

Projet Rapport Forms

Partie 1



Rapport

Participant : Berrebi Jérémie

Nom du référent : BRANCHET Bérengère

Client : DUONG Quoc Bao

TABLE DES MATIERES

1/ Présentation du projet :	3
-----------------------------------	---

Forms

2023

2/ Définition du problème :	3
3/ Proposition	4
4/ Objectif	4
I/ Fonctionnement générale du code :	5
II/ Fonctionnement plus en détail :	5
III/ Comment faire marcher l'application :	15
IV/ Perspectives :	21

Forms

2023

Le projet Rapport Forms

1/ Présentation du projet :

Le Projet s'inscrit dans le cadre du rattrapage de projet Pi2A4 de Jérémie Berrebi.

Le projet a été formulé par DUONG Quo Bao, client final du projet. Mais la solution peut être utilisable pour toute personne ayant la même problématique.

La référente du projet est Madame Bérangère Branchet.

Le thème du projet est la simplification, l'automatisation et la digitalisation d'un processus métier.

2/ Définition du problème :

Le problème qui se pose aujourd'hui concerne l'analyse des réponses des questionnaires Forms et la génération d'un rapport associé. L'utilisation de Microsoft Forms permet l'envoi d'un questionnaire créé avec celui-ci. Cet outil est simple, et permet de réaliser des questionnaires simplement. Après l'envoi des questionnaires, l'outil Microsoft Forms permet d'analyser les réponses des répondants. Microsoft Forms fournit des visualisations simples et intégrées à la visualisation de résultat des réponses du questionnaire. Cependant le client souhaite pouvoir réaliser un rapport avec les visualisations résultant du questionnaire. C'est-à-dire pouvoir écrire en dessous, gérer les visualisations affichées. Ainsi le projet consiste à réaliser un outil permettant une simplification de cette tâche.

Limitation de l'outil Microsoft Forms :

L'outil Microsoft Forms permet la création d'un questionnaire assez facilement. De plus la réponse obtenue peut être analysée et visualisée outil. De plus le fait que le questionnaire soit intégré à l'outil, la génération d'un compte rendu est assez facile pour l'outil Microsoft Forms, puisqu'il connaît déjà le nombre de questionnaires. L'outil fournit des visualisations assez simples pour l'analyse de chaque réponse. On notera que les visualisations sont principalement des histogrammes pour des questions à cocher ou des nuages de mots pour l'analyse des réponses textuelle. De plus on peut ajouter des analyses entre questions, et comprendre certaines relations entre répondant. On peut également supprimer les questions que l'on ne veut pas voir dans le rapport dans le cas où la question n'aurait pas forcément d'intérêt. Enfin si nous voulons exporter les résultats nous pouvons le faire. L'outil Microsoft Forms permet de générer un fichier Excel contenant les réponses, de plus permet de partager un lien récapitulatif pour visualiser facilement sur une page web les résultats du questionnaire. Et l'outil permet aussi de télécharger les résultats. Cependant un des

Forms

2023

principaux défauts de l'outil Microsoft Forms, et qu'il ne permet pas d'écrire en dessous des résultats des questionnaires un commentaire. Ce qui le problème actuel soulevé par le client.

3/ Proposition

Après avoir analysé les solutions proposées, nous avons convenu que la solution 2, serait la solution adoptée.

La solution consiste à la création d'une application web pour créer une page d'analyse des résultats. Cette page récupérera les données du tableau Excel de réponse et affichées question par questions les analyses des résultats avec une visualisation.

En fait l'idée est de reproduire la page Microsoft Forms d'analyse des résultats, et d'ajouter nos fonctionnalités souhaitées, notamment la seule que le client souhaite c'est-à-dire pouvoir ajouter des paragraphes de textes.

4/ Objectif

Le produit doit permettre l'analyse des réponses des questionnaires Forms. C'est-à-dire qu'à chaque question avoir une visualisation associée pour comprendre rapidement et simplement les résultats du questionnaire.

L'outil doit pouvoir permettre l'insertion de case de texte pour d'éventuels commentaire entre deux cellules de visualisation, pour commenter ou expliquer les résultats.

Enfin l'outil doit pouvoir générer un rapport, document Word, PDF.

L'outil doit être aussi pratique d'utilisation et simple à lancer pour travailler dessus.

Forms

2023

Rapport Technique

I/ Fonctionnement générale du code :

L'outil est une application web développée en Django. Le langage de programmation est essentiellement du Python pour la partie backend, Javascript et Html pour la partie frontend. L'idée de l'application est de pouvoir générer automatiquement et dynamiquement une page de rapport Html, et que l'on puisse ajouter du texte en dessus, et enfin l'exporter en PDF.

Le projet peut être découpé en 3 parties :

Première partie concernant l'importation des données, on doit pouvoir importer des données issues de MS Forms. Le choix est fait de ne prendre uniquement les sorties exportées par forms, donc .xlsx.

Deuxième partie concernant la visualisation des graphiques, on doit générer une page suivant le nombre de colonne du dataset, on aura un graphique. J'ai choisi de réaliser les graphiques avec la librairie Chart.js, qui permet de faire de beaux graphiques sur une page web, en intégrant les données à la page.

On a des boutons de customisation, notamment changer le graphique, ou ajouter un texte. Il faut différencier les questions choix multiple et ouverte. On a une détection automatique du type de question, et suivant cela l'affichage est différent.

Troisième partie concernant l'exportation du rapport, où l'on veut exporter le rapport créé, au format souhaité.

II/ Fonctionnement plus en détail :

Dossier :

Listings/images : dossier où sont stockés les words clouds

Listings/media : dossier où sont stockés les fichiers importés

Templates : dossier où sont stockés les pages html

Page Html :

Base.html

La page base.html est une page HTML qui sert de modèle pour toutes les autres pages de l'application web. Elle contient des éléments de base tels que le titre, la feuille de style CSS et les scripts JavaScript utilisés dans toutes les pages.

