



Conception détaillée

NIS CUBE

Abdelwaheb SEBA
Mohammed Ryad DERMOUCHE
Chatodit LOKONGA KONGOLO
Sedik SI MEHAND

Groupe L3L1
2024 - 2025

Informations sur le document

Les informations d'identification du document

Référence du document : NIS-CD-V1.00

Version du document : 1.00

Date du document : 02/03/2025

Auteur(s) : Groupe L3L1

Les éléments de vérification du document

Validé par : Nicolas DENIS

Validé le : 02/03/2025

Soumis le : 02/03/2025

Type de diffusion : Document électronique (.pdf)

Confidentialité : Réservé aux membres du groupe L3L1

L'encadrant monsieur Nicolas DENIS

Le responsable de l'UE monsieur David JANISZEK

Les membres du jury lors de la soutenance du projet

Table des matières

| | | |
|----------|-------------------------------------------------|-----------|
| 1 | Introduction | 3 |
| 1.1 | Contexte | 3 |
| 1.2 | Historique | 3 |
| 1.3 | Objectifs du document | 3 |
| 2 | Spécifications techniques | 4 |
| 2.1 | Maquette | 4 |
| 2.2 | Technologies utilisées | 13 |
| 2.3 | Base de données et modèle relationnel | 15 |
| 2.4 | Diagramme de classes | 20 |
| 2.5 | Diagramme de Gantt | 24 |
| 3 | Conclusion | 25 |

1 Introduction

1.1 Contexte

Avec l'évolution constante des cybermenaces et l'entrée en vigueur de la directive **NIS2** (Network and Information Security), les entreprises et administrations doivent renforcer leur niveau de sécurité et assurer leur conformité aux nouvelles exigences européennes. Afin de répondre à ce besoin, le projet **NIS CUBE** propose une solution innovante sous la forme d'une **interface web** permettant aux entreprises d'estimer leur niveau de mise en conformité et, par conséquent, d'améliorer leur sécurité.

Le principe du projet repose sur un processus d'**évaluation interactive** : les utilisateurs répondent à un ensemble de questions visant à mesurer la conformité des entités qu'ils représentent par rapport aux exigences de la directive NIS2. En fonction des réponses fournies, des **scores** sont attribués, et les résultats sont affichés sous forme de **représentations graphiques** afin d'identifier clairement les aspects à améliorer.

1.2 Historique

Historiquement, la première directive **NIS** (2016) avait pour objectif d'établir un niveau élevé commun de sécurité des réseaux et des systèmes d'information à travers l'Union européenne. Face à l'augmentation des menaces informatiques et à la nécessité d'impliquer un plus grand nombre d'acteurs, la directive **NIS2** est venue renforcer et étendre les obligations de sécurité dans plusieurs secteurs (énergie, transports, santé, infrastructures numériques, etc.).

Ces évolutions ont généré de nouveaux besoins : les organisations doivent désormais rendre compte de leur conformité plus précisément et faire preuve de davantage de réactivité en matière de cybersécurité. C'est dans ce contexte que s'inscrit **NIS CUBE**, afin de fournir aux acteurs concernés un outil simple et adapté pour piloter leur mise en conformité.

1.3 Objectifs du document

Ce document a pour objectif de fournir une vision complète et détaillée de l'ensemble des composantes techniques et organisationnelles du projet NIS CUBE. Il regroupe les éléments essentiels pour assurer la cohérence, la traçabilité et la planification du développement, à savoir :

- **Les maquettes de l'interface** : Présentation visuelle et ergonomique des écrans de l'application.
- **Les technologies utilisées** : Description des outils et langages choisis pour le développement du back-end, du front-end et de la gestion de la base de données.
- **Le modèle relationnel et la base de données** : Détail de la structure de stockage des données, avec la description des tables, champs, clés et contraintes.
- **Le diagramme de classes** : Illustration de l'architecture objet du projet, montrant les relations entre les différentes entités.
- **Le diagramme de Gantt** : Planification temporelle du projet incluant l'estimation des temps de développement et l'attribution des tâches.

Ce document servira de référence tout au long du développement, permettant de suivre les choix techniques et organisationnels, et d'assurer la conformité du produit final aux exigences initiales.

2 Spécifications techniques

2.1 Maquette

Cette section présente les maquettes du projet NIS CUBE ainsi que la palette de couleurs qui sert de guide pour l'interface utilisateur. Les images sont organisées selon l'ordre de navigation logique de l'application, depuis la page d'accueil jusqu'à la consultation de l'historique d'évaluation.

Palette de Couleurs de NIS CUBE

- **Couleurs Principales :**
 - `-primary: #2563eb`
Utilisée pour les éléments interactifs principaux (boutons, liens, icônes).
 - `-primary-hover: #1d4ed8`
Teinte plus foncée de `-primary`, indiquant l'état de survol (hover).
 - `-primary-focus: rgba(37, 99, 235, 0.125)`
Couleur semi-transparente appliquée lors de la sélection ou de l'activation d'un élément.
- **Couleurs de Fond :**
 - `-background: #f8fafc`
Couleur de fond principale, assurant une bonne lisibilité.
 - `-white: #ffffff`
Utilisée pour les cartes, formulaires et éléments se démarquant du fond principal.
- **Couleurs de Texte :**
 - `-text: #334155`
Couleur principale du texte, offrant un contraste élevé sur fond clair.
 - `-text-light: #64748b`
Couleur plus claire pour les sous-titres, légendes et informations secondaires.

Maquettes des Interfaces

1. Page d'accueil



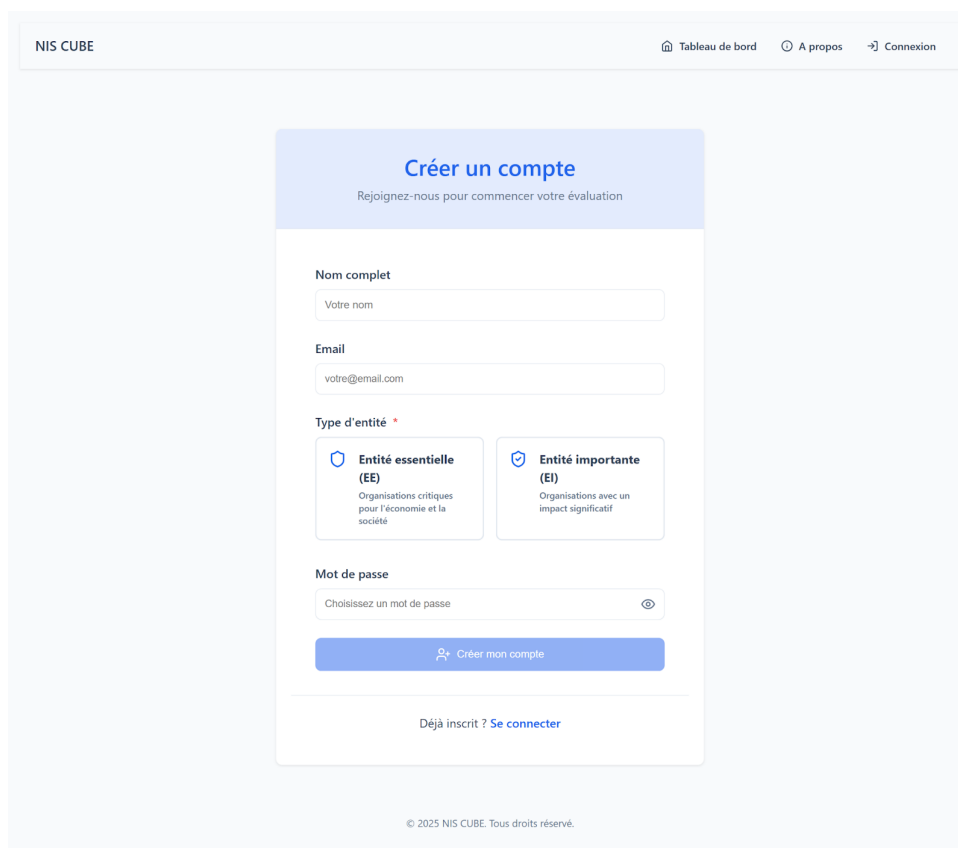
FIGURE 1 – Interface d'accueil de NIS CUBE

