Dimeck Raphael
Fradet Nael
Guenezan Théo
Makloufi Yanis
Les Prog' Râleurs Q2

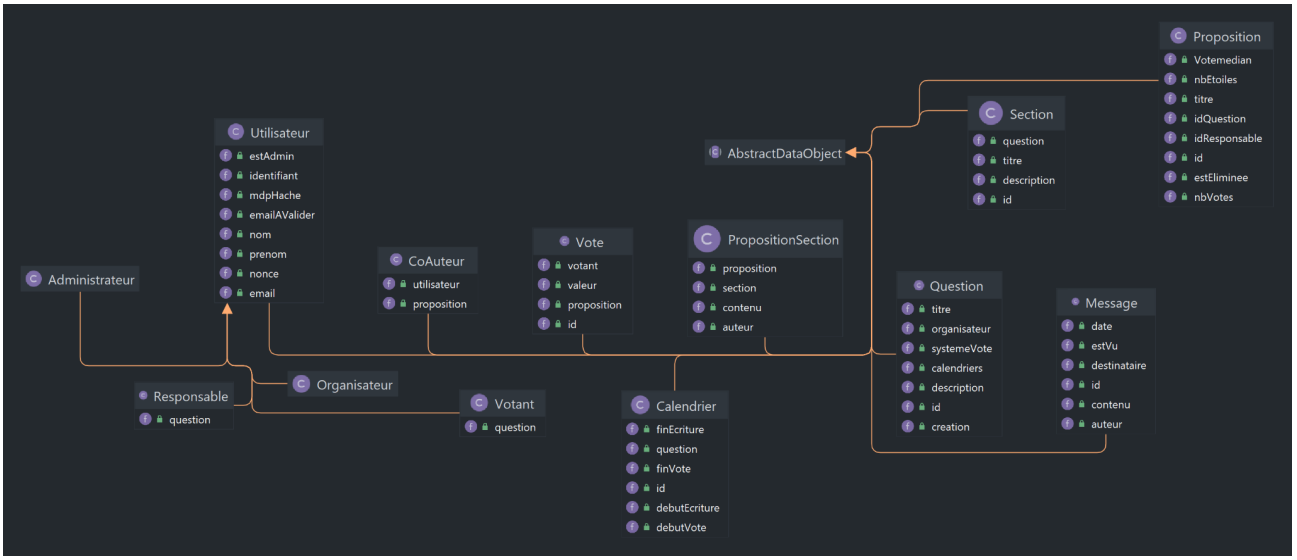
S3.A.01
Parcours RACDV

Cahier de conception

Table des matières :

Diagrammes de classe

Diagramme de classe : Objets

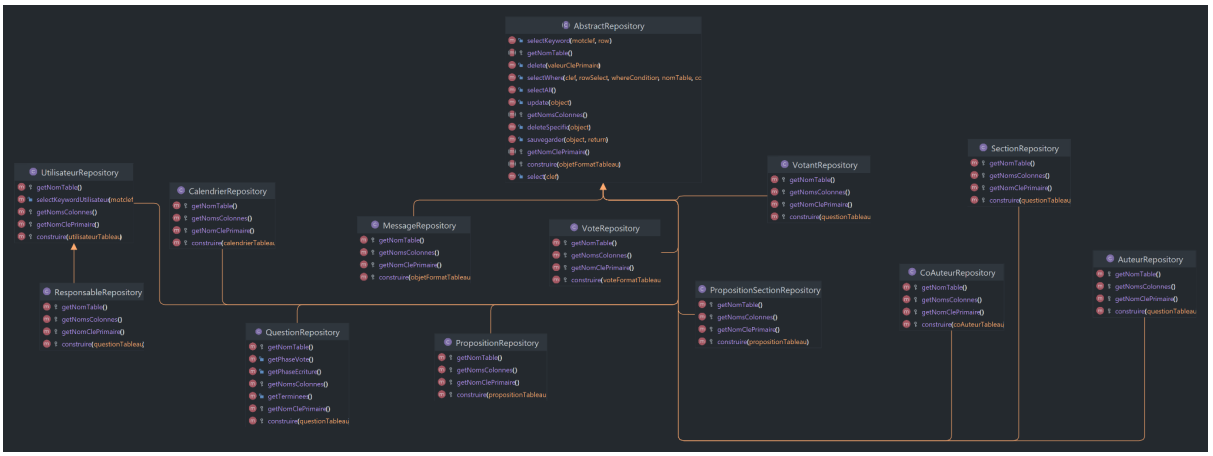


Ce diagramme de classe représente tous les objets, ils héritent de la classe abstraite “AbstractDataObject”. Elle possède une méthode : “formatTableau” qui permet de préparer les requêtes dans le repository principal.
Plus précisément, cette méthode est redéfinie dans toutes les classes des objets et retourne un tableau avec le nom des colonnes de la table correspondante ainsi qu’un tag.
Pour chaque colonne, elle associe la valeur de l’attribut de l’objet concerné.
Par exemple, lorsqu’on souhaite enregistrer un vote dans la base de donnée, la fonction “formatTableau” retournera : return new array(

```
"idvotantTag" => $this->votant->getIdentifiant(),
"idpropositionTag" => $this->proposition->getId(),
“valeurvoteTag" => $this->valeur
```

Ces classes contiennent aussi des fonctions très utiles qui ne sont pas représentées sur le schéma pour des soucis de lisibilité.

