ECOLE MAROCAINE DES SCIENCES DE L'INGENIEUR
Membre de HONORIS UNITED UNIVERSITIES

PROJET

GitHub

préparer par:

BOUSLIM Saloua
EL BAHRAOUY Fatiha

Sommaire

I. Introduction et Contexte Général

1. Introduction

Le présent cahier des charges a pour objectif de définir de manière précise et détaillée les besoins, les objectifs et les contraintes liés au GitHub. Il s'agit d'un document essentiel qui servira de référence tout au long du déroulement du projet, tant pour les parties prenantes internes que pour les prestataires externes.

2. Présentation Github

GitHub est une plateforme de développement, permettant aux programmeurs et aux développeurs stocker, gérer et suivre les versions de leur code source et partager le code informatique de leurs projets afin de travailler dessus de façon collaborative. On peut le considérer comme un Cloud dédié au code informatique.

Le code source des projets est hébergé dans différents langages de programmation, et les changements apportés à chaque itération sont gardés en mémoire. Les autres utilisateurs de GitHub peuvent passer en revue le code et proposer des modifications ou des améliorations.

3. Problématique

GitHub a été créé pour résoudre les problématiques liées à la gestion du développement de logiciels, à la collaboration entre développeurs et à la gestion des versions de code source. Il s'agit d'une plateforme de développement collaboratif qui permet aux développeurs de travailler sur des projets de logiciels de manière collaborative.

Voici quelques problématiques liées à la création de GitHub :

a) Gestion du code source

GitHub a été créé pour faciliter la gestion du code source. Il permet aux développeurs de stocker leur code de manière centralisée, de suivre les modifications apportées au code, de gérer les branches de développement, et de fusionner les contributions de plusieurs personnes.

b) Collaboration

GitHub a révolutionné la manière dont les développeurs collaborent sur des projets logiciels. Il a permis de faciliter la contribution de développeurs du monde entier à des projets open source en permettant la soumission de "pull requests" pour proposer des modifications au code.

c) Gestion des versions

GitHub intègre des fonctionnalités de gestion des versions qui permettent de suivre l'évolution du code au fil du temps. Cela facilite la gestion des versions stables du logiciel tout en conservant un historique complet des modifications.

d) Problématiques de sécurité

La création de GitHub a soulevé des questions liées à la sécurité, notamment en ce qui concerne la protection du code source et la gestion des droits d'accès. La sécurité est un enjeu majeur pour les projets open source et les entreprises qui utilisent GitHub.

e) Écosystème des outils

GitHub a donné naissance à un écosystème de nombreux outils et services complémentaires, ce qui a soulevé des questions sur l'intégration, la compatibilité et la dépendance vis-à-vis de la plateforme.

f) Impact sur le développement de logiciels

La popularité de GitHub a eu un impact sur la manière dont les logiciels sont développés et diffusés. Les pratiques de développement agile, de collaboration ouverte et de contribution communautaire sont devenues plus courantes grâce à GitHub.

4. Les enjeux

a) Collaboration Ouverte

Favoriser un environnement de collaboration ouverte et respectueuse. GitHub est un lieu de convergence pour les développeurs du monde entier, et maintenir une communauté inclusive est crucial.

b) Sécurité

Assurer la sécurité des dépôts de code en protégeant contre les attaques, en gérant les vulnérabilités et en préservant la confidentialité des données.

c) Intégration d'Outils Tiers

Améliorer l'écosystème d'intégrations avec des outils tiers pour permettre aux développeurs de personnaliser leurs flux de travail.

d) Évolutivité

S'adapter à la croissance continue du nombre d'utilisateurs et de projets pour garantir des performances optimales.

e) Monétisation et Modèle Économique

Trouver un équilibre entre fonctionnalités gratuites et payantes, ainsi que définir une tarification équitable.

f) Respect de la Vie Privée

Maintenir des normes élevées en matière de respect de la vie privée et de sécurité des données.

g) Gouvernance des Projets Open Source

Garantir la pérennité et l'ouverture des projets open source, ainsi qu'une gestion transparente des contributions.

h) Éducation et Formation

Fournir des ressources et des programmes pour soutenir l'éducation et la formation des développeurs.

i) Éthique et Responsabilité

S'engager à des pratiques transparentes, lutter contre les abus tels que le code malveillant, et promouvoir une utilisation éthique de la plateforme.

j) Innovation

Continuer à innover pour répondre aux besoins changeants des développeurs, en améliorant les outils, les flux de travail et en adoptant de nouvelles technologies.

5. Les défis

GitHub vise à résoudre de nombreux défis liés à la collaboration et à la gestion de projets de développement logiciel en fournissant une plateforme centralisée, des outils de gestion de versions, des fonctionnalités de suivi des problèmes, des capacités de CI/CD, et bien plus encore. Cela contribue à rendre le développement logiciel plus efficace, collaboratif et transparent.

6. Parties Prenantes Principales

GitHub rassemble une diversité de parties prenantes, créant ainsi un écosystème dynamique de développement collaboratif.

a) Développeurs

Utilisateurs principaux, contribuent au code, ouvrent des problèmes, et participent activement aux projets hébergés sur GitHub.

b) Mainteneurs de Projets

Responsables de la gestion des dépôts, examinent les demandes de tirage, définissent les orientations du projet et assurent la qualité du code.

c) Équipes de Développement

Comprend divers rôles tels que développeurs, testeurs, concepteurs. GitHub facilite la collaboration et la gestion de projet au sein des équipes.

d) Entreprises et Organisations

Utilisent GitHub pour héberger, gérer le code source et collaborer sur des projets internes ou open source.

e) Contributeurs Externes

Individus ou organisations contribuant à des projets open source sans nécessairement appartenir à l'équipe principale.

f) Utilisateurs de Logiciels

Bénéficient des projets open source hébergés sur GitHub, signalent des problèmes, suggèrent des améliorations.

g) Gestionnaires de Projet

Utilisent GitHub pour suivre les progrès, planifier les tâches, gérer les ressources et coordonner les activités.

h) Responsables de la Sécurité

Veillent au respect des normes de sécurité, identifient et corrigent les vulnérabilités.

i) Communauté Open Source

Individus et organisations contribuant aux projets open source, favorisant la collaboration.

j) Développeurs d'Outils Tiers

Créent des applications, extensions et intégrations pour compléter GitHub, améliorant son utilité.

7. Objectifs du GitHub

Les objectifs de GitHub visent à améliorer la productivité des développeurs, à encourager la collaboration ouverte et à soutenir le développement de logiciels de haute qualité, qu'il s'agisse de projets open source ou de projets d'entreprise.

Voici quelques objectifs du GitHub :

a) Favoriser l'open source

GitHub soutient activement la communauté open source en hébergeant de nombreux projets open source populaires. L'objectif est de promouvoir la collaboration ouverte, le partage de code source et le développement collaboratif dans le monde entier.

b) Offrir des outils pour le développement agile

GitHub propose des fonctionnalités qui soutiennent les méthodes de développement agile, telles que la gestion de projets, l'intégration continue et le déploiement continu (CI/CD), et la planification agile des tâches.Y

c) Faciliter la gestion de versions

L'objectif principal de GitHub est de faciliter la gestion des versions du code source grâce à Git. Il permet aux développeurs de suivre les modifications, de gérer des branches, de fusionner les modifications et de rétrograder les commits de manière transparente.

d) Fournir des outils de qualité

GitHub s'efforce de fournir des outils de qualité qui améliorent la productivité des développeurs. Cela comprend des fonctionnalités telles que la documentation intégrée, la gestion des tests, la sécurité du code et des analyses de performances.

e) Encourager l'apprentissage et le partage de connaissances

GitHub encourage l'apprentissage continu et le partage de connaissances entre les développeurs en offrant des forums, des wikis, des discussions et d'autres fonctionnalités de communication.

