BAB 2
LANDASAN TEORI

2.1 Customer Relationship Management

Pelanggan merupakan aset yang sangat penting karena tidak ada satupun lembaga atau perusahaan yang mampu bertahan ketika ditinggal oleh pelanggannya [9]. Oleh karena itu, komunikasi dengan pelanggan harus dilakukan seefektif mungkin agar perusahaan dapat terus menjalin kerjasama yang baik dengan pelanggan. *Customer relationship management* secara keseluruhan mencakup praktik, strategi, dan teknologi yang digunakan sebuah perusahaan untuk mengelola dan menganalisis data terkait interaksi pelanggan. Tujuan utama dalam penggunaan sistem CRM adalah meningkatkan hubungan dengan pelanggan sehingga dapat turut serta meningkatkan penjualan. Komponen-komponen yang terdapat dalam CRM dapat dikelompokkan ke dalam tiga bagian, yaitu :

a Manajemen

Komponen pertama, yaitu manajemen terdiri dari pihak-pihak yang terlibat dalam kegiatan interaksi dengan pelanggan, seperti *customer service officer* (CSO), tim *marketing*, dan tim *sales*. Selain itu, manajemen juga terkait dengan strategi perusahaan dalam menjalankan manajemen yang baik untuk mengelola hubungan dengan pelanggan.

b Hubungan

Hubungan atau *relationship* adalah tentang bagaimana perusahaan memberikan layanan dan solusi terbaik untuk setiap permasalahan dan kebutuhan pelanggan.

c Pelanggan

Komponen terakhir adalah pelanggan. Pelanggan merupakan bagian dari proses bisnis yang menjadi alasan mengapa perlu adanya CRM. Pelanggan secara keseluruhan meliputi pelanggan tetap dan pelanggan potensial atau calon pelanggan. Perlu adanya strategi untuk bisa mendapatkan pelanggan baru dan tetap mempertahankan pelanggan yang sudah ada.

Ketiga komponen utama dalam CRM tentunya akan sangat sulit terwujud tanpa adanya faktor-faktor pendukung yang dapat membantu dalam proses

pelaksanaannya. Salah satu faktor pendukung tersebut adalah sistem CRM yang berupa perangkat lunak. Sistem CRM mengkonsolidasikan informasi dan dokumen terkait pelanggan ke dalam sebuah pangkalan data (*database*) sehingga perusahaan dapat dengan mudah mengakses dan mengelolanya. Seiring berjalannya waktu, sistem CRM terus mengalami perkembangan sesuai dengan kebutuhan perusahaan. Berikut ini merupakan beberapa komopnen yang biasanya terdapat dalam sebuah sistem CRM :

a *Marketing*
Komponen pertama dalam sistem CRM, yaitu *marketing* yang dapat membantu dalam melakukan otomasi terhadap kegiatan berulang seperti pengiriman *email* untuk pelanggan maupun calon pelanggan.

b *Geolocation, location based services*
Pada sebuah sistem CRM juga terdapat fitur yang terintegrasi dengan GPS (*Global Positioning System*) untuk melacak kegiatan *salesman* atau melakukan pemetaan pelanggan.

c *Analytics & Report*
Analytics merupakan fitur yang dapat menampilkan data-data terkait pelanggan untuk dianalisis sehingga dapat meningkatkan kepuasan pelanggan. Fitur *analytics* ini juga tidak dapat dipisahkan dengan fitur *report* yang dapat memberikan laporan-laporan terkait kunjungan atau interaksi yang telah dilakukan dengan pelanggan.

d *Human Resources Management*
Sistem CRM juga dapat membantu mengelola informasi karyawan, seperti informasi kontak, ulasan kinerja, dan tunjangan dalam perusahaan. Hal ini memungkinkan departemen SDM untuk lebih efektif mengelola tenaga kerja internal.

