PENGUJIAN WHITE BOX DENGAN METODE BRANCH COVERAGE PADA FITUR AUTENTIKASI DAN VALIDASI OTP DALAM SISTEM INFORMASI PENGOLAHAN DATA TERITORIAL BERBASIS WEB UNTUK OPERASI MILITER

Yohanes Dwi Cahyono [1], Linda Wahyu Widianti [2], Sulvi Indah Pramuningsih [3]
[1] Politeknik Angkatan Darat
[2] STMIK Jakarta STI&K
[3] Politeknik Angkatan Darat
yohanes@poltekad.ac.id

ABSTRAK

Kemajuan teknologi informasi saat ini sangat pesat dan kompleks. Dalam konteks militer, sistem informasi berbasis web adalah alat bantu yang tidak hanya mengoptimalkan pekerjaan namun juga diperlukan untuk menjaga ketepatan, keamanan, dan integritas informasi yang ada. Fitur autentikasi dan validasi One-Time Password (OTP) berperan krusial untuk memastikan akses yang aman dan terkontrol. Namun, dalam implementasinya, sering kali terdapat tantangan dalam menjaga kualitas dan keandalan fitur tersebut, sehingga dibutuhkan pengujian menyeluruh, karena risiko bug dan celah keamanan dapat menghambat performa sistem. Penelitian ini bertujuan untuk menguji fitur autentikasi dan validasi OTP pada sistem informasi pengolahan data teritorial berbasis web. Fokus utamanya adalah memastikan kualitas dan keandalan sistem serta memberikan rekomendasi perbaikan yang diperlukan. Metode white box testing dengan teknik branch coverage digunakan dalam pengujian ini, yang mencakup analisis kode, pembuatan test case, dan evaluasi hasil pengujian menggunakan PHPUnit untuk memastikan setiap cabang logika dalam kode telah diuji dengan baik. Hasil pengujian menunjukkan bahwa sistem mampu menangani berbagai skenario tanpa adanya bug kritis, namun demikian terdapat beberapa rekomendasi perbaikan untuk meningkatkan keandalan sistem. Penelitian ini menegaskan pentingnya penerapan pengujian white box dalam menjamin kualitas sistem informasi, terutama dalam konteks militer yang sangat bergantung pada akurasi dan keamanan data, serta merekomendasikan pengembangan lebih lanjut untuk meningkatkan fungsionalitas dan keamanan sistem.

Kata Kunci: *White-box testing, branch coverage, sistem informasi, operasi militer, OTP*

1. PENDAHULUAN

Dalam era digital saat ini, sistem informasi memainkan peran yang sangat penting dalam mendukung efektivitas operasi militer. Pengelolaan data yang efisien dan akurat menjadi krusial untuk pengambilan keputusan yang tepat dalam situasi yang dinamis dan kompleks [1].

Oleh karena itu, pengembangan sistem informasi pengolahan data teritorial berbasis web dirancang untuk memenuhi kebutuhan operasional personel TNI AD, dengan fokus pada keamanan dan keandalan sistem [2].

Salah satu tantangan utama dalam pengembangan sistem informasi adalah memastikan bahwa sistem tersebut bebas dari kerentanan dan bug yang dapat mengganggu operasional. Pengujian yang cermat dan sistematis, seperti white box testing, menjadi sangat penting untuk menjamin integritas dan keandalan sistem informasi dalam lingkungan yang kritis [3].

Teknik branch coverage dalam white box testing memungkinkan pengujian menyeluruh terhadap setiap cabang logika dalam kode, sehingga dapat mendeteksi potensi masalah sebelum sistem diimplementasikan [4].

Lebih lanjut, penerapan fitur autentikasi yang kuat, seperti One-Time Password (OTP), menjadi salah satu langkah penting dalam meningkatkan keamanan sistem informasi. OTP memberikan lapisan tambahan perlindungan terhadap akses tidak sah, yang sangat penting dalam konteks operasi militer yang memerlukan keamanan data yang tinggi [5].

