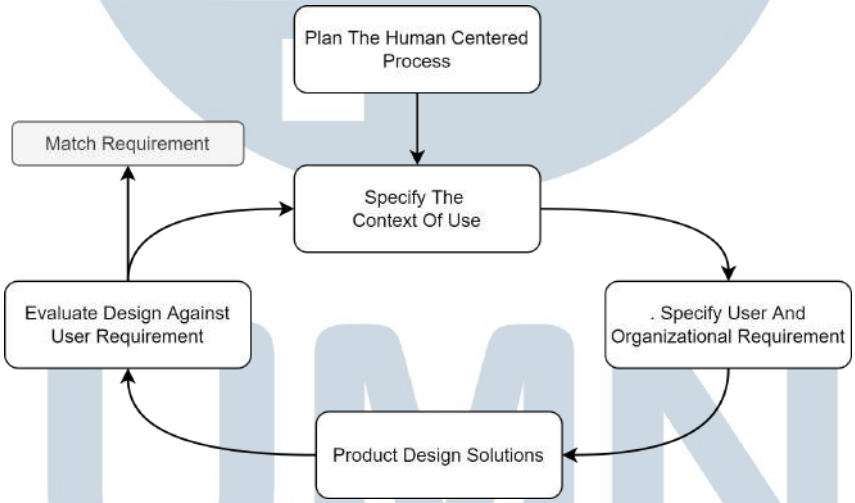
BAB 2
LANDASAN TEORI

2.1 *User Centered Design* (UCD)

User Centered Design (UCD) adalah metode desain sistem yang menempatkan pengguna sebagai pusat proses desain sistem yang termasuk tujuan atau sifat, konteks, dan lingkungan sistem didasarkan pada pengalaman pengguna [9]. Penerapan User Centered Design pada perancangan antarmuka yang didapatkan dari hasil pengumpulan data atau informasi yang diperlukan untuk membangun aplikasi [10]. Hal ini bertujuan untuk menekankan kebutuhan pengguna terhadap suatu produk yang akan dibangun sehingga dapat bermanfaat bagi pengguna dan nyaman untuk digunakan oleh pengguna tersebut [4].



Gambar 2.1. Aliran Proses User Centered Design (UCD)

Sumber: [11]

Prinsip-prinsip yang diterapkan pada metode UCD [11] yang ditunjukkan pada Gambar 2.1 yaitu:

1. *Specify the Context of Use*

   Memahami konteks dari kegunaan sistem yang akan dibangun. Konteks-konteks yang dimaksud seperti, siapa pengguna yang akan menggunakan sistem, alasan apa pengguna menggunakan sistem, dan dalam kondisi apa pengguna menggunakan sistem.

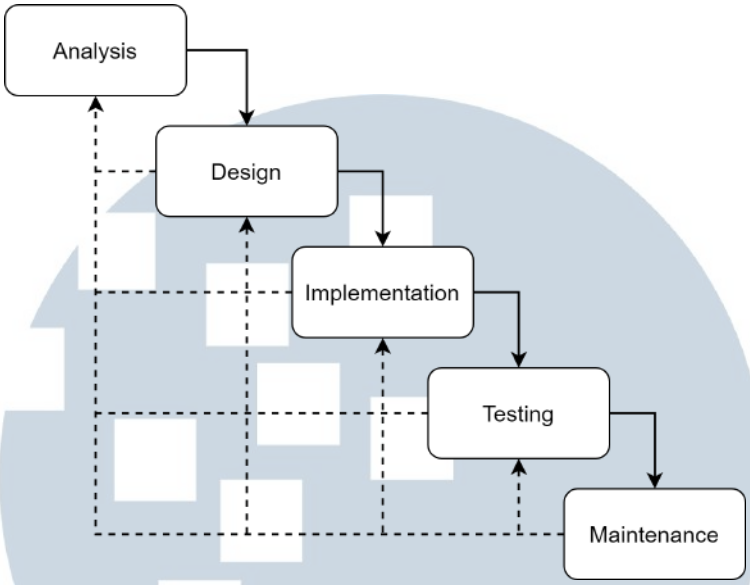
2. *Specify User and Organizational Requirement*
   Mengidentifikasi pengguna yang merupakan target dari pembangunan aplikasi. Survei dapat dilakukan untuk mengidentifikasi kebutuhan pengguna untuk mendapatkan data dan informasi untuk membuat desain aplikasi. Hasil data kebutuhan pengguna yang telah dianalisis dan dipilih akan dijadikan sebagai spesifikasi fitur dalam membuat desain.

