

Livrable	4
Sigle du cours	SEG2505
Professeur	Aziz Oukaira
Assistant (e)	Alexia Capo-Chichi
Groupe	G03



Table des Matières

Les diagrammes de classes UML	4
Design de l'application	6
Page de connexion et création d'un compte	6
Compte administrateur	7
Compte employé	9
Compte client	11
Backend de l'application	15
Test unitaires	16
Leçons apprises durant ce projet	17
Tableau avec rôle et contributions de chaque membre	18
Ligne de temps de développement pour le livrable 4	20
Conclusion	21

Membres du groupe

- Céline Wan Min Kee (celinemk), 300193369
- Evan Marth (emarth), 300166093
- Samy Touabi (SamyT-code), 300184721
- Tisham Islam (TishamIslam), 300189261
- Othniel Tiendrebeogo (othnielt), 300084968

Livrable 4: Implémentation des fonctionnalités liées au client et finalisation de l'application + Rapport final

Ce projet a pour but d'appliquer les connaissances acquises ce semestre afin de créer une application qui implémente des fonctionnalités de base pour les services offerts par une province imaginaire appelée Novigrad à ses résidents. Cette application est similaire à Service Ontario ou Services Québec qui permettent aux citoyens d'obtenir un permis de conduire, une pièce d'identité avec photo ou une carte Santé. L'application permet à l'utilisateur de se connecter en tant qu'administrateur, employé, et client. Chacun de ces types d'utilisateurs distinct aura accès à des fonctionnalités de l'application que les autres utilisateurs n'auront pas. Par exemple, les clients peuvent faire une demande de service, les employés peuvent approuver ou rejeter les demandes, et les administrateurs peuvent supprimer ou modifier les services.

Pour construire cette application et collaborer tout au long du semestre, nous avons utilisé Android Studio, Firebase et Github. Nous avons aussi essayé de connecter CircleCi avec l'application pour faciliter la vérification des tests unitaires mais cela a posé des problèmes que nous n'avions pas réussi à résoudre. On a donc abandonné l'intégration de CircleCi avec notre application.

1. Les diagrammes de classes UML

Pour compléter le diagramme UML de notre application, nous avons ajouté les classes que nous avions besoin afin de réaliser le livrable 4. Nous avons ajouté de nouvelles activités qui seront nécessaire pour ce livrable (Fonctionnalité client):

- ServiceSearch: page pour rechercher une succursale/service
- SubmitDemandePage: page pour soumettre une demande de service
- ClientDemandesPage: page où un client peut voir l'état de ses demandes courantes

On a aussi ajouté la classe Rating pour stocker les valeurs des évaluations pour succursales, avec un arbre correspondant dans notre base de données. Nous avons également ajouté une sous-classe de Demande nommée SearchDemande qui représente une Demande quand elle est recherchée en fonction des filtres, ainsi qu'une classe Time pour représenter un jour avec une heure.

Pour la diagramme UML, on a décidé d'omettre quelque méthodes et classes: chaque page a une méthode *onReturn(View)* (ou équivalente) pour aller à la page précédente et une méthode *onCreate()/onStart()* pour initialiser les variables et mettre des *listeners* pour la base de données. Donc, ils ne sont pas dans le diagramme pour le simplifier. Les composantes de UI (ex: boutons) sont aussi omises pour la simplification. Il y aussi quelques classes qui sont omises car elles représentent la même chose que des tableaux (chaque classe qui a un préfixe "List", ex: IntervalList et DemandeList). Ils fonctionnent identiquement aux tableaux/listes, mais ont aussi des fonctionnalités pour le UI. Les classes principales qui sont mises en Firebase (ex. UserAccount, Succursale, etc.) ont des méthodes *getter/setter* qui sont aussi omises de notre UML car Firebase nécessite l'utilisation des méthodes *getter* pour stocker les valeurs d'instance automatiquement. Par contre, on a décidé d'inclure les nouvelles méthodes *get/setUserInstance* car ce sont des méthodes uniques parmi nos classes, inspirées par le patron de singleton introduit en classe.

Ainsi notre diagramme UML final est représenté dans la Figure 1 ci-dessous.

