|  |
| --- |
| TITRE |

**Projet :** Projet volontaire

**Auteur :** Oliveira Ramos Dylan

**Ecole :** Centre professionnel du Nord vaudois - filière informatique

**Période :** Du 12 mai au 5 juin 2020

# Table des matières

[Table des matières 1](#_Toc40348283)

[Glossaire 2](#_Toc40348284)

[Analyse préliminaire 2](#_Toc40348285)

[Introduction 2](#_Toc40348286)

[Objectifs 2](#_Toc40348287)

[Planification initiale 3](#_Toc40348288)

[Analyse / Conception 4](#_Toc40348289)

[Concept 4](#_Toc40348290)

[Maquettes graphiques de la partie client 4](#_Toc40348291)

[Maquette graphique de la partie serveur 7](#_Toc40348292)

[Cas d’utilisation 8](#_Toc40348293)

[Partie client 8](#_Toc40348294)

[Cas n°1 : Connexion à l’application de messagerie avec les bons identifiants 8](#_Toc40348295)

[Cas n°2 : Connexion à l’application de messagerie avec les mauvais identifiants 8](#_Toc40348296)

[Cas n°3 : Création d’un compte utilisateur 9](#_Toc40348297)

[Cas n°4 : Création d’un compte utilisateur en insérant 2 mots de passe différents 9](#_Toc40348298)

[Cas n°5 : Envoi d’un message 10](#_Toc40348299)

[Cas n°5 : Réception d’un message 10](#_Toc40348300)

[Partie serveur 10](#_Toc40348301)

[Cas n°1 : Démarrage du serveur 10](#_Toc40348302)

[Cas n°2 : Fermeture du serveur 11](#_Toc40348303)

[Base de données 11](#_Toc40348304)

[Modèle conceptuel des données 11](#_Toc40348305)

[Modèle logique des données 11](#_Toc40348306)

[Stratégie de test 11](#_Toc40348307)

[Risques techniques 12](#_Toc40348308)

[Planification 12](#_Toc40348309)

[Dossier de conception 12](#_Toc40348310)

[Réalisation 12](#_Toc40348311)

[Dossier de réalisation 12](#_Toc40348312)

[Description des tests effectués 13](#_Toc40348313)

[Erreurs restantes 13](#_Toc40348314)

[Liste des documents fournis 13](#_Toc40348315)

[Conclusions 13](#_Toc40348316)

[Annexes 15](#_Toc40348317)

[Résumé du rapport du TPI 15](#_Toc40348318)

[Journal de travail 15](#_Toc40348319)

[Manuel d'Installation 15](#_Toc40348320)

[Manuel d'Utilisation 15](#_Toc40348321)

[Archives du projet 15](#_Toc40348322)

[Table des illustrations 15](#_Toc40348323)

[Sources 15](#_Toc40348324)

# Glossaire

**IDE** : Integrated Development Environment, il s'agit de l'environnement de développement. C'est un ensemble d'outils aidant au développement de logiciels.

**Logs :** Désigne un fichier permettant de stocker un historique des évènements attachés à un processus.

**MCD :** Le modèle conceptuel de données est une description graphique pour représenter des modèles de données sous la forme de diagrammes contenant des entités et des associations.

**MLD :** Le modèle logique de données reprend le contenu du MCD, mais précise la volumétrie, la structure et l'organisation des données telles qu'elles pourront être implémentées.

# Analyse préliminaire

## Introduction

Ce projet volontaire consiste à développer une application de messagerie instantanée réalisée par Dylan Oliveira Ramos pour le Centre professionnel du Nord vaudois de Sainte-Croix (filière informatique).

Etant donné la pandémie du COVID-19, le projet est entièrement réalisé à domicile. Le début de celui-ci est fixé au 12 mai 2020 et se termine le 5 juin 2020, sa durée est de 80 heures, soit 20 heures par semaine (4 semaines).

Le programme sera réalisé en C# sous forme de formulaires Windows en utilisant un IDE et sera connecté à un serveur avec une base de données.

## Objectifs

Le but de cette application est de pouvoir envoyer des messages d’un utilisateur connecté à un autre.

Afin que les utilisateurs puissent se connecter, un système de login doit être intégré à l’application. Pour les utilisateurs qui ne possèdent pas encore de compte, la création de celui-ci est indispensable, il faut donc intégrer un formulaire de création de compte.

Lorsqu’un message est envoyé, le destinataire le reçoit instantanément. Si le destinataire est hors-ligne, il recevra le message lors de sa prochaine connexion.

