Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer e-ISSN: 2548-964X
Vol. 3, No. 5, Mei 2019, hlm. 4415-4423 http://j-ptiik.ub.ac.id

Perancangan Aplikasi pada Pengelolaan Sewa Barang

Windya Hartasih[1], Denny Sagita Rusdianto[2], Adam Hendra Brata[3]

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

Abstrak

Dalam penyewaan, aktivitas yang biasa dilakukan penyedia jasa sewa memuat pemesanan barang, pengambilan barang, pengembalian barang, pengecekan stok barang, dan pembukuan. Namun, pada saat ini banyak penyedia jasa sewa yang menemui kendala dalam menjalankan aktivitas bisnis tersebut, seperti pada penjadwalan barang, pencatatan pesanan, perbedaan stok barang dan pembukuan rutin sehingga dapat menyebabkan kerugian dan masalah dengan pelanggan. Selain itu, penyedia jasa sewa memiliki keterbatasan infrastruktur, dengan masih menggunakan nota penyewaan, papan tulis, dan software Microsoft Excel dalam melakukan pembukuan. Oleh karena itu, penyedia jasa sewa membutuhkan aplikasi pengelolaan sewa barang yang *independent* atau mandiri, agar tidak bergantung pada infrastruktur atau membutuhkan infrastruktur yang lebih, sehingga jasa sewa hanya menggunakan *smartphone* saja tanpa perlu menambah infrastruktur lain dalam mengelola jasa sewa. Aplikasi ini dirancang menggunakan metode OOAD *(Object Oriented Analysis and Design*) dan web *service* REST. Perancangan yang dilakukan adalah perancangan service, arsitektur, komponen, data, dan antarmuka, serta desain arsitektur aplikasi. Perancangan aplikasi ini telah diuji menggunakan *Traceability Matrix, (Coupling Between Object Classes) CBO, (Response for a Class) RFC, (Lack of cohesion in methods1) LCOM1 metrics*, dan *(Lack of cohesion in methods2) LCOM2 metrics*. Hasil perancangan aplikasi ini memiliki kopling rendah karena rata-rata nilai CBO<=5 dan RFC<=100 sehingga *understandability* tinggi dan kohesi rendah karena memiliki LCOM1>1. Hasil perancangan juga memiliki *Adaptability* tinggi karena rata-rata nilai CBO diantara 1 hingga 3, rata-rata nilai RFC diantara 1 hingga 69, dan memenuhi nilai LCOM2 0 hingga 1. Selain itu, semua artefak dibuat berdasarkan *use case* yang telah dibuat sebelumnya, karena setiap kolom dan baris pada *traceability matrix* memiliki nilai 1.

Kata kunci: *penyewaan, infrastruktur, pengelolaan sewa barang, OOAD, web service REST*

Abstract

In rental services, activity that usually do by rental provider are item ordering, item taking, item returning, item checking, and bookkeeping. Nowadays, many rental providers have difficulties for running their business activity, such as item scheduling, order recording, differences between item inventory and routine bookkeeping that causes losses and problems with customers. Moreover, the rental providers have limited infrastructure, by keep using rental note, whiteboard, and Microsoft Excel software for bookkeeping. Therefore, the rental providers need an rental management application which is independent in order to not depend on infrastructure or need more infrastructure, so that the service providers can only use smartphone without adding other infrastructure to manage rental services. This application designed using OOAD (Object Oriented Analysis and Design) method and REST web service. Designing this application include service design, architecture design, component design, data design, user interface design, and application design architecture. This design of application has been tested using Traceability Matrix, (Coupling Between Object Classes) CBO, (Response for a Class) RFC, (Lack of cohesion in methods1) LCOM1 metrics, and (Lack of cohesion in methods2) LCOM2 metrics. The result of this application design has low coupling because of the CBO's average value<=5 and RFC's average value<=100 so it have high understandability and low cohesion because of LCOM>1. The result of this application design also has high adaptability because of the CBO's average value is between 1 and 3, RFC's average value between 1 and 69, and fulfill LCOM2's value between 0 and 1. Furthermore, all artifacts are made based on the use case that was made before, because each column and row in the traceability matrix have a value of 1.

