p-ISSN : 2746-7635
e-ISSN : 2808-5027

Merancang UML *(Unified Modelling Language)* Aplikasi e-JO Departemen Instalasi PT. ANDS

Muhammad Ridwan Arif Cahyono[1)]
Program Studi Teknologi Informasi, Politeknik Gajah Tunggal
ridwan@poltek-gt.ac.id

Nabila Putri Rendra[2)]
Program Studi Teknik Elektronika, Politeknik Gajah Tunggal
bilapr70@gmail.com

Anis Choirunnisa[3)]
Teknologi Informasi, Politeknik Gajah Tunggal
anis@poltek-gt.ac.id

ABSTRAK

Pada proses pengajuan *job order*, saat ini masih menggunakan kertas sebagai media pembuatan *job order* yang dapat menyebabkan kesulitan dalam pencarian dan pengarsipan dokumen. Selain itu, tidak ada sistem yang memadai untuk *monitoring job order* dan proses pengajuan *job order* juga memakan waktu cukup lama. Untuk mengatasi masalah tersebut, penelitian ini merancang sistem e-JO berbasis *website* dengan menggunakan UML (*Unified Modelling Language*) sebagai bahasa pemodelan visual bagi para pengguna untuk memudahkan para pengguna dalam memahami dan berinteraksi dengan sistem. Perancangan UML dilakukan sebagai langkah awal sebelum pengembangan aplikasi dimulai. Sistem ini dirancang untuk membuat proses lebih efisien dan terstruktur. UML yang digunakan mencakup 4 jenis diagram, yaitu *Use Case Diagram*, *Activity Diagram*, *Class Diagram*, dan *Sequence Diagram*. Perancangan aplikasi e-JO berbasis *website* berhasil dibangun dengan menggunakan UML, dengan melibatkan 7 aktor yaitu : Admin, *User*, *Dept. Head*, *Plant Head*, *Factory Head*, *Engineering Dept. Head*, dan *Member*.

Kata Kunci: *Job Order, UML, Aplikasi e-JO*

ABSTRACT

In the current job order process, paper is still used as the medium for creating job orders, leading to difficulties in document retrieval and archiving. Additionally, there is no adequate system for monitoring job orders, and the submission process is notably time-consuming. To address these issues, this study has designed a web-based e-JO system using UML (Unified Modeling Language) as a visual modeling tool to facilitate user understanding and interaction with the system. The UML design was carried out as an initial step before application development commenced. The system aims to enhance the efficiency and structure of the job order process. The UML employed includes four types of diagrams: Use Case Diagram, Activity Diagram, Class Diagram, and Sequence Diagram. The web-based e-JO application was successfully developed using UML and involves seven actors: Admin, User, Dept. Head, Plant Head, Factory Head, Engineering Dept. Head, and Member.

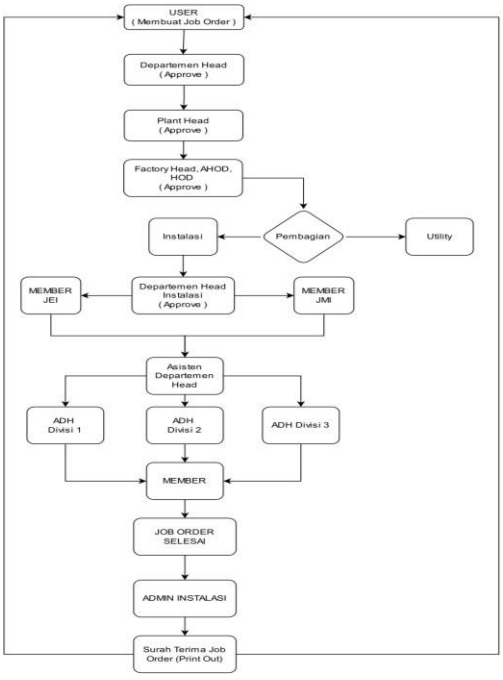
Key Words: *Job Order, UML, e-JO Application*

p-ISSN : 2746-7635
e-ISSN : 2808-5027

I. PENDAHULUAN

PT. ANDS adalah perusahaan yang bergerak di bidang industri manufaktur ban (tire). Dalam menjalankan proses produksinya, perusahaan ini memiliki beberapa departemen yang mendukung kelancaran operasional. Salah satu departemen yang dimiliki oleh perusahaan ini adalah Departemen Instalasi, yang bertugas khusus untuk memastikan kelancaran operasional perusahaan serta kualitas produk yang dihasilkan. Departemen Instalasi ini terbagi menjadi dua sub-departemen, yaitu *Joint Electric Installation* (JEI) dan *Joint Mechanic Installation* (JMI).

Dalam ruang lingkup kerjanya, departemen ini bertanggung jawab untuk melakukan proses perbaikan, pemasangan mesin baru, modifikasi mesin, dan pemindahan mesin sesuai dengan kebutuhan perusahaan. Untuk menjalankan tugasnya, Departemen Instalasi menerima formulir dari *user* yang berisi rincian tentang pengajuan *job order*, yang disebut formulir *Job Order*. Sebelum sampai ke pihak instalasi, formulir *Job Order* ini harus melalui beberapa tahapan. Berikut ini adalah tahapan formulir *Job Order*:



Gambar 1. Flow Chart Pengajuan Formulir Job Order

Gambar 1. menunjukkan adanya tahapan yang cukup panjang dan kompleks sebelum *job order* tersebut sampai ke pihak *member* atau pelaksana. Proses ini dimulai dengan pembuatan *job order* oleh *user*, kemudian harus melalui serangkaian persetujuan dari para pimpinan secara berurutan. Setiap tahap persetujuan ini memastikan bahwa *job order* yang dibuat memenuhi semua persyaratan dan standar yang telah ditetapkan. Setelah semua pimpinan memberikan persetujuan, *job order* tersebut dapat dilanjutkan ke tahap pelaksanaan oleh *member* atau pelaksana yang ditugaskan. *Member* yang ditugaskan membuat laporan harian sebagai arsip dari tugas yang telah dilaksanakan. Setelah pekerjaan selesai, *member* menyerahkan *form job order* kepada admin instalasi untuk dibuatkan STJO, kemudian diberikan kepada *user* sebagai bukti bahwa job order telah diselesaikan.

Tabel 1. Jumlah Pengajuan Job Order Bulan Januari – Desember Tahun 2023

No	Bulan	Jumlah JO
1	JANUARI	42
2	FEBRUARI	64
3	MARET	67
4	APRIL	50
5	MEI	43
6	JUNI	52
7	JULI	50
8	AGUSTUS	88
9	SEPTEMBER	70
10	OKTOBER	60
11	NOVEMBER	65
12	DESEMBER	79
	JUMLAH JO	730

Formulir *Job Order* yang digunakan saat ini masih menggunakan kertas. Berdasarkan Tabel
1. hasil observasi, jumlah pengajuan *job order* dari Januari hingga Desember 2023 mencapai 730 Job Order. Setiap Job Order membutuhkan lima lembar kertas, sehingga total kertas yang diperlukan dalam satu tahun adalah 3.650 lembar. Penggunaan formulir *job order* ini menyebabkan konsumsi kertas yang berlebihan, yang berpotensi meningkatkan jumlah sampah kertas. Berdasarkan permasalahan ini, dilakukan analisis menggunakan diagram fishbone untuk mengidentifikasi dan mencari akar penyebab dari masalah yang terjadi.

