SAE 1.01
DÉVELOPPEMENT D'UNE APPLICATION
DOSSIER DE RENDU DE LIVRABLE
7 OCTOBRE 2022

CONS'ÉCO

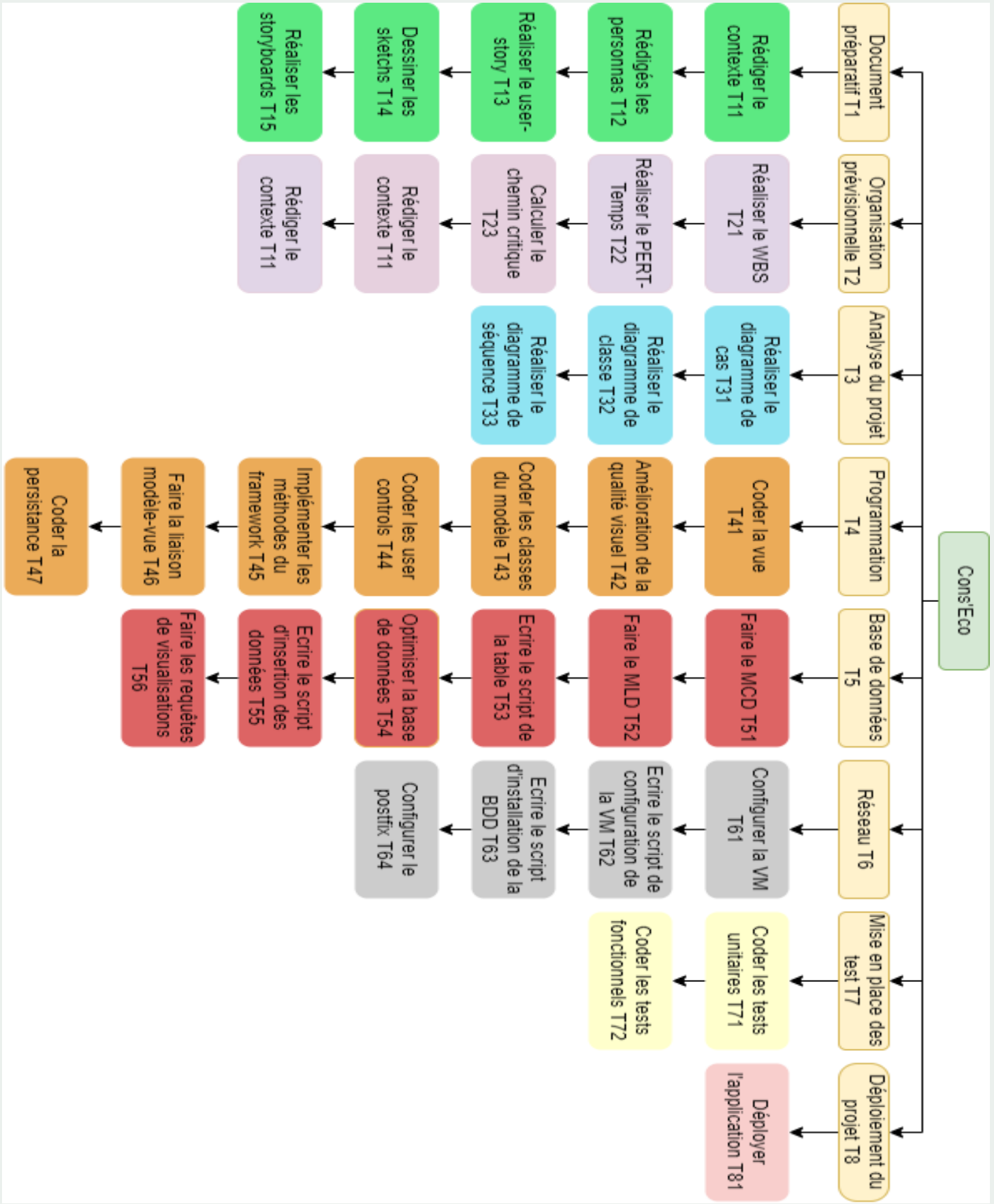
LACOTE RAPHAEL - LIVET HUGO - EVARD LUCAS -
MAYE NICOLAS - ASTOLFI VINCENT

Sommaire

A. Introduction

Dans ce dossier, nous allons définir toute la partie organisation du projet. En effet, dans le dernier dossier nous avons créé la base de notre projet et nous avons trouvé à quel besoin il devrait répondre. Désormais, nous devons découper notre projet en sous tâches afin de pouvoir visualiser plus simplement quel sera le travail à effectuer. Pour ce faire nous allons effectuer un WBS qui permettra de découper le projet en tâches principales qui seront elles-mêmes découpées en sous-tâches plus précises qui correspondront aux grandes étapes de notre projet. Après cela, il conviendra d'estimer la durée de chacune d'elles prévues dans le WBS. Cela nous permettra de calculer une durée globale au projet puis de déterminer des dates jalons qui correspondent soit au fin des grandes étapes du projet soit à des évènements importants (dossier à livrer par exemple). Suite à cela nous réaliserons un diagramme PERT qui nous permettra de déterminer un chemin critique et donc les tâches qui ne peuvent pas prendre de retard. Enfin, nous effectuerons une estimation des coûts prévisionnels pour permettre au client de pouvoir prévoir le budget nécessaire à ce projet. Pour finir, nous déterminerons les indicateurs de suivi de projet et qualités qui sont des facteurs humain ou non, ce qui permettra de savoir si le projet prend du retard ou si nous sommes dans les temps.

