

Récupération des AutoEvaluations

Table des matières

1	Analyse préliminaire	3
1.1	Introduction	3
1.2	Objectifs.....	4
1.3	Planification initiale	4
2	Analyse / Conception.....	5
2.1	Concept	5
2.2	Gestion de projet	6
2.2.1	Méthodologie de travail.....	6
2.2.2	Outil de versioning	7
2.3	Stratégie de test.....	8
2.4	Risques techniques	8
2.5	Planification	8
2.6	Dossier de conception	9
3	Réalisation.....	11
3.1	Dossier de réalisation	11
3.1.1	Environnement.....	11
3.1.2	Version du projet.....	11
3.1.3	Fichiers de configurations.....	12
3.1.4	Fichiers de scripts.....	14
3.1.5	Librairies externes	24
3.2	Description des tests effectués.....	24
3.3	Liste des documents fournis	25
3.4	Glossaire	26
4	Conclusion.....	26
5	Annexes.....	28
5.1	Résumé du rapport du TPI / version succincte de la documentation	28
5.2	Sources – Bibliographie.....	28
5.3	Journal de travail	31
5.4	Planification détaillée	39
5.5	Manuel de modifications	48
5.5.1	Modification des scripts	48
5.5.2	Compilation des scripts.....	49
5.5.3	Modifications du modèle d'auto-évaluations	50
5.6	Manuel d'Utilisation.....	55
5.6.1	Téléchargement.....	55
5.6.2	Configurations.....	56
5.6.3	Création des auto-évaluations	57
5.6.4	Rapatriement des auto-évaluations	58
5.7	Archives du projet.....	61

1 Analyse préliminaire

1.1 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 minimales. 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 paraît 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.

1.2 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.

1.3 Planification initiale

	23.01.2023	30.01.2023	06.02.2023	13.02.2023	20.02.2023	27.02.2023	06.03.2023	13.03.2023	20.03.2023
	Semaine 1	Semaine 2	Semaine 3	Semaine 4	Semaine 5	Semaine 6	Semaine 7	Semaine 8	Semaine 9
INFORMER : Analyse du projet									
PLANIFIER : Découper en tâches et concevoir									
DECIDER : données, interfaces, langage, etc.									
REALISER : Data (Collecte, Modélisation), Programmation									
TESTER : mise en commun des réalisations									
EVALUER : voir cahier des charges									
DOCUMENTER : Documentation									
Recherche d'informations									
Autre									
Imprévu									
Vacances									

Figure 1 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.

2 Analyse / Conception

2.1 Concept

Le concept de ce projet se résume en l'automatisation de la gestion des auto-évaluations. Ces auto-évaluations sont utilisées pour que les élèves se notes eux même selon ce qu'ils ont accompli durant les projets.

Voici à quoi va ressembler le projet.

- La première étape consiste au remplissage des données du projet par l'utilisateur. Par exemple, nom du projet, durée... Ainsi que la liste des élèves.
- Ensuite l'utilisateur lance le script puis choisi l'option de création des auto-évaluations. (1)
- Le script crée ensuite une auto-évaluation par élève.
- L'utilisateur doit ensuite envoyer les auto-évaluations aux élèves.
- Les élèves remplissent les auto-évaluations et l'envoie ensuite à l'enseignant.
- L'utilisateur relance le script puis choisit cette fois l'option de rattachement des auto-évaluations. (2)
- Le script rattaché ensuite toutes les auto-évaluations puis crée une synthèse.
- Pour finir, l'utilisateur corrige les auto-évaluations et les renvoie corrigées aux élèves.

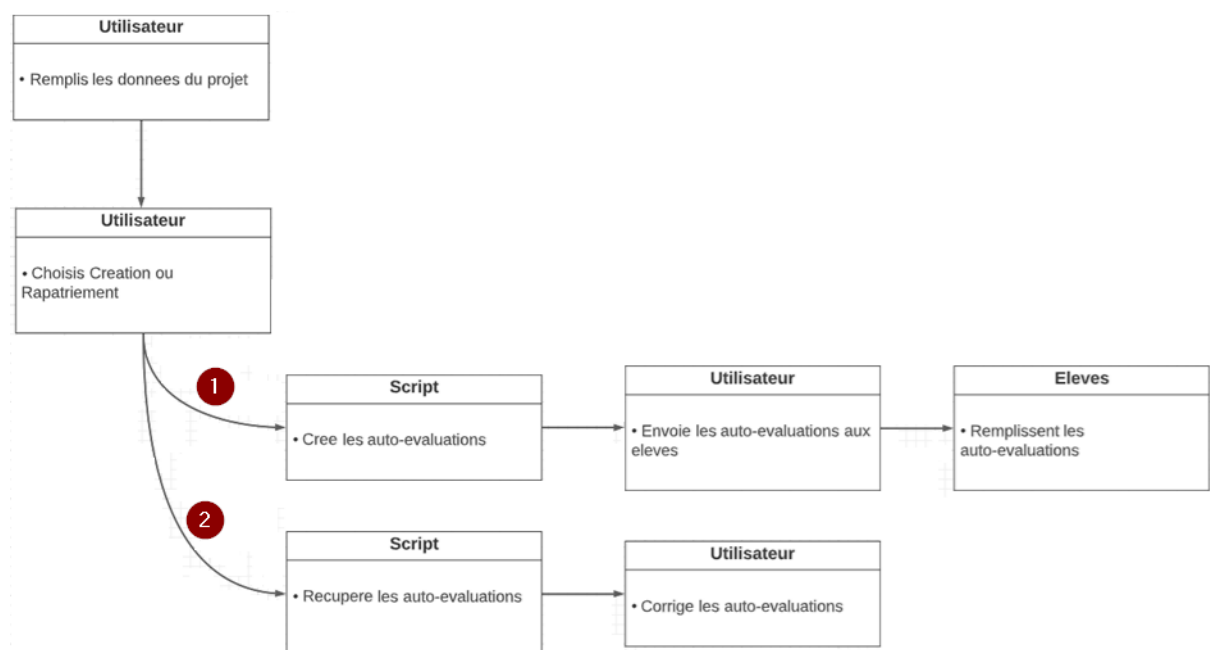


Figure 2 Schéma de fonctionnement

2.2 Gestion de projet

2.2.1 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.



Figure 3 Schéma méthode des 6 pas



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).



4. 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.



6. 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.

2.2.2 Outil de versioning

L'outil de versioning que nous avons utilisé, est Git avec la plateforme GitHub.

Vous trouverez le répertoire avec le lien suivant :

<https://github.com/SylvainPhilipona/Recuperation-des-AutoEvaluations>

Le répertoire à été organisé en deux branches. "Main" et "Dev". Toutes les fonctionnalités sont développées dans la branche "Dev", puis une fois fonctionnelles, la branche est "merged" dans la branche "Main".

2.3 Stratégie de test

Test	Impact en cas d'échec
Le script démarre et affiche l'interface	Impossibilité d'utiliser le projet.
Les auto-évaluations sont créées avec les données spécifiées dans le fichier de configuration	Impossibilité d'utiliser le projet.
Les résultats des auto-évaluations sont correctement récupérés	Synthèse des notes impossible
Les commentaires des élèves sont correctement récupérés	Correction par l'enseignant biaisée
Les auto-évaluations sont correctement exportées en PDF	Envoie des notes corrigées doit être fait à la main
Aucun droit ni configuration nécessitant un spécialiste n'est nécessaire	Utilisation limitée selon les personnes
Fonctionnement sur Windows 11	Le projet va très vite être obsolète

2.4 Risques techniques

Les risques techniques de ce projet, sont surtout liés à la compatibilité de la machine utilisé par l'utilisateur. Le projet ne devant pas nécessiter l'intervention d'un technicien pour installer le projet.

La solution est d'automatiser l'installation des modules lors de la première exécution du programme. De cette manière l'utilisateur n'a rien à faire et le programme installe tout ce dont il a besoin.

2.5 Planification

Pour ce projet, la méthodologie de projet qui sera utilisée est la méthode des 6 pas. Ce projet se déroule en 9 semaines. Du Lundi 23 Janvier 2023 au Vendredi 24 Mars 2023. Cela correspond à 202.5 heures.

Les semaines durent 22.5 heures sont découpées de la manière suivante.

Lundi -> 6h, Mardi -> 0h, Mercredi -> 6h45m, Jeudi -> 3h, Vendredi -> 6h45m

Les semaines ont ensuite été découpées selon les différentes étapes de la méthode des 6 pas. Tout d'abord, la collecte d'informations nécessaires pour le projet dans l'étape "Informer". Ensuite, l'étape "Planifier" qui implique l'organisation de toutes les autres étapes du projet. Puis, l'étape "Décider" pour choisir les outils et les moyens à utiliser pour le projet. Ensuite, vient l'étape "Réaliser" pour la mise en œuvre du projet. L'étape suivante, "Contrôler" ou "Tester", permet de vérifier que les fonctionnalités du projet sont opérationnelles. Enfin, l'étape "Evaluer" sert à vérifier que toutes les attentes du projet ont été correctement satisfaites. La méthode des six

étapes est facile à appliquer et peut être utilisée aussi bien pour des projets professionnels que personnels.

La planification détaillée se trouve au point **5.4**

2.6 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.

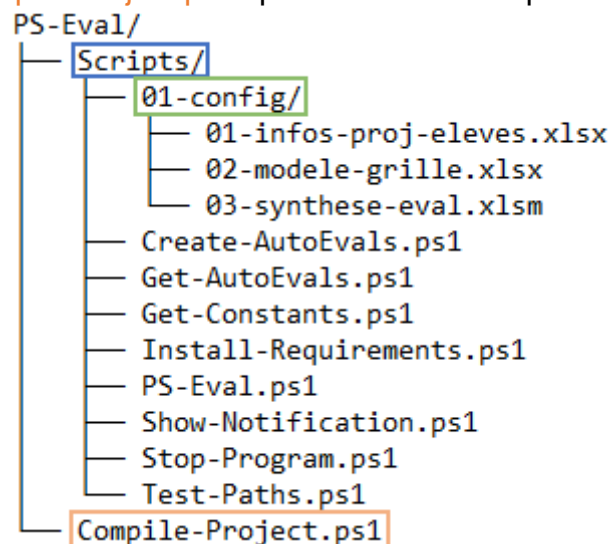


Figure 4 Arborescence du projet

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 facilité.

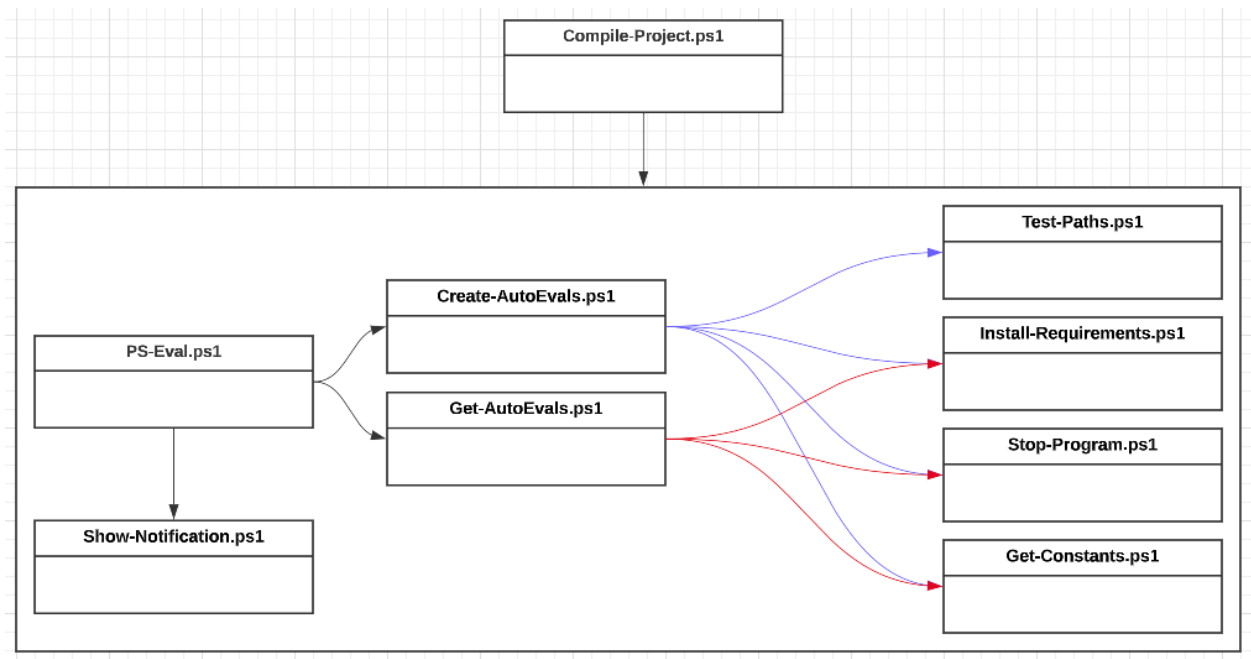


Figure 5 Schéma de fonctionnement du projet

3 Réalisation

3.1 Dossier de réalisation

3.1.1 Environnement

Pour ce projet, nous avons 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

3.1.2 Version du projet

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

3.1.3 Fichiers de configurations

Voici les différents fichiers de configuration que nous allons aborder :

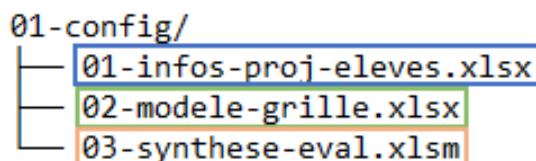


Figure 6 Fichiers de configuration

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.