Penelitian ini bertujuan untuk mengevaluasi efektivitas fitur autentikasi dan validasi OTP dalam sistem informasi pengolahan data teritorial berbasis web, serta memberikan rekomendasi untuk perbaikan yang dapat meningkatkan keandalan sistem.

2. TINJAUAN PUSTAKA

2.1. *Branch Coverage*

Branch coverage adalah metrik dalam white box testing yang mengukur persentase cabang logika yang telah diuji. Teknik ini memastikan bahwa setiap cabang “if”, “else”, “switch”, dan kondisi logika lainnya telah dijalankan untuk mendeteksi potensi bug [6].

2.2. Sistem Informasi berbasis OTP

OTP adalah metode autentikasi yang menghasilkan kata sandi sekali pakai untuk meningkatkan keamanan akses. Berbagai penelitian sebelumnya menunjukkan bahwa keberhasilan implementasi OTP sangat bergantung pada pengujian yang menyeluruh untuk menghindari kelemahan pada logika dan jalur kode [7].

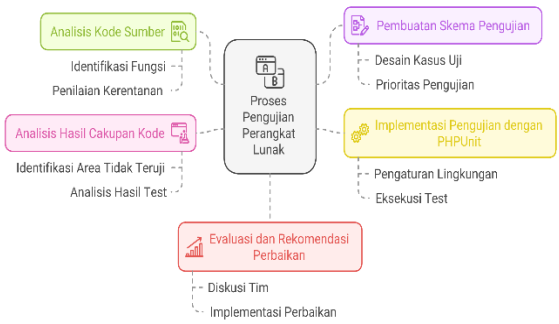
2.3. *Whitebox Testing*

Penggunaan metode white-box testing dalam pengujian web telah menjadi pendekatan yang sangat penting dalam industri pengembangan perangkat lunak. Dengan memberikan fokus pada evaluasi struktur internal aplikasi dan logika pemrograman, pengujian white-box memastikan bahwa setiap jalur kode dan skenario logika telah diuji secara menyeluruh untuk mengidentifikasi potensi kesalahan pada tahap awal pengembangan. Dengan memahami prinsip-prinsip dan praktik-praktik pengujian white-box, pengembang dapat meningkatkan kualitas aplikasi web mereka dengan memastikan bahwa kode program tidak hanya berfungsi dengan benar tetapi juga memenuhi standar keamanan dan efisiensi yang tinggi [8].

3. METODE PENELITIAN

Metodologi penelitian ini mencakup analisis kode sumber fitur autentikasi dan validasi OTP pada sistem berbasis web dengan menggunakan framework Laravel dan library Pragmarx/Google2FA untuk OTP [9]. Penelitian dimulai dengan mengidentifikasi titik-titik cabang dalam kode sumber untuk memahami struktur logika. Selanjutnya, skema pengujian dirancang dengan membuat test case berdasarkan skenario dari setiap cabang logika. Implementasi pengujian dilakukan menggunakan PHPUnit untuk menjalankan test case, sedangkan analisis cakupan kode menggunakan alat seperti Xdebug dan PHPUnit Code Coverage untuk mengukur persentase branch coverage.

Hasil pengujian dianalisis untuk mengidentifikasi cabang yang belum diuji, diikuti dengan evaluasi dan rekomendasi untuk perbaikan kode guna meningkatkan keamanan dan keandalan sistem. Langkah-langkah penelitian yang dilakukan seperti terlihat pada gambar.



Gambar 1. Langkah Penelitian

Kode sumber dari fitur autentikasi dan validasi OTP, pada tahap ini kode sumber dianalisis untuk mengidentifikasi semua titik cabang logika yang ada. hal ini bertujuan untuk memahami jalur eksekusi yang perlu diuji. Tahap berikutnya adalah pembuatan skema pengujian berdasarkan hasil analisis kode, skenario pengujian dirancang dalam bentuk test case yang mencakup setiap cabang logika. Setiap test case difokuskan pada satu jalur logika tertentu. Setelah itu dilakukan implementasi pengujian dengan PHPUnit berupa test case yang telah dirancang kemudian diimplementasikan menggunakan PHPUnit. Alat ini membantu menjalankan pengujian secara otomatis pada kode sumber. Hasil dari pengujian dianalisis menggunakan tool PHPUnit Code Coverage untuk mengukur persentase branch coverage. Langkah ini bertujuan untuk mengetahui cabang logika mana saja yang belum diuji. Pada tahap akhir dilakukan evaluasi dan rekomendasi perbaikan untuk meningkatkan cakupan pengujian dan mengatasi kelemahan pada kode yang teridentifikasi.

