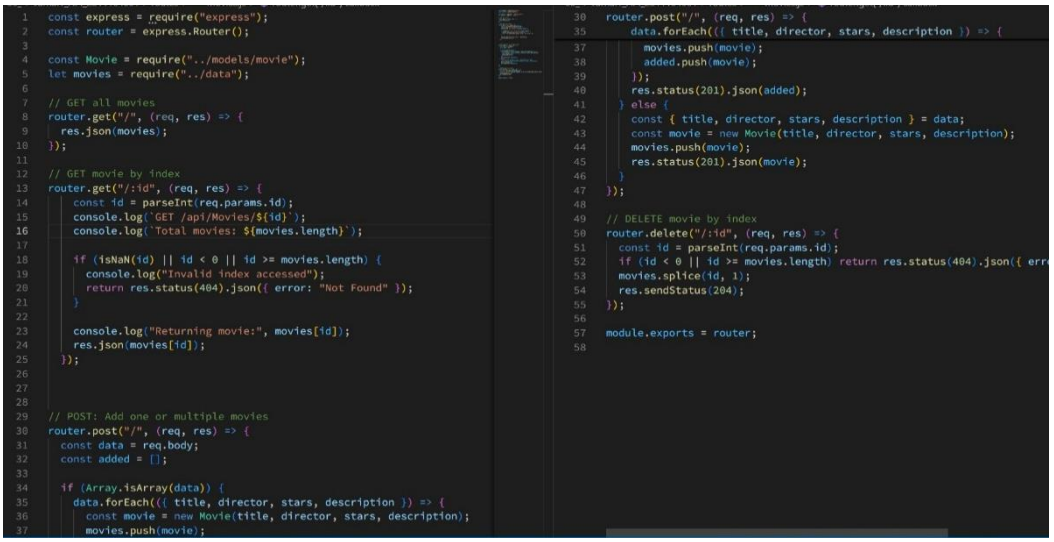
1. models/movie.js

```
class Mahasiswa {
  constructor(name, nim, course, year) {
    this.name = name;
    this.nim = nim;
    this.course = course;
    this.year = year;
  }
}

module.exports = Mahasiswa;
```

Penjelasan :
Kode tersebut mendefinisikan class `Mahasiswa` yang merepresentasikan objek mahasiswa dengan properti `name`, `nim`, `course`, dan `year`. Properti tersebut diatur melalui constructor saat objek dibuat. Class ini kemudian diekspor menggunakan `module.exports` agar bisa digunakan di file lain dalam proyek Node.js.

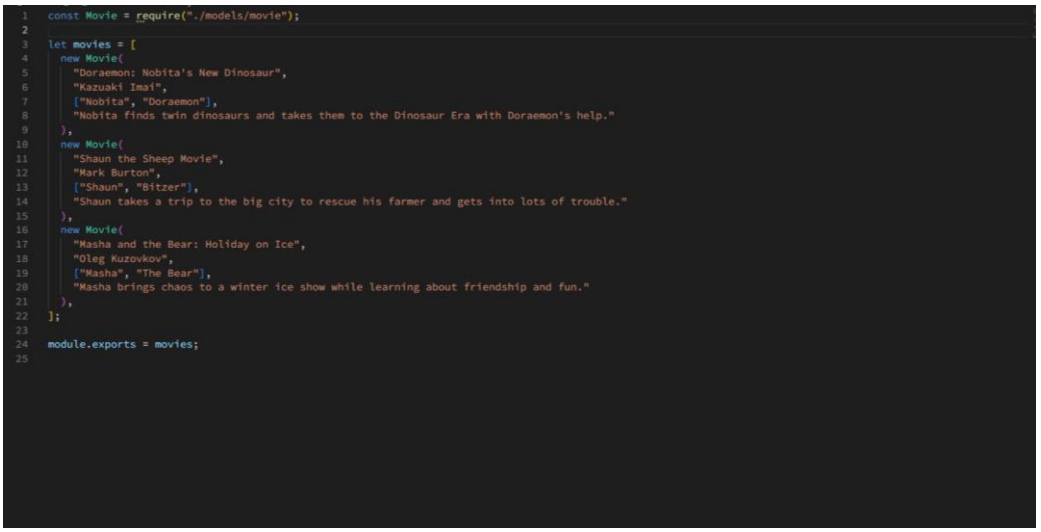
2. routes/movies.js



```
const express = require("express");
const router = express.Router();

const Movie = require("../models/movie");
let movies = require("../data");

// GET all movies
router.get("/", (req, res) => {
  res.json(movies);
});

// GET movie by index
router.get("/:id", (req, res) => {
    const id = parseInt(req.params.id);
    console.log(`GET /api/Movies/${id}`);
    console.log(`Total movies: ${movies.length}`);

    if (isNaN(id) || id < 0 || id >= movies.length) {
      console.log("Invalid index accessed");
      return res.status(404).json({ error: "Not Found" });
    }

    console.log("Returning movie:", movies[id]);
    res.json(movies[id]);
  });


// POST: Add one or multiple movies
router.post("/", (req, res) => {
  const data = req.body;
  const added = [];

  if (Array.isArray(data)) {
    data.forEach(({ title, director, stars, description }) => {
      const movie = new Movie(title, director, stars, description);
      movies.push(movie);
      added.push(movie);
    });
    res.status(201).json(added);
  } else {
    const { title, director, stars, description } = data;
    const movie = new Movie(title, director, stars, description);
    movies.push(movie);
    res.status(201).json(movie);
  }
});

// DELETE movie by index
router.delete("/:id", (req, res) => {
  const id = parseInt(req.params.id);
  if (id < 0 || id >= movies.length) return res.status(404).json({ err
  movies.splice(id, 1);
  res.sendStatus(204);
});

module.exports = router;
```