Diagramme de classe : Repository



Ci-dessus, un diagramme de classe qui représente tous les repository de notre projet. Les repository sont des classes qui permettent de faire le lien entre les objets (Question, proposition, vote etc..), et les tables de la base de données.

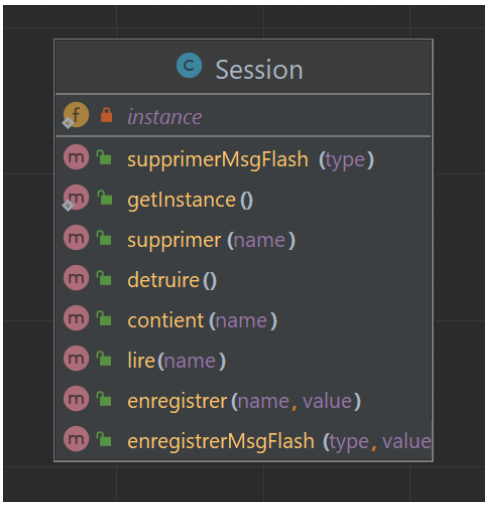
Toutes ces classes héritent de la classe “AbstractRepository” qui est une classe qui répertorie toutes les fonctions qui se chargent de faire les requêtes SQL.
Ainsi, pour factoriser le code, les classes filles doivent posséder 3 méthodes : (“getNomClePrimaire”, “getNomsColonnes, “getNomTable”).

La première permet d’avoir le nom de la colonne qui est clé-primaire dans la table, en cas de recherche dans la base de données notamment.
La seconde retourne le nom des colonnes dans la base de donnée sous forme d’un tableau, entre autres utilisé lors de l’enregistrement dans le base de donnée.
La dernière permet d’avoir le nom de la table correspondant à l’objet.

Combiné, ces méthodes permettent de définir l’objet pour le repository principal, et lui fournir les informations nécessaires aux interactions avec la base de données.

