

Implementasi Single Sign-On Menggunakan Protokol Openid Connect (OIDC) Pada Virtual Private Server (VPS)

Muhammad Aditya Sahrin, Rudi Heriansyah*, Dewi Sartika

Program Studi Teknik Informatika, Universitas Indo Global Mandiri Penulis

Korespondensi: rudi@uigm.ac.id

Adityashrn0052@gmail.com, rudi@uigm.ac.id, dewi@uigm.ac.id

Abstract

Single Sign-On (SSO) is an authentication method that allows users to access multiple website services using one account and one login process. OpenID Connect (OIDC) is a protocol based on OAuth 2.0 and provides user identity information through JWT (JSON Web Token) tokens. Virtual Private Server (VPS) is a service that can be configured according to user needs. Internet users have many accounts needed to access various online services such as email, social media, online shopping, and many more. Each online service usually has a different username and password. Users often find it difficult to remember each password needed. In addition, using the same password on each online service is also not recommended because it can increase security risks. Based on this problem, an SSO system is needed that is useful for helping users in logging in. The SSO system will be tested with two Dummy Students and Dummy Elearning clients using Black Box Testing with the Equivalence Partitioning technique. From the test results, it was obtained that the SSO system showed that 83% of the test scenarios were successfully carried out.

Keywords: Black Box Testing, OpenID Connect, Single Sign-On, Virtual Private Server.

1. Pendahuluan

1.1 Latar Belakang

Internet merupakan salah satu teknologi yang sangat berguna bagi kehidupan manusia saat ini. Dengan internet, dapat mengakses berbagai layanan online yang menawarkan kemudahan dan keuntungan bagi seluruh pengguna. Namun, untuk mengakses layanan online tersebut, pengguna membutuhkan akun yang terdiri dari username dan password yang berbeda-beda. Hal ini tentu saja menimbulkan masalah bagi pengguna internet.

Masalah utama yang dihadapi oleh pengguna internet adalah kesulitan untuk mengingat setiap password yang dibutuhkan untuk mengakses setiap layanan online. Selain itu, menggunakan password yang sama untuk setiap layanan online juga berisiko karena dapat menyebabkan kebocoran data atau pencurian identitas jika ada pihak yang tidak bertanggung jawab yang berhasil masuk ke salah satu akun pengguna. Oleh karena itu, pengguna internet membutuhkan sebuah solusi yang dapat mempermudah kita dalam mengelola akun-akun pengguna.

[Type here]

Salah satu solusi yang dapat digunakan pengguna adalah Single Sign-On (SSO) adalah sebuah teknologi yang mengizinkan pengguna jaringan agar dapat mengakses sebuah sumber daya atau layanan yang ada pada jaringan dan hanya dengan menggunakan satu akun saja[2]. Keuntungan menggunakan SSO yaitu tidak diharuskannya pengguna untuk melakukan login dan mengingat semua akun aplikasi secara terpisah[4]. Menurut penelitian yang dilakukan oleh[3], OpenID Connect (OIDC) adalah sebuah protokol SSO yang dibuat diatas protokol 2.0, protokol OIDC mempunyai tiga entitas yang saling berkaitan satu sama lain yaitu End-User, OpenID Provider, dan Relying Party. Keunggulan dari protokol OIDC dari protokol SSO lainnya, OIDC juga menggunakan JSON Web Token (JWT) sebagai format token, yang lebih ringkas dan mudah digunakan daripada format token lain seperti SAML.

Karena sistem SSO berjalan pada sebuah server maka digunakanlah sebuah Virtual Private Server (VPS) sebagai server yang membantu dalam menjalankan sistem SSO. SSO adalah teknologi masa depan yang dapat membantu pengguna dalam menjalani kehidupan digital pengguna[5].

1.2 Rumusan Masalah

Berdasarkan uraian pada latar belakang, maka dirumuskan masalah penelitian ini yaitu, bagaimana cara menerapkan sistem SSO menggunakan protokol OIDC pada sebuah VPS dan bagaimana cara menguji kinerja dari sistem SSO menggunakan protokol OIDC pada sebuah VPS.