Implementasi CRM dapat dikatakan berhasil jika sistem CRM mampu membantu perusahaan dalam meningkatkan keuntungan serta memberikan nilai lebih kepada pelanggan dengan harga yang kompetitif [10]. Penggunaan sistem CRM oleh perusahaan dalam beberapa waktu belakangan terus meningkat. Beberapa faktor yang mendorong pengadopsian sistem CRM, antara lain:

a Mengurangi penggunaan biaya terutama untuk membeli

b Peningkatan daya komputasi komputer

c Mengurangi biaya penyimpanan data

d Keandalan alat yang semakin canggih dapat digunakan untuk analisis data, penambangan data atau untuk visualisasi data. Oleh karenanya, biaya interaksi tersebut jauh lebih rendah daripada penggunaan cara tradisional.

e Peningkatan kesadaran akan pentingnya perilaku pelanggan saat ini dan pentingnya memberikan makna kepada pelanggan,

2.2 Firebase

Situs web dan aplikasi menggunakan layanan web tradisional untuk bertukar data dengan *server* sesuai dengan skema komunikasi *klien*-server [11]. Pendekatan ini secara umum membutuhkan banyak waktu untuk merancang layanan web, mengimplementasikan dan mengujinya, lalu mengimplementasikan dan menguji klien. Hal ini tentu meningkatkan waktu peluncuran produk dan biaya pengembangan aplikasi. Dalam hal infrastruktur, layanan web memerlukan *server* yang selalu aktif dengan alamat IP publik, yang dapat berupa komputer khusus atau mesin virtual di *cloud*. Saat ini, penggunaan layanan web di *cloud* ini biasanya lebih disukai karena keandalannya yang lebih tinggi dan biaya keseluruhan yang lebih rendah.

Firebase adalah suatu layanan berbasis *cloud* dari Google untuk memberikan kemudahan bahkan mempermudah para *developer* aplikasi dalam mengembangkan aplikasinya [12]. Firebase alias BaaS (*Backend as a Service*) merupakan solusi yang ditawarkan oleh Google untuk mempercepat pekerjaan *developer*. Dengan menggunakan Firebase, *apps developer* bisa fokus dalam mengembangkan aplikasi tanpa memberikan *effort* yang besar untuk urusan *backend*. Layanan Firebase terdiri dari dua jenis, yaitu Spark (layanan gratis) dan Blaze (layanan berbayar). Beragam fitur atau produk yang ditawarkan oleh Firebase untuk membantu pengembangan perangkat lunak, antara lain :

a Firebase Analytics

Fitur *analytics* adalah salah satu fitur pada Firebase yang digunakan sebagai koleksi data dan *reporting* untuk aplikasi Android maupun iOS. Fitur ini mempunyai kelebihan yang memungkinkan pengembang untuk bisa membuat segmentasi *user* berdasarkan *user attribute*. *User attribute* adalah

suatu parameter yang bisa digunakan sebagai *filter* yang bertujuan untuk *reporting* dan notifikasi.

b Firebase Cloud Messaging and Notifications

FCM (Firebase Cloud Messaging) menyediakan kemampuan untuk mengirimkan dan menerima pesan serta notifikasi ke setiap perangkat yang sudah terinstall aplikasi. Pengembang juga dapat menargetkan pesan ke perangkat yang telah berlangganan pada topik tertentu. Selain itu, bisa juga menargetkan hanya ke satu perangkat untuk mendapatkan informasi data yang terperinci. Biasanya hal ini dilakukan untuk proses pengujian.

c Firebase Authentication

Firebase Authentication adalah salah satu layanan *back-end* yang bertujuan untuk memudahkan pembangunan sistem autentikasi yang aman, sambil meningkatkan pengalaman *sign-in* dan *onboarding* bagi *end-user*. Produk Firebase ini mendukung penggunaan akun *email* dan *password*, telepon, *login* media sosial seperti Google, Twitter, Facebook, dan GitHub untuk autentikasi pengguna.

d Firebase Realtime Database

Firebase Realtime Database adalah *database* yang di-host melalui *cloud*. Data disimpan dan dieksekusi dalam bentuk JSON dan disinkronkan secara *realtime* ke setiap user yang terkoneksi. Hal ini berfungsi memudahkan pengembang dalam mengelola suatu *database* dengan skala yang cukup besar. Kemampuan lain dari Firebase Realtime Database adalah tetap responsif bahkan saat *offline* karena SDK Firebase Realtime Database menyimpan data langsung ke *disk device* atau memori lokal.

e Firebase Cloud Firestore

Cloud Firestore adalah *database* yang bersifat fleksibel dan terukur untuk pengembangan perangkat seperti seluler, web, dan *server* di Firebase dan Google Cloud Platform. Sama halnya dengan Firebase Realtime Database, Cloud Firestore membuat data dapat terkoneksi di aplikasi *user* melalui *listener realtime* dan menawarkan layanan secara *offline* untuk aplikasi seluler dan web. Dengan begitu, pengembang dapat membuat aplikasi yang *powerfull*, responsif, dan mampu bekerja tanpa bergantung pada latensi koneksi internet.