Champs		Valeurs
Nom du projet		P_Appro
Enseignant		Gilbert Gruaz
Visa Enseignant		GGZ
Nom	Prenom	Date debut
Capelli	Dorian	23.01.2023
Praz	Nolan	Date fin
Bolli	Joca	24.03.2023
Younes	Sayeh	Nbr de périodes total
Philipona	Sylvain	300
		Nbr de semaines
		10
		Classe
		CIN4b
		Evaluation numéro
		1

Figure 7 Configurations des auto-évaluations

Figure 8 Liste des élèves

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**

ÉVALUATION DES COMPÉTENCES EN PRATIQUE POUR LA FORMATION									
INFORMATICIEN-NE		Nom et Prénom : <u>NAME</u>		Nom du projet : <u>PROJECTNAME</u>		Année de formation - classe : <u>CLASSROOM</u>		Semaines : <u>NEWEEKS</u>	
		Enseignant : <u>TEACHER</u>		Dates : <u>DATESMERGED</u>					
		LARGEMENT ACQUIS		ACQUIS		PARTIELLEMENT ACQUIS		NON ACQUIS	
Compétences	PROFESSIONNELLES	Garantit les délais Travaille rapidement	<input type="checkbox"/> Parfaitement	<input type="checkbox"/> A peu près	<input type="checkbox"/> Pas assez	<input type="checkbox"/> Pas du tout			FAUX
		Produit un travail parfaitement utilisable Produit un travail transmissible Produit un travail sans nécessité de retouches	<input type="checkbox"/> Parfaitement	<input type="checkbox"/> A peu près	<input type="checkbox"/> Pas assez	<input type="checkbox"/> Pas du tout			FAUX
		Maîtrise les aspects techniques	<input type="checkbox"/> Parfaitement	<input type="checkbox"/> A peu près	<input type="checkbox"/> Pas assez	<input type="checkbox"/> Pas du tout			FAUX
		Travaille de façon autonome Se montre indépendant	<input type="checkbox"/> Parfaitement	<input type="checkbox"/> A peu près	<input type="checkbox"/> Pas assez	<input type="checkbox"/> Pas du tout			FAUX
	Méthodologie	Intègre les règles et les processus de travail	<input type="checkbox"/> Parfaitement	<input type="checkbox"/> A peu près	<input type="checkbox"/> Pas assez	<input type="checkbox"/> Pas du tout			FAUX
		Maîtrise les différents moyens de communication (oral et/ou écrit) Produit de la documentation de façon professionnelle Tient à jour un JNL TRAV de façon pertinente	<input type="checkbox"/> Parfaitement	<input type="checkbox"/> A peu près	<input type="checkbox"/> Pas assez	<input type="checkbox"/> Pas du tout			FAUX
		Réalise ses tâches en tenant compte des principes de durabilité Travaille de façon efficiente Respecte les infrastructures	<input type="checkbox"/> Parfaitement	<input type="checkbox"/> A peu près	<input type="checkbox"/> Pas assez	<input type="checkbox"/> Pas du tout			FAUX
	Sociales	Apporte des contributions positives au groupe Respecte les autres et des règles sociales Cherche des solutions constructives	<input type="checkbox"/> Parfaitement	<input type="checkbox"/> A peu près	<input type="checkbox"/> Pas assez	<input type="checkbox"/> Pas du tout			FAUX Ignoré

Figure 9 Modèle d'auto-évaluation

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.

[illegible]

Figure 10 Modèle de synthèse

3.1.4 Fichiers de scripts

Voici les différents fichiers de script que nous allons aborder :

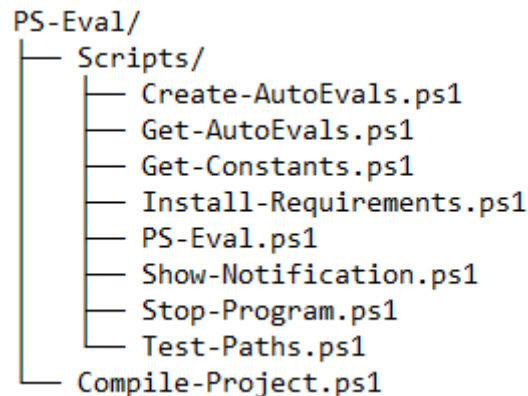


Figure 11 Arborescence du projet

3.1.4.1 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.

```

param (
    [string]$ConfigsPath,
    [string]$ModelPath,
    [string]$OutputPath
)
  
```

Figure 12 Paramètres pour la création des auto-évaluations

Toutes les constantes nécessaires au script sont récupérées depuis le fichier "Get-Constants.ps1".

```

$constants = .\Get-Constants.ps1
$ConfigSheet = $constants.ConfigFile.ConfigSheet
$StudentsSheet = $constants.ConfigFile.StudentsSheet
$CLASSROOM = $constants.RequiredInputs.CLASSROOM
$TEACHER = $constants.RequiredInputs.TEACHER
$PROJECTNAME = $constants.RequiredInputs.PROJECTNAME
$NBWEEKS = $constants.RequiredInputs.NBWEEKS
$DATESSTART = $constants.RequiredInputs.DATESSTART
$DATEEND = $constants.RequiredInputs.DATEEND

ConfigFile = @{
    ConfigSheet = 'configs'
    StudentsSheet = 'students'
}

RequiredInputs = @{
    CLASSROOM = 'Classe'
    TEACHER = 'Enseignant'
    PROJECTNAME = 'Nom du projet'
    NBWEEKS = 'Nbr de semaines'
    DATESSTART = 'Date debut'
    DATEEND = 'Date fin'
    VISA = 'Visa Enseignant'
}
  
```

Figure 13 Obtention des constantes

Figure 14 Déclaration des constante

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.

```
#Import the configs and students inputs
$Configs = (Import-Excel -Path $ConfigsPath -WorksheetName $ConfigSheet)
$Students = (Import-Excel -Path $ConfigsPath -WorksheetName $StudentsSheet)
```

Figure 15 Import du fichier de configurations

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.

```
try{
    $excel = New-Object -ComObject excel.application
}
catch [System.Runtime.InteropServices.COMException] {
    .\Stop-Program.ps1 -errorMessage "Excel n'est pas installé. Veuillez l'installer et recommencer !"
}
catch{
    .\Stop-Program.ps1 -errorMessage "Une erreur est survenue. Verifiez que Excel est bien installé et configuré !"
}
```

Figure 16 Création de l'Object Excel

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

```
#Import the model file
$excel.visible = $false
$workbook = $excel.Workbooks.Open($ModelPath)
$Sheet1 = $workbook.worksheets.item(1)

#Unprotect the sheet
$Sheet1.Unprotect()

#Replace the cells with the configs datas
$Sheet1.cells.find("NAME") = "$($student.Prenom) $($student.Nom)"
$Sheet1.cells.find("CLASSROOM") = $ConfigsHash[CLASSROOM]
$Sheet1.cells.find("TEACHER") = $ConfigsHash[TEACHER]
$Sheet1.cells.find("PROJECTNAME") = $ConfigsHash[PROJECTNAME]
$Sheet1.cells.find("NBWEEKS") = $ConfigsHash[NBWEEKS]
$Sheet1.cells.find("DATESMERGED") = "$($ConfigsHash[DATESTART].ToString("yyyy/MM/dd"))-$($ConfigsHash[DATEEND].ToString("yyyy/MM/dd"))"
```

Figure 17 Génération des auto-évaluations

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

```
#Save the new file as the student name (Overwrite if the file exists)
$filename = "$OutputPath\AutoEval-$( $student.Prenom + "-" + $student.Nom).xlsx"
Remove-Item -Path $filename -Force -Confirm:$false -ErrorAction SilentlyContinue
$workbook.Saveas($filename)
```

Figure 18 Enregistrement des auto-évaluations

3.1.4.2 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

```
param (
    [string]$ConfigsPath,
    [string]$SynthesisModelPath,
    [string]$FilesPath
)
```

Figure 19 Paramètres pour le rapatriement des auto-évaluations

Toutes les constantes nécessaires au script sont récupérées depuis le fichier "Get-Constants.ps1".

```
$constants = .\Get-Constants.ps1
$ConfigSheet = $constants.ConfigFile.ConfigSheet
$CLASSROOM = $constants.RequiredInputs.CLASSROOM
$PROJECTNAME = $constants.RequiredInputs.PROJECTNAME
$VISA = $constants.RequiredInputs.VISA
```

Figure 20 Obtention des constantes

```
ConfigFile = @{
    ConfigSheet = 'configs'
    StudentsSheet = 'students'
}

RequiredInputs = @{
    CLASSROOM = 'Classe'
    TEACHER = 'Enseignant'
    PROJECTNAME = 'Nom du projet'
    NBWEEKS = 'Nbr de semaines'
    DATESSTART = 'Date debut'
    DATEEND = 'Date fin'
    VISA = 'Visa Enseignant'
}
```

Figure 21 Déclaration des constantes

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.

```
#Import the configs and students inputs
$Configs = (Import-Excel -Path $ConfigsPath -WorksheetName $ConfigSheet)
```

Figure 22 Importation du fichier de configurations

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.

```
#Create the COM object
try{
    $excel = New-Object -ComObject excel.application
    $excel.visible = $false
}
catch [System.Runtime.InteropServices.COMException] {
    .\Stop-Program.ps1 -errorMessage "Excel n'est pas installé. Veuillez l'installer et recommencer !"
}
catch{
    .\Stop-Program.ps1 -errorMessage "Une erreur est survenue. Verifiez que Excel est bien installé et configuré !"
}
```

Figure 23 Création de l'Object Excel

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

```
#Open the auto eval
$WorkbookEval = $excel.Workbooks.Open($eval.FullName)
$SheetEval = $WorkbookEval.worksheets.item(1)

#Copy the auto eval in the synthesis file
$SheetEval.copy($WorkbooxSynthesis.sheets.item(1))
$WorkbookEval.Close()
```

Figure 24 Création de la synthèse

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"

```
#Save and close the object
# AutoEvals-ProjectName-Classe-Prof-01.xlsm
$ExcelFixedFormat = [Microsoft.Office.Interop.Excel.XlFileFormat]::xlOpenXMLWorkbookMacroEnabled
$FileName = "$FilePath\AutoEvals-$(($ConfigsHash[$PROJECTNAME])-$($ConfigsHash[$CLASSROOM])-$($ConfigsHash[$SVISA])"-1.xlsm"
$WorkbooxSynthesis.SaveAs($FileName,$ExcelFixedFormat)
```

