**MERANCANG DAN MENGIMPLEMENTASIKAN PUSTAKA KOMPONEN AUTENTIKASI PENGGUNA YANG AMAN UNTUK *BLAZOR WASM* DAN *.NET CORE***

**SKRIPSI**

**Oleh**

**Isaiah Jamiel 2301934474**

**Izra Giffari 2301939166**

Diagram

Description automatically generated

**Cyber Security Program**

**Cyber Security Study Program**

**School of Computer Science**

**Universitas Bina Nusantara**

**Jakarta**

**2023**

**MERANCANG DAN MENGIMPLEMENTASIKAN PUSTAKA KOMPONEN AUTENTIKASI PENGGUNA YANG AMAN UNTUK *BLAZOR WASM* DAN *.NET CORE***

**SKRIPSI**

**diajukan sebagai salah satu syarat**

**untuk gelar kesarjanaan pada**

**Program Teknik Informatika**

**School of Computer Science**

**Jenjang Pendidikan Strata-1**

**Oleh**

**Isaiah Jamiel 2301934474**

**Izra Giffari 2301939166**

Diagram

Description automatically generated

**Cyber Security Program**

**Cyber Security Study Program**

**School of Computer Science**

**Universitas Bina Nusantara**

**Jakarta**

**2023**

**MERANCANG DAN MENGIMPLEMENTASIKAN PUSTAKA KOMPONEN AUTENTIKASI PENGGUNA YANG AMAN UNTUK *BLAZOR WASM* DAN *.NET CORE***

**SKRIPSI**

**Disusun oleh:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Isaiah Jamiel** |  | **Izra Giffari** |  |
| **2301934474** |  | **2301939166** |  |

**Disetujui Oleh:**

**Pembimbing**

**Nadia, S.Kom., M.TI**

**D5782**

**Yohan Muliono, S.Kom., M.Kom.**

***Head of Cyber Security Study program***

**Universitas Bina Nusantara**

**Jakarta**

**2023**

# BAB 1

**PENDAHULUAN**

## Latar Belakang

Peningkatan ketergantungan pada aplikasi perangkat lunak di berbagai sektor telah membuat autentikasi pengguna menjadi komponen kritis keamanan perangkat lunak. Memastikan bahwa hanya pengguna yang terautorisasi yang dapat mengakses sistem atau aplikasi sangat penting untuk melindungi data sensitif dan mencegah akses yang tidak sah.

Autentikasi pengguna adalah komponen kritis keamanan perangkat lunak, karena memastikan bahwa hanya pengguna yang terautorisasi yang dapat mengakses sistem atau aplikasi. Terdapat banyak pendekatan yang berbeda terhadap autentikasi pengguna, mulai dari kombinasi username dan password sederhana hingga metode yang lebih canggih seperti autentikasi biometrik dan autentikasi multi-faktor (Mittal & Gupta, 2018). Namun, menerapkan autentikasi pengguna yang aman dapat menjadi proses yang kompleks dan memakan waktu, terutama bagi para pengembang yang mungkin tidak memiliki keahlian di bidang keamanan.

Pustaka komponen menyediakan solusi yang berguna untuk masalah ini dengan menawarkan kode yang sudah dibangun, diuji, dan didokumentasikan yang dapat dengan mudah disatukan ke dalam proyek perangkat lunak. Dengan menggunakan pustaka komponen untuk autentikasi pengguna, para pengembang dapat fokus pada pembangunan fungsi inti aplikasi mereka, bukan mengeluarkan waktu dan sumber daya untuk mengembangkan dan menguji mekanisme autentikasi (Mittal & Gupta, 2018).

Pustaka komponen autentikasi digunakan dalam kerangka kerja untuk membantu memverifikasi identitas pengguna yang mengakses sumber daya atau melakukan tindakan tertentu dalam sebuah aplikasi. Pustaka-pustaka ini biasanya menyediakan mekanisme untuk membuat dan mengelola akun pengguna, serta melakukan verifikasi kredensial pengguna antara lain kombinasi nama pengguna dan kata sandi.

Ada beberapa tujuan dalam penggunaan library autentikasi:

1. Untuk memastikan bahwa hanya pengguna yang terautorisasi yang dapat mengakses sumber daya yang sensitif atau terbatas.
2. Untuk memberikan cara yang konsisten dan aman untuk mengelola akun pengguna dan kata sandi di seluruh aplikasi.
3. Untuk mengurangi beban kerja developer dengan menyediakan set fungsi autentikasi yang sudah dibangun sebelumnya yang dapat dengan mudah diintegrasikan ke dalam sebuah aplikasi.
4. Untuk memberikan cara standar untuk menangani tugas autentikasi umum, seperti pemulihan kata sandi.
5. Secara keseluruhan, penggunaan library autentikasi dalam kerangka kerja membantu meningkatkan keamanan dan kemudahan penggunaan sebuah aplikasi, sambil juga memudahkan bagi developer untuk mengimplementasikan fungsi autentikasi.

Blazor telah memperoleh minat dan adopsi yang signifikan dalam komunitas developer sejak dirilis pada tahun 2018. Menurut Survei Pengembang Stack Overflow untuk tahun 2021, Blazor menduduki peringkat ke-13 kerangka kerja web paling populer di kalangan developer, yang merupakan peningkatan signifikan dari peringkat pada tahun-tahun sebelumnya.

Ada beberapa alasan mengapa Blazor mendapatkan popularitas dalam beberapa tahun terakhir:

1. Peningkatan produktivitas pengembang: Blazor memungkinkan pengembang menggunakan C# dan .NET untuk membangun aplikasi web statis sisi klien, yang bisa lebih produktif bagi pengembang yang sudah terbiasa dengan teknologi ini.
2. Basis kode bersama: Karena Blazor menggunakan CLR yang sama dengan ekosistem .NET, Blazor memungkinkan pengembang untuk berbagi kode antara sisi server dan klien dari suatu aplikasi, yang dapat meningkatkan pemeliharaan basis kode dan mengurangi duplikasi upaya.
3. Mengurangi ketergantungan pada JavaScript: Blazor memungkinkan pengembang membuat aplikasi web tanpa terlalu bergantung pada JavaScript, yang dapat bermanfaat bagi pengembang yang lebih suka bekerja dengan bahasa C# dan bahasa yang kompatibel dengan .NET lainnya. Blazor pun tetap memberikan opsi untuk berinteraksi dengan Javascript untuk mendukung fitur yang hanya ada di Javascript.
4. Performa yang ditingkatkan: Aplikasi Blazor akan di kompilasi ke WebAssembly, maka akan lebih cepat dibanding aplikasi berbasis JavaScript, terutama untuk aplikasi yang sering memerlukan pembaruan UI.

Secara keseluruhan, Blazor menawarkan sejumlah manfaat bagi pengembang dan organisasi, dan popularitasnya kemungkinan besar akan terus tumbuh karena semakin banyak orang yang menyadari kemampuannya.

Dengan naiknya popularitas Blazor dalam dunia web, kami melihat bahwa minim adanya library yang dapat membantu dalam menjaga keamanan website berbasis Blazor dan hal ini menyebabkan beberapa masalah autentikasi potensial yang mungkin dihadapi oleh para pengembang ketika bekerja dengan framework Blazor. Beberapa masalah umum termasuk:

* Mengimplementasikan autentikasi: Tergantung pada kebutuhan spesifik aplikasi, mengimplementasikan autentikasi dalam aplikasi Blazor dapat menjadi tantangan. Para pengembang mungkin perlu mengintegrasikan dengan penyedia autentikasi eksternal, seperti OAuth atau OpenID Connect, atau membangun solusi autentikasi kustom mereka sendiri.
* Mengelola sesi pengguna: Aplikasi Blazor menggunakan model client-side, yang berarti sesi pengguna harus dikelola di sisi client. Ini dapat lebih kompleks daripada mengelola sesi pengguna di sisi server, dan para pengembang mungkin perlu mempertimbangkan masalah seperti kadaluarsa sesi dan refresh.
* Menjamin keamanan data: Ketika bekerja dengan data sensitif, penting untuk memastikan bahwa data tersebut dikirim dan disimpan dengan aman. Ini dapat menjadi tantangan dalam aplikasi Blazor, karena data sering dikirim melalui jaringan dalam teks biasa. Para pengembang mungkin perlu mempertimbangkan implementasi enkripsi atau tindakan keamanan lain untuk melindungi data sensitif.
* Menangani kesalahan: Jika terjadi kesalahan selama proses autentikasi, sulit bagi para pengembang untuk men-debug dan mencari tahu masalah tersebut. Ini dapat menjadi terutama sulit jika kesalahan terjadi di sisi client aplikasi.

Mengingat juga banyaknya kasus kebocoran data-data di Indonesia dalam beberapa tahun terakhir ini yang memungkinkan data yang bocor adalah data terkait kredensial login. Maka dari itu, kami melihat bahwa perancangan library komponen autentikasi akan berdampak positif dalam dunia perkembangan web yang terus berkembang.

Tujuan dari tesis ini adalah merancang dan melakukan implementasi pustaka komponen autentikasi pengguna yang aman untuk digunakan dengan Blazor WebAssembly (WASM) dan .NET Core. Pustaka tersebut akan memberikan para pengembang dengan berbagai metode autentikasi yang dapat dengan mudah disatukan ke dalam aplikasi mereka, yang membantu untuk memastikan bahwa autentikasi pengguna diimplementasikan secara konsisten dan aman. Pustaka akan diuji dan dievaluasi untuk memastikan bahwa sesuai dengan standar keamanan yang dibutuhkan dan mudah digunakan oleh para pengembang.

## Rumusan Masalah

* + 1. Bagaimana cara membuat sebuah library komponen autentikasi yang modular dan mudah di implementasikan untuk para penggunanya?
    2. Bagaimana cara mengimplementasikan pustaka komponen autentikasi pengguna yang aman menggunakan Blazor WebAssembly dan .NET Core?
    3. Bagaimana cara menjalankan seluruh proses autentikasi dengan cepat namun tetap aman?

