INSTITUT DE TECHNOLOGIE

UNIVERSITÉ Clermont Auvergne

RAPPORT DE GESTION

BRUGIERE Patrick
JOURDAIN Antoine
RICHARD Anthony
ENAULT Léon
GOIGOUX Lucie

GROUPE 6

SOMMAIRE

INTRODUCTION

Notre projet est la création d'une application ludique et pédagogique pour apprendre l'anglais dans le cadre des études supérieures. L'application devra être totalement opérationnelle avant le 5 avril. Actuellement, nous avons rédigé tous les documents concernant la gestion du projet et avons commencé à créer le design du site et de l'application. Nous avons également commencé à développer en PHP et à réfléchir sur comment l'implémenter dans notre application. Afin de nous aider, nous utilisons des outils tels quel MS Project, Microsoft Word ou encore Excel nous permettant d'être productifs. En effet, la création et l'organisation d'un tel projet nécessite des logiciels faciles d'utilisation et rapidement compréhensibles. Avec ces outils, nous pouvons accéder aux documents n'importe où, savoir ce qui a été modifié et travailler dessus à n'importe quel moment. Dans ce rapport, nous allons détailler notre approche afin d'organiser ce projet par le biais de différents documents tels que notre WBS, notre diagramme de GANTT prévisionnel ou encore notre coût prévisionnel.

WBS

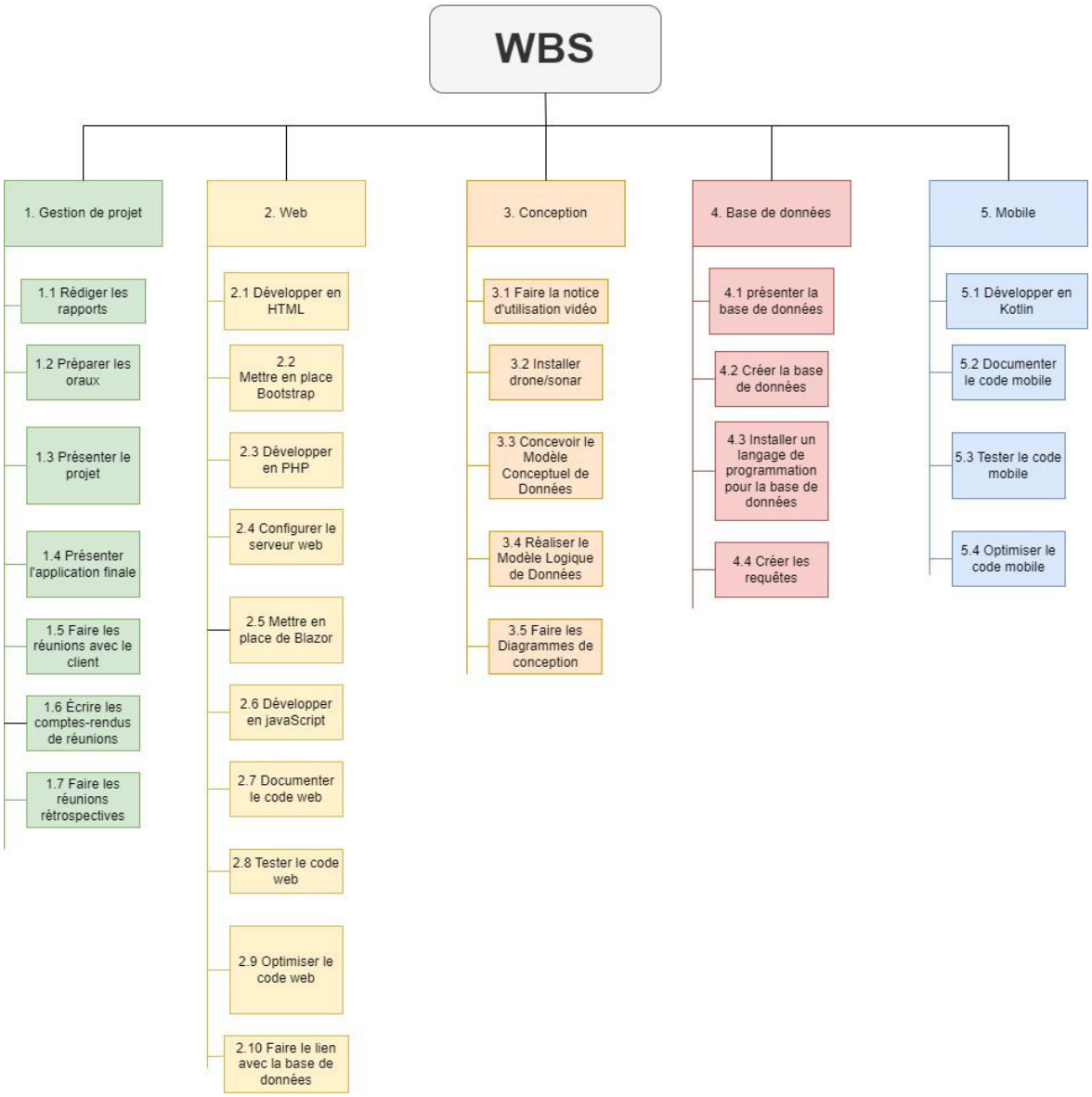
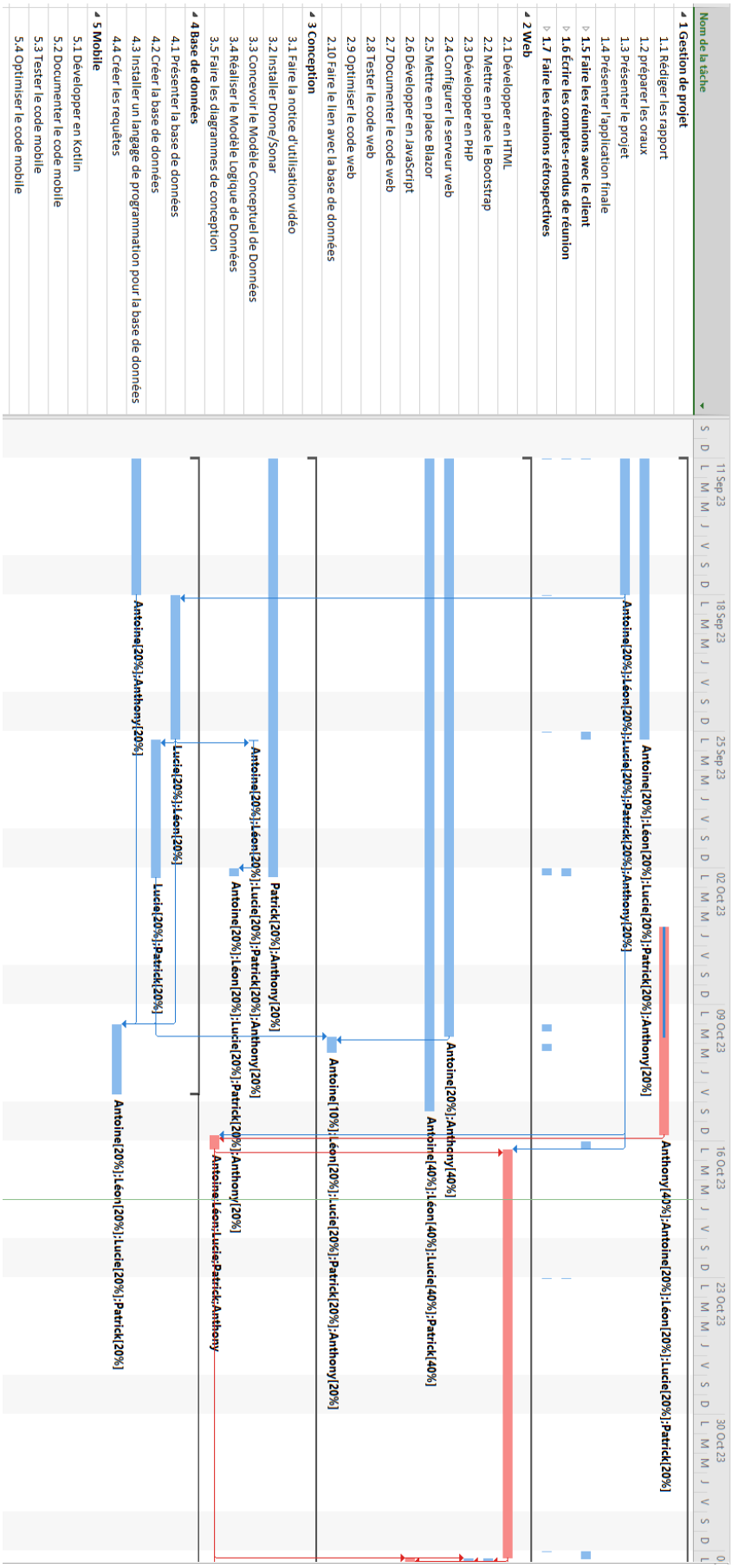
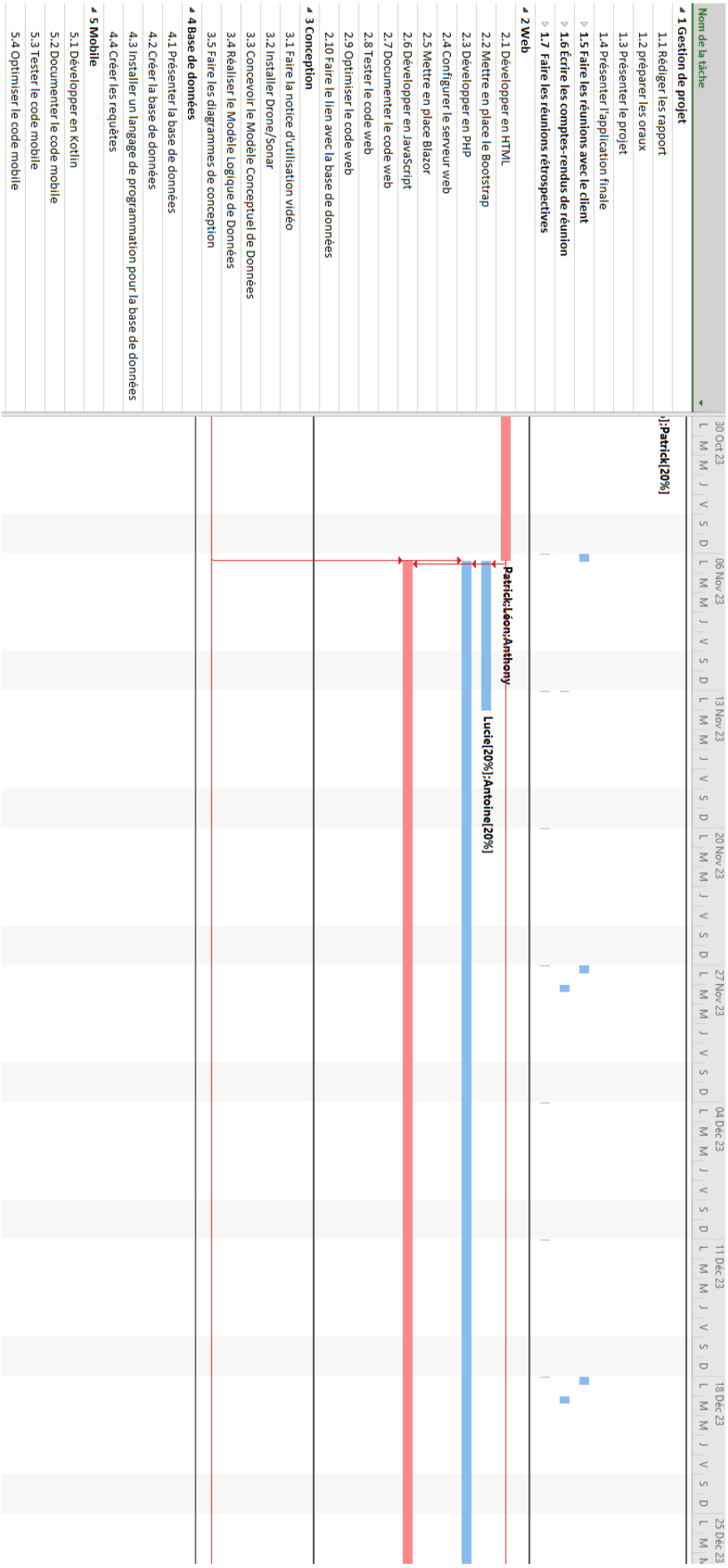
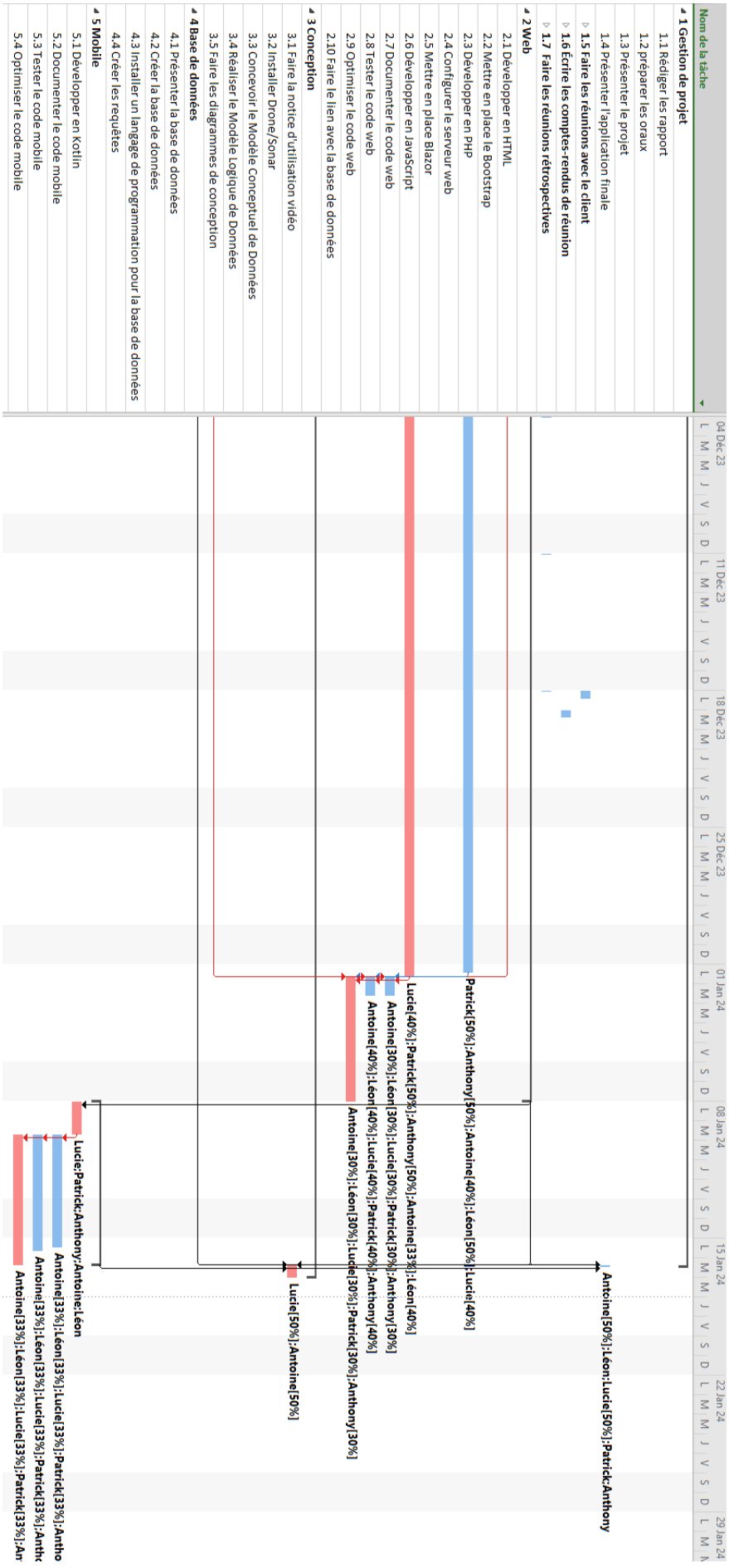


