NOM	BERBIGIER
Prénom	Thomas
Date de naissance	01/11/1991

Copie à rendre

TP – Développeur Web et Web Mobile

Documents à compléter et à rendre

Lien du git : https://github.com/ThomasBerbigier/ECF-ZOO-ARCADIA
Lien de l'outil de gestion de projet : https://trello.com/b/X7Betjvq/ecf-zoo-arcadia
Lien du déploiement : https://salty-scrubland-07219-ce96da39ee13.herokuapp.com/index.php
Login et mot de passe administrateur : admin@arcadia.fr VK3dDet!jzKB
Login et mot de passe employé : employe@arcadia.fr OZww7y#49wZk
Login et mot de passe vétérinaire : veterinaire@arcadia.fr ecONaq!Dk9

SANS CES ELEMENTS, VOTRE COPIE SERA REJETEE

Partie 1 : Analyse des besoins

Le projet vise à développer une application web destinée aux visiteurs d'un zoo, permettant une visualisation détaillée des animaux, des services proposés, des horaires du zoo, tout en renforçant la notoriété, l'image de marque et les valeurs mises en avant par l'établissement.

Les technologies clés utilisées incluent HTML, CSS, JavaScript, Bootstrap pour le front-end, PHP et Composer pour le back-end, ainsi que MariaDB et MongoDB avec PDO pour la gestion des données. L'application sera hébergée sur la plateforme Heroku.

Les principales fonctionnalités incluent une page d'accueil attractive comprenant une section avis des visiteurs, un menu de navigation intuitif, une vue complète des services disponibles et des habitats des animaux, des espaces distincts pour l'administration, les employés et les vétérinaires, un système de connexion sécurisé, une section de contact pour les visiteurs, et des statistiques détaillées sur la popularité des animaux.

Le public cible est composé principalement des visiteurs potentiels du zoo, cherchant à obtenir des informations sur les animaux, sur les services offerts, et planifier leur visite de manière efficace.

L'approche de développement inclut une analyse approfondie des besoins des utilisateurs, une phase de maquettage pour la conception visuelle, une intégration soignée des fonctionnalités, la mise en place de règles de gestion robustes, et enfin le déploiement sur Heroku avec un accent particulier sur la sécurité. Chaque choix

technique sera justifié, mettant en avant les avantages spécifiques des technologies utilisées pour répondre aux exigences fonctionnelles et sécuritaires du projet.

Le projet Zoo Arcadia vise à assurer la pérennité de l'entreprise Arcadia en augmentant sa visibilité et en promouvant le zoo. Ce projet est crucial pour améliorer la notoriété et l'image de marque de l'établissement. L'objectif principal de cette application web est de permettre aux visiteurs de visualiser les animaux, leur état de santé, les services offerts et les horaires du zoo, tout en renforçant la notoriété et l'image de marque du zoo.

Zoo Arcadia est une application web destinée aux visiteurs du zoo et aux professionnels intervenant au zoo. Elle comprend plusieurs pages, dont une page d'accueil qui présente le zoo, les différents habitats, les services, les animaux, et les avis des visiteurs. La page habitats offre une vue globale de tous les habitats et animaux du zoo, tandis que la page services présente une vue d'ensemble de tous les services proposés par le zoo. La page contact permet aux visiteurs d'envoyer un mail sur la boite mail du zoo.

Le public cible de cette application comprend les visiteurs du zoo, les administrateurs, les vétérinaires et les employés. Le projet doit permettre aux visiteurs de visualiser les animaux, leur état, visualiser les services et les horaires du zoo. Les visiteurs peuvent laisser des avis sur leur expérience, mais ces avis sont soumis à validation par un employé du zoo. L'application comprend également un système de connexion sécurisé, accessible uniquement à l'administrateur, aux employés et aux vétérinaires. L'administrateur peut créer des comptes pour les employés et les vétérinaires, et gérer les services, les horaires, les habitats et les animaux du zoo. Les vétérinaires peuvent enregistrer des informations sur l'état de santé des animaux, leur alimentation, et laisser des commentaires sur les habitats. Les employés peuvent valider les avis des visiteurs, enregistrer les informations sur la nourriture donnée aux animaux, et modifier les services.