f) Garantir la sécurité

GitHub travaille pour garantir la sécurité des dépôts en proposant des outils de détection de vulnérabilités, des fonctionnalités d'authentification et de contrôle d'accès, et en fournissant des mises à jour de sécurité.

g) Fournir des solutions pour les entreprises

GitHub propose des solutions adaptées aux besoins des entreprises, notamment la gestion des droits d'accès, la gestion de la sécurité et des fonctionnalités de collaboration avancées.

h) Évoluer avec la technologie

GitHub s'efforce de rester à jour avec les dernières technologies et les meilleures pratiques de développement, en offrant des fonctionnalités et des intégrations qui répondent aux besoins changeants des développeurs.

i) Gestion des Problèmes

GitHub offre un système robuste de suivi des problèmes (issues) qui permet aux utilisateurs de signaler des bugs, de demander des fonctionnalités et de discuter des améliorations. Cela facilite la communication entre les membres de l'équipe et les contributeurs externes.

j) Transparence et Traçabilité

GitHub favorise la transparence dans le processus de développement. Les utilisateurs peuvent voir qui a contribué à un projet, quelles modifications ont été apportées, et quand. Cela contribue à la traçabilité du code et facilite la résolution des problèmes.

k) Automatisation des Processus

Déploiement fonctionnalités telles que GitHub Actions, la plateforme vise à automatiser plusieurs aspects du cycle de vie du développement, y compris la construction, les tests et le déploiement.

l) Faciliter la Contribution Ouverte

GitHub encourage la contribution ouverte en permettant aux développeurs externes de proposer des modifications sous la forme de demandes d'extraction (pull requests). Cela favorise la croissance des communautés de développement.

m) Documentation

La plateforme intègre des fonctionnalités telles que les wikis pour permettre aux équipes de documenter leurs projets. Cela contribue à la création d'une documentation complète et accessible pour les utilisateurs et les contributeurs.

n) Soutien aux Projets Open Source

GitHub a été particulièrement important pour la communauté open source en fournissant un espace gratuit pour héberger des projets, en facilitant la contribution ouverte et en offrant des outils pour la collaboration à grande échelle.

8. Portée du GitHub

a) Les limites

GitHub est une plateforme puissante pour la gestion de projets de développement logiciel, mais elle présente également certaines limites et contraintes. Voici quelques limites du GitHub :

b) Coût

GitHub propose des services gratuits pour les projets open source, mais les dépôts privés et certaines fonctionnalités avancées sont payants. Cela peut être coûteux pour les entreprises ou les organisations qui ont besoin de fonctionnalités plus avancées.

c) Dépendance à l'Internet

GitHub est une plateforme basée sur le cloud, ce qui signifie que vous devez avoir une connexion Internet pour y accéder. Si la connexion Internet est interrompue, l'accès à vos dépôts et à vos outils de développement peut être limité.

d) Limites de stockage

Les comptes gratuits de GitHub ont des limites de stockage pour les fichiers et les dépôts. Si vous avez besoin de stocker de grandes quantités de données, vous pourriez devoir passer à un plan payant.

e) Taille des fichiers

GitHub a une limite de taille de fichier pour les téléchargements. Les fichiers très volumineux, tels que les ensembles de données, peuvent être problématiques.

f) Sécurité et confidentialité

Malgré que GitHub propose des outils de sécurité, héberger du code source sur une plateforme tierce implique toujours un certain risque en matière de sécurité et de confidentialité. Les entreprises doivent être attentives à la manière dont elles gèrent leurs données sensibles.

g) Personnalisation limitée

GitHub a des fonctionnalités de personnalisation limitées en ce qui concerne l'apparence et la convivialité de la plateforme. Certaines équipes peuvent souhaiter plus de flexibilité dans ce domaine.

h) Gestion de tâches complexe

GitHub propose des outils de suivi des problèmes et de gestion de projets, mais ils peuvent ne pas être aussi complets que certains autres outils de gestion de projet dédiés, ce qui peut être un défi pour les projets complexes.

i) Gestion de documents

Bien que GitHub prenne en charge la documentation dans les dépôts, il ne s'agit pas d'une solution de gestion de documents professionnelle. Pour la gestion de documents complexes, d'autres outils peuvent être plus appropriés.

j) Problèmes de performances

Les grands dépôts ou les projets très actifs peuvent rencontrer des problèmes de performances, en particulier dans les outils de suivi des problèmes, où la recherche et la navigation peuvent devenir lentes.

k) Politiques de sécurité

Certaines politiques de sécurité et de conformité spécifiques à une entreprise peuvent être difficiles à mettre en œuvre sur GitHub, ce qui peut limiter son utilisation dans certaines organisations.

9. Fonctionnalités

a) Gestion de Versions (Version Control)

- *Suivi des modifications de code source au fil du temps.*
- *Gestion des branches pour travailler sur des fonctionnalités isolées ou des correctifs de bogues.*
- *Fusion des modifications entre différentes branches.*

b) Collaboration et Travail d'Équipe

- *Collaboration simultanée sur un même projet par plusieurs développeurs.*
- *Suivi des problèmes (issues) pour signaler les bogues, demander des fonctionnalités ou discuter des tâches.*
- *Attribution de tâches aux membres de l'équipe.*
- *Discussions autour du code via les commentaires.*

c) Hébergement de Code Source

- *Stockage centralisé du code source pour un accès facile.*
- *Possibilité de rendre le code source public ou privé selon les besoins du projet.*

d) Intégration Continue (Continuous Integration, CI) et Déploiement Continu (Continuous Deployment, CD)

- *Intégration de services CI/CD pour automatiser les tests et le déploiement.*
- *Déclenchement de workflows automatisés à chaque modification du code source.*

e) Suivi des Progrès

- *Tableaux de bord pour suivre les progrès du projet.*
- *Rapports d'activité pour comprendre qui a contribué à quoi.*

f) Documentation Collaborative

- *Utilisation du Wiki pour créer une documentation collaborative.*
- *Intégration avec des outils de documentation tels que Markdown.*

g) Gestion de Projet

- *Utilisation des projets GitHub pour organiser et prioriser les tâches.*
- *Création de milestones pour suivre les objectifs à long terme.*

h) Hébergement de Sites Web

- *Hébergement de sites web statiques directement depuis un dépôt GitHub.*

i) Publication de Packages et de Bibliothèques

- *Publication et gestion de packages logiciels via le registre de packages intégré.*

j) Communauté Open Source

- *Collaboration sur des projets open source en facilitant la contribution de la communauté.*
- *Forking des projets pour créer des versions dérivées.*

10. Exigences Techniques

Pour accéder à GitHub et collaborer sur des projets, nous avons besoin des éléments suivants :

a) Accès à Internet

Nous devons disposer d'une connexion Internet pour accéder à la plateforme GitHub, car tout se fait en ligne.

b) Navigateur web

GitHub est compatible avec la plupart des navigateurs web modernes, tels que Google Chrome, Mozilla Firefox, Microsoft Edge, Safari, etc. Assurez-vous de maintenir votre navigateur à jour pour une expérience optimale.

c) Compte GitHub

Nous devons créer un compte GitHub pour pouvoir créer des dépôts, contribuer à des projets et utiliser toutes les fonctionnalités de la plateforme. Les comptes GitHub sont gratuits, mais il existe également des options payantes pour les fonctionnalités avancées.

d) Git

Bien que GitHub offre une interface web pour la gestion de versions, il est utile de disposer du logiciel Git sur votre ordinateur si vous prévoyez de cloner, pousser, tirer et gérer des dépôts GitHub localement. Vous pouvez télécharger Git sur le site officiel de Git.

e) Environnement de développement

Pour travailler sur des projets GitHub, vous aurez besoin d'un environnement de développement approprié pour écrire, tester et exécuter du code. Cela peut inclure des éditeurs de code, des IDE (environnements de développement intégrés), des outils de gestion de paquets, etc.

f) Connexion sécurisée

Il est recommandé de disposer d'une connexion Internet sécurisée, en particulier si vous travaillez sur des projets sensibles ou privés. GitHub prend en charge HTTPS pour des connexions sécurisées.

g) Compréhension de Git

Bien que cela ne soit pas strictement une exigence technique, il est essentiel de comprendre les concepts de base de Git, tels que les commits, les branches, les fusions, etc., pour utiliser efficacement GitHub. Vous pouvez trouver de nombreuses ressources en ligne pour apprendre Git.

