Ecole Nationale
Supérieure
d'Ingénieurs
de Tunis

الجمهورية التونسية
وزارة التعليم العالي والبحث العلمي
جامعة تونس
المدرسة الوطنية العليا للمهندسين بتونس

Réf : PFA2-2025-01

Rapport de Projet de Fin d'Année

de

Deuxième année en Génie Informatique

Présenté et soutenu publiquement le 07/05/2025

Par

Malouche Med Rayen, Tmimi Ines & Rachdi Med Amine

Outil de Gestion de Contenu

Composition du jury

Monsieur Boulares Mehrez	**Président**
Madame Ben Younes Ahlem	**Encadrante**
Monsieur Essid Nizar	**Encadrant**

Année universitaire : 2024-2025

Dédicaces

Nous dédions ce travail à nos familles respectives, qui nous ont toujours soutenus avec amour, patience et encouragements. Leur présence et leur confiance ont été une source inestimable de motivation tout au long de ce parcours.

À nos parents, pour les valeurs qu'ils nous ont transmises, leur foi en notre potentiel et leur appui sans faille.

À nos enseignants, qui ont su éveiller notre curiosité, renforcer notre rigueur et nous inspirer à donner le meilleur de nous-mêmes.

À nos camarades et amis, pour leur soutien moral, les échanges constructifs, et les moments de complicité qui ont rendu cette expérience encore plus enrichissante.

Enfin, à toutes les personnes qui, de près ou de loin, ont contribué à la réalisation de ce projet.

Remerciements

Nos remerciements les plus sincères vont à notre chère encadrante, Madame **Ahlem BEN YOUNES**, qui nous a accompagnés tout au long de ce travail. Nous la remercions non seulement pour le temps et les efforts qu'elle nous a consacrés lors de la réalisation de ce projet, mais aussi pour sa bienveillance, sa rigueur et sa volonté constante de nous guider vers l'excellence.

Nous tenons également à exprimer toute notre reconnaissance à Monsieur **Nizar ESSID**, encadrant technique de ce projet, pour son accompagnement précieux, ses conseils techniques avisés et sa disponibilité tout au long de cette aventure.

Nous remercions aussi chaleureusement les membres du jury pour nous avoir honorés en acceptant d'évaluer ce travail.

Enfin, nous exprimons notre gratitude à toutes les personnes ayant contribué, de près ou de loin, au bon déroulement de ce projet.

Table des matières

Table des figures

Table des tableaux

Introduction Générale

À l'ère du numérique, le développement d'applications web joue un rôle important dans la stratégie de digitalisation des entreprises, des institutions publiques et des acteurs du secteur associatif. Que ce soit pour automatiser des processus internes, offrir des services en ligne, ou renforcer la présence digitale, les applications web sont devenues des leviers incontournables de compétitivité et d'innovation.

Dans un contexte où la demande pour des solutions digitales personnalisées explose, la création de sites web dynamiques et modulables ne peut plus être l'apanage exclusif des développeurs professionnels. Le besoin de flexibilité, de rapidité de mise en œuvre et de réduction des coûts pousse à repenser les méthodes de développement traditionnelles. C'est précisément dans cette dynamique que s'inscrit notre projet.

Nous proposons une **plateforme de création d'applications web dynamiques**, pensée pour être à la fois **innovante, intuitive et accessible à tous**, y compris aux non-techniciens. Cette solution repose sur un **système de glisser-déposer (drag and drop)** qui permet à l'utilisateur de concevoir, structurer et personnaliser l'interface de son application sans écrire une seule ligne de code. Elle intègre également un moteur de gestion de données sur mesure, permettant de créer et manipuler une base de données personnalisée, de gérer dynamiquement les contenus, et de construire des parcours utilisateurs adaptés à des cas d'usage variés.

Cette approche vise à **démocratiser la création web**, en combinant **simplicité d'usage**, **flexibilité fonctionnelle** et **puissance technique**. Elle répond aussi aux limites identifiées dans les plateformes no-code existantes, notamment en matière de personnalisation avancée, de gestion multi-modules, ou encore d'intégration sécurisée de données et d'APIs.

Ce rapport a pour objectif de présenter en détail les fondements, les choix technologiques, ainsi que les étapes de conception et de développement de cette plateforme. Il est structuré autour des chapitres suivants :

Le premier chapitre, intitulé « Cadre général du projet », présente le contexte général du projet et une critique des solutions existantes ainsi que la solution proposée.

Le deuxième chapitre, « Analyse et spécification des besoin », est consacré à l'identification des besoins fonctionnels et non fonctionnels attendus de l'application et à l'analyse des différents cas d'utilisation.

Le troisième chapitre, « Etude conceptuelle » est dédié à la présentation des différents modèles conceptuels qui préparent le projet pour son implémentation.

Le quatrième chapitre, « Réalisation et mise en œuvre », présente l'environnement et les outils de développement utilisés et illustre les interfaces les plus pertinentes. Finalement, ce rapport est clôturé par une « Conclusion générale » pour récapituler le travail réalisé et donner quelques perspectives.

Chapitre 1 : Cadre général du projet

Introduction

L'objet de ce chapitre est de présenter le contexte général de notre projet, à savoir le développement d'une plateforme de création de sites web dynamiques sans code. Nous commencerons par analyser les solutions existantes dans le marché des constructeurs web visuels, en identifiant les principales limitations de ces derniers. Nous conclurons par une description du travail demandé dans le cadre de ce projet.

1. Contexte du projet

Aujourd'hui, créer un site web dynamique est souvent indispensable pour gérer un projet, présenter des services ou organiser du contenu en ligne. Pourtant, cela reste une tâche complexe qui demande du temps, des compétences techniques et souvent un budget important.

Notre projet est conçu pour simplifier ce processus. Il permet de créer des sites web dynamiques sans écrire une seule ligne de code. Grâce à une interface claire, intuitive, et basée sur le principe du drag and drop, les utilisateurs peuvent facilement structurer leurs pages, définir leur propre base de données, ajouter du contenu, et construire un site entièrement personnalisé.

2. Étude de l'existant

Cette section a pour objectif d'analyser le marché des plateformes de création de sites web dynamiques, en Tunisie et à l'international, afin de mieux comprendre ce qui est proposé en termes d'outils no-code et low-code. Cela nous permettra d'identifier les solutions existantes, leurs principales fonctionnalités, leurs limites, et ainsi en déduire les éléments clés à intégrer dans notre propre projet.

2.1. Description de l'univers

À la suite d'une recherche approfondie sur le web, nous avons pu identifier les principales plateformes de création de sites dynamiques, en Tunisie et à l'étranger. Celles-ci sont présentées dans le tableau 1.

Tableau 1 : Tableau descriptif de l'univers

Nom	Logo	Limites
Wordpress : WordPress est un système de gestion de contenu open-source utilisé pour créer une grande variété de sites web, allant des blogs personnels aux sites d'entreprise complexes.		**Complexité Technique** : La configuration avancée peut nécessiter des compétences techniques, ce qui peut être un obstacle pour les utilisateurs sans expérience en développement web. **Maintenance** : Nécessite une gestion régulière des mises à jour et de la sécurité, ce qui peut être chronophage. **Gestion des Sous-Plateformes** : La création et la gestion de sous-plateformes dynamiques ne sont pas natives et requièrent des configurations supplémentaires
WYSIWYG Web Builder : WYSIWYG Web Builder est un outil complet permettant de créer des sites web via une interface visuelle, sans recourir à du codage manuel.		**Personnalisation Avancée :** Malgré ses nombreuses fonctionnalités, il reste limité pour des projets nécessitant une logique métier complexe. **IA Restreinte :** Les outils d'IA intégrés se concentrent principalement sur la création et la modification d'images, et n'intègrent pas une analyse ou une automatisation avancée des données.

Adalo : Adalo est une plateforme glisser-déposer conçue pour créer des applications mobiles et web sans codage.		**Échelle des Projets :** Plus adapté aux petites applications ; les grandes applications avec des bases de données complexes peuvent rencontrer des limitations. **Flexibilité :** Moins flexible pour des personnalisations poussées ou des intégrations techniques spécifiques.
Glide : Glide transforme des feuilles de calcul en applications puissantes via un éditeur visuel intuitif.		**Dépendance aux Données :** Limité à des sources de données spécifiques comme Google Sheets, ce qui peut restreindre la portée des projets. **Complexité des Applications :** Les applications complexes nécessitant plusieurs sources de données ou une logique avancée sont difficiles à gérer avec Glide.
Wix : Wix est un créateur de sites web populaire offrant une interface glisser-déposer intuitive.		**Personnalisation Restreinte :** Bien que simple à utiliser, Wix est limité pour les designs et fonctionnalités très spécifiques. **Gestion de Projets Dynamiques :** Ne supporte pas nativement la gestion dynamique des sous-plateformes ou des tableaux de bord complexes.

Webflow : Webflow est une plateforme avancée de création de sites web offrant un éditeur visuel puissant destiné aux designers et développeurs.		**Courbe d'Apprentissage Élevée :** Malgré son interface visuelle, Webflow exige une compréhension des concepts techniques comme le box model et le CSS pour tirer pleinement parti de ses fonctionnalités. **Dépendance à l'Écosystème :** Les sites créés sur Webflow sont hébergés sur leurs serveurs, ce qui limite les options de migration et le contrôle des données. **Les plans de Webflow peuvent être coûteux**, en particulier pour -Les fonctionnalités avancées comme l'e-commerce. -L'hébergement et la gestion de multiples projets. Les outils de collaboration ne sont pas aussi avancés que sur des plateformes comme Figma pour les équipes travaillant sur un projet simultanément.
SquareSpace : Squarespace est une plateforme de création de sites web tout-en-un, largement utilisée pour les portfolios et les petites boutiques en ligne. Elle est reconnue pour son design élégant et ses modèles prédéfinis.		**Flexibilité limitée pour les designs sur mesure :** Bien que les modèles soient visuellement attrayants, la personnalisation avancée est limitée sans recourir à du code (CSS ou JavaScript). Ainsi, Les utilisateurs ne peuvent pas modifier profondément la structure ou le comportement des templates.

		Fonctionnalités restreintes pour les projets complexes : Squarespace est principalement conçu pour des sites simples et n'est pas adapté pour des applications web interactives, des tableaux de bord dynamiques ou des systèmes de gestion complexes. **Coût relativement élevé :** Les abonnements sont coûteux comparés à ce que la plateforme offre en termes de flexibilité et de personnalisation. (Les fonctionnalités e-commerce et les outils avancés nécessitent des plans premium.) **Gestion de données limitée :** Les options pour intégrer des bases de données complexes ou des systèmes tiers sont restreintes ce qui limite la plateforme pour des projets nécessitant une forte connectivité ou une gestion avancée des données.

Tilda : est une plateforme de création de sites web qui se distingue par son approche axée sur le design. Elle permet de créer des sites web visuellement attractifs en utilisant un système de blocs prédéfinis.		**Flexibilité limitée pour les projets complexes** : Tilda convient bien pour des sites simples et visuellement soignés, mais elle montre vite ses limites lorsqu'il s'agit de créer des fonctionnalités avancées ou de gérer des bases de données dynamiques. La création de sous-plateformes ou de tableaux de bord interactifs n'est pas possible. **Personnalisation avancée restreinte** : Malgré la diversité des blocs proposés, les options de personnalisation restent limitées. Pour aller plus loin, il faut souvent écrire du code HTML, CSS ou JavaScript, ce qui contredit l'approche no-code de la plateforme. **Peu adaptée aux applications dynamiques** : Tilda est pensée pour des sites statiques ou des pages de présentation. Elle ne permet pas de gérer un contenu dynamique ni d'interagir de manière complexe avec les utilisateurs, ce qui la rend inadaptée pour des projets nécessitant un backend solide

		ou une gestion poussée des données.

2.2. Critique de l'existant

Le marché des plateformes visuelles (glisser-déposer, IA) est bien développé, mais montre vite ses limites dès qu'on sort des projets simples. Voici les principaux points :

- **Pas de gestion claire des sous-plateformes** : Impossible de créer facilement plusieurs interfaces ou modules séparés sans passer par des réglages complexes.
- **Personnalisation limitée** : Pour des designs ou fonctions avancées, on atteint vite les limites. Il faut alors coder, ce qui casse l'esprit no-code.
- **Difficile d'accès pour les débutants** : Des outils comme Webflow sont puissants, mais trop techniques pour beaucoup d'utilisateurs.
- **Mauvaise gestion des données** : Difficile de connecter plusieurs sources ou de gérer des bases complexes.
- **Pas fait pour des applications professionnelles** : La plupart sont pensées pour des sites simples, pas pour gérer plusieurs dashboards ou rôles utilisateurs.
- **Intégration API limitée** : L'ajout d'APIs externes est souvent compliqué, et pas assez flexible.
- **Peu de choix de BDs** : Impossible de choisir ou connecter librement des bases SQL, NoSQL, etc.
- **Sécurité basique** : Peu d'options sérieuses pour gérer les accès, permissions ou traçabilité.

2.3. Tableau comparatif des solutions existantes

Le tableau 2 présente une comparaison qualitative des principales plateformes de création web en fonctions des critères importantes

Tableau 2 : Tableau comparatif des solutions existantes

Plateforme	Facilité d'utilisation	Personnalisation avancée	Gestion des sous-plateformes	Données dynamiques	Intégration API	User Roles	Sécurité	Coût
WordPress		X		X	X	X	X	X
WYSIWYG Web Builder	X		X					
Adalo	X	X		X	X	X	X	X
Glide	X	X		X			X	
Wix	X	X	X					X
Webflow		X	X	X	X	X	X	
Squarespace	X	X					X	X
Tilda	X	X						X

3. Description de la solution proposée

Dans le cadre de notre projet de fin d'année, nous allons développer une application web qui permet de créer des sites dynamiques sans avoir besoin de coder. Cela inclut la conception de l'interface visuelle, le développement des fonctionnalités techniques, ainsi que la mise en place d'une base de données personnalisable.

