GNG 2501 Modèle de rapport

LIVRABLE J: Manuel d'utilisation

Soumis par:

Groupe FA3.1

Hugo Paré, 300173735

Hedi Ben Abid, 300123192

Mathieu Symons, 300120812

Simon Jolicoeur, 300117133

5 novembre 2020

Université d'Ottawa

Abstrait

Notre projet consiste à la mise en œuvre d'une application *Android* d'une carte de sécurité du campus de l'Université d'Ottawa. Cette application est importante auprès de notre équipe ainsi que pour notre client, car la sécurité est essentielle afin d'assurer le bon fonctionnement de l'Université.

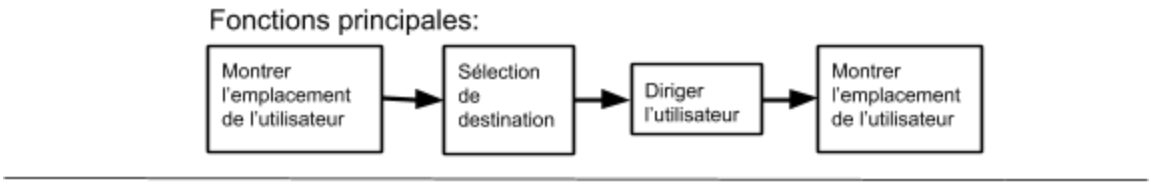
L'application utilise le *API* de *Google Maps* afin de créer la carte visible sur l'application et à partir de ce dernier, nous pouvons concevoir le système de navigation entre les bâtiments du campus. Les utilisateurs auront la possibilité de naviguer aux différents points de rassemblement ou au points de services (santé et sécurité) affichés sur leur écran. Afin de permettre la navigation intérieur, les utilisateurs pourront sélectionner un bâtiment en particulier et un plan d'étage apparaîtra. Ils pourront entrer dans la barre de recherche le code affiché dans le bâtiment pour se situer sur le plan d'étage.

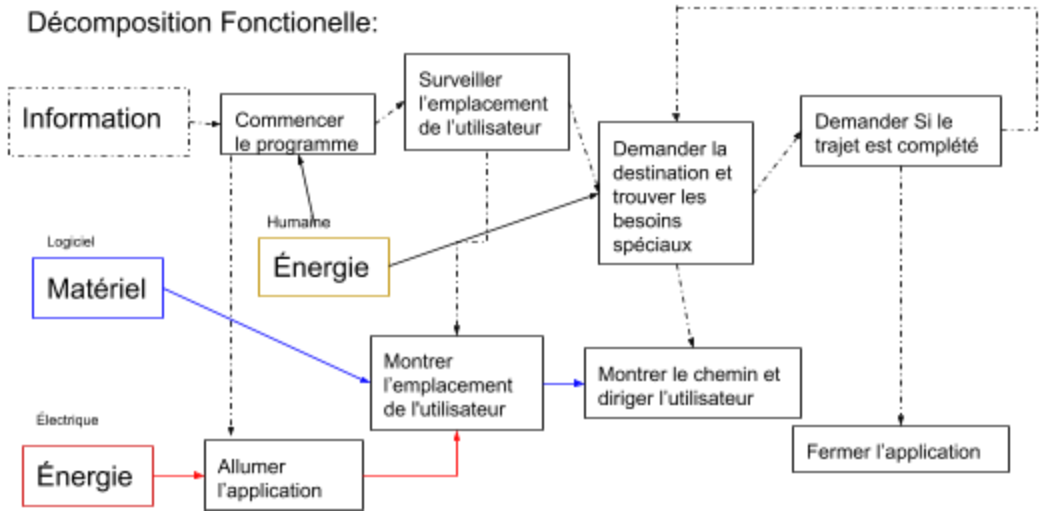
Nous souhaitons venir en aide aux membres de l'Université d'Ottawa afin de rendre le campus plus sécuritaire pour tous.

Table des matières

Annexes

Annexe 1 : Décomposition fonctionnelle





[1]

[1] Livrable C

Annexe 2 : Modèle d'affaires

<table>
<tr>
<td rowspan="2">Partenaires clés

- L'Université d'Ottawa
- Google Maps
- Utilisateurs</td>
<td>Activités clés

Gestion de la plateforme

Gestion des demandes</td>
<td rowspan="2">Proposition de valeur

Utilisateurs:
- Se situer plus facilement sur le campus
- Moins de temps perdu entre classes
- Savoir où se rendre en cas d'urgence

Client (Université d'Ottawa):
- Sécurité augmenté sur le campus</td>
<td>Relation avec le client

Avoir un email qui permet au client de nous contacter</td>
<td rowspan="2">Segments de la clientèle

Utilisateurs:
- Nouveau au campus
- Savoir où se situe les services de l'université
- Besoins spéciaux, handicaps (Navigation Accessible)</td>
</tr>
<tr>
<td>Ressources clés

Plateforme d'application mobile

Étudiants et employées de l'Université d'Ottawa</td>
<td>Canaux de distribution

Application mobile</td>
</tr>
<tr>
<td colspan="3">Structure des coûts

Le coût potentiel pour les fonctions avancées du logiciel HERE
Un coût de 45 $ par mois pour des options plus avancés</td>
<td colspan="2">Sources de revenus

Revenus venant des publicités</td>
</tr>
</table>

2

[2] Livrable F

Annexe 3 : Points de rassemblement

uOttawa: Assembly Locations for Building Evacuations
Main Campus Buildings

Building Names and/or Address	Primary/Outdoors	Short-Term/Indoors/ Inclement Weather	Long-Term / Inclement Weather
Academic Hall 135 Séraphin-Marion	Front of Tabaret	Tabaret	Montpetit Gym
Academic Writing Help Center (SASS) 110 University	85 University on sidewalk	UCU Concourse	Montpetit Gym
Alex Trebek Hall 157 Séraphin-Marion	Sidewalk across Séraphin-Marion	Tabaret	Montpetit Gym
ARC 25 Templeton	Across Templeton in front of Minto Sports Complex	Inside Minto Sports Complex	Montpetit Gym
Biosciences 30 Marie-Curie	Front of STEM or in front of CAREG	STEM	Montpetit Gym
CAREG 20 Marie-Curie	Front of Lamoureux	CRX	Montpetit Gym
Colonel By 161 Louis-Pasteur	Up-hill sidewalk along STEM or Lot Z	STEM	Montpetit Gym
CRX 145 Jean-Jacques-Lussier	Sidewalk across Louis-Pasteur	UCU Concourse	Montpetit Gym
Desmarais 55 Laurier	North side parking lot and South side along Laurier Street sidewalk	Tabaret	Montpetit Gym
D'Iorio 10 Marie-Curie	Front of STEM or front of CAREG	STEM	Montpetit Gym
Fauteux 57 Louis-Pasteur	Across Louis-Pasteur	UCU Concourse	Montpetit Gym
FSS 120 University	University Park (UCU)	UCU Concourse	Montpetit Gym
Gendron 30 Marie-Curie	Front of STEM or in front of CAREG	STEM	Montpetit Gym
GSAED Grad House 601 Cumberland	Across street on Thompson side	UCU Concourse	Montpetit Gym
Hagen Hall 115 Séraphin-Marion	Front of Tabaret	Tabaret	Montpetit Gym
Hamelin 70 Laurier	Between Simard and Perez Halls	Simard	Montpetit Gym
Lamoureux 145 Jean-Jacques-Lussier	Sidewalk along University Place	UCU Concourse	Montpetit Gym
Marion 140 Louis-Pasteur	Front of STEM	STEM	Montpetit Gym
Minto Sports Complex 801 King Edward &	Corner of King Edward & Mann and in front of ARC	ARC	Montpetit Gym

Building Names and/or Address	Primary/Outdoors	Short-Term/Indoors/ Inclement Weather	Long-Term / Inclement Weather
40 Templeton			
Montpetit 125 University	Front of Lamoureux, Place de l'Université and University Park (UCU)	UCU Concourse	Sports Complex
Morisset 65 University	Parking Lot K, 1st floor level terrace and park	UCU Concourse	Montpetit Gym
Perez 50 University	Front of Simard or sidewalk front of 601 Cumberland	Simard	Montpetit Gym
Sacred Heart Church 591 Cumberland	601 Cumberland	UCU Concourse	Montpetit Gym
Simard 60 University	Front of Perez	UCU Concourse	Montpetit Gym
SITE 800 King Edward	Sidewalk along Colonel By (770 K.E.) or along OLRT beside Colonel By	STEM	Montpetit Gym
STEM 150 Louis-Pasteur	Front of Colonel By (161 L-P) and front of Marion	SITE	Montpetit Gym
Tabaret 550 Cumberland	Front of Tabaret	Desmarais	Montpetit Gym
University Center 85 University	University Park, 90 University, sidewalk of Louis-Pasteur	FSS	Montpetit Gym
Vanier 136 Jean-Jacques-Lussier	Front of Lamoureux	UCU Concourse	Montpetit Gym
Visual Arts 100 Laurier	Front of Sacred Heart Church or Arts Court	Simard	Montpetit Gym
William Commanda 52 University	Arts Court	Simard	Montpetit Gym
15-17 Copernicus	Parking Lot P	Fauteux	Montpetit Gym
538-540-542 King Edward	Parking Lot P	Fauteux	Montpetit Gym
545-555 King Edward	Parking Lot T	585 King Edward	Montpetit Gym
554-556-558 King Edward	Parking Lot P	Fauteux	Montpetit Gym
562-598 King Edward	Parking Lot P	Fauteux	Montpetit Gym
585 King Edward	Parking Lot J	UCU Concourse	Montpetit Gym
600 King Edward	Between 598 & 562 King Edward	Fauteux	Montpetit Gym