f Firebase Hosting

Firebase Hosting merupakan suatu layanan *hosting* konten web. Hanya dengan satu instruksi, fitur ini dapat mengimplementasikan aplikasi web serta menyajikan konten statis maupun dinamis ke CDN (jaringan penayangan konten) global dengan cepat. Kegunaan dari Firebase Hosting itu sendiri, yaitu mampu menayangkan konten melalui koneksi yang begitu aman, mengirimkan konten secara cepat, dan mendukung semua jenis konten untuk di *hosting*, mulai dari *file* HTML dan CSS hingga API atau layanan mikro Express.js.

g Firebase Test Lab

Test Lab menyediakan infrastruktur berbasis *cloud* untuk menguji aplikasi Android. Dengan satu operasi, pengembang dapat memulai pengujian aplikasi mereka di berbagai perangkat dan konfigurasi perangkat [13]. Berbagai hasil pengujian seperti tangkapan layar, video, dan *log* tersedia di konsol Firebase. Meskipun *developer* belum menulis kode pengujian apa pun untuk aplikasinya, Test Lab dapat menjalankan aplikasi secara otomatis dan mencari kerusakan.

h Firebase Crash Reporting

Tidak ada aplikasi yang bebas *bug* bahkan setelah mengujinya selama ratusan kali [14]. Beberapa *bug* dalam aplikasi menyebabkan desain yang buruk dan beberapa menyebabkan pengalaman pengguna yang buruk. Investasi dalam pengujian tidak selamanya menjamin aplikasi bebas dari kegagalan. Oleh karena itu, sangat penting untuk dapat mengetahui detail dari kegagalan yang terjadi dalam aplikasi secara cepat. Firebase Crash Reporting merupakan solusi yang dapat membantu untuk mendapatkan laporan kegagalan berdasarkan tingkat keparahan dan juga memberikan detail informasi tentang kegagalan sistem yang terjadi.

2.3 Google Maps API

API atau *application programming interface* merupakan fungsi pemrograman yang disediakan oleh aplikasi atau layanan agar layanan tersebut dapat terintegrasi dengan aplikasi yang kita buat [15]. Berdasarkan rumusan tersebut, maka Google maps API merupakan sebuah fungsi pemrograman terintegrasi yang memungkinkan pengembang perangkat lunak untuk mengakses

data dan fungsi terkait Google Maps. Dengan menggunakan Google Maps API, pengembang dapat mengkustomisasi menghadirkan peta interaktif kepada pengguna [16]. Hal ini tentu dapat meningkatkan pengalaman pengguna dalam menggunakan aplikasi. Terdapat tiga buah fitur yang umum digunakan dalam Google Maps API, yaitu :

a Maps API

Maps API ini biasa disebut juga sebagai Dynamic Maps merupakan visualisasi dari tampilan peta digital milik Google. Dynamic Maps ini biasanya ditampilkan pada halaman *website* untuk bisa menghasilkan tampilan optimal. Namun, tidak menutup kemungkinan untuk digunakan pada perangkat *mobile*. Kelebihan dari Dynamic Maps diantaranya tampilan interaktif, dapat memiringkan, memutar, memperbesar / memperkecil, dan menggeser dengan kontrol penuh. Kemudian, Dynamic Maps dapat dilakukan kostumisasi dengan memasukan beberapa *layer* lain di atas peta, seperti gambar, batas wilayah tertentu, *heatmap*, serta dapat merubah *marker* dan menampilkan *info window* di atasnya.

b Routes API

Routes API memiliki 3 komponen pendukung di dalamnya, yaitu Directions API, Distance matrix API, dan Roads API. Directions API dapat mengembalikan informasi rute perjalanan, waktu tempuh, dan jarak tempuh dari satu lokasi ke lokasi yang lain. Berbeda dengan Directions API, Distance Matrix API tidak memberikan informasi rute perjalanan, namun hanya jarak dan waktu tempuh saja yang secara umum sering digunakan oleh pengguna. Terakhir, Roads API memiliki fitur yang dapat mengidentifikasi jalan yang dilalui kendaraan dan memberikan metadata tambahan tentang jalan tersebut, seperti batas kecepatan.