Figure 25 Enregistrement de la synthèse en xlsm

3.1.4.3 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**.

```
return [PSCustomObject]@{
    form = @{
        FormWidth = 400
        FormHeight = 170
        FormHeightAdvanced = 380
        InputsWidth = 350
        InputsHeight = 30
        LabelsHeight = 15
        SpaceBetweenInputs = 15
    }

    ConfigFile = @{
        ConfigSheet = 'configs'
        StudentsSheet = 'students'
    }

    RequiredInputs = @{
        CLASSROOM = 'Classe'
        TEACHER = 'Enseignant'
        PROJECTNAME = 'Nom du projet'
        NBWEEKS = 'Nbr de semaines'
        DATESSTART = 'Date debut'
        DATEEND = 'Date fin'
        VISA = 'Visa Enseignant'
    }
}

$constants = .\Get-Constants.ps1
$ConfigSheet = $constants.ConfigFile.ConfigSheet
$StudentsSheet = $constants.ConfigFile.StudentsSheet
$CLASSROOM = $constants.RequiredInputs.CLASSROOM
$TEACHER = $constants.RequiredInputs.TEACHER
$PROJECTNAME = $constants.RequiredInputs.PROJECTNAME
$NBWEEKS = $constants.RequiredInputs.NBWEEKS
$DATESSTART = $constants.RequiredInputs.DATESSTART
$DATEEND = $constants.RequiredInputs.DATEEND
```

Figure 26 Déclaration des constantes

Figure 27 Obtention des constantes

3.1.4.4 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".

```
# Install the NuGet package
Write-Host "Installation de NuGet" -ForegroundColor Green
Install-PackageProvider -Name NuGet -MinimumVersion 2.8.5.208 -Scope CurrentUser -Force -Confirm:$false | Out-Null

# Set PSGallery repo to trusted -> For the ImportExcel installation
if((Get-PSRepository -Name "PSGallery").InstallationPolicy -ne "Trusted"){
    Write-Host "Ajout du répertoire PSGallery en répertoire de confiance" -ForegroundColor Green
    Set-PSRepository -name "PSGallery" -InstallationPolicy Trusted
}

# Install the module ImportExcel
if(!(Get-Module -ListAvailable -name ImportExcel)){
    Write-Host "Installation de ImportExcel" -ForegroundColor Green
    Install-Module ImportExcel -Scope CurrentUser -Confirm:$false #https://github.com/dfinke/ImportExcel
}
```

Figure 28 Installation des modules

3.1.4.5 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".

```
$constants = .\Get-Constants.ps1
$FormWidth = $constants.form.FormWidth
$FormHeight = $constants.form.FormHeight
$FormHeightAdvanced = $constants.form.FormHeightAdvanced
$InputsWidth = $constants.form.InputsWidth
$InputsHeight = $constants.form.InputsHeight
$LabelsHeight = $constants.form.LabelsHeight
$SpaceBetweenInputs = $constants.form.SpaceBetweenInputs

form = @{
    FormWidth = 400
    FormHeight = 170
    FormHeightAdvanced = 380
    InputsWidth = 350
    InputsHeight = 30
    LabelsHeight = 15
    SpaceBetweenInputs = 15
}
```

Figure 29 Obtention des constantes

Figure 30 Déclaration des constantes

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

```
# .Net methods for hiding/showing the console in the background
Add-Type -Name Window -Namespace Console -MemberDefinition '
[DllImport("Kernel32.dll")]
public static extern IntPtr GetConsoleWindow();

[DllImport("user32.dll")]
public static extern bool ShowWindow(IntPtr hWnd, Int32 nCmdShow);
'

1 reference
function Hide-Console
{
    $consolePtr = [Console.Window]::GetConsoleWindow()
    #0 = hide
    [Console.Window]::ShowWindow($consolePtr, 0) | Out-Null
}
Hide-Console
```

Figure 31 Masquer la console PowerShell

Le formulaire est créé en Windows Forms

```
Add-Type -AssemblyName System.Windows.Forms
Add-Type -AssemblyName System.Drawing

# Global variable because its value is changed in an event
$global:advancedMode = $false

# Main form
$form = New-Object System.Windows.Forms.Form
$form.Text = 'Récupération des AutoEvaluations'
$form.Size = New-Object System.Drawing.Size($FormWidth,$FormHeight)
$form.StartPosition = 'CenterScreen'
$form.FormBorderStyle = 'FixedSingle'
$form.MaximizeBox = $false
$form.MinimizeBox = $false
$form.Topmost = $false
```

Figure 32 Création du formulaire

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

```
# Create auto-evals Button event
$createEvalsButton.Add_Click(
{
    # Lock the form and buttons
    $form.Enabled = $false

    # Trim the Output Path
    $outputPathInput.Text = $outputPathInput.Text.TrimEnd(' ')
    $outputPathInput.Text = $outputPathInput.Text.TrimEnd('\')

    try{
        # Display the toast notif
        .\Show-Notification.ps1 -ToastTitle "Recuperation des auto-évaluations" -ToastText "Lancement de la création des auto-évaluations. Cela peut prendre 1-2 minutes"

        # Start the creation
        .\Create-AutoEvals.ps1 -ConfigsPath $configPathInput.Text -ModelPath $modelPathInput.Text -OutputPath $outputPathInput.Text

        [System.Windows.Forms.MessageBox]::Show("Les auto-évaluations ont été créés avec succès !", "Création réussie")
    }
    catch{
        #Display the error message
        try{Stop-Transcript}catch{}
        [System.Windows.Forms.MessageBox]::Show($_, "Erreur d'exécution")
    }

    # Unlock the form and buttons
    $form.Enabled = $true
}
);
```

Figure 33 Démarrage de la création des auto-évaluations

3.1.4.6 Show-Notification.ps1

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

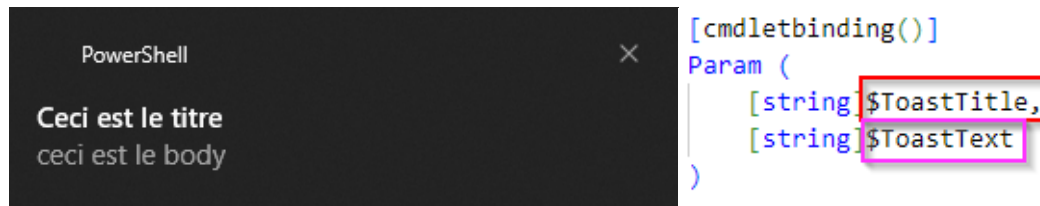


Figure 34 Notification

Figure 35 Titre et contenu de la notification

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

```
<toast>
  <visual>
    <binding template="ToastText02">
      <text id="1">Ceci est le titre</text>
      <text id="2">Ceci est le contenu</text>
    </binding>
  </visual>
</toast>
```

Figure 36 Code XML de la notification

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 peut 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.

3.1.4.7 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**.

```
param (
    [string]$errorMessage
)

try{
    # Stop the transcription
    Stop-Transcript | out-null
}
catch{}

# Throw the custom error message
throw $errorMessage
```

Figure 37 Fonction pour arrêter l'exécution

3.1.4.8 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.

```
param (
    [String[]]$paths
)

$notExistingPaths = @()

# Test all paths
foreach($path in $paths){
    if(!(Test-path $path)){
        $notExistingPaths += $path
    }
}

return $notExistingPaths
```

Figure 38 Fonction pour tester l'existence de chemins

3.1.4.9 Compile-Project.ps1

Ce script 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 facilité.

Les paramètres sont définis comme ci-dessous. Ce sont les valeurs proposées au prochain développeur.

```
param (
    [string]$compiled = "app-eval-projets.ps1",
    [string]$mainScript = "PS-Eval.ps1",
    [string]$constantsScript = "Get-Constants.ps1",
    [string]$configsPath = "01-config",
    [string]$scriptsPath = "./Scripts",
    [string]$outputPath = "./Recuperation-des-AutoEvaluations"
)
```

Figure 39 Paramètres pour la compilation

Tous les **scripts sont récupérés**, puis un **dossier de sortie est créé et est défini comme répertoire courant**. Les scripts "PS-Eval.ps1" et "Get-Constants.ps1" sont exclus. Vous verrez la raison à l'étape suivante.

```
# Get all scripts from the path
$scripts = Get-ChildItem -Path $scriptsPath -Filter *.ps1 -Exclude @($mainScript, $constantsScript) -Recurse

# Create the Output folder and move to it
New-Item $outputPath -ItemType Directory -Force | Out-Null
Set-Location $outputPath
```

Figure 40 Récupération des scripts

Tout le contenu des scripts est récupéré, puis les ".ps1" et "." sont retirés. Car la syntaxe pour appeler les scripts est la suivante : ".\Script-Exemple.ps1" et quand les fonctions sont définies dans le même script la syntaxe est la suivante : "Script-Exemple". Le fichier "Get-Constants.ps1" doit se retrouver en haut du script, car les autres scripts l'appellent tous. Le script "PS-Eval.ps1" doit lui être en dernier car c'est lui qui lance les autres scripts.

```
# Get all files content and wrap it in a 'Function'
# Remove all "." and ".ps1". Because in the files the scripts are called like this ".\My-Function.ps1" and when wrapped like this "My-Function"
# The result will be output in a file
$wrapContent = ""
$wrapContent += "Function $($constantsScript.Replace('.ps1', '')) {"
$wrapContent += [IO.File]::ReadAllText("$scriptsPath\$constantsScript").Replace(".", "").Replace(".ps1", "")
$wrapContent += "}"
foreach($script in $scripts){
    $wrapContent += "Function $($script.Name.Replace('.ps1', '')) {"
    $wrapContent += [IO.File]::ReadAllText($script.FullName).Replace(".", "").Replace(".ps1", "")
    $wrapContent += "}"
}
$wrapContent += [IO.File]::ReadAllText("$scriptsPath\$mainScript").Replace(".", "").Replace(".ps1", "")
$wrapContent >> $compiled
```

Figure 41 Compilation

Le dossier "01-config" est copié dans le dossier de sortie.

```
Copy-Item "..\$scriptsPath\$configsPath" -force -Recurse
```

Figure 42 Copie du dossier de configurations

Un fichier .bat est créé pour permettre à l'utilisateur de lancer facilement le projet et pour contourner la police d'exécution "Restricted".

```
# Create the launch file
```

```
New-Item -Path . -Name "start.bat" -Force | Out-Null
```

```
Set-Content "start.bat" "powershell -executionPolicy bypass -file $compiled"
```

Figure 43 Création du fichier de lancement

3.1.5 Bibliothèques externes

La seule bibliothèque externe utilisée dans ce projet est ImportExcel.

C'est une librairie qui permet d'effectuer des opérations Excel basiques très simplement. Elle a été utilisée dans les scripts "Create-AutoEvals.ps1" et "Get-AutoEvals.ps1"

3.2 Description des tests effectués

Test	Impact en cas d'échec	Résultat attendu	Résultat obtenu	Conclusion	Temps requis pour corriger et / ou finaliser
Le script démarre et affiche l'interface	Impossibilité d'utiliser le projet.	L'interface graphique est affichée	OK 100%	Rien à signaler	
Les auto-évaluations sont créées avec les données spécifiées dans le fichier de configuration	Impossibilité d'utiliser le projet.	Les auto-évaluations sont correctement créées	OK 100%	Rien à signaler	
Les résultats des auto-évaluations sont correctement récupérés	Synthèse des notes impossible	Les résultats sont rapatriés	OK 100%	Rien à signaler	

Les commentaires des élèves sont correctement récupérés	Correction par l'enseignant biaisée	Les commentaires sont rapatriés	OK 100%	Rien à signaler	
Les auto-évaluations sont correctement exportées en PDF	Envoie des notes corrigées doit être fait à la main	Les auto-évaluations sont exportées en PDF	OK 100%	Rien à signaler	
Aucun droit ni configuration nécessitant un spécialiste n'est nécessaire	Utilisation limitée selon les personnes	Tout utilisateur peut utiliser le projet sans erreurs	OK 100%	Rien à signaler	
Fonctionnement sur Windows 11	Le projet va très vite être obsolète		OK 100%	Le script sera utilisable lors de la migration du parc vers Windows 11	

3.3 Liste des documents fournis

- Le rapport de projet
- Le code source du projet
- Manuel d'utilisation
- Manuel de modifications
- Journal de travail
- Planification détaillée

Les documents cités ci-dessus sont disponible dans la partie annexe de ce document.

