

RAPPORT DE STAGE D'ÉTÉ

Présenté en vue de l'obtention de la

LICENCE EN SCIENCES INFORMATIQUE

Parcours : Génie Logiciel

Par

YESSIN TOUMI

Une application de validation des demandes d'achats

Réalisé au sein de Ooredoo



Anné Universitaire : 2024-2025

Remerciements

Je tiens tout d'abord à remercier la direction Achats et Logistique de la société Ooredoo de m'avoir accueilli dans ses locaux et ses bureaux.

Par ailleurs, je voudrais remercier mon encadreur Mr Walid SANAA, Directeur Adjoint Achats Techniques, pour sa disponibilité, son sens de former et son assistance qui ont assuré le bon déroulement de mon stage.

Je tiens également à remercier tout particulièrement et à témoigner toute ma reconnaissance aux techniciens, principaux intervenants dans l'expérience enrichissante qui m'a été offerte

Table des matières

Remerciements	i
Introduction Générale	1
1 Présentation de l'organisme d'accueil et du cadre général	2
1.1 Introduction	2
1.2 Présentation générale de l'organisme d'accueil	2
1.2.1 Ooredoo Tunisie	2
1.2.2 Organigramme de la société	2
1.2.3 Problématique	3
1.2.4 Étude de l'existant	3
1.2.5 Solution proposée	4
1.3 Conclusion	4
2 Analyse et Conception	5
2.1 Introduction	5
2.2 Analyse et spécifications des besoins	5
2.2.1 Identification des besoins fonctionnels	5
2.2.2 Identification des besoins non fonctionnels	5
2.2.3 Diagramme de cas d'utilisation globale	6
2.2.4 Diagramme de classe	6
2.3 Diagramme de séquence	7
2.4 Conclusion	8
3 Réalisation	9
3.1 Introduction	9
3.2 Environnement de travail	9
3.2.1 Environnement matériel	9
3.2.2 Environnement Logiciel	10
3.2.3 Technologies utilisées	10
3.3 Modules réalisés	12
3.3.1 Interface de comparaison	12
3.3.2 Lire les fichiers	12
3.3.3 Reconnaître les fichiers	13
3.3.4 Normalisation des données	14
3.3.5 La comparaison	14
3.3.6 La validation	15
3.4 Conclusion	16

Conclusion Générale	17
Bibliographie	18

Table des figures

1.1	Logo de l'entreprise Ooredoo	2
1.2	Organigramme de la société	3
1.3	ERP	3
1.4	Un Demande d'Achat	4
2.1	Diagramme de cas d'utilisation globale	6
2.2	Diagramme de classe	7
2.3	Diagramme de séquence	8
3.1	Logo du library Pandas	11
3.2	Logo du library Openpyxl	12
3.3	L'interface de comparaison	12
3.4	Lire les fichiers	13
3.5	Reconnaître les fichiers	13
3.6	Normalisation des données	14
3.7	La comparaison	15
3.8	La validation	15

Liste des tableaux

3.1	Environnement matériel	9
3.2	Environnement logiciel	10

Introduction Générale

Les avancées dans les nouvelles technologies de l'information et de la communication ont transformé de nombreux processus au sein des entreprises, rendant possibles des améliorations significatives en termes de productivité et d'efficacité. Dans ce contexte, l'automatisation des tâches répétitives est devenue une priorité pour de nombreuses organisations.

Chez la société Ooredoo, la validation des Demandes d'Achat (DA) constitue une étape essentielle du processus d'approvisionnement chez le département d'Achat. Actuellement, cette tâche repose sur une vérification manuelle des données, ce qui peut entraîner des erreurs et une perte de temps considérable.

Afin d'améliorer ce processus, la société a proposé un projet d'automatisation.

L'objectif principal de ce projet est de développer une application permettant d'automatiser le traitement de la comparaison et la validation des données.

Pour mener à bien ce projet, nous adoptons une méthodologie structurée, incluant l'analyse des processus existants, le développement et le test des scripts, ainsi que la documentation et la formation des utilisateurs finaux.

Notre rapport est organisé de la manière suivante :

Chapitre 1 : Présentation de l'organisme d'accueil et du cadre général

Chapitre2 : Analyse et conception

Chapitre3 : Réalisation

Chapitre 1

Présentation de l'organisme d'accueil et du cadre général

1.1 Introduction

Dans ce chapitre introductif, je vais aborder le cadre général de notre projet. Je présente en premier lieu l'organisme d'accueil Ooredoo suivi par une description de l'étude de l'existant et sa critique qui nous a permis de fixer des objectifs à atteindre pour réaliser le travail demandé.

1.2 Présentation générale de l'organisme d'accueil

1.2.1 Ooredoo Tunisie

Ooredoo Tunisie est un opérateur global leader offrant divers services dédiés aussi bien aux particuliers (Mobile, Fixe, data ..) qu'aux entreprises à travers des solutions de hosting, IoT, cloud et très haut débit. En tant qu'entreprise responsable ancrée dans la communauté. Ooredoo Tunisie est guidée par sa vision d'enrichir la vie digitale de ses clients et sa conviction qu'elle peut stimuler la croissance humaine en tirant parti des nouvelles technologies pour aider la communauté à réaliser son plein potentiel.



