

BAB III

ANALISIS DAN PERANCANGAN SISTEM

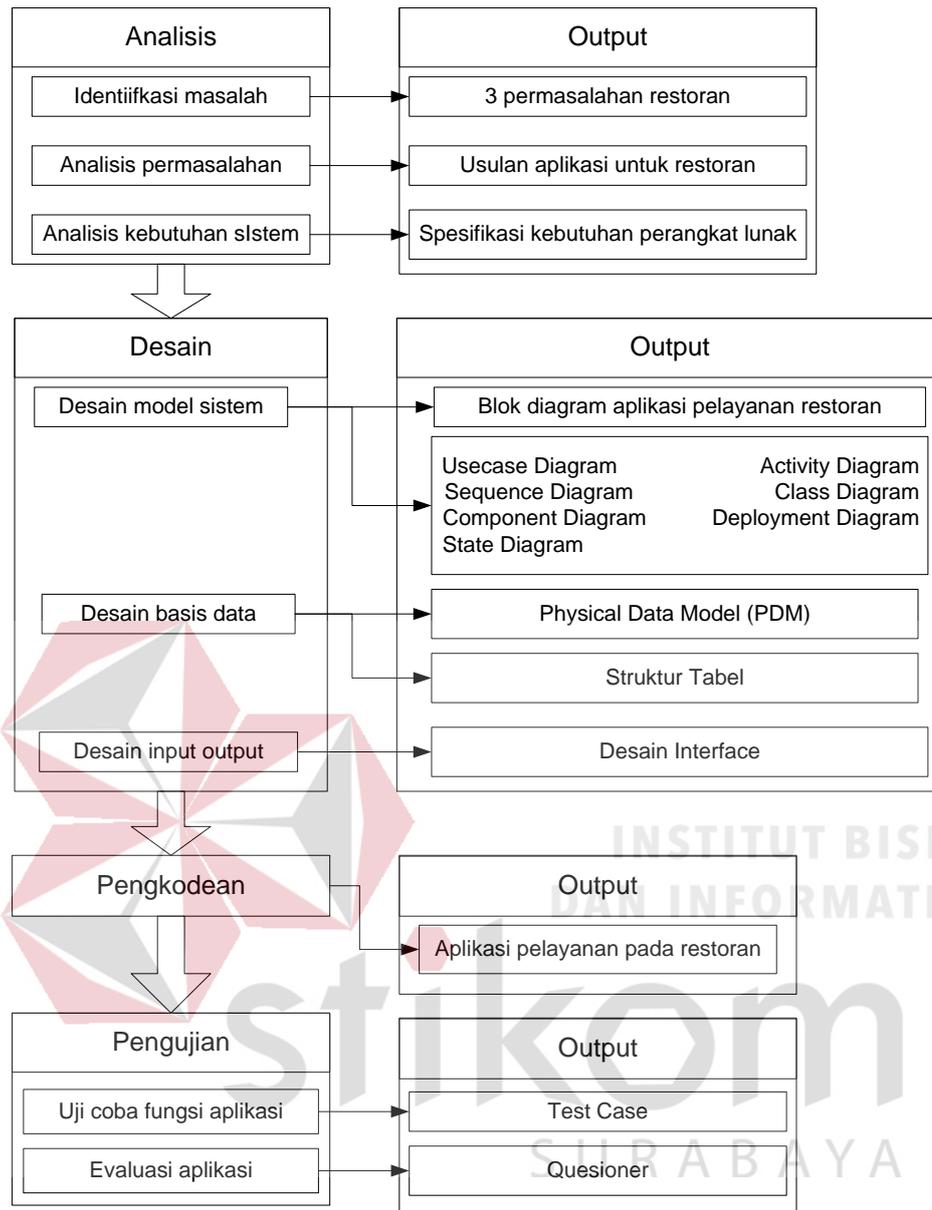
Pada tahap ini dilakukan analisis dan perancangan sistem. Menurut pressman (2001), model *waterfall* adalah model klasik yang bersifat sistematis, berurutan dalam membangun *software*. Disebut dengan *Waterfall* karena tahap demi tahap yang harus dilalui harus menunggu selesainya tahap sebelumnya dan berjalan berurutan. Terdiri dari tahap analisis, desain, pengkodean dan pengujian.

Tahap analisis yaitu proses pengumpulan kebutuhan khususnya pada perangkat lunak. Pada tahap analisis menjelaskan tahap analisis sistem yang didalamnya terdiri dari identifikasi permasalahan, analisis permasalahan dan analisis kebutuhan sistem.

Tahap desain digunakan untuk mengubah kebutuhan-kebutuhan analisis menjadi representasi ke dalam bentuk “blueprint” perangkat lunak. Pada tahap ini, terdiri dari desain model sistem, desain basis data, desain *input output*.

Pengkodean yaitu untuk dapat dimengerti oleh mesin (computer), maka desain harus diubah bentuknya menjadi bentuk yang dapat dimengerti oleh mesin yaitu ke dalam bahasa pemrograman melalui proses *coding*.

Pengujian dilakukan untuk memastikan semua pernyataan sudah diuji, serta menemukan kesalahan-kesalahan dan memastikan inputan dapat memberikan hasil yang sesuai dengan yang dibutuhkan. Untuk lebih jelasnya dapat dilihat pada gambar 3.1.



Gambar 3.1 Tahap Pengembangan Rancang Bangun Aplikasi Pelayanan pada Restoran

3.1 Analisis Sistem

Analisis sistem untuk aplikasi pelayanan restoran pada restoran meliputi identifikasi permasalahan, analisis permasalahan dan analisis kebutuhan sistem.

3.1.1 Identifikasi Permasalahan

Proses pelayanan yang terjadi saat ini yaitu pelayan mencari kursi kosong dengan menengok langsung meja mana yang masih kosong, apabila meja yang berada di lantai 1 (satu) tidak tersedia, maka pelayan berjalan menengok meja yang ada di lantai 2 (dua) pada restoran. Sehingga pelayan mengalami kesulitan dalam pencarian kursi.

Permasalahan yang kedua, pelayan mencatat pesanan *customer* pada selembar kertas, kemudian pelayan memasukkan data pesanan tersebut pada aplikasi desktop yang tersedia dan mencetaknya sebanyak 3 (tiga) lembar. Lembar pertama diberikan kepada *checker*, lembar kedua diberikan kepada *bartender*, lembar ketiga diletakkan pada meja *customer*. Rata-rata jumlah pengunjung dalam kondisi normal adalah ± 100 orang dan dalam kondisi ramai ± 200 orang. Dalam kondisi normal, *customer* mendapati proses pilih menu hingga menerima menu ± 30 menit. Pelayan akan semakin kerepotan pada saat restoran sedang ramai pengunjung, karena berkeliling dari meja *customer* ke meja komputer untuk merangkap pesanan, ke *checker* dan *bartender*.

Sulitnya pelayan dalam mengingat dan mengatur *booking/reservasi* meja dikarenakan data reservasi yang tidak tersusun rapi. Sehingga pelayan kesulitan menentukan dan mencari pesanan meja yang perlu dipersiapkan 1 jam sebelum jam yang ditentukan.

3.1.2 Analisis Permasalahan

Terdapat beberapa permasalahan yang didapatkan dari proses pelayanan pada restoran, diantaranya adalah:

1. Pelayan masih harus menengok langsung atau menyisiri ruangan restoran untuk dapat menemukan meja kosong yang sesuai baik yang ada di lantai 1 (satu) maupun lantai 2 (dua) pada restoran tersebut. Sehingga *customer* harus menunggu hingga pelayan menemukan meja kosong yang sesuai.
2. Pelayan masih menulis pesanan pada selembar kertas. Kemudian menginputkan menu pesanan pada aplikasi *desktop* untuk dirangkapkan 3 (tiga) lembar menu pesanan. Lembar pertama diberikan kepada *checker* , lembar kedua diberikan kepada *bartender*, lembar ketiga diletakkan pada meja *customer*. Pelayan harus benar-benar mengingat menu mana yang pada hari itu sedang kosong. Sehingga pelayan kesulitan dalam melakukan estimasi waktu pelayanan karena masih berkeliling kesana kemari.
3. Pelayan kesulitan menentukan dan mencari pesanan meja yang perlu dipersiapkan 1 jam sebelum jam yang ditentukan. Dengan cara mengingat kembali lembaran jadwal *booking*/pesanan yang tertempel pada meja kasir.

Setelah dilakukan identifikasi permasalahan, maka diperoleh gambaran mengenai hal-hal yang dapat membantu menyelesaikan permasalahan yang terjadi. Diantaranya adalah:

1. Membuat aplikasi yang dapat menampilkan denah meja kosong maupun terisi. Sehingga pelayan tidak perlu menyisiri ruangan baik yang ada di lantai 1 (satu) maupun lantai 2 (dua) pada restoran.
2. Membuat aplikasi yang dapat menampilkan menu pesanan, yang pada saat itu tersedia, menampilkan informasi stok menu yang habis, dan total harga pesanan.

3. Membuat aplikasi yang dapat mencatat menu pesanan yang dapat terintegrasi ke bagian *checker*, dan kasir.
4. Membuat aplikasi yang dapat menampilkan menu pesanan pada *bartender* dan *chef* yang dikontrol oleh *checker*.
5. Membuat aplikasi yang dapat menangani proses pembayaran berdasarkan nomor meja.
6. Membuat aplikasi yang dapat menangani proses penjadwalan *booking*/pemesanan meja. Sehingga pelayan segera mengetahui meja yang sedang di-booking.
7. Membuat aplikasi yang dapat memberikan informasi berupa laporan penjualan, laporan menu favorit, dan laporan *utility* meja berdasarkan harian maupun bulanan.

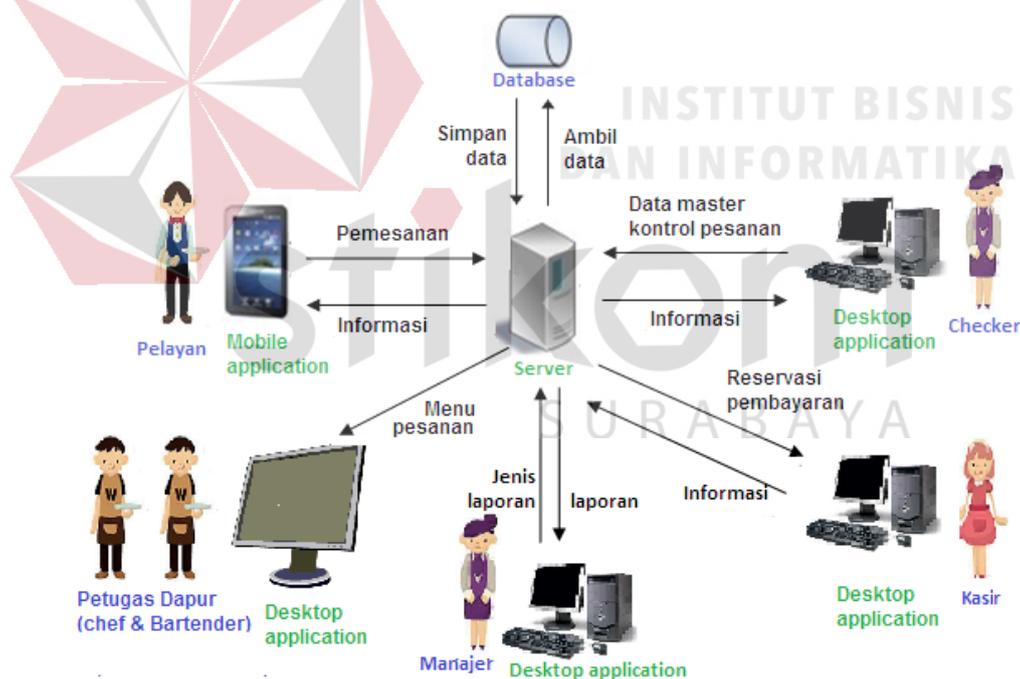
3.1.3 Analisis Kebutuhan Sistem

Berdasarkan permasalahan diatas maka dibutuhkan sistem aplikasi yang diantaranya dapat melakukan:

1. Proses Login:
 - a. Mengisi id dan password *login* berdasar hak akses user
2. Mengolah data master:
 - a. Menyimpan, ubah, hapus data master user
 - b. Menyimpan, ubah, hapus data master menu
 - c. Menyimpan, ubah, hapus data master ruangan
 - d. Mengatur denah meja per-ruangan
3. Memasukkan stok menu harian

- a. Mengisi jumlah stok per item menu
- b. Mengisi jumlah stok secara keseluruhan
4. Proses pemilihan meja:
 - a. Menampilkan denah meja berdasarkan ruangan
 - b. Menampilkan tanda meja yang isi dan kosong
 - c. Menandai meja sementara saat customer sedang pilih menu
 - d. Menggabungkan meja dan pindah meja
5. Proses mencatat menu pesanan :
 - a. Menampilkan menu berdasarkan jenis menu
 - b. Menampilkan informasi menu yang stoknya habis
 - c. Menampilkan harga pemesanan menu
 - d. Merubah pesanan (menambah dan mengurangi jumlah menu)
 - e. Mencatat pemesanan menu spesial
 - f. Mengirimkan *list* pesanan ke *checker*
 - g. Menampilkan status pesanan
6. Proses *checking* pesanan
 - a. Menampilkan *list* pesanan
 - b. Merubah status pesanan
 - c. Menampilkan riwayat pesanan
7. Proses pembayaran:
 - a. Menampilkan biaya yang harus dibayarkan berdasarkan nomor meja
 - b. Memotong total pembayaran dengan menggunakan voucher
 - c. Melakukan pembayaran
 - d. Mencetak struk

8. Proses *reservasi*:
 - a. Menyimpan data reservasi
 - b. Menghapus data reservasi
 - c. Mengubah data reservasi
 - d. Memilih meja
 - e. Menandai meja reservasi
9. Proses membuat laporan
 - a. Memilih jenis laporan
 - b. Menampilkan laporan harian
 - c. Menampilkan laporan bulanan



Gambar 3.2 Arsitektur Aplikasi Pelayanan pada Restoran Berbasis Android

Terdapat 2 (dua) aplikasi yang akan dibangun seperti yang terlihat pada gambar 3.2, yaitu *mobile application* untuk pelayan dengan menggunakan tablet

dan *desktop application* menggunakan komputer yang dioperasikan *checker*, kasir, petugas dapur (*bartender dan chef*) dan manajer. Tablet dan komputer tersebut terhubung dengan *server* lokal yang berisikan *database* dengan menggunakan jaringan *Wifi*. Semua data yang masuk disimpan didalam *database* milik *server* lokal.

User pelayan bertugas mencari meja apabila *customer* kesulitan mencari meja sesuai dengan yang dibutuhkan, mencatat menu pesanan *customer*, serta menyiapkan meja yang sudah *dibooking/dipesan* pada 1 (satu) jam sebelum jadwal agar meja tersebut tidak dapat digunakan oleh *customer* lain. *Mobile application* yang diakses oleh pelayan dapat menampilkan denah meja (tampak meja isi/kosong/terpesan sesuai dengan jenis ruangan, menampilkan menu tersedia, mencatat menu pesanan yang kemudian sistem akan otomatis mengirimkan menu pesanan ke *Checker* dan dapat melihat jadwal *booking/pemesanan* meja. Sistem akan otomatis menampilkan meja yang dipesan/ *booking* pada 1 (satu) jam sebelum jadwal dengan tanda meja berwarna "ungu" dan data pemesan (nama pemesan dan jam pemesanan)

User Checker bertugas dalam mengisi data master (menu, *user*, ruangan), mengisi data stok menu harian yang tersedia setiap harinya, mengontrol antrian menu pesanan (menampilkan beberapa pesanan ke layar bagian Dapur dan melakukan pengecekan menu pesanan yang sudah dibuatkan oleh petugas dapur). *Desktop application* pada *checker* berfungsi sebagai *maintenance* data master, pengisian stok menu harian guna menentukan jumlah stok menu yang akan disediakan, pengontrolan pesanan (merubah status pesanan "menunggu" menjadi "Proses" agar dapat menampilkan pesanan pada layar bagian dapur dan merubah

status pesanan "selesai" agar pesanan yang selesai dibuat tidak tampil pada layar bagian dapur). *Checker* dapat mengecek riwayat dari menu pesanan yang sudah selesai dibuat.

User bagian dapur bertugas hanya dalam membuat menu pesanan. Bagian dapur cukup hanya melihat menu pesanan pada layar bagian dapur (*chef* dan atau *bartender*). *Desktop application* yang ada dibagian dapur hanya digunakan untuk melihat menu pesanan yang akan dibuatkan. Menu-menu yang muncul pada layar dapur dikontrol oleh *checker*. Menu pesanan yang muncul pada layar dapur hanya beberapa *item* pesanan. Sebelum jam operasional dimulai, petugas dapur (*chef* dan *bartender*) dapat memilih jenis tampilan sesuai kebutuhan pada "form tampilan dapur" diantaranya adalah: makanan dan minuman digabung, makanan dan minuman dipisah, makanan saja dan minuman saja

User kasir bertugas dalam melakukan proses pembayaran dan menangani proses *reservasi/booking/pemesanan* meja. Pemesanan dapat dilakukan via telepon yang dilayani oleh petugas kasir maupun langsung di tempat. Aplikasi *dekstop* pada kasir berfungsi sebagai transaksi pembayaran dan transaksi *reservasi* dengan cara mencatat data pemesanan (nama pemesan, jadwal, lokasi meja dan no telepon yang bisa dihubungi). Jadwal *reservasi* otomatis akan muncul meja terblokir warna "ungu" pada denah meja di *mobile application* pada 1 (satu) jam sebelum jadwal pemesanan.

Serta *user* manajer bertugas dalam melihat laporan-laporan baik laporan harian maupun bulanan yang diperoleh dari data yang tersimpan pada *database*. Manajer dapat memilih jenis laporan: penjualan, *utility* meja dan menu favorit. *Desktop application* yang ada pada manajer berfungsi untuk memilih jenis laporan

yang diinginkan seperti laporan penjualan, laporan utility meja dan menu favorit baik harian maupun bulanan.