4. HASIL DAN PEMBAHASAN

4.1. Deskripsi Kode

Kode autentikasi yang dianalisis berada dalam kelas AuthenticationController, pustaka pragmarx / google2fa digunakan untuk menghasilkan kunci rahasia dan memverifikasi OTP. Berikut adalah fungsi utama yang terdapat dalam kode ini:

4.2. GenerateKey(Request $request)

Fungsi GenerateKey untuk menghasilkan kunci rahasia (secret key) untuk autentikasi dua faktor (2FA) yang digunakan dalam meningkatkan keamanan akun pengguna. Kunci ini dihasilkan secara acak selanjutnya diubah menjadi QR Code yang dapat dipindai oleh aplikasi autentikator yaitu Google Authenticator [10]. Setelah pemindaian, aplikasi tersebut akan menghasilkan kode verifikasi yang berubah secara periodik, memberikan lapisan perlindungan tambahan bagi akun pengguna.

4.3. ViewInsertOtp(Request $request)

ViewInsertOtp bertujuan untuk menampilkan halaman untuk memasukkan kode OTP (One-Time Password) yang diterima melalui aplikasi autentikator. Halaman ini memberikan antarmuka yang memungkinkan pengguna untuk memasukkan kode verifikasi yang bersifat sementara, yang digunakan untuk memverifikasi identitas mereka sebagai bagian dari proses autentikasi dua faktor (2FA) [5]. Setelah kode OTP yang valid dimasukkan, pengguna dapat melanjutkan ke proses berikutnya dengan aman.

4.4. CompleteRegistration(Request request)

CompleteRegistration bertujuan untuk memverifikasi kode OTP (One-Time Password) yang dimasukkan oleh pengguna sebagai bagian dari proses registrasi. Setelah pengguna mendaftar, sistem mengirimkan kode OTP yang bersifat sementara ke perangkat mereka, melalui autentikator. Pengguna kemudian memasukkan kode tersebut ke dalam formulir registrasi untuk memastikan bahwa mereka adalah pemilik sah dari akun tersebut. Verifikasi OTP ini menambah lapisan keamanan pada proses registrasi, memastikan bahwa hanya pengguna yang memiliki akses ke perangkat yang bisa menyelesaikan pendaftaran.

4.5. Login(Request $request)

Login digunakan untuk memvalidasi username dan password pengguna, serta memproses autentikasi berdasarkan konfigurasi autentikasi dua faktor (2FA). Setelah pengguna memasukkan kredensial login mereka, sistem memverifikasi kebenaran username dan password. Jika valid, langkah berikutnya adalah memeriksa konfigurasi 2FA, yang mengharuskan pengguna untuk memasukkan kode OTP (One-Time Password) yang dikirim melalui aplikasi autentikator . Proses ini memastikan bahwa hanya pengguna yang terautentikasi dengan benar yang dapat mengakses akun, memberikan lapisan keamanan tambahan untuk mencegah akses yang tidak sah.

4.6. VerifyOtp(Request $request)

VerifyOtp digunakan untuk memverifikasi kode OTP (One-Time Password) yang dimasukkan oleh pengguna saat login sebagai bagian dari proses autentikasi dua faktor (2FA). Setelah pengguna memasukkan username dan password yang valid, sistem mengirimkan kode OTP ke perangkat pengguna melalui aplikasi autentikator. Pengguna kemudian memasukkan kode tersebut ke dalam sistem. Jika kode OTP yang dimasukkan sesuai dengan yang dikirimkan, maka proses login berhasil dan pengguna diberikan akses ke akun mereka. Proses verifikasi OTP ini menambah lapisan keamanan tambahan untuk mencegah akses yang tidak sah.

