

Rapport de conception de l'application d'Analyse des Performances des Équipes de Hockey

Table des matières

1. Introduction	2
2. Extraction et Préparation des Données.....	2
4. Visualisation des Données.....	4
5. Intégration Web et Interaction	7
6. Conclusion	7
7. Annexes	8

1. Introduction

Ce projet a pour objectif d'analyser les performances des équipes de hockey en utilisant le web scraping pour extraire des données, puis en les explorant avec des outils statistiques et de visualisation.

L'application web interactive développée permet aux utilisateurs d'obtenir et d'analyser ces performances.

2. Extraction et Préparation des Données

L'extraction des données a été réalisée via web scraping en utilisant la bibliothèque BeautifulSoup et requests. Les informations ont été récupérées depuis le site Scrape This Site et sauvegardées sous format CSV.

Données extraites :

- Nom de l'équipe
- Année
- Victoires
- Défaites
- Défaites en prolongation (OT Losses)
- Pourcentage de victoires (Win %)
- Buts marqués (Goals For - GF)
- Buts encaissés (Goals Against - GA)

The screenshot shows an Excel spreadsheet titled "hockey_stats_new_york_islanders - Excel". The table has columns labeled A through R, with headers including "Nom", "Année", "Victoires", "Défaites", "OT Losses", "Win %", "Goals For", and "Goals Against". The data spans from row 1 to row 22. The "Goals For" column contains values such as 0.312, 0.425, 0.476, 0.429, etc., while the "Goals Against" column contains values like 223, 291, 335, 282, etc. The "Win %" column shows percentages ranging from approximately 26% to 45%. The "Année" column lists years from 1990 to 2011. The "Victoires" and "Défaites" columns show the number of wins and losses respectively. The "OT Losses" column indicates the number of games decided in overtime.

Nom	Année	Victoires	Défaites	OT Losses	Win %	Goals For	Goals Against
New York Isla	1990	25	45		0.312	223	290
New York Isla	1991	34	35		0.425	291	299
New York Isla	1992	40	37		0.476	335	297
New York Isla	1993	36	36		0.429	282	264
New York Isla	1994	15	28		0.312	126	158
New York Isla	1995	22	50		0.268	229	315
New York Isla	1996	29	41		0.354	240	250
New York Isla	1997	30	41		0.366	212	225
New York Isla	1998	24	48		0.293	194	244
New York Isla	1999	24	48	1	0.293	194	275
New York Isla	2000	21	51	3	0.256	185	268
New York Isla	2001	42	28	4	0.512	239	220
New York Isla	2002	35	34	2	0.427	224	231
New York Isla	2003	38	29	4	0.463	237	210
New York Isla	2005	36	40	6	0.439	230	278
New York Isla	2006	40	30	12	0.488	248	240
New York Isla	2007	35	38	9	0.427	194	243
New York Isla	2008	26	47	9	0.317	201	279
New York Isla	2009	34	37	11	0.415	222	264
New York Isla	2010	30	39	13	0.366	229	264
New York Isla	2011	34	37	11	0.415	203	255

Figure 1: Capture d'écran : Tableau des données extraites

3. Analyse Statistique

L'analyse des données a été effectuée avec Pandas et NumPy pour calculer divers indicateurs statistiques tels que :

- Moyenne, médiane, mode, écart-type.
- Corrélation entre victoires et buts marqués.
- Identification des meilleures performances des équipes.

The screenshot shows a web browser window with the title "Hockey Stats". The address bar indicates the page is at "127.0.0.1:5000". The main content displays a table of descriptive statistics for various hockey teams. A green button at the bottom allows users to "Télécharger les Résultats (CSV)". Below the table, a section titled "Statistiques :" is shown.

Statistique	Count	Mean	Min	25%	50%	75%	Max	Std
Année	21	2000.33	1990	1995	2000	2006	2011	6.61
Défaites	21	39.00	28	35	38	45	51	7.15
Goals Against	21	255.67	158	240	264	278	315	35.41
Goals For	21	225.62	126	201	224	239	335	42.65
OT Losses	12	7.08	1	3.75	7.5	11	13	4.23
Victoires	21	30.95	15	25	34	36	42	7.22
Win %	21	0.38	0.256	0.312	0.415	0.429	0.512	0.08

🏆 Meilleure Année :

Année	Win %
2001	0.512

Figure 2: Captures d'écran du tableau de la statistique descriptive

The screenshot shows a web browser window with the title "Hockey Stats". The address bar indicates the page is at "127.0.0.1:5000". The main content displays a table of correlations between different metrics. Below the table, a section titled "Corrélation :" is shown.

