Rapport TPassword

Cours de Développement d'Application
Sous la supervision de M. TOMCZAK Robert

DOCUMENT CONFIDENTIEL

DEBREU Romain

BOURLET Cyril

AKOBI Bancolé

DELPECH Nathan

I. DESCRIPTION

Un des grands enjeux de la cybersécurité est de formé les personnels et les utilisateurs a adopté une hygiène numérique. En effet la majorité des cyber-attaques ont pour vecteur l'humain. Le projet TPassword a pour but de sensibiliser sur la sécurité des mots de passe des utilisateurs, via un site web donnant de manière objective la sécurité de leurs mots de passes. Le projet inclus aussi une option de création de mots de passes sur des critères donnés par l'utilisateur et donnant le niveau de sécurité du mot de passe crée afin de palier à un potentiel mot de passe faible.

L'attention du lecteur sera attirée par le caractère **CONFIDENTIEL**, de ce document qui ne doit en aucun cas publié sur quelque plateforme que ce soit ou communiqué à un tiers, en effet ce document pourrait exposer le projet à des faille de sécurité importante.

II. TABLE DES MATIERES

III. INTRODUCTION

Comme rappelé dans la description, l'importance de la fragilité de l'humain aux cyber-attaques nécessite aujourd'hui une attention particulière. Tout d'abord dans la réaction aux tentatives d'ingénierie social, mais également la sureté des mots de passes. En effet un mot de passe trop faible ou la réutilisation du mot de passe sur plusieurs site différent, peut constituer une faille de sécurité grande ouverte au pirate informatique c'est dans un but de sensibilisation que TPassword s'inscrit dans ce contexte. En effet en donnant une mesure objective à l'utilisateur de la fragilité de son mot de passe, et en proposant un générateur de mot de passe « personnalisé », TPassword essaie du mieux qu'il puisse d'aider l'utilisateur à se rendre compte de sa vulnérabilité et de l'accompagné dans la résolution de cette vulnérabilité.

Les objectifs de TPassword sont de donner une mesure objective de la robustesse du mot de passe, de générer un mot de passe en fonction d'entrée utilisateur, d'être accessible facilement. Afin de répondre à ces objectifs plus précisément, TPassword devra donner une estimation du temps avant le craquage du mot de passe par force brute ou par attaque par dictionnaire de mot de passe ; de permettre à l'utilisateur de créer des mots de passe en choisissant des critères tel que la présence ou non de certain type de caractère ainsi que la longueur du mot de passe souhaité ; enfin la possibilité d'accès sur navigateur web voir sur internet.

Le projet se déroulant sur un temps limité, et au vu des connaissance limité de l'équipe en termes de développement d'application web (débutant), la sécurisation du site ne sera pas prise en compte dans le développement. On considéra aussi qu'un utilisateur n'est que de passage et que donc la création de compte pour l'utilisateur est inutile.

IV. ANALYSE DES BESOINS

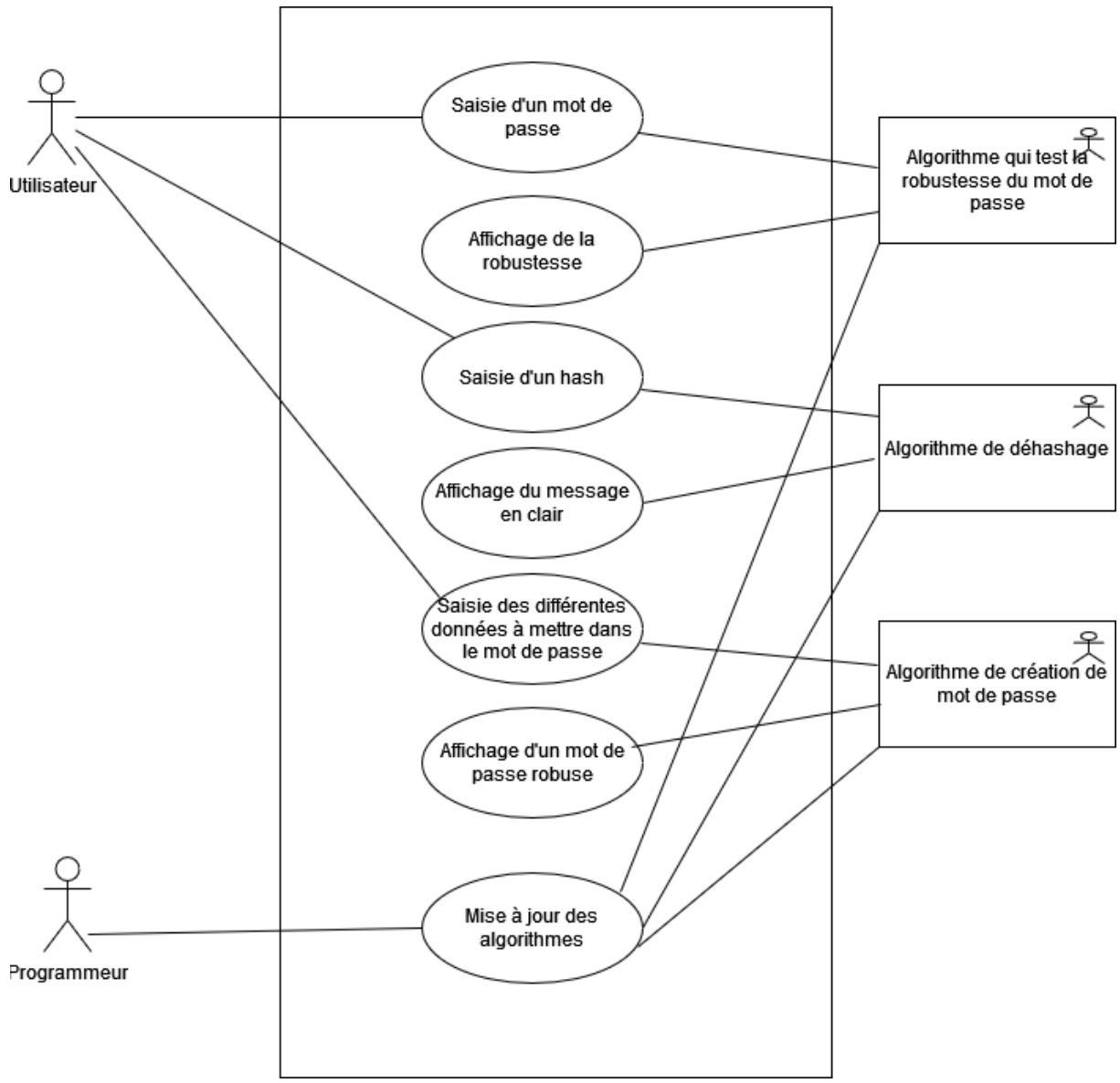
Les utilisateurs pourraient être de tout bord et milieu, le but de ce site est de sensibiliser un publique le plus large possible et d'être donc le plus accessible possible.

Les gestionnaires du système est dans ce cas l'équipe de développement elle-même, dans le cas de problème sur le système l'équipe devra être prête à régir et résoudre le problème