4.7. Logout()

Fungsi logout untuk menghapus sesi pengguna yang sedang aktif, sehingga mengakhiri akses ke aplikasi atau sistem. Setelah pengguna melakukan logout, sistem akan menghapus semua data sesi yang terkait dengan pengguna tersebut, seperti token autentikasi atau informasi login lainnya. Proses ini memastikan bahwa pengguna tidak lagi memiliki akses ke akun mereka tanpa melakukan login ulang. Setelah sesi dihapus, pengguna akan diarahkan kembali ke halaman login untuk memulai sesi baru jika ingin mengakses aplikasi atau sistem kembali. Hal ini memberikan lapisan keamanan tambahan dengan menghindari akses tidak sah setelah sesi berakhir.

4.8. Strategi Pengujian

Strategi Pengujian berfokus pada tiga aspek utama: Validasi Input, Logika Cabang, dan Error Handling. Pada validasi input, pengujian memastikan bahwa semua input yang diterima oleh fungsi sesuai dengan aturan bisnis yang ditetapkan, sehingga hanya data yang valid yang diproses. Uji logika cabang dilakukan untuk memastikan bahwa kondisi if dan else dalam kode telah diuji secara menyeluruh, memastikan jalur logika berfungsi sesuai harapan. Sementara itu, pengujian error handling bertujuan untuk memastikan bahwa kesalahan yang terjadi dapat ditangani dengan baik, memberikan umpan balik yang informatif kepada pengguna untuk meningkatkan pengalaman penggunaan aplikasi.

4.9. Lingkup Pengujian

Lingkup pengujian mencakup beberapa fungsi utama dalam sistem, yaitu GenerateKey, CompleteRegistration, Login, VerifyOtp, dan Logout. Setiap fungsi diuji untuk memastikan bahwa fungsionalitasnya berjalan dengan benar sesuai dengan tujuan yang telah ditentukan.

4.10. Implementasi Pengujian

Implementasi Pengujian dilakukan menggunakan PHPUnit, sebuah framework pengujian untuk PHP. Skenario pengujian disusun dengan seksama untuk menguji berbagai kondisi yang mungkin terjadi selama eksekusi fungsi-fungsi tersebut, memastikan sistem berfungsi dengan baik dan aman. Skenario pengujian disusun sebagai berikut:

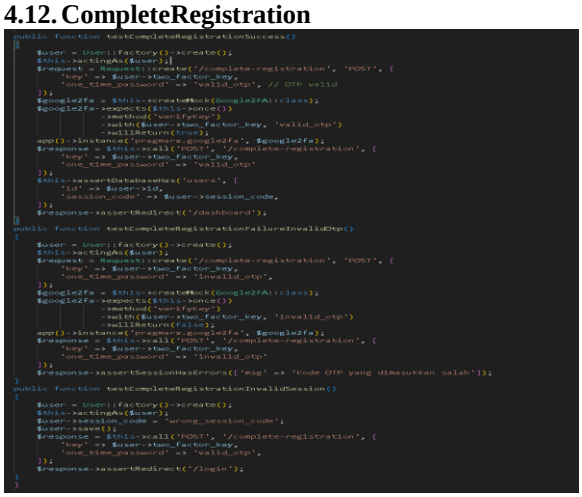
4.11. GenerateKey

```
public function testGenerateKeySuccess()
{
    $request = Request::create('/generate-key', 'POST', [
        'username' => 'testuser',
    ]);
    $google2fa = $this->createMock(Google2FA::class);
    $google2fa->expects($this->once())
              ->method('generateSecretKey')
              ->willReturn('testSecretKey');
    $google2fa->expects($this->once())
              ->method('getQRCodeInline')
              ->willReturn('QRImageData');
     app()->instance('pragmarx.google2fa', $google2fa);
    $response = $this->call('POST', '/generate-key', [
        'username' => 'testuser'
    ]);
    $this->assertSessionHas('registration_data');
    $this->assertEquals('testSecretKey', session('registration_data')['2fa_key']);
    $response->assertViewHas('QR_Image', 'QRImageData');
    $response->assertViewIs('auth.insert-key');
}
```

Gambar 2. Generate Key

CreateMock digunakan untuk membuat salinan objek Google2FA dan mengonfigurasi metode generateSecretKey serta getQRCodeInline agar mengembalikan nilai yang diinginkan saat dilakukan pengujian. assertSessionHas digunakan untuk menyimpan data yang dihasilkan 2fa_key, auth.insert-key digunakan untuk melakukan verivikasi QR Code dan tampilan sesuai yang diinginkan.