## Ruang Lingkup

Berikut ini adalah ruang lingkup penelitian yang dilakukan:

* Melakukan tinjauan literatur mengenai metode dan praktik terbaru dalam otentikasi pengguna, keamanan aplikasi web, serta penggunaan Blazor WASM dan .NET Core.
* Merancang pustaka komponen otentikasi pengguna yang aman dengan mempertimbangkan arsitektur dan fitur yang dibutuhkan dalam konteks Blazor WASM dan .NET Core.
* Melakukan pengujian terhadap pustaka komponen otentikasi untuk memastikan keamanan, kinerja, dan fungsionalitasnya. Mengevaluasi keberhasilan implementasi pustaka dalam memenuhi kebutuhan otentikasi pengguna.

## Tujuan dan Manfaat

Tujuan dilakukannya penelitian ini adalah sebagai berikut:

* + - 1. Memberikan solusi untuk developer dalam membangun situs berbasis Blazor dan .NET Core yang aman dengan mudah dan cepat secara pengmbangan.
      2. Meningkatkan kualitas keamanan website yang menggunakan kerangka kerja blazor dan .NET.

Adapun manfaat dari penelitian ini adalah:

* + - 1. Mempermudah developer dalam mengembangkan situs dengan sistem autentikasi yang aman.
      2. Menyediakan solusi autentikasi untuk developer .NET yang bisa dipercaya

## Metodologi

Penelitian yang dilakukan terdiri dari beberapa tahap serta metode yang digunakan, yaitu:

* + 1. **Metode Pengumpulan Data**

1. Interview

Teknik ini dilakukan dengan melakukan interview narasumber yang berkaitan dengan pustaka komponen dan pengembangan aplikasi berbasis web dengan tujuan memvalidasi masalah dan kebutuhan serta manfaat yang ditawarkan oleh komponen pustaka yang akan dibuat.

1. Studi Literatur

Studi literatur merupakan metode penelitian kualitatif yang mengumpulkan data dari berbagai sumber tertulis yang relevan dengan topik penelitian. Studi literatur akan digunakan untuk mengumpulkan, menganalisis, dan merangkum informasi yang relevan tentang berbagai konsep, teori, dan penelitian sebelumnya yang berhubungan dengan desain dan implementasi autentikasi pengguna, Blazor WASM, dan .NET Core.

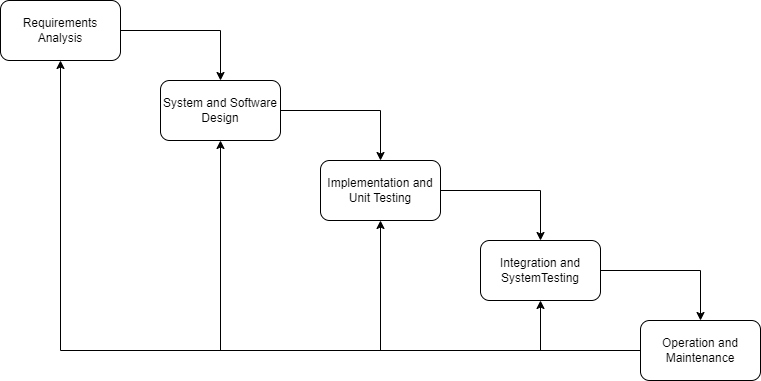
1. Perbandingan aplikasi sejenis

Teknik ini dilakukan untuk mengetahui fitur-fitur yang sudah ada maupun belum dari pustaka komponen yang sejenis. Hal ini dilakukan sebagai bahan pertimbangan untuk melihat perbedaan yang dapat dijadikan sebagau fungsi tambahan dan nilai pada pustaka komponen yang akan dibuat.

* + 1. **Metode Pengembangan**

Metode pengembangan atau *Software Development Life Cycle* (SDLC)yang diterapkan dalam penelitian ini adalah *Waterfall Model*. *Waterfall Model* adalah suatu pendekatan dalam rekayasa perangkat lunak yang digunakan untuk mengembangkan aplikasi. Metode ini melibatkan serangkaian tahapan yang dimulai dengan analisis masalah dan pengumpulan kebutuhan pengguna, perencanaan jadwal dan pembagian tugas selama pengembangan aplikasi, desain aplikasi menggunakan UML, pembangunan aplikasi sesuai dengan desain yang telah dibuat, dan akhirnya melakukan evaluasi terhadap aplikasi yang telah dibangun.

*Waterfall Mode*l yang akan digunakan untuk pengembangan komponen pustaka akan dibagi menjadi 5 tahap yaitu yaitu *Requirement Analysis, System and Software Design, Implementation and Unit Testing*, *Integration and System Testing*, dan *Operation and Maintenance.*



* + 1. **Metode Evaluasi**

Metode evaluasi yang akan digunakan mencakup uji fungsional dan penilaian keamanan. Berikut penjelasan lebih detail:

1. **Uji Fungsional**: Metode ini akan digunakan untuk mengevaluasi apakah komponen autentikasi pengguna yang telah dikembangkan berfungsi sesuai dengan persyaratan dan ekspektasi. Ini akan melibatkan pengujian setiap fungsi dan fitur yang ada dalam komponen untuk memastikan bahwa mereka bekerja dengan benar. Beberapa aspek yang akan diperiksa mencakup proses login, logout, pembuatan akun, pemulihan kata sandi, dan manajemen sesi pengguna.
2. **Penilaian Keamanan**: Metode ini akan digunakan untuk mengevaluasi sejauh mana pustaka komponen autentikasi pengguna yang dikembangkan aman dari ancaman dan serangan potensial. Ini akan melibatkan penggunaan berbagai teknik dan alat untuk mencoba menemukan dan mengeksploitasi kelemahan dalam sistem. Aspek yang akan diperiksa mencakup keamanan data pengguna, perlindungan terhadap serangan seperti Cross-Site Scripting (XSS) dan Cross-Site Request Forgery (CSRF), serta keefektifan fitur keamanan lainnya seperti hash password dan autentikasi dua faktor.

## Sistematika Penulisan

Penulisan Tugas Akhir ini dibagi menjadi 5 (lima) bab yaitu:

BAB 1 Pendahuluan

Bab ini berisi tentang latar belakang yang menjadi dasar dilakukannya penelitian, rumusan masalah, ruang lingkup, tujuan dan manfaat, metode penelitian yang digunakan, serta sistematika penulisan.

BAB 2 Tinjauan Pustaka

Dalam bab ini, akan dijelaskan landasan teori yang menjadi dasar panduan dalam pembuatan aplikasi ini, serta metode yang digunakan dalam penelitian ini.

BAB 3 Metode Penelitian

Bab ini berisi kerangka berpikir, metode penelitian dalam menganalisa masalah, perencanaan dalam membuat aplikasi, pengembangan algoritma, pengembangan pustaka komponen, perancangan sistem dengan UML, dan metode yang digunakan dalam membuat pustaka komponen.

BAB 4 Hasil dan Pembahasan

Bab ini berisi spesifikasi sistem, hasil pengembangan aplikasi secara keseluruhan, serta hasil uji coba dari metode yang digunakan. Metode yang digunakan dikaitkan dengan teori dan konsep dari literatur terkait. Bab ini juga berisi hasil evaluasi yang meliputi evaluasi sistem, dan evaluasi aplikasi secara keseluruhan.

BAB 5 Simpulan dan Saran

Bab ini berisi simpulan dari seluruh pembahasan dan menjawab rumusan masalah yang dipaparkan pada bab 1. Bab ini juga berisi saran dan usulan untuk pengembangan aplikasi yang lebih baik di waktu mendatang.

# BAB 2

**TINJAUAN PUSTAKA**

## Pustaka Komponen

Komponen pustaka adalah kumpulan komponen perangkat lunak yang dapat digunakan kembali yang dapat digunakan untuk membangun atau meningkatkan fungsi aplikasi perangkat lunak. Komponen-komponen ini biasanya dirancang untuk modul dan independen, sehingga dapat dengan mudah disatukan ke dalam berbagai sistem yang berbeda. (newtarget, 2022)

Pustaka komponen dapat mencakup berbagai jenis komponen yang berbeda, seperti elemen antarmuka pengguna, modul logika bisnis, komponen akses data, dan lainnya. Mereka mungkin dikembangkan dan dipelihara oleh satu organisasi, atau mungkin dikumpulkan dari berbagai sumber dan tersedia untuk digunakan oleh komunitas yang lebih besar. (newtarget, 2022)

## Autentikasi

Autentikasi adalah proses verifikasi identitas pengguna, perangkat, atau sistem (Jain, 2014). Ini merupakan aspek penting keamanan komputer yang memastikan hanya individu atau entitas yang terautorisasi yang dapat mengakses sumber daya atau melakukan tindakan pada sistem (Barker et al., 2018).

Terdapat beberapa metode autentikasi yang dapat digunakan untuk memverifikasi identitas pengguna atau sistem. Metode-metode tersebut antara lain:

* + 1. **Autentikasi Berbasis Kata Sandi**

Autentikasi berbasis sandi adalah suatu proses yang digunakan untuk memverifikasi identitas seseorang atau sebuah sistem dengan menggunakan sebuah kombinasi dari nama pengguna dan sandi (password). Sandi biasanya merupakan sebuah string (baris teks) yang diketahui hanya oleh pengguna yang bersangkutan, dan digunakan sebagai faktor autentikasi utama.

* + 1. **Autentikasi Dua Faktor**

Autentikasi dua faktor (two-factor authentication atau 2FA) adalah suatu proses yang digunakan untuk memverifikasi identitas seseorang atau sebuah sistem dengan menggunakan dua faktor yang berbeda. Dua faktor tersebut adalah faktor yang diketahui hanya oleh pengguna (seperti sandi atau kata sandi) dan faktor yang tidak diketahui oleh pengguna (seperti kode verifikasi yang dikirim ke nomor telepon atau alamat email pengguna).

* + 1. **Autentikasi Multifaktor**

Autentikasi faktor ganda (MFA) adalah proses keamanan yang memerlukan lebih dari satu metode otentikasi dari kategori credential yang independen untuk memverifikasi identitas pengguna. Ini berarti bahwa selain password, faktor kedua diperlukan untuk l*ogin* ke sistem atau mengakses informasi yang dilindungi. Tujuan MFA adalah untuk membuat lebih sulit bagi pengguna yang tidak sah untuk mengakses sistem atau data, bahkan jika mereka telah mendapatkan nama pengguna dan sandi yang valid.

