



# Rapport Projet Cloud

## Plateforme de Gestion des Clubs Parascolaires de l'ESI

### Réalisée par :

Nada EL ANSARI N°18

Youssef BACHIRI N°7

Mohammed Amine AKHSSAS N°44

Yahya LAALEG N°56

### Encadrée par :

Mme. Imane HILAL

Année Universitaire : 2024/2025

# Table des matières

<b>Résumé</b>	<b>3</b>
<b>Abstract</b>	<b>3</b>
<b>Introduction</b>	<b>4</b>
<b>Analyse des Besoins</b>	<b>5</b>
Contexte et Enjeux . . . . .	5
Cahier des charges . . . . .	5
Besoins Fonctionnels . . . . .	5
Besoins Techniques . . . . .	5
Contraintes . . . . .	6
<b>Architecture et Conception</b>	<b>7</b>
Présentation de l'Architecture . . . . .	7
Outils et Technologies Utilisés . . . . .	8
Technologie de développement . . . . .	8
Fournisseurs Cloud . . . . .	8
Services Utilisés . . . . .	9
<b>Développement</b>	<b>10</b>
Création de l'Application avec React . . . . .	10
Intégration des Services Cloud . . . . .	11
Azure Static Web Apps . . . . .	11
Azure Cosmos DB . . . . .	14
Firebase Authentication . . . . .	17
Azure Gateway . . . . .	20
Google Cloud SQL . . . . .	21
Google Cloud Armor . . . . .	24
<b>Conclusion</b>	<b>27</b>

# Table des figures

1	Architecture Cloud Native . . . . .	7
2	React . . . . .	8
3	Microsoft Azure . . . . .	8
4	Google Cloud Platform . . . . .	8
5	Azure Static Web Apps . . . . .	9
6	Azure Cosmos DB . . . . .	9
7	Firebase Authentication . . . . .	9
8	Azure Gateway . . . . .	9
9	Google Cloud SQL . . . . .	9
10	Google Cloud Armor . . . . .	9
11	Démarrage de l'application React . . . . .	11
12	Dépôt GitHub pour le projet . . . . .	12
13	connexion avec le projet git . . . . .	12
14	Définition du chemin de construction . . . . .	13
15	Création de l'application . . . . .	13
16	Déploiement automatique du code sur Github . . . . .	14
17	FrontEnd de l'application déployée . . . . .	14
18	Formulaire de postulation . . . . .	15
19	Compte Cosmos DB . . . . .	15
20	Création d'un container . . . . .	16
21	Configuration d'une base de données . . . . .	16
22	Création d'une AP . . . . .	17
23	Connexion avec l'application Reacts . . . . .	17
24	creation d'un projet firebase . . . . .	18
25	service d'authentification . . . . .	18
26	file de Configuration . . . . .	19
27	Firebase interface . . . . .	19
28	Création d'adresse IP publique et Réseau Virtuel pour la plateforme . .	20
29	Statistiques de la passerelle d'application - Connexions et débits . . . . .	20
30	Statistiques globales des demandes - Passerelle d'application . . . . .	21
31	Création d'une instance Cloud SQL . . . . .	21
32	Creation de la base de donnée clubs . . . . .	22
33	Connexion sa l'instance Cloud SQL via React . . . . .	23
34	Affichage des clubs dans FrontEnd de l'application . . . . .	24
35	Exemple de création depolitique GCA. . . . .	25
36	Quota dépassé pour les règles de sécurité Google Cloud Armor. . . . .	26

# Résumé

Ce projet cloud vise à développer une plateforme pour gérer les clubs parascolaires de l'ESI, où les utilisateurs se connectent via Firebase Authentication. Les utilisateurs normaux accèdent à une page contenant des informations sur les clubs (stockées dans Google Cloud SQL) et un formulaire pour postuler à des postes, dont les données sont analysées par un modèle Python intégré à React pour prédire la qualification des candidats, avec les résultats stockés dans Azure CosmosDB. Les administrateurs accèdent à une page dédiée affichant les informations des étudiants sélectionnés, également récupérées depuis CosmosDB. La plateforme, déployée sur Azure Static Web Apps, est sécurisée grâce à Google Cloud Armor et Azure API Management Gateway, offrant une gestion efficace et sécurisée des clubs.

# Abstract

This cloud project aims to develop a platform for managing parascolar clubs at ESI, where users log in via Firebase Authentication. Regular users are redirected to a page displaying club information (stored in Google Cloud SQL) and a form to apply for positions, with data analyzed by a Python model integrated into React to predict candidate suitability, and results stored in Azure CosmosDB. Administrators are redirected to a dedicated page showing information about selected students, also retrieved from CosmosDB. The platform, deployed on Azure Static Web Apps, is secured using Google Cloud Armor and Azure API Management Gateway, ensuring efficient and secure club management.

# Introduction

Dans le cadre de notre projet, nous avons développé une plateforme numérique dédiée à la gestion et à la mise en valeur des clubs parascolaires de l'École des Sciences de l'Information (ESI). Cette plateforme a pour objectif de fournir une interface intuitive permettant aux utilisateurs de découvrir les différents clubs, d'explorer leurs activités et de s'inscrire en tant que membres ou responsables.

Notre démarche méthodologique a suivi un processus structuré et progressif. Nous avons débuté par le choix de React comme framework principal, offrant une base robuste pour la création d'interfaces dynamiques et réactives. Une fois l'interface établie, nous avons intégré Firebase pour gérer l'authentification des utilisateurs, permettant une gestion fiable et sécurisée des connexions.

Pour le stockage et la gestion des données, nous avons exploité deux services complémentaires : Google Cloud SQL Server pour les informations sur les clubs et Azure CosmosDB pour les données liées aux inscriptions et aux résultats des prédictions. Ces prédictions sont générées par un modèle Python, intégré directement dans le projet React, qui évalue la compatibilité des étudiants avec les postes qu'ils souhaitent occuper.

Ensuite, la plateforme a été déployée sur Azure Static Web Apps, assurant une accessibilité et une performance optimales. La sécurité de la solution a été renforcée à l'aide d'Azure Gateway, pour une gestion sécurisée des flux, et de Google Cloud Armor, pour protéger contre les menaces en ligne et assurer une utilisation sans risque.

Ce projet représente une application pratique et innovante des technologies cloud modernes dans un contexte académique. Il combine une architecture distribuée performante, des outils cloud avancés, et des approches sécuritaires de pointe, afin d'offrir une expérience fluide et fiable à la communauté étudiante tout en valorisant les clubs parascolaires de l'ESI.

# Analyse des Besoins

## Contexte et Enjeux

L'École des Sciences de l'Information (ESI) dispose d'une diversité de clubs parascolaires qui jouent un rôle essentiel dans la dynamique étudiante. Cependant, il n'existe actuellement aucune plateforme centralisée permettant de gérer ou de mettre en valeur ces clubs. Cette absence se traduit par plusieurs difficultés, notamment :

- Une visibilité limitée des clubs auprès des étudiants.
- Une gestion manuelle et peu efficace des inscriptions aux clubs.
- Une absence d'outils numériques pour centraliser les informations et les interactions.

Face à ces enjeux, la mise en place d'une plateforme numérique s'impose comme une solution moderne et adaptée pour répondre aux besoins de la communauté étudiante et des responsables de clubs.

## Cahier des Charges

Le projet de développement de cette plateforme vise à répondre aux besoins suivants :

### Besoins Fonctionnels

1. **Visualisation des clubs :**
  - Afficher les informations essentielles de chaque club (nom, logo, description, activités principales, date de création).
2. **Inscription des utilisateurs :**
  - Permettre aux étudiants de s'inscrire facilement en ligne pour devenir membres d'un ou plusieurs clubs.
3. **Gestion des données des clubs :**
  - Offrir une interface simplifiée pour gérer les données des clubs, y compris les nouvelles adhésions.

### Besoins Techniques

1. **Accessibilité :**
  - La plateforme doit être accessible à tout étudiant disposant d'un navigateur web, depuis un ordinateur ou un appareil mobile.
2. **Sécurité :**
  - Assurer une gestion sécurisée des informations personnelles des utilisateurs via des mécanismes modernes d'authentification.
3. **Stockage :**
  - Mettre en place une solution fiable pour le stockage des données des clubs et des utilisateurs, avec une gestion optimisée des fichiers (logos des clubs, CVs des candidats, etc.).

## Contraintes

### 1. Technologiques :

- Utiliser des technologies cloud-native pour garantir l'évolutivité et la robustesse de la solution.

### 2. Délais :

- Réaliser le projet dans un délai raisonnable pour une mise en service rapide.

### 3. Simplicité d'utilisation :

- Proposer une interface utilisateur intuitive et adaptée aux étudiants et aux responsables de clubs.

Cette analyse des besoins constitue le socle de la conception et du développement de la plateforme. Elle vise à répondre aux attentes des utilisateurs tout en offrant une solution moderne, sécurisée et performante.

