

A

L'idée est globalement comprise, conservez le formalisme pour les rapports futurs
(ce format est souvent demandé en entreprise)



Hardis Group dans l'équipe Reflex

Rapport de stage

1^{er} année BTS SIO

Sommaire :

1.	Présentation de l'entreprise	3
1.1.	Situation géographique	4
1.2.	Organigramme de l'entreprise.....	4
2.	Mes missions	5
3.	Interface	5
3.1.	Exemple grossit.....	5
4.	Page application	6
4.1.	Interface graphique après la connexion à la base de données :	6
4.2.	Connexion à une base de données :	6
4.3.	Page des paramètres généraux :	7
4.4.	Page contexte :	7
4.5.	Page correspondance :	7
4.6.	Page des paramètres techniques :	8
4.7.	Pages Paramètre supplémentaire :	8
4.8.	Page d'initialisation des interfaces	9
4.9.	Page d'exportation.....	9
5.	Cette semaine :	10
5.1.	Rubrique générée	10
5.1.1.	Vérification	10
5.1.2.	Teste avec des expressions régulières (Regex)	10
6.	Création d'un outil personnel.....	11
6.1.	Problématique	11
6.2.	Résultat.....	11
7.	Conclusion	12
8.	Tableau Ressenti	12

1.Présentation de l'entreprise

Hardis est une société de services et de conseils en systèmes d'information, son siège social implantée à Seyssinet-Pariset. Créée en 1984, c'est l'un des leaders français du développement et de l'intégration de logiciels. L'entreprise compte aujourd'hui plus de 1300 collaborateurs répartis sur une trentaine d'agences réparties dans le monde. En 2022, elle a réalisé un chiffre d'affaires de 135 millions d'euros.

L'entreprise intervient dans divers secteurs : distribution, industrie, services, administrations. Ses domaines d'expertise couvrent la transformation digitale, les infrastructures cloud et le développement d'applications métiers. Cette polyvalence lui permet d'accompagner ses clients dans leurs projets numériques.

Résolument tournée vers l'innovation, Hardis dispose de centres d'expertise dans le cloud, le big data, la mobilité et les objets connectés. La société consacre chaque année 20% de son chiffre d'affaires à la R&D.

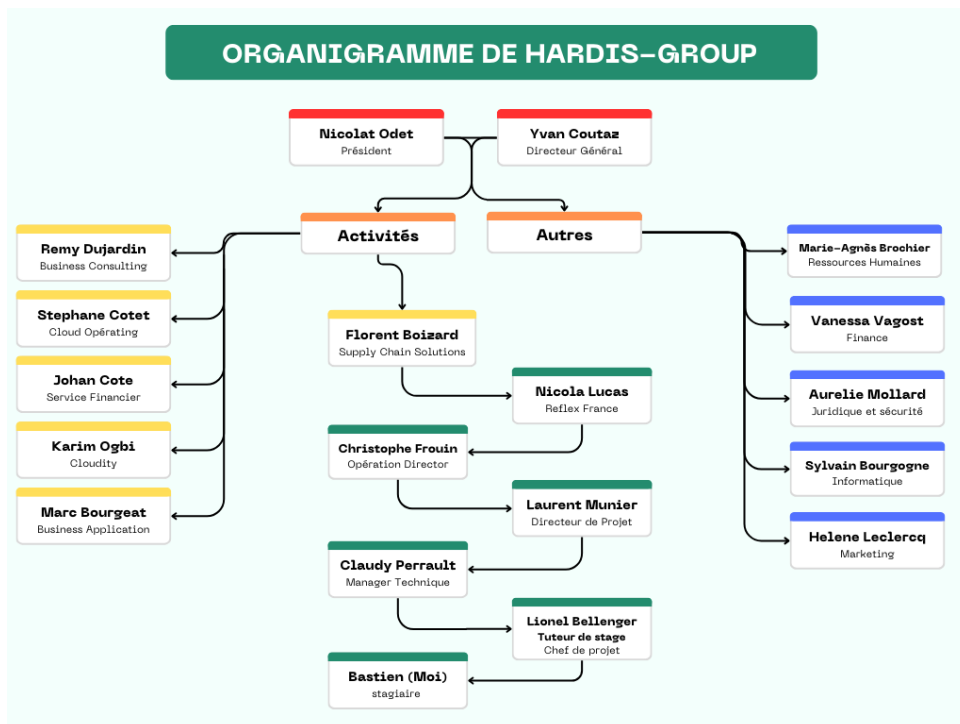
1.1. Situation géographique

Le siège social de l'entreprise est situé à Seyssinet-Pariset, ou je fais mon stage.



