BAB I

PENDAHULUAN

A. Latar Belakang

Karang Taruna adalah organisasi sosial kepemudaan yang memiliki peran penting dalam mengembangkan potensi generasi muda di Indonesia. Organisasi ini berfungsi sebagai wadah bagi para pemuda untuk berpartisipasi aktif dalam berbagai kegiatan sosial, budaya, dan ekonomi yang bermanfaat bagi masyarakat setempat. Di Kota Parepare, Karang Taruna telah berkontribusi dalam mengatasi berbagai permasalahan sosial, mempromosikan kebudayaan lokal, serta meningkatkan kesejahteraan ekonomi masyarakat melalui berbagai program yang inovatif dan berkelanjutan.

Namun, pengelolaan kegiatan Karang Taruna seringkali menghadapi berbagai tantangan, terutama dalam hal manajemen informasi dan koordinasi antaranggota. Banyak kegiatan yang berjalan tidak efektif karena kurangnya sistem yang mampu mengintegrasikan data dan informasi secara menyeluruh. Proses pencatatan kegiatan, pengelolaan sumber daya, dan komunikasi antaranggota masih banyak dilakukan secara manual, sehingga sering menimbulkan kendala dalam pelaksanaan dan evaluasi kegiatan.

Dalam era digital saat ini, teknologi informasi memegang peranan vital dalam meningkatkan efisiensi dan efektivitas organisasi. Penggunaan teknologi berbasis *website* dapat menjadi solusi dalam mengatasi berbagai permasalahan manajemen yang dihadapi oleh Karang Taruna. Sistem informasi manajemen yang

berbasis *website* memungkinkan pengelolaan data dan informasi yang lebih terstruktur, mudah diakses, dan terintegrasi dengan baik. Dengan adanya sistem ini, diharapkan setiap kegiatan dapat tercatat dengan rapi dan informasi dapat tersebar dengan cepat dan akurat kepada seluruh anggota.

Pengelolaan kegiatan Karang Taruna yang selama ini masih dilakukan secara manual seringkali menimbulkan berbagai kendala. Kesulitan dalam pencatatan dan pelacakan kegiatan menyebabkan informasi yang tidak akurat dan sulit diakses kembali ketika diperlukan. Penyebaran informasi yang lambat membuat anggota seringkali tidak mendapatkan informasi terkini mengenai kegiatan yang akan atau sedang berlangsung. Selain itu, kurangnya transparansi dan akuntabilitas dalam pengelolaan dana dan sumber daya dapat menimbulkan ketidakpercayaan di kalangan anggota dan masyarakat.

Dengan adanya sistem informasi manajemen berbasis *website*, diharapkan proses pengelolaan kegiatan dapat dilakukan dengan lebih efisien, cepat, dan transparan. Sistem ini memungkinkan anggota Karang Taruna untuk mengakses informasi , sehingga dapat melakukan koordinasi dengan lebih efektif.

Sistem informasi manajemen berbasis *website* diharapkan dapat membantu para anggota Karang Taruna dalam meningkatkan partisipasi aktif mereka dalam setiap kegiatan yang diselenggarakan. Dengan akses informasi yang mudah dan cepat, anggota dapat lebih terlibat dalam perencanaan, pelaksanaan, dan evaluasi kegiatan. Hal ini juga dapat meningkatkan rasa memiliki dan tanggung jawab 3 anggota terhadap organisasi, sehingga kegiatan Karang Taruna dapat berjalan lebih lancar dan mencapai tujuan yang telah ditetapkan.

B. Rumusan Masalah

Berdasarkan latar belakang masalah yang dipaparkan, rumusan masalah dalam penelitian ini adalah: bagaimana merancang dan membangun sistem informasi manajemen pengelolaan kegiatan karang taruna berbasis *website* di Kota Parepare.

C. Batasan Masalah

Berdasarkan rumusan masalah di atas, maka batasan masalah dalam penelitian ini adalah sebagai berikut:

1. Penelitian ini dibatasi hanya pada pengembangan sistem informasi manajemen berbasis *website*. Sistem ini tidak dikembangkan untuk *platform* lain seperti aplikasi *mobile* (Android atau iOS) atau *desktop*.
2. Sistem informasi tidak mencakup pengelolaan kegiatan Karang Taruna di kota atau daerah lain.
3. Fitur yang disediakan dalam sistem ini hanya meliputi pencatatan kegiatan, penyebaran informasi, pengelolaan keanggotaan, serta transparansi pengelolaan dana dan sumber daya. Fitur-fitur lain seperti pelatihan *online*, atau *e-commerce* tidak termasuk dalam ruang lingkup penelitian ini.

D. Tujuan Penelitian

Penelitian ini bertujuan untuk merancang dan mengembangkan sebuah sistem informasi manajemen pengelolaan kegiatan Karang Taruna berbasis *website* yang dapat diakses oleh anggota Karang Taruna Kota Parepare. Sistem ini diharapkan dapat meningkatkan efisiensi dan efektivitas pengelolaan kegiatan

Karang Taruna melalui pengelolaan data dan informasi yang lebih terstruktur, mudah diakses, dan terintegrasi dengan baik.

E. Manfaat Penelitian

Adapun manfaat yang dapat diberikan dari penelitian ini adalah sebagai berikut:

1. Bagi akademik

   Sebagai bahan referensi bagi peneliti lain untuk mengembangkan kemampuan di bidang pengembangan sistem informasi manajemen

2. Bagi penulis

   Penulis dapat mengimplementasikan ilmu yang didapat selama proses perkuliahan.

3. Bagi masyarakat

   Masyarakat dapat merasakan manfaat dari kegiatan Karang Taruna yang lebih terorganisir dan efektif, yang pada akhirnya dapat meningkatkan partisipasi dan koordinasi antar anggota dalam berbagai kegiatan yang diselenggarakan di Kota Parepare.

BAB II

TINJAUAN PUSTAKA

A. Kajian Teori

1. Sistem Informasi

Sistem informasi (SI) merupakan kumpulan komponen yang saling terkait, termasuk perangkat keras, perangkat lunak, data, prosedur, dan manusia, yang bekerja sama untuk mengumpulkan, memproses, menyimpan, dan mendistribusikan informasi untuk mendukung pengambilan keputusan dan pengendalian organisasi. Sederhananya, Sistem informasi adalah tulang punggung dari setiap bisnis modern, memungkinkan mereka beroperasi secara efisien dan efektif. Menurut Setiawan (2022), sistem informasi bukan sekadar kumpulan data, melainkan juga sebagai alat strategis yang dapat mengubah cara organisasi beroperasi.

Dalam era digital saat ini, Sistem informasi memainkan peran yang semakin krusial dalam berbagai aspek bisnis. Mulai dari pengelolaan data pelanggan, inventory, dan keuangan hingga otomatisasi proses bisnis dan analisis data, SI memungkinkan organisasi untuk membuat keputusan yang lebih baik, meningkatkan produktivitas, dan memperoleh keunggulan kompetitif:

Sistem Informasi terdiri dari beberapa komponen utama yang saling berinteraksi:

a. Perangkat Keras: Komponen fisik seperti komputer, *server*, jaringan, dan perangkat input/output
b. Perangkat lunak: *Program* yang mengontrol operasi perangkat keras dan memproses data, misalnya sistem operasi, aplikasi bisnis, dan *database*.
c. Data: Informasi yang dikumpulkan, disimpan, dan diproses oleh sistem, seperti data pelanggan, transaksi, dan laporan keuangan.
d. Prosedur: Kumpulan langkah-langkah yang mengatur penggunaan sistem, termasuk input data, pemrosesan data, dan distribusi informasi
e. Manusia: Individu yang berinteraksi dengan sistem, seperti pengguna, *admin*istrator, dan *programmer*.

Sistem informasi dapat diklasifikasikan berdasarkan fungsi dan penggunaannya, antara lain:

a. Sistem Informasi Manajemen (SIM): Menyediakan informasi kepada manajer untuk mendukung pengambilan keputusan.
b. Sistem Pendukung Keputusan (SPK): Membantu manajer dalam memecahkan masalah yang kompleks dan tidak terstruktur.
c. Sistem Informasi Akuntansi: Mengelola transaksi keuangan dan menghasilkan laporan keuangan.
d. Sistem Informasi Sumber Daya Manusia (SISDM): Mengelola data karyawan, seperti gaji, tunjangan, dan kinerja.
e. Sistem Informasi Produksi: Mengelola proses produksi, mulai dari perencanaan hingga pengiriman produk.

Dengan semakin tingginya ketergantungan pada SI, keamanan informasi menjadi isu yang sangat penting. Pelanggaran data dapat menyebabkan kerugian finansial yang besar, kerusakan reputasi, dan bahkan ancaman terhadap kelangsungan bisnis. Pratiwi (2021) berpendapat bahwa implementasi sistem informasi yang sukses memerlukan perencanaan yang matang dan melibatkan seluruh pemangku kepentingan. Oleh karena itu, organisasi perlu menerapkan langkah-langkah keamanan yang komprehensif untuk melindungi data mereka dari ancaman siber.

2. Karang Taruna

Karang Taruna adalah organisasi sosial yang menjadi wadah pengembangan generasi muda yang tumbuh dan berkembang atas dasar kesadaran dan tanggung jawab sosial dari, oleh, dan untuk masyarakat, terutama generasi muda di wilayah desa/kelurahan, dan terutama bergerak di bidang usaha kesejahteraan sosial. Karang Taruna merupakan organisasi sosial dan perkumpulan sosial yang dibentuk oleh masyarakat dan berfungsi sebagai sarana partisipasi masyarakat dalam melaksanakan Usaha Kesejahteraan Sosial (UKS).

Sebagai wadah pengembangan generasi muda, Karang Taruna adalah tempat diselenggarakannya berbagai upaya atau kegiatan untuk meningkatkan dan mengembangkan cipta, rasa, karsa, dan karya generasi muda dalam rangka pengembangan sumber daya manusia (SDM). Karang Taruna tumbuh dan berkembang atas dasar adanya kesadaran terhadap keadaan dan permasalahan di lingkungannya serta adanya tanggung jawab sosial untuk turut berusaha

menanganinya. Kesadaran dan tanggung jawab sosial tersebut merupakan modal dasar tumbuh dan berkembangnya Karang Taruna.

Karang Taruna tumbuh dan berkembang dari generasi muda, diurus atau dikelola oleh generasi muda, dan untuk kepentingan generasi muda dan masyarakat di wilayah desa/kelurahan atau komunitas adat sederajat. Karenanya, setiap desa/kelurahan atau komunitas adat sederajat dapat menumbuhkan dan mengembangkan Karang Tarunanya sendiri. Gerakannya di bidang Usaha Kesejahteraan Sosial berarti bahwa semua upaya, program, dan kegiatan yang diselenggarakan Karang Taruna ditujukan guna mewujudkan kesejahteraan sosial masyarakat, terutama generasi mudanya. Keberadaan Karang Taruna dengan berbagai kegiatan yang dilaksanakan selama ini bertumpu pada landasan hukum yang dimiliki, yang terus diperbaharui sesuai dengan tuntutan, kebutuhan, dan perkembangan masalah kesejahteraan sosial serta sistem pemerintahan yang terjadi.

3. *HTML*



Menurut (Nasution & Hasan M.A., 2018) HTML5 merupakan bahasa markah untuk mendesain dan memperlihatkan isi dari waring wera wanua, sebuah teknologi inti dari internet. HTML5 merupakan perbaikan kelima dari HTML (yang pertama kali dibuat di tahun 1990 dan versi keempatnya,

HTML4, di tahun 1997) dan sampai bulan juni 2011 masih dalam ekspansi. Tujuan utama dari ekspansi HTML5 ialah memberikan perbaikan pada teknologi HTML untuk menunjang teknologi multimedia yang terbaru, mudah dibaca oleh manusia dan juga dimengerti oleh mesin. HTML adalah bahasa pemrograman yang menjadi landasan dari terbentuknya *web*, yang mana secara bahasa HTML adalah bahasa dasar untuk *web scripting* bersifat client side untuk memperlihatkan data dalam bentuk teks, grafik, multimedia dan juga untuk merangkaikan antar tampilan hyperlink. (Bagye, 2020)

(Nirwan & MS, 2020) mengatakan bahwa HTML(Hypertext Markup Language) adalah suatu bahasa yang digunakan untuk membuat suatu halaman *website*, menunjukkan bermacam data di dalam suatu penjelajah *web* internet serta menerapkan format hiperteks simpel yang ditulis dalam berkas ASCII supaya bisa menciptakan tampilan yang terintegrasi. Dengan kata lain, berkas terbuat dalam fitur lunak pengolah kata beserta disimpan dalam format ASCII normal sehingga menjadi halaman *website* dengan perintah HTML.

4. Bootstrap



Gambar 2. 2 Logo Bootstrap

Bootstrap merupakan sebuah *framework* css yang memudahkan pengembang untuk membangun *website* yang menarik dan responsif. Tidak konsistensinya terhadap aplikasi individual membuat sulitnya untuk

mengembangkan dan pemeliharaannya. Bootstrap adalah css tetapi dibentuk dengan LESS, sebuah pre-prosessor yang memberi fleksibilitas dari css biasa. Bootstrap dapat dikembangkan dengan tambahan lainnya karena ini cukup fleksibel terhadap pekerjaan design butuhkan. (Sadewo, 2022)

Selain itu, Bootstrap juga memiliki fitur yang mencakup *library* dari JavaScript. Untuk Penggunaan dari framework ini digunakan untuk membantu dalam menyusun program aplikasi pada sisi *front end* (*client – side*). Untuk sekarang, Bootstrap sangat diminati oleh berbagai pengembang *web* melalui platform *Github* untuk membantu proses pembuatan desain aplikasi atau *website* yang lebih komprehensif dan modern.

5. *Website*

Website (situs *web*) merupakan alamat (URL) yang dapat digunakan untuk menyimpan data dan informasi berdasarkan topik tertentu. *Website* adalah kumpulan dari halaman situs dan dokumen yang tersebar di beberapa komputer *server* yang berada di penjuru dunia dan terhubung menjadi satu jaringan melalui jalur internet. *Website* adalah sekumpulan halaman yang terdiri dari beberapa halaman yang berisi informasi dalam bentuk data digital, baik berupa teks, gambar, video, dan animasi lainnya yang tersedia melalui jalur koneksi internet. (Ronaldo & Pasha, 2021) menyatakan bahwa terdapat dua jenis *website* diantaranya:

a. *Website* statis adalah *website* yang isinya tidak mudah untuk diganti oleh penggunanya, harus dengan mengubah cara coding di halaman *website* (merubah *database*).

b. *Website* dinamis adalah *website* yang mudah untuk beradaptasi dan otomatis dapat disesuaikan perubahan konten secara langsung tanpa ada perubahan cara coding di halaman *website*.

(Manurian et al., n.d.) mengatakan bahwa *website* dibagi menjadi dua jenis berdasarkan isi atau kontennya, yaitu:

a. *Website* Statis

Website statis merupakan *web* yang isinya tidak dapat diubah dengan cepat dan mudah. Isi dari *website* dinamis hanya dapat diubah dengan cara melakukan perubahan langsung pada isinya di file mentah *web* tersebut. Bahasa pemrograman yang biasa digunakan pada *web* ini adalah jenis client side scripting seperti HTML, Cascading Style Sheet (CSS).

b. *Website* Dinamis

Website dinamis adalah salah satu jenis *web* yang isinya dapat berubah setiap saat. Perubahan isi pada *website* dinamis dapat dilakukan secara langsung di internet melalui halaman control panel atau *admin*istrasi yang tersedia untuk *admin*istrator selama *user* memiliki hak akses yang sesuai. Bahasa pemrograman untuk *website* ini adalah PHP, ASP, NET, dan memanfaatkan *database* MySQL.

6. MySQL



MySQL adalah salah satu jenis *database* yang terkenal dan termasuk jenis RDBMS (Relational *Database* Management System). Kepopuleran MySQL disebabkan karena MySQL menggunakan bahasa SQL sebagai bahasa dasar untuk *query* dan bersifat *open source* di berbagai platform.

MySQL adalah *Relational Database Management System* (RDBMS) yang didistribusikan secara gratis dibawah lisensi GPL (*General Public License*), di mana setiap orang bebas untuk menggunakan MySQL, namun tidak boleh dijadikan produk turunan yang bersifat *closed source* atau komersial. MySQL sebenarnya merupakan turunan salah satu konsep utama dalam *database* sejak lama, yaitu SQL (*Structured Query Language*). SQL adalah sebuah konsep pengoperasian *database*, terutama untuk pemilihan atau seleksi dan pemasukan data yang memungkinkan pengoperasian data dikerjakan dengan mudah secara otomatis.

Sebagai *database server*, MySQL dapat dikatakan lebih unggul dibandingkan *database server* lainnya, terutama dalam kecepatan. Berikut beberapa keistimewaan MySQL, antara lain:

a. *Portability*

   MySQL dapat berjalan stabil pada berbagai sistem operasi seperti Windows, Linux, FreeBSD, Mac OS X *Server*, Solaris Amiga, dan masih banyak lagi.

b. *Multiuser*

   MySQL dapat digunakan oleh beberapa pengguna (*user*) dalam waktu yang bersamaan tanpa mengalami masalah atau konflik.

c. *Security*

MySQL memiliki beberapa lapisan keamanan seperti level subnetmask, nama host, dan izin akses pengguna dengan sistem perizinan yang mendetail serta kata sandi terenkripsi.

d. *Scalability* dan *Limits*

MySQL mampu menangani *database* dalam skala besar, dengan jumlah rekaman lebih dari 50 juta dan 60 ribu tabel serta 5 miliar baris. Selain itu, batas indeks yang dapat ditampung mencapai 32 indeks pada setiap tabelnya (Huda, 2022).

7. *PHP*



PHP adalah bahasa *server-side* yang secara khusus digunakan untuk aplikasi *web*. Bahasa PHP berjalan di sisi *server*, artinya ia berjalan pada *server* dan bisa dimasukkan di antara bahasa HTML (*HyperText Markup Language*). Oleh karena itu, kode PHP tidak akan terlihat lagi dan hasil akhir dalam bentuk HTML akan dikirimkan ke *browser* (Sutarman, 2020). Karena PHP dijalankan di komputer *server*, itu dikenal sebagai bahasa pemrograman sisi *server*. Ini berbeda dengan bahasa pemrograman sisi klien, seperti JavaScript, yang beroperasi pada klien, yaitu *web browser*.

Pada awalnya, PHP adalah singkatan dari Personal *Home Page*. PHP digunakan untuk mengembangkan situs *web* pribadi, sesuai dengan namanya.

Seiring berjalannya waktu, PHP berkembang menjadi bahasa pemrograman *web* yang sangat kuat, tidak hanya digunakan untuk membuat halaman *web* dasar tetapi juga untuk situs-situs populer seperti Wikipedia, WordPress, Joomla, dan lainnya yang dikunjungi oleh jutaan orang.

8. *PhpMyAdmin*

Phpmyadmin adalah sebuah aplikasi *open source* yang berfungsi untuk memudahkan manajemen *MySQL*. Dengan menggunakan *phpmyadmin*, anda dapat membuat *database*, membuat tabel, meng-*insert*, menghapus dan meng-*update data* dengan *GUI* dan terasa lebih mudah, tanpa perlu mengetikkan perintah *SQL* secara *manual*. Karena berbasis *web*, maka *phpmyadmin* dapat dijalankan di banyak *OS*, selama dapat menjalankan *web server* dan *MySQL*

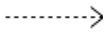
9. *Unified modelling language*

UML (*Unified Modeling Language*) adalah alat desain sistem berorientasi objek. Secara filosofis kemunculan UML terinspirasi dari konsep yang sudah ada yaitu konsep pemodelan *Object Oriented* (OO), karena konsep ini menganalogikan sistem seperti kehidupan nyata yang didominasi oleh objek dan digambarkan atau dicatat dalam simbol-simbol yang cukup spesifik. proses standar dan independen.

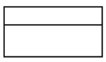
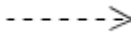
UML adalah pemodelan untuk membantu proses perancangan sistem sehingga meminimalisir kesalahan dalam membuat program. Penerapan UML ini menggambarkan struktur aktor yang terlibat, aktivitas setiap aktor, proses dan mekanisme dimana memberikan kemudahan perancangan dalam membantu proses pengkodingan menjadi sebuah aplikasi (Voutama, 2022).

UML memiliki tiga kategori utama yaitu *diagram* struktur, *diagram* perilaku, dan *diagram* interaksi. Adapun daftar simbol UML yaitu:

Tabel 2. 1 *Symbol Usecase Diagram*

NO.	GAMBAR	NAMA	KETERANGAN
1		*Actor*	Menspesifikasikan himpunan peran yang pengguna mainkan ketika berinteraksi dengan *usecase*.
2		*Dependency*	Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri (*independent*) akan mempengaruhi elemen yang bergantung padanya elemen yang tidak mandiri (*independent*).
4		*Generalization*	Menspesifikasikan bahwa *usecase* sumber secara eksplisit.
5		*Include*	Menspesifikasikan bahwa *usecase* target memperluas perilaku dari *usecase* sumber pada suatu titik yang diberikan.
6		*Extend*	Apa yang menghubungkan antara objek satu dengan objek lainnya.
7		*Association*	Menspesifikasikan paket yang menampilkan sistem secara terbatas.
8		*Usecase*	Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu aktor
9		*Collaboration*	Interaksi aturan-aturan dan elemen lain yang bekerja sama untuk menyediakan perilaku yang lebih besar dari jumlah dan elemen-elemennya (sinergi).
10		*Note*	Elemen fisik yang eksis saat aplikasi dijalankan dan mencerminkan suatu sumber daya komputasi

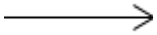
Tabel 2. 2 *Symbol Class Diagram*

NO.	GAMBAR	NAMA	KETERANGAN
1		*Generalization*	Hubungan dimana objek anak *(descendent)* berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk (*ancestor*).
2		*Nary Association*	Upaya untuk menghindari asosiasi dengan lebih dari 2 objek.
3		*Class*	Himpunan dari objek-objek yang berbagi atribut serta operasi yang sama.
4		*Collaboration*	Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu *actor*
5		*Realization*	Operasi yang benar-benar dilakukan oleh suatu objek.
6		*Dependency*	Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri *(independent)* akan mempengaruhi elemen yang bergantung padanya elemen yang tidak mandiri
7		*Association*	Apa yang menghubungkan antara objek satu dengan objek lainnya

Tabel 2. 3 *Symbol Sequence Diagram*

No	GAMBAR	NAMA	KETERANGAN
1		*LifeLine*	Objek *entity*, antarmuka yang saling berinteraksi.
2		*Message*	Spesifikasi dari komunikasi antar objek yang memuat in*form*asi-in*form*asi tentang aktivitas yang terjadi
3		*Message*	Spesifikasi dari komunikasi antar objek yang memuat in*form*asi-in*form*asi tentang aktivitas yang terjadi

Tabel 2. 4 *Symbol State Chart Diagram*

No	GAMBAR	NAMA	KETERANGAN
1		*State*	Nilai atribut dan nilai *Link* pada suatu waktu tertentu, yang dimiliki oleh suatu objek.
2		*Initial Pseudo State*	Bagaimana objek dibentuk atau diawali
3		*Final State*	Bagaimana objek dibentuk dan dihancurkan
4		*Transition*	Sebuah kejadian yang memicu sebuah *state* objek dengan cara memperbaharui satu atau lebih nilai atributnya
5		*Association*	Apa yang menghubungkan antara objek satu dengan objek lainnya.
6		*Node*	Elemen fisik yang eksis saat aplikasi dijalankan dan mencerminkan suatu sumber daya komputasi.

