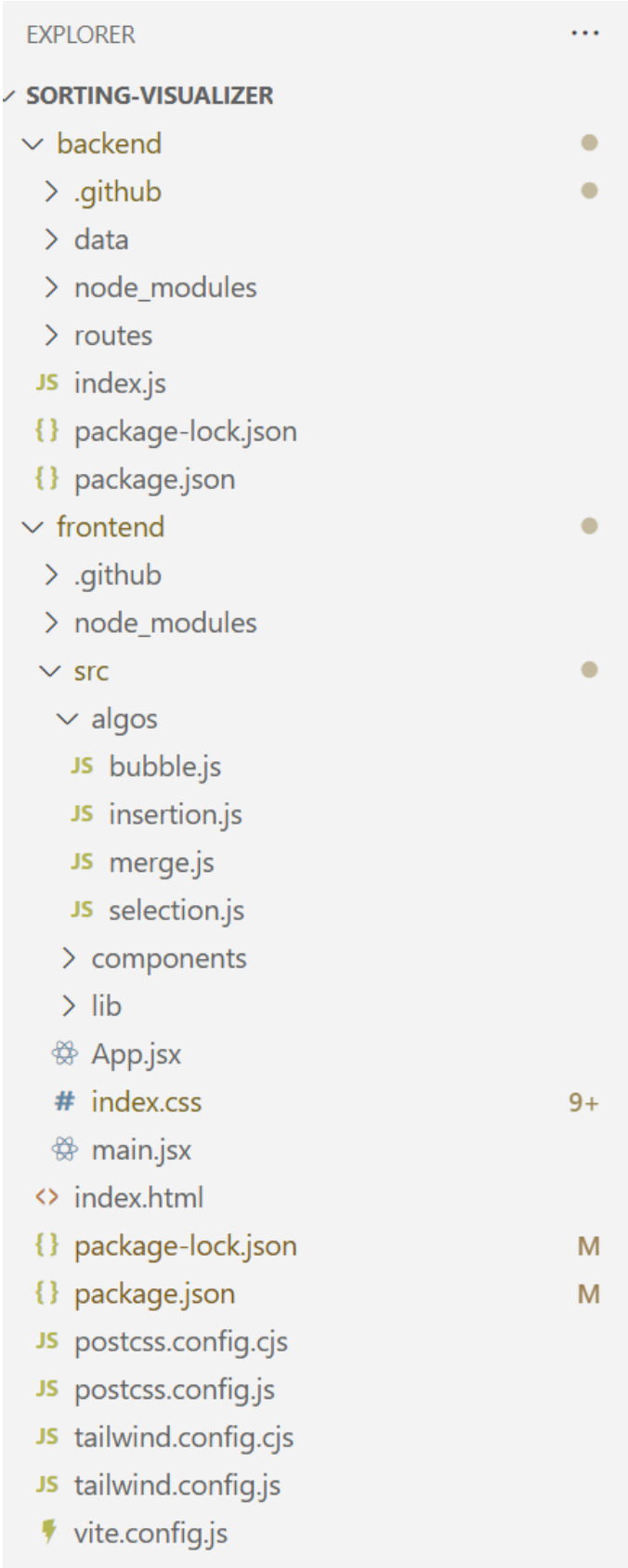
File Structure



CODE:

Backend:

index.js

backend > index.js > ...

```
const express = require('express')
const cors = require('cors')
const bodyParser = require('body-parser')
const logsRouter = require('./routes/logs')

const app = express()
const PORT = process.env.PORT || 5000

// Middleware
app.use(cors())
app.use(bodyParser.json())

// Routes
app.use('/api/logs', logsRouter)

// Health check
app.get('/', (req, res) => {
  res.send('Sorting Visualizer Backend Running!')
})

app.listen(PORT, () => console.log(`Server running on port ${PORT}`))
```