3.2 Perancangan Sistem

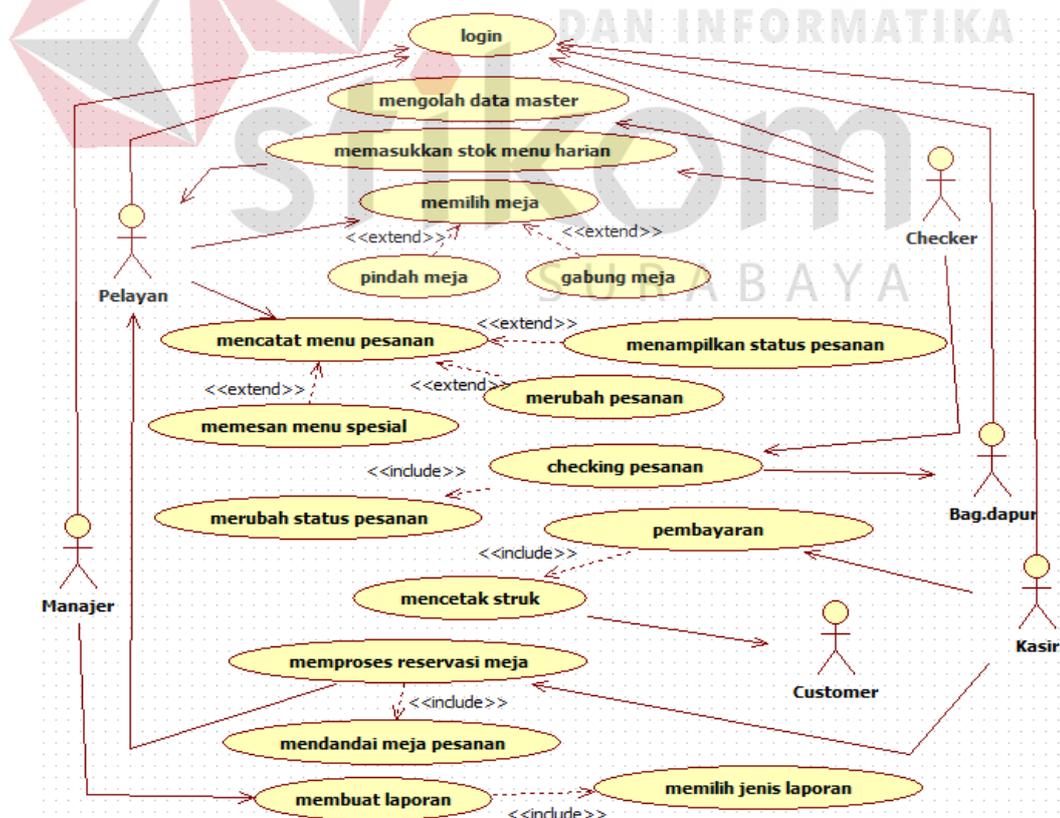
Perancangan sistem adalah tahap untuk memberikan gambaran yang jelas dari rancangan aplikasi yang akan dibuat, sehingga memudahkan pemahaman mengenai sistem yang dibangun. Tahap perancangan sistem ini meliputi: UML (meliputi *use Case Diagram*, *Activity Diagram*, *Sequence Diagram*, *Class Diagram*, *Statechart Diagram*, *Component Diagram*, *Deployment Diagram*, *Statechart diagram*), *Physical Data Model* (PDM), struktur tabel, Rancangan Desain *Input* dan *Output* dan rancangan questioner

3.2.1 Use Case Diagram

Use case diagram menyajikan interaksi antar *use case* dan *actor*. *Use case* digunakan untuk mengetahui yang terdapat didalam sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi tersebut. Dalam tahap ini, penggambaran *use case* tampak pada gambar 3.3.

Setelah melakukan analisa terhadap sistem, diketahui bahwa restoran memiliki pegawai pelayan, checker, bagian dapur, kasir dan manager, serta melayani pelanggan dalam proses bisnisnya. Untuk mencari actor, dilakukan identifikasi yang ada di dalam ruang lingkup (*Business Worker*) dan berada di ruang lingkup (*Business Actor*). Setelah melakukan identifikasi, ditemukan satu *Business Actor* yaitu pelanggan dan ditemukan 5 *Business Worker* yaitu pelayan, checker, bagian dapur, kasir dan manager.

Pelayan bertugas mencarikan meja kosong untuk pelanggan, kemudian mencatat menu pesanan, selanjutnya mengirimkan *list* pesanan tersebut ke *checker* dan *bartender*. Pelayan dapat menunjukkan *list* pesanan yang sudah di pesan *customer*. Pelayan mencatat nomor meja jika pelanggan melakukan pindah meja atau menggabung meja. *Checker* bertugas mengontrol stok dan mengontrol pesanan (menentukan menu yang dibuat oleh bagian dapur). Bagian dapur menerima *list* menu pesanan yang harus dibuat untuk diproses. Kasir bertugas menerima pembayaran dan mencatat *reservasi*. Sedangkan manajer bertugas membuat laporan. Dari uraian diatas, dapat diidentifikasi beberapa *usecase*, yaitu login, mengolah data master, memilih meja, mencatat menu pesanan, *checking* pesanan, pembayaran, *reservasi* meja, membuat laporan. Setelah ditemukan actor dan *usecase*, maka dapat digambar *usecase diagram* seperti pada gambar 3.3.



Gambar 3.3 Usecase Diagram Aplikasi Pelayanan pada Restoran

Berikut adalah penjelasan singkat dari masing-masing *use case diagram* aplikasi pelayanan pada restoran:

Tabel 3.1 Tabel *Flow of Event Usecase Login*

| | | |
|---------------------------------|--|--|
| <i>Use case Name</i> | Login | |
| <i>Brief Description</i> | Use case ini mengatur proses login user | |
| <i>Primary Actor</i> | Pelayan Manajer Checker Bag. Dapur Kasir | |
| <i>Secondary Actor</i> | - | |
| <i>Pre-Condition</i> | - | |
| <i>Post-Condition</i> | User masuk ke dalam sistem | |
| <i>Included Use case</i> | - | |
| <i>Basic Flow of Events</i> | <i>Actor's Action</i> | <i>Sistem's Response</i> |
| | 1. User memasukkan username dan password kemudian mengklik tombol login | 2. Sistem mengecek username dan password apakah sudah benar dengan cara mengambil data sesuai username yang dimasukkan user dari database dan membandingkan apakah password yang dimasukkan user setelah dienkripsi dengan MD5 sama dengan password yang tersimpan pada database 3. Jika username dan password benar, sistem menampilkan tampilan utama |
| <i>Alternate Flow of Events</i> | 3a. Jika username dan password salah, maka sistem akan menampilkan tampilan login dengan informasi login gagal | |
| <i>Extension Points</i> | - | |

Tabel 3.2 Tabel *Flow of Event Usecase Mengolah Data Master*

| | |
|--------------------------|---|
| <i>Use case Name</i> | Mengolah data master |
| <i>Brief Description</i> | Use case ini mengatur proses memasukkan, mengubah dan menghapus data master |
| <i>Primary Actor</i> | Checker |
| <i>Secondary Actor</i> | - |
| <i>Pre-Condition</i> | Checker sudah login ke dalam sistem |
| <i>Post-Condition</i> | Data master tersimpan dalam sistem |

| | | |
|---------------------------------|---|--|
| <i>Included Use case</i> | - | |
| <i>Basic Flow of Events</i> | <i>Actor's Action</i> | <i>Sistem's Response</i> |
| | 1. Checker memilih menu data master 3. Checker memasukkan, mengubah atau menghapus data master | 2. Sistem menampilkan form data master 4. Sistem menyimpan/menghapus data master pada database 5. Jika sistem berhasil menyimpan/menghapus data master, data akan muncul/hilang pada tabel |
| <i>Alternate Flow of Events</i> | 5a. Jika sistem gagal menyimpan/menghapus data master, maka sistem akan menampilkan pesan kesalahan | |
| <i>Extension Points</i> | - | |

Tabel 3.3 Tabel *Flow of Event Usecase* Memasukkan Stok Menu Harian

| | | |
|---------------------------------|--|--|
| <i>Use case Name</i> | Memasukkan stok menu harian | |
| <i>Brief Description</i> | Use case ini untuk memasukkan stok menu setiap hari | |
| <i>Primary Actor</i> | Checker | |
| <i>Secondary Actor</i> | Pelayan | |
| <i>Pre-Condition</i> | -Checker sudah login ke dalam sistem -Stok hari ini belum dimasukkan | |
| <i>Post-Condition</i> | Data stok tersimpan dalam sistem | |
| <i>Included Use case</i> | - | |
| <i>Basic Flow of Events</i> | <i>Actor's Action</i> | <i>Sistem's Response</i> |
| | 1. Checker memilih menu stok 3. Checker memasukkan jumlah stok tiap menu dan mengklik tombol simpan | 2. Sistem menampilkan tabel stok hari ini yang datanya diambil dari tabel menu dan diberi nilai awal setiap stok sejumlah 0. 4. Sistem menyimpan data stok menu yang memiliki jumlah stok lebih dari 0 ke database. 5. Jika sistem berhasil menyimpan data stok, data akan tampil pada tabel dan tombol simpan serta set semua tidak dapat digunakan |
| <i>Alternate Flow of Events</i> | 5a. Jika sistem gagal menyimpan data stok, maka sistem akan menampilkan pesan kesalahan | |

| | |
|-------------------------|---|
| <i>Extension Points</i> | - |
|-------------------------|---|

Tabel 3.4 Tabel *Flow of Event Usecase* memilih meja

| | | |
|---------------------------------|--|--|
| <i>Use case Name</i> | Memilih meja | |
| <i>Brief Description</i> | Use case ini digunakan untuk memilih meja yang akan digunakan oleh <i>customer</i> | |
| <i>Primary Actor</i> | Pelayan | |
| <i>Secondary Actor</i> | - | |
| <i>Pre-Condition</i> | -Pelayan sudah login ke dalam sistem -Data master ruangan dan meja sudah diisi oleh checker | |
| <i>Post-Condition</i> | Meja terpilih dan diberi penanda warna merah | |
| <i>Included Use case</i> | - | |
| <i>Basic Flow of Events</i> | <i>Actor's Action</i> | <i>Sistem's Response</i> |
| | 1. Pelayan menyentuh gambar meja dan mengklik tombol pilih meja | 2. Sistem menyimpan data meja sesuai dengan meja yang dipilih. Meja ditandai dengan status terpakai. 3. Sistem menampilkan halaman pemesanan menu |
| <i>Alternate Flow of Events</i> | - | |
| <i>Extension Points</i> | -Pindah meja -Gabung meja | |

Tabel 3.5 Tabel *Flow of Event Usecase* Mencatat Menu Pesanan

| | | |
|-----------------------------|---|---|
| <i>Use case Name</i> | Mencatat menu pesanan | |
| <i>Brief Description</i> | Use case ini digunakan untuk melakukan proses pemesanan menu | |
| <i>Primary Actor</i> | Pelayan | |
| <i>Secondary Actor</i> | - | |
| <i>Pre-Condition</i> | -Pelayan sudah login ke dalam system -Data master menu sudah diisi oleh checker -Data stok harian sudah diisi oleh checker -Pelayan sudah memilih meja | |
| <i>Post-Condition</i> | Menu pesanan <i>customer</i> tersimpan dalam di dalam system | |
| <i>Included Use case</i> | - | |
| <i>Basic Flow of Events</i> | <i>Actor's Action</i> | <i>Sistem's Response</i> |
| | 1. Pelayan mengklik menu yang ingin dipesan dan mengklik tombol simpan | 2. Sistem meverifikasi menu pesanan dengan cara mengecek apakah jumlah stok tersisa lebih |

| | | |
|---------------------------------|--|--|
| | | <p>besar/sama dengan jumlah yang dipesan</p> <p>3. Sistem menyimpan data pesanan pada database</p> <p>4. Jika pesanan tersimpan, sistem akan menampilkan halaman utama</p> |
| <i>Alternate Flow of Events</i> | <p>2a. Jika menu pesanan kehabisan stok, maka sistem akan menampilkan pesan kesalahan</p> <p>3a. Jika sistem gagal menyimpan pesanan, maka sistem akan menampilkan pesan kesalahan</p> | |
| <i>Extension Points</i> | <p>-Memesan menu special</p> <p>-Merubah pesanan</p> <p>-Menampilkan status pesanan</p> | |

Tabel 3.6 Tabel *Flow of Event Usecase* Memesan Menu Spesial

| | | |
|---------------------------------|--|---|
| <i>Use case Name</i> | Memesan menu spesial | |
| <i>Brief Description</i> | Use case ini digunakan untuk memberi catatan pada pesanan <i>customer</i> jika <i>customer</i> menginginkan menu dengan penanganan khusus | |
| <i>Primary Actor</i> | Pelayan | |
| <i>Secondary Actor</i> | - | |
| <i>Pre-Condition</i> | Pelayan telah mengklik menu yang diinginkan | |
| <i>Post-Condition</i> | Catatan menu khusus tersimpan dalam system | |
| <i>Included Use case</i> | - | |
| <i>Basic Flow of Events</i> | <i>Actor's Action</i> | <i>System's Response</i> |
| | <p>1. Pelayan mengklik tombol jumlah pesanan pada menu yang ingin ditambahkan catatan</p> <p>3. Pelayan memasukkan catatan menu special dan mengklik tombol ubah</p> | <p>2. Sistem menampilkan dialog pesanan special</p> <p>4. Sistem menyimpan catatan menu special pada list pesanan</p> |
| <i>Alternate Flow of Events</i> | - | |
| <i>Extension Points</i> | - | |

Tabel 3.7 Tabel *Flow of Event Usecase* Menampilkan Status Pesanan

| | |
|--------------------------|--|
| <i>Use case Name</i> | Menampilkan status pesanan |
| <i>Brief Description</i> | Use case ini digunakan untuk menampilkan pesanan <i>customer</i> |

| | | |
|---------------------------------|--|---|
| <i>Primary Actor</i> | Pelayan | |
| <i>Secondary Actor</i> | Checker | |
| <i>Pre-Condition</i> | Pelayan telah melakukan pemesanan menu | |
| <i>Post-Condition</i> | Tampil halaman list pesanan | |
| <i>Included Use case</i> | - | |
| <i>Basic Flow of Events</i> | <i>Actor's Action</i> | <i>Sistem's Response</i> |
| | 1. Pelayan memilih meja yang telah dipakai dan mengklik tombol lihat meja | 2. Sistem mengambil data pesanan pada database berdasarkan id meja yang dipilih 3. Sistem menampilkan list pesanan <i>customer</i> |
| <i>Alternate Flow of Events</i> | 2a. Jika sistem gagal mengambil data pesanan, maka sistem akan menampilkan pesan kesalahan | |
| <i>Extension Points</i> | - | |

Tabel 3.8 Tabel *Flow of Event Usecase* Gabung Meja

| | | |
|-----------------------------|--|--|
| <i>Use case Name</i> | Gabung meja | |
| <i>Brief Description</i> | Use case ini digunakan untuk melakukan penggabungan meja | |
| <i>Primary Actor</i> | Pelayan | |
| <i>Secondary Actor</i> | - | |
| <i>Pre-Condition</i> | -Pelayan telah melakukan pemesanan menu -Pelayan telah masuk ke halaman list pesanan | |
| <i>Post-Condition</i> | Meja yang digabung akan disimpan di sistem dan diberi tanda merah | |
| <i>Included Use case</i> | - | |
| <i>Basic Flow of Events</i> | <i>Actor's Action</i> | <i>Sistem's Response</i> |
| | 1. Pelayan mengklik tombol gabung meja 3. Pelayan memilih meja yang akan digabung dan mengklik tombol pilih | 2. Sistem akan menampilkan halaman peta meja 4. Sistem menyimpan data meja yang digabung pada database. Meja ditandai dengan status terpakai. 5. Jika sistem berhasil menyimpan data meja digabung, meja yang digabung (berstatus terpakai) akan bertanda merah. 6. Sistem menampilkan halaman list pesanan |
| <i>Alternate Flow of</i> | 4a. Jika sistem gagal menyimpan data meja digabung, maka | |

| | |
|-------------------------|---|
| <i>Events</i> | sistem akan menampilkan pesan kesalahan |
| <i>Extension Points</i> | - |

Tabel 3.9 Tabel *Flow of Event Usecase* Pindah Meja

| | | |
|---------------------------------|--|--|
| <i>Use case Name</i> | Pindah meja | |
| <i>Brief Description</i> | Use case ini digunakan untuk melakukan pindah meja | |
| <i>Primary Actor</i> | Pelayan | |
| <i>Secondary Actor</i> | - | |
| <i>Pre-Condition</i> | -Pelayan telah melakukan pemesanan menu -Pelayan telah masuk ke halaman list pesanan | |
| <i>Post-Condition</i> | Meja berpindah dari meja awal ke meja yang dipilih (penanda merah pada meja awal hilang dan muncul pada meja yang dipilih) | |
| <i>Included Use case</i> | - | |
| <i>Basic Flow of Events</i> | <i>Actor's Action</i> | <i>Sistem's Response</i> |
| | 1. Pelayan mengklik tombol gabung meja 3. Pelayan memilih meja yang akan ditempati dan mengklik tombol pilih | 2. Sistem akan menampilkan halaman peta meja 4. Sistem menyimpan data meja yang dipindah dan mengubah data meja lama. Meja lama diupdate dengan status selesai dan meja baru ditandai dengan status dipakai 5. Jika sistem berhasil menyimpan data meja dipindah, meja yang dipindah (berstatus dipakai) akan ditandai merah dan meja lama (berstatus selesai) tidak ditandai merah. 6. Sistem menampilkan halaman list pesanan |
| <i>Alternate Flow of Events</i> | 4a. Jika sistem gagal menyimpan data meja dipindah, maka sistem akan menampilkan pesan kesalahan | |
| <i>Extension Points</i> | - | |

Tabel 3.10 Tabel *Flow of Event Usecase* Mengubah Pesanan

| | |
|--------------------------|---|
| <i>Use case Name</i> | Mengubah pesanan |
| <i>Brief Description</i> | Use case ini dilakukan untuk mengubah pesanan |
| <i>Primary Actor</i> | Pelayan |

| | | |
|---------------------------------|---|--|
| <i>Secondary Actor</i> | - | |
| <i>Pre-Condition</i> | -Pelayan telah melakukan pemesanan menu -Pelayan telah masuk ke halaman list pesanan | |
| <i>Post-Condition</i> | Pesanan yang disimpan sistem berubah sesuai perubahan yang dilakukan | |
| <i>Included Use case</i> | - | |
| <i>Basic Flow of Events</i> | <i>Actor's Action</i> | <i>Sistem's Response</i> |
| | 1. Pelayan menambah atau mengurangi pesanan | 2. Sistem meverifikasi menu pesanan dengan mengecek apakah jumlah yang ditambah lebih kecil/sama dengan sisa stok menu jika pelayan melakukan penambahan. Jika melakukan pengurangan, maka tidak dilakukan verifikasi 3. Sistem menyimpan data pesanan yang sudah diupdate pada database 4. Jika pesanan tersimpan, sistem akan menampilkan halaman list pesanan |
| <i>Alternate Flow of Events</i> | 2a. Jika menu pesanan kehabisan stok, maka sistem akan menampilkan pesan kesalahan 3a. Jika sistem gagal menyimpan pesanan, maka sistem akan menampilkan pesan kesalahan | |
| <i>Extension Points</i> | - | |

Tabel 3.11 Tabel *Flow of Event Usecase Checking Pesanan*

| | | |
|-----------------------------|--|---|
| <i>Use case Name</i> | Checking pesanan | |
| <i>Brief Description</i> | Use case ini digunakan untuk mengontrol pesanan | |
| <i>Primary Actor</i> | Checker | |
| <i>Secondary Actor</i> | Bag. Dapur | |
| <i>Pre-Condition</i> | Pelayan sudah melakukan pemesanan menu | |
| <i>Post-Condition</i> | Status pesanan sesuai dengan yang ditentukan checker | |
| <i>Included Use case</i> | Mengubah status pesanan | |
| <i>Basic Flow of Events</i> | <i>Actor's Action</i> | <i>Sistem's Response</i> |
| | 1. Checker mengatur tampilan tabel pesanan dengan mengklik judul tabel | 2. Sistem mengambil data pesanan hari ini dari database dan menampilkan data sesuai dengan pengaturan checker |
| <i>Alternate Flow of</i> | - | |

| | |
|-------------------------|---|
| <i>Events</i> | |
| <i>Extension Points</i> | - |

