Djtechno : Jurnal Teknologi Informasi
Vol. 7, No. 2, Agustus 2026
E-ISSN: 2745-3758, P-ISSN : 2776-8546 DOI: 10.46576/djtechno

PEMODELAN *FINITE STATE MACHINE* UNTUK NOTIFIKASI *REAL-TIME* PADA PELACAKAN PESANAN MAKANAN

Zulfahmi Indra[1], Nafil Rizq Trianto [2], Azril Arfansyah[3], Arion Pardede[4], Alfarizi Wijaya[5], Muhammad Iqbal Fahrezzi[6], Calvin Sahputra Buulolo[7]
1,2,3,4,5,6,7) Prodi Ilmu Komputer, Fakultas Matematika dan Ilmu Pengetahuan Alam, Universitas Negeri Medan, Indonesia

Article Info

Article history:

Received: 10 Juni 2026
Revised: 15 Juni 2026
Accepted: 12 Juli 2026

ABSTRACT

Abstrak

Sistem pelacakan pesanan makanan pada platform layanan pesan-antar sering kali menghadapi permasalahan ketidaktransparanan informasi status secara *real-time*, yang menimbulkan miskomunikasi antara pelanggan, restoran, dan pengemudi. Penelitian ini bertujuan merancang dan mengimplementasikan sistem pelacakan pesanan makanan berbasis *Finite State Machine* (FSM) dengan integrasi notifikasi *real-time* melalui *Telegram Bot API*. Sistem dibangun menggunakan arsitektur *REST API* dan *WebSocket* berbasis *Node.js* dengan enam *state* deterministik, yaitu *NEW*, *CONFIRMED*, *PREPARED*, *PICKUP*, *DELIVERED*, dan *CANCELLED*, yang membentuk siklus hidup pesanan secara ketat. Metode pengembangan mengikuti pendekatan *waterfall* terstruktur, dan pengujian dilakukan menggunakan teknik *black-box testing* untuk memverifikasi fungsionalitas transisi *state* serta keandalan pengiriman notifikasi. Hasil penelitian menunjukkan bahwa pemodelan FSM secara efektif mencegah anomali transisi status melalui mekanisme penolakan otomatis pada setiap permintaan transisi yang tidak valid. Sistem berhasil mendistribusikan notifikasi secara *real-time* kepada penerima yang relevan berdasarkan peran, yakni pelanggan, restoran, dan pengemudi, pada setiap perubahan *state*. Seluruh skenario pengujian black-box menghasilkan *output* yang sesuai dengan hasil yang diharapkan. Penelitian ini membuktikan bahwa FSM merupakan fondasi desain yang andal untuk membangun sistem pelacakan layanan berbasis *web* yang transparan dan bebas dari kesalahan transisi status.

Kata Kunci: *Finite State Machine*, Pelacakan pesanan makanan, Notifikasi *real-time*, *Telegram Bot API*, *REST API*.

Abstract

Food delivery order tracking systems frequently encounter transparency issues regarding real-time status updates, leading to miscommunication among customers, restaurants, and drivers. This study aims to design and implement a food order tracking system based on Finite State Machine (FSM) with real-time notification integration through the Telegram Bot API. The system was built using a Node.js-based REST API and WebSocket architecture with six deterministic states, namely NEW, CONFIRMED, PREPARED, PICKUP, DELIVERED, and CANCELLED, which strictly govern the order lifecycle. The development method followed a structured waterfall approach, and testing was conducted using black-box testing techniques to verify state transition functionality and notification delivery reliability. The results demonstrate that FSM modeling effectively prevents status transition anomalies through an automatic rejection mechanism applied to every invalid transition request. The system successfully distributes real-time notifications to role-based recipients, including customers, restaurants, and drivers, upon each state change. All black-box testing scenarios produced outputs consistent with expected results. This study proves that FSM serves as a reliable design foundation for building transparent, web-based service tracking systems that are free from status transition errors.

Keywords: Finite State Machine, food order tracking, real-time notification, Telegram Bot API, REST API.

Djtechno : Jurnal Teknologi Informasi
Vol. 7, No. 2, Agustus 2026
E-ISSN: 2745-3758, P-ISSN : 2776-8546 DOI: 10.46576/djtechno

Corresponding Author:
E-mail : mrarion2006@gmail.com

1. PENDAHULUAN

Industri layanan pesan-antar makanan mengalami pertumbuhan yang signifikan seiring meningkatnya adopsi platform digital oleh masyarakat, khususnya di Indonesia. Namun, pertumbuhan tersebut kerap diiringi oleh permasalahan operasional yang mendasar: ketidakjelasan status pesanan secara *real-time*. Pelanggan sering kali tidak mengetahui apakah pesanan mereka telah dikonfirmasi oleh restoran, sedang diproses di dapur, telah diambil oleh pengemudi, atau sudah dalam perjalanan menuju lokasi pengiriman. Ketidaktransparanan informasi ini menimbulkan miskomunikasi antara pelanggan, pihak restoran, dan pengemudi, yang pada akhirnya berujung pada penurunan kepuasan layanan dan kemungkinan pembatalan pesanan. Permasalahan serupa juga dijumpai dalam sistem-sistem berbasis antrian di lingkungan layanan makanan maupun layanan pengaduan, di mana ketiadaan notifikasi perkembangan status membuat pelanggan merasa tidak diinformasikan dengan baik [1], [2], [3].

