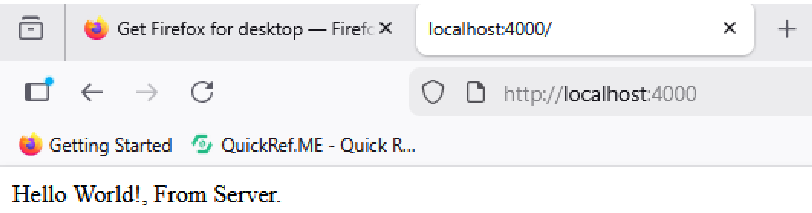
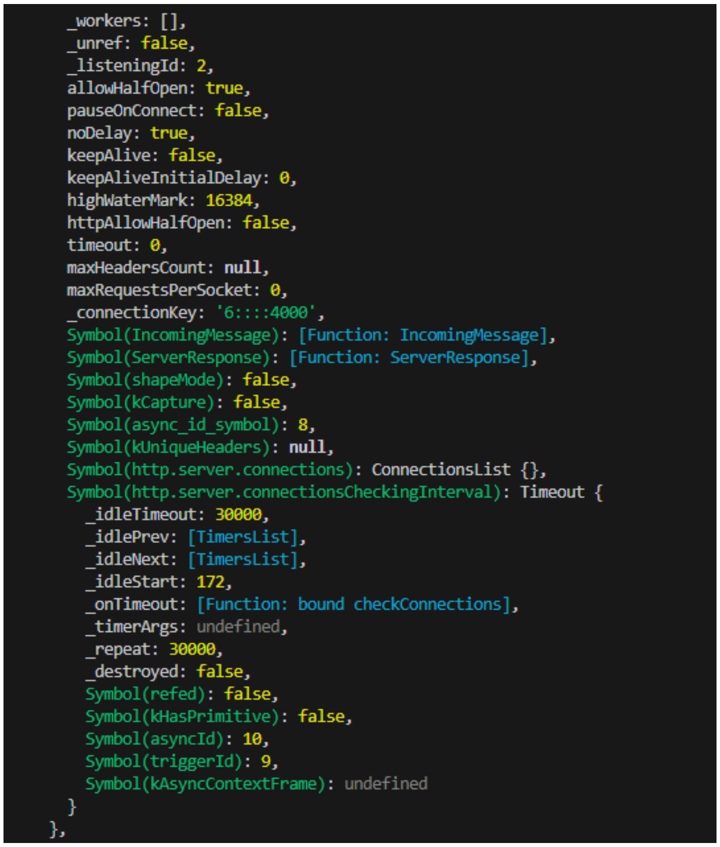
Practical No 8

Aim : Write A Program To Demonstrate The Use Of Middleware In Express.js.

```
import express from "express"
import { logger } from "./middleware/looger.js";


const port = 4000
const app = express()

// Middleware
app.use(express.json())
app.use(logger)

app.get("/", (req, res) => {
  res.send("Hello World!, From Server.")
})

app.listen(port, () => {
  console.log(`Server Started On : localhost:${port}`)
})
```



Hello World!, From Server.



```
    _workers: [],
    _unref: false,
    _listeningId: 2,
    allowHalfOpen: true,
    pauseOnConnect: false,
    noDelay: true,
    keepAlive: false,
    keepAliveInitialDelay: 0,
    highWaterMark: 16384,
    httpAllowHalfOpen: false,
    timeout: 0,
    maxHeadersCount: null,
    maxRequestsPerSocket: 0,
    _connectionKey: '6::::4000',
    Symbol(IncomingMessage): [Function: IncomingMessage],
    Symbol(ServerResponse): [Function: ServerResponse],
    Symbol(shapeMode): false,
    Symbol(kCapture): false,
    Symbol(async_id_symbol): 8,
    Symbol(kUniqueHeaders): null,
    Symbol(http.server.connections): ConnectionsList {},
    Symbol(http.server.connectionsCheckingInterval): Timeout {
      _idleTimeout: 30000,
      _idlePrev: [TimersList],
      _idleNext: [TimersList],
      _idleStart: 172,
      _onTimeout: [Function: bound checkConnections],
      _timerArgs: undefined,
      _repeat: 30000,
      _destroyed: false,
      Symbol(refed): false,
      Symbol(kHasPrimitive): false,
      Symbol(asyncId): 10,
      Symbol(triggerId): 9,
      Symbol(kAsyncContextFrame): undefined
    }
  },
```