JIPI (Jurnal Ilmiah Penelitian dan Pembelajaran Informatika)
Journal homepage: https://jurnal.stkippgritulungagung.ac.id/index.php/jipi
ISSN: 2540-8984
Vol. 9, No. 4, Desember 2024, Pp. 1768-1778

ROLE-BASED ACCESS CONTROL (RBAC) UNTUK SISTEM OTORISASI TERPUSAT BERBASIS FLASK STUDI KASUS PT. XYZ

Yoga Agung Prasetia*[1)], Danny Manongga[2)]
1. Universitas Kristen Satya Wacana, Indonesia
2. Universitas Kristen Satya Wacana, Indonesia

Article Info
Kata Kunci: Flask; Otorisasi; Organisasi; RBAC; Sistem

Keywords: Authorization; Flask; Organization; RBAC; System

Article history:
Received 29 September 2024
Revised 13 Oktober 2024
Accepted 4 November 2024
Available online 4 December 2024

DOI :
https://doi.org/10.29100/jipi.v4i1.5403

* Corresponding author.
Yoga Agung Prasetia
E-mail address:
672020193@student.uksw.edu

ABSTRAK
Perkembangan teknologi saat ini berpengaruh signifikan pada individu dan organisasi. PT. XYZ, sebagai salah satu perusahaan ritel terkemuka di Indonesia yang ikut memanfaatan teknologi dalam kegiatan bisnisnya, menghadapi tantangan terkait manajemen hak akses yang rumit pada sistem informasinya akibat dari struktur organisasi yang kompleks. Manajemen hak akses menjadi sangat penting dikarenakan perlu adanya pembatasan dalam mengakses informasi pada sistem untuk menjaga keamanan data dan integritas informasi. Oleh karena itu pada penelitian ini dibangunlah sistem otorisasi terpusat dengan tujuan untuk meningkatkan efisiensi dan efektivitas manajemen hak akses pada sistem informasi yang ada di PT. XYZ. Sistem otorisasi terpusat ini dibangun dengan mengadopsi model *Role Based Access Control* (RBAC) dan dalam proses pengembangannya menggunakan *Python Flask Framework.* Dalam penelitian ini, proses dan tahapan penelitian dilakukan secara berurutan dari studi literatur, identifikasi masalah, perancangan sistem, implementasi, dan pengujian. Hasil dari pengembangan sistem ini menunjukkan bahwa menggunakan model RBAC didalam proses pemberian hak akses sangat cocok dilakukan pada suatu organisasi.

ABSTRACT
The current advancement in technology significantly influences individuals and organizations. PT. XYZ, as one of the leading retail companies in Indonesia that actively utilizes technology in its business activities, faces challenges related to complex management access control in its information system due to the intricate organizational structure. Managing access control becomes crucial because there is a need for limitations in accessing information in the system to maintain data security and information integrity. Therefore, in this research, a centralized authorization system is constructed to enhance efficiency and effectiveness in access rights management at PT. XYZ. This centralized authorization system is developed by adopting the Role-Based Access Control (RBAC) model and utilizing the Python Flask Framework in its development process. In this study, the research process and stages were conducted sequentially from literature review, problem identification, system design, implementation, and testing. The results of the system development indicate that utilizing the RBAC model in the access rights granting process is highly suitable for an organization.

.

I. PENDAHULUAN

PERKEMBANGAN teknologi saat ini memberikan pengaruh besar baik dalam lingkup individu maupun organisasi. Sebagai salah satu perusahaan ritel terkemuka di Indonesia, PT. XYZ telah secara proaktif memanfaatkan kemajuan teknologi untuk meningkatkan efisiensi operasionalnya. Saat ini, PT. XYZ telah mengembangkan berbagai sistem informasi yang dirancang khusus untuk mendukung kelancaran proses bisnisnya[1]. Namun, salah satu tantangan yang dihadapi oleh PT. XYZ adalah masalah terkait manajemen hak akses yang semakin rumit dikarenakan dengan bertambahnya pegawai, departemen, dan struktur organisasi. Dalam sistem informasi yang ada pada PT. XYZ, sangat penting untuk memastikan bahwa pengguna memiliki batasan hak akses yang sesuai dengan peran mereka dalam perusahaan, sehingga mereka hanya dapat mengakses modul atau informasi yang relevan demi menjaga keamanan data dan integritas informasi[2]. Pendekatan yang telah

diimplementasikan adalah dengan memberikan hak akses secara langsung kepada pengguna sistem. Pendekatan ini menimbulkan masalah yaitu meningkatnya kompleksitas dalam proses pemberian akses, terutama mengingat jumlah pengguna dan modul dalam sistem akan terus bertambah. Sebaliknya, pendekatan yang lebih efisien adalah mengelompokkan hak akses berdasarkan peran atau fungsi di dalam organisasi, yang kemudian diberikan kepada pengguna. Pendekatan ini jauh lebih sederhana dan mempermudah manajemen hak akses.

RBAC merupakan salah satu dari berbagai pendekatan kontrol akses yang digunakan untuk mengatur hak akses pengguna di sebuah sistem yang didasari dengan peran atau jabatannya dalam organisasi[3]. Terdapat beberapa pendekatan kontrol akses yang sudah ada seperti *Mandatory Access Control* (MAC) dan *Discretionary Access Control* (DAC)[4]. Dalam pendekatan DAC dan MAC pemberian akses diberikan langsung pada subjek pengguna dimana hal ini akan mendatangkan masalah karena jumlah pengguna dan sumber daya di dalam sistem akan selalu bertambah sehingga sulit dalam pemeliharaan dan pengaturannya. RBAC menawarkan sejumlah keunggulan diantaranya yaitu perubahan antara peran dan izin berlangsung lebih lambat daripada antara peran dan pengguna yang mana akan mengurangi kompleksitas dan beban manajemen. RBAC dapat mendukung kebijakan keamanan sistem secara fleksibel, memberikan adaptabilitas yang tinggi terhadap perubahan sistem. Dalam hal operasional, alokasi otoritas menjadi lebih intuitif, mudah dipahami, dan mudah digunakan, dengan otoritas hierarkis yang sesuai untuk struktur level pengguna yang berjenjang. RBAC juga menonjolkan reusabilitas yang kuat[5]. Dengan RBAC maka pengguna akan dikelompokkan berdasarkan jabatannya dan jabatan akan diberikan akses ke menu dalam sistem, sehingga ini akan jauh lebih sederhana dan mudah dalam pemeliharaan dan pengaturannya. Berdasarkan hal tersebut, RBAC merupakan pendekatan yang sesuai untuk mengatasi tantangan yang dihadapi oleh PT. XYZ terkait manajemen hak akses.

