

DOSSIER PROFESSIONNEL (DP)

Nom de naissance ☐ BRUNET
Nom d'usage ☐
Prénom ☐ Bastien
Adresse ☐ 24 rue Charles Ponçy
83 000 TOULON

Titre professionnel visé

Développeur web et web mobile

MODALITÉ D'ACCÈS :

- ☒ Parcours de formation
- ☐ Validation des Acquis de l'Expérience (VAE)

Présentation du dossier

Le dossier professionnel (DP) constitue un élément du système de validation du titre professionnel.
Ce titre est délivré par le Ministère chargé de l'emploi.

Le DP appartient au candidat. Il le conserve, l'actualise durant son parcours et le présente **obligatoirement à chaque session d'examen.**

Pour rédiger le DP, le candidat peut être aidé par un formateur ou par un accompagnateur VAE.

Il est consulté par le jury au moment de la session d'examen.

Pour prendre sa décision, le jury dispose :

1. des résultats de la mise en situation professionnelle complétés, éventuellement, du questionnaire professionnel ou de l'entretien professionnel ou de l'entretien technique ou du questionnement à partir de productions.
2. du **Dossier Professionnel (DP)** dans lequel le candidat a consigné les preuves de sa pratique professionnelle.
3. des résultats des évaluations passées en cours de formation lorsque le candidat évalué est issu d'un parcours de formation
4. de l'entretien final (dans le cadre de la session titre).

[Arrêté du 22 décembre 2015, relatif aux conditions de délivrance des titres professionnels du ministère chargé de l'Emploi]

Ce dossier comporte :

- ▶ pour chaque activité-type du titre visé, un à trois exemples de pratique professionnelle ;
- ▶ un tableau à renseigner si le candidat souhaite porter à la connaissance du jury la détention d'un titre, d'un diplôme, d'un certificat de qualification professionnelle (CQP) ou des attestations de formation ;
- ▶ une déclaration sur l'honneur à compléter et à signer ;
- ▶ des documents illustrant la pratique professionnelle du candidat (facultatif)
- ▶ des annexes, si nécessaire.

Pour compléter ce dossier, le candidat dispose d'un site web en accès libre sur le site.

 <http://travail-emploi.gouv.fr/titres-professionnels>

Sommaire

Exemples de pratique professionnelle

Intitulé de l'activité-type n° 1

p.

5

- Intitulé de l'exemple n° 1p.....
- Intitulé de l'exemple n° 2p.....
- Intitulé de l'exemple n° 3p.....

Intitulé de l'activité-type n° 2

p.

- Intitulé de l'exemple n° 1p.....
- Intitulé de l'exemple n° 2p.....
- Intitulé de l'exemple n° 3p.....

Intitulé de l'activité-type n° 3

p.

- Intitulé de l'exemple n° 1p.....
- Intitulé de l'exemple n° 2p.....
- Intitulé de l'exemple n° 3p.....

Intitulé de l'activité-type n° 4

p.

- Intitulé de l'exemple n° 1p.....
- Intitulé de l'exemple n° 2p.....
- Intitulé de l'exemple n° 3p.....

Titres, diplômes, CQP, attestations de formation *(facultatif)*

p.

Déclaration sur l'honneur

p.

Documents illustrant la pratique professionnelle *(facultatif)*

p.

Annexes *(Si le RC le prévoit)*

p.

EXEMPLES DE PRATIQUE PROFESSIONNELLE

Activité-type 1

Développer la partie front-end d'une application web ou web mobile en intégrant les recommandations de sécurité.

CP 1 ▶

Installer et configurer son environnement de travail en fonction du projet web ou web mobile

1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :

Dans le cadre de ma formation, j'ai réalisé deux projets front-end :

- La **maquette HTML/CSS d'une page d'accueil pour un site de voyage**
- La **création de la page principale de mon portfolio personnel**

Pour pouvoir travailler efficacement sur ces projets, j'ai installé et configuré un environnement de développement adapté à chaque besoin en installant en premier lieu l'éditeur de code VS code. J'ai ensuite commencé par structurer les dossiers de travail (création des répertoires `css`, `images`, `assets`, etc.) et mis en place un système de versionnage avec Git.

J'ai également pris en compte les bonnes pratiques en matière de sécurité front-end de base, j'ai veillé à ne pas inclure d'informations sensibles dans les fichiers sources; en ne jamais inclure dans le front-end de clés API, d'identifiants de connexion ou mots de passe, emails, tokens, infos personnelles. Ces projets ont été réalisés dans un **contexte pédagogique**, avec des consignes précises fournies par les formateurs.

2. Précisez les moyens utilisés :

Éditeur de code : Visual Studio Code

- **Extensions utilisées** : Live Server (pour un rechargement automatique), Prettier (pour le formatage du code), HTML/CSS Support
- **Navigateurs pour les tests** : Google Chrome et Mozilla Firefox
- **Outils de versionnage** : Git (en local)
- **Organisation du projet** : arborescence de fichiers claire et modulaire (`index.html`, `style.css`, dossier `assets` permettant la séparation des fichiers de code rendant le projet plus lisible et facilitant la recherche d'une image, d'une police etc ...)
- **Normes et bonnes pratiques HTML/CSS** : indentation, commentaires, accessibilité de base, compatibilité multi-navigateurs.

3. Avec qui avez-vous travaillé ?

J'ai travaillé **seul** tout au long de ces projets. Ces réalisations ont été effectuées en **autonomie**, dans le cadre d'exercices pratiques proposés durant la formation.

4. Contexte

Nom de l'entreprise, organisme ou association *La Plateforme*

Chantier, atelier, service *Maquettes en HTML et CSS*

Période d'exercice Du : *01/04/2025* au : *30/04/2025*

5. Informations complémentaires (facultatif)

Activité-type 1

Développer la partie front-end d'une application web ou web mobile sécurisée

CP 2 ▶

Maquetter des interfaces utilisateur web ou web mobile,

1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :

(Projet concerné : Page principale de mon portfolio)

1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :

Dans le cadre de la création de mon **portfolio personnel**, j'ai réalisé une **maquette d'interface utilisateur** à l'aide du langage HTML et CSS, dans une approche centrée sur l'expérience utilisateur et la clarté visuelle.

Mon objectif était de concevoir une structure d'interface claire, lisible et facile à prendre en main, en respectant les grands principes de la conception d'interface : hiérarchisation des contenus, accessibilité, et adaptabilité visuelle.

La maquette a été pensée d'abord en version **desktop**, puis j'ai décliné l'organisation des blocs en vue d'une intégration responsive. Ce travail s'est fait dans un cadre **autonome**, sur la base de consignes pédagogiques fournies.

2. Précisez les moyens utilisés :

Outils de conception : HTML5, CSS3 (code directement écrit, sans logiciel graphique de type Figma)

- Méthodologie : réflexion sur la disposition des sections (accueil, présentation, compétences, projets, contact)
- Mise en page simulée via une structure sémantique HTML : **header**, **main**, **section**, **footer**
- Outils : Visual Studio Code + extensions (Live Server, Prettier)
- Organisation des fichiers : création d'un dossier **assets/** contenant les images, icônes, polices éventuelles

3. Avec qui avez-vous travaillé ?

J'ai travaillé **seul** tout au long de ce projet, en autonomie complète.

4. Contexte

Nom de l'entreprise, organisme ou association *La Plateforme*

Chantier, atelier, service *Page principale de mon portfolio*

Période d'exercice Du : *02/04/2025* au : *30/04/2025*

5. Informations complémentaires (facultatif)



INBIO

BUY NOW



WELCOME TO MY WORLD

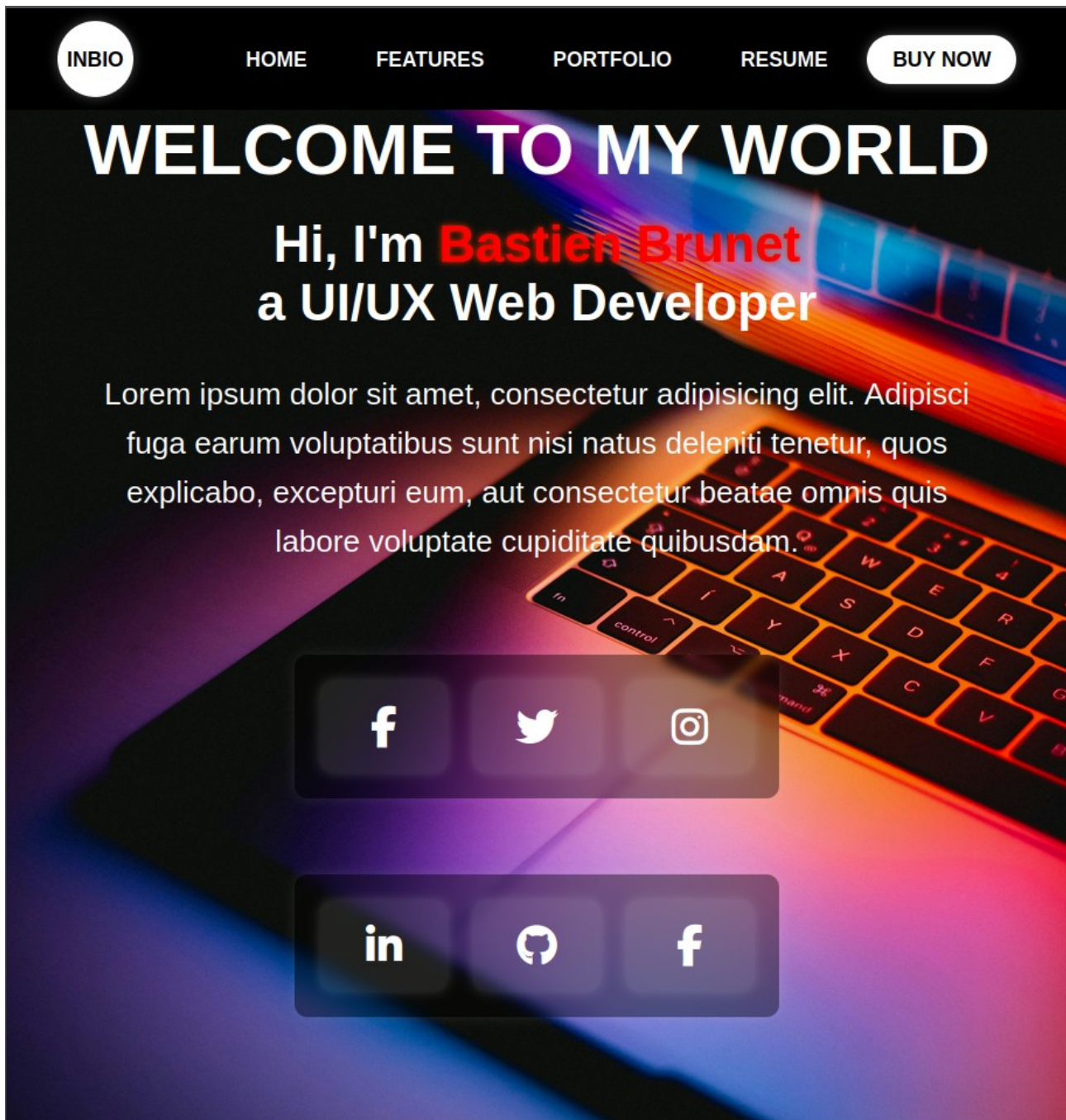
Hi, I'm **Bastien Brunet**
a UI/UX Web Developer

