|  |
| --- |
| Récupération des AutoEvaluations |

Table des matières

[1 Analyse préliminaire 3](#_Toc129531868)

[1.1 Introduction ✔ 3](#_Toc129531869)

[1.2 Objectifs✔ 4](#_Toc129531870)

[1.3 Planification initiale ✔ 4](#_Toc129531871)

[2 Analyse / Conception 4](#_Toc129531872)

[2.1 Concept 4](#_Toc129531873)

[2.2 Gestion de projet 5](#_Toc129531874)

[2.2.1 Méthodologie de travail ✔ 5](#_Toc129531875)

[2.2.2 Outil de versioning 6](#_Toc129531876)

[2.3 Stratégie de test 6](#_Toc129531877)

[2.4 Risques techniques 7](#_Toc129531878)

[2.5 Planification 7](#_Toc129531879)

[2.6 Dossier de conception✔ 7](#_Toc129531880)

[3 Réalisation 10](#_Toc129531881)

[3.1 Dossier de réalisation 10](#_Toc129531882)

[3.1.1 Environnement✔ 10](#_Toc129531883)

[3.1.2 Version du projet✔ 10](#_Toc129531884)

[3.1.3 Fichiers de configurations✔ 10](#_Toc129531885)

[3.1.4 Fichiers de scripts 13](#_Toc129531886)

[3.2 Description des tests effectués 16](#_Toc129531887)

[3.3 Erreurs restantes 17](#_Toc129531888)

[3.4 Liste des documents fournis 17](#_Toc129531889)

[4 Conclusions 17](#_Toc129531890)

[5 Annexes 18](#_Toc129531891)

[5.1 Résumé du rapport du TPI / version succincte de la documentation 18](#_Toc129531892)

[5.2 Sources – Bibliographie 18](#_Toc129531893)

[5.3 Journal de travail 18](#_Toc129531894)

[5.4 Manuel de modifications✔ 18](#_Toc129531895)

[5.4.1 Modification des scripts✔ 18](#_Toc129531896)

[5.4.2 Compilation des scripts✔ 19](#_Toc129531897)

[5.4.3 Modifications du modèle d’auto-évaluations✔ 20](#_Toc129531898)

[5.5 Manuel d'Utilisation ✔ 25](#_Toc129531899)

[5.5.1 Téléchargement✔ 25](#_Toc129531900)

[5.5.2 Configurations✔ 26](#_Toc129531901)

[5.5.3 Création des auto-évaluations✔ 27](#_Toc129531902)

[5.5.4 Rapatriement des auto-évaluations✔ 28](#_Toc129531903)

[5.6 Archives du projet 31](#_Toc129531904)

# Analyse préliminaire

## Introduction ✔

Actuellement les enseignants en informatique utilisent un système d’évaluation des projets d’élèves, qui a été réalisé avec Excel plus des macros. Ce système offre une fonctionnalité d’exportation des onglets des élèves dans des fichiers Excel sans macro, pour leur permettre de s’auto-évaluer.

Toutes ces manipulations sont effectuées à la main et prennent beaucoup de temps. Cela est donc propice aux erreurs notamment lors du rapatriement des évaluations qui est fait en copier-coller.

Le but de ce projet, est de proposer une solution d’automatisation la plus complète possible, afin de minimiser le travail requis ainsi que les erreurs.

La technologie qui a été choisie est PowerShell. Car étant un langage de script natif à Windows, les manipulations afin d’installer le projet sont minimes. De plus, le langage étant développé par Microsoft, il y’a de nombreux composants natifs permettant d’interagir avec les différents programmes / processus Windows. Notamment Microsoft Excel.

Le projet se déroule dans le même cadre que lors d’un TPI. L’environnement logiciel et matériel est le suivant : 1 PC standard de l’ETML, Visual Studio Code et un dépôt GitHub.

Les différents modules qui seront utiles au projet sont les suivants : 122-ScriptsMacros, 158-GererMigrationLogiciels, 226-Prog-ImplementerOO, 302-BureautiqueAvancee, 306-Projet-RealiserPetitProjet, 403-Prog-Structuree, 404-Prog-BaseeObjets

La méthodologie de projet qui sera utilisée, est la méthode des 6 pas. La taille du projet étant assez légère une méthode agile est trop volumineuse à mettre en place.

Le projet s’effectuant seul, la méthode des 6 pas parait une des plus simple à utiliser. De plus c’est une méthode que nous avons vu en module. Nous sommes donc à l’aise dans son utilisation.

## Objectifs✔

Les objectifs techniques du projet définis dans le cahier des charges, sont les suivants :

1. Les valeurs proposées par les élèves sont récupérées correctement

2. Les commentaires des élèves sont récupérés correctement

3. Toutes les valeurs des indicateurs sont récupérées correctement

4. Le dispositif est pertinent et simple à utiliser

5. Le dispositif est fiable. En cas d’incidents, les messages d’erreurs sont pertinents

6. Une modification de la structure de la grille est possible

7. Fonctionne sans autorisation spéciale, ni configuration exigeant l’intervention d’un spécialiste système.

## Planification initiale ✔



La planification initiale du projet a été élaborée en suivant méthode des 6 pas. Ainsi, le projet a été découpé en semaines, chacune de ces dernières étant subdivisée en quatre jours de travail. Ce qui correspond aux jours de projet.

# Analyse / Conception

## Concept

*Le concept complet avec toutes ses annexes:*

*Par exemple :*

* *Multimédia: carte de site, maquettes papier, story board préliminaire, …*
* *Bases de données: interfaces graphiques, modèle conceptuel.*
* *Programmation: interfaces graphiques, maquettes, analyse fonctionnelle…*
* *…*

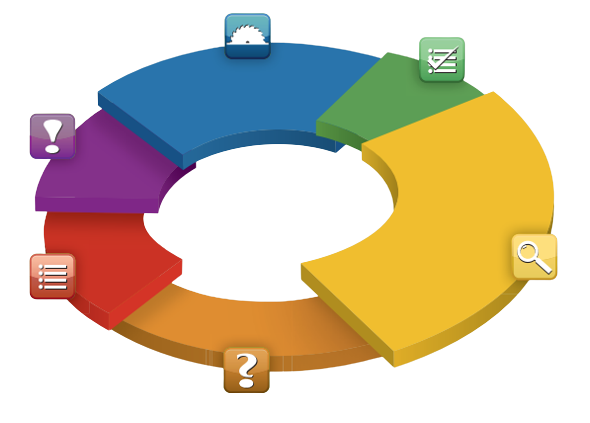
## Gestion de projet

### Méthodologie de travail ✔

La méthodologie que nous avons utilisé pour ce projet se nomme "La méthode des 6 pas/étapes".

Nous avons choisi cette méthode, car elle est relativement simple à mettre en place et en vue de l’envergure du projet et du nombre de personnes impliquées elle est efficace. De plus nous avons beaucoup utilisé cette méthode lors des différents projets tout au long de notre formation.