Gambar 2. *Fishbone Diagram*

Setelah dilakukan analisis dengan menggunakan diagram *fishbone* pada Gambar 2.

p-ISSN : 2746-7635
e-ISSN : 2808-5027

ditemukan beberapa masalah dalam proses dilakukan menggunakan media kertas, yang menyebabkan kesulitan dalam mencari formulir *job order* di arsip penyimpanan. Masalah berikutnya adalah proses pengantaran formulir *job order* yang diajukan memakan waktu cukup lama dan juga menyulitkan proses *monitoring.*

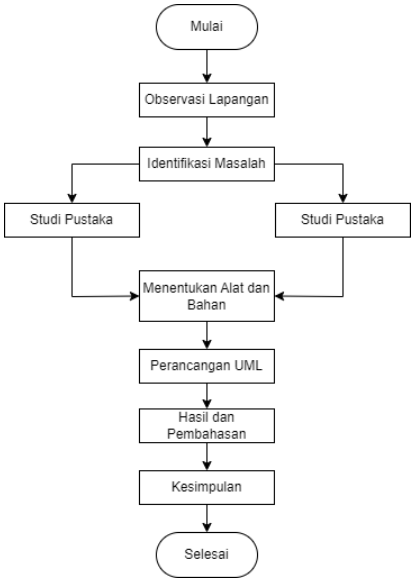
UML yang akan digunakan mencakup empat jenis diagram, yaitu *Use Case Diagram*, *Activity Diagram*, *Class Diagram*, dan *Sequence Diagram*. *Use Case Diagram* menggambarkan interaksi antara pengguna dan fungsi dari sistem. *Activity Diagram* menunjukkan aksi yang terstruktur dalam suatu sistem. *Class Diagram* menampilkan hubungan antar class, termasuk atribut dan fungsi dari suatu objek. *Sequence Diagram* menggambarkan urutan pesan yang dikirim antar objek untuk menyelesaikan suatu tugas atau proses. *order*.

II. METODE PENELITIAN

A. Alur Penelitian

Alur penelitian berisi berisikan tentang beberapa tahapan. Adapun alur penelitian dapat dilihat pada Gambar 3:

lebih terstruktur. Dengan adanya



Gambar 3. Alur Penelitian

B. Observasi Lapangan

Observasi adalah aktivitas pengumpulan data yang melibatkan wawancara dengan pengguna dan pengamatan langsung oleh peneliti. Wawancara dengan pengguna bertujuan untuk memahami proses pengajuan *job order* yang sedang berlangsung. Observasi ini bertujuan untuk mendapatkan gambaran tentang masalah yang terjadi di lapangan, sehingga hasil pengamatan dapat dianalisis dan dikembangkan menjadi rumusan masalah.

pengajuan dan *monitoring job order*. perancangan ini diharapkan dapat memudahkan *user* dalam melakukan proses pengajuan dan *monitoring job*

Use case diagram adalah sebuah cara untuk memodelkan bagaimana sistem informasi yang akan dibuat berperilaku. Ini melibatkan penggambaran berbagai interaksi yang mungkin terjadi antara pengguna dan sistem tersebut, serta menjelaskan langkah-langkah yang diambil oleh sistem dalam menanggapi setiap tindakan pengguna. *Activity diagram* adalah aktivitas dari sebuah sistem atau menu yang terdapat pada perangkat lunak [1].

Class diagram adalah diagram yang menampilkan hubungan antar *class*, termasuk atribut dan fungsi dari suatu objek. *Sequence diagram* adalah diagram yang menunjukkan bagaimana objek-objek berinteraksi satu sama lain dalam urutan waktu tertentu [2]. Hal ini disebabkan oleh tidak adanya sistem yang mengintegrasikan proses tersebut.

Oleh karena itu, untuk menyelesaikan masalah terkait proses pengajuan dan monitoring *job order*, maka dilakukan perancangan aplikasi

C. Identifikasi Masalah

Pada tahap ini, dilakukan identifikasi masalah yang terjadi di departemen instalasi. Dalam proses tersebut, terungkap bahwa salah satu permasalahan yang dihadapi yaitu terkait dengan penggunaan formulir *job order* yang masih menggunakan kertas, sehingga proses pengajuan memakan waktu yang cukup lama. Selain itu, *user* juga tidak dapat memantau status *job order* yang telah diajukan.

D. Studi Pustaka

Pada tahap ini, bertujuan untuk menganalisis kemajuan yang telah dicapai dalam proses perancangan UML dan pengembangan aplikasi yang menggunakan *website*. Selain itu, dilakukannya analisis mendalam terhadap riset sebelumnya yang relevan dengan topik yang akan dibahas. Tujuan dari analisis ini adalah untuk menyediakan landasan yang kuat dan berbasis bukti untuk mendukung tahapan selanjutnya dalam pengembangan aplikasi e-JO.

E. Studi Lapangan

Pada tahap ini, dilakukan observasi pengumpulan data dan informasi langsung dari lapangan. Tujuan dari studi lapangan ini untuk mendapatkan data yang diperlukan untuk melakukan penelitian.

F. Menentukan Alat dan Bahan

Pada tahap ini, dilakukan penentuan alat dan bahan yang digunakan untuk merancang aplikasi. Adapun alat yang

p-ISSN : 2746-7635
e-ISSN : 2808-5027

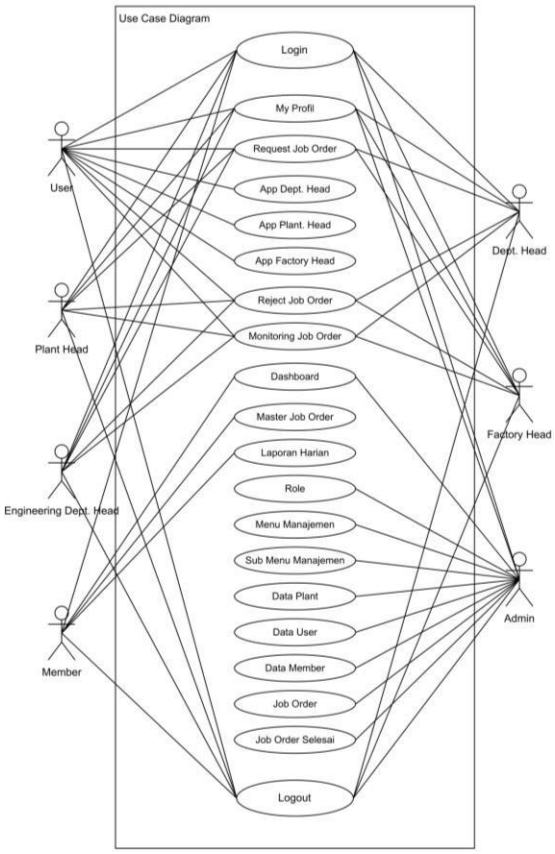
diperlukan yaitu sebuah laptop serta dukungan dari *software* Draw.io untuk membuat UML.

G. Perancangan UML

Pada tahap ini, dilakukan perancangan UML (*Unified Modeling Language*) sebagai langkah awal sebelum pengembangan aplikasi dimulai. Proses perancangan UML ini mencakup pembuatan berbagai diagram yang akan membantu dalam visualisasi struktur dan perilaku sistem yang akan dikembangkan. UML yang akan digunakan mencakup empat jenis diagram, yaitu *Use Case Diagram*, *Activity Diagram*, *Class Diagram*, dan *Sequence Diagram*.