Finite State Machine (FSM) merupakan model komputasi matematis yang memungkinkan representasi perilaku sistem ke dalam sejumlah *state* diskrit dengan transisi yang deterministik berdasarkan input atau aksi tertentu. Dalam konteks sistem informasi, FSM banyak diterapkan untuk memodelkan alur kerja yang memerlukan pelacakan status secara ketat, mulai dari pengembangan *game* dengan perilaku karakter adaptif, sistem kontrol dan keamanan pada perangkat keras, hingga aplikasi pengelolaan tugas berbasis *web* [4], [5], [6]. Daya tarik utama FSM terletak pada kemampuannya mencegah transisi status yang tidak valid: sistem hanya dapat berpindah dari satu *state* ke *state* lain apabila kondisi transisi terpenuhi, sehingga anomali status dapat dihilangkan secara struktural. Karakteristik ini sangat relevan untuk diterapkan dalam pelacakan pesanan makanan, di mana urutan status mulai dari pesanan baru,

Djtechno : Jurnal Teknologi Informasi
Vol. 7, No. 2, Agustus 2026
E-ISSN: 2745-3758, P-ISSN : 2776-8546 DOI: 10.46576/djtechno

dikonfirmasi, disiapkan, diambil pengemudi, hingga terkirim harus berjalan secara deterministik.

Integrasi notifikasi *real-time* ke dalam sistem pelacakan status merupakan kebutuhan yang semakin mendesak. Penelitian tentang sistem reservasi restoran berbasis *web* membuktikan bahwa notifikasi instan melalui API pihak ketiga seperti *Telegram Bot API* mampu meningkatkan efisiensi serta konsistensi komunikasi antara sistem dan manajemen secara signifikan [7]. Mekanisme serupa telah diimplementasikan pula pada sistem informasi pengaduan berbasis *web* yang menggunakan FSM sebagai pengendali alur kerja, di mana setiap perubahan status secara otomatis memicu notifikasi kepada pihak terkait melalui *email*. Pendekatan ini menunjukkan bahwa kombinasi FSM sebagai mesin pengendali status dan kanal notifikasi *real-time* sebagai media penyampaian informasi merupakan fondasi arsitektur yang kokoh untuk sistem pelacakan berbasis *web*.

Penelitian ini bertujuan merancang dan mengimplementasikan sistem pelacakan pesanan makanan yang menggunakan pemodelan FSM dengan enam *state* deterministik, yakni *NEW*, *CONFIRMED*, *PREPARED*, *PICKUP*, *DELIVERED*, dan *CANCELLED*. Sistem dibangun di atas arsitektur *REST API* dan *WebSocket* berbasis *Node.js* dan *Express.js*, serta mengintegrasikan notifikasi *real-time* melalui *Telegram Bot API*. Permasalahan utama yang dijawab adalah: bagaimana FSM dapat mencegah anomali transisi status pesanan, dan bagaimana arsitektur sistem dapat menjamin penyampaian notifikasi secara instan kepada seluruh pihak yang berkepentingan. Penelitian ini berkontribusi pada pengembangan sistem informasi manajemen pesanan yang transparan, efisien, dan dapat diandalkan dalam konteks operasional layanan pengiriman makanan, sekaligus memberikan referensi implementasi FSM pada domain layanan di luar pengembangan permainan digital [4].

2. METODE PENELITIAN

2.1. Rancangan Arsitektur Sistem

Sistem FSM Order Tracker dikembangkan menggunakan pendekatan pengembangan perangkat lunak berbasis metode *waterfall* yang terstruktur, mencakup tahapan analisis kebutuhan, perancangan, implementasi, dan pengujian secara berurutan. Arsitektur sistem mengadopsi pola *client-server* dengan tiga lapisan utama.

Djtechno : Jurnal Teknologi Informasi
Vol. 7, No. 2, Agustus 2026
E-ISSN: 2745-3758, P-ISSN : 2776-8546 DOI: 10.46576/djtechno

Lapisan pertama adalah antarmuka pengguna berbasis HTML, CSS, dan JavaScript murni yang berfungsi sebagai *Single Page Application*. Lapisan kedua adalah *server backend* berbasis *Node.js* dan *Express.js* yang menangani seluruh logika bisnis, pengelolaan *state* FSM, dan komunikasi *real-time*. Lapisan ketiga adalah *in-memory data store* yang menyimpan data pesanan secara sementara selama sesi server berjalan. Komunikasi antara klien dan server dilakukan melalui dua mekanisme: *REST API* berbasis HTTP untuk operasi CRUD pesanan, dan *WebSocket* menggunakan *Socket.io* untuk pembaruan status secara *real-time* kepada semua klien yang terhubung.

Arsitektur *REST API* yang diimplementasikan menyediakan sejumlah *endpoint* terstruktur. *Endpoint* $GET\ /api/orders$digunakan untuk mengambil daftar seluruh pesanan yang aktif. *Endpoint POST* $/api/orders$digunakan untuk membuat pesanan baru dengan status awal *NEW*. *Endpoint POST* $/api/orders/:id/transition$ digunakan untuk memicu transisi *state* berdasarkan aksi yang dikirimkan. *Endpoint POST* $/api/orders/:id/assign-driver$digunakan untuk menugaskan pengemudi pada pesanan yang telah mencapai *state PREPARED*. Selain itu, *endpoint* $GET/api/stats$ menyediakan statistik sistem secara agregat. Desain *REST API* yang eksplisit dan terstruktur ini mengikuti prinsip-prinsip pembangunan layanan *web* modern dan otomasi kontrol yang memisahkan tanggung jawab setiap *endpoint* berdasarkan fungsinya [8].

2.2. Desain *Finite State Machine*

Inti dari sistem ini adalah mesin FSM yang diimplementasikan dalam modul *fsm.js*. FSM dirancang sebagai *Deterministic Finite Automata* (DFA) yang didefinisikan secara formal sebagai 5-tupel M = (Q, Sigma, delta, S, F), di mana Q adalah himpunan *state*, Sigma adalah himpunan aksi input, delta adalah fungsi transisi, S adalah *state* awal, dan F adalah himpunan *state* akhir. Dalam konteks sistem ini, himpunan *state* Q terdiri dari enam elemen: *NEW*, *CONFIRMED*, *PREPARED*, PICKUP, *DELIVERED*, dan *CANCELLED*, yang dirancang sesuai kaidah mesin abstrak DFA [9], [10], [11].

