Sécurisation d'API

STREAM 4 GOOD

Raphaël Bellaire
Nessim El-sabbagh
Samir Jout
Anis Bahmani

UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE

Table des matières

1 Remerciements 3
2 Introduction 4
3 Analyse 5
3.1 Analyse du terrain 5
3.1.1 Schéma descriptif de l'architecture technique actuelle du projet 6
3.1.2 Schéma descriptif du système d'authentification actuel 7
3.2 Conduite de projet 8
3.2.1 Gestion du temps 8
3.2.2 Répartition du travail 9
3.2.3 Outils de communication 9
3.2.4 Trouver les informations 10
3.2.5 Trouver le problème principal 10
3.3 Le système : Vue fonctionnelle 11
3.4 Le système : Vue fonctionnelle 12
3.5 Qualité attendue 13
3.6 Choix d'implémentation 14
4 Architecture 17
4.1 Schéma d'architecture 17
4.2 Questionnaire 19
5 Partie Bilan 21
5.1 Retour sur le travail réalisé 21
5.2 Retour réflexif 22
6 Bibliographie 24
7 Annexes 25
7.1 Lien du répertoire GitHub 25
7.2 Code d'authentification initial 25
7.3 Code d'envoi du token signé 25
7.4 Choix du matériel initial 26
7.4.1 Raspberry Pi 26
7.4.2 Matériels allant avec le Raspberry Pi 26
7.4.3 Matériels liés à l'utilisation de carte à puce 28
7.4.4 Devis 29

1 Remerciements

Nous souhaitons remercier Monsieur Nicolas Herbaut qui nous a consacré du temps pour faire des réunions sur le projet, qui s'est rendu très disponible et réactif quant à nos sollicitations via Slack. Il nous a aidé tout au long de ce projet malgré les nombreuses difficultés liées au contexte sanitaire.

2 Introduction

La sécurité des systèmes d'information est un point majeur dans les entreprises de nos jours avec une augmentation constante du nombre d'attaques depuis une dizaine d'années. Nous avons décidé de participer au projet Stream 4 good avec la volonté d'emmagasiner des connaissances dans la cybersécurité.
L'objectif de notre mission consiste à permettre aux Raspberry pi du centre de recherche informatique de l'université d'accéder de manière sécurisée à des données sensibles stockées sur un serveur.

Nous allons travailler sur une authentification sécurisée d'un Raspberry pi lors de sa connexion au serveur destiné à l'extraction de données sur des plateformes de streaming comme YouTube ou Netflix. Initialement, l'ordinateur miniature récupère un login et un mot de passe qui sont générés pour se connecter. Le problème vient du fait qu'une personne mal intentionnée peut récupérer l'identifiant et le mot de passe générés et peut accéder à ce serveur, récupérer et détruire les données.
Nous allons développer une solution qui vérifie l'identité de l'ordinateur en plus de l'identifiant et du mot de passe.

3 Analyse

3.1 Analyse du terrain

Le projet Stream 4 good sur lequel nous travaillons concerne le serveur du centre de recherche de l'université. Cet outil est destiné à mener une étude quantitative sur les plateformes de streaming vidéo (Les 2 leaders Netflix et YouTube) en collectant des données pour établir des tendances sur la proposition et la recommandation de contenu en fonction de l'utilisation personnalisée. Pour la récupération d'un grand volume de données, une flotte de Raspberry pi est utilisée. Ces robots font du scrapping, ce qui signifie de l'extraction de contenu de site web.

Pour situer notre périmètre d'action au sein de l'architecture du projet , il convient de décrire l'existant en s'appuyant sur l'audit de l'existant effectué par les M1 app.
Ainsi, il y a :

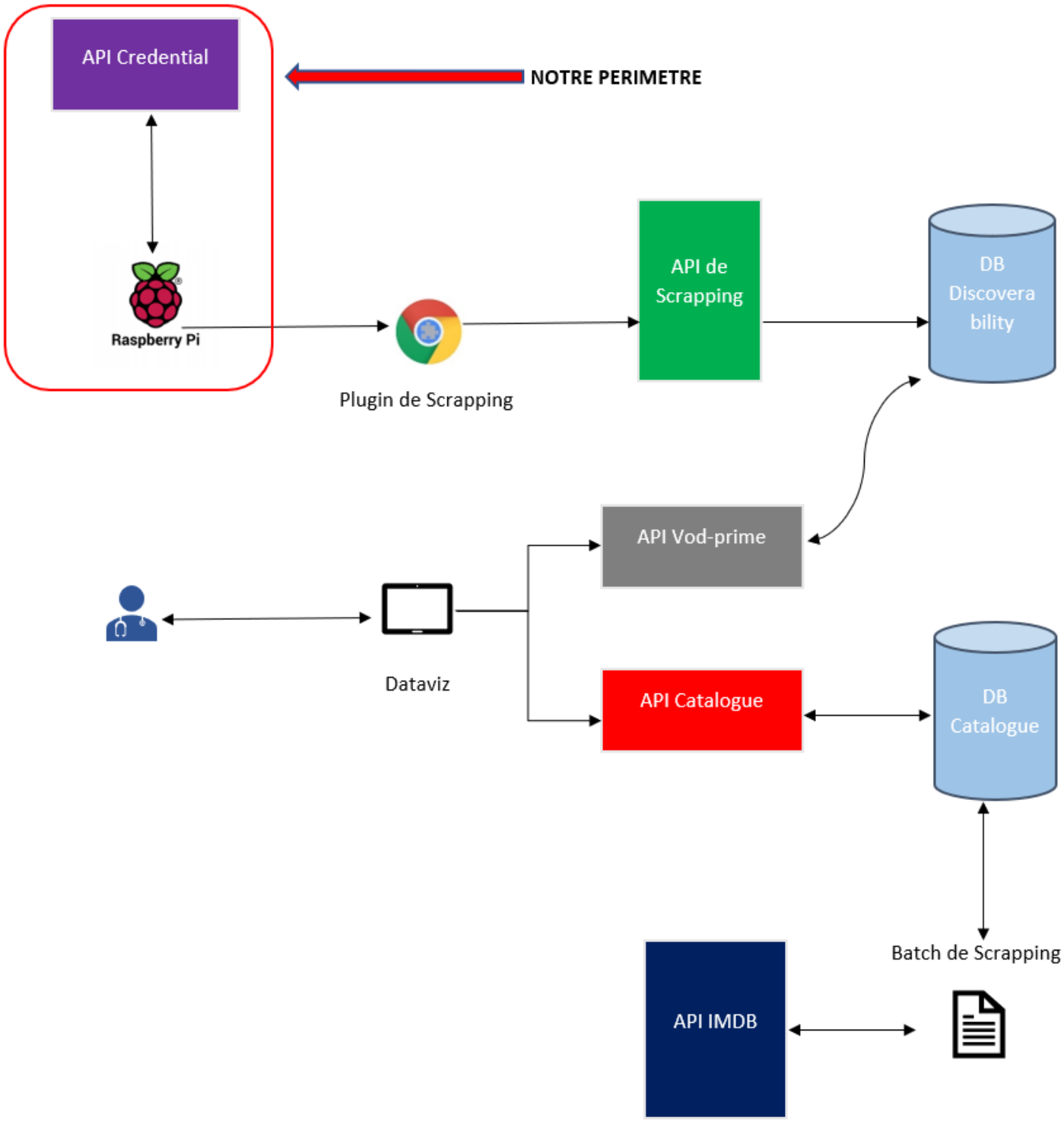
- Une API de scrapping (extraction de contenu) des pages clients
- Une API de restitution des données de scrapping (VOD prime)
- Une API de récupération des données du catalogue Netflix
- Un batch d'alimentation de la base
- Une base de données contenant les données clients
- Une base de données contenant les données du catalogue Netflix

L'accès à ces micro-services se fait par plusieurs interfaces :

- Une extension chrome envoyant les données au système via l'API de scrapping
- Un site qui permet de monitorer le système et d'accéder à de la datavisualisation utilisant l'API VOD-prime combinée à l'API de catalogue Netflix.