FIGURE 1.1 – Logo de l'entreprise Ooredoo

1.2.2 Organigramme de la société

L'organigramme de structure de l'entreprise est construit autour de 3 unités.

- Département IT,
- Département Administrative et financière,

— Département Marketing,
La figure 1.2 représente l'organisation hiérarchique du département Administrative et financière

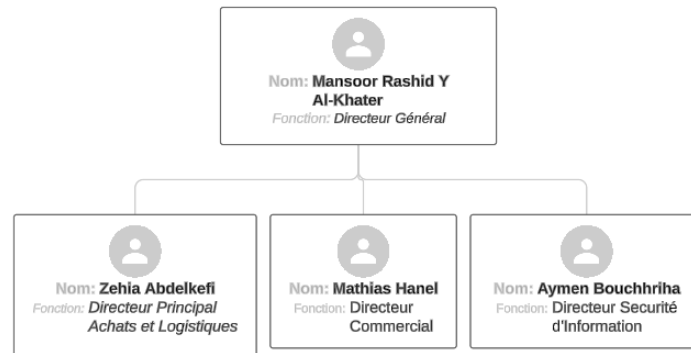


FIGURE 1.2 – Organigramme de la société

1.2.3 Problématique

Le processus de traitement d'achat chez le département Achat Ooredoo consiste à vérifier la conformité des Demandes d'Achat provenant du système ERP de la société avec les contrats : Les CATALOGUES et Les SHOPPING LIST, Cet tâche est très répétitif et il est sujet à des erreurs humaines.

1.2.4 Étude de l'existant

LE SYSTÈME ERP

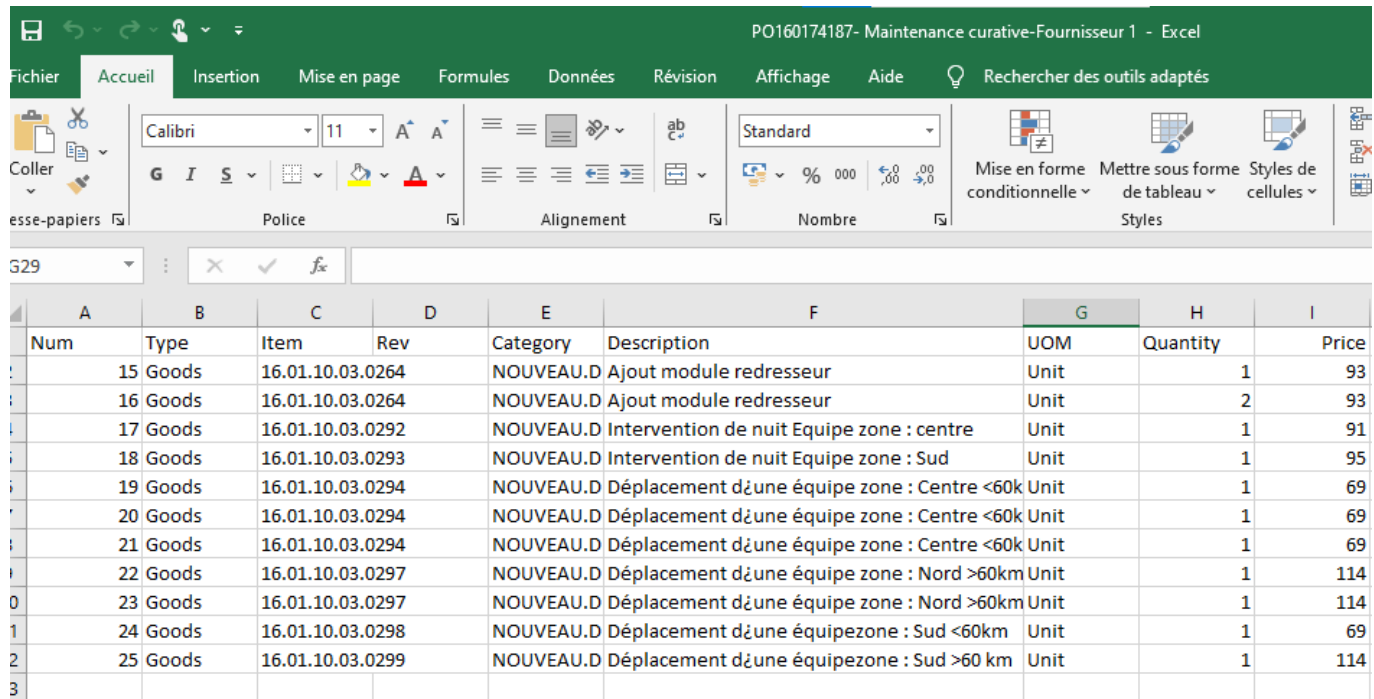
Un ERP (ENTREPRISE RESOURCE PLANNING), ou progiciel de gestion intégré, est un système d'information qui permet de gérer et d'intégrer l'ensemble des processus opérationnels d'une entreprise. Il regroupe différents modules fonctionnels, tels que la gestion des achats, des stocks, des ventes, de la comptabilité, des ressources humaines, et bien plus encore, afin de centraliser les données et d'améliorer la coordination entre les différentes activités de l'entreprise.



