

Table des matières

[1 Analyse préliminaire 4](#_Toc197328164)

[1.1 Introduction 4](#_Toc197328165)

[1.2 Objectifs 4](#_Toc197328166)

[1.3 Gestion de projet 5](#_Toc197328167)

[1.4 Planification initiale 5](#_Toc197328168)

[2 Analyse / Conception 5](#_Toc197328169)

[2.1 Contexte produit 5](#_Toc197328170)

[2.2 Contextes techniques 6](#_Toc197328171)

[2.2.1 Opérationnel 6](#_Toc197328172)

[2.2.2 Validation 6](#_Toc197328173)

[2.2.3 Développement 6](#_Toc197328174)

[2.3 Concept 6](#_Toc197328175)

[2.4 Analyse fonctionnelle 7](#_Toc197328176)

[2.5 Stratégie de test 7](#_Toc197328177)

[2.6 Risques techniques 7](#_Toc197328178)

[2.7 Planification 7](#_Toc197328179)

[3 Réalisation 8](#_Toc197328180)

[3.1 Points de design spécifiques 8](#_Toc197328181)

[*3.1.1* *…* 8](#_Toc197328182)

[*3.1.2* *…* 8](#_Toc197328183)

[*3.1.3* *…* 9](#_Toc197328184)

[3.2 Déroulement 9](#_Toc197328185)

[3.2.1 Sprints 9](#_Toc197328186)

[3.2.2 Stories 9](#_Toc197328187)

[3.3 Description des tests effectués 9](#_Toc197328188)

[3.4 Bilan 9](#_Toc197328189)

[3.4.1 Erreurs restantes 9](#_Toc197328190)

[3.4.2 Stories 9](#_Toc197328191)

[3.4.3 Dette technique 9](#_Toc197328192)

[3.5 Recours à l’intelligence artificielle 9](#_Toc197328193)

[3.6 Liste des documents fournis 10](#_Toc197328194)

[4 Conclusions 10](#_Toc197328195)

[5 Annexes 11](#_Toc197328196)

[5.1 Résumé du rapport du TPI / version succincte de la documentation 11](#_Toc197328197)

[5.2 Sources – Bibliographie 11](#_Toc197328198)

[5.3 Journal de travail 11](#_Toc197328199)

[~~5.4~~ ~~Manuel d'Installation~~ 11](#_Toc197328200)

[5.5 Manuel d'Utilisation 11](#_Toc197328201)

[5.6 Archives du projet 11](#_Toc197328202)

*NOTE L’INTENTION DES UTILISATEURS DE CE CANEVAS :*  
*Toutes les parties en italiques sont là pour aider à comprendre ce qu’il faut mettre dans cette partie du document. Elles n’ont donc aucune raison d’être dans le document final.*

*De plus, en fonction du type de projet, il est tout à fait possible que certains chapitres ou paragraphes n’aient aucun sens. Dans ce cas il est recommandé de les retirer du document pour éviter de l’alourdir inutilement.*

# Analyse préliminaire

## Introduction

Le but de ce projet est de créer une application qui permet aux utilisateurs de générer un journal de travail en utilisant les commit GitHub. Son nom est GitJournal

L’application se base sur la date des commits, la personne ayant effectué le commit, les métadonnées du commit. Les métadonnées comprennent le titre du commit et la description du commit

GitJournal va donc permettre de gagner beaucoup de temp car il permet de ne pas perdre de temp à remplir son journal de travail. De plus il permet d’exporter le JDT en pdf avec une belle mise en page et bien présentable.

*Ce chapitre décrit brièvement le projet, le cadre dans lequel il est réalisé, les raisons de ce choix et ce qu'il peut apporter à l'élève ou à l'école. Il n'est pas nécessaire de rentrer dans les détails (ceux-ci seront abordés plus loin) mais cela doit être aussi clair et complet que possible (idées de solutions). Ce chapitre contient également l'inventaire et la description des travaux qui auraient déjà été effectués pour ce projet.*

*Ces éléments peuvent être repris des spécifications de départ.*

## Objectifs

Objectifs de formation :

Ce projet et mon projet de TPI.

Objectifs du produit :

Ce produit doit permettre d’exporter un JDT en format PDF en se basant sur les commits dans une repository donné.

Il va également permettre à l’utilisateur de modifier les données des commits en supprimer et même en ajouter pour refléter au mieux le travail réalisé.

On peut exporter nos modifications dans un fichier et le réimporter sur un autre poste pour pouvoir transmettre notre rapport au format gitj ou travailler sur plusieurs postes.