Tabel 2. 5 *Symbol Activity diagram*

No.	GAMBAR	NAMA	KETERANGAN
1		*Activity*	Memperlihatkan bagaimana masing-masing kelas antarmuka saling berinteraksi satu sama lain
2		*Action*	*State* dari sistem yang mencerminkan eksekusi dari suatu aksi
3		*Initial Node*	Bagaimana objek dibentuk atau diawali.
4		*Activity Final Node*	Bagaimana objek dibentuk dan dihancurkan
5		*Fork Node*	Satu aliran yang pada tahap tertentu berubah menjadi beberapa aliran

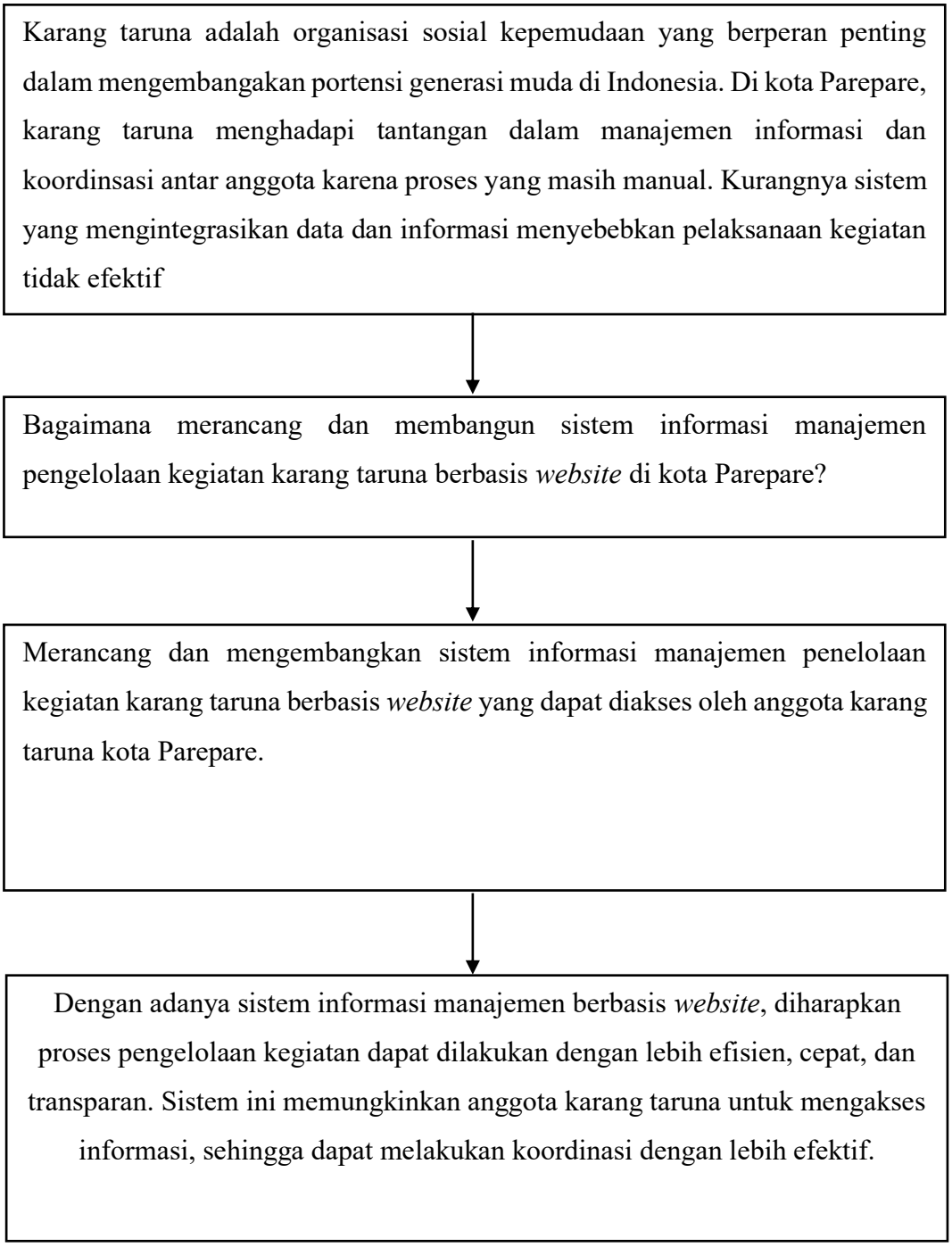
B. Kajian Hasil Penelitian Terdahulu

1. (Sudesta, 2021). "Sistem Informasi Karang Taruna Berbasis *Web* Sebagai Media Komunikasi Warga Masyarakat Pada Kantor Kelurahan Bojong Jaya." Penelitian ini bertujuan untuk mengembangkan sistem informasi Karang Taruna berbasis *web* yang berfungsi sebagai media komunikasi bagi

warga masyarakat. Sistem ini mencakup fitur seperti *login*, mengelola profil, potensi daerah, berita, agenda, dan galeri foto kegiatan. Sistem ini diimplementasikan menggunakan PHP dan MySQL, dengan metode perancangan *Unified Modeling Language* (UML).

2. (Nur Amanda, 2021). "Rancang Bangun Sistem Informasi Media Kegiatan Pada Lembaga Karang Taruna Kecamatan Sepatan Timur." Penelitian ini fokus pada pembuatan sistem informasi untuk mengelola kegiatan Karang Taruna. Sistem ini dikembangkan dengan metode Waterfall dan mencakup fitur-fitur untuk mengelola data kegiatan, anggota, dan publikasi informasi secara efektif.
3. (Rapiyanta & Regen, 2021). "Sistem Informasi Pengelolaan Kegiatan Karang Taruna (SIPEKATAR) Naralatu Agantuka Berbasis *Website*." Penelitian ini bertujuan untuk merancang dan membangun sistem informasi pengelolaan kegiatan Karang Taruna berbasis *web*. Sistem ini dirancang untuk meningkatkan efisiensi pengelolaan data kegiatan, menyediakan akses informasi yang cepat dan akurat bagi anggota, serta meningkatkan transparansi dalam pengelolaan dana dan sumber daya.

C. Kerangka Pikir



BAB III

METODE PENELITIAN

A. Jenis Penelitian

Jenis penelitian yang digunakan adalah penelitian kualitatif deskriptif, dimana informasi yang dikumpulkan berbentuk kata-kata, gambar, bukan angka.Dengan pendekatan field research dimana penelitian ini dilakukan dengan cara mengambil informasi di lapangan.

B. Lokasi dan Waktu Penelitian

Penelitian dilakukan di Sekretariat Karang Taruna Kota Parepare yang berlokasi di Jl. Jendral Sudirman No.12 Kelurahan Bumi Harapan. Penelitian ini akan berlangsung selama ± 2 (dua) bulan di tahun 2024.

C. Alat dan Bahan Penelitian

1. Perangkat keras

a. Laptop Acer Aspire Vero AV16 Ultra 7 155U:

- *Processor* : Intel® Core™ i5-1155G7 CPU @ 2.50GHz
- *Installed RAM* : 8GB
- *SSD* : 512GB

2. Perangkat lunak

a. Windows 10 Home Single Language

b. Visual Studio Code

c. XAMPP

d. PHPmy*Admin*

D. Teknik Pengumpulan Data

1. Studi pustaka

Studi pustaka merupakan strategi pengumpulan data dari berbagai sumber literatur termasuk buku, internet, makalah, jurnal dan bahan lain yang relevan dengan penelitian yang dilakukan.

2. Observasi

Observasi dilakukan dengan cara mengumpulkan data-data yang akan dijadikan bahan dasar dalam perancangan sistem.

3. Wawancara

Wawancara yaitu teknik mengumpulkan data dengan mencari atau memilah-milah informasi dalam literatur yang relevan dengan rumusan masalah.

E. Metode Pengujian

Dalam penelitian ini, digunakan 2 (dua) metode dalam pengujian datanya yaitu *Blackbox testing* dan *Whitebox testing*:

1. *Blackbox testing*

Blackbox testing berfokus pada fungsional dari program yang ada. Pada *Blackbox testing* diuji dengan cara menjalankan program kemudian diamati apakah program tersebut apakah berhasil atau tidak. *Blackbox testing* menggunakan teknik equivalence partitions yang merupakan pengujian berdasarkan masukan setiap menu yang terdapat pada program, setiap menu masukan dilakukan pengujian melalui klasifikasi dan pengelompokan berdasar fungsinya.

2. *Whitebox testing*

Whitebox testing bertujuan untuk mengetahui apakah struktur pada aplikasi yang dibuat sudah sesuai dengan ketentuan. *Whitebox testing* menitikberatkan pada pengujian dengan mengecek detail perancangan perangkat lunak. *Whitebox testing* dimulai dengan mendefinisikan semua alur dari perangkat lunak, kemudian membangun kasus yang akan digunakan dalam proses pengujian, kemudian menguji kasus tersebut untuk memperoleh hasilnya.

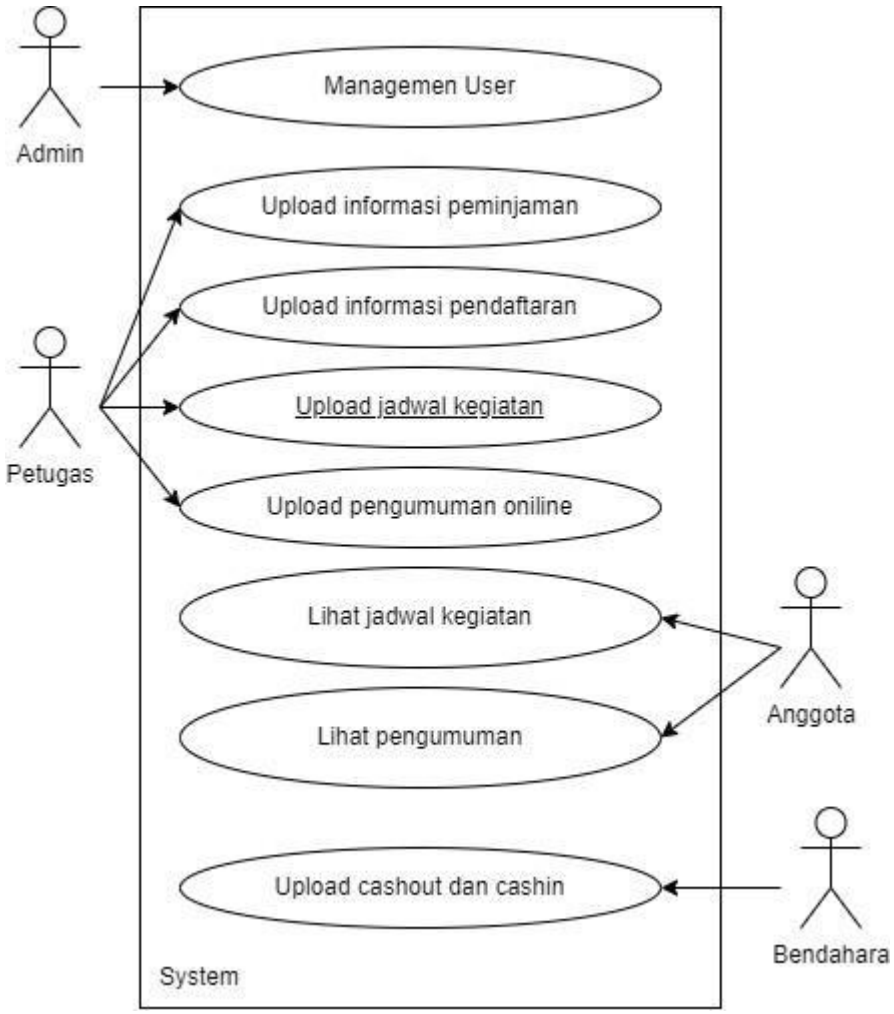
F. Desain Sistem

1. Desain sistem yang berjalan



Gambar 3. 1 *Usecase* sistem yang berjalan

2. Sistem yang diusulkan



Gambar 3. 2 *Usecase* sistem yang diusulkan

Untuk sistem yang diusulkan memiliki beberapa jenis *user* dimana ada *admin*, petugas, anggota, dan bendahara. Dimana untuk *admin* memiliki fitur untuk manajemen *user* dimana bisa melihat, membuat, mengubah, dan menghapus *user*. Untuk petugas sendiri memiliki fitur untuk *upload* informasi peminjaman, informasi pendaftaran, serta pengumuman *online*. Untuk anggota memiliki fitur untuk melihat jadwal kegiatan dan pengumuman sedangkan untuk *user* bendahara memiliki fitur untuk mengelola keuangan.

BAB IV

HASIL DAN PEMBAHASAN

A. Analisis Aliran Data UML

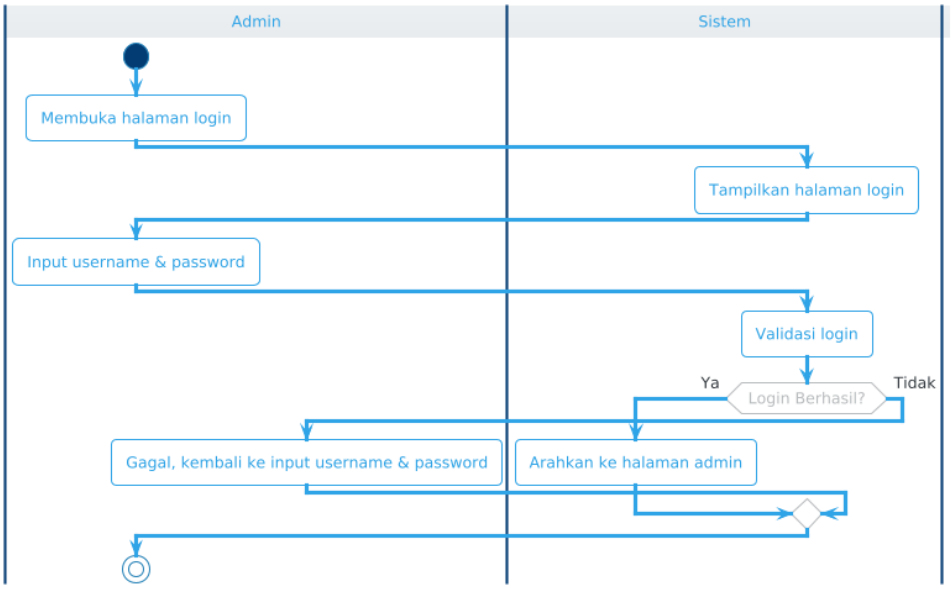
Dalam bagian ini, akan dibahas dua *diagram* UML yang menggambarkan aliran data dan interaksi dalam aplikasi Ensiklopedia Masakan Nusantara, yaitu *Activity diagram* dan *Sequence diagram*.

1. *Activity diagram*

Activity diagram digunakan untuk memodelkan aliran kerja atau aliran kontrol dari sebuah proses bisnis atau *usecase*. *Diagram* ini menunjukkan aktivitas-aktivitas yang terjadi dalam sistem dan urutan eksekusinya.

a. *Activity diagram admin*

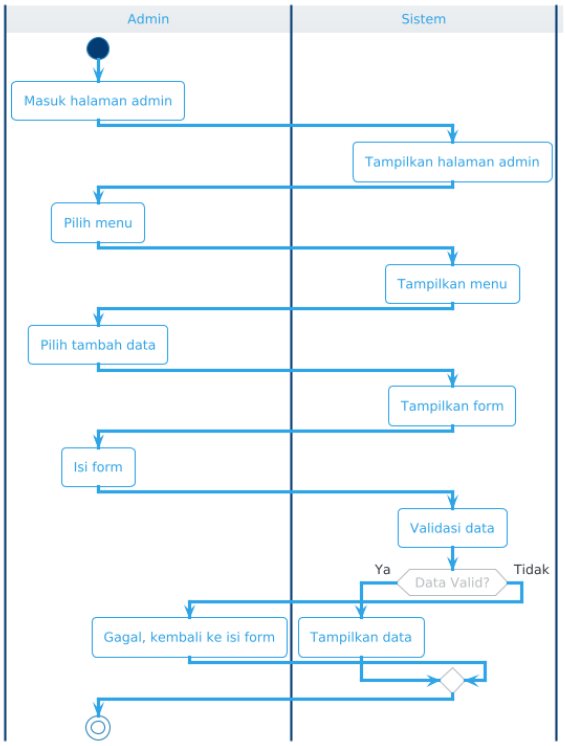
1) *Activity diagram login*



Gambar 4. 1 *Activity diagram login*

Pada gambar 4. 1 menjelaskan cara masuk sebagai *admin*. *Admin* harus terlebih dahulu mengakses situs *web*, setelah itu sistem akan menampilkan *form login* dan *admin* harus memasukkan nama pengguna dan kata sandi. Sistem kemudian akan melakukan validasi; jika informasi benar, maka akan masuk ke halaman *admin*; jika tidak, maka akan kembali ke halaman *login*.

2) *Activity diagram* tambah *user*

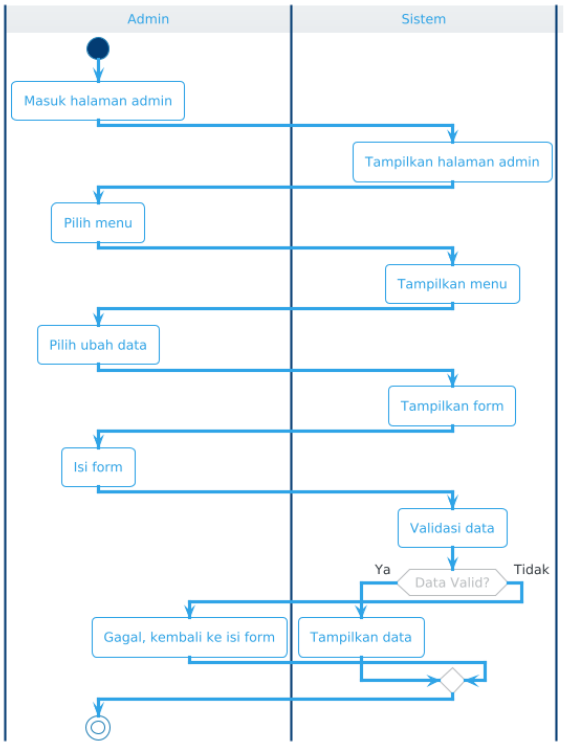


Gambar 4. 2 *Activity diagram* tambah *user*

Pada gambar 4. 2 menjelaskan langkah-langkah yang digunakan oleh *admin* untuk menambahkan *user*. Halaman *admin* ditampilkan oleh sistem ketika telah dibuka oleh *admin*. *Admin* kemudian memilih item menu tambah *user*, setelah itu sistem menampilkan halaman menu yang dipilih *admin* dan *admin* memilih tambah *user*. Form tambah

user kemudian akan ditampilkan oleh sistem. *Admin* selanjutnya diminta untuk melengkapi *form* tambah *user*. Setelah selesai, sistem akan mengecek data dan jika berhasil, sistem akan menampilkan halaman menu untuk *user* yang ditambahkan. Jika tidak berhasil, *admin*istrator harus mengisi formulir penambahan data dengan benar sekali lagi.

3) *Activity diagram* ubah data *user*

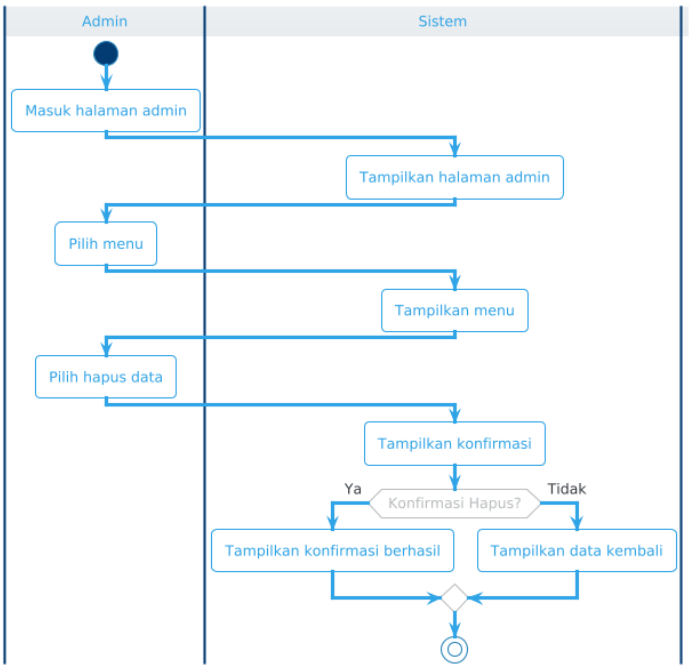


Gambar 4. 3 *Activity diagram* ubah data *user*

Pada gambar 4. 3 menjelaskan langkah-langkah yang harus dilakukan oleh *admin* untuk memodifikasi data. Hal pertama yang dilakukan *admin* adalah membuat folder *admin*. Kemudian, sistem akan menampilkan menu *admin*. Selanjutnya, *admin* memilih menu yang akan diubah datanya; sistem akan menampilkan judul menu yang

dipilih *admin*; terakhir, *admin* mengubah data. Sistem akan menampilkan data dalam bentuk tabel. Setelah *admin* mengisi *form* data, sistem akan memvalidasi informasi tersebut. Jika validasi berhasil, sistem akan menampilkan item menu dengan data yang dimasukkan, dan jika tidak, *admin*istrator disarankan untuk mengisi form data kembali dengan akurat.

4) *Activity diagram* hapus data *user*



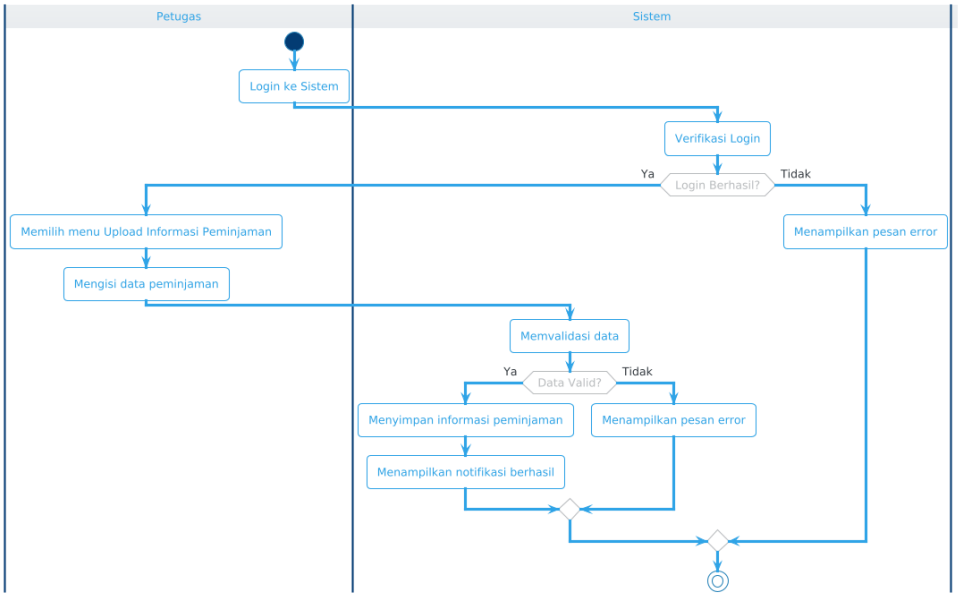
Gambar 4. 4 *Activity diagram* hapus data *user*

Pada gambar 4. 4 menjelaskan prosedur yang digunakan *admin* untuk menghapus data. Halaman *admin* ditampilkan oleh sistem ketika telah dibuka oleh *admin*. *Admin* kemudian memilih menu yang datanya akan dimusnahkan, sistem menampilkan halaman menu yang telah dipilih oleh *admin*, dan *admin* mengklik pilihan hapus data. Sistem kemudian akan menampilkan konfirmasi penghapusan. *Admin*

kemudian melakukan konfirmasi. Jika *admin* konfirmasi, data akan terhapus, jika tidak maka data batal dihapus.

b. *Activity diagram* petugas

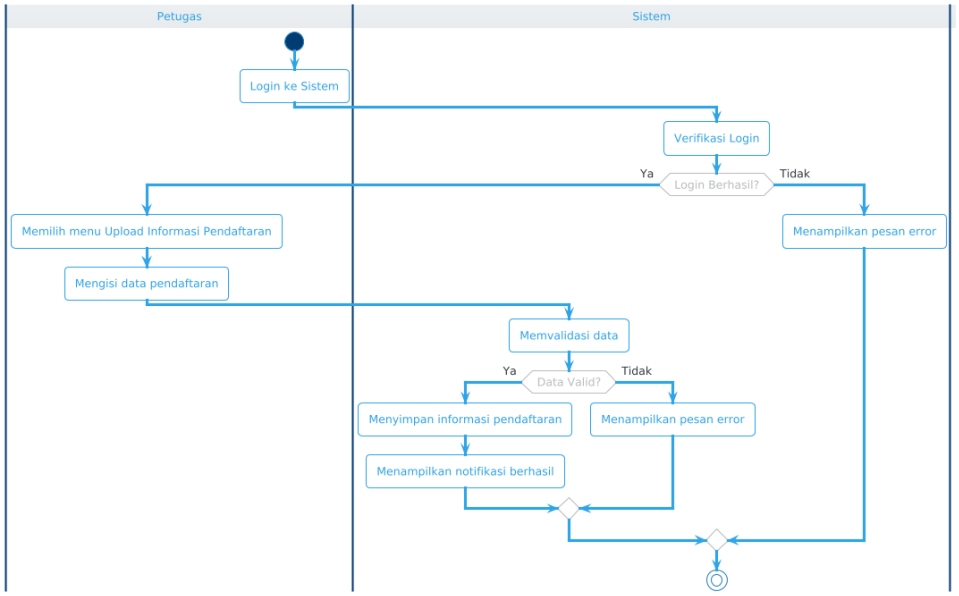
1) *Activity diagram upload* informasi peminjaman



Gambar 4. 5 *Activity diagram upload* informasi peminjaman

Pada gambar 4.5 ini menggambarkan alur proses unggah informasi pinjaman oleh petugas dalam sistem. Proses dimulai dengan petugas melakukan *login* ke sistem. Sistem akan memverifikasi *login*, jika valid maka petugas dapat melanjutkan ke tahap berikutnya, jika tidak, sistem akan menampilkan pesan error. Setelah *login* berhasil, petugas memilih menu unggah informasi pinjaman dan mengisi data pinjaman. Sistem kemudian memvalidasi data yang dimasukkan. Jika data valid, sistem akan menyimpannya dan menampilkan notifikasi bahwa proses berhasil. Namun, jika data tidak valid, sistem akan menampilkan pesan error dan meminta perbaikan data. Setelah itu, proses berakhir.

