Table des matières

[Description du sujet 2](#_Toc122289954)

[Diagramme des cas d’utilisation 2](#_Toc122289955)

[Description détaillée des cas d’utilisation 3](#_Toc122289956)

[Diagramme UML des classes 5](#_Toc122289957)

[Difficultés rencontrées 5](#_Toc122289958)

[Ajouts Bonis 5](#_Toc122289959)

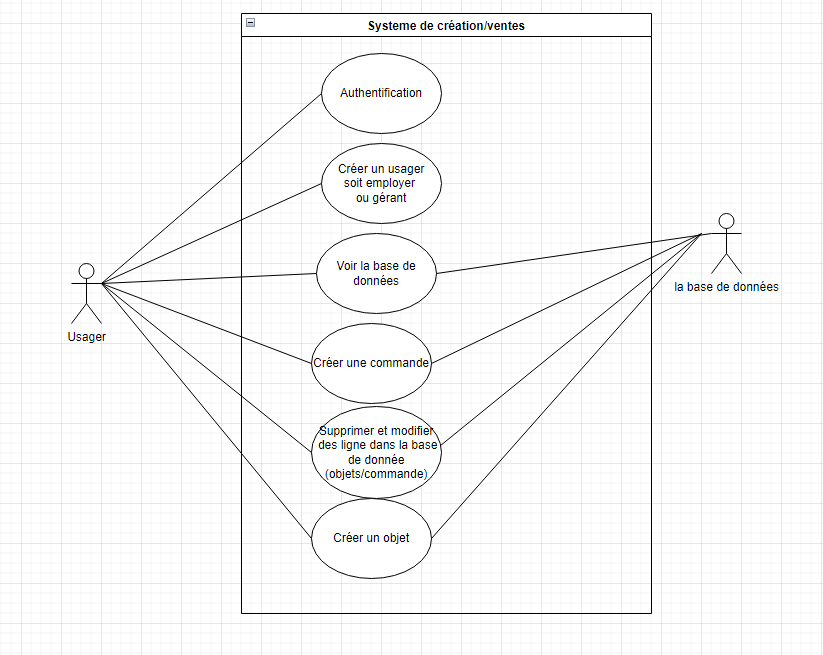
[Journal du temps de travail 6](#_Toc122289960)

[BIBLIOGRAPHIE 7](#_Toc122289961)

# Description du sujet

Mon application est un système de création d’objet dans le contexte d’une usine. L’application permet de créer/modifier/Supprimer des objets. Dans la première table de la base de données, nous avons les informations de l’objet créer et le nombre par heure qui sont créer. Dans la 2em table on y retrouve le nombre d’objet qui ont été commander et ses informations. On peut donc voir Crée de nouveau objets et faire des demandes de création(commande) pour ses objets.

# Diagramme des cas d’utilisation



# Description détaillée des cas d’utilisation

Nom : Voir la base de données

Description : permet de voir la base de données

Acteur : Usager

Données entrées : Soit le produit spécifique que l’on veut voire soit l’ensemble de la base de données

Scénario principal :

1. L’usager fait une demande au système pour accéder à la base de données(DB)(data grid view)

Nom : Ajouter une quantité de produit créer pour une commande

Description : ajoute dans la table commande une quantité de produit

Acteur : Usager

Données Entrée : Entrée le nom du produit et le nombre d’heure qui passe à le créer et l’usager.

Scénario Principal :

1. L’usager entre le nom du produit et le nombre d’heure à faire ce produit

Les informations sont ajoutées dans la base de données

Nom : Création d’un nouveau produit

Description : Crée un nouvel Item

Acteur : Usager

Données Entrée : les informations de l’objet : nom, nombre crée par heure, cout et prix.

Scénario Principal :

1. L’usager entre les informations de l’objet et ensuite il fait la confirmation

L’item est ajouté dans la base de données

Nom : Supprimer une ligne dans la base de données

Description : Elle permet de supprimer des informations de la base de données en cas d’erreur

Acteur : Usager

Données Entrée : L’usager clique une ligne et pèse un bouton

Scénario Principal :

1. L’usager fait une demande de supprimer une ligne dans la DB

Option disponible pour les 2 tables

Nom : Modifie une ligne dans la base de données

Description : Elle permet de modifier des informations de la base de données en cas d’erreur

Acteur : Usager

Données Entrée : l’usager clique une ligne et pèse un bouton

Scénario Principal :

1. L’usager fait une demande de modifier une ligne dans la DB

Option disponible pour les 2 tables

Nom : Connexion

Description : Elle permet à l’usager de se connecter avec un compte

Acteur : Usager

Données Entrée : L’usager a choisi l’employer/gérant

Scénario Principal :

1. L’usager se connecte après avoir choisi un usager et avoir cliquer le bouton Connecter

# Diagramme UML des classes

Une image contenant texte, intérieur

Description générée automatiquement

# Difficultés rencontrées

- Modifier et supprimer une ligne de la base de données. Pour une raison que je ne comprends pas les Querys pour supprimer et modifier ne fonctionne pas.

- Les problèmes avec les command SQL qui ne semble pas fonctionner dépendamment de la façon dont le code est écrit parfois.

- Comprendre comment passer des valeurs d’un formulaire à un autre. Si je veux prendre un objet qui a été instancier dans un formulaire dans un autre je dois le déclarer ‘’public static’’

# Ajouts Bonis

Formulaire MDI : car si dans le futur je voudrai ajouter d’autre chose au programme je peux simplement ajouter un onglet pour le nouveau formulaire.

string.Format : car c’est beaucoup plus lisible et plus évident à travailler avec que le typique ‘’…’’+valeur+ ‘’…’’

# Journal du temps de travail

|  |  |  |
| --- | --- | --- |
| **Tâche** | **NB d’heure** | **Cumulatif** |
| Recherche sur le sujet et inspiration | **3** | **3** |
| Description du projet | 1 | 4 |
| Diagramme des cas d’utilisation | 1 | 5 |
| Description des cas d’utilisation | 2 | 7 |
| Diagramme UML des classes | 2 | 9 |
| Codage des classes | 5 | 13 |
| Design de l’interface | 4 | 17 |
| Connecter les classes avec les formulaires | 10 | 27 |
| Ajout sql | 3 | 30 |
| Correction pour le rapport final | 2 | 32 |
| Documentation | 2 | 34 |
| Installation | 1 | 35 |

# Versionnage

<https://github.com/AnthonyHaspect/ProjetSynth-se>

# BIBLIOGRAPHIE

[Lambdas – Microsoft.com dotnet csharp](https://learn.microsoft.com/en-us/dotnet/csharp/language-reference/operators/lambda-expressions?ranMID=43674&ranEAID=FE4O7wtxe6g&ranSiteID=FE4O7wtxe6g-ASqN43X82pXBlgR_eRl98Q&epi=FE4O7wtxe6g-ASqN43X82pXBlgR_eRl98Q&irgwc=1&OCID=AID2200057_aff_7795_1243925&tduid=(ir__gqyzsywcc9kf6ic1i9019b0vj22xchlbje9mzwcq00)(7795)(1243925)(FE4O7wtxe6g-ASqN43X82pXBlgR_eRl98Q)()&irclickid=_gqyzsywcc9kf6ic1i9019b0vj22xchlbje9mzwcq00)