Ministry of Higher Education and Scientific Research - University of Mila
Institute of Mathematics and Computer Science - Department of Computer Science
Master 1 STIC – Advanced Web Development - 2025/2026

Practical Works 5 - Introduction to Node.js & Express Routing

Objective: Initialize your backend environment, understand how Express handles HTTP requests, and use Postman to test basic CRUD operations using temporary in-memory data.

Part 1: The First Server

Do not build this inside your React folder! Create a new folder named **dz-shop-backend**.

Initialize the environment:

Open your terminal in the new folder and run:

```
npm init -y

npm install express cors dotenv

npm install --save-dev nodemon
```

(Update your **package.json** to include **"dev": "nodemon server.js"**)

The Guided Setup (server.js):

Create server.js. We will set up the Express instance and a basic health check route.

```
const express = require('express');
const cors = require('cors');

const app = express();

// Middleware to read JSON bodies
app.use(express.json());
app.use(cors());

// A simple GET route to test our server
app.get('/health', (req, res) => {
  res.status(200).json({ message: 'DZ-Shop API is running!' });
});

app.listen(3000, () => {
  console.log('Server is running on http://localhost:3000');
});
```

Run **npm run dev** and test **http://localhost:3000/health** in your browser.

Part 2: Your Turn - Building the Product API

Instead of a database, we will use a temporary array for today.
Add this array to your **server.js** above your routes:

```
let products = [
  { id: 1, title: "Laptop", price: 999.99 },
  { id: 2, title: "Mouse", price: 25.00 }
];
```

Task A: The GET Route

Create a route to fetch all products.

- Method: GET
- URL: /api/products
- Logic: Use res.status(200).json(...) to send the products array.
- Test: Open Postman, make a GET request to that URL, and verify the data arrives.

Task B: The POST Route (Guided Challenge)

Now, write a route to add a new product to the array.

- **Method:** POST
- **URL:** /api/products
- **Skeleton Code:**

```
app.post('/api/products', (req, res) => {
  // 1. Extract the title and price from req.body
  const { title, price } = req.body;

  // 2. Write an IF statement: If title or price is missing,
  // return a 400 status with an error message.

  // 3. Create a new product object. Give it an ID (e.g., products.length + 1)

  // 4. Push the new product into the 'products' array.

  // 5. Send a 201 status code with the newly created product.
});
```

- **Test:** Open **Postman**, select POST, go to Body -> raw -> JSON, send a new product, and check if it gets added!