3.4 Glossaire

1. Foreach : Boucle permettant notamment de parcourir un tableau
2. ComObject : Objet PowerShell permettant de manipuler des composants Microsoft
3. PSGallery : Référentiel de modules PowerShell
4. MessageBox : Boite affichant un message à l'utilisateur
5. ExecutionPolicy : Police d'exécution de scripts PowerShell
6. Notification Toast : Notification Windows

4 Conclusion

Pour conclure ce projet, je vais procéder en plusieurs étapes. Les étapes sont les objectifs, les difficultés particulières, la suite possible pour le projet, un bilan personnel et pour finir un bilan de planification.

Les objectifs :

Nous allons aborder les points du cahier des charges et voir s'ils ont été atteints.

- **Les valeurs proposées par les élèves sont récupérées correctement :**
Ce point est à 100% fonctionnel
- **Les commentaires des élèves sont récupérés correctement :**
Ce point est à 100% fonctionnel
- **Toutes les valeurs des indicateurs sont récupérées correctement :**
Ce point est à 100% fonctionnel
- **Le dispositif est pertinent et simple à utiliser :**
Ce point est compliqué à valider car tout le monde est différent et les gens ne trouvent pas tous les mêmes choses faciles. Cependant en vue des tests les retours concernant ce point étaient très positifs.
- **Le dispositif est fiable. En cas d'incidents, les messages d'erreurs sont pertinents :**
Il est très difficile de dire à 100% que le dispositif est fiable car il n'a pas encore été testé à grande échelle. Cependant avec les différents tests effectués, ce point est considéré comme réalisé.
- **Une modification de la structure de la grille est possible :**
Ce point est à 100% fonctionnel
- **Fonctionne sans autorisation spéciale, ni configuration exigeant l'intervention d'un spécialiste système :**
Ce point est à 100% fonctionnel

Les difficultés particulières :

Une des grosses difficultés auxquelles nous nous sommes confrontés, à été la création des auto-évaluations. Car avec PowerShell ce n'est pas très simple d'interagir avec les fichiers Excel. De plus il y'a une quantité limitée de documentation à ce sujet. Cela nous à donc bien challengé lors de l'apprentissage de la manière de faire.

Les suites possibles du projet :

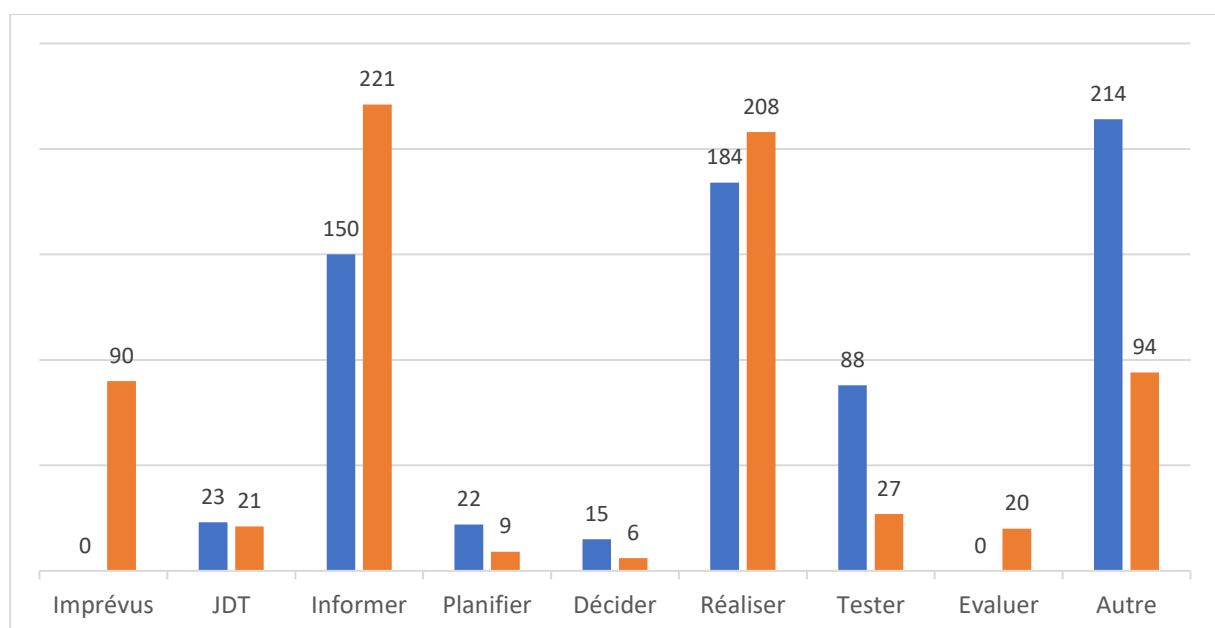
Le projet va être mis en production auprès des enseignants volontaires. Ils l'utiliseront à leurs guises et pourront le modifier / tenir à jours comme ils le désirent. Le code source étant accessible sur GitHub ils pourront sans soucis effectuer les modifications souhaitées.

Bilan personnel :

Nous avons particulièrement apprécié ce projet dans l'ensemble. Aimant beaucoup automatiser les actions, ce projet était très agréable. De plus nous avons pu consolider nos connaissances dans ce domaine. Nous avons également aimé apprendre à gérer des fichiers Excel avec PowerShell, ce qui peut être très utile dans le monde du travail. Ce sont donc des connaissances qui nous seront utile tout au long de notre carrière.

Bilan de planification :

Concernant notre planification, vous pouvez voir sur le graphique ci-dessous les principales différences entre la **planification** et la **réalisation**. Premièrement, nous pouvons constater 90 quarts d'heures en imprévus. Ce qui correspond à une semaine planifiée que nous avons finalement eue en moins pour la résiliation de notre projet. Nous pouvons voir que dans le point "Informer" qui correspond notamment au rapport, nous à pris plus de temps que prévu. Les tests ont cependant été très performants et concluant. En revanche le point "Autre" qui correspondait surtout à la préparation de la présentation à été beaucoup moins réalisé, à cause de la semaine en moins.



5 Annexes

5.1 Résumé du rapport du TPI / version succincte de la documentation

Situation de départ :

Le but de ce projet était de proposer un système d'automatisation pour la gestion des auto-évaluations. Il existait à la base un fichier modèle, et les auto-évaluations étaient créés à partir de ce fichier. Tout était fait à la main par les enseignants, ce qui était très couteux en temps.

Mise en œuvre :

Le programme devait permettre d'automatiser la création des auto-évaluations selon des données spécifiées (Nom du projet, durée, liste des élèves...). De plus il était aussi demandé de pouvoir créer une synthèse avec toutes les auto-évaluations une fois remplies par les élèves. Cette synthèse permet de corriger les auto-évaluations, puis une fonctionnalité devait permettre de les exporter en PDF.

Résultats :

Un script permettant la création et le rapatriement des auto-évaluations. Un fichier pour les configurations (Nom du projet, durée, liste des élèves...), un fichier modèle d'auto-évaluation et un fichier modèle de synthèse.

5.2 Sources – Bibliographie

1 : GitHub du module PowerShell ImportExcel

<https://github.com/dfinke/ImportExcel>

2 : Utilisation de Excel avec PowerShell

<https://techexpert.tips/powershell/powershell-creating-excel-file/>

3 : Ouverture d'un fichier Excel à l'aide de PowerShell

<https://stackoverflow.com/questions/37665118/how-to-open-excel-workbook-from-powershell-for-automation>

4 : Formatage de date en PowerShell

<https://stackoverflow.com/questions/2249619/how-to-format-a-datetime-in-powershell>

5 : Utilisation de JSON avec PowerShell

<http://ramblingcookiemonster.github.io/PowerShell-Configuration-Data/>

6 : Création d'un module manifeste

<https://learn.microsoft.com/en-us/powershell/scripting/developer/module/how-to-write-a-powershell-module-manifest?view=powershell-7.3>

7 : Vider un COM Object en PowerShell

<https://social.technet.microsoft.com/Forums/office/en-US/e5d8594b-b14b-4a54-913c-61089b5d9ab4/release-or-delete-a-com-object-from-powershell?forum=winserverpowershell>

8 : Transcription des logs

<https://learn.microsoft.com/en-us/powershell/module/microsoft.powershell.host/start-transcript?view=powershell-7.3>

9 : Gérer les exceptions en PowerShell

<https://learn-powershell.net/2015/04/09/quick-hits-finding-exception-types-with-powershell/>

10 : Problèmes avec un "Get-ChildItems"

<https://stackoverflow.com/questions/41700636/get-childitem-doesnt-work-with-include>

11 : Copier une feuille Excel dans une autre en PowerShell

<https://stackoverflow.com/questions/3226096/copy-excel-worksheet-from-one-workbook-to-another-with-powershell>

12 : Macro VBA pour modifier un tableau Excel

<https://learn.microsoft.com/en-us/answers/questions/864246/powershell-vba-and-excel>

13 : Parcourir toutes les feuilles Excel d'un fichier en VBA

<https://support.microsoft.com/en-us/topic/macro-to-loop-through-all-worksheets-in-a-workbook-feef14e3-97cf-00e2-538b-5da40186e2b0>

14 : Vider un tableau Excel en VBA

<https://vba1.com/table/clear-table-content/>

15 : Déclaration de variable en VBA

<https://www.automateexcel.com/vba/string-data-type/>

16 : Ajouter des données à un tableau Excel en VBA

<https://officetuts.net/excel/vba/add-data-to-table-using-vba/>

17 : Méthode de remplacement de caractère dans une string en VBA

<https://wellsr.com/vba/2016/excel/vba-replace-function-to-replace-characters-in-string/>

18 : Création de formulaire Windows Forms en PowerShell

<https://learn.microsoft.com/en-us/powershell/scripting/samples/creating-a-custom-input-box?view=powershell-7.3>

19 : Argument avec multiples valeurs en PowerShell

<https://stackoverflow.com/questions/15120597/passing-multiple-values-to-a-single-powershell-script-parameter>

20 : Evènement sur click de bouton en PowerShell

<https://social.technet.microsoft.com/wiki/contents/articles/25911.how-to-add-a-powershell-gui-event-handler-part-1.aspx>

21 : Création d'un label cliquable en PowerShell

<https://social.technet.microsoft.com/wiki/contents/articles/28746.powershell-gui-how-to-create-a-link-label-using-winforms.aspx>

22 : Stocker le contenu d'un fichier dans une variable

<https://stackoverflow.com/questions/7976646/powershell-store-entire-text-file-contents-in-variable>

23 : Création d'une variable globale en PowerShell

<https://stackoverflow.com/questions/68084565/powershell-wpf-boolean-resets-its-value-after-function>

24 : Création d'objets customisés en PowerShell

<https://powershellbyexample.dev/post/custom-object/>

25 : Affichage de notifications toast en PowerShell

<https://www.starwindsoftware.com/blog/how-to-display-notifications-on-windows-10-using-powershell>

