­­­­­­­­­­­

|  |  |
| --- | --- |
| **Ex. No: 7** | **Routing Implementation using ExpressJS** |
| **28.09.2023** |

**Aim:**ToImplement the routing feature(s) using the ExpressJS.

**Algorithm:**

1. Include the required header files thread creation and sleep() function.
2. Write a function that executes as a thread when it is called. (sleep – print - return)
3. thread\_id is declared to identify the thread in the system, we call pthread\_create() function to create a thread.
4. The pthread\_join() function for threads is the equivalent of wait() for processes. A call to pthread\_join blocks the calling thread until the thread with identifier equal to the first argument terminates.

**Program:**

const express = require('express');

const bodyParser = require('body-parser');

const app = express();

const port = 3000; // You can change this to any port you prefer

app.set('view engine', 'ejs'); // Use EJS as the template engine

app.use(bodyParser.urlencoded({ extended: false }));

app.use(express.static(\_\_dirname + '/public'));

const posts = []; // Simulated database for storing blog posts

// Home page route

app.get('/', (req, res) => {

res.render('home', { posts });

});

// New post form route

app.get('/newpost', (req, res) => {

res.render('newpost');

});

// Create a new post route

app.post('/create', (req, res) => {

const { title, content } = req.body;

const newPost = { title, content };

posts.push(newPost);

res.redirect('/');

});

// JSON API endpoint to fetch all posts

app.get('/api/posts', (req, res) => {

res.json(posts);

});

// Start the server

app.listen(port, () => {

console.log(`Server is running on port ${port}`);

});

**Output:**

Github Link: <https://github.com/kavin-t28/CS3809-Web-Technologies-Lab>

1. Initial webpage when the server is started.



1. New blog creation page

A screenshot of a computer

Description automatically generated

1. When we create new blogs the title of the blog gets updated in the home page

A screenshot of a computer

Description automatically generated

1. All the blog content for that session can be assesed in the /api/posts route in json formate

A screenshot of a computer

Description automatically generated

**Result:**

Therefore, we've successfully implemented a basic routing implementation using Express JS