State NEW merupakan *state* awal yang ditetapkan secara otomatis saat pesanan dibuat. Transisi dari *NEW* menuju *CONFIRMED* dipicu oleh aksi "confirm", yang merepresentasikan konfirmasi pesanan oleh pihak restoran. Dari *CONFIRMED*, sistem dapat bertransisi ke *PREPARED* melalui aksi "prepare" ketika makanan selesai disiapkan oleh dapur. *State PREPARED* memungkinkan transisi menuju PICKUP melalui aksi "pickup" ketika pengemudi mengambil pesanan dari restoran. Akhirnya, dari PICKUP

Djtechno : Jurnal Teknologi Informasi
Vol. 7, No. 2, Agustus 2026
E-ISSN: 2745-3758, P-ISSN : 2776-8546 DOI: 10.46576/djtechno

sistem bertransisi ke *DELIVERED* melalui aksi "deliver" sebagai penanda penyelesaian pengiriman. *State DELIVERED* dan *CANCELLED* adalah *state* terminal yang tidak memiliki transisi keluar. Aksi "cancel" dapat dipicu dari *state NEW*, *CONFIRMED*, *PREPARED*, maupun PICKUP, sehingga pembatalan dapat dilakukan pada hampir seluruh tahapan sebelum pengiriman selesai. Fungsi transisi delta diimplementasikan sebagai tabel transisi yang diperiksa setiap kali aksi diterima; apabila kombinasi *state* aktif dan aksi yang diterima tidak terdapat dalam tabel transisi, sistem menolak permintaan tersebut dan mengembalikan pesan kesalahan. Mekanisme penolakan inilah yang mencegah anomali status, seperti pesanan yang langsung berpindah dari *NEW* ke *DELIVERED* tanpa melalui tahapan yang seharusnya [12].

2.3. Integrasi Notifikasi *Real-Time*

Sistem notifikasi dirancang untuk mendistribusikan pesan kepada tiga entitas berbeda berdasarkan *state* yang aktif. Pelanggan menerima notifikasi pada seluruh transisi *state*. Pihak restoran menerima notifikasi pada *state NEW*, *CONFIRMED*, *PREPARED*, PICKUP, *DELIVERED*, dan *CANCELLED*. Pengemudi hanya menerima notifikasi mulai dari *state PREPARED* hingga *state* terminal, karena keterlibatan pengemudi baru relevan sejak makanan siap diambil. Integrasi dengan *Telegram Bot API* dilakukan melalui modul *telegram.js* yang mengirimkan pesan berformat Markdown ke *Chat ID* masing-masing penerima menggunakan metode HTTP *POST* ke *endpoint* sendMessage API Telegram. Penggunaan Telegram sebagai kanal notifikasi dipilih karena sifatnya yang gratis, instan, tidak memerlukan nomor telepon aktif secara terus-menerus, dan mendukung format pesan yang kaya [13], [14]. Notifikasi *WebSocket* juga dikirimkan secara bersamaan melalui *Socket.io* untuk memperbarui antarmuka *dashboard* secara *real-time* tanpa perlu memuat ulang halaman.

2.4. Teknik Pengujian

Pengujian sistem dilakukan menggunakan pendekatan *black-box testing* yang berfokus pada verifikasi fungsionalitas berdasarkan input dan *output* yang diamati, tanpa menganalisis kode internal. Pengujian mencakup empat aspek utama. Pertama, pengujian transisi *state* yang memverifikasi bahwa setiap aksi valid menghasilkan perpindahan *state* yang benar dan setiap aksi tidak valid ditolak sistem. Kedua, pengujian notifikasi yang memverifikasi bahwa pesan terkirim ke penerima yang tepat pada setiap transisi *state*. Ketiga, pengujian *REST API* menggunakan perangkat pengujian berbasis

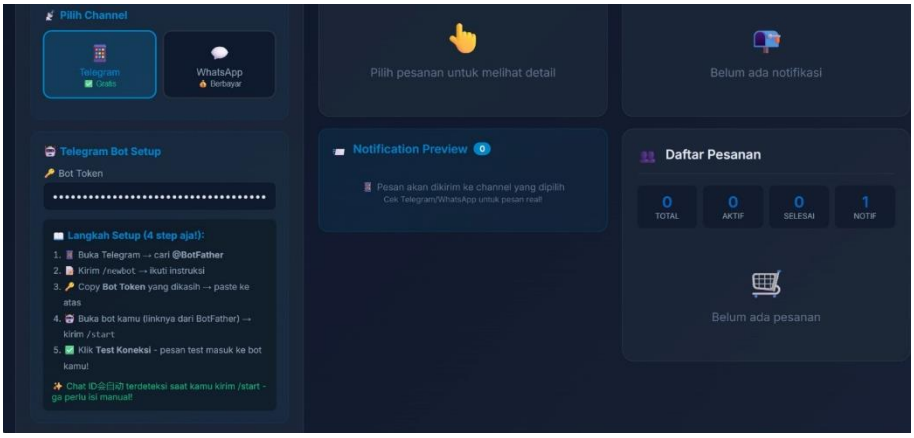
Djtechno : Jurnal Teknologi Informasi
Vol. 7, No. 2, Agustus 2026
E-ISSN: 2745-3758, P-ISSN : 2776-8546 DOI: 10.46576/djtechno

curl untuk memvalidasi respons HTTP dari setiap *endpoint*. Keempat, pengujian antarmuka yang memverifikasi pembaruan tampilan secara *real-time* melalui *WebSocket* saat *state* pesanan berubah. Skenario pengujian disusun untuk mencakup alur normal dari *NEW* hingga *DELIVERED*, alur pembatalan dari berbagai *state* antara, dan skenario anomali berupa percobaan transisi yang tidak valid untuk membuktikan keandalan mekanisme penolakan FSM.

