Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer e-ISSN: 2548-964X
Vol. 3, No. 8, Agustus 2019, hlm. 7616-7625 http://j-ptiik.ub.ac.id

Pengembangan Aplikasi Pemantauan Alat Berat Pertambangan menggunakan Teknologi *Geofencing* dengan Arsitektur *MVP*

Arif Luthfiansyah[1], Denny Sagita Rusdianto[2], Agi Putra Kharisma[3]

Program Studi Teknik Informatika, Fakultas Ilmu Komputer, Universitas Brawijaya
Email: [1]lutluthfi@student.ub.ac.id, [2]denny.sagita@ub.ac.id, [3]agi@ub.ac.id

Abstrak

MyVeego adalah sebuah produk berupa sistem yang menawarkan layanan berbasis lokasi untuk membantu bisnis dalam mengelola armada dan aset yang dimiliki oleh perusahaan dengan visibilitas lokasi, status, diagnostik, dan analisis mendalam secara *real-time* dari perusahaan PT. Astra Graphia Information Technology. Pertambangan memiliki area yang luas dan membuat jarak antara 2 (dua) area yang menjadi fokus berjauhan dan sulit dalam melakukan pemantauan dan mengelola pada saat melakukan aktivitas-aktivitas penambangan. Namun, MyVeego belum memiliki sebuah layanan atau modul yang dapat melakukan pemantauan visibilitas lokasi, waktu, dan status alat berat pertambangan secara *real-time*. Dengan memadukan teknologi *Geofencing*, *WebSocket*, dan dibangun pada arsitektur *Model-View-Presenter*, memungkinkan suatu sistem dapat mengetahui pergerakan masuk, keluar, dan tinggal suatu objek terhadap suatu area yang telah ditentukan dengan menampilkan visibilitas lokasi secara *real-time*. Rekayasa kebutuhan dilakukan sebagai tahap awal penelitian untuk mencari tahu proses bisnis dan kebutuhan sistem. Tahap rekayasa kebutuhan memberikan hasil 19 (sembilan belas) kebutuhan fungsional dan 1 (satu) kebutuhan non-fungsional. Kebutuhan tersebut digunakan sebagai dasar dalam melakukan tahap perancangan dan implementasi. Implementasi sistem dilakukan dengan memadukan teknologi GPS sebagai dasar sebuah sistem berbasis lokasi, teknologi *Geofencing*, dan teknologi *WebSocket* pada arsitektur *Model-View-Presenter* dengan bahasa pemrograman Kotlin di atas platform Android. Selanjutnya dari tahap implementasi dilakukan tahap pengujian dengan hasil *complexity number* 5.3 untuk pengujian unit, *complexitiy number* < 10 untuk pengujian integrasi. Tahap pengujian memberikan hasil validitas sebesar 100% di semua pengujian.

Kata kunci: *pemantauan, pertambangan, geofencing, model-view-presenter, websocket, real-time*

Abstract

MyVeego is product as system that offers location-based services to help bussiness to manage fleets and assets owned by the company with location visibility, status, diagnostics, and real-time in-depth analysis owned by PT. Astra Graphia Information Technology. Mining has a large area and makes a distance between 2 (two) areas that are focussed far apart and difficult to monitor and manage when conducting mining activities. However, MyVeego does not have a service or module that can monitor the visibility of the location, time, and status of mining vehicles in real-time. By integrating Geofencing technology, WebSocket, and built on the Model-View-Presenter architecture, it allows a system can know the movement of entering, exiting, and dwelling objects to a specified area by displaying location visibility in real-time. Requirements engineering is carried out as the initial phase of research to find out business processs and system requirements. The requirements engineering phase results in 19 (nineteen) functional requirements and 1 (one) non-functional requirements. These requirements are used as a basis in to conducting the design and implementation phases. System implementation is done by integrating GPS technology as the basis of a location-based system, Geofencing technology, and WebSocket technology on the Model-View-Presenter architecture with the Kotlin programming language on the Android platform. Furthermore, from the implementation phase, the testing phase is carried out the result of complexity number 5.3 for unit testing, complexity number < 10 for integration testing. The testing phase gives 100% validity results in all tests.

Keywords: *monitoring, mining, geofencing, model-view-presenter, websocket, real-time*

1. PENDAHULUAN

MyVeego adalah sebuah produk berupa sistem yang menawarkan layanan berbasis lokasi untuk membantu bisnis dalam mengelola armada dan aset yang dimiliki oleh perusahaan dengan visibilitas lokasi, status, diagnostik, dan analisis mendalam secara *real-time* dari perusahaan PT. Astra Graphia Information Technology. Dalam pengembangannya, Myveego menerapkan konsep *Microservices*. *Microservices* adalah sebuah teknik pengembangan perangkat lunak di mana setiap layanan memiliki prosesnya sendiri dan berkomunikasi dengan layanan lain melalui *Application Programming Interface* (API) dan diterapkan dalam satu atau beberapa server yang membentuk sebuah satu kesatuan sistem yang dinamis dan mudah berkembang (Parmar, 2014). Saat ini, Myveego memiliki beberapa layanan yang sudah berjalan, seperti *Dashboard*, *Asset Management System*, dan *Fleet Management System* khusus bidang logistik.

