Projet de bases de données

420-4GW-BB

**Analyse finale**

**Présentation du projet**

Travail présenté à

Jean-François Brodeur

Par

Antoine Gamache, Alex Tan et Anthony-Davis Philip

Collège de Bois-de-Boulogne

5 mars 2023

Table des matières

[Sommaire à l’exécutif 3](#_Toc134987597)

[Historique du document 4](#_Toc134987598)

[Sources 5](#_Toc134987599)

[Glossaire 6](#_Toc134987600)

[Description du projet 6](#_Toc134987601)

[Description 6](#_Toc134987602)

[Détail des besoins 6](#_Toc134987603)

[Portée 6](#_Toc134987604)

[Exigences 7](#_Toc134987605)

[Exigences non fonctionnelles 7](#_Toc134987606)

[Fonctionnement sur architecture web 7](#_Toc134987607)

[Sécurité 7](#_Toc134987608)

[Disponibilité 7](#_Toc134987609)

[Exigences fonctionnelles 7](#_Toc134987610)

[Liste des intervenants impliqués dans le projet 8](#_Toc134987611)

[Méthodologie de gestion de projet 8](#_Toc134987612)

[Contexte du projet 8](#_Toc134987613)

[Méthodologie de travail en équipe 9](#_Toc134987614)

[Philosophie de développement 9](#_Toc134987615)

[Documentation 9](#_Toc134987616)

[Instrumentation du code 9](#_Toc134987617)

[Modèle de la base de données 9](#_Toc134987618)

[Clé de lecture 10](#_Toc134987619)

[Interfaces utilisateur 11](#_Toc134987620)

[Interface utilisateur 1 – Page de connexion et inscription 11](#_Toc134987621)

[Interface utilisateur 2 – Page de sélection de compte google 12](#_Toc134987622)

[Interface utilisateur 3 – 404 12](#_Toc134987623)

[Interface utilisateur 4 – Page d’accueil 13](#_Toc134987624)

[Interface utilisateur 5 – Liste des tâches 13](#_Toc134987625)

[Interface utilisateur 6 – Création d’une tâche 14](#_Toc134987626)

[Interface utilisateur 7 – Modification d’une tâche 14](#_Toc134987627)

[Interface utilisateur 8 – Liste des projets 15](#_Toc134987628)

[Interface utilisateur 9 – Création de projet 15](#_Toc134987629)

[Interface utilisateur 10 – Modification de projet 16](#_Toc134987630)

[Interface utilisateur 11 –Création d’événements 16](#_Toc134987631)

[Interface utilisateur 11 –Création d’événements 17](#_Toc134987632)

[Interface utilisateur 12 – Modification d’événements 17](#_Toc134987633)

[Interface utilisateur 13 – Organisations 18](#_Toc134987634)

[Interface utilisateur 13.5 – Organisations : utilisateurs 18](#_Toc134987635)

[Interface utilisateur 14 – Créer organisation 19](#_Toc134987636)

[Interface utilisateur 15 –Modifier organisation 19](#_Toc134987637)

[Interface utilisateur 16 – Invité à une organisation 20](#_Toc134987638)

[Cas d’utilisation 21](#_Toc134987639)

[Diagramme des cas d’utilisation 22](#_Toc134987640)

[Cas d’utilisation 1 – Se connecter 23](#_Toc134987641)

[Cas d’utilisation 2 – Parcourir les tâches 24](#_Toc134987642)

[Cas d’utilisation 3 – Parcourir les évènements 25](#_Toc134987643)

[Cas d’utilisation 4 – Créer une tâche 26](#_Toc134987644)

[Cas d’utilisation 5 – Marquer une tâche comme terminée 27](#_Toc134987645)

[Cas d’utilisation 6 – Parcourir les projets 28](#_Toc134987646)

[Cas d’utilisation 7 – Créer un projet 29](#_Toc134987647)

[Cas d’utilisation 8 – Créer une organisation 30](#_Toc134987648)

[Annexe A – Principe de conception (SOLID DRY KISS) 31](#_Toc134987649)

[S.O.L.I.D 31](#_Toc134987650)

[S : Single Responsibility Principle (Responsabilité unique) 31](#_Toc134987651)

[O : Open/Closed (Ouvert/Fermé) 31](#_Toc134987652)

[L : Liskov substitution principle (Substitution de Liskov) 31](#_Toc134987653)

[I : Interface segregation principle (Ségrégation des interfaces) 31](#_Toc134987654)

[D : Dependency inversion principle (Inversion des dépendances) 31](#_Toc134987655)

[D.R.Y (Don’t repeat yourself) 31](#_Toc134987656)

[K.I.S.S (Keep it super simple) 32](#_Toc134987657)

# Sommaire à l’exécutif

Notre équipe, 727-apps, développe 727-tracker, une application web de gestion de projet pour aider les organisations lors de leur projet. Dans le logiciel, une page qui contient chacune les organisations, les projets et les tâches seront présents. Il y a aussi une page pour les notifications et une page pour les événements. Veuillez consulter le guide utilisateur afin d’en apprendre plus sur les différentes fonctions de l’application web.

Le projet suit une approche AGILE et TDD lors du développement. Cette approche a été favorisé pour permettre un déploiement stable et un développement plus efficace. Lors du développement, plusieurs outils comme Github, Discord, Clockify et Trello seront utilisés. Ces outils facilitent la gestion du projet au niveau de la communication, de la séparation des tâches ou de la réalisation des tâches assignées à chacun.

Ce document contient aussi le modèle de la base de données et les interfaces de l’application web. Ils permettent de représenter la structure des données tout autant au niveau du backend que le frontend.

# Historique du document

Voici un historique des modifications apportées à ce document entre ses différentes versions.

|  |  |  |
| --- | --- | --- |
| Version | Date | Changements |
| 1 | 2023-01-26 | Version initiale. |
| 2 | 2023-03-05 | Version livrable 2 |
| 3 | 2023-05-14 | Version finale |
|  |  |  |
|  |  |  |

Tableau 1 : historique du document.

