P-ISSN : 2580-4316
DOI: https://doi.org/10.37817/ikraith-informatika.v9i3 E-ISSN : 2654-8054

Pengembangan Sistem Deteksi SQL Injection Berbasis Pola Menggunakan Regular Expression dengan Pemindaian Real-Time pada Aplikasi Web

[1]Nandi Pura Nugraha1, [2]Aniq Astofa2
[1]Program Studi Teknik Informatika, Universitas Pamulang, Banten
[2]Program Studi Teknik Informatika, Universitas Pamulang, Banten

E-mail: [1] Nandipura09@gmail.com, [2] aniqastofa03@gmail.com

ABSTRAK

Penelitian ini bertujuan untuk mengembangkan sistem deteksi serangan SQL Injection berbasis web menggunakan metode Regular Expression (RegEx) sebagai pendekatan yang ringan dan ekonomis untuk deteksi awal kerentanan aplikasi. Kebaruan penelitian ini terletak pada perancangan dan implementasi 110 pola RegEx yang diklasifikasikan ke dalam enam kategori serangan, yaitu Error-Based, Union-Based, Boolean-Based, Stacked Query, Time-Based, dan Authentication Bypass, yang diintegrasikan ke dalam sistem pemindaian real-time berbasis web. Sistem dikembangkan menggunakan model Waterfall dengan teknologi FastAPI untuk backend dan React untuk frontend, serta dilengkapi fitur Role-Based Access Control (RBAC) dan dashboard visual untuk monitoring hasil deteksi. Berbeda dengan pendekatan berbasis machine learning yang memerlukan sumber daya komputasi tinggi dan dataset pelatihan, metode yang diusulkan berfokus pada efisiensi dan kemudahan implementasi. Hasil pengujian menunjukkan kinerja sistem yang sangat baik dengan tingkat keberhasilan 100% pada 25 skenario pengujian Black Box, nilai Cyclomatic Complexity sebesar 11 pada pengujian White Box yang menunjukkan sistem berada dalam kategori maintainable, serta tingkat kepuasan pengguna sebesar 4,2 dari skala 5,0. Penelitian ini memberikan kontribusi berupa pendekatan deteksi SQL Injection berbasis pola yang terstruktur, ringan, dan mudah diimplementasikan sehingga dapat dimanfaatkan sebagai sistem deteksi awal (early detection system) untuk meningkatkan keamanan aplikasi web secara mandiri.

Kata kunci : SQL Injection; Regular Expression; Keamanan Web; Deteksi Serangan; FastAPI

ABSTRACT

This article aims to develop a web-based SQL Injection detection system using the Regular Expression (RegEx) method as a lightweight and cost-effective approach for early vulnerability detection, with its novelty lying in the design and implementation of 110 RegEx patterns classified into six categories of SQL Injection attacks—Error-Based, Union-Based, Boolean-Based, Stacked Query, Time-Based, and Authentication Bypass—which are integrated into a real-time web-based scanning system. The system is developed using the Waterfall model, utilizing FastAPI for the backend and React for the frontend, and is enhanced with Role-Based Access Control (RBAC) and a visual dashboard for monitoring detection results. Unlike machine learning-based approaches that require high computational resources and training datasets, the proposed method emphasizes efficiency and ease of implementation. The testing results indicate that the system performs effectively, achieving a 100% success rate in 25 Black Box testing scenarios, while White Box testing produced a Cyclomatic Complexity value of 11, indicating that the system is maintainable, and user evaluation showed an average satisfaction score of 4.2 out of 5.0. This research contributes a structured, lightweight, and practical pattern-based SQL Injection detection approach that can be implemented as an early detection system to independently enhance web application security.

P-ISSN : 2580-4316
E-ISSN : 2654-8054 DOI: https://doi.org/10.37817/ikraith-informatika.v9i3

Keyword : Regular Expression; SQL Injection; Attack Detection; FastAPI; Web Security

1. PENDAHULUAN

Dalam era digital yang terus berkembang, keamanan informasi pada aplikasi berbasis web menjadi aspek yang sangat krusial. Peningkatan jumlah layanan digital berbanding lurus dengan meningkatnya potensi ancaman keamanan, salah satunya adalah serangan SQL Injection (SQLi). SQL Injection merupakan teknik serangan yang dilakukan dengan menyisipkan perintah SQL berbahaya melalui input pengguna untuk memperoleh akses tidak sah terhadap basis data aplikasi. Serangan ini dapat menyebabkan kebocoran data, manipulasi informasi, hingga pengambilalihan sistem secara keseluruhan.

Berdasarkan laporan Open Web Application Security Project (OWASP), kerentanan Injection masih termasuk dalam kategori risiko keamanan tertinggi pada aplikasi web dan secara konsisten masuk dalam daftar OWASP Top 10 Web Application Security Risks. Hal ini menunjukkan bahwa serangan SQL Injection masih menjadi ancaman serius yang perlu ditangani secara sistematis. Di Indonesia, tingkat kerentanan terhadap serangan ini juga masih cukup tinggi, terutama pada aplikasi yang belum menerapkan mekanisme validasi input yang memadai.

Penelitian sebelumnya telah mengusulkan berbagai pendekatan untuk mendeteksi serangan SQL Injection. Pendekatan berbasis machine learning, seperti Convolutional Neural Network (CNN) dan Recurrent Neural Network (RNN), menunjukkan tingkat akurasi yang tinggi dalam mengidentifikasi pola serangan. Namun, pendekatan tersebut memiliki keterbatasan dalam hal kebutuhan dataset yang besar, kompleksitas implementasi, serta konsumsi sumber daya komputasi yang tinggi. Kondisi ini menjadi kendala dalam penerapan pada lingkungan dengan sumber daya terbatas, terutama pada tahap pengembangan aplikasi skala kecil hingga menengah.

