

## 1- Commandes de recherche avancée( abdou )

Créer un fichier .txt dans les 5 dossiers et intégrer la phrase:

“Que la force soit avec toi.”

echo “Que la force soit avec toi.” >Bureau/mon\_texte.txt

echo “Que la force soit avec toi.” >Documents/mon\_texte.txt

echo “Que la force soit avec toi.” >Téléchargements/mon\_texte.txt

echo “Que la force soit avec toi.” >Vidéos/mon\_texte.txt

echo “Que la force soit avec toi.” >Images/mon\_texte.txt

Permet de visualiser tous les fichier dans lesquels figure le mot “force”:

grep -rl “force”

## 2- Compression et décompression de fichiers( abdou )

Créer un fichier Plateforme à l'intérieur du fichier Documents:

mkdir ~/Documents/Plateforme

Copie le fichier .txt qui se trouve dans le fichier Documents dans le fichier Plateforme:

cp ~/Documents/mon\_texte.txt ~/Documents/Plateforme

Se rendre dans le dossier Plateforme puis répétez cette commande 4 fois afin d'avoir 4 fichier dans le dossier:

cp mon\_texte.txt mon\_texte(2,3,4,5).txt

Taper la commande suivante afin de créer le fichier compresser:

tar czvf Plateforme.tar.gz Plateforme

Taper la commande suivante afin de créer le fichier décompression:

tar xzvf Plateforme.tar.gz Plateforme

# Manipulation de texte ( clement )

```
import csv
data = [
    ["Jean", "25 ans", "Paris"],
    ["Marie", "30 ans", "Lyon"],
    ["Pierre", "22 ans", "Marseille"],
    ["Sophie", "35 ans", "Toulouse"]
]
csv_file = "personnes.csv"
with open(csv_file, mode='w', newline='') as file:
    writer = csv.writer(file)
    writer.writerows(data)

print("Fichier CSV créé avec succès : ", csv_file)
Commande bash :
awk -F',' '{print $3}' personnes.csv
```

# GESTION DES PROCESSUS (ALLAOUI)

La commande **ps** recense tous les processus actifs sur notre système.

```
laplateforme@debian: ~  
laplateforme@debian:~$ ps  
  PID TTY          TIME CMD  
  7172 pts/0    00:00:00 bash  
  7928 pts/0    00:00:00 ps  
laplateforme@debian:~$
```

La commande **ps aux**, affiche tous les processus actifs avec des informations détaillées telles que l'utilisateur qui a lancé le processus, le PID, l'utilisation de la CPU, etc.

```
laplateforme@debian: ~  
laplateforme@debian:~$ ps aux  
USER          PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND  
root             1  0.0  0.3 168128 12676 ?        Ss   10:28   0:03 /sbin/init  
root             2  0.0  0.0      0     0 ?        S    10:28   0:00 [kthreadd]  
root             3  0.0  0.0      0     0 ?        I<   10:28   0:00 [rcu_gp]
```

Pour fermer un processus spécifique, on utilisera la commande **kill** suivie du **PID** du processus que l'on souhaite arrêter.

Par exemple, pour arrêter le processus avec le **PID 7940**,

```
laplate+ 7858 0.1 1.4 3017252 59380 ?        Sl   17:39   0:00 gjs /home/laplatefo  
root     7935 0.0 0.0      0     0 ?        I    17:43   0:00 [kworker/3:0-events  
laplate+ 7940 140 9.4 11286848 379084 ?        Sl   17:45   0:06 /usr/lib/firefox-es  
laplate+ 8000 1.8 0.8 215256 34712 ?        Sl   17:45   0:00 /usr/lib/firefox-es  
laplate+ 8038 19.4 3.0 10836160 121796 ?       Sl   17:45   0:00 /usr/lib/firefox-es  
laplate+ 8098 24.0 2.6 2449096 105616 ?       Sl   17:45   0:00 /usr/lib/firefox-es  
laplate+ 8148 12.1 1.7 2411620 69592 ?        Sl   17:45   0:00 /usr/lib/firefox-es  
laplate+ 8153 10.8 1.7 2411620 70084 ?        Sl   17:45   0:00 /usr/lib/firefox-es  
laplate+ 8198 10.8 1.7 2412216 68664 ?        Sl   17:45   0:00 /usr/lib/firefox-es  
laplate+ 8224 100 0.1 11216 4916 pts/0    R+   17:45   0:00 ps aux  
laplateforme@debian:~$
```

On exécutera la commande : **kill 7940**

Cela enverra le signal de terminaison par défaut (**SIGTERM**) au processus spécifié, lui demandant de s'arrêter proprement.