# Architecture et Conception

## Présentation de l'Architecture

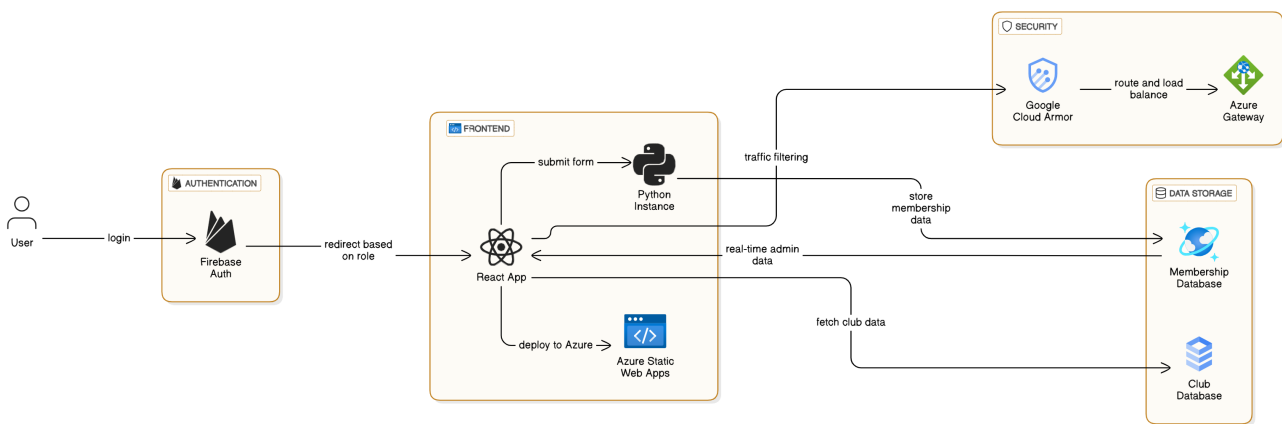


FIGURE 1 – Architecture Cloud Native

1. **Authentification** : L'authentification des utilisateurs se fait via Firebase Authentication. Lorsqu'un utilisateur se connecte, s'il s'agit d'un utilisateur normal (étudiant), il est redirigé vers une page d'accueil.
2. **Base de données des clubs** : Les informations sur les clubs de l'ESI (Nom, Description, Courriel et Nombre de Membres ) sont stockées dans une base de données SQL hébergée sur Google Cloud SQL. Sur la page d'accueil, les données des clubs sont récupérées et affichées.
3. **Formulaire d'adhésion et prédiction de rôle** : Sur la page d'accueil, un bouton permet d'accéder à un formulaire d'inscription aux clubs. Les informations saisies par l'étudiant dans ce formulaire sont envoyées à une instance Python intégrée dans le projet React pour effectuer une prédiction du rôle au sein du club. Les données saisies (informations personnelles, préférences, rôle attribué) sont ensuite stockées dans une base NoSQL hébergée sur Azure Cosmos DB.
4. **Accès administrateur** : Lorsqu'un utilisateur avec les droits administrateur se connecte, il est redirigé vers une page administrateur qui affiche, en temps réel, les données issues d'Azure Cosmos DB, présentant ainsi la liste des étudiants, leurs candidatures et les rôles qui leur ont été attribués.



## 5. Déploiement et sécurité :

- Le front-end (React + Python) est déployé sur Azure Static Web Apps.
- Pour la protection et la sécurisation du trafic, utiliser Google Cloud Armor afin de mettre en place des règles de filtrage et de prévention contre les attaques DDoS, et un Azure Gateway (Application Gateway ou API Management Gateway) pour gérer le routage, le load balancing et les politiques de sécurité supplémentaires.

## Outils et Technologies Utilisés

Pour répondre aux exigences du projet, nous avons intégré plusieurs technologies et services cloud des fournisseurs Azure et Google Cloud. Chaque service joue un rôle clé dans la mise en œuvre de la plateforme. Voici une présentation détaillée avec leurs logos :

### Technologie de développement front-end

La partie front-end a été développée avec le framework **React**, ce qui a permis de créer une interface utilisateur réactive et modulable, adaptée aux besoins de notre plateforme.

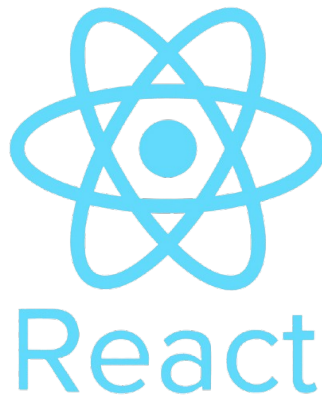


FIGURE 2 – **React**

### Fournisseurs Cloud



FIGURE 3 – **Microsoft Azure**



FIGURE 4 – **Google Cloud Platform**

## Services Utilisés



FIGURE 5 –  
**Azure Static Web Apps**



FIGURE 6 –  
**Azure Cosmos DB**



FIGURE 7 –  
**Firebase Authentication**



FIGURE 8 –  
**Azure Gateway**



FIGURE 9 –  
**Google Cloud SQL**



FIGURE 10 –  
**Google Cloud Armor**

## Description des Services

- **Azure Static Web Apps** : Hébergement de l'application React avec des performances élevées.
- **Azure Cosmos DB** : Stockage des données des clubs et des utilisateurs.
- **Firebase Authentication** : Gestion de l'authentification des utilisateurs.
- **Azure Gateway** : Sécurisation et gestion des communications entre les services.
- **Google Cloud SQL** : Gestion des bases de données relationnelles.
- **Google Cloud Armor** : Protection contre les menaces et renforcement de la résilience.

---

Ces services ont été choisis pour leur complémentarité et leur capacité à répondre aux besoins spécifiques du projet. Les logos ci-dessus illustrent les principaux outils technologiques employés.

# Développement

## Création de l'Application avec React

Le développement de la plateforme a commencé par la création d'une application React, un framework JavaScript populaire pour le développement d'applications web. Cette étape a été réalisée en suivant les commandes ci-dessous :

1. Installation de Node.js (nécessaire pour utiliser React).
2. Exécution de la commande suivante pour créer une nouvelle application React :

```
npx create-react-app my-club-platform
```

Cette commande génère une structure de projet contenant tous les fichiers nécessaires au démarrage d'une application React.

3. Accès au dossier de l'application nouvellement créée :

```
cd my-club-platform
```

4. Lancement de l'application en mode développement :

```
npm start
```

Cette commande démarre un serveur local et permet de visualiser l'application dans un navigateur web à l'adresse suivante : <http://localhost:3000>.

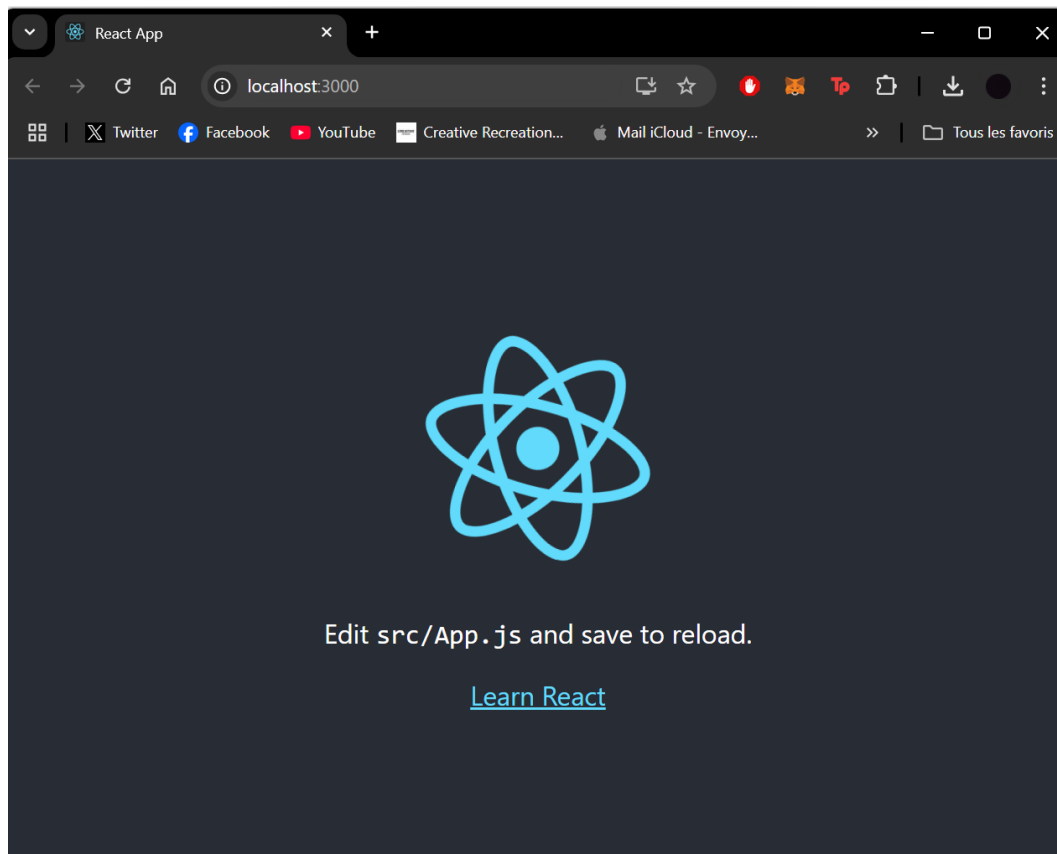


FIGURE 11 – Démarrage de l'application React

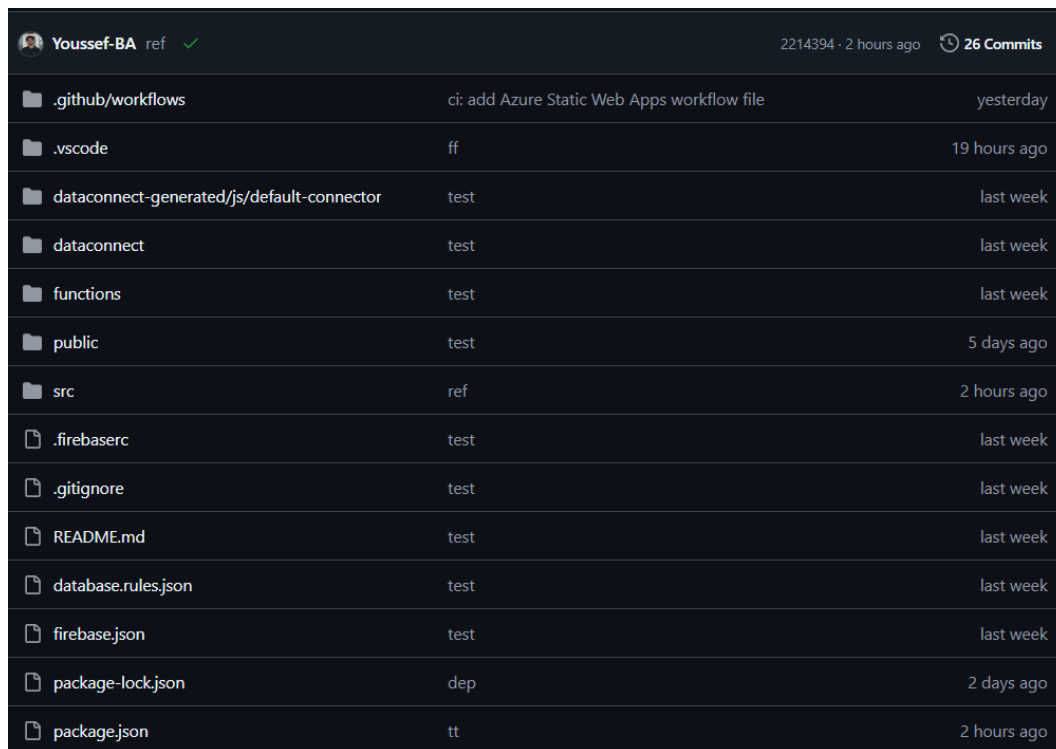
## Intégration des Services Cloud

L'intégration des différents services cloud (Azure et Google Cloud) a permis d'implémenter les fonctionnalités essentielles de la plateforme. Voici une description détaillée de chaque service utilisé et de son intégration :

