LAPORAN PRAKTIKUM
KONSTRUKSI PERANGKAT BERGERAK

MODUL II

AUTOMATA DAN TABLE-DRIVEN CONSTRUCTION

Disusun Oleh :

Salman Alfa Rizzi

2211104056

S1SE-06-02

Asisten Praktikum :

Muhamad Taufiq Hidayat

Dosen Pengampu :

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

PROGRAM STUDI S1 REKAYASA PERANGKAT LUNAK

DIREKTORAT TELKOM KAMPUS PURWOKERTO

2025

BAB I
PENDAHULUAN

A. DASAR TEORI

1. Finite State Machine (FSM) adalah model komputasi yang merepresentasikan sistem sebagai kumpulan **state** yang dapat berpindah dari satu state ke state lainnya berdasarkan input yang diterima. FSM sering digunakan dalam pengembangan perangkat lunak, terutama dalam **game development, sistem navigasi, dan automata theory**.

2. **Komponen FSM**

   1. **State** → Kondisi tertentu dalam sistem (misalnya: START, GAME, PAUSE, EXIT).

   2. **Transition** → Perpindahan dari satu state ke state lainnya berdasarkan input (misalnya: "ENTER" mengubah state dari START ke GAME).

   3. **Event/Input** → Perintah atau aksi yang memicu perubahan state (ENTER, ESC, BACK, dll.).

   4. **Initial State** → State awal sebelum terjadi transisi (START).

   5. **Final State** → State di mana sistem berhenti (EXIT).

3. **Metode Implementasi FSM**

   FSM dapat diimplementasikan dengan dua metode utama:

   1. **Automata-Based Construction** → Menggunakan switch-case untuk menangani transisi antar state.

   2. **Table-Driven Construction** → Menggunakan array atau objek sebagai lookup table untuk menentukan transisi berdasarkan input.

   Dalam praktikum ini, kedua metode digunakan untuk menunjukkan cara membangun FSM dalam JavaScript.

4. MAKSUD DAN TUJUAN

Praktikum ini bertujuan untuk:

1.Memahami konsep dasar Finite State Machine (FSM) dalam pengembangan perangkat lunak.
2.Mengimplementasikan FSM menggunakan metode Automata-Based dan Table-Driven Construction dalam JavaScript.
3.Mempelajari cara menangani input pengguna secara interaktif dengan menggunakan readline di Node.js.
4.Menganalisis perbedaan antara kedua metode implementasi FSM dalam hal efisiensi dan fleksibilitas.
5.Menguji sistem FSM yang telah dibuat dengan berbagai input untuk memastikan transisi berjalan sesuai skenario.

BAB II
IMPLEMENTASI (GUIDED)

Game_fsm.js

```
const readline = require("readline");

const rl = readline.createInterface({
    input: process.stdin,
    output: process.stdout
});

const State = {
    START: "START",
    GAME: "GAME",
    PAUSE: "PAUSE",
    HOME: "HOME",
    EXIT: "QUIT"
};

let state = State.START;

function runStateMachine() {
    console.log(`${state} SCREEN`);
    rl.question("Enter Command: ", (command) => {
        switch (state) {
            case State.START:
                if (command === "ENTER") state = State.GAME;
                else if (command === "QUIT") state = State.EXIT;
                break;
            case State.GAME:
                if (command === "ESC") state = State.PAUSE;
                break;
            case State.PAUSE:
                if (command === "BACK") state = State.GAME;
                else if (command === "HOME") state = State.START;
                else if (command === "QUIT") state = State.EXIT;
                break;
        }
        if (state !== State.EXIT) {
            runStateMachine();
        } else {
            console.log("EXIT SCREEN");
            rl.close();
        }
    });
}

runStateMachine();
```

table_lookup.js

```
function getDaysPerMonth(month) {
    const daysPerMonth = [31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31];
    return daysPerMonth[month - 1] || "Invalid month";
}

console.log(getDaysPerMonth(2)); // Output: 28
console.log(getDaysPerMonth(13)); // Output: Invalid month

function getGradeByScore(studentScore) {
    const grades = ["A", "AB", "B", "BC", "C", "D", "E"];
    const rangeLimit = [80, 70, 65, 60, 50, 40, 0];

    for (let i = 0; i < rangeLimit.length; i++) {
        if (studentScore >= rangeLimit[i]) {
            return grades[i];
        }
    }
    return "E";
}

console.log(getGradeByScore(75)); // Output: AB
console.log(getGradeByScore(45)); // Output: D
```