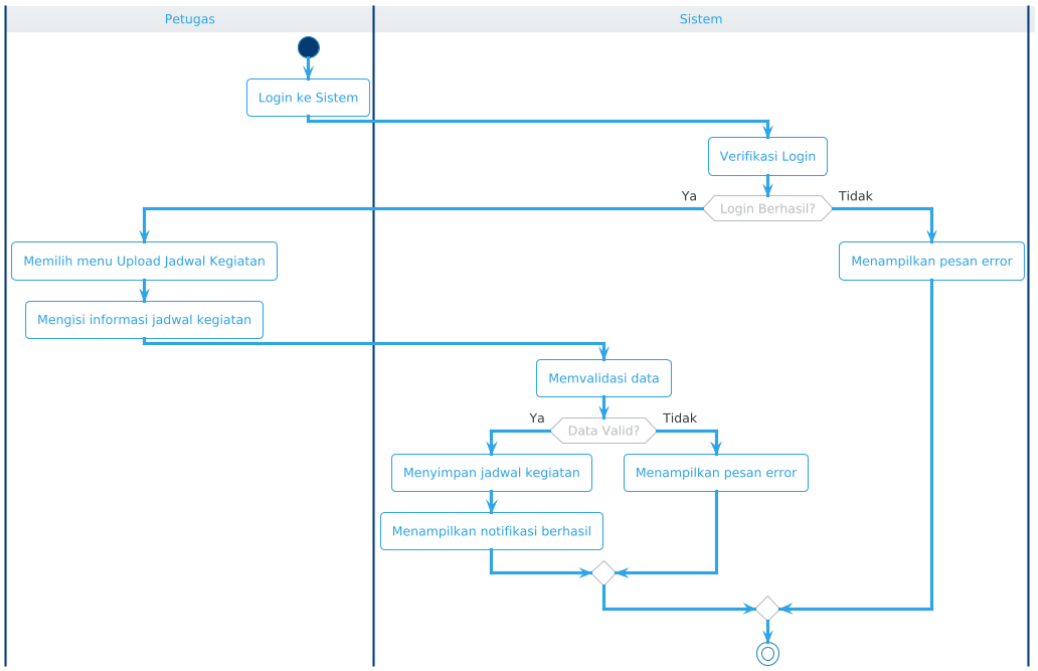
2) *Activity diagram upload* informasi pendaftaran



Gambar 4. 6 *Activity diagram* informasi pendaftaran

Pada Gambar 4.6 ini serupa dengan *diagram* sebelumnya, tetapi kali ini menggambarkan alur proses unggah informasi pendaftaran dalam sistem. Proses dimulai dengan petugas yang melakukan *login* ke sistem. Sistem akan memverifikasi *login*, jika berhasil, petugas dapat melanjutkan ke langkah berikutnya, sedangkan jika gagal, sistem akan menampilkan pesan error. Setelah *login* berhasil, petugas memilih menu unggah informasi pendaftaran dan mengisi data pendaftaran. Sistem kemudian memvalidasi data yang dimasukkan. Jika data valid, sistem akan menyimpan informasi pendaftaran dan menampilkan notifikasi bahwa proses berhasil. Jika data tidak valid, sistem akan menampilkan pesan error. Setelah itu, proses berakhir.

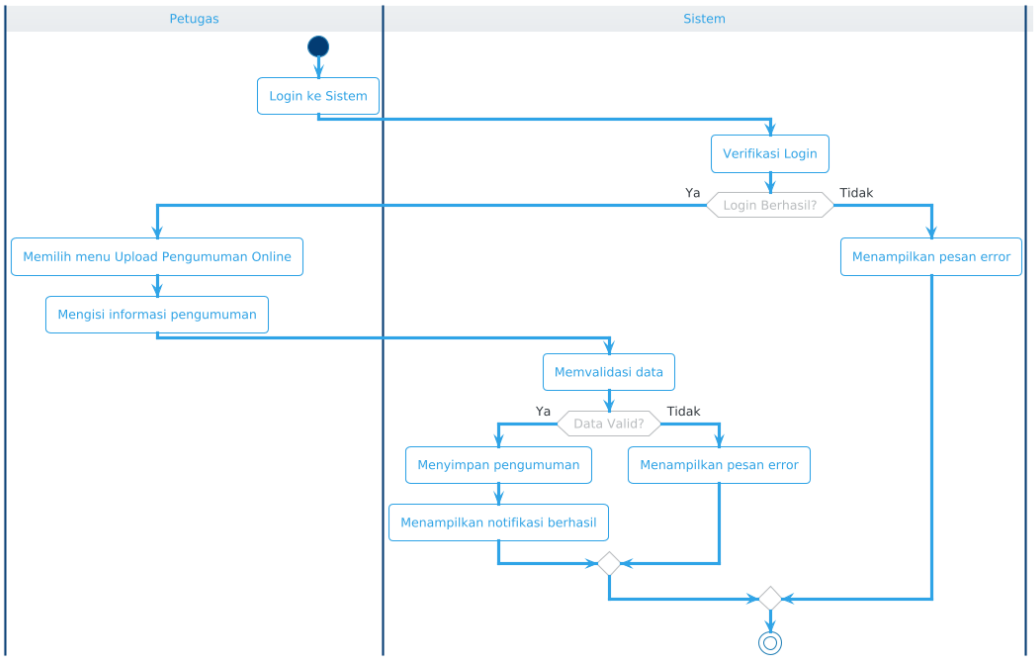
3) *Activity diagram upload* jadwal kegiatan



Gambar 4. 7 *Activity diagram upload* jadwal kegiatan

Pada gambar 4.7 ini menggambarkan alur proses unggah jadwal kegiatan oleh petugas dalam sistem. Proses dimulai dengan petugas melakukan *login* ke sistem. Sistem kemudian melakukan verifikasi *login*. Jika *login* berhasil, petugas dapat melanjutkan ke tahap berikutnya, tetapi jika gagal, sistem akan menampilkan pesan error. Setelah *login* berhasil, petugas memilih menu unggah jadwal kegiatan dan mengisi informasi jadwal kegiatan. Sistem kemudian memvalidasi data yang dimasukkan. Jika data valid, sistem akan menyimpan jadwal kegiatan dan menampilkan notifikasi bahwa proses berhasil. Namun, jika data tidak valid, sistem akan menampilkan pesan error. Setelah itu, proses berakhir.

4) *Activity diagram upload* pengumuman *online*

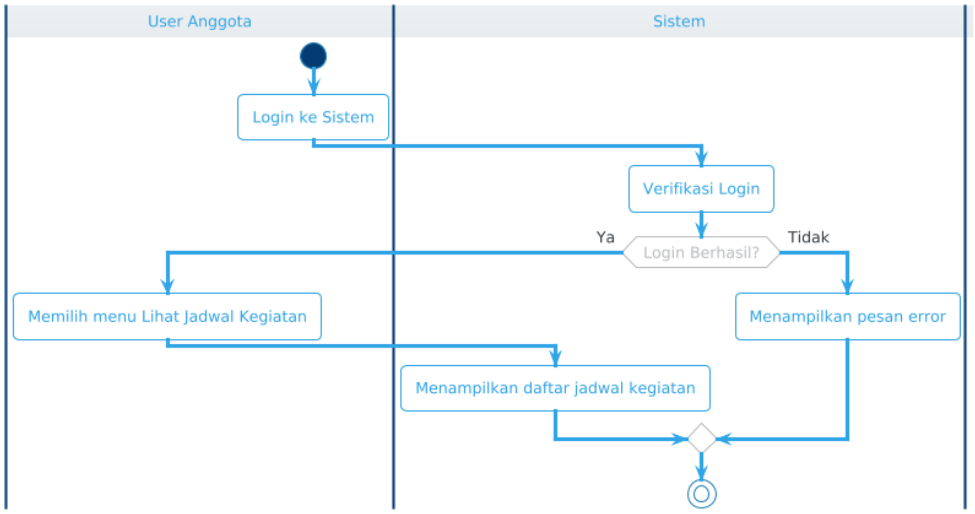


Gambar 4. 8 *Activity diagram upload* pengumuman *online*

Pada gambar 4.8 ini menunjukkan alur proses unggah pengumuman *online* oleh petugas dalam sistem. Proses dimulai dengan petugas melakukan *login* ke sistem, di mana sistem akan memverifikasi *login*. Jika *login* berhasil, petugas dapat melanjutkan ke langkah berikutnya, sedangkan jika gagal, sistem akan menampilkan pesan error. Setelah berhasil *login*, petugas memilih menu unggah pengumuman *online* dan mengisi informasi pengumuman. Sistem kemudian melakukan validasi data. Jika data yang dimasukkan valid, sistem akan menyimpan pengumuman dan menampilkan notifikasi bahwa proses berhasil. Namun, jika data tidak valid, sistem akan menampilkan pesan error. Setelah itu, proses berakhir.

c. *Activity diagram* Anggota

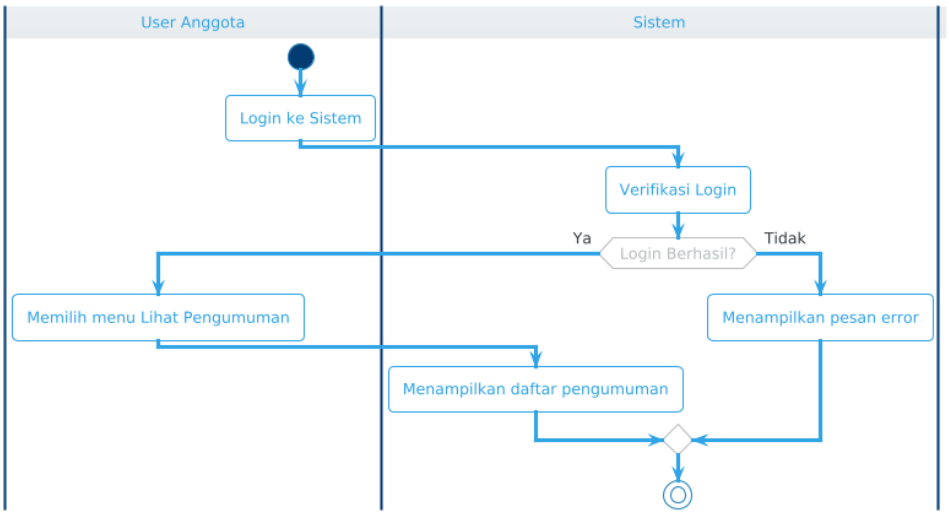
1) *Activity diagram* lihat jadwal kegiatan



Gambar 4. 9 *Activity diagram* lihat jadwal kegiatan

Pada gambar 4.9 aktivitas ini menggambarkan alur proses pengguna anggota dalam melihat jadwal kegiatan di sistem. Proses dimulai dengan *user* anggota melakukan *login* ke sistem, yang kemudian diverifikasi oleh sistem. Jika *login* berhasil, *user* dapat melanjutkan ke tahap berikutnya, sedangkan jika gagal, sistem akan menampilkan pesan error. Setelah berhasil *login*, *user* memilih menu "Lihat Jadwal Kegiatan", dan sistem akan menampilkan daftar jadwal kegiatan yang tersedia. Setelah daftar jadwal ditampilkan, proses berakhir.

2) *Activity diagram* lihat pengumuman

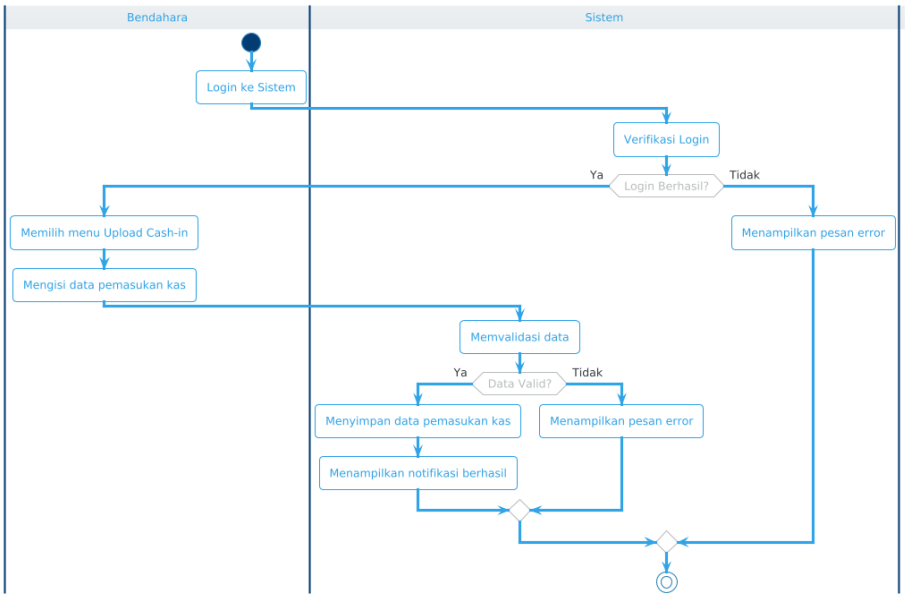


Gambar 4. 10 *Activity diagram* lihat pengumuman

Pada gambar 4.10 ini menunjukkan alur proses bagi *user* anggota dalam melihat pengumuman di sistem. Proses dimulai dengan *user* anggota melakukan *login* ke sistem. Sistem kemudian melakukan verifikasi *login*. Jika *login* berhasil, *user* dapat melanjutkan ke langkah berikutnya, sedangkan jika gagal, sistem akan menampilkan pesan error. Setelah berhasil *login*, *user* memilih menu "Lihat Pengumuman", dan sistem akan menampilkan daftar pengumuman yang tersedia. Setelah daftar pengumuman ditampilkan, proses berakhir.

d. *Activity diagram* Bendahara

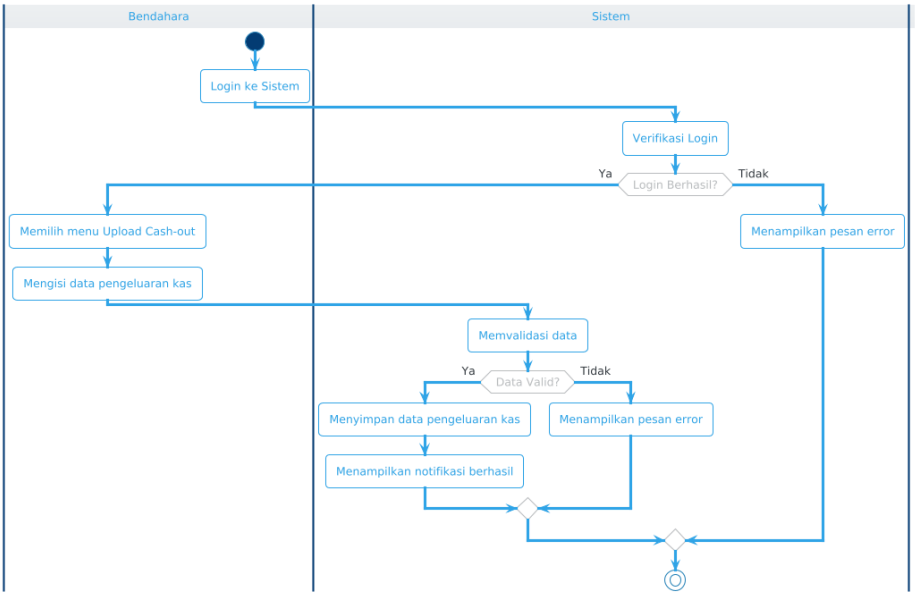
1) *Activity diagram upload cashin*



Gambar 4. 11 *Activity diagram upload cashin*

Pada gambar 4.11 ini menunjukkan alur proses yang dilakukan oleh bendahara dalam mengunggah data pemasukan kas (cash-in) ke dalam sistem. Proses dimulai dengan bendahara melakukan *login* ke sistem, kemudian sistem akan melakukan verifikasi *login*. Jika *login* berhasil, pengguna dapat melanjutkan ke langkah berikutnya, sedangkan jika gagal, sistem akan menampilkan pesan error. Setelah berhasil *login*, bendahara memilih menu "*Upload* Cash-In" dan menginput data pemasukan kas. Sistem kemudian memvalidasi data yang diinput. Jika data valid, sistem akan menyimpan informasi pemasukan kas dan menampilkan notifikasi berhasil. Namun, jika data tidak valid, sistem akan menampilkan pesan error. Setelah langkah ini selesai, proses berakhir.

2) *Activity diagram upload cashout*



Gambar 4. 12 *Activity diagram upload cashout*

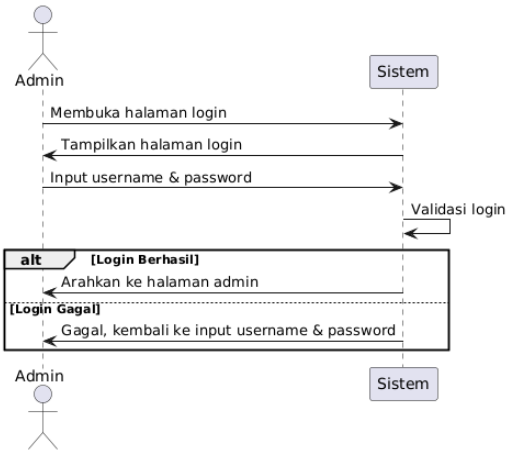
Pada gambar 4.12 ini menunjukkan proses bendahara dalam mengunggah data pengeluaran kas (cash-out) ke dalam sistem. Alur prosesnya mirip dengan unggahan pemasukan kas, namun fokus pada pengeluaran. Setelah bendahara *login* dan sistem melakukan verifikasi, jika *login* berhasil, pengguna dapat melanjutkan ke langkah berikutnya. Bendahara memilih menu "*Upload* Cash-Out" dan menginput data pengeluaran kas. Sistem kemudian melakukan validasi data. Jika data valid, sistem akan menyimpan informasi pengeluaran kas dan menampilkan notifikasi berhasil. Jika tidak valid, sistem akan menampilkan pesan error. Setelah langkah ini selesai, proses berakhir.

2. *Sequence diagram*

Sequence diagram merupakan *diagram* yang menggambarkan interaksi antara objek-objek dalam sistem secara berurutan berdasarkan waktu. *Diagram* ini menunjukkan bagaimana objek-objek saling bertukar pesan atau data untuk menyelesaikan suatu tugas atau *usecase*.

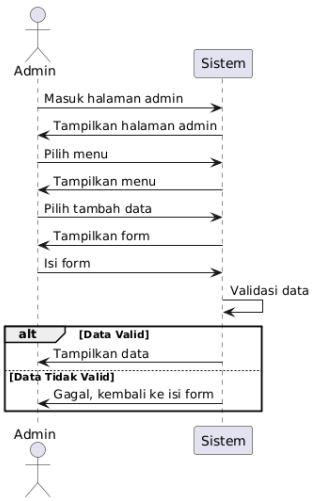
a. *Sequence diagram admin*

1) *Sequence diagram login*



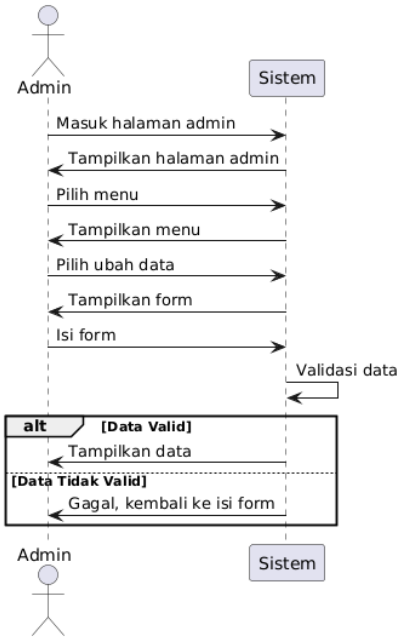
Gambar 4. 13 *Sequence diagram login*

2) *Sequence diagram* tambah *user*



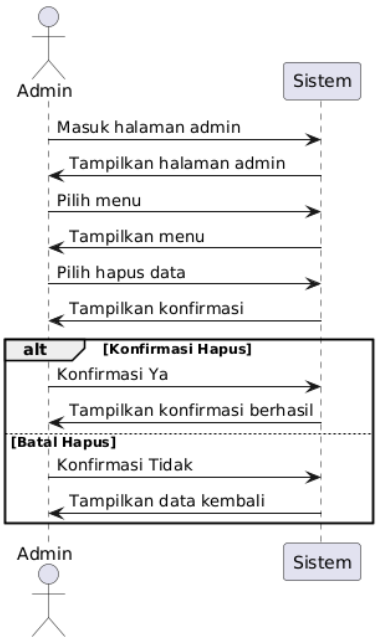
Gambar 4. 14 *Sequence diagram* tambah *user*

3) *Sequence diagram* ubah data *user*



Gambar 4. 15 *Sequence diagram* ubah data *user*

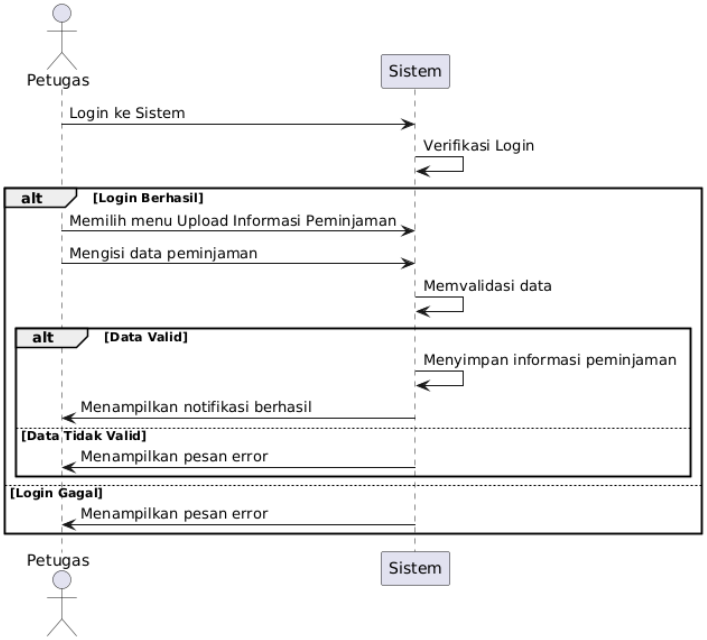
4) *Sequence diagram* hapus data *user*



Gambar 4. 16 *Sequence diagram* hapus data *user*

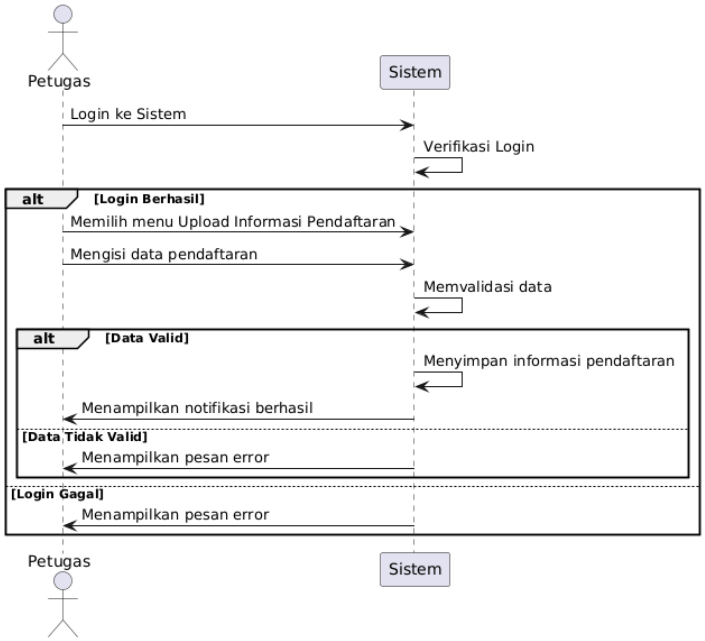
b. *Sequence diagram* Petugas

1) *Sequence diagram upload* informasi peminjaman



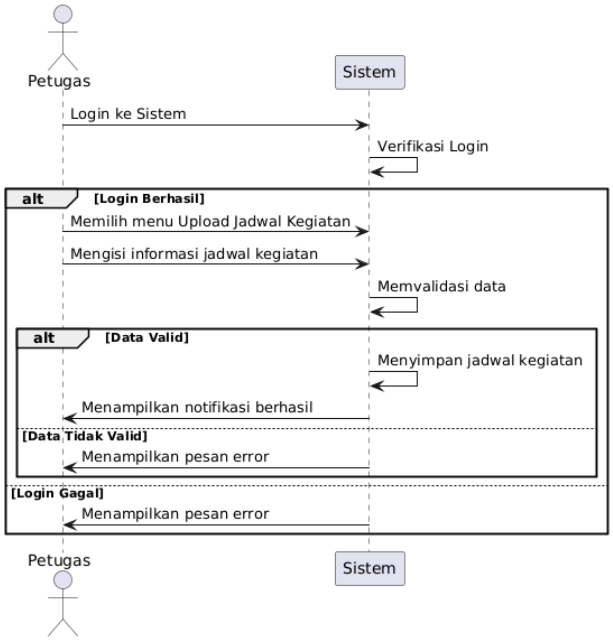
Gambar 4. 17 *Sequence diagram upload* informasi peminjaman

