



FACULTÉ DES SCIENCES ET TECHNIQUES

.NET

COMPTES-RENDU

RAPPORT

Conception et développement d'une application web de gestion des blogs

Étudiants :

Bachir Kassoum Ahmadou

Seba Tarek

Luc Perin Panta Pameni

Ichraq Elaidi

Enseignant :

Gauthier CABOT

FARIN Chris

TABLE DES MATIÈRES

Introduction Générale	3
1 Étude générale du projet et ses besoins	4
1.1 Présentation générale du projet	5
1.2 Solution adoptée	5
1.3 Étude fonctionnelle	5
1.3.1 Besoins fonctionnels	5
1.3.2 Besoins non-fonctionnels	5
1.4 Conduite de projet	5
1.4.1 Planification et Répartition du travail	6
1.5 Conclusion	6
2 Analyse et conception	7
2.1 Identification des acteurs et leurs rôles	8
2.2 Diagramme de cas d'utilisation	8
2.3 Diagramme de séquence	9
2.3.1 Système de connexion	9
2.4 Diagramme de classes	10
2.5 Conclusion	11
3 Étude technique	12
3.1 Besoins techniques	13
3.2 Architecture logicielle	13
3.2.1 Pourquoi l'architecture logicielle MVC	14
3.3 Outils et technologies	14
3.4 .NET 6	14
3.5 Tailwind CSS	15
3.6 Sql Server	15
3.7 Star UML	15
3.8 Conclusion	15

4	Réalisation et mise en oeuvre	16
4.1	L'inscription	17
4.2	L'authentification	17
4.3	Ajouter un blog	17
4.4	Ajouter des posts	17
4.5	Ajouter des commentaires	18
4.6	Lister les posts	18
4.7	Détails des posts	19
4.8	Difficultés rencontrées	19
4.8.1	Le manque de temps	19
4.8.2	Manque d'expérience en .NET	19
4.8.3	Configuration de tailwindCSS	19
4.9	Conclusion	19

INTRODUCTION GÉNÉRALE

Dans le contexte du master 2 GIL, le module de .NET s'implique en continu pour l'amélioration des niveaux des étudiants. S'inscrivant dans cette optique, notre projet qui vise la mise en place d'un système de générateur de "blogs" tout en respectant le cahier des charges fourni par le client.

Ce rapport retrace le travail réalisé, et il s'articule en quatre chapitres :

Le premier chapitre consiste à aborder de façon générale le projet, définir la problématique, les objectifs et les fonctionnalités du projet ainsi que la répartition du travail.

Le deuxième chapitre, qui présente une étape primordiale, consiste à analyser l'application en spécifiant les besoins fonctionnels et non fonctionnels, et faire la conception et la modélisation.

Le troisième chapitre est consacré à l'étude technique et la présentation de l'environnement logiciel utilisé pour la réalisation de l'application.

Le quatrième chapitre est consacré à une illustration graphique de l'application ainsi que les difficultés rencontrée.

Le dernier chapitre clôture le rapport.

CHAPITRE

1

ÉTUDE GÉNÉRALE DU PROJET ET SES BESOINS

Ce chapitre décrit l'environnement dans lequel le présent projet a été initié, afin d'avoir une meilleure compréhension de ce dernier. Ainsi on va présenter une description générale du projet puis notre démarche.

1.1 Présentation générale du projet

Aujourd'hui, l'accent est mis sur le soutien au partage des connaissances et à l'interaction productive. Compte tenu du développement du télétravail, l'importance des auto-formations, des processus de partage des connaissances efficaces revêtent plus d'importance que jamais. Comme définit dans le cahier des charges, la mission principale de notre application est d'assurer à l'utilisateur d'avoir un espace 'privé' pour poster du contenu. Dans ce sens l'utilisateur doit avoir la possibilité de gérer son espace.

1.2 Solution adoptée

Pour profiter et bien récolter les bénéfices de ce partage d'informations notre application permet :

- *Au superviseur : de créer un blog, d'assigner le blog à une personne, d'ajouter des postes, d'upload des images, de gérer la visibilité des blog.

- *Aux administrateurs : de gérer leurs blogs et créer des posts.