DIAGRAMME DE GANTT




Nom de la tâche
1 Gestion de projet
1.1 Rédiger les rapport
1.2 préparer les oraux
1.3 Présenter le projet
1.4 Présenter l'application finale
1.5 Faire les réunions avec le client
1.6 Écrire les comptes-rendus de réunion
1.7 Faire les réunions rétrospectives
2 Web
2.1 Développer en HTML
2.2 Mettre en place le Bootstrap
2.3 Développer en PHP
2.4 Configurer le serveur web
2.5 Mettre en place Blazor
2.6 Développer en JavaScript
2.7 Documenter le code web
2.8 Tester le code web
2.9 Optimiser le code web
2.10 Faire le lien avec la base de données
3 Conception
3.1 Faire la notice d'utilisation vidéo
3.2 Installer Drone/Sonar
3.3 Concevoir le Modèle Conceptuel de Données
3.4 Réaliser le Modèle Logique de Données
3.5 Faire les diagrammes de conception
4 Base de données
4.1 Présenter la base de données
4.2 Créer la base de données
4.3 Installer un langage de programmation pour la base de données
4.4 Créer les requêtes
5 Mobile
5.1 Développer en Kotlin
5.2 Documenter le code mobile
5.3 Tester le code mobile
5.4 Optimiser le code mobile
30 Oct 23
06 Nov 23
13 Nov 23
20 Nov 23
27 Nov 23
04 Déc 23
11 Déc 23
18 Déc 23
25 Déc 23
Patrick;Léon;Anthony
Lucie[20%];Antoine[20%]