2) *Sequence diagram upload* informasi pendaftaran



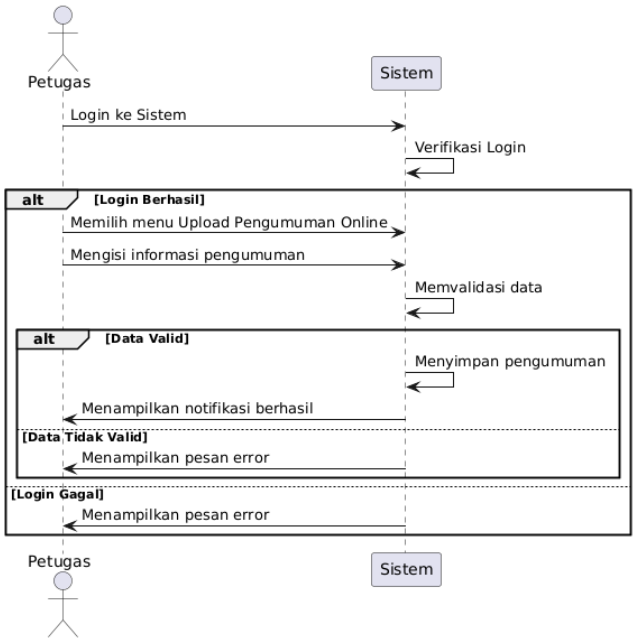
Gambar 4. 18 *Sequence diagram upload* informasi pendaftaran

3) *Sequence diagram upload* jadwal kegiatan



Gambar 4. 19 *Sequence diagram upload* jadwal kegiatan

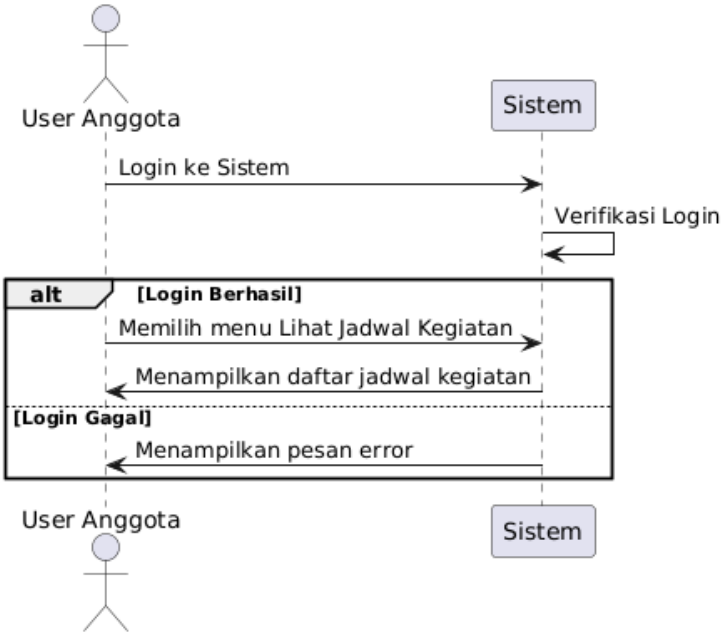
4) *Sequence diagram upload* pengumuman *online*



Gambar 4. 20 *Sequence diagram upload* pengumuman *online*

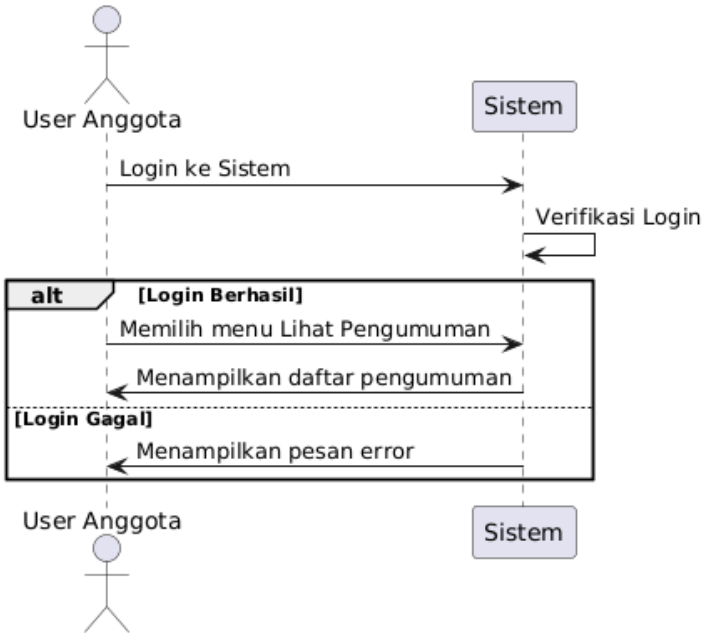
c. *Sequence diagram* Anggota

1) *Sequence diagram* lihat jadwal kegiatan



Gambar 4. 21 *Sequence diagram* lihat jadwal kegiatan

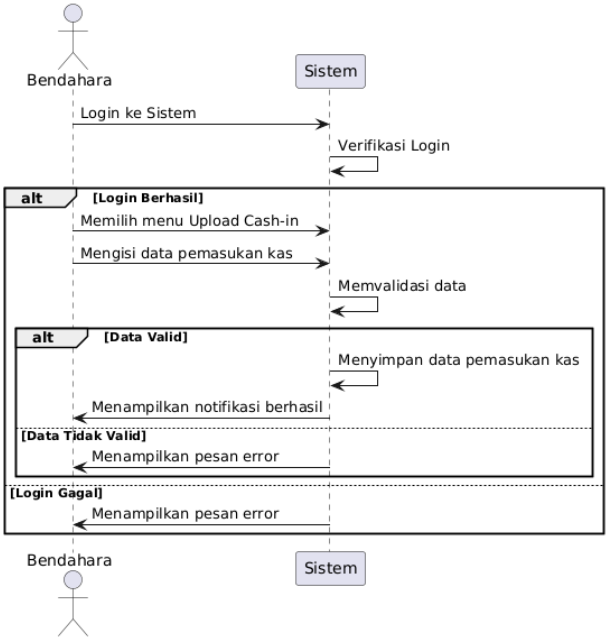
2) *Sequence diagram* lihat pengumuman



Gambar 4. 22 *Sequence diagram* lihat pengumuman

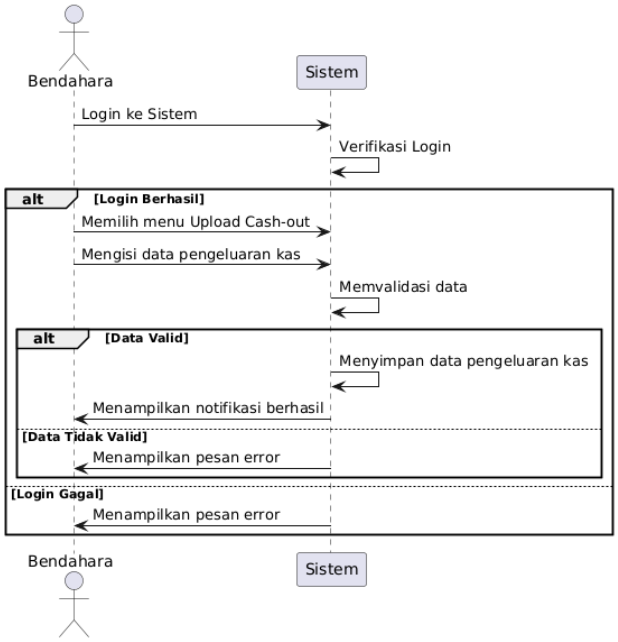
d. *Sequence diagram* Bendahara

1) *Sequence diagram upload cashin*



Gambar 4. 23 *Sequence diagram upload cashin*

2) *Sequence diagram upload cashout*



Gambar 4. 24 *Sequence diagram upload cashout*

B. Detail Sistem

1. Halaman beranda

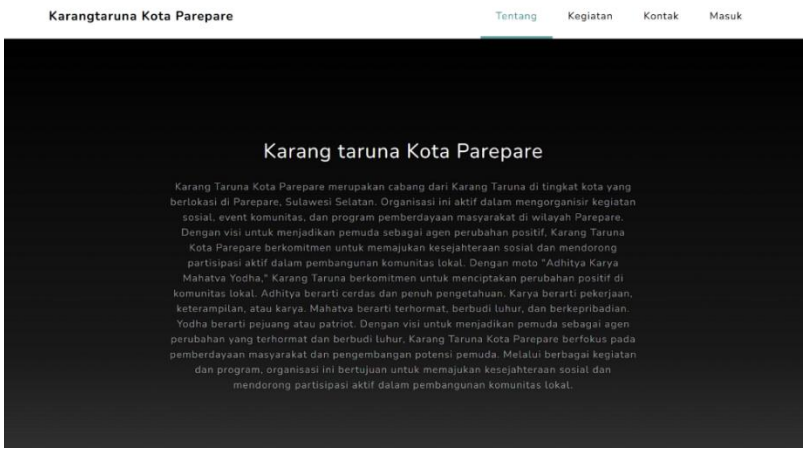
Halaman beranda/*home* merupakan tampilan utama yang memberikan gambaran umum tentang sistem, termasuk navigasi utama dan ringkasan informasi untuk memudahkan pengguna dalam memahami dan mengakses berbagai fungsi sistem dengan efisien.



Gambar 4. 25 Halaman beranda

2. Halaman *login*

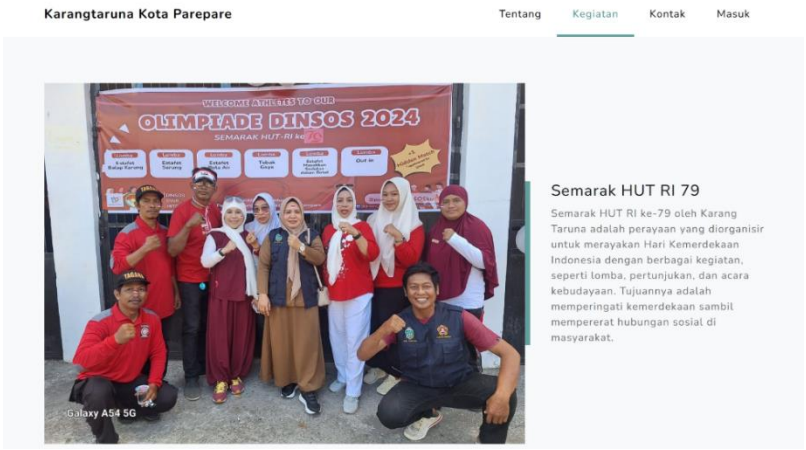
Halaman tentang/*login* berisi informasi tentang profil, visi, misi, sejarah, dan tujuan dari sistem atau organisasi yang mengelolanya.



Gambar 4. 26 Halaman tentang

3. Halaman kegiatan

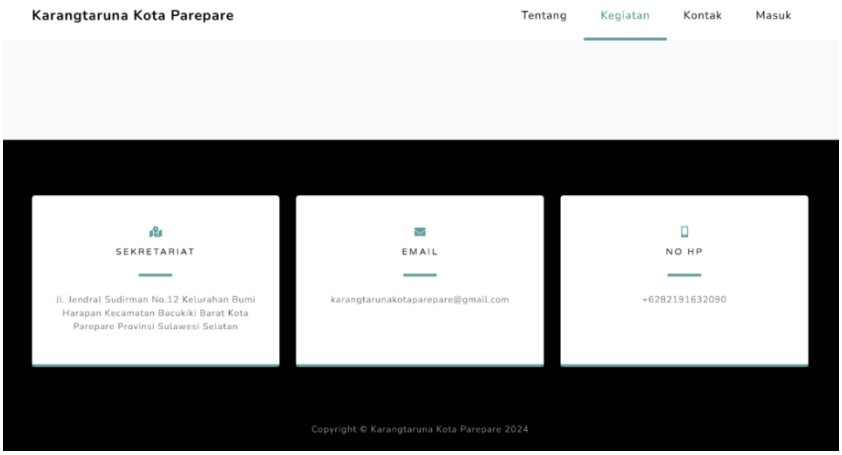
Halaman Kegiatan Karang Taruna menampilkan berbagai aktivitas yang telah atau akan dilaksanakan oleh organisasi. dilengkapi dengan deskripsi singkat, tanggal pelaksanaan, dokumentasi berupa foto atau video, serta dampak positif yang dihasilkan.



Gambar 4. 27 Halaman kegiatan

4. Halaman kontak

Halaman Kontak menyediakan informasi untuk menghubungi organisasi, seperti alamat, nomor telepon, email, dan media sosial.



Gambar 4. 28 Halaman kontak

5. Halaman *login*

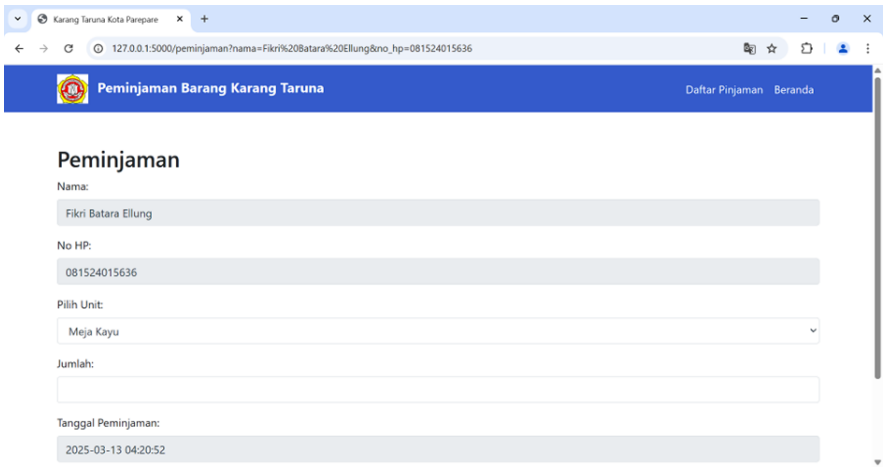
Halaman *Login* berfungsi sebagai pintu masuk bagi pengguna terdaftar untuk mengakses fitur khusus. Halaman ini biasanya berisi kolom untuk memasukkan *user*name/email dan kata sandi, serta tombol Masuk.



Gambar 4. 29 Halaman *login*

6. Halaman peminjaman barang karang taruna

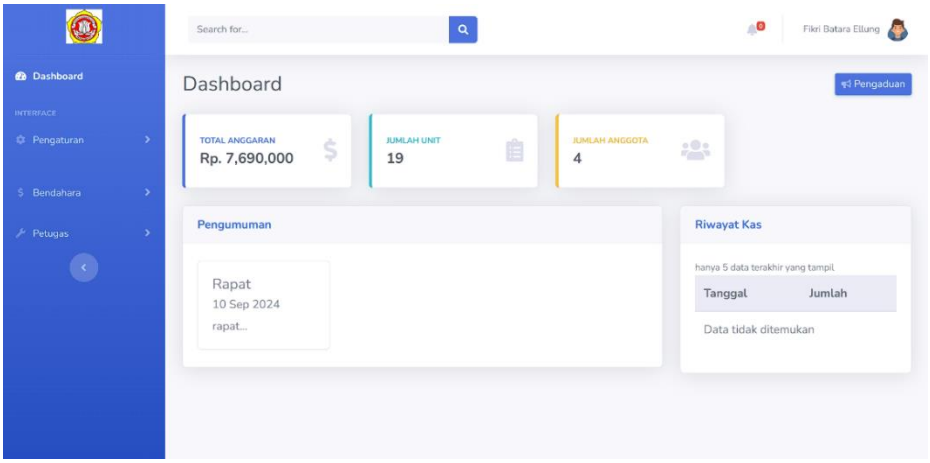
Halaman Peminjaman Barang Karang Taruna dirancang untuk memudahkan anggota dalam mengajukan peminjaman barang. Pada halaman ini, pengguna diminta mengisi beberapa informasi penting.



Gambar 4. 30 Halaman peminjaman barang karang taruna

7. Halaman *dashboard (admin)*

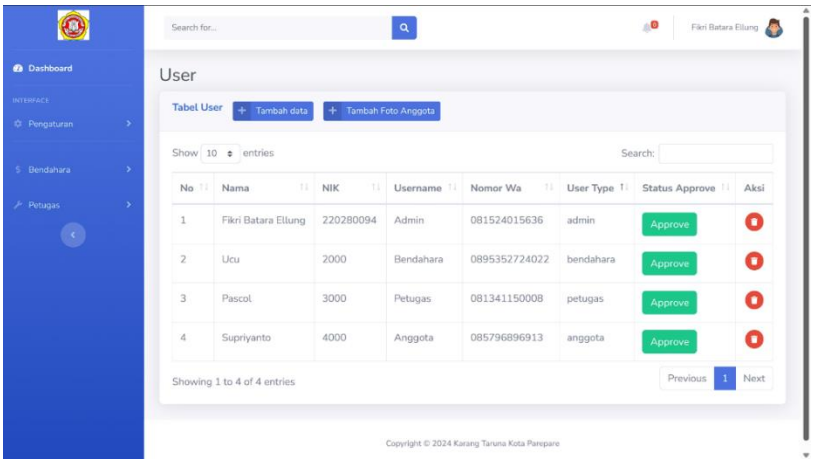
Halaman *Dashboard Admin* merupakan pusat kontrol bagi *admin* untuk mengelola seluruh aspek sistem. Biasanya, halaman ini menampilkan ringkasan data penting.



Gambar 4. 31 Halaman *dashboard admin*

8. Halaman pengaturan *user (admin)*

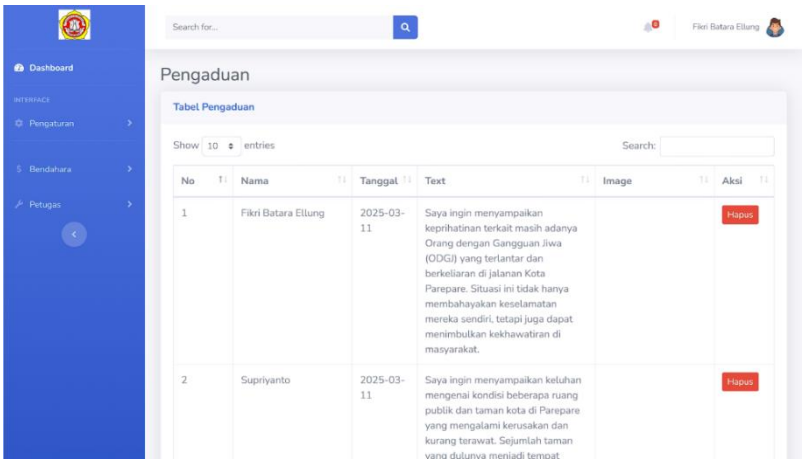
Halaman Pengaturan *User* (*Admin*) memungkinkan *admin* untuk mengelola data pengguna. Terdapat tombol *Approve* untuk menyetujui pengguna, dan tombol Hapus (ikon merah) untuk menghapus pengguna.



Gambar 4. 32 Halaman pengaturan *user*

9. Halaman pengaturan pengaduan (*admin*)

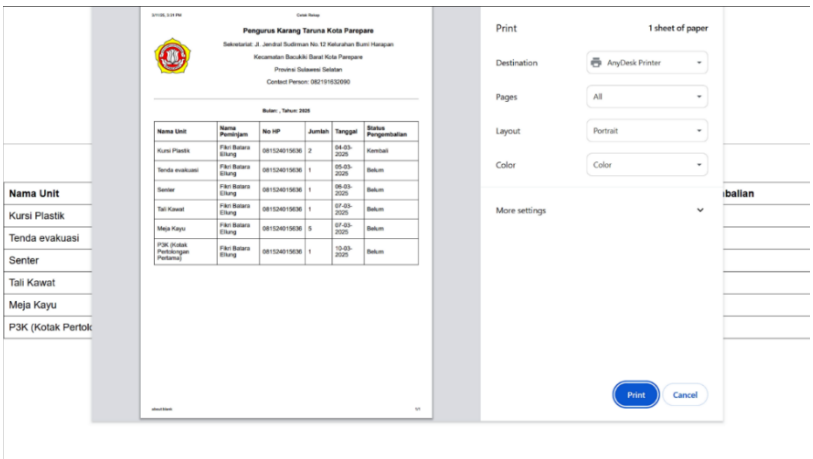
Halaman Pengaturan Pengaduan berfungsi untuk mengelola laporan atau keluhan dari masyarakat. *Admin* dapat melihat daftar pengaduan.



Gambar 4. 33 Halaman pengaturan pengaduan

10. Halaman laporan peminjaman (*admin*)

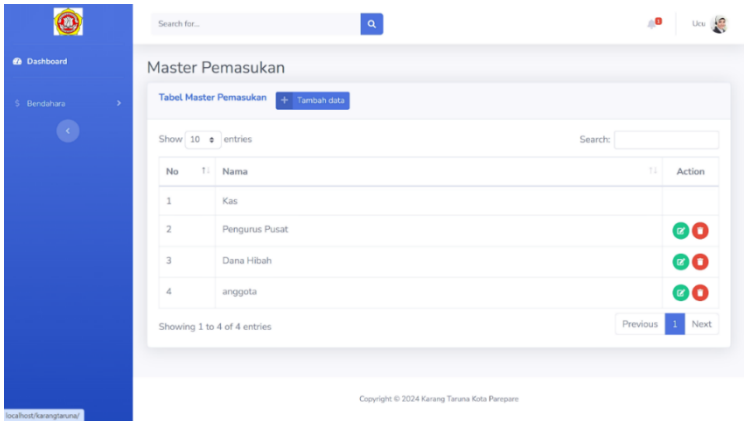
Halaman Laporan Peminjaman menampilkan dan mencetak laporan peminjaman barang yang dikelola organisasi. Halaman ini biasanya berisi daftar peminjaman dengan informasi yang bisa dilihat seperti dibawah ini.



