

First push code to github

[render.com](#) has the free place to deploy the application.

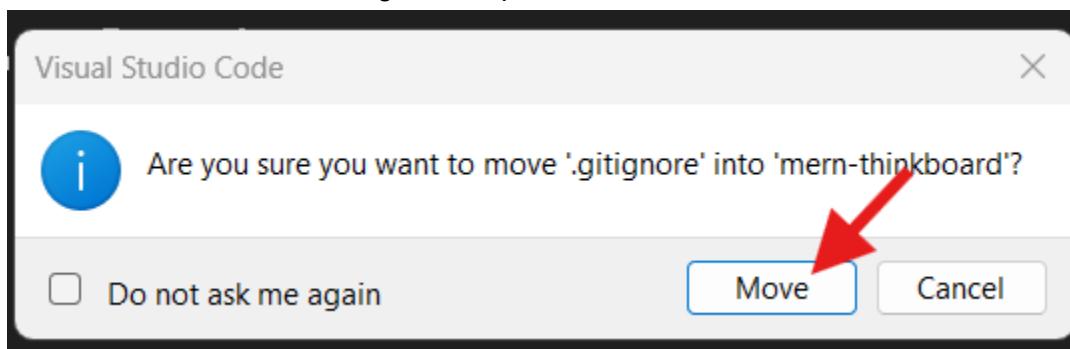
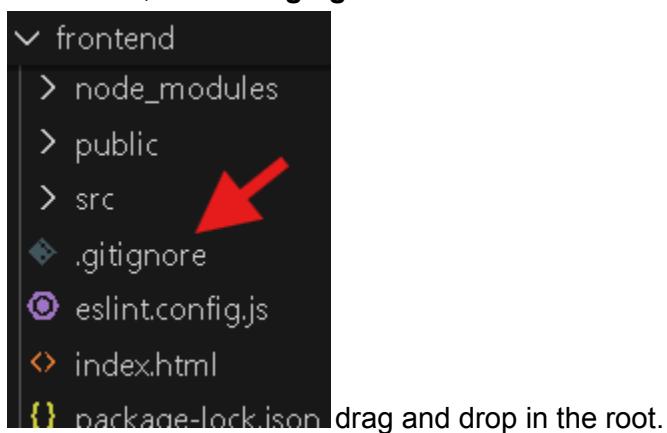
Github takes the code, and deploy in render platform.

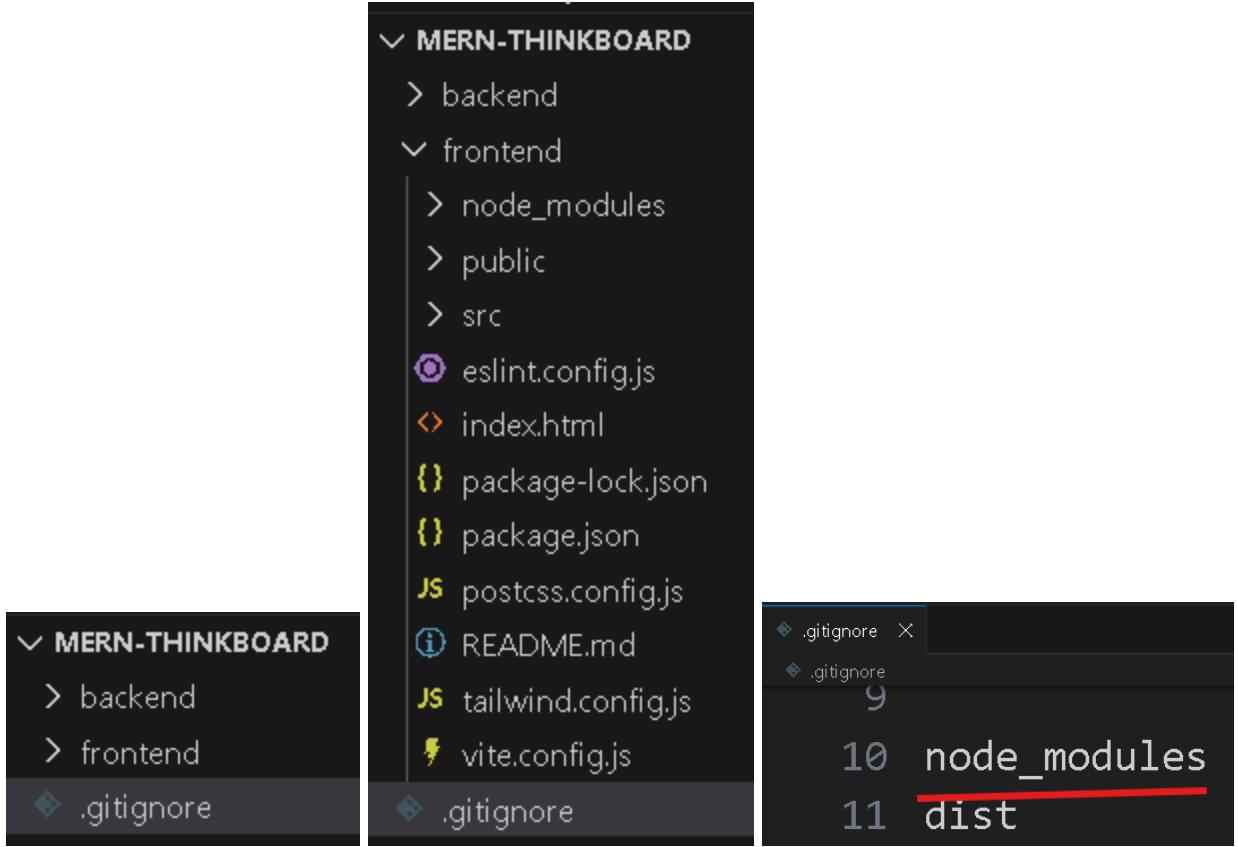
Need to hide .env file, during pushing to github. So no one can see credentials.

```
1 MONGO_URI=mongodb+srv://routhfamil  
2 PORT=5001  
3 UPSTASH_REDIS_REST_URL="https://fi  
4 UPSTASH_REDIS_REST_TOKEN="AadDAAIn
```

For this we need to create `.gitignore` file.

In frontend, we have `.gitignore` file:





now **gitignore** can be applied to both backend and frontend. **Add .env:**

```
24 *.sw?
25
26 .env
```

Stop the backend and frontend running.

In the root terminal:

The screenshot shows the VS Code interface. The left sidebar displays a file tree for a project named 'MERN-THINKBOARD'. It contains three main items: 'backend', 'frontend', and '.gitignore'. The 'backend' folder is expanded, showing its contents: 'node_modules', 'src', '.env', and 'package-lock.json'. The '.env' file is highlighted with a red arrow pointing to it. The bottom right corner of the file tree area has a small red dot. The terminal tab is active, showing a command-line prompt: 'PS C:\Users\kiran\OneDrive\Desktop\mern-thinkboard>'. Below the terminal, a code editor window is visible, containing the command 'git init' and its output: 'Initialized empty Git repository in c:/users/kiran/onedrive/desktop/mern-thinkboard/.git/'. The status bar at the bottom of the screen shows tabs for PROBLEMS, OUTPUT, DEBUG CONSOLE, TERMINAL (which is underlined), PORTS, and PLAYWRI.

Initialize by command: git init

```
PS C:\Users\kiran\OneDrive\Desktop\mern-thinkboard> git init
Initialized empty Git repository in c:/users/kiran/onedrive/desktop/mern-thinkboard/.git/
```

Gray colored are ignored: So when we push to github, these are ignored.

This screenshot shows the same file tree as the previous one, but with different file icons. The 'node_modules' and 'src' folders now have a grayed-out appearance, indicating they are ignored. The '.env' file also has a gray icon. A red arrow points to the '.env' file. The bottom right corner of the file tree area has a small red dot. The terminal tab is active, showing the command 'git init' and its output: 'Initialized empty Git repository in c:/users/kiran/onedrive/desktop/mern-thinkboard/.git/'.

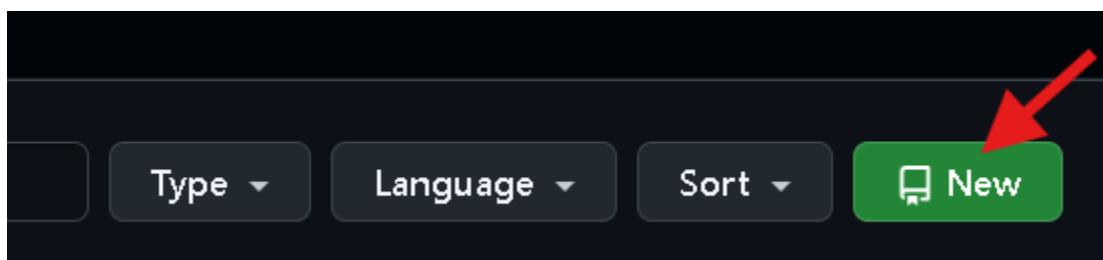
To add everything(end by dot): git add .

