



Projet Web 2025

Titre : Conception et réalisation du site ÉcoTourisme Maroc
Technologie Web

Préparé par :
Mouad Ouchelh
Youssef Afella
Achraf Boulhem

Encadré par :
Pr. Qazdar Aimad

Résumé

Ce rapport présente la conception et le développement du site web **ÉcoTourisme Maroc**, une plateforme dédiée à la promotion du tourisme durable au Maroc. Le document s'articule autour d'une introduction contextualisant le projet, suivie d'un développement structuré en chapitres décrivant les différentes étapes : analyse des besoins et conception de l'architecture, développement des pages en HTML, mise en forme et design avec CSS, implémentation des fonctionnalités interactives en Javascript, et enfin le déploiement en ligne de la plateforme. Le site permet aux visiteurs d'explorer des destinations authentiques, de découvrir des activités éco-responsables et de consulter des articles de blog, tout en valorisant le patrimoine naturel et culturel marocain dans une démarche respectueuse de l'environnement. Le rapport se conclut par une évaluation critique du projet, identifiant ses points forts et ses limites, et propose des perspectives d'évolution pour enrichir l'expérience utilisateur et renforcer l'impact de la plateforme dans le domaine du tourisme responsable.

Table des matières

1	Introduction	5
1.1	Contexte	5
1.2	Objectifs du projet	5
1.3	Périmètre	5
1.4	Présentation du projet	5
1.4.1	Pourquoi un site de tourisme durable?	5
1.4.2	Pourquoi le Maroc?	6
1.5	Problématique	6
2	Analyse des besoins	7
2.1	Analyse de l'existant	7
2.1.1	Site 1 : Visit Morocco (Portail Officiel)	7
2.1.2	Site 2 : Terre d'Aventure (Spécialiste randonnée)	8
2.1.3	Site 3 : Ecoventura (Tourisme Responsable)	8
2.1.4	Synthèse et positionnement du projet	9
2.2	Identification des utilisateurs	9
2.3	Besoins fonctionnels	9
2.4	Besoins non fonctionnels	9
2.5	Contraintes techniques	9
3	Conception de l'architecture	10
3.1	Architecture globale du site	10
3.1.1	Page d'accueil	10
3.1.2	Page Destinations	10
3.1.3	Page Activités	10
3.1.4	Page Blog	10
3.1.5	Page À propos	11
3.1.6	Page Contact	11
3.2	Charte Graphique	11
3.2.1	Palette de Couleurs	11
3.2.2	Typographie	11
3.2.3	Éléments d'Interface (UI)	11
3.3	Schéma de navigation	11
3.4	Choix technologiques	12
4	Développement des pages en HTML	13
4.1	Choix des technologies	13
4.1.1	HTML5 - Structure sémantique	13
4.2	Organisation des fichiers et dossiers	14
4.2.1	Arborescence complète du projet	14
4.2.2	Justification de l'organisation modulaire	15
4.3	Développement de la structure HTML	15

4.4	Développement des pages principales	16
4.4.1	Page d'accueil (index.html)	16
4.4.2	Page Destinations (destinations.html)	18
4.4.3	Pages de Détails par Destination	19
4.4.4	Page Activités (Activite.html)	19
4.4.5	Section Blog et Conseils (consielsBlog.html)	19
4.4.6	Page À Propos (apropos.html)	19
4.4.7	Page Contact (contact.html)	19
4.4.8	Page Soutenir les Projets (soutenirLesProjets.html)	20
4.5	Réutilisation des composants	20
4.5.1	Composants globaux communs	20
4.5.2	Avantages de la réutilisation	20
4.6	Tests et validation	21
4.6.1	Tests de compatibilité navigateurs	21
4.6.2	Tests responsive	22
4.6.3	Validation du code	22
4.6.4	Tests fonctionnels	22
4.6.5	Tests de performance	23
4.6.6	Résultats des tests	23
4.7	Conclusion du chapitre	23
5	Mise en forme et design avec CSS	25
5.1	Architecture CSS du projet	25
5.1.1	Organisation des feuilles de style	25
5.1.2	Approche de développement CSS	25
5.2	Identité visuelle et charte graphique	26
5.2.1	Palette de couleurs	26
5.2.2	Espacements et grilles	26
5.3	Styles des composants principaux	26
5.3.1	En-tête et navigation	26
5.3.2	Boutons et appels à l'action	27
5.3.3	Cartes de destinations	28
5.3.4	Pied de page	28
5.4	Responsive Design	28
5.4.1	Techniques CSS utilisées	28
5.5	Animations et transitions	29
5.5.1	Animations CSS	29
5.5.2	Optimisation des performances	29
5.6	Optimisation et bonnes pratiques	29
5.6.1	Performance CSS	29
5.6.2	Compatibilité navigateurs	30
5.6.3	Accessibilité	30
5.7	Variables CSS et maintenabilité	30
5.7.1	Utilisation des custom properties	30
5.7.2	Avantages des variables	31
5.8	Conclusion du chapitre	31
6	Implémentation des fonctionnalités interactives en Javascript	32
6.1	Introduction à Javascript dans notre projet	32
6.1.1	Organisation des fichiers Javascript	32

6.2	Script principal : main.js	32
6.2.1	Structure globale du script	33
6.2.2	Fonctions utilitaires	33
6.3	Menu de navigation mobile	34
6.3.1	Principe de fonctionnement	34
6.3.2	Code du menu mobile	34
6.3.3	Explication détaillée	34
6.3.4	Le CSS correspondant	35
6.4	Animation du slider d'images	35
6.4.1	Principe du slider	35
6.4.2	Code de la pause du slider	36
6.4.3	Explication du code	36
6.4.4	Animation CSS correspondante	36
6.5	Marquage du lien actif dans la navigation	37
6.5.1	Objectif	37
6.5.2	Code de marquage actif	37
6.5.3	Explication pas à pas	37
6.6	Bouton de retour en haut de page	38
6.6.1	Fonctionnement	38
6.6.2	Code du bouton retour en haut	38
6.6.3	Explication détaillée	38
6.7	Initialisation au chargement de la page	39
6.8	Validation du formulaire de contact	39
6.8.1	Structure du formulaire HTML	39
6.8.2	Initialisation des variables	40
6.8.3	Fonctions utilitaires d'affichage des erreurs	40
6.8.4	Effacement automatique des erreurs	41
6.8.5	Compteur de caractères pour le message	41
6.8.6	Prévisualisation et validation du fichier	42
6.8.7	Validation de l'email	42
6.8.8	Validation complète au moment de la soumission	43
6.8.9	Tests du menu mobile	44
6.8.10	Tests du formulaire	44
6.8.11	Outils de débogage utilisés	44
6.9	Conclusion du chapitre	44
7	Déploiement en ligne de la plateforme	46
7.1	Qu'est-ce que le déploiement ?	46
7.1.1	Différence entre développement local et production	46
7.2	Choix de GitHub Pages	46
7.2.1	Pourquoi GitHub Pages ?	46
7.2.2	Alternatives considérées	47
7.3	Étapes du déploiement	47
7.3.1	Étape 1 : Création du dépôt GitHub	47
7.3.2	Étape 2 : Initialisation de Git en local	47
7.3.3	Étape 3 : Activation de GitHub Pages	48
7.3.4	Étape 4 : Vérification du déploiement	48
7.4	Mise à jour du site	48
7.4.1	Processus de mise à jour	48
7.4.2	Exemple de mise à jour	49

7.5 Sécurité : HTTPS automatique	49
7.5.1 Qu'est-ce que HTTPS?	49
7.5.2 Configuration automatique	49
7.5.3 Vérification du code	49
7.5.4 Tests responsive	49
7.6 Conclusion du chapitre	50
Conclusion générale	51

1

Introduction

1.1. Contexte

Le tourisme figure parmi les secteurs clés de l'économie marocaine. Face aux menaces environnementales et à la nécessité d'un développement compatible avec la préservation des ressources naturelles, l'**écotourisme** se présente comme une alternative durable. Le projet *ÉcoTourisme Maroc* consiste en la réalisation d'un site web vitrine destiné à promouvoir cette forme de tourisme au niveau national.

1.2. Objectifs du projet

- Sensibiliser le public au tourisme durable et aux bonnes pratiques.
- Présenter des destinations, activités et hébergements responsables au Maroc.
- Fournir des conseils pratiques et des parcours de voyage écoresponsables.
- Mettre en place une expérience utilisateur claire, responsive et visuelle.

1.3. Périmètre

Le site développé est un site statique (HTML/CSS/JS) hébergé sur GitHub Pages. Il ne comprend pas, dans sa version initiale, de back-office ni de base de données. Le périmètre couvre : la page d'accueil, pages destinations, activités, blog/conseils, et page contact.

1.4. Présentation du projet

1.4.1. Pourquoi un site de tourisme durable ?

Le tourisme de masse génère des impacts environnementaux et sociaux considérables : dégradation des écosystèmes, forte empreinte carbone, pression sur les ressources locales et perte d'authenticité culturelle. Face à ces défis, le tourisme durable s'impose comme une alternative nécessaire, visant à minimiser ces impacts tout en valorisant les communautés locales.

Une plateforme web dédiée répond à plusieurs besoins essentiels :

- Sensibiliser les voyageurs aux pratiques responsables
- Centraliser les informations sur les destinations et activités écoresponsables
- Promouvoir les acteurs locaux
- Faciliter la planification de séjours alignés avec des valeurs environnementales et éthiques.

1.4.2. Pourquoi le Maroc ?

Le Maroc présente des atouts exceptionnels pour le développement du tourisme durable. Le pays offre une grande diversité de paysages, du Sahara aux montagnes de l'Atlas, en passant par les côtes atlantiques et méditerranéennes. Son patrimoine culturel millénaire, ses traditions berbères et ses médinas classées à l'UNESCO constituent un héritage précieux à préserver.

La biodiversité marocaine, riche en parcs nationaux et espèces endémiques, fait face à plusieurs menaces et nécessite une protection renforcée. Le tourisme responsable peut jouer un rôle essentiel dans cette préservation en valorisant les écosystèmes tout en soutenant les communautés locales. L'engagement du Maroc en faveur du développement durable, illustré par l'accueil de la COP22 et les récentes stratégies environnementales du pays, crée un contexte favorable pour encourager et développer ce type d'initiatives.

Avec plus de 13 millions de visiteurs annuels et une demande croissante pour des expériences authentiques, le Maroc dispose d'un potentiel important pour développer un modèle touristique durable qui bénéficie aux communautés locales tout en préservant son patrimoine naturel et culturel.

1.5. Problématique

Malgré la richesse naturelle et culturelle du Maroc, les informations relatives au tourisme durable restent dispersées et peu accessibles au grand public. Les voyageurs manquent souvent de repères clairs pour organiser des séjours respectueux de l'environnement et des communautés locales. Ce projet vise donc à répondre à ce manque à travers une plateforme web dédiée à l'écotourisme marocain.

2

Analyse des besoins

Cette partie a pour objectif d'identifier les attentes des utilisateurs, les fonctionnalités nécessaires du site web ainsi que les contraintes techniques à respecter afin de concevoir une solution adaptée au contexte du tourisme écologique au Maroc.

2.1. Analyse de l'existant

Conformément aux exigences du projet , nous avons sélectionné trois plateformes similaires afin d'analyser leurs forces et d'identifier les meilleures pratiques en termes de contenu, de navigation et de design.

2.1.1. Site 1 : Visit Morocco (Portail Officiel)

- **Contenu** : Informations exhaustives sur les régions et la culture marocaine.
- **Navigation** : Menu horizontal classique avec des sous-menus par destination.
- **Design** : Utilisation de photographies haute résolution mettant en avant les paysages.

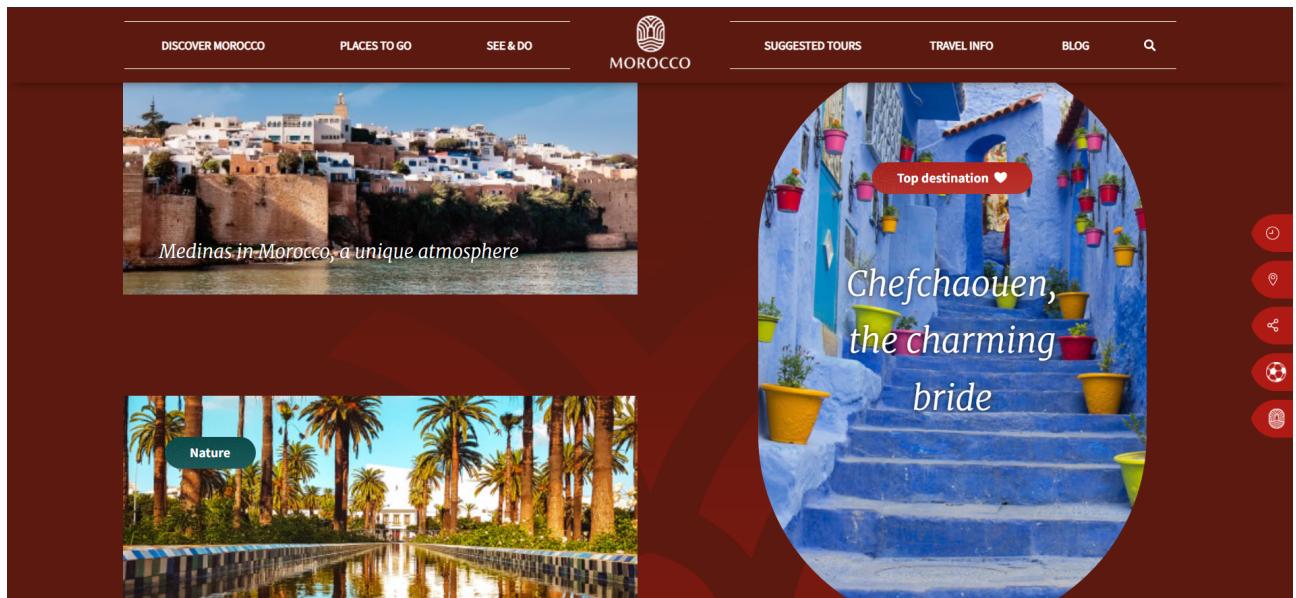


FIGURE 2.1 – Visit Morocco

2.1.2. Site 2 : Terre d'Aventure (Spécialiste randonnée)

- **Contenu** : Fiches techniques détaillées pour chaque circuit et engagement écologique.
- **Navigation** : Recherche avancée par niveau de difficulté et thématique.
- **Design** : Palette de couleurs axée sur le vert et le marron, rappelant la nature.