3. *Product Design Solutions*
   Perancangan desain mengikuti spesifikasi fungsional dan non-fungsional yang telah ditetapkan. Perancangan desain termasuk dalam pembuatan alur proses sistem *wireframe* dan *mockup*.

4. *Evaluate Design Against User Requirement*
   Melakukan evaluasi terhadap desain dan prototipe pada tahapan sebelumnya apakah tujuan dan kebutuhan yang sudah ditetapkan sesuai dengan hasil identifikasi kebutuhan pengguna. Pengujian dilakukan dengan memberikan tugas-tugas untuk menyelesaikan suatu rangkaian aktivitas pada hasil *mockup* aplikasi.

2.2 Software Development Life Cycle (SDLC) Waterfall

Model *waterfall* merupakan model yang ditemukkan oleh Royce pada tahun 1970 dan merupakan model tertua yang digunakan dalam pengembangan *software* yang termasuk ke dalam *classic life cycle* (siklus hidup klasik). Model ini identik dengan pengembangan sistem secara sekuensial dan proses akan semakin ke bawah (seperti air terjun) dan setiap tahapan pada model ini harus diselesaikan sebelum pindah ke tahapan selanjutnya [12]. Untuk menyakinkan bahwa setiap proses dapat selesai maka model ini juga bersifat rekursif yaitu setiap proses dapat diulang sampai tujuan pengembangan tercapai [13]. Keunggulan menggunakan metode *Waterfall* yaitu mudah untuk dipahami karena proses yang terurut, penerapan *feedback*, dan cocok digunakan untuk pengerjaan proyek yang sederhana dengan waktu singkat.



Gambar 2.2. Metode SDLC Waterfall

Sumber: [13]

Proses pengembangan model *waterfall* terdiri dari *analysis, design, implementation, testing,* dan *maintenance* [14] yang ditunjukkan pada Gambar 2.2.

1. *Analysis*

   Melakukan pengambilan data atau informasi untuk memenuhi kebutuhan *software* yang akan dibangun. Proses analisis berkaitan dengan spesifikasi kebutuhan yang dibutuhkan oleh perangkat lunak sehingga hasil yang didapatkan sesuai dengan kebutuhan user. Pengumpulan data atau informasi dapat dilakukan dengan wawancara, observasi, studi literatur, diskusi, dan survei.

2. *Design*

   Hasil analisis data atau informasi yang telah dilakukan selanjutnya dibangun rancangan desain *software* untuk mendapatkan gambaran yang jelas terkait perangkat lunak yang akan dibangun. Proses desain termasuk pada pembuatan arsitektur, struktur data, *wireframe*, dan proses setiap bagian pada perangkat lunak.

3. *Implementation*

   Pengimplementasian hasil desain yang telah dirancang ke dalam bentuk program. Hasil pada tahap ini yaitu hasil desain yang dapat diproses oleh

komputer melalui bahasa pemrograman dan sudah terintegrasi dengan sistem basis data.

4. *Testing*

   Pengujian pada perangkat lunak yang penting untuk memastikan aspek fungsional dapat berjalan dengan sesuai. Selain memastikan aspek fungsional berjalan dengan sesuai, pengujian ini juga mencegah terjadinya kesalahan *error* dan memeriksa kembali desain perangkat lunak telah sesuai. Seperti melakukan *debugging* sistem untuk menemukan *bug* dan gangguan pada sistem sehingga dapat segera diperbaiki.