1.2. Organigramme de l'entreprise

Voici comment se compose de manière général l'entreprise :



2. Mes missions

Il y a un nombre d'interfaces déterminé par une base de données, et ces interfaces sont exécutables en ligne de commande, qui extrait de base de données différentes informations (des code activité et dépôts) qui sont enregistré dans un fichier txt. Le but est de rendre graphiques toutes ces opérations afin de rendre la tâche plus simple. Elle comprend la connexion à une base de données parmi trois types (MSSQL, AS400, ORACLE) qui ont chacun des paramètres de connexion différents et comme on peut le voir dans l'onglet 'Paramètres'.

3. Interface

Les interfaces sont des programmes en Python, initialement utilisés en ligne de commande, qui prennent divers paramètres lors de leur lancement (voir page "Général" ou "Correspondance"). Ces interfaces sont des EDI (Échanges de Données Informatisés) qui, à partir du contenu d'une base de données, génèrent des données de longueur fixe identifiées par un code rubrique.

3.1. Exemple grossit

INT_4D est une interface qui concerne les numéros de série de clients. Bien qu'elle ne contienne qu'une seule rubrique, le principe reste le même que les autres. Il y a une requête principale, et pour chaque rubrique, une autre requête est effectuée afin de vérifier si elle contient des données. Si c'est le cas, la rubrique est appelée pour écrire son contenu dans le fichier texte cette opération en boucle pour chaque lignes reçus par la requête principal ce qui peut prendre un peu de temps à l'opération.

```
def rubrique_110(ligne, sequence, jsonData):
    Code_activite = 
    Code_article = 
    Numero_de_serie = 
    Code_regroupement_NS = 
    # recherche s'il existe des correspondances pour le code activité
    if Code_activite in jsonData["code_act"]:
        Code_activite = jsonData["code_act"][Code_activite].ljust(3, ' ')

    rub_110 = str(sequence).zfill(
        7) + "HL4D110" + Code_activite + Code_article + Numero_de_serie + Code_regroupement_NS

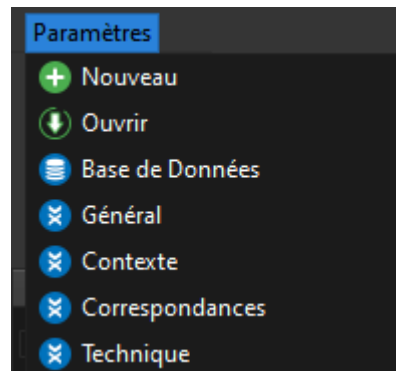
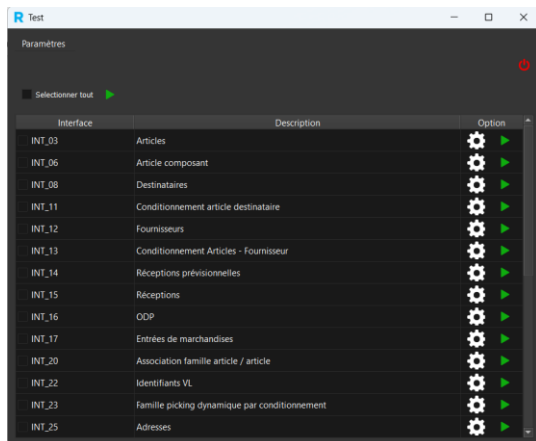
    output_file.write(rub_110 + "\n")
    sequence += 1
    return sequence
```

4. Page application

L'application contient différentes pages graphiques et voici à quoi elles servent.

4.1. Interface graphique après la connexion à la base de données :

Ici on peut voir une partie des interfaces qui vont être exécuté ^{es} ainsi que le menu ~~deroulant~~.