Voici en quoi consiste cette méthode.





#### 1. S'informer

La première étape consiste à comprendre et à enregistrer la tâche, à bien cerner le sujet et à pouvoir se faire une idée de l’objectif. Se procurer systématiquement des informations est, à ce stade, d’une

Importance capitale.

#### 2. Planifier

La planification est le moyen d’arriver au but. Pour cela, il faut l’imagination nécessaire pour mettre de l’ordre dans les opérations. Sur le plan du travail, il convient de choisir les outils et moyens auxiliaires, d’estimer les temps nécessaires et de fixer, pour les travaux en équipe, la répartition des tâches ainsi que les critères de contrôle et d’évaluation.

#### 3. Décider

Lorsque l’information et la planification ont été effectuées de manière optimale, il est plus difficile, en règle générale de prendre une décision. Lorsqu’on retarde des décisions, cela coûte beaucoup de temps et au pire des cas, il n’y a plus rien à décider, parce qu’on se trouve confronté à des contraintes matérielles. En équipe ou en petit groupe, il faudrait toujours, dans la mesure du possible prendre des décisions consensuelles (droit reposant sur un consensus qui a mis d’accord la grande majorité).

#### Une image contenant texte, outil Description générée automatiquement4. Réaliser

La réalisation ou exécution peut représenter la partie principale d’une tâche, comme c’est le cas pour les tâches de production, ou ne représenter qu’une petite partie de cette dernière lorsqu’il s’agit, par exemple, de l’organisation d’une manifestation. Le plan de travail établi au cours de l’étape de planification doit en principe être respecté et ne devrait pas être modifié à la légère.

#### 5. Contrôler

Chaque travail effectué doit être contrôlé avant d’être confié à des tiers. Contrôler signifie, par exemple, relire encore une fois, recalculer, comparer avec les prescriptions du cahier des charges, mesurer, estimer, etc. Un contrôle apporte de bons résultats lorsqu’il est effectué d’une autre manière ou par une personne différente de celles qui ont été employées pour faire le travail.

#### Une image contenant texte, clipart Description générée automatiquement6. Évaluer

Au moment de l’évaluation, nous revoyons éventuellement avec l’employeur la tâche exécutée. A cet effet, on passe encore une fois en revue le déroulement des opérations et on cherche à savoir ce qui a bien fonctionné et ce qui s’est moins bien passé et pourrait être amélioré la prochaine fois.

### Outil de versioning

## Stratégie de test

*Décrire la stratégie globale de test:*

* *types de des tests et ordre dans lequel ils seront effectués.*
* *les moyens à mettre en œuvre.*
* *couverture des tests (tests exhaustifs ou non, si non, pourquoi ?).*
* *données de test à prévoir (données réelles ?).*
* *les testeurs extérieurs éventuels.*

## Risques techniques

* *risques techniques (complexité, manque de compétences, …).*

*Décrire aussi quelles solutions ont été appliquées pour réduire les risques (priorités, formation, actions, …).*

## Planification

*Révision de la planification initiale du projet :*

* *planning indiquant les dates de début et de fin du projet ainsi que le découpage connu des diverses phases.*
* *partage des tâches en cas de travail à plusieurs.*

*Il s’agit en principe de la planification* ***définitive du projet****. Elle peut être ensuite affinée (découpage des tâches). Si les délais doivent être ensuite modifiés, le responsable de projet doit être avisé, et les raisons doivent être expliquées dans l’historique.*

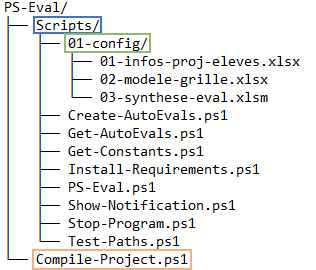
## Dossier de conception✔

Pour réaliser ce projet, nous avons utilisé un ordinateur standard de l’ETML sous le système d’exploitation Windows 10 Éducation. De plus nous avons utilisé le logiciel Visual Studio Code avec l’extension PowerShell pour le développement.

Pour l’utilisation du projet, il n’y aura pas besoin d’outils logiciels autre que ceux installés par défaut sur les PC. (Office365, explorateur de fichier, etc.)

La structure de fichiers se compose de la manière suivante :

* Un dossier ‘’Scripts‘’ contenant tous les scripts nécessaires ainsi qu’un dossier ‘’01-config‘’ contenant les fichiers de configuration et de modèle.
* Un fichier ‘’Compile-Project.ps1‘’ permettant de compiler le projet.



Voici la structure de code du projet :

* **PS-eval.ps1 :** affiche une interface graphique, puis permet à l’utilisateur de lancer la création et le rapatriement des auto-évaluations. Une notification est affichée pour avertir que les scripts peuvent prendre un certain temps à s’exécuter.

Pour la création, le script appelé est ‘’Create-AutoEvals.ps1 ‘’

Pour le rapatriement, le script appelé est ‘’Get-AutoEvals.ps1‘’

Pour l’affichage de la notification, le script appelé est ‘’Show-Notification.ps1‘’

* **Show-Notification.ps1 :** affiche une notification Windows pour l’utilisateur avec un message défini en argument.
* **Create-AutoEvals.ps1 :** crée les auto-évaluations des élèves selon les données spécifiées dans le fichier ‘’01-infos-proj-eleves.xlsx‘’ ainsi que selon le modèle ‘’02-modele-grille.xlsx‘’.

Pour installer les modules nécessaires, le script appelé est ‘’Install-Requirements.ps1‘’

Pour vérifier que les fichiers nécessaires existent bien, le script appelé est ‘’Test-Paths.ps1‘’

Pour obtenir les constantes, le script appelé est ‘’Get-Constants.ps1‘’

Pour arrêter le programme en cas d’erreur, le script appelé est ‘’ Stop-Program.ps1‘’

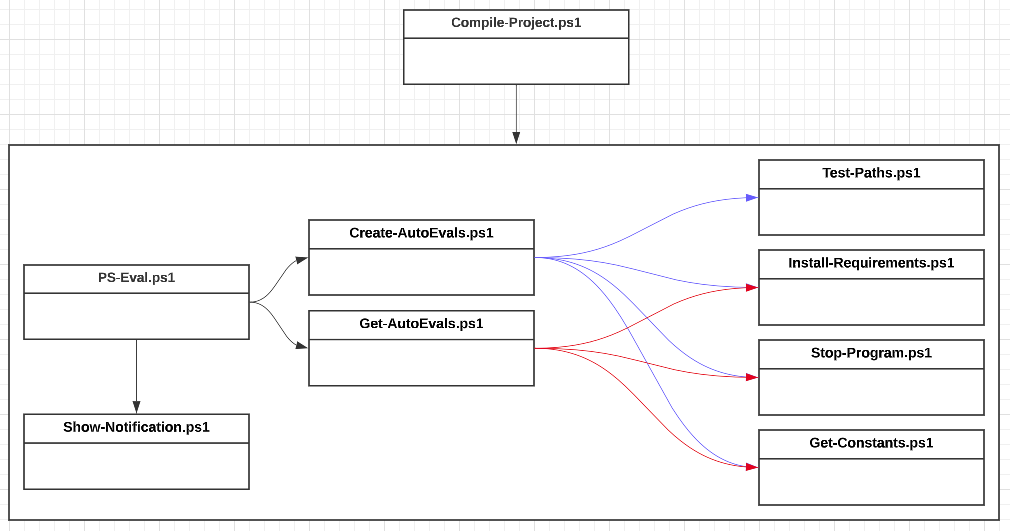
* **Get-AutoEvals.ps1 :** rapatrie les auto-évaluations qui ont été remplies par les élèves. Puis ce script crée une synthèse des évaluations dans un nouveau fichier.