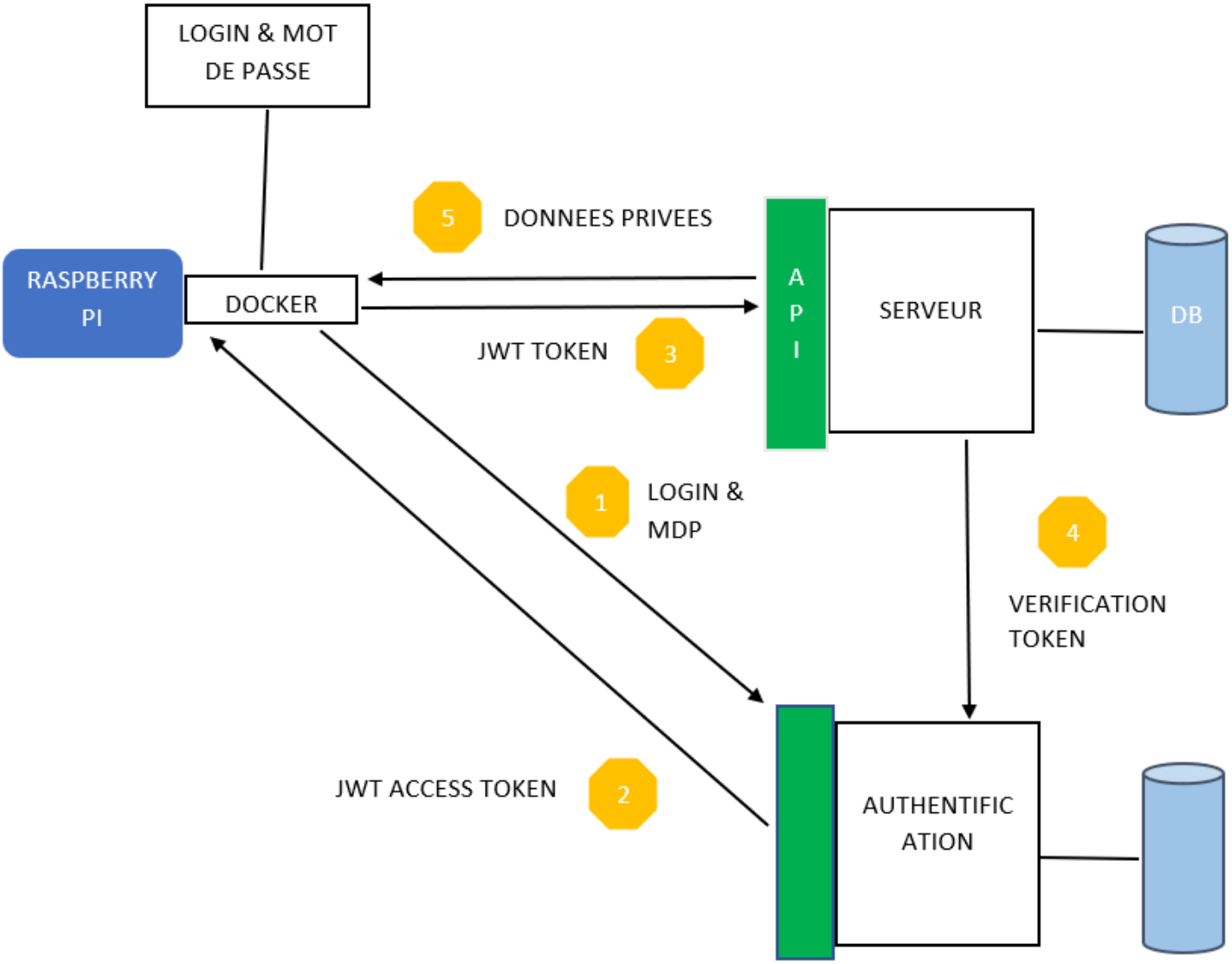
Notre objectif est d'effectuer la sécurisation des robots qui utilisent l'extension Chrome pour effectuer le scrapping. Pour être plus précis, ces bots récupèrent via une API spécifique (API credential) des informations de connexion Netflix pour effectuer de l'extraction de contenu. Nous souhaitons sécuriser l'accès à cette API.

3.1.1 Schéma descriptif de l'architecture technique actuelle du projet



Concernant le fonctionnement initial que nous avons relevé, le Raspberry Pi récupère un token de connexion qui est généré directement sur le serveur. Le problème vient du fait que ce token est systématiquement envoyé au Raspberry. Ainsi, une personne mal intentionnée peut récupérer le token, l'identifiant et le mot de passe générés et peut donc accéder à ce serveur, récupérer et/ou détruire les données (Schéma ci-dessous)

3.1.2 Schéma descriptif du système d'authentification actuel



Notre commanditaire est Nicolas Herbaut, enseignant-chercheur responsable de ce projet de serveur. Il souhaite que nous proposions un système d'authentification plus sécurisé. Plusieurs groupes d'étudiants allant de la L3 ou M2 travaillent sur ce projet. Il est donc primordial de se prémunir de possibles utilisations malveillantes des données du serveur. Nous nous sommes efforcés de proposer une solution sécurisée qui soit adaptée au niveau de la mise en place et du déploiement sur un grand nombre de Raspberry pi. Pour tester notre création, nous allons effectuer des tests en local avec du matériel similaire à celui du centre de recherche.

3.2 Conduite de projet

3.2.1 Gestion du temps

Afin de nous assurer de l'avancement de notre projet et du respect des dates butoirs, l'organisation et la planification du projet a été une de nos premières préoccupations. Nous avons démarré la réalisation de notre projet en janvier 2021 pour une restitution de nos travaux prévue en juin 2021, un temps imparti qui peut paraître long mais à mettre dans le contexte de notre rythme d'apprentis. De plus, en raison du passage en distanciel il a fallu nous adapter aux disponibilités de tous et assuré la gestion du projet sans pouvoir nous rencontrer physiquement. Notre projet étant axé sur la recherche, nous avons décidé d'adopter des principes agiles dans le but d'avancer sur notre projet en étant le moins possible dépendants des différentes parties qui forment qui le forment. Ainsi, nous avons planifié des réunions hebdomadaires qui nous ont permis d'assurer un suivi des avancées de chacun. Les différentes tâches ont été préalablement définies, découpées et objectivées selon leur temps de réalisation et leur priorité. Enfin, un rétroplanning nous a permis d'avoir une vue d'ensemble de la gestion du temps et des objectifs à court et long termes.

MAI - JUIN						
LUNDI	MARDI	MERCREDI	JEUDI	VENDREDI	SAMEDI	DIMANCHE
26	27	28	29	30	Férié 1	2
3	4	5	Réunion hebdomadaire 6	7	Férié 8	9
10	11	Réunion hebdomadaire 12	Férié 13	14	15	16
17	18	19	Réunion hebdomadaire 20	21	22	23
Férié 24	25	26	Réunion hebdomadaire 27	28	Réunion Mise au point 29	Mise en page dossier 30
Relecture finale dossier 31	Rendu dossier 1	2	3	Soutenance 4	5	6

3.2.2 Répartition du travail

Pour nous assurer une répartition du travail efficace et productive, il nous a paru évident de nous écouter, partager nos différentes expériences professionnelles, nos points forts ainsi que nos points faibles. Effectivement, nous avons tous par nos parcours et expériences des facilités ou préférences qui nous permettent d'être plus efficaces dans les décisions à prendre et notre production. Une telle répartition permet d'établir des référents et interlocuteurs privilégiés lors de nos recherches et partages d'informations. Une de nos principales missions pour se répartir le travail a été d'avoir une vision globale de l'architecture de notre projet mais aussi des choix technologiques et de de présentation finale. Pour cela un travail à la fois intéressant et fastidieux en termes de recherches d'informations a été nécessaire, car il nous a vite paru évident que pour prendre les décisions qui orientent la réalisation de notre projet il fallait être bien informé.