2. À propos



FIGURE 2 – Page « À propos » décrivant NIS2 et l'outil d'auto-évaluation

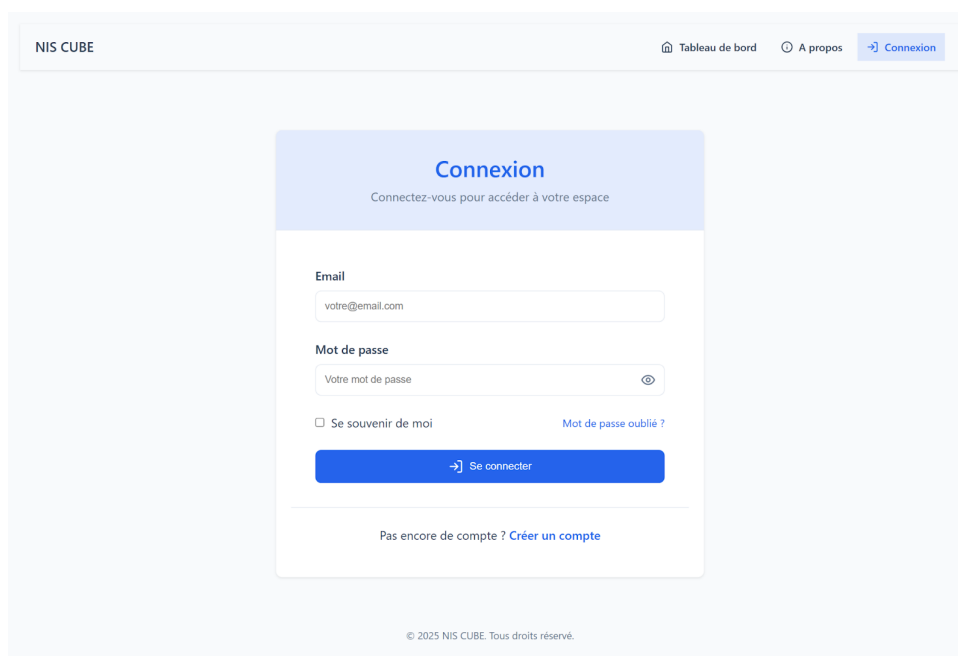
3. Inscription



The registration form is titled "Créer un compte" (Create an account) with the subtitle "Rejoignez-nous pour commencer votre évaluation" (Join us to start your evaluation). It is located within a header bar that includes the "NIS CUBE" logo and navigation links for "Tableau de bord" (Dashboard), "A propos" (About), and "Connexion" (Login). The form fields include: "Nom complet" (Full name) with a placeholder "Votre nom"; "Email" with a placeholder "votre@email.com"; "Type d'entité" (Entity type) with two options: "Entité essentielle (EE)" (Essential entity) described as "Organisations critiques pour l'économie et la société" (Critical organizations for the economy and society) and "Entité importante (EI)" (Important entity) described as "Organisations avec un impact significatif" (Organizations with a significant impact); and "Mot de passe" (Password) with a placeholder "Choisissez un mot de passe" (Choose a password) and a toggle for visibility. A blue button labeled "Créer mon compte" (Create my account) is positioned below the fields. At the bottom of the form, there is a link "Déjà inscrit ? Se connecter" (Already registered? Log in). The footer of the form area contains the copyright notice "© 2025 NIS CUBE. Tous droits réservés."

FIGURE 3 – Formulaire de création de compte

4. Connexion




The login form is titled "Connexion" (Login) with the subtitle "Connectez-vous pour accéder à votre espace" (Log in to access your space). It is located within a header bar that includes the "NIS CUBE" logo and navigation links for "Tableau de bord" (Dashboard), "A propos" (About), and "Connexion" (Login), which is highlighted. The form fields include: "Email" with a placeholder "votre@email.com"; and "Mot de passe" (Password) with a placeholder "Votre mot de passe" (Your password) and a toggle for visibility. Below the password field, there is a checkbox for "Se souvenir de moi" (Remember me) and a link "Mot de passe oublié ?" (Forgot password?). A blue button labeled "Se connecter" (Log in) is positioned below the fields. At the bottom of the form, there is a link "Pas encore de compte ? Créer un compte" (No account yet? Create an account). The footer of the form area contains the copyright notice "© 2025 NIS CUBE. Tous droits réservés."

FIGURE 4 – Interface de connexion pour accéder à l'espace personnel


5. Mon Profil

NIS CUBE

[Tableau de bord](#) [Historique](#) [Mon Profil](#) [A propos](#) [Déconnexion](#)

 **Mon Profil**

Gérez vos informations personnelles et vos préférences

 Informations personnelles


Nom d'utilisateur

jean.dupont


Adresse e-mail

jean.dupont@example.com


Type d'entité

 Entité essentielle (EE)

Organisations critiques pour l'économie et la société

 Entité importante (EI)

Organisations avec un impact significatif

 Sécurité

Mot de passe actuel

Entrez votre mot de passe actuel

Nouveau mot de passe


Entrez un nouveau mot de passe


Confirmer le mot de passe

Confirmez votre nouveau mot de passe

Annuler

Enregistrer les modifications

 Zone de danger

 Supprimer mon compte

© 2025 NIS CUBE. Tous droits réservés.