Committing with message: git commit -m "initial commit"

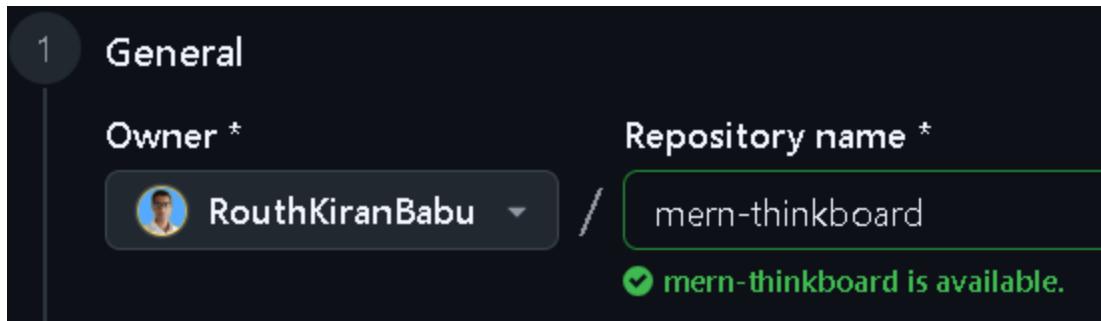
```
PS C:\Users\kiran\OneDrive\Desktop\mern-thinkboard> git commit -m "initial commit"
[master (root-commit) 472a2bf] initial commit
 31 files changed, 6190 insertions(+)
create mode 100644 .gitignore
create mode 100644 backend/package-lock.json
create mode 100644 backend/package.json
create mode 100644 backend/src/config/db.js
create mode 100644 backend/src/config/upstash.js
create mode 100644 backend/src/controllers/notesController.js
create mode 100644 backend/src/middleware/rateLimiter.js
create mode 100644 backend/src/models/Note.js
create mode 100644 backend/src/routes/notesRoutes.js
create mode 100644 backend/src/server.js
create mode 100644 frontend/README.md
create mode 100644 frontend/eslint.config.js
create mode 100644 frontend/index.html
create mode 100644 frontend/package-lock.json
create mode 100644 frontend/package.json
create mode 100644 frontend/postcss.config.js
create mode 100644 frontend/public/vite.svg
create mode 100644 frontend/src/App.jsx
create mode 100644 frontend/src/components/Navbar.jsx
create mode 100644 frontend/src/components>NoteCard.jsx
create mode 100644 frontend/src/components/NotesNotFound.jsx
create mode 100644 frontend/src/components/RateLimitedUI.jsx
create mode 100644 frontend/src/index.css
create mode 100644 frontend/src/lib/axios.js
create mode 100644 frontend/src/lib/utils.js
create mode 100644 frontend/src/main.jsx
create mode 100644 frontend/src/pages/CreatePage.jsx
```

Added to the staging area.

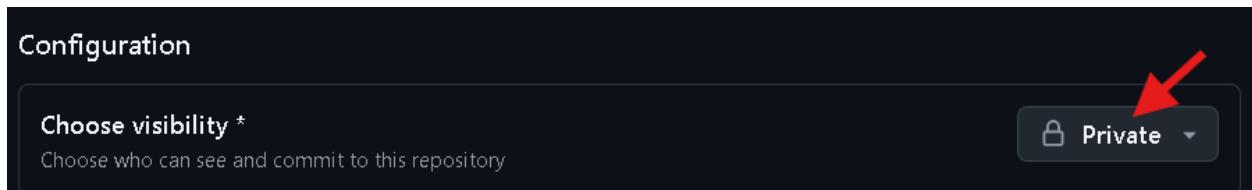
Create Github Repository:



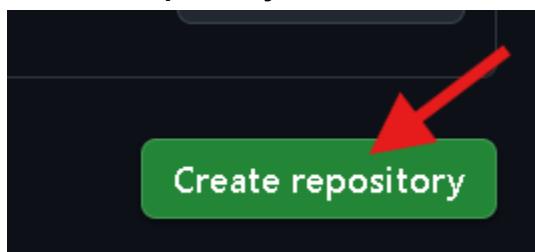
Provide name to the repository:



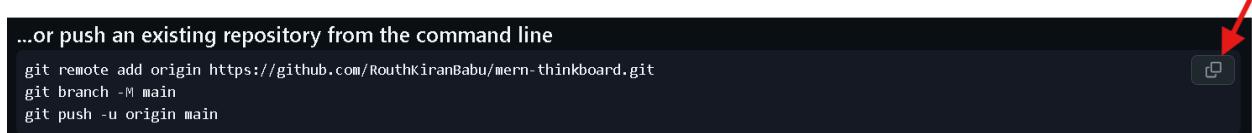
Initially make it as **private** then after publish make it as **public**:



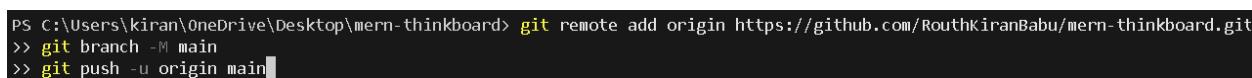
Create a Repository:



Copy the code to push to the repository:



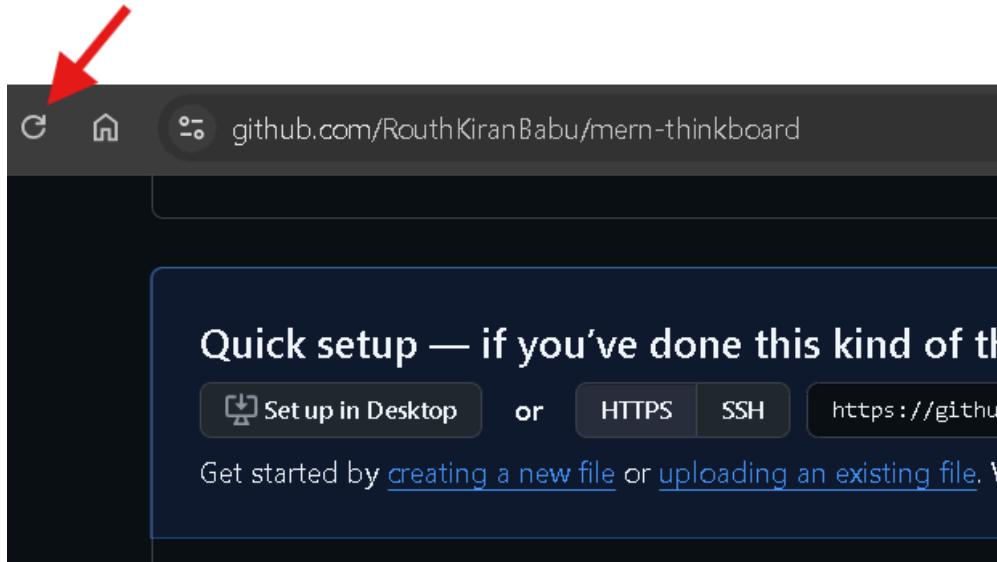
Paste in VSCode terminal:



Result:

```
Enumerating objects: 46, done.  
Counting objects: 100% (46/46), done.  
Delta compression using up to 16 threads  
Compressing objects: 100% (41/41), done.  
Writing objects: 100% (46/46), 57.92 KiB | 6.44 MiB/s, done.  
Total 46 (delta 1), reused 0 (delta 0), pack-reused 0 (from 0)  
remote: Resolving deltas: 100% (1/1), done.  
To https://github.com/RouthKiranBabu/mern-thinkboard.git  
 * [new branch]      main -> main  
branch 'main' set up to track 'origin/main'.  
PS C:\Users\kiran\OneDrive\Desktop\mern-thinkboard> □
```

Then refresh the page:



Has the **initial commit** message.

[Home](#) [mern-thinkboard](#) [github.com/RouthKiranBabu/mern-thinkboard](#)

RouthKiranBabu / **mern-thinkboard** 

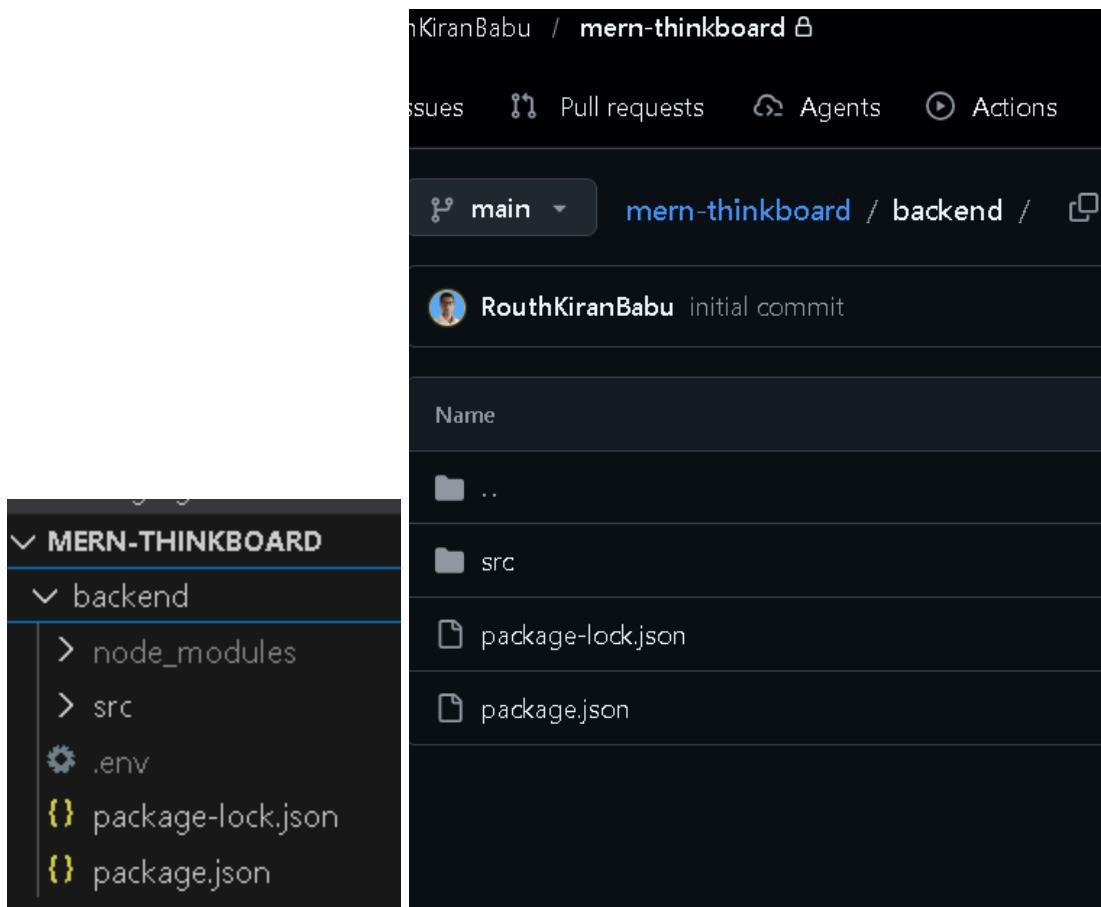
Issues Pull requests Agents Actions Projects