B. La décomposition du projet en tâches élémentaires - WBS



Description des tâches du WBS :

La première grande étape de la réalisation de notre application est la **réalisation de documents préparatifs** qui vont nous permettre de poser notre idée et de réfléchir à quel besoin elle va répondre. (T1)

- Dans un premier temps, nous allons **rédiger un contexte** qui va nous permettre de poser nos idées et de réfléchir à ce que fera vraiment notre application. Le contexte est un document rédigé qui explique l'application et son utilité sans aucun terme technique afin qu'il puisse être lu et compris de n'importe quelle personne même très éloignée du milieu de la programmation. (T11)
- La **réalisation des personas** va nous permettre d'imaginer des profils types d'utilisateurs de notre application et donc de réfléchir à comment notre application pourrait répondre à leurs besoins. (T12)
- Les **user-stories** sont une extension des personas. Ils vont nous permettre de raconter l'histoire des utilisateurs que nous avons imaginée et de réfléchir à comment ces derniers utilisent notre application (dans quel cas, comment, pourquoi?) (T13)
- Dessiner les **sketchs** consiste à créer les premières esquisses de notre application, de voir où seront placés les boutons, les informations etc… (T14)
- Les **storyboard**, permet de simuler une utilisation de l'application en détaillant à quoi sert chaque bouton, chaque page etc… de notre application. (T15)

Après avoir établi une idée globale du **fonctionnement de notre application** et quelles seront ses fonctionnalités, nous allons pouvoir nous organiser afin de réaliser au mieux notre application (gestion des délais, des budgets). (T2)

- Dans un premier temps nous avons réalisé le **WBS** ci-dessus afin de découper notre projet en tâches et sous-tâches afin de mieux réaliser le travail qui sera à accomplir. (T21)
- Après avoir découpé le projet en sous tâches nous allons réaliser le **PERT-temps**. Ce document va nous permettre de donner une durée à chacune de ces tâches et aussi de donner des antériorités à chacune de ces tâches. C'est-à-dire, de

dire quelle tâche nécessite d'en avoir terminé une autre au préalable. Nous allons aussi calculer, grâce à ces informations, les **dates de début et de fin de chacune des tâches** au plus tôt et au plus tard afin de pouvoir visualiser les timings de travail. (T22)

- Suite aux calculs des dates de début et de fin au plus tôt et au plus tard nous avons pu calculer des **marges** qui correspondent au temps que nous pouvons nous laisser avant de pouvoir lancer une tâche sans pour autant être en retard. Les tâches qui ont une marge nulle font partie du chemin critique. C'est-à-dire que si nous ne respectons pas les timings fixés sur ces tâches précises nous serons automatiquement en retard. (T23)
- A partir du **PERT-temps** nous allons nous répartir les tâches et dire dans lesquels nous aurons besoin d'être plusieurs à travailler dessus. Nous pourrons aussi déterminer une intensité pour chacune de ces tâches c'est-à -dire à la quantité d'effort qu'il sera nécessaire d'attribuer. (T24)
- Enfin, nous pourrons réaliser le **GANT prévisionnel** qui nous permettra de voir toute l'organisation que nous devrions respecter afin de finir le projet dans les temps. De plus, nous allons effectuer une estimation des coûts prévisionnels. (T25)

Une fois toute la partie organisation terminée, il nous faudra **analyser notre projet** afin de réfléchir à comment le programmer. (T3)

- Le premier diagramme à effectuer est le **diagramme de cas d'utilisation**. Il nous permettra de voir les différents acteurs de notre application (utilisateurs comme administrateurs) et de discerner les utilisations qu'ils feront de l'application. (T31)
- Le **diagramme de classe**, quant à lui, nous permettra de réfléchir à la structure interne de notre application. Il nous permettra de déterminer les différentes classes d'objet, comment elles seront reliées entre elles et quelles méthodes elles utiliseront. On pourra aussi observer les différents héritages, classes et interfaces. (T32)
- Enfin, le **diagramme de séquence** nous permettra de réfléchir à la structure interne de notre application et que se passera t'il entre une demande du client et la

réponse que lui enverra notre application, s'il y aura des intermédiaires, un passage par la base de données ou le serveur. (T33)

De l'analyse du projet en découle le début de la partie **programmation** qui va consister à la réalisation de notre projet (T4)

- Nous commencerons par **coder la vue.** Ainsi toute la partie visible par les utilisateurs sera effectuée. Cette partie sera donc très influencée par les sketchs que nous avons réalisés dans la tâche T14. (T41)
- Après avoir réalisé les vues de bases, il nous faudra **les améliorer tant au niveau de leurs responsivité qu' au niveau purement visuel** afin de les rendre plus agréable à l'œil. (T42)
- Pour coder le modèle, il nous faudra dans un premier temps **retranscrire les classes** que nous avons créées sur notre diagramme de classe afin de créer l'architecture de l'application. (T43)
- Les différentes vues de notre projet seront en majorité des **User-Control**. Il nous faudra réaliser le code behind de chacune d'entre elles afin de faciliter la navigation entre les différentes sections de l'application. Cela pourrait passer par la réalisation d'un Navigator par exemple. (T44)
- Une fois toutes les classes implémentées nous pourrons ajouter **leurs méthodes** qui permettront de faire toute les actions nécessaires au bon fonctionnement de l'application. (T45)
- Une fois les deux tâches précédemment terminées, nous ferons la **liaison entre le modèle et la vue.** Ainsi tout ce qui sera calculé ou donné par le modèle s'affiche dans la vue, ce qui permettra d'obtenir un premier produit fini de notre application. (T46)
- La dernière étape de la partie programmation sera de **coder une persistance,** ce qui va permettre de sauvegarder les données d'un même utilisateur entre deux utilisations. Un utilisateur se reconnaîtra grâce à son compte ce qui signifie que cette persistance devra être locale et distantes sur un serveur externe. (T47)

Pour sauvegarder les données des utilisateurs et faire des visualisations pour la partie d'aide à la gestion de budget, une **base de données** sera mise en place. (T5)

- Le **MCD** va nous permettre de faire le lien entre les différents éléments qui seront présents dans notre base de données. (T51)
- Le **MLD** va, quant à lui, nous permettre d'écrire d'une façon différente le MCD afin de pouvoir vérifier la cohérence de celui-ci et ainsi de peut être corriger les quelques erreurs que nous aurions pu faire. (T52)
- Le **script des tables** va créer toutes les tables ainsi que les attributs qui les composent dans la base de données. (T53)
- Notre base de données devrait pouvoir gérer des grandes plages de données. Ainsi, il faudrait que notre **base soit le plus optimisée possible** afin d'éviter des temps de latence qui pourraient dégrader l'expérience de l'utilisateur. (T54)
- Nous pourrons ensuite créer le **script** qui permettra de **récupérer les données** envoyées par l'application et les ranger au bon endroit dans notre base de données. (T55)
- Une fois tout cela terminé nous créerons les **requêtes** qui permettent de **créer les visualisations nécessaires** à l'outil d'aide à la gestion de budget. (T56)