11. Design et mise en page

a) Identité Visuelle

GitHub n'impose pas une identité visuelle standard pour les projets ou les utilisateurs. Les utilisateurs et les organisations sont libres de personnaliser l'apparence de leurs pages GitHub en utilisant des images, des logos, des descriptions, et des couleurs pour refléter leur identité visuelle.

b) Maquettes ou Prototypes

- **Maquettes (Wireframes)**

Pour les projets GitHub, les équipes peuvent créer des wireframes pour planifier l'agencement des éléments sur leurs pages. Des outils de conception tels que Figma, Sketch ou Adobe XD peuvent être utilisés pour créer des maquettes.

- **Prototypes Interactifs**

Pour des projets plus complexes, des prototypes interactifs peuvent être créés pour simuler l'expérience utilisateur. Ces prototypes peuvent être créés à l'aide d'outils de prototypage comme InVision ou Axure.

c) Personnalisation de l'Interface GitHub

- **Personnalisation des Pages de Profil**

Les utilisateurs et les organisations peuvent personnaliser leur page GitHub en ajoutant une photo de profil, une biographie, des liens vers des sites web externes et des réseaux sociaux, ainsi que des projets mis en avant.

- **Thèmes Personnalisés**

GitHub permet aux utilisateurs de personnaliser l'apparence de leur interface utilisateur en utilisant des thèmes. Il existe plusieurs thèmes prédéfinis, mais les utilisateurs peuvent également créer leurs propres thèmes.

- **Personnalisation des Dépôts**

Les propriétaires de dépôts ont un certain degré de personnalisation. Personnalise jouter un fichier README.md pour décrire le projet, ajouter des badges de statut, personnaliser les balises (tags) et les descriptions, et même personnaliser la page principale du dépôt.

- **Utilisation de GitHub Pages**

Les utilisateurs peuvent créer des sites web statiques à l'aide de GitHub Pages. Cela permet la création de pages web personnalisées pour les projets, les documentations, et les portfolios.

12. Contenu

a) Types de Contenu

- **Code Source**

Les dépôts GitHub contiennent principalement le code source des projets. Cela inclut des fichiers de programmation, des bibliothèques, des modèles, etc.

- **Fichiers README**

Chaque dépôt peut contenir un fichier README.md qui sert de documentation d'introduction. Il est souvent écrit en format Markdown et explique le but du projet, comment le configurer et comment contribuer.

- **Documentation Technique**

Des fichiers Markdown, HTML ou d'autres formats peuvent être utilisés pour créer des documents techniques détaillés, y compris des guides d'utilisation, des FAQ et des manuels pour les développeurs.

- **Problèmes (Issues)**

Les problèmes sont utilisés pour signaler des bugs, demander des fonctionnalités ou discuter de sujets liés au projet.

- **Demandes de Tirage (Pull Requests)**

Les demandes de tirage contiennent des modifications spécifiques proposées par un contributeur. Elles incluent du code, mais aussi des explications sur les changements effectués.

- ***Pages GitHub (*GitHub *Pages)***

GitHub Pages permet aux utilisateurs de créer des sites web statiques directement à partir de leurs dépôts. Ces sites peuvent être utilisés pour la documentation, les blogs, les portfolios, etc.

- **Collaboration et Discussions**

Les discussions dans les problèmes ou dans des forums spécifiques aux projets peuvent contenir des idées, des questions, des suggestions et d'autres formes de collaboration.

b) Création de Contenu sur GitHub :

- **Édition de Fichiers**

Les fichiers dans les dépôts peuvent être édités directement sur le site web de GitHub. Les modifications peuvent être soumises via des demandes de tirage.

- **Contributions Open Source**

Les utilisateurs peuvent contribuer aux projets open source en clonant (copiant) les dépôts, en faisant des modifications localement et en soumettant des demandes de tirage pour proposer leurs modifications aux mainteneurs du projet.

- **Documentation Collaborative**

La documentation peut être écrite collaborativement en utilisant le format Markdown. Les membres de l'équipe peuvent ajouter, modifier et supprimer du contenu en créant des pull requests.

- **Partage de Code**

Les utilisateurs peuvent créer des gists pour partager des extraits de code, des scripts ou des fichiers avec d'autres personnes.

- **Communication**

Les problèmes, les commentaires dans les demandes de tirage, et les discussions servent de moyens de communication entre les membres de l'équipe et la communauté.

- **Gestion de Projet**

Les tâches peuvent être créées et gérées via les problèmes et les tableaux Kanban, permettant aux équipes de suivre le travail à faire, en cours et terminé

- **Code Source**

Les dépôts GitHub contiennent principalement le code source des projets. Cela inclut des fichiers de programmation, des bibliothèques, des modèles, etc.

- **Fichiers README**

Chaque dépôt peut contenir un fichier README.md qui sert de documentation d'introduction. Il est souvent écrit en format Markdown et explique le but du projet, comment le configurer et comment contribuer.

- **Documentation Technique**

Des fichiers Markdown, HTML ou d'autres formats peuvent être utilisés pour créer des documents techniques détaillés, y compris des guides d'utilisation, des FAQ et des manuels pour les développeurs.

- **Problèmes (Issues)**

Les problèmes sont utilisés pour signaler des bugs, demander des fonctionnalités ou discuter de sujets liés au projet.

- **Demandes de Tirage (Pull Requests)**

Les demandes de tirage contiennent des modifications spécifiques proposées par un contributeur. Elles incluent du code, mais aussi des explications sur les changements effectués.

- **Pages GitHub (GitHub Pages)**

GitHub Pages permet aux utilisateurs de créer des sites web statiques directement à partir de leurs dépôts. Ces sites peuvent être utilisés pour la documentation, les blogs, les portfolios, etc.

- **Collaboration et Discussions**

Les discussions dans les problèmes ou dans des forums spécifiques aux projets peuvent contenir des idées, des questions, des suggestions et d'autres formes de collaboration.

c) Création de Contenu sur GitHub :

- **Édition de Fichiers**

Les fichiers dans les dépôts peuvent être édités directement sur le site web de GitHub. Les modifications peuvent être soumises via des demandes de tirage.

- **Contributions Open Source**

Les utilisateurs peuvent contribuer aux projets open source en clonant (copiant) les dépôts, en faisant des modifications localement et en soumettant des demandes de tirage pour proposer leurs modifications aux mainteneurs du projet.

- **Documentation Collaborative**

La documentation peut être écrite collaborativement en utilisant le format Markdown. Les membres de l'équipe peuvent ajouter, modifier et supprimer du contenu en créant des pull requests.

- **Partage de Code**

Les utilisateurs peuvent créer des gists pour partager des extraits de code, des scripts ou des fichiers avec d'autres personnes.

- **Communication**

Les problèmes, les commentaires dans les demandes de tirage, et les discussions servent de moyens de communication entre les membres de l'équipe et la communauté.

- **Gestion de Projet**

Les tâches peuvent être créées et gérées via les problèmes et les tableaux Kanban, permettant aux équipes de suivre le travail à faire, en cours et terminé

II. CONCEPTION

1. Diagramme de Cas d'utilisation

Acteurs

Sur un diagramme de cas d'utilisation, les acteurs représentent les entités externes qui interagissent avec le système. Dans le cas de GitHub, Voici les principaux acteurs de GitHub :

- **Utilisateur**

L'utilisateur représente toute personne qui utilise GitHub, que ce soit pour la consultation de projets, la collaboration sur du code, la gestion de problèmes, etc. Il peut être un contributeur, un mainteneur de projet, ou simplement un observateur.