### Azure Static Web Apps

Azure Static Web Apps a été utilisé pour héberger l'application React et garantir des performances élevées. Les étapes pour le déploiement sont :

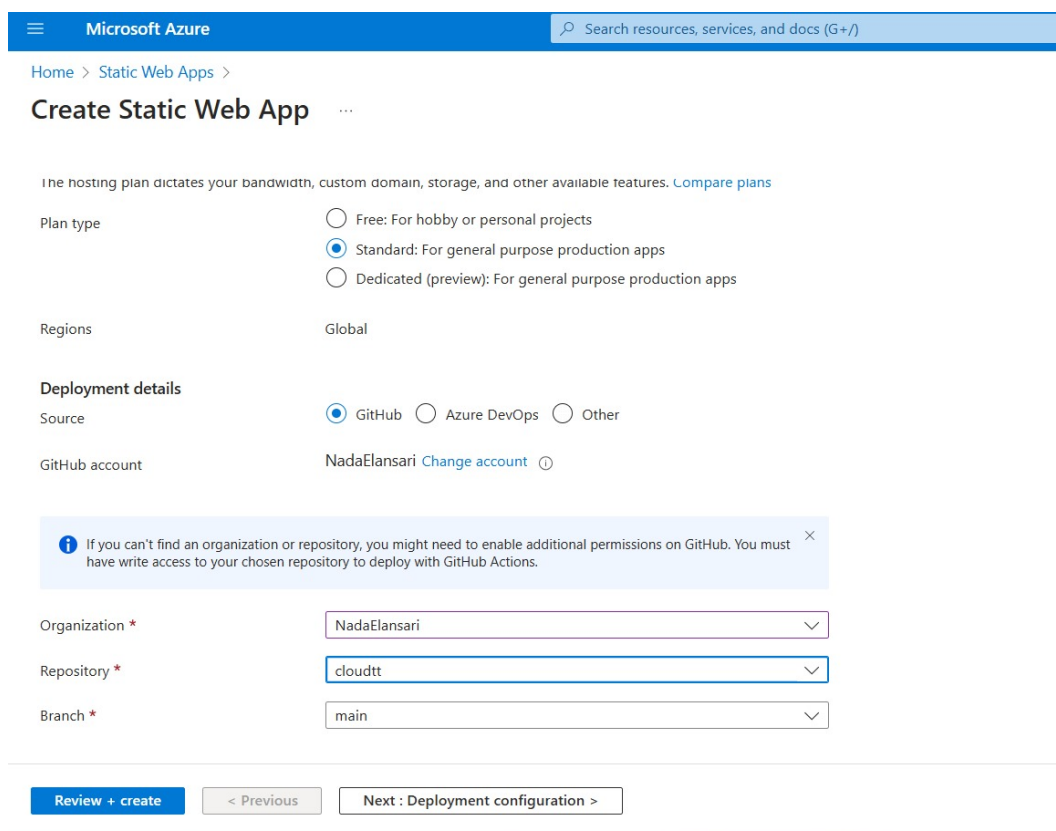
1. Création d'un dépôt GitHub pour le projet.



Youssef-BA ref ✓		2214394 · 2 hours ago	🕒 26 Commits
📁 .github/workflows	ci: add Azure Static Web Apps workflow file		yesterday
📁 .vscode	ff		19 hours ago
📁 dataconnect-generated/js/default-connector	test		last week
📁 dataconnect	test		last week
📁 functions	test		last week
📁 public	test		5 days ago
📁 src	ref		2 hours ago
📄 .firebaseerc	test		last week
📄 .gitignore	test		last week
📄 README.md	test		last week
📄 database.rules.json	test		last week
📄 firebase.json	test		last week
📄 package-lock.json	dep		2 days ago
📄 package.json	tt		2 hours ago

FIGURE 12 – Dépôt GitHub pour le projet

## 2. Connexion à Azure et sélection de l'option « Static Web Apps ».



Microsoft Azure Search resources, services, and docs (G+/I)

Home > Static Web Apps >

### Create Static Web App ...

The hosting plan dictates your bandwidth, custom domain, storage, and other available features. [Compare plans](#)

Plan type

☐ Free: For hobby or personal projects

☒ Standard: For general purpose production apps

☐ Dedicated (preview): For general purpose production apps

Regions

Global

Deployment details

Source

☒ GitHub ☐ Azure DevOps ☐ Other

GitHub account

NadaElansari [Change account](#) ⓘ

**ⓘ** If you can't find an organization or repository, you might need to enable additional permissions on GitHub. You must have write access to your chosen repository to deploy with GitHub Actions. ×

Organization \*

NadaElansari

Repository \*

cloudtt

Branch \*

main

[Review + create](#) [< Previous](#) [Next : Deployment configuration >](#)

FIGURE 13 – connexion avec le projet git

## 3. Configuration du déploiement en liant le dépôt GitHub à Azure, et définition des chemins de construction et de sortie (par défaut : /build).

Home > Static Web Apps >

Create Static Web App ...

Repository \*

cloudtt

Branch \*

main

Build Details

Enter values to create a GitHub Actions workflow file for build and release. You can modify the workflow file later in your GitHub repository.

Build Presets

React (detected)

These fields will reflect the app type's default project structure. Change the values to suit your app. [Learn more](#)

App location \*

/

Api location

e.g. "api", "functions", etc...

Output location

build

Workflow configuration

Click the button below to preview what the GitHub Actions workflow file will look like before setting up continuous deployment.

Preview workflow file

Review + create

< Previous

Next : Deployment configuration >

FIGURE 14 – Definition du chemin de construction

4. Création de l'application

Microsoft Azure

Rechercher dans les ressources, services et documents (G+/)

Copilot

youssef.bachiri@esi.ac.ma

Accueil >

StaticApp

Application web statique

Rechercher

Afficher l'application dans le navigateur

Actualiser

Supprimer

Gérer le jeton de déploiement

Envoyer vos commentaires

Vue d'ensemble

Journal d'activité

Contrôle d'accès (IAM)

Étiquettes

Diagnostiquer et résoudre les problèmes

Paramètres

Supervision

Automatisation

Aide

Bases

Groupe de ressources (...)

Abonnement (déplacer)

ID d'abonnement

Emplacement

Référence (SKU)

Étiquettes (modifier)

Démarrage

Surveillance

Historique de déploiement

URL

Source

Historique du déploiem...

