BAB 2
LANDASAN TEORI

2.1 Gizi Buruk

Gizi mempunyai peran penting dalam membina dan mempertahankan kesehatan seseorang seperti yang dibutuhkan oleh balita atau anak. Pemenuhan gizi merupakan kewajiban setiap orang untuk memelihara kesehatan.

Pada usia balita perkembangan kemampuan berbahasa, berkreatif, kesadaran sosial emosional, dan intelegensi anak berjalan cepat dan merupakan landasan perkembangan bagi anak selanjutnya. Pemberian nutrisi pada masa puncak pertumbuhan otak harus dimanfaatkan sebaik-baiknya gizi lengkap yang harus dapat dikonsumsi setiap hari (Kasdu, 2004)

Gizi buruk merupakan suatu kondisi kekurangan gizi pada tingkatan yang sudah berat, dimana status gizinya berada jauh di bawah standar. Gizi buruk akan terjadi manakala kebutuhan tubuh akan kalori, protein, atau bahkan keduanya tidak tercukupi. Menurut WHO salah satu masalah gizi buruk terjadi akibat konsumsi makanan yang tidak cukup mengandung energi dan protein serta karena adanya gangguan kesehatan. Anak disebut gizi buruk apabila berat badannya kurang dari berat badan normal. Sedangkan menurut Depkes RI (2005), gizi buruk adalah status gizi menurut berat badan (BB) dan tinggi badan (TB) dengan Z-score <-3 dan atau dengan tanda-tanda klinis (marasmus, kwasiorkor dan marasmus-kwasiorkor). Gizi buruk juga diartikan seseorang yang kurang gizi yang disebabkan oleh rendahnya konsumsi energi dan protein dalam makanan sehari-hari dan atau gangguan penyakit tertentu (Supariasa, Bakri, & Fajar, 2002).

(2.

Status gizi dihitung menggunakan z score. Z score adalah angka yang mengukur jarak suatu angka dari median. Berikut ini adalah rumus z score yang digunakan dalam penilaian status gizi:

$$Z\ score : \frac{Nilai\ Individu\ Subjek - Nilai\ Median\ Baku\ Rujukan}{Nilai\ Simpangan\ Baku\ Rujukan}$$

2.2 Analisis Data Spasial

Data spasial adalah data yang berkaitan dengan lokasi, berdasarkan geografi yang terdiri dari lintang-bujur dan wilayah. Analisis data spasial tidak dapat dilakukan secara global, artinya setiap lokasi mempunyai karakteristik sendiri. Sebagian besar pendekatan analisisnya merupakan eksplorasi data yang disajikan dalam bentuk peta tematik. Peta tematik juga disebut sebagai peta statistik atau peta tujuan khusus, menghasilkan gambaran penggunaan ruangan pada tempat tertentu sesuai dengan tema yang diinginkan. Berbeda dengan peta rujukan yang memperlihatkan pengkhususan geografi (hutan, jalan, perbatasan administratif), peta-peta tematik lebih menekankan variasi penggunaan ruangan daripada sebuah jumlah atau lebih dari distribusi geografis (Rachmawati, 2015). Distribusi geografis bisa berupa fenomena fisikal seperti iklim atau ciri-ciri khas manusia seperti kepadatan penduduk atau permasalahan kesehatan.

Data spasial di definisikan sebagai realisasi dari proses stokastik sebagai berikut:

$$Y(s) \equiv \{y(s), s \in D\} \tag{2.2}$$

dimana *D* adalah subset dari R^d. Data aktual dapat direpresentasikan oleh observasi $y = \{y(s_1), \ldots\ldots, y(s_n)\}$, himpunan ($s_1$,….$s_n$) menunjukkan unit spasial pengukuran diambil dari n area. D menjadi kontinu atau sekumpulan d – dimensi unit spasial yang dapat dihitung secara spasial kontinu atau proses acak diskrit (Gelfand, Diggle, Fuentes, & Guttorp, 2010).

2.3 *Gaussian Markov Random Field*

Pada geostatistical data, parameter di definisikan sebagai *latent stationary Gaussian field* (GF), sebuah fungsi dari beberapa *hyper-parameter* ψ yang terkait dengan distribusi prior p(ψ). Hal ini sama seperti mengasumsikan parameter θ berdistribusi Multivariate Normal dengan mean $\mu = (\mu_1, \ldots\ldots\ldots, \mu_n)$ dan matriks kovarians tersruktur spasial Σ (Blangiardo & Cameletti, 2013).

Dalam markov, elemen dari vektor parameter θ adalah independen dengan elemen yang lainnya. Matriks presisi $Q = \Sigma^{-1}$, menghasilkan komputasi yang bermanfaat. Dengan kata lain, untuk setiap pasangan elemen (i,j).

$$\theta_i \perp\!\!\!\perp \theta_j \mid \theta_{-ij} \leftrightarrow Q_{ij} = 0 \qquad (2.3)$$

Pola non-nol pada matriks presisi diberikan oleh proses struktur tetangganya *N(i)*. Demikian, $Q_{ij} \neq 0$ hanya jika $j \in \{i, N(i)\}$. Spesifikasi ini adalah *Gaussian Markov Random Field*.