4.12. CompleteRegistration



```
public function testCompleteRegistrationSuccess()
{
    $user = User::factory()->create();
    $this->actingAs($user);
    $request = Request::create('/complete-registration', 'POST', [
        'key' => $user->two_factor_key,
        'one_time_password' => 'valid_otp', // OTP valid
    ]);
    $google2fa = $this->createMock(Google2FA::class);
    $google2fa->expects($this->once())
              ->method('verifyKey')
              ->with($user->two_factor_key, 'valid_otp')
              ->willReturn(true);
    app()->instance('pragmarx.google2fa', $google2fa);
    $response = $this->call('POST', '/complete-registration', [
        'key' => $user->two_factor_key,
        'one_time_password' => 'valid_otp'
    ]);
    $this->assertDatabaseHas('users', [
        'id' => $user->id,
        'session_code' => $user->session_code,
    ]);
    $response->assertRedirect('/dashboard');
}
public function testCompleteRegistrationFailureInvalidOtp()
{
    $user = User::factory()->create();
    $this->actingAs($user);
    $request = Request::create('/complete-registration', 'POST', [
        'key' => $user->two_factor_key,
        'one_time_password' => 'invalid_otp',
    ]);
    $google2fa = $this->createMock(Google2FA::class);
    $google2fa->expects($this->once())
              ->method('verifyKey')
              ->with($user->two_factor_key, 'invalid_otp')
              ->willReturn(false);
    app()->instance('pragmarx.google2fa', $google2fa);
    $response = $this->call('POST', '/complete-registration', [
        'key' => $user->two_factor_key,
        'one_time_password' => 'invalid_otp'
    ]);
    $response->assertSessionHasErrors(['msg' => 'Kode OTP yang dimasukkan salah']);
}
public function testCompleteRegistrationInvalidSession()
{
    $user = User::factory()->create();
    $this->actingAs($user);
    $user->session_code = 'wrong_session_code';
    $user->save();
    $response = $this->call('POST', '/complete-registration', [
        'key' => $user->two_factor_key,
        'one_time_password' => 'valid_otp',
    ]);
    $response->assertRedirect('/login');
}
}
```

Gambar 3. Complete Registraton

Test pertama (testCompleteRegistrationSuccess): Menguji skenario di mana OTP valid dan pengguna

berhasil menyelesaikan registrasi. Memastikan sesi dan data pengguna diperbarui dan pengguna diarahkan ke dashboard, Test berikutnya pada fungsi (testComplete RegistrationFailureInvalidOtp) menguji skenario di mana OTP tidak valid, untuk memastikan pengguna diarahkan kembali dengan pesan error yang tepat. Test berikutnya pada fungsi (testCompleteRegistrationInvalidSession) untuk menguji skenario di mana sesi pengguna tidak valid dan memastikan mereka diarahkan ke halaman login.

4.13. Validasi Input

```php
public function login(Request $request)
    {
        $google2fa = app('pragmarx.google2fa');
        $validator = Validator::make($request->all(), [
            'username' => 'required',
            'password' => 'required|string',
        ]);
        if ($validator->fails()) {
            return response()->json($validator->errors(), 422);
        }
    }
```

Gambar 4. Kode Login

Pengujian kode validasi input dilakukan dengan memasukkan login tanpa username atau password, ekspektasi yang diharapkan akan muncul error dengan kode status 422.

