BAB II
TEORI PENUNJANG

Bab ini membahas mengenai teori penunjang yang berkaitan pada perancangan pengerjaan Tugas Akhir, beserta *tools* yang digunakan untuk membuat Tugas Akhir. Diantaranya, Penjadwalan Kuliah, Algoritma Genetika, Bahasa Pemograman WEB, Pemograman Berorientasi Objek, UML, MySQL, dan XAMPP.

2.1 Penjadwalan Kuliah

Penjadwalan Kuliah merupakan pengaturan penempatan waktu dan ruangan berdasarkan jumlah kuliah dan akademik sejenis, dengan memperhatikan sejumlah aturan yang berhubungan dengan kapasitas dan lokasi dari ruangan yang tersedia, waktu bebas yang diperlukan dan sejumlah aturan lain yang berkaitan dengan toleransi untuk dosen, dan hubungan antar mata kuliah khusus[13].

Menurut Kamus Besar Bahasa Indonesia, kuliah adalah pelajaran yg diberikan di perguruan tinggi. Terdapat berbagai istilah yang dikenal pada kegiatan kuliah (perkuliahan) di perguruan tinggi, diantaranya mahasiswa, dosen, mata kuliah, SKS, dan semester.

1. Mahasiswa adalah orang yang belajar di perguruan tinggi dan merupakan pelaku kegiatan kuliah di perguruan tinggi yang menerima pelajaran dari dosen.
2. Dosen adalah tenaga pengajar pada perguruan tinggi dan merupakan pelaku kegiatan kuliah di perguruan tinggi yang memberi pelajaran kepada mahasiswa.
3. Mata kuliah adalah satuan pelajaran di tingkat perguruan tinggi yang diajarkan oleh dosen kepada mahasiswa.
4. SKS (Satuan Kredit Semester), adalah satuan yang digunakan dalam Sistem Kredit Semester. Sistem kredit semester adalah suatu sistem penyelenggaraan perkuliahan di perguruan tinngi yang menyatakan beban studi mahasiswa, beban kerja dosen dan beban penyelenggaraan perkuliahan dengan

menggunakan Satuan Kredit Semester (SKS). Besarnya satu satuan kredit semester (1 SKS) dalam kegiatan perkuliahan pada berbagai perguruan tinggi terkadang berbeda namun ada juga yang sama. Di Universitas Pendidikan Indonesia, nilai 1 SKS dalam kegiatan perkuliahan adalah setara dengan alokasi waktu lima puluh (50) menit kegiatan tatap muka terjadwal antara dosen dengan mahasiswa.

Semester adalah satuan terkecil untuk lamanya masa program pendidikan. Jenjang pendidikan strata satu (S1) terdiri atas 8 semester. Satu tahun akademik (penyelenggaraan kegiatan perkuliahan dalam satu tahun) dibagi menjadi dua semester, yaitu semester ganjil dan semester genap.

2.2 Algoritma Genetika

Algoritma ini ditemukan di Universitas Michigan, Amerika Serikat oleh John Holland (1975) melalui sebuah penelitian dan dipopulerkan oleh salah satu muridnya, David Goldberg (1989). Dimana mendefenisikan algoritma genetika ini sebagai metode algoritma pencarian berdasarkan pada mekanisme seleksi alam dan genetik alam[2].

Algoritma genetika merupakan evaluasi atau perkembangan dunia komputer dalam bidang kecerdasan buatan (artificial intelligence). Kemunculan algoritma genetika ini terinspirasi oleh teori Darwin dan teori-teori dalam ilmu biologi, sehingga banyak istilah dan konsep biologi yang digunakan dalam algoritma genetika, karena sesuai dengan namanya, proses-proses yang terjadi dalam algoritma genetika sama dengan apa yang terjadi pada evaluasi biologi.

Dalam teori evolusi Darwin, suatu individu tercipta secara acak kemudian berkembang biak melalui proses reproduksi sehingga terbentuk sekumpulan individu sebagai populasi. Setiap individu dalam populasi mempunyai tingkat kebugaran yang berbeda-beda. Tingkat kebugaran ini menentukan seberapa kuat untuk tetap betahan hidup dalam populasinya. Sebagian individu tetap bertahan hidup dan sebagian lainnya mati.

Pada dasarnya, algoritma genetika merupakan metode pencarian yang didasarkan pada proses evolusi alamiah, yaitu terbentuknya populasi awal secara acak yang terdiri dari individu-individu dengan sifat yang bergantung pada gen-gen

dalam kromosomnya. Individu-individu melakukan proses reproduksi untuk melahirkan keturunan. Sifat keturunan dibentuk dari kombinasi sifat kedua induknya atau mewarisi sifat-sifat induknya[3].