Sebagai alternatif, metode Regular Expression (RegEx) dapat digunakan untuk mendeteksi pola serangan SQL Injection melalui pencocokan string berdasarkan pola tertentu. Metode ini memiliki keunggulan dalam hal kesederhanaan, kecepatan proses, serta kemudahan implementasi tanpa memerlukan pelatihan model. Meskipun demikian, penggunaan RegEx dalam deteksi SQL Injection masih memiliki keterbatasan dalam hal generalisasi terhadap pola serangan yang kompleks dan dinamis.

Berdasarkan kajian tersebut, terdapat kesenjangan penelitian (research gap) antara pendekatan berbasis machine learning yang memiliki akurasi tinggi namun kompleks dan membutuhkan sumber daya besar, dengan pendekatan berbasis pola seperti RegEx yang lebih ringan namun belum banyak dikembangkan dalam bentuk sistem deteksi yang terintegrasi, real-time, dan mudah digunakan oleh pengembang. Sebagian besar penelitian sebelumnya masih berfokus pada pengujian metode deteksi tanpa mengintegrasikannya ke dalam sistem yang dapat digunakan secara praktis dalam proses pengembangan perangkat lunak.

Oleh karena itu, penelitian ini mengusulkan pengembangan sistem deteksi SQL Injection berbasis web menggunakan metode Regular Expression yang dirancang sebagai solusi deteksi awal (early detection system) yang ringan, efisien, dan mudah diimplementasikan. Sistem ini dikembangkan agar dapat digunakan secara mandiri oleh pengembang tanpa memerlukan lisensi perangkat lunak berbayar. Selain itu,

DOI: https://doi.org/10.37817/ikraith-informatika.v9i3
P-ISSN : 2580-4316
E-ISSN : 2654-8054

sistem dilengkapi dengan fitur pemindaian real-time, dashboard visual untuk memantau hasil deteksi, Role-Based Access Control (RBAC) untuk pengelolaan hak akses pengguna, serta fitur ekspor hasil deteksi sebagai dokumentasi.

Kebaruan dari penelitian ini terletak pada perancangan dan implementasi sistem deteksi SQL Injection berbasis pola yang mengintegrasikan 110 aturan Regular Expression yang diklasifikasikan ke dalam enam kategori serangan, yaitu Error-Based, Union-Based, Boolean-Based, Stacked Query, Time-Based, dan Authentication Bypass. Selain itu, penelitian ini tidak hanya berfokus pada metode deteksi, tetapi juga pada pengembangan sistem terintegrasi yang mendukung pemindaian real-time, manajemen akses pengguna, serta visualisasi hasil deteksi dalam satu platform berbasis web. Pendekatan ini memberikan kontribusi praktis sekaligus konseptual dalam pengembangan sistem deteksi awal yang efisien, ekonomis, dan mudah diadopsi oleh pengembang.

Dengan demikian, penelitian ini diharapkan dapat memberikan alternatif solusi dalam meningkatkan keamanan aplikasi web secara mandiri serta mendukung peningkatan kesadaran keamanan sejak tahap awal pengembangan perangkat lunak

2. LANDASAN TEORI

2.1. Teori Perancangan Basis Data

Perancangan basis data merupakan fondasi penting dalam pembangunan sistem informasi karena menentukan konsistensi, integritas, dan keberlanjutan pengelolaan data, umumnya menggunakan pendekatan Relational Database Management System (RDBMS) dengan tabel yang terhubung melalui primary key dan foreign key. Dalam sistem deteksi SQL Injection berbasis web, perancangan dilakukan dengan menerapkan prinsip normalisasi untuk menghindari duplikasi data, pemodelan Entity-Relationship (ER) melalui ERD untuk menggambarkan entitas, atribut, dan relasi, serta transformasi ERD ke Logical Record Structure (LRS) guna menghasilkan struktur tabel logis yang siap diimplementasikan, lengkap dengan aturan kunci dan integritas referensial. Integritas data dijaga melalui penerapan entity integrity dan referential integrity agar data tetap akurat dan konsisten. Pada penelitian ini, SQLite dipilih sebagai basis data karena bersifat lightweight, portabel, tidak memerlukan server tambahan, mudah diintegrasikan, serta efisien dalam menangani operasi transaksi dan pencatatan log dalam jumlah besar, sehingga dinilai sesuai untuk mendukung sistem deteksi serangan berbasis web secara cepat, stabil, dan andal.

2.2. Teori Pengujian Sistem

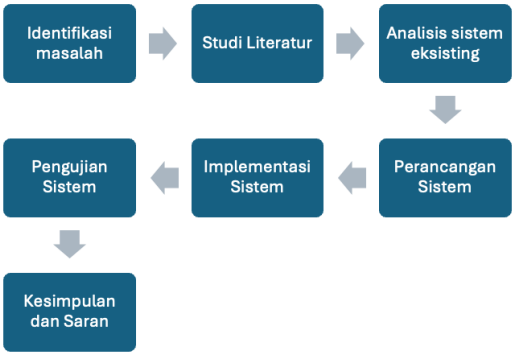
Pengujian sistem merupakan tahap penting dalam pengembangan perangkat lunak untuk memastikan sistem berjalan sesuai spesifikasi serta mendeteksi kesalahan sebelum digunakan pengguna. Secara umum, pengujian dibagi menjadi Black Box Testing yang berfokus pada validasi fungsi berdasarkan input dan output tanpa melihat kode internal, serta White Box Testing yang menitikberatkan pada pemeriksaan struktur logika program melalui teknik seperti statement coverage, branch coverage, dan path coverage. Untuk mengukur kompleksitas logika dan menentukan jumlah jalur uji minimum, digunakan metrik Cyclomatic Complexity (CC) yang menunjukkan tingkat kerumitan dan maintainability kode. Selain pengujian teknis, evaluasi juga dilakukan melalui user response menggunakan kuesioner untuk menilai aspek pengalaman pengguna seperti kemudahan, efisiensi, dan kepuasan. Dalam kerangka pemikiran penelitian ini, masalah utama berupa kerentanan SQL Injection diatasi dengan metode Regular Expression yang diimplementasikan

P-ISSN : 2580-4316
E-ISSN : 2654-8054 DOI: https://doi.org/10.37817/ikraith-informatika.v9i3

dalam sistem berbasis web, kemudian diuji melalui berbagai skenario serangan dan dianalisis untuk menilai efektivitas serta memberikan rekomendasi peningkatan keamanan.

