National Institut of Applied Sciences and Technology

UNIVERSITY OF CARTHAGE

Graduation Project

Speciality : **GL**

Flouci Developers API

Presented By

Marouane ELKAMEL

University Supervisor : **Mr SELLAOUTI Aymen**
Enterprise Supervisor : **Mr KALLEL Anis**

Presented the : –/–/**2019**

JURY

M. President	FLEN	(President)
Mme. Rapporteur	FLENA	(Rapporteur)

Annee Universitaire : 2018/2019

Dedication

Merci e tous ! Bonne journee

Table des Matieres

Liste des Figures

Liste des Tableaux

Summary

Ceci est le resume en franeais de votre projet. Il devra etre plus detaille que le resume se trouvant dans le verso de votre rapport.

Abstract

This is the english abstract of your project. It must be longer and presented in more details than the abstract you write on the back of your report.

General Introduction

Pour ecrire un bon rapport [1] de projet en informatique, il existe certaines regles e respecter. Certes, chacun ecrit son rapport avec sa propre plume et sa propre signature, mais certaines regles restent universelles [2].

La Table de matiere est la premiere chose qu'un rapporteur va lire. Il faut qu'elle soit :

— Assez detaillee [1]. En general, 3 niveaux de numeros suffisent ;

— Votre rapport doit etre reparti en chapitres equilibres, e part l'introduction et la conclusion, naturellement plus courts que les autres ;

— Vos titres doivent etre suffisamment personnalises pour donner une idee sur votre travail. eviter le : e Conception e, mais privilegier : e Conception de l'application de gestion des ... e Meme s'ils vous paraissent longs, c'est mieux que d'avoir un sommaire impersonnel.

Une introduction doit etre redigee sous forme de paragraphes bien ficeles. Elle est normalement constituee de 4 grandes parties :

1. Le contexte de votre application : le domaine en general, par exemple le domaine du web, de BI, des logiciels de gestion ?
2. La problematique : quels sont les besoins qui, dans ce contexte le, necessitent la realisation de votre projet ?
3. La contribution : expliquer assez brievement en quoi consiste votre application, sans entrer dans les details de realisation. Ne pas oublier qu'une introduction est censee introduire le travail, pas le resumer ;
4. La composition du rapport : les differents chapitres et leur composition. Il n'est pas necessaire de numeroter ces parties, mais les mettre plutet sous forme de paragraphes successifs bien lies.

1. Sans l'etre trop

Chapitre I

Project Scope

Plan

Introduction

This chapter is dedicated to the presentation of our project's general scope. This will include an introduction of the host company Kaoun and it's main product Flouci. Also we will give an overview of the developer API project. After that, we will describe the chosen methodology that we followed during the realization of our project.

1 Host company presentation

In The first section of the report, we will introduce Kaoun, The host company that made the project possible.

1.1 Presentation of Kaoun

Kaoun is a new FinTech company that builds reliable infrastructure for payments and credits in Tunisia, and whose mission is to enable all individuals and businesses to access

financial services using any phone, anywhere, anytime.

Kaoun's first product, Flouci, is a mobile and web payment application built on top of a unique decentralized inter and intrabank infrastructure that allows instant transactions for peer to peer transfers and merchant payments. Kaoun plans to work with governments, traditional banks, mobile operators, and microfinance institutions to fix the lag between technology adoption and financial inclusion by reducing the barriers to entry for the unbanked and the underbanked.

Figure I.1 – Kaoun logo

1.2 Presentation of Flouci

Flouci is the first wallet designed to innovate mobile payment in Tunisia. It serves as a quick, easy, and convenient way to open a bank account, send and receive money, and pay different merchants in-person or online, all from within the app.

— **Open an account :**
In order to create a Flouci account, you either link your Flouci wallet to an existing bank account or follow the step-by-step KYC (Know Your Customer) guide to create an account with one of our partner financial institutions. Once you have your wallet and your QR code, you can start sending and receiving money and paying merchants using your phone.

— **Send and receive money :**
Once you create a Flouci account, you can send and receive money to and from anybody that has a flouci account/wallet. It's easy, cheap, and secure. And the best part is that it's practically instant. All you have to do is enter their number or scan their personal QR code to access their profile. You then enter the amount and confirm. You both then receive a confirmation SMS and you're done.