2.2.1 Definisi Penting dalam Algoritma Genetika

Dalam algoritma genetika terdapat elemen-elemen yang digunakan sebagai representas sebuah variable, elemen tersebut saling berkaitan serta mempunyai peranan masing-masing , berikut adalah penjelesan elemen-elemen tersebut[3] :

1. *Allele*

Allele merupakan suatu nilai yang terdapat pada suatu gen. Satu gen mempunyai satu nilai yang disebut *allele*. *Allele* dapat direpresentasikan dalam notasi yang telah ditentukan (biner , float, integer, maupun karakter).

2. Gen

Gen merupakan nilai yang menyatakan satuan dasar yang membentuk suatu arti tertentu dalam satu kesatuan gen yang dinamakan kromosom. Gen mewakili satu atau lebih variabel sebagai solusi yang ingin dicapai. Gen dapat di representasikan dalam bentuk berikut :

String bit : 10011, 01101, 11101, dst.
Bilangan real : 65.65, -67.98, 592.88, dst.
Elemen permutasi : E2, E10, E6, dst.
Daftar aturan : R1, R2, R3, dst.
Elemen program : pemograman genetika.

3. Kromosom atau Individu

Kromosom atau individu merupakan gabungan dari kumpulan gen-gen yang membentuk nilai tertentu dan menyatakan solusi yang mungkin dari suatu permasalahan.

4. Populasi

Populasi merupakan sekumpulan individu yang akan diproses bersama dalam satu satuan siklus evolusi.

5. *Fitness*

Fitness merupakan suatu nilai yang menyatakan seberapa baik suatu individu yang didapatkan. Semakin besar nilai *fitness* semakin baik suatu individu untuk bertahan hidup.

6. Seleksi

Seleksi merupakan proses pemilihan dua individu terbaik atau memiliki nilai *fitness* yang tinggi yang akan dijadikan sebagai induk (*parent*) untuk digunakan dalam proses regenerasi. Seleksi juga berfungsi memberikan kesempatan reproduksi yang lebih besar bagi anggota populasi yang paling fit.

7. *Crossover*

Crossover merupakan proses pertukaran atau kawin silang gen-gen dari dua induk tertentu.

8. Mutasi

Mutasi merupakan proses pergantian salah satu gen yang terpilih dengan nilai tertentu.

9. Generasi

Generasi merupakan urutan iterasi dalam satu kesatuan proses genetika, dimana beberapa kromosom terbaik dari generasi sebelumnya bergabung dan membuat kromosom baru dengan solusi yang lebih baik.

10. *Offspring*

Offspring merupakan kromosom baru yang dihasilkan setelah melewati suatu generasi. Dalam hal ini offspring tersebut merupakan hasil kromosom yang dibangkitkan setelah melakukan proses seleksi.

2.2.2 Aplikasi Algoritma Genetika

Sejak pertama kali dirintis oleh John Holland pada tahun 1960-an, algoritma genetika telah dipelajari, diteliti dan diaplikasikan secara luas pada berbagai bidang. Algoritma banyak digunakan pada masalah praktis yang berfokus pada pencarian parameter-parameter optimal [11].

Keuntungan penggunaan algoritma genetika sangat jelas terlihat dari kemudahan implementasi dan kemampuannya untuk menentukan solusi seperti:

1. Ruang masalah sangat besar, kompleks, dan sulit dipahami.
2. Kurang atau bahkan tidak ada pengetahuan yang memandai untuk merepresentasikan masalah ke dalam ruang pencarian yang lebih sempit.
3. Tidak tersedianya analisis matematika yang memadai.
4. Ketika metode-metode konvensional sudah tidak mampu menyelesaikan masalah yang dihadapi.
5. Solusi yang diharapkan tidak harus paling optimal, tetapi cukup bisa diterima.
6. Terdapat batasan waktu, misalnya dalam real time systems atau system waktu nyata.

Algoritma genetika telah banyak diaplikasikan untuk penyelesaian masalah dan pemodelan dalam bidang teknologi, bisnis dan entertainment seperti:

1. Optimasi

Algoritma Genetika untuk optimasi numeric dan optimasi kombinatorial seperti Traveling Salesman Problem (TSP), perancangan *Intergrated Circuit* atau IC, job shop scheduling, optimasi video, dan suara.

2. Pemograman otomatis

Algoritma genetika telah digunakan untuk melakukan proses evolusi terhadap program komputer untuk merancang struktur komputasional, seperti *cellular automatic* dan *sorting networks*.