- **Collaborateur**

Un collaborateur est un utilisateur qui participe activement à un projet. Cela peut inclure la contribution de code, la création de problèmes, la gestion des demandes de tirage.

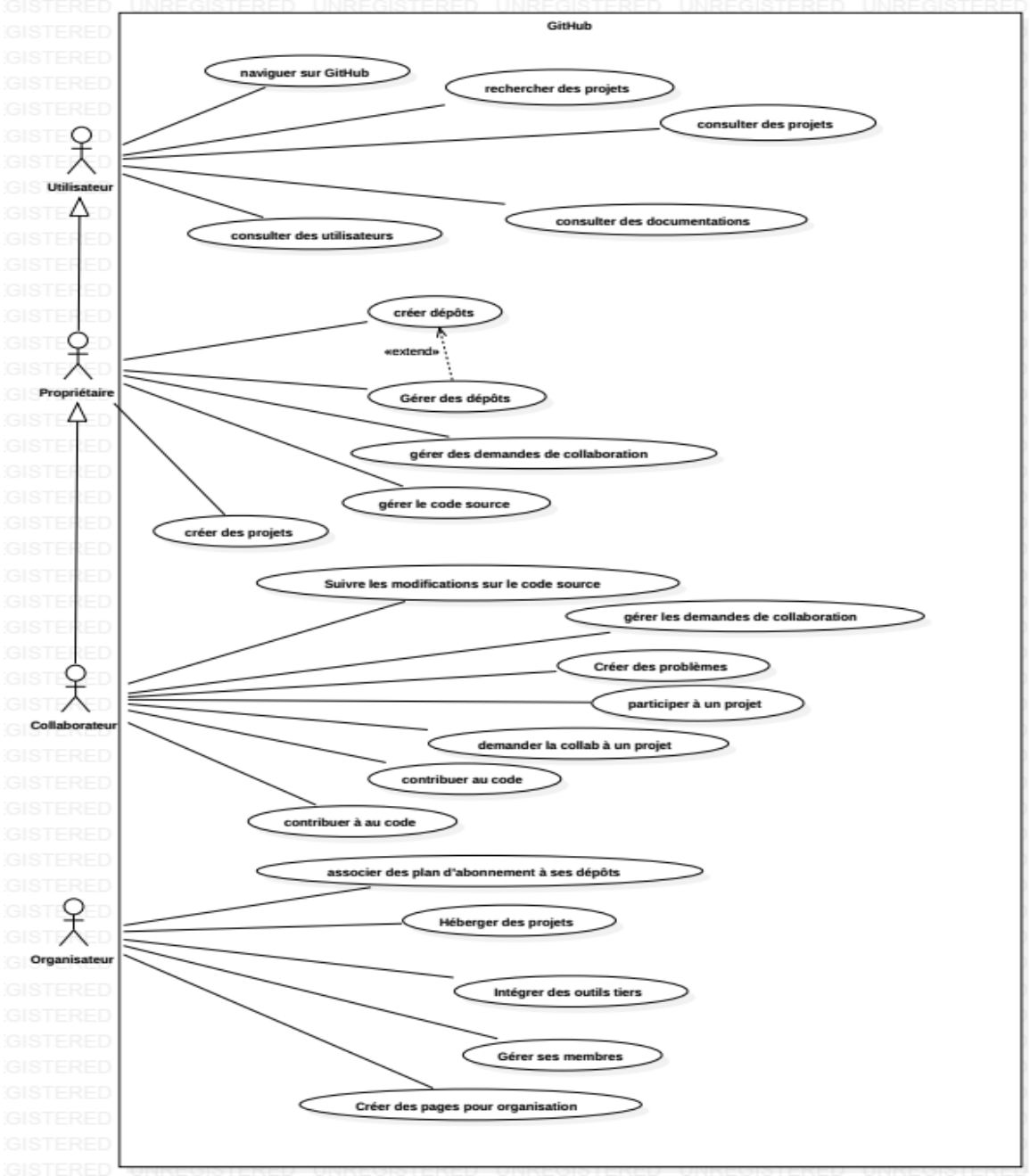
- **Propriétaire du projet**

Le propriétaire du projet est souvent le créateur initial du dépôt sur GitHub. Il a des autorisations spéciales pour gérer les paramètres du projet, accepter ou refuser des contributions.

- **Organisateur**

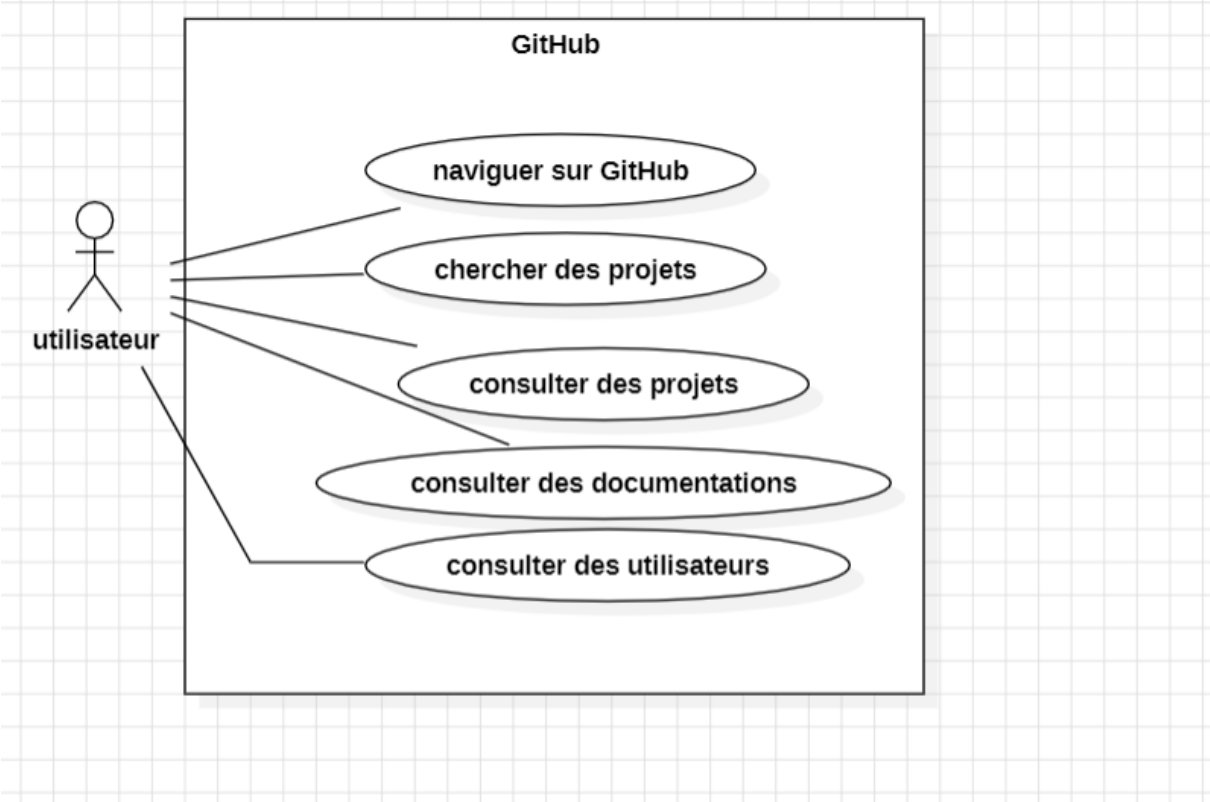
Une organisation sur GitHub regroupe plusieurs dépôts et membres sous une seule entité. Les organisations sont souvent utilisées pour représenter des entreprises, des groupes open source, des équipes.