La base de données sera hébergée sur un **serveur en réseau**. Il nous faudra donc imaginer une partie réseau. (T6)

- Dans un premier temps, nous configurons une **machine virtuelle** sur laquelle tournera notre serveur. (T61)
- Un script permettra de **configurer cette VM** pour que celle-ci soit possible de n'importe où, sur n'importe quel ordinateur et sans avoir rien à faire d'autre que de lancer le script. (T62)
- Une fois le serveur correctement initialisé, nous y installerons **notre base de données** afin qu'elle puisse être accessible de n'importe où. (T63)
- Nous configurons enfin un **serveur de mail** afin que lorsqu'un utilisateur utilisera l'option mot de passe oublié, il puisse recevoir un mail lui indiquant comment le modifier. (T64)

Quand toute la programmation sera terminée, il nous faudra mettre en place des **tests** afin de vérifier que toutes les fonctionnalités que nous avons créées fonctionnent dans tout cas de figure et de vérifier que nous avons réfléchi à toutes les éventualités possibles. (T7)

- Les **tests unitaires** nous permettront de tester chacune de nos méthodes indépendamment, précisément et sous plusieurs cas de figures les unes après les autres. (T71)
- Les **tests fonctionnels** vont nous permettre de vérifier en conditions réelles d'utilisation que l'application réagit bien comme elle est censée le faire et qu'elle ne possède aucun bug. (T72)

Enfin, nous pourrons **déployer l'application** sous forme d'un exécutable qui sera téléchargeable et qui permettra de lancer l'application depuis n'importe quel ordinateur qui possèderait cet exécutable. (T8 - T81)

C. Estimations des tâches, durée globale et dates jalons

Le **WBS** nous a permis de **décomposer notre projet** en sous tâches. Après l'avoir découpé ainsi, nous pouvons imaginer le temps qu'il sera nécessaire à sa réalisation.

Tâche	Intituler	Durée (1 heures)
T11	Rédiger le contexte	1
T12	Rédiger les personas	1
T13	Rédiger les users stories	2
T14	Dessiner les maquettes	6
T15	Réaliser le story-board	3
T21	Réaliser le WBS	1
T22	Réaliser le PERT-temps	1
T23	Calculer le chemin critique	1
T24	Réaliser le PERT-Charges	2
T25	Réaliser le GANT prévisionnels	4
T31	Réaliser le diagramme de cas d'utilisation	2
T32	Réaliser le diagramme de classe	4
T33	Réaliser le diagramme de séquence	2
T41	Coder la vue	20
T42	Amélioration de la qualité visuel	10
T43	Coder les classes du modèle	18
T44	Coder les user controls	25
T45	Implémenter les méthodes du framework	7
T46	Faire la liaison modèle-vue	20
T47	Coder la persistance	25
T51	Faire le MCD	2
T52	Faire le MLD	1
T53	Ecrire le script de la table	8
T54	Optimiser la base de donnée	20
T55	Ecrire le script d'insertion des données	6
T56	Faire les requêtes de visualisations	6
T61	Configurer la VM	2
T62	Ecrire le script de configuration de la VM	5
T63	Ecrire le script d'installation de la BDD	1
T64	Configurer le postfix	3
T71	Coder les tests unitaires	15
T72	Coder les tests fonctionnels	15
T81	Déployer l'application	5

Schéma montrant l'estimation de la durée des tâches

Le document ci-dessus représente donc les durées que nous avons estimées nécessaires à leurs réalisations. Nous pouvons remarquer que les tâches les plus longues sont celles du groupe T4. En effet, ce sont les **tâches charnières de notre projet**. Leur réalisation impactera directement la qualité de l'application finale. La partie test est elle aussi plus longue que les autres car il sera important de veiller à la qualité de notre application ainsi qu'à sa capacité à fonctionner sur le long terme. Dans la même optique de rendre l'application viable sur le long terme sans forcément avoir besoin de grosses maintenances, nous avons prévu de passer longtemps sur

l'optimisation de notre base de données afin qu'elle ne devienne pas surchargée ou lente à l'utilisation.

La durée globale de ce projet, avec ses durées par tâches, est de 113 heures de travail. Sachant que nous avons 120 heures de créneaux réservés à la réalisation de cette application, cette durée semble adaptée. Cependant, pour être sûr de ne pas se faire surprendre par la durée de ce projet, nous avons décidé de réaliser le **PERT-Charges** suivant afin de pouvoir mieux visualiser le taux d'investissement nécessaire par tâche ainsi que le nombre de personnes nécessaires pour effectuer celles-ci.