- *Aux Membres : Accéder aux blogs/posts publics et privés.

- *Aux Visiteurs : d'accéder aux blogs/posts publics, d'effectuer une inscription.

1.3 Étude fonctionnelle

Cette partie présente les aspects fonctionnels du système à développer, et décrit toutes les fonctionnalités que l'application doit assurer. Un diagramme de cas d'utilisation sera présent afin d'exprimer les besoins des utilisateurs.

1.3.1 Besoins fonctionnels

l'application permet de :

- * Ajouter des utilisateurs ;

- * Importer les images ;

- * Ajouter des blogs ;

- * Générer les posts ;

1.3.2 Besoins non-fonctionnels

on présente ci-dessous l'ensemble des contraintes à respecter pour garantir la performance du système.

Performance : L'application à développer doit être puissante à travers ses fonctionnalités et doit répondre de manière optimale à toutes les exigences des utilisateurs.

Sécurité : Les comptes des utilisateurs sont sécurisés par mot de passe haché et chaque utilisateur ne peut avoir qu'un seul compte créé.

Fiabilité : Bon fonctionnement de l'application web sans détection de défaillance.

Ergonomie : L'application doit être simple et conviviale en implémentant les conceptions de l'expérience utilisateur.

1.4 Conduite de projet

La réalisation d'une application débute lorsqu'un besoin est exprimé qui justifie sa création et se termine quand elle est mise en service. Entre temps, on doit passer par plusieurs phases permettant de rationaliser les différentes étapes qui interviendront tout au long du processus de développement.

Les trois phases résument le processus de développement, à savoir :

- la phase définition : qui permet de collecter les besoins, puis comment et sous quelles conditions sera réalisée l'application.
- la phase de développement : qui consiste à transformer les données collectées pendant la phase de définition en langage informatique.
- la phase support : qui permet d'effectuer les opérations de correction, d'amélioration et d'évolution.

Les différentes étapes du processus de développement sont un ensemble de phase appelé le cycle de vie de développement. L'objectif du cycle de vie à travers les différentes phases est de réaliser un livrable de qualité en détectant les erreurs au plus tôt durant le processus de développement.

1.4.1 Planification et Répartition du travail

Pour assurer le bon déroulement du projet, la phase de planification est une étape très importante permettant d'avoir une vision globale sur les étapes et phases de développement du projet, commençant par l'expression du besoin jusqu'au tests de l'application et le déploiement pour les clients.

Diagramme de Gantt

Gantt est un outil qui permet de planifier le projet et de rendre plus simple le suivi de son avancement en visualisant l'enchaînement et la durée des différentes tâches. Le planning de notre projet est représenté sous la forme d'un diagramme de Gantt comme la montre la figure ci-dessous.