Penelitian mengenai RBAC sebelumnya telah dilakukan dalam konteks lingkungan organisasi, sebagaimana terdokumentasi dalam penelitian yang berjudul ***"RBAC Architecture Design Issues in Institutions Collaborative Environment"***. Dalam kajian tersebut, peneliti mengulas berbagai permasalahan yang dihadapi oleh institusi kolaboratif yang memerlukan pengendalian akses bagi pengguna dengan tingkat akses yang berbeda terhadap sistem. Oleh karena itu, diperlukan suatu arsitektur yang sesuai untuk sistem semacam ini, guna menyediakan platform yang efisien dan aman. Hasil penelitian menunjukkan bahwa penerapan RBAC di dalam sebuah organisasi, terutama dalam konteks institusi kolaboratif yang menjadi fokus penelitian, dapat memberikan sistem yang lebih aman dan efisien dalam mengelola informasi serta sumber daya[6]. Pernyataan tersebut sejalan dengan arah penelitian yang sedang dilakukan. Dengan mengimplementasikan RBAC, diharapkan sistem informasi di PT. XYZ dapat memiliki otorisasi yang efektif dan efisien. Meskipun demikian, terdapat perbedaan signifikan dengan penelitian sebelumnya yang dilakukan oleh Dr. Asif. Fokus penelitian Dr. Asif terletak pada perancangan arsitektur hierarki RBAC untuk institusi, sedangkan penelitian ini lebih menitikberatkan pada pengembangan sistem otorisasi terpusat dengan pendekatan model RBAC.

Penelitian terdahulu yang memanfaatkan RBAC juga pernah dilakukan, yakni penelitian dengan judul **"Implementasi Role-Based Access Control (RBAC) Pada Pemanfaatan Data Kependudukan Ditingkat Kabupaten"** yang dilakukan oleh Rubiyanto. Penelitian ini berfokus pada proses pengamanan data kependudukan di database Sistem Informasi Administrasi Kependudukan (SIAK) dengan cara mengatur hak akses sistem informasi sebagai pengakses dalam mengakses data kependudukan di database SIAK. Hasil penelitian ini diharapkan dapat menjadi masukan penting dalam pembuatan Perjanjian Kerja Sama (PKS) untuk pemberian hak akses pada layanan data kependudukan di berbagai sistem pelayanan publik, khususnya di tingkat Kabupaten Wonosobo[7]. Perbedaan penelitian sekarang dengan studi yang dilakukan oleh Rubiyanto terletak pada pemberian hak akses. Rubiyanto memberikan hak akses kepada sistem informasi sebagai entitas pengakses untuk mengakses layanan data kependudukan melalui *web service*, sesuai dengan tugas dan peran yang terkait dalam sistem informasi tersebut. Sementara penelitian yang akan dilakukan fokus pada pemberian hak akses kepada individu atau pengguna berdasarkan departemen mereka dalam suatu sistem informasi.

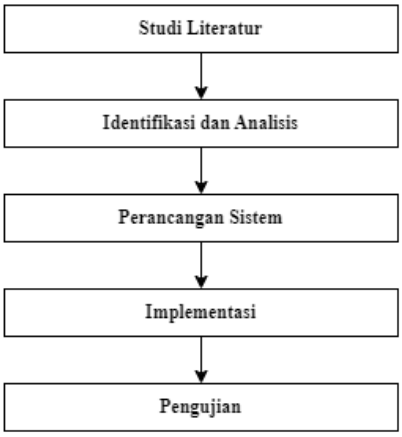
Penelitian lain yang membahas implementasi RBAC pada sistem aplikasi telah dilakukan dengan judul ***"Role-based Access Control in Educational Administration System"***. Fokus penelitian tersebut adalah masalah keamanan dalam sistem administrasi di konteks pendidikan. Dalam upaya menangani tantangan keamanan di sistem administrasi pendidikan, penelitian tersebut mengutamakan metode kontrol berbasis RBAC. Hasil dari penelitian tersebut mencakup pengembangan program RBAC yang memenuhi persyaratan keamanan sistem, meningkatkan kemampuan kontrol akses, dan efektif melindungi operasional sistem administrasi pendidikan[8]. Perbedaan signifikan dengan penelitian saat ini terletak pada penelitian saat ini yang fokus pada permasalahan keamanan dalam sistem informasi di lingkungan perusahaan ritel.