# Sources

Ce tableau rassemble les différentes sources desquelles nous avons pris nos informations pour ce document.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Code | Nom | Date | Description | Location |
| S-001 | Compte-rendu de l’entrevue du 1 septembre 2017 | 2017-09-01 | Les notes de l’entrevue avec le représentant du client, du 25 aout 2016. | Annexe A |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

Tableau 2 : les sources du document.

# Glossaire

# Description du projet[[1]](#footnote-2)

Grâce aux demandes que vous, M. ABC, nous avez demandées afin d’effectuer la conception de votre produit, nous avons réussi à vous rédiger ce document afin de pouvoir vous faire la présentation des différents aspects nécessaires à la conception de votre site web.

Ce document ira en détail sur la description précise du projet, les exigences, le style visuel général, le fonctionnement du système informatique, la structure du stockage des données, etc.

## Description

Le projet consiste d’une application web ayant pour but de faciliter la gestion d’un ou plusieurs projets. Le système aidera un utilisateur en créant une interface virtuelle simple dont laquelle il pourra connaitre ses échéanciers, connaitre ses collaborateurs et accomplir ses tâches de manière efficace et rapide.

## Détail des besoins

Plusieurs travailleurs éprouvent ou peuvent éprouvés un désir de procrastiner et cela peut causer une insatisfaction du côté du client lorsqu’ils reçoivent leurs produits finals. Ce projet a été mis de l’avant afin d’empêcher ce désir de procrastination et d’empêcher l’insatisfaction des clients.

Le système cherche à satisfaire ce besoin. Cela permettra une organisation d’un projet personnel ou professionnel de manière simple et qui encourage à la productivité.

## Portée

La portée du projet se limite à ces aspects :

* La gestion de projet à l’aide de tâches assignées à un utilisateur.
* La gestion de tâches et de leurs accomplissements.
* La liaison d’organisation à des projets.
* La liaison des projets à des tâches
* La connexion entre les tâches, les projets et les évènements au calendrier Google

L’étendue du projet consiste du bon fonctionnement d’une interface qui pourra créer un ou des projets contenant plusieurs utilisateurs, étant les membres de l’équipe du projet, et pouvant se séparer et accomplir des tâches.

Toutefois, notre application web ne fera pas ce qui suit :

* Elle ne supporte pas le partage de fichier de travail
* Elle ne s’occupe pas de l’authentification. C’est l’API de Google qui le fera.
* Elle ne prend pas en note le temps de travail.
* Elle ne permet pas de faire de la gestion d’entreprise poussée comme le rendement salarial.

# Exigences

Dans cette section, nous allons vous décrire les exigences que le système devra respecter.

## Exigences non fonctionnelles

### Fonctionnement sur architecture web

Le système devra être rapide. En effet, la rapidité de chargement, par exemple, impactera considérablement la productivité de nos utilisateurs.

### Sécurité

La sécurité devra être absolument impeccable dans notre système puisque, dans un système de gestion de projet, il y aura probablement des informations sur des projets secrets de différentes organisations. Si une fuite de données arrive, cela pourrait coûter cher à nos clients. L’encryption pourrait être une solution envisageable. L’authentification se fait via l’API de Google. La sécurité des comptes sera donc géré par Google.

### Disponibilité

Le système devra être disponible le plus possible. Si notre site n’est pas accessible, les utilisateurs ne pourront plus planifier leurs projets, ce qui pourrait ralentir leur productivité. Dans un contexte professionnel, c’est de l’argent perdu.

## Exigences fonctionnelles

Un utilisateur pourra aussi créer une organisation et associé d’autres utilisateurs à celle-ci. Il pourra aussi créer un projet et créer des tâches à l’intérieur de celui-ci. Il pourra associer des utilisateurs à ses projets et ses tâches et donner certains droits à certains d’entre eux. Il pourra aussi attacher des évènements à ses projets. Il pourra aussi faire la gestion de ses tâches et de ses projets. Finalement, un utilisateur lié à une tâche pourra la marquer comme complétée.

# Liste des intervenants impliqués dans le projet

1. Philip Anthony-Davis : Principal développeur d’interface, liaison entre le logiciel et Google, installation de l’environnement de développement.
2. Antoine Gamache : Développeur frontend et backend, conception des modèles de base de données MySQL et MongoDB, rédaction de document.
3. Alex Tan : Principal développeur backend, rédaction de test et de méthodes d’aide, rédaction de document.
4. Jean-François Brodeur : Client du projet.

# Méthodologie de gestion de projet

La gestion du projet s’est effectuée à l’aide de plusieurs outils. Pour toutes nouvelles fonctions à implémenter ou problèmes à régler, Discord a été le logiciel principalement utilisé pour communiquer entre les membres de l’équipe. Ensuite, l’utilisation de Github a été essentielle afin de gérer les différentes versions du projet. De plus, l’utilisation de Trello et de Clockify a permis de gérer les différentes étapes de développement et le temps consacré à la réalisation de chaque tâche.

# Contexte du projet

Le diagramme ci-dessous est un diagramme de contexte. Ce diagramme permet de s’assurer de la bonne compréhension de la structure du système à haut niveau avant de progresser plus loin. Il contient les acteurs primaires et secondaires présents dans le système.

A picture containing table

Description automatically generated

Chacune des catégories ont un nombre d’acteurs différents :

-Les acteurs primaires sont les utilisateurs, les chefs d’équipes et les administrateurs

-L’acteur secondaire est l’API de google

Figure 1 : diagramme de contexte

# Méthodologie de travail en équipe

Le travail d’équipe est effectué efficacement à l’aide de plusieurs outils comme Discord pour communiquer, GitHub pour gérer les différentes versions du projet ou Trello pour l’assignation des tâches. La majorité de l’équipe est assigné à la programmation frontend, tandis que le reste s’occupe du backend. Cependant, lorsqu’une tâche urgente est nécessaire en backend, le côté frontend peut apporter une certaine aide afin d’accélérer le développement du projet. L’équipe se réunisse officiellement deux fois par semaine afin de déterminer les tâches de chacun ou de discuter des problèmes rencontrés, mais discute souvent sur Discord pour quelconque clarification ou problème.

