Docker compose file:

```
version: "3.9"

services:
  frontend:
    build:
      context: ./ecommerce/ecommerce-frontend
      dockerfile: Dockerfile
    ports:
      - "3000:80"
    depends_on:
      - backend
    networks:
      - webnet

  backend:
    build:
      context: ./ecommerce
      dockerfile: Dockerfile
    ports:
      - "8000:8000"
    volumes:
      - sqlite_data:/app/db
    environment:
      - DJANGO_ALLOWED_HOSTS=localhost,127.0.0.1
      - CORS_ALLOWED_ORIGINS=http://localhost:3000
    networks:
      - webnet

volumes:
  sqlite_data:

networks:
  webnet:
```

Backend dockerfile:

```
FROM python:3.11-slim

ENV PYTHONDONTWRITEBYTECODE=1
ENV PYTHONUNBUFFERED=1

WORKDIR /app

# Install build dependencies for mysqlclient on Debian Bookworm
RUN apt-get update && \
    apt-get install -y --no-install-recommends \
        gcc \
        build-essential \
        libmariadb-dev \
        libmariadb-dev-compat \
        pkg-config \
        python3-dev \
    && rm -rf /var/lib/apt/lists/*

COPY requirements.txt .
RUN pip install --upgrade pip
RUN pip install -r requirements.txt
```

```
COPY . .

COPY docker-entrypoint.sh /usr/local/bin/
RUN chmod +x /usr/local/bin/docker-entrypoint.sh

EXPOSE 8000

ENTRYPOINT ["docker-entrypoint.sh"]
CMD ["gunicorn", "ecommerce.wsgi:application", "--bind", "0.0.0.0:8000", "--workers", "3"]
```

Docker-entrypoint.sh:

```
#!/bin/bash
set -e

python manage.py migrate --noinput
python manage.py collectstatic --noinput

exec "$@"
```

Frontedn dockerfile:

```
# 1) Build Angular app
FROM node:18 AS builder
WORKDIR /app

COPY package*.json ./
RUN npm ci --legacy-peer-deps

COPY . .
ENV NODE_OPTIONS=--max-old-space-size=4096
RUN npm run build -- --configuration production

# 2) Serve using Nginx
FROM nginx:alpine
COPY --from=builder /app/dist/ecommerce-frontend /usr/share/nginx/html
COPY nginx.conf /etc/nginx/conf.d/default.conf

EXPOSE 80
CMD ["nginx", "-g", "daemon off;"]
```

Nginx.conf:

```
server {
  listen 80;
  server_name _;

  root /usr/share/nginx/html;
  index index.html;

  # SPA routing support
  location / {
    try_files $uri $uri/ /index.html;
  }

  # Reverse proxy for Django API
  location /api/ {
    proxy_pass http://backend:8000/api/;
    proxy_set_header Host $host;
```

```
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
    }
}
```