Volume 2 Number 1 Special Issue (2024)
Oktober-Desember 2024
Page: 45-55

Tamilis Synex: Multidimensional Collaboration
https://edujavare.com/index.php/TLS/index
E-ISSN: 3026-3972

PENERAPAN FINITE STATE AUTOMATA TERHADAPPROSES PENDAFTARAN PRAKTIKUM PADA JURUSAN TEKNIK INFORMATIKA

Muhammad Oky Risaldi[1], Febrian Ray Gere SW[2], Indra Rajsya[3], Heru Sutejo[4]
[1,2,3,4] Universitas Sepuluh Nopember Papua
* Correspondence e-mail; muhammadokyrisaldi@gmail.com

Article history Submitted: 2024/12/08; Revised: 2024/12/15; Accepted: 2024/12/17:

Abstract Finite State Automata (FSA) is an abstract machine derived from formal language theory, widely applied in computing for validating structured processes. This study explores the application of FSA in automating the practicum registration system for Informatics Engineering students at Universitas Sepuluh Nopember Papua. The registration process previously encountered challenges, such as schedule conflicts between practicum and lectures, overcapacity of classes, and data input errors, which hindered academic activities. To address these issues, FSA was employed to ensure automated validation at each step of the registration process. The system operates through a series of states, beginning from course selection to practicum schedule validation, shift assignment, and confirmation. The FSA model designed consists of five states: initial state, course selection state, validation state, shift selection state, and confirmation state. If an error occurs, the system provides feedback and redirects students to correct their input. Using a structured approach, this model prevents scheduling conflicts and ensures data consistency. The methodology integrates observation, interviews, and literature reviews to identify problems and design the FSA model. The system was implemented using a programming framework, validated through simulations using both valid and invalid registration data. Results demonstrated that the system effectively minimized errors, improved the accuracy of practicum registration, and streamlined the process. In conclusion, the implementation of FSA significantly enhances the efficiency and organization of practicum registration, providing automated feedback and ensuring students successfully complete their registrations without conflicts or input errors.

Keywords Finite State Automata, practicum registration, automated validation, scheduling conflicts, efficiency

PENDAHULUAN

Teori automata adalah cabang ilmu yang membahas mesin-mesin abstrak dan memiliki keterkaitan erat dengan teori bahasa formal. Teori ini sangat bermanfaat dalam pengembangan ilmu komputer, baik di bidang perangkat keras (hardware)

maupun perangkat lunak (software) (Hopcroft et al., 2007). Finite State Automata (FSA) adalah mesin abstrak berbasis teori bahasa formal yang memastikan data sesuai format tertentu. FSA memodelkan proses matematika untuk mengolah masukan yang dikenali dan menghasilkan keluaran sesuai aturan sistem (Sipser, 2012). Finite State Automata (FSA) selalu berada dalam keadaan awal ketika mulai membaca tape. Pergantian state terjadi pada mesin setiap kali sebuah karakter terbaca. Ketika head telah mencapai akhir tape dan berada dalam state akhir, maka string yang ada pada tape dianggap diterima oleh Finite Automata (Lin, 2011).

Finite State Automata (FSA) terdiri dari dua jenis utama: Deterministic Finite Automata (DFA) dan Non-deterministic Finite Automata (NFA). Perbedaan utama keduanya terletak pada pola transisinya, di mana DFA hanya memungkinkan satu arah transisi untuk setiap state, sedangkan NFA dapat memiliki lebih dari satu arah transisi. FSA memiliki sifat-sifat sebagai berikut: pita masukkan berisi rangkaian simbol dari alfabet, di mana setiap kali membaca satu karakter, posisi read head berpindah ke simbol berikutnya. FSA selalu berada dalam status tertentu, dan jumlah status yang ada pada FSA adalah berhingga. Secara formal, Finite State Automata (FSA) didefinisikan sebagai 5-tupel, yaitu M=(Q, Σ, δ, S, F), dengan rincian: Q adalah himpunan state, Σ adalah himpunan simbol input, δ adalah fungsi transisi antar state, S adalah state awal, dan F adalah himpunan state akhir. Jumlah state akhir pada sebuah FSA dapat lebih dari satu (Hopcroft & Ullman, 1979).