Keywords: *rental services, infrastructure, rental management, OOAD, REST web service*

1. PENDAHULUAN

Bidang usaha penyewaan selalu menarik untuk di jalankan, salah satu penyebabnya adalah bisa menghasilkan banyak keuntungan dan cukup mudah untuk di jalankan, karena tidak perlu produksi, mencari supplier, dan lain sebagainya. Pada penyewaan, aktivitas yang biasa dilakukan pelanggan diantaranya yaitu memesan barang untuk disewa dan mengembalikan barang. Proses bisnis penyedia jasa sewa diantaranya memuat pelayanan pemesanan barang, pengecekan barang antara stok di gudang dengan di data atau rekap barang (inventaris barang), pelayanan pengembalian barang dan pelunasan, serta pembukuan rutin yang dilakukan pada periode per hari, per minggu, per bulan, atau per tahun untuk merekap barang apa saja yang sering disewa, mencatat pemasukan serta pengeluaran yang dilakukan pada penyedia jasa sewa, serta menyimpan pembukuan sehingga penyedia jasa sewa dapat mengetahui keuntungan yang didapatkan dan mengetahui perkembangan persewaan pada jangka waktu tertentu.

Namun saat ini, penyedia jasa sewa memiliki masalah pada aktivitas bisnis seperti penjadwalan barang, pencatatan pesanan, perbedaan stok barang, dan pembukuan sehingga dapat menyebabkan kerugian dan masalah dengan pelanggan. Oleh karena itu, perlu adanya teknologi komputer yang dapat membantu mengatasi permasalahan pada aktivitas bisnis penyedia jasa sewa yang telah didefinisikan sebelumnya.

Dengan adanya teknologi komputer yang berupa aplikasi, dapat membantu penyedia jasa sewa dalam mengatasi dan meminimalisir permasalahan yang terjadi. Penyedia jasa sewa melakukan aktivitas bisnis menggunakan buku pesanan, nota penyewaan, papan tulis, dan pada beberapa jasa sewa menggunakan software Microsoft Excel untuk melakukan pembukuan karena keterbatasan infrastruktur. Maka dari itu, penyedia jasa sewa membutuhkan aplikasi pengelolaan sewa barang yang *independent* atau mandiri, agar tidak bergantung pada infrastruktur atau membutuhkan infrastruktur yang lebih, sehingga jasa sewa hanya menggunakan *smartphone* saja tanpa perlu menambah infrastruktur lain dalam mengelola jasa sewa.

Oleh karena itu, pada penelitian ini, dirancang aplikasi pengelolaan sewa barang agar dapat membantu penyedia jasa sewa dalam mengelola sewa barang.

2. METODOLOGI PENELITIAN

Dalam penelitian ini, metode perancangan yang akan digunakan adalah OOAD (Object Oriented Analysis and Design). Pemilihan metode ini dikarenakan aplikasi yang dirancang berfokus pada pendefinisian class dan cara bagaimana mereka bekerja sama antara satu sama lain untuk menghasilkan kebutuhan penyedia jasa sewa. Adapun diagram alir dari metodologi penelitian ditunjukkan oleh Gambar 1.

Pada Gambar 1. Diagram alir metodologi penelitian dapat dijelaskan sebagai berikut.

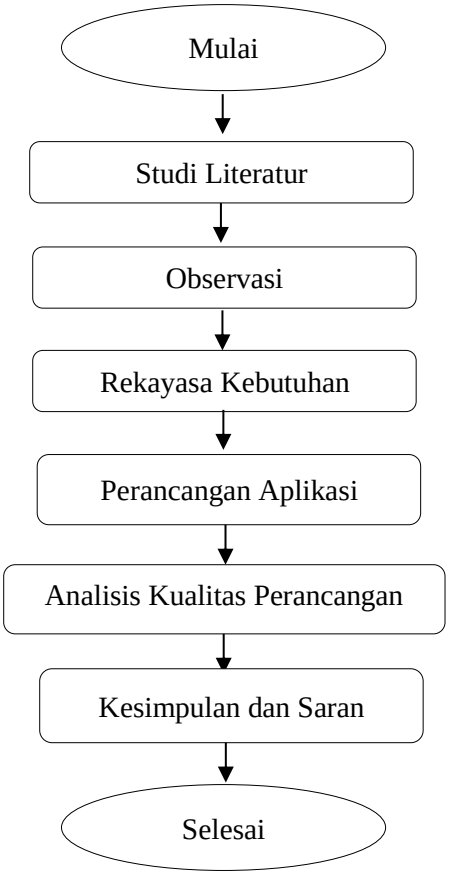
Penjelasan dasar teori didapat dari penelitian yang pernah dilakukan sebelumnya, *ebook*, *jurnal*, *conference proceedings*, dan beberapa literatur dari internet untuk studi literatur.

Penjelasan mengenai rekayasa kebutuhan adalah penjabaran proses bisnis dalam BPMN, identifikasi aktor, pendefinisian kebutuhan yang nantinya akan digambarkan dalam diagram *use case* dan dijabarkan dalam *scenario use case*.