2.4 Aproksimasi Laplace

Dalam matematika, metode Laplace dinamai Pierre-Simon Laplace, yang merupakan teknik yang digunakan untuk perkiraan integral dari bentuk (Rue, Martino, & Chopin, 2009):

$$\int_a^b e^{Lf(x)}\,dx \qquad (2.4)$$

Untuk L = 1 makan dapat dituliskan dan disolusikan sebagai berikut:

$$\int f(x)dx = \int \exp\big(log\big(f(x)\big)\big)\,dx \qquad (2.5)$$

dengan *f(x)* menyatakan fungsi densitas dari variabel acak X. Integral pada persamaan sebelumnya dapat diselesaikan sebagai berikut:

$$\int f(x)dx \approx exp(log f(x^*)) \int exp\left(-\frac{(x-x^*)^2}{2\sigma^{2*}}\right)dx \qquad (2.6)$$

dimana integral mengandung bentuk kernel dari fungsi densitas normal dengan rata-rata x^* dan varians σ^{2*}. Secara lebih jelas evaluasi integral dengan batas (a,b) dapat dituliskan sebagai:

$$\int_a^b f(x)dx \approx f(x^*)\sqrt{2\pi\sigma^{2*}}(\phi(b) - \phi(a)) \qquad (2.7)$$

dengan $\phi(.)$ menyatakan fungsi densitas kumulatif dari distribusi normal $N(x^*, \sigma^{2*})$. Konsep ini sangat berguna dalam menyelesaikan permaalahan dalam metode Bayesian.

2.5 *Integrated Nested Laplace Approximation* (INLA)

Model Spasial pada INLA di definisikan dengan *Bayesian Framework.* Metode MCMC (*Monte Carlo Markov Chain*) digunakan secara luas untuk model bayesian, tetapi terdapat keterbatasan pada perhitungannya. INLA dikembangkan untuk mengatasi permasalahan pada algoritma MCMC. Metode ini dikembangkan menggunakan lanten Gaussian Markov Random Field, yang lebih fleksibel untuk digunakan dalam beberapa tipe aplikasi yang berbeda CITATION Jay16 \l 1033 |(Mindra Jaya, Abdullah, Hermawan, & Ruchjana, 2016)}.

Model Spasial ini dibangun menggunakan Model Hirarki Bayesian dengan tiga tahap. Tahap pertama dalam pemodelan INLA untuk model univariate adalah mengidentifikasi distribusi data observasi y= (y_1, y_2…, y_n). Tahap kedua adalah mendefinisikan matriks presisi Q dengan laten gaussian dan tahap ketiga adalah proses kontrol *hyper-parameter*. Pendekatan yang paling umum dalam mendefinisikan distribusi dari y_i menurut parameter ϕ (umumnya rata-rata E(y_i)) dengan mendefinisikannya sebagai sebuah fungsi struktur aditive prediktor ηi

Model regresi bayes spasial pada awal nya terdiri dari residu terstruktur spasial yang dimodelkan dengan *conditional autoregresive* dan residu tak terstruktur spasial sebagai berikut:

$$\eta 1 - \log(\mu_i) - \propto + u_i + v_i \qquad (2.8)$$

Jika ditambahkan dengan beberapa covariate. Fungsi adaptif linear dapat dituliskan sebagai berikut (Blangiardo & Cameletti, 2013):

$$\eta i = \alpha + \sum_{m=1}^{M} \beta_m x_{mi} + \sum_{l=1}^{L} f(z_{li}) \qquad (2.9)$$

ηi adalah log resiko relatif pada wilayah ke –i. α pada formula tersebut merepresentasikan intersep, koefisien β = {β_1, … , β_K} menyatakan efek linear dari *covariate,* **x** = ($\mathbf{x}_1$, . . . , $\mathbf{x}_K$) terhadap resepon y; dan f={f_1(.),…,f_L(.)} merupakan sekumpulan fungsi dari covariate **z** = ($\mathbf{z}_1$, . . . , $\mathbf{z}_L$).

Fungsi $f_1(.)$ dapat mengakomodasi berbagai model, mulai dari *hierarchical regression* sampai ke *spatial model* (Rue, Martino, & Chopin, 2009).

Tujuan perhitungan bayesian pada INLA adalah menemukan densitas posterior marjinal untuk masing-masing elemen untuk semua paramater. Densitas posterior marjinal $(\theta_i|y)$, dapat diperoleh dengan mengintegrasikan *hyperparameter* sebagai berikut:

$$p(\theta_i|y) = \int p(\psi|y)p(\theta_i|\psi,y)d\psi \tag{2.10}$$

Fungsi densitas marginal $p(\psi|y)$ dari *hyper-parameter* ψ dapat diperoleh dari pendekatan Laplace dengan fungsi sebagai berikut:

$$\tilde{p}(\psi|y) \propto \frac{p(\theta_i,\psi,y)}{\tilde{p}(\theta_i|\psi,y)}|_{\theta=\theta^*(\psi)} \tag{2.11}$$

INLA menggunakan struktur hirarkis dimana tahap pertama terbentuk oleh fungsi *likelihood* dengan parameter yang diberikan θ = (α,β_k,f_i), dengan *hyperparameter* ψ dan data y.

