École Nationale Supérieure d'Informatique pour l'Industrie et l'Entreprise

Rapport de stage de première année

Étudiant : **KEFIF Bilal**
Encadrant académique : **BANNOUR Fetia**

Entities : **idIA tech** 4 Le Porche, 30730 Montpezat

Encadrants de stage : **BOUR Xavier**

Dates : **2 Juin, 2025 - 31 Juillet, 2025**

Remerciements

Merci à Idiatech de m'avoir accueillit durant la durée de mon stage et de m'avoir appris autant de choses. C'était mes premiers pas dans le monde de l'IT, où j'avais de réelles responsabilités et je remercie encore mon maître de stage, Xavier Bour de m'avoir fait confiance et de m'avoir accompagner durant cette période.

Résumé

Au cours de ce stage, j'ai intégré l'équipe technique d'idIA Tech afin de contribuer à deux axes complémentaires. Le premier consistait à développer des scripts Grimport pour différents clients, automatisant l'import de catalogues fournisseurs dans leurs CMS Prestashop. Ce travail a impliqué l'analyse des sites sources (rétro-ingénierie), l'extraction des informations pertinentes, leur transformation et leur intégration automatisée.

Le second axe concernait la remise en état du débogueur intégré à l'IDE Grimport, dont le code source Groovy/Java modifié par mes prédécesseurs était devenu incompilable. J'ai recherché et intégré manuellement de nombreuses dépendances manquantes, décommenté et corrigé du code, et configuré un environnement de compilation sous Eclipse avec Gradle.

Ces deux volets m'ont permis d'acquérir des compétences techniques en web mining, compilation de projets complexes et maintenance d'outils internes, tout en développant mon autonomie et ma capacité à m'adapter à de nouveaux langages en m'appuyant sur la documentation technique.

Glossaire

API (Application Programming Interface) Interface de programmation qui permet à deux applications de communiquer entre elles.

CMS (Content Management System) Système de gestion de contenu permettant de créer et gérer des sites web, comme Prestashop.

Grimport Langage et IDE propriétaire développé par idIA Tech pour automatiser l'import de catalogues fournisseurs dans les CMS.

IDE (Integrated Development Environment) Environnement de développement intégré regroupant un éditeur de code, un compilateur et des outils de débogage.

Gradle Outil de build et de gestion des dépendances pour les projets Java et Groovy.

Groovy Langage de programmation orienté objet basé sur la JVM, offrant une syntaxe souple et dynamique proche de Java.

Jar (Java ARchive) Format de fichier permettant de regrouper des classes Java compilées et leurs ressources dans une seule archive exécutable.

Captcha (Completely Automated Public Turing test to tell Computers and Humans Apart) Test permettant de différencier un utilisateur humain d'un programme automatisé.

Fiddler Outil d'analyse réseau permettant d'intercepter et d'examiner les requêtes HTTP/HTTPS.

Firefox Developer Edition Version du navigateur Firefox destinée aux développeurs, intégrant des outils avancés d'inspection et de débogage.

Maven Repository Dépôt public (mvnrepository.com) contenant des bibliothèques Java et Groovy sous forme de .jar.

Retro-engineering Analyse d'un site web ou d'une application existante afin d'en comprendre la structure et d'automatiser des actions.

Plugin GDT (Groovy Development Tools) Extension Eclipse permettant la prise en charge du langage Groovy dans l'IDE.

Table des figures

2.1 Présentation de l'IDE 4

3.1 Script Grimport simple 6
3.2 Exemple simple d'initialisation de cookie 8
3.3 Réponse serveur affichant une erreur 500 Internal Server Error 10

A.1 Interface cloud de l'IDE Grimport 16

Table des matières

Remerciements i

Résumé ii

Glossaire iii

Listes des Figures iv

1 Introduction : Environnement de travail et stage 1
- 1.1 idIA tech . . . 1
- 1.2 Département de travail . . . 1
- 1.3 Contexte / Problème . . . 1
- 1.4 Objectifs du stage . . . 2
- 1.5 Principales contributions . . . 2

2 Contexte et Préliminaires 3
- 2.1 Cahier des charges . . . 3
- 2.2 Méthodes et outils utilisés dans l'entreprise . . . 3
- 2.3 Intégration et positionnement du stage . . . 4

3 Développement des scripts clients 6
- 3.1 Exemple d'un script simple d'ajout de produit . . . 6
- 3.2 Problèmes rencontrés . . . 7
- 3.3 Outils à disposition . . . 8
- 3.4 Mise en œuvre des solutions . . . 9
  - 3.4.1 Démarche générale . . . 9
  - 3.4.2 Détails techniques . . . 9
  - 3.4.3 Résultats et discussion . . . 10