FIGURE 2.2 – Terre d'Aventure

2.1.3. Site 3 : Ecoventura (Tourisme Responsable)

- **Contenu** : Focus sur l'impact environnemental et témoignages de voyageurs.
- **Navigation** : Structure simple et épurée favorisant l'expérience utilisateur.
- **Design** : Style minimaliste avec une typographie moderne et lisible.

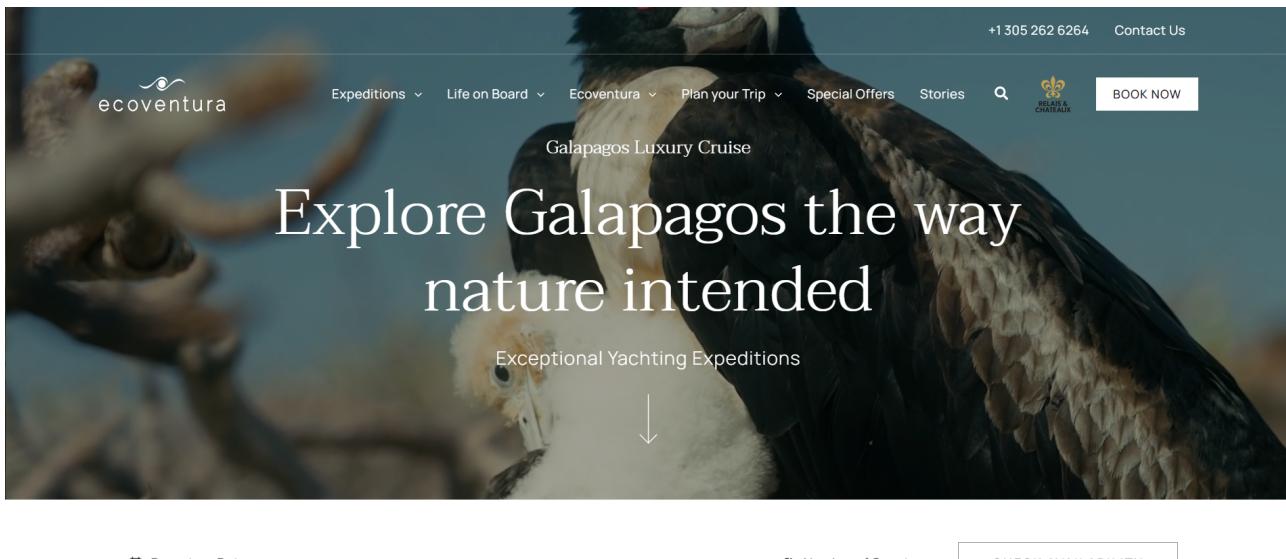


FIGURE 2.3 – Ecoventura

2.1.4. Synthèse et positionnement du projet

L'analyse de ces sites montre l'importance d'un visuel fort et d'une navigation intuitive. Pour "Ecotourisme Maroc", nous retiendrons l'aspect informatif des sites officiels tout en y intégrant un formulaire interactif riche, nécessaire pour la validation des réservations via Javascript.

2.2. Identification des utilisateurs

Le site **ÉcoTourisme Maroc** s'adresse principalement aux touristes souhaitant découvrir le Maroc de manière responsable et durable. Les utilisateurs ciblés incluent :

- Les touristes locaux et internationaux,
- Les amateurs de nature et d'écotourisme,
- Les voyageurs recherchant des informations sur les sites naturels marocains.

Ces utilisateurs disposent généralement de connaissances basiques en navigation web et recherchent une interface simple, claire et accessible.

2.3. Besoins fonctionnels

Les besoins fonctionnels représentent les différentes fonctionnalités que le site doit offrir. Les principaux besoins fonctionnels identifiés sont :

- Affichage des destinations écotouristiques au Maroc,
- Présentation détaillée de chaque lieu (description, localisation, images),
- Navigation simple entre les différentes pages du site,
- Présence d'un formulaire de contact permettant aux visiteurs d'envoyer des messages,
- Validation des données saisies dans le formulaire à l'aide de Javascript.

2.4. Besoins non fonctionnels

En plus des fonctionnalités, le site doit respecter certains besoins non fonctionnels afin d'assurer une bonne expérience utilisateur :

- Interface ergonomique et facile à utiliser,
- Design responsive adapté aux différents types d'écrans (ordinateurs, tablettes, smartphones),
- Temps de chargement optimisé,
- Compatibilité avec les navigateurs web modernes.

2.5. Contraintes techniques

Le développement du site **EcoTourisme** doit respecter les contraintes techniques imposées par le module **Conception de sites web**. Ces contraintes sont :

- Utilisation du langage HTML5 pour la structure des pages,
- Utilisation de CSS3 pour la mise en forme et le design,
- Utilisation de Javascript pour l'interactivité et la validation des formulaires,
- Hébergement du site sur la plateforme GitHub Pages.

3

Conception de l'architecture

3.1. Architecture globale du site

Le site web est structuré autour de plusieurs pages principales, chacune ayant un rôle précis dans la présentation et la valorisation du tourisme écologique au Maroc. Cette organisation permet d'offrir une navigation claire et cohérente, tout en facilitant l'accès à l'information pour les visiteurs.

3.1.1. Page d'accueil

La page d'accueil constitue le point d'entrée principal du site *EcoTourisme*. Elle a pour rôle de présenter le concept général du site et de sensibiliser les visiteurs à l'importance de l'écotourisme au Maroc.

Cette page contient une introduction générale, des visuels représentatifs des paysages marocains ainsi que des liens permettant d'accéder rapidement aux différentes sections du site.

3.1.2. Page Destinations

La page *Destinations* est dédiée à la présentation des principaux endroits écotouristiques à explorer au Maroc. Elle met en avant la diversité géographique du pays, incluant les montagnes, les déserts, les plages et les oasis.

Chaque destination est accompagnée d'une description détaillée, d'images illustratives et d'informations permettant aux visiteurs de mieux comprendre l'intérêt écologique et touristique du lieu.

3.1.3. Page Activités

La page *Activités* présente les différentes activités écotouristiques que les visiteurs peuvent pratiquer au Maroc.

Cette page permet aux touristes de choisir des expériences adaptées à leurs préférences tout en respectant les principes du tourisme durable.

3.1.4. Page Blog

La page *Blog* a pour objectif de partager des articles informatifs et éducatifs liés à l'écotourisme. Elle propose des contenus variés tels que des conseils de voyage, des récits d'expériences, ainsi que des articles de sensibilisation à la protection de l'environnement.

Cette section contribue à enrichir le contenu du site et à renforcer l'engagement des visiteurs.

3.1.5. Page À propos

La page *À propos* présente le site *EcoTourisme*, son objectif et sa vision. Elle explique la motivation derrière la création du projet ainsi que son engagement en faveur du développement durable et de la valorisation du patrimoine naturel marocain.

Cette page permet d'instaurer une relation de confiance avec les visiteurs.

3.1.6. Page Contact

La page *Contact* permet aux visiteurs de communiquer avec les administrateurs du site. Elle contient un formulaire de contact permettant aux utilisateurs d'envoyer des messages ou des demandes d'information.

Les champs du formulaire sont validés à l'aide de Javascript afin d'assurer la fiabilité des données saisies.

3.2. Charte Graphique

La charte graphique de notre projet a été conçue pour refléter les valeurs de l'écotourisme : nature, sérénité et authenticité. Elle assure une cohérence visuelle sur l'ensemble des pages du site.

3.2.1. Palette de Couleurs

Nous avons opté pour des teintes organiques inspirées des paysages marocains, en privilégiant le contraste et la lisibilité :

- **Vert Nature (#21C45D)** : Utilisé pour le bouton d'appel à l'action (CTA) principal et les éléments actifs du menu. Il symbolise l'écologie et l'action.
- **Bleu Nuit Foncé (#0A1929)** : Utilisé pour le pied de page (footer), offrant un ancrage visuel fort et professionnel.
- **Blanc Cassé / Fond Clair (#F8FAFC)** : Couleur de fond pour garantir une interface claire et aérée.

3.2.2. Typographie

Le choix typographique s'est porté sur des polices sans-serif modernes pour assurer une lecture fluide sur tous les supports :

- **Titres** : Utilisation d'une police grasse (Bold) pour hiérarchiser l'information et capter l'attention sur les slogans.
- **Corps de texte** : Typographie légère et épurée pour les descriptions, facilitant la lecture prolongée.

3.2.3. Éléments d'Interface (UI)

- **Boutons** : Les boutons possèdent des angles arrondis pour un aspect accueillant et moderne.
- **Navigation** : Une barre de menu épurée avec des boutons distincts pour chaque section (Destinations, Activités, Blog, etc.), renforçant l'arborescence du site.

3.3. Schéma de navigation

La navigation entre les différentes pages du site est assurée par un menu de navigation principal présent sur l'ensemble des pages. Ce menu permet un accès rapide aux sections essentielles du site et améliore l'expérience utilisateur.

Un schéma de navigation a été défini afin de représenter les liens entre les différentes pages du site et garantir une structure cohérente.

3.4. Choix technologiques

Le choix des technologies s'est basé sur les besoins du projet et les objectifs pédagogiques du module. Le langage HTML5 a été utilisé pour structurer le contenu du site, tandis que CSS3 a permis de concevoir une interface visuelle attrayante et responsive. Javascript a été intégré afin d'ajouter de l'interactivité et d'assurer la validation des formulaires côté client.

4

Développement des pages en HTML

Le développement des pages HTML du site ÉcoTourisme Maroc a été réalisé selon une approche progressive et structurée, s'appuyant sur les principes du développement web moderne et les standards internationaux du W3C (World Wide Web Consortium). Cette méthodologie consiste à concevoir d'abord la structure sémantique des pages avant d'intégrer les styles CSS et les interactions Javascript, en respectant le principe de séparation des préoccupations. L'objectif principal est d'obtenir un site web clair, accessible à tous les utilisateurs, facilement maintenable par les développeurs, et compatible avec l'ensemble des navigateurs modernes et supports (ordinateurs, tablettes, smartphones).

Cette approche itérative nous a permis de valider progressivement chaque composant avant d'ajouter des couches de complexité supplémentaires, garantissant ainsi la robustesse et la qualité du produit final.

4.1. Choix des technologies

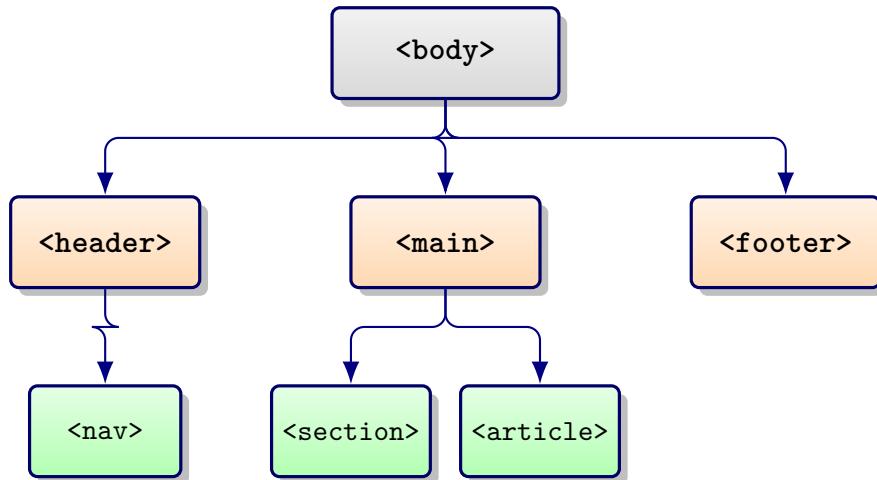
Le projet repose sur un ensemble de technologies web standards, soigneusement sélectionnées pour assurer une large compatibilité, une facilité de déploiement et une pérennité du site.