Les fonctionnalités principales et spécifiques de l'application comprennent une page d'accueil avec une présentation du zoo et des avis des visiteurs, un menu de navigation pour accéder à toutes les sections de l'application, une vue globale des services et des habitats, un espace administrateur, un espace employé, un espace vétérinaire, une page de connexion sécurisée, une page contact pour les visiteurs, et des statistiques sur la consultation des habitats. La sécurité sera assurée sur le front-end, le back-end, la base de données et les formulaires.

Le projet suivra les bonnes pratiques de développement, incluant l'utilisation de Git pour le contrôle de version. L'application sera déployée sur Heroku. Les choix techniques seront justifiés en fonction des avantages spécifiques des technologies utilisées. La documentation inclura un fichier README pour le déploiement en local, des fichiers de création et d'intégration de données, un manuel d'utilisation en format PDF, une charte graphique en format PDF, des maquettes (wireframes et mockups) pour les interfaces bureautiques et mobiles, et une documentation détaillant la gestion du projet. Le projet sera géré sous forme de kanban, avec un échéancier fixé au 23 juillet 2024.

Partie 2 : Spécifications technique

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

HTML (HyperText Markup Language) :
HTML est la norme de base pour créer des pages web. Il permet de structurer le contenu de l'application de manière sémantique, ce qui est essentiel pour l'accessibilité et le référencement. Utiliser HTML est une nécessité pour tout projet web car c'est la technologie fondamentale qui définit la structure du contenu sur le web.

CSS (Cascading Style Sheets) :
CSS est utilisé pour styliser les pages web et améliorer leur présentation visuelle. Il permet de séparer le contenu de la présentation, facilitant ainsi la maintenance et les mises à jour du design. Le choix de CSS est crucial pour créer une interface utilisateur attrayante et cohérente, tout en améliorant l'expérience utilisateur. CSS sera un complément à Bootstrap pour personnaliser davantage l'application.

JavaScript :
JavaScript est une technologie indispensable pour rendre les pages web interactives. Il permet d'ajouter des fonctionnalités dynamiques, comme les menus déroulants, les carrousels d'images, et les formulaires interactifs. JavaScript est choisi pour sa capacité à améliorer l'interactivité et l'engagement des utilisateurs sur le site web. JavaScript va permettre l'enregistrement des interactions spécifiques des utilisateurs.

Par exemple, l'implémentation d'un script JavaScript qui enregistre les clics sur les animaux.

Bootstrap :
Bootstrap est un framework CSS open-source qui facilite la création de sites web réactifs et mobiles. Le choix de Bootstrap est motivé par sa capacité à simplifier le développement front-end et à garantir une expérience utilisateur uniforme. Grâce à Bootstrap, l'implémentation des fonctionnalités telles que le carrousel, le menu de navigation, les offcanvas, les collapse, et les accordéons se feront facilement. Ces éléments sont essentiels pour offrir une interface utilisateur fluide et cohérente sur différents appareils, améliorant ainsi l'expérience des visiteurs.

MariaDB :
MariaDB est une base de données relationnelle qui offre des performances élevées, une sécurité robuste et une compatibilité avec MySQL. Elle est utilisée pour stocker des données structurées, telles que les informations sur les animaux, la nourriture, les habitats, les horaires, les comptes rendus, les utilisateurs, les rôles, les services, et les avis des visiteurs. Le choix de MariaDB est justifié par sa fiabilité, ses performances, et sa large adoption dans l'industrie.

MongoDB :
MongoDB est une base de données NoSQL qui permet de stocker des données non structurées ou semi-structurées. Elle est idéale pour gérer des données flexibles et évolutives, comme les statistiques de consultation des animaux. Le choix de MongoDB s'explique par sa capacité à gérer de grandes quantités de données variées et à s'adapter aux besoins changeants de l'application.