Forms

2023

Le contenu de la page est défini dans les blocs "content" et "messages". Ces blocs permettent d'insérer dynamiquement le contenu de chaque page spécifique. Cela signifie que le contenu réel de chaque page sera ajouté à la place de `{% block content %}{% endblock %}` et `{% block messages %}{% endblock %}`.

La page contient également une barre de navigation qui permet de naviguer entre les différentes pages de l'application. Cette barre de navigation contient des liens vers les pages d'importation, d'analyse et d'exportation de données.

En outre, la page contient un script JavaScript qui permet d'ajouter des paragraphes de commentaires à chaque colonne des résultats de l'analyse. Ce script permet aux utilisateurs de modifier ou de supprimer des commentaires existants et d'en ajouter de nouveaux.

Enfin, la page utilise plusieurs scripts externes pour ajouter des fonctionnalités à l'application. On a notamment la fonction pour ajouter un paragraphe `addparagraph`. Ces scripts incluent le framework `Chart.js` pour la visualisation des données et le plugin `datalabels` pour afficher les étiquettes de données sur les graphiques. Le script `jspdf` est également utilisé pour générer des fichiers PDF à partir des résultats de l'analyse.

En résumé, la page `base.html` est un modèle qui définit les éléments de base de l'application web, tels que la barre de navigation et les scripts JavaScript, et qui permet d'insérer dynamiquement le contenu de chaque page spécifique.

Importation.html

La page `importation.html` est une page HTML qui étend la page de base (`base.html`) via la commande `{% extends 'listings/base.html' %}` pour réutiliser les éléments communs à toutes les pages. Elle permet l'importation du fichier Excel (`.xlsx...`), si on importe les données autres que Excel comme csv on a un message d'erreur, le fichier est stocker dans le dossier média.

Le contenu de cette page est placé dans le bloc "content" défini dans la page de base. Le contenu est composé d'un titre principal ("Importation des données") et d'un formulaire pour télécharger un fichier Excel (`.xlsx`). Le formulaire a une méthode POST pour soumettre les données au serveur, et utilise la directive CSRF pour protéger contre les attaques de type Cross-Site Request Forgery. Le formulaire contient également un champ de fichier avec le nom "document", qui est requis. Enfin, il y a un bouton "Upload" qui permet de soumettre le formulaire. Quand on clique dessus les données dans la session storage sont supprimé. Si on

La page a également un bloc "messages" pour afficher des messages d'erreur ou de confirmation éventuels. Si la variable "messages" est définie, elle affichera chaque message dans un bloc div. Cela arrive quand l'utilisateur a mis un fichier différent de `.xlsx`.

Forms

2023

Analyse.html

Cette page Html crée une page web qui affiche des graphiques pour plusieurs questions d'un sondage en ligne, ainsi que les commentaires des utilisateurs pour chaque question. Cette page est générée automatiquement suivant le fichier en entré avec le nombre de colonne, plus il y a de colonne plus il y aura de question. On associe chaque paragraphe a une visualisation.

La page est basée sur un modèle de base 'listings/base.html' qui est étendu par ce template. Le contenu principal de la page est affiché dans un bloc appelé "content". Le code utilise également une boucle "for" pour afficher les graphiques et les commentaires pour chaque question dans le sondage.

Chaque graphique est créé avec la bibliothèque JavaScript Chart.js, qui permet de générer facilement différents types de graphiques. Les graphiques peuvent être un diagramme circulaire ou un histogramme, et l'utilisateur peut basculer entre ces types en utilisant une liste déroulante.

Le code utilise également des images de nuages de mots générées à partir des commentaires des répondant à la question.

Si l'utilisateur clique sur ajouter un commentaire il faut ajouter, modifier ou supprimer un commentaire.

La logique de la page est basée sur le type de question si elle est ouverte ou fermé. Le booléen "bool_open" est défini sur "True", ce qui fait apparaître l'image du nuage de mots et masque la liste déroulante des types de graphique. On génère un nuage de mots pour les questions ouverte , ou un bar plot, pie chart pour celle qui sont fermé.

Si l'utilisateur clique sur une autre option de la liste déroulante, la fonction JavaScript "updateChart" est appelée pour mettre à jour le graphique avec le nouveau type de graphique sélectionné.

En résumé, cette page HTML génère une page web interactive avec des graphiques et des commentaires générés à partir des données d'un sondage. Le code utilise la bibliothèque JavaScript Chart.js pour générer les graphiques et Jinja pour générer des modèles dynamiques en fonction des choix de l'utilisateur.

Exportation.html

Forms

2023

La page `exportation.html` étend également le template de base `base.html` à l'aide de `{% extends 'listings/base.html' %}`. Elle définit deux blocs, `content` et `messages`. Elle permet l'exportation en PDF ou Word de la page `analyse.html` en pdf ou word suivant le choix de l'utilisateur. Elle permet l'aperçu ou le téléchargement.

Dans le bloc `content`, il y a un formulaire qui permet de choisir le format d'exportation (PDF ou Word) et deux boutons, un pour télécharger le fichier exporté (Télécharger) et un pour prévisualiser le fichier exporté (Aperçu).

Le formulaire utilise la méthode POST pour soumettre les données au serveur et inclut un token CSRF avec `{% csrf_token %}` pour éviter les attaques CSRF (Cross-Site Request Forgery). Le choix de l'utilisateur est stocké dans une liste déroulante (`<select>`), où les options sont définies avec `<option>`. Le premier bouton utilise un lien `<a>` pour télécharger le fichier exporté, tandis que le deuxième bouton utilise du JavaScript (`onclick`) pour rediriger l'utilisateur vers la page de prévisualisation.

On affiche un message si l'utilisateur n'a pas mis cliquer sur sauvegarder.

On change l'url `u` on se dirige dépendant de la sélection word ou pdf. Pour résumé le code on a 2 boutons, qui envoie sur des requêtes dans la vue en post et get, en cliquant sur aperçu on renvoi les données de la session storage, puis on redirige la page vers la vue `not_upload`. En téléchargeant on télécharge le fichier.

pdf.html

La page permet l'affichage en PDF de la page `Analyse.html`, ainsi on fait la même chose, mais en enlevant certaines parties qu'on n'en souhaite pas voir dans le document PDF. C'est un template de fichier PDF. On peut ce que l'on veut dans le fichier généré.