— **Pay merchants :**
Through Flouci, you have access to a wide range of partner merchants across Tunisia.

You can pay through the app by just scanning the QR code shown on the counter of the merchant. No waiting lines, no more looking for change or realizing you forgot your wallet at home.

Figure I.2 – Flouci logo

2 Project overview

In this section, we will present the developer API project context, we will analyze some of the existing mobile payments API's. And finally, we will indicate our project goals.

2.1 Project Context

Flouci in it's first version made it possible for users to pay merchants in simple steps and without the need of cash. However the market is rapidly shifting toward online e-commerce sites with companies like Jumia, Tayara and many other introducing their solutions in Tunisia.

In it's current implementation Flouci is not able to integrate into any form of online payments due to the lack of it's implementation.

Facing this problem and a fast moving e-commerce market the company had to move toward implementing a solution for developers.

The API should make it possible for developers to link their Flouci wallets to the their e-commerce sites and add flouci as an easy and instant payment method.

2.2 Study of the existing

In The world of e-commerce a lot of payments methods exists. Most of the methods are credit card related and offer the possibility to pay using the card number and the 3 digits code.

In our case Flouci offer payments through QR codes scans. Although the payment steps on the user side is different the developer API should offer similar functionalities.

Below is a list of world leader online payment API's :

— **Stripe :**
The Stripe API is organized around REST. The API has predictable resource-oriented URLs, accepts form-encoded request bodies, returns JSON-encoded responses, and uses standard HTTP response codes, authentication, and verbs.
You can use the Stripe API in test mode, which does not affect your live data or interact with the banking networks. The API key you use to authenticate the request determines whether the request is live mode or test mode.

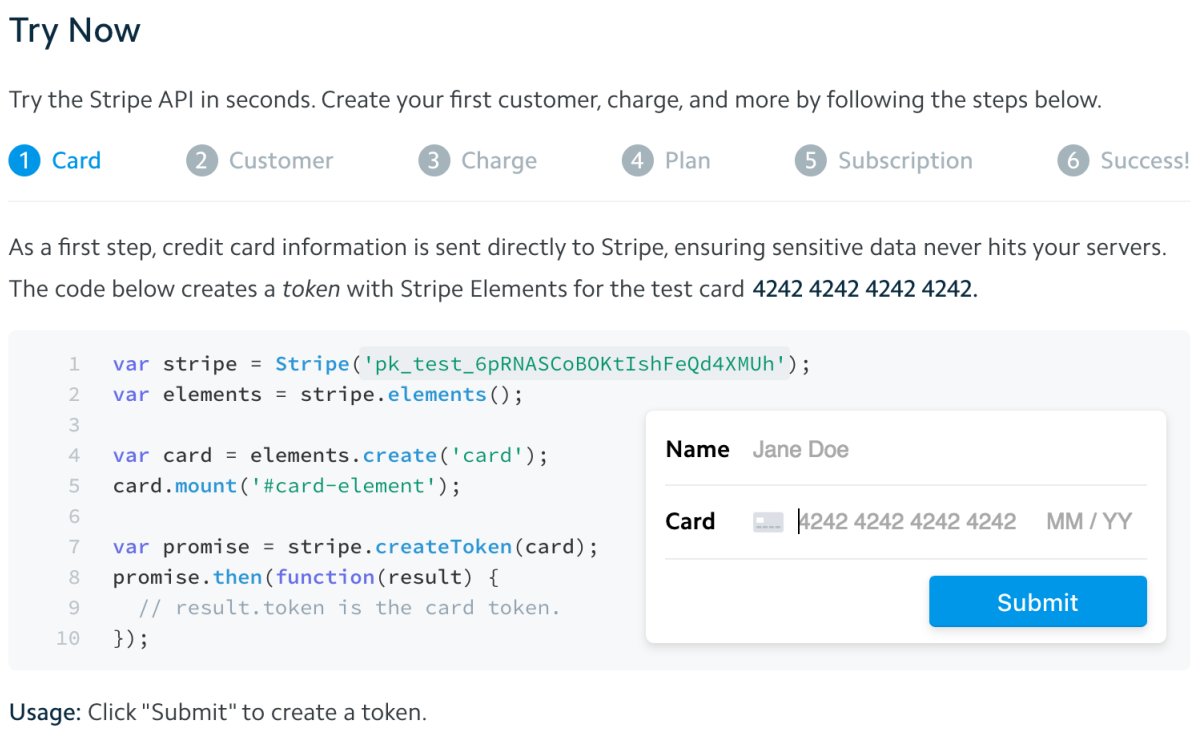

Figure I.3 – Stripe Api

— **Paypal :**
The PayPal APIs are HTTP-based RESTful APIs that use OAuth 2.0 for authorization. API request and response bodies are formatted in JSON.