Tahap 1: *Likelihood* $p(y|\theta,\psi)$

Dengan mengasumsikan ketergantungan independen, untuk sampel acak sebanyak *n* fungsi ditulis sebagai berikut:

$$p(y|\theta,\psi) = \prod_{i=1}^{n} p(y_i|\theta_i,\psi) \tag{2.12}$$

Tahap 2 : Laten Gaussian $p(\theta|\psi)$

Distribusi prior untuk parameter θ diasumsikan mengikuti distribusi multivariat normal:

$$p(\theta|\psi) = (2p)^{-\frac{n}{2}}|Q(\psi)|^{\frac{1}{2}}exp(-\frac{1}{2}\theta^T Q(\psi)\theta) \tag{2.13}$$

Tahap 3 : *Hyperparameter* $p(\psi)$

Fungsi distribusi posterior diperoleh sebagai berikut:

(2.14)

$$p(\theta,\psi|y) \propto p(\psi)\, x\, p(\theta|\psi)\, x\, p(y|\theta,\psi)$$

$$\propto p(\psi)\, x\, p(\theta|\psi)\, x \prod_{i=1}^{n} p(y_i|\theta_i,\psi)$$

$$\propto p(\psi)\, x\, |Q(\psi)|^{\frac{1}{2}} exp\left(-\frac{1}{2}\theta^T Q(\psi)\theta\right) x \prod_{i=1}^{n} p(y_i|\theta_i,\psi)$$

$$\propto p(\psi)\, x\, |Q(\psi)|^{\frac{1}{2}} exp\left(-\frac{1}{2}\theta^T Q(\psi)\theta + \sum_{i=1}^{n} \log(p(y_i|\theta_i,\psi))\right)$$

Setelah didapatkan fungsi distribusi posterior gabungan, untuk dapat memperoleh taksiran parameter model maka terlebih dahulu harus dicari fungsi densitas marjinal posterior untuk masing-masing parameter model seperti yang terlihat pada persamaan (2.10), dan setiap elemen *hyperparameter* dapat dinyatakan sebagai:

$$p(\psi_k|y) - \int p(\theta_i,\psi|y) d\psi_{-k} \tag{2.15}$$

Sehingga untuk sampai pada distribusi marjinal posterior diperlukan dua hal berikut. Pertama, mendapatkan fungsi densitas marjinal dari *hyperparameter* yang dikerjakan sebagai berikut:

$$p(\psi|y) = \frac{p(\theta,\psi|y)}{p(\theta|\psi,y)} \tag{2.16}$$

$$= \frac{p(\theta,\psi|y)p(\theta,\psi)}{p(y)} \frac{1}{p(\theta|\psi,y)}$$

$$= \frac{p(\theta,\psi|y)p(\theta|\psi)p(\psi)}{p(y)} \frac{1}{p(\theta|\psi,y)}$$

$$\propto \frac{p(\theta,\psi|y)p(\theta,\psi)p(\psi)}{p(\theta|\psi,y)}$$

$$\approx \frac{p(\theta,\psi|y)p(\theta,\psi)p(\psi)}{p(\theta|\psi,y)}\Big|_{\theta_{-i}=\theta^*_{-i}(\theta_i,\psi)}$$

Kedua, mendapatkan marjinal posterior untuk parameter yang akan di taksir sebagai berikut:

$$p(\theta_i|\psi,y) = \frac{p((\theta_i,\theta_{-i})|\psi|y)}{p(\theta_i|\theta_{-i},\psi,y)} \tag{2.17}$$

$$- \frac{p(\theta,\psi|y)}{p(\psi|y)} \frac{1}{p(\theta_i|\theta_{-i},\psi,y)}$$

$$= \frac{p(\theta,\psi|y)}{\tilde{p}(\psi|y)} \frac{1}{p(\theta_i|\theta_{-i},\psi,y)}$$

$$\propto \frac{p(\theta,\psi|y)}{p(\theta_i|\theta_{-i},\psi,y)}$$

$$\approx \frac{p(\theta,\psi|y)}{\tilde{p}(\theta_i|\theta_{-i},\psi,y)} |_{\theta_{-i}=\theta_{-i}^*(\theta_i,y)}$$

$$=: \tilde{p}(\theta_i|\psi,y)$$

2.5.1 Model Conditional Autoregressive (CAR)

Model CAR termasuk kedalam pendekatan area, model CAR merupakan model yang dikembangkan oleh Besag (1974) untuk mengakomodasi adanya ketergantungan spasial dalam kekeliruan. Spesifikasi Besag–York-Mollie (BYM) akan dimasukkan melalui model CAR. Model BYM banyak digunakan dalam pemodelan dan pemetaan kasus Bayesian/ Model CAR didefinisikan sebagai prior secara umum dimodelkan sebagai berikut (Besag, York, & Mollie, 1991):

$$u_i|u_{j\neq i} \sim Normal\left(\frac{\sum_{j\in N(i)} u_j}{\#N(i)}, \frac{\sigma^2}{\#N(i)}\right) \tag{2.18}$$

u_i adalah residual terstruktur spasial dimana $\#N(i)$ adalah jumlah wilayah yang berbagi batas dengan wilayah ke-i (tetangga dari wilayah tersebut). v_i adalah residual tak terstruktur spasial dimodelkan dengan $v_i \sim Normal\ (0, \sigma_v^2)$.