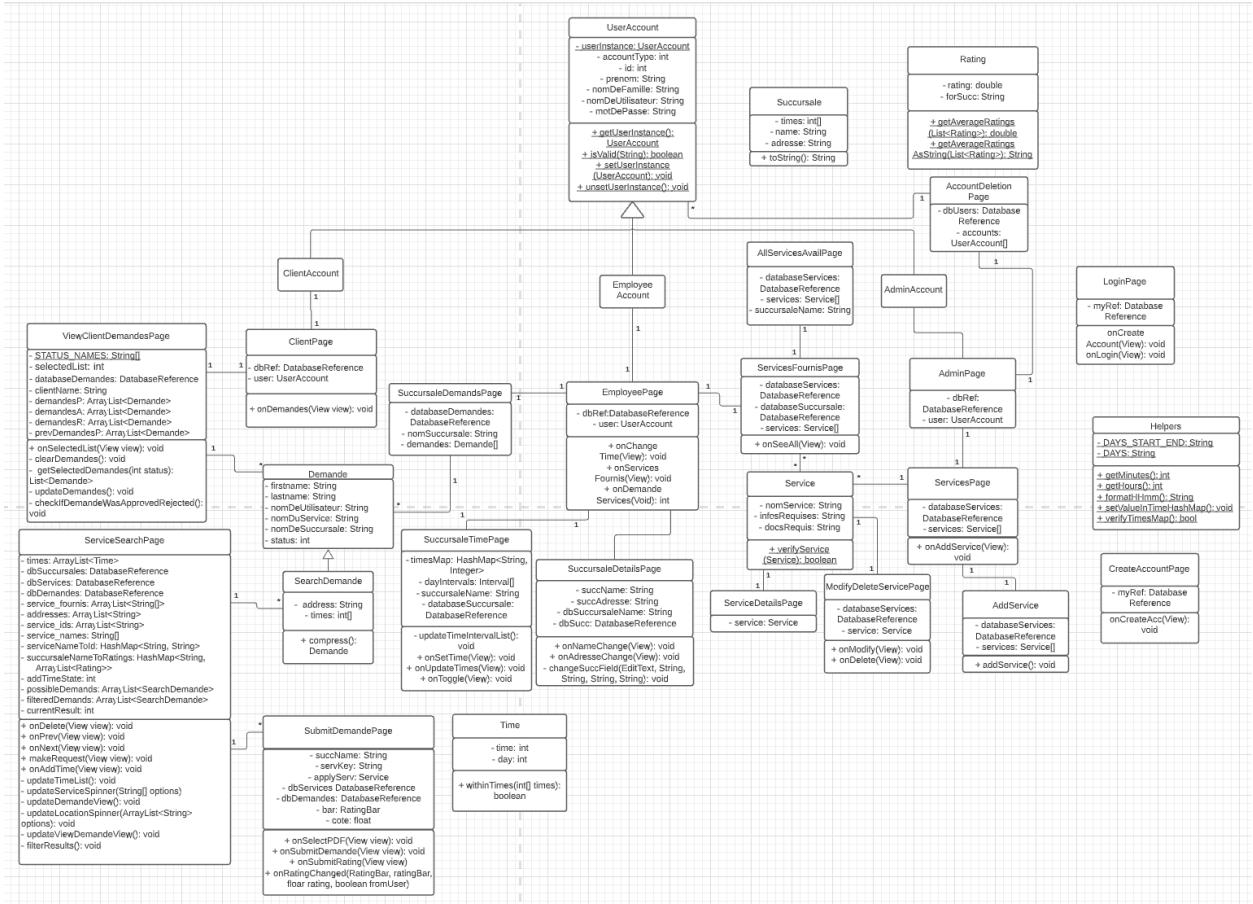


Figure 1: Diagramme UML

2. Design de l'application

Nous avons implémenté l'interface usager de notre application dans un style "Dark Mode" et avons utilisé le même thème de couleurs (couleur bleu vert, mauve) pour toutes les pages afin de rendre l'application simple et cohérente.

a. Page de connexion et création d'un compte

Sur la page de connexion, l'utilisateur entre son nom d'utilisateur et mot de passe. Il peut décider de voir ou pas ce qu'il écrit dans pour le mot de passe. Il y a des messages Toast également qui s'affichent lorsqu'il manque des informations, le mot de passe est incorrect ou le compte n'existe pas.

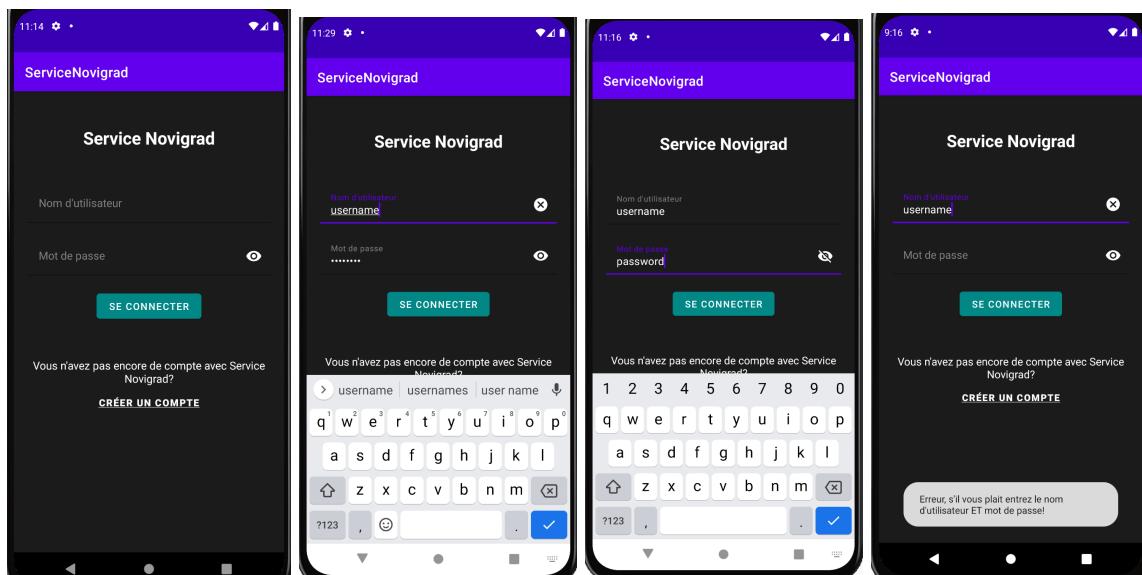


Figure 2: Interface usage de la page de connexion

Notre page de connexion permet aux utilisateurs de se connecter à leur compte et de démasquer le mot de passe qu'ils sont en train d'écrire afin qu'ils ne se trompent pas. Elle possède également un bouton qui nous mène vers la page de création d'un compte.

Pour créer un compte sur notre application, un utilisateur a besoin de fournir les informations suivantes: son prénom, son nom de famille, un nom d'utilisateur, un mot de passe ayant au moins 6 caractères et le type de compte qu'il veut créer.



Figure 3: Interface usage pour créer un compte

Évidemment, l'application vérifie si le nom d'utilisateur entré est déjà présent dans la base de données avant de l'y ajouter. Sinon, l'application affichera un message Toast pour prévenir l'utilisateur qu'il existe déjà un compte avec ce nom d'utilisateur. Des messages Toast s'affichent également lorsqu'il y a des champs de textes invalides.

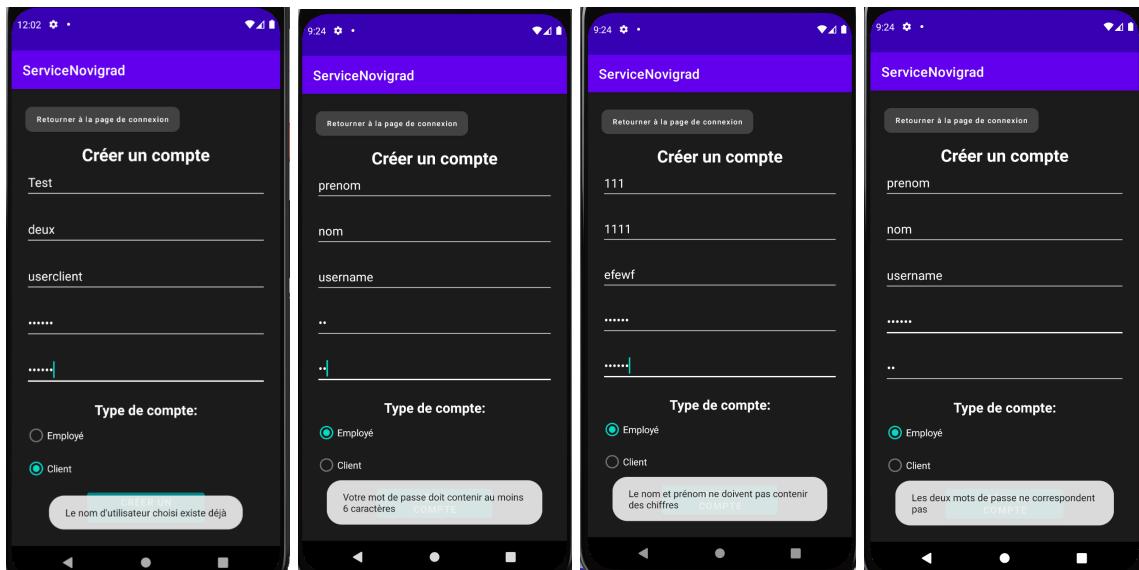


Figure 4: Aperçus de quelques messages Toast affichés

b. Compte administrateur

Lorsqu'on se connecte au compte administrateur pré-créé, il y a un message de bienvenue qui s'affiche ainsi que les différentes fonctionnalités liées au compte. Il y a 2 options sur la page car l'administrateur est capable de :