Afficher le workflow

Vue JSON

FIGURE 15 – Création de l'application

## 5. Déploiement automatique à chaque modification du code sur GitHub.

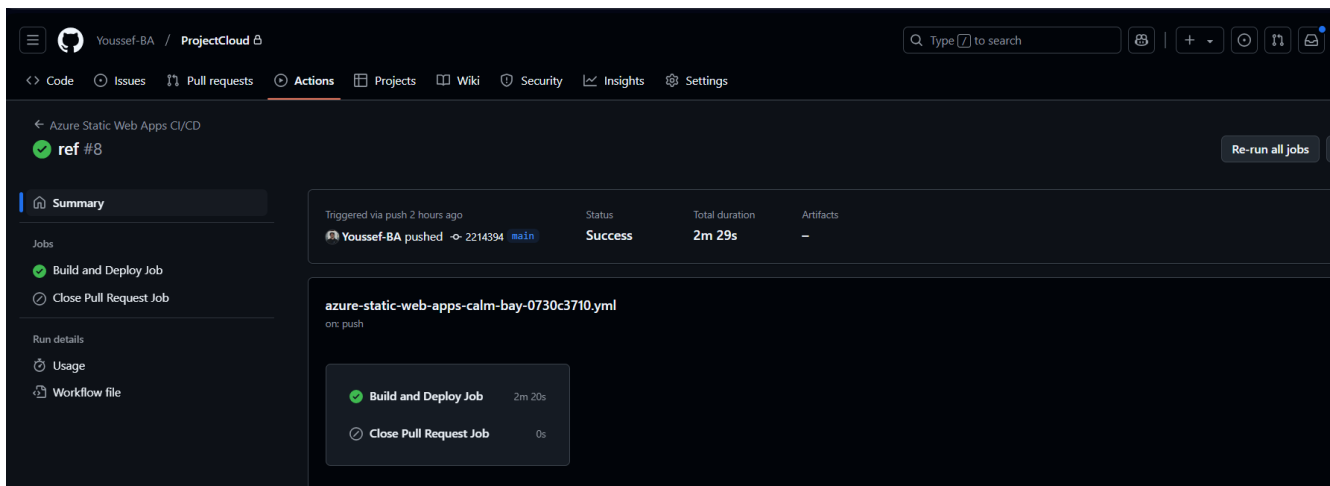


FIGURE 16 – Déploiement automatique du code sur Github

## 6. FrontEnd de l'application déployée.

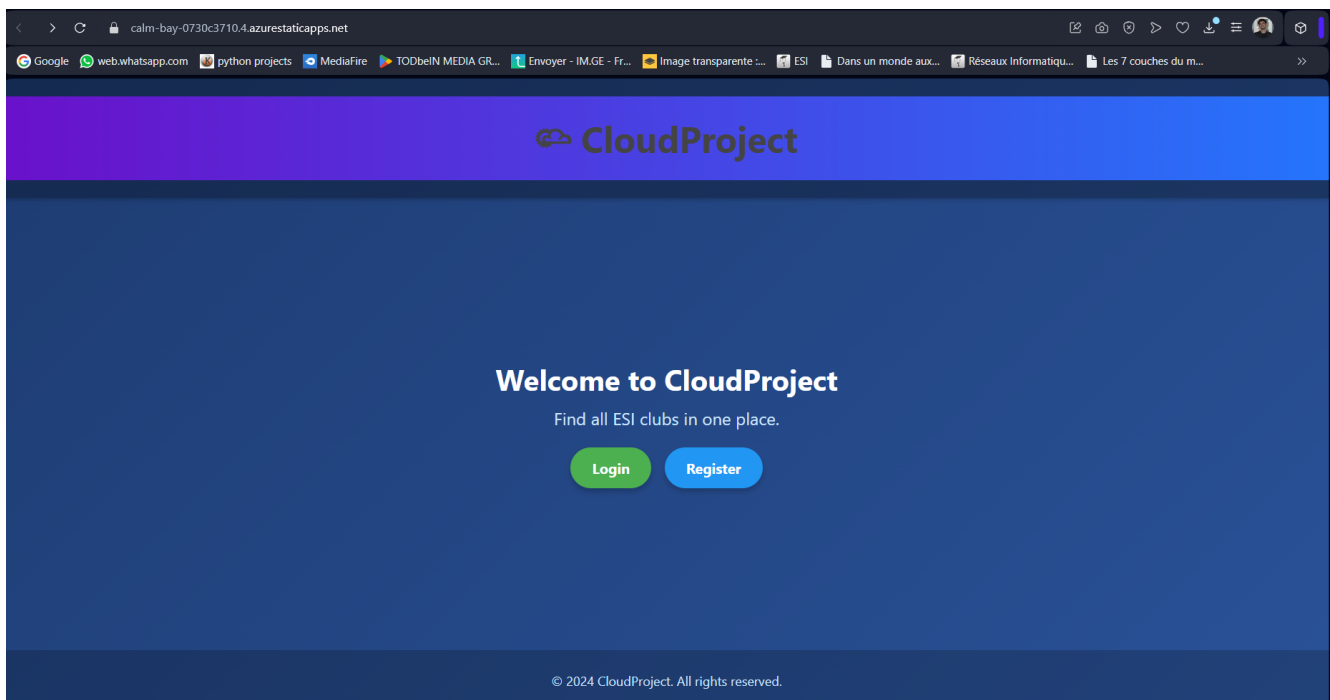


FIGURE 17 – FrontEnd de l'application déployée

## Azure Cosmos DB

Azure Cosmos DB a été utilisé pour le stockage des données des clubs et des utilisateurs. Voici les étapes principales :

1. Création du formulaire à stocker.

The screenshot shows a web browser window with the URL `calm-bay-0730c3710.4.azurestaticapps.net/apply/4`. The browser's address bar and tabs are visible at the top. The main content is a job application form titled "Formulaire d'Application". The form contains the following fields and values:

- Nom:** Bachiri
- Prénom:** Youssef
- Adresse Email:** test@gmail.com
- Numéro de Téléphone:** +21211111111
- Âge:** 28
- Sexe:** Homme

FIGURE 18 – Formulaire de postulation

## 2. Création d'un compte Cosmos DB.

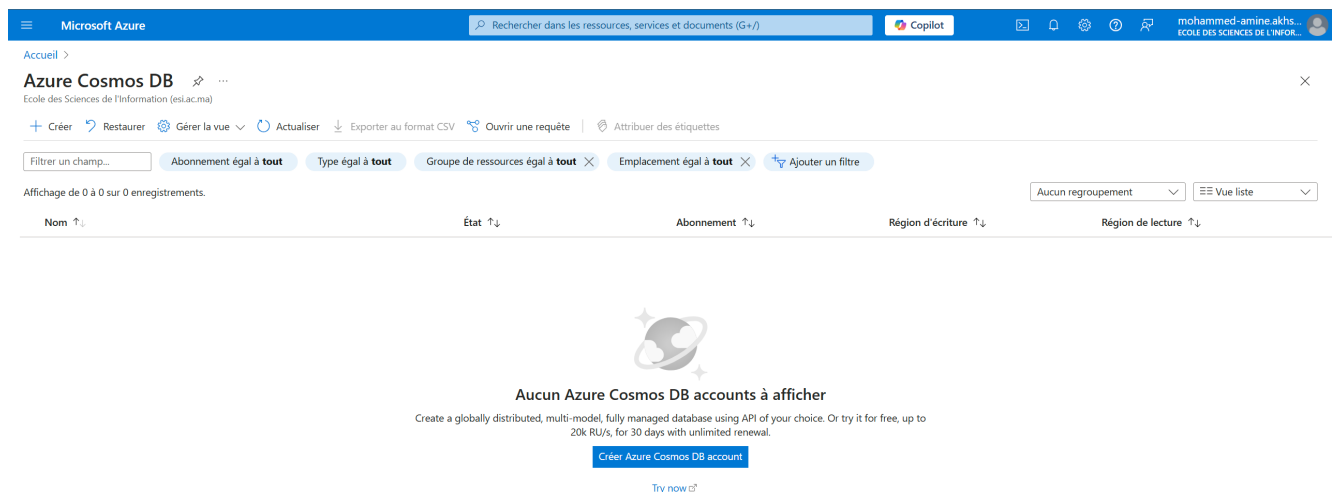


FIGURE 19 – Compte Cosmos DB

## 3. Création d'un container.



**New Container**

Your account currently has at least 1 database or container with provisioned RU/s. Billing will apply to this container if the total RU/s in your account exceeds 1000 RU/s. [Learn more](#)

**\* Database id**

☒ Create new ☐ Use existing

formu

☐ Share throughput across containers

**\* Container id**

formclub

**\* Partition key**

Required - first partition key e.g., /TenantId

Add hierarchical partition key

**\* Container throughput (autoscale)**

☒ Autoscale ☐ Manual

FIGURE 20 – Création d'un container

4. Configuration d'une base de données et d'une collection pour stocker les informations des clubs.

Home **formu...Items**

SELECT \* FROM c

id	/formu/cl...
<input checked="" type="checkbox"/> f9c82598-438e-4603-bb59-18...	
<input type="checkbox"/> ba4e69b5-7f51-4949-a712-98...	
<input type="checkbox"/> 78070b57-22c3-4805-a6c4-e5...	
<input type="checkbox"/> fd120aa6-7c1f-40a9-8a26-02...	

```

1 {
2   "nom": "nada",
3   "prenom": "Amine",
4   "email": "nadajean20@gmail.com",
5   "telephone": "0661544683",
6   "age": "20",
7   "sexe": "Homme",
8   "motivation": "SDSD",
9   "id": "f9c82598-438e-4603-bb59-18328ed24e60",
10  "_rid": "VA1SANf3aWEBAAAAAAAAAA==",
11  "_self": "dbs/VA1SAA==/colls/VA1SANf3aWE=/docs/VA1SANf3aWEBAAAAAAAAAA==/",
12  "_etag": "\"0800147a-0000-5100-0000-674e3c4d0000\"",
13  "_attachments": "attachments/",
14  "_ts": 1733180493
15 }

