


## 1 INFORMATIONS GENERALES

Apprenti(e) 1 :	Nom:	Prénom:
Apprenti(e) 2 :	Nom:	Prénom:
Lieu de travail :	ETML / Sébeillon 12 / 1004 Lausanne	
Client :	Nom: Melly	Prénom: JONATHAN
	 jonathan.melly@vd.educanet2.ch	
Dates de réalisation :	Semaine 3 à semaine 22 (17 semaines hors vacances)	
Horaire de travail :	4 périodes par semaine	
Temps total :	68 périodes au maximum	

## 2 PROCÉDURE

- Tous les apprentis réalisent le projet sur la base d'un cahier des charges.
- Le cahier des charges est présenté, commenté et discuté en classe.
- Les apprentis sont entièrement responsables de la sécurité et sauvegarde de leurs données.
- En cas de problèmes graves, les apprentis avertissent le client au plus vite.
- Les apprentis ont la possibilité d'obtenir de l'aide externe, mais ils doivent le mentionner.
- Les informations utiles à l'évaluation de ce projet sont disponibles au chapitre 8.

## 3 TITRE



### *Smart Thésaurus*

## 4 SUJET

On désire trouver plus facilement les informations liées à l'activité à l'ETML. Ainsi vous avez la mission de réaliser un programme qui va indexer les contenus suivants :

1. K:\inf\élèves\temp
2. Site web de l'etml
3. Serveur FTP (optionnel)

Les résultats de l'indexation seront utilisables au moyen d'une interface fonctionnant par mots clé (moteur de recherche)

## 5 MATÉRIEL ET LOGICIEL À DISPOSITION

- Un PC ETML avec Visual studio community
- Accès à Internet
- Répertoire de stockage sur le réseau (H :)

---

## 6 PRÉREQUIS

Modules 226 et 120. Modules 326 et 318 en parallèle.

---

## 7 CAHIER DES CHARGES

### 7.1 Objectif et portée du projet (objectifs SMART)

1. Réaliser un programme informatique de qualité
  - UML (classe, activité, séquence, cas d'utilisation, ...)
  - MCD pour les données
  - Design patterns (mvc, template, ...)
  - Tests unitaires et d'intégration
  - Documentation pertinente
2. Prouver que vous êtes digne de confiance lorsqu'on vous confie un projet
  - Planification (à jour)
  - Journal de travail
    - Format ETML ou commit log
  - Pro-activité
    - Poser des questions au client
    - Faire des démonstrations
    - Utiliser un système de versioning de code

### 7.2 Fonctionnalités requises (du point de vue de client)

1. Indexation
  - Sources possibles
    - K
    - Site web
    - Serveur FTP (optionnel)
  - Choix des éléments à indexer
    - Fichiers textes
    - Documents Office et OpenOffice
    - Documents PDF
    - Images (juste le nom et la référence, contenu optionnel)
  - Stockage des résultats d'indexation à choix entre
    - Base de données (SQLite, MariaDB, MongoDB)
    - Fichier CSV / XML
    - RAM

- Choix du mode de mise à jour
  - Manuel
  - Chaque heure
- 2. Recherche
  - Saisie des critères de recherche
    - Support des opérateurs + et – (comme dans Google)
  - Affichage des résultats par priorité de pertinence (meilleure correspondance)
    - Accès direct à la ressource originale (clic)
- 3. Documentation
  - Installation (y.c. matériel requis)
  - Utilisation

### 7.3 Travail à réaliser par les apprentis dans le cadre de ce projet

1. Créer et maintenir une planification et un journal de travail selon le modèle ETML
2. Concevoir le logiciel à l'aide de
  - a. Schémas UML (classe, activité, cas d'utilisation, séquence)
  - b. Réflexions sur le fonctionnement du logiciel → Schémas et choix pertinents
  - c. Petits programmes brouillon pour valider/invalidier les idées
3. Implémenter le logiciel à l'aide de C#
  - a. En appliquant le paradigme OO
  - b. En utilisant les patterns appropriés (template, factory, ...)
  - c. En protégeant le code avec des tests unitaires
  - d. En validant la qualité du produit global avec des tests d'intégration
  - e. En réalisant un installateur
4. Rédiger la documentation
  - a. Organisationnelle :
    - i. Planning et (journal au format etml ou commit log)
    - ii. Bilans
      1. Planification (différence prévu / réalité)
      2. Fonctionnalités (effectué / en cours / non réalisé)
      3. Personnels (qu'avez-vous appris)
  - b. Technique
    - i. Schémas/explications pour les développeurs
    - ii. Rapports de tests
  - c. D'installation et d'utilisation

## 7.4 Livrables

1. Planification et journal de travail (tenu à jour chaque semaine avec l'état d'avancement et le temps restant)
2. 1 archive au format **zip** contenant
  - a. Documentation (**1 pdf global**), regroupant les éléments décrits au point 7.3-4
  - b. Code source
  - c. Installateur
  - d. Auto-évaluation finale (dernière étape avant la livraison) **avec un commentaire factuel et précis sur chaque point** de cette dernière

## 8 Évaluation

1. 50% de l'auto-évaluation (pour autant que vos observations soient factuelles)
2. 50% de la grille suivante :

Objectifs et comportements observés	Poids	Révélateurs attendus
<b>Organisation / Comportement</b>	<b>1</b>	<b>Organisation / Comportement</b>
Livraison	3	Archive complète et livrée à temps
Planning	1	Dates correctes, tâches claires et suivi
Journal de travail	2	Dates correctes, facile à suivre/reprendre (statut et détails), liens et aide mentionnés
Doc : UML et MCD	2	Schémas corrects et pertinents pour comprendre le programme
Doc : tests	2	Liste des tests réalisés sur la version finale
Doc : manuel	1	Décrit l'installation et l'utilisation du programme
Doc : bilan	1	Reprend les écarts planifiés / réalisés et décrit les apprentissages réalisés
Doc : autoéval	1	Commentaires pour chaque étape
Pro-activité	2	Agit rapidement en cas de questionnement / inquiétude sensée
<b>Fonctionnalités</b>	<b>2</b>	<b>Fonctionnalités</b>
Indexation	2	K, site web et classeur educanet2
Indexation : maj	1	Manuel, quotidien, chaque heure et personnalisé
Recherche : critères	2	Saisie multicritères
Recherche : opérateurs	1	Opérateurs + et -
Recherche : résultats - > tri	1	Tri par pertinence (document où le mot apparaît le plus souvent)
Recherche : résultats - > clic	1	Clic pour accéder à la ressource
Installateur	1	Installation assistée
<b>Qualité du code</b>	<b>1</b>	<b>Qualité du code</b>
POO	1	Application découpées en classes fonctionnelles (pas de duplication de code ou 'if' inutile) respectant les principes OO (encapsulation, ...)
Designs patterns	1	Utilisés à bon escient (mvc, ...)
Conventions	2	Respect des conventions ETML (nommage, entêtes, ...)
Tests unitaires	1	Couverture à 100% des scénarios extrapolés à partir du cdc