BAB II
LANDASAN TEORI

2.1. Studi Literatur

2.1.1. *Human-Computer Interaction (Interaksi Manusia dan Komputer)*

Suatu sistem informasi harus dibangun dengan memerhatikan aspek-aspek dalam interaksi manusia dan komputer. Interaksi manusia dan komputer atau *human computer interaction* (HCI) merupakan sebuah disiplin ilmu yang mengkaji tentang desain, evaluasi, dan implementasi sistem komputasi yang interaktif untuk digunakan manusia serta studi tentang fenomena di sekitarnya (Hewett, dkk., 1992). Interaksi manusia-komputer adalah suatu proses yang terjadi ketika pengguna manusia dan sistem komputer bertemu, dalam artian luas, untuk mencapai sesuatu (Hartson & Pyla, 2012).

HCI berperan untuk menghasilkan sebuah sistem yang berguna, aman, produktif, efektif, efisien dan fungsional. Agar interaksi antara komputer dan pengguna menjadi efisien, sebuah antarmuka harus dirancang sesuai dengan kemampuan pengguna dalam pemrosesan informasi.

Desain antarmuka yang baik tidak dimulai dengan gambar, tapi dimulai dengan memahami pengguna: seperti apa mereka, mengapa mereka menggunakan perangkat lunak tertentu, dan bagaimana mereka dapat berinteraksi dengannya. Semakin banyak yang pengetahuan tentang pengguna, dan semakin berempati dengan mereka, proses perancangan akan semakin efektif. Perangkat lunak merupakan alat atau sarana bagi orang-orang yang menggunakannya untuk mencapai tujuan. Semakin baik pengembang sistem memenuhi tujuan tersebut, pengguna juga semakin puas.

2.1.2. *User Interface (UI)*

User Interface (UI) adalah bagian dari sistem yang bertindak sebagai perantara antara pengguna dan sistem yang memfasilitasi pengguna untuk berinteraksi dengan sistem secara efisien. UI adalah segala sesuatu yang

berhubungan dengan pengguna akhir saat menggunakan sistem secara fisik, persepsi, dan konseptual (Saha & Mandal, 2015). UI adalah hal yang dilihat pengguna di layar: simbol, konten, warna, latar belakang, dan komponen bergerak. Dengan demikian, UI mencakup banyak hal mengenai penggambaran visual. Desain UI mempertimbangkan semua komponen visual dan interaktif dari antarmuka perangkat, termasuk tombol, simbol, penyebaran, tipografi, desain warna, dan desain responsif (Branson, 2020).

User interface (UI) memiliki peran yang penting dalam meningkatkan kegunaan aplikasi karena merupakan media interaksi komputer dan manusia. Karena UI memberikan tampilan abstrak dari keseluruhan sistem kepada pengguna, keberhasilan suatu sistem sangat bergantung padanya. Jika sebuah aplikasi tidak memenuhi kebutuhan pengguna, maka aplikasi itu bisa dikatakan gagal. Oleh karena itu, mendesain UI sangat penting dalam proses *system design life cycle* (SDLC) (Saha & Mandal, 2015).

Desain *interface* merupakan proses kompleks yang melibatkan desainer, target audiens, dan klien; merupakan proses berulang, dengan fase penelitian pengguna, pengembangan ide, percobaan, pembuatan dan pengujian lebih lanjut, semuanya berkontribusi untuk membuat desain interaktif untuk memperoleh pengalaman pengguna yang tepat (Wood, 2014). Setiap bisnis yang mengandalkan sistem aplikasi perlu mengembangkan *user interface* yang baik. *User interface* yang baik adalah *user interface* yang mampu mewujudkan tampilan dan interaksi yang mudah dimengerti oleh pengguna (*user friendly*), karena UI memiliki peranan penting pada sebuah aplikasi yaitu sebagai penghubung antara pengguna dengan sistem aplikasi itu sendiri serta sebagai faktor kesuksesan aplikasi.

