GNG 1503[A]

Manuel d’utilisation et de produit pour le projet de conception

Système de Gestion d’inventaire

Soumis par:

JAYLN – Équipe FA34

Alexandre Strub, 300373984

Loïc Harvey-Boudreault, 300193523

Yacine Boubacar, 300293044

Joseph Tshimanga, 300292316

Nelly Ouedraogo, 300328096

10 décembre 2023

Université d’Ottawa

Table des matières

Liste de figures

Liste de tableaux

Liste d’acronymes et glossaire

Tableau 1. Acronymes

Acronyme	Définition
RFID	Radio Frequency Identification
CSS	Cascading Style Sheet
HTML	Hyper Text Markup Language
API	Interface de Programmation d’Application
IDE	Integrated Development Environment

Tableau 2. Glossaire

Terme	Acronyme	Définition
Entrepôt	N. A	Lieu de dépôt pour les marchandises
Backend	N. A	Partie du système informatique que l’utilisateur ne voit pas mais qui lui permet de réaliser des actions sur un site web
Breadboard	N. A	Dispositif qui permet de réaliser des montages électroniques simples sans avoir à souder les composants

1 Introduction

Ce manuel d'utilisation et de produit (MUP) fournit les informations nécessaires aux gestionnaires des entrepôts du Gouvernement du Canada pour utiliser efficacement le Système d'Inventaire Automatisé (SIA) et pour la documentation du prototype.

Ce projet a été effectué dans le contexte du cours GNG 1503[A] de l'Université d'Ottawa, au compte de notre client, Services Partagés Canada (SPC). Ce document comporte des informations générales sur notre produit comme sa configuration et son organisation, des instructions pour son utilisation, des conseils de dépannage, et sa documentation. Nous voulons permettre aux utilisateurs d'utiliser notre produit en toute sécurité et en toute confiance, et d'avoir accès à toutes les informations dont ils pourraient avoir besoin.

Attention! En accédant à ce document, vous acceptez de respecter la confidentialité de ses auteurs et, de ce fait, ne pas divulguer ses informations à un public non autorisé. Ce produit est, et reste, la propriété de ses créateurs.

Si la directive indiquée ci-dessus n'est pas respectée, nous nous réservons le droit de retirer l'autorisation d'utilisation ou de copie de tout contrevenant, et d'initialiser une procédure de bris de confidentialité en justice contre eux.

2 Aperçu

Toutes les entreprises faisant usage d'un système d'entrepôts, que ce soit Rona, Costco, Amazon, ou encore Home Dépôt, souffrent de pertes économiques et temporelles à la suite d'erreurs humaines. Ces problèmes étant très présents dans les entrepôts du Gouvernement du Canada, Services Partagés Canada nous a approchés pour que nous concevions un système d'inventaire automatisé qui permettrait d'assurer une meilleure fiabilité, et de réduire les pertes de temps et d'argent.

Les avantages de notre produit sont nombreux, car notre base de données Firebase s'assure de sa fiabilité, sa sécurité, et sa capacité de fonctionner en temps réel. Aussi, puisque notre système informatique utilise le langage de code Python, cela permet la modification et l'ajout faciles et simples de nouvelles fonctionnalités. De plus, notre interface possède toutes les qualités requises pour permettre aux utilisateurs d'inspecter, et de rechercher un objet voulu ainsi que de changer leurs paramètres à leur loisir.

Voici un aperçu général de notre système global :

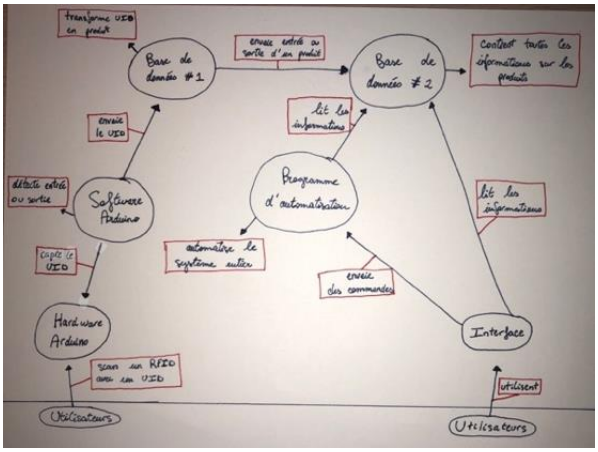


Figure 1. Aperçu général

2.1 Conventions

Aucunes conventions

2.2 Mises en garde & avertissements.

Ce produit comporte des composantes électriques théoriquement sans risque, mais dont une utilisation non-appropriée au produit peut mener à des blessures. Nous recommandons donc de bien lire les directives indiquées dans ce document, et nous excusons de toute responsabilité d'accident ci ces directives ne sont pas suivies comme indiqué.

De plus, par mesure de sécurité, nous vous prions de n'apporter aucun changement logiciel à notre produit sans l'avis professionnel de l'un des membres de notre équipe ou d'un expert en la matière. Cet aspect du produit ne comporte en soit aucun danger direct, mais peut causer des dégâts économiques considérables si mal utilisé ou configuré.

Attention! Une autorisation d'utilisation ou de copie est requise pour tout utilisateur non précédemment autorisé. Pour ce faire, il vous est prié de communiquer avec votre employeur, ou directement avec les créateurs de ce produit. Notez bien que l'obtention de cette autorisation n'est jamais garantie, et que nous nous réservons le droit de la retirer en tout temps.

3 Pour commencer

3.1 Considérations pour la configuration

Prototype physique :

Afin de configurer ce prototype, il faut avoir accès au matériel requis. Ainsi, pour un ensemble :

- 1 capteur Arduino pour RFID
- 1 breadboard Arduino
- 1 module ESP8266
- 7 fils femelle – femelle
- 1 boite RFID

Où brancher les fils :

Tableau 3. Brancher les fils

Plaque ESP8266	**Capteur Arduino pour RFID**
3V3	3.3V
D1	RST
GND	GND
D6	MISO
D7	MOSI
D5	SCK
D2	SDA

Voici à quoi ressemble le circuit branché au complet :

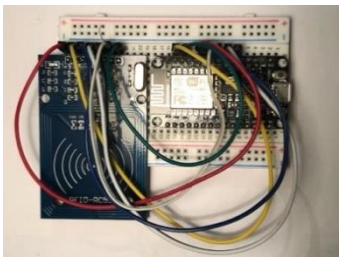


Figure 2. Circuit électrique branché

Ensuite, il suffit de l'insérer dans la boîte RFID, et de le connecter à un ordinateur :

Figure 3. Boîte RFID

Prototype logiciel :

La configuration de ce système est assez simple. Il suffit d'ouvrir l'interface HTML sur un moteur de recherche de votre ordinateur. Pour ce faire, vous pouvez aller dans le fichier où est sauvegardé ce site web, et l'ouvrir avec le moteur de votre choix :

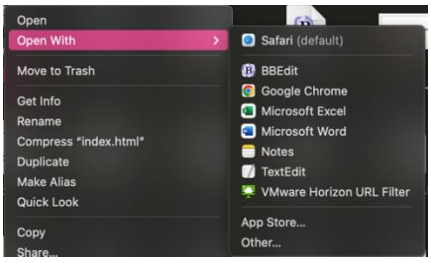


Figure 4. Ouvrir le site web

3.2 Considérations pour l'accès des utilisateurs

Seuls les employés de services partagés Canada peuvent accéder au site web et son contenu à l'aide de leurs numéros d'employés. Seuls les directeurs ont la capacité d'autoriser, de limiter ou d'annuler l'accès de tout utilisateur subordonné.