III. HASIL DAN PEMBAHASAN

1. Use Case Diagram



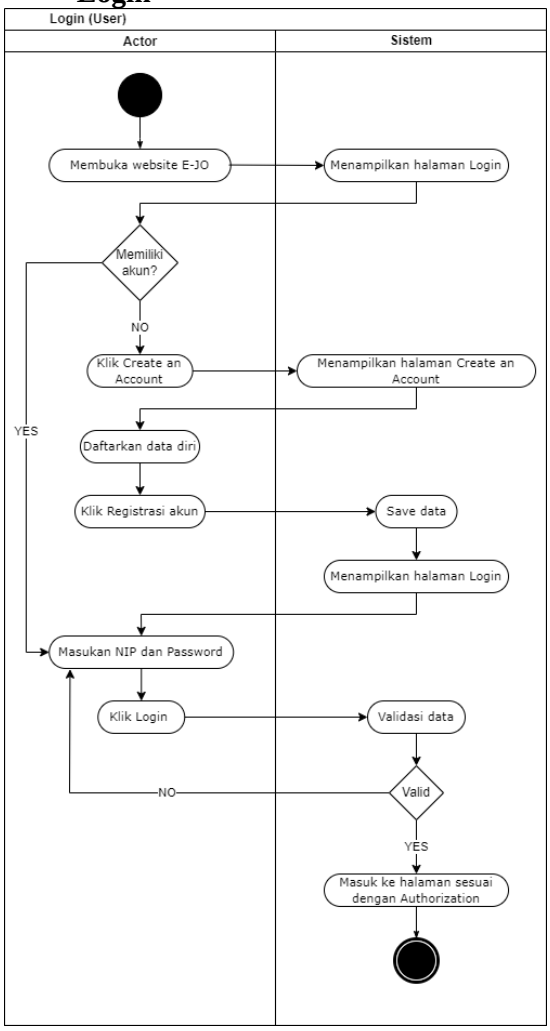
Gambar 4. Use Case Diagram

Use case diagram adalah diagram yang menggambarkan hubungan antara aktor dan *use case*. Diagram ini digunakan untuk analisis dan perancangan suatu sistem, membantu dalam memahami interaksi antara pengguna dan fungsi- fungsi yang disediakan oleh sistem.

Pada *use case diagram*, setiap aktor memiliki akses ke menu yang sesuai dengan garis yang telah digambarkan pada diagram tersebut. Garis-garis ini menghubungkan aktor dengan menu yang berisi fungsi-fungsi yang dapat dijalankan oleh aktor tersebut. Dengan kata lain, setiap garis menunjukkan hubungan antara aktor dan fungsi- fungsi spesifik yang tersedia dalam menu yang mereka akses. Hal ini membantu dalam menggambarkan dengan jelas interaksi antara pengguna (aktor) dan sistem, serta mempermudah dalam memahami fungsionalitas yang dimiliki oleh masing-masing aktor.

2. Activity Diagram

- **Login**



Gambar 5. Activity Diagram Login

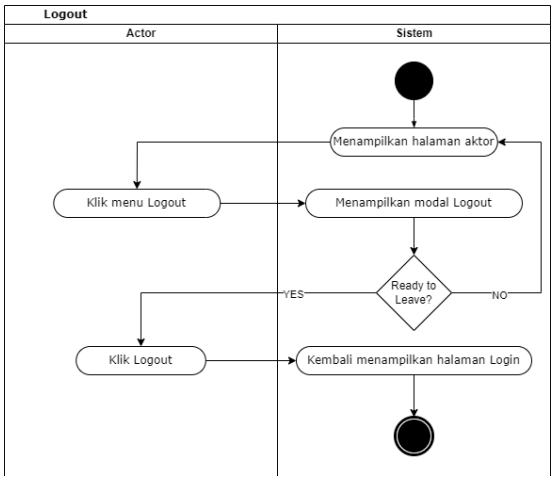
Berdasarkan *activity diagram* pada gambar diatas, actor dapat melakukan *login* dengan cara melakukan registrasi terlebih dahulu. Registrasi dilakukan dengan cara klik tombol "*Create an Account*", kemudian sistem akan menampilkan halaman registrasi. Aktor dapat mendaftarkan diri, lalu klik "Registrasi Akun" dan sistem akan menyimpan data aktor. Aktor dapat melakukan *login* dengan memasukan NIP dan *password*. Klik "*Login*" kemudian sistem akan menerima NIP dan *password*, memproses dan melakukan validasi pada sistem. Apabila data valid maka aktor akan masuk ke halaman sesuai otorisasinya. Namun, apabila data tidak tervalidasi maka sistem akan menampilkan pesan

p-ISSN : 2746-7635
e-ISSN : 2808-5027

kesalahan dan mengisi kembali NIP dan password benar.

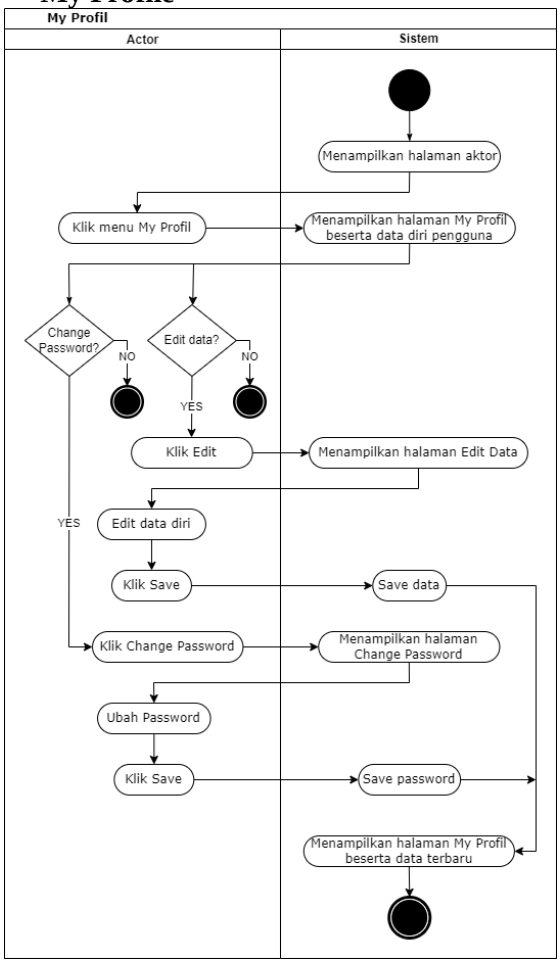
- **Logout**

Berdasarkan gambar activity diagram diatas, apabila aktor ingin logout dari aplikasi, aktor dapat mengklik menu logout. Sistem akan menampilkan modal logout berupa alert, kemudian aktor mengklik "Logout". Setelah itu sistem akan menampilkan halaman login.