Dalam proses penambangan, alat berat yang biasa digunakan adalah *dump truck* dan *excavator*. Terdapat 2 (dua) area yang menjadi fokus dalam kegiatan pertambangan, yaitu area penambangan dan area pembuangan. Area penambangan adalah area yang ditempati oleh *excavator* sebagai tempat memproduksi barang tambang, sedangkan area pembuangan adalah area tempat barang tambang dikumpul untuk dilanjutkan ke tahap selanjutnya dalam kegiatan pertambangan. Manajemen aktivitas alat berat pertambangan yang baik akan sangat menentukan terhadap jumlah barang tambang yang terkumpul sebagai hasil dari kegiatan pertambangan, serta menentukan pendapatan perusahaan tambang itu sendiri. Namun hal ini menjadi permasalahan yang dirasakan oleh perusahaan tambang. Sebabnya, area pertambangan yang luas membuat jarak antara 2 (dua) area yang menjadi fokus pada penambangan berjauhan sehingga akan sulit dalam melakukan pemantauan dan mengelola aktivitas alat berat pertambangan pada saat melakukan kegiatan pertambangan. Oleh karena itu, dibutuhkan sebuah aplikasi yang dapat melakukan pemantauan visibilitas lokasi, waktu, dan status terhadap aktivitas alat berat pertambangan secara *real-time* guna mendapatkan data koordinat lokasi, waktu yang dibutuhkan dalam setiap prosesnya, dan status aktivitas.

Dengan menggunakan teknologi Geofencing, aplikasi dapat mengetahui pergerakan alat berat pertambangan saat melakukan aktivitasnya dalam kegiatan pertambangan. Geofencing merupakan sebuah mekanisme yang membuat lingkaran atau perimeter maya pada area tertentu di map digital. Teknologi ini menyediakan deteksi entri, tinggal, dan keluar dari pengguna pada wilayah geografis tertentu. Setiap kali pengguna melintasi batas wilayah tersebut, sistem akan menghasilkan *event* yang dapat diolah sesuai dengan kepentingan aplikasi (Suyama and Inoue, 2016).

Dengan memadukan teknologi GPS sebagai dasar sebuah sistem berbasis lokasi, teknologi *Geofencing*, dan teknologi *WebSocket* yang dibangun pada arsitektur *Model-View-Presenter* dengan bahasa pemrograman Kotlin di atas platform Android. Memungkinkan suatu sistem dapat mengetahui pergerakan entri, tinggal, dan keluar dari alat berat pertambangan terhadap area-area yang telah ditentukan dan dengan menampilkan visibilitas lokasi secara *real-time*.

2. LANDASAN KEPUSTAKAAN

2.1 Penambangan

Penambangan adalah proses penggalian barang tambang dan merupakan salah satu tahapan yang dilakukan dalam rangkaian kegiatan pertambangan. Pasal 1 Undang-Undang Nomor 19 Tahun 2009 tentang Pertambangan Mineral dan Batubara dinyatakan bahwa; *"Penambangan adalah bagian kegiatan usaha pertambangan untuk memproduksi mineral dan/atau batubara dan mineral ikutannya"* (Republik Indonesia, 2009). Kegiatan penambangan memiliki 3 (tiga) metode berbeda, yaitu metode tambang terbuka, tambang bawah tanah, dan tambah bawah air.

Terdapat 2 (dua) jenis alat berat yang umum digunakan, yaitu *dump truck* sebagai alat berat angkut dan *excavator* sebagai alat berat gali dan muat (Redaksi, 2019). Terdapat istilah-istilah yang biasa digunakan dalam kegiatan penambangan, diantaranya adalah area tambang (*mine area*), area penyimpanan sementara (*stockpile area*), waktu saat tidak beroperasi (*idle*), siklus yang merupakan rangkaian aktivitas-aktivitas (*cycle*), waktu edar yang diperlukan unit untuk melakukan satu siklus atau perputaran kerja (*cycle time*), dan proses pemuatan (*loading*). Sebuah sistem yang dapat

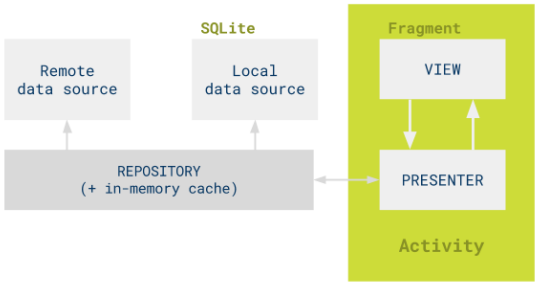
melakukan pemantauan terhadap aktivitas-aktivitas yang dilakukan dalam kegiatan penambangan akan membantu untuk perusahaan pertambangan. Pemantauan tersebut akan dipelajari untuk mengidentifikasi apa saja yang perlu diketahui terkait produktivitas.

2.2 Metodologi Model *Waterfall*

Metodologi model *waterfall* merupakan model pertama yang dipublikasikan dari proses pengembangan perangkat lunak yang berasal dari proses umum dalam rekayasa perangkat lunak. *Waterfall* model merupakan salah satu siklus pengembangan perangkat lunak (*Software Development Life Cycle*). Metodologi ini menuntut pengembang untuk melakukan perencanaan, penentuan, dan penjadwalan seluruh tahapan atau kegiatan proses pengembangan pada awal pelaksanaan sebelum dimulainya pengerjaan. Dalam prinsipnya, hasil dari setiap tahapan adalah sebuah dokumen yang telah disetujui, dengan demikian pengembang dapat melanjutkan ke tahap selanjutnya. Metodologi ini membutuhkan konsistensi terhadap tahapan rekayasa sebelum dan sesudahnya serta dokumentasi dihasilkan di setiap fase (Sommerville, 2011).

2.3 Arsitektur *Model-View-Presenter (MVP)*

Arsitetkur *model-view-presenter* (MVP) merupakan turunan dari pola arsitektur *model-view-controller* (MVC) dan merupakan salah satu pola arsitektur paling populer dalam mengatur lapisan presentasi pada aplikasi Android (Leiva, 2018). Pola arsitektur MVP memisahnya ke dalam 3 (tiga) bagian atau lapisan dasar, yaitu *model*, *view*, dan *presenter*. Di mana masing-masing memiliki tanggung jawab yang berbeda-beda, yaitu lapisan *model* bertanggung jawab untuk menangani semua mekanisme data yang dibutuhkan oleh aplikasi, lapisan *view* bertanggung jawab menampilkan susunan tampilan data ke layar perangkat dan meneruskan interaksi ke lapisan *presenter*, dan lapisan *presenter* berisikan seluruh algoritme dalam aplikasi sekaligus bertanggung jawab sebagai jembatan yang menghubungkan lapisan *view* dengan lapisan *model* (Ojeda-Guerra, 2015). Gambar 1 merupakan diagram pola arsitektur *model-view-presenter* (MVP).