2. Tinjauan Literatur

2.1 Single Sing-On (SSO)

Single Sign-On (SSO) adalah sebuah teknologi yang mengizinkan pengguna jaringan agar dapat mengakses sebuah sumberdaya dalam jaringan dan hanya dengan menggunakan satu akun[2]. Dengan demikian aplikasi yang berbasis SSO dapat mengingat satu nama pengguna dan satu kata sandi pengguna, sehingga pengguna tidak perlu melakukan proses pendaftaran ulang pada aplikasi yang berbeda.

SSO memiliki cara untuk menghindari login ganda adalah dengan mengidentifikasi subjek secara ketat dan memperkenalkan informasi autentikasi untuk digunakan dalam sistem atau kelompok sistem yang terpercaya. SSO dapat meningkatkan kegunaan jaringan secara keseluruhan dan pada saat yang sama dapat memusatkan pengelolaan dari parameter sistem yang relevan. Pengguna layanan lebih menyukai sistem SSO, namun pengelola layanan jaringan memiliki banyak tugas tambahan yang harus dilakukan, seperti perlunya perhatian ekstra untuk menjamin bukti-bukti autentikasi agar tidak tersebar dan tidak disadap pihak lain ketika melintasi jaringan[1].

2.2 Virtual Private Server (VPS)

Virtual Private Server adalah jenis hosting yang resourcenya digunakan secara pribadi. Jadi pada dasarnya shared hosting menggunakan server yang yang resourcenya dibagi oleh user lain, dan kelemahan dalam shared hosting ini adalah ketika ada user yang menggunakan resource berlebihan akan memiliki dampak ke user lainnya. Untuk Virtual Private Server, resource yang digunakan hanya oleh satu user[7].

Cara kerja VPS sama dengan seperti server fisik tetapi VPS membagi sumber daya seperti CPU, RAM, dan ruang penyimpanan menjadi beberapa lingkungan virtual yang terpisah satu sama lain. Walaupun begitu VPS dapat menjalankan sistem operasi dan aplikasi secara mandiri[6].

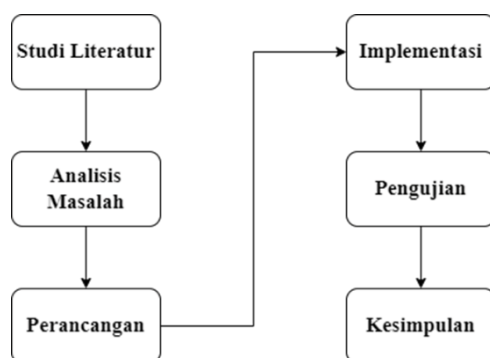
[Type here]

Keuntungan yang didapat 10 bagi pengguna jika menggunakan sebuah VPS, pengguna tidak perlu mengeluarkan biaya untuk membeli dan memelihara server fisik sendiri dan juga mendapatkan keamanan dan privasi yang lebih baik karena sumber daya server yang digunakan hanya dimiliki oleh satu pengguna atau kelompok pengguna tertentu.

3. Metode Penelitian

3.1 Tahapan Penelitian

Tahapan atau skema penelitian ini terbagi menjadi beberapa tahapan dan dapat dilihat pada Gambar 3.1.



Gambar 1 Tahapan Penelitian

3.2 Perancangan

3.2.1 Kebutuhan Software dan Hardware

Hal yang dibutuhkan dalam penelitian ini adalah kebutuhan perangkat keras (hardware) dan perangkat lunak (software), hal ini bertujuan untuk mendukung proses perancangan dan implementasi penelitian ini. Kebutuhan tersebut dapat dilihat pada tabel 3.1, 3.2, dan 3.3.

1. Server

Server yang digunakan pada penelitian ini merupakan VPS yang diinstall menggunakan Virtualbox dengan spesifikasi pada Tabel 1.

Tabel 1 Spesifikasi Server

Nama Komponen	Spesifikasi
<i>Processor</i>	1 VCPU
<i>Memory</i>	2048 MB
<i>Storage</i>	20 GB

2. Client

Client pada penelitian ini berguna untuk menjadi alat pendukung dalam pengujian sistem SSO. Adapun spesifikasinya pada Tabel 2.