N.B : Les directeurs seuls recevront des notifications en cas stockage insuffisant.

3.3 Accéder/installation du système

Pour configurer la partie physique du trieur d'inventaire, vous devez vous connecter en USB à l'ESP et à votre ordinateur, l'ESP et le capteur RFID doivent déjà être connectés à une maquette ou à un circuit. Sinon, contactez le support. Une fois l'ESP connecté à l'ordinateur, téléchargez l'IDE Arduino depuis le site Arduino et installez les bibliothèques suivantes :

ArduinoFirebase.h

ESP8266WiFi.h

SPI.h

Rendez-vous ensuite sur GitHub, où se trouvent tout le code nécessaire ainsi que les instructions. Téléchargez ces bibliothèques suivantes sur votre système ou système virtuel en fonction de votre utilisation:

- Flask
- Pyrebase
- Pip
- Render_Template
- Request

Et exécutez cette commande dans la ligne de commande:

–flask app bdd run ou exécuter le fichier python

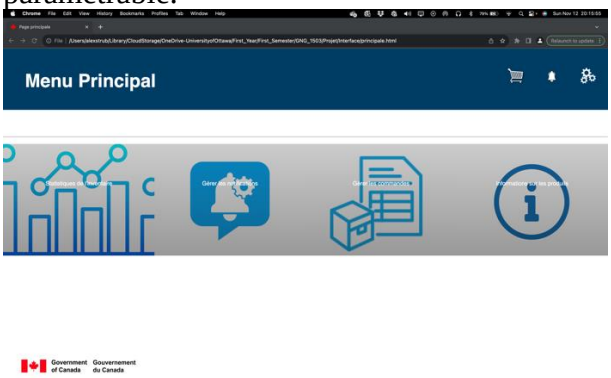
3.4 Organisation du système & navigation

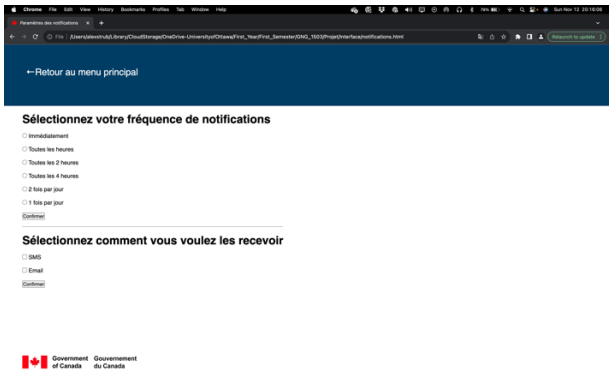
PROTOTYPE 1

Le prototype physique est constitué principalement d'un Arduino et d'un capteur RFID. Nous avons également utilisé des fils afin de connecter les composantes principales ainsi que coder dans l'Arduino IDE pour les lier ensemble et scanner les cartes.

PROTOTYPE 2

Le deuxième prototype est un prototype logiciel qui englobe toutes les pages de l'interface HTML
Les utilisateurs ont accès à la page principal du site web s'ils remplissent premièrement la page leur demandant le nom d'utilisateur et le mot de passe leur étant assigner par l'employeur. De la page principale, plusieurs options différentes s'offrent à eux, comme la page de notifications paramétrable.

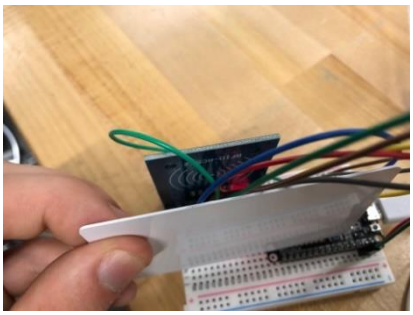


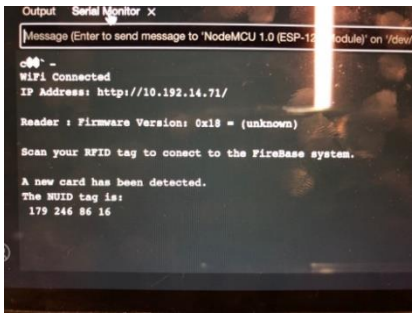


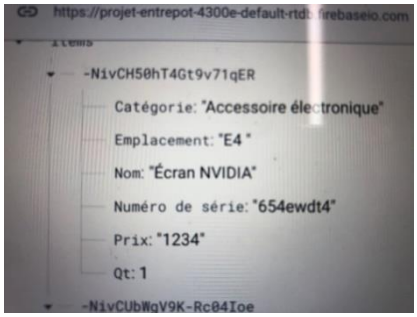
PROTOTYPE 3

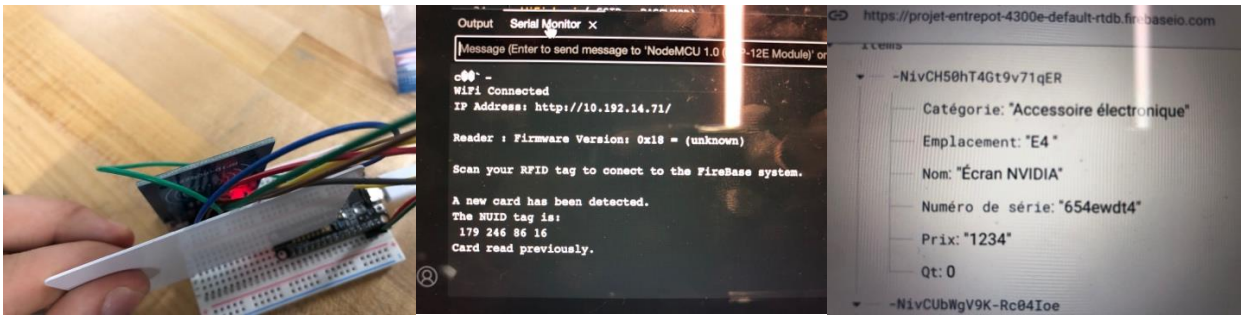
Le prototype 3 est un prototype surtout logiciel puisque c'est la liaison entre le système Arduino et la base de données, mais le système Arduino est déjà complété. Ce prototype consiste à changer les quantités des produits dans la base de données.

Lorsqu'on scanne une carte rfid, la quantité augmente de une unité. Lorsque cette même carte est rescannée, la quantité baisse de une unité.









PROTOTYPE 4

Le prototype 4 est simplement la liaison entre tout les prototypes.

