NOM	FOLLACO
Prénom	Ludovic
Date de naissance	30/05/1972

Copie à rendre

TP Développeur Web et Web Mobile

Documents à compléter et à rendre

D'ordre général :

1) J'ai analysé le CdC et les données d'entrée,
2) J'ai rédigé un planning Kanban sur l'application web https://clickup.com. Voir planning Kanban du projet en cliquant sur le lien suivant : https://www.follaco.fr/index.php?page=kanban. Cliquez sur une tâche pour voir les détails.
3) J'ai analysé quelques sites spécialisés dans la vente de véhicule d'occasion,
4) J'ai réalisé quelques drafts sur papier pour me donner une idée de la proposition sur laquelle j'allais travailler :
    a. du nombre de page
    b. du découpage par fonction
    c. du design que j'allais proposer,
5) J'ai schématisé l'architecture des dossiers que j'allais mettre en œuvre,
6) J'ai réalisé les diagrammes de cas d'utilisation disponible en cliquant sur ce lien : https://www.follaco.fr/doc/Ludovic_FOLLACO_Garage_PARROT_Diagramme_Cas_Utilisation.pdf
7) J'ai réalisé le diagramme de séquence « visiteurs » disponible en cliquant sur ce lien : https://www.follaco.fr/doc/Ludovic_FOLLACO_Garage_PARROT_Diagramme_Sequence_Visiteurs.pdf
8) J'ai réalisé le diagramme de classe disponible en cliquant sur ce lien : https://www.follaco.fr/doc/Ludovic_FOLLACO_Garage_PARROT_Diagramme_Classe.pdf
9) Pour s'approprié et tester l'application, vous pouvez utiliser le manuel d'utilisation accéssible en cliquant sur le lien suivant : https://www.follaco.fr/doc/Ludovic_FOLLACO_Garage_PARROT_Manuel_Utilisateur.pdf

Les technologies choisies

Compte tenu de la complexité du sujet, j'ai décidé d'exploiter plusieurs technologies me permettant d'aboutir au résultat attendu avec un maximum de performance.

1) HTML, CSS et Bootstrap pour le contenu et le style
2) Javascript pour les opérations exécutables par le navigateur
3) PHP 8.2.13 pour répondre aux requêtes HTTP et PDO pour les requêtes sql
4) MariaDB 10.11.5 pour le schéma de données
5) ionos.fr pour héberger l'application web Accès au site en ligne en cliquant sur le lien suivant : https://www.follaco.fr

Moyens utilisés

1) Visual Studio Code comme éditeur de texte,
    a. Composer
    b. Mysql Autocomplete
    c. php debug
    d. php extension pack

e. php intelephense
f. symfony for VSCode

2) Bash et powershell comme terminaux,
3) Moteur de recherche, forums, blogs, … pour obtenir des informations sur les meilleures méthodes à utiliser.
4) Gimp pour créer, redimensionner, rogner et exporter les images,
5) Git comme gestionnaire de versions et comme outil de sauvegarde en local,
6) Github comme outil de sauvegarde distant, gestionnaire de tâches et de partage pour le développement collaboratif. Accéder au dépôt en cliquant sur le lien suivant : https://github.com/lululafrite/GarageParrot.git
7) getbootstrap.com pour les informations nécessaires à l'exploitation du Framework Bootstrap,
8) APACHE comme serveur HTTP local pour tester le site au fur et mesure du développement,
9) Xdebug pour le débogage pas à pas avec VS Code
10) Edge comme navigateur en mode client ou développeur,
11) Filezilla pour transférer le site dans le domaine follaco.fr,
12) gitignore.io pour générer un fichier .gitignore de démarrage,
13) PageSpeed Insights pour mesurer la performance du site et apporter les améliorations proposées,

Activité – Type 1 : Développer la partie front-end d'une application web ou web mobile en intégrant les recommandations de sécurité

1) Avant de commencer à travailler sur le front-end, j'ai mis en place une charte graphique accessible en cliquant sur le lien suivant :https://www.follaco.fr/doc/Ludovic_FOLLACO_Garage_PARROT_Charte_Graphique.pdf
2) A l'aide des draft, de la charte graphique et des diagrammes préalablement réalisés, j'ai commencé à produire les maquettes dans figma. Dans un premier temps, j'ai réalisé les composants et les variantes, ensuite, j'ai réalisé les maquettes de bureau en 992px, et, celles pour mobile en 375px. Une fois achevé, j'ai créé le lien de partage, accessible en cliquant sur le lien suivant : https://www.follaco.fr/index.php?page=mokup
3) J'ai récupéré sur internet, 5 images par véhicule pour une quinzaine de véhicules. Je les ai retravaillées pour qu'elles entrent dans mes restrictions techniques. Restrictions : La 1er image de chaque véhicule doit être dimensionnée en 350px x 180px. Pour les autres il faut rogner les parties inutiles. Pour finir, exporter les images en png ou jpg ou webp.
4) A partir de ce moment, je suis passé sur la partie back-end avant de poursuivre le front-end
5) Le back-end étant suffisamment avancé avec la BD créée et accessible et les modèle (Class) disponibles dans le projet VS CODE, j'ai pu poursuivre le front-end
6) J'ai créé toutes mes pages : index (page d'accueil), car, carEdit, user, userEdit, connexion …
7) Ensuite ou en même temps (selon les besoins), j'ai créé les élements (modules duplicables avec include ou require) : Head, Header, navBar, sélecteur de page, footer, table d'horaires, formulaire de contact et le module de commentaires
8) J'ai créé les fichiers JS, les feuilles de styles CSS