3. METODOLOGI

Penelitian ini menggunakan pendekatan kualitatif dengan metode pengembangan sistem (system development research) yang mengadopsi model Waterfall. Model Waterfall dipilih karena memiliki tahapan yang terstruktur dan sistematis, sehingga sesuai untuk pengembangan sistem keamanan yang membutuhkan perencanaan yang matang dan terdokumentasi dengan baik. Model ini terdiri dari lima tahapan utama, yaitu analisis kebutuhan, desain sistem, implementasi, pengujian, dan pemeliharaan. Kerangka alur penelitian secara keseluruhan ditunjukkan pada Gambar 1.

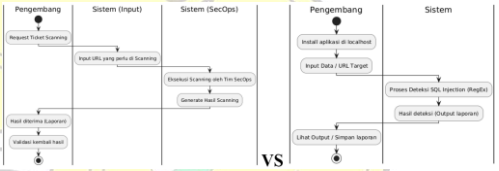


Gambar 1. Kerangka Alur Penelitian

1. Analisis Kebutuhan

Tahap analisis kebutuhan dilakukan untuk mengidentifikasi permasalahan pada sistem yang sedang berjalan serta menentukan kebutuhan sistem yang akan dikembangkan. Pengumpulan data dilakukan melalui metode wawancara dengan Manager Divisi Application Development di PT [Redacted]. Hasil wawancara menunjukkan bahwa sistem vulnerability scanning yang digunakan sebelumnya memiliki beberapa keterbatasan, antara lain biaya lisensi yang tinggi, proses pemindaian yang harus melalui sistem tiket dengan persetujuan berlapis, serta tidak adanya dukungan untuk pemindaian parsial. Kondisi tersebut menyebabkan proses pengujian keamanan menjadi kurang fleksibel dan tidak efisien.

Selain itu, studi literatur juga dilakukan untuk memperoleh landasan teoritis terkait SQL Injection, Regular Expression, serta metode deteksi serangan yang telah dikembangkan sebelumnya. Studi ini menjadi dasar dalam perancangan sistem usulan serta untuk mengidentifikasi kelebihan dan kekurangan dari pendekatan yang digunakan. Perbandingan antara sistem berjalan dan sistem usulan ditunjukkan pada Gambar 2.



Gambar 2. Perbandingan Sistem Berjalan dan Sistem Usulan

2. Desain Sistem

Tahap desain sistem bertujuan untuk merancang arsitektur dan komponen sistem yang akan dikembangkan. Perancangan dilakukan menggunakan pendekatan Unified Modeling Language (UML) yang meliputi Use Case Diagram, Activity Diagram, Sequence Diagram, dan Class Diagram untuk memodelkan interaksi sistem secara menyeluruh.

Perancangan basis data dilakukan menggunakan Entity Relationship Diagram (ERD) dan Logical Record Structure (LRS) untuk memastikan struktur data yang efisien dan terorganisasi. Basis data yang digunakan adalah SQLite, yang dipilih karena sifatnya ringan, portabel, dan sesuai untuk implementasi sistem berbasis lokal.

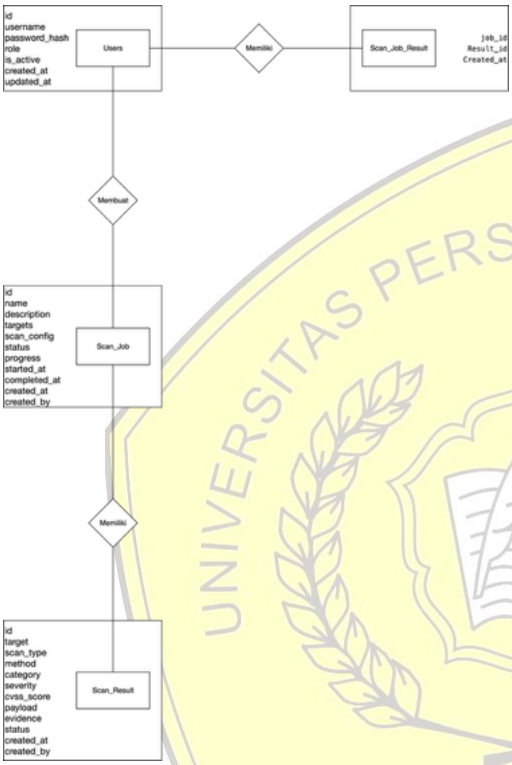
Sistem dirancang dengan empat entitas utama, yaitu:

- Users, untuk menyimpan data pengguna dan hak akses

DOI: https://doi.org/10.37817/ikraith-informatika.v9i3
P-ISSN : 2580-4316
E-ISSN : 2654-8054

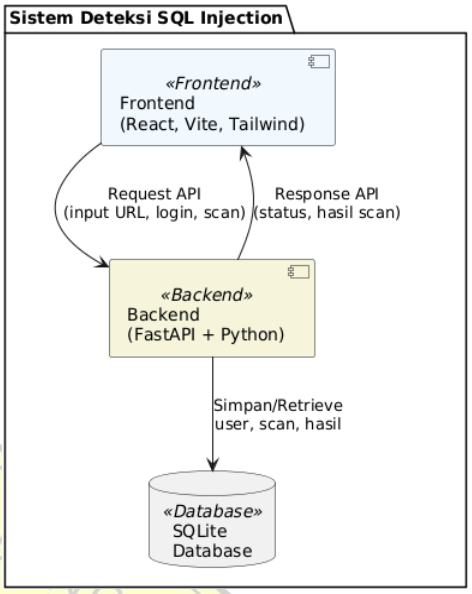
- ScanJobs, untuk mencatat aktivitas pemindaian
- ScanResults, untuk menyimpan hasil deteksi
- ScanJobResults, untuk relasi antara pekerjaan pemindaian dan hasilnya

Relasi antar entitas ditunjukkan pada Gambar 3.