Dalam artikel ini, akan dibahas penerapan model RBAC dalam pengembangan sistem otorisasi terpusat menggunakan Python Flask *Framework*. Flask adalah *micro web framework* yang menggunakan bahasa pemrograman Python. Penggunaan *framework* ini bertujuan untuk mempercepat pengembangan aplikasi, karena di

JIPI (Jurnal Ilmiah Penelitian dan Pembelajaran Informatika)
Journal homepage: https://jurnal.stkippgritulungagung.ac.id/index.php/jipi
ISSN: 2540-8984
Vol. 9, No. 4, Desember 2024, Pp. 1768-1778

dalam Flask telah disertakan *library* yang siap digunakan. Hal ini memungkinkan pembangunan aplikasi web tanpa perlu membuat semuanya dari awal[9]. Penelitian ini bertujuan untuk memberikan model RBAC sebagai solusi dari masalah yang dihadapi oleh perusahaan ataupun organisasi khususnya PT XYZ yang memiliki struktur jabatan yang jelas dalam mengatur pemberian hak akses. Dengan model RBAC hak akses tidak diberikan langsung ke individu penggunanya, melainkan disederahanakan dengan mengelompokan pengguna berdasarkan jabatan mereka. Model RBAC ini akan diterapkan pada sistem informasi di PT XYZ sehingga akan memudahkan pengaturan hak akses karena pemberikan hak akses didasari dengan jabatan pengguna yang menjadikan pengaturan lebih sederhana. Hasil penelitian ini akan menghasilkan sebuah sistem otorisasi terpusat dengan pendekatan model RBAC yang dapat diterapkan dalam lingkungan pengembangan sistem informasi di PT. XYZ.

II. METODE PENELITIAN



Gambar 1 Metode Penelitian

Penelitian ini mengikuti serangkaian langkah yang berurutan dan terus menerus. Dimulai dengan pemahaman konsep dasar RBAC beserta keunggulan dan kekurangannya dalam manajemen hak akses, kemudian dilanjutkan dengan identifikasi dan analisis terhadap sistem informasi yang sudah ada di perusahaan. Berdasarkan hasil analisis, dilakukan perancangan sistem untuk menerapkan RBAC, diikuti dengan tahap implementasi yang melibatkan pembangunan sistem sesuai dengan spesifikasi yang telah ditetapkan. Akhirnya, sistem yang telah dibangun diuji untuk memastikan bahwa itu berjalan dengan benar dan sesuai dengan kesepakatan di awal. Dengan demikian, penelitian ini dapat menarik kesimpulan yang tepat berdasarkan hasil penelitian dan pengujian sistem yang telah dilakukan.

A. Studi Literatur

Model DAC dan model MAC memberikan hak akses secara langsung kepada pengguna sistem. Model ini memiliki banyak masalah keamanan yang disebabkan oleh modul fungsional pada sistem yang semakin banyak, dan sumber daya objek yang perlu dilindungi tumbuh secara eksponensial, membuat sistem sulit dipelihara dan dikelola. Selain itu, model DAC terlalu fleksibel, dan otoritas operasional objek dapat diberikan sembarangan oleh pemilik subjek, mengancam keamanan sistem. Model MAC memiliki kekurangan sendiri, dan beban kerjanya besar dan sulit dikelola. Karena mekanisme manajemen otoritas yang ketat, efisiensi otorisasi rendah, dan fleksibilitas model yang rendah, sehingga tidak dapat diterapkan pada lingkungan terbuka dinamis. Oleh karena itu, DAC dan MAC sudah tidak dapat memenuhi kebutuhan aplikasi praktis, sehingga model RBAC muncul.

Model RBAC memiliki keunggulan dari model DAC dan MAC, dan cepat diakui oleh komunitas akademis. Dibandingkan dengan model kontrol akses tradisional, model RBAC menambahkan peran sebagai penghubung antara pengguna dan izin[10]. Pertama, setel kumpulan izin, kumpulan peran, dan kumpulan pengguna sesuai dengan kebutuhan sistem, dan atur hubungan alokasi antara pengguna dan peran, peran dan izin. Izin yang dimiliki seorang pengguna adalah jumlah izin yang dimiliki kumpulan perannya. Seluruh proses kontrol akses dibagi menjadi dua bagian penting: pengguna dan peran, peran dan izin, yang memiliki hubungan *many-to-many*, sehingga mewujudkan pemisahan logis antara pengguna dan izin. RBAC adalah teknologi yang terbukti dapat diterapkan pada sistem kontrol otorisasi berskala besar[5].

JIPI (Jurnal Ilmiah Penelitian dan Pembelajaran Informatika)
Journal homepage: https://jurnal.stkippgritulungagung.ac.id/index.php/jipi
ISSN: 2540-8984
Vol. 9, No. 4, Desember 2024, Pp. 1768-1778

Dengan menerapkan RBAC di sebuah organisasi, terutama dalam domain khusus seperti institusi atau perusahaan, akan menyediakan sistem yang lebih aman dan efisien untuk informasi dan sumber daya. Dalam konteks perusahaan, struktur hierarkis jabatan biasanya menentukan tingkat akses pegawai terhadap informasi perusahaan terkhususnya di sistem informasi perusahaan. Memberikan batasan akses secara individual pada setiap pegawai akan menjadi rumit dan tidak efisien, terutama dalam perusahaan yang memiliki ribuan pegawai. Lebih baik untuk mengelompokkan pegawai berdasarkan jabatan mereka, sehingga hak akses diberikan pada tingkat jabatan. Pendekatan ini lebih sederhana dan mudah dipelihara daripada memberikan hak akses pada tingkat individu. Dengan demikian, RBAC memungkinkan pengaturan akses yang lebih terstruktur dan efisien dalam organisasi.

B. Identifikasi dan Analisis

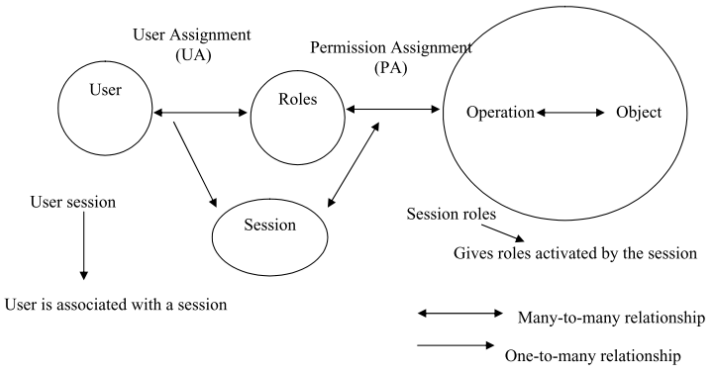
Langkah ini melakukan identifikasi dan analisis terhadap sistem informasi yang sudah ada di perusahaan. Dari idenfikasi dan analisis tersebut, ditemukan bahwa sistem informasi di PT XYZ untuk akses ke modul-modul dalam aplikasi terbatas dan diatur berdasarkan peran yang dimiliki oleh pengguna yang sedang *login*. Dalam struktur organisasi PT XYZ, peran dibentuk untuk menggambarkan fungsi pekerjaan yang berbeda, disesuaikan dengan peran keanggotaan yang didasarkan pada departemen. Berdasarkan pada analisis terhadap sistem informasi yang sudah ada di PT XYZ, diidentifikasi entitas-entitas yang akan terlibat dalam sistem otorisasi terpusat. Entitas-entitas tersebut dapat dilihat pada tabel berikut:

TABEL I
ENTITAS SISTEM OTORISASI TERPUSAT

Entitas	Deskripsi
User	Pegawai yang menggunakan sistem dan memiliki tanda pengenal yang unik yaitu NIK Pegawai.
Jabatan	Level jabatan pegawai yang memiliki nomor khusus dan informasi instansinya (perusahaan, department, dan lain-lain) untuk pengelolaan fungsi dan wewenang.
Modul	Bagian terpisah dari sistem atau program dengan fungsi khusus. Dapat berdiri sendiri atau berinteraksi dengan modul lain untuk mencapai tujuan sistem.

Entitas merupakan objek yang akan menjadi perhatian ketika akan membangun suatu sistem. Entitas dapat berupa manusia, tempat, benda, atau kondisi mengenai data yang dibutuhkan[11]. Dari hasil analisis di atas, entitas-entitas tersebut dapat dipetakan ke dalam model RBAC. Model RBAC dikenal sebagai metode yang sangat efektif dalam mengelola kontrol akses di lingkungan institusi dikarenakan dapat menyederhanakan pemberian hak akses.[6]. Dalam RBAC, setiap peran memiliki sejumlah hak istimewa yang dapat diberikan kepada pengguna[12]. Pemberian hak tersebut bukan secara langsung kepada pengguna melainkan kepada jabatan pengguna.

JIPI (Jurnal Ilmiah Penelitian dan Pembelajaran Informatika)
Journal homepage: https://jurnal.stkippgritulungagung.ac.id/index.php/jipi
ISSN: 2540-8984
Vol. 9, No. 4, Desember 2024, Pp. 1768-1778



Gambar 2 Model RBAC

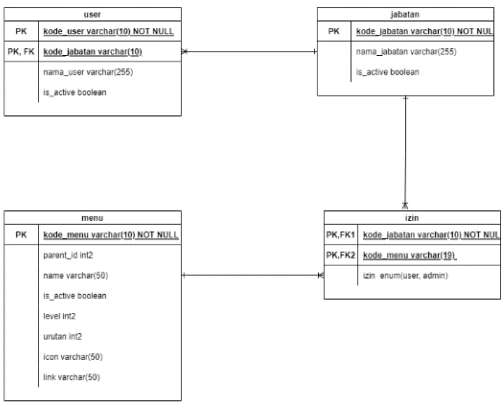
Gambar 2 merupakan komponen-komponen yang membentuk model RBAC. Komponen tersebut meliputi *user, roles, permission*, dan *session*. *User* merupakan elemen yang mewakili kumpulan pengguna atau sistem yang mengakses sistem. Dalam konteks PT XYZ, *user* dapat diidentifikasi sebagai pegawai perusahaan. Setiap *user* dapat terhubung dengan *roles* kemudian *roles* terhubung dengan *permission*. *Roles* adalah fungsi, pekerjaan, atau jabatan dalam sebuah organisasi yang memiliki mengenai wewenang dan tanggung jawab. Dalam penelitian ini *roles* merupakan jabatan pegawai. Sementara itu, *permission* merujuk pada persetujuan dari suatu mode akses tertentu terhadap satu atau lebih objek dalam sistem. Dalam konteks sekarang, *permission* adalah hak akses ke modul/menu dalam sistem informasi[13].

III. HASIL DAN PEMBAHASAN

Dalam penelitian ini, rancangan sistem mencakup arsitektur sistem dan *Entity Relationship Diagram* (ERD).

Gambar 3 Arsitektur Sistem

Gambar 3 menggambarkan arsitektur sistem, dengan proses pada lapisan *backend* yang di dalamnya terdapat *controller* dan *repository* yang dikembangkan menggunakan bahasa pemrograman Python, didukung oleh *framework* Flask, dan menggunakan basis data PostgreSQL sebagai pengelola data aplikasi. Data yang telah diolah di lapisan *backend* disimpan ke dalam database atau diintegrasikan ke dalam tampilan untuk kemudian diproses oleh JavaScript, sehingga dapat ditampilkan kepada pengguna melalui *browser*.



Gambar 4 *Entity Relationship Diagram*

JIPI (Jurnal Ilmiah Penelitian dan Pembelajaran Informatika)
Journal homepage: https://jurnal.stkippgritulungagung.ac.id/index.php/jipi
ISSN: 2540-8984
Vol. 9, No. 4, Desember 2024, Pp. 1768-1778

Entity Relationship Diagram dirancang sebagai alat bantu dalam proses pembuatan database[14]. Diagram ini dibuat dengan merinci alur pemberian izin kepada peran yang dimiliki oleh pengguna. Setiap pengguna memiliki jabatan yang mencerminkan peran mereka. Selanjutnya, jabatan akan memiliki izin untuk menentukan tindakan yang dapat dilakukan terhadap menu tertentu.

Tahap implementasi pada penelitian ini dimulai dengan membuat *Data of Access* (DAO). Jika dalam pengembangan aplikasi, DAO adalah sebuah konsep yang digunakan untuk komunikasi data antara database dengan aplikasi[15].