```

FIGURE 21 – Configuration d'une base de données

5. Création d'une API Azure Cosmos DB .

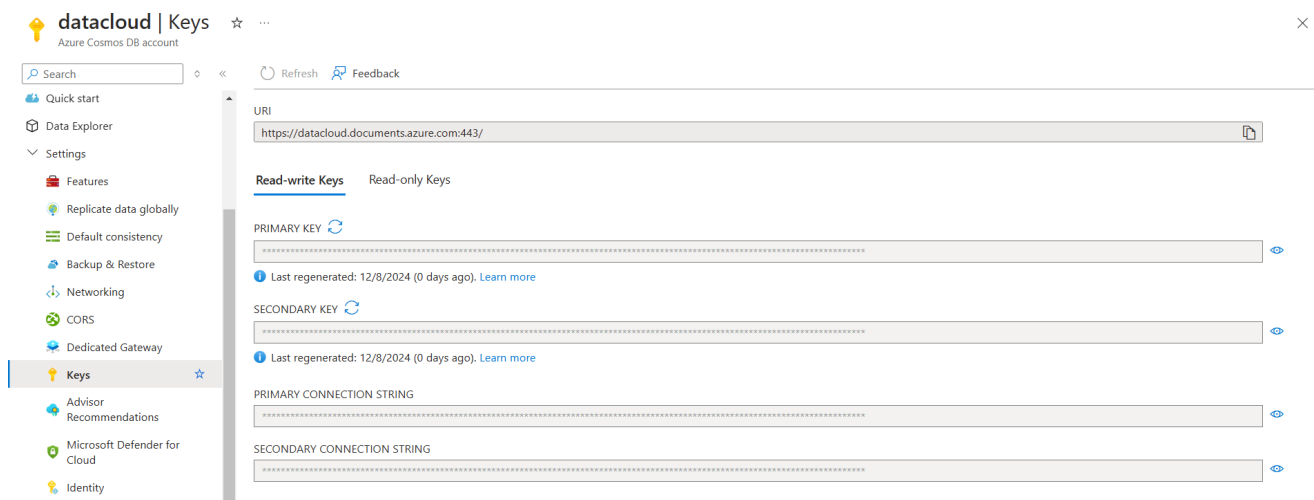


FIGURE 22 – Création d'une AP

6. Connexion de l'application React à Cosmos DB via une API REST en utilisant une clé d'accès et une URL d'extrémité.

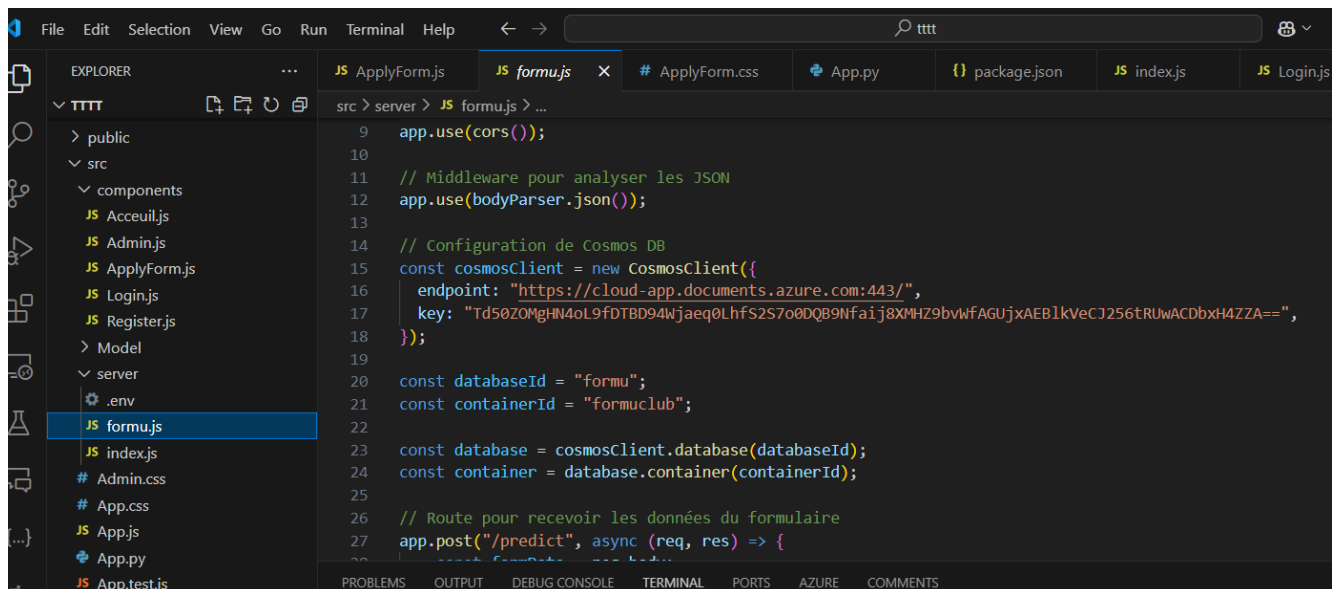


FIGURE 23 – Connexion avec l'application Reacts

## Firestore Authentication

Firestore Authentication a été utilisé pour gérer l'authentification des utilisateurs. Les étapes incluent :

