



Adaverse 2.0

- **Durée** : 2 semaines
- **Type de projet** : groupe de 4
- **Nom du repo** : `adaverse-2-[nom_github]`



Modalités de rendu

Vous pouvez développer sur la branche `main`, mais le projet devra être livré sur **deux branches distinctes** :

- `stable` → version **complète avec toutes les fonctionnalités obligatoires, sans bonus**.
- `bonus` → version enrichie avec les **fonctionnalités bonus**.

⚠ **Important** : Vous devez avoir *poussé* (`git push`) votre **dernier commit à minuit au plus tard à la date de rendu transmise**, afin qu'il soit pris en compte pour l'évaluation.



Objectif du projet

Ce projet consiste à ajouter un système d'authentification au projet `adaverse`. Les utilisateurs non connectés pourront continuer à voir les projets sur le site, mais il faudra se connecter pour proposer un nouveau projet. Il sera également possible d'ajouter des projets en favoris et de laisser des commentaires sur les projets lorsqu'on est connecté. Enfin, les utilisateurs pourront avoir un rôle admin. Dans ce cas, ils pourront publier et dépublier les projets du site.



Stack technique (obligatoire)

Élément	Technologie	Lien
Framework fullstack	Next.js	https://nextjs.org/
Framework CSS	TailwindCSS	https://tailwindcss.com/
ORM	Drizzle ORM	https://orm.drizzle.team/
Base de données	Neon	https://neon.com/
Langage	TypeScript	https://www.typescriptlang.org/
Authentification	Better Auth	http://better-auth.com/



Initialisation du projet

La stack étant la même que dans le projet `adaverse`, tu peux y reprendre les instructions. On va juste y ajouter Better Auth, ce que tu peux faire en utilisant le guide **installer Better Auth**.



Déploiement

Tu peux utiliser **Vercel**, la plateforme officielle pour héberger les applications **Next.js**.

Attention, n'oublie pas de spécifier la variable d'environnement pour te connecter à Neon sur Vercel (la même que dans ton fichier `.env`).

💡 Pensez à déployer le plus tôt possible, dès qu'une version basique fonctionnelle existe.

Pré-requis techniques

Pour mener à bien ce projet, il est nécessaire de maîtriser :

- les bases du **terminal**
- les bases du **JavaScript**
- les bases du **React**
- les bases du **SQL**
- les bases du **développement backend**
- l'utilisation d'une ORM
- l'utilisation de Next.js

Fonctionnalités à développer

⚠ **Réalisez toutes les fonctionnalités obligatoires avant de passer aux bonus.**

1. Système d'authentification

Le but de cette partie est d'ajouter le système d'authentification en utilisant Better Auth. Les utilisateur·ices doivent pouvoir créer un compte avec un email et un mot de passe, se connecter et se déconnecter.

- ☐ créer une page avec un formulaire d'inscription (nom d'utilisateur·ice, email, mot de passe)
- ☐ créer une page avec un formulaire de connexion (email et mot de passe)
- ☐ configurer Better Auth (création du client `auth.ts` et des actions `signup`, `signin` et `signout`)
- ☐ gérer les erreurs et états de chargement des formulaires
- ☐ ajouter un bouton de déconnexion dans la barre de navigation lorsque l'utilisateur·ice est connecté·e, sinon mettre 2 liens vers `signin` et `signup`
- ☐ depuis les pages d'inscription et connexion, si l'utilisateur·ice est connecté·e, iel doit être redirigé·e sur la page d'accueil automatiquement

2. Les commentaires

Le but de cette partie est d'ajouter des fonctionnalités qui manipulent des données liées aux utilisateur·ices.