Gambar 3 Entity Relationship Diagram Sistem

Selain itu, arsitektur sistem dirancang menggunakan pendekatan **client-server** dengan pemisahan yang jelas antara komponen frontend, backend, dan database. Komunikasi antar komponen dilakukan menggunakan **RESTful API**, sehingga sistem dapat dikembangkan dan dipelihara secara modular. Arsitektur sistem secara keseluruhan ditunjukkan pada Gambar 4.



Gambar 4. Arsitektur Sistem Deteksi SQL Injection

3. **Perancangan dan Implementasi Metode Regular Expression**

Penelitian ini menggunakan pendekatan eksperimental dalam pengembangan metode deteksi SQL Injection berbasis Regular Expression (RegEx). Proses ini melibatkan empat tahapan utama, yaitu perancangan pola RegEx, implementasi sistem, pengelolaan hak akses, serta pengujian sistem.

3.1 Perancangan Pola Regular Expression

Perancangan pola RegEx dilakukan dengan mengklasifikasikan pola serangan SQL Injection ke dalam enam kategori berdasarkan studi literatur dan dokumentasi Open Web Application Security Project (OWASP). Pendekatan ini bertujuan untuk meningkatkan cakupan deteksi terhadap berbagai jenis serangan SQL Injection. Enam kategori pola yang digunakan meliputi :

1. Error-Based Injection
   Digunakan untuk mendeteksi pesan kesalahan dari sistem basis sdata, seperti "SQL syntax error" atau kode kesalahan spesifik database.
2. Union-Based Injection
   Digunakan untuk mendeteksi

P-ISSN : 2580-4316
E-ISSN : 2654-8054 DOI: https://doi.org/10.37817/ikraith-informatika.v9i3

penggunaan klausa *UNION SELECT* yang sering digunakan untuk menggabungkan hasil query ilegal.

3. Boolean-Based Injection
   Digunakan untuk mendeteksi ekspresi logika yang selalu bernilai benar, seperti OR 1=1.
4. Stacked Query Injection
   Digunakan untuk mendeteksi perintah SQL berantai yang dipisahkan dengan tanda titik koma (;).
5. Time-Based Injection
   Digunakan untuk mendeteksi penggunaan fungsi penundaan seperti sleep() atau waitfor delay.
6. Authentication Bypass
   Digunakan untuk mendeteksi upaya melewati autentikasi dengan memanfaatkan kondisi logika tertentu.

Secara keseluruhan, terdapat 110 pola Regular Expression yang digunakan dalam penelitian ini. Klasifikasi lengkap pola RegEx ditunjukkan pada Tabel 1.

Tabel 1. Klasifikasi Kategori Pola Regular Expression

No	Kategori	Contoh Pola RegEx	Jumlah Pola	Deskripsi
1	Error-Based	`SQL syntax.*MySQL, ORA-[0-9]{4}`	43	Mendeteksi pesan error database
2	Union-Based	`union\s+select, order\s+by\s+[0-9]+`	16	Mendeteksi penggunaan UNION SELECT
3	Boolean-Based	`or\s+1\s*=\s*1, and\s+'1'\s*=\s*'1'`	15	Mendeteksi logika Boolean
4	Stacked Query	`;\s*(select`	insert	update
5	Time-Based	`sleep\s*\([0-9]+\), waitfor\s+delay`	10	Mendeteksi fungsi delay
6	Auth Bypass	`' OR '1'='1, admin'--`	18	Mendeteksi bypass autentikasi
	Total		**110**	

Penggunaan pendekatan berbasis pola ini dipilih karena memiliki keunggulan dalam hal kecepatan proses, kemudahan implementasi, serta tidak memerlukan pelatihan model seperti pada metode berbasis machine learning. Dengan demikian, metode ini lebih sesuai ntuk lingkungan dengan keterbatasan sumber daya.

3.2 Implementasi Sistem

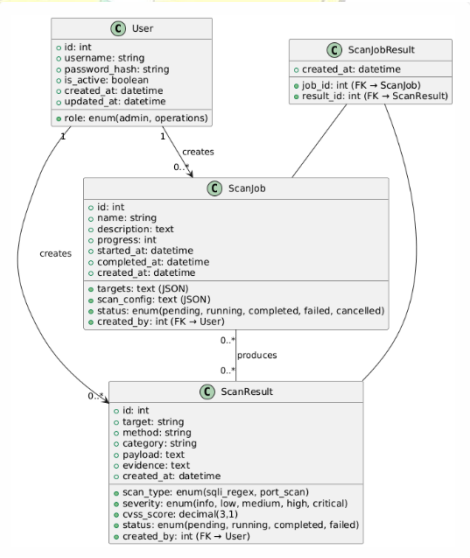
Implementasi sistem dilakukan dengan menggunakan teknologi modern berbasis web. Backend sistem dikembangkan menggunakan FastAPI dengan bahasa pemrograman Python versi 3.11, sedangkan frontend dikembangkan menggunakan ReactJS 18 dengan TypeScript.

Basis data menggunakan SQLite untuk mendukung penyimpanan data secara lokal dengan performa yang ringan. Antarmuka pengguna dirancang menggunakan **Tailwind CSS** untuk menghasilkan tampilan yang responsif dan mendukung berbagai perangkat, termasuk fitur **dark mode** untuk meningkatkan kenyamanan pengguna.

