Cahier des charges

Application HomeSkolar

Spécifications fonctionnelles

A contexte:

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 voit assigner un tuteur bénévole. Ce dernier a pour objectif de soutenir l'élève dans son apprentissage, à travers de courts rendez-vous prévus chaque semaine. Durant ces rendez-vous, le bénévole aide l'élève à réaliser ses devoirs et à s'organiser. Pour faciliter les interactions et la communication, HomeSkolar souhaite créer un nouveau site web qui doit permettre diverses fonctionnalités.

Specification fonctionnelles

Front-end (Interface utilisateur) :

- Le site doit être conçu avec une interface conviviale et responsive, permettant une utilisation fluide sur différentes tailles d'écran (ordinateurs de bureau, tablettes, smartphones).
- Utilisation de technologies front telles que HTML, CSS et JavaScript pour la structure, la mise en forme et l'interaction.
- Framework JavaScript tel que React, Angular ou Vue.js peut être utilisé pour faciliter le développement et l'organisation du code.

Page de connexion :

- Une page de connexion sécurisée doit être mise en place, permettant aux utilisateurs (élèves et tuteurs) de s'identifier avec leurs identifiants uniques (par exemple, adresse e-mail et mot de passe).
- Possibilité pour les utilisateurs de récupérer leur mot de passe en cas d'oubli (fonction "Mot de passe oublié").
- Si l identifiant fournis ne correspond a aucun identifiant de la base de données l utilisateur est redirige vers la page inscription

Page inscription:

- Une page inscription permettant aux utilisateurs n ayant pas de compte de s en créer un
- L identifiant sera stoïque en clair dans la base de données mais le mot de passe sera crypte

Page tableau de bord :

- Une page principale depuis laquelle il est possible d accéder a toute les pages du site et de voir son calendrier .

Base de données :

- Une base de données doit être utilisée pour stocker les informations des utilisateurs (élèves et tuteurs) et d'autres données importantes liées aux tâches et aux

événements.

- Les données sensibles, telles que les mots de passe, doivent être stockées de manière sécurisée à l'aide de techniques de hachage et de salage.

Authentification et autorisation :

- Un système d'authentification robuste doit être mis en place pour garantir que seuls les utilisateurs puissent accéder aux fonctionnalités spécifiques du site.
- Différents niveaux d'autorisation peuvent être mis en place pour les élèves, les tuteurs et les administrateurs du site.

Communication entre élèves et tuteurs :

- Une messagerie intégrée doit être mise en place pour permettre aux élèves et aux tuteurs de communiquer entre eux de manière sécurisée et conviviale.
- Possibilité d'envoyer des messages individuels d épingler des messages et de consulter tout les messages épinglé .

Consultation des tâches à réaliser :

- Chaque élève doit avoir accès à une liste de tâches à réaliser, attribuées par les tuteurs.
- Chaque utilisateur doit pouvoir se rajouter des taches et y avoir accès dans la page taches.
- Les tuteurs peuvent ajouter, modifier ou supprimer des tâches, et les élèves peuvent marquer les tâches comme terminées.

Calendrier des événements :

- Le site doit comporter un calendrier intégré qui affiche les différents événements prévus, tels que les sessions de tutorat, les événements , etc.
- Les événements peuvent être ajoutés, modifiés ou supprimés par les tuteurs, et les élèves doivent pouvoir consulter le calendrier pour prendre connaissance des événements à venir.

B Cas d’application et profils utilisateurs :

1 authentification:

1.1 inscription:

un nouvel utilisateur pourra créer un compte sur la page inscription en renseignant les différents champs du formulaire et en les validants en cliquant sur un bouton « créer mon compte » il sera redirige sur la page « connexion »

1.2 connexion :

un utilisateur possédant un compte pourra se connecter depuis la page connexion en renseignant les champs identifiant/mot_de_passe . Il pourra valider sa saisie sur un bouton « connexion » , il sera redirige sur la page « tableau de bord »

1.3 gestion du mot de passe/données personnels:

l’utilisateur aura accès a son profil sur la page « tableau de bord » en cliquant sur l onglet « profil » il sera redirige sur la page « profil » depuis lequel il pourra modifier son mot de passe et récupérer ses données personnels

2 Communication:

2.1 depuis la page « tableau de bord » l utilisateur pourra se rendre sur une page communication depuis lequel il pourra discuter avec son tuteur/eleve .