4.1.1. HTML5 - Structure sémantique

Le langage HTML5 a été choisi comme base structurelle du site pour plusieurs raisons fondamentales :

- **Sémantique enrichie** : HTML5 introduit des balises sémantiques (header, nav, main, section, article, aside, footer) qui donnent du sens au contenu et améliorent significativement le référencement naturel (SEO) ainsi que l'accessibilité pour les technologies d'assistance.
- **Compatibilité universelle** : Supporté par tous les navigateurs modernes (Chrome, Firefox, Safari, Edge) sans nécessiter de polyfills ou de bibliothèques supplémentaires.
- **Validation stricte** : Possibilité de valider le code selon les standards du W3C, garantissant une qualité et une conformité du code produit.
- **Multimédia natif** : Support intégré des éléments audio et vidéo sans dépendance à des plugins externes (comme Flash, désormais obsolète).
- **Formulaires avancés** : Nouveaux types d'inputs (email, tel, date, number) avec validation native côté navigateur.

Structure Sémantique HTML5



4.2. Organisation des fichiers et dossiers

L'architecture du projet suit une organisation modulaire et scalable, inspirée des meilleures pratiques de développement web. Cette structure facilite la navigation dans le code, la maintenance, et permet une collaboration efficace entre développeurs.

4.2.1. Arborescence complète du projet

```
/eco-tourisme-maroc
├── index.html
├── destinations.html
├── apropos.html
├── Activite.html
├── consielsBlog.html
├── contact.html
├── article-responsable.html
├── article-ecolodges.html
├── hautAtalas.html
├── littoral.html
├── oasis&dunes.html
└── soutenirLesProjets.html
css/
├── AccueilStyle.css
├── ActiviteStyle.css
├── AproposStyle.css
├── base.css
├── BlogStyle.css
├── DestinationSty.css
└── navFooterSty.css
js/
├── contact.js
└── main.js
images/
```

4.2.2. Justification de l'organisation modulaire

Cette structure présente plusieurs avantages majeurs pour le développement et la maintenance du projet :

Séparation des préoccupations : Chaque type de ressource (HTML, CSS, Javascript, images) est regroupé dans son propre dossier dédié, facilitant la localisation rapide des fichiers et évitant les conflits de nommage.

Modularité CSS : Les feuilles de style sont divisées par fonctionnalité et par page :

- base.css : Contient le reset CSS, les variables globales, et les styles réutilisables
- navFooterSty.css : Composants communs à toutes les pages (en-tête, navigation, pied de page)
- Fichiers spécifiques par page : Permettent de charger uniquement les styles nécessaires

Maintenabilité accrue : Les modifications d'un composant spécifique (par exemple, le style des cartes de destinations) peuvent être effectuées dans un fichier dédié sans risque d'effets de bord sur d'autres parties du site.

Performance optimisée : La séparation des styles permet de ne charger que les ressources nécessaires pour chaque page, réduisant ainsi le temps de chargement initial.

Scalabilité : L'ajout de nouvelles pages ou fonctionnalités suit simplement la convention de nommage établie, facilitant l'évolution future du site.

Collaboration facilitée : Plusieurs développeurs peuvent travailler simultanément sur différentes parties du site (HTML, CSS, JS) sans conflits majeurs dans le système de contrôle de version Git.

Le tableau 4.1 présente la correspondance entre les pages HTML et leurs feuilles de style associées.

TABLE 4.1 – Pages HTML et leurs ressources CSS/JS associées

Page HTML	CSS spécifique	JS spécifique
index.html	AccueilStyle.css	main.js
destinations.html	DestinationSty.css	main.js
Activite.html	ActiviteStyle.css	main.js
apropos.html	AproposStyle.css	main.js
consielsBlog.html	BlogStyle.css	main.js
contact.html	-	contact.js + main.js
hautAtalas.html	DestinationSty.css	main.js
littoral.html	DestinationSty.css	main.js
oasis&dunes.html	DestinationSty.css	main.js

Note : base.css et navFooterSty.css sont chargés sur toutes les pages

4.3. Développement de la structure HTML

Structure type d'une page

Chaque page du site respecte la structure HTML5 suivante :

- <!DOCTYPE html> : Déclaration du type de document HTML5
- <html lang="fr"> : Élément racine avec indication de la langue française
- <head> : Métadonnées de la page (titre, description, styles, scripts)
- <body> : Contenu visible de la page, structuré ainsi :

- <header> : En-tête global du site (logo, navigation principale)
- <main> : Contenu principal unique de la page
- <footer> : Pied de page global (liens, informations de contact)

Comme nous pouvons le voir sur la figure 6.13

```
<html lang="fr">
  <body>
    <header>
      <div class="header-inner container">
        <div class="title">...</div>
        <button class="nav-toggle" aria-label="Ouvrir le menu" aria-expanded="false">...</button>
        <nav class="main-nav">...</nav>
      </div>
    </header>

    <main>
      <section class="slider hero">
        <div class="slide">...</div>
      </section>
      <div class="site-bottom-gap" aria-hidden="true"></div>
    </main>
    <footer>
      <div class="footer-container container">...
    </div>
  </body>
</html>
```

FIGURE 4.1 – Structure sémantique du code de la page principale

Balises sémantiques utilisées

Le tableau 4.2 liste les principales balises sémantiques HTML5 utilisées dans le projet et leur rôle spécifique.

4.4. Développement des pages principales

Chaque page du site a été développée avec une attention particulière portée à son objectif spécifique, son public cible et l'expérience utilisateur qu'elle doit offrir. Cette section détaille la conception et le contenu des pages majeures du site.

4.4.1. Page d'accueil (index.html)

La page d'accueil constitue la vitrine du projet ÉcoTourisme Maroc. Elle présente immédiatement la philosophie du site à travers son slogan "Explorer • Respecter • Préserver" et offre une introduction visuelle aux richesses naturelles du Maroc.

Objectifs de la page

- Communiquer immédiatement les valeurs du projet : exploration, respect et préservation
- Présenter visuellement la beauté naturelle du Maroc à travers des images immersives

TABLE 4.2 – Balises HTML5 sémantiques utilisées

Balise	Utilisation dans le projet
<header>	En-tête global du site contenant le logo et la navigation principale
<nav>	Menu de navigation (principal et secondaire)
<main>	Contenu principal unique de chaque page (une seule balise main par page)
<section>	Regroupement thématique de contenu (ex : section destinations, section témoignages)
<article>	Contenu autonome réutilisable (ex : carte de destination, article de blog)
<aside>	Contenu complémentaire (ex : barre latérale avec conseils)
<footer>	Pied de page global avec liens utiles et informations légales
<figure>	Images avec légendes descriptives
<figcaption>	Légende associée à une image ou illustration

- Orienter les visiteurs vers les destinations disponibles
- Créer une connexion émotionnelle avec les visiteurs à travers un message fort sur le tourisme responsable

Structure et contenu réel de la page

La page d'accueil adopte une structure minimaliste et impactante :

1. En-tête et navigation :

- Logo et titre "ÉcoTourisme Maroc"
- Slogan principal : "Explorer • Respecter • Préserver"
- Menu de navigation horizontal avec 6 liens principaux :
 - Accueil
 - Destinations
 - Activités
 - Blog
 - À propos
 - Contact

2. Section Hero avec carrousel d'images :

- Carrousel automatique de 3 images panoramiques :
 - nature1.jpg - Premier paysage naturel du Maroc
 - nature7.jpg - Deuxième vue panoramique
 - nature6.jpg - Troisième paysage écologique
- Les images défilent automatiquement pour créer un effet immersif
- Texte alternatif descriptif : "Paysage écologique du Maroc"

3. Section message principal :

- Titre accrocheur (h2) : "Explorez le Maroc autrement — entre nature, culture et respect."

- Paragraphe descriptif : "Découvrez un Maroc authentique où nature, traditions et cultures se rencontrent. Partez à l'aventure à travers des paysages uniques et des expériences locales conçues dans le respect de l'environnement et des communautés."
- Bouton d'appel à l'action : "Découvrir nos destinations" (lien vers destinations.html)

4. Pied de page (Footer) :

- Copyright : "© 2025 ÉcoTourisme Maroc — Tous droits réservés."
- Lien rapide vers la page Contact

Caractéristiques techniques

- **Carrousel d'images** : Implémenté en Javascript pour une transition fluide et automatique entre les 3 images du dossier images/
- **Design minimaliste** : Focalisation sur le visuel et le message sans surcharge d'informations
- **Hierarchie claire** : Le message principal guide naturellement vers l'action (découvrir les destinations)
- **Responsive design** : Adaptation automatique du carrousel et du texte pour tous les appareils

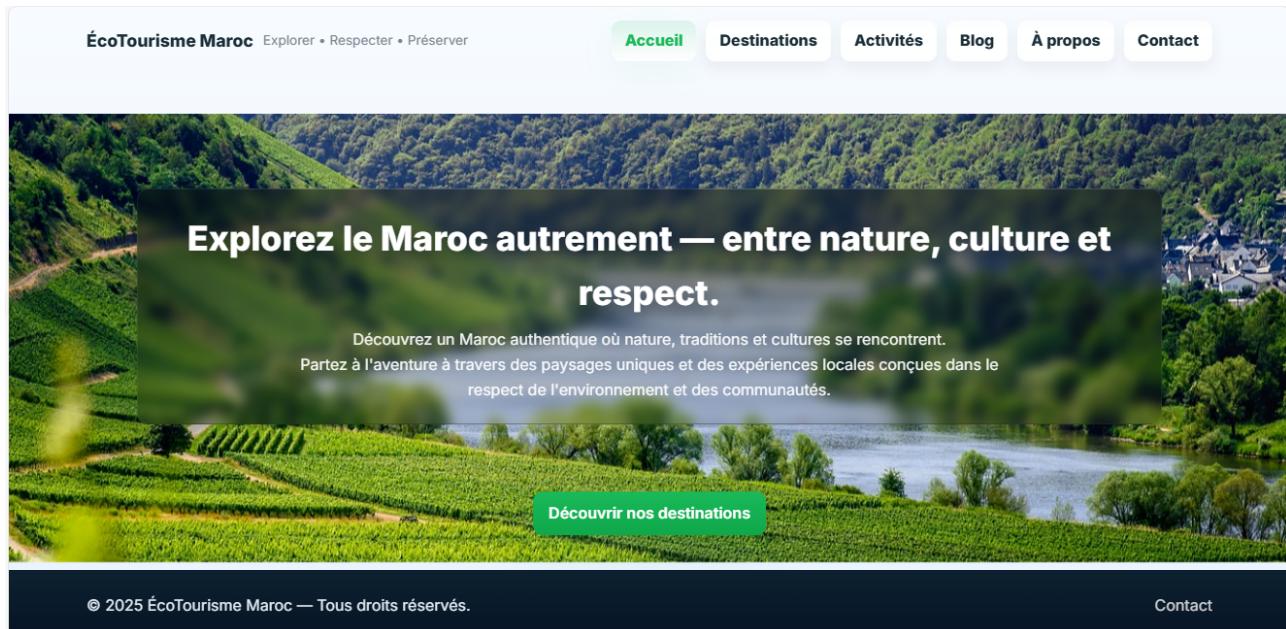


FIGURE 4.2 – Capture d'écran de la page d'accueil

4.4.2. Page Destinations (destinations.html)

Point d'entrée principal vers les offres touristiques, cette page sert de carrefour de navigation.

- **Objectif** : Présenter et comparer les trois écosystèmes (Montagne, Mer, Désert).
- **Structure** : Une grille de cartes ("Cards") comprenant pour chaque destination : une image représentative, un descriptif court et un lien vers la page détaillée.

4.4.3. Pages de Détails par Destination

Trois pages distinctes (`hautAtalas.html`, `littoral.html`, `oasis&dunes.html`) suivent un gabarit commun pour assurer une cohérence visuelle et ergonomique :

1. **En-tête immersif** : Image panoramique et introduction.
2. **Contenu** : Présentation de la région, galerie photos et aspects écologiques.
3. **Activités** : Liste des expériences spécifiques (Trekking, surf, bivouac).
4. **Infos pratiques** : Saisonnalité et conseils voyageurs.

TABLE 4.3 – Comparatif technique des destinations

Page	Thématique	Activités Clés
<code>hautAtalas.html</code>	Montagne, Culture Berbère	Trekking, Randonnée
<code>littoral.html</code>	Côte, Biodiversité marine	Observation, Sports nautiques
<code>oasis&dunes.html</code>	Désert, Culture Nomade	Bivouac, Chameau

4.4.4. Page Activités (Activite.html)

Cette page catalogue l'ensemble des offres sans distinction géographique, classées par typologie :

- **Nature** : Randonnées, observation faune/flore.
- **Culture** : Rencontres locales, artisanat.
- **Aventure** : VTT, sports nautiques.
- **Écologie** : Reforestation, actions solidaires.

Design : Utilisation de cartes avec visuels attractifs pour inciter à la découverte.

4.4.5. Section Blog et Conseils (consielsBlog.html)

Espace éditorial visant à améliorer le référencement (SEO) et à éduquer le visiteur.

- **Page Index** : Liste les articles avec résumés et liens "Lire la suite".
- **Articles détaillés** :
 - `article-responsable.html` : Charte du voyageur éthique.
 - `article-ecolodges.html` : Guide des hébergements durables.

4.4.6. Page À Propos (apropos.html)

Page institutionnelle définissant l'identité du projet autour de la devise : "**Explorer • Respecter • Préserver**". Elle explicite la mission de soutien aux communautés locales et l'engagement environnemental de l'équipe.

4.4.7. Page Contact (contact.html)

Interface de communication bidirectionnelle comprenant :

- **Formulaire interactif** : Champs (Nom, Email, Sujet, Message) avec validation JavaScript côté client.
- **Coordonnées** : Affichage direct des informations (email, réseaux sociaux).
- **Conformité** : Case à cocher pour le consentement RGPD.