Pour de plus ample information voire le cahier des charges

1. Spécification fonctionnelle

Diagramme fonctionnelle



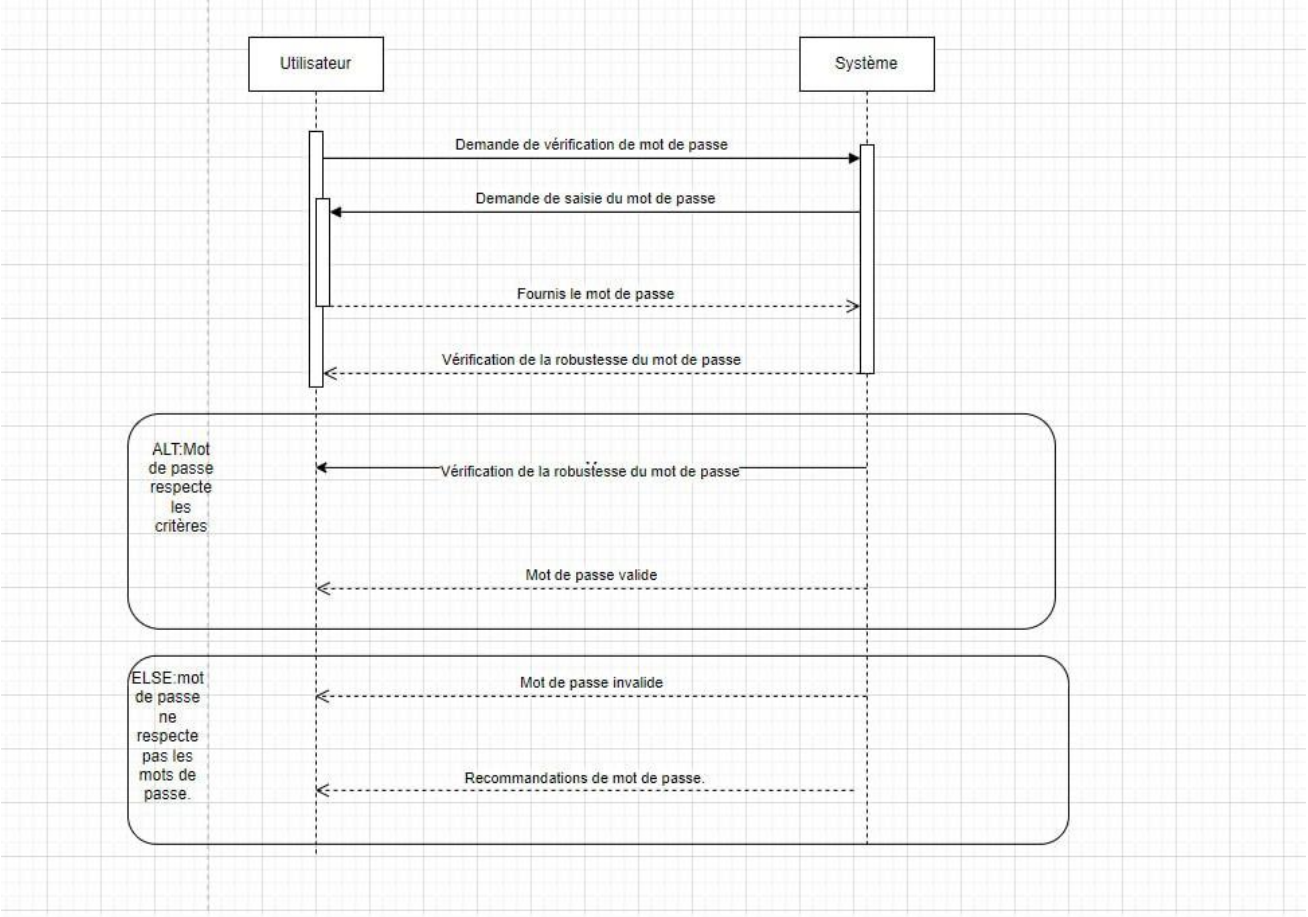
L'utilisateur ce vois offrir différent choix en arrivant sur la page, il doit pouvoir faire des action tel que tester un mot de passe, créée un mot de passe, aller voir la FAC etc. Le

programmeur ici l'équipe de développement assure les correctif et maintienne en place les algorithme.

CAS D'UTILISATION

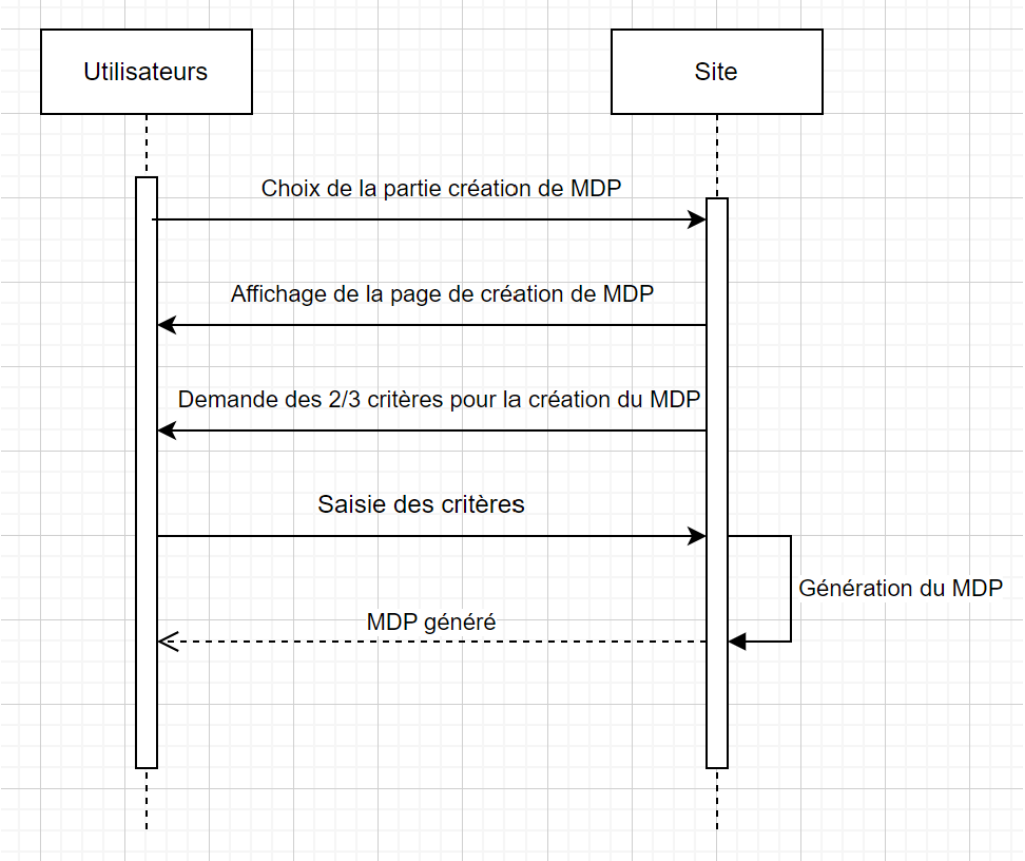
Test du mot de passe

Arrivé sur le site l'utilisateur peut entrer son mot de passe dans un encadré prévu à cet effet afin de teste son mot de passe il appuie sur un bouton afin de le soumettre au site. Le système vérifie si son mot de passe n'est pas dans une liste de mot de passe qui sont les plus utilisé, si le mot de passe n'est pas dans la liste alors le système estime le temps que le mot de passe mettra à ce faire trouver par une attaque par force brute. Les opérations doivent être presque instantané, sans aucun temps de chargement ou d'attente pour l'utilisateur.



Création de mot de passe

L'utilisateur entre des préférences afin de générer son mot de passe comme le nombre de caractère, la présence de chiffres, de majuscules, de caractères spéciaux. Le système à partir de ces informations génère un mot de passe, qui est générer aléatoirement. Cette fonctionnalité doit prendre le moins de temps possible, au maximum quelque seconde.



2. Spécification non fonctionnelle

SECURITE

D'un point de vue sécuritaire, un site proposant ce type de service doit bien évidement chiffré la communication entre le server et le client en SSL/TLS afin de mettre en œuvre le protocole HTTPS. Dans un second temps si le site venait à subir une cyber-attaque il devrait être strictement impossible de retrouver les mots de passe des utilisateurs pour cela aucun log des entrées des utilisateur ne sera gardé.

UI/UX

L'interface doit inspirer confiance on mettra un point d'honneur au fait que les informations rentrées par l'utilisateur ne seront jamais gardées par le système, les couleurs bleus et une base sur un thème sombre seront à privilégier.