# Philosophie de développement

Le développement de ce projet suit une approche AGILE. En utilisant cette approche, le projet est travaillé à travers de nombreuses petites étapes planifiées à la place d’un déploiement immédiat du logiciel au complet. Au niveau du développement, des tests et du déploiement de cette approche, le projet suit l’approche de développement TDD. Cette approche permet d’effectuer des tests et d’améliorer le code si une erreur se produit. De plus, pour assurer un déploiement plus stable, le projet utilise des types lors du développement des fonctions du projet grâce à TypeScript.

# Documentation

Afin de voir la documentation utilisateur de notre logiciel, veuillez consulter le document annexe à l’analyse.

# Instrumentation du code

Afin d’observer les changements, erreurs ou résultats du code, Winston a été utilisé pour les « logger ». À l’aide de ces « logs », le développement du projet est plus rapide. Ceci permet donc de mieux observer la performance du code et d’assurer un déploiement stable, sécuritaire et performant du logiciel. Nous avons aussi créé une fonction afin de facilement envoyer des messages clairs à la console.

# Modèle de la base de données

Voici le diagramme de la base de données que nous allons utiliser. Il a été créé à partir de la fonction Data Modeler dans le logiciel SqlDeveloper.

Ce diagramme couvre les principales sortes de données que nous aurons à enregistrer. D’abord, les informations sur les utilisateurs doivent nécessairement l’être. Même si nous utiliserons l’API de Google afin d’authentifier les usagers, il sera nécessaire de garder quelques informations afin de ne pas surcharger le système de demande vers l’API de Google. Les projets et les tâches devront aussi être enregistrer afin de pouvoir afficher un tableau de bord à jour. Il y a aussi les évènements (events) et les organisations. Le reste des tables servent principalement au bon fonctionnement du système et ne représentent pas quelque chose de concret dans celui-ci.

Diagram

Description automatically generated

Figure 2 : diagramme d'entités-relations

## Clé de lecture

Une organisation peut avoir un ou plusieurs membres.

Un membre doit faire partie d’une seule organisation.

Un membre doit représenter un seul utilisateur.

Un utilisateur peut être un ou plusieurs membres.

Un utilisateur peut être associé à une ou plusieurs assigned\_task.

Un utilisateur peut être associé à un ou plusieurs user\_project.

Un user\_project doit faire partie d’un seul utilisateur.

Un user\_project doit faire partie d’un seul projet.

Un projet peut avoir un ou plusieurs user\_project.

Un projet peut avoir un ou plusieurs évènements.

Un projet peut avoir une ou plusieurs tâches.

Une tâche doit faire partie d’un seul projet.

Une tâche peut être associé à une ou plusieurs assigned\_task.

Une assigned\_task doit être liée à une seule tâche.

Une assigned\_task doit être associée à un seul utilisateur

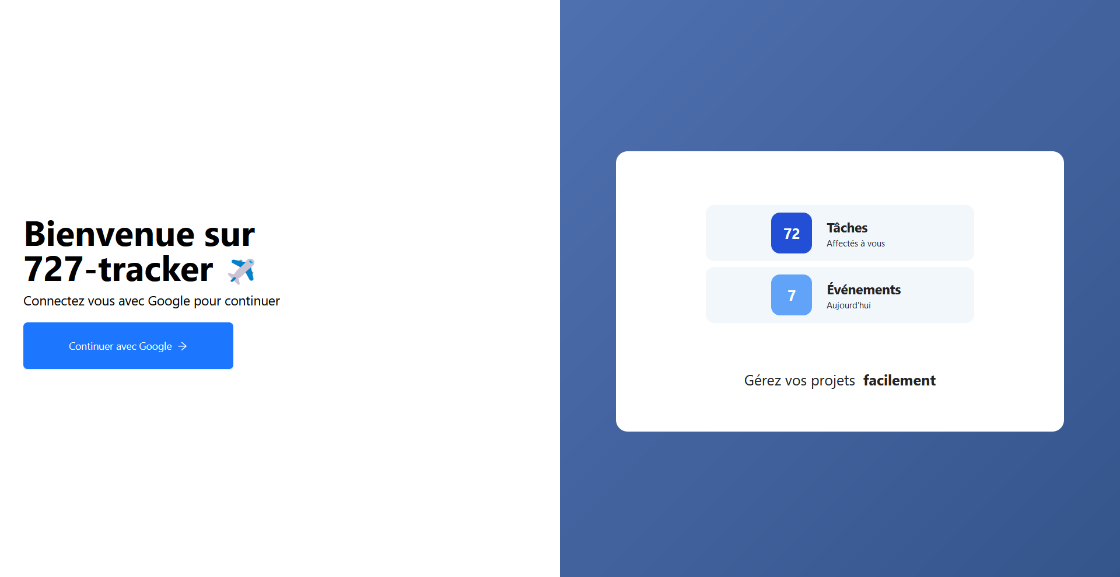
Un évènement doit être associé à un seul projet.

# Interfaces utilisateur

Voici nos interfaces utilisateur (ou bien UI). Toutes les pages comportent aussi des animations lors du chargement de l’information contenue. Certains éléments peuvent également changer dépendamment du contexte de la page.