4 Débogueur de l'IDE Grimport 12
- 4.1 Contexte du débogueur . . . 12
- 4.2 Difficultés rencontrées . . . 12
- 4.3 Démarche adoptée . . . 12
- 4.4 Résultats et limites . . . 13

5 Conclusion 14
- 5.1 Conclusion . . . 14
- 5.2 Leçons retenues . . . 14

5.3 Perspectives et étapes suivantes 14

A Développement Durable et Responsabilité Sociétale (DD&RS) 16
A.1 Impacts environnementaux 16
A.1.1 Activité de l'entreprise 16
A.1.2 Entreprise elle-même 16
A.2 Actions pour minimiser les impacts 17
A.2.1 Actions intégrées aux stratégies de l'entreprise 17
A.2.2 Actions en lien avec le stage et la stratégie de l'entreprise 17

Bibliographie 17

Chapitre 1

Introduction : Environnement de travail et stage

1.1 idIA tech

Fondée en 2009, idIA Tech est une société française spécialisée dans le web-mining, évoluant dans le secteur des technologies de l'information et de l'Internet. C'est une société à taille humaine, donc très proche de ses salariés et ses clients.

Son siège est situé à Montpezat, en Occitanie mais elle dispose également de plusieurs annexes en France, notamment à Bourg-la-Reine.

idIA Tech concentre son activité sur l'extraction de données issues de catalogues fournisseurs, particulièrement dans le domaine du e-commerce, qui automatise le travail des e-commerçants en important des catalogues entier de produits. Dans un contexte où le e-commerce est en forte croissance, l'automatisation de l'import de catalogues constitue un avantage compétitif essentiel pour les clients d'idIA Tech. Pour ce faire, elle s'apppuie sur un crawleur dynamique et sur Grimport, un langage de programmation développé par leurs équipes facilitant l'automatisation de l'intégration de données dans des CMS tels que Prestashop.

1.2 Département de travail

idIA Tech étant une PME (petite moyenne entreprise), l'entreprise ne dispose pas de départements structurés comme dans les grandes organisations. L'ensemble des projets est porté par une équipe réduite de développeurs, et de stagiaires avec qui j'étais en étroite collaboration, sous contrôle de notre maitre de stage.

1.3 Contexte / Problème

Mon stage s'inscrivait dans ce double contexte : traiter des dossiers clients en développant des scripts Grimport pour l'import de catalogues et contribuer à l'amélioration technique de l'IDE L'utilisation de Grimport facilite l'automatisation de l'import de catalogues fournisseurs, mais l'approche par crawler soulève plusieurs difficultés techniques au niveau du crawler.

Le travail sur le débogueur Grimport a présenté un autre type de difficultés, liées cette fois aux technologies utilisées vieillissante donc soulevait des problèmes de compatibilités. Celui ci bloquait les évolutions de l'IDE, ce qui freinait la modernisation de l'outil, et donc compliquait la conception des scripts Grimport.

1.4 Objectifs du stage

L'objectif principal de mon stage était de m'intégrer à l'équipe en tant qu'apprenti développeur Grimport. Pouvoir gérer des dossiers clients seul et répondre à leur cahier des charges . Parallèlement, un objectif plus personnel m'a été confié : travailler sur l'amélioration de l'IDE Grimport. Résoudre le problème de débogueur qui bloquait la mise à jour de l'IDE.

1.5 Principales contributions

Au cours de mon stage, j'ai d'abord contribué en tant que développeur Grimport à la réalisation de plusieurs scripts destinés à différents clients. Ces scripts permettaient d'importer automatiquement leurs catalogues fournisseurs dans leurs CMS, en particulier Prestashop, et d'adapter les extractions aux besoins propres à chaque projet. En parallèle, j'ai travaillé sur l'amélioration technique de l'IDE Grimport, en m'attaquant aux erreurs Java liées au code source Groovy modifié par mes prédécesseurs. Mon objectif était de parvenir à compiler ce code en .jar, de régler tous les problèmes de librairie et de dépendances que le projet débogueur contenait et de l'intégrer au projet, afin de restaurer le bon fonctionnement du débogueur.

Chapitre 2

Contexte et Préliminaires

2.1 Cahier des charges

Le cahier des charges de mon stage comportait deux volets :

— **M**ission principale (dossiers clients) : Développer, en Grimport, des scripts capables d'extraire automatiquement les informations contenues dans les catalogues fournisseurs (titres, descriptions, caractéristiques, images des produits ou encore avis des clients existants en plusieurs langues) et de les importer dans le CMS Prestashop de chaque client via l'analyse de structure des sites web et de retro-engineering. Les solutions mises en place devaient être robustes, adaptées aux spécificités de chaque site fournisseur et fournir un résultat fonctionnel ce qui nécessitait de bonnes connaissances en web.

