**Cahier des Charges – Github**

**DATE : 07/11/2023**

**Version : 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.

**Contexte**

**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](https://datascientest.com/cloud-computing-tout-savoir) 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 etproposer des modifications ou des améliorations.

**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 :

* **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.
* **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.
* **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.
* **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.
* **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.
* **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.

**Les enjeux**

GitHub joue un rôle crucial dans l'écosystème du développement de logiciels, et il existe plusieurs enjeux importants liés à cette plateforme. Voici les enjeux les plus importants :

* **Collaboration ouverte :** L'un des enjeux majeurs de GitHub est de promouvoir et de soutenir la collaboration ouverte. GitHub est le lieu de convergence de nombreuses communautés de développeurs, et il est essentiel de garantir un environnement de collaboration respectueux, inclusif et équitable.
* **Sécurité** : La sécurité des dépôts de code sur GitHub est cruciale. La plateforme doit faire face aux défis de la sécurité, notamment la protection contre les attaques, la gestion des vulnérabilités, et la protection des données sensibles.
* **Intégration d'outils tiers** : GitHub doit continuer à améliorer et à étendre son écosystème d'intégrations avec des outils tiers. Cela permet aux développeurs de personnaliser leur flux de travail et d'intégrer des outils de gestion de projet, d'automatisation, de CI/CD (Intégration continue / Déploiement continu) et bien d'autres.
* **Évolutivité :** Avec la croissance constante du nombre d'utilisateurs et de projets sur GitHub, l'évolutivité est un enjeu majeur. GitHub doit être en mesure de gérer efficacement des milliards de lignes de code et de garantir des performances optimales pour tous les utilisateurs.
* **Monétisation et modèle économique** : GitHub est passé d'un modèle basé sur l'open source à un modèle économique où certaines fonctionnalités sont payantes. L'équilibre entre les fonctionnalités gratuites et payantes, ainsi que la tarification, sont des enjeux importants pour GitHub.
* **Respect de la vie privée :** Avec le stockage de données sensibles et de code source confidentiel, GitHub doit maintenir des normes élevées en matière de respect de la vie privée et de sécurité des données.
* **Gouvernance des projets open source :** GitHub abrite de nombreux projets open source essentiels, et la gouvernance de ces projets peut être un enjeu. Il est important de garantir la pérennité et l'ouverture de ces projets, ainsi que d'assurer une gestion transparente des contributions.
* **Éducation et formation :** GitHub joue un rôle clé dans l'éducation et la formation des développeurs. Il doit continuer à offrir des ressources et des programmes pour aider les apprenants et les enseignants.
* **Éthique et responsabilité** : Les enjeux éthiques liés à l'utilisation de GitHub sont également importants. GitHub doit être transparent dans ses pratiques et s'engager à lutter contre les abus, tels que le code malveillant ou discriminatoire.
* **Innovation** : GitHub doit continuer à innover pour répondre aux besoins changeants des développeurs et des organisations. Cela inclut l'amélioration des outils de développement, des flux de travail, et l'adoption de nouvelles technologies.

**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.

**Les parties prenantes**

GitHub rassemble un large éventail de parties prenantes, de développeurs individuels aux grandes entreprises, en passant par les organisations open source, pour créer un écosystème de développement collaboratif dynamique. Voici les parties prenantes principales de GitHub :

