



**UNIVERSITE IBN ZOHR ECOLE SUPERIEURE DE  
L'INTELLIGENCE ARTIFICIELLE ET SCIENCE DE  
DONNEES – TAROUDANT**

**Rapport de mini-projet :**

**Thème :**

---

**Graph Maker : Application web pour l'analyse des fichiers Csv et Excel  
dédiée au secteur de population**

---

**Filière :**

**Management Et Gouvernance Des Systèmes D'information**

**Réalisé par :**

Hajar Lamghari

Yassine Ait Bouhou

Ilyas Hrizi

Yassine Tabakna

**Encadré par :**

Prof Loubna Karbil.

**Année Universitaire : 2024 – 2025**

# TABLE DES MATIERES

<b>INTRODUCTION GENERALE .....</b>	<b>3</b>
Chapitre 1 :_Etude Fonctionnelle et Conceptuelle.....	6
Introduction .....	6
Chapitre 2 :_Les outils, technologie de développement et les interfaces graphiques .....	10
Introduction .....	10
1.    Les outils et technologies de développement.....	10
1.1.  Python .....	10
1.2.  Html .....	10
1.3.  Css.....	10
2.    Outils et technologies de base de données :.....	11
2.1  Star UML .....	11
2.2  Firebase Firestore.....	11
3.    Framework et bibliothèques:.....	11
4.    Fonctions principales de code source : .....	11
5.    Interfaces Graphiques : .....	17
<b>5.1 Interface Authentification : .....</b>	<b>17</b>
<b>5.2 Interface Home : .....</b>	<b>18</b>
<b>5.3 Interface Clean Data : .....</b>	<b>18</b>
<b>5.4 Interface Analyse Data : .....</b>	<b>20</b>
<b>CONCLUSION.....</b>	<b>22</b>

## INTRODUCTION GENERALE

Avec l'émergence de la science des données et de la transformation numérique, la gestion et l'analyse des données sont devenues essentielles dans de nombreux domaines. Parmi les formats les plus couramment utilisés pour le stockage et le partage de données, les fichiers CSV (Comma-Separated Values) et Excel occupent une place prédominante. Cependant, l'analyse de ces fichiers peut s'avérer complexe, surtout pour les utilisateurs qui ne disposent pas des outils ou des compétences nécessaires.

L'analyse et la visualisation des données représentent des étapes cruciales pour extraire des informations utiles et présenter des résultats compréhensibles. Pourtant, l'absence d'outils conviviaux et accessibles limite souvent l'efficacité de ces tâches, obligeant les utilisateurs à se tourner vers des solutions à la fois complexes et coûteuses.

## **Contexte**

Dans un environnement où les volumes de données augmentent de manière exponentielle, il est indispensable de disposer d'outils intuitifs pour exploiter les fichiers CSV et Excel et en tirer des conclusions pertinentes. Étant donné que les données brutes sont souvent bruitées ou incomplètes, il est également crucial d'intégrer des fonctions de nettoyage et de préparation des données.

Dans ce projet, intitulé « Réalisation d'une application Web pour l'analyse et la visualisation de données CSV et Excel », nous cherchons à développer une solution permettant à des utilisateurs, même non techniques, de charger, traiter et visualiser des données de manière interactive. Ce projet vise à exploiter des bibliothèques modernes de Python telles que Pandas pour le traitement des données et Plotly ou Dash pour les visualisations interactives.

## **Problématique**

Les utilisateurs font souvent face à plusieurs défis lorsqu'ils tentent d'analyser des données stockées dans des fichiers CSV ou Excel :

- La difficulté à manipuler et nettoyer les données sans connaissances techniques avancées.
- L'absence d'une interface intuitive permettant d'explorer les données sans recourir à des lignes de code.
- La nécessité de produire rapidement des visualisations interactives pour interpréter les données.

Ces limitations créent une barrière pour les professionnels souhaitant exploiter leurs données efficacement. Par conséquent, une solution simple, adaptée à tous les niveaux de compétence et permettant d'automatiser ces tâches est essentielle.

## **Objectifs**

Dans le cadre de ce projet, notre objectif est de réaliser une application Web conviviale pour l'analyse et la visualisation des données issues de fichiers CSV et Excel concernant la population. Les principaux objectifs incluent :

- Faciliter l'authentification : Permettre à l'utilisateur de se connecter de manière sécurisée pour garantir la confidentialité des données.
- Charger et nettoyer les fichiers CSV et Excel : Fournir des outils pour gérer les valeurs manquantes, formater les colonnes et détecter les erreurs dans les données.
- Visualiser les données de manière interactive: Créer des graphes dynamiques et personnalisables, tels que des histogrammes, des courbes, des diagrammes en barres, etc.
- Améliorer l'expérience utilisateur : Proposer une interface simple et ergonomique adaptée à tous les profils d'utilisateurs.

- Exporter et partager les résultats : Offrir la possibilité de sauvegarder les visualisations au format image ou de les partager directement via l'application.

### **Organisation du rapport :**

Le contenu de ce rapport est organisé comme suit :

- Dans Le **premier chapitre** présente l'étude fonctionnelle (exigences fonctionnelles et non fonctionnelles) ainsi que l'étude conceptuelle, comprenant les diagrammes de conception tels que les diagrammes de cas d'utilisation, de séquence et de classe.
- Le **deuxième chapitre** décrit les outils et technologies utilisés pour le développement de l'application, notamment les bibliothèques Python pour le traitement des données et la création de visualisations interactives.
- Enfin, nous concluons avec une **conclusion** résumant les résultats obtenus et les perspectives futures pour améliorer l'application.