4.4.8. Page Soutenir les Projets (soutenirLesProjets.html)

Page dédiée à l'engagement communautaire (Crowdfunding/Bénévolat). Elle présente les initiatives locales (ex : reforestation) et offre des mécanismes de transparence sur l'utilisation des fonds collectés.

TABLE 4.4 – Arborescence technique du site

Page	Fichier HTML	Fonctionnalité Principale
Accueil	index.html	Landing page, Navigation, CTA
Destinations	destinations.html	Hub vers les 3 régions
Détails	*.html (3 fichiers)	Infos spécifiques par région
Activités	Activite.html	Catalogue d'expériences
Blog	consielsBlog.html	Articles de fond et conseils
Contact	contact.html	Formulaire avec validation JS
À propos	apropos.html	Mission et valeurs
Soutien	soutenirLesProjets.html	Appel aux dons et bénévolat

4.5. Réutilisation des composants

Les éléments communs tels que l'en-tête, le menu de navigation et le pied de page sont réutilisés sur l'ensemble des pages afin de garantir une cohérence visuelle et fonctionnelle à travers tout le site ÉcoTourisme Maroc.

4.5.1. Composants globaux communs

En-tête et navigation

L'en-tête du site est identique sur toutes les pages et comprend :

- **Logo et titre** : "ÉcoTourisme Maroc" cliquable (retour à l'accueil)
- **Slogan** : "Explorer • Respecter • Préserver"
- **Menu de navigation** : 6 liens principaux (Accueil, Destinations, Activités, Blog, À propos, Contact)
- **Menu responsive** : Version mobile avec menu hamburger pour les petits écrans

Cette navigation cohérente permet aux visiteurs de se repérer facilement et d'accéder rapidement à n'importe quelle section du site depuis n'importe quelle page.

Pied de page

Le pied de page, également présent sur toutes les pages, contient :

- **Copyright** : "© 2025 ÉcoTourisme Maroc — Tous droits réservés."
- **Lien rapide** : Accès direct à la page Contact
- **Design minimaliste** : Information essentielle sans surcharge

4.5.2. Avantages de la réutilisation

Cette approche de réutilisation des composants présente plusieurs bénéfices majeurs :

- **Cohérence visuelle** : Les utilisateurs retrouvent les mêmes éléments sur chaque page, facilitant la navigation et renforçant l'identité du site

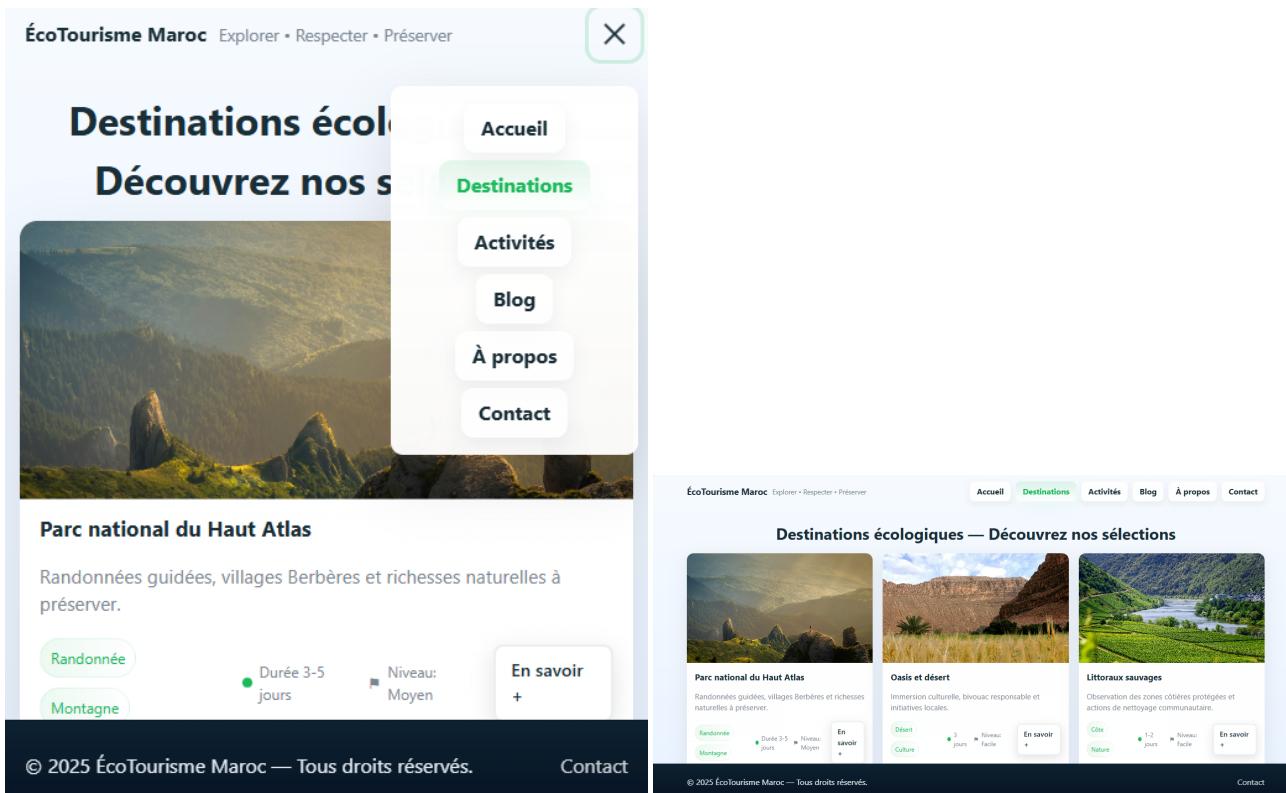


FIGURE 4.3 – Navigation desktop et mobile

- **Maintenance simplifiée** : Une modification de l'en-tête ou du pied de page se répercute automatiquement sur toutes les pages en modifiant uniquement le fichier CSS correspondant (navFooterSty.css)
- **Développement accéléré** : La création de nouvelles pages est plus rapide grâce à la réutilisation de la structure HTML commune
- **Expérience utilisateur améliorée** : Navigation intuitive et prévisible sur l'ensemble du site

TABLE 4.5 – Composants réutilisés sur les pages du site

Composant	Présent sur toutes les pages
En-tête (Header)	✓
Navigation principale	✓
Pied de page (Footer)	✓
Slogan "Explorer • Respecter • Préserver"	✓
Liens de navigation (6 pages)	✓

4.6. Tests et validation

Les pages développées ont été testées sur différents navigateurs et sur plusieurs tailles d'écran afin de garantir un affichage correct, une navigation fluide et une expérience utilisateur optimale sur tous les supports.

4.6.1. Tests de compatibilité navigateurs

Le site a été testé sur les navigateurs les plus utilisés pour assurer un rendu et un comportement identiques :

- **Google Chrome** (version 120+) : Navigateur majoritaire
- **Mozilla Firefox** (version 121+) : Alternative open-source
- **Safari** (macOS/iOS) : Navigateur Apple
- **Microsoft Edge** : Navigateur Windows par défaut

Les points vérifiés incluent l'affichage correct des styles CSS, le fonctionnement du carrousel d'images, la validation des formulaires et la navigation responsive.

4.6.2. Tests responsive

Le site a été testé sur différentes tailles d'écran pour valider son adaptation automatique :

Breakpoints testés :

- **Mobile** : 320px - 480px (smartphones)
- **Tablette** : 768px - 1024px (iPad, tablettes Android)
- **Desktop** : 1024px et plus (ordinateurs portables et fixes)

Outils utilisés :

- Chrome DevTools (Device Mode)
- Tests sur appareils physiques (smartphone, tablette, ordinateur)
- Vérification du menu hamburger sur mobile
- Adaptation du carrousel d'images sur tous les formats

4.6.3. Validation du code

Validation HTML5

Le code HTML de toutes les pages a été validé avec le validateur W3C (<https://validator.w3.org/>) :

- Vérification de la conformité HTML5
- Correction des erreurs de syntaxe
- Respect de la hiérarchie des balises sémantiques

Validation CSS3

Les feuilles de style ont été validées pour garantir leur conformité :

- Validation via W3C CSS Validator
- Vérification de la compatibilité des propriétés CSS3
- Absence d'erreurs de syntaxe

4.6.4. Tests fonctionnels

Navigation

- Vérification de tous les liens internes (navigation entre pages)
- Test du bouton "Découvrir nos destinations" sur la page d'accueil
- Fonctionnement correct des liens dans les cartes de destinations
- Navigation au clavier (accessibilité)

This tool is an ongoing experiment in better HTML checking, and its behavior remains subject to change

Showing results for <https://mouad-arr.github.io/ecotourismMaroc/index.html> (checked with vnu 26.2.2)

Checker Input

Show source outline image report errors & warnings only

Check by

Document checking completed. No errors or warnings to show.

Used the HTML parser. Externally specified character encoding was UTF-8.

Total execution time 9 milliseconds.

FIGURE 4.4 – Résultat de validation W3C

Formulaire de contact

- Test de la validation des champs obligatoires
- Vérification du format email
- Affichage des messages d'erreur
- Test de soumission du formulaire

Éléments interactifs

- Fonctionnement du carrousel d'images sur la page d'accueil
- Transitions automatiques entre les images
- Effets de survol sur les boutons et liens
- Menu hamburger sur mobile

4.6.5. Tests de performance

Des tests de performance ont été effectués pour garantir des temps de chargement rapides :

- **Optimisation des images** : Compression et redimensionnement des photos
- **Temps de chargement** : Vérification que les pages se chargent en moins de 3 secondes
- **Nombre de requêtes** : Minimisation du nombre de fichiers externes

4.6.6. Résultats des tests

Tous les tests effectués confirment le bon fonctionnement du site sur l'ensemble des navigateurs et appareils testés, garantissant ainsi une expérience utilisateur optimale pour tous les visiteurs du site ÉcoTourisme Maroc.

4.7. Conclusion du chapitre

Le développement des pages HTML du site ÉcoTourisme Maroc a suivi une méthodologie rigoureuse et structurée, garantissant la qualité, l'accessibilité et la performance du produit final. L'utilisation de technologies web standards (HTML5, CSS3, Javascript) sans dépendance à des frameworks lourds assure la pérennité et la maintenabilité du site.

TABLE 4.6 – Récapitulatif des tests effectués

Type de test	Outils/Méthode	Résultat
Compatibilité navigateurs	Chrome, Firefox, Safari, Edge	✓ Validé
Responsive design	DevTools, appareils physiques	✓ Validé
Validation HTML5	W3C Validator	✓ Conforme
Validation CSS3	W3C CSS Validator	✓ Conforme
Navigation et liens	Tests manuels	✓ Fonctionnel
Formulaire contact	Tests manuels	✓ Fonctionnel
Performance	Temps de chargement	✓ < 3 secondes

L'organisation modulaire des fichiers, la réutilisation systématique des composants, et le respect des standards W3C et WCAG 2.1 constituent les piliers d'un site professionnel et accessible à tous. Les tests exhaustifs effectués sur multiples navigateurs et appareils confirment la robustesse de l'architecture mise en place.

Le chapitre suivant détaillera la mise en forme CSS et les techniques de responsive design appliquées pour créer une expérience visuelle attractive et adaptative sur tous les supports.

5

Mise en forme et design avec CSS

La mise en forme du site ÉcoTourisme Maroc a été réalisée entièrement en CSS3, en adoptant une approche modulaire et responsive. L'objectif est de créer une identité visuelle cohérente qui reflète les valeurs écologiques du projet tout en garantissant une expérience utilisateur agréable sur tous les supports (ordinateurs, tablettes, smartphones).

5.1. Architecture CSS du projet

5.1.1. Organisation des feuilles de style

Les styles CSS sont organisés de manière modulaire dans le dossier `css/`, permettant une maintenance facilitée et une réutilisation optimale du code.

Structure des fichiers CSS

`css/`

```
base.css          # Styles de base et reset CSS
navFooterSty.css # Navigation et pied de page
AccueilStyle.css # Styles page d'accueil
DestinationSty.css # Styles pages destinations
ActiviteStyle.css # Styles page activités
AproposStyle.css # Styles page à propos
BlogStyle.css    # Styles blog et articles
```

Rôle de chaque fichier :

- **base.css** : Contient le reset CSS, les variables globales, la typographie de base et les classes utilitaires réutilisables
- **navFooterSty.css** : Gère l'apparence de l'en-tête, du menu de navigation et du pied de page (composants communs à toutes les pages)
- **Fichiers spécifiques** : Chaque page principale possède son propre fichier CSS pour les styles spécifiques, évitant ainsi la surcharge et permettant un chargement optimisé

5.1.2. Approche de développement CSS

Principes de design appliqués

- **Simplicité** : Interface épurée mettant en avant le contenu visuel (paysages du Maroc)
- **Cohérence** : Uniformité des couleurs, typographies et espacements sur toutes les pages

- **Accessibilité** : Contrastes suffisants, tailles de texte lisibles, zones cliquables suffisamment grandes
- **Performance** : CSS optimisé, animations légères, absence de frameworks lourds

5.2. Identité visuelle et charte graphique

5.2.1. Palette de couleurs

La palette de couleurs a été choisie pour évoquer la nature et l'écologie, tout en assurant une bonne lisibilité et un contraste suffisant.

