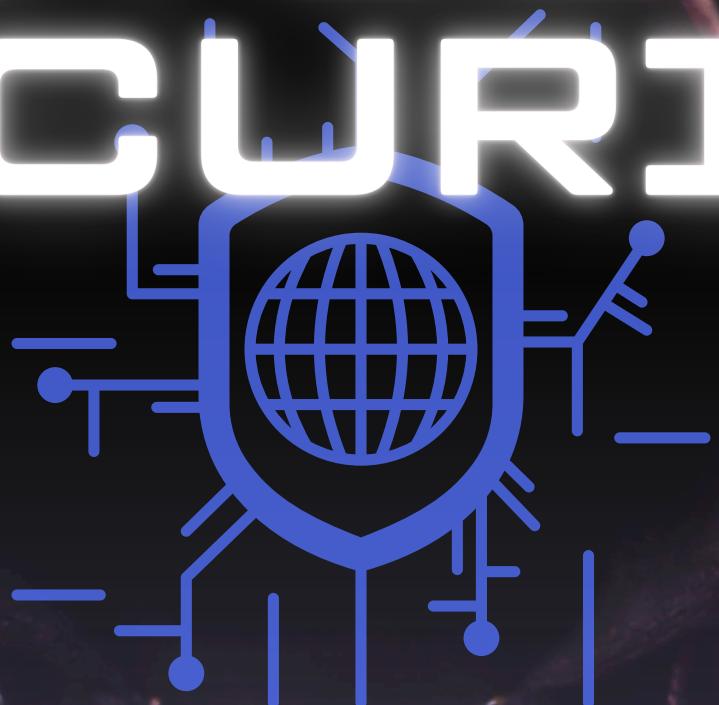


# CYBER SECURITY



Présenter par:

BOUCHAMA khadija

El guerdani Amal

Encadrer par:

Madame bouzaachane

# Cyber security

La cybersécurité est un domaine multidisciplinaire qui englobe toutes les technologies, processus, pratiques et mesures conçues pour protéger les systèmes informatiques, les réseaux, les données et les infrastructures numériques contre les menaces, les attaques et les vulnérabilités. Son objectif principal est d'assurer la confidentialité, l'intégrité et la disponibilité des informations numériques, ainsi que la protection des utilisateurs contre les activités malveillantes en ligne.