3.5 Quitter le système

Prototype physique :

Il suffit de débrancher les câbles de l'ESP-8266 et de l'ordinateur auquel il est connecté, de plier soigneusement les câbles et de les ranger dans un endroit facilement accessible.

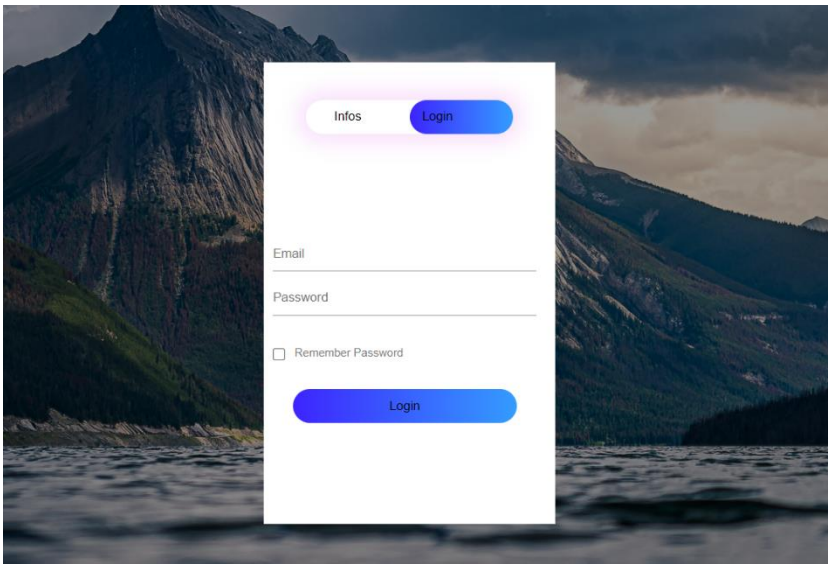
Prototype logiciel :

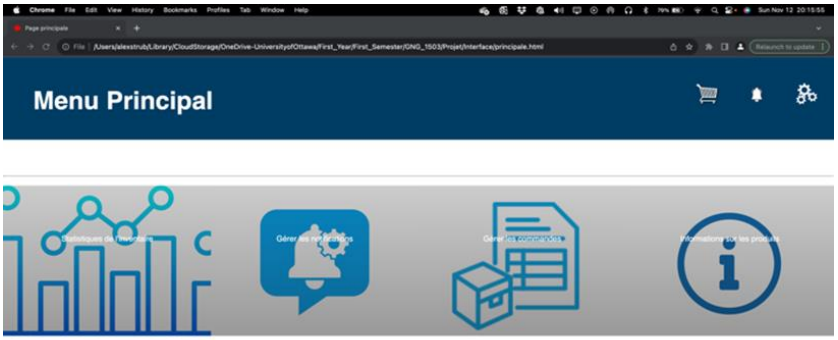
Pour le composant virtuel, la base de données et le site Web fonctionnent en permanence donc pas besoin d'éteindre le système. Si vous souhaitez quitter la page du site Web ayant la fonction de recherche, vous pouvez simplement appuyer sur le « x » en haut à droite de votre écran. Et pour fermer le système en python, il suffit de maintenir Ctrl et d'appuyer sur la touche « C », ce qui fermera le code dans la page du logiciel python.

4 Utiliser le système

Voici les différentes étapes pour utiliser notre produit :

-Premièrement pour avoir accès au site il vous sera premièrement demander de rentrer votre adresse courriel dans l'espace EMAIL et votre numéro dans l'espace PASSWORD d'employé et si ce dernier et valide vous aurez accès au site de la gestion d'inventaire





-Après avoir accéder au site web, vous vous retrouvez devant le menu principal où vous pouvez accéder aux différentes parties comme la liste des commandes effectuées, le centre de notifications et les paramètres (dans le coin supérieur droit).

- Et maintenant vous pouvez accéder aux statistiques, gérer les notifications, gérer les commandes et avoir des informations sur un produit spécifiques recherché en cliquant sur le menu de votre choix

-Dans le menu `gérer les notifications`, vous pouvez choisir la préférence de réception de notifications soit par courriel soit par SMS, ainsi que la fréquence de l'envoi des notifications.

-Pour avoir accès aux informations sur un produit, cliquer sur le bouton `information sur les produits`

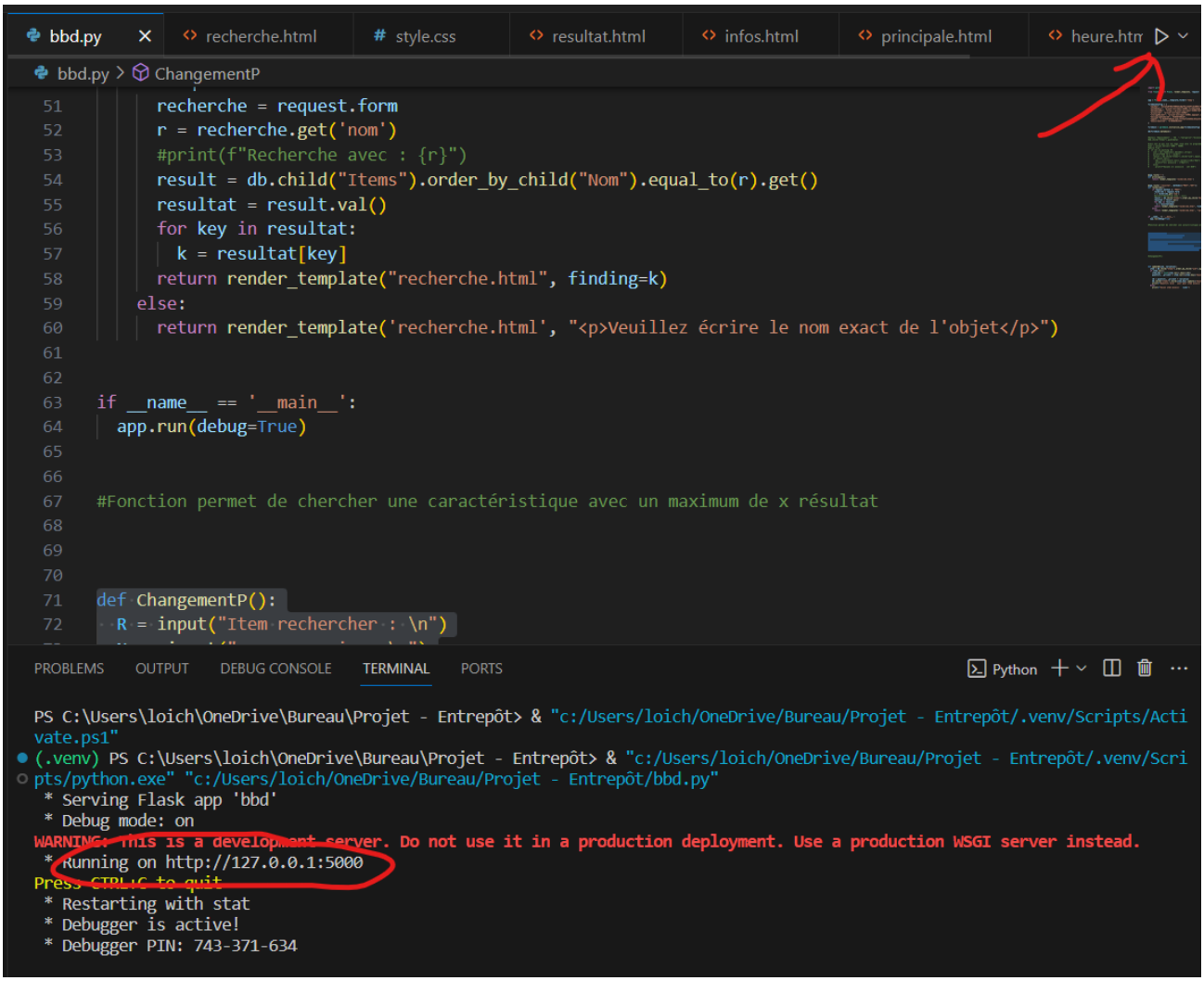
-Pour avoir accès aux statistiques cliquer sur le menu 'Statistiques de l'inventaire'

