BAB III

ANALISIS DAN PERANCANGAN SISTEM

Pada bab ini akan dibahas tentang identifikasi dan analisa permasalahan, solusi permasalahan dan perancangan dari aplikasi *automated customer service* yang akan dibagi kedalam tiga tahapan, yaitu tahap awal, tahap pengembangan, dan tahap akhir. Tahap awal terdiri dari analisa bisnis dan studi literatur. Untuk tahap pengembangan terdiri dari dari *requirements modeling, architecture design, concept design*, dan *code generation*. Dan tahap akhir berisikan *unit testing*, *integration testing*, *system testing*, dan uji akurasi aplikasi. Lebih jelasnya dari pembagian tahap tersebut dapat dilihat pada tabel berikut,

Tabel 3.1 Metodologi Penelitian

Tahapan	Kegiatan
Tahap Awal	- Analisis Bisnis - Studi Literatur
Tahap Pengembangan	- *Requirements Modeling* - *Architecture Design* - *Component Design* - *Code Generation*
Tahap Akhir	- *Unit Testing* - *Integration Testing* - *System Testing* - Uji Akurasi Aplikasi - Uji Perbandingan antara menggunakan aplikasi dan manual

3.1 Tahap Awal

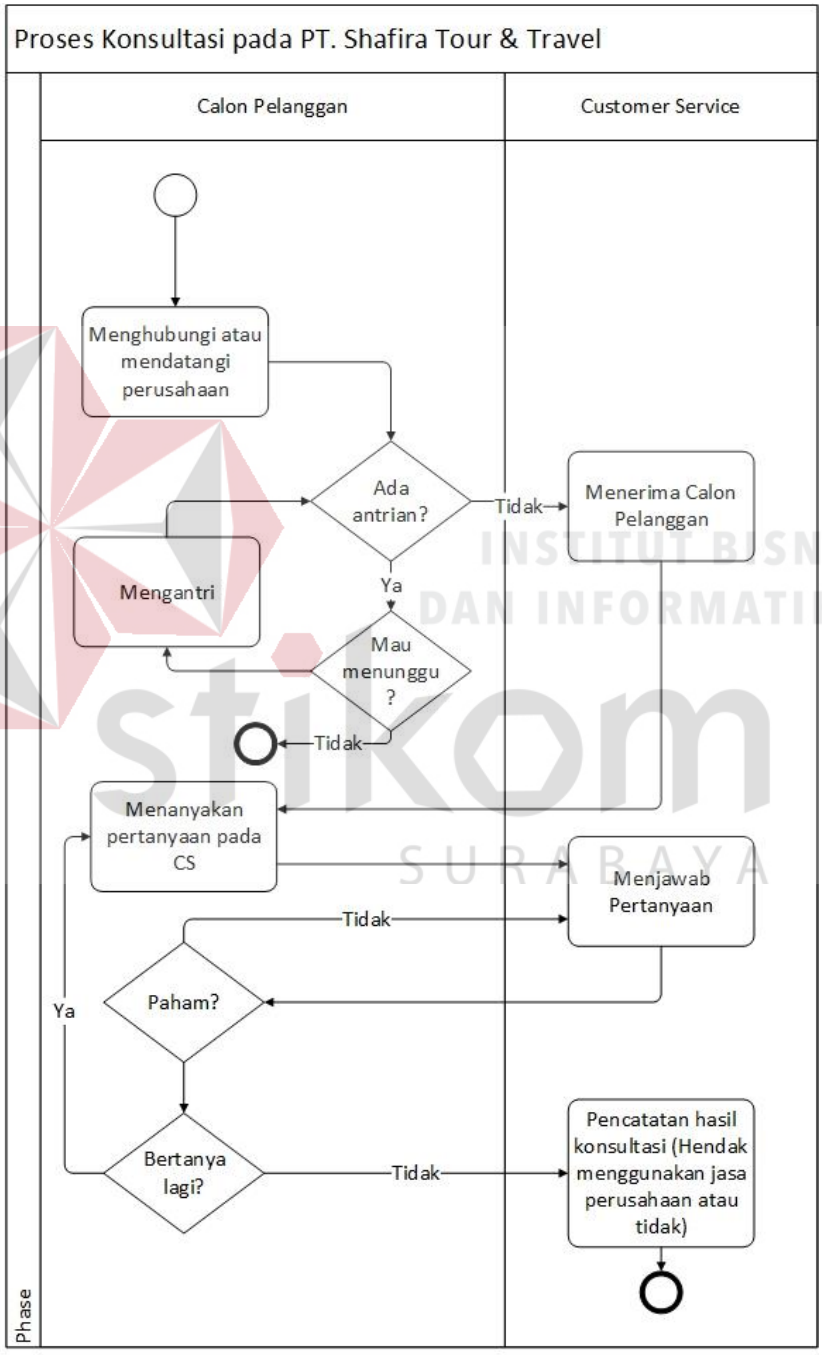
3.1.1 Analisis Bisnis

Analisis bisnis merupakan proses analisis dari proses bisnis dan mengidentifikasi permasalahan yang ada pada perusahaan PT. Shafira Tour & Travel. Kegiatan dari anilisis bisnis ini berupa wawancara dan observasi secara langsung pada perusahaan PT. Shafira Tour & Travel, yang dimana untuk proses wawancara akan dilakukan dengan pihak terkait yakni bagian *customer service*. Hasil dari wawancara dan observasi tersebut berupa proses konsultasi calon pelanggan yang dijalankan pada perusahaan saat ini, pihak-pihak yang terkait dalam menangani proses konsultasi tersebut, serta data durasi konsultasi yang pernah dilakukan dalam lima hari terakhir. Untuk data durasi konsultasi pada perusahaan PT. Shafira Tour & Travel dapat dilihat pada lampiran 1.