# Chapitre 1 :

## Etude Fonctionnelle et Conceptuelle

---

### Introduction

Une étude fonctionnelle et conceptuelle est une activité critique dans le développement de tout logiciel ou système. Cela implique de rassembler et d'analyser les exigences pour garantir que le système ou le logiciel répond aux besoins de ses utilisateurs et parties prenantes.

### 1-Exigences fonctionnelles :

Les exigences fonctionnelles décrivent les fonctionnalités que l'application doit offrir à ses utilisateurs :

#### **Authentification et gestion des utilisateurs :**

- Permettre aux utilisateurs de se connecter à l'application de manière sécurisée via un système de connexion avec identifiant et mot de passe.
- Assurer la gestion des sessions utilisateurs.

#### **Chargement des fichiers CSV et Excel :**

- Charger facilement des fichiers CSV et Excel depuis leur appareil.

#### **Affichage des données dans un tableau interactif**

- Les colonnes et les lignes s'ajustent automatiquement et prennent en charge défilement horizontal et vertical.
- Limitation à 5 lignes affichées par page.

#### **Graphiques interactifs**

- L'utilisateur peut générer des graphiques basés sur les données importées.
- Types de graphiques disponibles :
  - Scatter Plot (nuage de points) : pour visualiser des relations entre deux colonnes.
  - Bar Chart (graphique à barres) : pour comparer des valeurs.
- Les utilisateurs sélectionnent les axes X et Y via des menus déroulants.

#### **Adaptabilité et personnalisation**

- Une mise en page adaptative grâce à des styles CSS intégrés.

- Les utilisateurs peuvent personnaliser la configuration des graphes via un menu déroulant pour changer le type de graphique.

### **Fonctionnalités avancées de nettoyage**

- Implémentez des outils pour supprimer les valeurs nulles, remplacer des valeurs, ou détecter des anomalies.

### **Analyse des données :**

- Fournir des résumés statistiques de base (moyenne, médiane, écart-type, etc.).

Types d'analyses :

- **statistiques descriptives** : Calcule des statistiques descriptives comme la moyenne, l'écart-type, la médiane, les valeurs minimales/maximales, les quartiles, la somme et le nombre de valeurs pour la colonne sélectionnée. Ces statistiques sont affichées sous forme de tableau.
- **Analyse démographique**: Calcule la fréquence d'apparition des valeurs uniques dans la colonne sélectionnée (par exemple, fréquence des valeurs catégorielles)
- **Visualisation graphique d' Analyse démographique**: Crée un histogramme des valeurs de la colonne sélectionnée et l'affiche sous forme de graphique Plotly.
- **Analyse de corrélation**: Calcule et affiche une matrice de corrélation pour les colonnes numériques du jeu de données. Une visualisation sous forme de carte thermique (heatmap) de cette matrice de corrélation est également générée.

### **Visualisation des données :**

Créer des visualisations interactives incluant :

- Histogrammes
- Graphes en courbes
- Diagrammes à barres
- Nuages de points
- Graphiques circulaires
- Permettre la personnalisation des graphes (couleurs, titres, axes, etc.).

### **Exportation et partage des résultats :**

- Exporter les visualisations au format image (.png, .jpg) ou PDF.

## **2-Exigences non fonctionnelles :**

Les exigences non fonctionnelles, aussi connues sous le nom d'exigences techniques, définissent les normes de qualité et les limitations techniques auxquelles notre projet de moteur de recherche doit se conformer.

### **Performance :**

**Réactivité de l'application :** L'application doit répondre rapidement aux interactions des utilisateurs, telles que le téléchargement des fichiers CSV/Excel, le traitement des données et la génération de graphiques. L'objectif est d'assurer des temps de réponse inférieurs à 2 secondes pour les actions simples et inférieurs à 5 secondes pour les opérations complexes (par exemple, l'analyse de corrélation sur des ensembles de données volumineux).

**Traitement des données volumineuses :**

### **Fiabilité :**

**Stabilité et tolérance aux pannes :** L'application doit être conçue de manière à minimiser les risques de panne, avec une gestion d'erreurs robuste pour détecter et résoudre les problèmes sans interruption majeure du service. Par exemple, si un fichier est mal formaté ou si une analyse échoue, l'application doit afficher un message d'erreur clair sans planter.

### **Maintenabilité :**

**Code modulaire et bien structuré :** Le code de l'application doit être organisé en modules clairement définis, permettant de faciliter les mises à jour et l'ajout de nouvelles fonctionnalités (par exemple, de nouveaux types d'analyse ou de nouveaux formats de fichiers). L'utilisation de pratiques comme la séparation des responsabilités (par exemple, traitement des fichiers, génération des graphiques, gestion des erreurs) aide à maintenir un code propre et évolutif.

### **Sécurité :**

**Authentification et autorisation sécurisée :** L'application doit disposer d'un système d'authentification sécurisé pour permettre aux utilisateurs de se connecter et d'accéder à leurs données de manière protégée. Il est également essentiel de gérer les sessions utilisateur de manière sécurisée pour éviter les failles de sécurité.