Proses pendaftaran praktikum di Jurusan Teknik Informatika Universitas Sepuluh Nopember Papua sering menghadapi berbagai kendala yang memengaruhi efisiensi dan akurasi sistem. Salah satu masalah yang sering terjadi adalah ketidakcocokan jadwal praktikum dengan mata kuliah yang diambil mahasiswa, sehingga menyebabkan konflik jadwal yang menghambat kelancaran kegiatan akademik. Selain itu, kesalahan dalam pemilihan shift praktikum, seperti kapasitas kelas yang melebihi batas atau jadwal yang bertabrakan, juga sering menjadi sumber masalah.

Ketidaktepatan input data oleh mahasiswa maupun kurangnya validasi otomatis pada sistem pendaftaran yang ada menjadi faktor utama terjadinya kesalahan ini. Oleh karena itu, diperlukan sebuah sistem yang dapat meminimalkan kesalahan melalui validasi otomatis dan memastikan alur pendaftaran yang terstruktur. Metode pengumpulan data yang digunakan dalam penelitian ini adalah studi literatur, di mana referensi yang digunakan berasal dari artikel jurnal yang relevan dengan permasalahan yang dibahas. Pendekatan berbasis Finite State Automata (FSA) dapat

digunakan untuk membangun sistem yang mampu memverifikasi setiap langkah proses pendaftaran, mulai dari pemilihan mata kuliah hingga penjadwalan praktikum, sehingga menciptakan pengalaman yang lebih efisien dan terorganisir bagi mahasiswa (Rajasekaran & Devaraj, 2017).

Penelitian ini bertujuan untuk mengidentifikasi dan mengatasi kendala dalam proses pendaftaran praktikum, seperti ketidakcocokan jadwal praktikum dengan mata kuliah yang diambil mahasiswa serta kesalahan dalam pemilihan shift praktikum yang dapat mengganggu kelancaran kegiatan akademik. Selain itu, penelitian ini berfokus pada penerapan Finite State Automata (FSA) dalam validasi input data, pemilihan mata kuliah, dan penjadwalan praktikum, sehingga dapat meningkatkan efisiensi dan akurasi pengelolaan pendaftaran praktikum dengan menyediakan sistem yang otomatis mendeteksi dan mengoreksi kesalahan. Fokus sistem yang dibangun dalam penelitian ini hanya mencakup proses pendaftaran praktikum untuk mahasiswa Jurusan Teknik Informatika Universitas Sepuluh Nopember Papua. Penelitian ini tidak mencakup pengembangan perangkat keras atau sistem lain di luar pendaftaran praktikum.

METODE

Metode penelitian ini menggunakan pendekatan Finite State Automata (FSA) untuk membangun sistem pendaftaran praktikum di Universitas Sepuluh Nopember Papua. Langkah pertama adalah identifikasi masalah melalui observasi proses pendaftaran, wawancara dengan mahasiswa dan pengelola sistem, serta analisis dokumen terkait seperti panduan akademik dan data historis pendaftaran. Selanjutnya, dilakukan studi literatur untuk memahami konsep dasar FSA, implementasi pada sistem validasi otomatis, dan studi kasus serupa di institusi lain. Model FSA dirancang dengan menetapkan himpunan state, simbol input, fungsi transisi, state awal, dan state akhir, yang mencakup tahapan pemilihan mata kuliah, validasi input, dan konfirmasi pendaftaran. Implementasi model dilakukan melalui pemrograman algoritma FSA, validasi data pada setiap state, dan penyajian feedback otomatis bagi pengguna jika terjadi kesalahan. Simulasi sistem dilakukan untuk menguji skenario pendaftaran dengan data valid dan tidak valid, batas kapasitas kelas, serta konflik jadwal. Hasil simulasi dievaluasi untuk menilai efisiensi sistem dalam meminimalkan kesalahan dan kemudahan penggunaannya. Proses ini diakhiri dengan dokumentasi hasil yang mencakup desain model FSA, implementasi sistem, serta analisis hasil simulasi dan evaluasi untuk penyempurnaan lebih lanjut.