Fichier python :**Views.py**

Ce code contient des vues pour une application Django qui effectue une analyse de données à partir d'un fichier Excel. Voici une brève explication de chaque vue:

importation: Cette vue gère l'importation du fichier. Si un fichier Excel est téléchargé, il est enregistré, puis la vue redirige vers la vue `analyse`. Si le fichier n'a pas la bonne extension, un message d'erreur est affiché. La vue "importation" permet de télécharger un fichier Excel (.xlsx),

Forms

2023

de le stocker dans un dossier et d'appeler la fonction "readfile" pour lire les données du fichier. Si le fichier téléchargé n'est pas au format Excel, un message d'erreur sera affiché.

La fonction "delete_files" permet de supprimer tous les fichiers d'un dossier. Les dossiers où les fichiers seront téléchargés sont définis dans les variables "media_directory" et "images_directory". Si le fichier téléchargé est en format Excel, cette fonction supprimera tous les fichiers dans ces deux dossiers avant d'importer le nouveau fichier.

La variable "upload_file" est utilisée pour stocker le fichier téléchargé via le formulaire. La fonction "savefile" est appelée pour enregistrer le fichier dans le dossier "media_directory". Le nom du fichier est stocké dans la variable "name" et le chemin complet du fichier est stocké dans la variable "file_directory".

Enfin, la fonction "redirect" est appelée pour rediriger l'utilisateur vers la vue "analyse" qui affichera les données du fichier.

importation(request) : La fonction "importation" est une vue qui permet aux utilisateurs de télécharger un fichier Excel. Si le fichier téléchargé est au format ".xlsx", la fonction supprime tous les fichiers des répertoires "media_directory" et "images_directory", enregistre le fichier téléchargé avec "FileSystemStorage", et appelle la fonction "readfile" pour lire le fichier. Enfin, la fonction redirige l'utilisateur vers la vue "analyse" de l'application. Si le fichier n'est pas au format ".xlsx", la fonction affiche un message d'erreur.

analyse(request) : Cette fonction correspond à la vue qui permet d'analyser les données importées. Si le fichier a été importé avec succès, elle appelle la fonction 'context_creation' pour créer le contexte de données et renvoie le template 'analyse.html' avec le contexte. Sinon, elle affiche un message d'erreur.

La vue "analyse" affiche les données du fichier Excel. La fonction "context_creation" est appelée pour créer le contexte des données à afficher sur la page. Si aucun fichier n'a été téléchargé, un message d'erreur sera affiché.

La fonction "context_creation" crée le contexte des données à partir du fichier Excel en appelant les fonctions "importation_dataset", "clean_dataset", "questions_dataset", "mean_response_time", "questions_list", "bool_list_verbal", et "plot_cloud". Elle stocke les données dans un dictionnaire "context" pour pouvoir les afficher sur la page.

La vue "analyse" renvoie le template "analyse.html" avec le contexte créé.

Forms

2023

exportation(request) : Cette fonction correspond à la vue qui permet d'exporter les données. Elle renvoie le template 'exportation.html'.

Cette vue est une fonction qui traite les requêtes HTTP GET et POST et retourne une réponse HTTP. Elle commence par initialiser deux variables globales, data et export_format, avec une valeur vide pour data et 'pdf' pour export_format.

Lorsqu'une requête POST est reçue, elle vérifie si la variable file_directory existe dans les variables globales. Si c'est le cas, elle récupère le chemin absolu du dossier "images" à partir des paramètres de configuration de l'application, puis crée un contexte avec les données du formulaire et ajoute le chemin absolu du dossier "media" au contexte. Les données de la requête HTTP sont ensuite récupérées et stockées dans le contexte, puis les URLs des graphes et les commentaires sont transformés en listes et ajoutés aux données. Le contexte mis à jour est ensuite utilisé pour générer un fichier PDF à partir d'un modèle HTML, en utilisant la fonction render_to_pdf.

Lorsqu'une requête GET est reçue, elle récupère les données stockées dans la variable data, génère un fichier PDF à partir d'un modèle HTML en utilisant la fonction render_to_pdf, puis vérifie la valeur de export_format. Si elle est 'pdf', le fichier PDF est renvoyé comme une réponse HTTP avec le nom "Analyse_resultat_forms.pdf". Si elle est 'word', le fichier PDF est d'abord enregistré dans le répertoire "media" de l'application, puis converti en fichier Word et renvoyé comme réponse HTTP avec le nom "Analyse_resultat_forms.docx".

Si la requête n'est ni POST ni GET, elle renvoie une réponse HTTP "404 not found".

Voici une explication de chaque fonction de ce fichier views.py :

readfile(filename) : La fonction "readfile" prend le nom d'un fichier Excel et le lit à l'aide de la bibliothèque pandas. Les données du fichier sont stockées dans un objet pandas DataFrame nommé "data". La fonction calcule également le nombre de lignes et de colonnes dans le DataFrame, ainsi que le nombre de lignes contenant des valeurs manquantes.

context_creation(file_directory) : La fonction "context_creation" prend le chemin d'un fichier Excel et génère un dictionnaire de données pour alimenter le tableau de bord de l'application. La fonction appelle plusieurs autres fonctions pour nettoyer les données, analyser les questions posées aux participants de l'enquête, et créer des graphiques. La fonction renvoie le dictionnaire "context" qui contient des informations sur les colonnes du DataFrame, le nombre total de lignes, le temps de réponse moyen, les données du tableau de bord, et des indicateurs pour savoir si les questions sont ouvertes ou fermées.

