

**Mise en place d’une plateforme de reseau social headless**

Rapport



**25 novembre 2024**

**SUPINFO**

**85 Boulevard Marius Vivier Merle 69003 Lyon**

Membres du groupe

* Hamza BELYAHIOUI
* Fadel BIAOU
* Abdoul-waris KONATE

Table des matières

Liste des figures

[Figure 1:Archtitecture Azure SociApp. 6](#_Toc183514858)

Liste des Sigles et abréviations

|  |  |
| --- | --- |
| Sigles | Définition |
| IAAS | Infrastructure As A Service |
| IDE | Environnement de développement Intégré |
| PAAS | Platform As A Service |
| SAAS | Software As A Service |

# **Résumé**

La démocratisation de l’internet a transformé le monde en village planétaire facilitant ainsi le partage et la communication entre personne proche ou distante. Cependant, cela n’est possible que par des plateformes dédiées sur lesquelles des personnes partagent leur contenu les uns aux autres et plusieurs méthodes de conception et de déploiement existent pour réaliser de telles plateformes.

Pour ce projet, nous avons eu à développer une plateforme de réseau social basé sur le cloud Azure pour une start-up nommé Link Up qui se veut être un système backend évolutif et efficace, qui peut être intégré à n’importe quelle interface frontale, offrant ainsi une expérience de gestion de contenu transparente.

Pour la réalisation de cette plateforme, on a utilisé **Visual Studio** un environnement de developpement sur lequel nous avons implementer notre solution, **GITHUB** pour le versionning et le travail collaboratif , **Draw.io** pour la modelisation de notre solution, du **.Net** pour sa resistantce et aussi son integration rapide avec azure et pour le deploiement nous avons utilisé **Microsoft Azure.**

Le present document est structuré comme suit : en premier, une présentation des outils, en deuxième l’architecture Azure et la modelisation des données et en troisième la description des fonctionnalités et les mesures de sécurité.

# **Présentation des technologies et outils**

Pour mettre en place notre solution de partage de contenu, l’utilisation de certains outils et technologies nous ont facilité la tâche de la modélisation à la réalisation passant par la conception. Les lignes qui suivent porteront sur la présentation des outils utilisé.

1. Présentations des outils
   1. Environnement de développement

**Une image contenant Graphique, violet, Caractère coloré, symbole

Description générée automatiquement****Visual Studio** est IDE developpé par Microsoft, elle prends en charge plusieurs framework comme ASP.NET qui est un framework de développement web utilisé pour construire des applications web robustes et dynamiques. Ensemble, Visual Studio et ASP.NET offrent un puissant ensemble d'outils pour le développement d'applications web.

### Outils de gestion de version

**Une image contenant logo, Police, Graphique, symbole

Description générée automatiquementGitHub** est une plateforme en ligne qui permet d’héberger des dépôts Git. Git nous a permis de conserver l’historique des modifications et l’organisation effectuées sur notre projet. Ainsi nous pouvons rapidement identifier les changements effectués et de revenir à une ancienne version en cas de problème.

### Logiciel de modélisation

**Draw.io** est une plateforme en ligne ou desktop qui permet des dessins multi-plateforme. Son interface peut être utilisée pour créer des diagrammes tels que des organigrammes, des diagrammes UML ou dessiner des Architecture , notre architecture Azure y a été fait.

# 

### Langage de programmation

**.NET** est une plateforme de développement gratuite, multiplateforme et open source permettant de créer de nombreux types d’applications. Elle peut exécuter des programmes écrits dans plusieurs langages, **C#** étant le plus populaire. Elle repose sur un runtime hautes performances utilisé en production par de nombreuses applications à grande échelle.

### Fournisseur Cloud

**Azure** est la plateforme cloud computing développée par Microsoft. Elle offre une variété d'applications et de services aux particuliers, aux entreprises et aux gouvernements grâce à son infrastructure mondiale. Son nom évoque le cloud computing, ou nuage en informatique (l'externalisation des ressources informatiques d'une entreprise vers des centres de données distants).