 **mern-thinkboard** Private

 **main**   1 Branch  0 Tags 

File	Commit
 RouthKiranBabu	initial commit
 backend	initial commit
 frontend	initial commit
 .gitignore	initial commit

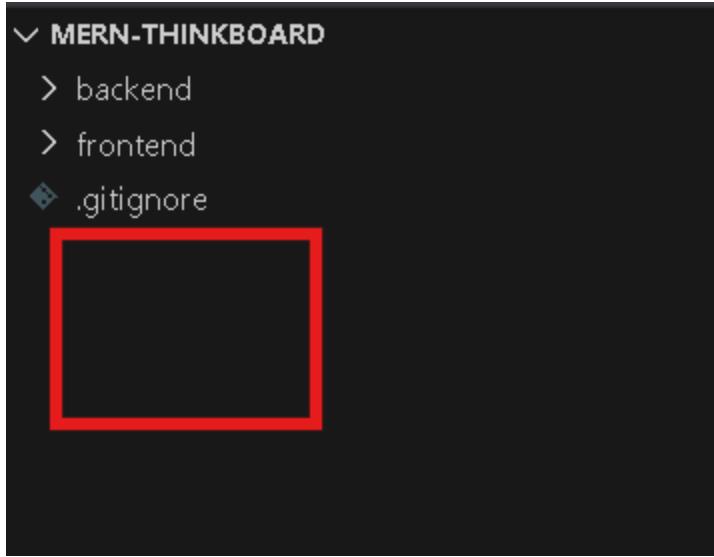
In **backend**:



Where **.env** and **node_modules** are ignored in github(its not available).

About render:

If render wants to deploy our app, then want to get the **node_modules**. Without that application would not work. So we can create a file, here



so that they can install the dependencies.

In terminal: `npm init -y`

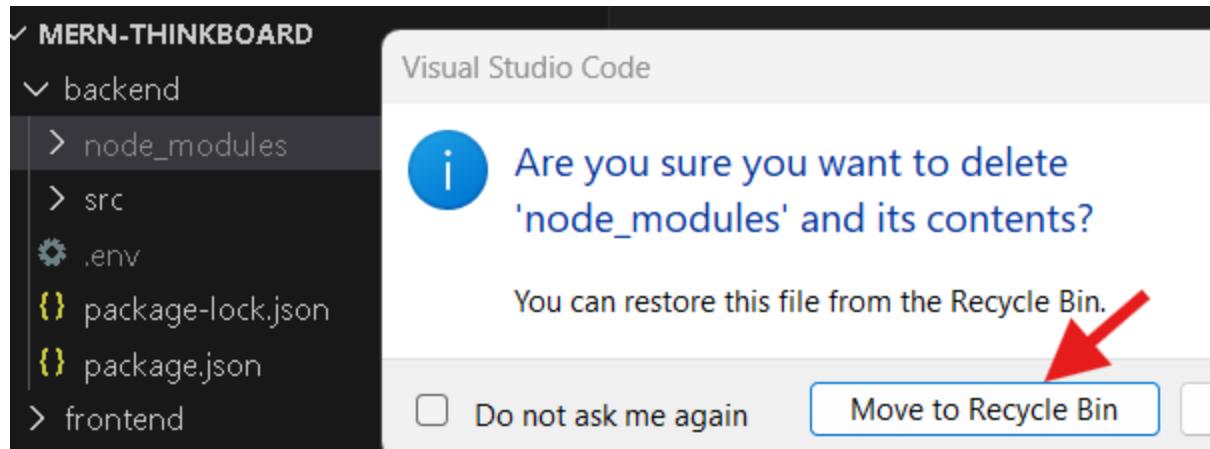
```
\Desktop\mern-thinkboard> npm init -y
e:\Desktop\mern-thinkboard\package.json
{
  "name": "MERN-THINKBOARD",
  "version": "1.0.0",
  "private": true,
  "scripts": {
    "start": "node backend.js"
  },
  "dependencies": {
    "express": "^4.17.1",
    "mongoose": "^5.7.1",
    "cors": "^2.8.5",
    "body-parser": "^1.19.0"
  }
}
```

Where we will have the build script, where we want to install **node_modules** for both backend and frontend.

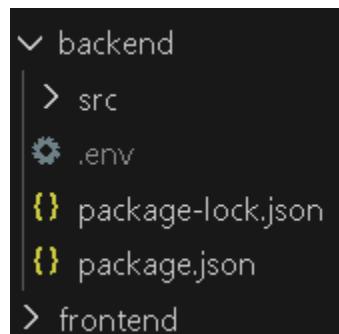
In `package.json`: `"build": "npm install --prefix backend && npm install --prefix frontend"`

```
0 package.json X
n > {} scripts > * build
  ▷ Debug
  "scripts": {
    "build": "npm install --prefix backend && npm install --prefix frontend"
  },
}
```

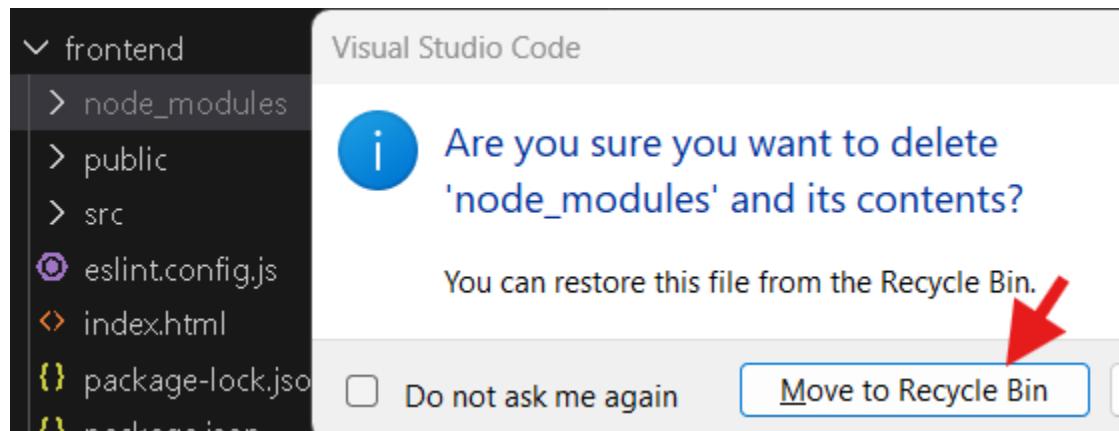
And try to delete **node_modules** in both frontend and backend.



Node_modules removed, from backend:



Now, Remove node_modules from frontend:



Node_modules removed:

```
✓ frontend
  > public
  > src
  ⚡ eslint.config.js
  ◀ index.html
  {} package-lock.json
  {} package.json
  JS postcss.config.js
  ⓘ README.md
  JS tailwind.config.js
  ⚡ vite.config.js
  ⚡ .gitignore
  {} package.json
```

Now let's try to run the command: `npm run build`

Which installs the dependencies for both backend as well as frontend.

```
PS C:\Users\kiran\OneDrive\Desktop\mern-thinkboard> npm run build

> mern-thinkboard@1.0.0 build
> npm install --prefix backend && npm install --prefix frontend

added 130 packages, and audited 132 packages in 2s
```

Then its adds `node_modules` at both backend as well as frontend.

```
✓ backend
  > node_modules 
  > src
  ⚡ .env
  {} package-lock.json
  {} package.json
  ✓ frontend
    > node_modules 
```

In production, we always want to build the frontend application, so that everything within frontend folder, will be optimised and put under a special folder called `dist`.