Tabel 3.12 Tabel *Flow of Event Usecase* Mengubah Status Pesanan

| | | |
|---------------------------------|--|---|
| <i>Use case Name</i> | Mengubah status pesanan | |
| <i>Brief Description</i> | Use case ini digunakan untuk mengubah status pesanan sesuai dengan kondisi actual | |
| <i>Primary Actor</i> | Checker | |
| <i>Secondary Actor</i> | Dapur | |
| <i>Pre-Condition</i> | - | |
| <i>Post-Condition</i> | Status pesanan berubah sesuai dengan status saat ini | |
| <i>Included Use case</i> | - | |
| <i>Basic Flow of Events</i> | <i>Actor's Action</i> | <i>Sistem's Response</i> |
| | 1. Checker memilih pesanan yang ingin diubah statusnya dan mengklik tombol ubah status | 2. Sistem mengubah status pesanan. Jika status saat ini adalah menunggu, maka akan diubah menjadi proses. Jika status saat ini adalah proses maka diubah 3. Jika status berhasil diubah, sistem akan menampilkan data pada tabel |
| <i>Alternate Flow of Events</i> | 2a. Jika sistem gagal mengubah status pesanan, sistem akan menampilkan pesan kesalahan | |
| <i>Extension Points</i> | - | |

Tabel 3.13 Tabel *Flow of Event Usecase* Pembayaran

| | | |
|-----------------------------|---|---|
| <i>Use case Name</i> | Pembayaran | |
| <i>Brief Description</i> | Use case ini digunakan dalam proses pembayaran | |
| <i>Primary Actor</i> | Kasir | |
| <i>Secondary Actor</i> | - | |
| <i>Pre-Condition</i> | Status seluruh pesanan <i>customer</i> telah diubah menjadi selesai | |
| <i>Post-Condition</i> | -Pembayaran disimpan dan meja diset menjadi tidak terpakai | |
| <i>Included Use case</i> | -Mencetak struk | |
| <i>Basic Flow of Events</i> | <i>Actor's Action</i> | <i>Sistem's Response</i> |
| | 1. Kasir memasukkan nomor meja dan mengklik tombol cari | 2. Sistem akan mencari dan mengambil data pesanan berdasarkan nomor meja yang diinputkan 3. Sistem menampilkan data pesanan, harga tiap menu dan |

| | | |
|---------------------------------|---|--|
| | 4. Kasir memasukkan jumlah pembayaran dan mengklik tombol bayar | menghitung total harga 5. Sistem menyimpan data pembayaran pada database dan mengubah status data meja dari dipakai menjadi selesai 6. Sistem menampilkan struk pembayaran |
| <i>Alternate Flow of Events</i> | 2a. Jika nomor meja yang dicari tidak memiliki pesanan, maka sistem akan menampilkan pesan kesalahan 4a. Jika kasir memasukkan jumlah pembayaran yang salah, maka sistem akan menampilkan pesan kesalahan 5a. Jika sistem gagal menyimpan data pembayaran, maka sistem akan menampilkan pesan kesalahan | |
| <i>Extension Points</i> | - | |

Tabel 3.14 Tabel *Flow of Event Usecase* Mencetak Struk

| | | |
|---------------------------------|---|--------------------------|
| <i>Use case Name</i> | Mencetak struk | |
| <i>Brief Description</i> | Use case ini digunakan untuk melakukan cetak struk | |
| <i>Primary Actor</i> | Kasir | |
| <i>Secondary Actor</i> | - | |
| <i>Pre-Condition</i> | Kasir telah melakukan proses pembayaran | |
| <i>Post-Condition</i> | Struk tercetak | |
| <i>Included Use case</i> | - | |
| <i>Basic Flow of Events</i> | <i>Actor's Action</i> | <i>Sistem's Response</i> |
| | 1. Kasir mengklik tombol cetak saat sistem menampilkan struk pada use case memproses pembayaran | 2. Sistem mencetak struk |
| <i>Alternate Flow of Events</i> | - | |
| <i>Extension Points</i> | - | |

Tabel 3.15 Tabel *Flow of Event Usecase* Memproses Reservasi Meja

| | | |
|--------------------------|---|--------------------------|
| <i>Use case Name</i> | Memproses reservasi meja | |
| <i>Brief Description</i> | Use case ini digunakan untuk melakukan proses reservasi | |
| <i>Primary Actor</i> | Kasir | |
| <i>Secondary Actor</i> | Pelayan | |
| <i>Pre-Condition</i> | - | |
| <i>Post-Condition</i> | Data reservasi tersimpan dalam sistem | |
| <i>Included Use case</i> | Menandai meja pesanan | |
| <i>Basic Flow of</i> | <i>Actor's Action</i> | <i>Sistem's Response</i> |

| | | |
|---------------------------------|---|--|
| <i>Events</i> | 1. Kasir memasukkan data reservasi | 2. Sistem meverifikasi data reservasi dengan mengecek apakah meja yang akan dipesan sudah dipesan atau belum dan apakah pesanan dengan meja yang sama sudah ada tapi pada rentang lebih dari 3 jam 3. Jika meja dan waktu yang diinginkan tersedia, maka sistem menyimpan data reservasi pada database 4. Sistem menampilkan data reservasi pada tabel |
| <i>Alternate Flow of Events</i> | 2a. Jika waktu dan atau meja yang dipilih sudah dipesan sebelumnya, maka sisem akan menampilkan pesan kesalahan 3a. Jika sistem gagal menyimpan reservasi, maka sisem akan menampilkan pesan kesalahan | |
| <i>Extension Points</i> | - | |

Tabel 3.16 Tabel *Flow of Event Usecase* Menandai Meja

| | | |
|---------------------------------|---|--|
| <i>Use case Name</i> | Menandai meja pesanan | |
| <i>Brief Description</i> | Use case ini digunakan untuk menandai meja yang telah digunakan | |
| <i>Primary Actor</i> | Kasir | |
| <i>Secondary Actor</i> | Pelayan | |
| <i>Pre-Condition</i> | Data reservasi telah tersimpan pada system | |
| <i>Post-Condition</i> | Data meja yang dipesan disimpan pada system | |
| <i>Included Use case</i> | - | |
| <i>Basic Flow of Events</i> | <i>Actor's Action</i> | <i>Sistem's Response</i> |
| | 1. Kasir melakukan reservasi sesuai use case memproses reservasi meja | 2. Sistem akan memberikan tanda warna ungu pada meja yang statusnya dibooking dan waktu saat ini berada pada rentang 1 jam sebelum jam booking sampai jam booking. |
| <i>Alternate Flow of Events</i> | - | |
| <i>Extension Points</i> | - | |

Tabel 3.17 Tabel *Flow of Event Usecase* Membuat Laporan

| | |
|--------------------------|--|
| <i>Use case Name</i> | Membuat laporan |
| <i>Brief Description</i> | Use case ini digunakan untuk membuat laporan |

| | | |
|---------------------------------|--|---|
| <i>Primary Actor</i> | Manager | |
| <i>Secondary Actor</i> | - | |
| <i>Pre-Condition</i> | Manager telah login ke dalam system | |
| <i>Post-Condition</i> | Tampil laporan | |
| <i>Included Use case</i> | Memilih jenis laporan | |
| <i>Basic Flow of Events</i> | <i>Actor's Action</i> | <i>Sistem's Response</i> |
| | 1. Manajer memilih menu laporan yang diinginkan 3. Manajer memasukkan data filter yang diinginkan dan mengklik tombol lihat laporan | 2. Sistem menampilkan filter laporan 4. Sistem mengambil data dari database sesuai filter dan menampilkan dalam bentuk laporan |
| <i>Alternate Flow of Events</i> | - | |
| <i>Extension Points</i> | - | |

Tabel 3.18 Tabel *Flow of Event Usecase* Memilih Jenis Laporan

| | | |
|---------------------------------|--|---------------------------------------|
| <i>Use case Name</i> | Memilih jenis laporan | |
| <i>Brief Description</i> | Use case ini digunakan untuk memilih jenis laporan yang akan dilihat | |
| <i>Primary Actor</i> | Manajer | |
| <i>Secondary Actor</i> | - | |
| <i>Pre-Condition</i> | -Manajer telah memilih menu laporan yang diinginkan Sistem telah menampilkan filter laporan | |
| <i>Post-Condition</i> | Jenis laporan terpilih | |
| <i>Included Use case</i> | | |
| <i>Basic Flow of Events</i> | <i>Actor's Action</i> | <i>Sistem's Response</i> |
| | 1. Manajer memilih jenis laporan dari salah satu pilihan pada combobox | 2. Filter terset sesuai jenis laporan |
| <i>Alternate Flow of Events</i> | - | |
| <i>Extension Points</i> | - | |

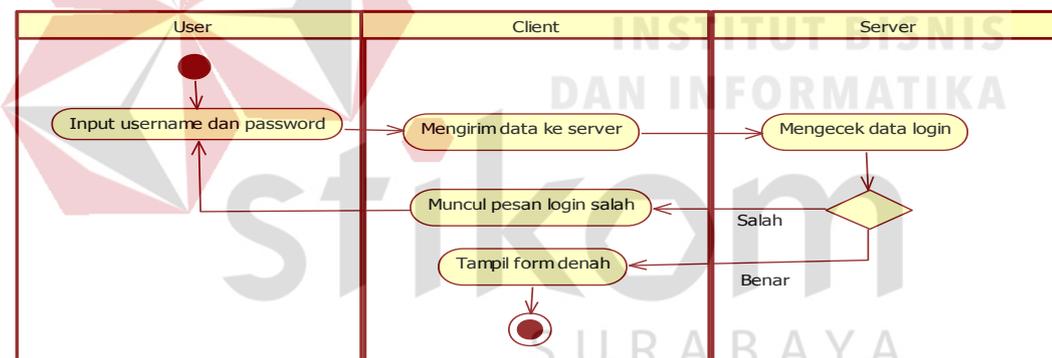
3.2.2 Activity Diagram

Activity Diagram atau Diagram aktivitas menggambarkan aliran kerja/ aktifitas dari sistem atau proses bisnis, bukan apa yang dilakukan oleh aktor.

Berikut beberapa penjelasan singkat mengenai *activity diagram* proses yang berkaitan dengan pelayanan restoran:

a. Activity Diagram Proses Login Mobile Application

Proses *Login* diawali oleh *user* pelayan memasukkan data *username* dan *password*. Data inputan dikirim oleh *mobile application* ke *server* dengan menggunakan jaringan internet lokal. *Server* mengecek data inputan tersebut apakah ada atau tidak pada *database* user. Jika data *login* tidak sesuai maka *mobile application* menampilkan pesan bahwa *login* gagal dan meminta untuk memasukkan *username* dan *password* yang benar. Jika data *login* yang dimasukkan sudah benar maka *mobile application* menampilkan denah meja. Untuk lebih jelasnya dapat dilihat pada gambar 3.4.

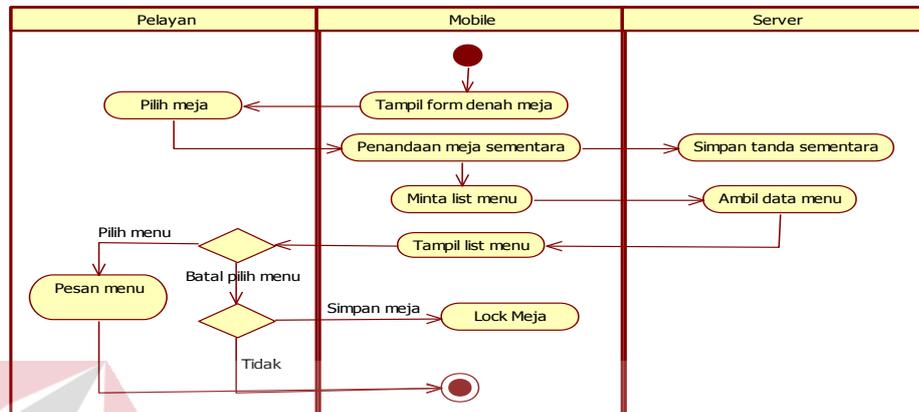


Gambar 3.4 Activity Diagram Proses Login

b. Activity Diagram Proses Pilih Meja

Mobile application menampilkan denah meja, pelayan memilih meja yang artinya *mobile application* menandai meja sementara untuk disimpan pada *server*. *Mobile application* meminta list menu pada *server*, *server* memberikan data menu yang tersedia sehingga tampil *list* menu pada *mobile application*. Apabila pelayan memilih menu maka dilakukan proses pemesanan menu. Jika

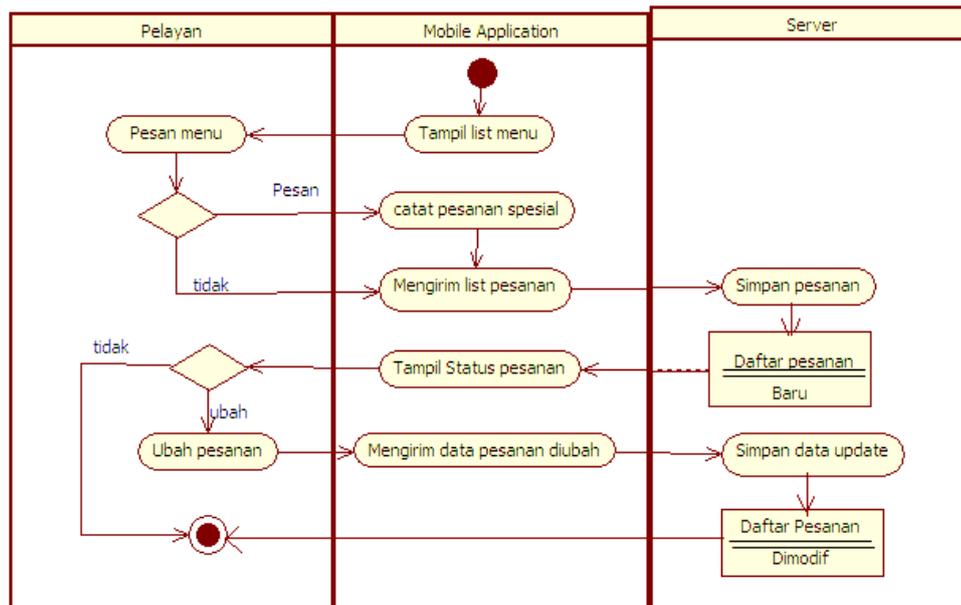
pelayan memilih batal pilih menu kemudian memilih simpan meja, maka sistem akan melakukan *lock* (penguncian tanda merah pada meja), jika tidak disimpan maka akan keluar dari sistem yang artinya tanda sementara akan hilang.



Gambar 3.5 Activity Diagram Proses Pilih Meja

c. Activity Diagram Proses Pemesanan Menu

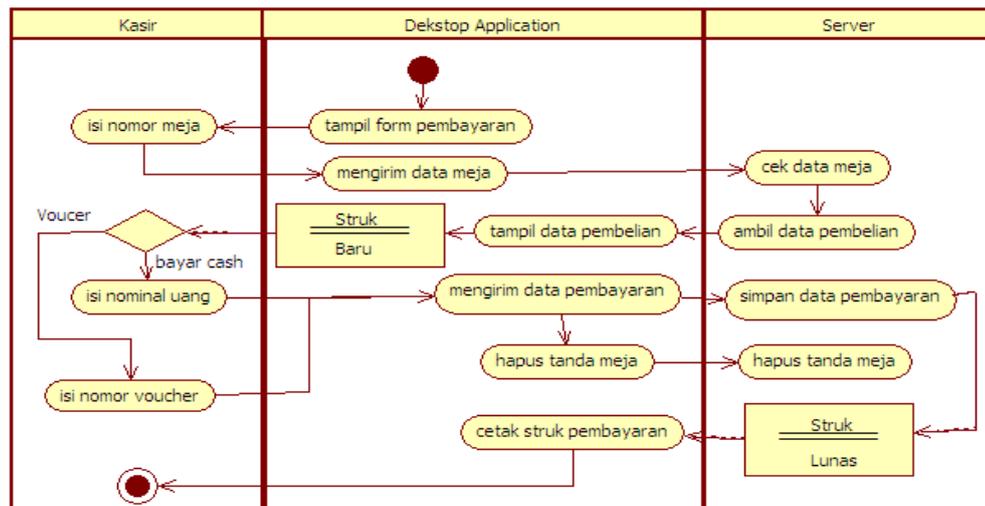
Mobile application menampilkan *list* menu yang tersedia, kemudian pelayan melakukan pemesanan menu (mencatat pesanan). Apabila ada pesanan menu spesial maka dilakukan pencatatan pesanan spesial jika tidak pesan maka sistem akan mengirimkan *list* pesanan ke *server* untuk disimpan. Kemudian menghasilkan entitas bisnis daftar pesanan baru. *Mobile application* akan menampilkan status pesanan. Apabila pelayan tidak ingin mengubah pesanan maka proses pemesanan dapat diakhiri. Jika ingin mengubah pesanan maka sistem akan mengirimkan data pesanan yang dirubah ke *server* untuk dilakukan simpan data update, maka menghasilkan entitas bisnis daftar pesanan modif. Alur proses seperti pada gambar 3.6.



Gambar 3.6 Activity Diagram Proses Pemesanan

d. Activity Diagram Proses Pembayaran

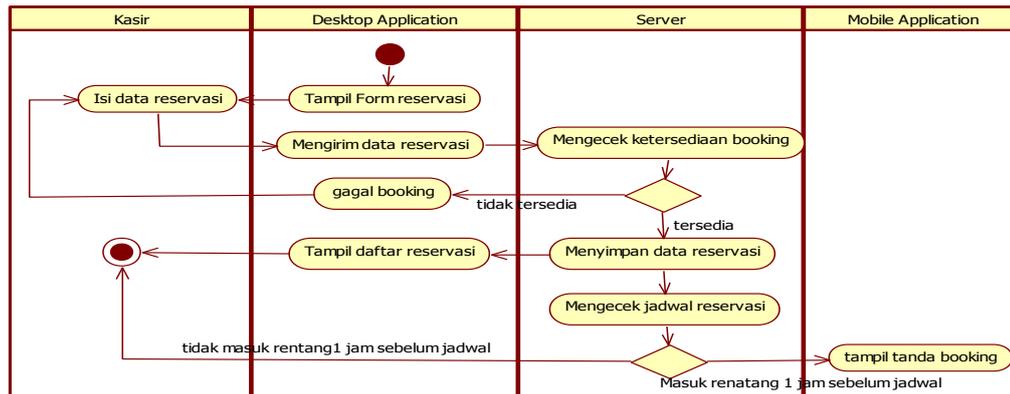
Proses pembayaran dilakukan oleh kasir dengan menggunakan aplikasi *desktop*. Aplikasi *desktop* menampilkan form pembayaran, petugas kasir mengisi nomor meja, kemudian aplikasi *desktop* mengirim data meja tersebut ke *server*. *Server* mengecek data meja yang diterima kemudian *server* mengambil data pembelian. Data pembelian tersebut tampil pada aplikasi *desktop*, sehingga menghasilkan entitas bisnis struk baru. Bila membayar dengan uang *cash* maka diminta memasukkan nominal uang yang diterima kemudian sistem akan mengirimkan data pembayaran ke *server*. *Server* menyimpan data pembayaran tersebut yang menghasilkan entitas bisnis struk lunas. Kemudian sistem dapat melakukan cetak struk pembayaran. Apabila menggunakan voucher maka dilakukan pengisian nomor voucher. Gambar dapat dilihat pada gambar 3.7.