FIGURE 5 – Gestion des informations personnelles et des préférences

6. Tableau de bord (sans évaluation)

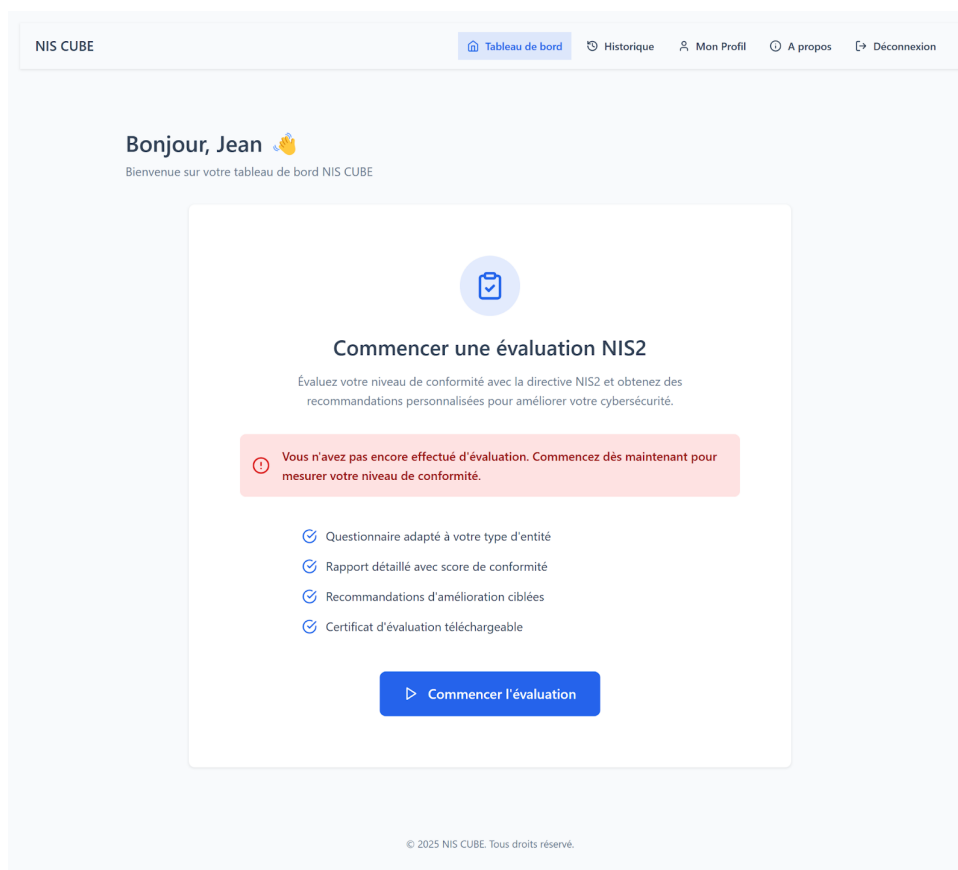


FIGURE 6 – Tableau de bord lorsqu'aucune évaluation n'a été réalisée

7. Tableau de bord (évaluation en cours)

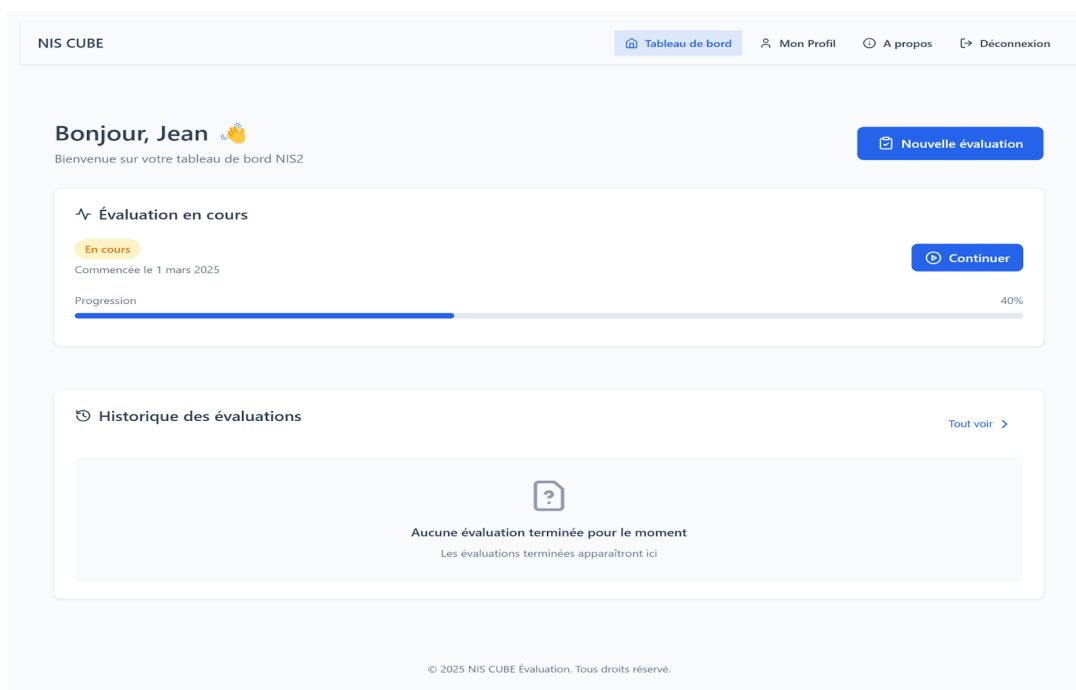


FIGURE 7 – Tableau de bord affichant l'avancement d'une évaluation

8. Tableau de bord (évaluation complète)

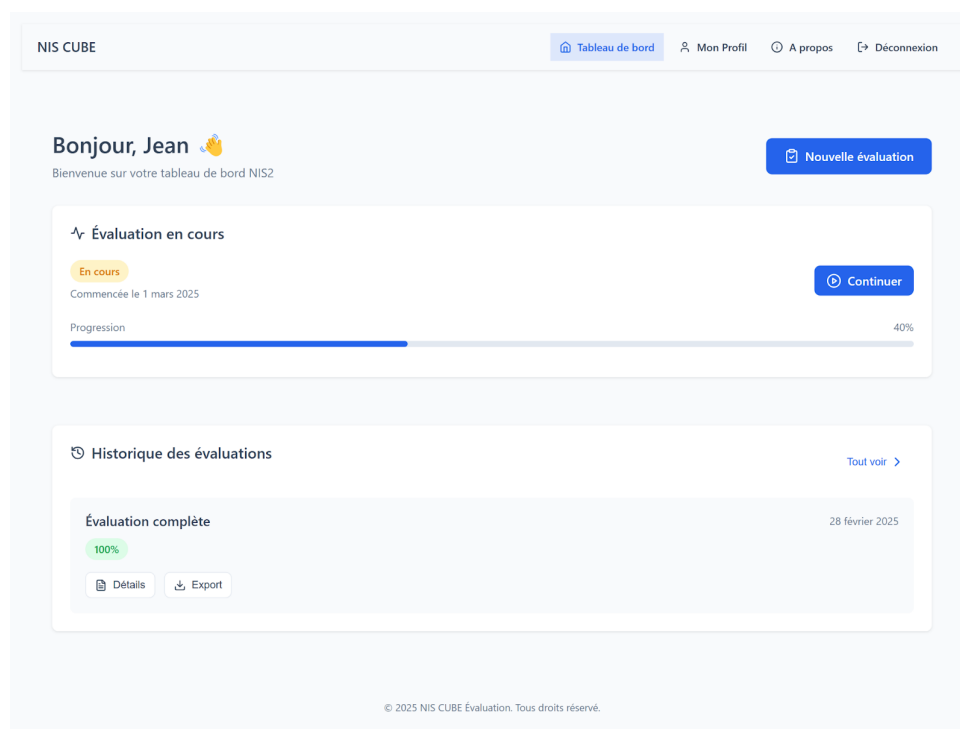


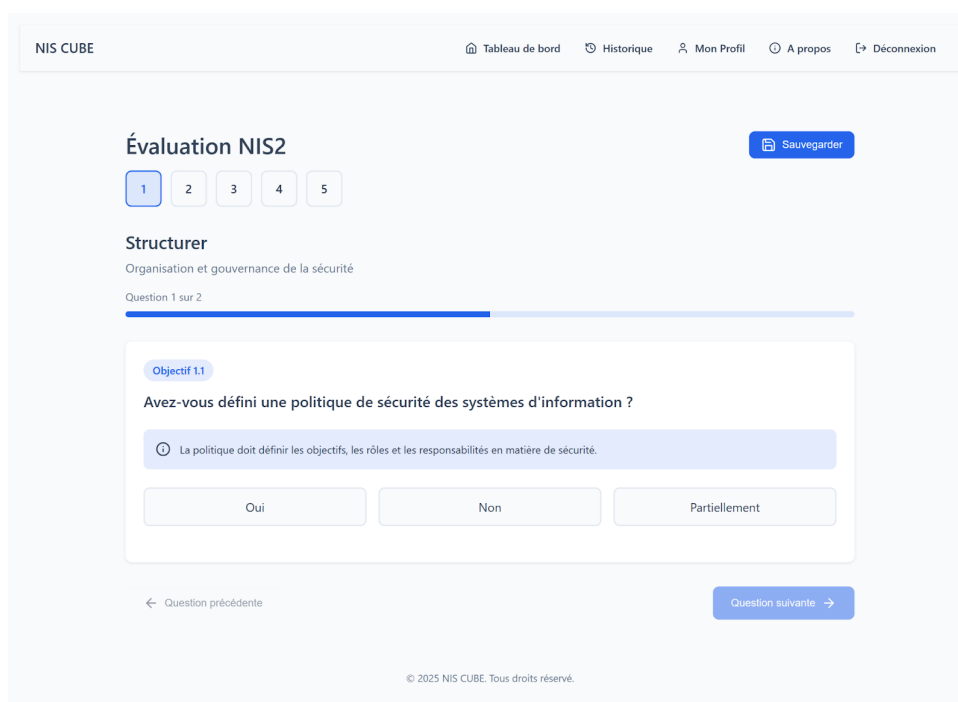
FIGURE 8 – Tableau de bord après l'achèvement d'une évaluation

