BAB III

PEMBAHASAN

3.1. Tinjauan Perusahaan

Hotel La Derra adalah perusaan yang bergerak dalam bidang jasa penyewaan kamar, adapun dalam latar belakang dari ruang lingkup kegiatan perusahaan, penulis akan menguraikan di bawah ini.

3.1.1. Sejarah Perusahaan

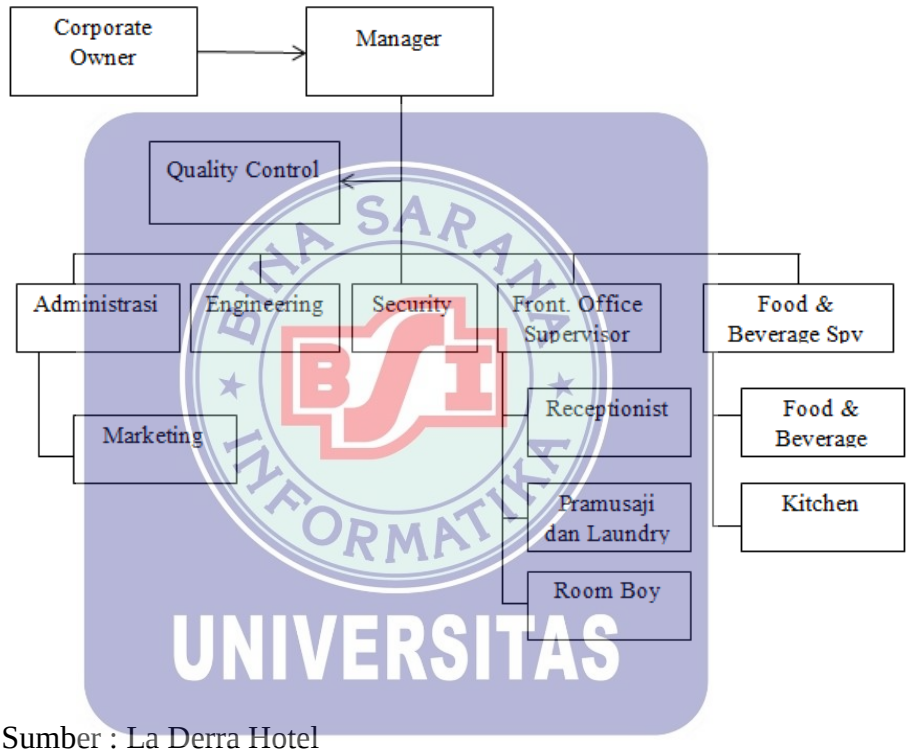
Hotel La Derra adalah salah satu hotel bintang satu di Kota Purwakarta, letaknya strategis berada di jantung kota Purwakarta tepatnya di jalan Ahmad Yani N0 5 Purwakarta, hanya 10 menit menuju pintu tol Jatiluhur. Selain letaknya strategis, Hotel La Derra juga merupakan hotel yang dekat dengan air mancur terbesar di Asia Tenggara (0,5 km) dan hanya 0,4 km jaraknya ke pusat pemerintahan kabupaten Purwakarta.

Hotel La Derra merupakan usaha perorangan yang dikelola oleh keluarga namun dikelola secara professional. Didirikan pada tanggal 23 februari 2013. Hotel La Derra memiliki kamar yang sangat reprsentatif dengan fasilitas AC, TV *cable* dan *hot watter*. Ruang *meeting, coffee shop, Mini market, Free parking* yang memadai. Untuk menjamin kenyamanan dan keamanan pengunjung di lokasi yang strategis dilengkapi dengan CCTV.

Demi mendukung mobilitas pengunjung selama menginap tersedia akses jaringan *wifi* di seluruh area hotel. Hotel La Derra Purwakarta adalah pilihan cerdas bagi pengunjung yg ingin menginap di hotel dengan harga terjangkaau, namun memberikan pelayanan terbaik. Menu makanan dapat disajikan atau *delivery* sesuai dengan pesanan.

3.1.2. Struktur Organisasi dan Fungsi

Struktur organisasi merupakan satu hal yang tidak bisa dipisahkan dari suatu perusahaan/instansi. Struktur organisasi sangat diperlukan untuk mencapai suatu tujuan dan menjadi penggerak suatu perusahaan karena berhubungan dengan suatu tanggung jawab yang saling berhubungan, sehingga tujuan organisasi dapat tercapai secara efektif . Adapun struktur organinsasi di Hotel La Derra.



Sumber : La Derra Hotel

Gambar III.1.
Struktur Organisasi

Berikut merupakan paparan mengenai penugasan-penugasan pada Hotel La Derra:

1. ***Manager***
   a. Mengatur dan meneiliti pemesanan, penerimaan, pelayanan kamar, dan kegiatan pengurus/pelayan hotel.
   b. Merencanakan dan mengawasi *coffe shop*, *restaurant*, kamar dan ruangan *meeeting.*

c. Memeriksa pembukuan dan kegiatan pembelian.

d. Memberi persetujuan penggunaan brankas oleh *front desk agent.*

e. Menetapkan pembuatan anggaran kebutuan hotel.

f. Memverifikasi setiap pengadaan dan pembelian barang.

g. Mengawasi dan monitor terhadaf *staff.*

h. Memastikan terpenuhinya standar K3 sesuai dengan peraturan perundangan-undangan.

i. Menyediakan informasi wisata lokal dan mengatur transportasi untuk kunjungan atau wisata kepada tamu.

j. Membuat jadwal shift pegawai.

k. Melaksanakan fungsi marketing.

l. Melaporkan secara periodik tentang aktifitas hotel kepada *owner*.

m. Menyusun RKAP tahunan.

n. Mengatur sumber daya yang dimiliki hotel.

o. Mendistribusikan rencana kerja yang ada.

2. *Quality Control*

a. Menilai dan memeriksa kepuasan tamu.

b. Mengawasi persiapan keamanan, kebun dan pemeliharaan barang-barang.

c. Adanya pemisah fungsi antara penjualan, penerimaan dan pelaporan atas penjualan.

d. Kamar dapat dipesan apabila kamar tersedia pada waktu yang diinginkan, disetujui oleh *front office.*

e. Setiap transaksi dicatat pada kartu buku reservasi.

f. Pengecekan kamar dilakukan secara rutin beserta fasilitasnya.

g. Pemisahan tugas antara pembuat daftar gaji dan fungsi kepegawaian.

h. Tarif upah diverifikasi oleh fungsi akuntansi.

i. Pembuatan daftar gaji dan upah diverifikasi.

3. *Front Office Supervisor*

a. Membantu tugas *front office* melakukan tugas operasional di *front office.*

b. Mengarahkan tugas operasional penerimaan tamu di *front office.*

c. Menangani keluhan tamu yang tidak bisa diatasi oleh *front desk agent.*

d. Memberi persetujuan transaksi *paid out* tamu untuk jumlah tertentu.

e. Memberi persetujuan penggunaan *housebank* oleh *front desk agent.*

f. Mengarahkan langkah persiapan penerimaan tamu grup.

g. Mengatur jadwal setiap staff *front desk.*

h. Mendukung kelancaran proses *check-in* dan *check-out* di *front office.*

i. Menangani kesulitan tamu dan staff di *front desk.*

j. Mengontrol operasional di seputar *front office* antara lain *lobby, restaurant*, koridor dan kamar.

k. Membuat laporan setiaf shift tentang temuan dan kejadian selama jam kerjanya.

l. Menyambut tamu *VIP* bersama dengan *front office manager*.

4. *Food and Beverage Superevisor*

a. Menetapkan menu masakan dan minuman sesuai dengan daftar menu yang sudah ditentukan.

b. Mengontrol operasional di seputar dapur dan kedai.

c. Memastikan bahwa pembayaran dan pencatatan pembayaran.

d. Melayani tamu atas segala macam keperluan yang mereka butuhkan selama menginap.

e. Menyeimbangkan *account* kas.

f. Melayani permintaan tamu mengenai kelengkapan kamar.

g. Menjaga keamanan dan kenyaman lingkungan *front desk* dan kedai.

h. Melakukan transaksi penjualan kedai.

i. Mendapat laporan kegiatan tugas *food and beverage.*

5. Administrasi

a. Pastikan bahwa pembayaran dan pencatatan pembayaran.

b. Pastikan semua kas dan setara kas di catat dan diseimbangkan pada awal dan akhir setiap akhir shift kerja.

c. Memverifikasi kredit pelanggan dan menetapkan bagaimana pelanggan dapat membayar akomodasinya.

d. Menerima pembayaran untuk account.

e. Menyeimbangkan account kas.

f. Memiliki tanggung jawab terhadap administrasi.

g. Meninjau rekening tamu yang check-out.

6. *Marketing*

a. Pastikan bahwa pembayaran dan pencatatan pembayaran.

b. Pastikan semua kas dan setara kas di catat dan diseimbangkan pada awal dan akhir setiap akir shift kerja.

c. Memverifikasi kredit pelanggan dan menetapkan bagaimana pelanggan dapat membayar akomodasinya.

d. Menerima pembayaran untuk account.

e. Menyeimbangkan account kas.

f. Meninjau rekening tamu yang check-out.

7. *Receptionist*

a. Menyambut tamu yang datang di *front desk* dengan ramah dan sopan.

b. Menyampaikan pesan yang ingin disampaikan.

c. Menjawab setiap pertanyaan yang diajukan tamu perusahaan.

d. Memberikan informasi setiap pertanyaan yang disampaikan di telepon.

e. Menyapa setiap tamu yang hadir.

f. Melakukan pendaftaran tamu.

g. Mencatat uang deposit tamu untuk menginap.

h. Menangani tamu *check-out.*

i. Memasukkan dan bertanggung jawab atas uang transaksi selama bertugas.

j. Mengitung dan bertanggung jawab atas uang transaksi selama bertugas .

k. Menjawab telepon yang masuk baik dari internal maupun external mentransfer langsung ke kamar yang dituju.

l. Melayani menyambungkan telepon baik dari tamu atau dari kamar.

8. *Food and Beverage*

a. Sebagai *sales promotion.*

b. Melayani tamu atas segala macam keperluan yang mereka butuhkan selama menginap.

c. Melayani permintaan tamu mengenai kelengkapan kamar.

d. Menjaga keamanan dan kenyamanan lingkungan *front desk* dan kedai.

e. Membersihkan ruang *front desk* dan kedai.

f. Melakukan transaksi penjualan kedai.

g. Melaporakan kegiatan tugas *food and beverage* kepada *supervisor.*

9. *Engineering*

a. Melakukan pemeliharaan terkait asset hotel.

b. Melakukan perbaikan jika ada kerusakan.

c. Memperbaiki atau menjaga kondisi kamar agar dapat dijual.

d. Menjalankan operasional kebutuhan hotel antara lain internet, listrik, air, dan telepon.

10. *Kitchen*

a. Memilih sumber makanan dan sayuran yang fresh dan higienis.

b. Memasak makanan yang sesuai orderan tamu.

c. Menyajikan makanan dan minuman.

d. Melakukan inventarisasi bahan makanan dan minuman.

e. Membuat transaksi pembelian dan penjualan makanan dan minuman.

f. Melaporkan kebutuhan bahan makanan dan minuman.

11. *Room boy*

a. Melayani tamu atas segala macam keperluan yang mereka butuhkan selama menginap.

b. Mengelola kamar-kamar dalam arti menjaga kebersihan, kebersihan, kenyamanan, serta kelengkapan fasilitas kamar.

c. Menjaga keamanan barang-barang milik hotel yang ada di dalam kamar jangan sampai hilang atau rusak.

d. Melaporkan kepada atasan bila mendapati alat-alat yang rusak agar dibuatkan *Work Order* (WO) kepada bagian engginering untuk segera diperbaiki.

e. Menjaga kebersihan kamar tamu.

f. Melayani permintaan tamu mengenai kelengkapan kamar.

g. Melayani permintaan tamu mengenai kelengkapan kamar.

h. Menjaga keamanan dan kenyamanan lingkungan Hotel.

i. Membersihkan teras dan halaman.

j. Menyiram tanaman.

k. Melaporkan kegiatan tugas malam kepada supervisor.

l. Membersihkan ruang *front desk* dan kedai.

12. Pramusaji dan *Laundry*

a. Melayani dan mengantar makanan dan minuman pagi hari.

b. Mengelola kamar-kamar dalam arti menjaga kebersihan, kebersihan, kenyamanan, serta kelengkapan fasilitas kamar.

c. Menjaga keamanan barang-barang milik hotel yang ada di dalam kamar jangan sampai hilang atau rusak.

d. Melakukan tugas *laundry* (Cuci dan Gosok).

e. Menjaga kebersihan kamar tamu pada saat piket siang.

f. Melayani permintaan tamu mengenai kelengkapan kamar.

13. *Security*

a. Melayani tamu atas segala macam keperluan yang mereka butuhkan selama menginap.

b. Melayani permintaan tamu mengenai kelengkapan kamar.

c. Mencatat Nomor kendaraan Tamu yang menginap di Hotel.

d. Mengatur Parkir Kendaraan Tamu Hotel.

e. Menjaga keamanan dan kenyamanan lingkungan hotel.

f. Menjaga kelancaran keluar masuk kendaraan Tamu hotel.

g. Membersihkan teras dan halaman.

h. Menyiram tanaman.

i. Melaporkan kegiatan tugas malam kepada supervisor.

j. Melayani tamu atas segala macam keperluan yang mereka butuhkan selama menginap.
k. Mengelola kamar-kamar dalam arti menjaga kebersihan, kebersihan, kenyamanan, serta kelengkapan fasilitas kamar.
l. Menjaga keamanan barang-barang milik hotel yang ada di dalam kamar jangan sampai hilang atau rusak.
m. Melaporkan kepada Supervisor bila mendapati alat-alat yang rusak agar dibuatkan *Work Order* (WO) kepada bagian enginering untuk segera diperbaiki.
n. Menjaga kebersihan kamar tamu.
o. Melaporkan kepada Supervisor bila mendapati alat-alat yang rusak agar dibuatkan *Work Order* (WO) kepada bagian engginering untuk segera diperbaiki.
p. Menjaga kebersihan kamar tamu.
q. Melayani permintaan tamu mengenai kelengkapan kamar

3.2. Tinjauan Kasus

3.2.1. Proses Bisnis Sistem Berjalan

Suatu prosedur atau tahap-tahap yang dilakukan sebelum memulai kegiatan untuk menyelesaikan suatu pekerjaan disebut prosedur sistem. Sesuai dengan ruang lingkup yang dibahas. Dalam penyusunan tugas ini, maka prosedur sistem berjalan yang diambil yaitu dari sistem penerimaan kas dan pengeluaran kas pada Hotel La Derra, Adapun prosesnya sebagai berikut:

1. Proses Pemesanan

   Resepsionis menerima tamu pada saat *check-in* lalu resepsionis akan menginput ke dalam *reservation list* (jika tamu melakukan *reservation by phone*) yang terdapat di dalam *Front office System (FOS).* Jika tamu yang datang secara langsung ke hotel *(walk in guest)* akan di input ke dalam *walk in*. Resepsionis menginput data tamu menurut kartu identitas tamu yang diberikan (KTP, SIM, atau passport) ke dalam *guest profile.*

2. Proses Pembayaran

   Pada saat tamu melakukan *check-in* dan akan membayar, resepsionis akan mempersiapkan *cash receipt* jika pembayaran dilakukan dengan uang tunai, jika pembayaran menggunakan *credit card,* resepsionis akan menginput ke dalam *credit card list*. Lalu resepsionis akan mencetak *guest bill* 1 rangkap untuk diberikan kepada tamu.

3. Proses Penerimaan Kas

   Setelah menerima pembayaran secara tunai dari tamu, setiap akhir *shift front office* akan memasukkan seluruh setoran tersebut ke dalam amplop *remittance of funddan* diberikan kterangan sejumlah setoran tersebut. Jumlah rupiah yang berada di amplop tersebut harus sesuai dengan laporan *night audit-cash report* yang tertera di *Front office System (FOS).* Pagi hari berikutnya Admin akan mengambil amplop *remittance of funddan* tersebut ke bagian resepsionis dan mencocokan dengan laporan yang tertera di *night audit-cash report* lalu dilakukan verifikasi ulang. Setelah terverifikasi admin akan menyerahkan laporan pendapatan ke manajer keuangan.

4. Proses Pengeluaran Kas

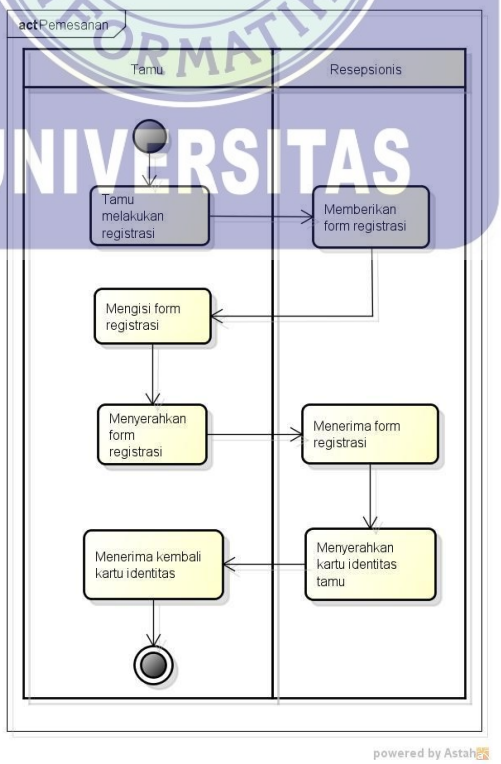
   Admin melakukan permintaan pengeluaran kas kepada manajer keuangan guna melakukan pembayaran rutin atau pengeluaran untuk pembelian barang dan kebutuhan sejenisnya dengan menyertakan bukti pengajuan pengeluaran kas, manajer keuangan akan melakukan pengajuan yang disertakan dengan bukti pengeluaran kas kepada owner untuk menyetujui sejumlah pengeluaran. Manajer keuangan mengeluarkan dana tunai yang diminta dan diberikannya kepada admin.