## Enkripsi dan Dekripsi

Enkripsi adalah proses penting dalam ilmu komputer dan keamanan siber yang melibatkan transformasi teks biasa atau data tak terenkripsi menjadi bentuk yang tidak dapat dibaca oleh siapa pun tanpa kunci dekripsi. Data yang terenkripsi disebut sebagai teks sandi. Enkripsi digunakan untuk melindungi informasi sensitif dari akses oleh individu seperti hacker atau kriminal.

Salah satu algoritma enkripsi tertua dan paling banyak digunakan adalah Advanced Encryption Standard (AES). AES adalah algoritma enkripsi simetris yang beroperasi pada blok data dengan ukuran tetap. Algoritma ini telah banyak diadopsi oleh organisasi dan pemerintah di seluruh dunia karena memiliki keamanan yang kuat.

Algoritma enkripsi penting lainnya adalah algoritma RSA, yang didasarkan pada kompleksitas matematika faktorisasi bilangan besar. RSA adalah algoritma enkripsi asimetris yang menggunakan kunci publik untuk enkripsi dan kunci privat untuk dekripsi. Algoritma ini telah banyak digunakan dalam mengamankan komunikasi, tanda tangan digital, dan pertukaran kunci.

Selain algoritma enkripsi dasar ini, terdapat kemajuan dalam teknik enkripsi seperti kriptografi kurva eliptik (ECC), enkripsi homomorfik, dan kriptografi pasca-kuantum. ECC dikenal karena kemampuannya untuk memberikan tingkat keamanan yang sama dengan algoritma enkripsi tradisional dengan ukuran kunci yang lebih kecil, sehingga cocok untuk perangkat yang terbatas sumber daya. Enkripsi homomorfik memungkinkan komputasi dilakukan pada data yang terenkripsi tanpa dekripsi, yang sangat berguna dalam skenario di mana privasi merupakan perhatian utama. Kriptografi pasca-kuantum berfokus pada pengembangan algoritma enkripsi yang tahan terhadap serangan oleh komputer kuantum, yang memiliki potensi untuk merusak banyak algoritma enkripsi yang ada.

## C#

C# adalah bahasa pemograman berorientasi pada obyek yang modern, inovatif, *open-source*, dan *cross-platform* (Microsoft, 2022). C# memiliki sistem tipe data yang kuat, sehingga setiap variabel dan konstanta harus memiliki tipe data (Microsoft, 2022). Meskipun C# merupakan bahasa yang berorientasi pada obyek pada akarnya, C# juga menawarkan paradigma lain untuk developer bisa menggunakan bahasa tersebut. Antara lain, C# memungkinkan developer untuk mengambil pendekatan fungsional dan bisa mengimplementasikan lambda (BillWagner, GeWarren, pkulikov, wuyuansushen, & DCtheGeek, 2022).

C# awalnya dibuat untuk menjadi pesaing bahasa Java*.* Pada versi-versi awal C#, bahasa buatan Microsoft ini sangat mirip dengan Java. Namun, semakin berkembangnya C#. C# semakin memiliki jati diri nya sendiri sebagai bahasa pemograman. C# mencoba untuk menghilangkan *boilerplate* dari struktur kodenya dan mulai menawarkan fitur-fitur yang mengarah kepada paradigma pemograman lain.

Salah satu fitur yang paling digemari oleh pengguna C# adalah LINQ. LINQ menjadi salah satu nilai jual paling tinggi dari C# atau bahasa lain di framework .NET. LINQ memungkinkan developer untuk mengolah data secara deklaratif. Dimana bahasa lain banyak yang belum memiliki fitur ini.

List<String> a;

a.Where(a => a == "test");

cth. LINQ

Selain LINQ, C# dan bahasa-bahasa lain di framework .NET juga memiliki sebuah IDE atau *Integrated development environment* yang para developer terutama di bidang korporat bisa percaya akan terus ada dan di support oleh *Microsoft* sendiri. IDE ini adalah Visual Studio. Visual studio memiliki banyak fitur, yang dimana pada masa Visual Studio baru di terbitkan ia merupakan IDE terbaik yang ada. Fitur-fitur seperti *debug,* pelengkap kode secara otomatis, dan juga fitur-fitur lain ini dinilai oleh perusahaan bisa membuat pemograman lebih cepat, sehingga bisa menghemat biaya.

## Framework

## .NET

.NET (dibaca dotnet) merupakan sebuah platform developer yang gratis, *cross-platform*, dan *open-source.* Aplikasi .NET dapat ditulis dalam 3 bahasa yang didukung secara resmi, yaitu C#, F#, dan Visual Basic. .NET memiliki API yang konsisten, sehingga aplikasi yang berbeda, jika berbasis .NET masih bisa berbagi kode dan menggunakan library yang sama (Microsoft, 2022).

.NET sebuah microsoft pertama kali dibuat untuk menyaingi Java dengan teknologi JVM nya. Untuk menyaingi *Java Virtual Machine* (disingkat JVM) milik Java, .NET memiliki teknologi nya sendiri yang dinamakan *Common Language Runtime* (disingkat CLR). CLR dibuat demi mempermudah programmer untuk mencapai beberapa hal, antara lain manajemen memori dan manajemen *thread*. CLR ini merupakan sebuah lapisan penghubung antara bahasa pemograman dan Sistem Operasi.

Seperti namanya, CLR dibuat untuk menyatukan lebih dari satu bahasa pemograman agar bisa berjalan dan berinteraksi antara satu sama lain melalui dua teknologi, *Common Language Specification* (disingkat CLS) dan *Common Type System* (disingkat CTS).

*Common language specification* merupakan seperangkat aturan dan larangan yang berfungsi untuk menjembatani antar bahasa pemograman. Hal ini memungkinkan untuk lebih dari satu bahasa digunakan di satu proyek yang sama untuk tidak saling merusak satu sama lain. CLS juga merupakan teknologi yang memungkinkan sebuah class di sebuah bahasa menerapkan *inheritance* sebuah class dari bahasa lain. Seperti, sebuah *class* di C# bisa melakukannya terhadap sebuah class di VB.NET. Namun, hal ini tidak sempurna dan memiliki larangan-larangan tertentu. Seperti, sebuah *class* di C++ tidak bisa di *inherit* oleh class di C#. Contoh lain adalah, saat memanggil *method* dari bahasa lain. Bahasa yang memanggilnya harus mengikuti aturan dari bahasa yang dipanggil. Salah satu contoh nya adalah, saat memanggil bagian-bagian dari kode di C#, VB.NET tetap harus mengikuti aturan di C#. Jadi, saat pemanggilan bagian dari C#, menjadi *case sensitive*.

Di sisi lain, *Common Type System* merupakan seperangkat aturan yang menyatukan tipe-tipe data yang digunakan di berbagai bahasa pemograman. CTS akan mengetahui bahwa int di C# adalah sama dengan integer di VB.NET. CTS akan menyatakan bahwa mereka kompatibel sehingga pemanggilan data bisa dilakukan dengan benar.

Kedua CTS dan CLS digunakan di dalam CLR untuk menginterpretasi kode yang telah di kompilasi. Saat ini, .NET memiliki 3 bahasa pemograman utama yang di dukung secara penuh, yakni C#, VB.NET, dan F#. Blazor WASM

## Blazor

Blazor adalah kerangka kerja berbasis .NET yang menggunakan teknologi *WebAssembly* untuk memungkinkan membuat sebuah situs statis sisi klien responsive tanpa menggunakan JavaScript. Meskipun demikian, Blazor tetap dapat memanggil library dan API Javascript melalui C# (Microsoft, 2022).

Salah satu keunggulan utama Blazor adalah kemampuannya untuk membuat aplikasi web yang sepenuhnya berbasis komponen. Dalam Blazor, pengembang dapat membuat komponen UI yang dapat digunakan kembali dan dirangkai bersama untuk membentuk aplikasi yang lebih besar. Setiap komponen memiliki logika dan tampilan sendiri, dan dapat berinteraksi dengan komponen lain dalam aplikasi. Ini memungkinkan pengembang untuk membagi aplikasi menjadi bagian-bagian yang lebih kecil, yang dapat dikelola dan diuji secara terpisah.

Blazor mendukung dua mode utama: Blazor Server dan Blazor WebAssembly. Pada mode Blazor Server, logika aplikasi dijalankan di server dan UI diperbarui secara real-time melalui koneksi jaringan ke browser pengguna. Ini memungkinkan pengembang untuk membuat aplikasi web yang responsif dengan interaksi yang halus, tanpa harus mengirimkan kode C# ke browser. Di sisi lain, Blazor WebAssembly memungkinkan pengembang untuk menjalankan kode C# langsung di browser pengguna melalui teknologi WebAssembly. Ini memberikan pengalaman pengguna yang lebih cepat dan lebih kaya, karena kode aplikasi dieksekusi di sisi klien.

Blazor juga mendukung integrasi yang baik dengan ekosistem .NET yang ada. Pengembang dapat menggunakan banyak library dan alat yang sudah ada dalam ekosistem .NET untuk mempercepat proses pengembangan. Selain itu, Blazor juga mendukung penggunaan komponen UI dari Bootstrap dan Material Design, serta integrasi dengan layanan cloud seperti Azure.

Dengan fitur-fitur yang kuat dan kemampuan untuk menghindari ketergantungan pada JavaScript, Blazor menawarkan pendekatan yang menarik untuk pengembangan aplikasi web. Dengan menggunakan C# dan .NET, pengembang dapat memanfaatkan kekuatan dan produktivitas bahasa dan kerangka kerja yang sudah mapan, sambil tetap menghasilkan aplikasi web yang modern dan responsif.

## Unified Modeling Language

Dalam pengembangan aplikasi, terdapat desain atau model yang telah dibuat sebagai panduan dalam proses pengembangan aplikasi. Menurut Larman (2018), Unified Modeling Language (UML) adalah bentuk pemodelan yang terstandarisasi yang merepresentasikan sebuah sistem menggunakan berbagai jenis diagram.

* + 1. **Data Flow Diagram**

Diagram Aliran Data (DFD) adalah representasi grafis dari "aliran" data melalui sistem informasi, memodelkan aspek prosesnya. DFD sering digunakan dalam analisis atau desain sistem dan perangkat lunak, untuk menggambarkan bagaimana data diproses dalam suatu sistem.