### **Interface utilisateur et expérience (UX/UI) :**

**Convivialité :** Interfaces intuitives et faciles à utiliser, même pour des utilisateurs non expérimentés.

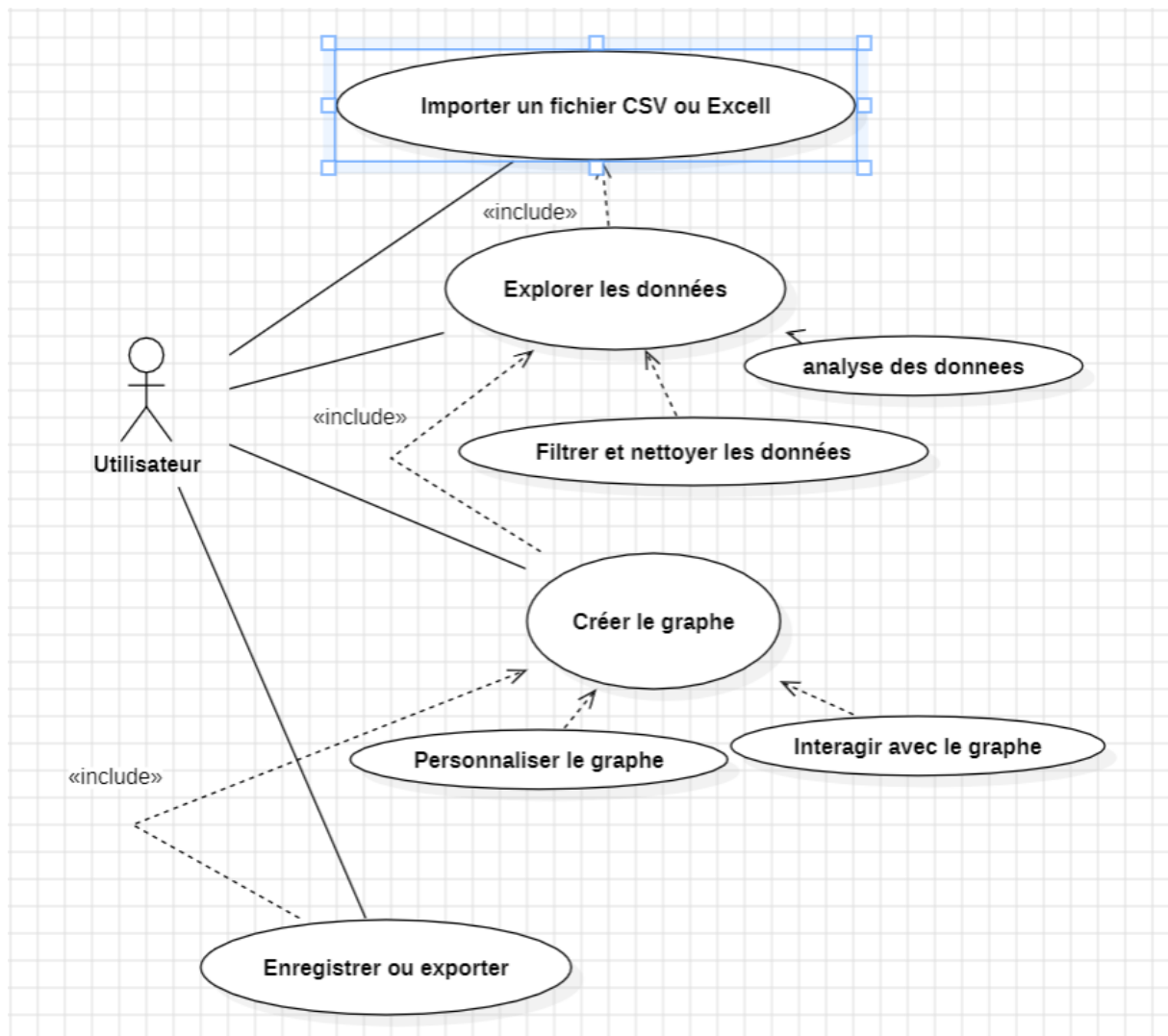
**Réactivité :** L'application est responsive, c'est-à-dire s'adapte aux différentes tailles d'écrans (ordinateur de bureau, tablette, téléphone mobile).

**Design attractif et moderne :** Le design esthétique, avec une navigation fluide et des composants bien organisés pour améliorer l'expérience utilisateur.

## **3-Conception :**



**Diagramme de cas d'utilisation :** Les diagrammes de cas d'utilisation modélisent la manière dont les utilisateurs, représentés sous forme de figurines appelées « acteurs », interagissent avec le système. Ce type de diagramme UML est une vue d'ensemble des relations entre les acteurs et les systèmes, ce qui en fait un excellent outil pour présenter notre application d'analyse.



Ce diagramme de cas d'utilisation montre que l'utilisateur peut importer des données, les nettoyer, les analyser, les visualiser sous forme de graphique, personnaliser le graphique, l'enregistrer, et enregistrer les résultats d'analyses.

# Chapitre 2 :

## Les outils, technologie de développement et les interfaces graphiques

---

### Introduction

Le développement et l'intégration d'outils et de technologies ont considérablement évolué au fil des ans, avec l'introduction de divers langages de programmation, frameworks et bibliothèques. L'utilisation de ces outils et technologies a contribué à l'amélioration des performances et de l'efficacité de notre application Graph Maker .

### 1. Les outils et technologies de développement

#### 1.1. Python

Python sert principalement à la création de scripts et à l'automatisation. En effet, ce langage permet d'automatiser les interactions avec les navigateurs web ou les interfaces graphiques d'applications.



Cependant, le scripting et l'automatisation sont loin d'être les seules utilisations de ce langage. Il permet également la programmation d'applications, la création de services web ou d'API REST, ou encore la méta programmation et la génération de code.

Par ailleurs, ce langage peut également être utilisé dans le domaine de la science des données et du Machine Learning. Avec l'essor de l'analyse des données dans toutes les industries, ce domaine devient l'un de ses principaux cas d'utilisation.