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Adipisci
fuga earum voluptatibus sunt nisi natus deleniti tenetur, quos
explicabo, excepturi eum, aut consectetur beatae omnis quis
labore voluptate cupiditate quibusdam.



Ci-dessus, capture d'écran en mode "mobile" (avec menu burger) et capture d'écran en mode "desktop".

Capture d'écran en mode "tablette" :



Activité-type 1

Développer la partie front-end d'une application web ou web mobile en intégrant les recommandations de sécurité.

CP 3 ► Réaliser des interfaces utilisateurs statiques web et web mobile,

1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :

Pour ces deux projets, j'ai intégré des interfaces statiques à partir des maquettes réalisées ou fournies.

Mon travail a consisté à transformer les maquettes en **pages HTML/CSS fonctionnelles, adaptables (responsive) et accessibles**, sans interaction JavaScript.

J'ai utilisé **Flexbox** et **Grid CSS** pour organiser les éléments, ajouté des **media queries** pour l'adaptabilité mobile, et respecté les règles de sémantique HTML pour renforcer l'accessibilité. J'ai également veillé à la compatibilité entre navigateurs (Chrome et Firefox) et respecté les recommandations de base en matière de sécurité front-end en évitant d'inclure adresse e-mail en clair dans le HTML (risque de spam), des liens vers des pages privées ou d'administration, des clés API ou jetons d'authentification (même de test), pas d'infos sensibles en clair dans le code, Le travail a été mené en autonomie, dans un cadre pédagogique, sur des consignes précises.

2. Précisez les moyens utilisés :

Langages : HTML5, CSS3

- **Techniques de mise en page** : Flexbox, Grid CSS
- **Adaptabilité responsive** : Media queries (@media screen and (max-width: . . .)), unités relatives (em, %, vw, vh) et insertion d'un menu burger pour le mode mobile.
- **Structure de projet** : Arborescence organisée, avec un dossier `assets/` pour les ressources (images, icônes, polices)
- **Outils** :
 - Visual Studio Code + Live Server
 - Navigateurs Chrome et Firefox pour les tests
 - Outils développeur (DevTools) pour tester les résolutions mobiles

- **Bonnes pratiques :**
- HTML sémantique (nav, main, section, footer)
- Respect des standards d'accessibilité
- Pas d'informations sensibles dans le front-end
- Respect des conventions de nommage (classes CSS claires et logiques)

3. Avec qui avez-vous travaillé ?

J'ai travaillé **seul** sur l'ensemble de ces projets. Les interfaces ont été conçues et codées en **autonomie**, en respectant les maquettes et les consignes de mise en page fournies dans le cadre de la formation.

4. Contexte

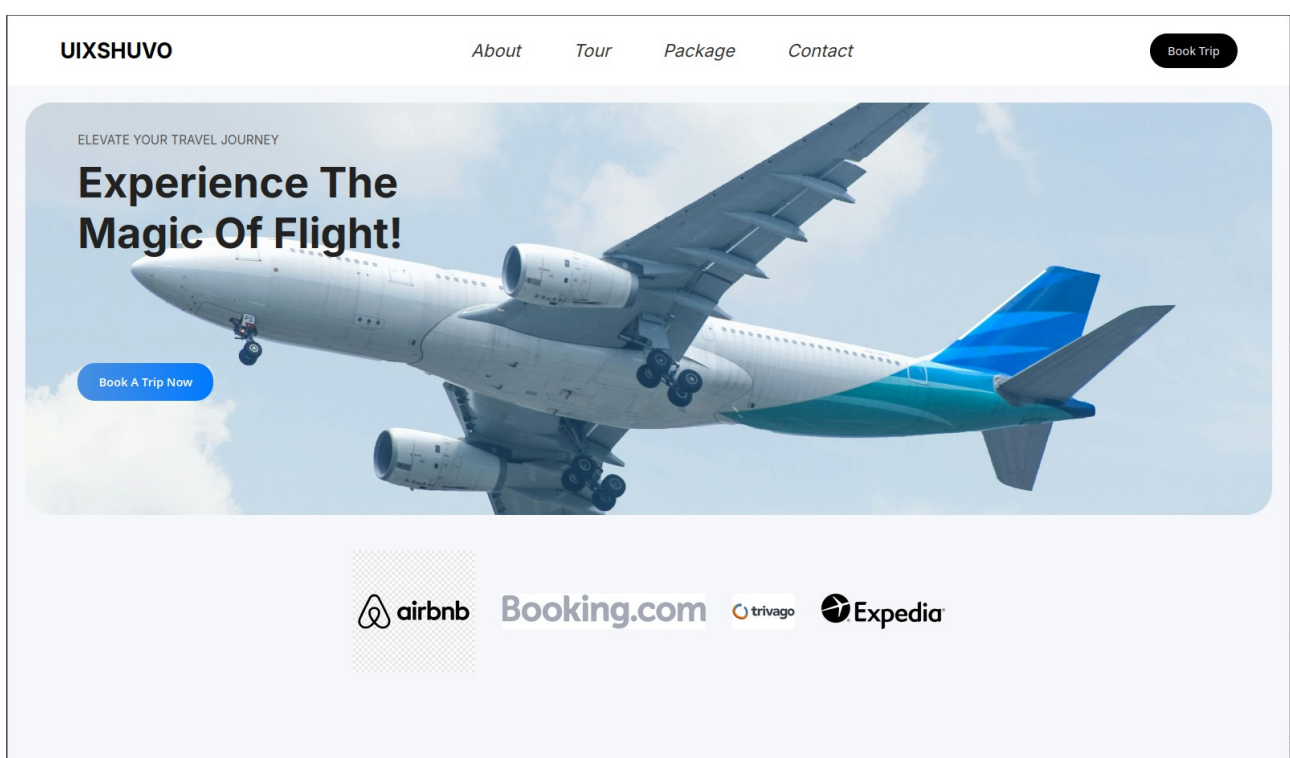
Nom de l'entreprise, organisme ou association *La Plateforme*

Chantier, atelier, service *Maquettes statiques site de voyage en HTML / CSS*

Période d'exercice ▶ Du : *02/04/2025* au : *30/04/2025*

5. Informations complémentaires (facultatif)

Voici les captures d'écran desktop et responsive :





Booking.com



Popular Destination

Unleash Your Wanderlust With SkyWings



Forest Wild Life



Forest Wild Life



Forest Wild Life

Journey To The Skies Made Simple!

Travelling is a wonderful way to explore new places, learn about different cultures, and gain unique experiences.

Find Your Destination

Book A Ticket

Pay & Start Journey

Forest Wild Life

Forest Wild Life

Forest Wild Life

Journey To The Skies Made Simple!

Travelling is a wonderful way to explore new places, learn about different cultures, and gain unique experiences.