Forms

2023

Le reste des fonctions sont des fonctions utilitaires appelées par la fonction 'context_creation' pour nettoyer les données, calculer des statistiques et générer des graphiques. Parmi ces fonctions, on peut citer :

- **importation_dataset(file_directory)** : Cette fonction permet d'importer le fichier et de renvoyer un objet DataFrame de pandas contenant les données.
- **clean_dataset(dataset)** : Cette fonction permet de nettoyer les données en supprimant les lignes contenant des valeurs manquantes.
- **questions_dataset(dataset)** : Cette fonction permet de créer un sous-ensemble des données ne contenant que les colonnes correspondant aux questions posées dans l'enquête.
- **mean_response_time(dataset_cleaned)** : Cette fonction calcule le temps moyen de réponse à l'enquête.
- **questions_list(column_names)** : Cette fonction prend en paramètre une liste de noms de colonnes et renvoie une liste de tuples (i, i+1, nom_de_colonne) pour chaque colonne.
- **bool_list_verbal(dataset_questions)** : Cette fonction prend en paramètre le sous-ensemble de données créé par la fonction 'questions_dataset' et renvoie une liste de booléens indiquant si chaque question est ouverte (True) ou fermée (False).
- **plot_cloud(column_values, column_number)** : Cette fonction prend en paramètre les valeurs d'une colonne et son numéro et génère un nuage de mots pour cette colonne, qu'elle enregistre dans le dossier images.

Visualisation.py

Ce code comporte plusieurs fonctions qui permettent de nettoyer, filtrer et normaliser un jeu de données (dataset) contenant des réponses à des questions d'un questionnaire. Et permet de créer les visualisations des nuages de mots.

La fonction `importation_dataset(file_directory)` prend en entrée le chemin d'accès à un fichier Excel et retourne un DataFrame pandas.

La fonction `metadata(questions)` prend en entrée une liste de questions et retourne une liste contenant les métadonnées de ces questions (nom, prénom, e-mail, date de naissance, etc.).

La fonction `filter_question(dataset)` prend en entrée un DataFrame pandas et retourne une liste de questions filtrée selon des critères définis (ne pas prendre les colonnes "Total points" ou "Quiz feedback", etc.).

La fonction `question_cleaning(list_question)` prend en entrée une liste de questions et retourne cette liste après avoir nettoyé les caractères spéciaux.

Forms

2023

La fonction `name_columns(questions)` prend en entrée une liste de questions et retourne une liste contenant les noms des colonnes pour le DataFrame pandas.

La fonction `questions_list(questions)` prend en entrée une liste de questions et retourne une liste contenant un tuple pour chaque question. Ce tuple associe le nom de la colonne pour cette question avec la question elle-même.

La fonction `clean_dataset(dataset)` prend en entrée un DataFrame pandas et retourne ce DataFrame après avoir nettoyé les colonnes, filtré les questions, renommé les colonnes, etc.

La fonction `questions_dataset(dataset)` prend en entrée un DataFrame pandas et retourne un DataFrame contenant seulement les questions.

La fonction `mean_response_time(dataset_cleaned)` prend en entrée un DataFrame pandas nettoyé et retourne le temps moyen (en minutes) que les participants ont pris pour remplir le questionnaire.

La fonction `remove_not_multichoice(dataset,liste)` prend en entrée un DataFrame pandas et une liste de questions et retourne une liste de questions qui sont des choix multiples.

La fonction `keep_is_verbal(dataset,liste)` prend en entrée un DataFrame pandas et une liste de questions et retourne une liste de questions qui sont verbales.

La fonction `bool_list_verbal(dataset)` prend en entrée un DataFrame pandas et retourne une liste booléenne qui indique pour chaque question si elle est verbale ou non.

La fonction `clean(s)` prend en entrée une chaîne de caractères et retourne cette chaîne de caractères après avoir nettoyé les caractères spéciaux.

La fonction `normalize_text(s,langue='french')` prend en entrée une chaîne de caractères et retourne cette chaîne de caractères après avoir normalisé le texte (singulier, enlever les stopwords, etc.).

La fonction `preprocessing(list_sentence)` prend en entrée une liste de chaînes de caractères et retourne une liste de chaînes de caractères normalisés.

Utils.py :

Ici on met les fonction qui nous sont utiles pour l'application .

Forms

2023

Le module `Utils.py` contient une fonction nommée `render_to_pdf` qui utilise les bibliothèques `Django`, `xhtml2pdf` et `io` pour générer un document PDF à partir d'un modèle Django (template).

La fonction prend deux arguments : `template_src` qui est le chemin d'accès au fichier de modèle (template) utilisé pour générer le document PDF, et `context_dict` qui contient le contexte utilisé pour le rendu du modèle (template).

La fonction charge le template en utilisant la fonction `get_template` de Django, puis le rend en utilisant le contexte fourni en appelant la fonction `render`. Le résultat est stocké dans un objet `BytesIO` (`BytesIO` est utilisé pour stocker les données en mémoire et les manipuler comme un fichier binaire).

Ensuite, la fonction utilise la bibliothèque `xhtml2pdf` pour convertir le HTML généré à partir du template en un document PDF, en appelant la fonction `pisaDocument`. Si la conversion se déroule sans erreur, le contenu du fichier PDF est retourné dans un objet `HttpResponse`, avec le type de contenu défini comme `"application/pdf"`. Si la conversion échoue, la fonction retourne `None`.

Urls.py

Ce code Python définit les chemins d'URL pour une application web Django.

La variable `urlpatterns` est une liste de `path()` qui correspond à une vue ou une fonction spécifique pour chaque URL.

Chaque `path()` a deux arguments obligatoires :

- un chemin (une chaîne de caractères) qui correspond à l'URL à partir de la racine du domaine,
- une fonction de vue qui est appelée lorsque cette URL est visitée.

Dans ce cas, les URLs sont :

- `/admin/` qui est l'URL de l'interface d'administration de Django.
- `/importation/` qui correspond à la vue `views.importation` qui gère l'importation des données.
- `/analyse/` qui correspond à la vue `views.analyse` qui gère l'analyse des données.
- `/exportation/` qui correspond à la vue `views.exportation` qui gère l'exportation des données.