— **M**ission secondaire (IDE Grimport) : Analyser et corriger le code source Groovy/Java du débogueur, afin de compiler une version fonctionnelle en .jar et de l'intégrer correctement au projet. L'objectif était de lever les problèmes liés aux dépendances et aux bibliothèques manquantes, hérités des modifications apportées par mes prédécesseurs sur l'IDE eclispe.

2.2 Méthodes et outils utilisés dans l'entreprise

Je travaillais principalement en autonomie, surtout sur les dossiers clients où je m'occupais également de la relation client et discutais avec afin d'affiner le cahier des charges qu'il me fournissaient. Mais pour le projet débogueur, j'étais en collaboration avec d'autres développeurs et stagiaires de l'équipe.
Les principaux outils et technologies utilisés étaient :**Grimport :** langage et IDE propriétaire de l'entreprise, utilisé pour développer les scripts de crawling et d'intégration.

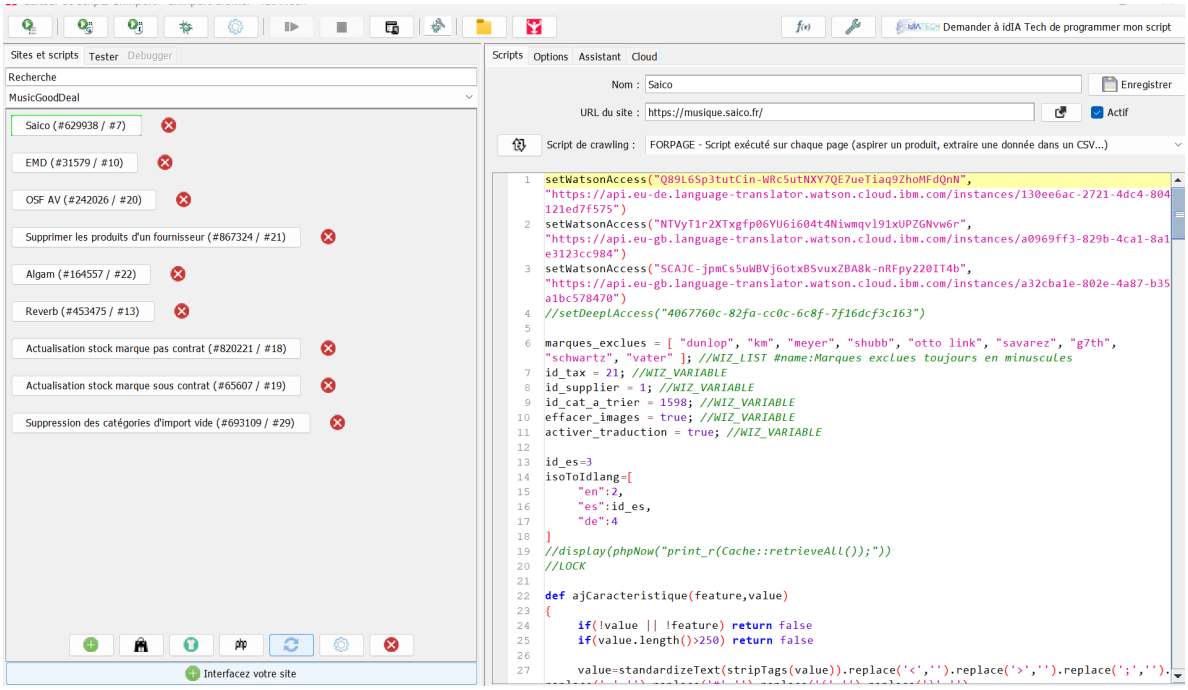

FIGURE 2.1 – Présentation de l'IDE

—

— **Prestashop :** CMS e-commerce utilisé par la majorité des clients d'idIA Tech.

— **Firefox Developer Edition :** afin d'analyser le code source des sites fournisseurs et identifier les éléments pertinents (sélecteurs CSS, Regex).

— **Fiddler Classic :** outil de capture réseau permettant de récupérer et analyser les requêtes effectuées par le crawler, afin de comprendre l'origine et la nature des erreurs rencontrées.

— **Notepad++ :** utilisé pour comparer les différences entre plusieurs requêtes réseau, faciliter l'analyse de ces requêtes et écrire les regex dont j'avais besoin

— **Gradle :** outil de build et de gestion des dépendances, utilisé pour la compilation du code Groovy.

— **Java / Groovy :** langages nécessaires pour analyser et corriger le code source du débogueur Grimport.

2.3 Intégration et positionnement du stage

Dès mon arrivée, j'ai suivi une formation interne sur Grimport et son IDE. Grâce à cette montée en compétences rapide, j'ai pu prendre en charge directement des dossiers clients et développer des scripts d'import adaptés à leurs besoins.