#### 1.2. Html

HTML5(Hyper Text Mark up Langage 5) est la nouvelle révision majeure d'HTML, format de données conçues pour représenter les pages web. Cette version est devenue stable depuis 2012.



Le langage comprend une couche application avec de nombreuses API. HTML est un langage de balisage utilisé pour la création de pages web et d'autres documents qui sont destinés à être affichés sur le World Wide Web. Il est utilisé pour structurer le contenu d'une page web en utilisant des balises qui indiquent au navigateur web comment afficher le contenu.

#### 1.3. Css

CSS (Cascading Style Sheets) est un langage de feuille de style utilisé pour décrire la présentation visuelle des pages web écrites en HTML ou XHTML. CSS permet de définir des styles pour différents éléments HTML tels que les polices de caractères, les couleurs, les marges, les bordures, les images de fond et bien d'autres encore



## 2. Outils et technologies de base de données :

Dans notre mini projet , la base de données est utilisée principalement pour stocker les informations des utilisateurs et permettre leur connexion à l'application via un système d'authentification sécurisé. Voici les principales technologies et pratiques utilisées :

### 2.1 Star UML



StarUML est un logiciel de modélisation UML open source et gratuit. Il fournit une interface graphique utilisateur pour créer des diagrammes UML pour diverses applications logicielles. Le nom "StarUML" signifie "Star Unified Modeling Language", ce qui reflète l'objectif du logiciel de fournir un outil complet de modélisation UML pour les développeurs de logiciels.

### 2.2 Firebase Firestore



La base de données utilisée est Firebase Firestore, une base de données NoSQL en temps réel qui permet de stocker des informations structurées sous forme de collections et de documents. Dans ce projet, une collection appelée "users" est utilisée pour stocker les informations personnelles de chaque utilisateur, telles que leur nom, email et mot de passe.

## 3. Framework et bibliothèques:

**Plotly Dash** : Framework principal pour construire une interface utilisateur interactive et moderne.

**Pandas** : Pour le nettoyage, la manipulation et l'analyse des données issues des fichiers CSV et Excel.

**Matplotlib et Plotly Graph Objects** : Pour la création de graphiques interactifs et personnalisables.

**Dash Bootstrap Components** : Pour améliorer le design de l'interface utilisateur en utilisant les styles de Bootstrap.

**Scikit-learn** : Utilisé pour certaines analyses avancées, comme la classification ou les prédictions si nécessaire.

**Base64** : est une méthode de codage qui permet de convertir des données binaires (comme des images, des fichiers, etc.) en une chaîne de caractères ASCII. Cela est utile pour transmettre des données binaires via des formats textuels, comme dans les emails ou sur le web.

## 4. Fonctions principales de code source :

### Fonction d'inscription d'utilisateurs:

```
def signup_user(first_name, last_name, username, password, email, gender):
    try:
        hashed_password = bcrypt.hashpw(password.encode('utf-8'), bcrypt.gensalt())
        existing_user = users_ref.where('username', '==', username).limit(1).stream()

        for user in existing_user:
            return "Username already exists."

        new_user = {
            'first_name': first_name,
            'last_name': last_name,
            'username': username,
            'password': hashed_password.decode('utf-8'), # Store as string
            'email': email,
            'gender': gender,
        }
        # Add the document to Firestore
        users_ref.add(new_user)
        print(f"Adding new user to firestore: {new_user}")
        return "User successfully created!"

    except Exception as e:
        print(f"Error: {e}")
        return "An error occurred during sign-up."
```

---

- **Hachage du mot de passe :**

Le mot de passe fourni est haché à l'aide de la bibliothèque **bcrypt** pour assurer la sécurité des données sensibles.

- **Vérification de l'existence du nom d'utilisateur :**

La fonction interroge la base Firestore pour vérifier si le nom d'utilisateur (username) existe déjà.

- **Création d'un nouvel utilisateur :**

Si le nom d'utilisateur est unique, la fonction crée un dictionnaire contenant les informations de l'utilisateur.

- **Gestion des erreurs :**

Si une exception survient (par exemple, un problème de connexion à Firestore), la fonction capture l'erreur et retourne un message générique.

### Fonction de Mise a jour d'Analyses :

```

def update_analysis(contents, column, analysis_type, n_clicks):
    if contents is None:
        raise PreventUpdate

    # Gestion des erreurs de fichier
    try:
        df = parse_contents(contents)
        file_size = len(contents.encode('utf-8')) / 1024 # Taille en Ko
        file_status = f"Fichier chargé avec succès : {file_size:.2f} Ko"
    except Exception as e:
        return [], None, None, f"Erreur : {str(e)}", None, None

    # Mettre à jour les colonnes dans le menu déroulant
    columns = df.columns.tolist()
    column_options = [{'label': col, 'value': col} for col in columns]

    if not column:
        return column_options, None, None, None, None, file_status
    download_link = None

    if analysis_type == 'stats':
        # Analyse des statistiques descriptives
        try:
            df[column] = pd.to_numeric(df[column], errors='coerce')
            if df[column].isnull().all():
                return column_options, column, None, "Erreur : La colonne '{column}' contient uniquement des valeurs non numériques ou est vide.", None, file_status

            stats = {
                "Moyenne": df[column].mean(),
                "Écart-type": df[column].std(),
                "Médiane": df[column].median(),
                "Valeur Min": df[column].min(),
                "Valeur Max": df[column].max(),
                "1er Quartile": df[column].quantile(0.25),
                "3e Quartile": df[column].quantile(0.75),
                "Somme": df[column].sum(),
                "Nombre de valeurs": df[column].count()
            }
        except:
            return column_options, column, None, "Erreur : La colonne '{column}' contient uniquement des valeurs non numériques ou est vide.", None, file_status

```

La fonction **`update_analysis`** prend des données (**`contents`**), une colonne spécifique (**`column`**), et un type d'analyse (**`analysis_type`**) et effectue une analyse ou une visualisation spécifique en fonction de ces entrées. Elle retourne des résultats sous forme de tableaux, graphiques, ou liens de téléchargement pour l'utilisateur.