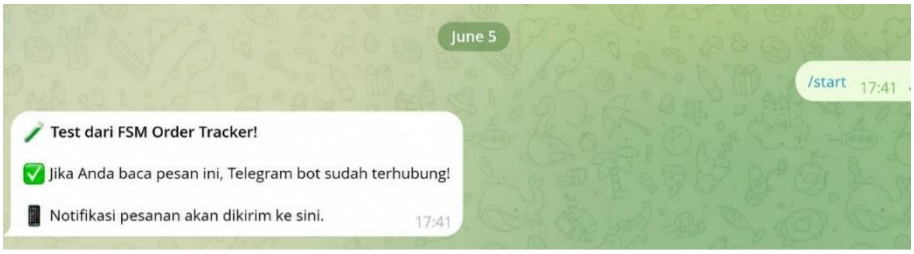
3. HASIL DAN PEMBAHASAN

3.1. Implementasi Antarmuka *Dashboard* dan Konfigurasi Sistem

Sistem FSM Order Tracker diimplementasikan sebagai aplikasi *web* satu halaman yang terbagi ke dalam tiga panel fungsional utama. Panel kiri menangani konfigurasi *webhook* Telegram dan formulir pembuatan pesanan. Panel tengah menampilkan daftar pesanan aktif beserta kontrol transisi *state*. Panel kanan menyajikan *log* notifikasi yang diperbarui secara *real-time* melalui *WebSocket*. Sebelum sistem beroperasi, pengguna menghubungkan Bot Token Telegram melalui panel konfigurasi yang tersedia, dan inisialisasi dilakukan dengan mengirimkan perintah */start* pada aplikasi Telegram untuk merekam *Chat ID* secara otomatis. Mekanisme *auto-detect Chat ID* ini menghilangkan kebutuhan konfigurasi manual yang berpotensi menimbulkan kesalahan input.



Gambar 1 Antarmuka *dashboard* utama FSM Order Tracker

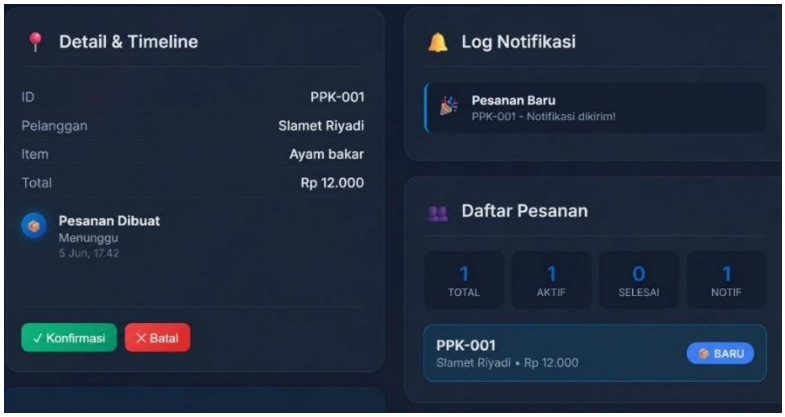


Gambar 2 Inisialisasi bot Telegram melalui *command /start*

Djtechno : Jurnal Teknologi Informasi
Vol. 7, No. 2, Agustus 2026
E-ISSN: 2745-3758, P-ISSN : 2776-8546 DOI: 10.46576/djtechno

3.2. Pembuatan Pesanan dan Penetapan *State* Awal FSM

Setiap pesanan yang dibuat oleh sistem secara otomatis ditempatkan pada *state* awal *NEW*. Dari *state* ini, FSM hanya mengizinkan dua jalur transisi yang valid: aksi "Konfirmasi" menuju *state CONFIRMED*, atau aksi "Batal" menuju *state* terminal *CANCELLED*. Pembatasan jalur transisi ini merupakan manifestasi langsung dari sifat deterministik automata, di mana untuk setiap pasangan *state* aktif dan input yang diterima, hanya terdapat tepat satu keputusan transisi yang mungkin diambil. Anomali seperti pesanan yang langsung berpindah ke *state PICKUP* atau *DELIVERED* tanpa melalui tahapan yang semestinya tidak dapat terjadi secara struktural.

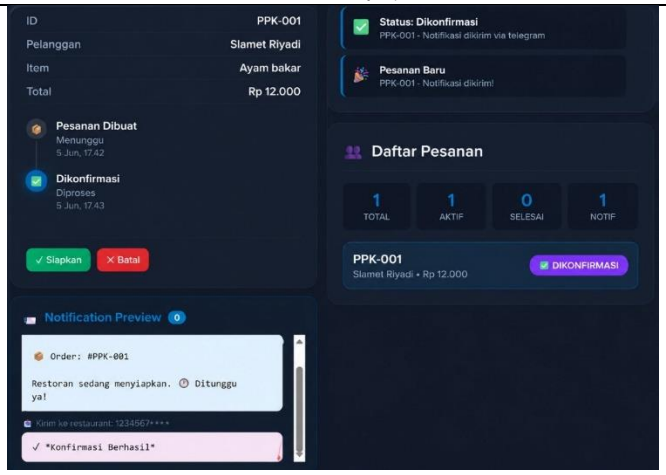


Gambar 3 *State* awal *NEW* dengan dua opsi transisi valid

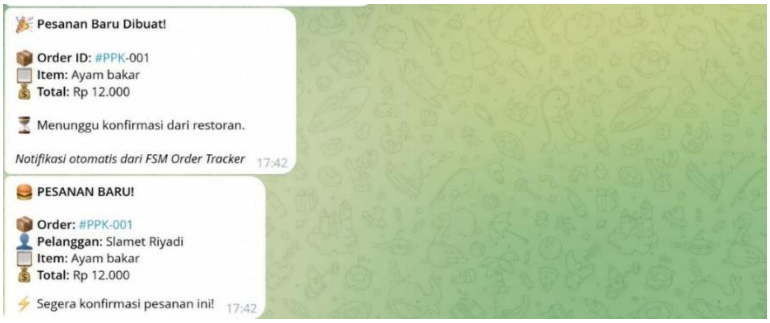
3.3. Visualisasi Transisi *State* FSM dan Notifikasi Berdasarkan Peran