Concernant l'IDE, le débogueur avait déjà été développé par mes prédécesseurs. Toutefois, pour parvenir à compiler le projet, une partie du code source Groovy avait été mise en commentaires, ce qui entraînait des bugs et des incohérences..
Mon rôle a donc consisté à discuter avec ceux qui étaient sur la mission avant moi pour comprendre d'où pourrait venir les problèmes et rétablir un code fonctionnel, à décommenter proprement ces parties, puis à résoudre les problèmes restants liés aux librairies, aux dépendances et aux erreurs Java, afin d'obtenir une version compilable et exploitable du débogueur.

Chapitre 3

Développement des scripts clients

3.1 Exemple d'un script simple d'ajout de produit

Avant d'aborder les difficultés rencontrées, il est utile de présenter un exemple de script de base développé avec Grimport. Celui-ci illustre la méthode générale utilisée pour extraire les informations d'un produit sur un site fournisseur et les importer automatiquement dans un CMS comme Prestashop.

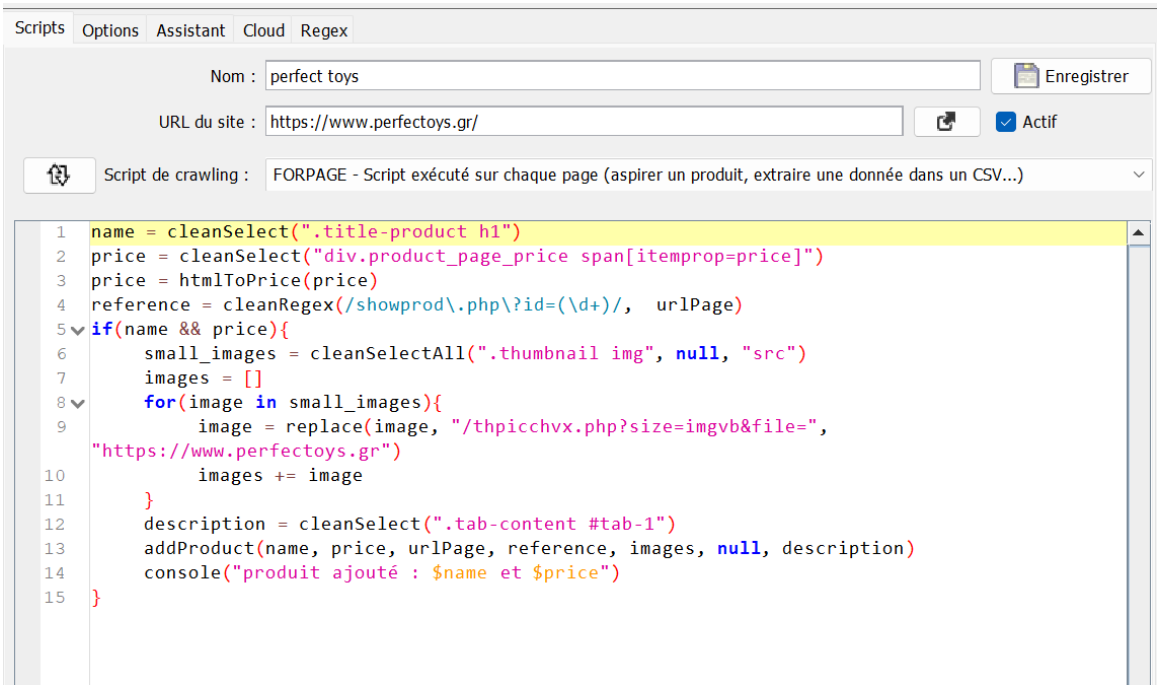

FIGURE 3.1 – Script Grimport simple

La démarche suivie repose sur quatre étapes principales :

— **Analyse de la structure du site fournisseur :** Grâce aux outils de développement intégrés dans Firefox ou n'importe quel autre navigateur, j'identifie les balises HTML et les sélec-

teurs CSS contenant les informations pertinentes (nom du produit, prix, description, images). Lorsque nécessaire, des expressions régulières (Regex) permettent d'extraire des valeurs spécifiques, comme une référence produit dans l'URL.

— **Récupération des données :** Le script utilise des fonctions Grimport pour extraire les textes associés aux balises HTML. On peut aussi utiliser cleanRegex où on sélectionne directement une partie du code source grâce aux expressions régulières.

— **Transformation et formatage :** Les données brutes extraites doivent être mises en forme pour correspondre aux standards du CMS. Par exemple, le prix récupéré en texte HTML est converti en valeur numérique (htmlToPrice), et les liens d'images sont complétés pour être valides.

— **Import dans le CMS :** Les fonctions addProduct ou UpdateoraddProduct permet de communiquer avec le CMS et de lui fournir l'ensemble des informations collectées (nom, prix, description, images, référence produit, etc.). Chaque produit est ainsi automatiquement ajouté au catalogue du client ou alors modifé s'il existe déja, sans intervention manuelle.

