GÉNÉRATEUR D'EXERCICES

Résistance des Matériaux

Spécification Fonctionnelle et Technique

Contexte	Université de Bordeaux
Domaine	Résistance des matériaux, PFS, Poutres isostatiques
Utilisateurs	1 professeur, ~30 étudiants
Budget	Gratuit (stack open source)

TABLE DES MATIÈRES

1. VUE D'ENSEMBLE DU PROJET

Objectif

Créer une plateforme web permettant à un professeur de générer des exercices de résistance des matériaux via l'intelligence artificielle, de les publier à ses étudiants avec des variantes personnalisées (anti-triche), et d'obtenir une correction automatique avec suivi statistique.

Problématique résolue

Problème	Solution apportée
Exercices identiques = triche facile	Variantes automatiques par étudiant (valeurs différentes)
Correction manuelle chronophage	Correction automatique avec tolérance paramétrable
Pas de suivi individuel	Tableau de bord avec stats par étudiant et par exercice
Création d'exercices longue	Génération IA + bibliothèque de modèles réutilisables

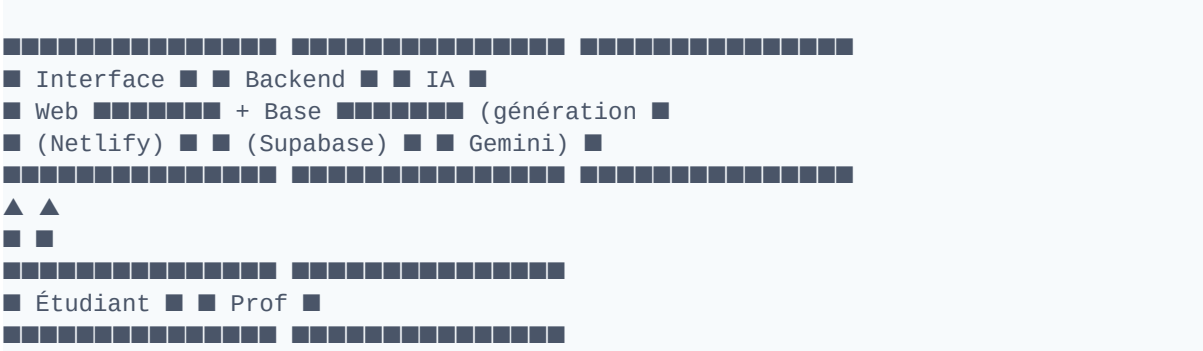
Acteurs

Acteur	Rôle	Actions principales
Professeur	Administrateur	Génère, valide, publie exercices. Gère étudiants. Consulte stats.
Étudiant	Utilisateur	Consulte exercices, soumet réponses, voit feedback.
Système IA	Générateur	Génère énoncés, plages de valeurs, résolutions détaillées.

2. ARCHITECTURE TECHNIQUE

Vue d'ensemble

L'architecture suit un modèle classique frontend/backend avec une couche IA pour la génération de contenu.



Composants

Composant	Technologie	Rôle
Frontend	Next.js / React	Interface utilisateur responsive
Hébergement	Netlify (gratuit)	Déploiement automatique, CDN
Backend + Auth	Supabase (gratuit)	API, authentification, base PostgreSQL
Base de données	PostgreSQL (Supabase)	Stockage exercices, users, stats
Stockage fichiers	Supabase Storage	Images des exercices
IA Génération	Google Gemini API (gratuit)	Génération énoncés et résolutions

3. CONCEPTS CLÉS : TYPES ET MODÈLES

Deux niveaux de réutilisation

Le système distingue deux concepts fondamentaux pour organiser les exercices :

TYPE (Squelette mathématique)

Un TYPE définit la structure mathématique d'un exercice. Il contient les formules, les variables impliquées et l'unité de la réponse. Les types sont soit pré-remplis (bibliothèque de base), soit créés par le professeur.

Élément	Exemple : Flexion poutre bi-appuyée
Formules	Mf = (P×L)/4, I = (b×h³)/12, σ = (Mf×y)/I
Variables	L (longueur), b (largeur), h (hauteur), P (charge)
Unité réponse	MPa

MODÈLE (Exercice sauvegardé)

Un MODÈLE est un exercice complet personnalisé que le professeur sauvegarde pour le réutiliser ultérieurement. Il contient tout du TYPE plus l'énoncé, les plages de valeurs, l'image optionnelle, la tolérance, etc.