c Places API

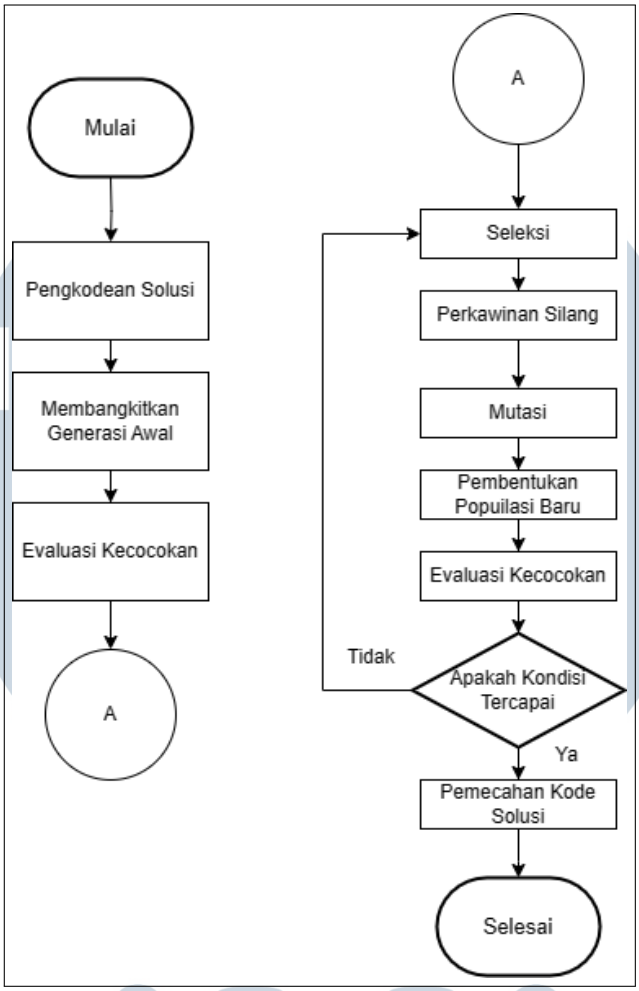
Places API ini merupakan salah satu komponen Google Maps Platform API yang menyediakan lebih dari 200 juta data lokasi di seluruh dunia yang dimiliki oleh Google. Places API ini dapat digunakan untuk pencarian alamat, melihat informasi detail tentang suatu tempat, serta foto-foto berbagai lokasi yang tersimpan dalam Google Maps.

2.4 Algoritma Genetika

Terdapat dua jenis algoritma optimisasi yang biasa digunakan dalam proses pemecahan masalah TSP, yaitu *heuristic* dan *metaheuristic*. *Heuristic* adalah teknik yang ditujukan untuk memecahkan masalah lebih cepat disaat penggunaan teknik tradisional seperti *dynamic programming* dinilai terlalu lambat [17]. *Metaheuristic*, di sisi lain, adalah teknik tingkat tinggi berbasis *heuristic* yang mencari, menghasilkan, atau memilih *heuristic* yang dapat memberikan solusi yang cukup baik untuk masalah optimisasi. Walaupun algoritma dengan pendekatan *heuristic* dapat memberikan solusi yang memadai dalam beberapa kondisi, namun belum dapat menjamin solusi yang optimal [18]. Oleh karena itu, dalam penelitian ini, digunakan algoritma *metaheuristic*, yaitu algoritma genetika, yang memiliki kemampuan untuk mencari solusi secara efisien dan mendekati solusi optimal.

Algoritma genetika (GA) merupakan algoritma optimisasi yang terinspirasi dari proses seleksi alam. Algoritma ini berbasis pencarian populasi, yang menggunakan konsep *survival of fittest* [19]. Prinsip dasar di balik algoritma genetika adalah menghasilkan dan mempertahankan populasi individu yang diwakili oleh kromosom. Setiap kromosom terdiri atas sekumpulan gen yang merepresentasikan solusi dari masalah. Kromosom-kromosom tersebut akan melewati proses evolusi melalui tahap seleksi, reproduksi, dan mutasi. Proses evolusi ini berlanjut hingga mencapai titik di mana solusi yang optimal atau mendekati optimal dapat ditemukan [20]. Tahap-tahap proses yang dilakukan dalam algoritma genetika meliputi inisialisasi populasi awal, evaluasi individu, seleksi, rekombinasi (crossover), dan mutasi.