HASIL DAN PEMBAHASAN

Daftar Praktikum

Gambar 1. Daftar Praktikum

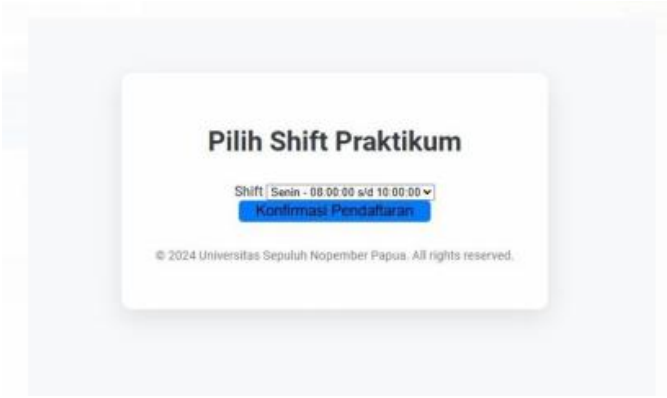
Pendaftaran Praktikum Valid

Gambar 2. Form Pendaftaran Praktikum Valid

Pendaftaran Praktikum Tidak Valid

Gambar 3. Pendaftaran Praktikum Tidak Valid

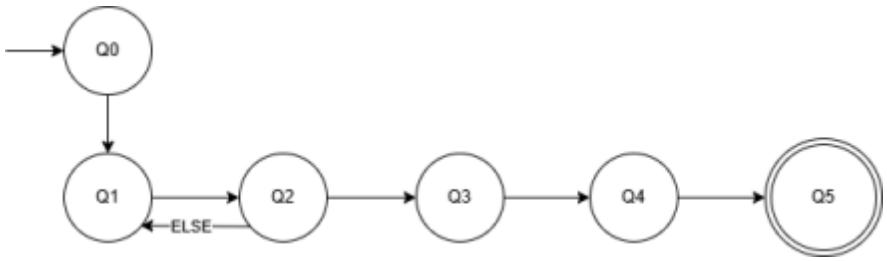
Form Shift



Gambar 4. Form Shift

Pembahasan

Model FSA



Gambar 5. Model FSA

Model FSA ini merepresentasikan alur pendaftaran praktikum dengan lima state utama yang merepresentasikan tahapan proses. Proses dimulai dari state awal q0, di mana mahasiswa membuka halaman pendaftaran dan melakukan aksi selectMataKuliah untuk memilih matakuliah, yang membawa sistem ke state q1. Di q1, mahasiswa memilih praktikum melalui aksi selectPraktikum, dan sistem melanjutkan ke state q2 untuk melakukan validasi data pendaftaran. Pada state q2, jika validasi berhasil melalui aksi submitValidation (valid), sistem berpindah ke state q3, di mana mahasiswa dapat memilih shift melalui aksi selectShift. Selanjutnya, sistem masuk ke state q4, tempat mahasiswa mengonfirmasi data pendaftaran melalui aksi submitConfirmation, yang akhirnya membawa sistem ke state q5 sebagai state akhir, menandakan bahwa pendaftaran berhasil. Namun, jika validasi di state q2 gagal melalui aksi submitValidation (invalid), sistem mengarahkan kembali ke state q1, memungkinkan mahasiswa memperbaiki data yang salah. Alur ini memastikan validasi berlapis dan alur logis untuk setiap langkah pendaftaran.

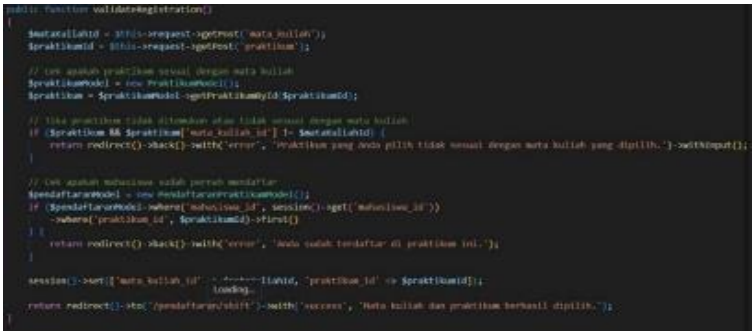
Fungsi Index

```php
public function index()
{
    $mataKuliahModel = new MataKuliahModel();
    $praktikumModel = new PraktikumModel();

    $data['mata_kuliah'] = $mataKuliahModel->findAll();
    $data['praktikum'] = $praktikumModel->findAll();

    return view('registration', $data); // Ganti dengan nama view yang sesuai
}
```