1. Création d'un projet dans la console Firebase.

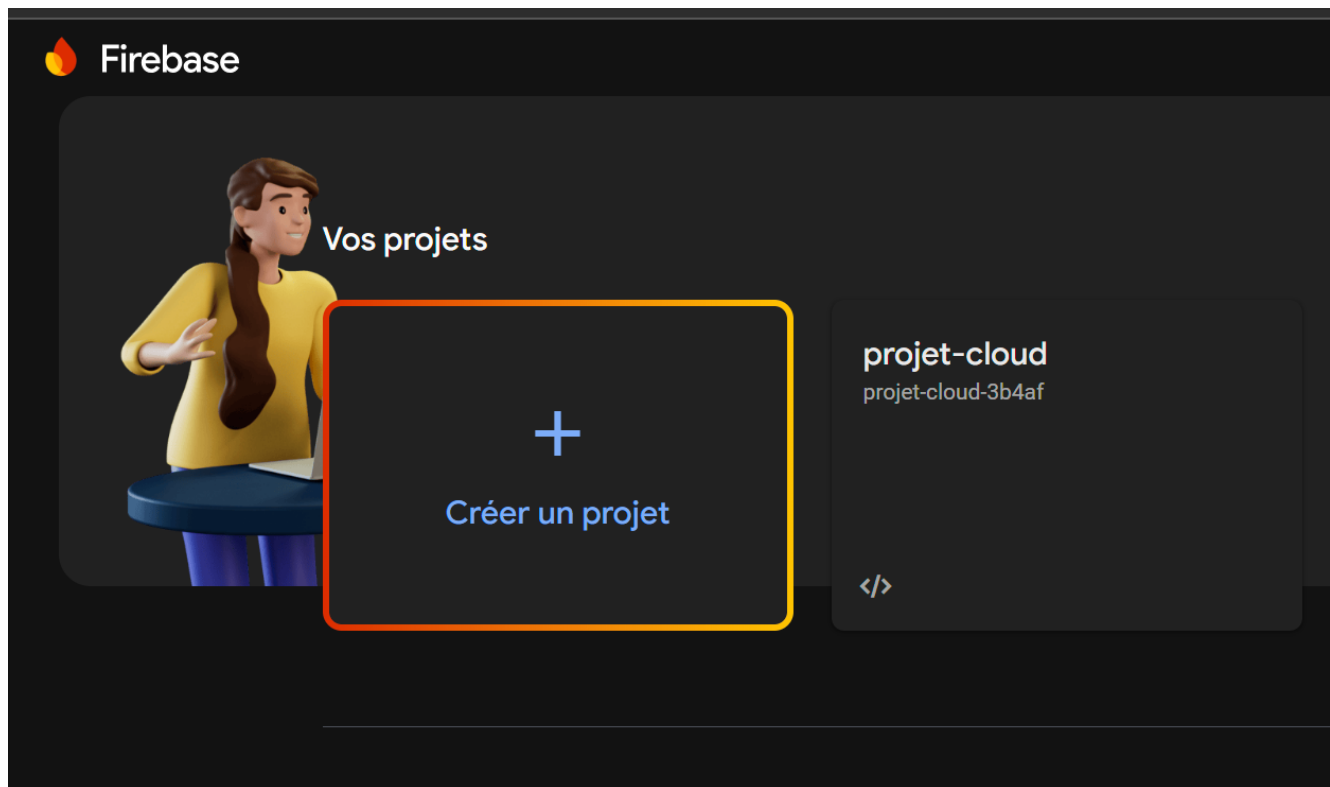


FIGURE 24 – creation d'un projet firebase

2. Activation du service d'authentification par email/mot de passe.

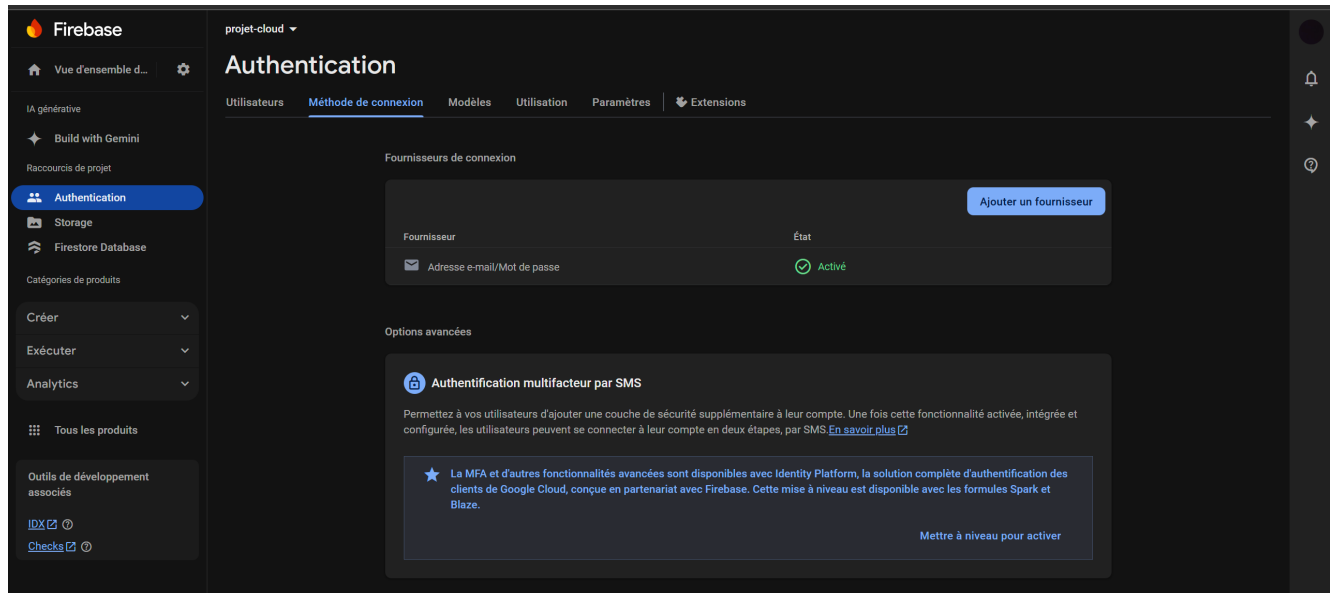


FIGURE 25 – service d'authentification

3. Ajout de la configuration Firebase dans l'application React via le fichier `firebase.js`.

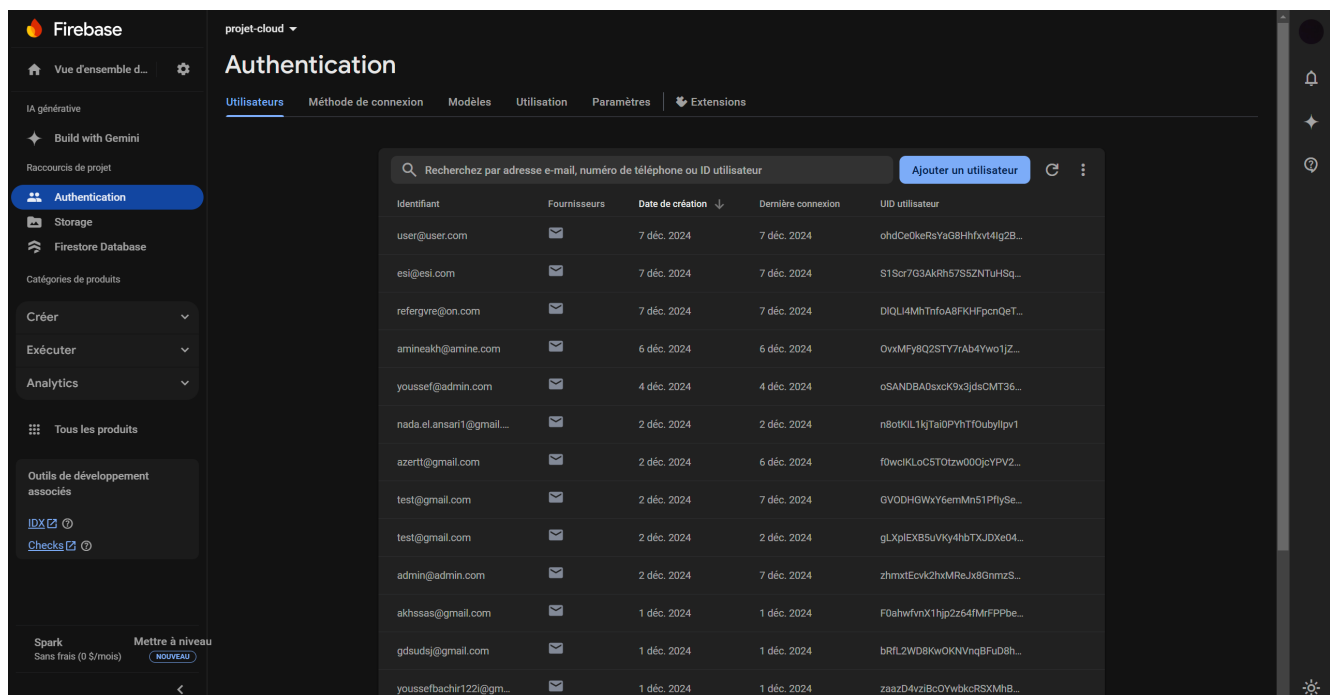
```

Cloud-app > src > JS firebase.js > ...
1  // Import Firebase modules
2  import { initializeApp } from "firebase/app";
3  import { getAuth } from "firebase/auth"; // Import getAuth from firebase/auth
4
5  // Your Firebase configuration
6  const firebaseConfig = {
7    apiKey: "AIzaSyDmGNoHCqz9_EgAjpqK_Fn0KuoaYcKxrME",
8    authDomain: "projet-cloud-3b4af.firebaseio.com",
9    projectId: "projet-cloud-3b4af",
10   storageBucket: "projet-cloud-3b4af.firebaseio.com",
11   messagingSenderId: "760982235420",
12   appId: "1:760982235420:web:c60b22f356081a4450b2d9",
13   measurementId: "G-MHGJV266TJ"
14 };
15
16 // Initialize Firebase
17 const app = initializeApp(firebaseConfig);
18
19 // Initialize Firebase Authentication
20 export const auth = getAuth(app); // Fix: Export the auth instance

```

FIGURE 26 – file de Configuration

#### 4. Interface d'authentification dans Firebase



The screenshot shows the Firebase Authentication console for a project named 'projet-cloud'. The 'Utilisateurs' (Users) tab is selected, displaying a table of users. The table has columns for 'Identifiant', 'Fournisseurs', 'Date de création', 'Dernière connexion', and 'UID utilisateur'. There are 15 users listed, each with an email address, a provider icon (email), a creation date, a last login date, and a unique UID.

Identifiant	Fournisseurs	Date de création	Dernière connexion	UID utilisateur
user@user.com	✉	7 déc. 2024	7 déc. 2024	ohdCe0keRsYaG8Hhfv4ig2B...
esi@esi.com	✉	7 déc. 2024	7 déc. 2024	S1Scr7G3AkRh57SSZNTuHSq...
refergvre@on.com	✉	7 déc. 2024	7 déc. 2024	DIQLI4MhTrfoA8FKHfpcnQeT...
amineakh@amine.com	✉	6 déc. 2024	6 déc. 2024	OvxMFy8Q2STY7tAb4Ywo1jZ...
youssef@admin.com	✉	4 déc. 2024	4 déc. 2024	oSANDBA0sxcK9x3jdsCMT36...
nada.el.ansari@gmail...	✉	2 déc. 2024	2 déc. 2024	n8otKIL1kJTai0PyhTfoublylpv1
azertt@gmail.com	✉	2 déc. 2024	6 déc. 2024	f0wciKLoC5T0tzw000jcYPV2...
test@gmail.com	✉	2 déc. 2024	7 déc. 2024	GVODHGWxY6emMn51PflySe...
test@gmail.com	✉	2 déc. 2024	2 déc. 2024	gLXpiEXBSuVky4hbTXJDxe04...
admin@admin.com	✉	2 déc. 2024	7 déc. 2024	zhmxIEcvk2htMRzJh8nmzS...
akhsas@gmail.com	✉	1 déc. 2024	1 déc. 2024	F0ahwfvnX1hjpZ264TmFPPbe...
gdsudsj@gmail.com	✉	1 déc. 2024	1 déc. 2024	bRfL2WD8KvOKNvngBFu0Sh...
youssefbachir122@gm...	✉	1 déc. 2024	1 déc. 2024	zaazD4vziBcOYwbkCRSMh8...

FIGURE 27 – Firebase interface

# Azure Gateway

Azure Gateway a été utilisé pour sécuriser les API entre les différents services.

