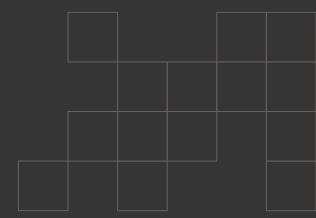
# Backend Dasar dengan Node.js + Express Bagian 3

Mengimplementasikan operasi CRUD (Create, Read, Update, Delete) untuk data Catatan



### Tujuan Pembelajaran

- Dasar JavaScript untuk backend
- Cara kerja Express dan routing
- Struktur proyek backend modern
- Membaca dan memodifikasi controller & router
- Validasi dan migrasi database

### Struktur Folder

```
catatan-api/
    config/
    └─ db.js
    controller/
      - catatanController.js
  - middlewares/
    - validate.js
    migrations/
    node modules/
    routes/
    — catatan.js
   validators/
       catatanValidator.js
        paramValidator.js
       queryValidator.js
    .env
    knexfile.js
    package-lock.json
    package.json
    server.js
```

### controller/catatanController.js

### Validasi → express-validator



express-validator adalah **middleware** untuk **validasi dan sanitasi** input di aplikasi Express.js. Library ini dibangun di atas validator.js, dan sangat membantu untuk:

- Memastikan data yang dikirim user valid, misalnya: email valid, password tidak kosong, usia harus angka, dsb.
- Membersihkan input dari karakter-karakter tak aman (sanitasi), contohnya menghindari XSS.

Perintah untuk menambahkan express-validator:

npm install express-validator

### Kapan Harus Pakai express-validator?

### Gunakan ketika:

- Membuat API yang menerima data dari user.
- Ingin pastikan input valid sebelum diproses.
- Ingin hindari validasi manual yang panjang dan rawan kesalahan.

### validators/catatanValidator.js

```
const { body } = require("express-validator");

// Validasi untuk CREATE
const createCatatanRules = () => [
 body("isi")
    .notEmpty()
    .withMessage("Isi tidak boleh kosong")
    .isLength({ min: 3 })
    .withMessage("Isi minimal 3 karakter"),

body("kategori")
    .optional()
    .isLength({ max: 50 })
    .withMessage("Kategori maksimal 50 karakter"),
];
```

```
// Validasi untuk UPDATE (semua optional, tapi tetap
divalidasi jika dikirim)
const updateCatatanRules = () => [
  body("isi")
    .optional()
    .isLength({ min: 3 })
    .withMessage("Isi minimal 3 karakter"),

  body("kategori")
    .optional()
    .isLength({ max: 50 })
    .withMessage("Kategori maksimal 50 karakter"),
];

module.exports = {
  createCatatanRules,
  updateCatatanRules,
};
```

## validators/paramValidator.js



```
const { param } = require("express-validator");

const validateIdParam = () => [
  param("id")
    .exists()
    .withMessage("ID wajib disertakan")
    .bail()
    .isInt({ gt: 0 })
    .withMessage("ID harus berupa angka lebih dari 0"),
];

module.exports = { validateIdParam };
```

### middlewares/validate.js



```
const errors = validationResult(reg);
 if (!errors.isEmpty()) {
 next();
module.exports = validate;
```

### routes/catatan.js

```
router.get("/catatan/:id", controller.getCatatanById);
```

```
validateIdParam(),
validateIdParam(),
```

### server.js



```
const express = require("express");
const app = express();
const catatanRoutes = require("./routes/catatan");
app.use(express.json());
app.get("/", (req, res) => {
    res.send("API Catatan Siap!");
});

// Daftarkan routing catatan
// Prefix api
app.use("/api", catatanRoutes);
app.listen(5000, () => console.log("Server jalan di http://localhost:5000"));
```