Berikut ini adalah komponen utama dari diagram aliran data:

|  |  |  |  |
| --- | --- | --- | --- |
| No | Gambar | Nama | Keterangan |
| 1 | C:\Users\izra.giffari\Downloads\Untitled Diagram.drawio (9).png | *Processes* | *Processes* adalah operasi-operasi yang mengubah data dari input menjadi output. |
| 2 | C:\Users\izra.giffari\Downloads\Untitled Diagram.drawio (8).png | *Entity* | *Entity* adalah elemen eksternal yang berinteraksi dengan sistem, seperti pengguna atau sistem eksternal. |
| 3 | C:\Users\izra.giffari\Downloads\Untitled Diagram.drawio (7).png | *Data Flow* | *Data Flow* adalah saluran melalui mana data bergerak. Mereka direpresentasikan sebagai panah yang menunjukkan arah pergerakan data. |
| 4 | C:\Users\izra.giffari\Downloads\Untitled Diagram.drawio (5).png | *Data Store* | *Data Store* adalah tempat di mana data disimpan untuk digunakan nanti, seperti *database* atau file |

## Prototyping

Prototyping merupakan metode yang digunakan dalam proses pengembangan awal produk atau sistem di mana versi sederhana atau sketsa awal dari produk dibuat. Metode ini menghasilkan suatu model yang berfungsi atau sebuah representasi yang mencerminkan karakteristik inti, kemampuan, dan interaksi pengguna dari produk yang direncanakan.

Tujuan dari prototyping adalah memberikan cara untuk mengeksplorasi, menguji, dan memperbaiki ide-ide, desain, dan konsep sebelum berkomitmen dalam alokasi waktu dan sumber daya yang besar untuk pengembangan tahap lanjut. Melalui prototyping, para stakeholder diberi kesempatan untuk melihat dan berinteraksi dengan versi konkret atau interaktif dari produk, memberikan pandangan dan masukan yang bisa sangat berharga.

Dengan demikian, prototyping dapat menjadi alat penting dalam proses pengembangan perangkat lunak, membantu untuk mengurangi risiko, mengecek kelayakan, dan memastikan bahwa produk akhir akan memenuhi kebutuhan pengguna dan pemangku kepentingan.

## Github

GitHub merupakan layanan berbasis web yang menawarkan platform terpusat untuk kontrol versi dan kolaborasi dalam pengembangan perangkat lunak. Dengan beragam fitur dan alat yang tersedia, platform ini mendukung manajemen dan pertukaran kode, memfasilitasi kerja sama antar pengembang, pencatatan perubahan, dan kontribusi yang efisien ke dalam proyek.

## Privasi dan Proteksi Data Pengguna

Privasi dan perlindungan data pengguna merujuk pada prinsip, kebijakan, dan praktik yang bertujuan untuk melindungi kerahasiaan, integritas, dan ketersediaan data pengguna sambil menghormati hak privasi individu. Ini melibatkan penanganan, penyimpanan, dan penggunaan yang bertanggung jawab terhadap informasi pribadi dan sensitif yang dikumpulkan dari pengguna.

Privasi data pengguna berfokus pada memastikan bahwa individu memiliki kendali atas informasi pribadi mereka dan mengetahui bagaimana data mereka dikumpulkan, digunakan, dan dibagikan. Ini melibatkan memperoleh persetujuan yang diberikan secara sadar dari pengguna, memberikan pemberitahuan privasi yang jelas dan transparan, serta memberikan pengguna kemampuan untuk melaksanakan hak-hak mereka terkait data mereka.

Perlindungan data pengguna, di sisi lain, melibatkan penerapan langkah-langkah teknis, organisasional, dan hukum untuk melindungi data pengguna dari akses, penggunaan, pengungkapan, perubahan, atau penghancuran yang tidak sah. Ini meliputi praktik keamanan, seperti enkripsi, kontrol akses, anonimisasi data, audit keamanan berkala, dan prosedur respons terhadap pelanggaran, guna melindungi data pengguna dari ancaman eksternal dan penyalahgunaan internal.

## Security Testing

Security Testing merupakan salah satu aspek penting dalam siklus hidup pengembangan perangkat lunak (Software Development Life Cycle - SDLC). Security Testing adalah proses penilaian, pengujian dan penemuan kerentanan dalam sistem perangkat lunak untuk menentukan apakah data dan sumber daya sistem dilindungi dari potensi ancaman.

Secara umum, tujuan dari Security Testing adalah untuk mengidentifikasi semua kemungkinan kelemahan dan celah dalam sistem yang dapat dihasilkan oleh perangkat lunak yang tidak tepat atau cacat, atau ancaman yang dihasilkan oleh pihak luar. Proses ini dirancang untuk mengevaluasi efektivitas atau kekurangan pengendalian keamanan.

Terdapat beberapa jenis Security Testing, yang meliputi:

* *Intrusion Testing*: Ini melibatkan percobaan untuk merusak atau mengakses sistem secara ilegal.
* *Penetration Testing*: Ini adalah jenis pengujian intrusi yang sangat terstruktur dan melibatkan pencarian celah keamanan yang mungkin dimanfaatkan oleh penyerang.
* *Authentication Testing*: Ini melibatkan pengecekan sistem tentang bagaimana sistem memverifikasi identitas pengguna dan bagaimana sistem menangani otentikasi yang gagal.
* *Access Testing*: Ini adalah proses penilaian aturan akses sistem dan apakah mereka berfungsi seperti yang diharapkan.

Penting untuk memahami bahwa keberhasilan Security Testing sangat bergantung pada serangkaian variabel yang tidak hanya bergantung pada perangkat lunak yang digunakan, tetapi juga pada lingkungan sistem dan potensi ancaman.

## Database Providers

Database providers adalah komponen perangkat lunak yang bertindak sebagai penghubung antara aplikasi dan database, memfasilitasi interaksi antara keduanya. Dengan kata lain, database provider adalah lapisan antara aplikasi dan database yang mengkonversi perintah yang diberikan oleh aplikasi ke dalam bahasa yang dapat dimengerti dan dijalankan oleh database.

Database providers biasanya mendukung bahasa SQL (Structured Query Language), yang digunakan untuk membuat, memanipulasi, dan mengambil data dari database relasional. Namun, beberapa database provider juga mendukung database non-relasional (NoSQL) dan bahasa query khusus mereka.

## SDLC Waterfall

Model Waterfall adalah metode SDLC (Software Development Life Cycle) yang menekankan alur langkah-langkah yang logis sepanjang siklus hidup pengembangan perangkat lunak, mirip dengan aliran air terjun yang turun secara bertahap. Metodologi ini adalah proses desain yang sekuensial, di mana kemajuan dilihat sebagai mengalir secara stabil ke bawah (seperti air terjun).

Menurut Sommerville, model air terjun pengembangan perangkat lunak adalah model berurutan yang membagi pengembangan perangkat lunak menjadi fase-fase yang berbeda. Setiap fase harus diselesaikan sebelum fase berikutnya dapat dimulai. Fase-fase dari model air terjun yang diuraikan oleh Sommerville adalah:

1. *Requirements Analysis*, adalah layanan, batasan, dan tujuan sistem ditetapkan melalui konsultasi dengan pengguna sistem. Kemudian, hal-hal ini didefinisikan secara rinci.
2. *System and Software Design*, merupakan proses desain sistem mempartisi persyaratan ke sistem perangkat keras atau perangkat lunak. Ini membentuk arsitektur sistem secara keseluruhan. Desain perangkat lunak melibatkan mengidentifikasi dan menggambarkan abstraksi sistem perangkat lunak yang mendasar dan hubungan di antaranya.
3. *Implementation and Unit Testing*, merupakan tahap dimana desain perangkat lunak diwujudkan sebagai serangkaian program atau unit program. Pengujian unit melibatkan memverifikasi bahwa setiap unit memenuhi spesifikasinya.
4. *Integration and System Testing*, adalah tahap yang dimana unit program atau program individu diintegrasikan dan diuji sebagai sistem lengkap untuk memastikan bahwa persyaratan perangkat lunak telah terpenuhi. Setelah pengujian, sistem perangkat lunak diserahkan kepada pelanggan.
5. *Operation and Maintenance*, ini adalah fase siklus hidup terpanjang. Sistem diinstal dan digunakan secara praktis. Pemeliharaan melibatkan perbaikan kesalahan yang tidak ditemukan pada tahap-tahap sebelumnya dalam siklus hidup, meningkatkan implementasi unit sistem, dan meningkatkan layanan sistem seiring ditemukannya persyaratan baru.

# BAB 3

**METODE PENILITIAN**

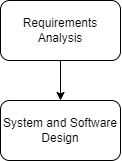
## Kerangka Berpikir

Kerangka berpikir yang diterapkan penulis selama proses pengembangan pustaka komponen untuk skripsi ini. Secara metodologis mengangkat tiga elemen penting yaitu, desain sistem, implementasi sistem, dan keamanan sistem.

* Desain Sistem

Desain sistem mengemuka sebagai tahapan fundamental dan elemen utama dalam kerangka berpikir ini. Konsep ini merujuk kepada proses rinci dari mendefinisikan arsitektur, modul, komponen, dan antarmuka dari pustaka autentikasi pengguna. Pendekatan terhadap desain sistem harus secara holistik, mempertimbangkan berbagai faktor seperti persyaratan fungsional dan non-fungsional, kebutuhan pengguna dan kenyamanan penggunaan, serta adaptasi teknologi yang relevan seperti Blazor WASM dan .NET Core.

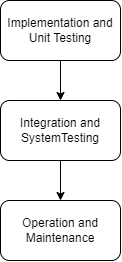
Fase desain melibatkan penelitian mendalam tentang praktek terbaik dalam desain sistem autentikasi, mencakup pengidentifikasian fitur-fitur kunci yang perlu diintegrasikan ke dalam sistem kita. Dalam fase desain sistem ini penulis membagi 2 tahapan utama yaitu, *requirement analysis* dan *system and software design.*



1. *Requirements Analysis*, pemahaman mengenai pengalaman pengguna juga menjadi elemen integral dalam fase desain, sehingga sistem yang dirancang dapat menawarkan antarmuka yang intuitif dan responsif.
2. *System and Software Design*, tahapan ini adalah hasil gambaran menyeluruh mengenai bagaimana setiap komponen dan modul saling berinteraksi dan bekerja sama untuk memberikan fungsionalitas yang diinginkan.