Pada tahap perancangan aplikasi dilakukan desain arsitektur aplikasi untuk menjelaskan arsitektur *software* yang digunakan, perancangan *service* untuk membuat konsep *service* dan kemampuan dari setiap *service* (Erl, 2017), perancangan komponen untuk mendeskripsikan logika proses yang sistem harus lakukan (Shelly & Rosenblatt, 2012), perancangan antarmuka untuk mendeskripsikan bagaimana pengguna berinteraksi dengan sistem (Shelly & Rosenblatt, 2012), perancangan data untuk memahami bagaimana data akan diorganisir, disimpan, dan dikelola (Shelly & Rosenblatt, 2012), serta perancangan arsitektur menggunakan diagram UML untuk membantu dalam membangun perangkat lunak sehingga memahami spesifikasi sistem dan rancangan sistem (Pressman, 2010).

Pada tahap analisis kualitas perancangan dilakukan pengujian dasar untuk mengetahui apakah aplikasi yang dirancang sudah sesuai dengan kebutuhan dan pengujian kualitas untuk mengukur kohesi, kopling, *adaptability*, dan *understandability* aplikasi.

Pada tahap kesimpulan dan saran akan berisi kesimpulan akhir yang menjawab rumusan masalah yang telah dijabarkan dan berisi saran untuk perancangan selanjutnya di waktu yang akan datang.



Gambar 1. Diagram alir metodologi penelitian

3. Rekayasa Kebutuhan

3.1. Analisis Kebutuhan

Setelah dilakukan elisitasi kebutuhan, didapatkan proses bisnis penyedia jasa sewa secara umum yang kemudian dijabarkan dengan menggunakan BPMN (*Business Process Model and Annotation*). Proses pengelolaan sewa barang yang terjadi pada penyedia jasa sewa memuat 13 proses, dimana salah satunya adalah proses mendaftarkan pemilik. Pada 5 proses seperti melakukan pembayaran sewa, melayani pemesanan barang, melayani pengambilan barang, melayani pengembalian barang, dan melakukan pembukuan masing-masing memiliki sub proses untuk mendefinisikan detail aktivitas. Aktor yang terlibat meliputi aplikasi pengelolaan sewa barang, admin, pemilik, dan staf, dimana pemilik dituliskan dua kali pada aktor karena pemilik memiliki 2 jenis proses, yaitu proses yang hanya bisa dilakukan oleh pemilik sendiri dan proses yang bisa dilakukan oleh pemilik dan staf.

3.2. Identifikasi Aktor

Aktor merupakan seseorang ataupun sistem yang dapat berinteraksi dengan sistem. Adapun aktor dalam aplikasi ini ditunjukkan pada Tabel 1.

Tabel 1. Aktor Sistem

No.	Aktor	Deskripsi
1	Pengguna	Merupakan generalisasi admin, pemilik, dan staf yang belum masuk ke dalam sistem
2	Pemilik	Merupakan pemilik dari penyedia jasa sewa yang dapat melakukan semua fungsi yang dilakukan staf, membuat akun staf, mambuat promo, serta melakukan pembayaran sewa aplikasi.
3	Staf	Merupakan staf dari penyedia jasa sewa yang melayani sewa barang pelanggan, melakukan pembukuan dan inventarisasi barang.
4	Admin	Merupakan penyedia layanan aplikasi yang dapat membuat akun pemilik agar pemilik dapat menggunakan aplikasi dan memvalidasi pembayaran sewa oleh pemilik.

Adapun kebutuhan fungsional dalam aplikasi ini ditujukkan pada tabel 2.

Tabel 2. Kebutuhan Fungsional Sistem

No.	Kode Kebutuhan Sistem	Deskripsi Kebutuhan
1	SRS-SB-F-1	Sistem dapat menyediakan fungsi *log in* bagi pengguna
2	SRS-SB-F-2	Sistem dapat menyediakan fungsi

