



UNIVERSITÉ DE HAUTE-ALSACE

PROJET FIN D'ANNEE

## **EDX : A Comprehensive Educational Management Platform**

*Hadjarsi Mohamed El Fateh  
Boulaabi Ahmed*

Supervisé par  
Professeur Joel Heinis

Date de soumission : 12-06-2024

## Résumé

---

Ce rapport présente la conception et le développement d'EDX (EDucation eXpert), une plateforme de gestion éducative conçue pour améliorer l'efficacité administrative et l'expérience des élèves, des enseignants et des administrateurs académiques. EDX intègre une variété de fonctionnalités, notamment la gestion des étudiants et des enseignants, la planification des cours, la disposition des sièges et le suivi de présence en temps réel. En utilisant des technologies Web modernes et une architecture de base de données robuste, l'objectif d'EDX est de rationaliser les processus d'administration éducative, en fournissant des interfaces intuitives aux utilisateurs. Le développement de la plateforme a suivi un mélange de méthodologies agiles et RAD (Rapid Application Development), garantissant une amélioration itérative et une réactivité aux retours. Les fonctionnalités clés incluent la création de plans de salle de classe, les utilisateurs, les demandes d'inscription et la gestion des horaires, les sièges et la présence des étudiants, ainsi que la gestion des dépôts Github. Le système a été construit avec des technologies qui garantissent la fiabilité, l'évolutivité et la facilité d'utilisation. Ce rapport détaille la conception, la mise en œuvre, les défis rencontrés et les améliorations futures potentielles de la plateforme EDX.

## Table des matières

---

<b>1 Contexte et Objectifs</b>	<b>5</b>
<b>2 Présentation de la structure du rapport</b>	<b>6</b>
<b>3 Fonctionnalités clés</b>	<b>7</b>
3.1 Gestion des étudiants . . . . .	7
3.2 Gestion des enseignants . . . . .	7
3.3 La disposition des sièges . . . . .	7
3.4 Planification des cours . . . . .	8
3.5 Suivi de présence en temps réel . . . . .	8
<b>4 Architecture du système</b>	<b>8</b>
4.1 Présentation de l'architecture du système . . . . .	8
4.2 Les composants . . . . .	8
4.3 Next.js . . . . .	8
4.4 TypeScript . . . . .	9
4.4.1 Côté client (frontend) . . . . .	9
4.4.2 Côté serveur (Backend) . . . . .	10
4.4.3 Outils de développement et de construction . . . . .	10
4.4.4 CI/CD et déploiement . . . . .	10
<b>5 Conception et planification du projet</b>	<b>12</b>
5.0.1 Aperçu de l'idée initiale du projet . . . . .	12
5.0.2 Discussions et rencontres avec le professeur Joel Heinis . .	12
5.0.3 Objectifs et buts fixés pendant la phase de planification .	12
5.1 Processus de développement . . . . .	13
5.1.1 Description de la méthodologie de développement utilisée	13
5.1.2 Cycles de développement itératifs et Sprints . . . . .	13
5.1.3 Outils et technologies utilisés . . . . .	13
5.1.4 Gestion et atténuation des risques . . . . .	13
5.2 Phase de conception . . . . .	14
5.2.1 Réunions de conception initiales et séances de brainstorming . . . . .	14
5.2.2 Conception et interface (Frontend) . . . . .	14
5.2.3 Commentaires du professeur Joel Heinis et ajustements effectués . . . . .	16
<b>6 Conception du système</b>	<b>17</b>
6.1 Architecture système détaillée . . . . .	17

6.1.1	Conception de schéma de la base de données . . . . .	17
6.2	Interface Utilisateur . . . . .	18
6.2.1	Next.js App-dir . . . . .	19
6.3	Backend . . . . .	21
6.3.1	Principes de conception . . . . .	21
6.3.2	Sélection et intégration des composants . . . . .	22
<b>7</b>	<b>Explication détaillée de la mise en œuvre de la fonctionnalité clés</b>	<b>25</b>
7.1	Gestion des étudiants . . . . .	25
7.2	Gestion des enseignants . . . . .	26
7.3	Disposition des sièges . . . . .	27
7.4	Planification des cours . . . . .	27
7.5	Sécurité et Authentification . . . . .	28
7.6	Optimisation des Performances . . . . .	28
7.7	Intégration et déploiement continus . . . . .	29
<b>8</b>	<b>Tests et Evaluation</b>	<b>30</b>
8.1	Stratégies et méthodologies de test . . . . .	30
8.1.1	Outils automatisés . . . . .	30
8.1.2	Tests de performance . . . . .	30
8.1.3	Tests de construction Next.js et contrôle de qualité . . . . .	32
8.1.4	TypeScript et évaluations par les pairs . . . . .	32
8.1.5	Créer des tests et une assurance qualité . . . . .	32
8.2	Évaluation des performances du système . . . . .	32
8.2.1	Futurs plans de tests . . . . .	32
<b>9</b>	<b>Défis et Solutions</b>	<b>34</b>
9.1	Problèmes rencontrés lors du développement . . . . .	34
9.1.1	Contraintes de temps . . . . .	34
9.1.2	Configuration de Supabase et Drizzle ORM . . . . .	34
9.1.3	Problèmes du fournisseur d'authentification Supabase . . . . .	34
9.1.4	Complexités avec dnd-kit . . . . .	34
9.1.5	Canevas infini pour planificateur de pièce . . . . .	34
9.1.6	Calendrier glisser-déposer . . . . .	35
9.1.7	transmission de propriétés en React . . . . .	35
9.1.8	Animation avec Framer Motion . . . . .	35
9.1.9	Problèmes de déploiement . . . . .	35
9.2	Comment ces défis ont été relevés et résolus . . . . .	35
9.3	Dernières réflexions . . . . .	36

<b>10 Résultats et Discussion</b>	<b>37</b>
10.1 Résumé des résultats obtenus . . . . .	37
10.2 Comparaison avec les objectifs initiaux . . . . .	37
10.3 Discussion sur l'efficacité et l'impact d'EDX . . . . .	38
<b>11 Travaux futurs</b>	<b>39</b>
11.1 Améliorations potentielles et fonctionnalités supplémentaires . .	39
11.2 Vision à long terme de la plateforme . . . . .	39
<b>12 Conclusion</b>	<b>39</b>

### 1 Contexte et Objectifs

Une plateforme de gestion éducative centralisée, efficace et conviviale a toujours été nécessaire. Notre projet, EDX (EDucation eXpert), a été lancé sous la direction du professeur Joel Heinis, qui nous a chargé et guidé tout au long du développement d'un système complet pour répondre aux besoins des établissements d'enseignement.

L'objectif global était de créer une plate-forme Web permettant aux administrateurs, aux enseignants et aux étudiants d'accéder à un ensemble d'outils essentiels, améliorant ainsi l'efficacité opérationnelle et l'expérience utilisateur.

EDX vise à intégrer des fonctionnalités traditionnellement réparties sur divers outils tiers. En regroupant ces fonctionnalités sur une plate-forme unique, EDX a le potentiel de rationaliser les processus administratifs en une seule solution hébergée localement, de réduire la dépendance à l'égard des logiciels externes et de servir de référentiel unique d'outils pédagogiques.

**EDX est simple, cohérent et efficace.**



FIGURE 1 – Logo de EDX

## **2 Présentation de la structure du rapport**

---

Ce rapport est structuré pour fournir un aperçu complet du développement et de la mise en œuvre de la plateforme EDX. Il comprend les sections suivantes :

- **Présentation générale du système** : Une description de haut niveau d'EDX, mettant en évidence ses principales caractéristiques et fonctionnalités.
- **Méthodologie** : Un aperçu du processus de développement et des méthodologies utilisées, y compris les outils et technologies utilisés.
- **Conception système** : Descriptions détaillées de l'architecture du système, de la conception de l'interface utilisateur et des interactions des modules.
- **Implémentation** : Informations sur la mise en œuvre des fonctionnalités clés, y compris les extraits de code.
- **Teste et évaluation** : Une explication des stratégies de test, des cas de test et des résultats, ainsi qu'une évaluation des performances du système.
- **Défis et solutions** : Une discussion sur les problèmes rencontrés au cours du développement et sur la manière dont ils ont été résolus.
- **Résultats et discussion** : Une synthèse des résultats obtenus, par rapport aux objectifs initiaux, et une discussion sur l'efficacité et l'impact d'EDX.
- **Travaux futurs** : Améliorations potentielles et fonctionnalités supplémentaires pour la plateforme.
- **Conclusion** : Un récapitulatif des objectifs et des réalisations du projet, ainsi que des réflexions finales sur l'importance du projet.

À la fin de ce rapport, nous visons à démontrer comment EDX répond non seulement à certains des besoins actuels des administrateurs pédagogiques et des enseignants, mais établit également les bases des avancées futures.

## 3 Fonctionnalités clés

---

### 3.1 Gestion des étudiants

La fonctionnalité de gestion des étudiants d'EDX permet aux administrateurs et aux enseignants de gérer efficacement les informations sur les étudiants.

Ceci comprend :

- **Profils des étudiants** : Création et mise à jour de profils d'étudiants détaillés avec la possibilité pour les étudiants d'ajouter des informations personnelles, académiques et de contact.
- **Inscription** : Gérer les processus d'inscription des étudiants et tenir à jour les dossiers des étudiants inscrits.
- **Assiduité** : Suivi et gestion des dossiers de fréquentation des étudiants en temps réel.

### 3.2 Gestion des enseignants

La fonctionnalité de gestion des enseignants fournit des outils pour gérer les informations et les horaires des enseignants. Les fonctionnalités clés incluent :

- **Profils des enseignants** : Création et maintenance de profils avec des informations personnelles, de contact et professionnelles.
- **Planification** : Gérer les plannings des professeurs et assigner les cours.

### 3.3 La disposition des sièges

La fonction Disposition des sièges aide à organiser les plans de salle de classe. Elle propose :

- **Plans de salle dynamiques** : Créer et ajuster les plans de salle en fonction de la disposition des salles de classe.
- **Attribution de sièges aux étudiants** : Attribuer des sièges aux étudiants manuellement ou automatiquement, ceci est particulièrement utile pour les examens.
- **Plans interactives** : Fournir des plans visuels de la disposition des sièges pour une référence facile. De plus, la possibilité d'imprimer les sièges au format PDF
- **Intégration de base de données** : Stockage des plans de salle dans la base de données pour de futurs références et ajustements.
- **Gestion du dépôt** : Gérer la création et l'accès aux dépôts pour les séances et les étudiants.

### **3.4 Planification des cours**

La planification des cours dans EDX rationalise le processus de planification et d'organisation des horaires de cours. Cette fonctionnalité comprend :

- **Création d'horaires** : Possibilité de créer des horaires et des plannings pour les enseignants.
- **Attribution des salles de classe** : Affectation des cours aux salles de classe.

### **3.5 Suivi de présence en temps réel**

Le suivi de présence en temps réel garantit une surveillance précise de la présence des étudiants. Les fonctionnalités incluent :

- **Gestion de présence des étudiants** : Enregistrement des présences en temps réel pendant les cours.

## **4 Architecture du système**

### **4.1 Présentation de l'architecture du système**

Nous avons choisi de mettre en œuvre une architecture modulaire pour garantir l'évolutivité, la maintenabilité et la facilité d'intégration.

