Dockerfile for nodejs appn

# Use an official Node.js runtime as a parent image

FROM node:18

# Set the working directory

WORKDIR /app

# Copy package.json and package-lock.json

COPY package\*.json ./

# Install dependencies

RUN npm install

# Copy the rest of the app

COPY . .

# Expose port 3000

EXPOSE 3000

# Command to run the application

CMD ["node", "index.js"]

Docker-compose.yml

version: "3"

services:

  app:  #this is srvice name

    build: . //docker file must be in current diectory

    ports:

      - "3001:3000"

    volumes:

      - .:/app

      - /app/node\_modules

    environment:

      - PORT=3000

    command: ["node", "index.js"]

map with correct port since while creating our iamge we expose port 3000 for container this match with compose file

**What does this mean?**

* It is **equivalent to**:

yaml

CopyEdit

build:

context: .

* It tells Docker to:
  + Look in the **current directory (./)** for the Dockerfile.
  + Use the **default Dockerfile** for building the image.

### ****When to Use Which?****

✅ **Use build: .** → When your Dockerfile is in the **same directory** as docker-compose.yml.  
✅ **Use build: with context & dockerfile** → When:

* The **Dockerfile is located in another directory**.
* You need **better control** over the build process.

### ****1.**** build: ****with Context & Dockerfile (Your Code)****

build:

context: ./

dockerfile: Dockerfile

### ****🔹 What does this mean?****

* **context: ./** → The build context is set to the current directory (./).
* **dockerfile: Dockerfile** → It explicitly specifies Dockerfile as the build file.

This allows flexibility if:

* The **Dockerfile is located elsewhere** (e.g., dockerfile: ./custom.Dockerfile).
* You want to specify a **different build context**.

# Use Node.js base image

FROM node:18

# Set working directory

WORKDIR /app

# Copy package.json and install dependencies

COPY package\*.json ./

RUN npm install

# Copy the rest of the application

COPY . .

# Expose port

EXPOSE 5000

# Start the backend server

CMD ["node", "server.js"]

Dockerfile in backend dir

Dockerfile in frontend dir

# Use Node.js base image

FROM node:18

# Set working directory

WORKDIR /app

# Copy package.json and install dependencies

COPY package\*.json ./

RUN npm install

# Copy the rest of the application

COPY . .

# Expose port

EXPOSE 3000

# Start the frontend server

CMD ["npm", "start"]

version: "3"

services:

  backend:

    build:

      context: ./backend

      dockerfile: Dockerfile.backend

    ports:

      - "5000:5000"

    volumes:

      - ./backend:/app #mount ./backedn dir with app directory inside the cotnainer

    environment:

      - MONGO\_URI=mongodb://mongo:27017/mydatabase

    depends\_on:

      - mongo

  frontend:

    build:

      context: ./frontend #use frontned for building a docker image

      dockerfile: Dockerfile.frontend

    ports:

      - "3000:3000"

    volumes:

      - ./frontend:/app

      - /app/node\_modules

    depends\_on:

      - backend

  mongo:

    image: mongo

    ports:

      - "27017:27017"

    volumes:

      - mongo-data:/data/db

volumes:

  mongo-data:

as the docker file located in backend folder we need to give build ./backend