

EXPERIMENT-9

Name: Jayanaath S

Subject: Full Stack

Section: 23BCC-1

Subject Code:23CSP-339

UID: 23BCC70022

Date: 10-11-2025

- **Aim:**

To deploy a React application using **Nginx** as a web server, enabling it to be served efficiently over HTTP.

- **Theory:**

A React is a JavaScript library for building user interfaces. By default, React apps run on a development server using npm start. For production, the React app must be built into static files (HTML, CSS, JavaScript), which can be served by a web server.

Nginx is a high-performance web server that can serve static content, reverse proxy requests, and handle load balancing. Serving a React app through Nginx improves performance, enables caching, and makes it accessible via standard HTTP ports (like 80 or 443).

Key points:

- React builds into a /build folder containing index.html and static assets.
- Nginx serves index.html for all routes in single-page applications (SPA) using a try_files directive.
- Production deployment ensures optimized, minified assets.

- **Procedure:**

1. **Build the React App**

- Navigate to your React project folder:

```
cd my-react-app
```

- Build the production version:

```
npm run build
```

- This creates a build/ directory containing all static files.

2. **Install Nginx (if not installed)**

- For Ubuntu/Debian:

```
sudo apt update
```

```
sudo apt install nginx
```

3. Configure Nginx

- Open Nginx configuration file:

```
sudo nano /etc/nginx/sites-available/default
```

- Replace the server block with:

```
server {  
    listen 80;  
    server_name your_domain_or_IP;  
  
    root /var/www/my-react-app/build;  
    index index.html;  
  
    location / {  
        try_files $uri /index.html;  
    }  
}
```

`try_files $uri /index.html;` ensures React routing works properly.

4. Copy Build Files to Nginx Root

```
sudo cp -r build/* /var/www/my-react-app/
```

- Ensure proper permissions:

```
sudo chown -R www-data:www-data /var/www/my-react-app
```

5. Restart Nginx

```
sudo systemctl restart nginx
```

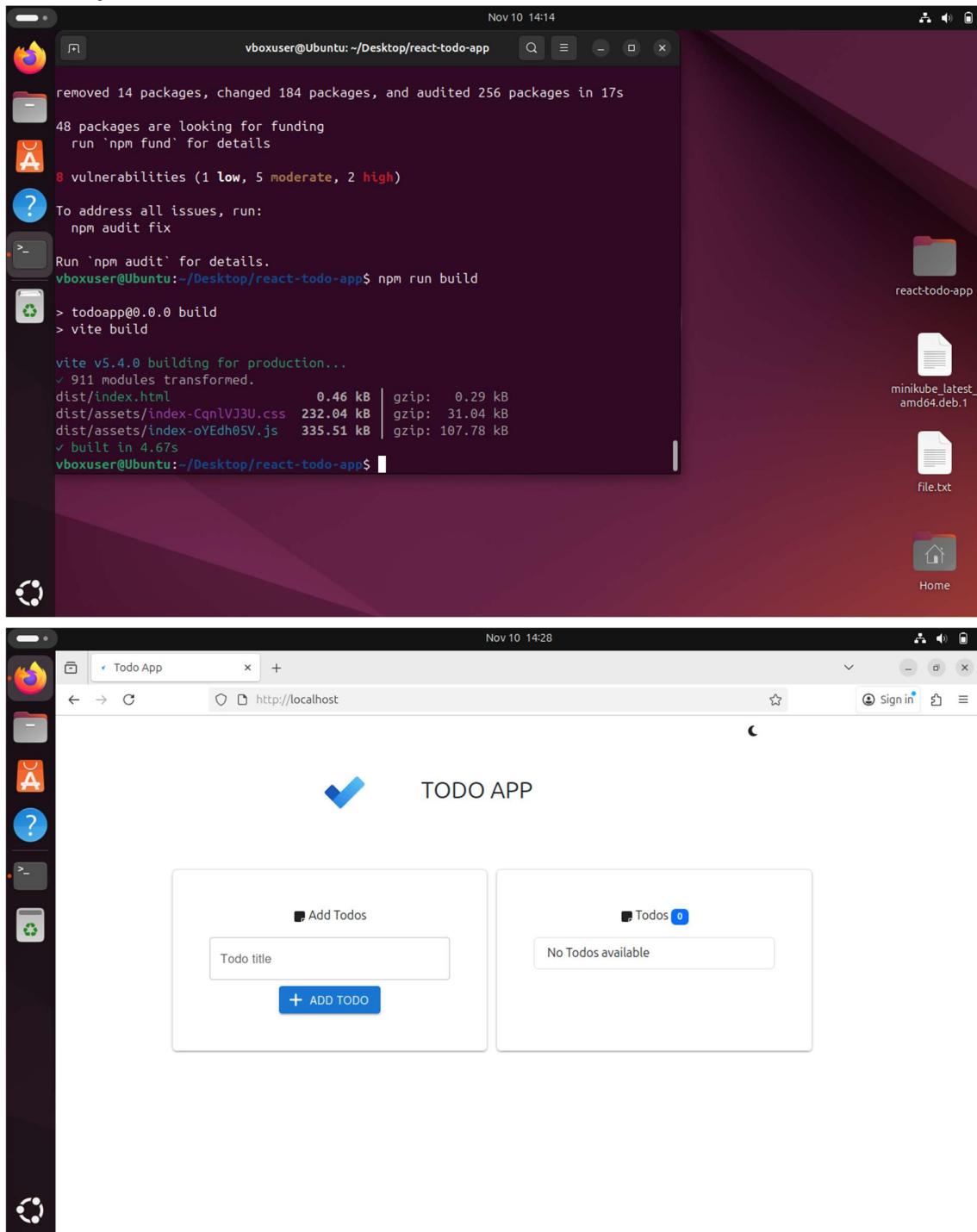
6. Test the Application

- Open a browser and visit `http://your_domain_or_IP`.
- The React app should load successfully

- **Result**

The React application was successfully deployed and served using Nginx. The app is accessible over HTTP, with all routes handled correctly. Static assets like CSS, JS, and images are served efficiently.

- **Output:**



A screenshot of a Linux desktop environment showing two windows. The top window is a terminal window titled 'vboxuser@Ubuntu:~/Desktop/react-todo-app'. It displays the output of an npm audit command, showing 14 packages removed, 184 packages changed, and 256 packages audited in 17s. It also shows 48 packages looking for funding and 8 vulnerabilities (1 low, 5 moderate, 2 high). The user is prompted to run 'npm audit fix' to address all issues. The bottom window is a web browser titled 'Todo App' showing the deployed React application at <http://localhost>. The application has a logo with a checkmark and the text 'TODO APP'. On the left, there's a form with a placeholder 'Todo title' and a blue 'ADD TODO' button. On the right, it says 'Todos 0' and 'No Todos available'.

```
removed 14 packages, changed 184 packages, and audited 256 packages in 17s
48 packages are looking for funding
  run 'npm fund' for details
8 vulnerabilities (1 low, 5 moderate, 2 high)

To address all issues, run:
  npm audit fix

Run `npm audit` for details.
vboxuser@Ubuntu:~/Desktop/react-todo-app$ npm run build

> todoapp@0.0.0 build
> vite build

vite v5.4.0 building for production...
✓ 911 modules transformed.
dist/index.html          0.46 kB | gzip:  0.29 kB
dist/assets/index-CqnlVJ3U.css 232.04 kB | gzip: 31.04 kB
dist/assets/index-oYEdh05V.js 335.51 kB | gzip: 107.78 kB
✓ built in 4.67s
vboxuser@Ubuntu:~/Desktop/react-todo-app$
```