FIGURE 1.3 – ERP

LE DÉMARCHE DU PROCESS

Les Demandes d'Achat viennent de l'équipe technique à partir du système ERP , la figure 1.4 est un exemple d'un Demande D'Achat



Num	Type	Item	Rev	Category	Description	UOM	Quantity	Price
15	Goods	16.01.10.03.0264		NOUVEAU.D	Ajout module redresseur	Unit	1	93
16	Goods	16.01.10.03.0264		NOUVEAU.D	Ajout module redresseur	Unit	2	93
17	Goods	16.01.10.03.0292		NOUVEAU.D	Intervention de nuit Equipe zone : centre	Unit	1	91
18	Goods	16.01.10.03.0293		NOUVEAU.D	Intervention de nuit Equipe zone : Sud	Unit	1	95
19	Goods	16.01.10.03.0294		NOUVEAU.D	Déplacement d'une équipe zone : Centre <60k	Unit	1	69
20	Goods	16.01.10.03.0294		NOUVEAU.D	Déplacement d'une équipe zone : Centre <60k	Unit	1	69
21	Goods	16.01.10.03.0294		NOUVEAU.D	Déplacement d'une équipe zone : Centre <60k	Unit	1	69
22	Goods	16.01.10.03.0297		NOUVEAU.D	Déplacement d'une équipe zone : Nord >60km	Unit	1	114
23	Goods	16.01.10.03.0297		NOUVEAU.D	Déplacement d'une équipe zone : Nord >60km	Unit	1	114
24	Goods	16.01.10.03.0298		NOUVEAU.D	Déplacement d'une équipe zone : Sud <60km	Unit	1	69
25	Goods	16.01.10.03.0299		NOUVEAU.D	Déplacement d'une équipe zone : Sud >60 km	Unit	1	114

FIGURE 1.4 – Un Demande d'Achat

L'agent du département Achat intervient pour traiter cette demande d'achat avant qu'elle devienne un bon de commande. Le traitement consiste à vérifier la conformité des Demandes d'Achats avec les contrats de la société avec les fournisseurs (LE CATALOGUE - LE SHOPPING LIST). Si oui , il valide cet demande , sinon le fichier ne passe pas

1.2.5 Solution proposée

Ma mission dans ce projet est de développer un algorithme backend python qui s'intéresse à automatiser le processus mentionné, c'est extraire des données des fichiers qui sont de format Excel et faire le comparatif avec les contrats avant de valider la demande d'achat. Cet solution bénéficie l'entreprise , elle permet de garantir :

- La productivité : allège le travail des employés dû à la possibilité d'automatisation de certains processus.
- La facilité d'utilisation : L'interface de comparaison est simple et intuitive pour faciliter la navigation et l'utilisation pour tous les utilisateurs.

1.3 Conclusion

Au cours de ce chapitre, j'ai présenté la société d'accueil ainsi que ses activités. Ensuite, j'ai dégagé la problématique et exposé une solution. Dans le chapitre suivant, je mène une étude théorique afin d'analyser les besoins et d'étudier la conception générale du système.

Chapitre 2

Analyse et Conception

2.1 Introduction

Dans ce chapitre, Je définis les besoins fondamentaux pour notre application . J'identifie les aspects fonctionnels et non fonctionnels. A travers un diagramme de cas d'utilisation, je visualise les interactions entre l'utilisateur et le système .

2.2 Analyse et spécifications des besoins

2.2.1 Identification des besoins fonctionnels

Les besoins fonctionnels définissent les actions spécifiques que l'application doit permettre aux utilisateurs d'accomplir, telles que consulter les fichiers Excel disponibles, exécuter le script et suivre la validation des fichiers. Ils représentent les fonctionnalités essentielles pour répondre aux besoins de l'utilisateur. Pour garantir le bon fonctionnement de notre application, nous devons identifier les besoins fonctionnels essentiels. Voici les principes besoins du système :

- Lire les fichiers Excel : Le système doit pouvoir lire les fichiers Excel ajoutés par l'utilisateur.
- Identifier les fichiers par leur type (Demande d'Achat , Shopping List , Catalogue)
- Comparer le DA avec les contrats : Comparer chaque article du DA avec lui meme dans les contrats selon les données de l'article.
- Ecrire un fichier Excel : Le rapport de validation sera un fichier Excel(l'output)

2.2.2 Identification des besoins non fonctionnels

Les besoins non fonctionnels décrivent les exigences de performance, de sécurité et d'utilisabilité de l'application. Cela inclut des aspects tels que la sécurité des données, la rapidité de réponse et la convivialité de l'interface. Ces critères sont essentiels pour garantir une expérience utilisateur optimale. Pour assurer la réussite de mon application Desktop, nous nous engageons à respecter les besoins non fonctionnels suivantes :

Performance : Mon application doit être rapide et disponible en permanence.

Sécurité des données : Je dois s'assurer que les informations des fichiers disponibles sont bien protégées. Cela signifie que nous utilisons des mesures de sécurité avancées.

Facilité d'utilisation : L'interface de notre application doit être simple et intuitive, avec des instructions claires pour faciliter la navigation et l'utilisation pour tous les utilisateurs.

