W3 PRACTICE

Express Basics + POST + Middleware

At the end of this practice, you can

- ✓ **Create** and run a express.js HTTP server
- ✓ **Implement** route handling using express.js
- ✓ Parse form data from POST requests with middleware.
- ✓ Apply middleware concept to logging

Get ready before this practice!

- ✓ **Read** the following documents to understand the nature of Express.js:
  https://expressjs.com/

- ✓ **Read** the following documents to know more about Express.js's built-in middleware's:
  https://expressjs.com/en/resources/middleware.html

- ✓ **Read** the following documents to understand MDN: HTTP POST:
  https://developer.mozilla.org/en-US/docs/Web/HTTP/Reference/Methods/POST

- ✓ **Read** the following documents to array filter:
  https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Array/filter

How to submit this practice?

- ✓ Once finished, push your **code to GITHUB**
- ✓ Join the **URL of your GITHUB** repository on LMS

*Student's name: **Someth Phay***

IDTB100019

SE G2 Gen10 Year2 Term3

Lect. Kim Ang Kheang

S2 - PRACTICE - ExpressJS 1

GitHub Repo:

https://github.com/PhaySometh/Y2_Term3_W3_S2-PRACTICE-ExpressJS_1.git

EXERCISE 1 – *Refactoring*

Goals

- ✓ Take advantage of Express.js framework's flexibility and minimalism
- ✓ Refactor code from node.js's built-in HTTP Module

Refactor the source code of EXERCISE 2 & 3 in Week 2 to Express.js

Q1 – What challenges did you face when using the native http module that Express.js helped you solve?

➔ **Answer:**

- **Manual routing:** You must check URL and method manually.
- **No body parsing:** You handle request data as raw streams.
- **No middleware:** Must manually implement things like logging or auth.
- **Verbose responses:** Setting headers and sending JSON is tedious.

Q2 – How does Express simplify route handling compared to the native HTTP server?

➔ With native http, routing is done with if checks on req.url and req.method.

With Express, you just use:

```
app.get('/route', handler);
```

It's cleaner, readable, and easier to manage.

Q3 – What does middleware mean in Express, and how would you replicate similar behavior using the native module?

➔ **Express middleware**: Functions run before route handlers using app.use().

Native alternative: You chain functions manually and call next() yourself.

```
function logger(req, res, next) {
  console.log(req.url);
  next();
}
```

Express makes middleware easy and standardized.

EXERCISE 2 – API for Course Records

*For this exercise you will start with a **START CODE (EX-2)***

Goals

- ✓ Understand Route Parameters (:param)
- ✓ Work with Query Parameters (?key=value)
- ✓ Implement Conditional Logic for Filtering
- ✓ Build Real-World Web API Behavior
- ✓ Practice Defensive Programming

Context

You are building a backend API for a university's course catalog. Each course has the following fields

```
{
  "id": "CSE101",
  "title": "Introduction to Computer Science",
  "department": "CS",
  "level": "undergraduate",
  "credits":  3,
  "instructor": "Dr. KimAng",
  "semester": "fall"
}
```

Q1 - Create a route

```
GET /departments/:dept/courses
```

EXAMPLE

```
/departments/CSE/courses
```

Q2 - Accept query parameters to filter the result:

- `level` → e.g., `undergraduate, graduate`
- `minCredits` → integer
- `maxCredits` → integer
- `semester` → `fall, spring,` etc.
- `instructor` → partial match

EXAMPLE

```
/departments/CSE/courses?level=undergraduate&minCredits=2&semester=fall
```

Q3 - **Return** a JSON array of courses that match:

- The :dept from the route parameter
- The filter criteria from query parameters

Q4 – Handle Edge Cases

- **Invalid credit ranges** (minCredits > maxCredits)
- No **matching courses**
- **Missing** or **unsupported** query parameters (ignore them silently)