Diagramme de cas d'utilisation global

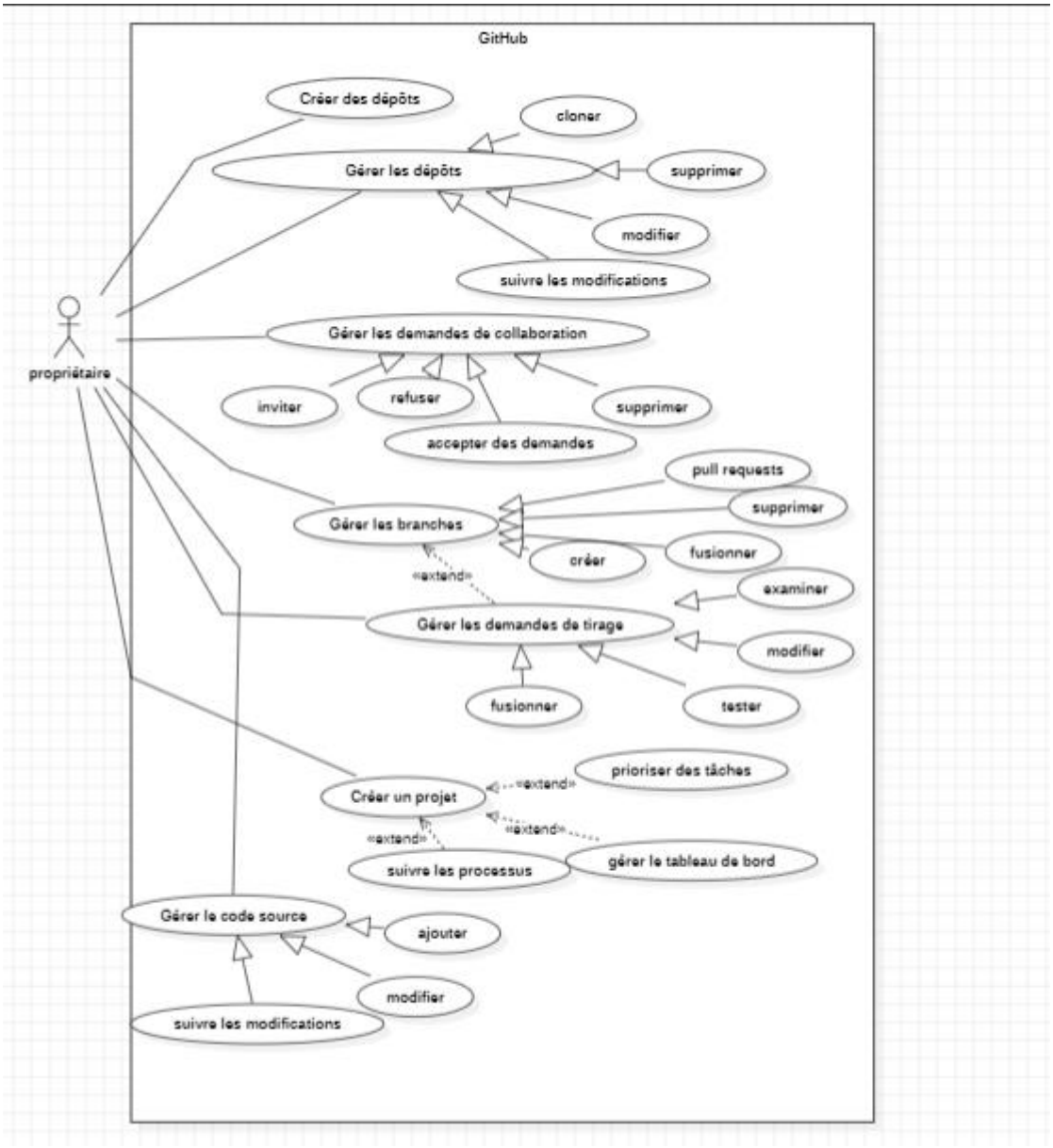


Sous diagrammes de cas d'utilisation

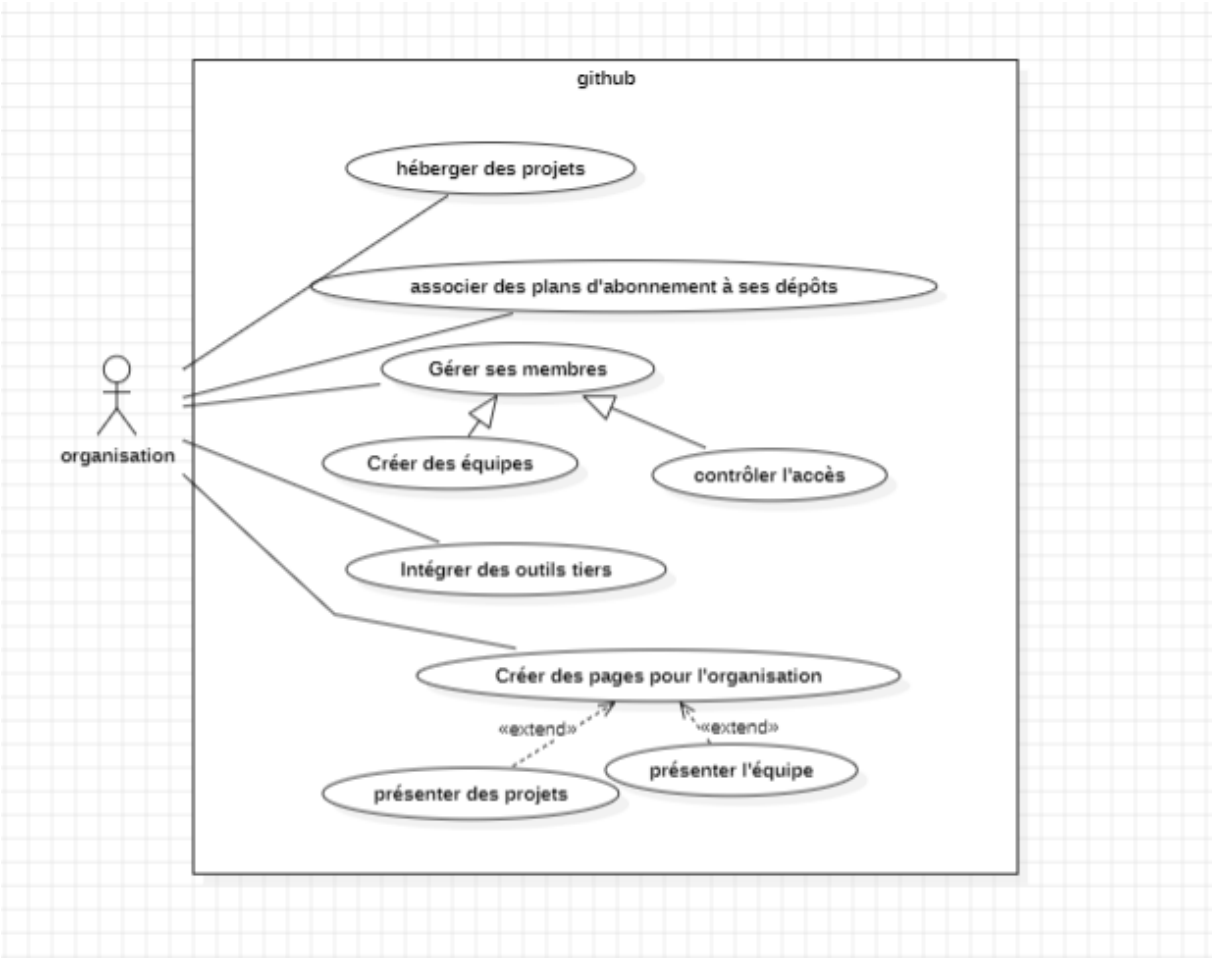
- **Utilisateur global :**



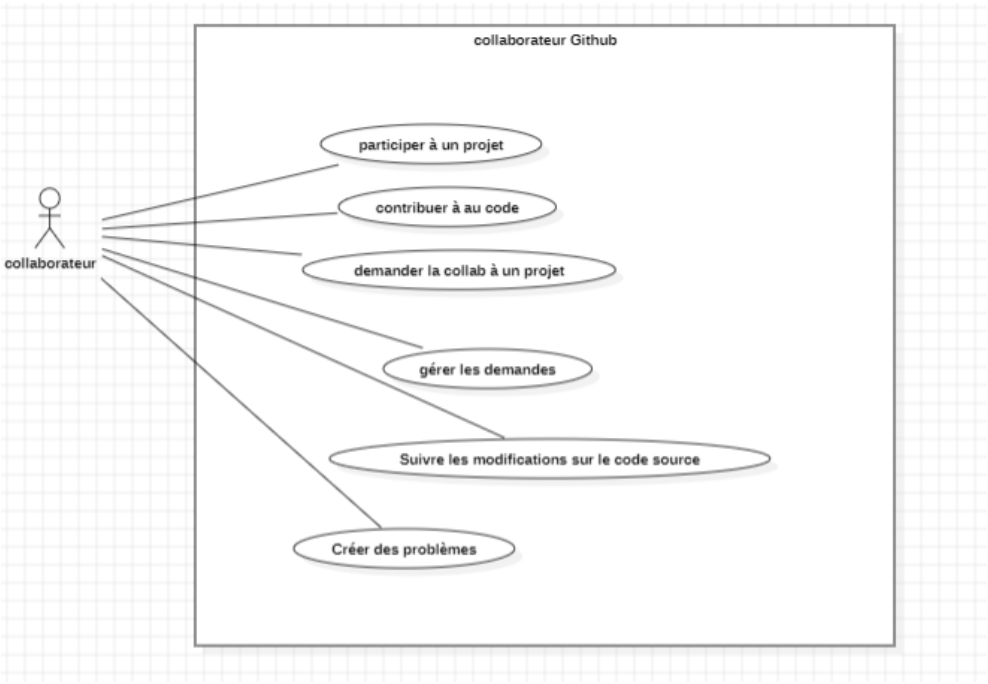
- **Propriétaire :**



- **Organisation**



- **Collaborateur**



REMARQUE : les actions des propriétaires, collaborateurs et organisations nécessitent une authentification au système.

2. Diagramme de Classe

Le diagramme de classe est un schéma utilisé en génie logiciel pour présenter les classes et les interfaces des systèmes ainsi que leurs relations. Voici les classes principales de GitHub :

- **Repository (Dépôt)**

Représente un espace où le code source, les fichiers et l'historique des versions sont stockés. Il peut être public ou privé.

- **User (Utilisateur)**

Un individu qui possède un compte GitHub et interagit avec les dépôts, crée des pull requests, des issues, etc.

- **Organization (Organisation)**