2.5.2 Fraksi Spasial

Fraksi spasial adalah proporsi varians spasial yang menunjukkan bahwa variabilitas dapat dijelaskan oleh struktur spasial yang sudah dibentuk. Nilai σ_u^2 adalah varians dari spesifikasi *conditional autoregressive*, sedangkan σ_v^2 adalah varians komponen marginal tidak terstruktur. Maka dari itu, keduanya tidak dapat dibandingkan secara langsung. Namun, dapat memperkirakan varians marginal posterior untuk efek terstruktur secara empiris sebagai berikut (Blangiardo & Cameletti, 2013):

$$S_u^2 = \frac{\sum_{i=1}^{n}(u_i - \bar{u})^2}{n-1} \tag{2.19}$$

$\bar{u}$ adalah rata-rata dari *u*, dan di bandingkan dengan varians marginal posterior untuk efek tidak terstruktur (σ_v^2). Maka, fraksi spasial dapat didefinisikan sebagai berikut:

$$frac_{spatial} = \frac{S_u^2}{(S_u^2 + \sigma_v^2)} \tag{2.20}$$

2.6 *R Programming*

R adalah bahasa pemrograman yang baik untuk komputasi statistika. Hal ini mirip dengan bahasa S yang dikembangkan oleh AT&T Bell Laboratories oleh Rick Becker, John Chambers dan Allan Wilks. Ada beberapa macam versi untuk R, antara lain R untuk Unix, Windows, dan berbagai macam Mac. Selain itu R juga dapat berjalan di berbagai arsitektur computer seperti *Intel, PowerPC, Alpha Sistem*, dan *Sistem Sparc*. Sumber Kode dari setiap komponen R tersedia secara bebas sehingga dapat diadaptasikan dengan baik. R memiliki keterbatasan dalam penanganan dataset yang sangat besar karena semua perhitungan dilakuka dalam memori utama komputer (Torgo, 2011).

R adalah bahasa fungsional, dimana terdapat inti bahasa yang menggunakan bentuk standar notasi aljabar, yang memungkinkan perhitungan numerik seperti 2+3, atau 3^11. Selain itu tersedia pula fasilitas perhitungan dengan menggunakan fungsi. R menyediakan berbagai macam tool statistik mulai dari pemodelan linier dan pemodelan non linier, uji statistik klasik, analisis time-series, klasifikasi, clustering, dan lain-lain. R juga menyediakan

tool teknik grafis yang bertujuan untuk menampilkan data hasil olahan secara visual dalam bentuk grafik (Vries & Meys, 2012).

Ada beberapa alasan menggunakan R yaitu sebagai berikut (Ihaka & Gentleman, 1996):

1. Serbaguna

   Pemrograman R berorientasi objek dan memiliki banyak *library* yang sangat bermanfaat yang dikembangkan oleh kontributor. Pengguna bebas menambah dan mengurangi *library* tergantung kebutuhan. R juga memiliki *interface* pemrograman C, Phyton bahkan Java. Jadi selain Bahasa R ini serbaguna, penggunanya juga dapat menjadi lebih kreatif dalam mengembangkan *library*.
2. Interaktif

   R dilengkapi konektivitas ke *database* server, OLAP, maupun format data *web service* seperti XML, *spreadsheet* dan sebagainya, sehingga apabila data set berubah hasil analisis juga dapat segera ikut berubah.
3. Berbasis S yaitu turunan dari tool statistik komersial S-Plus

   R hampir seluruhnya kompatibel dengan S-Plus. Artinya sebagian besar kode program yang dibuat oleh S dapat dijalankan di S-Plus kecuali fungsi-fungsi yang sifatnya *add-on packages* atau tambahan yang dibuat oleh kontributor proyek R.
4. Populer

   R merupakan bahasa yang popular digunakan oleh peneliti bidang statisitika. R juga popular untuk aplikasi kuantitatif di bidang keuangan.

2.7 *R Studio* dan *R Shiny*

RStudio merupakan *Integrated Development Environment* (IDE) untuk tampilan antarmuka dari *R Programming*. *RStudio* adalah perangkat lunak yang *open source* yang dapat dijalankan pada *desktop* atau *browser* yang terhubung dengan *server RStudio*.

RStudio dapat membangun aplikasi *web* yang interaktif dan mem-visualisasikan hasil analisis dari *R Programming*. *RStudio* memiliki *console*,

editor sintax yang mendukung eksekusi kode langsung dan juga beserta alat bantu untuk plot data, *history*, *debug* dan manajemen *workspace*. Aplikasi yang dibuat melalui RStudio dapat dijalankan (*deploy*) secara premis dengan *server R Studio* atau aplikasi tersebut di *hosting* ke *cloud* yang disediakan oleh Rstudio (*rstudio.org).*

R Shiny merupakan sebuah *framework* yang menyatukan analisis statistika dan visualisasi data berbasis *website* yang memungkinkan pengguna R dapat menciptakan sebuah *website* interaktif. *R Shiny* dapat digunakan dengan cara mengunduh dan memasang *package Shiny* melalui CRAN (*The Comperenshive R Archive Network)* terlebih dahulu pada *R Studio.* Aplikasi web *Shiny* bersifat reaktif, yang berarti semua keluaran hasil akan diperbarui secara otomatis sebagai respon dari interaksi pengguna (Resnizky, 2015).

2.8 *Database*

Database adalah koleksi yang berhubungan secara logis dari data-data yang saling berhubungan dan deskrpsi dari data, dirancang untuk memenuhi kebutuhan informasi dari suatu organisasi (Connolly, Thomas, & Carolyn, 2010).