* Implementasi Sistem

Sebagai kelanjutan logis dari fase desain, proses bergerak menuju tahapan implementasi sistem. Pada fase ini penulis membagi menjadi 3 tahapan utama yaitu, *Implementation and Unit Testing*, *Integration and System Testing*, dan *Operation and Maintenance*



1. *Implementation and Unit Testing,* melibatkan pengubahan desain sistem yang telah dirumuskan menjadi realitas dalam bentuk baris-baris kode. Setiap elemen, mulai dari arsitektur hingga modul spesifik, perlu diterjemahkan ke dalam kode yang efisien, mudah dibaca, dan dapat dipelihara. Setiap modul atau elemen yang telah diimplementasi perlu dilakukan rangkaian testing sehingga menjamin bahwa setiap bagian kode mengikuti prinsip-prinsip pemrograman yang solid dan standar industri terkini.
2. *Integration and System Testing,* setelah fase implementasi, pustaka komponen perlu diintegrasikan ke dalam platform teknologi target yaitu Blazor WASM dan .NET Core yang kemudian dilakukan testing kembali untuk penyesuaiannya.
3. *Operation and Maintenance,* pada tahap ini pengujian dan pemeliharaan berkelanjutan harus berjalan secara paralel dengan berkembangnya teknologi autentikasi terbaru.

* Keamanan Sistem

Keamanan muncul sebagai elemen yang paling krusial dalam kerangka berpikir ini. Skripsi ini menempatkan keamanan sebagai prioritas utama dalam setiap tahapan desain dan implementasi sistem.

Konsep keamanan ini mencakup serangkaian tindakan preventif dan protektif, seperti pemilihan algoritma enkripsi yang kuat dan terpercaya, perlindungan terhadap serangan umum seperti injeksi kode dan serangan brute force, serta implementasi kontrol keamanan lainnya seperti autentikasi dua faktor. Sebelum sistem akhirnya diluncurkan, keamanan sistem harus diuji dan dievaluasi secara menyeluruh untuk memastikan semua standar dan protokol keamanan telah terpenuhi.

## Analisis Masalah

Dalam proses analisis masalah dalam pengembangan komponen pustaka ini, penulis melakukan analisis terhadap komponen pustaka serupa yang ada. Penelitian ini bertujuan untuk mengeksplorasi pengalaman dan solusi yang telah diterapkan dalam konteks yang serupa. Dengan mempelajari komponen pustaka serupa, peneliti dapat mengidentifikasi masalah yang umum terjadi, hambatan yang dihadapi, serta solusi yang telah sukses diterapkan. Informasi ini sangat berharga dalam memahami panorama masalah dan mencari pendekatan yang efektif dalam pengembangan aplikasi yang sedang dirancang.

Selain itu, penulis menggunakan metode analisis wawancara untuk mengetahui kebutuhan pengguna dari berbagai sumber, sehingga informasi hasil wawancara dapat menjadi referensi dalam merancang dan mengimplementasikan pustaka komponen autentikasi pengguna yang aman. Wawancara dilakukan kepada tiga profesi yang berbeda, yang masing-masing memiliki perspektif dalam pengembangan dan penggunaan sistem autentikasi.

### **Permasalahan dan Kebutuhan**

Pada aplikasi autentikasi pengguna, harus ada dasar pembuatan yang merujuk kepada kebutuhan dan tantangan yang dihadapi dalam bidang keamanan siber, terutama dalam hal autentikasi pengguna. Dengan adanya pustaka komponen autentikasi ini, pengembang dapat lebih berfokus pada aspek teknikal dari aplikasi yang mereka bangun, dan tidak perlu membuang banyak waktu dan sumber daya untuk mengimplementasikan sistem autentikasi dari awal.

Aplikasi ini bertujuan untuk mencapai goal utama yaitu sebagai sarana otomatisasi dalam pembuatan sistem autentikasi, membantu kolaborasi antara pengembang dengan sistem atau aplikasi yang mereka buat, serta membantu proses transfer pengetahuan antara pengembang.

Cakupan dari fungsi-fungsi yang akan dibuat dalam pustaka ini harus mencakup enkripsi dan verifikasi data pengguna, manajemen sesi pengguna, dan dukungan untuk berbagai jenis metode autentikasi. Fitur-fitur ini diprioritaskan karena selain berdasarkan rancangan awal yang sudah dibuat, beberapa fitur juga ditambahkan berdasarkan hasil riset dan wawancara yang dilakukan dalam proses pengumpulan data.

Cakupan fungsi yang dipilih juga disederhanakan dengan tujuan untuk membangun pustaka yang mudah dipahami dan mudah digunakan oleh pengembang. Dengan demikian, pengembang dapat lebih mudah dan cepat dalam mengimplementasikan sistem autentikasi yang aman dan handal pada aplikasi mereka.

### **Analisis Komponen Pustaka Serupa**

* Passport.Js

Passport.js adalah pustaka autentikasi yang populer dan banyak digunakan untuk aplikasi Node.js. Passport.js adalah salah satu pustaka komponen *middleware* yang paling terkenal di Node.Js dengan 21.500 stars dan 1.200 forks di GitHub. Passport.js dikenal karena fleksibilitas dan modularitasnya berkat strateginya. Keragaman lebih dari 500 strategi autentikasi, termasuk autentikasi biasa (nama pengguna dan kata sandi), OAuth, JWT, dan SAML, menyediakan berbagai pilihan yang dapat memenuhi sebagian besar kebutuhan autentikasi.

Passport menyediakan kerangka kerja yang fleksibel yang memungkinkan aplikasi untuk menggunakan mekanisme otentikasi apa pun. Passport mengurangi kompleksitas dalam melakukan *request* otentikasi menjadi *coding* yang sederhana.



* Auth0

Auth0 menawarkan berbagai fitur yang penting untuk manajemen identitas, termasuk otentikasi sosial, pengelolaan pengguna dan grup, serta integrasi dengan berbagai layanan pihak ketiga. Dengan Auth0, pengembang dapat dengan mudah mengintegrasikan sistem autentikasi yang aman dalam aplikasi mereka tanpa perlu membangun sistem dari awal. Selain itu, Auth0 juga menggunakan JSON Web Tokens (JWT) dalam sistem autentikasinya, yang merupakan teknologi yang sudah diakui keamanannya dan bisa menjadi pertimbangan dalam merancang pustaka komponen autentikasi.

Namun, penting untuk dicatat bahwa Auth0 adalah platform yang berbayar dan bisa menjadi solusi yang mahal bagi beberapa aplikasi atau perusahaan.

### **Analisis Berdasarkan Wawancara**

Penulis melakukan wawancara kepada satu orang manajer proyek perangkat lunak, yang memiliki pengalaman dalam merencanakan dan mengelola pengembangan aplikasi, termasuk sistem autentikasi. Manajer proyek ini dapat memberikan informasi tentang apa saja tantangan dan hambatan yang sering dihadapi dalam proses pengembangan dan implementasi sistem autentikasi.

Selanjutnya, penulis juga melakukan wawancara kepada satu orang pengembang junior yang memiliki pengalaman dalam menggunakan dan mengimplementasikan pustaka autentikasi. Dari pengembang ini, penulis dapat mengetahui apa saja kesulitan dan hambatan yang sering dihadapi dalam proses penggunaan dan implementasi pustaka autentikasi, serta apa saja fitur dan kemampuan yang diharapkan dari pustaka tersebut.

Akhirnya, penulis juga melakukan wawancara kepada satu orang pen tester junior, yang memiliki pengetahuan dan pengalaman dalam melakukan pengetesan keamanan terhadap sistem autentikasi. Dari pen tester ini, penulis dapat mengetahui apa saja celah keamanan dan kerentanan yang sering ditemukan dalam sistem autentikasi, serta apa saja strategi dan teknik yang dapat digunakan untuk mencegah dan mengatasi celah-celah tersebut.

* + - * Project Manager

Pertanyaan pertama yang diajukan oleh penulis adalah mengenai framework yang digunakan narasumber untuk projek-projeknya serta mengapa narasumber memilih framework tersebut. Narasumber menggunakan framework Angular dan .NET untuk melakukan development aplikasi, adapun alasan narasumber memilih framework tersebut dikarenakan framework-framework tersebut umum digunakan di kalangan developer sehingga banyak informasi-informasi mengenai framework tersebut dan juga banyak tools-tools seperti debugging tools serta *library* yang memudahkan developer untuk melakukan pengembangan aplikasi.

Pertanyaan kedua yang diajukan oleh penulis untuk mengetahui peran library dalam fase pengembangan suatu aplikasi. Narasumber menjawab library dapat mempersingkat waktu suatu tugas development yang cukup rumit. Akan tetapi developer perlu memahami terlebih dahulu library-library yang digunakan agar tugas development dapat dikerjakan dengan cepat dan mudah.

Pertanyaan ketiga yang diajukan penulis adalah mengenai bagaimana perbedaan bahasa pemrograman dalam framework backend dan framework frontend dapat mempengaruhi pengembangan aplikasi. Narasumber menjawab dengan adanya perbedaan bahasa pemrograman backend dan frontend maka diperlukan sumber daya manusia untuk mengerjakan backend dan frontend.

Pertanyaan keempat yang diajukan penulis adalah apabila framework backend dan frontend yang digunakan menggunakan bahasa yang sama, bagaimana hal tersebut mempengaruhi sebuah pengembangan aplikasi. Narasumber menjawab bahwa hal tersebut tentu dapat mempengaruhi suatu projek pengembangan aplikasi, mulai dari sumber daya manusia yang dibutuhkan lebih sedikit karena seseorang developer hanya perlu menguasai satu bahasa pemrograman dan juga hal ini dapat mempermudah project manajer untuk melakukan perputaran developer dalam suatu projek.

Pertanyaan kelima adalah mengenai pentingnya aspek keamanan dalam suatu aplikasi. Narasumber menjelaskan bahwa projek yang narasumber kerjakan dan pernah kerjakan sudah berskala *enterprise* dan berkaitan dengan jasa finansialyang berarti aspek keamanan merupakan suatu hal yang sangat penting dalam pengembangan aplikasi. Dalam pengembangannya sendiri pasti akan ada tahap penetration testing. Penetration testing tersebut dapat dilakukan dari pihak internal maupun eksternal.