Gambar 3.7 Activity Diagram Proses Pembayaran

e. Activity Diagram Proses Reservasi /Booking Meja

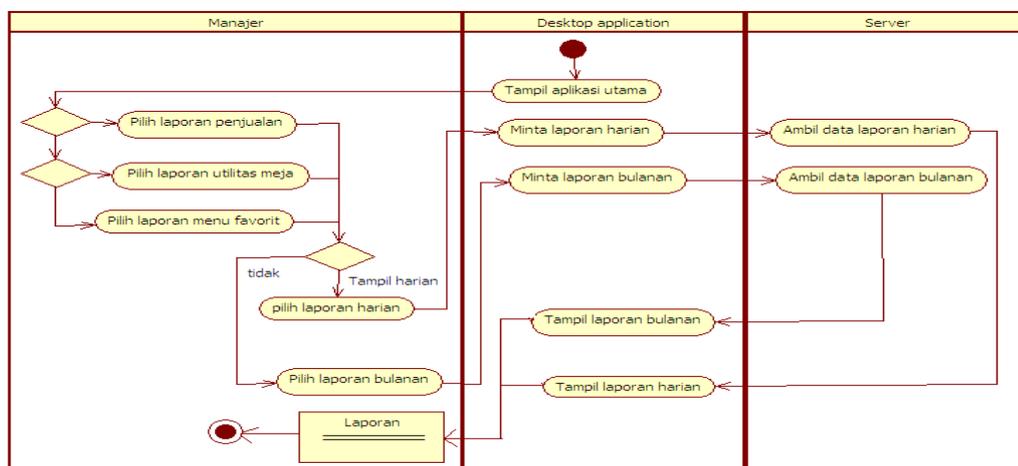
Proses *reservasi* dilakukan oleh petugas kasir. Petugas kasir melayani *reservasi* melalui telepon atau bisa juga dengan melayani secara langsung di restoran. *Desktop application* akan menampilkan form *reservasi*, kemudian pelayan menginputkan data yang diperlukan untuk dilakukan *reservasi/ booking/ pesan meja*. *Desktop application* mengirimkan data *reservasi* tersebut ke *server* untuk dilakukan pengecekan ketersediaan *booking*. Jika tidak tersedia, maka akan gagal *reservasi*. Jika tersedia *server* menyimpan data *reservasi* tersebut, kemudian menampilkan daftar *reservasi* pada list *reservasi*. Server melakukan pengecekan jadwal *reservasi*, apabila masuk pada 1(satu) jam sebelum jadwal yang dipesan, maka akan tampil tanda meja sedang *booking/ dipesan* pada *mobile application* dengan tanda warna ungu. Meja yang sama hanya bisa di *booking* dengan beda waktu 3 (tiga) jam. Dapat dilihat pada gambar 3.8.



Gambar 3.8 Activity Diagram Proses *Reservasi /Booking* Meja

f. Activity Diagram Proses Laporan

Proses pembuatan laporan dilakukan oleh manajer. *Desktop application* menampilkan tampilan aplikasi utama. Jika manager memilih laporan penjualan, kemudian pilih jenis laporan harian, maka *desktop application* akan meminta laporan harian ke *server*. *Server* mengambil data laporan, menampilkan laporan harian pada *desktop application*. Jika manager memilih laporan berdasarkan bulanan, maka akan meminta laporan harian pada *server*, *server* mengambil data laporan bulanan yang muncul pada *desktop application*. Laporan bulanan dan harian menghasilkan entitas bisnis laporan. Gambar terlihat pada gambar 3.9.



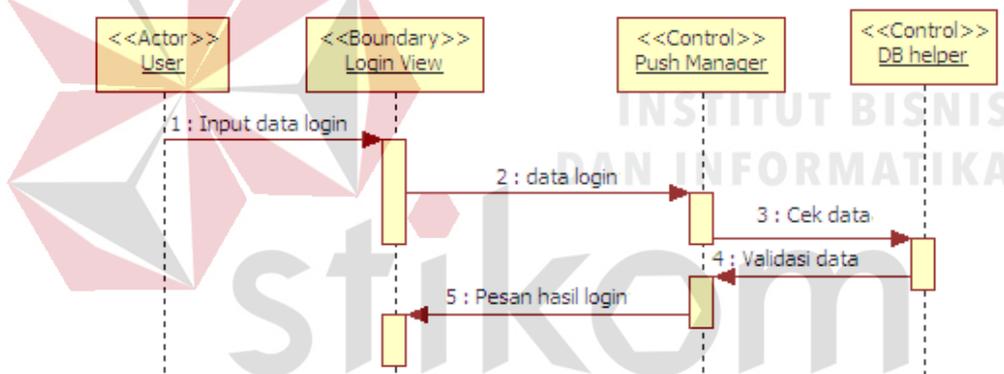
Gambar 3.9 Activity Diagram Proses Laporan

3.2.3 Sequence Diagram

Sequence diagram menggambarkan kegiatan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan *message* yang dikirim dan diterima antar objek

a. Sequence Diagram Proses Login

User (Pelayan / *Checker* / Bagian Dapur / Kasir / Manajer) melakukan proses *login* pada *login view* setiap akan menggunakan aplikasi *mobile/desktop*. Proses *login* dilakukan dengan cara *input data login* (nama & password). *Database* mengecek data login tersebut, ke *Db helper* kemudian memvalidasi data yang menghasilkan pesan *login*.

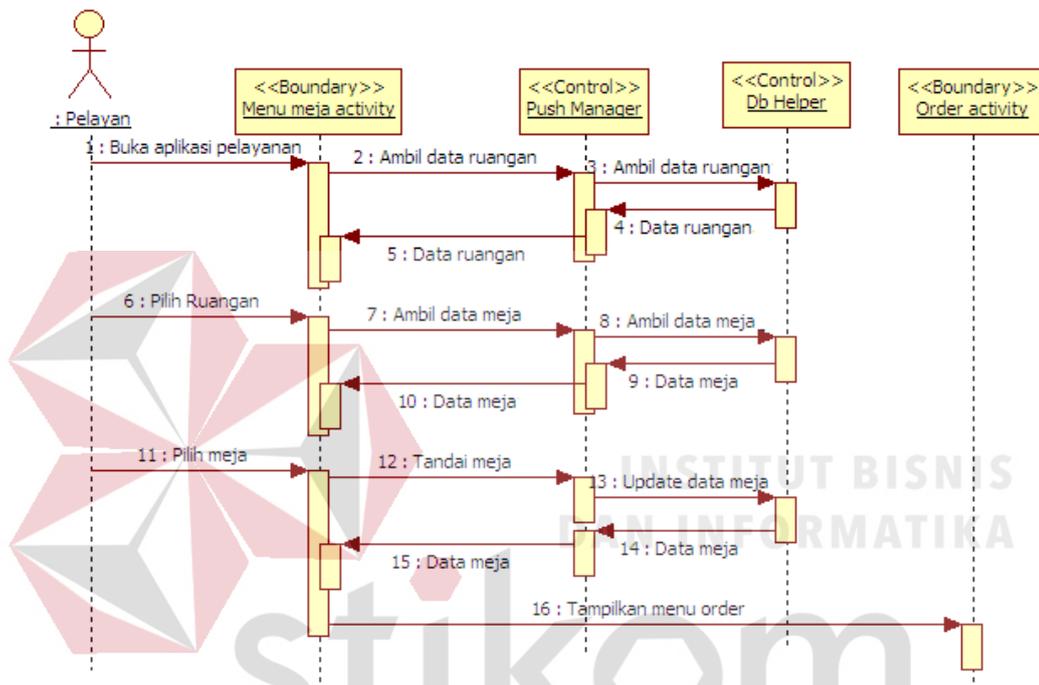


Gambar 3.10 *Sequence Diagram* Proses Login

b. Sequence Diagram Proses Pilih Meja

Pelayan membuka aplikasi pelayanan pada form menu meja, kemudian dilakukan ambil data ruangan, push manager ambil data ruangan pada *Db Helper*. *Db Helper* mengirim data ruangan ke push manager untuk dikirimkan ke form menu meja. Pelayan memilih ruangan, form menu meja mengambil data meja. Push manager mengambil data meja pada *db helper*. *Db helper* mengirimkan data meja ke push manager yang kemudian diterukan ke form menu meja. Pelayan

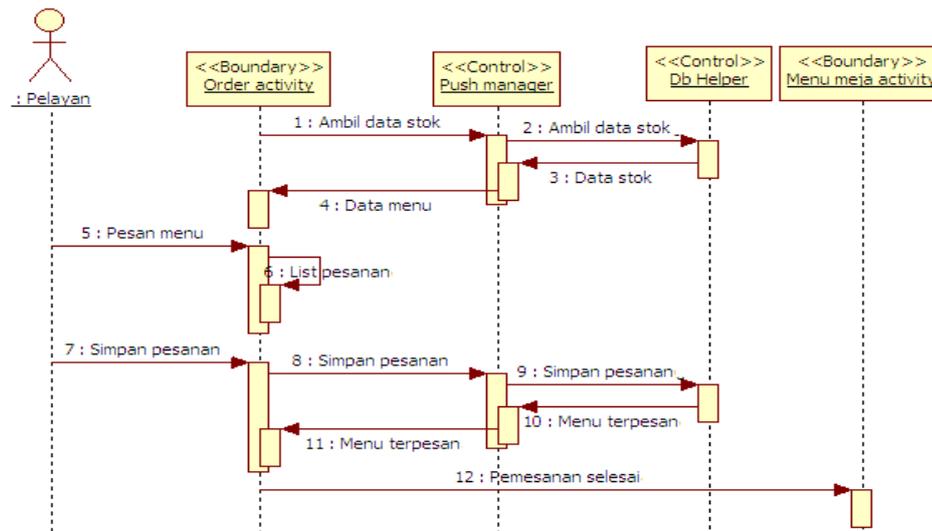
memilih meja pada form menu meja. Kemudian dilakukan penandaan meja pada push manager . Db helper melakukan update data meja yang menghasilkan data meja. Data meja di kirim ke push manajer untuk di tampilkan pada form menu meja. Form menu meja mengirimkan tampilan menu order pada form *order* Seperti pada gambar 3.11.



Gambar 3.11 Sequence Diagram Proses Pilih meja

c. Sequence Diagram Proses Pemesanan Menu

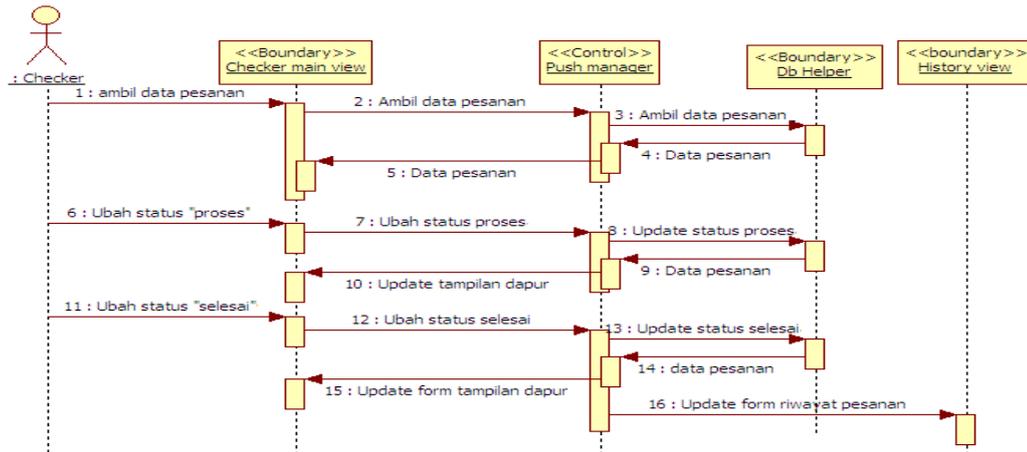
Form *order* mengambil data stok pada push manager kemudian mengambil data *stok* pada Db helper. Db helper mengirim data *stok* pada push manager kemudian ke form *order*. Pelayan memesan menu pada form *order*, kemudian menampilkan *list* pesanan. Pelayan menyimpan pesanan pada form *order* kemudian dikirimkan ke push manager dan Db helper. Db helper mengirimkan menu terpesan pada push manager dan form *order*. Form *order* mengirimkan pemesanan selesai pada form menu meja.



Gambar 3.12 Sequence Diagram Proses Pilih Meja dan Pesan Menu

d. Sequence Diagram Proses Checking Pesanan

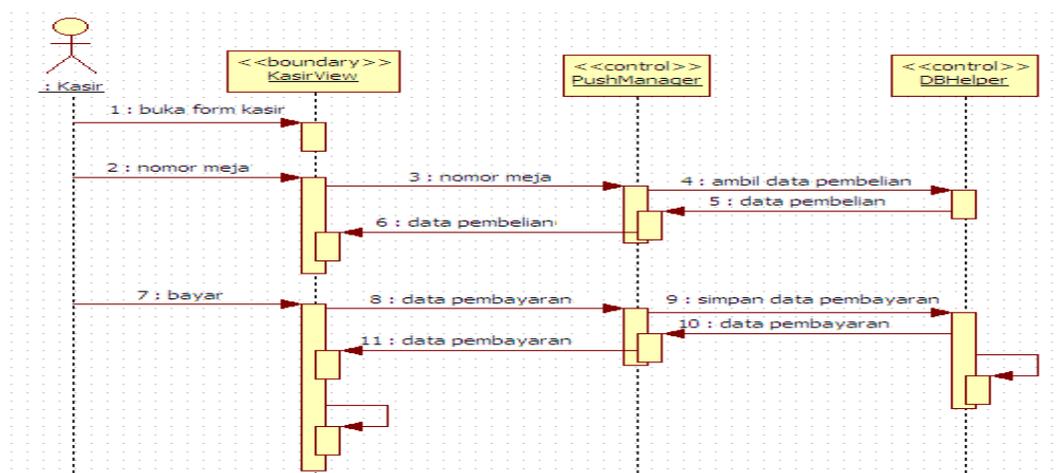
Checker mengambil data pesanan pada form chacker main view, kemudian ke push manager dan Db Helper. Db helper mengirimkan data pesanan kemudian ke push manager dan pada form chacker main view. Pelayan mengubah status pesanan “proses” pada form cheicking, kemudian dikirimkan ke push manager dan db helper melakukan update status pesanan. Db helper mengirimkan data pesanan ke push manager dan form checking melakukan update tampilan dapur. *Checker* mengubah status pesanan “selesai” pada form *checking*, push manager dan db helper melakukan update status kemudian mengirimkan data pesanan pada push manager dan form *checking* melakukan update form tampilan dapur. Push manager update form riwayat pesnan pada form histori.



Gambar 3.13 Sequence Diagram Proses Checking Pesanan

e. Sequence Diagram Proses Pembayaran

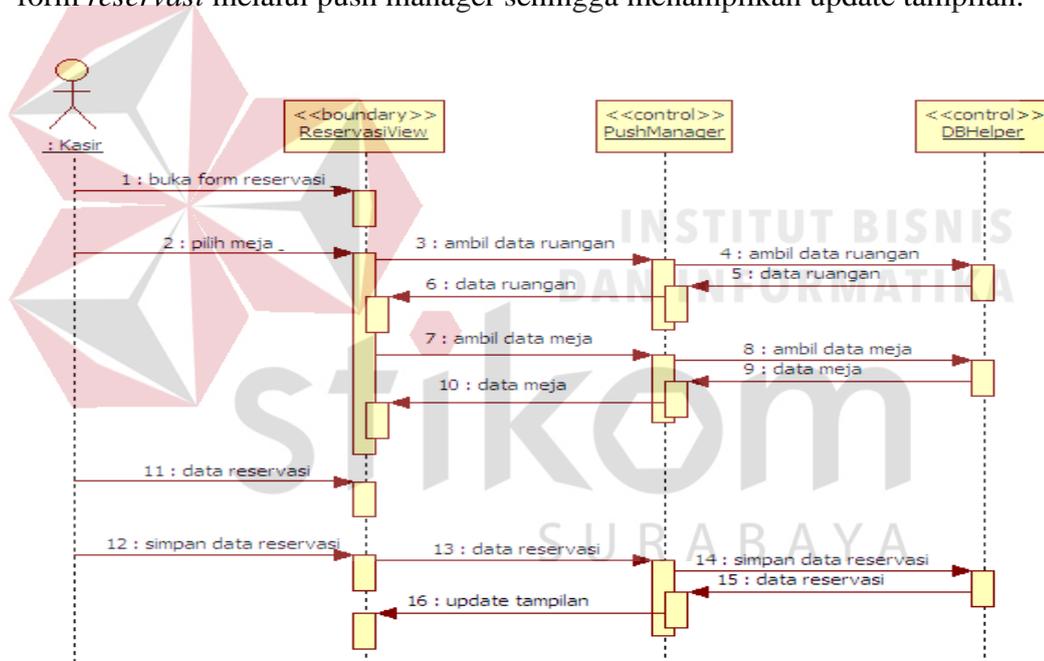
Petugas kasir membuka form pembayaran, kemudian *input* nomor meja pada form kasir. Nomor meja dikirim ke push manager dan db helper mengambil data pembelian. Data pembelian dikirim ke push manager dan form kasir view. Kasir melakukan pembayaran, menghasilkan data pembayaran untuk dikirim ke push manager dan db helper melakukan simpan data pembayaran. Db helper mengirimkan data pembayaran beserta melakukan update status meja. Kemudian muncul data pembelian pada form kasir dan dilakukan cetak struk.



Gambar 3.14 Sequence Diagram Proses Pembayaran

f. Sequence Diagram Proses *Reservasi*

Petugas kasir membuka form *reservasi*. Memilih meja pada form *reservasi*, kemudian mengambil data ruangan pada push manager dan db helper. Db helper mengirimkan data ruangan ke push manager dan form *reservasi*. Kemudian mengambil data meja ke db helper dan menghasilkan data meja. Data meja dikirimkan ke form *reservasi*. Kasir mengisi data reservasi kemudian menyimpan data reservasi. Data *reservasi* dikirim push manager ke db helper untuk dilakukan simpan data reservasi. Db helper menampilkan data reservasi ke form *reservasi* melalui push manager sehingga menampilkan update tampilan.

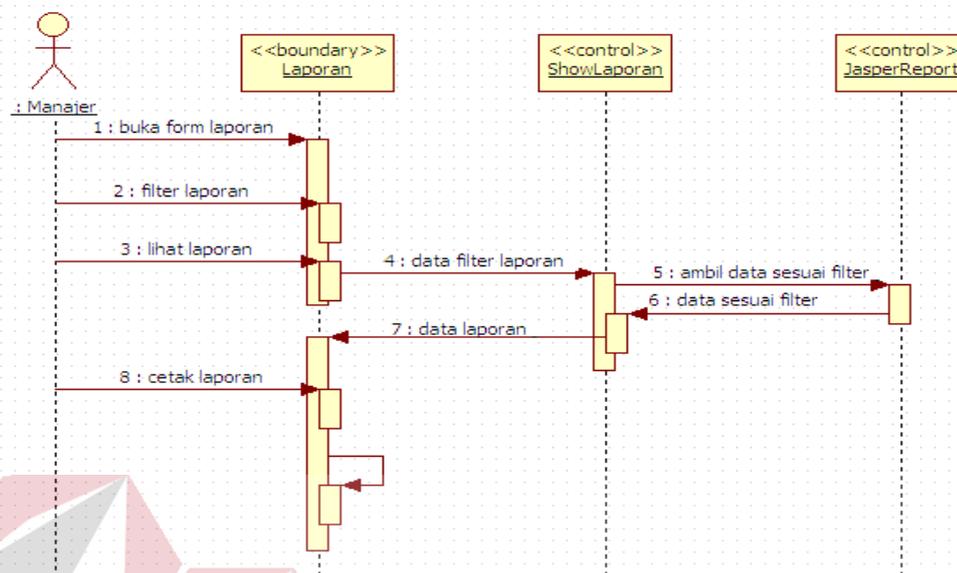


Gambar 3.15 Sequence Diagram Proses *Reservasi*

g. Sequence Diagram Proses Pembuatan laporan

Manajer membuka form laporan dan memfilter laporan. Kemudian lihat laporan pada form laporan. Form laporan mengirim data filter laporan pada show laporan. Jasper report mengambil data sesuai filter yang diterima untuk kemudian

dikirimkan data sesuai filter dan ditampilkan data laporan pada form laporan. Pelayan melakukan cetak laporan, maka form laporan mencetak laporan tersebut.