Once you install the dependencies, run the command **npm run built** but run this under the frontend **--prefix frontend**

```
"scripts": {  
  "build": "npm install --prefix backend && npm install --prefix frontend && npm run build --prefix frontend"  
},
```

Code is:

```
"scripts": {  
  "build": "npm install --prefix backend && npm install --prefix frontend && npm run build --prefix frontend"  
}
```

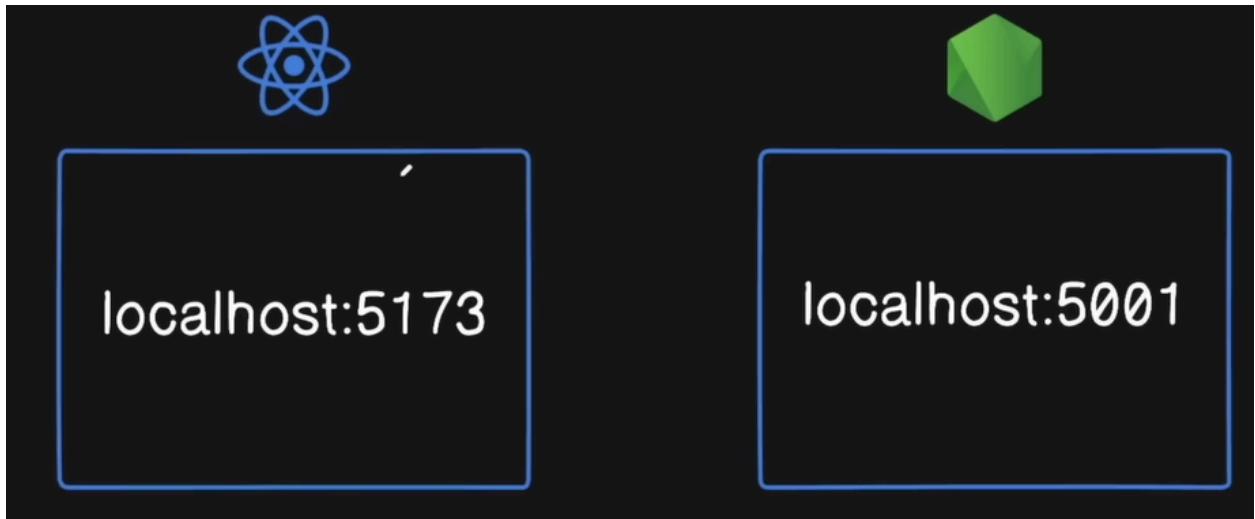
Run the code in terminal: npm run build

```
\mern-thinkboard> npm run build  
  
vite v7.3.1 building client environment for production...  
  
📦 daisyUI 4.12.24  
└─ ✓ 1 theme added https://daisyuui.com/docs/themes  
   ❤️ Support daisyUI project: https://opencollective.com/daisyuui  
  
✓ 1777 modules transformed.  
dist/index.html          0.46 kB | gzip: 0.29 kB  
dist/assets/index-DLQ72rmS.css 37.42 kB | gzip: 6.98 kB  
dist/assets/index-jueUIET7.js 286.99 kB | gzip: 95.26 kB  
✓ built in 2.83s  
PS C:\Users\kiran\OneDrive\Desktop\mern-thinkboard>
```

```
└─ frontend  
    └─ dist  
        > assets  
        < index.html  
        📜 vite.svg  
        > node modules
```

dist folder, which is the optimised version of react application had build.

Now we want to deploy everything.



React application under port **5173**, api under port **5001**.

So bring the react under the same domain like the image below:



If we have both client under the same domain, we can get rid of CORS errors

Example

You have a frontend at `http://localhost:3000`

And an API backend at: `http://api.example.com`

For two domain, we have to setup the CORS

For one domain, we can get rid of CORS configurations.

In server.js from backend:

Write little bit code, to deploy the code correctly.

Import path package: no need to do npm install

```
1 import path from "path"
```

We are going to use different middleware coming from express:

Just below routes

```
37 app.use("/api/notes", notesRoutes)
38
39 app.use(express.static(path.join()))
```

Need to provide path from frontend dist folder.

Create variable:

```
16 const PORT = process.env.PORT || 5001
17 const __dirname = path.resolve()
```



Now need to go one **up**, to visit frontend, then one **into it** then **dist** folder.

_dirname: under the backend

./: for one up

/frontend/dist: to visit dist folder

```
38 app.use("/api/notes", notesRoutes)
39 |
40 app.use(express.static(path.join(__dirname, "../frontend/dist")))
```

app.use(express.static(path.join(__dirname, "../frontend/dist"))))

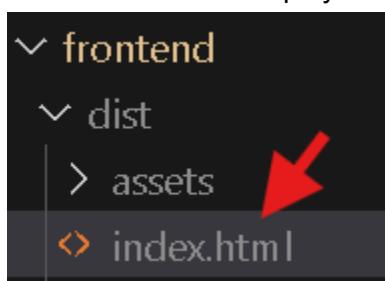
Which actually says serve dist as the static asset.

If we get any request other than: app.use("/api/notes", notesRoutes)

We want to serve our react application:

React application will be started at: index.html

Where we want to display



We want to do this, under production: when application under render.com

app.use(express.static(path.join(__dirname, "../frontend/dist"))))

```
app.get("*", (req, res) => {
  res.sendFile(path.join(__dirname, "../frontend", "dist", "index.html"))
})
38 app.use("/api/notes", notesRoutes)
39 |
40 app.use(express.static(path.join(__dirname, "../frontend/dist")))
41 |
42 app.get("*", (req, res) => {
43   res.sendFile(path.join(__dirname, "../frontend", "dist", "index.html"))
44 })
```

If it is in production: use if statement

```
if(process.env.NODE_ENV === "production"){
    app.use(express.static(path.join(__dirname, "../frontend/dist")))

    app.get("*", (req, res) => {
        res.sendFile(path.join(__dirname, "../frontend", "dist", "index.html"))
    })
}
```

Code is:

```
if(process.env.NODE_ENV === "production"){
    app.use(express.static(path.join(__dirname, "../frontend/dist")))

    app.get("*", (req, res) => {
        res.sendFile(path.join(__dirname, "../frontend", "dist", "index.html"))
    })
}
```

Also do that for CORS configuration: if its not in production

```
21 // First try to remove the cors error, then use middleware
22 // removing error in home page url
23 // if it's not in production, do the cors configuration
24 if(process.env.NODE_ENV !== "production"){
25     app.use(
26         cors({
27             origin: "http://localhost:5173",
28         })
29     );
30 }
```

Code is:

```
// First try to remove the cors error, then use middleware
// removing error in home page url
// if it's not in production, do the cors configuration
if(process.env.NODE_ENV !== "production"){
    app.use(
        cors({
            origin: "http://localhost:5173",
        })
    );
}
```

Code in server.js:

```
import express from "express"
import dotenv from "dotenv"
import cors from "cors"
import path from "path"

import notesRoutes from "./routes/notesRoutes.js"
import { connectDB } from "./config/db.js"
import rateLimiter from "./middleware/rateLimiter.js"

dotenv.config()

//console.log(process.env.MONGO_URI)

const app = express()
// if process.env.PORT is undefined then PORT = 5001(by default value)
const PORT = process.env.PORT || 5001
const __dirname = path.resolve()

//connectDB()

// First try to remove the cors error, then use middleware
// removing error in home page url
// if it's not in production, do the cors configuration
if(process.env.NODE_ENV !== "production"){
    app.use(
        cors({
            origin: "http://localhost:5173",
        })
    );
}

// middleware
app.use(express.json()) // this middleware will parse JSON bodies: req.body
app.use(rateLimiter)

// Our simple custom middleware
// app.use((req, res, next) => {
//     console.log(`Req method: ${req.method}.\\nReq URL: ${req.url}.`)
//     next()
// })

app.use("/api/notes", notesRoutes)
```

```

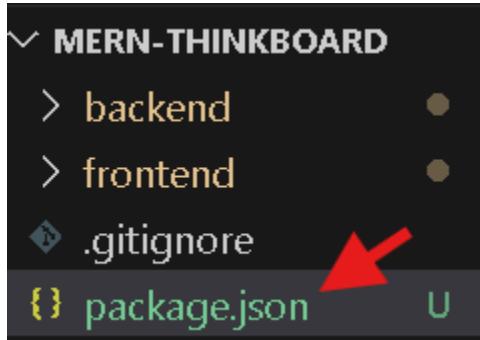
if(process.env.NODE_ENV === "production"){
    app.use(express.static(path.join(__dirname, "../frontend/dist")))
}

app.get("*", (req, res) => {
    res.sendFile(path.join(__dirname, "../frontend", "dist", "index.html"))
})

connectDB().then(() =>{
    app.listen(PORT, () => {
        console.log("Server started on PORT:", PORT)
    })
})

```

Once we build our application, we want to start it: so we use the start command that render will run



A screenshot of a code editor displaying the 'package.json' file from the 'MERN-THINKBOARD' project. The file contains the following JSON configuration:

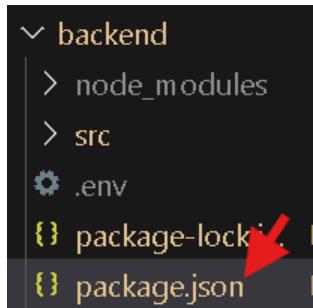
```

{
  "scripts": {
    "build": "npm install --prefix backend && cd frontend && npm install",
    "start": "npm run start --prefix backend"
  }
}

```

The 'start' script is underlined with a red color, indicating it is the script being run.

Under the backend, it runs the script:



it runs this script:

```
"scripts": {  
  "dev": "nodemon src/server.js",  
  "start": "node src/server.js"  
},
```

which starts the application.

