dicoding

DBS Foundation
Creating IMPACT IMPROVING LIVES ENABLING Change

Hapi Plugin dan Data Validation

Speaker Name | Position

DBS FOUNDATION CODING CAMP

dicoding

DBS Foundation

Creating IMPACT IMPROVING LIVES ENABLING Change

Outline

- Hapi Plugin
- Validasi Data
- Joi
- Custom Exceptions

dicoding

DBS Foundation

Creating IMPACT IMPROVING LIVES ENABLING Change

Objectives

- Mengetahui Hapi Plugin.
- Memisahkan logika bisnis melalui Hapi Plugin.
- Mengetahui pentingnya validasi data.
- Menggunakan Joi sebagai data validator untuk memastikan data yang dikirim pengguna adalah valid.
- Mengimplementasikan custom error untuk merespons client error secara spesifik.

dicoding

DBS Foundation

Creating IMPACT IMPROVING LIVES ENABLING Change

Hapi Plugin

Hapi Plugin

- Plugin pada Hapi digunakan untuk **memudahkan** proses **memisah-misahkan** komponen aplikasi yang Anda buat.
- Komponen dapat berupa **logika bisnis** ataupun **utilitas** yang **sering digunakan** (reusable utilities).
- Dengan demikian, kode menjadi **modular**, **mudah dipelihara**, dan dapat dengan mudah diperluas.

dicoding

DBS Foundation

Creating IMPACT IMPROVING LIVES ENABLING Change

Hapi Plugin

Membuat Plugin

- Untuk membuat Plugin pada Hapi, caranya cukup dengan **membuat sebuah objek** yang memiliki properti `register` sebagai fungsi.
- Fungsi `register` ini akan dijalankan ketika **plugin didaftarkan** pada Hapi server.

```
const notesPlugin = {
  register: async (server, options) => {
    // contoh, menetapkan routing untuk /notes
    const notes = options.notes;
    server.route([
      {
        method: 'GET',
        path: '/notes',
        handler: () => {
          return notes;
        }
      }
    ])
  },
}
```

dicoding

DBS Foundation — Creating IMPACT IMPROVING LIVES ENABLING Change

Mendaftarkan Plugin

- Untuk mendaftarkan Plugin pada Hapi, caranya cukup melalui method `await server.register()`.
- Daftarkan plugin dengan memberikan parameter objek yang memiliki properti `plugin` dan `options`.

```
const notesPlugin = require('./notesPlugin');
...

const init = async () => {
  const server = Hapi.server();

  // registrasi satu plugin
  await server.register({
    plugin: notesPlugin,
    options: { notes: [] },
  });

  await server.start();
};

...
```

dicoding

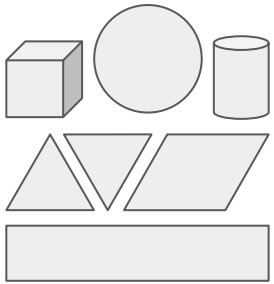
DBS Foundation

Creating IMPACT IMPROVING LIVES ENABLING Change

Validasi Data

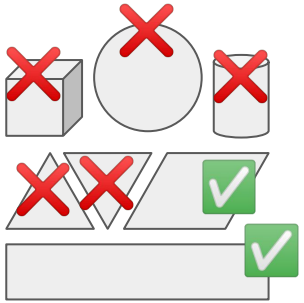
Validasi Data

Proses membandingkan **data yang diperoleh** dengan **serangkaian aturan** yang telah dikonfigurasi atau ditentukan sebelumnya untuk memastikan bahwa data tersebut sesuai dengan persyaratan.



Persyaratan:

- Bangun datar.
- Memiliki titik sudut.
- Minimal 4 sisi.



Validasi Data

- **“Never trust your user”** merupakan salah satu klise yang harus dipedomani ketika mengembangkan aplikasi Back-End.
- Dengan kata lain: Jangan pernah mempercayai informasi apa pun yang dikirimkan oleh pengguna.
- Validasi data memastikan bahwa **data bersih**, **dapat digunakan**, dan **akurat**. Hanya data tervalidasi yang seharusnya boleh disimpan, diimpor, atau digunakan.

Fungsi Validasi Data

- Memastikan pengguna mengirimkan data yang sesuai.
- Mencegah resource dipenuhi oleh data “sampah” atau tak berkualitas.
- Meminimalkan kemungkinan kejahatan yang dilakukan oleh pengguna seperti SQL Injection.
- Memastikan data yang dikirim tidak mengandung unsur yang berbahaya atau dapat mengganggu sistem.