Gambar 3.16 *Sequence Diagram* Proses Pembuatan Laporan

3.2.4 Class Diagram

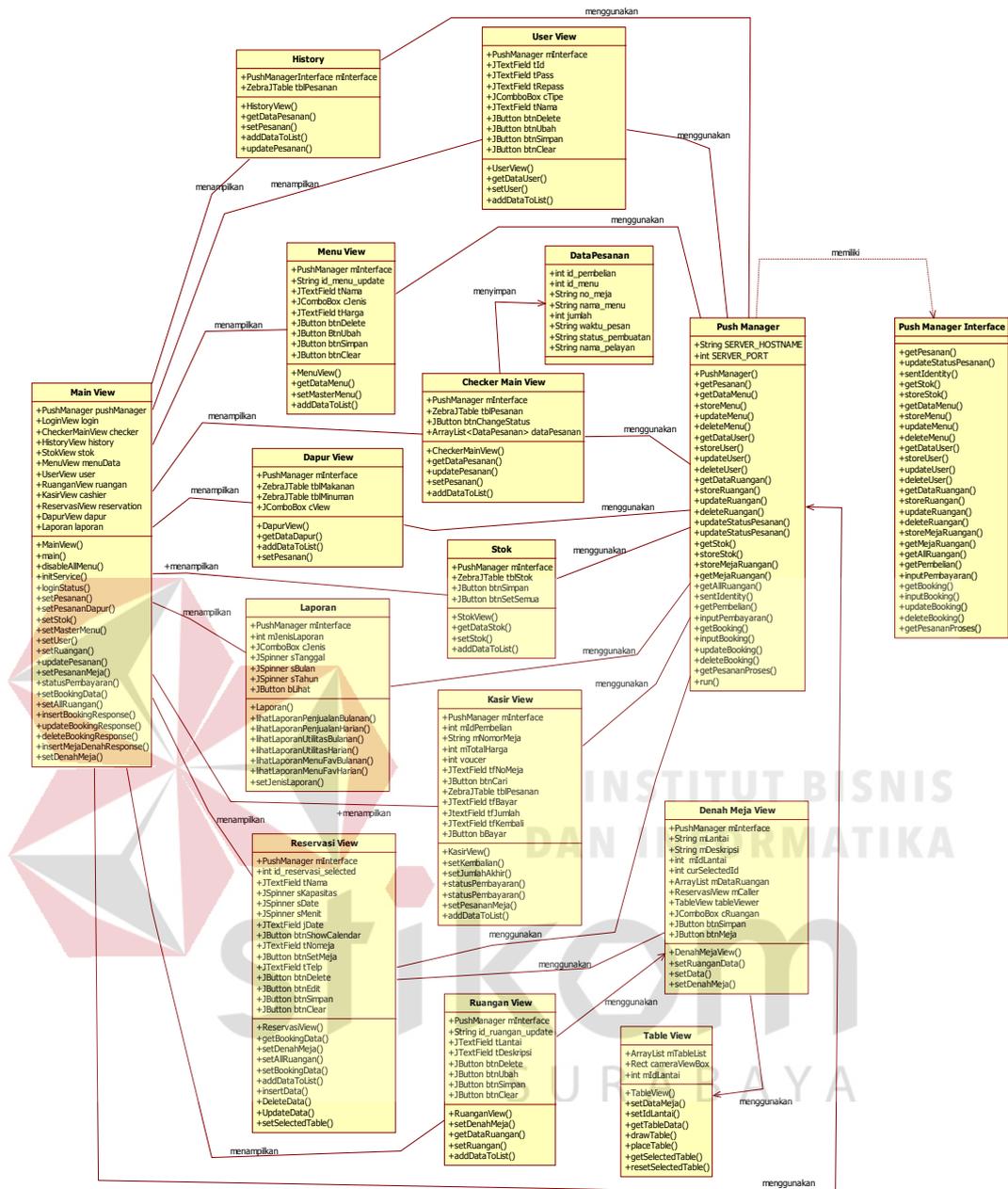
Menunjukkan interaksi antar kelas dalam sistem. *Class* mengandung informasi dan tingkah laku (*behavior*) yang berkaitan dengan informasi tersebut. Sebuah kelas pada diagram kelas dibuat untuk setiap tipe obyek pada diagram sequensial atau diagram kolaborasi.

Class diagram dibagi menjadi 3 (tiga) bagian berdasarkan program yang dibangun, yaitu *client* pada aplikasi *desktop*, *client* pada aplikasi *mobile* android dan *server*. *Class diagram* tersebut digambarkan seperti pada gambar 3.17, 3.18 dan 3.19.

a. *Client Desktop*

Fungsi dari masing-masing *class* yang ada pada *class client desktop* adalah:

- 1 MainView: *main class* yang bertugas menampilkan tampilan utama dan memajemen tampilan lain dalam system.
- 2 *History* : memajemen tampilan dan proses yang terkait riwayat pesanan.
- 3 MenuView: memajemen tampilan dan proses yang terkait master menu.
- 4 DapurView: memajemen tampilan dan proses yang terkait tampilan dapur. Fungsinya adalah menampilkan menu makanan/minuman yang akan diproses oleh *chef* dan *bartender*
- 5 Laporan: memajemen tampilan dan proses yang terkait laporan.
- 6 ReservasiView: memajemen tampilan dan proses terkait *reservasi* tempat.
- 7 UserView: memajemen tampilan dan proses yang terkait manajemen user. Fungsinya adalah menambah, mengubah, menghapus dan menampilkan data user yang tercatat di sistem.
- 8 CheckerMainview: memajemen tampilan dan proses yang terkait manajemen pembuatan menu pesanan.
- 9 Stok: memajemen tampilan dan proses yang terkait manajemen stok
- 10 KasirView: memajemen tampilan dan proses yang terkait kasir.
- 11 RuanganView: memajemen tampilan dan proses yang terkait master ruangan.
- 12 DenahMejaView: Kelas RuanganView digunakan untuk memajemen tampilan dan proses yang terkait denah dari sebuah ruangan.
- 13 TableView: Kelas TableView adalah sebuah kelas yang melakukan penampilan meja pada denah meja.
- 14 PushManager: Kelas PushManager adalah sebuah kelas yang bertugas memajemen komunikasi antara *client* dan *server*.



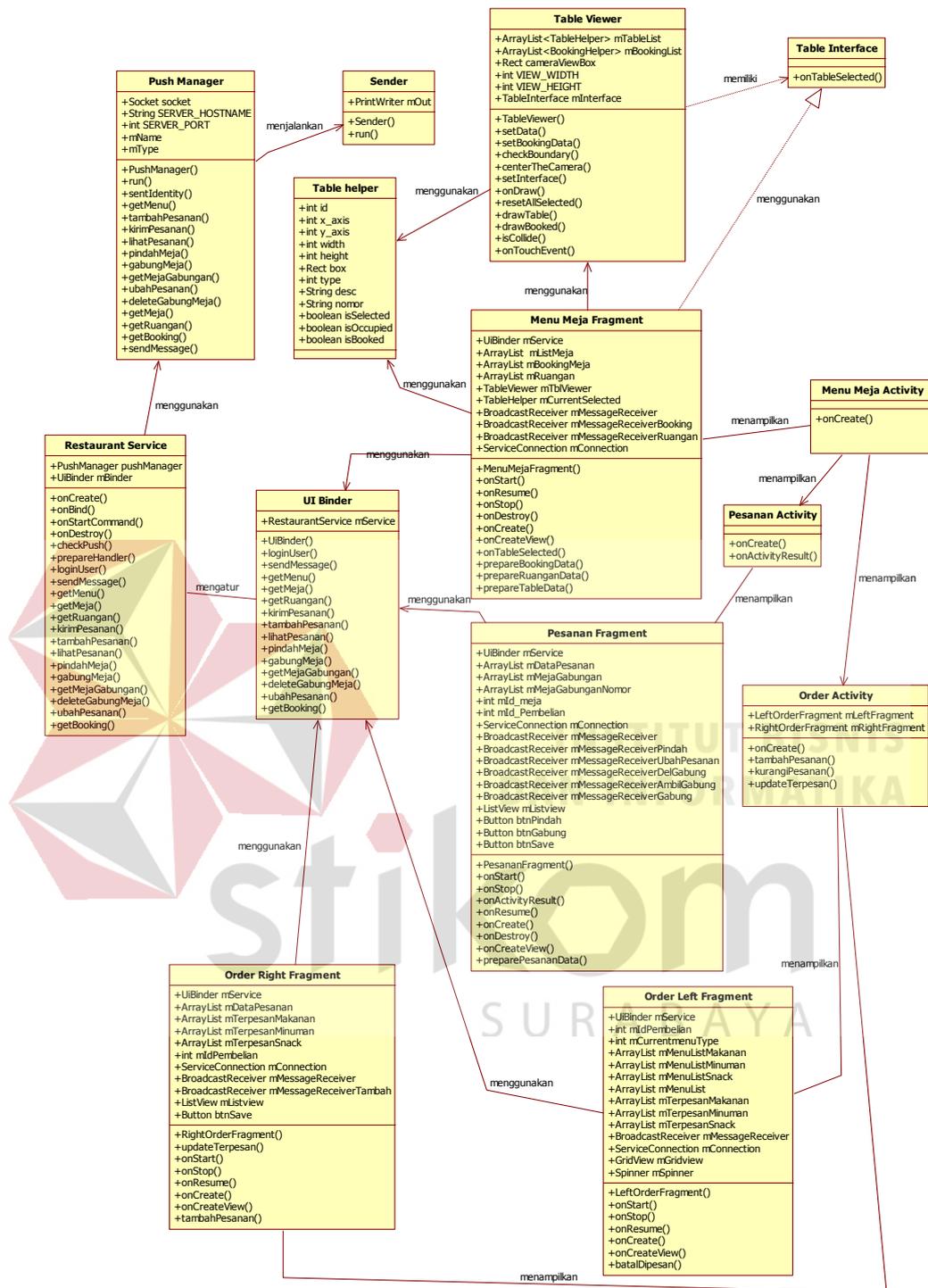
Gambar 3.17 Class Diagram Client Desktop

b. Client Android

Fungsi dari masing-masing class yang ada pada class client android adalah:

- 1 Restaurant.Service: mengatur seluruh proses yang terjadi di *background* (proses yang tidak tampak oleh pengguna karena tidak memiliki *user interface*).

- 2 UIBinder: penghubung antara User Interface (dalam android biasanya adalah Activity atau Fragment dengan Service).
- 3 Pushmanager: manajemen komunikasi antara *client* dan *server*.
- 4 TableViewer: penampilan meja pada denah meja, menggambar meja sesuai dengan jenis, ukuran dan lokasi meja sesuai data pada sistem.
- 5 MenuMejaActivity : turunan dari kelas Activity bawaan android. Kelas untuk menampilkan user interface yang terkait menampilkan data meja.
- 6 MenuMejaFragment: turunan dari kelas Fragment bawaan android. Fragment merepresentasikan perilaku/ sebagian *user interface* dalam sebuah Activity.
- 7 OrderActivity: turunan dari kelas Activity bawaan android. digunakan untuk menampilkan user interface yang terkait pemesanan menu makanan.
- 8 OrderLeftFragment: turunan dari kelas Fragment bawaan android. digunakan untuk menampilkan *user interface* dan proses yang terkait menampilkan *list* dari menu yang tersedia di sistem.
- 9 OrderRightFragment: turunan dari kelas Fragment bawaan android. Untuk menampilkan *user interface* dan proses yang terkait menampilkan *list* menu yang akan dipesan oleh pelanggan.
- 10 PesananActivity: turunan dari kelas Activity bawaan android. Menampilkan user interface terkait manajemen menu yang telah dipesan pelanggan.
- 11 PesananFragment: turunan dari kelas Fragment bawaan android. menampilkan *user interface* dan proses yang terkait menampilkan *list* menu telah dipesan oleh pelanggan.

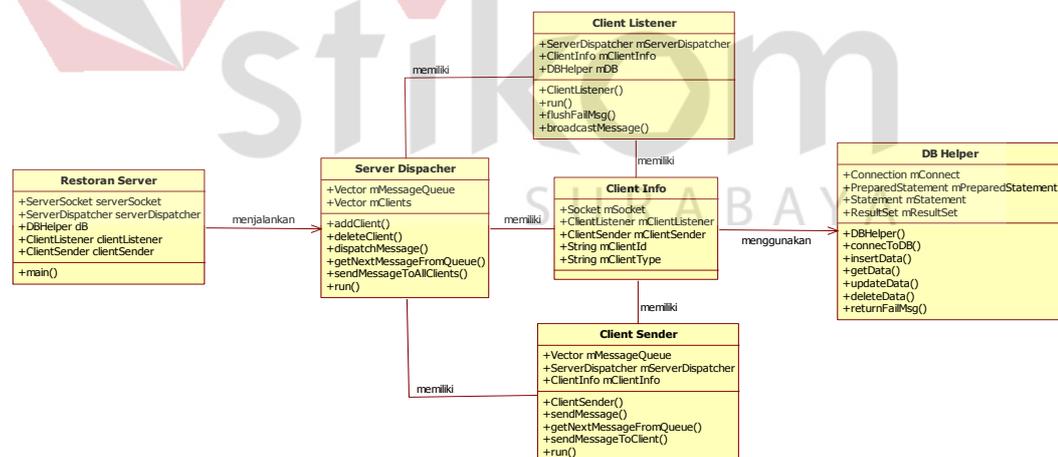


Gambar 3.18 Class Diagram Client Android

c. Server

Fungsi dari masing-masing class yang ada pada class server adalah:

- 1 RestaurantServer: *main class* yang bertugas menginisialisasi proses pada *server* dan menerima koneksi dari *client*.
- 2 ServerDispatcher: untuk melakukan *broadcast* pesan . Fungsi utamanya mengambil pesan dari *message queue* (antrian pesan) dan mengirimkannya ke seluruh *client* yang terkoneksi dengan *server*, menggunakan TCP/IP Socket
- 3 ClientListener: menerima *request* dari *client*. Fungsi utamanya adalah menerima pesan/*request* dari *client*,
- 4 ClientInfo: menyimpan data *client* yang sedang terkoneksi dengan *server* sehingga dengan mudah dapat diakses oleh objek lain dalam sistem.
- 5 ClientSender: untuk mengirim respon ke *client*. Fungsi utamanya adalah mengirim pesan yang diberikan oleh *ServerDispatcher* kepada *client*.
- 6 DBHelper: untuk manajemen koneksi dengan database dan menerima permintaan terkait database (*read, update, delete*) dari objek lain dalam sistem.

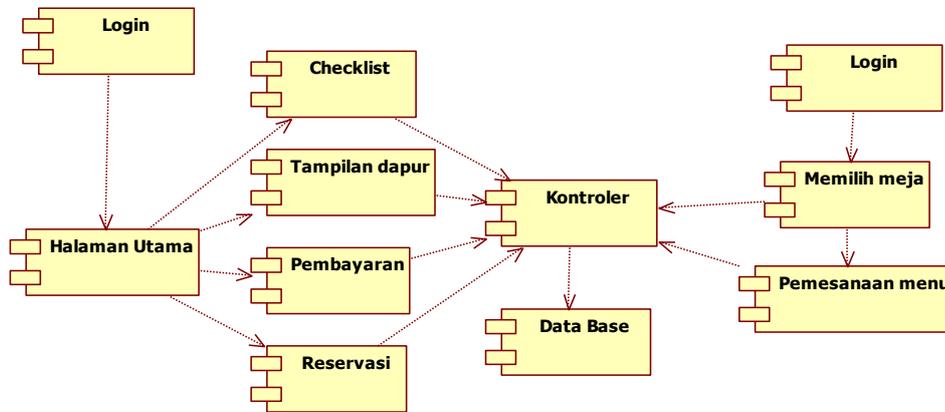


Gambar 3.19 Class Diagram Server

3.2.5 Component Diagram

Component diagram atau diagram komponen adalah diagram UML yang menampilkan komponen dalam sistem dan hubungan antar mereka. Terdapat 2

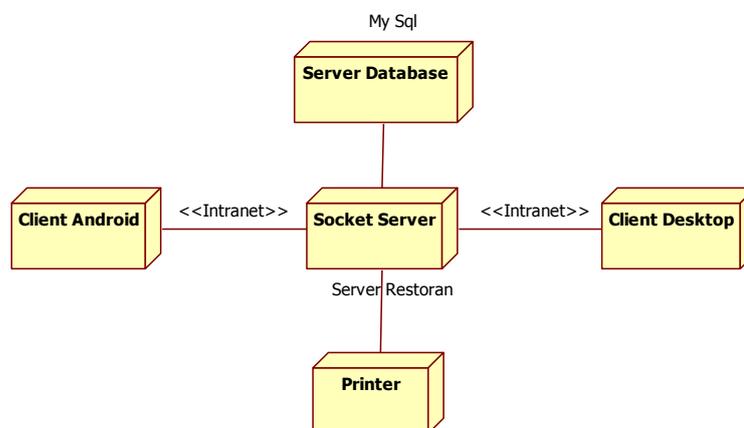
bagian komponen, yaitu komponen dalam aplikasi mobile dan komponen pada aplikasi *desktop* yang saling terhubung pada kontroler seperti pada gambar 3.20.



