LAPORAN PRAKTIKUM

KONSTRUKSI PERANGKAT LUNAK

MODUL 08

Runtime Configuration dan Internationalization

Disusun Oleh :

Aji Prasetyo Nugroho / 2211104049

S1SE-06-2

Asisten Praktikum :

Muhammad Taufiq Hidayat

Dosen Pengampu :

Riyan Dwi Yulian Prakoso, S.Kom., M.Kom.

PROGRAM STUDI S1 SOFTWARE ENGINEERING

FAKULTAS INFORMATIKA

TELKOM UNIVERSITY PURWOKERTO

2025

A. LANDASAN TEORI

1. Runtime Configuration

   1.) Pengertian

   Runtime Configuration adalah proses pengelolaan nilai konfigurasi aplikasi yang dapat diatur atau diubah saat aplikasi dijalankan (runtime), bukan saat build time. Konfigurasi ini memungkinkan pengembang untuk membuat aplikasi yang lebih fleksibel dan mudah diadaptasi pada berbagai lingkungan seperti pengembangan (development), pengujian (testing), dan produksi (production).

   2.) Tujuan

   Tujuan utama runtime configuration adalah untuk:

   a. Memisahkan logika aplikasi dari data konfigurasi.
   b. Memungkinkan perubahan perilaku aplikasi tanpa perlu re-build.
   c. Meningkatkan keamanan dengan tidak menyimpan data sensitif langsung dalam kode sumber.
   d. Mendukung deploy ke berbagai lingkungan dengan konfigurasi yang berbeda.

   3.) Implementasi Umum

   Di lingkungan Node.js, runtime configuration biasanya menggunakan:

   a. Environment variables (ENV): seperti process.env.API_KEY
   b. File konfigurasi: seperti config.json atau .env
   c. Framework-level config: seperti next.config.js pada Next.js

2. Internationalization (i18n)

   1.) Pengertian

   Internationalization (disingkat i18n karena terdiri dari huruf “i” di awal, “n” di akhir, dan 18 huruf di antaranya) adalah proses desain dan pengembangan perangkat lunak sehingga dapat dengan mudah diadaptasi ke berbagai bahasa dan wilayah tanpa memerlukan perubahan kode yang signifikan.

   2.) Tujuan

   Tujuan dari internationalization antara lain:

   a. Memperluas jangkauan aplikasi ke pengguna global.
   b. Memberikan pengalaman pengguna yang relevan secara lokal (lokalisasi).
   c. Mendukung berbagai format tanggal, mata uang, dan sistem penulisan.

3.) Lokalisasi (l10n)

Lokalisasi adalah langkah lanjutan dari i18n yang menerapkan konten dan format yang spesifik untuk budaya atau bahasa tertentu, seperti:

a. Menerjemahkan teks UI
b. Menyesuaikan format waktu dan tanggal (contoh: DD/MM/YYYY vs MM/DD/YYYY)
c. Menyesuaikan tata letak (misalnya Right-to-Left untuk bahasa Arab/Hebrew)

4.) Implementasi Umum

Beberapa metode dan alat umum dalam i18n:

a. Library seperti i18next, react-intl, atau next-i18next
b. File terjemahan berbasis JSON (misalnya en.json, id.json)
c. Penggunaan fallback language jika terjemahan tidak tersedia