		log out bagi admin, pemilik, dan staf
3	SRS-SB-F-3	Sistem dapat merekam riwayat pemesanan pada pelanggan
4	SRS-SB-F-4	Sistem dapat menyediakan fungsi bagi pemilik untuk membuat akun staf baru
5	SRS-SB-F-5	Sistem dapat menyediakan fungsi bagi pemilik untuk upload bukti pembayaran sewa aplikasi
6	SRS-SB-F-6	Sistem dapat melakukan penjadwalan barang pada barang yang akan keluar
7	SRS-SB-F-7	Sistem dapat merekam barang yang sering diminta oleh pelanggan
8	SRS-SB-F-8	Sistem dapat merekam informasi barang
9	SRS-SB-F-9	Sistem menyediakan fungsi untuk menambah pemeliharaan barang
10	SRS-SB-F-10	Sistem dapat menyediakan fungsi untuk melayani booking barang pelanggan
11	SRS-SB-F-11	Sistem dapat menyediakan fungsi untuk melayani pemesanan barang pelanggan
12	SRS-SB-F-12	Sistem dapat menyediakan fungsi untuk melayani pengambilan barang pelanggan
13	SRS-SB-F-13	Sistem dapat menyediakan fungsi untuk melayani pengembalian barang pelanggan
14	SRS-SB-F-14	Sistem dapat menghitung biaya tambahan sesuai dengan waktu keterlambatan pengembalian barang
15	SRS-SB-F-15	Sistem dapat menyediakan fungsi untuk memasukkan denda atau perbaikan sebagai penggantian barang yang rusak atau hilang
16	SRS-SB-F-16	Sistem dapat menyediakan fungsi untuk merekam pengeluaran pada jasa sewa
17	SRS-SB-F-17	Sistem dapat menyediakan fungsi untuk membuat laporan keuangan dengan jenjang waktu per hari, per bulan, dan per tahun
18	SRS-SB-F-18	Sistem dapat menyediakan fungsi untuk mengirim nota pemesanan ke email pelanggan
19	SRS-SB-F-19	Sistem dapat menyediakan fungsi bagi pemilik untuk membuat ketentuan promo
20	SRS-SB-F-20	Sistem dapat menyediakan fungsi bagi admin untuk membuat akun pemilik
21	SRS-SB-F-21	Sistem dapat menyediakan fungsi bagi admin untuk memvalidasi pembayaran sewa aplikasi
22	SRS-SB-F-22	Sistem hanya dapat menampilkan halaman pembayaran aplikasi pada akun pemilik ketika pemilik belum melakukan upload bukti pembayaran

		sewa hingga waktu jatuh tempo
23	SRS-SB-F-23	Sistem dapat mengirimkan notifikasi upload ulang bukti pembayaran sewa aplikasi kepada pemilik
24	SRS-SB-F-24	Sistem hanya dapat menampilkan halaman layanan tidak tersedia ketika pemilik belum melakukan upload bukti pembayaran sewa hingga waktu jatuh tempo pada akun staf
25	SRS-SB-F-25	Sistem dapat menghapus pemesanan barang pelanggan ketika barang tidak diambil hingga hari pengambilan barang habis

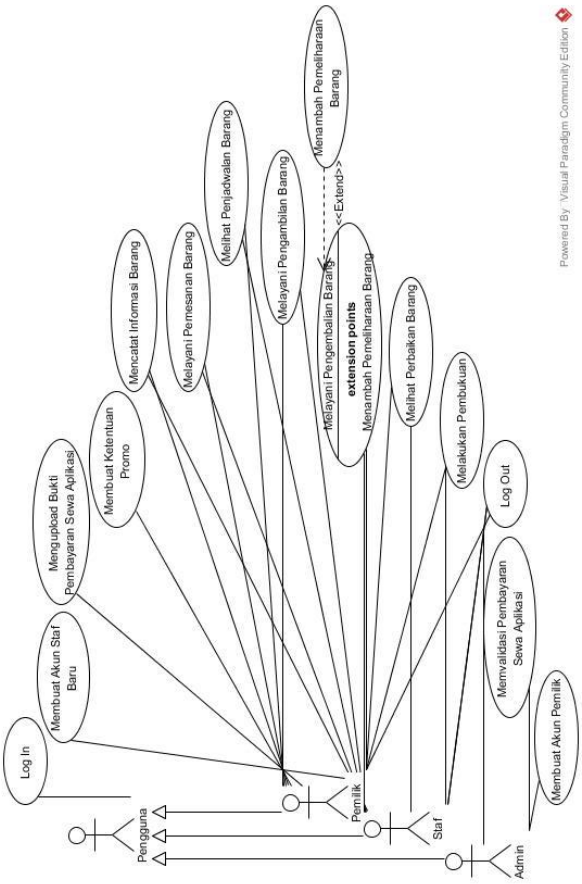
Adapun kebutuhan non fungsional dalam aplikasi ini ditujukkan pada Tabel 3.

Tabel 3. Kebutuhan Non Fungsional Sistem

No.	Kode Kebutuhan Sistem	Deskripsi Kebutuhan
1	SRS-SB-NF-1	*Availability* yaitu sistem dapat digunakan oleh pemilik dan staf dimana pun dan kapan saja

3.3. Diagram *Use case*

Diagram *use case* berisi sejumlah aksi yang dilakukan oleh aktor kepada sistem untuk mencapai tujuan tertentu. Diagram *use case* ditunjukkan dalam Gambar 3.