9. Présentation de l'objectif avant évaluation



FIGURE 9 – Description de l'objectif avant de démarrer l'évaluation

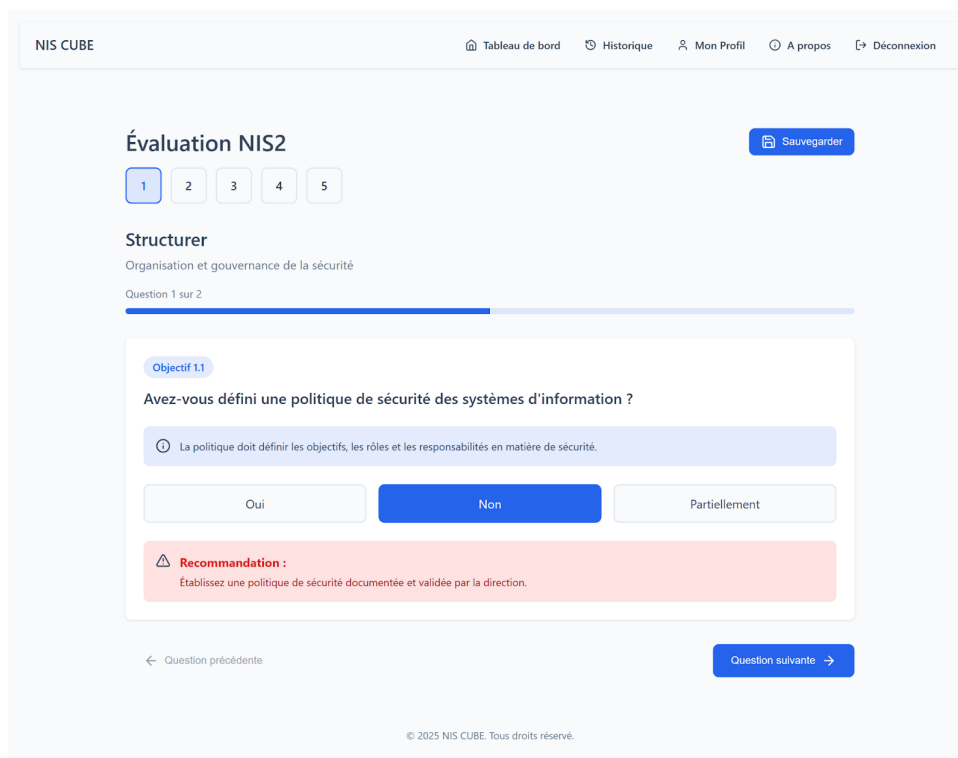
10. Durant l'évaluation



The screenshot shows the 'Évaluation NIS2' interface. At the top, there's a navigation bar with 'NIS CUBE' and links for 'Tableau de bord', 'Historique', 'Mon Profil', 'A propos', and 'Déconnexion'. Below this, the title 'Évaluation NIS2' is followed by a 'Sauvegarder' button. A progress bar shows five steps, with the first step '1' selected. The section is titled 'Structurer' with the subtitle 'Organisation et gouvernance de la sécurité'. It indicates 'Question 1 sur 2'. The main content area is titled 'Objectif 1.1' and asks 'Avez-vous défini une politique de sécurité des systèmes d'information ?'. Below the question, a text box contains the instruction: 'La politique doit définir les objectifs, les rôles et les responsabilités en matière de sécurité.' At the bottom, there are three buttons: 'Oui', 'Non', and 'Partiellement'. Navigation buttons 'Question précédente' and 'Question suivante' are at the bottom. A copyright notice '© 2025 NIS CUBE. Tous droits réservés.' is at the very bottom.

FIGURE 10 – Interface de question durant l'évaluation

11. Recommandations affichées durant l'évaluation



This screenshot is identical to the previous one, but with the 'Non' button selected. A new red box with a warning icon and the title 'Recommandation :' appears below the question. The text inside the box reads: 'Établissez une politique de sécurité documentée et validée par la direction.' The rest of the interface remains the same.

FIGURE 11 – Suggestions proposées en fonction des réponses de l'utilisateur

12. Affichage des résultats après l'évaluation d'un objectif

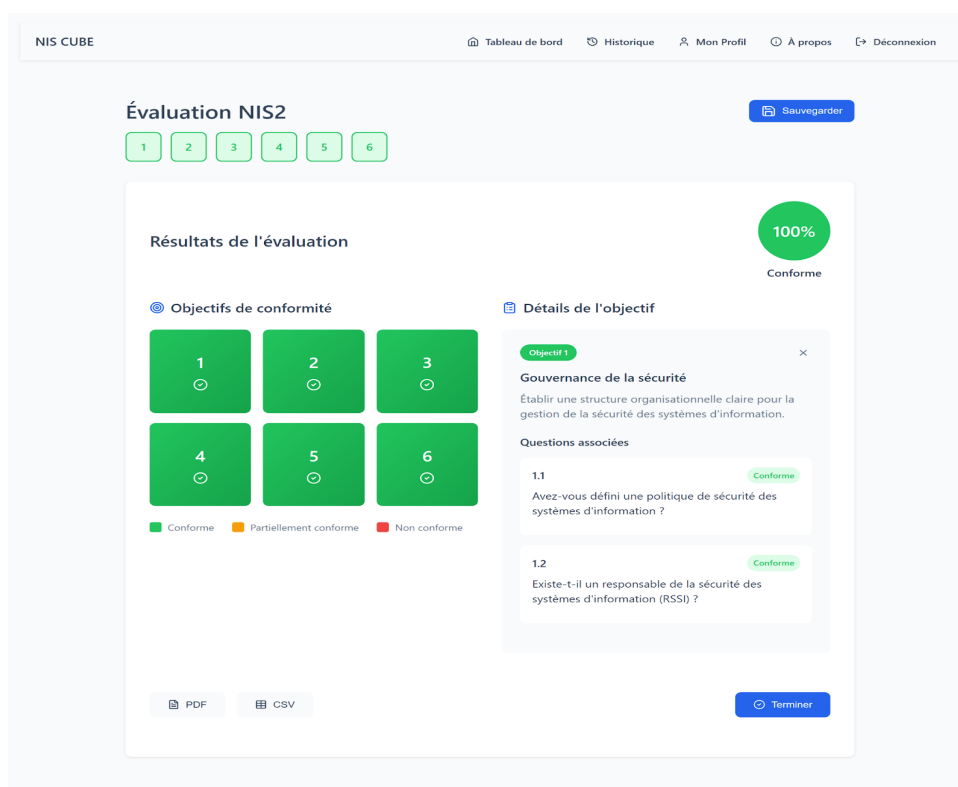


FIGURE 12 – Affichage des résultats détaillés après l'évaluation d'un objectif

13. Historique des évaluations

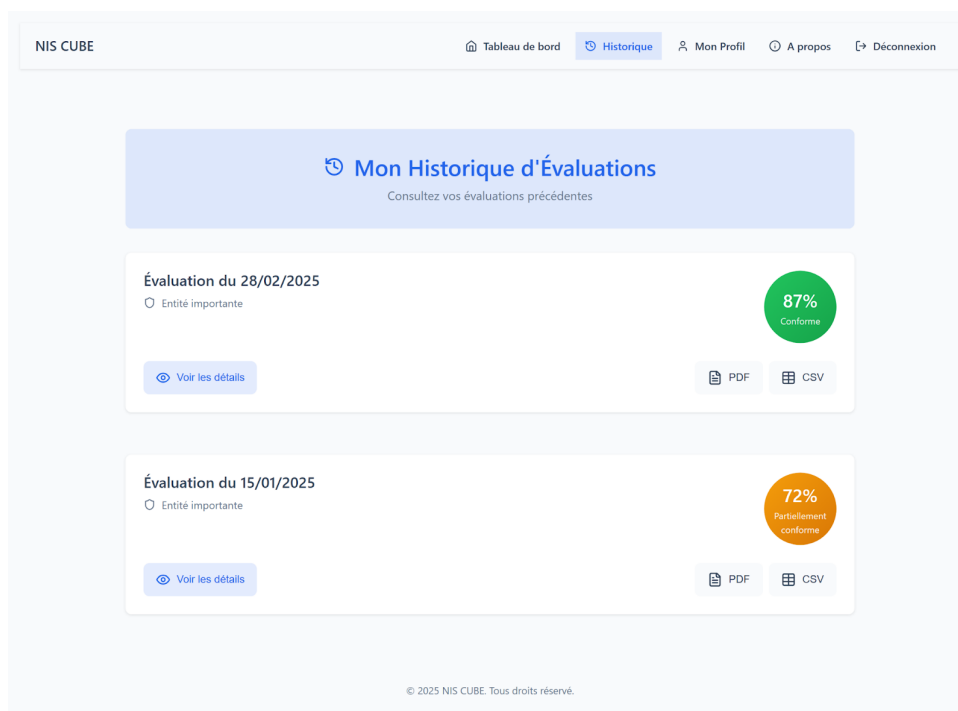


FIGURE 13 – Historique des évaluations effectuées par l'utilisateur

Ces maquettes, associées à la palette de couleurs, définissent l'aspect visuel et l'expérience utilisateur de NIS CUBE. Elles garantissent une cohérence graphique et ergonomique, tout en respectant les spécifications fonctionnelles décrites dans le Cahier des Charges.