Parmi les fonctionnalités attendues, on trouve :

- **Système de glisser-déposer pour construire les pages facilement :** L'utilisateur pourra choisir et organiser des blocs pour créer la structure de son site.
- **Définition des champs de la base de données :** Il pourra personnaliser les données qu'il souhaite gérer sur son site.
- **Gestion du contenu :** Le contenu pourra être ajouté, modifié ou supprimé directement depuis l'application.

- **Aperçu en temps réel du site :** L'utilisateur verra immédiatement le résultat de ses actions.

L'objectif est de proposer un outil simple, complet et accessible, qui permet à chacun de créer son propre site sans difficulté. Depuis son navigateur, l'utilisateur pourra construire l'architecture de son site, ajouter du contenu, personnaliser l'apparence et publier son projet. La plateforme offrira une interface claire, des blocs prêts à l'emploi et des options faciles à utiliser.

4. Choix méthodologique

Dans le cadre de ce projet, la méthode de prototypage a été retenue. Cette approche consiste en la réalisation rapide de maquettes ou de versions préliminaires du produit, favorisant ainsi une interaction continue avec l'encadrant. Ce dernier intervient tout au long du processus de développement, de la phase de conception à celle de finalisation. L'implication régulière de l'encadrant permet d'obtenir des retours fréquents et structurés, facilitant l'ajustement progressif du produit.
Par ailleurs, une modélisation à l'aide du langage UML (Unified Modeling Language) sera adoptée afin de formaliser les besoins, structurer les fonctionnalités et représenter clairement l'architecture du système à travers des diagrammes adaptés (cas d'utilisation, classes, séquences, etc.).

Conclusion

Dans ce chapitre, nous avons dressé un état des lieux des plateformes de création web actuellement disponibles sur le marché. Cette analyse nous a permis d'identifier plusieurs lacunes, notamment en matière de personnalisation avancée, de gestion dynamique de données et d'accessibilité pour les non-développeurs.

Sur cette base, nous avons pu définir les fonctionnalités essentielles que notre application devra intégrer pour se démarquer. Le chapitre suivant sera consacré à l'analyse des utilisateurs cibles ainsi qu'à la définition précise des besoins fonctionnels et techniques de notre solution.

Chapitre 2 : Analyse et spécification des besoins

Introduction

Ce chapitre traite la spécification et l'analyse des besoins fonctionnels et non fonctionnels de l'application et les expose en s'appuyant sur la modélisation UML.

1. Spécification des besoins

La spécification des besoins constitue la première phase dans le cycle de vie de tout logiciel à développer. Elle permet de bien comprendre le contexte du projet : identification des acteurs, identifications des fonctionnalités attendues, précision des risques etc.

1.1. Identification des besoins fonctionnels

Le tableau 3 illustre les besoins fonctionnels de notre projet.

Tableau 3 : Tableau des besoins fonctionnels

Besoin fonctionnel	Description	Exigences
Gestion des sites	Permet aux utilisateurs de créer de nouveaux sites web via l'interface. Permet d'éditer les paramètres généraux d'un site (nom, domaine, thème, etc.). Possibilité de supprimer un site avec confirmation préalable.	- Interface utilisateur intuitive - Validation des données - Historique des modifications
Gestion et composition des pages	Générer une nouvelle page en définissant son type (statique, dynamique, tableau de bord, etc.). Modifier le contenu et la structure des pages via une interface visuelle.	- Éditeur de pages WYSIWYG - Modèles de pages prédéfinis - Support des médias (images, vidéos)

	Supprimer une page du site avec une confirmation préalable. Structurer l'arborescence des pages pour la navigation.	
Gestion des plans de navigation des pages	Permettre de configurer les chemins d'accès aux différentes pages. Modifier les routes et les redirections en cas de changement de structure. Supprimer une route inutilisée sans impacter l'expérience utilisateur.	- Interface graphique pour la gestion des routes - Redirections automatiques - Gestion des erreurs 404
Gestion des graphiques	Sélectionner et ajouter des graphiques (barres, lignes, camemberts, etc.) aux tableaux de bord. Modifier les couleurs, légendes et formats des graphiques. Connecter les graphiques à des bases de données ou fichiers sources. Mettre à jour dynamiquement les	- Bibliothèque de graphiques variés (barres, lignes, secteurs, etc.) - Mise à jour en temps réel - Personnalisation des styles

	graphiques en fonction des nouvelles données.	
Gestion des utilisateurs	Permet à un administrateur d'ajouter de nouveaux utilisateurs avec rôles et permissions. Utilisation de JWT pour une connexion sécurisée. Définition des accès pour chaque utilisateur (Admin, Éditeur, Lecteur). Modifier les informations ou désactiver un utilisateur.	- Système d'authentification sécurisé (JWT) - Gestion des rôles et permissions - Interface de gestion des utilisateurs

1.2. Identification des besoins non-fonctionnels

Ergonomie : Nous allons concevoir une interface intuitive et fluide avec Angular 18 en utilisant des composants UI modernes. Le drag and drop sera intégré pour simplifier l'expérience utilisateur. Des tests utilisateurs seront réalisés pour affiner l'UX.

Fiabilité : L'application doit être stable et performante. Nous utiliserons Spring Boot pour gérer les requêtes efficacement, en optimisant les requêtes MongoDB et en mettant en place un monitoring des erreurs avec des logs centralisés.

Sécurité : L'authentification et l'autorisation seront gérées avec JWT pour sécuriser les sessions. Les données sensibles seront chiffrées et des contrôles d'accès basés sur les rôles seront mis en place. Des tests de sécurité réguliers seront effectués.

Maintenabilité et réutilisation : Le code sera structuré de manière modulaire avec Angular et Spring Boot pour faciliter les mises à jour et les améliorations futures. L'utilisation de Git/GitHub assure une bonne gestion des versions et du travail collaboratif.

2. Analyse

L'analyse permet de dégager les différents acteurs qui peuvent agir avec la nôtre application et donne une représentation fonctionnelle des différents acteurs et fonctionnalités du système sous la forme d'un diagramme de cas d'utilisation

2.1. Identification des acteurs :

Utilisateur : L'acteur utilisateur représente tout individu inscrit sur la plateforme disposant d'un compte personnel. Une fois authentifié, il peut accéder à un espace privé lui permettant de gérer son propre profil ainsi que les sites web qu'il administre. Ce dernier peut également, en fonction de ses privilèges, interagir avec certaines pages et routes associées aux sites, à condition d'être connecté au préalable.

Responsable Utilisateur : L'acteur Responsable Utilisateur hérite des privilèges de l'utilisateur tout en disposant de droits supplémentaires lui permettant de gérer les bases de données ainsi que les utilisateurs de la plateforme. Il s'agit donc d'un acteur intermédiaire jouant un rôle clé dans la supervision des activités techniques et administratives relatives aux sites.

Admin : L'acteur Admin constitue le niveau le plus élevé dans la hiérarchie des utilisateurs. En plus des fonctionnalités accessibles aux autres rôles, il possède un accès exclusif à la gestion des responsables. Grâce à son tableau de bord, il contrôle l'ensemble des opérations critiques de la plateforme et garantit le bon fonctionnement de son écosystème.

2.2. Diagramme des cas d'utilisation global

Le diagramme de cas d'utilisation est une représentation du comportement du système de point de vue de l'utilisateur. C'est une définition des besoins qu'attend un utilisateur du système. Il contient tous les cas d'utilisation en liaison directe ou

indirecte avec les acteurs. Dans le diagramme de cas d'utilisation global représenté par la figure 1, nous modélisons tous les cas d'utilisations de base afin d'avoir une vue globale de fonctionnement de l'application

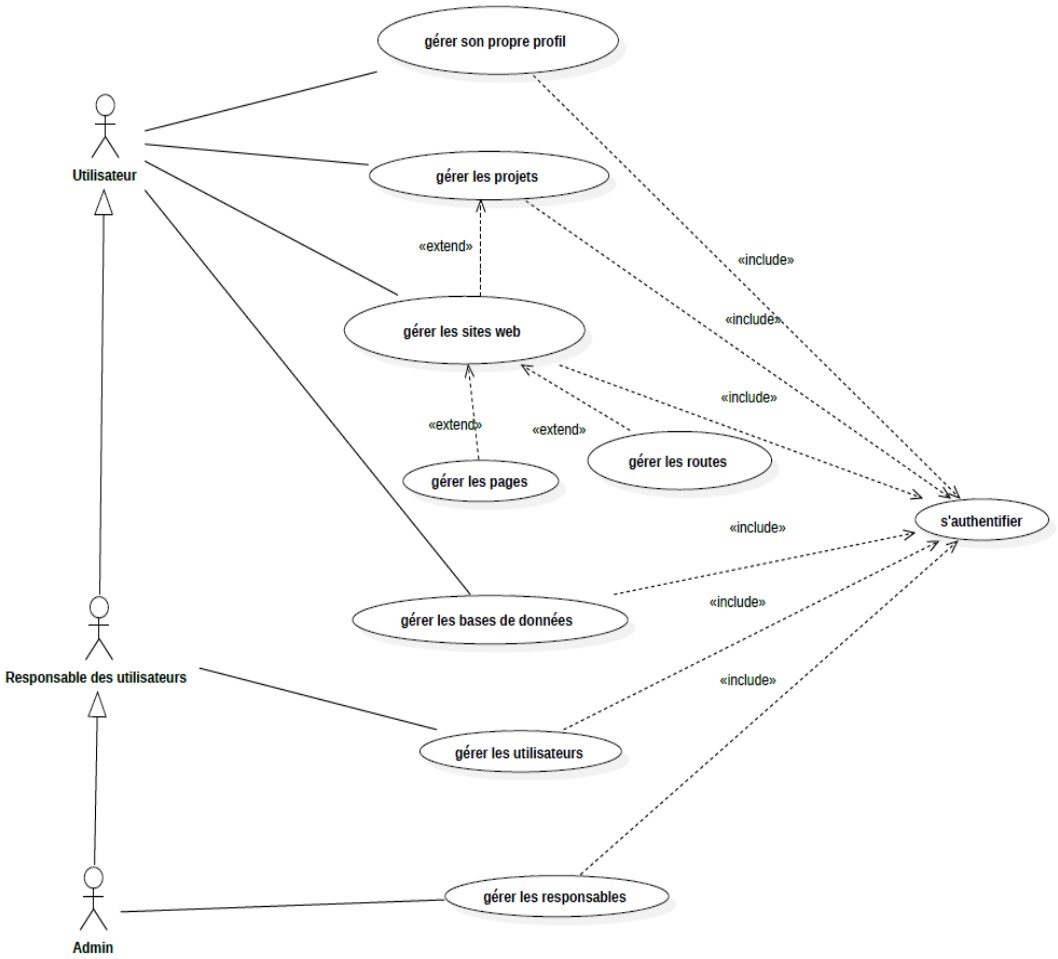


Figure 1 : Diagramme des cas d'utilisation global

2.3. Raffinement des cas d'utilisation

Dans cette rubrique nous détaillons quelques cas d'utilisation en utilisant les diagrammes de cas d'utilisation raffinés, si c'est nécessaire, les descriptions textuelles et les diagrammes de séquences système du scénario nominal de chacun des cas d'utilisation choisi.

2.3.1 Raffinement de cas d'utilisation « Gérer les pages »

- ✓ **Diagramme du cas d'utilisation « Gérer les pages »**

La figure 2 illustre le raffinement du cas d'utilisation intitulé « Gérer les pages ». Ce raffinement détaille les différentes actions accessibles à l'acteur Utilisateur

lorsqu'il interagit avec le système pour administrer le contenu des pages. En sélectionnant cette fonctionnalité, l'utilisateur a la possibilité de :

Ajouter une nouvelle page : il peut créer une page vierge qu'il pourra ensuite personnaliser en ajoutant des éléments de contenu (textes, images, sections, etc.).

Supprimer une page existante : il peut retirer définitivement une page du site, en cas de besoin de restructuration ou de suppression d'un contenu obsolète.

Modifier une page : il peut éditer une page existante, c'est-à-dire mettre à jour son contenu, son apparence ou sa structure à travers l'éditeur intégré. Ce processus permet à l'utilisateur de maintenir son site à jour et d'adapter l'interface aux besoins évolutifs sans intervention technique supplémentaire.

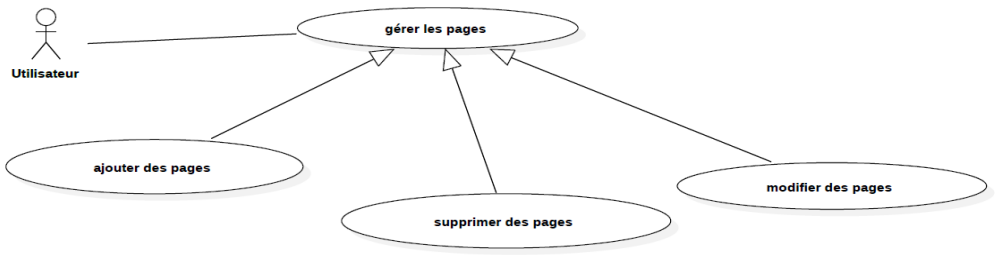


Figure 2 : Diagramme du cas d'utilisation raffiné de « Gérer les pages »

✓ **Description textuelle du cas d'utilisation « Gérer les pages »**

Le tableau 4 présente la description textuelle du cas d'utilisation « Gérer les pages »

Tableau 4 : Description textuelle du cas d'utilisation « Gérer les pages »

Titre :	Gérer les pages
Acteur :	Utilisateur
Résumé :	Ce cas d'utilisation permet à l'utilisateur de gérer les pages de son site web en ajoutant, supprimant ou modifiant des pages existantes via l'interface de l'application.