2.1.3. *User Experience (UX)*

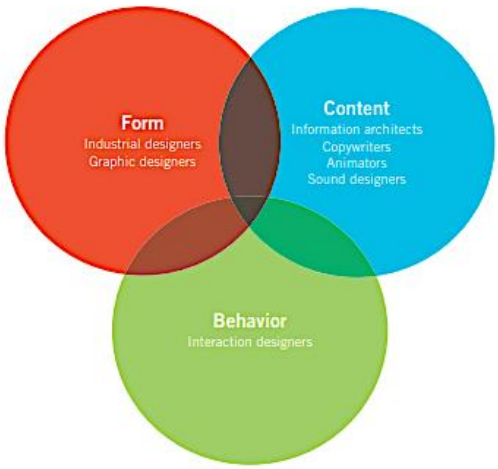
Menurut ISO (*International Organization for Standardization*), definisi terbaru *User Experience (UX)* adalah persepsi dan respon pengguna yang dihasilkan dari penggunaan sistem; termasuk interaksi pengguna dengan pihak yang menyediakan atau mengembangkan sistem tersebut, mulai dari menemukan sistem, mengadopsi dan menggunakannya, hingga penggunaan akhir (Bevan, dkk., 2016).

UX adalah keseluruhan dampak atau pengaruh yang dirasakan oleh pengguna sebagai hasil interaksi dan penggunaan suatu sistem, perangkat, atau produk, termasuk pengaruh daya guna, kebermanfaatan, dan dampak emosional selama interaksi, dan menikmati memori setelah interaksi. "Interaksi" yang dimaksud mencakup melihat, menyentuh, dan berpikir tentang sistem atau produk (Hartson & Pyla, 2012).

Sederhananya, UX adalah bentuk emosional yang dirasakan pengguna setelah menggunakan sistem informasi. Berdasarkan pengertian tersebut, dalam membangun UX yang berkualitas, penting untuk membangun hubungan dengan pengguna secara mendalam.

Desain UX merupakan cara untuk membangun hubungan antara organisasi, perangkat, dan penggunanya. Desain UX dilakukan untuk menyelidiki, membuat, dan meningkatkan semua bagian interaksi pengguna dengan perangkat organisasi agar memenuhi kebutuhan penggunanya (Branson, 2020).

Menurut Cooper, et al. (2014), sebuah proyek desain membutuhkan perhatian yang cermat pada integrasi sejumlah disiplin desain untuk mencapai *user experience* yang sesuai. Dalam hal ini, desain UX memiliki tiga masalah yang saling tumpang tindih, yaitu *form* (bentuk), *behavior* (perilaku), dan *content* (konten). Seperti yang digambarkan pada Gambar 2.1.



Gambar 2.1 Tiga masalah desain UX (Cooper, dkk., 2014)

Desain interaksi berfokus pada desain perilaku, tetapi juga berkaitan tentang bagaimana perilaku itu berhubungan dengan bentuk dan konten. Demikian pula dengan arsitektur informasi yang berfokus pada struktur konten, tetapi juga berkaitan dengan perilaku yang menyajikan akses terhadap konten dan cara konten disajikan kepada pengguna. Desain industri dan desain grafis memperhatikan bentuk (visual) produk dan layanan, tetapi juga perlu memperhatikan perilaku dan konten untuk memastikan bahwa bentuknya itu mendukung penggunaan.

2.1.4. *Task Centered System Design (TCSD)*

Task Centered System Design (TCSD) merupakan metode untuk membangun *Human Computer Interaction* (HCI) yang berguna untuk mengidentifikasi kebutuhan tugas dan kebutuhan pengguna. Metode ini menggunakan pendekatan berdasarkan *task* (tugas) sehari-hari dari pengguna. Jadi, alur dalam sistem yang akan dibangun mengikuti alur aktivitas dalam kehidupan nyata. Perancangan sistem dimulai dengan mengidentifikasi pengguna dan tugas-tugas representatif yang harus dilakukan dengan sistem (Lewis & Rieman, 2006).

Menurut Greenberg (2004), metode TCSD ini meliputi 4 (empat) tahapan, yaitu:

a. *Identification*, bertujuan untuk menghasilkan daftar pengguna dan *representative tasks* yang dapat dikelola dan memberikan cakupan realistis tentang siapa yang akan menggunakan sistem dan menentukan jenis tugas mereka. Untuk mencapai tujuan ini, pertama-tama harus diidentifikasi tugas-tugas yang dilakukan pengguna, kemudian dituliskan sebagai deskripsi *task* (tugas), dan kemudian memvalidasi deskripsi untuk memastikan itu telah mewakili kenyataan. Langkah-langkah tersebut dirinci di bawah ini.