PHP :
PHP est un langage de programmation côté serveur largement utilisé pour le développement web. Il permet de créer des pages web dynamiques et interactives. PHP est choisi pour sa facilité d'intégration avec les bases de données, sa compatibilité avec la plupart des serveurs web, et sa large communauté de développeurs.

PDO (PHP Data Objects) :
PDO est une extension de PHP qui fournit une interface pour accéder aux bases de données. Elle permet de préparer et d'exécuter des requêtes SQL de manière sécurisée, en protégeant contre les injections SQL. Le choix de PDO est motivé par sa flexibilité et sa sécurité dans la gestion des interactions avec les bases de données.

Composer :
Composer est un gestionnaire de dépendances pour PHP. Il permet de gérer les bibliothèques et les packages nécessaires au projet, assurant ainsi que toutes les dépendances sont correctement installées et mises à jour. Le choix de Composer s'explique par sa capacité à simplifier la gestion des dépendances nécessaires pour l'envoi de mail lors de la création d'un utilisateur (PHPMailer), l'envoi de mail via la page contact, et pour intégrer MongoDB, notre base de données non relationnelle, efficacement dans notre application (MongoDB PHP Library).

Heroku :
Heroku est une plateforme cloud qui facilite le déploiement, la gestion et la scalabilité des applications. Elle permet de déployer rapidement l'application web et de s'assurer qu'elle est accessible en tout temps. Le choix de Heroku est justifié par sa simplicité d'utilisation, ses capacités de gestion automatique des ressources, et son support pour de nombreuses technologies, incluant PHP, MariaDB et MongoDB.

En résumé, chaque technologie est choisie en fonction de ses avantages spécifiques et de sa capacité à répondre aux besoins fonctionnels et techniques du projet Zoo Arcadia. L'utilisation combinée de ces technologies assure une application web robuste, performante et évolutive, tout en offrant une expérience utilisateur optimale.

Pour mettre en place notre environnement de travail pour le projet Zoo Arcadia, une série d'étapes ordonnées est à suivre:

Installation de Visual Studio Code (VSCode)

Il faut installer Visual Studio Code, un éditeur de code open-source très populaire parmi les développeurs web. Pour ce faire, VSCode se télécharge depuis le site officiel et s'installe facilement sur une machine de développement. VSCode offre des extensions puissantes pour les langages utilisés, tels que PHP, JavaScript, HTML et CSS, ce qui permet de configurer un environnement de développement riche en fonctionnalités. Les extensions comme PHP Intelephense pour l'intelligence de code PHP et Live Server pour le rechargement automatique des pages HTML s'avèrent particulièrement utiles.

Téléchargement et Installation de XAMPP

Pour créer un environnement de serveur local, XAMPP est nécessaire, il inclut Apache, MariaDB et PHP. XAMPP simplifie grandement le développement local en fournissant tous les outils nécessaires dans un seul package. Il se télécharge puis s'installe depuis le site officiel. Ensuite, via le panneau de contrôle , peuvent être démarrés les modules Apache et MariaDB, permettant ainsi d'exécuter l'application PHP et de gérer notre base de données relationnelle localement.

Configuration de la Base de Données

En ce qui concerne les données, MariaDB est utilisé pour les données relationnelles et MongoDB pour les données non relationnelles. Avec XAMPP, le panneau de contrôle donne accès à phpMyAdmin (interface web en PHP pour administrer à distance le SGBD MariaDB), il permet de créer la base de données avec MariaDB. Pour MongoDB, l'installation MongoDB Community Edition est requise pour développer en local, l'outil MongoDB Compass permet gérer et visualiser les données NoSQL.

Gestion des Dépendances avec Composer

Composer, le gestionnaire de dépendances pour PHP, est indispensable pour le projet. L'installation se fait en suivant les instructions sur le site officiel de Composer. Grâce à Composer, les bibliothèques essentielles s'installent facilement, PHPMailer pour l'envoi de mails lors de la création d'un utilisateur et via la page contact, ainsi que MongoDB PHP Library pour l'intégration avec MongoDB. Ces bibliothèques se trouvent facilement via le site packagist.org. Composer permet également de gérer automatiquement les mises à jour de ces dépendances, assurant ainsi que le projet reste à jour avec les dernières versions des bibliothèques utilisées.