EXAMPLES

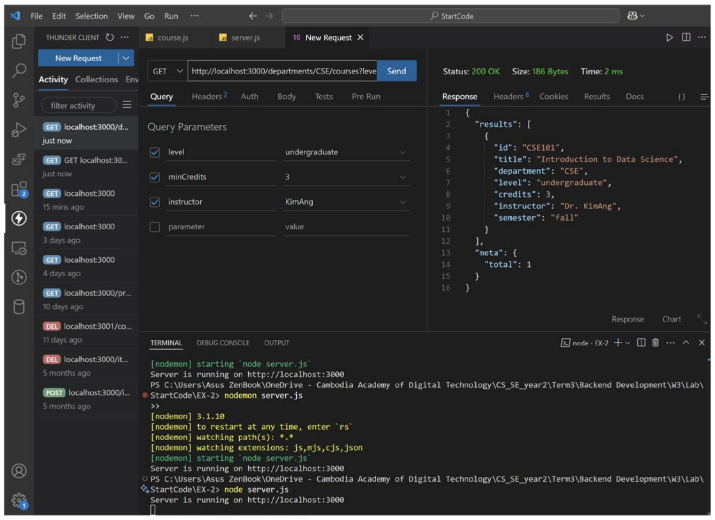
REQUEST
`/departments/CSE/courses?level=undergraduate&minCredits=3&instructor=KimAng`
RESPONSE

```
{
  "results": [
    {
      "id": "CSE101",
      "title": "Introduction to Data Science",
      "department": "CSE",
      "level": "undergraduate",
      "credits":  3,
      "instructor": "Dr. KimAng",
      "semester": "fall"
    }
  ],
  "meta": {
    "total": 1
  }
}
```

EDGE CASES

- `http://localhost:3000/departments/CSE/courses`
- `http://localhost:3000/departments/CSE/courses?level=undergraduate`
- `http://localhost:3000/departments/CSE/courses?minCredits=4`
- `http://localhost:3000/departments/CSE/courses?instructor=smith&semester=fall`

My response:


File Edit Selection View Go Run
StartCode
THUNDER CLIENT
course.js
server.js
New Request
Activity Collections Env
filter activity
GET localhost:3000/d... just now
GET GET localhost:30... just now
GET localhost:3000 15 mins ago
GET localhost:3000 3 days ago
GET localhost:3000 4 days ago
GET localhost:3000/pr... 10 days ago
DEL localhost:3001/co... 11 days ago
DEL localhost:3000/it... 5 months ago
POST localhost:3000/i... 5 months ago
Send
Query Headers 2 Auth Body Tests Pre Run
Response Headers 6 Cookies Results Docs
Response Chart
TERMINAL DEBUG CONSOLE OUTPUT
node - EX-2

GET http://localhost:3000/departments/CSE/courses?leve

Query Parameters

✓	level	undergraduate
✓	minCredits	3
✓	instructor	KimAng
	parameter	value

Status: 200 OK Size: 186 Bytes Time: 2 ms

```
{
  "results": [
    {
      "id": "CSE101",
      "title": "Introduction to Data Science",
      "department": "CSE",
      "level": "undergraduate",
      "credits": 3,
      "instructor": "Dr. KimAng",
      "semester": "fall"
    }
  ],
  "meta": {
    "total": 1
  }
}
```

```
[nodemon] starting `node server.js`
Server is running on http://localhost:3000
PS C:\Users\Asus ZenBook\OneDrive - Cambodia Academy of Digital Technology\CS_SE_year2\Term3\Backend Development\W3\Lab\
StartCode\EX-2> nodemon server.js
>>
[nodemon] 3.1.10
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): *.*
[nodemon] watching extensions: js,mjs,cjs,json
[nodemon] starting `node server.js`
Server is running on http://localhost:3000
PS C:\Users\Asus ZenBook\OneDrive - Cambodia Academy of Digital Technology\CS_SE_year2\Term3\Backend Development\W3\Lab\
StartCode\EX-2> node server.js
Server is running on http://localhost:3000
```

EXERCISE 3 – *Enhance an API with Middleware*

Goal

Your goal is to modularize and secure your course filtering API using **Express middleware**. Middleware helps keep your code clean, reusable, and extensible.

Q1 - Create a middleware function that logs the following for every request:

- HTTP method (GET, POST, etc.)
- Request path (e.g., `/departments/CSE/courses`)
- Query parameters
- Timestamp in ISO format

✓ **Apply this middleware globally** so it logs **all incoming requests** to the server.

Q2 - Create a route-specific middleware to **validate query parameters**:

- `If minCredits or maxCredits are present, ensure they are valid integers.`
- `If minCredits > maxCredits, return 400 Bad Request with an error message.`

✓ **Apply this middleware only** to the `/departments/:dept/courses route.`

Q3 – (*Bonus*) Token-Based Authentication Middleware

Simulate basic API security:

- Require a token query parameter (e.g., ?token=xyz123)
- If the token is missing or incorrect, respond with 401 Unauthorized.