Find Your Destination

Book A Ticket

Pay & Start Journey



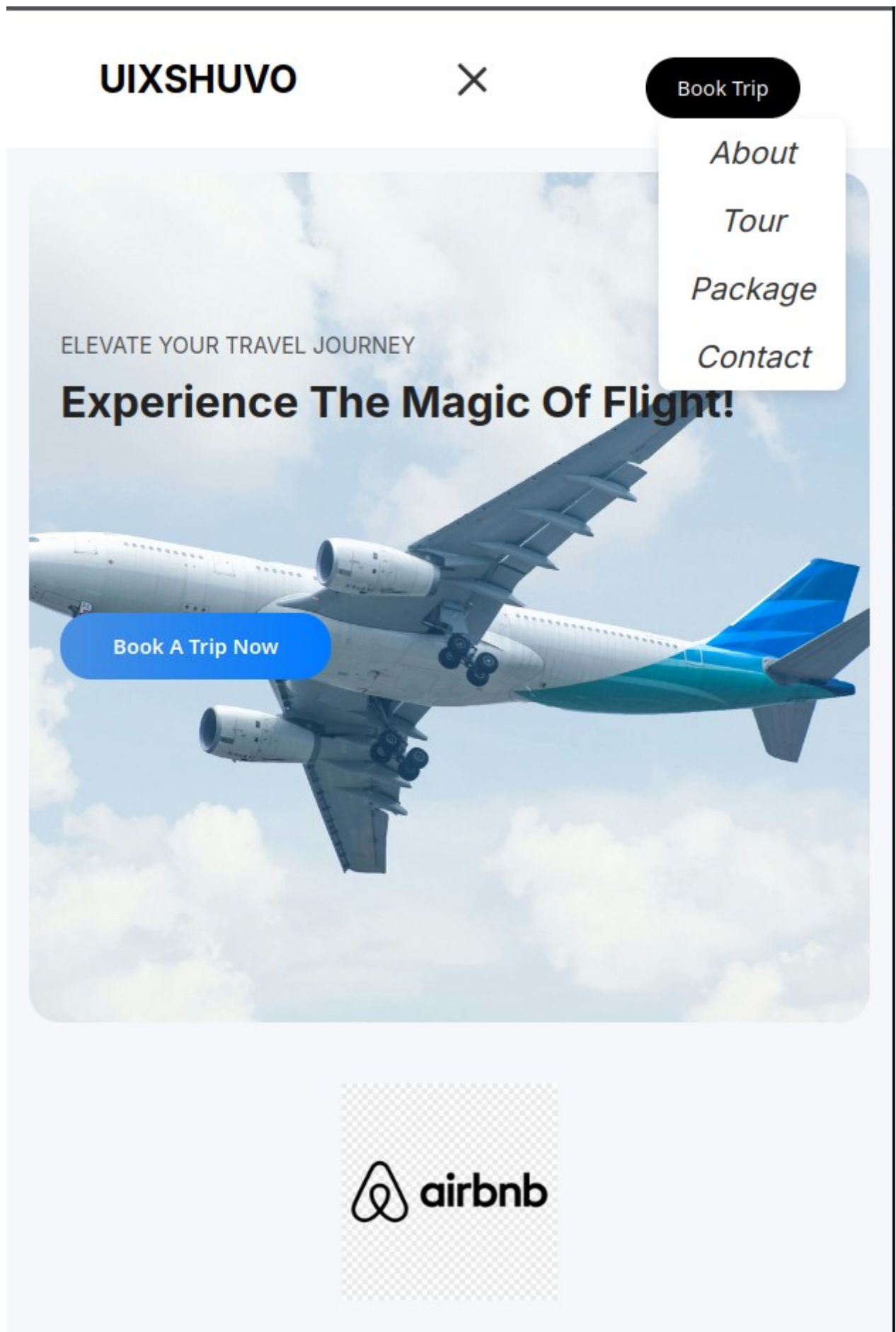
UNLEASH WANDERLUST WITH SKYWINGS

Travelling is a wonderful way to explore new places, learn about different cultures, and gain unique experiences.

20% OFF
Till 28 September, 2023

Book A Flight Now →

Et voici ci dessous les captures d'écran mobile :





UNLEASH WANDERLUST WITH SKYWINGS

Travelling is a wonderful way to explore new places, learn about different cultures, and gain unique experiences.

20% OFF

Till 28 September, 2023

Book A Flight
Now →



Booking.com



Popular Destination

Unleash Your Wanderlust With SkyWings



Forest Wild Life



UIXSHUVO



Book Trip

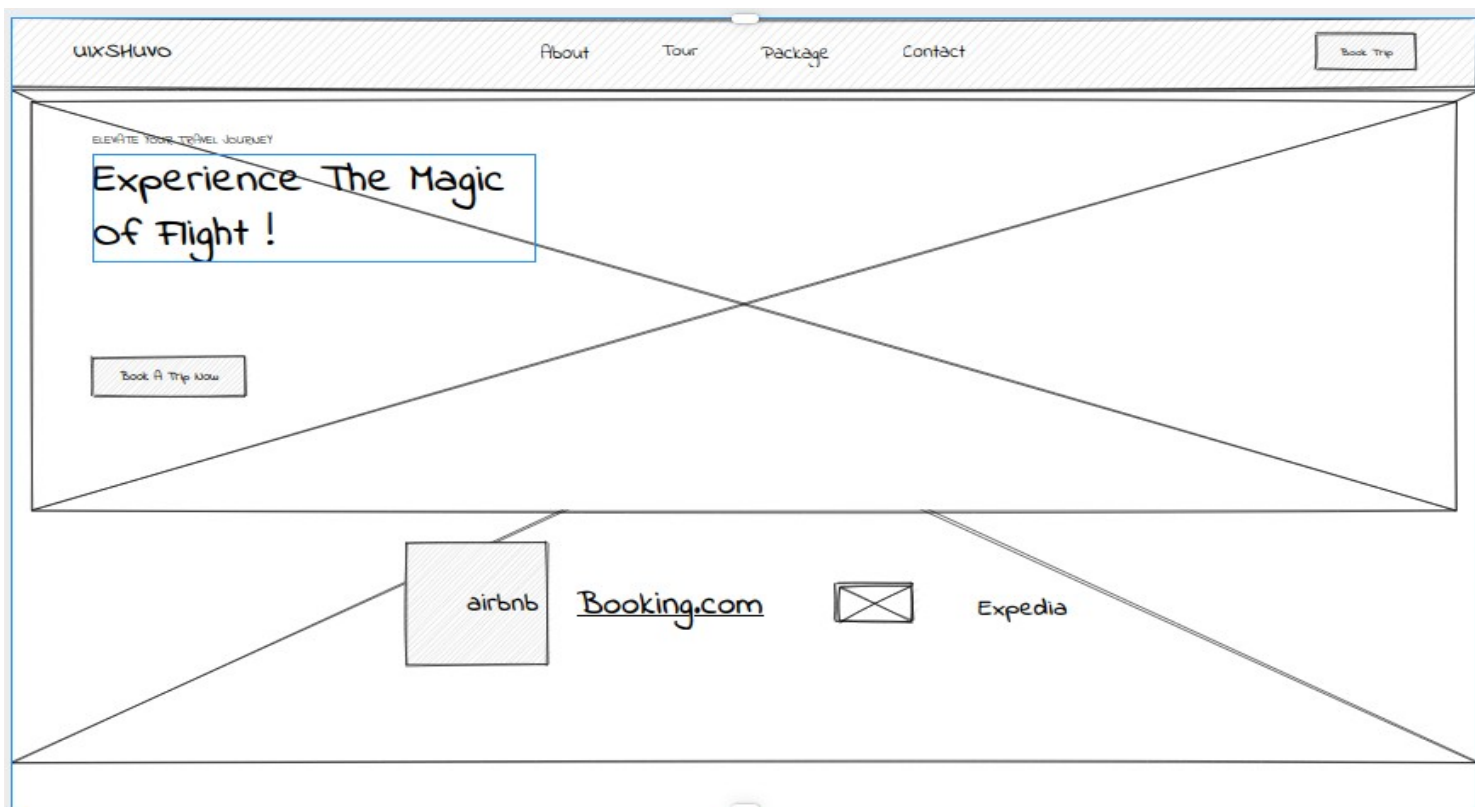
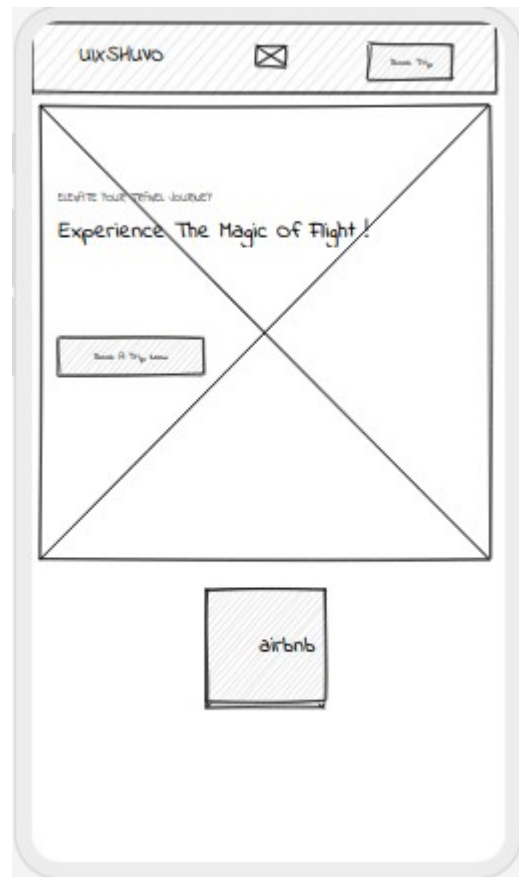
ELEVATE YOUR TRAVEL JOURNEY

Experience The Magic Of Flight!

Book A Trip Now



Voici quelques illustrations ci dessous des wireframes de cette maquette, d'abord en mode mobile puis desktop :