Gambar 3. Diagram *use case* sistem

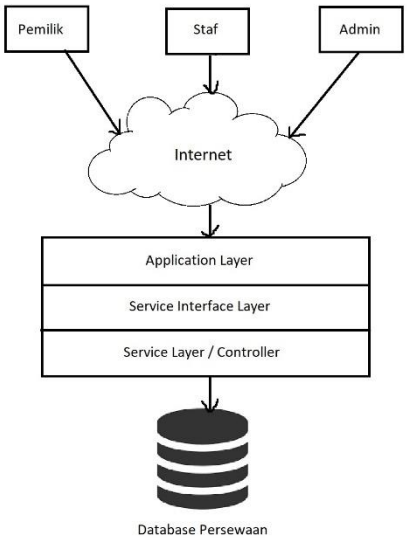
Pada Gambar 3. Diagram *use case* terdiri dari 15 *use case*, diantaranya *use case log in*, membuat akun staf baru, meng*upload* bukti pembayaran aplikasi, membuat ketentuan promo, mencatat informasi barang, melayani pemesanan barang, melihat penjadwalan barang, melayani pengambilan barang, melayani pengembalian barang, menambah pemeliharaan barang, melihat perbaikan barang, melihat pembukuan, log out, memvalidasi pembayaran sewa aplikasi, dan membuat akun pemilik. Pada *use case* menambah pemeliharaan barang mengextend *use case* melayani pengembalian barang, karena *use case* menambah pemeliharaan barang merupakan pilihan dari *use case* pengembalian barang, jadi bisa dilakukan atau tidak *use case* menambah pemeliharaan barang tersebut.

4. PERANCANGAN APLIKASI

4.1. Desain Arsitektur Aplikasi

Aplikasi ini dibuat menggunakan web *service*. Keuntungan dari penggunaan web *service* adalah dapat berjalan disemua *platform* dan diimplementasikan dengan bahasa

pemrograman yang berbeda (*interoperability*), komponen *service* dapat digunakan kembali (*reusabillity*), penyebaran informasi ke pengguna mudah, dan pengembangan cepat (Pirnau & Botezatu, 2016). Tipe web *service* yang digunakan adalah REST atau RESTful karena mudah untuk dikembangkan (Zhu, et al., 2013). Desain arsitektur aplikasi ditunjukkan pada Gambar 4.



Gambar 4. Desain arsitektur aplikasi

Pada Gambar 4. untuk mengakses aplikasi, maka perlu menggunakan internet, kemudian pemilik, staf, dan admin dapat mengakses aplikasi pada *application layer*. *Service interface layer* dibuat diantara *application layer* (*boundary*) dan *service layer* untuk menyaring masukan dari pengguna, agar tidak langsung mengakses ke *service layer* pada server. *Service layer* berisi logika bisnis pengelolaan jasa sewa, *service interface layer* berisi daftar layanan yang tersedia pada aplikasi, dan *application layer* berisi tampilan pengguna untuk menerima masukan dan mengirim permintaan *service* pengguna. Arsitektur *software* yang digunakan adalah 3 *tier*, karena terdiri dari 3 lapisan, dimana setiap lapisan memiliki tugas masing-masing. Pada lapisan pertama, bertanggung jawab untuk mengumpulkan informasi terkait pengelolaan sewa barang, pada lapisan kedua bertanggung jawab untuk mengirimkan informasi dari lapisan pertama ke lapisan ketiga dan melakukan seleksi informasi sebelum diteruskan ke lapisan ketiga. Pada lapisan ketiga, melakukan CRUD (*Create, Read, Update, Delete*) pada *database.*

4.2. Perancangan *Service*

Perancangan *service* mendefinisikan kandidat *service* apa saja yang akan disediakan oleh aplikasi ini. Setiap kandidat *service* memuat proses-proses didalamnya. Aplikasi ini menyediakan 6 kandidat *service*, yaitu Pemesanan Barang *Service*, Pengambilan Barang *Service*, Pengembalian Barang *Service*, Inventaris Barang *Service*, Manajemen Toko *Service*, dan Admin *Service*. Masing-masing kandidat *service* memiliki proses-proses atau sub *service* didalamnya.

4.3. Perancangan Arsitektur

Arsitektur web *service* digunakan dalam perancangan aplikasi ini, dimana web *service* yang digunakan adalah REST yang menggunakan fitur web sehingga *class* yang dirancang terdiri dari *class controller* (.php) dan *boundary* (.java). Pada *class controller* terdapat *class service interface layer* dan *service layer*. Pada *class service layer* berisi fungsi-fungsi seperti penghitungan, penjadwalan, pengiriman nota, dan fungsi lainnya yang melakukan proses untuk membantu pengguna, sedangkan *class boundary* membantu pengguna dalam penginputan atau penerimaan data lalu dikirimkan ke *service interface layer* untuk dilakukan seleksi dan pemilihan *service* sesuai dengan masukan pengguna pada *class boundary*. Oleh karena itu, setiap *class boundary* hanya terhubung dengan *class service interface layer* saja, kecuali pada *class boundary* logintoko, login*user*, mainfragmentberanda, dan mainfragmentadmin yang terhubung dengan kelas *controller* login langsung karena *login* bukan termasuk *service* dan tidak mengandung proses bisnis. Komunikasi antar *class* menggunakan json sehingga pengiriman dan permintaan data menggunakan json. Pada *class* diagram juga terdapat *class* tambahan yang terdiri dari *class adapter* untuk mendefinisikan isi dari recyclerView pada *boundary* dan *class main fragment* untuk menggabungkan *fragment* pada *boundary.*