3. *Machine learning*

Algoritma genetika telah berhasil diaplikasikan untuk memprediksi struktur protein. Algoritma genetika juga berhasil diaplikasikan dalam perancangan neural networks (jaringan syaraf tiruan) untuk melakukan proses evolusi terhadap aturan-aturan pada *learning classifier systems* atau *symbolic prosuction systems*. Algoritma genetika juga digunakan untuk mengontrol robot.

4. Model Ekonomi

Algoritma genetika telah digunakan untuk memodelkan proses-proses inovasi dan pembangunan bidding strategies.

5. Model Sistem Imunisasi

Algoritma genetika telah berhasil digunakan untuk memodelkan berbagai aspek pada sistem imunisasi alamiah, termasuk somatic mulation selama kehidupan individu dan menentukan keluarga dengan gen ganda (multi - gen families) sepanjang waktu evolusi.

6. Model Ekologi

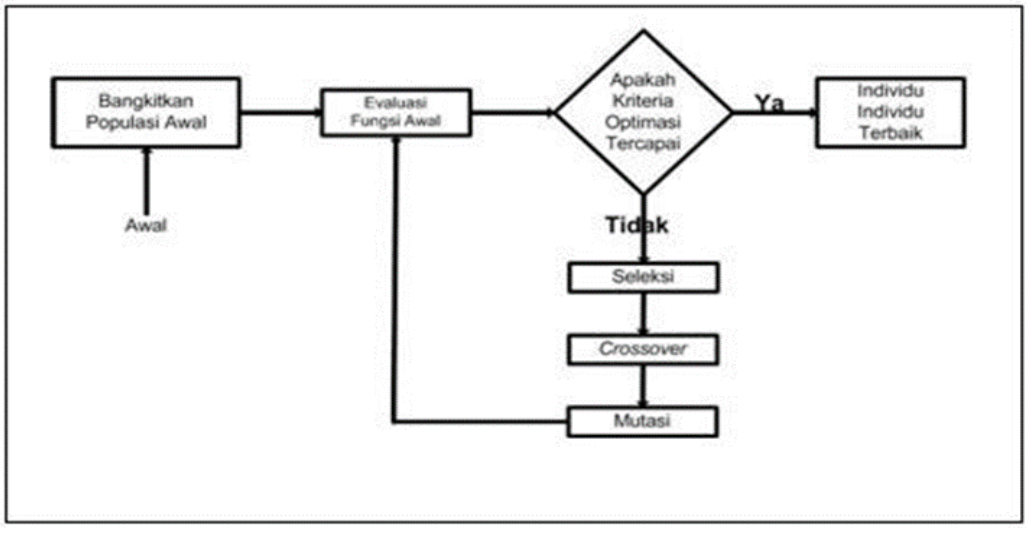
Algoritma genetika telah berhasil digunakan untuk memodelkan fenomena ekologis seperti *host-parasite co-evolutions*, simbiosis dan aliran sumber daya dalam ekologi.

7. Interaksi antara Evolusi dan Belajar

Algoritma genetika telah digunakan untuk mempelajari bagaimana proses belajar suatu individu bisa mempengaruhi proses evolusi suatu species dan sebaliknya.

2.2.3 Struktur Umum Algoritma Genetika

Algoritma genetika memberikan suatu pilihan bagi penentuan nilai parameter dengan meniru cara reproduksi genetika, pembentukan kromosom baru serta seleksi alami seperti yang terjadi pada makhluk hidup. Algoritma Genetika secara umum dapat diilustrasikan dalam diagram alir berikut ini[1]:



Gambar 2.1 Diagram Alir Algoritma Genetika

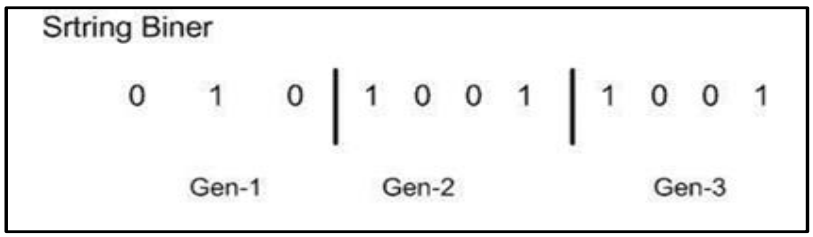
Pada algoritma ini, teknik pencarian dilakukan sekaligus atas sejumlah solusi yang mungkin dikenal dengan istilah populasi. Individu yang terdapat dalam satu populasi individu yang terdapat dalam satu populasi disebut dengan istilah **kromosom**, Charles L Karr (1999). Kromosom ini merupakan suatu solusi yang masih berbentuk simbol. Populasi awal dibangun secara acak, sedangkan populasinya merupakan hasil evolusi kromosomkromosom melalui iterasi yang disebut dengan istilah **generasi**. Pada setiap generasi kromosom akan melalui proses evaluasi dengan menggunakan alat ukur yang disebut **fungsi *fitness***. Nilai *fitness* dari suatu kromosom akan menunjukkan kualitas kromosom dalam populasi tersebut [2].