- Ajouter, modifier et supprimer des services
- Supprimer des comptes clients et employés

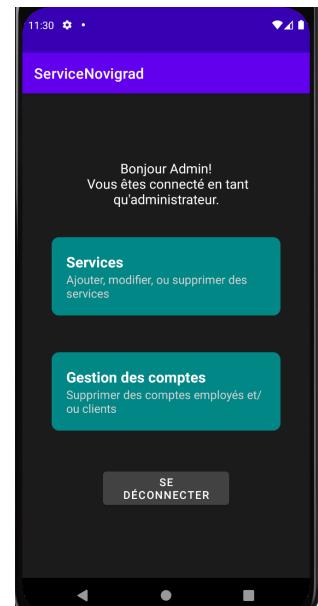


Figure 5: Page de bienvenue pour administrateur

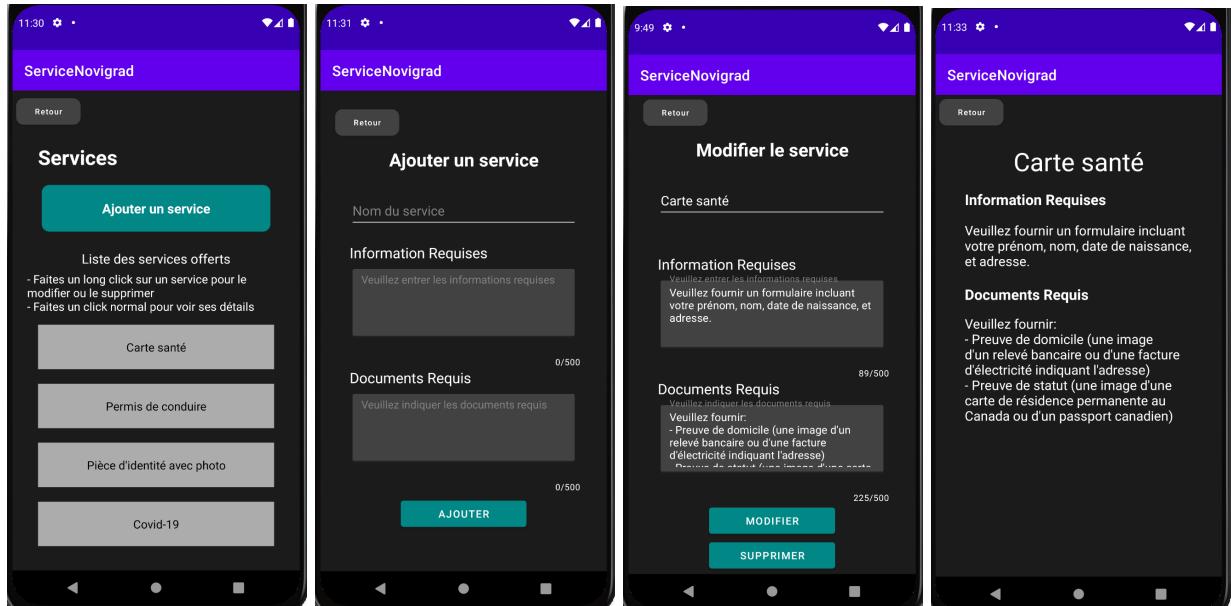


Figure 6: Fonctionnalités d'ajouter, modifier et supprimer des services

Des messages Toast s'affichent lorsqu'il y a une action qui a été effectué avec succès ou il y a des champs de textes invalides:

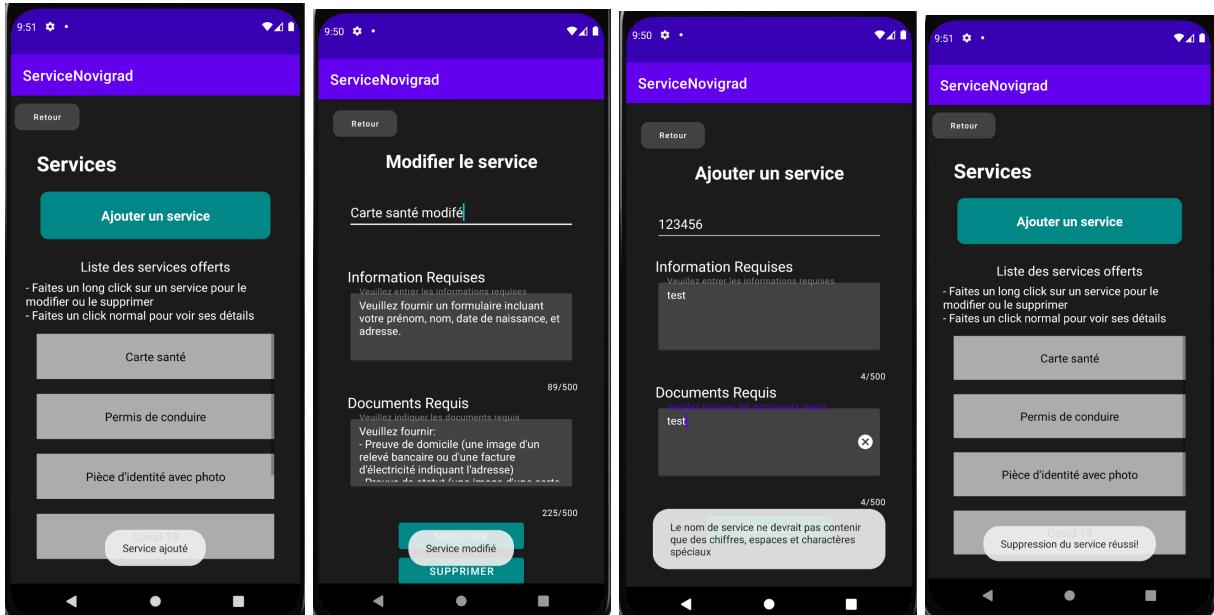


Figure 7: Aperçus de quelques messages Toast affichés

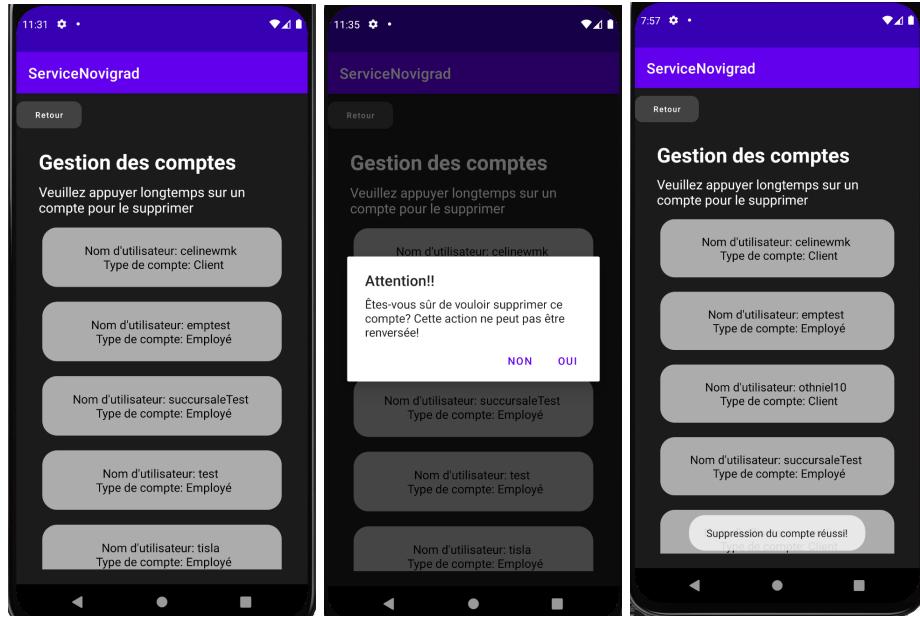


Figure 8: Fonctionnalité pour supprimer un compte

c. Compte employé

Lorsqu'on se connecte à un compte employé, il y a un message de bienvenue qui s'affiche ainsi que les différentes fonctionnalités liées au compte. Il y a 4 options sur la page car l'employé est capable de:

- Ajouter et supprimer des services fournis par la succursale
- Entrer et modifier ses heures de travail
- Voir les demandes de services
- Modifier les détails de la succursale

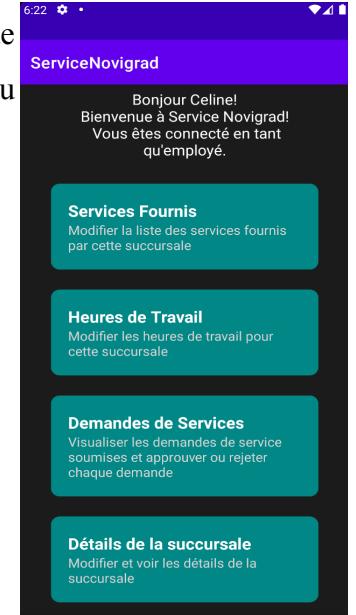


Figure 9: Page de bienvenue pour l'employé

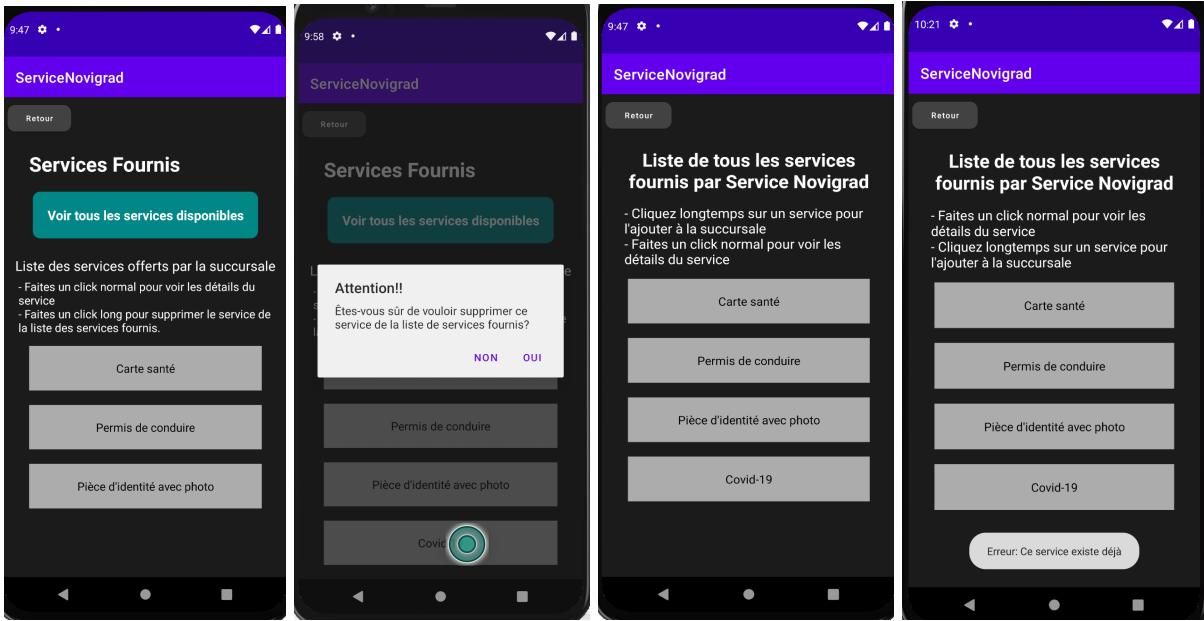


Figure 10: Fonctionnalités pour les services fournis par la succursale avec message Toast