5. Proses Laporan

   Pendapatan dan pengeluaran yang sudah diverifikasi oleh admin lalu diberikan kepada manager keuangan dan diverifikasi kembali agar tidak terjadi kesalahan pada saat akan disetorkan kepada *owner* setiap bulannya.

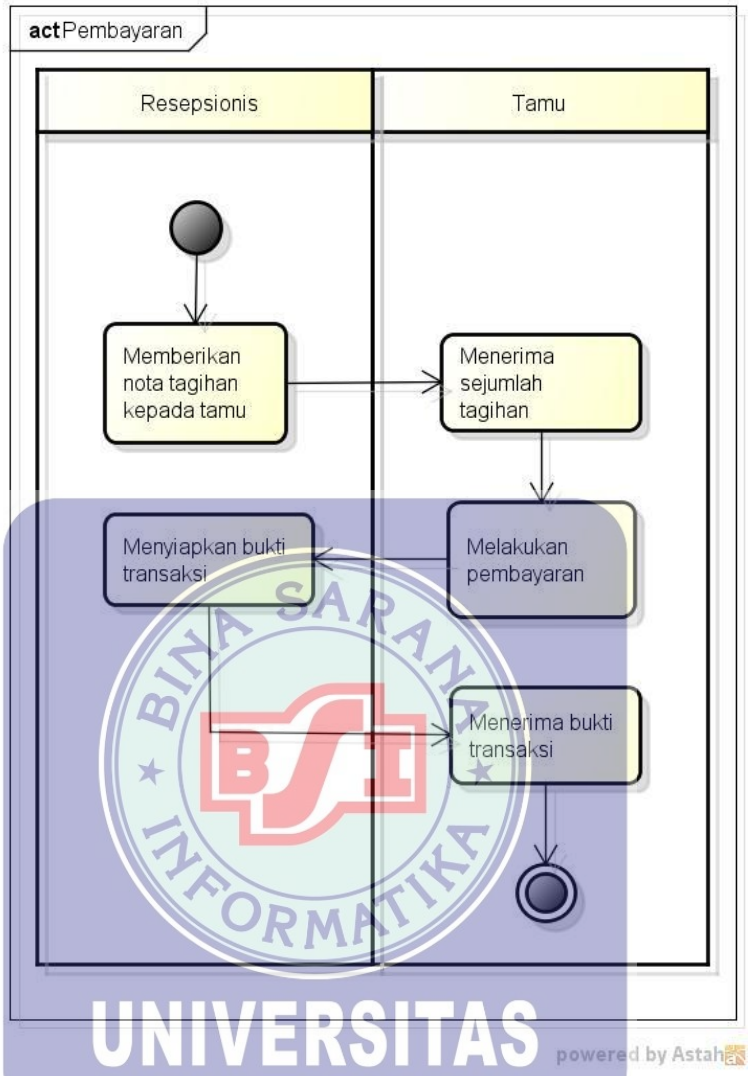
3.2.2. *Activity Diagram*

1. *Activity Diagram* Proses Pemesanan



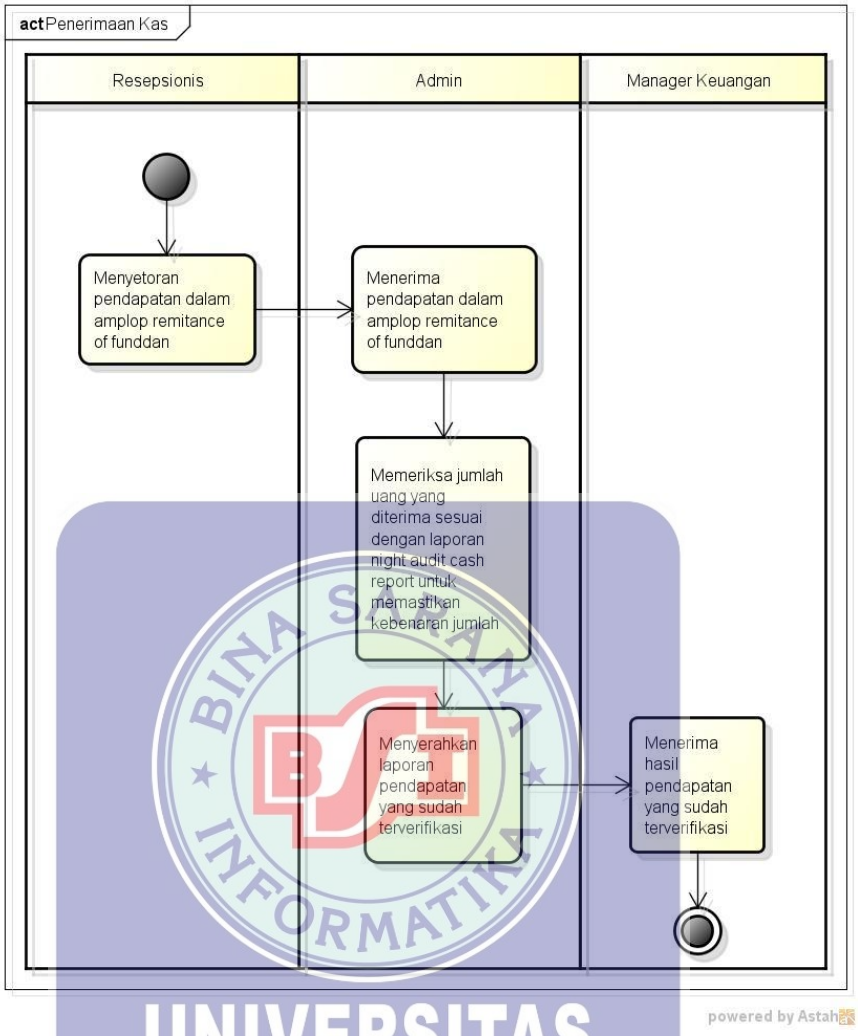
Gambar III.2.
***Activity Diagram* Proses Pemesanan**

2. *Activity Diagram* Proses Pembayaran



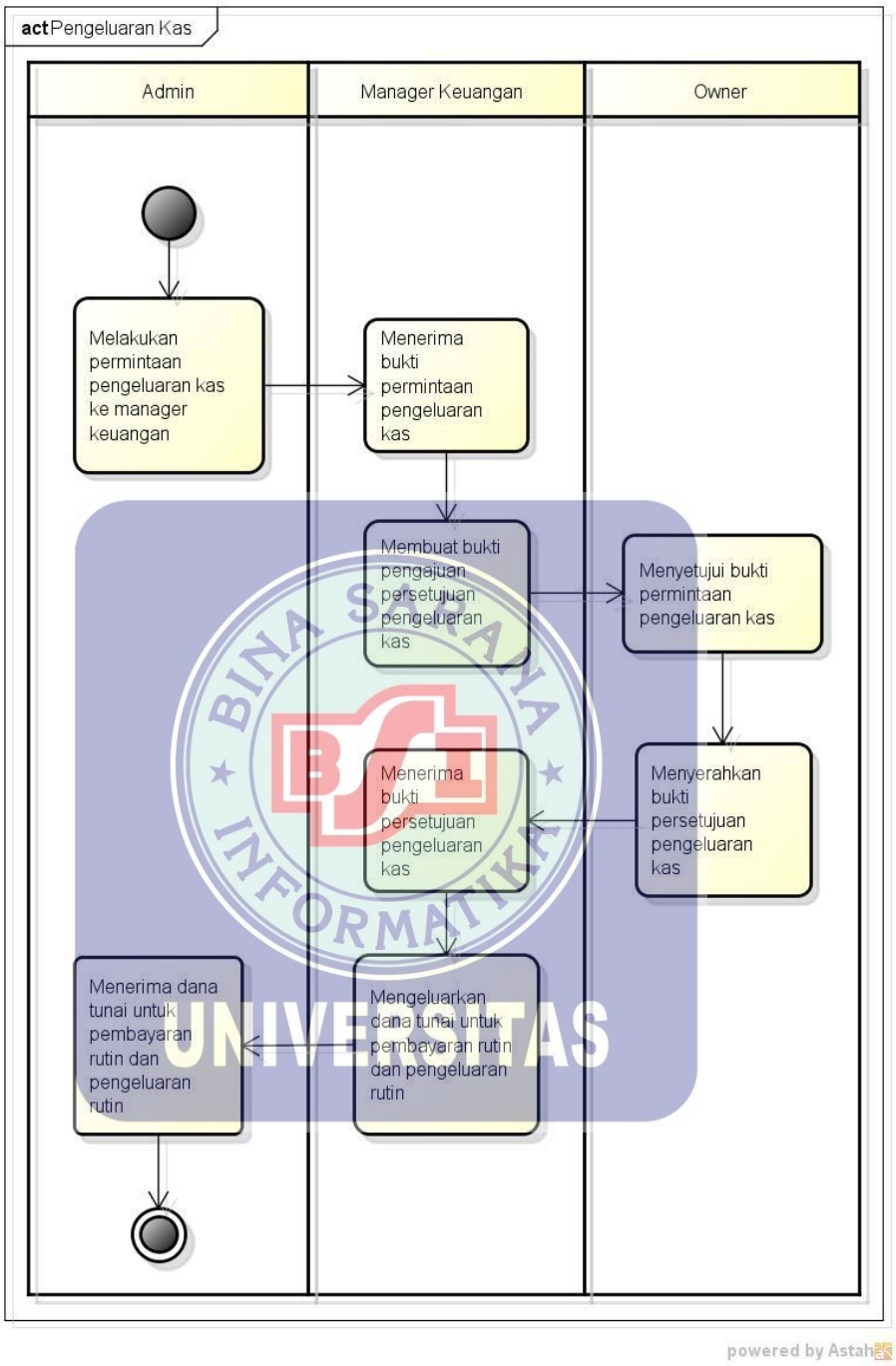
Gambar III.3.
***Activity Diagram* Proses Pembayaran**

3. *Activity Diagram* Proses Penerimaan Kas



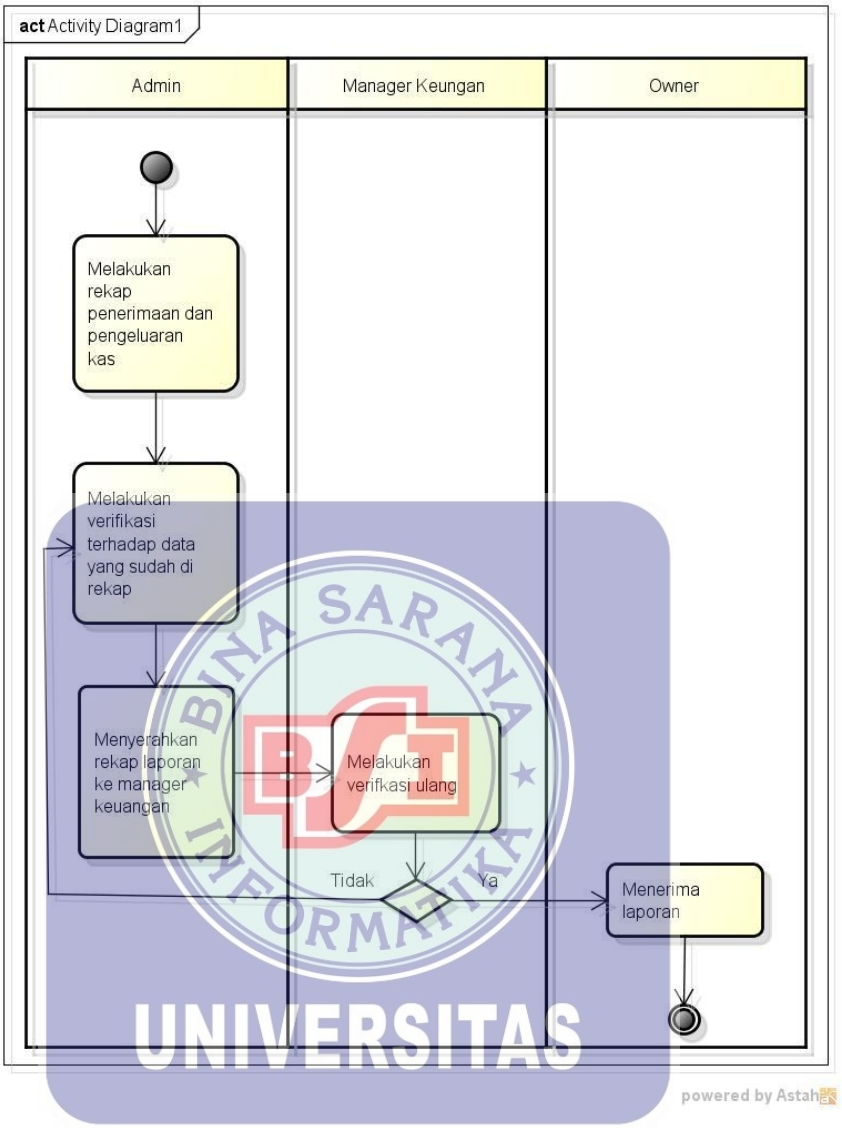
Gambar III.4.
***Activity Diagram* Proses Penerimaan Kas**

4. *Activity Diagram* Proses Pengeluaran Kas



Gambar III.5.
***Activity Diagram* Proses Pengeluaran Kas**

5. *Activity Diagram* Proses Laporan



Gambar III.6.
***Activity Diagram* Proses Laporan**

3.2.3. Spesifikasi Bentuk Dokumen Masukan

1. Nama Dokumen : Kartu Identitas

   Fungsi : Untuk melakukan registrasi *check-in*.

   Sumber : Tamu

   Tujuan : *Receptionist*

   Media : Kertass

Jumlah : 1 Lembar

Frekuensi : Setiap kali akan melakukan *check-in*.

Bentuk : Lihat lampiran A-1

2. Nama Dokumen : Form Registrasi

Fungsi : Untuk melakukan registrasi *check-in*.

Sumber : Tamu

Tujuan : *Receptionist*

Media : Kertas

Jumlah : 1 Lembar

Frekuensi : Setiap kali akan melakukan *check-in*.

Bentuk : Lihat lampiran A-2

3. Nama Dokumen : Bukti permintaan pengeluaran kas

Fungsi : Sebagai permintaan pengeluaran kas

Sumber : Admin

Tujuan : Manajer keuangan

Media : Kertas

Jumlah : 1 lembar

Frekuensi : Setiap ada permintaan

Bentuk : Lihat lampiran A-3

3.2.4. Spesifikasi Bentuk Dokumen Keluaran

1. Nama Dokumen : *Guest bill*

Fungsi : Sebagai bukti tagihan untuk tamu.

Sumber : Kasir

Tujuan : Tamu

Media : Kertas

Jumlah : 1 Lembar

Frekuensi : Setiap kali ada pembayaran tagihan

Bentuk : Lihat lampiran B-1

2. Nama Dokumen : Laporan penerimaan

Fungsi : Sebagai rekap penerimaan harian

Sumber : Kasir

Tujuan : Admin

Media : Kertas

Jumlah : 1 Lembar

Frekuensi : Setiap hari

Bentuk : Lihat lampiran B-2

3. Nama Dokumen : Pengeluaran harian

Fungsi : Sebagai pencatatan pengeluaran harian masing-masing divisi

Sumber : Manajer keuangan

Tujuan : Admin

Media : File Ms. Excel

4. Nama Dokumen : Laporan pengeluaran

Fungsi : Sebagai rekap pengeluaran kas bulanan

Sumber : Admin

Tujuan : Owner

Media : Kertas

Jumlah : -

Frekuensi : Setiap akhir bulan

Bentuk : File Ms. Excel

3.2.5. Permasalahan Pokok

Setiap proses bisnis dari sebuah perusahaan tidak akan lepas dari permasalahan yang dihadapi begitu juga pada Hotel La Derra ini. Penulis mencatat beberapa permasalahan pokok yang dihadapi terutama dalam sistem informasinya antara lain :

1. Dalam proses reservasi tamu yang chck-in dan check-out belum menggunakan sistem yang terintegrasi dengan baik.
2. Pencatatan pendapatan dan pengeluaran Hotel masih mengandalkan microsoft excel untuk rekapitulasinya sehingga dapat memakan waktu lebih lama.
3. Rentan terhadap manipulasi karena tidak adanya pembatasan akses dan dokumen tidak terlindungi.

3.2.6. Pemecahan Masalah

Permasalahan yang dihadapi oleh Hotel La Derra memerlukan solusi untuk meminimalisasi atau bahkan memcahkan masalah yang ada. Adapun pemecahan yang diusulkan penulis antara lain :

1. Perlu dibuatnya sebua sistem untuk mencatat data *check-in dan check-out* dan ketersediaan kamar di Hotel.
2. Perlu dibuatkan sebuah aplikasi untuk mencatat pendapatan dan pengeluaran Hotel yang lebih baik dan terintegrasi dengan sistem check-in dan check-out.
3. Membuat sebuah sistem dengan membatasi hak akses masing-masing pengguna untuk meminimalisasi atau menghapus manipulasi data.