Tabel 2 Spesifikasi Client

Nama Komponen	Spesifikasi
<i>Processor</i>	AMD Ryzen 3 3250U
<i>Memory</i>	DDR4 4096MB
<i>Storage</i>	SSD 250GB

3. Software

Software pada penelitian ini berguna untuk menjadi alat pendukung dalam pengujian sistem SSO. Adapun software apa saja yang akan diinstal dapat dilihat pada Tabel 3

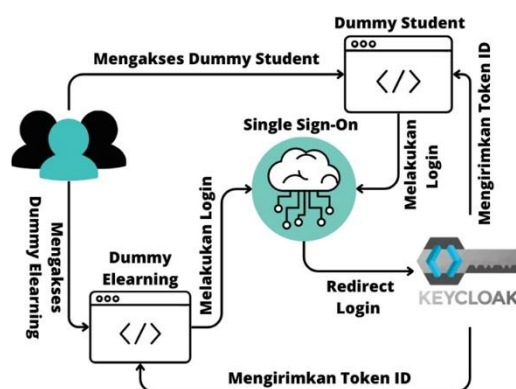
Tabel 3 Daftar Kebutuhan Perangkat Lunak (Software)

Nama Komponen	Nama Software
<i>Identity Provider</i>	Keycloak
<i>Operating System</i>	Windows 11, Debian 10 (Buster)
<i>Package</i>	Docker, Apache2, NodeJS
<i>Browser</i>	Brave Browser
<i>VPS</i>	VirtualBox

Untuk memastikan bahwa software yang dibutuhkan dapat berjalan dengan lancar dan optimal, maka versi dari perangkat lunak (software) yang digunakan harus sesuai dengan spesifikasi perangkat keras (hardware).

3.2.2 Desain Sistem

Dalam studi kasus penelitian ini, pengguna dapat mengakses dua aplikasi, yaitu dummy student dan dummy elearning dengan menggunakan SSO. Dummy student adalah sebuah layanan website yang berfungsi sebagai informasi untuk mahasiswa UIGM seperti melihat KHS ataupun KRS, sedangkan dummy elearning adalah sebuah layanan website untuk pembelajaran online mahasiswa UIGM.



Gambar 2 Desain sistem Single Sign-On (SSO)

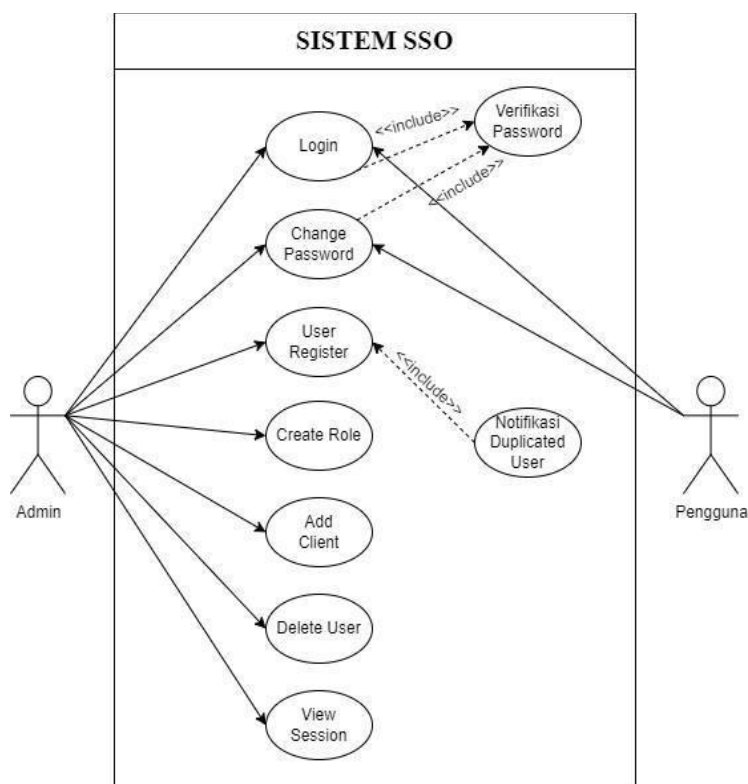
Pada Gambar 2 dijelaskan bahwa keycloak dapat menghasilkan sebuah token ID yang berisi informasi identitas pengguna, yang dapat digunakan oleh layanan website lain untuk mengenali pengguna. Untuk mengakses dummy student dan dummy elearning dengan SSO, pengguna harus mengunjungi halaman web dummy student atau dummy elearning dan memilih opsi login

[Type here]

dengan SSO. Setelah itu, pengguna akan diarahkan ke halaman web *login* keycloak, di mana pengguna harus memasukkan *username* dan *password* yang valid.