### **4.2 Les composants**

Les principaux composants du système EDX et leurs interactions sont :

### **4.3 Next.js**

Next.js est la base d'EDX. Bien qu'il soit considéré comme un framework Backend, il permet de contrôler à la fois le backend et le frontend, il s'appuie sur React.js, un framework Javascript pour créer des composants d'interface utilisateur dynamiques et réutilisables, qui contrairement à PHP et J2EE qui restituent le HTML sur le serveur et envoyez-le au client, React.js gère tout cela côté client, rendant l'interface utilisateur plus interactive, réactive et dynamique. Next.js améliore encore le DX (Developer Experience) et l'UX (User Experience) de React.js en fournissant une couche backend permettant de construire l'application complète sur une seule base de code. De plus, une pléthore d'optimisations qui ressemblent beaucoup à la façon dont PHP et les serveurs gèrent le service aux utilisateurs, comme l'ISR (Incremental Static Regeneration) qui restitue les parties statiques du site Web sur le serveur avant de les servir à l'utilisateur (Server-Side Rendering), et permet aux parties dynamiques du site de s'exécuter sur le client pour une réactivité et une réactivité en temps réel (Client-Side Rendering).

Next.js allie le meilleur des deux mondes du rendu côté serveur traditionnel et des frameworks modernes côté client, offrant les avantages en termes de performances et de référencement du rendu côté serveur avec l'expérience utilisateur dynamique et l'architecture basée sur les composants du côté client.

#### 4.4 TypeScript

TypeScript est un sur-ensemble de JavaScript fortement typé qui améliore le processus de développement en ajoutant des types statiques. Cela permet de détecter les erreurs au moment de la compilation, plutôt qu'au moment de l'exécution. Typescript garantit un code robuste et maintenable, permettant aux développeurs de créer facilement des applications sécurisées. En intégrant TypeScript dans EDX, nous garantissons une meilleure qualité de code, une productivité améliorée des développeurs et une pérennité.



FIGURE 2 – Logos de Next.js et TypeScript

##### 4.4.1 Côté client (frontend)

Le côté client d'EDX est construit à l'aide de technologies Web modernes pour fournir une interface utilisateur réactive et intuitive. Il comprend :

- **Tailwind CSS** : Un framework CSS axé sur l'utilitaire pour styliser l'application.
- **Framer Motion** : Utilisé pour créer des animations et des transitions dans l'interface utilisateur.
- **D3.js** : Une bibliothèque JavaScript pour produire des visualisations de données dynamiques et interactives dans les navigateurs Web.
- **Zod** : Une bibliothèque de déclaration et de validation de schéma typeScript pour valider les entrées utilisateur et garantir l'intégrité des données.
- **Shadcn UI** : Une bibliothèque de composants qui fournit un ensemble de composants d'interface utilisateur accessibles, personnalisables et composable.
- **Radix UI** : Un ensemble de composants accessibles de bas niveau pour créer des applications Web accessibles et de haute qualité.

#### 4.4.2 Côté serveur (Backend)

Le backend d'EDX gère la logique métier et le traitement des données. Cela est composé de :

- **Supabase** : Un backend en tant que service fournissant des solutions d'authentification, de base de données et de stockage via des microservices.
- **PostgreSQL** : Un système de gestion de base de données relationnelle robuste pour stocker et gérer les données.
- **Drizzle ORM** : Un outil de mappage objet-relationnel (ORM) qui simplifie les interactions avec les bases de données de manière sécurisée.
- **Jotai** : Une bibliothèque de gestion d'état pour les applications React, utilisée pour gérer l'état global de manière simple et évolutive.
- **ESLint** : Un outil d'analyse de code statique pour identifier et résoudre les problèmes dans le code JavaScript et TypeScript.
- **Vercel** : Une plateforme cloud pour les sites statiques et les fonctions sans serveur qui simplifie le déploiement et fournit un hébergement rapide et fiable.

#### 4.4.3 Outils de développement et de construction

Un ensemble d'outils de développement et de construction sont utilisés pour rationaliser le processus de développement et garantir la qualité du code :

- **pnpm** : Un gestionnaire de packages rapide et économique en espace disque pour JavaScript et TypeScript, utilisé pour gérer les dépendances du projet.
- **Node.js** : Un environnement d'exécution JavaScript utilisé pour créer des applications côté serveur évolutives et exécuter des scripts de développement.
- **Bun** : Un environnement d'exécution et un gestionnaire de packages JavaScript tout-en-un qui offre une exécution rapide, des outils complets et une prise en charge intégrée de nombreuses fonctionnalités JavaScript modernes.

#### 4.4.4 CI/CD et déploiement

Les processus d'intégration continue et de déploiement continu (CI/CD) sont essentiels pour maintenir un flux de développement fluide. Les outils et scripts clés incluent :

- **Drizzle-Kit** : Pour les tâches de gestion des schémas de base de données et de migration.

- **Supabase CLI** : Pour la gestion du projet et les déploiements Supabase.
- **Custom Scripts** : Des scripts tels que `changelog`, `lint`, `push`, `pull`, `generate`, `drop`, `check`, and `up` pour automatiser diverses tâches de développement.



FIGURE 3 – Tailwind, Supabase, Drizzle-ORM, Framer-Motion, Jotai, Pnpm

## 5 Conception et planification du projet

La méthodologie derrière le développement d'EDX a commencé avec la phase de conception et de planification, qui comprenait :

### 5.0.1 Aperçu de l'idée initiale du projet

L'idée initiale d'EDX était de créer une plateforme complète de gestion éducative qui centraliserait diverses fonctionnalités nécessaires aux administrateurs, aux enseignants et aux étudiants. L'objectif était de rationaliser les tâches administratives et d'améliorer l'expérience éducative globale.

### 5.0.2 Discussions et rencontres avec le professeur Joel Heinis

Tout au long de la phase de planification, des discussions et des réunions régulières ont eu lieu avec le professeur Joel Heinis. Ces réunions ont été cruciales pour affiner l'idée et la mise en œuvre du projet, fixer des objectifs réalistes et garantir que la plateforme atteindra ses objectifs. Le professeur Heinis a fourni des informations et des conseils précieux qui ont façonné le développement d'EDX.

Le professeur Heinis nous a guidé tout au long des premières étapes de développement en rédigeant un document détaillé détaillant toutes les fonctionnalités et flux de travail nécessaires aux utilisateurs. De plus, il a fourni le schéma de base de diverses tables de base de données, qui ont servi de base à nos modèles de données et à nos interactions.

### 5.0.3 Objectifs et buts fixés pendant la phase de planification

Au cours de la phase de planification, plusieurs objectifs et buts clés ont été établis :

- Développement d'une interface conviviale pour les administrateurs, les enseignants et les étudiants.
- L'intégration des fonctionnalités de base telles que la gestion des étudiants et des enseignants, la disposition des sièges, le suivi de la présence en temps réel et la gestion des examens avec la création dynamique des dépôts Github pour les étudiants.
- Garantir que la plate-forme est évolutive, sécurisée et capable de gérer un grand nombre d'utilisateurs et de données.
- L'utilisation des technologies Web modernes et des meilleures pratiques pour améliorer les performances et la réactivité de la plateforme.

## **5.1 Processus de développement**

### **5.1.1 Description de la méthodologie de développement utilisée**

Le développement d'EDX a suivi une combinaison de méthodologies Agile et Rapid Application Development (RAD). Cette approche a permis un développement itératif, un feedback continu et un prototypage rapide.

### **5.1.2 Cycles de développement itératifs et Sprints**

Le processus de développement a été organisé en cycles itératifs, chacun durant une semaine. Chaque cycle, ou sprint, se concentrerait sur des fonctionnalités ou améliorations spécifiques. À la fin de chaque sprint, nous avons examiné les progrès, recueilli les commentaires du professeur Heinis et planifié les prochaines étapes. Ce processus itératif a permis de garantir que le projet restait sur la bonne voie et pouvait s'adapter à tout changement ou nouvelle exigence.

### **5.1.3 Outils et technologies utilisés**

Une variété d'outils et de technologies ont été utilisés tout au long du processus de développement pour rationaliser les flux de travail et garantir la qualité du code :

- **Contrôle de version** : Git et GitHub pour contrôler les versions et la collaboration.
- **Gestion de projet** : Les "issues" de Github pour le suivi des tâches et des progrès.
- **Outils de développement** : Visual Studio Code comme éditeur de code principal, avec des extensions pour TypeScript, ESLint, etc.
- **Communication** : On a utiliser Discord pour la communication et la coordination d'équipe.
- **CI/CD** : GitHub et Vercel pour une intégration et un déploiement continu, garantissant que les modifications de code sont automatiquement testées et déployées.

### **5.1.4 Gestion et atténuation des risques**

Au cours du processus de développement, nous avons identifié plusieurs risques qui pourraient potentiellement avoir un impact sur le projet. Ceux-ci comprenaient des défis techniques, des contraintes de calendrier et des changements potentiels dans les exigences. Pour atténuer ces risques, nous avons adopté une approche proactive en planifiant des réunions régulières, en maintenant une communication claire et en utilisant des outils de gestion de projet pour suivre les progrès et résoudre les problèmes rapidement.

## 5.2 Phase de conception

### 5.2.1 Réunions de conception initiales et séances de brainstorming

La phase de conception a commencé par la prise en compte des exigences et des schémas de base, ainsi que par un brainstorming sur les défis et les flux de travail afin d'écrire au mieux un schéma de base de données robuste dès le départ. Au cours de ces sessions, nous avons discuté de la vision globale de la plateforme EDX, en identifiant les fonctionnalités clés, les interactions des utilisateurs et les technologies. De nombreuses idées ont été proposées et évaluées afin de déterminer la meilleure approche pour créer un système intuitif et efficace.