Pertanyaan keenam yang diajukan penulis adalah untuk mengetahui proses autentikasi pada aplikasi yang narasumber pernah kerjakan dan untuk mengetahui juga apakah proses tersebut menggunakan autentikasi. Narasumber menjelaskan bahwa terdapat proses autentikasi dalam aplikasinya dan dalam pengembangan proses autentikasi pada aplikasi tersebut menggunakan library yang telah dikembangkan sendiri dari pihak internal. Adapun alasan menggunakan library pihak internal karena lebih terjamin keamanannya meskipun tidak begitu fleksibel untuk digunakan.

Pertanyaan ketujuh yang diajukan penulis adalah untuk mengetahui kemungkinan penggunaan library autentikasi dari pihak eksternal dan apabila ada kriteria tertentu agar suatu library eksternal dapat digunakan. Narasumber menjawab bahwa memungkinkan untuk menggunakan library autentikasi dari pihak eksternal. Namun harus ada pertimbangannya, mulai dari standar keamanannya, efektivitasnya dalam implementasi, serta kompleksitas pemakaiannya.

* + - * Junior Pentester

Pertanyaan pertama yang diajukan penulis adalah untuk mengetahui relevansi autentikasi dalam keamanan sebuah aplikasi berbasis web. Narasumber menjawab bahwa autentikasi merupakan salah satu komponen terpenting dalam keamanan suatu aplikasi. Hal ini dikarenakan autentikasi merupakan pintu utama suatu user dapat mengakses layanan-layanan yang terdapat pada aplikasi tersebut. Dengan adanya autentikasi, user dapat diidentifikasi dan diverifikasi sehingga user tersebut dapat mengakses data-datanya pada aplikasi tersebut. Selain itu, dengan adanya autentikasi pengguna akan diberi rasa keamanan karena tidak sembarang orang bisa mengakses data-data pribadinya.

Pertanyaan kedua yang diajukan penulis adalah untuk mengetahui protokol keamanan yang biasa digunakan untuk melakukan autentikasi. Narasumber memberi beberapa protokol keamanan diantaranya adalah OAuth 2.0, OpenId Connect, dan JSON Web Token.

Pertanyaan ketiga yang diajukan penulis adalah mengenai cara pengujian pustaka komponen autentikasi pengguna untuk memastikan keamanan dan efektivitasnya. Untuk pengujian pustaka komponen autentikasi, narasumber menjawab terdapat beberapa hal yang perlu dilakukan pengujian diantaranya, pengujian pustaka komponen itu sendiri, pengujian pada saat melakukan implementasi ke aplikasi, pengujian saat integrasi ke *database,* dan yang terakhir adalah pengujian fungsionalitasnya itu sendiri.

Pertanyaan keempat yang diajukan penulis adalah untuk mengetahui hal yang harus diperhatikan dalam pembaruan dan pemeliharaan library autentikasi dalam konteks keamanan kedepannya. Narasumber menjawab bahwa library harus di-patching terus menerus sesuai dengan kebutuhan developer dan standar keamanan autentikasi yang paling terbaru.

* + - * Junior Developer

Pertanyaan pertama yang diajukan penulis adalah untuk mengetahui pengalaman junior developer dalam penggunaan mekanisme autentikasi pada aplikasi yang telah atau sedang dikerjakannya. Narasumber menjawab aplikasi yang telah dikembangkannya menggunakan mekanisme autentikasi dan juga otorisasi, yang dimana tiap *API* pada *backend* memerlukan sebuah key untuk diakses dan untuk mendapatkan key tersebut diperlukan proses autentikasi terlebih dahulu. Akan dari segi pengkodingannya narasumber belum terlalu memahami cara kerjanya.

Pertanyaan kedua yang diajukan penulis adalah mengenai manfaat utama dalam penggunaan library dalam pengembangan suatu aplikasi. Manfaat yang didapatkan oleh narasumber dengan menggunakan library adalah efesiensi waktu dalam pengerjaan sebuah library yang seharusnya sebuah tugas memakan waktu sehari menjadi hanya puluhan menit saja dan juga mempermudah tugas pengembangan yang kompleks.

Pertanyaan ketiga yang diajukan penulis adalah bagaimana developer dapat menentukan waktu yang tepat untuk menggunakan suatu library atau tidak. Narasumber menjelaskan bahwa penggunaan library yang tepat harus disesuaikan dengan tugas pengembangan dan juga library yang akan digunakan. Apabila ada suatu library yang memiliki fungsionalitas yang sama sesuai dengan apa yang ingin dikembangkan maka lebih baik menggunakan library. Akan tetapi, library harus mudah digunakan dan tidak memakan waktu yang banyak untuk mempelajari menggunakannya.

Pertanyaan keempat yang diajukan penulis adalah kriteria library yang akan digunakan oleh developer saat akan melakukan tugas pengembangannya. Narasumber menjelaskan bahwa kriteria library yang biasa digunakan adalah library yang memiliki fungsionalitas sesuai dengan tugas pengembangannya. Selain itu, library tersebut sebaiknya memiliki pengguna yang banyak sebab apabila ada suatu bug terkait library tersebut maka mudah untuk menemukan penyelesaiannya di internet. Yang terakhir adalah dokumentasi, setiap library wajib ada dokumentasi, hal ini mempermudah developer dalam mengimplementasikan library ke suatu sistem.

### **Usulan Pemecahan Masalah**

Dari serangkaian permasalahan dan kebutuhan yang telah diidentifikasi, tim penulis merumuskan serangkaian solusi untuk mengatasi masalah tersebut. Mengenai tantangan dalam autentikasi pengguna di aplikasi Blazor dan .NET, penulis merancang ide untuk membuat pustaka komponen otentikasi yang aman dan dapat digunakan secara efisien.

Untuk mengatasi permasalahan kerumitan dalam proses otentikasi, pustaka yang dirancang akan menyediakan komponen yang dapat digunakan kembali untuk mengautentikasi pengguna dalam berbagai konteks, dengan menjamin keamanan dan integritas data pengguna.

Untuk isu terkait kesulitan dalam implementasi fitur autentikasi, pustaka ini diharapkan dapat memfasilitasi proses implementasi dengan menyediakan komponen otentikasi yang sudah siap pakai dan mudah disesuaikan dengan kebutuhan pengembangan aplikasi Blazor dan .NET.

Untuk masalah transfer pengetahuan antar developer, pustaka ini akan dilengkapi dengan dokumentasi yang lengkap dan jelas, sehingga developer lain yang ingin menggunakan pustaka ini dapat dengan mudah memahami cara kerja dan metode implementasinya.

Berikut adalah perbandingan antara proses autentikasi pengguna sebelum dan sesudah penggunaan pustaka komponen autentikasi yang dirancang.

## Perancangan Sistem

1. Kebutuhan Teknologi
   1. Framework

Framework yang digunakan untuk pengembangan pustaka komponen adalah sebagai berikut

* .NET Core

*Back-end* aplikasi dikembangkan menggunakan framework .NET Core dengan bahasa pemrograman C#

* Blazor WASM

*Front-end* aplikasi dikembangkan menggunakan framework .NET Core dengan bahasa pemrograman C#

* 1. Basis Data

Database yang digunakan untuk pengembangan serta pengujian pustaka komponen adalah MariaDB. MariaDB digunakan sebagai tempat menyimpan data-data terkait kredensial user.

1. Aplikasi Pendukung
   1. Visual Studio

Visual Studio digunakan sebagai code editor untuk

membuat kodingan front-end dan backend

* 1. Draw.io

Draw.io merupakan situs web untuk membuat diagram dan

penulis gunakan untuk membuat diagram seperti Activity Diagram,

Data Flow Diagram, dan diagram-diagram lainnya

* 1. Github

Github digunakan sebagai code versioning tools untuk melakukan kolaborasi dalam pengembangan pustaka komponen ini.

1. Diagram *Unified Modeling Language*
   * 1. Activity Diagram

A picture containing diagram, text, plan, technical drawing

Description automatically generated A picture containing diagram, text, plan, technical drawing

Description automatically generated A diagram of a system

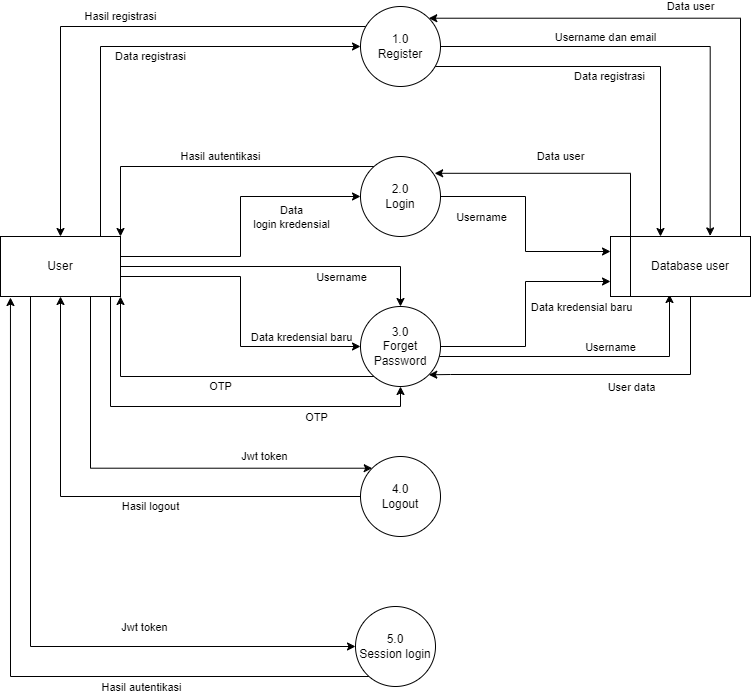
Description automatically generated with low confidence A picture containing text, screenshot, diagram, line

Description automatically generated

A picture containing text, diagram, sketch, drawing

Description automatically generated

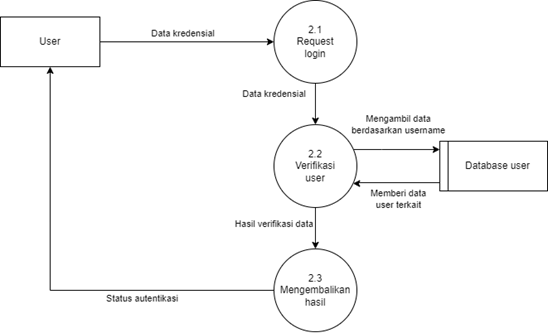
* + 1. Data Flow Diagram



A picture containing diagram, text, circle, screenshot

Description automatically generated A picture containing diagram, text, sketch, drawing

Description automatically generated



A diagram of a flowchart

Description automatically generated with low confidence

# BAB 4

**HASIL DAN PEMBAHASAN**

# Spesifikasi Sistem

Untuk menggunakan pustaka komponen autentikasi secara optimal, diperlukan perangkat komputer yang memenuhi persyaratan sistem agar tidak terjadi masalah teknis. Di bawah ini adalah spesifikasi yang diperlukan untuk menggunakan pustaka komponen autentikasi Blazor WASM dan .NET.