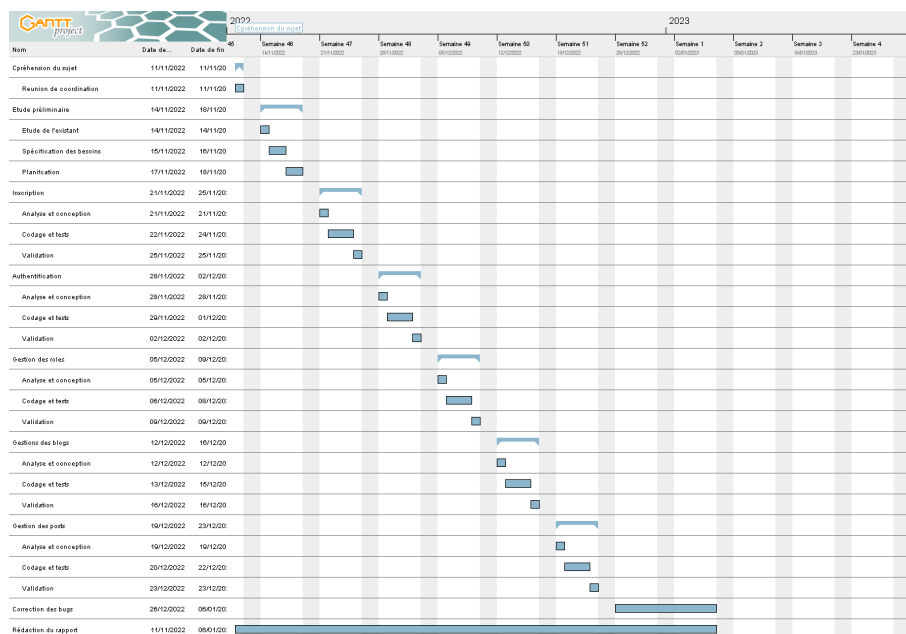


FIGURE 1.1 – Diagramme de Gantt

1.5 Conclusion

Au travers ce chapitre, on a fait un bref tour sur l'environnement dans lequel notre projet s'est déroulé, en présentant brièvement le sujet ainsi que ses diverses étapes projetées de façon chronologique afin de mener à bien le projet. Dans le chapitre suivant, on va présenter la conception du projet qui est une phase indispensable.

CHAPITRE

2

ANALYSE ET CONCEPTION

Ce chapitre est dédié à l'aspect conceptuel de ce présent projet. Il présente alors une conception préliminaire pour passer à la modélisation du système à l'aide des diagrammes UML.

2.1 Identification des acteurs et leurs rôles

Dans le cadre de ce projet, les acteurs qui interagissent avec le système et leurs rôles sont ainsi détaillés ci-après.

Acteurs	Rôles
Au superviseur	Créer un blog
	Assigner le blog à une personne
	Gérer la visibilité de son blog
	Effectuer n'importe quelle fonctionnalité
Au Administrateur	Gérer son blog
	Ajouter des posts
Membre	Accéder aux blogs publics et privés
	Ajouter des commentaires
Visiteur	Accéder aux blogs publics
	Effectuer une inscription.

2.2 Diagramme de cas d'utilisation

Avant tout développement, il convient de répondre à la question :

“A quoi va servir le logiciel ?”

En UML, on établit des Diagrammes de Cas d'Utilisation pour répondre à cette question.

Un cas d'utilisation modélise un service rendu par le système. Il exprime les interactions acteurs/système et apporte une valeur ajoutée notable à l'acteur concerné.

Pour chaque acteur identifié dans la partie précédente, nous rechercherons les différents cas d'utilisation selon lesquels il utilise le système. La figure suivante montre un diagramme de cas d'utilisation du système.