*Ce chapitre énumère les objectifs du projet. L'atteinte ou non de ceux-ci devra pouvoir être contrôlée à la fin du projet. Les objectifs pourront éventuellement être revus après l'analyse.*

*Ces éléments peuvent être repris des spécifications de départ.*

*Séparer les objectifs en deux catégories : les objectifs de formation et les objectifs du produit.*

## Gestion de projet

Pour ce projet je vais utiliser la méthode agile, elle permet une gestion plus flexible du temp et des tâches et permet grâce au sprint review et user stories validée par le CP de ne pas s’égarer.

Il y aura au total 2 sprints durant ce projet, ils vont durer approximativement 2 semaines.

Lors de ses sprints, il y aura une sprint review avec le chef de projet pour faire un point sur la progression du projet.

Lors du début de chaque journée/ demi-journée je fais un résumé de ce que j’ai fais la fois d’avant et ce que je compte faire aujourd’hui.

***Décrivez comment vous allez gérer votre projet. Quels sont les outils, les pratiques, les personnes impliquées, les rôles,… ?***

## Planification initiale

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| No | Objectifs | | Durée | Dates | Date Sprint Review |
| 1 | Donner son Token à l'app | ~2H | ~46H30 | 05.05  Au  16.05 | Vendredi 16.05.2025  9H00 |
| Changer de Repository | ~10H30 |
| Afficher mes commits sous forme de JDT | ~12H |
| Reprendre mon JDT sur un autre poste | ~10H |
| Documentation | ~10H |
| 2 | Gérer les entrées | ~10H50 | ~36H50 | 05.05  Au  16.05 | Vendredi 23.05.2025  9H00 |
| Exporter en PDF | ~18H |
| Documentation | ~8H |

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 28.04 - 02.05 | | 05.05 - 09.05 | | 12.05 - 16.05 | | 19.05 - 23.05 | | 26.05 - 30.05 | | 02.06 - 06.06 | |
|  | Matin | Après Midi | Matin | Après Midi | Matin | Après Midi | Matin | Après Midi | Matin | Après Midi | Matin | Après Midi |
| Lundi | - | | 05:35 | | 05:35 | | 05:35 | | 03:10 | Examens | 03:10 | - |
| Mardi | - | | - | | - | | - | | - | | - | |
| Mercredi | - | | 07:15 | | 07:15 | | 07:15 | | 07:15 | | - | |
| Jeudi | - | | - | 03:10 | - | 03:10 | - | 03:10 | Férié | | - | |
| Vendredi | 07:15 | | 07:15 | | 07:15 | | 07:15 | | Férié | | - | |

Le total d’heures dans ma planification n’atteint pas les 90H prévue pour ce projet. Dans ces heures, je n’ai pas pris en compte le vendredi 2 mai 2025. C’est le jour où j’ai reçu mon CDC et j’ai pris la décision de ne pas l’inclure dans mon premier sprint. Ce vendredi a duré 7H15.

# Analyse / Conception

## Contexte produit

GitJournal est un program desktop fait pour Windows (10 et 11).

Il est fait pour être utilisé par un seul utilisateur à la fois. Comme l’application requiert un PAT pour se connecter à github, un seul token peut être introduit à la fois ce qui rend l’application mono utilisateur.

Le système exploite les métadata des commits GitHub et prend en compte les modifications de l’utilisateur enregistrées dans un fichier gitj.



***Placer le produit dans son contexte d’utilisation. Par exemple :***

* ***Site Web Internet***
* ***Application, Web, intranet***
* ***Application mobile***
* ***Infrastructure dupliquer des sites multiples***

***Présenter les utilisateurs du système ainsi que les rôles qu’ils peuvent avoir***

***Présenter les données/information que le système prend en charge***

*Au minimum :*

* *Un ou plusieurs schémas de contexte montrant le produit dans son environnement d’utilisation, ainsi que ses utilisateurs. Ce type schéma doit être accompagné d’explications textuelles*

## Contextes techniques

### Opérationnel

Pour le contexte opérationnel, GitJournal est exécuté sur n’importe quelle machine sous Windows 10 ou 11.

Avec ou sans droits administrateurs.

Accès à internet et spécialement GitHub, peu importe le lieu, si on veut récupérer des informations depuis GitHub, on va avoir besoin de connexion et que GitHub soit accessible pour afficher des informations de repository ou pour actualiser un fichier gitj. Pour ouvrir un fichier gitj, il faut juste l’application GitJournal.

GitJournal fait appel à l’API de GitHub via PAT (Personal access token).

L’application utilise un installer (exe ou msi) pour installer l’application.