Si l'on souhaite maintenant terminer un processus de manière forcée, on utilisera la commande **kill** avec le signal **SIGKILL (9)**. **Exemple : kill -9 7940**

Cela force la terminaison immédiate du processus sans lui laisser le temps de s'arrêter proprement.

# SURVEILLANCE DES RESSOURCES SYSTÈME (ALLAOUI)

Pour mettre en place une surveillance en temps réel de l'utilisation du CPU, de la mémoire et d'autres ressources système.

On peut le faire soit avec **vmstat** ou **top** avec l'aide de **awk** pour extraire les

**vmstat** : c'est un utilitaire de surveillance, qui fournit également des informations sur l'activité de bloc I/O et de la CPU en plus de la mémoire.

C'est un très bon outil pour mesurer les performances de Linux

```
laplateforme@debian:~$ vmstat 1
procs -----memory----- --swap-- -----io----- -system-- -----cpu-----
r  b   swpd   free   buff  cache   si   so    bi    bo    in   cs us sy id wa st
1  0    9020 356940  74652 2178996    0    0    11     2   16   26  0  0 100  0  0
2  0    9020 356940  74652 2178996    0    0     0     0  251  352  0  0  99  0  0
1  0    9020 356940  74652 2178996    0    0     0    12  687 1063  0  1  99  0  0
```

Pour le faire avec on utilisera la commande :

```
vmstat 5 10 | awk '{print $1,$2,$3,$4,$5,$6,$7,$8,$9,$10}' > stats.csv
```

**5** -> indique un intervalle de 5 secondes entre chaque relevé(requête)

**10** -> Spécifie le nombre total de relevés à effectuer

Cela signifie que ça va effectuer 10 relevés avec une collecte toutes les 5 secondes

Ensuite, on va extraire avec awk les 10 valeurs de chaque colonne et les enregistrer dans un fichier CSV qu'on nommera **stats.csv**. Chaque ligne du fichier CSV contiendra les valeurs des 10 colonnes

The screenshot displays a Linux desktop with a terminal window and a LibreOffice Calc spreadsheet. The terminal window shows the command `vmstat 5 10 | awk '{print $1,$2,$3,$4,$5,$6,$7,$8,$9,$10}' > stats.csv` being executed. The LibreOffice Calc spreadsheet shows the resulting data in a CSV file, with columns labeled 'procs', 'memory', 'swap', 'io', 'system', and 'cpu'. The data is organized into rows, with the first row containing the headers and the subsequent rows containing the values for each column.

**top** : C'est un genre de gestionnaire de tâches qui permet de faire beaucoup de choses avec les processus.

Avec cette application on peut :

- Lister et interagir avec les processus en cours d'exécution
- Visualiser l'utilisation CPU et mémoire globale et par processus
- Vérifier l'utilisation globale de la CPU et mémoire