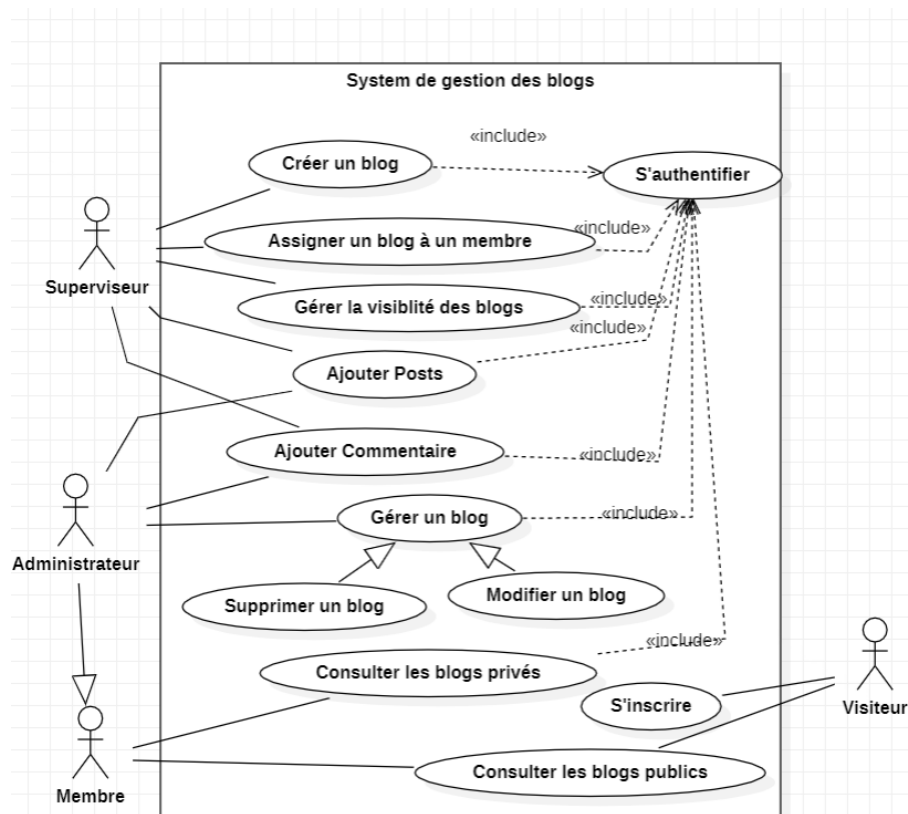


FIGURE 2.1 – Diagramme de cas d'utilisation

2.3 Diagramme de séquence

2.3.1 Système de connexion

dans ce diagramme l'email est représenté par matricule.

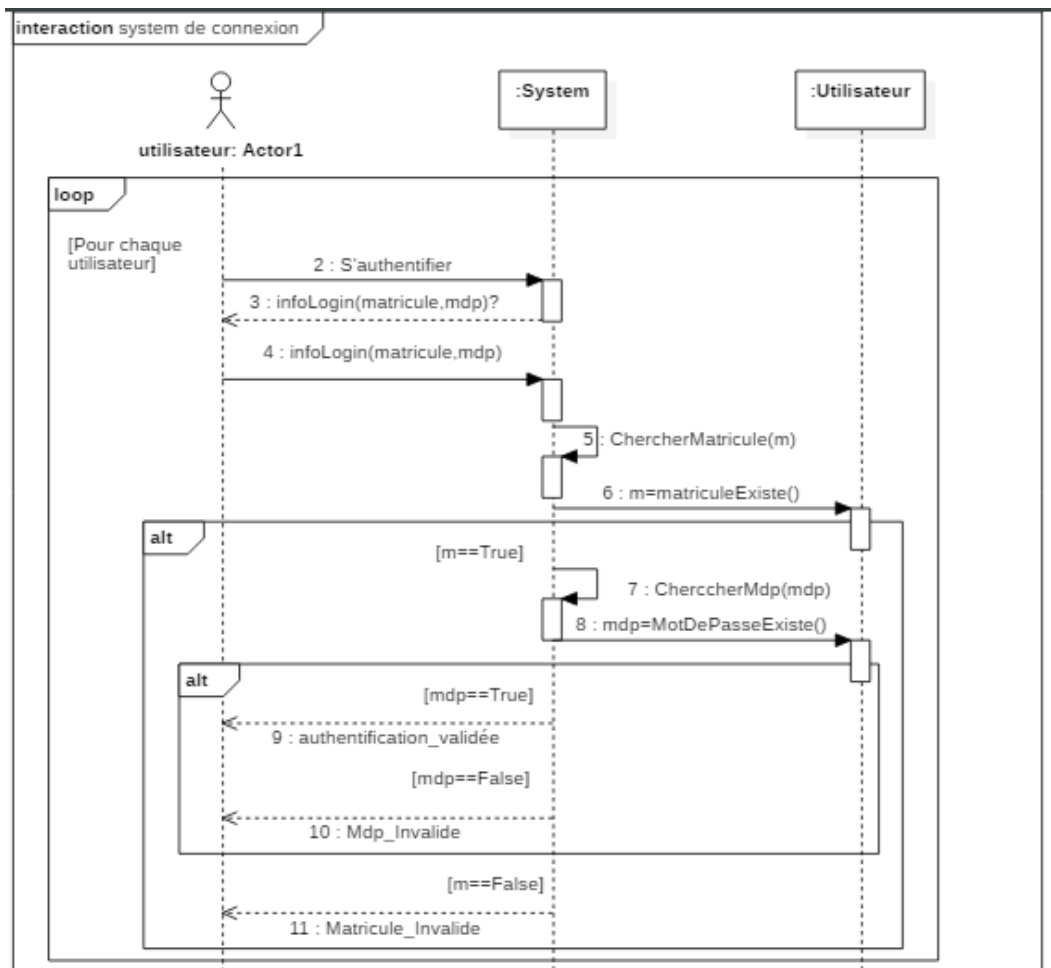


FIGURE 2.2 – Diagramme de séquence/S'authentifier

2.4 Diagramme de classes

Les diagrammes de classes sont l'un des types de diagrammes les plus utiles en langage UML, ils jouent un rôle central dans le design orientés objet. Ils fournissent une représentation graphique des classes, associations, héritages et dépendances qui composent le système ce qui décrit clairement la structure statique du système. La figure suivante représente le diagramme de classes global du projet :