Un regroupement d'utilisateurs et de dépôts, souvent utilisé par des équipes ou des entreprises pour gérer et organiser leurs projets.

- **Commit**

Une unité de modification dans un dépôt Git. Elle représente un ensemble de modifications apportées à un ou plusieurs fichiers.

- **Pull Request (Demande de fusion)**

Une requête pour fusionner des modifications d'une branche vers une autre. C'est le moyen par lequel les contributions sont proposées et examinées avant d'être fusionnées dans la branche principale.

- **Issue (Problème)**

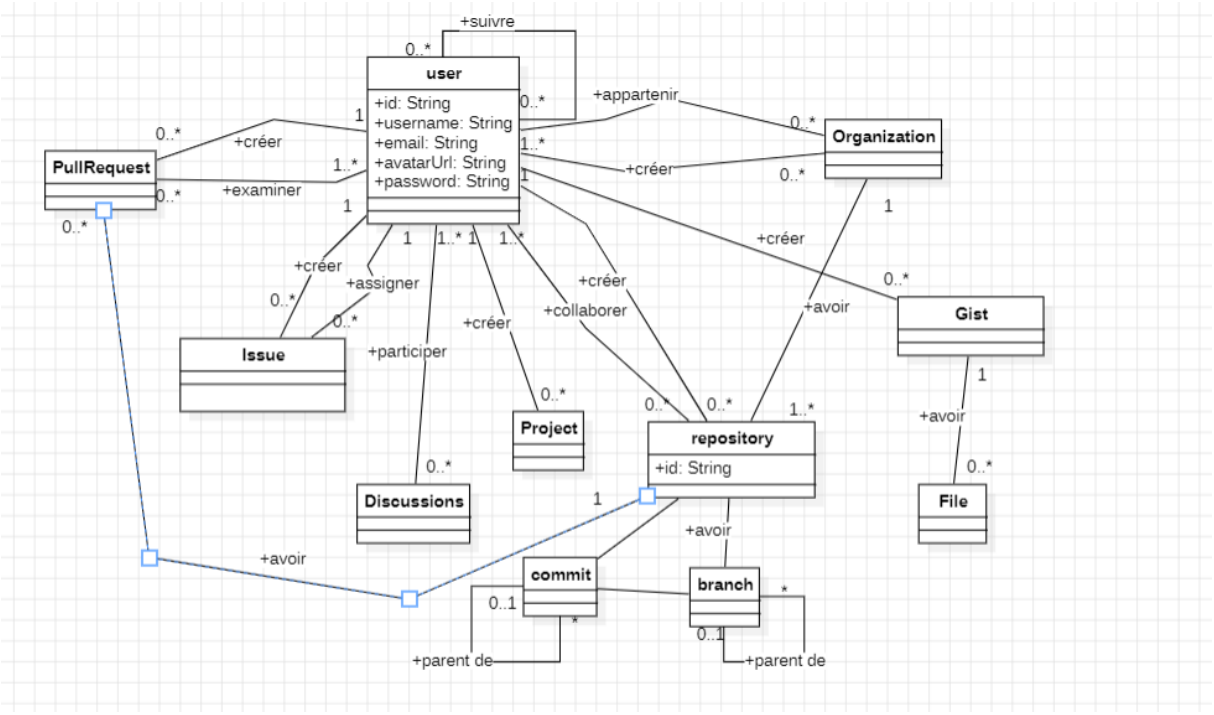
Utilisé pour suivre les tâches, les bogues, les demandes de fonctionnalités, etc. Cela permet une communication structurée autour des problèmes du code.

- **Projects(projets)**

Utilisé pour la gestion de projet intégré à la plateforme GitHub. Il permet aux équipes de planifier, organiser et suivre leur travail directement depuis leurs dépôts GitHub.

Etc.