B. GUIDED

1. BankTransferApp.js
   Source Code

```
const readline = require('readline');
const BankTransferConfig = require('../config/BankTransferConfig');

class BankTransferApp {
  constructor() {
    this.config = new BankTransferConfig().config;
    this.rl = readline.createInterface({
      input: process.stdin,
      output: process.stdout
    });
  }

  askQuestion(query) {
    return new Promise(resolve => this.rl.question(query, resolve));
  }

  async run() {
    const lang = this.config.lang;

    const promptAmount = lang === "en" ?
      "Please insert the amount of money to transfer: " :
      "Masukkan jumlah uang yang akan di-transfer: ";

    const amountStr = await this.askQuestion(promptAmount);
    const amount = parseFloat(amountStr);

    const fee = amount <= this.config.transfer.threshold
      ? this.config.transfer.low_fee
      : this.config.transfer.high_fee;
```

```
    const total = amount + fee;

    if (lang === "en") {
      console.log(`Transfer fee = ${fee}`);
      console.log(`Total amount = ${total}`);
    } else {
      console.log(`Biaya transfer = ${fee}`);
      console.log(`Total biaya = ${total}`);
    }

    console.log(lang === "en" ? "Select transfer method:" : "Pilih metode
transfer:");
    this.config.methods.forEach((method, idx) => {
      console.log(`${idx + 1}. ${method}`);
    });

    await this.askQuestion(lang === "en" ? "Choose a method (press Enter after): " :
"Pilih metode (tekan Enter setelah): ");

    const confirmationPrompt = lang === "en" ?
      `Please type "${this.config.confirmation.en}" to confirm the transaction: ` :
      `Ketik "${this.config.confirmation.id}" untuk mengkonfirmasi transaksi: `;

    const confirmationInput = await this.askQuestion(confirmationPrompt);

    if (
      (lang === "en" && confirmationInput.trim().toLowerCase() ===
this.config.confirmation.en) ||
      (lang === "id" && confirmationInput.trim().toLowerCase() ===
this.config.confirmation.id)
    ) {
      console.log(lang === "en" ? "The transfer is completed" : "Proses transfer
berhasil");
    } else {
      console.log(lang === "en" ? "Transfer is cancelled" : "Transfer dibatalkan");
    }

    this.rl.close();
  }
}

module.exports = BankTransferApp;
```

2. BankTransferConfig.js

   Source Code

```
const fs = require('fs');
const path = require('path');

class BankTransferConfig {
  constructor() {
    this.configPath = path.join(__dirname, '../data/bank_transfer_config.json');
```

```
    this.defaultConfig = {
      lang: "en",
      transfer: {
        threshold: 25000000,
        low_fee: 6500,
        high_fee: 15000
      },
      methods: ["RTO (real-time)", "SKN", "RTGS", "BI FAST"],
      confirmation: {
        en: "yes",
        id: "ya"
      }
    };
    this.config = this.loadConfig();
  }

  loadConfig() {
    if (fs.existsSync(this.configPath)) {
      const data = fs.readFileSync(this.configPath, 'utf8');
      return JSON.parse(data);
    } else {
      this.saveConfig(this.defaultConfig);
      return this.defaultConfig;
    }
  }

  saveConfig(config) {
    fs.writeFileSync(this.configPath, JSON.stringify(config, null, 2));
  }
}

module.exports = BankTransferConfig;
```

3. BankTransferData.json
   Source Code

```
{
    "lang": "en",
    "transfer": {
      "threshold": 25000000,
      "low_fee": 6500,
      "high_fee": 15000
    },
    "methods": ["RTO (real-time)", "SKN", "RTGS", "BI FAST"],
    "confirmation": {
      "en": "yes",
      "id": "ya"
    }
  }
```

4. Index.js

   Source Code

```
const BankTransferApp = require('./app/BankTransferApp');

const app = new BankTransferApp();
app.run();
```

   Output

```
PS D:\Modul 08 – Runtime Configuration dan Internationalization> node index.js
Please insert the amount of money to transfer: 1000
Transfer fee = 6500
Total amount = 7500
Select transfer method:
1. RTO (real-time)
2. SKN
3. RTGS
4. BI FAST
Choose a method (press Enter after): 1
Please type "yes" to confirm the transaction: yes
The transfer is completed
```

```
PS D:\Modul 08 – Runtime Configuration dan Internationalization> node index.js
Please insert the amount of money to transfer: 1000
Transfer fee = 6500
Total amount = 7500
Total amount = 7500
Select transfer method:
1. RTO (real-time)
Select transfer method:
1. RTO (real-time)
2. SKN
1. RTO (real-time)
2. SKN
2. SKN
3. RTGS
3. RTGS
4. BI FAST
4. BI FAST
Choose a method (press Enter after): 2
Please type "yes" to confirm the transaction: yes
The transfer is completed
```

```
PS D:\Modul 08 – Runtime Configuration dan Internationalization> node index.js
Please insert the amount of money to transfer: 1000
Transfer fee = 6500
Total amount = 7500
Select transfer method:
Transfer fee = 6500
Total amount = 7500
Select transfer method:
Select transfer method:
1. RTO (real-time)
1. RTO (real-time)
2. SKN
3. RTGS
4. BI FAST
Choose a method (press Enter after): 3
Please type "yes" to confirm the transaction: yes
The transfer is completed
```

```
PS D:\Modul 08 – Runtime Configuration dan Internationalization> node index.js
Please insert the amount of money to transfer: 1000
Transfer fee = 6500
Total amount = 7500
Select transfer method:
1. RTO (real-time)
2. SKN
3. RTGS
4. BI FAST
Choose a method (press Enter after): 4
Please type "yes" to confirm the transaction: yes
The transfer is completed
PS D:\Modul 08 – Runtime Configuration dan Internationalization>
```