Gambar 1 Diagram Pola Arsitektur *Model-View-Presenter* (MVP)
(Kwon, 2017)

2.4 Teknologi Geofencing

Teknologi Geofencing merupakan sebuah mekanisme yang membuat perimeter maya berbentuk lingkaran pada area tertentu di atas map digital. Tujuannya adalah untuk membuat sebuah perimeter maya berbentuk lingkaran pada area geografis di atas map digital. Teknologi Geofencing menyediakan deteksi apakah pengguna sedang masuk (*enter*) ke dalam area perimeter maya, sedang bergerak atau tinggal (*dwell*) dalam area perimeter maya, atau sedang bergerak keluar (*exit*) dari area perimeter maya (Suyama and Inoue, 2016). Sebuah informasi dapat diberikan melalui pesan notifikasi dengan memanfaatkan *event* sebagai hasil dari deteksi. Gambar 2 adalah ilustrasi dari teknologi geofencing.

Gambar 2 Ilustrasi Teknologi Geofencing
(Hoek, 2016)

2.5 Teknologi WebSocket

WebSocket adalah *push-technology* dan merupakan standar protokol baru dalam komunikasi secara *real-time* pada aplikasi dengan menyediakan saluran komunikasi *full-duplex* yang didasari pada protokol TCP/IP (Darsiwan, 2016). Berdasarkan hasil perbandingan terhadap 4 (empat) jenis *push-technology*, dengan memadukan WebSocket ke dalam suatu sistem, memungkinkan sistem tersebut dapat memperoleh komunikasi secara

real-time, lebih sedikit kehilangan data, hemat sumber daya *server*, mengurangi jumlah *bandwidth*, dan dapat digunakan secara bersama-sama lebih banyak. Oleh karena itu, teknologi WebSocket merupakan *push-technology* pilihan terbaik untuk aplikasi yang memerlukan komunikasi secara *real-time* (Zhang and Shen, 2013).

2.6 Pengujian Perangkat Lunak

2.6.1 White-Box Testing

White-box testing berarti pendekatan sistematis untuk pengujian di mana pengetahuan tentang kode program digunakan untuk merancang tes kesalahan. Tujuannya adalah untuk merancang tes yang menyediakan beberapa tingkat cakupan program (Sommerville, 2011). Salah satu teknik dalam *white-box testing* untuk mendapatkan pengujian yang efisien dan efektif adalah dengan menggunakan metode *basis path testing*. Teknik yang menggunakan tingkat kompleksitas (*cyclomatic complexity*) pada suatu program sebagai panduan untuk menentukan jalur dasar dan kasus uji (Kurniawan, 2007).

2.6.2 Black-Box Testing

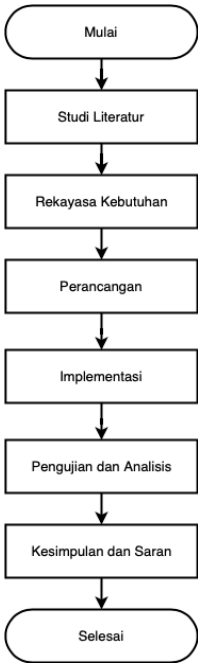
Black-box testing berarti pengujian yang berfokus terhadap kebutuhan yang menjadi persyaratan suatu perangkat lunak. Pengujian ini bukan sebagai alternatif dari *white-box testing*, melainkan untuk saling melengkapi (Kurniawan, 2007). *Black-box testing* diuji dengan tanpa memfokuskan struktur atau algoritme kode, detail implementasi, dan pengetahuan terhadap jalur internal perangkat lunak. Sehingga hanya difokuskan terhadap masukan serta keluaran dari perangkat lunak.

2.6.3 Compatibility Testing

Compatibility testing merupakan pengujian perangkat lunak untuk mencari kesalahan (*error*) yang dikarenakan keadaan lingkungan (*environment*) yang bervariasi di mana sistem dijalankan (Pressman, 2010). Platform, memori, jenis jaringan, resolusi atau dimensi layar, dan sistem operasi adalah termasuk sebagai varian dalam *compatibiliy testing*. Ada beberapa karakteristik yang bisa digunakan dalam *compatibility testing*, yaitu *adaptability*, *replaceability*, dan *installability*. *Adaptability* berarti perangkat lunak memiliki kemampuan beradaptasi dan dapat disesuaikan dengan berbagai lingkungan yang ditentukan. *Replaceability* berarti perangkat lunak memiliki kemampuan penggantian di lingkungan yang sama, yaitu peningkatan versi (*upgrade*) atau penurunan versi (*downgrade*). *Installability* berarti perangkat lunak memiliki kemampuan pemasangan (*install*) yang mudah secara efektif (Liu, 2014).

3. METODOLOGI PENELITIAN

Tahapan penelitian dimulai dengan dilakukannya studi literatur, rekayasa kebutuhan, perancangan, implementasi, pengujian dan analisis, serta diakhiri dengan kesimpulan dan saran. Gambar 3 adalah diagram alur metodologi penelitian.



Gambar 3. Metodologi Penelitian

3.1 Studi Literatur

Studi literatur berarti penelitian memasuki tahap mengumpulkan referensi guna menyusun landasan teori sebagai pendukung dalam melakukan penelitian. Penelitian menggunakan artikel, buku, situs web, hasil penelitian sebelumnya, dan jurnal ilmiah sebagai sumber referensi. Sumber-sumber yang digunakan diharuskan berhubungan dengan topik yang didiskusikan dalam penelitian.

