Experiment No 9

Aim : CI/CD Deployment with GitHub Actions + Render/Vercel

Code :

Render.yaml

```yaml
services:
  - type: web
    name: realtime-websocket-chat
    env: node
    plan: free
    buildCommand: npm install
    startCommand: npm start
    envVars:
      - key: NODE_ENV
        value: production
      - key: FRONTEND_URL
        value: https://fsd-practical-9-vercel-o4g1.vercel.app/
    healthCheckPath: /
```

Vercel.json

vercel.json > [ ] routes

You, 12 minutes ago | 1 author (You)

```json
{
  "version": 2,
  "builds": [
    {
      "src": "server.js",
      "use": "@vercel/node"
    },
    {
      "src": "index.html",
      "use": "@vercel/static"
    }
  ],
  "routes": [
    {
      "src": "/socket.io/(.*)",
      "dest": "/server.js"
    },
    {
      "src": "/(.*)",
      "dest": "/index.html"
    }
  ]
}
```

You, 12 minutes ago

Output : (Check it out - live app)

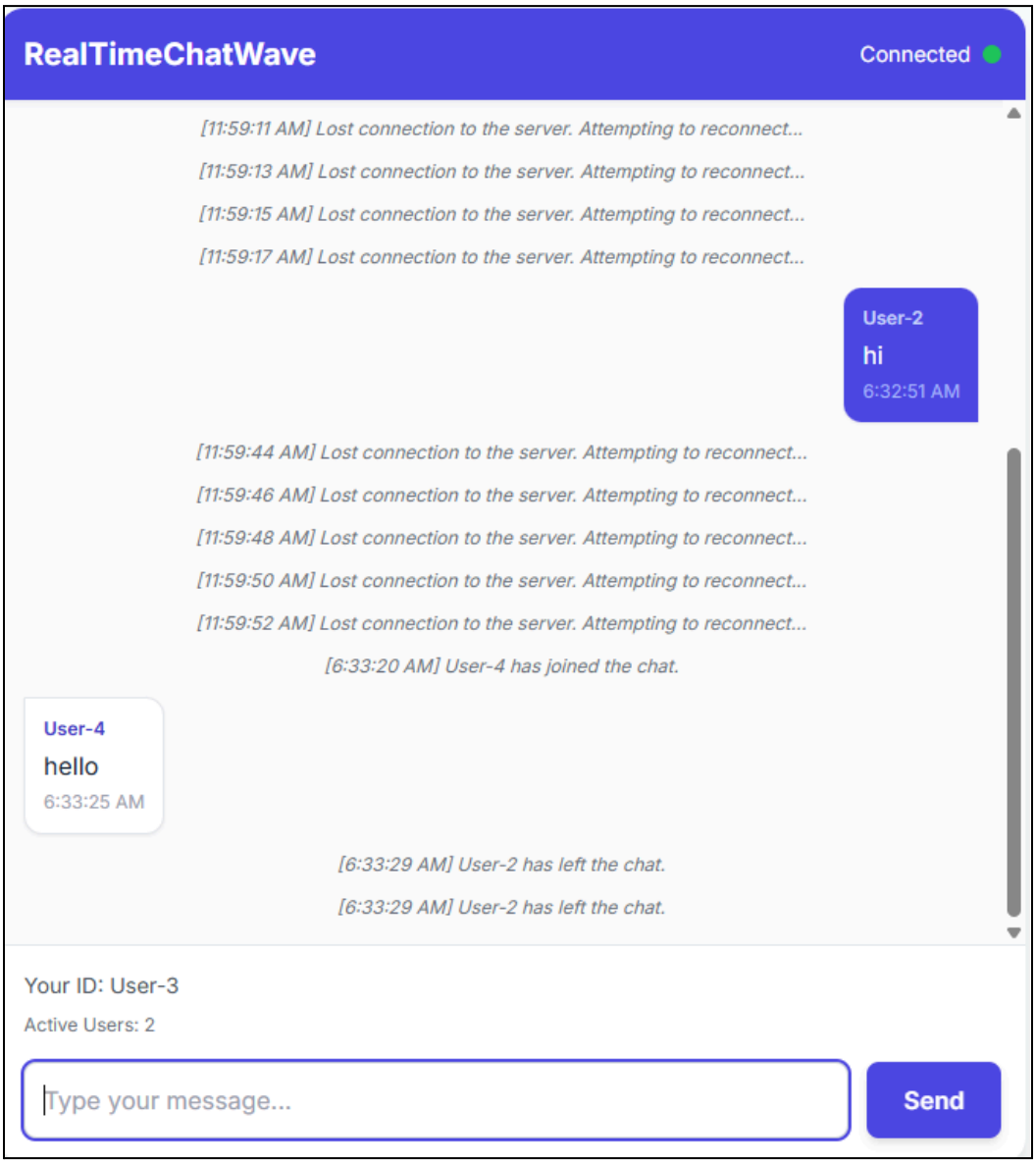

Figure 9.1 - real time chatting between two users thru websockets

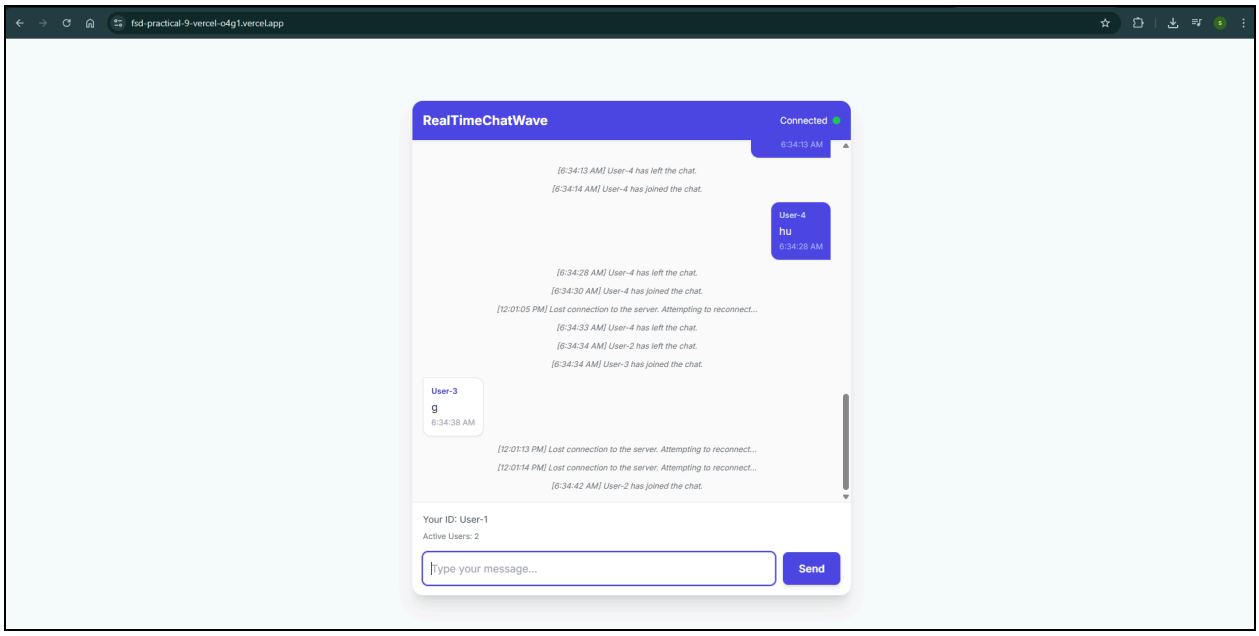

Figure 9.2 - working vercel frontend page - connecting to the render backend

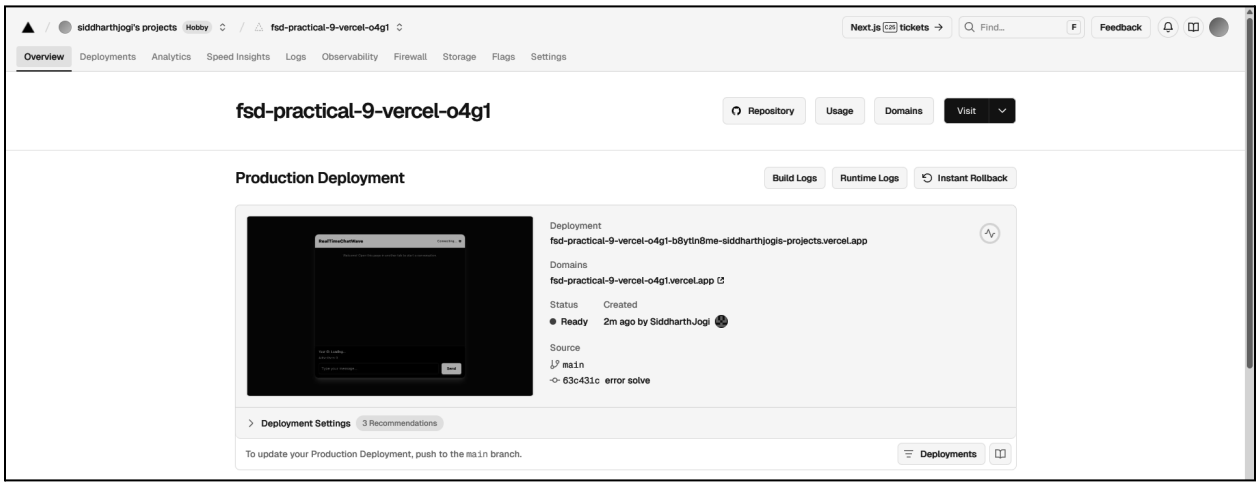

Figure 9.3 - Vercel config

Figure 9.4 - Render config

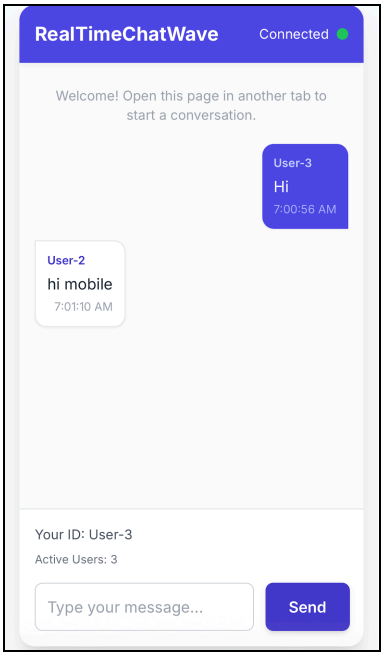

Figure 9.5 - Cross Platform Working

Conclusion - Thus we have deployed the frontend of our website to vercel and the backend of the website to render , and used it to create a persistent websocket connection on the production stage