Préconditions :	L'utilisateur doit être authentifié dans le système.
Scénario nominal :	1.L'utilisateur accède à la section "Gestion des pages" depuis le tableau de bord. 2. Le système affiche la liste des pages existantes. L'utilisateur peut choisir : ✓ D'ajouter une nouvelle page, ✓ De modifier une page existante, ✓ Ou de supprimer une page. 3.Si l'utilisateur choisit d'ajouter une page, un formulaire de création est présenté. 4.Si l'utilisateur choisit de modifier une page, l'éditeur de page s'ouvre avec le contenu actuel chargé. 5.Si l'utilisateur choisit de supprimer une page, une confirmation est demandée. 6.Le système enregistre les modifications et met à jour la liste des pages.
Scénario alternative :	**Dans l'étape 6 :** o L'utilisateur peut annuler la suppression en cliquant sur "Annuler". o Le système retourne à la liste des pages sans effectuer de suppression. **Dans l'étape 5 :** o L'utilisateur peut annuler l'édition d'une page, le système conserve la version précédente sans modification.

Scénario d'erreur	**Dans l'étape 4 ou 5 :** o Le système affiche un message d'erreur si les champs requis lors de l'ajout ou de la modification sont vides. o Le système affiche un message d'erreur en cas d'échec de la sauvegarde des modifications (exemple : problème de connexion au serveur).

2.3.2 Diagramme de séquence système du cas d'utilisation « Gérer les pages »

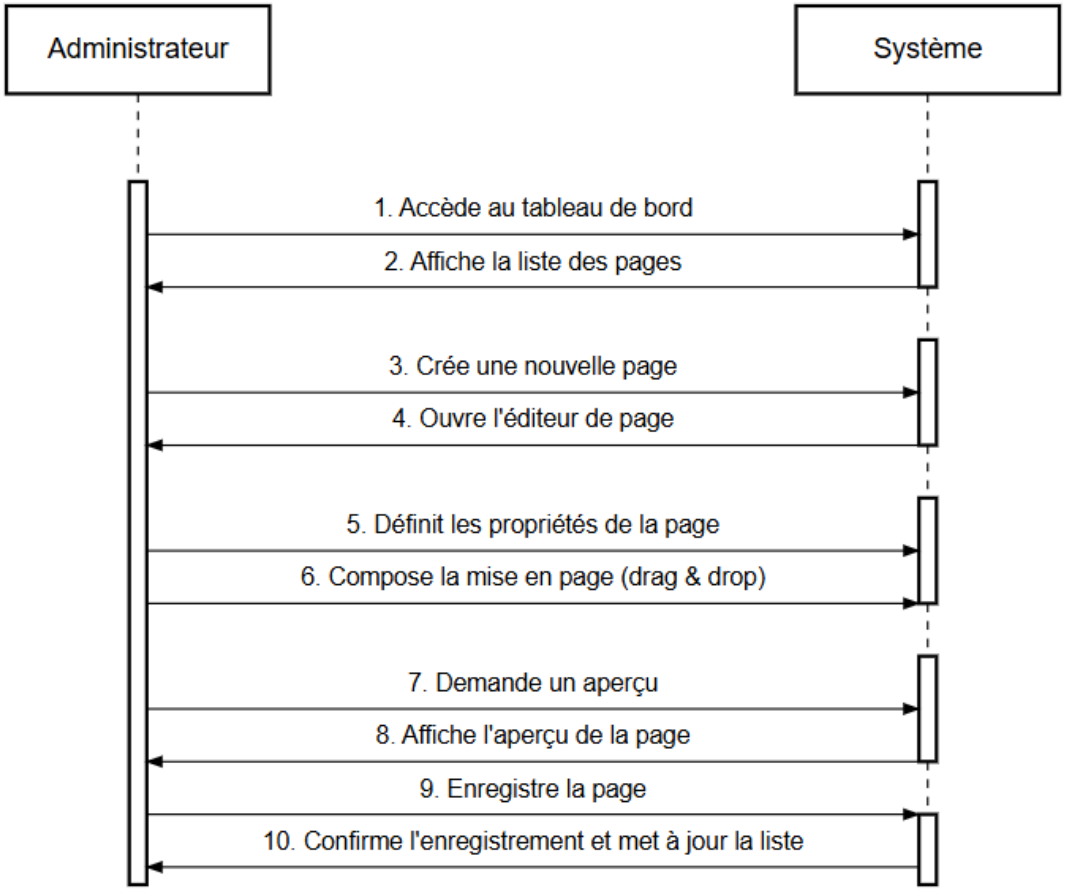
Les étapes de déroulement du cas d'utilisation « Gérer les pages » sont décrites par le diagramme de séquence illustré par la figure 3

Figure 3 : Diagramme système de « Gérer les pages »

2.3.3 Raffinement de cas d'utilisation « Gérer les bases de données »

La figure 4 illustre le raffinement du cas d'utilisation intitulé « Gérer les bases de données ». Ce raffinement détaille les différentes actions accessibles à l'acteur Utilisateur lorsqu'il interagit avec le système pour administrer la gestion des bases de données.

En sélectionnant cette fonctionnalité, l'utilisateur peut :



Ajouter une nouvelle base de données : il peut créer une base qu'il pourra ensuite personnaliser en y ajoutant des éléments de contenu.

Supprimer une base de données existante : il peut retirer définitivement une base du site, en cas de besoin de restructuration ou de suppression d'un contenu obsolète.

Modifier une base de données : il peut éditer une base existante, c'est-à-dire mettre à jour son contenu, son apparence ou sa structure à travers l'éditeur intégrée

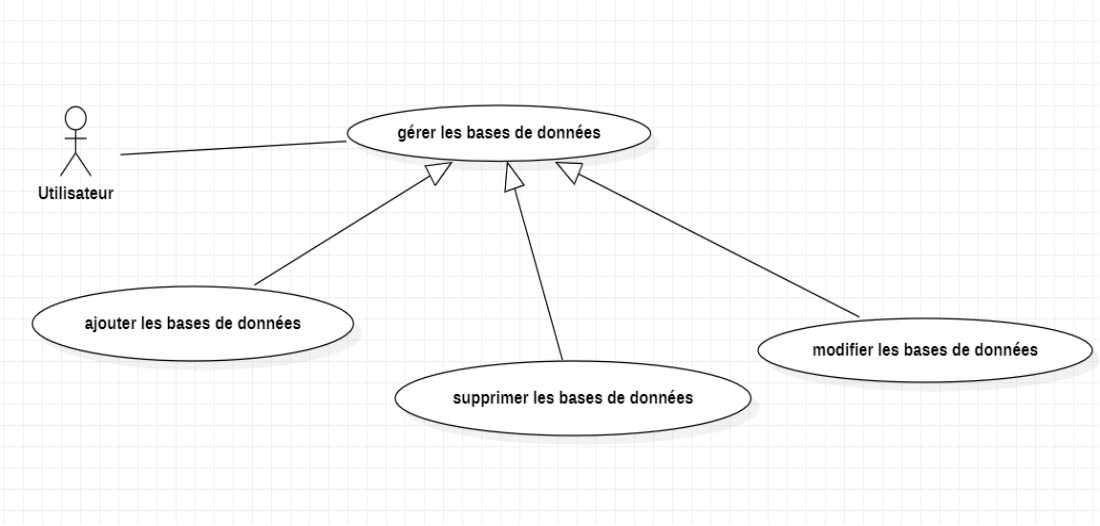


Figure 4 : Raffinement du cas d'utilisation gérer les bases de données

Conclusion

Ce chapitre a permis de cerner les besoins du projet, aussi bien fonctionnels que non-fonctionnels, et d'en analyser la structure à travers l'identification des acteurs et la modélisation des cas d'utilisation. Cette étape est essentielle pour orienter correctement la conception et le développement.

Chapitre 3 : Etude Conceptuelle

Introduction

Ce chapitre est dédié à la phase de conception qui est la phase la plus importante pour la mise en place d'un logiciel ; elle permet de décrire, d'une manière détaillée, les différentes fonctionnalités d'un système dans le but de simplifier sa mise en œuvre. Nous mettons l'accent, principalement, sur le modèle architectural choisi (design pattern) et le diagramme de classe qui reflète les différentes entités de la base de données et les relations entre elles

1. Modèle architectural

Aujourd'hui, les programmes informatiques ont gagné en sophistication, ce qui engendre des instructions codées qui peuvent être ardues à déchiffrer et à faire évoluer. Il devient donc essentiel d'avoir une structure qui agence le code et distribue les tâches entre différentes strates. L'objectif est de renforcer l'unité interne et de diminuer les interdépendances.

C'est précisément le rôle des modèles d'organisation logicielle. Ils aident à distinguer les règles de fonctionnement de la manière dont le programme est montré à l'utilisateur. Ceci permet d'assurer que les différentes parties du système fonctionnent de manière autonome, ce qui facilite la réutilisation des instructions codées, leur adaptation et leur capacité à supporter des modifications.

Pour notre réalisation, nous avons opté pour une organisation de type MVVM pour la partie Frontend (Angular), dont le schéma est présenté à la figure 5.

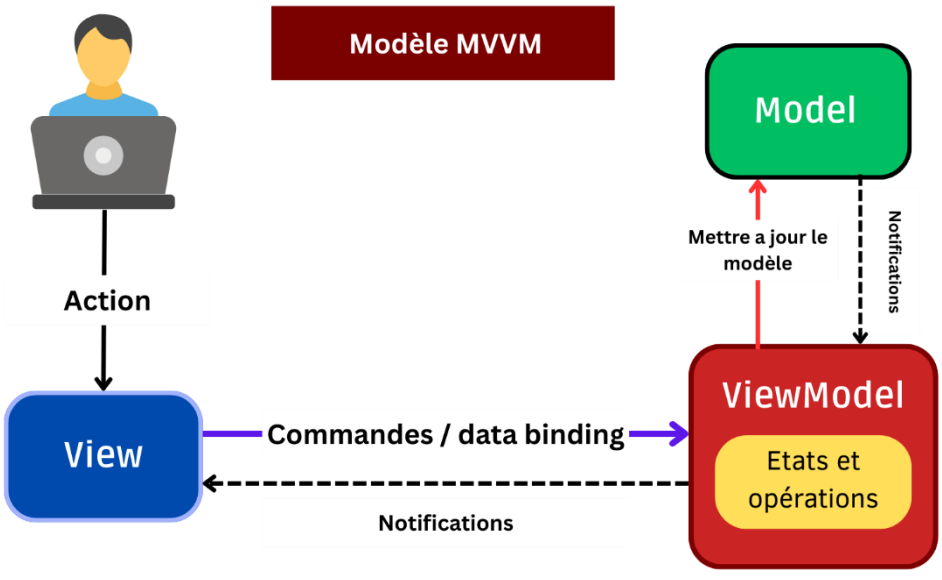


Figure 5 : Modèle MVVM

Pour la partie serveur (Backend), nous avons adopté une architecture **à plusieurs niveaux**, souvent désignée comme une variante de l'architecture **MVC (Model-View-Controller)** ou une architecture **en couches**. Cette approche vise également à séparer les préoccupations et à organiser le code de manière logique. Les composants clés de notre modèle Backend sont les suivants :

- **Modèle (Model/Entité) :** Représente les données de notre application, telles que les informations sur les utilisateurs, les produits, etc. Ces modèles sont souvent des objets Java qui correspondent aux tables de notre base de données.
- **Référentiel (Repository) :** Est responsable de l'accès aux données. Il fournit des méthodes pour interagir avec la base de données (par exemple, enregistrer, récupérer, modifier, supprimer des données).
- **Service :** Contient la logique métier de notre application. C'est ici que les règles de fonctionnement spécifiques à notre domaine sont implémentées. Les services utilisent les référentiels pour accéder aux données et effectuent les traitements nécessaires.
- **Contrôleur (Controller) :** Gère les requêtes provenant de l'interface utilisateur (Frontend) ou d'autres clients. Il reçoit les requêtes, fait appel aux services pour effectuer les opérations nécessaires et renvoie une réponse.
- **Objet de Transfert de Données (DTO - Data Transfer Object) :** Est utilisé pour transporter des données entre les différentes couches, notamment entre la couche de service et la couche de contrôleur. Cela permet de structurer les données de manière spécifique pour chaque interaction et d'éviter d'exposer directement les modèles de données.