✓ This middleware can be applied **either globally or to specific routes**.

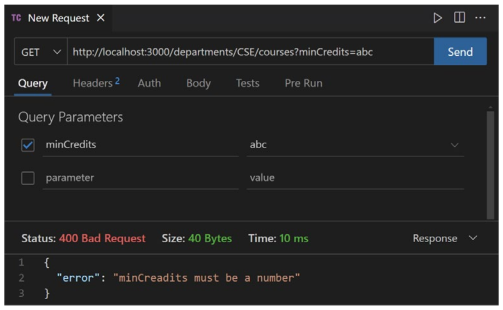
Deliverables

- `logger.js` – contains your logging middleware.
- `validateQuery.js` – contains your validation middleware.
- `auth.js` (optional) – contains your token authentication middleware.
- `server.js` – where you apply middleware and define the course filtering route.

Test cases

```
GET /departments/CSE/courses?minCredits=abc
```

⟶ should return `400 Bad Request`



New Request

GET http://localhost:3000/departments/CSE/courses?minCredits=abc Send

Query Parameters

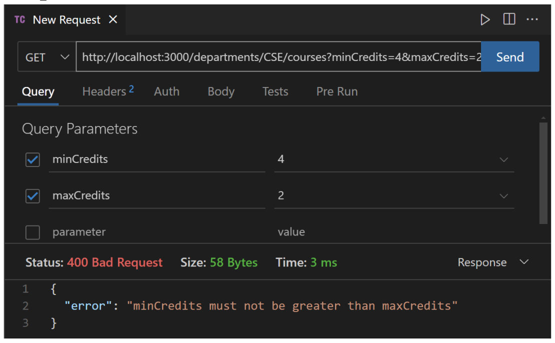
✓	minCredits	abc
	parameter	value

Status: 400 Bad Request Size: 40 Bytes Time: 10 ms

```
{
  "error": "minCreadits must be a number"
}
```

```
GET /departments/CSE/courses?minCredits=4&maxCredits=2
```

⟶ should return `400 Bad Request`



New Request

GET http://localhost:3000/departments/CSE/courses?minCredits=4&maxCredits=2 Send

Query Parameters

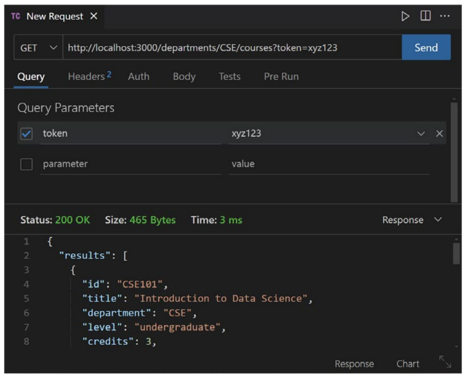
✓	minCredits	4
✓	maxCredits	2
	parameter	value

Status: 400 Bad Request Size: 58 Bytes Time: 3 ms

```
{
  "error": "minCredits must not be greater than maxCredits"
}
```

```
GET /departments/CSE/courses?token=xyz123
```