Ketika aksi "Konfirmasi" dieksekusi, sistem secara simultan memperbarui *state* internal pesanan menjadi *CONFIRMED*, mendistribusikan notifikasi melalui *Telegram Bot API* kepada pelanggan dan restoran, serta menyiarkan pembaruan melalui *WebSocket* ke seluruh klien yang terhubung. Kontrol aksi pada antarmuka diperbarui secara otomatis menjadi tombol "Siapkan" sebagai satu-satunya transisi valid dari *state CONFIRMED*. Notifikasi yang dikirimkan menggunakan templat pesan yang berbeda untuk masing-masing penerima sesuai perannya: pelanggan menerima konfirmasi bahwa pesanannya sedang diproses, sementara restoran menerima pesan yang mendorong segera memulai persiapan makanan [15], [16].

Djtechno : Jurnal Teknologi Informasi
Vol. 7, No. 2, Agustus 2026
E-ISSN: 2745-3758, P-ISSN : 2776-8546 DOI: 10.46576/djtechno

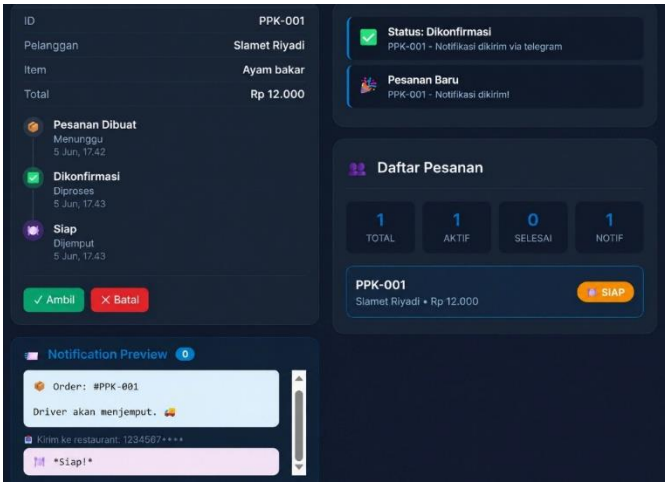


Gambar 4 Transisi ke *state CONFIRMED* dan *log* notifikasi *real-time*



Gambar 5 Distribusi notifikasi multi-penerima pada *state NEW*

Transisi ke *state PREPARED* terjadi ketika restoran mengeksekusi aksi "Siapkan". Pada tahap ini, pengemudi mulai diikutsertakan sebagai penerima notifikasi karena keterlibatannya menjadi relevan sejak makanan siap untuk diambil. Antarmuka memperbarui kontrol aksi menjadi tombol "Ambil", yang secara semantik merepresentasikan pemindahan tanggung jawab dari pihak restoran kepada pengemudi. Pelanggan menerima pemberitahuan bahwa makanan sedang menunggu penjemputan, sementara pengemudi mendapat informasi nama restoran beserta rincian item pesanan yang harus diambil.

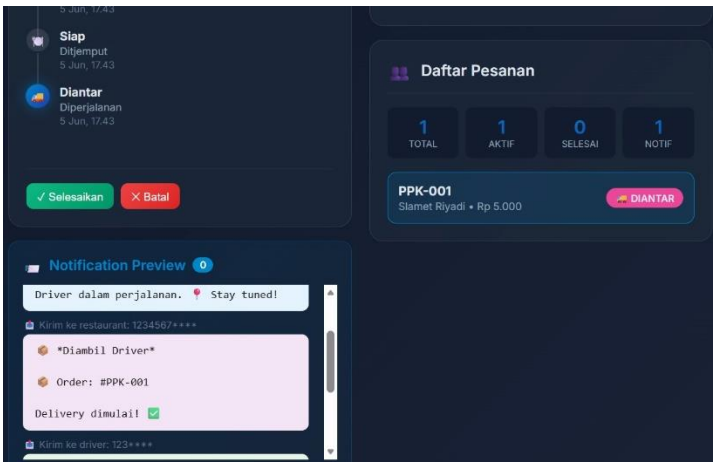


Gambar 6 Transisi ke *state PREPARED* dan pembaruan kontrol aksi

Djtechno : Jurnal Teknologi Informasi
Vol. 7, No. 2, Agustus 2026
E-ISSN: 2745-3758, P-ISSN : 2776-8546 DOI: 10.46576/djtechno

Gambar 7 Notifikasi beruntun pada transisi *CONFIRMED* dan *PREPARED*

Saat aksi "Ambil" dieksekusi, FSM bertransisi ke *state PICKUP* dan tombol aksi final "Selesai" muncul pada antarmuka. Pengemudi menerima instruksi untuk menuju titik pengambilan di restoran, sementara pelanggan menerima konfirmasi bahwa pengemudi telah berangkat menuju lokasi pengiriman. *Log* pratinjau notifikasi pada *dashboard* mencatat parameter pengiriman secara kronologis untuk keperluan penelusuran.