V. PLANIFICATION

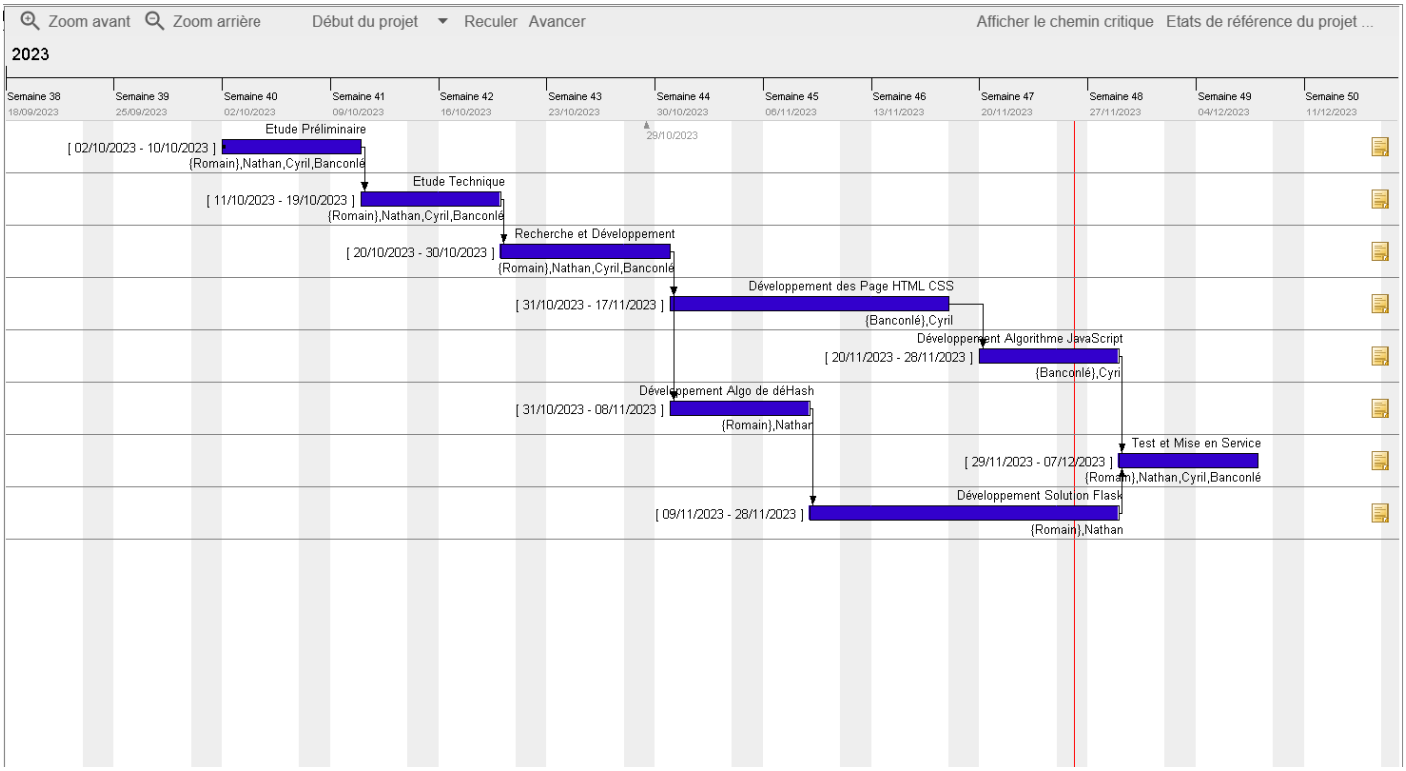
1. Etapes du projet

Tableau récapitulatif

Nom de l'étape	Description	Temps	Personnels
Etude préliminaire	Trouver le projet et commencé à réfléchir aux solutions Réalisation de la planification	1 séance / 1 semaine	Tout le monde
Etude technique	Crée le cahier des charges Création définitive de la planification Trouver les technologies à appliquer	1 séance / 1 semaine	Tout le monde
Recherche et Développement	Perfectionnement et apprentissage des technologie retenue Recherche sur les méthodes afin de craquer un mot de passe Recherche de service similaire pour le design	1 séance / 1 semaine	Tout le monde
Développement FRONTEND	Développement des page HTML, réalisation de feuille de style CSS et de quelque que script JavaScript	2 séances / 2 semaines	Cyril Bourlet Bancolé Akobi
Développement Algorithme	Développement d'algorithme d'attaque par dictionnaire et d'estimation du temps de craquage d'un mot de passe	1 séance / 1 semaine	Romain Debreu Nathan Delpech
Développement d'algorithme FRONTEND	Développement d'algorithme JavaScript afin d'amélioré l'expérience utilisateur	1 séance / 1 semaine	Cyril Bourlet Bancolé Akobi
Développement de la solution Flask	Développement de la solution Flask	2 séances / 2 semaines	Romain Debreu Nathan Delpech
Fusion BACK-FRONT Et TEST	Fusion du frontend et backend intégration des pages sur la solution flask Ainsi que test de toute les fonctionnalité	1 séance / 1 semaine	Tout le monde
Déploiement (Non planifié)	Déployé chez un hébergeur l'application web	0 séance	Romain Debreu

Voici un tableau représentant les principale phase du projet

Diagramme de Gant



2. Ressources

TABLEAU DES RESSOURCES HUMAINE

Romain DEBREU	**Responsable/ Développeur**	**Equipe Backend**
Nathan DELPECH	**Développeur**	**Equipe Backend**
Bancolé AKOBI	**Développeur**	**Equipe Frontend**
Cyril BOURLET	**Développeur**	**Equipe Frontend**

Voici un tableau des ressources humaine disponible sur le projet avec le nom le prénom leur rôle et leur équipe d'affectation

RESSOURCE FINANCIERE ET MATERIELLE

A disposition : plateforme GitHub, ordinateur portable personnel, sale de INSA/UPHF

Budget alloué : 0 €

3. Critère d'acceptation

Le site doit pouvoir être exécuter en local, et au moins tester les mots de passe en donnant une approximation vague du temps pour le craquer, sur une page agréable à regarder

4. Risques

Risque	Gestion du risque
Impossibilité d'accès au locaux	Travail en remote
Maladie d'un membre de l'équipe	Réorganisation des équipe et allongement de certaine tache
Formation insuffisante	Réduction des objectif
Effacement du code	Plan de sauvegarde

Voici un tableau représentant le risque identifié et leur gestion

VI. COMPTE RENDU - ROMAIN DEBREU

1. Introduction

Les taches qui m'ont été confié fus les suivantes, réalisation d'un algorithme d'estimation du temps de craquage d'un mot de passe, la réalisation du code en backend (solution python Fask), une partie de la fusion backend et frontend, la mise en œuvre du déploiement chez un hébergeur du site (si possible) ; ainsi que la gestion de l'équipe, notamment la réparation de tache et la planification.

2. Analyse

Tache	Scénario	Point critique
Algorithme d'estimation de craquage d'un mot de passe	L'utilisateur entre un mot de passe sur la page Le site revoie une estimation du temps avant de craquer le mot de passe	Donné une estimation réaliste du temps avant de craquer un mot de passe
Réalisation du code server	L'utilisateur peut naviguer de manière simple sur le site La réalisation de l'algorithme d'estimation du temps de craquage du mot de passe est réalisée sur en backend et revoie sur la page	Formation sur la solution backend choisis Sécurité (non essentiel mais appréciable) Résolution des bugs
Déploiement chez un hébergeur	Déployer le site complet chez un hébergeur afin que celui-ci soit disponible sur internet	Trouver un hébergeur peu onéreux permettant l'exécution de la solution backend

Tableau récapitulatif des taches à réaliser

Recherche de solution :

Framework et langage de programmation possible pour le backend :

Flask, Django, PHP, Nodejs.

Solution retenue : Flask pour sa facilité d'apprentissage

Possibilité d'hébergeur pour :

OVH, AWS, Azure, PythonAnyWhere

Solution retenue : PythonAnyWhere pour sa facilité d'utilisation, sont coût, l'exécution de python et notamment du Framework Flask

3. Conception :

Algorithme d'estimation de temps de craquage

On se trouve dans un cas simple de combinatoire en mathématique. En effet afin d'estimer le temps de craquage d'un mot de passe il faut connaitre le nombre d'opération maximal que devrait réaliser un ordinateur avant trouver le mot de passe. Pour cela on réalise qu'il s'agit du p-permutation ou p est égale au nombre de lettre du mot de passe. Pour trouver ce résultat on fait donc le nombre de possibilité (nombre de lettre possible) exposant le nombre de lettre dans le mot. Afin d'obtenir le temps on prend ce résultat et on le divise par le nombre d'opération qu'il peut effectuer en sur une seconde (ici le nombre de lettre tester).