4.3. Perancangan Komponen

Pada perancangan komponen, komponen yang dirancang diantaranya memuat algoritma pada *controller* (termasuk *service interface layer*) yang berada dalam file .php dan algoritma pada *boundary* serta *class* tambahan yang berada dalam file .java. Dalam perancangan komponen ini, dijabarkan semua

pseudocode pada *controller*, *boundary*, dan *class* tambahan. *Pseudocode* yang dibuat adalah *pseudocode* 12 *class controller*, 43 *class boundary* serta 12 *class* tambahan.

Pada *pseudocode class controller* sil.php yang merupakan *service interface layer* pada web *service* berisi *method-method* penunjang sub kandidat *service* pada kandidat *service* dan *controller* ini menyeleksi data json dari *boundary* untuk diteruskan ke *service* yang sesuai. Pada *service interface layer*, terdapat *method* detailBukti2 yang merupakan *service* admin untuk mengirim status dan id pemilik ke *controller* akun pemilik untuk dilakukan validasi bukti pembayaran sewa.

4.4. Perancangan Data

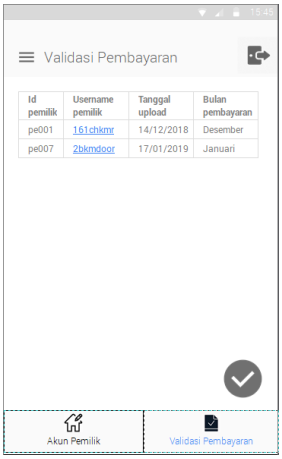
Pada perancangan data dibuat ERD, yang terdiri dari 19 tabel, dimana 4 diantaranya merupakan *weak entity* atau *entity* yang lemah karena tidak memiliki primary key pada atributnya. Tabel yang dirancang diantaranya yaitu tabel pesanan_pelanggan, pesanan_barang, toko, pemilik, staf, promo, isi_promo, jenis_barang, barang, permintaan barang, pembayaran, denda, perbaikan, pengiriman, bukti_pembayaran, penjadwalan, pemeliharaan, pembukuan, dan log untuk menyimpan data update tabel.

4.5. Perancangan Antarmuka

Pada perancangan antarmuka, dibuat *mock up* halaman yang terdiri dari 43 halaman, yaitu halaman *log in user*, *log in* toko, beranda, form jenis barang, detail jenis barang, detail barang, form barang, penjadwalan barang, pemesanan barang, manajemen toko, form pemesanan 1, form pemesanan 2, detail pesanan 1, detail pesanan 2, detail pesanan 3, form permintaan barang, permintaan barang, pengambilan barang, pengembalian barang, staf, form staf, detail staf, promo, form promo, form pembayaran sewa, pembayaran sewa, form pengeluaran, pembukuan harian, pembukuan bulanan, pembukuan tahunan, detail pembukuan harian, detail pembukuan bulanan, detail pembukuan tahunan, pemeliharaan barang, perbaikan barang, beranda admin, form pemilik, form toko, detail pemilik, validasi pembayaran, detail bukti pembayaran, pembayaran valid, dan halaman layanan tidak tersedia.

Pada Gambar 5. Merupakan halaman pemesanan barang yang berfungsi untuk menampilkan daftar pesanan pelanggan. Halaman pemesanan barang memuat recyclerView pesanan, menu pemesanan barang, manajemen toko, dan inventaris barang, serta imageView perbaikan barang, permintaan barang, *log out*, dan tambah pesanan. Halaman ini tampil ketika menu atau fragment pemesanan barang dipilih.

Gambar 5. Halaman antarmuka pemesanan barang



Gambar 6. Halaman antarmuka validasi pembayaran

Pada Gambar 6. Merupakan halaman validasi pembayaran admin yang berfungsi untuk menampilkan daftar bukti pembayaran sewa pemilik. Halaman validasi pembayaran memuat tableLayout semua bukti pembayaran dan imageView pembayaran valid dan *log out*. Halaman ini tampil ketika menu atau *fragment* validasi pembayaran dipilih. Halaman pembayaran valid dapat tampil ketika imageView pembayaran valid ditekan.