2.2.4 Komponen Utama Algoritma Genetika

Komponen-komponen utama algoritma genetika saling berketergantungan dan berkaitan satu sama lain, berikut penjelasan dari komponen tersebut[1].

2.2.4.1 Teknik Pengkodean

Teknik pengkodean merupakan bagian penting dalam algortima genetika. Proses ini diperlukan dalam kaitan dengan peranan kromosom sebagai representasi penyelesaian masalah. Kromosom gabungan dari gen-gen yang membentuk nilai tertentu. Satu gen akan mewakili satu variable, agar dapat diproses melalui algoritma genetika. Gen dapat direpresentasikan dalam bentuk : *string bit*, bilangan real, daftar aturan, elemen permutasi, elemen program, atau representasi lainnya yang dapat diimplementasikan untuk operator genetika.



Gambar 2.2 Teknik Pengkodean Biner pada Operator Genetika

Demikian juga, kromosom dapat direpresentasikan dengan menggunakan :

- String bit : 011, 01101, 11101, dst.
- Bilangan Real : 65.65, -67.98. 562.88, dst.

- Elemen Program : pemrograman genetika
- Struktur lainnya

2.2.4.2 Membangkitkan Populasi

Membangkitkan populasi awal adalah proses membangkitkan sejumlah individu secara acak atau melalu *procedure* tertentu. Ukuran untuk populasi tergantung pada masalah yang akan diselesaikan dan jenis operator genetika yang diimplementasikan. Setelah ukuran populasi ditentukan, kemudian dilakukan pembangkitan populasi awal.

Teknik dalam pembangkitan populasi awal ini ada beberapa cara, diantaranya adalah sebagai berikut :

1. Random search

Pencarian solusi dimulai dari suatu titik uji tertentu. Titik uji tersebut dianggap sebagai alternatif solusi yang disebut sebagai populasi.

1. Random Generator

Random generator melibatkan pembangkitan bilangan random untuk nilai setiap gen sesuai dengan representasi kromosom yang akan digunakan.

2. Pendekatan tertentu

Cara ini dengan memasukkan nilai tertentu kedalam gen dari populasi awal yang dibentuk.

3. Permutasi Gen

Cara ini dengan penggunaan permutasi josephus dalam kombinatorial.

Pembentukan populasi awal yang digunakan adalah random search, dengan pencarian solusi berdasarkan satu titik uji yaitu dari data tugas mengajar.

2.2.4.3 Nilai *Fitness*

Suatu individu atau kromosom dievaluasi berdasarkan suatu fungsi tertentu sebagai ukutan performainya. Didalam evolusi alam, individu yang bernilai *fitness* tinggi yang akan bertahan hidup. Sedangkan bernilai *fitness* rendah akan mati.

Pada masalah optimasi, jika solusi yang dicari adalah memaksimalkan sebuah fungsi h (dikenal sebagai masalah maksimasi), maka nilai *fitness* yang digunakan

adalah nilai dari fungsi h tersebut, yakni *f* = h (dimana *f* adalah nilai *fitness*). Tetapi jika masalahnya adalah meminimalkan fungsi h, maka fungsi h tidak bisa digunakan secara langsung. Hal ini disebabkan adanya aturan bahwa individu yang memiliki nilai *fitness* tinggi lebih mampu bertahan hidup pada generasi berikutnya. Oleh karena itu nilai *fitness* yang digunakan adalah *f* = 1/h , yang artinya semakin kecil nilai h, semakin besar nilai *f* . Tetapi hal ini akan menjadi masalah jika h bisa bernilai 0, yang mengakibatkan *f* bernilai tak terhingga. Untuk mengatasinya, h perlu ditambah sebuah bilangan yang dianggap sangat kecil sehingga nilai *fitness* dapat dilihat pada persamaan 1 :

$$f = \frac{1}{(h+a)} \quad \text{.....(1)}$$

Dimana a adalah bilangan yang dianggap sangat kecil dan bervariasi sesuai dengan masalah yang akan diselesaikan. Penentuan nilai *fitness* sangat berpengaruh pada performasi algoritma genetika secara keseluruhan.

Nilai *fitness* yang digunakan untuk mencapai nilai optimal dalam penelitian ini adalah 1. Hasil nilai fitness tersebut adalah dari aturan yang telah ditetapkan dengan memberikan nilai 0 untuk aturan yang tidak dilanggar dan 1 untuk aturan yang dilanggar. Maka bila dimasukan pada rumus persamaan 1 akan menghasilkan nilai *fitness* 1, dengan h adalah mengenai aturan-aturan yang telah ditetapkan dan a adalah nilai terkecil yang digunakan adalah 1.