Attention des choix ont été réaliser lors de la réalisation de cet algorithme, en effet on à choisi que le nombre de possibilité serait déterminé par le type de caractère présent dans le mot (présence de : minuscule, majuscule, chiffre, caractère spéciaux). Nous avons choisi de même d'estimer le temps pour un ordinateur classique et non un super calculateur. Nous avons décider de nous basé sur cette source afin de trouver les paramètre de l'équation que nous effectuons : https://patrowl.io/fr/le-tableau-de-la-resistance-des-mots-de-passe/

Psudo-code :

```
Chaine mdp = recup(mdp);

int len = mdp.longueur;

int min = 0 ;

int maj = 0 ;

int num = 0 ;

int spe = 0 ;

int pos = 0 ;

int temp = 0 ;

pour i de 0 à len pas de 1

{

   Si mdp[i] in "abcdefghijklmnopqrstwxyz"

   Alors min = 1

   Si mdp[i] in "ABCDEFGHIJKLMNOPQRSTUVWXYZ"

   Alors maj = 1 ;

   Si mdp[i] in "0123456789"
```

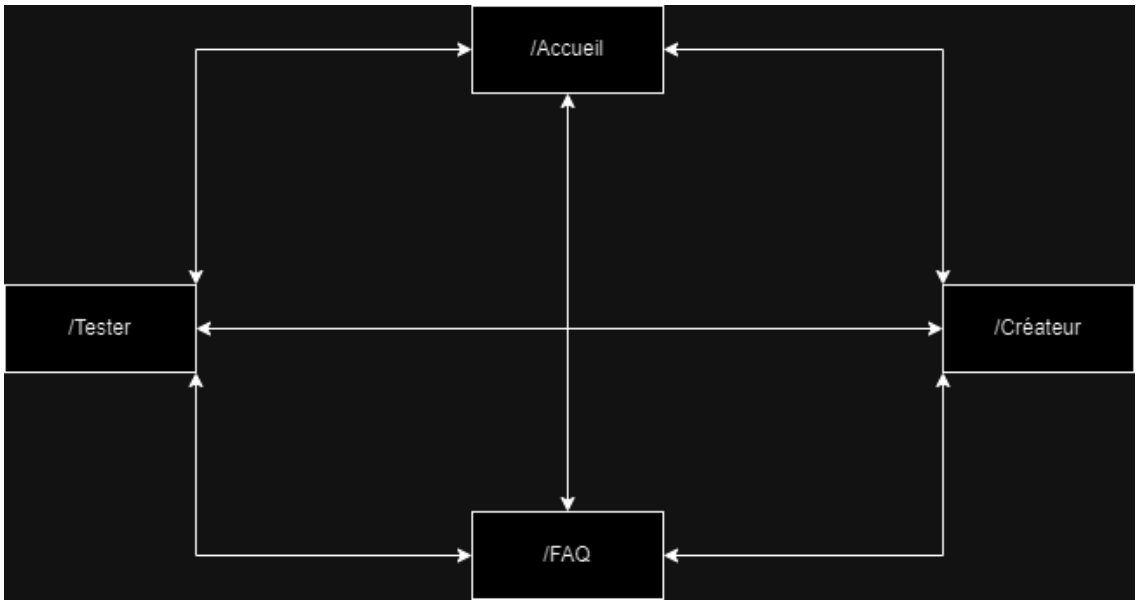
```
    Alors num = 1 ;
    Si mdp[i] in "~@#_^¨*%/.+:;=§!" // Pour l'instant je
    spe = 1 ;
}
Si(min = 1) Alors pos += 26;
Si(maj = 1) Alors pos += 26;
Si(num = 1) Alors pos += 10;
Si(spe = 1) Alors pos += 16;

temps = pos ^ len / 1_000_000
retourne temps
```

Code de la solution backend

Nous rappelons que la solution choisit est Flaks donc que la solution est écrite en python

Architecture du site :



4. Implémentation

Réalisation de plusieurs fonctions liées aux décorateurs, représentant les pages. Réalisation d'un bloque d'HTML intégrer à toutes les pages. Compartimentation du code, les fonctions principales sont disposées dans un module python crée pour l'occasion. Pour le déploiement création d'une adresse mail spécifique, transfert des fichiers chez l'hébergeur création d'un compte sur PyrhonAnyWhere.

Plusieurs problèmes ont été rencontrer, le retour de la fonction crée afin de déterminer le temps de craquage, renvoyé le temps en seconde ce qui est incompréhensible pour l'utilisateur ; solution réalisation d'une fonction de conversion du temps en seconde en minute, heure, mois, année. Difficulté mineure lors de la réalisation du bloc de code html lié a une mauvaise connaissance du Framework Flask (Erreur bête lié à l'oublie d'une ligne de code) ainsi qu'une mauvaise compréhension entre l'équipe front et back sur le fichier attendu ; solution a été d'ajouté les ligne manquante et de revoir les fichier frontend. Enfin problème dans le code server lors du déploiement, le chemin du fichier contenant la liste des mots de passes pour l'attaque par dictionnaire été en relatif ce qui ne fonctionnais pas ; la solution trouvée a été de mettre le chemin en absolu.

5. Test

Stratégie de test pour l'algorithme d'estimation du temps, test avec différente valeur d'entré et comparaison avec les données présentes chez la source (dans le tableau).

Stratégie de test pour la fonction de conversion, test avec plusieurs entrée et comparaison avec le résultat attendu.

Stratégie de test pour le code backend et déploiement, test d'accès, test de toute les fonctionnalité et comparaison avec le résultat attendu.

Les tests ont été effectué à la main

6. Discussion

Les choix effectués ont été en cohérence avec les contraintes, une monté en compétence sur le Framework Flask est à noter. Les algorithmes développés sont efficaces et fonctionne correctement. Si le site venait à être plus connu ou à durer dans le temps, un audit de sécurité serait nécessaire. En effet de nombreuse failles doivent être présentes néanmoins leur répercussion resterais limité pour l'utilisateur, étant donné qu'il n'y a aucune base de données. Aucune limitation n'est trouvée même si d'autre fonctionnalité prévue dans le cahier des charges a été écarté pour leur non-pertinence comme une fonctionnalité de hachage et de déhachage. Comparé aux solutions existantes, un nom de domaine propre pourrait être un plus, sinon les fonctionnalités proposées sont similaires à celle déjà présente sur internet, et parfois plus avancé avec la présence de l'attaque par dictionnaire de mot de passe qui n'est pas présent sur tous les sites a priori.

VII. COMPTE RENDU – NATHAN DELPECH

1. Introduction :

Les tâches qui m'ont été confiée étaient tout d'abord de réaliser un algorithme de test de mot de passe par dictionnaire en python, un algorithme de hashage de mot de passe et une partie de la fusion entre frontend et backend avec Flask.

2. Analyse :

Tâches	Scénario	Situation Finale
Algorithme d'attaque par dictionnaire	L'algorithme vérifiera si le mot de passe saisi est dans une liste des 10 000 000 de mots passes les plus utilisés.	Si le mot de passe est dans la liste, il est alors trouvé instantanément.
Algorithme de hashage de mot de passe	L'algorithme permet de hasher un mot de passe saisi par l'utilisateur, mais également de hasher tous les mots de passes de la liste des 10 000 000 de mots passes selon la méthode choisie.	Non implémenté dans le projet car inutile dans la version finale.
Fusion entre frontend et backend avec Flask	Après avoir fini l'interface du site Web et les différents algorithmes, nous avons combiné les deux pour obtenir la version finale.	Obtenir un site Web de testeur et de création de mots de passes fonctionnel.

Recherches de solutions :

Nous aurions pu programmer les algorithmes en javascript pour les intégrer au fichier HMTL, mais nous avons finalement décidé de le faire sur Python avec Flask pour que les algorithmes s'effectuent du côté serveur. Car, pour l'attaque par dictionnaire, il est plus compliqué d'importer une liste de 10 000 000 de mots de passes sur le poste de l'utilisateurs que d'effectuer les tests sur le serveur.

3. Conception :

Pour l'attaque par dictionnaire sur le mot de passe saisi, nous avons décidé une liste répertoriant les 10 000 000 de mots de passes les plus utilisé dans le monde :

https://github.com/danielmiessler/SecLists/blob/master/Passwords/Common-Credentials/10-million-password-list-top-1000000.txt

Pour l'algorithme de hashage de mots de passe, il était question de pouvoir hasher un mot de passe grâce à la bibliothèque hashlib.py et d'incorporer les différentes méthodes dans une fonction laissant le choix aux utilisateurs.

4. Implémentation :

Nous souhaitions implémenter l'algorithme de hashage de mot de passe dans une autre page Web. Le problème étant que nous voulions faire un hashage et un déhashage grâce à un dictionnaire. Nous aurions donc dû avoir une liste de hash avec leur version déhashé pour pouvoir comparer le hash saisi par l'utilisateur avec la liste pour trouver une potentiel similitude. Evidemment, cela n'aurait pas fonctionné à tous les coups car la base d'un algorithme de hashage est de ne pas pouvoir déhasher le résultat. Nous avons abandonné le projet de faire cette partie du site Web par manque de temps et de compréhension du sujet.

5. Tests :

Test de l'algorithme d'attaque par dictionnaire fait « à la main » en saisissant différents mots de passes compris dans la liste pour vérifier le bon fonctionnement.

Test de l'algorithme de hashage en saisissant un mot de passe pouvant être déhasher (grâce à https://crackstation.net/ par exemple), et en vérifiant l'exactitude du hash.

Test du site complet après la fusion entre le backend et le frontend.

6. Discussion :

Le site fonctionne comme nous l'avions prévu à la base, exceptés la partie hash et dehash. Il n'y a pas de réelle limitation si ce n'est dans la création d'un mot de passe car il n'est pas aussi personnalisable que ce que l'on souhaitait à la base.