2.2 depuis la page communication il doit être possible pour l utilisateur d épingler un message .

2.3 l utilisateur doit pouvoir accéder a tout les message épinglé d une conversation.

2.4 l utilisateur doit être notifie pour tout message non lu .

3 rencontre tuteur élève :

3.1 depuis la page « tableau de bord » l utilisateur pourra se rendre sur une page « calendrier ».

3.2 cette page correspond a une page de calendrier classique sur laquelle s’affichent les différents événements et rendez-vous de l’utilisateur.

3.3 prendre un rdv ?

(3.5 en cliquant sur un jour avoir les rdv du jour qui s affichent et pouvoir les rejoindre)

4 taches :

4.1 depuis la page « tableau de bord » l utilisateur pourra se rendre sur une page « tache »

4.2 un tuteur pourra depuis cette page donner a son élèves les différentes taches a réaliser pour la prochaine rencontre

4.3 un utilisateur doit pouvoir visualiser les différentes taches qu'il doit réaliser

4.4 un élève doit être notifié pour chaque nouvelle tache

4.5 un utilisateur doit pouvoir se donner une tache a lui même depuis cette même page

Différents types utilisateur :

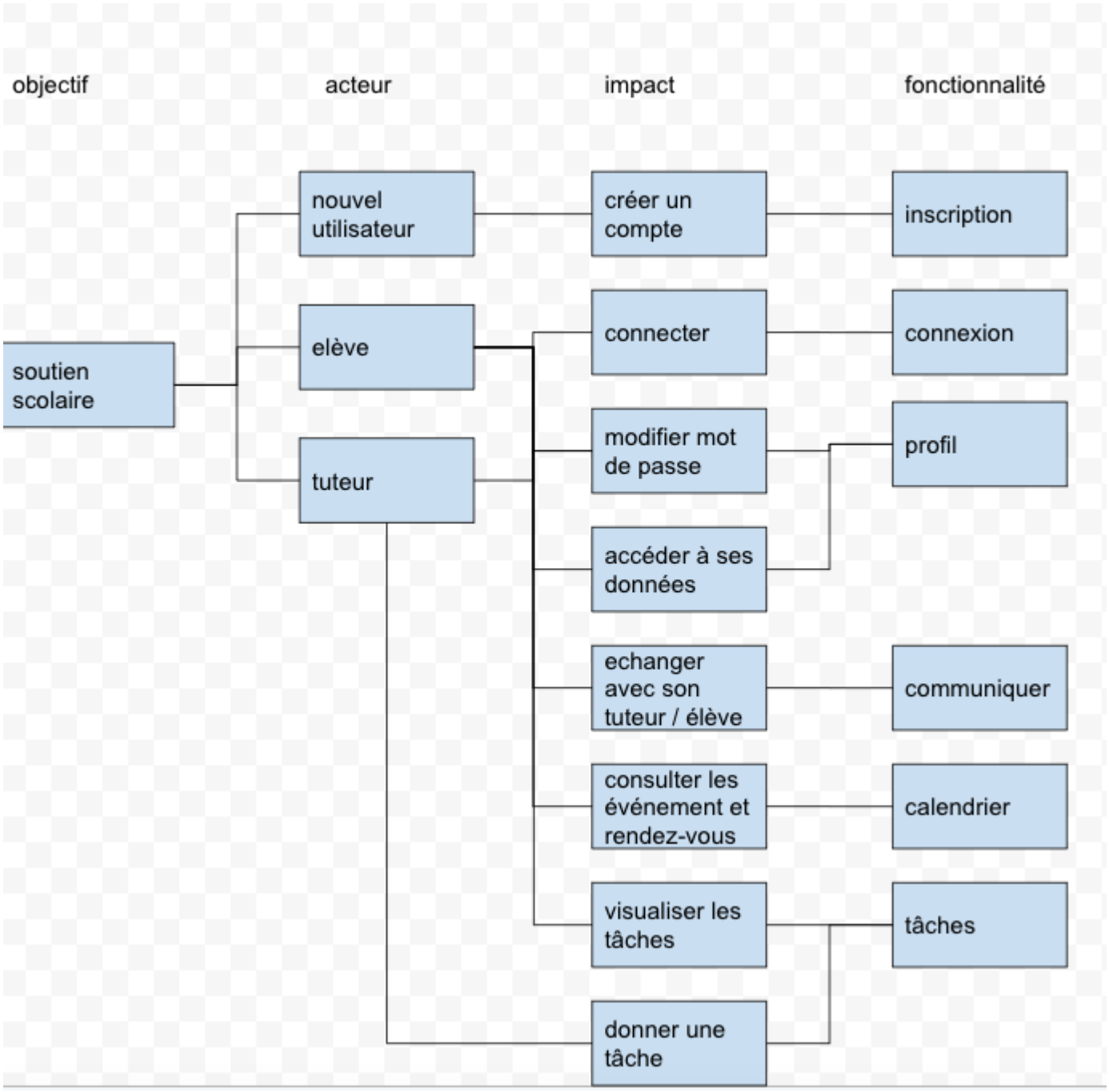
1 nouvel utilisateur

il s agit d un utilisateur n ayant pas encore de compte Chew homeskolar il n a accès qu'aux pages connexion et inscription

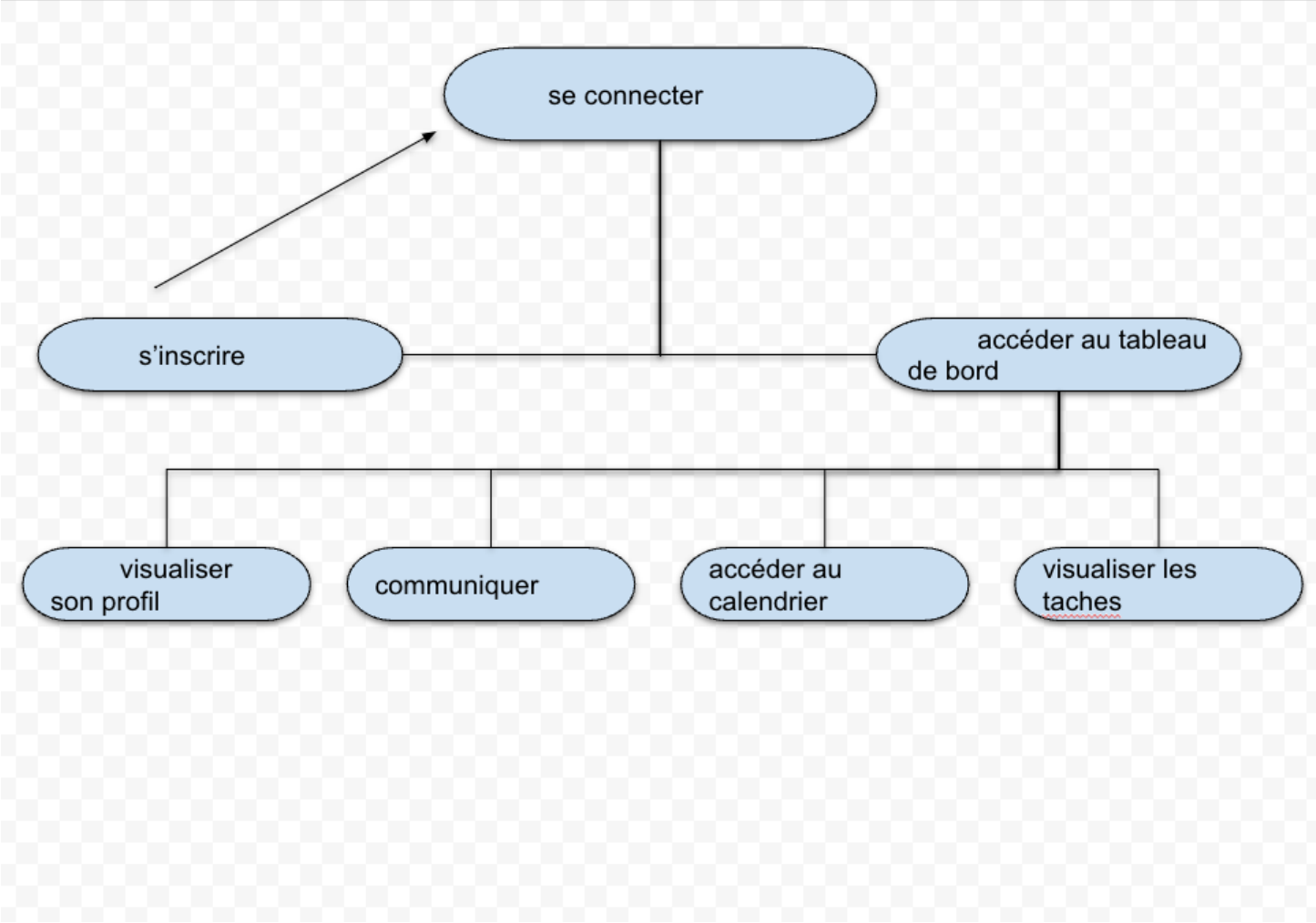
2 utilisateur :