* + 1. **Spesifikasi Software**

Spesifikasi software yang diperlukan untuk mengimplementasikan pustaka komponen autentikasi Blazor WASM dan .NET.

* 1. Operating System: Windows 10
  2. Web Browser: Google Chrome, Mozilla Firefox, Edge
  3. Database: Postgresql, Mysql, MariaDB, MongoDB, MSSQL
  4. Framework: Blazor WASM dan .NET 6
     1. **Spesifikasi Hardware**

Spesifikasi software yang diperlukan untuk mengimplementasikan pustaka komponen autentikasi Blazor WASM dan .NET.

1. Processor: Intel/AMD
2. RAM: 8 Gigabyte

# Prosedur Penggunaan Pustaka Komponen

### **Prosedur Penggunaan Pustaka Komponen Blazor (Front End)**

### Tambahkan *reference* ke *file* SeComps.dll dan CommonModelsLib.dll ke projek Blazor WASM anda.

### A screenshot of a computer Description automatically generated

### A black background with white text Description automatically generated with low confidence

### Tambahkan Komponen **Login** dengan parameter **DivClass** dengan CSS yang diinginkan dan **ForgotPasswordRef** dengan halaman *Forgot Password* anda.

### 

### Hasil yang dihasilkan akan menjadi seperti ini.

### 

### Tambahkan Komponen **RegisterModular** dengan parameter **DivClass** dengan CSS yang diinginkan dan **TitleText** dengan halaman judul yang anda inginkan.

### 

### Hasil yang didapatkan akan seperti berikut:

### A screenshot of a login form Description automatically generated with medium confidence

### Tambahkan Komponen **ResetPassword** dengan parameter **DivClass** dengan CSS yang diinginkan dan **TitleText** dengan halaman judul yang anda inginkan.

### 

### Hasil yang didapatkan akan sebagai berikut:

### A screenshot of a login form Description automatically generated with medium confidence

### **Prosedur Penggunaan Pustaka Komponen .NET Core (*Back End*)**

### Tambahkan *reference* ke *file* SecureBackendAuthorizer.dll dan CommonModelsLib.dll ke projek Blazor WASM anda.

### A screenshot of a computer Description automatically generated

### A screenshot of a computer Description automatically generated with medium confidence

### Di Appsetings.json program anda, tambahkan informasi berikut dengan penyesuaian kredensial anda.

### A screenshot of a computer screen Description automatically generated with low confidence

* + - 1. Di Program.cs, tambahkan kode-kode berikut dengan penyesuian pada penyedia basis data yang anda gunakan (contoh disini menggunakan MariaDB)

A picture containing text, screenshot, font

Description automatically generated

* + - 1. Pustaka Komponen ini akan secara otomatis membuat *Endpoint* berbasis JSON seperti berikut:

A picture containing text, screenshot, number, font

Description automatically generated A screenshot of a computer

Description automatically generated with medium confidence

# Evaluasi Sistem

* + 1. Security Testing

1. Testing for Credentials Transported over an Encrypted Channel

Pengujian untuk transportasi kredensial melalui saluran yang terenkripsi telah memberikan hasil yang baik. Implementasi yang digunakan dalam pustaka ini mengutamakan keamanan dan privasi pengguna dengan menjamin bahwa setiap kali kredensial pengguna dikirim antara client dan server, mereka dienkripsi dengan standar yang kuat dan teknologi enkripsi modern.

Protokol HTTPS digunakan untuk semua komunikasi antara klien dan server, memastikan bahwa data dikirim dalam saluran yang aman dan terenkripsi. Hal ini mencegah potensi "man-in-the-middle" atau serangan pengintipan yang dapat mencoba untuk mencuri atau memanipulasi data selama transportasi. Selain itu, pustaka komponen autentikasi ini juga mengimplementasikan mekanisme keamanan tambahan seperti HSTS (HTTP Strict Transport Security) yang memaksa seluruh komunikasi untuk dilakukan melalui HTTPS.

Dalam melakukan pengujian, kredensial pengguna seperti username dan password dikirimkan melalui saluran yang terenkripsi dan perjalanan data tersebut dipantau. Tidak ada bukti pelemahan enkripsi atau kebocoran data yang terdeteksi selama proses pengujian ini. Ini menunjukkan bahwa pustaka komponen autentikasi yang dibangun telah berhasil dalam memastikan bahwa kredensial pengguna yang sangat penting dikirimkan dalam lingkungan yang aman dan terlindungi. Oleh karena itu, dapat disimpulkan bahwa dalam aspek pengiriman kredensial melalui saluran yang terenkripsi, pustaka ini telah memenuhi standar keamanan yang ditetapkan.

1. Testing for Default Credentials

Pengujian terhadap kredensial default telah menunjukkan hasil yang positif. Sistem yang dikembangkan melalui penelitian ini dirancang untuk memaksa setiap pengguna, termasuk admin, untuk menentukan kredensial mereka sendiri saat pertama kali melakukan setup atau mengkonfigurasi sistem. Setiap pengguna diharuskan untuk memilih username dan password mereka sendiri, yang harus memenuhi kriteria keamanan yang kuat, termasuk campuran huruf, angka, dan simbol.

Selama proses pengujian, setiap upaya untuk menggunakan kredensial default yang umum (seperti "admin" / "admin" atau "user" / "password") untuk mengakses sistem selalu mengalami kegagalan. Hal ini mengkonfirmasi bahwa pustaka autentikasi ini telah berhasil mencegah penggunaan kredensial default yang dapat menjadi celah keamanan yang signifikan.

Oleh karena itu, berdasarkan hasil pengujian, dapat disimpulkan bahwa dalam hal mencegah penggunaan kredensial default, pustaka komponen autentikasi yang dibangun dalam penelitian ini telah berhasil memenuhi tujuan utamanya yaitu memberikan tingkat keamanan yang tinggi untuk pengguna Blazor WASM dan .NET Core.

1. Testing for Weak Lock Out Mechanism

Untuk mencegah serangan ini, pustaka komponen autentikasi yang telah dikembangkan mengimplementasikan mekanisme penguncian yang kuat. Setelah sejumlah upaya login yang gagal yang telah ditentukan sebelumnya, akun pengguna akan terkunci secara otomatis. Pengguna kemudian akan perlu melakukan reset kata sandi melalui alamat email yang terdaftar atau menghubungi administrator untuk membuka kunci akun mereka.

Selama proses pengujian, upaya untuk melakukan login yang gagal berulang kali diakibatkan dengan penguncian akun, sehingga membuktikan bahwa mekanisme penguncian berfungsi dengan efektif. Mekanisme ini telah berhasil mencegah upaya akses yang tidak sah, dan oleh karena itu, telah meningkatkan keamanan secara keseluruhan dari pustaka autentikasi pengguna.

Berdasarkan hasil pengujian ini, dapat disimpulkan bahwa dalam aspek melindungi dari mekanisme penguncian yang lemah, pustaka komponen autentikasi yang dikembangkan dalam penelitian ini telah berhasil memenuhi tujuan utamanya untuk memberikan solusi keamanan yang andal dan kuat bagi pengguna Blazor WASM dan .NET Core.

1. Testing for Bypassing Authentication Schema

Penulis telah melakukan pengujian untuk kemungkinan pengabaian skema autentikasi. Tujuan dari pengujian ini adalah untuk memastikan bahwa skema autentikasi yang dibangun dalam pustaka ini tahan terhadap upaya bypass dan menjaga integritas proses autentikasi.

Dalam pustaka komponen autentikasi yang telah dikembangkan, setiap permintaan ke server yang memerlukan autentikasi harus disertai dengan token autentikasi yang valid. Token ini dikeluarkan oleh server saat pengguna berhasil login, dan harus digunakan untuk setiap permintaan selanjutnya. Jika token ini tidak ada atau tidak valid, permintaan akan ditolak oleh server.

Selama proses pengujian, upaya untuk mengakses sumber daya yang dijaga tanpa token autentikasi atau dengan token yang tidak valid selalu ditolak oleh server. Selain itu, upaya untuk mengubah token juga tidak berhasil, karena setiap token dikaitkan dengan pengguna tertentu dan tidak dapat digunakan oleh pengguna lain.

Berdasarkan hasil pengujian, dapat disimpulkan bahwa dalam aspek perlindungan terhadap upaya bypass skema autentikasi, pustaka komponen autentikasi yang dikembangkan dalam penelitian ini telah berhasil memenuhi tujuannya. Ini menunjukkan bahwa solusi yang dirancang dan diimplementasikan dalam penelitian ini mampu memberikan tingkat keamanan yang tinggi dan memastikan proses autentikasi yang benar dan aman untuk pengguna Blazor WASM dan .NET Core.

1. Testing for Browser Cache Weakness

Proses pengujian untuk kelemahan cache browser telah dilakukan. Tujuan dari pengujian ini adalah untuk memastikan bahwa informasi pengguna dan data sensitif lainnya tidak tersimpan dalam cache browser, yang bisa diakses oleh pihak ketiga yang tidak berwenang.

Pustaka autentikasi yang dikembangkan untuk penelitian ini telah dirancang dengan pertimbangan keamanan yang ketat. Hal ini mencakup penerapan pengaturan kebijakan cache yang tepat untuk mencegah penyimpanan data sensitif di cache browser. Misalnya, penggunaan header HTTP seperti "Cache-Control: no-store, no-cache, must-revalidate", yang menginstruksikan browser untuk tidak menyimpan respons dari server.