2.2.3 Diagramme de cas d'utilisation globale

Le diagramme de cas d'utilisation sert à donner une vision globale du comportement fonctionnel d'un système. Il exprime les besoins des utilisateurs du système, identifie les fonctions les plus importantes et représente les interactions entre les acteurs et leurs attentes vis-à-vis du système. Le diagramme de cas d'utilisation global qui présente l'ensemble des fonctionnalités relatives à notre application est donné par la figure 2.1 .

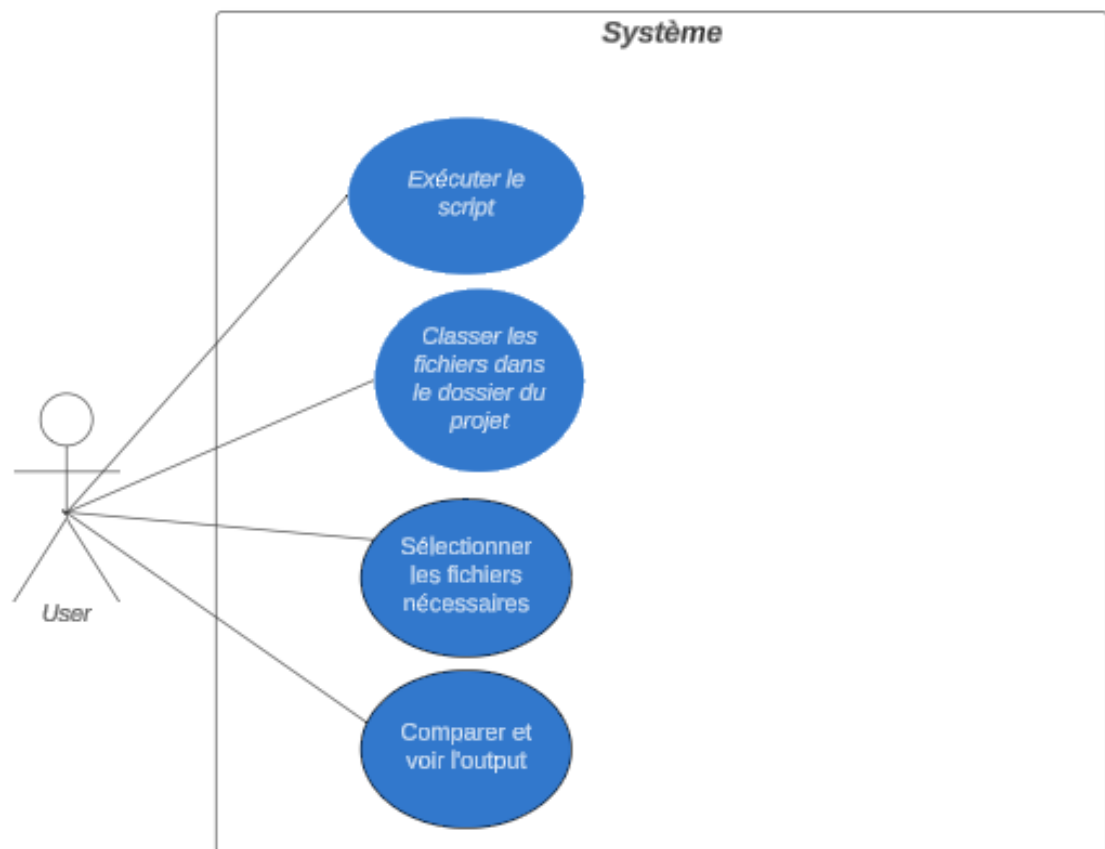


FIGURE 2.1 – Diagramme de cas d'utilisation globale

2.2.4 Diagramme de classe

Le diagramme de classe est un schéma qui décrit l'aspect statique du système. Il permet de fournir les objets du système interagissant en vue de réaliser les cas d'utilisation. En effet, les principaux éléments de ce diagramme sont les classes et leur association. La figure 2.2 illustre le diagramme de classes de notre application.