Gambar 6. Activity Diagram Logout

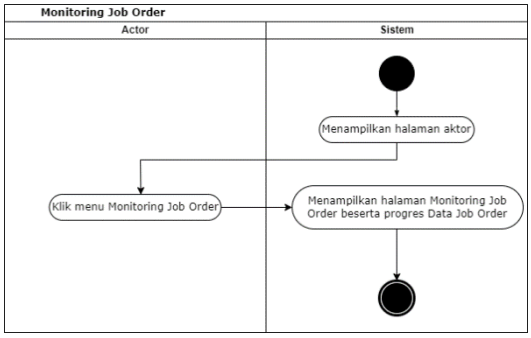
- **My Profile**



Gambar 7. Activity Diagram My Profile

Berdasarkan *activity diagram* pada gambar diatas, aktor masuk ke halaman *my profile*. Apabila aktor ingin mengubah *password*, aktor dapat mengklik tombol "*Change Password*". Kemudian sistem akan mengarahkan ke halaman change *password*. Pada halaman ini, aktor dapat mengubah *password* sesuai yang diinginkan. Jika sudah mengubah *password*, aktor dapat mengklik tombol "*Save*". Sistem akan menyimpan data tersebut dan kembali ke halaman *my profile*. Dan apabila aktor ingin mengubah biodata, aktor dapat mengklik tombol "Edit". Sistem akan mengarahkan ke halaman edit, kemudian aktor dapat mengubah biodata sesuai keinginan aktor. Klik tombol "*Save*" dan sistem akan menyimpan biodata serta menampilkan halaman *my profile* dengan menampilkan data terbaru.

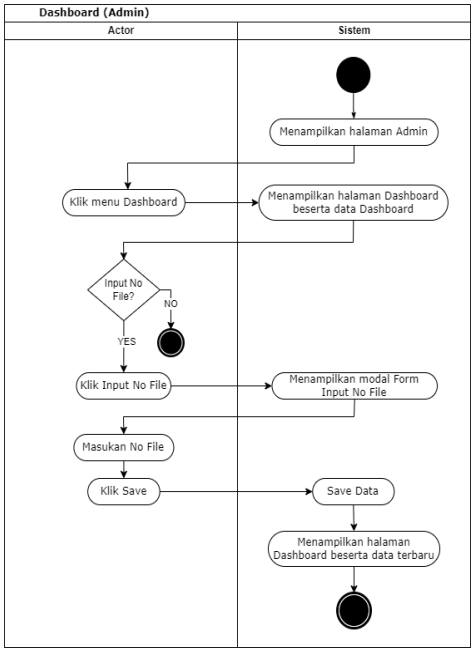
- **Monitoring Job Order**



Gambar 8. Activity Diagram Monitoring Job Order

Berdasarkan *activity diagram* pada gambar diatas, aktor masuk ke halaman

- **Dashboard Admin**

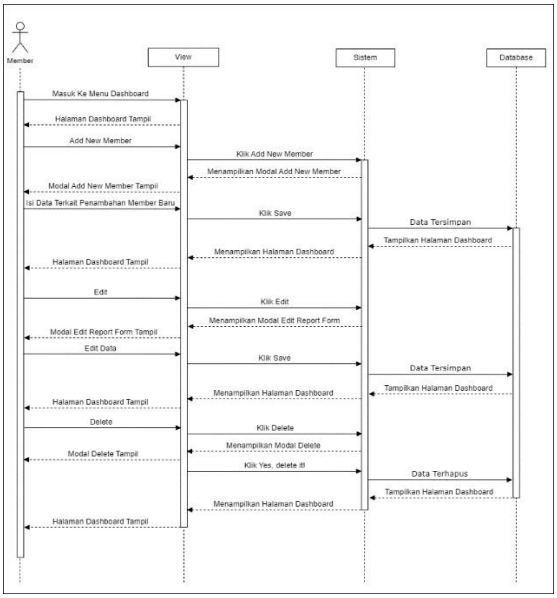


Gambar 9. Activity Diagram Dashboard

p-ISSN : 2746-7635
e-ISSN : 2808-5027

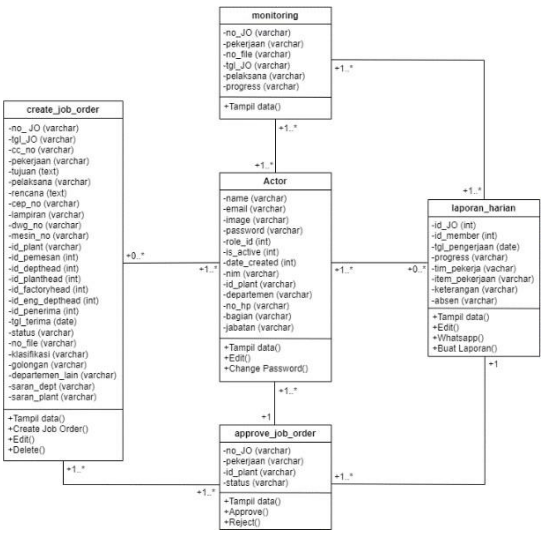
Berdasarkan gambar *activity diagram* diatas, admin masuk ke halaman *dashboard*,

- **Request Job Order User**



Gambar 10. Request Job Order User

kemudian data pada halaman *dashboard* tampil. Apabila admin ingin menginput nomor *file job order* yang sudah di *approve* oleh pimpinan, dapat dilakukan dengan cara klik tombol "*Input No File*", kemudian sistem menampilkan modal *form input no file*. Admin dapat mengisi *no file*, kemudian klik "*Save*". Setelah itu sistem akan menampilkan kembali halaman *dashboard*. dapat melihat *progress job order* yang telah dibuat oleh *user*. *monitoring job order*. Pada halaman ini aktor.

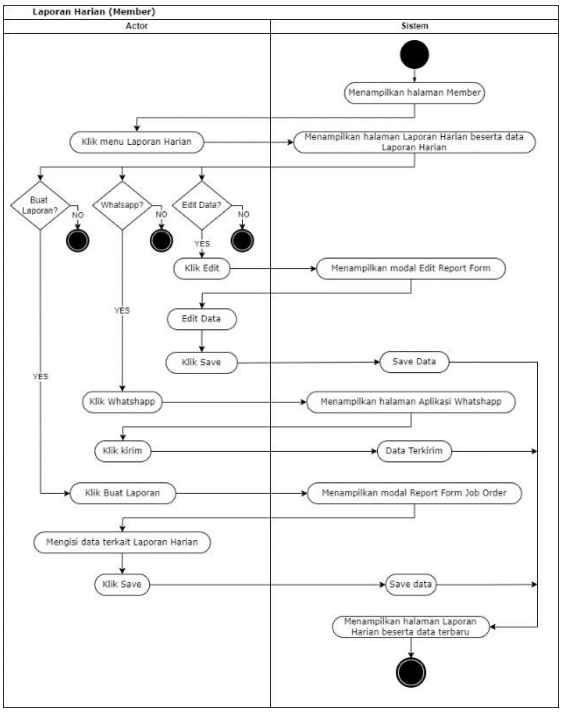


Gambar 11. Activity Diagram Request Job Order User

Berdasarkan *activity diagram* pada gambar diatas, *user* masuk ke halaman *request job order*. Pada halaman *request job order*, apabila *user* ingin membuat *job order*, *user* dapat mengklik tombol "*Create Job Order*". Kemudian, sistem akan menampilkan modal *create job order*, lalu *user* dapat mengisi data terkait permintaan *job order* yang ingin diajukan. Setelah itu, *user* mengklik tombol "*Save*", sehingga sistem akan menyimpan data tersebut dan menampilkan halaman *request job order* dengan data terbaru. Jika *user* ingin mengedit data, *user* dapat mengklik tombol "Edit". Sistem akan menampilkan modal edit, kemudian *user* dapat mengedit data. Setelah itu, *user* mengklik tombol "*Save*", sehingga sistem akan menyimpan data tersebut dan menampilkan halaman r*equest job order* dengan data terbaru. Jika *user* ingin menghapus salah satu data pada halaman *request* halaman *request job order.*