Selama proses pengujian, cache browser diperiksa setelah proses autentikasi dan selama sesi pengguna. Tidak ada bukti dari informasi autentikasi pengguna atau data sensitif lainnya yang tersimpan dalam cache. Ini membuktikan bahwa mekanisme proteksi yang telah diterapkan dalam pustaka efektif dalam mencegah pengeksposan data melalui kelemahan cache browser.

Dengan demikian, berdasarkan hasil pengujian, dalam aspek perlindungan terhadap kelemahan cache browser, pustaka komponen autentikasi yang telah dikembangkan dalam penelitian ini telah berhasil memenuhi tujuan utamanya untuk memberikan tingkat keamanan yang kuat bagi pengguna Blazor WASM dan .NET Core. Ini menunjukkan efektivitas dan keandalan solusi yang dirancang dan diimplementasikan dalam penelitian ini dalam mengatasi tantangan keamanan dalam pengembangan aplikasi web modern.

1. Testing for Weak Password Policy

Pengujian untuk kebijakan kata sandi yang lemah telah dilakukan. Tujuan dari pengujian ini adalah untuk memastikan bahwa pustaka autentikasi yang dibangun memaksakan kebijakan kata sandi yang kuat, yang penting untuk melindungi akun pengguna dari serangan bruteforce atau penggunaan kata sandi yang dapat ditebak.

Pustaka autentikasi yang dikembangkan dalam penelitian ini mengimplementasikan kebijakan kata sandi yang kuat. Ini mencakup persyaratan minimal panjang kata sandi, serta persyaratan untuk mencakup kombinasi dari huruf besar, huruf kecil, angka, dan karakter khusus. Hal ini membuat kata sandi lebih sulit untuk ditebak atau ditembus melalui metode bruteforce.

Selama proses pengujian, upaya untuk mendaftar atau mengubah kata sandi ke kata sandi yang lemah (seperti "123456", "password", atau "qwerty") selalu ditolak oleh sistem. Ini menunjukkan bahwa pustaka telah berhasil dalam menerapkan dan memaksakan kebijakan kata sandi yang kuat.

Oleh karena itu, berdasarkan hasil pengujian, dalam hal mencegah kebijakan kata sandi yang lemah, pustaka komponen autentikasi yang dikembangkan dalam penelitian ini telah berhasil memenuhi tujuan utamanya, yaitu memberikan keamanan yang tinggi bagi pengguna Blazor WASM dan .NET Core melalui penerapan kebijakan kata sandi yang kuat.

1. Testing for Weak Password Change or Reset Functionalities

Penulis telah melakukan pengujian terhadap fungsi perubahan atau reset kata sandi yang lemah. Tujuan dari pengujian ini adalah untuk memastikan bahwa fungsi perubahan atau reset kata sandi yang dibangun dalam pustaka ini dirancang dengan baik dan tidak dapat disalahgunakan oleh pihak ketiga yang tidak berwenang.

Pustaka autentikasi yang dikembangkan dalam penelitian ini memiliki fungsi perubahan atau reset kata sandi yang kuat. Ketika pengguna ingin mengubah atau mereset kata sandi mereka, mereka diharuskan untuk memverifikasi identitas mereka, misalnya melalui verifikasi OTP. Selain itu, kebijakan kata sandi yang kuat juga diterapkan saat mereset atau mengubah kata sandi, untuk memastikan bahwa kata sandi baru yang dipilih oleh pengguna juga aman.

Selama proses pengujian, upaya untuk mengubah atau mereset kata sandi tanpa verifikasi yang tepat selalu gagal. Selain itu, upaya untuk mengubah atau mereset kata sandi ke kata sandi yang lemah juga ditolak oleh sistem. Hal ini membuktikan bahwa pustaka telah berhasil dalam menerapkan fungsi perubahan atau reset kata sandi yang kuat dan aman.

Berdasarkan hasil pengujian, dalam aspek melindungi dari fungsi perubahan atau reset kata sandi yang lemah, pustaka komponen autentikasi yang dikembangkan dalam penelitian ini telah berhasil memenuhi tujuannya. Ini menunjukkan bahwa solusi yang dirancang dan diimplementasikan dalam penelitian ini mampu memberikan tingkat keamanan yang tinggi dan menjaga integritas proses autentikasi untuk pengguna Blazor WASM dan .NET Core.

1. Testing for Weaker Authentication in Alternative Channel

Penulis telah dilakukan pengujian terhadap autentikasi yang lebih lemah dalam saluran alternatif. Tujuan pengujian ini adalah untuk memastikan bahwa pustaka autentikasi ini mampu memberikan tingkat keamanan yang konsisten di semua saluran, baik itu saluran utama atau alternatif.

Dalam penelitian ini, pustaka autentikasi telah dirancang untuk memaksakan standar keamanan yang sama di semua saluran. Baik itu autentikasi melalui aplikasi web, API, atau melalui saluran alternatif lainnya, pengguna diharuskan untuk memberikan kredensial yang valid dan memenuhi kebijakan keamanan yang telah ditetapkan.

Selama proses pengujian, upaya untuk melakukan autentikasi melalui saluran alternatif dengan kredensial yang lemah atau tidak valid selalu ditolak oleh sistem. Ini menunjukkan bahwa pustaka telah berhasil dalam menerapkan kebijakan keamanan yang konsisten di semua saluran.

Oleh karena itu, berdasarkan hasil pengujian, dalam aspek mencegah autentikasi yang lebih lemah di saluran alternatif, pustaka komponen autentikasi yang dikembangkan dalam penelitian ini telah berhasil memenuhi tujuan utamanya, yaitu memberikan keamanan yang tinggi dan konsisten bagi pengguna Blazor WASM dan .NET Core di semua saluran. Ini menunjukkan efektivitas dan keandalan solusi yang dirancang dan diimplementasikan dalam penelitian ini dalam mengatasi tantangan keamanan dalam pengembangan aplikasi web modern

1. Testing for Multi-Factor Authentication

Proses pengujian untuk autentikasi multi-faktor telah dilakukan. Tujuan dari pengujian ini adalah untuk memastikan bahwa pustaka autentikasi yang dikembangkan mampu mendukung dan memaksakan autentikasi multi-faktor, yang merupakan praktek terbaik dalam keamanan siber dan memberikan lapisan keamanan tambahan untuk akun pengguna.

Pustaka autentikasi yang telah dikembangkan dalam penelitian ini telah berhasil mendukung autentikasi multi-faktor. Selain membutuhkan kata sandi yang aman, pengguna juga diminta untuk memverifikasi identitas mereka melalui faktor kedua, seperti kode OTP yang dikirim ke email mereka atau aplikasi autentikasi.

Selama proses pengujian, upaya untuk mengakses akun tanpa memasukkan faktor kedua autentikasi selalu gagal. Ini membuktikan bahwa pustaka telah berhasil dalam menerapkan dan memaksakan autentikasi multi-faktor.

Dengan demikian, berdasarkan hasil pengujian, dalam hal mendukung dan menerapkan autentikasi multi-faktor, pustaka komponen autentikasi yang dikembangkan dalam penelitian ini telah berhasil memenuhi tujuan utamanya. Hasil ini menunjukkan bahwa solusi yang dirancang dan diimplementasikan dalam penelitian ini mampu memberikan tingkat keamanan yang tinggi dan memastikan proses autentikasi yang aman bagi pengguna Blazor WASM dan .NET Core.

### **Black-box Testing**

Berikut tabel hasil *black-box* testing pada pustaka komponen autentikasi:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Skenario Pengujian | Proses | Hasil yang Diharapkan | Hasil yang Didapatakan | Kesimpulan |
| Register (Positive Test) | User menginput  data yaitu  username, email, dan password | User berhasil mendaftarkan akun dan data tersimpan di *database* | User berhasil mendaftarkan akun dan data tersimpan di *database* | Sukses |
| Register (Negative Test) | User menginput  data yaitu  username atau email yang sudah dipakai | User mendapatkan notifikasi username atau email telah digunakan | User mendapatkan notifikasi username atau email telah digunakan | Sukses |
| Login (Positive Test) | User menginput  data yaitu  username dan password | User berhasil masuk ke akun dan diarahkan ke halaman yang telah ditentukan | User berhasil masuk ke akun dan diarahkan ke halaman yang telah ditentukan | Sukses |
| Login (Negative Test) | User menginput  data yaitu  username dan password  yang tidak sesuai | User gagal masuk ke akun dan dapat notifikasi salah password atau username | User gagal masuk ke akun dan dapat notifikasi salah password atau username | Sukses |
| Forgot Password Generate OTP (Positive Test) | User menginput email dan menekan tombol submit. | User menerima otp diemailnya | User menerima otp diemailnya | Sukses |
| Forgot Password Generate OTP (Negative Test) | User menginput email yang tidak sesuai dan menekan tombol submit | Terdapat notifikasi email tidak ditemukan | Terdapat notifikasi email tidak ditemukan | Sukses |
| Forgot Password Submit OTP (Positive Test) | User mengisi OTP | User memasuki ke halaman ganti password | User memasuki ke halaman ganti password | Sukses |
| Forgot Password Submit OTP (Positive Test) | User mengisi OTP yang salah | Terdapat notifikasi otp salah | Terdapat notifikasi otp salah | Sukses |
| Forgot Password Change Password (Positive Test) | User mengisi password baru | User berhasil mengganti passwordnya | User berhasil mengganti passwordnya | Sukses |
| Forgot Password Change Password (Negative Test) | User mengisi password baru dengan format password yang sama dengan sebelumnya | Terdapat notifikasi password pernah digunakan sebelumnya | Terdapat notifikasi password pernah digunakan sebelumnya | Sukses |
| Logout (Positive Test) | User menekan button logout | User keluar dari akun | User keluar dari akun | Sukses |
| Logout (Negative Test) | User tidak menekan button logout | User tidak keluar dari akun | User tidak keluar dari akun | Sukses |
| Login with token (Positive Test) | User melakukan login dengan token | User berhasil masuk ke akun dan diarahkan ke halaman yang telah ditentukan | User berhasil masuk ke akun dan diarahkan ke halaman yang telah ditentukan | Sukses |
| Login with token (Negative Test) | User melakukan login dengan token yang telah expired | User gagal masuk ke akun | User gagal masuk ke akun | Sukses |

# BAB 5

# REFERENCES