26 : Affichage de notifications toast en PowerShell

<https://den.dev/blog/powershell-windows-notification/>

5.3 Journal de travail

Semaine 1		Début: lundi, 23 janvier 2023	
Tâche	Durée [1/4 h.]	Explications: qu'est-ce qui se fait et comment ?	Liens, références, ...
Lundi 24p			
Autre	18	• Présentations des stages des élèves, attribution des projets	
REALISER : Data (Collecte, Modélisation), Programmation	5	• Création du git • Création de l'arborescence initiale • Récupération des données à exporter et création d'une structure pour input les données des élèves dans les scripts	
Rédaction JNLTRAV	1		
Mercredi 27			
INFORMER : Analyse du projet	6	• Analyse du cahier des charges avec M.Gruaz	
PLANIFIER : Découper en tâches et concevoir	9	• Planification initiale, création des user stories sur git	https://github.com/users/SylvainPhilipona/projects/1/views/1
DECIDER : données, interfaces, langage, etc.	6	• Recherche d'informations concernant les technologies PowerShell pour réaliser ce projet	https://github.com/dfinke/ImportExcel
Rédaction JNLTRAV	1		
REALISER : Data (Collecte, Modélisation), Programmation	5	• Récupération de données de cellules excel via PowerShell	https://techexpert.tips/powershell/powershell-creating-excel-file/ https://stackoverflow.com/questions/37665118/how-to-open-excel-workbook-from-powershell-for-automation
Jeudi 12			
Imprévu	6	• Présentation de M.Sahli sur les TPI • Demande à M.Ferrari et M.Sahli pour mon chef de TPI	
INFORMER : Analyse du projet	4	• Analyse des critères d'évaluation du TPI	
Rédaction JNLTRAV	2	• Remplissage du JDT • Mise en forme du fichier	
Vendredi 27			
Imprévu	6	• Mise à jour de mon poste • Appel avec mon maître de stage concernant un script	
REALISER : Data (Collecte, Modélisation), Programmation	16	• Récupération des données d'inputs pour les auto evals (nom projet, prof...) • Ajout des données dans les auto evals et création des fichiers pour chaque élèves • Ajout des dates formatées dans les auto evals • Optimisation de l'installation automatique des packages • Création d'une VM pour les tests • Protection des cellules du modèle	https://stackoverflow.com/questions/2249619/how-to-format-a-datetime-in-powershell
Rédaction JNLTRAV	1		
INFORMER : Analyse du projet	4	• Rédaction du rapport	
Total semaine	90	Max. 90	

Figure 44 Journal de travail semaine 1

Semaine	2	Début: lundi, 30 janvier 2023	
Tâche	Durée [1/4 h.]	Explications: qu'est-ce qui se fait et comment ?	Liens, références, ...
		Lundi 24p	
REALISER : Data (Collecte, Modélisation), Programmation	20	<ul style="list-style-type: none"> Ajout de vérification des données en input par l'utilisateur (Fichiers existants, données présentes...) Optimisation de la modification en ajoutant des constantes dans un fichier Config Ajout du logging des actions réalisées Vérification que excel est installé 	<ul style="list-style-type: none"> http://ramblingcookiemonster.github.io/PowerShell-Configuration-Data/ https://learn.microsoft.com/en-us/powershell/scripting/developer/module/how-to-write-a-powershell-module-manifest?view=powershell-7.3 https://social.technet.microsoft.com/Forums/office/en-US/e5d8594b-b14b-4a54-913c-61089b5d9ab4/release-or-delete-a-com-object-from-powershell?forum=winserverpowershell https://learn.microsoft.com/en-us/powershell/module/microsoft.powershell.host/start-transcript?view=powershell-7.3 https://learn-powershell.net/2015/04/09/quick-hits-finding-exception-types-with-powershell/
Rédaction JNLTRAV	2		
TESTER : mise en	2	* Test de génération sur un environnement clean (VM) et avec des valeurs différentes dans les	
		Mercredi 27	
REALISER : Data	11	* Optimisation de la gestion d'erreurs	https://stackoverflow.com/questions/41700636/
INFORMER : Analyse	2	* Discussion avec M-Gruaz à propos du projet	
Autre	9	* Cours se finissant plus tôt	
TESTER : mise en	4	* Test de la génération des auto evals sur 2 machines de l'etmi et une VM	
Rédaction JNLTRAV	1		
		Jeudi 12	
Rédaction JNLTRAV	2		
REALISER : Data (Collecte, Modélisation), Programmation	10	* Ajout d'une fonction de recherche de cellule par nom	
		Vendredi 27	
REALISER : Data (Collecte, Modélisation), Programmation	19	<ul style="list-style-type: none"> Récupération des données et des notes des auto evals Mise en commun des AutoEvals dans un seul fichier Problème de liaison de fichiers excel 	<ul style="list-style-type: none"> https://stackoverflow.com/questions/3226096/copy-excel-worksheet-from-one-workbook-to-another-with-powershell https://learn.microsoft.com/en-us/answers/questions/864246/powershell-vba-and-excel
INFORMER : Analyse du projet	8	* Rapport	
Total semaine	90	Max. 90	

Figure 45 Journal de travail semaine 2

Semaine 3		Début: lundi, 6 février 2023	
Tâche	Durée [1/4 h.]	Explications: qu'est-ce qui se fait et comment ?	Liens, références, ...
Lundi 24p			
REALISER : Data (Collecte, Modélisation), Programmation	23	<ul style="list-style-type: none"> Résolution d'un problème de liaisons de fichier Excel lors de la génération de la synthèse Modification de la création de la synthèse : Les auto evals sont copiés dans le fichier modèle puis ce fichiers est "Save As" pour créer le fichier final Création d'une macro pour rapatrier les notes 	https://support.microsoft.com/en-us/topic/macro-to-loop-through-all-worksheets-in-a-workbook-feef14e3-97cf-00e2-538b-5da40186e2b0 https://vba1.com/table/clear-table-content/ https://www.automateexcel.com/vba/string-data-type/ https://officetuts.net/excel/vba/add-data-to-table-using-vba/ https://support.microsoft.com/en-us/topic/macro-to-loop-through-all-worksheets-in-a-workbook-feef14e3-97cf-00e2-538b-5da40186e2b0
Rédaction JNLTRAV	1		
Mercredi 27			
REALISER : Data (Collecte, Modélisation), Programmation	15	<ul style="list-style-type: none"> Export des auto evals en PDF 	<ul style="list-style-type: none"> https://wellsr.com/vba/2016/excel/vba-replace-function-to-replace-characters-in-string/
Autre	3	<ul style="list-style-type: none"> Présentation de M.Gruaz sur les critères d'évaluation des TPI 	
TESTER : mise en commun des réalisations	7	<ul style="list-style-type: none"> Test des scripts sur une VM avec Office 	
Rédaction JNLTRAV	2		
Jeudi 12			
INFORMER : Analyse du projet	12	Documentation	
Vendredi 27			
REALISER : Data (Collecte, Modélisation), Programmation	19	<ul style="list-style-type: none"> Tentative d'une interface graphique en XAML pour lancer les scripts cela n'a pas fonctionné, je vais passer sur du Windows.Forms avec Powershell car je l'ai déjà fait pour un ancien projet 	
INFORMER : Analyse du projet	7	Documentation	
Rédaction JNLTRAV	1		
Total semaine	90	Max. 90	

Figure 46 Journal de travail semaine 3

Semaine	4	Début: lundi, 13 février 2023	
Tâche	Durée [1/4 h.]	Explications: qu'est-ce qui se fait et comment ?	Liens, références, ...
		Lundi 24p	
Vacances	24	Vacances	
		Mercredi 27	
Vacances	27	Vacances	
		Jeudi 12	
Vacances	12	Vacances	
		Vendredi 27	
Vacances	27	Vacances	
Total semaine	90	Max. 90	

Figure 47 Journal de travail semaine 4

Semaine	5	Début: lundi, 20 février 2023	
Tâche	Durée [1/4 h.]	Explications: qu'est-ce qui se fait et comment ?	Liens, références, ...
		Lundi 24p	
Vacances	24	Congé	
		Mercredi 27	
Autre	2	• Présentation de M.Gruaz sur les critères d'évaluation des TPI	
INFORMER : Analyse du projet	12	• Rapport	
REALISER : Data (Collecte, Modélisation), Programmation	13	• Création d'une interface graphique en Windows.Form pour la création ou la récupération des auto-évaluations	https://learn.microsoft.com/en-us/powershell/scripting/samples/creating-a-custom-input-box?view=powershell-7.3 https://social.technet.microsoft.com/wiki/contents/articles/25911-how-to-add-a-powershell-gui-event-handler-part-1.aspx https://stackoverflow.com/questions/15120597/passing-multiple-values-to-a-single-powershell-script-parameter
		Jeudi 12	
INFORMER : Analyse du projet	12	• Rapport	
		Vendredi 27	
INFORMER : Analyse du projet	26	• Rapport	
Rédaction JNLTRAV	1		
Total semaine	90	Max. 90	

Figure 48 Journal de travail semaine 5

Semaine 6		Début: lundi, 27 février 2023	
Tâche	Durée [1/4 h.]	Explications: qu'est-ce qui se fait et comment ?	Liens, références, ...
		Lundi 24p	
Autre	3	Présentation de M.Gruaz sur les critères d'évaluation des TPI	
REALISER : Data (Collecte, Modélisation), Programmation	15	<ul style="list-style-type: none"> • Ajout de messages d'erreurs selon chaque cas possible • Gestion des erreurs • Ajout des entêtes des fichiers de script 	
TESTER : mise en commun des réalisations	6	Test du programme en environnement VM	
		Mercredi 27	
REALISER : Data (Collecte, Modélisation), Programmation	18	<ul style="list-style-type: none"> • Ajout des entêtes des fichiers de script • Ajout du mode avancé pour modifier les chemins • Mise à jour du script de compilation avec des constantes 	<ul style="list-style-type: none"> • https://social.technet.microsoft.com/wiki/contents/articles/28746-powershell-gui-how-to-create-a-link-label-using-winforms.aspx • https://stackoverflow.com/questions/7976646/powershell-store-entire-text-file-contents-in-variable
TESTER : mise en commun des réalisations	5	• Test des scripts sur plusieurs machines de la classe + une VM	
INFORMER : Analyse du projet	3	• Documentation	
Rédaction JNLTRAV	1		
		Jeudi 12	
REALISER : Data (Collecte, Modélisation), Programmation	11	<ul style="list-style-type: none"> • Ajout des entêtes des fichiers de script • Optimisation du mode avancé • Ajout d'un fichier de constantes 	<ul style="list-style-type: none"> • https://stackoverflow.com/questions/6808456/powershell-wpf-boolean-resets-its-value-after-function • https://powershellbyexample.dev/post/custom-object
Rédaction JNLTRAV	1		
		Vendredi 27	
TESTER : mise en commun des réalisations	3	• Test des scripts sur plusieurs machines de la classe + une VM	
INFORMER : Analyse du projet	24	<ul style="list-style-type: none"> • Documentation utilisateur • Rapport 	
Total semaine	90	Max. 90	

Figure 49 Journal de travail semaine 6

Semaine 7		Début: lundi, 6 mars 2023	
Tâche	Durée [1/4 h.]	Explications: qu'est-ce qui se fait et comment ?	Liens, références, ...
Lundi 24p			
INFORMER : Analyse du projet	15	• Documentation utilisateur	
REALISER : Data (Collecte, Modélisation), Programmation	8	• Ajout d'une notification toast lors du lancement des scripts	<ul style="list-style-type: none"> • https://www.starwindsoftware.com/blog/how-to-display-notifications-on-windows-10-using-powershell • https://den.dev/blog/powershell-windows-notification/
Rédaction JNLTRAV	1		
Mercredi 27			
Rédaction JNLTRAV	1		
EVALUER : voir cahier des charges	20		
INFORMER : Analyse du projet	6	<ul style="list-style-type: none"> • Manuel utilisateur • Manuel modification des scripts • Manuel Compilation des scripts • Manuel modification du modèle d'auto-évaluations 	
Jeudi 12			
INFORMER : Analyse du projet	12	• Manuel modification du modèle d'auto-évaluations	
Vendredi 27			
INFORMER : Analyse du projet	25	<ul style="list-style-type: none"> • Dossier de conception • Dossier de réalisation 	
Rédaction JNLTRAV	1		
Autre	1	Création d'un script d'envoi de mails pour Monsieur Lymberis	
Total semaine	90	Max. 90	