il s agit d un utilisateur ayant un compte chez homeskolar il a accès aux pages identification , connexion , tableau de bord , profil , communication , calendrier et tache cette catégorie regroupe 2 types d utilisateur :l élève et le tuteur leurs utilisations particulière sont détaillés plus haut

C Périmètre fonctionnel :



D Arborescence :



E Description des pages:

Page d'Accueil / Identification:

- Une page d'accueil simple avec un formulaire d'identification (connexion) pour les utilisateurs enregistrés.
- Un bouton "S'inscrire" redirigeant vers la page d'inscription.

Page d'Inscription:

- Un formulaire d'inscription avec des champs pour le nom, l'adresse e-mail, le mot de passe et la confirmation du mot de passe.
- Un bouton "S'inscrire" pour créer un nouveau compte.

Page Mot de Passe Oublié:

- Un formulaire pour saisir l'adresse e-mail associée au compte.
- Un bouton "Envoyer le lien de réinitialisation" pour envoyer un e-mail de réinitialisation du mot de passe.

Page Tableau de Bord:

- Un en-tête avec le nom de l'utilisateur connecté et un bouton de déconnexion.
- Une barre de navigation avec des onglets pour accéder aux différentes fonctionnalités (Profil, Calendrier, Discussion, Tâches).
- Un calendrier affichant les événements à venir (tutorats et autres événements).
- Une section avec des statistiques ou informations spécifiques à l'utilisateur.

Page Calendrier:

- Un calendrier interactif affichant les rendez-vous et les événements à venir.
- Des boutons pour passer entre les vues mensuelle, hebdomadaire et quotidienne du calendrier.
- Des icônes ou codes de couleur pour distinguer différents types d'événements.

Page Discussion:

- Une liste de discussions disponibles.
- Les utilisateurs peuvent cliquer sur une discussion pour afficher les messages et y répondre.

Page Profil:

- Une section affichant les informations personnelles de l'utilisateur (nom, e-mail, etc.).
- Un formulaire pour modifier les informations du profil.
- Un bouton pour mettre à jour les modifications.

Page Tâches:

- Une liste de tâches à accomplir.
- Des options pour ajouter de nouvelles tâches, supprimer des tâches et marquer des tâches comme terminées.

F Users Story

1 En tant que nouvel utilisateur,

- Je veux m'inscrire en fournissant mon nom, mon adresse e-mail et un mot de passe pour créer un compte sur homeSkolar.