Gambar 3.20. *Component Diagram* Aplikasi pelayanan

3.2.6 *Deployment Diagram*

Deployment diagram menunjukkan pandangan secara fisik dari suatu sistem dan menunjukkan bagaimana sistem diimplementasikan di perangkat nyata. *Deployment diagram* menampilkan semua node dalam suatu jaringan dan hubungan di antara mereka. *Deployment diagram* pada Rancang bangun aplikasi pelayanan ini dapat dilihat pada gambar 3.21



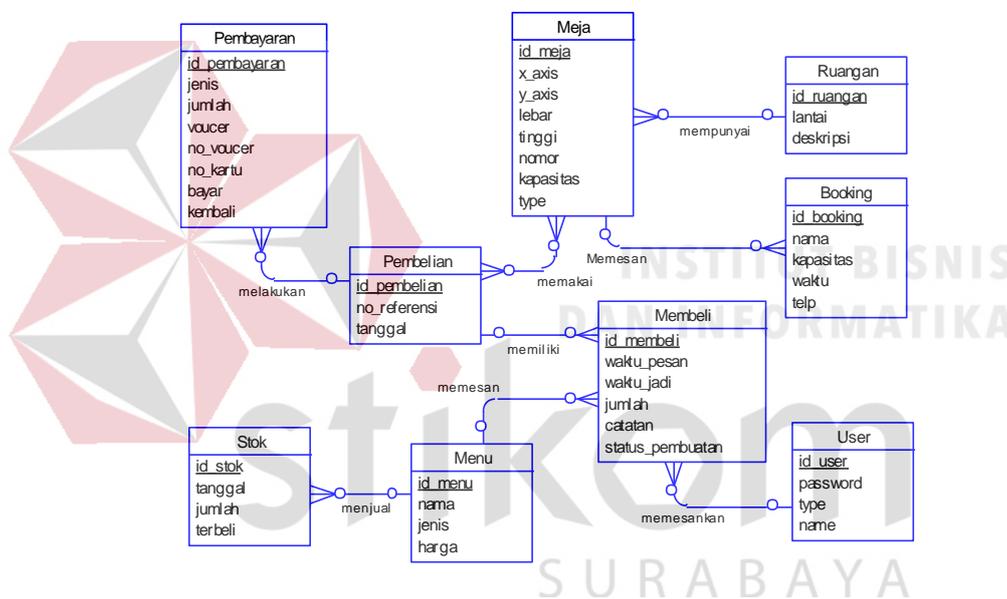
Gambar 3.21. *Deployment Diagram* Aplikasi pelayanan

3.2.7 Entity Relational Diagram (ERD)

Pada *Entity Relational Diagram* (ERD) menjelaskan mengenai relasi antar tabel yang terdapat pada Rancang Bangun Aplikasi Pelayanan pada Restoran Berbasis Android ini diantaranya adalah:

a. Conceptual Data Model (CDM)

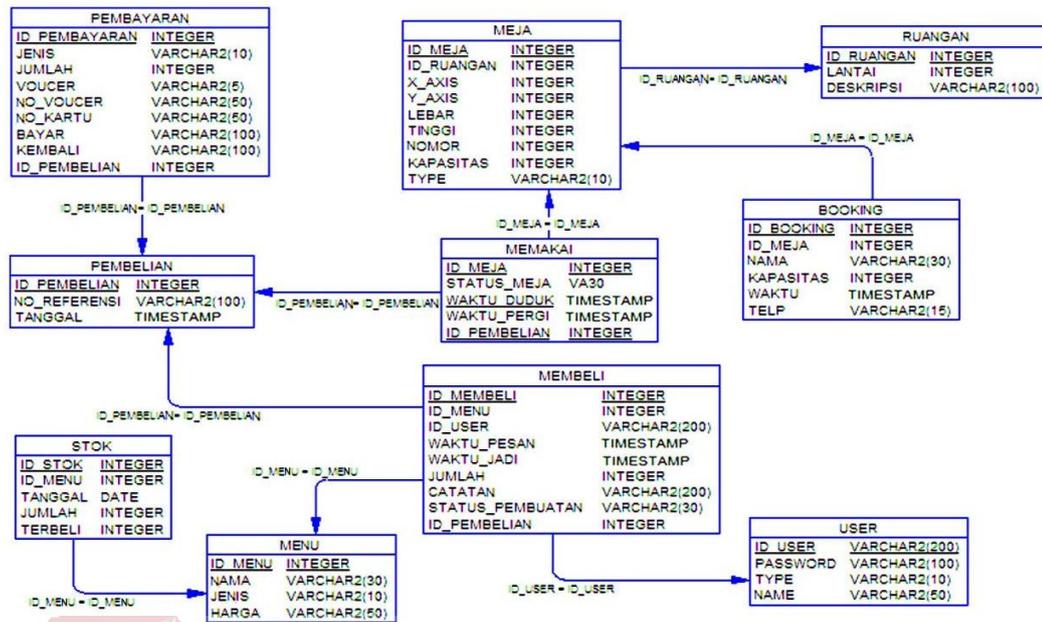
Menggambarakan secara keseluruhan struktur basis data yang dirancang untuk aplikasi. Berikut adalah gambar dari CDM Rancang Bangun Aplikasi Pelayanan pada Restoran



Gambar 3.22 Conceptual Data Model

b. Physical Data Model (PDM)

Menggambarakan secara detail struktur basis data yang dirancang untuk aplikasi. PDM merupakan *generate* dari CDM Berikut adalah gambar dari PDM Rancang Bangun Aplikasi Pelayanan pada Restoran.



Gambar 3.23 Physical Data Model

3.3 Struktur Tabel

Struktur *database* menggambarkan *entity* yang terdapat dalam *database* yang digunakan pada Rancang Bangun Aplikasi Pelayanan Pada Restoran Berbasis Mobile Android. Adapun Struktur *database* tersebut adalah:

1. Tabel Master *User*

Primary key : id_user

Fungsi : Menyimpan data pengguna/ *user*

Tabel 3.19 Struktur Tabel Master *User*

| No | Nama Field | Tipe Data | Size | Keterangan |
|----|------------|-----------|------|--------------------------|
| 1. | id_user | varchar | 200 | Id pengguna |
| 2. | Password | Varchar | 100 | <i>Password</i> |
| 3. | Type | Varchar | 10 | Bagian pekerjaan |
| 4. | Name | Varchar | 50 | Nama lengkap <i>user</i> |

2. Tabel master Ruangan

Primary key : id_ruangan

Fungsi : Menyimpan data ruangan restoran

Tabel 3.20 Struktur Tabel Master Ruang

| No | Nama Field | Tipe Data | Size | Keterangan |
|----|------------|-----------|------|--------------------|
| 1. | Id_ruangan | Integer | 11 | Id ruangan |
| 2. | Lantai | Integer | 11 | Lantai ruangan |
| 3. | Deskripsi | Varchar | 100 | Keterangan ruangan |

3. Tabel master Meja

Primary key : id_meja

Foreign key : id_ruangan

Fungsi : menyimpan data keseluruhan meja

Tabel 3.21 Struktur Tabel Master Meja

| No | Nama Field | Tipe Data | Size | Keterangan |
|----|------------|-----------|------|--|
| 1. | id_meja | Integer | 11 | Id meja |
| 2. | id_ruangan | Integer | 11 | Id ruangan |
| 3. | x_axis | Integer | 11 | Posisi meja pada sumbu X (koordinat kartesius) |
| 4. | y_axis | Integer | 11 | Posisi meja pada sumbu Y (koordinat kartesius) |
| 5. | Lebar | Integer | 11 | Lebar meja pada denah |
| 6. | Tinggi | Integer | 11 | Panjang meja pada denah |
| 7. | Nomor | Varchar | 10 | Nomor meja |
| 8. | Kapasitas | Integer | 11 | Kapasitas meja |
| 9. | Type | Varchar | 10 | Jenis meja |

4. Tabel master Menu

Primary key : id_menu

Fungsi : Meyimpan data menu

Tabel 3.22 Struktur Tabel Master Menu

| No | Nama Field | Tipe Data | Size | Keterangan |
|----|------------|-----------|------|------------|
| 1. | id_menu | Integer | 11 | Id menu |
| 2. | nama | Varchar | 30 | Nama menu |
| 3. | jenis | Varchar | 10 | Jenis menu |
| 4. | harga | Integer | 11 | Harga menu |

5. Tabel Stok

Primary key : id_stok

Foreign key : id_menu

Fungsi : Menyimpan stok menu yang tersedia untuk setiap harinya.

Tabel 3.23 Struktur Tabel Stok

| No | Nama Field | Tipe Data | Size | Keterangan |
|----|------------|-----------|------|--------------------------------|
| 1. | id_stok | Integer | 11 | Id stok |
| 2. | id_menu | Integer | 11 | Id menu |
| 3. | tanggal | date | - | Tanggal stok menu |
| 4. | jumlah | Integer | 11 | Jumlah stok yang tersedia |
| 5. | terbeli | Integer | 11 | Jumlah stok yang telah terbeli |

6. Tabel Memakai

Primary key : id_pembelian, id_meja, waktu_duduk

Foreign key : id_pembelian, id_meja, waktu_duduk

Fungsi : Menyimpan data meja yang digunakan konsumen

Tabel 3.24 Struktur Tabel Memakai

| No | Nama Field | Tipe Data | Size | Keterangan |
|----|--------------|-----------|------|-----------------------------------|
| 1. | id_pembelian | Integer | 11 | Id pembelian |
| 2. | id_meja | Integer | 11 | Id meja |
| 3. | status meja | varchar | 30 | Status meja |
| 4. | waktu_duduk | Timestamp | - | Waktu saat meja digunakan |
| 5. | waktu_pergi | Timestamp | - | Waktu saat meja selesai digunakan |

7. Tabel Membeli

Primary key : id_membeli,

Foreign key : id_menu, id_user, id_pembelian

Fungsi : Menyimpan data pesanan

Tabel 3.25 Struktur Tabel Membeli

| No | Nama Field | Tipe Data | Size | Keterangan |
|----|------------|-----------|------|------------|
| 1 | id_membeli | Integer | 11 | Id membeli |

| | | | | |
|----|------------------|-----------|-----|--|
| 2. | id_pembelian | Integer | 11 | Id pembelian |
| 3. | id_menu | Integer | 11 | Id menu |
| 4. | waktu_pesan | Timestamp | - | Waktu saat pesanan disimpan |
| 5. | waktu_jadi | Timestamp | - | Waktu saat pesanan sudah selesai dibuatkan |
| 6. | jumlah | Integer | 11 | Jumlah pesanan per menu |
| 7. | catatan | Varchar | 200 | Keterangan menu khusus |
| 8. | status_pembuatan | Varchar | 30 | Status menu pesanan |
| 9. | id_user | Varchar | 200 | Id <i>user</i> pelayan yang melayani |

8. Tabel Pembelian

Primary key : id_pembelian

Fungsi : Menyimpan data pembelian

Tabel 3.26 Struktur Tabel Pembelian

| No | Nama Field | Type Data | Size | Keterangan |
|----|--------------|-----------|------|-------------------|
| 1. | id_pembelian | Integer | 11 | Id pembelian |
| 2. | no_reverensi | Varchar | 100 | Nomor reverensi |
| 3. | tanggal | timestamp | - | Tanggal pembelian |

9. Tabel Pembayaran

Primary key : id_pembayaran

Foreign key : id_pembelian

Fungsi : Menyimpan data pembayaran

Tabel 3.27 Struktur Tabel Pembayaran

| No | Nama Field | Type Data | Size | Keterangan |
|----|---------------|-----------|------|--------------------------|
| 1. | id_pembayaran | Integer | 11 | Id pembayaran |
| 2. | jenis | Varchar | 50 | Jenis cara pembayaran |
| 3. | jumlah | Integer | 11 | Biaya pembayaran |
| 4. | id_pembelian | Integer | 11 | Id pembelian |
| 5. | voucer | Varchar | 5 | Nominal voucer |
| 6. | no_voucer | Varchar | 50 | Nomor voucer |
| 7. | no_kartu | Varchar | 50 | Nomor kartu |
| 8. | bayar | Varchar | 100 | Uang yang diterima kasir |
| 9. | kembali | Varchar | 100 | Uang kembalian |

10. Tabel *Booking*

Primary key : id_booking, waktu

Foreign key : id_meja

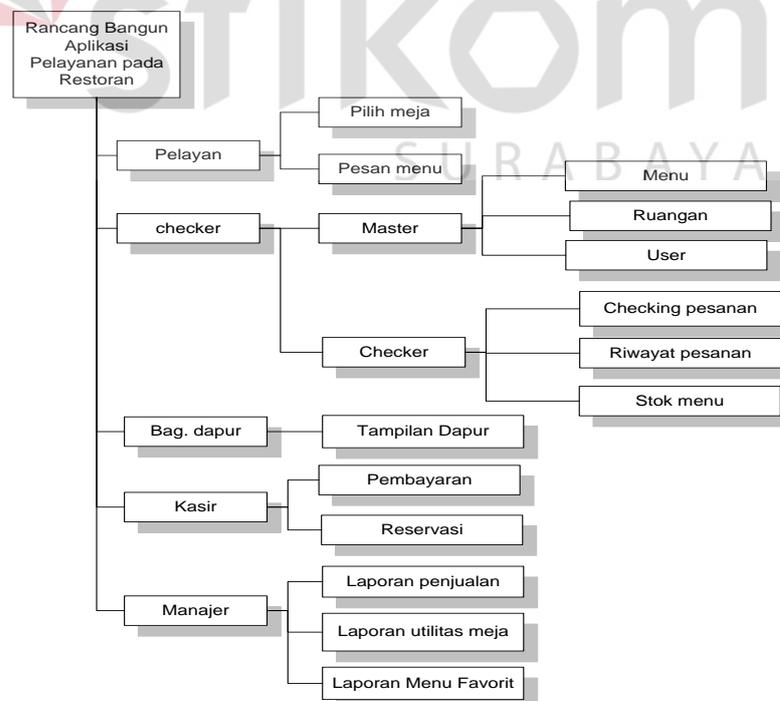
Fungsi : Menyimpan data *booking* meja

Tabel 3.28 Struktur Tabel *Booking*

| No | Nama <i>Field</i> | Tipe Data | Size | Keterangan |
|----|-------------------|-----------|------|----------------------------------|
| 1. | id_booking | Integer | 11 | Id <i>booking</i> meja |
| 2. | id_meja | Integer | 11 | Id meja |
| 3. | nama | Varchar | 30 | Nama pemesan (<i>customer</i>) |
| 4. | kapasitas | Integer | 11 | Jumlah kursi yang dipesan |
| 5. | Waktu | Timestamp | - | Waktu pemesanan |
| 6. | telp | Varchar | 15 | No tlp pemesan |

3.4 Desain *Interface*

Desain *interface* digunakan sebagai panduan dalam membuat aplikasi. Sehingga memudahkan dalam memperkirakan tampilan serta kebutuhan alur aplikasi. Desain *interface* dirancang berupa *sitemap* seperti gambar 3.24.



Gambar 3.24 *Sitemap* Desain *Interface* Aplikasi

Sitemap tersebut terdapat 5 *user*. Masing-masing *user* memiliki akses penggunaan aplikasi, antara lain:

a. Pelayan:

1. Pilih meja: Pelayan melakukan proses pemilihan meja, baik mencari meja yang kosong, membuat status meja menjadi isi (blok warna merah) maupun mengecek meja yang dipesan/*booking*
2. Pesan Menu: Pelayan melakukan pemesanan menu, mulai dari melihat menu, memasukkan daftar pesanan, mengecek status menu hingga melakukan perubahan menu

b. *Checker* terdiri dari master dan checking

1. Form master berisikan: Master menu, master ruangan dan master *user*
2. Form master *checker* berisikan:
 - *Checking* pesanan: *Checker* mengontrol pesanan menu dengan merubah status “proses” maupun “selesai”
 - Riwayat pesanan: *Checker* dapat melihat histori menu dari hasil *checking* pesanan.
 - Stok Menu: *Checker* memasukan jumlah stok menu tersedia setiap harinya

c. Bagian Dapur: Petugas dapur hanya dapat melihat menu-menu yang harus dibuatkan, tampilan menu bisa disesuaikan berdasarkan kebutuhan, seperti tampil makanan dan minuman menjadi satu, tampil makanan dan minuman secara terpisah, tampil makanan sendiri dan tampil minuman sendiri.

d. Kasir

1. Pembayaran: Kasir memproses pembayaran berdasar nomor meja yang dipilih

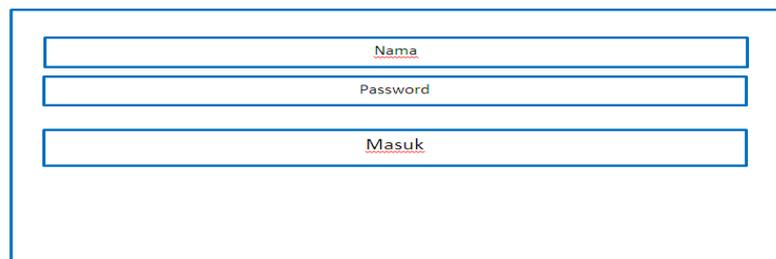
2. *Reservasi*: Kasir memproses pemesanan meja/*booking* meja dengan memasukkan data yang dibutuhkan.
- e. Manajer:
1. Laporan Penjualan: Laporan didapat dari hasil penjualan.
 2. Laporan utilitas meja: Merupakan laporan lamanya pemakaian meja, dihitung dari jam buka operasional restoran hingga tutup restoran.
 3. Laporan menu favorit: Dihitung dari jumlah pesanan selama jam operasional restoran. Sehingga didapatkan menu yang banyak dipesan oleh *customer*.

Setelah didapatkan sitemap, tahap berikutnya adalah melakukan desain tampilan aplikasi atau desain *interface*. Desain *interface* digunakan sebagai acuan dalam proses pembuatan aplikasi. Desain *interface* dibagi menjadi beberapa

3.4.1 Desain *Interface Mobile application*

a. *Login*

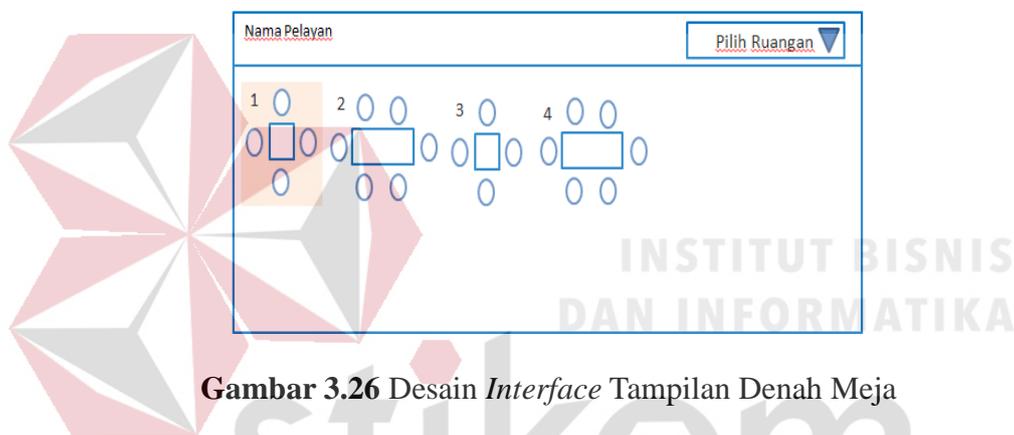
Pelayan bertugas mengakses *mobile application*. Pelayan memasukkan “Nama” dan “Password” sesuai dengan data yang tersimpan pada Master *User*. Kemudian tekan tombol “Masuk”. Jika benar maka tampilan akan masuk ke form “Denah meja”. Jika salah, pelayan diminta memasukkan “Nama” dan “Password” dengan benar.



Gambar 3.25 Desain *Interface Login Mobile application*

b. Tampilan Denah Meja

Setelah proses *login* sukses, sistem akan menampilkan denah meja. Pelayan dapat memilih tampilan denah meja berdasarkan jenis ruangan yang ada dalam restoran. Meja dengan adanya tanda blok warna merah menunjukkan bahwa meja tersebut telah terisi, blok warna ungu menunjukkan meja tersebut telah dipesan/*booking* dan bila tidak ada blok maka meja tersebut sedang kosong. Pada saat memilih meja, maka sistem akan menandai meja sementara sehingga tidak dapat dipilih oleh calon customer lain.



Gambar 3.26 Desain *Interface* Tampilan Denah Meja

c. Pemesanan menu

Saat *customer* sudah menempati meja yang diinginkan, pelayan menekan gambar meja. Maka akan tampil pilihan menu pada kolom sebelah kiri dan daftar yang telah dipesan pada kolom sebelah kanan.

Pada kolom kiri, terdapat pilihan: makanan, minuman dan *snack*. Jika pelayan memilih tampilan makanan, maka akan muncul semua menu makanan beserta harga. Menu yang pada hari itu sedang kosong ditandai dengan tulisan menu berwarna merah sehingga tidak dapat dilakukan pemesanan pada menu tersebut.

Pada kolom sebelah kanan adalah daftar dari menu yang dipesan oleh *customer*. Daftar tersebut berisikan nama menu, jumlah yang dipesan, total harga per menu dan total harga secara keseluruhan (belum termasuk PPN 10%). Jika daftar pesanan sudah sesuai, maka tekan tombol “Simpan”. Daftar pesanan tersebut tersimpan pada *database server* dan otomatis muncul pada form “Checking Pesanan” yang ada pada layar *checker* dengan status “Menunggu” yang artinya masih dalam tahap menunggu giliran untuk dibuatkan.