1) Menemukan bagaimana orang-orang yang sebenarnya melakukan tugas mereka yang sebenarnya, karena pada prinsipnya, TCSD itu mengusahakan realisme. Mengamati dan/atau mewawancarai pengguna akhir yang sebenarnya, menghubungi pengguna saat ini atau calon pengguna, mengobservasi bagaimana mereka melakukan aktivitas tugas mereka dan mewawancarai tentang hal yang mereka lakukan.

2) Menuliskan hasil observasi dan wawancara sebagai deskripsi *task* yang baik. Deskripsi *task* ini harus mematuhi 5 (lima) kriteria yang sangat penting, yaitu:
   (a) Menggambarkan apa yang ingin dilakukan pengguna tetapi tidak mengatakan bagaimana pengguna akan melakukannya. Deskripsi tidak boleh menyertakan mekanika antarmuka apa pun tentang bagaimana tugas sebenarnya dilakukan. Ini dilakukan dengan mengidentifikasi tujuan pengguna, serta langkah-langkah konkret yang akan mereka ambil untuk mencapai tujuan tersebut, tidak peduli sistem apa yang digunakan.
   (b) Harus sangat spesifik, menjelaskan dengan tepat apa yang ingin dilakukan pengguna, termasuk item aktual yang pada akhirnya ingin dimasukkan pengguna ke dalam sistem dan informasi atau hasil apa yang diinginkan pengguna. Hal ini penting karena sistem harus menyediakan data konkret, bukan imajiner.
   (c) Harus menggambarkan pekerjaan secara lengkap. Deskripsi harus mengalir melalui semua aspek tugas, dari awal hingga akhir. Hal ini penting karena deskripsi lengkap dipakai untuk mempertimbangkan bagaimana fitur antarmuka akan saling bekerja. Deskripsi lengkap juga digunakan untuk menentukan cara memasukkan informasi dan output dilakukan melalui desain antarmuka tertentu.
   (d) Harus mengatakan siapa penggunanya dan mencerminkan minat mereka yang sebenarnya. Deskripsi harus menyebutkan pengguna yang sebenarnya, dan harus menyertakan apa yang mereka ketahui atau tidak ketahui tentang melakukan tugas dan menggunakan komputer. Hal ini penting karena keberhasilan suatu desain sangat dipengaruhi oleh pengetahuan pengguna.
   (e) Sebagai satu set, deskripsi *task* mengidentifikasi cakupan yang luas dari pengguna dan jenis *task*. Secara kolektif, deskripsi harus mengidentifikasi tipikal pengguna yang 'diharapkan', pengguna sesekali, dan pengguna yang tidak biasa. Selain itu, deskripsi harus mengidentifikasi tugas rutin yang khas, tugas yang jarang tetapi penting, dan tugas yang tidak terduga atau aneh. Ini penting karena diperlukan untuk menentukan cakupan desain

sistem, yaitu tugas dan grup pengguna mana yang harus disertakan dalam desain, dan mana yang dapat diabaikan.

3) Memvalidasi deskripsi *task*, dengan menyampaikan kembali kepada orang yang dideskripsikan. Mereka harus memvalidasi apakah deskripsi ini cukup meringkas aktivitas tugas mereka. Secara khusus, mereka harus memeriksa untuk melihat apakah kumpulan deskripsi ini sudah cukup mencakup calon pengguna akhir, apakah masing-masing *task* benar-benar mewakili apa yang dilakukan pengguna, dan apakah rincian deskripsi ini sudah realistis.

b. *User-centered requirements analysis*, adalah tahap kedua, yaitu analisis kebutuhan sistem berdasarkan hasil identifikasi untuk menentukan tipe-tipe *user* dan *task* mana saja yang butuh dimasukkan ke dalam sistem. Tahap ini dimulai dengan menentukan mengidentifikasi tipe pengguna mana yang akan didukung oleh desain. Hal ini dapat dilakukan dengan mengelompokkan pengguna. Kemudian dilanjutkan dengan menentukan *task* mana yang perlu dimasukkan ke dalam sistem, mengidentifikasi tugas-tugas yang akan ditangani secara efisien dan efektif oleh desain. Karena setiap deskripsi berpusat pada tugas, deskripsi tugas harus diurutkan dengan kriteria: (a) benar-benar harus dimasukkan; (b) harus dimasukkan jika memungkinkan; (c) bisa dimasukkan; (d) dikecualikan.