Selain data durasi konsultasi, terdapat juga informasi berupa proses konsultasi calon jemaah pada PT. Shafira Tour & Travel. Proses konsultasi dimulai dari calon pelanggan yang ingin tahu lebih spesifik mengenai layanan yang diberikan oleh perusahaan bertanya kepada bagian *customer service* perusahaan, baik itu datang secara langsung ke kantor, ke tempat expo, telepon, sms, ataupun *email*. Segala pertanyaan yang diajukan oleh pihak calon pembeli akan dijawab oleh bagian *customer service*.

Setelah sesi tanya-jawab yang dilakukan oleh calon pelanggan dilakukan, pihak *customer service* akan mencatat secara garis besar kegiatan yang terjadi. Hal ini akan berlanjut sampai dengan didapat keputusan akhir dari calon pelanggan akan menggunakan jasa yang ditawarkan atau tidak. Dan setiap keputusan dari calon pelanggan tersebut akan dibuatkan sebuah laporan dari hasil konsultasi yang

nantinya menunjukkan berapa banyak calon pelanggan yang akan menggunakan jasa perusahaan dan akan dihubungkan dengan pengeluaran perusahaan dalam memasarkan jasa tersebut. Untuk lebih jelasnya, proses konsultasi jemaah pada PT. Shafira Tour & Travel dapat dilihat pada gambar berikut,



Gambar 3.1 BPMN Proses Konsultasi Calon Jemaah PT. Shafira Tour & Travel

Berdasarkan proses konsultasi tersebut permasalahan yang terjadi pada PT. Shafira Tour & Travel adalah terjadinya antrean pelanggan yang akan berdampak pada menurunnya kepuasan pelanggan. Sehingga jika hal ini berlanjut secara terus menerus akan mengakibatkan menurunnya pendapatan perusahaan secara tidak langsung.

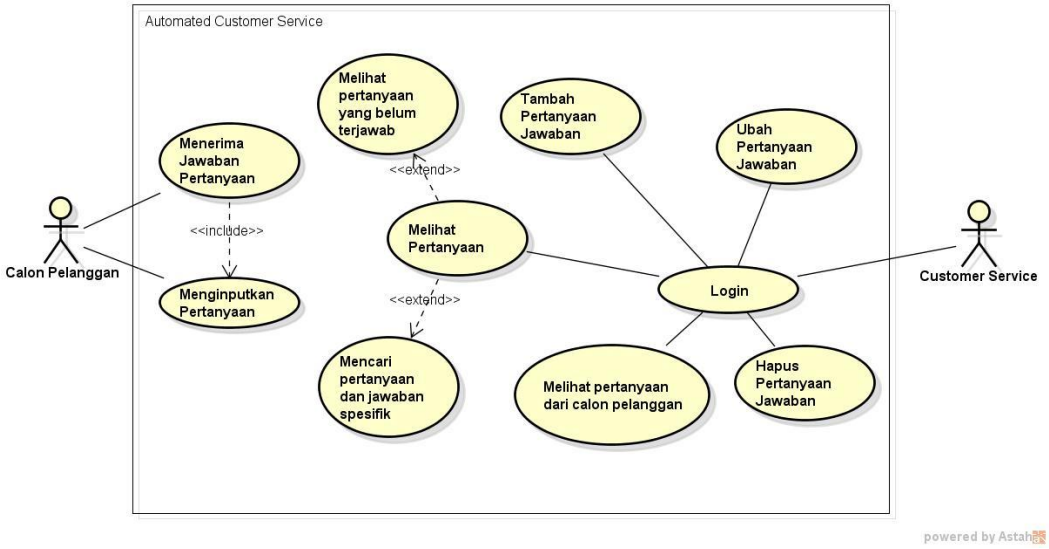
3.1.2 Studi Literatur

Studi Literatur merupakan proses yang membantu dalam mencari penyelesaian permasalahan yang dimiliki oleh perusahaan. Studi literatur ini dapat diperoleh dari buku ataupun internet. Literatur-literatur yang menjadi acuan dalam menyelesaikan permasalahan yang ada adalah *customer service*, *automasi*/otomasi, *knowledge-based system*, *text mining*, *natural language processing*, *system development life-cycle*, dan *v-shape model*.

3.2 Tahap Pengembangan

3.2.1 *Requirements Modeling*

Requirements Modeling merupakan tahapan yang bertujuan untuk mengetahui kebutuhan dari system yang akan dibangun. Dalam menggambarkan kebutuhan tersebut, akan dibagi kedalam beberapa diagram yaitu *usecase diagram, activity diagram, sequence diagram, dan class diagram*. Yang pertama adalah *usecase* diagram, dimana diagram ini dibuat dengan dasar dari hasil analisis bisnis yang telah dilakukan pada tahapan awal. Lebih jelasnya untuk *usecase diagram* yang dibangun dapat dilihat pada gambar berikut.



