

Soutenance

Home Skolar

Présentation:

HomeSkolar est une association qui met en relation des enfants en difficulté scolaire et des bénévoles à distance.

Elle a pour objectif de permettre à tout élève, où qu'il soit, d'accéder à un soutien scolaire.

Chaque élève inscrit sur le site se verra assigner un tuteur bénévole.

Cahier De charges

- **Authentification**
 - Les élèves et tuteurs doivent avoir la possibilité de s'inscrire, se connecter et gérer leur mots de passe et autres données personnels
- **Communication**
 - Les élèves doivent être en mesure de communiquer avec leur tuteurs et d'épingler des messages
- **Rencontres**
 - Les élèves doivent pouvoir prendre rendez-vous avec leur tuteurs grâce à un calendrier commun
- **Tâches**
 - les élèves vont être notifié au sein de l'application d'une liste de tâches à réaliser pour la prochaine rencontre.

Spécifications fonctionnelles

1- Objectif du système

HomeSkolar est une plateforme web qui permet de mettre en relation un élève en difficulté scolaire et des tuteurs bénévoles.

L'objectif principal est de permettre à chaque élève, où qu'il soit, d'accéder à un soutien scolaire adapté à ses besoins.

Spécifications fonctionnelles

2- Acteurs

- **Élève**
 - S'inscrit sur la plateforme, consulte les disponibilités, demande RDV, accède aux séances, consulte le suivi, échange avec le tuteur via messagerie.
- **Tuteur**
 - S'inscrit, indique ses matières et compétences, définit ses disponibilités, accepte/refuse les RDV, remplit le suivi des séances, échange avec l'élève via messagerie.
- **Administrateur**
 - Gère l'inscription et les comptes des utilisateurs, assigne élèves ↔ tuteurs, consulte statistiques et rapports.

Spécifications fonctionnelles

3- Cas d'usage fonctionnels

- Gestion des utilisateurs
 - Inscription / connexion
 - Gestion du profil
- Gestion des séances
 - Assignment tuteur ↔ élève
 - Gestion des disponibilités
 - Confirmation des RDV
- Suivi pédagogique
 - Accès aux séances (visioconférence)
 - Remplissage du suivi
 - Consultation des suivis

Spécifications fonctionnelles

3- Cas d'usage fonctionnels

- **Messagerie interne**
 - Communication sécurisée entre binôme élève/tuteur
 - Restrictions : seuls les utilisateurs assignés peuvent échanger des messages
- **Notifications et alertes**
 - Demande ou confirmation de RDV
 - Réception d'un message
 - Modifications de planning ou annulations de séance
- **Administration**
 - Gestion des comptes utilisateurs (création, suppression, modification)
 - Assignment des élèves aux tuteurs
 - Consultation de rapports et statistiques : nombre de séances, progression des élèves, taux de disponibilité des tuteurs

Spécifications fonctionnelles

4- Règles métier

- Un élève ne peut être assigné qu'à un seul tuteur à la fois.
- Un tuteur peut gérer plusieurs élèves selon sa disponibilité.
- Les séances ne peuvent être réservées que sur des créneaux disponibles.
- Seuls les binômes assignés peuvent accéder à la messagerie et aux séances
- Les suivis de séance sont visibles uniquement par le tuteur concerné, l'élève et l'administrateur

Spécifications fonctionnelles

5- Critères de qualité

- **Sécurité:** authentification et chiffrement des mots de passe
- **Accessibilité:** interface responsive, compatible desktop et mobile
- **Fiabilité:** sauvegarde régulière des données et notifications fiables.
- **Performance:** accès aux données et visioconférence sans latence excessive

Veille Technologique

- Back-end
 - Spring
- Front-end
 - TailwindCSS
 - React
- Base(s) de Donnée(s)
 - PostgreSQL
 - MongoDB (optionnel)
- Gestion temps réel et messagerie
 - WebSocket avec Spring
 - Firebase Cloud Messaging

Veille Technologique

- Architecture recommandée
 - Backend : Spring Boot + Spring Security + Spring Data JPA
 - Base de données principale: PostgreSQL
 - Base optionnelle pour messages : MongoDB
 - WebSocket pour messagerie temps réel
 - Intégration Jitsi / WebRTC pour visioconférences
 - JWT + HTTPS + RBAC pour sécurité
- Avantages
 - Code robuste et maintenable
 - Communauté Java importante
 - Compatible microservices pour scalabilité future
 - Bonne intégration avec front-end React ou Vue

Veille Technologique

- L'architecture retenue repose sur un modèle microservices dans lequel chaque service possède sa propre base de données, conformément au principe d'autonomie et de découplage.
- Le système utilise une approche polyglot persistence, où différents microservices exploitent des technologies de stockage adaptées à leurs besoins.
- PostgreSQL pour les données relationnelles structurées, et MongoDB pour les données non structurées ou semi-structurées.
- Cette séparation garantit l'indépendance des services, améliore la scalabilité, et évite les dépendances inter-bases de données

Spécifications Techniques

1- Frameworks Java pour le backend

Spring Boot

- Framework mature, très utilisé pour les applications web et microservices. Supporte REST, sécurité, base de données.
- Vs Jakarta EE : Spring Boot plus simple à configurer, plus riche en communauté et documentation.