Voici à quoi il ressemble :

```
laplateforme@debian: ~  
top - 23:30:16 up 12:19, 1 user, load average: 0.17, 0.05, 0.01  
Tasks: 307 total, 1 running, 306 sleeping, 0 stopped, 0 zombie  
%Cpu(s): 0.0 us, 0.0 sy, 0.0 ni, 99.9 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st  
MiB Mem : 3913.2 total, 346.0 free, 1645.0 used, 2201.2 buff/cache  
MiB Swap: 976.0 total, 967.2 free, 8.8 used, 2268.2 avail Mem
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
732	root	20	0	241212	11328	7552	S	0.3	0.3	1:00.12	vmtoolsd
10401	laplate+	20	0	2943628	59128	45032	S	0.3	1.5	0:00.36	gjs
10475	laplate+	20	0	11604	5040	3136	R	0.3	0.1	0:00.06	top
1	root	20	0	169328	14024	9256	S	0.0	0.3	0:05.70	systemd
2	root	20	0	0	0	0	S	0.0	0.0	0:00.10	kthreadd
3	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	rcu_gp
4	root	0	-20	0	0	0	I	0.0	0.0	0:00.00	rcu_par_gp

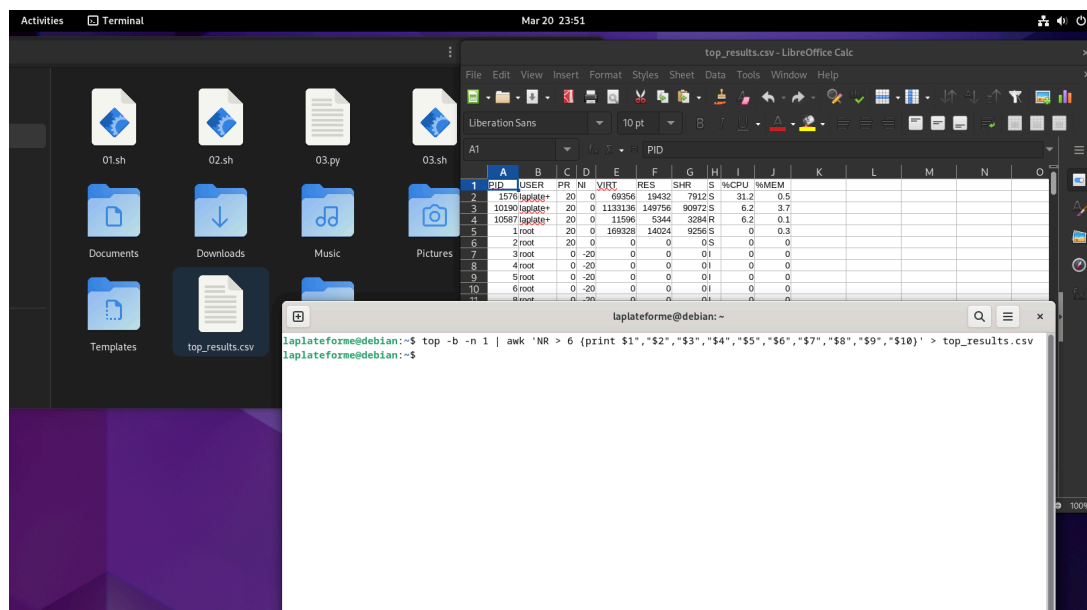
Pour mettre en place la surveillance avec top, on utilisera la commande :

**top -b -n 1 | awk 'NR > 6 {print \$1,\$2,\$3,\$4,\$5,\$6,\$7,\$8,\$9,\$10}' > top\_results.csv**

**-b** -> Cette option est utilisée pour activer le mode **batch**. En mode batch, **top** collecte les informations sur les processus et les affiche une seule fois, puis se termine immédiatement. Cela est utile pour automatiser la surveillance des performances ou pour extraire des données à des fins de traitement ultérieur.

**-n 1** -> Cette option permet à la commande **top** de s'arrêter automatiquement après un ? nombre de répétitions.

**NR** -> Cette option est pour permettre à la commande de **awk** d'extraire les informations à partir d'une ligne spécifique



# Scripting avancé( abdou )

méthode pour créer le fichier.sh et ainsi renseigner le script:  
nano nomduscript.sh.

Puis rentrer les ligne de commande afin de créer le script demander:

```
#!/bin/bash
```

```
# Répertoire à sauvegarder  
répertoire source="Plateforme"
```

```
# Répertoire où sauvegarder les archives  
répertoire sauvegarde="Plateforme"
```

```
# Fonction pour créer une sauvegarde  
sauvegarder() {  
    # Créer le répertoire de sauvegarde s'il n'existe pas déjà  
    mkdir -p "$répertoire sauvegarde"  
  
    # Nom de l'archive avec la date et l'heure actuelles  
    nom_archive="Plateforme $(date +%Y%m%d_%H%M%S').tar.gz"
```

```
    # Chemin complet de l'archive  
    chemin archive="$répertoire sauvegarde/$nom_archive"
```

```
    # Archiver le répertoire Plateforme  
    tar czf "$chemin_archive" "$répertoire_source"
```