Penjelasan :
Kode tersebut merupakan router Express yang digunakan untuk mengelola data film dalam aplikasi Node.js. Data film diambil dari file `data.js` dan dibentuk menggunakan class `Movie`. Router ini menyediakan beberapa endpoint, yaitu `GET /` untuk menampilkan semua film, `GET /:id` untuk menampilkan film berdasarkan indeks, `POST /` untuk menambahkan satu atau beberapa film baru, dan `DELETE /:id` untuk menghapus film berdasarkan indeks. Semua data diproses dalam format JSON dan router ini diekspor agar dapat digunakan di file utama server.

3. data.js



```
const Movie = require("./models/movie");

let movies = [
  new Movie(
    "Doraemon: Nobita's New Dinosaur",
    "Kazuaki Imai",
    ["Nobita", "Doraemon"],
    "Nobita finds twin dinosaurs and takes them to the Dinosaur Era with Doraemon's help."
  ),
  new Movie(
    "Shaun the Sheep Movie",
    "Mark Burton",
    ["Shaun", "Bitzer"],
    "Shaun takes a trip to the big city to rescue his farmer and gets into lots of trouble."
  ),
  new Movie(
    "Masha and the Bear: Holiday on Ice",
    "Oleg Kuzovkov",
    ["Masha", "The Bear"],
    "Masha brings chaos to a winter ice show while learning about friendship and fun."
  ),
];

module.exports = movies;

```

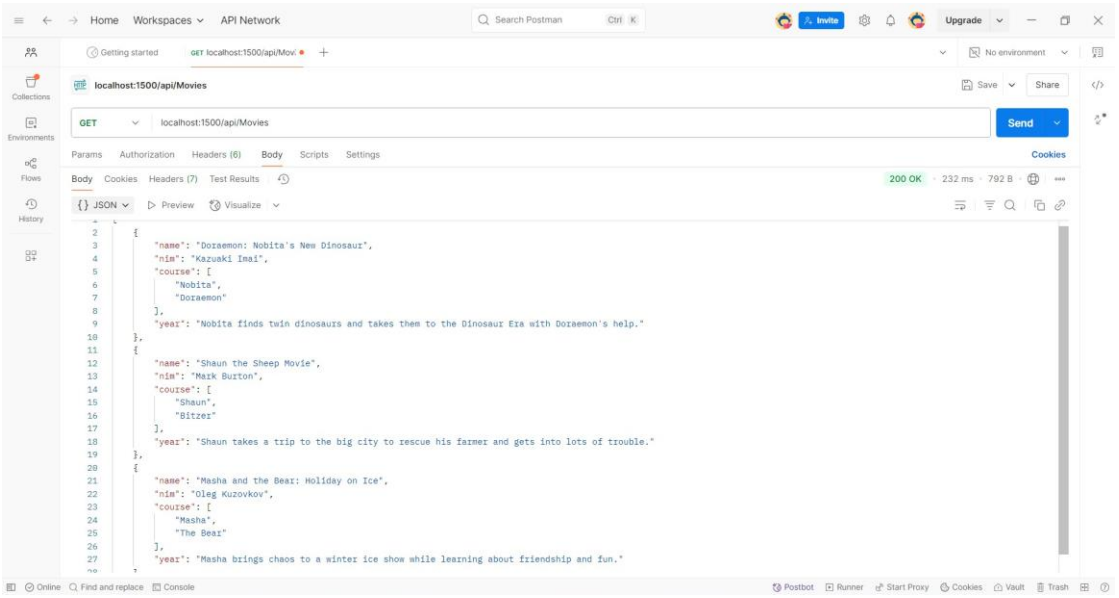
Penjelasan :
Kode tersebut membuat daftar tiga film dalam bentuk array menggunakan class `Movie`, lengkap dengan judul, sutradara, karakter, dan deskripsi. Data film ini disimpan dalam variabel `movies` dan diekspor agar bisa digunakan di file lain dalam aplikasi Node.js.