C. BONUS CHALLANGE

1. BankTransferApp.js

   Source Code

```
const readline = require('readline');
const BankTransferConfig = require('../config/BankTransferConfig');

class BankTransferApp {
  constructor() {
    this.config = new BankTransferConfig().config;
    this.rl = readline.createInterface({
      input: process.stdin,
      output: process.stdout
    });
  }

  askQuestion(query) {
    return new Promise(resolve => this.rl.question(query, resolve));
  }
```

```
  async run() {
    const lang = this.config.lang;

    const promptAmount = lang === "en" ?
      "Please insert the amount of money to transfer: " :
      "Masukkan jumlah uang yang akan di-transfer: ";

    const amountStr = await this.askQuestion(promptAmount);
    const amount = parseFloat(amountStr);

    const fee = amount <= this.config.transfer.threshold
      ? this.config.transfer.low_fee
      : this.config.transfer.high_fee;

    const total = amount + fee;

    if (lang === "en") {
      console.log(`Transfer fee = ${fee}`);
      console.log(`Total amount = ${total}`);
    } else {
      console.log(`Biaya transfer = ${fee}`);
      console.log(`Total biaya = ${total}`);
    }

    console.log(lang === "en" ? "Select transfer method:" : "Pilih metode
transfer:");
    this.config.methods.forEach((method, idx) => {
      console.log(`${idx + 1}. ${method}`);
    });

    await this.askQuestion(lang === "en" ? "Choose a method (press Enter after): " :
"Pilih metode (tekan Enter setelah): ");

    const confirmationPrompt = lang === "en" ?
      `Please type "${this.config.confirmation.en}" to confirm the transaction: ` :
      `Ketik "${this.config.confirmation.id}" untuk mengkonfirmasi transaksi: `;

    const confirmationInput = await this.askQuestion(confirmationPrompt);

    if (
      (lang === "en" && confirmationInput.trim().toLowerCase() ===
this.config.confirmation.en) ||
      (lang === "id" && confirmationInput.trim().toLowerCase() ===
this.config.confirmation.id)
    ) {
      console.log(lang === "en" ? "The transfer is completed" : "Proses transfer
berhasil");
    } else {
      console.log(lang === "en" ? "Transfer is cancelled" : "Transfer dibatalkan");
    }

    this.rl.close();
```

```
  }
}

module.exports = BankTransferApp;
```

2. BankTransferConfig.js

   Source Code

```
const fs = require('fs');
const path = require('path');

class BankTransferConfig {
  constructor() {
    this.configPath = path.join(__dirname, '../data/bank_transfer_config.json');
    this.defaultConfig = {
      lang: "en",
      transfer: {
        threshold: 25000000,
        low_fee: 6500,
        high_fee: 15000
      },
      methods: ["RTO (real-time)", "SKN", "RTGS", "BI FAST"],
      confirmation: {
        en: "yes",
        id: "ya"
      }
    };
    this.config = this.loadConfig();
  }

  loadConfig() {
    if (fs.existsSync(this.configPath)) {
      const data = fs.readFileSync(this.configPath, 'utf8');
      return JSON.parse(data);
    } else {
      this.saveConfig(this.defaultConfig);
      return this.defaultConfig;
    }
  }

  saveConfig(config) {
    fs.writeFileSync(this.configPath, JSON.stringify(config, null, 2));
  }
}

module.exports = BankTransferConfig;
```