2.8.1 MySQL

MySQL adalah *server database* relasional yang menawarkan mekanisme manajemen data yang disebut mesin penyimpanan. MySQL populer karena fleksibilitasnya yang dapat berjalan pada berbagai sistem operasi. MySQL memiliki fitur yang beraneka ragam seperti *full-text indexing* dan *searching* untuk mendukung *data mining* dari kolom teks. Pengambilan *query* juga merupakan salah satu fitur MySQL yang sangat efektif untuk menyimpan *query* yang dipilih

2.8.2 PhpMyAdmin

PhpMyAdmin adalah suatu program *open source* yang berbasis *web* yang dibuat menggunakan aplikasi PHP. Program ini digunakan untuk mengakses *database* MySQL. Program ini mempermudah dan mempersingkat kerja penggunanya. Dengan kelebihannya, para pengguna awam tidak harus paham sintak-sintak SQL dalam pembuatan *database* dan tabel (Firdaus, 2007).

2.9 Interaksi Manusia dan Komputer

Interaksi manusia dan komputer adalah perpaduan antara ilmu desain dan ilmu komputer dengan menerapkan prinsip desain yang baik kedalam komputer melalui alat-alat komputer canggih (Shneiderman & Plaisant, 2010)

2.9.1 Delapan Aturan Emas

Menurut Shneiderman dan Plaisant (2010) delapan aturan emas terdiri atas:

1. Konsistensi

   Aturan ini memberikan informasi agar aplikasi yang dikembangkan memiliki aturan yang pasti dan tidak berubah, di dalam suatu aplikasi. Misalnya aksi yang dilakukan dalam setiap proses yang mirip harus memiliki urutan yang sama, penggunaan jenis huruf, warna, letak, dan penulisan juga diperhatikan dalam aturan ini.

2. Menyediakan kebutuhan universal

   Perlunya mengenali kebutuhan yang beragam terhadap *user.* Setiap user memiliki kemampuan yang berbeda – beda. Untuk *user* pemula diperlukan fitur seperti penjelasan, sedangkan untuk user ahli dapat diberikan fitur cara pintas. Sehingga *user* dapat dengan mudah dalam menggunakan aplikasi.

3. Menawarkan umpan balik informatif

   Untuk setiap tindakan, harus disertakan dengan sistem umpan balik. Untuk tindakan yang sering dilakukan dan tidak terlalu penting, dapat diberikan respon yang sederhana. Sedangkan ketika tindakan merupakan hal yang penting, maka umpan balik sebaiknya lebih substansial. Presentasi visual dari obyek yang menarik dengan menyediakan lingkungan yang nyaman untuk menunjukkan perubahan secara eksplisit.

4. Desain dialog untuk memberikan keadaan akhir

   Setiap rangkaian aksi harus diakhiri dengan memberikan dialog pada aplikasi. Umpan balik ini akan memberikan informasi kepada pengguna bahwa aksi yang dilakukan sudah pada tahap yang terakhir.

5. Menghindari kesalahan

Merancang sistem sedemikian rupa sehingga pengguna tidak dapat membuat kesalahan serius, misalnya, item menu *grayout* yang tidak tepat dan tidak memungkinkan karakter abjad di bidang entri numerik. Jika pengguna membuat kesalahan, antarmuka harus mendeteksi kesalahan dan menawarkan instruksi sederhana, konstruktif, dan spesifik untuk pemulihan.

6. Mengizinkan pembalikan tindakan

   Hal ini dapat mengurangi kekawatiran pengguna karena pengguna mengetahui kesalahan yang telah dilakukan dapat dibatalkan, sehingga pengguna tidak takut untuk mengeksplorasi pilihan – pilihan.

7. Dukungan pusat kendali internal

   Aplikasi yang menarik biasanya memberikan kebebasan kepada pengguna dalam mengatur aplikasi yang digunakan seperti pengaturan suara dan warna. Kebebasan ini tentu saja masih dalam ruang lingkup yang mampu dikerjakan oleh aplikasi tersebut dan sesuai dengan kebutuhan.

8. Mengurangi beban ingatan jangka pendek

   Manusia memiliki kapasitas yang terbatas dalam mengingat. User interface yang dirancang harus menghindari rancangan dimana pengguna harus mengingat informasi yang bersangkutan. Informasi tersebut dapat berupa simbol dan gambar yang digunakan pada aplikasi.

2.9.2 Lima Faktor Manusia Terukur

Menurut Shneiderman dan Plaisant (2010) terdapat lima faktor manusia terukur yang harus dipertimbangkan dalam perancangan antarmuka antara lain:

1. Waktu belajar

   Jumlah waktu yang diperlukan oleh *user* untuk mempelajari melakukan sejumlah aksi yang diperlukan dalam menyelesaikan suatu tugas.

2. Kecepatan kinerja

Ukuran berapa lama waktu yang di butuhkan suatu fungsi atau serangkaian tugas di dalam aplikasi tersebut dijalankan.

3. Tingkah kesalahan user

   Jumlah dan jenis kesalahan yang dilakukan oleh *user* dalam menyelesaikan suatu tugas.

4. Daya ingat

   Kemampuan *user* dalam mengingat pengetahuan tentang antarmuka setelah satu jangka waktu tertentu.