logs.js

backend > routes > logs.js > ...

```
const express = require('express')
const fs = require('fs')
const path = require('path')

const router = express.Router()
const filePath = path.join(__dirname, '../data/logs.json')

// GET all logs
router.get('/', (req, res) => {
  try {
    const logs = JSON.parse(fs.readFileSync(filePath, 'utf8'))
    res.json(logs)
  } catch (err) {
    console.error(err)
    res.status(500).json({ error: 'Failed to read logs' })
  }
})

// POST a new log
router.post('/', (req, res) => {
  try {
    const logs = JSON.parse(fs.readFileSync(filePath, 'utf8'))
    const newLog = req.body
    logs.push(newLog)
    fs.writeFileSync(filePath, JSON.stringify(logs, null, 2))
    res.status(201).json({ message: 'Log saved!' })
  } catch (err) {
    console.error(err)
    res.status(500).json({ error: 'Failed to save log' })
  }
})

module.exports = router
```

logs.json

backend > data > logs.json > ...


```
[
  {
    "algorithm": "Bubble",
    "size": 40,
    "duration": 45767,
    "timestamp": "2025-10-11T20:17:27.698Z"
  },
  {
    "algorithm": "Bubble",
    "size": 10,
    "duration": 2336,
    "timestamp": "2025-10-11T20:23:04.693Z"
  },
  {
    "algorithm": "Bubble",
    "size": 10,
    "duration": 2129,
    "timestamp": "2025-10-11T20:23:10.750Z"
  },
  {
    "algorithm": "Selection",
    "size": 10,
    "duration": 1623,
    "timestamp": "2025-10-11T20:23:15.965Z"
  },
  {
    "algorithm": "Insertion",
    "size": 10,
    "duration": 1463,
    "timestamp": "2025-10-11T20:23:41.626Z"
  },
  {
    "algorithm": "Bubble",
    "size": 40,
    "duration": 36132,
    "timestamp": "2025-10-11T20:25:18.578Z"
  },
  {
    "algorithm": "Bubble",
    "size": 10,
    "duration": 2310,
    "timestamp": "2025-10-11T20:30:25.888Z"
  },
  {
    "algorithm": "Bubble",
    "size": 36,
    "duration": 30857,
    "timestamp": "2025-10-11T20:31:38.157Z"
  },
  {
    "algorithm": "Bubble",
    "size": 10,
    "duration": 2027,
    "timestamp": "2025-10-11T20:32:30.895Z"
  },
  {
    "algorithm": "Selection",
    "size": 10,
    "duration": 1685,
    "timestamp": "2025-10-11T20:33:03.101Z"
  },
  {
    "algorithm": "Insertion",
    "size": 10,
    "duration": 2095,
    "timestamp": "2025-10-11T20:33:37.780Z"
  },
  {
    "algorithm": "Merge",
    "size": 10,
    "duration": 1651,
    "timestamp": "2025-10-11T20:33:50.552Z"
  },
  {
    "algorithm": "Merge",
    "size": 10,
    "duration": 1698,
    "timestamp": "2025-10-11T20:33:55.775Z"
  },
  {
    "algorithm": "Bubble"
```

Frontend:

SortingVisualizer.jsx

frontend > src > components > SortingVisualizer.jsx > SortingVisualizer > run

```jsx
import React, { useState, useEffect } from 'react'
import { bubble } from '../algos/bubble'
import { selection } from '../algos/selection'
import { insertion } from '../algos/insertion'
import { mergeSort } from '../algos/merge'
import { motion } from 'framer-motion'
import { logRun } from '../lib/api'

const ALGS = { Bubble: bubble, Selection: selection, Insertion: insertion, Merge: mergeSort }

function rand(n, max = 220) {
  const a = []
  for (let i = 0; i < n; i++) a.push(Math.floor(Math.random() * max) + 6)
  return a
}

export default function SortingVisualizer() {
  const [arr, setArr] = useState(() => rand(40))
  const [size, setSize] = useState(40)
  const [speed, setSpeed] = useState(30)
  const [alg, setAlg] = useState('Bubble')
  const [running, setRunning] = useState(false)
  const [result, setResult] = useState(null)

  useEffect(() => setArr(rand(size)), [size])

  function reset() {
    if (running) return
    setArr(rand(size))
    setResult(null)
  }

  async function run() {
    if (running) return
    setRunning(true)
    setResult(null)

    const fn = ALGS[alg]
    const steps = fn(arr)
    const copy = arr.slice()
    const t0 = Date.now()

    for (let s = 0; s < steps.length; s++) {
      const it = steps[s]
      if (it.type === 'compare') {
        await sleep(speed)
      } else if (it.type === 'swap') {
        const { i, j } = it
        const tmp = copy[i]
        copy[i] = copy[j]
        copy[j] = tmp
        setArr(copy.slice())
        await sleep(speed)
      } else if (it.type === 'overwrite') {
        copy[it.idx] = it.val
        setArr(copy.slice())
        await sleep(speed)
      }
    }

    const duration = Date.now() - t0
    logRun({ algorithm: alg, size, duration, timestamp: new Date().toISOString() })
    setResult(`Algorithm: ${alg} | Size: ${size} | Time: ${duration} ms`)
    setRunning(false)
  }

  function sleep(ms) {
    return new Promise((r) => setTimeout(r, ms))
  }

  return (
    <div className="p-4 max-w-6xl mx-auto">
      {/* Controls */}
      <div className="flex flex-wrap gap-4 items-center justify-between mb-4">
        <div className="flex gap-4 items-center flex-wrap">
          <label className="text-white font-semibold">Algorithm:</label>
          <select
            value={alg}
            onChange={(e) => setAlg(e.target.value)}
            className="p-2 rounded bg-gray-700 text-white"
```

SortingVisualizer.jsx

frontend > src > components > SortingVisualizer.jsx > SortingVisualizer > run

```jsx
export default function SortingVisualizer() {
          >
            {Object.keys(ALGS).map((k) => (
              <option key={k} value={k}>
                {k}
              </option>
            ))}
          </select>

          <label className="text-white font-semibold">Array Size:</label>
          <input
            type="range"
            min="10"
            max="150"
            value={size}
            onChange={(e) => setSize(Number(e.target.value))}
            className="w-32"
          />
          <span className="text-white">{size}</span>

          <label className="text-white font-semibold">Speed (ms):</label>
          <input
            type="range"
            min="5"
            max="200"
            value={speed}
            onChange={(e) => setSpeed(Number(e.target.value))}
            className="w-32"
          />
          <span className="text-white">{speed} ms</span>
        </div>

        <div className="flex gap-2">
          <button
            className="px-3 py-1 bg-red-600 rounded hover:bg-red-700 disabled:opacity-50"
            onClick={reset}
            disabled={running}
          >
            Randomize
          </button>
          <button
            className="px-3 py-1 bg-green-600 rounded hover:bg-green-700 disabled:opacity-50"
            onClick={run}
            disabled={running}
          >
            Start
          </button>
        </div>
      </div>

      {/* Bars */}
      <div
        style={{ height: Math.min(520, Math.max(260, size * 3.5)) }}
        className="relative bg-[rgba(255,255,255,0.02)] rounded-md p-3"
      >
        <div className="flex items-end gap-1 h-full">
          {arr.map((v, i) => (
            <motion.div
              key={i}
              className="bar bg-gradient-to-b from-cyan-400/80 to-sky-600/80 rounded"
              style={{ height: `${v}px`, width: `${Math.max(4, Math.floor(1000 / size))}px` }}
              layout
              transition={{ type: 'spring', stiffness: 300, damping: 30 }}
            />
          ))}
        </div>
      </div>

      {/* Result */}
      {result && (
        <div className="mt-4 p-2 bg-green-800 text-white rounded">
          {result}
        </div>
      )}
    </div>
  )
}
```

bubble.js

frontend > src > algos > bubble.js > ...

```js
export function bubble(arr) {
  const steps = []
  const a = arr.slice()
  const n = a.length

  for (let i = 0; i < n - 1; i++) {
    for (let j = 0; j < n - i - 1; j++) {
      // Step: compare
      steps.push({ type: 'compare', i: j, j: j + 1 })

      if (a[j] > a[j + 1]) {
        // Step: swap
        [a[j], a[j + 1]] = [a[j + 1], a[j]]
        steps.push({ type: 'swap', i: j, j: j + 1 })
      }
    }
  }

  return steps
}
```