Figure 50 Journal de travail semaine 7

Semaine 8		Début: lundi, 13 mars 2023	
Tâche	Durée [1/4 h.]	Explications: qu'est-ce qui se fait et comment ?	Liens, références, ...
INFORMER : Analyse du projet	43	<ul style="list-style-type: none"> Dossier de réalisation Présentation 	
Rédaction JNLTRAV	1		
Autre	46	Présentations	
Total semaine	90	Max. 90	
Semaine 9		Début: lundi, 20 mars 2023	
Tâche	Durée [1/4 h.]	Explications: qu'est-ce qui se fait et comment ?	Liens, références, ...
Imprévu	90	Semaine en moins pour le projet	
Total semaine	90	Max. 90	

Figure 51 Journal de travail semaine 8

5.4 Planification détaillée

Semaine 1			
Tâche	Durée [1/4 h.]	Explications: qu'est-ce qui se fait et comment ?	Liens, références, ...
		Lundi 24p	
Rédaction JNLTRAV	1	Rédaction du journal de travail	
Autre	23	Présentations de retour de stage	
		Mercredi 27	
INFORMER : Analyse du projet	15	Analyse du cahier des charges	
Rédaction JNLTRAV	1	Rédaction du journal de travail	
PLANIFIER : Découper en tâches et concevoir	11	Planification	
		Jeudi 12	
Rédaction JNLTRAV	1	Rédaction du journal de travail	
PLANIFIER : Découper en tâches et concevoir	11	Planification	
		Vendredi 27	
Rédaction JNLTRAV	1	Rédaction du journal de travail	
DECIDER : données, interfaces, langage, etc.	15	Choix de la méthode	
REALISER : Data (Collecte, Modélisation), Programmation	11	Récupération des données dans un fichier excel	
Total semaine	90	Max. 90	

Figure 52 Planification semaine 1

Semaine	2		
Tâche	Durée [1/4 h.]	Explications: qu'est-ce qui se fait et comment ?	Liens, références, ...
		Lundi 24p	
Rédaction JNLTRAV	1		
REALISER : Data (Collecte, Modélisation), Programmation	21	Création des auto-évaluations	
TESTER : mise en commun des réalisations	2	Test du script	
		Mercredi 27	
Rédaction JNLTRAV	1		
REALISER : Data (Collecte, Modélisation), Programmation	26	Création des auto-évaluations	
		Jeudi 12	
Rédaction JNLTRAV	1		
REALISER : Data (Collecte, Modélisation), Programmation	11	Création des auto-évaluations	
		Vendredi 27	
Rédaction JNLTRAV	1		
REALISER : Data (Collecte, Modélisation), Programmation	26	Récupération des auto-évaluations	
Total semaine	90	Max. 90	

Figure 53 Planification semaine 2

Semaine 3			
Tâche	Durée [1/4 h.]	Explications: qu'est-ce qui se fait et comment ?	Liens, références, ...
		Lundi 24p	
Rédaction JNLTRAV	1		
REALISER : Data (Collecte, Modélisation), Programmation	21	Création des auto-évaluations	
TESTER : mise en commun des réalisations	2	Test du script	
		Mercredi 27	
Rédaction JNLTRAV	1		
REALISER : Data (Collecte, Modélisation), Programmation	26	Création des auto-évaluations	
		Jeudi 12	
Rédaction JNLTRAV	1		
REALISER : Data (Collecte, Modélisation), Programmation	11	Création des auto-évaluations	
		Vendredi 27	
Rédaction JNLTRAV	1		
REALISER : Data (Collecte, Modélisation), Programmation	26	Récupération des auto-évaluations	
Total semaine	90	Max. 90	

Figure 54 Planification semaine 3

Semaine 4			
Tâche	Durée [1/4 h.]	Explications: qu'est-ce qui se fait et comment ?	Liens, références, ...
		Lundi 24p	
Vacances	24	Vacances	
		Mercredi 27	
Vacances	27	Vacances	
		Jeudi 12	
Vacances	12	Vacances	
		Vendredi 27	
Vacances	27	Vacances	
Total semaine	90	Max. 90	

Figure 55 Planification semaine 4

Semaine 5			
Tâche	Durée [1/4 h.]	Explications: qu'est-ce qui se fait et comment ?	Liens, références, ...
		Lundi 24p	
Vacances	24		
		Mercredi 27	
Rédaction JNLTRAV	1		
REALISER : Data (Collecte, Modélisation), Programmation	26	Création des auto-évaluations	
		Jeudi 12	
Rédaction JNLTRAV	1		
TESTER : mise en commun des réalisations	11	Création de la synthèse	
		Vendredi 27	
Rédaction JNLTRAV	1		
REALISER : Data (Collecte, Modélisation), Programmation	26	Création de la synthèse	
Total semaine	90	Max. 90	

Figure 56 Planification semaine 5

Semaine 6			
Tâche	Durée [1/4 h.]	Explications: qu'est-ce qui se fait et comment ?	Liens, références, ...
		Lundi 24p	
Rédaction JNLTRAV	1		
INFORMER : Analyse du projet	23	Documentation	
		Mercredi 27	
Rédaction JNLTRAV	1		
REALISER : Data (Collecte, Modélisation), Programmation	26	Interface graphique	
		Jeudi 12	
Rédaction JNLTRAV	1		
REALISER : Data (Collecte, Modélisation), Programmation	11	Interface graphique	
		Vendredi 27	
Rédaction JNLTRAV	1		
TESTER : mise en commun des réalisations	26	Interface graphique	
Total semaine	90	Max. 90	

Figure 57 Planification semaine 6

Semaine 7			
Tâche	Durée [1/4 h.]	Explications: qu'est-ce qui se fait et comment ?	Liens, références, ...
		Lundi 24p	
Rédaction JNLTRAV	1		
TESTER : mise en commun des réalisations	23	Test du programme sur plusieurs environnements	
		Mercredi 27	
Rédaction JNLTRAV	1		
TESTER : mise en commun des réalisations	26	Test du programme sur plusieurs environnements	
		Jeudi 12	
Rédaction JNLTRAV	1		
INFORMER : Analyse du projet	11	Documentation	
		Vendredi 27	
Rédaction JNLTRAV	1		
INFORMER : Analyse du projet	26	Documentation	
Total semaine	90	Max. 90	

Figure 58 Planification semaine 7

Semaine 8			
Tâche	Durée [1/4 h.]	Explications: qu'est-ce qui se fait et comment ?	Liens, références, ...
		Lundi 24p	
Rédaction JNLTRAV	1		
INFORMER : Analyse du projet	23	Documentation	
		Mercredi 27	
Rédaction JNLTRAV	1		
INFORMER : Analyse du projet	26	Documentation	
		Jeudi 12	
Rédaction JNLTRAV	1		
Autre	11	Documentation	
		Vendredi 27	
Rédaction JNLTRAV	1		
INFORMER : Analyse du projet	26	Documentation	
Total semaine	90	Max. 90	

Figure 59 Planification semaine 8

Semaine 9			
Tâche	Durée [1/4 h.]	Explications: qu'est-ce qui se fait et comment ?	Liens, références, ...
		Lundi 24p	
Autre	24	Préparation de la présentation	
		Mercredi 27	
Autre	27	Préparation de la présentation	
		Jeudi 12	
Autre	12	Présentations	
		Vendredi 27	
Autre	27	Présentations	

Figure 60 Planification semaine 9

5.5 Manuel de modifications

5.5.1 Modification des scripts

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

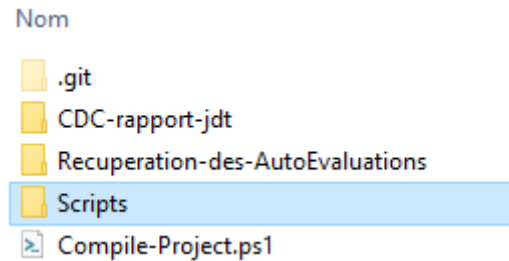


Figure 61 Dossier de scripts

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

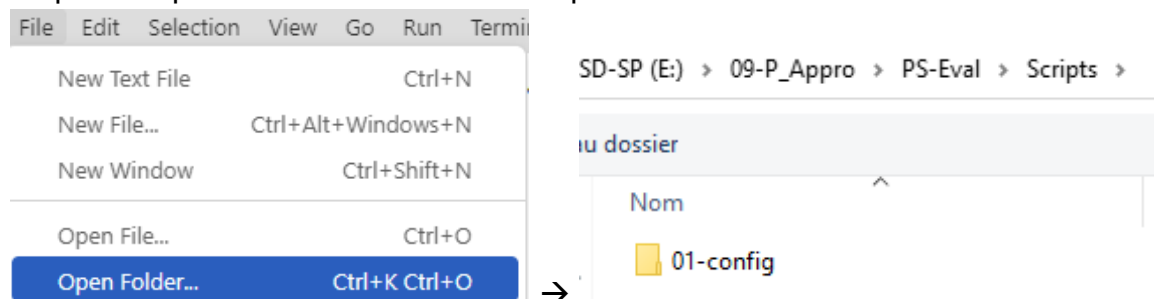


Figure 62 Ouverture du dossier dans VS Code

Figure 63 Dossier de config

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

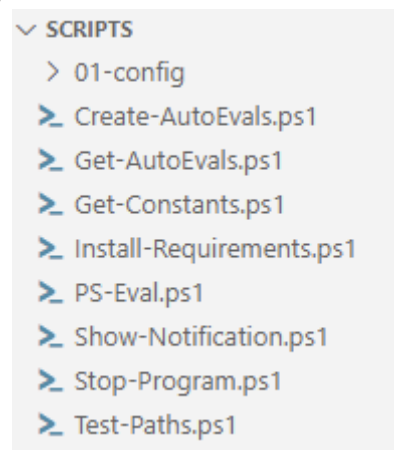


Figure 64 Liste des scripts

⚠ 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 !

5.5.2 Compilation des scripts

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

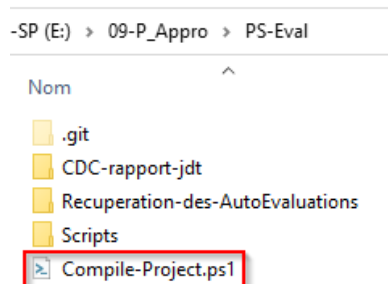


Figure 65 Fichier de configurations

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

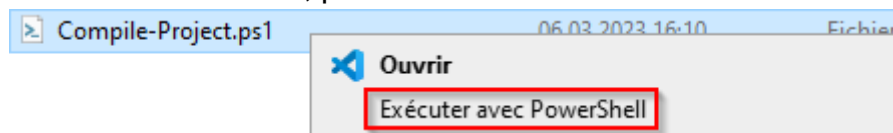


Figure 66 Compilation

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

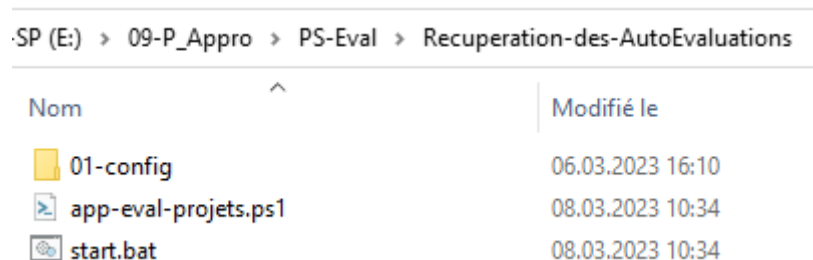


Figure 67 Résultat de la compilation

⚠ 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 !