Forms

2023

- `/analyse_erreur/` qui correspond à la vue `views.not_upload` qui affiche un message d'erreur si l'utilisateur n'a pas téléchargé de fichier.
- `/exportation/pdf` qui correspond à la vue `views.pdf_view` qui génère un PDF à partir des données.

Chaque `path()` a également un argument optionnel `name` qui est utilisé pour donner un nom à l'URL. Cela peut être utile pour référencer cette URL dans d'autres parties du code.

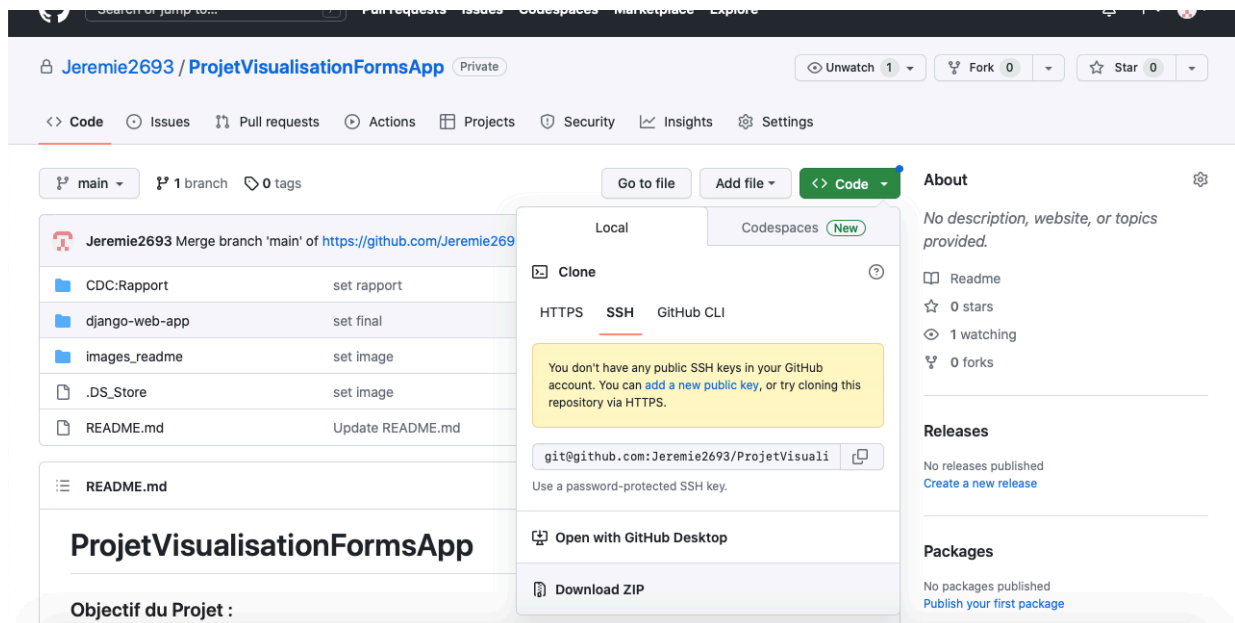
Forms

2023

III/ Comment faire marcher l'application :

Etape Installation :

Etape 1 : Sur le Github télécharger le projet



- Etape 2 (Optionnel) : déplacer le dossier téléchargé dans vos fichier application, ou vous voulez.

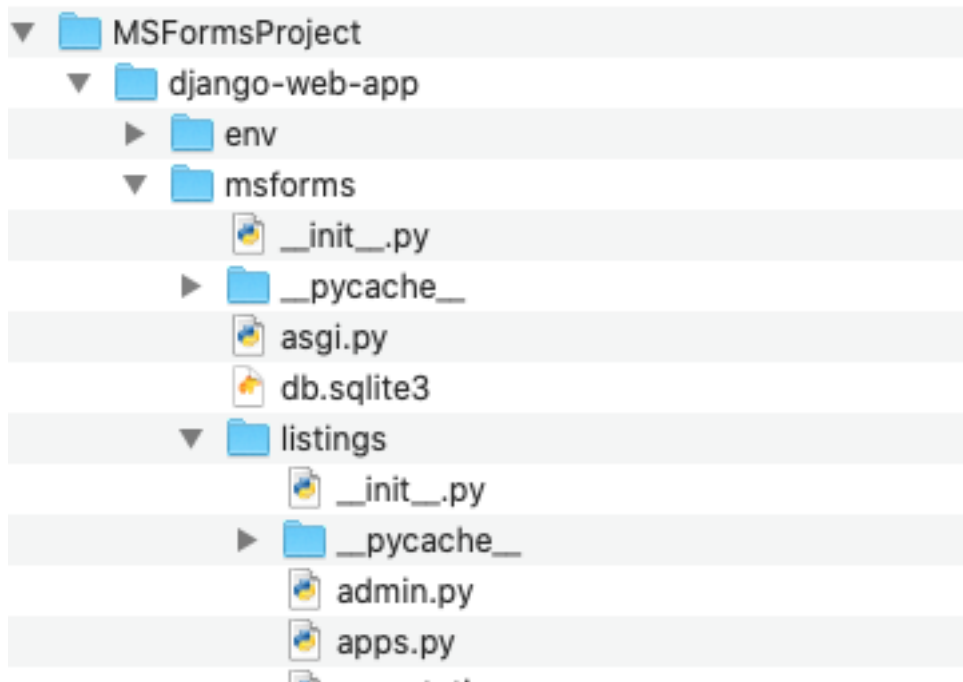
Forms

2023

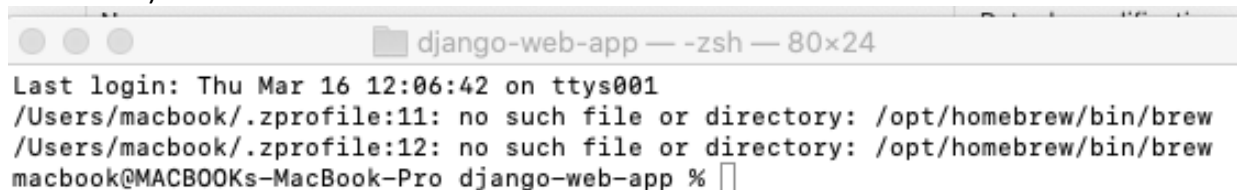
Etape Ouverture de l'application

En local :

Etape 1 : Ouverture du dossier projet.