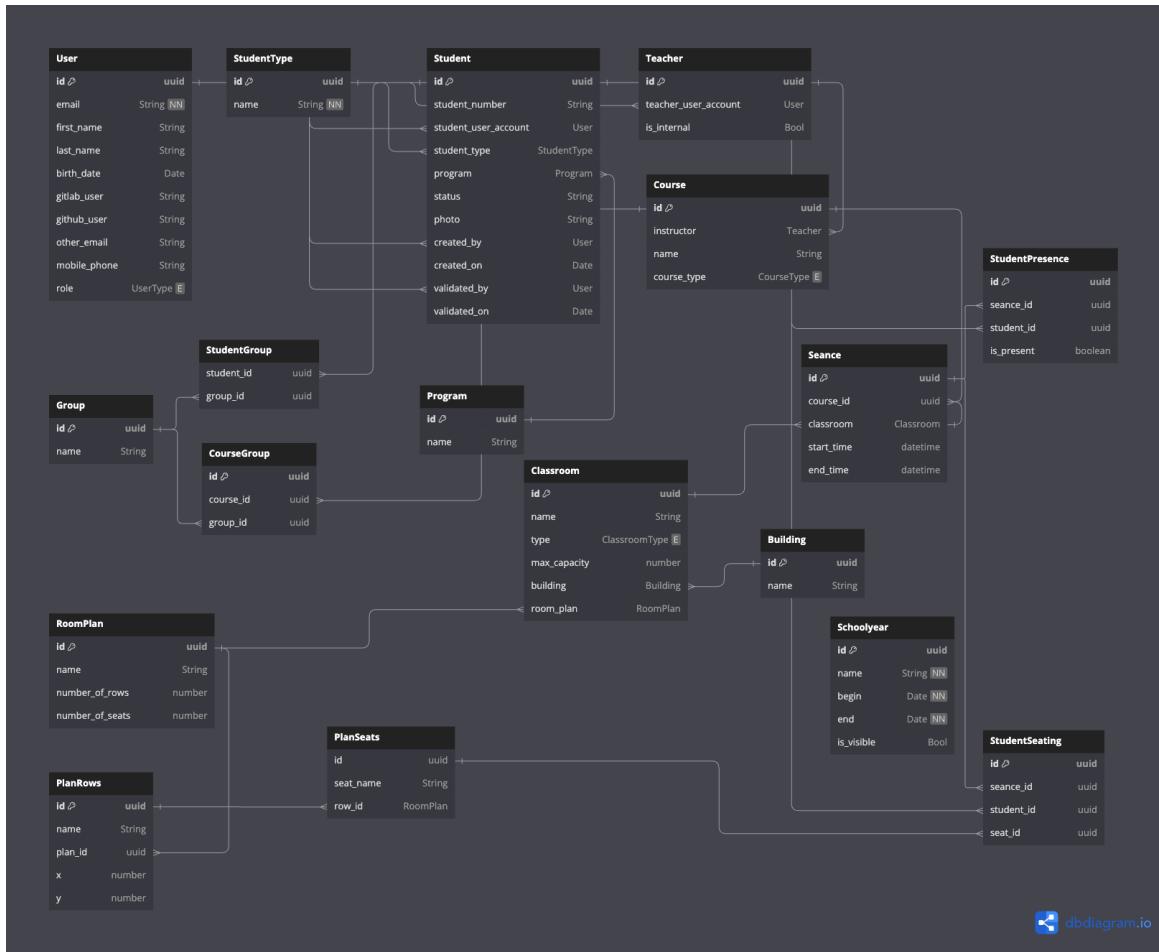


FIGURE 4 – Schéma de la base de données

### 5.2.2 Conception et interface (Frontend)

Suite aux séances de brainstorming, nous avons procédé à la création d'un design d'interface utilisateur avec figma pour les pages principales, ainsi que du design graphique pour la marque EDX. Pour gagner du temps et travailler plus rapidement, nous n'avons pas créé de design pour chaque page, mais nous avons créé un système de conception qui permettrait à chaque page d'avoir une apparence cohérente avec le reste de l'application Web.

Nous avons choisi de ne faire aucun wireframing et avons décidé de créer l'interface utilisateur directement, ce qui nous fait gagner du temps et définit les contraintes et les directives pour le backend.

Des composants réutilisables et dynamiques ont été créés pour améliorer encore l'efficacité de notre développement. Nous avons également utilisé la bibliothèque d'interface utilisateur de shadcn pour prototyper rapidement des mises en page propres et réactives.

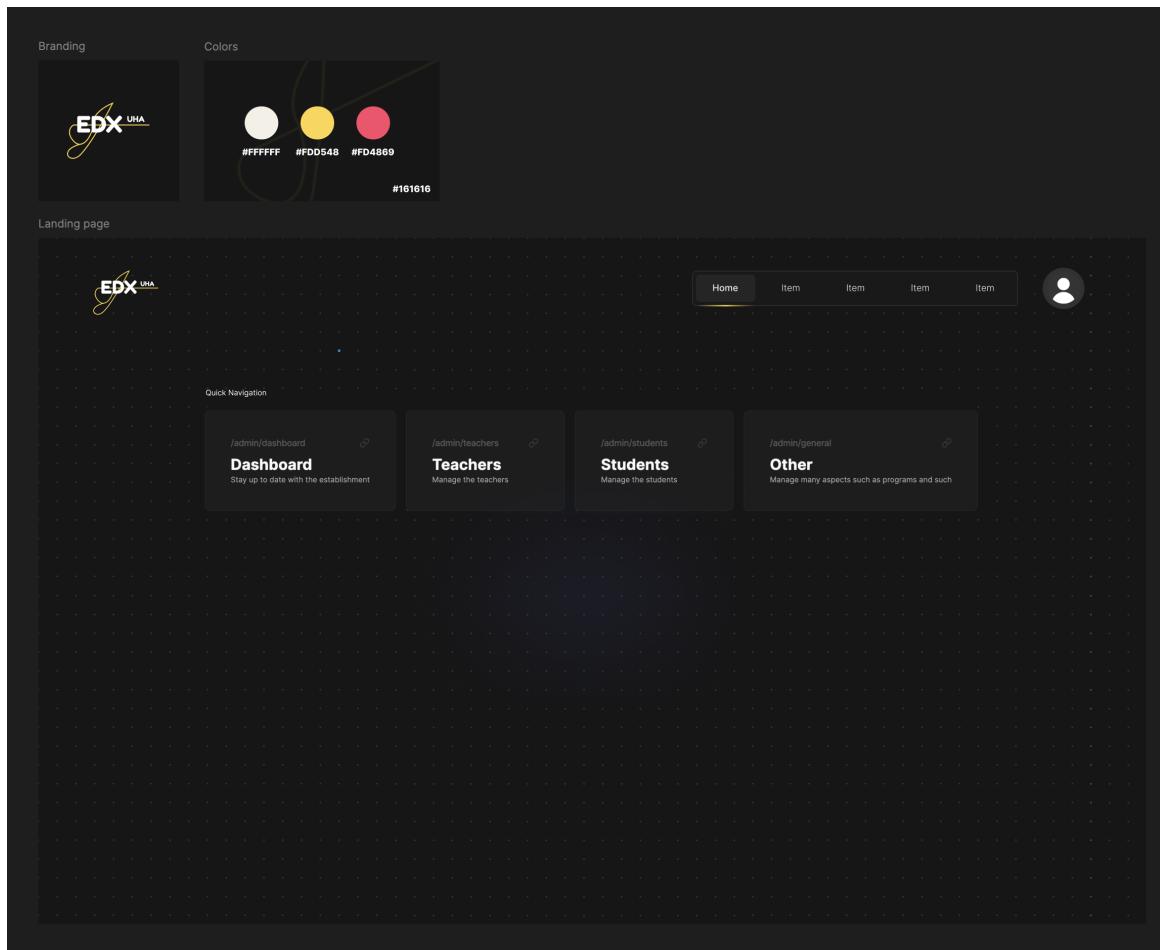


FIGURE 5 – Conception initiale de la page d'accueil et de l'image de marque

The screenshot shows a dark-themed dashboard with the following sections:

- Programs**: Manage Programs Here. Contains four items: IM M1, IM M2, MIAGE M1, and MIAGE M2. Each item has "Modify" and "Delete" buttons.
- Courses**: Manage Courses Here. Contains two items: Architecture N-Tier (Instructor: John Doe) and Development Web (Instructor: John Doe). Each item has "Modify" and "Delete" buttons.
- Student Types**: Manage StudentTypes Here. Contains three items: FA, FI, and FO. Each item has "Modify" and "Delete" buttons.
- School Years**: Manage School Years Here. Contains three items: 2023-2024, 2022-2023, and 2021-2022. Each item has "Modify" and "Delete" buttons.

FIGURE 6 – Conception de la page du tableau de bord d'administration

The screenshot shows a "Planner" page with the following interface elements:

- Header**: Includes the EDX logo, navigation links (Home, Login, Sign Up), and a user icon.
- Main Area**: A grid titled "Row A" containing three icons labeled FST-01, FST-02, and FST-03.
- Right Sidebar**:
  - Parameters**: Plan's Name (R-TP-100), Number of rows (32).
  - Blocks**: Two icons representing different block types.

FIGURE 7 – Conception de la page de planification

### 5.2.3 Commentaires du professeur Joel Heinis et ajustements effectués

Tout au long du projet, le professeur Joel Heinis a fourni des commentaires et des conseils continus. Il a passé en revue nos progrès, offrant des critiques constructives et des suggestions d'amélioration. Ce processus itératif de conception et de retour d'information a permis de garantir que le produit final

était à la fois fonctionnel et convivial.

## 6 Conception du système

### 6.1 Architecture système détaillée

L'architecture système d'EDX est conçue pour garantir l'évolutivité, la fiabilité et la facilité de maintenance. L'architecture comprend plusieurs couches et composants, chacun responsable d'aspects spécifiques du système.

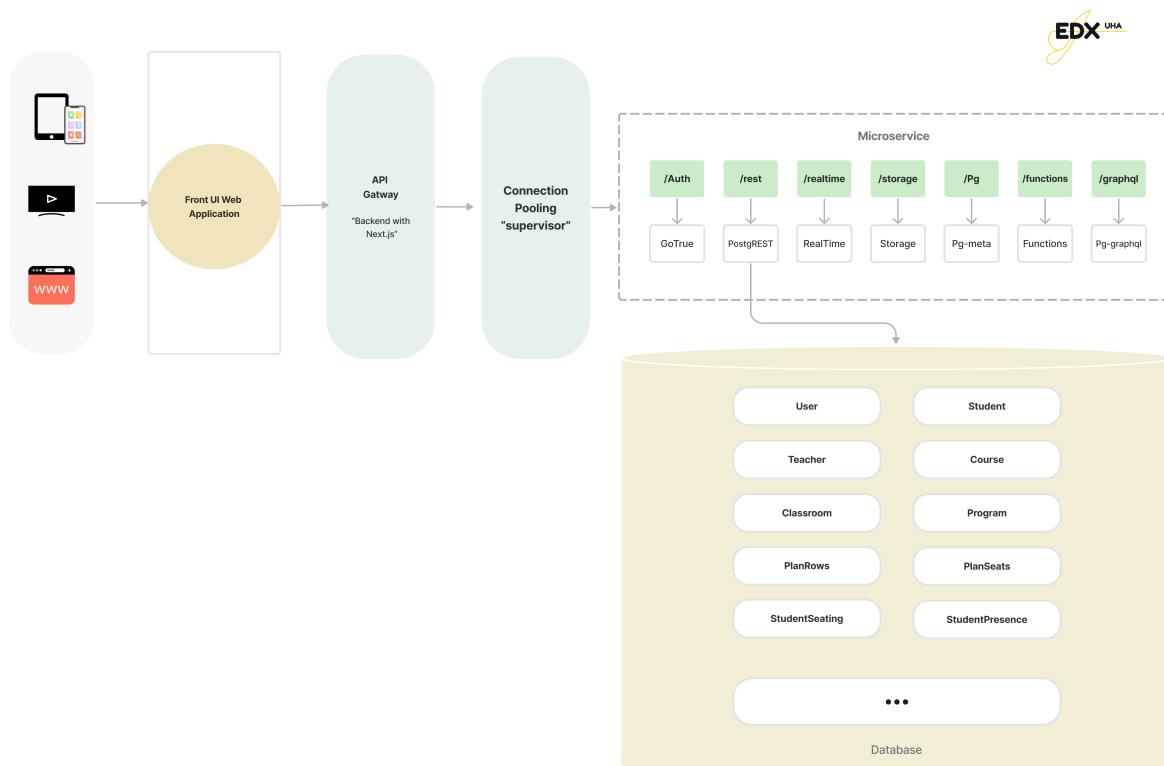


FIGURE 8 – Architecture système détaillée d'EDX

#### 6.1.1 Conception de schéma de la base de données

Le schéma de base de données, défini à l'aide de Drizzle ORM, garantit l'intégrité des données et des requêtes efficaces avec sécurité de type. Il comprend des tableaux pour les utilisateurs, les cours, les relevés de présence et la disposition des sièges, chacun avec des relations et des contraintes appropriées pour maintenir la cohérence.

