NOM	CHEVRAY
Prénom	LAURANE
Date de naissance	28/11/1993

Copie à rendre

TP – Développeur Web et Web Mobile

Documents à compléter et à rendre

Lien du git : https://github.com/buffsum/ECF **/ git@github.com:buffsum/ECF.git**
Lien de l'outil de gestion de projet : https://tricolor-scallop-df3.notion.site/ECF-63b86d2d577e4d65b5c15ec15db683af
Lien du déploiement : https://shielded-citadel-17199-a7b955f75ade.herokuapp.com/
Login et mot de passe administrateur : login : admin@admin.fr **/ mdp : admin**

SANS CES ELEMENTS, VOTRE COPIE SERA REJETEE

Partie 1 : Analyse des besoins

1. Effectuez un résumé du projet en français d'une longueur d'environ 20 lignes soit 200 à 250 mots

Dans le cadre de mon projet de développement web pour le zoo Arcadia, j'ai travaillé avec une attention particulière sur le back-end et l'intégration de fonctionnalités interactives. Au départ, j'avais de bonnes connaissances en HTML/CSS, mais concernant le front le vrai défi pour moi a été d'utiliser Bootstrap, un framework que j'avais peu expérimenté durant ma formation.

Pour le back-end, j'ai choisi Python et Flask plutôt que PHP, car j'avais déjà pu voir les bases de PHP pendant la formation, j'ai donc voulu explorer une autre technologie. Ce choix s'est avéré judicieux, même si le développement back-end a été mon point faible en raison du manque de temps pour approfondir ce domaine pendant la formation. Mes connaissances préalables en langage C m'ont beaucoup aidé, notamment pour gérer les variables, les class et les système d'incrémentations, etc.

Ce qui m'a donné le plus de fil à retordre a été de gérer le comptage des consultations et les redondances de variables dans certaines fonctions. J'ai consacré beaucoup de temps à ces aspects, et au global j'aurais aimé avoir plus de temps pour peaufiner le projet et améliorer la mise en page.

Malgré ces difficultés, j'ai beaucoup apprécié travailler sur ce projet. L'expérience m'a permis de renforcer mes compétences en développement web et de mieux comprendre les aspects techniques liés à la création d'une application fonctionnelle et attrayante. Je pense que ce projet peut-être une bonne base pour moi pour mettre en place de nouvelle applications web encore plus performantes.

2. Exprimez le cahier des charges, l'expression du besoin ou les spécifications fonctionnelles du projet

Le projet consiste à développer une application web pour le zoo Arcadia, un parc écologique situé en Bretagne (près de la forêt de Brocéliande). Le directeur, José, souhaite que l'application reflète les valeurs d'écologie et d'indépendance énergétique du zoo. L'application s'adresse aux visiteurs, ainsi qu'aux employés du zoo, et inclut plusieurs fonctionnalités.

Les visiteurs pourront consulter les informations sur les animaux, les habitats, les services proposés par le zoo (restauration, visite guidée, petit train), et laisser des avis. Ils pourront également visualiser les horaires et contacter le zoo via un formulaire.

Pour les employés, l'application inclut un espace de gestion des services et des avis, ainsi qu'un suivi de la consommation de nourriture des animaux. Les vétérinaires disposeront d'un espace dédié pour renseigner l'état de santé des animaux et donner des recommandations sur les habitats. Enfin, l'administrateur aura la possibilité de créer et gérer les comptes utilisateurs (employés, vétérinaires), ainsi que de consulter un tableau de bord affichant des statistiques sur les consultations des animaux.

Aucune technologie n'est obligatoire pour cet ECF, à l'exception de l'utilisation d'une base de données relationnelle et non relationnelle. Un système de gestion de projet (Trello, Jira, Notion etc.) est demandé pour organiser les tâches, et l'application doit être déployée sur une plateforme comme Heroku, Vercel, etc. Le projet nécessite également la documentation technique, une charte graphique et des maquettes.

Partie 2 : Spécifications technique

1. Spécifiez les technologies que vous avez utilisé en justifiant les conditions d'utilisation et pourquoi le choix de ses éléments

Front-end

- **HTML :** Pour structurer le contenu des pages web.
- **CSS :** Pour styliser et mettre en forme les pages web. J'utilise également des media queries pour rendre le site responsive.

Justification : HTML/CSS sont des standards du web, indispensables pour toute application web.

Bootstrap : Un framework CSS pour faciliter la création de designs responsives et modernes.

Justification : m'a permis de gagner du temps en utilisant des composants pré-stylisés et en facilitant la création de designs responsives. M'évitant de passer par des étapes de maquettages trop poussées (mockups)

JavaScript : Pour ajouter des interactions dynamiques à vos pages web.

Justification : est essentiel pour ajouter des fonctionnalités interactives et améliorer l'expérience utilisateur.