Dalam konteks penelitian ini, algoritma genetika digunakan sebagai pendekatan *metaheuristic* untuk memecahkan masalah TSP. Pendekatan ini memungkinkan pencarian solusi yang mendekati optimal dengan waktu komputasi yang lebih efisien dibandingkan dengan metode eksak. Algoritma genetika telah banyak diterapkan dalam penyelesaian TSP dan menghasilkan solusi yang baik. Namun, penelitian-penelitian sebelumnya masih terus melakukan pengembangan dalam hal pengoptimalan kombinasi operator genetik, penggunaan strategi seleksi, dan pendekatan pencarian lokal untuk meningkatkan kualitas solusi yang ditemukan. Gambaran umum aliran proses algoritma genetika dapat dilihat pada Gambar 2.1.



Gambar 2.1. Tahap-tahap Algoritma Genetika

Sumber: [21]

Berikut ini merupakan penjabaran dari proses yang terjadi dalam algoritma genetika :

a Pengkodean Solusi & Membangkitkan Generasi Awal

Pada awal algoritma genetika, dilakukan pengkodean solusi yang mengubah setiap solusi dalam domain masalah menjadi bentuk kromosom, yang merepresentasikan informasi genetik solusi. Setelah pengkodean, dilakukan pembentukan populasi awal atau generasi P(0) dengan memilih solusi secara acak dari ruang pencarian yang mungkin. Populasi awal yang acak ini bertujuan untuk memperkenalkan variasi dan mengeksplorasi berbagai solusi potensial dalam ruang pencarian. Langkah pengkodean solusi dan pembentukan populasi awal adalah langkah krusial dalam memulai algoritma

genetika dan menentukan dasar untuk iterasi selanjutnya dalam mencari solusi optimal.

b Evaluasi Kecocokan (*Fitness*)

Sebelum melakukan operasi seleksi, kecocokan setiap individu kromosom dievaluasi menggunakan fungsi *fitness* yang tidak lain adalah fungsi objektif itu sendiri. Nilai kecocokan untuk setiap kromosom adalah nilai atau jarak ke berbagai *node*. Formula dari penghitungan nilai kecocokan ditentukan berdasarkan masalah yang ingin diselesaikan. Salah satu metode yang dapat digunakan dalam proses perhitungan nilai kecocokan adalah *weighted sum*. *Weighted sum* adalah operasi matematika yang menggabungkan beberapa nilai dengan menetapkan bobot pada setiap nilai dan kemudian menghitung jumlah mereka. Dalam konteks fungsi perhitungan nilai *fitness* untuk algoritma genetika, *weighted sum* digunakan untuk menggabungkan kontribusi dari komponen atau faktor yang berbeda. Berikut ini adalah contoh rumus perhitungan nilai kecocokan menggunakan *weighted sum* pada Persamaan 2.1.

$$\textit{Nilai Kecocokan} = w_1 \cdot A + w_2 \cdot B + w_3 \cdot C \tag{2.1}$$

Keterangan:

A, B, dan C: Faktor-faktor yang digunakan dalam perhitungan algoritma.

w_1, w_2, w_3: Nilai bobot yang dapat disesuaikan berdasarkan tingkat kepentingan.

Dalam Persamaan 2.1, $w_1, w_2, \ldots, w_n$ adalah bobot yang ditetapkan untuk setiap komponen, sedangkan $A, B, \ldots, N$ adalah nilai-nilai yang berkaitan dengan komponen-komponen tersebut. Rumus ini memungkinkan penyesuaian prioritas atau pentingnya masing-masing komponen dalam penentuan nilai kecocokan.

Pada fungsi objektif yang ingin dimaksimalkan, nilai kecocokan yang lebih besar mengindikasikan solusi yang lebih baik. Sebaliknya, pada fungsi objektif yang ingin diminimalkan, nilai kecocokan yang lebih kecil menunjukkan solusi yang lebih baik. Oleh karena itu, indikator suatu nilai kecocokan dianggap baik atau tidak tergantung pada tujuan optimisasi.

c Seleksi

Proses seleksi dalam algoritma genetika adalah langkah penting untuk memilih individu-individu yang akan menjadi orangtua untuk reproduksi dalam populasi. Tujuan dari seleksi adalah untuk meningkatkan kemungkinan individu-individu yang memiliki kualitas tinggi ditransmisikan ke generasi berikutnya. Berikut ini adalah penjelasan mengenai tiga metode seleksi yang umum digunakan dalam algoritma genetika, yaitu *roulette wheel*, *rank selection*, dan *tournament selection*:

- *Roulette Wheel Selection*
  Metode seleksi *roulette wheel* menggunakan pendekatan proporsional untuk memilih individu-individu. Setiap individu memiliki probabilitas seleksi yang proporsional terhadap nilai *fitness*-nya. Semakin tinggi *fitness* individu, semakin besar probabilitasnya untuk dipilih sebagai orangtua.
- *Rank Selection*
  Metode *rank selection* mengevaluasi individu-individu berdasarkan peringkat mereka berdasarkan *fitness*. Setiap individu diberi peringkat berdasarkan *fitness*-nya, kemudian probabilitas seleksi ditentukan berdasarkan peringkat mereka. Individu dengan peringkat lebih tinggi memiliki probabilitas seleksi yang lebih besar. Penggunaan metode dapat mengurangi pengaruh dominasi individu dengan *fitness* tertinggi.
- *Tournament Selection*
  Metode *tournament selection* melibatkan seleksi acak beberapa individu dari populasi dan memilih individu terbaik dari grup tersebut. Jumlah peserta turnamen ditentukan sebelumnya. Kemudian, individu dengan *fitness* terbaik dari grup tersebut dipilih sebagai orangtua. Kelebihan dari metode ini adalah memungkinkan individu dengan *fitness* rendah memiliki peluang untuk menjadi orangtua jika mereka berada dalam grup yang kuat dan tidak memerlukan perhitungan probabilitas, sehingga lebih efisien secara komputasi.

d Perkawinan Silang

Perkawinan silang adalah proses yang digunakan untuk menghasilkan keturunan dengan menggabungkan informasi genetik dari dua kromosom orang tua. Tujuan dari perkawinan silang adalah memperkenalkan variasi genetik baru dan mengkombinasikan sifat-sifat yang baik dari orangtua. Beberapa jenis metode yang dapat digunakan dalam perkawinan silang, yaitu:

- *One Point Crossover*

  One Point Crossover melibatkan pemilihan satu titik pemotongan (*crossover point*) yang sama pada kedua orangtua. Seluruh gen setelah titik pemotongan dipertukarkan antara kedua orangtua untuk menghasilkan keturunan. Metode ini adalah metode yang paling sederhana dan mudah diimplementasikan dalam algoritma genetika.

- *Multi Point Crossover*

  Pada proses perkawinan *multi point* terdapat beberapa titik pemotongan (*crossover points*) yang sama pada kedua orangtua. Gen-gen di antara titik-titik pemotongan tersebut akan dipertukarkan antara kedua orangtua untuk menghasilkan keturunan baru.

- *Uniform Crossover*

  Uniform Crossover melibatkan pertukaran gen secara acak antara kedua orangtua dengan menggunakan masker (*mask*) yang dihasilkan. Setiap gen dalam keturunan memiliki probabilitas tertentu untuk berasal dari orangtua tertentu. Penggunaan metode *uniform crossover* docok untuk masalah dengan keterkaitan lokal antar gen yang tinggi.

- *Davis' Order Crossover (OX1)*

  Davis' Order Crossover, juga dikenal sebagai OX1, adalah metode crossover khusus untuk masalah permutasi (urutan). Metode ini mempertahankan urutan relatif gen antara kedua orangtua dalam keturunan. Sebuah segmen gen dipilih secara acak dari satu orangtua, dan kemudian urutan gen dari segmen tersebut dipertahankan pada posisi yang sesuai dalam keturunan. Gen-gen yang tersisa diisi dalam urutan yang tersisa, menghindari duplikasi gen.

e Mutasi

Proses mutasi dalam algoritma genetika digunakan untuk menjaga keragaman dalam populasi dengan mengubah nilai gen yang dipilih secara acak. Tujuan lain dari mutasi adalah untuk mencegah konvergensi prematur dan menjelajahi ruang pencarian solusi yang lebih luas. Berikut adalah penjelasan tentang proses mutasi dalam algoritma genetika beserta variasi implementasinya:

- *Bit Flip Mutation*

  Bit Flip Mutation adalah variasi mutasi paling sederhana, yang biasanya

digunakan dalam kromosom dengan representasi biner. Dalam mutasi ini, satu atau lebih bit secara acak dipilih dan diubah nilainya, yaitu dari 0 menjadi 1 atau sebaliknya.