  La figure 6 illustre l'architecture de notre backend :

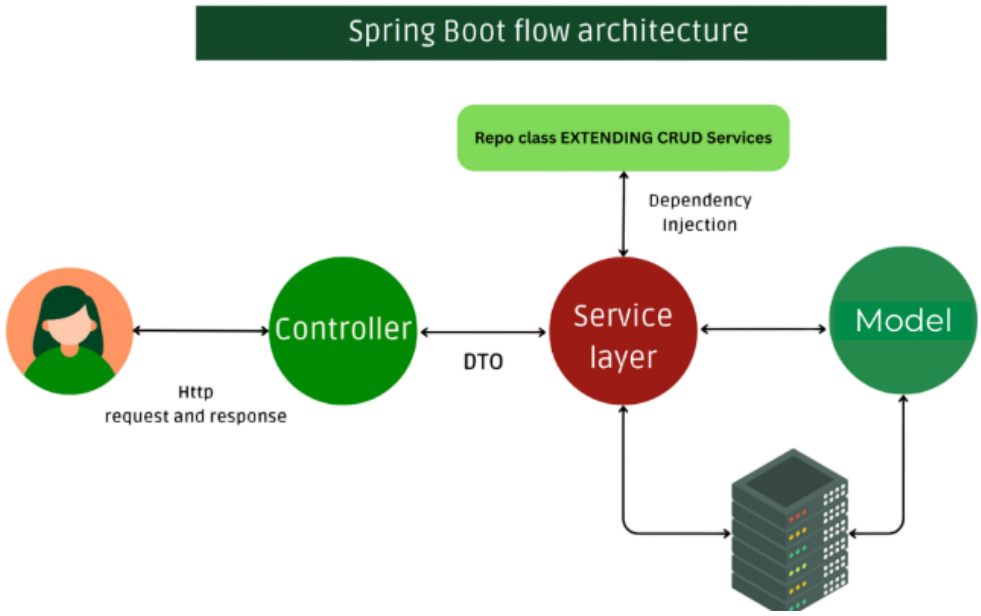


Figure 6 : Architecture backend du spring boot

2. Diagramme de classe

Un diagramme de classes modélise la structure statique d'un système logiciel en décrivant ses classes, leurs attributs et les relations qui les unissent.

Le diagramme de classes illustré par la figure 7 représente l'architecture de notre application.

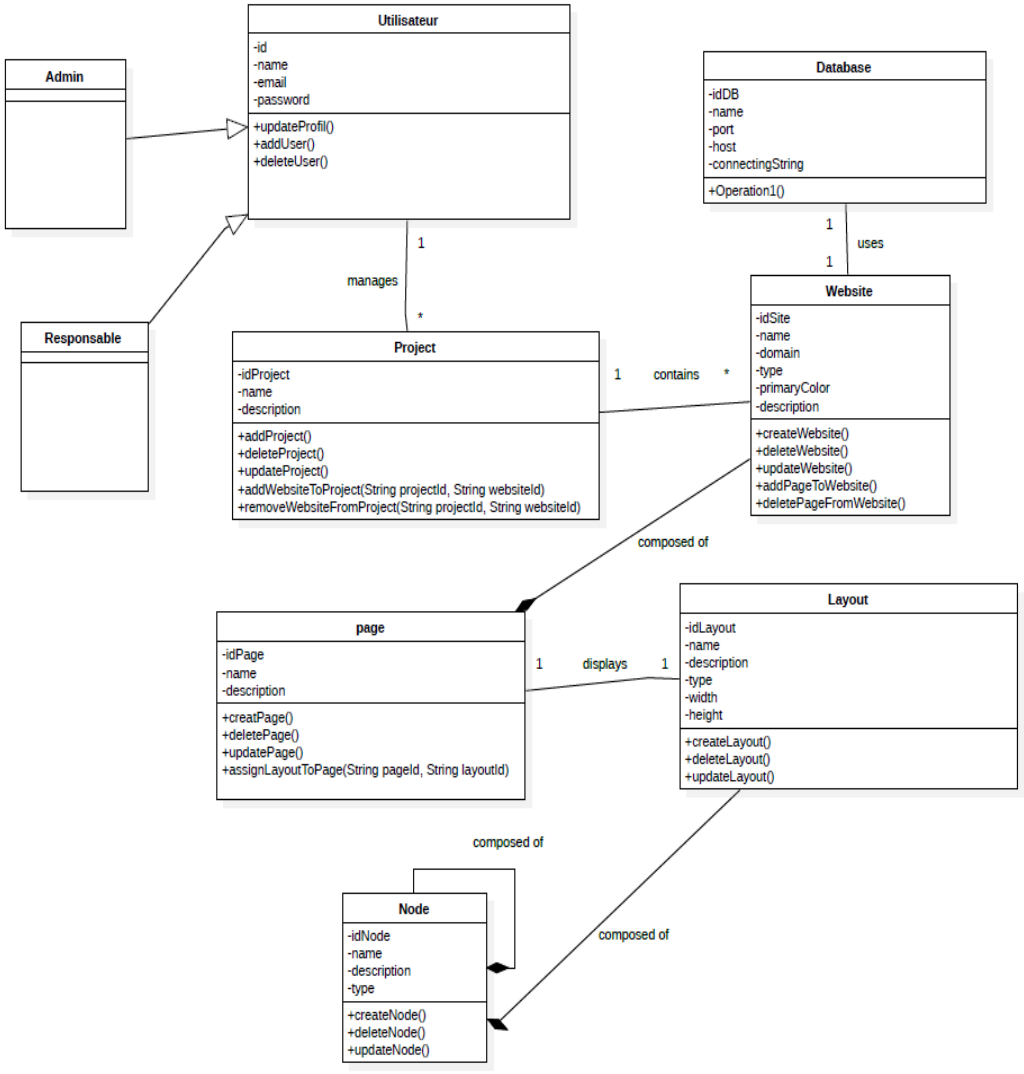


Figure 7 : Diagramme de classes

Utilisateur : Cette classe représente un utilisateur du système avec des attributs d'identification (id, name, email, password). Elle contient des méthodes pour gérer le profil utilisateur (updateProfile, addUser, deleteUser). Un utilisateur peut être soit un Admin, soit un Responsable, et il peut gérer plusieurs projets.

Admin : Cette classe est une spécialisation d'Utilisateur qui représente un administrateur du système. Elle hérite des propriétés et méthodes de la classe Utilisateur et peut avoir des fonctionnalités supplémentaires d'administration.

Responsable : Cette classe est une spécialisation d'Utilisateur qui représente un responsable de projet. Elle hérite des propriétés et méthodes de la classe Utilisateur et est associée à la gestion de projets spécifiques.

Project : Cette classe représente un projet dans le système avec des attributs comme idProject, name et description. Elle contient des méthodes pour gérer les projets (addProject, deleteProject, updateProject) et pour associer des sites web aux projets (addWebsiteToProject, removeWebsiteFromProject). Un projet est géré par un utilisateur et peut contenir plusieurs sites web.

Website : Cette classe représente un site web qui fait partie d'un projet avec des attributs comme idSite, name, domain, type, primaryColor et description. Elle contient des méthodes pour gérer les sites web (createWebsite, deleteWebsite, updateWebsite) et pour gérer les pages (addPageToWebsite, deletePageFromWebsite). Un site web utilise une base de données et est composé de plusieurs pages.

Database : Cette classe représente une base de données utilisée par un site web avec des attributs comme idDB, name, port, host et connectionString. Elle contient des méthodes pour effectuer des opérations sur la base de données (Operation). Une base de données peut être utilisée par plusieurs sites web.

Page : Cette classe représente une page d'un site web avec des attributs comme idPage, name et description. Elle contient des méthodes pour gérer les pages (createPage, deletePage, updatePage) et pour associer des layouts aux pages (assignLayoutToPage). Une page affiche un layout et fait partie d'un site web.

Layout : Cette classe représente une mise en page utilisée par les pages avec des attributs comme idLayout, name, description, type, width et height. Elle contient des méthodes pour gérer les layouts (createLayout, deleteLayout, updateLayout). Un layout est composé de plusieurs nœuds.

Node : Cette classe représente un élément constitutif d'un layout avec des attributs comme idNode, name, description et type. Elle contient des méthodes pour gérer les nœuds (createNode, deleteNode, updateNode). Les nœuds sont les composants de base qui forment un layout.

Ce modèle permet de concevoir une interface modulaire et personnalisable, où les utilisateurs peuvent construire dynamiquement des applications web en glissant-déposant divers éléments sur leurs pages.

3. Diagramme de séquence détaillé du cas d'utilisation « gérer les pages »

Pour mieux représenter le processus de gestion des pages d'un site web par glisser-déposer de composants et conteneurs, nous avons décomposé le diagramme de séquence global en plusieurs sous-diagrammes. Cela permet de clarifier chaque étape, d'éviter la surcharge visuelle et de refléter la répartition des responsabilités entre les différents contrôleurs : ajout des nodes, enregistrement de pages, et sauvegarde du site dans la base de données.

3.1. Diagramme de séquence du partie « Création d'une node »

La figure 8 présente le diagramme de séquence de la création d'une node

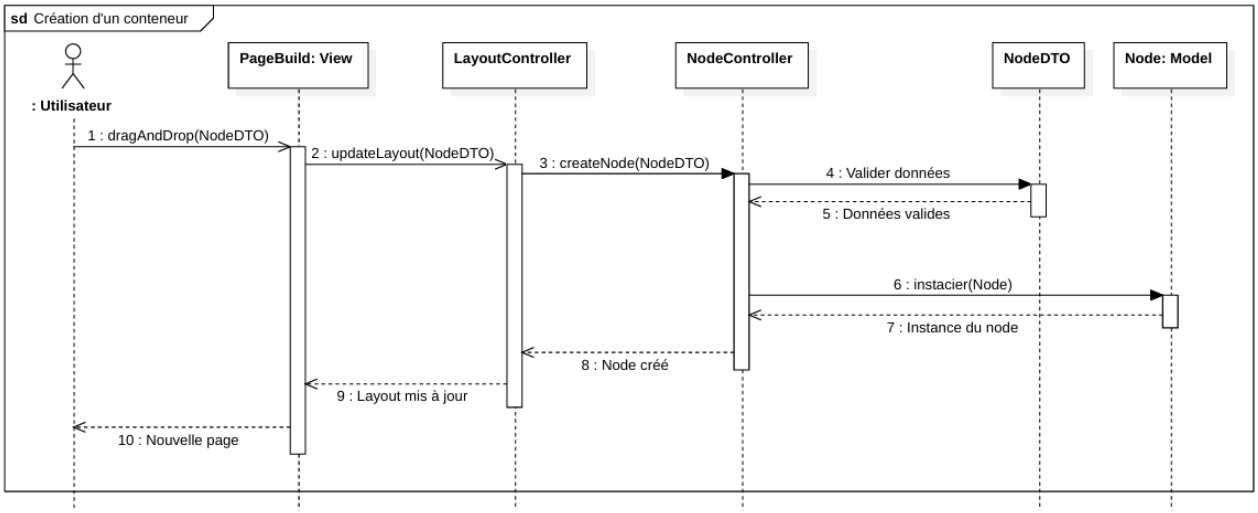


Figure 8 : Diagramme de séquence de la création d'une node

3.2. Diagramme de séquence « Enregistrement et chargement d'une page »

Le diagramme présenté par la figure 9 illustre le processus d'enregistrement et du chargement d'une page, où les conteneurs et composants déjà construits sont assemblés et structurés avant d'être intégrés au site.

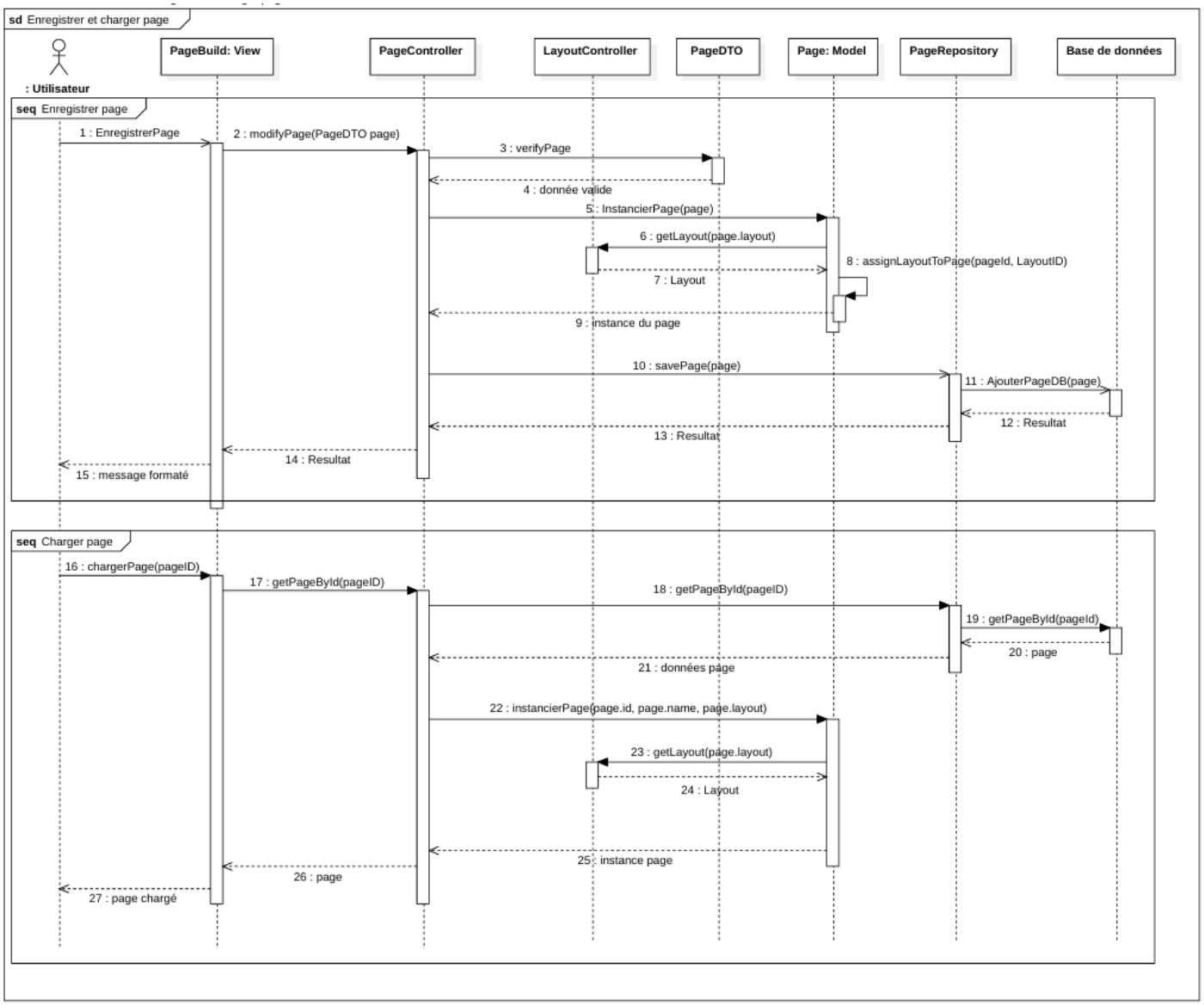


Figure 9 : Diagramme de séquence enregistrement et chargement d'une page

4. Étude de l'ergonomie de l'interface

Cette section est consacrée à l'étude des aspects ergonomiques de l'interface afin d'assurer une utilisation intuitive et agréable pour l'utilisateur.

