**Bookstore Application Documentation**

**Table of Contents**

[1. Prerequisites](https://chat.openai.com/c/66a686f7-856a-431e-8023-4ff29869b7cc" \l "1-prerequisites" \t "https://chat.openai.com/c/_new)

[1.1 Docker](https://chat.openai.com/c/66a686f7-856a-431e-8023-4ff29869b7cc" \l "11-docker" \t "https://chat.openai.com/c/_new)

[1.2 Kubernetes (Minikube)](https://chat.openai.com/c/66a686f7-856a-431e-8023-4ff29869b7cc" \l "12-kubernetes-minikube" \t "https://chat.openai.com/c/_new)

[1.3 kubectl](https://chat.openai.com/c/66a686f7-856a-431e-8023-4ff29869b7cc" \l "13-kubectl" \t "https://chat.openai.com/c/_new)

[1.4 MongoDB](https://chat.openai.com/c/66a686f7-856a-431e-8023-4ff29869b7cc" \l "14-mongodb" \t "https://chat.openai.com/c/_new)

[2. Installation](https://chat.openai.com/c/66a686f7-856a-431e-8023-4ff29869b7cc" \l "2-installation" \t "https://chat.openai.com/c/_new)

[2.1 Build and Push Docker Image](https://chat.openai.com/c/66a686f7-856a-431e-8023-4ff29869b7cc" \l "21-build-and-push-docker-image" \t "https://chat.openai.com/c/_new)

[2.2 Launch MongoDB Instance](https://chat.openai.com/c/66a686f7-856a-431e-8023-4ff29869b7cc" \l "22-launch-mongodb-instance" \t "https://chat.openai.com/c/_new)

[2.3 Deploy Python Application on Kubernetes](https://chat.openai.com/c/66a686f7-856a-431e-8023-4ff29869b7cc" \l "23-deploy-python-application-on-kubernetes" \t "https://chat.openai.com/c/_new)

[3. Usage](https://chat.openai.com/c/66a686f7-856a-431e-8023-4ff29869b7cc" \l "3-usage" \t "https://chat.openai.com/c/_new)

[4. Configuration](https://chat.openai.com/c/66a686f7-856a-431e-8023-4ff29869b7cc" \l "4-configuration" \t "https://chat.openai.com/c/_new)

[4.1 MongoDB Configuration](https://chat.openai.com/c/66a686f7-856a-431e-8023-4ff29869b7cc" \l "41-mongodb-configuration" \t "https://chat.openai.com/c/_new)

[4.2 Docker Configuration](https://chat.openai.com/c/66a686f7-856a-431e-8023-4ff29869b7cc" \l "42-docker-configuration" \t "https://chat.openai.com/c/_new)

**1. Prerequisites**

Let's start by ensuring that you have all the tools needed to set up the Bookstore application:

**1.1 Docker**

Docker is a powerful tool for containerization, making it easy to package and distribute applications. To install Docker, visit the [official Docker website](https://docs.docker.com/get-docker/) and follow the installation instructions for your operating system.

**1.2 Kubernetes (Minikube)**

Kubernetes is an open-source container orchestration platform. Minikube allows you to run a Kubernetes cluster locally for development and testing purposes. Install Minikube by following the instructions provided [here](https://minikube.sigs.k8s.io/docs/start/).

**1.3 kubectl**

`kubectl` is the command-line tool used to interact with Kubernetes clusters. Install it by visiting the official Kubernetes documentation [here](https://kubernetes.io/docs/tasks/tools/install-kubectl/).

**1.4 MongoDB**

MongoDB is a NoSQL database, and our application uses it to store book-related data. Install MongoDB by following the instructions available [here](https://docs.mongodb.com/manual/installation/).

**2. Installation**

Now that we have the prerequisites sorted, let's dive into the installation process.

**2.1 Build and Push Docker Image**

The Bookstore application is packaged as a Docker container, and the first step is to build and push the Docker image.

```bash

# Build the Docker image

docker build -t yourusername/bookstore-app:latest .

# Push the Docker image to Docker Hub

docker push yourusername/bookstore-app:latest

```

Replace `yourusername` with your Docker Hub username. If you don't have a Docker Hub account, it's straightforward to create one [here](https://hub.docker.com/).

**2.2 Launch MongoDB Instance**

Now, let's set up the database. You have two options:

**2.2.1 Local MongoDB**

If you have MongoDB installed locally, start it with:

```bash

mongod

```

**2.2.2 MongoDB with Docker**

If you prefer using Docker for MongoDB, run the following command:

```bash

docker run -d -p 27017:27017 --name mongo mongo:latest

```

This command pulls the latest MongoDB Docker image and starts a MongoDB container. The `-p` flag maps the container's port 27017 to the host's port 27017.

**2.3 Deploy Python Application on Kubernetes**

Now, it's time to deploy the Python Flask application on Kubernetes. Apply the provided Kubernetes configuration files:

```bash

# Apply Kubernetes Deployment and Service files

kubectl apply -f deployment.yaml

kubectl apply -f mongo-service.yaml

kubectl apply -f app-service.yaml

```

These commands use `kubectl` to apply the Kubernetes configuration files. The `deployment.yaml` file deploys the Python application, and the `mongo-service.yaml` and `app-service.yaml` files define services for MongoDB and the Python application, respectively.

**3. Usage**

With the installation completed, let's explore how to use the Bookstore application.

- Open your web browser and go to [http://localhost](http://localhost).

- If you're using a remote cluster, use the cluster's IP address instead.

The application provides various endpoints for managing books, genres, and more. Explore the app, add, update, and delete books to get hands-on experience.

**4. Configuration**

Now, let's discuss how you can tailor the Bookstore application to fit your environment.

**4.1 MongoDB Configuration**

The MongoDB URI is set in the `app.py` file. Open the file and find the following line:

```python

app.config["MONGO\_URI"] = "mongodb://mongo:27017/myDatabase"

```

Change the URI according to your MongoDB instance. The format is `mongodb://<hostname>:<port>/<database>`.

**4.2 Docker Configuration**

In `deployment.yaml`, find:

```yaml

image: yourusername/bookstore-app:latest

```

Update `yourusername` and adjust the tag if you're using versioning.