Nom de la tâche
1 Gestion de projet
1.1 Rédiger les rapport
1.2 préparer les oraux
1.3 Présenter le projet
1.4 Présenter l'application finale
1.5 Faire les réunions avec le client
1.6 Écrire les comptes-rendus de réunion
1.7 Faire les réunions rétrospectives
2 Web
2.1 Développer en HTML
2.2 Mettre en place le Bootstrap
2.3 Développer en PHP
2.4 Configurer le serveur web
2.5 Mettre en place Blazor
2.6 Développer en JavaScript
2.7 Documenter le code web
2.8 Tester le code web
2.9 Optimiser le code web
2.10 Faire le lien avec la base de données
3 Conception
3.1 Faire la notice d'utilisation vidéo
3.2 Installer Drone/Sonar
3.3 Concevoir le Modèle Conceptuel de Données
3.4 Réaliser le Modèle Logique de Données
3.5 Faire les diagrammes de conception
4 Base de données
4.1 Présenter la base de données
4.2 Créer la base de données
4.3 Installer un langage de programmation pour la base de données
4.4 Créer les requêtes
5 Mobile
5.1 Développer en Kotlin
5.2 Documenter le code mobile
5.3 Tester le code mobile
5.4 Optimiser le code mobile
04 Déc 23
11 Déc 23
18 Déc 23
25 Déc 23
01 Jan 24
08 Jan 24
15 Jan 24
22 Jan 24
29 Jan 24
Patrick[50%];Anthony[50%];Antoine[40%];Léon[50%];Lucie[40%]
Lucie[40%];Patrick[50%];Anthony[50%];Antoine[33%];Léon[40%]
Antoine[30%];Léon[30%];Lucie[30%];Patrick[30%];Anthony[30%]
Antoine[40%];Léon[40%];Lucie[40%];Patrick[40%];Anthony[40%]
Antoine[50%];Léon;Lucie[50%];Patrick;Anthony
Antoine[30%];Léon[30%];Lucie[30%];Patrick[30%];Anthony[30%]
Lucie[50%];Antoine[50%]
Lucie;Patrick;Anthony;Antoine;Léon
Antoine[33%];Léon[33%];Lucie[33%];Patrick[33%];Antho
Antoine[33%];Léon[33%];Lucie[33%];Patrick[33%];Anthc
Antoine[33%];Léon[33%];Lucie[33%];Patrick[33%];An

PERT-TEMPS

Nom de la tâche	Durée	Début au plus tôt	Fin au plus tôt	Début au plus tard	Fin au plus tard	Marge totale	Noms ressources	Prédécesseurs
1 Gestion de projet	**184,8 heures**	**Lun 11/09/23**	**Mar 16/01/24**	**Jeu 05/10/23**	**Mer 17/01/24**	**4,2 heures**		
1.1 Rédiger les rapport	40,9 heures	Jeu 05/10/23	Dim 15/10/23	Jeu 05/10/23	Dim 15/10/23	0 heure	AR;AJ;LE;LG;PB	
1.2 préparer les oraux	8,5 heures	Lun 11/09/23	Lun 25/09/23	Lun 15/01/24	Mer 17/01/24	180,5 heures	AJ;LE;LG;PB;AR	
1.3 Présenter le projet	3 heures	Lun 11/09/23	Lun 18/09/23	Dim 15/10/23	Dim 15/10/23	50,9 heures	AJ;LE;LG;PB;AR	
1.4 Présenter l'application finale	2,8 heures	Mar 16/01/24	Mar 16/01/24	Mar 16/01/24	Mer 17/01/24	4,2 heures	AJ;LE;LG;PB;AR	2 ; 4 ; 5
1.5 Faire les réunions avec le client	**127 heures**	**Lun 11/09/23**	**Lun 18/12/23**	**Mar 16/01/24**	**Mer 17/01/24**	**62 heures**	AJ;LE;LG;PB;AR	
1.6 Écrire les comptes-rendus de réunion	**134,5 heures**	**Lun 11/09/23**	**Mar 19/12/23**	**Mar 16/01/24**	**Mer 17/01/24**	**54,5 heures**	AJ;LE;LG;PB;AR	
1.7 Faire les réunions rétrospectives	**125,5 heures**	**Lun 11/09/23**	**Lun 18/12/23**	**Mar 16/01/24**	**Mer 17/01/24**	**63,5 heures**	AJ;LE;LG;PB;AR	
2 Web	**155,88 heures**	**Lun 11/09/23**	**Lun 08/01/24**	**Lun 16/10/23**	**Lun 08/01/24**	**0 heure**		
2.1 Développer en HTML	6,67 heures	Lun 16/10/23	Lun 06/11/23	Lun 16/10/23	Lun 06/11/23	0 heure	PB;LE;AR	1.3 ; 3.5
2.2 Mettre en place le Bootstrap	12,5 heures	Lun 06/11/23	Mar 14/11/23	Lun 01/01/24	Lun 08/01/24	78,81 heures	LG;AJ	2.1
2.3 Développer en PHP	75 heures	Lun 06/11/23	Lun 01/01/24	Lun 06/11/23	Lun 01/01/24	2,98 heures	PB;AR;AJ;LE;LG	2.1 ; 3.5
2.4 Configurer le serveur web	17,5 heures	Lun 11/09/23	Mar 10/10/23	Mar 19/12/23	Mar 02/01/24	133,88 heures	AJ;AR	
2.5 Mettre en place Blazor	43,75 heures	Lun 11/09/23	Sam 14/10/23	Mar 05/12/23	Lun 08/01/24	112,13 heures	AJ;LE;LG;PB	
2.6 Développer en JavaScript	77,98 heures	Lun 06/11/23	Lun 01/01/24	Lun 06/11/23	Lun 01/01/24	0 heure	LG;PB;AR;AJ;LE	2.1 ; 3.5
2.7 Documenter le code web	6,67 heures	Lun 01/01/24	Mar 02/01/24	Mar 02/01/24	Lun 08/01/24	6,67 heures	AJ;LE;LG;PB;AR	2.1 ; 2.3 ; 2.6 ; 3.5
2.8 Tester le code web	7,5 heures	Lun 01/01/24	Mar 02/01/24	Mar 02/01/24	Lun 08/01/24	5,83 heures	AJ;LE;LG;PB;AR	2.1 ; 2.3 ; 2.6 ; 3.5
2.9 Optimiser le code web	13,33 heures	Lun 01/01/24	Lun 08/01/24	Lun 01/01/24	Lun 08/01/24	0 heure	AJ;LE;LG;PB;AR	2.1 ; 2.3 ; 2.6 ; 3.5
2.10 Faire le lien avec la base de données	4,5 heures	Mar 10/10/23	Mer 11/10/23	Mar 02/01/24	Lun 08/01/24	133,88 heures	AJ;LE;LG;PB;AR	2.4 ; 4.2
3 Conception	**189 heures**	**Lun 11/09/23**	**Mer 17/01/24**	**Dim 15/10/23**	**Mer 17/01/24**	**0 heure**		
3.1 Faire la notice d'utilisation vidéo	7 heures	Mar 16/01/24	Mer 17/01/24	Mar 16/01/24	Mer 17/01/24	0 heure	LG;AJ	2 ; 4 ; 5
3.2 Installer Drone/Sonar	12,5 heures	Lun 11/09/23	Lun 02/10/23	Lun 15/01/24	Mer 17/01/24	176,5 heures	PB;AR	
3.3 Concevoir le Modèle Conceptuel de Données	2 heures	Lun 25/09/23	Lun 25/09/23	Mar 16/01/24	Mar 16/01/24	178 heures	AJ;LE;LG;PB;AR	4.1
3.4 Réaliser le Modèle Logique de Données	1 heure	Lun 02/10/23	Lun 02/10/23	Mar 16/01/24	Mer 17/01/24	178 heures	AJ;LE;LG;PB;AR	3.3
3.5 Faire les diagrammes de conception	4 heures	Dim 15/10/23	Lun 16/10/23	Dim 15/10/23	Lun 16/10/23	0 heure	AJ;LE;LG;PB;AR	1.1 ; 1.3