Building Names and/or Address	Primary/Outdoors	Short-Term/Indoors/ Inclement Weather	Long-Term / Inclement Weather
603 King Edward	Parking Lot J	585 King Edward	Montpetit Gym
631 King Edward	Parking Lot U	UCU Concourse	Montpetit Gym
647 King Edward	Parking Lot U	141 Louis-Pasteur	Montpetit Gym
190 Laurier	Parking Lot T	585 King Edward	Montpetit Gym
129-139-141 Louis-Pasteur 100 Marie-Curie	Front of STEM, sidewalk Brooks side	LMX / CRX	Montpetit Gym
109, 113 Osgoode	Parking Lot J	UCU Concourse	Montpetit Gym
1 Stewart	Parking Lot C	Tabaret	Montpetit Gym
30-32-34-36-38-40 Stewart	Parking Lot C	Tabaret	Montpetit Gym
143-145-147 Séraphin-Marion	Sidewalk across Séraphin-Marion	Tabaret	Montpetit Gym
100 Thomas-Moore	Sidewalk Fauteux side	Fauteux	Montpetit Gym
200 Wilbrod	Parking Lot	Tabaret	Montpetit Gym

3

3

https://www.uottawa.ca/are-you-ready/sites/www.uottawa.ca.are-you-ready/files/assembly_locations_for_uottawa_building_evacuations_-_main_campus_en.pdf

1 Introduction

L’accessibilité et la sécurité sont des facteurs importants de la vie de tous les jours. Ils sont encore plus cruciaux en cas d’urgence. L’Université d’Ottawa met en priorité le bien-être de ses employés et de ses étudiants et ils savent que naviguer sur le campus n’est pas toujours évident. Pour cette raison, le personnel de l’Université d’Ottawa a demandé aux étudiants du groupe FA3.1 de créer une application qui oriente les utilisateurs et rend le campus plus accessible.

La navigation sur le campus est importante pour tous. Une mince poignée de personnes connaissent tous les points de rassemblement à travers le campus, c’est pourquoi notre application vise autant les étudiants que les professeurs, ainsi que le restant du personnel de l’Université. En cas d’urgence, les utilisateurs auront accès à toutes les informations nécessaires afin de se déplacer rapidement et efficacement au point de rassemblement le plus près ou à la pharmacie.

Nous nous démarquons des autres applications de sécurité, car nous avons la sécurité des gens et de l’université à cœur. Afin de rendre le campus un endroit plus sécuritaire, nous offrons aux utilisateurs des besoins spécifiques à ce campus, par exemple les multiples points de services importants ou les points de rassemblement. Les utilisateurs pourront naviguer sur le campus d’un bâtiment à l’autre en plus d’avoir un système de localisation à l’aide des plans intérieurs des bâtiments. Pour finir, notre but ultime est de rendre notre application le plus accessible possible en permettant à nos clients et utilisateurs de télécharger l’application gratuitement afin que chacun puisse trouver leur chemin rapidement en cas d’urgence.

2 Manuel d’utilisation

2.1 Attributs, fonctions et capabilités

2.1.1 Implementation du API de *Google Maps*

Afin de simplifier le code de notre application, nous avons décidé d'implémenter une carte API de *Google Maps* (*voir Appendice 5.1*). Cette fonctionnalité est la base de notre application, sans elle nous aurions dû passer beaucoup de temps à concevoir notre propre carte du campus virtuel. De plus, plusieurs des fonctions offertes par cette carte de *Google Maps* nous ont été utiles.

Tout d’abord, nous avons fait une demande pour avoir accès à une clé de la carte IPA. Après avoir reçu cette permission, nous nous sommes tout de suite lancés dans l’implémentation de la carte sur notre application déjà conçue. L’apparence de la carte est identique à celle de *Google Maps*, il sera alors plus facile pour les utilisateurs de naviguer avec la carte virtuelle.

L’implémentation d’une carte API de *Google Maps* nous permettra d’ajouter plusieurs fonctionnalités comme restreindre cette même carte à certaines coordonnées, ajouter des points quelconque sur la carte virtuelle, géolocaliser les utilisateurs de l’application ou bien créer un système de navigation entre ces différents points.

2.1.2 Restriction de la carte au campus

Afin de restreindre les utilisateurs de l'application à une région spécifique, celle du campus de l'Université d'Ottawa, nous avons utilisé une fonction qui nous permet d'insérer des coordonnés précis. Dû au fait que le campus de l'Université n'est pas simplement constitué d'une région géométrique simple, il y a eu deux solutions spécifiques que nous avons considérées.

La première solution était de restreindre la carte par les dimensions exactes du campus (*voir Appendice 5.4)*. Ceci nous posait plusieurs défis, puisque le programme devient plus compliqué quand la forme désirée n'est pas simple. Il nous faudrait faire plus de recherche afin d'arriver à restreindre la carte sous forme d'un polygone précis. La deuxième solution, et celle qui a été choisie, était de restreindre la carte par un rectangle qui inclut toutes les régions du campus désirées (*voir Appendice 5.5)*. Cette solution posait seulement la simple complication qu'il y aurait trop d'espace moins important et hors campus inclus dans les bornes de la carte. Le groupe a donc décidé que ce problème n'avait aucun effet négatif sur le but de l'application et sur les spécifications désirées par le client.

Le but de restreindre la carte est de rendre notre application plus simple d'utilisation. Nous voulons que notre application soit le plus simple possible d'utilisation, alors il est évident que restreindre la carte à seulement le campus permet de se situer plus facilement. De plus, pour nos futures fonctionnalités, il sera important d'avoir déjà cerné la carte.

2.1.3 Points de service

Pour accentuer la fonction sécuritaire de notre application, nous avons ajouté les points de rassemblements de chaque bâtiment (*voir Annexe 3*) en cas d'évacuation ainsi que la pharmacie sur le campus. Pour l'instant, il serait important d'ajouter de simples points sur la carte de campus pour ne pas rendre la carte virtuelle surchargée d'informations pour les utilisateurs.

Nous avons trouvé un site de l'Université d'Ottawa qui indique le point de rassemblement de chaque bâtiment à l'extérieur. De plus, le site indique aussi des endroits de rassemblement, dans un cas de mauvaise température, à court terme et à long terme. Ces derniers sont alors intérieurs. Il suffit de trouver leurs coordonnées géographiques à l'aide de *Google Maps* et nous allons pouvoir les situer sur l'application. L'application affichera alors un symbole désignant un point de rassemblement (*voir Appendice 5.3*). Nous avons par contre seulement indiqué les points de rassemblement extérieurs pour l'instant.

Suite à ces points de rassemblement, nous avons rendu ces derniers interactifs avec la géolocalisation de l'utilisateur. C'est-à-dire que l'application trouvera un itinéraire vers le point de rassemblement choisi et en même temps pour éviter de la confusion chez l'utilisateur. Nous avons aussi associer la navigation entre le point actuel de l'utilisateur et ce point de rassemblement désiré. Selon notre client, il est aussi important d'ajouter une fonction d'urgence très accessible pour trouver le chemin le plus rapide à un de ces points.[4]

[4] Livrable G

2.1.4 Coordonnées des bâtiments

Afin de permettre plusieurs autres fonctionnalités, nous avons ajouté les coordonnées de quelques bâtiments. Comme mentionné dans la *Section 2.1.3*, il serait important d'ajouter de simples points sur la carte de campus afin de ne pas surcharger la carte de dessins et du même fait rendre la carte virtuelle moins claire aux yeux de l'utilisateur. De plus, un projet futur pourrait être de changer les points représentant les bâtiments par une démarcation par périmètre des bâtiments (*voir Appendice 5.2*).

Afin d'afficher les bâtiments du campus sur notre carte virtuelle du campus (*voir Appendice 5.7*), nous avons besoin des coordonnées exactes de ces bâtiments. Comme nous voulons nous concentrer sur les fonctionnalités de l'application, nous avons choisi deux bâtiments que nous avons ajoutés : le FSS et Simard.

Suite à l'implémentation de ces bâtiments dans l'application, nous avons rendu ces derniers interactifs avec la géolocalisation de l'utilisateur. C'est-à-dire que l'application trouvera un itinéraire vers le bâtiment désiré à partir de la localisation de l'utilisateur. Nous avons aussi associer la navigation entre le point actuel de l'utilisateur dans un bâtiment vers un point de rassemblement choisi. L'ajout de ces bâtiments est nécessaire pour introduire d'autres fonctions dans notre application comme les plans des étages intérieurs et le positionnement de l'utilisateur dans cette bâtisse.[5]

[5] Livrable G

2.1.5 Navigation extérieure

Afin de faciliter notre application, nous avons choisi d'utiliser les cartes API de *Google Maps*. Plusieurs fonctionnalités viennent en pair avec ce dernier, comme le système de navigation extérieur (*voir Appendice 5.8*). Comme les autres parties de notre application, nous allons commencer la navigation entre bâtiments à petite échelle afin de le perfectionner. L'algorithme va commencer de l'emplacement actuel de l'utilisateur et se rendre au point ou à l'édifice choisi.

L'algorithme de navigation fonctionne comme suit. L'utilisateur choisit sa destination a un certain bâtiment, des directions pour se rendre apparaîtront. Une fois arrivé près de l'édifice, l'utilisateur pourra sélectionner l'édifice sur l'application et un plan d'étages se fera voir (*voir Section 2.1.6*). Nous avons aussi implémenté l'algorithme pour se rendre aux différents points de rassemblement ainsi qu'à la pharmacie du campus. Comme la navigation extérieur est une des parties les plus importantes de notre application, nous avons passé beaucoup de temps sur sa création. Nous espérons qu'elle pourra facilement être apportée à plus grande échelle lorsqu'un autre groupe aura le temps.

Notre but est d'améliorer la sécurité sur le campus, c'est pourquoi la navigation extérieure est une nécessité afin que les utilisateurs puissent se rendre à leur destination ou, en cas d'urgence, au point de rassemblement le plus proche. La navigation doit alors être faite le plus rapidement possible.