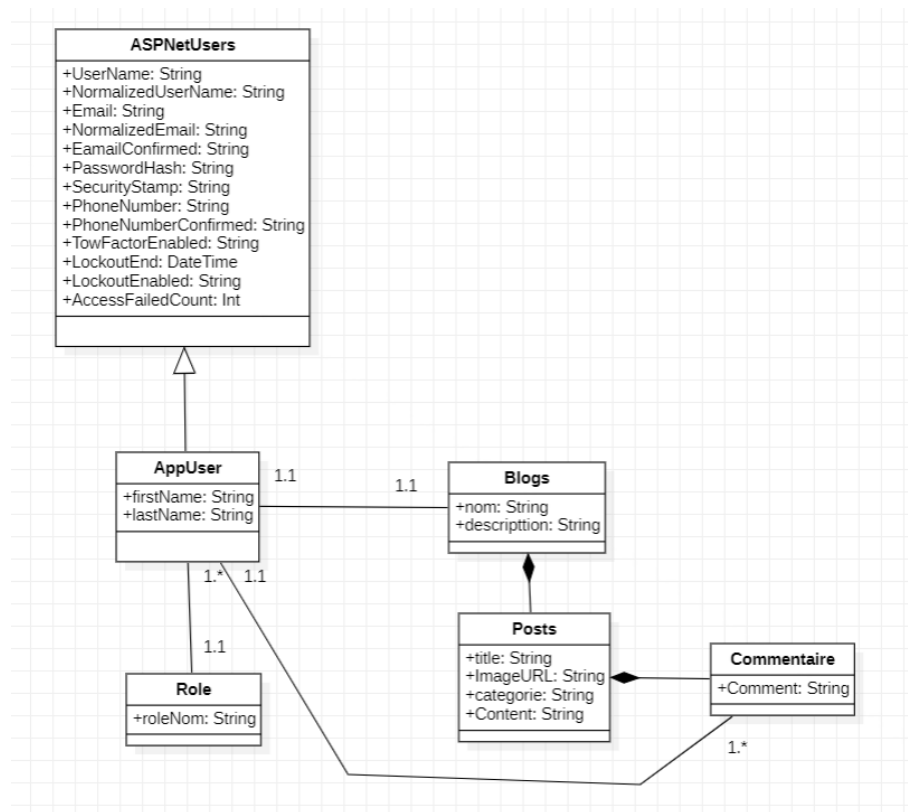


FIGURE 2.3 – Diagramme de classe

2.5 Conclusion

Ce chapitre était alors une présentation de vue conceptuelle de la solution à mettre en place. Ainsi, un ensemble de diagrammes UML ont été présentés dans ce chapitre. Le chapitre suivant présente alors l'étude technique du projet.

CHAPITRE

3

ÉTUDE TECHNIQUE

L'étude technique consiste à faire une analyse des besoins et contraintes techniques. Ce chapitre présente alors une capture des besoins techniques, puis les spécifications logicielles de ce présent projet et le choix de technologie de développement.

3.1 Besoins techniques

La capture des besoins techniques liste les spécifications auxquelles l'application doit supporter, en faisant abstraction des besoins fonctionnels présenté précédemment.

Exigence	Description
Facile à connecter	l'utilisateur se connecte à l'application facilement.
Maintenable	L'application doit être facile à maintenir.
Disponible	L'application doit être toujours disponible pour tous les utilisateurs.
Sécurisée	L'accès aux données doit être sécurisé.

