Dossier de définition des besoins pour l'hébergement de l'application Eclat D'Etoile

S2 B1 Groupe 9

TEAM CHARTREUSE

1 2 7 2 5 5 0

Cédric Colin
Marvyn Levin
Baptiste Dulieux
Timothée Meyer

02/04/2024

Sommaire

1 Description de l'application présentée

Eclat d'étoile s'est donné pour objectif de rendre accessible la peinture professionnelle au plus grand nombre. En collaboration avec l'innovante agence web Team Chartreuse, nous voulons offrir une plateforme digitale remplissant un double objectif : d'une part, la facilité et la fluidité de l'expérience utilisateur ; de l'autre l'extrême rigueur dans le choix et la conception de nos produits.

Dans le chapitre suivant, nous vous présenterons une vision détaillé de l'application web, en passant de nos offres à son fonctionnement technique.

1.1 Description non technique de l'application

L'application développé est un site Web de e-commerce pour l'entreprise Eclat D'Etoile, spécialisée dans la vente de peintures et de couleurs pour l'intérieurs, l'extérieur, etc...

Elle offre aux utilisateurs une expérience d'achat en ligne fluide et intuitive, mettant en avant une large gamme de produits de haute qualité, des descriptions complètes, des photos complémentaires et un service client réactif.

Les fonctionnalités demandées par le client sont les suivantes:

- **Catalogue de produits** varié avec des descriptions détaillées et des avis clients
- **Fonctionnalités de recherche** avancées avec des filtres par catégorie, prix, volume, mots, finition, etc
- **Tunnel de conversion** optimisé pour guider les utilisateurs tout au long du processus d'achat
- **Options de paiement** sécurisées incluant les cartes de crédit, les virements bancaires, etc
- **Gestion des stocks** en temps réel avec des alertes automatiques en cas de stock faible
- **Interface d'administration** backoffice

1.2 Description technique de l'application

L'architecture logicielle de l'application est basée sur une structure Web classique, utilisant les langages Python, SQL, HTML, CSS pour le développement. L'arborescence des fichiers est organisée de manière à séparer les composants front-end et back-end, avec une base de données relationnelle pour stocker les informations sur les produits, les commandes, les utilisateurs, les commentaires, les historiques et etc.

Langages, biliothèques, outils utilisés:

- **Python** pour le développement back-end
- **SQL** pour la gestion de la base de données
- **HTML**, **CSS**, **JavaScript** pour le développement front-end
- Framework **Flask** pour le développement Web
- Framework **Chart.js** pour la création de graphiques interactif pour l'administration
- **PythonAnyWhere** comme hébergeur pour les tests de l'application

Le déploiement de la nouvelle version de l'application en production se fera à l'aide d'un système de gestion de version tel que GitHub, permettant au client de transférer facilement les mises à jour sur le serveur d'hébergement PythonAnyWhere. Les dépendances de l'application seront gérées à l'aide d'un gestionnaire de paquets tel que pip pour Python.

2 Description des besoins

Dans cette section, nous décrirons les besoins nécessaires pour assurer le bon fonctionnement de l'application dans un environnement de production. Il est essentiel de prendre en compte les langages de programmation, les bibliothèques, les systèmes de gestion de base de données (SGBD), ainsi que les ressources matérielles et les comptes utilisateurs requis. De plus, nous aborderons la question de l'administration à distance de l'environnement de production et la facilité de déploiement des mises à jour de l'application. Enfin, nous discuterons des services optionnels tels que le choix du serveur web et du serveur de messagerie, en proposant des alternatives pertinentes pour répondre aux besoins spécifiques du client.

2.1 Besoins contraints

Pour que l'application fonctionne correctement, certains éléments sont indispensables et doivent être installés sur le serveur d'hébergement.

Voici les besoins contraints:

- **python 3.10** doit être installé avec la librairie **venv** et **pip** afin de gérer les dépendances de l'application.
- Une **application WSGI** afin de pouvoir mettre en place l'API flask.
- un **Système de Gestion de Bases de Données (SGBD)** compatible avec Flask est nécessaire pour stocker et gérer les données de l'application, par exemple MariaDB;
- un **espace disque** suffisant doit être disponible sur le serveur pour stocker toutes les données générées par l'application;
- des **comptes utilisateurs** avec les permissions appropriées doivent être configurés pour assurer une gestion adéquate du serveur.