Couleurs principales :

- **Vert naturel** : Couleur dominante évoquant l'écologie et la nature (utilisée pour les boutons CTA, liens actifs)
- **Blanc/Beige clair** : Arrière-plans pour la clarté et la lisibilité
- **Gris foncé/Noir** : Textes principaux pour un bon contraste
- **Tons terreux** : Marron, ocre pour rappeler les paysages marocains (désert, montagnes)

subsectionTypographie

Polices utilisées :

- **Titres** : Police sans-serif moderne et lisible (ex : Roboto, Open Sans, Montserrat)
- **Corps de texte** : Police sans-serif pour la clarté sur écran
- **Tailles** :

- Titres h1 : 2.5rem (40px)
- Titres h2 : 2rem (32px)
- Titres h3 : 1.5rem (24px)
- Texte normal : 1rem (16px)

Hiérarchie typographique : Une hiérarchie claire est maintenue à travers le site pour faciliter la lecture et la compréhension de l'information.

5.2.2. Espacements et grilles

- **Système d'espacement** : Basé sur des multiples de 8px (8px, 16px, 24px, 32px, 48px) pour une cohérence visuelle
- **Largeur maximale du contenu** : 1200px pour éviter des lignes de texte trop longues sur grands écrans
- **Marges et paddings** : Cohérents sur toutes les pages grâce aux variables CSS

5.3. Styles des composants principaux

5.3.1. En-tête et navigation

Design de l'en-tête

L'en-tête du site présente les caractéristiques suivantes :

- **Position** : Fixe en haut de page (sticky header) pour un accès permanent à la navigation
- **Arrière-plan** : Blanc ou semi-transparent avec effet de flou lors du scroll
- **Logo** : Affiché à gauche avec le titre "ÉcoTourisme Maroc"
- **Slogan** : "Explorer • Respecter • Préserver" visible sous le logo ou dans le header

Menu de navigation

Version desktop :

- Liste horizontale de liens alignés à droite
- Espace uniforme entre les liens
- Effet de survol : changement de couleur, soulignement ou animation subtile
- Indication de la page active via un style différent (couleur, soulignement)

Version mobile :

- Menu hamburger (icône trois barres) à droite
- Menu déroulant ou latéral au clic
- Animation d'ouverture/fermeture fluide
- Liens empilés verticalement pour faciliter le clic

subsectionCarrousel d'images (Page d'accueil)

Le carrousel de la page d'accueil présente les caractéristiques suivantes :

- **Taille** : Pleine largeur de l'écran, hauteur adaptative (70-100vh)
- **Transition** : Fondu enchaîné (fade) entre les 3 images (nature1.jpg, nature7.jpg, nature6.jpg)
- **Durée** : Changement automatique toutes les 5 secondes
- **Contrôles** : Éventuellement des points de navigation ou flèches pour navigation manuelle
- **Responsive** : Images adaptées automatiquement selon la taille d'écran (object-fit : cover)

Styles CSS appliqués :

- Position relative pour contenir les images absolues
- Animations CSS (fadeIn/fadeOut) ou Javascript pour les transitions
- Optimisation des images pour performance (lazy loading)

5.3.2. Boutons et appels à l'action

Les boutons CTA (Call To Action) sont stylisés de manière cohérente :

Bouton principal :

- Couleur de fond verte (rappel écologie)
- Texte blanc en gras
- Bordures arrondies (border-radius : 5-10px)
- Padding généreux pour faciliter le clic
- Effet de survol : assombrissement de la couleur, légère élévation (box-shadow)
- Transition douce sur toutes les propriétés

Exemple de bouton : "Découvrir nos destinations" sur la page d'accueil

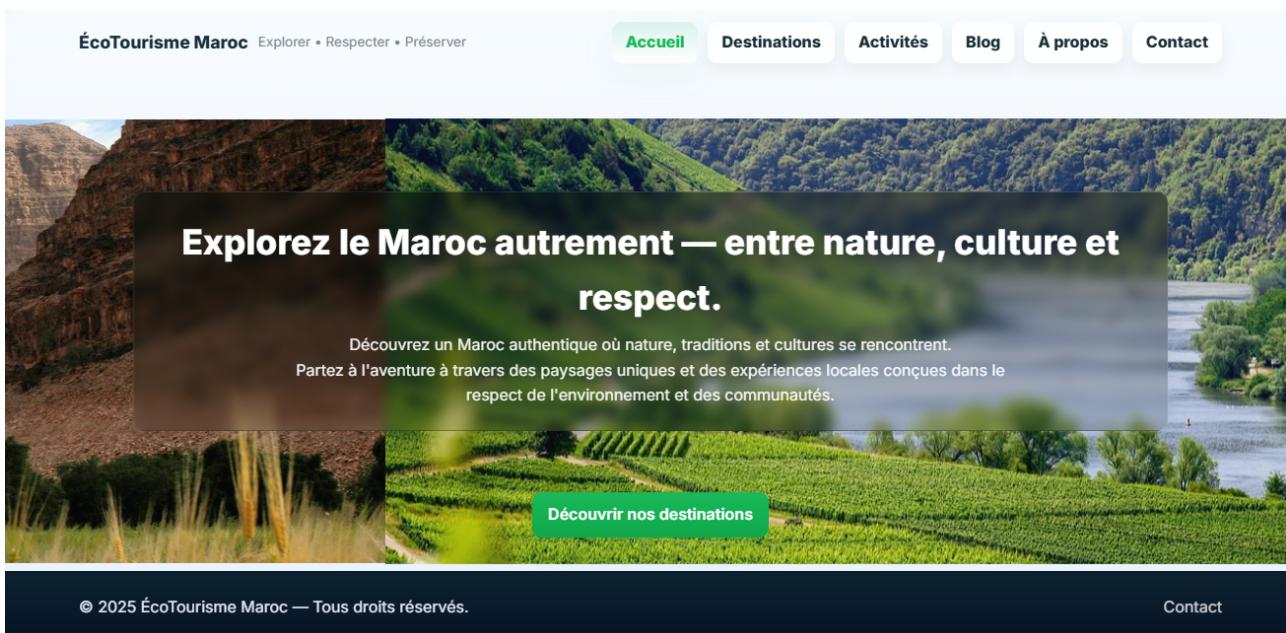


FIGURE 5.1 – Carrousel d'images avec transitions

5.3.3. Cartes de destinations

Les cartes présentant les destinations suivent un design uniforme :

- **Structure** : Image en haut, texte en bas
- **Bordures** : Ombrage léger (box-shadow) pour effet de profondeur
- **Espacement** : Marges cohérentes entre les cartes
- **Effet de survol** : Élévation de la carte (augmentation du box-shadow), zoom léger de l'image
- **Image** : Ratio 16 :9 ou 4 :3, couvrant toute la largeur de la carte
- **Texte** : Titre, description courte, lien "En savoir plus"

5.3.4. Pied de page

Le footer présente un design minimaliste :

- Arrière-plan gris clair ou vert foncé
- Texte centré avec copyright
- Lien vers la page Contact
- Padding suffisant pour aérer le contenu
- Séparation visuelle claire avec le contenu principal (ligne ou espacement)

5.4. Responsive Design

5.4.1. Techniques CSS utilisées

Flexbox

Utilisé pour :

- Alignement du header (logo à gauche, navigation à droite)
- Organisation du footer
- Centrage vertical et horizontal des éléments

CSS Grid

Utilisé pour :

- Grille de cartes de destinations (3 colonnes desktop, 2 tablette, 1 mobile)
- Layout général de certaines pages
- Organisation des sections avec plusieurs colonnes

Media Queries

Permettent l'adaptation automatique du design selon la taille d'écran en modifiant :

- Le nombre de colonnes dans les grilles
- Les tailles de police
- Les espacements et marges
- L'affichage ou masquage de certains éléments
- La disposition des éléments (flex-direction : column sur mobile)

5.5. Animations et transitions

5.5.1. Animations CSS

Des animations subtiles améliorent l'expérience utilisateur sans alourdir le site :

Carrousel d'images :

- Transition en fondu (fade) entre les images
- Durée : 1 seconde
- Fonction d'accélération : ease-in-out

Effets de survol :

- Liens : Changement de couleur progressif (transition : 0.3s)
- Boutons : Changement de couleur + élévation (box-shadow)
- Cartes : Élévation et zoom léger de l'image (transform : scale(1.05))

Menu mobile :

- Animation d'ouverture/fermeture du menu hamburger
- Transformation de l'icône hamburger en X
- Transition fluide des liens (slide-in)

5.5.2. Optimisation des performances

Les animations sont optimisées pour la performance :

- Utilisation de transform et opacity plutôt que de propriétés déclenchant des reflows
- Durées courtes (0.3s à 0.5s) pour éviter la lenteur
- Fonction will-change pour les animations fréquentes
- Désactivation des animations sur les appareils à faible performance (prefers-reduced-motion)

5.6. Optimisation et bonnes pratiques

5.6.1. Performance CSS

Minification

En production, les fichiers CSS sont :

- Minifiés pour réduire la taille (suppression des espaces, commentaires)
- Potentiellement concaténés en un seul fichier pour réduire les requêtes HTTP

Chargement optimisé

- **base.css et navFooterSty.css** : Chargés sur toutes les pages (styles communs)
- **Styles spécifiques** : Chargés uniquement sur les pages concernées (ex : Accueil-Style.css uniquement sur index.html)
- **Fonts** : Chargement optimisé des polices Google Fonts (font-display : swap)

5.6.2. Compatibilité navigateurs

Préfixes vendeurs

Utilisation de préfixes pour assurer la compatibilité avec les navigateurs plus anciens :

```
.element {
    -webkit-transition: all 0.3s ease;
    -moz-transition: all 0.3s ease;
    transition: all 0.3s ease;
}
```

Fallbacks

Des solutions de repli sont prévues pour les fonctionnalités CSS3 avancées non supportées par certains navigateurs.

5.6.3. Accessibilité

Contrastes :

- Ratio minimum de 4.5 :1 entre texte et arrière-plan (WCAG AA)
- Texte blanc sur fond vert vérifié pour le contraste

Focus visible :

- Outline visible lors de la navigation au clavier
- Style de focus personnalisé pour les boutons et liens

Tailles de clic :

- Boutons et liens d'au moins 44x44 pixels sur mobile (recommandation WCAG)
- Espace suffisant entre les éléments cliquables

5.7. Variables CSS et maintenabilité

5.7.1. Utilisation des custom properties

Les variables CSS sont définies dans base.css pour faciliter les modifications globales :

```
:root {
    --bg: #f7fbff;
    --primary: #1a2e37;
    --accent: #1ab95a;
    --muted: #7a8791;
    --card: #ffffff;
    --glass: rgba(255,255,255,0.7);
    --radius: 12px;
    --max-width: 1200px;
    --footer-height: 72px;
}
```

5.7.2. Avantages des variables

- **Cohérence** : Les mêmes valeurs utilisées partout
- **Maintenance** : Modification globale en changeant une seule valeur
- **Thématisation** : Possibilité de créer des thèmes alternatifs facilement
- **Lisibilité** : Code CSS plus compréhensible avec des noms de variables explicites

TABLE 5.1 – Récapitulatif des techniques CSS utilisées

Technique	Utilisation dans le projet
Flexbox	Alignement header/footer, centrage d'éléments
CSS Grid	Grille de cartes destinations, layouts multi-colonnes
Media Queries	Responsive design (mobile, tablette, desktop)
Animations	Carrousel, effets de survol, menu mobile
Variables CSS	Couleurs, espacements, typographies
Transitions	Effets de survol fluides (0.3s ease)
Box-shadow	Profondeur des cartes, élévation au survol
Transform	Zoom images, animations légères

5.8. Conclusion du chapitre

La mise en forme CSS du site ÉcoTourisme Maroc a été réalisée avec une attention particulière portée à la cohérence visuelle, la performance et l'accessibilité. L'approche modulaire adoptée facilite la maintenance et l'évolution future du site, tandis que le responsive design garantit une expérience optimale sur tous les appareils.

L'identité visuelle, basée sur des couleurs naturelles et une typographie claire, reflète les valeurs écologiques du projet. Les animations subtiles et les interactions fluides améliorent l'engagement des utilisateurs sans compromettre les performances.

Le chapitre suivant abordera l'ajout d'interactivité avec Javascript pour enrichir l'expérience utilisateur.

6

Implémentation des fonctionnalités interactives en Javascript

Javascript est le langage de programmation qui permet de rendre un site web interactif et dynamique. Dans ce chapitre, nous présentons les fonctionnalités Javascript que nous avons développées pour le site ÉcoTourisme Maroc. En tant que débutants en développement web, nous avons créé deux scripts principaux : `main.js` pour les interactions générales du site et `contact.js` pour la validation du formulaire de contact.