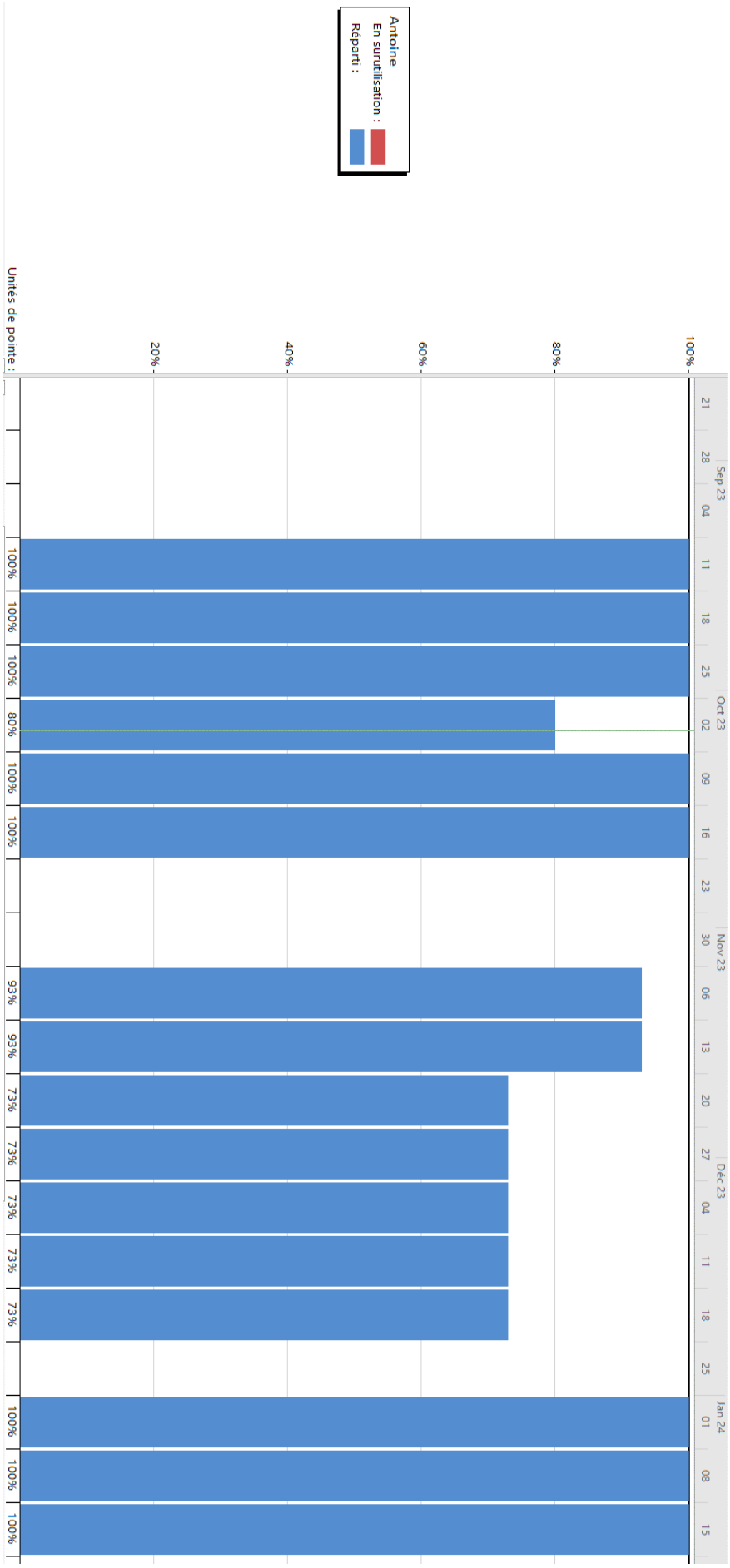
Nom de la tâche	Durée	Début au plus tôt	Fin au plus tôt	Début au plus tard	Fin au plus tard	Marge totale	Noms ressources	Prédécesseurs
4 Base de données	**38 heures**	**Lun 11/09/23**	**Ven 13/10/23**	**Lun 01/01/24**	**Mar 16/01/24**	**141,38 heures**		
4.1 Présenter la base de données	5 heures	Lun 18/09/23	Lun 25/09/23	Lun 01/01/24	Mar 02/01/24	138,38 heures	LG;LE	1.3
4.2 Créer la base de données	5 heures	Lun 25/09/23	Lun 02/10/23	Mar 02/01/24	Mar 02/01/24	138,38 heures	LG;PB	4.1
4.3 Installer un langage de programmation pour la base de données	2,5 heures	Lun 11/09/23	Lun 18/09/23	Mer 03/01/24	Lun 08/01/24	154,5 heures	AJ;AR	
4.4 Créer les requêtes	25 heures	Mar 10/10/23	Ven 13/10/23	Lun 08/01/24	Mar 16/01/24	144 heures	AJ;LE;LG;PB	4.1 ; 4.2 ; 4.3
5 Mobile	**26,12 heures**	**Lun 08/01/24**	**Mar 16/01/24**	**Lun 08/01/24**	**Mar 16/01/24**	**0 heure**		
5.1 Développer en Kotlin	14 heures	Lun 08/01/24	Mar 09/01/24	Lun 08/01/24	Mar 09/01/24	0 heure	LG;PB;AR;AJ;LE	2
5.2 Documenter le code mobile	6,06 heures	Mar 09/01/24	Lun 15/01/24	Lun 15/01/24	Mar 16/01/24	6,06 heures	AJ;LE;LG;PB;AR	5.1
5.3 Tester le code mobile	9,09 heures	Mar 09/01/24	Lun 15/01/24	Mer 10/01/24	Mar 16/01/24	3,03 heures	AJ;LE;LG;PB;AR	5.1
5.4 Optimiser le code mobile	12,12 heures	Mar 09/01/24	Mar 16/01/24	Mar 09/01/24	Mar 16/01/24	0 heure	AJ;LE;LG;PB;AR	5.1

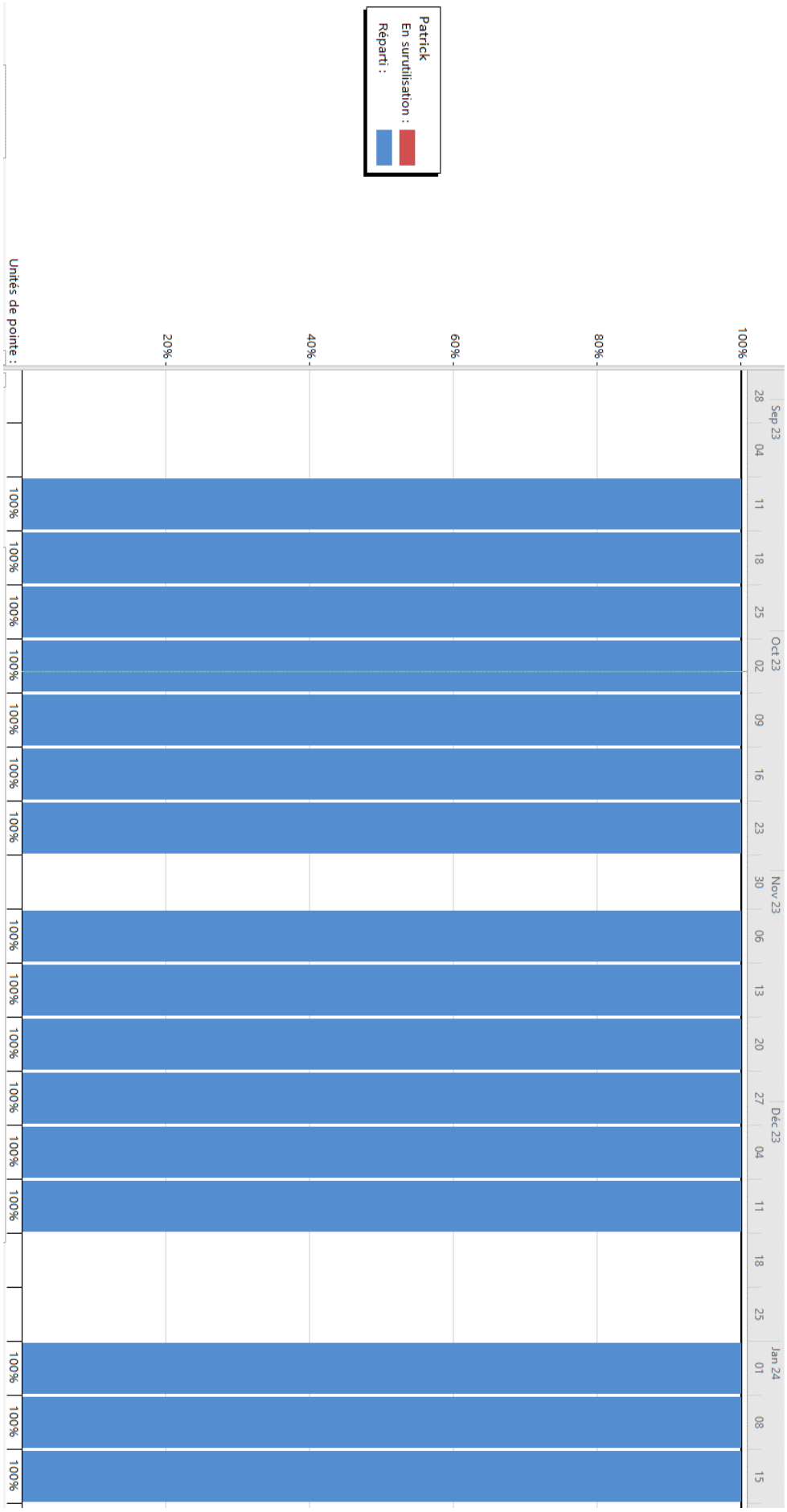
Détails des ressources :