Design Pattern

Le Patron Singleton



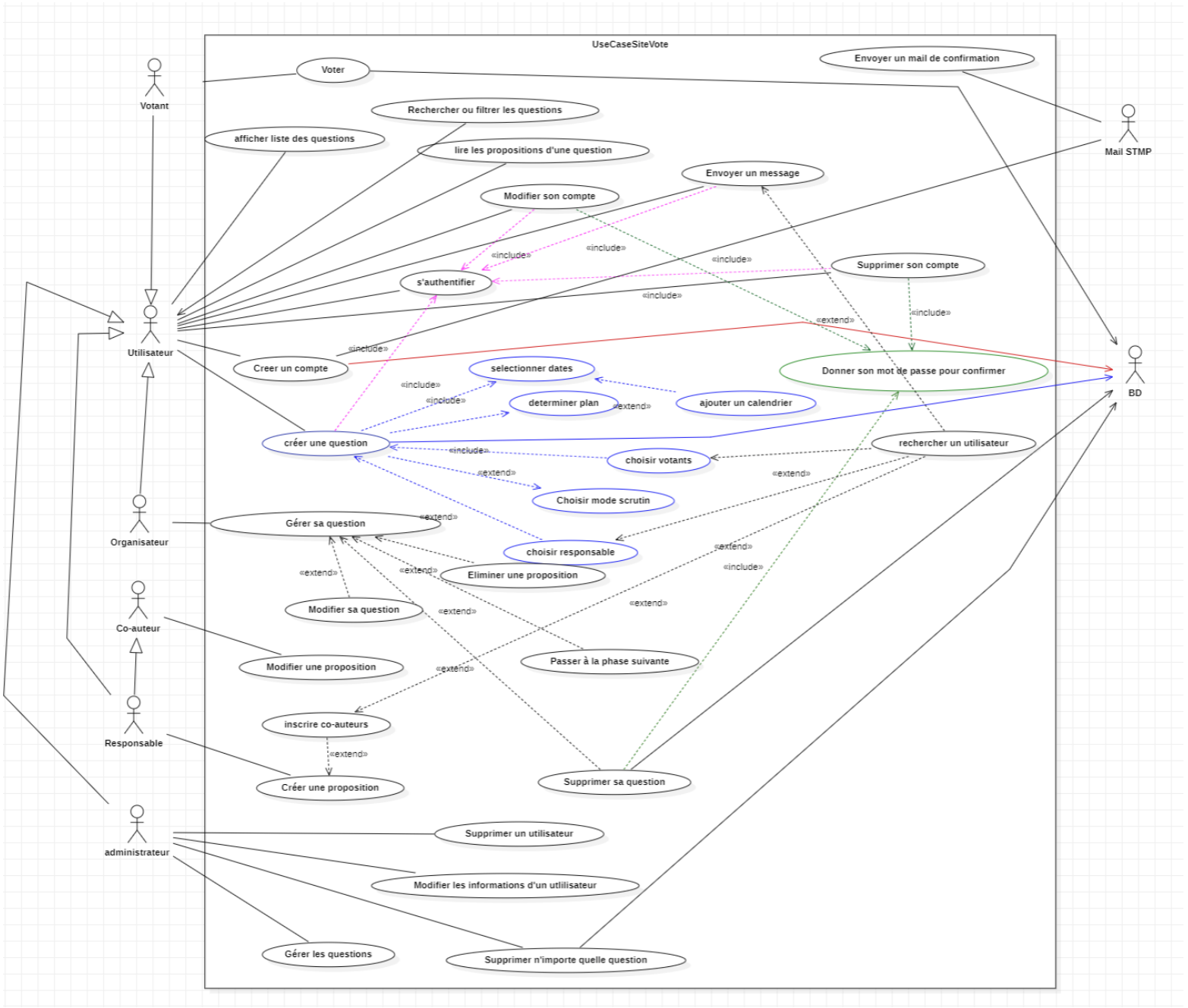
Ci-dessus la classe session, pour cette classe nous utilisons le patron singleton. On peut voir qu'elle possède une instance initialisée à null et une fonction getInstance() permettant d'y accéder ou de l'instancier si jamais elle vaut null.

```php
class Session
{
    private static ?Session $instance = null;

    /**
     * @throws Exception
     */
    private function __construct()
    {
        if (session_start() === false) {
            throw new Exception( message: "La session n'a pas réussi à démarrer.");
        }
    }

    public static function getInstance(): Session
    {
        if (is_null(static::$instance))
            static::$instance = new Session();
        return static::$instance;
    }
```

Voici une partie du code de cette classe, on constate bien que getInstance() appelle le constructeur si l'instance vaut null, retourne l'instance sinon.

Ce singleton permet de s'assurer qu'il existe qu'une variable de session, en évitant d'appeler la fonction session_start() à chaque fois que l'on souhaite l'utiliser.