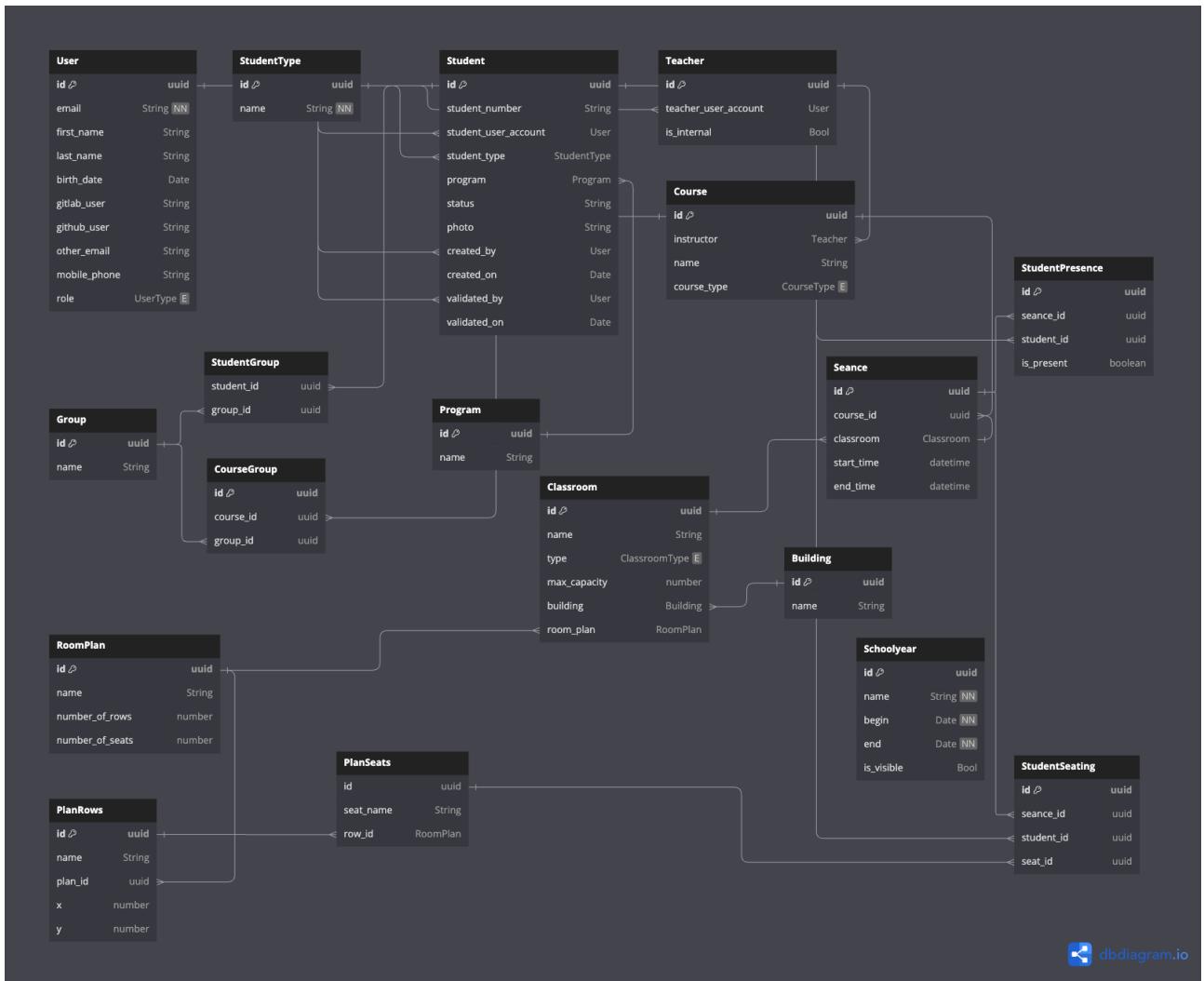


FIGURE 9 – Schéma de la base de données

```

export const users = pgTable("users", {
  id: uuid("id").defaultRandom().primaryKey().notNull(),
  email: text("email").notNull().unique(),
  firstName: text("first_name"),
  lastName: text("last_name"),
  birthDate: date("birth_date"),
  gitlabUser: text("gitlab_user").unique(),
  githubUser: text("github_user").unique(),
  otherEmail: text("other_email"),
  mobilePhone: text("mobile_phone"),
  role: UserType("role"),
});

export const teachers = pgTable("teachers", {
  id: uuid("id").defaultRandom().primaryKey().notNull(),
  teacherUserAccount: uid("teacher_user_account").references(() => users.id),
  { onDelete: "cascade" },
  { isInternal: boolean("is_internal") },
});

```

```

export const studentTypes = pgTable("student_types", {
  id: uid("id").defaultRandom().primaryKey().notNull(),
  name: text("name").notNull().unique(),
});

export const students = pgTable("students", {
  id: uid("id").defaultRandom().primaryKey().notNull(),
  studentNumber: text("student_number").unique(),
  studentUserAccount: uid("student_user_account").references(() => users.id),
  { onDelete: "cascade" },
  studentType: uid("student_type").references(() => studentTypes.id),
  { onDelete: "cascade" },
  status: text("status"),
  photo: text("photo"),
  createdBy: uid("created_by").references(() => users.id),
  { onDelete: "cascade" },
  createdOn: date("created_on"),
  validatedBy: uid("validated_by").references(() => users.id),
  { onDelete: "cascade" },
  validatedOn: date("validated_on"),
});

```

FIGURE 10 – Exemples de définitions de schéma écrites en typescript

## 6.2 Interface Utilisateur

L'interface utilisateur vise à offrir une expérience intuitive et réactive à tous les utilisateurs.

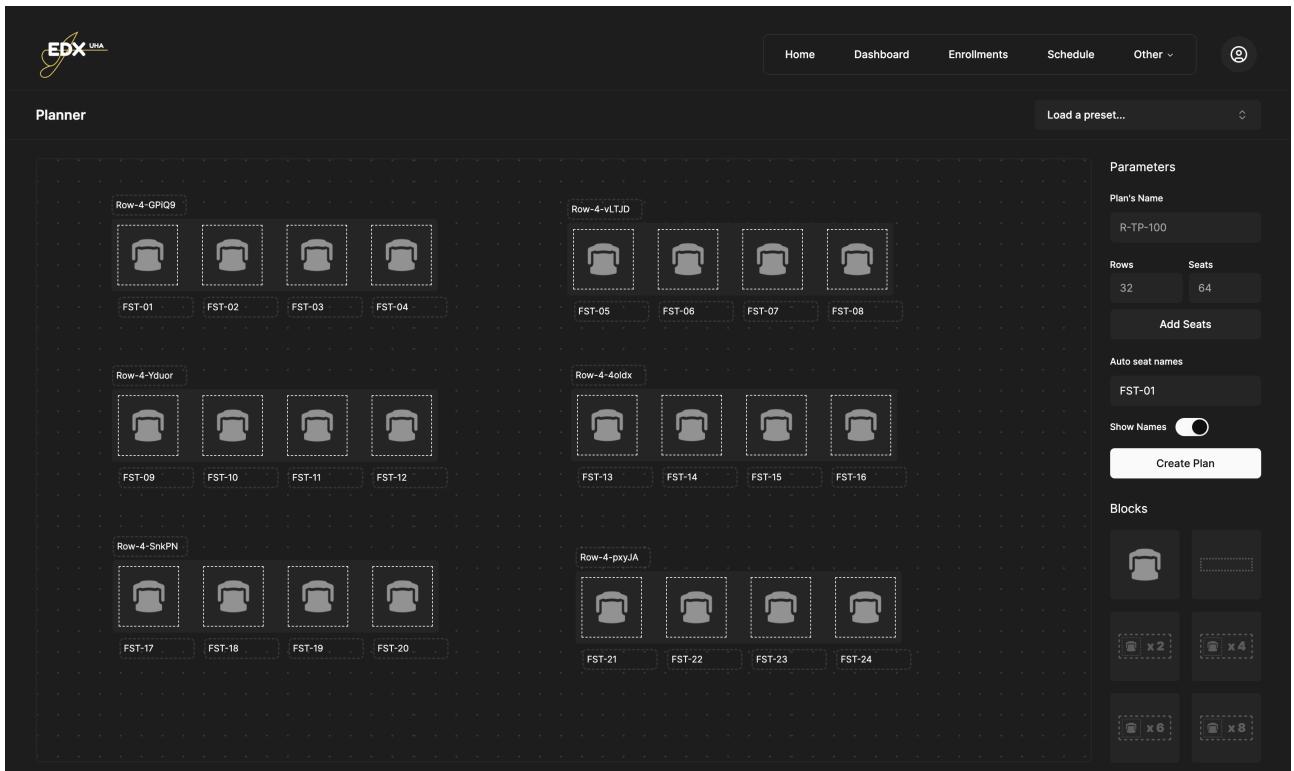


FIGURE 11 – Exemple de page terminée de l’interface utilisateur EDX

Les principes clés de l’interface utilisateur comprennent :

- **Cohérence** : Assurer une apparence cohérente sur toutes les pages et tous les composants.
- **Intuitivité** : Mettre l’accent sur la facilité d’utilisation et l’intuitivité sur toutes les pages.
- **Qualité de vie** : L’importance de rendre l’expérience utilisateur la meilleure possible.
- **Efficacité** : l’interface s’adapte parfaitement aux différentes tailles d’écran et aux différents appareils.

#### 6.2.1 Next.js App-dir

Le répertoire ‘app’ est une nouvelle introduction à Next.js permettant de gérer les routes, les composants et les gestionnaires d’API. Ce qui suit décrit les éléments clés de la structure de répertoires ‘app’, comme indiqué dans la structure de fichiers fournie :

**Répertoires de niveau supérieur** le répertoire ‘src/app’ est structuré pour inclure divers répertoires de niveau supérieur correspondant aux différentes fonctionnalités de l’EDX plate-forme :

```

migrations
└── meta
public
└── images
    └── icons
src
└── app
    └── (room-planner)
        ├── seance
        │   └── [id]
        │       └── planner
        ├── signup
        ├── student
        │   └── profil
        │       └── dashboard
        ├── admin
        │   └── general
        │       └── dashboard
        │   └── classrooms
        └── forgot-password
    └── api
        └── auth
            └── callback
    └── reset-password
    └── timeline
    └── teacher
        └── dashboard
    └── login
components
└── ui
    └── molecules
        ├── Planner
        │   └── state
        │   └── utilities
        │   └── building-blocks
        ├── Student
        ├── Admin
        └── Timeline
            └── state
        └── Teacher
        └── Header
            └── sub-components
atoms
lib
└── server-action
└── providers
└── utils
└── supabase
    └── planner
        └── other
    └── teachers
    └── students
    └── users
    └── timeline

```

FIGURE 12 – Structure du fichier de projet

- **room-planner** : Contient des sous-répertoires pour gérer les fonctionnalités de planification de salle, y compris ‘seance’ avec des itinéraires dynamiques comme ‘[id]’ et un composant ‘Planner’.
- **signup** : Gère les fonctionnalités d’inscription des utilisateurs.
- **student** : Comprend les sous-répertoires ‘profil’ et ‘dashboard’ pour gérer les profils et les tableaux de bord des étudiants.
- **admin** : Contient les sous-répertoires ‘general’, ‘dashboard’, et ‘classrooms’ pour les fonctionnalités d’administrateur.
- **forgot-password** : Gère les processus de récupération de mot de passe.
- **api** : Contient des gestionnaires de routes API, y compris un répertoire ‘auth’ avec un gestionnaire ‘callback’
- **reset-password** : Gère les fonctionnalités de réinitialisation du mot de passe.
- **timeline** : Contient des fonctionnalités de gestion de chronologie.
- **teacher** : Inclut les fonctionnalités de ‘dashboard’ et de ‘login’ de l’enseignant.