Pour pouvoir accéder aux commandes allez dans le menu `gérer les commandes`

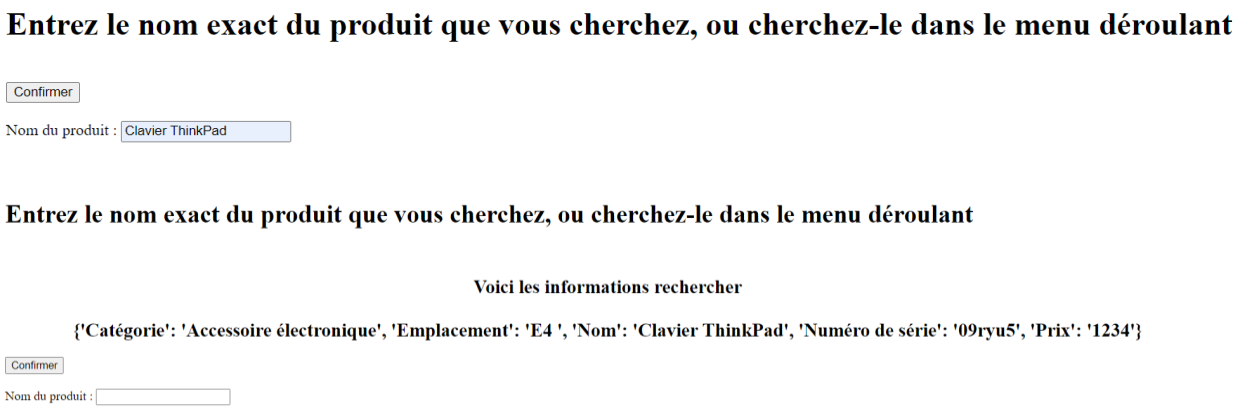
La sous-section suivante fourni des instructions détaillées, étape par étape, sur la façon d'utiliser la fonction de recherche.

4.1 Fonction de recherche

Pour avoir accès à la fonction de recherche, il faut ouvrir un fichier HTML créer par la librairie Flask en python. Pour ce faire il faut simplement faire courir (run) le code python une fois pour obtenir un lien http (vers un fichier HTML créé par flask).



Pour ouvrir le lien http il suffi de maintenir le bouton ctrl sur le clavier tout en appuyant sur le lien d'un clic gauche, ce qui ouvrira une page dans votre navigateur internet (Google Chrome, Firefox, etc.). Cette page permet la recherche d'un produit enregistré dans la base de données Firebase (Clavier ThinkPad pour cet exemple).



Une erreur s'affichera si l'utilisateur ne tape pas le nom exact d'un des produits dans la base de données Firebase. La page affichera une erreur du type « URL Not Found », ce qui démontre que la fonction de recherche n'a pas réussi à trouver le produit épelé.

4.1.1 <Sous-fonction/Sous-caractéristique donnée>

Notre système ne comprend aucune sous-fonctions.

5 Dépannage & assistance

Pour la récupération nous avons utilisé le site web GitHub y en sauvegardant nos systèmes à chaque modification apportée à ces systèmes. GitHub permet la récupération des sauvegardes ultérieur, ce qui est particulièrement utile si jamais une nouvelle version de code d'un des systèmes est corrompues ou présente des erreurs qui ne se produisait pas dans les anciennes versions.

5.1 Messages ou comportements d'erreur

Les connexions filaires avec la breadboard ou susceptibles d'être déconnectées, mais elles peuvent facilement être remises en place, si un fil se casse, il faudra en acheter un nouveau. Si la configuration est correcte, il ne devrait y avoir aucune erreur. Cependant, si vous recevez l'une des erreurs suivantes, voici comment les résoudre. Si vous obtenez une erreur « pip: command not found » ou un « 'pip' is not recognized as an internal or external command,
operable program or batch file.» erreur, vous devrez réinstaller python et pip. Assurez-vous de cocher ajouter python à PATH.

5.2 Considérations spéciales

Lors de l'utilisation du site web, il peut arriver certaines circonstances spéciales ou certaines erreurs qui peuvent être dus à:

-La surcharge du système (le système fonctionne lentement)

-La défaillance du système (le système n'exécute pas correctement les commandes)

-Piratages

-Un bug (le système ne répond plus)

Et pour ce faire dans le cas d'une surcharge du système, d'une défaillance du système ou d'un piratage, il faut préalablement contacter le service de maintenance au +18195764682 pour un dépannage rapide.

Dans le cas d’un bug du système, essayer de rafraichir la page ou de redémarrer le système.

Si le problème persiste veuillez contacter le service de maintenance au numéro suivant: +15142983637

5.3 Entretien

Pour la bonne fonctionnalité du site internet il faut effectuer:

-Une sauvegarde régulière

-Mise à jour régulière du système

-Rafraichir la page

-Vérification de la Sécurité

Pour la bonne fonctionnalité de la base de données il faut :

- Faire une surveillance continuelle de l’espace de stockage

-Faire une mise à jour régulière

-Nettoyage des données (supprimer les données obsolètes)

-Effectuer des tests de performances (pour évaluer la réactivité de la base de données sous charges importantes)

5.4 Assistance

En cas d’urgence veuillez contacter notre équipe d’assistance d’urgence disponible aux numéros suivants :

-+15142983637(Loïc) ; courriel de l’université d’Ottawa : lharv029@uottawa.ca

-+18195764682(Joseph)