Sistem juga dilengkapi dengan mekanisme **Role-Based Access Control (RBAC)** untuk mengatur hak akses pengguna. Terdapat dua jenis peran dalam sistem, yaitu:

- **Admin**, yang memiliki akses penuh untuk mengelola pengguna dan hasil pemindaian
- **Operational**, yang hanya dapat melakukan pemindaian dan melihat hasilnya sendiri

Struktur kelas sistem ditunjukkan pada



Gambar 5 Class Diagram Sistem

DOI: https://doi.org/10.37817/ikraith-informatika.v9i3
P-ISSN : 2580-4316
E-ISSN : 2654-8054

4. Pengujian Sistem

Pengujian sistem dilakukan untuk memastikan bahwa sistem yang dikembangkan berfungsi sesuai dengan kebutuhan serta memiliki kualitas yang baik. Pengujian dilakukan menggunakan tiga pendekatan utama, yaitu:

1. Black Box Testing

   Pengujian ini dilakukan untuk mengevaluasi fungsionalitas sistem tanpa melihat struktur kode internal. Sebanyak 25 skenario pengujian digunakan untuk memastikan setiap fitur berjalan sesuai dengan spesifikasi.
2. White Box Testing

   Pengujian ini dilakukan untuk mengevaluasi logika internal sistem. Parameter yang digunakan meliputi:
   - Statement Coverage
   - Branch Coverage
   - Path Coverage
   - Cyclomatic Complexity

   Pengujian ini bertujuan untuk memastikan bahwa seluruh jalur logika dalam sistem telah diuji secara menyeluruh.
3. User Response Testing

   Pengujian ini dilakukan untuk mengevaluasi pengalaman pengguna terhadap sistem. Metode yang digunakan adalah kuesioner dengan skala Likert (1–5) yang melibatkan tiga responden dengan peran berbeda, yaitu Manager, Team Lead, dan Programmer. Hasil pengujian ini digunakan untuk mengukur tingkat kepuasan pengguna terhadap sistem yang dikembangkan.

5. Lingkungan Pengembangan

Pengembangan dan pengujian sistem dilakukan menggunakan perangkat dengan spesifikasi sebagai berikut:

- Perangkat: MacBook Air M1 (2020)
- Prosesor: Apple M1 8-core CPU
- RAM: 8 GB
- Penyimpanan: 256 GB SSD
- Sistem Operasi: macOS Sequoia 15.5

Perangkat lunak yang digunakan meliputi Visual Studio Code sebagai editor kode serta browser berbasis Mozilla Firefox untuk pengujian antarmuka dan fungsionalitas sistem.

5. HASIL DAN PEMBAHASAN

Hasil dan pembahasan berisi hasil analisis fenomena di wilayah penelitian yang relevan dengan tema kajian. Hasil penelitian hendaknya dibandingkan dengan teori dan temuan penelitian yang relevan)

Algoritmaatau Program

Algoritmaatau program dianggap sebagai gambar, tetapi dituliskan menggunakan font yang tidak proporsional—lebarsemua font sama, lebar sama dengan lebar m atau w —dan mempunyai kaki (serif), sehingga dapat dibedakan antara I (ibesar) dan l (l kecil), misalnya Courier New dengan besar huruf

P-ISSN : 2580-4316
E-ISSN : 2654-8054 DOI: https://doi.org/10.37817/ikraith-informatika.v9i3

maksimal 10 point. Contoh algoritma dapat dilihat dalam Gambar 2.

```
1. Baca file 'Mohon dibaca dulu.dot'
2. Ikuti petunjuk di dalamnya.
3. Buat makalah dengan mengedit
   file 'Template TRANSIT.dot' dan
   simpan sebagai file '*.doc'.
4. Jika Anda mengikuti petunjuk,
   maka jalankan langkah 5.
5. Tulis 'Editing lebih mudah'
6. Jika tidak menggunakan template,
   maka jalankan langkah 7 sampai
   langkah 11.
7. Tulis 'Perhatian!!!! '
8. Tulis 'Komputer mahal ini'
9. Tulis 'sedang berfungsi'
10.Tulis 'sebagai mesin ketik manual'
11.Tulis 'ha ha ha ha'
```

Gambar 1.Algoritma Penulisan Makalah TRANSIT

6. KESIMPULAN

Implementasi sistem deteksi SQL Injection berbasis web pada penelitian ini menghasilkan sebuah aplikasi bernama SecureScan yang dirancang untuk mendeteksi pola serangan SQL Injection menggunakan metode Regular Expression (RegEx). Sistem dikembangkan menggunakan arsitektur client-server dengan FastAPI sebagai backend dan ReactJS sebagai frontend, sehingga mampu memberikan performa yang responsif serta antarmuka yang interaktif.

Secara umum, sistem terdiri dari enam modul utama, yaitu Login, Dashboard, New Scan, Scan Result, Scan History, dan User Management. Pengembangan modul-modul tersebut bertujuan untuk mendukung proses deteksi secara menyeluruh, mulai dari autentikasi pengguna hingga pengelolaan hasil pemindaian.

1. Implementasi antarmuka dirancang dengan prinsip usability dan responsivitas, sehingga dapat digunakan oleh pengguna teknis maupun non-teknis. Setiap halaman memiliki fungsi spesifik yang saling terintegrasi dalam mendukung proses deteksi serangan SQL Injection. Halaman login menyediakan mekanisme autentikasi berbasis username dan password dengan validasi secara real-time. Sistem memberikan umpan balik langsung apabila terjadi kesalahan autentikasi, sehingga meningkatkan pengalaman pengguna serta mengurangi kemungkinan kesalahan input. Dashboard berfungsi sebagai pusat informasi yang menampilkan ringkasan hasil pemindaian, meliputi jumlah total scan, jumlah temuan kritis, serta distribusi tingkat keparahan. Visualisasi data ditampilkan dalam bentuk grafik dan card statistik untuk mempermudah interpretasi hasil oleh pengguna. Pendekatan visual ini membantu pengguna dalam memahami kondisi keamanan sistem secara cepat dan intuitif.