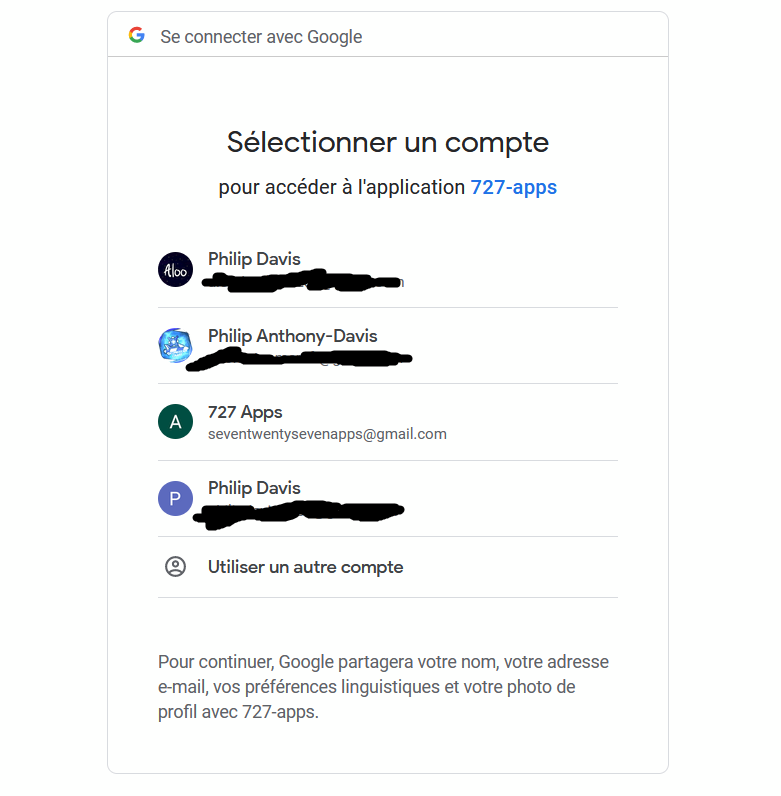
## Interface utilisateur 1 – Page de connexion et inscription

Cette page agit comme introduction vers le site web. Il est impossible d’accéder à d’autres pages sans être connecté. L’utilisateur peut cliquer sur le bouton ‘continuer avec google’ pour être redirigé vers une page de connexion google.



## Interface utilisateur 2 – Page de sélection de compte google

Cette page est offerte par google cloud. Elle permet à l’utilisateur de sélectionner le compte qu’il veut utiliser pour se connecter à notre site web.



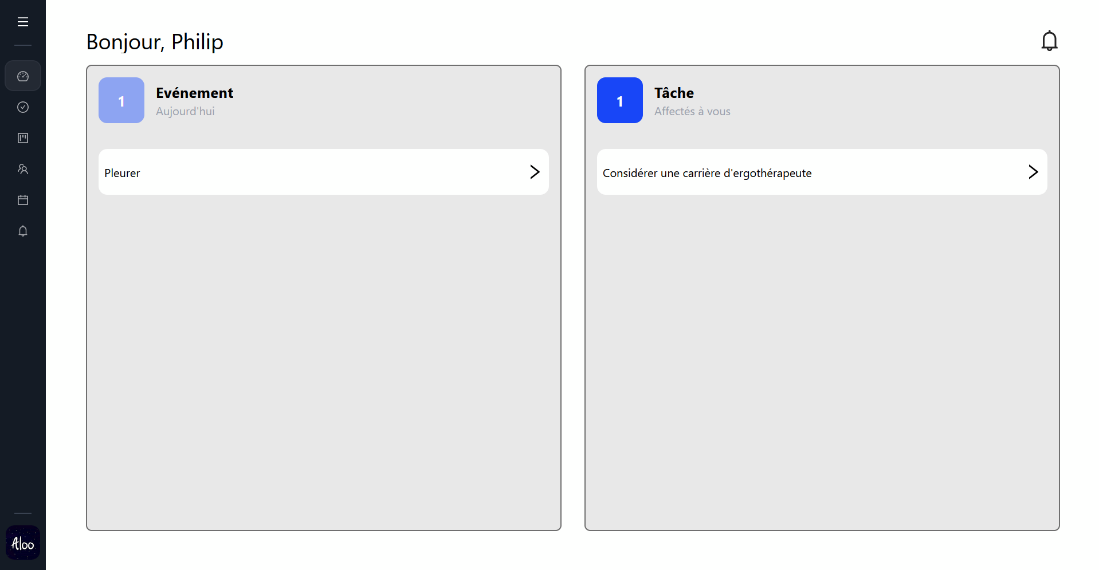
## Interface utilisateur 3 – 404

Cette page s’affiche lorsque l’utilisateur tente d’ouvrir une page qui n’existe pas.



## Interface utilisateur 4 – Page d’accueil

Cette page est le tableau de bord de l’utilisateur. Elle permet à l’utilisateur de rapidement voir ses événements et ses tâches. Chaque élément peut être sélectionné pour naviguer vers la page correspondante. À gauche, il y a une barre de navigation permettant de naviguer vers les différentes sections de notre site web. Cette barre de navigation peut être agrandie et elle se retrouve sur toutes les pages de notre site à quelques exceptions.



## Interface utilisateur 5 – Liste des tâches

Cette page affiche toutes les tâches de l’utilisateur. Il peut également choisir de filtrer par projet, et d’afficher la page sous une autre forme, avec des cartes plutôt qu’une liste.



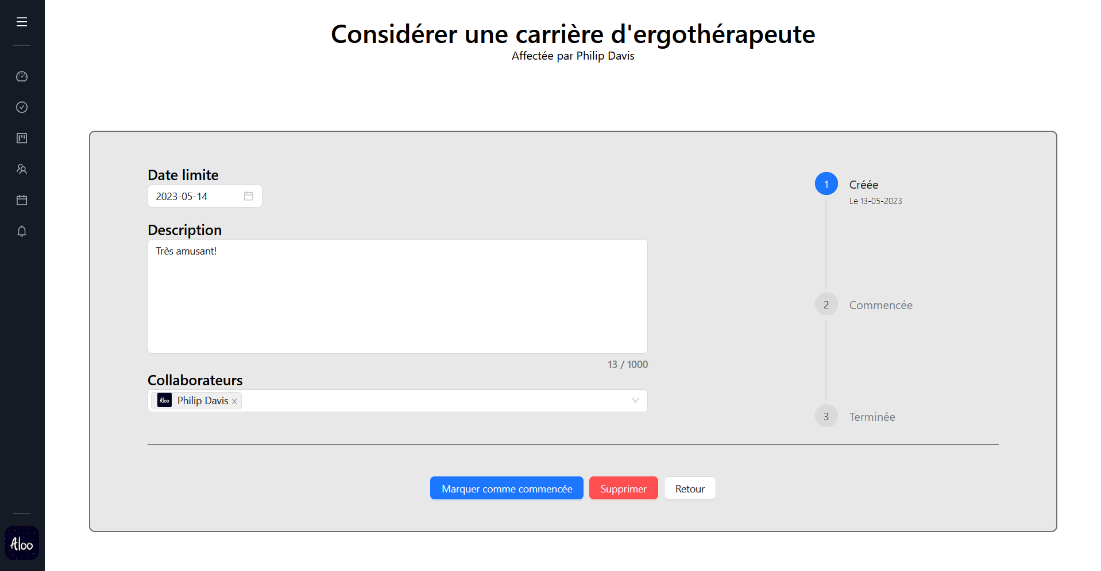
## Interface utilisateur 6 – Création d’une tâche

Cette page permet à l’utilisateur de créer une tâche. Il peut choisir un titre, une description, une date limite, des collaborateurs et un projet associé. Une prévisualisation de la tâche est également offerte si l’espace le permet.