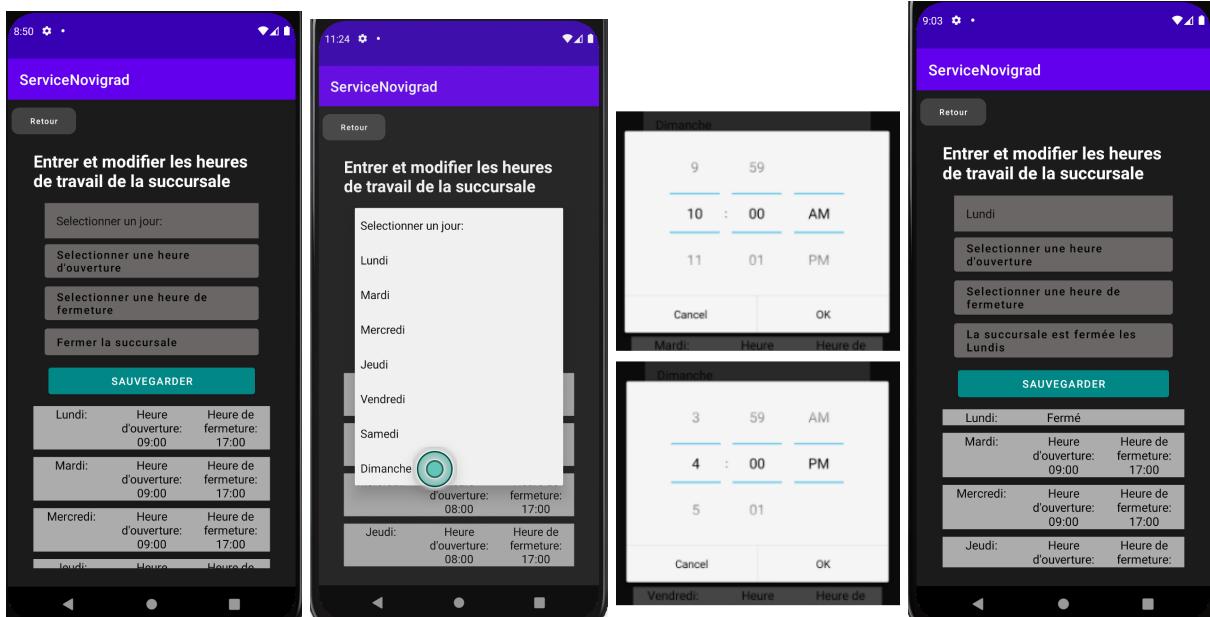


Figure 11: Fonctionnalités pour les heures de travail d'une succursale

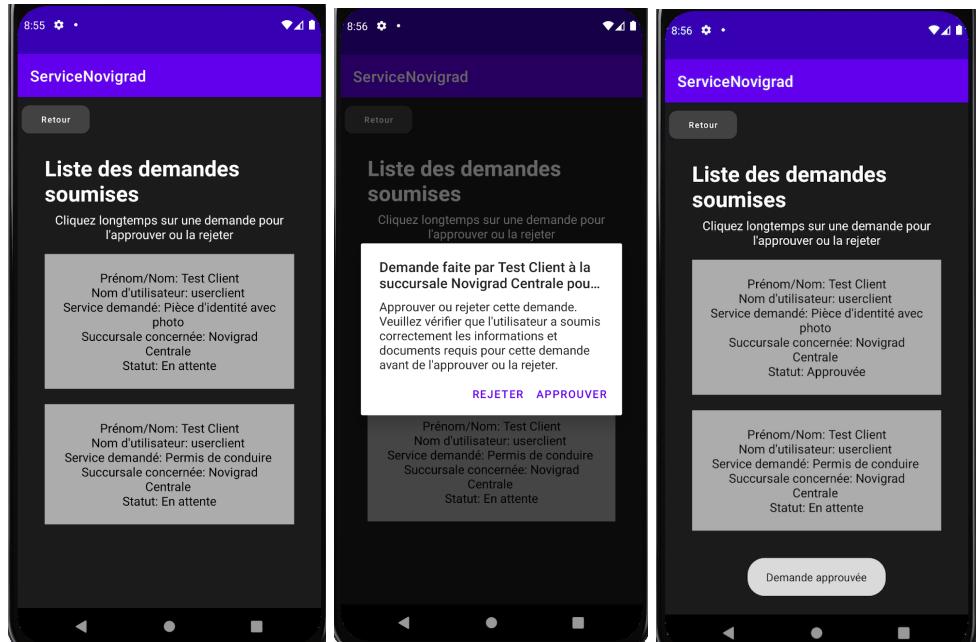


Figure 12: Fonctionnalité pour approuver/rejeter une demande

d. Compte client

Lorsqu'on se connecte à un compte client, il y a un message de bienvenue qui s'affiche ainsi que les différentes fonctionnalités liées au compte. Il y a 2 options sur la page car le client est capable de:

- Rechercher une succursale, soumettre sa demande et évaluer la succursale
- Voir ses demandes soumises à différentes succursales et s'ils sont approuvées/rejetées