c. *Design as scenario*, merupakan tahapan penentuan desain antarmuka, proses dan data yang dibutuhkan oleh sistem baru dan pembuatan desain sistem berdasarkan *scenario* yaitu deksripsi dan analisis kebutuhan sistem yang sudah dibuat pada tahap sebelumnya. Setiap desain harus mempertimbangkan bagaimana fitur-fitur dapat bekerja sama untuk membantu pengguna menyelesaikan pekerjaan mereka yang sebenarnya. Seiring ide desain diwujudkan, rancangan sistem dapat dinilai dan antarmuka dapat dengan cepat dimodifikasi, dengan melihat seberapa baik antarmuka tersebut mendukung cerita pada rangkaian inti deskripsi pengguna/tugas. Artinya, dapat dilakukan *task-centered walkthrough* “kecil-kecilan” untuk melihat seberapa baik antarmuka yang dibuat serta fitur-fiturnya.

d. *Walkthrough evaluation*, merupakan tahapan untuk mengevaluasi hasil akhir dari desain sistem yang telah dibuat. Pada tahap ini, dibuat skenario penggunaan untuk memudahkan evaluasi. Skenario penggunaan menggabungkan desain antarmuka dengan salah satu deskripsi pengguna/tugas. Pada fase ini, dipilih salah satu skenario dan dilakukan penelusuran (*walkthrough*) yang berpusat pada tugas (*task-centered*). Dengan sebuah *walkthrough*, diceritakan kisah konkret tentang apa yang akan dilakukan dan dilihat oleh pengguna tertentu langkah demi langkah saat melakukan tugas khususnya.

2.1.5. *USE Questionnaire*

Usability adalah aspek HCI yang ditujukan untuk memastikan bahwa suatu interaksi manusia-komputer itu efektif, efisien, dan memuaskan bagi pengguna (Hartson & Pyla, 2012). Sebuah sistem harus memenuhi aturan dan prinsip *usability* (daya guna), yaitu sistem harus mudah digunakan, nyaman bagi pengguna, serta mudah dipelajari. Aturan dan prinsip yang digunakan dalam proses menghasilkan antarmuka sistem dengan *usability* yang baik harus sederhana dan fokus pada bagian tertentu (Firmansyah, 2016). *Usability* adalah tingkat kemudahan suatu sistem untuk dapat digunakan oleh pengguna dalam memenuhi kebutuhan mereka secara efektif dan efisien.

Usability suatu sistem dapat diukur dan diuji dengan mengunakan instrumen kuesioner yang menjadi tuntunan dalam pengambilan dan pengelolaan data yang berkaitan dengan efektivitas, efisiensi, dan kepuasan pengguna terhadap suatu sistem informasi. Salah satu paket kuesioner yang bisa dipakai untuk menguji *usability* adalah *USE Questionnaire*, karena kuesioner ini telah meliputi aspek-aspek dalam standar *usability*.

Useful, Satisfaction, and Ease of Use (USE) Questionnaire adalah alat usability testing yang terdiri dari 30 pernyataan yang dikelompokkan ke dalam 4 dimensi utama yaitu *usefulness, ease of use, ease of learning*, dan *satisfaction*. Kuesioner ini berbentuk poin skala penilaian Likert, yang mana pengguna berperan untuk menilai kesetujuan terhadap setiap poin pernyataan, mulai dari sangat tidak setuju hingga sangat setuju (Lund, 2016). *USE Questionnaire* dinilai lebih baik dari

pada kuesioner SUS (*System Usability Testing*) untuk pengujian usability (Purnamasari, dkk., 2021). Templat kuesioner USE dapat dilihat pada Tabel 2.1.