5. Kepuasan subjektif

   Kepuasan masing-masing individu terhadap semua aspek yang ada dalam antarmuka.

2.9.3 *Black Box Testing*

Setiap aplikasi yang telah dibuat perlu dilakukan pengujian untuk memastikan bahwa aplikasi yang sudah dibuat dapat berjalan sesuai dengan kebutuhan. *Black box testing* adalah sebuah teknik pengujian fungsional yang merancang *test case* berdasarkan informasi dan spesifikasi (Nidhra, Srinivas, & Dondeti, 2012)

2.10 *Flowchart*

Flowchart merupakan teknik analisis untuk menggambarkan aspek-aspek dari sistem informasi dengan jelas, konsisten, dan logis. *Flowchart* menggunakan simbol untuk menggambarkan secara logis proses transaksi dan aliran data melalui sistem, serta menggunakan representasi bergambar yang mudah untuk dipahami dengan narasi yang terperinci (Romney & Steinbart, 2012).

Flowchart di bedakan menjadi 5 jenis *flowchart*, antara lain *system flowchart*, *document flowchart*, *schematic flowchart*, program *flowchart*, *process flowchart*. Masing-masing jenis *flowchart* akan di jelaskan berikut ini:

1. *System Flowchart*

   System flowchart dapat didefinisikan sebagai bagan yang menunjukkan arus pekerjaan secara keseluruhan dari sistem. Bagan

ini menjelaskan urut-urutan dari prosedur-prosedur yang ada didalam sistem. Bagan alir sistem menunjukkan apa yang dikerjakan di sistem.

2. *Document Flowchart*

   Bagan alir dokumen (*document flowchart*) atau disebut juga bagan alir formulir (*form flowchart*) atau *paperwork flowchart* merupakan bagan alir yang menunjukkan arus dari laporan dan formulir termasuk tembusan-tembusannya.

3. *Schematic Flowchart*

   Bagan alir skematik (*schematic flowchart*) merupakan bagan alir yang mirip dengan bagan alir sistem, yaitu untuk menggambarkan prosedur di dalam sistem. Perbedaannya adalah, bagan alir skematik selain menggunakan simbol-simbol bagan alir sistem, juga menggunakan gambar-gambar komputer dan peralatan lainnya yang digunakan. Maksud penggunaan gambar-gambar ini adalah untuk memudahkan komunikasi kepada orang yang kurang paham dengan simbol-simbol bagan alir. Penggunaan gambar-gambar ini memudahkan untuk dipahami, tetapi sulit dan lama menggambarnya.

4. *Program Flowchart*

   Bagan alir program (*program flowchart*) merupakan bagan yang menjelaskan secara rinci langkah-langkah dari proses program. Bagan alir program dibuat dari derivikasi bagan alir sistem.

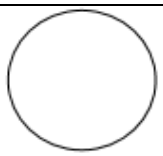
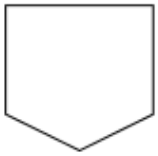
   Bagan alir program dapat terdiri dari dua macam, yaitu bagan alir logika program (*program logic flowchart*) dan bagan alir program komputer terinci (*detailed computer program flowchart*). Bagan alir logika program digunakan untuk menggambarkan tiap-tiap langkah di dalam program komputer secara logika. Bagan alat- logika program ini dipersiapkan oleh analis sistem. Bagan alir program komputer terinci (*detailed computer program flow-chart*) digunakan untuk menggambarkan instruksi-instruksi program komputer secara terinci. Bagan alir ini dipersiapkan oleh pemrogram.

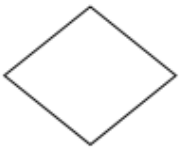
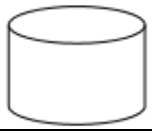
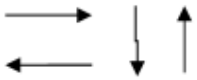
5. *Process Flowchart*

Bagan alir proses (*process flowchart*) merupakan bagan alir yang banyak digunakan. Bagan alir ini juga berguna bagi analis sistem untuk menggambarkan proses dalam suatu prosedur.

Pada tabel berikut ini menjelaskan simbol-simbol yang ada dalam *flowchart.* Simbol-simbol tersebut dapat dilihat dari Tabel 2.1

Tabel 2. 1 Simbol-simbol *Flowchart*

No.	Simbol	Keterangan
1.		Simbol *Start* atau *End* yang mendefinisikan awal atau akhir dari sebuah *flowchart.*
2.		Simbol proses atau pengolahan data, mempresentasikan sebuah operasi
3.		Simbol yang menyatakan bagian dari program (sub program).
4.		Simbol masukan atau keluaran dari atau ke sebuah pita *magnetic*.
5.		Simbol *Input/Output* yang mendefinisikan masukan dan keluaran proses.
6.		Simbol konektor untuk menyambung proses pada lembar kerja yang sama.
7.		Simbol konektor untuk menyambung proses pada lembar kerja yang berbeda.
8.		Simbol masukan atau keluaran dari atau ke sebuah dokumen.

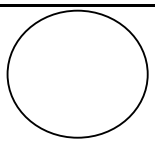
9.		Simbol untuk memutuskan proses lanjutan dari kondisi tertentu.
10.		Simbol *database* atau basis data.
11.		Simbol yang menyatakan piranti keluaran, seperti layar monitor, printer, dll.
12.		Simbol yang mendefinisikan proses yang dilakukan secara manual.
13.		Simbol untuk menghubungkan antar proses atau antar simbol.

