

Rapport de Projet – Phase 1

1. Répartition des tâches

Membre du groupe	Tâches principales
Church William	Pages circuits et inscription, organisation des fichiers
Kuganesan Arun	Début du site, choix des couleurs, pages d'accueil et admin
Bouhou Haytham	Pages de profil et de connexion

2. Problèmes rencontrés

2.1 Problèmes humains

- Désaccord sur les couleurs du site (Arun voulait les changer, les autres non)
- Décalage dans les moments de travail, compliquant la communication

2.2 Problèmes organisationnels

- Travail collaboratif difficile via Git : chacun travaillait localement en parallèle
- Risques de conflits lors des modifications simultanées

2.3 Problèmes techniques

- Conflits de version et de synchronisation sur Git
 - Fusion difficile en cas de travail simultané
-

3. Solutions et améliorations

- Organisation du travail : chacun évite de modifier les mêmes fichiers en même temps
 - Choix collectif d'une palette de couleurs pour limiter les désaccords
 - Meilleure planification des sessions de développement
-

4. Archivage

- Le projet est versionné via **Git**
 - Les mises à jour sont enregistrées par **commit**, avec création de **branches** si besoin
 - Certaines versions sont aussi conservées en formats **ODT / DOCX** pour le rapport
 - Suivi des modifications pour éviter les conflits
-

5. Mise à jour

- **Date de mise à jour** : 16/02/2025

Changements apportés :

- Ajout des premiers fichiers du projet
- Documentation des premiers problèmes rencontrés (techniques, humains, organisationnels)
- Liste des solutions mises en place pour faciliter le travail collaboratif
- Mise à jour de la répartition des tâches

Rapport de Projet – Phase 2

1. Répartition des tâches

Membre du groupe	Tâches principales
Church William	PHP (toutes les pages), CSS, gestion JSON
Kuganesan Arun	HTML, CSS, gestion des fichiers JSON de voyages, PHP (connexion)
Bouhou Haytham	JSON (utilisateurs et voyages), gestion des images

2. Format des données

Pour cette phase, les données sont structurées en **fichiers JSON** pour faciliter la sérialisation, la gestion et la flexibilité du stockage.

2.1 Fichier `utilisateurs.json`

Contient un tableau d'utilisateurs, avec les champs suivants :

- `id` : identifiant unique
- `nom`, `prénom`
- `email` : utilisé pour l'authentification
- `mot_de_passe` : haché avec `bcrypt`
- `date_naissance`, `date_inscription`
- `admin` : booléen (`true` si admin)
- `nombre_voyages`
- `telephone`
- `historique` (*facultatif*) : liste des voyages effectués (titre, prix, fichier)

2.2 Fichiers `voyage01.json`, `voyage02.json`, etc.

Chaque voyage est un fichier JSON indépendant contenant :

- `titre`
- `dates` : { `debut`, `fin`, `durée` }
- `étapes` : liste d'étapes avec dates, localisation et options (hébergement, activités, etc.)
- `prix_total`
- `statut` : "payé", "en cours de modification", etc.

2.3 Fichier `paiements.json`

Contient l'historique des paiements simulés :

- `coordonnées_bancaires` : (CB, expiration, CVV — données simulées)
- `utilisateur` : ID utilisateur
- `voyage` : ID voyage
- `date_transaction`

2.4 Pourquoi JSON ?

- Format simple, léger et lisible
 - Nativement pris en charge par PHP (`json_encode` / `json_decode`)
 - Parfait pour les échanges de données structurées dans un projet web
-

3. Fonctionnalités Implémentées

3.1 Inscription & Connexion

- Vérification des données utilisateur
- Hash sécurisé du mot de passe
- Stockage dans `utilisateurs.json`
- Session PHP pour la connexion

3.2 Gestion des rôles

- Utilisateurs avec ou sans droits administrateurs (`admin: true/false`)
- Accès restreint pour les pages sensibles

3.3 Affichage des voyages

- Liste synthétique : titre, date, prix, nombre d'étapes
- Affichage dans la page d'accueil ou une page dédiée

3.4 Recherche

- Barre de recherche filtrant les voyages par titre

3.5 Personnalisation

- Choix d'options par étape : hébergement, activités, transport
- Préparation du voyage avant validation

3.6 Récapitulatif et Paiement

- Récapitulatif personnalisé du voyage sélectionné
 - Saisie simulée des coordonnées bancaires
 - Enregistrement dans `paiements.json` si validation
-

