Cloud Ready Hack

Challenge Description

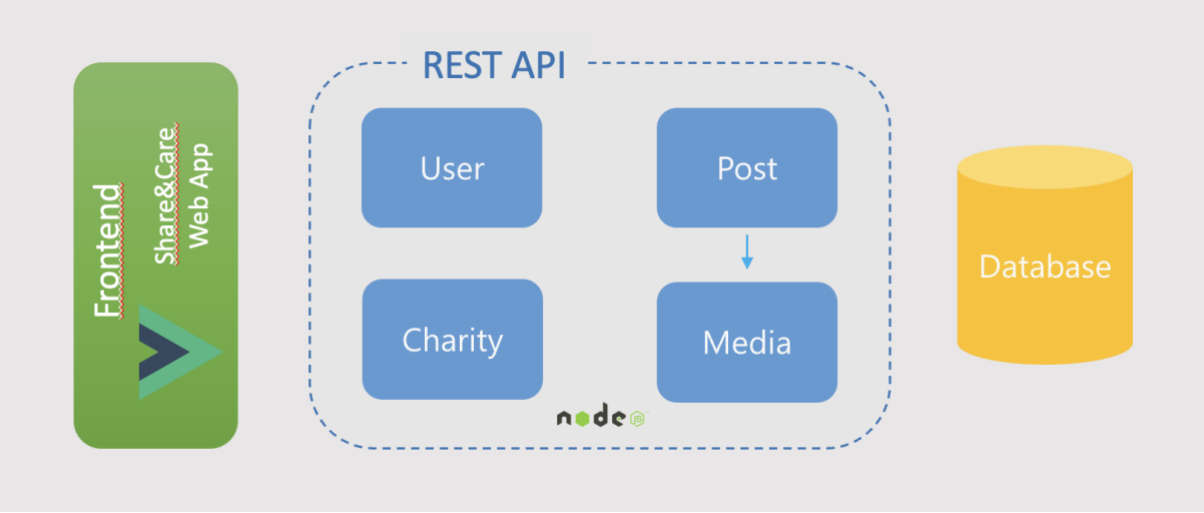
Challenge 2: Let the orchestration begin!

Now that you've worked with containers, you can tell this is a fantastic and practical technology. However, you do not want to run direct Docker commands whenever you deploy a new version of your application or service. Therefore, you need an orchestrator to help you scale on demand and better utilize your resources.

Since the whole concept of containers and container orchestrators is relatively new to your CTO, the board raises some concerns. Your CTO would like you to do a technical experiment to validate that deploying your application in a Kubernetes environment is possible.

But first things first! You previously downloaded .sql scripts for the databases. Now it's time to execute them! You need to provision a PostgreSQL server using Azure Database for PostgreSQL Flexible service and create a database for each API. You'll use .sql scripts to load the schema and the data to all the databases, so your APIs can use them.

In the previous challenge, you pushed your images to your team's Container Registry in Azure. Your assignment in this challenge is to create an Azure Kubernetes Service cluster in your team's project and deploy your application. To achieve this, leverage the documentation in the reading materials.



Once you deploy all your services, you should confirm they can talk to each other and the database. The best way to test this is to find a way to step into the API itself and execute an API call that will contact the database to pull the data. Post and Media APIs are also exchanging data. It makes sense to test this connection, too.

Now we need to put it all together and connect the APIs with the web application. Currently, your APIs are not exposed to the outside world and are reachable only from

inside the cluster. The web application uses a single URL to access all the APIs, which means all the APIs need to be accessible via a single hostname, but on different paths. You need to create a resource that will act as a gateway and enable your website (and other potential clients) to reach them on a single hostname (e.g., domain name or IP address) but on different paths.

Last but not least, your CTO cares about security, and storing secrets in the configuration files is not secure. You'll need to research and find a better way to keep your database's **hostname**, **username**, **password**, and **name** safe. Documentation might help.

Definition of done:

- You successfully deployed a Azure Database for PostgreSQL Flexible service, logged in, created all the databases, and executed the SQL scripts downloaded in the previous challenge
- You successfully created a Kubernetes cluster using Azure Kubernetes Service.
- You must demonstrate that at least one pod for each component of the Share&Care application is running.
- You must demonstrate that the components in your cluster can connect to other parts or resources (all APIs can communicate with their respective DBs, and Post API can communicate with Media API)
- Ingress Controller is deployed, and ingresses for all APIs are configured.
- Share&Care Web can access all the APIs, and is working correctly.
- Use secrets to store database name, user, and password

Reading materials:

- Quickstart: Create server - Azure portal - Azure Database for PostgreSQL - Flexible Server | Microsoft Learn
- PostgreSQL: Documentation: 14: 26.1. SQL Dump
- Migrate a database - Azure Database for PostgreSQL - Single Server | Microsoft Learn
- Encrypted connectivity using TLS/SSL in Azure Database for PostgreSQL - Flexible Server | Microsoft Learn
- Learn yaml in Y Minutes (learnxinyminutes.com)
- How YOU actually use Kubernetes, starring YAML files (softchris.github.io)
- Intro to YAML: Kubernetes Deployment Examples & Templates (mirantis.com)
- Quickstart: Deploy an AKS cluster by using the Azure portal - Azure Kubernetes Service
- Kubernetes on Azure tutorial - Deploy a cluster - Azure Kubernetes Service
- Kubernetes on Azure tutorial - Deploy an application - Azure Kubernetes Service
- Create an ingress controller in Azure Kubernetes Service (AKS) - Azure Kubernetes Service
- Ingress | Kubernetes
- Secrets | Kubernetes
- Distribute Credentials Securely Using Secrets | Kubernetes