Malgré l'existence de testeur de mots de passes en lignes, notre site à la particularité de pouvoir générer des mots de passes personnalisables résistant.

VIII. COMPTE RENDU – BANCOLE AKOBI

1. Introduction

Le présent rapport documente le développement du frontend pour une application de création de mots de passe, de test, et un FAQ. En tant que membre de l'équipe frontend, je me suis occupé en grande partie de la conception visuelle, la création du logo, la mise en œuvre du test de mots de passe, la réalisation de la moitié de la page de test, et la création du FAQ. L'ensemble du projet a impliqué l'utilisation de HTML, JavaScript, bibliothèques JavaScript externes, CSS, et l'application Canva pour le traitement d'images.

2. Phase d'Analyse et de Recherche:

Identification des scenarios et cas d'utilisation

Pour définir les fonctionnalités de l'application, nous avons identifié les scénarios typiques et critiques.

Scénario	Description
Création de Mot de Passe	L'utilisateur peut générer un mot de passe en spécifiant la longueur et les critères désirés.
Test de Robustesse du Mot de Passe	Le système doit évaluer la robustesse du mot de passe généré en fonction de certains critères(faite bien-sûr par la partie backend) mais solution javascript moins pertinente existante
Utilisation du FAQ	Les utilisateurs peuvent accéder au FAQ pour obtenir des réponses à leurs questions.
Conception Visuelle et Logo	Le processus de conception visuelle, y compris la création du logo, doit être pris en charge
Traduction de la Page	Une fonctionnalité de traduction de la page doit être mise en œuvre. (Anglais et Français)

Recherche de solutions et choix technologiques

Technologie	Description
HTML	Langage de balisage pour la structure de la page web.
JavaScript	Langage de programmation utilisé pour la logique de l'application.
Bibliothèques JavaScript	*https://unpkg.com/feather-icons* *https://translate.google.com/translate_a/element.js?cb=googleTranslateElementInit*
CSS	Langage de style pour la mise en forme et la présentation de la page.
Application Canva	Utilisation de Canva pour la conception graphique, recadrage et création d'images.

3. Conception

La conception de l'application a été guidée par la nécessité d'offrir une interface utilisateur intuitive, une génération de mots de passe robustes et une expérience globale positive. Les principaux aspects de la conception comprennent l'architecture logicielle, la conception de l'interface utilisateur, et l'intégration du logo.

Pour garantir la maintenabilité et l'extensibilité de l'application, nous avons adopté une architecture de type Modèle-Vue-Contrôleur (MVC). Cette approche a permis de séparer clairement la logique métier, la gestion des données, et la présentation.

Tableau d'Architecture Logicielle

Composant	***Description***
Modèle	*Gestion des données relatives à la génération de mot de passe , y compris les critères de l'utilisateur .*
Vue	*Affichage de l'interface utilisateur et intégration du logo.*
Contrôleur	*Gestion de la logique métier, notamment la génération de mots de passe et la traduction de la page.*

Conception de l'Interface Utilisateur

La conception de l'interface utilisateur a été réalisée en utilisant HTML et CSS pour définir la structure et le style de la page. L'utilisation de Canva a été intégrée pour créer le logo et

ajuster les éléments graphiques de manière à garantir une esthétique cohérente.

Le logo, conçu pour refléter la sécurité et la robustesse, a été intégré de manière à renforcer l'identité visuelle de l'application. Les choix de couleurs, de polices, et de disposition ont été pensés pour une expérience utilisateur optimale

4. Implémentation

L'implémentation a été réalisée en utilisant JavaScript pour la logique de l'application. La génération de mots de passe robustes a été mise en œuvre en tenant compte des critères de longueur et des types de caractères souhaités par l'utilisateur.

Génération de Mot de Passe

Le script JavaScript fourni illustre le processus de génération de mot de passe. La longueur du mot de passe est déterminée par l'utilisateur, qui peut également choisir d'inclure des caractères spéciaux, des chiffres, et des lettres majuscules/minuscules. Un soin particulier a été apporté pour garantir la robustesse du mot de passe généré, évaluant sa force en fonction de critères prédéfinis.Sans oublier une fonction de copie servant à l'utilisateur pour la copie de son mot de passe.

Proposition de testeur de mot de passe utilisable mais moins pertinent

```
function evaluerMotDePasse() {
  const motDePasse = document.getElementById('password').value;

  // Longueur du mot de passe
  const longueur = motDePasse.length;

  // Complexité en se basant sur le type de caractères utilisés
  const complexite = calculerComplexite(motDePasse);

  // Calcul de la robustesse en pourcentage
  const tempsEstime = calculerTempsEstime(longueur, complexite);
  const robustesse = calculerRobustesse(tempsEstime);

  // Arrondir le temps estimé en jours à l'entier le plus proche
  const tempsEstimeEnJoursArrondi = Math.round(tempsEstime / (1000 * 60 * 60 * 24));

  return {
    robustesse: Math.min(robustesse, 100).toFixed(2), // La robustesse ne peut pas dépasser 100%
    tempsEstime: tempsEstime,
```

```
    tempsEstimeEnJours: tempsEstimeEnJoursArrondi,
    couleur: getColorForRobustness(robustesse)
  };
}

function calculerComplexite(motDePasse) {
  const majuscules = /[A-Z]/g;
  const minuscules = /[a-z]/g;
  const nombres = /[0-9]/g;
  const speciaux = /[!@#$%^&*()_+\-=\[\]{};':"\\|,.<>\/?]+/;

  let scoreComplexite = 0;

  // Vérification des majuscules, minuscules, chiffres et caractères spéciaux
  if (motDePasse.match(majuscules)) scoreComplexite += 0.25;
  if (motDePasse.match(minuscules)) scoreComplexite += 0.25;
  if (motDePasse.match(nombres)) scoreComplexite += 0.25;
  if (motDePasse.match(speciaux)) scoreComplexite += 0.5; // Caractères spéciaux sont considérés plus complexes

  return scoreComplexite;
}

function calculerTempsEstime(longueur, complexite) {
  // Temps estimé pour pirater en se basant sur la capacité d'un ordinateur
  // Formule simplifiée, à adapter en fonction de la puissance de l'ordinateur
  const tempsEstime = Math.pow(10, longueur + complexite);

  return tempsEstime;
```

```
}

function calculerRobustesse(tempsEstime) {
  // Arbitrairement ajusté pour mieux correspondre à la réalité
  const MAX_TEMPS_ESTIME = Math.pow(10, 10); // Maximum de temps estimé, par exemple 10^10

  // Calcul de la robustesse en pourcentage
  return Math.min(100, (tempsEstime / MAX_TEMPS_ESTIME) * 100); // La robustesse ne peut pas dépasser 100%
}

function getColorForRobustness(robustesse) {
  if (robustesse <= 20) {
    return 'red';
  } else if (robustesse <= 40) {
    return 'orange';
  } else if (robustesse <= 60) {
    return 'yellow';
  } else if (robustesse <= 80) {
    return 'greenyellow';
  } else {
    return 'green';
  }
}

const passwordInput = document.getElementById('password');
const resultElement = document.getElementById('result');
```

```
passwordInput.addEventListener('input', () => {
  const evaluation = evaluerMotDePasse();

  const tempsEstimeEnSecondes = evaluation.tempsEstime / 1000;
  const tempsEstimeEnMinutes = tempsEstimeEnSecondes / 60;
  const tempsEstimeEnHeures = tempsEstimeEnMinutes / 60;
  const tempsEstimeEnJours = tempsEstimeEnHeures / 24;
  const tempsEstimeEnMois = tempsEstimeEnJours / 30; // Estimation approximative d'un mois
  const tempsEstimeEnAnnees = tempsEstimeEnMois / 12; // Estimation approximative d'une année

  let tempsEstimeAffichage;

  if (tempsEstimeEnAnnees >= 1) {
    tempsEstimeAffichage = `environ ${Math.round(tempsEstimeEnAnnees)} an${tempsEstimeEnAnnees >= 2 ? 's' : ''}`;
  } else if (tempsEstimeEnMois >= 1) {
    tempsEstimeAffichage = `environ ${Math.round(tempsEstimeEnMois)} mois`;
  } else if (tempsEstimeEnJours >= 1) {
    tempsEstimeAffichage = `environ ${Math.round(tempsEstimeEnJours)} jour${tempsEstimeEnJours >= 2 ? 's' : ''}`;
  } else if (tempsEstimeEnHeures >= 1) {
    tempsEstimeAffichage = `environ ${Math.round(tempsEstimeEnHeures)} heure${tempsEstimeEnHeures >= 2 ? 's' : ''}`;
  } else if (tempsEstimeEnMinutes >= 1) {
    tempsEstimeAffichage = `environ ${Math.round(tempsEstimeEnMinutes)} minute${tempsEstimeEnMinutes >= 2 ? 's' : ''}`;
  } else {
    tempsEstimeAffichage = `environ ${Math.round(tempsEstimeEnSecondes)} seconde${tempsEstimeEnSecondes >= 2 ? 's' : ''}`;
```

```
 }


 // Mettez à jour les éléments HTML correspondants
 resultElement.innerHTML = `
  Robustesse du mot de passe : <span style="color: ${evaluation.couleur}; font-weight: bold;">${evaluation.robustesse}%</span>
  <br>
  Temps estimé pour pirater : ${tempsEstimeAffichage}`;
});
```

L'ensemble de ces éléments démontre une approche réfléchie et approfondie dans la conception et l'implémentation de l'application de génération de mots de passe. L'architecture logicielle solide, l'interface utilisateur attrayante, et les fonctionnalités avancées contribuent à offrir une solution complète et bien pensée.