6 Documentation du produit

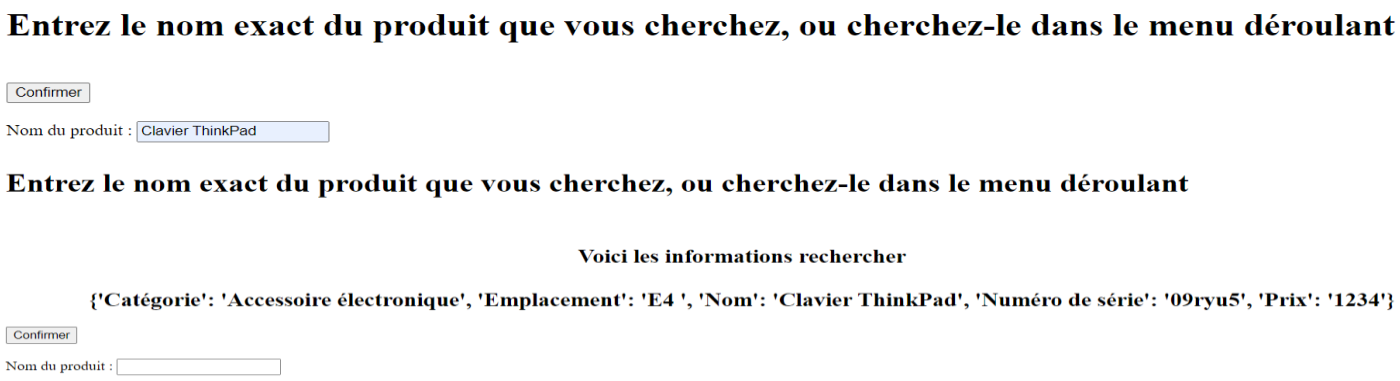
Tout d'abord le prototype final est consisté de trois systèmes : l'interface, le système Arduino et le programme d'automatisation. Ceci a été décidé à la suite de la création de notre énoncé de problème qui est comme suit : Créer un système de suivi d'inventaire au compte de Services Partagés Canada afin de réduire leurs pertes d'argent, de temps et de fiabilité. Ce système sera capable d'identifier, de localiser et de documenter les produits, ainsi que d'automatiser l'inventaire avec un réseau de notifications et de commandes intelligentes.

L'interface est un site web utilisant le langage HTML pour afficher l'information en provenance de notre base de données.

Le système Arduino est ce qui permet la lecture des cartes RFID pour changer la quantité des produits dans la base de données associées à ces cartes (utilisant une clé uid).

Le programme d'automatisation quant à lui est le backend de notre prototype. C'est-à-dire qu'il englobe la base de données, où toute l'information sur les produits est stockée, et le code en python qui permet la recherche de l'information sur un certain produit rechercher à travers l'interface (le site web).

Puisque nous devions nous préparer pour la journée de conception, ce prototype devait pouvoir présenter quelques pages HTML (qui est l'interface), scanner les cartes RFID pour changer les quantités afficher dans la base de données et afficher l'information lors d'une recherche faite via une des pages HTML. Jusqu'à présent, l'affichage de l'information à la suite d'une recherche se fait seulement à travers le code python. Malheureusement, aucune solution n'a encore été présenter puisque d'autre recherche sur le sujet s'impose.



Faute de temps, plusieurs autres fonctionnalités choisis initialement ont du être abandonnés comme le changement de prix via l'interface, l'automatisation du système (un comptage de la quantité fait de manière automatique et non faite manuellement par le système Arduino), la

documentation des produits (par l'entremise de graphique, aussi fait de manière automatique) et l'envoie de notifications (diverses, par exemple envoyer une notification suite à une quantité d'un produit plus bas que le seuil minimum prescrit). La fonction du changement de prix a été complété en python, mais elle n'a pas pu être lié à une des pages HTML. Ci-dessus est la fonction python du changement de prix.

```python
def ChangementP():
  R = input("Item rechercher : \n")
  Np = input("nouveau prix : \n")

  resultp=db.child("Items").order_by_child("Nom").equal_to(R).get()
  if resultp.val():
    item_key = list(resultp.val().keys())[0]
    db.child("Items").child(item_key).update({"Prix":Np})
    print(f"Le prix des '{R}' a été mis à jour ")
```

Voici notre NDM pour le prototype final :

#	Matériaux/composants	Prix/Unité ($)	Quantité	Prix total ($)
1	Base de données Firebase	0	2	0
2	Arduino IDE	0	1	0
3	Site web	0	1	0
4	Python IDE	0	1	0
5	Site GitHub	0	1	0
6	Logiciel Proteus	0	1	0
7	Plaque ESP8266 E12	15	2	30
8	Capteur RFID (makerLab Store)	1	2	2
9	Half breadboard (makerLab Store)	2.5	2	5
10	Jumper Wires (makerLab Store)	0.1	40	4
11	Petite Boîte de carton	0	2	0
12	Grosse boîte de carton	0	2	0

Base de données Firebase :

https://firebase.google.com/?gad_source=1&gclid=Cj0KCQiA4NWrBhD-ARIsAFCKwWvrbeJ3PT8-EvdBI8XavcQRU_DkLqGWpTtX_yMzpaxQjuJSSztlU5UaAiBmEALw_wcB&gclsrc=aw.ds

Arduino IDE: https://www.arduino.cc/en/software

Python IDE (Visual Studio Code): https://code.visualstudio.com/docs/languages/python

Site GitHub : https://github.com

Logiciel Proteus:

https://www.labcenter.com/proteus_pcb/?gclid=Cj0KCQiA4NWrBhD-ARIsAFCKwWsiC-Gy8_Phw2DNyed6etLNvbokkf4seXNBIpqtD9hoOFrz1-r3GEkaAgyTEALw_wcB

Plaque ESP8266 E12 : Trouver dans makerlab (pas dans l'inventaire sur le site web du makerlab)

Capteur RFID: Obtenus par l'entremise du makerlab store (pas dans l'inventaire sur le site web du makerlab)

Half breadboard: https://makerstore.ca/shop/ols/products/breadboard/v/B15-HLF

Jumper wires: https://makerstore.ca/shop/ols/products/jumper-cables-per-10

6.1 Programme d'automatisation

Notre sous-système 1 porte sur le programme d'automatisation, en d'autres mots le backend de notre système informatique.

6.1.1 NDM (Nomenclature des Matériaux)

#	Matériaux/composants	Prix/Unité ($)	Quantité	Prix total ($)
1	Python IDE	0	1	0

Python IDE (Visual Studio Code): https://code.visualstudio.com/docs/languages/python