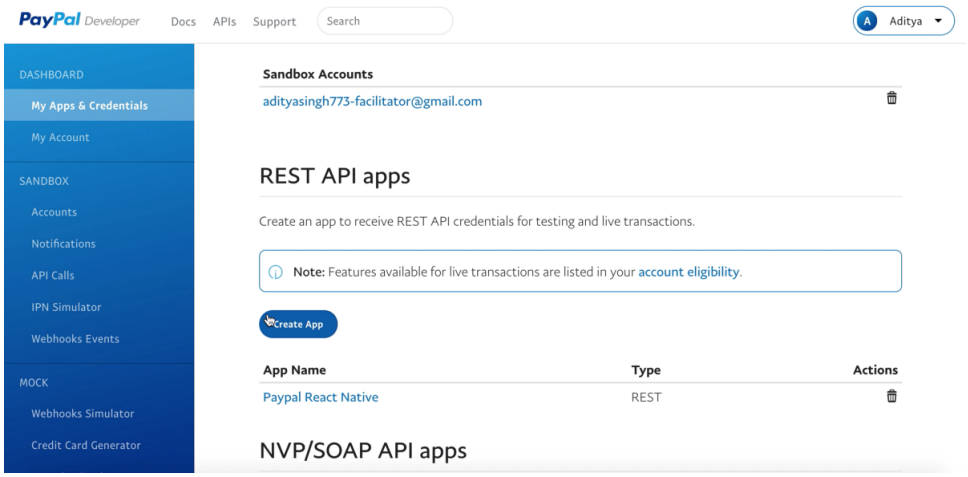

Figure I.4 – Stripe Api

— **Twint :**
Twint is the closest implementation to Flouci, it offers payments through QR code scans. The plugin allows QR code generation on the web page, it also creates a code for each transaction, It serves to confirm payments.

Figure I.5 – Twint Payment Method

2.3 Project goals

To bring developers to the Flouci world and allow e-commerce owners to introduce a mobile payment solution Kaoun has decided to build it's own developer API from scratch and provide an easy way to accept payments in few simple steps. This presented the opportunity for us to turn this problem into the main objective of our graduation project.

By the end of our project, we need to achieve these goals :

— **Create Account :**
Any developer should be able to create a Flouci developer account from the web platform, basic informations are needed to open an account.
Also it should be possible to use an existing Flouci account and switch it to developer mode.

— **Create App and link Flouci wallet :**
The web interface should offer the possibility to create an App and link an existing wallet to the specific configuration. A two steps verification system should be put in place to verify ownership of the wallet. To verify transaction developer could choose between two modes :
  — **Active mode :** Activate an endpoint to verify transactions by id.
  — **Passive mode :** Configure a webhook to receive transactions info once validated.
A unique token is generated for each app to allow client integration.

— **Integrate Flouci client :**
Once the App is configured the developer can easily add Flouci as a payment method with the token provided.

— **Check App analytics :**
Every App should enable a dashboard for the developer to monitor sales and check a set of KPI's.

3 Methodology

In this section, we will go through the importance of having a fixed methodology in a software development project, as well as our choice for this project and the reasons behind it.

3.1 Agile methodology

In every professional IT project, it is essential to adopt a methodology of work in order to guarantee a good organization of tasks and to define the different phases through which the project passes. Since their appearance, agile development methods have considerably improved the quality of the software and have reduced their production time. Thanks to its nature, incremental and collaborative, such a methodology allows to take into account the evolution of the customer's needs. Agile methodologies are based on four main values :

— They promote interaction between the various parties involved in the project by in relation to processes and tools.

— They replace exhaustive documentation with concrete functionalities.
— The client participates in the project throughout its implementation instead of defining contracts that formalize the relationship with the client.
— It is necessary to accept the likely changes during the implementation process.