Relation	Valeur
Victoires-GF	0.687
Win%-GA	-0.177

📈 Évolution des Performances :

Année	Win %
2001	0.512

Figure 3: Capture d'écran de la corrélation

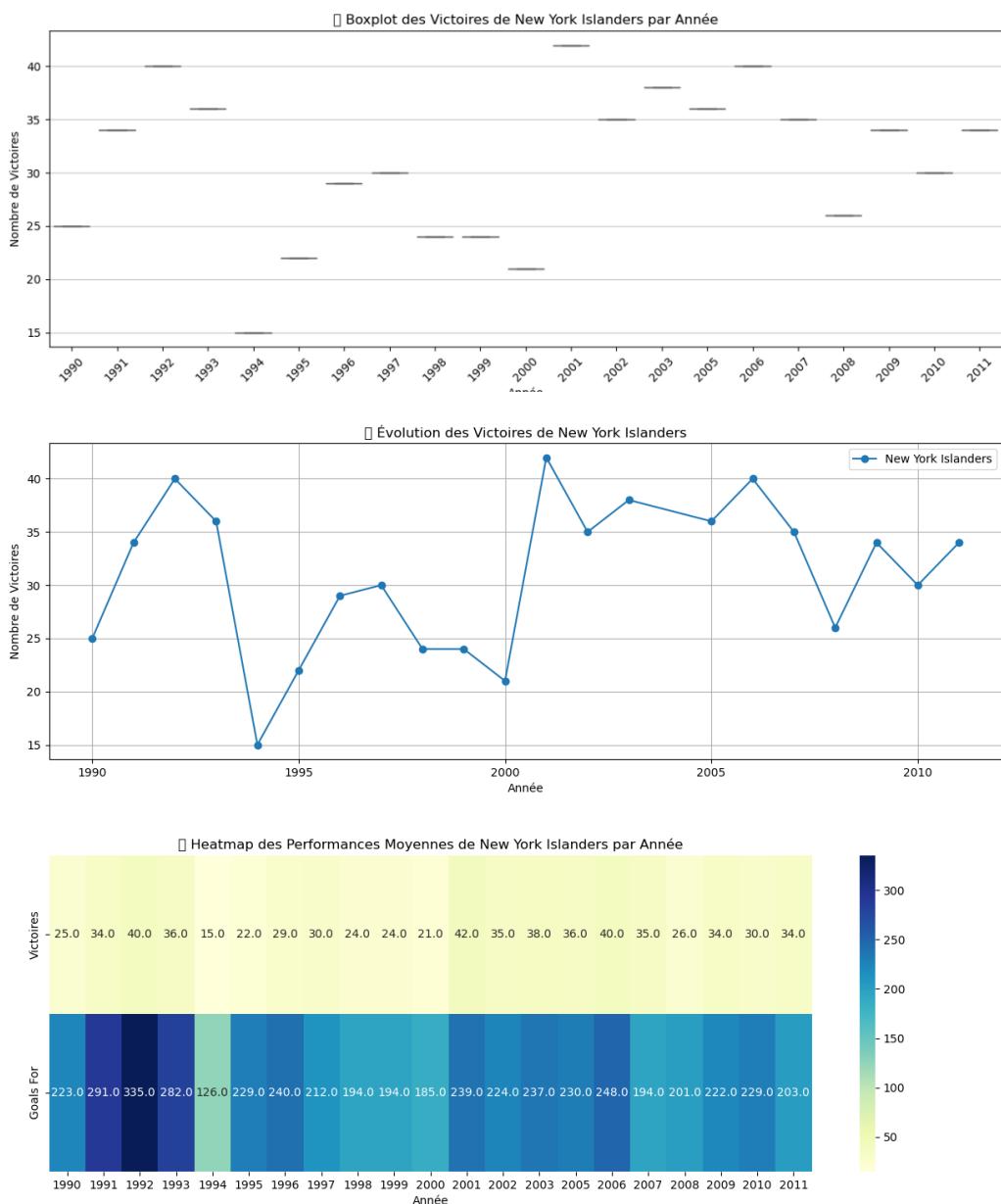