Gambar 6. Fungsi Index

Fungsi index dalam kode tersebut bertujuan untuk menyiapkan dan mengirimkan data ke halaman view di aplikasi berbasis CodeIgniter 4 (CI4). Dalam fungsi ini, dua model diinisialisasi, yaitu MataKuliahModel untuk mengakses data mata kuliah dan PraktikumModel untuk mengakses data praktikum. Selanjutnya, fungsi findAll() pada kedua model digunakan untuk mengambil semua data dari masing-masing tabel terkait di database. Data yang diperoleh kemudian disusun ke dalam sebuah array bernama $data dengan kunci 'mata_kuliah' dan 'praktikum'. Array ini kemudian diteruskan ke view dengan menggunakan fungsi view() untuk memuat halaman yang ditentukan, dalam hal ini view bernama 'registration'. Fungsi ini memungkinkan data mata kuliah dan praktikum ditampilkan dalam halaman view tersebut, yang dapat digunakan untuk berbagai kebutuhan seperti pendaftaran atau pengelolaan data.

Fungsi Validasi Registrasi



Gambar 7. Fungsi Validasi Registrasi

Fungsi validateRegistration bertujuan untuk memastikan validitas pendaftaran mahasiswa ke sebuah praktikum. Fungsi ini dimulai dengan mengambil data input dari form, yaitu ID mata kuliah dan ID praktikum, menggunakan metode $this->request->getPost(). Selanjutnya, menggunakan model PraktikumModel, fungsi ini mengambil data praktikum berdasarkan ID praktikum yang diberikan. Validasi pertama dilakukan dengan mengecek apakah praktikum

yang dipilih sesuai dengan mata kuliah yang telah dipilih. Jika tidak sesuai, proses dikembalikan ke halaman sebelumnya dengan pesan kesalahan dan input yang sebelumnya dimasukkan. Setelah itu, validasi kedua dilakukan untuk memastikan mahasiswa belum pernah mendaftar ke praktikum yang sama. Cek ini dilakukan menggunakan Pendaftaran Praktikum Model, yang mencari data mahasiswa dan praktikum di tabel pendaftaran. Jika ditemukan bahwa mahasiswa sudah terdaftar, proses juga dikembalikan dengan pesan kesalahan. Jika semua validasi terpenuhi, data ID mata kuliah dan ID praktikum disimpan dalam sesi, dan mahasiswa diarahkan ke halaman pemilihan shift dengan pesan sukses.

Fungsi Select Shift

```
public function selectShift()
{
    $shiftModel = new ShiftPraktikumModel();
    $praktikumId = session()->get('praktikum_id');

    // Mengambil data shift berdasarkan praktikum
    $data['shifts'] = $shiftModel->getShiftByPraktikum($praktikumId);

    return view('select_shift', $data); // Ganti dengan nama view yang sesuai
}
```

Gambar 8. Fungsi select Shift

Fungsi select Shift bertujuan untuk menampilkan daftar shift yang tersedia berdasarkan praktikum yang telahdipilih sebelumnya oleh mahasiswa. Fungsi ini dimulai dengan mengambil ID praktikum yang tersimpan dalam sesi melalui session()->get('praktikum_id'). Kemudian, model Shift Praktikum Model digunakan untuk mengambil data shift yang terkait dengan praktikum tersebut melalui metode getShiftByPraktikum($praktikumId). Data shift yang diperoleh disimpan dalam array $data dengan kunci 'shifts'. Setelah itu, fungsi memuat halaman view bernama 'select_shift' (atau nama view lain yang sesuai), yang akan menampilkan daftar shift tersebut untuk dipilih oleh mahasiswa. Hal ini memungkinkan mahasiswa untuk melanjutkan proses pendaftaran dengan memilih shift yang diinginkan.

Fungsi Confirmasi Registrasi

```
public function confirmRegistration()
{
    $mataKuliahId = session()->get('mata_kuliah_id');
    $praktikumId = session()->get('praktikum_id');
    $shiftId = $this->request->getPost('shift');

    // Simpan pendaftaran ke database
    $pendaftaranModel = new PendaftaranPraktikumModel();
    $data = [
        'mahasiswa_id' => session()->get('mahasiswa_id'), // Ambil ID mahasiswa dari session
        'praktikum_id' => $praktikumId,
        'shift_id' => $shiftId,
        'status' => 'Menunggu Konfirmasi',
        'tanggal_pendaftaran' => date('Y-m-d H:i:s'),
    ];

    $pendaftaranModel->insert($data);

    // Redirect ke halaman sukses
    return redirect()->to('/pendaftaran/success')->with('success', 'Pendaftaran berhasil, silakan tunggu konfirmasi.');
}
```