3.2.3 Outils de communication

Choisir les bons outils de communication a été essentiel pour nous assurer une communication synchrone et asynchrone sans ambiguïté. Il a fallu choisir des outils qui permettent de communiquer rapidement, des outils qui nous permettent d'archiver et d'autres pour partager nos avancées.

slack	- Communication avec Monsieur Nicolas Herbaut - Information sur les échéances et attendus
DISCORD	- Communication entre l'équipe - Partage de ressources - Appels/Visioconférences pour les différentes réunions
	- Rédaction du dossier - Assemblage des parties pour la rédaction du dossier
GitHub	- Partage du code

3.2.4 Trouver les informations

Trouver les bonnes informations nécessite de la rigueur et être en mesure de faire du tri. Nous avons distingué deux types d'informations nécessaires à la bonne tenue de notre projet. Les informations qui nous ont orienté dans l'architecture du projet nous ont principalement été fournies par Monsieur Nicolas Herbaut qui nous a permis de faire le lien entre notre réalisation, les attentes et l'ensemble du projet stream4good. A partir de cela, nous avons pu définir les éléments clés d'architecture mais aussi avoir une idée du niveau de criticité des différents besoins. Les informations de type documentation quant à elles proviennent principalement de nos différentes recherches sur internet. Elles ont été nécessaires dans la réalisation technique du projet. Enfin, la recherche se basant toujours sur de l'existant, nous avons tenté de trouver des personnes ayant mené des projets similaires ou ayant eu des problématiques communes aux nôtres.

3.2.5 Trouver le problème principal

Plusieurs complications ont été rencontrées lors de la réalisation du projet. Les premières ont été d'avancer avec des notions qui ne nous sont pas familières, comprendre des concepts, prendre des décisions sans pouvoir effectuer des tests au préalable notamment quand il a pu s'agir de matériel. Nous faisions parfois face à des arbitrages difficiles et qui avaient un impact sur le déroulement du projet. Nous avons dû partir dans une démarche de démonstration de faisabilité en raison du temps imparti ce qui a ajouté de la complexité. Enfin la séparation et distribution des tâches a parfois été difficile en raison de leurs dépendances.

3.3 Le système : Vue fonctionnelle

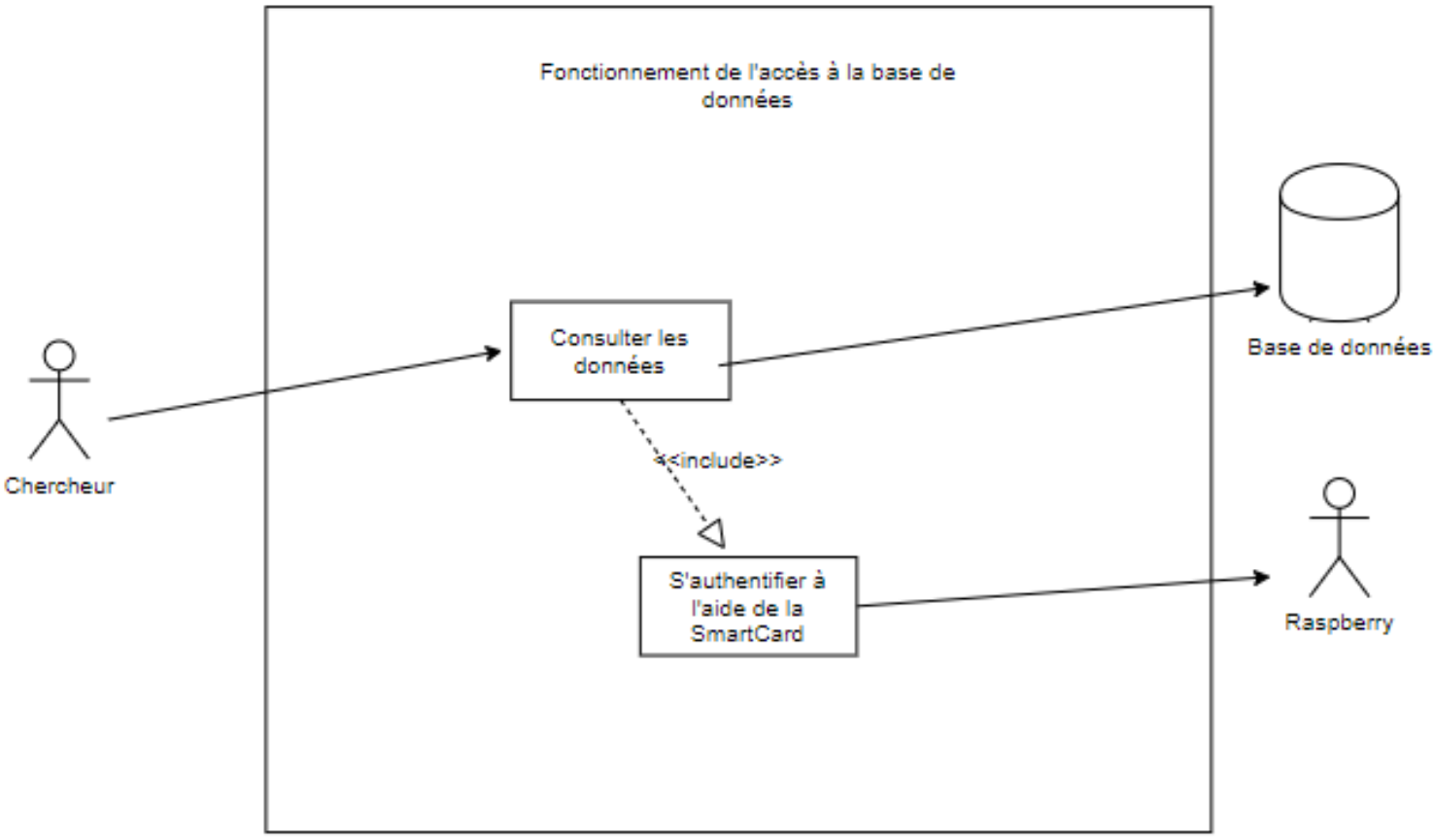


Diagramme de cas d'utilisation

Les chercheurs voulant accéder aux données doivent se connecter au serveur via un Raspberry. L'accès au serveur est sécurisé et nécessite une SmartCard.