Pour installer les modules nécessaires, le script appelé est ‘’Install-Requirements.ps1‘’

Pour obtenir les constantes, le script appelé est ‘’Get-Constants.ps1‘’

Pour arrêter le programme en cas d’erreur, le script appelé est ‘’ Stop-Program.ps1‘’

* **Test-Paths.ps1 :** Teste si plusieurs chemins de fichiers / dossiers donnés en arguments existent. Retourne une liste avec tous les fichiers / dossiers non existants.
* **Install-Requirements.ps1 :** Installe automatiquement les modules nécessaires au bon fonctionnement des scripts
* **Stop-Program.ps1 :** Arrête l’exécution du script qui l’appelle et retourne une erreur spécifiée en argument
* **Get-Constants.ps1 :** Contient toutes les constantes, et les retourne en tant qu’objet.
* **Compile-Project.ps1 :** Récupère le contenu de tous les scripts mentionnés ci-dessus, et les compile en un seul. Cela permet d’utiliser le projet avec plus de facilitée.



# Réalisation

## Dossier de réalisation

### Environnement✔

Pour ce projet, nous avions à disposition un ordinateur standard de l’ETML. Les spécifications sont les suivantes :

* Windows 10 Education 21H2 64 Bits
* Intel i7-11700 2.50Ghz
* RAM 32 Gb 3200 MHz

L’environnement logiciel est le suivant :

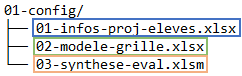
* Visual Studio Code, Version 1.76.1
* GitHub Desktop, Version 3.2.0
* PowerShell, Version 5.1.19041.2364

### Version du projet✔

Ce projet est en version 1.0. Car nous sommes partis de 0 pour le réaliser.

### Fichiers de configurations✔

Voici les différents fichiers de configuration que nous allons aborder :

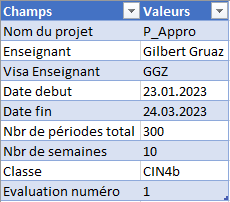
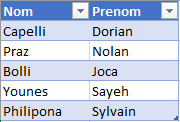


**01-infos-proj-eleves.xlsx :**

Ce fichier contient la liste des élèves ainsi que les différentes informations du projet.

Les champs doivent être remplis par l’enseignant avant de lancer création des auto-évaluations.

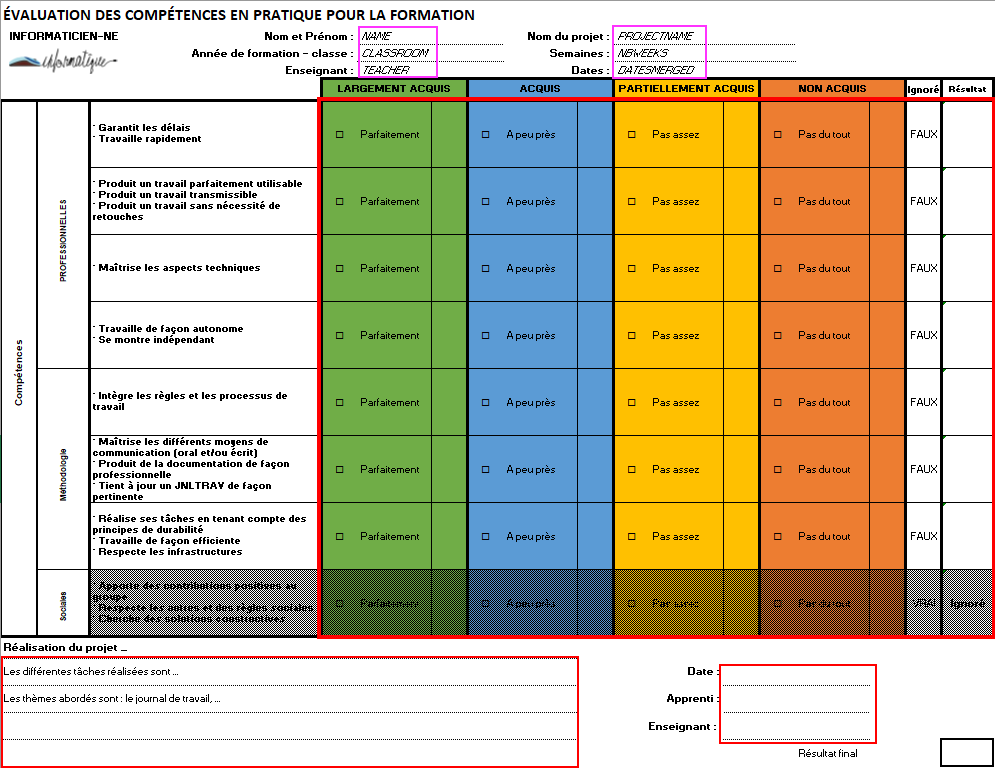
Ces valeurs seront ensuite utilisées pour la génération des auto-évaluations.



**02-modele-grille.xlsx :**

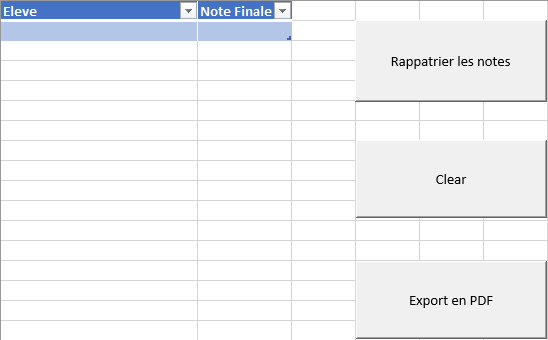
Ce fichier est le modèle d’auto-évaluations qui est actuellement utilisé par les enseignants. Les données de titre (en rose) sont les champs qui seront remplacés par les valeurs définies dans le fichier ‘’01-infos-proj-eleves.xlsx‘’

La feuille Excel est verrouillée contre les modifications, sauf les cellules encadrées en rouge



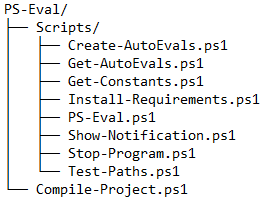
**03-synthese-eval.xlsm :**

Ce fichier est le modèle de synthèse des notes. Une fois les auto-évaluations rapatriées, il contient un onglet par évaluation. De plus, dans l’onglet ‘’MASTER‘’, se trouvent trois boutons permettant respectivement de : Rapatrier les notes dans le tableau, vider le tableau et exporter les auto-évaluations en PDF.