Contoh Validasi Data

- Contoh:
  - **Validasi data tanggal valid dengan format DD/MM/YYYY**

❌ **TIDAK VALID** ❌

- 29/02/2023
- 12/13/1990
- 03-07-2023
- 03/Jul/2023
- 03/07/23

✅ **VALID** ✅

- 25/02/2023
- 24/12/1990
- 03/07/2023

Contoh Validasi Data

- Contoh:
  - **Validasi data password minimal 8 karakter dengan kombinasi hanya angka dan huruf**

❌ **TIDAK VALID** ❌

- Pvk624
- seBzDZaH
- f6GA-B}3

✅ **VALID** ✅

- UBf2tpHG
- 76CLbJZJve

dicoding

DBS Foundation

Creating IMPACT IMPROVING LIVES ENABLING Change

Joi

Joi

- Tools yang populer dan andal untuk melakukan validasi data di JavaScript.
- Joi dibuat oleh Sideway yang merupakan pengembang dari Hapi framework.
- Dalam memvalidasi data, Joi menggunakan pola **schema description**.

Schema pada Joi

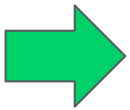
- Untuk membuat schema pada Joi, gunakan method Joi.object().

```
// membuat objek schema
const schema = Joi.object({
  username: Joi.string().alphanum().min(3).max(30).required(),
  password: Joi.string().min(6).required(),
  repeatPassword: Joi.string().required().valid(Joi.ref("password")),
  email: Joi.string().email().required(),
});
```

Validasi Data Menggunakan Joi

Untuk melakukan validasi data, gunakan fungsi `validate()` dari objek schema yang sudah dibuat, lalu berikan nilai atau objek yang hendak divalidasi.

```
// memvalidasi objek berdasarkan schema
const validationResult = schema.validate({
  username: 'harryp',
  password: 'supersecretpassword',
  repeatPassword: 'supersecretpassword',
  email: 'harry@potter.com'
});
```

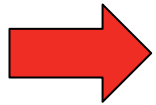


```
{
  value: {
    username: 'harryp',
    password: 'supersecretpassword',
    repeatPassword: 'supersecretpassword',
    email: 'harry@potter.com'
  }
}
```

Validasi Data Menggunakan Joi

Jika data objek tidak memenuhi schema validasi, maka Joi akan mengembalikan data sebagai berikut

```
const validationResult = schema.validate({
        username: "harryp",
        password: "supersecretpassword",
        repeat_password: "supersecretpassword",
        email: "harry",
      });
```



```
{
  "value": {
    "username": "harryp",
    "password": "supersecretpassword",
    "repeat_password": "supersecretpassword",
    "email": "harry"
  },
  "error": {
    "_original": {
      "username": "harryp",
      "password": "supersecretpassword",
      "repeat_password": "supersecretpassword",
      "email": "harry"
    },
    "details": [
      {
        "message": "email must be a valid email",
        "path": ["email"],
        "type": "string.email",
        "context": {
          "value": "harry",
          "invalids": ["harry"],
          "label": "email",
          "key": "email"
        }
      }
    ]
  }
}
```

dicoding

DBS Foundation

Creating IMPACT IMPROVING LIVES ENABLING Change

Custom Exceptions

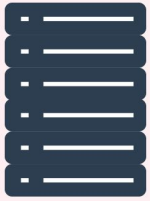
Custom Exceptions

Client

POST/notes

Response code: 200 (OK)
Message: Maaf gagal.

Apa? Bagian mana yang salah?



Server

Client

POST/notes

Response code: 400 (Bad request)
Message: Untuk memasukkan catatan, silakan lampirkan data catatannya.

Oke, baiklah!

Server

Custom Exceptions

- Saat terjadi kesalahan, misalnya tidak ditemukannya resource atau proses validasi data gagal; error yang dibangkitkan selalu menggunakan class `Error` secara generic.
- Hal ini menyulitkan untuk membedakan apa yang menyebabkan error tersebut dibangkitkan.

dicoding

DBS Foundation — Creating IMPACT IMPROVING LIVES ENABLING Change

Custom Exceptions

- Agar dapat mudah mengidentifikasi error secara lebih spesifik, perlu menerapkan **teknik custom error**.
- Custom error bisa dibuat dengan meng-warisi class Error.
- Untuk memanggil, cukup gunakan statement throw.

```
class ClientError extends Error {
  constructor(message, statusCode = 400) {
    super(message);
    this.statusCode = statusCode;
    this.name = 'ClientError';
  }
}

module.exports = ClientError;
```

```
throw new ClientError('Delete failed. Id not found');
```

dicoding

Inherits Custom Exceptions

Kita juga dapat membuat Custom Exceptions baru seperti `InvariantError` dan `NotFoundError` dengan mewarisi class `ClientError` yang kita buat sebelumnya.

```
1  const ClientError = require('./ClientError');
2
3  class InvariantError extends ClientError {
4    constructor(message) {
5      super(message);
6      this.name = 'InvariantError';
7    }
8  }
9
10 module.exports = InvariantError;
```

```
1  const ClientError = require('./ClientError');
2
3  class NotFoundError extends ClientError {
4    constructor(message) {
5      super(message, 404);
6      this.name = 'NotFoundError';
7    }
8  }
9
10 module.exports = NotFoundError;
```

dicoding

DBS Foundation
Creating IMPACT IMPROVING LIVES ENABLING Change

Thank You


nama.instagram


nama.facebook


nama.twitter


alamat.email@gmail.com


nama.youtube

DBS FOUNDATION CODING CAMP