4.14. Logika Cabang

```php
public function testVerifyOtpValid()
{
    $user = User::factory()->create(['2fa_key' => 'testKey']);

    $google2fa = app('pragmarx.google2fa');
    $validOtp = $google2fa->generateKey($user->two_factor_key);

    $response = $this->actingAs($user)->post('/verify-otp', [
        'one_time_password' => $validOtp,
    ]);
    $response->assertRedirect('/dashboard');
}
```

Gambar 5. Kode Validasi OTP

Ekspektasi dari pengujian tahap ini adalah ketika OTP yang dimasukkan benar maka akan diarahkan ke halaman dashboard.

```php
public function testVerifyOtpInvalid()
{
    $user = User::factory()->create(['2fa_key' => 'testKey']);
    $response = $this->actingAs($user)->post('/verify-otp', [
        'one_time_password' => '000000',
    ]);
    $response->assertSessionHasErrors(['msg' => 'Kode OTP yang dimasukkan salah']);
}
```

Gambar 6. Kode OTP salah

Ketika pengguna memasukkan kode OTP salah maka harapannya akan keluar pesan error kode OTP yang dimasukkan salah.

4.15. Error Handling Logout

```php
public function logout(){
    try {
        $userId = auth('web')->id();
        if (!$userId) {
            return redirect('/login')->with('error',
            'Pengguna tidak ditemukan atau belum login');
        }
        $user = User::where('id', $userId)->first();
        if (!$user) {
            return redirect('/login')->with('error', 'Pengguna tidak ditemukan');
        }
        $user->session_code = null;
        $user->session_start = null;
        $user->save();
        return redirect('/login')->with('success', 'Berhasil logout');
    } catch (\Exception $e) {
        return redirect('/login')->with('error',
        'Terjadi kesalahan saat logout: ' . $e->getMessage());
    }}
```

Gambar 7. Error Handling

Kode ini untuk mengimplementasikan penanganan kesalahan, dan ekspektasinya dapat menangani berbagai masalah yang mungkin terjadi saat proses logout. Dalam kode tersebut dilakukan pengecekan apakah pengguna sudah login, apakah pengguna ditemukan dalam database. Selain itu kode *try-catch* dibuat untuk menangkap kesalahan tak terduga yang mungkin terjadi saat pengambilan data atau penghapusan sesi, sehingga dapat menangani kasus ketidaksengajaan seperti pengguna yang tidak ditemukan di database meskipun id login valid,
Selain itu masalah teknis seperti kesalahan database, atau kesalahan sesi yang tidak terduga dan juga memberikan umpan balik yang jelas kepada pengguna ketika ada masalah, yaitu mengarahkan kembali ke halaman login dengan pesan yang tepat.

4.16. Hasil Uji *White Box*

Pengujian sistem autentikasi dilakukan menggunakan pendekatan white-box testing, yang mencakup validasi input, verifikasi OTP, dan logout. Hasil pengujian menunjukkan bahwa sistem telah berhasil menangani berbagai skenario, termasuk validasi input yang tidak valid, penerimaan OTP valid, penolakan OTP tidak valid, dan proses logout yang sesuai. Pada fungsi login, sistem mampu mengidentifikasi input tidak valid dengan memberikan respons error dan pesan kesalahan yang informatif. Pada fungsi verifyOtp, pengujian memastikan jalur logika menangani OTP valid dengan mengarahkan pengguna ke dashboard, sementara OTP tidak valid ditolak dengan pesan error. Fungsi logout terbukti menghapus sesi pengguna dan mengarahkan pengguna ke halaman login. Tidak ada bug kritis yang ditemukan selama pengujian, menunjukkan bahwa sistem cukup andal untuk digunakan dalam lingkungan produksi.

Tabel 1. Hasil pengujian

Fungsi	Skenario	Input	Ekspektasi	Hasil
login	Validasi input kosong	username='', password='xxxx'	Error 422, pesan kesalahan pada kolom username.	Lulus
verifyOtp	OTP valid	OTP dihasilkan dari kunci rahasia	Redirect ke /dashboard.	Lulus
verifyOtp	OTP tidak valid	OTP: '000000'	Pesan error "Kode OTP yang dimasukkan salah".	Lulus
logout	Logout	Pengguna login	Sesi pengguna dihapus, pengguna menjadi guest.	Lulus