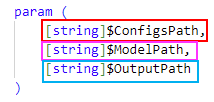
### Fichiers de scripts

Voici les différents fichiers de script que nous allons aborder :



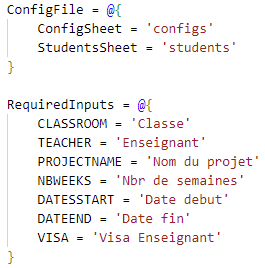
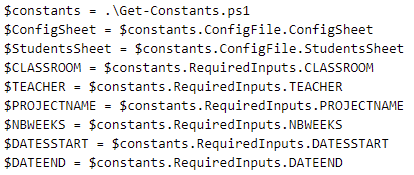
#### Create-AutoEvals.ps1✔

Ce script permet de créer les auto-évaluations des élèves selon un fichier de configuration et un modèle. Ces auto-évaluations seront créées dans un répertoire. Ces fichiers sont définis en arguments.



Toutes les constantes nécessaires au script sont récupérées depuis le fichier

‘’Get-Constants.ps1‘’.



Le script test ensuite ces différents points :

* Vérifie que le dossier de sortie existe et sinon le crée.
* Vérifie que le fichier de configuration et de modèle existe
* Vérifie que le fichier de configuration contient bien tous les champs nécessaires

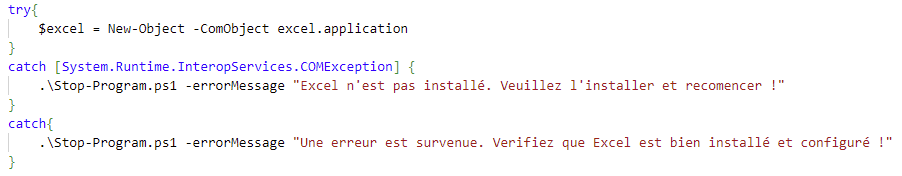
Les configurations et la liste des élèves sont récupérés à l’aide du module ImportExcel.



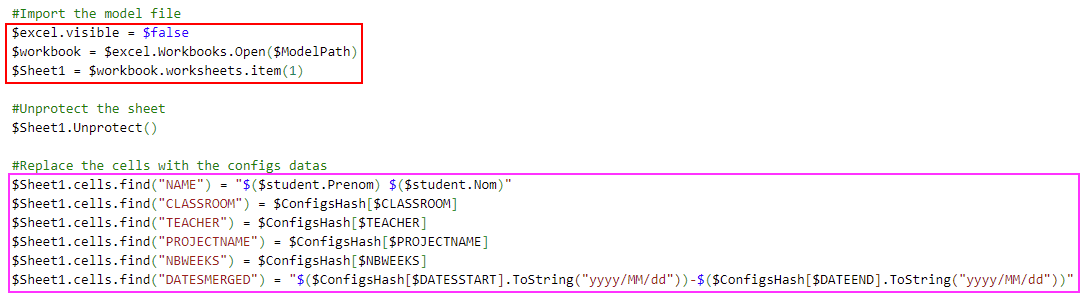
Une boucle ‘’Foreach‘’ parcourt ensuite la liste des élèves pour créer leurs auto-évaluations.

Le ‘’ComObject‘’ Excel est instancié pour pouvoir interagir avec les fichiers Excel.

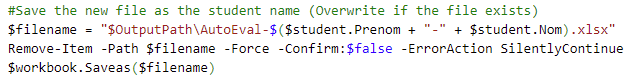
Une erreur est générée si Excel n’est pas installé ou configuré.



Le fichier modèle est importé, puis les cellules de titre sont modifiées avec les valeurs du fichier de configuration.

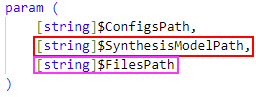


Les fichiers des auto-évaluations sont enregistrés au format ‘’AutoEval-Prenom-Nom.xlsx‘’



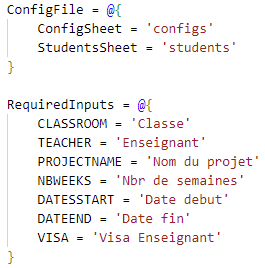
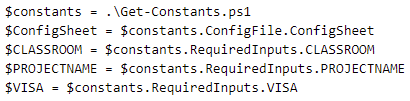
#### Get-AutoEvals.ps1✔

Ce script rapatrie les auto-évaluations remplies par les élèves, puis crée une synthèse des notes selon le modèle. Ces auto-évaluations sont récupérées d’un répertoire défini en arguments



Toutes les constantes nécessaires au script sont récupérées depuis le fichier

‘’Get-Constants.ps1‘’.



Le script test ensuite ces différents points :

* Vérifie que le dossier contenant les auto-évaluations existe.
* Vérifie que le fichier de configuration et de modèle existe
* Vérifie que le dossier contenant les auto-évaluations en contient au moins une

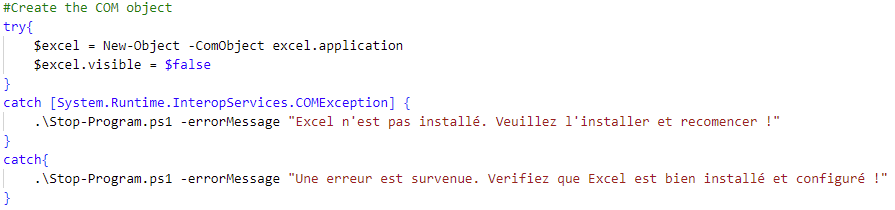
Les configurations sont récupérées à l’aide du module ImportExcel.



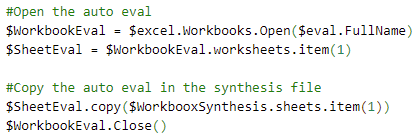
Une boucle ‘’Foreach‘’ parcourt ensuite les fichiers du dossier contenant les auto-évaluations pour les rapatrier.

Le ‘’ComObject‘’ Excel est instancié pour pouvoir interagir avec les fichiers Excel.

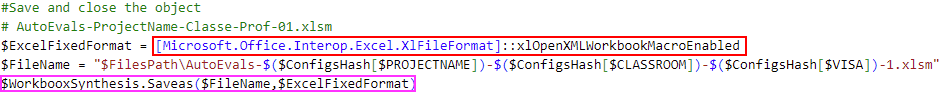
Une erreur est générée si Excel n’est pas installé ou configuré.