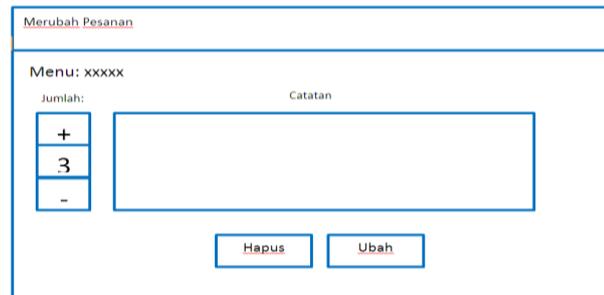
| Meja: 4-1 | | Makanan | Menu pesanan | Simpan |
|------------------------|----------------------|--------------------------------|----------------------|----------------------|
| Rp. 000 | Rp. 000 | Rp. 000 | Rp. 000 | Rp. 000 |
| <input type="text"/> | <input type="text"/> | <input type="text"/> | <input type="text"/> | <input type="text"/> |
| Makanan | Makanan | Makanan | Makanan | Makanan |
| Rp. 000 | Rp. 000 | Rp. 000 | Rp. 000 | Rp. 000 |
| <input type="text"/> | <input type="text"/> | <input type="text"/> | <input type="text"/> | <input type="text"/> |
| Makanan | Makanan (Habis) | Makanan | Makanan | Makanan |
| Total pesanan: Rp. 000 | | | | |
| xxxxxxxx | Rp. 000 | <input type="text" value="3"/> | | |
| xxxxxxxx | Rp. 000 | <input type="text" value="3"/> | | |
| xxxxxxxx | Rp. 000 | <input type="text" value="3"/> | | |

Gambar 3.27 Desain *Interface* Pemesanan Menu

d. Merubah Pesanan dan Pesan Menu Spesial

Pelayan dapat merubah pesanan apa bila *customer* ingin merubah pesanan yang sebelumnya sudah dipesan. Pelayan dapat menambah atau mengurangi pesanan saat pesanan dalam status “Menunggu”. Bila menu pesanan dengan status “proses”, maka tidak dapat dilakukan pengurangan menu, namun dapat dilakukan penambahan.

Kolom catatan merupakan kolom dari pemesanan menu spesial. Menu spesial adalah menu yang dengan catatan khusus sesuai permintaan *customer*. Catatan dari menu spesial ini akan muncul di samping menu yang dipilih ketika tampil pada form *checking* menu.



Gambar 3.28 Desain *Interface* Merubah Pesanan dan Pesan Menu Spesial

e. Status Pesanan, Gabung Meja dan Pindah meja

Pelayan dapat mengecek status pesanan *customer* berdasarkan nomor meja yang dipilih. Status pesanan tampil pada samping menu pesanan. Status pesanan “Menunggu” artinya menu pesanan masih belum dibuatkan/ masih antri untuk dibuatkan. Status pesanan “Proses” artinya menu pesanan tersebut dalam tahap pembuatan. Status pesanan “Selesai” artinya menu pesanan tersebut sudah selesai dibuatkan.

Bila *customer* ingin menggabungkan meja, maka pelayan akan menekan meja yang awal digunakan, kemudian menekan tombol “Pindah Meja”. Sistem akan menampilkan denah meja kembali, pelayan memilih nomor meja yang ingin ditambahkan. Maka meja yang ditambahkan tersebut akan muncul tanda blok merah dan pelayan dapat melihat meja berapa saja yang digabungkan.

Bila *customer* ingin pindah meja. Pelayan memilih meja yang ditempati oleh *customer* sebelumnya, pilih tombol “Pindah Meja”, kemudian sistem menampilkan denah meja. Pelayan memilih meja baru yang akan ditempati oleh *customer*. Otomatis *blocking* meja juga akan ikut berpindah.

| | | |
|-----------------------------|--------------------------------|-------------|
| Meja:2-1 | Pindah meja | Gabung Meja |
| Total pesanan: Rp.000 | | |
| Xxxxxxxx (Selesai) Rp. 000 | <input type="text" value="3"/> | |
| Xxxxxxxx (proses) Rp. 000 | <input type="text" value="3"/> | |
| Xxxxxxxx (Menunggu) Rp. 000 | <input type="text" value="3"/> | |

Gambar 3.29 Desain *Interface* Status Pesanan, Gabung Meja dan Pindah meja

3.4.2 Desain *Interface Desktop application*

a. *Login*

Hal utama yang dilakukan agar dapat masuk ke aplikasi pelayanan *desktop* adalah melakukan proses *login*. Nama *user* dan *Password* yang dimasukkan berdasarkan *User Id* yang tersimpan pada *database Master User*. *User* yang dapat mengakses *desktop application* adalah *checker*, bagian Dapur, kasir dan manajer. Jika data *login* sudah diisi maka tekan tombol "*login*". Bila data *login* benar maka *user* dapat masuk pada menu utama. Jika data *login* salah maka *user* diminta kembali memasukkan data *login* dengan benar.

| | |
|-----------|---------------------------------------|
| Login | |
| Nama User | <input type="text" value="Nama"/> |
| Password | <input type="text" value="Password"/> |
| | <input type="button" value="Login"/> |

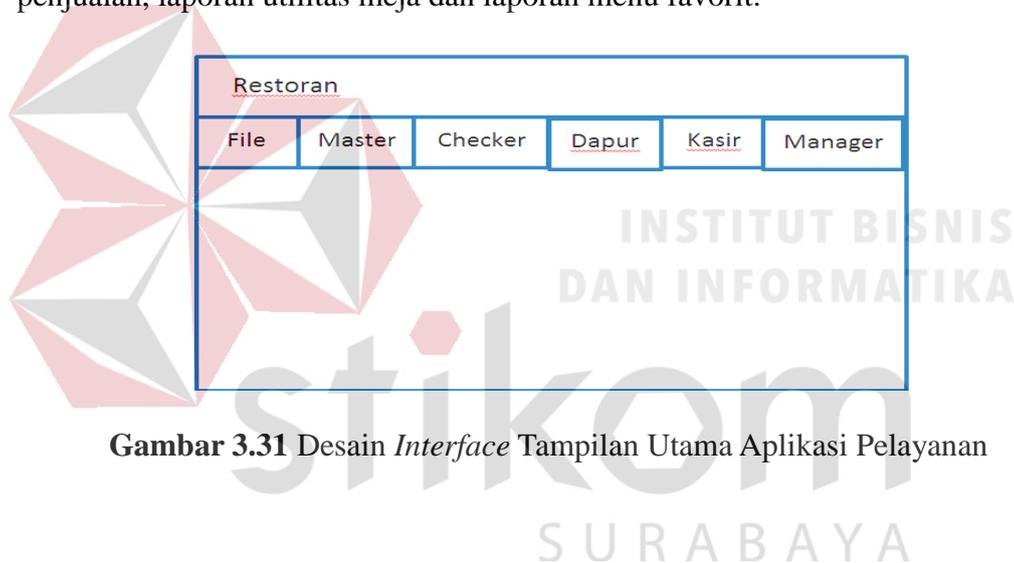
Gambar 3.30 Desain *Interface Login Desktop application*

b. *Tampilan Utama Aplikasi Pelayanan*

Setelah proses *login* sukses, Maka sistem akan menampilkan form menu utama sesuai dengan hak akses masing-masing. *User checker* mempunyai

hak akses form master dan checker. *User* petugas dapur mempunyai hak akses form dapur. *User* petugas kasir mempunyai hak akses form kasir. *User* Manajer mempunyai hak akses form manajer.

Kolom “file” berisikan form “*login*” maupun “*logout*”. Kolom Maser berisikan data master menu, ruangan dan *user*, kolom *checker* berisikan form *checking* pesanan, Riwayat pesanan dan stok menu. Kolom dapur berisikan tampilan menu yang dibuat. Kolom kasir berisikan form pembayaran dan form *reservasi* (pemesanan meja), serta kolom manajer berisikan form laporan penjualan, laporan utilitas meja dan laporan menu favorit.



Gambar 3.31 Desain *Interface* Tampilan Utama Aplikasi Pelayanan

c. Master Menu

Master menu merupakan form untuk mengisi data-data menu seperti: nama menu, jenis menu dan harga. Bila data menu sudah dimasukkan maka tekan tombol “simpan” dan sistem akan otomatis menampilkan data menu. Tombol “hapus” digunakan untuk menghapus data yang sudah tersimpan. Tombol “ubah” digunakan untuk merubah data menu yang sebelumnya sudah tersimpan. Tombol “bersihkan form” digunakan untuk membersihkan kolom data yang tidak jadi disimpan. Data menu yang tersimpan, akan tersimpan pada *database*.

| Master Menu | | | |
|--|----------------------|-------|-------|
| Nama | <input type="text"/> | | |
| Jenis | Makanan ▼ | | |
| Harga | <input type="text"/> | | |
| <input type="button" value="Hapus"/> <input type="button" value="Ubah"/> <input type="button" value="Simpan"/> <input type="button" value="Bersihkan Form"/> | | | |
| Id menu | Nama | Jenis | Harga |
| | | | |
| | | | |
| | | | |

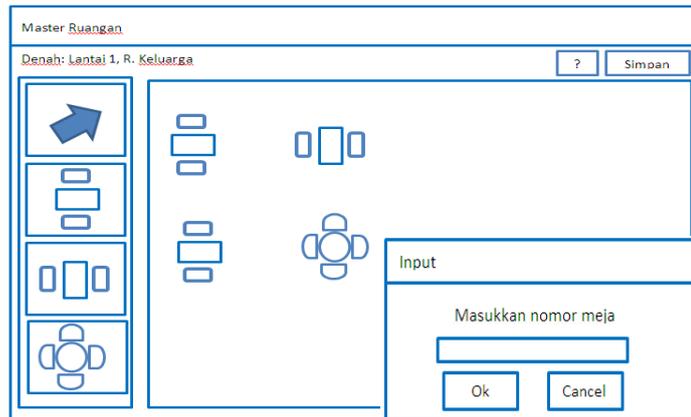
Gambar 3.32 Desain *Interface* Master Menu

d. Master Ruangan

Form Master ruangan merupakan form yang digunakan untuk mengisi data-data ruangan yang ada pada restoran. Kolom “lantai” diisi nomor lantai. Kolom “deskripsi” diisi mengenai jenis ruangan yang ada. Bila data ruangan sudah dimasukkan maka tekan tombol “simpan” seperti pada gambar 3.33. Kemudian mengisi denah meja yang terdapat pada kolom denah meja dengan menekan “aksi (klik). Sistem akan menampilkan form untuk membuat denah meja seperti pada gambar dan mengisi nomor meja 3.34. Bila denah meja sudah selesai dibuat tekan tombol “simpan”.

| Master Ruangan | | | |
|--|----------------------|-----------|------------|
| Lantai | <input type="text"/> | | |
| Deskripsi | <input type="text"/> | | |
| <input type="button" value="Hapus"/> <input type="button" value="Ubah"/> <input type="button" value="Simpan"/> <input type="button" value="Bersihkan Form"/> | | | |
| Id Ruangan | Lantai | Deskripsi | Denah meja |
| | | | |
| | | | |
| | | | |

Gambar 3.33 Desain *Interface* Master Ruangan



Gambar 3.34 Desain *Interface Setting Denah Meja*

e. Master User

Master *User* digunakan untuk mengisi data pengguna. “*Username*” berisikan nama *user*. “*Password*” berisikan *password* dari *user*. “*Re-password*” merupakan pengisian ulang *password*. “*Tipe*” berisikan jenis jabatan *user*, diantaranya: Pelayan, *Checker*, Bagian dapur, Kasir dan Manajer. “*Nama*” berisikan nama lengkap *user*.

Bila data *user* sudah dimasukkan maka tekan tombol “simpan”. Tombol “hapus” digunakan untuk menghapus data yang sudah tersimpan. Tombol “ubah” digunakan untuk merubah data *user* yang sebelumnya sudah tersimpan. Tombol “bersihkan form” digunakan untuk membersihkan kolom data.

| Id user | Tipe | Nama |
|---------|------|------|
| | | |
| | | |
| | | |

Gambar 3.35 Desain *Interface Master User*

f. Checking Pesanan

Form *checking* pesanan digunakan untuk mengontrol pesanan. Tampilan urutan pesanan berdasarkan waktu pesanan. Menu yang baru masuk akan muncul dengan status “menunggu”. Menu yang akan ditampilkan pada layar dapur dirubah menjadi status ”proses”. Menu yang sudah selesai dibuatkan dihilangkan dari *list checking* pesanan. Untuk mengubah status maka tekan tombol “ganti status”. Dapat pula dilakukan multi *select* pada menu yang ingin dirubah statusnya

| Checking Pesanan | | | | | |
|------------------|-------------|-----------|--------|--------|---------|
| Checker: | | | | | |
| Nomor Meja | Waktu Pesan | Nama Menu | Jumlah | Status | Pelayan |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |

Ganti Status

Gambar 3.36 Desain *Interface* Checking Pesanan

g. Riwayat Pesanan

Form riwayat pesanan digunakan untuk melihat menu-menu yang sudah dibuat atau bestatus “selesai” yang artinya pesanan tersebut tidak terdapat pada form *checking* pesanan.

| Riwayat Pesanan | | | | |
|-----------------|---------------|-----------|--------|--------|
| Checker: | | | | |
| No. meja | Waktu pesanan | Nama menu | Jumlah | Status |
| | | | | |
| | | | | |
| | | | | |
| | | | | |

Gambar 3.37 Desain *Interface* Riwayat Pesanan

h. Stok Menu

Form stok menu diisi setiap hari sebelum jam operasional restoran dibuka. Form ini digunakan untuk mengisi jumlah stok makanan yang tersedia pada hari tersebut hanya dilakukan 1 (Satu) kali pengisian. Bila pada hari tersebut ada menu yang kosong, maka kolom jumlah diisi dengan angka 0 (nol). Tombol “Set semua” digunakan untuk mengisi jumlah semua stok menu secara keseluruhan, sehingga tidak perlu mengisi jumlah stok satu persatu.

| Stok Menu | | | | |
|-------------------|-----------|-------|-------------|--|
| Stok tanggal: / / | | | | <input type="button" value="Set semua"/> |
| Id menu | Nama menu | Jenis | Jumlah stok | Terbeli |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | <input type="button" value="Simpan"/> |

Gambar 3.38 Desain *Interface* Stok Menu

i. Tampilan Dapur

Form tampilan dapur digunakan untuk memilih jenis tampilan pada layar dapur. Form ini berisikan menu yang harus dibuat oleh petugas dapur (*chef* dan *bartender*). Jenis tampilannya terdiri dari: tampil makan dan minuman digabung, makanan dan minuman dipisah, makanan saja atau minuman saja.

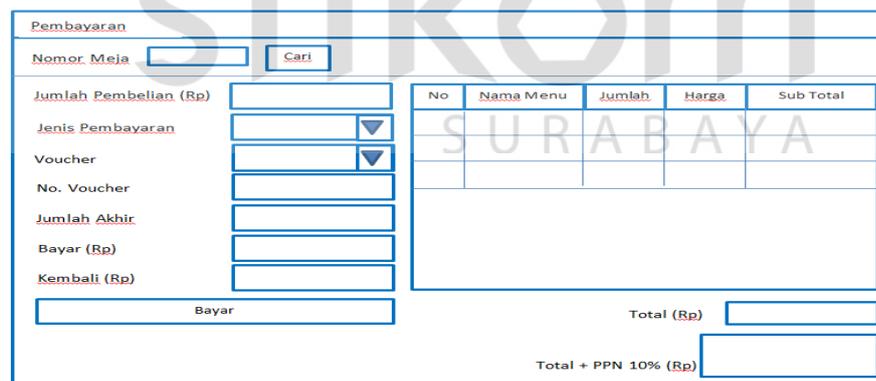
| Dapur | | | | | |
|----------------|--------|---------|---------|--------|----------------------------------|
| Jenis Tampilan | | | | | <input type="button" value="▼"/> |
| Nama ma | Jumlah | Catatan | Nama mi | Jumlah | Catatan |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |

Gambar 3.39 Desain *Interface* Dapur

j. Pembayaran

Form pembayaran digunakan pelayan untuk melakukan proses pembayaran dengan mengisi nomor meja yang sesuai kemudian tekan tombol “cari”, maka data menu yang harus dibayar akan muncul pada kolom sebelah kanan. “Total (Rp)” merupakan total pesanan sebelum ditambah biaya PPN. “Total + PPN 10% (Rp)” merupakan total biaya setelah ditambahkan PPN dan yang harus dibayar oleh *customer*. Data pembayaran diperoleh dari *database*.

Kolom “jumlah pembelian” berisi total pembelian. “Jenis pembayaran” berisi: *cash*, kartu debit, kartu kredit. Bila menggunakan voucher diisi dengan memilih nominal potongan voucher. “Jumlah akhir” berisikan biaya yang harus dibayar setelah ada pemotongan. “Bayar” adalah jumlah uang yang diterima. “Kembali (Rp)” adalah total kembalian dari uang yang diterima. Bila data pembayaran sudah diisi maka tekan tombol “bayar”. Struk pembayaran tercetak. Pembayaran dapat dilakukan bila menu yang dipesan berstatus “selesai”.



The screenshot shows a web-based payment interface titled "Pembayaran". It features a search bar for "Nomor Meja" with a "Cari" button. Below this are several input fields: "Jumlah Pembelian (Rp)", "Jenis Pembayaran" (a dropdown menu), "Voucher" (a dropdown menu), "No. Voucher", "Jumlah Akhir", "Bayar (Rp)", and "Kembali (Rp)". A "Bayar" button is located at the bottom left. On the right side, there is a table with the following columns: "No", "Nama Menu", "Jumlah", "Harga", and "Sub Total". Below the table, there are two more input fields: "Total (Rp)" and "Total + PPN 10% (Rp)".

Gambar 3.40 Desain *Interface* Pembayaran

k. Reservasi

Form *reservasi* digunakan untuk mengisi data pemesanan/*booking* meja yang dilakukan oleh petugas kasir. Kapasitas orang menentukan jumlah kursi

yang dibutuhkan. Waktu pemesanan berisikan jam dan menit dan tanggal. Dapat dilakukan pilih banyak meja sesuai kebutuhan.

Nomor meja muncul otomatis setelah memilih denah meja dengan menekan tombol “set meja”. Kemudian mengisi nomor telepon. Data *reservasi* disimpan pada *database server*. Tanda blok warna “ungu” akan muncul pada denah aplikasi *mobile android* ketika 1 (satu) jam menjelang waktu yang dipesan.

| Id Reservasi | Id Meja | No. Meja | Nama | Kapasitas | Waktu | No Telp. |
|--------------|---------|----------|------|-----------|-------|----------|
| | | | | | | |
| | | | | | | |

Gambar 3.41 Desain *Interface Reservasi*

I. Laporan Penjualan

Form Laporan Penjualan Merukan form untuk menampilkan data penjualan. Kolom jenis laporan untuk menampilkan laporan bulanan atau harian. Kemudian isi tanggal, bulan dan tahun yang diinginkan. Tekan tombol “Lihat laporan”, maka laporan penjualan tampil sesuai dengan yang diinginkan. Form laporan didapat dari hasil penjualan yang tersimpan pada *database*.