Gambar 4. 34 Halaman laporan peminjaman

11. Halaman *master* pemasukan (bendahara)

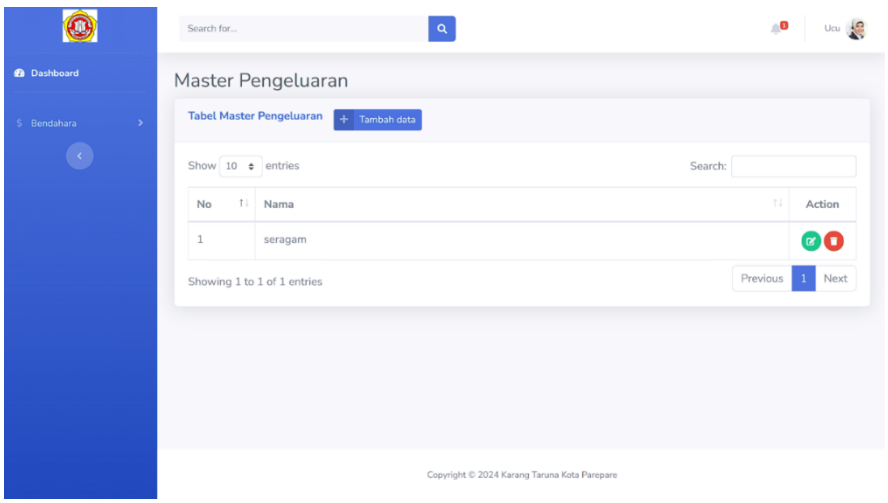
Halaman *Master* Pemasukan (Bendahara) mengelola sumber pemasukan organisasi. Bendahara dapat melihat daftar pemasukan yang mencakup nama-nama sumber dana, seperti Kas, Pengurus Pusat, Dana Hibah, dan Anggota.



Gambar 4. 35 Halaman *master* pemasukan

12. Halaman *master* pengeluaran (bendahara)

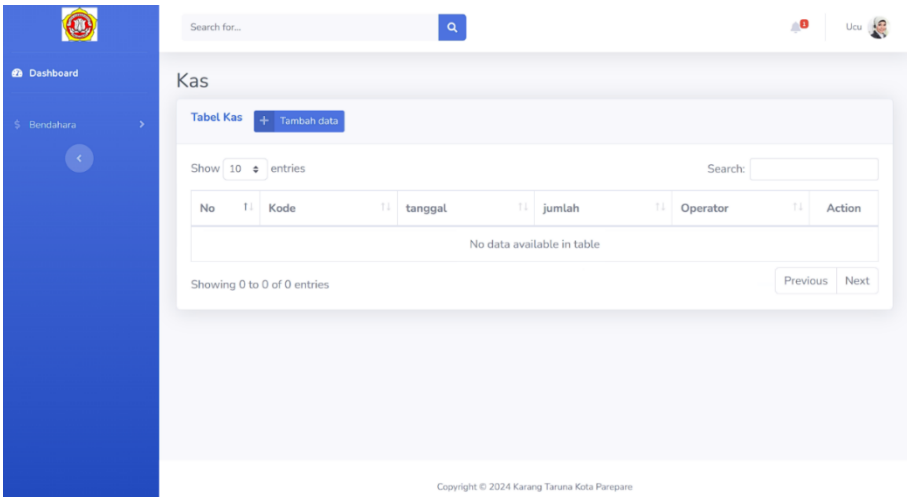
Halaman *Master* Pengeluaran digunakan untuk mengelola data pengeluaran organisasi.



Gambar 4. 36 Halaman *master* pengeluaran

13. Halaman transaksi kas (bendahara)

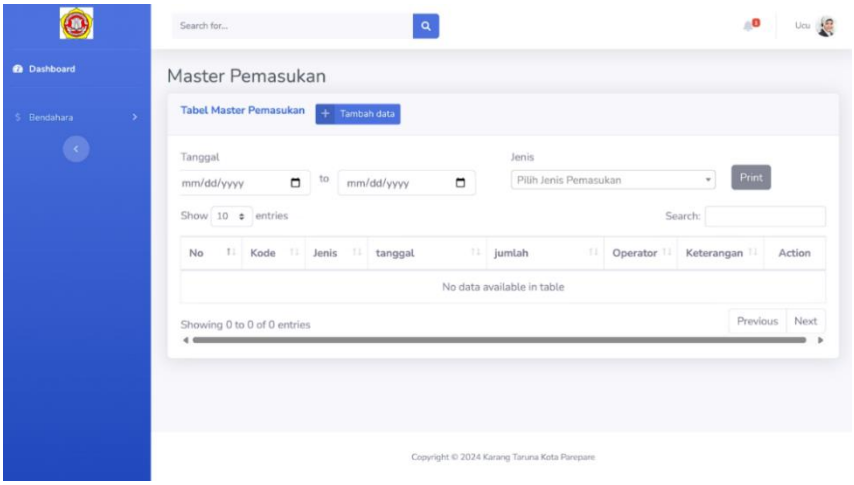
Halaman Transaksi Kas digunakan untuk mencatat dan memantau arus kas organisasi, baik pemasukan maupun pengeluaran.



Gambar 4. 37 Halaman transaksi kas

14. Halaman transaksi pemasukan (bendahara)

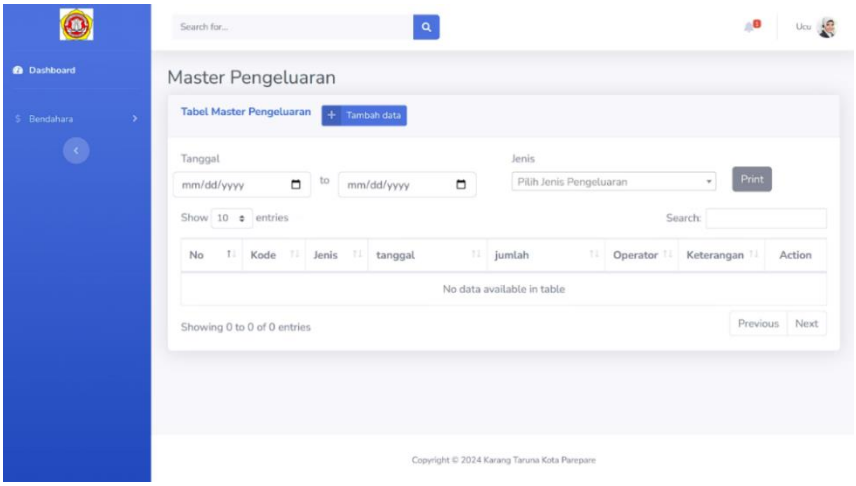
Pada halaman Transaksi Pemasukan, bendahara dapat menambahkan pemasukan dengan mengisi informasi seperti tanggal pemasukan, sumber dana, jumlah, dan keterangan.



Gambar 4. 38 Halaman transaksi pemasukan

15. Halaman transaksi pengeluaran (bendahara)

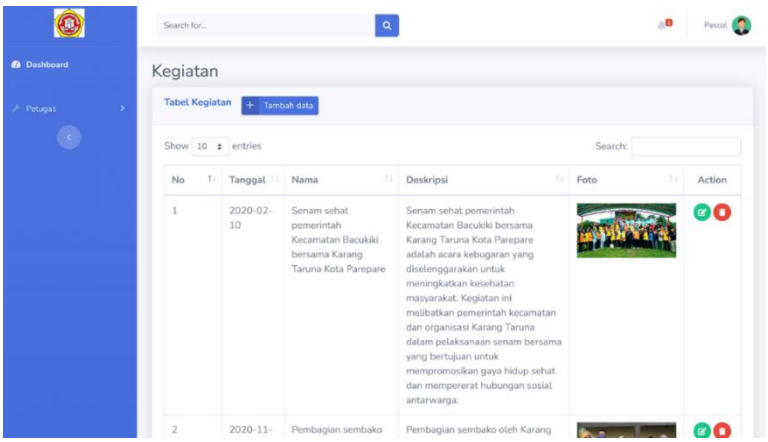
Halaman Transaksi Pengeluaran digunakan untuk mencatat pengeluaran organisasi. Bendahara mengisi tanggal pengeluaran, jenis pengeluaran, jumlah, dan keterangan.



Gambar 4. 39 Halaman transaksi pengeluaran

16. Halaman kegiatan (petugas)

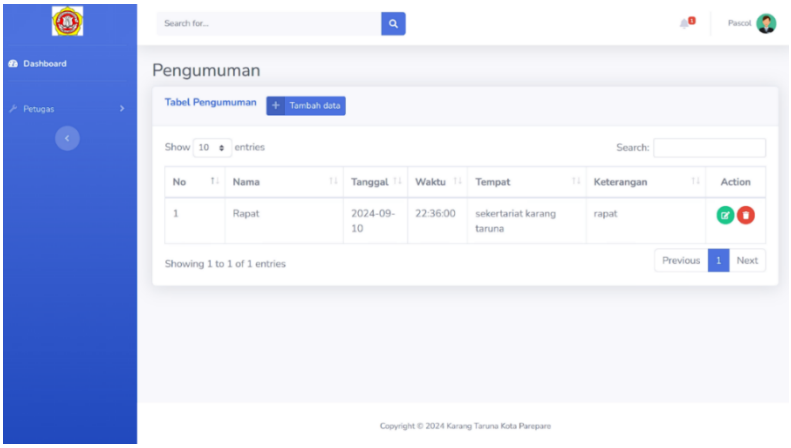
Halaman *dashboard* kegiatan mengelola berbagai aktivitas yang dilakukan oleh organisasi. *Admin* atau petugas dapat menambahkan kegiatan baru dengan menekan tombol Tambah Data.



Gambar 4. 40 Halaman *dashboard* kegiatan

17. Halaman pengumuman (petugas)

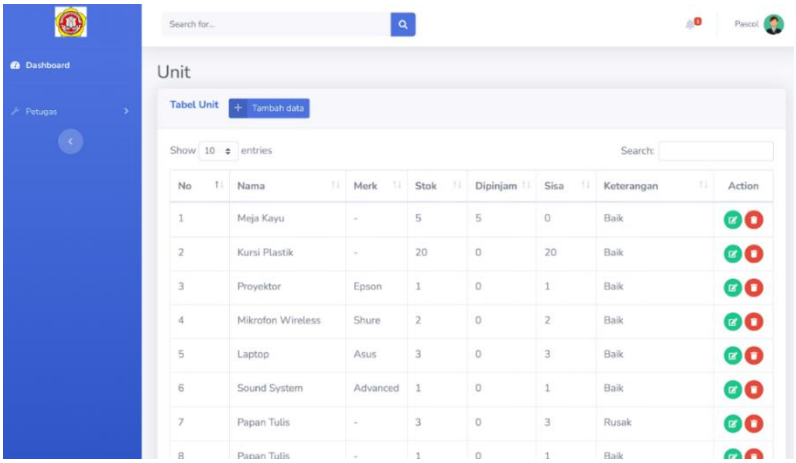
Halaman *dashboard* pengumuman berfungsi untuk mengelola informasi penting yang perlu disampaikan kepada anggota atau masyarakat. Dalam halaman ini, terdapat tabel yang memuat data.



Gambar 4. 41 Halaman pengumuman

18. Halaman Unit (petugas)

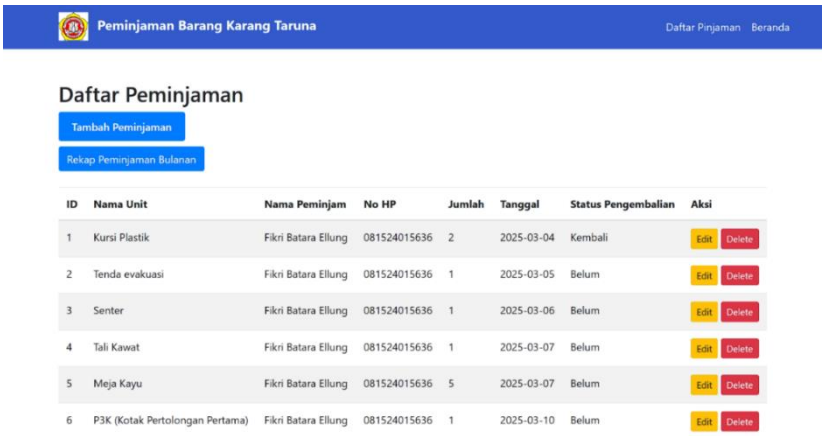
Halaman Unit untuk mengelola data inventaris atau barang yang dimiliki oleh organisasi. Dalam halaman ini, terdapat tabel yang menampilkan informasi penting.



Gambar 4. 42 Halaman *dashboard* unit

19. Halaman data pinjaman (petugas)

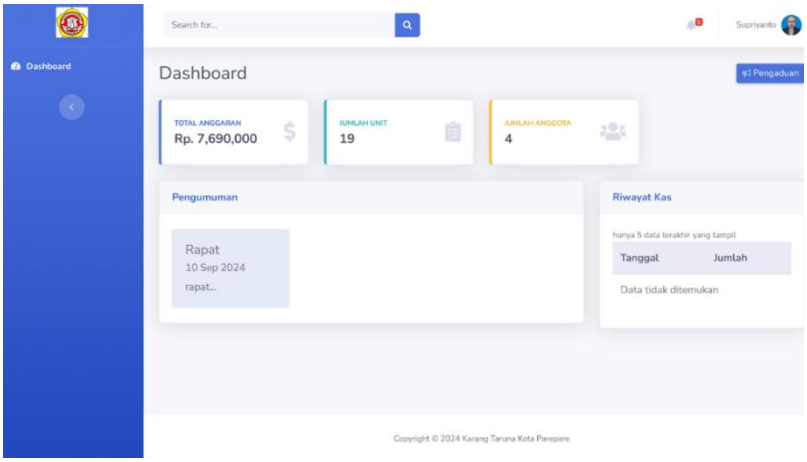
Halaman data pinjaman untuk mencatat dan memantau proses peminjaman barang. Dalam halaman ini terdapat tabel yang memuat informasi seperti ID peminjaman, nama unit yang dipinjam, nama peminjam, nomor HP, jumlah barang yang dipinjam, tanggal peminjaman, serta status pengembalian. Halaman ini memudahkan *admin* atau petugas dalam melacak status barang yang dipinjam dan memastikan proses pengelolaan inventaris berjalan dengan teratur.



Gambar 4. 43 Halaman data pinjaman

20. Halaman *dashboard* (anggota)

Halaman *Dashboard* (Anggota) pada sistem informasi Karang Taruna menyajikan ringkasan informasi penting yang dapat diakses oleh anggota. Di bagian atas, terdapat tiga kotak ringkasan yang menampilkan Total Anggaran, Jumlah Unit, dan Jumlah Anggota. Di bawahnya, terdapat dua bagian utama: yaitu pengumuman dan riwayat kas. Selain itu, terdapat tombol Pengaduan di pojok kanan atas, yang memungkinkan anggota mengirimkan laporan atau keluhan langsung melalui sistem.



Gambar 4. 44 Halaman *dashboard* anggota

C. Pengujian Sistem

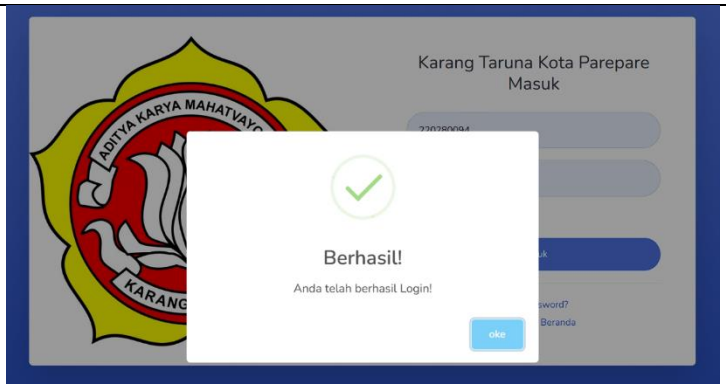
Pengujian sistem bertujuan untuk mengidentifikasi kesalahan, kesenjangan, atau kekurangan dalam sistem sebelum digunakan oleh pengguna akhir. Ada dua pendekatan utama dalam pengujian sistem ini, yaitu pengujian *Blackbox* dan pengujian *whitebox*.

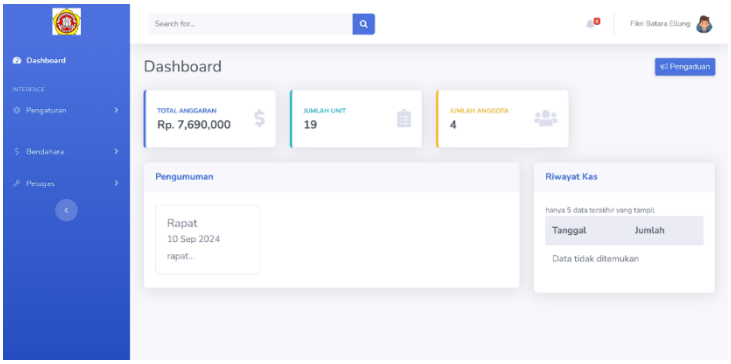
1. *Blackbox*

a. *Blackbox testing login* berhasil

Tabel 4. 1 *Blackbox testing login* berhasil

Tes faktor	Hasil	Keterangan
Memasukkan *email* atau *password* yang benar	✓	Sistem berhasil menampilkan halaman *admin/dashboard*.
Screenshot		

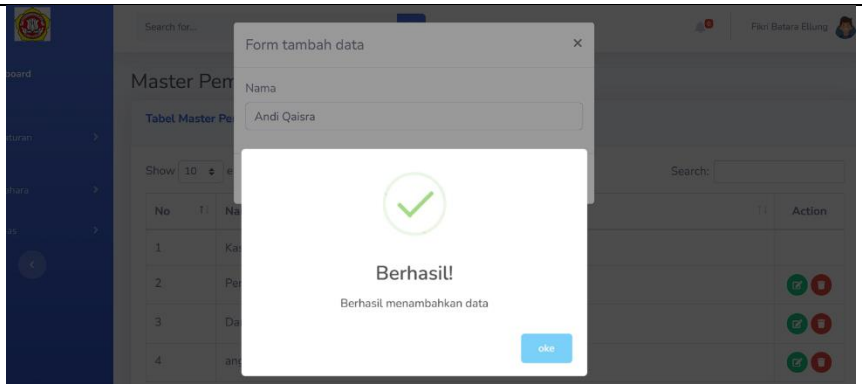




b. *Blackbox testing* tambah data

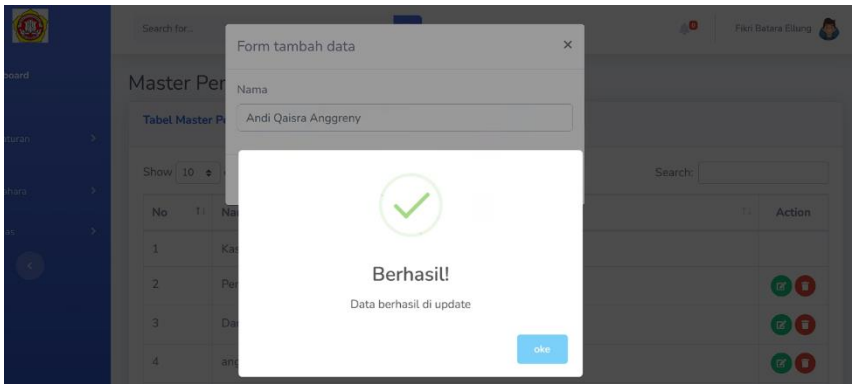
Tabel 4. 2 *Blackbox testing* tambah data

Tes faktor	Hasil	Keterangan
Admin mengisi *form* tambah dan menekan tombol simpan	✓	Sukses, sebuah notifikasi muncul bahwa data telah berhasil ditambahkan.
Screenshot		



c. *Blackbox testing* ubah data

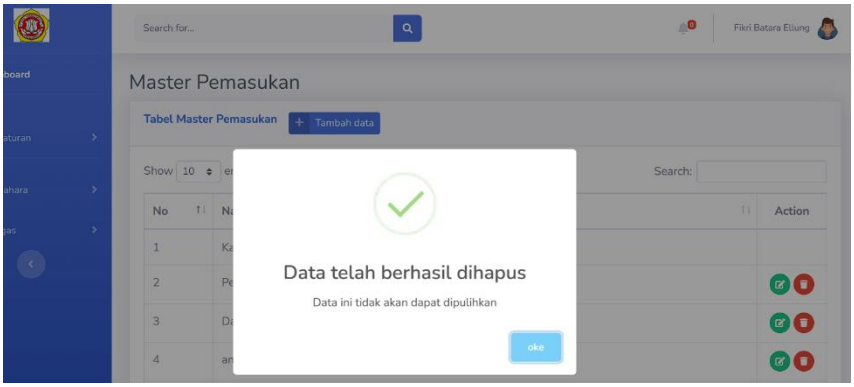
Tabel 4. 3 *Blackbox testing* ubah data

Tes faktor	Hasil	Keterangan
Admin mengisi *form* ubah data dan menekan tombol simpan	✓	Sukses, sebuah notifikasi muncul bahwa data telah berhasil diubah.
Screenshot		
		

d. *Blackbox testing* hapus data

Tabel 4. 4 *Blackbox testing* hapus data

Tes faktor	Hasil	Keterangan
Admin menghapus salah satu data	✓	Sukses, sebuah notifikasi muncul

		bahwa data telah berhasil dihapus.
Screenshot		
		

e. *Blackbox testing* halaman beranda

Tabel 4. 5 *Blackbox testing* halaman beranda

Tes faktor	Hasil	Keterangan
User pertama kali mengakses aplikasi	✓	Sukses, tampil halaman beranda
Screenshot		
		

f. *Blackbox testing* halaman *login*

Tabel 4. 6 *Blackbox testing* halaman *login*

Tes faktor	Hasil	Keterangan

User menekan tombol navigasi *login*	✓	Sukses, tampil halaman *login*
Screenshot		
		

g. *Blackbox testing* halaman kegiatan

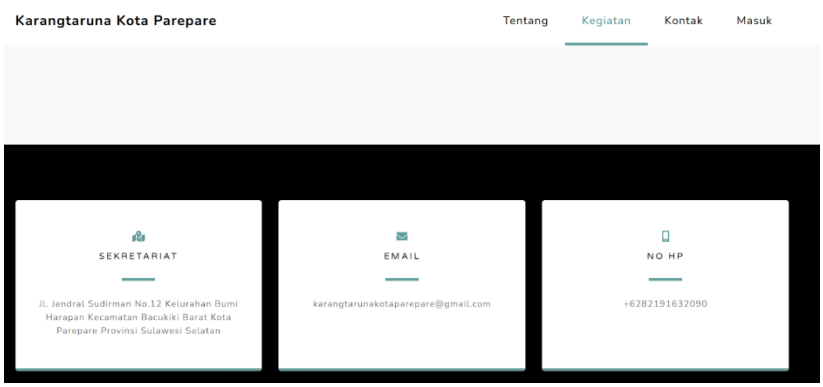
Tabel 4. 7 *Blackbox testing* halaman kegiatan

Tes faktor	Hasil	Keterangan
User menekan tombol navigasi kegiatan	✓	Sukses, tampil halaman kegiatan
Screenshot		
		

h. *Blackbox testing* halaman kontak

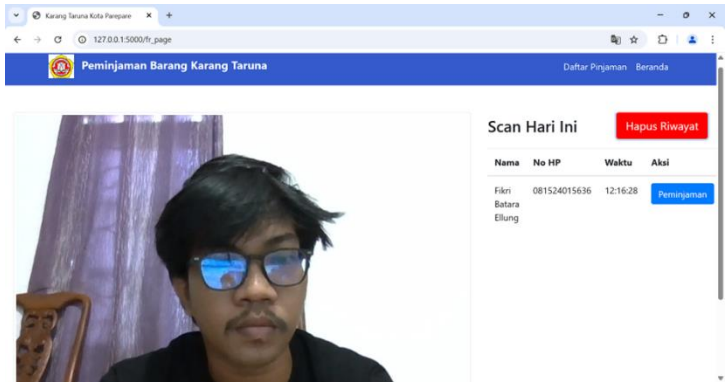
Tabel 4. 8 *Blackbox testing* halaman kontak

Tes faktor	Hasil	Keterangan

User menekan tombol navigasi kontak	✓	Sukses, tampil halaman kontak
Screenshot		
		

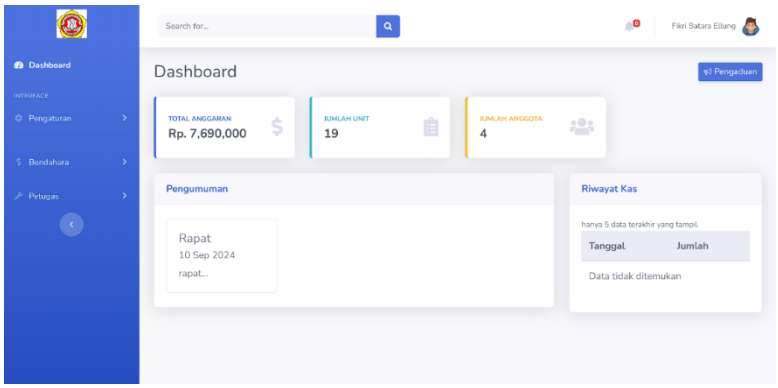
i. *Blackbox testing* peminjaman barang karang taruna