6.1.2 Liste d'équipements

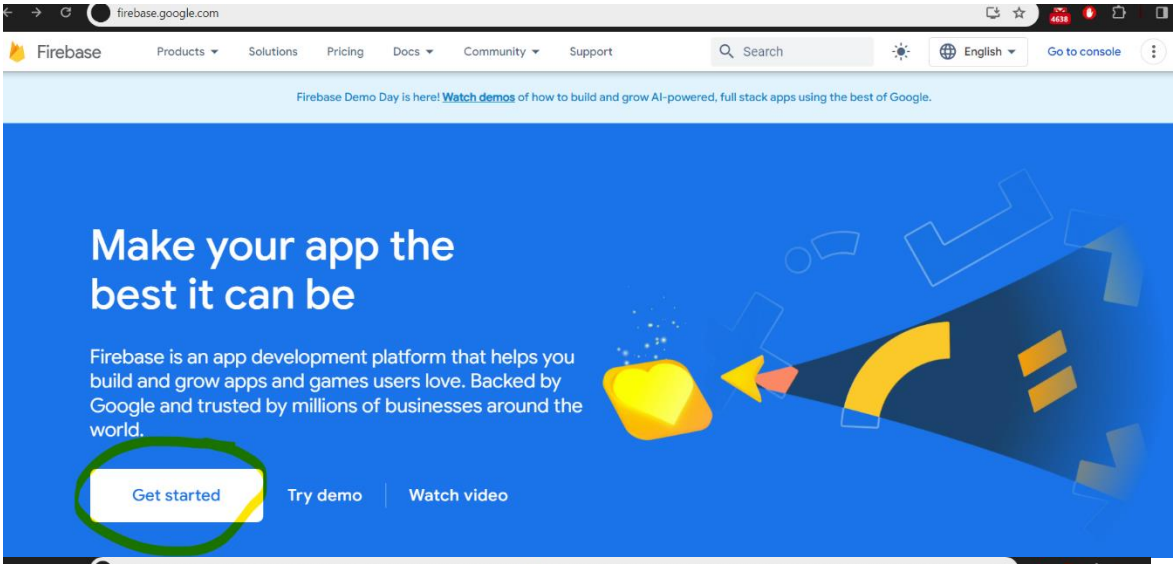
- Ordinateur
- Wifi
- Stockage pour au moins 10 GB

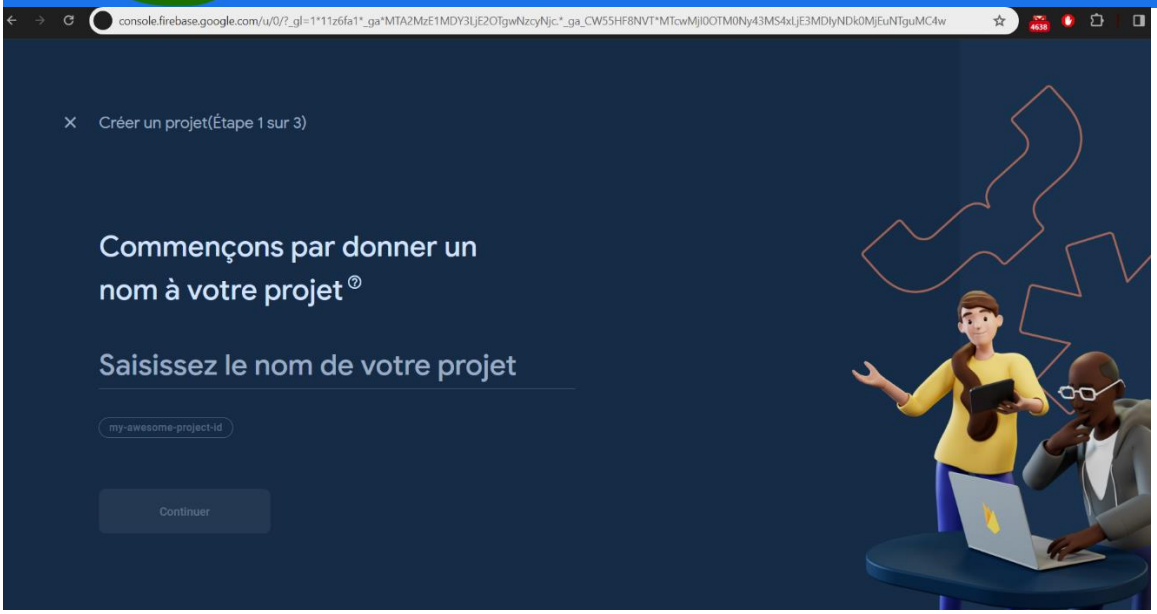
6.1.3 Instructions

Pour commencer, il faut installer le logiciel de Python qui sera utiliser pour pouvoir coder (ce projet a utilisé le Python IDE de Visual Studio Code). Ensuite dans le logiciel python il faut installer la librairie pyrebase et flask (de flask il faut « import » ou installer les librairie Flask, render_template et request). La première librairie permet d'interagir avec la base de données Firebase, alors que la deuxième permet d'envoyer des commandes au site web codé en HTML. Ce

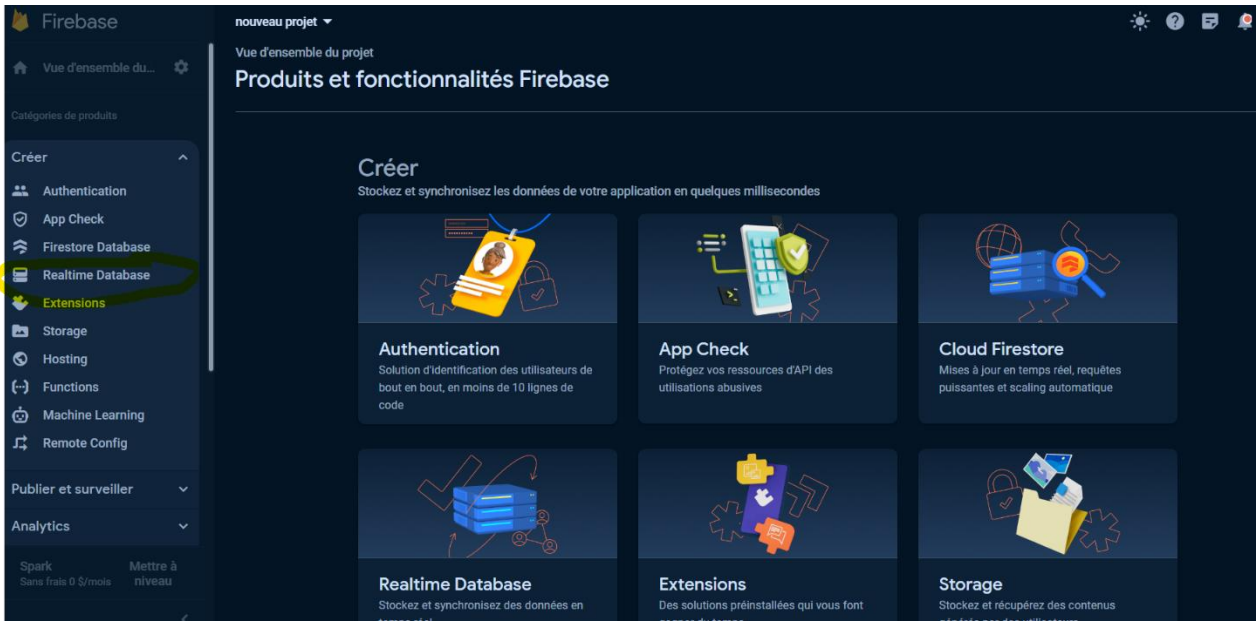
sous-système travail étroitement avec la base de données et le site web, donc pour des questions de simplifications cette partie parlera aussi de ces deux sections rapidement.

Premièrement, pour pouvoir interagir avec la base de données il faut créer un chemin lien la base de données au code python. Pour ce faire, il faut commencer par se créer un projet dans Firebase.