4.2. Connexion à une base de données :

En fonction de la base choisie :

The 'Paramètre BD' window for Oracle configuration includes the following fields:

- Type BD: Oracle
- Host: Entrez l'host
- SID: Entrez le SID/Service Name
- User: Entrez le nom d'utilisateur
- Password: Entrez le mot de passe
- Port: Entrez le port
- Schema / Database: Entrez le schéma/base de données
- Buttons: Vérifier, Enregistrer, Quitter

The 'Paramètre BD' window for AS400 configuration includes the following fields:

- Type BD: AS400
- Host: Entrez l'host
- User: Entrez le nom d'utilisateur
- Password: Entrez le mot de passe
- Port: Entrez le port
- Schema / Database: Entrez le schéma/base de données
- Buttons: Vérifier, Enregistrer, Quitter

The 'Paramètre BD' window for MSSQL configuration includes the following fields:

- Type BD: MSSQL
- Host: Entrez l'host
- Auth Method: SQL Server
- User: Entrez le nom d'utilisateur
- Password: Entrez le mot de passe
- Port: Entrez le port
- Schema / Database: Entrez le schéma/base de données
- User Bypass: Entrez l'utilisateur de contournement
- Buttons: Vérifier, Enregistrer, Quitter

4.3. Page des paramètres généraux :

Paramètre de base lors de l'envoi de la requêtes SQL pour toute les interfaces

Paramètres Généraux

Code Do * ADM

MESD * RE+

Ref. Hvt Stock REPRISE

Include Art

Cor. Cmde CRC

DLC ver DDM ☒

Art. désactivés ☐

Rempl fourm ret ☒ 0

Enregistrer Quitter

4.4. Page contexte :

Permet de visualiser et d'enregistré les valeurs sélectionnées

Contexte

Tous les dépôts ☒ Toutes les activités ☐

Liste des dépôts

	Code	Libellé
<input type="checkbox"/>	CHA	CENPAC Disponible
<input type="checkbox"/>	CVB	CENPAC CVB
<input type="checkbox"/>	FR1	RAJA PARIS NORD 2

Listes des activités

	Code	Libellé
<input type="checkbox"/>	LGS	Raja France

Enregistrer Quitter

4.5. Page correspondance :

Initialiser via du JSON situé dans une base de données

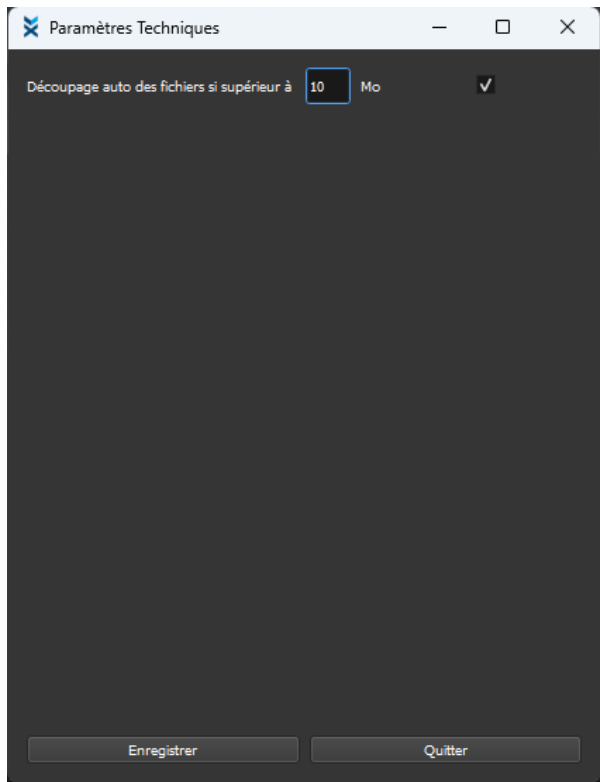
Correspondances

Key	Value
code_dpo	xxx
code_act	xxx
code_vl	xx
code_proprietaire	xxx
code_qual	xxx
code_type_supp	XXX YYY
code_taille	XXX YYY