2.2 Besoins ouverts

En plus des éléments indispensables, certains aspects de l'environnement de production peuvent être personnalisés en fonction des besoins et des préférences du client.

Voici les besoins ouverts:

- le **choix du serveur Web** pour l'hébergement de l'application, permi les options populaires telles que Apache ou Nginx;
- le **choix du nom de domaine** pour la mise en production, qui peut être crucial pour la visibilité et la reconnaissance de l'application en ligne;
- la **configuration de la sécurité** du serveur, comprenant notamment la mise en place d'un pare feux et la sécurisation des communications avec HTTPS pour protéger les données des utilisateurs.

2.3 Sécurisation - Utilisation de TLS

Sécuriser un site web, c'est avant tout protéger les données des utilisateurs et l'infrastructure employée. Les menaces sont nombreuses et ne dépendent pas toujours des sécurités misent en place face aux utilisateurs.

En effet, un site qui n'est pas au standard **HTTPS** est moins sécurisé et peut entrainer des vols de données, il est alors facile pour un individu malveillant d'intercepter les communications et en profiter pour voler les données des utilisateurs ou effectuer des actions non-autorisées.

Dans cette section, nous allons étudier les différentes menaces auxquels l'application « Eclat de Toile » fera face et nous proposerons des solutions adaptées.

2.3.1 Les attaques par déni de service

Une attaque par déni de service consiste à bloquer l'accès à un site aux yeux des utilisateurs.

2.3.1.1 Définition juridique

> Une attaque en déni de service ou en déni de service distribué (DDoS pour Distributed Denial of Service en anglais) vise à rendre inaccessible un serveur par l'envoi de multiples requêtes jusqu'à le saturer ou par l'exploitation d'une faille de sécurité afin de provoquer une panne ou un fonctionnement fortement dégradé du service. Ce type d'attaque peut être d'une grande gravité pour l'organisation qui en est victime. Durant l'attaque, le site ou service n'est plus utilisable, au moins temporairement, ou difficilement, ce qui peut entraîner des pertes directes de revenus pour les sites marchands et des pertes de productivité.
>
> L'attaque est souvent visible publiquement, voire médiatiquement, et laisse à penser que l'attaquant aurait pu prendre le contrôle du serveur, donc potentiellement accéder à toutes ses données, y compris les plus sensibles (données personnelles, bancaires, commerciales…) : ce qui porte directement atteinte à l'image et donc la crédibilité du propriétaire du site auprès de ses utilisateurs, clients, usagers, partenaires, actionnaire

source

2.3.1.2 Les types d'attaque par déni de service

Nom	Dangerosité	Fonctionnement
HTTP flood	**Sévère**	Cette attaque peut être comparé à une situation où l'on actualise sans-arrêt un navigateur web sur de nombreux ordinateurs différents à la fois. Cela provoque alors littéralement une vague soudaine de requêtes qui submergent le serveur, provoquant un déni de service. Cela peut être une attaque très simple avec une seule IP menaçante, mais peut aussi être très complexe et utiliser un véritable réseau d'appareils malveillants (souvent piraté).
Attaque protocolaire	**Modérée**	Ces attaques se caractérisent par une tentative d'épuiser toutes les ressources des serveurs ou des équipements réseau (notamment le pare-feu et les équilibres de charge)
SYN flood	**Sévère**	Cette attaque consiste à envoyer des requêtes mais à ne pas respecter la négociation en TCP, en ne répondant pas aux réponses du serveur attaqué.

Nom	Dangerosité	Fonctionnement
		La machine attaquée répond à chaque demande de connexion, puis attend la dernière étape de la négociation, qui n'a jamais lieu, ce qui épuise les ressources de la cible dans le processus.
Attaque volumétrique	**Modérée**	Cette catégorie d'attaques tente de créer une saturation en consommant toute la bande passante disponible entre la cible et Internet. De grandes quantités de données sont envoyées vers la cible en utilisant une forme d'amplification ou un autre moyen de créer un trafic massif, comme des requêtes provenant d'un botnet.

2.3.1.3 Se défendre contre ces attaques.

Dans le cas des attaques protocolaires et SYN flood, une configuration stricte au niveau du pare-feu permettra de s'en protéger.

Concernant les attaque volumétriques et HTTP flood, le serveur ne peut pas se protéger lui-même, il faut alors utiliser des services tel que cloudflare.com (Service en question) ou encore la suite AWS (Service en question) qui proposent des services permettant de se protéger de ces menaces.