4.1. Conception du logo et approche visuelle

La première étape de la conception de notre interface a été la création d'un logo et le choix d'une approche visuelle moderne, professionnelle et épurée. Cette démarche visait à véhiculer une image de simplicité et d'efficacité, tout en restant fidèle aux principes d'ergonomie et d'accessibilité. Le logo, ainsi que l'esthétique générale, ont été conçus pour créer une identité forte et cohérente, adaptée à un usage prolongé.

La figure 10 présente le logo de l'application développée:

Figure 10 : Logo de l'application

4.2. Choix de la palette de couleurs

Afin de soutenir l'approche visuelle sobre et professionnelle, nous avons sélectionné une palette de couleurs en respectant le principe de ne pas dépasser sept couleurs principales. Les couleurs choisies sont présentés dans le tableau 5.

Tableau 5 : Tableau descriptif des couleurs choisies

Couleur	Code couleur	Description
Couleur principale	#007bff	Bleu principal
Couleur secondaire	#0056b3	Bleu secondaire foncé
Fond sombre	#1a1a1a	Arrière-plan sombre
Fond clair	#f8f9fa	Arrière-plan clair
Texte sombre	#333333	Couleur du texte foncé
Texte clair	#f0f0f0	Couleur du texte clair

4.3. Implémentation d'une hiérarchie visuelle : le schéma en "F"

Pour structurer efficacement le contenu, nous avons utilisé une hiérarchie visuelle inspirée du schéma en "F" conçu par le groupe d'experts « Nielsen Norman Group », qui permet de capter l'attention de l'utilisateur de manière naturelle. Nous avons également intégré une barre de navigation latérale (side nav bar) qui permet une navigation fluide et intuitive. Le contenu se charge dynamiquement dans la section principale, offrant ainsi une expérience cohérente et facile d'accès à toutes les sections de l'application.

En figure 11, l'interface prototype adopte une structure en "F".

Figure 11 : Prototype d'interface utilisant le schéma en « F »

Cette limitation a été choisie pour éviter toute surcharge cognitive, tout en maintenant une certaine cohérence esthétique et un contraste optimal. Les couleurs ont été choisies pour renforcer l'aspect moderne et intuitif de l'interface, tout en facilitant la lisibilité et la distinction des éléments.

Conclusion

Ce chapitre a défini les fondations conceptuelles du projet, en abordant l'architecture du système, la modélisation UML, l'ergonomie de l'interface, les aspects de sécurité ainsi que la qualité du code. Ces éléments assurent une base solide pour la suite du développement.

Chapitre 4 : Mise en œuvre

Introduction

Ce chapitre sert à la présentation de l'environnement matériel et logiciel utilisés lors du développement de l'application et à l'exposition du travail réalisé.

1. Environnement Matériel

Ce projet a été exécuté en employant un ordinateur équipé des spécifications détaillés dans le Tableau 6

Tableau 6 : Environnement matériel

Caractéristique	PC
Processeur	Intel CORE I5
Ram	32 GO
Système d'exploitation	Windows 11 professionnel

2. Environnement logiciel

Cette section présente les outils logiciels et les technologies utilisés pour la mise en place de l'application "Web Builder"

Le tableau 7 résume les logiciels que nous avons utilisé dans notre travail.

Tableau 7 : Environnement logiciel

Logiciel	Logo	Description
WebStorm		[1]**WebStorm** est un environnement de développement intégré (IDE) puissant et intelligent spécialement conçu par JetBrains pour le développement web en JavaScript et TypeScript. Il offre une gamme complète d'outils et de fonctionnalités pour faciliter la création, l'édition, le débogage et le test

		de code pour des applications web modernes.
Intellij IDEA		[2] **IntelliJ IDEA** est un environnement de développement intégré (IDE) développé par JetBrains, reconnu pour sa puissance et son ensemble complet de fonctionnalités destinées aux développeurs de logiciels, en particulier pour le développement en **Java**.
MongoDB Compass		[3]**MongoDB Compass** est une interface utilisateur graphique (GUI) officielle développée par MongoDB Inc. pour interagir avec les bases de données **MongoDB**. Il permet aux développeurs et aux administrateurs de visualiser, de manipuler et d'analyser leurs données MongoDB de manière intuitive sans avoir à écrire de commandes shell complexes.
Postman		[4]**Postman** est une plateforme collaborative pour le développement d'API (Interface de Programmation Applicative). Elle est principalement utilisée pour construire, tester, documenter et partager des API. Postman offre une interface utilisateur graphique conviviale pour envoyer des requêtes HTTP (GET, POST, PUT, DELETE, etc.) à des serveurs d'API et

		examiner les réponses.
Github		[5] **GitHub** est une plateforme d'hébergement de code en ligne basée sur Git, un système de contrôle de version distribué. Fondamentalement, il permet aux développeurs de stocker, de gérer et de collaborer sur leurs projets de code avec d'autres personnes.

3. Technologies utilisées

Le tableau 8 résume les technologies que nous avons utilisé dans notre travail

Tableau 8 : Technologies utilisées

Technologie	Logo	Description
Angular		[6] **Angular** est un framework de développement web open source, basé sur TypeScript, utilisé pour construire des interfaces utilisateur (UI) riches et dynamiques pour des applications web monopages (Single-Page Applications - SPA). Il est maintenu par Google et une large communauté de développeurs.
Spring Boot		[7]**Spring Boot** est un framework Java open source qui simplifie radicalement la création et le déploiement d'applications autonomes basées

		sur Spring.
MongoDB		[8]**MongoDB** est une base de données NoSQL orientée documents qui stocke les données dans des structures flexibles de type JSON (appelées BSON en interne). Contrairement aux bases de données relationnelles traditionnelles avec leurs tables et schémas rigides, MongoDB permet aux documents au sein d'une même collection d'avoir des structures différentes, offrant ainsi une grande agilité pour les développeurs et les données évolutives
Snyk		[9]**Snyk** est une plateforme de sécurité pour les développeurs d'applications et cloud qui permet de sécuriser l'ensemble de leur application en identifiant et corrigeant les vulnérabilités de leurs premières lignes de code à leur cloud de production.
SonarQube		[10]**SonarQube** est un logiciel open source de gestion de la qualité du code. Il est utilisé pour inspecter le code source des logiciels et applications en développement et détecter des bugs, vulnérabilités de sécurités, instances de code dupliqué et autres anomalies pouvant nuire à la qualité du code source, et ainsi au

		fonctionnement de l'application qui en résulte.
Git		[11]**Git** est un système de contrôle de version distribué (DVCS) conçu pour suivre les modifications apportées aux fichiers au fil du temps**.** Son objectif principal est de permettre à plusieurs personnes de travailler simultanément sur un même projet, de gérer l'historique des modifications et de faciliter la collaboration sans écraser le travail des autres.
Bootstrap		[12] **Bootstrap** est une librairie (ou framework) CSS open source, conçue pour faciliter et accélérer le développement d'interfaces web responsives et esthétiquement plaisantes.

La figure 12 présente l'architecture utilisée, en mettant en évidence les technologies employées à chaque couche de l'application

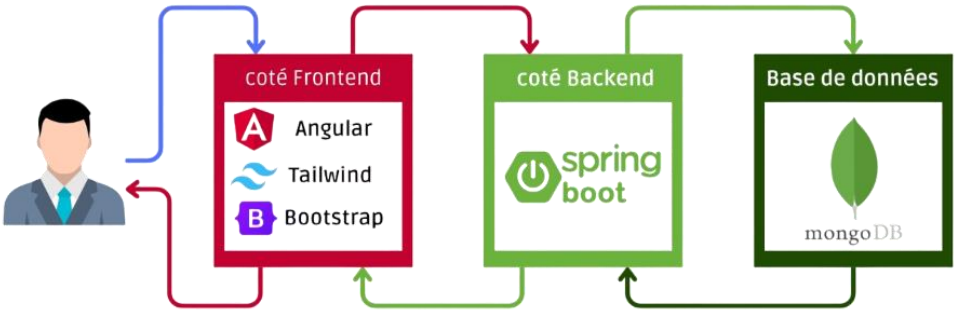


Figure 12 : Technologies utilisées par couche de l'application

4. Considérations des mesures de sécurité

Afin de garantir la sécurité de notre application, nous avons mis en place une architecture de sécurité solide. Du côté backend, nous avons intégré **Spring Security** pour gérer l'authentification et l'autorisation. L'authentification est basée sur des **JSON Web Tokens (JWT)**, permettant ainsi une gestion sécurisée et sans état des sessions utilisateur. Lorsqu'un utilisateur s'authentifie, un jeton est généré et doit être inclus dans chaque requête ultérieure pour accéder aux ressources protégées. Nous avons également mis en œuvre des **contrôles d'accès basés sur les rôles**, garantissant que seules les personnes autorisées peuvent effectuer certaines actions. Par ailleurs, l'**API Jakarta Validation** est utilisée pour valider les données entrantes, réduisant les risques d'injections malveillantes. Côté frontend, des **gardiens de routes Angular (route guards)** assurent que seules les personnes connectées peuvent accéder aux vues sensibles. Des configurations supplémentaires telles que les **politiques CORS**, les **en-têtes de sécurité HTTP**, ainsi que le hachage des mots de passe (via BCrypt) complètent notre dispositif de sécurité. Ces mesures assurent la confidentialité, l'intégrité et la protection des données tout au long de l'utilisation de l'application.

Le figure 13 présente une vue descriptive de notre architecture implémentée

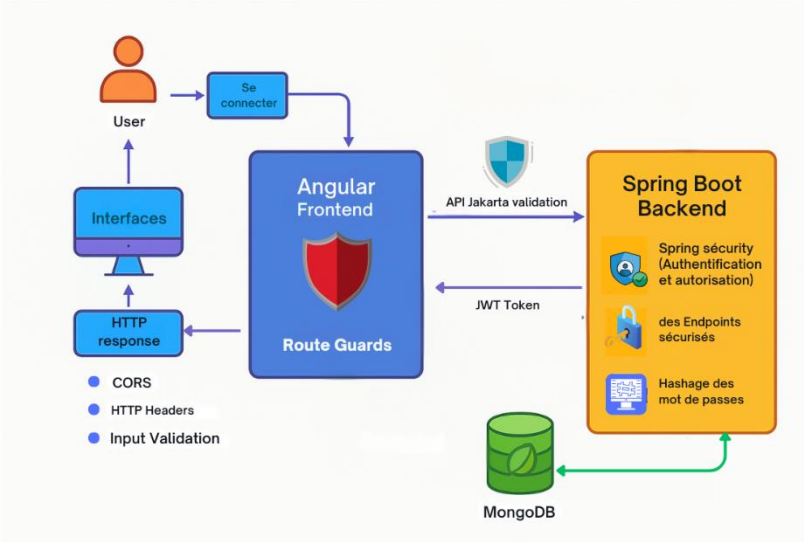


Figure 13 : Architecture de sécurité

5. Approche de qualité du code

Dans le cadre de notre projet, nous avons adopté une démarche rigoureuse visant à garantir à la fois la qualité du code et la sécurité de l'application. Pour cela,

nous avons mis en œuvre les principes du **Clean Code**, favorisant un code lisible, maintenable et peu sujet aux erreurs grâce à une structure claire, une bonne nomenclature et la réduction de la duplication. Afin de renforcer cette approche, nous avons intégré **SonarQube** dans notre pipeline de développement pour effectuer une analyse statique du code, permettant de détecter les mauvaises pratiques, les bugs potentiels ainsi que les dettes techniques. Les rapports générés nous ont guidés dans l'amélioration continue de notre base de code. En parallèle, nous avons utilisé **Snyk**, un outil spécialisé dans la sécurité des dépendances, afin d'identifier les vulnérabilités connues dans les bibliothèques utilisées. Grâce aux alertes de Snyk, nous avons pu mettre à jour ou remplacer les paquets concernés, réduisant ainsi les risques d'exploitation malveillante. L'intégration conjointe de ces outils nous a permis d'assurer une meilleure **fiabilité, sécurité et pérennité** de notre solution logicielle.

6. Interfaces de l'application

Nous présentons, ici, l'ensemble des interfaces développées pour l'application :

6.1. Page d'acceuil

La page d'accueil **(figure 14)** est la première interface que l'utilisateur voit au lancement de l'application. Elle comprend un en-tête avec le logo "Web App Builder", une barre de navigation (Fonctionnalités, Modèles, Tarifs, etc.) et des boutons "Connexion" / "Inscription". La section principale présente un titre accrocheur ("Build Your Website With Ease") et une description du service, accompagnés de boutons d'action ("Commencer gratuitement" et "Voir la démo"). Plus bas, trois atouts clés sont mis en avant via des cartes visuelles : des modèles professionnels, une édition par glisser-déposer et une compatibilité multi-appareils. Un appel à action final invite à créer son site ("Prêt à créer votre site ?"), tandis que le pied de page regroupe les liens utiles (Produit, Société, Ressources) et les mentions légales. (Note : L'accès aux fonctionnalités nécessite une inscription préalable.)