insertion.js

frontend > src > algos > insertion.js > insertion

```js
export function insertion(a) {
  const arr = a.slice(), steps = [];
  for (let i = 1; i < arr.length; i++) {
    let j = i;
    while (j > 0) {
      steps.push({ type: 'compare', i: j - 1, j });
      if (arr[j - 1] > arr[j]) {
        steps.push({ type: 'swap', i: j - 1, j });
        [arr[j - 1], arr[j]] = [arr[j], arr[j - 1]];
      } else break;
      j--;
    }
  }
  return steps;
}
```

insertion.js M | merge.js M

frontend > src > algos > merge.js > ...

```js
export function mergeSort(a) {
  const arr = a.slice(), steps = [];
  function merge(l, m, r) {
    const L = arr.slice(l, m + 1), R = arr.slice(m + 1, r + 1);
    let i = 0, j = 0, k = l;
    while (i < L.length && j < R.length) {
      steps.push({ type: 'compare', i: l + i, j: m + 1 + j });
      if (L[i] <= R[j]) steps.push({ type: 'overwrite', idx: k, val: L[i] }), arr[k++] = L[i++];
      else steps.push({ type: 'overwrite', idx: k, val: R[j] }), arr[k++] = R[j++];
    }
    while (i < L.length) steps.push({ type: 'overwrite', idx: k, val: L[i] }), arr[k++] = L[i++];
    while (j < R.length) steps.push({ type: 'overwrite', idx: k, val: R[j] }), arr[k++] = R[j++];
  }
  function ms(l, r) { if (l < r) { const m = (l + r) >> 1; ms(l, m); ms(m + 1, r); merge(l, m, r); } }
  ms(0, arr.length - 1);
  return steps;
}
// --- IGNORE ---
```

insertion.js M | merge.js M | selection.js

frontend > src > algos > selection.js > selection

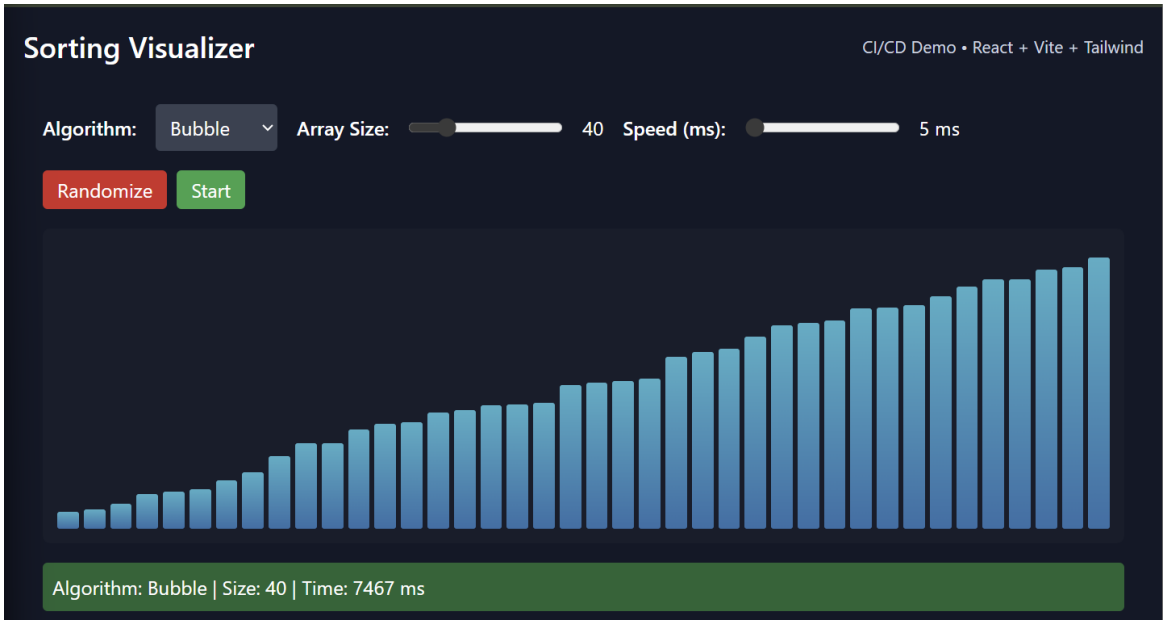
```
export function selection(a){
const arr=a.slice(); const steps=[]; const n=arr.length;
for(let i=0;i<n-1;i++){
let min=i;
for(let j=i+1;j<n;j++){
steps.push({type:'compare', i:min, j});
if(arr[j]<arr[min]) min=j;
}
if(min!==i){ steps.push({type:'swap', i, j:min}); const t=arr[i]; arr[i]=arr[min]; arr[min]=t }
}
return steps;
}
```