3.3. Analisis Kebutuhan *Software*

3.3.1. Analisis Kebutuhan

Sebuah sistem dapat berjalan dengan baik jika dalam analisa kebutuhan saat perancangan sistem tepat sesuai permasalahan yang dihadapi, analisa ini meliputi perangkat lunak digunakan hingga kebutuhan pengguna. Berikut analisa kebutuhan berdasarkan kebutuhan pengguna:

A. Bagian Admin

A1. Admin berhak melakukan *login*

A2. Admin berhak mengelola kas msauk

A3. Admin berhak mengelola kas keluar

A4. Admin berhak mengelola perkiraan

A5. Admin berhak membuat laporan

B. Bagian Manajer Keuangan

B1. Manajer berhak melakukan *login*

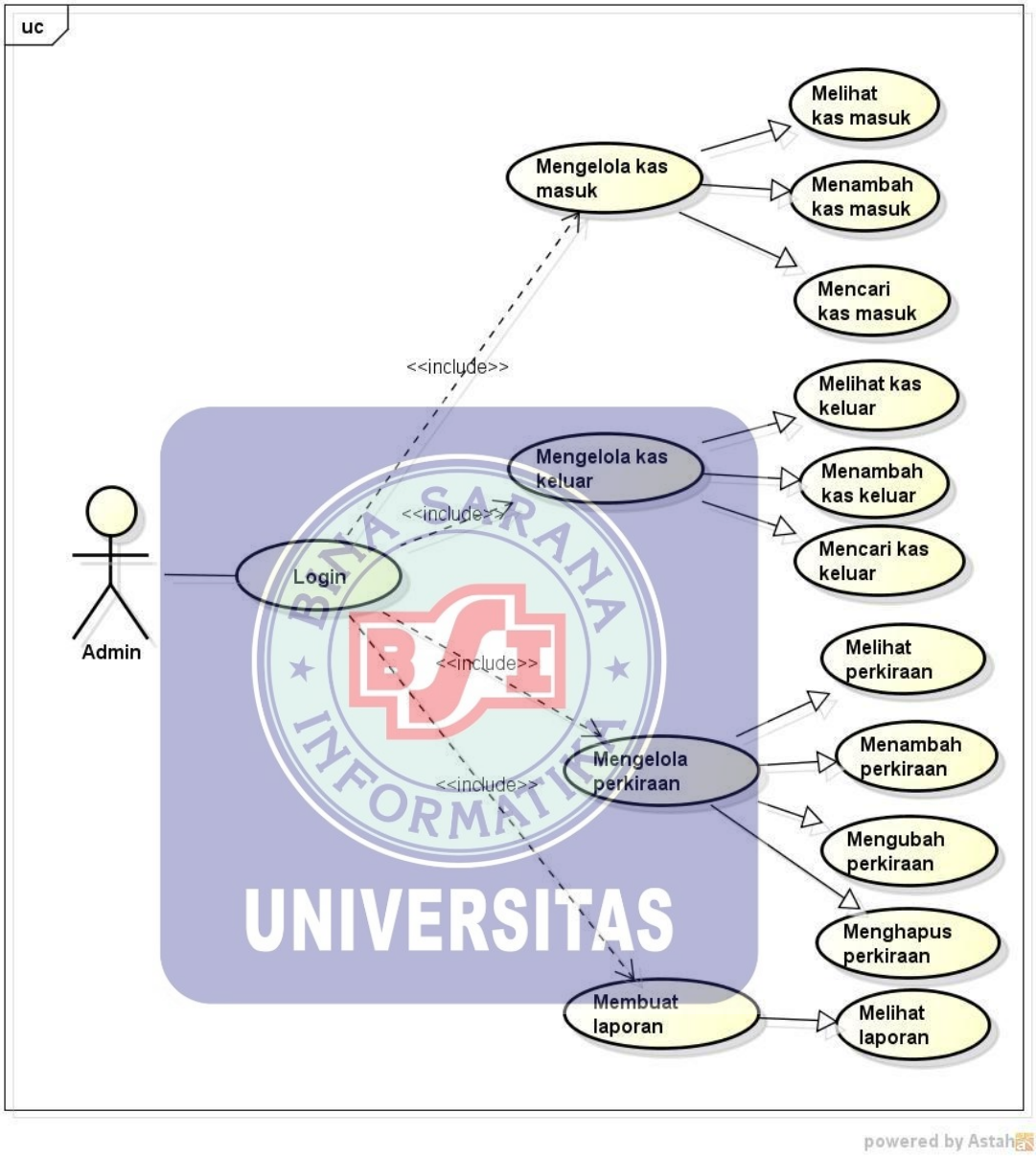
B2. Manajer berhak mengelola pengguna

B3. Manajer berhak mengakses laporan

B4. Manajer berhak membuat daftar jasa

3.3.2. *Usecase Diagram*

1. Deskripsi *Usecase Diagram* Bagian Admin



Gambar III.7.
***Usecase Diagram* Bagian Admin**

Tabel III.1.
Deskripsi *Usecase Diagram* Melakukan *Form Login*

Aksi Aktor	Reaksi sistem
Skenario Normal	
1. Membuka aplikasi *login*	
	2. Menampilkan *form login*
3. Menginput *Id.* Pengguna dan *Password* yang benar	
4. Klik tombol *login*	
	5. Memeriksa validasi *login*
	6. *Id*. Pengguna dan *Password* benar
	7. Masuk ke menu utama
Skenario Alternatif	
1. Membuka aplikasi *login*	
	2. Menampilkan form *login*
3. Menginput *Id.* Pengguna dan *password* yang salah	
4. Klik tombol login	
	5. Memeriksa validasi *login*
	6. *Id*. Pengguna dan *Password* salah
	7. Memeriksa validasi sistem
	8. Sistem dan *Password* salah
	9. Menampilkan pesan bahwa data masukkan tidak valid
10. Menginput ulang *Id.* Pengguna dan *Password*	
11. Klik tombol *login*	
	12. Validasi login
	13. *Id.* Pengguna dan *Password* benar
	14. Masuk ke menu utama

Tabel III.2.
Deskripsi *Usecase Diagram* Menambah *Form* Kas Masuk

Aksi Aktor	Reaksi Sistem
Skenario Normal	
1. Klik tambah	
	2. Membuat no. kas masuk baru
3. Memasukkan data kas masuk sesuai dengan data	
4. Klilk simpan	
	5. Memeriksa valid tidaknya data masukan
	6. Menyimpan kas masuk ke basis data
	7. Menampilkan kas masuk yang disimpan
Skenario Alternatif	
1. Klik tambah	
	2. Membuat no. kas masuk baru
3. Memasukkan data kas masuk yang tidak lengkap	
4. Klik simpan	
	5. Memeriksa validasi data
	6. Menampilkan pesan bahwa data masukan tidak valid
7. Menginput ulang data kas masuk sesuai dengan data	
8. Klik tombol simpan	
	9. Memeriksa valid tidaknya data masukan
	10. Menyimpan kas masuk ke basis data
	11. Menampilkas kas masuk yang disimpan

Tabel III.3.
Deskripsi *Usecase Diagram* Menambah *Form* Kas keluar

Aksi Aktor	Reaksi Sistem
Skenario Normal	
1. Klik simpan	
	2. Membuat no. kas keluar baru
3. Memasukkan data kas keluar sesuai dengan data	
4. Klilk simpan	
	5. Memeriksa valid tidaknya data masukan
	6. Menyimpan kas keluar ke basis data
	7. Menampilkan kas keluar yang disimpan
Skenario Alternatif	
1. Klik tambah	
	2. Membuat no. Kas keluar baru
3. Memasukkan data kas keluar yang tidak lengkap	
4. Klik simpan	
	5. Memeriksa data
	6. Data tidak valid dan menampilkan pesan
7. Menginput ulang data kas keluar sesuai dengan data	
8. Klik tombol simpan	
	9. Memeriksa valid tidaknya data masukan
	10. Menyimpan kas keluar ke basis data
	11. Menampilkas kas keluar yang disimpan

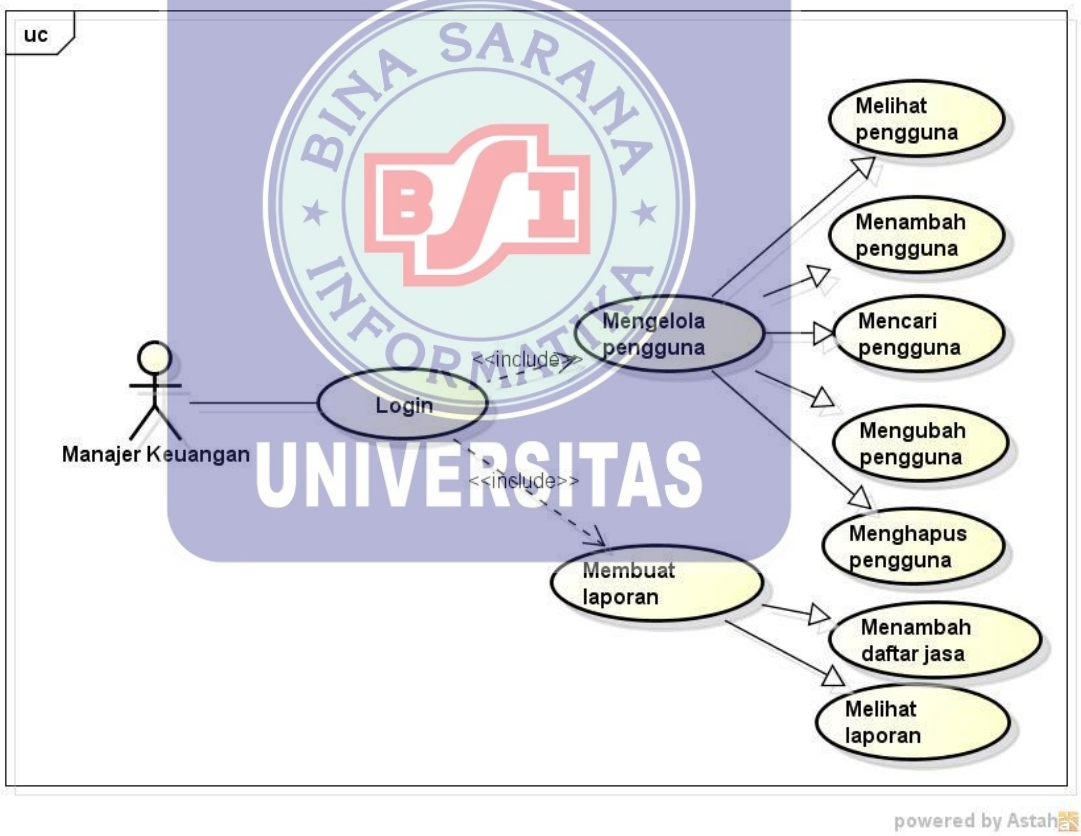
Tabel III.4.
Deskripsi *Usecase Diagram* Menambah *Form* Perkiraan

Aksi Aktor	Reaksi Sistem
Skenario Normal	
1. Klik tambah	
	2. Membuat kode akun baru
3. Memasukka kode akun sesuai dengan data	
4. Klik simpan	
	5. Memeriksa valid tidaknya data masukan
	6. Menyimpan kode akun ke basis data
	7. Menampilkan kode akun yang di simpan
Skenario Alternatif	
1. Klik tambah	
	2. Membuat kode akun baru
3. Memasukkan kode akun sesuai dengan data	
4. Klik simpan	
	5. Memeriksa data
	6. Data tidak valid dan menampilkan pesan
7. Menginput ulang kode akun	
8. Klik tombol simpan	
	9. Memeriksa valid tidaknya data masukan
	10. Menyimpan kode akun ke basis data
	11. Menampilkas kode akun yang disimpan

Tabel III.5.
Deskripsi *Usecase Diagram* Menghapus *Form* Perkiraan

Aksi Aktor	Reaksi Sistem
Skenario Normal	
1. Memilih data	
2. Klik hapus	
	3. Menampilkan konfirmasi apakah data akan benar-benar di hapus
4. Pilih Ya	
	5. Menghapus data perkiraan dari basis data
	6. Menampilkan pesan bahwa data sukse di hapus

2. Deskripsi *Usecase Diagram* Bagian *Manager* Keuangan



Gambar III.8.
***Usecase Diagram* Bagian *Manager* Keuangan**

Tabel III.6.
Deskripsi *Usecase Diagram* Melihat *Form* Data Pengguna

Aksi Aktor	Reaksi Sistem
Skenario Normal	
	1. Memeriksa status *login*
	2. Menampilkan data pengguna yang dicari
3. Memilih *id.* Pengguna yang dicari	
	4. Menampilkan data pengguna

Tabel III.7.
Deskripsi *Usecase Diagram* Menambah *Form* Data Pengguna

Aksi Aktor	Reaksi Sistem
Skenario Normal	
1. Klik tambah	
	2. Membuat id. Pengguna baru
3. Memasukkan id. Pengguna sesuai dengan data	
4. Klik simpan	
	5. Memeriksa valid tidaknya data masukan
	6. Menyimpan data id. Pengguna ke basis data
	7. Menampilkan pesan suskses disimpan
Skenario Alternatif	
1. Klik tambah	
	2. Membuat id. Pengguna baru
3. Memasukkan id. Pengguna sesuai dengan data	
4. Klik simpan	
	5. Memeriksa data

	6. Data tidak valid dan menampilkan pesan
7. Menginput ulang id. Pengguna	
8. Klik tombol simpan	
	9. Memeriksa vali tidaknya data masukan
	10. Menyiman id. Pengguna ke basis data
	11. Menampilkan id. Pengguna yang disimpan

Tabel III.8.
Deskripsi *Usecase Diagram* Mencari Data Pengguna

Aksi Aktor	Reaksi Sistem
Skenario Normal	
1. Memilih data	
2. Masukkan id.pengguna yang akan di cari	
3. Klik cari	
	4. Mencari data id. Pengguna yang akan di cari
	5. Menampilkan data id. Pengguna yang di cari

Tabel III.9.
Deskripsi *Usecase Diagram* Mengubah Data Pengguna

Aksi Aktor	Reaksi Sistem
Skenario Normal	
1. Memilih data	
2. Masukkan id. Pengguna yang akan di ubah	
3. Klik ubah	
	4. Mencari data id. Pengguna yang akan di ubah

	5. Menampilkan data id. Pengguna yang di ubah
6. Memilih data id. Pengguna yang akan di ubah	
	7. Menampilkan semua kolom data id. Pengguna yang akan di ubah
Skenario Alternatif	
1. Memilih data	
2. Masukkan id. Pengguna yang akan di ubah	
3. Klik ubah	
	4. Memeriksa data
	5. Data tidak valid dan menampilkan pesan
6. Menginput ulang kode akun	
7. Klik tombol ubah	
	8. Memeriksa valid tidaknya data masukan
	9. Menyimpan id.pengguna yang telah di ubah ke basis data
	10. Menampilkan id. Pengguna yang telah di simpan

Tabel III.10.
Deskripsi *Usecase Diagram* Menghapus Data Pengguna

Aksi Aktor	Reaksi Sistem
Skenario Normal	
1. Memilih data	
2. Klik hapus	
	3. Menampilkan konfirmasi apakah data akan benar-benar di hapus
4. Pilih Ya	
	5. Menghapus data pengguna dari basis data
	6. Menampilkan pesan bahwa data sukses di hapus

Tabel III.11.
Deskripsi Usecase Diagram Menambah Data Jasa

Aksi Aktor	Reaksi Sistem
Skenario Normal	
1. Klik tambah	
	2. Membuat kode jasa baru
3. Memasukkan kode jasa sesuai dengan data	
4. Klik simpan	
	5. Memeriksa valid tidaknya data masukan
	6. Menyimpan data kode jasa ke basis data
	7. Menampilkan pesan suskses disimpan
Skenario Alternatif	
1. Klik tambah	
	2. Membuat kode jasa baru
3. Memasukkan kode jasa sesuai dengan data	
4. Klik simpan	
	5. Memeriksa data
	6. Data tidak valid dan menampilkan pesan
7. Menginput ulang kode jasa	
8. Klik tombol simpan	
	9. Memeriksa valid tidaknya data masukan
	10. Menyiman kode jasa ke basis data
	11. Menampilkan kode jasa yang disimpan

