Ex. No. 12	Implementing a Blue-Green Deployment Strategy for a Node.js Application
25/10/2024	

AIM

To implement a Blue-Green Deployment strategy for a Node.js application using Jenkins, Docker, and Docker Hub to achieve zero-downtime deployment.

Algorithm

1. **Set Up the Node.js Application:**
    - Create the project directory and initialize a Node.js project.
    - Install dependencies (e.g., Express) and create a basic server.
2. **Create a Dockerfile:**
    - Set up the Dockerfile to use a Node.js base image and configure the web server.
3. **Build and Push Docker Image to Docker Hub:**
    - Build the Docker image locally and push it to Docker Hub for easy access during deployment.
4. **Configure Jenkins for Blue-Green Deployment:**
    - Set up Jenkins plugins, create a pipeline job, and set up credentials.
    - Expose Jenkins to the internet for GitHub webhooks.
5. **Define the Jenkins Pipeline Script:**
    - Write and configure the pipeline script in Jenkins to handle blue-green deployment.
6. **Verify and Test Zero-Downtime Deployment:**
    - Make a code change and trigger the pipeline, observing the blue and green environments in action.

Input

1. **Set up Node.js Project**

```
mkdir blue-green-deployment-app
cd blue-green-deployment-app
npm init -y
npm install express --save
```

2. **Create the Server File**

```
const express = require('express');
const app = express();
const PORT = 3000;
app.get('/', (req, res) => res.send('Hello from Blue-Green Deployment!'));
app.listen(PORT, () => console.log(`App running on
http://localhost:${PORT}`));
```

3. **Initialize Git Repository and Push to GitHub**

```
git init
git add .
git commit -m "Initial commit for Blue-Green Deployment Lab"
git remote add origin https://github.com/NakshathraP/Blue-Green.git
git push -u origin master
```

4. **Dockerfile Content**

```
FROM node:14
WORKDIR /usr/src/app
COPY package*.json ./
RUN npm install
COPY . .
EXPOSE 3000
CMD ["node", "server.js"]
```

5. **Jenkins Pipeline Script**

```
pipeline {
    agent any
    stages {
        stage('Clone Repository') {
            steps { ... }
        }
        stage('Build Docker Image') { steps { ... } }
        stage('Push Docker Image') { steps { ... } }
        stage('Deploy to Blue Environment') { steps { ... } }
        stage('Test Blue Deployment') { steps { ... } }
        stage('Switch to Green Environment') { steps { ... } }
    }
    post { always { ... } }
}
```

Output

1. **Setup Node.js Project**:

- Node.js application initialized, Express installed.
- Server file (`server.js`) created with response: "Hello from Blue-Green Deployment!"

```
C:\Users\naksh\Downloads>cd DevOpsCode

C:\Users\naksh\Downloads\DevOpsCode>mkdir blue-green-deployment-app

C:\Users\naksh\Downloads\DevOpsCode>   cd blue-green-deployment-app

C:\Users\naksh\Downloads\DevOpsCode\blue-green-deployment-app>   npm init -y

Wrote to C:\Users\naksh\Downloads\DevOpsCode\blue-green-deployment-app\packa
ge.json:

{
  "name": "blue-green-deployment-app",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [],
  "author": "",
  "license": "ISC"
}



C:\Users\naksh\Downloads\DevOpsCode\blue-green-deployment-app>   npm install
 express --save

added 65 packages, and audited 66 packages in 16s

13 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
```

2. **Dockerfile Created**:

- Dockerfile created, specifying the use of a Node.js base image.

```
Dockerfile > ...
1   FROM node:14
2   WORKDIR /usr/src/app
3   COPY package*.json ./
4   RUN npm install
5   COPY . .
6   EXPOSE 3000
7   CMD ["node", "server.js"]
```

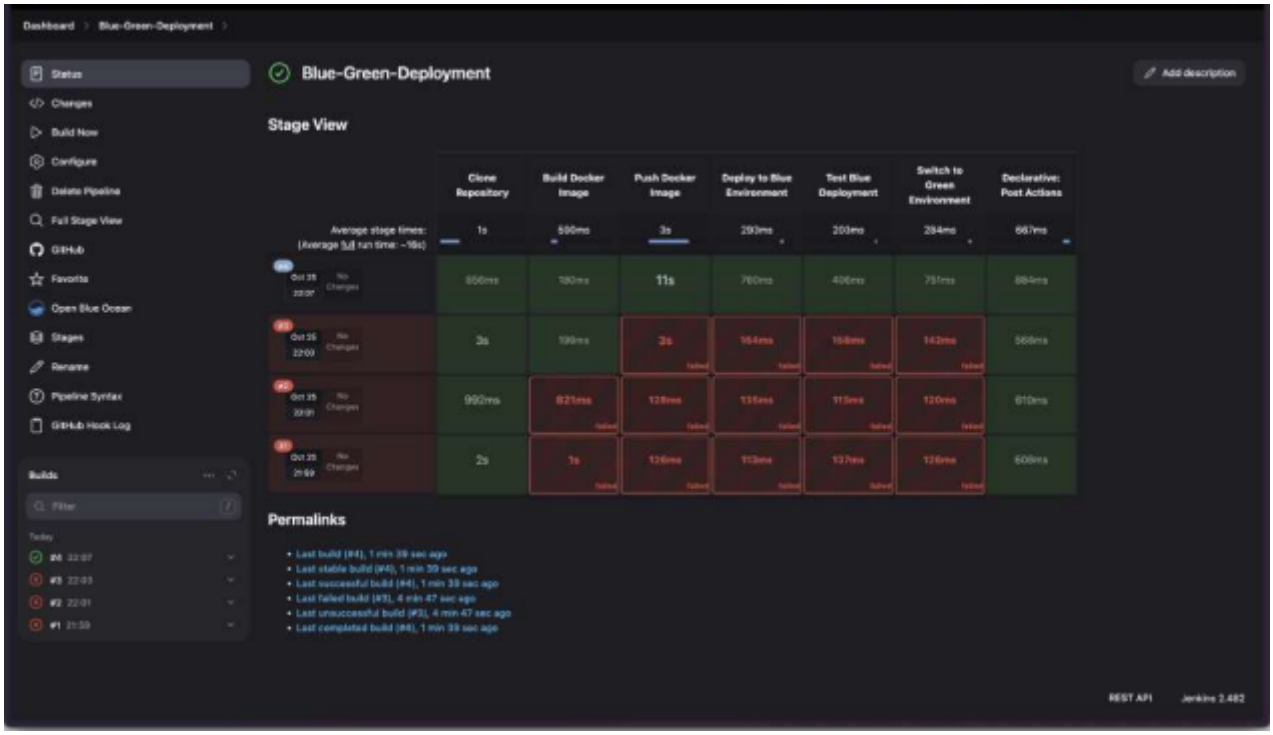
3. **Docker Image Built and Pushed to Docker Hub**:

- Docker image successfully built and pushed to Docker Hub.

4. **Jenkins Pipeline Configured**:

- Pipeline created in Jenkins, with webhooks enabled.

5. **Zero-Downtime Deployment Observed**:
    - Verified blue-green deployment with zero-downtime as the application switched between environments.



Result

The Blue-Green Deployment strategy was successfully implemented, ensuring zero-downtime deployment for the Node.js application by using Jenkins and Docker to alternate between blue and green environments.