# **Présentation de l’architecture Azure**

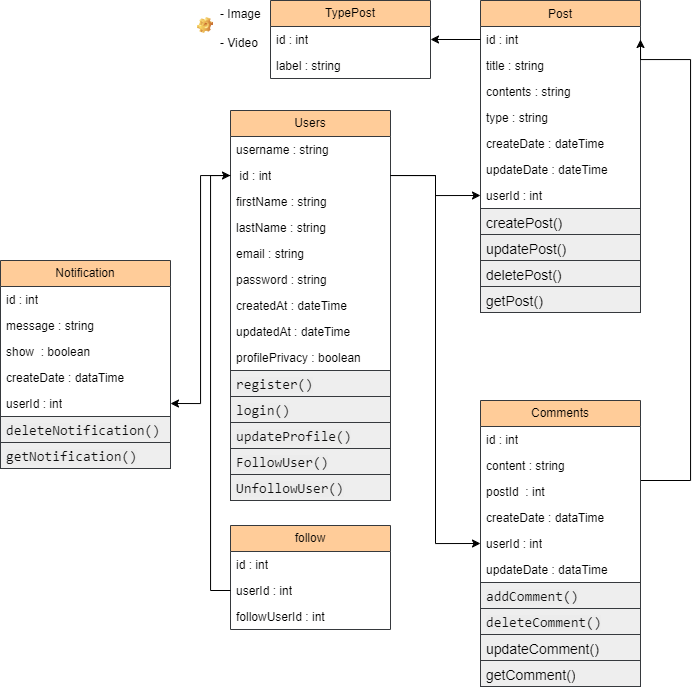
Microsoft Azure est fournisseur de services cloud qui permet de développer des services informatiques selon ses besoins allant du IAAS au SAAS. Elle offre une multitude de service et surtout permet de minimiser certains coûts en ne payant que pour ce qu’on consomme. Pour notre projet nous avons choisi le modèle PAAS car il nous permet de disposer d’un environnement de développement complet sans avoir à vous préoccuper de toute l’infrastructure de développement.

Une image contenant capture d’écran, diagramme, texte, ligne

Description générée automatiquementLa figure 1 ci-dessous représente l’architecture choisit pour la mise en place de notre solution

Figure 1:Architecture Azure SociApp.

# **Modélisation du système**



**Users** :

Représente les utilisateurs avec leurs informations personnelles (nom, email, etc.) et des actions comme s'inscrire, se connecter, ou suivre d'autres utilisateurs.

**Post** :

Contient les publications créées par les utilisateurs, avec des actions pour créer, mettre à jour ou supprimer des posts.

**Comments** :

Permet aux utilisateurs de commenter les posts, avec des fonctionnalités pour ajouter, supprimer ou modifier un commentaire.

**TypePost** :

Définit le type de publication (image, vidéo, etc.), associé à chaque post.

**Notification** :

Gère les notifications des utilisateurs, comme les messages ou alertes.

**Follow** :

Gère les relations entre les utilisateurs (qui suit qui).

# **Conception et Description des fonctionnalités**

**Api :**

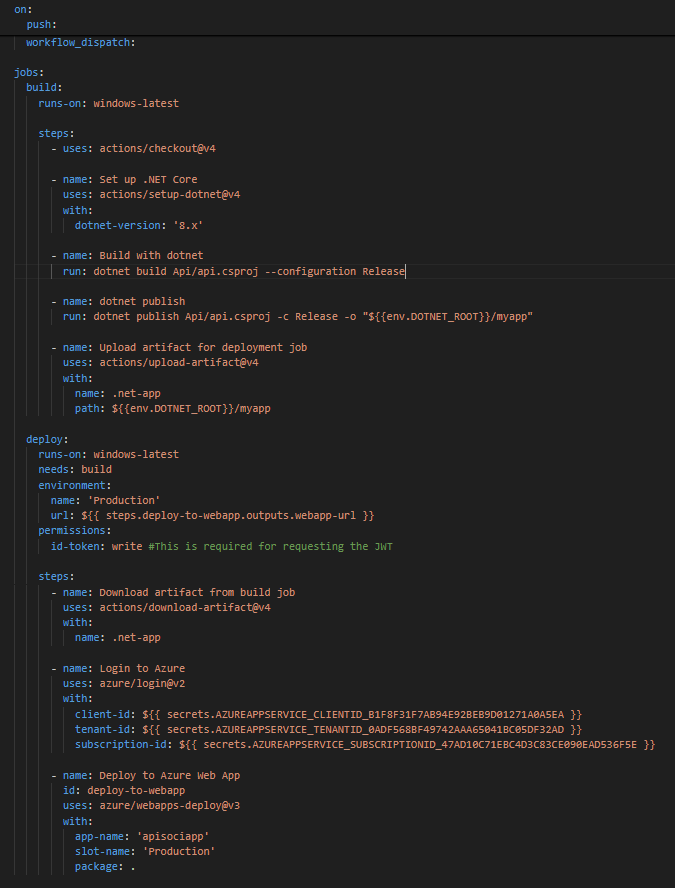
**Création de mon App Service**

**Sur le portail Azure on a créé Un App service**

**Configuration des paramètres nécessaires**

Une fois l'App Service créé, Nous configurons la connexion avec le dépôt GitHub. Cela permet de générer automatiquement un pipeline dans mon workflow. Ce pipeline sera déclenché chaque fois qu’on réalisera un commit sur la branche main.