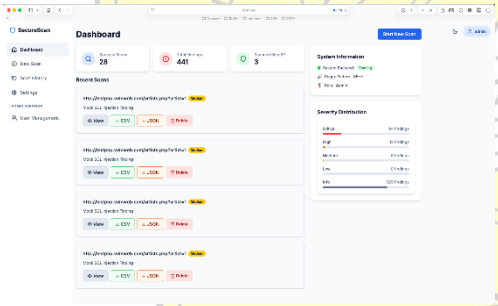
Fitur New Scan memungkinkan pengguna untuk memasukkan target berupa URL atau alamat IP yang akan dipindai. Proses validasi dilakukan secara otomatis untuk memastikan format input sesuai. Selama proses pemindaian berlangsung, sistem menampilkan log aktivitas secara real-time, sehingga pengguna dapat memantau proses deteksi secara transparan. Pada halaman hasil pemindaian, sistem menampilkan informasi detail terkait temuan yang terdeteksi. Informasi tersebut meliputi kategori serangan, metode deteksi, tingkat keparahan (severity level), skor CVSS, payload yang teridentifikasi, serta bukti (evidence) berupa respons dari server. Sebagai contoh, sistem berhasil mendeteksi pola serangan Union-Based SQL Injection dengan tingkat keparahan tinggi melalui pencocokan pola RegEx seperti union\s+select. Hal ini menunjukkan bahwa metode Regular Expression mampu mengenali pola serangan yang umum digunakan dalam eksploitasi basis data. Sebaliknya, pada input yang tidak mengandung pola berbahaya, sistem memberikan status “Safe” dengan skor CVSS 0.0. Hasil ini menunjukkan bahwa sistem tidak hanya mampu mendeteksi serangan, tetapi juga

DOI: https://doi.org/10.37817/ikraith-informatika.v9i3
P-ISSN : 2580-4316
E-ISSN : 2654-8054

dapat membedakan input normal secara akurat, sehingga meminimalkan terjadinya false positive.

Halaman Scan History menyediakan riwayat hasil pemindaian dalam bentuk tabel yang dilengkapi fitur pencarian dan penyaringan berdasarkan target, jenis serangan, dan tingkat keparahan. Implementasi ini memudahkan pengguna dalam melakukan analisis historis terhadap hasil deteksi.

Fitur User Management dirancang untuk mendukung pengelolaan pengguna dengan menerapkan konsep Role-Based Access Control (RBAC). Pengguna dengan peran Admin memiliki akses penuh terhadap seluruh data, sedangkan pengguna Operational hanya dapat mengakses data miliknya sendiri. Implementasi RBAC ini meningkatkan keamanan sistem dengan membatasi akses sesuai peran pengguna.



Gambar 6. Halaman Dashboard

2. Pengujian Black Box Testing

Pengujian sistem dilakukan menggunakan metode Black Box Testing untuk mengevaluasi fungsionalitas tanpa melihat struktur internal kode. Metode ini dipilih karena mampu menguji kesesuaian output sistem terhadap kebutuhan pengguna.

Berdasarkan hasil pengujian pada 25 skenario yang mencakup seluruh modul utama, sistem menunjukkan tingkat keberhasilan sebesar 100% sebagaimana ditampilkan pada Tabel 2. Hal ini menunjukkan bahwa seluruh fitur, termasuk autentikasi, pemindaian, pengelolaan pengguna, dan kontrol akses, berjalan sesuai dengan spesifikasi yang dirancang.

Namun demikian, pengujian ini hanya berfokus pada aspek fungsionalitas. Evaluasi lebih lanjut seperti pengujian performa dan pengujian penetrasi (penetration testing) diperlukan untuk mengukur ketahanan sistem terhadap serangan yang lebih kompleks.

Tabel 2 Ringkasan Hasil Pengujian Black Box Testing

No	Modul	Jumlah Skenario	Hasil PASS	Persentase
1	Auth	4	4	100%
2	Dashboard	2	2	100%
3	Scanner	4	4	100%
4	Scan History	4	4	100%
5	Export	3	3	100%
6	User Management	4	4	100%
7	Security	2	2	100%
	Total	**25**	**25**	**100 %**

Hasil pengujian menunjukkan bahwa seluruh fungsi sistem berjalan dengan baik dengan tingkat keberhasilan mencapai 100%. Hal ini menunjukkan bahwa sistem telah memenuhi kebutuhan fungsional yang telah ditentukan pada tahap perancangan.

3. Analisis Hasil Deteksi

Hasil implementasi menunjukkan bahwa sistem mampu mendeteksi serangan SQL Injection berbasis pola, khususnya jenis Union-Based SQL Injection. Penggunaan metode Regular Expression memungkinkan sistem mengenali pola serangan klasik dengan tingkat akurasi yang tinggi.

Selain itu, sistem juga mampu:

- Membedakan input aman dan berbahaya secara konsisten

P-ISSN : 2580-4316
E-ISSN : 2654-8054 DOI: https://doi.org/10.37817/ikraith-informatika.v9i3

- Menghasilkan skor risiko menggunakan standar CVSS
- Menyediakan evidence untuk validasi hasil deteksi

Namun, pendekatan berbasis pola memiliki keterbatasan, yaitu kurang efektif dalam mendeteksi serangan SQL Injection yang telah mengalami obfuscation atau encoding kompleks. Hal ini sejalan dengan penelitian oleh Halfond et al. (2006) yang menyatakan bahwa metode signature-based detection memiliki keterbatasan terhadap serangan yang dimodifikasi.

4. **Perbandingan dengan Penelitian Terdahulu**

Untuk menunjukkan kontribusi penelitian, dilakukan perbandingan dengan beberapa penelitian internasional terkait deteksi SQL Injection.