Etape 2 : ouverture du terminal du dossier (clic droit sur django-web-app + nouveau terminal du dossier)



Etape 3 : (écrire le script pour lancer le serveur web en local)

Forms

2023

```
```shell
```

```
source env/bin/activate
cd msforms
python3 manage.py runserver
```

```
```
```

```
macbook@MACBOOKs-MacBook-Pro django-web-app % source env/bin/activate
cd msforms
python3 manage.py runserver
Watching for file changes with StatReloader
Performing system checks...

System check identified no issues (0 silenced).
March 16, 2023 - 12:05:04
Django version 4.1.6, using settings 'msforms.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CONTROL-C.
```

Etape 4 : aller sur la page : port d'écoute + /importation/ ici :
<http://127.0.0.1:8000/importation/>



The screenshot shows a web browser window with the address bar displaying 127.0.0.1:8000/importation/. The page has a navigation bar with three links: [Importation des données](#), [Analyse des résultats](#), and [Exportation des résultats](#). The main heading is **Importation des données**. Below the heading, the text reads "Veuillez importer une source de donnée (.xlsx) :". There are two buttons: "Choisir le fichier" and "Upload". The "Choisir le fichier" button is disabled, and the text "aucun fichier sél." is displayed next to it.

Forms

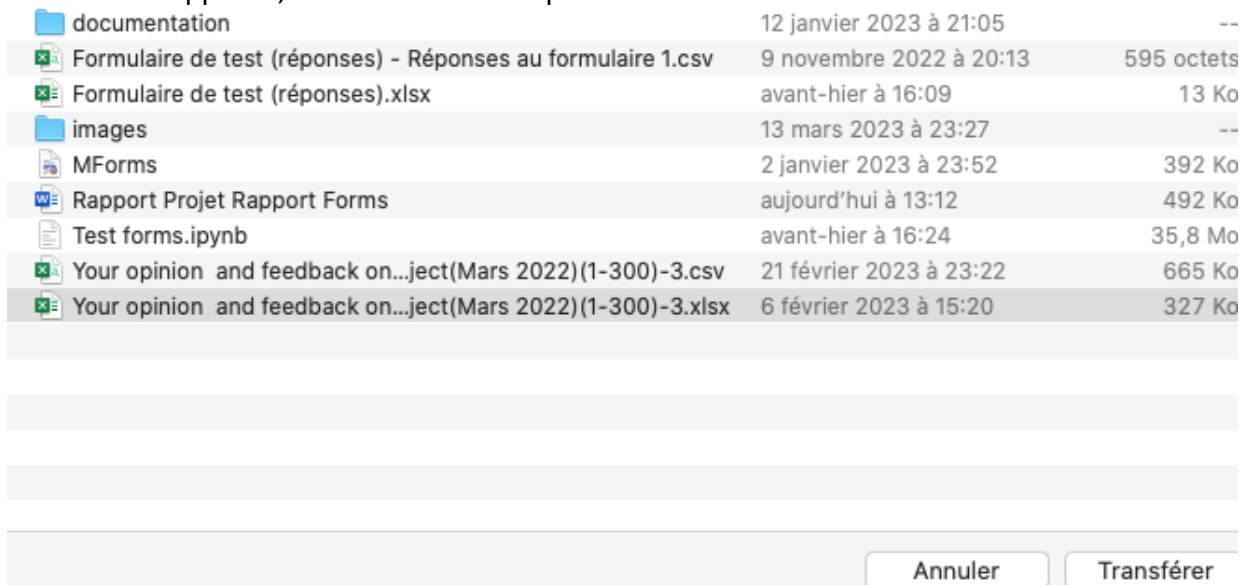
2023

Etape Utilisation de l'application

Etape 5 : utiliser l'outil : cliquer sur le bouton choisir le fichier



Une fenêtre apparaît, choisir le fichier a uploader :



| | | |
|--|-------------------------|------------|
| documentation | 12 janvier 2023 à 21:05 | -- |
| Formulaire de test (réponses) - Réponses au formulaire 1.csv | 9 novembre 2022 à 20:13 | 595 octets |
| Formulaire de test (réponses).xlsx | avant-hier à 16:09 | 13 Ko |
| images | 13 mars 2023 à 23:27 | -- |
| MForms | 2 janvier 2023 à 23:52 | 392 Ko |
| Rapport Projet Rapport Forms | aujourd'hui à 13:12 | 492 Ko |
| Test forms.ipynb | avant-hier à 16:24 | 35,8 Mo |
| Your opinion and feedback on...ject(Mars 2022)(1-300)-3.csv | 21 février 2023 à 23:22 | 665 Ko |
| Your opinion and feedback on...ject(Mars 2022)(1-300)-3.xlsx | 6 février 2023 à 15:20 | 327 Ko |