2 En tant qu'élève ou tuteur,

- Je veux pouvoir me connecter à mon compte en utilisant mon identifiant et mon mot de passe pour accéder à mes fonctionnalités spécifiques.

3 En tant qu'élève,

- Je veux pouvoir consulter un tableau de bord pour accéder a toute les fonctionalitees du site et visualiser mon calendrier.

4 En tant qu'élève,

- Je veux accéder à une page de discussion pour communiquer avec d'autres utilisateurs et épingler des messages .

5 En tant qu'élève,

- Je veux pouvoir consulter mon profil pour voir mes informations personnelles et les modifier si nécessaire.

6 En tant qu'élève,

- Je veux voir les événements à venir (événements et tutorats) dans un calendrier .

7 En tant qu'élève,

- Je veux ajouter des tâches à accomplir et les visualiser sur une page dédiée pour mieux organiser mon travail scolaire.

8 En tant que tuteur,

- Je veux avoir les mêmes fonctionnalités qu'un élève, y compris l'accès au profil, à la discussion, au tableau de bord, au calendrier et à la gestion des tâches.

9 En tant que tuteur,

- Je veux pouvoir planifier des tutorats avec mes élèves pour les aider dans leur apprentissage.

10 En tant que tuteur,

- Je veux pouvoir attribuer des tâches spécifiques à mes élèves et suivre leur progression.

Spécifications techniques

Frontend (React) :

Langages : JavaScript (ES6+), HTML, CSS.

Bibliothèque : React.js avec Redux pour la gestion de l'état global de l'application.

Gestionnaire de paquets : npm ou yarn.

Utilisation de composants réutilisables pour assurer la modularité et la maintenabilité du code.

Intégration de Redux pour la gestion de l'état global de l'application.

Utilisation de React Router pour la gestion des routes et de la navigation entre les différentes pages.

Implémentation d'une interface utilisateur réactive et conviviale.

Backend (Django) :

Langage : Python.

Framework : Django pour la gestion des routes, des middlewares, et la gestion de la base de données.

Base de données : Utilisation de Django ORM avec une base de données PostgreSQL pour le stockage des données.

Cryptage des mots de passe avant le stockage dans la base de données.

Utilisation de JSON Web Token (JWT) pour l'authentification et l'autorisation des utilisateurs.

Gestion des e-mails : Intégration de Django Email pour envoyer des e-mails aux utilisateurs.

Validation des données d'entrée pour prévenir les attaques par injection et assurer la sécurité.

Base de données (PostgreSQL) :

Utilisation de PostgreSQL pour le stockage des données.

Modélisation des tables de la base de données pour les utilisateurs, les tutorats, les tâches, et les discussions.

Indexation appropriée pour améliorer les performances des requêtes.

Assurer la sécurité de la base de données en configurant l'accès et les autorisations correctement.

Sécurité :

Implémentation d'un système d'authentification basé sur JWT pour sécuriser les accès aux ressources.

Cryptage des mots de passe avant le stockage dans la base de données.

Validation appropriée des données d'entrée pour prévenir les attaques par injection (XSS, SQL injection, etc.).

Protection contre les attaques par force brute en mettant en place des politiques de verrouillage de compte et des contrôles de débit.

Fonctionnalités spécifiques :

Implémentation d'un mécanisme d'inscription avec vérification de l'unicité de l'adresse e-mail.

Envoi d'e-mails aux utilisateurs pour la réinitialisation du mot de passe lorsqu'ils cliquent sur le bouton "Mot de passe oublié".

Gestion des rendez-vous (tutorats) sur le serveur Django et stockage dans la base de données.

Gestion des tâches sur le serveur Django et stockage dans la base de données.

Déploiement :

Le frontend (React) et le backend (Django) seront déployés sur des serveurs distincts ou, idéalement, sur une plateforme de cloud computing (AWS, Azure, etc.).

Configuration des serveurs pour une bonne performance, sécurité et évolutivité**.**

Tests et Qualité du Code :

Écriture de tests unitaires et de tests d'intégration pour assurer la stabilité et la qualité du code.