Activité – Type 2 : Développer la partie back-end d'une application web ou web mobile en intégrant les recommandations de sécurité

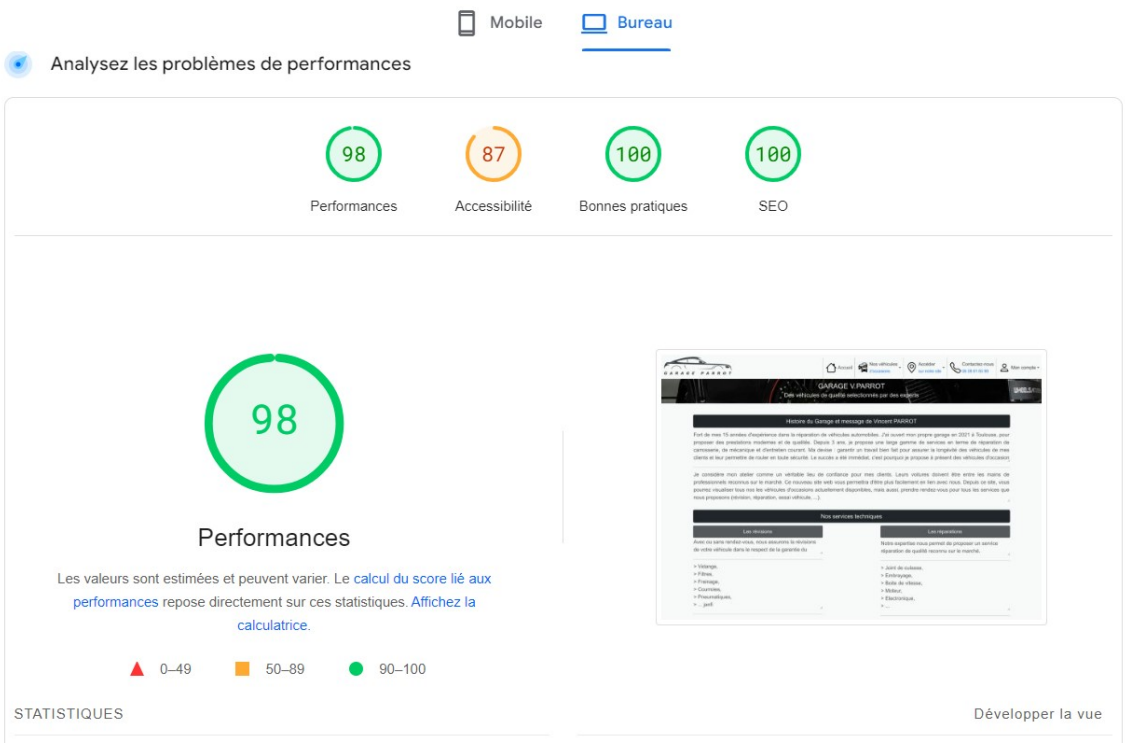
Compte tenu du besoin technique, j'ai décidé d'appliquer une architecture MVC (Model, View, Controller).

1) À l'aide des diagrammes et de la maquette figma, j'ai commencé par créer le schéma de données et les tables correspondantes
2) Ensuite, pour sécuriser l'application, j'ai créé un fichier index (qui n'est pas la page d'accueil) à la racine du dossier public pour en faire un « routeur », initialiser les variables superglobales, et traiter les titres en fonctions de la page active.
3) A l'issue, j'ai créé les dossiers que j'avais préalablement définis lors de l'étude préliminaire,
4) Ensuite, j'ai créé les modèles et les contrôleurs
5) J'ai réalisé les tests et les débogages,
6) Pour finir j'ai téléchargé le projet chez l'hébergeur ionos

Reste à faire

1) Amélioration des performances en tenant compte des recommandations de Pagespeed Insights, notamment en mode mobile.
2) Ajouter des articles avec des images sur la page d'accueil
3) Modération des commentaires avant publication (actuellement, publication immédiate)
4) Mettre les script JS dans des fichiers JS séparés

Analyse des performances avec Pagespeed insights



Mobile Bureau

Analysez les problèmes de performances

87	89	96	100
Performances	Accessibilité	Bonnes pratiques	SEO

87

Performances

Les valeurs sont estimées et peuvent varier. Le calcul du score lié aux performances repose directement sur ces statistiques. Affichez la calculatrice.

▲ 0–49 ■ 50–89 ● 90–100