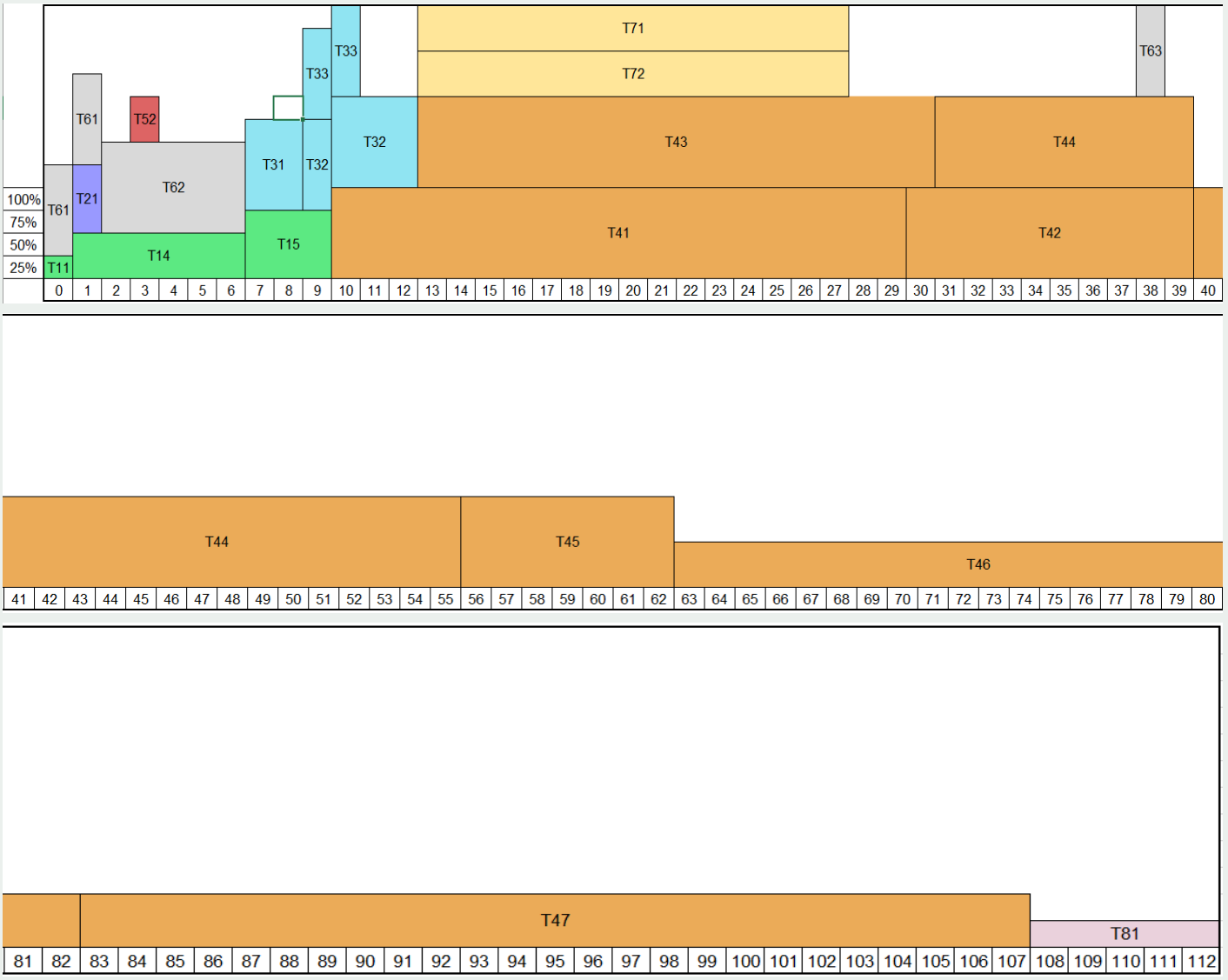
Tâche	Temps	Ressources	intensité (%)
T11	1	R - H - L - N - V	25
T12	1	H - L	25
T13	2	H - L	25
T14	6	R - N	50
T15	3	N	75
T21	1	R - H - L - N - V	75
T22	1	V - L - H	75
T23	1	V - H	25
T24	2	V - L	75
T25	4	V - L - H	75
T31	2	N - R - V - H - L	100
T32	4	N - R - H - L	100
T33	2	N - R - V - H - L	100
T41	20	R - H - L - N - V	100
T42	10	R - H - L - N - V	100
T43	18	R - H - L - N - V	100
T44	25	R - H - L - N - V	100
T45	5	R - H - L - N - V	100
T46	20	R - H - L - N - V	50
T47	25	R - L - H - N - V	50
T51	2	L - R - V	50
T52	1	L - R - H - N	50
T53	8	L - H - V - R	100
T54	20	L - H - R	75
T55	6	L	75
T56	6	L - H	50
T61	2	H - N - V	100
T62	5	L - N - V	100
T63	1	H - L - N - R - V	100
T64	3	H - R	50
T71	15	R - H - L - N - V	50
T72	15	R - H - L - N - V	50
T81	5	R - H - L - N - V	25

PERT-Charge prévisionnel de notre projet.

Après avoir réalisé ce PERT-Charge globale du projet, nous avons réalisé les **PERT-Charge personnel** de chacune des personnes présentes sur ce projet qui sont : Raphaël (R) , Hugo (H), Nicolas (N), Lucas (L), Vincent (V). Nous avons donc dans un premier temps réaliser la version non optimisée de ceux-ci.