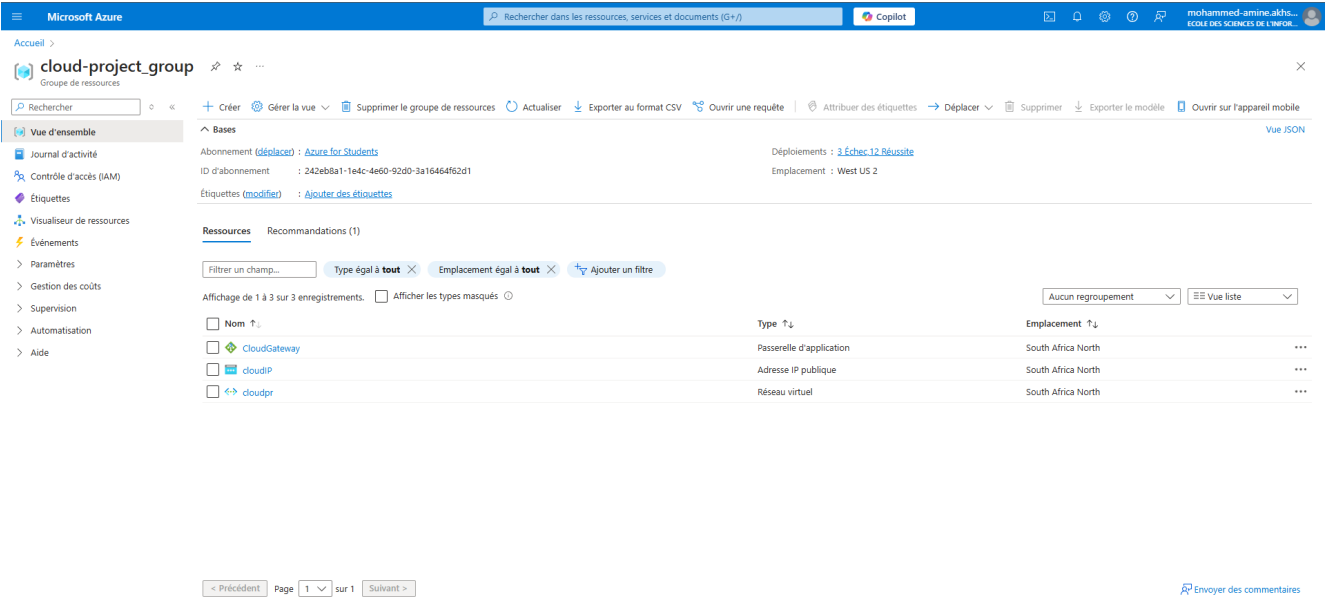


FIGURE 28 – Création d’adresse IP publique et Réseau Virtuel pour la plateforme

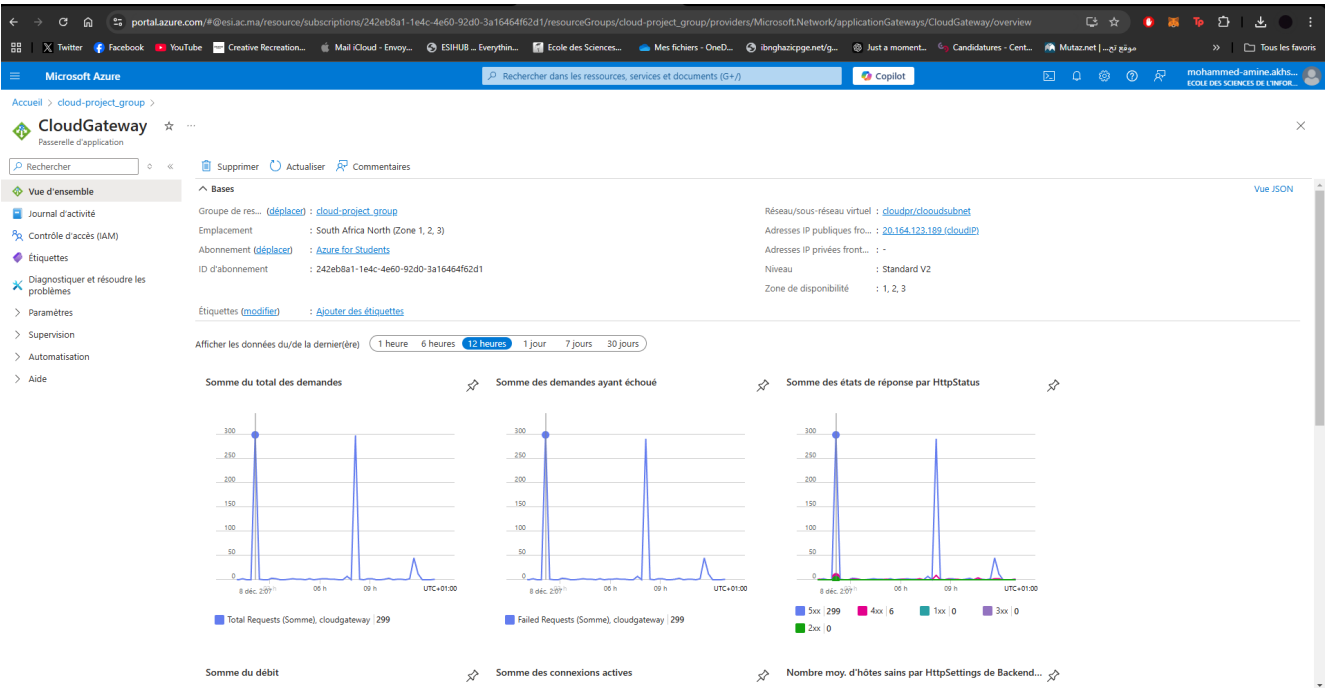


FIGURE 29 – Statistiques de la passerelle d’application - Connexions et débits

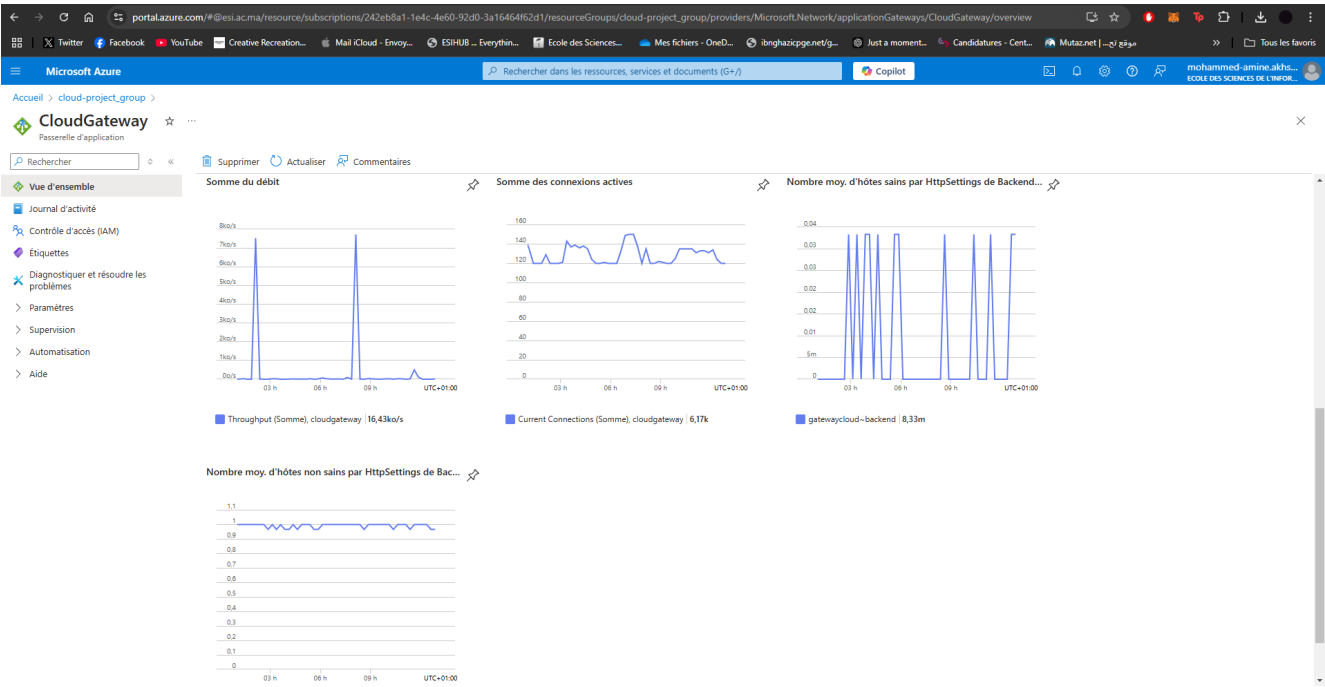


FIGURE 30 – Statistiques globales des demandes - Passerelle d'application

## Google Cloud SQL

Google Cloud SQL a été utilisé pour gérer les bases de données relationnelles. Les étapes sont :

1. Création d'une instance Cloud SQL dans la console Google Cloud.

### ← Créer une instance SQL Server

Choisissez un préréglage pour cette édition. Vous pouvez personnaliser les préréglages ultérieurement selon vos besoins.

Préréglage d'édition  
Bac à sable

[COMPARER LES PRÉRÉGLAGES DES ÉDITIONS](#)

#### Informations sur l'instance

Version de la base de données \*  
SQL Server 2019 Express

ID d'instance \*  
cloudproject

Mot de passe \*  
..... [GÉNÉRER](#)

Le nom d'utilisateur par défaut de votre compte administrateur de service est "sqlserver".  
[En savoir plus](#)

#### Choisir la disponibilité régionale et zonale

Pour améliorer les performances, vos données doivent être à proximité des services qui les utilisent. Le choix de la région est définitif, tandis que vous pouvez modifier la zone à tout moment.

Région

#### Résumé

Édition Cloud SQL	Enterprise
Région	us-central1 (Iowa)
Version de la base de données	SQL Server 2019 Express
Processeurs virtuels	1 processeurs virtuels
RAM	3,75 Go
Cache de données	Désactivé
Stockage	10 Go
Connexions	Adresse IP publique
Sauvegarde	Automatiques
Disponibilité	Zone unique
Récupération à un moment précis	Désactivée
Débit du réseau (Mo/s)	250 sur 250
IOPS	Lecture : 300 sur 12 000 Écriture : 300 sur 10 000
Débit du disque (Mo/s)	Lecture : 4,8 sur 200,0 Écriture : 4,8 sur 200,0

#### Estimation des prix (sans remises)

Ces éléments représentent uniquement les ressources de calcul, de mémoire et de stockage Cloud SQL, et reflètent la façon dont vous avez configuré votre instance jusqu'à présent. Les remises

FIGURE 31 – Création d'une instance Cloud SQL

2. Configuration de la base de données SQL Server.

En savoir plus'. At the bottom are two buttons: 'CRÉER' (blue) and 'ANNULER' (grey)." data-bbox="65 114 898 390"/>

Créer une base de données

Nom de la base de données \* clubs

Vous devez suivre les règles d'identifiant de SQL Server. [En savoir plus](#)

CRÉER ANNULER

FIGURE 32 – Creation de la base de donnée clubs

3. Connexion à l'instance Cloud SQL depuis l'application React via une API backend.

```
// server.js
const express = require("express");
const sql = require("mssql");
const cors = require("cors");

const app = express();
app.use(cors());

const dbConfig = {
  user: "sqlserver",
  password: "1234",
  server: "34.148.85.62", // Adresse IP de l'instance
  database: "clubs", // Nom de la base de données
  options: {
    encrypt: true,
    trustServerCertificate: true, // À utiliser si SSL n'est pas correctement configuré
  },
};

app.get("/clubs", async (req, res) => {
  try {
    const pool = await sql.connect(dbConfig);
    const result = await pool.request().query("SELECT * FROM clubs");
    res.json(result.recordset);
  } catch (err) {
    console.error(err);
    res.status(500).send("Erreur lors de la récupération des données.");
  }
});

const PORT = 5000;
app.listen(PORT, () => {
  console.log(`Server running on port ${PORT}`);
});
```

FIGURE 33 – Connexion sa l'instance Cloud SQL via React

#### 4. Affichage des clubs dans FrontEnd de l'application



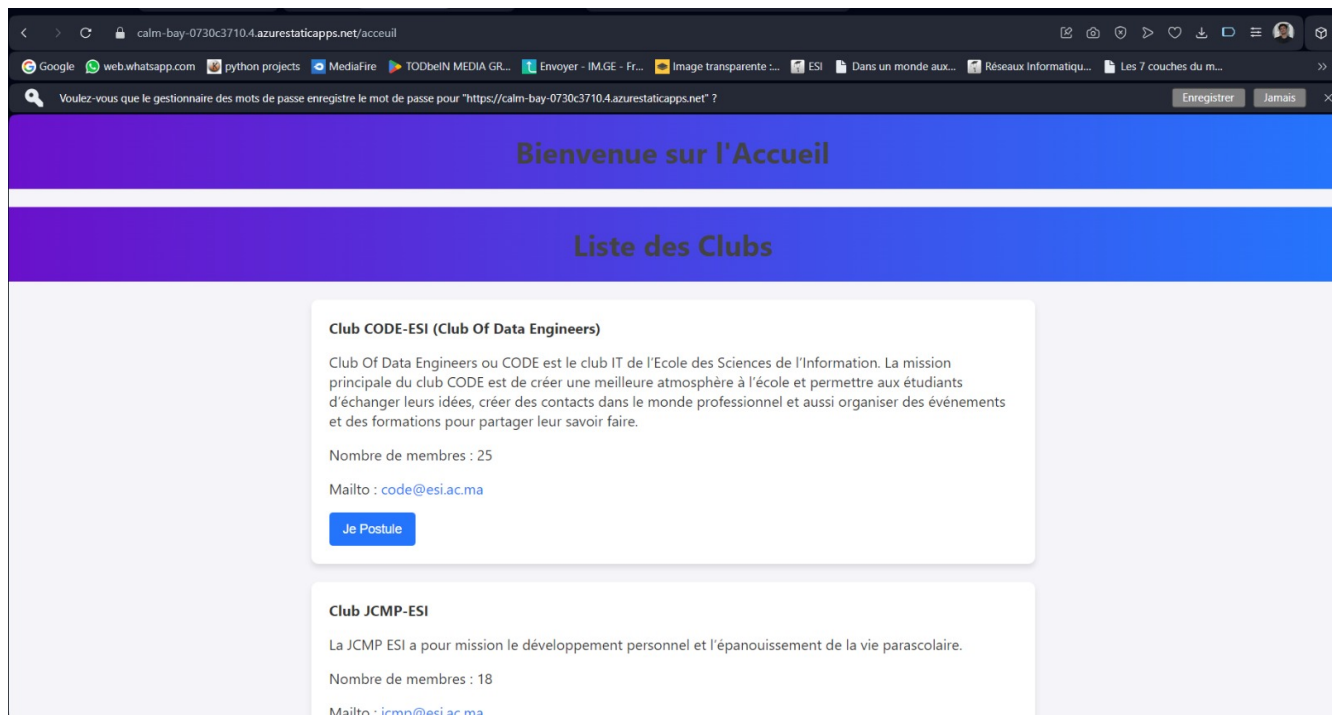


FIGURE 34 – Affichage des clubs dans FrontEnd de l'application

## Google Cloud Armor

Google Cloud Armor a été utilisé pour protéger l'application contre les menaces externes et garantir sa disponibilité. Les étapes incluent :

1. Création de politiques de sécurité dans Google Cloud Armor.
2. Définition des règles pour bloquer les requêtes malveillantes et limiter les attaques DDoS.
3. Surveillance du trafic et ajustement des règles en temps réel. Dans cette section, nous décrivons l'intégration d'un Load Balancer et de Google Cloud Armor (GCA) pour sécuriser une application déployée sur Azure Static Web Apps. Nous présentons également les échecs rencontrés lors de la configuration.