3.2.3 Use Case Diagram

Pada Gambar 3.3 menunjukkan rancangan sistem SSO yang akan dikembangkan dalam penelitian ini. Sistem SSO dirancang menggunakan Use Case Diagram untuk melihat apa saja yang dapat dilakukan oleh admin dan pengguna.



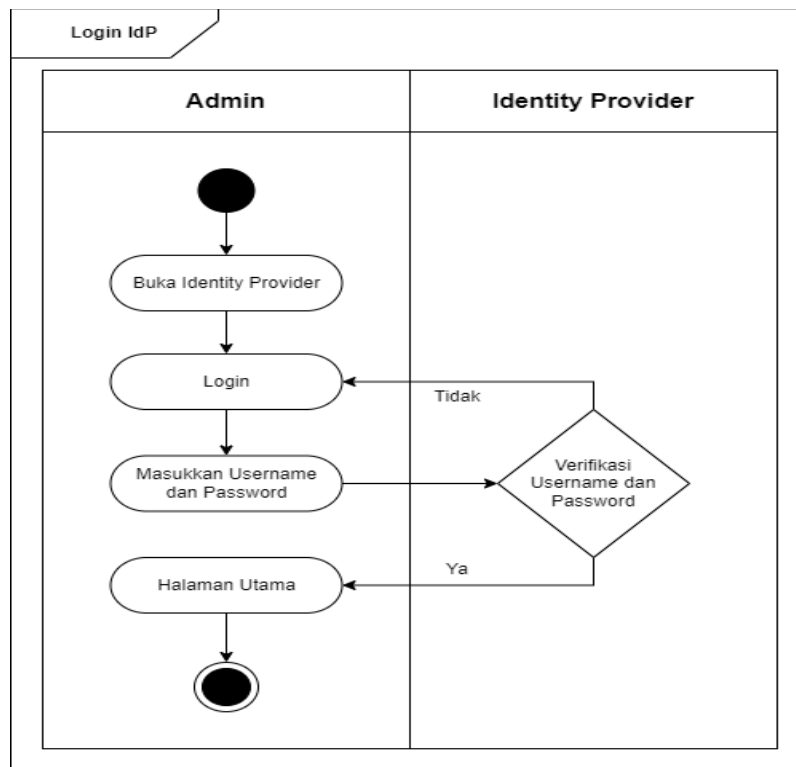
Gambar 3 Perancangan Sistem SSO

Pada Gambar 3 pengguna hanya bisa melakukan *login* dan mengubah *password* pada *Identity Provider*. Sedangkan *admin* dapat menambahkan pengguna baru ke *Identity Provider*, *admin* juga dapat menambahkan *client* baru untuk diintegrasikan ke dalam sistem SSO, *admin* dapat menambahkan *role* baru untuk mengelompokkan pengguna dan membatasi akses pengguna, *admin* juga dapat menghapus *user* untuk memutus akses bagi pengguna, dan *admin* dapat melihat *session* yang berjalan pada *Identity Provider*.