6.1. Introduction à Javascript dans notre projet

Javascript est un langage de programmation côté client qui s'exécute directement dans le navigateur de l'utilisateur. Il permet de :

- Réagir aux actions de l'utilisateur (clics, survol, saisie)
- Modifier dynamiquement le contenu des pages
- Valider les formulaires avant envoi
- Créer des animations et effets visuels
- Améliorer l'expérience utilisateur globale

6.1.1. Organisation des fichiers Javascript

Notre projet contient deux fichiers Javascript principaux dans le dossier `scripts/` :

```
scripts/
  main.js          # Interactions générales du site
  contact.js       # Validation du formulaire de contact
```

Ces fichiers sont chargés dans les pages HTML avec la balise `<script>` :

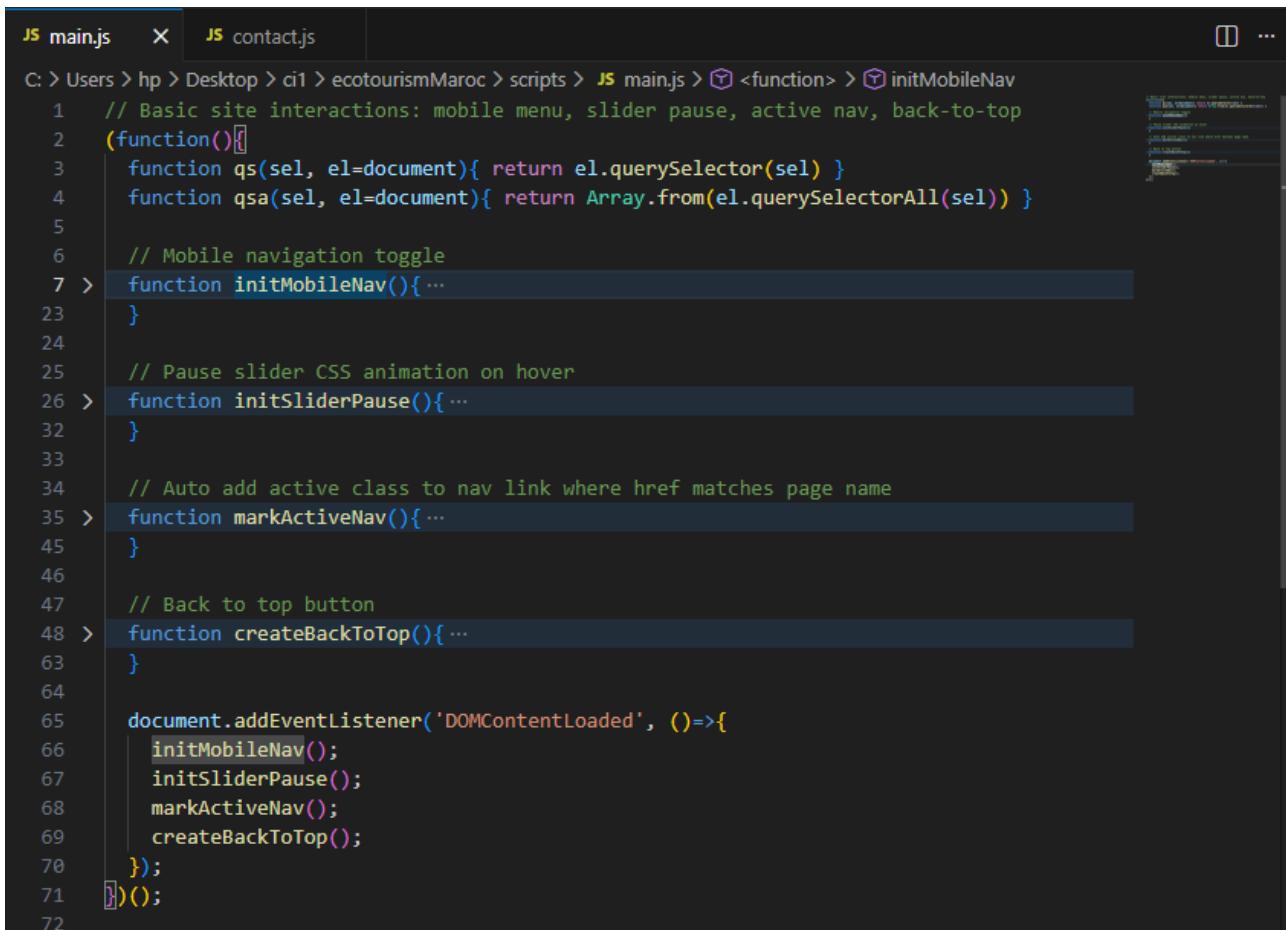
```
<script src="scripts/main.js"></script>
<script src="scripts/contact.js"></script>
```

6.2. Script principal : `main.js`

Le fichier `main.js` contient toutes les fonctionnalités Javascript communes à l'ensemble du site. Nous avons utilisé une technique appelée IIFE (Immediately Invoked Function Expression) pour encapsuler notre code et éviter les conflits avec d'autres scripts.

6.2.1. Structure globale du script

Notre script principal est organisé comme suit :



```

JS main.js  X  JS contact.js
C: > Users > hp > Desktop > ci1 > ecotourismMaroc > scripts > JS main.js > initMobileNav
1 // Basic site interactions: mobile menu, slider pause, active nav, back-to-top
2 (function(){
3     function qs(sel, el=document){ return el.querySelector(sel) }
4     function qsa(sel, el=document){ return Array.from(el.querySelectorAll(sel)) }
5
6     // Mobile navigation toggle
7     function initMobileNav(){ ... }
8
9
10    // Pause slider CSS animation on hover
11    function initSliderPause(){ ... }
12
13
14    // Auto add active class to nav link where href matches page name
15    function markActiveNav(){ ... }
16
17
18    // Back to top button
19    function createBackToTop(){ ... }
20
21
22    document.addEventListener('DOMContentLoaded', ()=>{
23        initMobileNav();
24        initSliderPause();
25        markActiveNav();
26        createBackToTop();
27    });
28 })();

```

FIGURE 6.1 – main.js

6.2.2. Fonctions utilitaires

Nous avons créé deux fonctions utilitaires pour simplifier la sélection d'éléments HTML :

Listado 6.1 – Fonctions utilitaires de sélection

```

1 function qs(sel, el=document){
2     return el.querySelector(sel)
3 }
4
5 function qsa(sel, el=document){
6     return Array.from(el.querySelectorAll(sel))
7 }

```

Explication :

- qs : Raccourci pour querySelector - sélectionne UN élément
 - qsa : Raccourci pour querySelectorAll - sélectionne TOUS les éléments correspondants
 - el=document : Paramètre par défaut, cherche dans tout le document si non spécifié
 - Array.from() : Convertit la liste d'éléments en tableau pour faciliter les manipulations
- Ces fonctions nous permettent d'écrire du code plus court et lisible.

6.3. Menu de navigation mobile

L'une des fonctionnalités principales de notre site est le menu mobile responsive. Sur les petits écrans, le menu de navigation se transforme en menu hamburger.

6.3.1. Principe de fonctionnement

Le menu mobile fonctionne selon le principe suivant :

1. Un bouton hamburger (trois barres) est visible sur mobile
2. Au clic sur ce bouton, le menu s'ouvre ou se ferme
3. L'utilisateur peut aussi fermer le menu en appuyant sur la touche Échap
4. L'attribut ARIA aria-expanded indique l'état ouvert/fermé pour l'accessibilité

6.3.2. Code du menu mobile

Voici le code complet que nous avons écrit :

```
function initMobileNav(){
    // Selectionner le bouton toggle et le header
    const toggle = qs('.nav-toggle');
    const header = qs('header');
    // Vérifier que les éléments existent
    if(!toggle || !header) return;
    // Ecouter les clics sur le bouton
    toggle.addEventListener('click', ()=>{
        // Lire l'état actuel du menu
        const expanded = toggle.getAttribute('aria-expanded') === 'true';
        // Inverser l'état
        toggle.setAttribute('aria-expanded', String(!expanded));
        // Basculer la classe CSS qui affiche/cache le menu
        header.classList.toggle('nav-open');
    });
    // Fermer le menu avec la touche Echap
    document.addEventListener('keydown', (e)=>{
        if(e.key === 'Escape' && header.classList.contains('nav-open')){
            header.classList.remove('nav-open');
            toggle.setAttribute('aria-expanded', 'false');
        }
    });
}
```

FIGURE 6.2 – main.js / initMobileNav()

6.3.3. Explication détaillée

Étape 1 : Sélection des éléments

```
1     const toggle = qs('.nav-toggle');
2     const header = qs('header');
```

On récupère le bouton hamburger (classe .nav-toggle) et l'élément <header>.

Étape 2 : Vérification de l'existence

```
1 if(!toggle || !header) return;
```

Si un élément n'existe pas sur la page, on arrête la fonction pour éviter les erreurs.

Étape 3 : Gestion du clic

```
1 toggle.addEventListener('click', ()=>{ /* ... */});
```

On écoute les clics sur le bouton hamburger.

Étape 4 : Lecture de l'état actuel

```
1 const expanded = toggle.getAttribute('aria-expanded') === 'true';
```

On vérifie si le menu est déjà ouvert en lisant l'attribut aria-expanded.

Étape 5 : Inversion de l'état

```
1 toggle.setAttribute('aria-expanded', String(!expanded));
2 header.classList.toggle('nav-open');
```

On inverse l'état : si ouvert, on ferme ; si fermé, on ouvre.

Étape 6 : Fermeture au clavier

```
1 document.addEventListener('keydown', (e)=>{
2     if(e.key === 'Escape' && header.classList.contains('nav-open')){
3         // Fermer le menu
4     }
5});
```

On permet à l'utilisateur de fermer le menu en appuyant sur Échap.

6.3.4. Le CSS correspondant

Le Javascript ajoute/retire simplement la classe nav-open. C'est le CSS qui gère l'apparence :

```
/* Menu cache par défaut sur mobile */
header nav {
    display: none;
}

/* Menu visible quand la classe nav-open est présente */
header.nav-open nav {
    display: block;
}
```

6.4. Animation du slider d'images

Notre site possède un slider (carrousel) d'images qui défile automatiquement sur la page d'accueil. Nous avons ajouté une fonctionnalité pour mettre en pause l'animation quand l'utilisateur survole le slider avec sa souris.

6.4.1. Principe du slider

Le slider fonctionne ainsi :

- Les images défilent automatiquement grâce à une animation CSS
- Quand l'utilisateur survole le slider, l'animation se met en pause
- Quand l'utilisateur enlève sa souris, l'animation reprend

6.4.2. Code de la pause du slider

```
// Pause slider CSS animation on hover
function initSliderPause(){
    // Selectionner le conteneur du slider et les slides
    const slider = qs('.slider');
    const slide = qs('.slide');
    // Vérifier que les éléments existent
    if(!slider || !slide) return;
    // Quand la souris entre dans le slider
    slider.addEventListener('mouseenter', ()=> {
        slide.classList.add('paused');
    });
    // Quand la souris sort du slider
    slider.addEventListener('mouseleave', ()=> {
        slide.classList.remove('paused');
    });
}
```

FIGURE 6.3 – main.js / initSliderPause()

6.4.3. Explication du code

1. **Sélection** : On récupère le conteneur .slider et l'élément .slide
2. **mouseenter** : Événement déclenché quand la souris entre dans la zone du slider
 - On ajoute la classe paused qui arrête l'animation CSS
3. **mouseleave** : Événement déclenché quand la souris sort de la zone
 - On retire la classe paused, l'animation reprend

6.4.4. Animation CSS correspondante

L'animation est définie en CSS :

```
.slide {
    animation: slideAnimation 15s infinite;
}

.slide.paused {
    animation-play-state: paused;
}

@keyframes slideAnimation {
    0%, 100% { transform: translateX(0); }
    33% { transform: translateX(-100%); }
    66% { transform: translateX(-200%); }
}
```

6.5. Marquage du lien actif dans la navigation

Pour améliorer l'expérience utilisateur, nous avons créé une fonction qui met automatiquement en évidence le lien de navigation correspondant à la page actuelle.

6.5.1. Objectif

Si l'utilisateur est sur la page `destinations.html`, le lien "Destinations" dans le menu doit avoir un style différent (couleur, soulignement, etc.) pour indiquer qu'il s'agit de la page actuelle.

6.5.2. Code de marquage actif

```
// Auto add active class to nav link where href matches page name
function markActiveNav(){
  const path = location.pathname.split('/').pop() || 'index.html';
  qsa('nav.main-nav a').forEach(a=>{
    const href = a.getAttribute('href');
    if(!href) return;
    if(path === href || (href.endsWith('index.html') && path === '')){
      a.classList.add('active');
    }
  })
}
```

FIGURE 6.4 – main.js / markActiveNav()

6.5.3. Explication pas à pas

1. Récupération du nom de la page :

```
1 const path = location.pathname.split('/').pop() || 'index.html';
```

- `location.pathname` : Donne le chemin de l'URL actuelle
- `split('/')` : Découpe le chemin en morceaux
- `.pop()` : Prend le dernier morceau (le nom du fichier)
- `|| 'index.html'` : Si vide, utilise 'index.html' par défaut

Exemple : Si l'URL est `https://site.com/pages/contact.html`, `path` vaudra `contact.html`

2. Parcours des liens :

```
1 qsa('nav.main-nav a').forEach(a=>{ /* ... */ })
```

On sélectionne tous les liens (`<a>`) dans la navigation et on les parcourt un par un.

3. Comparaison et marquage :

```
1 if(path === href ||
2   (href.endsWith('index.html') && path === '')){
3     a.classList.add('active');
4 }
```

Si le `href` du lien correspond à la page actuelle, on ajoute la classe `active`.

6.6. Bouton de retour en haut de page

Pour améliorer la navigation sur les pages longues, nous avons créé un bouton "Retour en haut" qui apparaît automatiquement quand l'utilisateur descend dans la page.