- *Swap Mutation*
  Swap Mutation melibatkan pertukaran posisi dua gen dalam kromosom. Dalam mutasi ini, dua posisi gen dipilih secara acak, dan nilai gen pada kedua posisi tersebut ditukar.
- *Inversion Mutation*
  Pada *Inversion Mutation*, sebuah segmen gen dalam kromosom dipilih secara acak, kemudian urutan gen pada segmen tersebut dibalik (di-invers). Dengan demikian, urutan relatif gen dalam segmen tersebut akan berubah, sementara gen-gennya tetap sama.

f Pembentukan Populasi Baru
Setiap nilai kecocokan (*fitness*) keturunan baru yang diperoleh setelah perkawinan silang dan mutasi ditentukan pada tahap ini. Pada populasi baru, kromosom lama dari populasi saat ini yang memiliki nilai kecocokan lebih tinggi akan tetap dipertahankan.

g Kondisi Akhir
Seleksi, perkawinan silang, mutasi dan pembentukan populasi dilanjutkan sampai beberapa generasi. Kromosom dengan nilai terbaik pada setiap generasi diambil untuk mewakili solusi optimal.

2.5 *User Acceptance Test*

User acceptance testing (UAT) adalah tahap pengujian terakhir dan terpenting dari empat tahapan pengujian perangkat lunak yang umum dilakukan [22]. Dalam tahapan ini, pengujian sistem dilakukan untuk menentukan apakah sistem telah memenuhi kebutuhan pengguna dan dapat mendukung semua skenario bisnis dan pengguna. UAT dilakukan oleh *client* dan *end-user*. Beberapa kegunaan dari proses pengujian ini, yaitu memastikan kriteria dan spesifikasi yang sebelumnya telah ditentukan dan disepakati dalam kontrak, memeriksa *software* telah dibuat sesuai dengan peraturan atau tidak melanggar ketentuan hukum, dan memastikan alur kerja perangkat lunak ketika masuk tahap produksi. Tiga cakupan proses yang terdapat pelaksanaan UAT, yaitu:

a Perencanaan

Perencanaan untuk pembuatan UAT perlu dilakukan dari awal proses karena ada keputusan dan persiapan yang harus dilakukan selama proses berlangsung. Apabila perencanaan baru dilakukan di akhir proyek, kemungkinan besar akan timbul masalah yang menyebabkan penundaan sehingga target penyelesaian proyek akan mundur dari waktu yang telah ditentukan. Perencanaan UAT bisa dilakukan dengan menyusun *test plan* yang memiliki komponen umum seperti tanggal, kondisi lingkungan, pelaku, peran dan tanggung jawab, hasil dan proses analisis, serta *entry-exit criteria*.

b Persiapan

Dalam melangsungkan pengujian, dibutuhkan data pengujian yang pembuatannya rumit dan membutuhkan sumber daya yang besar. Pembuatan data dapat dilakukan dengan metode dimasukkan langsung oleh pengguna atau menggunakan data internal yang terdapat di database. Apabila data dimasukkan oleh pengguna, maka dapat didefinisikan dan didokumentasikan secara tepat karena pengguna melakukan *input* data sesuai dengan persyaratan yang akan diuji. Selain menyiapkan data, hal yang juga perlu dipersiapkan yaitu memperhatikan apakah sistem dapat digunakan di lingkungan bisnis sehari-hari. Biasanya diperlukan komputer uji dan lingkungan yang mampu melakukan simulasi bisnis sesungguhnya.

c Pengelolaan dan Eksekusi

Pengguna sistem sebagai penguji, bertanggung jawab untuk mengidentifikasi kasus yang akan di tes, membuat data tes, dan menjalankan UAT. Pada akhir proses pengujian, pengguna perlu menyimpulkan apakah pengujian berhasil sehingga persyaratan terpenuhi atau tidak. Apabila terjadi kesalahan dalam pengujian, maka perlu perbaikan yang dicatat dan dilacak.