3. BankTransferData.json

   Source Code

```
{
    "lang": "id",
    "transfer": {
      "threshold": 25000000,
      "low_fee": 6500,
      "high_fee": 15000
    },
    "methods": ["RTO (real-time)", "SKN", "RTGS", "BI FAST", "VA (Virtual
Account)"],
    "confirmation": {
      "en": "yes",
      "id": "ya"
    }
  }
```

4. Index.js

   Source Code

```
const BankTransferApp = require('./app/BankTransferApp');

const app = new BankTransferApp();
app.run();
```

   Output

   1.) Ubah bahasa default menjadi id lalu jalankan lagi programnya.

```
PS D:\Modul 08 – Runtime Configuration dan Internationalization> node index.js
Masukkan jumlah uang yang akan di-transfer: 7000
Biaya transfer = 6500
Total biaya = 13500
Pilih metode transfer:
1. RTO (real-time)
2. SKN
3. RTGS
4. BI FAST
5. VA (Virtual Account)
Pilih metode (tekan Enter setelah): 5
Ketik "ya" untuk mengkonfirmasi transaksi: ya
Proses transfer berhasil
PS D:\Modul 08 – Runtime Configuration dan Internationalization>
```

2.) Tambahkan opsi metode transfer baru di file JSON.

```
PS D:\Modul 08 – Runtime Configuration dan Internationalization> node index.js
Masukkan jumlah uang yang akan di-transfer: 7000
Biaya transfer = 6500
Total biaya = 13500
Pilih metode transfer:
1. RTO (real-time)
2. SKN
3. RTGS
4. BI FAST
5. VA (Virtual Account)
```

3.) Apa yang terjadi kalau input transfer amount bukan angka?

Bisa mengakibatkan total amount menjadi NaN dan Tidak ada validasi atau peringatan kepada pengguna.

```
PS D:\Modul 08 – Runtime Configuration dan Internationalization> node index.js
Masukkan jumlah uang yang akan di-transfer: aji
Biaya transfer = 15000
Total biaya = NaN
Pilih metode transfer:
1. RTO (real-time)
2. SKN
3. RTGS
4. BI FAST
5. VA (Virtual Account)
Pilih metode (tekan Enter setelah): 5
Ketik "ya" untuk mengkonfirmasi transaksi: ya
Proses transfer berhasil
PS D:\Modul 08 – Runtime Configuration dan Internationalization>
```

D. PENJELASAN PROGRAM

Program yang terdiri dari beberapa file seperti BankTransferApp.js, BankTransferConfig.js, BankTransferData.json, dan index.js ini merupakan sebuah simulasi aplikasi transfer bank sederhana yang dikonfigurasi berdasarkan data eksternal dan interaksi pengguna melalui terminal atau baris perintah. File BankTransferData.json berperan sebagai sumber data konfigurasi utama, termasuk pengaturan bahasa (lang), batas jumlah transfer (threshold), biaya transfer (low_fee, high_fee), metode transfer yang tersedia, serta konfirmasi dalam berbagai bahasa. Informasi ini kemudian diimpor oleh BankTransferConfig.js, yang membentuk antarmuka untuk mengakses data tersebut dalam format programatik. Selanjutnya, BankTransferApp.js menjadi pusat logika aplikasi, yang menangani alur utama mulai dari input nominal transfer, perhitungan biaya berdasarkan batasan nilai, hingga pemilihan metode dan konfirmasi akhir dari pengguna.

Di sisi lain, index.js bertugas sebagai titik awal eksekusi program, di mana ia menjalankan fungsi utama dari BankTransferApp.js. Program ini memanfaatkan modul readline untuk membaca input dari pengguna secara interaktif melalui terminal, menanyakan jumlah uang yang akan ditransfer, menghitung biaya transfer sesuai dengan jumlah tersebut, menampilkan daftar metode transfer, dan menunggu konfirmasi pengguna. Setelah pengguna memilih metode dan memberikan konfirmasi akhir, sistem akan mencetak detail transaksi. Seluruh struktur program ini menunjukkan pemisahan tanggung jawab yang baik antara konfigurasi, logika, dan antarmuka pengguna, menjadikannya modular dan mudah diperluas di masa depan, seperti penambahan bahasa baru atau metode transfer tambahan.