3.2 Rekayasa Kebutuhan

Rekayasa kebutuhan berarti penelitian telah memulai tahapan untuk mengumpulkan informasi dan kemudian dianalisis untuk mendapatkan solusi yang menjadi kebutuhan

pengguna dan harus dimiliki oleh perangkat lunak. Rekayasa kebutuhan dilakukan dalam beberapa tahapan, dimulai dengan elisitasi dan analisis kebutuhan, spesifikasi kebutuhan, identifikasi aktor, dan terakhir adalah pemodelan kebutuhan.

Elisitasi kebutuhan bertujuan untuk mencari tahu dan mendapatkan proses bisnis serta pihak-pihak yang terlibat di dalamnya. Dalam pembuatan aplikasi pemantauan alat berat pertambangan, elisitasi kebutuhan dilakukan dengan cara observasi melalui sebuah dokumen yang diberikan oleh divisi *Digital Solution and Services* (DSS) departemen *IoT and Automation* PT. Astra Graphia Information Technology (AGIT). Selanjutnya dilakukan analisis kebutuhan dari hasil elisitasi untuk mendapatkan aktor yang berperan dan kebutuhan yang harus dimiliki oleh perangkat lunak.

Kebutuhan dari hasil elisitiasi didefinisikan sebagai kebutuhan fungsional atau kebutuhan non-fungsional. Kemudian dilakukan spesifikasi untuk mencari tahu fungsi yang harus dimiliki oleh perangkat lunak. Kemudian kebutuhan hasil elisitiasi dimodelkan dengan metode *unified modeling language* (UML). Sehingga melalui rekayasa kebutuhan memberikan hasil berupa identifikasi aktor, *use case diagram*, dan *use case scenario*.

3.3 Perancangan

Perancangan berarti kebutuhan perangkat lunak telah terdefinisikan, termodelkan, dan siap untuk diterjemahkan ke dalam beberapa gambaran dengan metode *unified modeling language* (UML). Perancangan dikerjakan melalui beberapa tahap, yaitu perancangan arsitektur, perancangan basis data (*database*), perancangan komponen, dan perancangan antarmuka.

3.4 Implementasi

Implementasi berarti perangkat lunak akan memasuki fase proses pembuatan yang menerjemahkan *pseudocode* sebagai hasil perancangan komponen menjadi kode program. Proses implementasi harus merujuk dari hasil perancangan sistem. Implementasi dikerjakan melalui beberapa tahap, yaitu spesifikasi sistem, implementasi basis data (*database*), implementasi kode program, dan implementasi antarmuka.

3.5 Pengujian dan Analisis

Pengujain dan analisis berarti perangkat lunak pada penelitian telah selesai diimplementasi. Pengujian dilakukan melalui proses verifikasi dan validasi untuk memperoleh hasil apakah perangkat lunak bersesuaian dengan hasil rekayasa kebutuhan dan perancangan yang dilakukan pada tahap sebelumnya sebagai bagian dari tahap penelitian. Verifikasi dilakukan melalui pengujian unit dan integrasi dengan mempergunakan metode *basis path testing* untuk *white-box testing*. Validasi dilakukan pada kebutuhan fungsional dengan menggunakan *black-box testing*. Pengujian *compatibility* digunakan terhadap kebutuhan non-fungsional. Data yang dipergunakan pada tahap pengujian bersifat *dummy* atau bukan merupakan data yang sebenarnya.

3.6 Kesimpulan dan Saran

Kesimpulan dan saran berarti penelitian telah memasuki tahapan terakhir apabila tahap-tahap sebelumnya dalam metodologi telah selesai dilakukan. Mendapatkan kesimpulan berdasarkan hasil dari seluruh tahapan dalam metodologi yang telah dikerjakan dan digunakan sebagai jawaban dari rumusan masalah dan tujuan dalam penelitian. Saran digunakan sebagai masukan dalam memperbaiki kurang tepatnya hal yang dilakukan dalam penelitian dan menjadi pertimbangan dalam mengembangkan sistem selanjutnya ke arah yang lebih baik.

4. REKAYASA KEBUTUHAN

4.1 Elisitasi Kebutuhan

Elisitasi kebutuhan dilakukan dengan teknik wawancara dan observasi. Hasil dari kegiatan elisitasi kebutuhan berupa informasi terkait permasalahan yang dihadapi oleh perusahaan PT. Astra Graphia Information Technology (AGIT), di mana MyVeego bersama dengan departemen IoT and Automation berencana mengembangkan sistem pemantauan alat berat yang lebih berfokus pada kegiatan penambangan. Sistem pemantauan alat berat diberi nama dengan Mining Management. Hal-hal yang akan dipantau diantaranya adalah pergerakan, aktivitas yang dilakukan, dan waktu yang dibutuhkan pada setiap aktivitas dalam kegiatan penambangan. Selain sebagai sistem pemantauan, Mining Management juga bertujuan untuk memberikan informasi terkait

produktivitas hasil penambangan.

4.2 Analisis Kebutuhan

Melakukan analisis terhadap hasil elisitasi kebutuhan yang telah dilakukan melalui teknik wawancara dan observasi. Analisis kebutuhan memberikan hasil pada kegiatan analisis kebutuhan berupa proses bisnis (*to-be*), identifikasi aktor, dan daftar kebutuhan. Berdasarkan proses bisnis (*to-be*), identifikasi aktor, dan daftar kebutuhan tersebut, kemudian akan dimodelkan dengan pemodelan berorientasi objek *unified modeling language* (UML) berupa *use case diagram* dan *use case scenario*.

Dalam proses bisnis (*to-be*) penambangan, setiap aktivitas seperti perpindahan lokasi dan status aktivitas yang sedang dilakukan akan dipantau. Terdapat 2 (dua) tempat (*area*) yang menjadi fokus dalam proses bisnis (*to-be*) penambangan, yaitu *disposal area* (tempat pembuangan) dan *mining area* (tempat penambangan) dengan berdasarkan pada posisi petugas *excavator*. Setiap titik lokasi dari *disposal* area dan *mining* area diberikan *geofence* berdiameter 500 meter. Proses pengangkutan barang tambang dari *mining area* menuju *disposal area* dihitung sebagai 1 (satu) siklus.