airbnb

Booking.com

trivago

~~Expedia~~

Popular Destination

Unleash Your wanderlust with Skywings



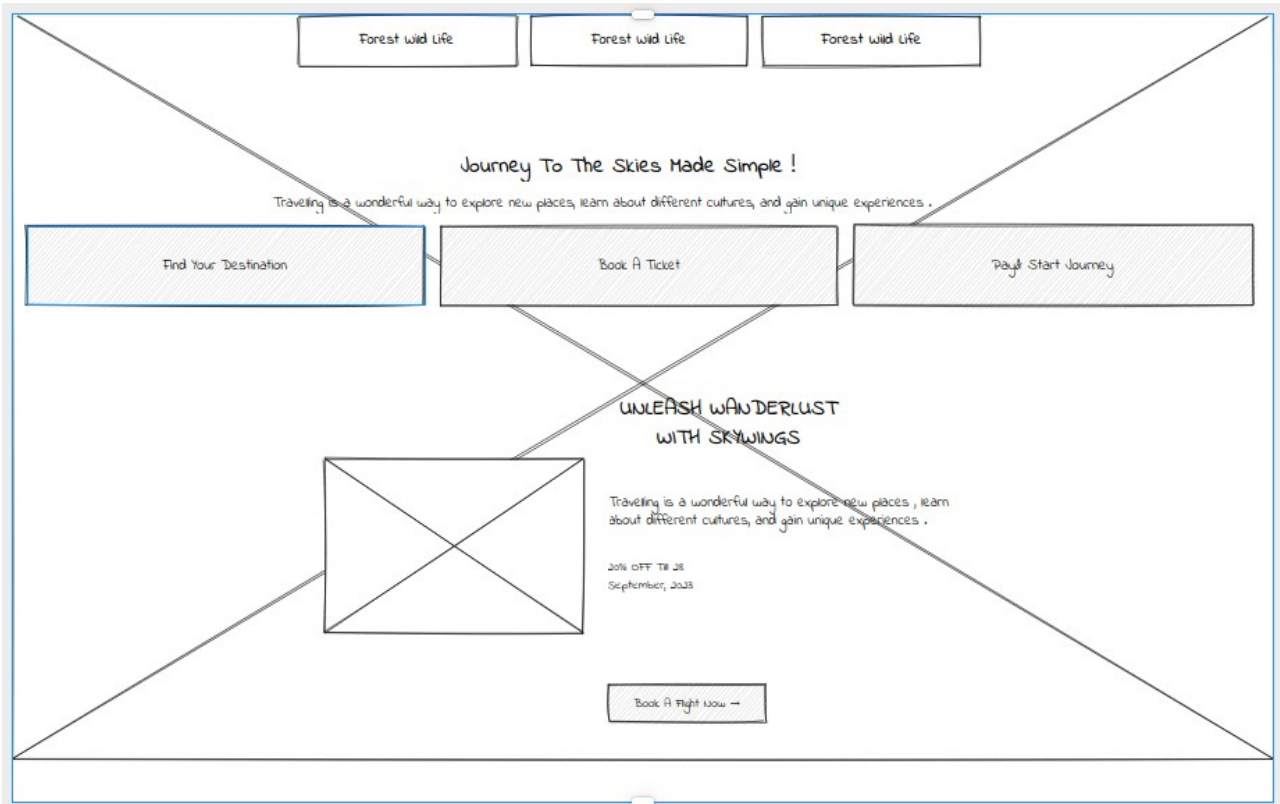
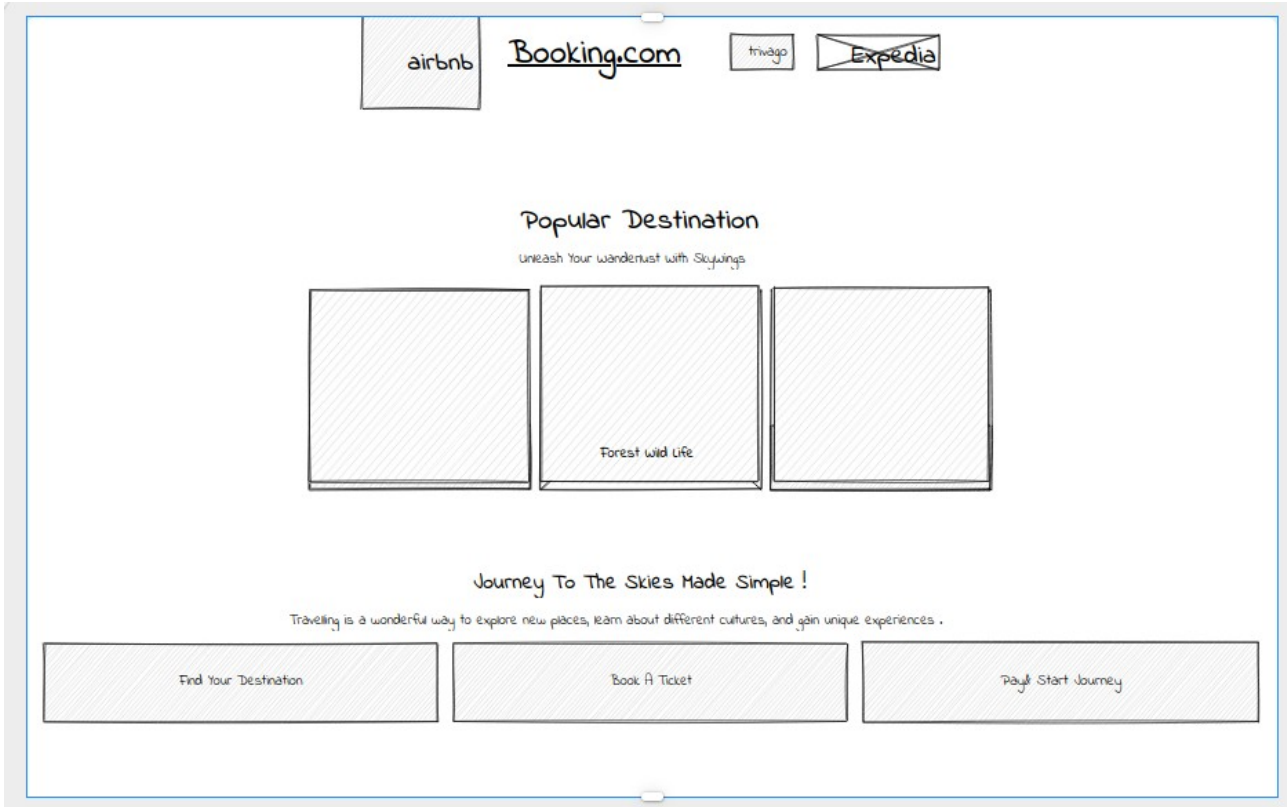
Journey To The Skies Made Simple !

Travelling is a wonderful way to explore new places, learn about different cultures, and gain unique experiences .

Find Your Destination

Book A Ticket

Pay! Start Journey



Activité-type 1

Développer la partie front-end d'une application web ou web mobile en intégrant les recommandations de sécurité.

CP 4 ▶

Développer la partie dynamique des interfaces utilisateur web ou web mobile

1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :

Dans le cadre de ce projet, j'ai conçu et développé une **application web de gestion de tâches** (*Todo List*) permettant à l'utilisateur d'ajouter, de marquer comme terminée, de supprimer et de visualiser des tâches en temps réel. L'objectif était de démontrer la capacité à rendre une interface web dynamique et interactive, sans utiliser de frameworks externes.

J'ai commencé par structurer l'interface en HTML et CSS, en m'assurant qu'elle soit à la fois esthétique et responsive. Ensuite, j'ai implémenté toute la **logique JavaScript** en programmation orientée objet à l'aide d'une classe `TodoApp`.

Les tâches ont été réalisées dans un **environnement de développement local**, en respectant une démarche de test à chaque ajout de fonctionnalité.

Enfin, j'ai intégré la **persistance des données** via le `localStorage`, permettant à l'utilisateur de retrouver sa liste de tâches même après fermeture du navigateur.

2. Précisez les moyens utilisés :

Pour ce projet, j'ai utilisé les **langages du web** suivants :

- **HTML5** : pour la structure de la page.
- **CSS3** : pour la mise en forme, la responsivité et les animations.
- **JavaScript (vanilla JS)** : pour la logique dynamique, les événements, le DOM, la manipulation des données et leur stockage local.

J'ai utilisé :

- **Visual Studio Code** comme éditeur de code.
- Le navigateur **Google Chrome** pour le test et le débogage.
- L'outil **DevTools** pour inspecter les éléments et suivre les erreurs JavaScript.
- Le **localStorage** du navigateur pour stocker les données côté client.
- Des captures d'écran pour documenter le fonctionnement de l'application.

Un fichier README .md a également été rédigé pour décrire le projet, son fonctionnement et ses principales fonctionnalités.

3. Avec qui avez-vous travaillé ?

Ce projet a été **entièrement réalisé seul**.

J'ai conçu, codé et testé l'ensemble de l'application de manière autonome, en appliquant les compétences acquises lors de ma formation, notamment pour la partie dynamique des interfaces web.

4. Contexte

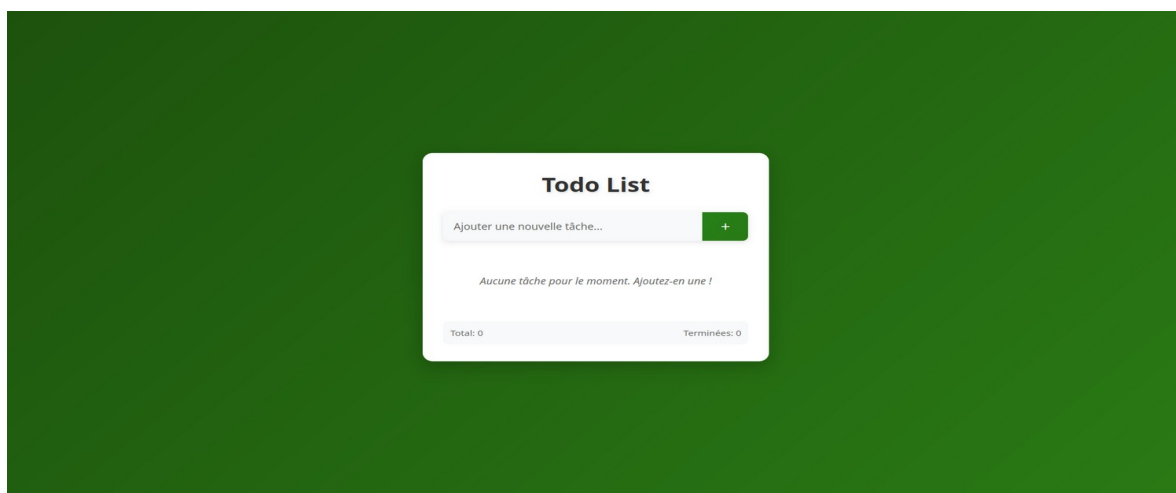
Nom de l'entreprise, organisme ou association *La Plateforme*