3.4 Le système : Vue fonctionnelle

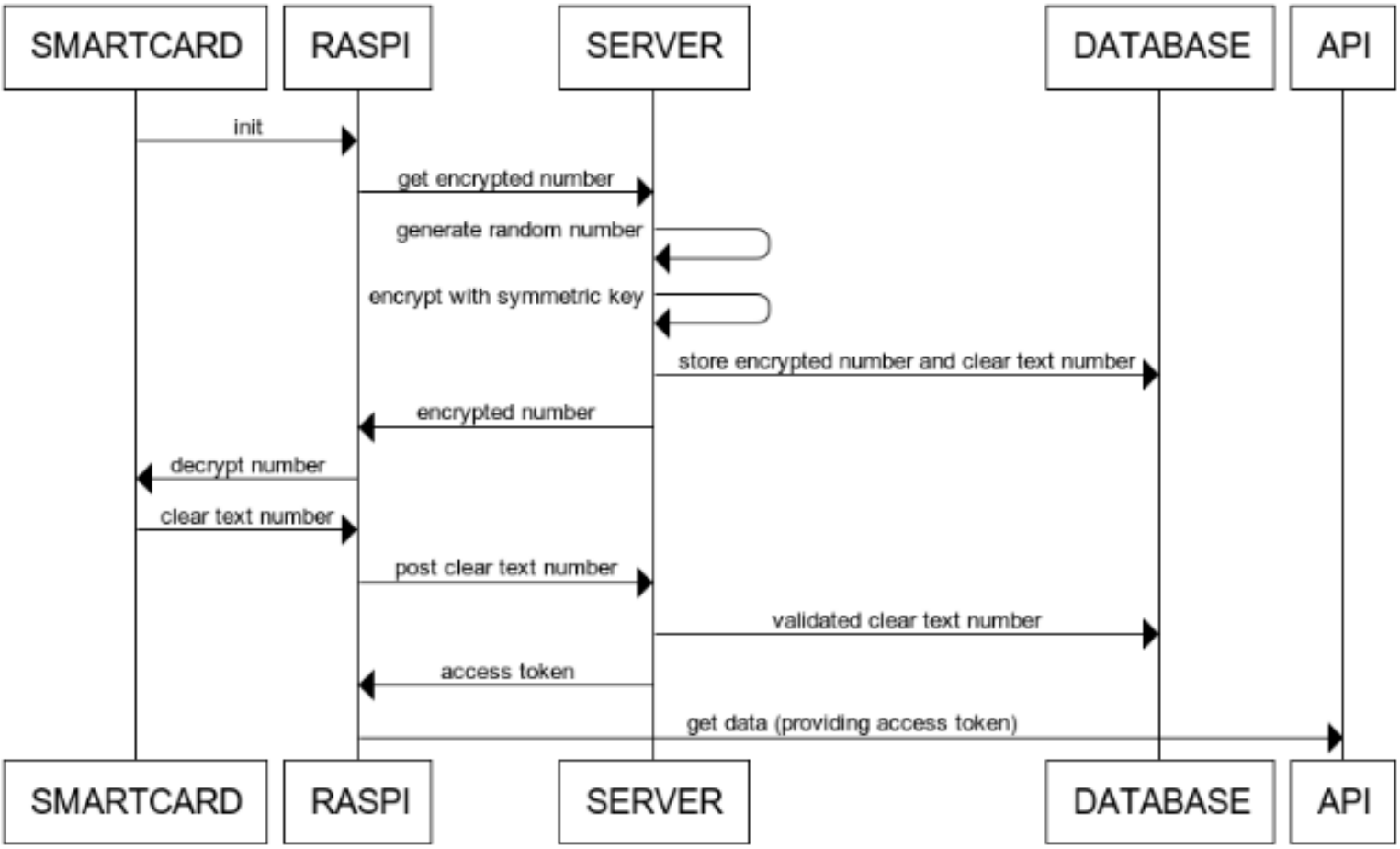


Diagramme de séquence

3.5 Qualité attendue

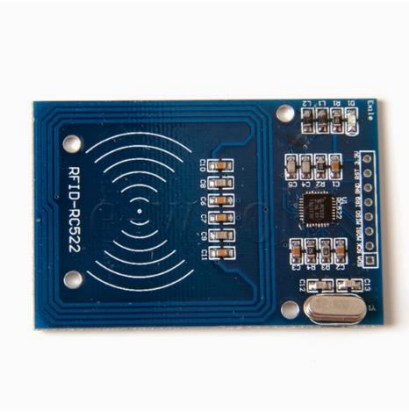
Notre objectif consiste à mettre en place une méthode de sécurisation qui soit la plus fiable possible. Notre solution doit être robuste, c'est-à-dire que nous devons effectuer une batterie de tests la plus complète possible pour identifier des failles de sécurité.
Les fonctions de sécurité intégrées dans le framework Django sont adaptées à la protection des données de notre serveur contre plusieurs types d'attaques (cross-site scripting, injection SQL). En parallèle, cet outil améliore la sécurité de notre API en évitant les erreurs de sécurité courantes liées au codage en Python.

Concernant l'évolutivité, notre solution est implémentée dans les langages Django, SQLite, Java. Ces langages sont courants et nous permettent ou même d'autres développeurs de pouvoir reprendre l'implémentation pour des évolutions futures. Pour faciliter cela, il est indispensable de commenter précisément le code informatique. De plus, nous pouvons nous appuyer pour chaque technologie sur des documentations très riches que l'on trouve facilement sur Internet. Nous nous efforçons de respecter les principes SOLID qui sont des principes de conception destinés à produire des architectures logicielles plus compréhensibles, flexibles et maintenables. Au sein du code, chaque fonction ou méthode n'a qu'une seule responsabilité (Principe de responsabilité unique). Si d'autres étudiants venaient à reprendre notre travail, ils comprendraient facilement le rôle de chaque fonction.

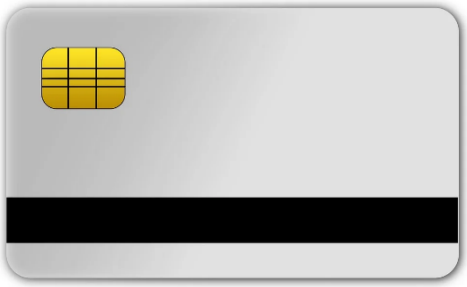
Au niveau de la maintenance, notre mission est restreinte à la phase d'authentification. Après s'être assuré du bon fonctionnement de notre développement dans un environnement local, nous l'étendons à la production. Si un bug apparait, nous sommes en mesure d'effectuer des correctifs directement sur le code. Nous conservons toutes les versions fonctionnelles du code dans un répertoire GitHub, ce qui permet de conserver le serveur disponible en permanence, même en cas de problème sur une nouvelle version. De plus, nous utilisons Maven et Pip pour faciliter la mise à jour des dépendances. Ainsi, nous réduisons les possibilités d'erreurs lors d'intégrations des développements en production. Il serait également pertinent d'utiliser le site Owasp qui permet de faire un diagnostic d'application et d'estimer le niveau de sécurité.

3.6 Choix d'implémentation

Nous avions premièrement étudié les possibles technologies d'identification sécurisée à mettre en place pour les Raspberry Pi. Dans un premier temps, nous nous sommes renseignés sur ce que l'on appelle la Radio Frequency IDentification traduisible en français par "identification par radiofréquence". Le RFID est une technologie qui permet à presque n'importe quel objet d'être identifié sans fil à l'aide de données transmises par ondes radio. Cette technologie est utilisée pour les cartes bancaires, les pass de transport, etc. C'est une technologie qui se démocratise de plus en plus avec l'avènement des cartes sans contact.



Carte RFID



Carte à puce

D'autre part , nous nous sommes aussi renseignés sur la technologie Java Card, technologie moins récente mais toujours utilisée et qui a fait ses preuves par son utilisation sur les cartes vitales, cartes de crédits. L'écriture de code pour l'exécution de comportements voulus est prévue. Cette technologie, compatible avec les standards Smart Card existants, permet de faire tourner des applications Java appelées « applets » sur des cartes à puce ayant peu de mémoire (quelques dizaines de Ko). Plusieurs applets peuvent coexister sur un même Javacard. Une des forces majeures de Java Card est la possibilité de développer des applications utilisant un langage simple comme Java, que nous connaissons.

De plus, c'est un système qui possède une sécurité accrue sécurité donnée à divers aspects de cette technologie :

- Encapsulation de données (Les données sont stockées dans l'application Java, séparé du matériel et du système d'exploitation);
- Applet Firewall (Différentes applications sont séparés mutuellement par des pare-feu empêchant l'accès aux données);
- Le chiffrement (algorithmes utilisant DES, 3DES, AES, RSA, etc.)
- Applet (Machine d'état qui traite uniquement des commandes entrantes, et répond en envoyant des données à l'interface du système).

Après des recherches sur ces deux technologies, le choix d'une smart-card basé sur java card nous semble plus approprié puisque d'après un document du gouvernement américain, la technologie RFID à différents problèmes de sécurité. Parmi ces problèmes, nous avons le skimming (accès en lecture aux informations contenus dans le badge RFID), eavesdropping (interception de la transmission de données) et le tracking (permet de suivre les déplacements du badge). De plus, nous avons vu que la technologie java card a été développée dans le but d'améliorer les performances en termes de sécurité des smart-card. Javacard offre aussi des classes qui permettent de mettre en place simplement un système de signature dont nous aurons besoin.
En effet, nous avons privilégié l'aspect sécurisé de Java Card car notre objectif est de proposer une identification plus sûre que celle qui est déjà en place.

Nous avons ensuite eu besoin de créer notre serveur local pour pouvoir tester une authentification d'un Raspberry Pi avec une carte à puce. Plusieurs solutions s'offraient à nous pour le développement back-end (développement côté serveur). Le langage le plus connu est le PHP. Il présente les avantages d'être Open Source, polyvalent, facile à utiliser et possède une sécurité intégrée. Il est souvent utilisé avec le framework Symfony. De l'autre côté Django est un des frameworks Web incontournables pour les développeurs Web. Il est basé sur le langage de programmation Python. Avec un ensemble de fonctionnalités appropriées, Django réduit la quantité de code et simplifie la création d'applications Web et se traduit par un développement plus rapide, fournit des fonctionnalités de sécurité robustes. Chacune des solutions présente ses avantages et garantit un bon niveau de sécurité. Nous avons choisi d'utiliser Django car cela nous permet de développer plus rapidement.