Tabel III.12.
Deskripsi Usecase Diagram Mengubah Data Jasa

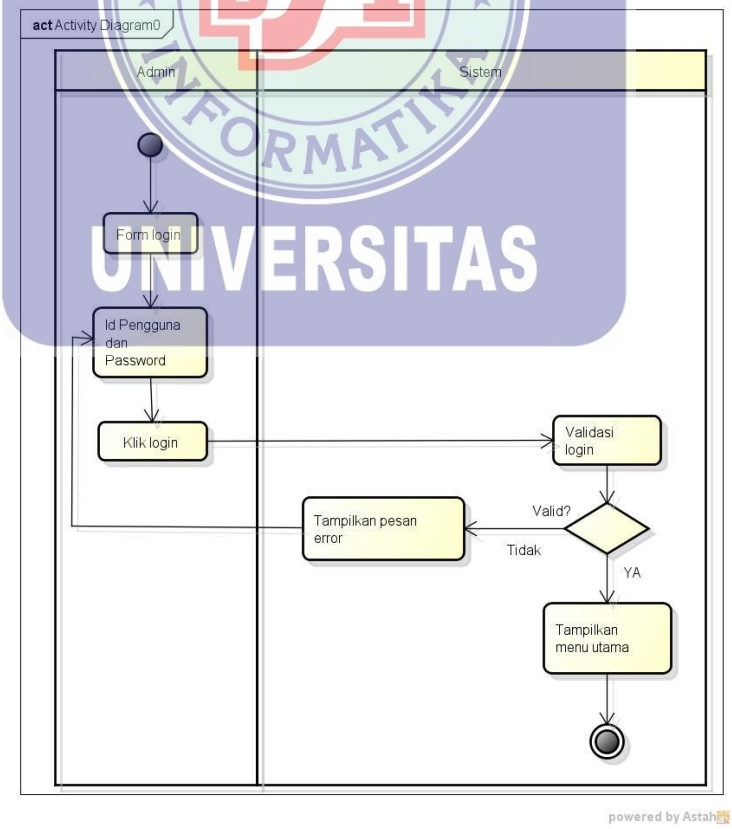
Aksi Aktor	Reaksi Sistem
Skenario Normal	
1. Memilih data	
2. Masukkan kode jasa yang akan di ubah	
3. Klik ubah	
	4. Mencari data kode jasa yang akan di ubah
	5. Menampilkan data kode jasa yang di ubah
6. Memilih data kode jasa yang akan di ubah	
	7. Menampilkan semua kolom data kode jasa yang akan di ubah
Skenario Alternatif	
1. Memilih data	
2. Masukkan kode jasa yang akan di ubah	
3. Klik ubah	
	4. Memeriksa data
	5. Data tidak valid dan menampilkan pesan
6. Menginput ulang kode jasa	
7. Klik tombol ubah	
	8. Memeriksa valid tidaknya data masukan
	9. Menyimpan kode jasa yang telah di ubah ke basis data
	10. Menampilkan kode jasa yang telah di simpan

Tabel III.13.
Deskripsi Usecase Diagram Menghapus Data Jasa

Aksi Aktor	Reaksi Sistem
Skenario Normal	
1. Memilih data	
2. Klik hapus	
	3. Menampilkan konfirmasi apakah data akan benar-benar di hapus
4. Pilih Ya	
	5. Menghapus data jasa dari basis data
	6. Menampilkan pesan bahwa data sukses di hapus

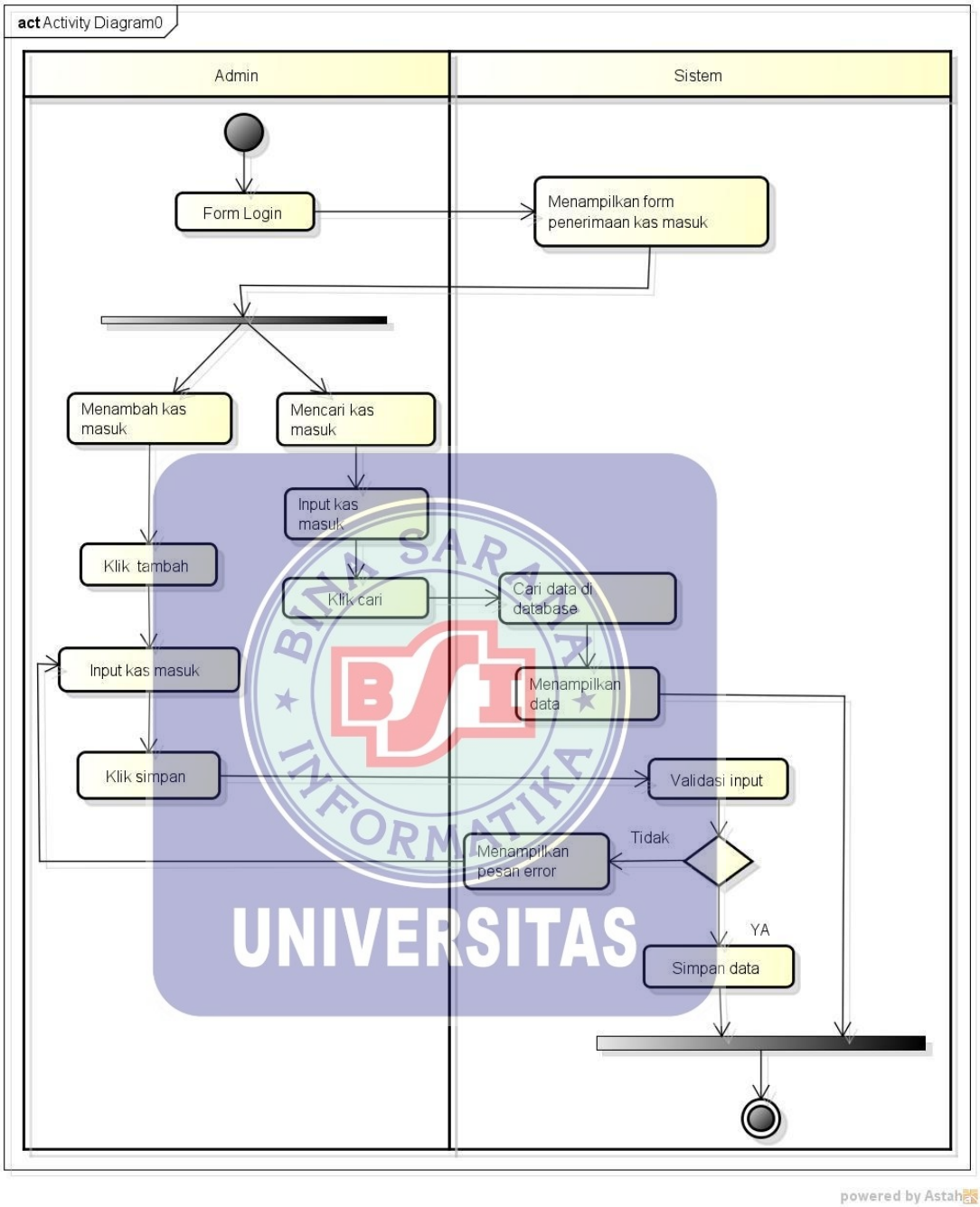
3.3.3. *Activity Diagram*

1. *Activity Diagram* Bagian Admin Melakukan *Login*



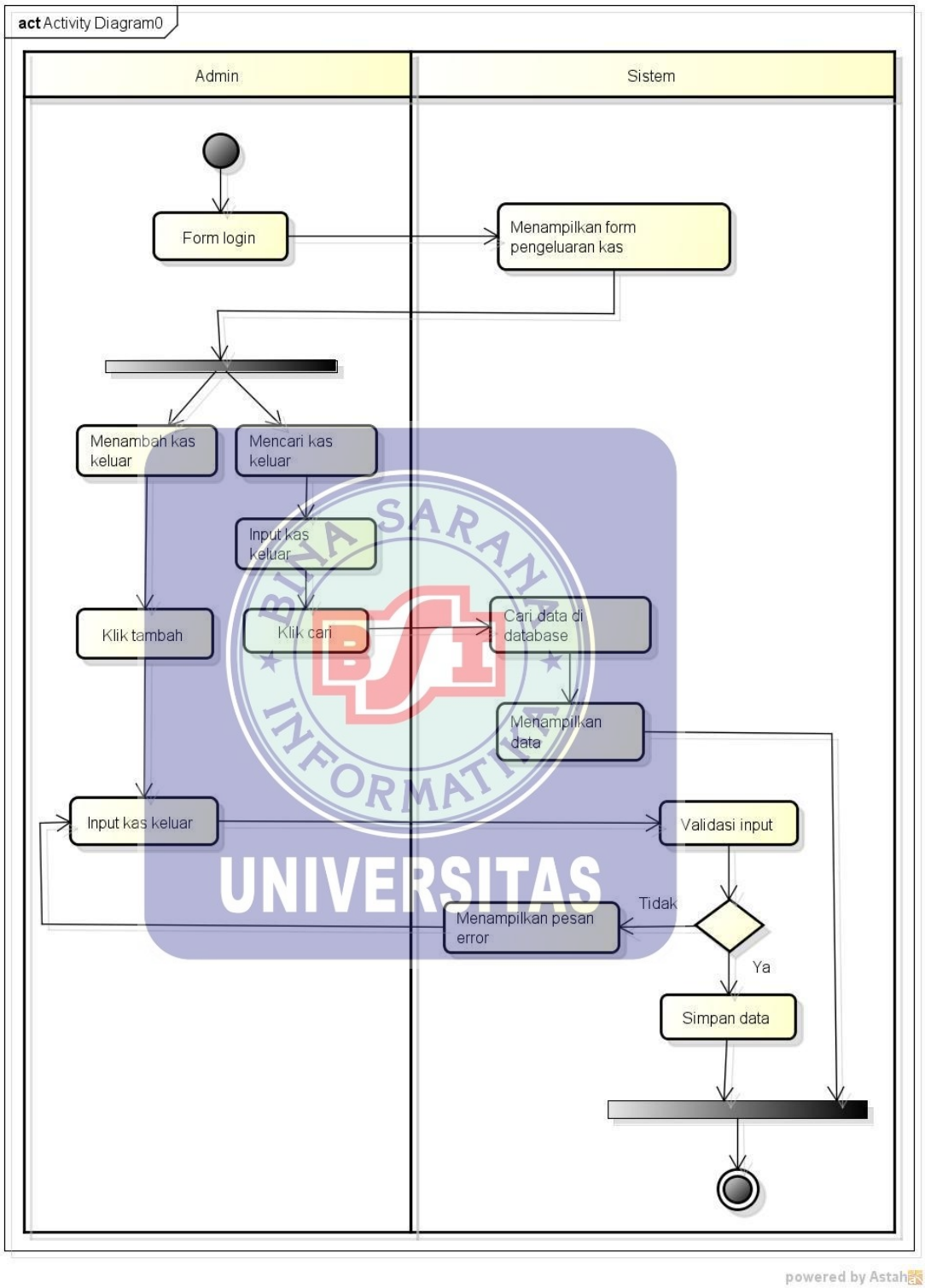
Gambar III.9.
Activity Diagram* Bagian Admin Melakukan *Login

2. *Activity Diagram* Bagian Admin Mengelola Kas Masuk



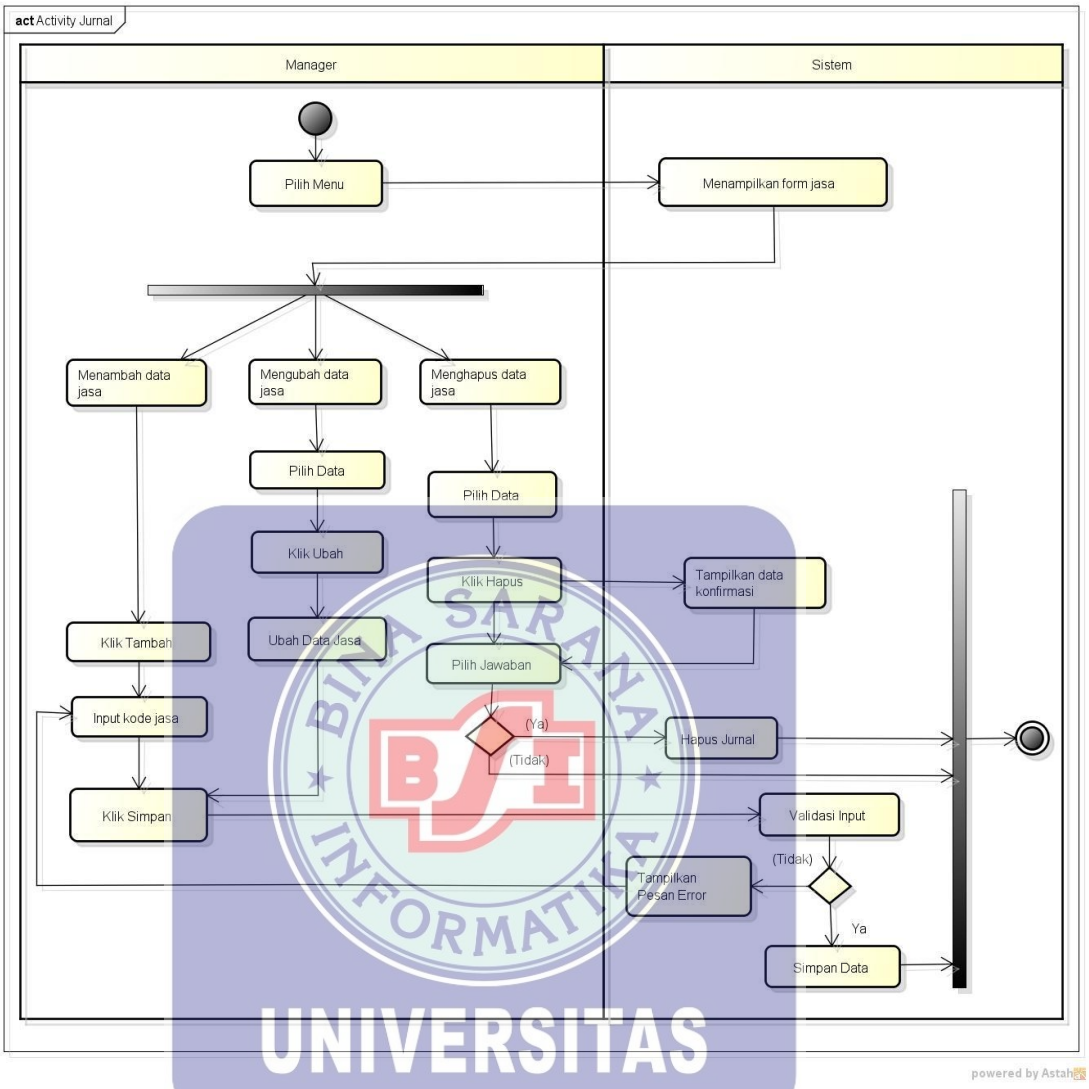
Gambar III.10.
***Activity Diagram* Bagian Admin Mengelola Kas Masuk**

3. *Activity Diagram* Bagian Admin Mengelola Kas Keluar



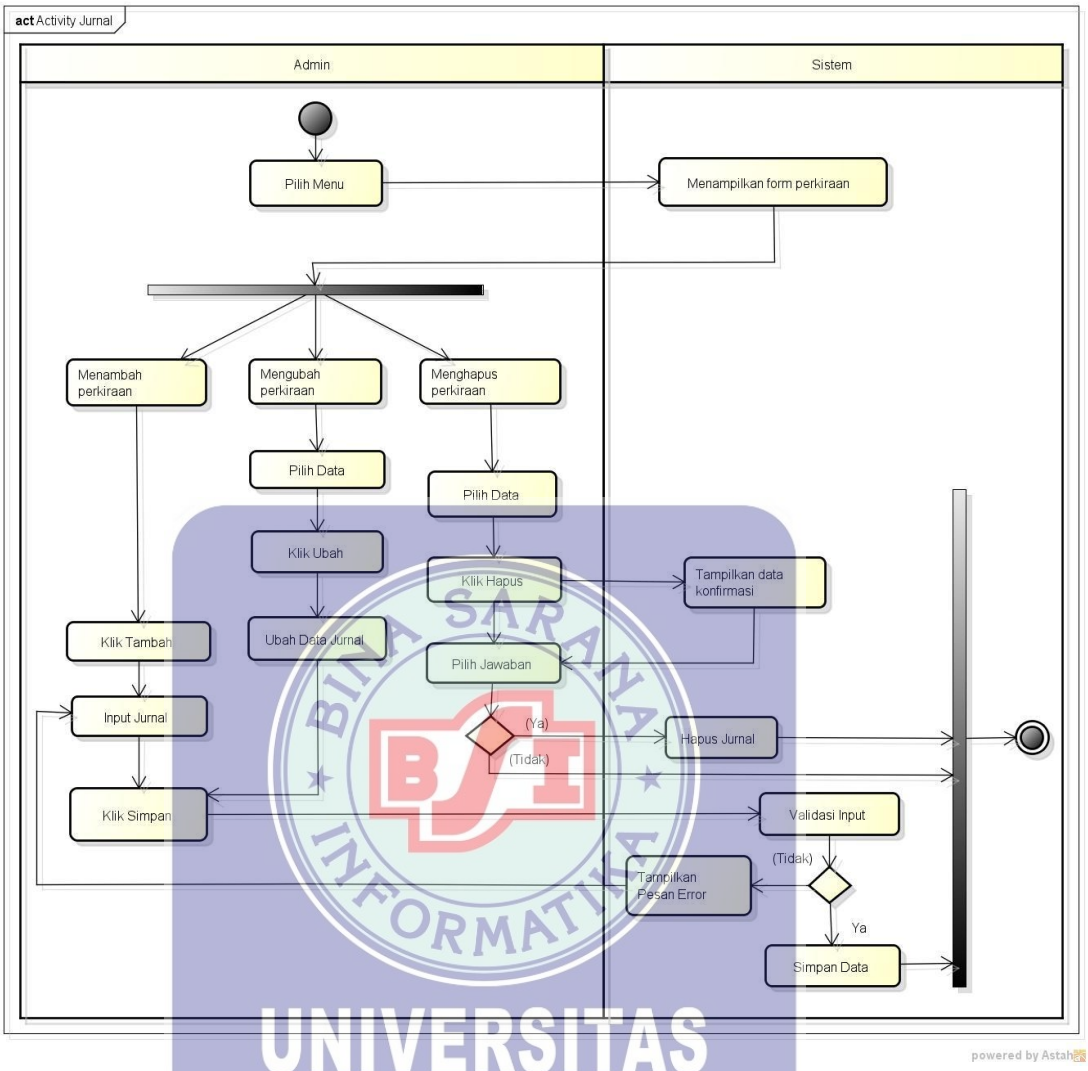
Gambar III.11.
***Activity Diagram* Bagian Admin Mengelola Kas Keluar**