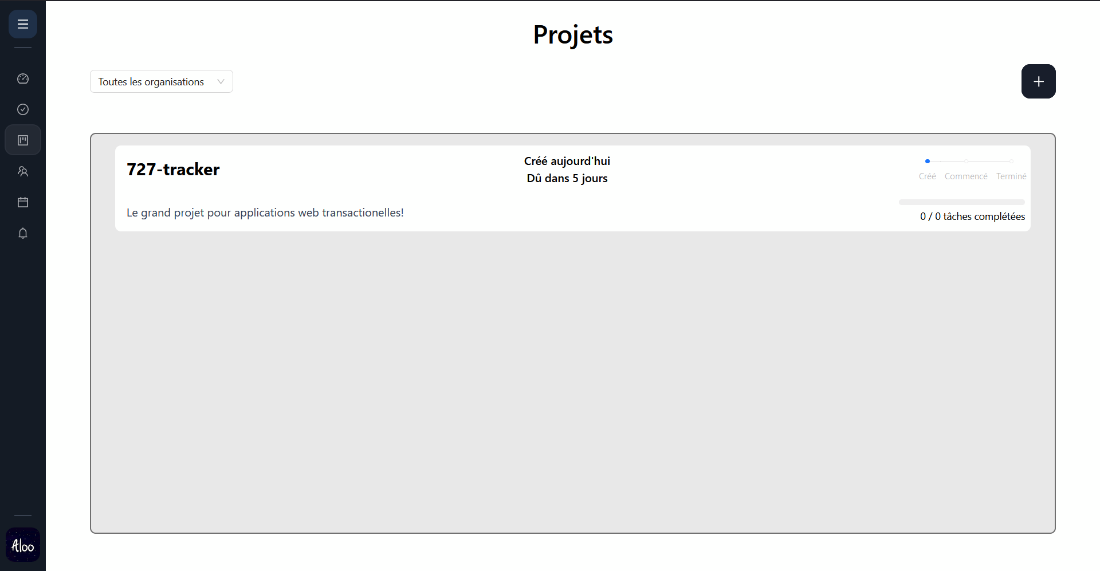
## Interface utilisateur 7 – Modification d’une tâche

Cette page permet à l’utilisateur de modifier une tâche déjà créée s’il en est le propriétaire. Il peut changer la description, les collaborateurs, la date limite et le statut. Il peut également supprimer la tâche.



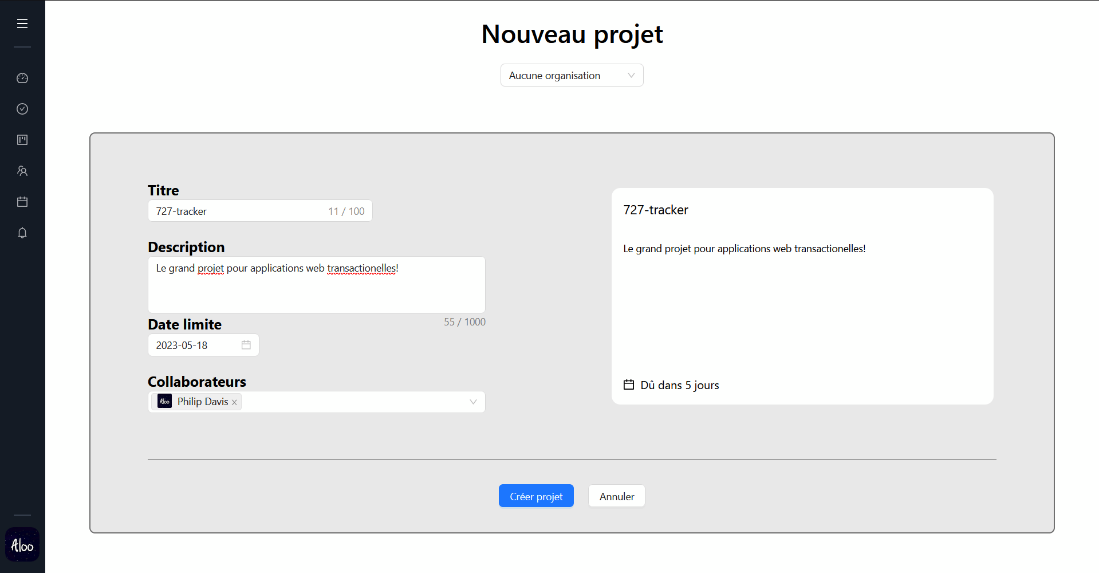
## Interface utilisateur 8 – Liste des projets

Cette page permet de voir tous les projets auxquels ils sont affectés. Ils peuvent aussi voir des informations utiles comme le statut du projet, le nombre de tâches complétées, la date de création et la date limite. Il est également possible de filtrer les projets par organisations.



## Interface utilisateur 9 – Création de projet

Cette page permet à l’utilisateur de créer un projet. Cette page fonctionne exactement de la même façon que la page de création de tâche. La seule différence est qu’on crée un projet relié à une organisation, et non une tâche reliée à un projet.



## Interface utilisateur 10 – Modification de projet

Cette page permet à l’utilisateur de modifier un projet qui lui appartient. Elle fonctionne de la même façon que la page de modification de tâches. Il y a aussi un bouton « voir tâches » qui permet d’afficher toutes les tâches appartenant à ce projet.