Annuler Transférer

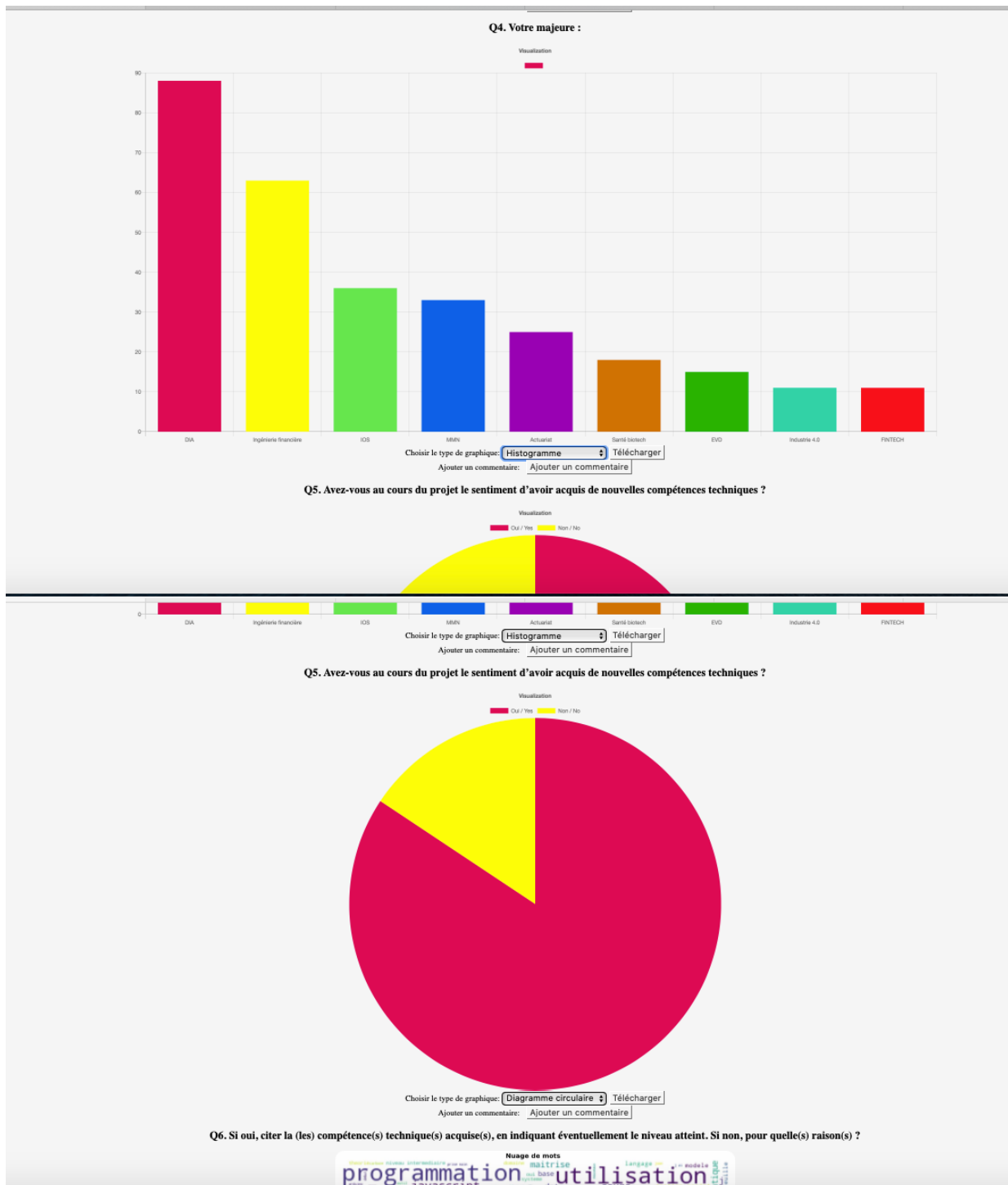
Cliquer sur Transférer :

Etape 6 : cliquer sur upload (redirection sur la page Analyse)

Etape 7 : Etape analyse de résultat changement de graphique, ajout de texte.

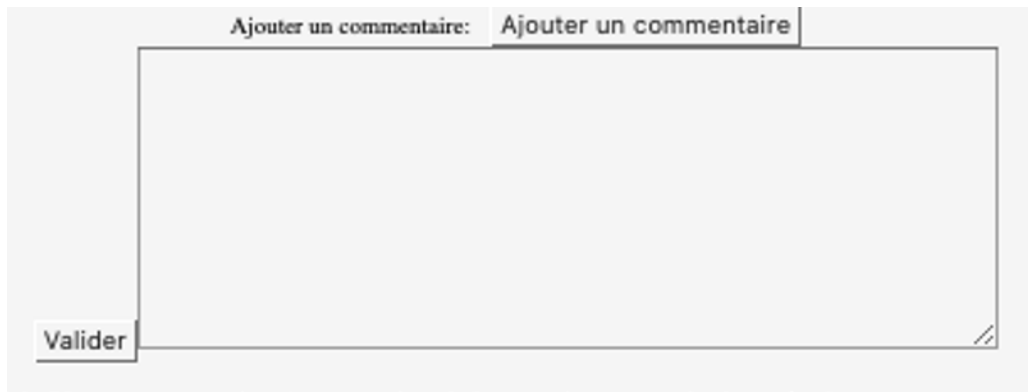
Forms

2023



Forms

2023

A screenshot of a web form for adding comments. At the top, there are two tabs, both labeled 'Ajouter un commentaire:'. Below the tabs is a large, empty rectangular text area. In the bottom-left corner of the form, there is a button labeled 'Valider'. In the bottom-right corner, there is a small icon of a pencil inside a square.

- Etape 8: cliquer sur le bouton sauvegarder de la page si vous voulez sauvegarder le texte et la sélection des images dans le rapport pdf généré.

Etape 9 : cliquer sur Exportation des résultats : choisir le format puis exporter, avec aperçu ou télécharger.