- **Request Job Order Pimpinan**

Gambar 11. Activity Diagram Request Job Order

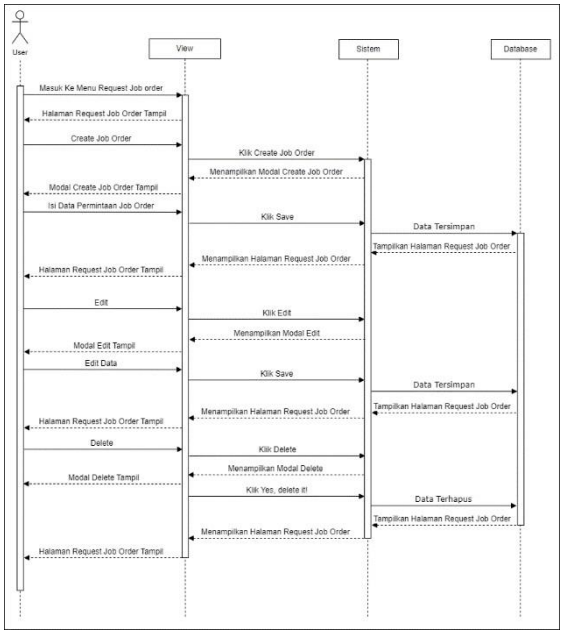


Pimpinan berdasarkan *activity diagram* pada gambar diatas, pimpinan masuk ke halaman *request job order*. Apabila pimpinan setuju pada *job order* yang diajukan oleh *user*, pimpinan dapat mengklik tombol "*Approve*" pada *job oder* yang akan di *approve*. Sistem akan menampilkan modal *form* JO *approve*, kemudian klik "*Yes, approve it*!". Sistem akan menyimpan data tersebut dan menampilkan halaman *request job order*. Jika Pimpinan tidak setuju pada *job order* yang sudah diajukan oleh *user*, Pimpinan dapat mengklik tombol "*Reject*"

p-ISSN : 2746-7635
e-ISSN : 2808-5027

pada job order yang akan di *reject*. Sistem akan menampilkan modal *reject*, kemudian pimpinan dapat memasukan saran pada *job order*. Klik "*Save*" dan sistem akan menyimpan data tersebut kemudian menampilkan halaman *request job order*.

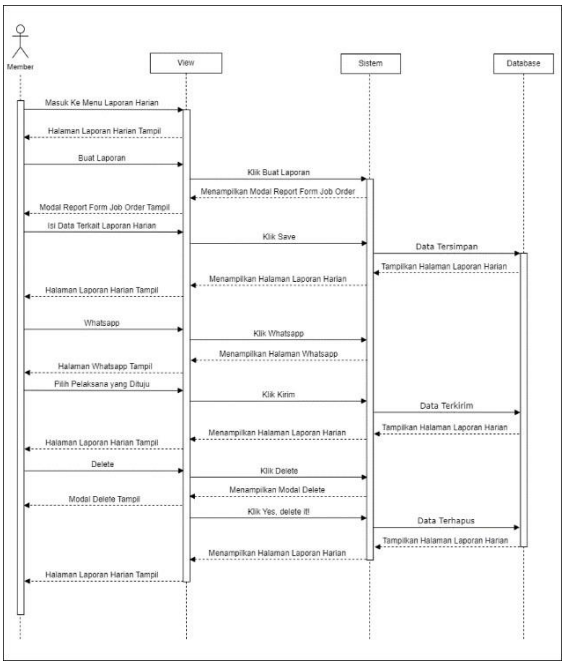
- **Dashboard Member**



Gambar 12. Activity Diagram Dashboard Member

Berdasarkan gambar *activity diagram* diatas, *member* masuk ke halaman *dashboard*, kemudian sistem akan menampilkan halaman *dashboard* beserta datanya. Apabila *member* ingin menambah anggota baru, *member* dapat mengklik tombol "*Add New Member*". Kemudian, sistem akan menampilkan modal *add new member*, lalu *member* mengisi data terkait penambahan anggota baru. Setelah itu, admin mengklik tombol "*Save*", sehingga sistem akan menyimpan data tersebut dan menampilkan halaman data *member* dengan data terbaru. Jika *member* ingin menghapus salah satu data anggota pada halaman data *member*, *member* dapat mengklik tombol "*Delete*", kemudian sistem akan menampilkan modal *delete* berupa *alert*. Klik tombol "*Yes, delete it*!", sehingga sistem akan menghapus data tersebut dan kembali ke halaman data *member*. *job order*, *user* dapat mengklik tombol "*Delete*", kemudian sistem akan menampilkan modal *delete* berupa *alert*. Klik tombol "*Yes, delete it*!", sehingga sistem akan menghapus data tersebut dan kembali ke

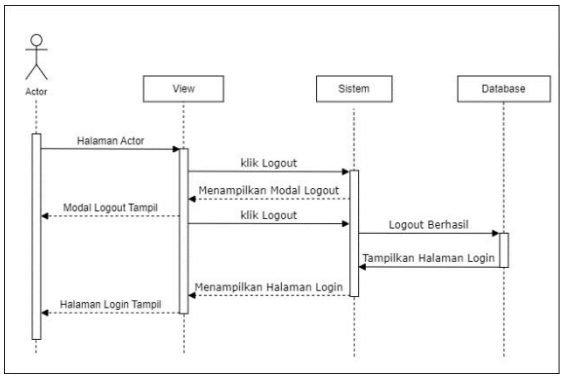
- **Master Job Order Member**



Gambar 13. Activity Diagram Master Job Order Member

Berdasarkan gambar *activity diagram* diatas, *member* masuk ke halaman *master job order*, kemudian sistem akan menampilkan halaman *master job order* berserta datanya. Pada halaman ini *member* membuat laporan terkait *job order* yang baru masuk. *Member* dapat mengklik tombol "Buat Laporan", kemudian sistem akan menampilkan modal *report form job order*. Setelah itu, *member* memasukan data terkait *job order* yang baru pertama kali dikerjakan. Klik "*Save*", kemudian sistem menyimpan data dan kembali menampilkan halaman *master job order*.