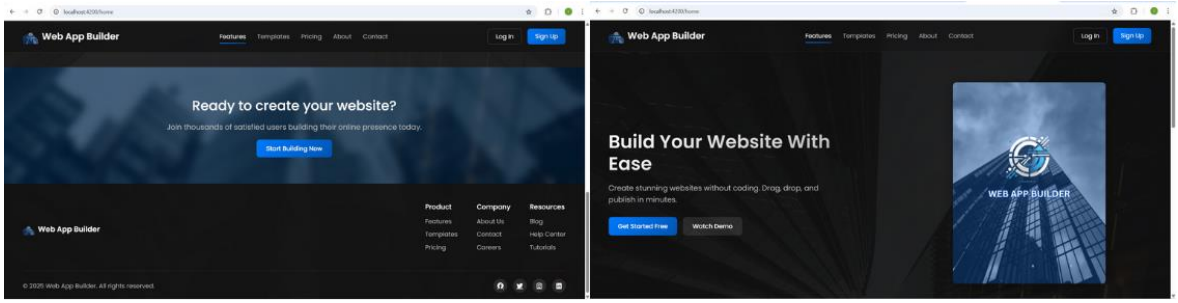


Figure 14 : Interface d'accueil

6.2. Module Authentication

Interface de connexion

La page de connexion **(figure 15)** permet aux utilisateurs d'accéder à leur compte personnel. Elle comprend un formulaire avec deux champs obligatoires : un pour l'adresse email (avec validation automatique du format **(figure 16)**) et un autre pour le mot de passe. Des messages d'erreur s'affichent dynamiquement si les champs sont mal renseignés. Un bouton "Log In" (qui devient "Logging in..." pendant le chargement) valide la connexion. Les utilisateurs ont également la possibilité de se connecter via Google (bouton "Continue with Google") ou de réinitialiser leur mot de passe via le lien "Forgot Password ?". Une illustration de l'application accompagne le formulaire sur le côté droit. (Note : Tous les champs doivent être valides pour permettre la soumission du formulaire. **(Figure 16)**)

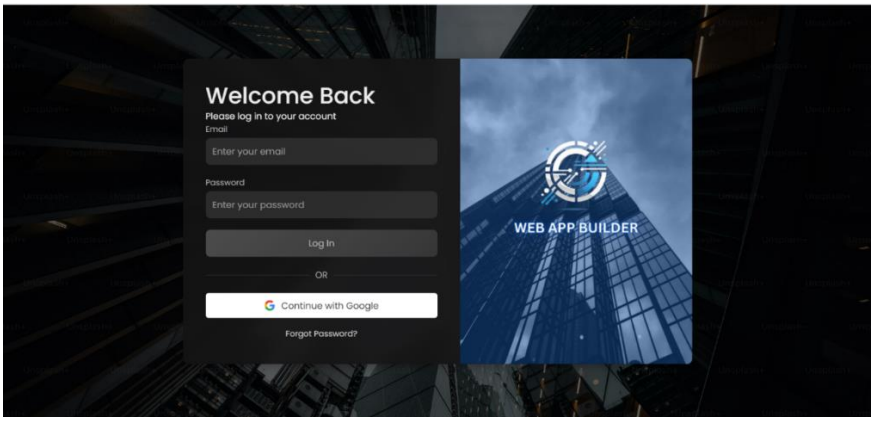


Figure 15 : Interface de connexion

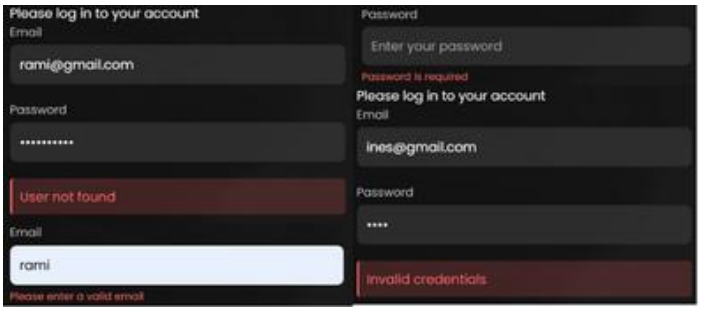


Figure 16 : Gestion des erreurs pour l'interface de connexion

Interface d'inscription

La page d'inscription **(figure 17)** permet aux nouveaux utilisateurs de créer un compte sur la plateforme. Elle présente un formulaire complet avec plusieurs champs obligatoires : le nom complet, l'adresse email (avec vérification du format), et un mot de passe (d'au moins 6 caractères) **(figure 18)**. Un menu déroulant permet de

sélectionner un rôle (User, Admin ou Manager), tandis qu'un second menu optionnel permet d'assigner un utilisateur responsable. Des messages d'erreur s'affichent en temps réel si les champs ne respectent pas les critères requis **(figure 18)**. Après validation, le bouton "Create User" (qui devient "Creating..." pendant le traitement) soumet le formulaire. Un lien "Back to Login" permet de retourner à la page de connexion. Une illustration de l'application accompagne le formulaire sur le côté droit. (Note : Tous les champs marqués comme obligatoires doivent être remplis correctement pour finaliser l'inscription.)

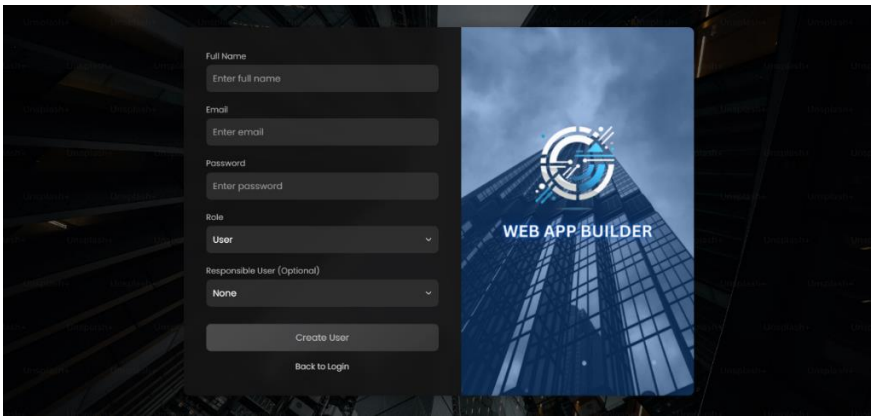


Figure 17 : Interface d'inscription

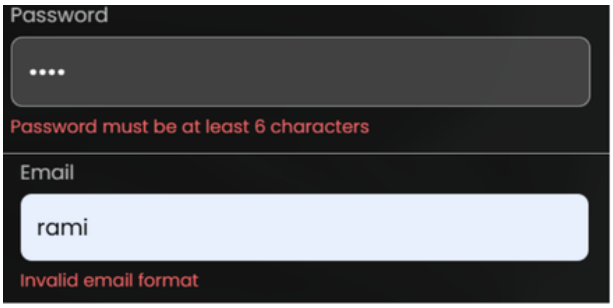


Figure 18 : Gestion des erreurs pour l'interface d'inscription

6.3. Module Tableau de bord

Interface tableau de bord

Le tableau de bord **(figure 19)** est l'interface principale qui s'affiche après la connexion. Il se compose d'une barre latérale de navigation et d'une zone de contenu principal. En haut, un en-tête affiche le titre "Dashboard" et un bouton "Create New Website" (visible uniquement pour les utilisateurs non-responsables). La section de profil est directement accessible pour modification.

Pour les responsables, une section "Users Management" permet de gérer les utilisateurs (visualisation, ajout via "Ajouter un utilisateur" ou suppression). Tous les

utilisateurs ont accès aux sections "My Projects" et "My Databases" qui listent leurs projets et bases de données respectifs (avec possibilité d'en ajouter via "Add New Project/Database" ou de consulter la liste complète via "View All").

Des fenêtres modales apparaissent pour chaque action de création (site web, projet, base de données ou utilisateur) ou modification de profil. (Note : Les options disponibles varient selon le rôle de l'utilisateur connecté.)

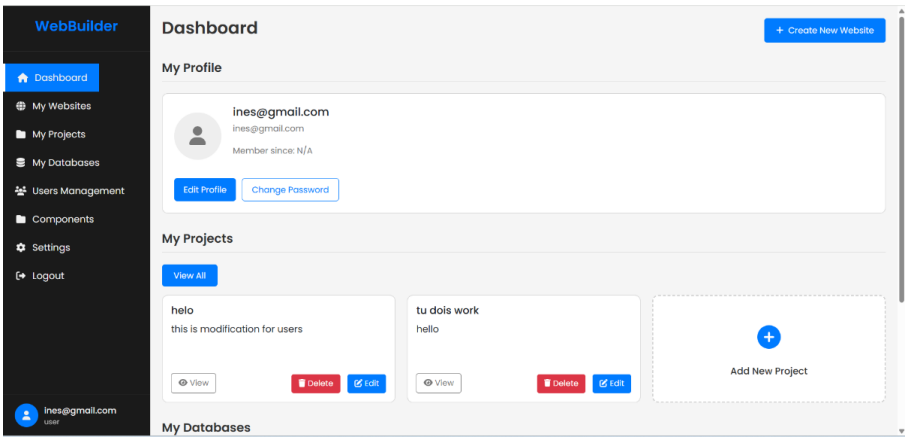


Figure 19 : Interface Dashboard

Interface de modification de profile

La fenêtre modale "Edit Profile" **(figure 20)**, accessible depuis le tableau de bord, permet aux utilisateurs de modifier leurs informations personnelles. Elle contient un formulaire avec trois champs : le nom complet (obligatoire), l'adresse email (obligatoire avec validation de format), et un sélecteur de fichier pour changer la photo de profil. Les boutons "Cancel" et "Save Changes" permettent respectivement de fermer la fenêtre sans enregistrer ou de sauvegarder les modifications.

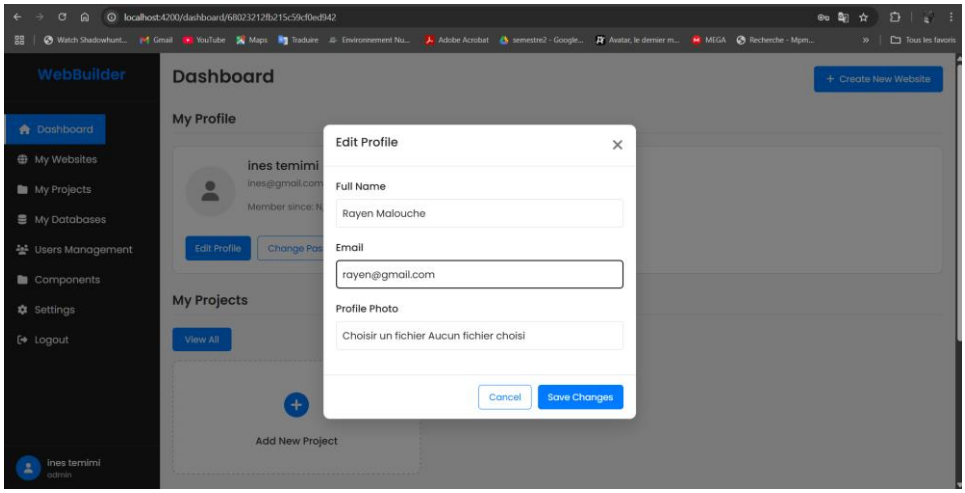


Figure 20 : Interface de modification de profile

6.4. Module Utilisateur

Interface de Gestion d'utilisateur

La page "Users" **(figure 21)** est une interface dédiée à la gestion des utilisateurs, accessible aux administrateurs et responsables. Elle affiche la liste complète des utilisateurs sous forme de cartes, chacune présentant les informations d'un utilisateur. Un champ de recherche en haut ("Search users...") permet de filtrer dynamiquement la liste **(figure 22)**. Le bouton "Create New User" ouvre une fenêtre modale pour ajouter un nouvel utilisateur. Chaque carte utilisateur inclut des options de suppression. (Note : Cette fonctionnalité est réservée aux rôles ayant les droits de gestion des utilisateurs.)