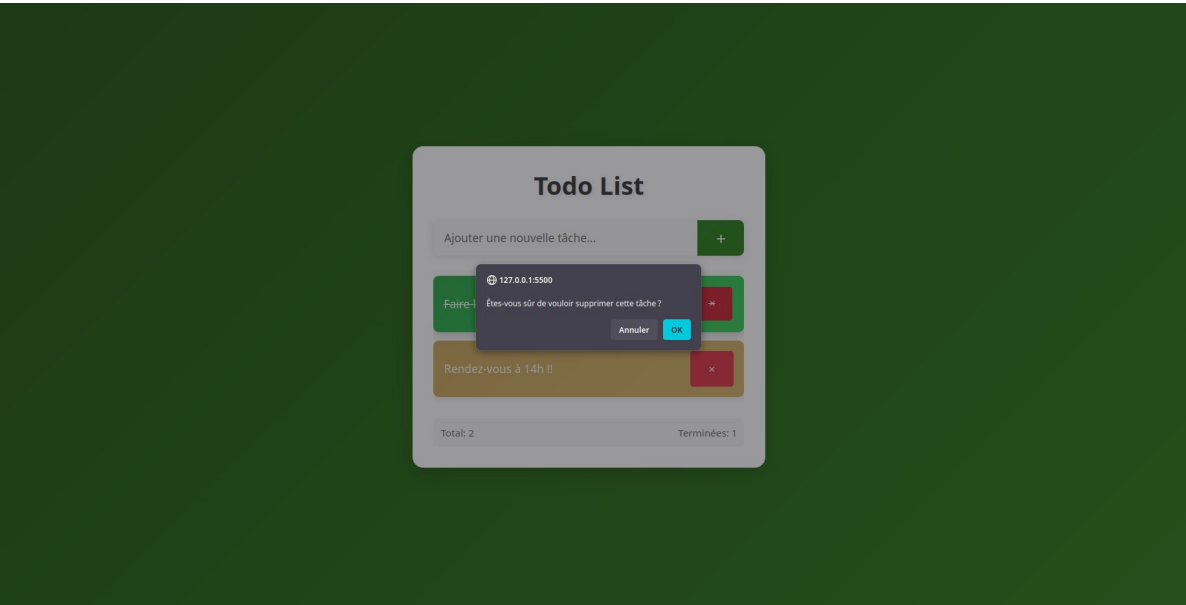
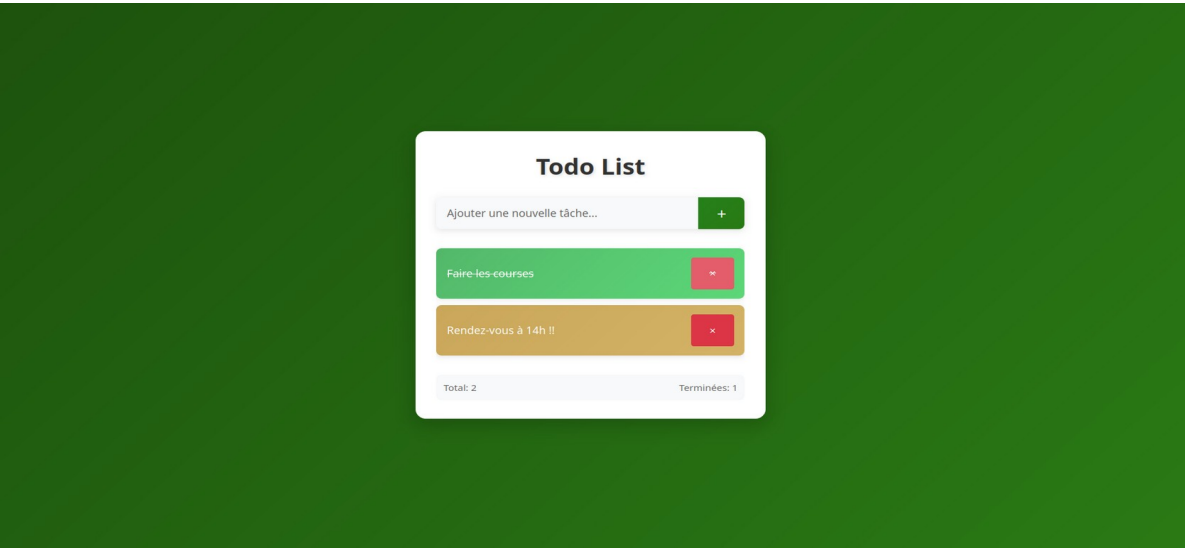
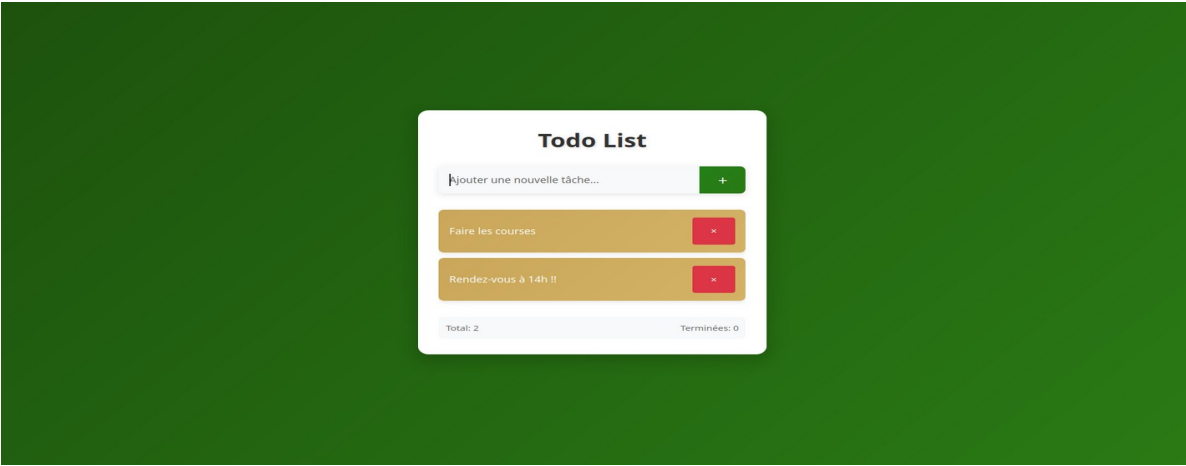
Chantier, atelier, service *Appllication de gestion des tâches (Todolist)*

Période d'exercice ► Du : *01/05/2025* au : *15/05/2025*

5. Informations complémentaires (facultatif)

Voici quelques captures d'écrans illustrant la todo-list :





Activité-type 2

Développer la partie back-end d'une application web ou web mobile sécurisée

CP 5 ➤

Mettre en place une base de données relationnelle

1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :

Tâches accomplies pour la base de données relationnelle :

1. Configuration MySQL existante :

- Fichier db.js avec pool de connexions MySQL
- Configuration complète (host, user, password, database)
- Base nommée "mini-shop-back"

2. Structure relationnelle définie : fichier init.sql :

- Tables avec relations : categories, products, orders, users
- Clés étrangères : products.categories_id → categories.id
- Tables de liaison : products_has_orders, products_has_users
- Contraintes relationnelles : ON DELETE, ON UPDATE

3. Contrôleurs opérationnels :

- productController.js, authController.js, etc.
- Requêtes SQL déjà écrites, voir requêtes SQL en partie 5 (informations complémentaires) .

2. Précisez les moyens utilisés :

Tâches accomplies pour la base de données relationnelle :

1. ANALYSE ET CONCEPTION.

- **Analyse du domaine métier** : Boutique d'instruments de musique
- **Identification des entités** : Produits, Catégories, Marques, Utilisateurs, Commandes
- **Définition des relations** : Hiérarchiques (catégories/marques), One-to-Many, Many-to-Many

Conditions :

- **Contexte** : Projet e-commerce “Mini-Shop”
- **Objectif** : Structure flexible pour site de vente en ligne.
- **Contraintes** : **Performance, sécurité, évolutivité**

Conditions fonctionnelles :

- **Domaine** : E-commerce
- **Évolutivité** : Support de nouvelles catégories
- **Flexibilité** : Spécifications JSON variables
- **Multilingue** : Prêt pour internationalisation

Conditions de développement :

- **Express.js** : API REST avec contrôleurs dédiés
- **Angular 19**: Frontend moderne
- **Migration** : Coexistence avec MySQL existant
- **Testing** : Données de test incluses

3. Avec qui avez-vous travaillé ?

Pour ce projet j’ai travaillé en autonomie total.

4. Contexte

Nom de l'entreprise, organisme ou association *La Plateforme*

Chantier, atelier, service *Boutique en ligne "Mini-shop"*

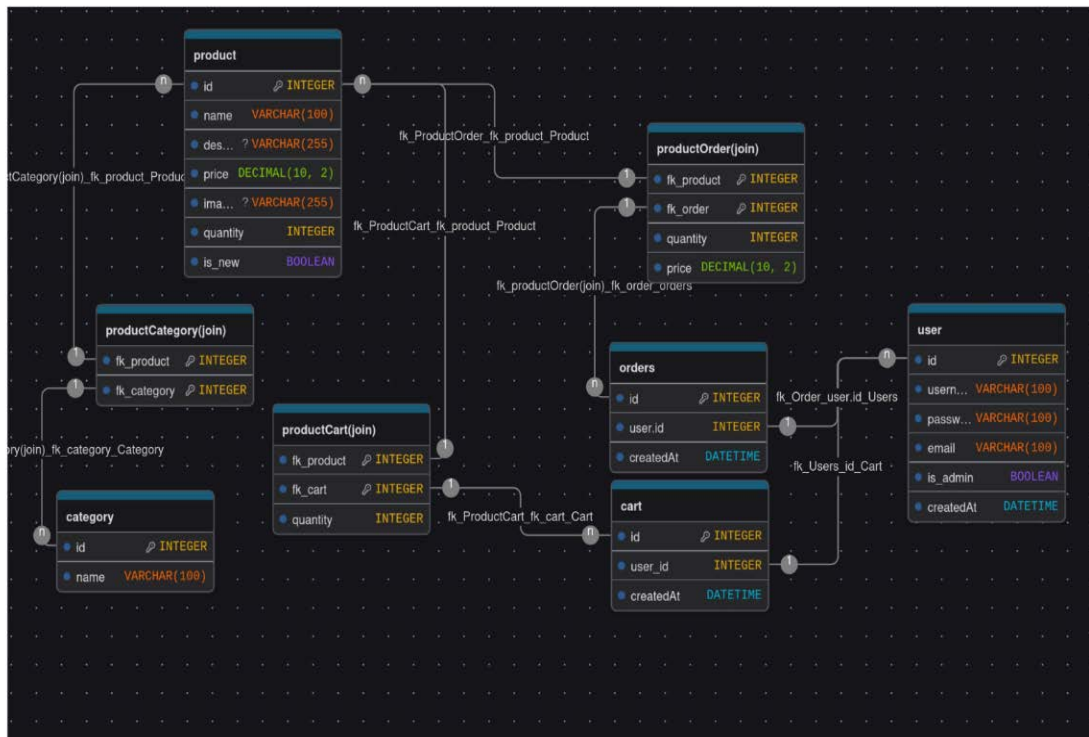
Période d'exercice Du : *17/06/2025* au : *31/08/2025*