2.1.6 Plans d'étages des bâtiments

Pour faciliter le déplacement des utilisateurs à l'intérieur des bâtiments, nous allons afficher le plan des étages sur l'application (*voir Appendice 5.9*). Comme nous voulons venir en aide au plus grand nombre de personnes possibles, nous devons ajouter plus de fonctionnalités que seulement les points de rassemblement.

Afin d'y arriver nous allons commencer avec une partie du campus et la perfectionner avant de s'étendre à de plus grands projets. Nous avons déjà les plans d'étage de deux bâtiments. Afin de progresser, nous allons devoir demander les plans des autres édifices. Jusqu'à maintenant, nous ne voulons pas faire de navigation dans les étages des bâtiments, car nous aurions besoin de matériel très coûteux pour y arriver. Nous nous sommes alors penchés vers une autre solution afin de guider les utilisateurs à l'intérieur des bâtiments (*voir Section 2.1.7*). L'utilisateur pourra donc ouvrir le plan d'étage et changer selon l'étage où il se situe en appuyant sur des flèches pointant vers le haut et le bas.

Suite à l'ajout du périmètre des bâtiments, nous souhaitons que l'utilisateur puisse appuyer sur ce même bâtiment et le plan d'étage affichera (*voir Appendice 5.9*). Il pourra ensuite monter ou descendre les étages simplement en pesant sur un bouton. L'implémentation d'un plan d'étages est nécessaire afin d'ajouter la fonction de la reconnaissance de la position de l'utilisateur. De plus, il aura l'option de quitter cette page lorsqu'il sera sorti du bâtiment.[6]

[6] Livrable G

2.1.7 Reconnaissance de la position à l'intérieur des bâtiments

Afin de compléter les plans d'étages à l'intérieur des bâtiments, nous avons décidé d'ajouter une fonction qui permet d'afficher la position actuelle de l'utilisateur à l'intérieur du bâtiment où il se situe (*voir Appendice 5.10*). Nous croyons que cette fonction leur permettra de se situer plus facilement sur la carte virtuelle.

Pour le faire, nous avons ajouté une barre de navigation sur la même page que les plans d'étages. Ensuite, nous allons créer des panneaux indiquant un certain code qui sera affiché à plusieurs endroits sur les murs à l'intérieur des différents bâtiments. L'utilisateur pourra entrer ce code dans la barre de recherche de l'application et un point apparaîtra indiquant sa position. Cette fonction est semblable à celle des cartes de localisation utilisées dans les centres d'achats. Une fois que l'utilisateur se déplace, il pourra rentrer un autre code dans l'application et sa position sur le plan d'étages sera mise à jour. De cette manière, l'utilisateur pourra avoir des indices pour savoir s'il se déplace dans la bonne direction.

Nous croyons que cette fonctionnalité est importante pour la navigation intérieur et aidera les utilisateurs, par contre, elle comporte toujours certains défauts. Il serait plus facile de naviguer à l'intérieur avec des directions ainsi qu'une localisation en temps réelle comme celle de notre navigation intérieure, mais cela apporte plusieurs difficultés. Comme il est impossible que la localisation par GPS détermine l'étage sur laquelle l'utilisateur se situe, il faudrait investir dans des émetteurs *Bluetooth* qui seraient disposés à plusieurs endroits sur chaque étage. Toutefois, cette option est beaucoup trop dispendieuse pour notre budget fourni.

2.2 Étapes de création de notre application

Cette section permet d'expliquer le procès utilisé pour implémenter les fonctions décrites dans la *section 2.1* du manuel d'utilisation. Ceci permet aux groupes de projet futurs intéressés dans la conception d'une application similaire à la nôtre de pouvoir se référencer au procédé vécu par notre groupe.

2.2.1 Implémentation du API de *Google Maps*

L'utilisation des fonctions offertes par *Google* devient très simple quand le projet inclut l'implémentation du *Google* API. Par contre, ceci requiert certaines étapes avant que *Google* ne permette l'utilisation de ses fonctions à un groupe de projet indépendant.

Pour débuter, nous avons eu besoin de créer un compte Google comme développeur de projet. Ceci nous permet ensuite de faire une demande directement à *Google* pour nous donner permission d'utiliser l'API de *Google Maps*. La réponse de *Google* peut prendre beaucoup de temps, donc il est important de demander la permission aussitôt que la décision a été prise. Dans notre cas, la permission d'utilisation nous a été envoyée sous quarante-huit heures. L'étape finale de pouvoir implémenter l'API est de démontrer sur l'application que les fonctions offertes par *Google Maps* ont été utilisées pour compléter le projet.

Pour y arriver, simplement inclure le code affiché (*voir Appendice 5.11*). Pour plus d'explications, *Google Maps* explique comment incorporer leur carte API à partir de leur site Web (*voir Bibliographie*).

2.2.2 Restriction de la carte au campus

En termes des fonctions offertes par Google, l'une des plus facile à utiliser est celle qui permet de restreindre les bornes de la carte. L'implémentation de cette fonction dans un projet requiert simplement trois étapes: déterminer les coordonnées des bornes désirées, implémenter les bornes dans le code, et implémenter du rembourrage comme désiré.

Dans le cas de notre équipe, il est devenu beaucoup plus simple d'implémenter les bornes carrées du campus de l'Université d'Ottawa (*voir Appendice 5.5*) plutôt que les bornes polygonales réelles (*voir Appendice 5.4*). Ceci a causé que la prochaine étape soit beaucoup plus simple. Maintenant que les coordonnées des coins de la région bornée sont déterminés, il faut maintenant les implémenter dans le code (*voir Appendice 5.12*) pour un exemple complet de cette fonction. Il faut définir deux coins avec de nouvelles latitudes et longitudes et ensuite inclure ces deux points dans la fonction builder.include pour générer le carrée qui forme la borne de la carte. Pour finaliser cette étape, il faut ensuite aller chercher la longueur et la largeur de l'appareil utilisé pour changer la grandeur de la carte montrée sur l'écran.

Pour finaliser l'implémentation de cette fonction, il est très important d'inclure une certaine valeur de rembourrage en-dehors des bornes implémentées. Ceci sert à éviter un certain degré de confusion pour certains utilisateurs qui ne peuvent pas se repérer quand ils sont très proches des bornes extérieures du campus. Il faut simplement donner une valeur de rembourrage désiré à une variable et de l'inclure dans les fonctions nécessaires.

2.2.3 Points de service et coordonnées des bâtiments

L'implémentation de marqueurs pour les points de service et les bâtiments requiert un très petit nombre d'étapes pour compléter. La seule raison que cette étape pourrait prendre longtemps est due à la façon dont le groupe a décidé d'implémenter ces points (*voir Appendice 5.13*). On a eu besoin d'ajouter manuellement chaque point voulu pour qu'ils soient présents dans l'application.

Les étapes pour ajouter un point unique sur l'application sont relevés à êtres très simples. Il faut premièrement déterminer un emplacement spécifique en coordonnées comme latitude et longitude. Ceci nous permet ensuite d'ajouter cette valeur à la fonction de Google *googleMap.addMarker()* en utilisant la définition *.position()*. Il est également important de donner un nom au marqueur qui apparaît dans l'application avec la définition *.title()*. L'étape finale est d'implémenter la ligne de code qui permettrait à l'application de changer la position de la caméra à ce marqueur. Ceci requiert l'utilisation de la fonction *googleMaps.moveCamera()*. Pendant que ceci semble long pour compléter l'implémentation de plusieurs points de service, cette méthode semble être la méthode la plus efficace s'il n'y a pas trop de marqueurs qui doivent êtres ajoutés.

À partir des points des bâtiments, nous allons les rendre interactifs en ajoutant une fonctionnalité (*voir Section 2.2.5*) qui permet d'ouvrir un plan d'étage de l'intérieur des bâtiments.

2.2.4 Navigation extérieure

Pour arriver à naviguer sur le campus, nous avons dû importer des fonctions de *Google Maps* et des fichiers *Json*. L'application offrira aux utilisateurs l'option de choisir leur destination en plus de déterminer leur position initiale.

Comme vous pouvez l'observer dans le code (*voir Appendice 5.15*), on appelle les coordonnées initiales et ceux de la destination sous forme de longitude et latitude. Ensuite, l'application importe un *URL* qui facilite la création de la navigation. L'utilisateur aura l'option d'entrer le nom du bâtiment dans la barre de recherche. Des directions apparaîtront pour se diriger vers la destination choisie. Présentement, nous tentons d'ajouter les points de rassemblement comme des "bâtiments" afin que les utilisateurs puissent aussi écrire le point de rassemblement dans la barre de recherche, au lieu d'appuyer sur le marqueur sur la carte virtuelle.

À partir de la navigation extérieure, notre application prend maintenant vie. Après avoir créé cette fonctionnalité, nous allons pouvoir passer aux petits détails qui feront une grande différence pour assurer la sécurité et la navigation. Nous allons ajouter les plans des étages intérieurs (*voir Section 2.2.5*) ainsi qu'un semblant de navigation intérieur (*voir Section 2.2.6*).

2.2.5 Plans d'étages des bâtiments

Afin d'aider les utilisateurs à naviguer à l'intérieur des bâtiments, nous avons incorporé une fonction qui permet d'afficher le plan des étages intérieurs du bâtiment choisi. Nous avons déterminé que cette fonctionnalité est un aspect fondamental au bon fonctionnement de l'application.

Le code pour réussir à ajouter des plans d'étages des bâtiments (*voir Appendice 5.15*) est assez simple. Comme c'est déjà une fonction de *Java*, nous avons seulement suivi le gabarit. La fonction *onClick* permet de déterminer si l'utilisateur appuie sur l'écran et si l'endroit correspond à une location prédéterminée comme un bâtiment, l'application ouvrira alors une image qui est enregistrée sur un de nos ordinateurs pour l'instant. Nous avons les plans intérieurs de deux bâtiments jusqu'à présent : celui du bâtiment FSS et celui du bâtiment Simard.