- **Laporan harian**

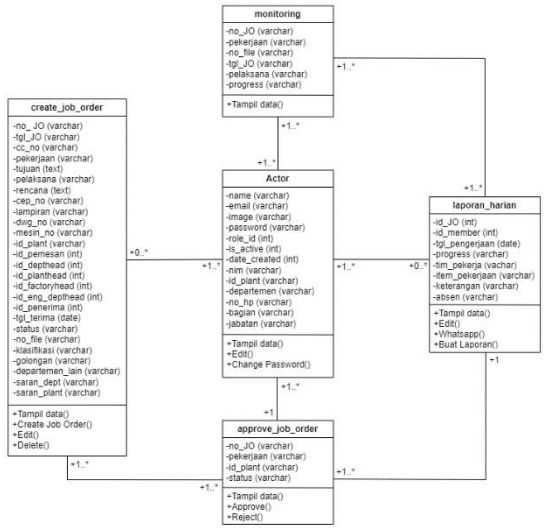


Gambar 14. Activity Diagram Laporan Harian

p-ISSN : 2746-7635
e-ISSN : 2808-5027

Berdasarkan gambar activity diagram diatas, *member* masuk ke halaman laporan harian. Pada halaman laporan harian, apabila *member* ingin membuat laporan harian, *member* dapat mengklik tombol "Laporan Harian". Kemudian, sistem akan menampilkan modal *report form job order*, lalu *member* dapat mengisi data terkait *job order* yang sedang dikerjakan sebagai laporan harian. Setelah itu, *member* mengklik tombol "*Save*", sehingga sistem akan menyimpan data tersebut dan menampilkan halaman laporan harian dengan data terbaru. Jika *member* ingin mengedit data, *member* dapat mengklik tombol "Edit". Sistem akan menampilkan modal edit *report form*, kemudian *member* dapat mengedit data. Setelah itu, *member* mengklik tombol "*Save*", sehingga sistem akan menyimpan data tersebut dan menampilkan halaman laporan harian. Apabila *member* ingin mengirim data *job order* melalui whatsapp, *member* dapat mengklik tombol "Whatsapp", kemudian sistem akan menampilkan halaman aplikasi whatsapp dan mengirim pesan tersebut, kemudian kembali ke halaman laporan harian.

- **Class Diagram**



Gambar 15. Class Diagram

Berdasarkan gambar *class diagram* atas, terdapat lima class utama yang terdiri dari *Actor, Create Job Order, Approve Job Order*, Laporan Harian, dan *Monitoring*. Penggunaan *class diagram* ini dalam sistem pengajuan *job order* melibatkan beberapa individu dalam berbagai tahapan prosesnya.

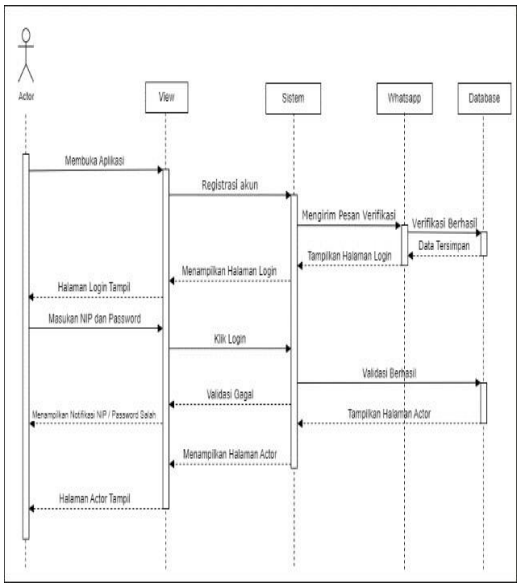
Pada class *Actor*, terdapat *user* yang bertugas membuat *job order*. Setelah *job order* dibuat, kemudian dikirimkan kepada pimpinan untuk mendapatkan persetujuan. *Class Approve Job Order* mencakup tahapan di mana *job order* harus disetujui oleh berbagai pimpinan, termasuk *Dept. Head, Plant Head, Factory Head, dan Engineering Dept. Head.* Persetujuan dari semua pimpinan yang relevan diperlukan sebelum *job order* dapat dilanjutkan ke tahap berikutnya.

Setelah *job order* disetujui, dkemudian dikirimkan kepada *member* yang bertugas untuk melaksanakan pekerjaan yang tertera dalam *job order*. *Member* bertanggung jawab untuk menyelesaikan pekerjaan sesuai dengan instruksi yang diberikan dalam *job order*. Selain itu, *member* juga bertugas untuk membuat laporan harian yang merinci progres pekerjaan yang telah dilakukan yang terdapat dalam *class* laporan harian.

Class monitoring memiliki fungsi penting dalam sistem ini. *Class* ini berperan sebagai monitoring terhadap *job order* yang sedang dilaksanakan. Laporan yang dibuat oleh *member* akan dimasukkan ke dalam sistem monitoring, yang memungkinkan *user* dan pimpinan untuk memantau status dan kemajuan j*ob order*. Dengan adanya fitur monitoring ini, baik pengguna maupun pimpinan dapatmemastikan bahwa *job order* telah dilaksanakan sesuai dengan rencana dan dapat mengetahui kapan pekerjaan tersebut telah selesai. Secara keseluruhan, *class diagram* ini menggambarkan alur proses yang terstruktur dalam pengajuan, persetujuan, pelaksanaan, pelaporan, dan pemantauan *job order* dalam suatu sistem yang selesai. Secara keseluruhan, *class diagram* ini menggambarkan alur proses yang terstruktur dalam pengajuan, persetujuan, pelaksanaan, pelaporan, dan pemantauan *job order* dalam suatu sistem

3. Sequence Diagram

- **Login**



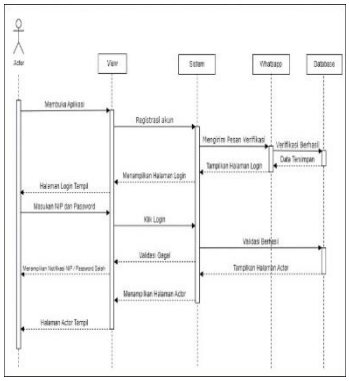
Gambar 16. *Sequence Diagram Login*

p-ISSN : 2746-7635
e-ISSN : 2808-5027

melibatkan banyak pihak. Setiap *class* memiliki peran dan fungsinya masing-masing yang saling terkait untuk memastikan kelancaran dalam pengelolaan *job order*.

4. Sequence Diagram

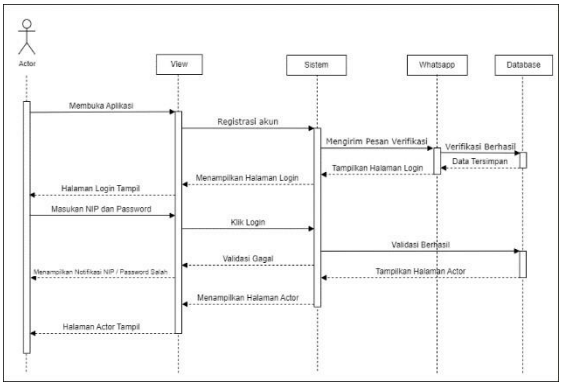
- **Login**



Gambar 16. *Sequence Diagram Login*

Pada gambar *sequence diagram* diatas, pertama-tama aktor akan masuk ke halaman *login*. Apabila aktor tidak memiliki akun, aktor diharuskan registrasi terlebih dahulu. Sistem akan memproses dan mengirim pesan verifikasi berhasil melalui whatsapp. Jika regitrasi berhasil, sistem akan menampilkan kembali halaman *login*. Kemudian aktor dapat memasukan NIP dan *Password*. Sistem akan mengirim data tersebut untuk divalidasi. Apabila data yang dimasukan salah, maka akan menampilkan pesan gagal atau tidak valid. Jika data yang dimasukan benar kemudian *database* akan menyimpan dan sistem akan menampilkan halaman aktor.