4. Visualisation des Données

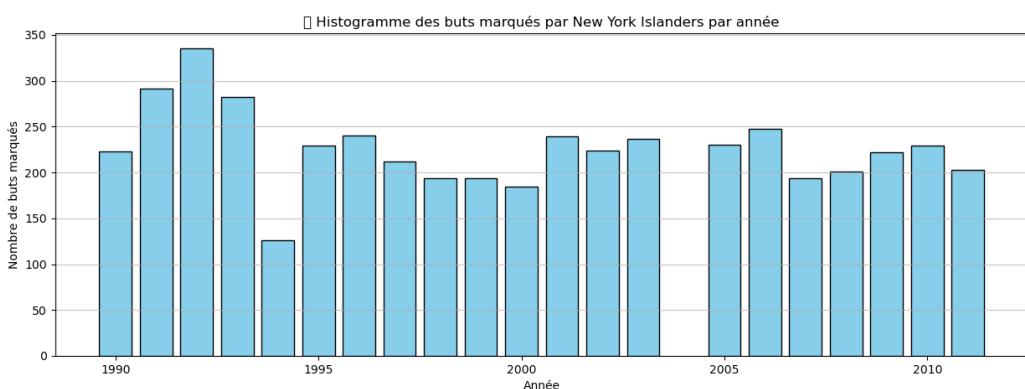
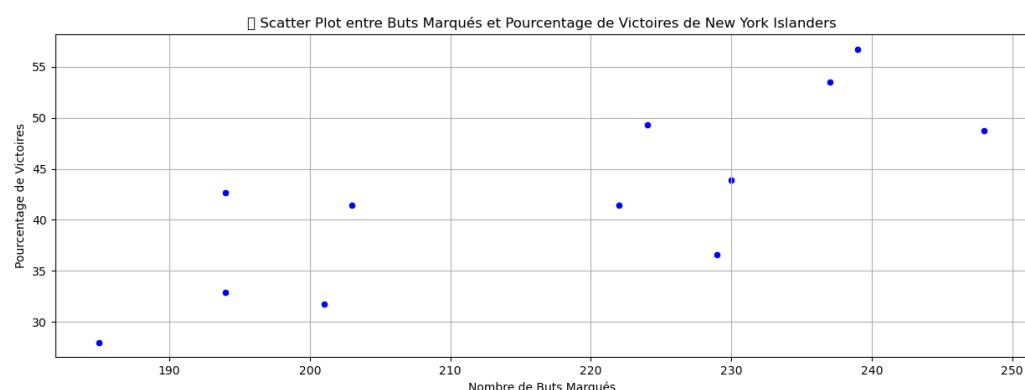
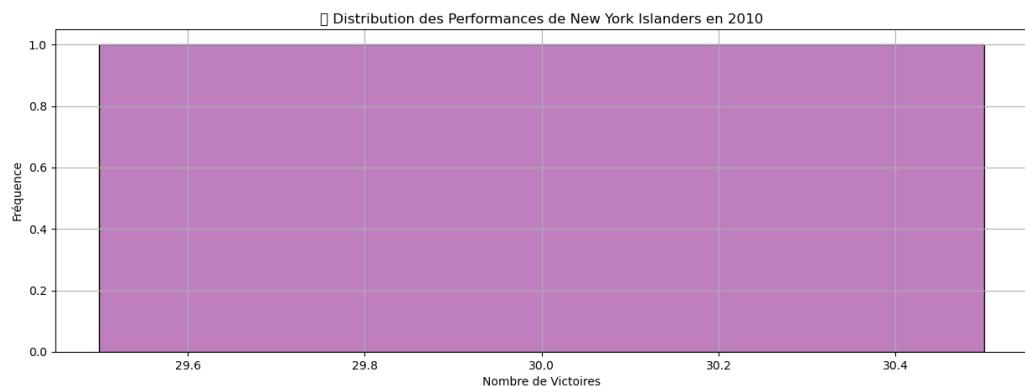
Les données ont été visualisées à l'aide de Matplotlib et Seaborn.