A screenshot of a computer

Description automatically generated with medium confidence

## Interface utilisateur 11 –Création d’événements

Cette page permet à l’utilisateur de voir ses événements dans un calendrier. L’utilisateur peut sélectionner une date sur le calendrier ou un mois et voir tous les événements s’y trouvant.

A screenshot of a calendar

Description automatically generated with medium confidence

## Interface utilisateur 11 –Création d’événements

Cette page permet à l’utilisateur de créer un nouvel événement. Cette page fonctionne de la même façon que la page de création de tâche, mais contient deux dates (début et fin).

A screenshot of a computer

Description automatically generated with medium confidence

## Interface utilisateur 12 – Modification d’événements

Cette page permet à l’utilisateur de voir un événement, le modifier et le supprimer. Les dates et les collaborateurs peuvent être changés.

A screenshot of a computer

Description automatically generated with medium confidence

## Interface utilisateur 13 – Organisations

Cette page permet à l’utilisateur de voir toutes les organisations dans lesquelles il se trouve accompagnée de statistiques qui peuvent être sélectionnées pour aller sur la page correspondante.

A screenshot of a computer

Description automatically generated with medium confidence

Interface utilisateur 13.5 – Organisations : utilisateurs  
Lorsque l’utilisateur sélectionne le bouton « utilisateurs » d’une organisation, il peut voir et gérer les membres, créer un lien d’invitation, quitter l’organisation ou la supprimer.

A screenshot of a computer

Description automatically generated with medium confidence

## Interface utilisateur 14 – Créer organisation

Cette page permet à l’utilisateur de créer une organisation. Un titre et une image doivent être donnés pour créer l’organisation.

A screenshot of a computer

Description automatically generated with medium confidence

## Interface utilisateur 15 –Modifier organisation

Cette page permet à l’utilisateur de modifier une organisation sur laquelle il a des permissions.

A screenshot of a computer

Description automatically generated with medium confidence

## Interface utilisateur 16 – Invité à une organisation

Cette page, qui est accessible à partir d’un lien unique envoyé par une personne ayant des permissions dans une organisation, permet à l’utilisateur d’accepter une invitation vers une organisation.

A screenshot of a computer

Description automatically generated with medium confidence

# Cas d’utilisation

Le diagramme suivant résume les interactions entre les utilisateurs et le système

Voici le résumé du diagramme :

1. Un utilisateur peut se connecter :
   1. Le processus inclura une authentification par l’API de google afin de permettre la connexion.
2. Une fois connecté, un utilisateur peut :
   1. Parcourir les projets ou crée un projet
      1. Lors de cet accès, il pourra effectuer plusieurs actions comme modifier l’état des tâches ou consulter le calendrier.
   2. Se déconnecter de son compte
   3. Crée une tâche
3. Un chef d’équipe peut :
4. Expulser un membre de l’équipe
5. Assigner une tâche
6. Un administrateur peut :
   1. Supprimer une organisation
   2. Supprimer une ou plusieurs tâches
   3. Modifier une tâche
   4. Supprimer un projet

## Diagramme des cas d’utilisation

Diagram, engineering drawing

Description automatically generated

Figure X : diagramme de cas d’utilisation

## Cas d’utilisation 1 – Se connecter

*Se référer à la figure 3 pour une idée de l’interface associée.*

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | CU-06 | | **Nom du cas d’utilisation** | | Se connecter | | **Version** | 1.0 |
| **Brève description** | | | Un utilisateur souhaite se connecter à son compte | | | | | |
| **Acteurs** | | | Utilisateur  L’API de Google | | | | | |
| **Exigences spéciales** | | | | | | | | |
| Exigence spéciale 1. | | | L’utilisateur doit avoir un compte Google. | | | | | |
| **Pré-conditions** | | | | | | | | |
| **Post-conditions** | | | | | | | | |
| Post-condition 1. | | | L’utilisateur arrive sur la page Home. | | | | | |
| **Flux d’évènements** | | | | | | | | |
| Flux de base | | | 1. L’utilisateur clique sur le bouton pour se connecter/créer un compte. 2. L’API de Google se charge de l’authentification 3. Si le compte Google n’est relié à aucun compte dans la base de données, le système en crée un nouveau 4. L’utilisateur est connecté au serveur | | | | | |
| Flux alternatif 1 :  Pas de compte Google | | | * + - 1. L’API de Google n’affiche aucun compte       2. L’utilisateur n’a pas d’autre choix que de soit annuler où fermer la page       3. Le système annule tout enregistrement dans la base de données | | | | | |
| **Historique des versions** | | | | | | | | |
| 2017-09-01 | | V1.0 | | Rédaction initiale. | | Antoine Gamache | | |

## Cas d’utilisation 2 – Parcourir les tâches

*Se référer à la figure 4, 5 ou 6 pour une idée de l’interface associée.*

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | CU-06 | | **Nom du cas d’utilisation** | | Parcourir les tâches | | **Version** | 1.0 |
| **Brève description** | | | L’utilisateur souhaite obtenir plus de détail sur une tâche | | | | | |
| **Acteurs** | | | Utilisateur | | | | | |
| **Exigences spéciales** | | | | | | | | |
| **Pré-conditions** | | | | | | | | |
| Pré-condition 1. | | | L’utilisateur doit être connecté | | | | | |
| **Post-conditions** | | | | | | | | |
| Post-condition 1. | | | L’utilisateur peut obtenir des informations sur une tâche | | | | | |
| **Flux d’évènements** | | | | | | | | |
| Flux de base | | | 1. L’utilisateur clique sur la tâche voulu  2. Le système le dirige vers la page de la tâche | | | | | |
| Flux alternatif 1 :  Manque de tâche | | | 1. Aucune tâche n’est attribuée à l’utilisateur  2. Le système affiche un message d’erreur et invite l’utilisateur de revenir à la page Tache | | | | | |
| **Historique des versions** | | | | | | | | |
| 2017-09-01 | | V3.0 | | Rédaction finale. | | Antoine Gamache | | |

