# On n’avait pas cours avec McGonagall ? N°13

**1. Introduction**

L’objectif de ce projet était de créer un **exécutable Windows** capable de récupérer l’emploi du temps d’un utilisateur via l’API de **WigorServices**, en utilisant soit les identifiants de connexion, soit un fichier de **cookies**.

Le projet devait également répondre aux exigences suivantes :

* Être entièrement documenté.
* Être testé avec une couverture unitaire supérieure à 80 %.
* Utiliser des technologies adaptées au défi (Python, Tkinter, Requests, BeautifulSoup).
* Fournir une justification claire des choix techniques.

**2. Choix techniques et justification**

1. **Langage et environnement**
   * **Python 3.11+** a été choisi pour sa simplicité et la richesse de ses bibliothèques, notamment pour le web scraping et la génération d’interfaces GUI.
   * La création d’un **.exe Windows** se fait via **PyInstaller**, garantissant la portabilité de l’application sur toutes les machines Windows sans installation préalable de Python.
2. **Interface utilisateur**
   * **Tkinter** a été utilisé pour sa simplicité et son intégration native avec Python.
   * L’interface permet :
     + de sélectionner un fichier cookies.
     + de saisir l’URL complète de l’emploi du temps (avec hash).
     + d’afficher l’emploi du temps récupéré ou de sauvegarder le HTML pour debug en cas d’erreur.
3. **Communication avec l’API Wigor**
   * **Requests** gère la session HTTP et l’envoi de cookies.
   * Les headers HTTP incluent un **User-Agent** et un **Referer** pour éviter le blocage serveur.
4. **Analyse du contenu HTML**
   * **BeautifulSoup** extrait les éléments pertinents des pages HTML retournées par Wigor.
   * Chaque cours est récupéré sous forme de dictionnaire { "raw\_title": ... }.
5. **Débogage et export HTML**
   * En cas d’absence de cours, le HTML complet est sauvegardé sur le bureau et ouvert automatiquement dans le navigateur, facilitant l’inspection manuelle.

**3. Architecture et fonctionnement**

L’application suit l’architecture suivante :

[GUI Tkinter] --> [Lecture Cookies & URL] --> [Session Requests] --> [Récupération HTML] --> [Parsing BeautifulSoup] --> [Affichage ou Export HTML]

* La classe principale WigorClient gère les sessions HTTP et le parsing.
* Les fonctions auxiliaires permettent la lecture de cookies depuis un fichier texte et la sauvegarde automatique du HTML pour debug.

**4. Tests unitaires et couverture**

* Les fonctions critiques ont été testées avec **unittest**, incluant :
  + Lecture et parsing des cookies.
  + Récupération d’HTML simulé et extraction des cours.
* Les tests ont permis d’obtenir une **couverture > 80 %**, validant le bon fonctionnement des modules clés.

Une image contenant texte, capture d’écran, logiciel, Logiciel multimédia

Le contenu généré par l’IA peut être incorrect.

**5. Utilisation**

1. Lancer l’exécutable wigor\_client\_gui.exe.
2. Sélectionner le fichier cookies via l’interface.
3. Coller l’URL complète de l’emploi du temps.
4. L’application affiche les cours ou sauvegarde le HTML pour debug.

**6. Conclusion**

Le projet permet à un utilisateur Windows de récupérer rapidement son emploi du temps via WigorServices de manière sécurisée et portable.

* Les choix techniques (Python, Tkinter, Requests, BeautifulSoup) sont cohérents et adaptés au besoin.
* La génération d’un exécutable Windows facilite la distribution.
* La sauvegarde automatique du HTML et l’ouverture dans un navigateur garantissent un **débogage simple et efficace**.

Le projet est ainsi **documenté, testé et fonctionnel**, répondant aux critères du défi.

**7. Annexes**

* Exemple de fichier cookies.

Cookie.txt

ASP.NET\_SessionId=j5abhprozxet4bwg2tnpdxzb; .DotNetCasClientAuth=611E5648861909DB2DB6528E8BF2BE9A5BF47989DF39D42BEF1285773B8E254B3DFF1D867A0D97D0BC78DF9EE83F4018D227869BFF775D989C1D63B5C0AB00ABE00C7E2F04D8440688941F0F955A92690907B9E3A031991E1DDCB5EBC362EE04DAF479D5C9AA11B420C5D3B7A59ED27D80256BA304794C77EDB9FD3AC0C39924B3ED1D9A4CA74323F811DA5509C9C82F35947E823A64D57FF16CE0D13DCD7AFA798AA0514D1AAA6CFEC00B88130698458938446FB6A20D936C1DCE532A0DBBC35C671DB28447ED53DCC7E7632E2D6A3A

* Exemple d’URL Wigor complète.

https://ws-edt-cd.wigorservices.net/WebPsDyn.aspx?action=posEDTLMS&serverID=C&Tel=amandine.jolibois&date=10/13/2025&hashURL=C07D065A638EB6602CCBCC1CFB39D1C6492459F1939257F84AB62BA2C8F80618B9EB6932DE1FD0C08A9B06D2CA70F814514F81AB548396767EC49C9C8721E728

* Capture d’écran de l’interface graphique.

Une image contenant texte, capture d’écran, Police, nombre

Le contenu généré par l’IA peut être incorrect.

* Exemple de fichier HTML sauvegardé pour debug.

Une image contenant texte, capture d’écran, logiciel, Icône d’ordinateur

Le contenu généré par l’IA peut être incorrect.