Backend Application Documentation

# 1. Introduction

This document provides a comprehensive overview of a backend application designed for scalable and efficient operation. The application utilizes FastAPI for its web framework, enabling asynchronous request handling and providing a robust foundation for building RESTful APIs. Additionally, the system leverages Celery for task queuing to manage background jobs, enhancing the application's performance and scalability. Containerization is achieved through Docker, ensuring consistent environments across development and production.

# 2. Installation and Setup

To set up the development environment, ensure that Docker and Python are installed. Dependencies are managed using a `requirements.txt` file. Run `pip install -r requirements.txt` to install Python dependencies. For Docker-based setups, the included `Dockerfile` and Kubernetes .yaml files streamline the containerization process.

# 3. Dockerfile

The `Dockerfile` specifies the application's containerization process, defining the base image, environment setup, and entry points. This allows for the deployment of an isolated and consistent application environment.

# 4. `main.py` Walkthrough

The `main.py` file serves as the entry point to the FastAPI application, outlining route definitions and initializing the application instance. It details the application's core functionalities and endpoint implementations.

# 5. `tasks.py` Deep Dive

In `tasks.py`, asynchronous task definitions and configurations are specified. This facilitates the execution of background jobs, leveraging Celery to manage task queues efficiently.

# 6. Kubernetes Configuration for FastAPI and Redis and Fastapi-Hpa

## 6.1 FastAPI Kubernetes Configuration (`fastApi.yaml`)

The `fastApi.yaml` file defines the Kubernetes deployment and service for the FastAPI application. It outlines the deployment strategy, including replica sets for scaling, container specifications, and resource allocation. The service definition facilitates access to the FastAPI application within the Kubernetes cluster, specifying the port mapping and service type.

## 6.2 Redis Kubernetes Configuration (`redis.yaml`)

Similarly, the `redis.yaml` file describes the deployment and service configuration for Redis, acting as the application's data store. The deployment configuration ensures that Redis runs in a highly available manner, with considerations for persistence and performance. The service definition allows other components of the application to communicate with Redis through a stable interface.

## 6.3 Horizontal Pod Autoscaler Kubernetes Configuration (`fast-hpa.yaml`)

.This section should elaborate on the Horizontal Pod Autoscaler (HPA) configuration for the FastAPI application, using the **fast-hpa.yaml** file. The HPA automatically adjusts the number of pods in a deployment based on observed CPU utilization or other selected metrics. This ensures that the application can handle varying loads efficiently without manual intervention.

# 7. Run Command to start the Application k8s

## 7.1 kubectl apply -f .\redis.yaml

The created deployment and service redis ,please check **kubectl get pods** command is redis is run fine or not.

## 7.2 kubectl apply -f .\fastApi.yaml

The created deployment and service of fastapi ,please check **kubectl get pods** command is fastapi running fine or not.

## 7.3 kubectl apply -f ./fastapi-hpa.yaml

The HPA automatically adjusts the number of pods in a deployment based on observed CPU utilization or other selected metrics

**Note**

In case you do not have metrics server then this set of command.

1. **kubectl apply -f** [**https://github.com/kubernetes-sigs/metrics-server/releases/latest/download/components.yaml**](https://github.com/kubernetes-sigs/metrics-server/releases/latest/download/components.yaml)
2. **kubectl get deployment metrics-server -n kube-system**
3. **kubectl get hpa**

**Additional k8s command**

1. kubectl logs
2. kubectl get deployment
3. kubectl get service
4. kubectl get pods
5. kubectl delete deployment <name >
6. kubectl service deployment <name >