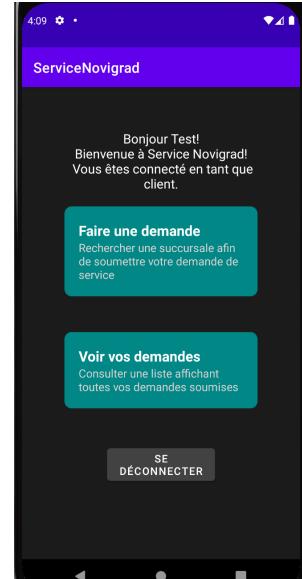


Figure 13: Page de bienvenue pour le client

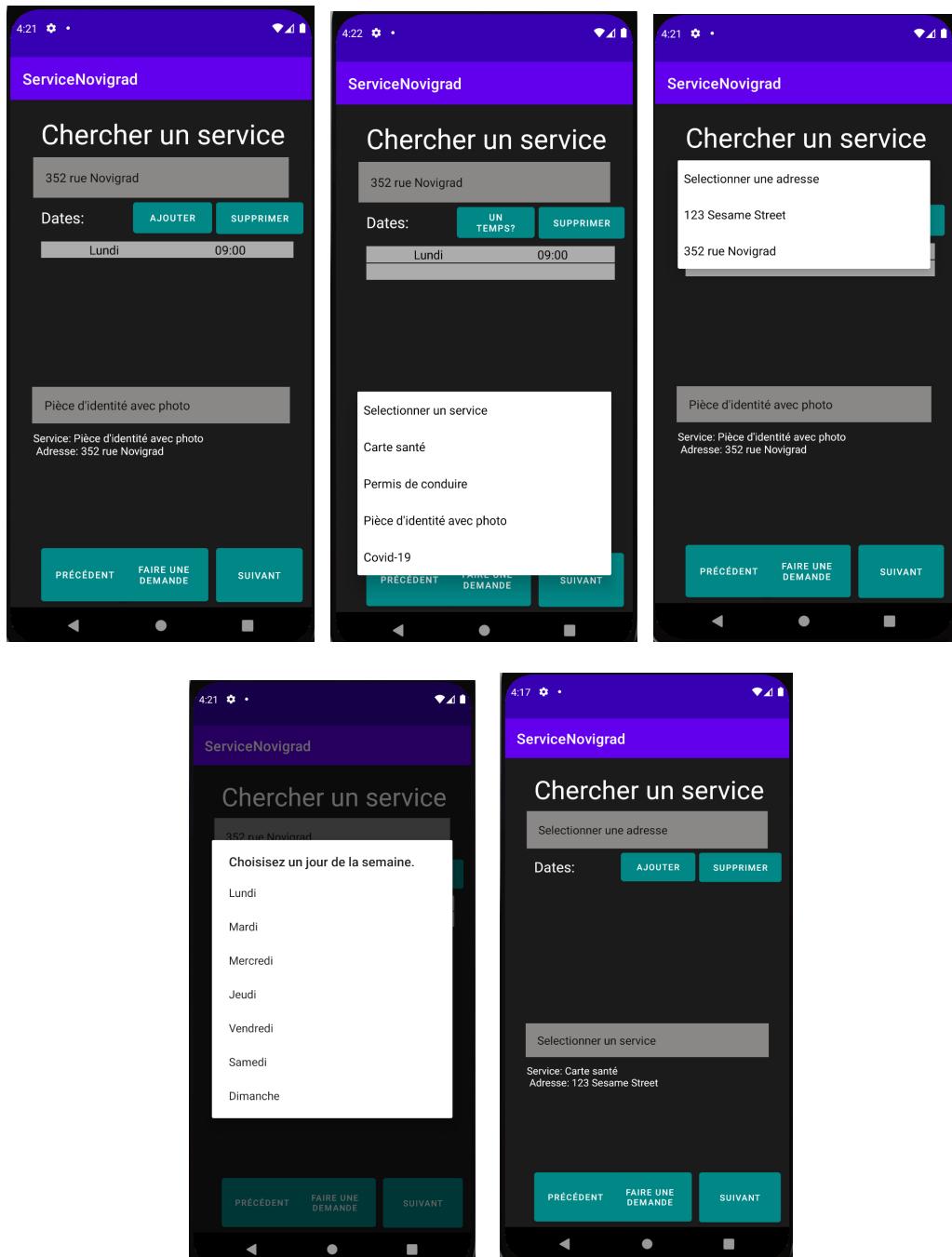


Figure 14: Fonctionnalité pour rechercher une succursale

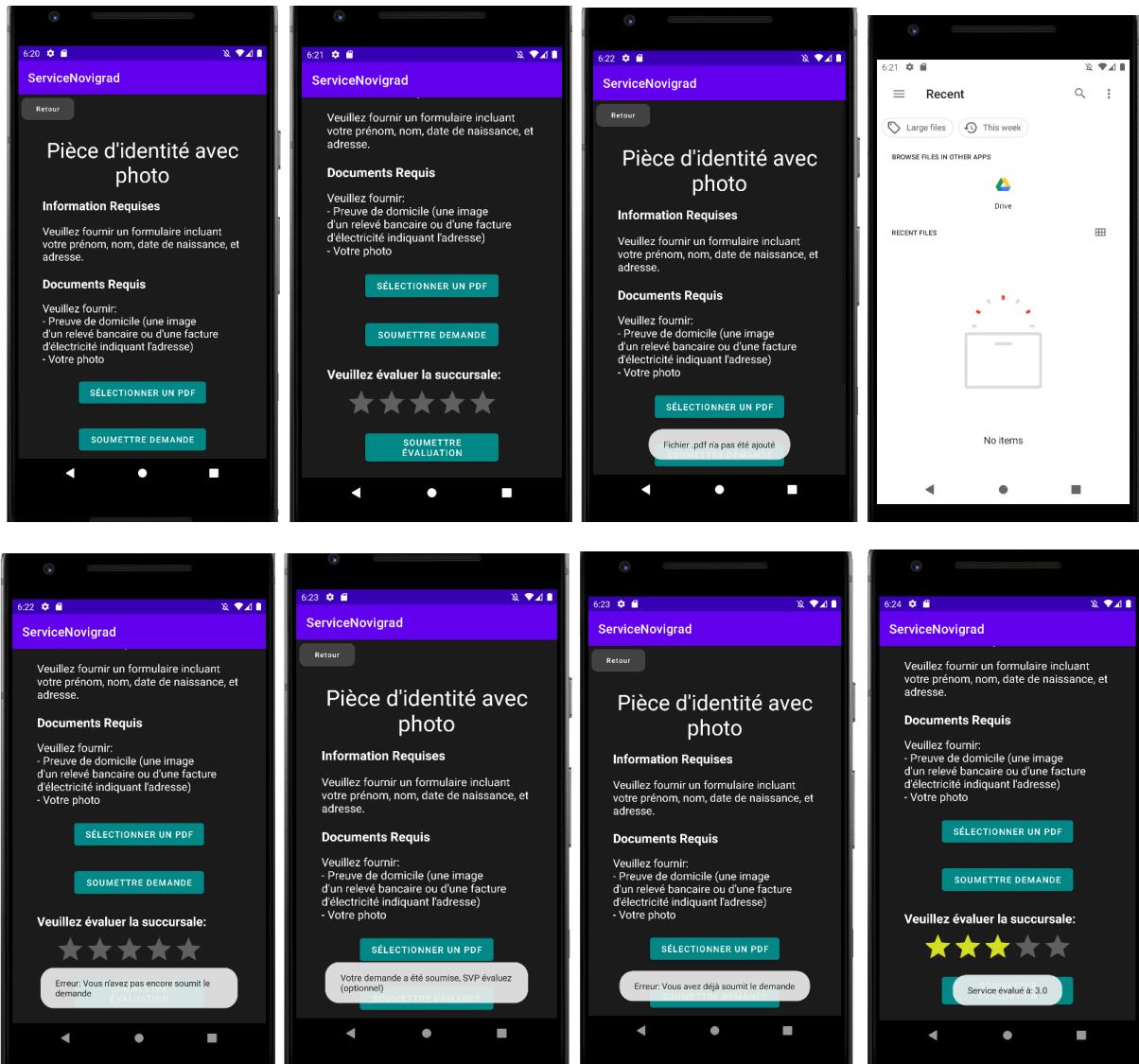


Figure 15: Fonctionnalité pour soumettre une demande et évaluer la succursale

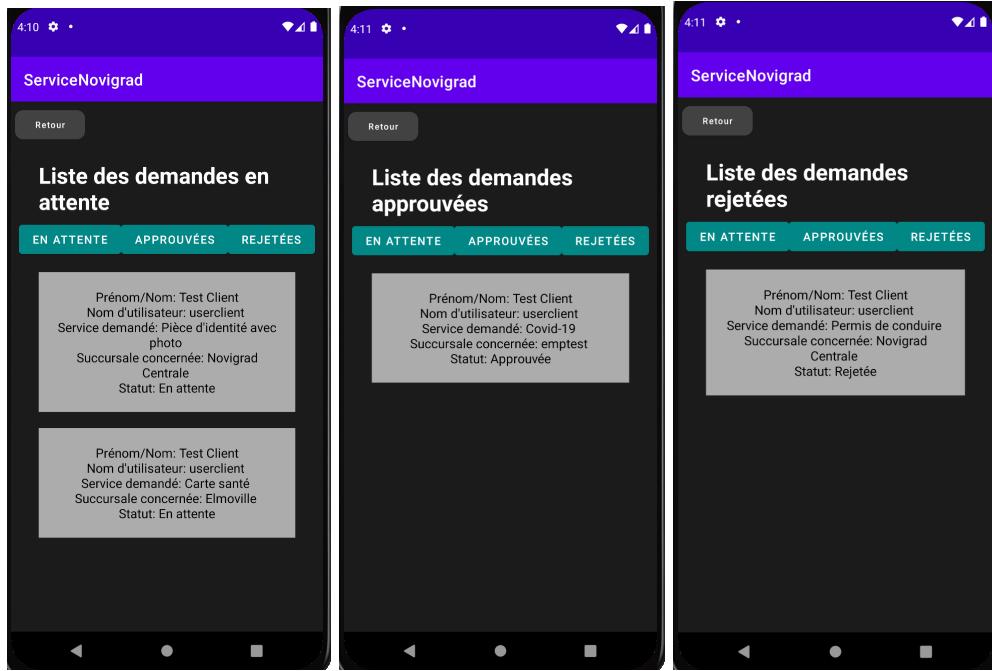


Figure 16: Fonctionnalité pour voir les demandes soumises du côté client

Le client reçoit une notification lorsqu'une de ses demandes a été approuvée ou rejetée.