We chose the Kanban methodology for our project for three reasons :

— Visually see work in progress.
— Empower teams to self-manage visual processes and workflows.
— Kanban boards can easily be managed on different free tools, like Trello, MeisterTask or GitKraken which we will be using in our project.

3.2 Kanban methodology

In general, Kanban is a scheduling system for lean and other JIT processes. In a Kanban process, there are physical (or virtual) "cards" called Kanban that move through the process from start to finish. The aim is to keep a constant flow of Kanban so that as inventory is required at the end of the process, just that much is created at the start.

When used for software development, Kanban uses the stages in the software development lifecycle (SDLC) to represent the different stages in the manufacturing process. The aim is to control and manage the flow of features (represented by Kanban cards) so that the number of features entering the process matches those being completed.

Kanban is an agile methodology that is not necessarily iterative. Processes like Scrum have short iterations which mimic a project lifecycle on a small scale, having a distinct beginning and end for each iteration. Kanban allows the software be developed in one large development cycle. Despite this, Kanban is an example of an agile methodology because it fulfils all twelve of the principles behind the Agile manifesto, because whilst it is not iterative, it is incremental.

The principle behind Kanban that allows it to be incremental and Agile, is limited throughput. With no iterations a Kanban project has no defined start or end points for individual work items ; each can start and end independently from one another, and work items have no pre-determined duration for that matter. Instead, each phase of the lifecycle is recognized as having a limited capacity for work at any one time. A small work item is created from the prioritized and unstarted requirements list and then begins the development process, usually with some requirements elaboration. A work item is not allowed to move on to the next phase until some capacity opens up ahead. By controlling the number of tasks active at any one time, developers still approach the overall project incrementally which gives the opportunity for Agile principles to be applied.

Kanban projects have Work In Progress (WIP) limits which are the measure of capacity

that keeps the development team focused on only a small amount of work at one time. It is only as tasks are completed that new tasks are pulled into the cycle. WIP limits should be fine-tuned based on comparisons of expected versus actual effort for tasks that complete.

Kanban does not impose any role definition as say, Scrum does and along with the absence of formal iterations, role flexibility makes Kanban attractive to those who have been using waterfall-style development models and want to change but are afraid of the initial upheaval something like Scrum can cause while being adopted by a development team.

3.3 Lean software development

Lean Software Development is a set of principles to deliver software according to the principles of lean manufacturing. In a lean environment, activities or processes that result in the expenditure of effort and/or resources towards goals that are not producing value for the customer should be eliminated. Essentially, lean is centered on preserving value with less work. Lean approaches are often called six-sigma or Just-In Time (JIT).

Conclusion

This first chapter allowed us to define the general boundaries of our project.

We gave an introduction of our host company and the motivations behind the project. We studied the project context and the existing similar projects, we took a look into the three biggest implementations.

In The end we set the basis of the project by fixing a methodology to follow in the development and realization of the project.

Chapitre II

Conception

Plan

Introduction

La partie conception de l'application ***n'est pas toujours obligatoire***. En effet, quand notre travail consiste en une etude theorique, ou une mise en place d'un systeme par exemple, il est inutile voire obsolete de faire un diagramme de classes ou de sequence.
Quand il s'agit de developpement, par contre, la partie conception s'impose.

1 Recommadations

En general, il faut suivre les regles suivantes :

— Choisir une methodologie de travail : un processus unifie, une methode agile ;

2 Diagrammes

Il faut Bien choisir les diagrammes adequats pour votre application. En general, les diagrammes obligatoires sont les diagrammes de cas d'utilisation, de classe et de sequence. Vous pouvez ajouter en plus le diagramme qui vous semble pertinent : par exemple, pour une application sur plusieurs tiers, il est interessant de montrer le diagramme de deploiement ;

— Les diagrammes doivent etre clairs, lisibles et bien expliques, sans pour autant nous submerger de details. Des explications trop longues deviennent ennuyeuses ;

— Si un diagramme est trop grand, vous pouvez le diviser, le representer sous forme de plusieurs diagrammes, ou vous abstraire de certains details. Si c'est impossible, imprimez

le sur une grande page (A3), quitte e la plier ensuite. Le plus important est que tous les mots soient lisibles.