5. *Maintenance*

   Proses selanjutnya yang dilakukan yaitu pemeliharaan atau perbaikan yang mungkin terjadi ketika aplikasi digunakan oleh user. Hal-hal yang mungkin terjadi seperti ditemukkan *output* yang tidak sesuai, kesalahan *error* yang ditemukan oleh user, adanya kebutuhan baru, atau peningkatan keandalan dari perangkat lunak. Kegiatan pemeliharaan tambahan pada tahap ini dapat dilakukan seperti menyesuaikan perangkat lunak dengan lingkungan, mengakomodasi *user requirement* yang baru, dan meningkatkan keandalan dari *software* [13].

2.3 Collaborative Filtering

Sistem rekomendasi merupakan bidang yang dimanfaatkan untuk mengatasi persoalan terkait pencarian suatu informasi. Penerapan sistem rekomendasi diterapkan pada bidang yang memiliki jumlah data yang banyak dan terus berkembang. Salah satu algoritma yang digunakan untuk membuat rekomendasi yaitu algoritma *Collaborative Filtering*. *Collaborative filtering* (CF) adalah konsep yang digunakan untuk melakukan prediksi berdasarkan opini dari pengguna lain terhadap *item* yang mungkin disukai/diminati oleh pengguna [8]. Hasil rekomendasi ditentukan dari pengguna yang serupa dan merekomendasikan apa yang disukai kepada pengguna yang memiliki kesamaan. Hasil rekomendasi dapat berupa prediksi (numerik) yang mengindikasikan tingkat kesukaan seseorang terhadap sebuah *item* dan daftar berisi *item* yang direkomendasikan. Keunggulan dari *recommender system* yaitu dapat secara efektif membantu pengguna untuk mendapatkan informasi yang bermanfaat dan sesuai dengan minat pengguna. Adanya peningkatan jumlah informasi di internet, sistem rekomendasi merupakan

Rancang Bangun Aplikasi..., Queen Virginia Jeanifer Tambayong, Universitas Multimedia Nusantara

pilihan yang sesuai untuk memecahkan hal tersebut [15]. Sebagai contoh aplikasi yang menggunakan sistem rekomendasi yaitu Youtube, Amazon, Netflix, dan Facebook. Untuk memperoleh data *rating*, dibedakan menjadi dua yaitu:

1. Secara Eksplisit, ketika pengguna diminta untuk memberikan opini/pendapat terhadap suatu *item* secara langsung.
2. Secara Implisit, ketika pengguna tidak menyadari telah memberikan masukkan kepada sistem. Sehingga sistem secara otomatis mendapatkan preferensi pengguna.

Collaborative filtering merupakan salah satu algoritma yang digunakan para peneliti untuk membangun *recommender system*. Beberapa penelitian terkait seperti yang dilakukan oleh Wijaya dan Alfian dalam pembangunan sistem rekomendasi laptop menggunakan Collaborative Filtering dan Content-Based Filtering [8], penelitian pembangunan teknologi informasi menggunakan metode User-Based Collaborative Filtering oleh Alim, Solichin, dan Painem [16], dan pembangunan sistem rekomendasi drama Korea menggunakan *User-Based Collaborative Filtering*.

Algoritma *User Based Collaborative Filtering* merupakan algoritma yang untuk memberikan rekomendasi dengan menemukan *item* yang disukai oleh user dan menemukan user lain yang memiliki ketertarikan yang sama dengan user tersebut. Secara statistika, memanfaatkan sekumpulan tetangga (*neighbour)* dalam hal ini yaitu user yang akan menggabungkan sekumpulan kesukaan dari *neighbour* untuk menghasilkan suatu rekomendasi atau prediksi. Tujuan *Collaborative Filtering* diterapkan untuk menghitung nilai prediksi rating terhadap item yang kosong atau belum diberikan rating.