⟶ should succeed if token middleware is active



New Request

GET http://localhost:3000/departments/CSE/courses?token=xyz123 Send

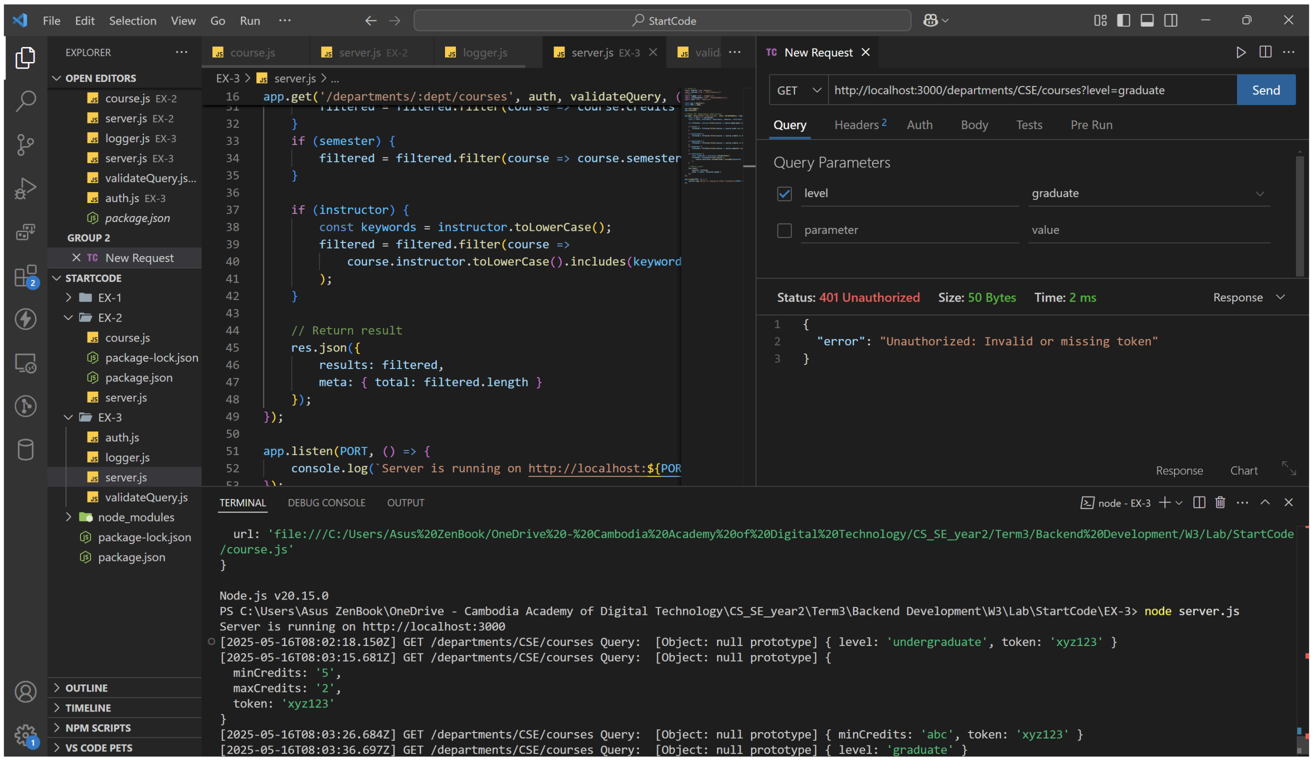
Query Parameters

✓	token	xyz123
	parameter	value

Status: 200 OK Size: 465 Bytes Time: 3 ms

```
{
  "results": [
    {
      "id": "CSE101",
      "title": "Introduction to Data Science",
      "department": "CSE",
      "level": "undergraduate",
      "credits": 3,
```

My response:



```
app.get('/departments/:dept/courses', auth, validateQuery, (
    }
    if (semester) {
        filtered = filtered.filter(course => course.semester
    }

    if (instructor) {
        const keywords = instructor.toLowerCase();
        filtered = filtered.filter(course =>
            course.instructor.toLowerCase().includes(keyword
        );
    }

    // Return result
    res.json({
        results: filtered,
        meta: { total: filtered.length }
    });
});

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

GET http://localhost:3000/departments/CSE/courses?level=graduate

Query Parameters

✓ level	graduate
parameter	value

Status: 401 Unauthorized Size: 50 Bytes Time: 2 ms

```
{
  "error": "Unauthorized: Invalid or missing token"
}
```

```
  url: 'file:///C:/Users/Asus%20ZenBook/OneDrive%20-%20Cambodia%20Academy%20of%20Digital%20Technology/CS_SE_year2/Term3/Backend%20Development/W3/Lab/StartCode
/course.js'
}

Node.js v20.15.0
PS C:\Users\Asus ZenBook\OneDrive - Cambodia Academy of Digital Technology\CS_SE_year2\Term3\Backend Development\W3\Lab\StartCode\EX-3> node server.js
Server is running on http://localhost:3000
[2025-05-16T08:02:18.150Z] GET /departments/CSE/courses Query:  [Object: null prototype] { level: 'undergraduate', token: 'xyz123' }
[2025-05-16T08:03:15.681Z] GET /departments/CSE/courses Query:  [Object: null prototype] {
  minCredits: '5',
  maxCredits: '2',
  token: 'xyz123'
}
[2025-05-16T08:03:26.684Z] GET /departments/CSE/courses Query:  [Object: null prototype] { minCredits: 'abc', token: 'xyz123' }
[2025-05-16T08:03:36.697Z] GET /departments/CSE/courses Query:  [Object: null prototype] { level: 'graduate' }
```