Tous les fichiers d’auto-évaluations sont ajoutés au modèle de synthèse en tant qu’onglet Excel.



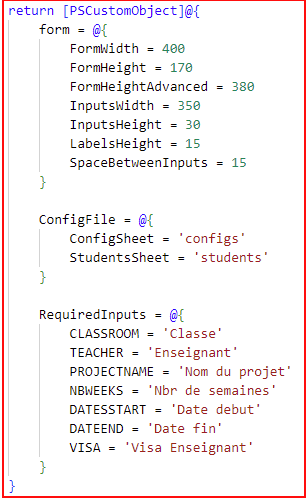
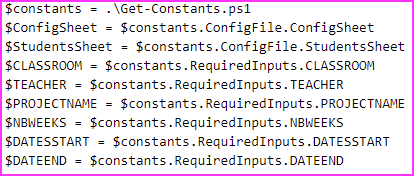
Une fois cette opération faite pour tous les fichiers, le modèle est SaveAs en tant que fichier ‘’xlsm‘’. C’est un fichier Excel prenant en charge les macros. Le nom du fichier est le suivant : ‘’AutoEvals-NomProjet-Classe-Prof-1.xlsm‘’



#### Get-Constants.ps1 ✔

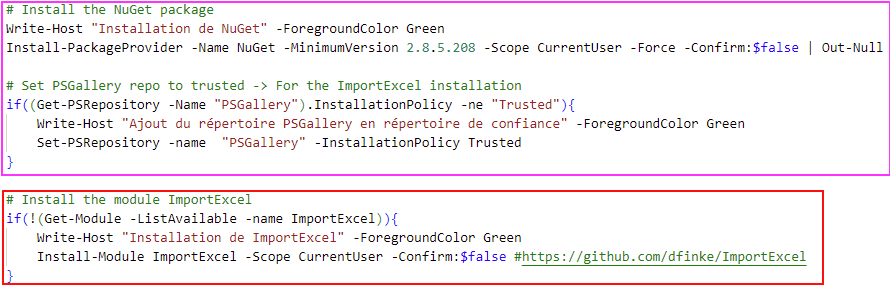
Ce script contient toutes les constantes nécessaires au fonctionnement du projet.

Il retourne un Object PowerShell qui sera accédé par les scripts l’appelant.

#### Install-Requirements.ps1 ✔

Ce script permet d’automatiser l’installation des modules et autres installations requises. Un test est effectué pour voir si le module est installé et si ce n’est pas le cas, on l’installe. Pareil pour le répertoire ‘’PSGallery‘’.

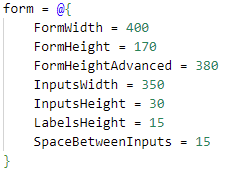
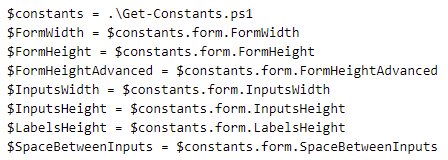


#### PS-Eval.ps1 ✔

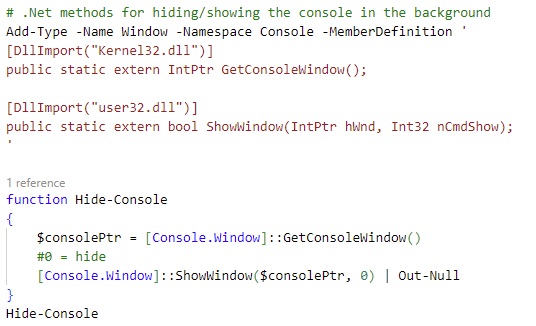
Ce script est la base du projet. Il affiche une interface graphique en Windows Forms afin de permettre à l’utilisateur de créer et rapatrier les auto-évaluations.

Toutes les constantes nécessaires au script sont récupérées depuis le fichier

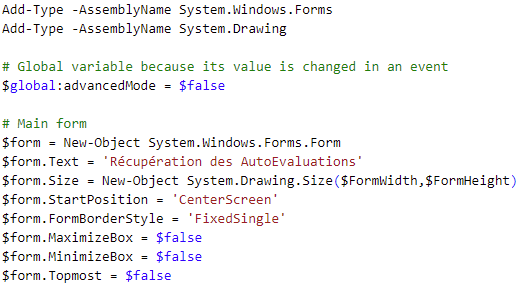
‘’Get-Constants.ps1‘’.



La console PowerShell est ensuite masquée, pour avoir uniquement le formulaire affiché à l’utilisateur

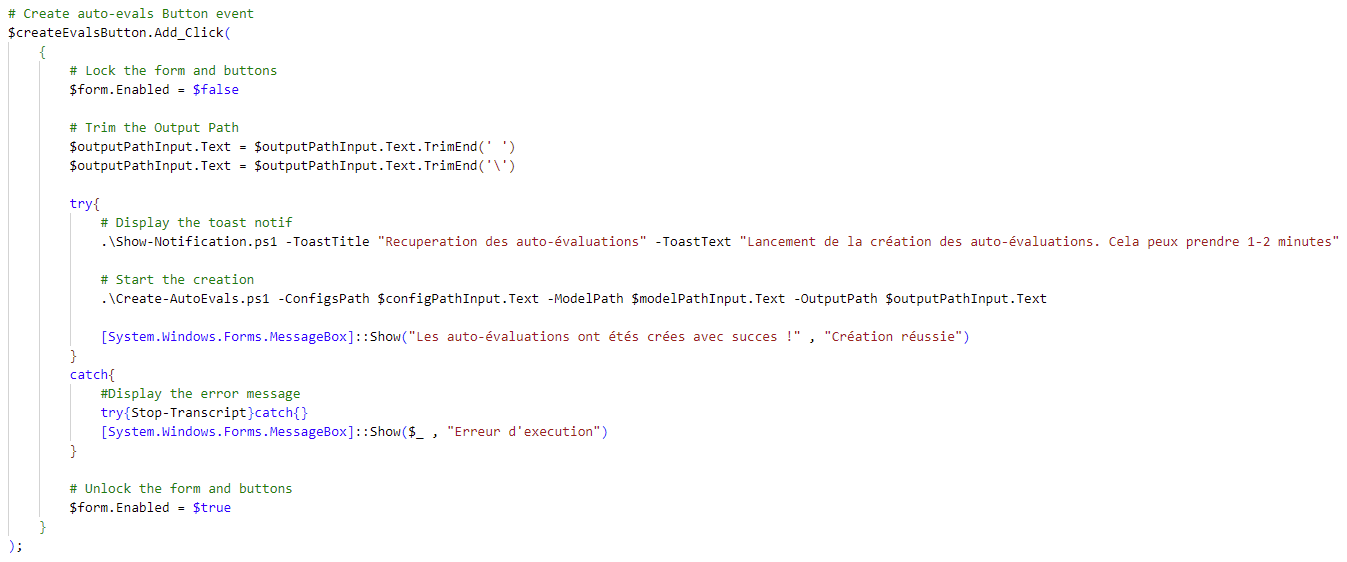


Le formulaire est crée en Windows Forms



Sur les boutons de Création et de Rapatriement, un évènement ‘’Add\_Click‘’ est ajouté. Cet évènement s’exécute lors d’un clic sur le bouton.