Tabel 4. 9 *Blackbox testing* peminjaman barang karang taruna

Tes faktor	Hasil	Keterangan
User melakukan pengaduan peminjaman barang di *website*	✓	Sukses, hasil scan wajah hari ini untuk melakukan peminjaman
Screenshot		
		

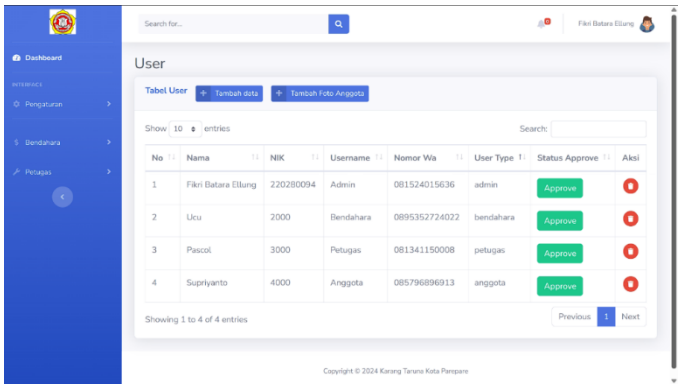
j. *Blackbox testing* halaman *dashboard admin*

Tabel 4. 10 *Blackbox testing* halaman *dashboard admin*

Tes faktor	Hasil	Keterangan
Admin berhasil masuk menggunakan akun *admini*strator	✓	Sukses, tampil halaman *dashboard admin*
Screenshot		
		

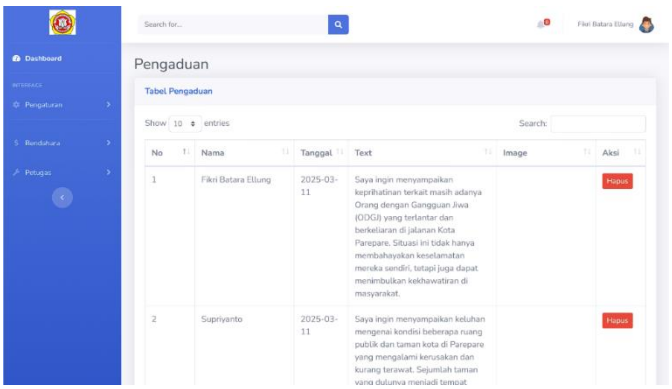
k. *Blackbox testing* halaman pengaturan *user*

Tabel 4. 11 *Blackbox testing* halaman pengaturan *user*

Tes faktor	Hasil	Keterangan
Admin menekan tombol navigasi pengaturan dan memilih pilihan pengaturan *user*	✓	Sukses, tampil halaman pengaturan *user*
Screenshot		
		

l. *Blackbox testing* halaman pengaturan pengaduan

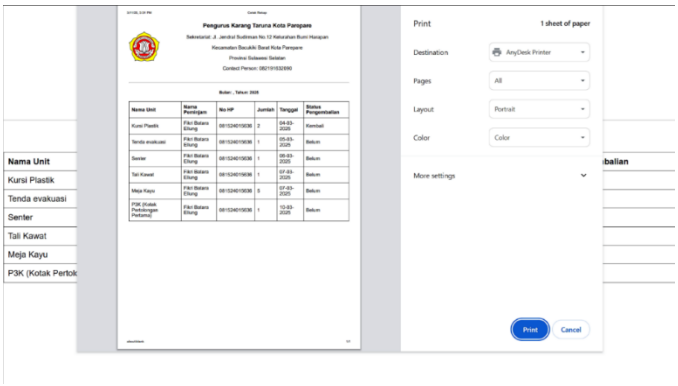
Tabel 4. 12 *Blackbox testing* halaman laporan peminjaman

Tes faktor	Hasil	Keterangan
Admin menekan tombol navigasi pengaturan dan memilih pilihan pengaduan	✓	Sukses, tampil halaman pengaturan pengaduan
Screenshot		
		

m. *Blackbox testing* halaman laporan peminjaman

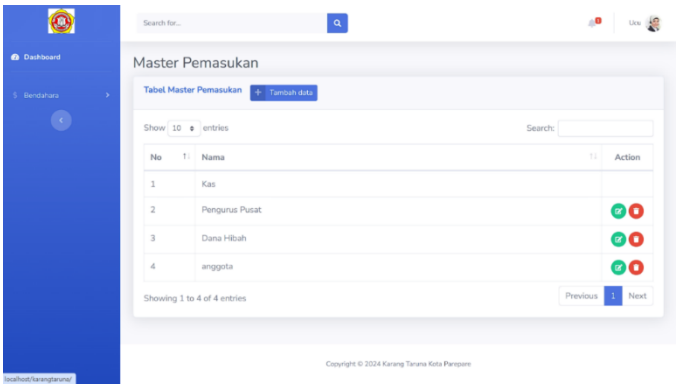
Tabel 4. 13 *Blackbox testing* halaman laporan peminjaman

Tes faktor	Hasil	Keterangan
Admin menekan tombol navigasi petugas dan memilih pilihan laporan peminjaman	✓	Sukses, tampil halaman laporan peminjaman
Screenshot		



n. *Blackbox testing master* pemasukan

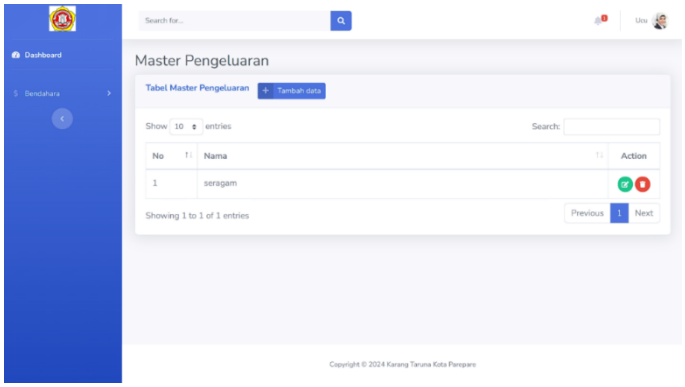
Tabel 4. 14 *Blackbox testing* halaman pemasukan

Tes faktor	Hasil	Keterangan
Bendahara menekan tombol navigasi bendahara dan memilih pilihan pemasukan	✓	Sukses, tampil halaman pemasukan
Screenshot		
		

o. *Blackbox testing master* pengeluaran

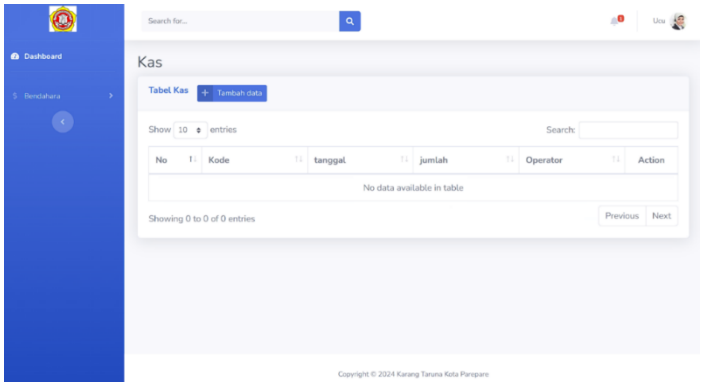
Tabel 4. 15 *Blackbox testing* halaman pengeluaran

Tes faktor	Hasil	Keterangan
Bendahara menekan tombol navigasi	✓	Sukses, tampil halaman pemasukan

bendahara dan memilih pilihan pengeluaran		
Screenshot		
		

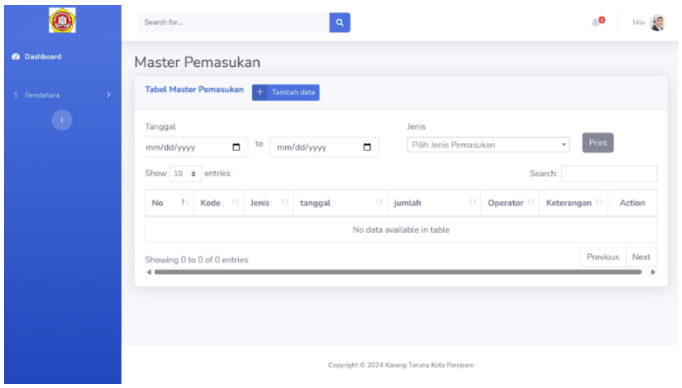
p. *Blackbox testing transaksi* kas

Tabel 4. 16 *Blackbox testing* halaman kas

Tes faktor	Hasil	Keterangan
Bendahara menekan tombol navigasi bendahara dan memilih pilihan kas	✓	Sukses, tampil halaman kas
Screenshot		
		

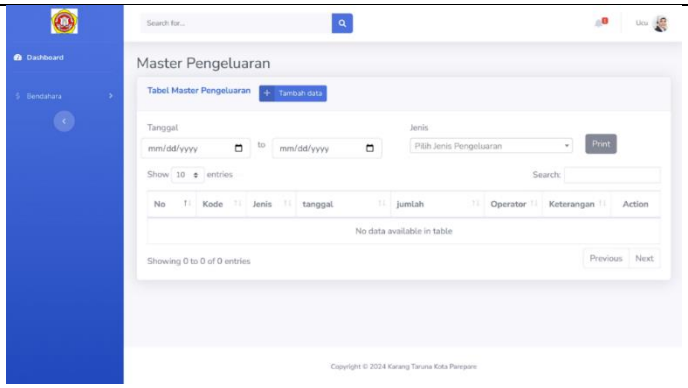
q. *Blackbox testing transaksi* pemasukan

Tabel 4. 17 *Blackbox testing* halaman pemasukan

Tes faktor	Hasil	Keterangan
Bendahara menekan tombol navigasi bendahara dan memilih pilihan pemasukan	✓	Sukses, tampil halaman pemasukan
Screenshot		
		

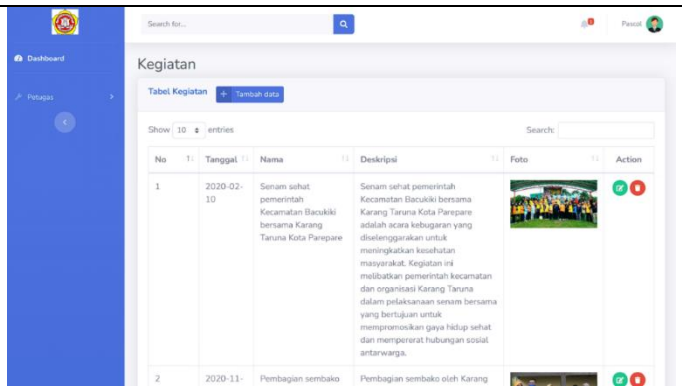
r. *Blackbox testing* transaksi pengeluaran

Tabel 4. 18 *Blackbox testing* halaman pengeluaran

Tes faktor	Hasil	Keterangan
Bendahara menekan tombol navigasi bendahara dan memilih pilihan pengeluaran	✓	Sukses, tampil halaman pengeluaran
Screenshot		
		

s. *Blackbox testing dashboard* kegiatan

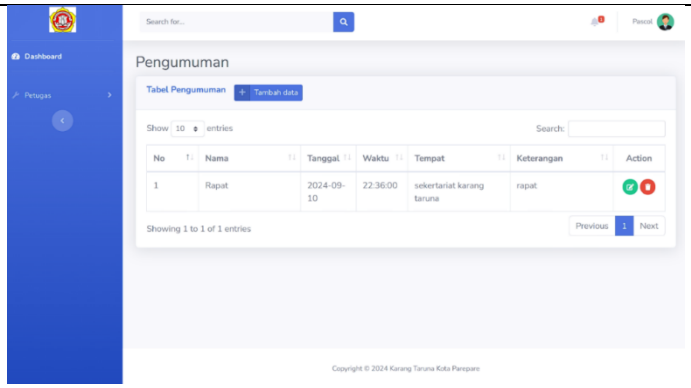
Tabel 4. 19 *Blackbox testing* kegiatan

Tes faktor	Hasil	Keterangan
Petugas menekan tombol navigasi petugas dan memilih pilihan kegiatan	✓	Sukses, tampil halaman kegiatan di *user* petugas
Screenshot		
		

t. *Blackbox testing* pengumuman

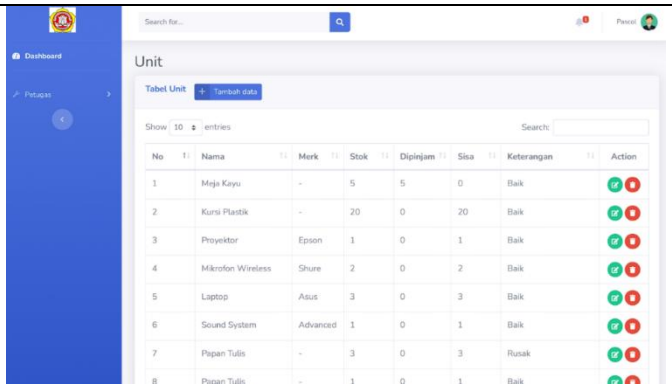
Tabel 4. 20 *Blackbox testing* pengumuman

Tes faktor	Hasil	Keterangan
Petugas menekan tombol navigasi petugas dan memilih pilihan pengumuman	✓	Sukses, tampil halaman pengumuman
Screenshot		



u. *Blackbox testing halaman* unit

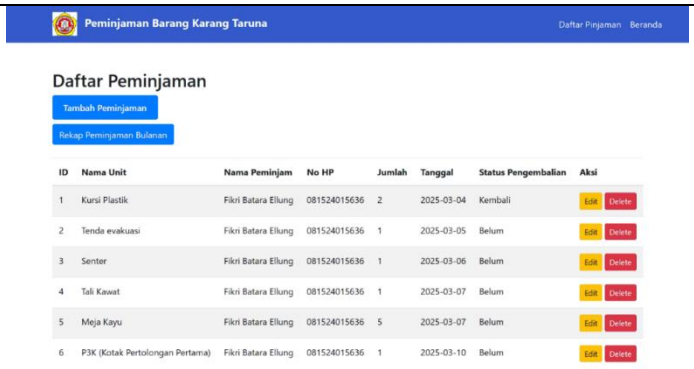
Tabel 4. 21 *Blackbox testing* halaman unit

Tes faktor	Hasil	Keterangan
Petugas menekan tombol navigasi petugas dan memilih pilihan unit	✓	Sukses, tampil halaman unit
Screenshot		
		

v. *Blackbox testing* data pinjaman

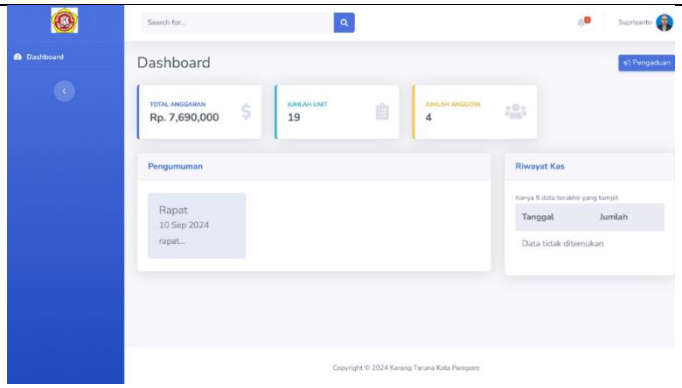
Tabel 4. 22 *Blackbox testing* data pinjaman

Tes faktor	Hasil	Keterangan
Petugas menekan tombol navigasi	✓	Sukses, tampil halaman data pinjaman

petugas dan memilih pilihan data pinjaman		
Screenshot		
		

w. *Blackbox testing dashboard* anggota

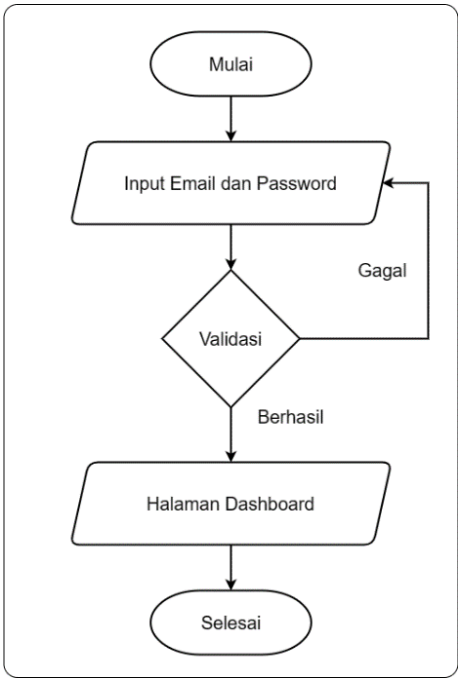
Tabel 4. 23 *Blackbox testing dashboard* anggota

Tes faktor	Hasil	Keterangan
Anggota berhasil masuk menggunakan akun anggota	✓	Sukses, tampil halaman *dashboard* anggota
Screenshot		
		

2. Whitebox

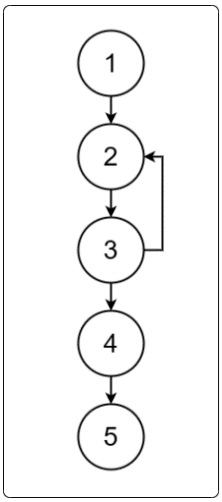
a. *Whitebox testing login* berhasil

1) *Flowchart*



Gambar 4. 45 *Flowchart login* berhasil

2) *Flowgraph*



Gambar 4. 46 *Flowgraph login* berhasil

Berdasarkan gambar 4. 46 diatas, dilakukan perhitungan sebagai berikut:

(1) Menghitung *cyclomatic complexity* V(G) pada *edge* dan *node*

Pada rumus : V(G) = E – N + 2

E (*edge*) = 5

N (*node*) = 5

P (predikat *node*) = 1

Penyelesaian :

V(G) = E – N + 2

= 5 – 5 + 2

= 2

Predikat (P) = P + 1

= 1 + 1

= 2

(2) Berdasarkan perhitungan *cyclomatic complexity* dari *Flowgraph* diatas memiliki *region = 2*

(3) *Independent path* pada *Flowgraph* yaitu:

Path 1 = 1 – 2 – 3 – 2

Path 2 = 1 – 2 – 3 – 4 – 5

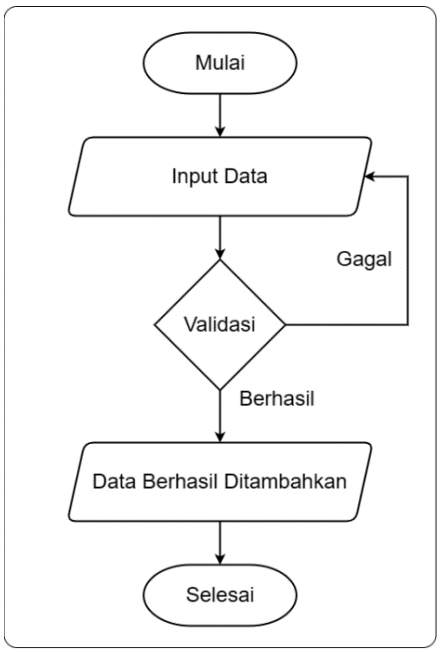
(4) Grafik matriks *login* berhasil

Tabel 4. 24 Grafik matriks *login* berhasil

	1	2	3	4	5	E-1
1		1				1 - 1 = 0
2			1			1 – 1 = 0
3		1		1		2 – 1 = 1
4					1	1 – 1 = 0
5						0
	SUM (E + 1)					1 + 1 = 2

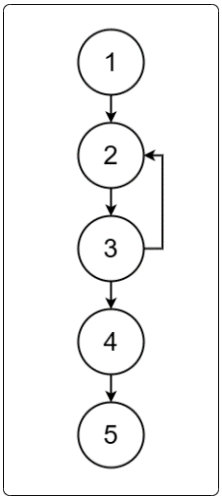
b. *Whitebox testing* tambah data

1) *Flowchart*



Gambar 4. 47 *Flowchart* tambah data

2) *Flowgraph*



Gambar 4. 48 *Flowgraph* tambah data

Berdasarkan gambar 4. 48 diatas, dilakukan perhitungan sebagai berikut:

(1) Menghitung *cyclomatic complexity* V(G) pada *edge* dan *node*

Pada rumus : V(G) = E – N + 2

E (*edge*) = 5

N (*node*) = 5

P (Predikat *node*) = 1

Penyelesaian :

V(G) = E – N + 2

= 5 – 5 + 2

= 2

Predikat (P) = P + 1

= 1 + 1

= 2

(2) Berdasarkan perhitungan *cyclomatic complexity* dari *Flowgraph* diatas memiliki *region = 2*

(3) *Independent path* pada *Flowgraph* yaitu:

Path 1 = 1 – 2 – 3 – 2

Path 2 = 1 – 2 – 3 – 4 – 5

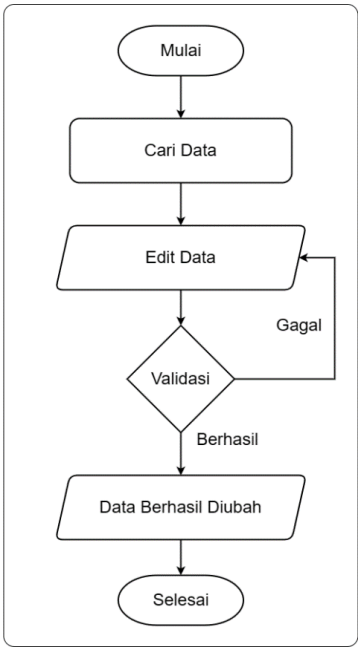
(4) Grafik matriks tambah data

Tabel 4. 25 Grafik matriks tambah data

	1	2	3	4	5	E-1
1		1				1 - 1 = 0
2			1			1 – 1 = 0
3		1		1		2 – 1 = 1
4					1	1 – 1 = 0
5						0
	SUM (E + 1)					1 + 1 = 2

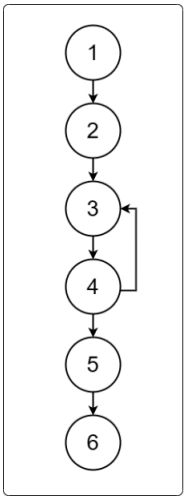
c. *Whitebox testing* ubah data

1) *Flowchart*



Gambar 4. 49 *Flowchart* ubah data

2) *Flowgraph*



Gambar 4. 50 *Flowgraph* ubah data

Berdasarkan gambar 4. 50 diatas, dilakukan perhitungan sebagai berikut:

(1) Menghitung *cyclomatic complexity* V(G) pada *edge* dan *node*

Pada rumus : V(G) = E – N + 2

E (*edge*) = 6

N (*node*) = 6

P (Predikat *node*) = 1

Penyelesaian :

V(G) = E – N + 2

= 6 – 6 + 2

= 2

Predikat (P) = P + 1

= 1 + 1

= 2

(2) Berdasarkan perhitungan *cyclomatic complexity* dari *Flowgraph* diatas memiliki *region = 2*

(3) *Independent path* pada *Flowgraph* yaitu:

Path 1 = 1 – 2 – 3 – 4 – 3

Path 2 = 1 – 2 – 3 – 4 – 5 – 6

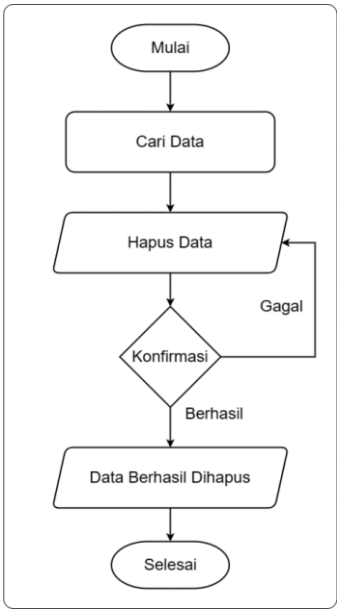
(4) Grafik matriks ubah data

Tabel 4. 26 Grafik matriks ubah data

	1	2	3	4	5	6	E-1
1		1					1 - 1 = 0
2			1				1 – 1 = 0
3				1			1 – 1 = 0
4			1		1		2 – 1 = 1
5						1	1 – 1 = 0
6							0
	SUM (E + 1)						1 + 1 = 2