3.2 Architecture logicielle

L'architecture MVC est l'une des architectures logicielles les plus utilisées pour les applications Web, elle se compose de 3 modules :

- * Modèle : noyau de l'application qui gère les données, permet de récupérer les informations dans la base de données, de les organiser pour qu'elles puissent ensuite être traitées par le contrôleur.
- * Vue : composant graphique de l'interface qui permet de présenter les données du modèle à l'utilisateur.
- * Contrôleur : composant responsable des prises de décision, gère la logique du code qui prend des décisions, il est l'intermédiaire entre le modèle et la vue.

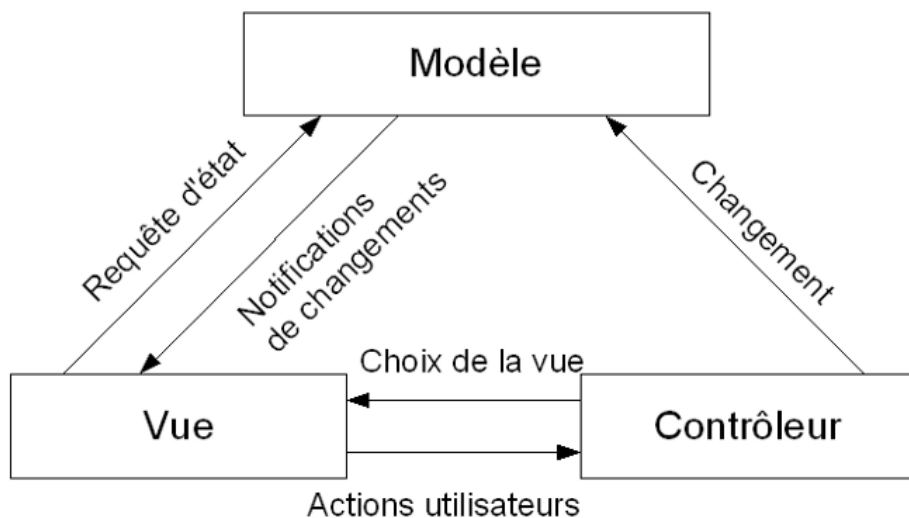


FIGURE 3.1 – Architecture logicielle MVC

Le contrôleur va demander au modèle les données, les analyser, prendre des décisions et renvoyer le texte à afficher à la vue.

Architecture

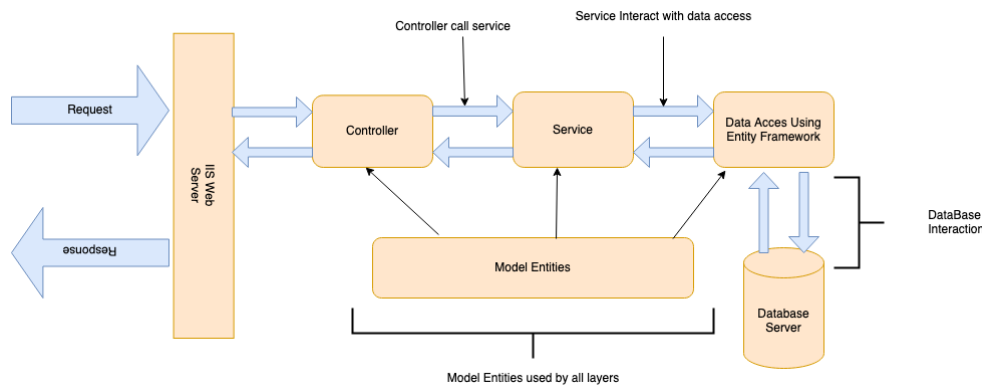


FIGURE 3.2 – Architecture

3.2.1 Pourquoi l'architecture logicielle MVC

- * Meilleure organisation du code ;
- * Diminution de la complexité lors de la conception ;
- * Conception claire et efficace grâce à la séparation des données de la vue et du contrôleur ;
- * Possibilité de réutilisation de code dans d'autres applications ;
- * Un gain de temps de maintenance et d'évolution du site ;
- * Une plus grande souplesse pour organiser le développement du site entre différents développeurs ;

3.3 Outils et technologies

3.4 .NET 6

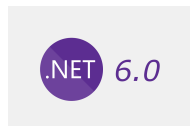


FIGURE 3.3 – Logo NET 6

.NET 6 fournit les dernières parties du plan d'unification .NET . .NET 6 unifie le Kit de développement logiciel (SDK), les bibliothèques de base et le runtime sur les applications mobiles, de bureau, IoT et cloud. En plus de cette unification, l'écosystème .NET 6 offre :

Développement simplifié : la prise en main est simple. Les nouvelles fonctionnalités de langage en C 10 réduisent la quantité de code que vous devez écrire. Et les investissements dans la pile web et les API minimales facilitent l'écriture rapide de microservices plus petits et plus rapides.