Gambar 3.2 *Usecase Diagram Automated Customer Service*

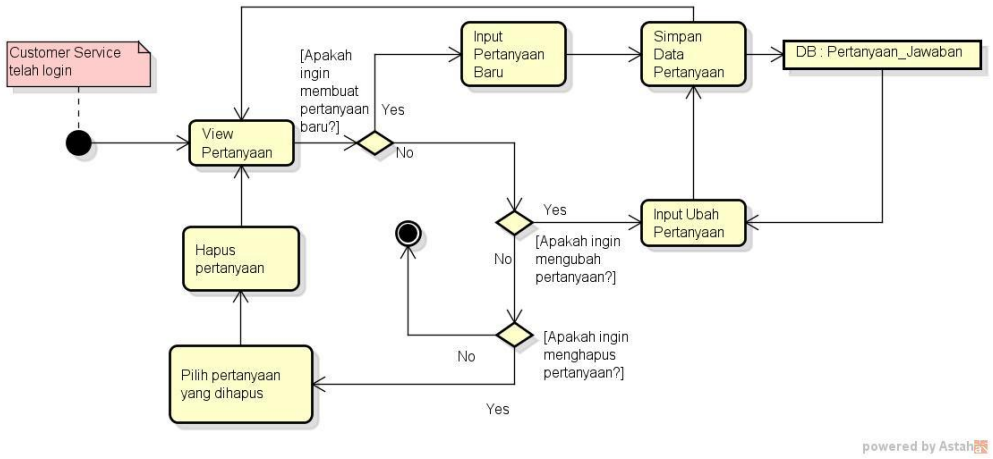
Untuk *usecase diagram automated customer service* dari aplikasi yang akan dibangun terdiri dari 2 (dua) aktor yaitu *customer service* dan calon pelanggan. Selain aktor terdapat 7 (tujuh) aksi, yaitu aksi menginputkan pertanyaan, menerima jawaban, login, *maintenance* data pertanyaan, melihat data pertanyaan, melihat pertanyaan yang belum terjawab, dan melihat pertanyaan dari calon pelanggan. Aksi menerima jawaban pertanyaan akan dapat diakses apabila calon pelanggan telah melakukan aksi menginputkan pertanyaan, begitupula dengan aksi maintenance pertanyaan, melihat data pertanyaan, dan melihat pertanyaan dari calon pelanggan dapat diakses apabila *customer service* telah melakukan aksi login terlebih dahulu. Sedangkan untuk aksi melihat pertanyaan yang belum terjawab adalah aksi tambahan dari melihat data pertanyaan dimana aksi ini terpicu apabila *customer service* ingin melihat pertanyaan yang belum terjawab oleh sistem.

Beberapa kebutuhan yang telah dijabarkan pada gambar diatas apabila diubah kedalam bentuk tabel adalah sebagai berikut,

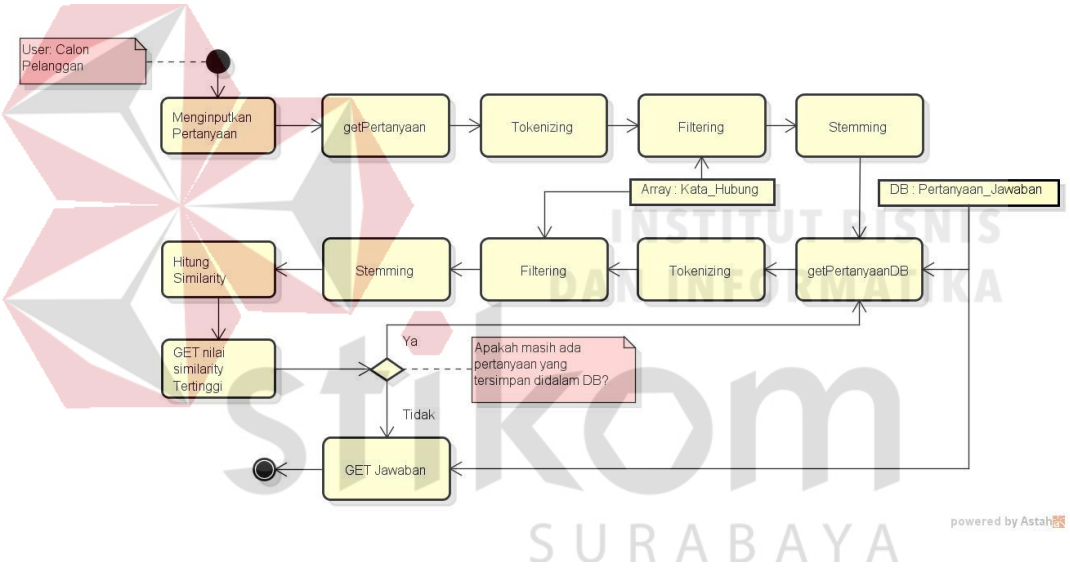
Tabel 3.2 Kebutuhan Aplikasi

No	Nama Kebutuhan	Jenis Kebutuhan
1.	Calon pelanggan dapat menginputkan pertanyaan	Fungsional
2.	Calon pelanggan dapat menerima jawaban pertanyaan	Fungsional
3.	*Customer service* dapat melakukan tambah pertanyaan dan jawaban	Fungsional
4.	*Customer service* dapat melakukan ubah pertanyaan dan jawaban	Fungsional
5.	*Customer service* dapat melakukan hapus pertanyaan dan jawaban	Fungsional
6.	*Customer service* dapat melihat data pertanyaan dan jawaban	Fungsional
7.	*Customer service* dapat melihat pertanyaan yang diajukan oleh calon pelanggan	Fungsional
8.	*Customer service* dapat melihat pertanyaan yang belum terjawab	Non-fungsional
9.	*Customer service* dapat mencari pertanyaan dan jawaban spesifik	Non-fungsional

Kemudian, diagram selanjutnya adalah *activity diagram*. *Activity diagram* merupakan diagram yang digunakan untuk menggambarkan alur sistem *automated customer service* berdasarkan sudut pandang penggunanya. *Activity diagram* yang dibangun dibagi kedalam dua sisi, yaitu untuk menjawab pertanyaan dan *maintenance* pertanyaan dan jawaban.