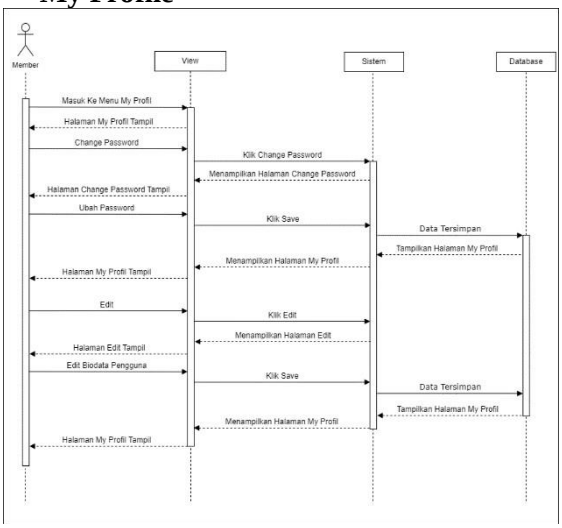
- **Logout**



Gambar 17. *Sequence Diagram Logout*

Berdasarkan gambar *sequence diagram* diatas, aktor dapat keluar dari aplikasi dengan menekan tombol *logout*. Kemudian sistem akan memproses dan *database* menyimpan lalu sistem mengarahkan ke halaman *login* yang menandakan *logout* berhasil.

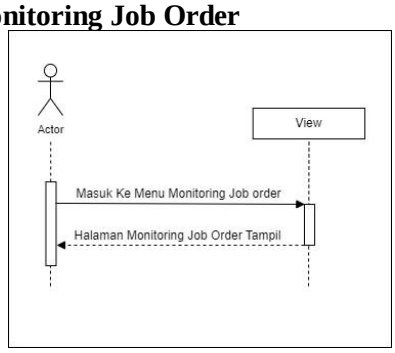
- **My Profile**



Gambar 18. *Sequence Diagram My Profile*

Berdasarkan gambar *sequence diagram* di atas, aktor masuk ke halaman *my profile*. Pada halaman ini, aktor dapat mengedit biodata. Setelah melakukan perubahan, sistem akan memproses dan menyimpan data yang telah diperbarui ke *database*. Data berhasil tersimpan, sistem akan menampilkan kembali halaman *my profile*. Jika aktor ingin mengubah kata sandi, aktor dapat memasukkan kata sandi lama dan kata sandi baru. Sistem kemudian akan memproses dan menyimpan perubahan kata sandi baru ke *database*. Data berhasil tersimpan, sistem akan menampilkan kembali halaman *my profil*e.

- **Monitoring Job Order**

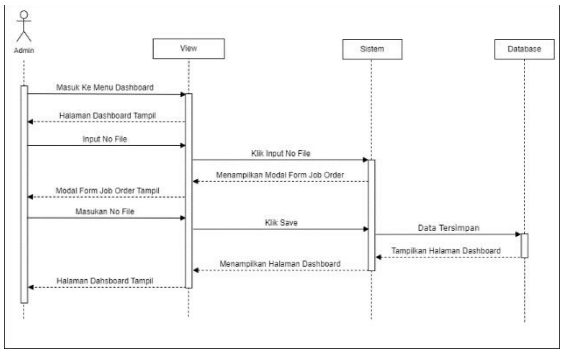


Gambar 19. *Sequence Diagram Monitoring Job Order*

Berdasarkan *sequence diagram* diatas, Aktor masuk ke halaman monitoring *job order*. Pada halaman ini, aktor apat melihat proses data *job order* yang telah diajukan oleh *user*.

p-ISSN : 2746-7635
e-ISSN : 2808-5027

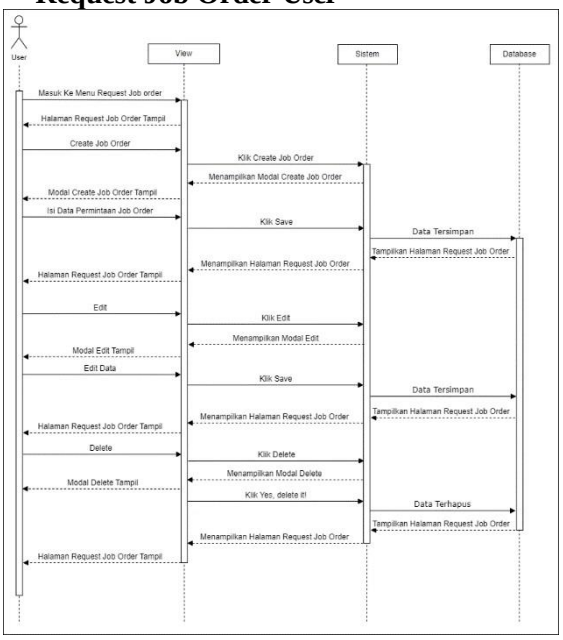
- **Dashborad Admin**



Gambar 20. *Sequence Diagram*

Dashnoard admin berdasarkan gambar squeence diagram diatas, admin masuk ke halaman *dashboard*. Pada halaman ini admin dapat menginput nomor *file*, dengan memasukan nomor *file* pada *job order* yang tersedia. Kemudian, sistem akan memproses dan *database* akan menyimpan data tersebut. Data berhasil tersimpan, sistem akan menampilkan halaman *dashboard* kembali.

- **Request Job Order User**



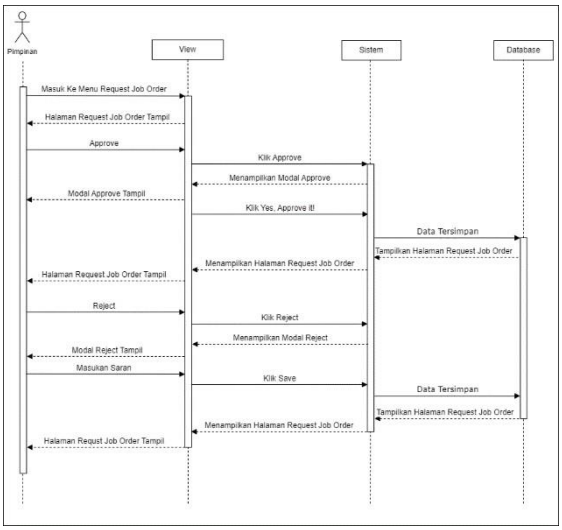
Gambar 21. *Sequence Diagram Request Job Order User*

Berdasarkan gambar *sequence diagram* di atas, *user* masuk ke halaman *request job order*. Pada halaman ini, *user* dapat *create job order*, kemudian sistem akan memproses dan menyimpan data tersebut ke *database*. Data berhasil tersimpan, sistem akan menampilkan halaman *request job order* beserta data terbarunya.

User juga dapat mengedit data. Setelah melakukan perubahan, sistem akan memproses dan menyimpan data yang telah diperbarui ke *database*. Data berhasil tersimpan, sistem akan menampilkan kembali ke halaman *request job order*. Apabila *user* ingin menghapus salah satu data *job order*, *user* dapat memilih dan menghapus data yang diinginkan. Sistem kemudian akan memproses dan menghapus data *plant* tersebut dari *database*. Jika penghapusan berhasil, sistem akan menampilkan kembali halaman *request job order*.