Visualisations générées :

- Courbe d'évolution des victoires par année.
- Histogramme du nombre de buts marqués par année.
- Boxplot des victoires par équipe.
- Heatmap des performances moyennes par année.
- Scatter plot entre le nombre de buts marqués et le pourcentage de victoires.

⚠️ Captures d'écran :





5. Intégration Web et Interaction

L'application web a été développée avec Flask ou FastAPI, permettant aux utilisateurs de rechercher une équipe et d'afficher les résultats analytiques sous forme graphique.

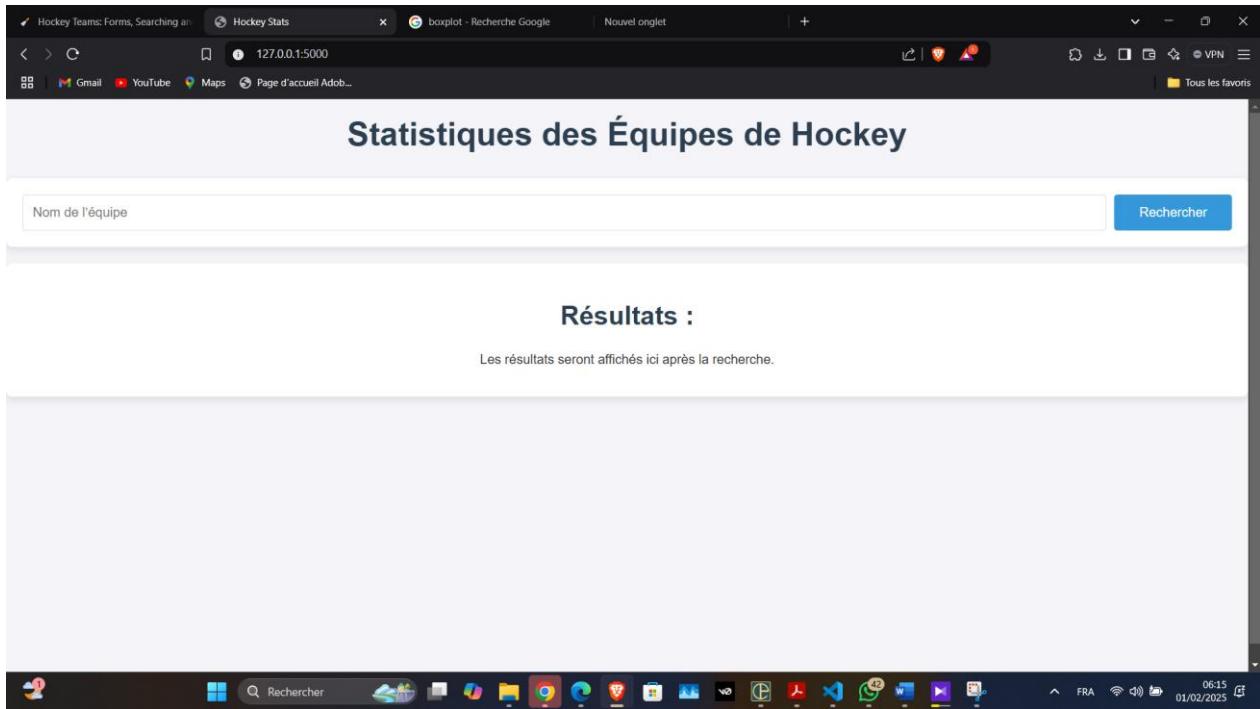


Figure 4: Capture d'écran : Interface utilisateur affichant les résultats

6. Conclusion

Cette analyse a permis d'identifier les tendances de performance des équipes de hockey au fil des saisons. La corrélation entre les buts marqués et les victoires met en évidence l'importance de l'efficacité offensive. L'application développée offre un outil intuitif pour explorer et visualiser ces données de manière interactive.

