TD 2 — Prompt structuré et petit workflow

1. Préparer le terrain

Cette séance prolonge le TD 1. Vous allez travailler sur **un document technique que vous connaissez bien** : c'est la meilleure façon de repérer les erreurs d'un assistant.

> **Remarque**
>
> Le travail principal (sections 1 à 4) se fait en autonomie ou en début de séance 3. La section « Pour aller plus loin » (script de contrôle) est facultative.

Exercice 1 — Choisir son document

Choisir un texte de 2 à 5 pages, de votre domaine (informatique, systèmes d'information, gestion de projet…).

Par exemple :

- un extrait du cahier des charges ou du rapport de l'un de vos projets de L3 ou d'avant ;
- la documentation d'une bibliothèque ou d'une API que vous utilisez (README, guide de prise en main) ;
- un extrait de RFC ou de norme (HTTP, JSON, SQL…) ;
- un article d'informatique en accès libre (par exemple sur HAL ou arXiv).

Outils : Duck.ai et/ou l'AnythingLLM du cours. Utilisez **au moins deux modèles différents** (un petit modèle et un grand).

> **Remarque**
>
> Ne déposez que des documents que vous avez le droit de transmettre à un service en ligne : **pas de données personnelles, pas de code ou de document confidentiel** de stage ou d'entreprise.

2. Un prompt qui produit une sortie exploitable

Exercice 2 — Fiche d'extraction structurée

L'objectif est d'obtenir une fiche toujours au même format, sans texte autour, que l'on pourrait donner à un autre programme.

Déposez (ou collez) votre document puis demandez une **fiche au format JSON** en suivant ce gabarit :

```
{
  "titre": "",
  "auteurs": [],
  "annee": null,
```

```
  "probleme_traite": "",
  "technologies": [],
  "methode": "",
  "resultats_principaux": [],
  "limites_signalees": []
}
```

Dans votre prompt, précisez :

1. le rôle (par exemple : « ingénieur logiciel chargé d'une veille technique ») ;
2. que la réponse ne contient **que** le JSON, sans introduction ni balises Markdown ;
3. que, si une information n'est pas dans le texte, la valeur est `null` ou une liste vide : **ne rien inventer**.

Puis, pour **chaque modèle**, complétez le tableau.

Champ	Modèle 1 : exact / partiel / inventé	Modèle 2 : exact / partiel / inventé
titre		
auteurs		
année		
problème traité		
technologies		
méthode		
résultats		
limites		

Réponse :

Exercice 3 — Exiger des preuves

Obtenir pour chaque champ le passage du texte qui le justifie, puis le vérifier.

Relancez la demande en ajoutant : « pour chaque champ, donne la **citation exacte** du texte qui le justifie, entre guillemets ».

Ensuite, pour chaque citation, cherchez-la dans votre document (recherche de texte).

Citations trouvées telles quelles	Citations modifiées ou introuvables

Réponse :

3. Petit workflow en trois étapes

Exercice 4 — Extraction, vérification, synthèse

Enchaîner trois prompts spécialisés et repérer où les erreurs survivent.

1. **Étape 1 – extraction** : reprenez votre fiche JSON de l'exercice précédent (modèle 1).
2. **Étape 2 – vérification** : avec un **autre modèle**, donnez le document et la fiche, et demandez : « compare cette fiche au texte et liste **uniquement les écarts**, en citant le passage du texte ».
3. **Étape 3 – synthèse** : à partir de la fiche corrigée, demandez une synthèse de 10 lignes pour **une cible précise** (au choix : un chef de projet, un client non informaticien, un étudiant de L1).

Répondez ensuite :

1. L'étape 2 a-t-elle trouvé tous les écarts que vous aviez repérés vous-même ? Lesquels sont passés ?
2. L'étape 3 a-t-elle introduit des éléments absents de la fiche ?
3. À quel endroit placeriez-vous **obligatoirement** un contrôle humain ?
4. Dessinez le workflow (étapes, entrées, sorties) comme un petit schéma de traitement : quelles étapes pourraient être **automatisées par un script**, lesquelles non ?

Réponse :

4. Allers-retours : la dérive

Exercice 5 — Reformuler, reformuler, reformuler

Observer ce qui se perd quand un document passe plusieurs fois par un modèle.

Prenez un paragraphe de 10 à 15 lignes de votre document, de préférence avec des **valeurs précises** (versions, délais, limites, codes d'erreur, complexités). Demandez au modèle :

1. de le réécrire en langage simple ;
2. de réécrire ce résultat de façon experte ;
3. de refaire encore une fois ces deux étapes.

Comparez la version finale au paragraphe d'origine et notez ce qui a changé (valeurs, noms propres, restrictions, nuances).

Réponse :

5. Pour aller plus loin : contrôler par programme

Facultatif. Ces exercices appliquent à la sortie d'un assistant ce que l'on fait pour toute donnée qui entre dans une application : la **valider par programme**.

Exercice 6 — Valider la sortie par programme

Ne pas se fier à l'œil : écrire un petit contrôleur qui accepte ou rejette la fiche.

Écrivez (vous-même, ou avec l'assistant, puis relisez-le) un script Python qui lit la réponse du modèle dans un fichier et vérifie que :

1. c'est du **JSON valide** (`json.loads`) ;
2. toutes les **clés du gabarit** sont présentes, et **aucune autre** ;
3. les **types** sont corrects (`auteurs` et `technologies` sont des listes, `annee` est un entier ou `null`…).

Appliquez-le aux réponses des deux modèles, plusieurs fois chacun, puis complétez.

Essais	Modèle 1	Modèle 2
Fiche acceptée (sur 5 essais)		
Cause principale des rejets		

Que faut-il prévoir dans une application qui exploite cette sortie quand elle est rejetée ?

Réponse :

Exercice 7 — Vérifier les citations par programme

Automatiser le contrôle des citations de l'exercice « Exiger des preuves ».

Complétez votre script pour tester si chaque citation est contenue dans le document (après avoir normalisé les espaces et les retours à la ligne), et comparez avec votre vérification manuelle. Pourquoi une citation correcte peut-elle être rejetée par le script ?

Réponse :

Ce document est distribué selon les termes de la licence **Creative Commons BY-ND 4.0**. Vous pouvez trouver le texte de la licence à l'adresse suivante : https://creativecommons.org/licenses/by-nd/4.0/

Il utilise la police **Atkinson Hyperlegible Next** (Braille Institute of America, Inc.) afin d'améliorer la lisibilité.

ASSISTÉ PAR L'IA