2.2.4.4 Seleksi

Pembentukan susunan kromosom pada suatu populasi baru biasanya dilakukan secara proporsional sesuai dengan nilai *fitness*nya.Suatu metode seleksi yang umumnya digunakan adalah *roulette-wheel*. Metode seleksi dengan mesin roulette ini merupakan metode yang paling sederhana dan sering dikenal dengan nama *stochastic sampling with replacement.* Dimana masing-masing kromosom menempati potongan lingkaran pada roda *roulette wheel* secara proporsional sesuai dengan nilai *fitness*nya. Kromosom yang memiliki nilai *fitness* lebih besar menempati potongan lingkaran yang lebih besar dibandingkan kromosom bernilai *fitness* rendah.

2.2.4.5 *Crossover* (Pindah Silang)

Sebuah kromosom yang mengarah pada solusi yang bagus dapat diperoleh dari proses memindah-silangkan dua buah kromosom. Pindah silang dapat berakibat buruk jika ukuran populasinya sangat kecil. Dalam suatu populasi yang sangat kecil, suatu kromosom dengan gen-gen yang mengarah ke solusi akan sangat cepat menyebar ke kromosom-kromosom lainnya. Untuk mengatasi masalah tersebut digunakan sebuah aturan bahwa pindah silang hanya dapat dilakukan dengan sebuah probabilitas tertentu pc. Pindah silang dapat dilakukan hanya jika suatu bilangan random [0,1] yang dibangkitkan kurang dari pc yang ditentukan. Pada umumnya pc diset mendekati 1 , misalnya 0,8.

Ada beberapa cara yang dapat digunakan untuk melakukan pindah silang sesuai dengan teknik pengkodean (*encoding)* yaitu sebagai berikut [1] :

1. *Binary encoding*

a. *Crossover* satu titik

Memilih satu titik tertentu, selanjutnya nilai biner sampai titik crossovernya dari induk pertama digunakan dan sisanya dilanjutkan dengan nilai biner dari induk kedua.

b. *Crossover* N-titik

Pada metode ini setiap kromosom induk dipotong menjadi N+1 bagian. Kromosom anak yang pertama dihasilkan dengan mewariskan bagian potongan urutan ganjil dari induk yang pertama dan mewariskan bagian potongan dalam urutan genap dari induk kedua. Demikian juga dengan kromosom anak yang kedua dihasilkan dengan cara yang sama dengan menggabungkan bagian yang tersisa dari kedua kromosom induknya.

c. *Crossover uniform*

Pada metode penyilangan ini, perlu dibangkitkan terlebih dahulu topeng penyilangan (*crossover mask*) yang terdiri atas kode biner sebanyak gen dalam kromosom. Gen-gen dalam kromosom anak ditentukan berdasarkan kode biner topeng penyilang. Untuk kromosom anak yang pertama, jika kode pada kedudukan topeng penyilangan bernilai satu, maka gen diwariskan dari induk pertama,

sedangkan jika induk kode pada topeng penyilangan bernilai nol, maka gen diwariskan dari induk kedua.

2. *Permutation encoding*

Memilih satu titik tertentu, nilai permutation sampai titik crossover. Pada induk pertama digunakan lalu sisanya dilakukan scan terlebih dahulu, jika nilai permutasi pada induk kedua belum ada pada *offspring* nilai tersebut ditambahkan.

3. *Tree encoding*

Memilih satu titik tertentu dari tiap induk, dan menggunakan tree dibawah titik pada induk pertama dan tree dibawah induk kedua.

2.2.4.6 Mutasi

Mutasi merupakan proses mengubah nilai dari satu atau beberapa gen dalam suatu kromosom. Mutasi ini berperan untuk menggantikan gen yang hilang dari populasi akibat seleksi yang memungkinkan munculnya kembali gen yang tidak muncul pada inisialisasi populasi. Metode mutasi yang digunakan adalah mutasi dalam pengkodean nilai. Proses mutasi dalam pengkodean nilai dapat dilakukan dengan berbagai cara, salah satunya yaitu dengan memilih sembarang posisi gen pada kromosom, nilai yang ada tersebut kemudian dirubah dengan suatu nilai tertentu yang diambil secara acak.

Contoh:

Kromosom sebelum mutasi : 1 ***3*** 4 ***7*** 6

Kromosom sesudah mutasi : 1 ***2*** 4 ***8*** 6

2.2.4.7 Syarat Berhenti

Proses optimasi yang dilakukan oleh algoritma genetika akan berhenti setelah suatu syarat berhenti terpenuhi. Beberapa syarat berhenti yang biasa digunakan adalah batas nilai fungsi fitness, batas waktu komputasi, banyak generasi dan terjadinya konvergensi. Syarat berhenti yang sering digunakan adalah banyak generasi, tetapi tidak menutup kemungkinan untuk dipilih kombinasi beberapa syarat berhenti.