4. index.js

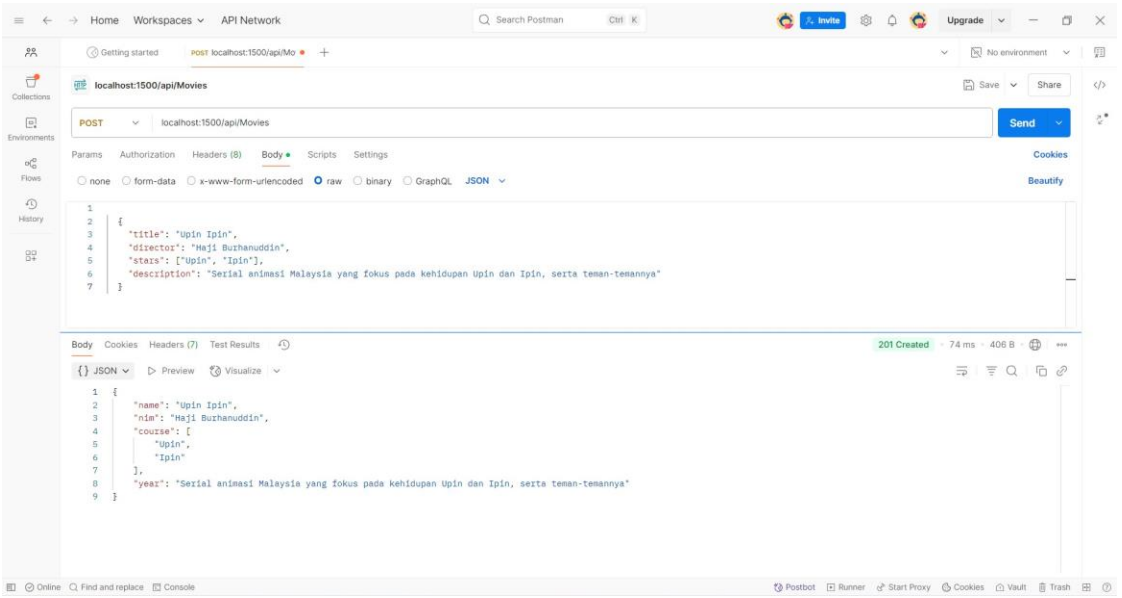
```
const express = require("express");
const app = express();
const movieRoutes = require("./routes/movies");

app.use(express.json());
app.use("/api/Movies", movieRoutes);

const PORT = 1500;
app.listen(PORT, () => {
  console.log(`Server running at http://localhost:${PORT}/api/Movies`);
});

```

Penjelasan :
Kode tersebut membuat server Express dan menghubungkan rute film dari file `routes/movies`. Middleware `express.json()` digunakan untuk memproses data JSON dari request. Server berjalan pada port 1500 dan melayani endpoint utama di `/api/Movies`.