Élément	Exemple : Modèle 'Flexion passerelle bois'
Basé sur	TYPE flexion bi-appuyée
Énoncé	Une passerelle en bois de longueur {L} m...
Plages	L [3-6m], b [15-25cm], h [30-50cm], P [5-20kN]
Image	passerelle.png
Tolérance	±3%

Flux de création

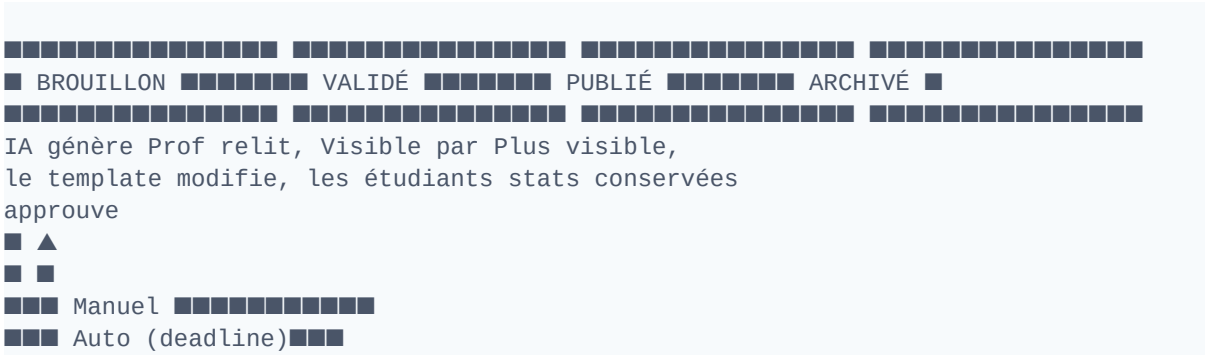
```
TYPE (formules fixes)
■
■ Prof choisit un type
▼
IA GÉNÈRE (énoncé, plages, résolution)
■
■ Prof ajuste et valide
▼
EXERCICE
■
■■■■ Prof publie aux étudiants
■
■■■■ Prof sauvegarde en MODÈLE (optionnel)
■
■ Plus tard : "Réutiliser"
▼
NOUVEL EXERCICE (pré-rempli)
```

Ce que l'IA génère vs ce qui est fixe

Élément	Fixe ou Généré ?	Détail
Formules	FIXE	Définies par le TYPE choisi
Variables	FIXE	Découlent des formules
Unité réponse	FIXE	Définie par le TYPE
Énoncé contextualisé	GÉNÉRÉ	Mise en situation réaliste
Plages min/max	GÉNÉRÉ	Valeurs cohérentes proposées
Résolution détaillée	GÉNÉRÉ	Explication pédagogique étape par étape

4. CYCLE DE VIE D'UN EXERCICE

États possibles



Détail des états

État	Description	Actions possibles
BROUILLON	Exercice généré par l'IA, en attente de validation	Modifier, Supprimer, Valider
VALIDÉ	Approuvé par le prof, prêt à publier	Modifier, Publier, Supprimer
PUBLIÉ	Visible et accessible aux étudiants	Archiver (manuel ou auto)
ARCHIVÉ	Terminé, plus accessible aux étudiants	Consulter stats, Afficher correction

Règles de transition

- Modifier un exercice PUBLIÉ = création d'un NOUVEL exercice (l'ancien reste intact)
- Archivage automatique si une date limite est définie et dépassée
- Archivage manuel possible à tout moment par le professeur
- Option d'afficher la correction aux étudiants après archivage

5. WORKFLOW UTILISATEUR

Workflow Professeur

Étape 1 : Génération d'exercices

1. Le prof choisit un TYPE dans la bibliothèque (ex: Flexion poutre bi-appuyée)
2. Il clique sur 'Générer avec l'IA'
3. L'IA génère : énoncé contextualisé, plages de valeurs, résolution détaillée
4. L'exercice est créé en statut BROUILLON

Étape 2 : Validation

Le prof visualise et peut modifier :

- L'énoncé (texte libre)
- Les plages de variables : min, max, mode (libre avec arrondi OU paliers fixes)
- La tolérance de réponse (ex: ±2%)

Il peut consulter (lecture seule) : formules utilisées, résolution détaillée

Étape 3 : Publication

Lors de la publication, le prof définit :