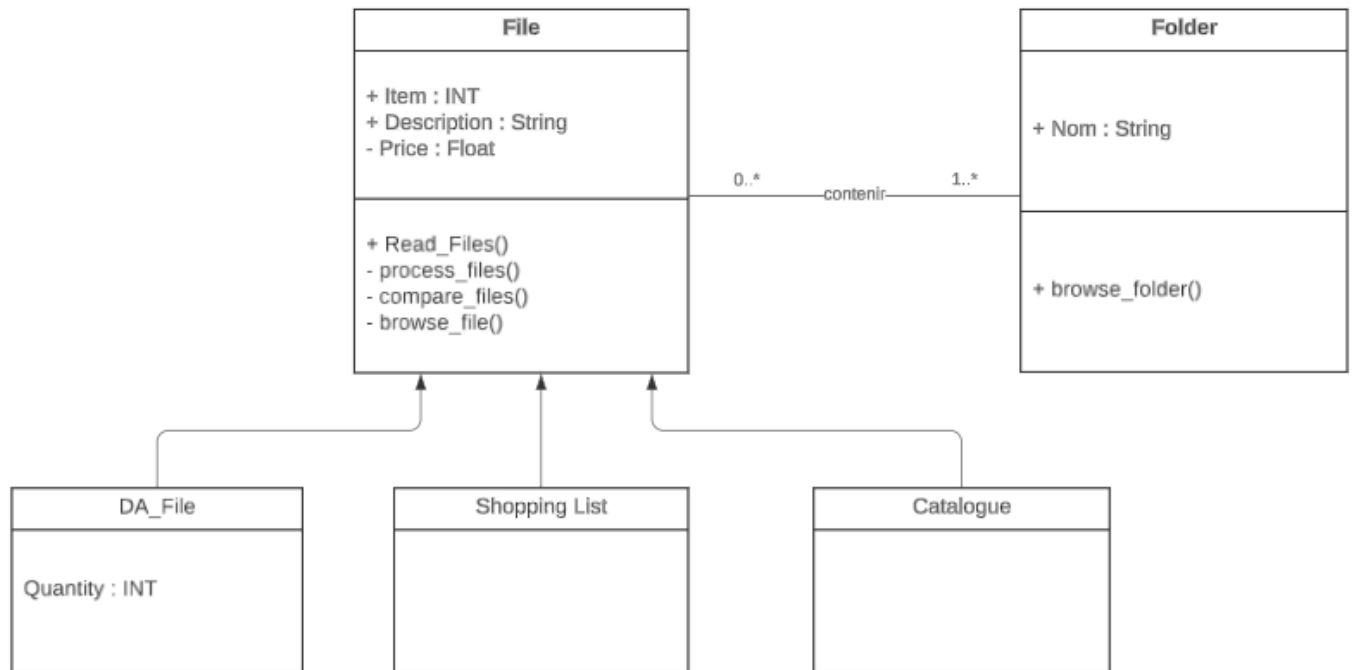


FIGURE 2.2 – Diagramme de classe

2.3 Diagramme de séquence

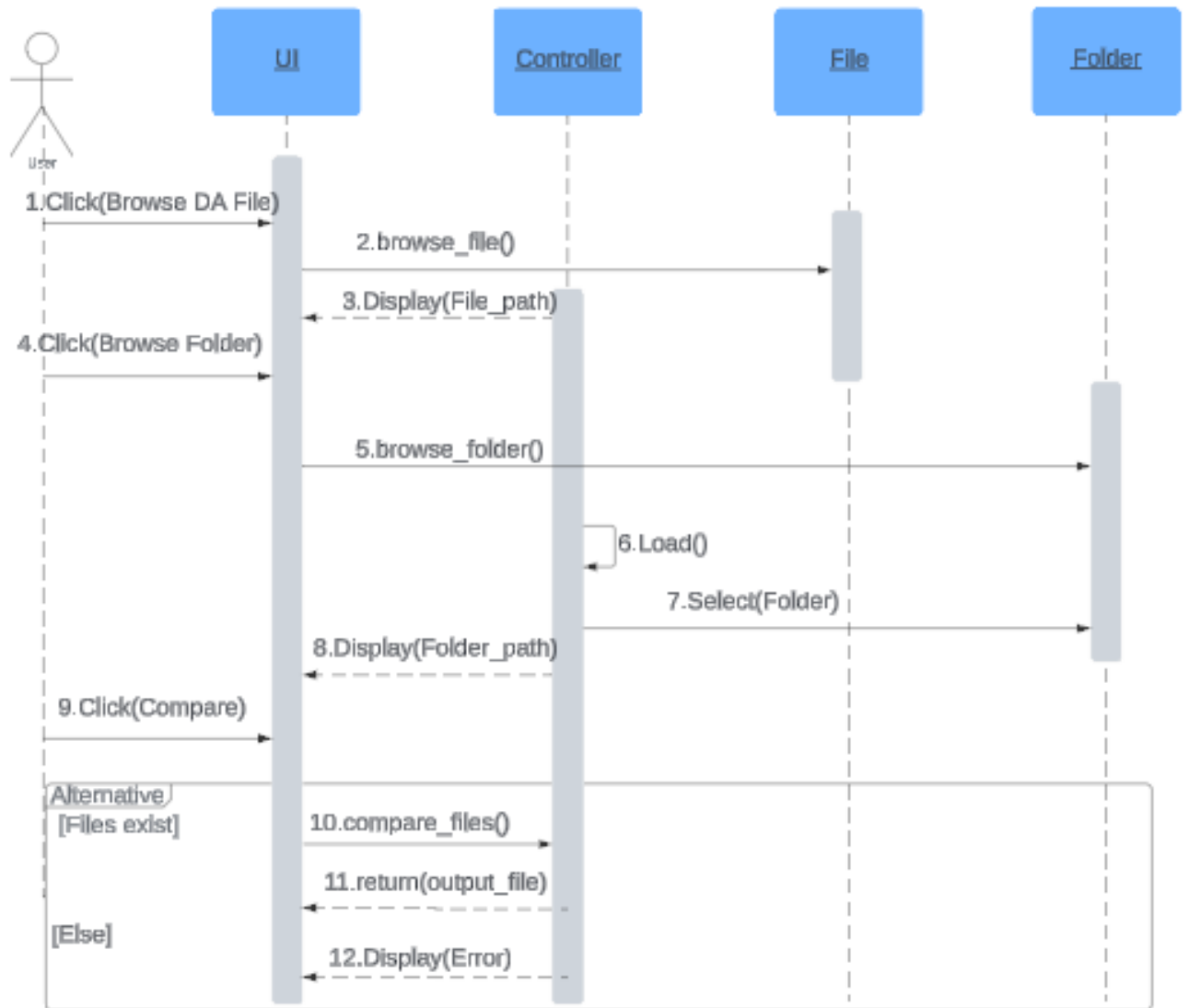


FIGURE 2.3 – Diagramme de séquence

2.4 Conclusion

Ce chapitre était utile en premier lieu, pour la récapitulation des besoins fonctionnels et non fonctionnels nécessaires pour la réalisation de notre projet et pour la conception générale. On procédera à la réalisation du projet.