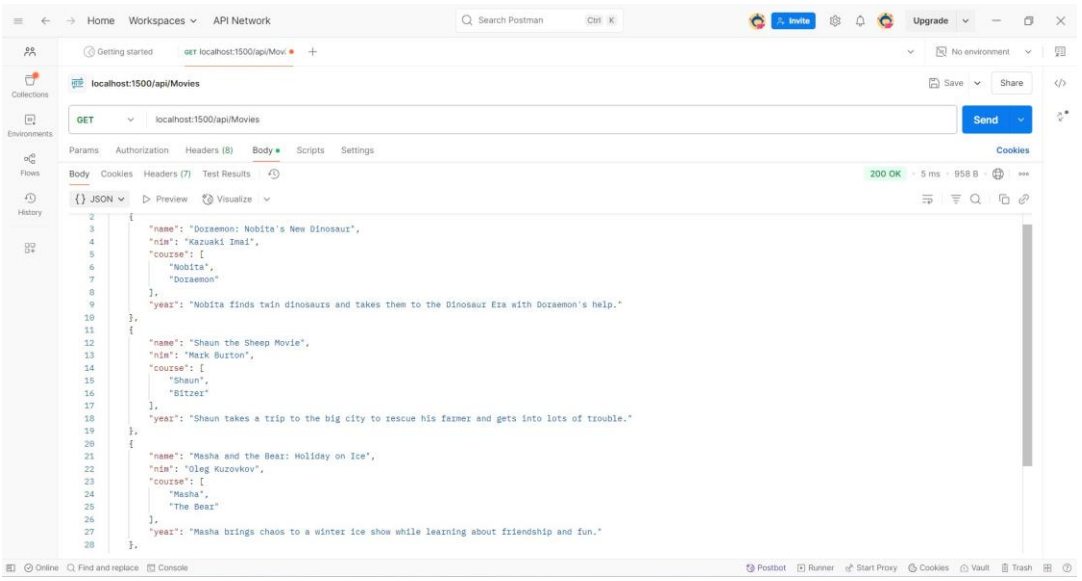
5. GET localhost:1500/api/Movies, untuk menampilkan 3 list Movies