Source: <https://spring.io/projects/spring-boot>
<https://www.youtube.com/watch?v=gJrjgg1KVL4>

Spécifications Techniques

1- Frameworks Java pour le backend

Spring Security

- Gestion complète de l'authentification et des autorisations, support JWT, OAuth2.
- Vs Apache Shiro : Spring Security mieux intégré à Spring Boot et plus utilisé en entreprise.

Source: <https://spring.io/projects/spring-security>

<https://www.youtube.com/watch?v=nhsdPVXhbHo&vl=en>

Spécifications Techniques

1- Frameworks Java pour le backend

Spring Data JPA

- Simplifie l'accès aux bases de données relationnelles (PostgreSQL/MySQL).
- Simplifie l'accès aux bases de données relationnelles (PostgreSQL/MySQL)

Source: <https://spring.io/projects/spring-data-jpa>

<https://youtu.be/dKK2dZVLFug?si=hSdH2mSi1KHm3wcr>

Spécifications Techniques

2- Base de données

PostgreSQL

- Base relationnelle robuste, ACID compliant, idéale pour gérer comptes, RDV, suivis.
- Vs MySQL : PostgreSQL plus riche en fonctionnalités (JSONB, transactions complexes).

Source: <https://www.postgresql.org/>

<https://www.oracle.com/fr/database/definition-postgresql/>

Spécifications Techniques

2- Base de données

MongoDB

- Pour stocker l'historique des messages et logs de sessions si besoin de flexibilité.
- Vs PostgreSQL : MongoDB plus flexible pour documents JSON, PostgreSQL plus structuré.

Source: <https://www.mongodb.com/>

<https://www.youtube.com/watch?v=dKK2dZVLFug>

Spécifications Techniques

3-Front-End

- Rapidité de développement
- Cohérence visuelle & design system
- Flexibilité technique (React obligatoire)
- vs Bootstrap qui est un framework component-based avec des UI prédéfinies mais un bundle CSS plus lourd

Source: <https://tailwindcss.com/>

<https://youtu.be/mr15Xzb1Ook?si=tRbTtPlbZ5Dhp5cz>

Spécifications Techniques

4- Gestion temps réel et messagerie

WebSocket avec Spring

- Supporte messagerie temps réel pour chat élève ↔ tuteur et notifications instantanées.
- Supporte messagerie temps réel pour chat élève ↔ tuteur et notifications instantanées

Source: <https://spring.io/guides/gs/messaging-stomp-websocket>

<https://youtu.be/TywlS9iAZCM?si=pqY6TQgxcy1UfmgY>

Spécifications Techniques

4- Gestion temps réel et messagerie

Firebase Cloud Messaging

- Notifications push multi-plateformes (optionnel pour mobile).
- Vs WebSocket : FCM gère push hors app, WebSocket pour temps réel in-app.

Source: <https://firebase.google.com/docs/cloud-messaging>

<https://youtu.be/vAoB4VbhRzM?si=PWIsbdLOnvAJxNHJ>

Spécifications Techniques

5- Visioconférence

Jitsi Meet

- Open source, intégrable via iframe ou API REST/SDK, pas de dépendance à un service payant.
- Vs Zoom SDK : Jitsi gratuit, respect vie privée, Zoom plus fiable mais payant.

Source: <https://jitsi.org/>

<https://marketplace.microsoft.com/en-us/product/virtual-machines/cloud-infrastructure-services.jitsi-meet?tab=Overview>