2.2 Technologies utilisées

Dans cette section, nous détaillons les technologies utilisées pour le développement du projet NIS CUBE, allant du développement back-end et front-end à la gestion de la base de données, en passant par l'hébergement et le déploiement de l'application.

Développement du backend et frontend

Le choix des technologies pour le développement web repose sur plusieurs critères : simplicité d'utilisation, performance, scalabilité et large communauté de support. Pour assurer une cohérence technologique dans le projet, nous avons opté pour des frameworks JavaScript aussi bien pour le back-end que pour le front-end.

Node.js Node.js est un environnement d'exécution JavaScript côté serveur qui fonctionne sur tous les systèmes d'exploitation (Windows, Linux, macOS). Il permet d'exécuter du code JavaScript en dehors d'un navigateur, facilitant ainsi le développement en unifiant le langage entre le front-end et le back-end. Node.js est reconnu pour ses performances élevées et sa capacité à gérer un grand nombre de connexions simultanées grâce à son modèle basé sur des événements et une boucle d'événements non bloquante. L'accès à `npm` (Node Package Manager) permet l'intégration de nombreuses bibliothèques open source, accélérant ainsi le développement.

Express.js Express.js est un framework web minimaliste pour Node.js qui simplifie la gestion des routes et des requêtes HTTP. Il permet d'organiser efficacement les fonctionnalités de l'application et facilite la gestion des réponses dynamiques. En s'appuyant sur la puissance de Node.js, Express.js assure un développement rapide et structuré des API REST, essentielles pour la communication entre le front-end et le back-end.

Vue.js Vue.js est un framework JavaScript progressif utilisé pour le développement du front-end. Il est reconnu pour sa simplicité d'apprentissage et sa flexibilité. Contrairement à d'autres frameworks front-end nécessitant des prérequis comme TypeScript ou JSX, Vue.js est accessible à tous les développeurs ayant des bases en JavaScript. Il offre un rendu réactif et performant, optimisant l'expérience utilisateur. Son intégration avec des bibliothèques comme Vue Router et Vuex permet de développer des applications web modernes et dynamiques avec une gestion efficace de l'état.

Développement de la base de données

La gestion et le stockage des données dans NIS CUBE nécessitent un SGBD robuste et performant. PostgreSQL a été choisi pour ses nombreuses fonctionnalités et sa conformité aux standards SQL.

PostgreSQL PostgreSQL est un système de gestion de base de données relationnelles (SGBDR) open-source hautement avancé. Il est conçu pour offrir une intégrité des données optimale avec un support complet des transactions ACID (Atomicité, Cohérence, Isolation, Durabilité). PostgreSQL est compatible avec de nombreux systèmes d'exploitation et offre des fonctionnalités avancées telles que :

- Support des types de données JSON pour des structures de données flexibles.
- Indexation performante et optimisation des requêtes complexes.
- Sécurité renforcée avec gestion des rôles et authentification avancée.
- Extensibilité grâce à des modules et extensions tels que PostGIS pour la gestion des données géospatiales.

Grâce à ses performances et à sa flexibilité, PostgreSQL est le choix idéal pour gérer les données critiques du projet.

Serveur VPS et déploiement

L'hébergement et le déploiement de l'application sont des aspects essentiels garantissant la disponibilité et la sécurité du service. Nous avons opté pour un serveur VPS (Virtual Private Server) afin d'assurer un environnement stable et performant.

Serveur VPS Un serveur VPS est une solution d'hébergement qui allie la flexibilité d'un serveur dédié avec la rentabilité d'un hébergement mutualisé. Contrairement à l'hébergement mutualisé où les ressources sont partagées entre plusieurs utilisateurs, un VPS fournit un environnement isolé avec un accès root. Ce choix permet :

- Une meilleure personnalisation du serveur (installation de dépendances spécifiques).
- Une isolation des ressources garantissant des performances optimales.
- Une sécurité renforcée grâce à une configuration avancée du pare-feu et à l'installation de certificats SSL.

Déploiement Pour garantir une mise en production fiable et évolutive, nous déployons l'application sur un VPS configuré avec **Nginx** en tant que serveur web et reverse proxy. Nginx permet de gérer efficacement les requêtes entrantes, d'assurer une répartition de charge et de servir de pare-feu applicatif pour renforcer la sécurité. L'automatisation des déploiements est assurée via des scripts de déploiement permettant de mettre à jour l'application sans interruption majeure du service.

Le choix de ces technologies garantit un développement efficace, une maintenance simplifiée et une évolutivité permettant de répondre aux besoins futurs du projet NIS CUBE.

2.3 Base de données et modèle relationnel

Dans le cadre du projet NIS CUBE, nous avons d'abord élaboré un **Modèle Conceptuel de Données (MCD)**, puis nous en avons dérivé le **Modèle Logique de Données (MLD)**. Le MCD permet de représenter, de façon conceptuelle et indépendante des choix techniques, les entités du domaine, leurs attributs et leurs relations. Le MLD, quant à lui, formalise ces entités sous forme de tables et de clés, en vue d'une implémentation dans un SGBD.

Modèle Conceptuel de Données (MCD)

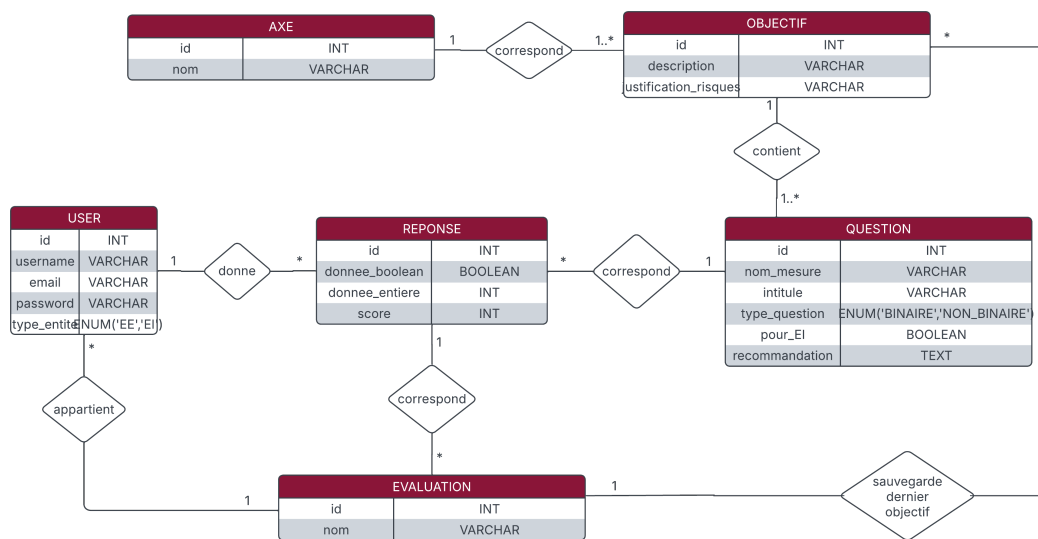


FIGURE 14 – Modèle Conceptuel de Données (MCD) de NIS CUBE

Comme le montre la Figure 14, nous distinguons les entités suivantes : AXE, OBJECTIF, QUESTION, USER, REPONSE et EVALUATION. Les cardinalités indiquent comment chaque entité est reliée aux autres (ex. un AXE peut contenir plusieurs OBJECTIF, un OBJECTIF peut être lié à plusieurs QUESTION, etc.). Cette représentation conceptuelle sert de base pour définir précisément les relations entre les données et leurs attributs.