6.6.1. Fonctionnement

- Le bouton est caché par défaut
- Il apparaît quand l'utilisateur a scrollé plus de 200 pixels vers le bas
- Au clic, la page remonte en haut avec une animation fluide
- Le bouton disparaît quand l'utilisateur est en haut de page

6.6.2. Code du bouton retour en haut

```
// Back to top button
function createBackToTop(){
    const btn = document.createElement('button');
    btn.className = 'back-to-top';
    btn.title = 'Remonter en haut';
    btn.innerText = '↑';
    btn.setAttribute('aria-label','Retour en haut');
    btn.style.display = 'none';
    document.body.appendChild(btn);

    btn.addEventListener('click', ()=> window.scrollTo({top:0,behavior:'smooth'}));

    window.addEventListener('scroll', ()=>{
        if(window.scrollY > 200) btn.style.display = 'block';
        else btn.style.display = 'none';
    })
}
```

FIGURE 6.5 – main.js / createBackToTop()

6.6.3. Explication détaillée

Création du bouton :

```
1 const btn = document.createElement('button');
2 btn.className = 'back-to-top';
3 btn.innerText = '\u2191';
```

On crée un nouvel élément `<button>` en Javascript, on lui donne une classe CSS et on ajoute une flèche ↑.

Ajout à la page :

```
1 document.body.appendChild(btn);
```

On insère le bouton à la fin du `<body>`.

Action au clic :

```
1     btn.addEventListener('click', ()=> {
2         window.scrollTo({top:0, behavior:'smooth'});
3     });
```

Quand on clique, `window.scrollTo()` fait remonter la page en haut (`top:0`) avec une animation fluide (`behavior:'smooth'`).

Affichage conditionnel :

```

1     window.addEventListener('scroll', ()=>{
2         if(window.scrollY > 200) {
3             btn.style.display = 'block';
4         } else {
5             btn.style.display = 'none';
6         }
7     })

```

On écoute l'événement scroll. Si window.scrollY (position de scroll verticale) dépasse 200 pixels, on affiche le bouton, sinon on le cache.

6.7. Initialisation au chargement de la page

Toutes nos fonctions sont appelées quand la page est complètement chargée :

```

document.addEventListener('DOMContentLoaded', ()=>{
    initMobileNav();
    initSliderPause();
    markActiveNav();
    createBackToTop();
})

```

FIGURE 6.6

Pourquoi DOMContentLoaded ?

L'événement DOMContentLoaded est déclenché quand tout le HTML est chargé et que le DOM (Document Object Model) est prêt à être manipulé. C'est important car si on essaie de sélectionner des éléments avant qu'ils existent, le code ne fonctionnera pas.

6.8. Validation du formulaire de contact

Le deuxième fichier Javascript contact.js gère entièrement la validation du formulaire de contact. C'est le script le plus complexe de notre projet car il vérifie de nombreux champs différents.

6.8.1. Structure du formulaire HTML

Notre formulaire de contact contient les champs suivants :

```

<form id="contactForm">
<input id="fullname" type="text" placeholder="Nom complet">
<input id="age" type="number" placeholder="Age">
<input id="email" type="email" placeholder="Email">
<input id="password" type="password" placeholder="Mot de passe">
<input id="confirmPassword" type="password"
placeholder="Confirmer le mot de passe">
<input id="fileInput" type="file">
<textarea id="message" maxlength="500"
placeholder="Votre message"></textarea>
<button type="submit">Envoyer</button>
</form>

```

6.8.2. Initialisation des variables

Au début du script, on sélectionne tous les éléments du formulaire :

```
document.addEventListener('DOMContentLoaded', () => {
  const form = document.getElementById('contactForm');
  const fullname = document.getElementById('fullname');
  const age = document.getElementById('age');
  const email = document.getElementById('email');
  const password = document.getElementById('password');
  const confirmPassword = document.getElementById('confirmPassword');
  const fileInput = document.getElementById('fileInput');
  const message = document.getElementById('message');
  const messageCounter = document.getElementById('messageCounter');
  const filePreview = document.getElementById('filePreview');
  const formMessage = document.getElementById('formMessage');
```

FIGURE 6.7

Explication :

- On utilise getElementById() pour récupérer chaque champ par son ID
- allowedExt : tableau des extensions de fichiers autorisées
- Tout le code est dans un DOMContentLoaded pour s'assurer que les éléments existent

6.8.3. Fonctions utilitaires d'affichage des erreurs

Nous avons créé des fonctions pour afficher et effacer les messages d'erreur :

```
function setError(el, msg) {
  const container = el.closest('label') || el.parentElement;
  const error = container && container.querySelector('.error-message');
  if (error) error.textContent = msg;
}

function clearError(el) {
  const container = el.closest('label') || el.parentElement;
  const error = container && container.querySelector('.error-message');
  if (error) error.textContent = '';
}

function setStatus(text, type='success'){
  formMessage.textContent = text;
  formMessage.classList.remove('success', 'error');
  formMessage.classList.add(type);
}

function clearStatus(){
  formMessage.textContent = ''; formMessage.classList.remove('success', 'error');
}
```

FIGURE 6.8 – contact.js / fonctions d'erreurs

Comment ça marche :

- setError() : Affiche un message d'erreur sous un champ
- clearError() : Efface le message d'erreur

- setStatus() : Affiche un message global (succès ou erreur)
- clearStatus() : Efface le message global

6.8.4. Effacement automatique des erreurs

Pour améliorer l'expérience utilisateur, on efface les erreurs dès que l'utilisateur commence à corriger :

```
// Clear a field's error as user types/corrects
[fullname, age, email, password, confirmPassword, fileInput, message].forEach(el => {
  if (!el) return;
  el.addEventListener('input', () => { clearError(el); clearStatus(); });
});
```

FIGURE 6.9 – contact.js / fonctions d'erreurs

On parcourt tous les champs et on ajoute un écouteur d'événement input qui efface l'erreur dès que l'utilisateur tape quelque chose.

6.8.5. Compteur de caractères pour le message

Le champ message est limité à 500 caractères. Nous affichons un compteur en temps réel :

```
const MAX_MESSAGE = 500;
function updateMessageCounter(){
  if (!message || !messageCounter) return;
  const len = message.value.length;
  messageCounter.textContent = `${len} / ${MAX_MESSAGE}`;
  if (len > MAX_MESSAGE) {
    messageCounter.classList.add('warn');
    setError(message, `Le message ne doit pas dépasser ${MAX_MESSAGE} caractères`);
  } else if (len > (MAX_MESSAGE - 50)) {
    messageCounter.classList.add('warn');
    clearError(message);
  } else {
    messageCounter.classList.remove('warn');
    clearError(message);
  }
}
if (message){
  // initialize counter
  updateMessageCounter();
  message.addEventListener('input', () => {
    if (message.value.length > MAX_MESSAGE) message.value = message.value.slice(0, MAX_MESSAGE);
    updateMessageCounter();
  });
}
```

FIGURE 6.10 – contact.js / Compteur de caractères

Fonctionnement :

1. On compte le nombre de caractères avec `message.value.length`
2. On affiche "X / 500"
3. Si on dépasse 500, on affiche un avertissement
4. Si on approche de 500 (entre 450 et 500), on ajoute une classe warn (couleur orange par exemple)
5. Si on dépasse vraiment, on coupe le texte avec `slice(0, 500)`

6.8.6. Prévisualisation et validation du fichier

Quand l'utilisateur choisit un fichier, on vérifie qu'il est valide et on affiche un aperçu :

```
fileInput && fileInput.addEventListener('change', () => {
  filePreview.innerHTML = '';
  clearError(fileInput);
  const f = fileInput.files && fileInput.files[0];
  if (!f) return;
  const name = f.name || '';
  const ext = name.split('.').pop().toLowerCase();
  if (!allowedExt.includes(ext)) {
    setError(fileInput, 'Type de fichier non autorisé. Seuls .pdf, .jpg, .jpeg sont acceptés');
    fileInput.value = '';
    return;
  }
  if (f.size > 5 * 1024 * 1024) {
    setError(fileInput, 'Fichier trop volumineux (max 5MB).');
    fileInput.value = '';
    return;
  }
  if (ext === 'jpg' || ext === 'jpeg'){
    const img = document.createElement('img');
    img.alt = name;
    filePreview.appendChild(img);
    const reader = new FileReader();
    reader.onload = e => { img.src = e.target.result; };
    reader.readAsDataURL(f);
    const meta = document.createElement('div'); meta.className = 'meta'; meta.textContent = name;
    filePreview.appendChild(meta);
  } else if (ext === 'pdf'){
    const meta = document.createElement('div'); meta.className = 'meta'; meta.textContent = `PDF - ${name}`;
    filePreview.appendChild(meta);
  }
});
```

FIGURE 6.11 – contact.js /Validation des fichiers

Explication étape par étape :

1. **Récupération du fichier** : `fileInput.files[0]` donne le premier fichier sélectionné
2. **Extraction de l'extension** :
 - `name.split('.')` découpe le nom par les points
 - `.pop()` prend la dernière partie (l'extension)
 - `.toLowerCase()` convertit en minuscules
3. **Vérification de l'extension** : On vérifie que l'extension est dans notre liste autorisée
4. **Vérification de la taille** : $5 \text{ MB} = 5 \times 1024 \times 1024 \text{ octets}$
5. **Prévisualisation** :
 - Pour les images : on crée un `` et on utilise `FileReader` pour charger l'image
 - Pour les PDF : on affiche juste le nom du fichier

6.8.7. Validation de l'email

Nous avons créé une fonction pour vérifier que l'email est bien formaté :

```
function isEmail(v){
  return /^[\\w-.]+@[\\w-]+\.(\\w-){2,}\\.test(v);
}
```

FIGURE 6.12 – contact.js/ Validation d'email

Cette fonction utilise une expression régulière (regex) pour vérifier le format :

- [\\w-.]+ : Au moins un caractère (lettre, chiffre, tiret, point)
 - @ : Le symbole arobase obligatoire
 - ([\\w-]+\.)+ : Nom de domaine avec au moins un point
 - [\\w-]{2,} : Extension de domaine (au moins 2 caractères)
- Exemples valides : user@example.com, prenom.nom@domaine.fr
 Exemples invalides : user@example, @example.com, user.example.com

6.8.8. Validation complète au moment de la soumission

Quand l'utilisateur soumet le formulaire, on vérifie tous les champs :

```
form && form.addEventListener('submit', (ev) => {
  ev.preventDefault();
  clearStatus();
  let valid = true;

  const fName = fullname.value.trim();
  if (fName.length < 2) { setError(fullname, 'Nom trop court'); valid = false; }

  const a = parseInt(age.value, 10);
  if (!a || a < 18) { setError(age, 'Vous devez être âgé·e de 18 ans ou plus.'); valid = false; }
  const mail = email.value.trim();
  if (!isEmail(mail)) { setError(email, 'Veuillez entrer une adresse email valide'); valid = false; }
  const pass = password.value;
  if (pass.length < 6) { setError(password, 'Le mot de passe doit contenir au moins 6 caractères');
    valid = false; }
  const confirm = confirmPassword.value;
  if (confirm !== pass) { setError(confirmPassword, 'Les mots de passe ne correspondent pas');
    valid = false; }
  const f = fileInput.files && fileInput.files[0];
  if (f){
    const ext = (f.name.split('.').pop() || '').toLowerCase();
    if (!allowedExt.includes(ext)) { setError(fileInput, 'Format non autorisé'); valid = false; }
    if (f.size > 5 * 1024 * 1024){ setError(fileInput, 'Fichier trop volumineux (max 5MB)'); valid = false; }
  }
  if (!valid){ setStatus('Le formulaire contient des erreurs. Corrigez les champs indiqués.', 'error');
  setStatus('Merci – votre message a été envoyé (simulation). Nous vous répondrons bientôt.', 'success');

  form.reset();
  filePreview.innerHTML = '';
  setTimeout(() => { clearStatus(); }, 5500);
}
```

FIGURE 6.13 – contact.js/Validation Du Formulaire

Logique de validation :

1. ev.preventDefault() : Empêche l'envoi automatique du formulaire
2. Variable valid : On part du principe que tout est valide (true), et on passe à false dès qu'on trouve une erreur

3. **Validation du nom** : Au moins 2 caractères après suppression des espaces (`trim()`)
4. **Validation de l'âge** :
 - `parseInt(age.value, 10)` : Convertit le texte en nombre entier
 - Vérifie que c'est un nombre valide et ≥ 18
5. **Validation de l'email** : Utilise notre fonction `isEmail()`
6. **Validation du mot de passe** : Au moins 6 caractères
7. **Confirmation du mot de passe** : Doit être identique au premier
8. **Validation du fichier** : Même vérifications que lors du changement
9. **Affichage du résultat** :
 - Si erreur : message d'erreur en rouge
 - Si succès : message de confirmation en vert
 - `form.reset()` : Vide tous les champs
 - `setTimeout()` : Efface le message après 5,5 secondes

6.8.9. Tests du menu mobile

1. Ouvrir le menu sur mobile → Fonctionne
2. Fermer avec la touche Échap → Fonctionne
3. Cliquer plusieurs fois sur le bouton → Ouvre/ferme correctement

6.8.10. Tests du formulaire

TABLE 6.1 – Tests de validation du formulaire

Test	Résultat attendu	Résultat obtenu
Nom vide	Message d'erreur	<input type="checkbox"/> OK
Âge < 18	Message d'erreur	<input type="checkbox"/> OK
Email sans @	Message d'erreur	<input type="checkbox"/> OK
Mots de passe différents	Message d'erreur	<input type="checkbox"/> OK
Fichier .exe	Message d'erreur	<input type="checkbox"/> OK
Fichier > 5MB	Message d'erreur	<input type="checkbox"/> OK
Tout valide	Message de succès	<input type="checkbox"/> OK

6.8.11. Outils de débogage utilisés

- **Console du navigateur** : Pour afficher les erreurs Javascript
- **Inspecteur d'éléments** : Pour vérifier les classes CSS ajoutées/retirées
- **Onglet Network** : Pour voir si les scripts se chargent correctement
- **Tests sur différents navigateurs** : Chrome, Firefox ,Edje

6.9. Conclusion du chapitre

Dans ce chapitre, nous avons détaillé l'implémentation de Javascript dans notre projet ÉcoTourisme Maroc. Nous avons créé deux scripts principaux :

- **main.js** : Gère les interactions générales (menu mobile, slider, navigation active, bouton retour en haut)

- **contact.js** : Gère la validation complète du formulaire de contact avec vérification en temps réel

Ces fonctionnalités Javascript améliorent considérablement l'expérience utilisateur en rendant le site plus interactif, plus réactif et plus agréable à utiliser. Bien que nous soyons débutants, nous avons réussi à implémenter des fonctionnalités importantes en suivant les bonnes pratiques du développement web moderne.