5.5.3 Modifications du modèle d'auto-évaluations

5.5.3.1 Préparation à la modification

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

.SP (E:) > 09-P_Appro > PS-Eval > Scripts > 01-config	
Nom	Modifié le
01-infos-proj-eleves.xlsx	01.03.2023 09:06
02-modele-grille.xlsx	06.03.2023 16:10
03-synthese-eval.xlsm	01.03.2023 09:06

Figure 68 Fichier modèle

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

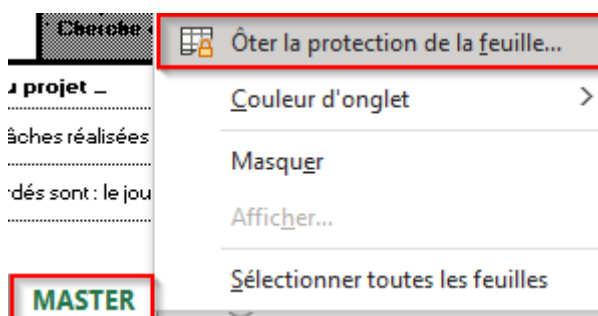


Figure 69 Ôter la protection de la feuille

5.5.3.2 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"

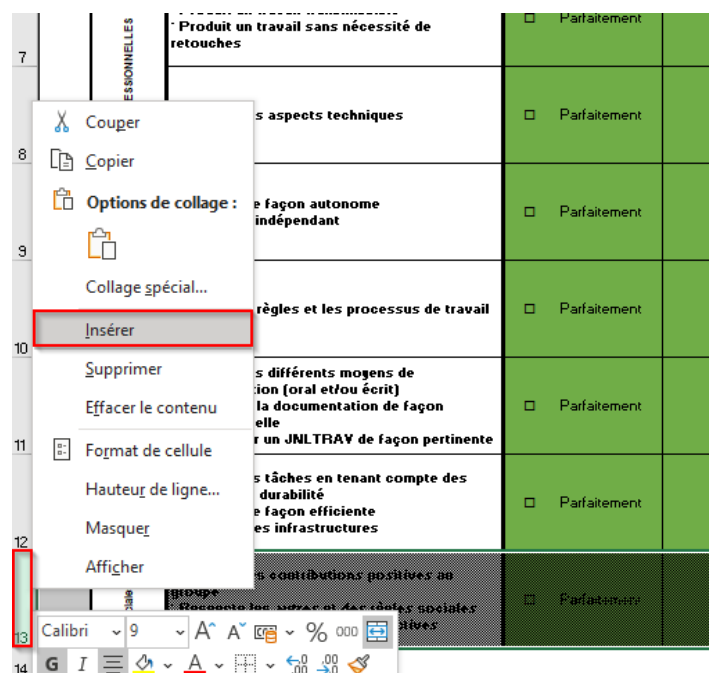


Figure 70 Insertion d'une ligne

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.

12	Réalise ses tâches en tenant compte des principes de durabilité Travaille de façon efficiente Respecte les infrastructures	<input type="checkbox"/> Parfaitement	<input type="checkbox"/> A peu près	<input type="checkbox"/> Pas assez	<input type="checkbox"/> Pas du tout	FAUX	
13							

Figure 71 Duplication des données

Ajoutez ensuite le titre de la compétence et la description

Test	Ceci est une compétence test Une autre compétence	<input type="checkbox"/> Parfaitement	<input type="checkbox"/> A peu près
------	------------------------------------------------------	---------------------------------------	-------------------------------------

Figure 72 Titre et description

5.5.3.3 Modification des champs de titre :

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

Nom et Prénom :	<u>NAME</u>	Nom du projet :	<u>PROJECTNAME</u>
Année de formation - classe :	<u>CLASSROOM</u>	Semaines :	<u>NB WEEKS</u>
Enseignant :	<u>TEACHER</u>	Dates :	<u>DATESMERGED</u>

Figure 73 Données de titre

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

Figure 74 Modification des données de titre

Dans le gestionnaire de noms, supprimez l'ancienne valeur

Figure 75 Modification dans le gestionnaire de noms

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

Ouvrez maintenant le fichier "03-synthese-eval.xlsm"

C > SSD-SP (E:) > 09-P_Appro > PS-Eval > Scripts > 01-config	
Nom	Modifié le
01-infos-proj-eleves.xlsx	01.03.2023 09:06
02-modele-grille.xlsx	09.03.2023 13:17
03-synthese-eval.xlsm	08.03.2023 14:27

Figure 76 Modèle de synthèse

Dans l'onglet "Développeur" cliquez sur "Visual Basic"

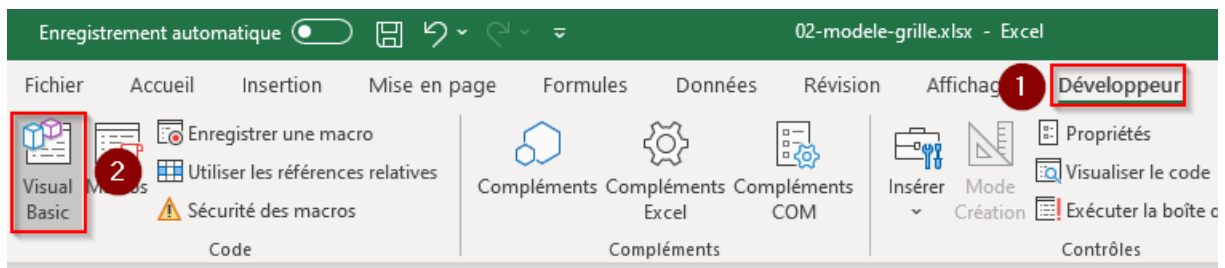


Figure 77 Modification du VBA

Ouvrez l'onglet "VBAProject" puis le module "ModSynthesis".

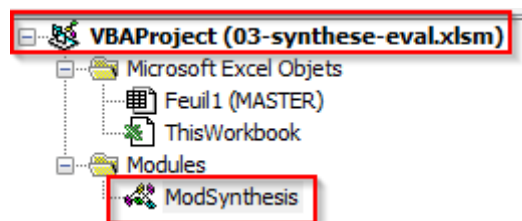


Figure 78 Ouverture du module VBA

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

Enregistrez et fermez ensuite le fichier.

```

name = Current.Range("NAME")
classe = Current.Range("CLASSROOM")
teacher = Current.Range("TEACHER")
projectname = Current.Range("PROJECTNAME")
nbweeks = Current.Range("NBWEEKS")
datesmerged = Current.Range("DATESMERGED")
finalnote = Current.Range("FINALNOTE")

```

Figure 79 Modification des données de titre dans le VBA

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

```

141 #Replace the cells with the configs datas
142 $Sheet1.cells.find("NAME") = "$($student.Prenom) $($student.Nom)"
143 $Sheet1.cells.find("CLASSROOM") = $ConfigsHash[$CLASSROOM]
144 $Sheet1.cells.find("TEACHER") = $ConfigsHash[$TEACHER]
145 $Sheet1.cells.find("PROJECTNAME") = $ConfigsHash[$PROJECTNAME]
146 $Sheet1.cells.find("NBWEEKS") = $ConfigsHash[$NBWEEKS]
147 $Sheet1.cells.find("DATESMERGED") = "$($ConfigsHash[$DATESSTART]).Tc
148

```

Figure 80 Modification des données de titre dans le PowerShell

5.5.3.4 Protection de la feuille

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

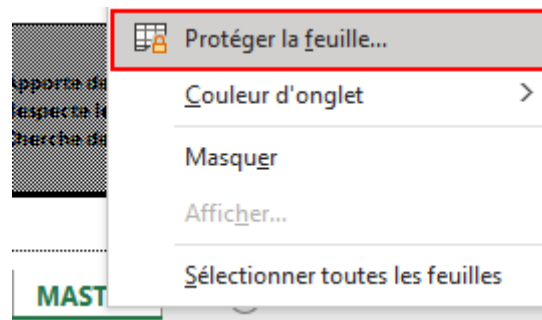


Figure 81 Protéger la feuille

Sélectionnez les checkbox suivantes, puis cliquez sur "OK"

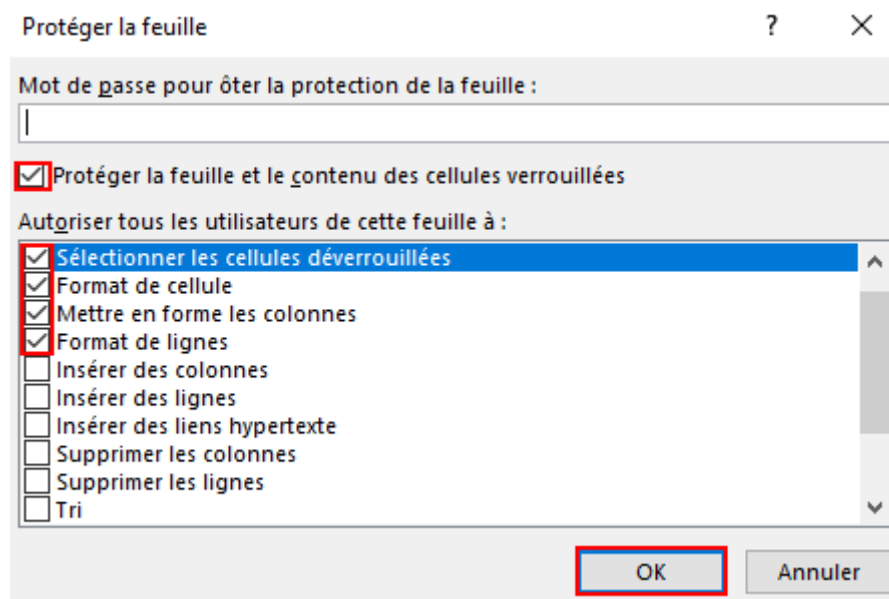


Figure 82 Sélection des options de verrouillage

5.6 Manuel d'Utilisation

5.6.1 Téléchargement

Pour télécharger le programme vous le trouverez sur [GitHub](#). Cliquez sur "Code" puis "Download ZIP"

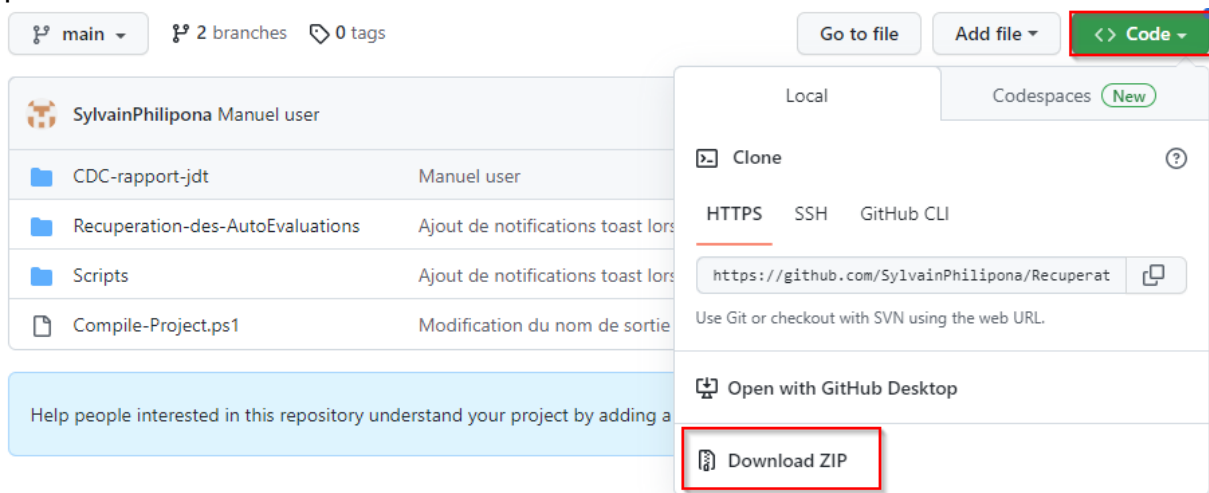


Figure 83 Téléchargement du projet

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"

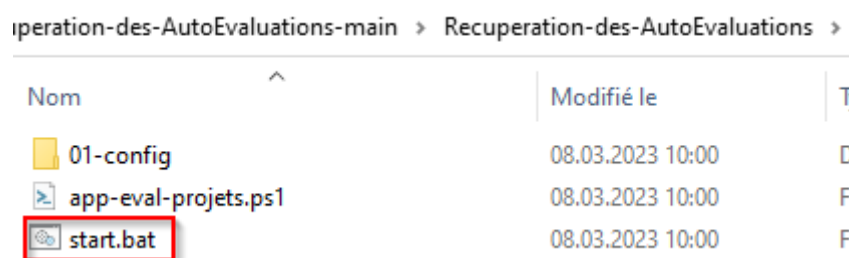


Figure 84 Fichier de démarrage

Si vous avez l'option "Débloquer" cliquez dessus. Sinon faites uniquement "OK"

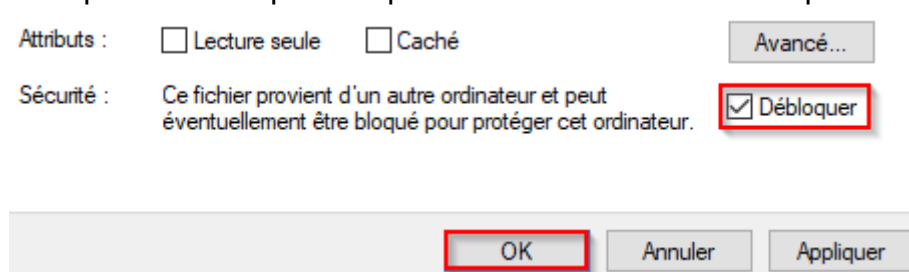


Figure 85 Déblocage du fichier de démarrage

⚠ La première exécution du programme peut être particulièrement lente.

