420-440-SF
PROGRAMMATION D'APPLICATIONS
TRAVAIL PRATIQUE #3

- Compte pour 30% de la session.
- Travail à réaliser en équipe de 3.
- 2 remises

## Objectifs pédagogiques

Ce travail vise une intégration des concepts et notions apprises dans le cadre du cours par la réalisation d'une application complète écrite en C++ tout en respectant une démarche de développement structurée. Il mettra également à profit plusieurs technologies facilitant le développement logiciel.

## Contexte

On vous demande de réaliser une **application de votre choix** dans laquelle vous appliquerez toutes les compétences acquises dans le cadre de ce cours.

Votre application peut être un jeu, une application de base de données, une application serveur ou toute autre application qui vous permette de rencontrer les spécifications fournies dans la section suivante (voir « contraintes sur l'application »).

Vous devez également produire la documentation associée à votre projet (tests unitaires).

## Contraintes sur l'application

Vous devrez définir un projet d'envergure en fonction du nombre de personnes dans votre équipe.

Les éléments suivants doivent être intégrés dans votre travail :

- Séparation de la solution en projet : Votre solution doit être divisée en projet afin de séparer les responsabilités.
- Architecture testable unitairement : L'ensemble du projet doit être testé unitairement. Des tests d'intégration doivent être créés pour les classes utilisant une BD ou des fichiers.
- Gestion d'erreurs: votre projet doit gérer correctement ses erreurs; vous devez utiliser les exceptions chaque fois que cela est approprié.
- Conservation de données : L'application doit pouvoir conserver des données sur le support de votre choix (BD, fichier XML, json, fichier texte, etc.)

## Calendrier des remises

| Semaine | Groupe du lundi - mercredi |
|---|---|
| 12 | Mardi 21 avril<br>Révision examen 2<br>**Distribution TP3** |
| | Vendredi 24 avril<br>Examen 2 |
| 13 | Mardi 28 avril<br>*Temps TP*<br>**Validation Projet** |
| | Vendredi 1 mai<br>*Temps TP* |
| 14 | Mardi 5 mai<br>*Temps TP*<br>**Remise 1 - Sprint 0 (23h59)** |
| | Vendredi 8 mai<br>*Temps TP* |
| 15 | Mardi 12 mai<br>*Temps TP* |
| | Vendredi 15 mai<br>*Temps TP* |
| 16 | Mardi 19 mai<br>*Temps TP* |
| | Jeudi 21 mai (horaire du vendredi)<br>*Présentation TP*<br>**Remise final - 23h59** |

→*Toutes les remises doivent comprendre les corrections des remises précédentes.*

## Validation du projet

1. Choisissez votre sujet

2. Faites approuver votre sujet par le professeur. Assurez-vous de choisir un contexte qui respecte les contraintes et qui représente un niveau de difficulté normal (ni trop facile, ni trop difficile).

## Remise 1 – Sprint 0 (20% de la note)

Dans votre projet, sur Pivotal Tracker :

- **Ajoutez** une description de votre projet dans ***Project Profile*** (voir exemple annexe 1 pour exemple de description)
- **Ajoutez** les ***User Stories*** (voir exemple annexe 1)
- **Estimez** la complexité de chacun des ***User Stories***
- **Priorisez** les ***User Stories***

Afin de faciliter la découverte des ***User Stories*** et de mieux cibler les besoins, il est conseillé de faire par exemple, des ébauches de maquettes.

De même, pour le développement du squelette de l'application (premier **User Story)** il peut être utile de faire une première réflexion en créant des diagrammes de classes et de séquences.

**Remise 1**

Sur GitHub :

- Le code doit contenir un ***User Story*** fonctionnel.
- Idéalement, choisir un ***User Story*** qui passe à travers toutes les couches de l'application (exemple de l'interface utilisateur à la BD)

| **Évaluation** | |
|---|---|
| **Squelette de la solution (développement d'un User Story)**<br>Qualité du code<br>Qualité des tests<br>Solution divisée en projet<br>Clarté et simplicité<br>Respect des standards<br>User Story entièrement fonctionnel | 15% |
| **Professionnalisme**<br>Qualité des user stories et de la description sur Pivotal Tracker<br>Le projet est toujours compilable sur GitHub<br>Le code de l'application sur GitHub doit toujours être couvert par des tests | 5% |