Utilisation d'outils de vérification du code (linters) pour maintenir des normes de codage cohérentes.

Gestion de Version :

Utilisation d'un système de contrôle de version tel que Git pour gérer le code source et faciliter la collaboration entre les développeurs.

Spécification page par page :

Frontend (React) :

Page d'Accueil / Identification :

- Création d'une page d'accueil avec un formulaire de connexion pour les utilisateurs enregistrés.
- Utilisation de composants React pour la mise en page et la gestion des interactions.
- Gestion des erreurs de saisie pour afficher des messages d'erreur pertinents.
- Intégration d'un bouton "S'inscrire" redirigeant vers la page d'inscription.

Page d'Inscription :

- Création d'une page d'inscription avec un formulaire comprenant des champs pour le nom, l'adresse e-mail, le mot de passe et la confirmation du mot de passe.
- Utilisation de composants React pour la mise en page et la gestion des interactions.
- Validation des champs du formulaire côté client pour s'assurer que les données sont correctes avant l'envoi au serveur.
- Affichage de messages d'erreur en cas de saisie incorrecte.
- Intégration d'un bouton "S'inscrire" pour envoyer les informations au serveur et créer un nouveau compte.

Page Mot de Passe Oublié :

- Création d'une page pour le processus de réinitialisation du mot de passe.
- Utilisation d'un formulaire pour saisir l'adresse e-mail associée au compte.
- Intégration d'un bouton "Envoyer le lien de réinitialisation" pour envoyer une demande de

réinitialisation au serveur.

- Affichage d'un message de confirmation à l'utilisateur après l'envoi du lien de réinitialisation.

Page Tableau de Bord :

- Création d'une page de tableau de bord avec une barre de navigation en haut pour accéder aux différentes fonctionnalités.
- Utilisation de composants React pour afficher les statistiques ou les informations spécifiques à l'utilisateur.
- Intégration d'un calendrier interactif pour afficher les événements à venir.
- Gestion des onglets pour rediriger l'utilisateur vers les pages correspondantes en fonction de son choix.

Page Calendrier :

- Création d'une page de calendrier interactif avec différentes vues (mensuelle, hebdomadaire, quotidienne) pour afficher les rendez-vous et les événements à venir.
- Utilisation de composants React pour la mise en page et la gestion des interactions du calendrier.
- Utilisation d'icônes ou de codes de couleur pour distinguer différents types d'événements.
- Gestion des interactions utilisateur pour ajouter, modifier ou supprimer des événements du calendrier.

Page Discussion :

- Création d'une page de discussion avec une liste de discussions disponibles.
- Utilisation de composants React pour afficher les messages de chaque discussion.
- Gestion des interactions utilisateur pour afficher les détails d'une discussion sélectionnée et pour envoyer de nouveaux messages**.**

Page Profil:

- Frontend :
- Création d'une page de profil avec les informations personnelles de l'utilisateur.
- Utilisation de composants React pour afficher et éditer les informations du profil.
- Gestion des interactions utilisateur pour mettre à jour les informations du profil.

Page Tâches:

- Frontend :
- Création d'une page de gestion des tâches avec une liste de tâches à accomplir.
- Utilisation de composants React pour afficher les détails de chaque tâche.
- Gestion des interactions utilisateur pour ajouter, supprimer et marquer les tâches comme terminées.

Backend (DJANGO) :

Page d'Accueil / Identification (Backend - Django) :

- Mise en place de vues pour la gestion de la connexion (authentification) des utilisateurs enregistrés.
- Vérification des informations de connexion côté serveur (adresse e-mail, mot de passe).
- Utilisation de JWT pour générer un token d'authentification pour les utilisateurs connectés**.**

Page d'Inscription (Backend - Django) :

- Mise en place de vues pour gérer l'inscription des nouveaux utilisateurs.
- Vérification que l'adresse e-mail fournie n'est pas déjà utilisée par un autre compte.
- Cryptage du mot de passe avant de le stocker dans la base de données.
- Envoi d'une confirmation d'inscription par e-mail.