Modèle Logique de Données (MLD)

À partir du MCD, nous avons établi le **Modèle Logique de Données (MLD)** en tenant compte des exigences du projet (types de données, clés primaires, contraintes, etc.). Ce MLD se concrétise sous la forme de cinq tables : AXE, OBJECTIF, QUESTION, USER, REPONSE et EVALUATION.

Cette section présente la structure relationnelle de la base de données pour le projet NIS CUBE. Chaque table est décrite ci-dessous avec ses champs, types, descriptions et contraintes.

Description détaillée des Tables

Table AXE

| Champ | Type | Description | Contraintes |
|--------|---------------------|----------------------------------------------|---------------|
| axe_id | INT (PK, AUTO-INCR) | Identifiant unique de l'axe. | Clé primaire. |
| nom | VARCHAR(...) | Nom de l'axe (ex. «Structurer», «Protéger»). | Obligatoire. |

Commentaires : Un AXE représente un ensemble d'objectifs NIS2. Chaque axe regroupe plusieurs objectifs.

Table OBJECTIF

| Champ | Type | Description | Contraintes |
|-----------------------|----------------------|--------------------------------------------------------------------------------|---------------------------------------------------------------|
| objectif_id | INT (AUTO-INCR) | Identifiant unique de l'objectif. | Partie de la clé primaire. |
| axe_id | INT (FK) | Référence à l'axe associé. | Partie de la clé primaire et doit correspondre à AXE(axe_id). |
| description | TEXT ou VARCHAR(...) | Description de l'objectif (ex. «L'entité recense ses systèmes d'information»). | Obligatoire. |
| justification_risques | TEXT ou VARCHAR(...) | Justification et risques associés si l'objectif n'est pas atteint. | Optionnel. |

Commentaires : La table OBJECTIF appartient à un AXE. La clé primaire est composite et se compose de (objectif_id, axe_id).

Table USER

| Champ | Type | Description | Contraintes |
|-------------|---------------------|--------------------------------------------------------------------|-----------------------------------|
| user_id | INT (PK, AUTO-INCR) | Identifiant unique de l'utilisateur. | Clé primaire. |
| username | VARCHAR(...) | Nom d'utilisateur. | Unique et obligatoire. |
| email | VARCHAR(...) | Adresse e-mail. | Unique et obligatoire (pour MFA). |
| password | VARCHAR(...) | Mot de passe haché. | Jamais stocké en clair. |
| type_entite | ENUM('EE','EI') | Type d'entité : Entité Essentielle (EE) ou Entité Importante (EI). | Obligatoire. |

Commentaires : Chaque utilisateur représente une entité. Le champ email est utilisé pour l'authentification et l'envoi de codes MFA.

Table EVALUATION

| Champ | Type | Description | Contraintes |
|---------------------|-----------------|---------------------------------------------------------------|---------------------------------------------------------------------|
| evaluation_id | INT (AUTO-INCR) | Identifiant unique de l'évaluation. | Partie de la clé primaire. |
| nom | VARCHAR(...) | Nom ou titre donné à l'évaluation par l'utilisateur. | Obligatoire. |
| user_id | INT | Référence à l'utilisateur ayant initié cette évaluation. | Partie de la clé primaire et clé étrangère vers USER ; Obligatoire. |
| id_dernier_objectif | INT (nullable) | Référence à l'objectif sur lequel l'utilisateur s'est arrêté. | Clé étrangère vers OBJECTIF ; Optionnel (nullable). |

Commentaires : Chaque évaluation est initiée par un utilisateur (user_id) et peut être nommée pour permettre une meilleure organisation des sessions d'évaluation.

La clé primaire est composée de evaluation_id et user_id, garantissant qu'un utilisateur peut avoir plusieurs évaluations distinctes.

Le champ id_dernier_objectif permet à l'utilisateur de reprendre son évaluation là où il s'était arrêté. Il référence un objectif spécifique dans la table OBJECTIF. S'il est NULL, cela signifie que l'évaluation n'a pas encore été commencée ou qu'elle a été finalisée.

Table QUESTION

| Champ | Type | Description | Contraintes |
|----------------|-------------------------------|---------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------|
| question_id | INT (AUTO-INCR) | Identifiant unique de la question. | Partie de la clé primaire. |
| axe_id | INT | Référence à l'axe de l'objectif associé. | Partie de la clé étrangère vers OBJECTIF ; Obligatoire. |
| objectif_id | INT | Référence à l'objectif associé. | Partie de la clé primaire et de la clé étrangère vers OBJECTIF ; Obligatoire. |
| intitule | TEXT ou VARCHAR(...) | Texte de la question. | Obligatoire. |
| nom_mesure | VARCHAR(...) | Nom de la mesure associée, si la question est liée à une mesure de conformité. | Optionnel (nullable). |
| type_question | ENUM('BINAIRE','NON_BINAIRE') | Indique si la réponse attendue est binaire (oui/non) ou un score sur une échelle de 0 à 10. | Obligatoire. |
| pour_EI | BOOLEAN | Indique si la question s'applique aux Entités Importantes. | Obligatoire. |
| recommandation | TEXT (nullable) | Texte de recommandation affiché à l'utilisateur en fonction de sa réponse, si applicable. | Optionnel (nullable). |

Commentaires : Chaque question dépend d'un OBJECTIF (défini par le couple (axe_id, objectif_id)). La clé primaire de la table QUESTION est composite et se compose de (question_id, axe_id, objectif_id).

Le champ nom_mesure est renseigné si la question relève d'une mesure spécifique ; sinon, il reste NULL.

Le champ recommandation permet d'afficher un conseil personnalisé à l'utilisateur en fonction de sa réponse, notamment si la conformité n'est pas atteinte. Ce champ reste NULL si aucune recommandation spécifique n'est nécessaire.

Table REPONSE

| Champ | Type | Description | Contraintes |
|----------------|-----------------|---------------------------------------------------------------------------------|---------------------------------------------------------------------------|
| reponse_id | INT (AUTO-INCR) | Identifiant unique de la réponse. | Partie de la clé primaire. |
| evaluation_id | INT (FK) | Référence à l'évaluation dans laquelle cette réponse a été donnée. | Partie de la clé primaire, doit correspondre à EVALUATION(evaluation_id). |
| user_id | INT (FK) | Référence à l'utilisateur qui a répondu. | Partie de la clé primaire, doit correspondre à USER(user_id). |
| question_id | INT (FK) | Référence à la question concernée. | Partie de la clé primaire, doit correspondre à QUESTION(question_id). |
| score | INT | Score attribué à la réponse. | Valeur comprise entre 0 et 10. |
| donnee_boolean | BOOLEAN | Valeur de réponse pour une question de type binaire. | Nullable. |
| donnee_entiere | INT | Valeur numérique saisie pour une question non binaire (comprise entre 0 et 10). | Nullable. |

Commentaires : La table REPONSE enregistre les réponses (ou scores) des utilisateurs pour chaque question dans une évaluation donnée. La clé primaire est **composite** et se compose de (reponse_id, evaluation_id, user_id, question_id). Cela permet de garantir qu'une même question peut être répondue plusieurs fois par un même utilisateur, mais dans des évaluations différentes.

Outre le champ score (compris entre 0 et 10), deux colonnes supplémentaires permettent de stocker la donnée réelle en fonction du type de question :

- donnee_boolean pour une réponse binaire,
- donnee_entiere pour une réponse numérique.

Ainsi, la base de données permet de lier les utilisateurs, leurs réponses, les questions, les évaluations, les objectifs et les axes. Ce modèle relationnel offre la flexibilité nécessaire pour gérer les différentes configurations de questions (liées directement à un objectif ou accompagnées d'un nom de mesure dans le cas de mesures spécifiques) et pour distinguer les entités essentielles (EE) des entités importantes (EI).

2.4 Diagramme de classes

Le diagramme de classes ci-dessous (Figure 15) illustre l'architecture objet du projet **NIS CUBE**. Il met en avant les entités principales (reflétant la base de données) ainsi que des classes de services, chargées de la logique métier (scoring, recommandations, génération de rapports, etc.). Nous décrivons ici les classes et leurs méthodes associées aux exigences fonctionnelles (F1 à F9).