Gambar 3.3 *Activity Diagram Maintenance* Pertanyaan dan Jawaban



Gambar 3.4 *Activity Diagram* Menjawab Pertanyaan

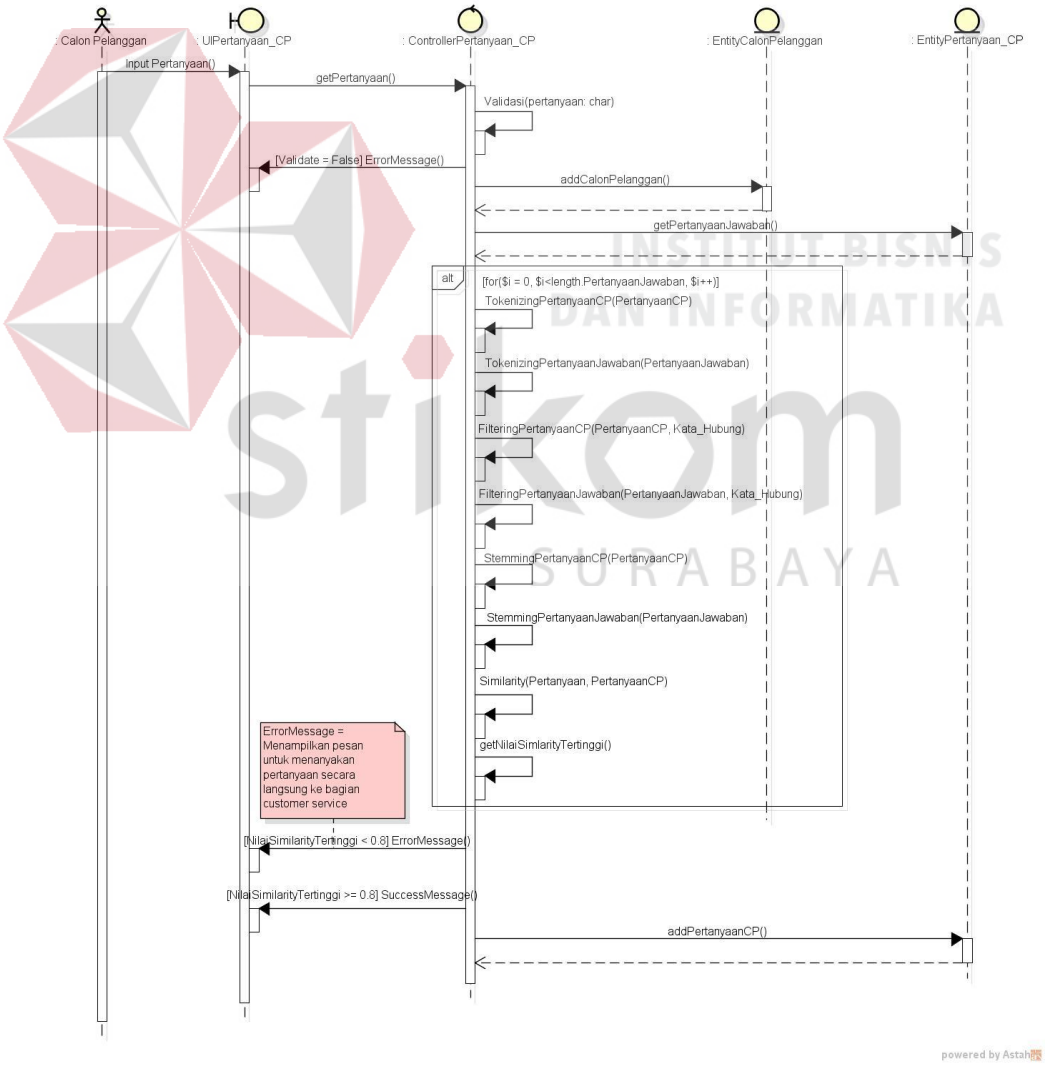
Untuk proses *maintenance* pertanyaan dan jawaban dimulai dari *customer service* yang telah login dapat melihat pertanyaan kemudian dilanjutkan dengan pilihan apakah hendak membuat pertanyaan baru atau tidak. Apabila *customer service* memutuskan untuk membuat pertanyaan baru maka pengguna akan menginputkan data pertanyaan baru kemudian akan disimpan kedalam sistem dan berakhir pada aksi untuk melihat daftar pertanyaan kembali.

Jika *customer service* tidak ingin membuat pertanyaan baru, maka akan dialihkan pada keputusan untuk mengubah pertanyaan. Jika ingin mengubah pertanyaan maka pengguna menginputkan data pertanyaan dan jawaban yang baru kemudian sistem akan mengubah data yang lama menjadi data yang baru. Dan apabila sampai pada posisi ini pengguna tidak inign mengubah data pertanyaan dan jawaban, maka muncul pertanyaan baru berupa apakah pengguna akan menghapus pertanyaan. Apabila pengguna ingin menghapus pertanyaan maka pengguna harus memilih data mana yang ingin dihapus dan jika sudah pasti maka sistem akan menghapus pertanyaan tersebut. Jika tidak, maka sistem *maintenance* pertanyaan dan jawaban berakhir.