Ensuite, nous avions besoin d'une API pour assurer une interaction entre l'utilisateur et le serveur. Nous avons choisi un framework dans la continuité de celle utilisée pour le serveur (Django), en l'occurrence le Django Rest Framework.
Le REST va permettre de créer une API décrivant les ressources utilisées. Le grand avantage d'utiliser une API REST est que nous avons une séparation entre Client et Serveur. Ainsi de multiples Clients peuvent être développés afin de consommer les ressources mises à disposition par l'API. Le django rest framework va nous permettre de créer très facilement et en très peu de lignes de code une API REST.

Django utilise SQLite par défaut. Il s'agit du choix le plus simple car SQLite est inclus dans Python, il n'y a pas d'autres applications à installer pour utiliser ce type de base de données. De plus, notre base de données n'est pas destinée à stocker un très grand nombre de tables et de données pour le moment. Nous avons donc choisi l'option la plus rapide et facile à mettre en place et à reproduire sur d'autres postes dans un souci de faciliter une possible évolution.

Pour tester la solution que nous proposons à travers ce projet de recherche et développement, nous avons choisi de commander du matériel comprenant des cartes à puce et un lecteur adapté. Cependant, des problèmes liés à la livraison des codes des cartes nous a incités à opter pour une solution alternative.
Nous avons donc utilisé un émulateur permettant de simuler le comportement d'une carte à puce Java Card.

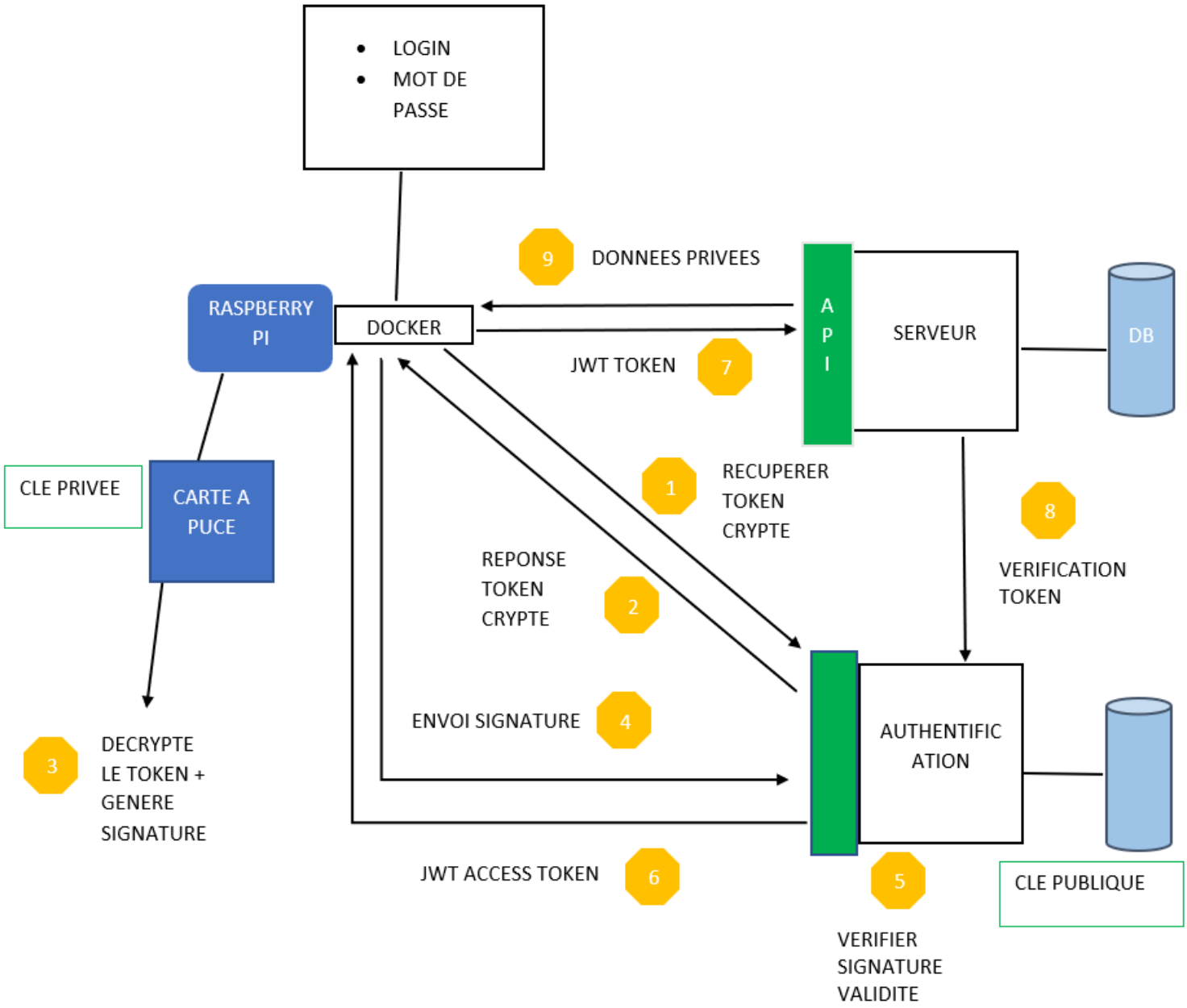
jCardSim est un simulateur permettant de développer et de déboguer rapidement des applications Java Card. De plus, jCardSim est un excellent environnement pour l'apprentissage de la programmation d'applets Java Card. Il est très utilisé par des étudiants dans le cadre de projet personnel ou scolaire.

Dans le cadre d'un projet portant sur la sécurisation d'un serveur, utiliser un système de cryptographie apparaît comme une évidence.
Le RSA est un algorithme asymétrique utilisé entre autres pour les cartes de crédit et les e-commerces.
L'un des principaux avantages de cet algorithme est qu'il ne nécessite aucun transfert de clé entre expéditeur et destinataire donc aucune personne ne peut comprendre le message crypté sans la clé publique. C'est l'algo le plus utilisé bien qu'il soit plus lent que l'algorithme AES. Il s'agit de l'autre standard en termes de d'algorithme de cryptage.
Finalement, plus qu'une question de sécurité, c'est une question d'usage qui détermine l'utilisation de l'une ou de l'autre norme.

Il est aisé de comprendre pourquoi ces deux technologies sont souvent combinées. Beaucoup de solutions sécurisées se basent sur un cryptage des données par AES, plus rapide. Mais pour obtenir la clé permettant le décryptage, l'expéditeur utilise souvent le système RSA. Après concertation de l'équipe, nous avons choisi le système de cryptographie RSA car il correspondait à nos besoins et nous avions déjà une idée de sa mise en place.

4 Architecture

4.1 Schéma d'architecture



Notre solution s'articule autour du même dispositif que celui de l'existant avec un Raspberry Pi qui correspond à un robot utilisé pour le scrapping, les serveurs d'authentification et des données de scrapping. On y ajoute aussi une carte à puce et son lecteur pour authentifier le Raspberry Pi.

Les étapes sont les suivantes :

- On effectue premièrement une requête GET pour récupérer un token crypté généré depuis le serveur d'authentification.
- Une fois reçu, le Raspberry Pi confie le décryptage du token à la carte à puce. Celle-ci renvoie une signature une fois sa tâche accomplie.
- Cette signature est envoyée au serveur d'authentification.
- Le serveur vérifie sa validité.
- Par la suite, le processus de récupération des données reste inchangé.

4.2 Questionnaire

N	Question	Réponse
1	Votre architecture vous permettrait-t-elle de supporter facilement une autre langue utilisateur (français et anglais par exemple) ?	Oui. L'interface où l'utilisateur interagit pour s'authentifier peut comporter une option langue.
2	Evaluez la difficulté du passage de votre application web à du mobile	Ne s'applique pas
3	La modularité est un critère important de qualité des architectures (Un composant ou classe n'a qu'un seul objectif). Les composants de votre architecture respectent-ils ce principe de modularité ?	Oui , nous avons essayé de respecter les principes SOLID pour notre implémentation. Chaque classe et fonction applique le principe de responsabilité unique.
4	Des éléments de votre architecture seraient-ils réutilisables dans un autre projet ?	Oui, l'architecture du montage de sécurisation avec une carte à puce peut être reprise pour une autre application avec un système d'authentification.
5	Votre architecture permettrait-elle de changer de SGBD facilement ?	Oui dans notre environnement de production puisque nous utilisons un ORM (Object Relational Mapping). Il permet de travailler directement avec des objets python dans notre cas.
6	Si vous deviez changer totalement la réalisation de l'interface graphique, y-aurait-il des composants ou classes inchangé(e)s ?	Ne s'applique pas, notre interface est simplement conçue pour une authentification.