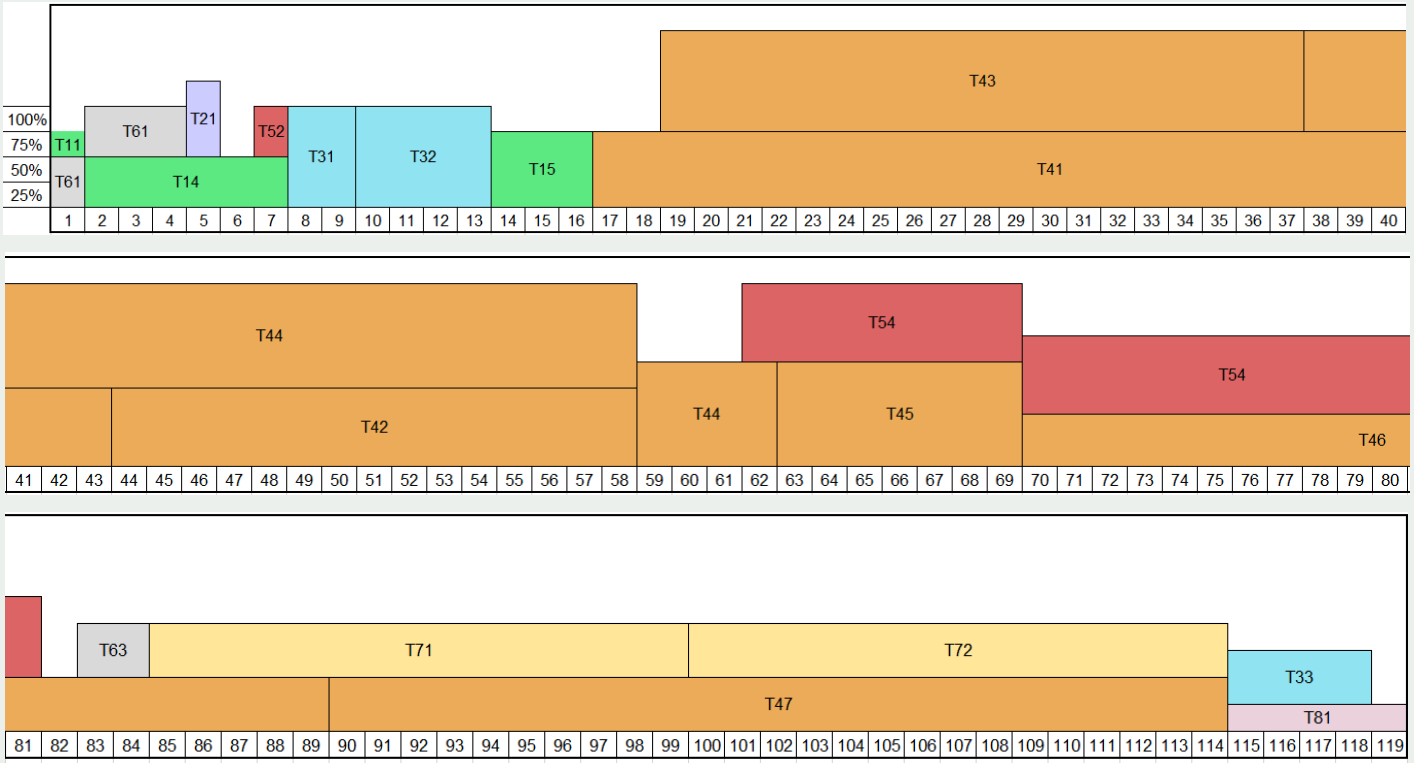
Les tâches ont été distribués selon plusieurs critères:

- Toutes les **personnes** du groupe devaient participer à **toutes les compétences**
- Les membres du groupe ont participé à l'élaboration des tâches qu'ils aimaient bien faire et celles où ils étaient **les plus compétents**.
- Un membre ne devait **pas travailler plus qu'un autre**



PERT-Charge du projet

Ce PERT-Charge générale nous a permis d'observer que, dans l'état actuel des choses, le projet est presque irréalisable. Ainsi, nous avons réalisé le **PERT-charge optimisé** afin de nous permettre de savoir quelle tâche faire à quel moment dans le meilleur des cas et de voir si le projet était réalisable si nous respections ces nouveaux délais.



PERT-Charge optimisé du projet

Sur ce PERT-Charge optimisé (celui de Nicolas), on peut remarquer qu'à certains moments il doit travailler à plus de 100% de ses capacités, cela signifie que les membres devront **travailler en dehors des créneaux** réservés au projet. Cela sera nécessaire afin de pouvoir finir le projet dans les temps. A d'autres moments, ils travaillent à moins de 100%, ce temps lui permettra de vérifier que tout est correctement réalisé et **d'aider les autres membres** sur leurs tâches.

Ce PERT-Charge optimisé agit comme une aide et ce n'est donc pas une nécessité absolue de le respecter, ainsi il **pourra être modifié** en cas d'imprévus, de retard ou d'avance d'un des membres sur le projet.

Pour reconnaitre les dates jalons de notre projet nous avons plusieurs indicateurs possible :

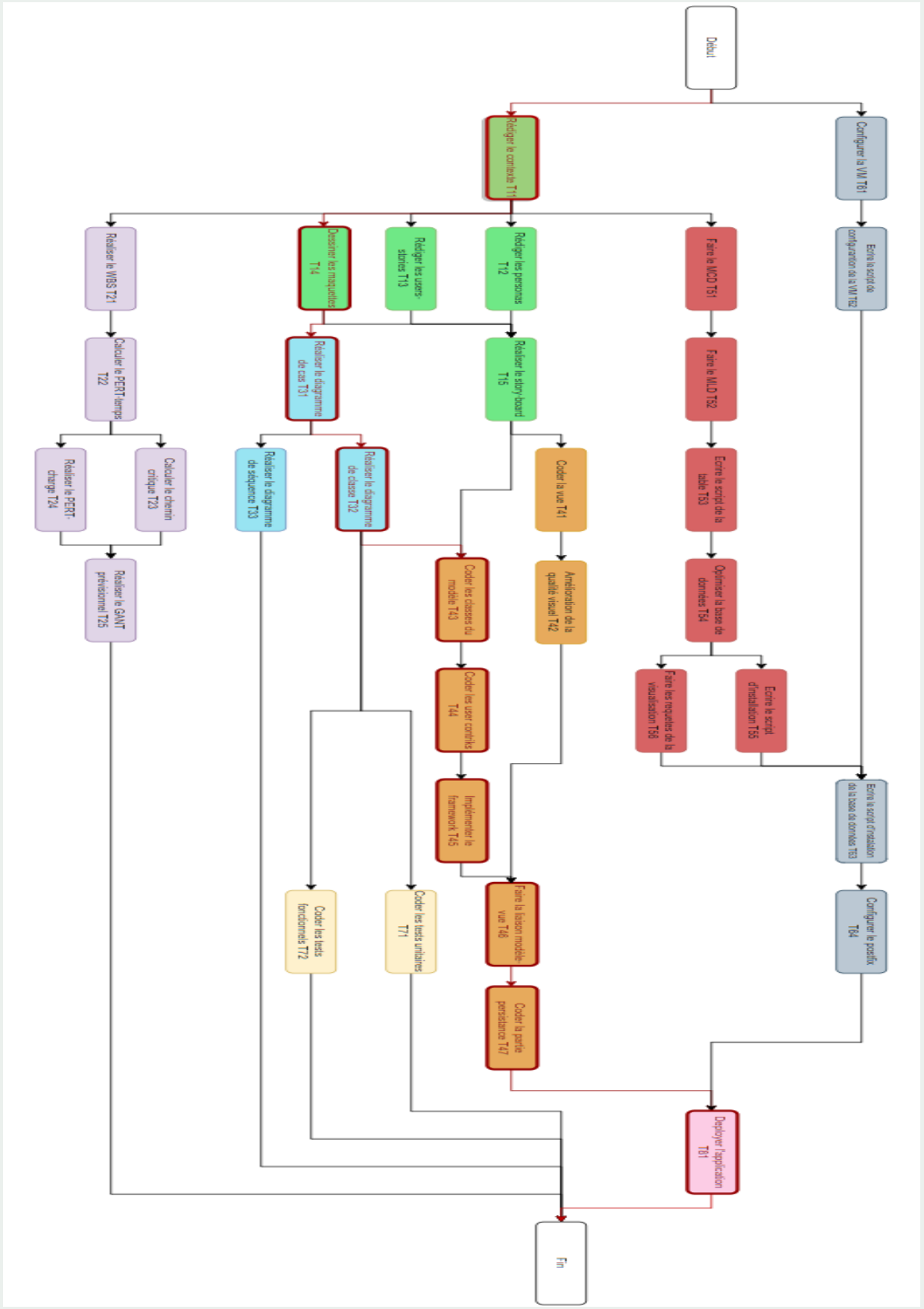
- Dans un premier temps les dates de rendu obligatoires d'avancement de projet comme par exemple le dossier à rendre pour le 30 Septembre ou encore celui à rendre pour le 7 Octobre (celui-ci). Ces **dossiers** seront **à rendre régulièrement** tout le long du projet et représentent chacun les **jalons de l'avancement de notre projet.**
- La date de fin du projet représente la date jalon final. Elle est prévue pour le … et entre le jour de début et de fin du projet nous sera laissé **120 heures dédiées** pour la réalisation de ce projet, ce qui devrait nous laisser largement le temps de réaliser notre projet si nous sommes bien organisés.
- Enfin, chacune de nos **fins de sprint** représenterait une date jalon. En effet, les sprints sont un **moyen de découper le projet en différentes étapes**. Leurs fins peuvent donc servir de jalon à respecter pour finir le projet dans les temps.
- Nous pourrons regrouper ces différentes dates jalons directement sur le **dépôt CodeFirst** grâce à la **fonctionnalité jalon** qui nous permettra de créer des dates jalons et d'y assigner un certain nombre de tickets. Dans la logique, tous les tickets assignés devront être fermés avant la fin de ce dernier.