Gambar 9. Fungsi Confirmasi Registrasi

Fungsi confirm Registration bertugas untuk menyelesaikan proses pendaftaran praktikum dengan menyimpan data pendaftaran mahasiswa ke dalam database. Fungsi ini dimulai dengan mengambil data dari sesi, yaitu ID mata kuliah dan ID praktikum, serta data shift yang dipilih dari input form menggunakan $this-> request-> get Post ('shift'). Selanjutnya, model PendaftaranPraktikumModel digunakan untuk menyimpan data pendaftaran ke tabel database. Data yang disimpan mencakup ID mahasiswa (diambil dari sesi), ID praktikum, ID shift, status pendaftaran yang diset sebagai "Menunggu Konfirmasi", dan waktu pendaftaran saat ini menggunakan fungsi date ('Y-m-d H:i:s'). Setelah data berhasil disimpan ke database menggunakan metode insert, mahasiswa diarahkan ke halaman sukses dengan pesan konfirmasi bahwa pendaftaran telah berhasil dan diminta menunggu proses konfirmasi lebih lanjut. Fungsi ini memastikan bahwa data pendaftaran tercatat dengan lengkap dan sesuai alur sistem.

Fungsi Success

```
public function success()
{
    $mahasiswaId = session()->get('mahasiswa_id'); // Ambil ID mahasiswa dari session

    // Ambil data pendaftaran berdasarkan mahasiswa yang login
    $pendaftaranModel = new PendaftaranPraktikumModel();
    $pendaftaran = $pendaftaranModel->getPendaftaranByMahasiswa($mahasiswaId);

    // Load model Mahasiswa, Praktikum, dan Shift untuk mendapatkan informasi tambahan
    $mahasiswaModel = new MahasiswaModel();
    $praktikumModel = new PraktikumModel();
    $shiftModel = new ShiftPraktikumModel();

    foreach ($pendaftaran as &$data) {
        // Ambil nama mahasiswa
        $data['mahasiswa'] = $mahasiswaModel->getMahasiswaById($data['mahasiswa_id'])['nama'];

        // Ambil nama praktikum
        $data['praktikum'] = $praktikumModel->getPraktikumById($data['praktikum_id'])['nama_praktikum'];

        // Ambil informasi shift
        $shift = $shiftModel->getShiftPraktikumById($data['shift_id']);
        $data['shift_hari'] = $shift['hari'];
        $data['shift_jam_mulai'] = $shift['jam_mulai'];
        $data['shift_jam_selesai'] = $shift['jam_selesai'];
    }

    return view('success', ['pendaftaran' => $pendaftaran]); // Pass data to the view
}
```

Gambar 10. Fungsi Success

Fungsi success bertujuan untuk menampilkan informasi pendaftaran praktikum yang berhasil dilakukan oleh mahasiswa. Fungsi ini diawali dengan mengambil ID mahasiswa dari sesi menggunakan session ()- > get ('mahasiswa_id'). Kemudian, model Pendaftaran Praktikum Model digunakan untuk mengambil semua data pendaftaran yang terkait dengan mahasiswa tersebut melalui metode get Pendaftaran By Mahasiswa ($mahasiswaId). Untuk melengkapi informasi pada data pendaftaran, fungsi ini memuat tiga model tambahan, yaitu MahasiswaModel, PraktikumModel, dan ShiftPraktikumModel. Dalam proses iterasiterhadap data pendaftaran, fungsi ini menambahkan nama mahasiswa, nama praktikum, serta detail shift (hari, jam mulai, dan jam selesai) dengan mengambil data masing- masing dari model terkait. Setelah semua informasi diproses dan dilengkapi,

data tersebut diteruskan ke halaman view bernama 'success' untuk ditampilkan kepada mahasiswa. Fungsi ini memastikan bahwa mahasiswa dapat melihat detail lengkap dari pendaftaran yang telah mereka lakukan, menciptakan pengalaman pengguna yang informatif dan transparan.