- **Statistiques descriptives ('stats')** : La fonction calcule et affiche des statistiques telles que la moyenne, l'écart-type, la médiane, etc., de la colonne choisie. Elle génère également un lien pour télécharger les résultats sous forme de fichier PDF.
- **Analyse démographique ('demo')** : Cette analyse compte la fréquence d'apparition de chaque valeur unique dans la colonne et l'affiche sous forme de tableau. Un lien pour télécharger les résultats en PDF est aussi généré.
- **Visualisation graphique ('viz')** : Un histogramme est généré pour visualiser la distribution des valeurs dans la colonne sélectionnée.

- **Analyse de corrélation ('corr') :** Si plusieurs colonnes numériques sont disponibles, une matrice de corrélation est calculée et affichée sous forme de tableau et de graphique. Un lien pour télécharger la matrice de corrélation en PDF est également proposé

### Fonction de remplissage des valeurs manquantes:

```
# Fonction pour traiter les colonnes selon les types dominants
def rem_v_manquante(data, colonne): 1 usage
    if colonne in data.columns:
        column_data = data[colonne]
        if column_data.isna().all():
            data[colonne] = 0
        else:
            numeric_values = pd.to_numeric(column_data, errors='coerce') #Convertir les valeurs en numériques
            mean_value = round(numeric_values.mean(), 2) # Calculer la moyenne des valeurs numériques
            if numeric_values.notna().sum() >= (column_data.size - numeric_values.notna().sum()):
                data[colonne] = numeric_values.fillna(mean_value)
            else:
                text_values = column_data[numeric_values.isna()]
                mode_value = text_values.mode()[0] if not text_values.empty else "N/A"
                data[colonne] = column_data.apply(
                    lambda x: mode_value if pd.isna(x) else x if isinstance(x, str) else mode_value
                )
    return data
```

La fonction gère les valeurs manquantes dans une colonne spécifique d'un DataFrame.

- Si la colonne est numérique (float64 ou int64), les valeurs manquantes sont remplacées par la moyenne.
- Si la colonne est catégorielle ou textuelle (object ou category), les valeurs manquantes sont remplacées par la valeur la plus fréquente.
- Retour : Le DataFrame avec les valeurs manquantes traitées.

### Fonction de conversion en Dataframe:

```
def parse_data(contents, filename):
    content_type, content_string = contents.split(',')
    decoded = base64.b64decode(content_string)
    try:
        if 'csv' in filename:
            df = pd.read_csv(io.StringIO(decoded.decode('utf-8')))
        elif 'xls' in filename or 'xlsx' in filename:
            df = pd.read_excel(io.BytesIO(decoded))
        else:
            return html.Div(["Le format du fichier n'est pas supporté."])
        return df
    except Exception as e:
        return html.Div([f"Erreur lors du traitement du fichier : {str(e)}"])
```

- **Décodage du contenu encodé :**

Le contenu encodé en base64 est divisé en deux parties : le type MIME et la chaîne de données encodées. Les données encodées sont ensuite décodées pour obtenir le contenu brut du fichier.

- **Identification du format du fichier.**

- **Gestion des exceptions :**

Si une erreur survient lors de la lecture ou du décodage du fichier, un message d'erreur descriptif est renvoyé.

- **Retour du résultat :**

- Si le fichier est correctement traité, un DataFrame pandas contenant les données est retourné.
- En cas d'erreur ou de format non supporté, un élément HTML contenant un message d'erreur est retourné.

### Fonction de génération de graphe:

```
def update_graph(x_column, y_column, graph_type, contents, filename):
    # Declare global_df as global to share it across callbacks
    global global_df
    global_df = None

    if contents is not None:
        global_df = parse_data(contents, filename) # Parse the uploaded file

    # Ensure data is available and columns are selected
    if global_df is None or x_column is None or y_column is None or global_df.empty:
        return go.Figure()

    # Generate the graph based on the selected type
    if graph_type == 'scatter':
        fig = go.Figure(
            data=[
                go.Scatter(
                    x=global_df[x_column],
                    y=global_df[y_column],
                    mode='markers',
                    marker=dict(size=10, color='blue'),
                )
            ],
            layout=go.Layout(
                title="Scatter Plot",
                xaxis_title=x_column,
                yaxis_title=y_column
            )
        )
    elif graph_type == 'bar':
        fig = go.Figure(
            data=[
                go.Bar(
                    x=global_df[x_column],
                    y=global_df[y_column],
                    marker=dict(color='blue'),
                )
            ],
            layout=go.Layout(
                title="Bar Chart",
                xaxis_title=x_column,
                yaxis_title=y_column
            )
        )
    )
```

```

elif graph_type == 'violin':
    fig = go.Figure(
        data=[
            go.Violin(
                y=global_df[y_column],
                name=y_column,
                box_visible=True,
                meanline_visible=True,
                marker=dict(color='blue'),
            )
        ],
        layout=go.Layout(
            title="Violin Plot",
            yaxis_title=y_column
        )
    )

elif graph_type == 'histogram':
    fig = go.Figure(
        data=[
            go.Histogram(
                x=global_df[x_column],
                marker=dict(color='blue'),
            )
        ],
        layout=go.Layout(
            title="Histogram",
            xaxis_title=x_column,
            yaxis_title="Frequency"
        )
    )

elif graph_type == 'pie':
    fig = go.Figure(
        data=[
            go.Pie(
                labels=global_df[x_column],
                values=global_df[y_column],
                hole=0.3,
            )
        ],
        layout=go.Layout(
            title="Pie Chart",
        )
    )

else:
    fig = go.Figure()

return fig