D. PERT, détermination chemin critique, GANTT prévisionnel

				Dates au plus tot		Dates au plus tard		
Tâche	Intituler	Tache anterieure	rée (1 heur	Début	Fin	Début	Fin	Marges
T11	Rédiger le contexte		1	0	1	0	1	0
T12	Rédiger les personas	T11	1	1	2	9	10	8
T13	Rédiger les users stories	T11	2	1	3	8	10	7
T14	Dessiner les maquettes	T11	6	1	7	1	7	0
T15	Réaliser le story-board	T12 / T13 / T14	3	7	10	10	13	3
T21	Réaliser le WBS	T11	1	1	2	104	105	103
T22	Réaliser le PERT-temps	T21	1	2	3	105	106	103
T23	Calculer le chemin critique	T22	1	3	4	108	109	105
T24	Réaliser le PERT-Charges	T21	2	2	4	107	109	105
T25	Réaliser le GANT prévisionnels	T23 / T24	4	4	8	109	113	105
T31	éaliser le diagramme de cas d'utilisati	T14	2	7	9	7	9	0
T32	Réaliser le diagramme de classe	T31	4	9	13	9	13	0
T33	Réaliser le diagramme de séquence	T31	2	9	11	111	113	102
T41	Coder la vue	T15	20	10	30	33	53	23
T42	Amélioration de la qualité visuel	T41	10	30	40	53	63	23
T43	Coder les classes du modèle	T15 / T32	18	13	31	13	31	0
T44	Coder les user controls	T43	25	31	56	31	56	0
T45	Implémenter les méthodes du framework	T44	7	56	63	56	63	0
T46	Faire la liaison modèle-vue	T45 / T42	20	63	83	63	83	0
T47	Coder la persistance	T46	25	83	108	83	108	0
T51	Faire le MCD	T11	2	1	3	67	69	66
T52	Faire le MLD	T51	1	3	4	69	70	66
T53	Ecrire le script de la table	T52	8	4	12	70	78	66
T54	Optimiser la base de donnée	T53	20	12	32	78	98	66
T55	Ecrire le script d'insertion des données	T54	6	32	38	98	104	66
T56	Faire les requêtes de visualisations	T54	6	32	38	98	104	66
T61	Configurer la VM		2	0	2	97	99	97
T62	Ecrire le script de configuration de la VM	T61	5	2	7	99	104	97
T63	Ecrire le script d'installation de la BDD	T62 / T56 /T55	1	38	39	104	105	66
T64	Configurer le postfix	T63	3	39	42	105	108	66
T71	Coder les tests unitaires	T32	15	13	28	98	113	85
T72	Coder les tests fonctionnels	T32	15	13	28	98	113	85
T81	Déployer l'application	T64/T47	5	108	113	216	113	0