Ce processus illustre la logique générale d'un script Grimport : analyser la structure du site (phase de retro engineering), extraire les informations pertinentes, transformer les données et les transmettre au CMS cible. Sur un site simple, ce type de script permet déjà de gagner un temps considérable en automatisant une tâche répétitive qui serait très coûteuse manuellement.

3.2 Problèmes rencontrés

Lors de la conception et de l'adaptation des scripts Grimport pour les clients, plusieurs difficultés techniques se sont présentées :

— **Traduction des descriptions produits** Certains cahiers des charges exigeaient que les produits et leurs descriptions soient disponibles en plusieurs langues. Or, de nombreux sites fournisseurs ne proposaient pas de versions traduites, et les solutions de traduction automatique donnaient des résultats de qualité insuffisante.

— **Présence de Captchas** Certains sites utilisaient des Captchas afin de bloquer l'accès automatisé aux données, ce qui empêchait le crawler d'extraire les informations produits.

— **Navigation du crawler** Sur certains sites, le crawler n'était pas capable de distinguer correctement les différents onglets et rencontrait des difficultés pour accéder aux fiches produits.

— **Reconnaissance des pages produits** Le crawler ne différenciait pas toujours les véritables fiches produits des autres pages du site (panier, page d'accueil, conditions générales, etc.), ce qui perturbait l'import.

— **Accès restreint aux revendeurs** Certains fournisseurs limitaient leurs catalogues aux revendeurs connectés. L'accès aux pages produits nécessitait alors une authentification préalable et la gestion des cookies de session.

3.3 Outils à disposition

Pour mener à bien le développement des scripts clients, j'ai pu m'appuyer sur un ensemble d'outils et de fonctions déjà implémentés dans Grimport IDE. Ces outils étaient pensés pour faciliter le web-mining, l'extraction de données et l'intégration dans les CMS.

Fonctions natives de Grimport : De nombreuses fonctions de base permettent de naviguer dans une page web et d'extraire des informations :

— cleanSelect et cleanSelectAll : récupérer du texte ou des ensembles d'éléments à partir de sélecteurs CSS.

— cleanRegex : extraire des valeurs à partir d'expressions régulières (par exemple une référence produit dans une URL).

— htmlToPrice : convertir un texte en valeur numérique pour les prix.

— addProduct : fonction clé qui envoie les informations collectées (nom, prix, images, description, etc.) vers le CMS Prestashop.

Gestion des cookies :
Plusieurs fonctions intégrées permettent de manipuler les cookies et donc de gérer des connexions ou des sessions utilisateur :

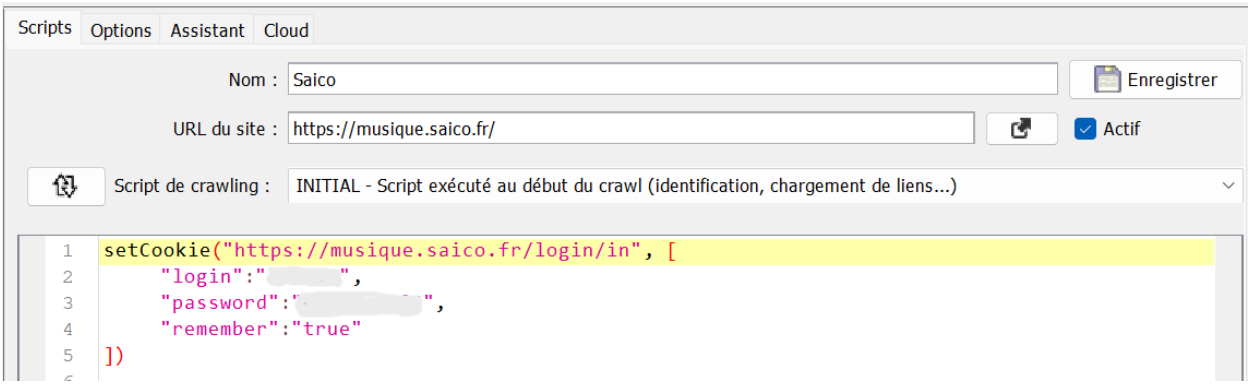

FIGURE 3.2 – Exemple simple d'initialisation de cookie

— getCookies, getCookieValue : récupération des cookies d'un domaine.

— setCookie, addCookie : modification ou ajout de cookies pour simuler une session.

— clearCookie, clearAllCookies : suppression de cookies pour repartir d'une navigation vierge.

Interaction avec Firefox Developer Edition :
Grimport propose une fonction firefox() qui permet d'interagir directement avec un navigateur Firefox Developer Edition.
Il y'a plusieurs actions possibles : naviguer vers une URL, exécuter du JavaScript, récupérer le code source de la page (sourceCode), ou encore gérer les frames et les cookies via le navigateur.
Cet outil est utile lorsque l'identification est complexe ou que le site bloque les méthodes classiques d'extraction.