```

---

- **Validation des entrées :**

La fonction vérifie si les colonnes `x_column` et `y_column` sont spécifiées et si le DataFrame global (`global_df`) n'est pas vide. Si l'une de ces conditions n'est pas remplie, elle retourne une figure Plotly vide.

- **Création d'un graphique selon le type :**

**Scatter Plot (nuage de points) :**

Si le type de graphique est "`scatter`", un nuage de points est créé à l'aide de `go.Scatter`, avec des marqueurs personnalisés (taille, couleur).

- **Bar Chart (graphique à barres) :**

Si le type de graphique est "`bar`", un graphique à barres est créé à l'aide de `go.Bar`, avec des barres colorées en bleu.

- **Gestion des types de graphiques non pris en charge :**

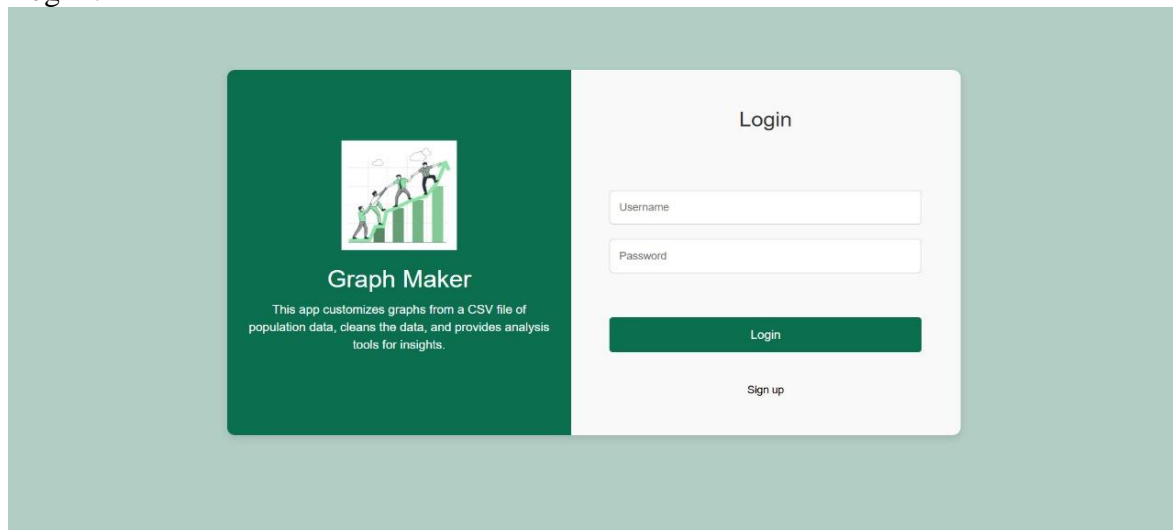
Si le type de graphique spécifié n'est pas reconnu, une figure vide est renvoyée.



## 5. Interfaces Graphiques :

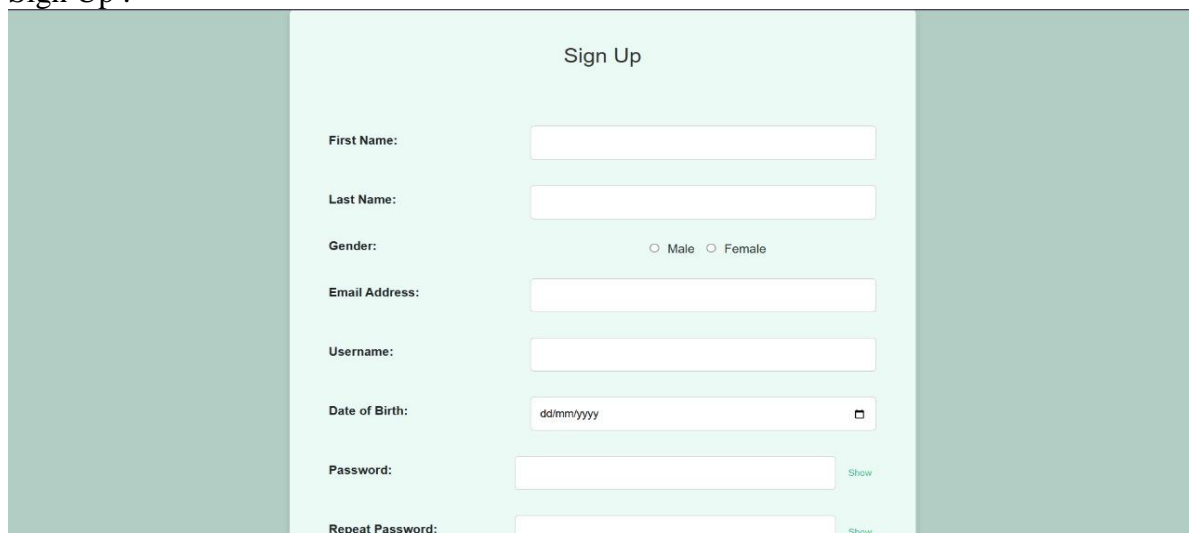
### 5.1 Interface Authentication :