```
    # Enregistrer l'archive dans un fichier d'historique  
    echo "$(date +%Y-%m-%d %H:%M:%S) : $nom_archive" >> "$répertoire  
sauvegarde/history.txt"
```

```
    # Afficher un message de confirmation  
    echo "Sauvegarde terminée : $chemin archive"  
}
```

```
# Fonction pour afficher l'historique des sauvegardes  
afficher historique() {  
    echo "Historique des sauvegardes :"  
    cat "$répertoire sauvegarde/history.txt"  
}
```

```
# Vérifier si une action est spécifiée en argument  
if [ "$1" == "sauvegarder" ]; then  
    # Appeler la fonction pour créer la sauvegarde  
    sauvegarder
```

```

elif [ "$1" == "historique" ]; then
    # Appeler la fonction pour afficher l'historique
    afficher historique
else
    echo "Utilisation : "
    echo "Pour effectuer une sauvegarde : $0 sauvegarder"
    echo "Pour afficher l'historique des sauvegardes : $0 historique"
fi

```

Attribuer les droits d'exécution, afin de pouvoir exécuter le script avec l'utilisateur :  
 chmod +x nomduscript.sh

## Automatisation des mises à jour logicielles( clement )

```

sudo apt update
echo "Recherche des mises à jour disponibles..."
if ! updates=$(apt list --upgradable 2>/dev/null); then
    echo "Erreur: Impossible de récupérer la liste des mises à jour."
    exit 1
fi
if [[ -z $updates ]]; then
    echo "Aucune mise à jour disponible."
else
    echo "Mises à jour disponibles : "
    echo "$updates"
    read -p "Voulez-vous procéder à la mise à jour des logiciels ? (O/n) " choice
    case "$choice" in
        y|Y|[oO]|[yY]|[eE]|[sS]|"")
            echo "Mise à jour en cours..."
            sudo apt upgrade -y
            echo "Mise à jour terminée."
            ;;
        n|N|[nN]|[oO])
            echo "Mise à jour annulée."
            ;;
        *)
            echo "Choix non reconnu, mise à jour annulée."
            ;;
    esac
fi

```

# Gestion des dépendances logicielles (clement)

```
if [[ $EUID -ne 0 ]]; then
    echo "Ce script doit être exécuté en tant que root."
    exit 1
fi
```

```
install_packages() {
    echo "Installation de $1..."
    apt-get install -y "$1" >/dev/null 2>&1
    if [ $? -eq 0 ]; then
        echo "$1 installé avec succès."
    else
        echo "Erreur lors de l'installation de $1."
        exit 1
    fi
}
```

```
read -p "Voulez-vous installer Apache ou Nginx ? (a/n) " web_server
case "$web_server" in
    a|A)
        install_packages apache2
        ;;
    n|N)
        install_packages nginx
        ;;
    *)
        echo "Choix non reconnu, installation annulée."
        exit 1
        ;;
esac
```

```
# Installation de phpMyAdmin
install_packages phpmyadmin
```

```
# Installation d'un système de gestion de base de données relationnelle (MySQL ou MariaDB)
```



```
read -p "Voulez-vous installer MySQL ou MariaDB ? (m/M) " db_server
case "$db_server" in
    m|M)
        install_packages mysql-server
        ;;
    *)
        install_packages mariadb-server
        ;;
esac