4. *Activity Diagram* Bagian *Manager* Mengelola Jasa



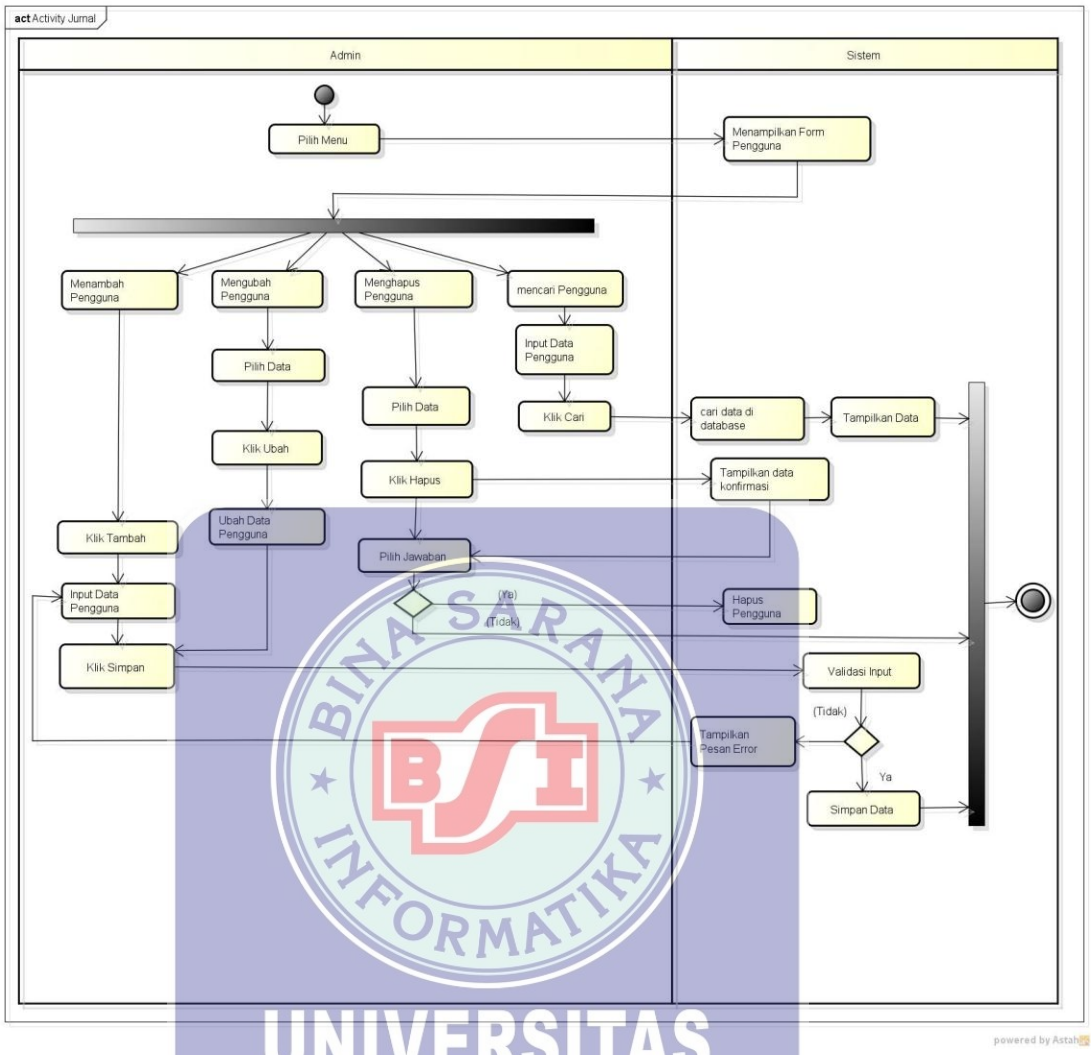
Gambar III.12.
***Activity Diagram* Bagian Manager Mengelola Data Jasa**

5. *Activity Diagram* Bagian Admin Mengelola Perkiraan



Gambar III.13.
***Activity Diagram* Bagian Admin Mengelola Perkiraan**

6. *Activity Diagram* Bagian *Manager* Mengolah Pengguna

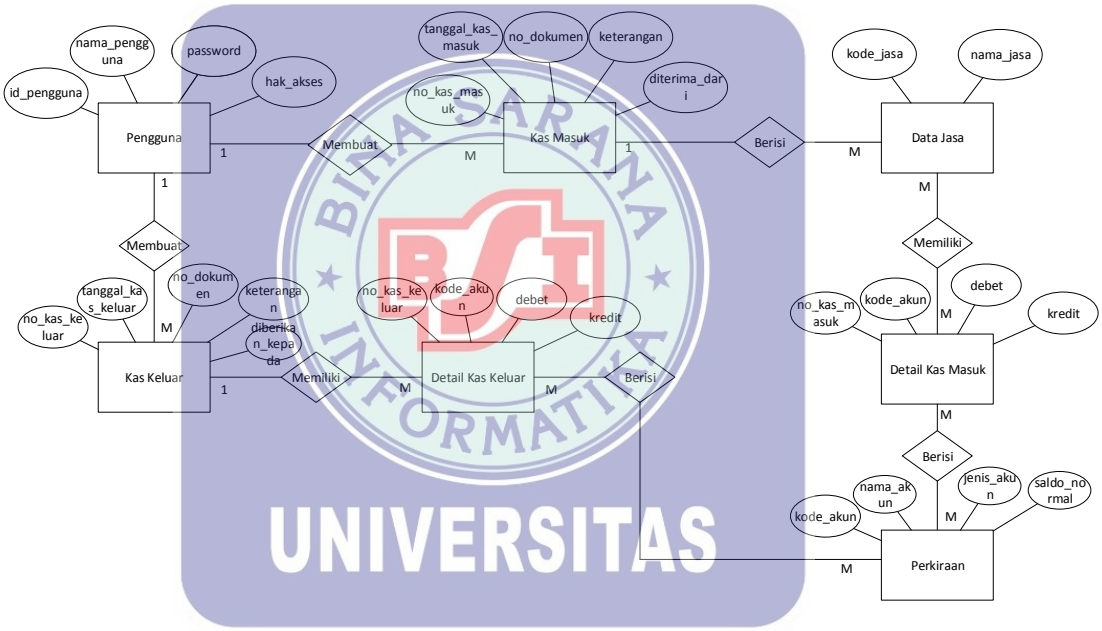


Gambar III.14.
***Activity Diagram* Bagian *Manager* Keuangan Mengolah Perkiraan**

3.4. Desain

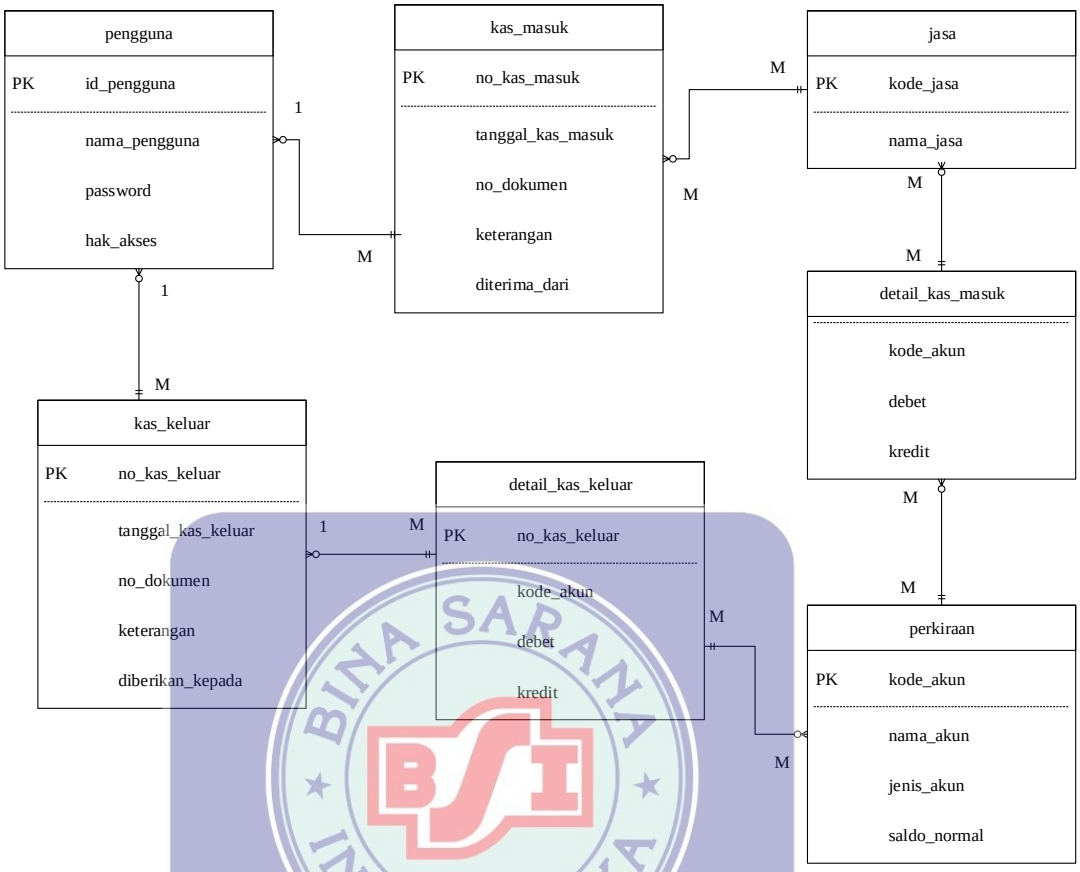
Proses desain akan menerjemahkan syarat kebutuhan sebuah perancangan perangkat lunak yang diperkirakan sebelum dibuat *codingan*, proses ini berfokus pada *usecase diagram*, *activity diagram*, *entity relationship diagram* (ERD), *logical record structure* (LRS) dan *sequence*. Dengan menggunakan UML (*Unified Modelling Language*) sebagai salah satu alat bantu yang sangat handal di dunia pengembangan sistem yang berorientasi obyek.

3.4.1. *Entity Relationship Diagram* (ERD)



Gambar III.15.
***Entity Relationship Diagram* (ERD)**

3.4.2. *Logical Record Structure* (LRS)



Gambar III.16.
***Logical Record Structure* (LRS)**

3.4.3. Spesifikasi *File*

a. Spesifikasi *File* Kas Masuk

Nama *Database* : la_derra_hotel.sql

Nama *File* : kas masuk

Akronim : kas masuk.sql

Tipe *File* : *File* Transaksi

Media Penyimpanan : Hardisk

Akses File : Random

Panjang *Record* : 70 Byte

Kunci Field : no_kas_masuk

Tabel.III.14.
Spesifikasi *File* Kas Masuk

No.	Elemen Data	Nama Field	Type	Size	Keterangan
1	No. Kas Masuk	no_kas_masuk	*Char*	10	*Primary Key*
2	Tanggal Kas Masuk	tanggal_kas_masuk	*Date*		
3	No. Dokumen	no_dokumen	*Char*	12	
4	Keterangan	Keterangan	*Varchar*	35	
5	Diterima Dari	diterima_oleh	*Varchar*	20	

b. Spesifikasi *File* Kas Keluar

Nama *Database* : la_derra_hotel.sql

Nama *File* : kas keluar

Akronim : kas keluar.sql

Tipe *File* : *File* Transaksi

Media Penyimpanan : Hardisk

Akses *File* : Random

Panjang *Record* : 90 Byte

Kunci Field : no_kas_keluar

Tabel.III.15.
Spesifikasi *File* Kas Keluar

No.	Elemen Data	Nama Field	Type	Size	Keterangan
1	No. Kas Keluar	no_kas_keluar	*Char*	10	*Primary Key*
2	Tanggal Kas Keluar	tanggal_kas_keluar	*Date*		
3	No. Dokumen	no_dokumen	*Char*	20	
4	Keterangan	Keterangan	*Varchar*	40	
5	Diberikan_kepada	diberikan_kepada	*Varchar*	20	

c. Spesifikasi *File* Jasa

Nama *Database* : la_derra_hotel.sql

Nama *File* : jasa

Akronim : jasa.sql

Tipe *File* : *File* Master

Media Penyimpanan : Hardisk

Akses *File* : Random

Panjang *Record* : 33 Byte

Kunci Field : kode_jasa

Tabel.III.16.
Spesifikasi *File* Jasa

No.	Elemen Data	Nama Field	Type	Size	Keterangan
1	Kode Jasa	Kode_jasa	*Char*	3	*Primary Key*
2	Nama Jasa	Nama_jasa	*Varchar*	30	

d. Spesifikasi *File* Perkiraan

Nama *Database* : la_derra_hotel.sql

Nama *File* : perkiraan

Akronim : perkiraan.sql

Tipe *File* : *File* Master

Media Penyimpanan : Hardisk

Akses *File* : Random

Panjang *Record* : 101 Byte

Kunci Field : kode_akun

Tabel.III.17.
Spesifikasi *File* Perkiraan

No.	Elemen Data	Nama Field	Type	Size	Keterangan
1	Kode Akun	kode_akun	*Char*	5	*Primary Key*
2	Nama Akun	nama_akun	*Varchar*	45	
3	Jenis Akun	jenis_akun	*Varchar*	45	
4	Saldo Normal	saldo_normal	*Varchar*	6	

e. Spesifikasi *File* Detail Kas Masuk

Nama *Database* : la_derra_hotel.sql

Nama *File* : detail kas masuk

Akronim : detail kas masuk.sql

Tipe *File* : *File* Transaksi

Media Penyimpanan : Hardisk

Akses *File* : Random

Panjang *Record* : 15 Byte

Kunci Field : no_kas_masuk

Tabel.III.18.
Spesifikasi *File Detail* Kas Masuk

No.	Elemen Data	Nama Field	Type	Size	Keterangan
1	No. Kas Masuk	no_kas_masuk	*Char*	10	*Primary Key*
2	Kode Akun	kode_akun	*Char*	5	
3	Debet	Debet	*Double*		
4	Kredit	Kredit	*Double*		

f. Spesifikasi *File Detail* Kas Keluar

Nama *Database* : la_derra_hotel.sql

Nama *File* : detail kas keluar

Akronim : detail kas keluar.sql

Tipe *File* : *File* Transaksi

Media Penyimpanan : Hardisk

Akses *File* : Random

Panjang *Record* : 15 Byte

Kunci Field : no_kas_keluar

Tabel.III.19.
Spesifikasi *File Detail* Kas Keluar

No.	Elemen Data	Nama Field	Type	Size	Keterangan
1	No. Kas Keluar	no_kas_keluar	*Char*	10	*Primary Key*
2	Kode Akun	kode_akun	*Char*	5	
3	Debet	Debet	*Double*		
4	Kredit	Kredit	*Double*		

g. Spesifikasi *File* Pengguna

Nama *Database* : la_derra_hotel.sql

Nama *File* : pengguna

Akronim : pengguna.sql

Tipe *File* : *File* Master

Media Penyimpanan : Hardisk

Akses *File* : Random

Panjang *Record* : 130 Byte

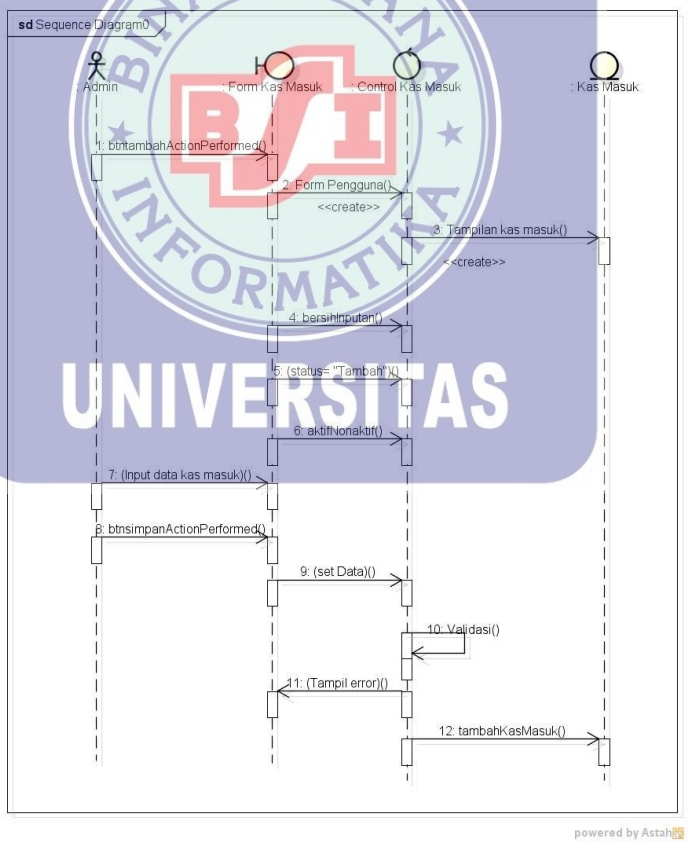
Kunci Field : id_pengguna

Tabel.III.20.
Spesifikasi *File* Pengguna

No.	Elemen Data	Nama Field	Type	Size	Keterangan
1	Id. Pengguna	id_pengguna	*Varchar*	20	*PrimaryKey*
2	Nama Pengguna	nama_pengguna	*Varchar*	35	
3	Password	Password	*Varchar*	35	
4	Hak Akses	hak_akses	*Varchar*	40	