Tabel 2.1 Pernyataan *USE Questionnaire*

No.	Indikator	Dimensi
1.	Aplikasi ini membantu saya menjadi lebih efektif.	*Usefulness* (Kegunaan)
2.	Aplikasi ini membantu saya menjadi lebih produktif.	
3.	Aplikasi ini bermanfaat.	
4.	Aplikasi ini membuat saya lebih bisa mengontrol aktivitas dalam kehidupan saya	
5.	Aplikasi ini membuat tugas saya menjadi lebih mudah untuk diselesaikan.	
6.	Aplikasi ini menghemat waktu saya.	
7.	Aplikasi ini memenuhi kebutuhan saya.	
8.	Aplikasi ini melakukan hal yang sesuai dengan harapan saya.	
9.	Aplikasi ini mudah digunakan.	*Ease of Use* (Kemudahan Pengguna)
10.	Aplikasi ini simpel/sederhana untuk digunakan.	
11.	Aplikasi ini ramah pengguna (mudah dipahami).	
12.	Langkah-langkah penggunaan aplikasi ini sangat simpel.	
13.	Aplikasi ini dapat melakukan penyesuaian (fleksibel).	
14.	Aplikasi ini dapat digunakan tanpa upaya (*effort*) yang terlalu besar.	
15.	Saya dapat menggunakan aplikasi ini tanpa intruksi tertulis.	
16.	Saya tidak melihat adanya ke-tidak-konsistenan pada saat saya menggunakan aplikasi ini.	
17.	Baik pengguna yang jarang maupun pengguna yang sering akan menyukai aplikasi ini.	
18.	Kesalahan saya lakukan dalam aplikasi ini dapat diperbaiki dengan cepat dan mudah.	
19.	Saya selalu berhasil menggunakan aplikasi ini setiap saat.	
20.	Saya belajar menggunakan aplikasi ini dengan cepat	*Ease of Learning* (Kemudahan Mempelajari)
21.	Saya dengan mudah mengingat cara menggunakan aplikasi ini.	
22.	Saya mudah mempelajari cara menggunakan aplikasi ini.	
23.	Saya dengan cepat menjadi terampil terhadap aplikasi ini.	
24.	Saya puas dengan aplikasi ini.	*Satisfaction* (Kepuasan Pengguna)
25.	Saya akan merekomendasikan aplikasi ini kepada rekan saya.	
26.	Aplikasi ini menyenangkan untuk digunakan.	
27.	Aplikasi ini bekerja sesuai dengan yang saya inginkan.	
28.	Saya terkesan dengan aplikasi ini.	
29.	Saya merasa perlu memiliki/menggunakan aplikasi ini.	
30.	Aplikasi ini nyaman untuk digunakan.	

Sumber: Lund, 2016

2.1.6. *Unified Modelling Language (UML)*

Unified Modeling Language merupakan sebuah "bahasa" standar industri dalam hal visualisasi, perancangan, dan dokumentasi sistem perangkat lunak (Dharwiyanti dan Wahono, 2003). Karena UML juga memiliki konsep dasar yang mencakup kelas dan operasi, maka UML lebih cocok untuk perancangan perangkat lunak yang menggunakan pemrograman berorientasi objek (Hariyanto, 2004). Dalam penelitian ini, UML digunakan dalam tahapan TCSD yang ketiga yaitu *design as scenario*, untuk merancang dan memvisualisasikan alur interaksi antara pengguna dengan sistem.

a. Diagram *Use Case*

Diagram *use case* merupakan diagram UML yang memberikan gambaran fungsionalitas dari suatu sistem. Diagram *use case* terdiri dari beberapa elemen antara lain aktor, *case*, dan relasi (Hariyanto, 2004).

1) Aktor adalah pengguna yang berinteraksi dengan suatu sistem. Aktor menjelaskan peran-peran di dalam sistem. Contoh penamaan aktor seperti admin, pelanggan, pegawai, dan lain sebagainya.
2) *Case* merupakan gambaran fitur sistem.
3) Relasi merupakan keterhubungan yang terjadi antara case satu dengan *case* lainnya. Ada beberapa relasi yang terjadi pada diagram usecase antara lain: *include*, *extends*, dan *communicate*.

b. Spesifikasi *Use Case*

Spesifikasi *use case* merupakan penjabaran dari diagram *use case*. Spesifikasi dilakukan berdasarkan *case* yang ada pada diagram *use case*, terdiri dari (Hariyanto, 2004):