Untuk penjabaran dari *activity diagram* menjawab pertanyaan dimulai dari calon pelanggan menginputkan pertanyaan. Setelah itu sistem akan melakukan proses *tokenizing*, *filtering*, dan *stemming*. *Tokenizing* merupakan proses pemecahan satu kalimat menjadi beberapa kata yang tersimpan kedalam array. *Filtering* digunakan untuk membuat kata hubung yang dimiliki. *Stemming* bertujuan untuk merubah setiap kata yang tersimpan didalam array menjadi kata dasar. Apabila rangkaian proses tersebut telah selesai, maka sistem akan mengambil semua pertanyaan yang telah dimiliki dalam basis data. Dan setiap pertanyaan dari basis data tersebut akan dilakukan proses *tokenizing*, *filtering*, *stemming*, dan menghitung *similarity*. Untuk proses *similarity* yang dilakukan adalah mencocokan hasil *stemming* dari masukan calon pelanggan dengan hasil *stemming* pertanyaan dari basis data. Setelah itu akan dihitung tingkat *similarity* nya dan diambil nilai *similarity* tertinggi. Apabila nilai *similarity* tertinggi mencapai nilai 0.8 atau 80% maka sistem akan memberikan jawaban berdasarkan pertanyaan dari basis data

yang memiliki nilai *similarity* tertinggi tersebut, jika nilai tersebut tidak mencapai 0.8 atau 80% maka sistem akan memberikan saran untuk melakukan konsultasi secara langsung pada bagian *customer service* yang ada di perusahaan.

Setelah *usecase* dan *activity diagram* dibuat selanjutnya adalah *sequence diagram*. Sama seperti halnya *activity diagram*, *sequence diagram* merupakan gambaran sistem akan tetapi menggunakan sudut pandang sistem. Dalam *sequence diagram* akan lebih fokus akan proses yang ada dalam sistem. Untuk lebih jelasnya dapat dilihat pada gambar berikut,



Gambar 3.5 *Sequence Diagram* Menjawab Pertanyaan

Berdasarkan *sequence diagram* aplikasi *automated customer service* diatas, aplikasi ini dimulai dari calon pelanggan menginputkan pertanyaan kedalam aplikasi melalui *interface* yang ada. Kemudian pertanyaan tersebut akan ditangkap oleh *controller* untuk dilakukan validasi. Apabila dalam proses validasi ini terdapat kesalahan, maka dari kesalahan tersebut akan muncul pada *interface* aplikasi. Validasi yang dilakukan berupa validasi panjang masukan dan tipe masukan.

Apabila dalam proses validasi tidak ada masalah, yang dilakukan *controller* adalah menyimpan data calon pelanggan kedalam basis data. Setelah itu *controller* akan mengambil semua data pertanyaan yang tersimpan didalam basis data. Setiap data pertanyaan dari basis data tersebut akan diolah kedalam proses *tokenizing*, *filtering*, dan *stemming* bersama dengan pertanyaan yang telah diinputkan oleh calong pelanggan.

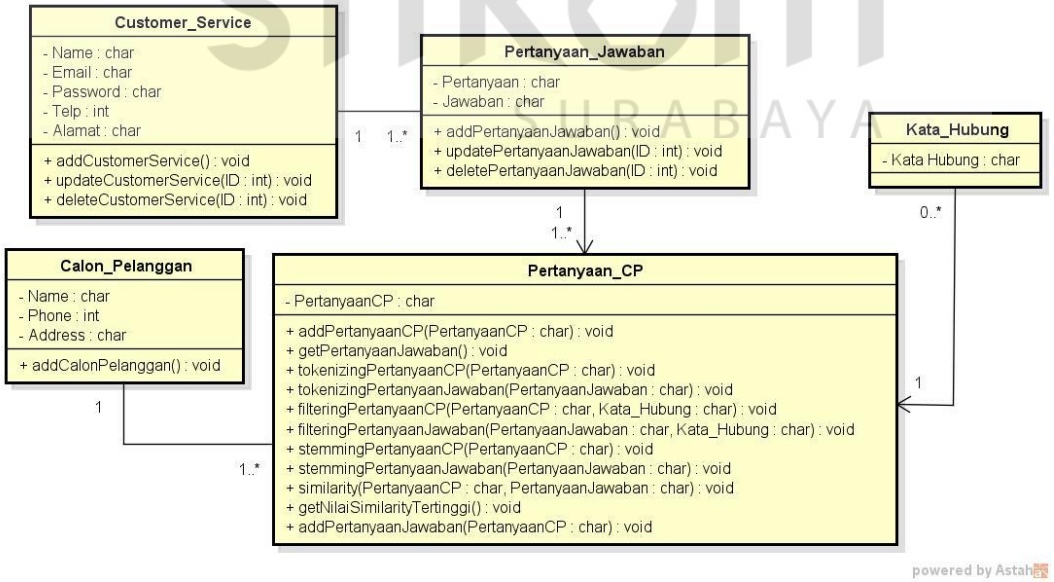
Tokenizing merupakan proses pemecahan kalimat menjadi beberapa kata yang disimpan kedalam *array*. *Filtering* digunakan untuk menghapus kata-kata yang tidak diperlukan, dalam hal ini adalah kata hubung, dari *array* yang telah dibuat pada proses tokenizing. Kemudian *stemming* merupakan proses untuk mengubah setiap kata dalam *array* menjadi kata dasar atau bentuk dasar dari kata-kata tersebut.

Setelah proses *tokenizing*, *filtering*, dan *stemming* selesai yang dilakukan adalah menghitung *similarity* atau kemiripan antara pertanyaan yang diinputkan oleh calon pelanggan dan pertanyaan yang diambil dari basis data. Kemudian akan diambil data pertanyaan dari basis data yang memiliki nilai *similarity* paling tinggi dari beberapa kali proses yang diulang sejumlah dengan banyaknya pertanyaan dalam basis data. Apabila nilai *similarity* tertinggi mencapai 0.8 maka controller

akan memberikan jawaban berdasarkan pertanyaan yang memiliki nilai *similarity* tertinggi tersebut, akan tetapi jika nilai similarity tertinggi tidak bias mencapai 0.8 maka aplikasi akan menampilkan pesan kepada calon pelanggan melalui *interface* untuk menyarankan bertanya secara langsung kepada *customer service* yang ada di perusahaan. Proses terakhir yang dilakukan selanjutnya adalah menyimpan data pertanyaan yang telah diinputkan oleh calon pelanggan kedalam basis data.