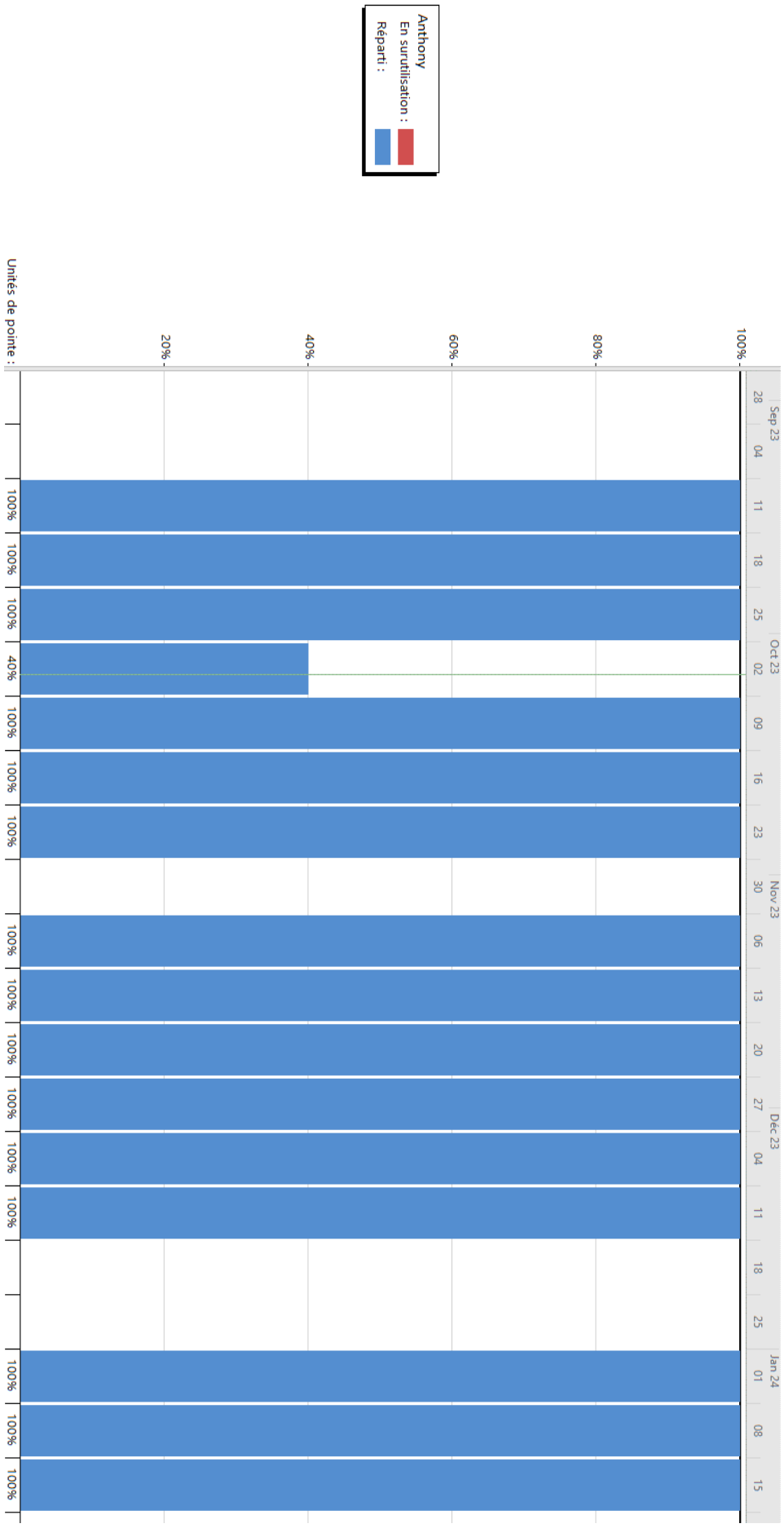
- AJ : Antoine Jourdain, Développeur
- LG : Lucie Goigoux, Développeuse
- PB : Patrick Brugière, Développeur
- AR : Anthony Richard, Développeur
- LE : Léon Enault, Développeur