Enregistrer Quitter

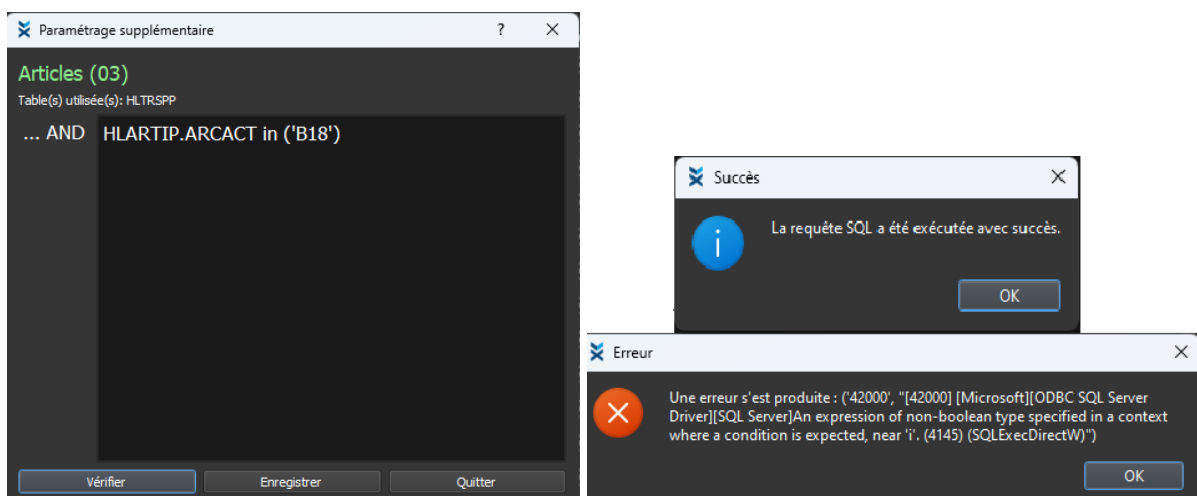
4.6. Page des paramètres techniques :

Permet le découpage du fichier généré par l'interface si la taille du fichier est supérieure à celle choisie (non obligatoire). Dans ce cas, le fichier est découpé via trois programmes Python déjà existants. Le processus de découpage comprend le tri des données des rubriques générés dans différents fichiers texte.



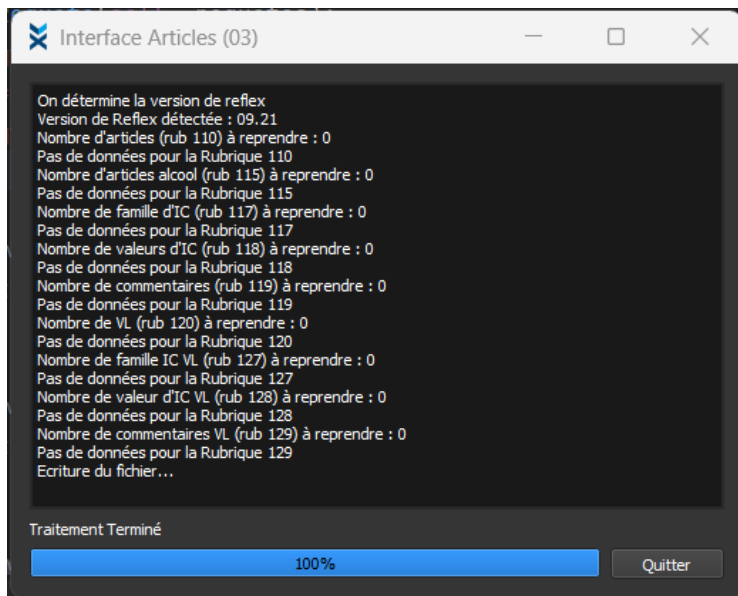
4.7. Pages Paramètre supplémentaire :

Permet d'ajouter du SQL en plus à la requête si l'on veut préciser la recherche et de pouvoir vérifier si la requête est juste ou non.