Diagramme de classes

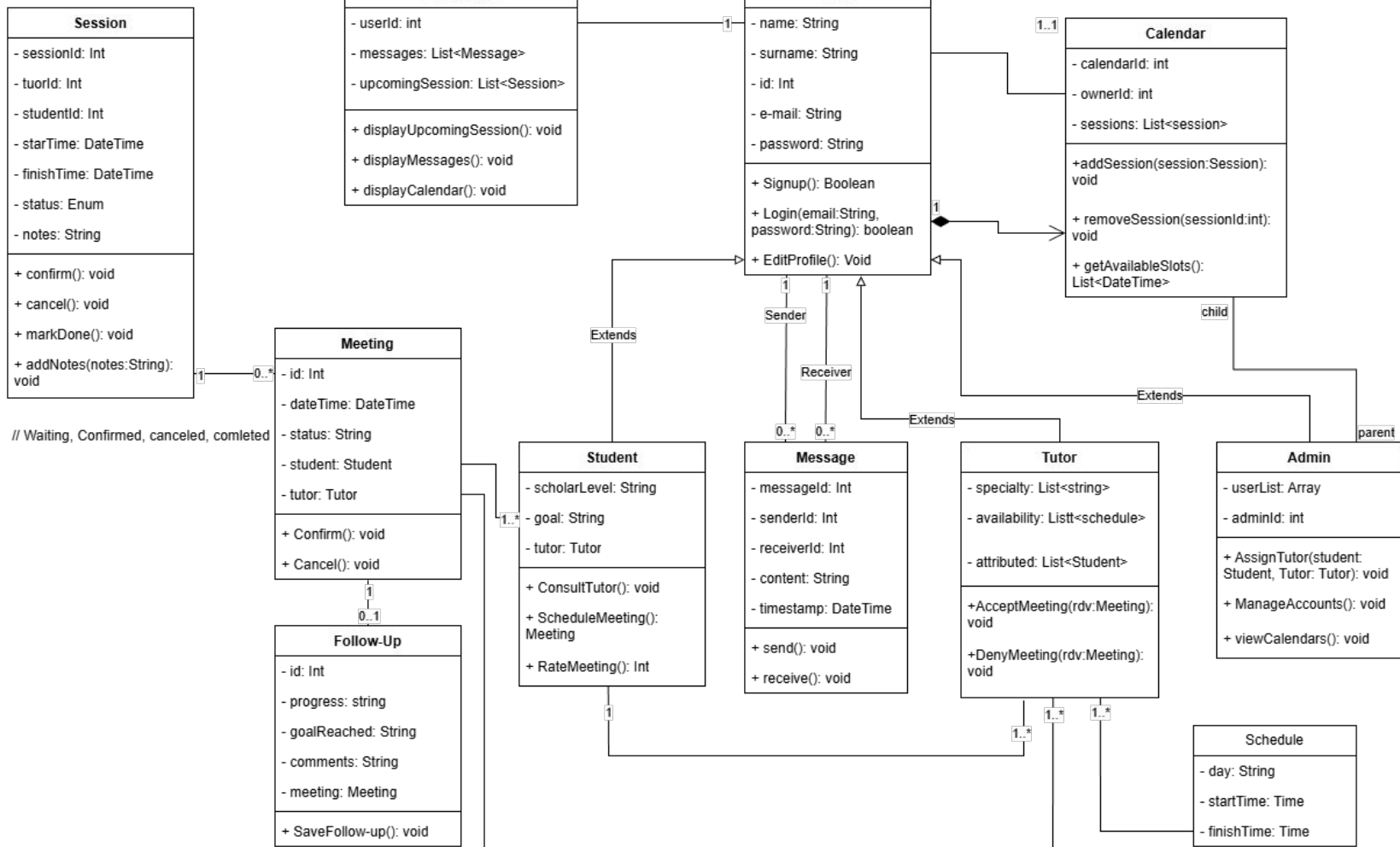
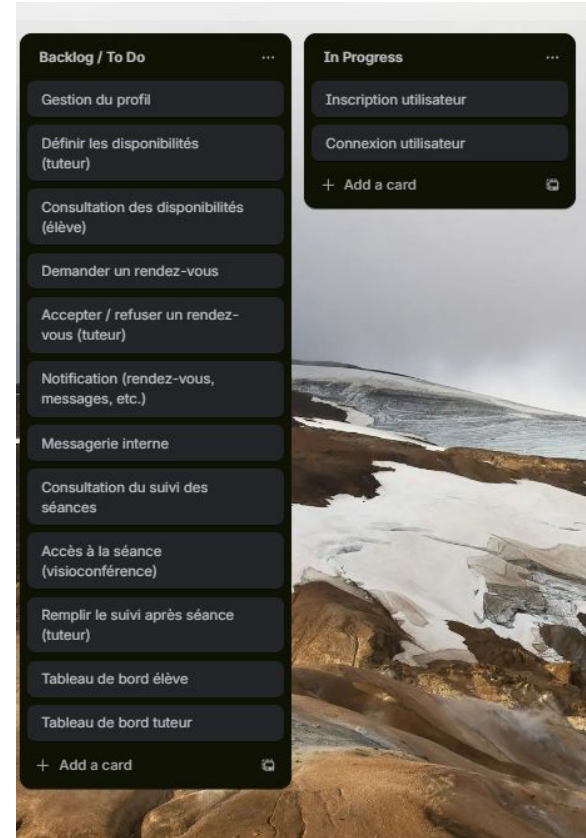
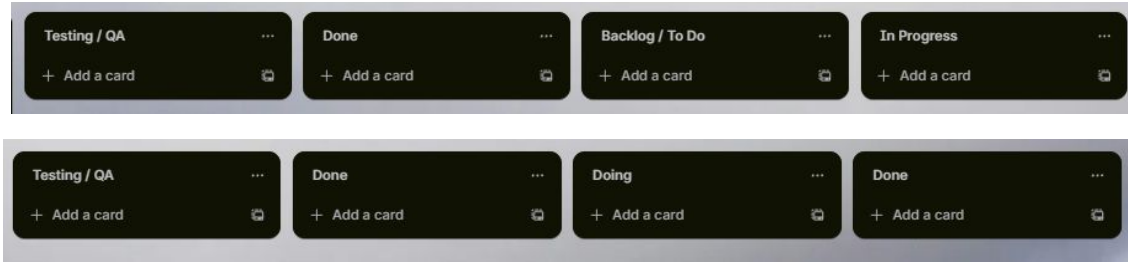


Tableau kanban

**Vide a titre d'exemple*



Merci

Fin de la présentation

Cordialement