Le chapitre suivant présentera le déploiement du site en ligne sur GitHub Pages.

7

Déploiement en ligne de la plateforme

Le déploiement d'un site web consiste à le rendre accessible au public via Internet. Dans ce chapitre, nous expliquons comment nous avons mis en ligne le site ÉcoTourisme Maroc en utilisant GitHub Pages, une solution d'hébergement gratuite et simple pour les sites statiques.

7.1. Qu'est-ce que le déploiement ?

Le déploiement est l'étape finale du développement web qui permet de :

- Rendre le site accessible à tout le monde via une URL
- Passer de l'environnement de développement (ordinateur local) à la production (serveur en ligne)
- Permettre aux utilisateurs de visiter le site depuis n'importe où dans le monde

7.1.1. Différence entre développement local et production

TABLE 7.1 – Développement local vs Production

Développement local	Production (en ligne)
Fichiers sur votre ordinateur	Fichiers sur un serveur
Accessible uniquement par vous	Accessible par tout le monde
URL : localhost ou file :///	URL : https ://site.com
Modifications instantanées	Nécessite redéploiement

7.2. Choix de GitHub Pages

Pour héberger notre site, nous avons choisi GitHub Pages, une solution gratuite proposée par GitHub.

7.2.1. Pourquoi GitHub Pages ?

Avantages :

- **Gratuit** : Hébergement illimité sans frais
- **Simple** : Déploiement automatique en quelques clics
- **HTTPS gratuit** : Certificat SSL automatique pour sécuriser le site

- **CDN intégré** : Le site se charge rapidement partout dans le monde
- **Pas de publicité** : Contrairement aux hébergeurs gratuits classiques
- **Intégration Git** : Mise à jour facile via Git

Limitations :

- Uniquement pour les sites statiques (HTML, CSS, Javascript)
- Pas de base de données
- Pas de PHP ou autres langages serveur

Pour notre projet (site vitrine statique), GitHub Pages est parfaitement adapté.

7.2.2. Alternatives considérées

Nous avons comparé plusieurs solutions :

TABLE 7.2 – Comparaison des solutions d'hébergement

Service	Prix	Complexité
GitHub Pages	Gratuit	Facile
Netlify	Gratuit	Facile
Vercel	Gratuit	Moyenne
Hostinger	2-10€/mois	Difficile

GitHub Pages a été retenu pour sa simplicité et son intégration native avec Git.

7.3. Étapes du déploiement

Le déploiement de notre site sur GitHub Pages s'est fait en plusieurs étapes simples.

7.3.1. Étape 1 : Création du dépôt GitHub

Nous avons créé un dépôt (repository) sur GitHub pour stocker notre code :

1. Connexion à github.com
2. Clic sur "New repository"
3. Nom du dépôt : ecotourismMaroc
4. Visibilité : Public (obligatoire pour GitHub Pages gratuit)
5. Création du dépôt

7.3.2. Étape 2 : Initialisation de Git en local

Sur notre ordinateur, dans le dossier du projet :

Explication des commandes :

- `git init` : Initialise un nouveau dépôt Git
- `git add .` : Ajoute tous les fichiers au suivi Git
- `git commit -m "..."` : Enregistre les modifications avec un message
- `git remote add origin ...` : Connecte le dépôt local à GitHub
- `git push` : Envoie le code sur GitHub

7.3.3. Étape 3 : Activation de GitHub Pages

Dans les paramètres du dépôt sur GitHub :

1. Aller dans **Settings** (paramètres du dépôt)
2. Cliquer sur **Pages** dans le menu latéral
3. Dans "Source", sélectionner :
 - Branch : `main`
 - Folder : `/` (root)
4. Cliquer sur **Save**
5. Attendre 1-2 minutes

GitHub Pages génère automatiquement le site à l'adresse :

`https://mouad-arr.github.io/ecotourismMaroc/`

7.3.4. Étape 4 : Vérification du déploiement

Après quelques minutes, nous avons :

1. Ouvert l'URL du site dans le navigateur
2. Vérifié que toutes les pages s'affichent correctement
3. Testé les liens de navigation
4. Vérifié que les images se chargent
5. Testé le menu mobile
6. Vérifié le formulaire de contact

Résultat : Le site est en ligne et accessible publiquement! ☐

7.4. Mise à jour du site

Un des avantages de GitHub Pages est la facilité de mise à jour. Chaque fois que nous modifions le code et que nous le poussons sur GitHub, le site se met à jour automatiquement.

7.4.1. Processus de mise à jour

Listado 7.1 – Mise a jour du site

```
1 # 1. Faire des modifications dans le code
2
3 # 2. Voir les fichiers modifiés
4 git status
5
6 # 3. Ajouter les modifications
7 git add .
8
9 # 4. Commit avec un message descriptif
10 git commit -m "Ajout: nouvelle destination Oasis du Sud"
11
12 # 5. Envoyer sur GitHub
13 git push origin main
14
15 # 6. Attendre 1-2 minutes : le site se met à jour automatiquement
```

Temps de déploiement : Entre 30 secondes et 2 minutes après le push.

7.4.2. Exemple de mise à jour

Nous avons fait plusieurs mises à jour après le déploiement initial :

- Correction de fautes d'orthographe
- Ajout de nouvelles images
- Amélioration du formulaire de contact
- Optimisation du menu mobile

Chaque mise à jour a suivi le même processus simple : modifier → commit → push.

7.5. Sécurité : HTTPS automatique

GitHub Pages active automatiquement HTTPS (protocole sécurisé) pour notre site. C'est très important pour plusieurs raisons :

7.5.1. Qu'est-ce que HTTPS ?

HTTPS (HyperText Transfer Protocol Secure) est la version sécurisée du HTTP. Il utilise le chiffrement SSL/TLS pour protéger les données.

Avantages de HTTPS :

- **Sécurité** : Les données échangées sont chiffrées
- **Confiance** : Les navigateurs affichent un cadenas vert
- **SEO** : Google favorise les sites HTTPS dans les résultats de recherche
- **Moderne** : Standard actuel du web

7.5.2. Configuration automatique

Avec GitHub Pages :

1. Le certificat SSL est généré automatiquement
2. Il se renouvelle automatiquement
3. Aucune configuration manuelle nécessaire
4. C'est totalement gratuit

Il suffit de cocher "Enforce HTTPS" dans les paramètres GitHub Pages (déjà activé par défaut).

7.5.3. Vérification du code

Nous avons vérifié que notre code HTML et CSS est correct :

- **HTML** : Validation avec le W3C Validator (validator.w3.org)
- **CSS** : Validation avec le CSS Validator
- **Javascript** : Vérification des erreurs dans la console du navigateur

Toutes les erreurs détectées ont été corrigées avant le déploiement final.

7.5.4. Tests responsive

Nous avons testé l'affichage du site sur différentes tailles d'écran :

- Mobile (320px - 480px)
- Tablette (768px - 1024px)
- Desktop (> 1200px)

Tous les tests étaient positifs : le site s'affiche correctement partout.

7.6. Conclusion du chapitre

Le déploiement de notre site ÉcoTourisme Maroc sur GitHub Pages s'est déroulé avec succès. Nous avons réussi à :

- Mettre le site en ligne gratuitement
- Obtenir une URL publique et un certificat HTTPS
- Établir un processus simple pour les mises à jour futures
- Optimiser le site pour de bonnes performances

Le site est maintenant accessible à l'adresse <https://mouad-arr.github.io/ecotourismMaroc/> et peut être consulté par n'importe qui dans le monde.

Cette expérience de déploiement nous a permis de comprendre :

- Le fonctionnement de Git et GitHub
- La différence entre développement local et production
- L'importance de l'optimisation pour le web
- Le processus complet de mise en ligne d'un site web

Conclusion générale

Le projet ÉcoTourisme Maroc nous a permis de découvrir et de pratiquer les technologies fondamentales du développement web moderne : HTML5, CSS3 et Javascript. À travers ce projet, nous avons créé un site web complet, du code initial au déploiement en ligne.

Objectifs atteints

Nous avons réussi à créer un site web fonctionnel qui répond aux objectifs fixés :

- **Site responsive** : Le site s'adapte correctement aux mobiles, tablettes et ordinateurs
- **Navigation intuitive** : Menu clair avec des liens fonctionnels vers toutes les pages
- **Design attractif** : Interface moderne avec des couleurs cohérentes et des images de qualité
- **Interactivité Javascript** : Menu mobile, slider, formulaire validé, bouton retour en haut
- **Déploiement réussi** : Site en ligne et accessible publiquement

Compétences acquises

Ce projet nous a permis d'apprendre et de pratiquer de nombreuses compétences techniques :

HTML

- Structure sémantique des pages
- Formulaires et validation
- Organisation du contenu
- Balises meta et SEO de base

CSS

- Flexbox et CSS Grid pour les layouts
- Media Queries pour le responsive design
- Animations et transitions
- Variables CSS pour la cohérence
- Gestion des couleurs et typographie

Javascript

- Manipulation du DOM
- Gestion des événements
- Validation de formulaires
- Création de fonctionnalités interactives
- Bonnes pratiques de code

Git et déploiement

- Versioning du code avec Git
- Utilisation de GitHub
- Déploiement avec GitHub Pages
- Workflow de mise à jour

Difficultés rencontrées et solutions

En tant que débutants, nous avons rencontré plusieurs défis :

Responsive Design

Difficulté : Adapter le site aux différentes tailles d'écran

Solution : Utilisation de Flexbox, CSS Grid et Media Queries après étude de tutoriels et exemples

Javascript

Difficulté : Comprendre la logique de programmation et la manipulation du DOM

Solution : Découpage en petites fonctions, tests fréquents dans la console, documentation MDN

Git

Difficulté : Comprendre les concepts de commit, push, branches

Solution : Apprentissage progressif des commandes de base, aide du professeur

Points forts du projet

- **Fonctionnalité complète** : Le site contient toutes les pages prévues et toutes les fonctionnalités fonctionnent
- **Design cohérent** : Charte graphique respectée sur toutes les pages
- **Accessibilité** : Navigation au clavier, attributs ARIA, messages d'erreur clairs
- **Performance** : Images optimisées, code validé
- **En ligne** : Site déployé et accessible publiquement

Améliorations futures possibles

Si nous devions continuer à développer ce site, nous pourrions ajouter :

Court terme

- Plus de destinations et d'activités
- Galerie photos avec lightbox
- Système de recherche de destinations
- Blog avec articles sur l'écotourisme
- Page FAQ (questions fréquentes)

Moyen terme

- Backend avec base de données

- Système de réservation en ligne
- Comptes utilisateurs
- Système de commentaires et avis
- Newsletter

Long terme

- Application mobile
- Carte interactive avec géolocalisation
- Système de recommandations personnalisées
- Partenariats avec éco-lodges
- Intégration de paiement en ligne

Impact du projet

Au-delà de l'apprentissage technique, ce projet a un objectif de sensibilisation au tourisme durable au Maroc. Notre site vise à :

- Promouvoir les destinations naturelles marocaines
- Encourager des pratiques de voyage responsables
- Valoriser le patrimoine culturel et environnemental
- Soutenir les communautés locales

Remerciements

Nous tenons à remercier :

- Le Professeur Qazdar Aimad pour son encadrement et ses conseils
- L'ENSA pour la formation en technologies web
- Nos camarades pour les échanges et l'entraide

Conclusion finale

Ce projet a été une expérience d'apprentissage très enrichissante. Partir d'une page blanche et arriver à un site web complet et déployé en ligne nous a donné confiance en nos capacités de développement web.

Nous avons découvert que créer un site web demande de la rigueur, de la patience et de la créativité. Chaque problème rencontré nous a appris quelque chose de nouveau. Le résultat final, bien qu'il soit celui de débutants, est fonctionnel et nous en sommes fiers.

Cette expérience nous a donné envie de continuer à apprendre le développement web et à améliorer nos compétences. Le site ÉcoTourisme Maroc est maintenant en ligne à l'adresse :

<https://mouad-arr.github.io/ecotourismMaroc/>

C'est le début d'une aventure dans le monde du développement web, et nous sommes motivés pour continuer à apprendre et à créer.