Ensuite, le script de création ou de rapatriement est lancé. Si une erreur est retournée, on l’affiche avec une MessageBox



#### Show-Notification.ps1✔

Ce script affiche une notification Windows avec le titre et le contenu spécifié en paramètres.

 Une image contenant texte

Description générée automatiquement

Les notifications sont définies en XML, Sous le format suivant :

Une image contenant texte

Description générée automatiquement

Il existe bien un module PowerShell prenant en charge l’affichage de notifications.

Cependant, le module doit être installé ce qui prend du temps. Le but principal de ce script, est d’afficher une notification pour avertir l’utilisateur que l’exécution du script peux prendre du temps. Donc si la notification d’avertissement n’arrive pas tout de suite cela n’a pas vraiment d’utilité. C’est pour cette raison que nous n’avons pas utilisé le module, mais recodé en XML.

#### Stop-Program.ps1✔

Ce script permet d’arrêter l’exécution des scripts, tout en stoppant la journalisation des actions et en retournant un message d’erreur spécifié en paramètres.

Une image contenant texte

Description générée automatiquement

#### Test-Paths.ps1 ✔

Ce script teste si plusieurs chemins de fichiers / dossiers donnés en arguments existent. Retourne une liste avec tous les fichiers / dossiers non existants. Cette liste est vide si tout les fichiers / dossiers existent.

Une image contenant texte

Description générée automatiquement

#### Compile-Project.ps1

*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 / version succincte de la documentation

## Sources – Bibliographie

*Liste des livres utilisés (Titre, auteur, date), des sites Internet (URL) consultés, des articles (Revue, date, titre, auteur)… Et de toutes les aides externes (noms)*

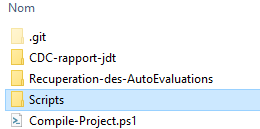
## Journal de travail

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Durée** | **Activité** | **Remarques** |
|  |  |  |  |
|  |  |  |  |

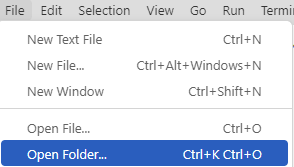
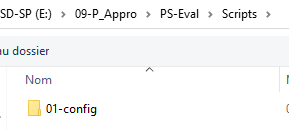
## Manuel de modifications✔

### Modification des scripts✔

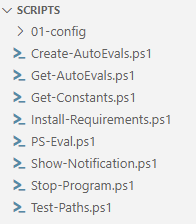
Tous les fichiers de scripts se trouvent dans le dossier ‘’Scripts’’ du GitHub.



Le plus simple est d’ouvrir le dossier depuis Visual Studio Code

 🡪

Vous trouverez sur le côté gauche la liste des fichiers de scripts



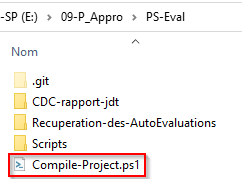
⚠ Pour modifier les scripts il faut suivre quatre points :

* Tous les scripts doivent être dans le dossier ‘’Scripts’’
* Le script ‘’PS-Eval.ps1’’ est le script principal qui lance les autres
* Quand vous appelez une fonction, par exemple ‘’Show-Notification.ps1’’ veillez bien à utiliser cette syntaxe : ‘’.\Show-Notification.ps1’’
* Quand vous utilisez un chemin relatif veillez bien à utiliser cette syntaxe : ‘’./Chemin/etc’’

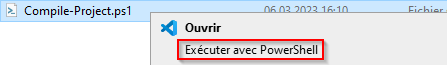
Si un de ces points n’est pas respecté, la compilation des scripts ne fonctionnera pas !

### Compilation des scripts✔

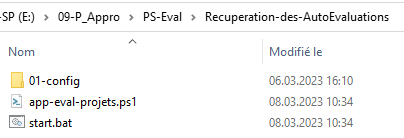
Dans le dossier du GitHub, vous trouverez le fichier ‘’Compile-Project.ps1’’



Faites un clic-droit sur ce fichier, puis ‘’Exécuter avec PowerShell’’



Cela va créer / remplacer le dossier ‘’Recuperation-des-AutoEvaluations’’ et compiler tous les scripts en un seul. Un fichier ‘’start.bat’’ va être crée dans le but d’ignorer la ‘’ExecutionPolicy’’. Le dossier ‘’01-config’’ contenant les fichiers de configurations va être copié.



⚠ Il est nécessaire d’avoir respecté les points mentionnés dans le chapitre ‘’Modifications des scripts’’

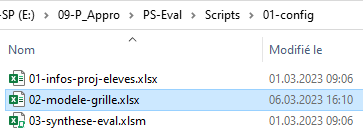
* *Tous les scripts doivent être dans le dossier ‘’Scripts’’*
* *Le script ‘’PS-Eval.ps1’’ est le script principal qui lance les autres*
* *Quand vous appelez une fonction, par exemple ‘’Show-Notification.ps1’’ veillez bien à utiliser cette syntaxe : ‘’.\Show-Notification.ps1’’*
* *Quand vous utilisez un chemin relatif veillez bien à utiliser cette syntaxe : ‘’./Chemin/etc’’*

Si un de ces points n’est pas respecté, le script ne fonctionnera pas !

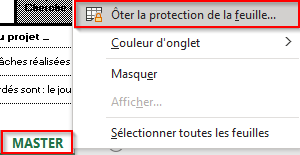
### Modifications du modèle d’auto-évaluations✔

#### Préparation à la modification

Dans le dossier ‘’Scripts/01-config’’, ouvrez le fichier ‘’02-modele-grille.xlsx’’

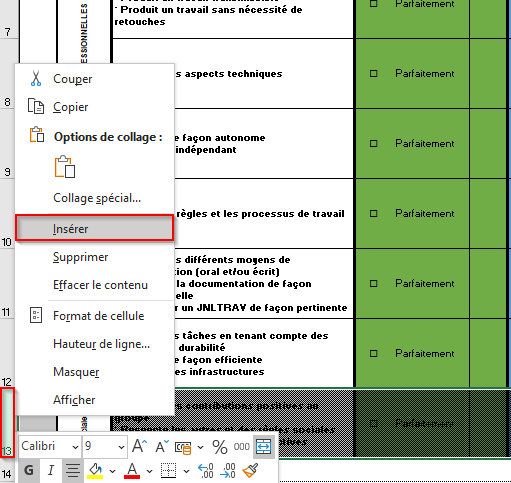


La première étape avant d’effectuer des modifications, est d’ôter la protection de la feuille.