1) Tujuan *use case*
2) Deskripsi
3) Skenario
4) Kondisi awal
5) Kondisi akhir

2.1.7. *Prototype*

Prototype adalah salah satu bentuk dari desain yang memungkinkan pihak *stakeholder* (pihak yang berkepentingan) untuk berinteraksi dengan sistem dan dapat mengeksplorasi kesesuaian desain dengan kebutuhan. *Prototype* digunakan untuk mendiskusikan atau mengevaluasi ide bersama dengan pemangku kepentingan; merupakan alat komunikasi dan cara yang efektif bagi desainer untuk mengeksplorasi ide-ide desain (Sharp, dkk., 2019). Definisi paling dasar *prototype* adalah versi simulasi atau sampel produk akhir, yang digunakan untuk pengujian sebelum peluncuran. *Prototyping* berguna dalam proses menguji dan mengomunikasikan desain antarmuka pengguna yang menghemat waktu dan uang.

Prototype terbagi ke dalam dua jenis berbeda yaitu, *low-fidelity prototyping* dan *high-fidelity prototyping* (Sharp, dkk., 2019):

a. *Low-fidelity prototype*

Low-fidelity prototype merupakan prototipe yang tidak terlalu mirip dengan produk akhir, juga tidak menampilkan fungsionalitas. *Low-fidelity prototype* umumnya digunakan pada tahap awal pengembangan, desain konseptual. Prototipe *low-fidelity* tidak dimaksudkan untuk disimpan dan diintegrasikan ke dalam produk akhir (Sharp, dkk., 2019).

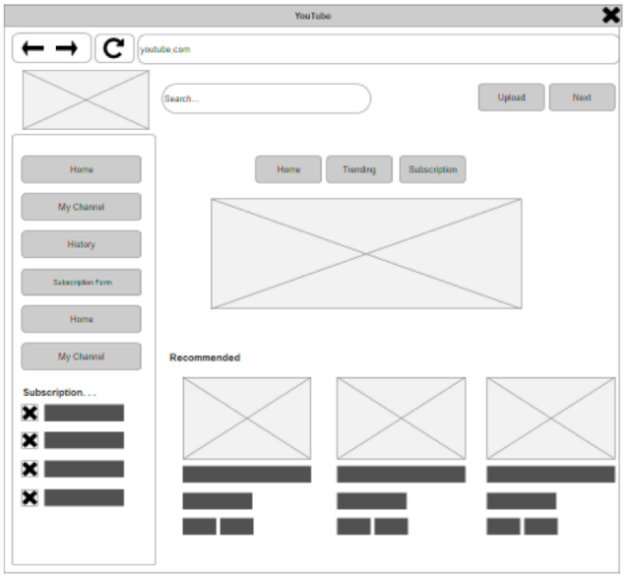
b. *High-fidelity prototype*

Prototipe *high-fidelity* terlihat lebih mirip seperti produk akhir dan biasanya menyediakan lebih banyak fungsi daripada prototipe *low-fidelity*. Misalnya, prototipe sistem perangkat lunak yang dikembangkan dengan bahasa pemrograman yang dapat dieksekusi (Sharp, dkk., 2019).

2.1.8. *Wireframe*

Menurut Segara (2019), *wireframing* adalah sebuah proses dalam membuat desain yang penting untuk membuat struktur informasi terlebih dahulu sehingga penempatan fitur-fitur informasi pada sebuah halaman website dapat menjadi lebih jelas dan mudah dipahami. *Wireframe* berfungsi untuk menentukan hierarki informasi dalam sebuah desain dan penataletakan struktur informasi agar sesuai

dengan model informasi yang diharapkan oleh pengguna (user). *Wireframe* mempermudah penyusunan konten dalam antarmuka sistem dan pengalaman pengguna. Menurut Hemm (2014), *wireframe* adalah cetak biru dasar yang menggambarkan bentuk dan fungsi inti yang ditemukan pada satu layar halaman web atau aplikasi. Contoh *wireframe* dapat dilihat pada Gambar 2.2.