7	Quels seraient les effets d'un éventuel changement dans le schéma de la base de données (ajout d'une nouvelle table ou nouvel attribut, changement du nom d'une table ou d'un attribut, changement sur un type d'attribut, sur une contrainte…) dans l'application ?	Les noms des tables et les champs que l'on utilise sont spécifiés dans le code Python de l'API. Si l'on modifie un nom de champ ou de table, il faudra aussi modifier le code pour garder le service fonctionnel.

5 Partie Bilan

5.1 Retour sur le travail réalisé

Ce projet était pour notre équipe une manière de se confronter à un problème de cybersécurité qui s'applique aussi bien dans le contexte scolaire que celui du monde de l'entreprise. En plus de se former sur l'aspect de la sécurité, primordial pour un système d'information, cela nous a initié au développement de méthodes de recherche face à une problématique complexe et sur un sujet dans lequel nous étions novices.
Nous avons pu proposer une solution de sécurisation de l'accès au serveur qui consiste à ajouter une authentification du Raspberry pi à l'aide d'une carte à puce en plus des informations de connexion.

Cependant, nous avons pu rencontrer des difficultés lors de la réalisation du ce travail d'équipe et la situation liée à la crise épidémique nous a obligés à travailler presque exclusivement en ligne. Il nous a été ardu dans un premier temps de comprendre l'objectif de la mission et les moyens d'y répondre. Par la suite, des contraintes liées à des livraisons de matériel nous ont empêchés de tester notre montage. Nous avons décidé d'utiliser un émulateur pour poursuivre ce travail de développement. Enfin, nous nous sommes heurtés à des problèmes de compatibilité du cryptage entre les langages Python et Java. Mais chaque membre de l'équipe a fait preuve de persévérance pour trouver des solutions ou des alternatives pour poursuivre notre progression. Nous avons des qualités différentes qui ont été mises au service du groupe, ce qui nous a permis de proposer une solution pertinente en respectant les contraintes de planning. De plus, notre responsable de projet Monsieur Herbaut s'est montré très disponible pour répondre à nos questions et éclaircir certaines zones d'ombre.

La réalisation d'un projet comporte généralement des limites et des axes d'améliorations. Notre production n'y fait pas exception et il convient d'identifier ces éléments.

Premièrement, il s'agissait d'un projet de recherche avec une forte complexité. Nous avons testé une solution sur un serveur local. Nous n'avons pas pu la tester sur le serveur effectif du centre de recherche. Par conséquent, il convient de dire que la prochaine étape du projet est le déploiement de la solution sur le serveur d'extraction des données.

Ensuite, il est important de souligner le surcoût qu'engendre cette solution via l'achat de matériel (Lecteur de cartes, cartes à puces). En effet, chaque carte sera associée à une machine. De plus, la valeur unitaire d'une carte à puce est de l'ordre de vingt euros, ce qui représente un certain budget si l'on souhaite l'appliquer à l'ensemble des Raspberry Pi.

Concernant les évolutions futures, cette mission a aussi vocation à mettre en place une méthode d'authentification sécurisée avec une partie concernant le matériel (Hardware) et une partie logiciel (Software) applicable à d'autres périmètres du projet selon les besoins futurs. En effet, il y a un besoin d'acquérir un savoir-faire réutilisable dans d'autres contextes de la même manière qu'au sein un projet de recherche et développement.

5.2 Retour réflexif

La « recherche et développement » est un regroupement d'activités qui peut s'avérer indispensable pour les entreprises afin d'améliorer ou de trouver une solution concernant un aspect d'un produit, et donc, de compléter ce dernier.

Notre projet commun consistait à sécuriser l'accès à une API chargée de l'authentification de robots par le moyen le plus efficace. Ce projet pouvait donc être associé à de la « recherche et développement ». Au cours de ce travail collectif, nous avons pu découvrir les différentes facettes d'une mission de ce type sur le long terme. Certains aspects nous ont surpris et ont parfois été la cause de problèmes dans l'avancement du projet. En effet, nous avons appris que la phase d'analyse et de conception d'une solution peut être parfois une étape complexe. C'est notamment cet aspect de ce projet qu'il l'a distingué de tous les autres projets que nous avions pu effectuer jusqu'à présent. Comment aurions-nous pu mieux aborder la phase de recherche ?

L'une des premières difficultés a été de reprendre la phase de recherche après une coupure, notamment lors de périodes en entreprise ou de semaines denses au niveau du travail scolaire. Il a fallu se replonger régulièrement dans le contexte du projet et reprendre les recherches à partir du point sur lequel nous nous étions arrêtés auparavant. Cela prend souvent un certain temps qui, finalement, n'est absolument pas négligeable une fois cumulé. Trouver une solution à cette difficulté nous semble compliqué puisque nous ne pouvons pas consacrer une longue période uniquement dédiée au projet compte tenu de notre situation d'apprenti.

Le second problème rencontré est étroitement lié au premier. Il s'agit de l'estimation de la durée des différentes étapes de la recherche. En effet, contrairement aux projets habituels dans le cadre d'une matière scolaire, il nous était complexe d'estimer la durée nécessaire pour trouver la solution que nous allions mettre en place étant donné que chaque membre de notre équipe était novice sur ce type de sujet. Nous avions découvert le contexte. Mais avant de cerner les étapes que nous allions mettre en place pour proposer une solution de sécurisation, nous avons dû nous renseigner individuellement sur le sujet et nous approprier le sujet. Nous avons été aiguillés par Monsieur Nicolas Herbaut vers certaines solutions à étudier comme les cartes RFID et les JavaCard. De plus, nous avons dû réitérer les recherches après avoir formulé un premier devis contenant le matériel nécessaire à Monsieur Nicolas Herbaut, car ces derniers ne correspondaient pas à ces attentes.

Ces deux premières difficultés, ajoutées à l'alternance entre les périodes de cours et les périodes en entreprise, nous ont posé problèmes. Cependant, pour pallier ces difficultés, nous avons travaillé sur nos méthodes de répartition du travail et d'organisation du temps.

Pour être le plus efficace dans l'avancement du projet avec les contraintes citées précédemment, chaque membre avait une tâche qui lui était confiée sur le court terme. Par exemple, Raphaël s'est chargé de définir les plannings, des recherches sur le matériel et de l'installation des Raspberry, Nessim et Samir se sont occupés de la mise en place du serveur local et de l'architecture du montage et Anis s'est chargé des

recherches sur les différentes technologies à utiliser. Lors de réunion de groupe, chacun a exposé son travail, ses questions et pris en compte l'avis des autres membres. De cette façon, il nous était possible de couvrir le plus de sujets possibles, de mettre en commun nos recherches au cours des réunions hebdomadaires et d'être le plus efficace possible.

Ce projet nous aura donc permis d'apprendre à rechercher efficacement en équipe des solutions dans le cadre d'un projet « recherche et développement », en plus d'avoir d'acquis des connaissances et de l'expérience dans le domaine de la sécurité informatique.