# Installation de Node.js avec npm
install_packages nodejs
install_packages npm

# Installation de Git
install_packages git

echo "Installation terminée avec succès."
```

## Sécuriser ses scripts ( clément )

-Injection de commandes (Command Injection) : Si les scripts n'effectuent pas correctement la validation et l'échappement des entrées utilisateur, cela peut permettre à un attaquant d'injecter des commandes malveillantes dans les scripts et d'exécuter des actions non autorisées sur le système.

-Accès non autorisé (Unauthorized Access) : Les scripts exécutés avec des privilèges élevés, tels que ceux nécessitant l'accès root, peuvent être exploités pour accéder à des ressources sensibles ou modifier des fichiers système critiques.

-Exposition de données sensibles (Sensitive Data Exposure) : Si les scripts manipulent des données sensibles telles que des mots de passe, des clés API, des informations d'identification, etc., sans prendre les mesures appropriées pour les protéger, ces données pourraient être exposées à des tiers non autorisés.

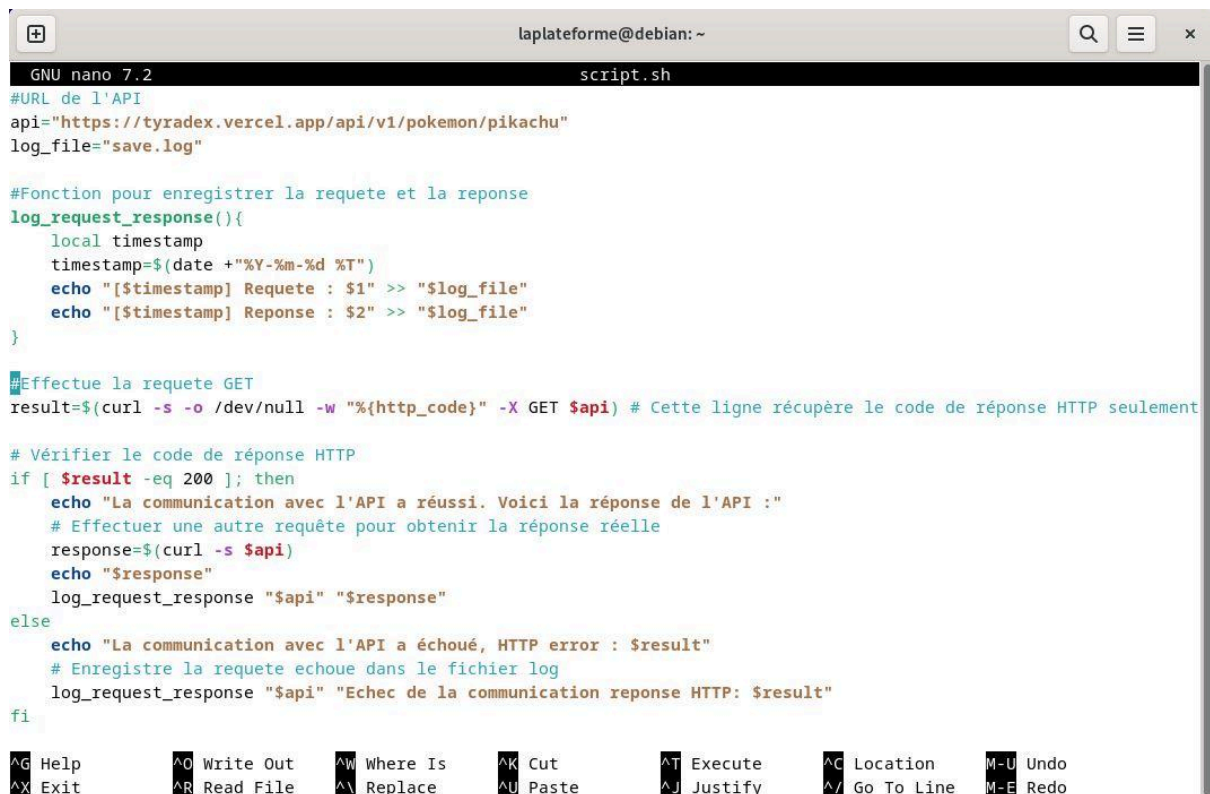
-Exécution de code malveillant (Remote Code Execution) : Des vulnérabilités dans les scripts pourraient permettre à des attaquants distants d'exécuter du code malveillant sur le serveur, compromettant ainsi la sécurité du système.

# UTILISATION D'API WEB DANS UN SCRIPT (ALLAOU)

Ici on va essayer de communiquer avec l'API =  
*<https://tyradex.vercel.app/api/v1/pokemon/pikachu>*

Ensuite pour assurer une communication secure, on vérifie et s'assure que notre API utilise un **TLS** (anciennement connu sous le nom de **SSL**) en regardant si son URL commence par **https://** au lieu de **http://**

*PS : TLS chiffre les messages en transit, protégeant ainsi les informations sensibles telles que les identifiants d'API.*



```
GNU nano 7.2                                script.sh
#URL de l'API
api="https://tyradex.vercel.app/api/v1/pokemon/pikachu"
log_file="save.log"