3.2.4 Activity Diagram

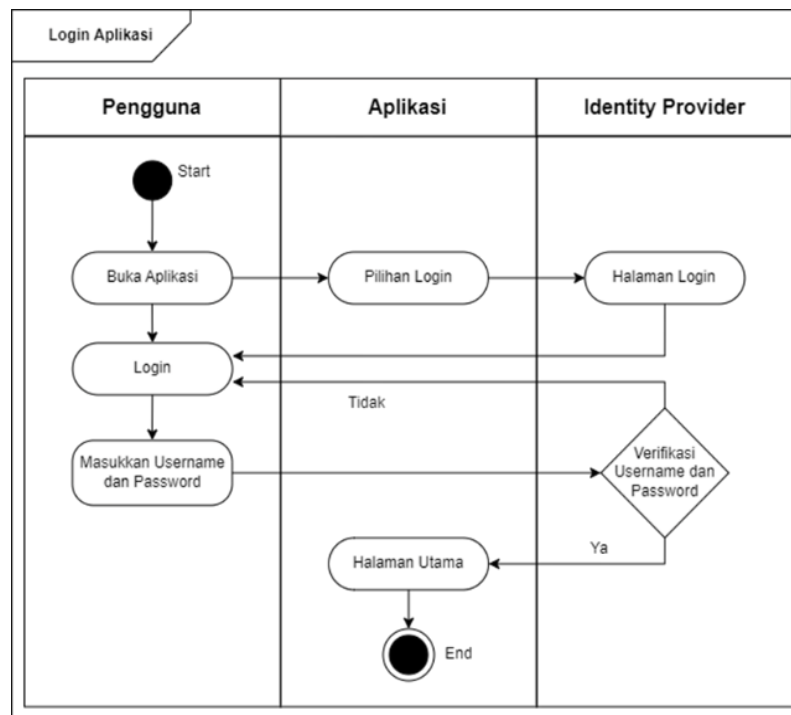
Salah satu tahapan dalam pembuatan sistem SSO adalah membuat activity diagram yang menggambarkan alur aktivitas yang dilakukan oleh admin dan pengguna. Activity diagram ini berfungsi untuk mengidentifikasi fitur-fitur yang tersedia bagi admin dan pengguna dalam sistem SSO, serta hubungan antara aktivitas-aktivitas tersebut. Activity diagram ini membantu dalam merancang dan menguji sistem SSO secara efektif.