Utilisation d'APIs externes :

Grimport intègre aussi des fonctions pour interagir avec des services tiers :

— OpenAI API via gptQuestion : permet de poser des questions en langage naturel et de recevoir une réponse générée par un modèle d'intelligence artificielle.

— API antiCaptcha : permet de résoudre automatiquement des Captchas grâce à un service externe.

3.4 Mise en œuvre des solutions

3.4.1 Démarche générale

Pour résoudre les difficultés rencontrées lors du développement des scripts clients, j'ai adopté une approche pragmatique consistant à combiner différents outils et fonctions déjà disponibles dans Grimport IDE. Mon travail s'articulait autour de trois volets :

— L'analyse des requêtes réseau pour comprendre et contourner les blocages liés aux Captchas et à l'authentification.

— L'utilisation d'APIs externes pour la traduction et la reformulation de contenus.

— L'exploitation de fonctions avancées de Grimport (gestion des cookies, interaction avec Firefox, exécution de scripts JavaScript) afin d'automatiser au maximum les scénarios de navigation.

3.4.2 Détails techniques

Captchas et problèmes de cookies de connexion :
Lorsque le crawler rencontrait un Captcha ou une authentification bloquée, je lançais une analyse réseau avec Fiddler. Cet outil intercepte les requêtes HTTP et permet d'identifier les causes d'erreur. Très souvent, les Captchas et les problèmes de cookies généraient des erreurs HTTP POST 500, liées à une mauvaise gestion de la requête côté serveur (session invalide, cookie manquant ou expiré). Cette analyse me permettait d'adapter mes scripts en conséquence.

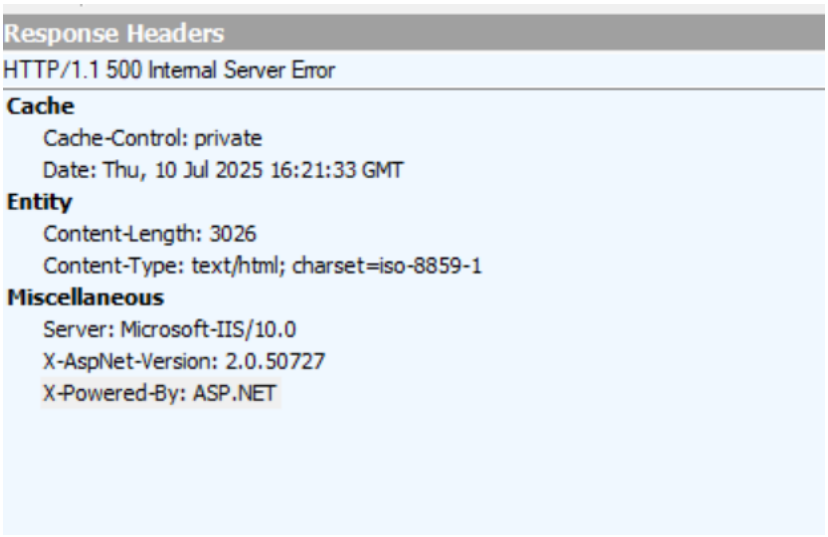

FIGURE 3.3 – Réponse serveur affichant une erreur 500 Internal Server Error

Traduction et reformulation :
Pour répondre aux demandes de traduction multilingue des descriptions produits, il aurait été possible d'utiliser l'API Deepl. Toutefois, nous avons préféré recourir à l'API ChatGPT via la fonction gptQuestion, car elle permet de prendre en compte le contexte du texte source et de produire une traduction plus ciblée et plus naturelle. De plus, ChatGPT pouvait être utilisé non seulement pour traduire, mais aussi pour reformuler certains contenus, ce qui enrichissait la qualité des descriptions importées.

Connexion et authentification :
Deux méthodes principales étaient possibles pour gérer la connexion aux sites réservés aux revendeurs :

— Avec setCookie : en injectant directement les informations de connexion (login, mot de passe, paramètre de session) dans le domaine concerné, comme dans l'exemple ci-dessous :

— Avec firefox() et JavaScript : En exécutant des requêtes JavaScript qui simulaient le remplissage des champs de connexion avant le lancement du crawler. Cette approche permettait d'émuler une authentification manuelle dans le navigateur, tout en gardant un script automatisé.

Navigation manuelle avec Firefox :
Dans certains cas où le crawler ne progressait pas (par exemple lorsque plusieurs fiches produits partageaient la même URL), la solution consistait à contourner le problème en utilisant uniquement la fonction firefox(). Le script parcourait alors le site en s'appuyant directement sur le code source des pages et sur les informations de pagination disponibles. L'extraction se faisait produit par produit, de manière plus proche d'une navigation manuelle, où l'on simulait un clic humain sur page précédente et suivante mais toujours automatisée par script.