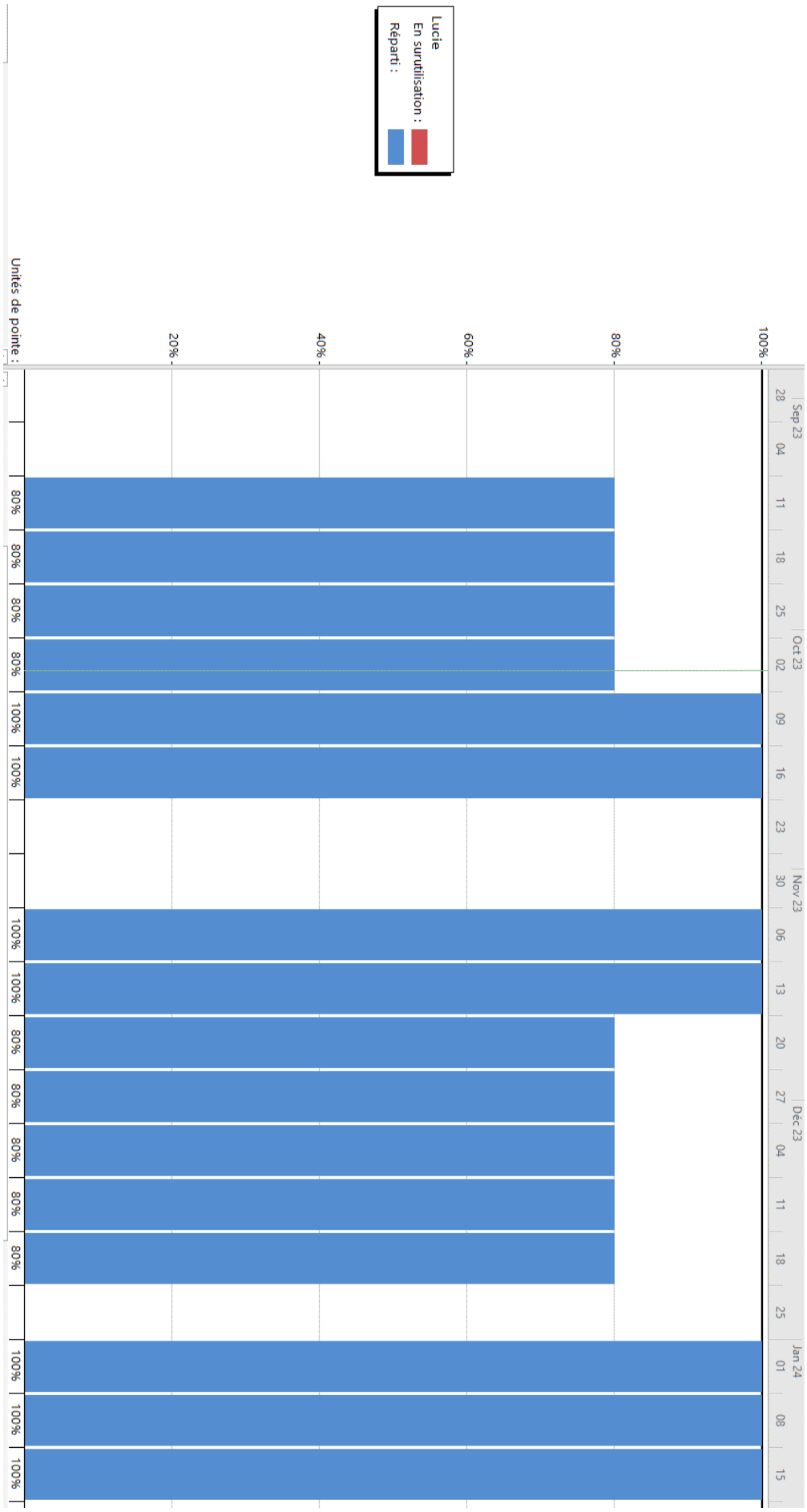
PERT-CHARGE

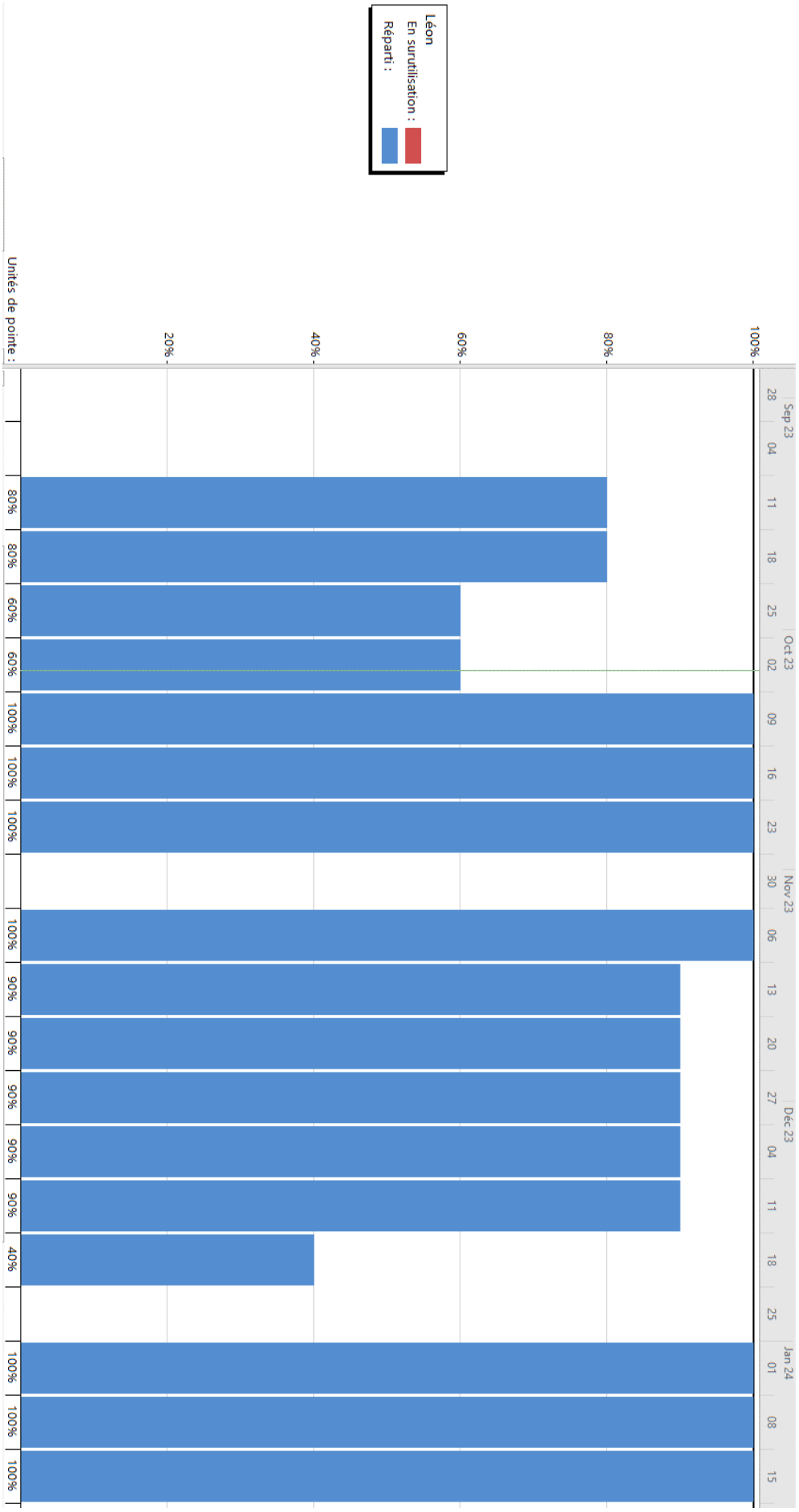
Le PERT-Charge a pour but de montrer comment nous, développeurs, sommes utilisés en tant que ressources au fil des semaines et de nos heures attribuées de SAE.
Afin d'établir ces modélisations visuelles, nous avons dû nous répartir les tâches. Ces dernières ont été attribuées en fonction des compétences et de la charge de travail déjà présente sur chacun des emplois du temps.
Cependant, nous avons rencontré un problème : nous étions surchargés lors de certaines semaines en raison de notre engagement sur plusieurs tâches à la fois.
Grâce à MS Project et son système de nivelage automatique, nous avons pu lisser l'utilisation des ressources et ainsi permettre aux développeurs de ne pas être surutilisés tout au long du projet. Ci-dessous, le détail de l'utilisation des développeurs au fil des semaines.


Antoine
En surutilisation :
Réparti :
Unités de pointe :
20%
40%
60%
80%
100%
Sep 23
Oct 23
Nov 23
Déc 23
Jan 24
100%
100%
100%
80%
100%
100%
93%
93%
73%
73%
73%
73%
73%
100%
100%
100%

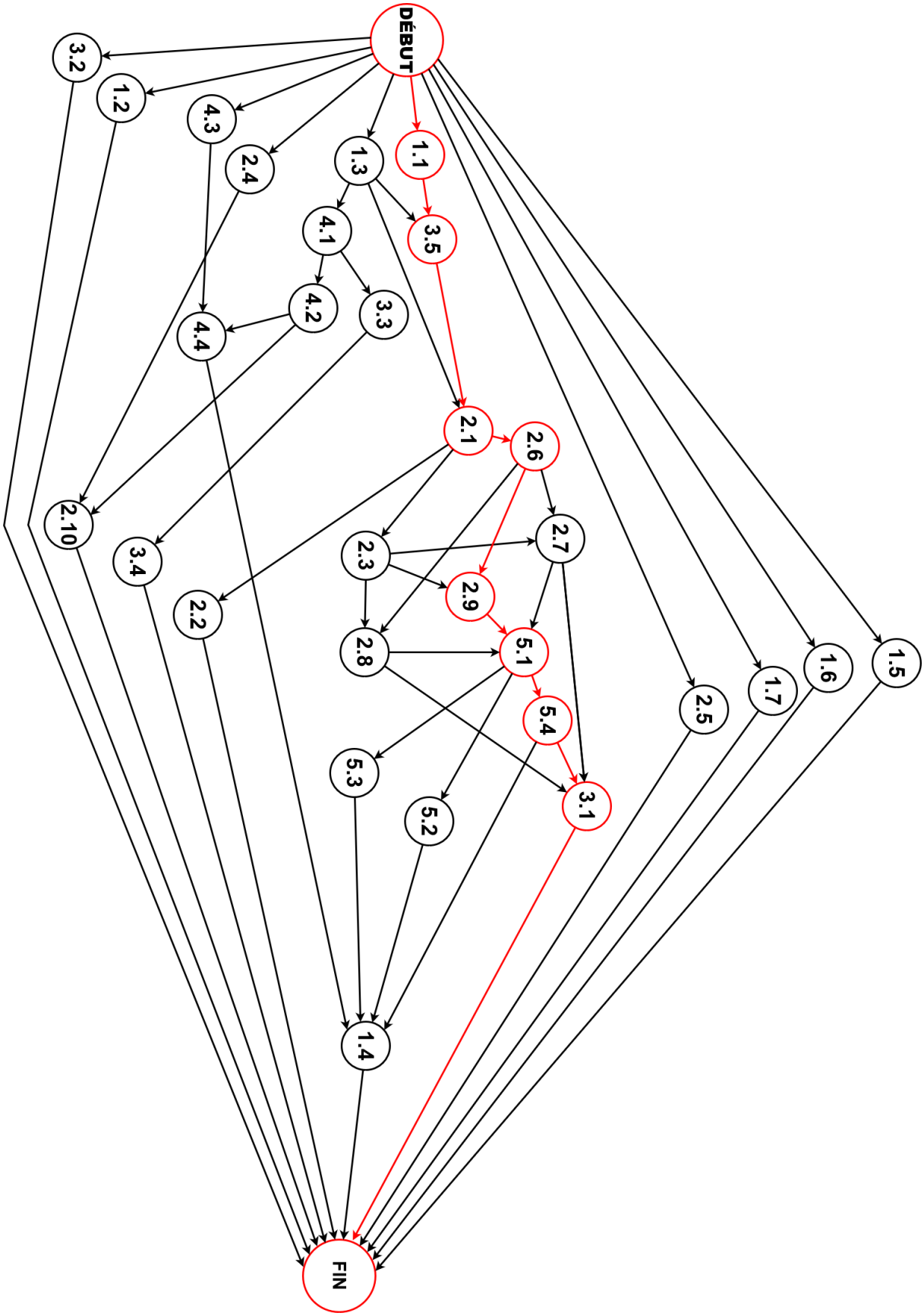

Patrick
En surutilisation :
Réparti :
Unités de pointe :
20%
40%
60%
80%
100%
Sep 23
Oct 23
Nov 23
Déc 23
Jan 24
100%


Anthony
En surutilisation :
Réparti :
Unités de pointe :
20%
40%
60%
80%
100%
Sep 23
Oct 23
Nov 23
Dec 23
Jan 24
100%
40%


Lucie
En surutilisation :
Réparti :
Unités de pointe :
20%
40%
60%
80%
100%
Sep 23
Oct 23
Nov 23
Déc 23
Jan 24
80%
100%


Léon
En surutilisation :
Réparti :
Unités de pointe :
20%
40%
60%
80%
100%
Sep 23
Oct 23
Nov 23
Déc 23
Jan 24
80%
80%
60%
60%
100%
100%
100%
100%
90%
90%
90%
90%
90%
40%
100%
100%
100%