# LE MACHINE LEARNING

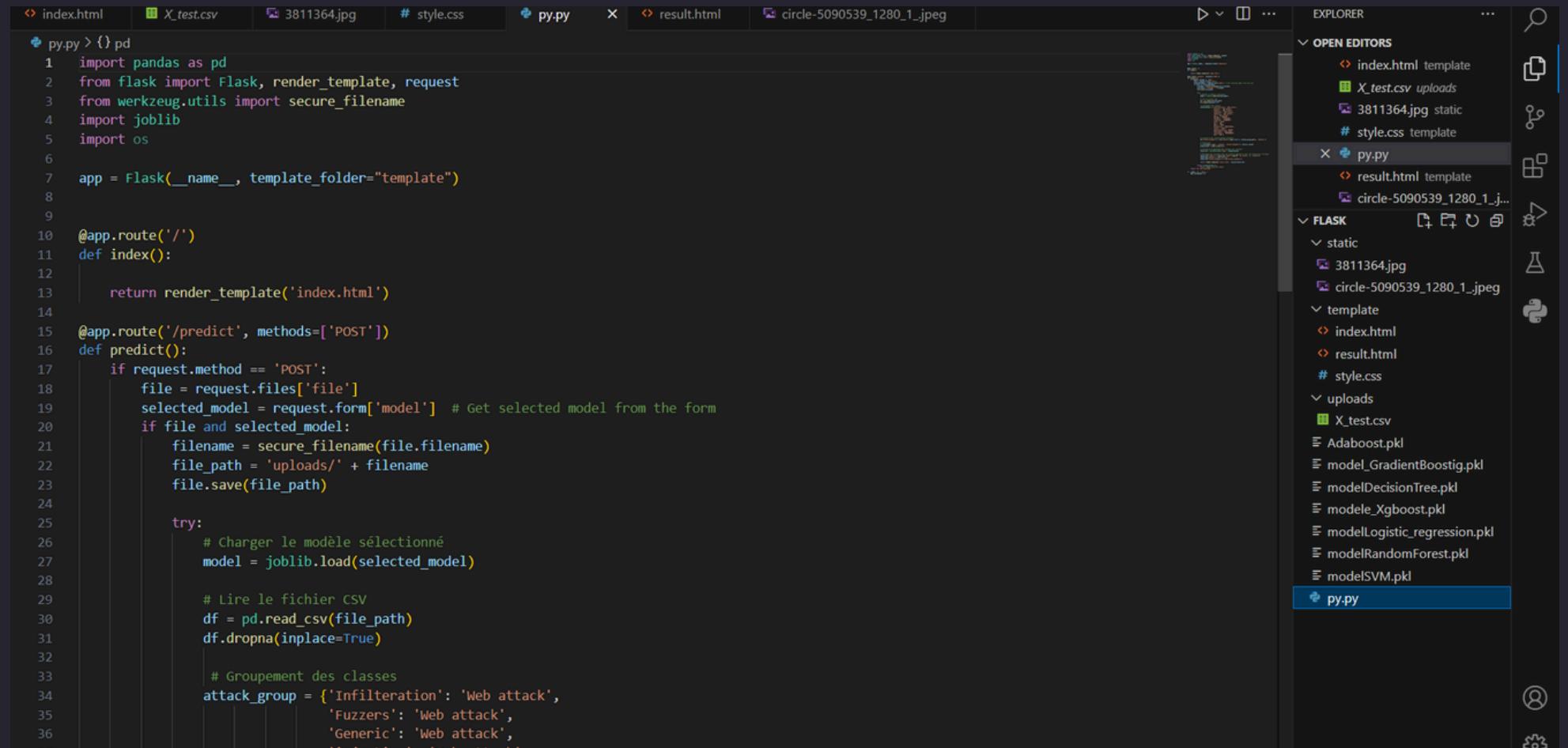
Le machine learning, ou apprentissage automatique en français, est un sous-domaine de l'intelligence artificielle (IA) qui se concentre sur le développement de techniques permettant aux ordinateurs d'apprendre à partir de données et d'améliorer leurs performances sans être explicitement programmés pour chaque tâche spécifique. En d'autres termes, il permet aux machines d'acquérir des connaissances et des compétences à partir de l'expérience.

# DÉTECTION DES INTRUSIONS DANS LE DOMAINE DE LA CYBERSÉCURITÉ APPROCHE AVEC MACHINE LEARNING

La détection des intrusions dans le domaine de la cybersécurité est un processus essentiel visant à identifier les activités malveillantes ou suspectes sur un réseau ou un système informatique. Traditionnellement, cette détection était basée sur des règles prédéfinies et des signatures connues de logiciels malveillants ou d'activités suspectes. Cependant, avec l'évolution des cybermenaces et la sophistication des attaques, ces approches sont devenues moins efficaces.

L'approche avec le machine learning, ou apprentissage automatique, offre une solution plus dynamique et adaptative à ce problème. Plutôt que de se baser uniquement sur des règles fixes, les techniques de machine learning permettent aux systèmes de sécurité de s'entraîner sur de grands ensembles de données pour reconnaître les schémas et les comportements associés à des activités malveillantes.

- Ce code est une application Flask qui permet de charger un modèle pré-entraîné, de prédire des étiquettes sur des données téléchargées par l'utilisateur via un formulaire web, et d'afficher les résultats.