***Placer le produit dans son contexte d’opérations.***

***Quelles sont les machines impliquées ?***

***Combien sont-elles ?***

***Comment communiquent-elles ?***

***Quels sont les services et/ou applications tierces utilisées (p.ex : base de données) ?***

***Où se situe votre travail ?***

***Justifier les choix***

***Ce chapitre doit impérativement contenir au moins une représentation graphique***

*Au minimum :*

* *Un ou plusieurs schémas de contexte montrant le produit dans son environnement technique. Ce type schéma doit être accompagné d’explications textuelles*

### Validation

Les tests de validation se passent sur des PC sous windows 10 (22H2), excepté ils ne sont pas le poste où se passe le développement.

Les tests sont effectués sur au moins 2 postes.

Il n’y aura pas de droits administrateurs.

Accès à internet et spécialement GitHub.

GitJournal fait appel à l’API de GitHub via PAT (Personal access token)

Pour ces tests il faut un repository de quelqu’un qui a comme moi formater son repository de façon que GitJournal soit capable d’utiliser les metadata pour créer un JDT complet. Cela importe peu à qui appartient le repository, il peu être le repo de GitJournal. Il faut juste que le PAT soit celui du propriétaire du repository.

Le code est cloné depuis de repository GitHub.

Aucun fichier en plus du code et du fichier de config de l’application (si il y en a un) ne sont présents.

Le code est exécuté directement depuis Visual Studio 2022.

Le programme sera fait en .Net 8, il faudra donc que la machine ait bien .Net 8 d’installé.

***Faire la même description que ci-dessus, mais pour l’environnement dans lequel votre produit doit être validé avant d’être déclaré publiable.***

***Expliquer comment le produit sera déployé dans le contexte de production une fois validé.***

***Justifier les choix***

### Développement

Le développement se passe sur un PC sous windows 10 (22H2).

Il n’y aura pas de droits administrateurs.

Accès à internet et spécialement GitHub.

GitJournal fait appel à l’API de GitHub via PAT (Personal access token)

Pour les données, je vais utiliser le repository pour ce projet car j’ai fait en sorte de formater mes commits pour qu’ils soient utilisable par GitTools.

Liens vers le repository : <https://github.com/Morgan-DD/GitJournal>

Pour passer de l’environnement de développement à l’environnement de validation, il n’y a pas grand-chose à faire excepté que le repository soit à jour.

Le développement se passera sur Visual Studio 2022.

Le programme sera fait en .Net 8, il faudra donc que la machine ait bien .Net 8 d’installé.

***Décrire l’environnement dans lequel le projet va être réalisé.***

***Ce chapitre doit permettre à tout nouvel arrivant dans l’équipe de projet d’installer son environnement de travail***

***Expliquer comment le produit sera déployé dans le contexte de validation une fois que vous l’estimez prêt.***

***Justifier les choix***

## Concept

*Le concept complet avec toutes ses annexes :*

*Un ou plusieurs schéma conceptuel, accompagné d’explications textuelles, de type :*

* *Modèle conceptuel des données. Expliquer chaque entité/attribut possédant une particularité.*
* *Schémas d’architecture montrant la structure interne du produit*
* *Diagramme de flux*
* *Diagramme d’état.*

*Générez les images à partir des applications utilisées (draw.io, visio, workbench, figma, …). N’ayez recours à la capture d’écran que s’il n’y a pas d’autre moyen de faire.*

*Par exemple :*

* *Multimédia : carte de site, ~~maquettes papier~~, story board préliminaire, …*
* *Bases de données : interfaces graphiques, modèle conceptuel.*
* *~~Programmation : interfaces graphiques, maquettes, analyse fonctionnelle…~~* ***Ceci est dans l’analyse fonctionnelle ci-dessous***
* *…*

## Analyse fonctionnelle

***Décrivez de manière précise en vous appuyant sur des éléments graphiques (maquettes, schémas) la manière dont vous envisagez que votre utilisateur va travailler avec votre produit.***

***Possibilité (si IceScrum est utilisé) : Reprendre le contenu des User Stories d’IceScrum : Story + tests d’acceptance (avec IceTools) + maquettes***

## Stratégie de test

***En faisant référence au contexte de validation précédemment décrit,***

*Décrire la stratégie globale de test :*

* *types de des tests et ordre dans lequel ils seront effectués.*
* *~~les moyens à mettre en œuvre.~~*
* *couverture des tests (tests exhaustifs ou non, si non, pourquoi ?).*
* *données de test à prévoir (données réelles ?)* ***et comment elles seront mises en place****.*
* *les testeurs extérieurs éventuels.*

## Risques techniques

* *risques techniques (complexité, matériel critique, disponibilté de systèmes externes, manque de compétences…).*

*Décrire aussi quelles solutions ont été appliquées pour réduire les risques (priorités, formation, actions, …).*