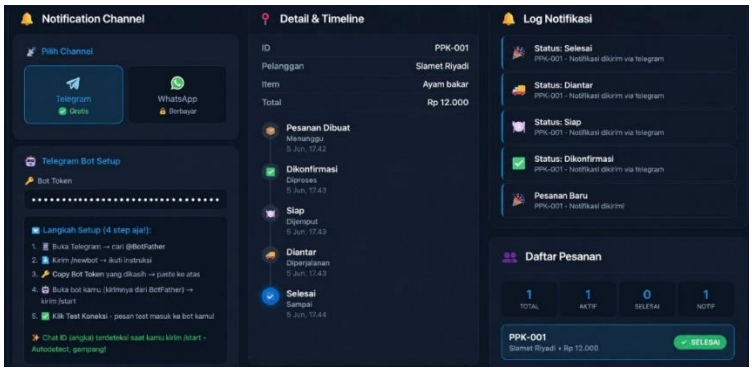
Gambar 8 Transisi ke *state PICKUP* dan kemunculan tombol aksi "Selesai"

Gambar 9 Notifikasi pelibatan pengemudi pada *state PICKUP*

Djtechno : Jurnal Teknologi Informasi
Vol. 7, No. 2, Agustus 2026
E-ISSN: 2745-3758, P-ISSN : 2776-8546 DOI: 10.46576/djtechno

3.4. Pencapaian *State* Terminal dan Penyelesaian Siklus FSM

Eksekusi aksi "Selesai" membawa FSM mencapai *state* terminal *DELIVERED*. Pada kondisi ini, seluruh tombol aksi transisi menghilang dari antarmuka karena tidak ada lagi jalur transisi yang tersedia dari *state* terminal. *Timeline* pesanan menampilkan rekam jejak lengkap dari seluruh transisi yang telah dilalui sejak *state NEW*. Sistem mengirimkan notifikasi penutup kepada pelanggan dan pengemudi sebagai konfirmasi final penyelesaian pengiriman, yang sekaligus menandai berakhirnya siklus komputasi pelacakan pesanan untuk transaksi yang bersangkutan.



Gambar 10 *State* terminal *DELIVERED* dengan *timeline* lengkap

Gambar 11 Notifikasi final pada *state* terminal *DELIVERED*

3.5. Pengujian Black-Box Fungsi Transisi FSM

Pengujian black-box dilakukan untuk memverifikasi seluruh fungsi transisi FSM dan mekanisme notifikasi pada FSM Order Tracker. Skenario mencakup alur normal *NEW* hingga *DELIVERED*, alur pembatalan dari berbagai *state* antara, dan percobaan transisi tidak valid melalui manipulasi permintaan API secara langsung. Hasil pengujian disajikan pada Tabel 1.

Djtechno : Jurnal Teknologi Informasi
Vol. 7, No. 2, Agustus 2026
E-ISSN: 2745-3758, P-ISSN : 2776-8546 DOI: 10.46576/djtechno

Tabel 1 Hasil pengujian black-box fungsi transisi FSM

No	Skenario Pengujian	Hasil yang Diharapkan	Hasil Pengujian	Kesimpulan
1	Tombol "Konfirmasi" diklik pada *state NEW*	*State* berpindah ke *CONFIRMED*; notifikasi terkirim ke pelanggan dan restoran; tombol aksi berubah menjadi "Siapkan"	Sesuai harapan	Valid
2	Tombol "Siapkan" diklik pada *state CONFIRMED*	*State* berpindah ke *PREPARED*; notifikasi terkirim ke tiga penerima; tombol aksi berubah menjadi "Ambil"	Sesuai harapan	Valid
3	Tombol "Ambil" diklik pada *state PREPARED*	*State* berpindah ke *PICKUP*; notifikasi terkirim; tombol "Selesai" muncul	Sesuai harapan	Valid
4	Tombol "Selesai" diklik pada *state PICKUP*	*State* berpindah ke *DELIVERED*; notifikasi penutup terkirim; tombol aksi hilang	Sesuai harapan	Valid
5	Tombol "Batal" diklik pada *state NEW*	*State* berpindah ke *CANCELLED*; notifikasi terkirim ke pelanggan dan restoran; tombol aksi hilang	Sesuai harapan	Valid
6	Tombol "Batal" diklik pada *state CONFIRMED*	*State* berpindah ke *CANCELLED*; notifikasi terkirim ke penerima yang sesuai	Sesuai harapan	Valid
7	Tombol "Batal" diklik pada *state PREPARED*	*State* berpindah ke *CANCELLED*; notifikasi terkirim ke penerima yang sesuai	Sesuai harapan	Valid
8	Aksi "deliver" dikirim via API pada pesanan berstatus *CONFIRMED*	Server mengembalikan HTTP 400; *state* tidak berubah; tidak ada notifikasi terkirim	Sesuai harapan	Valid
9	Aksi "*pickup*" dikirim via API pada pesanan berstatus *NEW*	Server mengembalikan HTTP 400; *state* tidak berubah; tidak ada notifikasi terkirim	Sesuai harapan	Valid
10	Aksi "confirm" dikirim via API pada pesanan berstatus *DELIVERED*	Server mengembalikan HTTP 400; *state* tidak berubah; tidak ada notifikasi terkirim	Sesuai harapan	Valid
11	Pembaruan status via *WebSocket* saat transisi *state* terjadi	Seluruh klien terhubung menerima pembaruan tampilan *real-time* tanpa reload halaman	Sesuai harapan	Valid

Seluruh sebelas skenario pada Tabel 1 menghasilkan *output* yang sesuai dengan yang diharapkan. Skenario nomor 8, 9, dan 10 secara khusus menguji ketahanan lapisan validasi FSM terhadap percobaan *bypass* melalui permintaan API langsung, dan ketiganya membuktikan bahwa mekanisme penolakan beroperasi di tingkat server secara independen dari antarmuka pengguna. Integritas *state* pesanan tetap terjaga meskipun permintaan transisi tidak valid dikirimkan dengan melewati kontrol visual pada *dashboard*.