Selain *sequence diagram* menjawab pertanyaan, masih terdapat beberapa *sequence diagram* yang diantaranya adalah *sequence diagram* melihat pertanyaan, *sequence diagram* tambah pertanyaan, *sequence diagram* ubah pertanyaan, dan *sequence diagram* hapus diagram. *Sequence diagram* tersebut dapat dilihat pada bagian lampiran.

Diagram terakhir yang ada pada *requirements modeling* adalah *class diagram*. *Class diagram* yang dibangun untuk aplikasi *automated customer service* dapat dilihat pada gambar berikut,

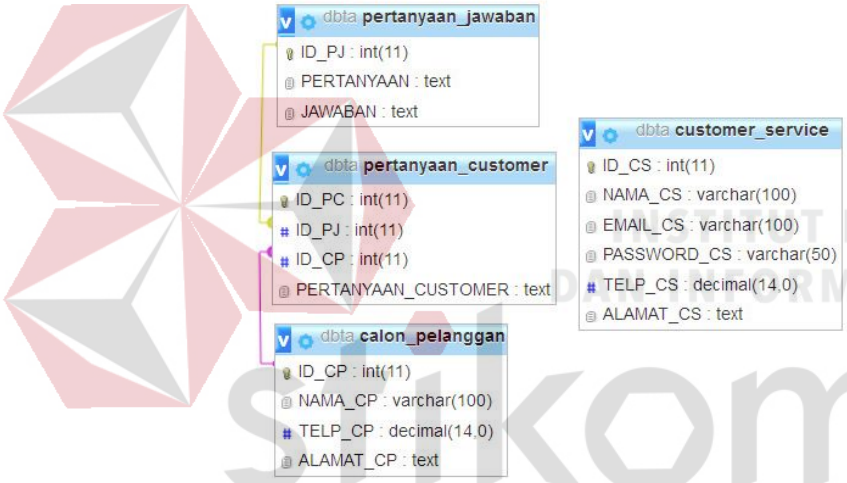


Gambar 3.6 *Class Diagram* Aplikasi *Automated Customer Service*

Berdasarkan *class diagram* diatas, terdapat 5 kelas, yaitu kelas Customer Servie, kelas Pertanyaan dan Jawaban, kelas Pertanyaan Calon Jemaah, kelas Calon Jemaah dan kelas Kata_Hubung. Masing-masing kelas tersebut memiliki relasi masing-masing dimana untuk relasi antar kelas tersebut berupa relasi one-to-many.

3.2.2 *Architecture Design*

Pada tahapan ini akan menghasilkan rancangan arsitektur data yang ada pada aplikasi *automated customer service*. Rancangan arsitektur data tersebut dapat dilihat pada gambar berikut,

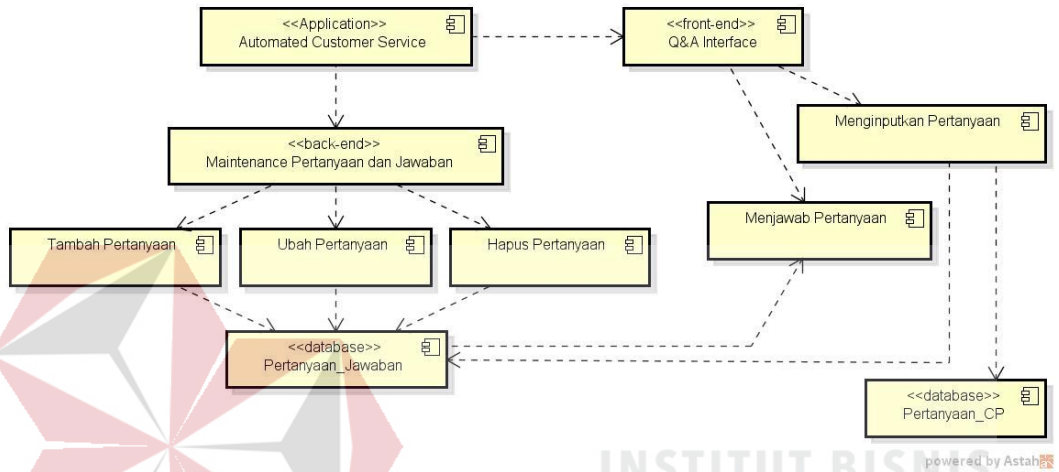


Gambar 3.7 Struktur Basis Data Aplikasi *Automated Customer Service*

Berdasarkan struktur basis data aplikasi *automated customer service* di atas, terdiri dari 4 (empat) tabel, yaitu tabel pertanyaan_jawaban untuk menyimpan data pertanyaan dan jawaban, table pertanyaan_customer untuk menyimpan data pertanyaan yang pernah diajukan oleh calon pelanggan, tabel calon_pelanggan untuk menyimpan data calon pelanggan yang pernah menggunakan aplikasi *automated customer service*, dan tabel customer_service untuk menyimpan data pengguna yang dapat melakukan tambah, ubah, dan hapus pertanyaan dan jawaban.