Cette fonctionnalité débloque plusieurs futurs possibles ajouts chez de nouvelles équipes, comme une navigation intérieure (*voir Section 2.1.7*). De plus, les futurs étudiants qui développeront notre projet pourront modifier les plans afin d'ajouter des points de services importants dans le bâtiment comme les ascenseurs ou l'administration.

2.3 Fonctionnement de l’application

Dans cette section du manuel, nous allons expliquer le fonctionnement de l’application du point de vue de l’utilisateur. Comme l’application ne sera pas publiée sur le magasin électronique *Google Play*, les étapes de fonctionnement sont basées sur notre émulateur de *Android Studio*.

Une fois l’application installée sur votre téléphone intelligent, lancez l’application. Une carte restreinte au campus de l’Université d’Ottawa devrait apparaître ainsi qu’un marqueur montrant votre position initiale. Deux options vous seront offertes, celle de naviguer vers un bâtiment ou celle de naviguer vers un point de rassemblement.

Afin de naviguer vers un point de rassemblement ou d’un bâtiment particulier, il suffit d’appuyer sur leur marqueur. Des directions vers la destination choisie apparaîtront (*voir Appendice 5.8*) à partir de la position initiale de l’utilisateur. Les marqueurs de la carte sont interactifs avec les utilisateurs.

Une fois que les utilisateurs sont près du bâtiment déterminé, ils pourront sélectionner un plan intérieur de ce bâtiment (*voir Appendice 5.9*). À l’intérieur du bâtiment, vous trouverez des codes affichés au mur que vous pourrez entrer dans la barre de navigation de l’application. Un point apparaîtra sur les plans d’étages afin de vous aider à vous situer dans l’édifice.

Pour plus d’informations ou pour signaler des possibles modifications à faire, nous vous recommandons de nous contacter par courriel (*uo.directions@gmail.com*). Nous allons répondre dans le plus bref délais avec plus d'informations.

2.4 Instructions pour l'entretien de l'application

Pour ce qui en est de l'entretien, nous croyons que nous avons codé l'application de manière facilement compréhensible. La majorité de l'entretien nécessaire suivant les fonctions implémenter seront l'ajout de coordonnées ou la correction de certaines coordonnées dans le cas où ils sont fautifs. La prochaine équipe qui voudra développer ou améliorer notre application pourra alors remédier facilement certains de nos problèmes.

Afin d'assurer des rétroactions de nos clients, nous avons créé un courriel qui sera affiché dans l'application (*uo.directions@gmail.com*). Les utilisateurs auront la possibilité de nous envoyer leurs conseils ou difficultés avec notre application. Il est important d'avoir cette particularité afin d'avoir une autre perspective sur les modifications à faire. Bien sûr, l'équipe qui reprendra aura accès au courriel.

Pour modifier l'espace de restriction afin d'ajouter un campus qui ne se situe pas dans l'espace rectangulaire (*voir Appendice 5.5*), il faut se diriger à la section du code indiqué (*voir Appendice 5.12*). Pour changer les coordonnées des bordures de la carte, vous devez remplacer les nombres écrits dans la parenthèse. Pour aider à trouver les coordonnées désirées, il suffit d'aller voir sur *Google Maps*. La modification de cette section sera importante si vous voulez ajouter le campus Roger Guindon, 99 Bank ou Lees. De plus, si certains bâtiments du campus principal ne sont pas inclus, il faudra réduire la zone en changeant les coordonnées dans le code (*voir Appendice 5.14*).

Pour ajouter ou modifier les coordonnées des points de rassemblement ou des bâtiments sur la carte virtuelle, il faut se diriger vers la section du code indiqué (*voir Appendice 5.13*). Pour ajouter un point, il suffit de copier le code en modifiant les

données. Vous aurez besoin de changer le nom de la variable pour le nom que vous désirez et de remplacer les coordonnées du nouveau point de rassemblement. N'oubliez pas de changer toutes les variables dans cette partie du code afin que l'application puisse bien afficher le nouveau point de service. Pour modifier le point de service, en cas de manque de précision au niveau des coordonnées, il suffit seulement de changer les nombres en parenthèses. Le premier nombre signifie la latitude et le deuxième la longitude.

Pour finir, si vous n'avez pas trouvé exactement ce que vous cherchiez dans cette section, le code inclura, en commentaire, une courte description de ce que la fonction fait. De plus, la meilleure manière de se tenir au courant des erreurs présentes dans l'application est de prendre les rétroactions des utilisateurs par courriel.

2.5 Directives et précautions liées à la santé et sécurité

Dès le départ du développement de notre produit, la santé et sécurité a été un des facteurs les plus importants étant donné que le client en a mis beaucoup d'emphase. Le but de notre application est d'améliorer la santé et la sécurité sur le campus de l'*Université d'Ottawa* tout en permettant à l'utilisateur de naviguer le campus de façon plus efficace. Non seulement que l'utilisateur peut se rendre à une classe plus rapidement, il peut aussi suivre un trajet vers un point d'intérêt en cas d'urgence. De plus, avoir une application mobile en cas d'urgence peut être avantageux puisque l'utilisateur peut se rendre où il veut en suivant le trajet le plus rapide.

Pendant l'utilisation de l'application, il est important de prendre les précautions nécessaires pour rester en sécurité. Un appareil mobile peut être une distraction et met donc la sécurité personnelle de l'utilisateur en péril. Il est nécessaire que l'utilisateur doit être conscient de son environnement avant, afin d'éviter un accident ou une blessure liée à une distraction. Il serait important d'aviser les utilisateurs de ne pas entièrement se fier sur notre application. Les points de services sur le campus (*voir Annexe 3*) devraient déjà être connus des utilisateurs. Nous ne voulons pas rendre le personnel et les étudiants dépendant de l'application pour situer les points de rassemblement.

2.6 Instructions techniques pour dépannage de l'application

Dans cette section, il s’agira de nos conseils pour le dépannage de l’application en cas de défaillance. Malheureusement, il y a très peu d’options disponibles afin de remédier au problème, nous allons tout de même en proposer quelques-unes.

Comme la majorité des problèmes sont reliés au code de l’application, si vous avez ajouter de nouvelles fonctionnalités et que ces dernières créent un problème, il est recommandé que vous retourniez à une ancienne version stable. Avec cette version, veuillez tranquillement implémenter les fonctions une par une afin de bien cerner le problème.

Un autre moyen de résoudre d’éventuels problèmes serait de consulter notre bibliographie. Les sites que nous avons utilisés pour créer l’application sont tous présentés dans cette section. Si le problème ne peut pas être résolu par les sites offerts dans la section bibliographie, nous vous recommandons de consulter le site de *Google Maps Platform* ou bien d’autres sites ou forum semblables.

3 Conclusion

Pendant le développement de ce prototype, notre équipe a su surmonter plusieurs difficultés. Nous en avons appris beaucoup à propos de la programmation en Java ainsi que la manipulation des cartes API de *Google Maps*. Seulement un membre de notre équipe avait déjà de l'expérience en programmation avec Java, alors cela nous a permis d'expérimenter de découvrir ou d'approfondir un nouveau domaine.

Afin de bien comprendre les demandes de notre client, nous avons dû se mettre dans ses souliers. Ce développement de prototype nous a permis d'empathiser avec notre client, ce qui sera une compétence très utile plus tard.

Durant la première moitié de la session, nous avons pris beaucoup de retard sur le prototype. Nous avons dû organiser les étapes de conception de notre projet plus vigoureusement afin de nous rattraper. Nous nous sommes pris deux semaines d'avance et nous avons pris plus au sérieux notre plan de projet. Nous souhaitons transposer cette organisation dans nos projets d'équipes futures.

Lors de la conception du prototype, nous avons remarqué que certains avaient plus d'expérience dans certains domaines. Nous avons donc ajusté les tâches de chaque membre selon leurs forces et faiblesses. Cette qualité est importante pour le bon fonctionnement de l'équipe.

En conclusion, nous avons beaucoup appris sur le développement d'un produit passant par la décomposition fonctionnelle (*voir Annexe 1*), vers le modèle d'affaires d'un produit (*voir Annexe 3*) finissant par le marketing et les ventes. Dans ce livrable, vous avez maintenant tout l'information nécessaire afin de comprendre et d'améliorer notre prototype. La cause de ce projet est importante, nous espérons que la cause vous

tiendra autant à cœur que nous. Les équipes intéressées auront certaines de nos recommandations afin d’améliorer le prototype dans les pages suivantes.

Recommandations

Suivant la création de notre prototype, malgré tous nos efforts, l'application ne comporte toujours pas la totalité des fonctionnalités que nous aurions espéré. Voici quelques recommandations que la prochaine équipe pourra ajouter afin de rendre l'application plus complète en plus d'améliorer son accessibilité. Nous les avons classés en ordre selon leur facilité de leur addition ou de l'importance de la fonctionnalité afin d'améliorer l'application.

Premièrement, l'ajout des emplacements des ascenseurs sur les plans intérieurs serait une addition simple et importante. En ajoutant cette fonctionnalité, la navigation avec l'application deviendrait plus facile pour les personnes souffrant d'un handicap. De plus, le client souhaite que la navigation soit accessible pour tout le monde, incluant les personnes ayant des besoins spéciaux. En simplement ajoutant un icône d'un ascenseur ne serait pas dérangeant.

Deuxièmement, il serait important d'ajouter une fonction d'urgence au bas de l'écran afin d'indiquer le trajet pour se rendre au point de rassemblement le plus proche. Jusqu'à maintenant, nous avons seulement implémenter une navigation si l'utilisateur appuie sur un des points de rassemblement. Cette fonctionnalité sera plus complexe, mais aidera grandement à l'aspect sécurité de l'application, ce qui est grandement encouragé par notre client.