5. Informations complémentaires (facultatif)

Requêtes SQL :

```
1  CREATE DATABASE mini_shop CHARACTER SET utf8mb4 COLLATE utf8mb4_unicode_ci;
2
3
4  CREATE TABLE users (
5      id int AUTO_INCREMENT PRIMARY KEY,
6      email VARCHAR(255) NOT NULL UNIQUE,
7      password varchar(255) NOT NULL
8  )
9
10 CREATE TABLE products (
11     id int AUTO_INCREMENT PRIMARY KEY,
12     name VARCHAR(255) NOT NULL,
13     description TEXT,
14     price DECIMAL(10, 2) NOT NULL,
15     stock int DEFAULT 0,
16     image_url VARCHAR(255)
17 );
18
19 CREATE TABLE categories(
20     id int AUTO_INCREMENT PRIMARY KEY,
21     name VARCHAR(255) NOT NULL UNIQUE,
22     description TEXT
23 );
24
25 CREATE TABLE orders (
26     id INT AUTO_INCREMENT PRIMARY KEY,
27     user_id INT,
28     order_date DATETIME,
29     FOREIGN KEY (user_id) REFERENCES users(id)
30 );
31
32
33
34 -- table pour panier --
35
36 -- Table pour stocker les paniers (un par utilisateur)
```

MCD Mini-Shop :



Activité-type 2

Développer la partie back-end d'une application web ou web mobile sécurisée

CP 6 ▶

Développer les composants d'accès aux données SQL et NoSQL

1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :

Description des tâches réalisées dans le projet Soundora :

Dans le cadre du projet Soundora (boutique en ligne d'instruments de musique et d'accessoires de musique) , j'ai développé la partie back-end de l'application en utilisant Node.js et Express, avec une base de données Supabase (PostgreSQL, SQL).

Voici les tâches et opérations que j'ai effectuées :

- Conception du schéma de base de données (tables utilisateurs, produits, catégories, commandes) dans Supabase.
- Écriture de requêtes SQL pour le CRUD (création, lecture, mise à jour, suppression) des différentes entités (exemples de requêtes SQL dans 5. Informations complémentaires),
- Développement de contrôleurs Express pour chaque entité, assurant la communication entre l'API et la base de données.
- Utilisation du SDK Supabase côté back-end pour interagir de façon sécurisée avec la base (requêtes paramétrées, gestion des erreurs).
- Mise en place de l'authentification sécurisée (Supabase Auth, JWT) et de la gestion des droits d'accès aux données.
- Gestion des opérations transactionnelles pour garantir la cohérence des commandes et des paiements.
- Tests des endpoints API avec des jeux de données réels.

2. Précisez les moyens utilisés :

Voici les moyens que j'ai utilisés :

- Environnement de développement sécurisé (variables d'environnement, gestion des clés API).
- Outils : Node.js, Express, Supabase, PostgreSQL, Postman pour les tests d'API, GitHub pour le versionnement.
- Documentation du code et des schémas de données.

3. Avec qui avez-vous travaillé ?

Travail réalisé en autonomie

4. Contexte

Nom de l'entreprise, organisme ou association *La Plateforme*

Chantier, atelier, service *Boutique en ligne d'instruments et d'accessoires de musique "Soundora"*

Période d'exercice *▶ Du : 17/06/2025 au : 31/08/2025*

5. Informations complémentaires (facultatif)

Exemples de requêtes SQL utilisées :

1. Récupérer tous les produits du catalogue :

```
SELECT * FROM products;
```

→ Cette requête retourne la liste complète des produits enregistrés dans la table **products**

2. Ajout d'un produit dans la table *products* :

```
INSERT INTO products (name, description, price, category_id, brand_id, image_url)
VALUES ('Guitare électrique', 'Guitare 6 cordes, idéale pour le rock.', 499.99, 1, 2,
'https://exemple.com/guitare.jpg');
```

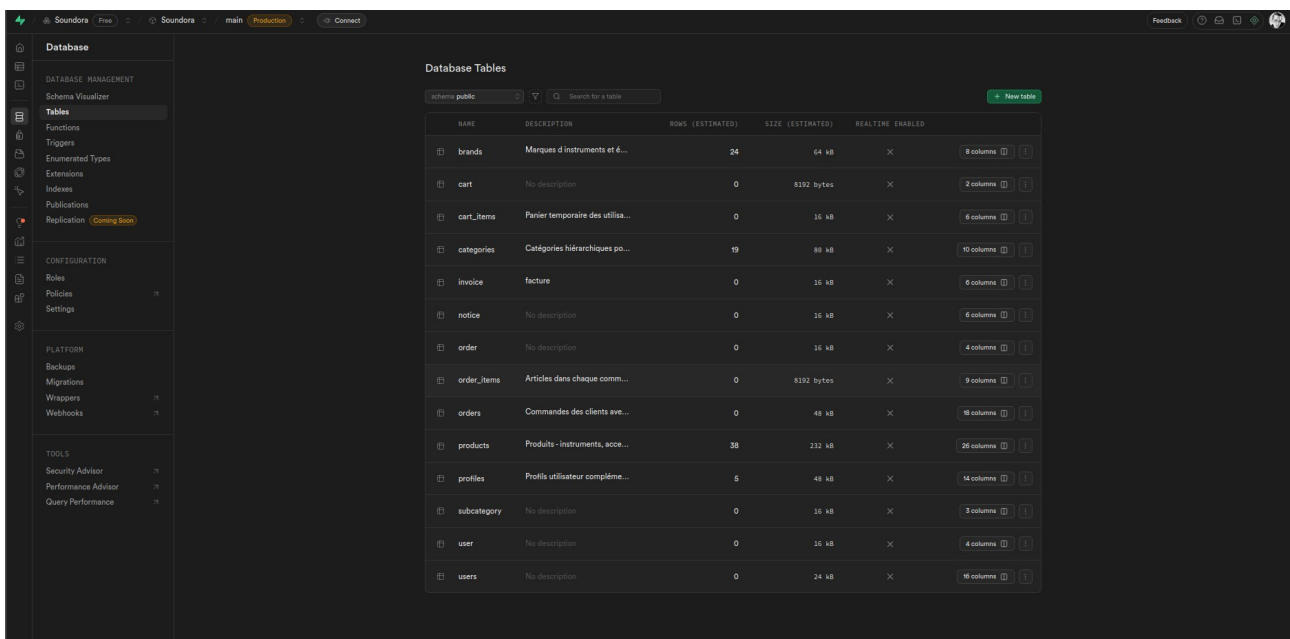
→ Cette requête ajoute un produit avec son nom, sa description, son prix, la catégorie et la marque associées, ainsi qu'une image.

3. Jointure pour afficher tous les produits avec leur catégorie et leur marque :

```
SELECT
  p.id, p.name, p.price, c.name AS category, b.name AS brand
FROM
  products p
JOIN
  categories c ON p.category_id = c.id
JOIN
  brands b ON p.brand_id = b.id;
```

→ Cette requête retourne la liste des produits avec le nom de leur catégorie et de leur marque.

Voici les tables de la database de mon site Soundora :



The screenshot shows the Soundora database management interface. On the left is a sidebar with navigation options: Database, DATABASE MANAGEMENT (Schema Visualizer), Tables, Functions, Triggers, Enumerated Types, Extensions, Indexes, Publications, Replication (Coming Soon), CONFIGURATION (Roles, Policies, Settings), PLATFORM (Backups, Migrations, Wrappers, Webhooks), and TOOLS (Security Advisor, Performance Advisor, Query Performance). The main area is titled 'Database Tables' and displays a table with columns: NAME, DESCRIPTION, ROWS (ESTIMATED), SIZE (ESTIMATED), REALTIME ENABLED, and a column for the number of columns. The table lists 15 tables in the 'public' schema.

NAME	DESCRIPTION	ROWS (ESTIMATED)	SIZE (ESTIMATED)	REALTIME ENABLED	COLUMNS
brands	Marques d'instruments et d'accessoires	24	64 kB	X	8 columns
cart	No description	0	8192 bytes	X	2 columns
cart_items	Panier temporaire des utilisateurs	0	16 kB	X	6 columns
categories	Catégories hiérarchiques pour les instruments	19	88 kB	X	10 columns
invoice	facture	0	16 kB	X	6 columns
notice	No description	0	16 kB	X	6 columns
order	No description	0	16 kB	X	4 columns
order_items	Articles dans chaque commande	0	8192 bytes	X	9 columns
orders	Commandes des clients avec adresse de livraison	0	48 kB	X	18 columns
products	Produits - instruments, accessoires	38	232 kB	X	26 columns
profiles	Profil utilisateur complet	5	48 kB	X	14 columns
subcategory	No description	0	16 kB	X	3 columns
user	No description	0	16 kB	X	4 columns
users	No description	0	24 kB	X	16 columns

Activité-type 2

Développer la partie back-end d'une application web ou web mobile sécurisée

CP 7 ▶

Développer des composants métier côté serveur

1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :

Dans le projet Soundora, j'ai conçu et développé les composants métier du back-end en Node.js/Express.

J'ai implémenté la logique métier pour la gestion des utilisateurs, des produits, des catégories, du panier, des commandes et du paiement en ligne (Stripe).

J'ai structuré le code en contrôleurs (ex : orderController.js, cartController.js, authController.js) pour séparer la logique métier de l'accès aux données; voir exemple en partie 5.

Précision de la gestion des règles métier :

- J'ai mis en place des contrôleurs côté serveur (Node.js/Express) pour centraliser la logique métier de l'application.
- Pour chaque fonctionnalité (création de commande, ajout au panier, paiement, etc.), j'ai défini des règles précises :
 - Vérification que l'utilisateur est bien authentifié avant d'accéder à certaines routes (ex : passer commande, voir l'historique).
 - Validation des données reçues du frontend : contrôle des champs obligatoires, des formats (ex : email, prix positif, quantité non nulle), et gestion des erreurs en cas de données invalides.
 - Calcul automatique du total du panier et des frais de livraison lors de la création d'une commande.
 - Gestion des stocks : vérification de la disponibilité des produits avant validation d'une commande, et mise à jour des quantités en base après achat.
 - Attribution de droits spécifiques selon le rôle de l'utilisateur (ex : seuls les admins peuvent ajouter ou supprimer des produits, seuls les clients peuvent commander).
 - Intégration du paiement Stripe : la commande n'est validée que si le paiement est accepté, grâce à la gestion des webhooks Stripe.