## Remise 2 – remise finale (80% de la note)

C'est la dernière version déposée sur GitHub (avant la date et l'heure de remise) qui sera corrigée.

| Évaluation | |
|---|---|
| **Tests unitaires et tests d'intégration**<br>Qualité des tests<br>Couverture de tests<br>Choix et réalisation des jeux d'essais pour les tests d'intégration | 25% |
| **Qualité du code**<br>Noms des méthodes et attributs significatifs<br>Clarté et simplicité<br>Respect des standards<br>Gestion des erreurs | 25% |
| **Fonctionnalités**<br>Le programme est entièrement fonctionnel<br>Respect des contraintes<br>Il n'y a aucune erreur à l'exécution.<br>Il n'y a aucun bogue connu. | 20% |
| **Professionnalisme**<br>Avancement régulier du projet sur GitHub et Pivotal Tracker<br>Qualité des commentaires sur GitHub et Pivotal Tracker<br>Le projet est toujours compilable sur GitHub<br>Le code de l'application sur GitHub doit toujours être couvert par des tests | 10% |

## Complément sur l'évaluation

- Le travail doit être reparti également entre chacun des coéquipiers. Un coéquipier qui participe moins pourra se faire retrancher des points. La participation sera suivie en classe et par les remises GitHub.

- À noter : Si le professeur le juge à propos, toute étudiante ou tout étudiant pourra être convoqué à une rencontre d'évaluation pour vérifier son degré d'acquisition des connaissances et d'appréhension de la solution proposée.

- Tous les biens livrables de chacune des étapes devront être remis à temps et selon les modalités spécifiées.

- Étant donné la nature séquentielle des étapes à réaliser, aucun retard ne pourra être toléré. Toute dérogation sans motif raisonnable sera sujette à une pénalité pouvant aller jusqu'au refus du travail en entier.

- Dans l'éventualité où vous récupériez du code existant ailleurs (Internet, MSDN, etc.), vous devez clairement indiquer la source ainsi que la section de code en question. De plus, vous devez le retravailler pour qu'ils répondent aux normes du cours, le cas échéant. Tout travail plagié ou tout code récupéré d'une source externe et non mentionnée peut entraîner la note zéro (0) pour l'ensemble du travail.

- Conformément à la politique concernant la qualité du français écrit, le travail est sujet à une pénalité pouvant aller jusqu'à 10% du total des points attribuables.

- Finalement, il est important de noter que le niveau de difficulté du projet ainsi que son niveau d'achèvement seront considérés lors de l'évaluation du projet par le professeur.

# Annexe 1

## Exemples de description et de *User Stories*

### Exemple pour un jeu

***Description***

« Super Matou » est un jeu de divertissement pour les jeunes enfants (5 à 10 ans) où l'utilisateur, dans le rôle du matou, doit manger le plus de souris possible dans un laps de temps donné. Le chat est toutes les souris cohabitent dans un même terrain. Les souris tentent de s'enfuir du chat lorsque celui-ci est dans leur champ de vision. Le chat gagne en rapidité lorsqu'il mange une souris. Les souris quant à elle, gagne en rapidité lorsqu'elle mange des morceaux de fromages qui se trouvent dans le terrain.

***User Stories***

- Un joueur peut se déplacer dans le terrain
- Un joueur peut manger toutes les souris avant la fin du compteur
- Un joueur peut aller plus vite en mangeant des souris
- Un joueur peut voir le tableau des pointages
- Un joueur peut choisir un niveau de difficulté
- Le pointage d'un joueur est calculé en fonction du temps et du nombre de souris mangées.
- Une souris peut rechercher les fromages et les manger pour aller plus vite
- Une souris peut s'enfuir du chat lorsqu'il est trop proche
- La partie se termine si le temps est écoulé ou quand toutes les souris ont été mangées
- Le tableau des pointages s'affiche à la fin de la partie

### Exemple pour une application de gestion

***Description***

« GestionPlus » est un système de facturation et de gestion d'inventaire. Les employés accèdent facilement et aisément à toutes les informations pertinentes des produits.

***User Stories***

- L'utilisateur peut créer une facture pour un client
- L'utilisateur peut afficher les informations sur un produit
- L'utilisateur peut ajouter un produit à l'inventaire
- L'utilisateur peut modifier un produit de l'inventaire
- L'utilisateur peut supprimer un produit à l'inventaire