Untuk menghitung nilai kemiripan antar user menggunakan persamaan *Euclidean Distance* yang ditunjukkan pada Persamaan 2.1. Berikut rumus menghitung nilai kemiripan antar user [17].

$$D(x,y) = \sqrt{\sum_{i=1}^{n}(X_i - Y_i)^2} \tag{2.1}$$

Keterangan:

D(x,y) = Nilai *distance* antara user x dan user y

n = Jumlah total *item*

Xi = *Rating* user X terhadap *item* i

Rancang Bangun Aplikasi..., Queen Virginia Jeanifer Tambayong, Universitas Multimedia Nusantara

Yi = *Rating* user Y terhadap *item* i

Berdasarkan Persamaan 1 akan didapatkan nilai *Distance*, semakin besar nilai yang didapatkan semakin besar perbedaan *rating* yang diberikan oleh pengguna tersebut. Sebaliknya, semakin kecil nilai Distance, semakin besar kemiripan *rating* terhadap kedua pengguna tersebut. Untuk menghitung nilai similaritas, berdasarkan nilai *distance* yang telah didapatkan. Berikut persamaan untuk menghitung nilai similaritas yang ditunjukkan pada Persamaan 2.2 [17].

$$sim(x,y) = \frac{1}{1 + D(x,y)} \tag{2.2}$$

Keterangan:

sim(x,y) = Nilai kemiripan antara user x dan user y

D(x,y) = Nilai *distance* antara user x dan user y

Rentang nilai similaritas yaitu rentang 0 sampai dengan 1, semakin tinggi nilai similaritas yang didapatkan maka semakin tinggi kemiripan antar user.

Setelah mencari nilai kemiripan antar user, selanjutnya menghitung prediksi *rating* dari item berdasarkan informasi yang telah diberikan oleh user-user. Untuk melakukan prediksi, dilakukan perhitungan *rating* dengan membandingkan kemiripan *item* dengan *item* lainnya dari pengguna yang memberikan *rating*. Metode yang digunakan yaitu *Weighted Sum* yang ditunjukkan pada Persamaan 2.3. Berikut rumus perhitungan prediksi *Weighted Sum* [17].

$$Pred_{u,i} = \frac{\sum_{i=1}^{m} R_{v,i} \times sim(u,v)}{\sum_{i=1}^{m} |sim(u,v)|} \tag{2.3}$$

Keterangan:

Pred(u,i) = Prediksi user u terhadap *item* i

sim(u,v) = Nilai kemiripan antara user u dan user v

m = Jumlah total user

R v,i = *Rating* oleh user v kepada *item* i

2.4 *Black Box Testing*

Black Box testing adalah metode yang dilakukan dalam pengujian perangkat lunak yang berfokus pada spesifikasi fungsional [5]. Spesifikasi fungsional berkaitan dengan aspek yang berada pada sistem tersebut sehingga pengembang sistem membuat serangkaian kondisi input untuk membuktikan batasan-batasan

Rancang Bangun Aplikasi..., Queen Virginia Jeanifer Tambayong, Universitas Multimedia Nusantara

dari sistem tersebut. Keuntungan yang diberikan menggunakan metode *Black Box* yaitu penguji mengenyampingkan pengetahuan terkait bahasa pemrograman [18]. Hal ini bertujuan untuk melihat dari sudut pandang pengguna terkait fungsi-fungsi fungsional pada sistem dapat berjalan sesuai dengan fungsinya. Hal-hal yang menjadi fokus dari *black box testing* yaitu:

1. Menemukan kesalahan pada desain antarmuka.
2. Adanya fungsi yang tidak sesuai.
3. Kesalahan pada performa.
4. Kesalahan pada basis dan struktur data.

2.5 *Usability*

Usability merupakan pengukuran dari tingkat kemudahan pengguna dalam menggunakan perangkat lunak. Pengertian lain dari *Usability* yaitu sebagai suatu ukuran untuk pengguna dapat mengakses fungsionalitas dari sebuah sistem dengan efektif, efisien, dan memuaskan untuk mencapai tujuan tertentu [19]. Tingkat *usability* suatu perangkat lunak atau aplikasi dapat berbeda tergantung bagaimana pengguna dapat menyelesaikan serangkaian tugas-tugas [20]. Untuk melakukan pengujian usability maka terdapat kriteria-kriteria yang harus dipenuhi untuk mengukur *usability* [21] yaitu:

1. Kemampuan Belajar (*Learnability*)
   Kriteria untuk mengukur kemampuan pengguna belajar ketika menggunakan aplikasi untuk pertama kali. Hal ini untuk mengukur aplikasi yang diuji memberikan kemudahan bagi pengguna, khususnya untuk pengguna yang baru pertama kali mencoba.
2. Efisiensi (*Efficiency*)
   Kriteria untuk mengukur tingkat kecepatan pengguna dalam menyelesaikan tugas-tugas setelah mempelajari aplikasi.
3. Dapat Diingat (*Memorability*)
   Kemampuan pengguna dalam mengingat fitur-fitur dan cara mengoperasikan aplikasi. Kriteria yang diukur yaitu bagaimana pengguna dapat dengan mudah mengingat hal-hal yang ada pada produk tersebut.

4. Kesalahan (*Error*)
   Kriteria untuk mengukur tingkat terjadinya kesalahan oleh pengguna ketika menjalankan aplikasi. Pengujian yang dilakukan untuk mengetahui adanya kesalahan pada *website* sehingga mengganggu pengguna.

5. Kepuasaan (*Satisfaction*)
   Kriteria untuk mengukur tingkat kepuasaan pengguna ketika menggunakan aplikasi. Pengukuran kriteria ini untuk mengukur apakah pengguna merasa senang atau nyaman ketika menggunakan aplikasi.

Beberapa penelitian menggunakan *usability* telah diterapkan di beberapa area. Seperti pembuatan website untuk *e-commerce* yang menggunakan *usability testing* [5]. Aplikasi inventaris untuk pengolahan informasi di sekolah [6] dan aplikasi pada bidang pariwisata [7]. Aplikasi dengan kegunaan yang rendah dapat menyebabkan pemborosan waktu, pengguna merasa frustasi, pengguna tidak ingin kembali atau mengunjungi aplikasi.

2.6 Skala Likert

Skala Likert atau *Likert Scale* dikembangkan oleh Rensis Likert pada tahun 1932 dan merupakan metode dasar dalam penskalaan dalam penelitian [22]. Pengertian dari skala likert yaitu pengukuran sikap dan pendapat yang diberikan responden melalui serangkaian pertanyaan yang diberikan melalui kuesioner [23]. Setiap jawaban dapat direpresentasikan sebagai pernyataan dengan gradasi nilai positif atau nilai negatif. Skala nilai yang diberikan pada setiap pernyataan yaitu skala 1-5. Parameter bobot nilai dalam penilaian menggunakan skala Likert ditunjukkan pada Tabel 2.1.

Tabel 2.1. Tabel Skala Likert
Sumber: [19]

Pernyataan	Skor
Sangat Setuju (SS)	5
Setuju (S)	4
Netral (N)	3
Tidak Setuju (TS)	2
Sangat Tidak Setuju (STS)	1

Menurut Sugiyono [23], perhitungan atau formula untuk menghitung persentase hasil penilaian skor ditunjukkan pada Persamaan 2.4.

$$PS = \frac{(SS \times 5) + (S \times 4) + (N \times 3) + (TS \times 2) + (STS \times 1)}{5 \times Jumlah\ Responden} \times 100\% \qquad (2.4)$$

dimana,

PS = Persentase Skor
SS = Jumlah Jawaban Sangat Setuju
S = Jumlah Jawaban Setuju
N = Jumlah Jawaban Netral
TS = Jumlah Jawaban Tidak Setuju
STS = Jumlah Jawaban Sangat Tidak Setuju

Hasil akhir dari perhitungan persentase akan digunakan sebagai penentu kelas data yang ditunjukkan pada Tabel 2.2. Interval pada setiap skor ditentukkan berdasarkan jumlah kriteria yang digunakan.

Tabel 2.2. Tabel Persentase Interval
Sumber: [23]

Presentase Interval	**Kriteria**
0% - 19,99%	Sangat Tidak Setuju
20% - 39,99%	Tidak Setuju
40% - 59,99%	Netral
60% - 79,99%	Setuju
80% - 100%	Sangat Setuju

UNIVERSITAS MULTIMEDIA NUSANTARA