Tous les messages doivent être stockés dans une base de données afin qu’ils puissent être lus à tout moment.

Pour terminer, toutes les actions des utilisateurs doivent être enregistrées sous forme de logs sur le serveur.

## Planification initiale

# Analyse / Conception

## Concept

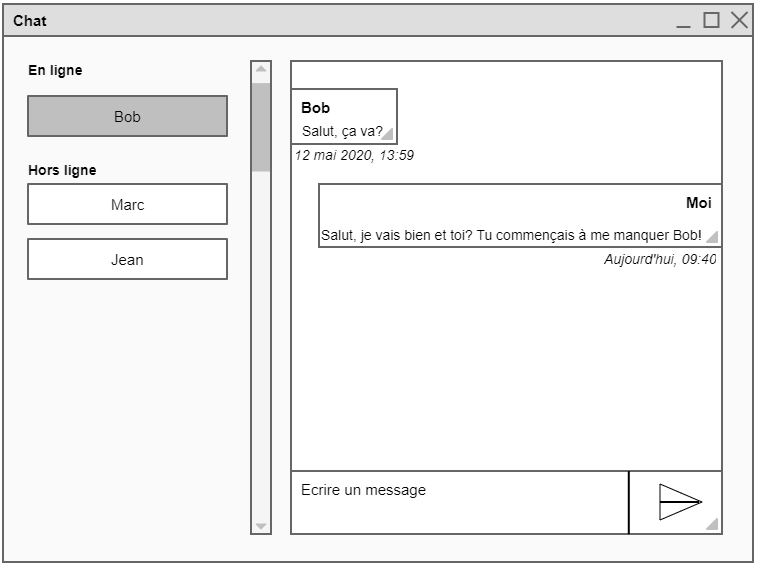
### Maquettes graphiques de la partie client

La page ci-dessous permet à l’utilisateur de se connecter ou d’accéder à la page de création d’un compte s’il n’en a pas. La page ci-dessous permet à l’utilisateur de se créer un compte.

1 - Page de login

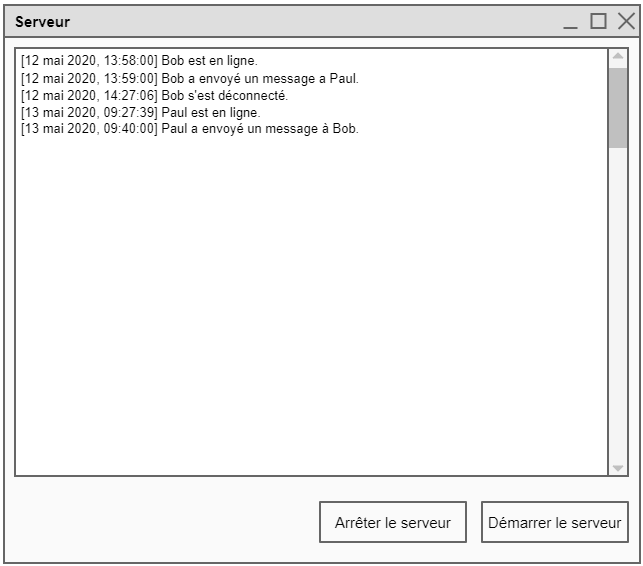
2 - Page de création d'un compte

3 - Page de chat

La page ci-dessous permet aux utilisateurs de communiquer entre eux une fois qu’ils se sont connectés.

4 - Page de chat

### Maquette graphique de la partie serveur

La page ci-dessous permet de démarrer ou arrêter le serveur et de voir ce qu’il se passe sur les clients en temps réel.

5 - Page de logs

## Cas d’utilisation

### Partie client

|  |
| --- |
| Cas n°1 : Connexion à l’application de messagerie avec les bons identifiants |

|  |  |
| --- | --- |
| **Précondition** | L’utilisateur est sur la page de login |
| **Scénario** | 1. L’utilisateur insère son nom d’utilisateur 2. L’utilisateur insère son mot de passe 3. L’utilisateur clique sur « Se connecter » |
| **Performance attendue** | La page de chat apparaît en moins de 5 secondes. |

|  |
| --- |
| Cas n°2 : Connexion à l’application de messagerie avec les mauvais identifiants |

|  |  |
| --- | --- |
| **Précondition** | L’utilisateur est sur la page de login |
| **Scénario** | 1. L’utilisateur insère un nom d’utilisateur 2. L’utilisateur insère un mot de passe 3. L’utilisateur clique sur « Se connecter » |
| **Performance attendue** | Un message d’erreur apparaît instantanément. |

