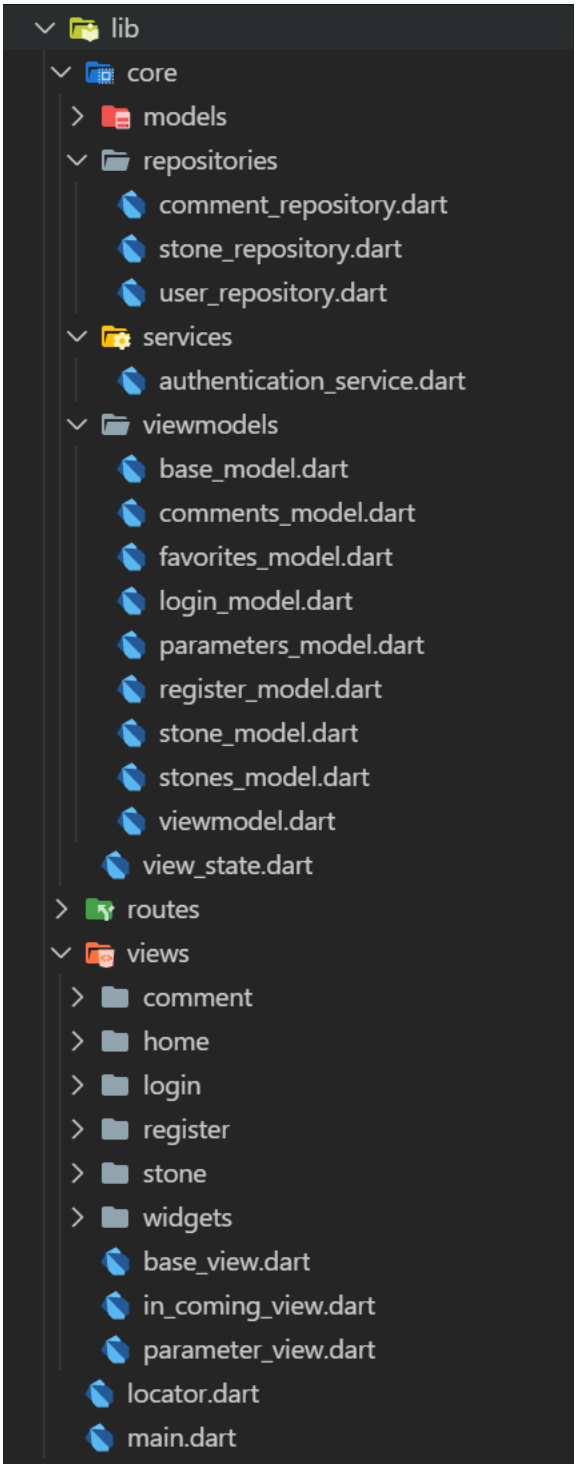


Illustration 8 : Arborescence front

5. Conception graphique

5.1. Brief créatif

Le graphisme de l'application se doit de représenter la communauté utilisant l'application. Elle doit parler a tous tout en restant moderne et épurée. Il est donc recherché une cohésion et un équilibre entre les couleurs de la palette tout en rappelant le logo qui est la première image visible de l'application.

5.2. Aperçu de charte graphique

Illustration 9 : Logo

#080B12	#445269	#BEBEBE	#F4AD31
#021534	#2A356C	#B8C8E2	#A9204C

Illustration 10 : Palette de couleur de l'application

Typographie : Roboto

5.3. Images

En tant qu'application encyclopédique, Edelsten devra illustrer chaque pierre précieuse et semi-précieuse. L'application se doit d'être riche et fournie en photographies de chaque ressource. Celles-ci seront mise à disposition par la communauté venant de leurs photographies personnelles ou de banque d'images libre de droit. Elles seront dans un format optimisé pour l'application, de bonne qualité et de taille convenable pour ne pas ralentir les chargements web et s'afficher correctement sur un appareil mobile.

5.4. Wireframe et Descriptif des champs

Illustration 11 : Accueil

Fenêtre d'accueil proposant 3 choix :

- Se connecter : Redirige vers l'authentification
- S'inscrire : Redirige vers l'inscription
- Continuer en tant que visiteur : Redirige vers l'encyclopédie avec les droits d'un utilisateur « Visiteur »

Illustration 12 : Authentification

- Email : Restriction sur la forme et sur la validité de l'email (partie locale + caractère séparateur (@) + adresse du serveur
- Mot de passe : Restriction sur la forme (6 caractères minimum, 1 chiffre obligatoire, 1 lettre obligatoire, 1 majuscule obligatoire)
- Se connecter : Redirige vers l'encyclopédie
- Mot de passe oublié ? : Envoie un email pour réinitialiser le mot de passe
- S'inscrire : Redirige vers une connexion par un réseau social (Facebook ou Google)
- Pas de compte ? : Redirige vers la page d'inscription
- Continuer en tant que visiteur : Redirige vers l'encyclopédie avec les droits d'un utilisateur « Visiteur »

Illustration 13 : Section encyclopédie

- Recherche : Permet de rechercher dans la base de données de l'encyclopédie une ressource
- Champ cliquable de la ressource : Redirige vers la section résumée de la ressource et les sections liées.
- 4 onglets :
    - Encyclopédie : Redirige vers la section Encyclopédie
    - Espace de notes : Redirige vers l'espace personnel de l'utilisateur
    - Favoris : Redirige vers la section favori de l'application
    - Paramètre : Redirige vers la section paramètre de l'application