KODE PROGRAM 1
DATA OF ACCESSS (DAO)

```
from app.lib.postgresKonektor import PostgresDatabase

def get_data_menu_user(id_jabatan: str):
    db = PostgresDatabase()
    query = """
        SELECT get_menu(%(kode_jabatan)s) AS hasil;
    """
    param = {"kode_jabatan": id_jabatan}
    return db.execute(query, param)

def get_data_user(id_user):
    db = PostgresDatabase()
    query = """
        SELECT * FROM ms_user WHERE kode_user =
%(id_user)s;
    """
    param = {"id_user": id_user}
    return db.execute(query, param)
```

Dalam kode program 1, terdapat dua fungsi yang dibentuk: `get_data_menu_user(id_jabatan)` dan `get_data_user(id_user)`. Setiap fungsi memiliki tujuan khusus, dimana `get_data_user(id_user)` bertujuan untuk mengambil data pengguna di database dan memiliki return value yaitu data pengguna berupa nip, nama, dan lain-lain. Sedangkan `get_data_menu_user(id_jabatan)` bertujuan untuk mengambil *menu* yang dapat diakses oleh pengguna aplikasi berdasarkan perannya. Implementasi fungsi-fungsi ini dilakukan melalui pembuatan bahasa procedural PL/pgSQL pada PostgreSQL seperti pada kode program 2 dibawah ini:

KODE PROGRAM 2
PEMBUATAN PL/PGSQL PADA POSGRESQL

```
DROP FUNCTION IF EXISTS get_menu(varchar);

CREATE FUNCTION get_menu(p_kode_jabatan varchar)
RETURNS jsonb
LANGUAGE plpgsql
AS $$
DECLARE
   menu_roles jsonb;
BEGIN

…

   RETURN(
      SELECT jsonb_build_object(
         'menu', (SELECT jsonb_agg(x ORDER BY x.urutan)
FROM temp_menu1 x),
         'menu_role', menu_roles
      )
   );
END$$;
```

PL/pgSQL adalah salah satu dari beberapa bahasa prosedural yang didukung oleh PostgreSQL. Ekstensi ini digunakan dalam sistem manajemen basis data PostgreSQL untuk memungkinkan pengguna membuat fungsi

JIPI (Jurnal Ilmiah Penelitian dan Pembelajaran Informatika)
Journal homepage: https://jurnal.stkippgritulungagung.ac.id/index.php/jipi
ISSN: 2540-8984
Vol. 9, No. 4, Desember 2024, Pp. 1768-1778

prosedural yang dapat dieksekusi di dalam *server* PostgreSQL[16]. Kode program 2 melakukan manipulasi pada tabel-tabel dalam database, termasuk tabel *user*, jabatan, izin, dan menu. Hasil keluaran dari program ini adalah daftar izin yang dimiliki oleh seorang pengguna ke menu berdasarkan jabatannya.

KODE PROGRAM 3
CONTROLLER ATAU ROUTE

```
@app.get("/")
@roles_required('user_1')
def index():
    return render_template("menu1.html")

@app.get('/index2')
@roles_required('admin_2')
def index2():
    return render_template("menu2.html")

@app.route("/login", methods=['GET', 'POST'])
def login():
    ...
    if is_login_succes == True:
        return login_success()

    flash('Pass / Id salah')
    return render_template("login.html")
```

Pada kode program 3, *route* atau *controller* akan mematuhi aturan akses yang telah ditetapkan untuk mengakses *route*. Aturan ini diterapkan melalui penggunaan *decorator* @roles_required. *Decorator* adalah fungsi atau metode yang memungkinkan untuk memodifikasi atau menambahkan perilaku pada fungsi[17]. Penggunaan *decorator* @roles_required diintegrasikan dalam kode dengan tujuan tertentu, yaitu untuk membatasi hak akses terhadap menu, sesuai dengan kriteria kesesuaian peran dari pengguna yang sedang login dengan peran yang diwajibkan dalam decorator @role_required. Pembuatan decorator tersebut dilakukan pada kode program 4 dibawah ini:

KODE PROGRAM 4
DECORATOR ROLE_REQUIRED

```
def roles_required(*required_roles) -> Response:
    def wrapper(fn):
        @login_required
        @wraps(fn)
        def inner(*args, **kwargs):
            nonlocal required_roles
            add_user_role(required_roles)

            if not all(rr in current_user.roles for rr in re-
quired_roles):
                return handle_authz_failure(
                    f"Hanya user dengan role <b>{re-
quired_roles}</b> yang boleh!"
                )

            return fn(*args, **kwargs)
        return inner
    return wrapper
```

Kode program 4 merupakan pembuatan *decorator* @roles_required, yang melakukan pengecekan apakah peran yang disimpan pada *session* pengguna yang sedang login memenuhi kriteria yang ditetapkan dalam *decorator* tersebut. *Session* adalah tempat menyimpan informasi pengguna secara semi-permanen, artinya data tersebut akan tetap tersedia selama periode waktu tertentu[18].

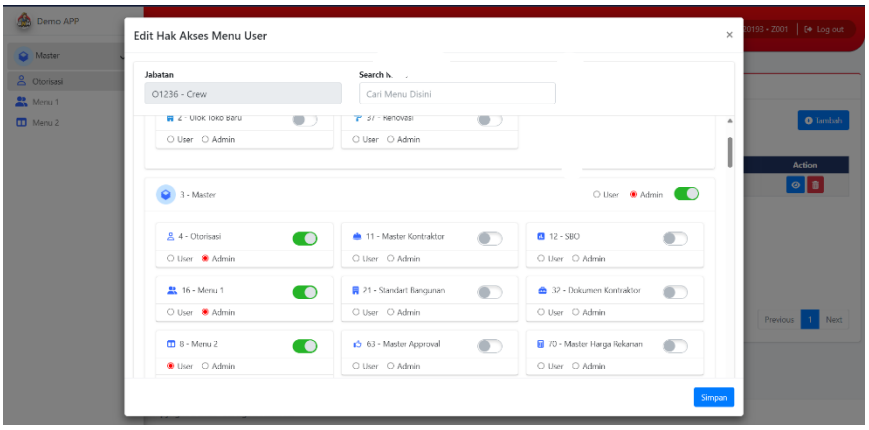
Adapun *session* yang disimpan setelah pengguna berhasil login dapat dilihat dalam gambar berikut:

JIPI (Jurnal Ilmiah Penelitian dan Pembelajaran Informatika)
Journal homepage: https://jurnal.stkippgritulungagung.ac.id/index.php/jipi
ISSN: 2540-8984
Vol. 9, No. 4, Desember 2024, Pp. 1768-1778


```
{
  "kode_user": "672020193",
  "kode_jabatan": "L",
  "nama_user": "Yoga",
  "roles": ["user_1", "user_2"]
}
```

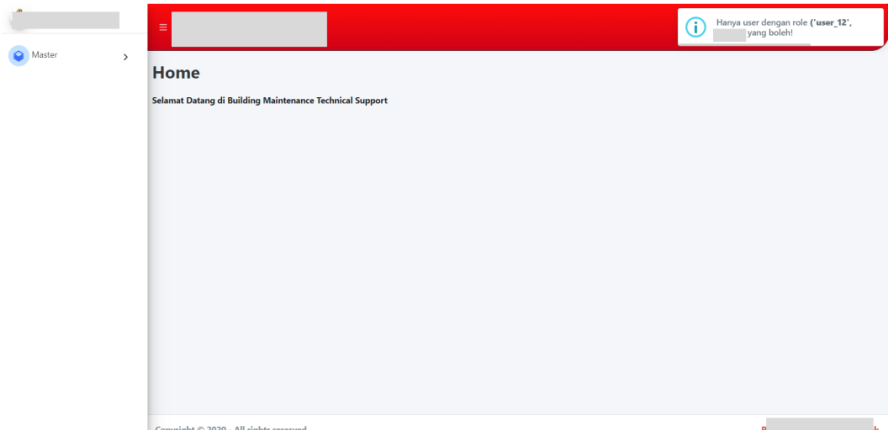

Gambar 5 Session Pengguna

Pada gambar 5 *session* memiliki *key* "roles" yang memiliki nilai array dimana elemennya adalah izin pengguna terhadap modul sehingga dapat dijelaskan bahwa pegawai dengan kode 672020193 memiliki akses ke menu dengan kode 1 dan kode 2 sebagai user atau hanya dapat membaca saja.



Gambar 6 Tampilan Sistem

Gambar 6 menampilkan antarmuka sistem otorisasi terpusat untuk mengelola hak akses. Pada sistem tersebut jabatan ”Crew” yang merupakan salah satu dari banyak jabatan di PT XYZ diberikan izin untuk mengakses menu master otorisasi, menu 1 dan menu 2. Pengaturan hak akses pada gambar tersebut diberikan berdasarkan jabatan, bukan pada pengguna. Pendekatan ini sesuai dengan konsep *Role-Based Access Control* (RBAC), di mana hak akses atau izin diberikan berdasarkan jabatan pengguna, bukan secara langsung pada pengguna itu sendiri.



Gambar 7 Tampilan pengguna tidak memiliki akses ke menu

Gambar 7 menyajikan antarmuka yang menunjukkan penerapan konsep *Role-Based Access Control* (RBAC). Ketika seorang pengguna sedang ingin mengakses sebuah menu, gambaran visual tersebut menampilkan pemberitahuan dengan pesan eksplisit tentang penolakan akses. Hal ini terjadi dikarenakan peran atau jabatan pengguna tidak cocok atau tidak diperbolehkan untuk mengakses menu tersebut.

JIPI (Jurnal Ilmiah Penelitian dan Pembelajaran Informatika)
Journal homepage: https://jurnal.stkippgritulungagung.ac.id/index.php/jipi
ISSN: 2540-8984
Vol. 9, No. 4, Desember 2024, Pp. 1768-1778

Setelah mengimpelentasikan ke kode program dan sistem otorisasi sudah terbangun maka untuk selanjutnya dilakukan pengujian sistem terhadap sistem otorisasi terpusat dengan menggunakan metode *test case*. Pendekatan pengujian ini melibatkan pengujian secara manual, di mana tim pengembang harus melakukan pengujian satu per satu terhadap setiap fitur atau fungsionalitas yang akan diuji. Penggunaan *test case* dalam pengujian mempermudah pengembangan proyek perangkat lunak, memastikan bahwa proyek tersebut sesuai dengan keinginan dan kebutuhan klien. Dalam implementasinya, *test case* digunakan untuk membandingkan hasil aktual dengan hasil yang diharapkan, memungkinkan tim pengembang untuk mengidentifikasi dan memperbaiki kesalahan yang terdapat dalam perangkat lunak[19]. Berikut adalah hasil pengujian yang telah dilakukan:

TABEL 2
PENGUJIAN MENGGUNAKAN TEST CASE

Fitur	ID	Deskripsi	Kondisi Awal	Tahapan Tes	Hasil yang Diharapkan	Hasil	Status
Create Hak Akses	TC01	Menambahkan hak akses baru dengan data yang valid	Sudah berada dihalaman otorisasi	1. Klik tombol “Tambah”. 2. Modal tampil. 3. Memilih Jabatan 4. Memilih menu 5. Klik tombol “Simpan”.	Kembali ke halaman tabel hak akses dan data hak akses baru akan muncul pada tabel hak akses	Sesuai	*Pass*
Update Hak Akses	TC02	Mengubah hak akses terpilih dengan data yang valid	Sudah berada dihalaman otorisasi dan data yang akan diubah sudah ada ditabel	1. Pilih data yang ingin diubah 2. Klik icon “edit”. 3. Modal tampil 4. Memilih menu 5. Klik tombol “Simpan”.	Kembali ke halaman tabel hak akses dan data hak akses yang sudah diubah akan muncul dibaris pertama	Sesuai	*Pass*
Delete Hak Akses	TC03	Mengapus hak akses terpilih	Sudah berada dihalaman otorisasi dan data yang akan dihapus sudah ada ditabel	1. Pilih data yang ingin dihapus 2. Klik ikon “delete”. 3. *Pop-up* konfirmasi tampil 4. Klik tombol “Yakin”.	Kembali ke halaman tabel hak akses dan data hak akses yang sudah dihapus tidak muncul di tabel	Sesuai	*Pass*
Akses Menu	TC04	Mengakses menu dengan kondisi bahwa jabatan pengguna memiliki akses ke menu tesebut.	Sudah berada dihalaman beranda dan pengguna memiliki hak akses yang sesuai	1. Memilih menu di sidebar. 2. Klik menu yang dipilih.	Pengguna dapat mengakses menu tersebut tanpa ada masalah	Sesuai	*Pass*
	TC05	Mengakses menu dengan kondisi bahwa jabatan pengguna tidak memiliki akses ke menu tesebut.	Sudah berada dihalaman beranda dan pengguna tidak memiliki hak akses yang sesuai	1. Memilih menu di sidebar. 2. Klik menu yang dipilih.	Pengguna tidak dapat mengakses menu tersebut dan muncul peringatan “Jabatan anda tidak memiliki akses ke menu”	Sesuai	*Pass*

Dalam tabel 2, terlihat bahwa seluruh fitur atau fungsi dalam sistem otorisasi terpusat telah memenuhi spesifikasi yang ditetapkan oleh perusahaan untuk tim pengembangan. Gambar tersebut juga menampilkan perbandingan antara hasil yang diharapkan dan hasil nyata. Informasi ini akan berguna bagi tim pengembang sebagai acuan untuk melakukan perbaikan pada fitur atau fungsi yang sedang dalam tahap pengujian.

Hasil pengembangan sistem otorisasi terpusat menunjukkan bahwa pengembangan sistem otorisasi terpusat menggunakan model *Role-Based Access Control* (RBAC) dan Flask sebagai *framework* pembuatan website telah memberikan banyak manfaat dalam pemberian hak akses. Penelitian ini bertujuan untuk mengatasi masalah yang dihadapi oleh PT XYZ, di mana sebelumnya hak akses diberikan secara langsung kepada pengguna, menyebabkan ketidakmampuan untuk mengakomodasi perubahan dinamis dalam kondisi perusahaan. Dalam implementasinya,

model RBAC memberikan pendekatan yang terstruktur dalam mengelola hak akses pengguna, meningkatkan keamanan sistem dengan memastikan bahwa setiap pengguna hanya memiliki akses yang sesuai dengan tugas dan tanggung jawab mereka. Pengaturan dan pembaruan peran serta hak akses dapat dilakukan secara terpusat, mengurangi waktu dan upaya dalam administrasi sistem. Namun, perlu diperhatikan bahwa RBAC memiliki keterbatasan, seperti kesulitan dalam menangani skenario akses yang kompleks dan kurang fleksibel dalam mengelola perubahan peran pengguna. Selain itu, dibandingkan dengan *Attribute-Based Access Control* (ABAC), RBAC memiliki keterbatasan dalam hal jumlah atribut yang dapat digunakan untuk mengatur akses. ABAC, yang memungkinkan pengaturan akses yang lebih dinamis dan kompleks berdasarkan atribut-atribut pengguna dan sumber daya, dapat menjadi pilihan yang lebih sesuai dalam lingkungan di mana kebutuhan akan kontrol akses yang granular sangat penting[20]. RBAC kurang efektif dalam beberapa situasi, seperti ketika sistem memiliki pengguna unik yang memiliki tanggung jawab mencakup beberapa peran, ketika sistem menghadapi logika bisnis yang kompleks yang sulit untuk didefinisikan dalam peran, ketika skala sistem meningkat menjadi besar dan memerlukan administrasi konstan untuk menyesuaikan izin peran, dan ketika otorisasi didasarkan pada parameter dinamis. Namun demikian, hasil pengujian menunjukkan bahwa sistem yang dikembangkan menggunakan pendekatan RBAC dan Flask mampu memenuhi kriteria yang diharapkan oleh PT XYZ. Secara keseluruhan, implementasi RBAC dan Flask telah membawa manfaat signifikan bagi PT XYZ dalam mengelola hak akses pengguna, meningkatkan keamanan, efisiensi operasional, dan kepatuhan dalam lingkungan organisasi yang kompleks.

IV. KESIMPULAN

Pada penelitian ini dilakukan pembangunan aplikasi Sistem Otorisasi Terpusat dengan mengimplementasikan model RBAC dan *framework* Flask. Berdasarkan hasil dan pembahasan yang ada dapat disimpulkan bahwa aplikasi Sistem Otorisasi Terpusat dapat digunakan sebagai langkah awal dalam pemecahan masalah jika terjadi kendala pada proses pemberian hak akses. Pemberian hak akses pada peran lebih meningkatkan efisiensi dan efektivitas dalam memberikan dan mencabut hak akses pada pengguna. Pada aplikasi yang dibangun masih terdapat kekurangan berupa tampilan aplikasi yang tidak responsif jika dibuka menggunakan perangkat yang memiliki ukuran layar kecil seperti Android dan iOS, sehingga tampilan program menjadi berantakan dan sulit digunakan. Pada pengembangan berikutnya diharapkan dapat memperhatikan pada bagian responsivitas tampilan aplikasi.