Troisièmement, l'amélioration de la navigation intérieur pourrait aider à la navigation des utilisateurs. Il est difficile d'ajouter une telle fonctionnalité car les GPS de *Google Maps* n'ont pas de précision pour déterminer l'étage où l'utilisateur se situe. Il

faudra alors utiliser une application externe ou bien créer un algorithme en lien avec les codes qui seront affichés sur les murs (*voir Section 2.1.7*).

Quatrièmement, avant la publication de l'application, il faudrait étendre les fonctionnalités de l'application à tout le campus. C'est-à-dire ajouter tous les points de services, les bâtiments, les cartes intérieurs ainsi que les fonctions offertes à l'intérieur des bâtiments. Comme nous voulions tester le fonctionnement de ces dernières, nous les avons seulement conçu sur une plus petite échelle.

Cinquièmement, l'ajout d'effets auditifs serait important afin d'aider les personnes malvoyantes à naviguer sur le campus. Comme dit plus haut, le client veut que l'application soit accessible à toutes personnes ayant des besoins spéciaux. Cette fonctionnalité sera par contre plus difficile.

4 Bibliographie

Google Maps Platform. (2020). Get an API Key. Tiré de https://developers.google.com/maps/documentation/android-sdk/get-api-key

Google Maps Platform. (2020). Camera and View. Tiré de https://developers.google.com/maps/documentation/android-sdk/views#maps_android_camera_and_view_common_map_movements-java

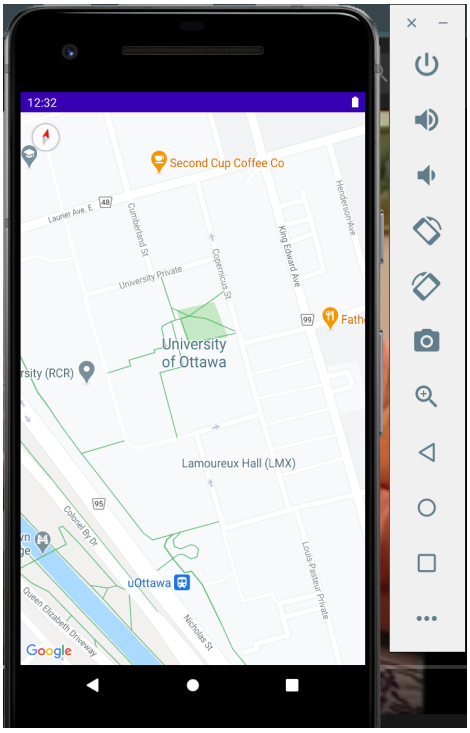
Google Maps Platform. (2020). Markers. Tiré de https://developers.google.com/maps/documentation/android-sdk/marker

Stack Overflow. (2014). Show marker details with image onclick marker OpenStreetMap. Tiré de https://stackoverflow.com/questions/23108709/show-marker-details-with-image-onclick-marker-openstreetmap

Google Maps Platform. (2020). Google Maps Platform Documentation. Tiré de https://developers.google.com/maps/documentation