We can test it out without deploying,

In terminal: **npm run build**

```
mern-thinkboard> npm run build
```

Which is going to build our application. Install all dependencies.

```
vite v7.3.1 building client environment for production...  
  
📦 daisyUI 4.12.24  
└─ ✓ 1 theme added https://daisyui.com/docs/themes  
   ❤️ Support daisyUI project: https://opencollective.com/daisyui  
  
✓ 1777 modules transformed.  
dist/index.html          0.46 kB | gzip: 0.29 kB  
dist/assets/index-DLQ72rmS.css 37.42 kB | gzip: 6.98 kB  
dist/assets/index-jueUIET7.js 286.99 kB | gzip: 95.26 kB  
✓ built in 2.78s
```

Code in .env:

In [render.com](#):

Set NODE_ENV=development

```
5 NODE_ENV=development
```

For testing:

Set NODE_ENV=production

```
backend > ⚙ .env
2 PORT=5001
3 UPSTASH_REDIS_REST_U
4 UPSTASH_REDIS_REST_T
5 NODE_ENV=production
```

Code is:

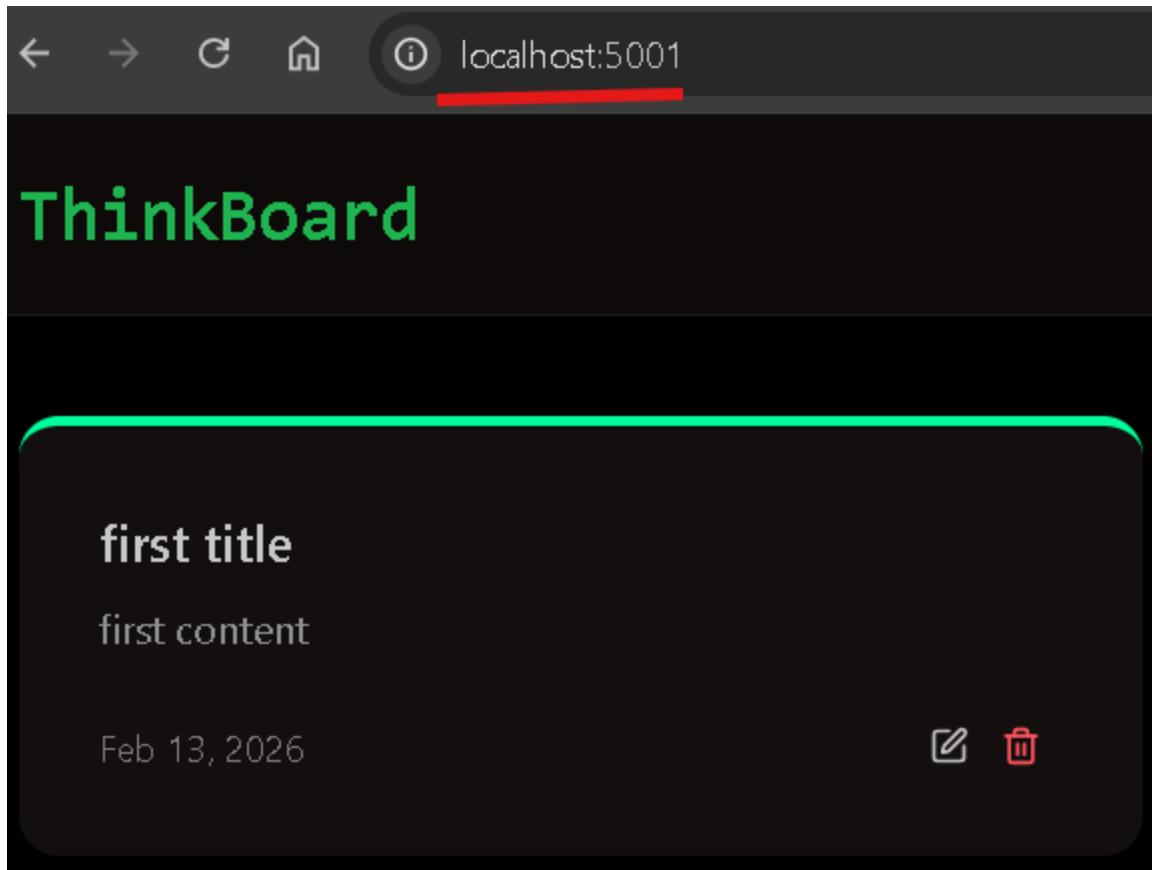
```
MONGO_URI=mongodb+srv://routhfamily123_db_user:dRCgH5MOBbEmJAW@cluster0.dxejp
0q.mongodb.net/notes_db?appName=Cluster0
PORT=5001
UPSTASH_REDIS_REST_URL="https://fit-boar-4819.upstash.io"
UPSTASH_REDIS_REST_TOKEN="AadDAAIncDI2YTMOTQwZTA3MTM0ZDk1YjgwMWUzYm
UwN2RiOGU3YXAyNDI4MTk"
NODE_ENV=development
```

Run the application:

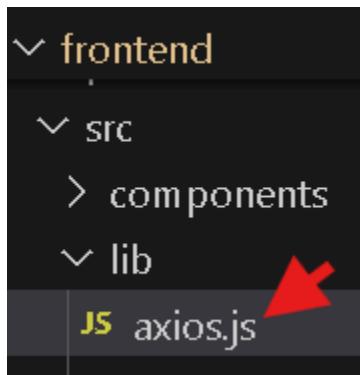
Command in terminal: **npm run start**

```
op\mern-thinkboard> npm run start
```

Visit: <http://localhost:5001/>



When we deploy the application, we get the different **URL**, we don't know what render gives. In production we don't have something like **localhost**. So need to make dynamic.



Code in [axios.js](#):

```
import axios from "axios"
```

```
const BASE_URL = import.meta.env.MODE === "development" ? "http://localhost:5001/api" :  
"/api"
```

```
const api = axios.create({  
  baseURL: BASE_URL,
```

})

export default api

In .env file:

5 NODE_ENV=development

Now commit our changes to github:

For adding to github: git add .

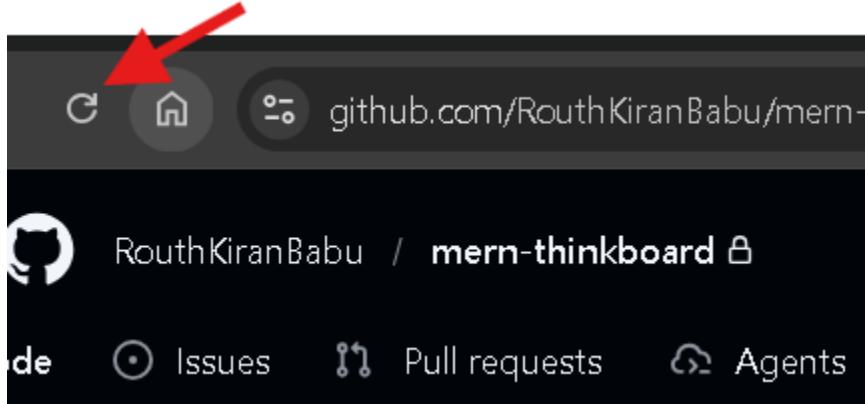
```
c:\p\mern-thinkboard> git add .
```

For commit with message: git commit -m "Prepared for the deployment"

```
c:\p\mern-thinkboard> git commit -m "Prepared for the deployment"
```

```
c:\p\mern-thinkboard> git push
```

Refresh the page:



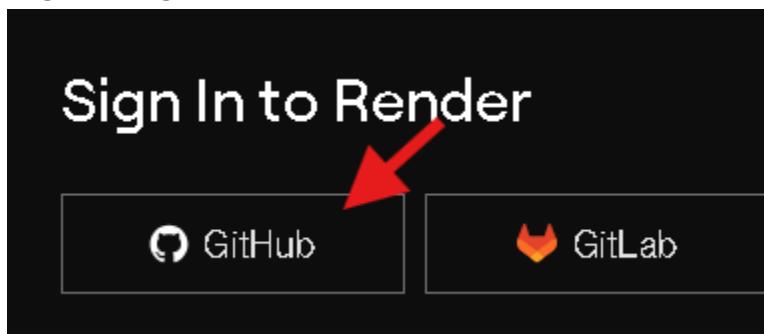
 RouthKiranBabu	Prepared for the deployment
 backend	Prepared for the deployment
 frontend	Prepared for the deployment
 .gitignore	initial commit
 package.json	Prepared for the deployment

Render.com:

Visit: <https://render.com/>



Login with github account:



Render by **Render** would like permission to:



Verify your GitHub identity (RouthKiranBabu)



Know which resources you can access



Act on your behalf

[\(?\) Learn more](#)

Resources on your account



Email addresses (read)

View your email addresses

[Learn more about Render](#)

[Cancel](#)

[Authorize Render](#)



Create an account

One last step — please confirm your email below.

Email

routhfamily123@gmail.com

By signing up you agree to our terms of service and privacy policy.

Create Account



Almost there!

We've sent you an email at **routhfamily123@gmail.com**.

Please follow the instructions in the email.

Resend Verification Email



Check gmail:

Hi there,

Welcome to Render! Please verify your email address by clicking the button below:

[Verify your email >](#)



This link will expire in 24 hours.

Create a new workspace

Workspaces are shared areas where teams deploy and o

What would you like to call your workspace? Optional

My workspace

What will you use this workspace for?

 Work

 Personal projects

Other

How many developers (including yourself) will be working together?

Just me

2-10

11-50

51-150

More than 150

Next →



Tell us about what you're building

We'll use this to tailor your onboarding experience

What are you looking to build with Render? Optional

Website / landing page

AI-native app

E-commerce

Prototype / MVP

Side project

Backend

Background

What capabilities matter most for your project on Render? Optional

AI/ML

Observability

Scalability

Compliance

Developer velocity

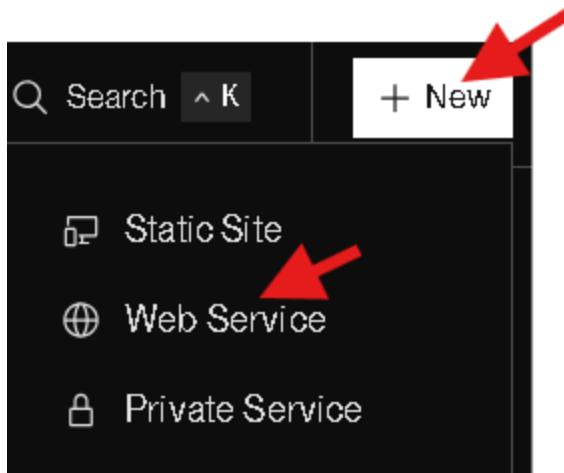
Low-downtime migration

Other

Finish →

Skip Survey

Now lets deploy the repository:



The interface shows a navigation bar at the top with tabs: 'Git Provider' (selected), 'Public Git Repository', and 'Existing Image'. The main content area has a dark background. In the center, the text 'No repositories found' is displayed in bold. Below it, a message says 'Connect your Git provider to deploy from your existing repositories.' Two buttons are shown: 'GitLab' with a red heart icon and 'Bitbucket' with a blue square icon. A red arrow points to the 'GitHub' button below. The GitHub button has a white background with a black border, a black octocat icon, and the word 'GitHub'.

