**DevOps Workshop: Full Instruction Guide for Static Express App CI/CD Pipeline on Render**

This workshop walks participants through setting up a complete DevOps workflow for a static website served via an Express.js backend. It includes:

* Local development
* Git/GitHub setup
* Dockerization
* GitHub Actions CI/CD
* Render deployment

**1. Intro to DevOps & CI/CD**

1. **Local Express App Setup**
2. **GitHub Project Setup**
3. **Dockerization**
4. **CI/CD with GitHub Actions**
5. **Deploying to Render**
6. **Live Demo: Code → Push → Deploy**

**Local Project Setup**

1. Create your folder and files:

mkdir samplesite && cd samplesite

npm init -y

npm install express

1. Create the following files:
   * server.js
   * public/index.html
   * public/css/style.css
   * public/main/main.js
2. Basic server.js:

const express = require('express');

const path = require('path');

const app = express();

const PORT = process.env.PORT || 3000;

app.use(express.static(path.join(\_\_dirname, 'public')));

app.get('/health', (req, res) => res.send('OK'));

app.listen(PORT, () => {

console.log(`Server running on http://localhost:${PORT}`);

});

1. Test locally:

node server.js

Visit: [http://localhost:3000](http://localhost:3000/)

**GitHub Setup**

1. Initialize Git:

git init

git remote add origin https://github.com/YOUR\_USERNAME/YOUR\_REPO.git

git add .

git commit -m "initial commit"

git push -u origin main

1. **Dockerize the App**
2. Create a Dockerfile:

FROM node:18-alpine

WORKDIR /app

COPY . .

RUN npm install

EXPOSE 3000

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

1. Build and run:

docker build -t foodhub-app .

docker run -p 3000:3000 foodhub-app

1. **💡 Configure GitHub Actions CI/CD**

**A screen shot of a computer

AI-generated content may be incorrect.**

**4. Set Up Render Deploy Hook**

1. Go to [Render.com](https://render.com/) → Create a Web Service from your GitHub repo.
2. After service is deployed, go to **Settings** > **Deploy Hooks** > copy the URL.
3. In GitHub:
   * Go to Settings > Secrets > Actions
   * Add new secret:
     + Name: RENDER\_DEPLOY\_HOOK
     + Value: *(Paste the URL)*
4. **Test It All Live**
5. Make a small change (like a comment)
6. Push to main:

git add .

git commit -m "trigger deploy"

git push

Show:

* + GitHub actions tab pipeline running
  + Triggered build on render
  + Deployed app URL is updated

**Conclusion**

* Emphasize speed and automation benefits
* CI/CD increases confidence in every code push
* Easily extensible to include testing, Docker, Kubernetes, and monitoring