5. ANALISIS KUALITAS PERANCANGAN

5.1. Pengujian Dasar

Hasil pengujian menggunakan

Tracebility Matrix adalah setiap kolom dan baris pada traceability matrix memiliki nilai 1, yang berarti semua *use case* dapat dilacak ke semua artefak yang telah dibuat. *Use case* termasuk efektif karena semua *use case* menghasilkan artefak menurut baris dan *use case* tidak ada yang kehilangan sumber artefak atau berasal dari *requirement* menurut kolom (Kong & Yuan, 2009).

5.2. Pengujian Kualitas

Pengujian dilakukan menggunakan perhitungan *(Coupling Between Object Classes) CBO, (Response for a Class) RFC, (Lack of cohesion in methods1) LCOM1 metrics,* dan *(Lack of cohesion in methods2) LCOM2 metrics*.

Menurut Akwukwuma-Udo (2015), tingkat *adaptability* jika menggunakan metrik CBO, RFC, LCOM1, LCOM2 akan ditunjukkan pada Tabel 4.

Tabel 4. Tingkat *adaptability* berdasarkan metrik CBO, RFC, LCOM1, dan LCOM2

Metrik	***Adaptable***	***Fairly adaptable***	***Poorly adaptable***
CBO	1-3	4-5	>5
RFC	1-69	70-100	>100
LCOM1	1	-	>1
LCOM2	0-1	2	>2

Pada Tabel 4. tingkat *adaptability* dapat dibagi menjadi *adaptable, fairly adaptable* atau adaptable cukup, dan *poorly adaptable* atau *adaptable* buruk. Selain *adaptability*, nilai CBO dan RFC dapat menentukan tingkat kopling, dimana kopling rendah akan terjadi hanya ketika CBO<=5 dan RFC <=100, pada kondisi lainnya akan menghasilkan kopling yang tinggi (Akwukwuma & Udo, 2015). Kopling yang tinggi akan mengurangi *understandability* pada sistem karena akan menyebabkan modul sulit untuk dipahami, dirubah, atau benar dengan sendirinya jika saling berhubungan dengan modul lainnya (D, 2012). Pada nilai LCOM1 dan LCOM2, dapat menentukan tingkat kohesi, dimana kohesi tinggi akan terjadi hanya ketika LCOM1<=1 dan LCOM2<=2, pada kondisi lainnya akan menghasilkan kohesi yang rendah (Akwukwuma & Udo, 2015).

5.2.1 Penghitungan CBO dan RFC

Rata-rata nilai CBO = 2,166 berarti nilai CBO diantara 1 hingga 3, sedangkan rata-rata nilai RFC 11,166 berarti nilai RFC diantara 1 hingga 69, sehingga jika CBO diantara 1 hingga 3 dan RFC diantara 1 hingga 69 maka kopling=*adaptable* atau berarti hubungan antar modul sangat mudah dalam beradaptasi ketika ada perubahan dalam sistem (nilai *adaptability* tinggi). Semakin sedikit fungsi atau rendah nilai RFC maka semakin mudah dilakukan *testing* dan *debug class* karena memiliki kompleksitas yang rendah. CBO yang rendah berarti memiliki kopling yang rendah, sehingga dapat menambah kemudahan dalam pemahaman (*understandability* tinggi) karena modul mudah untuk dipahami dan dirubah.

Kopling yang dihasilkan termasuk rendah (rata-rata nilai CBO<=5 dan RFC<=100) karena mayoritas setiap *controller* hanya terhubung dengan *class controller* sil.php saja sehingga membantu mengurangi kopling.

5.2.2 Penghitungan LCOM1 dan LCOM2

Semua *class* (kecuali sil.php) memiliki tingkat *adaptability* tinggi (sangat mudah dalam beradaptasi) karena 2 *class* memenuhi LCOM1=1 dan LCOM2 0 hingga 1 dan 9 *class* memenuhi LCOM1>1 dan LCOM2 0 hingga 1. Pada 2 *class* memiliki kohesi yang tinggi karena LCOM<=1 dan LCOM2<=2 sedangkan 9 *class* lainnya memiliki kohesi yang rendah, karena memiliki kondisi lainnya, tidak memenuhi LCOM<=1, hanya memenuhi LCOM2<=2 saja. Semakin rendah nilai LCOM1, maka akan semakin tinggi kohesi sedangkan pada LCOM2, semakin rendah nilai maka semakin tinggi nilai *adaptability*. Pada *class* sil.php memiliki nilai LCOM2 yang tinggi, sehingga tingkat *adaptability* menjadi rendah dan kohesi rendah.