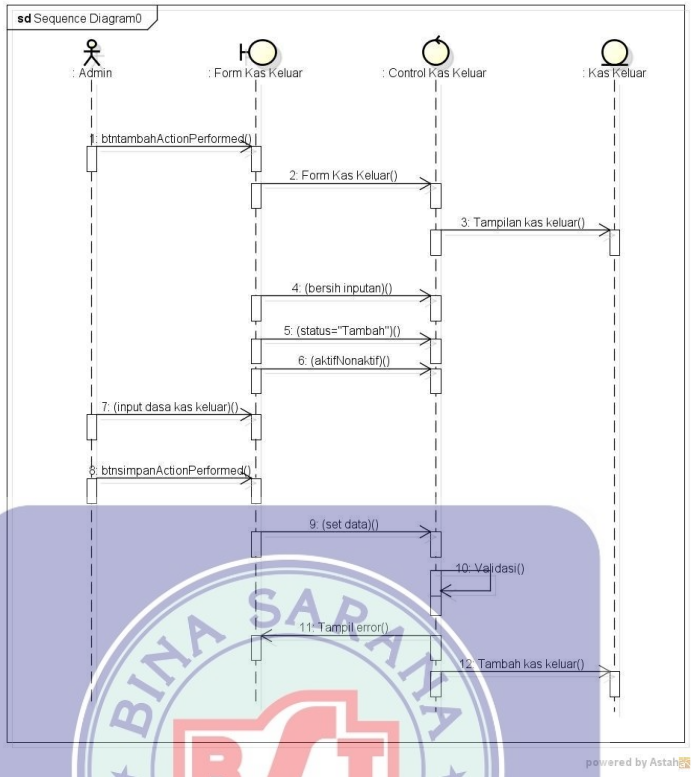
3.4.4. *Sequence Diagram*

1. *Sequence Diagram* Penerimaan Kas



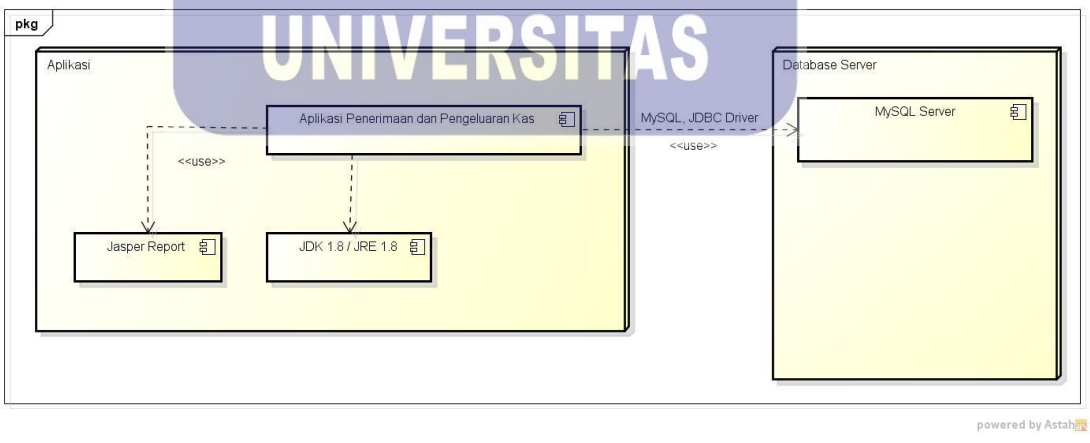
Gambar III.17.
***Sequence Diagram* Penerimaan Kas**

2. *Sequence Diagram* Pengeluaran Kas



Gambar III.18.
***Sequence Diagram* Pengeluaran Kas**

3.4.5. *Deployment Diagram*



Gambar III.19.
***Deployment Diagram* Sistem Penerimaan Dan Pengeluaran Kas**

3.4.6. *User Inteface*

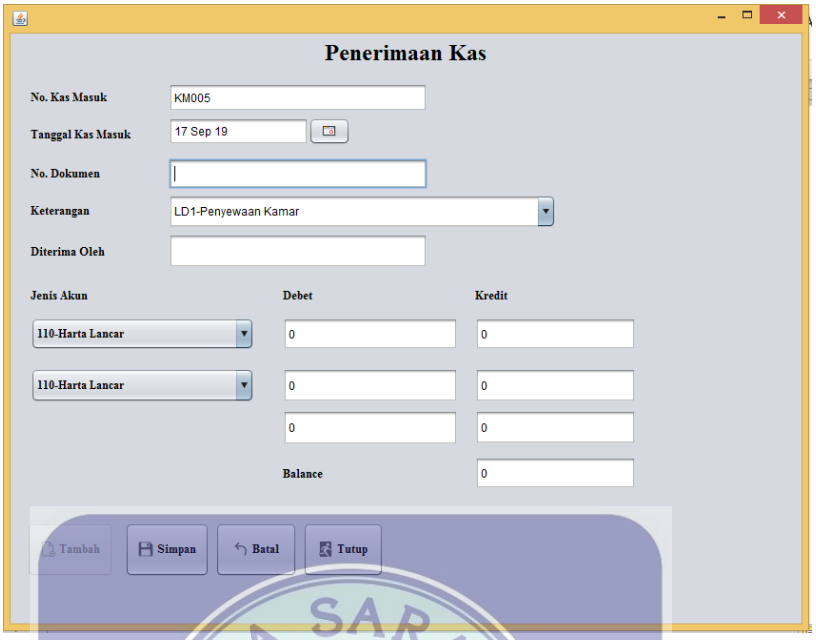
a. Tampilan *Form Login*

Gambar III.20.
User Interface Form Login

b. Tampilan *Form* Menu

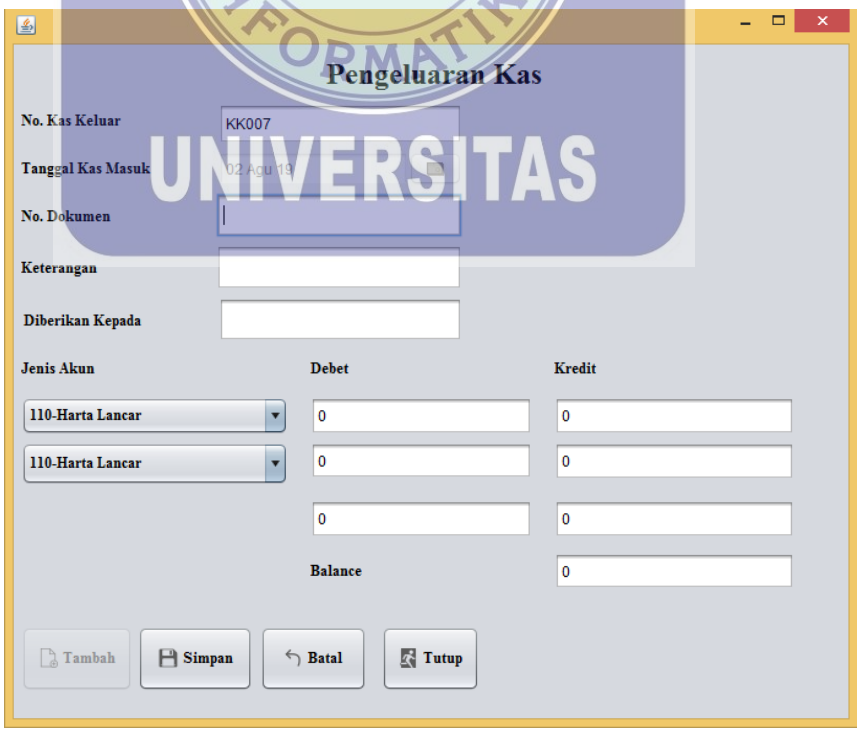
Gambar III.21.
***User Interface Form* Menu Utama**

c. Tampilan *Form* Peneriman Kas



Gambar III.22.
***User Interface Form* Penerimaan Kas**

d. Tampilan *Form* Pengeluaran Kas



Gambar III.23.
***User Interface Form* Pengeluaran Kas**

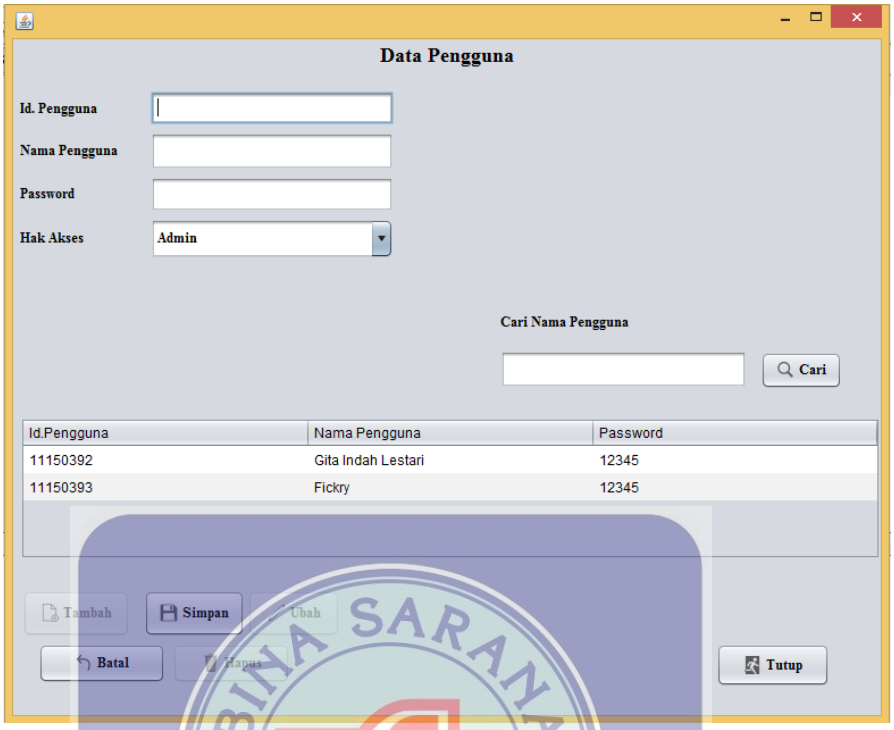
e. Tampilan *Form* Jasa

Gambar III.24.
***User Interface Form* Pengeluaran Kas**

f. Tampilan *Form* Perkiraan

Gambar III.25.
***User Interface Form* Perkiraan**

g. Tampilan *Form* Pengguna



Gambar III.26.
***User Interface Form* Pengguna**

h. Tampilan *Form* Laporan

Gambar III.27.
***User Interface Form* Laporan**

3.5. Implementasi

3.5.1. *Code Generation*

1. Model Penerimaan Kas

```
package model;

import java.sql.Connection;
import java.sql.Date;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.ArrayList;
import java.util.List;
import java.util.logging.Level;
import java.util.logging.Logger;

public class KasMasuk {

    private Connection Koneksi;
    private String no_kas_masuk;
    private Date tanggal_kas_masuk;
    private String no_dokumen;
    private String keterangan;
    private String diterima_dari;

    private String PesanError;

    public KasMasuk() {
        try {
            Class.forName("com.mysql.jdbc.Driver");
            Koneksi =
DriverManager.getConnection("jdbc:mysql://localhost/la_derra_hotel?", "root", "");
        } catch (ClassNotFoundException | SQLException ex) {
            this.PesanError = ex.getMessage();
        }
    }

    public Connection getKoneksi() {
        return Koneksi;
    }

    public void setKoneksi(Connection Koneksi) {
        this.Koneksi = Koneksi;
    }
    public String getNo_kas_masuk() {
        return no_kas_masuk;
    }

    public void setNo_kas_masuk(String no_kas_masuk) {
```

```
        this.no_kas_masuk = no_kas_masuk;
    }

    public Date getTanggal_kas_masuk() {
        return tanggal_kas_masuk;
    }

    public void setTanggal_kas_masuk(Date tanggal_kas_masuk) {
        this.tanggal_kas_masuk = tanggal_kas_masuk;
    }

    public String getNo_dokumen() {
        return no_dokumen;
    }

    public void setNo_dokumen(String no_dokumen) {
        this.no_dokumen = no_dokumen;
    }

    public String getKeterangan() {
        return keterangan;
    }

    public void setKeterangan(String keterangan) {
        this.keterangan = keterangan;
    }

    public String getDiterima_dari() {
        return diterima_dari;
    }

    public void setDiterima_dari(String diterima_dari) {
        this.diterima_dari = diterima_dari;
    }

    public String getPesanError() {
        return PesanError;
    }

    public void setPesanError(String PesanError) {
        this.PesanError = PesanError;;
    }

    public String NoKasMasukAuto() {
        try {
            String Cmd = "SELECT * FROM kas_masuk ORDER BY
SUBSTR(no_kas_masuk,3,5)* 1 desc";
            Statement st = Koneksi.createStatement();
            ResultSet rs = st.executeQuery(Cmd);

            String No = "";
            if(rs.next()){
                No = rs.getString("no_kas_masuk");
                No  = No.substring(2,5);
```

```
            No = "KM"+ String.format("%03d", Integer.parseInt(No)+1);
        }else{
            No="KM001";
        }
    return No;
    } catch (SQLException ex) {
        Logger.getLogger(KasMasuk.class.getName()).log(Level.SEVERE, null, ex);
        return "";
    }

}

public List<KasMasuk> daftarKasMasuk() {
    this.PesanError = "";
    try {
        String Cmd = "SELECT * FROM kas masuk";
        List<KasMasuk> lp = new ArrayList();
        Statement st = Koneksi.createStatement();
        ResultSet rs = st.executeQuery(Cmd);
        while (rs.next()) {
            KasMasuk km = new KasMasuk();
            km.setNo_kas_masuk(rs.getString("no_kas_masuk"));
            km.setTanggal_kas_masuk(rs.getDate("tanggal_kas_masuk"));
            km.setNo_dokumen(rs.getString("no_referensi"));
            km.setKeterangan(rs.getString("keterangan"));
            km.setDiterima_dari(rs.getString("id_pengguna"));

            lp.add(km);
        }
        return lp;
    } catch (SQLException ex) {
        this.PesanError = ex.getMessage();
        return null;
    }
}

public List<KasMasuk> cariNoKas(String No) {
    this.PesanError = "";
    try {
        String Cmd = "SELECT * FROM kas masuk WHERE no_kas_masuk LIKE=?";
        List<KasMasuk> lp = new ArrayList();
        PreparedStatement ps = Koneksi.prepareStatement(Cmd);
        String Nama;
        ps.setString(1, "%" + No + "%");
        ResultSet rs = ps.executeQuery();
        KasMasuk km = new KasMasuk();

        while (rs.next()) {
            km.setNo_kas_masuk(rs.getString("no_kas_masuk"));
            km.setTanggal_kas_masuk(rs.getDate("tanggal_kas_masuk"));
            km.setNo_dokumen(rs.getString("no_dokumen"));
            km.setKeterangan(rs.getString("keterangan"));
            km.setDiterima_dari(rs.getString("diterima_oleh"));
```

```
            lp.add(km);
          }
          return lp;
        } catch (SQLException ex) {
          this.PesanError = ex.getMessage();
          return null;
        }
    }
    public int tambahKasMasuk() {
        this.PesanError = "";
        try {
          String Cmd = "INSERT INTO KAS_MASUK (no_kas_masuk,
tanggal_kas_masuk, no_dokumen, keterangan, diterima_oleh) VALUES (?,?,?,?,?)";
          PreparedStatement km = Koneksi.prepareStatement(Cmd);

          km.setString(1, this.no_kas_masuk);
          km.setDate(2, this.tanggal_kas_masuk);
          km.setString(3, this.no_dokumen);
          km.setString(4, this.keterangan);
          km.setString(5, this.diterima_dari);
          return km.executeUpdate();
        } catch (SQLException ex) {
          this.PesanError = ex.getMessage();
          return -1;
        }
    }

    public int ubahJurnal(String Kode) {
        this.PesanError = "";
        try {
          String Cmd = "UPDATE kas masuk SET tanggal_kas_masuk=?,
no_dokumen=?, keterangan=?, diterima_oleh=? WHERE no_kas_masuk=?";
          PreparedStatement km = Koneksi.prepareStatement(Cmd);

          km.setDate(1, this.tanggal_kas_masuk);
          km.setString(2, this.no_dokumen);
          km.setString(3, this.keterangan);
          km.setString(4, this.diterima_dari);
          return km.executeUpdate();
        } catch (SQLException ex) {
          this.PesanError = ex.getMessage();
          return -1;
        }
    }

    public int hapusJurnal(String Kode) {
        this.PesanError = "";
        try {
          String Cmd = "DELETE FROM kas masuk WHERE no_kas_masuk=?";
          PreparedStatement ps = Koneksi.prepareStatement(Cmd);
          ps.setString(1, Kode);
          return ps.executeUpdate();
        } catch (SQLException ex) {
          this.PesanError = ex.getMessage();
```

```
            return -1;
        }
    }

    public KasMasuk cariNo(String no_kas_masuk) {
        this.PesanError = "";
        try {
            String Cmd = "SELECT * FROM kas masuk WHERE no_kas_masuk=?";
            PreparedStatement ps = Koneksi.prepareStatement(Cmd);

            ps.setString(1, no_kas_masuk);
            ResultSet rs = ps.executeQuery();

            if (rs.next()) {
                this.setNo_kas_masuk(rs.getString("no_kas_masuk"));
                this.setTanggal_kas_masuk(rs.getDate("tanggal_kas_masuk"));
                this.setNo_dokumen(rs.getString("no_dokumen"));
                this.setKeterangan(rs.getString("keterangan"));
                this.setDiterima_dari(rs.getString("diterima_dari"));
            } else {
                this.setNo_kas_masuk(null);
                this.setTanggal_kas_masuk(null);
                this.setNo_dokumen(null);
                this.setKeterangan(null);
                this.setDiterima_dari(null);
            }
            return this;
        } catch (SQLException ex) {
            return null;
        }
    }
}
```

