Snigdha Ghosh(23ICS10004)

Practice 1 - Backend

Title: Middleware Implementation for Logging and Bearer Token Authentication

Objective:
To learn how to build and integrate middleware functions in an Express.js application that:
1. Logs every incoming request (method, URL, timestamp).
2. Secures certain routes using a Bearer token.

Concept Overview:
Middleware are functions that execute between the request and response in Express.js. Global middleware (like a logger) applies to all routes. Authentication middleware checks headers and decides if the request should continue.

Steps / Procedure:

Step 1: Initialize a Node.js project

```
mkdir middleware-demo cd middleware-demo npm init -y
npm install express
```

Step 2: Create server.js

```
const express = require('express'); const app = express(); const PORT = 3000;

// Logging Middleware app.use((req, res, next) => {
const timestamp = new Date().toISOString();
console.log(`[${timestamp}] ${req.method} ${req.url}`);
next(); });

// Authentication Middleware const authenticateToken = (req, res, next) => { const authHeader = req.headers['authorization']; if (!authHeader) { return res.status(401).json({ message: 'Authorization header missing or incorrect' }); }

  const token = authHeader.split(' ')[1]; if (token !== 'mysecrettoken') { return res.status(401).json({ message: 'Invalid token' }); }

  next();
};

// Public Route app.get('/public', (req, res) => {
res.send('This is a public route. No authentication required.');
});
// Protected Route

app.get('/protected', authenticateToken, (req, res) => { res.send('You have accessed a protected route with a valid Bearer token!'); });

app.listen(PORT, () => { console.log(`Server running on http://localhost:${PORT}`); });
```

Step 3: Run the Server

```
node server.js
```

Step 4: Test the Routes

■ Public Route curl

```
http://localhost:3000/public
```

■ Protected Route (without token)

```
curl http://localhost:3000/protected
```

■ Protected Route (with correct token) curl -H "Authorization: Bearer mysecrettoken" http://localhost:3000/protected

Expected Console Output: [2025-10-23T12:34:56.789Z] GET /public [2025-10-23T12:35:02.123Z] GET /protected [2025-10-23T12:35:10.456Z] GET /protected

Result:

- Logging middleware logs all incoming requests.
- Protected route is only accessible when the correct Bearer token is provided.- The system correctly returns HTTP 200 or 401 as expected.