Build and Deploy JavaScript Project with Nexus Artifact Storage

This guide explains how to build and deploy a JavaScript project on Server A, and store the generated build artifact in a Nexus Repository Manager hosted on Server B.

⬜ Server A (Build & Deploy)

Step 1: Install Prerequisites on Server A

Run the following on **Server A (build/deploy server)**:

```
sudo apt update && sudo apt upgrade -y
sudo apt install curl wget gnupg openjdk-11-jdk nodejs npm -y
```

Verify installation:

```
java -version
node -v
npm -v
```

Step 2: Clone or Copy the JavaScript Project

```
git clone https://github.com/your-org/your-js-project.git
cd your-js-project
```

Step 3: Build the Project

```
npm install
npm run build
```

This generates a `build/` or `dist/` directory containing the compiled project files.

Step 4: Deploy the App Locally (on Server A)

For a **frontend project (React, Vue, etc.):**

```
sudo apt install nginx -y
sudo rm -rf /var/www/html/*
sudo cp -r build/* /var/www/html/
sudo systemctl restart nginx
```

For a **Node.js backend (Express):**

```
npm install -g pm2
pm2 start index.js
```

Step 5: Create a .tgz Artifact for Nexus

```
npm pack
```

This creates a file like `your-app-name-1.0.0.tgz`.

⎙ Server B – Nexus Repository Manager

Step 1: Install Java and Nexus

```
sudo apt update
sudo apt install openjdk-11-jdk -y
cd /opt
sudo wget https://download.sonatype.com/nexus/3/latest-unix.tar.gz
sudo tar -xvzf latest-unix.tar.gz
sudo mv nexus-3* nexus
sudo useradd -r -m nexus
sudo chown -R nexus:nexus nexus
echo 'run_as_user="nexus"' | sudo tee /opt/nexus/bin/nexus.rc
```

Create systemd service:

```
sudo nano /etc/systemd/system/nexus.service
```

Paste:

```
[Unit]
Description=Nexus Repository
After=network.target

[Service]
Type=forking
ExecStart=/opt/nexus/bin/nexus start
ExecStop=/opt/nexus/bin/nexus stop
User=nexus
Restart=on-abort

[Install]
WantedBy=multi-user.target
```

Enable and start:

```
sudo systemctl daemon-reexec
sudo systemctl enable nexus
sudo systemctl start nexus
```

Access Nexus at `http://<server-B-ip>:8081`

Default credentials:

```
User: admin
Password: cat /opt/sonatype-work/nexus3/admin.password
```

Step 2: Create npm Hosted Repository

1. Open Nexus in the browser.
2. Go to **Settings → Repositories → Create Repository**
3. Select **npm (hosted)**
4. Name: `npm-hosted`
5. Deployment Policy: `Allow redeploy`
6. Save changes.

Repository URL:

```
http://<server-B-ip>:8081/repository/npm-hosted/
```

⬛ Step 6: Push Artifact from Server A → Nexus (Server B)

Step 1: Configure npm to use Nexus

Create `.npmrc`:

```
nano .npmrc
```

Add:

```
registry=http://<server-B-ip>:8081/repository/npm-hosted/
```

//<server-B-ip>:8081/repository/npm-hosted/:_authToken=<your-nexus-api-token>
```

### Step 2: Publish the artifact

```
npm publish ./your-app-name-1.0.0.tgz --registry=http://<server-B-ip>:8081/repository/npm-hosted/
```

Expected output:

```
+ your-app-name@1.0.0
```

⬇ Optional: Download and Deploy on Another Server

```
npm install your-app-name@1.0.0 --registry=http://<server-B-ip>:8081/repository/npm-hosted/
```

Ensure `.npmrc` is configured with the same Nexus credentials.

✅ Summary

Task	Done On	Command / Notes
Install Java, Node.js, npm	Server A	`apt install` commands
Build project	Server A	`npm install && npm run build`
Deploy locally	Server A	Copy to `/var/www/html/` or use `pm2`
Package as .tgz	Server A	`npm pack`
Install and run Nexus	Server B	Download, extract, and configure
Create npm hosted repo	Server B	Via Nexus UI
Configure npmrc and push .tgz	Server A	`npm publish` to Nexus repo
(Optional) Download artifact	Any server	`npm install` from Nexus

Would you like a script to automate build + publish on Server A or instructions to download manually without npm?