#### Ajout d’un champ de compétence :

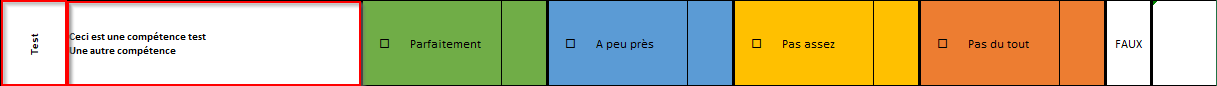
Sélectionnez une ligne des critères d’évaluation, puis faite un clic-droit sur le numéro de ligne. Cliquez ensuite sur ‘’Insérer’’



Sélectionnez les cellules suivantes de la ligne du dessus, puis double-cliquez sur le carré en bas à droite pour étendre les valeurs et formules.



Ajoutez ensuite le titre de la compétence et la description

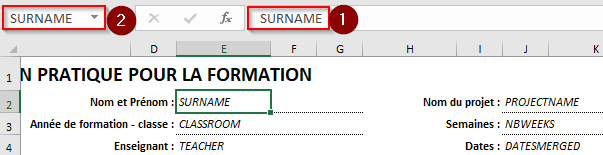


#### Modification des champs de titre :

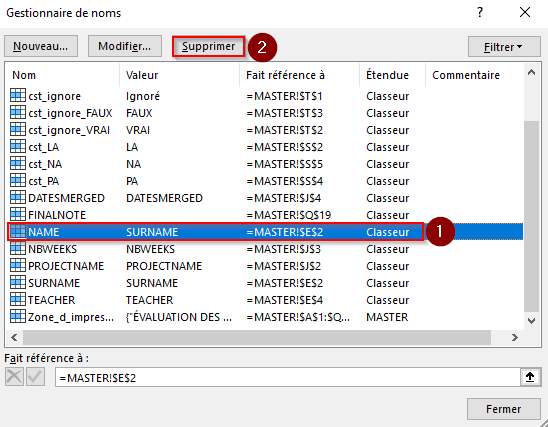
Les champs de titre sont les cellules contenant les variables qui seront modifiés à la génération des auto-évaluations.



Sur les différentes cellules de titre, définissez une valeur dans la cellule (1), puis nommez cette cellule avec la même valeur (2).

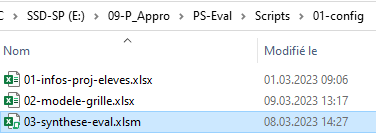


Dans le gestionnaire de noms, supprimez l’ancienne valeur

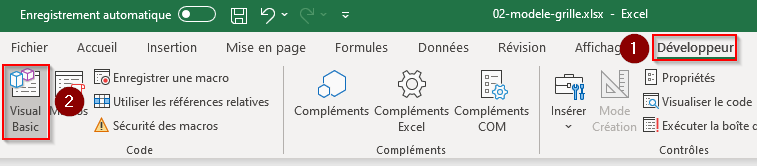


Une fois ces modifications effectuées, enregistrez et fermez le fichier.

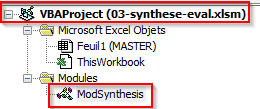
Ouvrez maintenant le fichier ‘’03-synthese-eval.xlsm‘’



Dans l’onglet ‘’Développeur‘’ cliquez sur ‘’Visual Basic‘’

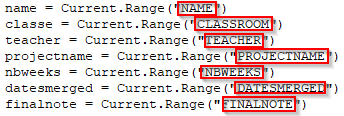


Ouvrez l’onglet ‘’VBAProject‘’ puis le module ‘’ModSynthesis‘’.

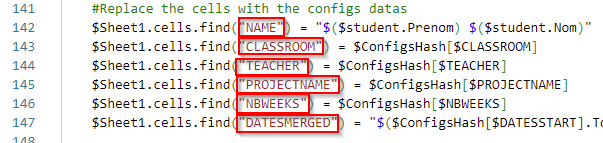


Dans la méthode ‘’Create‘’ changez les valeurs suivantes selon votre modification.

Enregistrez et fermez ensuite le fichier.

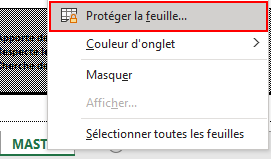


Rendez-vous ensuite dans le fichier ‘’Create-AutoEvals.ps1‘’ du dossier ‘’Scripts‘’

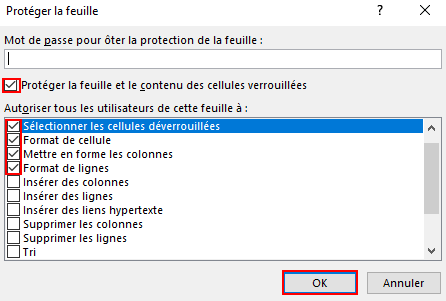


#### Protection de la feuille

Une fois toutes les modifications effectuées n’oubliez pas de reprotéger la feuille Excel pour éviter les modifications non voulues.



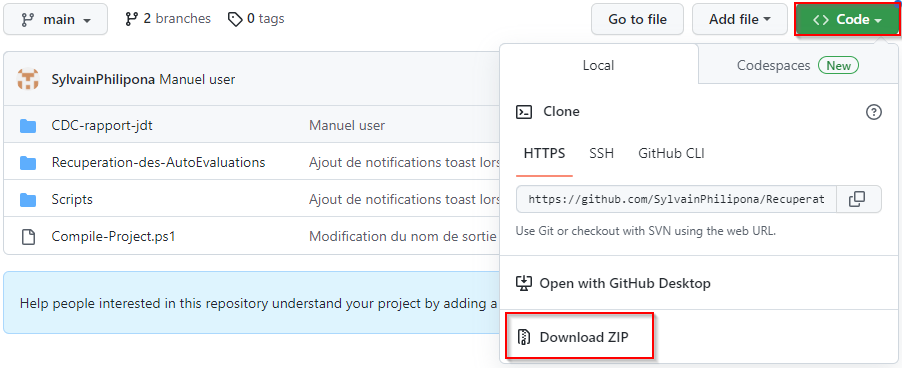
Sélectionnez les checkbox suivantes, puis cliquez sur ‘’OK‘’



## Manuel d'Utilisation ✔

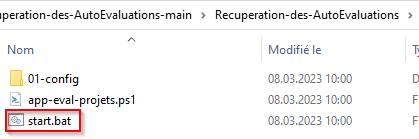
### Téléchargement✔

Pour télécharger le programme vous le trouverez sur [GitHub](https://github.com/SylvainPhilipona/Recuperation-des-AutoEvaluations). Cliquez sur ‘’Code’’ puis ‘’Download ZIP’’

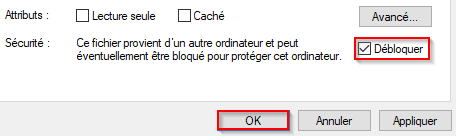


Après avoir télécharger le projet en .zip, veuillez extraire le contenu de l’archive et vous rendre dans le dossier ‘’Recuperation-des-AutoEvaluations’’,

Puis faites clic-droit sur le fichier ‘’start.bat’’ puis ‘’Propriétés’’



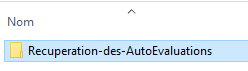
Si vous avez l’option ‘’Débloquer’’ cliquez dessus. Sinon faites uniquement ‘’OK’’



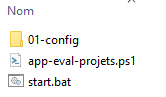
⚠ La première exécution du programme peux être particulièrement lente.