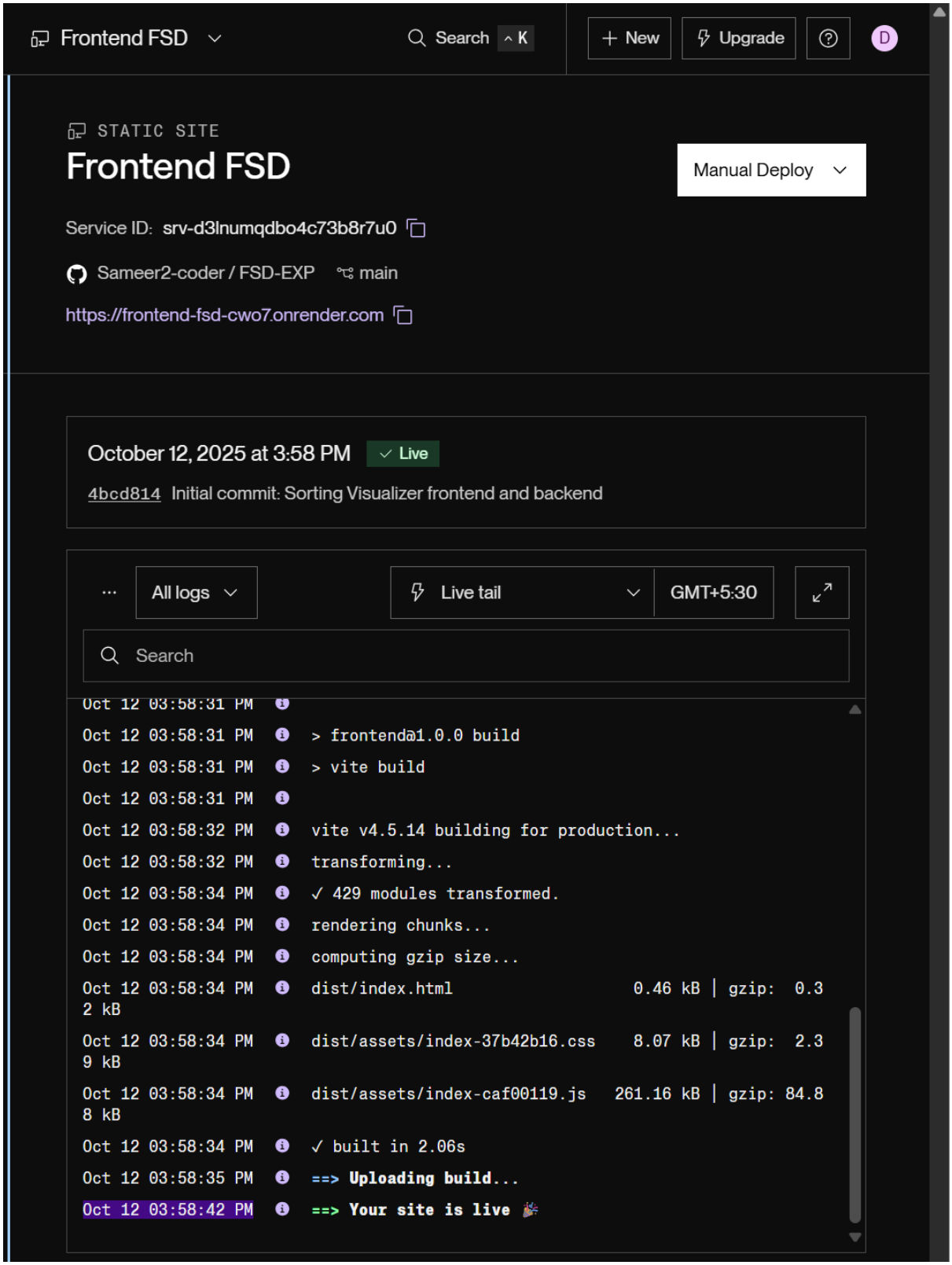
insertion.js M | merge.js M | App.jsx