2.2.4.8 Penentuan Parameter Algoritma

Parameter algoritma merupakan salah satu bagian penting dalam penerapan algoritma gentika yang tidak mudah untuk ditentukan secara pasti. Tidak ada aturan yang pasti untuk menentukan parameter algoritma, baik probabilitas crossover (Pc), probabilitas mutasi (Pm), maupun ukuran populasi. Hal ini tidak terlepas dari prinsip algoritma genetika yang mengandalkan bilangan acak hampir dalam setiap langkahnya, mulai dari pembentukan populasi awal, proses penyilangan atau proses mutasi. Bilangan acak yang berbeda pasti akan menyebabkan hasil yang berbeda pula [10]. Ada beberapa rekomendasi yang dapat digunakan , antara lain :

1. Untuk permasalahan yang memiliki kawasan solusi cukup besar , De Jong merekomdasikan untuk nilai parameter kontrol :
   (Popsize ; Pc ; Pm) = (50 ; 0,6; 0,001)
2. Bila rata-rata *fitness* setiap generasi digunakan sebagai indikator, maka Grefensttete merekomendasikan :
   (Popsize ; Pc ; Pm) = (30 ; 0,95; 0,01)
3. Bila *fitness* dari individu terbaik dipantau pada setiap generasi, maka usulannya adalah :
   (Popsize ; Pc ; Pm) = (80 ; 0,45; 0,01)
4. Ukuran populasi sebaiknya tidak lebih kecil dari 30, untuk sembarang jenis permasalahan.

Pada penelitian ini rekomendasi yang digunakan adalah rekomendasi no 3 , dipilih rekomendasi c dikarenakan untuk mencapai tujuan yang diinginkan dengan *fitness* = 1 ,harus dipantau setiap generasi. Maka dipilih rekomendasi no 3 dengan Popsize (ukuran populasi) 80, Pc (probabilitas *crossover*) 0,45. Untuk ukuran populasi tidak digunakan dikarenakan telah menggunakan banyak pengecekan serta ingin mencapai tujuan dengan *fitness* = 1 .

2.3 Bahasa Pemograman Web

Bahasa pemograman web adalah bahasa yang digunakan untuk membuat sebuah halaman web. Ada banyak bahasa pemograman web, namun yang akan dibahas dan digunakan dalam membangun aplikasi ini adalah: PHP, HTML, JavaScript, dan CSS[5].

2.3.1 HTML (*Hyper Text Markup Language*)

HTML adalah sebuah bahasa markah yang digunakan untuk membuat sebuah halaman web, menampilkan berbagai informasi di dalam sebuah penjelajah web internet dan pemformatan hiperteks sederhana yang ditulis dalam berkas format *ASCII* agar dapat menghasilkan tampulan wujud yang terintegrasi[6].

2.3.2 Javascript

JavaScript adalah bahasa pemograman berbasis *prototype* yang berjalan disisi klien. Jika kita berbicara dalam konteks web, sederhananya, kita dapat memahami JavaScript sebagai bahasa pemograman yang berjalan khusus untuk di browser atau halaman web agar halaman web menjadi lebih hidup. Kalua dilihat dari suku katanya terdari dua suku kata, yaitu Java dan Script. Java adalah bahasa pemograman berorientasi objek, sedangkan Script adalah serangkaian instruksi program[5].

2.3.3 CSS (*Cascading Style Sheet*)

CSS adalah kependekan dari *Cascading Style Sheet*. CSS merupakan salah satu kode pemrograman yang bertujuan untuk menghias dan mengatur gaya tampilan/layout halaman web supaya lebih elegan dan menarik. CSS adalah sebuah teknologi internet yang direkomendasikan oleh *World Wide Web Consortium* atau W3C pada tahun 1996. Awalnya, CSS dikembangkan di SGML pada tahun 1970, dan terus dikembangkan hingga saat ini. CSS telah mendukung banyak bahasa markup seperti HTML, XHTML, XML, SVG (*Scalable Vector Graphics*) dan Mozilla XUL (XML *User Interface Language*) [5].