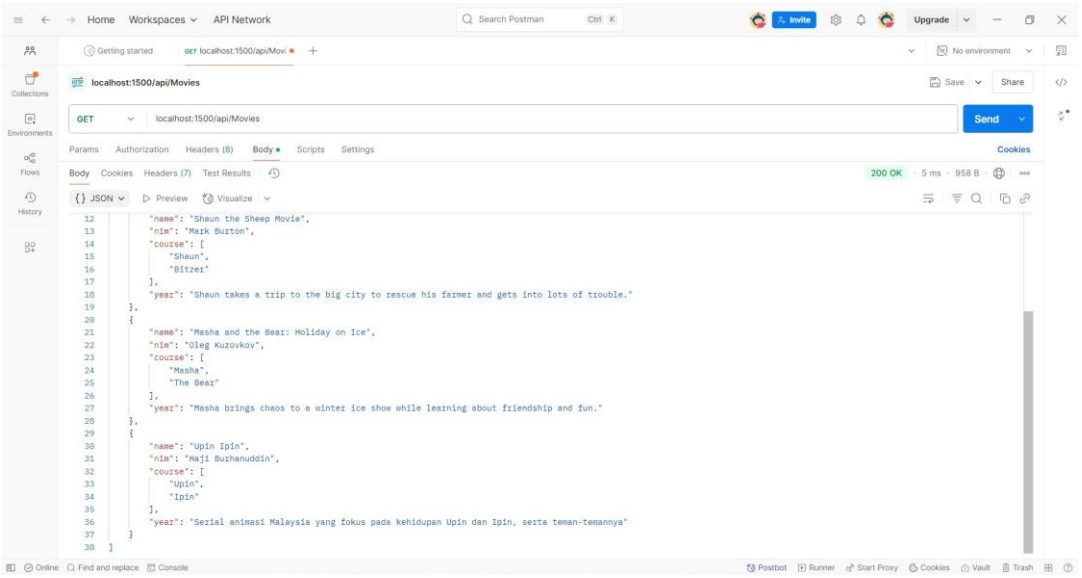


6. POST localhost:1500/api/Movies, untuk menambahkan list upin ipin

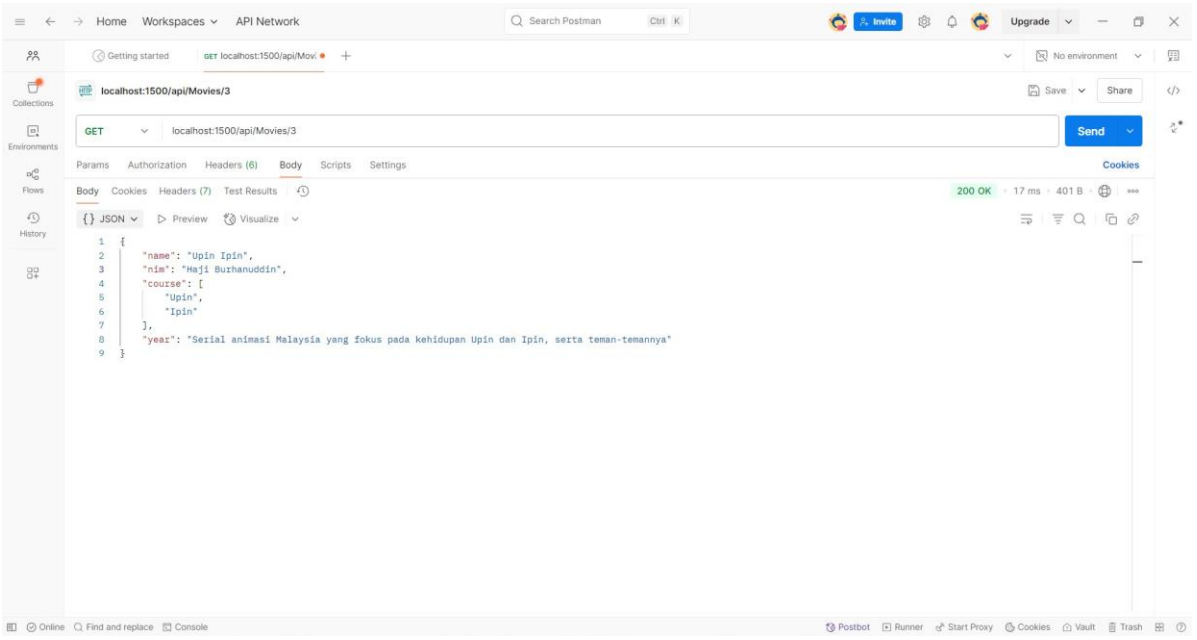


7. GET localhost:1500/api/Movies, menampilan list dan ada tambahan film upin ipin

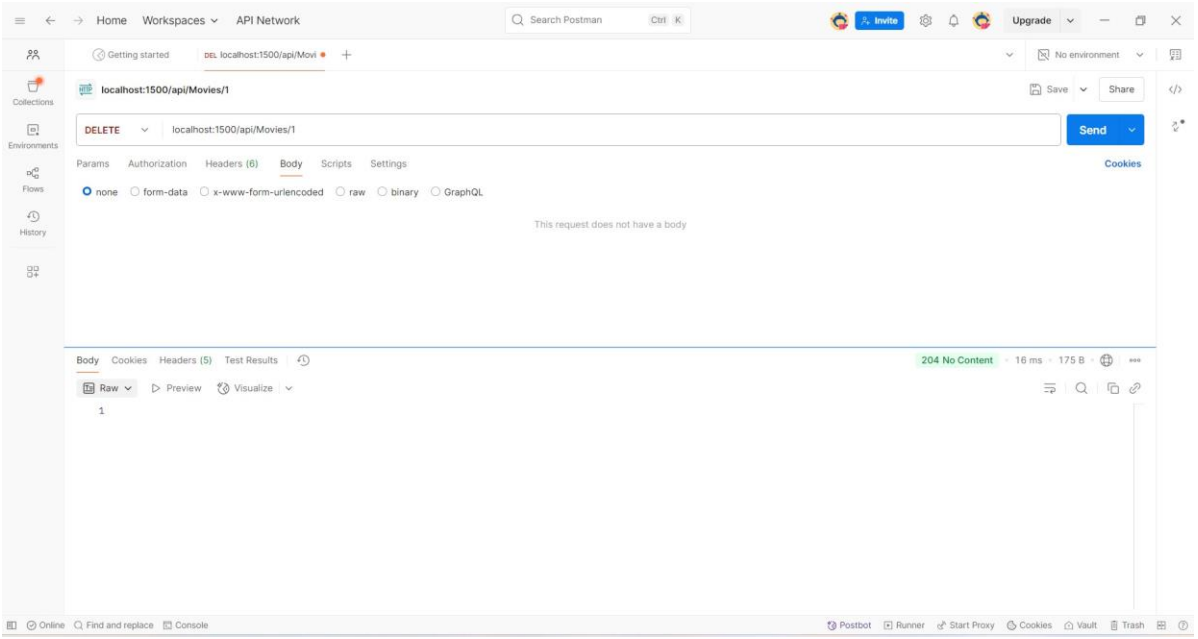




8. POST localhost:1500/api/Movies/3, untuk memanggil list yang baru ditambahkan



9. DELETE localhost:1500/api/Movies/1, untuk menghapus list dengan index 1



10. GET localhost:1500/api/Movies, memanggil list dan index 1 sudah tidak ada