5.6.2 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



Figure 86 Dossier du projet

Figure 87 Dossier renommé

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

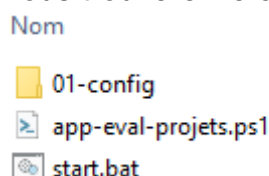


Figure 88 Contenu du dossier

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

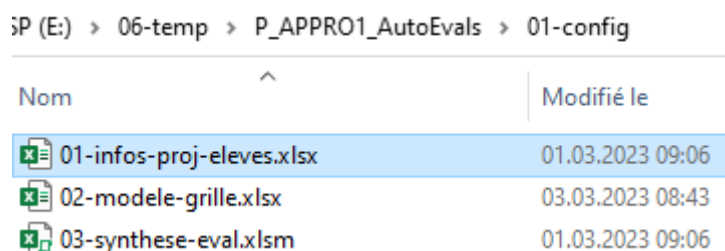
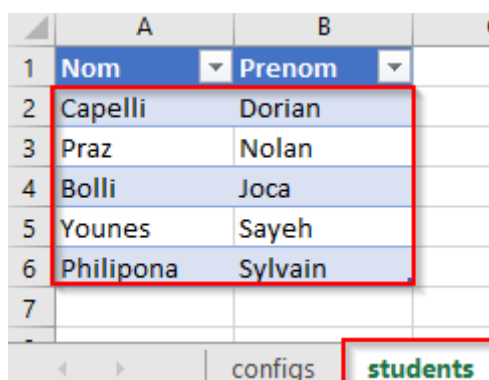


Figure 89 Fichier de configuration

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



	A	B	C
1	Nom	Prenom	
2	Capelli	Dorian	
3	Praz	Nolan	
4	Bolli	Joca	
5	Younes	Sayeh	
6	Philipona	Sylvain	
7			

Figure 90 Liste des élèves

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

	A	B
1	Champs	Valeurs
2	Nom du projet	P_Appro
3	Enseignant	Gilbert Gruaz
4	Visa Enseignant	GGZ
5	Date debut	23.01.2023
6	Date fin	24.03.2023
7	Nbr de périodes total	300
8	Nbr de semaines	10
9	Classe	CIN4b
10	Evaluation numéro	1
11		

← → configs students

Figure 91 Configurations du projet

5.6.3 Création des auto-évaluations

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

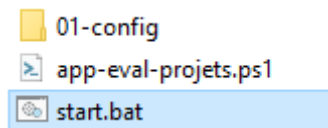


Figure 92 Démarrage du programme

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

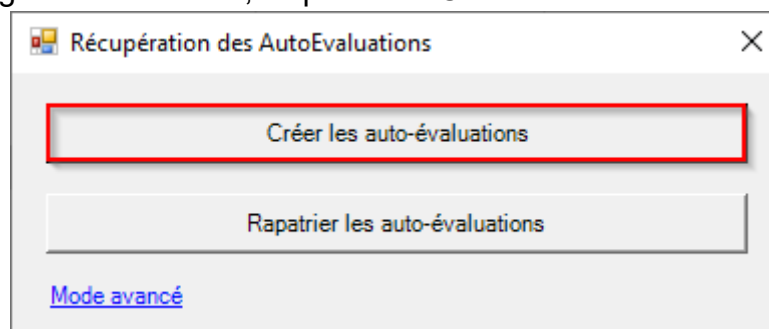


Figure 93 Menu de création

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

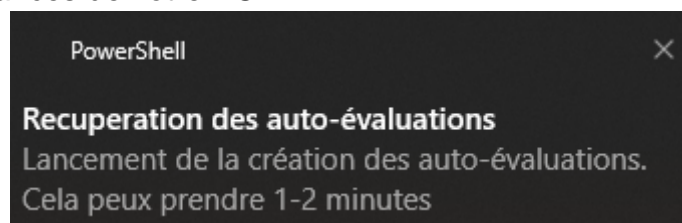


Figure 94 Notification Toast

Une fois les auto-évaluations générées, ce message apparaîtra. Cliquez sur "OK"

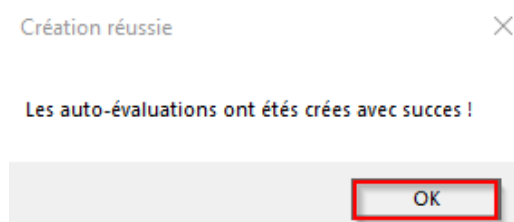


Figure 95 Pop-up de résultat

Un dossier "02-evaluations" sera créé contenant toutes les auto-évaluations. Envoyez les auto-évaluations aux élèves pour qu'ils les remplissent.

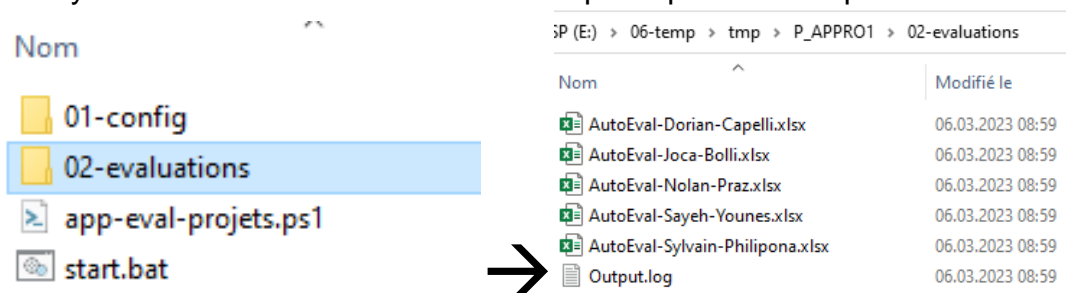


Figure 96 Dossier de sortie

Figure 97 Contenu du dossier de sortie

5.6.4 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"

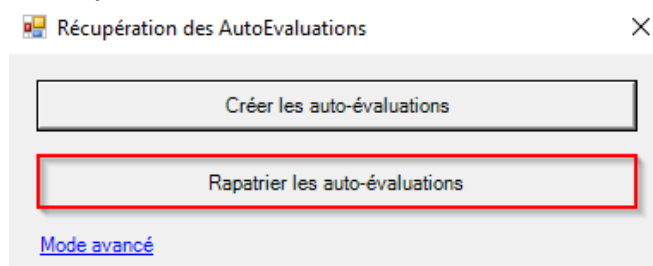


Figure 98 Menu de rapatriement

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

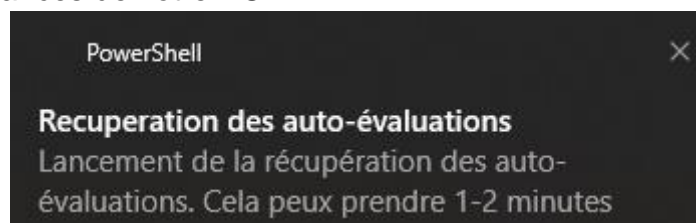


Figure 99 Notification Toast

Une fois les auto-évaluations récupérées, ce message apparaîtra. Cliquez sur "OK"

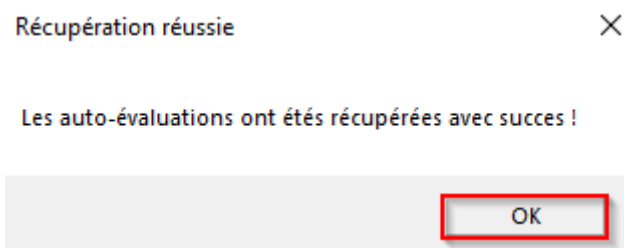


Figure 100 Pop-up de résultat

Rendez-vous dans le dossier "02-evaluations"

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

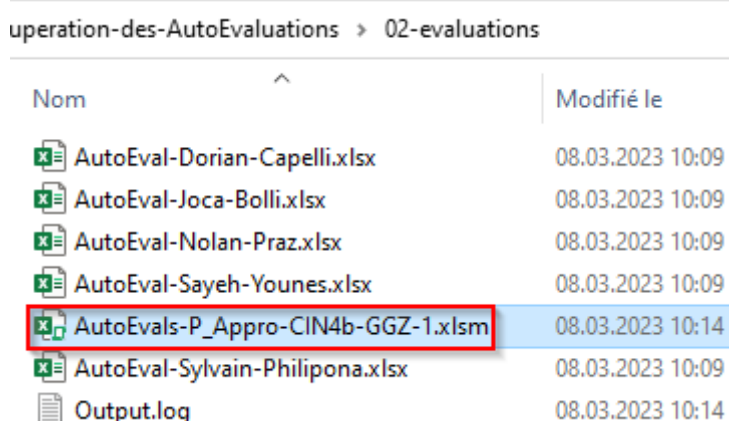


Figure 101 Fichier de synthèse

Une fois ouvert pensez à activer les macros

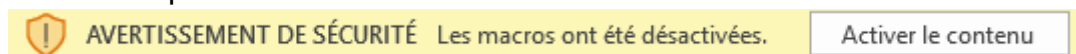


Figure 102 Activation des macros

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"

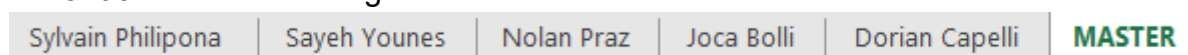


Figure 103 Onglets lors de la synthèse

Dans l'onglet "MASTER", cliquez sur "Rapatrier les notes" et "Export en PDF"

[illegible]

Figure 104 Rapatriement et export des auto-évaluations

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

Eleve	Evaluation rendue	Note Finale
Sylvain Philipona	Oui	LA
Sayeh Younes	Oui	LA
Nolan Praz	Oui	A
Joca Bolli	Oui	LA
Dorian Capelli	Oui	PA

Figure 105 Synthèse des notes

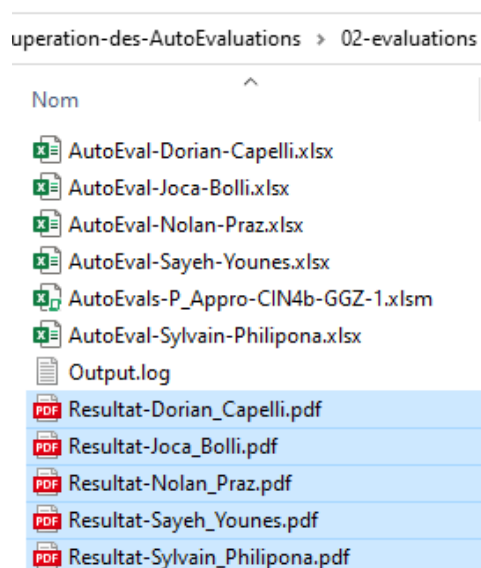


Figure 106 Exports en PDF

5.7 Archives du projet

Lien vers le répertoire Git du projet :

<https://github.com/SylvainPhilipona/Recuperation-des-AutoEvaluations>