- ☐ Lorsqu'un·e utilisateur·ice propose un projet, faire en sorte qu'il soit lié à son compte en base de données
- ☐ Afficher le nom de l'utilisateur·ice qui l'a proposé dans la liste des projets et sur la page de détail du projet
- ☐ Permettre à n'importe quelle personne connectée de poster un commentaire sur un projet.
 - ☐ Les commentaires devront être sauvegardés en base de données
 - ☐ Les commentaires devront avoir un message ainsi qu'une date de publication

- ☐ Les commentaires seront affichés sous la page de détail de chaque projet, du plus récent au plus ancien, avec leur date, leur auteur ·ice et le message
- ☐ Dans la liste des projets, on affichera le nombre total de commentaires
- ☐ Les utilisateur ·ices pourront supprimer et modifier **leurs propres** commentaires

3. Le rôle admin

Dans cette partie, on va ajouter un rôle admin aux utilisateur ·ices. Les fonctionnalités listées ci-dessous ne devront être accessibles qu'aux admin.

- ☐ Supprimer un commentaire de n'importe quel ·le utilisateur ·ice
- ☐ Créer une page qui liste **tous** les projets et qui permet de publier ou de désactiver la publication d'un projet (attention : si un ·e utilisateur ·ice n'est pas admin, iel devra être automatiquement redirigé ·e vers l'accueil depuis cette page)
- ☐ Bannir un ·e utilisateur ·ice

4. Mobile et accessibilité

Attention, votre site devra être utilisable sur mobile avec un design adapté.

Les éléments utilisés devront également être accessibles (à minima avoir 100% dans les catégories Lighthouse accessibilité et best practices).

Bonus

- Partage de projets sur les réseaux sociaux
- Page de profil
- Connexion et gestion de compte avec GitHub
- Intégration d'une bibliothèque de validation (zod, valibot, arktype, etc.)
- Système de favoris
- Ajouter un système d'upload d'image

Bonus ++

- Système de recherche de projets (filtre, barre de recherche, "fulltext search")
- Associer plusieurs utilisateur ·ices à un projet
- Bonus du projet **adaverse**
- Soyez créatif ·ves !

Modalité d'évaluation

Soutenance (20 minutes)

L'évaluation se déroulera sous forme de soutenance. Chaque membre du groupe devra présenter une partie des fonctionnalités sur lesquelles iel a travaillé.

Chaque personne devra être capable :

- d'expliquer clairement son code à l'oral,
- de réaliser une démonstration fonctionnelle,

- de naviguer aisément dans son code durant la présentation.

Cette première partie durera **environ 3 minutes par personne**.

Ensuite, chaque membre du groupe répondra à **une à deux questions techniques**, afin de vérifier ses compétences (basées sur le référentiel *Compétences à acquérir* ci-dessous). Cette phase durera **environ 2 minutes par personne**.

Démo (5 minutes)

Le groupe devra préparer une **présentation complète du projet**, comprenant :

- une démo fluide et représentative,
- des slides clairs et synthétiques,
- une explication des **choix techniques** effectués,
- une mise en avant de l'**organisation du travail en équipe** (répartition des tâches, workflow Git, communication, etc.).

L'objectif est de montrer votre compréhension globale du projet, autant sur le plan technique que sur le plan méthodologique.

⚠ Attention à bien répartir le temps de parole entre les 4 membres du groupe

Compétences à acquérir

- ☐ Je sais créer un système de comptes utilisateur · ices avec email et mot de passe (signin, signup, signout)
- ☐ Je sais utiliser les cookies pour sauvegarder une session utilisateur · ice
- ☐ Je sais charger des données en fonction d'un compte utilisateur · ice connecté · e
- ☐ Je sais protéger l'accès à certaines données si l'utilisateur · ice est connecté · e ou non
- ☐ Je sais lier des données à un · e utilisateur · ice (commentaires, favoris)
- ☐ Je sais gérer plusieurs rôles utilisateur · ices (user, admin)
- ☐ Je sais effectuer des redirections entre les pages (par exemple, vers /auth/signin si l'utilisateur · ice n'est pas connecté · e)