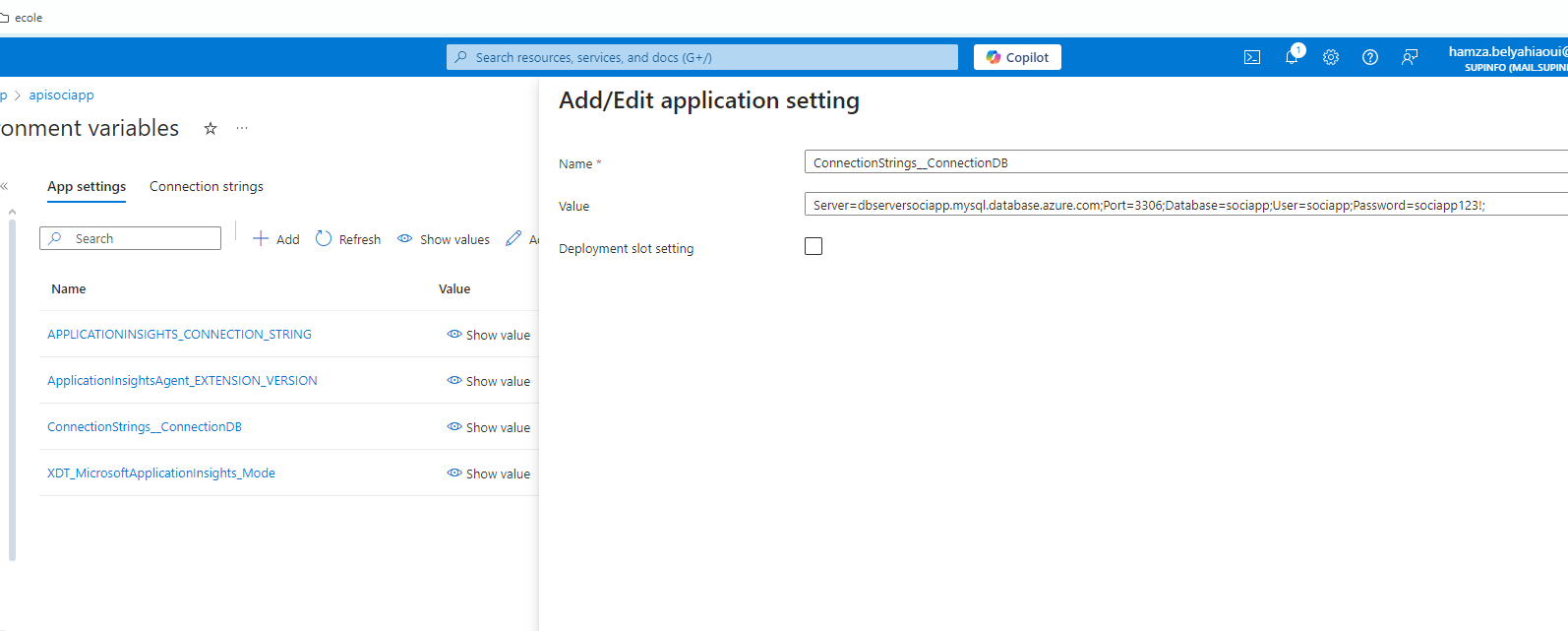
Ainsi, après chaque commit, le pipeline s'occupera de la construction (build) de l’API, puis procédera au déploiement sur l'App Service. Une fois le déploiement

terminé, nous tester mon API en utilisant l'URL fournie par l'App Service.

**DB :**

Nous avons créé une base de données dans Azure en utilisant **Azure Database for MySQL**. Une fois la base de données mise en place, nous la lions à l’application web hébergée dans un App Service sur Azure. Pour ce faire, nous récupérons la chaîne de connexion de la base de données MySQL dans le portail Azure. Cette chaîne contient toutes les informations nécessaires, telles que l'hôte, le nom d'utilisateur, le mot de passe et le nom de la base de données.

Ensuite ajoutons cette chaîne de connexion dans les paramètres de l'App Service de l’application. Nous utilisons, les variables d'environnement pour stocker cette chaîne de connexion afin qu'elle soit disponible au moment de l'exécution de mon API .NET. Cela permet à l'API d'interagir avec la base de données de manière sécurisée et flexible, sans exposer directement les informations sensibles dans le code source.