- Titre affiché aux étudiants
- Image optionnelle (upload)
- Date limite (optionnelle) avec archivage automatique
- Affichage de la correction après archivage (oui/non)

→ Le système génère automatiquement une VARIANTE unique par étudiant (valeurs différentes calculées selon les plages)

Étape 4 : Suivi et archivage

- Consultation des statistiques en temps réel
- Archivage manuel ou automatique (deadline)
- Optionnel : sauvegarde de l'exercice en MODÈLE pour réutilisation

Workflow Étudiant

1. Connexion avec le compte créé par le professeur
2. Visualisation de la liste des exercices disponibles (avec deadlines)
3. Ouverture d'un exercice : affichage de SA variante (valeurs uniques)
4. Saisie de la réponse (champ numérique + unité)
5. Soumission → feedback immédiat : correct/incorrect
6. Possibilité de réessayer (nombre illimité de tentatives)
7. Après archivage : accès à la correction si activée par le prof

6. STRUCTURE DES DONNÉES

Schéma relationnel

Table	Description	Champs principaux
users	Professeurs et étudiants	id, email, nom, role (prof/étudiant), créé_par
types	Types d'exercices (squelettes)	id, nom, catégorie, formules, variables, unité, créé_par
templates	Modèles sauvegardés (bibliothèque prof)	id, type_id, titre, énoncé, plages_json, image_url, tolérance, créé_par
exercises	Exercices créés	id, type_id, template_id (nullable), titre, énoncé, plages_json, statut, image_url, tolé
exercise_instances	Variante par étudiant	id, exercise_id, student_id, valeurs_json, réponse_attendue
attempts	Tentatives de réponse	id, instance_id, réponse_donnée, correct (bool), timestamp

Relations entre tables

```
types ■■■■■■■■ exercises ■■■■■■■■ exercise_instances ■■■■■■■■ attempts
■ ■
■ ■
templates users
(bibliothèque) (étudiants)
```

Exemple de données JSON

plages_json (dans exercises) :

```
{
"L": {"min": 2, "max": 8, "mode": "palier", "pas": 0.5, "unite": "m"},
"b": {"min": 10, "max": 30, "mode": "libre", "decimales": 0, "unite": "cm"},
"h": {"min": 20, "max": 50, "mode": "libre", "decimales": 0, "unite": "cm"},
"P": {"min": 5, "max": 50, "mode": "palier", "pas": 5, "unite": "kN"}
}
```

valeurs_json (dans exercise_instances) :

```
{
"L": 4.5,
"b": 15,
"h": 35,
"P": 25
}
```

7. INTERFACES UTILISATEUR

Interface Professeur

Écran : Édition d'exercice

L'écran d'édition présente toutes les informations générées par l'IA de manière claire :

- Section ÉNONCÉ : texte modifiable avec variables entre accolades {L}, {b}...
- Section FORMULES UTILISÉES : liste des formules (lecture seule)
- Section RÉSOLUTION DÉTAILLÉE : étapes de calcul (lecture seule, repliable)
- Section VARIABLES : pour chaque variable, champs min/max et choix du mode
- Section FORMULE FINALE : formule de calcul et tolérance
- APERÇU : prévisualisation avec valeurs aléatoires + réponse calculée

Mode de génération des valeurs

Mode	Description	Exemple
Valeurs libres	N'importe quelle valeur entre min et max, arrondie	L = 3.7 m (arrondi 1 décimale)
Paliers	Valeurs par incréments fixes	L = 2, 2.5, 3, 3.5... (pas de 0.5)

Écran : Publication

- Titre affiché
- Upload image (optionnel, drag & drop)
- Date limite : aucune OU date/heure précise
- Checkbox : archivage automatique après deadline
- Checkbox : afficher correction après archivage

Écran : Bibliothèque de modèles

- Liste des modèles sauvegardés avec recherche et filtres
- Pour chaque modèle : titre, variables, date création, nombre d'utilisations
- Actions : Utiliser, Modifier, Supprimer

Écran : Statistiques

- Par exercice : taux de réussite, nombre tentatives moyen, distribution des réponses
- Par étudiant : exercices complétés, score global, progression

Interface Étudiant

Écran : Liste des exercices

- Exercices disponibles avec statut (à faire, réussi, échoué)
- Deadline affichée si définie ('dans 3 jours', 'demain'...)
- Notification visuelle pour deadlines proches

Écran : Exercice

- Image (si uploadée par le prof)
- Énoncé avec les valeurs spécifiques à l'étudiant
- Champ de saisie numérique + unité
- Bouton Soumettre
- Feedback immédiat après soumission
- Possibilité de réessayer si incorrect

8. FONCTIONNALITÉS DÉTAILLÉES

Système anti-triche

Chaque étudiant reçoit une variante unique de l'exercice. Les valeurs numériques sont générées aléatoirement dans les plages définies par le professeur. Ainsi, même si deux étudiants échangent leurs réponses, celles-ci seront incorrectes car les valeurs diffèrent.