Chapitre 3

Réalisation

3.1 Introduction

Cette partie contient le dernier volet de notre rapport , elle portera sur l'implémentation de notre projet. Je présenterais en premier lieu l'ensemble des environnements (matériel et logiciel). Puis j'aborderais en second lieu l'exposition des principales modules développées traduisant le déroulement de l'application

3.2 Environnement de travail

Dans cette section , je m'intéresse à mon environnement de travail. Je présente, en premier lieu, mon environnement logiciel en détaillant les différentes technologies et outils mises en oeuvre lors de la réalisation de notre solution.

3.2.1 Environnement matériel

Je vais présenter dans ce qui suit l'environnement matériel dans lequel mon application a été développée. Les caractéristiques de chaque machine sont indiquées dans la table 3.1





TABLE 3.1 – Environnement matériel

Machine	Caractéristiques
Lenovo 81IDE	Système d'exploitation : Windows 10 Professionnel 64 Bits Écran : 15.6 pouces Processeur : Intel Core I3-7020U Mémoire : 8 Go Disque dur : 1 To Carte graphique : NVIDIA GeForce GTX 1050
DELL E5530	Système d'exploitation : Windows 10 Professionnel 64 Bits Écran : 15.6 pouces Processeur : Intel Core I7-3520U Mémoire : 8 Go Disque dur : 1 To Carte graphique : Nvidia GeForce MTX110

3.2.2 Environnement Logiciel

Le tableau 3.2 récapitule tous les logiciels que nous avons utilisé pour mener à termes le projet :

TABLE 3.2 – Environnement logiciel

Logiciel	Description	Logo
Pycharm	PyCharm est un IDE dédié à la programmation en Python. Il offre une excellente prise en charge des frameworks python (Django, Flask, Pyramid...). En outre il se caractérise par la vérification et correction instantanée des erreurs, une encore la refactorisation de code.	
Python	Python est un interpréteur du langage Python qui utilise Pycharm pour compiler le code écrit en Python.	
Github	GitHub est un logiciel de versioning open source utilisé pour le partage de code source. Il facilite la création et gestion des changements dans les différents projets des logiciels.	
Staruml	StarUML est un outil riche utilisé pour la modélisation UML. Il dispose de plusieurs fonctionnalités et facile à utiliser ainsi que son interface. Il permet aux utilisateurs de créer des diagrammes.	

3.2.3 Technologies utilisées

Pandas

Pandas est une bibliothèque open-source pour le langage Python, largement utilisée pour la manipulation et l'analyse de données.

Elle offre des structures de données flexibles et expressives telles que les DataFrames, qui facilitent la manipulation de données tabulaires et étiquetées. Avec Pandas, vous pouvez effectuer des opérations de nettoyage, de transformation, de regroupement et de fusion de données de manière rapide et efficace.



FIGURE 3.1 – Logo du library Pandas

Pandas est particulièrement apprécié pour sa capacité à gérer de grandes quantités de données avec des performances optimisées, ainsi que pour son intégration facile avec d'autres bibliothèques scientifiques Python telles que NumPy, Matplotlib et SciPy.

Tkinter

Tkinter est la bibliothèque standard de Python pour la création d'interfaces graphiques. Elle est incluse avec la distribution standard de Python, ce qui en fait une option facilement accessible pour les développeurs souhaitant créer des applications de bureau. Tkinter offre une large gamme de widgets tels que des boutons, des étiquettes, des champs de texte, des cadres, et bien d'autres, permettant de concevoir des interfaces utilisateur interactives.

L'un des avantages de Tkinter est sa simplicité et sa rapidité de mise en œuvre, ce qui en fait un choix idéal pour les petits projets ou pour les développeurs débutants. De plus, il permet la création d'interfaces multiplateformes, fonctionnant aussi bien sur Windows, macOS que Linux.

Tkinter est utilisé par des développeurs du monde entier pour des applications allant des outils éducatifs aux logiciels de gestion.

Openpyxl

Openpyxl est une bibliothèque Python libre et open source spécialisée dans la lecture et l'écriture de fichiers Excel (formats .xlsx et .xlsm). Elle permet de manipuler les feuilles de calcul Excel de manière programmatique, offrant une grande flexibilité pour automatiser des tâches répétitives et complexes liées aux données Excel.

Avec Openpyxl, vous pouvez créer des classeurs Excel, ajouter et modifier des feuilles de calcul, insérer des formules, définir des styles, créer des graphiques et bien plus encore. Cette bibliothèque est particulièrement utile pour les applications d'analyse de données, de reporting, et de génération de rapports automatisés.



FIGURE 3.2 – Logo du library Openpyxl

Openpyxl est largement adoptée par les développeurs travaillant avec des données Excel, dans des secteurs variés comme la finance, l'éducation, et la recherche.