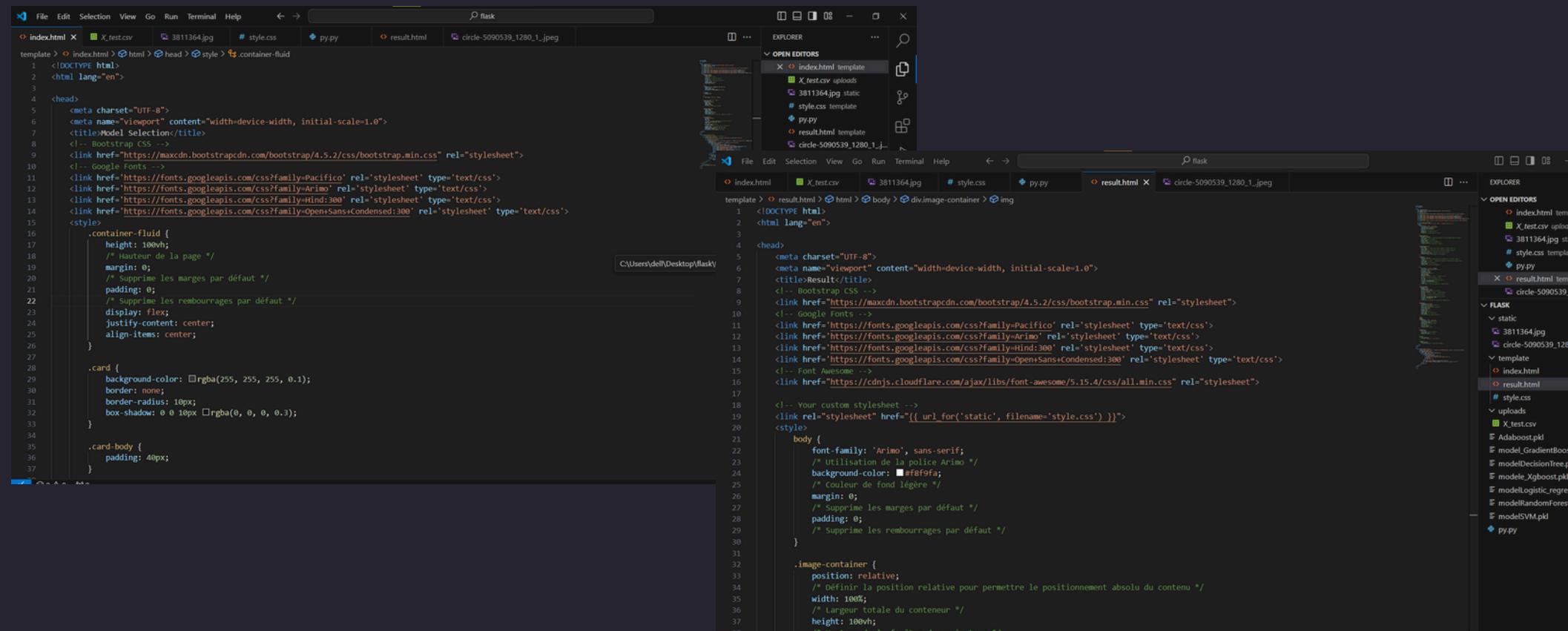


```

1 import pandas as pd
2 from flask import Flask, render_template, request
3 from werkzeug.utils import secure_filename
4 import joblib
5 import os
6
7 app = Flask(__name__, template_folder="template")
8
9
10 @app.route('/')
11 def index():
12
13     return render_template('index.html')
14
15 @app.route('/predict', methods=['POST'])
16 def predict():
17
18     if request.method == 'POST':
19         file = request.files['file']
20         selected_model = request.form['model'] # Get selected model from the form
21
22         if file and selected_model:
23             filename = secure_filename(file.filename)
24             file_path = 'uploads/' + filename
25             file.save(file_path)
26
27             try:
28                 # Charger le modèle sélectionné
29                 model = joblib.load(selected_model)
30
31                 # Lire le fichier CSV
32                 df = pd.read_csv(file_path)
33                 df.dropna(inplace=True)
34
35                 # Groupement des classes
36                 attack_group = {'Infiltration': 'Web attack',
37                                 'Fuzzers': 'Web attack',
38                                 'Generic': 'Web attack',
39                                 ...
40
41             except Exception as e:
42                 return f"Error: {e}"
43
44     return render_template('result.html', prediction=prediction)
45
46
47 if __name__ == '__main__':
48     app.run()