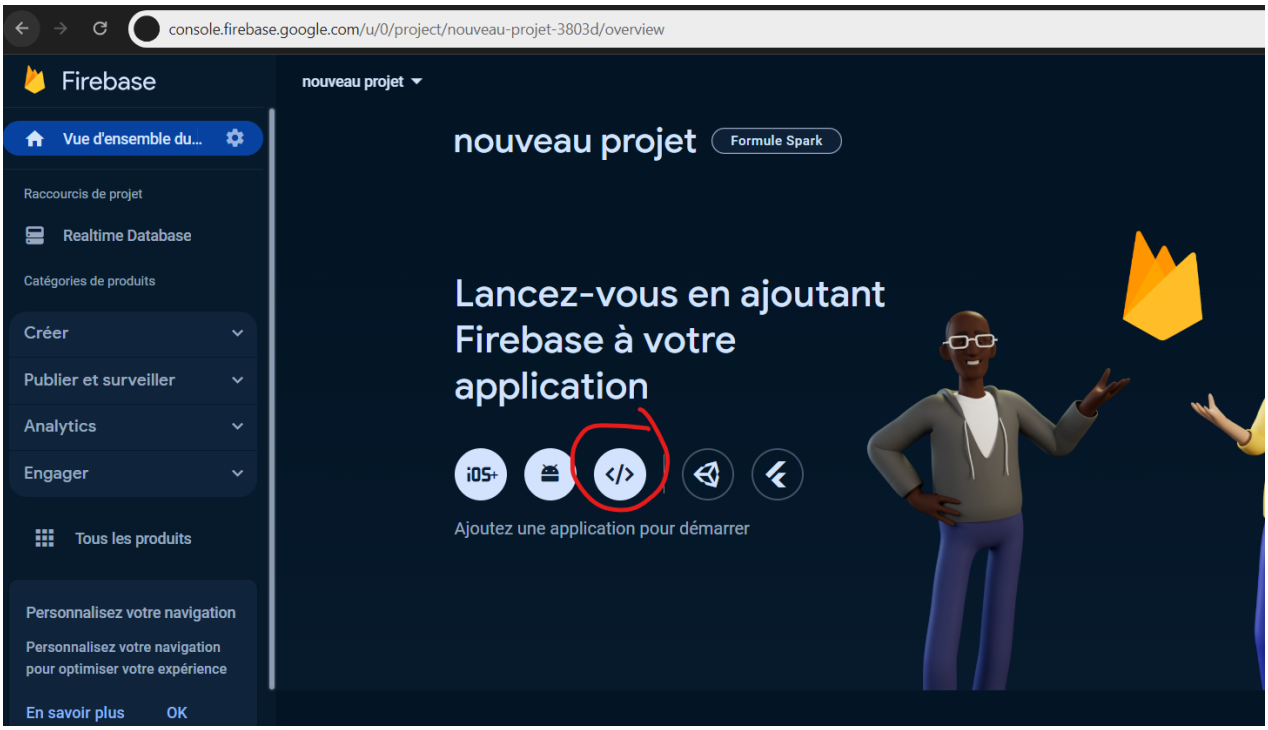




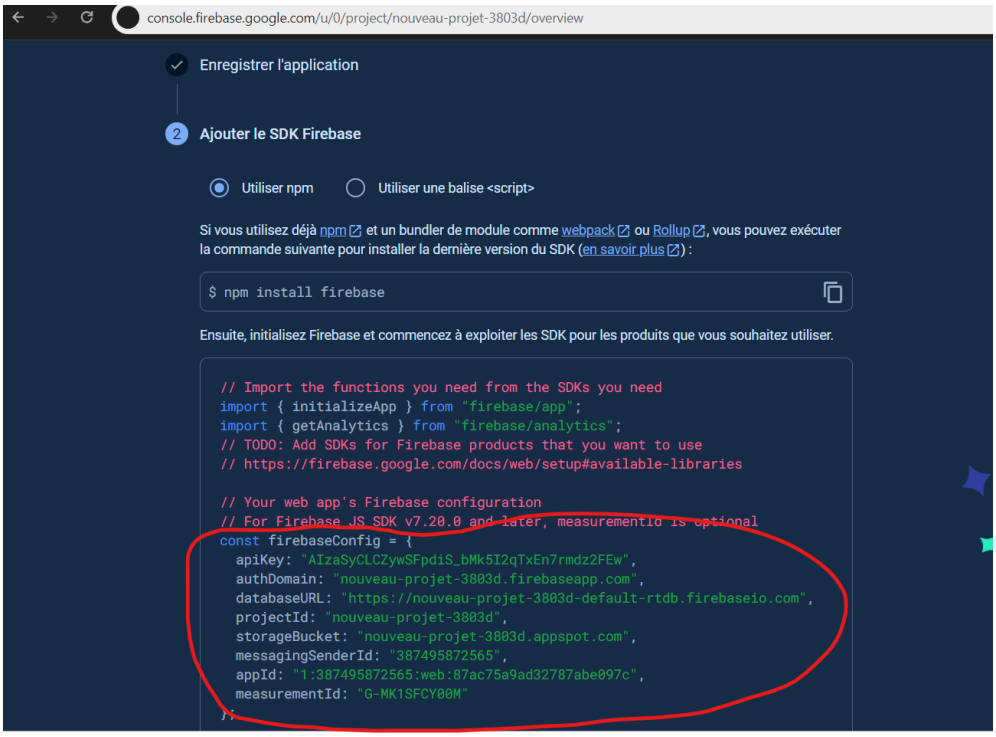
Une fois le projet créer, dans la section « Créer » à droite de l'écran il faut sélectionner la partie « Realtime Database » pour que la base de données soit construite. De là, le site vous demandera où sera localisé le serveur de la base de données et ensuite il vous demandera de commencer en mode verrouillé ou en mode test. Le mode test est celui que l'on veut puisqu'il permettra à la base de données de rendre publique toute information qui y est stocké. En autres mots il autorise l'accès à n'importe qui ayant votre clé API ou votre chemin de base de données de voir vos informations dans la base de données et de la modifier.