Diagramme de cas d’utilisation

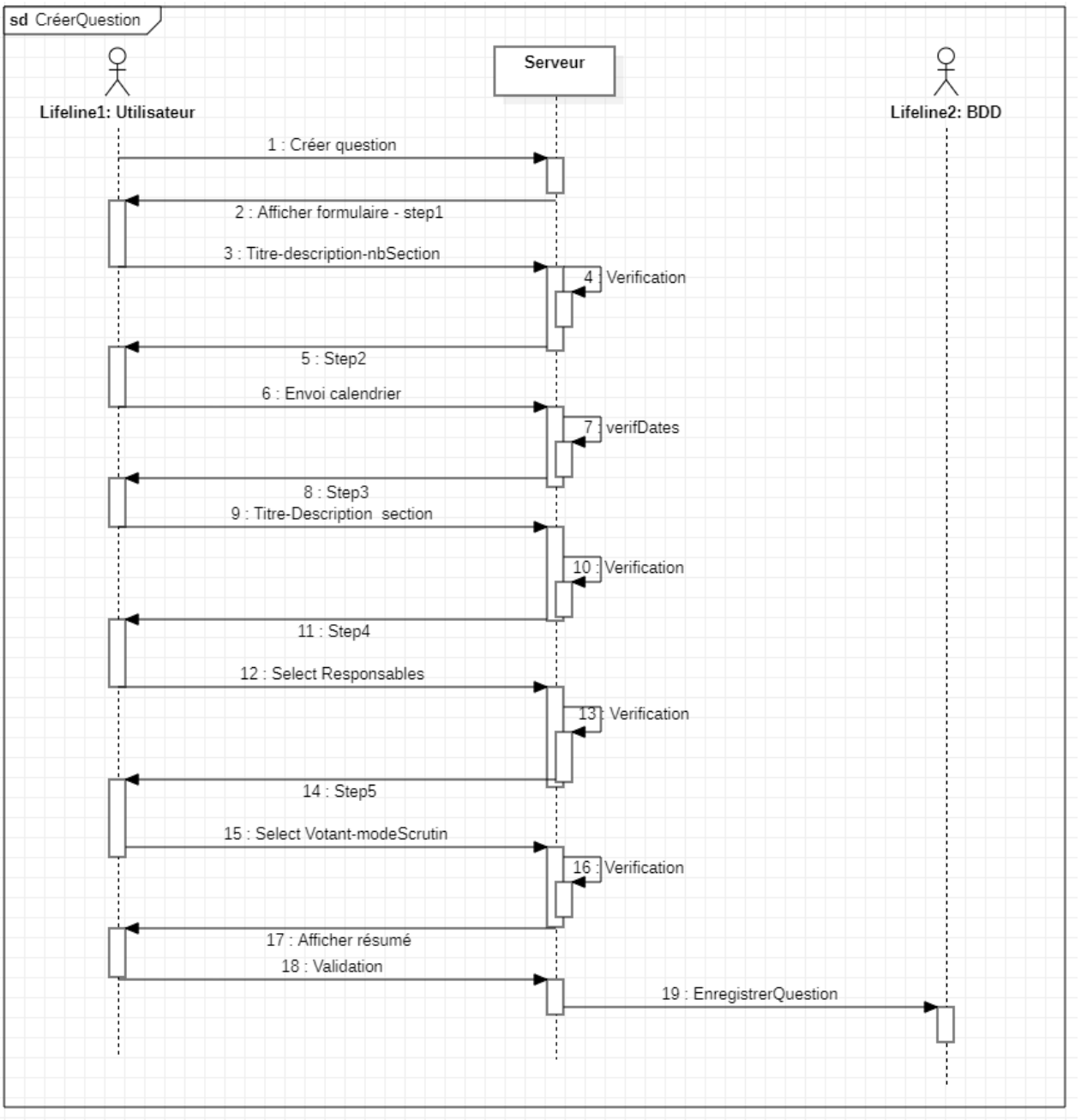


Le diagramme ci-dessus nous aide à visualiser les différents acteurs qui interviennent sur le site et leurs actions.
La base de données et le service d’email sont des acteurs qui n'interagissent pas avec les fonctionnalités du site, tandis que les autres sont des personnes réelles qui peuvent utiliser les différentes fonctionnalités.
Par défaut, l’acteur utilisateur est déconnecté. Tous les autres acteurs sont connectés.
Nous pouvons voir que pour créer une question, un utilisateur doit être connecté.

Aussi, pour créer une question, nous sommes obligés de saisir un titre, une description, un nombre de sections, le calendrier, le titre et la description de chacune des sections, les votants et responsables ainsi que le système de vote.
Tous les utilisateurs peuvent créer un compte, le modifier ou le supprimer.
La plupart des actions de suppression (question, compte, proposition) demandent une confirmation à l'utilisateur, en lui demandant son mot de passe pour sécuriser davantage ces opérations.
Aussi, pour sélectionner des utilisateurs, si on souhaite par exemple envoyer un message à un autre utilisateur ou sélectionner des votants, responsables ou co-auteurs, on peut rechercher un utilisateur.
Cette recherche s'applique automatiquement sur le nom, prénom et identifiant des utilisateurs.
Enfin, l'administrateur est un "super-utilisateur" qui peut supprimer n'importe quelle question qui pourrait être inappropriée, il peut aussi supprimer un compte qui pose souci à la communauté.
Un responsable d'une question peut gérer sa question, la supprimer, la modifier ou encore passer à la phase suivante. Il peut aussi éliminer une proposition, si la question a plusieurs phases de votes.