2.3.4 PHP (Hypertext Preprocessor)

PHP adalah salah satu bahasa pemograman *script* yang dirancang untuk membangun aplikasi web. Ketika dipanggil dari *web browser,* program yang ditulis dengan *PHP* akan di parsing di dalam *web server* oleh interpreter *PHP* dan diterjemahkan ke dalam dokumen *HTML,* yang selanjutnya akan di tampilkan kembali ke web browser. Karena pemrosesan program *PHP* dilakukan

dilingkungan web server, *PHP* dikatakan sebagai bahasa sisi server (*server-side*). Oleh sebab itu, kode PHP tidak akan terlihat pada saat user memilih perintah "View Source" pada web browser yang mereka gunakan[6].

2.4 Basis Data

Basis data adalah kumpulan informasi yang disimpan di dalam komputer secara sistematik sehingga dapat diperiksa menggunakan suatu program komputer untuk memperoleh informasi dari basis data tersebut. Perangkat lunak yang digunakan untuk mengelola dan memangil kueri (*query*) basis data disebut sistem manajemen basis data (*database management sistem*, *DBMS*)[7].

2.5 Pemograman Berorentasi Objek

Pemrograman berorientasi objek atau *Object-Oriented Programming* merupakan paradigma pemrograman yang berorientasikan kepada objek. Semua data dan fungsi di dalam paradigma ini dibungkus dalam kelas-kelas atau objek-objek. Bandingkan dengan logika pemrograman terstruktur, setiap objek dapat menerima pesan, memproses data, dan mengirim pesan ke objek lainnya. Model data berorientasi objek dikatakan dapat memberi fleksibilitas yang lebih, kemudahan mengubah program, dan digunakan luas dalam teknik piranti lunak skala besar. Salah satu *tools* dan teknik dalam pengembangan sistem berorientasi objek adalah menggunakan UML[8].

2.6 UML (Unified Modelling Language)

UML merupakan bahasa yang telah menjadi standar untuk melakukan perancang, *visualisasi* dan mendokumentasikan sistem pada piranti lunak, dengan UML kita dapat memodelkan aplikasi yang dapat berjalan pada sistem operasi dan dapat ditulis menggunakan berbagai macam bahasa pemrograman, namun bahasa pemrograman yang sering digunakan dalam pembuatan *software* yaitu bahasa yang berorientasi objek seperti C++, JAVA, C# atau VB.NET [9][10].

Beberapa jenis diagram UML yaitu:

1. *Use Case* Diagram

   Menekankan kepada apa yang akan diperbuat sistem, sebuah *use case* menggambarkan interaksi aktor dengan sistem, aktor dapat berupa manusia maupun mesin. Sebuah *use case* dapat *include* terhadap *use case* lain, dapat digambarkan ketika ada *use case* yang *include* dengan *use case* yang di eksekusi maka *use case* yang *include* tersebut akan ikut dieksekusi, tidak hanya satu *use case* yang dapat *include* pada sebuah *use case,* lebih dari satu *use case* dapat *include* pada sebuah *use case.* Sebuah *use case* pun dapat meng-*extend use case* lain dengan *behavior-nya* sendiri.

2. *Class* Diagram

   Class Diagram adalah sebuah sepesifikasi yang diinstansikan akan menghasilkan objek, *class* diagram memiliki tiga area pokok : nama, atribut, dan metode. Pada atribut dan metode dapat memiliki salah satu sifat ini. *private, protected, public* . Hubungan antara *class* pada *class* diagram:

   a. Asosisasi, yaitu hubungan statis antara *class.* Asosiasi *class* pun memiliki atribut berupa *class* lain, panah penunjuk menunjukan arah *query* antara *class.*
   b. Agregasi, yaitu hubungan yang menyatakan bagian yang di dalamnya terdiri atas *class-class.*
   c. Pewarisan, yaitu hubungan hirarki antara *class. Class* dapat diturunkan kepada *class* lain.
   d. Hubungan dinamis, yaitu rangkaian pesan yang di-passing dalam satu *class* kepada *class* lain.

3. *Statechart* Diagram

   Diagram ini digunakan untuk menggambarkan perpindahan atau perubahan keadaan dari satu state ke state lainya. Dalam satu *class* dapat memiliki lebih dari satu state.

4. *Activity* Diagram

   Diagram ini menggambarkan bagaimana masing-masing alir berjalan. Sehingga *activity* diagram dapat menggambaran proses pararel yang mungkin terjadi pada eksekusi-eksekusi lain.

5. *Sequence* Diagram

   Diagram ini menggambarkan interaksi antara objek di dalam dan objek di sekitar sistem. Diagram ini dapat digunakan untuk menggambarkan skenario dan langkah-langkah yang dilakukan.

6. *Collaboration* Diagram

   Diagram ini menggambarkan interaksi antara objek, namun lebih menekankan pada peranan masing-masing objek seperti *sequence* diagram dan bukan merupakan penyampaian pesan, setiap pesan memiliki *sequence number*.