5. KESIMPULAN DAN SARAN

Berdasarkan hasil penelitian yang dilakukan dalam pengujian white box dengan metode branch coverage terhadap Sistem Informasi Pengolahan Data Teritorial berbasis web, dapat disimpulkan bahwa pengujian ini berhasil memastikan bahwa setiap jalur logika dalam sistem telah diuji secara menyeluruh. Pendekatan ini memastikan bahwa seluruh fungsi berjalan sesuai dengan kebutuhan fungsional yang ditentukan, dengan meminimalkan risiko bug pada jalur eksekusi yang mungkin terjadi. Pengujian menunjukkan bahwa sistem ini dapat diimplementasikan dengan baik untuk membantu personel TNI AD dalam melaksanakan tugas operasi militer. Untuk pengembangan lebih lanjut, disarankan untuk menambahkan pengujian pada kondisi ekstrem seperti batas waktu atau beban tinggi, serta integrasi dengan teknologi baru guna meningkatkan fungsionalitas. Penelitian tambahan mengenai aspek keamanan sistem, termasuk pengujian penetrasi dan mitigasi risiko, juga diperlukan untuk memastikan sistem tetap aman, andal, dan efektif dalam mendukung tugas operasional.

DAFTAR PUSTAKA

[1] C. B. Chatlina, A. Mulyana, and M. Amalia, "Pengaruh Perkembangan Teknologi Informasi Dan Komunikasi Terhadap Kualitas Hubungan Sosial Dalam Keluarga," *KOMUNITAS J. Ilmu Sosiol.*, vol. 7, no. 1, pp. 19–38, 2024.

[2] S. Triharto, R. Pratama, and D. T, "5. Pengamanan Data Sistem Informasi Disinfolahtaau," *TNI Angkatan Udar.*, vol. 2, no. 1, pp. 1–10, 2023, doi: 10.62828/jpb.v2i1.55.

[3] A. Fahma Rosyada, I. Sukirman, M. Afrizal Nur, and A. Saifudin, "Pengujian Sistem Informasi Aplikasi Perpustakaan basis Website Menggunakan White Box Testing," *J. Multidisiplin ilmu*, vol. 1, no. 6, pp. 1034–1039, 2022, [Online]. Available: https://journal.mediapublikasi.id/index.php/oktal

[4] Rusfandi, "White Box Testing," Raharja.Ac.Id. [Online]. Available: https://www.imperva.com/learn/application-security/white-box-testing/

[5] S. Wibawa, S. Suryanto, and R. Ningsih, "Perlindungan Data Digital Dengan Time-Based One-Time Password (TOTP)," *INSANtek*, vol. 5, no. 1, pp. 30–36, 2024, doi: 10.31294/insantek.v5i1.3495.

[6] K. Kalfin, R. A. Ibrahim, and G. S. Laksito, "Optimization of White Box Testing by Utilizing Branching and Repeating Structures in Java Programs Using Base Path," *Int. J. Math. Stat. Comput.*, vol. 2, no. 2, pp. 85–89, 2024, doi: 10.46336/ijmsc.v2i2.98.

[7] * Asyura *et al.*, "Enhancing Authentication Security: Analyzing Time-Based One-Time Password Systems," *Int. J. Comput. Technol. Sci.*, no. 3, pp. 56–70, 2024.

[8] Keployio, "Understanding White Box Testing: An In-Depth Guide." [Online]. Available: https://medium.com/@keployio/understanding-white-box-testing-an-in-depth-guide-0cd763195e09

[9] A. Kasastra, "Implementing Time-based OTP as Two Factor Authentication (2FA) in Laravel." [Online]. Available: https://thearkas.medium.com/implementing-time-based-otp-as-two-factor-authentication-2fa-in-laravel-a267ff5c05e4

[10] K. Subandi, D. M. Ahmad, V. I. Sugara, A. S. Aryani, and H. Hermawan, "Protection Second Layer Menggunakan Multi Factor Authenticator Untuk Memverfikasi Keabsahan Account Email Di Dalam Active Directory," *J. Teknoinfo*, vol. 18, no. 2, p. 434, 2024, doi: 10.33365/jti.v18i2.4035.