RESEAU PERT


DÉBUT
3.2
1.2
4.3
2.4
1.3
1.1
4.1
3.5
4.2
3.3
4.4
2.1
2.6
2.10
3.4
2.3
2.7
2.2
2.9
2.8
5.1
1.6
1.5
1.7
2.5
5.4
5.3
3.1
5.2
1.4
FIN

ESTIMATION DES COUTS PREVISIONNELS

Notre évaluation des coûts prévisionnels par développeur s'aligne avec les différentes solutions disponibles sur le marché de l'informatique. En effet, nous avons évalué notre prix de l'heure à un taux standard de 55€. Nous nous sommes basés sur le salaire brut médian pour un développeur qui est de 19,23€ par heure (source : talent.fr). Ainsi, nous établissons par rapport à nos compétences, un salaire brut à 13€ de l'heure. Nous avons aussi des charges patronales qui s'élèvent à 45,9% (en 2020) du salaire brut à l'heure soit 5,97€ (source : insee.fr).
Par la suite, nous ajoutons différentes charges liées à la création et l'existence de l'application :

- Hébergement : le serveur web, la base de données mais aussi l'application doivent être hébergés sur un serveur qui a un coût de protection et de maintenance. En divisant par 5 le coût initial d'un hébergement classique pour ce type d'application, qui s'élevait à 1€, nous obtenons des frais de 0,20€ par développeur.

- Maintenance du logiciel : les services que nous utilisons pour le développement de l'application sont amenés à être mis à jour sur le long terme. Il est donc nécessaire, pour la pérennité du projet, de maintenir à jour, sur une durée minimum de 10 ans, les services et fonctionnalités que nous proposons. Dans le cadre d'un amortissement sur 10 ans, nous avons défini un prix de 20€ ce qui permet de faire vivre l'application sur cette durée.

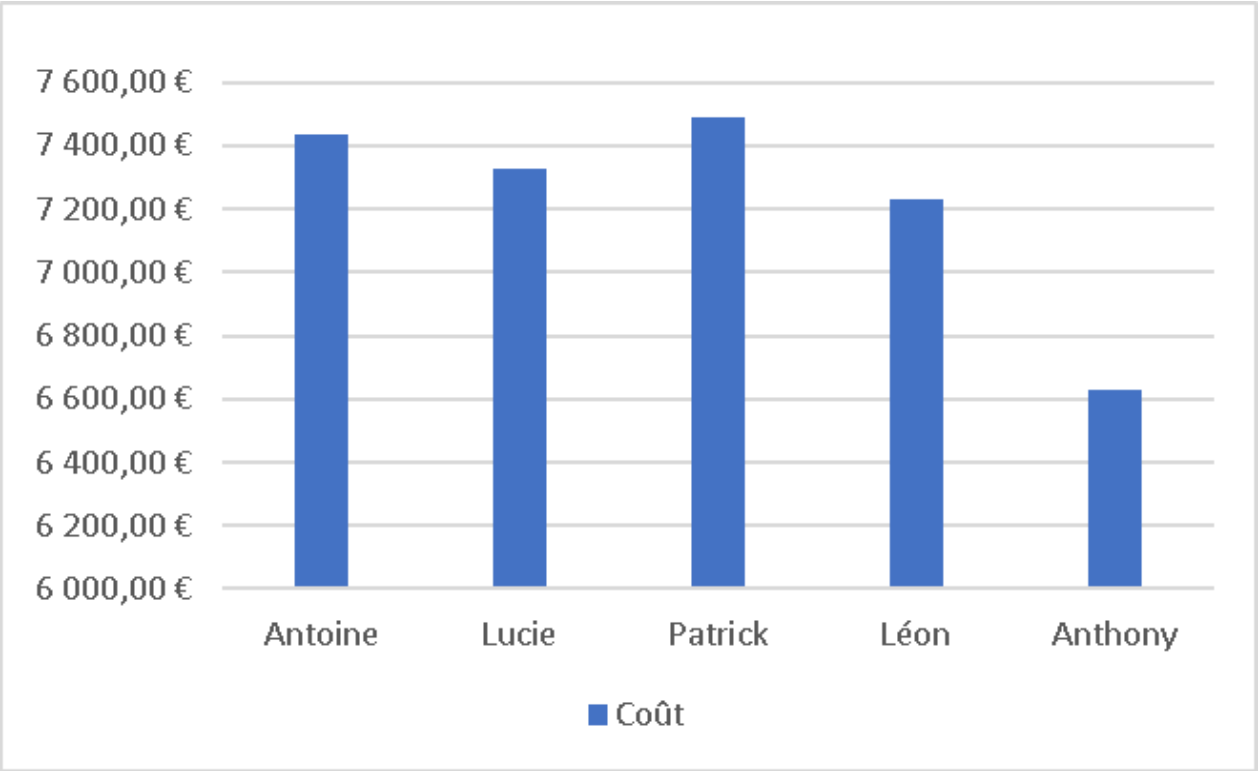
- Énergies : les coûts liés aux énergies, comme l'électricité, augmentent de plus en plus dans le contexte actuel. Cependant, nous arrivons à garder un coût à 0,10€ par heure grâce aux matériels récents que nous possédons.

- Locaux : notre prix s'élève à 2€ par heure. Nous y incluons le nettoyage mais aussi toutes les charges qui peuvent être causées par l'utilisation des bâtiments comme la vétusté de la structure, l'assurance, les impôts ou encore les frais tels que la connexion internet, les lignes téléphoniques et autres équipements.

- Usure du matériel : nous utilisons quotidiennement et de manière intensive nos machines, ces dernières subissent une usure naturelle et perdent en performance. Il est donc nécessaire de les remplacer pour éviter les pannes ou l'obsolescence. Par conséquent, nous estimons l'usure de notre matériel à 13,73€ par heure.

En additionnant l'ensemble des coûts, y compris les charges et le salaire brut, nous parvenons à un coût de 55€ par heure pour chaque développeur. Cependant, il est impératif de tenir compte que ce projet impliquera un investissement global de presque 657 heures de travail. Par conséquent, la somme totale pour le développement de cette application s'élève à 36 115,70€.

Ci-dessous, il s'agit d'un diagramme représentant les coûts de chaque développeur de l'équipe. Nous sommes pratiquement tous égaux sauf Anthony qui travaillerait 10 heures de moins sur ce projet car ce dernier est actuellement en procédure pour un échange international et a besoin de temps pour remplir ses documents administratifs.



INDICATEURS DE SUIVI DE PROJET ET DE QUALITE

Tout d'abord, il est important de définir ce qu'est le suivi de projet ainsi que la qualité pour notre équipe. Cela nous permet d'être plus efficaces lorsque nous devons analyser nos indicateurs et par conséquent, nous améliorons notre productivité.
Le suivi de projet est la manière dont nous organisons le projet par rapport aux différentes contraintes imposées par le product owner (PO) et les ressources disponibles telles que le temps ou le budget. Nous considérons que le niveau de qualité concerne la partie technique de notre projet, en l'occurrence le code et le bon fonctionnement de l'application.

Nous possédons différents indicateurs que nous avons détaillés ci-dessous :

<u>Pour le suivi de projet :</u>

- Indicateur de Temps : nous analysons l'écart entre le GANTT prévisionnel et le GANTT réel pour voir si nous sommes en avance, en retard ou dans les temps. Nous pouvons aussi voir sur le PERT-Temps les marges dont nous disposons pour allonger certaines tâches lorsque nous rencontrons des difficultés.

- Indicateur de Finance : nous analysons l'écart entre le budget déjà consommé et le budget initialement prévu pour chaque tâche. Nous optimisons de ce fait les ressources restantes par rapport au reste du projet.

- Indicateur de Respect du besoin : nous vérifions régulièrement si le travail de l'équipe correspond aux consignes du PO grâce aux tests de l'équipe lors de nos réunions. À chaque « sprint review » avec le PO, ce dernier teste avec nous ce qui a été produit afin de confirmer que le rendu corresponde à ses attentes.

- Indicateur de Performance d'équipe : nous organisons à chaque fin de sprint une réunion rétrospective pour relever les points positifs et négatifs. Ainsi, nous permettons à l'ensemble de l'équipe de se remettre en question et ainsi, nous rectifions notre organisation pour le prochain sprint. De plus, nous calculons la vélocité de l'équipe, c'est-à-dire le nombre de story points (niveau de difficulté d'une tâche) qu'elle est capable de faire par sprint.

Pour la qualité :

- Indicateur de Qualité du code : nous utilisons Drone pour effectuer des tests unitaires et lancer l'analyse du code par SonarQube. Par conséquent, nous obtenons différents résultats :
    - Un taux de couverture du code que nous souhaitons avoir à 80% ;
    - Un indice de sécurité du code que nous voulons impérativement au niveau le plus élevé (A) ;
    - Des retours de Code Smells : nous voulons qu'il n'y ait pas d'erreurs de conventions de nommage, de duplication ou encore d'optimisation.