Output:

```
salmans-MacBook-Air:praktikum2 salmanalfarizzi$ node game_fsm.js
START SCREEN
Enter Command: ENTER
GAME SCREEN
Enter Command: ESC
PAUSE SCREEN
Enter Command: HOME
START SCREEN
Enter Command: ENTER
GAME SCREEN
Enter Command: ESC
PAUSE SCREEN
Enter Command: QUIT
EXIT SCREEN
```

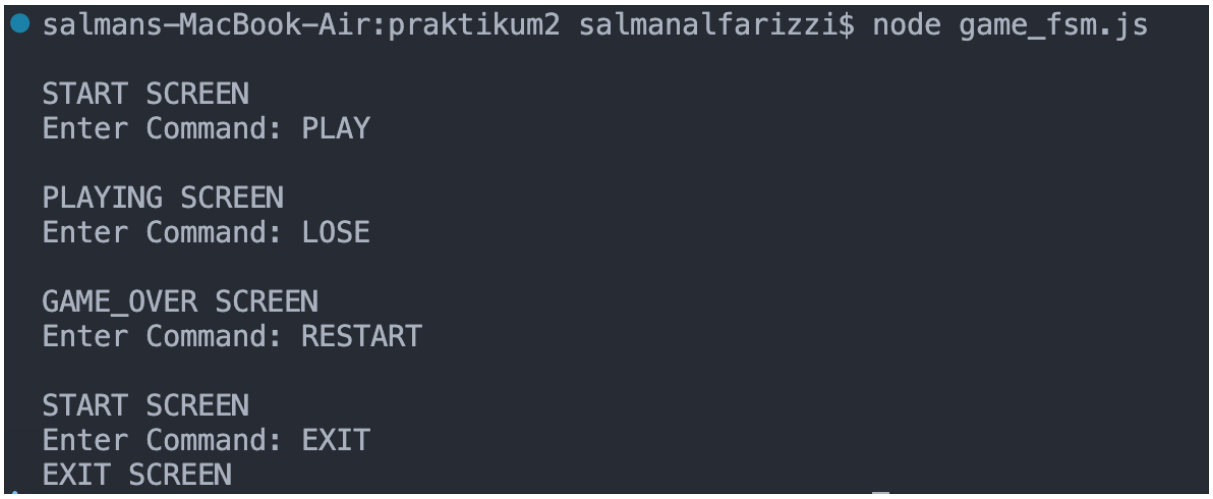
```
salmans-MacBook-Air:praktikum2 salmanalfarizzi$ node table_lookup.js
28
Invalid month
AB
D
```

BAB III
PENUGASAN (UNGUIDED)

Source Code:

```
const readline = require("readline");

const rl = readline.createInterface({
    input: process.stdin,
    output: process.stdout
});

const State = {
    START: "START",
    PLAYING: "PLAYING",
    GAME_OVER: "GAME_OVER"
};

let state = State.START;

function runStateMachine() {
    console.log(`\n${state} SCREEN`);
    rl.question("Enter Command: ", (command) => {
        switch (state) {
            case State.START:
                if (command === "PLAY") state = State.PLAYING;
                else if (command === "EXIT") state = null;
                break;
            case State.PLAYING:
                if (command === "LOSE") state = State.GAME_OVER;
                else if (command === "EXIT") state = null;
                break;
            case State.GAME_OVER:
                if (command === "RESTART") state = State.START;
                else if (command === "EXIT") state = null;
                break;
        }

        if (state) {
            runStateMachine();
        } else {
            console.log("EXIT SCREEN");
            rl.close();
        }
    });
}

runStateMachine();
```

Output:


salmans-MacBook-Air:praktikum2 salmanalfarizzi$ node game_fsm.js
START SCREEN
Enter Command: PLAY
PLAYING SCREEN
Enter Command: LOSE
GAME_OVER SCREEN
Enter Command: RESTART
START SCREEN
Enter Command: EXIT
EXIT SCREEN