3.4.3 Résultats et discussion

L'utilisation combinée de ces outils a permis de débloquer la majorité des situations rencontrées. Les Captchas et problèmes de cookies ont pu être identifiés rapidement grâce à Fiddler, les traductions ont gagné en qualité avec l'API ChatGPT, et l'authentification a été gérée soit par injection de

cookies, soit par automatisation via Firefox.
Néanmoins, ces solutions ne sont pas parfaites :

— l'appel à des APIs externes (ChatGPT, antiCaptcha) génère des coûts et dépend de la disponibilité des services

— l'utilisation intensive de Firefox ralentit les scripts et introduit une dépendance fragile au navigateur

— certaines pages dynamiques très complexes restent difficiles à automatiser complètement.

Malgré ces limites, ces approches ont permis d'obtenir des scripts robustes, capables de traiter la diversité des catalogues fournisseurs rencontrés en conditions réelles.

Chapitre 4

Débogueur de l'IDE Grimport

4.1 Contexte du débogueur

Le débogueur intégré à l'IDE Grimport avait été fortement modifié par mes prédécesseurs. Ces modifications, effectuées sur le code source Groovy/Java, avaient introduit de nombreuses incohérences et rendu le projet incompilable. Plusieurs parties du code avaient été commentées, certaines bibliothèques manquaient, et aucune version binaire précompilée n'était disponible pour la version de Groovy utilisée (2.5.9). Mon objectif était de restaurer une version fonctionnelle et compilable du débogueur.

4.2 Difficultés rencontrées

Plusieurs obstacles sont rapidement apparus :

— **Dépendances manquantes** : de nombreuses bibliothèques étaient absentes du projet, certaines difficiles à trouver sous forme .jar (par exemple `org.apache.tools.ant.BuildFileTest`).
— **Compatibilités Java / Groovy / Gradle** : le projet utilisait une version ancienne de Groovy, plus compatible avec les versions récentes de Java et de Gradle.
— **Code commenté** : certaines portions avaient été désactivées, entraînant des erreurs de cohérence qu'il fallait analyser et corriger.
— **Absence de jar officiel** : depuis Groovy 1.*, il n'existe plus de jar complet prêt à l'emploi sur Maven, seulement des modules dispersés, ce qui compliquait l'intégration.

4.3 Démarche adoptée

Pour résoudre ces problèmes, j'ai adopté une approche progressive :

— Inventaire des erreurs de compilation à partir des logs Gradle.
— Recherche et intégration des dépendances manquantes (*mvnrepository.com*) et centralisation des .jar sur un dépôt GitHub.
— Décommentage progressif du code et corrections des incompatibilités Java/Groovy.
— Tests de compilation avec plusieurs versions de Java (8, 11, 17) pour identifier la combinaison la plus stable.

Mais malgré toutes ces tentatives toujours le même problème lors de la compilation de groovy 2.5.9 à l'aide de Gradle. Donc j'ai décider de remonter directement à la version officielle de groovy en essayant de la recompiler sur ma machine et ainsi ajouter la dépendance au projet débogueur mais impossible de

compiler cette version. Les seules pistes que j'avais était que la version de java bloquait la compilation car il n'est pas censer y'avoir d'erreurs dans la version officielle qu'a sortit groovy.

4.4 Résultats et limites

À l'issue de ce travail :

— Une partie des dépendances manquantes ont été retrouvées et intégrées manuellement.

— L'environnement Eclipse est désormais capable d'ouvrir et de comprendre le projet Groovy/-Java, sans afficher 999+ erreurs à l'écran.

— Le code source du débogueur est plus cohérent et testable prémunisé des erreurs java existantes auparavant.

Cependant, certaines bibliothèques restent introuvables ou incompatibles et empêchent encore la compilation complète du projet en un jar fonctionnel. Le projet n'est donc pas encore totalement exploitable, mais il est désormais documenté et beaucoup plus facile à reprendre.

Chapitre 5

Conclusion

5.1 Conclusion

Ce stage m'a permis de travailler sur deux axes complémentaires. D'une part, le développement de scripts Grimport pour différents clients, automatisant l'import des catalogues fournisseurs dans Prestashop et améliorant ainsi l'efficacité du processus. Cela m'a permit de faire mes premiers pas dans la relation client, établir un cahier des charges et de la charge de travail demandée pour un projet précis.
D'autre part, la reprise et la remise en état du débogueur de l'IDE Grimport, dont le code source Groovy/Java modifié était devenu incompilable, m'a sérieusement sortit de ma zone de confort et m'a pousser à une très grande réflexion et minutie dans tout ce que je modifiais. J'ai réellement pu prendre conscience de mes capacités.
Ces deux volets m'ont conduit à mobiliser des compétences en web-mining, en rétro-ingénierie, en gestion de dépendances et en intégration d'outils dans un environnement complexe.
Le travail réalisé a permis de livrer des scripts robustes et de satisfaire les clients ainsi que de m'avanturer dans un projet java/groovy réellement complexe, ouvrant la voie à sa compilation future.