6 Bibliographie

Stream 4 Good
https://stream-for-good.miage.dev/

Direct Programming,"Fiabilité, performance, évolutivité",
https://directprogramming.webnode.fr/qualite/

Axopen, Maintenance applicative - TMA, Qu'est-ce que la maintenance applicative ?
https://www.axopen.com/maintenance-applicative-lyon/

Wikipédia, SOLID (informatique)
SOLID (informatique) — Wikipédia (wikipedia.org)

Oswasp, Who is the OWASP® Foundation?
OWASP Foundation | Open Source Foundation for Application Security

Developpez.com, Introduction à la programmation de Javacard sous Windows
https://julienb.developpez.com/tutoriels/java/introjavacard/

BookWiki, Java Card
https://boowiki.info/art/plate-forme-java/java-card.html

Back4app, Les 10 principaux langages de programmation backend
https://blog.back4app.com/fr/les-10-principaux-langages-de-programmation-backend/

WayToLearnX, "C'est quoi Django ?"
https://waytolearnx.com/2020/02/cest-quoi-django-avantages-et-inconvenients.html

blog.juanwolf, "Créez aisément une API rest grâce au django rest framework !"
https://blog.juanwolf.fr/fr/posts/programming/creer-api-rest-django-rest-framework/#quest-quune-api-rest-

Appvizer, Comprendre le cryptage AES pour assurer la sécurité de vos données
Cryptage AES ▶ fonctionnement, différence avec RSA et cas d'usage, logiciels cryptage AES | Appvizer

Zenk-security.com, "Cryptographie asymétrique,L' exemple de RSA"
Cryptographie asymetrique L exemple de RSA.pdf (zenk-security.com)

7 Annexes

7.1 Lien du répertoire GitHub

https://github.com/sjout/JavacardRPiAuth

7.2 Code d'authentification initial

```python


session = requests.Session()
payload = {"client_id": "dashboard-vuejs", "grant_type": "password", "scope": "dashboard-vuejs", "username": vod_user,
           "password": vod_password}
resp = session.post('https://auth.vod-prime.space/auth/realms/discoverability/protocol/openid-connect/token',
                    data=payload)
access_token = resp.json()["access_token"]
headers = {"Authorization": f"Bearer {access_token}"}
credentials = session.get("https://credentials.vod-prime.space/providers/netflix", headers=headers).json()
single_credentials_link = credentials["links"][0]["href"]
```

7.3 Code d'envoi du token signé

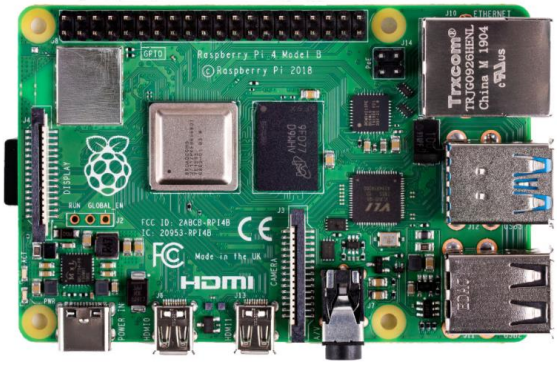
```
    * Send the signed encrypted token
     *
     * @param apdu apdu APDU that requested the signed encrypted token
     * @param sw   sw response sw code
     */
    private void sendSignedToken(APDU apdu, short sw) {
        try {
            // 1. Get encrypted token from APDU buffer
            byte[] buffer = apdu.getBuffer();
            short tokenLength = apdu.setIncomingAndReceive();

            // 2. Prepare transient memory
            byte[] encryptedToken = JCSystem.makeTransientByteArray(tokenLength, JCSystem.CLEAR_ON_RESET);
            byte[] signature = JCSystem.makeTransientByteArray(tokenLength, JCSystem.CLEAR_ON_RESET);

            // 3. Copy encrypted token from buffer to transient byte array
            Util.arrayCopyNonAtomic(buffer, ISO7816.OFFSET_CDATA, encryptedToken, (short) 0, tokenLength);

            // 4. Decrypt encrypted token
            Cipher cipher = Cipher.getInstance("RSA/ECB/OAEPWithSHA-256AndMGF1Padding");
            cipher.init(Cipher.DECRYPT_MODE, privateKey);
            byte[] clear_message = cipher.doFinal(encryptedToken);

            // 5. Sign using SHA256
            Signature signer = Signature.getInstance("SHA256withRSA");
            signer.initSign(privateKey);
            signer.update(clear_message);

            // 6. Copy encrypted signature to transient byte array
            Util.arrayCopyNonAtomic(signer.sign(), (short) 0, signature, (short) 0, tokenLength);

            // 7. Send data
            apdu.setOutgoing();
            apdu.setOutgoingLength(tokenLength);
            apdu.sendBytesLong(signature, (short) 0, tokenLength);
        } catch (NoSuchAlgorithmException | SignatureException | InvalidKeyException | NoSuchPaddingException |
BadPaddingException | IllegalBlockSizeException e) {
            throw new RuntimeException(e);
```

7.4 Choix du matériel initial

7.4.1 Raspberry Pi

Nous avons choisi de prendre un micro-ordinateur récent, donc performant, supportant aussi bien Ethernet que wifi et possédant une mémoire RAM de 2 Go.

Micro-ordinateur choisi : Raspberry Pi 4 Modèle B
Prix : 64€94 TTC
Lien du produit : https://www.ldlc-pro.com/fiche/PB00273915.html



7.4.2 Matériels allant avec le Raspberry Pi

7.4.2.1 *La carte mémoire*

Nous avons besoin de stocker des informations lors de nos différents tests d'authentification et le Raspberry ne possède pas. La carte SD Raspberry 16 gigas est facile d'utilisation (installation simplifiée de l'OS) et permet un espace de stockage suffisant.

Carte mémoire choisie : RASPBERRY CARTE MICRO-SD 16 GO AVEC NOOBS.
Référence : RB-NOOBS-32GB-PI4
Prix : 12€50
Lien du produit : https://www.ldlc-pro.com/fiche/PB00273937.html

7.4.2.2 Le câble d'alimentation

Le Raspberry ne fonctionnant que sur secteur, le câble d'alimentation est impératif. De plus, il est conseillé de prendre celui de la marque Raspberry pour ne pas endommager l'ordinateur pour des problèmes de compatibilité.

Chargeur choisi : Adaptateur secteur officiel compatible
Raspberry Pi 4B
Référence : RB-PS-PI4-WH
Prix : 12€50
Lien du produit : https://www.ldlc-pro.com/fiche/PB00273953.html

7.4.2.3 Le boîtier

On choisit d'utiliser un boîtier pour assurer la protection des pièces du Raspberry pi. Nous avons aussi vu cette possibilité d'associer ce boîtier à un écran tactile, qui rend l'utilisation du Raspberry assez pratique et permet une optimisation du montage des différentes composantes.

Référence : RB-DISPLAY-CASE-BK
Prix : 19€94
Lien du produit : https://www.ldlc-pro.com/fiche/PB00250994.html

7.4.3 Matériels liés à l'utilisation de carte à puce

7.4.3.1 Lecteur Smart Card

Nous avons choisi un lecteur qui est compatible avec une large gamme de cartes et qui puisse nous permettre d'écrire sur la carte à puce. Les informations personnelles apparaissent automatiquement à l'écran sans avoir à être saisies manuellement, ce qui permet de gagner en efficacité et qui évite une erreur de saisie. De plus, nous avons choisi un lecteur qui se branche via USB pour éviter de faire des soudures ou d'autres réglages techniques qui demandent un savoir-faire. Le lecteur de carte puce Ewent Ew1052 correspond parfaitement à nos besoins.