3.3 Modules réalisés

Dans cette partie, je vais présenter l'implémentation de la solution proposée. Je vais aussi exposer quelques captures d'écran de l'interface développée et du code .

3.3.1 Interface de comparaison

C'est la seule interface de ma solution , elle permet de parcourir les fichiers nécessaires et de comparer et recevoir le fichier de validation.

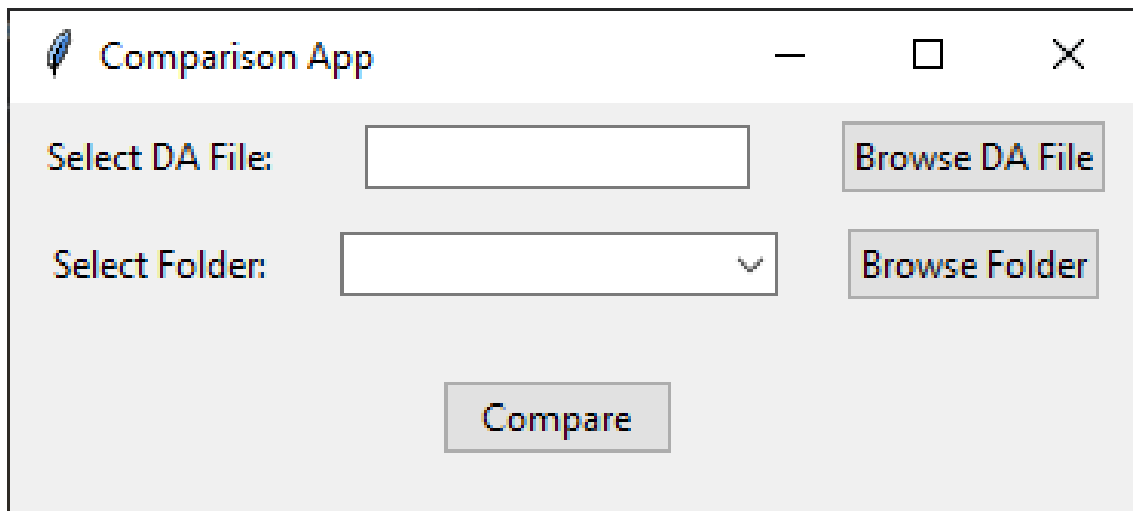


FIGURE 3.3 – L'interface de comparaison

3.3.2 Lire les fichiers

C'est la partie où on lit les fichiers.

```

1 usage  YessinToumi
def read_files(folder_path):
    dataframes = {'da': None, 'catalogue': None, 'sl': None}
    if not os.path.exists(folder_path):
        print(f"Folder '{folder_path}' does not exist")
        return dataframes
    files = [f for f in os.listdir(folder_path) if f.endswith(('.csv', '.xlsx', '.xls'))]
    if not files:
        print(f"No relevant files found in '{folder_path}'")
        return dataframes
    for file in files:
        file_path = os.path.join(folder_path, file)
        if file.endswith('.xlsx') or file.endswith('.xls'):
            df = pd.read_excel(file_path)
        else:
            df = pd.read_csv(file_path)

```

FIGURE 3.4 – Lire les fichiers

3.3.3 Reconnaître les fichiers

La partie où le système reconnaît les fichiers selon les 3 types qu'on avait :

- DA (Demande d'Achat)
- SHOPPING LIST
- CATALOGUE

```

    if 'da' in file.lower():
        dataframes['da'] = df
        print(f"Identified as DA file: {file}")
    elif 'catalogue' in file.lower():
        dataframes['catalogue'] = df
        print(f"Identified as Catalogue file: {file}")
    elif 'sl' in file.lower():
        dataframes['sl'] = df
        print(f"Identified as Shopping List file: {file}")

if dataframes['da'] is None:
    print("DA file needs to be present in the folder")
if dataframes['catalogue'] is None:
    print("Catalogue file needs to be present in the folder")
if dataframes['sl'] is None:
    print("SL file needs to be present in the folder")
return dataframes

```

FIGURE 3.5 – Reconnaître les fichiers

3.3.4 Normalisation des données

C'est la partie de la normalisation des données avant de traiter les fichiers, la normalisation dans ce contexte est transformer les données en format standard et consistant.

```
supplier_name = folder_name.split('_')[0]
# Prepare the DataFrames
da_data = dataframes['da'][['Item', 'Description', 'Price', 'Quantity']]
catalogue_data = dataframes['catalogue'][['Item', 'Description', 'Price']]
sl_data = dataframes['sl'][['Item', 'Description', 'Price']]
# Replace the ? in the Description column
da_data['Description'] = da_data['Description'].str.replace('?', '')
catalogue_data['Description'] = catalogue_data['Description'].str.replace(
# Rename columns for consistency
da_data.rename(columns={'Description': 'Description_DA', 'Price': 'Price_D
```