- Suivi de l'état des commandes (en attente, payée, expédiée, annulée) et mise à jour automatique selon les actions de l'utilisateur ou du backoffice.
- J'ai documenté chaque règle métier dans le code et dans la documentation technique pour faciliter la compréhension et la maintenance du projet.

Conditions de réalisation :

Environnement technique :

Le projet a été développé sur un poste sous Linux, avec Node.js et Angular installés localement. Le code source est versionné sur GitHub, ce qui permet un suivi des évolutions et un travail collaboratif. L'éditeur principal utilisé est Visual Studio Code, avec des extensions facilitant le développement (lint, auto-format, intégration Git...).

• Contraintes matérielles et logicielles :

Le développement nécessite un ordinateur avec une connexion Internet, Node.js, npm, et un navigateur moderne pour le front-end. L'API back-end fonctionne en local ou via Docker, et la base de données est gérée via Supabase (PostgreSQL) ou MySQL selon les besoins de test.

• Outils et ressources :

- **Gestion de version** : Git et GitHub pour le suivi, la sauvegarde et la collaboration.
- **Gestion de projet** : Kanban (GitHub Projects), documentation dans le dossier docs.
- **Tests et validation** : Postman pour tester les API, console navigateur pour le front-end, logs serveur pour le back-end.
- **Sécurité** : Utilisation de variables d'environnement (.env), gestion des tokens JWT, et authentification via Supabase.
- **Déploiement** : Docker Compose pour simuler l'environnement de production, scripts npm pour automatiser les tâches courantes.

• Organisation du travail :

Le projet a été découpé en tâches : conception de la base de données, développement du back-end (API REST, logique métier), puis du front-end (UI responsive, intégration Stripe, gestion de l'authentification). Chaque étape a été validée par des tests manuels et des revues de code.

•**Contraintes spécifiques :**

- Respect des bonnes pratiques de sécurité (protection des routes, gestion des erreurs, validation des entrées).
- Accessibilité et responsive design pour garantir une expérience utilisateur optimale sur mobile et desktop.
- Documentation systématique du code et des choix techniques pour faciliter la maintenance et la transmission du projet.

2. Précisez les moyens utilisés :

•**Environnement technique :**

- Développement du back-end avec Node.js et Express pour la gestion des routes, de la logique métier et des contrôleurs.
- Utilisation de Supabase (PostgreSQL) comme base de données relationnelle, avec gestion des tables, des relations et des requêtes SQL.
- Intégration de l'API Stripe pour la gestion des paiements en ligne et des webhooks de confirmation.
- Utilisation de JWT (JSON Web Token) pour l'authentification sécurisée et la gestion des sessions utilisateurs.
- Mise en place de middlewares Express pour la validation des entrées, la gestion des droits d'accès et la sécurisation des routes sensibles.

•**Outils et pratiques de développement :**

- J'ai utilisé **Visual Studio Code** comme éditeur principal pour écrire et organiser le code du back-end.
- Le code a été versionné avec **Git** et hébergé sur **GitHub**, ce qui m'a permis de sauvegarder chaque étape du projet, de revenir en arrière si besoin, et de travailler proprement.

- J'ai utilisé **Postman** pour tester les différentes routes de l'API : cela m'a permis de vérifier que chaque fonctionnalité (connexion, ajout au panier, paiement, etc.) fonctionnait bien indépendamment du frontend.
- J'ai mis en place des **variables d'environnement** (fichier .env) pour sécuriser les informations sensibles comme les clés d'API ou les identifiants de la base de données.
- J'ai documenté le code avec des commentaires clairs pour expliquer la logique métier, les paramètres attendus et les retours des fonctions.
- J'ai structuré le projet en plusieurs fichiers et dossiers : contrôleurs pour la logique métier, routes pour les points d'entrée de l'API, services pour l'accès aux données, ce qui facilite la maintenance et l'évolution du projet.
- J'ai appliqué les bonnes pratiques de sécurité : validation des données reçues, gestion des erreurs, séparation des droits d'accès selon le rôle de l'utilisateur (admin, client...).

3. Avec qui avez-vous travaillé ?

Pour ce projet j'ai travaillé en total autonomie

4. Contexte

Nom de l'entreprise, organisme ou association *La Plateforme*

Chantier, atelier, service *Boutique en ligne d'instruments et d'accessoires de musique "Soundora"*

Période d'exercice *▶ Du : 17/06/2025 au : 31/08/2025*

5. Informations complémentaires (facultatif)

Voici ci dessous le code du fichier api.js représentant les différentes routes de l'application :

```

routes > JS api.js > ...
1 import express from "express"; // Importe Express pour créer le routeur
2 import * as authController from "../controllers/authController.js"; // Contrôleur d'authentification
3 import * as productSupabaseController from "../controllers/productSupabaseController.js"; // NOUVEAU : Contrôleur produits Supabase (remplace productController)
4 import * as categoryController from "../controllers/categoryController.js"; // Contrôleur catégories
5 import * as brandController from "../controllers/brandController.js"; // Contrôleur marques
6 import * as cartController from "../controllers/cartController.js"; // Contrôleur panier
7 import * as orderController from "../controllers/orderController.js"; // Contrôleur commandes
8 import * as stripeController from "../controllers/stripeController.js"; // NOUVEAU : Contrôleur Stripe pour paiements
9 import * as testController from "../controllers/testController.js"; // Contrôleur de test
10 import checkJwt from "../middleware/checkJwt.js"; // Middleware JWT pour protéger les routes
11 import checkSupabaseAuth from "../middleware/checkSupabaseAuth.js"; // Middleware Supabase Auth
12
13
14 const router = express.Router(); // Crée un routeur Express
15
16 // =====
17 // ROUTES DE TEST
18 // Permettent de vérifier le bon fonctionnement de Supabase
19 // =====
20 router.get("/test/connection", testController.testConnection); // Test connexion Supabase
21 router.get("/test/tables", testController.listTables); // Liste des tables disponibles
22
23 // =====
24 // ROUTES D'AUTHENTIFICATION
25 // Gestion des utilisateurs avec Supabase Auth
26 // =====
27 router.post("/auth/register", authController.register); // Inscription d'un utilisateur
28 router.post("/auth/login", authController.login); // Connexion et obtention du token JWT
29 router.post("/auth/logout", authController.logout); // Déconnexion utilisateur
30 router.get("/auth/me", checkSupabaseAuth, authController.getCurrentUser); // Récupération de l'utilisateur actuel (protégé)
31
32 // =====
33 // ROUTES POUR LES PRODUITS - Version Supabase Avancée
34 // Remplace l'ancien productController par productSupabaseController
35 // Nouvelles fonctionnalités : pagination, filtres, recherche, slugs SEO
36 // =====
37 // ROUTE PRINCIPALE : Liste des produits avec filtres et pagination
38 // Exemples d'utilisation :
39 // GET /api/products - Tous les produits (page 1, 10 par page)
40 // GET /api/products?page=2&limit=20 - Page 2 : 20 produits par page

```

Utilisation de Copilot

Complétions de code

Messages de conversation

Requêtes Premium

Des requêtes Premium payantes supplémentaires

Gérer les requêtes Premium payantes

Réinitialisations de l'allocation 1 septembre 2025.

Workspace Index

Remote Index

Ch

Complétions de code

Tous les fichiers

JavaScript

Suggestions de modification suivantes

Répéter

Masquer les complétions pendant

```

routes > JS api.js > ...
41 // GET /api/products?category=guitares - Seulement les guitares
42 // GET /api/products?brand=fender&min_price=500 - Marque Fender, prix min 500€
43 // GET /api/products?search=stratocaster - Recherche "stratocaster"
44 router.get("/products", productSupabaseController.getAllProducts);
45
46 // ROUTE PRODUITS FEATURED : Produits mis en avant
47 // GET /api/products/featured?limit=6 - 6 produits featured (défaut)
48 // GET /api/products/featured?limit=12 - 12 produits featured
49 router.get("/products/featured", productSupabaseController.getFeaturedProducts);
50
51 // ROUTE RECHERCHE : Recherche textuelle rapide
52 // GET /api/products/search?q=guitare&limit=10 - Recherche "guitare", max 10 résultats
53 // GET /api/products/search?q=gibson - Recherche "gibson"
54 router.get("/products/search", productSupabaseController.searchProducts);
55
56 // ROUTE PRODUIT INDIVIDUEL : Récupération par slug (SEO friendly)
57 // GET /api/products/gibson-les-paul-standard-2024 - Produit via slug
58 // GET /api/products/fender-stratocaster-american - Autre exemple
59 // IMPORTANT : cette route doit être EN DERNIER pour éviter les conflits avec "featured" et "search"
60 router.get("/products/:slug", productSupabaseController.getProductBySlug);
61
62 // =====
63 // ROUTES POUR LES CATÉGORIES
64 // Gestion des catégories de produits (guitares, basses, etc.)
65 // =====
66 router.get("/categories", categoryController.getAllCategories); // Liste toutes les catégories
67 router.get("/categories/:id", categoryController.getCategoryById); // Récupère une catégorie spécifique
68 router.post("/categories", checkJwt, categoryController.createCategory); // Création d'une catégorie (protégé)
69 router.put("/categories/:id", checkJwt, categoryController.updateCategory); // Mise à jour (protégé)
70 router.delete("/categories/:id", checkJwt, categoryController.deleteCategory); // Suppression (protégé)
71
72 // =====
73 // ROUTES POUR LES MARQUES
74 // Gestion des marques de produits (Fender, Gibson, etc.)
75 // =====
76 router.get("/brands", brandController.getAllBrands); // Liste toutes les marques
77 router.get("/brands/:id", brandController.getBrandById); // Récupère une marque par ID
78 router.get("/brands/slug/:slug", brandController.getBrandBySlug); // Récupère une marque par slug
79
80 // =====