1. *Activity Diagram Login Identity Provider*



Gambar 4 *Activity Diagram Login Identity Provider*

2. *Activity Diagram Login Aplikasi*

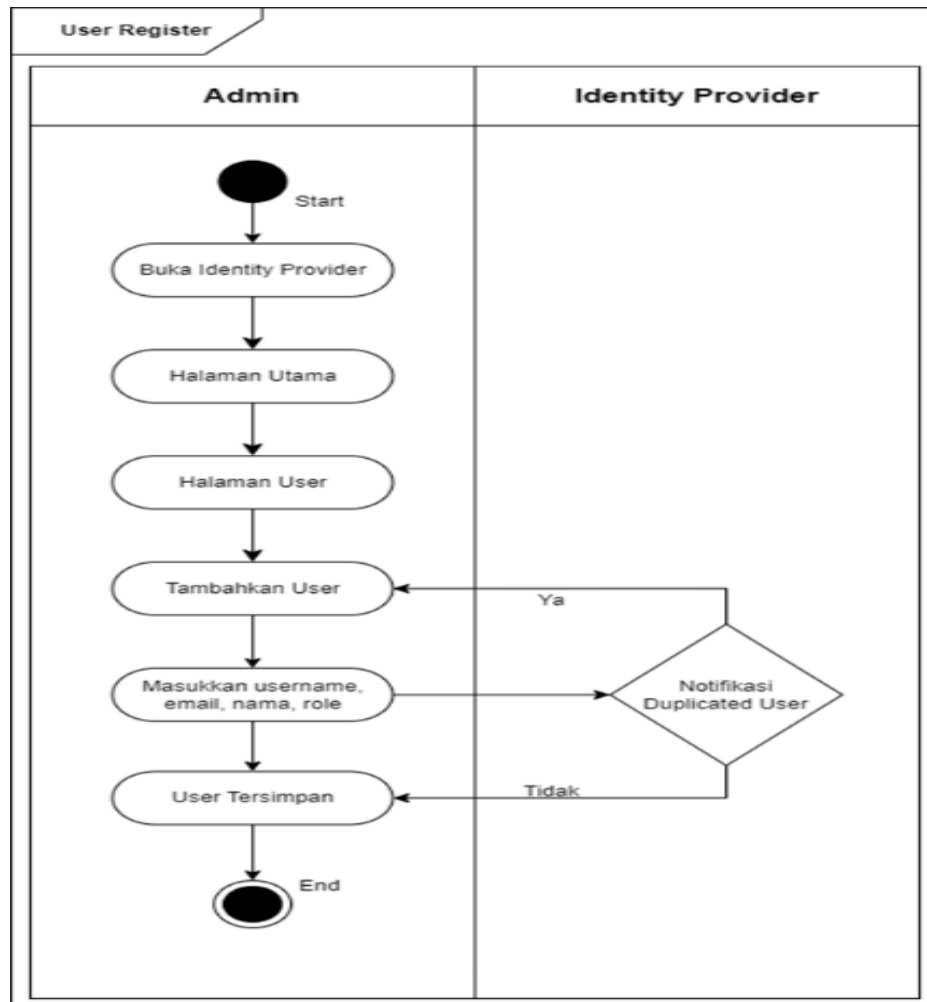


Gambar 5 *Activity Diagram Login Aplikasi*

[Type here]

Pada Gambar 5 dijelaskan bagaimana alur *login* dari sistem *SSO*, ketika pengguna akan mengakses aplikasi dan ingin melakukan *login*, pengguna akan memilih *login* menggunakan *Identity Provider* yang digunakan sistem *SSO*, setelah itu pengguna akan diarahkan ke halaman *login* yang ada pada *Identity Provider*.

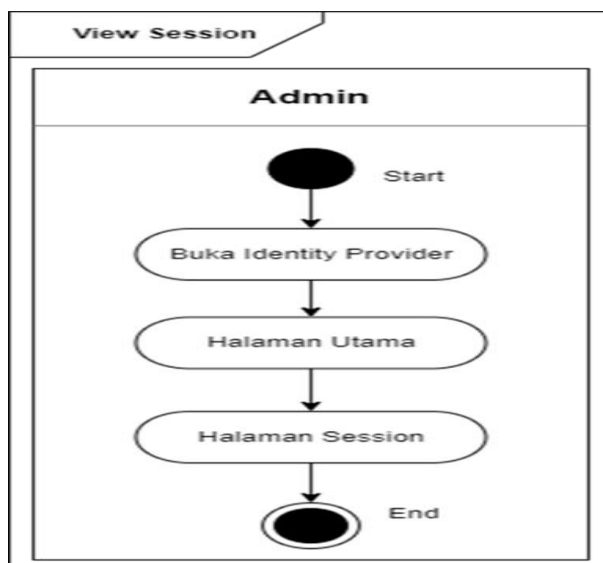
3. Activity Diagram User Register



Gambar 6 Activity Diagram User Register

Dilihat pada Gambar 6 *Admin* dapat menambahkan seorang pengguna baru ke *Identity Provider*. Hal yang diperlukan *admin* ketika ingin menambahkan pengguna baru ke *Identity Provider* adalah *username*, *email*, *nama*, dan *role*. *Admin* akan memilih peran atau akses yang diberikan kepada pengguna baru, seperti hak akses *administrator* atau pengguna biasa.

4. Activity Diagram View Session



Gambar 7 Activity Diagram View Session

Dapat dilihat seperti pada gambar 3.7 menggambarkan bahwa admin dapat melihat sesi pengguna yang sedang berlangsung dalam sistem SSO atau yang sedang mengakses SSO pada Identity Provider. Hal ini dimaksudkan agar admin dapat mengawasi dan mengamankan sistem SSO. Selain itu, admin juga dapat melihat informasi IP Address dan *client* yang digunakan oleh pengguna. Jadi apabila sistem SSO mengalami kebocoran data admin dapat memantau pengguna pada Admin Console dan melakukan reset *password* pengguna.

3.3 Pengujian

Setelah membangun sistem SSO, maka akan dilakukan sebuah pengujian dengan metode *Black Box Testing*. Pengujian ini dilakukan untuk membuktikan bahwa sistem ini memudahkan proses *login*, pengujian ini juga untuk memverifikasi dan memvalidasi kualitas sistem SSO. *Black Box Testing* untuk menguji tampilan antarmuka (*interface*) dari SSO agar mudah digunakan oleh pengguna, sistem ini akan diuji dengan teknik *Equivalence Partitioning*.

4. Hasil dan Pembahasan

4.1 Implementasi

Tahapan selanjutnya setelah tahapan rancangan (seperti yang telah dibahas pada bab III) adalah implementasi. Implementasi bertujuan untuk menerapkan sistem yang sudah dirancang sebelumnya. Pada tahapan implementasi ini akan menjelaskan konfigurasi *server* sampai dengan perbandingan *login*.