```

## Et voici les codes HTML



The image shows two side-by-side browser windows. The left window displays the 'index.html' template, which contains a form with fields for file upload and model selection. The right window displays the 'result.html' template, which shows the predicted result ('Web attack') based on the input provided.

```

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Model Selection</title>
</head>
<body>
    <h1>Model Selection</h1>
    <form method="post" enctype="multipart/form-data">
        <label>Select Model:</label>
        <select name="model">
            <option value="Adaboost.pkl">Adaboost</option>
            <option value="model_GradientBoosting.pkl">Gradient Boosting</option>
            <option value="modelDecisionTree.pkl">Decision Tree</option>
            <option value="modele_Xgboost.pkl">Xgboost</option>
            <option value="modelLogistic_regression.pkl">Logistic Regression</option>
            <option value="modelRandomForest.pkl">Random Forest</option>
            <option value="modelSVM.pkl">SVM</option>
        </select>
        <br>
        <input type="file" name="file" accept="image/*">
        <br>
        <input type="submit" value="Predict" />
    </form>
</body>

```

```

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Result</title>
</head>
<body>
    <h1>Result</h1>
    <p>The image was classified as: <strong>Web attack</strong></p>
</body>

```

## Une Explication de chaque partie du code :

### 1. Import des modules :

- **pandas**: Pour la manipulation des données.
- **Flask**: Pour créer l'application web.
- **render\_template, request**: Pour rendre les modèles HTML et gérer les requêtes HTTP.
- **secure\_filename**: Pour sécuriser les noms de fichiers lorsqu'ils sont téléchargés.
- **joblib**: Pour charger le modèle d'apprentissage automatique.
- **os**: Pour les opérations sur les systèmes d'exploitation.

### 2. Initialisation de l'application Flask :

- **app = Flask(\_\_name\_\_, template\_folder="template")**: Crée une instance de l'application Flask, en spécifiant le dossier où se trouvent les modèles HTML.

### 3. Route racine ('/') :

- Cette route renvoie le modèle HTML 'index.html' à l'utilisateur lorsqu'il accède à la page d'accueil de l'application.

### 4. Fonction de prédiction ('/predict') :

- Cette route est invoquée lorsque l'utilisateur soumet le formulaire de téléchargement de fichier.
- La fonction vérifie si la méthode de requête est POST.
- Elle récupère le fichier téléchargé et le modèle sélectionné à partir du formulaire.
- Elle charge le modèle sélectionné à l'aide de **joblib.load()**.
- Les données du fichier CSV téléchargé sont lues dans un DataFrame pandas.
- Une colonne 'Attack\_Category' est ajoutée au DataFrame en fonction de la correspondance de la colonne 'Attack' avec un dictionnaire **attack\_group**.
- Les prédictions sont effectuées sur les données en utilisant le modèle chargé.
- Les résultats sont stockés dans un DataFrame.
- Les étiquettes des résultats sont mappées en utilisant un dictionnaire pour représenter les prédictions comme "Attaque" ou "Pas d'attaque".
- Les résultats sont rendus dans le modèle HTML 'result.html'.