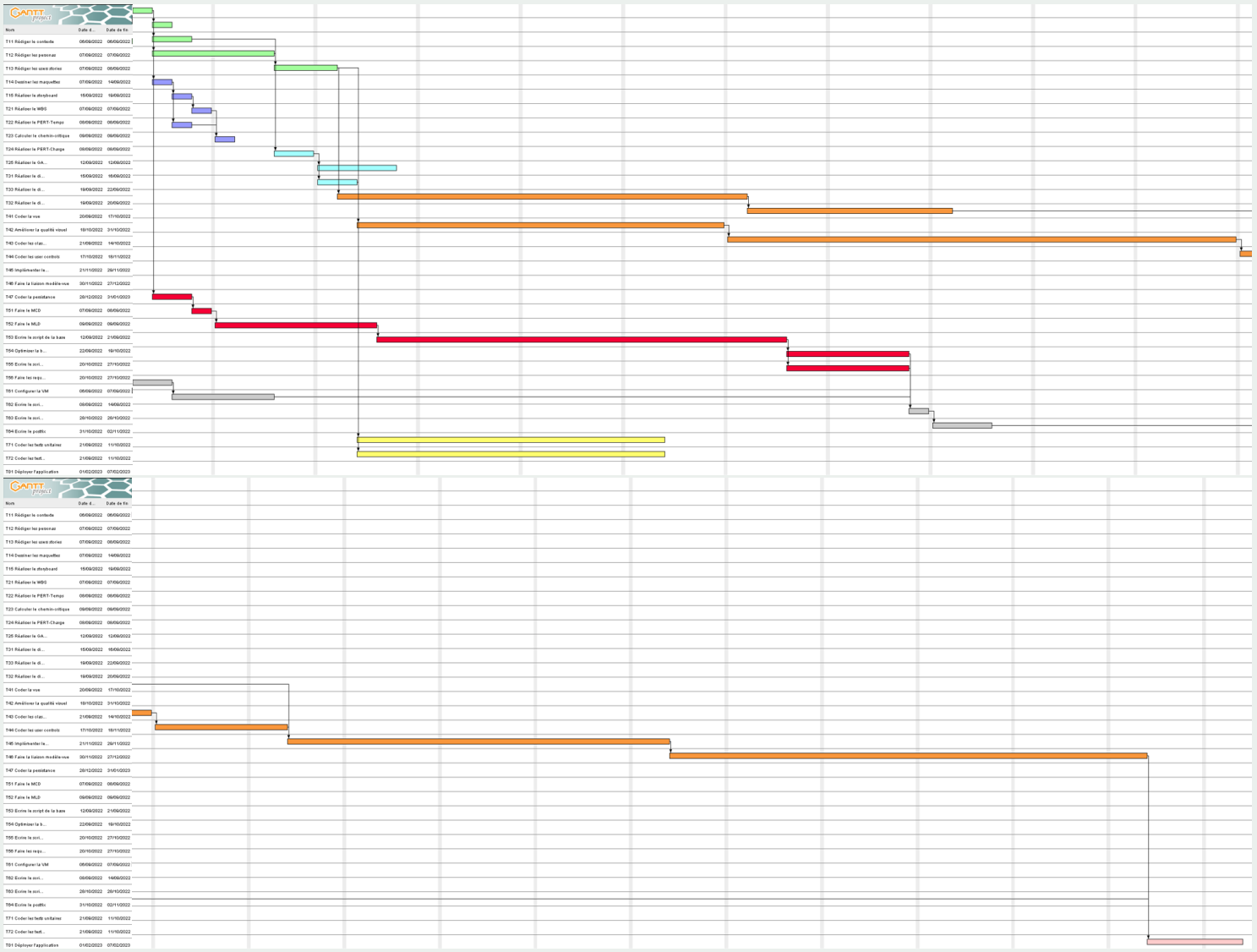
PERT-temps

Ce **PERT-Temps** nous a permis de réfléchir à quelle tâche nécessite la complétion d'autres en amont. Ainsi, nous avons pu calculer les dates au plus tôt et au plus tard puis de constater que certaines tâches nous laisseraient une grande marge. Ce qui signifie que celles-ci ne sont pas déterminantes dans l'avancement du projet et que si nous devions rattraper un retard sur d'autres avec une marge plus faible, il faudrait le faire en **décalant les tâches ayant des durées de marges élevées**.



Détermination du chemin critique

Dans le schéma ci-dessus représentant notre ordre des tâches tout au long du projet nous pouvons observer, entouré en rouge, le chemin critique. Il correspond aux tâches possédant une marge de zéro. C'est-à-dire, que si l'une de ces tâches n'est pas effectuer dans le temps qui lui est attribué alors le retard accumulé ne pourra plus être rattrapé sauf si nous travaillons hors des horaires prévus pour le projet. Ainsi, les **tâches formant le chemin critique** sont les tâches que nous nous devons d'exécuter avec la plus grande attention et en essayant au possible de **respecter le temps que nous avons prévu d'y affecter.**



GANTT Prévisionnel

Sur ce **GANTT**, nous pouvons voir que les tâches qui prendront le plus de temps sont celles portant sur la partie programmation (en orange), étant donné qu'elles font partie du chemin critique, il sera donc important de **ne pas prendre du retard sur celle-ci.** On remarque également que la plupart de ces tâches se situent au début du projet, il nous faudrait les répartir équitablement tout au long du projet.
On voit selon ces prévisions que la **date limite du projet devrait être dépassée.** Il sera donc important de travailler chez nous en plus des heures dédiées afin de terminer le projet dans les temps.

E. Estimation des coûts prévisionnel

Pour pouvoir prévoir les coûts nous avons réalisés les tableaux suivants :

Cout d'un développeur (€/heure)	44,04
temps de travail (en heures)	120
nombre de développeurs sur le projet	5
Total (€)	26424

Matériel	Ordinateurs portables	Maintient du serveur (à l'année)	License logiciel microsoft
Prix unitaire	549,99	1020	33,5
quantité	5	1	5
Total (€)	2749,95	1020	167,5

Tableaux d'estimations des coûts prévisionnels.

Dans un premier temps, nous avons estimé le coût de notre travail. Le coût moyen d'un développeur est de 44,04€ par heure (voir source). Nous sommes 5 pour ce projet et le travail devrait durer un **total de 120 heures** (heures spécialement réservées pour le projet). Ainsi, le coût de la main-d'œuvre devrait se retrouver à 26 424€ pour cette

application. Cependant, comme vu dans les PERT-charges optimisés ci-dessus, il sera impossible de finir ce projet en 120 heures si nous voulons y inclure toutes les fonctionnalités prévus. La solution pourrait donc être d'effectuer des **heures supplémentaires**. Ces heures seraient majorées à un minimum de 10%. Néanmoins, ici il n'y aura sûrement pas d'accord d'entreprise, on utilisera donc les taux légaux de majoration. Ainsi, les 8 premières heures supplémentaires seraient majorées à 25%, soit 55.05€ de l'heure pendant 8 heures. Puis 50% de majoration par heure, soit 66.06€ de l'heure. Si nous prenons l'exemple du PERT-Charges optimisé de Nicolas, nous observons qu'il devra effectuer 30 heures supplémentaires s'il respecte tous ses délais. Nous pouvons donc calculer sa rémunération totale : 83*44.04+8*55.05+22*66.06 = **5 549,04€** ce qui représente un total pour les 5 membres de l'équipe : **27 745,20€**