#Fonction pour enregistrer la requete et la reponse
log_request_response(){
    local timestamp
    timestamp=$(date +"%Y-%m-%d %T")
    echo "[${timestamp}] Requete : $1" >> "$log_file"
    echo "[${timestamp}] Reponse : $2" >> "$log_file"
}

#Effectue la requete GET
result=$(curl -s -o /dev/null -w "%{http_code}" -X GET $api) # Cette ligne récupère le code de réponse HTTP seulement

# Vérifier le code de réponse HTTP
if [ $result -eq 200 ]; then
    echo "La communication avec l'API a réussi. Voici la réponse de l'API :"
    # Effectuer une autre requête pour obtenir la réponse réelle
    response=$(curl -s $api)
    echo "$response"
    log_request_response "$api" "$response"
else
    echo "La communication avec l'API a échoué, HTTP error : $result"
    # Enregistre la requete echoue dans le fichier log
    log_request_response "$api" "Echec de la communication reponse HTTP: $result"
fi

^G Help      ^O Write Out  ^W Where Is   ^K Cut        ^T Execute    ^C Location   M-U Undo
^X Exit      ^R Read File  ^\ Replace    ^U Paste      ^J Justify    ^_ Go To Line  M-E Redo
```

**Curl** -> est un outil permettant de transférer des données depuis ou vers un serveur à l'aide d'URL.

Il prend en charge les protocoles suivants : DICT, FILE, FTP, FTPS, GOPHER, GOPHERS, HTTP, HTTPS, IMAP, IMAPS, LDAP, LDAPS, MQTT, POP3, POP3S, RTMP, RTMPS, RTSP, SCP, SFTP, SMB, SMBS, SMTP, SMTPS, TELNET, TFTP, WS et WSS.

**-X GET** -> spécifie que l'on effectue une requête **HTTP GET**

**-s** -> L'option **-s**, également appelée **--silent**, indique à **curl** de supprimer la sortie qu'il affiche normalement sur la console.

Cela signifie qu'on ne verra pas les barres de progression, les en-têtes ou le contenu transféré lui-même en utilisant **-s**

**-o /dev/null** -> **output**, cette option indique à **curl** de rediriger la sortie du transfert vers **/dev/null**.

**/dev/null** est un fichier spécial dans les systèmes Unix-like qui rejette tout ce qui est écrit dedans. En pratique, cela signifie que le contenu téléchargé ne sera pas stocké sur notre machine.

On l'utilise par ici par exemple lorsqu'on ne s'intéresse qu'au code de statut **HTTP** et pas au contenu lui-même.

**-w "%{http\_code}"** -> Cette option permet de personnaliser la sortie de **curl**.

Plus précisément, elle indique à **curl** d'afficher uniquement le code de statut **HTTP** à la fin du transfert.

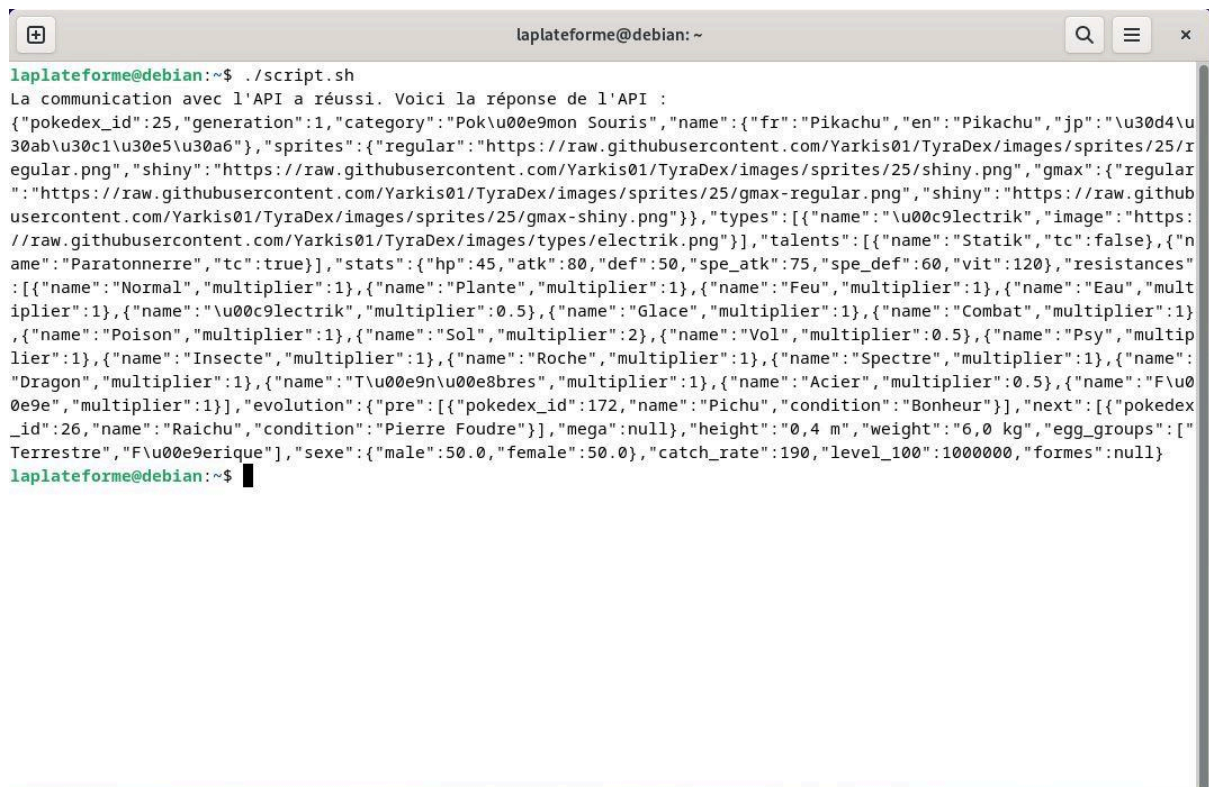
Le code de statut HTTP est un nombre à trois chiffres qui indique le résultat d'une requête HTTP.

Par exemple, un code **200** signifie que la demande a été réussie, tandis qu'un code **404** signifie que la ressource demandée n'a pas été trouvée.

## RÉSULTAT :

Exécution du script après un :

**Un succès**