Proses bisnis (*to-be*) penambangan dimulai dengan menujunya petugas dump truck ke *mining area* yang telah ditentukan berdasarkan jarak dan jumlah antrian. Selanjutnya dilakukan aktivitas pemuatan (*loading*) barang tambang ke *dump truck*. Selanjutnya petugas menunju kembali ke *disposal area*. Aktivitas terakhir adalah pengeluaran (*dumping*) barang tambang dari *dump truck* ke tempat pembuangan untuk diteruskan ke tahap selanjutnya setelah penambangan.

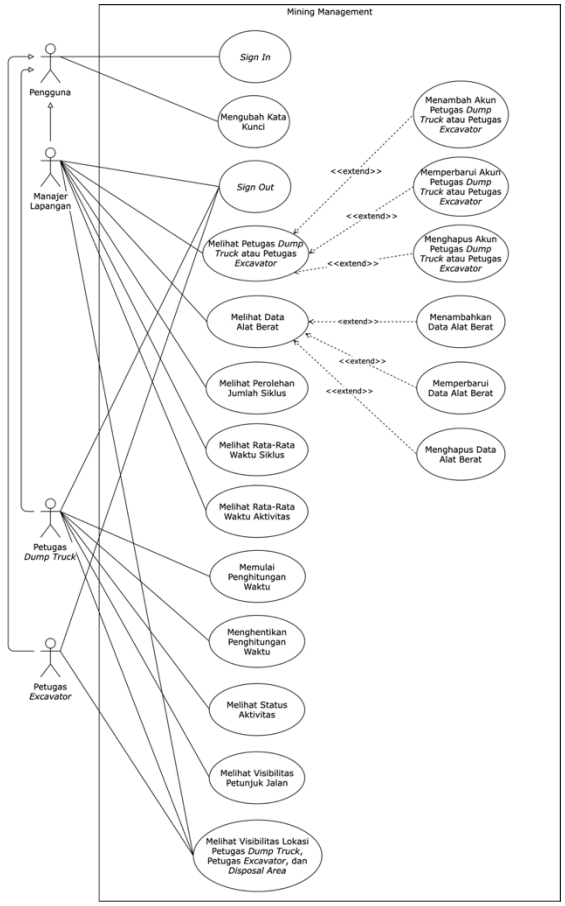
4.3 Identifikasi Aktor

Ditemui aktor-aktor yang memiliki peran dalam aplikasi pemantauan alat berat pertambangan sebagai bagian dari hasil analisis kebutuhan, yaitu pengguna, petugas *dump truck*, petugas *excavator*, dan manajer lapangan. Pengguna berupa aktor yang belum memiliki hak akses, wewenang, dan tervalidasi terhadap sistem. Petugas *dump truck* merupakan aktor yang memiliki tugas untuk mengoperasikan *dump truck*. *Dump truck* adalah alat berat muat yang digunakan untuk membantu proses pengantaran barang tambang. Petugas *excavator* merupakan aktor yang memiliki tugas untuk mengoperasikan *excavator*. *Excavator* adalah alat berat gali yang digunakan untuk membantu proses penggalian barang tambang dan memindahkannya ke *dump truck*. Manajer lapangan merupakan aktor yang memiliki tugas untuk memimpin, melakukan pengawasan, dan pemantauan terhadap aktivitas-aktivitas pertambangan, serta bertanggung jawab terhadap jalannya kegiatan penambangan.

4.4 Spesifikasi Kebutuhan

Spesifikasi kebutuhan berarti pendefinisian kebutuhan pengguna yang harus dipenuhi oleh sistem berdasarkan hasil analisis kebutuhan. Spesifikasi kebutuhan menemukan 19 (sembilan belas) kebutuhan fungsional dan 1 (satu) kebutuhan non-fungsional sebagai hasil dari spesifikasi kebutuhan.

4.5 Pemodelan Kebutuhan



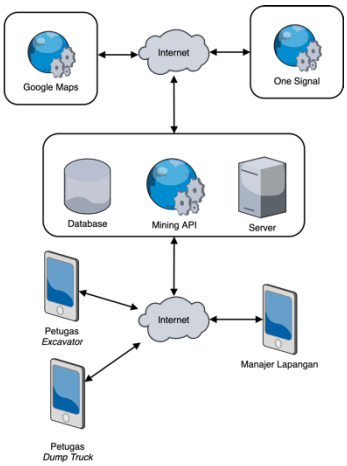
Gambar 4. *Use Case Diagram* Mining Management

Kebutuhan yang sudah didefinisikan dalam tahap spesifikasi kebutuhan, kemudian dimodelkan menggunakan metode *unified modeling language* (UML) berupa *use case diagram* dan dilakukan penentuan skenario yang

terjadi pada setiap kebutuhan tersebut. Gambar 4 adalah *use case diagram* dari Mining Management.