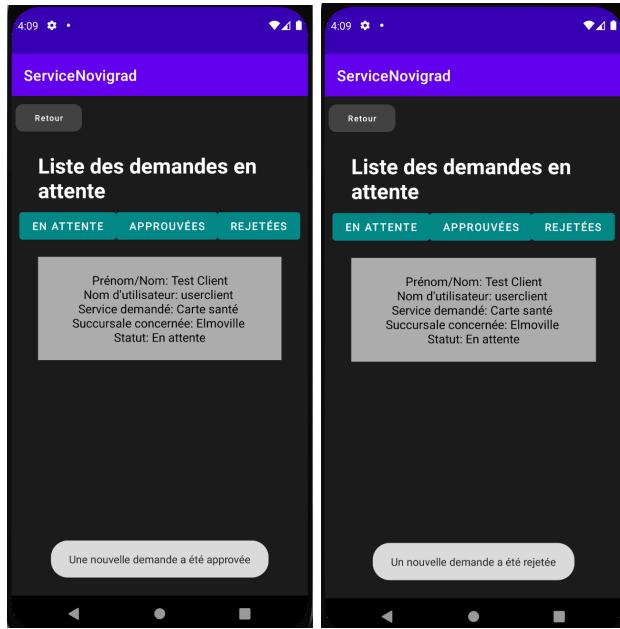


Figure 17: Notification reçu par le client

3. Backend de l'application

Pour garder en mémoire les comptes créés, les services proposés, les différentes succursales et les demandes de services soumises, nous utilisons Firebase. Notre base de données possède 5 branches principales:

- Une branche “demandes” qui garde en mémoire les demandes soumises
- Une branche “ratings” qui garde en mémoire les notes de chaque succursale
- Une branche “services” qui garde en mémoire les services offerts par Service Novigrad
- Une branche “succursales” qui garde en mémoire les informations sur chaque succursale
- Une branche “users” qui garde en mémoire tous les comptes de Service Novigrad

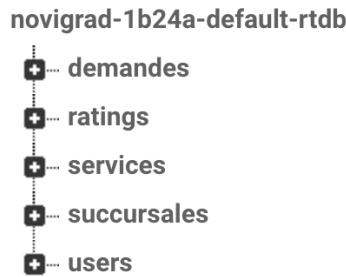


Figure 6: Les différentes branches de notre RealTime Database sur Firebase



Figure 7: Aperçus des branches “users”, “succursales” et “ratings” dans Firebase

Pour la branche “users”, un accountType d'une valeur 0 signifie que c'est un compte client, une valeur de 1 signifie que c'est un compte employé et une valeur de 2 que c'est un compte administrateur.

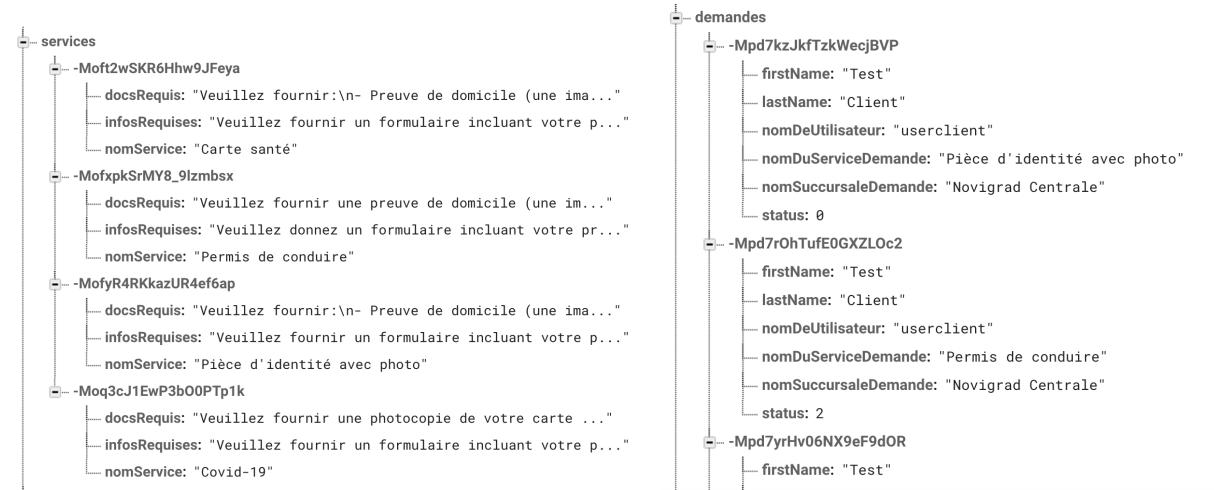


Figure 8: Aperçus des branches “services” et “demandes” dans Firebase

Pour la branche “demandes”, un statut d’une valeur 0 pour une demande signifie qu’elle est en attente, une valeur de 1 signifie qu’elle a été approuvé et une valeur de 2 signifie qu’elle a été rejetée.

4. Test unitaires

Nous avons écrit des tests unitaires Java afin de tester le bon fonctionnement de différentes méthodes de notre application.