```
laplateforme@debian: ~  
laplateforme@debian:~$ ./script.sh  
La communication avec l'API a réussi. Voici la réponse de l'API :  
{  
  "pokedex_id": 25,  
  "generation": 1,  
  "category": "Pok\u00e9mon Souris",  
  "name": {  
    "fr": "Pikachu",  
    "en": "Pikachu",  
    "jp": "\u30d4\u30ab\u30c1\u30e5\u30a6",  
    "sprites": {  
      "regular": "https://raw.githubusercontent.com/Yarkis01/TyraDex/images/sprites/25/regular.png",  
      "shiny": "https://raw.githubusercontent.com/Yarkis01/TyraDex/images/sprites/25/shiny.png",  
      "gmax": {  
        "regular": "https://raw.githubusercontent.com/Yarkis01/TyraDex/images/sprites/25/gmax-regular.png",  
        "shiny": "https://raw.githubusercontent.com/Yarkis01/TyraDex/images/sprites/25/gmax-shiny.png"}  
      },  
      "types": [{  
        "name": "\u00c9lectrik",  
        "image": "https://raw.githubusercontent.com/Yarkis01/TyraDex/images/types/electrik.png"}  
      ],  
      "talents": [{  
        "name": "Statik",  
        "tc": false},  
        {"name": "Paratonnerre", "tc": true}],  
      "stats": {  
        "hp": 45, "atk": 80, "def": 50, "spe_atk": 75, "spe_def": 60, "vit": 120},  
      "resistances": [{  
        "name": "Normal", "multiplier": 1},  
        {"name": "Plante", "multiplier": 1},  
        {"name": "Feu", "multiplier": 1},  
        {"name": "Eau", "multiplier": 1},  
        {"name": "\u00c9lectrik", "multiplier": 0.5},  
        {"name": "Glace", "multiplier": 1},  
        {"name": "Combat", "multiplier": 1},  
        {"name": "Poison", "multiplier": 1},  
        {"name": "Sol", "multiplier": 2},  
        {"name": "Vol", "multiplier": 0.5},  
        {"name": "Psy", "multiplier": 1},  
        {"name": "Insecte", "multiplier": 1},  
        {"name": "Roche", "multiplier": 1},  
        {"name": "Spectre", "multiplier": 1},  
        {"name": "Dragon", "multiplier": 1},  
        {"name": "T\u00e9n\u00e9bres", "multiplier": 1},  
        {"name": "Acier", "multiplier": 0.5},  
        {"name": "F\u00e9e", "multiplier": 1}],  
      "evolution": {  
        "pre": [{"pokedex_id": 172, "name": "Pichu", "condition": "Bonheur"}],  
        "next": [{"pokedex_id": 26, "name": "Raichu", "condition": "Pierre Foudre"}],  
        "mega": null, "height": "0,4 m", "weight": "6,0 kg", "egg_groups": ["Terrestre", "F\u00e9erique"], "sexe": {"male": 50.0, "female": 50.0}, "catch_rate": 190, "level_100": 1000000, "formes": null}  
}
```

## Un échec