### Rôle du Load Balancer

Le Load Balancer (équilibreur de charge) joue un rôle essentiel dans la gestion du trafic réseau entrant. Ses principales fonctions incluent :

- La distribution équitable du trafic réseau entre plusieurs serveurs backend pour éviter les surcharges.
- La redondance, garantissant la disponibilité même en cas de panne d'un serveur.
- L'amélioration des performances grâce à l'optimisation de l'utilisation des ressources backend.

### Intégration de Google Cloud Armor (GCA)

Google Cloud Armor (GCA) est un pare-feu applicatif web (WAF) qui protège les applications contre des attaques comme les injections SQL, les attaques par déni de service (DDoS) ou les scripts intersites (XSS). L'intégration de GCA dans Azure Static Web Apps consiste à :

- (a) Configurer un Load Balancer pour rediriger le trafic de l’application via Google Cloud Armor.
- (b) Créer des règles de sécurité spécifiques (par exemple, blocage d’adresses IP ou géolocalisation).
- (c) Tester et valider la configuration pour garantir une protection optimale.

La figure 35 montre une tentative de création de politique de sécurité Google Cloud Armor.

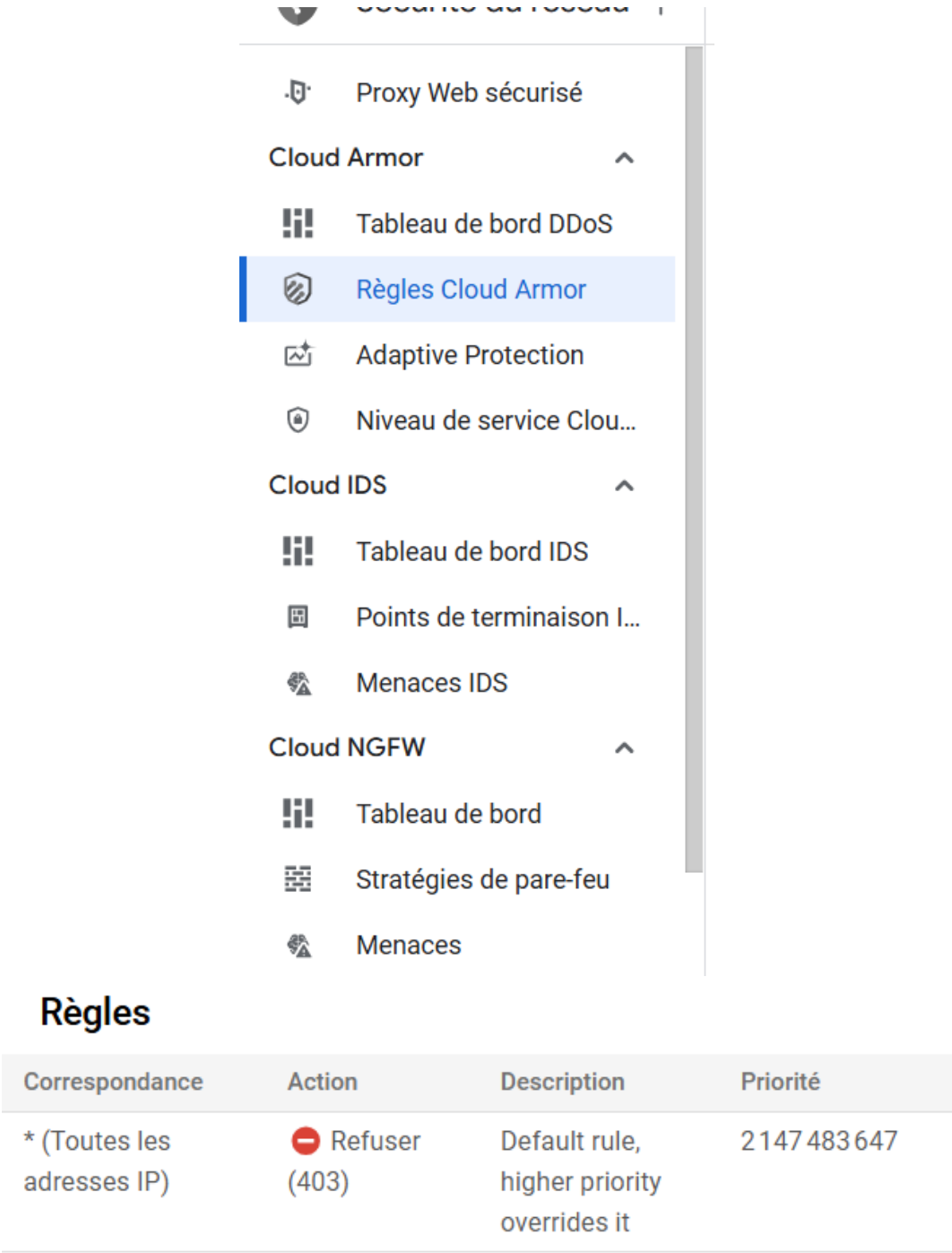
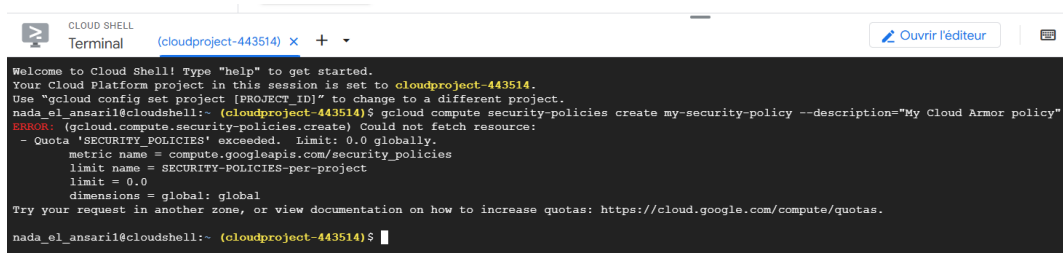


FIGURE 35 – Exemple de création de politique GCA.

Problèmes rencontrés

Lors de la configuration, nous avons rencontré une erreur liée aux quotas imposés par Google Cloud Platform (GCP). La figure 36 montre que la limite de règles de sécurité autorisées par projet (SECURITY\_POLICIES) a été atteinte.



```
Cloud Shell
Terminal (cloudproject-443514) x +
Ouvrir l'éditeur

Welcome to Cloud Shell! Type "help" to get started.
Your Cloud Platform project in this session is set to cloudproject-443514.
Use "gcloud config set project [PROJECT_ID]" to change to a different project.
nada_el_ansari@cloudshell:~ (cloudproject-443514)$ gcloud compute security-policies create my-security-policy --description="My Cloud Armor policy"
ERROR: (gcloud.compute.security-policies.create) Could not fetch resource:
- Quota 'SECURITY_POLICIES' exceeded. Limit: 0.0 globally.
  metric name = compute.googleapis.com/security_policies
  limit name = SECURITY_POLICIES-per-project
  limit = 0.0
  dimensions = global: global
Try your request in another zone, or view documentation on how to increase quotas: https://cloud.google.com/compute/quotas.
nada_el_ansari@cloudshell:~ (cloudproject-443514)$
```

FIGURE 36 – Quota dépassé pour les règles de sécurité Google Cloud Armor.

### Causes de l'échec :

- Le quota pour les politiques de sécurité (SECURITY\_POLICIES) était fixé à 0 pour ce projet.
- Aucune augmentation de quota n'a pu être demandée dans les délais impartis.
- L'application Azure Static Web Apps nécessitait une redirection via un Load Balancer configuré pour interagir avec GCA, ce qui n'a pas pu être réalisé.

# Conclusion

Dans le cadre de ce projet, nous avons développé une plateforme numérique visant à faciliter la gestion et la mise en valeur des clubs parascolaires de l'École des Sciences de l'Information (ESI). Grâce à une approche cloud-native, nous avons intégré des services modernes tels qu'Azure Static Web Apps, Firebase Authentication, et Google Cloud SQL, pour offrir une solution performante, sécurisée, et évolutive.

Ce travail nous a permis de mettre en pratique des compétences clés en développement web, en gestion de services cloud, et en intégration de technologies distribuées. Nous avons également relevé plusieurs défis techniques, notamment la configuration des services cloud et leur interconnexion, qui ont été surmontés grâce à une planification rigoureuse et à l'utilisation d'outils adaptés.

Cette plateforme représente une avancée significative pour améliorer les interactions au sein de la communauté étudiante, en rendant les clubs plus accessibles et en automatisant les processus d'inscription. Les fonctionnalités développées, telles que la visualisation des clubs et la gestion des inscriptions, répondent aux besoins identifiés dans l'analyse préliminaire.