2. 11 *Data Flow Diagram* (DFD)

Data Flow Diagram (DFD) atau dalam bahasa indonesia menjadi Diagram Alir Data (DAD) adalah representasi grafik yang menggambarkan aliran informasi dan transformasi informasi yang diaplikasikan sebagai data yang mengalir dari masukan (*input*) dan keluaran (*output*) (Sukamto & Shalahuddin, 2014). *Data Flow Diagram* (DFD) menunjukkan bagaimana data bergerak melalui suatu sistem informasi tetapi tidak menunjukkan logika program atau langkah-langkah pengolahan.

Notasi-notasi pada DFD adalah sebagai berikut:

Tabel 2. 2 Simbol-simbol *Data Flow Diagram* **(DeMarco, 1979)**

No	Simbol	Keterangan
1.		Proses atau fungsi atau prosedur, pada pemodelan perangkat lunak yang akan diimplementasikan.
2.		Penyimpanan data yang disimpan dan digunakan dalam pembuatan aplikasi pada pemrograman terstruktur
3.		Entitas luar (*external entity*) atau masukkann (*input*) atau keluaran (*output*) atau orang yang memakai/berinteraksi dengan perangkat lunak yang dimodelkan atau *system* lain yang terkait dengan aliran data dari sistem yang dimodelkan.
4.		Aliran data merupakan data yang dikirim antar-proses penyimpanan ke proses, atau dari proses ke masukan (*input*) atau keluaran (*output*).

Jenis-jenis DFD dibagi menjadi tiga tingkatan, dimana masing-masing *level* tersebut menggambarkan detail dari *level* sebelumnya, berikut penjelasan tiga jenis DFD tersebut:

1. DFD *Level 0* (Diagram Konteks)

   DFD *Level 0* menggambarkan sistem yang akan dibuat sebagai suatu entitas tunggal yang berinteraksi dengan orang maupun sistem lain. DFD *Level 0* digunakan untuk mengambarkan interaksi antara sistem yang akan dikembangkan dengan entitas luar.

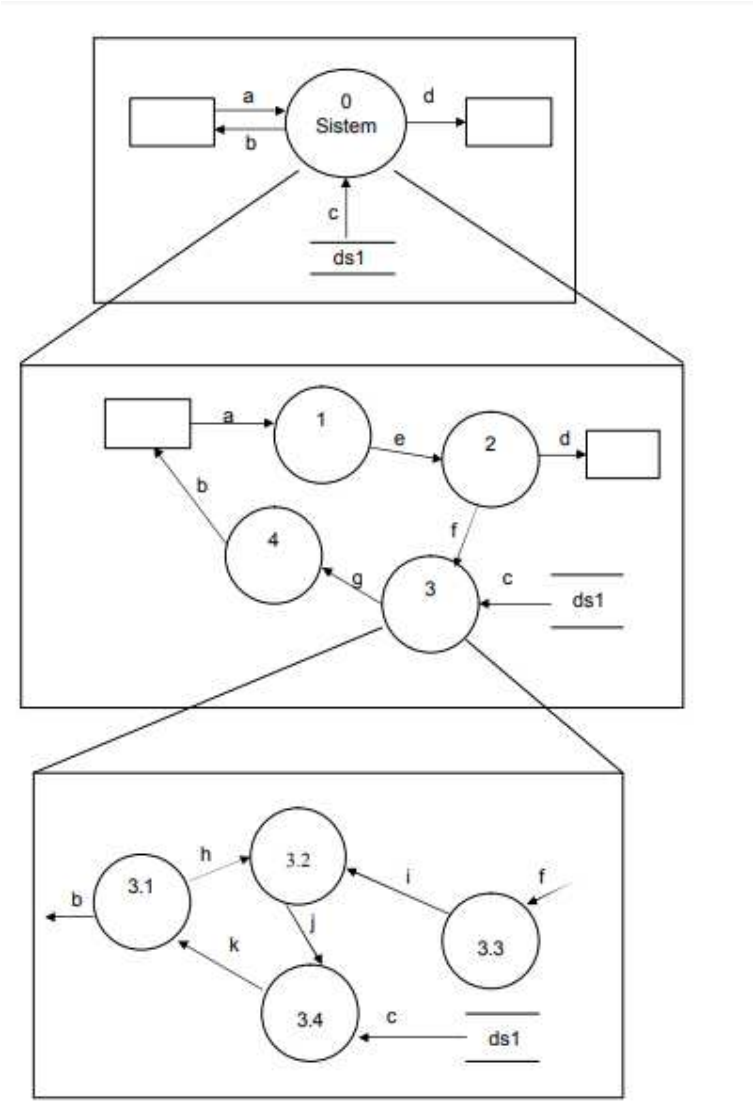
2. DFD *Level 1* (Diagram 0)

   Level ini merupakan sebuah proses yang terdapat di *Level 0* yang dipecahkan menjadi beberapa proses lainnya. Sebaiknya maksimum 7 proses untuk sebuah diagram konteks.

3. DFD *Level 2* (Diagram Rinci)

   Pada level ini merupakan diagram yang merincikan diagram *Level 1*. Tanda * pada proses menandakan bahwa proses tersebut tidak dapat dirincikan lagi. Penomoran yang dilakukan berdasarkan urutan proses.