5.2 Leçons retenues

Ce stage a été une occasion d'apprentissage riche :

— Renforcement des compétences techniques : développement de scripts d'extraction et d'intégration de données, gestion d'APIs externes (traduction, Captcha), manipulation d'outils comme Gradle, Fiddler, Firefox Developer Edition.
— Amélioration notable de ma capacité à m'adapter à différents langages en m'appuyant sur la documentation technique.
— Acquisition d'une méthodologie de diagnostic et de résolution de problèmes sur un projet existant et modifié par d'autres.
— Développement de l'autonomie et de la communication avec les clients.
— Compréhension des enjeux liés à la maintenance et à l'évolution d'un IDE interne dans une entreprise.

5.3 Perspectives et étapes suivantes

Plusieurs pistes peuvent être envisagées pour prolonger ce travail :

— Finaliser la compilation complète du débogueur en identifiant ou en remplaçant les bibliothèques encore manquantes.
— Améliorer la documentation technique du débogueur pour faciliter sa prise en main par de futurs développeurs.

Ces perspectives permettront de consolider les acquis du stage et de poursuivre la modernisation de l'IDE Grimport tout en offrant des outils plus performants aux utilisateurs finaux.

Annexe A

Développement Durable et Responsabilité Sociétale (DD&RS)

A.1 Impacts environnementaux

A.1.1 Activité de l'entreprise

idIA Tech est une société spécialisée dans l'extraction et l'intégration automatisée de données pour le e-commerce. Son activité n'a pas d'impact direct sur l'environnement comparable à celui d'une activité industrielle, mais elle consomme des ressources numériques (serveurs, stockage, bande passante). L'entreprise limite ses impacts en mutualisant l'exécution de ses scripts sur un cloud interne, réduisant ainsi la consommation énergétique des postes de travail des collaborateurs.

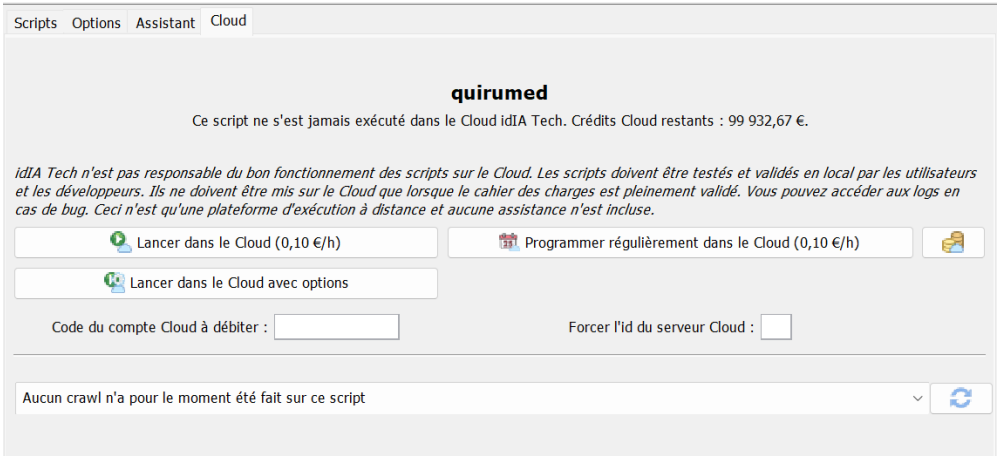

FIGURE A.1 – Interface cloud de l'IDE Grimport

A.1.2 Entreprise elle-même

L'entreprise, à taille humaine, encourage le télétravail et la dématérialisation des échanges, ce qui contribue à réduire les déplacements et l'usage du papier.

A.2 Actions pour minimiser les impacts

A.2.1 Actions intégrées aux stratégies de l'entreprise

idIA Tech a intégré à sa stratégie une démarche de réduction de son empreinte numérique : optimisation du code et mutualisation des serveurs pour limiter la consommation énergétique, choix d'outils open-source lorsque c'est possible pour réduire les coûts et favoriser des solutions pérennes. Cela a un coût pour l'utilisateur donc il est attentif aux scripts qu'il lance en production pour éviter de payer un surplus qu'il aurait pu éviter en rendant son code plus efficace.

A.2.2 Actions en lien avec le stage et la stratégie de l'entreprise

Dans le cadre de mon stage, l'utilisation de l'infrastructure cloud pour tester et exécuter les scripts Grimport a permis de limiter l'usage intensif de mon poste de travail et donc sa consommation électrique. J'ai également documenté mes travaux sous forme numérique, sans impression papier.