3.6. Analisis Pencegahan Anomali Status

Hipotesis penelitian bahwa pemodelan FSM mampu mencegah anomali transisi status secara struktural terbukti terpenuhi melalui dua mekanisme yang bekerja secara berlapis. Mekanisme pertama berupa penolakan transisi tidak valid di sisi server yang beroperasi tanpa bergantung pada antarmuka. Mekanisme kedua berupa penyesuaian

Djtechno : Jurnal Teknologi Informasi
Vol. 7, No. 2, Agustus 2026
E-ISSN: 2745-3758, P-ISSN : 2776-8546 DOI: 10.46576/djtechno

dinamis kontrol aksi pada antarmuka yang hanya menampilkan pilihan transisi yang valid dari *state* aktif. Kedua mekanisme ini memastikan siklus hidup pesanan selalu mengikuti jalur yang telah didefinisikan dalam tabel fungsi transisi delta FSM. Penelitian pada sistem informasi pengaduan maupun implementasi *behavior NPC* pada *game* berbasis FSM membuktikan bahwa transisi tidak valid dapat dicegah secara sistematis melalui aturan transisi yang eksplisit [17], [18]. Pada sistem manajemen tugas berbasis *REST API*, integrasi FSM terbukti meningkatkan kejelasan logika bisnis sekaligus mencegah perubahan status yang tidak sah. Implementasi FSM pada sistem kontrol robot lengan pun membuktikan konsistensi keputusan transisi dalam kondisi operasional yang berulang.

3.7. Perbandingan dengan Penelitian Sejenis

FSM Order Tracker yang dikembangkan pada penelitian ini dapat diposisikan di antara beberapa implementasi FSM dalam domain informasi yang serupa. Sistem informasi pengaduan berbasis FSM pada organisasi logistik, sistem manajemen bengkel berbasis DFA, dan sistem pelacakan status pada perangkat food court masing-masing membuktikan efektivitas pendekatan *state*-based dalam menjaga keteraturan alur kerja layanan [19], [20]. FSM Order Tracker melengkapi penelitian-penelitian tersebut dengan fokus spesifik pada integrasi notifikasi Telegram multi-penerima yang bersifat *role-based*, serta penggabungan *REST API* dan *WebSocket* sebagai mekanisme pembaruan ganda yang saling melengkapi. Pemodelan automata yang bersifat deterministik, sebagaimana dikonfirmasi dalam berbagai penelitian tentang simulator DFA dan implementasi mesin abstrak, menjadi fondasi yang menjamin validitas logika transisi. Aspek keterlacakan status dan keamanan alur kerja yang menjadi perhatian pada sistem berbasis FSM di domain permainan digital maupun sistem kontrol juga terwujud dalam FSM Order Tracker melalui *log* notifikasi kronologis yang dapat ditelusuri secara menyeluruh melalui antarmuka *dashboard*.

4. SIMPULAN

Penelitian ini berhasil membuktikan bahwa pemodelan *Finite State Machine* dengan enam *state* deterministik (*NEW*, *CONFIRMED*, *PREPARED*, *PICKUP*, *DELIVERED*, *CANCELLED*) merupakan pendekatan yang efektif untuk membangun sistem pelacakan pesanan makanan yang terstruktur, transparan, dan bebas dari anomali transisi status. Hipotesis penelitian terjawab secara konkret melalui mekanisme penolakan transisi

Djtechno : Jurnal Teknologi Informasi
Vol. 7, No. 2, Agustus 2026
E-ISSN: 2745-3758, P-ISSN : 2776-8546 DOI: 10.46576/djtechno

tidak valid di sisi server yang berjalan secara konsisten pada seluruh skenario pengujian, membuktikan bahwa FSM tidak sekadar memodelkan alur kerja secara konseptual, melainkan juga menegakkan integritas status secara struktural dalam implementasi nyata. Arsitektur sistem yang menggabungkan *REST API* berbasis *Node.js* untuk pengelolaan pesanan, *WebSocket* melalui *Socket.io* untuk pembaruan *real-time* pada *dashboard*, dan *Telegram Bot API* untuk notifikasi multi-penerima berbasis peran (pelanggan, restoran, dan pengemudi) terbukti bekerja secara sinergis dalam menyampaikan informasi status kepada seluruh pemangku kepentingan secara instan pada setiap terjadinya transisi *state*. Sistem yang dihasilkan memberikan kontribusi pada pengembangan solusi manajemen pesanan di industri kuliner digital, sekaligus memperluas cakupan aplikasi FSM ke domain layanan berbasis *web* yang selama ini lebih banyak dikaji dalam konteks pengembangan permainan digital, sistem kontrol perangkat keras, dan otomasi mesin abstrak.

REFERENCES

[1] D. M. Divito, A. S. Budi, and E. Setiawan, "Implementasi Finite State Machine pada Sistem Notifikasi Pesanan Food Court," *J. Pengemb. Teknol. Inf. dan Ilmu Komput.*, vol. 6, no. 7, pp. 3247–3253, 2022.

[2] Farhanuddin and Triase, "Sistem Informasi Pengaduan Dengan Fitur Tracking Status Menggunakan Model Finite State Machine," *Decod. J. Pendidik. Teknol. Inf.*, vol. 5, no. 3, pp. 953–966, 2025.