Login :



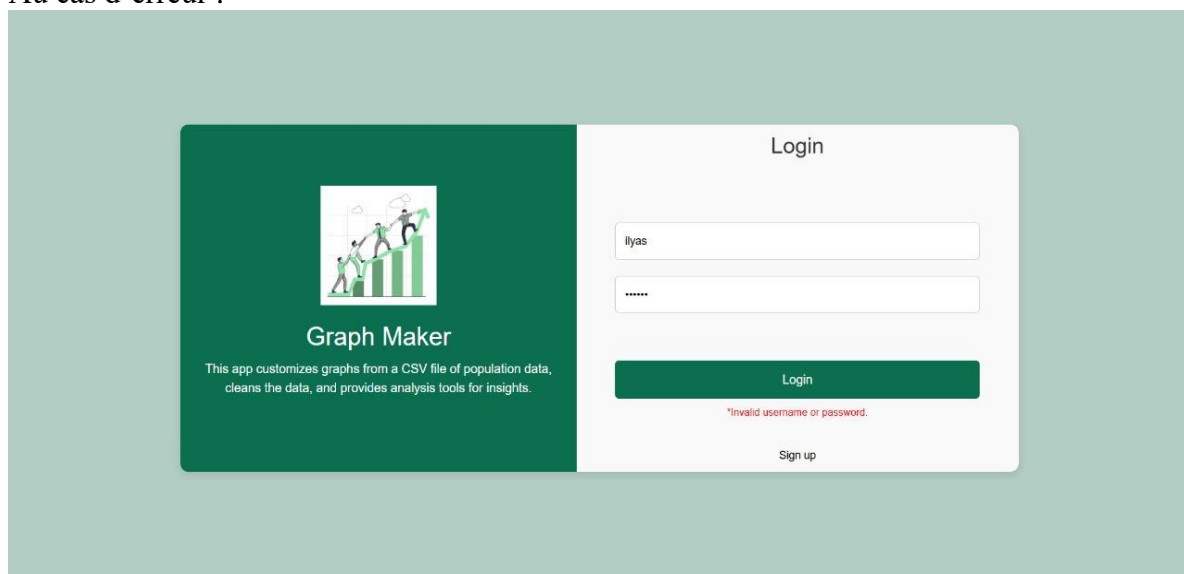
The login interface is displayed on a light green background. On the left, there is a dark green card with a white icon of three people climbing a bar chart. Below the icon, the text reads "Graph Maker" and "This app customizes graphs from a CSV file of population data, cleans the data, and provides analysis tools for insights." On the right, there is a white card with the title "Login". It contains two input fields: "Username" and "Password". Below these fields is a dark green "Login" button. At the bottom of the white card is a "Sign up" link.

Sign Up :



The sign up interface is displayed on a light green background. It features a white card with the title "Sign Up". The form includes several input fields: "First Name:", "Last Name:", "Email Address:", "Username:", and "Date of Birth:" (with a date picker icon). There are also radio buttons for "Gender:" with options "Male" and "Female". Below the "Date of Birth:" field is a "Password:" field with a "Show" link. At the bottom is a "Repeat Password:" field with a "Show" link.

Au cas d'erreur :



The login interface is shown with an error message. The "Username" field contains the text "Ilyas" and the "Password" field contains "\*\*\*\*\*". Below the "Login" button, a red error message reads "Invalid username or password." The "Sign up" link is visible at the bottom.

Si tout est passé bien , l'utilisateur est dans Home directement.

## 5.2 Interface Home :



L'utilisateur maintenant a le choix , soit de générer le graphe qui veut a travers l'importation du fichier csv ou Excel, soit d'Analyser data ou la nettoyer , après la possibilité de sauvegarder le graphe visualisée en format png et finalement la possibilité de quitter l'application en cliquant sur *sign out*.

## Exemple de Visualisation :



## 5.3 Interface Clean Data :

Home

Analyse Data

Clean Data

Save PNG

Upload File

Sign Up / Sign In

# Nettoyage des données

Charger un fichier CSV

Choisir une colonne

Supprimer Duplicats

Remplir Manquants

Corriger Age

Nettoyer Texte

Corriger Dates

Avant Nettoyage :

Home

Analyse Data

Clean Data

Save PNG

Upload File

Sign Up / Sign In

# Nettoyage des données

Charger un fichier CSV

Choisir une colonne

Supprimer Duplicats

Remplir Manquants

Corriger Age

Nettoyer Texte

Corriger Dates

Le fichier a été chargé avec succès.

Employee_ID	Name	Department	Salary	Joining_Date
101	John	HR	5000	2020-01-15
102	Anna	Finance	5500	2019-12-01
103	maria	It	six_thousand	2021-02-30
104	john	HR		15-01-2020
105	ANNA		4800	
101	John	IT	5000	2020-01-15
106	George	Finance	6000	2022-06-15
107	Maria	It	6200	2021/02/15
108	lucas	HR	4900	2019-03-01
109	anna	finance	5500	2021-02-30

Après Nettoyage :

Home

Analyse Data

Clean Data

Save PNG

Upload File

Sign Up / Sign In

# Nettoyage des données

Charger un fichier CSV

Joining\_Date

Supprimer Duplicats

Remplir Manquants

Corriger Age

Nettoyer Texte

Corriger Dates

Les valeurs manquantes dans la colonne 'Joining\_Date' ont été remplies.