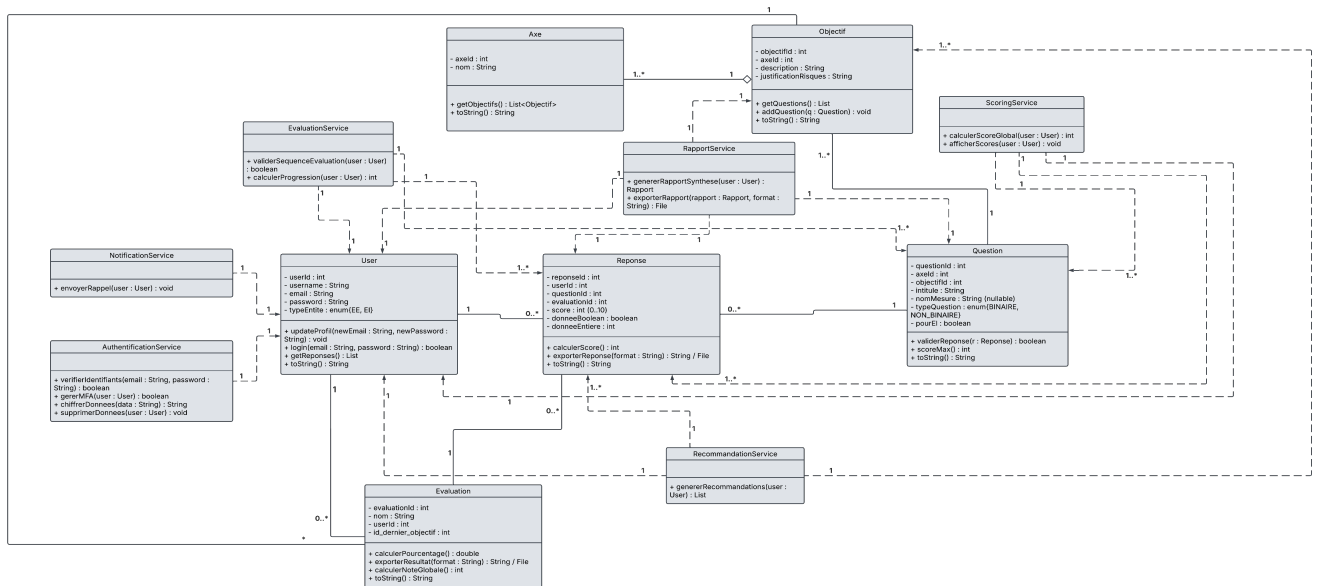


FIGURE 15 – Diagramme de classes du projet NIS CUBE

Classes de domaine

Classe Axe

- **Attributs :**
 - - axeId : int
 - - nom : String
- **Méthodes :**
 - + getObjectifs() : List<Objectif> (Récupération des objectifs liés à l'axe.)
 - + toString() : String

Classe Objectif

- **Attributs :**
 - - objectifId : int
 - - axeId : int
 - - description : String
 - - justificationRisques : String

- **Méthodes :**
 - + getQuestions() : List<Question>
 - + addQuestion(q : Question) : void
 - + toString() : String

Classe Question

- **Attributs :**
 - - questionId : int
 - - axeId : int
 - - objectifId : int
 - - intitule : String
 - - nomMesure : String (nullable)
 - - typeQuestion : enum{BINAIRE, NON_BINAIRE}
 - - pourEI : boolean
- **Méthodes :**
 - + validerReponse(r : Reponse) : boolean (*Contrôle la cohérence d'une réponse par rapport au type de question.*)
 - + scoreMax() : int (*Retourne la note maximale : 10 pour NON_BINAIRE, 0 ou 10 pour BINAIRE.*)
 - + toString() : String

Classe User

- **Attributs :**
 - - userId : int
 - - username : String
 - - email : String
 - - password : String (haché)
 - - typeEntite : enum{EE, EI}
- **Méthodes :**
 - + updateProfil(newEmail : String, newPassword : String) : void (*F3.2 — Mise à jour du profil.*)
 - + login(email : String, password : String) : boolean (*F2 — Connexion, potentiellement couplée à MFA.*)
 - + getReponses() : List<Reponse> (*F7.1 — Historique des évaluations.*)
 - + toString() : String

Classe Reponse

- **Attributs :**
 - - reponseId : int
 - - userId : int
 - - questionId : int
 - - evaluationId : int
 - - score : int (0..10)
 - - donneeBoolean : boolean (nullable)
 - - donneeEntiere : int (nullable)

- **Méthodes :**

- + calculerScore() : int (*F4.2 — Analyse et scoring : calcule le score effectif.*)
- + exporterReponse(format : String) : String / File (*F8 — Export des résultats, ex. PDF ou CSV.*)
- + toString() : String

Classe Evaluation

- **Attributs :**

- - evaluationId : int
- - nom : String
- - userId : int
- - id_dernier_objectif : int

- **Méthodes :**

- + calculerPourcentage() : double (*Calcule le pourcentage de progression de l'évaluation.*)
- + exporterResultat(format : String) : String / File (*F8 — Génère un fichier PDF ou CSV contenant les résultats de l'évaluation.*)
- + calculerNoteGlobale() : int (*Retourne la note globale obtenue sur l'ensemble des questions.*)
- + toString() : String

Classes de services (logique métier)

Pour répondre aux exigences fonctionnelles (F1 à F9), certaines méthodes transverses sont regroupées dans des classes de service.

Classe EvaluationService

- **Responsabilité :** gérer la logique d'évaluation, l'enchaînement des questions et la progression (F5).

- **Méthodes :**

- + validerSequenceEvaluation(user : User) : boolean (*F5.1 — Vérifie que l'utilisateur a répondu aux questions d'une section avant de passer à la suivante.*)
- + calculerProgression(user : User) : int (*F5.2 — Indicateur de progression en % du questionnaire global.*)

Classe ScoringService

- **Responsabilité :** centraliser le calcul des scores (F4.2) et l'affichage final (F6).

- **Méthodes :**

- + calculerScoreGlobal(user : User) : int (*Parcourt toutes les Réponse de l'utilisateur, calcule le score global.*)
- + afficherScores(user : User) : void (*F6 — Gère la présentation des scores, ex. sous forme de graphiques ou d'indicateurs.*)

Classe RecommendationService

- **Responsabilité :** générer des recommandations en fonction des réponses (F4.3).

- **Méthodes :**

- + genererRecommandations(user : User) : List<String> (*Analyse les Réponse et propose des conseils personnalisés.*)

Classe RapportService

- **Responsabilité** : produire des rapports de synthèse (F4.4).
- **Méthodes** :
 - + genererRapportSynthese(user : User) : Rapport (*Contient le score global, la liste des objectifs atteints, etc.*)
 - + exporterRapport(rapport : Rapport, format : String) : File (*F8 — Export du rapport en PDF/CSV.*)

Classe NotificationService

- **Responsabilité** : gérer les rappels et alertes automatiques (F7.2).
- **Méthodes** :
 - + envoyerRappel(user : User) : void (*Envoie un e-mail/notification pour rappeler à l'utilisateur de mettre à jour son évaluation.*)

Classe AuthentificationService

- **Responsabilité** : assurer la sécurité et la gestion des connexions (F2, F9).
- **Méthodes** :
 - + verifierIdentifiants(email : String, password : String) : boolean (*Contrôle l'authentification de base.*)
 - + gererMFA(user : User) : boolean (*Envoie un code MFA, le vérifie, etc.*)
 - + chiffrerDonnees(data : String) : String (*F9 — Sécurité et gestion des données, ex. cryptage ou hashing.*)
 - + supprimerDonnees(user : User) : void (*F9.2 — Suppression des données personnelles.*)

Lien avec les exigences fonctionnelles (F1 à F9)

- **F1 (Page d'accueil)** : se traduit essentiellement par une vue front-end, sans méthode directe dans les classes de domaine.
- **F2 (Inscription et Connexion)** : User.login() et AuthentificationService.verifierIdentifiants() / gererMFA().
- **F3.2 (Mise à jour du profil)** : User.updateProfil(...).
- **F4.1 (Agrégation des questions)** : Objectif.getQuestions() ou Objectif.addQuestion(...) + EvaluationService.
- **F4.2 (Analyse et scoring)** : Reponse.calculerScore() et ScoringService.calculerScoreGlobal().
- **F4.3 (Génération de recommandations)** : RecommandationService.genererRecommandations().
- **F4.4 (Rapports de synthèse)** : RapportService.genererRapportSynthese(...).
- **F5.1 (Séquence et validation)** : EvaluationService.validerSequenceEvaluation(...).
- **F5.2 (Indicateur de progression)** : EvaluationService.calculerProgression(...).
- **F6 (Affichage des scores)** : ScoringService.afficherScores(...) ou inclus dans RapportService.
- **F7.1 (Historique des évaluations)** : User.getReponses() (retourne l'historique), + vue front-end.
- **F7.2 (Rappels automatiques)** : NotificationService.envoyerRappel(...).
- **F8 (Export des résultats)** : Reponse.exporterReponse(...) ou RapportService.exporterRapport(...).
- **F9 (Sécurité et gestion des données)** : AuthentificationService.chiffrerDonnees(...) et AuthentificationService.supprimerDonnees(...).