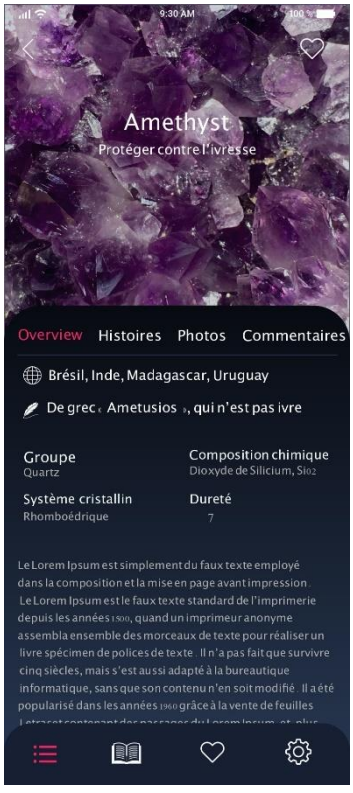


Illustration 14 : Section résumé

- Résumé : Redirige vers la section Résumé de la ressource
- Histoire : Redirige vers la section Histoire de la ressource
- Photos : Redirige vers la section Photos de la ressource
- Commentaire : Redirige vers la section commentaire de la ressource
- Champ Pays : Localisation la pierre
- Champ Etymologie : Origine du nom de la pierre
- Champ Groupe : Famille de la pierre
- Champ Composition chimique : Formule chimique de la pierre
- Champ Système cristallin : Caractéristique de symétrie de la pierre
- Champ Dureté : Dureté de la pierre sur l'échelle de Mohs
- Champ Description : Informations complémentaires descriptives sur la pierre
- 4 onglets :
    - Encyclopédie : Redirige vers la section Encyclopédie
    - Espace de notes : Redirige vers l'espace personnel de l'utilisateur
    - Favoris : Redirige vers la section Favoris de l'application
    - Paramètre : Redirige vers la section Paramètres de l'application

Illustration 15 : Section histoire

- Résumé : Redirige vers la section Résumé de la ressource
- Histoire : Redirige vers la section Histoire de la ressource
- Photos : Redirige vers la section Photos de la ressource
- Commentaire : Redirige vers la section commentaire de la ressource
- Champ Description : Information historique sur la pierre
- 4 onglets :
    - Encyclopédie : Redirige vers la section Encyclopédie
    - Espace de notes : Redirige vers l'espace personnel de l'utilisateur
    - Favoris : Redirige vers la section Favoris de l'application
    - Paramètre : Redirige vers la section Paramètres de l'application

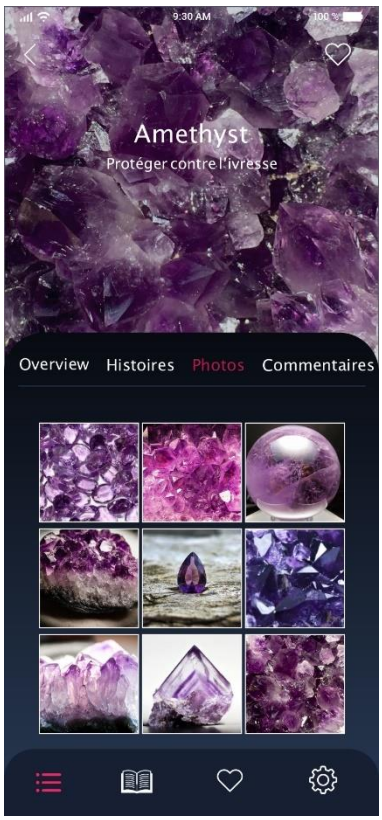


Illustration 16 : Section photos

- Résumé : Redirige vers la section Résumé de la ressource
- Histoire : Redirige vers la section Histoire de la ressource
- Photos : Redirige vers la section Photos de la ressource
- Commentaire : Redirige vers la section commentaire de la ressource
- Champ Photos : Photos de la pierre
- 4 onglets :
    - Encyclopédie : Redirige vers la section Encyclopédie
    - Espace de notes : Redirige vers l'espace personnel de l'utilisateur
    - Favoris : Redirige vers la section Favoris de l'application
    - Paramètre : Redirige vers la section Paramètres de l'application

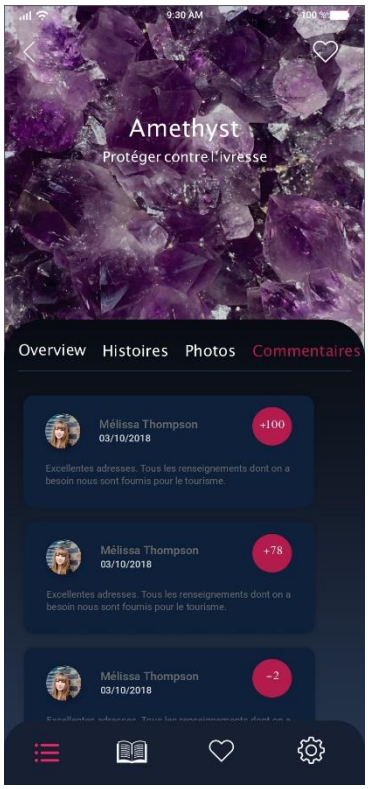


Illustration 17 : Section commentaire

- Résumé : Redirige vers la section Résumé de la ressource
- Histoire : Redirige vers la section Histoire de la ressource
- Photos : Redirige vers la section Photos de la ressource
- Commentaire : Redirige vers la section commentaire de la ressource
- Champ Commentaire :
    - Utilisateur : Nom de l'utilisateur
    - Date : Date de publication du commentaire
    - Like/Dislike : Nombre de like/dislike du commentaire
    - Corps : Corps du commentaire
- 4 onglets :
    - Encyclopédie : Redirige vers la section Encyclopédie
    - Espace de notes : Redirige vers l'espace personnel de l'utilisateur
    - Favoris : Redirige vers la section Favoris de l'application
    - Paramètre : Redirige vers la section Paramètres de l'application