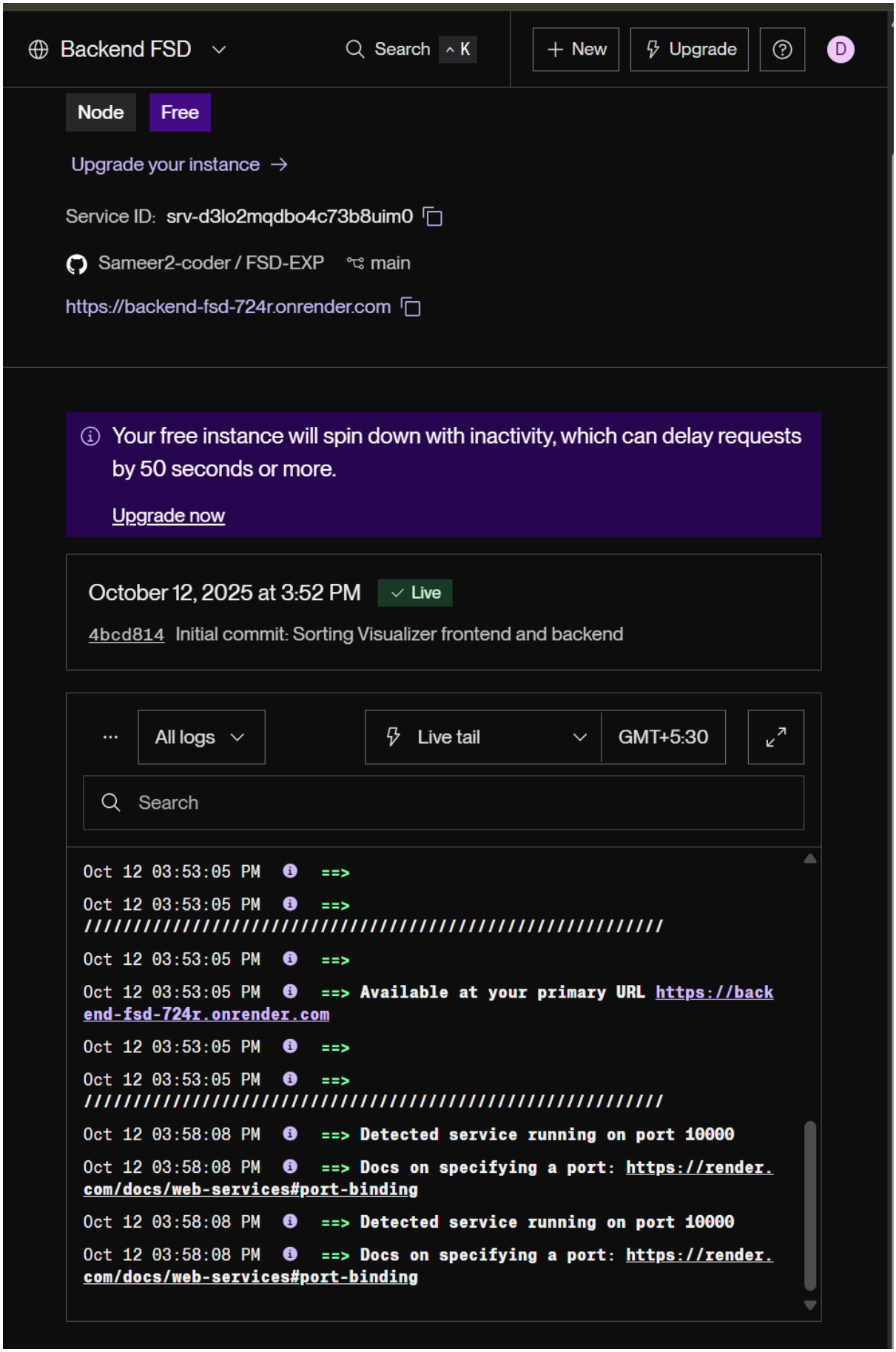
frontend > src > App.jsx > ...