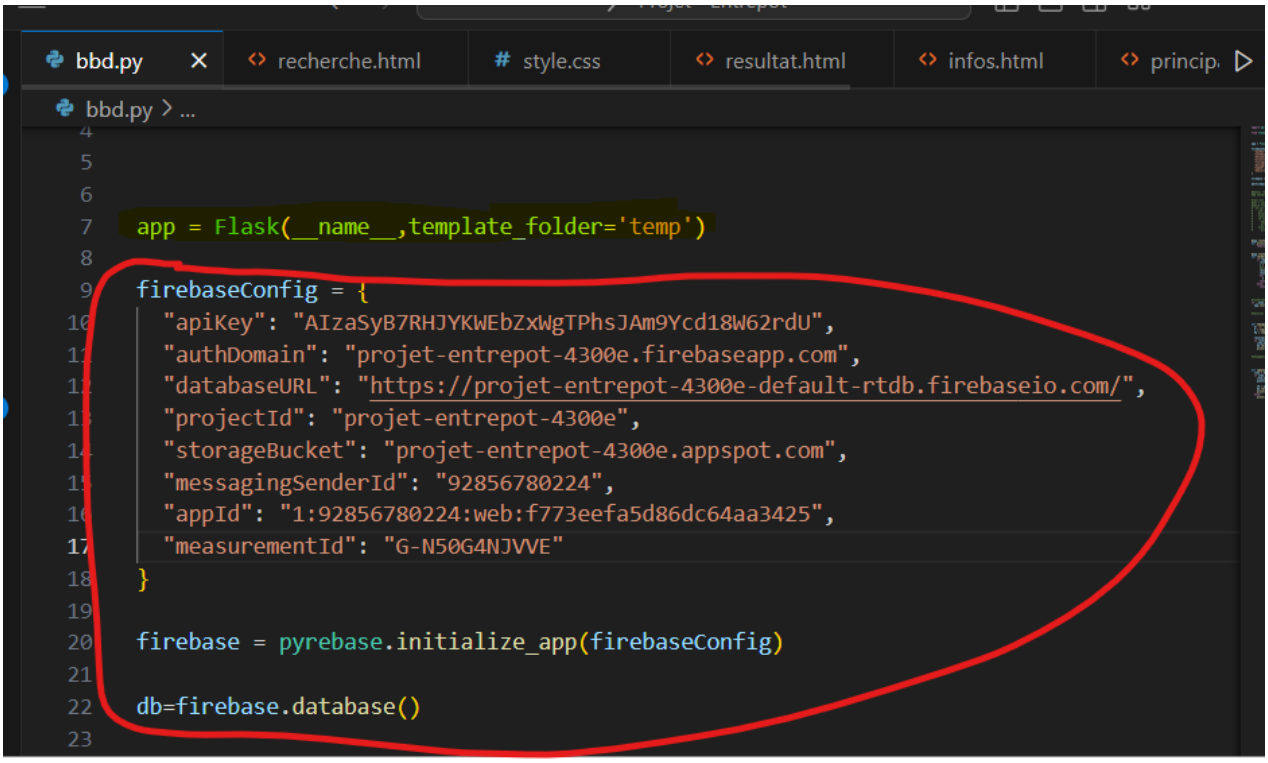


Après avoir créé la base de données il faut maintenant aller chercher le SDK Firebase permettant parler à Firebase au travers python. Pour cela il faut créer une application pour le site web comme montré dans la figure ci-bas.



Puis, après avoir choisi le nom de l'application, Firebase vous donnera le SDK. Ceci doit être copié et coller dans python tout en assignant une variable à ceci.





La ligne de code qui est surlignée est nécessaire lors de l'utilisation des pages HTML dans le code en python. Le « template_folder » est le porte document où doivent être tout fichier HTML utilisé

pour ce projet puisque cela précise à python où regarder lorsqu'on appel les différentes fichier HTML.

```python
@app.route('/')
def principale():
    return render_template('recherche.html')


@app.route('/resultat', methods=["POST","GET"])
def recherche():
    if request.method == "POST":
      recherche = request.form
      r = recherche.get('nom')
      #print(f"Recherche avec : {r}")
      result = db.child("Items").order_by_child("Nom").equal_to(r).get()
      resultat = result.val()
      for key in resultat:
        k = resultat[key]
      return render_template("recherche.html", finding=k)
    else:
      return render_template('recherche.html', "<p>Veuillez écrire le nom exact de l'objet</p>")


if __name__ == '__main__':
  app.run(debug=True)
```

Pour notre fonction de recherche, la route du « render_template » doit être le même que le nom du fichier HTML où l'action de cette fonction sera faite, les méthodes « POST » et « GET » explique que cette fonction recherche de l'information (en provenance de la base de données, « GET ») et l'affiche par l'entremise du chemin flask nommé « /resultat » (le « POST »).

6.2 Essais & validation

Nous avons effectué plusieurs essais afin de valider notre prototype. Nous avons adapté les essaies en fonction des prototypes.

D'abord, nous avons l'essai 1 relié au prototype 1 (ciblé analytique) qui avait pour but de vérifier le code Arduino. Nous avons fait la simulation du circuit final à l'aide du logiciel Proteus que nous avons connecté au code Arduino. Cela nous a permis de voir le comportement du circuit.

Ensuite le deuxième essai portait sur l'interface. Pour cet essai nous avons fait un sondage. Nous avons interrogé des utilisateurs potentiels sur l'esthétique et la facilité d'utilisation. Nous avons même fait appel à des fonctionnaire du gouvernement afin de s'assurer que cela correspondait aux critères du gouvernement

Aussi, nous avons fait un autre essai qui était la compatibilité entre l'Arduino et la bases de données mais ils ne se liait pas ensemble. Ce problème persista jusqu'à ce qu'on remplace notre plaque Arduino Uno pour une plaque Arduino ESP8266. Pour revérifier la connexion entre les deux systèmes nous avons scanner les cartes RFID à maintes reprises pour avoir comme résultat un changement dans la quantité (dans la base de données) des produits associés à ces cartes RFID.

Enfin l'essai 4 réunissait toutes les composantes du projet. On arrivait donc à effectuer une recherche de produit à travers le site web et le produit pouvait être scanner et compter dans la base de données.

7 Conclusions et recommandations pour les travaux futurs

Pour conclure, durant la réalisation de ce produit nous avons appris à travailler en équipe ce qui impliquait d'avoir confiance envers les autres membres, avoir une bonne communication, savoir s'organiser et de respecter ses engagements. Mais nous avons aussi appris certaines compétences techniques comme le codage et l'électronique, le sens de priorités et les responsabilités individuelles. S'il faut parler des pistes productives, il faut donc parler de faire de recherches de programmation afin de créer un système d'automatisation et de créer un système de notifications mais aussi de penser à la création d'une application.

Dans la mesure où nous aurions eu plus de temps pour réaliser ce produit, nous aurions essayé de finir toutes les fonctionnalités de notre comme par exemple de règlementer le système de notifications et se rassurer de la fonctionnalité du système d'automatisation.

Durant les trois mois de conception de ce produit nous sommes malheureusement arrivés a abandonner certaines fonctionnalités du produit comme la réalisation d'une application, la création d'un système d'automatisation et la réalisation d'un système de notifications liés à notre interface.

8 Bibliographie

- https://firebase.google.com/docs/guides
- https://www.baeldung.com/cs/http-get-vs-post#:~:text=The%20GET%20method%20is%20limited,which%20has%20no%20such%20limitation.
- https://flask.palletsprojects.com/en/3.0.x/
-

APPENDICES

9 APPENDICE I: Fichiers de conception

Table 1. Documents référencés

Nom du document	**Emplacement du document et/ou URL**	**Date d'émission**
GitHub du projet	https://github.com/UnknownFighter/GNG1503---Systeme	10 octobre 2023
Lien Makerrepo	https://makerepo.com/lharv/1874.systme-dinventaire-automatique-et-de-suivi-fa34	21 novembre 2023

10 APPENDICE II: Autres Appendices