**Répertoire des composants** Le répertoire ‘components’ sous ‘src’ est organisé pour améliorer la réutilisabilité et la maintenabilité des composants de l’interface

utilisateur :

- **ui** : Contient des éléments d’interface utilisateur courants.
- **molecules** : Comprend des composants composites tels que ‘Planner’, ‘Student’, ‘Admin’, ‘Timeline’, ‘Teacher’, and ‘Header’, chacun ayant son propre état, ses propres utilitaires et ses propres blocs de construction.

**Lib Directory** Le répertoire ‘lib’ contient des utilitaires et actions qui prennent en charge les principales fonctionnalités de l’application :

- **server-action** : Contient des actions côté serveur.
- **providers** : Comprend les fournisseurs de divers services.
- **utils** : Fonctions d’utilité générale.
- **supabase** : Il s’agit de l’élément essentiel de l’intégration backend, notamment :
  - **schema.ts** : Définit le schéma de la base de données à l’aide de Drizzle ORM, garantissant des interactions sécurisées avec la base de données.
  - **hooks** : Hooks personnalisés pour communiquer avec le backend Supabase.
  - **queries and mutations** : Organisé en sous-répertoires tels que ‘planner’, ‘other’, ‘teachers’, ‘students’, ‘users’, and ‘timeline’. Ces fichiers contiennent des requêtes et des mutations écrites avec Drizzle ORM pour interagir avec Supabase.

**Public Directory** Le répertoire ‘public’ comprend des actifs statiques tels que des images et des icônes, organisés sous les sous-répertoires ‘images’ and ‘icons’.

### 6.3 Backend

Le backend d’EDX est conçu pour fournir une infrastructure robuste, évolutive et de type sécurisé qui prend en charge les différentes fonctionnalités de la plate-forme et garantit une expérience utilisateur transparente. Les principes de conception et le choix des composants visent à obtenir des performances élevées, une facilité de développement et une pérennité du système.

#### 6.3.1 Principes de conception

La conception du système backend est guidée par les principes suivants :

- **Evolutivité** : S’assurer que le système peut gérer des quantités de travail croissantes et s’adapter à la croissance.
- **Sécurité** : Tirer parti des technologies de type sécurisé pour réduire les erreurs d’exécution et améliorer la qualité du code.
- **Performance** : Utiliser des outils et des technologies efficaces pour offrir une expérience réactive et rapide.

- **Facilité de développement** : Choisir des composants qui rationalisent le processus de développement et améliorent la productivité.

### 6.3.2 Sélection et intégration des composants

Le choix des composants backend est crucial pour atteindre les principes de conception souhaités. Voici un aperçu des raisons pour lesquelles des composants spécifiques ont été sélectionnés et de la manière dont ils fonctionnent ensemble :

**Supabase, PostgreSQL et Drizzle ORM** Nous avons choisi Supabase car il fournit un ensemble incroyable d'outils tels que l'authentification, des bases de données en temps réel, des fonctions sans serveur et le stockage. Il simplifie le développement backend en proposant des micro-services prédéfinis qui s'intègrent parfaitement au reste du système.

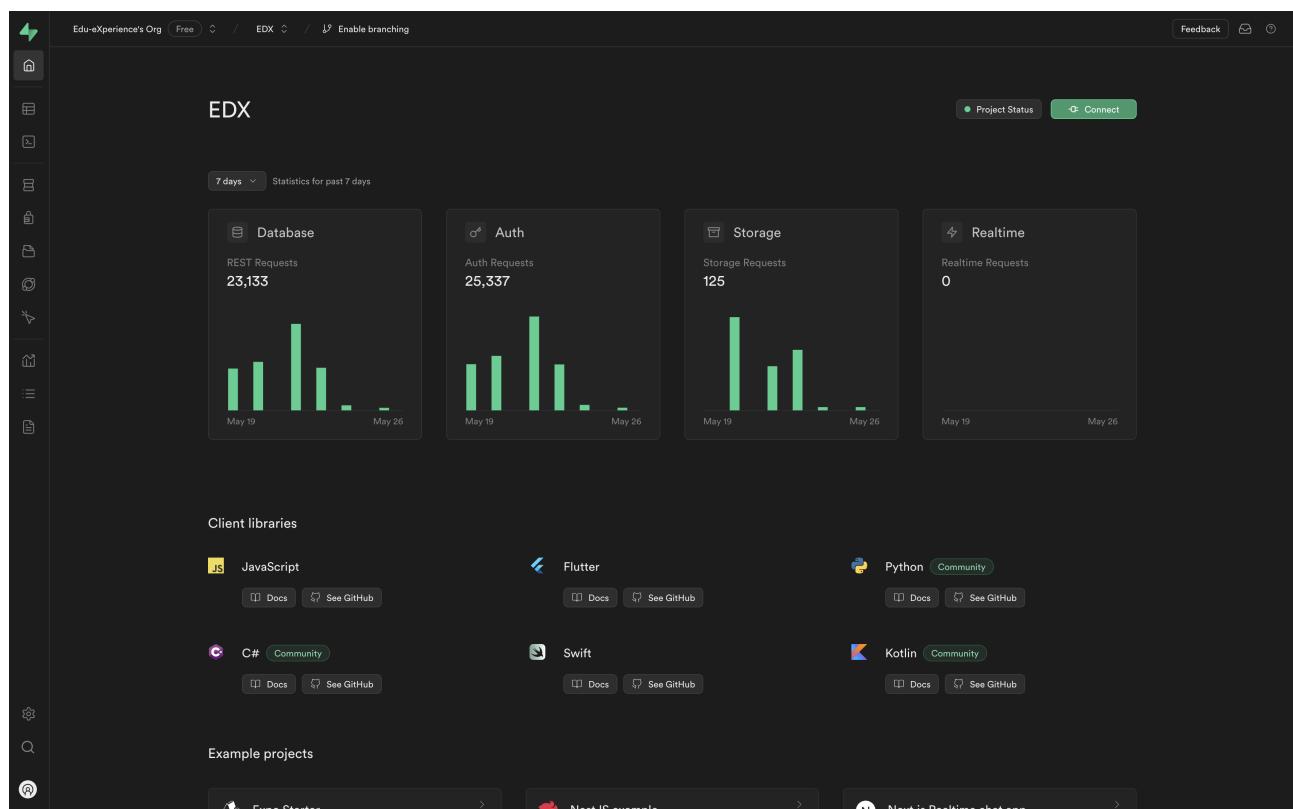


FIGURE 13 – tableau de bord de la Supabase

The screenshot shows the Supabase Database Tables page. On the left, there's a sidebar with navigation links for Database Management (Tables, Functions, Triggers, etc.), Access Control (Roles, Policies), Platform (Backups, Wrappers, Migrations, Webhooks), and Tools (Schema Visualizer, Query Performance, Security Advisor, Performance Advisor). A note in the sidebar says: "Replication section has been renamed. It can be now found under Publications". The main area displays a table of tables with columns: Name, Description, Rows (Estimated), Size (Estimated), Realtime Enabled, and a more options menu. The tables listed are: academic\_backgrounds, buildings, classrooms, course\_programs, courses, groups, internships, plan\_rows, plan\_seats, programs, room\_plans, schoolyears, seance\_groups, seances, skills\_language, and student\_groups.

FIGURE 14 – Tables de Supabase

The screenshot shows the Supabase Table Editor for the 'users' table. The table has columns: id (uuid), email (text), first\_name (text), last\_name (text), birth\_date (date), and gitlab\_id (text). The data includes rows for Admin, Alice, John, Peter, Dorothy, Sherlock, Abe, Huckleberry, Tom, Jane, Oliver, Winnie, Harry, Ahmed, Bill, and Doe. There are also some rows with placeholder values like 'teacher@gmail.com' and 'john.hill@gmail.com'. The table editor interface includes a sidebar with a tree view of all tables in the database, a top bar with schema dropdown, filter, sort, and insert buttons, and a bottom bar with page navigation, refresh, and definition buttons.

FIGURE 15 – Table des utilisateurs sur Supabase

PostgreSQL est robuste et évolutif tout en étant un système de base de données relationnelle.

Nous avons utilisé Drizzle-orm pour garantir la sécurité des types de bout en bout, et cela nous permet également de créer des requêtes complexes en typescript, il construit des instructions préparées pour interagir avec la base de données, ce qui améliore encore la latence de notre backend.



```

export async function createPlan(
  plan_name: string,
  cards: Card[],
  rowData: RowDataType
) {
  const db = await sql.connect();
  db.use('student_seating');

  function countSeatsInRowsData(rowData: RowDataType): number {
    let totalSeats = 0;
    for (let key in rowData) {
      if (rowData.hasOwnProperty(key)) {
        totalSeats += rowData[key].cards.length;
      }
    }
    return totalSeats;
  }

  const cleanCards = cards.filter((c) => c.type === "row");

  // Mutation Prep
  const prepRoomPlans = db
    .insert("roomPlans")
    .values([
      {
        name: sql.placeholder("plan_name"),
        numberOfRows: sql.placeholder("number_of_rows"),
        numberofSeats: sql.placeholder("number_of_seats"),
      }
    ])
    .returning()
    .prepare("create-room-plan");

  const prepPlanRows = db
    .insert("planRows")
    .values([
      {
        name: sql.placeholder("row_name"),
        planId: sql.placeholder("plan_id"),
        x: sql.placeholder("x"),
        y: sql.placeholder("y"),
      }
    ])
    .returning()
    .prepare("add-plan-row");

  const prepSeats = db
    .insert("planSeats")
    .values([
      {
        rowId: sql.placeholder("row_id"),
        seatName: sql.placeholder("seat_name"),
      }
    ])
    .returning()
    .prepare("add-plan-seat");

  // Execution
  const roomPlan = await prepRoomPlans.execute({
    plan_name: plan_name,
    number_of_rows: cleanCards.length,
    number_of_seats: countSeatsInRowsData(rowData),
  });

  for (let key in rowData) {
    if (rowData.hasOwnProperty(key)) {
      let R = rowData[key];
      let RCard = cards.find((c) => c.id === R.id);
      let row = await prepPlanRows.execute({
        row_name: R.name.trim() === "" ? R.name : R.id,
        plan_id: roomPlan[0].id,
        x: RCard.coordinates.x,
        y: RCard.coordinates.y,
      });

      R.cards.map(async (card) => {
        await prepSeats.execute({ row_id: row[0].id, seat_name: card.text });
      });
    }
  }

  return roomPlan;
}

export async function saveStudentSeating(seance_id: string, seating: Seating) {
  const deleteSeating = db
    .delete("studentSeatings")
    .where(eq("studentSeatings.seanceId", sql.placeholder("seanceId")))
    .prepare("delete-student-seating");

  const insertSeating = db
    .insert("studentSeatings")
    .values([
      {
        seanceId: sql.placeholder("seanceId"),
        seatId: sql.placeholder("seatId"),
        studentId: sql.placeholder("studentId"),
      }
    ])
    .prepare("insert-student-seating");

  await deleteSeating.execute({ seanceId: seance_id });

  for (const seatId in seating) {
    if (seating.hasOwnProperty(seatId)) {
      const seatInfo = seating[seatId];

      await insertSeating.execute({
        seanceId: seance_id,
        seatId: seatInfo.seat_id,
        studentId: seatInfo.student.id,
      });
    }
  }

  return { success: true };
}