Référence : EW1052
Prix : 11€69
Lien : [Ewent Ew1052 Lecteur de Carte à Puce et graveur Externe USB Digital Signature, Noir: Amazon.fr: Informatique]

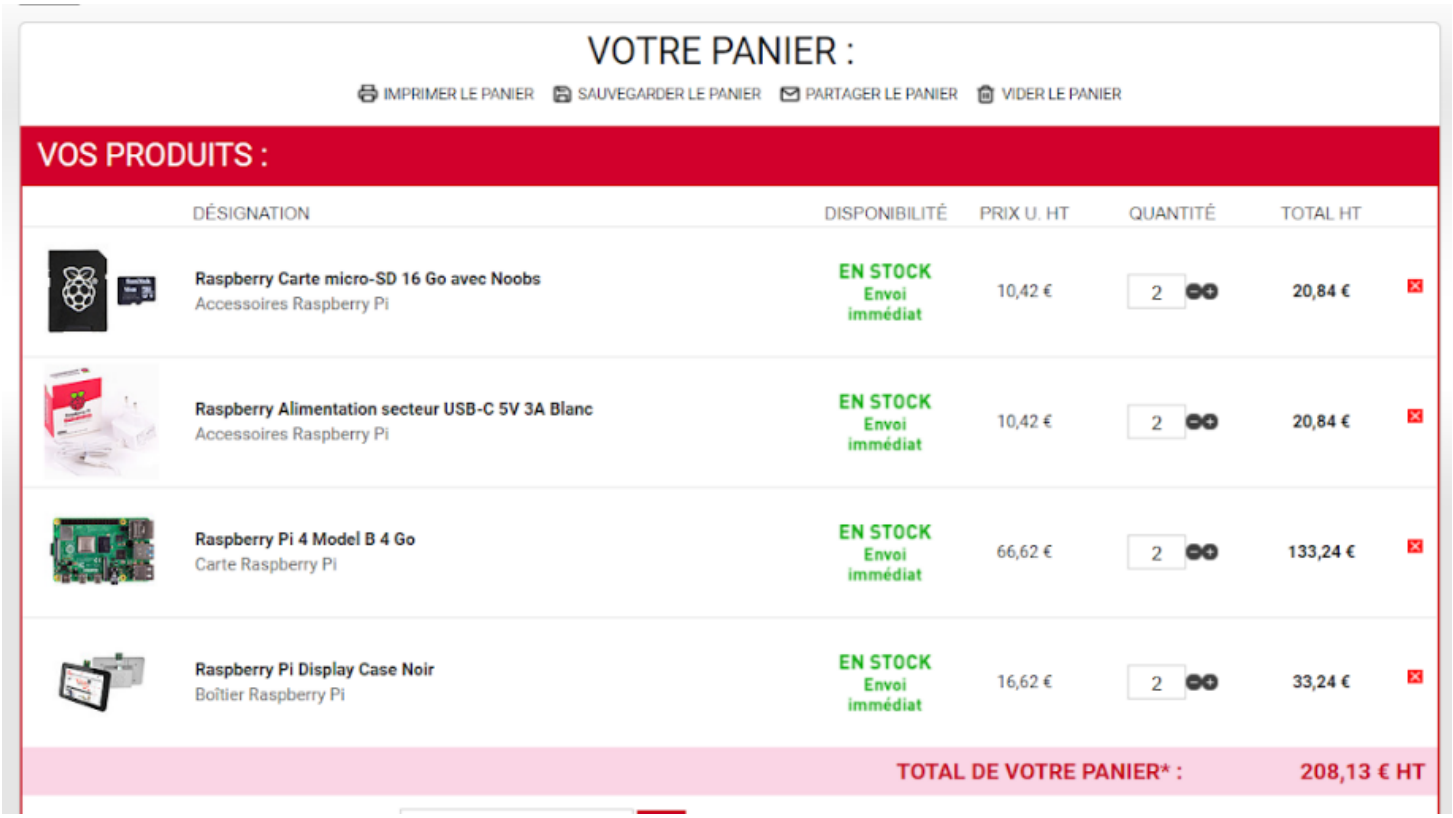
7.4.3.2 Cartes à puces

Au niveau des cartes à puce, il faut que notre carte soit compatible ISO 7816 (c'est une norme d'identification électronique avec contact). Les cartes à puce garantissent une sécurité élevée des données. Nous nous sommes orientés vers cette solution qui allie à la fois un coût modéré et une sécurité forte.

Référence : Smart IC Contact Cartes
Prix : 30€99
[YIQINGLTD FM4428 Smart IC Contact Cartes, ISO7816 PVC Blanc Carte à Puce,Compatible avec sle4428 (25pcs): Amazon.fr: High-tech]

7.4.4 Devis

- Ldlc-pro



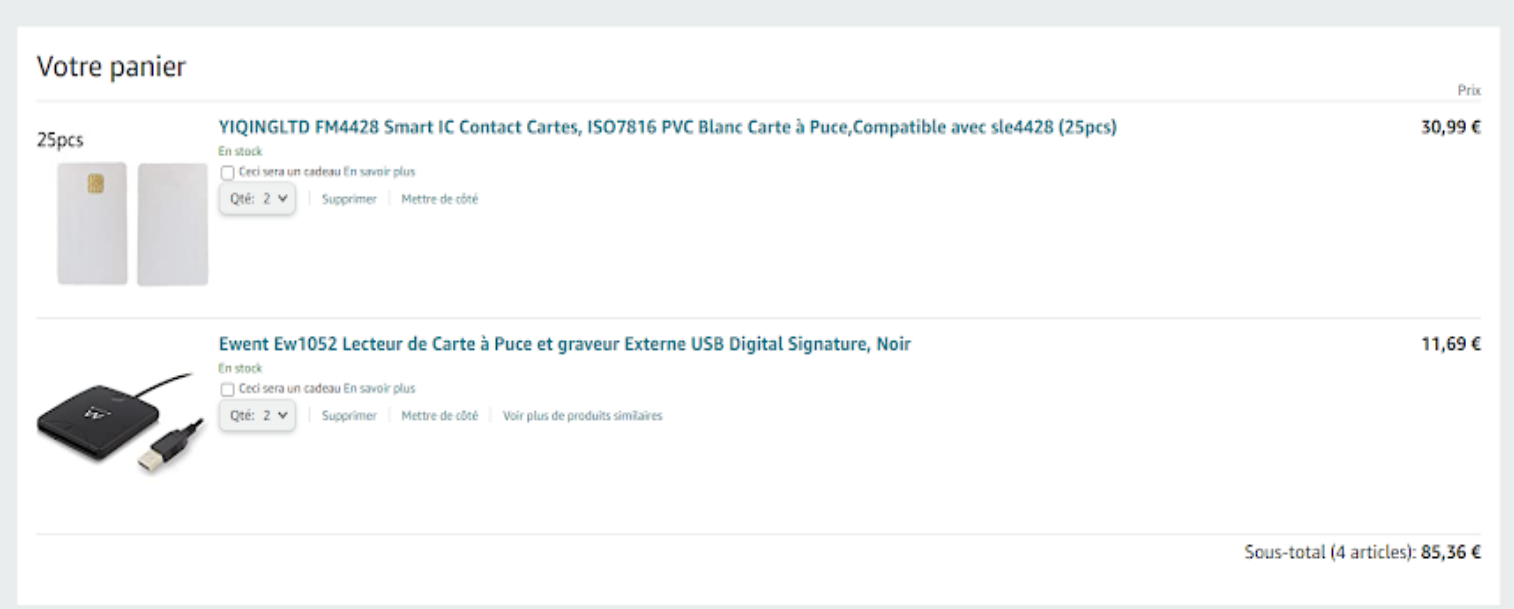
VOTRE PANIER :

IMPRIMER LE PANIER | SAUVEGARDER LE PANIER | PARTAGER LE PANIER | VIDER LE PANIER

VOS PRODUITS :

	DÉSIGNATION	DISPONIBILITÉ	PRIX U. HT	QUANTITÉ	TOTAL HT
	Raspberry Carte micro-SD 16 Go avec Noobs Accessoires Raspberry Pi	EN STOCK Envoi immédiat	10,42 €	2	20,84 €
	Raspberry Alimentation secteur USB-C 5V 3A Blanc Accessoires Raspberry Pi	EN STOCK Envoi immédiat	10,42 €	2	20,84 €
	Raspberry Pi 4 Model B 4 Go Carte Raspberry Pi	EN STOCK Envoi immédiat	66,62 €	2	133,24 €
	Raspberry Pi Display Case Noir Boîtier Raspberry Pi	EN STOCK Envoi immédiat	16,62 €	2	33,24 €
				TOTAL DE VOTRE PANIER* :	208,13 € HT

- [Amazon]


Votre panier
Prix
25pcs
YIQINGLTD FM4428 Smart IC Contact Cartes, ISO7816 PVC Blanc Carte à Puce,Compatible avec sle4428 (25pcs)
30,99 €
En stock
Ceci sera un cadeau En savoir plus
Qté: 2
Supprimer
Mettre de côté
Ewent Ew1052 Lecteur de Carte à Puce et graveur Externe USB Digital Signature, Noir
11,69 €
En stock
Ceci sera un cadeau En savoir plus
Qté: 2
Supprimer
Mettre de côté
Voir plus de produits similaires
Sous-total (4 articles): 85,36 €