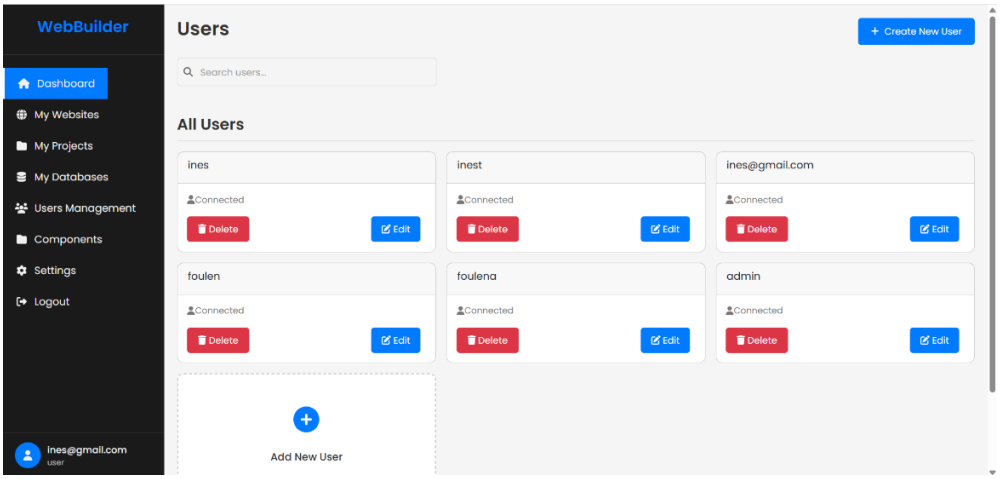


Figure 21 : Interface de gestion d'utilisateurs

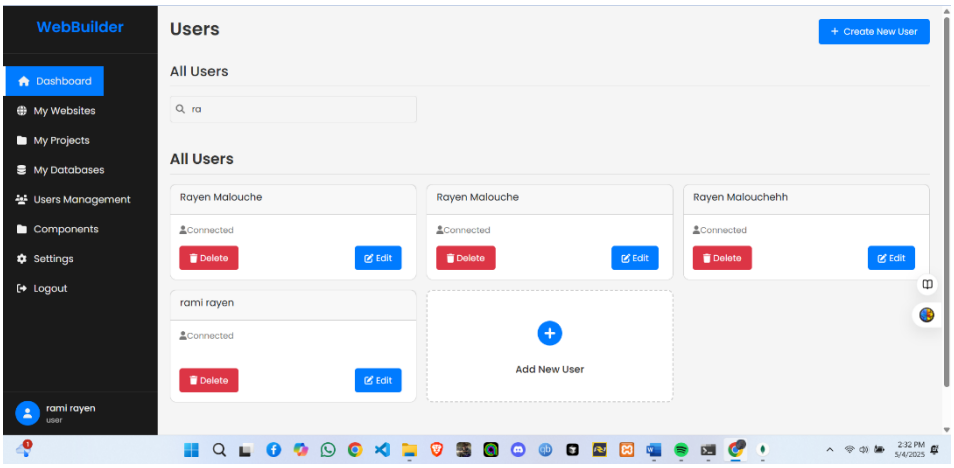


Figure 22 : Barre de recherche

Interface de création d'utilisateur

La fenêtre modale "Create New User" **(figure 23)** permet aux administrateurs d'ajouter de nouveaux utilisateurs au système. Elle contient un formulaire avec quatre

champs obligatoires : le nom complet, l'adresse email (avec validation automatique du format), un mot de passe, et un sélecteur de rôle (Admin ou User). Un bouton "Cancel" permet de fermer la fenêtre sans enregistrement, tandis que le bouton "Create User" valide et soumet le formulaire. (Note : Seuls les utilisateurs avec des privilèges d'administration peuvent accéder à cette fonctionnalité.)

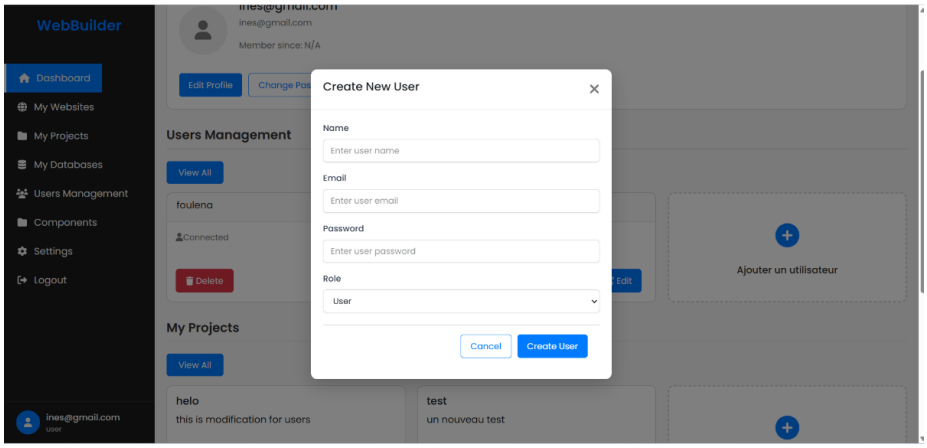


Figure 23 : Interface de création d'utilisateur

6.5. Module Projets

Interface de gestion de projets

La page "Projects" **(figure 24)** est l'interface centrale de gestion des projets. Elle présente une liste complète de tous les projets sous forme de cartes visuelles, avec une barre de recherche ("Search projects...") permettant de filtrer les résultats en temps réel. Un bouton "Create New Project" situé en haut à droite permet d'ajouter un nouveau projet via une fenêtre modale dédiée. Chaque carte de projet inclut des options de suppression. (Note : Cette page est accessible à tous les utilisateurs connectés, mais les fonctionnalités peuvent varier selon les permissions de chaque rôle.)

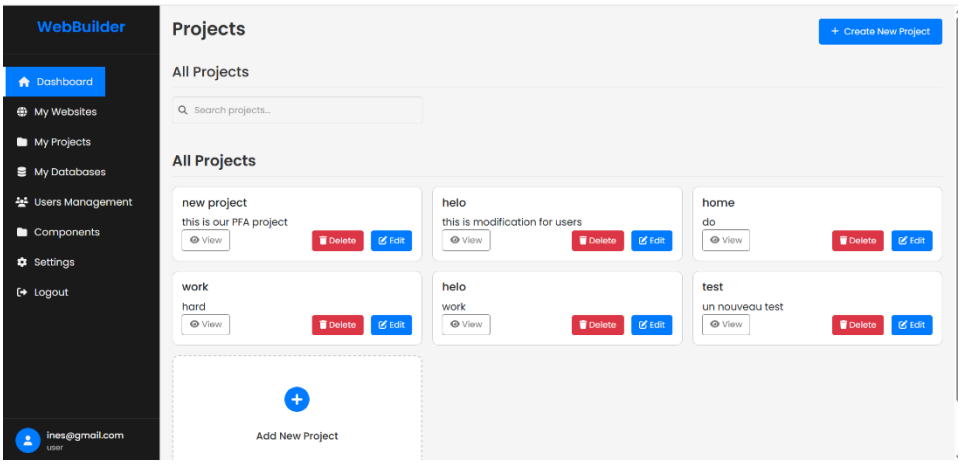


Figure 24 : Interface de gestion de projets

Interface de création de projet

La fenêtre modale "Create New Project" **(figure 25)** permet aux utilisateurs de créer un nouveau projet. Elle comprend un formulaire avec deux champs principaux : le nom du projet (obligatoire) et une description (optionnelle). Un bouton "Cancel" permet d'annuler la création, tandis que le bouton "Create Project" valide et enregistre le nouveau projet. (Note : Tous les utilisateurs authentifiés peuvent accéder à cette fonctionnalité pour créer leurs propres projets.)

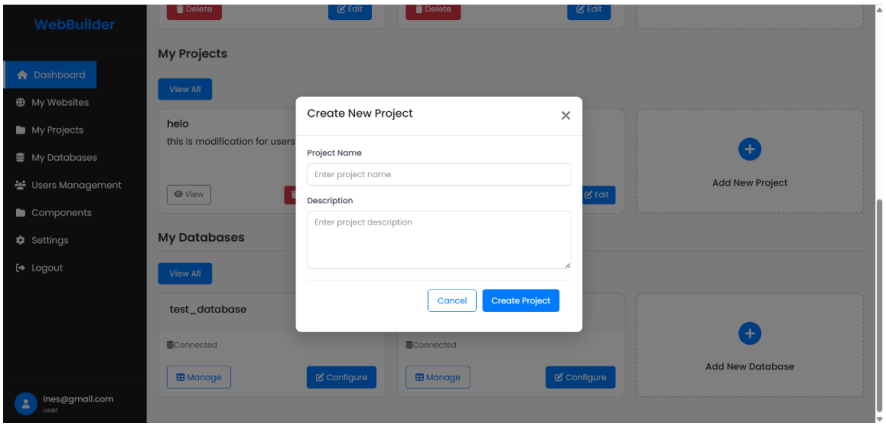


Figure 25 : Interface de création de projet

6.6. Module Sites

Interface de gestion de sites

La page "Web sites" **(figure 26)** est l'interface principale pour gérer les sites web créés par l'utilisateur. Elle affiche tous les sites sous forme de cartes organisées dans une grille, avec une barre de recherche en haut ("Search websites...") pour filtrer rapidement les résultats. Un bouton "Create New Website" permet d'ajouter un nouveau site via une fenêtre modale dédiée. Chaque carte de site web propose des

options d'édition et de suppression. (Note : Cette fonctionnalité est accessible à tous les utilisateurs authentifiés, avec des capacités de modification limitées aux sites qu'ils ont créés.)

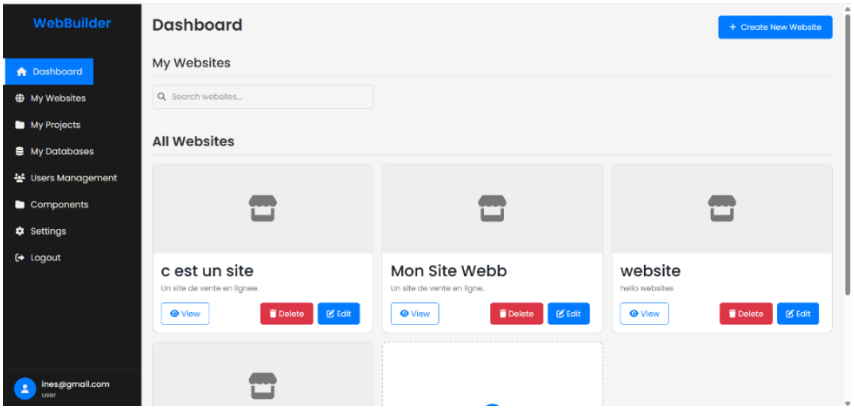


Figure 26 : Interface de gestion des sites

Interface de gestion de pages

La page "Website Pages" **(figure 27)** est l'interface de gestion des pages d'un site web. Elle présente une liste complète des pages existantes sous forme de cartes cliquables, avec un bouton "Add Page" en haut à droite permettant d'ajouter de nouvelles pages via une fenêtre modale. Chaque carte de page offre des options d'édition et de suppression. (Note : Cette interface est accessible aux utilisateurs autorisés à modifier la structure du site web.)

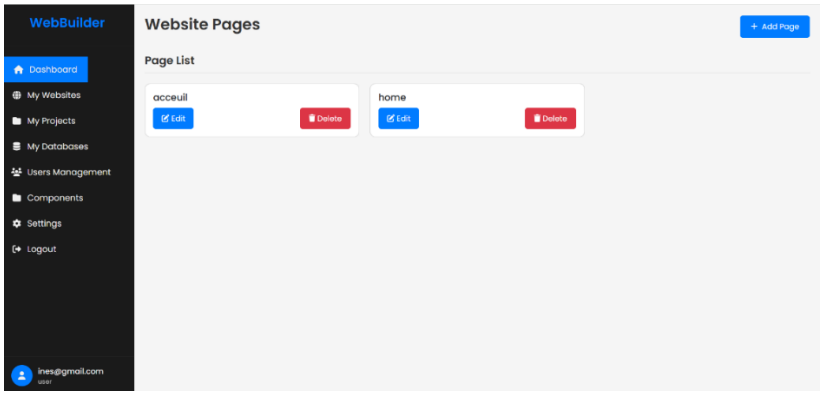


Figure 27 : Interface de gestion de pages

Interface de création de page

La fenêtre modale "Créer une nouvelle page" **(figure 28)** permet aux utilisateurs d'ajouter des pages à leur projet. Elle contient un formulaire minimaliste avec un seul champ obligatoire pour le nom de la page. Les boutons "Annuler" et "Créer la page" permettent respectivement de fermer la fenêtre sans sauvegarder ou de valider la création. (Note : Cette fonctionnalité est accessible lors de l'édition d'un projet existant.)

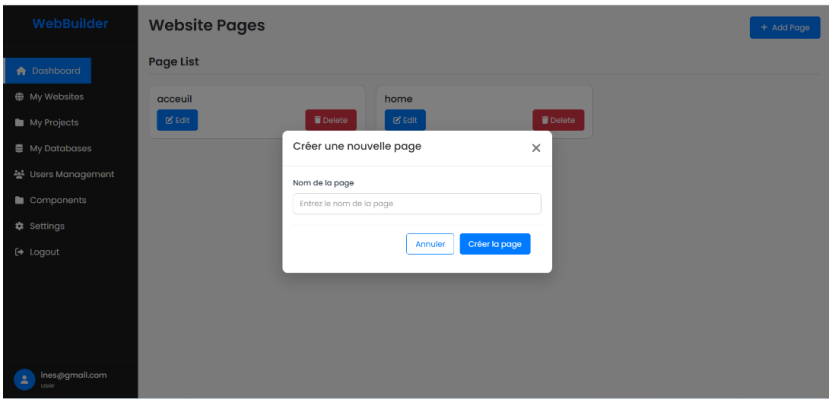


Figure 28 : Interface de création de page

6.7. Module Base de Données

Interface de création de base de données

La fenêtre modale "Create New Database" **(figure 29)** permet aux utilisateurs de configurer et créer une nouvelle base de données. Elle comprend un formulaire détaillé avec plusieurs champs : le nom de la base (obligatoire), l'hôte, le type (SQL, NoSQL, H2), le port, une description, ainsi que les identifiants de connexion (username et password). Trois boutons d'action sont disponibles : "Cancel" pour annuler, "Test Connection" pour vérifier la configuration, et "Create Database" pour finaliser la création.