Tabel 4 Hasil Pengujian Nomor tabel Tanpa nomor Bab 4.

Kode Test	Hasil Test
L01	Sesuai
L02	Tidak Sesuai
L03	Sesuai
L04	Tidak Sesuai

R01	Sesuai
R02	Sesuai
R03	Sesuai
R04	Sesuai
R05	Sesuai
R06	Sesuai
DS01	Sesuai
DE01	Sesuai

Jumlah Pengujian	12
Sesuai	10
Tidak Sesuai	2

Berdasarkan hasil pengujian pada Tabel 4, dapat dilihat bahwa ada 12 kasus uji yang dilakukan, dan 10 di antaranya berhasil. Dengan menggunakan rumus tingkat keberhasilan sistem $SSO = \text{jumlah kasus uji yang berhasil} / \text{jumlah kasus uji total}$, maka dapat dihitung tingkat keberhasilan sistem SSO adalah $10/12=0.83$. Dengan begitu dapat diperoleh tingkat keberhasilan 83% pada pengujian *black box* dengan teknik *equivalence partition*.

4.1.1 Konfigurasi Server

Setelah menginstall VirtualBox dan menyesuaikan spesifikasi *Virtual Private Server (VPS)* pada tabel 3.1, langkah pertama adalah melakukan instalasi sebuah *VPS* menggunakan aplikasi Virtualbox. Tahapan selanjutnya melakukan instalasi terhadap *package* yang akan digunakan, *Docker* berfungsi sebagai *container* yang menjalankan *keycloak* dari sistem *Single Sign-On*, sedangkan *Apache2* berfungsi untuk menyediakan layanan web terhadap *client*. Kedua *package* tersebut diinstal kedalam *VPS* yang sudah diinstalasi sebelumnya.

4.1.2 Perbandingan Login

Login SSO dan *login biasa* adalah dua metode *login* yang berbeda dalam hal proses otentikasi dan pengelolaan akun pengguna. *Login biasa* menuntut pengguna untuk menghafal dan memasukkan *username* dan *password* untuk tiap layanan website yang digunakan, yang dapat menyulitkan pengguna jika memiliki banyak akun. Berikut ini adalah tabel perbandingan antara *login SSO* dan *login biasa* berdasarkan beberapa aspek:

Tabel 5 Perbandingan Login

Aspek	<i>Login dengan SSO</i>	<i>Login Biasa</i>
Kecepatan	Memungkinkan <i>login</i> cepat ke berbagai layanan dengan satu kali autentikasi.	Memerlukan <i>login</i> terpisah untuk setiap layanan yang berbeda.

Kemudahan Pengguna	Pengguna hanya perlu mengingat satu set akun (nama pengguna dan kata sandi) untuk mengakses banyak layanan.	Pengguna harus mengingat akun unik untuk setiap layanan dan <i>login</i> secara terpisah.
Manajemen Akses	<i>Admin</i> dapat dengan mudah mengatur dan mengelola hak akses pengguna ke berbagai layanan melalui satu titik kontrol.	<i>Admin</i> harus secara terpisah mengatur hak akses pengguna untuk setiap layanan yang digunakan.
Pengalaman Pengguna	Memungkinkan pengalaman pengguna yang mulus, karena pengguna tidak perlu memasukkan akun berulang kali saat mengakses layanan yang berbeda.	Pengguna mungkin merasa kurang nyaman karena harus memasukkan akun setiap kali mereka ingin mengakses layanan baru.
Ketergantungan	Bergantung pada infrastruktur dan keberadaan sistem <i>SSO</i> yang andal	Tidak bergantung pada infrastruktur <i>SSO</i> tertentu dan dapat diimplementasikan secara mandiri oleh setiap layanan.