### Configurations✔

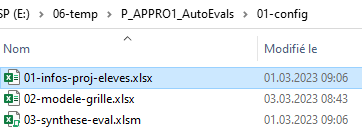
Dans un répertoire, mettez le dossier de scripts que vous avez téléchargé, puis renommez-le en lui attribuant le nom que vous désirez. Par exemple le nom du projet dont vous faites les auto-évaluations

 🡪 

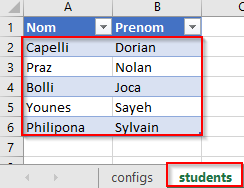
Ensuite entrez dans le dossier, vous trouverez le contenu suivant :



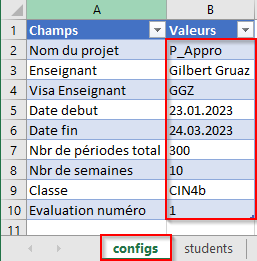
Rendez-vous dans le dossier ‘’01-config’’ puis ouvrez le fichier ‘’01-infos-proj-eleves.xlsx’’



Dans l’onglet Excel ‘’students’’ entrez la liste des élèves dont vous voulez générer les auto-évaluations

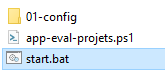


Ensuite dans l’onglet ‘’configs’’ entrez les informations concernant le projet, puis enregistrez le fichier.

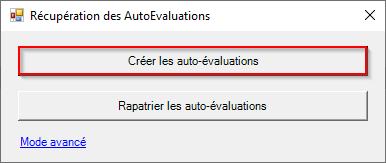


### Création des auto-évaluations✔

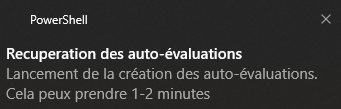
Rendez-vous dans le dossier principal, puis double-cliquez sur le fichier ‘’start.bat’’. Ce fichier démarre le programme.



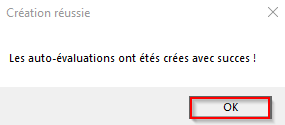
Une fois le programme démarré, cliquez sur ‘’Créer les auto-évaluations’’



Cette Pop-Up va apparaitre pour informer que cela peut prendre un peu de temps selon les performances de votre PC

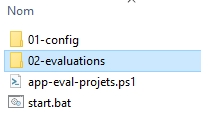
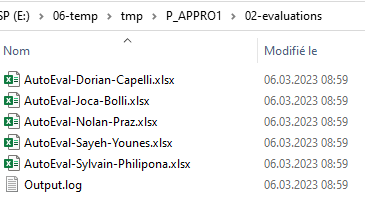


Une fois les auto-évaluations générées, ce message apparaitra. Cliquez sur ‘’OK’’



Un dossier ‘’02-evaluations’’ sera créé contenant toutes les auto-évaluations.

Envoyez les auto-évaluations aux élèves pour qu’ils les remplissent.

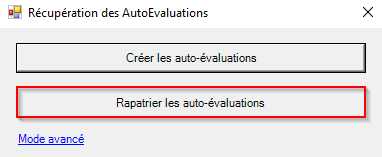
 🡪

### Rapatriement des auto-évaluations✔

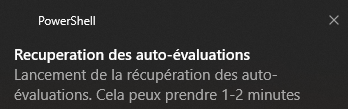
Une fois les auto-évaluations remplies et rendues par les élèves, remettez-les dans le dossier

‘’02-evaluations’’ puis relancez le script avec le fichier ‘’start.bat’’

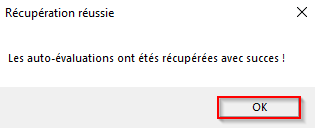
Cette fois cliquez sur ‘’Rapatrier les auto-évaluations’’



Cette Pop-Up va apparaitre pour informer que cela peut prendre un peu de temps selon les performances de votre PC

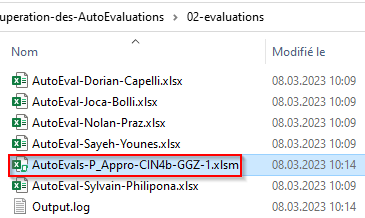


Une fois les auto-évaluations récupérées, ce message apparaitra. Cliquez sur ‘’OK’’



Rendez-vous dans le dossier ‘’02-evaluations’’

Un fichier ‘’.xlsm’’ avec la synthèse des auto-évaluations sera créé. Ouvrez-le.



Une fois ouvert pensez à activer les macros

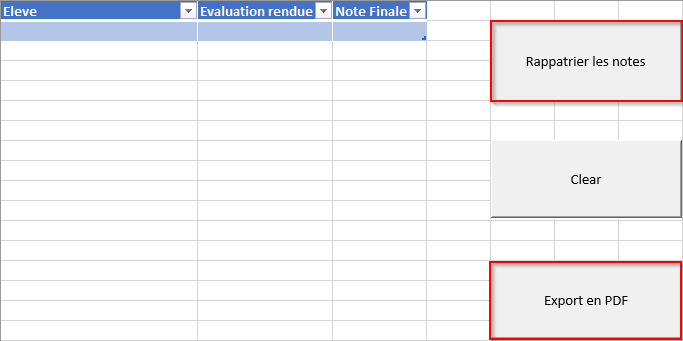


Le fichier de synthèse contient un onglet pour chaque auto-évaluation, ainsi qu’un onglet ‘’MASTER’’

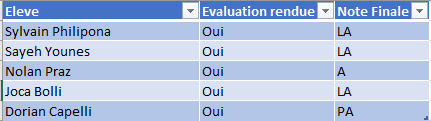
Faites les modifications désirées dans les auto-évaluations des élèves, puis une fois fini rendez-vous dans l’onglet ‘’MASTER’’

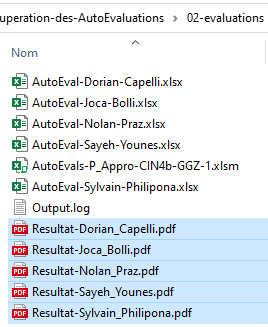


Dans l’onglet ‘’MASTER’’, cliquez sur ‘’Rapatrier les notes’’ et ‘’Export en PDF’’



Les notes seront synthétisées dans le tableau et les auto-évaluations exportées en PDF dans le dossier ‘’02-evaluations’’





## Archives du projet

*Media, … dans une fourre en plastique*