```

FIGURE 16 – Exemple de fichier de mutation

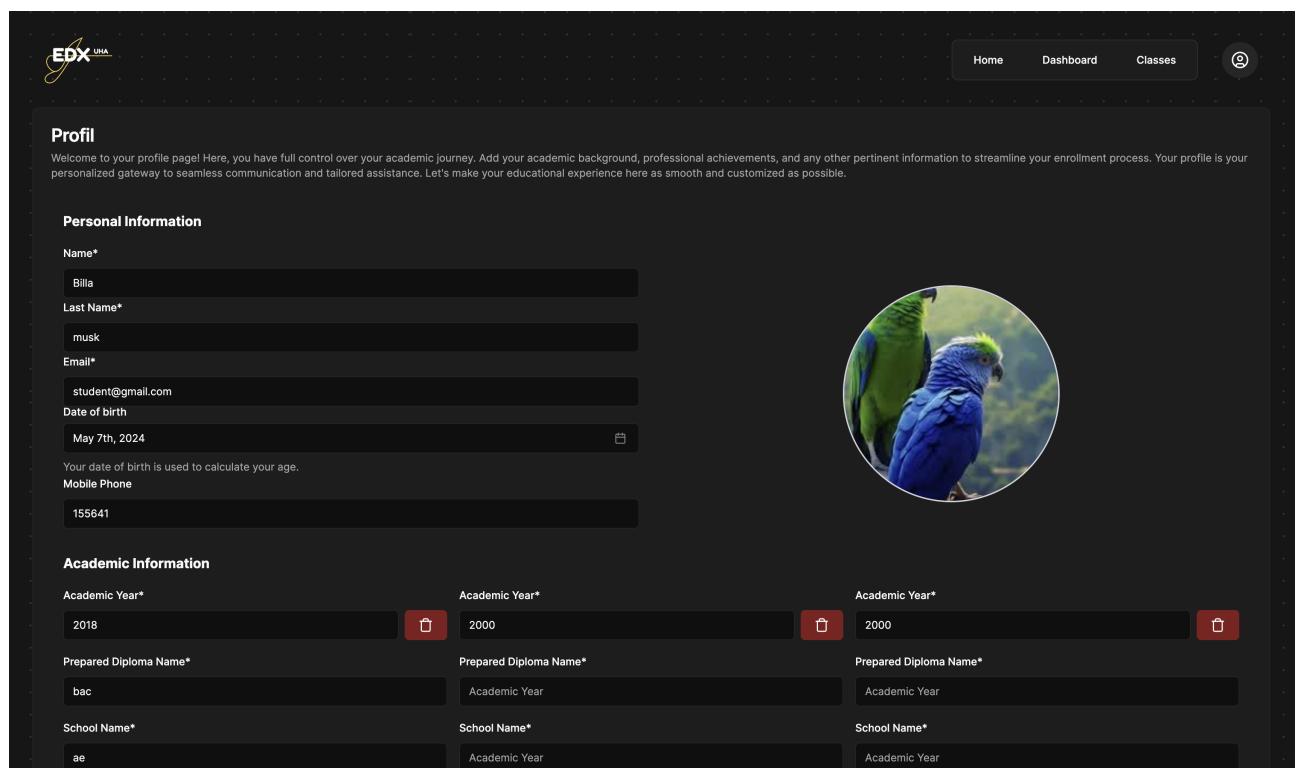
**Vercel** Nous déployons sur Vercel en raison de ses capacités d'hébergement rapide et fiable et de déploiement continu.

## 7 Explication détaillée de la mise en œuvre de la fonctionnalité clés

### 7.1 Gestion des étudiants

La fonctionnalité de gestion des étudiants a été mise en œuvre pour gérer toutes les données et processus liés aux étudiants. Ceci comprend :

- **Profiles des étudiants** : Les profils sont créés et mis à jour à l'aide de formulaires qui valident les entrées avec Zod. Les données sont stockées dans la base de données PostgreSQL via Drizzle ORM. Le schéma comprend des champs pour les informations personnelles, les détails académiques et les informations de contact.
- **Inscription** : Le processus d'inscription comporte plusieurs étapes, notamment la soumission et l'approbation des formulaires. Le statut de chaque demande d'inscription est suivi dans la base de données.
- **Présence** : Le suivi des présences en temps réel est réalisé à l'aide d'une combinaison d'interfaces frontales et d'API backend. Les enseignants peuvent noter les présences à l'aide d'une interface utilisateur réactive et les données sont immédiatement mises à jour dans la base de données.



The screenshot shows the 'Profil' (Profile) page of the EDX LMS. At the top, there's a navigation bar with 'Home', 'Dashboard', 'Classes', and a user icon. The main content area is titled 'Profil' and contains a welcome message: 'Welcome to your profile page! Here, you have full control over your academic journey. Add your academic background, professional achievements, and any other pertinent information to streamline your enrollment process. Your profile is your personalized gateway to seamless communication and tailored assistance. Let's make your educational experience here as smooth and customized as possible.' Below this, there are two sections: 'Personal Information' and 'Academic Information'. The 'Personal Information' section includes fields for Name\*, Last Name\*, Email\*, Date of birth (May 7th, 2024), and Mobile Phone (155641). It also features a circular profile picture of two parrots. The 'Academic Information' section includes fields for Academic Year (2018, 2000, 2000), Prepared Diploma Name (bac, Academic Year), and School Name (ae, Academic Year). Each field has a red trash can icon to its right.

FIGURE 17 – Student Profile Page

The screenshot shows the Teacher Dashboard Page. At the top, there are three main statistics boxes: 'Students' (2350), 'Courses' (4), and 'Enrollment Requests' (80). Below these, the 'Enrollment Requests' section displays a single request from 'Liam Johnson' (liam@example.com) to 'Master 1 IM'. Action buttons for 'Accept' and 'Refuse' are shown. The main content area is titled 'Students' and lists six students with their names, emails, programs, phones, and actions (Modify, Delete).

Student	Program	Phone	Actions
Abe Assoki assoki@gmail.com	Master 1 IM	061234858	
Jonathan Hill john.hill@gmail.com	Master 1 IM	0612738162	
Alice Wonderland alice.wonderland@example.com	Master 1 IM		
Peter Pan peter.pan@example.com	Master 1 IM		
Dorothy Gale dorothy.gale@example.com	Master 1 IM		

FIGURE 18 – Teacher Dashboard Page

The screenshot shows the Presence Tracking Page. A student profile card for 'Oliver Twist' (ID 1008) is displayed, featuring a circular icon with a person symbol, the name 'Oliver Twist', and a timestamp. Below the card are two buttons: a green one with a checkmark and a red one with a minus sign.

FIGURE 19 – Presence Tracking Page

## 7.2 Gestion des enseignants

La gestion des enseignants se concentre sur la tenue de registres précis des informations et des horaires des enseignants :

- **Profiles des enseignants** : À l'instar des profils étudiants, les profils enseignants sont gérés à l'aide de formulaires validés par Zod. Les données comprennent des informations professionnelles, les matières enseignées et les coordonnées.
- **Planification** : Le système de planification permet aux enseignants de créer des séances. Cette fonctionnalité fournit une vue de calendrier implémentée avec des conteneurs glisser-déposer.

### **7.3 Disposition des sièges**

La fonctionnalité Disposition des sièges a été conçue pour faciliter la création et la gestion des plans de salle de classe :

- **Plans de salle dynamiques** : Les utilisateurs peuvent créer et ajuster des plans de salle via une interface glisser-déposer construite avec D3.js et dnd-kit. Les mises en page sont stockées dans la base de données et peuvent être imprimées au format PDF.
- **Attribution de sièges aux étudiants** : Les étudiants se voient attribuer des sièges manuellement ou automatiquement.
- **Maps interactive** : Des plans visuels de la disposition des sièges sont générés et mis à jour en temps réel, permettant des ajustements et des références faciles.

### **7.4 Planification des cours**

La planification des cours implique la création et la gestion des horaires de cours :

- **Création d'horaires** : Les enseignants peuvent créer des horaires à l'aide d'une interface conviviale. Les horaires sont stockés dans la base de données et peuvent être consultés et modifiés selon les besoins.
- **Attribution des salles de classe** : Les cours sont attribués à des salles de classe spécifiques.
- **Intégration du calendrier** : L'intégration future avec les calendriers personnels et institutionnels permettra des horaires synchronisés.

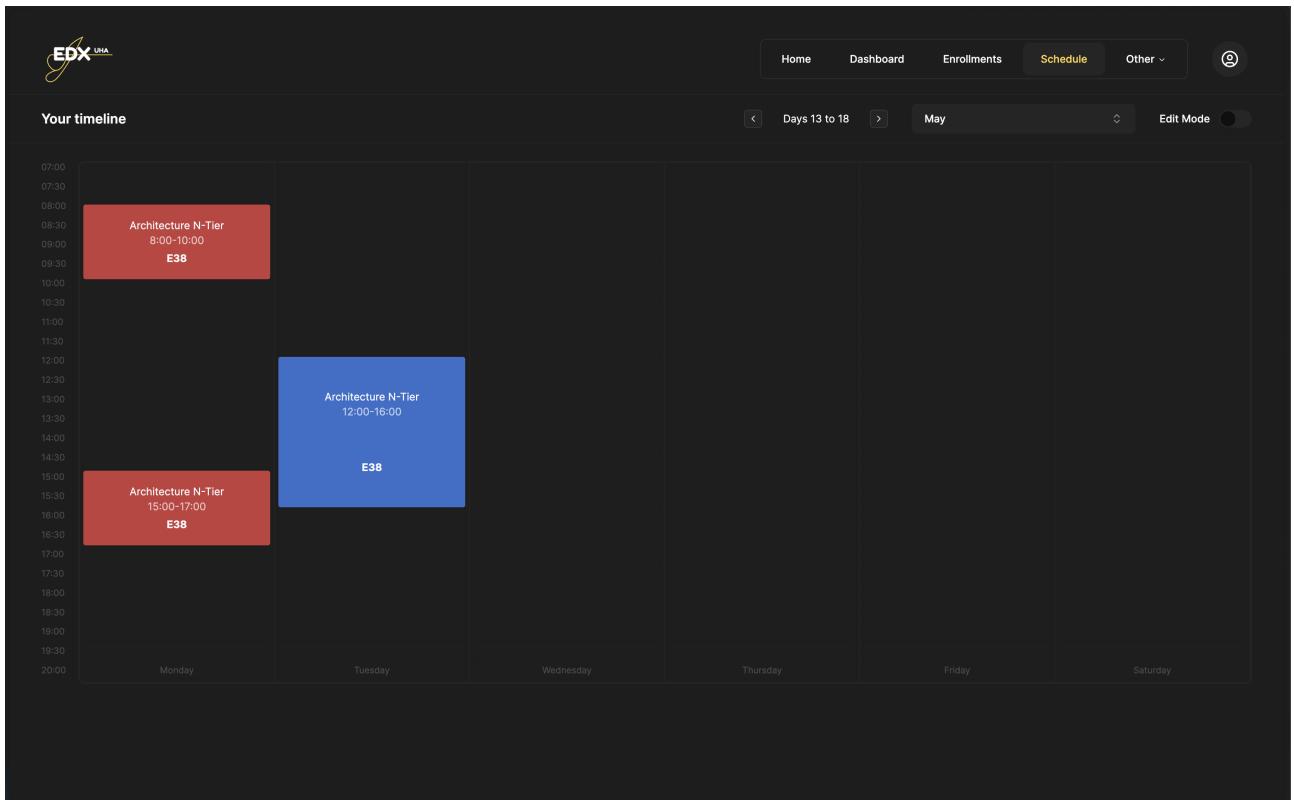


FIGURE 20 – Teacher's schedule page

## 7.5 Sécurité et Authentification

La sécurité est un aspect crucial d'EDX, garantissant que les données sont protégées et que l'accès est contrôlé :

- **Authentication** : Le service d'authentification de Supabase est utilisé pour gérer les connexions et les inscriptions des utilisateurs. Les jetons JWT sont utilisés pour sécuriser les requêtes API.
- **Autorisation** : Le contrôle d'accès basé sur les rôles (RBAC) est important pour garantir que les utilisateurs ne peuvent accéder qu'aux ressources appropriées à leurs rôles.
- **Protection des données** : Outre le cryptage TLS, les données sensibles sont cryptées et des pratiques de codage sécurisées sont suivies pour éviter les vulnérabilités.