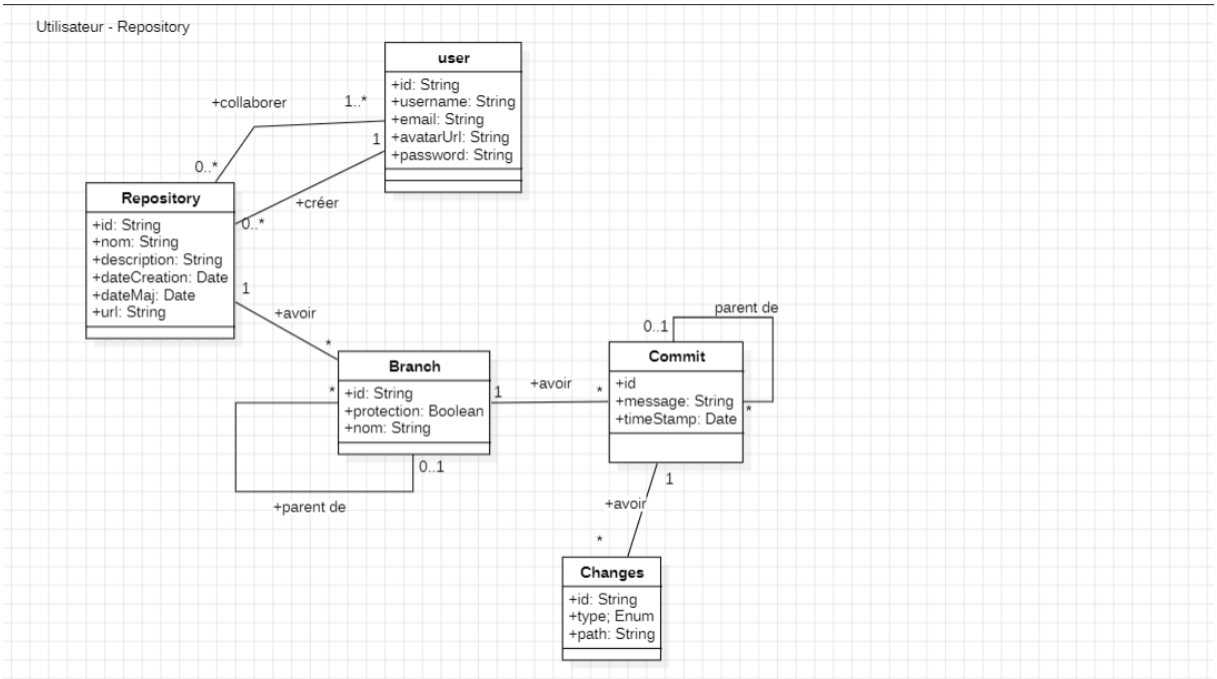
Diagramme de classes global



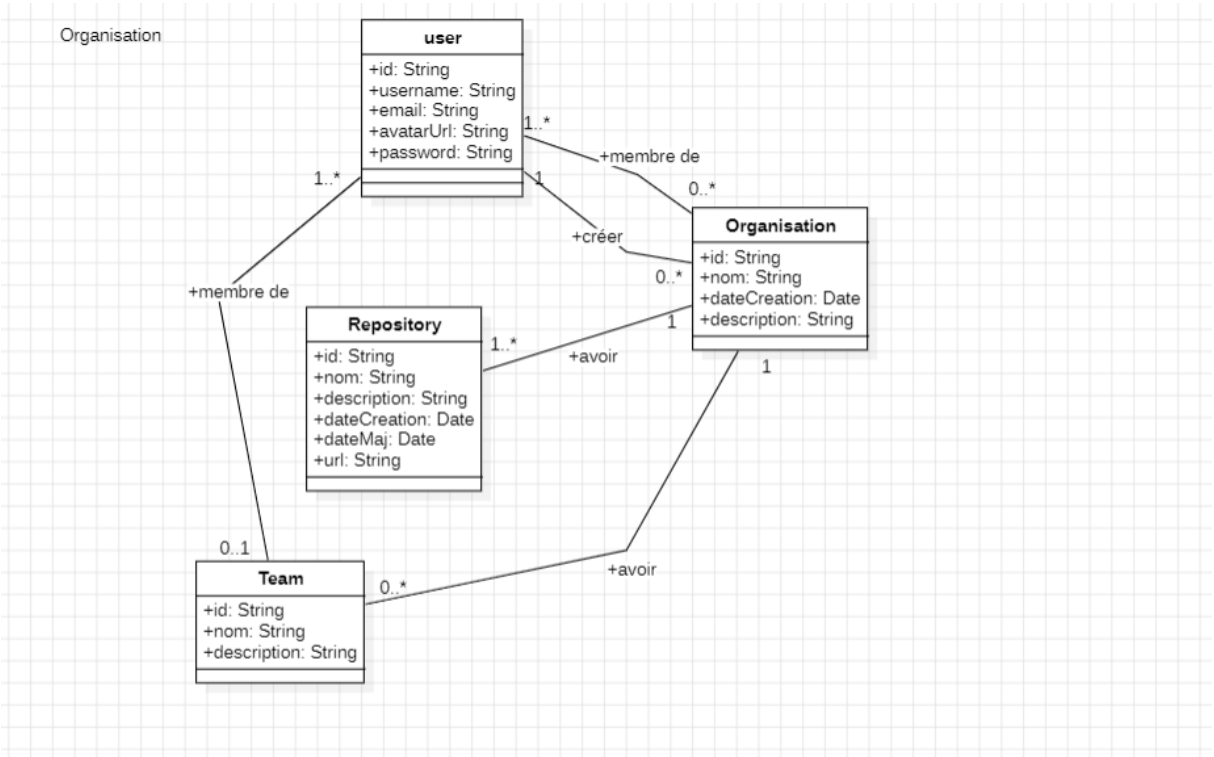
Sous diagrammes de classes

On n'a pas développé tout le diagramme de classes puisque GitHub est un très grand site d'où il contient plusieurs fonctionnalités. Voici quelques diagrammes de classes que nous avons développé :