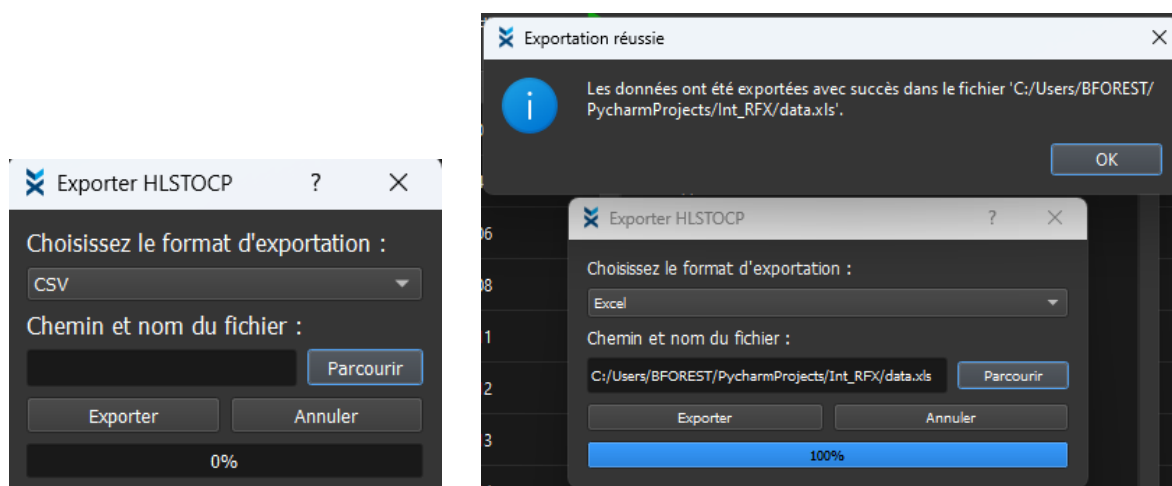
4.8. Page d'initialisation des interfaces

Cette page permet de voir le suivi de l'avancement de l'opération des interfaces. Le bouton quitter est non cliquable jusqu'à ce que l'opération soit finis comme ici. Le titre et le contenu change en fonction de l'interface choisit.



4.9. Page d'exportation

Cette page permet d'exporter les données de la table HLSTOCP à la fin de l'exécution de l'interface 17 dans fichier csv ou Excel (.xls) avec affichage en cas d'erreur ou de réussite.



5. Cette semaine :

Lors de ma cinquième et dernière semaine, j'ai essentiellement effectué des tests sur les 33 interfaces pour vérifier qu'il n'y avait pas d'erreurs, quel que soit le type de base choisie ou l'interface utilisée, et qu'elles fonctionnaient seules ou plusieurs en même temps. Cette tâche m'a pris énormément de temps. J'ai également ajouté des blocs try/except en Python pour pouvoir limiter les plantages de l'application en cas d'erreur. Enfin, j'ai réalisé une documentation pour l'utilisation de l'application en respectant la mise en page de l'entreprise.

5.1. Rubrique générée

Lorsqu'on utilise les différentes interfaces, elles génèrent un fichier texte contenant des rubriques, comme expliqué précédemment. Il faut vérifier que les requêtes fonctionnent correctement et qu'elles renvoient les bonnes données.

5.1.1. Vérification

Dans ce fichier texte, il faut vérifier que tous les champs nécessaires sont renseignés dans chaque rubrique pour pouvoir les exploiter plus tard.

5.1.2. Teste avec des expressions régulières (Regex)

Pour tester les rubriques générées par les requêtes, une expression régulière (regex) par rubrique est proposée en bleu. Il suffit de saisir la regex et la rubrique générée correspondante dans Regex101.com, dans l'encadré vert. Ensuite, en rose, on peut voir les différentes suites de caractères qui correspondent à celles dans l'encadré jaune. Cela permet de vérifier que toutes les données sont bien saisies et que tout fonctionne correctement.

The image shows a screenshot of a Python script on the left and a Regex101.com match result on the right. The Python script is for parsing a specific data format, likely a medical record, and uses a regular expression to extract fields. The Regex101.com interface shows the same regular expression being tested against a sample string, with the match result displayed in a table.

Python Script (Left):

```
# partie 110
# regex pour check :
# ^\d{7}HL17110(.{3})(.{16})(.{2})(.{3})(.{3})(.{7})(.{9})
exist_fourn = gei[9]
code_dpo = str(gei[1]).strip().ljust(_width: 3, _fillchar: ' ')
code_act = gei[2].ljust(3, ' ')
code_art = gei[3]
code_vl = gei[4].ljust(2, ' ')
code_proprietaire = gei[5].ljust(3, ' ')
code_qual = gei[6].ljust(3, ' ')
qte_vl_b = str(gei[7]).strip().zfill(7)
poids_net = float(gei[8])

if exist_fourn is None and top_rempl_four == '1':
    code_four = code_fourn_rempl.ljust(13, ' ')
else:
    code_four = gei[9].ljust(13, ' ')

ref_cond_fourn1 = gei[10].ljust(20, ' ')
ref_commande = gei[11].ljust(16, ' ')
```

Regex101.com Match Result (Right):