Pada tabel perbandingan *login* 4.6 diatas adalah hasil dari penelitian yang dilakukan dan mendapatkan hasil tersebut, tidak ada satu cara *login* yang sempurna untuk semua situasi. Baik *login* dengan *SSO* maupun *login* biasa memiliki kelebihan dan kekurangan masing-masing dari beberapa aspek, seperti kemudahan penggunaan, kecepatan, dan ketergantungan. Oleh karena itu, kita perlu mempertimbangkan faktor-faktor tersebut sebelum memilih cara *login* yang paling sesuai untuk aplikasi atau layanan yang kita gunakan. Tetapi apabila ada aplikasi atau layanan website lebih dari satu yang ada dalam satu jaringan, contohnya seperti google yang mempunyai gmail, drive, dan youtube. Tentunya dengan menggunakan *login SSO* itu dapat mempermudah dalam melakukan sebuah *login*, karena pengguna tidak harus melakukan *login* satu per satu kedalam aplikasi atau layanan website yang ingin diakses, *login SSO* juga dapat menghemat waktu dari pengguna ketika melakukan *login*.

5. Kesimpulan

Setelah melalui beberapa tahapan dalam penelitian implementasi Single Sign-On (*SSO*) menggunakan protokol OpenID Connect (*OIDC*) pada Virtual Private Server (*VPS*), berdasarkan hasil dari implementasi *Single Sing-On (SSO)*, dapat ditarik kesimpulan bahwa :

1. Sistem *SSO* yang menggunakan protokol *OIDC* dapat berjalan pada sebuah *VPS*.
2. Hasil pengujian dari sistem *SSO* yang dilakukan menggunakan *Black Box Testing* dengan teknik *Equivalence Partitions* memperoleh tingkat keberhasilan 83% pada

pengujian terhadap kinerja dari sistem *SSO* yang dibangun dengan menggunakan beberapa skenario pengujian.

3. Dummy client dapat berfungsi sebagai alat pengujian sistem *SSO*. Dummy client dalam penelitian ini digunakan untuk menguji kinerja dari sistem *SSO* untuk memastikan bahwa sistem *SSO* beroperasi secara efektif dan dapat diandalkan.
4. Sistem *SSO* yang dibangun masih memiliki kekurangan dalam segi keamanan, contohnya belum menggunakan metode keamanan tambahan *MFA (Multi-Factor Authentication)* dalam melakukan login.

Referensi

1. Q. Aini, U. Rahardja, R. S. Naufal, D. Stmik, R. Jurusan, S. Informasi, S. Stmik, and S. Komputer, "Penerapan Single Sign On Dengan Google Pada Website Berbasis Yii Framework Application Single Sign On With Google The Website Based On Yii Framework," vol. 8, no. 1, 2018.
2. Ardiyah, R. Okra, H. A. Musril, I. Bukittinggi, and K. Kunci, "Perancangan Sistem Absensi Siswa Dengan Menerapkan *SSO (Single Sign On)* Di SMKN 1 Lembah Melintang," *Humantech Jurnal Ilmiah Multi Disiplin Indonesia*, no. N.D. [Online]. Available: <http://siapsi.fetrinet.com/>.
3. L. Holtmann, J. Schwenk, and Dr.-I. V. Mladenov, "Single Sign-On Security: Security Analysis Of Real-Life Openid Connect Implementations," N.D.
4. R. Saputra, "Pengembangan Sistem Single Sign On Pt Telkom Akses Dengan Menerapkan Protokol Openid Connect," N.D. [Online]. Available: <https://journals.upi-yai.ac.id/index.php/ikraith-informatika/issue/archive>.
5. Z. R. Mair, *Teori dan Peraktek Sistem, Operasi*. Yogyakarta: Deepublish, 2018.
6. Z. R. Mair and D. T. Ariska, "Optimalisasi Kinerja Jaringan Komputer Berbasis Router Pada Dinas Kependudukan Dan Catatan Sipil Kabupaten Musi Banyuasin," *Jurnal TIPS: Jurnal Teknologi Informasi dan Komputer Politeknik Sekayu*, vol. VIII, no. 1, pp. 46-54, Jan.-Jun. 2018.
7. M. A. Zein, U. Y. K. S. Hedyanto, and A. Almaarif, "Hardening Sistem Operasi Virtual Private Server Fakultas Rekayasa Industri Berdasarkan Nist SP 800-123," *JIPI (Jurnal Ilmiah Penelitian Dan Pembelajaran Informatika)*, vol. 8, no. 1, pp. 230–241, 2023. [Online]. Available: <https://doi.org/10.29100/jipi.v8i1.3438>.