2. *Form* Penerimaan Kas

```
package Form;

import java.awt.Dimension;
import java.awt.Toolkit;
import java.util.List;
import java.util.ListIterator;
import javax.swing.DefaultComboBoxModel;
import javax.swing.JOptionPane;
import model.DetailKasMasuk;
import model.Perkiraan;
import model.KasMasuk;

public class FormKasMasuk extends javax.swing.JFrame {
String kondisi = "";
    String no_kas_masuk = "";
    KasMasuk kas_masuk = new KasMasuk();
    DetailKasMasuk detail_kas_masuk = new DetailKasMasuk();
    Perkiraan perkiraan = new Perkiraan();
    DefaultComboBoxModel tm1 = null;
```

```
    DefaultComboBoxModel tm2 = null;

    public FormKasMasuk() {
        initComponents();
        Toolkit tk = Toolkit.getDefaultToolkit();
        Dimension d = tk.getScreenSize();
        int x, y;
        x = (int) ((d.getWidth() - getSize().width) / 2);
        y = (int) ((d.getHeight() - getSize().height) / 2);

        setLocation(x, y);
        bersih_Inputan();
        listAkun();
        aktif_Button();
    }

    private void Balance() {
        double balance = 0;
        double ttldebet = 0;
        double ttlkredit = 0;
        try {

            ttldebet = Double.parseDouble(t_Debet1.getText())
                    + Double.parseDouble(t_Debet2.getText());
            ttlkredit = Double.parseDouble(t_Kredit1.getText())
                    + Double.parseDouble(t_Kredit2.getText());
            balance = ttldebet - ttlkredit;

            ttl_debet.setText(String.valueOf(ttl_debet));
            ttl_kredit.setText(String.valueOf(ttl_kredit));

            ttl_balance.setText(String.valueOf(ttl_balance));
        } catch (Exception e) {
            ttldebet = 0;
            ttlkredit = 0;
            balance = 0;
            ttl_debet.setText(String.valueOf(ttldebet));
            ttl_kredit.setText(String.valueOf(ttlkredit));
            ttl_balance.setText(String.valueOf(balance));
        }
    }

    private void listAkun() {
        List<Perkiraan> lp = perkiraan.daftarPerkiraan();
        System.out.println(perkiraan.getPesanError());
        ListIterator li = lp.listIterator();
        tm1.removeAllElements();
        tm2.removeAllElements();
        while (li.hasNext()) {
            perkiraan = (Perkiraan) li.next();
            tm1.addElement(perkiraan.getKode_Akun()          +          "-"          +
perkiraan.getNama_Akun());
            tm2.addElement(perkiraan.getKode_Akun()          +          "-"          +
perkiraan.getNama_Akun());
```

```
        }
    }

    private void aktif_Button() {
        btn_tambah.setEnabled("".equals(this.kondisi));
        btn_simpan.setEnabled("".equals(this.kondisi));
        btn_batal.setEnabled("".equals(this.kondisi));

        txt_no_kas_masuk.setEditable(false);
        txt_tanggal_kas_masuk.setEnabled(false);
        txt_keterangan.setEditable(!"".equals(this.kondisi));
        txt_akun1.setEnabled(!"".equals(this.kondisi));
        txt_akun2.setEnabled(!"".equals(this.kondisi));

        t_Debet1.setEnabled(!"".equals(this.kondisi));
        t_Debet2.setEnabled(!"".equals(this.kondisi));

        t_Kredit1.setEnabled(!"".equals(this.kondisi));
        t_Kredit2.setEnabled(!"".equals(this.kondisi));

        txt_no_dokumen.setEditable(!"".equals(this.kondisi));

    }

    private void bersih_Inputan() {
        txt_no_kas_masuk.setText("");
        txt_tanggal_kas_masuk.setDate(new java.util.Date());
        txt_keterangan.setText("");
        txt_akun1.setSelectedItem("");
        txt_akun2.setSelectedItem("");
        txt_no_dokumen.setText("");

        t_Debet1.setText("0");
        t_Debet2.setText("0");

        t_Kredit1.setText("0");
        t_Kredit2.setText("0");

        ttl_debet.setText("0");
        ttl_kredit.setText("0");
        ttl_balance.setText("0");

    }
    private void txt_akun1ActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
    }

    private void btn_tambahActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        this.kondisi = "Tambah";
        this.no_kas_masuk = "";
        bersih_Inputan();
        aktif_Button();
```

```
        txt_no_kas_masuk.setText(kas_masuk.NoKasMasukAuto());
        txt_no_dokumen.requestFocus();

    }

    private void t_Kredit2ActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        Balance();
    }
    private void btn_simpanActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        if ("".equals(txt_no_kas_masuk.getText())
                || txt_tanggal_kas_masuk.getDate() == null
                || "".equals(txt_keterangan.getText())
                || "".equals(txt_no_dokumen.getText())
                || "".equals(txt_akun1.getSelectedItem().toString())
                || "".equals(txt_akun2.getSelectedItem().toString())) {
            JOptionPane.showMessageDialog(rootPane, "Mohon isi dulu semua data",
"Error", JOptionPane.ERROR_MESSAGE);
            return;
        }
        if (Double.parseDouble(ttl_balance.getText()) > 0.0) {
            JOptionPane.showMessageDialog(rootPane, "Kas masuk belum balance,
silahkan seimbangkan dahulu lajur debet dan kredit", "Rrror",
                    JOptionPane.ERROR_MESSAGE);
            return;
        }
        kas_masuk.setNo_kas_masuk(txt_no_kas_masuk.getText());
        kas_masuk.setTanggal_kas_masuk(new                    java.sql.Date(new
java.util.Date().getTime()));
        kas_masuk.setKeterangan(txt_keterangan.getText());
        kas_masuk.setNo_dokumen(txt_no_dokumen.getText());
        kas_masuk.setDiterima_dari(txt_diterima_dari.getText());

        if ("Tambah".equals(kondisi)) {
            String[] Akun;
            if (kas_masuk.tambahKasMasuk() > 0) {
                detail_kas_masuk.setNoKasMasuk(txt_no_kas_masuk.getText());

                Akun = txt_akun1.getSelectedItem().toString().split("-");
                detail_kas_masuk.setKode_Perkiraan(Akun[0]);
                detail_kas_masuk.setDebet(Double.parseDouble(t_Debet1.getText()));
                detail_kas_masuk.setKredit(Double.parseDouble(t_Kredit1.getText()));
                detail_kas_masuk.tambah_kas_masuk();

                JOptionPane.showMessageDialog(rootPane, "Data kas_masuk berhasil
disimpan!", "Info", JOptionPane.INFORMATION_MESSAGE);
            } else {
                JOptionPane.showMessageDialog(rootPane, "Data kas_masuk gagal
disimpan. \n" + kas_masuk.getPesanError(), "Error",
                        JOptionPane.ERROR_MESSAGE);
                return;
            }
        }
```

```
    this.kondisi = "";
    this.no_kas_masuk = "";
    bersih_Inputan();
    aktif_Button();
}

private void btn_batalActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    this.kondisi = "";
    this.no_kas_masuk = "";
    bersih_Inputan();
    aktif_Button();
}

private void btn_tutupActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    dispose();
}

private void t_Debet1ActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    Balance();
}

private void t_Kredit1ActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    Balance();
}
private void t_Debet1KeyTyped(java.awt.event.KeyEvent evt) {
    // TODO add your handling code here:
    Balance ();
}

private void t_Kredit1KeyTyped(java.awt.event.KeyEvent evt) {
    // TODO add your handling code here:
    Balance ();
}

private void t_Kredit2KeyTyped(java.awt.event.KeyEvent evt) {
    // TODO add your handling code here:
    Balance ();
}

private void t_Kredit2KeyPressed(java.awt.event.KeyEvent evt) {
    // TODO add your handling code here:
    Balance ();
}

private void t_Kredit2KeyReleased(java.awt.event.KeyEvent evt) {
    // TODO add your handling code here:
    Balance ();
}

private void t_Debet1KeyPressed(java.awt.event.KeyEvent evt) {
```

```
    // TODO add your handling code here:
    Balance ();
}

private void t_Debet1KeyReleased(java.awt.event.KeyEvent evt) {
    // TODO add your handling code here:
    Balance ();
}

private void t_Debet2KeyPressed(java.awt.event.KeyEvent evt) {
    // TODO add your handling code here:
    Balance ();
}

private void t_Debet2KeyReleased(java.awt.event.KeyEvent evt) {
    // TODO add your handling code here:
    Balance ();
}

private void t_Debet2KeyTyped(java.awt.event.KeyEvent evt) {
    // TODO add your handling code here:
    Balance ();
}
private void t_Kredit1KeyReleased(java.awt.event.KeyEvent evt) {
    // TODO add your handling code here:
    Balance ();
}

private void t_Kredit1KeyPressed(java.awt.event.KeyEvent evt) {
    // TODO add your handling code here:
    Balance ();
}
```

3. Model Pengeluaran Kas

```
package model;

import java.sql.Connection;
import java.sql.Date;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.ArrayList;
import java.util.List;
import java.util.logging.Level;
import java.util.logging.Logger;

public class KasKeluar {

    private Connection Koneksi;

    private String no_kas_keluar;
```

```
    private Date tanggal_kas_keluar;
    private String no_dokumen;
    private String keterangan;
    private String diberikan_kepada;

    private String PesanError;

    public KasKeluar() {
        try {
            Class.forName("com.mysql.jdbc.Driver");
            Koneksi                                                    =
DriverManager.getConnection("jdbc:mysql://localhost/la_derra_hotel?", "root", "");
        } catch (ClassNotFoundException | SQLException ex) {
            this.PesanError = ex.getMessage();
        }
    }

    public Connection getKoneksi() {
        return Koneksi;
    }

    public void setKoneksi(Connection Koneksi) {
        this.Koneksi = Koneksi;
    }

    public String getNo_kas_keluar() {
        return no_kas_keluar;
    }

    public void setNo_kas_keluar(String no_kas_keluar) {
        this.no_kas_keluar = no_kas_keluar;
    }

    public Date getTanggal_kas_keluar() {
        return tanggal_kas_keluar;
    }

    public void setTanggal_kas_keluar(Date tanggal_kas_keluar) {
        this.tanggal_kas_keluar = tanggal_kas_keluar;
    }


    public String getNo_dokumen() {
        return no_dokumen;
    }

    public void setNo_dokumen(String no_dokumen) {
        this.no_dokumen = no_dokumen;
    }

    public String getKeterangan() {
        return keterangan;
    }
```

```
    public void setKeterangan(String keterangan) {
        this.keterangan = keterangan;
    }

    public String getDiberikan_kepada() {
        return diberikan_kepada;
    }

    public void setDiberikan_kepada(String diberikan_kepada) {
        this.diberikan_kepada = diberikan_kepada;
    }

    public String getPesanError() {
        return PesanError;
    }

    public void setPesanError(String PesanError) {
        this.PesanError = PesanError;
    }

    public String NoKasKeluarAuto() {
        try {
            String Cmd = "SELECT * FROM kas keluar ORDER BY
SUBSTR(no_kas_keluar,3,5)* 1 desc";
            Statement st = Koneksi.createStatement();
            ResultSet rs = st.executeQuery(Cmd);

            String No = "";
            if(rs.next()){
                No = rs.getString("no_kas_keluar");
                No  = No.substring(2,5);
                No = "KK"+ String.format("%03d", Integer.parseInt(No)+1);
            }else{
                No="KK001";
            }
        return No;
        } catch (SQLException ex) {
            Logger.getLogger(KasKeluar.class.getName()).log(Level.SEVERE, null, ex);
            return "";
        }

    }

    public List<KasKeluar> daftarKasKeluar() {
        this.PesanError = "";
        try {
            String Cmd = "SELECT * FROM kas_keluar";
            List<KasKeluar> lp = new ArrayList();
            Statement st = Koneksi.createStatement();
            ResultSet rs = st.executeQuery(Cmd);
            while (rs.next()) {
                KasKeluar kk = new KasKeluar();
                kk.setNo_kas_keluar(rs.getString("no_kas_keluar"));
                kk.setTanggal_kas_keluar(rs.getDate("tanggal_kas keluar"));
```

```
            kk.setNo_dokumen(rs.getString("no_dokumen"));
            kk.setKeterangan(rs.getString("keterangan"));
            kk.setDiberikan_kepada(rs.getString("diberikan_kepada"));

            lp.add(kk);
        }
        return lp;
    } catch (SQLException ex) {
        this.PesanError = ex.getMessage();
        return null;
    }
}

public List<KasKeluar> cariNoKas(String No) {
    this.PesanError = "";
    try {
        String Cmd = "SELECT * FROM kas keluar WHERE no_kas_keluar LIKE=?";
        List<KasKeluar> lp = new ArrayList();
        PreparedStatement ps = Koneksi.prepareStatement(Cmd);
        String Nama;
        ps.setString(1, "%" + No + "%");
        ResultSet rs = ps.executeQuery();
        KasKeluar kk = new KasKeluar();

        while (rs.next()) {
            kk.setNo_kas_keluar(rs.getString("no_kas_keluaar"));
            kk.setTanggal_kas_keluar(rs.getDate("tanggal_kas_keluar"));
            kk.setNo_dokumen(rs.getString("no_dokumen"));
            kk.setKeterangan(rs.getString("keterangan"));
            kk.setDiberikan_kepada(rs.getString("diberikan_kepada"));

            lp.add(kk);
        }
        return lp;
    } catch (SQLException ex) {
        this.PesanError = ex.getMessage();
        return null;
    }
}

public int tambahKasKeluar() {
    this.PesanError = "";
    try {
        String Cmd = "INSERT INTO KAS_KELUAR (no_kas_keluar,
tanggal_kas_keluar, no_dokumen, keterangan, diberikan_kepada) VALUES (?,?,?,?,?)";
        PreparedStatement kk = Koneksi.prepareStatement(Cmd);

        kk.setString(1, this.no_kas_keluar);
        kk.setDate(2, this.tanggal_kas_keluar);
        kk.setString(3, this.no_dokumen);
        kk.setString(4, this.keterangan);
        kk.setString(5, this.diberikan_kepada);
        return kk.executeUpdate();
    } catch (SQLException ex) {
```

```
            this.PesanError = ex.getMessage();
            return -1;
        }
    }

    public int ubahJurnal(String Kode) {
        this.PesanError = "";
        try {
            String Cmd = "UPDATE kas_keluar SET tanggal_kas_keluar=?,
no_dokumen=?, keterangan=?, diberikan_kepada=? WHERE no_kas_keluar=?";
            PreparedStatement kk = Koneksi.prepareStatement(Cmd);

            kk.setDate(1, this.tanggal_kas_keluar);
            kk.setString(2, this.no_dokumen);
            kk.setString(3, this.keterangan);
            kk.setString(4, this.diberikan_kepada);
            return kk.executeUpdate();
        } catch (SQLException ex) {
            this.PesanError = ex.getMessage();
            return -1;
        }
    }

    public int hapusJurnal(String Kode) {
        this.PesanError = "";
        try {
            String Cmd = "DELETE FROM kas_keluar WHERE no_kas_keluar=?";
            PreparedStatement ps = Koneksi.prepareStatement(Cmd);
            ps.setString(1, Kode);
            return ps.executeUpdate();
        } catch (SQLException ex) {
            this.PesanError = ex.getMessage();
            return -1;
        }
    }

    public KasKeluar cariNo(String no_kas_keluar) {
        this.PesanError = "";
        try {
            String Cmd = "SELECT * FROM kas_keluar WHERE no_kas_keluar=?";
            PreparedStatement ps = Koneksi.prepareStatement(Cmd);

            ps.setString(1, no_kas_keluar);
            ResultSet rs = ps.executeQuery();

            if (rs.next()) {
                this.setNo_kas_keluar(rs.getString("no_kas_keluar"));
                this.setTanggal_kas_keluar(rs.getDate("tanggal_kas_keluar"));
                this.setNo_dokumen(rs.getString("no_dokumen"));
                this.setKeterangan(rs.getString("keterangan"));
                this.setDiberikan_kepada(rs.getString("diterima_dari"));
            } else {
                this.setNo_kas_keluar(null);
                this.setTanggal_kas_keluar(null);
```

```
                this.setNo_dokumen(null);
                this.setKeterangan(null);
                this.setDiberikan_kepada(null);
            }
            return this;
        } catch (SQLException ex) {
            return null;
        }
    }
}
```