Back-end

Langage de programmation et framework :

- Python
- Flask : Un micro-framework Python pour développer des applications web.
- Flask-WTF : Pour gérer les formulaires web de manière sécurisée.
- Flask-Migrate : Pour gérer les migrations de base de données.
- Flask-Mail : Pour envoyer des emails depuis l'application

Justification

- Flask est léger et flexible, ce qui le rend idéal pour des projets de petite à moyenne envergure. Il est également facile à apprendre et à utiliser.
- Flask-WTF simplifie la gestion des formulaires et améliore la sécurité en intégrant CSRF protection.

- Flask-Migrate facilite la gestion des schémas de base de données et les migrations.
- Flask-Mail permet d'intégrer facilement des fonctionnalités d'envoi d'emails.

J'ai choisi cette Stack back-end pour éviter de me perdre dans des configurations complexes. Flask est un framework minimaliste, plus simple que Django pour un premier projet, tout en étant puissant pour ce que je devais faire. Flask est bien documenté, et j'ai pu l'utiliser pour créer mes routes (pages de connexion, d'inscription, etc.) de manière simple

Base de données

- SQLite : Une base de données relationnelle légère et intégrée.
- SQLAlchemy : Un ORM (Object-Relational Mapping) pour interagir avec la base de données de manière plus intuitive.

Justification : SQLAlchemy permet de manipuler la base de données en utilisant des objets Python, ce qui simplifie le code et améliore la maintenabilité.

Hébergement

Heroku : Une plateforme cloud qui permet de déployer facilement des applications web.

Justification : via le GitHub pack student, Heroku offre un plan gratuit pour déployer des applications Flask, ce qui m'a permis d'héberger mon back-end avec une intégration facile de SQLite. Il est également facile à configurer via mon terminal, ce qui m'a permis de me concentrer sur le code plutôt que sur la gestion du serveur. Heroku est simple à utiliser et offre une intégration facile avec les dépôts GitHub, ce qui facilite le déploiement continu.

GitHub : Pour héberger des sites statiques

Justification : J'utilise toujours GitHub pour mes projets pour plus de sureté et être sure de pouvoir sauvegarder et documenter (à l'aide des commits) mon projet en continu.

(Voir la page « Stack technique » sur mon notion pour plus d'informations : https://www.notion.so/Stack-technique-43bd1dbe16474c65be036b4d29ecb49e)

2. Comment avez-vous mis en place votre environnement de travail ? Justifiez vos choix. (README.md)

Voici une copie de mon README.md :

```
### Prérequis

- **Python** : Assurez-vous d'avoir Python installé sur votre machine. Suivez les instructions ci-dessous pour installer Python si nécessaire.
- **Git** : Assurez-vous d'avoir Git installé pour cloner le repository. Vous pouvez télécharger Git depuis [git-scm.com](https://git-scm.com/downloads).

### Étape 1 - Cloner le repository GitHub

Ouvrez un terminal et exécutez la commande suivante pour cloner le repository :
>git clone git@github.com:buffsum/ECF.git
>cd votre-repository

### Étape 2 - Installer Python si ce n'est pas le cas
Ouvrez un terminal et exécutez la commande suivante pour cloner le repository :

## Debian/Ubuntu :
>sudo apt update
>sudo apt install python3 python3-venv python3-pip

## MacOs :
>/bin/bash -c "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/HEAD/install.sh)"
>brew install python

### Étape 3 - Créer et activer un environnement virtuel
Créez un environnement virtuel pour isoler les dépendances de votre projet :

## MacOs :
>python -m venv venv
>source venv/bin/activate
```

```
## Debian/Ubuntu :
>python3 -m venv .venv
>source .venv/bin/activate

### Étape 4 - Installer les dépendances
Installez les dépendances nécessaires à partir du fichier requirements.txt :

## MacOs :
>pip install -r requirements.txt

## Debian/Ubuntu :
>python3 -m pip install -r requirements.txt

### Étape 5 - Configurer la base de données
Initialisez la base de données en exécutant les commandes suivantes dans le terminal :

## MacOs :
>flask reset-db
>./start_mac.sh

## Debian/Ubuntu :
>flask reset-db
>./start_linux.sh

### Étape 6 - Lancer l'application (dans un autre terminal)
>flask run

L'application sera accessible à l'adresse suivante : http://127.0.0.1:5000
```

(Voir la page « Configuration de mon environnement de travail, déploiement, démarches et étapes » sur mon notion pour plus d'informations : https://www.notion.so/Configuration-de-mon-environnement-de-travail-d-ploiement-d-marches-et-tapes-7d615bac8c5c483eabfd552e54cc5db7)

3. Énumérez les mécanismes de sécurité que vous avez mis en place, aussi bien sur vos formulaires que sur les composants front-end ainsi que back-end.

Sécurité des Formulaires :

- **Validation des Données** : J'utilise Flask-WTF pour valider les entrées des utilisateurs, ce qui inclut la vérification de la longueur minimale des mots de passe et la validation des adresses email.
- **Protection CSRF** : J'emploie Flask-WTF pour générer et vérifier des tokens CSRF, empêchant ainsi les attaques de type Cross-Site Request Forgery.
- **Messages d'Erreur** : Des messages d'erreur spécifiques sont affichés pour guider les utilisateurs en cas de validation échouée.