- Retour des bugs sur l'application : il est important d'avoir une application sans bug. Le nombre de retours effectués et leur gravité sont des indicateurs de l'efficacité de notre code.
  Ces différents retours peuvent être faits par :
    - le PO dans le cadre des démonstrations et de l'utilisation de l'application ;
    - l'équipe dans le cadre des vérifications ;
    - Sonar.

OUTILS DE COMMUNICATION PRIVILEGIES

Au sein de l'équipe, nous communiquons principalement par Discord pour prévoir des réunions ou pour se répartir les tâches lorsqu'il faut que nous travaillons en dehors des créneaux de SAE. Cette application nous parait simple d'utilisation pour les discussions sur le projet.
Nous utilisons les mails pour la transmission de documents. Le recours à WordOnline nous aide à visualiser les progressions de chaque rapport et met ainsi à disposition de tous la dernière version des documents, modifiables simultanément par chaque membre du groupe.
Avec notre PO et notre tutrice de projet, nous privilégions les mails pour effectuer des entretiens toutes les trois semaines, de manière séparée, afin de faire le bilan des sprints.

Organisation du travail sur les sprints

Nos sprints ont pour but de développer les fonctionnalités du backlog validé par notre PO. Les intervalles entre chaque rendez-vous représentent le temps des sprints. Nous avons décidé, en accord avec notre PO, que les sprints de trois semaines étaient optimaux au développement d'une telle application. Cette durée offre la flexibilité nécessaire au développement de fonctionnalités complexes et à la correction des problèmes qui peuvent être liés à leur implémentation. De plus, elle permet d'ajuster notre travail en fonction des changements de besoins du PO, ce qui favorise l'adaptation en cours de projet.

Après chaque rendez-vous, nous nous réunissons et nous décidons des fonctionnalités qui devront être développées avant la prochaine rencontre. À ce moment-là, nous formons des duos/trios pour chaque fonctionnalité à coder. Évidemment, si une équipe est en difficulté, nous faisons appel à l'intelligence collective et chacun peut aider les autres.

Dès qu'une user story du backlog est terminée, nous la faisons valider par les autres membres du groupe : ils peuvent donner leur avis sur la qualité du code et ce qu'ils voudraient améliorer. Nous voulons que chaque fonctionnalité soit collective et que tout le monde ait son mot à dire sur chaque ajout de l'application. Nous voulons que chaque décision soit prise collectivement, autant que possible.

Pour que chacun soit au courant de l'avancée du projet, nous nous servons d'un KANBAN, où un ticket est une user story. Les colonnes sont :

- Non catégorisé : ce sont les tâches qui ne sont pas à faire dans notre sprint actuel.

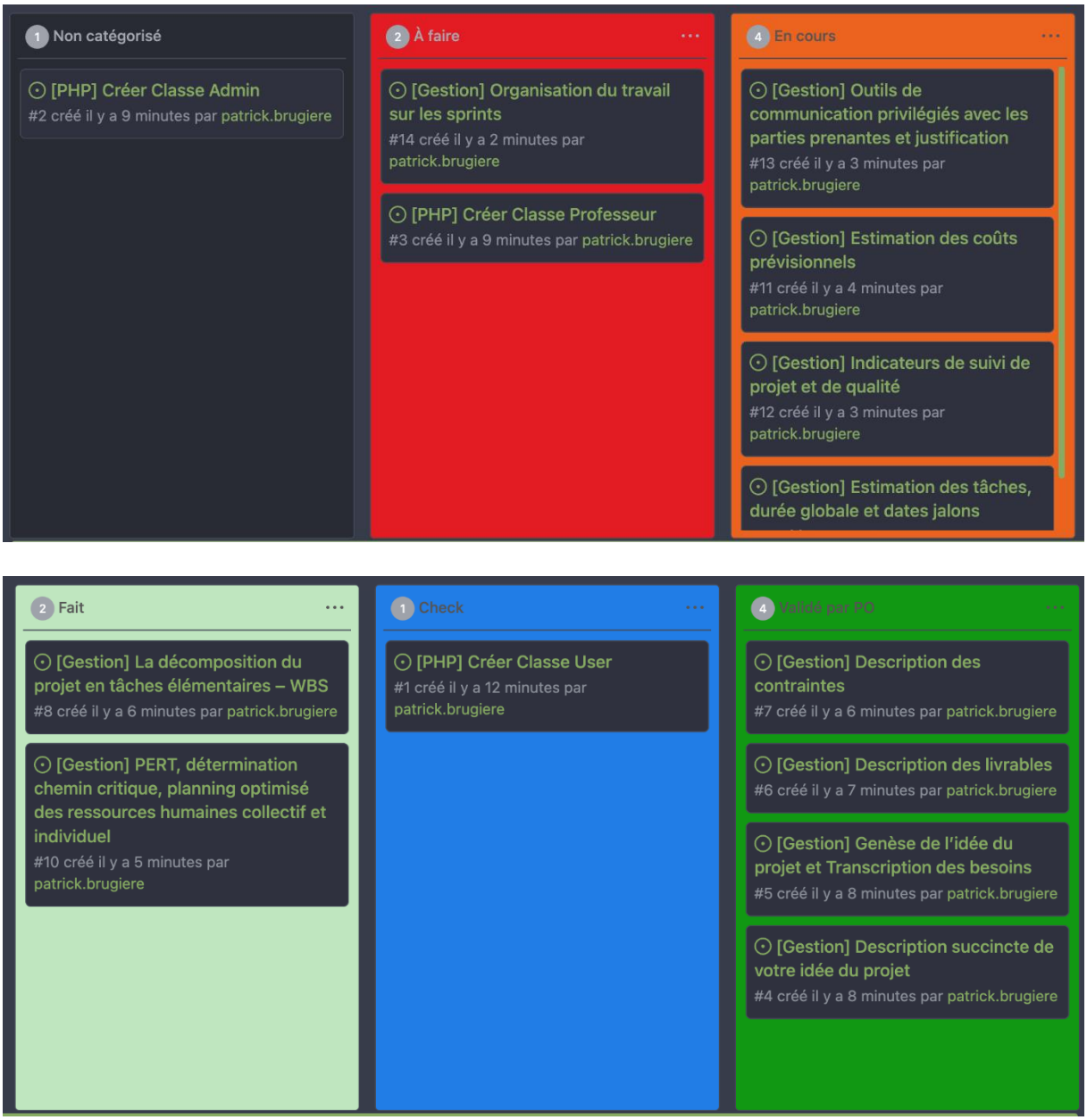
- À faire : ce sont les tâches à faire durant notre sprint, les tickets sont tous créés au début afin de faciliter la gestion de ce dernier, grâce à l'option de jalon disponible sur code first.

- En cours : ce sont les tâches en cours de développement, il est possible d'y assigner une ou plusieurs étiquettes pour savoir dans quel langage est développée cette fonctionnalité. Ces dernières nous aident aussi à connaître l'état d'avancement des tickets (bug, demande d'aide, question pour le PO).

- Fait : lorsqu'une fonctionnalité est terminée et testée, elle est déplacée dans cette catégorie en attendant d'être vérifiée et validée par toute l'équipe.

- Vérifié par l'équipe : ce sont les tâches qui sont totalement terminées et qui ont été validées par l'ensemble de l'équipe. Le code a été relu, compris et testé, ce qui permet au groupe d'avancer plus rapidement, de comprendre et développer les futures fonctionnalités plus facilement.

- Validé par le PO : il s'agit des tâches qui ne vont plus être modifiées, hors optimisation vers la fin du projet, car ces dernières ont été approuvées par le PO.



Ci-dessus, il s'agit d'un exemple de notre Kanban de la semaine 41. Il n'est pas forcément vrai pour toutes les tâches mais illustre l'organisation de notre équipe.

De plus, chaque ticket aura une étiquette de couleur en fonction de sa priorité, classée en fonction de la méthode MOSCOW, soit :

- Must (vert)
- Should (bleu)
- Could (orange)
- Won't (rouge)

À chaque fin de sprint, nous nous réunissons pour préparer la rencontre sur l'avancement du projet avec le PO. Nous en profitons pour réaliser une rétrospective de notre travail en équipe, dans le but d'optimiser nos résultats.

CONCLUSION

Grâce aux différents indicateurs et outils de gestion de projet, nous essayons de respecter notre GANTT prévisionnel tout en améliorant continuellement les performances de l'équipe afin de répondre au mieux aux attentes budgétaires, temporelles et fonctionnelles du client.

N.B : Notre rapport est disponible sur codeFirst avec ce lien pour plus de lisibilité :
https://codefirst.iut.uca.fr/git/antoine.jourdain/SAE_2A_Anglais/src/branch/master/Documentation