Diagrammes de séquence

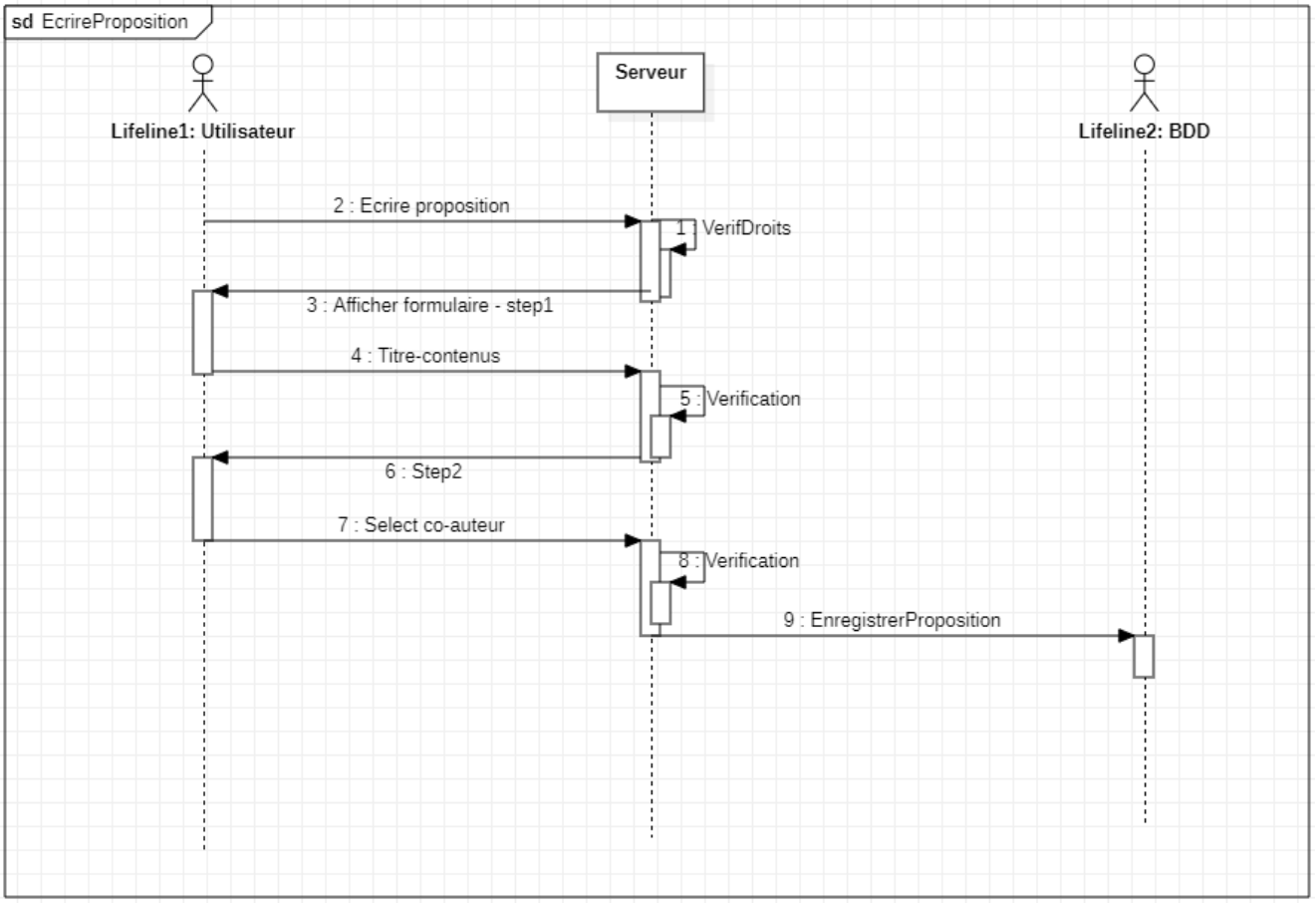
Créer une question



Ce diagramme de séquence représente l'action de créer une question.
Ici, l'utilisateur est un utilisateur connecté au site.
Tout d'abord, l'utilisateur demande le formulaire, que le serveur lui envoie.
Ensuite, s'ensuit une série d'étapes par exemple dans le step1, l'utilisateur devra entrer le titre, la description et le nombre de sections de la question et ainsi de suite.
Après chaque étape, le serveur effectue une vérification des données saisies, par exemple pour le calendrier il vérifie la cohérence des dates.

Une fois toutes les étapes terminées, le serveur affiche un résumé de la question que l'utilisateur veut créer et ce dernier devra valider sa saisie. Une fois validé, le serveur enregistre dans la base de données la question qui vient d'être créé, après avoir vérifié de nouveau la cohérence des données.