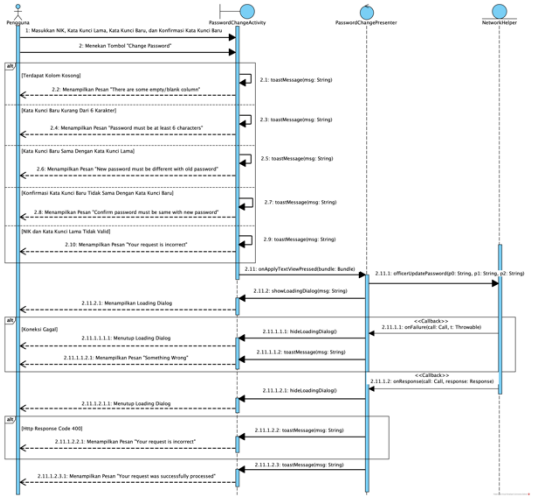
5. PERANCANGAN

Mining Management membutuhkan sebuah *service* sebagai penghubung komunikasi dan sekaligus penerjemah antara 2 (dua) atau lebih perangkat atau bahasa yang berbeda. *Service* tersebut dikembangkan dengan menggunakan bahasa pemrograman Javascript dan didukung oleh *framework* ExpressJS. Supaya tercapainya *real-time communcation*, teknologi WebSocket diimplementasikan ke dalam *service* dengan dukungan *framework* Socket.IO. Pada sisi aplikasi perangkat bergerak, Mining Management dikembangkan dengan mengimplementasikan pola arsitektur *model-view-presenter* (MVP) yang keunggulannya bisa dimanfaatkan untuk sebuah perangkat lunak yang memiliki sebuah *service* yang akan berjalan sendiri tanpa ada kebergantungan dengan *Activity/Fragment* dan memadukan teknologi WebSocket yang didukung oleh *framework* Socket.IO. Gambar 5 adalah arsitektur Mining Management secara keseluruhan.



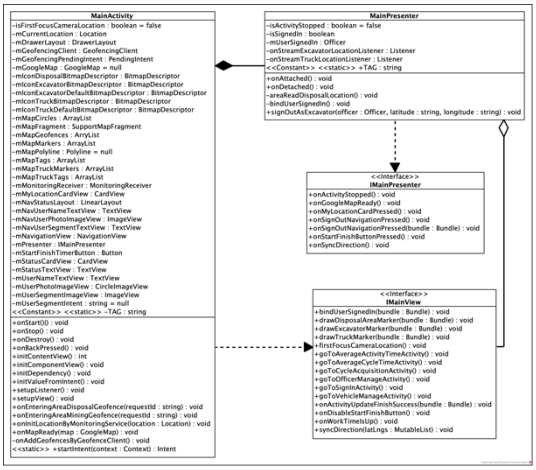
Gambar 5. Arsitektur Sistem Mining Management Keseluruhan

Perancangan *sequence diagram* merupakan bagian dalam tahap perancangan yang melakukan pemodelan kebutuhan berdasarkan tahap sebelum perancangan menjadi *sequence diagram*. *Sequence diagram* mengilustrasikan interaksi yang terjadi dalam sistem dengan memfokuskan pada urutan pesan, bersama dengan spesifikasi kejadian yang sesuai pada *lifeline* (Fakhroutdinov, 2009b). Gambar 6 adalah *sequence diagram* Mining Management pada kebutuhan mengubah kata kunci.



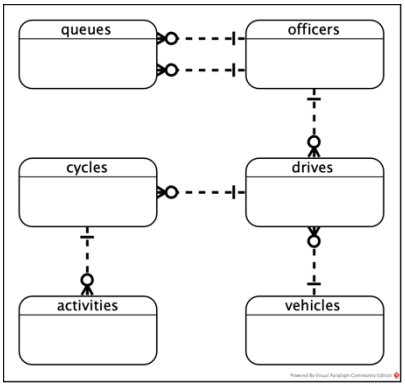
Gambar 6. *Sequence Diagram* Mengubah Kata Kunci

Perancangan *class diagram* merupakan bagian dalam tahap perancangan yang melakukan pemodelan rancangan arsitektur pada tingkat *class* hingga ke fungsi menjadi *class diagram*. *Class diagram* berarti diagram struktur yang menunjukkan struktur sistem yang dirancang pada tingkat *class* dan *interface*. Menggambarkan tentang batasan (*constraints*) dan relasi (*assosiations*, *generalizations*, *dependencies*) (Fakhroutdinov, 2009a). Mining Management pada sisi aplikasi perangkat bergerak dikembangkan dengan mengimplementasikan pola arsitektur *model-view-presenter* (MVP). Gambar 7 adalah *class diagram* pada *package* main.



Gambar 7. *Class Diagram Package* Main

Perancangan basis data memberikan hasil berupa *conceptual data model* (CDM). CDM menjelaskan entitas dan relasi antar entitas, tidak menjelaskan atribut atau nama kolom yang terdapat dalam entitas. Gambar 8 adalah CDM dari Mining Management.



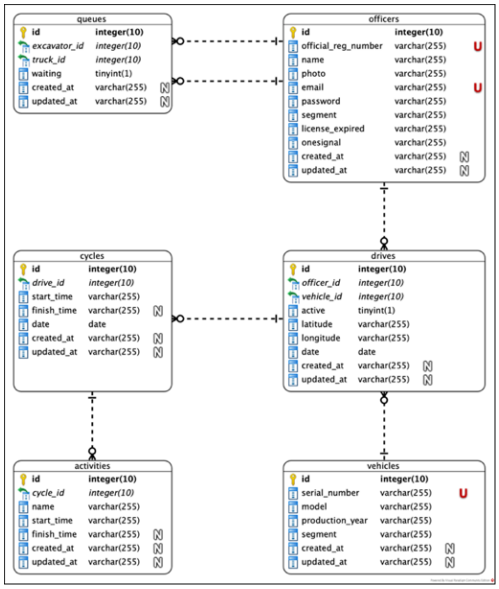
Gambar 8. *Conceptual Data Model* (CDM) Mining Management

Perancangan komponen merupakan bagian dari tahap perancangan yang merencanakan algoritme atau struktur kode yang akan digunakan. Algoritme atau struktur kode tersebut ditulis menggunakan *pseudocode*. *Pseudocode* merupakan cara penulisan algoritme yang bersifat bukan sintaksis. Perancangan antarmuka merupakan bagian dari tahap perancangan yang merencanakan tampilan antarmuka terhdap sistem yang dikembangkan berdasarkan kebutuhan yang didapat. Perancangan antarmuka akan menghasilkan *mock-up* yang akan memvisualisasikan antarmuka pada sistem.