Le diagramme de classes décrit ainsi :

- Les **classes de domaine** : Axe, Objectif, Question, User, Reponse, correspondant aux tables principales de la base de données.
- Les **classes de service** : EvaluationService, ScoringService, RecommandationService, RapportService, NotificationService et AuthentificationService, responsables de la logique métier (scoring, rappels, export, sécurité, etc.).

Chaque exigence fonctionnelle (F1 à F9) est couverte par au moins une méthode dans ces classes ou services, garantissant une architecture modulaire et évolutive. Les méthodes associées aux entités (création, lecture, mise à jour, suppression) et aux services (scoring, recommandation, rapport, notification) assurent la cohérence entre les besoins exprimés dans le Cahier des Charges et la mise en œuvre technique.

2.5 Diagramme de Gantt

Le diagramme de Gantt ci-dessous illustre la planification détaillée du projet NIS CUBE. Il présente les différentes tâches nécessaires à la mise en place du projet, leur enchaînement chronologique ainsi que leur durée.

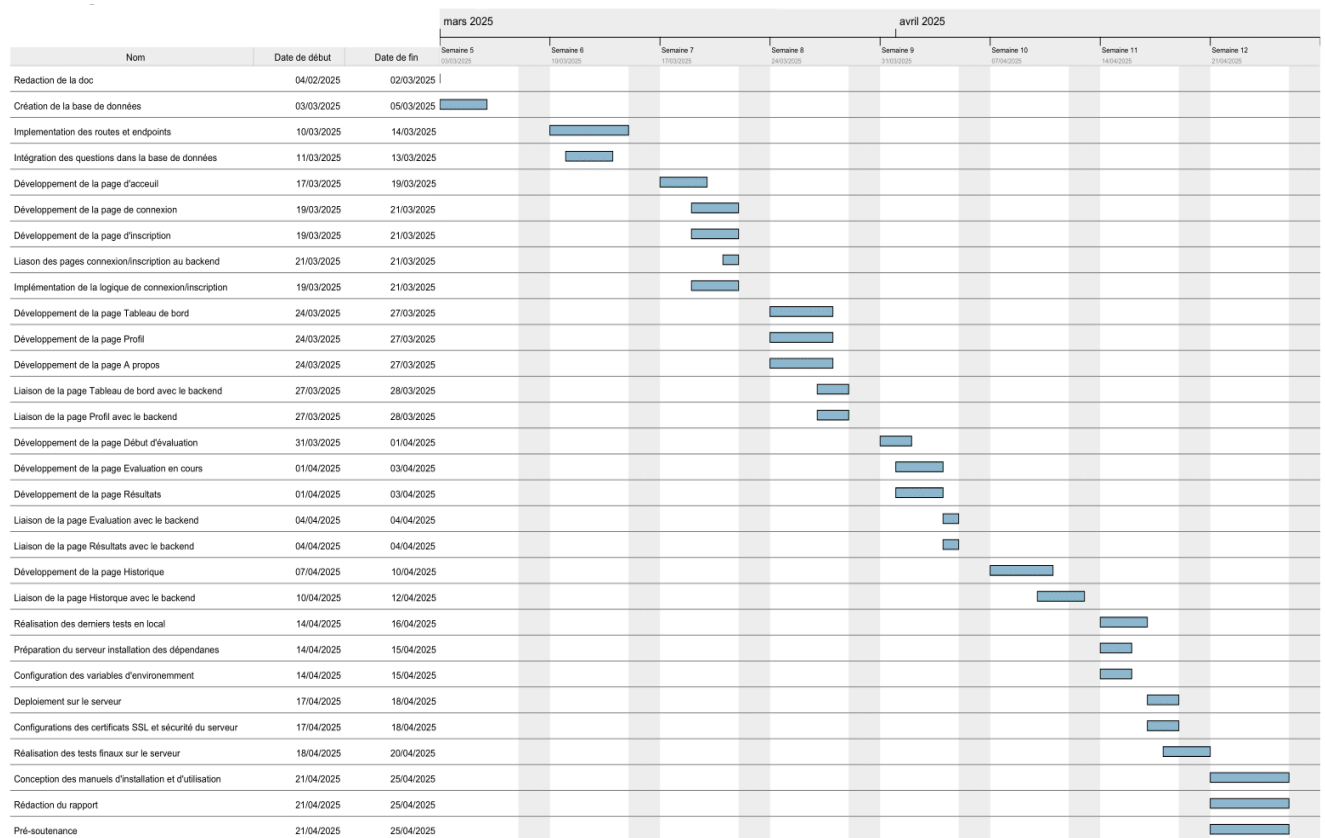


FIGURE 16 – Diagramme de Gantt du projet NIS CUBE

Analyse du planning

Le diagramme de Gantt se divise en plusieurs phases essentielles :

- **Phase 1 : Conception et mise en place de la base de données** (03/03/2025 - 11/03/2025)
 - Création de la base de données.
 - Intégration des questions dans la base de données.
 - Mise en place des routes et endpoints pour l'API.
- **Phase 2 : Développement des interfaces utilisateur** (17/03/2025 - 10/04/2025)
 - Développement des pages principales : accueil, connexion, inscription, tableau de bord, profil, évaluation, résultats, historique.
 - Intégration des pages front-end avec le backend.
- **Phase 3 : Tests et finalisation du développement** (14/04/2025 - 20/04/2025)
 - Tests locaux des fonctionnalités.
 - Préparation du serveur et configuration des dépendances.
 - Déploiement et sécurisation du serveur.
- **Phase 4 : Documentation et soutien** (21/04/2025 - 25/04/2025)
 - Rédaction des manuels d'utilisation et du rapport final.
 - Préparation à la soutenance.

Synthèse

Le diagramme de Gantt met en évidence la gestion efficace du temps pour assurer le bon déroulement du projet. Chaque tâche est planifiée de manière à garantir une progression fluide du développement à la livraison finale.

L'organisation en plusieurs phases successives permet une meilleure gestion des risques et assure la validation des différentes fonctionnalités avant leur mise en production.

3 Conclusion

Le projet **NIS CUBE** a été conçu dans l'objectif de fournir un outil efficace et intuitif d'auto-évaluation de la conformité à la directive **NIS2**. Tout au long de ce document, nous avons détaillé les aspects techniques et fonctionnels du projet, depuis la modélisation des données jusqu'à l'implémentation des interfaces utilisateur et la planification du développement.

Grâce à une architecture modulaire et une conception centrée sur l'utilisateur, le projet assure :

- Une gestion claire et efficace des évaluations et des réponses.
- Une interface ergonomique facilitant la navigation et l'accès aux résultats.
- Une intégration fluide entre le frontend et le backend.
- Une sécurité renforcée avec la gestion des comptes et des données utilisateurs.

Le diagramme de Gantt présenté dans la section précédente a permis d'assurer un suivi rigoureux du développement, garantissant ainsi une mise en œuvre dans les délais impartis. De plus, les choix technologiques effectués permettent d'assurer la maintenabilité et l'évolutivité du projet.