***Attention : le manque de temps n’est pas un risque technique***

## Planification

*Révision de la planification initiale du projet :*

* *~~planning indiquant les dates de début et de fin du projet ainsi que le découpage connu des diverses phases.~~*
* *~~partage des tâches en cas de travail à plusieurs.~~*

*~~Il s’agit en principe de la planification~~* ***~~définitive du projet~~****~~. Elle peut être ensuite affinée (découpage des tâches). Si les délais doivent être ensuite modifiés, le responsable de projet doit être avisé, et les raisons doivent être expliquées dans l’historique.~~*

***Cette section n’est présente que si la planification initiale a dû être revue suite à l’analyse***

# Réalisation

## Points de design spécifiques

***Ce chapitre est constitué de plusieurs sous-chapitre.***

***Chaque sous-chapitre explique un point de design technique particulier, quelque chose que vous avez dû inventer pour répondre au besoin et qui ne peut pas s’expliquer par de simples commentaires dans le code.***

***Il s’agit d’explications techniques sur le fonctionnement du système. Les explications sont appuyées par des diagrammes, ou de très brefs éléments de code.***

***NE PAS mettre ici des pratiques usuelles que tout professionnel de la branche connaît déjà. Par exemple, n’EXPLIQUEZ PAS ICI CE QU’EST LE PATTERN MVC.***

***Exemple (simplifié à l’extrême) : Protection contre des formulaires mal intentionnés ou modifiés***

* ***Au moment de générer le formulaire, le script php :***
  + ***Concatène les noms de tous les champs contenus dans le formulaire***
  + ***Calcule un hash SHA256 de la chaîne obtenue***
  + ***Ajoute un input nommé « CSRF » de type hidden dans le form***
* ***A la réception du POST du fromulaire***
  + ***Concatène les noms des indices de $\_POST***
  + ***Calcule un hash SHA256 de la chaîne obtenue***
  + ***Vérifie que la valeur du champ CSRF correspond***

### *…*

### *…*

### *…*

## Déroulement

### Sprints

***Pour chaque sprint :***

1. ***Résumer le déroulement du sprint, citer le résultat (objectif) de sa revue***
2. ***Donner la synthèse de la Sprint retrospective***
3. ***Donner le planning du sprint suivant***

### Stories

***Résumer comment s’est passé la réalisation de chaque story, ses difficultés, les alternatives envisagées mais rejetées, ses surprises, …***

## Description des tests effectués

*~~Pour chaque partie testée de votre projet, il faut décrire:~~*

* *~~les conditions exactes de chaque test~~*
* *~~les preuves de test (papier ou fichier)~~*
* *~~tests sans preuve: fournir au moins une description~~*

***Faire la synthèse de tous les tests d’acceptance effectués tout au long du projet***

## Bilan

### Erreurs restantes

*S'il reste encore des erreurs :*

* *Description détaillée*
* *Conséquences sur l'utilisation du produit*
* *Actions envisagées ou possibles*

### Stories

***Ce qu’on pensait faire vs ce qu’on a fait***

### Dette technique

***Reporter la*** [***dette technique***](https://www.premaccess.com/qu-est-ce-que-dette-technique-comment-la-maitriser/#:~:text=La%20dette%20technique%20survient%20quand,de%20plus%20en%20plus%20fr%C3%A9quents.) ***connue. S’appuyer sur la pratique des // TODO***

## Recours à l’intelligence artificielle

***Comment avez-vous utilisé l’IA dans votre projet.***

***Si vous ne l’avez pas utilisée, pourquoi ?***

***Ce chapitre doit contenir au minimum 200 mots***

## Liste des documents fournis

*Lister les documents fournis au client avec votre produit, en indiquant les numéros de versions*

* *le rapport de projet*
* *le manuel d'Installation (en annexe)*
* *le manuel d'Utilisation avec des exemples graphiques (en annexe)*
* *autres…*

# Conclusions

*Développez en tous cas les points suivants :*

* *Objectifs atteints / non-atteints*
* *Bilan personnel : points positifs / négatifs*
* *Difficultés particulières*
* *Suites possibles pour le projet (évolutions & améliorations)*

# Annexes

## Résumé du rapport du TPI / version succincte de la documentation

## Sources – Bibliographie

*Liste des livres utilisés (Titre, auteur, date), des sites Internet (URL) consultés, des articles (Revue, date, titre, auteur) … Et de toutes les aides externes (noms)*

## Journal de travail

|  |  |  |  |
| --- | --- | --- | --- |
| **~~Date~~** | **~~Durée~~** | **~~Activité~~** | **~~Remarques~~** |
|  |  |  |  |
|  |  |  |  |

***Référence à votre journal de travail (en PDF)***

## ~~Manuel d'Installation~~

## Manuel d'Utilisation

## Archives du projet

*Media, … dans une fourre en plastique*