## Cas d’utilisation 3 – Parcourir les évènements

*Se référer à la figure 4 ou pour une idée de l’interface associée.*

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | CU-06 | | **Nom du cas d’utilisation** | | Parcourir les évènements | | **Version** | 1.0 |
| **Brève description** | | | L’utilisateur souhaite obtenir plus de détail sur un évènement | | | | | |
| **Acteurs** | | | Utilisateur  API Google | | | | | |
| **Exigences spéciales** | | | | | | | | |
| **Pré-conditions** | | | | | | | | |
| Pré-condition 1. | | | L’utilisateur doit être connecté | | | | | |
| **Post-conditions** | | | | | | | | |
| Post-condition 1. | | | L’utilisateur peut obtenir des informations sur un évènement | | | | | |
| **Flux d’évènements** | | | | | | | | |
| Flux de base | | | 1. L’utilisateur clique sur l’évènement voulu  2. Le système dirige l’utilisateur vers la page des évènements | | | | | |
| Flux alternatif 1 :  Manque d’évènement | | | 1. Aucun évènement n’est attribué à l’utilisateur  2. Le système affiche qu’il n’y a pas d’évènement | | | | | |
| **Historique des versions** | | | | | | | | |
| 2017-09-01 | | V3.0 | | Rédaction finale. | | Antoine Gamache | | |

## Cas d’utilisation 4 – Créer une tâche

*Se référer à la figure 5 et 7 pour une idée de l’interface associée.*

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | CU-06 | | **Nom du cas d’utilisation** | | Créer une tâche | | **Version** | 1.0 |
| **Brève description** | | | L’utilisateur souhaite créer une tâche | | | | | |
| **Acteurs** | | | Utilisateur | | | | | |
| **Exigences spéciales** | | | | | | | | |
| **Pré-conditions** | | | | | | | | |
| Pré-condition 1. | | | L’utilisateur doit être connecté | | | | | |
| **Post-conditions** | | | | | | | | |
| Post-condition 1. | | | Une tâche est créée dans le système | | | | | |
| **Flux d’évènements** | | | | | | | | |
| Flux de base | | | 1. L’utilisateur clique sur le bouton +  2. La système le dirige vers la page « Créer tâche »  3. L’utilisateur remplie les informations nécessaires et confirme  4. Le système crée la tâche et envoie l’utilisateur à la page précédente | | | | | |
| Flux alternatif 1 :  Champ invalide | | | 1. Les champs remplies par l’utilisateur sont invalides  2. Le système ne crée pas la tâche et envoie un message d’erreur indiquant qu’il reste des champs importants à remplir | | | | | |
| **Historique des versions** | | | | | | | | |
| 2017-09-01 | | V1.0 | | Rédaction initiale. | | Antoine Gamache | | |

## Cas d’utilisation 5 – Marquer une tâche comme terminée

*Se référer à la figure 6 pour une idée de l’interface associée.*

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | CU-06 | | **Nom du cas d’utilisation** | | Marquer une tâche comme terminée | | **Version** | 1.0 |
| **Brève description** | | | L’utilisateur souhaite marquer une tâche comme terminée | | | | | |
| **Acteurs** | | | Utilisateur | | | | | |
| **Exigences spéciales** | | | | | | | | |
| **Pré-conditions** | | | | | | | | |
| Pré-condition 1. | | | L’utilisateur doit être connecté | | | | | |
| Pré-condition 2. | | | L’utilisateur doit sélectionner une tâche | | | | | |
| **Post-conditions** | | | | | | | | |
| Post-condition 1. | | | L’état de la tâche est modifié | | | | | |
| **Flux d’évènements** | | | | | | | | |
| Flux de base | | | 1. L’utilisateur clique le bouton pour terminer une tâche  2. Le système modifie la tâche comme étant terminé. | | | | | |
| **Historique des versions** | | | | | | | | |
| 2017-09-01 | | V3.0 | | Rédaction finale. | | Antoine Gamache | | |

## Cas d’utilisation 6 – Parcourir les projets

*Se référer à la figure 8 et 9 pour une idée de l’interface associée.*

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | CU-06 | | **Nom du cas d’utilisation** | | Parcourir les projets | | **Version** | 1.0 |
| **Brève description** | | | L’utilisateur souhaite obtenir plus de détail sur un projet | | | | | |
| **Acteurs** | | | Utilisateur | | | | | |
| **Exigences spéciales** | | | | | | | | |
| **Pré-conditions** | | | | | | | | |
| Pré-condition 1. | | | L’utilisateur doit être connecté | | | | | |
| **Post-conditions** | | | | | | | | |
| Post-condition 1. | | | L’utilisateur peut obtenir des informations sur un projet | | | | | |
| **Flux d’évènements** | | | | | | | | |
| Flux de base | | | 1. L’utilisateur clique sur le projet voulu  2. Le système le dirige vers la page du projet | | | | | |
| Flux alternatif 1 :  Manque de projet | | | 1. Aucun projet n’est attribué à l’utilisateur  2. Le système affiche un message d’erreur et invite l’utilisateur de revenir à la page Projets | | | | | |
| **Historique des versions** | | | | | | | | |
| 2017-09-01 | | V3.0 | | Rédaction finale. | | Antoine Gamache | | |

## Cas d’utilisation 7 – Créer un projet

*Se référer à la figure 8 et 10 pour une idée de l’interface associée.*

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | CU-06 | | **Nom du cas d’utilisation** | | Créer un projet | | **Version** | 1.0 |
| **Brève description** | | | L’utilisateur souhaite créer un projet | | | | | |
| **Acteurs** | | | Utilisateur | | | | | |
| **Exigences spéciales** | | | | | | | | |
| **Pré-conditions** | | | | | | | | |
| Pré-condition 1. | | | L’utilisateur doit être connecté | | | | | |
| **Post-conditions** | | | | | | | | |
| Post-condition 1. | | | Un projet est créé dans le système | | | | | |
| **Flux d’évènements** | | | | | | | | |
| Flux de base | | | 1. L’utilisateur clique sur le bouton +  2. La système le dirige vers la page « Créer projet »  3. L’utilisateur remplie les informations nécessaires et confirme  4. Le système crée le projet et envoie l’utilisateur à la page précédente | | | | | |
| Flux alternatif 1 :  Champ invalide | | | 1. Les champs remplies par l’utilisateur sont invalides  2. Le système ne crée pas le projet et envoie un message d’erreur indiquant qu’il reste des champs importants à remplir | | | | | |
| **Historique des versions** | | | | | | | | |
| 2017-09-01 | | V1.0 | | Rédaction initiale. | | Antoine Gamache | | |