Contoh gambar levelisasi DFD dapat dilihat pada Gambar 2.1 berikut:

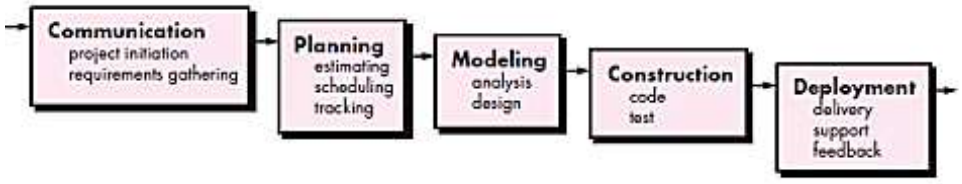


Gambar 2. 1 Levelisasi DFD

2.12 *Waterfall Development*

Model *waterfall* sering disebut dengan *classic life cycle* yang bersifat sistematis, berurutan dalam pembangunan *software.* Nama model ini sebenarnya adalah "*Linear Sequential Model"*. Waterfall pertama kali diperkenalkan oleh Royce pada tahun 1970 dengan tujuh langkah. Model ini mengalami banyak perbaikan dan perubahan diantaranya adalah perubahan langkah dari tujuh menjadi lima langkah (Pressman, 2010).

Langkah-langkah dalam model *waterfall* menurut Pressman:



Gambar 2. 2 Model *Waterfall* **(Pressman, 2010)**

1. *Commnication*

   Langkah ini merupakan analisis terhadap kebutuhan *software*, dan tahap untuk mengadakan pengumpulan data dengan melakukan pertemuan dengan customer, maupun mengumpulkan data-data tambahan baik yang ada di jurnal, artikel, maupun dari internet.

2. *Planning*

   Langkah perencanaan yang menjelaskan tentang estimasi tugas-tugas teknis yang akan dilakukan, resiko-resiko yang dapat terjadi, sumber daya yang diperlukan dalam membuat sistem, produk kerja yang ingin dihasilkan, penjadwalan kerja yang akan dilaksanakan, dan *tracking* proses pengerjaan sistem.

3. *Modeling*

   Proses *modeling* ini akan menerjemahkan syarat kebutuhan ke sebuah perancangan *software* yang dapat diperkirakan sebelum dibuat *coding*. Proses ini berfokus pada rancangan struktur data, arsitektur *software*,

representasi *interface,* dan detail (algoritma) prosedural. Tahapan ini akan menghasilkan dokumen yang disebut *software requirement.*

4. *Construction*

   Langkah *Construction* ini merupakan proses penerjemahan bentuk desain menjadi kode atau bentuk/bahasa yang dapat dibaca oleh mesin. Setelah pengkodean selesai, dilakukan pengujian terhadap sistem dan juga kode yang sudah dibuat. Tujuannya untuk menemukan kesalahan yang mungkin terjadi untuk nantinya diperbaiki.

5. *Deployment*

   Langkah ini bisa dikatakan final dalam pembuatan sebuah *software* atau sistem. Setelah melakukan analisis, desain dan pengkodean maka sistem yang sudah jadi akan digunakan oleh *user*. Kemudian *software* yang telah dibuat harus dilakukan pemeliharaan secara berkala.

Keuntungan menggunakan model *waterfall* adalah prosesnya lebih terstruktur, hal ini membuat kualitas *software* baik dan tetap terjaga. Dari sisi *user* juga lebih menguntungkan, karena dapat merencanakan dan menyiapkan kebutuhan data dan proses yang diperlukan sejak awal. Penjadwalan juga menjadi lebih menentu, karena jadwal setiap proses dapat ditentukan secara pasti. Sehingga dapat dilihat jelas target penyelesaian pengembangan program. Dengan adanya urutan yang pasti, dapat dilihat pula perkembangan untuk setiap tahap secara pasti. Dari sisi lain, model ini merupakan jenis model yang bersifat dokumen lengkap sehingga proses pemeliharaan dapat dilakukan dengan mudah.

Kekurangan yang utama dari model ini adalah kesulitan dalam mengakomodasi perubahan setelah proses dijalani. Fase sebelumnya harus lengkap dan selesai sebelum mengerjakan fase berikutnya. Model *waterfall* bersifat kaku sehingga sulit melakukan perubahan di tengah proses. Jika terdapat kekurangan proses/prosedur dari tahap sebelumnya, maka tahapan pengembangan harus dilakukan mulai dari awal lagi. Hal ini akan memakan waktu yang lebih lama. Karena jika proses sebelumnya belum selesai sampai akhir, maka proses selanjutnya juga tidak dapat berjalan. Oleh karena itu, jika terdapat kekurangan dalam permintaan *user* maka proses pengembangan harus

dimulai kembali dari awal. Karena itu, dapat dikatakan proses pengembangan software dengan metode waterfall bersifat lambat.

2.13 *Website*

Website adalah keseluruhan halaman-halaman *web* yang terdepat dalam sebuah domain yang mengandung informasi. Sebuah *website* biasanya dibangun atas banyak halaman *web* yang saling berhubungan. Hubungan antara satu halaman *web* dengan halaman *web* lainnya disebut *hyperlink*, sedangkan teks yang dijadikan media penghubung disebut *hypertext* (Yuhefizar, Mooduto, & Hidayat, 2009).