5. Tests Unitaires et Validation

Strategie de tests

La stratégie de tests a été élaborée pour garantir la fiabilité et la stabilité de l'application. Elle comprend des tests unitaires pour chaque composant et des scénarios de tests couvrant l'ensemble des fonctionnalités. Des cas de tests ont été conçus pour évaluer la génération de mots de passe, la robustesse des mots de passe, et la fonctionnalité de traduction.

Cas de tests et scenarios

a. Test de Génération de Mot de Passe :

- Vérifie que le mot de passe généré respecte la longueur spécifiée par l'utilisateur.

- S'assure que les caractères spéciaux, les chiffres, et les lettres sont inclus selon les choix de l'utilisateur

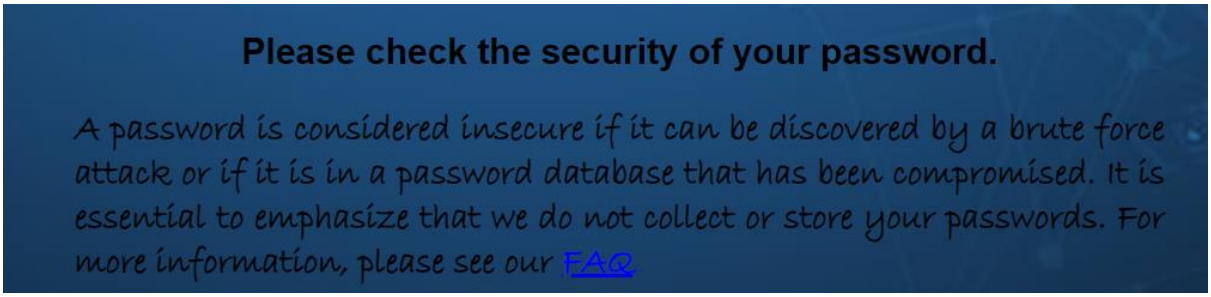
.

Exemple:

b. Test de Traduction de la Page :

- *Vérifie que le changement de langue fonctionne correctement.*
- *S'assure que le contenu de la page est traduit de manière cohérente.*

Exemple:



c-Résultats et Interprétation

Les tests unitaires ont confirmé que la génération de mots de passe respecte les critères spécifiés par l'utilisateur, tandis que le test de robustesse a montré que les mots de passe générés sont conformes aux standards de sécurité. La fonction de traduction a été validée avec succès, assurant une expérience multilingue sans faille.

d-Bugs Identifiés et Résolution

Aucun bug majeur n'a été identifié lors des tests, démontrant une mise en œuvre robuste. Cependant, des ajustements mineurs ont été apportés pour améliorer la convivialité de l'interface utilisateur en fonction des retours des tests.

e-Automatisation des Tests

L'automatisation des tests a été envisagée pour les scénarios critiques, mais une approche manuelle a été privilégiée étant donné la taille modérée de l'application. Cela a permis une flexibilité accrue pour s'adapter aux changements rapides.

6. Tests d'Intégration (Collectif):

Le site fonctionne correctement en associant la partie frontend effectuée avec Cyril Bourlet et la partie Backend avec contenant 4 pages qui ont longtemps été modelées par les 4 membres de l'équipe pour arriver à ce résultat final. Les tests d'intégration ont été essentiels pour garantir une collaboration efficace entre les différents composants de l'application. Des simulations de scénarios d'utilisation réelle ont été effectuées pour évaluer la cohérence et la fluidité des interactions entre les fonctionnalités.

7. Discussion

Retour sur les choix effectues

Les choix technologiques ont été validés par les tests et ont démontré leur adéquation pour le développement de l'application. L'utilisation de Canva pour la conception graphique a permis une intégration harmonieuse du logo et une esthétique visuelle attrayante.

Limitations du logiciel actuel

Bien que l'application réponde aux besoins spécifiés, des améliorations potentielles pourraient inclure une plus grande variété d'options de génération de mots de passe ou l'intégration de méthodes avancées de test de robustesse.

Comparaison avec d'autres solutions existantes

L'application se distingue par sa simplicité d'utilisation et son approche intuitive de la génération de mots de passe. Comparée à d'autres solutions, elle offre une expérience utilisateur conviviale sans compromettre la sécurité.

6. SITOGRAPHIE :

https://dyma.fr/javascript?campaignId=9196007966&device=c&utm_source=google&gad_source=1&gclid=Cj0KCQiAyeWrBhDDARIsAGP1mWRigg-HGBlcJ60DsBZ8wCc5ek-ij-thCruKYSNXL9P-V_PW__5BGREaAjL_EALw_wcB

https://www.generateur-motdepasse.com

https://www.motdepasse.xyz

https://inforisque.fr/fiches-pratiques/tester-mot-de-passe.php

https://password.kaspersky.com/fr/

https://nothing2hide.org/fr/verifier-la-robustesse-de-votre-mot-de-passe/

https://www.dashlane.com/fr/features/password-generator

Et bien d'autres !

IX. COMPTE RENDU – CYRIL BOURLET

1. Introduction

Nous avons divisé le backend et le frontend, donc étant dans l'équipe de frontend avec Bancolé Akobi pour créer des pages web, nous nous sommes mis d'accord sur ce que chacun devait faire et donc j'ai eu la tâche de créer la page d'accueil et la page test de mot de passe.

2. Analyse

Tache	Procéder	Point critique
Page d'accueil	Le but de la page d'accueil c'est qu'elle soit attirante pour l'utilisateur et de pouvoir naviguer sur les différentes page (test, générer, FAQ) pour avoir un accès facile à toutes les possibilités du site.	La page doit avoir des informations importantes et attirantes.
Page de test de mot de passe	Le but de la page Test c'est que l'utilisateur puisse savoir quels types d'attaque il peut subir pour pouvoir se protéger et donc de savoir si les mots de passes qu'il utilise sont facile à cracker en l'entrant dans la barre de saisie pour que le backend le vérifie et renvoie le temps estimé.	Il faut être clair sur les informations données pour que l'utilisateurs puisse comprendre facilement et qu'il ait confiance au site.