Page Mot de Passe Oublié (Backend - Django) :

- Mise en place de vues pour gérer le processus de réinitialisation du mot de passe.
- Génération d'un token de réinitialisation de mot de passe.
- Envoi d'un e-mail contenant le lien de réinitialisation avec le token.

Page Tableau de Bord (Backend - Django) :

- Mise en place de vues pour récupérer les informations spécifiques à l'utilisateur à afficher sur le tableau de bord.
- Requêtes pour récupérer les événements à venir (tutorats et autres événements) à afficher dans le calendrier.

Page Calendrier (Backend - Django) :

- Mise en place de vues pour gérer les opérations CRUD (Créer, Lire, Mettre à jour, Supprimer) pour les événements du calendrier (tutorats et autres événements).
- Validation des données envoyées par l'utilisateur avant de les enregistrer dans la base de données.

Page Discussion (Backend - Django) :

- Mise en place de vues pour récupérer les discussions disponibles et les messages associés à chaque discussion.
- Vues pour gérer l'ajout de nouveaux messages dans une discussion existante**.**

Page Profil (Backend - Django) :

- Mise en place de vues pour récupérer les informations du profil de l'utilisateur et les mettre à jour**.**

Page Tâches (Backend - Django) :

- Mise en place de vues pour gérer les opérations CRUD (Créer, Lire, Mettre à jour, Supprimer) pour les tâches de l'utilisateur.
- Validation des données envoyées par l'utilisateur avant de les enregistrer dans la base de données.

•

Ces spécifications techniques servent de guide pour le développement de l'application "homeSkolar" et doivent être adaptées en fonction des besoins spécifiques du projet et des contraintes techniques.

Diagramme de classes

Le diagramme doit utiliser la nomenclature UML. Inclure chaque classe obligatoire pour l'application. Assurez-vous qu'il est extensible pour favoriser données.

Utilisateur : id, nom, adresseEmail, motDePasse : getId(), getNom(), getAdresseEmail()

Eleve : listeTaches, listeDiscussions listeEvenement: ajouterTache(tache), supprimmerTache(Tache), getListeTaches(), getListeDiscussions() , getListeEvenement()

Tuteur : listeTutorats : planifierTutorat(tutorat), attribuerTache(eleve, tache), getListeTutorats()

Tutorat : id, date, sujet, eleve, tuteur : getId(), getDate(), getSujet(), getEleve(), getTuteur()

Tache : id, description, dateLimite, statut : getId(), getDescription(), getDateLimite(), getStatut(), setStatut(statut)

Calendrier : id, evenements : ajouterEvenement(evenement), supprimerEvenement(evenement), getListeEvenements()

Evenement : id, date, titre, description : getId(), getDate(), getTitre(), getDescription()

Discussion : id, eleve, tuteur, messages : getId(), getEleve(), getTuteur(), getMessages(), ajouterMessage(message)

Message : id, auteur, contenu, date : getId(), getAuteur(), getContenu(), getDate()