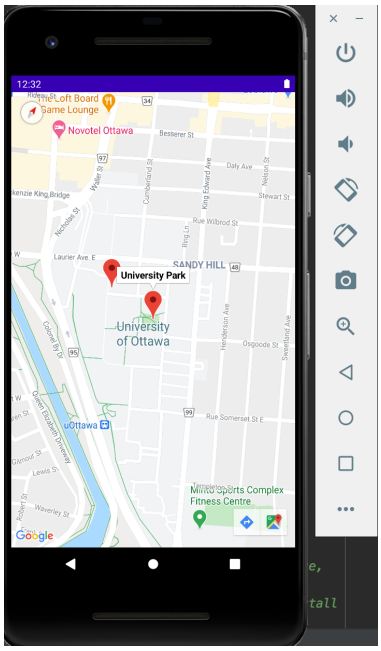
5 Appendices

5.1 Carte API de *Google Maps*

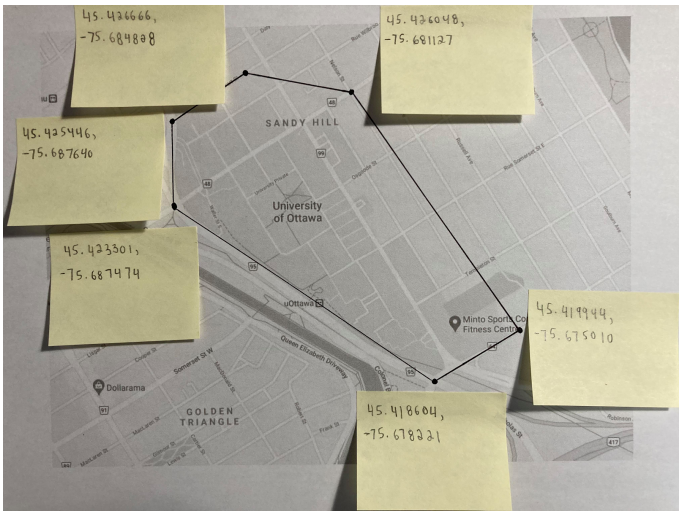


5.2 Démarcation du périmètre d’un bâtiment

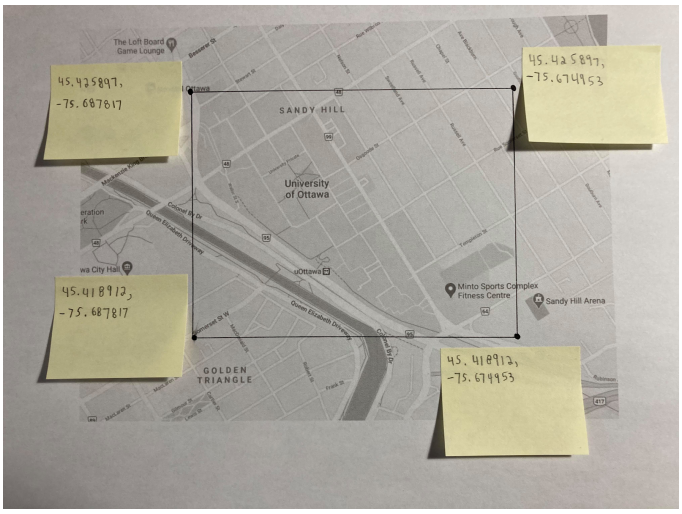
5.3 Points de services



5.4 Croquis de restriction de la carte de manière polygonale

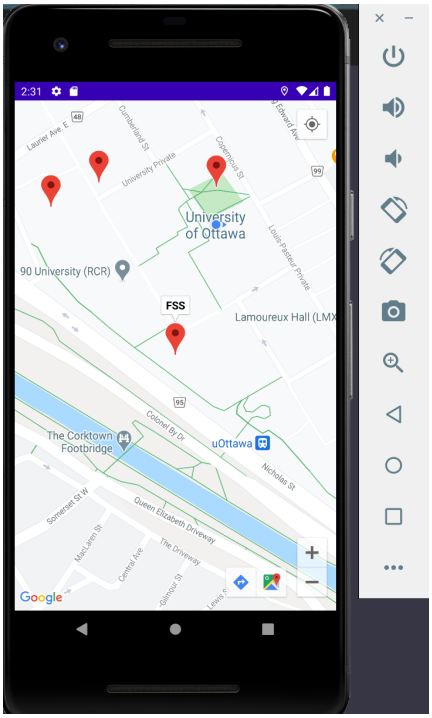


5.5 Croquis de restriction de la carte de manière rectangulaire



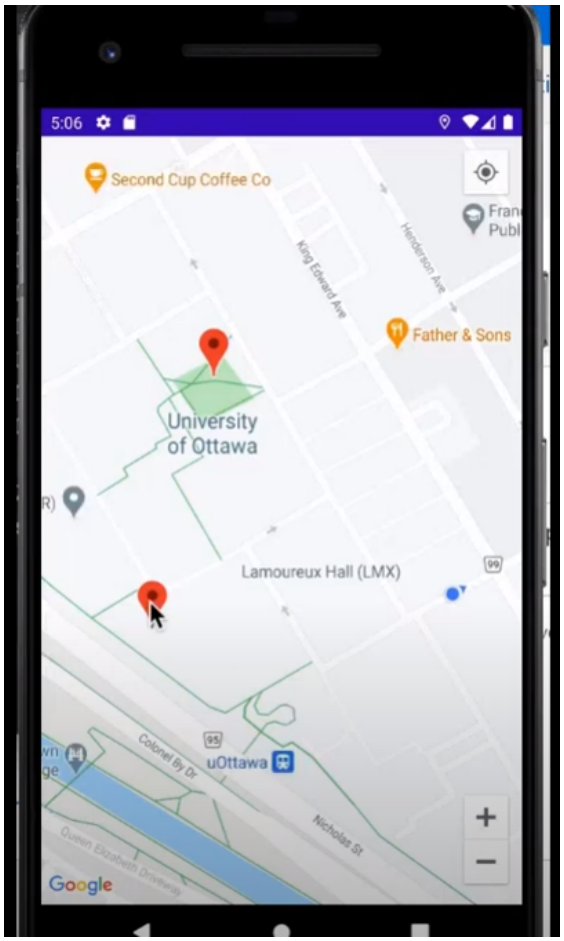
5.6 Croquis des points de services

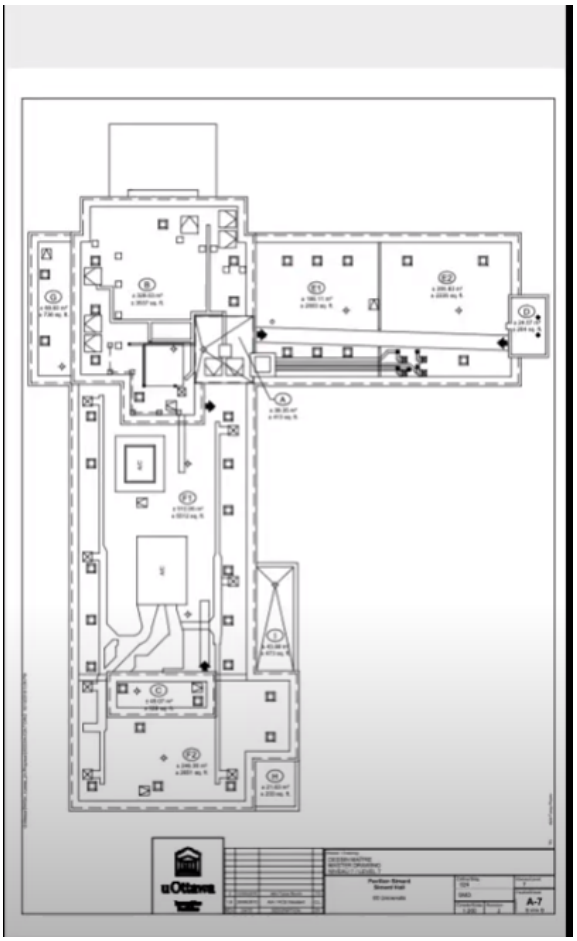
5.7 Coordonnées des bâtiments



5.8 Navigation extérieure

5.9 Plans des étages des bâtiments





5.10 Code Java de l'implémentation du API de *Google Maps*

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_maps);
    // Obtain the SupportMapFragment and get notified when the map is ready to be used.
    SupportMapFragment mapFragment = (SupportMapFragment) getSupportFragmentManager()
            .findFragmentById(R.id.map);
    mapFragment.getMapAsync(this);
```

5.11 Code Java pour restreindre la carte au campus

```
// This method restricts the area to lock in with the university area
    // Get latlong for corners
    LatLng one = new LatLng(45.425163, -75.687664);
    LatLng two = new LatLng(45.419748, -75.675492);

    LatLngBounds.Builder builder = new LatLngBounds.Builder();

    // Add to builder
    builder.include(one);
    builder.include(two);

    LatLngBounds bounds = builder.build();

    // Get width and height to current display screen
    int width = getResources().getDisplayMetrics().widthPixels;
    int height = getResources().getDisplayMetrics().heightPixels;

    // 20% padding
    int padding = (int) (width * 0.20);

    // Set latlong bounds
    mMap.setLatLngBoundsForCameraTarget(bounds);

    // Move camera to fill the bound to screen
    mMap.moveCamera(CameraUpdateFactory.newLatLngBounds(bounds, width, height, padding));
```

5.12 Code Java pour afficher un point de service

```
// affiche un marker sur Front of Perez
LatLng front_of_perez = new LatLng(45.423566, -75.684983);
googleMap.addMarker(new MarkerOptions()
        .position(front_of_perez)
        .title("Front of Perez"));
googleMap.moveCamera(CameraUpdateFactory.newLatLng(front_of_perez));


// affiche un marker sur University Park
LatLng university_park = new LatLng(45.423512, -75.683127);
googleMap.addMarker(new MarkerOptions()
        .position(university_park)
        .title("University Park"));
googleMap.moveCamera(CameraUpdateFactory.newLatLng(university_park));
```

5.13 Code Java pour afficher les bâtiments du campus

```
    // Get a marker on FSS
    LatLng fss = new LatLng(45.421640, -75.683774);
    googleMap.addMarker(new MarkerOptions()
            .position(fss)
            .title("FSS"));
    googleMap.moveCamera(CameraUpdateFactory.newLatLng(fss));

    // Get a marker on Simard
    LatLng simard = new LatLng(45.423294, -75.685736);
    googleMap.addMarker(new MarkerOptions()
            .position(simard)
            .title("Simard"));
    googleMap.moveCamera(CameraUpdateFactory.newLatLng(fss));
```

5.14 Code Java pour la navigation extérieure

```
private String getDirectionsUrl(LatLng origin, LatLng dest) {

    // Origin of route
    String str_origin = "origin=" + origin.latitude + "," + origin.longitude;

    // Destination of route
    String str_dest = "destination=" + dest.latitude + "," + dest.longitude;

    // Sensor enabled
    String sensor = "sensor=false";
    String mode = "mode=driving";

    // Building the parameters to the web service
    String parameters = str_origin + "&" + str_dest + "&" + sensor + "&" + mode;

    // Output format
    String output = "json";

    // Building the url to the web service
    String url = "https://maps.googleapis.com/maps/api/directions/" + output + "?" + par


    return url;
}
```

```
    private String downloadUrl(String strUrl) throws IOException {
        String data = "";
        InputStream iStream = null;
        HttpURLConnection urlConnection = null;
        try {
            URL url = new URL (strUrl);

            urlConnection = (HttpURLConnection) url.openConnection();

            urlConnection.connect();

            iStream = urlConnection.getInputStream();

            BufferedReader br = new BufferedReader(new InputStreamReader (iStream));

            StringBuffer sb = new StringBuffer();

            String line = "";
            while ((line = br.readLine()) != null) {
                sb.append(line);
            }

            data = sb.toString();

            br.close();

        } catch (Exception e) {
            Log.d("Exception", e.toString());
        } finally {
            iStream.close();
            urlConnection.disconnect();
        }
        return data;
    }
}
```

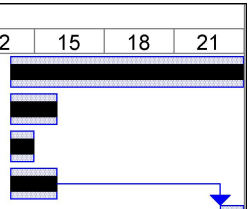
5.15 Code Java pour afficher les plans d'étages

```
}
public class SearchData extends Activity {
    ImageButton search;