Meilleures performances : .NET 6 est le framework web de pile complète le plus rapide, ce qui réduit les coûts de calcul si vous exécutez dans le cloud.

Productivité ultime : .NET 6 et Visual Studio fournissent un rechargement à chaud, de nouveaux outils Git, une modification intelligente du code, des diagnostics robustes et des outils de test, et une meilleure collaboration d'équipe.

3.5 Tailwind CSS



FIGURE 3.4 – Tailwind CSS

Tailwind CSS est un framework CSS open source. La principale caractéristique de cette bibliothèque est que, contrairement à d'autres frameworks CSS comme Bootstrap, elle ne fournit pas une série de classes prédéfinies pour des éléments tels que des boutons ou des tableaux. Au lieu de cela, il crée une liste de classes CSS « utility » qui peuvent être utilisées pour styliser chaque élément en mélangeant et en faisant correspondre.

3.6 Sql Server



FIGURE 3.5 – Sql Server

Microsoft SQL Server est une plate-forme de données d'entreprise permettant de gérer et stocker dans des bases de données tout type d'information. SQL Server intègre par défaut des outils de gestion, d'administration et de développement de bases de données. Déploiement par un setup, mise en oeuvre et administration par des interfaces graphiques intuitives.

3.7 Star UML



FIGURE 3.6 – Logo StarUML

StarUML est un logiciel de modélisation UML(Unified Modeling Language) disponible en OpenSource. Via cette plateforme, on peut concevoir une dizaine de types de diagrammes. Il est notamment possible de créer des classes, des objets, des activités ou bien des séquences compatibles avec le standard UML.

3.8 Conclusion

Ce chapitre présente une étude technique, en commençant par la présentation des exigences techniques, et passant par une présentation de l'architecture MVC du système et finalement les outils et les technologies utilisés.

CHAPITRE

4

RÉALISATION ET MISE EN OEUVRE

Dans ce présent chapitre on va présenter la phase de réalisation de ce système en présentant des captures d'écrans des scénarios d'utilisation de l'application réalisée.

4.1 L'inscription

Elle permet de créer un compte personnel qui est lié à un nom d'utilisateur et un mot de passe choisis par l'utilisateur pour profiter pleinement des fonctionnalités offertes par ce site.

4.2 L'authentification

Pour accéder à l'application, chaque utilisateur doit se connecter avec son email et son mot de passe.

FIGURE 4.1 – authentification

4.3 Ajouter un blog

Seul l'admin qui peut ajouter des blogs et les affecter aux membres pour devenir des administrateurs du blogs.

FIGURE 4.2 – Ajouter un blog

4.4 Ajouter des posts

Un administrateur d'un blog peut ajouter des posts avec une image et si il souhaite publier le post immédiatement, il change juste la visibilité/status de ce dernier . Sinon, il peut le laisser en brouillon" pour le publier plus tard.

Title *

L'URL de l'image *

Categorie *

Contenu

Status

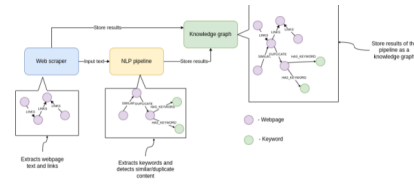
DRAFT

Enregistrer

FIGURE 4.3 – Ajouter un post

4.5 Ajouter des commentaires

Un membre peut ajouter des commentaire aux posts.



Data collection and modeling workflow. Image by the author.
The data collection and preprocessing consist of three parts.

Commentaires

Envoyer

Thank you for sharing your detailed and informative blog post on optimizing a website using web scraping and natural language processing techniques. Your walkthrough of the process, including the provided code examples

Pameli Panta
Joined in August 2014

FIGURE 4.4 – Ajouter un commentaire

4.6 Lister les posts


Un membre/un visiteur peut voir les posts.