KESIMPULAN

Finite State Automata (FSA) dalam proses pendaftaran praktikum di Jurusan Teknik Informatika Universitas Sepuluh Nopember Papua adalah bahwa FSA dapat menyediakan solusi efektif untuk memvalidasi dan memverifikasi alur pendaftaran secara otomatis. Dengan sistem yang terstruktur mulai dari pemilihan mata kuliah, jadwal praktikum, hingga shift praktikum, FSA dapat memastikan bahwa mahasiswa tidak mengalami kesalahan dalam memilih jadwal atau shift praktikum yang berbenturan dengan mata kuliah yang diambil atau melebihi kapasitas kelas. Proses ini menjamin bahwa pendaftaran praktikum berjalan dengan lancar dan terorganisir. Jika terjadi kesalahan, sistem memberikan umpan balik dan memandu mahasiswa untuk memperbaiki pilihan mereka hingga pendaftaran praktikum berhasil dilakukan. Dengan demikian, penerapan FSA dalam pendaftaran praktikum dapat meningkatkan efisiensi, mengurangi kesalahan input, dan menciptakan pengalaman pendaftaran yang lebih terstruktur bagi mahasiswa.

REFERENSI

Ahmad, P. S., & Informatika, U. (2024). "Penerapan Finite State Automata pada Proses Pendaftaran Praktikum Jurusan Informatika Universitas Ahmad Dahlan." *Jurnal Teknik Informatika*, 10(1), 14–18.

Aziz, F. (2021). "Penerapan Konsep Finite State Automata Dalam Proses Pendaftaran Kelas Kursus Bahasa Inggris Pada Tempat Kursus." *Matics*, 12(2), 93–98. https://doi.org/10.18860/mat.v12i2.9330.

Di, S., Teknik, F., & Uncp, K. (2020). "1549-2900-1-Sm." *Jurnal Teknik Informatika*, 10, 5–9.

Febrian, A., Selan, M. O., Purnama, R., Bernada, R. S., & Saifudin, A. (2023). "Pemanfaatan Finite State Automata dalam Pencarian Klinik serta Rumah Sakit Terdekat." *BIIKMA Bul. Ilm. Ilmu Komput. dan Multimed.*, 1(1), 166–173. Available at https://jurnalmahasiswa.com/index.php/biikma.

Handayani, K., Ismunandar, D., Putri, S. A., & Gata, W. (2021). "Penerapan Finite State Automata Pada Vending Machine Susu Kambing Etawa." *Matics*, 12(2), 87–92.

https://doi.org/10.18860/mat.v12i2.9270.

Hopcroft, J. E., & Ullman, J. D. (1979). *Automata Theory, Languages, and Computation*. Addison-Wesley.

Hopcroft, J. E., Motwani, R., & Ullman, J. D. (2007). *Introduction to Automata Theory, Languages, and Computation* (3rd ed.). Pearson Education.

Hopcroft, J. E., Motwani, R., & Ullman, J. D. (2007). *Introduction to Automata Theory, Languages, and Computation* (3rd ed.). Pearson Education.

Lin, C. (2011). *Formal Languages and Automata Theory*. Cambridge University Press.

NurkhalifahAkbal, A., Julianto, R., Nisa, S. K., & Saifudin, A. (2023). "Pengarsipan Dokumen Akreditasi Sekolah Menggunakan Penerapan Finite State Automata." *BIIKMA Bul. Ilm. Ilmu Komput. dan Multimed.*, 1(2), 189–197. Available at https://jurnalmahasiswa.com/index.php/biikma.

Rajasekaran, S., & Devaraj, D. (2017). "Application of Finite State Automata in Workflow Systems." *International Journal of Advanced Research in Computer Science*, 8(7), 1234-1240.

Rivanie, T. (2020). "Implementasi Finite State Automata dalam Proses Registrasi Workout Plan pada Pusat Kebugaran." *Matics*, 12(1), 94. https://doi.org/10.18860/mat.v12i1.8573.

Shubuh, S., Kurniawati, L., Hermaliani, E. H., & Gata, W. (2023). "Implementasi Konsep Finite State Automata (FSA) Untuk Pengolahan Data Santri Pada Pondok Pesantren." *J. Ilm. Ilmu Komput.*, 9(1), 23–27.

Sipser, M. (2012). *Introduction to the Theory of Computation* (3rd ed.). Cengage Learning.

Widodo, A. Z. P., Gata, W., Rahayu, S., Putra, J. L., & Kurniawati, L. (2022). "Implementasi Finite State Automata pada Mesin Otomatis Es Krim." *JTIK (Jurnal Teknol. Inf. dan Komunikasi)*, 6(4), 519–525. https://doi.org/10.35870/jtik.v6i4.522.

Yulianty, V., et al. (2021). "Penerapan Finite State Automata Pada Pengajuan Berkas." *Proceedings*, 282–289.