Tabel 3. State of the Art Penelitian SQL Injection Detection

No	Peneliti & Tahun	Metode	Kelebihan	Kekurangan
1	Halfond et al. (2006)	AMNESIA (Static + Dynamic Analysis)	Akurasi tinggi, analisis komprehensif	Kompleks dan berat
2	Shar & Tan (2013)	Machine Learning	Adaptif terhadap pola baru	Membutuhkan dataset besar
3	Gupta & Sharma (2015)	Pattern Matching	Cepat dan ringan	Rentan bypass
4	Singh et al. (2019)	Hybrid Approach	Deteksi lebih luas	Implementasi kompleks
5	Penelitian ini (2025)	Regex-based Detection + Web System	Real-time, ringan, mudah diimplementasikan	Terbatas pada pola tertentu

5. Pembahasan dan Kontribusi Penelitian

Penelitian ini memberikan kontribusi dalam pengembangan sistem deteksi serangan SQL Injection berbasis web yang mengintegrasikan pendekatan ringan berbasis pola dengan arsitektur aplikasi modern. Sistem yang dikembangkan memiliki beberapa keunggulan utama, yaitu:

1. Mampu melakukan pemindaian secara real-time terhadap input pengguna
2. Mengintegrasikan hasil deteksi ke dalam dashboard visual interaktif
3. Mengimplementasikan mekanisme Role-Based Access Control (RBAC) untuk pengelolaan hak akses
4. Menggunakan standar Common Vulnerability Scoring System (CVSS) dalam mengukur tingkat risiko

Pendekatan berbasis Regular Expression yang digunakan dalam penelitian ini terbukti efektif dalam mendeteksi serangan SQL Injection klasik, khususnya yang berbasis pola seperti Union-Based SQL Injection. Hal ini sejalan dengan penelitian Supartini (2023) yang menyatakan bahwa metode pattern matching memiliki performa yang baik dalam mendeteksi serangan berbasis signature.

Dibandingkan dengan pendekatan lain seperti machine learning dan deep learning, metode yang digunakan dalam penelitian ini memiliki keunggulan dari sisi efisiensi komputasi dan kemudahan implementasi. Penelitian oleh Xie et al. (2019) menunjukkan bahwa metode berbasis CNN mampu meningkatkan akurasi deteksi, namun membutuhkan sumber daya komputasi yang lebih besar dan dataset pelatihan yang kompleks. Dengan demikian, pendekatan dalam penelitian ini lebih sesuai untuk implementasi pada sistem dengan keterbatasan sumber daya.

Namun demikian, metode berbasis Regular Expression memiliki keterbatasan dalam mendeteksi serangan SQL Injection yang telah mengalami obfuscation, encoding, atau modifikasi payload yang kompleks. Hal ini juga didukung oleh penelitian Alghawazi et al. (2023) yang menunjukkan bahwa pendekatan deep learning lebih adaptif terhadap variasi serangan modern.

DOI: https://doi.org/10.37817/ikraith-informatika.v9i3
P-ISSN : 2580-4316
E-ISSN : 2654-8054

Oleh karena itu, pengembangan selanjutnya dapat difokuskan pada:

1. Integrasi metode machine learning untuk meningkatkan kemampuan adaptasi system
2. Penambahan deteksi berbasis anomaly detection
3. Pengujian menggunakan dataset serangan yang lebih kompleks dan real-world
4. Integrasi dengan sistem DevSecOps atau pipeline CI/CD
5. Dengan pengembangan tersebut, sistem diharapkan mampu memberikan deteksi yang lebih komprehensif terhadap berbagai jenis serangan keamanan aplikasi web

6. KESIMPULAN

Penelitian ini berhasil mengembangkan sistem deteksi serangan SQL Injection berbasis web menggunakan metode Regular Expression yang terbukti efektif, efisien, dan ekonomis. Sistem yang dikembangkan mampu mendeteksi berbagai pola serangan SQL Injection secara real-time dengan tingkat keberhasilan yang tinggi berdasarkan hasil pengujian yang dilakukan.

Hasil pengujian Black Box menunjukkan tingkat keberhasilan sebesar 100% pada 25 skenario pengujian, yang mengindikasikan bahwa seluruh fungsi sistem berjalan sesuai dengan spesifikasi yang dirancang. Selain itu, pengujian White Box menghasilkan nilai Cyclomatic Complexity sebesar 11, yang menunjukkan bahwa sistem berada pada tingkat kompleksitas sedang dan relatif mudah untuk dipelihara dan dikembangkan lebih lanjut.

Dari sisi pengguna, sistem juga menunjukkan tingkat penerimaan yang baik dengan rata-rata skor kepuasan sebesar 4,2 dari skala 5,0. Hal ini menunjukkan bahwa sistem tidak hanya berfungsi dengan baik secara teknis, tetapi juga memiliki usability yang memadai.

Hasil penelitian ini menunjukkan bahwa metode Regular Expression dapat digunakan sebagai pendekatan deteksi awal (early detection) yang praktis tanpa memerlukan sumber daya komputasi yang besar. Dibandingkan dengan tools komersial, sistem yang dikembangkan memiliki keunggulan dari sisi biaya, fleksibilitas, dan kemudahan implementasi, sehingga dapat menjadi solusi alternatif bagi pengembang aplikasi web.

Namun demikian, metode ini masih memiliki keterbatasan dalam mendeteksi serangan yang bersifat kompleks dan adaptif. Oleh karena itu, penelitian selanjutnya disarankan untuk mengintegrasikan metode Regular Expression dengan pendekatan lain, seperti machine learning atau analisis perilaku, guna meningkatkan akurasi dan cakupan deteksi.