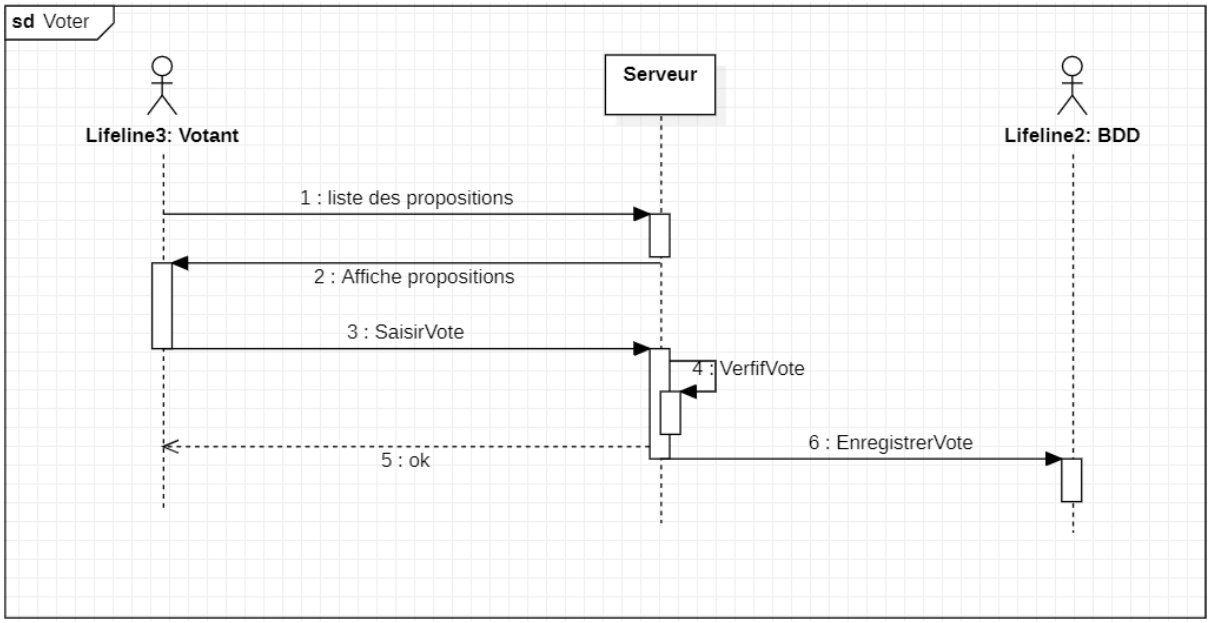
Ecrire une proposition



Ce diagramme représente l'écriture d'une proposition en réponse à une question posée.
L'utilisateur est connecté au site et est responsable pour la question donnée.
L'utilisateur demande d'écrire une proposition, le serveur vérifie qu'il possède les droits pour effectuer cette action, et qu'il n'a pas déjà créé une proposition.
Il lui affiche le formulaire s'il peut écrire une proposition sur cette question.
La première étape consiste à entrer un titre pour la proposition ainsi que le contenu pour chaque section.
La deuxième étape consiste à désigner des co-auteurs qui vont aider à écrire la proposition.
Comme pour la création d'une question, le serveur effectue des vérifications à chaque étape. Une fois les sélections validées, le serveur enregistre la proposition dans la

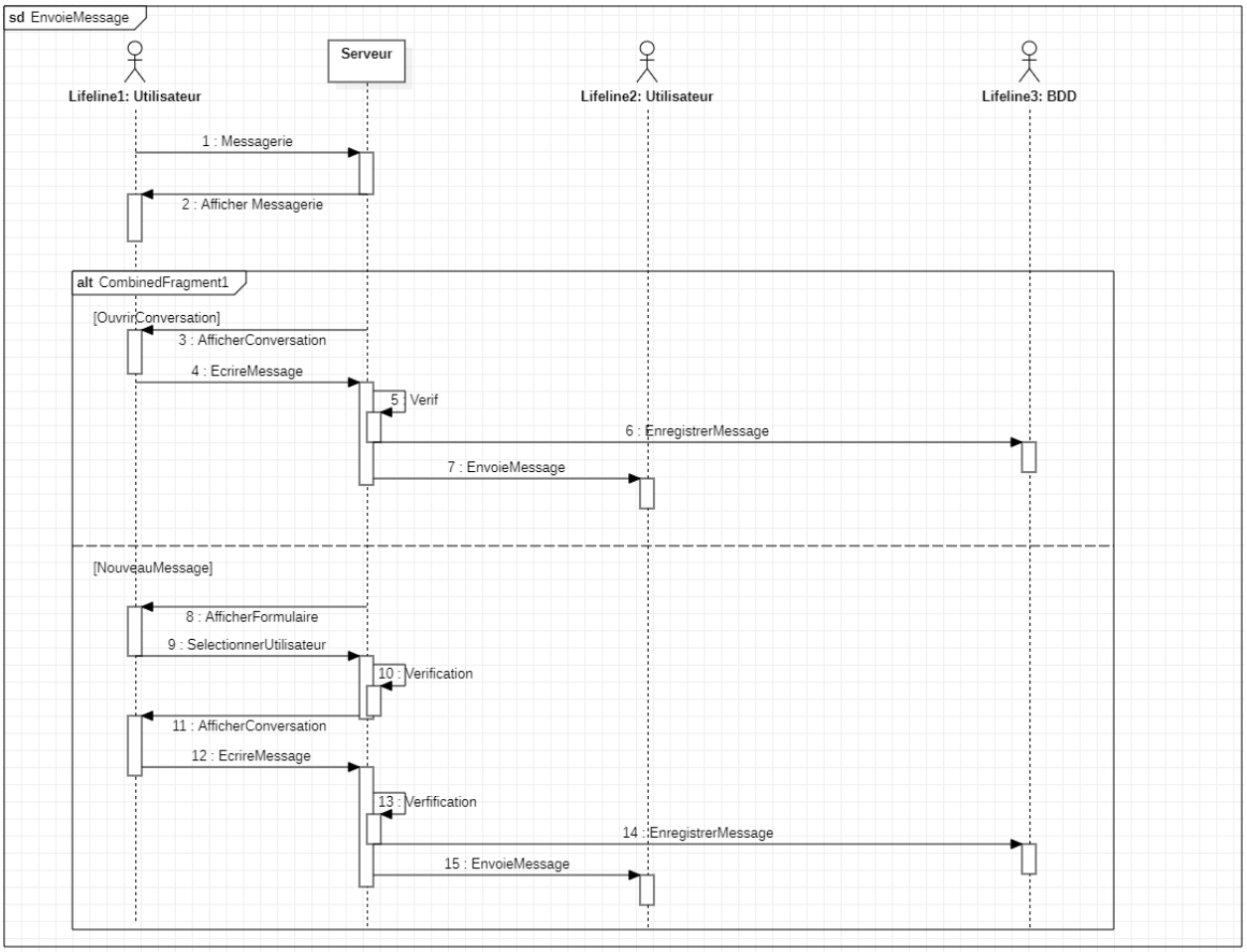
base de données après une dernière vérification de la cohérence des données, et de la taille de données saisies.