FIGURE 3.6 – Normalisation des données

3.3.5 La comparaison

C'est la partie de la comparaison des fichiers. La comparaison se base sur comparer chaque ligne du demande d'achat, qui représente un article, avec le même article dans les contrats par la description d'article et le prix.

```
def process_files(da_file_path, selected_folder):
    # Perform the Comparison with catalogue
    comparison_results_catalogue = pd.merge(da_data, catalogue_data, on='Item', how='inner')

    comparison_results_catalogue['Supplier'] = supplier_name
    comparison_results_catalogue['Description_Match'] = comparison_results_catalogue.apply(
        lambda row: 'ok' if row['Description_DA'] == row['Description'] else 'nok', axis=1)

    comparison_results_catalogue['Price_Match'] = comparison_results_catalogue.apply(
        lambda row: 'ok' if row['Price_DA'] == row['Price'] else 'nok', axis=1)
    # Warning Column
    comparison_results_catalogue['Warning'] = comparison_results_catalogue.apply(
        lambda row: 'DA Price is higher than Catalogue price' if row['Price_DA'] > row['Price'] else '', axis=1)
    # Handle Missing Values
    # comparison_results_catalogue['Quantity'].fillna(0, inplace=True) (method not working in pandas 3.0)
    comparison_results_catalogue.fillna(value={'Quantity': 0}, inplace=True)
    # Perform the Comparison with sl
    comparison_results_sl = pd.merge(da_data, sl_data, left_on='Description_DA', right_on='Description', how='inner')
    comparison_results_sl.rename(columns={'Item_x': 'Item_DA', 'Item_y': 'Item_SL'}, inplace=True)
    comparison_results_sl['Description_Match'] = comparison_results_sl.apply(
        lambda row: 'ok' if row['Description_DA'] == row['Description'] else 'nok', axis=1)
    comparison_results_sl['Price_Match'] = comparison_results_sl.apply(
        lambda row: 'ok' if row['Price_DA'] == row['Price'] else 'nok', axis=1)
    # Warning Column
    comparison_results_sl['Warning'] = comparison_results_sl.apply(
        lambda row: 'DA Price is higher than SL price' if row['Price_DA'] > row['Price'] else '', axis=1)
```

FIGURE 3.7 – La comparaison

3.3.6 La validation

C'est la partie du validation du DA. La validation se fait si la comparaison retourne l'article est le meme dans tous les colonnes , sinon le fichier ne se valide pas
La validation est une sortie (OUTPUT) qui est un rapport contenant tous les articles avec le resultat de validation (Valide ou pas)

```
# Output the Results
with pd.ExcelWriter('comparison_results.xlsx') as writer:
    comparison_results_catalogue.to_excel(writer, sheet_name='DA_catalogue_Comparison')
    comparison_results_sl.to_excel(writer, sheet_name='DA_SL_Comparison', index=False)
    print(f"Comparison report saved to comparison_results.xlsx")
```

FIGURE 3.8 – La validation

3.4 Conclusion

Dans ce chapitre j'ai décrit l'environnement de travail matériels et logiciels suivi par une présentation de l'interface et du description du code.

La complexité de cette réaslisation s'appuie essentiellement sur la normalisation des fichiers et la comparaison pour faire une validation vraie et précise .

Conclusion Générale

Mon projet de fin d'année a été une opportunité pour moi de comprendre les besoins et les défis rencontrés dans le domaine de la gestion des données et des processus de validation au sein d'une entreprise. En me immergeant dans les technologies de traitement des données et de développement logiciel, j'ai pu découvrir un environnement de travail stimulant et innovant.

L'objectif de mon travail était de développer une application de bureau permettant la comparaison automatique des données des Demandes d'Achat (DA) avec les catalogues et listes de courses des fournisseurs. Cette application vise à automatiser et à optimiser le processus de validation, garantissant ainsi une meilleure conformité et une réduction des erreurs manuelles.

Dans un premier temps, j'ai étudié les processus existants pour identifier les problématiques et les besoins spécifiques qui justifient le développement de notre solution.

J'ai ensuite procédé à l'analyse et à la spécification des besoins fonctionnels et non fonctionnels de l'application, en les modélisant à l'aide de diagrammes de cas d'utilisation. À partir de ces spécifications, j'ai entamé la conception globale et détaillée de l'application en utilisant divers diagrammes UML, tels que les diagrammes de classes et de séquences. La phase de conception a été suivie de l'implémentation, où nous avons développé les interfaces principales et les fonctionnalités clés de notre solution.

Enfin, j'ai expliqué le fonctionnement et l'architecture du système, tout en présentant les résultats obtenus à travers des tests de l'application. J'ai pu atteindre les principaux objectifs que nous nous étions fixés.

À la fin de ce projet, j'estime avoir développé une solution viable et efficace pour la gestion et la validation des données de Demandes d'Achat. Cependant, des améliorations et des évolutions de la version actuelle sont envisageables afin de répondre à des besoins futurs et d'optimiser davantage le système. Parmi ces perspectives, je propose l'ajout de nouvelles fonctionnalités, l'amélioration de l'interface et l'intégration de cet solution comme application téléchargeable dans d'autres systèmes de l'entreprise.

Sur le plan professionnel, ce stage a été une occasion précieuse de mettre en pratique les connaissances acquises durant notre formation et de les enrichir par de nouvelles compétences en développement de logiciels dans le traitement des données et en gestion de données. Je remercie tous ceux qui m'ont soutenus et guidés tout au long de ce projet, et je suis convaincu que les compétences acquises seront d'une grande utilité dans mon carrière future.

Bibliographie