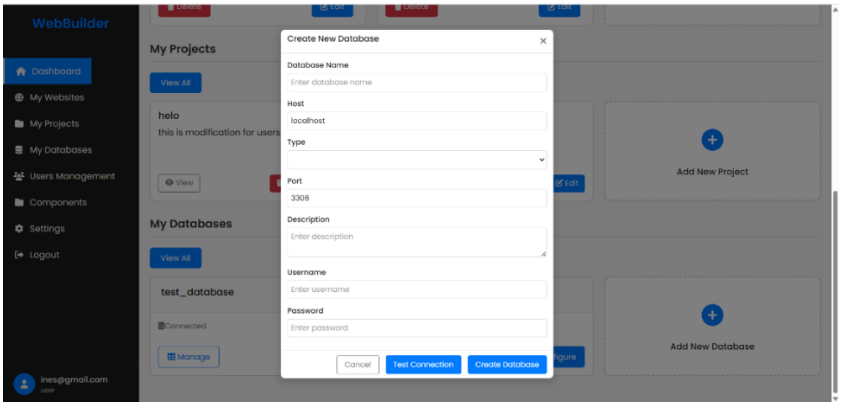


Figure 29 : Interface de création de base de données

Interface d'éditeur de base de données

L'interface "Database Management" **(figure 30)** est l'espace dédié à la gestion et à l'édition d'une base de données spécifique ("Products Database" dans cet exemple). Elle affiche en haut un panneau d'informations techniques (type MySQL, taille, nombre de tables, etc.) avec l'état de connexion. Un bouton "Create New Table" permet d'ajouter des tables via un composant dédié. La section principale liste les tables existantes avec des options d'édition et de suppression pour chacune.

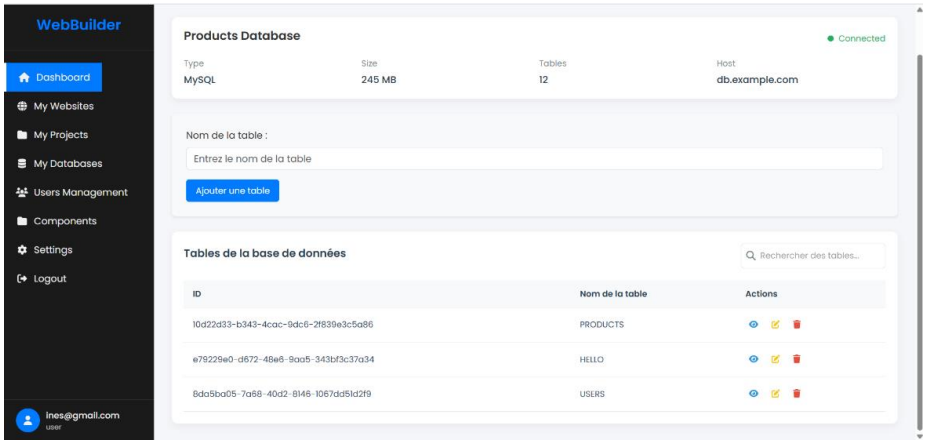


Figure 30 : Interface éditeur de base de données

6.8. Module Layout

L'interface "Layout" **(figure 31)** est un éditeur visuel permettant de créer des pages web via un système de glisser-déposer. Elle se compose de trois colonnes principales :

- Colonne de gauche : Liste des éléments (Containers, Text, Columns, Widgets, etc.) à glisser
- Zone centrale : Espace de travail où les éléments sont déposés et organisés hiérarchiquement (Rows > Columns > Content)
- Colonne de droite : Panneau de configuration avec onglets pour :
    - Propriétés du nœud sélectionné (nom, type, dimensions)
    - Paramètres d'apparence (couleur de fond)
    - Options globales du layout

Fonctionnalités clés :

- Mode Preview/Edit pour basculer entre édition et visualisation
- Boutons Undo/Redo pour annuler/rétablir les actions
- Bouton "Clear" pour réinitialiser la composition
- Bouton "Save" générant un fichier HTML téléchargeable contenant exactement la structure créée **(figure 32)**

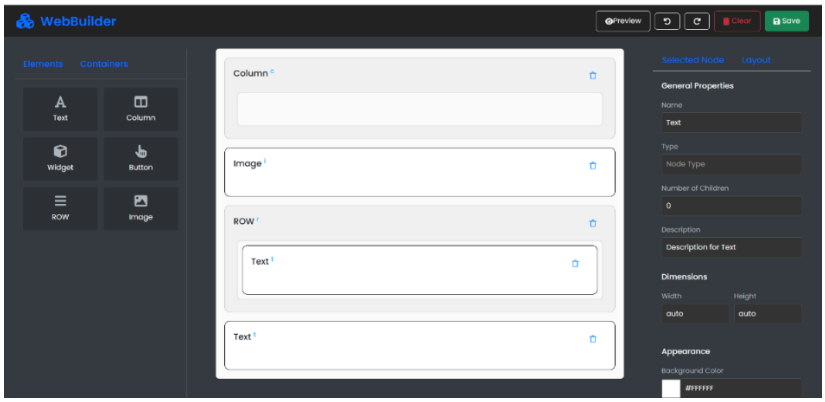


Figure 31 : Interface layout

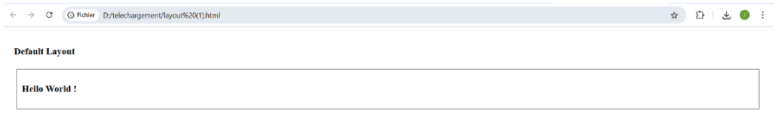


Figure 32 : Exemple d'un fichier html téléchargé localement et lancé sur le web

7. Exemple de scénario complet

La **figure 33** illustre un exemple type du processus de création d'un site web à partir de l'application Web App Builder :

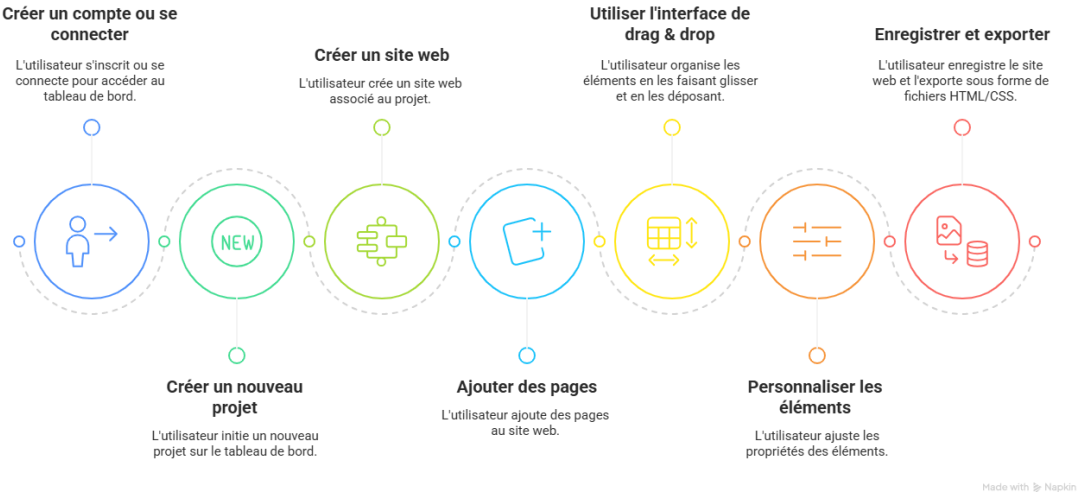


Figure 33 : Processus de création de site web

Conclusion

Ce chapitre a présenté la mise en œuvre du projet, en détaillant l'environnement technique utilisé, les technologies adoptées, ainsi que les différentes interfaces de l'application. L'ensemble permet d'illustrer concrètement les fonctionnalités développées et leur intégration dans un scénario d'utilisation complet.

Conclusion Générale

Ce projet de création d'une plateforme de développement d'applications web dynamiques a permis de démontrer qu'il est possible de concilier simplicité d'utilisation et puissance technique, tout en répondant aux besoins croissants de flexibilité et de personnalisation des utilisateurs. L'intégration d'une interface de type glisser-déposer et d'un moteur de gestion de données sur mesure a permis de surmonter les limitations des solutions no-code existantes, offrant ainsi une expérience plus complète et adaptable aux exigences variées des utilisateurs.

Tout au long de ce projet, nous avons pris soin d'identifier les besoins spécifiques de nos utilisateurs cibles et de mettre en œuvre des solutions techniques innovantes pour y répondre. L'approche adoptée, visant à offrir un outil à la fois accessible et puissant, a permis de lever les barrières techniques souvent rencontrées par les non-développeurs, tout en garantissant une performance optimale et une sécurité accrue.

En ce qui concerne les perspectives, plusieurs axes d'amélioration et d'enrichissement de la plateforme peuvent être envisagés. Parmi ceux-ci, on pourrait explorer l'ajout de fonctionnalités de collaboration en temps réel, l'intégration d'outils d’analyse de données avancés ou encore l'élargissement de la personnalisation à travers l'usage de plugins tiers. L'extension de la plateforme à d'autres types de projets numériques, tels que les applications mobiles ou les systèmes embarqués, pourrait également ouvrir de nouvelles perspectives.

En somme, ce projet représente une étape importante dans la démocratisation du développement web, en rendant accessible à un plus grand nombre d'utilisateurs la possibilité de créer des applications puissantes et sur-mesure, tout en respectant les exigences modernes de flexibilité, de rapidité et de sécurité.

Références

[1] Site officiel de JetBrains, «Getting started with WebStorm,» JetBrains, [En ligne]. Available: https://www.jetbrains.com/help/webstorm/getting-started-with-webstorm.html. [Accès le 12 Avril 2025].

[2] Site officiel de JetBrains IntelliJ IDEA, «IntelliJ IDEA overview,» JetBrains, [En ligne]. Available: https://www.jetbrains.com/help/idea/discover-intellij-idea.html. [Accès le 12 Avril 2025].

[3] Site officiel de MongoDB Compass, «What is MongoDB?,» MongoDB, Inc, [En ligne]. Available: https://www.mongodb.com/docs/manual/. [Accès le 12 Avril 2025].

[4] Site officiel de Postman, «Postman documentation overview,» Postman, [En ligne]. Available: https://learning.postman.com/docs/introduction/overview/. [Accès le 12 Avril 2025].

[5] G. (. A. G. G. Docs, «À propos de GitHub et Git,» GitHub, [En ligne]. Available: https://docs.github.com/fr/get-started/start-your-journey/about-github-and-git. [Accès le 12 Avril 2025].

[6] A. (. I. t. A. A. O. Documentation., «What is angular ?,» Angular, [En ligne]. Available: https://v17.angular.io/guide/what-is-angular. [Accès le 12 Avril 2025].

[7] S. (. S. B. R. Documentation, «Qu'est-ce que Java Spring Boot ?,» IBM, [En ligne]. Available: https://www.ibm.com/fr-fr/topics/java-spring-boot. [Accès le 12 Avril 2025].

[8] I. (. I. t. M. M. D. MongoDB, «MongoDB : tout savoir sur la base de données NoSQL orientée document,» DataScientest, [En ligne]. Available: https://datascientest.com/mongodb. [Accès le 12 Avril 2025].

[9] S. L. (. S. O. S. Documentation., «What is Snyk?,» snyk, [En ligne]. Available: https://docs.snyk.io/what-is-snyk. [Accès le 12 Avril 2025].

[10] Syloé, «Qu’est-ce que SonarQube ?,» Syloé, [En ligne]. Available: https://www.syloe.com/glossaire/sonarqube/. [Accès le 12 Avril 2025].

[11] S. &. S. B. (. P. G. (. e. A. Chacon, «Getting Started - What is Git?,» [En ligne]. Available: https://git-scm.com/book/en/v2/Getting-Started-What-is-Git%3F. [Accès le 12 Avril 2025].

[12] B. C. (. B. Documentation, «Get started with Bootstrap,» Bootstrap, [En ligne]. Available: https://getbootstrap.com/docs/5.2/getting-started/introduction/. [Accès le 12 Avril 2025].

Résumé

Dans le cadre de notre Projet de Fin d'Année à l'**ENSIT**, nous avons conçu et développé une application web innovante qui permet de créer des sites dynamiques sans compétences en programmation. Grâce à une interface intuitive basée sur le **glisser-déposer**, notre plateforme offre aux utilisateurs la possibilité de structurer leurs pages, de définir des bases de données personnalisées, de gérer le contenu et de visualiser leur site en temps réel. Ce projet vise à faciliter la création de solutions web sur mesure tout en restant accessible, flexible et évolutif.

Mots clés : création de site web, glisser-déposer, no-code, base de données, visualisation en temps réel.

Abstract

As part of our Final Year Project at **ENSIT**, we designed and developed an innovative web application that enables users to build dynamic websites without any coding skills. Using a drag-and-drop interface, our platform allows users to structure pages, define customizable databases, manage content, and preview their website in real time. The project's main goal is to simplify custom web development while ensuring accessibility, flexibility, and scalability.

Keywords: website builder, drag-and-drop, no-code, database management, real-time preview.

الملخص

كجزء من مشروع نهاية العام لدينا في المدرسة الوطنية العليا للمهندسين بتونس، قمنا بتصميم وتطوير تطبيق ويب مبتكر يسمح لك بإنشاء مواقع ديناميكية دون مهارات البرمجة. من خلال واجهة السحب والإفلات البديهية، توفر منصتنا للمستخدمين القدرة على هيكلة صفحاتهم، وتحديد قواعد البيانات المخصصة، وإدارة المحتوى، وتصور موقعهم في الوقت الحقيقي. يهدف هذا المشروع إلى تسهيل إنشاء حلول ويب مصممة خصيصًا مع الحفاظ على إمكانية الوصول إليها ومرنتها وقابليتها للتطوير.

الكلمات المفاتيح :إنشاء مواقع، بدون كود، سحب وإفلات، قواعد بيانات، معاينة فورية، المدرسة الوطنية العليا للمهندسين بتونس.