Correction automatique

- La réponse attendue est calculée automatiquement pour chaque variante
- Tolérance paramétrable (ex: ±2%) pour accepter les arrondis
- Comparaison numérique avec la réponse de l'étudiant
- Feedback immédiat : correct/incorrect

Gestion des deadlines

- Date limite optionnelle pour chaque exercice
- Affichage du temps restant côté étudiant
- Notification visuelle à l'approche de la deadline
- Archivage automatique une fois la deadline passée
- Archivage manuel possible à tout moment

Gestion des comptes

- Le professeur crée les comptes étudiants
- Authentification via Supabase Auth (email/mot de passe)
- Rôles distincts : professeur et étudiant

Bibliothèque de types

Le système est livré avec une bibliothèque de types pré-remplis couvrant les cas classiques en résistance des matériaux. Le professeur peut également créer ses propres types.

Catégorie	Types inclus
Flexion	Poutre bi-appuyée (charge ponctuelle, répartie), Poutre encastrée-libre
Torsion	Arbre cylindrique plein, Arbre cylindrique creux
Traction/Compression	Barre simple, Système hyperstatique
Contraintes thermiques	Barre encastrée, Assemblage bi-matériau
PFS	Systèmes isostatiques divers

9. STACK TECHNIQUE

Choix technologiques (100% gratuit)

Besoin	Solution	Tier gratuit
Frontend	Next.js + React	Open source
Hébergement frontend	Netlify	100 GB bande passante/mois
Backend + API	Supabase	50k requêtes/mois
Base de données	PostgreSQL (Supabase)	500 MB
Authentification	Supabase Auth	50k utilisateurs actifs/mois
Stockage fichiers	Supabase Storage	1 GB
Génération IA	Google Gemini API	1500 requêtes/jour

Configuration Google Gemini

L'IA de génération utilise l'API Gemini de Google. La configuration est simple et gratuite :

1. Avoir un compte Google (Gmail personnel ou universitaire)
2. Aller sur Google AI Studio : https://aistudio.google.com
3. Cliquer sur 'Get API Key' → 'Create API Key'
4. Copier la clé et la configurer dans l'application

Aucune carte bancaire n'est requise. Le tier gratuit offre 1500 requêtes/jour, largement suffisant puisque l'API n'est appelée que lors de la génération d'exercices (pas lors des réponses des étudiants).

Limites des tiers gratuits

Service	Limite	Usage estimé (30 étudiants)
Supabase - BDD	500 MB	~10 MB
Supabase - Auth	50k utilisateurs/mois	31 utilisateurs
Supabase - Storage	1 GB	~100 MB (images)
Netlify - Bande passante	100 GB/mois	~2 GB/mois
Netlify - Build	300 min/mois	~10 min/mois
Gemini - Requêtes	1500/jour	~10-20/jour max

Dimensionnement

Pour 30 étudiants avec une utilisation régulière (quelques exercices par semaine), les tiers gratuits sont largement suffisants. Toutes les limites sont au moins 10x supérieures aux besoins réels du projet.

Évolutivité

Si le projet grandit (plus d'étudiants, plus de professeurs), les solutions choisies permettent une montée en charge facile avec des plans payants abordables (~19$/mois pour Netlify Pro, ~25$/mois pour Supabase

Pro).

RÉCAPITULATIF

Fonctionnalité	Inclus
Génération d'exercices par IA	✓
Variantes uniques par étudiant (anti-triche)	✓
Correction automatique avec tolérance	✓
Bibliothèque de types pré-remplis	✓
Bibliothèque de modèles personnels	✓
Validation par le professeur avant publication	✓
Deadlines avec archivage automatique	✓
Upload d'images par exercice	✓
Statistiques par étudiant et par exercice	✓
Gestion des comptes étudiants par le prof	✓
Affichage correction après archivage	✓
Tentatives multiples autorisées	✓
100% gratuit	✓