4. Tests et Validation

4.1 Utilisateurs de test

Administrateur

- Email : `zz@gmail.com` — Mot de passe : `ZZ`
- `thomas@gmail.com` – Mot de passe : `P`

Utilisateurs

- `caryl@gmail.com` – Mot de passe : `LEBEST`
- `2@gmail.com` – Mot de passe : `2`
- `pierre@gmail.com` – Mot de passe : `Brique`

4.2 Voyages de test

- Plusieurs voyages test générés en JSON pour simuler l'ensemble des étapes :
 - consultation
 - personnalisation
 - récapitulatif
 - paiement simulé

Rapport de Projet – Phase 3

1. Répartition des tâches

Membre du groupe	Tâches réalisées
Church William	Filtre et Trie dynamique en JS, JS du profil, affichage récapitulatif, cookies de thème, Edition de profil en direct
Kuganesan Arun	Vérification JS des formulaires, charte graphique dynamique, CSS
Bouhou Haytham	Panier dynamique (JS+PHP)/ Supression AJAX, page admin avec simulation d'attente

2. Fonctionnalités dynamiques ajoutées (JavaScript – Front-End)

2.1 Changement de thème dynamique

- Bouton permettant de basculer entre thème sombre et clair sans rechargement.
- Choix enregistré dans un cookie.
- À chaque chargement, le thème est restauré automatiquement selon ce cookie.

2.2 Vérification des formulaires (inscription & connexion)

- Vérifications JS avant l'envoi au serveur.
- Messages d'erreur affichés dynamiquement sous les champs concernés.
- Icône œil pour masquer/afficher les mots de passe.
- Compteurs de caractères en temps réel pour le mot de passe et le pseudo.

2.3 Edition directe du profil

- Champs du profil non modifiables par défaut.
- Boutons individuels pour activer ou annuler l'édition.
- Affichage conditionnel d'un bouton "Soumettre" si au moins un champ est modifié.

2.4 Simulation d'attente sur page admin

- Lors de la modification du statut admin de l'utilisateur, un délai simulé est appliqué (désactivation temporaire du bouton).
- Réactivation automatique après quelques secondes (changement du bouton "rendre admin"/"retirer admin").


2.5 Tri dynamique des résultats de recherche

- Possibilité de trier (prix, durée, nombres d'étapes) et filtrer (Epoque, lieu, prix) les voyages sans rechargement de la page.

2.6 Recalcul dynamique du prix d'un voyage

- À chaque modification (options, nombre de personnes), le prix estimé est recalculé en JavaScript en direct.
- La valeur reste cohérente avec celle du serveur à l'étape de paiement.

3. Fonctionnalité de Panier (PHP + JS)

- À chaque fois qu'un voyage est personnalisé, il est ajouté au panier côté serveur via la session.
- Le panier est visible dans le header (icône ) avec un menu déroulant :
 - Voyages non encore payés y sont listés.
 - Chaque entrée comprend : image, titre, lien vers le paiement, et bouton de suppression.
- La suppression s'effectue en AJAX :
 - Mise à jour dynamique de l'interface sans rechargement.
 - Actualisation du compteur d'éléments dans le panier.
 - Message de confirmation affiché en popup JS.

4. Organisation du code

- Tout le JavaScript a été déplacé dans des fichiers `.js` externes.
- Le CSS a été organisé pour supporter plusieurs thèmes (foncé/clair), mais sans être optimal. Le thème foncé a été ajouté à tous les fichiers, plutôt de le mettre dans un unique fichier, cela sera corrigé à la face 4.
- Le panier utilise `panier.php`, `panier.css`, `panier.js` et `supprimer_panier.php` pour une architecture claire.

- Sécurisation des accès aux fichiers `panier.php` et `supprimer_panier.php` via redirection (`index.php`) si appel direct non autorisé.
-

5. Tests et validation

- Tests réalisés sur :
 - les formulaires (validation, accessibilité, feedback utilisateur) ;
 - la persistance du thème via cookies ;
 - le bon fonctionnement du panier dynamique (ajout, suppression, compteur) ;
 - le recalcul du prix lors des personnalisations ;
 - l'affichage et la réactivité du tri.
 - Tests utilisateurs effectués avec des comptes de test pré-créés.
-

6. Bilan de la phase 3

- Le site a considérablement gagné en réactivité et ergonomie grâce au JavaScript.
 - Toutes les fonctionnalités demandées par la phase 3 ont été traitées.
 - Le projet respecte désormais une architecture propre (fichiers JS/CSS séparés, PHP structuré).
-