2.6 *End User Computing Satisfaction* (EUCS) dan Skala Likert

End User Computing Satisfaction (EUCS) merupakan sebuah metode yang digunakan untuk mengukur tingkat kepuasan pengguna terhadap suatu sistem aplikasi dengan membandingkan antara harapan dan pengalaman pengguna saat menggunakan aplikasi [23]. Evaluasi dengan menggunakan model EUCS lebih menekankan kepuasan (*satisfaction*) pengguna terhadap 5 aspek, yaitu isi (konten), keakuratan, format, waktu dan kemudahan penggunaan dari sistem. Berikut uraian

dari kelima aspek pengukuran yang terdapat di dalam EUCS:

a Konten

Mengukur apakah sistem menghasilkan informasi yang sesuai dengan kebutuhan pengguna. Semakin lengkap modul dan sistem informasi maka tingkat kepuasan dari pengguna akan semakin tinggi.

b Akurasi

Mengukur kepuasan pengguna dari sisi keakuratan data ketika sistem menerima *input* kemudian diolah menjadi informasi. Keakuratan sistem diukur dengan melihat seberapa sering sistem menghasilkan *output* yang salah ketika mengolah *input* dari pengguna, selain itu dapat dilihat pula seberapa sering terjadi *error* atau kesalahan dalam proses pengolahan data.

c Format

Mengukur kepuasan pengguna dari sisi tampilan dan estetika antarmuka sistem, apakah antarmuka dari sistem itu menarik serta memudahkan pengguna ketika menggunakan sistem sehingga secara tidak langsung dapat berpengaruh terhadap tingkat efektifitas dari pengguna.

d Kemudahan Pengguna

Mengukur kepuasan pengguna dari sisi kemudahan pengguna dalam menggunakan sistem seperti proses memasukan data, mengolah data dan mencari informasi yang dibutuhkan.

e Ketepatan Waktu

Mengukur kepuasan pengguna dari sisi ketepatan waktu sistem dalam meyajikan atau menyediakan data dan informasi yang dibutuhkan oleh pengguna. Sistem yang tepat waktu dapat dikategorikan sebagai sistem *real-time*, berarti setiap permintaan atau *input* yang dilakukan oleh pengguna akan langsung diproses dan *output* akan ditampilkan secara cepat tanpa harus menunggu lama.

Skala yang digunakan untuk mengukur tingkat kepuasan pengguna dalam EUCS adalah skala likert. Skala likert digunakan untuk mengukur pendapat, sikap, dan persepsi yang sangat berguna untuk menghitung semua indikator terhadap sebuah objek penelitian [24]. Karakteristik skala likert, meliputi:

a Jawaban terkait. Pertanyaan wajib dikaitkan dengan jawaban kalimat, terlepas dari apakah hubungan antara butir pertanyaan dan juga kalimat terbukti.

b Jenis skala. Pertanyaan wajib memiliki dua posisi ekstrem dan juga opsi jawaban perantara.

c Jumlah opsi jawaban. Penting untuk disebutkan bahwa walaupun skala Likert yang paling umum memiliki lima *item*, penggunaan lebih dari lima *item* membantu untuk menghasilkan ketepatan jauh lebih tinggi dalam hasil.

d Meningkatkan reliabilitas skala. Peneliti sering kali memaksimalkan ujung skala dengan membuat skala tujuh poin agar mencapai batas atas reliabilitas skala.

e Menggunakan skala lebar. Sebagai aturan umum, skala likert dan juga skala lainnya merekomendasikan bahwa lebih baik menggunakan skala selebar mungkin. Seseorang selalu dapat menciutkan jawaban ke dalam kelompok yang ringkas, jika sesuai untuk pengukran dalam teknik analisis data yang dipergunakan.

f Kurangnya opsi netral. Dengan mempertimbangkan detil ini, skala terkadang dibatasi menjadi sejumlah kategori genap (biasanya empat) untuk menghilangkan kemungkinan "netral" pada skala survei "pilihan paksa (*forced choice*)".

g Variabel intrinsik. Catatan primer likert secara pasti menyatakan bahwa mungkin ada variabel penelitian inheren yang nilainya menandai umpan balik atau sikap responden, dan variabel yang paling banyak mendasari ini adalah tingkat interval.

Rumus yang digunakan untuk perhitungan skor tingkat kepuasan pengguna serta keterangan hasil pengukuran berdasarkan skala likert, dapat dilihat pada Persamaan 2.2 dan Tabel 2.1 berikut.

$$SkorTingkatKepuasan = \frac{TotalSkorYangDiperoleh}{TotalSkorTertinggi} * 100\% \qquad (2.2)$$

Tabel 2.1. Keterangan Hasil Pengukuran Tingkat Kepuasan Pengguna

No.	Skor Tingkat Kepuasan	Keterangan
1	0% - 34.99%	Sangat Tidak Puas
2	35% - 50.99%	Tidak Puas
3	51% - 65.99%	Netral
4	66% - 80.99%	Puas
5	81% - 100%	Sangat Puas

Sumber: [25]