2.1 Diagramme de Sequence

Un diagramme de sequence :

— Represente un scenario possible qui se deroule dans un cas d'utilisation. Vous n'etes donc pas obliges de montrer tous les cas d'execution possibles ;

— Represente l'interaction entre les objets : donc normalement, toutes les instances definies dans un diagramme de sequences doivent correspondre e des classes qui se trouvent dans le diagramme des classes ;

— Il existe parfois des dizaines de diagrammes de sequences possibles. Choisissez certains d'entre eux e mettre dans le rapport (2 ou 3). Privilegiez les diagrammes les plus importants (et non, l'authentification n'en fait pas partie !).

2.2 Diagramme de Classes

Un diagramme de classes :

— Doit etre fidele e l'architecture logicielle choisie. Si vous utilisez le MVC, alors les trois couches doivent etre representees dans le diagramme de classes grece aux packages ;

— Les stereotypes sont fortement conseilles. Si vous developpez une application web, n'hesitez pas e utiliser les stereotypes de la figure II.1 :

Figure II.1 – Les stereotypes

— Attention e ne pas confondre classes et tables : evitez la tentation de mettre des id partout.

Conclusion

Faire ici une petite recapitulation du chapitre, ainsi qu'une introduction du chapitre suivant.

Chapitre III

Realisation

Plan

Introduction

Ce chapitre porte sur la partie pratique ainsi que la bibliographie.

1 Outils et langages utilises

L'etude technique peut se trouver dans cette partie, comme elle peut etre faite en parallele avec l'etude theorique (comme le suggere le modele 2TUP). Dans cette partie, il faut essayer de convaincre le lecteur de vos choix en termes de technologie. Un etat de l'art est souhaite ici, avec un comparatif, une synthese et un choix d'outils, meme tres brefs.

2 Presentation de l'application

Il est tout e fait normal que tout le monde attende cette partie pour coller e souhait toutes les images correspondant aux interfaces diverses de l'application si chere e votre coeur, mais abstenez vous ! Il FAUT mettre des imprime ecrans, mais bien choisis, et surtout, il faut les scenariser : Choisissez un scenario d'execution, par exemple la creation d'un nouveau client, et montrer les differentes interfaces necessaires pour le faire, en expliquant brievement le comportement de l'application. Pas trop d'images, ni trop de commentaires : concis, encore et toujours.

evitez ici de coller du code : personne n'a envie de voir le contenu de vos classes. Mais vous pouvez inserer des snippets (bouts de code) pour montrer certaines fonctionnalites [3][2], si vous en avez vraiment besoin. Si vous voulez montrer une partie de votre code, les etapes d'installation ou de configuration, vous pourrez les mettre dans l'annexe.

2.1 Exemple de tableau

Vous pouvez utiliser une description tabulaire d'une eventuelle comparaison entre les travaux existants. Ceci est un exemple de tableau : Tab III.1.

Tableau III.1 – Tableau comparatif

	Col1	**Col2**	**Col3**	**Col4**
Row1		X		
Row2	X			
Row3	X	X	X	X
Row4	X		X	X
Row5	X		X	X
Row6	X		X	X
Row7	X		X	
Row8	X	X	X	

2.2 Exemple de Code

Voici un exemple de code Java, avec coloration syntaxique III.1.

Listing III.1 – Helloworld Java

```
public class HelloWorld {
//la methode main
  public static void main(String[] args) {
      System.out.println("Hello, World");
  }

}
```

3 Remarques sur la bibliographie

Votre bibliographie doit repondre e certains criteres, sinon, on vous fera encore et toujours la remarque dessus (et parfois, meme si vous pensez avoir tout fait comme il faut, on peut vous faire la remarque quand meme : chacun a une conception tres personnelle de comment une bibliographie devrait etre).

- Une bibliographie dans un bon rapport doit contenir plus de livres et d'articles que de sites web : apres tout c'est une biblio. Privilegiez donc les ouvrages reconnus et publies pour vos definitions, au lieu de sauter directement sur le premier article wikipedia ;
- Les elements d'une bibliographie sont de preference classes par ordre alphabetique, ou par themes (et ordre alphabetique pour chaque theme) ;
- Une entree bibliographique doit etre sous la forme suivante :
  - Elle doit contenir un identifiant unique : represente soit par un numero [1] ou par le nom du premier auteur, suivi de l'annee d'edition [Kuntz, 1987] ;
  - Si c'est un livre : Les noms des auteurs, suivi du titre du livre, de l'editeur, ISBN/ISSN, et la date d'edition ;
  - Si c'est un article : Les noms des auteurs, le titre , le journal ou la conference, et la date de publication ;
  - Si c'est un site web ou un document electronique : Le titre, le lien et la date de consultation ;
  - Si c'est une these : nom et prenom, titre de la these, universite de soutenance, annee de soutenance, nombre de pages ;
  - Exemples :

    [*Bazin*, 1992] BAZIN R., REGNIER B. Les traitements antiviraux et leurs essais therapeutiques. Rev. Prat., 1992, 42, 2, p.148-153.