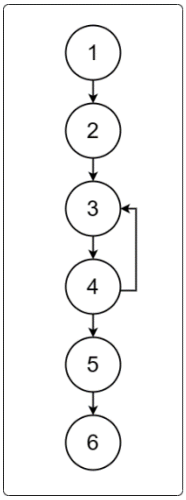
d. *Whitebox testing* hapus data

1) *Flowchart*



Gambar 4. 51 *Flowchart* hapus data

2) *Flowgraph*



Gambar 4. 52 *Flowgraph* hapus data

Berdasarkan gambar 4. 52 diatas, dilakukan perhitungan sebagai berikut:

(1) Menghitung *cyclomatic complexity* V(G) pada *edge* dan *node*

Pada rumus : V(G) = E – N + 2

E (*edge*) = 6

N (*node*) = 6

P (Predikat *node*) = 1

Penyelesaian :

V(G) = E – N + 2

= 6 – 6 + 2

= 2

Predikat (P) = P + 1

= 1 + 1

= 2

(2) Berdasarkan perhitungan *cyclomatic complexity* dari *Flowgraph* diatas memiliki *region = 2*

(3) *Independent path* pada *Flowgraph* yaitu:

Path 1 = 1 – 2 – 3 – 4 – 3 – 5 – 6

Path 2 = 1 – 2 – 3 – 4 – 5 – 6

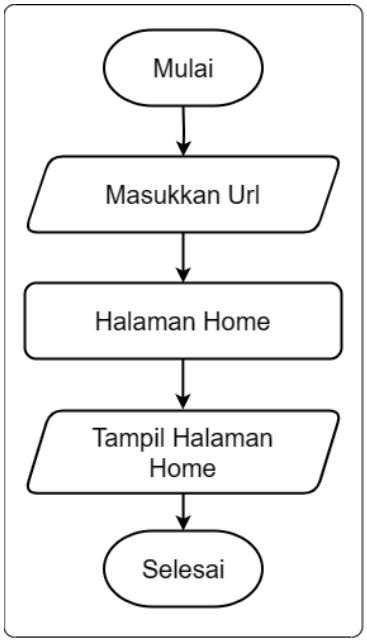
(4) Grafik matriks hapus data

Tabel 4. 27 Grafik matriks hapus data

	1	2	3	4	5	6	E-1
1		1					1 - 1 = 0
2			1				1 – 1 = 0
3				1			1 – 1 = 0
4			1		1		2 – 1 = 1
5						1	1 – 1 = 0
6							0
	SUM (E + 1)						1 + 1 = 2

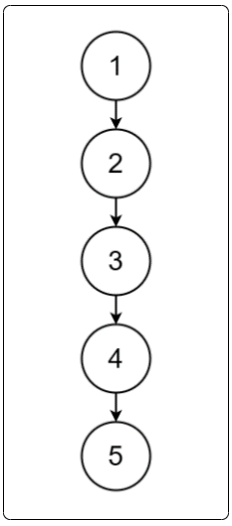
e. *Whitebox testing* halaman beranda/*home*

1) *Flowchart*



Gambar 4. 53 *Flowchart* halaman beranda/*home*

2) *Flowgraph*



Gambar 4. 54 *Flowgraph* halaman beranda/*home*

Berdasarkan gambar 4. 54 diatas, dilakukan perhitungan sebagai berikut:

(1) Menghitung *cyclomatic complexity* V(G) pada *edge* dan *node.*

Pada rumus : V(G) = E – N + 2

E (*edge*) = 4

N (*node*) = 5

P (Predikat *node*) = 0

Penyelesaian :

V(G) = E – N + 2

= 4 – 5 + 2

= 1

Predikat (P) = P + 1

= 0 + 1

= 1

(2) Berdasarkan perhitungan *cyclomatic complexity* dari *Flowgraph* diatas memiliki *region = 1*

(3) *Independent path* pada *Flowgraph* yaitu:

Path 1 = 1 – 2 – 3 – 4 – 5

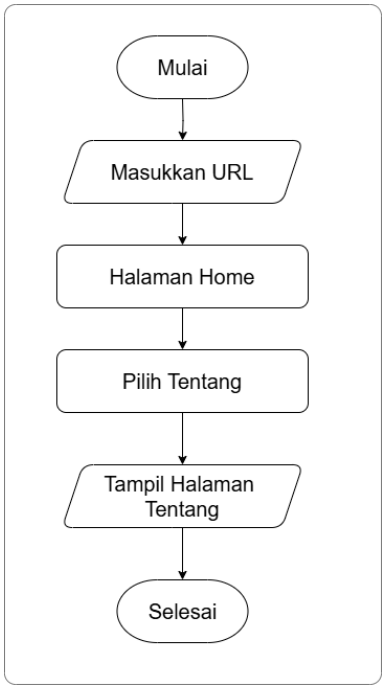
(4) Grafik matriks halaman beranda/*home*

Tabel 4. 28 Grafik matriks halaman beranda/*home*

	1	2	3	4	5	E-1
1		1				1 - 1 = 0
2			1			1 – 1 = 0
3				1		1 – 1 = 0
4					1	1 – 1 = 0
5						0
	SUM (E + 1)					0 + 1 = 1

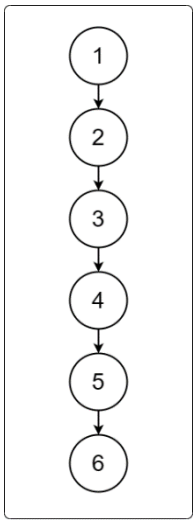
f. *Whitebox testing* halaman *login*

1) *Flowchart*



Gambar 4. 55 *Flowchart* halaman tentang/*login*

2) *Flowgraph*



Gambar 4. 56 *Flowgraph* halaman tentang/*login*

Berdasarkan gambar 4. 56 diatas, dilakukan perhitungan sebagai berikut:

(1) Menghitung *cyclomatic complexity* V(G) pada *edge* dan *node.*

Pada rumus : V(G) = E – N + 2

E (*edge*) = 5

N (*node*) = 6

P (Predikat *node*) = 0

Penyelesaian :

V(G) = E – N + 2

= 5 – 6 + 2

= 1

Predikat (P) = P + 1

= 0 + 1

= 1

(2) Berdasarkan perhitungan *cyclomatic complexity* dari *Flowgraph* diatas memiliki *region = 1*

(3) *Independent path* pada *Flowgraph* yaitu:

Path 1 = 1 – 2 – 3 – 4 – 5 – 6

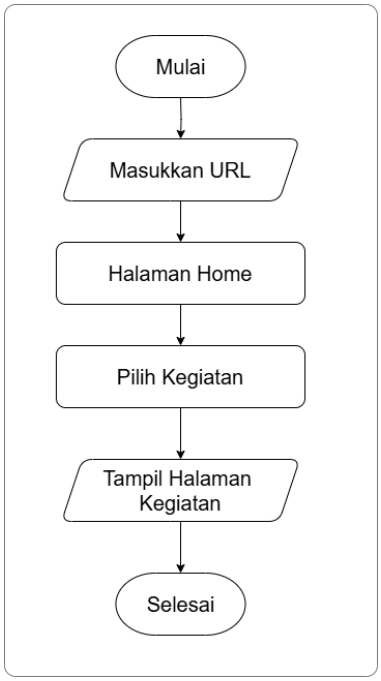
(4) Grafik matriks halaman tentang/*login*

Tabel 4. 29 Grafik matriks halaman tentang/*login*

	1	2	3	4	5	6	E-1
1		1					1 - 1 = 0
2			1				1 – 1 = 0
3				1			1 – 1 = 0
4					1		1 – 1 = 0
5						1	1 – 1 = 0
6							0
	SUM (E + 1)						0 + 1 = 1

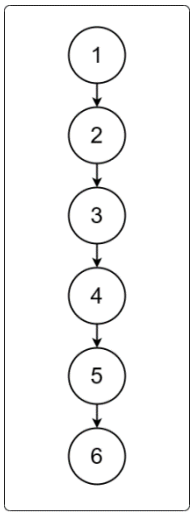
g. *Whitebox testing* halaman kegiatan

1) *Flowchart*



Gambar 4. 57 *Flowchart* halaman kegiatan

2) *Flowgraph*



Gambar 4. 58 *Flowgraph* halaman kegiatan

Berdasarkan gambar 4. 58 diatas, dilakukan perhitungan sebagai berikut:

(1) Menghitung *cyclomatic complexity* V(G) pada *edge* dan *node.*

Pada rumus : V(G) = E – N + 2

E (*edge*) = 5

N (*node*) = 6

P (Predikat *node*) = 0

Penyelesaian :

V(G) = E – N + 2

= 5 – 6 + 2

= 1

Predikat (P) = P + 1

= 0 + 1

= 1

(2) Berdasarkan perhitungan *cyclomatic complexity* dari *Flowgraph* diatas memiliki *region = 1*

(3) *Independent path* pada *Flowgraph* yaitu:

Path 1 = 1 – 2 – 3 – 4 – 5 – 6

(4) Grafik matriks halaman kegiatan

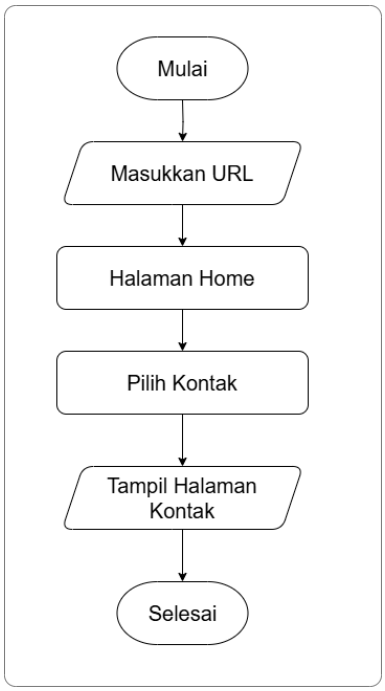
Tabel 4. 30 Grafik matriks halaman kegiatan

	1	2	3	4	5	6	E-1
1		1					1 - 1 = 0

2			1				1 – 1 = 0
3				1			1 – 1 = 0
4					1		1 – 1 = 0
5						1	1 – 1 = 0
6							0
	SUM (E + 1)						0 + 1 = 1

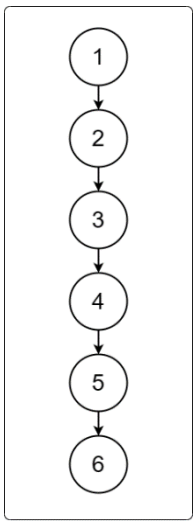
h. *Whitebox testing* halaman kontak

1) *Flowchart*



Gambar 4. 59 *Flowchart* halaman kontak

2) *Flowgraph*



Gambar 4. 60 *Flowgraph* halaman kontak

Berdasarkan gambar 4. 60 diatas, dilakukan perhitungan sebagai berikut:

(1) Menghitung *cyclomatic complexity* V(G) pada *edge* dan *node.*

Pada rumus : V(G) = E – N + 2

E (*edge*) = 5

N (*node*) = 6

P (Predikat *node*) = 0

Penyelesaian :

V(G) = E – N + 2

= 5 – 6 + 2

= 1

Predikat (P) = P + 1

= 0 + 1

= 1

(2) Berdasarkan perhitungan *cyclomatic complexity* dari *Flowgraph* diatas memiliki *region = 1*

(3) *Independent path* pada *Flowgraph* yaitu:

Path 1 = 1 – 2 – 3 – 4 – 5 – 6

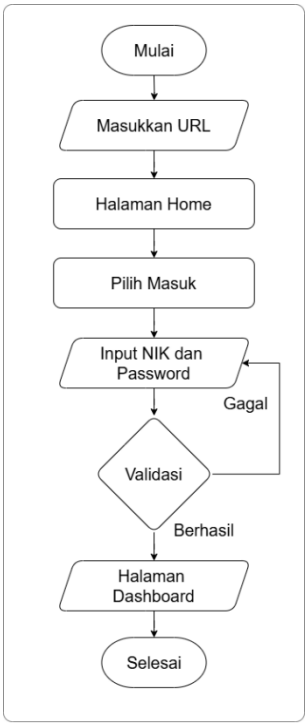
(4) Grafik matriks halaman kontak

Tabel 4. 31 Grafik matriks halaman kontak

	1	2	3	4	5	6	E-1
1		1					1 - 1 = 0
2			1				1 – 1 = 0
3				1			1 – 1 = 0
4					1		1 – 1 = 0
5						1	1 – 1 = 0
6							0
	SUM (E + 1)						0 + 1 = 1

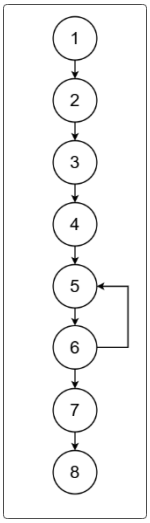
i. *Whitebox testing* halaman *dashboard admin*

1) *Flowchart*



Gambar 4. 61 *Flowchart dashboard admin*

2) *Flowgraph*



Gambar 4. 62 *Flowgraph dashboard admin*

Berdasarkan gambar 4. 62 diatas, dilakukan perhitungan sebagai berikut:

(1) Menghitung *cyclomatic complexity* V(G) pada *edge* dan *node.*

Pada rumus : V(G) = E – N + 2

E (*edge*) = 8

N (*node*) = 8

P (Predikat *node*) = 1

Penyelesaian :

V(G) = E – N + 2

= 8 – 8 + 2

= 2

Predikat (P) = P + 1

= 1 + 1

= 2

(2) Berdasarkan perhitungan *cyclomatic complexity* dari *Flowgraph* diatas memiliki *region* = 2

(3) *Independent path* pada *Flowgraph* yaitu:

Path 1 = 1 – 2 – 3 – 4 – 5 – 6 – 7 – 8

Path 2 = 1 – 2 – 3 – 4 – 5 – 6 – 5 – 6 – 7 – 8

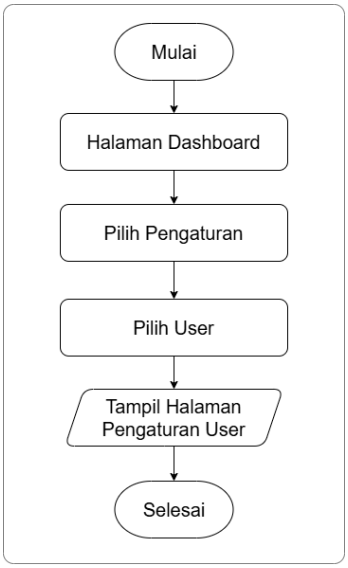
(4) Grafik matriks *dashboard admin*

Tabel 4. 32 Grafik matriks *dashboard admin*

	1	2	3	4	5	6	7	8	E - 1
1		1							1 – 1 = 0
2			1						1 – 1 = 0
3				1					1 – 1 = 0
4					1				1 – 1 = 0
5						1			1 – 1 = 0
6					1		1		2 – 1 = 1
7								1	1 – 1 = 0
8									0
	SUM (E + 1)								1 + 1 = 2

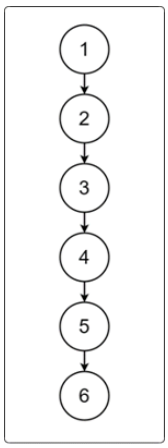
j. *Whitebox testing* halaman pengaturan *user*

1) *Flowchart*



Gambar 4. 63 *Flowchart* halaman pengaturan *user*

2) *Flowgraph*



Gambar 4. 64 *Flowgraph* halaman pengaturan *user*

Berdasarkan gambar 4. 64 diatas, dilakukan perhitungan sebagai berikut:

(1) Menghitung *cyclomatic complexity* V(G) pada *edge* dan *node.*

Pada rumus : V(G) = E – N + 2

E (*edge*) = 5

N (*node*) = 6

P (Predikat *node*) = 0

Penyelesaian :

V(G) = E – N + 2

= 5 – 6 + 2

= 1

Predikat (P) = P + 1

= 0 + 1

= 1

(2) Berdasarkan perhitungan *cyclomatic complexity* dari *Flowgraph* diatas memiliki *region = 1*

(3) *Independent path* pada *Flowgraph* yaitu:

Path 1 = 1 – 2 – 3 – 4 – 5 – 6

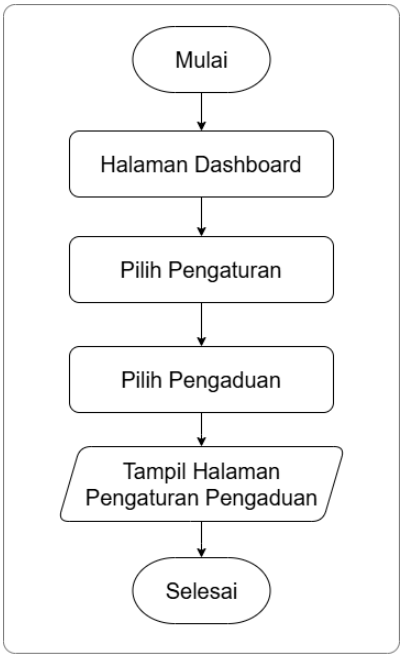
(4) Grafik matriks halaman pengaturan *user*

Tabel 4. 33 Grafik matriks halaman pengaturan *user*

	1	2	3	4	5	6	E-1
1		1					1 - 1 = 0
2			1				1 – 1 = 0
3				1			1 – 1 = 0
4					1		1 – 1 = 0
5						1	1 – 1 = 0
6							0
	SUM (E + 1)						0 + 1 = 1

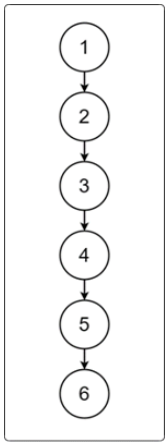
k. *Whitebox testing* halaman pengaturan pengaduan

1) *Flowchart*



Gambar 4. 65 *Flowchart* halaman pengaturan pengaduan

2) *Flowgraph*



Gambar 4. 66 *Flowgraph* halaman pengaturan pengaduan

Berdasarkan gambar 4. 66 diatas, dilakukan perhitungan sebagai berikut:

(1) Menghitung *cyclomatic complexity* V(G) pada *edge* dan *node.*

Pada rumus : V(G) = E – N + 2

E (*edge*) = 5

N (*node*) = 6

P (Predikat *node*) = 0

Penyelesaian :

V(G) = E – N + 2

= 5 – 6 + 2

= 1

Predikat (P) = P + 1

= 0 + 1

= 1

(2) Berdasarkan perhitungan *cyclomatic complexity* dari *Flowgraph* diatas memiliki *region = 1*

(3) *Independent path* pada *Flowgraph* yaitu:

Path 1 = 1 – 2 – 3 – 4 – 5 – 6

(4) Grafik matriks halaman pengaturan pengaduan

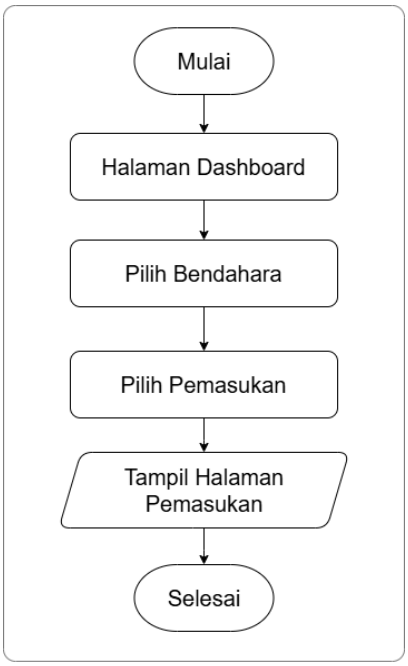
Tabel 4. 34 Grafik matriks halaman pengaturan pengaduan

	1	2	3	4	5	6	E-1
1		1					1 - 1 = 0
2			1				1 – 1 = 0
3				1			1 – 1 = 0
4					1		1 – 1 = 0

5						1	1 – 1 = 0
6							0
	SUM (E + 1)						0 + 1 = 1

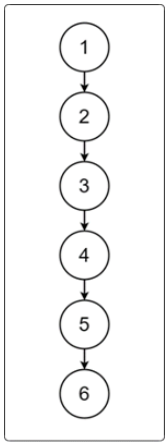
l. *Whitebox testing master* pemasukan

1) *Flowchart*



Gambar 4. 67 *Flowchart* halaman pemasukan

2) *Flowgraph*



Gambar 4. 68 *Flowgraph* halaman pemasukan

Berdasarkan gambar 4. 68 diatas, dilakukan perhitungan sebagai berikut:

(1) Menghitung *cyclomatic complexity* V(G) pada *edge* dan *node.*

Pada rumus : V(G) = E – N + 2

E (*edge*) = 5

N (*node*) = 6

P (Predikat *node*) = 0

Penyelesaian :

V(G) = E – N + 2

= 5 – 6 + 2

= 1

Predikat (P) = P + 1

= 0 + 1

= 1

(2) Berdasarkan perhitungan *cyclomatic complexity* dari *Flowgraph* diatas memiliki *region = 1*

(3) *Independent path* pada *Flowgraph* yaitu:

Path 1 = 1 – 2 – 3 – 4 – 5 – 6

(4) Grafik matriks halaman pemasukan

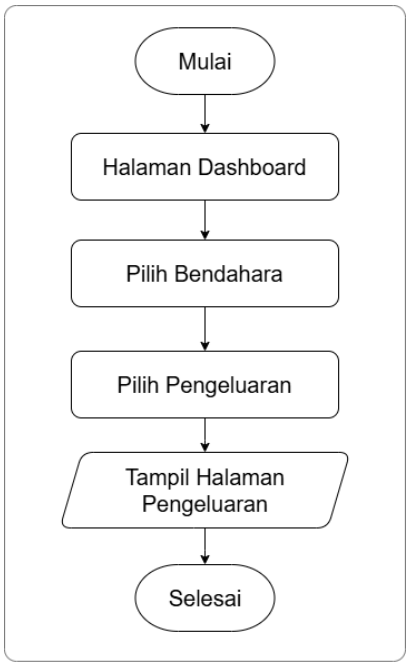
Tabel 4. 35 Grafik matriks halaman pemasukan

	1	2	3	4	5	6	E-1
1		1					1 - 1 = 0
2			1				1 – 1 = 0
3				1			1 – 1 = 0
4					1		1 – 1 = 0
5						1	1 – 1 = 0

6							0
	SUM (E + 1)						0 + 1 = 1

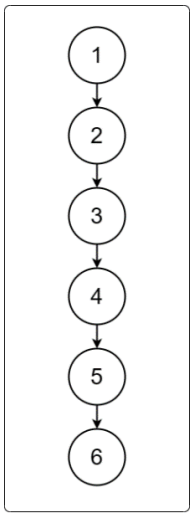
m. *Whitebox testing master* pengeluaran

1) *Flowchart*



Gambar 4. 69 *Flowchart* halaman pengeluaran

2) *Flowgraph*



Gambar 4. 70 *Flowgraph* halaman pengeluaran

Berdasarkan gambar 4. 70 diatas, dilakukan perhitungan sebagai berikut:

(1) Menghitung *cyclomatic complexity* V(G) pada *edge* dan *node.*

Pada rumus : V(G) = E – N + 2

E (*edge*) = 5

N (*node*) = 6

P (Predikat *node*) = 0

Penyelesaian :

V(G) = E – N + 2

= 5 – 6 + 2

= 1

Predikat (P) = P + 1

= 0 + 1

= 1

(2) Berdasarkan perhitungan *cyclomatic complexity* dari *Flowgraph* diatas memiliki *region = 1*

(3) *Independent path* pada *Flowgraph* yaitu:

Path 1 = 1 – 2 – 3 – 4 – 5 – 6

(4) Grafik matriks halaman pengeluaran

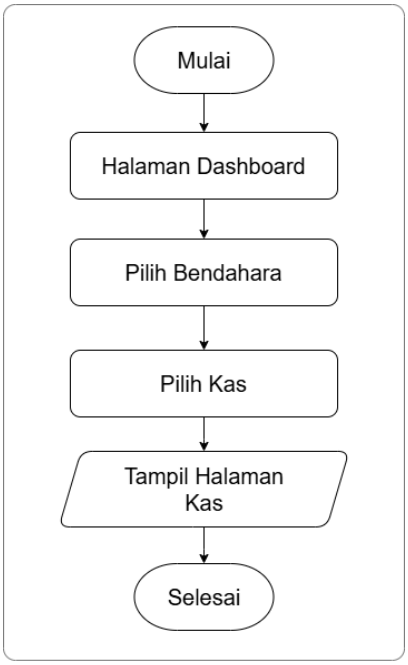
Tabel 4. 36 Grafik matriks halaman pengeluaran