3.2.3 *Component Design*

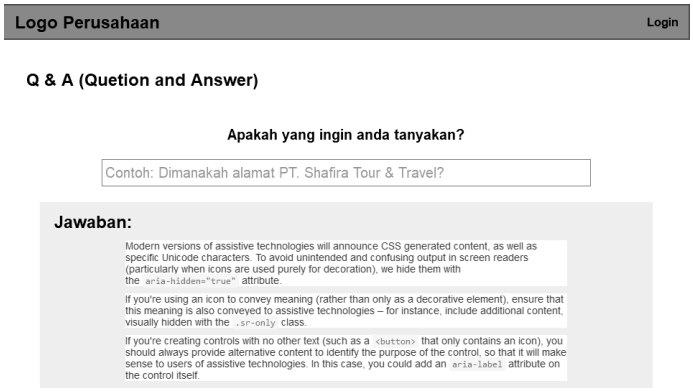
Componen design merupakan proses sebelum dilakukannya koding (*code generation*). Hasil dari *component design* ini berupa *component diagram* dan desain antar-muka dari aplikasi yang akan dibangun. Untuk *component diagram* aplikasi *automated customer service* dapat dilihat pada bagan berikut,



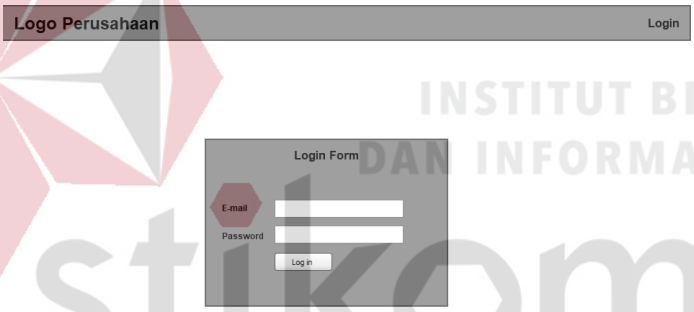
Gambar 3.8 *Component Diagram Automated Customer Service*

Secara garis besar aplikasi *automated customer service* terbagi kedalam 2 bagian, yaitu untuk mengelola atau *maintenance* pertanyaan dan jawaban dan untuk tampilan antar muka dari aplikasi *automated customer service*. Untuk tampilan antar muka dari aplikasi berisikan mengajukan pertanyaan dan menjawab pertanyaan yang merupakan satu kesatuan. Kemudian, untuk mengelola pertanyaan dan jawaban terpecah menjadi 3 komponen, yaitu tambah pertanyaan dan jawaban, ubah pertanyaan dan jawaban dan hapus pertanyaan dan jawaban. Untuk tambah pertanyaan, ubah pertanyaan, dan hapus pertanyaan memiliki relasi dengan basis data Pertanyaan_Jawaban sedangkan untuk menginputkan pertanyaan memiliki relasi dengan basis data Pertanyaan_Jawaban dan Pertanyaan_CP.

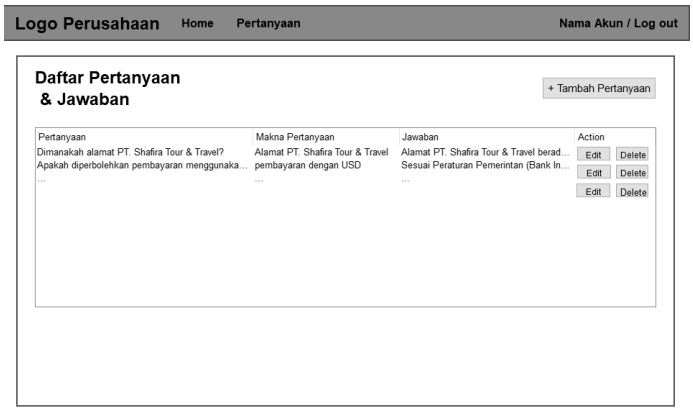
Berdasarkan komponen yang telah dijabarkan diatas, hasil dari desain antar-muka yang dihasilkan adalah sebagai berikut,



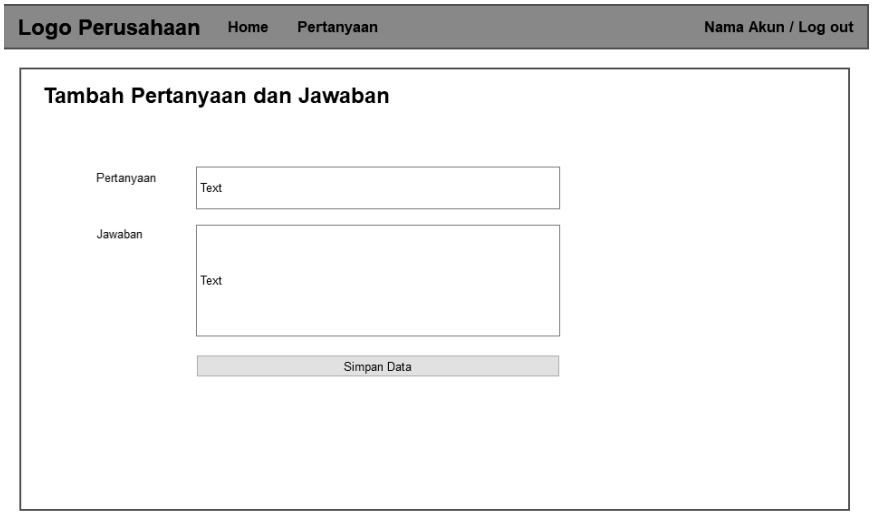
Gambar 3.9 Desain Front-End *Automated Customer Service*



Gambar 3.10 Desain Login Page *Automated Customer Service*



Gambar 3.11 Desain Tampilan Daftar Pertanyaan & Jawaban



Gambar 3.12 Desain Tampilan Form Tambah dan Ubah Data

3.2.4 *Code Generation*

Code generation merupakan tahapan pembuatan aplikasi berdasarkan kebutuhan yang telah ditentukan pada proses *requirements modeling* sampai dengan *component design*. Hasil dari proses ini merupakan aplikasi yang menjadi solusi dari permasalahan yang ada pada perusahaan. Bahasa yang digunakan untuk membangun solusi yang diajukan adalah PHP 7.0.13 dengan *framework* laravel versi 5.4.