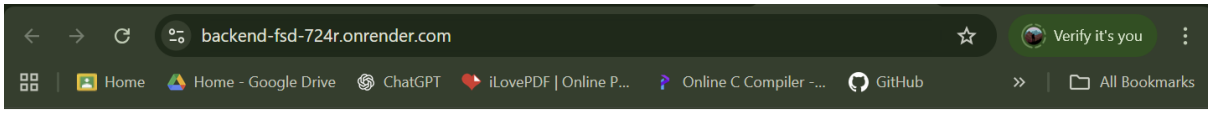
```
import React from 'react'
import Navbar from './components/Navbar'
import SortingVisualizer from './components/SortingVisualizer'

function App() {
  return (
    <div className="min-h-screen bg-gray-900 text-white p-4">
      <Navbar />
      <SortingVisualizer />
    </div>
  )
}

export default App
```

OUTPUT:




Frontend FSD
Search K
New
Upgrade
D
STATIC SITE
Frontend FSD
Manual Deploy
Service ID: srv-d3lnumqdbo4c73b8r7u0
Sameer2-coder / FSD-EXP
main
https://frontend-fsd-cwo7.onrender.com
October 12, 2025 at 3:58 PM
Live
4bcd814 Initial commit: Sorting Visualizer frontend and backend
All logs
Live tail
GMT+5:30
Search
Oct 12 03:58:31 PM
Oct 12 03:58:31 PM > frontend@1.0.0 build
Oct 12 03:58:31 PM > vite build
Oct 12 03:58:31 PM
Oct 12 03:58:32 PM vite v4.5.14 building for production...
Oct 12 03:58:32 PM transforming...
Oct 12 03:58:34 PM ✓ 429 modules transformed.
Oct 12 03:58:34 PM rendering chunks...
Oct 12 03:58:34 PM computing gzip size...
Oct 12 03:58:34 PM dist/index.html 0.46 kB | gzip: 0.3
2 kB
Oct 12 03:58:34 PM dist/assets/index-37b42b16.css 8.07 kB | gzip: 2.3
9 kB
Oct 12 03:58:34 PM dist/assets/index-caf00119.js 261.16 kB | gzip: 84.8
8 kB
Oct 12 03:58:34 PM ✓ built in 2.06s
Oct 12 03:58:35 PM ==> Uploading build...
Oct 12 03:58:42 PM ==> Your site is live


Backend FSD
Search ^ K
+ New
Upgrade
D
Node
Free
Upgrade your instance →
Service ID: srv-d3lo2mqdbo4c73b8uim0
Sameer2-coder / FSD-EXP
main
https://backend-fsd-724r.onrender.com
Your free instance will spin down with inactivity, which can delay requests by 50 seconds or more.
Upgrade now
October 12, 2025 at 3:52 PM
Live
4bcd814 Initial commit: Sorting Visualizer frontend and backend
All logs
Live tail
GMT+5:30
Search
Oct 12 03:53:05 PM ==>
Oct 12 03:53:05 PM ==>
//
Oct 12 03:53:05 PM ==>
Oct 12 03:53:05 PM ==> Available at your primary URL https://backend-fsd-724r.onrender.com
Oct 12 03:53:05 PM ==>
Oct 12 03:53:05 PM ==>
//
Oct 12 03:58:08 PM ==> Detected service running on port 10000
Oct 12 03:58:08 PM ==> Docs on specifying a port: https://render.com/docs/web-services#port-binding
Oct 12 03:58:08 PM ==> Detected service running on port 10000
Oct 12 03:58:08 PM ==> Docs on specifying a port: https://render.com/docs/web-services#port-binding



Sorting Visualizer Backend Running!

LINKS:-

https://frontend-fsd-cwo7.onrender.com/

https://github.com/Sameer2-coder/FSD-EXP