De plus, il semble nécessaire de **fournir le matériel nécessaire** à la bonne réalisation du projet. En effet, fournir le matériel adapté est l'un des devoirs de l'employeur. Ainsi, nous avons pris pour exemple les coûts de l'ordinateur portable Lenovo ideapad 3 car il nous semble être un bon rapport qualité prix quant à la réalisation de ce projet.
Cependant, il reviendra à l'employeur de choisir les ordinateurs qui seront fournis tant que ceux-ci sont suffisants à la réalisation du projet dans les meilleures conditions.
Cet ordinateur à un prix unitaire de 549,99€. Nous sommes 5 à réaliser ce projet. Ainsi, le coût total du matériel sera de 2749,95€.

De plus, il faut prendre en compte le prix de la **licence logiciel microsoft** qui sera nécessaire à la réalisation des différents dossiers de documentation ou de gestion de projet. Cette licence a un coût de 39.99€ par personne. Ces licences coûtent donc **199.95€** au total pour l'employeur.

Pour la programmation nous avons prévu d'utiliser le logiciel **Microsoft Visual Studio** qui est un logiciel **gratuit**, il n'entraînera donc pas de coûts supplémentaires. Cependant, ce logiciel n'est pas toujours le plus fiable ou stable. Ainsi, si l'entreprise souhaite offrir les meilleures conditions de travail à son équipe, il existe des

alternatives telles que **Rider** qui sont aussi des logiciels qui permettent la création d'application WPF mais nécessite de **payer une licence**. Pour Rider, la license coûte 14,90€ par mois et par personnes, ce qui reviendrait à 14,90*5*4=**298€** étant donné que nous serons 5 à travailler sur le projet et que le projet devrait être fini dans 4 mois.

Enfin, l'exécutable de notre application sera publié sur une page web hébergée, la base de données doit être accessible en ligne par le client à n'importe quel instant. Un serveur mail sera aussi déployé afin de gérer les mots de passe oubliés. Nous avons opté pour **l'hébergement d'un serveur** chez OVH car c'est un hébergeur connu ayant pour réputation d'offrir de bonnes performances et une sécurité avancée sur leurs produits, dont le coût de maintien du serveur est de 85*12=**1020€ par an hors taxes.**

F. Indicateurs de suivi du projet et de qualité

Les indicateurs de suivi de projet dont nous pourrons disposer sont :

- Les **remarques de notre tuteur de projet Mr. BOUHOURS.**
- Les **dossiers** à rendre régulièrement.
- L'utilisation de tableau **KanBan** pour nos périodes de sprint.
- L'utilisation d'un **Burn Down Charts.**

Explications :

Après chaque heure de créneau réservé à l'avancée de notre projet, nous devons **tenir au courant notre tuteur** de projet Mr. BOUHOURS. Ainsi, il nous indiquerait si notre projet prend du retard ou non. De plus, il nous a indiqué qu'il nous dirait si nous partions dans la mauvaise direction que ça soit tant au niveau de l'organisation

ou au niveau du contenu de l'application elle-même. Ainsi nous aurions un bon indicateur de suivi de projet et de qualité à la fois.

Les **dossier à rendre** régulièrement dont nous parlons dans les dates jalons seront eux aussi de bons indicateurs de suivi de projet car il nous permettrons de pouvoir nous rendre compte de si nous sommes dans les temps au niveau de l'avancée du projet ou si, au contraire, il nous faudra accélérer la cadence sans quoi nous risquerions de nous retrouver en retard et la qualité de nos rendus s'en retrouverait impactés négativement.

Nous utiliserons aussi un **tableau KanBan** qui nous permettra de suivre régulièrement notre projet. En effet, ces tableaux vont nous permettre de répartir nos tâches en fonction de leurs avancement et donc de voir en temps réel quelles tâches sont les plus avancées ou lesquelles sont encore à effectuer ou en cours de développement. De plus, nous voulons utiliser une méthode agile Scrum pour nous organiser. Ainsi, pour chaque **Sprint** nous aurons un KanBan associé qui nous permettra de suivre l'avancée de notre sprint.

Enfin, nous souhaiterions mettre en place un **Burn Down Charts** pendant nos périodes de Sprint afin de pouvoir suivre jour après jour l'avancement de nos tâches et l'avancement de sprint en cours tout en observant la somme des efforts qu'il nous reste à accomplir et en la comparant à la courbe idéal d'avancement de sprint.

Tous ces indicateurs de suivi de projet feront l'objet d'une **note** nous permettant ainsi de savoir si la qualité globale de notre projet est satisfaisante ou non. De plus les **avertissements ou conseils** de Mr. Bouhours nous permettront eux aussi de connaître la qualité de notre projet régulièrement.

G. Conclusion

En conclusion, nous avons pu remarquer grâce à ce dossier que ce projet est très **long et conséquent en termes de travail demandé.** Il va demander la réalisation d'un très grand nombre de tâches, lesquelles sont pour la plupart plutôt longues. Si nous voulons réaliser ce projet à temps, nous allons devoir travailler hors des créneaux réservés au projet ou bien enlever certaines fonctionnalités prévues initialement qui ne pourront pas être implémentées dans les temps.
De plus, ce projet possède un coût plutôt élevé, ce qui signifie que le client doit se projeter et qu'il faudra **maintenir une grande communication entre lui et l'équipe de réalisation** afin que celui-ci continue de croire au projet et continue d'investir dedans.

H. Source

Coût moyen d'un développeur par heure :
https://www.monpro.fr/prix/developpement-informatique/

Coût de la license Microsoft office 2019 :
https://agm-software.com/produit/microsoft-office-home-business-2019-pc-mac/