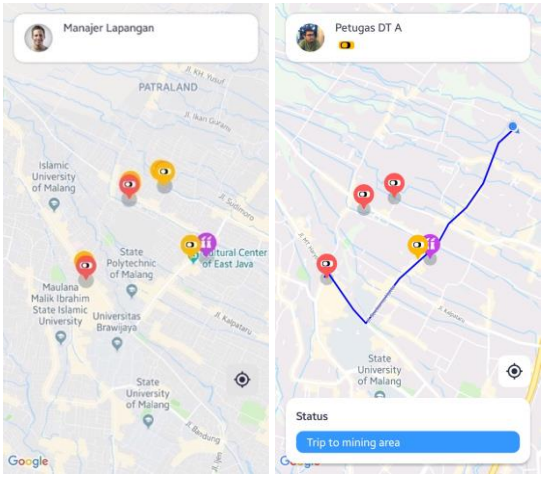
6. IMPLEMENTASI

Hasil perancangan basis data yang berupa *conceptual data model* (CDM) diimplementasi menggunakan *relational database management system* (RDBMS). *Physical data model* (PDM) dimanfaatkan sebagai representasi terhadap basis data (*database*) yang telah diimplementasikan. PDM memberitahu seluruh struktur tabel, nama kolom, batasan kolom (*constraint*), *primary key*, *foreign key*, dan relasi antar tabel (1keydata, 2001). Gambar 9 adalah *physical data model* (PDM) dari Mining Management.

Pseudocode merupakan hasil dari perancangan kode program dan diimplementasi menggunakan bahasa pemrograman Kotlin versi 1.3 dan Javascript pada service atau REST API (*Representation State Transfer Application Programming Interface*). Hasil perancangan antarmuka yang berupa *mock-up* diimplementasi menggunakan bahasa pemrograman XML (*Extensible Markup Language*) yang menjadi bahasa standar dalam mengembangkan aplikasi perangkat bergerak berbasis Android. Gambar 10 adalah contoh halaman yang telah diimplementasikan.



Gambar 9. *Physical Data Model* (PDM) Mining Management



(a) (b)

Gambar 10. Implementasi Antarmuka: (a) Halaman Utama Manajer Lapangan; (b) Halaman Utama Petugas *Dump Truck*

7. PENGUJIAN DAN ANALISIS

Verifikasi dilakukan melalui pengujian unit dan integrasi, dengan mempergunakan metode *basis path testing* untuk *white-box testing.* Validasi dilakukan terhadap kebutuhan fungsional dan non-fungsional dengan mempergunakan *black-box testing* serta menggunakan pengujian *compatibility* terhadap kebutuhan non-fungsional. Data yang dipergunakan pada tahap pengujian bersifat *dummy* atau bukan merupakan data yang sebenarnya.

Pengujian unit berarti pengujian berfokus terhadap kode program sebagai unit terkecil dalam suatu sistem. Kode program dalam fungsi yang menjadi unit terkecil akan diuji untuk

diperoleh apakah kode program tersebut layak untuk digunakan. Penngujian integrasi bertujuan untuk menemukan kesalahan yang berlangsung dalam interaksi antar unit individual dalam sistem. Pengujian integrasi dikerjakan dengan menerapkan metode *white-box testing* dengan pendekatan *top-down*. Pendekatan *top-down* berarti pengujian dilakukan dimulai dari unit bagian atas menuju ke unit bagian bawah. Fungsi drawDisposalMarker() dari kelas MainActivity adalah fungsi yang dipilih untuk dilakukan pengujian integrasi.

Pengujian validasi berarti pengujian yang dikerjakan guna mengetahui apakah perangkat lunak dibangun sesuai pada kebutuhan yang telah didefinisikan dalam tahap rekayasa kebutuhan. Pengujian validasi dikerjkan dengan mempergunakan metode *black-box testing*. Pengujian *compatibility* dilakukan untuk mencari kesalahan (*error*) pada sistem yang dikarenakan keadaan lingkungan (*environment*) yang berbeda-beda di mana sistem dijalankan dan digunakan. Pengujian compatibility pada penelitian ini berfokus pada karakteristik *adaptability*. Parameter pengujian *compatibility* pada penelitian ini adalah ukuran dimensi layar, sistem operasi, dan merek produsen perangkat yang termasuk ke dalam adaptabilitas perangkat keras (*hardware adaptability*) dan adaptabilitas sistem operasi (*operating system adaptability*). Tabel 1 adalah hasil pengujian *compatiblity*.

Tabel 1. Hasil Pengujian *Compatibility*

Device	Parameter	
	Dimensi Layar	Sistem Operasi
Google 7 API 21 (Tablet)	Valid	Valid
Google 7 API 22 (Tablet)	Valid	Valid
Google Nexus 9 API 23 (Tablet)	Valid	Valid
Google Nexus 9 API 24 (Tablet)	Valid	Valid
Google Nexus 9 API 25 (Tablet)	Valid	Valid

8. KESIMPULAN DAN SARAN

Hasil elisitasi kebutuhan dianalisis pada tahap analisis kebutuhan dan memberikan hasil yang terdiri dari: (1) Proses bisnis *to-be*, (2) Identifikasi aktor yang berperan dalam sistem sejumlah 4 (empat) aktor, yaitu: pengguna, manajer lapangan, petugas *dump truck*, dan petugas *excavator*, (3) Spesifikasi kebutuhan sejumlah 19 (sembilan belas) kebutuhan fungsional dan 1 (satu) kebutuhan non-fungsional, (4) *Use case diagram* dan *use case scenario* merupakan hasil dari kebutuhan sistem yang telah dimodelkan.