REGULAR EXPRESSION: `/^\d{7}HL17110(.{3})(.{16})(.{2})(.{3})(.{3})(.{7})(.{9})/gm` (1 match (176 steps, 1.1ms))

TEST STRING: `0000001HL17110FR1LGSZVP1*****40RAJSTD0*****REPRISE*****`

EXPLANATION:

Group	Match	Value
Group 1	14-17	FR1
Group 2	17-20	LGS
Group 3	20-36	ZVP1
Group 4	36-38	40
Group 5	38-41	RAJ
Group 6	41-44	STD
Group 7	44-51	000
Group 8	51-60	0000
Group 9	60-73	REPRISE

6. Création d'un outil personnel

Lorsque j'avais du temps libre, je ne voulais plus perdre de temps à supprimer les fichiers générés un par un ou à passer par l'invite de commande Windows. J'ai donc eu l'idée de créer une mini interface graphique pour une utilisation strictement personnelle.

6.1. Problématique

À chaque lancement d'une interface, un minimum d'un fichier et un maximum de sept fichiers de données sont générés, en plus du fichier de log. Lors d'un test effectué, on peut aller jusqu'à 50 fichiers générés.

6.2. Résultat

Ce petit programme permet, avec trois clics, de supprimer les fichiers dont je n'avais pas l'utilité (pour ma part).

La fonction "supprimer_files" permet de recevoir le motif des fichiers à supprimer (en rouge). Il peut s'agir des logs, des données générées ou des fichiers générés puis découpés. Grâce au motif et au paquetage "glob", le programme récupère la liste des fichiers à supprimer (en vert) et les supprime un par un (en jaune). Il affiche ensuite les noms des fichiers supprimés ou s'il y a une erreur lors de la suppression.

Code :

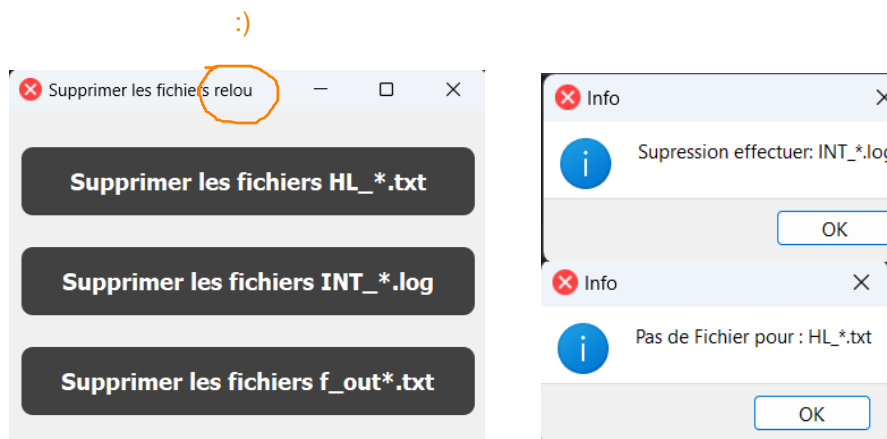
```
def supprimer_files(self, pattern):
    files = glob.glob(pattern)
    if not files:
        return
    for file in files:
        try:
            os.remove(file)
            print(f'Supprimé : {file}')
        except Exception as e:
            print(f'Erreur de suppression {file}: {e}')
    QMessageBox.information(self, 'Info', f'Suppression effectuer: {pattern}')

def supprimer_hl_files(self):
    self.supprimer_files('HL_*.txt')

def supprimer_int_files(self):
    self.supprimer_files('INT_*.log')

def supprimer_fout_files(self):
    self.supprimer_files('f_out*.txt')
```






Affichage :



7. Conclusion

Ce stage en développement informatique chez Hardis Group a été une expérience enrichissante qui m'a permis d'approfondir mes connaissances en Python, notamment sur les packages et les interfaces Qt5, ainsi que sur les bases de données. J'ai également découvert la communication dynamique entre plusieurs classes. Cette approche d'apprentissage a été super intéressante au sein d'une équipe à l'écoute et motivante. C'est une expérience très positive qui m'a permis de faire de nouvelles rencontres professionnelles intéressantes

8. Tableau Ressenti

Lundi	Mardi	Mercredi	Jeudi	Vendredi
				
Réalisation de tests	Réalisation de tests	Correction d'erreurs	Réalisation de la documentation	Présentation de l'application devant 7 Chef de projet !