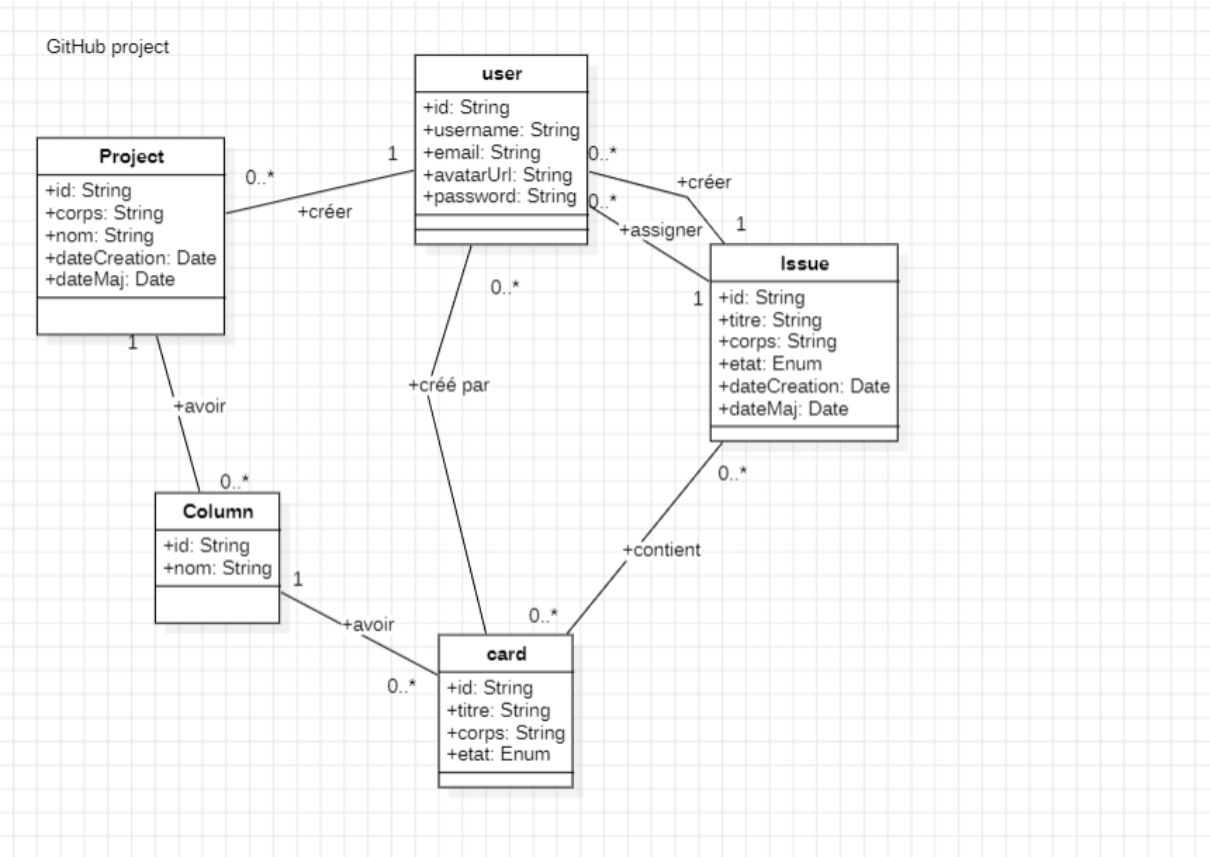
- **Utilisateur avec Repository**



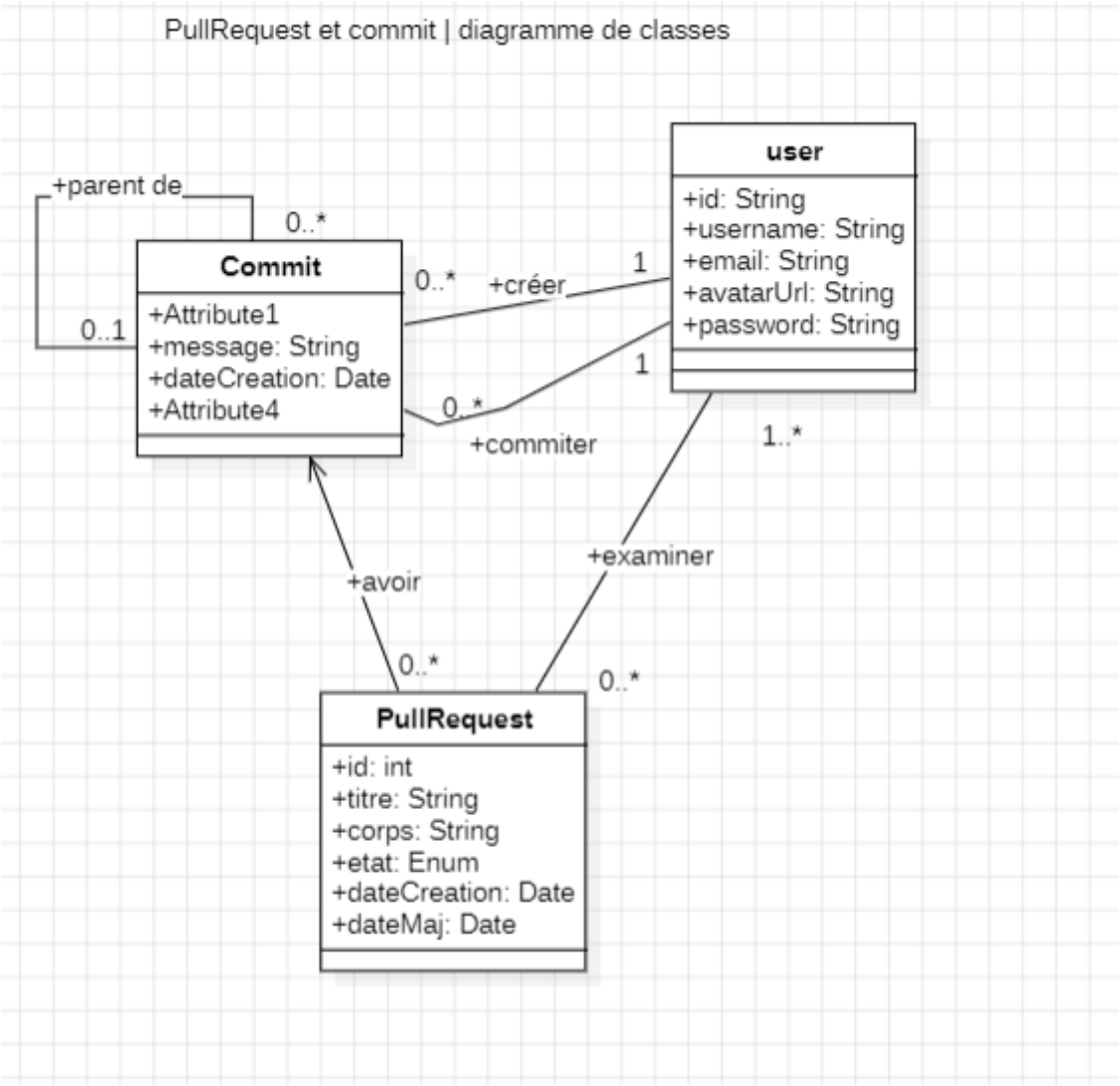
- **Organisation avec Repository**



- **L'outil projet**



- **Pull request et commit**



3. Diagramme de séquence

Diagramme de séquence pour commit

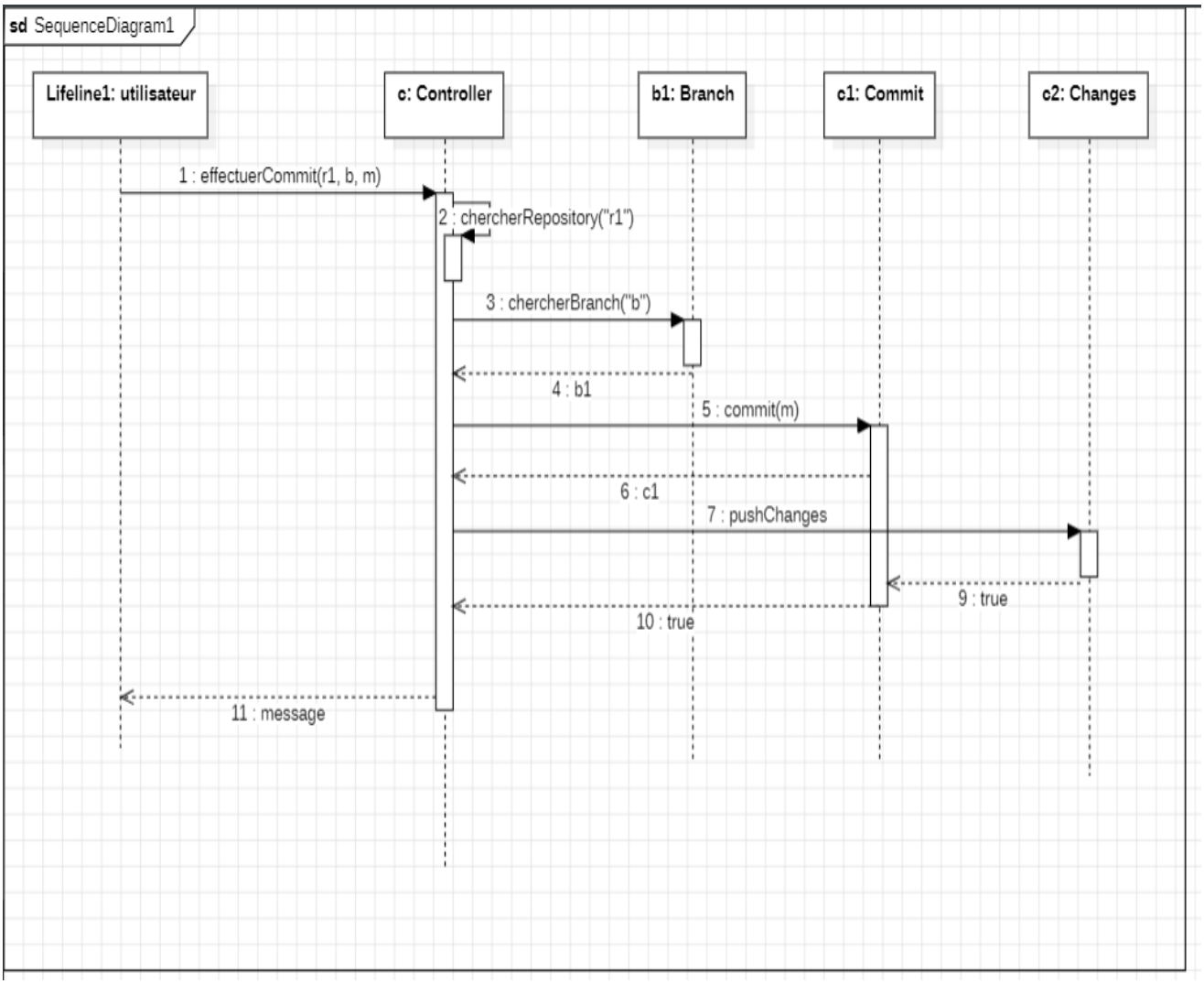


Diagramme de séquence pour une demande de collaboration