Perancangan arsitektur sistem memberikan hasil berupa struktur sistem secara menyeluruh. Perancangan *sequence diagram* memberikan hasil berupa *sequence diagram* yang mengilustrasikan interaksi yang terjadi di sistem. Perancangan *class diagram* memberikan hasil berupa *class diagram* yang merepresentasikan struktur dalam aplikasi Mining Management dengan mempergunakan pola arsitektur *Model-View-Presenter* (MVP) yang memberikan salah satu manfaat berupa pemisahan *background-task* dengan *Activity/Fragment* sehingga tidak tergantung dengan siklus hidup dari *Activity/Fragment*. Perancangan basis data (*database*) memberikan hasil berupa *conceptual data model* (CDM). Perancangan komponen memberikan hasil berupa *pseudocode* algoritme yang digunakan pada suatu fungsi. Perancangan antarmuka memberikan hasil berupa *mock-up* yang akan memvisualisasikan antarmuka sistem yang dibangun.

Implementasi basis data (*database*) yang memberikan hasil *physical data model* (PDM). Implementasi komponen yang memberikan hasil berupa kode program yang dipergunakan dalam sistem. Implementasi antarmuka yang memberikan hasil berupa tangkapan antarmuka sistem. Aplikasi pemantauan alat berat pertambangan diimplementasikan dengan menggunakan pola arsitektur *model-view-presenter* (MVP) dengan memadukan teknologi Geofencing dan WebSocket.

Pengujian unit dan pengujian integrasi memberikan hasil valid dan nilai *complexity number* untuk masing-masing pengujian di bawah 10. Dengan demikian, algoritme yang dipergunakan pada aplikasi pemantauan alat berat pertambangan adalah *high testability*. Pengujian validasi pada 19 (sembilan belas) kebutuhan fungsional memberikan hasil valid pada 51 (lima puluh satu) kasus uji. Pengujian *compatibility* dikerjakan pada Firebase Test Lab dan memberikan perolehan bahwa sistem dapat diinstalasi dan dijalankan dengan normal diberbagai perangkat dengan sistem operasi dan dimensi layar yang beraneka.

9. DAFTAR PUSTAKA

1keydata, 2001. *Physical Data Model*. [online] Tersedia di: <https://www.1keydata.com/datawarehousing/physical-data-model.html> [Diakses 26 May 2019].

Darsiwan, 2016. *Apa itu WebSocket*. [online] Tersedia di: <https://www.codepolitan.com/menegtahui-apa-itu-websocket> [Diakses 13 Dec. 2018].

Fakhroutdinov, K., 2009a. *UML Class and Object Diagrams Overview*. [online] Tersedia di: <https://www.uml-diagrams.org/class-diagrams-overview.html> [Diakses 10 Mar. 2019].

Fakhroutdinov, K., 2009b. *UML Sequence Diagrams*. [online] Tersedia di: <https://www.uml-diagrams.org/sequence-diagrams.html> [Diakses 10 Mar. 2019].

Hoek, M., 2016. *Making mobile apps smarter with geofence tehcnology*.

Kurniawan, T.A., 2007. Pengujian Struktur Program Dengan Pengujian Jalur Dasar (Basis Path Testing) : Teori Dan Aplikasi. *Eeccis*, [online] 1(1), pp.29–32. Tersedia di: <http://jurnaleeccis.ub.ac.id/index.php/eeccis/article/viewFile/357/266>.

Kwon, T., 2017. *Android MVP 무작정 따라하기 - Google Architecture의 Model*. [online] Tersedia di: <https://thdev.tech/androiddev/2017/01/09/Android-MVP-Model-Two/> [Diakses 21 May 2019].

Leiva, A., 2018. *MVP for Android: how to organize the presentation layer*. [online] Tersedia di: <https://antonioleiva.com/mvp-android/> [Diakses 13 Dec. 2018].

Liu, Z., 2014. Research on Software Security and Compatibility Test for Mobile Application. *Fourth edition of the International Conference on the Innovative Computing Technology (INTECH 2014)*, [online] pp.140–145. Tersedia di: <https://ieeexplore.ieee.org/document/6927764>.

Ojeda-Guerra, C.N., 2015. A Simple Software Development Methodology Based on MVP for Android Applications in a Classroom Context. [online] pp.1429–1434. Tersedia di: <https://ieeexplore.ieee.org/document/7363258>.

Parmar, K., 2014. *Microservice Architecture – A Quick Guide*. [online] Tersedia di: <https://www.javacodegeeks.com/2014/06/microservice-architecture-a-quick-guide.html> [Diakses 5 Dec. 2018].

Pressman, R.S., 2010. *Software Engineering*. 7th ed. [online] New York: McGraw-Hill. Available at: <http://dinus.ac.id/repository/docs/ajar/RPL-7th_ed_software_engineering_a_practitioners_approach_by_roger_s._pressman_.pdf>.

Redaksi, 2019. *Jenis dan Fungsi Alat Berat Pada Tambang Batubara*. [online] Tersedia di: <https://icononline.id/post/jenis-dan-fungsi-alat-berat-pada-tambang-batubara-> [Diakses 9 Apr. 2019].

Republik Indonesia, 2009. *Undang-Undang Nomor 4 Tahun 2009 Tentang Pertambangan Mineral dan Batubara*. Tersedia di: <https://www.hukumonline.com>.

Sommerville, I., 2011. *Software Engineering 9th Edition*. USA: Pearson Education, Inc.

Suyama, A. and Inoue, U., 2016. Using geofencing for a disaster information system. *2016 IEEE/ACIS 15th International Conference on Computer and Information Science, ICIS 2016 - Proceedings*. [online] Tersedia di: <https://ieeexplore.ieee.org/document/7550849>.

Zhang, L. and Shen, X., 2013. Research and development of real-time monitoring system based on WebSocket technology. *Proceedings - 2013 International Conference on Mechatronic Sciences, Electric Engineering and Computer, MEC 2013*, [online] pp.1955–1958. Tersedia di: <https://ieeexplore.ieee.org/document/6885373>.