|  |  |
| --- | --- |
| Cas n°3 : Création d’un compte utilisateur | |
| **Précondition** | L’utilisateur est sur la page de création d’un compte |
| **Scénario** | 1. L’utilisateur insère un nom d’utilisateur 2. L’utilisateur insère un mot de passe 3. L’utilisateur insère le même mot de passe dans le champ de vérification 4. L’utilisateur clique sur « Créer » |
| **Performance attendue** | Le compte est créé et la page de chat apparaît en moins de 5 secondes. |

|  |  |
| --- | --- |
| Cas n°4 : Création d’un compte utilisateur en insérant 2 mots de passe différents | |
| **Précondition** | L’utilisateur est sur la page de création d’un compte |
| **Scénario** | 1. L’utilisateur insère un nom d’utilisateur 2. L’utilisateur insère un mot de passe 3. L’utilisateur n’insère pas le même mot de passe dans le champ de vérification 4. L’utilisateur clique sur « Créer » |
| **Performance attendue** | Le compte n’est pas créé et un message d’erreur apparaît. |

|  |  |
| --- | --- |
| Cas n°5 : Envoi d’un message | |
| **Précondition** | L’utilisateur est sur la page de chat |
| **Scénario** | 1. L’utilisateur sélectionne un contact 2. L’utilisateur écrit un message 3. L’utilisateur clique sur le bouton d’envoi |
| **Performance attendue** | Le message est envoyé et il est affiché à l’écran avec la date d’envoi. |

|  |  |
| --- | --- |
| Cas n°5 : Réception d’un message | |
| **Précondition** | L’utilisateur est sur la page de chat |
| **Scénario** | 1. L’utilisateur sélectionne un contact |
| **Performance attendue** | Tous les messages envoyés sont affichés. |

### Partie serveur

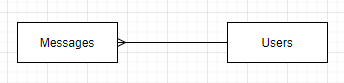
|  |  |
| --- | --- |
| Cas n°1 : Démarrage du serveur | |
| **Précondition** | L’utilisateur est sur la page de logs |
| **Scénario** | 1. L’utilisateur clique sur « Démarrer le serveur » |
| **Performance attendue** | Un message disant que le serveur a démarré apparaît et toutes les actions de utilisateurs à venir apparaissent. |

|  |  |
| --- | --- |
| Cas n°2 : Fermeture du serveur | |
| **Précondition** | L’utilisateur est sur la page de logs |
| **Scénario** | 1. L’utilisateur clique sur « Arrêter le serveur » |
| **Performance attendue** | Un message disant que le serveur a été arrêté apparaît et les utilisateurs ne peuvent plus s’envoyer de messages. |

## Base de données

### Modèle conceptuel des données

Le MCD pour cette application est très simple, il faut une table « Utilisateurs » et une table « Messages ». Un utilisateur doit pouvoir envoyer et recevoir plusieurs messages.



6 - MCD

### Modèle logique des données

**Table « Users »**

Le champ « UserId » contient le numéro d’identifiant unique de chaque utilisateur.

Le champ « UserName » contient le nom d’utilisateur unique de chaque utilisateur.

Le champ « UserPassword » contient le mot de passe haché de chaque utilisateur.

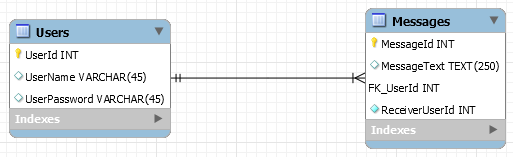
**Table « Messages »**

Le champ « MessageId » contient le numéro d’identifiant unique de chaque message.

Le champ « MessageText » contient le message.

Le champ « FK\_UserId » contient le numéro d’identifiant de l’utilisateur qui a envoyé le message.

Le champ « ReceiverUserId » contient le numéro d’identifiant de l’utilisateur destinataire.



## Stratégie de test

Les tests s’effectuent une fois les fonctionnalités terminées. Ceux-ci sont repris des cas d’utilisation et se réalisent avec des données fictives.

Les tests doivent être effectués par :

* Le candidat
* Le chef de projet
* Trois personnes non informaticiennes

## Risques techniques

Etant donné que je n’ai jamais créé d’application en réseau, je risque de devoir utiliser du temps pour aller m’informer sur internet.