No repositories found

Connect your Git provider to deploy from your existing repositories.

GitLab Bitbucket

Configure your Git provider to give Render permission to access your repositories.

GitHub

Install on your personal account Routh Kiran Babu

for these repositories:

All repositories

This applies to all current and future repositories owned by the user.
Also includes public repositories (read-only).

Only select repositories

Select at least one repository. Also includes public repositories.

with these permissions:

- ✓ **Read** access to Dependabot alerts, code, and metadata
- ✓ **Read** and **write** access to actions, checks, commit status, deployments, environments, issues, pull requests, releases, and workflows

User permissions

Render can also request users' permission to the following resources. These permissions will be requested and authorized on an individual-user basis.

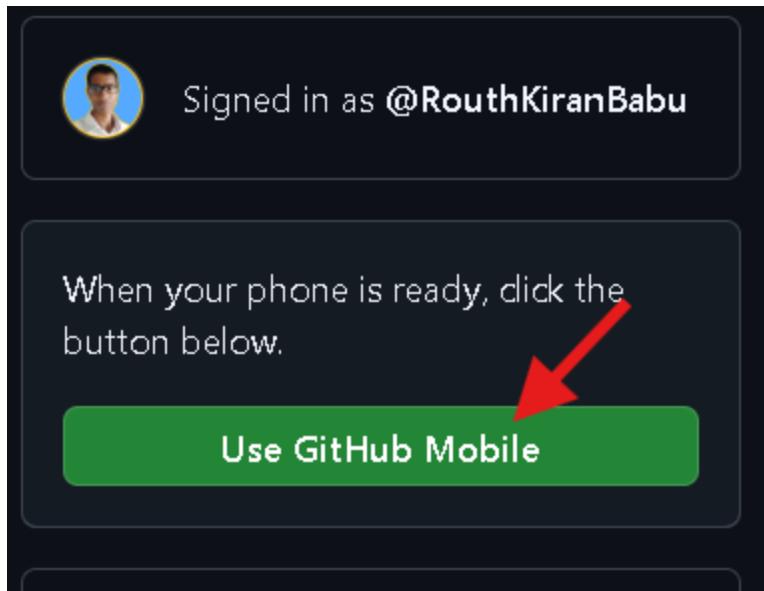
- ✓ **Read** access to email addresses



Install

[Cancel](#)

You will be redirected to the GitHub App's site to complete setup.



Install github in mobile, open with internet connection

Source Code RouthKiranBabu / mern-thinkboard • 41m ago

```
{ package.json X upstash.js axios.js } scripts > abc start
  description : "main": "index.js"
  ▷ Debug
  "scripts": {
    "build": "npm run build",
    "start": "npm start"
  },
}
```

A screenshot of a GitHub repository page. At the top, there are links for "Source Code" and the repository owner "RouthKiranBabu / mern-thinkboard" with a timestamp of "41m ago". Below this, a file list shows "package.json X", "upstash.js", and "axios.js". Under the "scripts" section, there is a comment "▷ Debug" followed by the script definitions: "build": "npm run build" and "start": "npm start". The "build" command is underlined in red, and the "start" command is also underlined in red.

So, make

Build Command

Render runs this command to build your app before each deploy.

```
$ npm run build
```

Start Command

Render runs this command to start your app with each deploy.

```
$ npm run start
```

For hobby projects

Free

\$0 / month

512 MB (RAM)

0.1 CPU



Copy everything from .env file:

```
package.json X upstash.js axios.js .env X server.js  
.env  
MONGO_URI=mongodb+srv://routhfamily12  
PORT=5001  
UPSTASH_REDIS_REST_URL="https://fit-b  
UPSTASH_REDIS_REST_TOKEN="AadDAAIncDI  
NODE_ENV=development
```

Paste it in:

Environment Variables

Set environment-specific config and secrets (such as API keys), then read those values from your code. [Learn more](#).

MONGO_URI

.....

PORT

.....

UPSTASH_REDIS_REST_URL

.....

UPSTASH_REDIS_REST_TOKEN

.....

NODE_ENV

.....

+ Add Environment Variable

Add from .env

No need to provide NODE_ENV:

NODE_ENV

development



Environment Variables

Set environment-specific config and secrets (such as API keys), then read those values from your code. [Learn more](#).

MONGO_URI

.....

PORT

.....

UPSTASH_REDIS_REST_URL

.....

UPSTASH_REDIS_REST_TOKEN

.....

+ Add Environment Variable

Add from .env

For free plan:

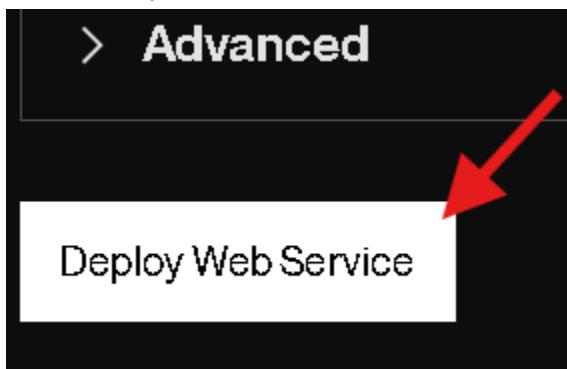
It becomes inactive after 15 minutes, eg: you deployed the application, but didn't visited the application. It will go inactive, if you visit the application it will take about 1 minute to load.

⚠ Upgrade to enable more features

Free instances spin down after periods of inactivity. They do not support SSH access, scaling, one-off jobs, or persistent disks.

Select any paid instance type to enable these features.

Then deploy the web service:



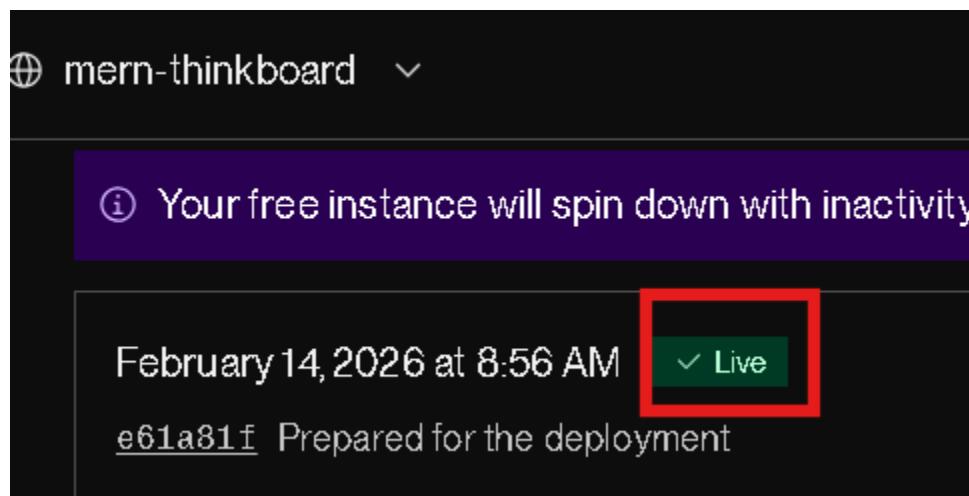
Will take around 2 minutes:

```
Feb 14
08:56:56 AM    ==> Cloning from https://github.com/RouthKiranBabu/mern-thinkboard
08:56:57 AM    ==> Checking out commit e61a81fe532562ed73868bb6957189d98412f3e1 in branch main
08:57:01 AM    ==> Using Node.js version 22.22.0 (default)
08:57:01 AM    ==> Docs on specifying a Node.js version: https://render.com/docs/node-version
08:57:05 AM    ==> Running build command 'npm run build'...
08:57:05 AM
08:57:05 AM    > mern-thinkboard@1.0.0 build
08:57:05 AM    > npm install --prefix backend && npm install --prefix frontend && npm run build --prefix frontend
08:57:05 AM
08:57:07 AM
08:57:07 AM    added 130 packages, and audited 132 packages in 2s
08:57:07 AM
08:57:07 AM    18 packages are looking for funding
08:57:07 AM        run 'npm fund' for details
08:57:08 AM
✖ 08:57:08 AM    9 vulnerabilities (3 low, 1 moderate, 4 high, 1 critical)
```