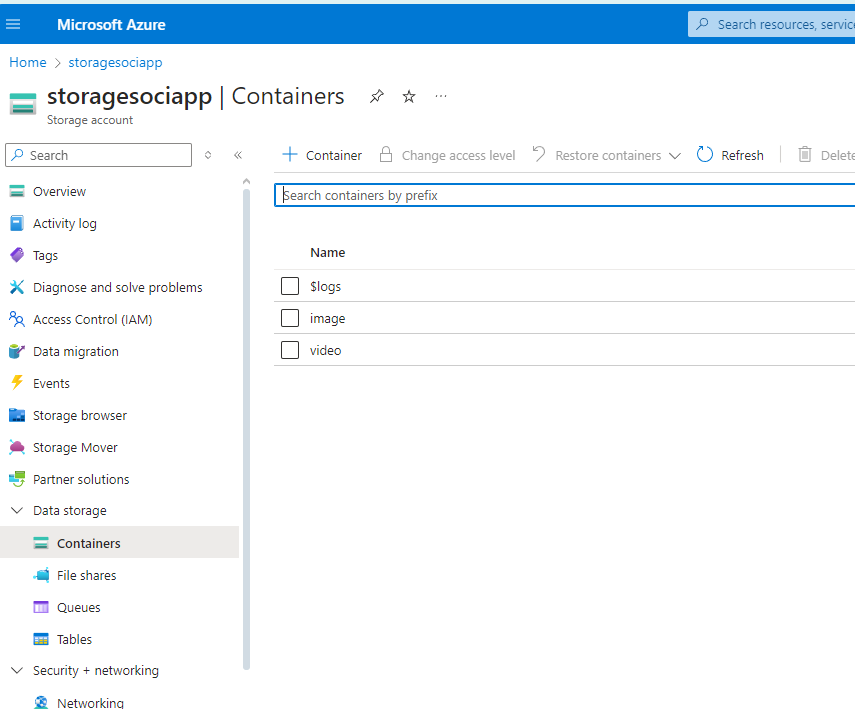


Dans Azure App Service, dans **Setting > Environment variables**.

Nous ajoutons une variable d'environnement **ConnectionString\_\_ConnectionDB** avec la chaîne de connexion.

Storage :

Pour stocker les images et vidéos du projet, nous avons choisi **Azure Blob Storage**. Un **compte de stockage** a d’abord été créé et un **conteneur** sur Azure, puis ajouté la **chaîne de connexion** dans **appsettings.json** du projet. En utilisant le package **NuGet**  **.Storage.Blobs**, un service pour gérer l'upload des fichiers vers Azure. Ce service stocke les fichiers dans le conteneur et récupère l'URL du fichier, qu’ on enregistre ensuite dans la base de données.

Dans l'API, nous ajoutons une méthode pour recevoir les fichiers via **POST** et les stocker dans Azure. Nous avons aussi configuré des **permissions de sécurité** pour contrôler l'accès aux fichiers. Après avoir testé l'upload, l'application a été deployé sur **Azure App Service**, offrant ainsi une solution fiable et sécurisée pour le stockage des fichiers multimédia dans l'application.

* Points améliorations

Le projet pourrait se voir amélioré en 3 grande phase :

* **Plan Fonctionnel**

Ajouter une map pour connaitre la position, Intégrer un chat ce qui facilitera les échanges entre les créateurs et ceux qui les suivent, mettre à jour les **feeds**, Permettre les likes

* **Plan architectural**

Ajouter **Azure Key Vault** pour mieux sécuriser les informations sensibles, en optimisant la gestion des logs avec **Application Insights**, équilibrer les charges à l’aide de **loadBalancer** ou **Api Gateway.** Aussi, on peut conteneuriser les différentes parties de la solution et les déployer à l’aide de **Azure kubernetes Services,** et **Azure Cognitive Services** pour rajouter un service Ia de détection d’images par exemple. Et pour finir, renforcer l’intégration avec **GITHUB ACTIONS** par des workflows (Tests unitaires et d'intégration, Analyse de sécurité avec des outils comme Dependabot, Déploiements progressifs (Blue-Green ou Canary) pour éviter les interruptions de service)et utiliser **Ansible ou Terraform**

* **Plan managérial**

Pour le monitoring, Visualisation via **Power BI** peut être ajoutée, avoir une idée de l’état de santé avec **Azure Health,** Azure Blob Storage tiers de performance pour catégoriser les fichiers **(Hot, cool, cold, archive)** et **Alertes proactives**

Conclusion

Ce projet social, réalisé avec une API en .NET et un front-end en ReactJs, a été déployé avec les solutions Azure, telles que **App Service** pour l'API, **Azure Storage** pour les données non structurées et **Azure Database for MySQL** pour la gestion des données. le processus de déploiement a été assuré via **GitHub Actions**.