Deploying React + Node.js Using AWS Lambda & S3

1. Host React App on S3

- Build the React app using `npm run build` inside the client folder.
- Create an S3 bucket (enable static website hosting).
- Upload contents of `build/` to the S3 bucket.
- Add a bucket policy to allow public access.
- Access the app via S3 static website URL.

2. Deploy Node.js Backend to Lambda (Manual Setup)

- Create a new Lambda function (Node.js 18.x).
- Add handler code to respond to API Gateway requests.
- Create HTTP API in API Gateway and connect it to Lambda.
- Enable CORS in API Gateway settings.
- Use the API endpoint in the React app to fetch data.

3. Deploy Node.js Backend with Serverless Framework (Recommended)

- Install Serverless CLI: `npm install -g serverless`
- Initialize a project: `serverless create --template aws-nodejs`
- Write Lambda logic in `handler.js`
- Define configuration in `serverless.yml`
- Deploy using `serverless deploy`
- Use the returned API endpoint in your frontend.

4. Connect Frontend with Backend

- In React, call the Lambda API using axios or fetch.
- Example: `axios.get('https://your-api-id.amazonaws.com/dev/api/hello')`
- Ensure CORS headers are present in Lambda response.

5. Optimize & Secure

- Use CloudFront for HTTPS and global CDN caching.
- Set up Route 53 for a custom domain.
- Use Secrets Manager for environment variables.

- Configure IAM roles for secure Lambda execution.

Summary Table

Frontend: React app hosted on S3

Backend: Node.js deployed on AWS Lambda

Routing: API Gateway

Optional: CloudFront, Route 53, GitHub Actions for CI/CD