Gestion du Code Source avec Git et GitHub

Pour le contrôle de version, utilisation de Git, et pour l'hébergement du code source, GitHub. Initialisation d'un dépôt Git local dans le répertoire de projet, puis création d'un dépôt distant sur GitHub. Cela permet de créer des branches pour développer des fonctionnalités individuelles, et de fusionner les modifications via des pull-requests. GitHub fournit un historique clair des modifications et facilite la gestion des versions du code.

Déploiement sur Heroku

Pour le déploiement de l'application, Heroku est une plateforme cloud qui simplifie le déploiement et la gestion des applications. Tout d'abord, installation de l'interface de ligne de commande Heroku CLI et création d'une application Heroku pour le projet. L'utilisation des add-ons JawsDB pour MariaDB et ObjectRocket pour MongoDB est nécessaire afin de gérer nos bases de données sur Heroku. Ces add-ons doivent être configurés, puis implémentés dans le code dans une condition afin que l'application puisse être utilisée en local ou via Heroku. Le dossier vendor et le fichier .env doivent être mis dans un fichier .gitignore afin de ne pas être commités.

Documentation et Gestion de Projet

Pour assurer une documentation claire et complète, un fichier README.md détaillant les étapes pour configurer l'environnement local est donné. Également un tableau Kanban sur Trello pour la gestion de projet, qui permet de suivre l'avancement des tâches et de prioriser les efforts de manière efficace.

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.

Les mécanismes de sécurité mis en place dans l'application :

header('X-Frame-Options: DENY'); : Empêche l'inclusion de l'application dans des cadres externes pour se protéger contre les attaques de clicjacking.

htmlspecialchars() : Utilisé pour convertir les caractères spéciaux en entités HTML, prévenant ainsi les attaques XSS en rendant inoffensives les balises potentiellement dangereuses.

CSRF Token : Généré et vérifié pour chaque formulaire afin de prévenir les attaques CSRF en assurant que les soumissions proviennent bien de l'application.

!empty() et isset() : Utilisés pour vérifier la présence de données avant de les utiliser.

Attribut required sur les formulaires : Garantit que les champs obligatoires sont remplis avant la soumission, améliorant la fiabilité des données reçues.

Gestion des erreurs : Messages d'erreur génériques pour les erreurs d'identification afin de ne pas divulguer d'informations sensibles aux utilisateurs non autorisés.

Méthode POST sur les formulaires : Utilisation exclusive de la méthode POST pour la soumission des données, réduisant les risques d'exposition des données sensibles dans l'URL.

filter_input() : Utilisé pour filtrer et valider les données d'entrée, assurant qu'elles correspondent au type attendu et prévenant les attaques XSS et d'injection de code.

$_SERVER['REQUEST_METHOD'] === 'POST' : Vérifie que la méthode de requête est POST, sécurisant les actions de modification de données.

session_set_cookie_params() : Configuré pour sécuriser les cookies de session en définissant des paramètres appropriés comme le chemin, la durée, utilisé seulement en connexion sécurisée, et la sécurité HTTPOnly.

Requêtes préparées (PDO) : Utilisées pour préparer et exécuter des requêtes SQL sécurisées, empêchant les injections SQL en séparant les données des instructions SQL.

Limitation accès BDD : Les utilisateurs de base de données ont des permissions minimales nécessaires pour leurs tâches spécifiques, réduisant ainsi les risques de compromission des données.

Hachage des mots de passe : Les mots de passe sont hachés avant d'être stockés en base de données, utilisant l'algorithme bcrypt pour une sécurité renforcée contre les attaques de récupération de mots de passe.

Utilisation de mot de passe fort : Politiques strictes pour les mots de passe, exigeant une longueur minimale et l'inclusion de caractères spéciaux (normes CNIL).