```

```

routes > JS api.js > ...
83 // =====
84 router.get("/cart", checkJwt, cartController.getCart); // Récupère le panier de l'utilisateur (protégé)
85 router.post("/cart/items", checkJwt, cartController.addToCart); // Ajoute un article au panier (protégé)
86 router.put("/cart/items/:id", checkJwt, cartController.updateCartItem); // Met à jour un article du panier (protégé)
87 router.delete("/cart/items/:id", checkJwt, cartController.removeFromCart); // Supprime un article du panier (protégé)
88 router.get("/cart/count", checkJwt, cartController.getCartCount); // Récupère le nombre d'articles dans le panier (protégé)
89
90 // =====
91 // ROUTES POUR LES COMMANDES
92 // Gestion des commandes et historique d'achat
93 // =====
94 router.post("/orders", checkJwt, orderController.createOrder); // Création d'une commande (protégé)
95 router.get("/orders", checkJwt, orderController.getUserOrders); // Récupère les commandes de l'utilisateur (protégé)
96 router.get("/orders/:order_id", checkJwt, orderController.getOrderDetails); // Récupère une commande spécifique (protégé)
97
98 // =====
99 // ROUTES STRIPE PAIEMENT
100 // Gestion des paiements via Stripe Checkout
101 // =====
102 router.post("/stripe/create-checkout-session", checkSupabaseAuth, stripeController.createCheckoutSession); // Créer session Stripe (protégé)
103 router.get("/stripe/session-status/:sessionId", stripeController.getSessionStatus); // Vérifier statut session
104
105 // =====
106 // ROUTES STRIPE TEST (POUR DÉVELOPPEMENT) - ACTIVÉES POUR TESTS
107 // Routes simples pour tester Stripe sans authentification
108 // =====
109 router.post("/stripe/test-simple", stripeController.createTestSessionSimple); // Test session 10€ (non protégé)
110 router.post("/stripe/test-complete", stripeController.createTestSessionComplete); // Test session 99€ (non protégé)
111
112 // =====
113 // WEBHOOK STRIPE (SANS AUTHENTIFICATION)
114 // Stripe appelle cette route pour confirmer les paiements
115 // IMPORTANT: Cette route ne doit PAS avoir de middleware d'auth
116 // Le middleware raw est déjà appliqué dans server.js
117 // =====
118 router.post("/stripe/webhook", stripeController.stripeWebhook);
119
120 // === EXPORT DU ROUTEUR POUR L'UTILISER DANS server.js ===
121 export default router;

```

Et voici le fichier .env pour la sécurité des variables d'environnement :

```

1 # Configuration du serveur
2 PORT=3010
3
4 # URL Frontend pour redirections Stripe
5 FRONTEND_URL=http://localhost:4200
6
7 # Configuration Supabase (remplace MySQL)
8 SUPABASE_URL=https://lohumrjasdauvpqgjvhd.supabase.co
9 SUPABASE_ANON_KEY=eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpc3MiOiJzdXBhYmFzZSIsInJlZiI6ImxvaHVtcmphc2RhZDZwcWdqdmhkIiwicm9sZSI6ImFub24iLCJpYXQiOjE3NTAxNTkyMzksImV4cCI6IjA2NTczNTIzOjAwLjE0OTQyODUwMDQyOTk5RzqnXEA-zE9KV1UBsoo
10 SUPABASE_SERVICE_KEY=eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpc3MiOiJzdXBhYmFzZSIsInJlZiI6ImxvaHVtcmphc2RhZDZwcWdqdmhkIiwicm9sZSI6ImNlcjZpY2Vfcm9sZSIsImldCI6MTc1MDU1OTIzOWSiZXhwIjoyMDY1NzM1MjM1fQ.yq2UA71ljX31WdxKQ5hN37MG6lorg0s7XY1FgQ7rqWk
11
12 # JWT Secret pour l'authentification
13 JWT_SECRET=6RUygh01V8CAsAjIEaq7a2ikbaz6aih+lxJuizAw47zW12NgxVPzXonfdk8RyE/4QdcfcwiTHfnKFHbyllg+A==
14
15 # Configuration Stripe (MODE TEST pour projet étudiant)
16 STRIPE_PUBLISHABLE_KEY=pk_test_51RqscFGtJYVPcJeyWCxcZFKEYl8RZNZYkl12wYz4ouZMc7B9V15Gu02KZWoa94L9CYXMsbG6La7N7SGWZTSfziS00MoCcQ1GD
17 STRIPE_SECRET_KEY=sk_test_51RqscFGtJYVPcJeymYuKSJgKfImxRKW8CSzc2ODEDxzsjIbjM7f2wEZkpqBr3vhKhofj576wGQRcu2xy mc0GYygW300E2717oH0
18 STRIPE_WEBHOOK_SECRET=whsec_c1f029953b5043a6a2973fb0264788278ec346fe0c67686838664aec50ccee74
19
20 # URL de connexion PostgreSQL pour Supabase
21 DATABASE_URL=postgres://postgres:Foiel312%402512@db.lohumrjasdauvpqgjvhd.supabase.co:5432/postgres

```

Activité-type 2

Développer la partie back-end d'une application web ou web mobile sécurisée

CP 8 ▶

Documenter le déploiement d'une application dynamique (déploiement en cours avec plesk)

1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :

Tâches et opérations effectuées :

Dans le cadre du projet Soundora, j'ai conçu et développé l'ensemble du back-end de l'application, en mettant l'accent sur la sécurité et la modularité.

J'ai structuré le serveur avec Node.js et Express, créé des routes RESTful pour la gestion des utilisateurs, des produits, des commandes et des paniers, et mis en place des middlewares pour l'authentification JWT et la gestion des droits d'accès.

J'ai conçu le schéma relationnel de la base de données MySQL (MCD/MLD), rédigé les scripts SQL, et utilisé phpMyAdmin pour la gestion et la visualisation des données.

Pour la partie paiement, j'ai intégré Stripe afin de permettre des transactions sécurisées, en gérant la création de sessions de paiement, la réception des webhooks et la validation des paiements côté serveur.

Le déploiement a été réalisé via l'interface Plesk : j'ai transféré les fichiers du projet, configuré l'environnement Node.js, installé les dépendances, paramétré les variables d'environnement (clés Stripe, accès BDD), et relié le nom de domaine fourni par mon organisme de formation (bastien-brunet.students-laplateforme.io) à l'adresse IP dédiée (82.165.185.52).

J'ai également configuré le serveur web (HTTPS, redirections, droits d'accès) et la base de données via Plesk, puis testé l'ensemble du processus de bout en bout.

Conditions de réalisation :

Le projet a été mené dans le cadre d'un dossier professionnel de fin d'année, sur un hébergement mutualisé avec Plesk fourni par l'organisme de formation. J'ai utilisé un environnement Linux, Git pour le versionnement, supabase et phpMyAdmin pour la gestion de la base. Le nom de domaine et l'adresse IP m'ont été attribués par la plateforme, ce qui a facilité la mise en ligne et la configuration DNS.

J'ai veillé à respecter les bonnes pratiques de sécurité (gestion des secrets, HTTPS, droits d'accès restreints).

2. Précisez les moyens utilisés :

Voici les moyens que j'ai utilisés :

Environnement de développement :

Utilisation de Visual Studio Code pour l'édition du code, avec des extensions facilitant le développement Node.js, la gestion Git et la coloration syntaxique SQL.

Node.js et npm ont permis de gérer les dépendances et d'exécuter le serveur localement pour les tests.

•Gestion de versions et collaboration :

Git a été utilisé pour le suivi des modifications, la création de branches et la gestion des versions du projet. GitHub a servi de plateforme de partage, de sauvegarde et de collaboration, permettant également de documenter le projet via le README.

•Base de données :

MySQL Workbench a servi à la modélisation du schéma relationnel (MCD/MLD) et à la génération des scripts SQL.

phpMyAdmin, accessible via Plesk, a permis la gestion, la visualisation et la modification des données en production.

•Déploiement et hébergement :

Plesk a été utilisé pour le déploiement du projet : transfert des fichiers, configuration de l'environnement Node.js, gestion des variables d'environnement, installation des dépendances, gestion du nom de domaine (bastien-brunet.students-laplateforme.io) et de l'adresse IP (82.165.185.52), configuration du certificat SSL pour le HTTPS, et gestion des accès à la base de données.

•Services externes :

Stripe a été intégré pour la gestion des paiements en ligne, avec utilisation de l'API, gestion des clés secrètes en variables d'environnement, et configuration des webhooks pour le suivi des transactions.

•**Documentation et organisation :**

Rédaction de la documentation technique (README, scripts SQL, schémas de base de données, procédures de déploiement) pour assurer la maintenabilité et la reproductibilité du projet.

3. Avec qui avez-vous travaillé ?

Travail réalisé en autonomie.

4. Contexte

Nom de l'entreprise, organisme ou association *La Plateforme*

Chantier, atelier, service *Boutique en ligne d'instruments et d'accessoires de musique "Soundora"*

Période d'exercice ▶ Du : *17/06/2025* au : *31/08/2025*

5. Informations complémentaires (facultatif)

DOSSIER PROFESSIONNEL (DP)

Titres, diplômes, CQP, attestations de formation

(facultatif)

Intitulé	Autorité ou organisme	Date
Cliquez ici.	Cliquez ici pour taper du texte.	Cliquez ici pour sélectionner une date.

Déclaration sur l'honneur

Je soussigné(e) [prénom et nom] [Cliquez ici pour taper du texte.](#).....,

déclare sur l'honneur que les renseignements fournis dans ce dossier sont exacts et que je suis
l'auteur(e) des réalisations jointes.

Fait à .. [Cliquez ici pour choisir une date](#)..... le .. [Cliquez ici pour choisir une date](#).....

pour faire valoir ce que de droit.

Signature :

bastien brunet

Documents illustrant la pratique professionnelle

(facultatif)

Intitulé

Cliquez ici pour taper du texte.

DOSSIER PROFESSIONNEL (DP)

ANNEXES

(Si le RC le prévoit)