7. Annexes

The screenshot shows a code editor interface with two tabs open, both titled 'tp_final'. The left tab contains the 'app.py' file, and the right tab contains the 'visualizer.py' file.

app.py (Left Tab):

```
from flask import Flask, render_template, request, jsonify, send_file
from scraper import HockeyScraper
from analyzer import HockeyAnalyzer
from visualizer import HockeyVisualizer
import os
import subprocess

class HockeyApp:
    def __init__(self):
        self.app = Flask(__name__)
        self.scraper = HockeyScraper()
        self.setup_routes()

    # * Appel du script de scraping
    self.run_scraper()

    def run_scraper(self):
        """Exécute scrape_all_pages.py"""
        print("Lancement du script de scraping...")
        subprocess.run(["python", "scrape_all_teams.py"], check=True)

    def setup_routes(self):
        @self.app.route('/')
        def home():
            return render_template('index.html')

        @self.app.route('/search', methods=['POST'])
        def search():
            team_name = request.form.get('team_name')
            # Scraping automatique des données
```

visualizer.py (Right Tab):

```
import seaborn as sns # Importation de seaborn
import os
import random

class HockeyVisualizer:
    """Initialisation avec le fichier CSV spécifique à une équipe."""
    def __init__(self, team_name):
        self.team_name = team_name
        self.file_path = f'static/hockey_stats_{team_name.replace(" ", "_").lower()}.csv'
        self.dataset_df = self.load_data()
        self.graphs_dir = "static/graphs/"
        os.makedirs(self.graphs_dir, exist_ok=True) # Création du dossier pour les images

    def load_data(self):
        """Charge le dataset depuis un fichier CSV."""
        if not os.path.exists(self.file_path):
            print("Fichier introuvable : {}".format(self.file_path))
            return None
        return pd.read_csv(self.file_path)

    def save_plot(self, fig, filename):
        """Sauvegarde un graphique."""
        filepath = os.path.join(self.graphs_dir, filename)
        fig.savefig(filepath)
        plt.close(fig) # Fermer la figure pour éviter les conflits
        return filepath

    def plot_team_wins_over_years(self):
        """Trace et sauvegarde l'évolution des victoires d'une équipe sur plusieurs années."""
        if not self.dataset_df is None or not self.dataset_df.empty:
            print("Aucune donnée disponible pour ({})".format(self.team_name))
            return None
```

```

File Edit Selection View Go Run Terminal Help ← → ⌘ tp_final
EXPLORER ... app.py 1 app.py 2 tp_final 1 visualizer.py 4 scraper.py 1 scrape_all_teams.py 8 README.md hockey_stats_new_york_islanders.csv
static
index.html
test
dataset.csv
LICENSE.chromedriver
scrape_all_teams.py
$SOUPE 4.docx
analyzer.py
app.py
chromedriver.exe
dataset.csv
Doc2.pdf
Groupe 4.docx
Groupe 4.pdf
hockey_stats_new_york_islanders.csv
LICENSE.chromedriver
README.md
scrape_all_teams.py
scraper.py
tpy
THIRD_PARTY_NOTICES.chromedriver
visualizer.py
... OUTLINE TIMELINE
main 53 95 Git Logs BLACKBOX Chat Add Logs CyberCoder Improve Code Share Code Link UTF-8 CRLF Python 3.13.1 Go Live BLACKBOX Open Chat Go Live tabnine basic Prettier
06:17 FRA 01/02/2025

```

```

cmd - python app.py
Lancement du script de scraping...
DevTools listening on ws://127.0.0.1:53723/devtools/browser/214a94dc-e97f-494d-9ce1-417ddef34cccd
2025-02-01 06:14:52,315 - Le fichier 'dataset.csv' existe déjà. Le scraping sera sauté.
2025-02-01 06:14:54,450 - Navigateur fermé.
* Serving Flask app 'app'
* Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on http://127.0.0.1:5000
Press CTRL+C to quit
* Restarting with watchdog (windowsapi)
Lancement du script de scraping...
DevTools listening on ws://127.0.0.1:53744/devtools/browser/ac4a0e73-b3d3-451e-ad0e-873dce6eb6cd
2025-02-01 06:15:04,072 - Le fichier 'dataset.csv' existe déjà. Le scraping sera sauté.
2025-02-01 06:15:06,165 - Navigateur fermé.
* Debugger is active!
* Debugger PIN: 124-353-261
127.0.0.1 - - [01/Feb/2025 06:15:14] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [01/Feb/2025 06:15:14] "GET /static/style.css HTTP/1.1" 304 -
127.0.0.1 - - [01/Feb/2025 06:15:14] "GET /static/script.js HTTP/1.1" 304 -

```