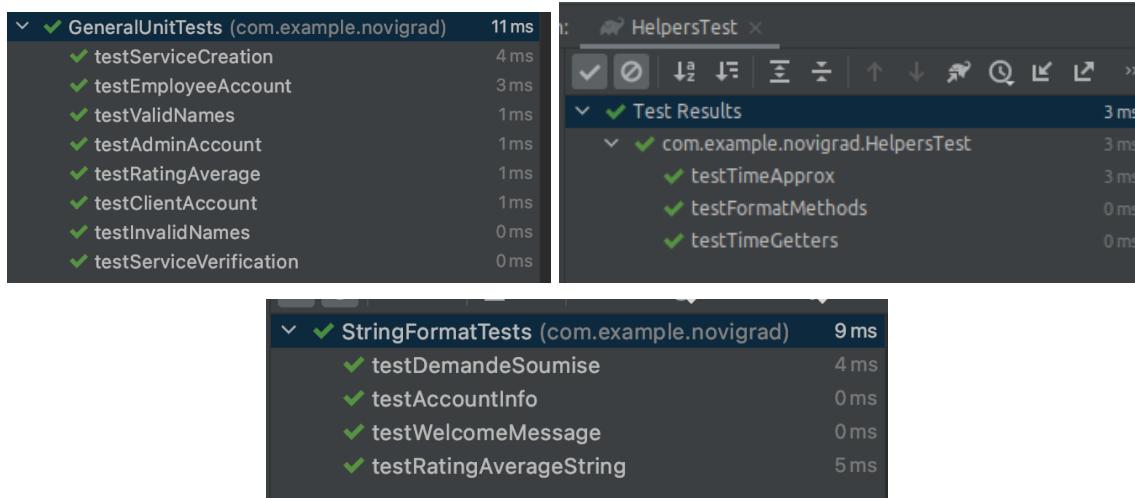


Figure 13: Résultats des tests unitaires

5. Leçons apprises durant ce projet

Ce projet de développement d'une application nous a permis de mettre en pratique les connaissances apprises tout au long du semestre. En plus de l'apprentissage du développement d'application, nous avons aussi implémentés des diagrammes UML en suivant un processus logique et en suivant les standards de développement professionnel afin d'organiser nos idées et ainsi réduire le temps de conception de l'application. En outre, nous avons appris à utiliser GitHub, un outil de contrôle de la source qui permet à un groupe de développeurs de travailler de manière cohérente sur un projet. L'utilisation de GitHub est très répandue dans le monde du génie logiciel et de l'informatique, et savoir comment l'utiliser sera certainement un atout important dans nos futures carrières. Aussi, nous avons appris comment utiliser des bases de données telles que SQLite et FireBase pour stocker et gérer l'information afin de pouvoir l'utiliser à notre guise dans l'application. Par exemple, un utilisateur client peut demander à ce qu'un service soit ajouté à une succursale, et plus tard un employé de la succursale peut voir la demande et l'approuver ou la rejeter, donnant ainsi un aspect de continuité et cohérence à l'application. Enfin, ce projet nous a appris comment travailler avec un groupe de cinq personnes pour travailler ensemble à distance. Tenter tant bien que mal de respecter les dates limites et décider comment répartir les tâches entre nous a renforcé des compétences sociales et a mis au test nos aptitudes d'organisation et d'autorégulation.

Ainsi, ce projet nous a préparé à être des meilleurs travailleurs non seulement dans le domaine du génie logiciel, mais aussi dans tout autre domaine nécessitant un travail d'équipe.

6. Tableau avec rôle et contributions de chaque membre

Livrable 1	Céline:
	<ul style="list-style-type: none"> - Développement du diagramme UML - Design du front-end - Implémentation des messages Toast
	Evan:
	<ul style="list-style-type: none"> - Développement du diagramme UML - Backend pour ajouter des comptes créés
	<p>Samy:</p> <ul style="list-style-type: none"> - Développement du diagramme UML - Design du front-end - Implémentation des messages Toast
Livrable 2	Tisham:
	<ul style="list-style-type: none"> - Développement du diagramme UML - Implémentation des messages Toast - Débogage
	Othniel:
	<ul style="list-style-type: none"> - Implémentation des messages Toast
	<p>Céline:</p> <ul style="list-style-type: none"> - Développement du diagramme UML - Design du front-end + navigation sur l'application - Backend pour ajout de services
Livrable 2	Evan:
	<ul style="list-style-type: none"> - Développement du diagramme UML - Test Unitaires - Débogage
	<p>Samy:</p> <ul style="list-style-type: none"> - Développement du diagramme UML - Design du front-end - Implémentation de messages Toast
	<p>Tisham:</p> <ul style="list-style-type: none"> - Développement du diagramme UML - Design du front-end - Backend pour modification/suppression de services et suppression de comptes
	<p>Othniel</p> <ul style="list-style-type: none"> - Test Unitaires

Livrable 3	<p>Céline:</p> <ul style="list-style-type: none"> - Développement du diagramme UML - Design du front-end - Backend pour les demandes de services
	<p>Evan:</p> <ul style="list-style-type: none"> - Développement du diagramme UML - Backend pour les heures de travail
	<p>Samy:</p> <ul style="list-style-type: none"> - Développement du diagramme UML - Design du front-end
	<p>Tisham:</p> <ul style="list-style-type: none"> - Développement du diagramme UML - Backend pour les heures de travail et services de la succursale - Test unitaires
	<p>Othniel:</p> <ul style="list-style-type: none"> - Documentation du code
Livrable 4	<p>Céline:</p> <ul style="list-style-type: none"> - Développement du diagramme UML - Préparation de la présentation - Design du front-end - Test unitaires
	<p>Evan:</p> <ul style="list-style-type: none"> - Design du front-end - Backend pour la recherche d'une succursale
	<p>Samy:</p> <ul style="list-style-type: none"> - Design du front-end - Préparation de la présentation
	<p>Tisham:</p> <ul style="list-style-type: none"> - Développement du diagramme UML - Design du front-end - Backend pour demande approuvée/rejetée et la soumission de demande
	<p>Othniel:</p> <ul style="list-style-type: none"> - N/A

7. Ligne de temps de développement pour le livrable 4

- 03/12/2021: Évaluation des besoins du livrable 3 + Continuation du développement du diagramme UML + Développement front-end
 - Tisham et Samy ont commencé à implémenter l'interface usager lié au compte client
- 04/12/2021: Front-end + backend
 - Céline a mis à jour la page de bienvenue du client afin qu'elle suit le même style que celle des pages administrateur et employé
 - Tisham a implémenté les soumissions de demande faites par le client, ainsi que approuver/rejeter une demande
 - Samy et Céline ont commencé à préparer les diapositives pour la présentation
 - Samy a mis à jour la page pour modifier les services afin qu'elle ressemble davantage à la page pour créer des services
- 05/12/2021 et 06/12/2021:
 - Evan a travaillé sur la fonctionnalité de recherche d'une succursale
 - Préparation des Google slides pour la présentation orale
- 07/12/2021
 - Présentation!
 - Tisham et Samy ont ajouté la page pour soumettre un demande
- 08/12/2021:
 - Céline a ajouté les tests unitaires
 - Tisham a terminé la fonctionnalité d'évaluation d'une succursale et de connexion de page de recherche d'une succursale au page pour soumettre une demande.
 - Quelques corrections de bogues par Tisham
 - Soumission du livrable

8. Conclusion

Dans ce livrable nous avons fini le développement de notre application pour le Service Novigrad qui offre des services à ses clients. Nous avons cette fois-ci implémenté les fonctionnalités du compte client de l'application. Le client peut rechercher une succursale en fonction de filtres, soumettre une demande de service et évaluer la succursale. Il peut également voir l'état de chacune de ses demandes. Nous avons aussi implémenté des tests unitaires additionnels afin de tester le bon fonctionnement de nos méthodes. Maintenant, notre base de données gardent en mémoire les comptes, les services, les informations sur les succursales, les demandes soumises et les notes attribuées à chacune des succursales.