```
laplateforme@debian:~$ ./script.sh
La communication avec l'API a échoué, HTTP error : 000
laplateforme@debian:~$
```

Aperçu des requêtes et réponse sauvegarder envoyer vers l'API -> **save.log**

```
laplateforme@debian: ~
laplateforme@debian:~$ cat save.log
[2024-03-25 21:45:56] Requete : https://tyradex.vercel.p/api/v1/pokemon/pikachu
[2024-03-25 21:45:56] Requete : Echec de la communication reponse HTTP: 000
[2024-03-25 21:46:30] Requete : https://tyradex.vercel.app/api/v1/pokemon/pikachu
[2024-03-25 21:46:30] Requete : {"pokedex_id":25,"generation":1,"category":"Pok\u00e9mon Souris","name":{"fr":"Pikachu","en":"Pikachu","jp":"\u30d4\u30ab\u30c1\u30e5\u30a6"},"sprites":{"regular":"https://raw.githubusercontent.com/Yarkis01/TyraDex/images/sprites/25/regular.png","shiny":"https://raw.githubusercontent.com/Yarkis01/TyraDex/images/sprites/25/shiny.png","gmax":{"regular":"https://raw.githubusercontent.com/Yarkis01/TyraDex/images/sprites/25/gmax-regular.png","shiny":"https://raw.githubusercontent.com/Yarkis01/TyraDex/images/sprites/25/gmax-shiny.png"}}, "types":[{"name":"\u00c9lectrik","image":"https://raw.githubusercontent.com/Yarkis01/TyraDex/images/types/electrik.png"}], "talents":[{"name":"Statik","tc":false}, {"name":"Paratonnerre","tc":true}], "stats":{"hp":45,"atk":80,"def":50,"spe_atk":75,"spe_def":60,"vit":120}, "resistances":[{"name":"Normal","multiplier":1}, {"name":"Plante","multiplier":1}, {"name":"Feu","multiplier":1}, {"name":"Eau","multiplier":1}, {"name":"\u00c9lectrik","multiplier":0.5}, {"name":"Glace","multiplier":1}, {"name":"Combat","multiplier":1}, {"name":"Poison","multiplier":1}, {"name":"Sol","multiplier":2}, {"name":"Vol","multiplier":0.5}, {"name":"Psy","multiplier":1}, {"name":"Insecte","multiplier":1}, {"name":"Roche","multiplier":1}, {"name":"Spectre","multiplier":1}, {"name":"Dragon","multiplier":1}, {"name":"T\u00e9n\u00e9bres","multiplier":1}, {"name":"Acier","multiplier":0.5}, {"name":"F\u00e9e","multiplier":1}], "evolution":{"pre":{"pokedex_id":172,"name":"Pichu","condition":"Bonheur"},"next":{"pokedex_id":26,"name":"Raichu","condition":"Pierre Foudre"},"mega":null,"height":"0,4 m","weight":"6,0 kg"},"egg_groups":["Terrestre","F\u00e9erique"],"sexe":{"male":50.0,"female":50.0},"catch_rate":190,"level_100":100000,"formes":null}
laplateforme@debian:~$
```