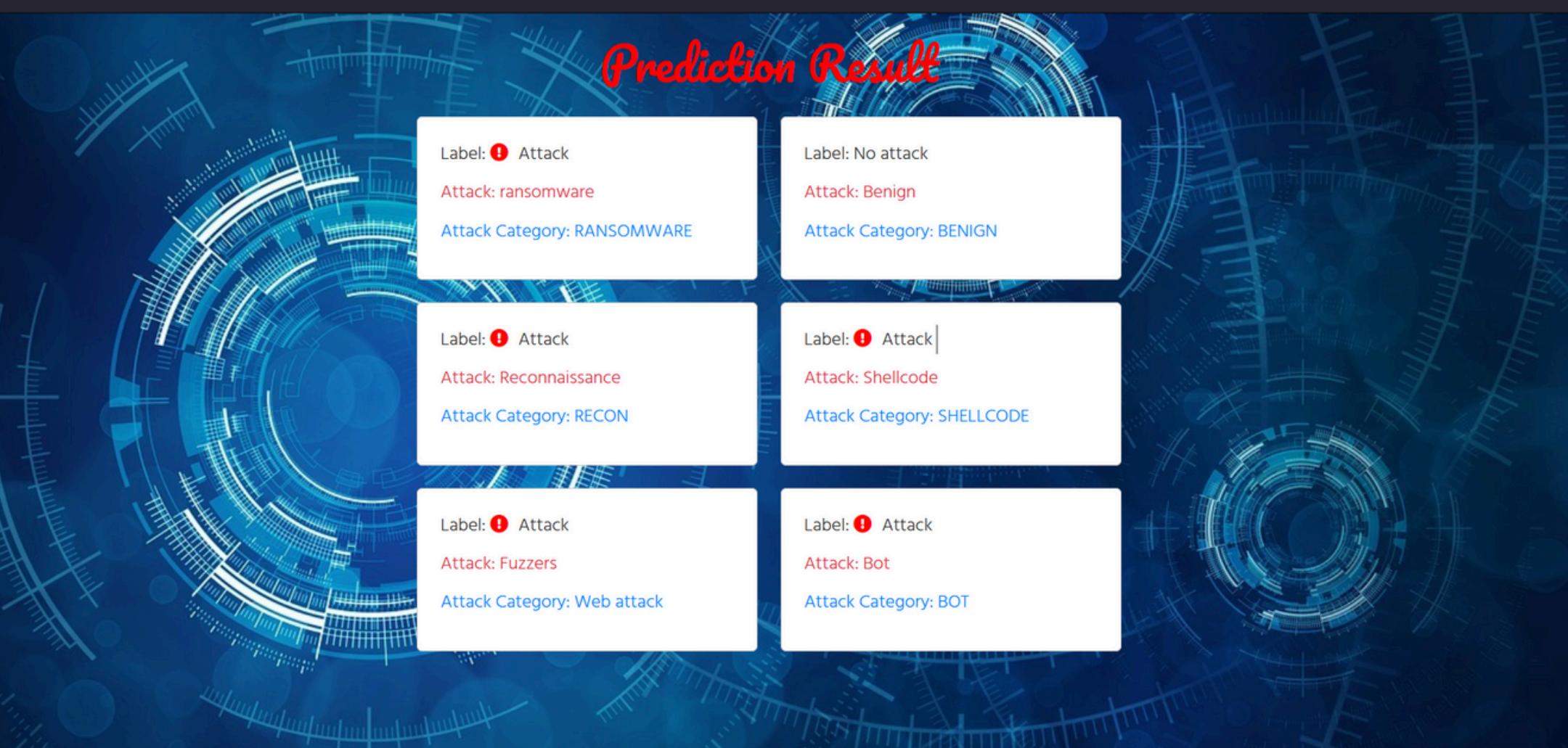
### 5. Exécution de l'application :

- **if \_\_name\_\_ == '\_\_main\_\_': app.run(debug=True)**: Cette ligne garantit que l'application est exécutée uniquement si le script est exécuté directement (pas importé comme module) et active le mode de débogage.

**interface utilisateur** intuitive pour l'application Flask de **prédiction de détection d'intrusion réseau**. Il permet aux utilisateurs de sélectionner un modèle de prédiction parmi plusieurs options, de télécharger un fichier de données à partir de leur appareil, puis de soumettre ces données à l'application pour obtenir des prédictions. L'arrière-plan dynamique et l'utilisation de Bootstrap garantissent une expérience utilisateur attrayante et réactive, améliorant ainsi l'accessibilité et la convivialité de l'application.



**une interface utilisateur** pour afficher les prédictions générées par le modèle. Il présente les résultats de manière claire et structurée, incluant des informations telles que les étiquettes prédites, le type d'attaque et la catégorie d'attaque. En intégrant harmonieusement avec Flask, ce fichier HTML offre une expérience utilisateur améliorée grâce à une présentation visuellement attrayante et une communication fluide entre l'application et l'interface utilisateur.



# Conclusion

L'application Flask de détection d'intrusion réseau offre un outil précieux pour analyser et prévoir les attaques potentielles sur un réseau informatique. En permettant aux utilisateurs de sélectionner différents modèles pré-entraînés et de télécharger leurs propres données pour la prédiction, l'application offre une flexibilité et une adaptabilité accrues. La présentation claire et structurée des résultats, avec des détails sur les types d'attaques prédites, facilite la compréhension et l'interprétation des informations. Cette application fournit ainsi une ressource efficace pour renforcer la sécurité des réseaux et protéger les systèmes informatiques contre les menaces potentielles.