2.3.2 Protection des données

La protection des données est essentielle pour garantir la confidentialité, l'intégralité et la disponibilité des informations stockées et traitées par l'application.

Voici quelques mesures de protection des données qui peuvent être mises en place:

- le **chiffrement des données** est important car les informations des utilisateurs, les transactions financières doivent être cryptées lorsqu'elles sont stockées dans la base de données et lorsqu'elles transitent sur le réseau. Le protocole TLS (Trasnport Layer Security) sera utilisé pour remedier au chiffrement des communications entre le serveur et les clients, assurant ainsi la confidentialité des données.

Le filtrage statique de paquets examine les en-têtes des paquets de données pour décider de leur sort en se basant sur l'adresse source et de destination, le protocole utilisé et les numéros de port. Le filtrage dynamique, quant à lui, prend des décisions basées sur l'état de la connexion et peut ajuster les règles de filtrage en temps réel.;

- une **politique de gestion des accès** doit être mise en place avec une politique stricte pour contrôler qui peut accéder aux données sensibles de l'application. Cela implique l'attribution de permissions spécifiques aux utilisateurs et la surveillance des activités d'accès pour détecter tout comportement suspect;

Il s'agit en effet de protéger la confidentialité et prévenir les accès non autorisés. Les réglementations DSP2, NIS2 ou la RGPD obligent les professionnels de ces secteurs à montrer patte blanche. Pour les entreprises opérant dans la finance, ou celles ayant un seul acteur à réaliser un ensemble

de tâches qui pourrait conduire à des fraudes, un système IAM permet d'inclure les règles de segmentation des droits ou SoD (Segregation of Duties) et de simplifier les contrôles.

- la **sauvegarde des données** récurente doit être mise en place pour prévenir la perte de données en cas d'incident ou de défaillance du système. Les sauvegardes doivent être stockées de manière sécurisée et périodiquement testées pour s'assurer de leur intégrité.

Ce processus se fait matériellement et il existe plusieurs solutions de stockage : les disques durs HDD et SSD, les NAS et le cloud avec un espace entre 20Go et 100Go pour stocker toutes les informations.

2.3.3 Chiffrement TLS

Le chiffrement TLS est généralement utilisé dans le standard **HTTPS** et permet d'apporter une couche de sécurité supplémentaire en applicant une méthode de chiffrement lors de l'échange de données entre le client et le serveur.

Afin de mettre en place ce chiffrement, il est nécessaire de configurer, au choix:

- **Apache**: En enregistrant les certificats TLS, apache s'occupera alors de cette sécurité;
- **Le DNS**: De nombreux fournisseurs de noms de domaines proposent un chiffrement des connections.

Il est recommandé d'utiliser la version la plus récente de TLS pour garantir la meilleure sécurité possible. À l'heure actuelle, **TLS 1.3** est la dernière version et est généralement préconisée pour assurer la sécurité optimale des connexions HTTPS (59% des sites l'utilise et le supporte).

2.3.4 Pare-feux

Les pare-feux sont des dispositifs de sécurité qui contrôlent et filtrent le trafic réseau entrant et sortant du serveur, en fonction de règles prédéfinies. Il existe différents pare-feux au niveau du protocole **TCP/IP**, **SSL/TLS** ou bien encore **ICMP**. Et l'utilisation de pare-feux tels que **Windows Firewall**, **ufw** ou **Cisco**.

Afin de configurer le pare-feux il faut procéder par étapes:

1. le **filtrage du trafic** est nécessaire afin de bloquer le trafic réseau malveillant ou non autorisé, notamment les attaques par déni de service distribué (DDoS), les scans de ports, les tentatives d'intrusion, etc;
2. la **protection contre les instrusions** doit être mise en place pour détecter et bloquer définitivement ces tentatives d'intrusion en analysant le comportement du trafic réseau et en identifiant les modèles suspects ou les signatures d'attaques connues;
3. la **surveillance continue** doit être mise en place afin de surveiller en temps réel le réseau afin de détecter les activités anormales ou les tentatives d'exploitation de vulnérabilités. Les alertes peuvent être générées en cas de détection d'une menace potentielle.

En mettant en place ces mesures de protection des données et des pare-feux efficaces, l'application obtiendra une sécurité renforcée et réduira considérablement les risques d'incidents liés à la sécurité des informations.