Voter



Ce diagramme représente le vote sur une ou plusieurs propositions par un utilisateur.
Ici, le votant est un utilisateur connecté qui possède le droit de vote sur la question.
L'utilisateur demande la liste des propositions, le serveur lui affiche.
Sur cette page il pourra voir la liste des propositions ainsi que voter pour ces dernières.
Ensuite, une vérification est effectuée pour voir si l'utilisateur a bien le droit de voter, puis si la valeur du vote est valable (dans la cas du jugement majoritaire), et que la proposition n'a pas été éliminée.
Cette vérification est importante pour éviter que les résultats ne soient faussés.
Enfin, le serveur effectue une vérification et l'enregistre dans la base de données, puis envoie la validation à l'utilisateur.

Envoyer un message



Ce diagramme représente l'envoi d'un message de l'utilisateur 1 à l'utilisateur 2 via notre site.
L'utilisateur 1 est un utilisateur connecté et l'utilisateur 2 est un utilisateur existant et enregistré dans la base de données.
L'utilisateur 1 demande d'ouvrir sa messagerie, le serveur lui affiche.
Deux choix s'offre à lui :

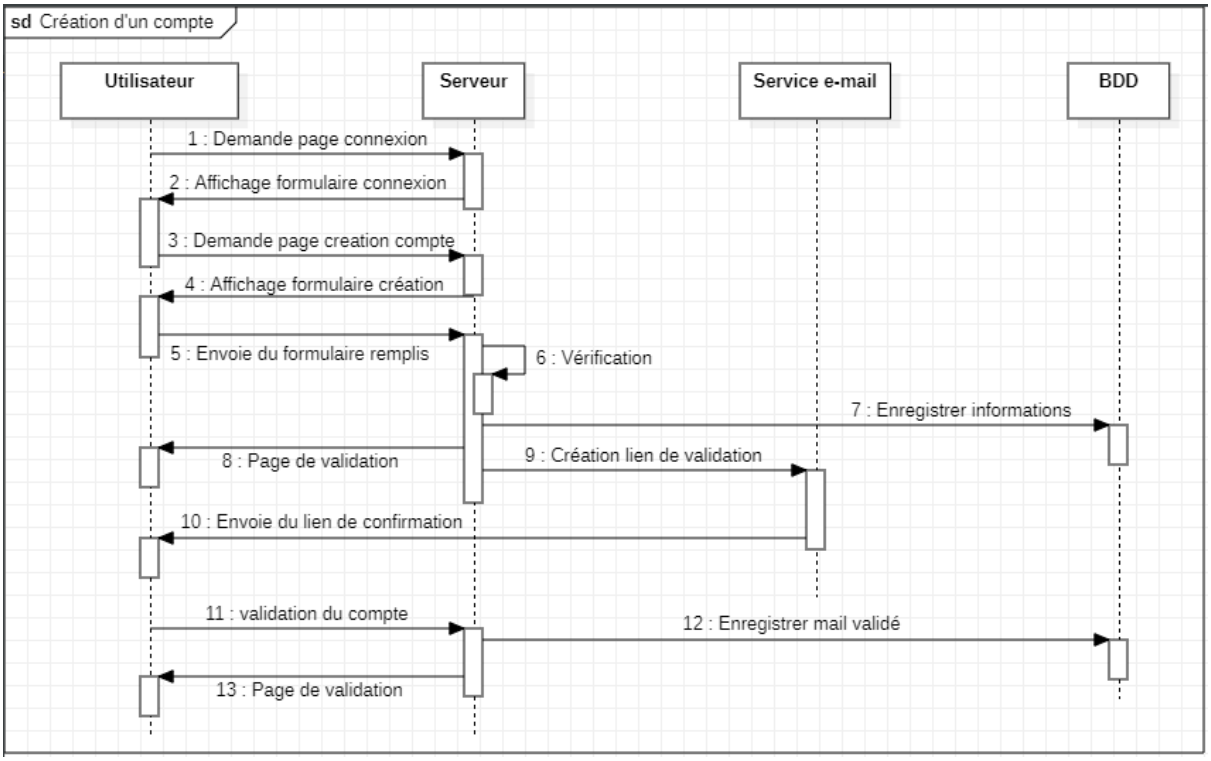
Cas 1 : Ouvrir une conversation déjà commencée avec l'utilisateur 2

Dans ce cas, le serveur lui affiche la conversation avec l'utilisateur, il peut donc écrire un nouveau message. Le serveur enregistre le message dans la base de données avant de l'envoyer à l'utilisateur 2.