- **Request Job Order Pimpinan**

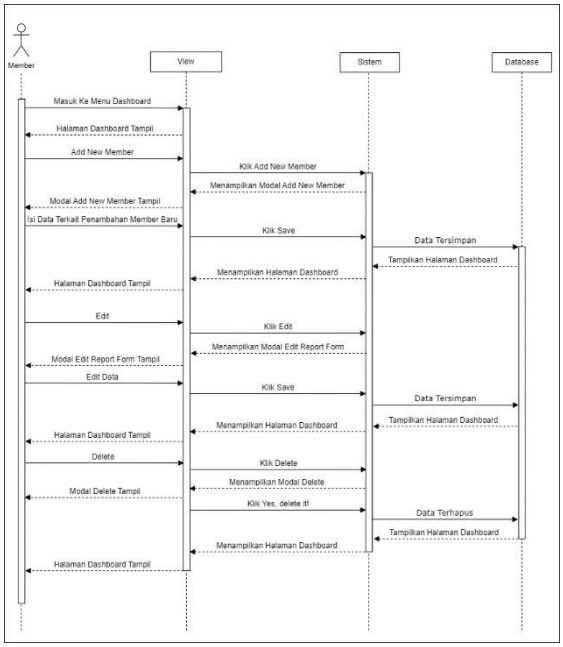
Berdasarkan gambar *sequence diagram* di atas, pimpinan masuk ke halaman *request job order*. Pada halaman ini, pimpinan dapat *approve job order* apabila pimpinan setuju. Kemudian sistem akan memproses dan database menyimpan. Data berhasil tersimpan, sistem akan menampilkan kembali halaman *request job order*. Pimpinan juga dapat *reject job order* apabila pimpinan tidak setuju. Kemudian sistem akan memproses dan database menyimpan. Data berhasil tersimpan, sistem akan menampilkan kembali halaman *request job order*.



Gambar 22. *Sequence Diagram Request Job Order*

p-ISSN : 2746-7635
e-ISSN : 2808-5027

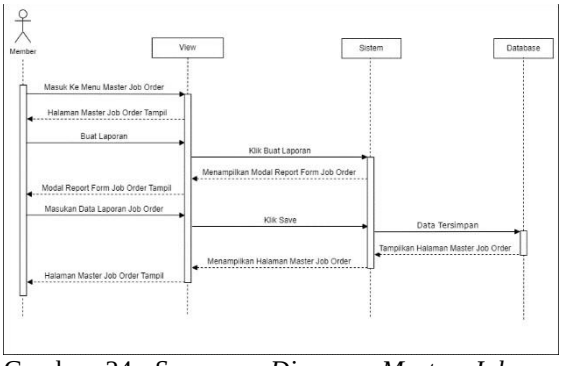
- **Dashboard Member.**



Gambar 23. *Sequence Diagram Dashboard Member*

Berdasarkan gambar *sequence diagram* diatas, *member* masuk ke halaman *dashboard.* Pada halaman ini, *member* dapat menambahkan anggota baru. Sistem akan memproses dan menyimpan data tersebut di *database*. Data berhasil tersimpan, sistem akan menampilkan kembali halaman *dashboard* beserta data terbarunya. *Member* juga dapat mengedit data anggotanya. Setelah melakukan perubahan, sistem akan memproses dan menyimpan data yang telah diperbarui ke *database*. Data berhasil tersimpan, sistem akan menampilkan kembali ke halaman *dashboard*. Apabila *member* ingin menghapus salah satu anggota, *member* dapat memilih dan menghapus anggotanya. Kemudian sistem akan memproses dan menghapus anggota tersebut dari *database*. Jika penghapusan berhasil, sistem akan menampilkan kembali halaman *dashboard.*

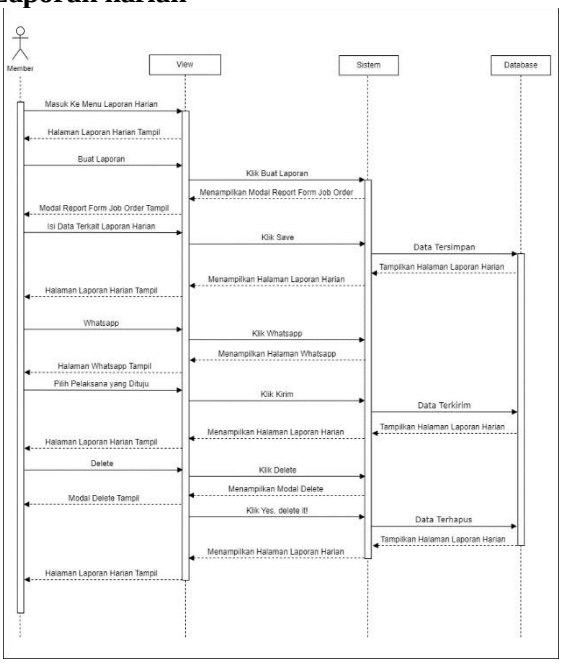
- **Master Job Order Member**



Gambar 24. *Sequence Diagram Master Job Order Member*

Berdasarkan gambar *sequence diagram* diatas, *member* masuk ke halaman master *job order*. Pada halaman ini, *member* dapat membuat laporan terkait *job order* yang baru pertama kali masuk. Sistem kemudian memproses dan *database* menyimpan data tersebut. Data berhasil tersimpan, sistem akan menampilkan kembali halaman *dashboard*.

- **Laporan harian**



Gambar 25. *Sequence Diagram*

Laporan Harian Berdasarkan gambar *activity diagram* diatas, *member* masuk ke halaman laporan harian. Pada halaman laporan harian, *member* dapat membuat laporan harian dengan mengisi data terkait *job order* yang sedang dikerjakan. Kemudian sistem akan memproses dan database menyimpan data tersebut lalu menampilkan halaman laporan harian dengan data terbaru. *Member* juga dapat mengedit data. Setelah melakukan perubahan, sistem akan memproses dan menyimpan data yang telah diperbarui ke *database*. Data berhasil tersimpan, sistem akan menampilkan kembali ke halaman laporan harian. *Member* dapat mengirim data *job order* melalui whatsapp, kemudian sistem akan mengarahkan aplikasi whatsapp dan *database* menyimpan. Data berhasil terkirim, sistem akan menampilkan kembali ke laporan harian.

IV. KESIMPULAN

Perancangan aplikasi e-JO berbasis *website* berhasil dibangun dengan menggunakan UML, dengan melibatkan 7 aktor yaitu : Admin, *User*, *Dept. Head*, *Plant Head*, *Factory Head*, *Engineering Dept*. *Head*.

DAFTAR PUSTAKA

p-ISSN : 2746-7635
e-ISSN : 2808-5027

[1] I. Nurlita, R. Anggraini, S. I. Akuntansi, and U. Gunadarma, "Analysis and Design of Incoming and Outgoing Cash Accounting Information Systems at Kilometer 28 Laundry using the Pieces and Waterfall Methods with Unified Modeling Language (UML) Tools Analisis dan Perancangan Sistem Informasi Akuntansi Kas Masuk dan Keluar pada Laundry Kilometer 28 menggunakan Metode Pieces dan Waterfall dengan Unified Modelling Languange (UML) Tools," vol. 2, no. 6, pp. 1065–1090, 2023.

[2] M. Nazir, S. F. Putri, and D. Malik, "Perancangan Aplikasi E-VOTING Menggunakan Diagram UML (Unified Modelling Language)," *J. Ilm. Komput. Terap. dan Inf.*, vol. 1, no. 1, pp. 5–9, 2022, [Online]. Available: http://journal.polita.ac.id/index.php/politati/article/view/99