Mayoritas *class* memiliki LCOM1>1 sehingga nilai termasuk tinggi yang menyebabkan kohesi yang dihasilkan rendah. Hal ini dapat terjadi, karena terdapat fungsi tambahan yang kurang relevan pada *class* tersebut, sehingga fungsi tersebut dengan fungsi lainnya tidak memiliki hubungan, dengan alasan fungsi tambahan dimasukkan ke dalam *class* tersebut untuk tujuan efisiensi desain *class*.

6. KESIMPULAN

Kesimpulan yang didapatkan dari penelitian ini:

1. Hasil rekayasa kebutuhan pada aplikasi pengelolaan sewa barang ini adalah aplikasi pengelolaan sewa barang ini mempunyai 25 kebutuhan fungsional dan 1 kebutuhan non fungsional yang dapat membantu jasa sewa barang dalam mengelola jasa sewanya. Kemudian, dilakukan beberapa pemodelan kebutuhan untuk memudahkan pengembang dalam menggambarkan aplikasi, seperti *use case* diagram, *class* diagram, *package* diagram, dan *sequence* diagram.

2. Hasil rekayasa kebutuhan yang didefinisikan sesuai dengan permasalahan pada penyedia jasa sewa, karena penentuan kebutuhan fungsional diperoleh berdasarkan permasalahan terkait pemesanan barang, perbedaan pembukuan, dan stok barang. Hasil validasi dan verifikasi kebutuhan sesuai dengan permasalahan dan proses bisnis hasil elisitasi kebutuhan pada jasa sewa.

3. Hasil perancangan pada aplikasi pengelolaan sewa barang ini adalah diperoleh perancangan *service*, perancangan arsitektur, perancangan data, perancangan komponen, dan perancangan antarmuka. Dalam perancangan arsitektur terdapat rancangan *sequence* diagram, *class* diagram, dan *package* diagram yang dijelaskan dengan rinci. Pada perancangan data dari ER diagram hingga PDM (Physical Data Model). Pada perancangan komponen terdapat *pseudocode* dari *class controller* dan *service* yang akan digunakan pada aplikasi. Pada perancangan antarmuka berisi *mock-up* atau gambaran aplikasi yang akan dibangun dan skenario *prototype*.

4. Hasil perancangan aplikasi pengelolaan sewa barang ini memenuhi kebutuhan yang telah didefinisikan sebelumnya karena perancangan yang dibuat telah memuat kebutuhan-kebutuhan yang telah didefinisikan sebelumnya dan perancangan dibuat berdasarkan pemodelan kebutuhan pada rekayasa kebutuhan.

5. Kualitas hasil perancangan aplikasi pengelolaan sewa barang ini berdasarkan hasil pengujian dasar, dapat disimpulkan bahwa semua artefak dibuat berdasarkan *use case* yang telah dibuat sebelumnya karena setiap kolom dan baris pada *traceability matrix* memiliki nilai 1. Pada hasil pengujian kualitas dapat disimpulkan bahwa aplikasi memiliki kopling rendah karena rata-rata nilai CBO<=5 dan RFC<=100, kohesi rendah karena pada mayoritas *class* memiliki LCOM1>1, *understandability* tinggi karena kopling rendah, dan *adaptability* tinggi karena rata-rata nilai CBO diantara 1 hingga 3, sedangkan rata-rata nilai RFC diantara 1 hingga 69, serta semua *class* (kecuali sil.php) memenuhi nilai LCOM2 0 hingga 1.

7. DAFTAR PUSTAKA

Akwukwuma, V. V. N. & Udo, E. N., 2015. Predicting Adaptability of Object Oriented Software Using Metrics and Threshold Values. *The Pacific Journal of Science and Technology*, 16(2), pp. 124 - 134.

D, S., 2012. *Evaluation of Software Understandability Using Software Metrics*, Odisha: Department of Computer Science and Engineering National Institute of Technology, Rourkela.

Erl, T., 2017. *Service Oriented Architecture Analysis and Design for Services and Microservices.* 2nd ed. s.l.:Service Tech Press.

Kong, L. & Yuan, T., 2009. *Extension Features Driven Use Case Model for Requirement Traceability.* Harbin, IEEE.

Shelly, G. B. & Rosenblatt, H. J., 2012. *Systems Analysis and Design.* 9th ed. Boston: Course Technology.

Pressman, R. S., 2010. *Software Engineering : a practitioner's approach.* New York: McGraw-Hill.

Zhu, J., Cai, H. & Bu, F., 2013. *Identifying Restful Web Services in Service-Oriented Software Product Line.* Shanghai, IEEE.

Herrero, J. L., F., L. & P, C., 2011. *Web services and web components*. Badajoz, IEEE.

Pirnau, C. & Botezatu, M. A., 2016. Service-Oriented Architecture (SOA) and Web Services. *Database Systems Journal*, VII(4), pp. 32-39.