    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_search_data);
        search =(ImageButton)findViewById(R.id.search);
        search.setOnClickListener(new View.OnClickListener()   {
            public void onClick(View v)  {
                try {
                    ConnectService();
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }
```

5.16 Plan de projet

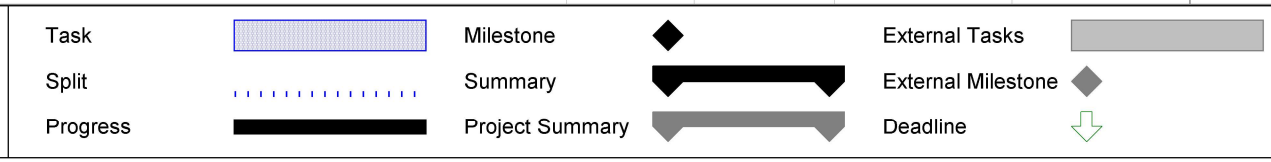
ID	Task Name	Duration	Person	Start	Finish
1	**1. Livrable A**	**8 days**	**Tous**	**Mon 20-09-14**	**Wed 20-09-23**
2	1.1 Contrat d'équipe	2 days	Tous	Mon 20-09-14	Tue 20-09-15
3	1.1.1 Plan de communication	1 day	Tous	Mon 20-09-14	Mon 20-09-14
4	1.2 Préparation Rencontre Client 1	2 days	Tous	Mon 20-09-14	Tue 20-09-15
5	1.3 Rencontre Client 1	1 day	Tous	Wed 20-09-23	Wed 20-09-23
6	1.3.1 Poser les questions	1 day	Mathieu	Thu 20-09-24	Thu 20-09-24
7	1.3.2 Noter les réponses du client	1 day	Hugo	Thu 20-09-24	Thu 20-09-24
8	**2. Livrable B**	**3 days**	**Tous**	**Wed 20-09-23**	**Sat 20-09-26**
9	2.1 Identifications besoins et métriques	2 days	Hugo	Wed 20-09-23	Thu 20-09-24
10	2.2 Étalonnage et spécifications	1 day	Mathieu	Fri 20-09-25	Fri 20-09-25
11	2.3 Plan de projet B	3 days	Hugo	Wed 20-09-23	Fri 20-09-25
12	**3. Livrable C**	**5 days**	**Tous**	**Sun 20-09-27**	**Thu 20-10-01**
13	3.1 Conception préliminaire	3 days	Tous	Sun 20-09-27	Tue 20-09-29
14	3.1.1 Décomposition fonctionnelle	1 day	Mathieu	Sun 20-09-27	Sun 20-09-27
15	3.1.2 Génération des concepts	1 day	Tous	Sun 20-09-27	Sun 20-09-27
16	3.1.3 Évaluation des concepts	1 day	Tous	Mon 20-09-28	Mon 20-09-28
17	3.1.4 Développement de concepts	1 day	Hugo	Mon 20-09-28	Mon 20-09-28
18	3.1.5 Améliorer ce concept	1 day	Hugo	Tue 20-09-29	Tue 20-09-29
19	3.1.6 Représenter ce concept	1 day	Hugo	Tue 20-09-29	Tue 20-09-29
20	3.1.7 Comparaison avec spécifications	1 day	Tous	Tue 20-09-29	Tue 20-09-29
21	3.2 Étude de faisabilité	2 days	Simon	Wed 20-09-30	Thu 20-10-01
22	3.3 Plan de projet C	1 day	Tous	Thu 20-10-01	Thu 20-10-01
23	3.4 Rencontre Client 2	1 day	Tous	Mon 20-10-05	Mon 20-10-05
24	3.4.1 Poser les questions	1 day	Mathieu	Tue 20-10-06	Tue 20-10-06
25	3.4.2 Noter les réponses du client	1 day	Hugo	Tue 20-10-06	Tue 20-10-06
26	**4. Livrable D**	**4 days**	**Tous**	**Mon 20-10-05**	**Thu 20-10-08**
27	4.1 Rétroaction du client	1 day	Hugo	Mon 20-10-05	Mon 20-10-05
28	4.2 Mis à jour de notre concept détaillé	1 day	Hugo	Tue 20-10-06	Tue 20-10-06
29	4.3 Hypothèses de produit	1 day	Hedi	Tue 20-10-06	Tue 20-10-06
30	4.4 Validation des hypothèses	1 day	Mathieu	Tue 20-10-06	Tue 20-10-06
31	4.5 Documentation de notre prototype	1 day	Hedi	Tue 20-10-06	Tue 20-10-06
32	4.6 Analyse de notre prototype par rapport à nos spécifications cibles	1 day	Hugo	Wed 20-10-07	Wed 20-10-07



Project: Squelette de plan GNG
Date: Thu 20-12-03

Task
Split
Progress
Milestone
Summary
Project Summary
External Tasks
External Milestone
Deadline

ID	Task Name	Duration	Person	Start	Finish
33	4.7 Informations sur notre prochaine rencontre	1 day	Mathieu	Thu 20-10-08	Thu 20-10-08
34	4.8 Nomenclature des matériaux et composantes	1 day	Hugo	Thu 20-10-08	Thu 20-10-08
35	4.9 Plan de projet D	4 days	Hugo	Mon 20-10-05	Thu 20-10-08
36	**5. Livrable E**	**5 days**	**Tous**	**Fri 20-10-09**	**Thu 20-10-15**
37	5.1 Présentation sur le progrès de notre projet	1 day	Tous	Mon 20-10-19	Mon 20-10-19
38	5.2 Évaluation des pairs	1 day	Tous	Mon 20-10-19	Mon 20-10-19
39	**6. Livrable F**	**5 days**	**Tous**	**Sat 20-10-17**	**Thu 20-10-22**
40	6.1 Type de modèle d'affaire	1 day	Tous	Mon 20-10-19	Mon 20-10-19
41	6.2 Tableau de modèle d'affaire	1 day	Mathieu	Tue 20-10-20	Tue 20-10-20
42	6.3 Hypothèses de base et faisabilité	2 days	Simon	Wed 20-10-21	Thu 20-10-22
43	6.4 Plan de projet F	3 days	Hugo	Mon 20-10-19	Wed 20-10-21
44	6.5 Rencontre Client 3	1 day	Tous	Mon 20-11-02	Mon 20-11-02
45	**7. Livrable G**	**9 days**	**Tous**	**Sat 20-10-24**	**Wed 20-11-04**
46	7.1 Rétroaction du client	1 day	Hugo	Tue 20-11-03	Tue 20-11-03
47	7.2 Prototype 2	12 days	Hedi	Mon 20-10-19	Mon 20-11-02