## 7.6 Optimisation des Performances

L'optimisation des performances garantit le fonctionnement fluide et efficace d'EDX :

- **Répartition du code** : La division du code intégrée de Next.js garantit que seul le code nécessaire est chargé, améliorant ainsi les temps de chargement.
- **Caching** : La mise en cache stratégique est mise en œuvre à l'aide des fonctions Edge de Vercel pour réduire la charge du serveur et accélérer les

temps de réponse.

- **Optimisation de la base de données** : Des techniques d'indexation et d'optimisation des requêtes sont utilisées pour garantir que les opérations de base de données sont efficaces et rapides.

## 7.7 Intégration et déploiement continus

Les processus CI/CD sont mis en place pour rationaliser le développement et le déploiement :

- **Tests automatisés** : Les tests de code et les tests d'intégration sont exécutés automatiquement du côté de Vercel pour garantir la qualité du code.
- **Pipelines de déploiement** : Vercel gère les déploiements, permettant des mises à jour et des restaurations automatiques si nécessaire.

## 8 Tests et Evaluation

---

### 8.1 Stratégies et méthodologies de test

Compte tenu de la portée et du calendrier du projet EDX, nous avons utilisé plusieurs stratégies pour garantir la qualité et les performances de la plateforme. Au lieu des tests unitaires et d'intégration traditionnels, nous nous sommes concentrés sur les outils automatisés et les évaluations par les pairs.

#### 8.1.1 Outils automatisés

Nous avons utilisé Google Lighthouse et GTmetrix pour évaluer quelques indicateurs clés tels que les performances, l'accessibilité, les meilleures pratiques et le référencement de la plateforme EDX. Ces outils nous ont donné des informations complètes et des recommandations concrètes pour améliorer la qualité de notre application Web.

#### 8.1.2 Tests de performance

- **Temps de chargement des pages :** Les deux outils ont fourni des mesures détaillées sur les temps de chargement des pages, mettant en évidence les domaines dans lesquels des optimisations pourraient réduire les temps de chargement et améliorer l'expérience utilisateur.
- **Accessibilité :** Les audits Lighthouse comprenaient des contrôles d'accessibilité pour garantir que la plate-forme est utilisable par les personnes handicapées. Des recommandations ont été mises en œuvre pour améliorer les scores d'accessibilité.
- **SEO :** Les deux outils ont fourni des informations sur les meilleures pratiques de référencement, nous aidant ainsi à optimiser la plateforme pour les moteurs de recherche.
- **Bonnes pratiques :** Les recommandations de Lighthouse et GTmetrix sur les meilleures pratiques de développement Web ont été mises en œuvre pour améliorer la sécurité, les performances et l'expérience utilisateur globale.

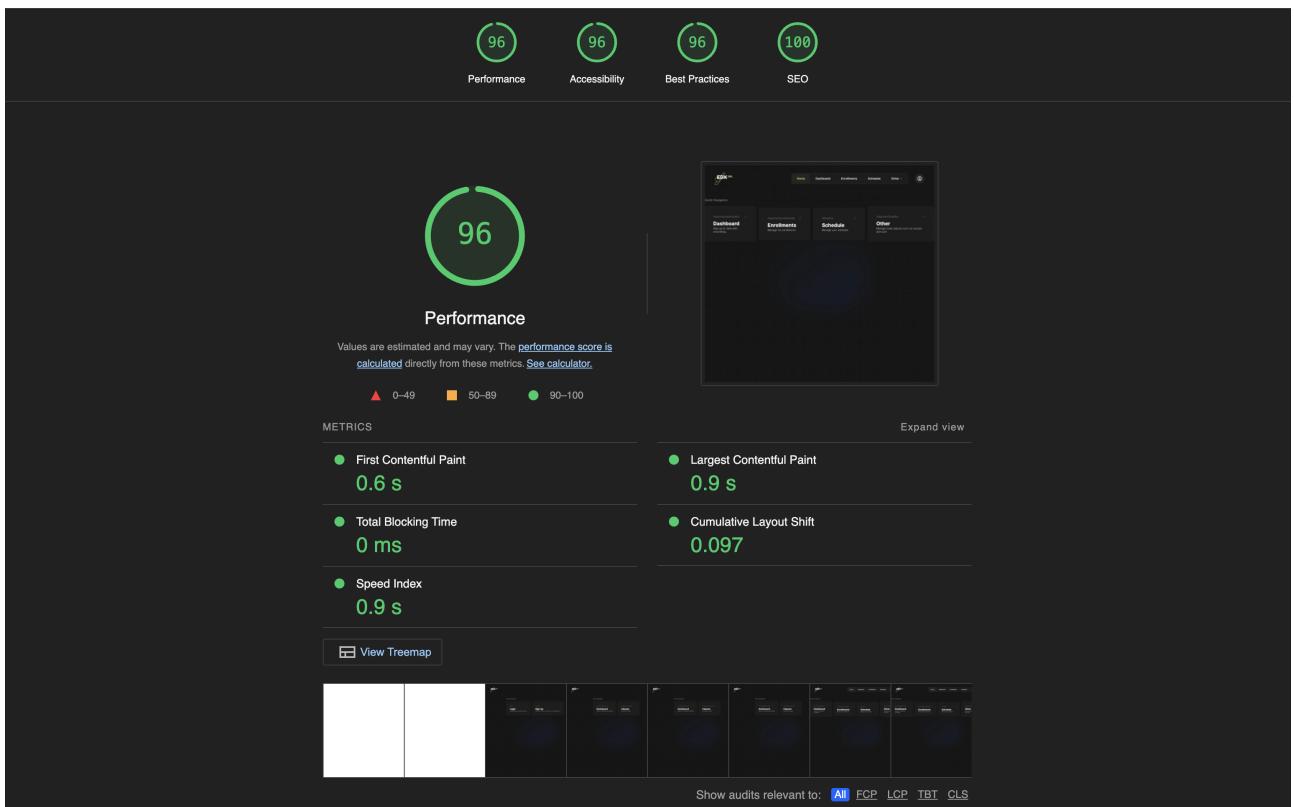


FIGURE 21 – Rapport de Lighthouse

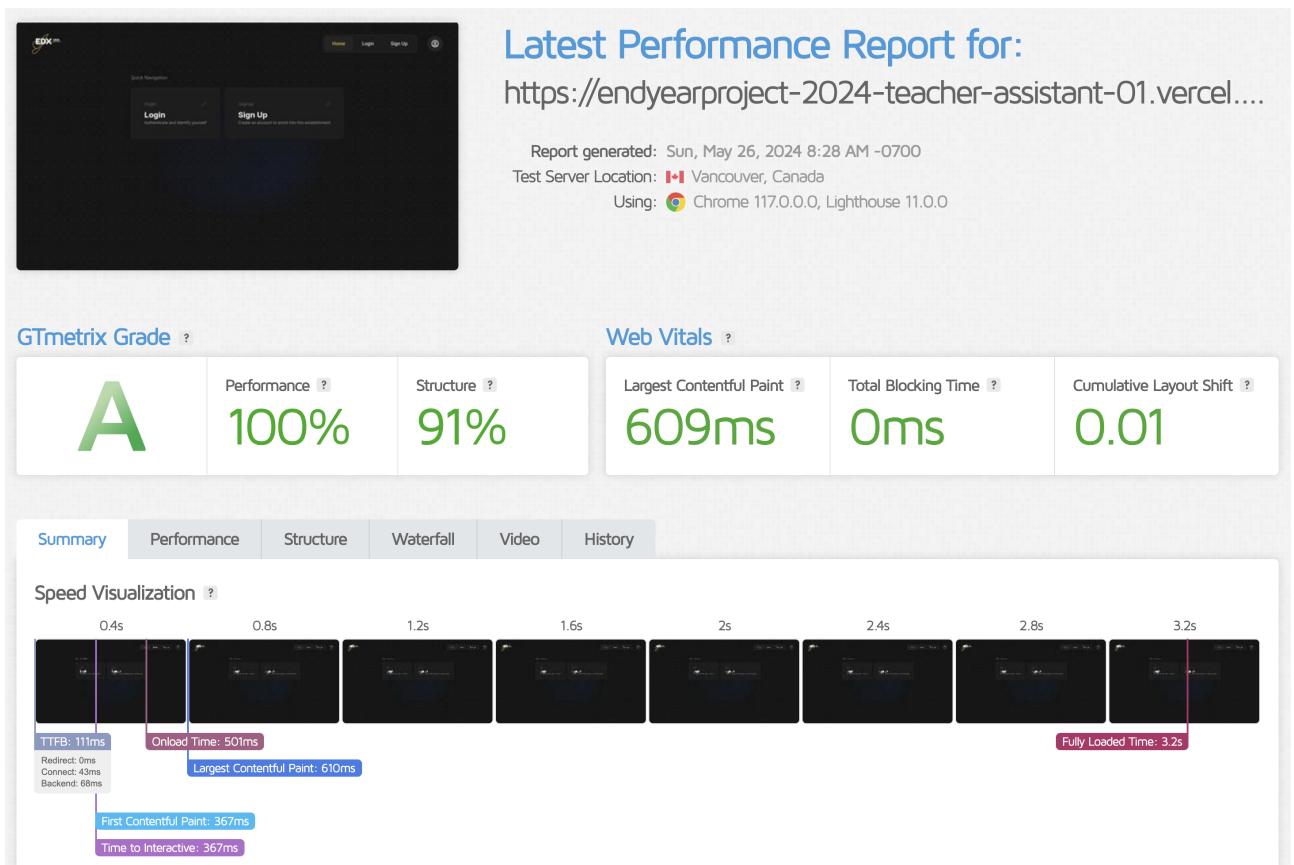


FIGURE 22 – Rapport de GTMetrix

#### **8.1.3 Tests de construction Next.js et contrôle de qualité**

Next.js fournit des tests intégrés pour les processus de construction. Nous avons exploité ces fonctionnalités pour garantir que notre application se construit correctement et fonctionne correctement.

#### **8.1.4 TypeScript et évaluations par les pairs**

TypeScript a joué un rôle crucial dans le maintien de la qualité du code et la prévention des erreurs d'exécution. En appliquant la sécurité des types, TypeScript a aidé à détecter les erreurs dès le début du processus de développement. De plus, nous avons effectué régulièrement des examens par les pairs pour garantir la qualité et la cohérence du code.

#### **8.1.5 Créer des tests et une assurance qualité**

Les tests de build Next.js ont été régulièrement exécutés pour garantir que l'application se compile sans erreurs et que tout problème au moment de la construction est identifié et résolu rapidement.