Postcrafts Accueil Mes Posts luc-perin.panta-pameni@univ-rouen.fr Se deconnecter


Lire nos derniers Articles

Create custom landing pages with Rareblocks that converts more visitors than any website.


Voir tous les articles →



Software development-07 janvier 2023
Serverless In 2023 — A Shift In Focus



Software development-07 janvier 2023
Never Worry About Cold Starts in AWS Lambda Again



Software development-07 janvier 2023
UX developer and software developer at the

FIGURE 4.5 – Lister les posts

4.7 Détails des posts

Un membre/un visiteur peut voir aussi les détails des posts.

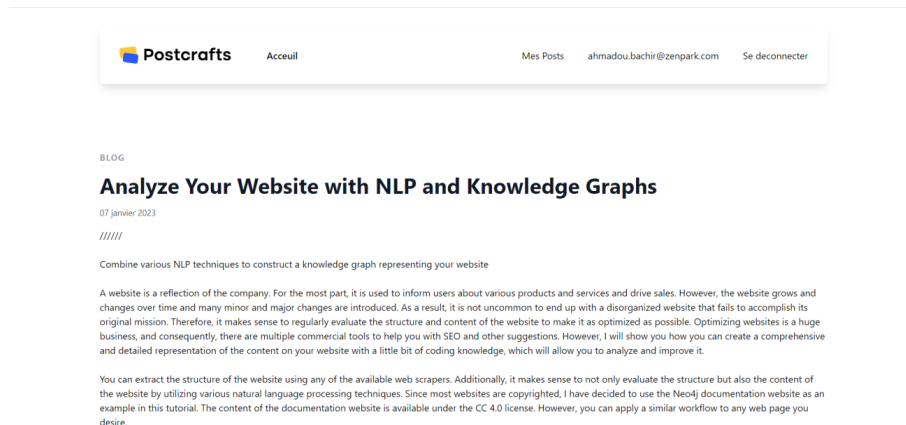


FIGURE 4.6 – détails d’un post

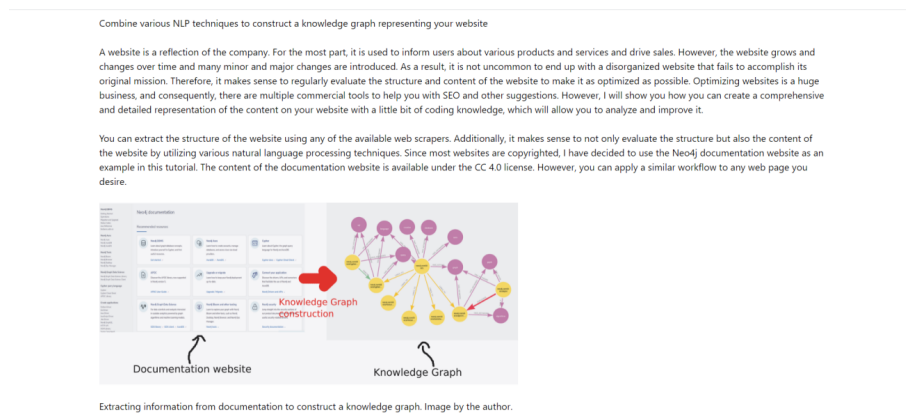


FIGURE 4.7 – détails d’un post

4.8 Difficultés rencontrées

4.8.1 Le manque de temps

Tous les membres de l’équipe sont des alternants, on était obligé à articuler notre temps entre sa charge de travail quotidienne (alternance et autres projets des autres modules) et la charge « projet .NET »... une gestion rigoureuse de nos plans de charge est alors utile.

4.8.2 Manque d’expérience en .NET

Des formations et tutoriels ont été suivis par certains membres de l’équipe.

4.8.3 Configuration de tailwindCSS

Pour intégrer tailwindCSS dans l’application ASP.NET on a pris du temps et pour que cela marche.

4.9 Conclusion

Dans ce dernier chapitre, on a présenté l’application en expliquant quelques captures d’écran.