3. Conception

Diagramme de séquence page d'accueil

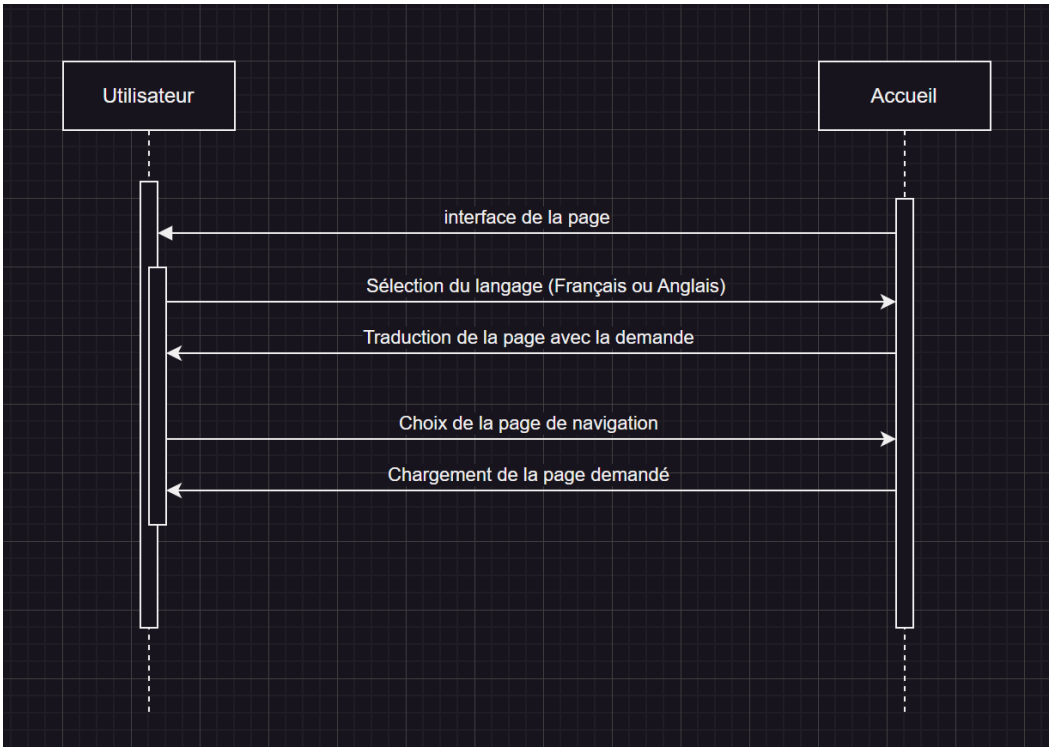
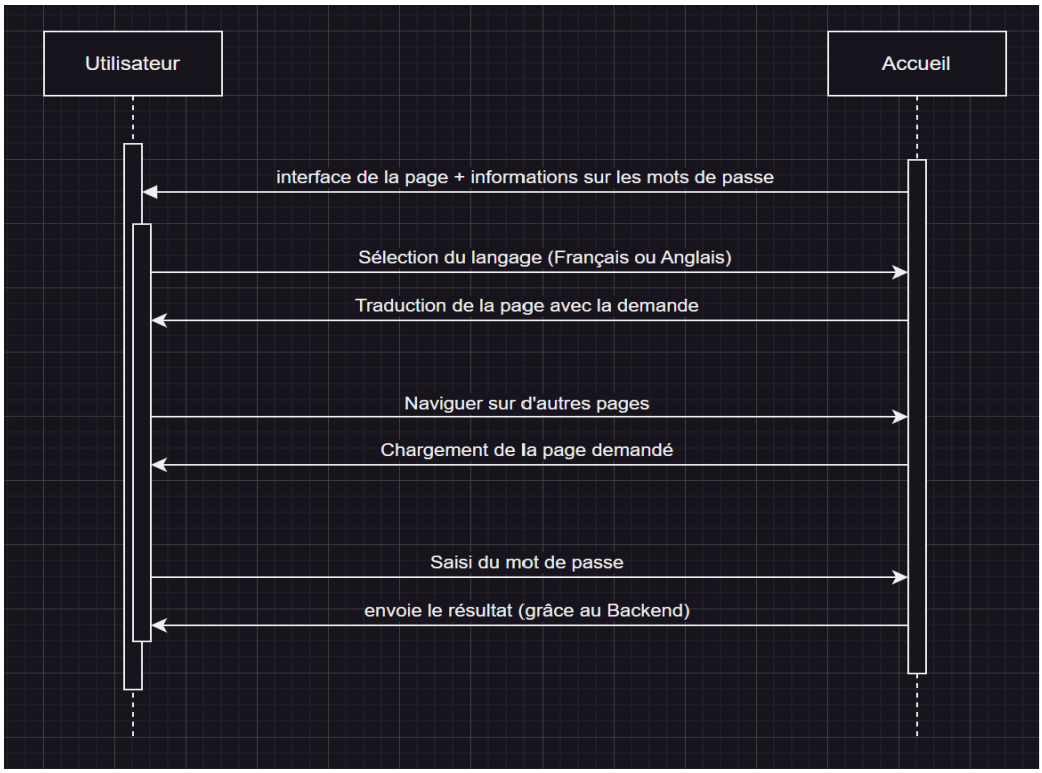


Diagramme de séquence page de test de mot de passe



Pour la page d'accueil l'utilisateur doit être attiré par le site pouvoir :

- Aller sur la page tester
- Aller sur la page générer
- Aller sur la page FAQ
- Traduire le texte en anglais

Pour la page de test de mot de passe l'utilisateur doit pouvoir :

- Naviguer sur les autres pages aussi
- Entrer son mot de passe
- Avoir le résultat (le temps estimé)
- Avoir des informations sur les différentes attaques possibles

Pour les pages web j'ai utilisé NotePad++ pour le langage HTML, CSS pour réaliser le design des pages et de leurs fonctionnalités. Nous avons utilisé aussi un peu de JavaScript par exemple pour la traduction en anglais (fonctionnalité dans la barre de navigation du site).

Pour la réalisation des pages en HTML et CSS j'ai dû faire des recherches pour savoir comment utiliser certaines balises et pouvoir comprendre comment fonctionnaient les tableaux, etc…. Puis la difficulté principale et de faire une page propre et jolie ce qui implique l'ajustement de certaines choses comme les textes (couleur, positionnement et taille) et pour les images il ne faut pas avoir des déformations importantes pour que cela reste normale et attirant pour l'utilisateurs. Il fallait donc joué avec le design pour que le rendu soit correct.

4. Implémentation

Nous avons découpé les pages accueil, test, génération, FAQ. Nous avons mis le style des pages en CSS dans des fichiers différents pour chacune d'elles, ceci aide à la compréhension dans le groupe.

D'abord nous avons créé la barre de navigation du site que l'on a ensuite intégrer directement avec le backend pour éviter la répétition de toutes ces lignes.

Pour la page d'accueil j'ai utilisé un tableau que j'ai divisé en plusieurs colonnes et j'ai créé des lignes avec une taille différentes pour ajuster les images et le texte comme je le souhaité.

Pour la page de test j'ai d'abord créé un tableau dans l'entête pour pouvoir insérer son mot de passe dans la barre de saisie et pouvoir l'envoyer dans un bouton pour l'estimation du temps de craquage avec le backend. En dessous du tableau j'ai mis les différentes attaques possibles que l'utilisateur peut subir avec un schéma explicatif à chaque fois.

5. Test

Nous avons testé chaque fonctionnalité des pages.

Pour la page d'accueil il faut pouvoir naviguer facilement avec les autres pages. Ceci a été vérifié par l'équipe puis ensuite par le groupe quand le site a été déployé sur internet.

Pour la page de test la fonctionnalité principale était que l'utilisateur entre son mot de passe pour qu'il sache en combien de temps il pourrait être récupéré. Puis ensuite la navigation se déroule de la même manière pour toutes les pages donc la page test répond aussi aux attentent de l'utilisateur et u groupe.

6. Discussion

Le début de projet était le plus important en termes de discussion pour savoir ce que chacun attendait des autres et pour savoir comment se déroulerai ce projet.

Nous nous sommes mis d'accord d'abord sur ce que chacun devait faire.

Ensuite lors de la réalisation des pages nous avons beaucoup discuté de l'esthétique car il ne faut pas oublier que c'est un travail de groupe et qu'il faut que les pages soit au goût de toute le monde et aussi des utilisateurs puis j'ai discuté pas mal avec mon coéquipier de frontend pour savoir comment on avait procédé techniquement et tactiquement pour la réalisation de nos pages.

X. CONCLUSION

1. Bilan du projet TPassword

Le projet a été réaliser dans le délai imparti malgré les perturbations lié aux alertes à la bombe. Le site a été déployé sur un Sever d'un hébergeur. Les principales fonctions de la page fonctionnent correctement : testeur de mot de passe, créateur de mot de passe.