* **Développeurs** : Les développeurs sont les utilisateurs principaux de GitHub. Ils contribuent au code source, créent des dépôts, ouvrent des problèmes, créent des demandes de tirage et participent activement aux projets hébergés sur la plateforme.
* **Mainteneurs de projets** : Les mainteneurs sont responsables de la gestion des dépôts et des projets sur GitHub. Ils examinent les demandes de tirage, gèrent les problèmes, définissent les orientations du projet et assurent la qualité et l'intégrité du code source.
* **Équipes de développement** : Les équipes de développement comprennent les développeurs, les testeurs, les concepteurs, les chefs de projet et d'autres rôles impliqués dans la création et la maintenance de logiciels. GitHub facilite la collaboration entre les membres de l'équipe et la gestion de projet.
* **Entreprises et organisations** : De nombreuses entreprises et organisations utilisent GitHub pour héberger leurs projets logiciels, gérer le code source, suivre les problèmes et collaborer sur des projets internes ou open source.
* **Contributeurs externes** : Les contributeurs externes sont des individus ou des organisations qui apportent des contributions à des projets open source hébergés sur GitHub. Ils n'appartiennent pas nécessairement à l'équipe principale du projet, mais contribuent à son développement.
* **Utilisateurs de logiciels** : Les utilisateurs de logiciels bénéficient également de GitHub, car de nombreux projets open source y sont hébergés. Ils peuvent signaler des problèmes, suggérer des améliorations et contribuer à l'amélioration des logiciels.
* **Gestionnaires de projet** : Les gestionnaires de projet utilisent GitHub pour suivre les progrès, planifier les tâches, gérer les ressources et coordonner les activités de développement.
* **Responsables de la sécurité** : Les responsables de la sécurité veillent à ce que les projets GitHub respectent les normes de sécurité, identifient et corrigent les vulnérabilités, et mettent en place des politiques de sécurité pour protéger les dépôts.
* **Communauté open source** : La communauté open source est une partie prenante majeure de GitHub. Elle inclut des individus et des organisations qui contribuent aux projets open source, partagent leurs connaissances et favorisent la collaboration.
* **Développeurs d'outils tiers** : Les développeurs d'outils tiers créent des applications, des extensions et des intégrations qui complètent GitHub, améliorant ainsi son utilité pour les utilisateurs.

**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 :

* **Faciliter la collaboration** : GitHub vise à faciliter la collaboration entre les développeurs en fournissant des outils de gestion de versions, des fonctionnalités de suivi des problèmes, des demandes de tirage et d'autres fonctionnalités permettant aux développeurs de travailler ensemble de manière efficace.
* **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.
* **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.
* **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.
* **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.
* **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.
* **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.
* **Promouvoir la transparence** : En fournissant des outils de suivi des problèmes publics et de gestion des projets, GitHub encourage la transparence et la participation de la communauté.
* **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.
* **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.

**Portée du projet :**

**Limites du GitHub**

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 :

* **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.
* **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.
* **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.
* **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.
* **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.
* **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.
* **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.
* **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.
* **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.
* **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.

**Portée du GitHub**

La portée de GitHub peut varier en fonction des besoins spécifiques d'un projet ou d'une organisation. Voici une liste générale des fonctionnalités et des exigences techniques que vous pourriez considérer lors de la mise en place d'un projet sur GitHub :

**Fonctionnalités :**

**1. Gestion des Référentiels (Repositories) :**

* Création et gestion de référentiels pour stocker le code source.
* Options pour les référentiels publics et privés.

**2. Contrôle de Version :**

* Utilisation de Git pour suivre les modifications du code source.
* Gestion des branches pour le développement parallèle.

**3. Collaboration et Suivi :**

* Problèmes (Issues) pour signaler les bugs, demandes de fonctionnalités, et autres tâches.
* Demandes de Tirage (Pull Requests) pour proposer et réviser les modifications du code.
* Discussions pour des conversations plus larges autour des sujets techniques.

**4. Gestion de Projet :**

* Tableaux Kanban pour suivre les tâches en cours et à faire.
* Milestones pour regrouper les problèmes et les demandes de tirage par version ou objectif.

**5. Documentation :**

* Wiki pour héberger la documentation du projet.
* Prise en charge des fichiers Markdown pour la documentation et les guides.

**6. Intégration Continue et Déploiement Continu (CI/CD) :**

* Intégration avec des outils CI/CD pour automatiser les tests et le déploiement.

**7. Sécurité :**

* Analyse statique du code pour détecter les problèmes de sécurité potentiels.
* Alertes de sécurité pour les dépendances vulnérables.

**8. Personnalisation et Extensibilité :**

* Utilisation d'intégrations et d'actions GitHub pour personnaliser le flux de travail.
* Création de badges de statut pour afficher l'état actuel du projet.

**Exigences Techniques**

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

* **Accès à Internet** : Nous devons disposer d'une connexion Internet pour accéder à la plateforme GitHub, car tout se fait en ligne.
* **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.
* **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.
* **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.
* **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.
* **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.
* **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.
* **Design et mise en page** :

**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.

**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.

**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. Ils peuvent ajouter 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.
* **Contenu:**

**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.

**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