48	7.3 Documentation de notre prototype	1 day	Mathieu	Tue 20-11-03	Tue 20-11-03
49	7.4 Analyse de notre prototype par rapport à nos spécifications cibles	1 day	Simon	Wed 20-11-04	Wed 20-11-04
50	**8. Livrable H**	**11 days**	**Tous**	**Mon 20-11-09**	**Thu 20-11-19**
51	8.1 Rapport d'économie	7 days	Tous	Fri 20-11-06	Sat 20-11-14
52	8.1.1 Liste des différents coûts	2 days	Hugo	Tue 20-11-10	Wed 20-11-11
53	8.1.2 Compte de profits sur 3 ans	2 days	Hugo	Thu 20-11-12	Fri 20-11-13
54	8.1.3 Seuil de rentabilité	1 day	Hugo	Sat 20-11-14	Sat 20-11-14
55	8.1.4 Hypothèses	1 day	Mathieu	Sun 20-11-15	Sun 20-11-15
56	8.2 Vidéo argumentaire	5 days	Tous	Sun 20-11-15	Thu 20-11-19
57	8.2.1 Introduction	3 days	Hugo	Sun 20-11-15	Tue 20-11-17
58	8.2.2 Problèmatique	4 days	Mathieu	Sun 20-11-15	Wed 20-11-18
59	8.2.3 Solution proposée	3 days	Hugo	Tue 20-11-17	Thu 20-11-19
60	8.3 Plan de projet H	3 days	Hugo	Tue 20-11-17	Thu 20-11-19
61	**9. Livrable I**	**10 days**	**Tous**	**Fri 20-11-20**	**Thu 20-12-03**
62	9.1 Présentation de notre projet	1 day	Tous	Thu 20-12-03	Thu 20-12-03
63	9.1.1 Motiver le problème	1 day	Tous	Fri 20-11-20	Sun 20-11-22
64	9.1.2 Exigences, solutions et alternatives	2 days	Tous	Mon 20-11-23	Tue 20-11-24



Project: Squelette de plan GNG
Date: Thu 20-12-03

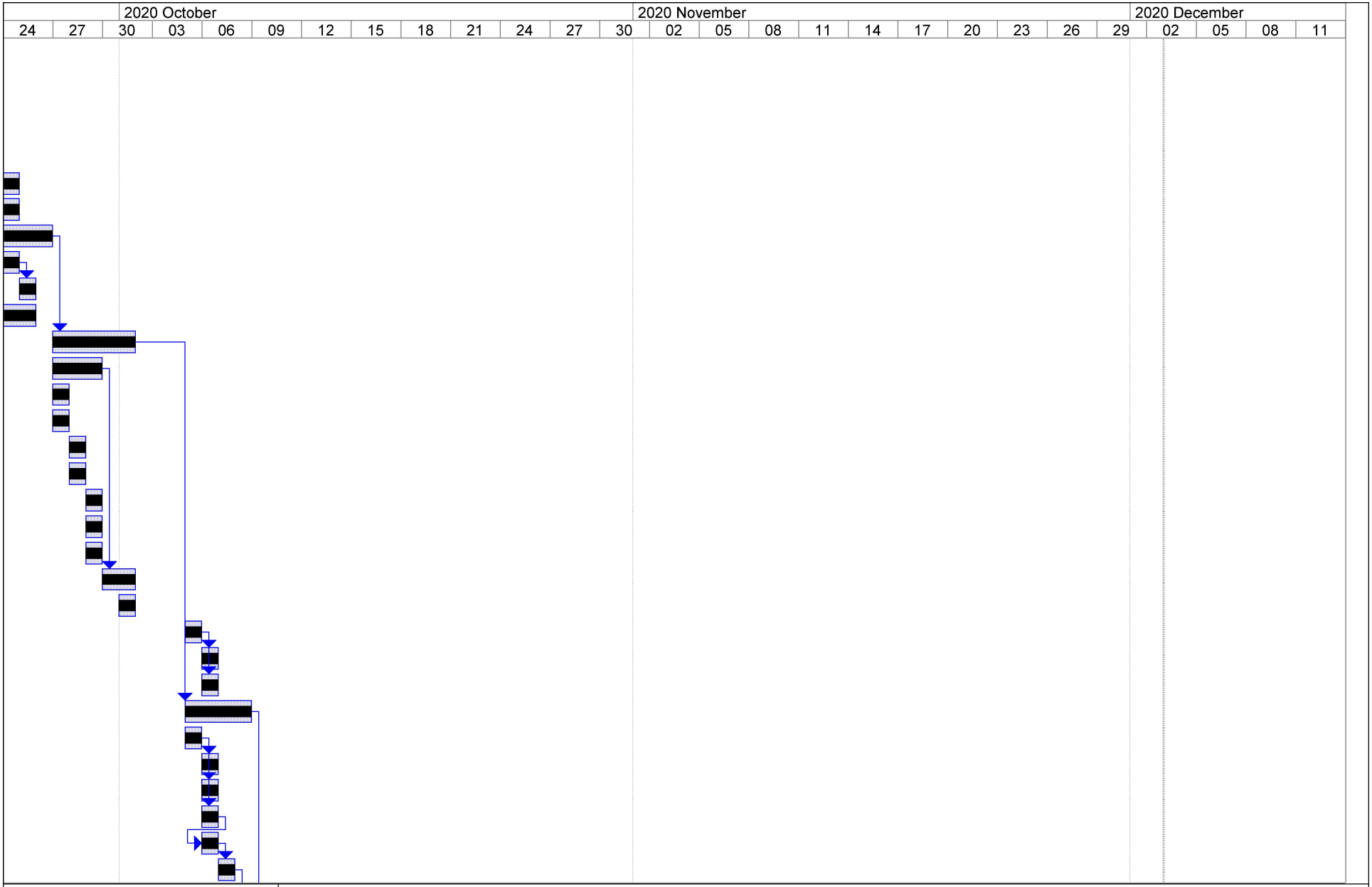
Task
Split
Progress
Milestone
Summary
Project Summary
External Tasks
External Milestone
Deadline

ID	Task Name	Duration	Person	Start	Finish
65	9.1.3 Démarquation de la concurrence	2 days	Tous	Wed 20-11-25	Thu 20-11-26
66	9.1.4 Produit en action	2 days	Tous	Fri 20-11-27	Mon 20-11-30
67	9.2 Répondre aux questions	1 day	Tous	Thu 20-12-03	Thu 20-12-03
68	9.3 Plan de projet I	1 day	Hugo	Thu 20-12-03	Thu 20-12-03
69	**10. Livrable J**	**10 days**	**Tous**	**Fri 20-11-20**	**Thu 20-12-03**
70	10.1 Page titre	1 day	Hugo	Fri 20-11-20	Sat 20-11-21
71	10.2 Abstrait	1 day	Hugo	Mon 20-11-23	Mon 20-11-23
72	10.3 Table des matière, liste des figures / tableaux	1 day	Hugo	Tue 20-11-24	Tue 20-11-24
73	10.4 Introduction	1 day	Mathieu	Wed 20-11-25	Wed 20-11-25
74	10.5 Corps principal	3 days	Hugo	Thu 20-11-26	Mon 20-11-30
75	10.6 Conclusions et recommandations	1 day	Mathieu	Tue 20-12-01	Tue 20-12-01
76	10.7 Bibliographies et appendices	1 day	Hugo	Wed 20-12-02	Wed 20-12-02
77	10.8 Plan de projet J	1 day	Hugo	Thu 20-12-03	Thu 20-12-03
78	**11. Livrable K**	**6 days**	**Tous**	**Fri 20-12-04**	**Thu 20-12-10**
79	11.1 Présentation finale	1 day	Tous	Thu 20-12-10	Thu 20-12-10
80	11.1.1 Résumé de nos livrables	1 day	Mathieu	Fri 20-12-04	Fri 20-12-04
81	11.1.2 Solutions	1 day	Hugo	Sat 20-12-05	Sat 20-12-05
82	11.1.3 Décisions prises	2 days	Hedi	Mon 20-12-07	Tue 20-12-08
83	11.1.4 Épreuves et difficultés	1 day	Simon	Wed 20-12-09	Wed 20-12-09
84	11.2 Plan de projet K	1 day	Hugo	Thu 20-12-10	Thu 20-12-10
85	**12. Livrable L**	**6 days**	**Tous**	**Fri 20-12-04**	**Thu 20-12-10**
86	12.1 Exploration de propriétés intellectuelles	1 day	Hugo	Fri 20-12-04	Fri 20-12-04
87	12.2 Relations entre produit et propriétés intellectuelles	2 days	Hugo	Sat 20-12-05	Mon 20-12-07
88	12.3 Importance des propriétés intellectuelles	2 days	Hugo	Tue 20-12-08	Wed 20-12-09
89	12.4 Manipulation des propriétés intellectuelles	1 day	Mathieu	Thu 20-12-10	Thu 20-12-10
90	12.4 Plan de projet L	1 day	Hugo	Thu 20-12-10	Thu 20-12-10

09 | 12 | 15 | 18 | 21

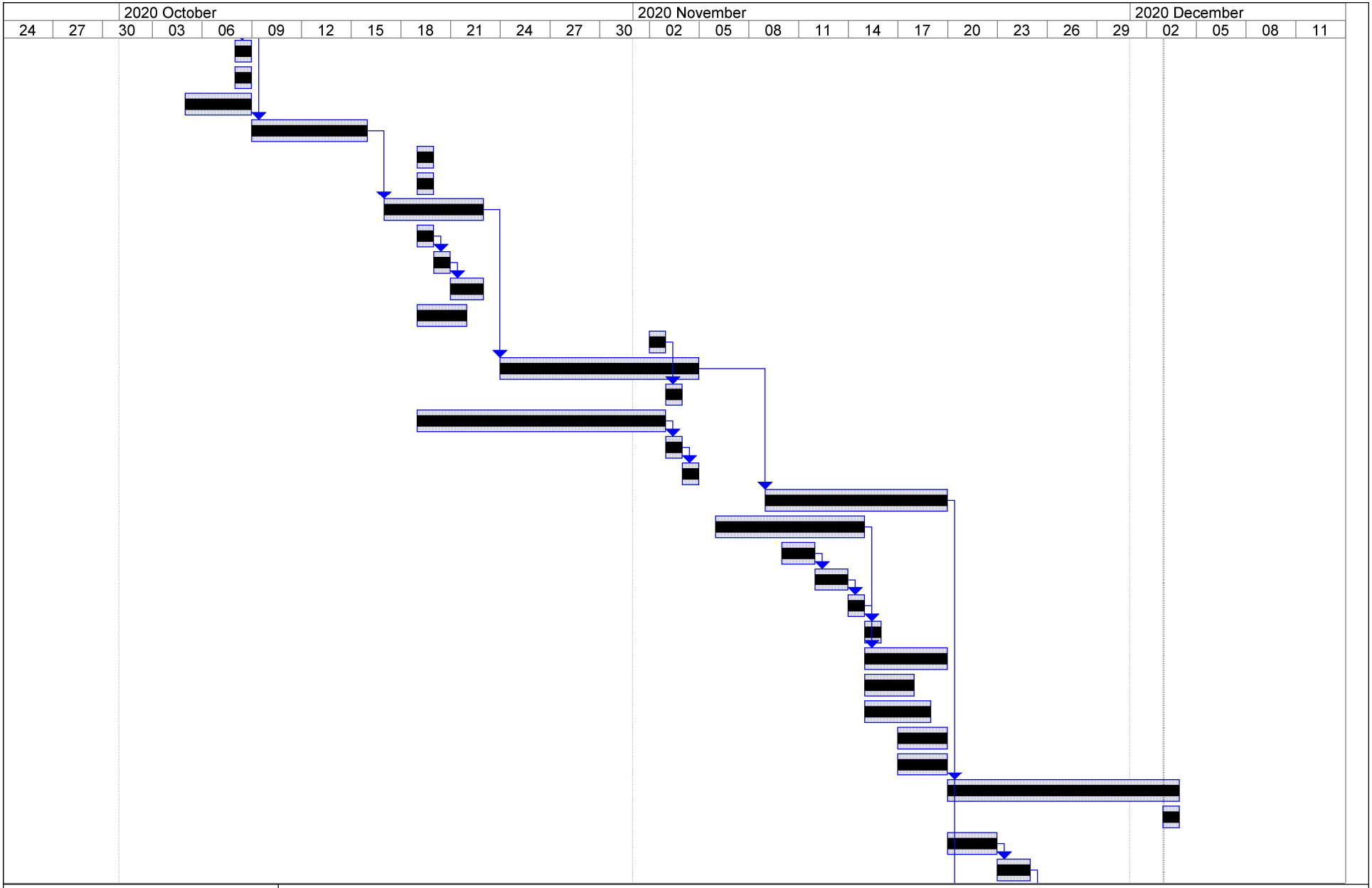
Project: Squelette de plan GNG
Date: Thu 20-12-03

Task | Split | Progress | Milestone | Summary | Project Summary | External Tasks | External Milestone | Deadline



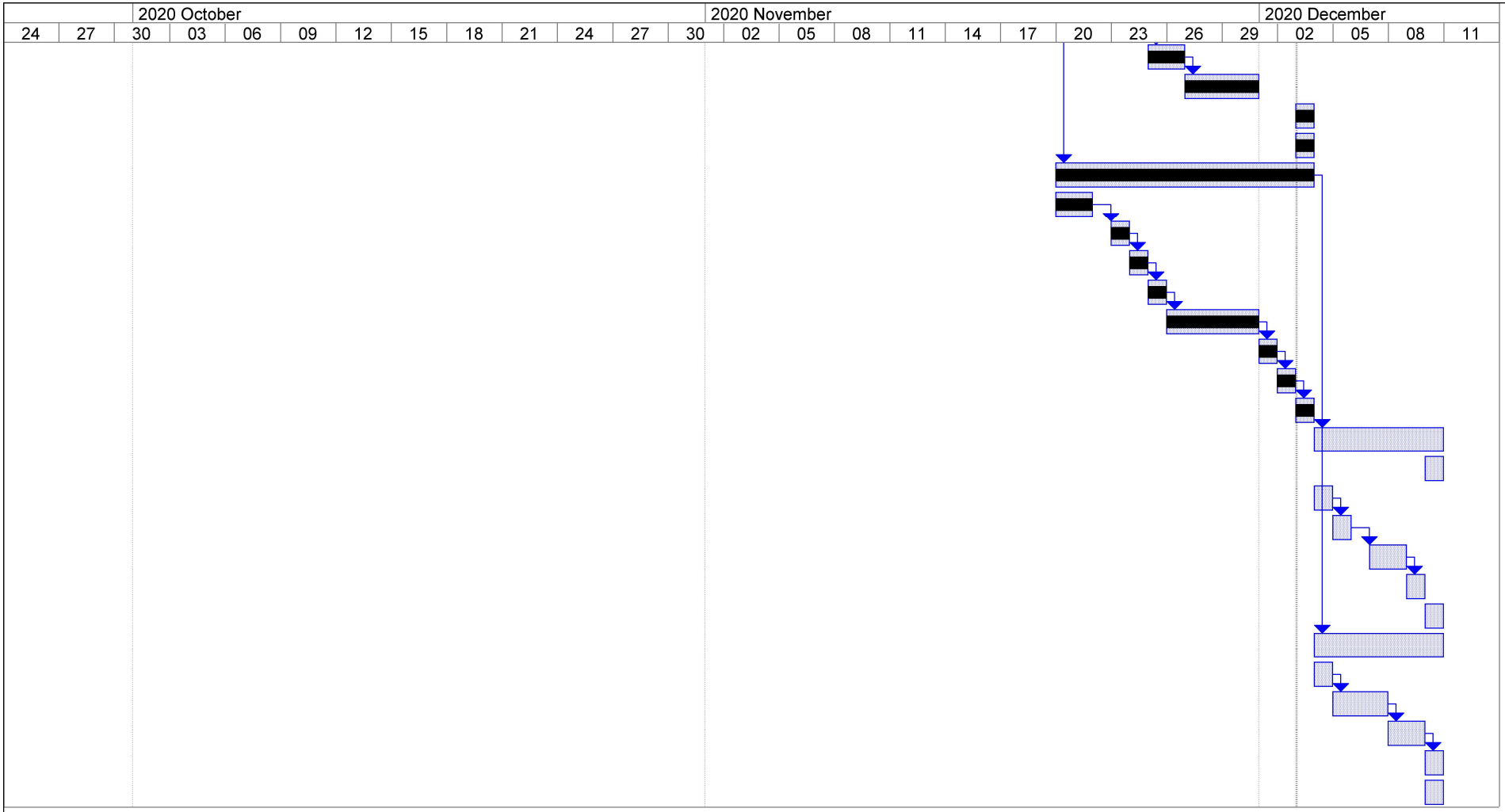
Project: Squelette de plan GNG
Date: Thu 20-12-03

Task
Split
Progress
Milestone
Summary
Project Summary
External Tasks
External Milestone
Deadline



Project: Squelette de plan GNG
Date: Thu 20-12-03

Task		Milestone		External Tasks	
Split		Summary		External Milestone	
Progress		Project Summary		Deadline	



Project: Squelette de plan GNG
Date: Thu 20-12-03

Task
Split
Progress
Milestone
Summary
Project Summary
External Tasks
External Milestone
Deadline