**Assignment : Build Something**

**Description:** The purpose of this software is to demonstrate capability of building microservice-based architecture with a persistent DB that is deployed on Kubernetes. This software is for handing User Management for different types of Departments in an Organization.

**Architectural Design:** The application is built with below technology stack:

* DB as MySQL 🡪 This is a common DB being used by both the microservices having different tables
* Two Microservices based system consists of following modules:
  + Flaskapi to perform the User Management in terms of Search, Add, Modify and Delete
  + Flaskapi2 to perform Department Management to validate Valid Users in a Department
* Both the microservices are built using a popular framework known as **Python Flask** for building web applications and APIs in Python. It provides developers with a quick and easy way to create RESTful APIs that can be used by other software applications.
* MiniKube is used which helps us to spin up Kubernetes environment along with dependencies. All the components can be easily deployed on local Kubernetes single-node cluster using Minikube. We have installed minikube from public domain <https://minikube.sigs.k8s.io/docs/start/> including defining Env Variables.
* Based on Source to Image (S2I) model, we have kept our source code in <https://github.com/atifkalam/education.git>. Git helps us with an important feature of version controlling and manage multiple branches as required.
* The microservices built are horizontally scalable independent to each other where User can modify the configurations through respective .yml file by changing the Replicas parameter depending on how many POD a user expect to spin for each of this microservice.
* The image of DB MySQL is pulled from the dockerhub.com which is an official MySQL image and latest version.
* The persistent volume of the DB is Mounted and manged under the .yml file of the mysql.yml so the data remains contained if pods are restarting.
* From Security aspects, we have used Base64 Encoding mechanism to handle the DB password thus enhances our security.
* The DB is also running as an independent Service which is deployed on Kubernetes cluster.
* The application microservices port is exposed through minikube service for incoming requests.

The below representation of this software is of a 3-tier architecture developed covering the components mentioned above:

**A screenshot of a computer

Description automatically generated**

**Benefits:** Flask is a powerful framework for developing microservices and it has several advantages as listed below:

* It is Lightweight and simple and provides inbuilt features to develop microservices.
* This is very flexible to add and remove components as required by developers.
* It allows to develop microservices with RESTful design by providing tools and libraries.
* Additionally, it is very well aligned to the principles of python language thus is a good choice when developing with Python as a language.

**Challenges:**

* As stated, it’s a lightweight framework and hence it is difficult to handle complex logics in the current architecture especially applying security features like authentications.
* The API does not currently have authentications in place and hence this is an improvement area.

**Link of configuration management repository**: The repo consists of:

* Deploys a MySQL server on a Kubernetes cluster.
* Attaches a persistent volume to it, so the data remains contained if pods are restarting.
* Deploys a Flask API to add, delete and modify users in the MySQL database.
* Deploys a Flask API2 to add, delete and modify Departments linked to Users in the MySQL database.
* The entire report is available in Github at [atifkalam/education at BuildSomething (github.com)](https://github.com/atifkalam/education/tree/BuildSomething).