DAFTAR PUSTAKA

[1] H. Madiistriyatno, "Pengamanan Pengelolaan Hak Akses Web Berbasis Yii Framework," *Syntax J. Inform.*, vol. 7, no. 1, pp. 52–63, 2018.
[2] M. K. Kabier, A. A. Yassin, Z. A. Abduljabbar, and S. Lu, "Role Based Access Control Using Biometric the in Educational System," *Basrah Res. Sci.*, vol. 49, no. 1, pp. 85–101, 2023, doi: 10.56714/bjrs.49.1.8.
[3] J. Barkley, A. Cincotta, D. Ferraiolo, S. Gavrila, and R. Kuhn, "Role Based Access for the World Wide Web," *Natl. Inf. Syst. Secur. Conf.*, no. September 2014, 1997.
[4] B. Jayant.D, U. Swapnaja A, A. Sulabha S, and M. Dattatray G, "Analysis of DAC MAC RBAC Access Control based Models for Security," *Int. J. Comput. Appl.*, vol. 104, no. 5, pp. 6–13, 2014, doi: 10.5120/18196-9115.
[5] J. Zhao and J. Sun, "Research on Access Control Model Based on RBAC Model in Microservice Environment," *J. Phys. Conf. Ser.*, vol. 1437, no. 1, 2020, doi: 10.1088/1742-6596/1437/1/012031.
[6] M. U. Aftab, A. Nisar, M. Asif, A. Ashraf, and B. Gill, "RBAC architecture design issues in institutions collaborative environment," *Int. J. Comput. Sci. Issues*, vol. 10, no. 4, pp. 216–221, 2013.
[7] Rubiyanto, Selo, and Widyawan, "Implementasi Role-Based Access Control (Rbac) Pada Pemanfaatan Data Kependudukan Ditingkat Kabupaten," *Poster 021*, no. November, pp. 1–10, 2017.
[8] L. Dongdong, X. Shiliang, Z. Yan, T. Fuxiao, N. Lei, and Z. Jia, "Role-based access control in educational administration system," *MATEC Web Conf.*, vol. 139, pp. 1–8, 2017, doi: 10.1051/matecconf/201713900120.
[9] D. Ghimire, "Comparative study on Python web frameworks: Flask and Django," *Metrop. Univ. Appl. Sci.*, no. May, pp. 13–33, 2020, [Online]. Available: https://urn.fi/URN:NBN:fi:amk-2020052513398
[10] A. Sasturkar, P. Yang, S. D. Stoller, and C. R. Ramakrishnan, "Policy analysis for Administrative Role-Based Access Control," *Theor. Comput. Sci.*, vol. 412, no. 44, pp. 6208–6234, 2011, doi: 10.1016/j.tcs.2011.05.009.
[11] K. 'Afiifah, Z. F. Azzahra, and A. D. Anggoro, "Analisis Teknik Entity-Relationship Diagram dalam Perancangan Database Sebuah Literature Review," *Intech*, vol. 3, no. 2, pp. 18–22, 2022, doi: 10.54895/intech.v3i2.1682.
[12] D. F. Ferraiolo, J. F. Barkley, and D. R. Kuhn, "A role-based access control model and reference implementation within a corporate intranet," *ACM Trans. Inf. Syst. Secur.*, vol. 2, no. 1, pp. 34–64, 1999, doi: 10.1145/300830.300834.
[13] S. Osborn, R. Sandhu, and Q. Munawer, "Configuring Role-Based Access Control to Enforce Mandatory and Discretionary Access Control Policies," *ACM Trans. Inf. Syst. Secur.*, vol. 3, no. 2, pp. 85–106, 2000, doi: 10.1145/354876.354878.
[14] S. M. Pulungan, R. Febrianti, T. Lestari, N. Gurning, and N. Fitriana, "Analisis Teknik Entity-Relationship Diagram Dalam Perancangan Database," *J. Ekon. Manaj. dan Bisnis*, vol. 1, no. 2, pp. 98–102, 2023, doi: 10.47233/jemb.v1i2.533.
[15] MUDZAKKIR TOHA, "IMPLEMENTASI FRAMEWORK SPRING MVC UNTUK PEMBUATAN SISTEM INFORMASI MANAJEMEN E COMMERCE TUGAS," *Int. J. Dev. Manag. Rev.*, vol. 5, no. 1, pp. 212–224, 2010, [Online]. Available: http://publications.lib.chalmers.se/records/fulltext/245180/245180.pdf%0Ahttps://hdl.handle.net/20.500.12380/245180%0Ahttp://dx.doi.org/10.1016/j.jsames.2011.03.003%0Ahttps://doi.org/10.1016/j.gr.2017.08.001%0Ahttp://dx.doi.org/10.1016/j.precamres.2014.12
[16] B. Shaik and D. K. Chemuduru, "PL/pgSQL Essential Extensions BT - Procedural Programming with PostgreSQL PL/pgSQL: Design Complex Database-Centric Applications with PL/pgSQL," B. Shaik and D. K. Chemuduru, Eds., Berkeley, CA: Apress, 2023, pp. 293–309. doi:

10.1007/978-1-4842-9840-4_19.

[17] A. A. Wibowo Putri and Y. A. Susetyo, "Implementation of Flask for Stock Checking in Distribution Center & Store on Monitoring Stock Application in Pt. Xyz," *J. Tek. Inform.*, vol. 3, no. 5, pp. 1265–1274, 2022, doi: 10.20884/1.jutif.2022.3.5.334.

[18] S. Vivian and H. S. Rismon, "Pemrograman Web dengan PHP dan MySQL - Google Books," *Penerbit SPARTA*, no. January 2005, pp. 1–122, 2018.

[19] A. N. Hasibuan and T. Dirgahayu, "Pengujian dengan Unit Testing dan Test case pada Proyek Pengembangan Modul Manajemen Pengguna," *J. Inform. Univ. Islam Indones.*, vol. 2, no. 1, pp. 103–109, 2020.

[20] X. Jin, R. Krishnan, and R. Sandhu, "A Unified Attribute-Based Access Control Model Covering DAC, MAC and RBAC BT - Lecture Notes in Computer Science," *Lect. Notes Comput. Sci.*, vol. 7371, no. Chapter 4, pp. 41–55, 2012, [Online]. Available: http://dx.doi.org/10.1007/978-3-642-31540-4_4%5Cnpapers2://publication/doi/10.1007/978-3-642-31540-4_4