Gambar 3.42 Desain *Interface Laporan Penjualan*

m. Laporan Utilitas Meja

Form Laporan utilitas meja merupakan form untuk menampilkan laporan lamanya pemakaian meja yang ada pada restoran. Kolom jenis laporan berisi pilihan untuk menampilkan laporan bulanan atau harian. Kemudian isi tanggal, bulan dan tahun yang diinginkan. Tekan tombol “Lihat laporan”, maka laporan utilitas meja tampil sesuai dengan yang diinginkan.

Gambar 3.43 Desain *Interface* Laporan Utilitas Meja

n. Laporan Menu Favorit

Form laporan Menu Favorit digunakan untuk menampilkan peringkat menu. Menu yang banyak dipesan berada pada urutan pertama. Kolom jenis laporan berisi pilihan untuk menampilkan laporan bulanan atau harian. Kemudian isi tanggal, bulan dan tahun yang diinginkan. Tekan tombol “Lihat laporan”, maka laporan menu favorit tampil sesuai dengan yang diinginkan.

Gambar 3.44 Desain *Interface* Laporan Menu Favori

3.5 Desain Test Case

3.5.1 Aplikasi pada Pelayan

a. Desain Uji Coba Form Login

Uji coba form login pada aplikasi mobile maupun dekstop berfungsi untuk mengetahui fungsi *login* dapat berjalan dengan lancar, seperti tabel 3.29

Tabel 3.29 Desain *Test Case* Form *Login*

| Test Case | Tujuan | Input | Output Diharapkan | Status |
|-----------|--|---|--|--------|
| 1 | Mengetahui respon sistem terhadap data <i>user</i> valid | Nama <i>user</i> dan <i>password</i> pelayan | <i>User</i> masuk kedalam halaman denah meja | |
| 2 | Mengetahui respon sistem terhadap data <i>user</i> tidak valid | Nama <i>user</i> dan <i>password</i> yang salah | <i>User</i> tidak dapat masuk kedalam sistem. Tetap pada halaman login dan muncul pesan “login failed” | |

b. Desain Uji Coba Form Pilih Meja

Uji coba form pilih meja pada aplikasi *mobile* pelayan untuk mengetahui proses pilih meja kosong/isi/*terbooking* berjalan sesuai dengan yang diharapkan.

Tabel 3.30 Desain *Test Case* Form Pilih meja

| Test Case | Tujuan | Input | Output Diharapkan | Status |
|-----------|--|---|---|--------|
| 3 | Mengetahui respon sistem ketika memilih meja yang kosong | Pilih meja tanpa ada blok merah / ungu | masuk pada halaman pilih menu | |
| 4 | Mengetahui respon sistem ketika memilih meja yang terisi | Pilih meja yang terdapat blok warna merah | muncul <i>list</i> menu yang dipesan | |
| 5 | Mengetahui respon sistem ketika memilih meja yang sedang <i>terbooking</i> | Pilih meja yang terdapat blok warna ungu | muncul pesan “meja sudah <i>dibooking</i> oleh Bapak/Ibu ...) | |

c. Desain Uji Coba Form Pemesanan

Uji coba form pemesanan pada aplikasi mobile pelayan berfungsi untuk mengetahui proses pemesanan menu yang tersedia/kosong dan tambah/kurang

Tabel 3.31 Desain Test Case Form Menu

| Test Case | Tujuan | Input | Output Diharapkan | Status |
|-----------|--|--|--|--------|
| 6 | Mengetahui respon sistem ketika memesan menu yang tersedia | pilih menu yang berwarna hijau, | Tampilan kembali ke menu meja | |
| 7 | Mengetahui respon sistem ketika memesan menu yang kosong | klik menu makanan yang berwarna merah | Tidak terjadi apa-apa | |
| 8 | Mengetahui respon sistem ketika menambahkan / mengurangi/ menghapus menu | Klik tombol “+” atau (-) atau “delete” pada form “menambahkan pesanan” | Jumlah menu bertambah/ berkurang/ hilang dari <i>list</i> “menu pesanan” | |

d. Desain Uji Coba Form Pesan Menu Spesial

Uji coba form pesan menu spesial berfungsi untuk mengetahui proses pencatatan menu spesial sudah berfungsi dengan baik.

Tabel 3.32 Desain Test Case Form Pesan Menu Spesial

| Test Case | Tujuan | Input | Output Diharapkan | Status |
|-----------|--|---------------------|---|--------|
| 9 | Mengetahui respon sistem ketika memesan menu spesial | isi kolom “catatan” | Muncul form “menambahkan pesanan” dan menu berisi catatan | |

e. Desain Uji Coba Form Status Pesanan

Uji coba form status pesanan pada aplikasi *mobile* pelayan berfungsi untuk menguji bila mengubah pesanan dengan status menunggu, proses, selesai seperti tabel 3.33.

Tabel 3.33 Desain *Test Case* Form Status Pesanan

| Test Case | Tujuan | Input | Output Diharapkan | Status |
|-----------|---|---------------------------------------|----------------------------|--------|
| 10 | Mengetahui respon sistem ketika mengubah pesanan berstatus “menunggu” | Mengurangi jumlah <i>item</i> menu | Tampilan data menu berubah | |
| 11 | Mengetahui respon sistem ketika mengubah pesanan berstatus “proses” | Tekan jumlah menu berstatus “proses” | Tidak terjadi apa-apa | |
| 12 | Mengetahui respon sistem ketika mengubah pesanan berstatus “selesai” | Tekan jumlah menu berstatus “selesai” | Tidak terjadi apa-apa | |

f. Desain Uji Coba Form Pindah Meja

Uji coba form pindah meja pada aplikasi *mobile* pelayan berfungsi untuk mengecek apa yang terjadi bila pindah meja

Tabel 3.34 Desain *Test Case* Form Pindah Meja

| Test Case | Tujuan | Input | Output Diharapkan | Status |
|-----------|---|--|--|--------|
| 13 | Mengetahui respon sistem ketika pindah ke meja yang kosong | Pindah ke meja yang tidak ada blok warna | muncul pesan “pindah meja berhasil dilakukan ke meja x-x” | |
| 14 | Mengetahui respon sistem ketika pindah ke meja yang sudah terisi | Pindah ke meja yang terblok warna merah | Tidak terjadi apa-apa | |
| 15 | Mengetahui respon sistem ketika pindah ke meja yang sudah dipesan | Pindah ke meja yang terblok warna ungu | muncul pesan “meja sudah <i>dibooking</i> oleh bapak/ibu”sukses | |

g. Desain Uji Coba Form Gabung Meja

Berfungsi untuk mengecek bila dilakukan penggabungan meja.

Tabel 3.35 Desain *Test Case* Form Gabung Meja

| Test Case | Tujuan | Input | Output Diharapkan | Status |
|-----------|-------------------|----------------|-------------------|--------|
| 16 | Mengetahui respon | gabung ke meja | muncul pesan | |

| | | | | |
|----|---|---|---|--|
| | sistem ketika gabung meja ke meja yang kosong | yang tidak ada blok warna | “Gabung meja berhasil” | |
| 17 | Mengetahui respon sistem ketika gabung meja pada meja yang sudah terisi | gabung ke meja yang terblok warna merah | Tidak terjadi apa-apa | |
| 18 | Mengetahui respon sistem ketika gabung ke meja yang dipesan | gabung ke meja yang terblok warna ungu | muncul pesan “meja sudah <i>dibooking</i> oleh bapak/ibu ...” | |

3.5.1 Desain Uji Coba Fungsi Aplikasi pada Checker

a. Desain Uji Coba Form Login

Uji coba form *login* pada aplikasi *desktop* bagian *checker* berfungsi untuk menguji fungsi login berjalan sesuai yang diharapkan

Tabel 3.36 Desain *Test Case* Form Login

| Test Case | Tujuan | Input | Output Diharapkan | Status |
|-----------|--|-----------------------------------|-------------------------------|--------|
| 19 | Mengetahui respon sistem terhadap data <i>user</i> valid | Nama user dan password checker | Muncul pesan “Login berhasil” | |
| 20 | Mengetahui respon sistem terhadap data <i>user</i> tidak valid | Nama user dan password yang salah | Muncul pesan “gagal” | |

b. Desain Uji Coba Form Master Menu

Uji coba form master menu pada aplikasi *desktop* bagian *checker* berfungsi untuk mengetahui fungsi dari proses pengisian data master menu

Tabel 4.37 Desain *Test Case* Form Master Menu

| Test Case | Tujuan | Input | Output Diharapkan | Status |
|-----------|---|--|------------------------|--------|
| 21 | Mengetahui respon sistem ketika Mengisi data master menu dengan lengkap dan benar | Mengisi nama menu, pilih jenis menu, harga | Data masuk dalam tabel | |
| 22 | Mengetahui respon sistem | Mengosongi | Muncul pesan “ | |

| | | | | |
|----|---|----------------------|-------------------------|--|
| | ketika terdapat kolom inputan yang kosong | kolom harga | isikan seluruh data” | |
| 23 | Mengetahui respon sistem ketika mengubah data menu | tekan tombol “ubah” | Data pada table berubah | |
| 24 | Mengetahui respon sistem ketika menghapus menu yang tersimpan | tekan tombol “hapus” | Data pada table hilang | |

c. Desain Uji Coba Form Master Ruangan

Uji coba form master ruangan pada aplikasi *desktop checker* berfungsi untuk mengetahui fungsi dari proses pengisian data master ruangan.

Tabel 3.38 Desain *Test Case* Form Master Ruangan

| Test Case | Tujuan | Input | Output Diharapkan | Status |
|-----------|--|---|-------------------------------------|--------|
| 25 | Mengetahui respon sistem ketika mengisi data master ruangan dengan benar dan lengkap | Mengisi data lantai, deskripsi dan mengisi denah meja | Data masuk dalam tabel | |
| 26 | Mengetahui respon sistem ketika terdapat kolom master ruangan yang kosong | Hanya mengisi kolom lantai | Muncul pesan “ isikan seluruh data” | |
| 27 | Mengetahui respon sistem ketika mengubah data ruangan | tekan tombol “ubah” | Data dalam table berubah | |
| 28 | Mengetahui respon sistem ketika menghapus menu yang tersimpan | Klik ruangan yang ingin dihapus kemudian tekan tombol “hapus” | Data pada table hilang | |

d. Desain Uji Coba Form Master User

Uji coba form master *user* pada aplikasi *desktop checker* berfungsi untuk mengetahui fungsi dari proses pengisian data master. seperti pada tabel 3.39.

Tabel 3.39 Rancangan *Test Case* Form Master *User*

| Test Case | Tujuan | Input | Output Diharapkan | Status |
|-----------|--|---|------------------------------------|--------|
| 29 | Mengetahui respon sistem ketika mengisi data master <i>user</i> dengan lengkap dan benar | Mengisi <i>username</i> , <i>password</i> dan <i>re-password</i> | Data masuk dalam tabel | |
| 30 | Mengetahui respon sistem ketika terdapat kolom inputan master yang kosong | Mengisi <i>username</i> , <i>password</i> dan <i>re-password</i> dan pilih tipe | Muncul pesan “isikan seluruh data” | |
| 31 | Mengetahui respon sistem ketika mengubah data <i>user</i> | tekan tombol “ubah” | Data pada table berubah | |
| 32 | Mengetahui respon sistem ketika menghapus <i>user</i> yang tersimpan | Tekan tombol “hapus” | Data pada table hilang | |

e. **Desain Uji Coba Form *Checking* Pesanan**

Uji coba form *checking* pesanan pada aplikasi *desktop* bagian *checker* bertujuan mengetahui fungsi pengontrolan pesanan sesuai yang diharapkan

Tabel 3.40 Desain *Test Case* Form *Checking* Pesanan

| Test Case | Tujuan | Input | Output Diharapkan | Status |
|-----------|---|---|--|--------|
| 33 | Mengetahui respon sistem ketika Mengubah status pesanan menjadi “proses” | Pilih menu pesanan berstatus “menunggu” kemudian klik tombol “ganti status” | Status pada table berubah menjadi “proses” | |
| 34 | Mengetahui respon sistem ketika mengubah status pesanan menjadi “selesai” | Pilih menu pesanan berstatus proses kemudian tekan tombol “ganti status” | Data pada table hilang | |

f. **Desain Uji Coba Form Riwayat Pesanan**

Uji coba form riwayat pesanan pada aplikasi *desktop* bagian *checker* bertujuan untuk mengetahui fungsi pembatalan pesanan pada *list* riwayat.

Tabel 3.41 Desain *Test Case* Form Riwayat Pesanan

| Test Case | Tujuan | Input | Output Diharapkan | Status |
|-----------|--|---|---|--------|
| 35 | Mengetahui respon sistem ketika membatalkan status pesanan “selesai” | Klik <i>item</i> pesanan yang ingin dirubah | Data hilang dari table dan muncul di menu checker | |

g. Desain Uji Coba Form Stok Menu

Uji coba form stok menu pada aplikasi *desktop* bagian *checker* berfungsi untuk mengetahui proses pengisian stok berjalan sesuai yang diharapkan.

Tabel 3.42 Desain *Test Case* Form Stok Menu

| Test Case | Tujuan | Input | Output Diharapkan | Status |
|-----------|---|--|--|--------|
| 36 | Mengetahui respon sistem ketika mengisi jumlah stok menu per- <i>item</i> | Mengisi jumlah stok lalu klik “simpan” | Stok tersimpan dan tombol “set semua” dan “simpan” tidak dapat di klik | |
| 37 | Mengetahui respon sistem ketika mengisi jumlah stok menu secara keseluruhan | Tekan tombol “set semua” | Muncul form “masukkan jumlah stok” | |

3.5.2 Desain Uji Coba Fungsi Aplikasi pada Bagian Dapur

a. Desain Uji Coba Form Tampilan Dapur

Uji coba form dapur pada aplikasi *desktop* bagian dapur berfungsi untuk mengetahui fungsi pilihan tampilan untuk layar dapur.

Tabel 3.43 Desain *Test Case* Form Tampilan Dapur

| Test Case | Tujuan | Input | Output Diharapkan | Status |
|-----------|---|--|--|--------|
| 38 | Mengetahui respon sistem ketika memilih jenis tampilan makanan dan minuman digabung | Pilih jenis tampilan lalu pilih “gabung makanan dan minuman” | Tampil list makanan dan minuman dalam satu tabel | |
| 39 | Mengetahui respon | Pilih jenis | Tampil | |

| | | | | |
|----|--|---|---|--|
| | sistem ketika memilih jenis tampilan makanan dan minuman secara terpisah | tampilan lalu pilih “pisah makanan dan minuman” | makanan dan minuman dalam masing-masing tabel | |
| 40 | Mengetahui respon sistem ketika memilih jenis tampilan makanan | Pilih jenis tampilan lalu pilih “hanya makanan” | Tampil makanan saja | |
| 41 | Mengetahui respon sistem ketika memilih jenis tampilan minuman | Pilih jenis tampilan lalu pilih “hanya minuman” | Tampil minuman saja | |

3.5.3 Desain Uji Coba Fungsi Aplikasi pada Kasir

a. Desain Uji Coba Form Pembayaran

Uji coba form pembayaran pada aplikasi *desktop* bagian kasir bertujuan untuk mengetahui fungsi proses pembayaran sesuai dengan yang diharapkan.

Tabel 4.44 Desain *Test Case* Form Pembayaran

| Test Case | Tujuan | Input | Output Diharapkan | Status |
|-----------|---|--|---|--------|
| 42 | Mengetahui respon sistem ketika mengisi data pembayaran dengan benar | Mengisi data pembayaran | Muncul form print struk | |
| 43 | Mengetahui respon sistem ketika data pembayaran tidak lengkap | Mengkosongi kolom bayar | Muncul pesan “isikan seluruh data” | |
| 44 | Mengetahui respon sistem ketika mengisi nomor meja yang salah | Mengisi nomor meja yang kosong | Muncul pesan “data tidak ditemukan” | |
| 45 | Mengetahui respon sistem ketika jumlah nominal yang dibayarkan kurang | Mengisi nominal “bayar” kurang dari total pembayaran | Muncul pesan “jumlah pembayaran kurang” | |

b. Desain Uji Coba Form *Reservasi*

Uji coba form dapur pada aplikasi *desktop* bagian kasir berfungsi untuk mengetahui fungsi proses *reservasi* berjalan sesuai dengan yang diharapkan.

Tabel 3.45 Rancangan *Test Case* Form *Reservasi*

| Test Case | Tujuan | Input | Output Diharapkan | Status |
|-----------|---|--|------------------------------------|--------|
| 46 | Mengetahui respon sistem ketika menyimpan data <i>reservasi</i> dengan benar | Mengisi data <i>reservasi</i> dengan lengkap | Data muncul pada tabel | |
| 47 | Mengetahui respon sistem ketika menyimpan data <i>reservasi</i> tidak lengkap | Tanpa mengisi data no.Telp | Muncul pesan "Isikan seluruh data" | |
| 48 | Mengetahui respon sistem ketika mengubah data <i>reservasi</i> | Mengubah data yang sudah tersimpan. | Data pada table berubah | |
| 49 | Mengetahui respon sistem ketika menghapus data <i>reservasi</i> | Menghapus item <i>reservasi</i> yang sudah tersimpan | Data pada table hilang | |

3.5.4 Desain Uji Coba Fungsi Aplikasi pada Manajer

a. Desain Uji Coba Form Laporan Penjualan

Uji coba form laporan penjualan pada aplikasi *desktop* bagian manajer berfungsi untuk mengetahui fungsi dari proses menampilkan laporan penjualan.

Tabel 3.46 Desain *Test Case* Form Laporan Penjualan

| Test Case | Tujuan | Input | Output Diharapkan | Status |
|-----------|---|---------------------------------|----------------------------------|--------|
| 50 | Mengetahui respon sistem ketika menampilkan laporan penjualan harian | Memilih jenis laporan "harian" | Tampil laporan penjualan harian | |
| 51 | Mengetahui respon sistem ketika menampilkan laporan penjualan bulanan | Memilih jenis laporan "bulanan" | Tampil laporan penjualan bulanan | |

b. Desain Uji Coba Form Laporan Utilitas Meja

Uji coba form laporan utilitas meja aplikasi *desktop* bagian Manajer berfungsi mengetahui fungsi sistem saat menampilkan laporan utilitas meja

Tabel 3.47 Desain *Test Case* Form Laporan Utilitas Meja

| Test Case | Tujuan | Input | Output Diharapkan | Status |
|-----------|---|---------------------------------|--------------------------------------|--------|
| 52 | Mengetahui respon sistem ketika menampilkan laporan utilitas meja harian | Memilih jenis laporan "harian" | Tampil laporan utilitas meja harian | |
| 53 | Mengetahui respon sistem ketika menampilkan laporan utilitas meja bulanan | Memilih jenis laporan "bulanan" | Tampil laporan utilitas meja bulanan | |

c. Desain Uji Coba Form Laporan Menu Favorit

Uji coba form laporan menu favorit aplikasi *desktop* bagian Manajer bertujuan untuk mengetahui fungsi dari proses menampilkan laporan menu favorit

Tabel 3.48 Desain *Test Case* Form Laporan Menu Favorit

| Test Case | Tujuan | Input | Output Diharapkan | Status |
|-----------|--|---------------------------------|-------------------------------------|--------|
| 54 | Mengetahui respon sistem ketika menampilkan laporan menu favorit harian | Memilih jenis laporan "harian" | Tampil laporan menu favorit harian | |
| 55 | Mengetahui respon sistem ketika menampilkan laporan menu favorit bulanan | Memilih jenis laporan "bulanan" | Tampil laporan menu favorit bulanan | |