[3] M. Kamuri, S. D. I. Mau, and M. W. Malo, "Sistem Informasi Aplikasi Pelayanan Pengaduan Masyarakat Dikantor Pertanahan Kabupaten Sumba Barat Daya," *Repeater Publ. Tek. Inform. dan Jar.*, vol. 4, no. 1, pp. 90–104, 2026.

[4] R. Muzawi, Y. Efendi, U. Rio, and D. Toresa, "Security System For Motorcycle Using Raspberry Pi and Fingerprint With Mobile-Based Finite State Machine (FSM) Method," *Innov. Res. Informatics*, vol. 5, no. 1, pp. 14–21, 2023.

[5] A. Israel, W. Coendrad, R. Peday, and H. Sutejo, "Implementasi NFA dan DFA untuk pemodelan alur layanan pada sistem informasi bengkel berbasis web," *J. Informatics, Intell. Tecnhnology, Cybersecurity Softw.*, vol. 1, no. 1, pp. 48–54, 2026.

[6] B. Wicaksono and Y. A. Sekolah, "IMPLEMENTASI FINITE STATE MACHINE (FSM) PADA *GAME* EDUKASI 2D UNTUK PENGENALAN FASILITAS UMUM ANAK USIA DINI," *JATI (Jurnal Mhs. Tek. Inform.*, vol. 9, no. 6, pp. 9715–9722, 2025.

[7] R. Wadhwa, "ENSURING DATA CONSISTENCY IN ENTERPRISE SYSTEMS THROUGH EVENT DRIVEN MICROSERVICES AND DISTRIBUTED TRANSACTION MANAGEMENT," *Int. J. Comput. Sci. Eng. Res. Dev.*, vol. 06, no. 1, pp. 24–51, 2023.

[8] D. Erivianto and A. Dani, "Implementasi Teori Finite State Machine Pada Sistem Kontrol Robot Lengan Berbasis Arduino dan Python," *J. Artif. Intell. Digit. Bus.*, vol. 4, no. 4, pp. 5185–5195, 2025.

[9] Agustin, A. Evel, Susanti, and Rahmaddeni, "Implementasi Metode Finite State Machine Pada Permainan Tradisional Setatak Berbasis Android," *J. Tek. Inform. dan Sist. Inf.*, vol. 8, no. 2, pp. 738–751, 2021.

[10] K. Astoni, F. Aziz, F. Said, D. Andriyanto, and W. Gata, "Penerapan Finite State Automata Pada Mesin Tiket Otomatis Bus Damri Di Bandara Internasional Yogyakarta," *Paradigma*, vol. 23, no. 2, pp. 167–173, 2021.

[11] M. F. Nabhan *et al.*, "Simulator Mesin Deterministic Finite Automata (DFA) Berdasarkan Diagram Transisi Menggunakan Python," *J. Sist. Komput. dan Kecerdasan Buatan*, vol. 7, no. 1, pp. 23–30, 2023.

[12] S. Farhan, I. Fitri, and Fauziah, "Implementasi Transisi Finite State Automata dengan Aplikasi Mesin Abstrak DFA dan NFA Berbasis Android," *J. Ilmu-ilmu Inform. dan Manaj. STMIK*, vol. 15, no. 1, pp.

Djtechno : Jurnal Teknologi Informasi
Vol. 7, No. 2, Agustus 2026
E-ISSN: 2745-3758, P-ISSN : 2776-8546 DOI: 10.46576/djtechno

22–29, 2021.

[13] M. R. S, M. F. Gozali, and S. Mallu, "PENGEMBANGAN WEBSITE RESERVASI ONLINE RESTORAN YANG TERINTEGRASI DENGAN TELEGRAM SEBAGAI NOTIFIKASI REAL- TIME," *J. Comput. Sci. Inf. Technol.*, vol. 2, no. 3, pp. 357–369, 2025.

[14] V. M. Ayu, W. Gata, J. L. Putra, F. Frieyadie, and H. B. Novitasari, "Finite State Automata pada Vending Machine Pembuat Teh Otomatis," *JTIK (Jurnal Teknol. Inf. dan Komunikasi)*, vol. 6, no. 4, pp. 561–570, 2022.

[15] R. H. Rusyard, H. Farisi, and W. Purnomo, "PENGEMBANGAN FITUR TASK MANAGEMENT PADA DELOS AQUAHERO SERVICE MENGGUNAKAN FINITE STATE MACHINE," *J. Pengemb. Teknol. Inf. dan Ilmu Komput.*, vol. 9, no. 6, pp. 1–10, 2025.

[16] A. Fernando, Guntoro, L. Costaner, M. Devaga, and Lisnawita, "Penerapan Metode Finite State Machine Pada Game Pembelajaran Matematika," *J. Karya Ilm. Multidisiplin*, vol. 3, no. 1, pp. 60–68, 2023.

[17] A. Pranselga, I. R. Setiawan, and W. Apriandari, "Implementasi Finite State Machine Pada Karakter NPC Musuh Dalam Game Adventure In Java," *Jutisi J. Ilm. Tek. Inform. dan Sist. Inf.*, vol. 10, no. 3, pp. 392–404, 2021.

[18] D. Maulana, M. N. Dila, and M. I. Putri, "Penerapan State Pattern dan Retrieval-based Chatbot pada NPC Game Visual Novel di Unity Engine," *J. Buana Inform.*, vol. 16, no. 2, pp. 103–113, 2025.

[19] A. Toha, J. L. Putra, and G. Windu, "Penerapan Finite State Automata Pada Vending Machine Cangklong Rokok," *J. Ilmu Komput. dan Bisnis*, vol. 13, no. 2, pp. 23–32, 2022.

[20] A. F. A. A. Kusuma and R. D. Handayani, "Implementation of Inventory Checking System Using Finite State Machine in Role Playing Game," *Int. J. Comput. Inf. Syst.*, vol. 5, no. 4, pp. 280–285, 2024.