    [*Anderson*, 1998] ANDERSON P.JF. Checklist of criteria used for evaluation of metasites. [en ligne]. Universite du Michigan, Etats Unis. Site disponible sur : http ://www.lib.umich.edu/megasite/critlist.html.(Page consultee le 11/09/1998).
  - Dans le texte du rapport, on doit obligatoirement citer la reference en faisant appel e son identifiant, juste apres avoir utilise la citation. Si ceci n'est pas fait dans les regles, on peut etre accuse de plagiat.

Conclusion

Voile.

General Conclusion

C'est l'une des parties les plus importantes et pourtant les plus negligees du rapport. Ce qu'on ne veut pas voir ici, c'est combien ce stage vous a ete benefique, comment il vous a appris e vous integrer, e connaetre le monde du travail, etc.
Franchement, personne n'en a rien e faire, du moins dans cette partie. Pour cela, vous avez les remerciements et les dedicaces, vous pourrez vous y exprimer e souhait.
La conclusion, c'est tres simple : c'est d'abord le resume de ce que vous avez raconte dans le rapport : vous reprenez votre contribution, en y ajoutant ici les outils que vous avez utilise, votre maniere de proceder. Vous pouvez meme mettre les difficultes rencontrees. En deuxieme lieu, on y met les perspectives du travail : ce qu'on pourrait ajouter e votre application, comment on pourrait l'ameliorer.

Bibliographique

[1] Lilia Sfaxi and Souheib Yousfi. *Pour bien ecrire un rapport*. Departement Math-info (2015). 1

[2] Mr. Latex. Debuter avec Latex. www.latex.com, (2008). [En ligne ; consulte le 19-Juillet-2008]. 1, 13

[3] Souheib Yousfi and Lilia Sfaxi. *Rapport Latex*. Departement Math-info (2015). 13

Annexe : Remarques Diverses

— Un rapport doit toujours etre bien numerote ;
— De preference, ne pas utiliser plus que deux couleurs, ni un caractere fantaisiste ;
— Essayer de toujours garder votre rapport sobre et professionnel ;
— Ne jamais utiliser de je ni de on, mais toujours le nous (meme si tu as tout fait tout seul) ;
— Si on n'a pas de paragraphe 1.2, ne pas mettre de 1.1 ;
— TOUJOURS, TOUJOURS faire relire votre rapport e quelqu'un d'autre (de preference qui n'est pas du domaine) pour vous corriger les fautes d'orthographe et de franeais ;
— Toujours valoriser votre travail : votre contribution doit etre bien claire et mise en evidence ;
— Dans chaque chapitre, on doit trouver une introduction et une conclusion ;
— Ayez toujours un fil conducteur dans votre rapport. Il faut que le lecteur suive un raisonnement bien clair, et trouve la relation entre les differentes parties ;
— Il faut toujours que les abreviations soient definies au moins la premiere fois oe elles sont utilisees. Si vous en avez beaucoup, utilisez un glossaire.
— Vous avez tendance, en decrivant l'environnement materiel, e parler de votre ordinateur, sur lequel vous avez developpe : ceci est inutile. Dans cette partie, on ne cite que le materiel qui a une influence sur votre application. Que vous l'ayez developpe sur Windows Vista ou sur Ubuntu n'a aucune importance ;
— Ne jamais mettre de titres en fin de page ;
— Essayer toujours d'utiliser des termes franeais, et eviter l'anglicisme. Si certains termes sont plus connus en anglais, donner leur equivalent en franeais la premiere fois que vous les utilisez, puis utilisez le mot anglais, mais en italique ;
— eviter les phrases trop longues : clair et concis, c'est la regle generale !

Rappelez vous que votre rapport est le visage de votre travail : un mauvais rapport peut eclipser de l'excellent travail. Alors pretez-y l'attention necessaire.