4. *Form* Pengeluaran Kas

```
package Form;

import java.awt.Dimension;
import java.awt.Toolkit;
import java.util.List;
import java.util.ListIterator;
import javax.swing.DefaultComboBoxModel;
import javax.swing.JOptionPane;
import model.DetailKasKeluar;
import model.Perkiraan;
import model.KasKeluar;

public class FormKasKeluar extends javax.swing.JFrame {

String kondisi = "";
    String no_kas_keluar = "";

    KasKeluar kas_keluar = new KasKeluar();
    DetailKasKeluar detail_kas_keluar = new DetailKasKeluar();
    Perkiraan perkiraan = new Perkiraan();

    DefaultComboBoxModel tm1 = null;
    DefaultComboBoxModel tm2 = null;

    public FormKasKeluar() {
        initComponents();
        Toolkit tk = Toolkit.getDefaultToolkit();
        Dimension d = tk.getScreenSize();
        int x, y;
        x = (int) ((d.getWidth() - getSize().width) / 2);
        y = (int) ((d.getHeight() - getSize().height) / 2);

        setLocation(x, y);
        bersih_Inputan();
        listAkun();
        aktif_Button();
    }
    private void Balance () {
        double balance = 0;
        double ttldebet = 0;
        double ttlkredit = 0;
```

```
        try {

            ttldebet = Double.parseDouble(t_Debet1.getText())
               + Double.parseDouble(t_Debet2.getText());
            ttlkredit = Double.parseDouble(t_Kredit1.getText())
               + Double.parseDouble(t_Kredit2.getText());
            balance = ttldebet - ttlkredit;

            ttl_debet.setText(String.valueOf(ttl_debet));
            ttl_kredit.setText(String.valueOf(ttl_kredit));

            ttl_balance.setText(String.valueOf(ttl_balance));
        } catch (Exception e) {
            ttldebet = 0;
            ttlkredit = 0;
            balance = 0;
            ttl_debet.setText(String.valueOf(ttldebet));
            ttl_kredit.setText(String.valueOf(ttlkredit));
            ttl_balance.setText(String.valueOf(balance));
        }
    }

    private void listAkun(){
        List<Perkiraan> lp = perkiraan.daftarPerkiraan();
        System.out.println(perkiraan.getPesanError());
        ListIterator li = lp.listIterator();
        tm1.removeAllElements();
        tm2.removeAllElements();

        while (li.hasNext()) {
            perkiraan = (Perkiraan) li.next();
            tm1.addElement(perkiraan.getKode_Akun()          +          "-"          +
perkiraan.getNama_Akun());
            tm2.addElement(perkiraan.getKode_Akun()          +          "-"          +
perkiraan.getNama_Akun());
        }
    }

    private void aktif_Button() {
        btn_tambah.setEnabled("".equals(this.kondisi));
        btn_simpan.setEnabled("".equals(this.kondisi));
        btn_batal.setEnabled("".equals(this.kondisi));
        txt_no_kas_keluar.setEditable(false);
        txt_tanggal_kas_keluar.setEnabled(false);
        txt_keterangan.setEditable(!"".equals(this.kondisi));
        txt_akun1.setEnabled(!"".equals(this.kondisi));
        txt_akun2.setEnabled(!"".equals(this.kondisi));

        t_Debet1.setEnabled(!"".equals(this.kondisi));
        t_Debet2.setEnabled(!"".equals(this.kondisi));

        t_Kredit1.setEnabled(!"".equals(this.kondisi));
        t_Kredit2.setEnabled(!"".equals(this.kondisi));
```

```
        txt_no_dokumen.setEnabled(!"".equals(this.kondisi));
    }

        private void bersih_Inputan() {
            txt_no_kas_keluar.setText("");
            txt_tanggal_kas_keluar.setDate(new java.util.Date());
            txt_keterangan.setText("");
            txt_akun1.setSelectedItem("");
            txt_akun2.setSelectedItem("");
            txt_no_dokumen.setText("");

            t_Debet1.setText("0");
            t_Debet2.setText("0");

            t_Kredit1.setText("0");
            t_Kredit2.setText("0");

            ttl_debet.setText("0");
            ttl_kredit.setText("0");
            ttl_balance.setText("0");
        }
private void t_Kredit2ActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        Balance ();
    }

    private void btn_tambahActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        this.kondisi = "Tambah";
        this.no_kas_keluar = "";
        bersih_Inputan();
        aktif_Button();

        txt_no_kas_keluar.setText(kas_keluar.NoKasKeluarAuto());
        txt_no_dokumen.requestFocus();
    }
    private void btn_simpanActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        if ("".equals(txt_no_kas_keluar.getText())
                || txt_tanggal_kas_keluar.getDate() == null
                || "".equals(txt_keterangan.getText())
                || "".equals(txt_no_dokumen.getText())
                || "".equals(txt_akun1.getSelectedItem().toString())
                || "".equals(txt_akun2.getSelectedItem().toString())) {
            JOptionPane.showMessageDialog(rootPane, "Mohon isi dulu semua data",
"Error", JOptionPane.ERROR_MESSAGE);
            return;
        }
        if (Double.parseDouble(ttl_balance.getText()) > 0.0) {
            JOptionPane.showMessageDialog(rootPane, "Kas keluar belum balance,
silahkan sseimbangkan dahulu lajur debet dan kredit", "Rrror",
                    JOptionPane.ERROR_MESSAGE);
            return;
        }
```

```
        kas_keluar.setNo_kas_keluar(txt_no_kas_keluar.getText());
        kas_keluar.setTanggal_Kas_Keluar(new java.sql.Date(new
java.util.Date().getTime()));
        kas_keluar.setKeterangan(txt_keterangan.getText());
        kas_keluar.setNo_dokumen(txt_no_dokumen.getText());
        kas_keluar.setDiberikan_kepada(txt_diberikan_kepada.getText());

        if ("Tambah".equals(kondisi)) {
            String[] Akun;
            if (kas_keluar.tambahKasKeluar()> 0) {
                detail_kas_keluar.setNoKasKeluar(txt_no_kas_keluar.getText());

                Akun = txt_akun1.getSelectedItem().toString().split("-");
                detail_kas_keluar.setKode_Perkiraan(Akun[0]);
                detail_kas_keluar.setDebet(Double.parseDouble(t_Debet1.getText()));
                detail_kas_keluar.setKredit(Double.parseDouble(t_Kredit1.getText()));
                detail_kas_keluar.tambah_kas_keluar();

                JOptionPane.showMessageDialog(rootPane, "Data kas_keluar berhasil
disimpan!", "Info", JOptionPane.INFORMATION_MESSAGE);
            } else {
                JOptionPane.showMessageDialog(rootPane, "Data kas_keluar gagal
disimpan. \n" + kas_keluar.getPesanError(), "Error",
                        JOptionPane.ERROR_MESSAGE);
                return;
            }
        }
        this.kondisi = "";
        this.no_kas_keluar = "";
        bersih_Inputan();
        aktif_Button();
    }

    private void btn_batalActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        this.kondisi = "";
        this.no_kas_keluar = "";
        bersih_Inputan();
        aktif_Button();
    }

    private void btn_tutupActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        dispose ();
    }

    private void t_Debet1KeyTyped(java.awt.event.KeyEvent evt) {
        // TODO add your handling code here:
        Balance ();
    }

    private void t_Kredit1KeyTyped(java.awt.event.KeyEvent evt) {
        // TODO add your handling code here:
        Balance ();
```

```
}

private void t_Kredit2KeyTyped(java.awt.event.KeyEvent evt) {
    // TODO add your handling code here:
    Balance ();
}

private void t_Kredit2KeyPressed(java.awt.event.KeyEvent evt) {
    // TODO add your handling code here:
    Balance ();
}

private void t_Kredit2KeyReleased(java.awt.event.KeyEvent evt) {
    // TODO add your handling code here:
    Balance ();
}

private void t_Debet1KeyPressed(java.awt.event.KeyEvent evt) {
    // TODO add your handling code here:
    Balance ();
}
private void t_Debet1KeyReleased(java.awt.event.KeyEvent evt) {
    // TODO add your handling code here:
    Balance ();
}
private void t_Debet2KeyPressed(java.awt.event.KeyEvent evt) {
    // TODO add your handling code here:
    Balance ();
}

private void t_Debet2KeyReleased(java.awt.event.KeyEvent evt) {
    // TODO add your handling code here:
    Balance ();
}

private void t_Debet2KeyTyped(java.awt.event.KeyEvent evt) {
    // TODO add your handling code here:
    Balance ();
}

private void t_Kredit1KeyReleased(java.awt.event.KeyEvent evt) {
    // TODO add your handling code here:
    Balance ();
}

private void t_Kredit1KeyPressed(java.awt.event.KeyEvent evt) {
    // TODO add your handling code here:
    Balance ();
}
```

3.5.2. *Blackbox Testing*

Tabel III.21.
Pengujian Program Halaman Admin *Login*

No.	Skenario Pengujian	*Test Case*	Hasil Yang Diharapkan	Hasil Pengujian	Kesimpulan
1.	Id. Pengguna dan Password tidak diisi Kemudian Klik Login	Id. Pengguna (kosong) Password (kosong)	Sistem akan menolak dan menampilkan pesan "Id. Pengguna dan Password tidak boleh kosong"	Sesuai harapan	Valid
2.	Id. Pengguna diisi dan Password tidak diisi Kemudian Klik Login	Id. Pengguna (11150392) Password (kosong)	Sistem menolak akan menolak dan menampilkan pesan "Id. Pengguna dan Password keliru"	Sesuai harapan	Valid
3.	Id. Pengguna dan Password diisi Kemudian Klik Login	Id. Pengguna (11150392) Password (12345)	Sistem akan menerima dan akan masuk ke halaman menu utama	Sesuai harapan	Valid

Tabel.III.22.
Pengujian Program Mengelola Penerimaan Kas

No.	Skenario Pengujian	*Test Case*	Hasil yang Diharapkan	Hasil Pengujian	Kesimpulan
1.	Form tambah data kas masuk jika field isian data kas masuk di kosongkan Kemudian Klik Simpan	No. Dokumen (kosong) Keterangan (kosong) Diterima dari (kosong)	Sistem akan menolak dan menampilkan pesan "Mohon isi dulu semua data"	Sesuai harapan	valid

		Akun debet (kosong) Akun kredit (kosong)			
2.	Form tambah data kas masuk diisi kecuali akun kredit di kosongkan Kemudian Klik Simpan	No. Dokumen (1) Keterangan (Penyewaan kamar) Diterima dari (Tamu) Akun debet (350000) Akun kredit (kosong)	Sistem akan menolak dan menampilkan pesan "Kas masuk belum balance, silahkan seimbangkan dahulu lajur debet dan kredit"	Sesuai harapan	Valid
3.	Form tambah data kas masuk diisi Kemudian Klik Simpan	No. Dokumen (1) Keterangan (Penyewaan kamar) Diterima dari (Tamu) Akun debet (350000) Akun kredit (350000)	Sistem akan menyimpan data	Sesuai harapan	Valid

Tabel.III.23.
Pengujian Program Mengelola Pengeluaran Kas

No.	Skenario Pengujian	*Test Case*	Hasil yang Diharapkan	Hasil Pengujian	Kesimpulan
1.	Form tambah data kas keluar jika field isian data kas keluar di kosongkan Kemudian Klik Simpan	No. Dokumen (kosong) Keterangan (kosong) Diberikan kepada (kosong) Akun debet	Sistem akan menolak dan menampilkan pesan "Mohon isi dulu semua data"	Sesuai harapan	Valid

		(kosong) Akun kredit (kosong)			
2.	Form tambah data kas masuk diisi kecuali akun kredit di kosongkan Kemudian Klik Simpan	No. Dokumen (1) Keterangan (Penyewaan kamar) Diterima dari (Tamu) Akun debet (350000) Akun kredit (kosong)	Sistem akan menolak dan menampilkan pesan "Kas masuk belum balance, silahkan seimbangkan dahulu lajur debet dan kredit"	Sesuai harapan	Valid
3.	Form tambah data kas masuk diisi Kemudian Klik Simpan	No. Dokumen (1) Keterangan (Penyewaan kamar) Diterima dari (Tamu) Akun debet (350000) Akun kredit (350000)	Sistem akan menyimpan data	Sesuai harapan	Valid

Tabel.III.23.
Pengujian Program Mengelola Pengeluaran Kas

No.	Skenario Pengujian	*Test Case*	Hasil yang Diharapkan	Hasil Pengujian	Kesimpulan
1.	Form tambah data kas keluar jika field isian data kas keluar di kosongkan Kemudian Klik Simpan	No. Dokumen (kosong) Keterangan (kosong) Diberikan kepada (kosong) Akun debet (kosong)	Sistem akan menolak dan menampilkan pesan "Mohon isi dulu semua data"	Sesuai harapan	Valid

		Akun kredit (kosong)			
2.	Form tambah data jasa diisi lengkap Kemudian Klik Simpan	Kode jasa (LD1) *Nama jasa* (Penyewaan kamar)	Sistem akan menyimpan data	Sesuai harapan	Valid
3.	Form Ubah data pengguna lalu pilih data yang akan di ubah Kemudian Klik Simpan	Kode jasa (LD1) *Nama jasa* (Penyewaan kamar)	Sistem akan menyimpan data dan menampilkan pesan "Data berhasil di ubah"	Sesuai harapan	Valid
4.	Form Hapus data pengguna lalu pilih data yang akan di hapus Kemudian Klik Hapus	Kode jasa (LD1) *Nama jasa* (Penyewaan kamar)	Sistem akan menampilkan pesan "Anda ingin menghapus data ini?" jika Ya maka akan menampilkan pesan "Data berhasil di hapus"	Sesuai harapan	Valid

Tabel.III.25.
Pengujian Program Mengelola Data Perkiraan

No.	Skenario Pengujian	*Test Case*	Hasil yang Diharapkan	Hasil Pengujian	Kesimpulan
1.	Form tambah data perkiraan jika field isian data perkiraan di kosongkan Kemudian Klik Simpan	Kode akun (kosong) Nama akun (kosong)	Sistem akan menolak dan menampilkan pesan "Mohon isi data dulu"	Sesuai harapan	Valid
2.	Form tambah data perkiraan diisi lengkap	Kode (111) Nama akun (Kas)	Sistem akan menyimpan data	Sesuai harapan	Valid

	Kemudian Klik Simpan	Jenis akun (Kas) Saldo normal (Debet)			
3.	Form Ubah data perkiraan lalu pilih data yang akan di ubah Kemudian Klik Simpan	Kode (111) Nama akun (Kas) Jenis akun (Kas) Saldo normal (Debet)	Sistem akan menyimpan data dan menampilkan pesan "Data berhasil di ubah"	Sesuai harapan	Valid
4.	Form Hapus data perkiraan lalu pilih data yang akan di hapus Kemudian Klik Hapus	Kode (111) Nama akun (Kas) Jenis akun (Kas) Saldo normal (Debet)	Sistem akan menampilkan pesan "Hapus data ini?" jika Ya maka akan menampilkan pesan "Konfirmasi" "Data berhasil di hapus"	Sesuai harapan	Valid

Tabel.III.26.
Pengujian Program *Manager* Keuangan Mengelola Data Pengguna

No.	Skenario Pengujian	*Test Case*	Hasil yang Diharapkan	Hasil Pengujian	Kesimpulan
1.	Form tambah data pengguna jika field isian data pengguna di kosongkan Kemudian Klik Simpan	Id. pengguna (kosong) *Password* (kosong)	Sistem akan menolak dan menampilkan pesan "Mohon isi data dulu semua data"	Sesuai harapan	Valid
2.	Form tambah data pengguna diisi lengkap Kemudian	Id. pengguna (11150392) *Password*	Sistem akan menyimpan data	Sesuai harapan	Valid

	Klik Simpan	(12345)			
3.	Form Ubah data pengguna lalu pilih data yang akan di ubah Kemudian Klik Simpan	Id. pengguna (11150392) *Password* (12345)	Sistem akan menyimpan data dan menampilkan pesan "Data berhasil di ubah"	Sesuai harapan	Valid
4.	Form Hapus data pengguna lalu pilih data yang akan di hapus Kemudian Klik Hapus	Id. pengguna (11150392) *Password* (12345)	Sistem akan menampilkan pesan "Anda ingin menghapus data ini?" jika Ya maka akan menampilkan pesan "Data berhasil di hapus"	Sesuai harapan	Valid
5.	Form Cari data pengguna lalu ketik data pengguna yang akan di cari Kemudian Klik Cari	Id. pengguna (11150392) *Password* (12345)	Sistem akan mncari data pengguna	Sesuai harapan	Valid

3.5.3. Spesifikasi *Hardware* dan *Software*

1. Spesifikasi *Hardware*

Spesifikasi *hardware* yang penulis maksud adalah seperangkat atau elemen elektronik yang membantu *system* penulis usulkan sehingga program yang diusulkan dapat bekerja dengan maksimal. Perangkat keras yang akan disarankan penulis ke perusahaan adalah sebagai berikut:

a. Laptop : ASUS-E202S

b. *Processor* : Intel(R) Celeron(R) CPU N3060 @1.60GHz 1.60GHz

c. *RAM* : 2 GB

d. *Harddisk* : 2 GB

e. Monitor : 11.6 Inch

f. *Keyboard* : 82 keys

2. Spesifikasi *Software*

Perangkat lunak (*Software)* adalah suatu rangkaian atau susunan instruksi dengan urutan-urutan yang benar. Penggunaan suatu perangkat keras pun akan selalu disertai dengan penggunaan perangkat lunak. Adapun perangkat lunak yang penulis gunakan dalam membantu tugas akhir ini adalah sebagai berikut:

a. Sistem Operasi : Minimal Windows 7

b. Bahasa Pemrograman : Java

c. DBMS : MySQL

d. Antivirus : Avast


BINA SARANA
BSI
INFORMATIKA
UNIVERSITAS