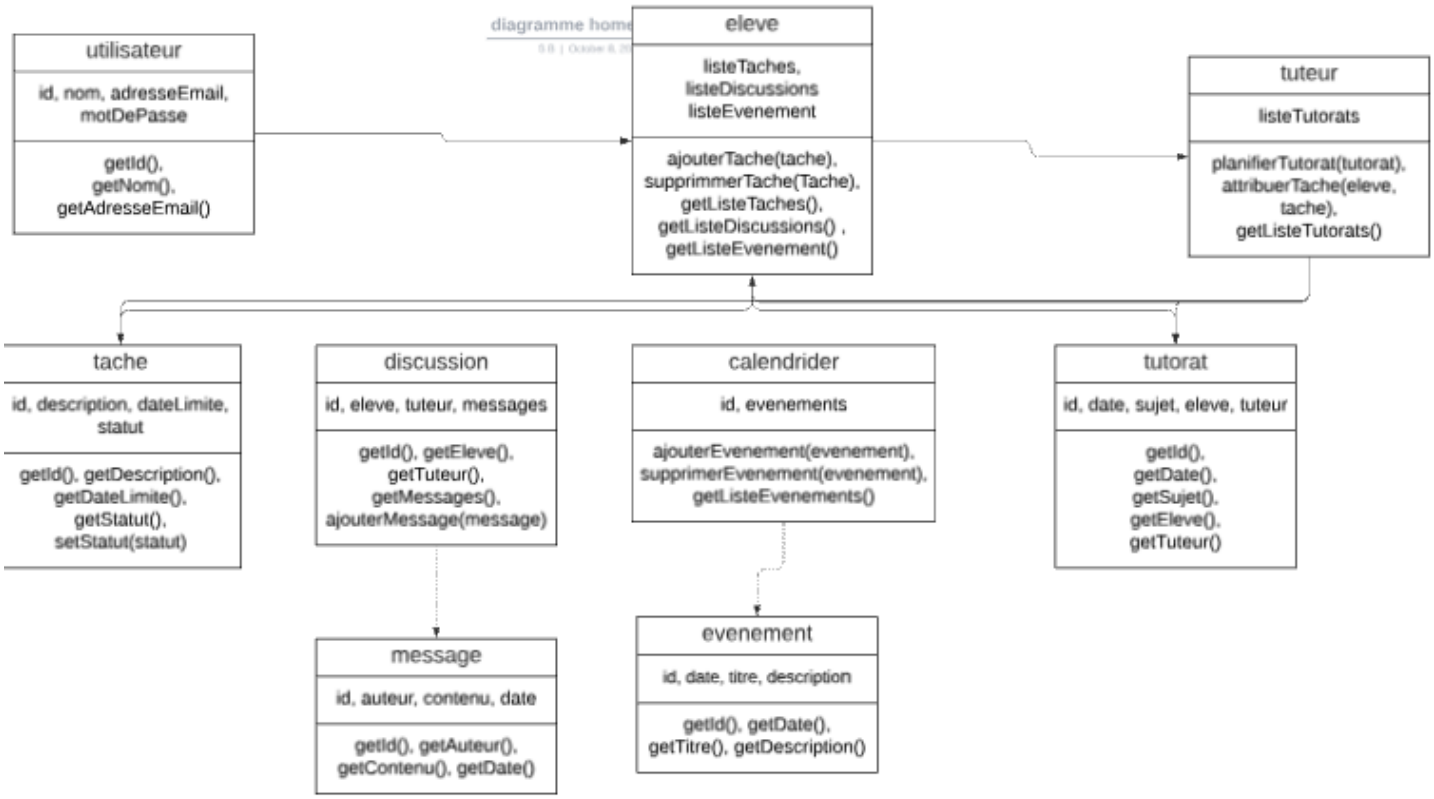


diagramme home
utilisateur
id, nom, adresseEmail, motDePasse
getId(), getNom(), getAdresseEmail()
eleve
listeTaches, listeDiscussions listeEvenement
ajouterTache(tache), supprimmerTache(Tache), getListeTaches(), getListeDiscussions() , getListeEvenement()
tuteur
listeTutorats
planifierTutorat(tutorat), attribuerTache(eleve, tache), getListeTutorats()
tache
id, description, dateLimite, statut
getId(), getDescription(), getDateLimite(), getStatut(), setStatut(statut)
discussion
id, eleve, tuteur, messages
getId(), getEleve(), getTuteur(), getMessages(), ajouterMessage(message)
calendrider
id, evenements
ajouterEvenement(evenement), supprimerEvenement(evenement), getListeEvenements()
tutorat
id, date, sujet, eleve, tuteur
getId(), getDate(), getSujet(), getEleve(), getTuteur()
message
id, auteur, contenu, date
getId(), getAuteur(), getContenu(), getDate()
evenement
id, date, titre, description
getId(), getDate(), getTitre(), getDescription()