Cas 2 : Commencer une nouvelle conversation avec l'utilisateur 2

Dans ce cas, le serveur envoie un formulaire où l'utilisateur devra sélectionner l'utilisateur à qui envoyer un message. Le serveur effectue une vérification et affiche un formulaire où l'utilisateur pourra écrire et envoyer son message. Le serveur enregistre le message dans la base de données avant de l'envoyer à l'utilisateur 2.

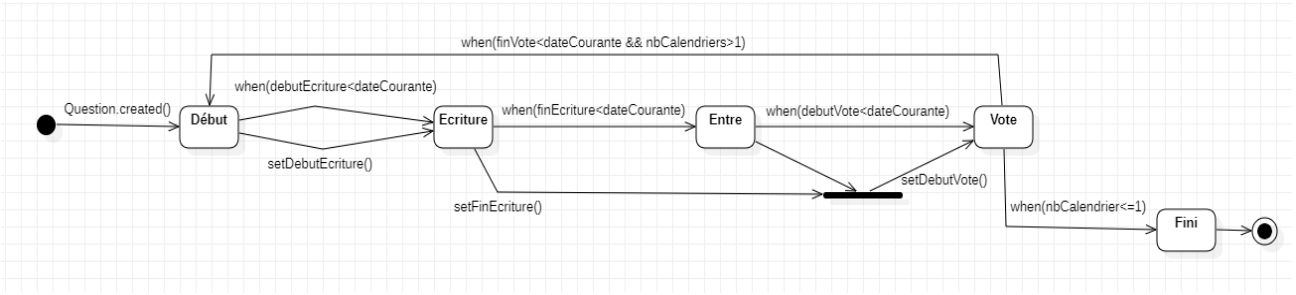
Créer un compte



Celui-ci symbolise les actions réalisées lors de la création d'un compte par un visiteur du site.
L'utilisateur souhaite se créer un compte, il se rend sur le bouton connexion de la barre de navigation
Le serveur affiche le formulaire de connexion
Ensuite, l'utilisateur clique sur le lien "Pas encore de compte ?"
Le site renvoie un formulaire ou l'ont peut renseigner un identifiant, nom, prénom, email et mot de passe afin de créer un compte
Le site récupère les informations et vérifie en interne si les champs ont été correctement remplis et validés, et finit par enregistrer les informations dans la base de données. Par la suite, le service d'email va envoyer un lien à l'adresse renseigner pour confirmer celle-ci.
L'utilisateur valide son compte en cliquant sur le lien fourni dans le mail.
Avec cette confirmation, le site va transmettre la confirmation a la base de données et compléter les informations.

Diagrammes d'état-transition

Etats d’une question



Ce diagramme représente les différentes phases que prend une question de la création de celle-ci à son résultat. Ses phases sont nécessaires pour structurer les différentes étapes du scrutin et ainsi définir ce que les utilisateurs ont le droit de faire selon leur rôle.

- La phase “Début” est attribuée dès la création d’une question, elle permet à tous utilisateurs de consulter la question et les sections afin de voir si elle est intéressante à leurs yeux. Pour ceux qui ont été désigné responsable, cela leur permet de réfléchir à une proposition adéquate.
  Aussi, cette phase permet au responsable de modifier sa question, et d’éliminer des propositions s’il s’agit d’une question multi-phase qui a déjà passé une phase de vote.

- La phase "Écriture" définit la période de temps ou les responsables de la question peuvent écrire et publier leurs propositions, ils peuvent également la modifier uniquement durant cette période.

- La phase "Vote" permet aux votants qui ont été désignés au préalable de voter pour la proposition qu’ils préfèrent (leur manière de voter dépend du mode de scrutin sélectionnés par l’organisateur de la question)

- La phase "Entre" intervient entre les phases d’écriture et de vote, elle permet aux votants de lire les différentes propositions en prévision de leur choix.

- La phase "Fini" signe la fermeture de la question et rend accessible les résultats.

debutEcriture/debutVote : la date du début de la phase décrite
finEcriture/finVote : la date de fin de la phase décrite
Les fonctions setDebutEcriture(), setFinEcriture(), setDebutVote(), permettent de commencer ou de mettre fin à une phase manuellement.