**Documentation: Containerization of Full-Stack Application**

**Project Overview:**

The project to containerize the full-stack application and management. Docker is used to containerize the frontend, backend, and database,by Docker Compose. A shell script automates the build, tag, push, and deployment process.

**Prerequisites:**

1.Install Docker and Docker Compose.

2.Create a Docker Hub account.

3.Install Git for cloning the project repository.

**Project Components:**

1.React Frontend: Serves the user interface.

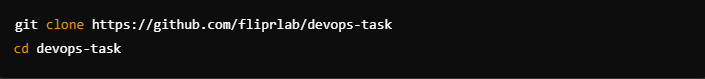
2.Express Backend: Handles API requests and business logic.

3.MongoDB Database: Stores application data.

**Steps to Containerize the Application:**

1. **Clone the Repository**

Clone the GitHub repository containing the project:



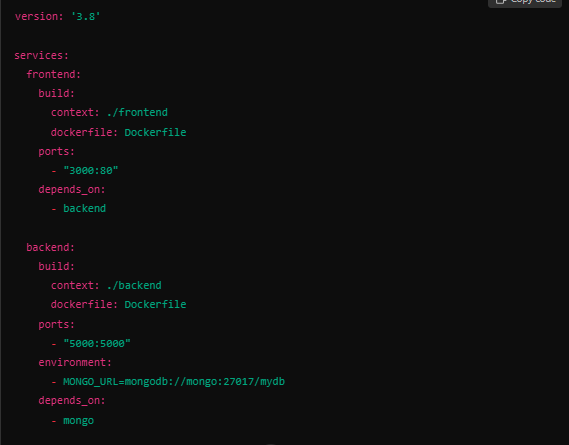
### **2. Create Dockerfiles**

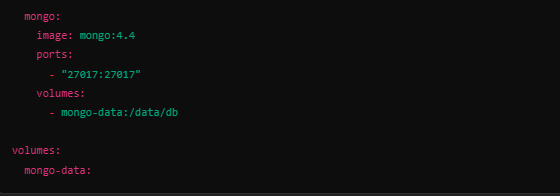
#### Frontend Dockerfile

Backend Dockerfile



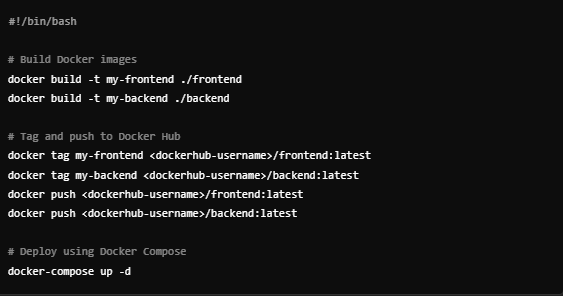
### **3. Create Docker Compose File**





**4**. **Write Automation Shell Script**

Create a script deploy.sh for automating the deployment process:



**Docker Commands:**

**For Image and Container Management**

**1.Build an image:**

****

**2.show Docker images:**

****

**3.Run a container:**

****

**Docker Compose Commands:**

**1.Start services:**

****

**2.Stop services:**

****

### 

### **Docker Hub Commands:**

**1.Log in to Docker Hub**:



**2.Push an image**:



**3.Pull an image**:



**Testing the Application:**

**1.Run the shell script:**

****

**2.Test the application:**

* Frontend: http://localhost:3000
* Backend: http://localhost:5000
* MongoDB: Check logs or access via MongoDB client.

## **Conclusion**

By following this documentation, you can successfully containerize and deploy a full-stack application. Docker simplifies deployment, improves portability, and ensures consistency across environments.