3.3 Tahap Akhir

3.3.1 *Unit Testing*

Pada tahap ini akan dilakukan pengujian berdasarkan fungsi-fungsi yang dimiliki dari tiap-tiap objek. Fungsi-fungsi tersebut antara lain *tokenizing*, *filtering*, *stemming*, dan *semantic*. Untuk desain testing yang akan dilakukan pada tahapan ini adalah sebagai berikut,

Tabel 3.3 Desain *Unit Testing*

No	Nama Fungsi	Inputan	Hasil yang diharapkan
1.	Tokenizing	Anton sedang mempelajari matakuliah akuntansi sebelum UTS	[Anton], [sedang], [mempelajari], [matakuliah], [akuntansi], [sebelum], [UTS]
2.	Filtering	Saya pergi ke malang untuk mengunjungi teman berlibur di sana	[Saya], [pergi], [malang], [mengunjungi], [teman], [berlibur], [sana]
3.	Stemming	Ardi sempat melihat kecelakaan yang menewaskan dua orang	[Andi], [sempat], [lihat], [kecelakaan], [yang], [tewas], [dua], [orang]
4.	Semantic / Similarity	Data 1: Apakah saya bisa memiliki buku itu? Data 2: Apakah saya bisa mendapatkan buku itu?	Nilai similairity: 80 – 90%

3.3.2 *Integration Testing*

Integration testing merupakan tahapan uji coba yang melibatkan dari kesatuan fungsi yang telah ditentukan. Dalam hal ini, *integration testing* akan dilakukan dari proses *tokenizing* hingga *semantic*. Perbedaan yang terjadi antara *unit testing* dan *integration testing* adalah pada *unit* testing dilakukan uji coba dari tiap-tiap fungsi yang telah dibuat, sedangkan pada *integration testing* merupakan uji coba yang melibatkan semua fungsi yang terkait satu sama lain, sehingga semua fungsi tersebut dilakukan dalam 1 (satu) kali proses uji coba. Untuk desain *integration testing* yang dilakukan dapat dilihat pada tabel berikut,

Tabel 3.4 Desain *Integration Testing*

No	Nama Fungsi	Inputan	Hasil yang diharapkan
1.	Menjawab pertanyaan	Data calon pelanggan beserta data pertanyaan yang diajukan, contoh: Data customer: M Hanif Data pertanyaan: Kapankah keberangkatan haji tahun 2017?	Data calon pelanggan dan data pertanyaan dapat tersimpan kedalam basis data. Selain itu, aplikasi dapat memberikan jawaban sesuai dengan pertanyaan yang diajukan.
2.	Maintenance Pertanyaan (tambah pertanyaan)	Data pertanyaan beserta jawaban dari pertanyaan tersebut, contoh: Data pertanyaan: Data jawaban:	Data pertanyaan dan jawaban dapat tersimpan kedalam basis data.
3.	Maintenance Pertanyaan (ubah pertanyaan)	Data pertanyaan beserta jawaban dari pertanyaan tersebut, contoh: Data pertanyaan: Data jawaban:	Data pertanyaan dan jawaban yang telah tersimpan dalam basis data dapat diubah dengan data pertanyaan dan jawaban yang baru
4.	Maintenance Pertanyaan (hapus pertanyaan)	ID pertanyaan yang ingin dihapus	Data pertanyaan dan jawaban yang telah tersimpan dalam basis data dapat dihapus, selama tidak memiliki relasi dengan basis data dengan pertanyaan yang diajukan oleh calon pelanggan. Apabila data tersebut memiliki relasi dengan tabel yang lain, maka akan menampilkan pesan *error* beserta penjelasannya *error*-nya.

3.3.3 *System Testing*

System testing merupakan tahapan pengujian sistem secara keseluruhan dengan data riil. Data yang digunakan merupakan data Q&A yang dimiliki oleh perusahaan saat ini dengan jumlah ± 45 pertanyaan dan jawaban. Sehingga dapat menggambarkan skenario dari penerapan sistem itu nantinya.

3.3.4 Uji Akurasi Aplikasi

Merupakan tahap pengujian akhir, dimana hasil dari sesi konsultasi menggunakan aplikasi *automated customer service* akan dicocokkan dengan sesi konsultasi manual. Kemudian hasil dari sesi konsultasi tersebut akan dihitung menggunakan formula sebagai berikut,

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \times 100\%$$

$$Error\ Rate = (1 - Accuracy) \times 100\%$$

Sumber : (Hamilton, 2012)

Keterangan:

TP = True Positive (hasil manual dan hasil aplikasi benar)

TN = True Negative (hasil manual salah, hasil aplikasi benar)

FP = False Positive (hasil manual benar, hasil aplikasi salah)

FN = False Negative (hasil manual dan aplikasi salah)

3.3.5 Uji Perbandingan

Pada pengujian ini akan dilakukan perbandingan antara jumlah antrean setelah menggunakan aplikasi dan jumlah antrean tanpa menggunakan aplikasi. Dimana nantinya akan digunakan untuk menilai apakah dengan adanya aplikasi *automated customer service* dapat mengurangi jumlah antrean yang ada perusahaan.