

Objectifs

- Mise en œuvre de la plateforme de développement QtCreator.
- Prise en main des outils DevOps (documentation du code, contrôle de version, formatage du code, Gitlab)
- Connaître les documents (CDC, DDC, PDP, DDV) du projet.
- Réalisation de l'itération 1 de l'application → Création d'un menu et gestion du choix de l'utilisateur

Durée 6 heures

I. Mise en place de l'environnement de développement QtCreator

Tâche1. À l'aide du projet modèle fourni, configurer votre environnement de développement.

Le modèle est disponible à l'adresse suivante :

<https://forge.apps.education.fr/gestionlogsed/modele-projet-gestionlog>

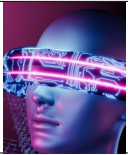
- Si nécessaire, suivre les instructions sur le [site de ressources](#) pour prendre en main et configurer QtCreator pour un modèle de projet fourni.

II. Prise en main du projet

Tâche2. Lire les documents de conception du projet :

- Cahier des charges de l'application (GestionLog_CDC.pdf)
- Dossier de conception de l'application (GestionLog_DDC.pdf)
- Planning de développement de l'application (GestionLog_PDP.pdf)
- Dossier de Vérification de l'application (GestionLog_DDV.pdf)

Les documents de conception sont votre référence en ce qui concerne la réalisation de ce projet. Vous devrez vous y référer régulièrement tout au long du projet.



III. Itération 1 (Création du menu et gestion du choix de l'utilisateur)

Vous allez réaliser ce programme en suivant les étapes ci-après.

Pour chacune des étapes, vous disposez de l'exécutable dans le dossier **ExecutablesIteration1** pour vérifier le résultat à obtenir. (les programmes s'exécutent dans le terminal de votre ordinateur)

III.1 Procédure d'exécution d'un programme dans le terminal

- Se placer dans le dossier `ExecutablesIteration1`

```
cd ExecutablesIteration1
```

- Exécuter le programme désiré en indiquant son nom précédé de `./`

Exemple

```
./etape3
```

```
s/3-Iterations/ExecutablesIteration1$ ./etape3
CIEL - Gestion centralisée de logs
Menu
Choisir une option
1 - Afficher log sudo
2 - Afficher et enregistrer log ssh
3 - pocoGetAllLog
4 - pocoGetOneLog
5 - pocoPostSSHLog
0 - Sortir du programme
█
```

Figure 1

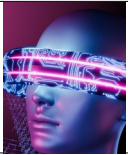
Illustration 1: Exécution d'une des étapes dans le terminal

III.2 Étape 1

III.2.1 Afficher " CIEL– Gestion Centralisée de logs "

Conseil → Utiliser la fonction `cout`

Q1 - Comment avez-vous réalisé l'étape 1 ?



III.2.2 Documentation

Tâche3. Suivre les instructions sur le site [de ressources](#) pour documenter correctement votre code

Résultats attendus :

- ✓ Le fichier `main.cpp` doit être correctement documenté
- ✓ La fonction `main()` doit être correctement documentée

III.2.3 Sauvegarde de votre travail

Tâche4. Suivre les instructions [Git premiers pas](#) (parties 1 et 2) pour sauvegarder votre travail sur le serveur Gitlab de la section

Résultats attendus :

- ✓ Un dépôt `<VOTRE_NOM>_GESTION_LOG` doit être configuré sur le serveur de la section
- ✓ Votre travail (**liste fichiers ci-dessous**) doit être sauvegardé sur ce dépôt
 - `CMakeLists.txt`
 - `Doxyfile`
 - `main.cpp`
 - `readme.md`
 - `.gitignore`
 - `.gitlab-ci.yml`

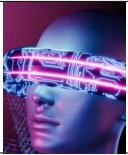
III.2.4 Validation

Appeler l'enseignant pour qu'il valide votre travail

III.3 Étape 2 – Afficher Le menu

Conseil → Utiliser la fonction `cout` et `endl`

Q2 - Comment avez-vous réalisé l'étape 2 ?



III.4 Étape 3 – Demander à l'utilisateur quel choix il souhaite sélectionner et écrire son choix à l'écran

Conseil → Utiliser les fonctions `cout` et `cin`

Conseil → Utiliser une variable de type entier

Q3 - Quel type de variable C++ avez-vous utilisé ?

Q4 - Quel nom avez-vous donné à votre variable ?

III.5 Étape 4 – En fonction du choix de l'utilisateur, afficher l'option désirée

Conseil → Utiliser l'alternative `if()...else`

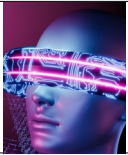
Q5 - Où placez-vous la condition correspondant au choix de l'utilisateur dans le `if()...else` ?

Q6 - Quel est le rôle des accolades ?

III.6 Étape 5 – Traiter le cas où l'utilisateur choisit une option invalide par un message d'erreur.

Conseil → Utiliser l'alternative `if()...else`

Q7 - Qu'avez-vous ajouté à l'étape précédente pour traiter le cas d'erreur ?



IV. Outils DevOps

Les outils DevOps recommandés sur ce module (et tous les autres) sont disponibles sur le site de ressource.

Vous avez déjà réalisé une partie du travail ci-après lors de la réalisation de l'itération 1.

Cependant, avant toute remise de livrable, vous devrez vérifier la qualité de votre travail et notamment respecter les standards demandés, concernant le formatage, la documentation et le versionning de votre code.

IV.1 Formatage du code

Tâche5. Suivre la page [Standard de codage](#) pour formater correctement votre code

Résultats attendus :

- ✓ Le code doit être correctement indenté
- ✓ Les noms de variables doivent respecter la convention de nommage

IV.2 Documentation du code

Tâche6. Suivre la page [Documenter son code](#) pour documenter correctement votre code

Résultats attendus :

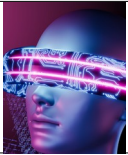
- ✓ Le fichier `main.cpp` doit être correctement documenté
- ✓ La fonction `main()` doit être correctement documentée
- ✓ Les points de fonctionnement délicat du programme doivent être explicités

IV.3 Sauvegarde de votre travail

Tâche7. Suivre la page [Git Workflow basique](#) pour sauvegarder votre travail sur le serveur GitLab de la section. Vous devrez créer un tag pour votre TP.

Résultats attendus :

- ✓ Un dépôt `<VOTRE_NOM>_GESTION_LOG` doit être configuré sur le serveur de la section
- ✓ Un **tag v1.0** doit être créé sur votre dépôt
- ✓ Votre travail (**liste fichiers ci-dessous**) doit être sauvegardé sur ce dépôt
 - `CMakeLists.txt`
 - `Doxyfile`
 - `main.cpp`
 - `readme.md`
 - `.gitignore`
 - `.gitlab-ci.yml`



IV.4 Validation

Appeler l'enseignant pour qu'il valide votre travail

V. Livrable

Sur le serveur Moodle de la section

→ Votre compte rendu de TP avec les réponses aux questions de celui-ci

Sur le serveur Gitlab de la section

→ Le projet <VOTRE_NOM>_Gestion_Log, avec le tag v1.0