7. *Component* Diagram

   Diagram ini menggambarkan struktur dan hubungan antara komponen perangkat lunak, dengan ketergantungan diantaranya, komponen bisa berupa *interface*, yaitu kumpulan layanan yang disediakan sebuah komponen untuk komponen lain.

8. *Deployment Diagram*

   Menggambarkan detail bagaimana komponen di-*deploy* dalam instruktur sistem, pada apakah komponen akan terletak ? mesin, *server* atau perangkat keras lainya yang bersifat fisikal.

2.7 Perangkat Lunak Pendukung

Perangkat lunak pendukung merupakan perangkat lunak yang digunakan dalam mendukung pembangunan perangkat lunak. Berikut penjelasan dari perangkat lunak pendukung :

2.7.1 Netbeans IDE

Netbeans adalah sebuah aplikasi Integrated Development Environment (IDE) yang berbasiskan Java dari Sun Microsystems yang berjalan di atas swing. Swing merupakan sebuah teknologi Java untuk pengembangan aplikasi dekstop yang dapat berjalan pada berbagai macam platform seperti windows, linux, Mac OS X dan Solaris. Sebuah IDE merupakan lingkup pemrograman yang di integrasikan ke dalam suatu aplikasi perangkat lunak yang menyediakan Graphic User Interface (GUI), suatu kode editor atau text, suatu compiler dan suatu debugger.

Netbeans juga dapat digunakan progammer untuk menulis, meng-compile, mencari kesalahan dan menyebarkan program netbeans yang ditulis dalam bahasa pemrograman java namun selain itu dapat juga mendukung bahasa pemrograman lainnya dan program ini pun bebas untuk digunakan dan untuk membuat professional dekstop, enterprise, website, and mobile applications dengan Java language, C/C++, dan bahkan dynamic languages seperti PHP, JavaScript, Groovy, dan Ruby[6].

2.7.2 MySQL

MySQL adalah sebuah perangkat lunak sistem manajemen basis data SQL atau DBMS yang multithread, multi-user, dengan sekitar 6 juta instalasi di seluruh dunia. MySQL AB membuat MySQL tersedia sebagai perangkat lunak gratis di bawah lisensi GNU General Public License (GPL), tetapi mereka juga menjual dibawah lisensi komersial untuk kasus-kasus dimana penggunaannya tidak cocok dengan penggunaan GPL. Tidak seperti PHP atau Apache yang merupakan software yang dikembangkan oleh komunitas umum, dan hak cipta untuk kode sumber dimiliki oleh penulisnya masing-masing, MySQL dimiliki dan disponsori oleh sebuah perusahaan komersial Swedia yaitu MySQL AB. MySQL AB memegang penuh hak cipta hampir atas semua kode sumbernya. Kedua orang Swedia dan satu orang Finlandia yang mendirikan MySQL AB adalah: David Axmark, Allan Larsson, dan Michael "Monty" Widenius[6].

2.7.3 XAMPP

XAMPP merupakan singakatan dari X (program yang dapat dijalankan diberbagai sistem operasi), Apache, MySQL, PHP, Perl. XAMPP merupakan tool yang menyediakan paket perangkat lunak ke dalam satu buah paket. Terdapat Apache (web server), MySQL (database), PHP (server side scripting), Perl, FTP Server, phpMyAdmin dan berbagai pustaka bantu lainnya. Dengan menginstall XAMPP tidak perlu melakukan instalasi dan konfigurasi web server Apache, PHP dan MySQL secara manual. XAMPP akan menginstalasi dan mengkofigurasikannya secara otomatis[6].

2.8 Alpha BetaTesting

Alpha testing merupakan simulasi atau pengujian terhadap *software* yang telah dibangun yang dilakukan pengembang, calon pengguna aplikasi, pelanggan, maupun tim uji sebelum *sofware* tersebut dipasarkan. Sedangkan pengujian Beta merupakan pengujian yang dilakuakan setelah *software* sudah digunakan oleh pengguna, untuk menentukan apakah aplikasi tersebut memenuhi kebutuhan pengguna. Pengujian Beta sering digunakan utntuk mendapatkan umpan balik dari pengguna *software*[12].

2.9 Metode Black-box Testing

Menurut Ron Patton, Black-box testing merupakan istilah yang digunakan penguji perangkat lunak untuk menggambarkan bagaimana pendekatan pengujian mereka. Dalam pengujian black-box, penguji hanya tahu apa yang seharusnya dilakukan oleh software, namun penguji tidak dapat melihat bagaimana hal itu bisa beroperasi. Hal ini yang membuat penulis memilih menggunakan metode Black box, agar pengujian berfokus pada persyaratan fungsional perangkat lunak untuk melihat apakah program aplikasi menghasilkan output yang diinginkan atau tidak[12].