	1	2	3	4	5	6	E-1
1		1					1 - 1 = 0
2			1				1 – 1 = 0
3				1			1 – 1 = 0
4					1		1 – 1 = 0
5						1	1 – 1 = 0

6							0
	SUM (E + 1)						0 + 1 = 1

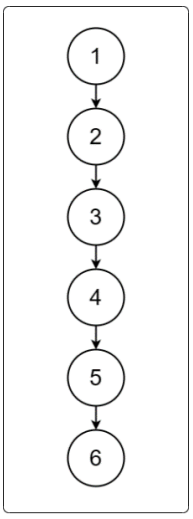
n. *Whitebox testing* halaman transaksi kas

1) *Flowchart*



Gambar 4. 71 *Flowchart* halaman kas

2) *Flowgraph*



Gambar 4. 72 *Flowgraph* halaman kas

Berdasarkan gambar 4. 72 diatas, dilakukan perhitungan sebagai berikut:

(1) Menghitung *cyclomatic complexity* V(G) pada *edge* dan *node.*

Pada rumus : V(G) = E – N + 2

E (*edge*) = 5

N (*node*) = 6

P (Predikat *node*) = 0

Penyelesaian :

V(G) = E – N + 2

= 5 – 6 + 2

= 1

Predikat (P) = P + 1

= 0 + 1

= 1

(2) Berdasarkan perhitungan *cyclomatic complexity* dari *Flowgraph* diatas memiliki *region = 1*

(3) *Independent path* pada *Flowgraph* yaitu:

Path 1 = 1 – 2 – 3 – 4 – 5 – 6

(4) Grafik matriks halaman kas

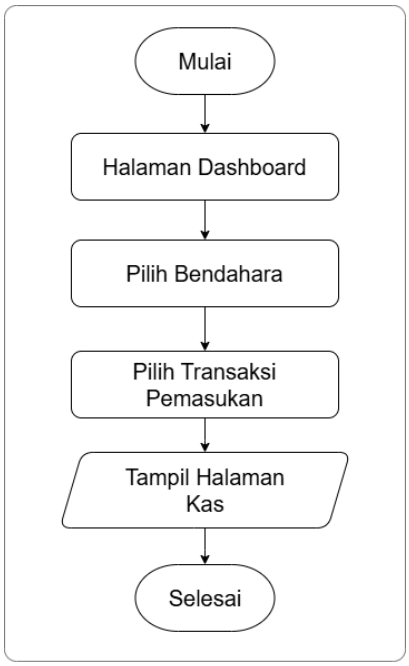
Tabel 4. 37 Grafik matriks halaman kas

	1	2	3	4	5	6	E-1
1		1					1 - 1 = 0
2			1				1 – 1 = 0
3				1			1 – 1 = 0
4					1		1 – 1 = 0
5						1	1 – 1 = 0

6							0
	SUM (E + 1)						0 + 1 = 1

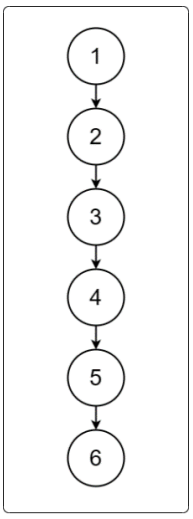
o. *Whitebox testing* halaman transaksi pemasukan

1) *Flowchart*



Gambar 4. 73 *Flowchart* halaman transaksi pemasukan

2) *Flowgraph*



Gambar 4. 74 *Flowgraph* halaman transaksi pemasukan

Berdasarkan gambar 4. 74 diatas, dilakukan perhitungan sebagai berikut:

(1) Menghitung *cyclomatic complexity* V(G) pada *edge* dan *node.*

Pada rumus : V(G) = E – N + 2

E (*edge*) = 5

N (*node*) = 6

P (Predikat *node*) = 0

Penyelesaian :

V(G) = E – N + 2

= 5 – 6 + 2

= 1

Predikat (P) = P + 1

= 0 + 1

= 1

(2) Berdasarkan perhitungan *cyclomatic complexity* dari *Flowgraph* diatas memiliki *region = 1*

(3) *Independent path* pada *Flowgraph* yaitu:

Path 1 = 1 – 2 – 3 – 4 – 5 – 6 – 6

(4) Grafik matriks halaman transaksi pemasukan

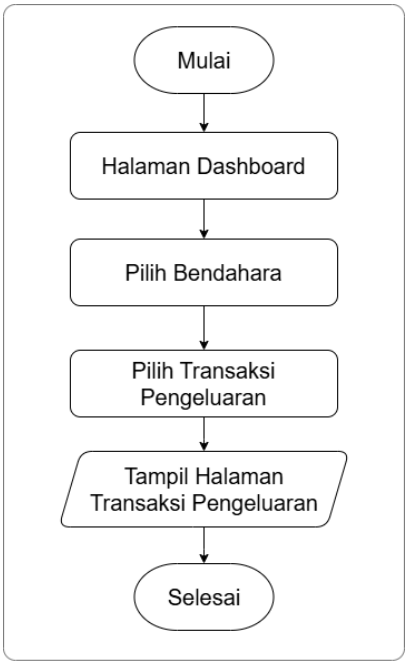
Tabel 4. 38 Grafik matriks halaman transaksi pemasukan

	1	2	3	4	5	6	E-1
1		1					1 - 1 = 0
2			1				1 – 1 = 0
3				1			1 – 1 = 0
4					1		1 – 1 = 0
5						1	1 – 1 = 0

6							0
	SUM (E + 1)						0 + 1 = 1

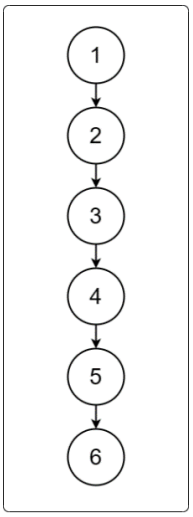
p. *Whitebox testing* halaman transaksi pengeluaran

1) *Flowchart*



Gambar 4. 75 *Flowchart* halaman transaksi pengeluaran

2) *Flowgraph*



Gambar 4. 76 *Flowgraph* halaman transaksi pengeluaran

Berdasarkan gambar 4. 76 diatas, dilakukan perhitungan sebagai berikut:

(1) Menghitung *cyclomatic complexity* V(G) pada *edge* dan *node.*

Pada rumus : V(G) = E – N + 2

E (*edge*) = 5

N (*node*) = 6

P (Predikat *node*) = 0

Penyelesaian :

V(G) = E – N + 2

= 5 – 6 + 2

= 1

Predikat (P) = P + 1

= 0 + 1

= 1

(2) Berdasarkan perhitungan *cyclomatic complexity* dari *Flowgraph* diatas memiliki *region = 1*

(3) *Independent path* pada *Flowgraph* yaitu:

Path 1 = 1 – 2 – 3 – 4 – 5 – 6

(4) Grafik matriks halaman transaksi pengeluaran

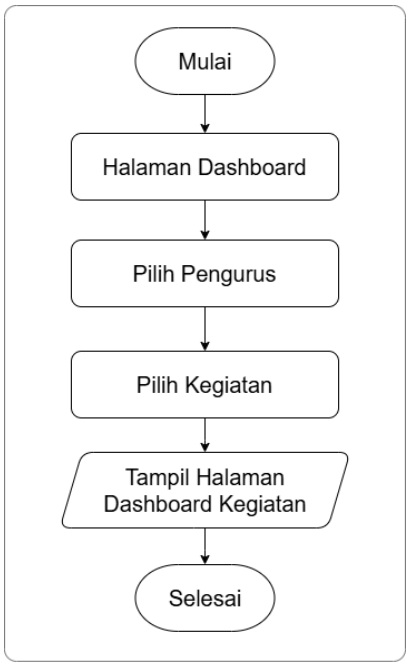
Tabel 4. 39 Grafik matriks halaman transaksi pengeluaran

	1	2	3	4	5	6	E-1
1		1					1 - 1 = 0
2			1				1 – 1 = 0
3				1			1 – 1 = 0
4					1		1 – 1 = 0
5						1	1 – 1 = 0

6							0
	SUM (E + 1)						0 + 1 = 1

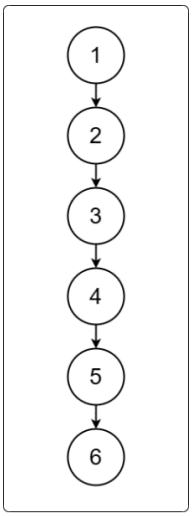
q. *Whitebox testing dashboard* kegiatan

1) *Flowchart*



Gambar 4. 77 *Flowchart* halaman *dashboard* kegiatan

2) *Flowgraph*



Gambar 4. 78 *Flowgraph* halaman *dashboard* kegiatan

Berdasarkan gambar 4. 78 diatas, dilakukan perhitungan sebagai berikut:

(1) Menghitung *cyclomatic complexity* V(G) pada *edge* dan *node.*

Pada rumus : V(G) = E – N + 2

E (*edge*) = 5

N (*node*) = 6

P (Predikat *node*) = 0

Penyelesaian :

V(G) = E – N + 2

= 5 – 6 + 2

= 1

Predikat (P) = P + 1

= 0 + 1

= 1

(2) Berdasarkan perhitungan *cyclomatic complexity* dari *Flowgraph* diatas memiliki *region = 1*

(3) *Independent path* pada *Flowgraph* yaitu:

Path 1 = 1 – 2 – 3 – 4 – 5 – 6

(4) Grafik matriks halaman *dashboard* kegiatan

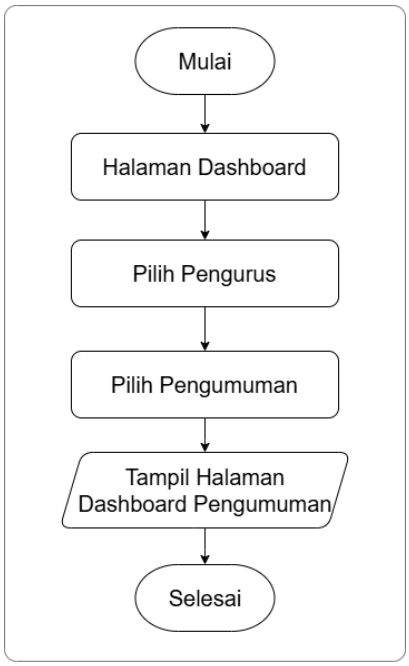
Tabel 4. 40 Grafik matriks halaman *dashboard* kegiatan

	1	2	3	4	5	6	E-1
1		1					1 - 1 = 0
2			1				1 – 1 = 0
3				1			1 – 1 = 0
4					1		1 – 1 = 0
5						1	1 – 1 = 0

6							0
	SUM (E + 1)						0 + 1 = 1

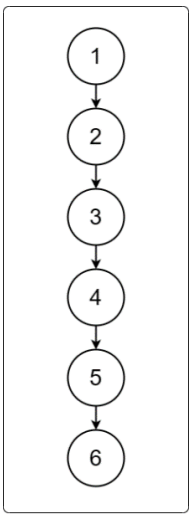
r. *Whitebox testing dashboard* pengumuman

1) *Flowchart*



Gambar 4. 79 *Flowchart dashboard* pengumuman

2) *Flowgraph*



Gambar 4. 80 *Flowgraph dashboard* pengumuman

Berdasarkan gambar 4. 80 diatas, dilakukan perhitungan sebagai berikut:

(1) Menghitung *cyclomatic complexity* V(G) pada *edge* dan *node.*

Pada rumus : V(G) = E – N + 2

E (*edge*) = 5

N (*node*) = 6

P (Predikat *node*) = 0

Penyelesaian :

V(G) = E – N + 2

= 5 – 6 + 2

= 1

Predikat (P) = P + 1

= 0 + 1

= 1

(2) Berdasarkan perhitungan *cyclomatic complexity* dari *Flowgraph* diatas memiliki *region = 1*

(3) *Independent path* pada *Flowgraph* yaitu:

Path 1 = 1 – 2 – 3 – 4 – 5 – 6

(4) Grafik matriks halaman *dashboard* pengumuman

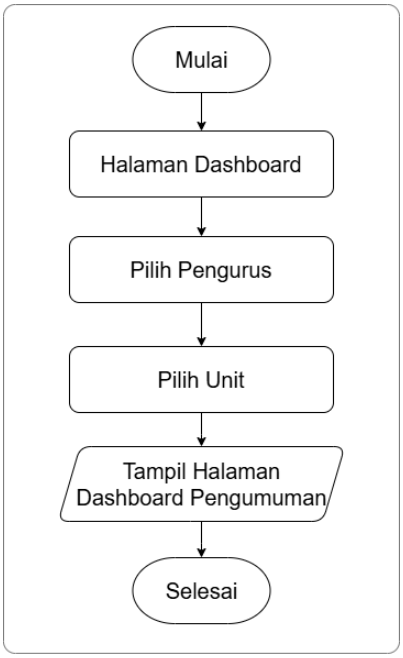
Tabel 4. 41 Grafik matriks halaman *dashboard* pengumuman

	1	2	3	4	5	6	E-1
1		1					1 - 1 = 0
2			1				1 – 1 = 0
3				1			1 – 1 = 0
4					1		1 – 1 = 0
5						1	1 – 1 = 0

6							0
	SUM (E + 1)						0 + 1 = 1

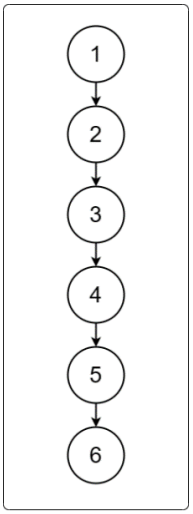
s. *Whitebox testing dashboard* unit

1) *Flowchart*



Gambar 4. 81 *Flowchart* halaman *dashboard* unit

2) *Flowgraph*



Gambar 4. 82 *Flowgraph* halaman *dashboard* unit

Berdasarkan gambar 4. 82 diatas, dilakukan perhitungan sebagai berikut:

(1) Menghitung *cyclomatic complexity* V(G) pada *edge* dan *node.*

Pada rumus : V(G) = E – N + 2

E (*edge*) = 5

N (*node*) = 6

P (Predikat *node*) = 0

Penyelesaian :

V(G) = E – N + 2

= 5 – 6 + 2

= 1

Predikat (P) = P + 1

= 0 + 1

= 1

(2) Berdasarkan perhitungan *cyclomatic complexity* dari *Flowgraph* diatas memiliki *region = 1*

(3) *Independent path* pada *Flowgraph* yaitu:

Path 1 = 1 – 2 – 3 – 4 – 5 – 6

(4) Grafik matriks halaman *dashboard* unit

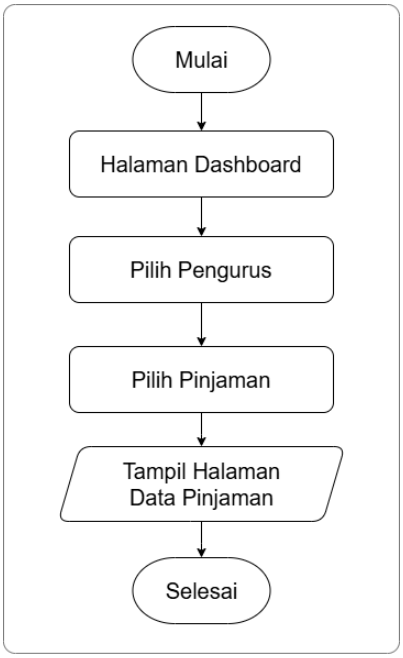
Tabel 4. 42 Grafik matriks halaman *dashboard* unit

	1	2	3	4	5	6	E-1
1		1					1 - 1 = 0
2			1				1 – 1 = 0
3				1			1 – 1 = 0
4					1		1 – 1 = 0
5						1	1 – 1 = 0

6							0
	SUM (E + 1)						0 + 1 = 1

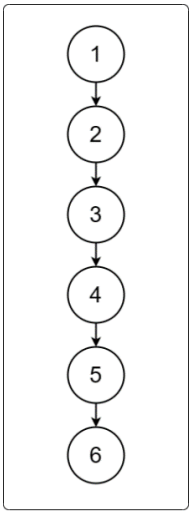
t. *Whitebox testing* data pinjaman

1) *Flowchart*



Gambar 4. 83 *Flowchart* halaman data pinjaman

2) *Flowgraph*



Gambar 4. 84 *Flowgraph* halaman data pinjaman

Berdasarkan gambar 4. 84 diatas, dilakukan perhitungan sebagai berikut:

(1) Menghitung *cyclomatic complexity* V(G) pada *edge* dan *node.*

Pada rumus : V(G) = E – N + 2

E (*edge*) = 5

N (*node*) = 6

P (Predikat *node*) = 0

Penyelesaian :

V(G) = E – N + 2

= 5 – 6 + 2

= 1

Predikat (P) = P + 1

= 0 + 1

= 1

(2) Berdasarkan perhitungan *cyclomatic complexity* dari *Flowgraph* diatas memiliki *region = 1*

(3) *Independent path* pada *Flowgraph* yaitu:

Path 1 = 1 – 2 – 3 – 4 – 5 – 6

(4) Grafik matriks halaman data pinjaman

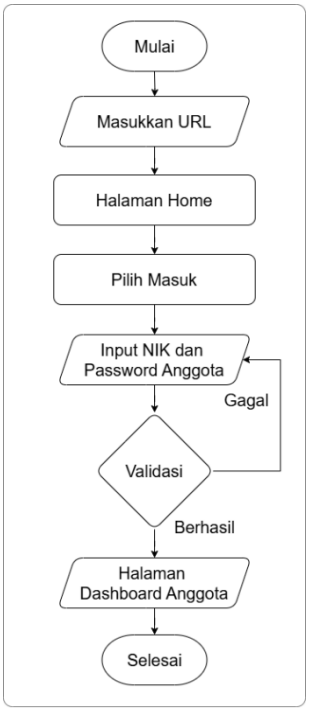
Tabel 4. 43 Grafik matriks halaman data pinjaman

	1	2	3	4	5	6	E-1
1		1					1 - 1 = 0
2			1				1 – 1 = 0
3				1			1 – 1 = 0
4					1		1 – 1 = 0
5						1	1 – 1 = 0

6							0
	SUM (E + 1)						0 + 1 = 1

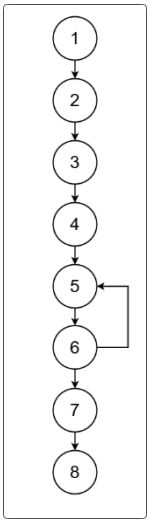
u. *Whitebox testing dashboard* anggota

1) *Flowchart*



Gambar 4. 85 *Flowchart* halaman *dashboard* anggota

2) *Flowgraph*



Gambar 4. 86 *Flowgraph* halaman *dashboard* anggota

Berdasarkan gambar 4. 86 diatas, dilakukan perhitungan sebagai berikut:

(1) Menghitung *cyclomatic complexity* V(G) pada *edge* dan *node.*

Pada rumus : V(G) = E – N + 2

E (*edge*) = 8

N (*node*) = 8

P (Predikat *node*) = 0

Penyelesaian :

V(G) = E – N + 2

= 8 – 8 + 2

= 2

Predikat (P) = P + 1

= 1 + 1

= 2

(2) Berdasarkan perhitungan *cyclomatic complexity* dari *Flowgraph* diatas memiliki *region = 2*

(3) *Independent path* pada *Flowgraph* yaitu:

Path 1 = 1 – 2 – 3 – 4 – 5 – 6 – 7 – 8

Path 2 = 1 – 2 – 3 – 4 – 5 – 6 – 5 – 6 – 7 – 8

(4) Grafik matriks halaman *dashboard* anggota

Tabel 4. 44 Grafik matriks halaman *dashboard* anggota

	1	2	3	4	5	6	7	8	E - 1
1		1							1 – 1 = 0
2			1						1 – 1 = 0
3				1					1 – 1 = 0
4					1				1 – 1 = 0
5						1			1 – 1 = 0
6					1		1		2 – 1 = 1
7								1	1 – 1 = 0
8									0
	SUM (E + 1)								1 + 1 = 2

BAB V

PENUTUP

A. Kesimpulan

Berdasarkan hasil penelitian yang telah dilakukan, dapat disimpulkan bahwa:

1. Sistem Informasi Manajemen Pengelolaan Kegiatan Karang Taruna Berbasis *Website* yang telah dikembangkan dapat membantu dalam pencatatan, pengelolaan, dan penyebaran informasi terkait kegiatan Karang Taruna di Kota Parepare secara lebih efektif dan efisien.
2. Fitur peminjaman barang dengan pemindaian wajah yang telah diimplementasikan meningkatkan keamanan dan akurasi dalam proses peminjaman. Dengan adanya verifikasi identitas berbasis pengenalan wajah, hanya anggota resmi yang dapat meminjam barang, sehingga mengurangi resiko kehilangan atau penyalahgunaan inventaris Karang Taruna.

B. Saran

1. Mengingat keterbatasan sistem yang saat ini hanya berbasis *website*, disarankan untuk mengembangkan versi aplikasi mobile (Android/iOS). Hal ini akan meningkatkan aksesibilitas dan kemudahan bagi pengguna dalam mengelola serta mengakses informasi kegiatan Karang Taruna kapan saja dan di mana saja.
2. Untuk meningkatkan efektivitas sistem, disarankan agar sistem dapat diintegrasikan dengan platform lain seperti media sosial untuk memperluas jangkauan informasi.

DAFTAR PUSTAKA

Bagye, W. (2020). MISI (Jurnal Manajemen Informatika & Sistem Informasi). LPPM STMIK Lombok, 3, 123–130.

Manurian, W., Mubarok, I., Sera Agustin, A., & Sania, N. (n.d.). Perancangan Sistem Informasi Pencatatan Point Pelanggaran Tata Tertib Siswa Berbasis *Website* Pada SMK YP Karya 1 Tanggerang.

Nasution, N., & Hasan Mhd Arief. (2018). Rancang Bangun Aplikasi Pembuatan *Web* Blog Berbasis *Web* Menggunakan HTML 5. Invotek Polbeng - Seri Informatika, 3, 68–72.

Nur Amanda, I. N. (2021). Rancang Bangun Sistem Informasi Media Kegiatan Pada Lembaga Karang Taruna Kecamatan Sepatan Timur. JIKA (Jurnal Informatika), 5(1), 104.

Pratiwi, D. (2021). Strategi Implementasi Sistem Informasi di Organisasi. Yogyakarta: Andi.

Rapiyanta, P. T., & Regen, T. B. P. (2021). Sistem Informasi Pengelolaan Kegiatan Karang Taruna (SIPEKATAR) Naralatu Agantuka Berbasis *Website*. Prosiding Nasional Rekayasa Teknologi Industri dan Informasi XIV Tahun 2019, 324-331.

Ronaldo, M., & Pasha, D. (2021). Sistem Informasi Pengelolaan Data Santri Pondok Pesantren An-Ahl Berbasis *Website*. Telefortech, 2(1), 17–20.

Sadewo, M. H. (2022). Aplikasi Rental Mobil Menggunakan Framework Bootstrap (Studi Kasus: Rental Mobil Wartini).

Setiawan, A. (2022). Manajemen Sistem Informasi: Teori dan Praktik. Bandung: Informatika.

Sofiyan, A., Linarta, A., & Sugiarto, M. I. (2017). Aplikasi Layanan Informasi Absen Nilai Berbasis SMS Gayewey Menggunakan PHP pada SMK Taruna Persada Dumai. Jurnal Informatika, Manajemen Dan Komputer, 9(2), 60– 69.

Sudesta. (2021). Sistem Informasi Karang Taruna Berbasis *Web* Sebagai Media Komunikasi Warga Masyarakat Pada Kantor Kelurahan Bojong Jaya. Jurnal Topik Global.

Voutama, A. (2022). Sistem Antrian Cucian Mobil Berbasis *Website* Menggunakan Konsep CRM dan Penerapan UML. Komputika: Jurnal Sistem Komputer, 11(1), 102–111. https://doi.org/10.34010/komputika.v11i1.4677