Selain itu, pengembangan sistem dapat diperluas dengan menambahkan deteksi terhadap jenis serangan lain seperti Cross-Site Scripting (XSS) dan Cross-Site Request Forgery (CSRF), serta integrasi dengan pipeline CI/CD untuk mendukung pengujian keamanan secara otomatis. Dengan demikian, sistem yang dikembangkan dapat menjadi solusi keamanan aplikasi web yang lebih komprehensif dan adaptif terhadap perkembangan ancaman siber.

DAFTAR PUSTAKA

Achyani, & Saumi. (2019). Penerapan metode waterfall dalam pengembangan perangkat lunak. *Jurnal Sistem Informasi*, *6*(1), 89–96.

Al Azhar, & Harwahyu, R. (2023). Analisis keamanan informasi bagi pengguna website menggunakan Kalilinux melalui teknik SQL injection. *Multitek Indonesia, 25*(1), 65–72.

Alghawazi, M., Alghazzawi, D., & Alarifi, S. (2023). Deep learning architecture for detecting SQL injection attacks based on RNN autoencoder model. *Mathematics, 11*(15), 3286.

Ananda, Prasetyo, R., & Wijaya, T. (2024). Pemanfaatan Visual Studio Code dalam pembelajaran pemrograman web. *Jurnal Pendidikan Teknologi Informasi, 8*(2), 145–153.

Anggraini, D., Putra, A., & Sari, M. (2022). Pengujian black box untuk validasi fungsionalitas perangkat lunak. *Jurnal Teknologi Informasi dan Ilmu Komputer (JTIIK), 9*(4), 789–796.

Behl, A., & Behl, K. (2017). *Cybersecurity and cyberwar: What everyone needs to know*. Oxford University Press.

DQLab. (2022). *SQL injection: Pengertian, cara kerja, dan pencegahan*. https://dqlab.id/sql-injection

Fernandes, G. R., & Lina, I. M. (2024). Website penetration testing with SQL injection technique using SQLMAP on Termux. *Jurnal E-Komtek (Elektro-Komputer-Teknik), 8*(2), 286–293.

Gupta, S., & Sharma, B. B. (2015). SQL injection attack detection using pattern matching techniques. *International Journal of Computer Science and Information Technologies (IJCSIT)*.

Halfond, W. G. J., Viegas, J., & Orso, A. (2006). A classification of SQL injection attacks and countermeasures. *IEEE*.

Handrianto, & Sanjaya. (2020). Pengembangan sistem informasi menggunakan model waterfall. *Jurnal Teknologi Informasi, 8*(2), 156–163.

Hardiani, Nugroho, S., & Pratama, A. (2022). Website sebagai media publikasi dan penyimpanan data dalam era digital. *Jurnal Ilmu Komputer, 9*(3), 234–242.

Kurniawan, N. D., Firdaus, A. M., Abriansah, F. R., & Ferdian, P. R. (2025). Analisis kerentanan SQL injection menggunakan SQLMAP pada Kali Linux. *STRING (Satuan Tulisan Riset dan Inovasi Teknologi), 10*(1), 45–54.

Mardiana, I., Wahyudin, & Junaeti, E. (2024). Pengembangan learning management system dengan framework Laravel dan Tailwind CSS. *Jurnal Ilmu Komputer, 10*(1), 67–78.

Nugraha, F. A., & Susetyo, Y. A. (2023). Analisis perbandingan performa database DuckDB dan SQLite pada pengolahan big data. *JIPI (Jurnal Ilmiah Penelitian dan Pembelajaran Informatika), 8*(3), 1052–1060.

Open Web Application Security Project. (2021). *OWASP top 10: Injection*. https://owasp.org/www-project-top-ten/

Putra, & Nugroho. (2023). White box testing dalam pengujian struktur internal kode program. *Jurnal RESTI (Rekayasa Sistem dan Teknologi Informasi), 7*(3), 567–574.

Rahmadani, & Saputra. (2021). Perancangan basis data sebagai fondasi sistem informasi yang andal. *Jurnal RESTI (Rekayasa Sistem dan Teknologi Informasi), 5*(2), 301–308.

Raina, Utama, & Suakanto. (2024). Pengembangan React JS pada frontend website pengaduan dan pelayanan publik menggunakan metode Scrum. *Jurnal Teknologi Informasi, 12*(3), 234–245.

Riandhanu, A. (2022). Peran website dalam perkembangan teknologi informasi. *Jurnal Informatika, 7*(4), 178–185.

Rocha, A., Alves, R., & Pedrosa, T. (2023). Query log analysis for

SQL injection detection. In *Proceedings of the International Conference on Information Systems Security and Privacy* (pp. 471–476).

Sari, N. C., Solichan, A., Ansor, B., Ramdani, A. P., Al Amin, M. Z., Khaira, M., & Al Ubaidah, A. R. M. (2024). Deteksi kerentanan SQL injection pada website menggunakan vulnerability assessment. *JODI (Journal of Digital Informatics), 2*(1), 9–17.

Scarfone, K., & Mell, P. (2018). *Guide to intrusion detection and prevention systems (IDPS)*. National Institute of Standards and Technology (NIST).

Shar, L. K., & Tan, H. B. K. (2013). Defeating SQL injection. *Computer Journal*.

Singh, J., Kumar, S., & Singh, P. (2019). Hybrid approach for SQL injection detection. *IEEE Access*.

Supriyanto, B., Sumijan, & Yuhandri. (2024). Audit keamanan website menggunakan Acunetix web vulnerability (studi kasus di SMK Muhammadiyah 3 Terpadu Pekanbaru). *Jurnal CoSciTech (Computer Science and Information Technology), 5*(1), 134–143.

Supartini, R. (2023). Deteksi serangan SQL injection pada website dengan menggunakan metode regular expression. *Progressive Information, Security, Computer, and Embedded System (PISCES), 1*(2), 45–52.

Xie, X., Ren, C., Fu, Y., Xu, J., & Guo, J. (2019). SQL injection detection for web applications based on elastic-pooling CNN. *IEEE Access, 7*, 151475–151481.