## Cas d’utilisation 8 – Créer une organisation

*Se référer à la figure 12 et 13 pour une idée de l’interface associée.*

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | CU-06 | | **Nom du cas d’utilisation** | | Créer une organisation | | **Version** | 1.0 |
| **Brève description** | | | L’utilisateur souhaite créer une organisation | | | | | |
| **Acteurs** | | | Utilisateur | | | | | |
| **Exigences spéciales** | | | | | | | | |
| **Pré-conditions** | | | | | | | | |
| Pré-condition 1. | | | L’utilisateur doit être connecté | | | | | |
| **Post-conditions** | | | | | | | | |
| Post-condition 1. | | | Une organisation est créée dans le système | | | | | |
| **Flux d’évènements** | | | | | | | | |
| Flux de base | | | 1. L’utilisateur clique sur le bouton +  2. La système le dirige vers la page « Créer organisation »  3. L’utilisateur remplie les informations nécessaires et confirme  4. Le système crée l’organisation et envoie l’utilisateur à une page contenant un lien qu’il peut partager pour que les autres utilisateurs puissent rejoindre l’organisation | | | | | |
| Flux alternatif 1 :  Champ invalide | | | 1. Les champs remplies par l’utilisateur sont invalides  2. Le système ne crée pas l’organisation et envoie un message d’erreur indiquant qu’il reste des champs importants à remplir | | | | | |
| **Historique des versions** | | | | | | | | |
| 2017-09-01 | | V1.0 | | Rédaction initiale. | | Antoine Gamache | | |

# Annexe A – Principe de conception (SOLID DRY KISS)

## S.O.L.I.D

### S : Single Responsibility Principle (Responsabilité unique)

Notre logiciel suit le principe de la responsabilité unique. En effet, pendant son développement, l’une de nos priorités étaient d’assurer une rédaction de code propre. Pour ce faire, nous nous sommes assurés que nos fonctions n’avaient qu’une seule responsabilité. Cette méthode de rédaction est visible dans notre code *backend*.Nos *components* agissent de la même manière à cause de leur nature. En effet, les *components* représentent déjà des concepts. Ils ne peuvent donc pas influencer des concepts en dehors d’eux-mêmes.

### O : Open/Closed (Ouvert/Fermé)

Notre logiciel ne suit pas le principe de l’ouverture à l’extension et de la fermeture à la modification. En effet, nous n’avons pas codé en OOP (Programmation orienté-objet). Les concepts tels que l’héritage et l’abstraction n’ont pas été utilisés lors de notre rédaction de code. Bien que les *components* en React soient semblables à des classes en Java, il n’est pas possible de facilement implémenter ces concepts de programmation.

### L : Liskov substitution principle (Substitution de Liskov)

Notre logiciel ne suit pas le principe de la substitution de Liskov. Comme pour le principe de l’ouverture et de la fermeture, nous n’avons pas utilisé ce principe de rédaction puisque React n’utilise pas les notions de l’orienté objet. La notion de sous-type n’est pas applicable à React.

### I : Interface segregation principle (Ségrégation des interfaces)

Notre logiciel suit le principe de la ségrégation des interfaces. En effet, un utilisateur ne verra que à quoi il peut voir. Par exemple, un utilisateur faisant partie d’une organisation sans avoir les droits administrateurs ne pourra avoir accès à toutes les options concernant cette organisation, telles que l’expulsion d’un utilisateur ou la suppression de l’organisation. Ainsi, l’interface de l’utilisateur reste propre et le plus simple possible.

### D : Dependency inversion principle (Inversion des dépendances)

Notre logiciel ne suit pas le principe de l’inversion des dépendances. En effet, les modules de haut niveau dépendent parfois des modèles de bas niveau à cause TypeScript et de comment la référence de module fonctionne. Dans TypeScript, il est nécessaire d’exporter directement les *components* afin de les utiliser et de les modifier. Ainsi, le principe de l’inversion des dépendances ne peut pas être respecté.

## D.R.Y (Don’t repeat yourself)

Notre logiciel suit le principe D.R.Y grâce aux fonctionnalités de React. En effet, le développement avec React peut se faire avec ce qu’on appelle des *components*. C’est l’équivalent des classes en Java. Ces *components* sont utilisés afin de pouvoir représenter un concept visuel sur notre site web. Ainsi, nous n’avons pas à réécrire de code pour faire fonctionner deux aspects de notre site web qui semblent similaires.

Prenons l’exemple de la barre de navigation. En effet, chaque item de la barre de navigation marche de manière semblable. La seule différence est l’endroit où il vous dirige. De plus, l’item pour les notifications possède un compteur afin de pouvoir savoir rapidement combien de notification l’utilisateur possède. Chacun de ces items fonctionne avec le même *component*. Ainsi, il n’y a pas de répétition de code.

## K.I.S.S (Keep it super simple)

Notre logiciel suit partiellement le principe K.I.S.S. En effet, il est vrai que certaines parties du code pourrait être simplifiées. Il nous arrivait parfois de voir une tâche à faire et de penser à tous les potentiels problèmes qu’il pourrait y avoir (des *edge-cases*). Si nous regardons les scripts pour la base de données, nous pouvons voir des fonctions qui ne sont pas utilisés à cause d’un mauvais jugement au début de leur conception.

Toutefois, d’autres endroits dans le code sont très simples et ne possèdent aucune fonction de trop ou qui n’est pas nécessaire. Par exemple, la page de notifications possède le strict minimum pour qu’elle puisse marcher. Même s’il y aurait des choses dans le futur que l’on voudrait ajouter à cette page, elles n’ont pas été prises en compte lors de la conception initiale.

Finalement, ce concept n’était pas une priorité lors du développement de notre logiciel, mais nous l’avons toutefois respecté à certains moments.

1. https://adonis.lalib.fr/E9782370541062.pdf [↑](#footnote-ref-2)