D'un point de vue Frontend, les objectifs d'interface utilisateur et expérience utilisateur ont été remplis, l'utilisateur peut naviguer simplement sur les pages, la charte graphique a été appliqué. Les algorithmes frontend développé s'exécute convenablement et en temps raisonnable

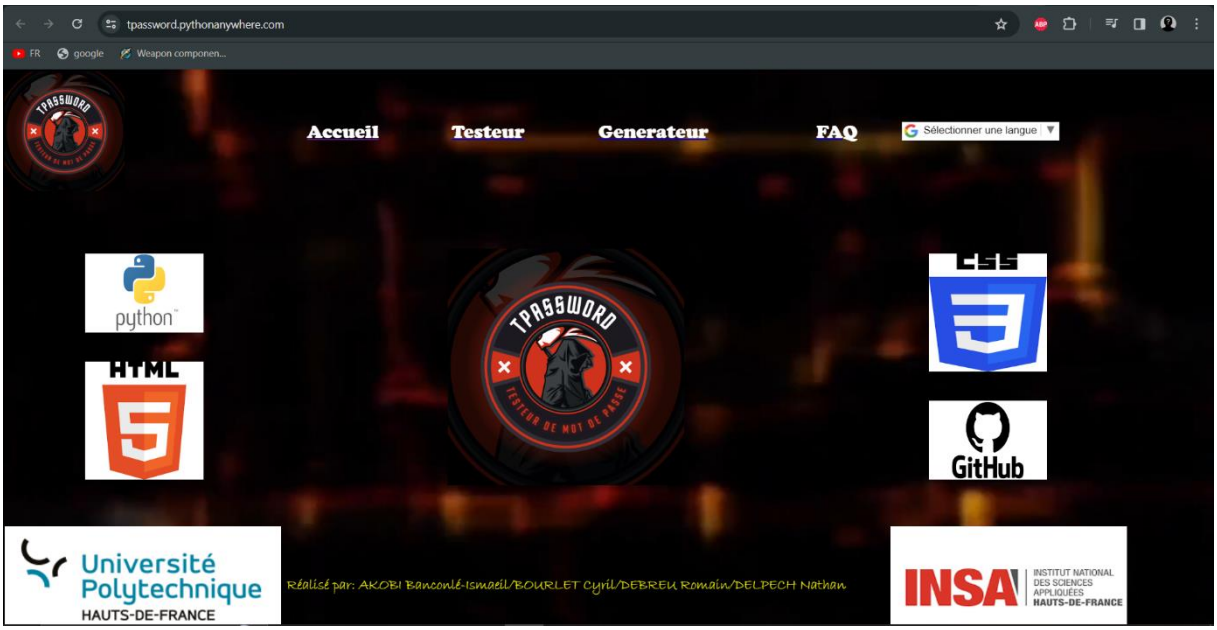
D'un point de vue backend l'algorithme s'effectue correctement, et l'interaction entre les pages fonctionne correctement. La solution Flask semble pour l'instant rester la solution la plus optimiser. Le déploiement sur l'hébergeur PythonAnyWhere c'est correctement déroulé, pour l'instant la solution d'hébergement gratuit semble être suffisante. Attention nous attirons l'attention du lecteur sur le fais que ce site restera en ligne jusqu'au 15/01/2024

En définitive, le projet a été réaliser avec succès et sans retard, en accomplissant les objectifs donnés dans le cahier des charges

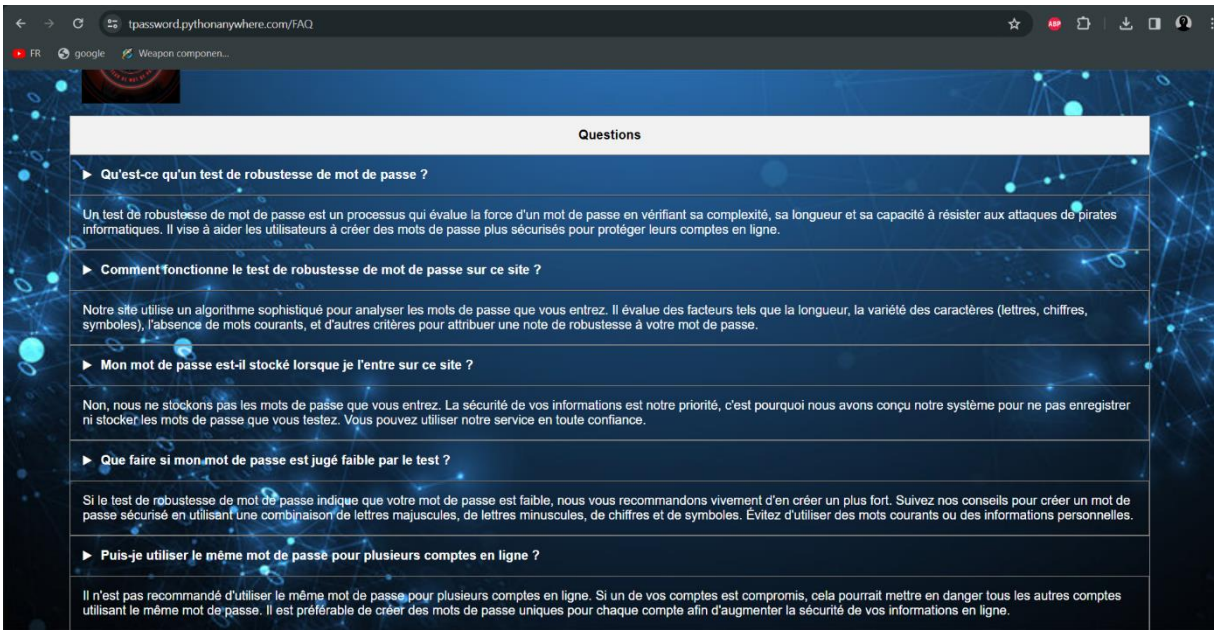
2. Annexe

GitHub : https://github.com/Nephadeo/TPassword

Capture d'écran du site :




Accueil
Testeur
Generateur
FAQ
Sélectionner une langue
Générateur de Mots de Passe
Inclure caractères spéciaux
Inclure des chiffres
Inclure des lettres
Longueur du Mot de Passe
12
Générer un Mot de Passe
Copier
Assurez-vous de stocker votre mot de passe généré dans un endroit


tpassword.pythonanywhere.com/FAQ
Questions
Qu'est-ce qu'un test de robustesse de mot de passe ?
Un test de robustesse de mot de passe est un processus qui évalue la force d'un mot de passe en vérifiant sa complexité, sa longueur et sa capacité à résister aux attaques de pirates informatiques. Il vise à aider les utilisateurs à créer des mots de passe plus sécurisés pour protéger leurs comptes en ligne.
Comment fonctionne le test de robustesse de mot de passe sur ce site ?
Notre site utilise un algorithme sophistiqué pour analyser les mots de passe que vous entrez. Il évalue des facteurs tels que la longueur, la variété des caractères (lettres, chiffres, symboles), l'absence de mots courants, et d'autres critères pour attribuer une note de robustesse à votre mot de passe.
Mon mot de passe est-il stocké lorsque je l'entre sur ce site ?
Non, nous ne stockons pas les mots de passe que vous entrez. La sécurité de vos informations est notre priorité, c'est pourquoi nous avons conçu notre système pour ne pas enregistrer ni stocker les mots de passe que vous testez. Vous pouvez utiliser notre service en toute confiance.
Que faire si mon mot de passe est jugé faible par le test ?
Si le test de robustesse de mot de passe indique que votre mot de passe est faible, nous vous recommandons vivement d'en créer un plus fort. Suivez nos conseils pour créer un mot de passe sécurisé en utilisant une combinaison de lettres majuscules, de lettres minuscules, de chiffres et de symboles. Évitez d'utiliser des mots courants ou des informations personnelles.
Puis-je utiliser le même mot de passe pour plusieurs comptes en ligne ?
Il n'est pas recommandé d'utiliser le même mot de passe pour plusieurs comptes en ligne. Si un de vos comptes est compromis, cela pourrait mettre en danger tous les autres comptes utilisant le même mot de passe. Il est préférable de créer des mots de passe uniques pour chaque compte afin d'augmenter la sécurité de vos informations en ligne.

3. Bibliographie / Sitographie

Formation CSS/HTML/JavScript :

https://www.youtube.com/watch?v=u5W2NWItytc&list=PLrSOXFDHBtfE5tpw0bjMevWxMWXotiSdO&pp=iAQB

https://www.youtube.com/watch?v=02Xs2ySaXcs&list=PLrSOXFDHBtfGxf_PtXLu_OrjFKt4_dqB_&pp=iAQB

Formation sur Flask :

https://www.youtube.com/watch?v=Ihp_cG7c2Rk&list=PLV1TsfPiCx8PXHsHeJKvSSC8zfi4Kvcfs

Site de référence :

https://www.motdepasse.xyz

https://generateurdemotdepasse.fr

https://dyma.fr/javascript?campaignId=9196007966&device=c&utm_source=google&gad_source=1&gclid=Cj0KCQiAyeWrBhDDARIsAGP1mWRigg-HGBlcJ60DsBZ8wCc5ek-ij-thCruKYSNXL9P-V_PW__5BGREaAjL_EALw_wcB

https://inforisque.fr/fiches-pratiques/tester-mot-de-passe.php

https://www.dashlane.com/fr/features/password-generator

https://password.kaspersky.com/fr/

https://nothing2hide.org/fr/verifier-la-robustesse-de-votre-mot-de-passe/

Test des mots de passe :

https://patrowl.io/fr/le-tableau-de-la-resistance-des-mots-de-passe/

https://github.com/danielmiessler/SecLists/tree/master/Passwords/Common-Credentials