Employee_ID	Name	Department	Salary	Joining_Date
101	john	HR	5000	2020-01-15
102	anna	Finance	5500	2019-12-01
103	maria	It	5362.5	2021-02-30
104	john	HR	5362.5	15-01-2020
105	anna	HR	4800	2020-01-15
101	john	IT	5000	2020-01-15
106	george	Finance	6000	2022-06-15
107	maria	It	6200	2021/02/15
108	lucas	HR	4900	2019-03-01
109	anna	finance	5500	2021-02-30

## 5.4 Interface Analyse Data :

Upload File

Sign Up / Sign In

Home

Analyse Data

Clean Data

Save PNG

Analyse des Données de Population

Analysez les colonnes de données facilement.

Glissez et déposez un fichier CSV ou Excell cliquez pour sélectionner un fichier

Fichier chargé avec succès : 3.46 Ko

Choisissez une colonne :

Age\_of\_woman

Choisissez le type d'analyse :

Sélectionnez un type d'analyse

Statistiques Descriptives

Analyse Démographique

Visualisation Graphique d'Analyse Démographique

Analyse De Corrélation

### Exemple d'Analyse :

Glissez et déposez un fichier CSV ou Excell cliquez pour sélectionner un fichier

Fichier chargé avec succès : 0.47 Ko

Choisissez une colonne :

Salary

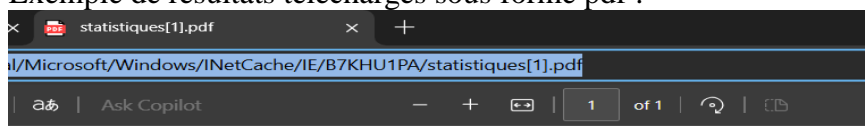
Choisissez le type d'analyse :

Statistiques Descriptives

Statistique	Valeur
Moyenne	5362.5
Écart-type	526.2739645035518
Médiane	5250
Valeur Min	4800
Valeur Max	6200
1er Quartile	4975
3e Quartile	5625
Somme	42900
Nombre de valeurs	8

Télécharger les résultats (PDF)

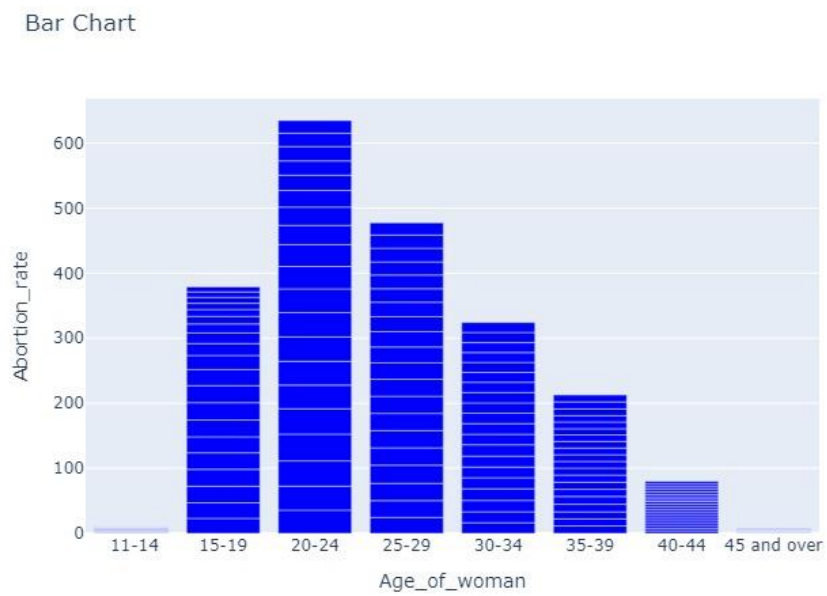
### Exemple de résultats téléchargés sous forme pdf :



#### Résultats de l'analyse (Statistiques)

Statistique: Moyenne | Valeur: 5362.5  
Statistique: Écart-type | Valeur: 526.2739645035518  
Statistique: Médiane | Valeur: 5250.0  
Statistique: Valeur Min | Valeur: 4800.0  
Statistique: Valeur Max | Valeur: 6200.0  
Statistique: 1er Quartile | Valeur: 4975.0  
Statistique: 3e Quartile | Valeur: 5625.0  
Statistique: Somme | Valeur: 42900.0  
Statistique: Nombre de valeurs | Valeur: 8.0

## Exemple de **Save Png**:



# CONCLUSION

Le mini-projet Graph Maker démontre une approche pratique et interactive pour traiter les données à travers quatre grandes tâches principales : l'inscription des utilisateurs, le ménage des données, l'analyse, et la visualisation.

Grâce à une interface conviviale et aux fonctionnalités robustes offertes par Python et ses bibliothèques comme Plotly Dash et Pandas, le projet répond efficacement aux besoins des utilisateurs en matière de gestion et d'interprétation des données.

Ce projet met également en avant l'importance de la collaboration entre différentes technologies, notamment l'intégration de bases de données sécurisées via Firebase Firestore et l'utilisation de bibliothèques spécialisées pour garantir la fiabilité et la performance des traitements.

Graph Maker constitue une base solide pour de futurs développements, permettant d'élargir ses fonctionnalités et d'améliorer encore davantage l'expérience utilisateur.

Ce projet a permis de consolider des compétences clés dans des domaines variés, allant de la programmation Python au développement d'applications interactives, tout en renforçant une méthodologie structurée pour la conception et l'exécution de solutions logicielles.