## Planification

La planification détaillée est accessible sur Github : <https://github.com/dylanramos/MessagingApp/projects>

Celle-ci contient les tâches à faire, en cours et terminées. Les tâches sont ajoutées chaque jour à mesure que le projet avance.

## Dossier de conception

### Outils

**Système d’exploitation**

* Windows 10 Professionnel 64 bits.

**Réseau**

* Accès à un routeur pour obtenir des adresse IP

**IDE**

* Microsoft Visual Studio 2019

**Base de données**

* Package NuGet pour Visual Studio : SQLite

# Réalisation

## Dossier de réalisation

<https://material.io/resources/icons/?icon=verified_user&style=baseline>

using System.Net;

using System.Net.Sockets;

*Décrire la réalisation "physique" de votre projet*

* *les répertoires où le logiciel est installé*
* *la liste de tous les fichiers et une rapide description de leur contenu (des noms qui parlent !)*
* *les versions des systèmes d'exploitation et des outils logiciels*
* *la description exacte du matériel*
* *le numéro de version de votre produit !*
* *programmation et scripts: librairies externes, dictionnaire des données, reconstruction du logiciel - cible à partir des sources.*

*NOTE : Evitez d’inclure les listings des sources, à moins que vous ne désiriez en expliquer une partie vous paraissant importante. Dans ce cas n’incluez que cette partie…*

## Description des tests effectués

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

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

## Erreurs restantes

*S'il reste encore des erreurs:*

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

## Liste des documents fournis

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

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

# Conclusions

*Développez en tous cas les points suivants:*

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

# Annexes

## Résumé du rapport du TPI

## Journal de travail

## Manuel d'Installation

## Manuel d'Utilisation

## Archives du projet

*Media, … dans une fourre en plastique*

## Table des illustrations

[1 - Page de login 4](file:///C:\Users\Dylan\Desktop\MessagingApp\MessagingApp\Documentation\Rapport%20de%20projet.docx#_Toc40272064)

[2 - Page de création d'un compte 5](file:///C:\Users\Dylan\Desktop\MessagingApp\MessagingApp\Documentation\Rapport%20de%20projet.docx#_Toc40272065)

[3 - Page de chat 5](file:///C:\Users\Dylan\Desktop\MessagingApp\MessagingApp\Documentation\Rapport%20de%20projet.docx#_Toc40272066)

[4 - Page de chat 6](file:///C:\Users\Dylan\Desktop\MessagingApp\MessagingApp\Documentation\Rapport%20de%20projet.docx#_Toc40272067)

[5 - Page de logs 7](file:///C:\Users\Dylan\Desktop\MessagingApp\MessagingApp\Documentation\Rapport%20de%20projet.docx#_Toc40272068)

## Sources

# Bibliographie

Allexy. (2006, janvier 30). *TCP/IP Chat Application Using C#*. Récupéré sur Code Project: https://www.codeproject.com/Articles/12893/TCP-IP-Chat-Application-Using-C

Brian. (2012, août 22). *C# Async Sockets Part 1: Basics*. Récupéré sur YouTube: https://www.youtube.com/watch?v=Bq1JhTHlxek

CaptJiggly. (2012, mai 21). *C# Sockets Multiple Connection 1 - Accepting Connections*. Récupéré sur YouTube: https://www.youtube.com/watch?v=cHq2lYLA4XY

CaptJiggly. (2012, mai 22). *C# Sockets Multiple Connection 2 - Receiving Data/Handling Disconnection*. Récupéré sur YouTube: https://www.youtube.com/watch?v=p8Nlxtj0sV4

Dotnetperls. (consulté en mai 2020). *C# Split String Examples*. Récupéré sur Dotnetperls: https://www.dotnetperls.com/split

Stack Overflow. (2009, mars). *How do I update the GUI from another thread?* Récupéré sur Stackoverflow: https://stackoverflow.com/questions/661561/how-do-i-update-the-gui-from-another-thread

Stack Overflow. (2016, janvier). *Check Whether a TextBox is empty or not*. Récupéré sur Stackoverflow: https://stackoverflow.com/questions/34298857/check-whether-a-textbox-is-empty-or-not/34299121#34299121

TeapotDev. (2012, juillet 26). *Simple Instant Messenger with SSL Encryption in C#*. Récupéré sur Code Project: https://www.codeproject.com/Articles/429144/Simple-Instant-Messenger-with-SSL-Encryption-in-Cs

Tutorials, C. (2017, mars 11). *Client Server programming in C# (Chat application)*. Récupéré sur YouTube: https://www.youtube.com/watch?v=X16IyNbcAr0