### **8.2 Évaluation des performances du système**

Dans l'ensemble, les performances de la plateforme EDX ont été évaluées à l'aide des critères suivants :

- **Temps de chargement** : Les évaluations ont montré des temps de chargement acceptables avec des opportunités d'optimisation supplémentaire.
- **Accessibilité** : Les scores d'accessibilité étaient généralement élevés, avec des améliorations continues basées sur les recommandations d'audit.
- **SEO** : Les performances SEO de la plateforme ont été satisfaisantes, toutes les recommandations critiques des audits étant mises en œuvre.
- **Meilleures pratiques** : La plateforme a adhéré aux meilleures pratiques en matière de développement Web, comme l'indiquent les scores élevés des audits Lighthouse et GTmetrix.

#### **8.2.1 Futurs plans de tests**

Bien que la stratégie de test actuelle ait fourni des informations précieuses, les plans futurs comprennent :

- **Stress Testing** : Pour évaluer les performances de la plateforme sous charge, des tests de stress seront effectués. Cela nécessitera plus de temps et de ressources que ceux actuellement disponibles.
- **Tests unitaires et d'intégration** : À l'aide de Storybook et de Jest, on peut implémenter des tests unitaires et d'intégration pour nous s'assurer que chaque partie s'emboîte et fonctionne bien dans le système.

- **Tests automatisés de bout en bout** : On peut Utiliser des outils tels que Cypress ou Selenium pour automatiser les tests de bout en bout, garantissant que les interactions des utilisateurs fonctionnent comme prévu.

L'objectif de ces tests et évaluations est de garantir qu'EDX reste fiable, efficace et facile à utiliser lorsqu'il se développera davantage.

## 9 Défis et Solutions

### 9.1 Problèmes rencontrés lors du développement

Tout au long du développement d'EDX, nous avons été confrontés à plusieurs défis qui ont nécessité des solutions créatives et de la persévérance. Voici les principaux défis et comment nous les avons relevés :

#### 9.1.1 Contraintes de temps

Nous sommes très passionnés par le projet et avons une forte envie d'innover, les contraintes de temps étaient un défi de taille. Notre objectif était de fournir un produit de haute qualité dans un délai limité, ce qui nécessitait une gestion efficace du temps et une priorisation des tâches, mais nous limitait néanmoins en termes d'objectifs que nous souhaitions atteindre.

#### 9.1.2 Configuration de Supabase et Drizzle ORM

Au départ, la configuration de Supabase et de Drizzle ORM était capricieuse. Nous avons rencontré plusieurs problèmes liés à la configuration et à la stabilité, qui ont nécessité plusieurs révisions du code pour garantir une configuration stable et robuste.

#### 9.1.3 Problèmes du fournisseur d'authentification Supabase

Le fournisseur d'authentification Supabase provoquait parfois des problèmes, perturbant le flux d'authentification. Pour résoudre ce problème, nous avons écrit de nouveaux hooks d'authentification personnalisés, qui ont fourni une solution plus fiable et adaptée à nos besoins.

#### 9.1.4 Complexités avec dnd-kit

dnd-kit, est puissant mais manque de documentation complète et ne propose que des exemples simples. Cela nous a amené à passer un temps considérable à comprendre ses subtilités et à développer des interactions complexes sans guides. Malgré le temps investi au départ, le résultat final était très efficace et en valait la peine.

#### 9.1.5 Canevas infini pour planificateur de pièce

Créer un canevas infini pour la page de planification de la pièce était particulièrement difficile. L'intégration de d3.js avec dnd-kit a conduit à des conflits entre les deux bibliothèques. Nous avons expérimenté de nombreuses solutions, chacune introduisant de nouvelles limitations, jusqu'à ce que nous développions une méthode qui évite ces conflits sans imposer de restrictions sur les fonctionnalités.

#### **9.1.6 Calendrier glisser-déposer**

Nous souhaitions avoir une expérience unique et interactive pour la gestion des plannings, cela nous a amené à passer un temps considérable à écrire des fonctions complexes pour gérer toutes sortes d'interactions différentes afin de garantir la meilleure expérience aux utilisateurs.

#### **9.1.7 transmission de propriétés en React**

Nous avons rapidement rencontré le problème de transmission de propriétés, un problème courant dans le développement de React.js où les props doivent être transmises à travers de nombreuses couches de composants. Nous avons résolu cela en adoptant Jotai pour la gestion de l'état, ce qui a rationalisé la manipulation de l'état et réduit la complexité.

#### **9.1.8 Animation avec Framer Motion**

La mise en œuvre d'animations complexes avec Framer Motion a révélé les limites de leur orchestrateur d'animation. Nous avons dû expérimenter diverses approches, souvent par essais et erreurs, pour trouver des solutions répondant à nos exigences. Malgré les défis initiaux, nous avons réalisé les animations souhaitées.

#### **9.1.9 Problèmes de déploiement**

Le déploiement de l'application a posé un autre défi. Vercel nécessite un compte pro pour les déploiements directement à partir du dépôt de l'organisation UHA. Pour contourner ce problème, nous avons transféré le dépôt sur un compte personnel et déployé à partir de là.

### **9.2 Comment ces défis ont été relevés et résolus**

- **Configuration de Supabase et Drizzle ORM :** Nous avons effectué des tests approfondis et affiné de manière itérative notre configuration jusqu'à ce que nous obtenions une configuration stable.
- **Problèmes du fournisseur d'authentification Supabase :** En développant des hooks d'authentification personnalisés, nous avons contourné les limitations du fournisseur d'authentification par défaut et amélioré la fiabilité.
- **Complexités avec dnd-kit :** Des tests et une exploration approfondis nous ont permis de comprendre en profondeur dnd-kit et d'implémenter des fonctionnalités avancées sans guides externes.
- **Infinite Canvas for Room Planner :** Grâce à des expérimentations et des itérations persistantes, nous avons développé une solution qui intègre efficacement d3.js et dnd-kit sans conflits.

- **Prop Drilling dans React** : L'implémentation de Jotai pour la gestion des états a simplifié notre gestion des états et éliminé le problème du forage des props.
- **Animation avec Framer Motion** : Nous avons surmonté les limites de Framer Motion en expérimentant différentes méthodes jusqu'à ce que nous trouvions des solutions efficaces pour des animations complexes.
- **Contraintes de temps** : Une gestion efficace du temps, la priorisation des tâches et la concentration sur les fonctionnalités clés nous ont permis de respecter nos délais sans compromettre la qualité.
- **Problèmes de déploiement** : Le transfert du dépôt vers un compte personnel nous a permis de déployer l'application sur Vercel, contournant les limitations du compte d'organisation UHA.

### 9.3 Dernières réflexions

Malgré les nombreux défis auxquels nous avons été confrontés, nous sommes restés dévoués et passionnés par le projet. Nous sommes reconnaissants de l'opportunité offerte par le professeur Joel Heinis, qui nous a permis d'apprendre, d'innover et de créer une plateforme dont nous sommes fiers. Les obstacles que nous avons rencontrés n'ont fait que renforcer notre détermination et ont contribué au développement d'une plateforme de gestion éducative robuste et efficace.

### 10 Résultats et Discussion

---

#### 10.1 Résumé des résultats obtenus

Le développement d'EDX a conduit à la création d'une plateforme complète de gestion pédagogique qui intègre diverses fonctionnalités dans un système cohérent. Les principales réalisations comprennent :

- Implémentation de fonctionnalités de gestion des étudiants et des enseignants.
- Développement de disposition dynamique des sièges et de modules de planification des cours.
- Intégration du suivi de présence en temps réel.
- Utilisation des technologies Web modernes pour garantir une interface réactive et conviviale.
- Mise en place d'une infrastructure backend robuste avec Supabase, PostgreSQL et Drizzle ORM.

#### 10.2 Comparaison avec les objectifs initiaux

Nos objectifs initiaux étaient de créer une plateforme centralisée qui améliore l'efficacité opérationnelle et l'expérience utilisateur des établissements d'enseignement. Les résultats de notre projet s'alignent étroitement sur ces objectifs :

- **Objectif** : Développer une interface conviviale pour les administrateurs, les enseignants et les étudiants.
- **Résultat** : L'interface utilisateur de la plateforme est intuitive, cohérente et réactive, répondant efficacement aux besoins des utilisateurs.
- **Objectif** : Intégrer des fonctionnalités de base telles que la gestion des étudiants et des enseignants, la disposition des sièges et le suivi de présence en temps réel.
- **Résultat** : Toutes les fonctionnalités de base ont été intégrées avec succès, offrant une expérience utilisateur transparente.
- **Objectif** : Garantir l'évolutivité, la sécurité et les performances.
- **Résultat** : L'infrastructure backend est évolutive et sécurisée, avec des performances optimisées.
- **Objectif** : Utiliser les technologies Web modernes.
- **Résultat** : Des technologies telles que Next.js, TypeScript et Tailwind CSS ont été utilisées efficacement pour améliorer la plate-forme.

### **10.3 Discussion sur l'efficacité et l'impact d'EDX**

L'efficacité d'EDX est évidente dans sa capacité à centraliser et à rationaliser diverses tâches administratives. La plateforme a démontré un potentiel important pour améliorer l'efficacité opérationnelle des établissements d'enseignement en :

- Réduire la dépendance à l'égard de plusieurs outils tiers.
- Fournir des données en temps réel pour une meilleure prise de décision.
- Améliorer la communication et la coordination entre les administrateurs, les enseignants et les étudiants.

L'impact d'EDX s'étend au-delà des fonctionnalités immédiates, offrant une solution évolutive qui peut évoluer avec les besoins de l'institution.

L'utilisation de technologies modernes garantit qu'EDX reste adaptable et capable d'intégrer les avancées futures.

## 11 Travaux futurs

### 11.1 Améliorations potentielles et fonctionnalités supplémentaires

Bien qu'EDX ait atteint ses objectifs principaux, il existe plusieurs domaines d'amélioration potentielle et de fonctionnalités supplémentaires :

- **Amélioration des rapports** : Développer des outils de reporting avancés pour des analyses détaillées sur les performances des étudiants, leur assiduité et d'autres mesures.
- **Support multilingue** : Ajouter le support de plusieurs langues pour répondre à un public plus large.
- **Intégration du calendrier** : Intégrer des systèmes de calendrier externes pour une meilleure gestion des plannings.
- **Tests de stress** : Effectuer des tests de stress pour s'assurer que la plateforme peut gérer un grand nombre d'utilisateurs simultanés.
- **Génération de rapports** : Générer des rapports sur les modèles et les tendances de fréquentation.
- **Résolution des conflits** : Détecter et résoudre les conflits de planification.
- **Notifications** : Envoi d'alertes et de notifications pour les mises à jour liées à la présence.

### 11.2 Vision à long terme de la plateforme

À long terme, nous souhaitons qu'EDX évolue vers un écosystème éducatif à part entière pour tous les aspects de la gestion éducative. Ceci comprend :

- **Fonctionnalité étendue** : Ajoutez continuellement de nouvelles fonctionnalités et améliorations en fonction des commentaires des utilisateurs et des tendances émergentes en matière de technologie éducative.
- **Intégration avec d'autres systèmes** : Garantissez une intégration transparente avec d'autres outils et plates-formes pédagogiques pour offrir une expérience unifiée.

## 12 Conclusion

En conclusion, EDX a fait des progrès significatifs pour devenir un outil essentiel pour la gestion éducative. Les bases posées par ce projet ouvrent la voie à une amélioration et à une expansion continues, en mettant l'accent sur l'amélioration de l'expérience éducative pour tous.