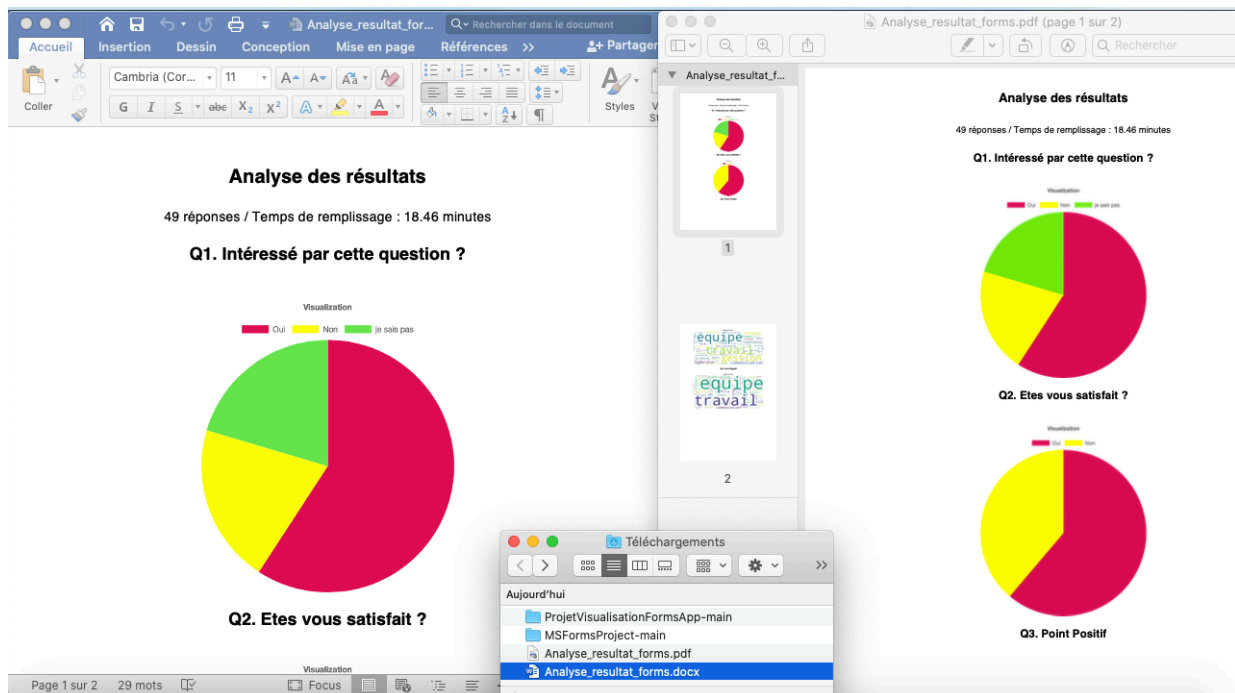
A screenshot of a web page titled 'Exportations des résultats'. At the top, there are three links: 'Importation des données', 'Analyse des résultats', and 'Exportation des résultats'. The main heading is 'Exportations des résultats'. Below it, the text 'Veuillez choisir un format d'exportation :' is displayed. Underneath, there is a section 'Exporter en :'. This section contains a dropdown menu with 'PDF' selected, and two buttons: 'Télécharger' and 'Aperçu'.

Note : cliquer sur aperçu avant de télécharger

Résultat :

Forms

2023



Après déploiement, il y a juste besoin d'aller sur la page web associée à l'application. Et utiliser l'application comme à l'étape 6.

IV/ Perspectives :

Le plus dur a été fait tout ce qui vient après c'est de la déco.

Les 3 parties les plus dur du projet ont été réaliser, à savoir, l'importation et le pré processing, l'analyse et le rendu en html et enfin l'exportation PDF et Word.

J'ai réussi à exporter et faire en sorte de générer un pdf dynamique et un word, ce qui a été une tâche très dure, mais qui aujourd'hui marche.

Maintenant on peut passer à la déco et s'amuser avec le code pour rendre jolie l'application en termes de graphisme. Des graphiques plus beaux, des boutons plus beaux...

Aussi le rapport généré peut être amélioré, par une typographie choisie, un filigrane ajouté.

L'application web présentée ici présente de nombreuses perspectives d'amélioration. En effet, il est toujours possible de perfectionner une application informatique, et plusieurs améliorations pourraient être apportées.

La correction des bugs constitue également une priorité pour améliorer l'application. Un bug surprenant est la superposition des graphiques, qui peut survenir lorsqu'on change l'affichage

Forms

2023

pour un bar-plot, et qui entraîne un changement de graphique non voulu. Il serait possible de supprimer le graphique de la liste des graphiques lorsqu'on change de type d'affichage pour résoudre ce problème. La version 2 aura ces bugs corrigés, et une version 2 sera uploadé dans quelques mois.

Par ailleurs, plusieurs nouvelles fonctionnalités pourraient être ajoutées pour améliorer l'application. Il serait notamment intéressant d'intégrer de nouveaux types de graphiques, en plus des pie-charts et des bar-charts. Des options de personnalisation pourraient également être mises en place, comme le choix de la couleur ou du nombre de mots affichés. Un bouton pour supprimer les paragraphes, les questions ou les graphiques qui ne sont pas pertinents serait également utile. Enfin, déployer l'application web pour la rendre plus facilement accessible depuis une URL serait un élément clé pour améliorer l'expérience utilisateur.

En somme, il existe de nombreuses pistes d'amélioration pour cette application web. En optimisant l'expérience utilisateur et en ajoutant de nouvelles fonctionnalités, cette application pourrait répondre encore mieux aux besoins de ses utilisateurs

V/ Conclusion :

Ce projet s'est avéré être très intéressant car il a permis de développer de nouvelles compétences, notamment dans le domaine du développement d'application web. Toutefois, il a été extrêmement difficile à réaliser, car il comporte des fonctionnalités avancées qui ne sont pas simples à mettre en place. En effet, les fonctionnalités d'importation et d'exportation sont particulièrement complexes, car elles impliquent l'intégration d'images et de graphiques chart.js dans le résultat final, ce qui est rarement vu dans les projets de ce genre. De plus, il est nécessaire de stocker le texte saisi par l'utilisateur afin de l'exporter dans le PDF avec les commentaires.

De plus, la partie visualisation est également très difficile, car il est rare de trouver des projets dans lesquels la source de données n'est pas statique. Il est donc nécessaire de développer en généralisant par rapport à un fichier que l'on ne possède pas. Il est également nécessaire d'intégrer des graphiques, de changer les graphiques sur la page et de générer automatiquement la page, qui peut varier en fonction de la source de données. Tout cela ajoute une complexité supplémentaire à la réalisation du projet.

Enfin la partie exportation en pdf a été la partie la plus difficile. Car on doit créer un PDF dynamique et qui intègrent des images et surtout des canvas chart.js. Car on a beaucoup de contraintes associées qui limite la façon de coder. Puis il y a aussi l'exportation en Word.