Fonction verifyUserLoginPassword : Méthode sécurisée pour vérifier la correspondance du nom d'utilisateur et du mot de passe lors de l'authentification, utilisant des techniques sécurisées de comparaison de hachage.

Vérification $_SESSION['role'] : Assure que seuls les utilisateurs autorisés peuvent accéder aux fonctionnalités spécifiques en vérifiant leur rôle dans la session, empêchant l'accès non autorisé.

Sécurité sur les fichiers uploadés : Nettoyage du nom de fichier, vérification de l'extension de fichier et limitation de taille, déplacement sécurisé du fichier.

Accès à la plateforme Heroku : Authentification à deux facteurs.

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

Le projet utilise plusieurs technologies pour assurer son bon fonctionnement. Assurer la sécurité de ces composants est crucial pour protéger les données des utilisateurs et maintenir la réputation de l'application. Le recours à un agrégateur de flux RSS s'est alors montré pertinent (feedly.com), afin d'avoir accès aux articles sur le thème de la sécurité informatique via plusieurs abonnements, notamment CERT-FR, RSS - Actualités CNIL, etc.

L'objectif de cette veille technologique est d'identifier et de corriger les vulnérabilités potentielles dans les composants technologiques utilisés par l'application. Cela inclut la mise à jour des bibliothèques, la configuration des serveurs et la vérification des pratiques de sécurité recommandées.

Pour Mongodb, consultation de base de données de vulnérabilités comme CVE Details pour les vulnérabilités spécifiques à MongoDB. Un article du CERT-FR est publié en date du 4 Juillet 2024 (flux RSS), émettant une vulnérabilité dans MongoDB avec un risque d'atteinte à l'intégrité des données.

L'article donne les systèmes affectés : libbson versions antérieures à 1.27.1 . J'effectue alors mes recherches pour savoir si mon système est concerné, via la commande dans le terminal : mongod --version , je ne vois alors pas de version libbson d'indiquée, une recherche sur google vient me confirmer que je ne suis pas concerné par cette vulnérabilité car libbson est une librairie pour C.

Pour PHP, je consulte des outils comme OWASP pour identifier les vulnérabilités communes dans les applications PHP. Les directives du site me disent de vérifier que Composer soit bien à jour, et que mes requêtes soient bien préparées et les variables filtrées.

En effectuant cette veille technologique, nous avons pu identifier les vulnérabilités potentielles et appliquer les correctifs nécessaires pour sécuriser notre application. Le suivi du flux RSS assure que l'application reste sécurisée.

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

Lors du développement de mon application, je devais connecter celle-ci à une base de données MongoDB pour stocker les clics sur les animaux afin d'avoir une vision de la popularité des animaux. N'ayant pas d'expérience préalable avec MongoDB, j'ai dû effectuer des recherches pour comprendre comment établir cette connexion en utilisant le pilote PHP. J'ai trouvé des ressources précieuses sur le site officiel de MongoDB, notamment la section dédiée aux pilotes PHP. Les instructions détaillées et les exemples de code fournis m'ont permis de configurer correctement la connexion à la base de données. Une bonne compréhension en anglais était nécessaire, mais j'ai pu intégrer MongoDB dans mon application sans difficultés majeures.
Source : https://www.mongodb.com/docs/drivers/php

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

Connection à un Serveur MongoDB sur votre machine locale

Si vous utilisez un serveur MongoDB sur votre machine en local à des fins de développement au lieu d'utiliser un cluster Atlas, vous devez compléter les étapes suivantes :

1. Téléchargez la version Communautée ou Entreprise de MongoDB Serveur .
2. Installez et configurez MongoDB Serveur.
3. Démarrez le serveur.

Après avoir démarré votre serveur MongoDB avec succès, spécifiez votre chaîne de caractères dans le code de connexion de votre driver.

Si votre Serveur MongoDB est utilisé localement, vous pouvez utiliser la chaîne de caractères de connexion `"mongodb://localhost:<port>"` où `<port>` est le numéro de port de votre serveur que vous avez configuré afin de recevoir les connexions futures.

1. **Autres ressources**

2. **Informations complémentaires**