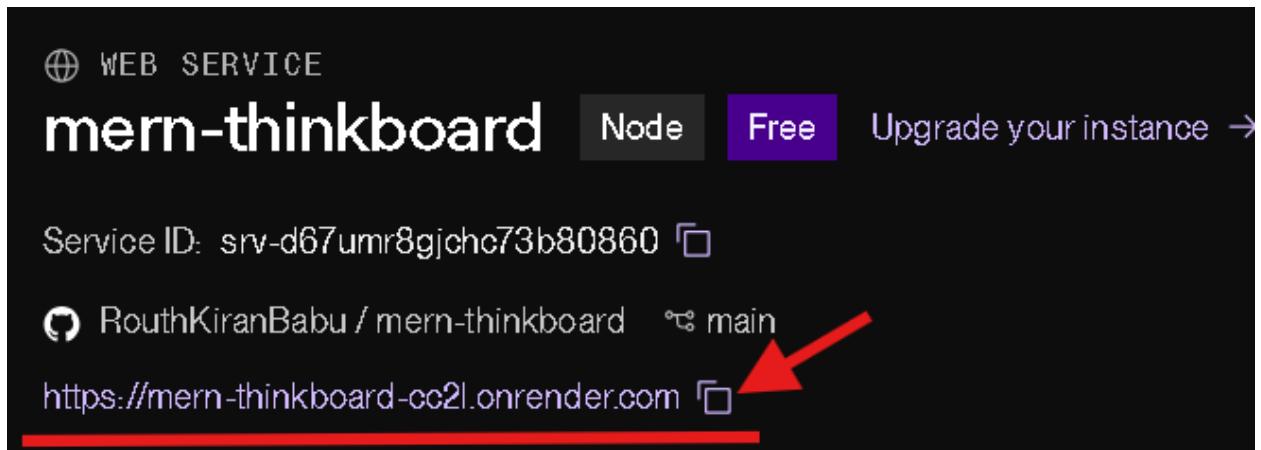
```
08:57:08 AM          npm audit fix
08:57:08 AM
08:57:08 AM          Run `npm audit` for details.
08:57:13 AM
08:57:13 AM          added 251 packages, and audited 253 packages in 5s
08:57:13 AM
08:57:13 AM          56 packages are looking for funding
08:57:13 AM          run `npm fund` for details
08:57:13 AM
08:57:13 AM          1 high severity vulnerability
08:57:13 AM
08:57:13 AM          To address all issues, run:
08:57:13 AM          npm audit fix
08:57:13 AM
08:57:13 AM          Run `npm audit` for details.
08:57:13 AM
```

```
Feb 14
08:57:13 AM          🌟 daisyUI 4.12.24
08:57:14 AM          |- ✓ i theme added      https://daisyyui.com/docs/themes
08:57:14 AM          |└ ★ Star daisyUI on GitHub https://github.com/saadeghi/daisyyui
08:57:14 AM
08:57:16 AM          ✓ 1777 modules transformed.
08:57:16 AM          rendering chunks...
08:57:16 AM          computing gzip size...
08:57:16 AM          dist/index.html           0.46 kB | gzip: 0.29 kB
08:57:16 AM          dist/assets/index-DLQ72rmS.css 37.42 kB | gzip: 6.98 kB
08:57:16 AM          dist/assets/index-B4u-Tkeg.js 286.98 kB | gzip: 95.25 kB
08:57:16 AM          ✓ built in 3.00s
08:57:42 AM          ==> Uploading build...
08:57:49 AM          ==> Uploaded in 4.1s. Compression took 2.3s
08:57:49 AM          ==> Build successful 🎉
08:58:03 AM          ==> Deploying...
08:58:03 AM          ==> Setting WEB_CONCURRENCY=1 by default, based on available CPU
```

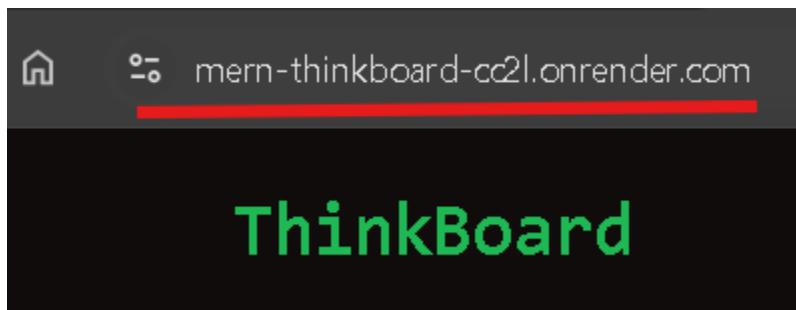
```
08:58:03 AM      => Setting max_concurrent=1 by default, based on available CPU
08:58:16 AM [wfk4c] ==> Running 'npm run start'
08:58:17 AM [wfk4c]
08:58:17 AM [wfk4c] > mern-thinkboard@1.0.0 start
08:58:17 AM [wfk4c] > npm run start --prefix backend
08:58:17 AM [wfk4c]
08:58:18 AM [wfk4c]
08:58:18 AM [wfk4c] > backend@1.0.0 start
08:58:18 AM [wfk4c] > node src/server.js
08:58:18 AM [wfk4c]
08:58:24 AM [wfk4c] MongoDB connected Successfully...
08:58:24 AM [wfk4c] Server started on PORT: 5001
08:58:34 AM      ==> Your service is live 🎉
08:58:34 AM      ==>
08:58:34 AM      ==> /////////////////////////////////
```



Got a Link:

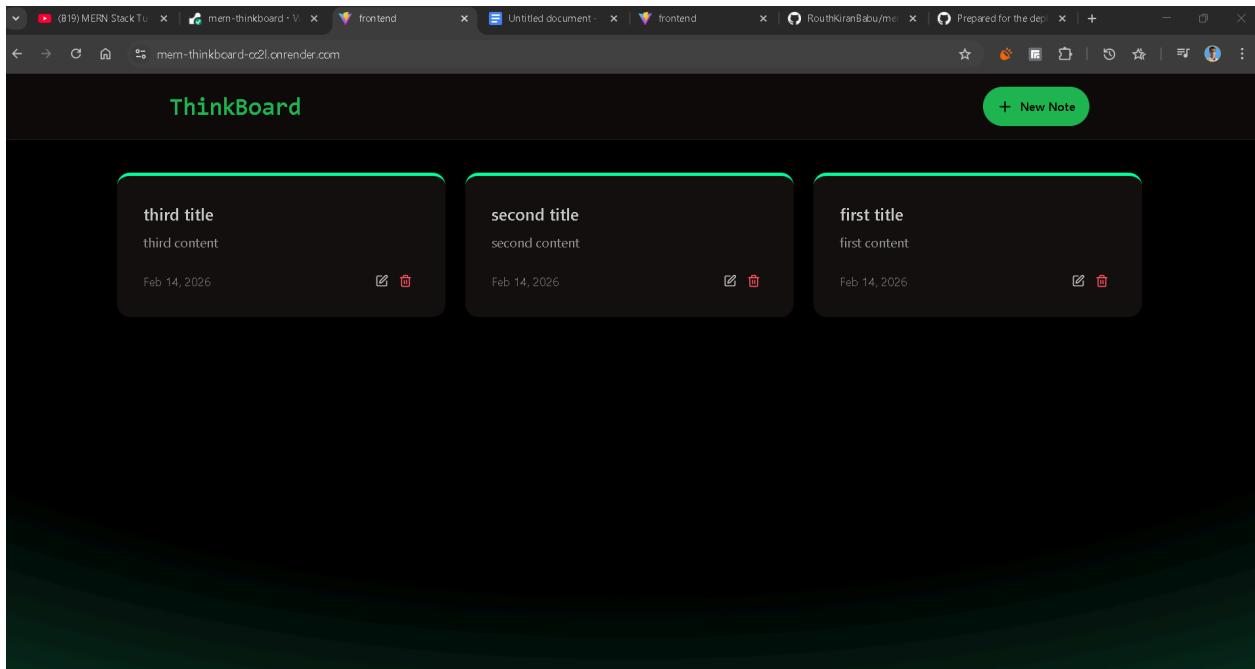


React Application is live at given link below, and Opens live:

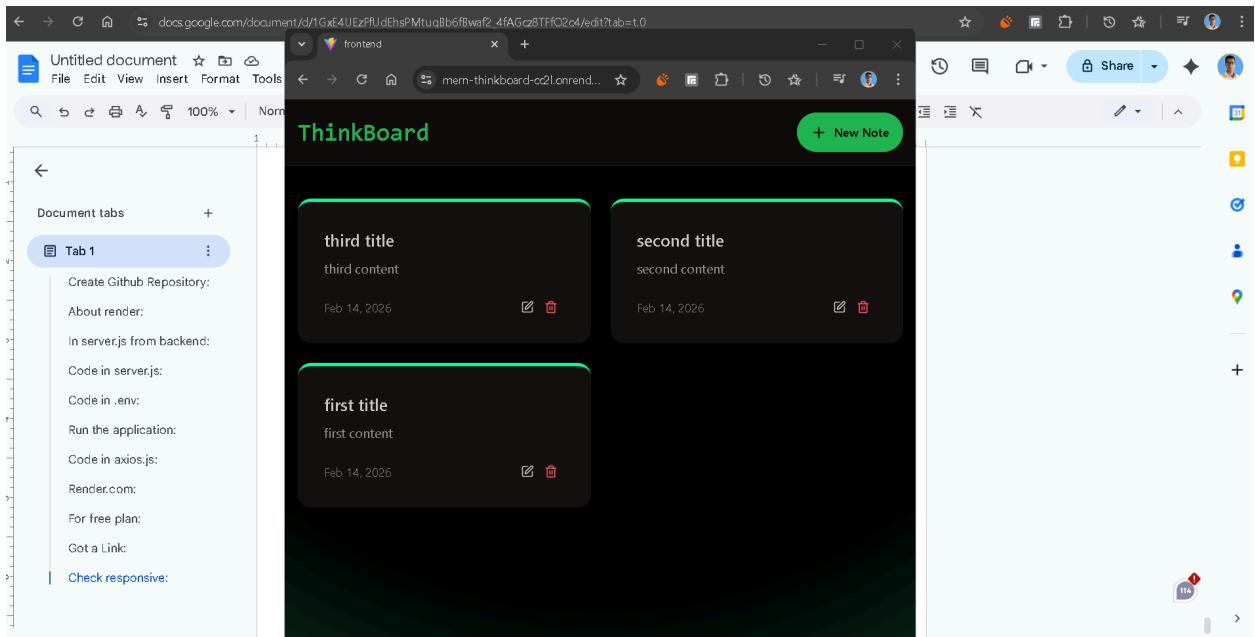


Check responsive:

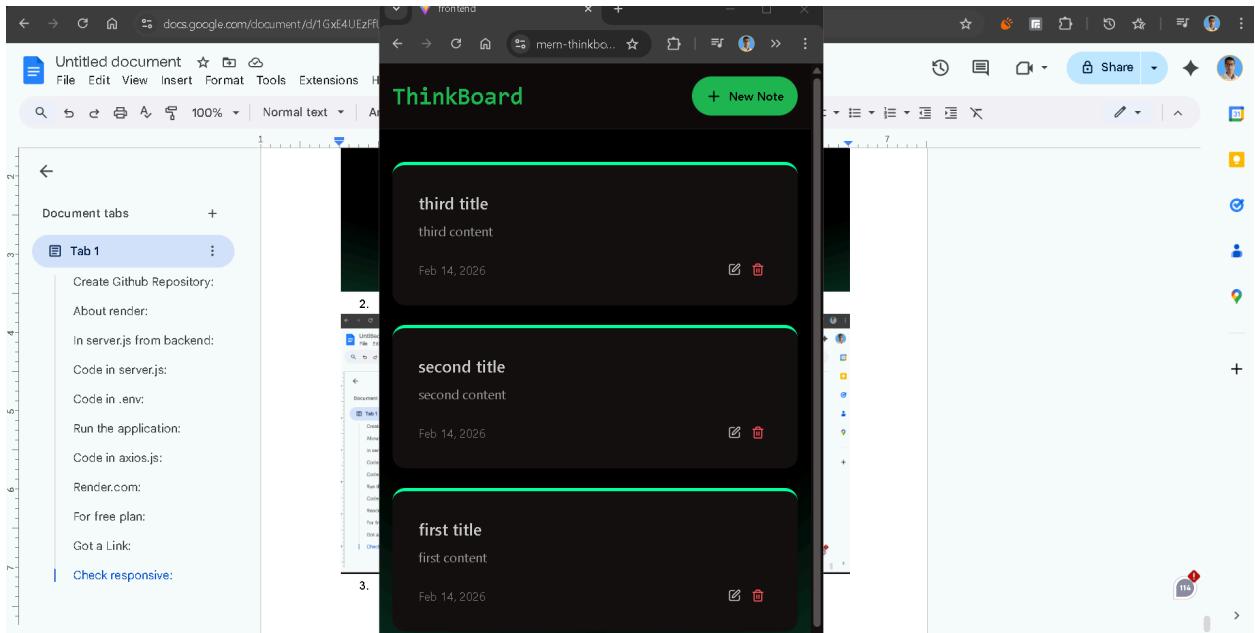
1. Full width



2. Medium Width



3. Minimum width



Change in Code: Does Effects render

In [upstash.js](#), changing the rate limiter.

```
const ratelimit = new Ratelimit({
  redis: Redis.fromEnv(),
  limiter: Ratelimit.slidingWindow(101, "60 s")
})
```

```
JS upstash.js M X
backend > src > config > JS upstash.js > ...
6
7 // create a ratelimiter that allows 100 requests per minute
8 const ratelimit = new Ratelimit({
9   redis: Redis.fromEnv(),
10  limiter: Ratelimit.slidingWindow(101, "60 s")
11})
12
```

Now to upload to github:

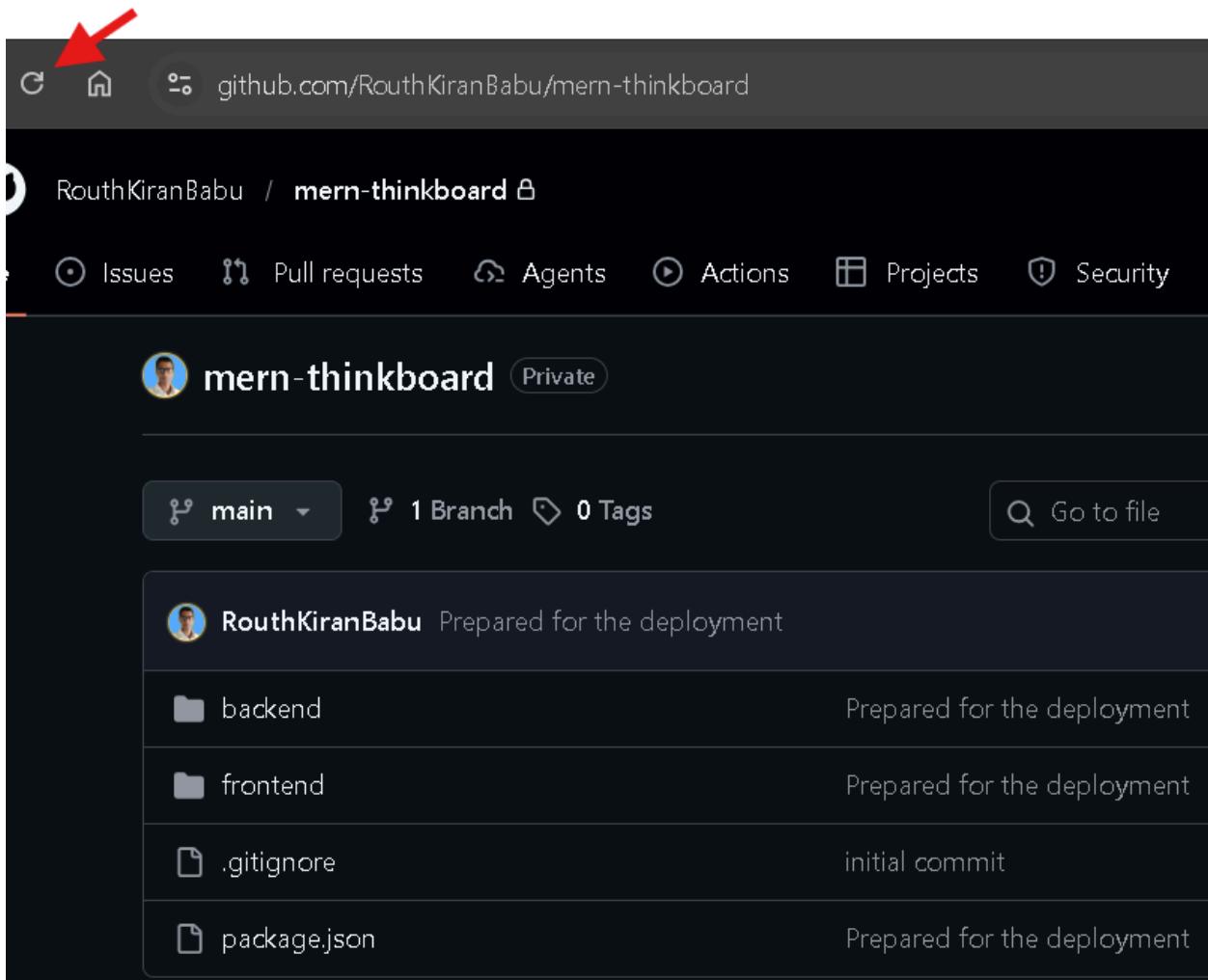
Code in terminal:

```
\mern-thinkboard> git add .
\mern-thinkboard> git commit -m "upated rate Limit from 100 to 101."
[master 12]
\mern-thinkboard> git push
```

Result:

```
+--o 01 (use --depth=1 if you are on a slow connection)  git push  
Enumerating objects: 11, done.  
Counting objects: 100% (11/11), done.  
Delta compression using up to 16 threads  
Compressing objects: 100% (6/6), done.  
Writing objects: 100% (6/6), 518 bytes | 518.00 KiB/s, done.  
Total 6 (delta 4), reused 0 (delta 0), pack-reused 0 (from 0)  
remote: Resolving deltas: 100% (4/4), completed with 4 local objects.  
To https://github.com/RouthKiranBabu/mern-thinkboard.git  
  e61a81f..253a0d4 main -> main  
PS C:\Users\kiran\OneDrive\Desktop\mern-thinkboard> █
```

When you make any changes, render see those changes from github account,
Refresh the page to see changes:



The screenshot shows a GitHub repository page for 'mern-thinkboard'. At the top, there are navigation icons for 'Commit' (highlighted with a red arrow), 'Home', and 'Repository'. The repository URL 'github.com/RouthKiranBabu/mern-thinkboard' is displayed. Below the header, the repository name 'RouthKiranBabu / mern-thinkboard' is shown, along with links for 'Issues', 'Pull requests', 'Agents', 'Actions', 'Projects', and 'Security'. The main content area displays the repository's structure: 'main' branch (1 Branch, 0 Tags), a file list, and deployment status for 'backend', 'frontend', '.gitignore', and 'package.json'. Deployment status for 'backend' and 'frontend' is 'Prepared for the deployment', while '.gitignore' and 'package.json' are 'initial commit'.

File	Status
backend	Prepared for the deployment
frontend	Prepared for the deployment
.gitignore	initial commit
package.json	Prepared for the deployment

mern-thinkboard (Private)

main 1 Branch 0 Tags Go to file

RouthKiranBabu upated rate Limit from 100 to 101.

backend upated rate Limit from 100 to 101.

So it wil try to deploy it, load it

SERVICE NAME	STATUS	RUNTIME	REGION	UPDATED
⊕ mern-thinkboard	✓ Deployed	Node	Oregon	<1m

To get latest version.

SERVICE NAME 1

⊕ mern-thinkboard

Logs available in it:



Deploy live for 253a0d4: upaded rate Limit from 100 to 101.

February 14, 2026 at 9:17 AM



Deploy started for 253a0d4: upaded rate Limit from 100 to 101.

New commit via Auto-Deploy

February 14, 2026 at 9:15 AM



Deploy live for e61a81f: Prepared for the deployment

February 14, 2026 at 8:58 AM



First deploy started for e61a81f: Prepared for the deployment

February 14, 2026 at 8:56 AM