Sécurité des Composants Front-End :

- **Utilisation de HTTPS** : J'ai configuré le serveur pour utiliser HTTPS afin de chiffrer les communications entre le client et le serveur.
- **Sanitisation des Entrées Utilisateur** : J'utilise des bibliothèques JavaScript pour échapper les entrées utilisateur, prévenant ainsi les attaques XSS (Cross-Site Scripting).
- **Content Security Policy (CSP)** : Une politique CSP est mise en place pour restreindre les sources de scripts et autres ressources, renforçant la sécurité contre les injections de contenu malveillant.

Sécurité des Composants Back-End :

- **Hashing des Mots de Passe** : Les mots de passe sont hachés avec des bibliothèques comme werkzeug.security avant d'être stockés dans la base de données.
- **Authentification et Autorisation** : Des mécanismes d'authentification vérifient l'identité des utilisateurs et des rôles sont utilisés pour contrôler l'accès aux différentes parties de l'application (admin, employee, user).

- **Protection contre les Injections SQL** : L'utilisation de SQLAlchemy comme ORM aide à prévenir les injections SQL en interagissant avec la base de données de manière sécurisée.
- **Gestion des Sessions** : Les sessions sont sécurisées, avec une configuration appropriée de la durée de vie des sessions et des cookies.
- **Journalisation et Surveillance** : Des mécanismes de journalisation permettent de suivre les activités suspectes et les tentatives d'intrusion.

4. Décrivez une veille technologique que vous avez effectuée, sur les vulnérabilités de sécurité.

Pour assurer la sécurité de mon application, j'ai réalisé une veille technologique ciblée sur les vulnérabilités de sécurité courantes. Cette veille a inclus plusieurs étapes clés :

- **Recherche et Lecture d'Articles** : J'ai consulté des articles de sécurité et des rapports de vulnérabilités sur des sites spécialisés

- **Suivi des Forums et Communautés** : J'ai suivi des forums et des communautés de développeurs, tels que Stack Overflow et Reddit, où des discussions sur les vulnérabilités et les solutions de sécurité sont fréquentes. Cela m'a permis de rester informé des nouvelles menaces et des réponses de la communauté

- **Analyse des Bulletins de Sécurité** : J'ai examiné les bulletins de sécurité publiés par les développeurs de frameworks et bibliothèques que j'utilise, comme Flask et SQLAlchemy. Les bulletins fournissent des informations sur les failles découvertes et les correctifs disponibles.

- **En parler autour de moi :** J'ai aussi essayé d'en discuter dans mon entourage avec des personnes ayant soit une appétence pour la cyber-sécurité, soit qui travaillent dans ce domaine

Cette veille technologique n'est pas parfaite car les vulnérabilités de sécurité n'est pas le sujet sur lequel j'ai pu passer le plus de temps, mais elle a eu un impact direct sur mes choix de sécurité pour le projet. Grâce à cette vigilance, j'ai pu développer une application plus sécurisée, minimisant les risques liés aux vulnérabilités courantes, même si je sais que je pourrai faire beaucoup mieux.

Partie 3 : Recherche

1. **Décrivez une situation de travail ayant nécessité une recherche durant le projet à partir de site anglophone. N'oubliez pas de citer la source**

La majorité des sites et forums que j'ai dû utiliser pour m'aider à résoudre mes problèmes étaient en anglais. Lors du développement du projet, j'ai rencontré des difficultés avec la fonction chargée de compter le nombre de consultations pour chaque animal « consultation_counts ». Cette fonction, censée incrémenter le compteur à chaque consultation (donc à chaque fois qu'un utilisateur cliquait sur « Avis du vétérinaire », présentait des problèmes tels que des incrémentations incorrectes ou des confusions entre les id des enregistrements vétérinaires (vet_records) et ceux des animaux (animals)

Pour résoudre ce problème, j'ai consulté plusieurs ressources en ligne :

- Stack Overflow : J'ai cherché des discussions et des solutions sur le comptage et la gestion des données en Python et Flask. J'ai trouvé des conseils sur la manière d'optimiser le traitement des données et d'éviter les erreurs courantes liées aux identifiants. Une discussion utile était celle-ci sur le comptage des occurrences dans des ensembles de données.
- Reddit :Le subreddit r/learnprogramming a été utile pour obtenir des conseils sur le débogage de fonctions complexes. J'ai partagé mon problème et reçu des recommandations sur l'utilisation appropriée des structures de données et des boucles en Python.
- Python Documentation : J'ai consulté la documentation officielle de Python pour mieux comprendre le comportement des dictionnaires et des boucles. Cette ressource m'a aidé à ajuster la logique de comptage pour s'assurer que les données étaient correctement accumulées.

2. **Mentionnez l'extrait du site anglophone qui vous a aidé dans la question précédente en effectuant une traduction en français.**

Partie 4 : Informations complémentaire

1. **Autres ressources**
2. **Informations complémentaires**