Gambar 2.2 Contoh *Wireframe*

(Sumber: sis.binus.ac.id, 2020)

Proses *wireframe* berfokus pada pendefinisian bagaimana teks, grafik, dan informasi lainnya akan ditampilkan pada halaman. Palet warna yang dipakai dibatasi pada hitam dan putih, dan menggunakan garis dan bentuk sederhana untuk mewakili penempatan konten. Fokus dari *wireframing* adalah memberi petunjuk di mana elemen navigasi, teks, dan grafik akan diletakkan di layer, bukan pada tampilan grafis atau cara teks dibaca (Hemm, 2014).

Wireframe juga dapat diartikan sebagai sebuah kerangka (*framework*) sederhana yang menghubungkan komponen-komponen yang ada didalamnya. Sebuah rancangan *wireframe* secara visual berbentuk tidak lebih dari susunan kotak dan persegi yang menggambarkan sebuah elemen gambar dan atau pun susunan teks (Segara, 2019). *Wireframe* berfokus pada fungsionalitas dan struktur konten produk. Pengguna akan memfokuskan umpan balik mereka pada masalah fungsional, bukan preferensi visual.

Wireframe adalah panduan grafis yang mewakili desain UI tanpa desain visual atau elemen *branding*. Desainer atau praktisi UX menggunakannya untuk mendefinisikan item di layar berdasarkan kebutuhan pengguna. *Wireframing* juga memungkinkan pengguna untuk memahami antarmuka melalui tombol dan menu pada diagram. Hal ini membantu dalam mengkomunikasikan pengalaman dengan pengguna potensial tanpa menggunakan elemen desain grafis serta menjadi sarana dalam membuat prototipe dan mengeksplorasi ide (Deacon, 2020).

2.1.9. *Figma Prototyping*

Dalam mewujudkan aplikasi penjualan rumah ini hingga menjadi sebuah *prototype,* diperlukan *tools* untuk desain *prototype.* Pada penelitian ini, digunakan aplikasi bernama Figma. Figma merupakan aplikasi desain kolaboratif yang menyediakan *tools* desain dan fitur yang lengkap untuk perancangan UI/UX. Figma dapat digunakan untuk membuat desain *prototype* UI/UX yang interaktif (Staiano, 2022).

2.1.10. PT Mega Lavender Jaya Raya

PT Mega Lavender Jaya Raya merupakan sebuah perusahaan pengembang perumahan di Pontianak yang berdiri pada tahun 2018 dan saat ini sedang mengembangkan proyek perumahan Mega Lavender Residence. Proyek perumahan ini dibangun di lahan seluas 7 hektar dan berlokasi di Jl. Raya Desa Kapur, Kabupaten Kubu Raya. Saat ini proses pemasaran dan penjualan proyek perumahan ini masih berlangsung.

Menurut Peraturan Pemerintah Republik Indonesia Nomor 12 Tahun 2021, perumahan merupakan suatu kumpulan rumah sebagai bagian dari permukiman, baik perkotaan maupun perdesaan, yang dilengkapi dengan prasarana, sarana, dan utilitas umum sebagai hasil upaya pemenuhan rumah yang layak huni. Jenis rumah dibedakan berdasarkan pelaku pembangunan dan penghunian meliputi: rumah komersial, rumah umum, rumah swadaya. PT Mega Lavender Jaya Raya yang menjadi studi kasus dalam penelitian ini termasuk dalam penyelenggara perumahan komersial. Rumah komersial adalah jenis rumah yang diselenggarakan untuk

mendapatkan keuntungan sesuai dengan kebutuhan masyarakat (UU Nomor 1 Tahun 2011).

2.2. Tinjauan Pustaka

Daftar penelitian terdahulu yang membahas topik sejenis dengan penelitian ini dijabarkan dalam Tabel 2.2.

Tabel 2.2 Penelitian terdahulu

No.	**Penelitian Terdahulu**	
1.	Judul	*Analysis and Design of User Interface and User Experience (UI/UX) ECommerce Website PT. Pentasada Andalan Kelola Using Task Centered System Design (TCSD) Method*
	Tahun	2019
	Peneliti	Zhafirah Indira Paramarini Hardianto dan Karmilasari
	Metode	*Task Centered System Design (TCSD)* dan *USE Questionnaire*
	Hasil	Hasil skor uji dimensi *satisfaction* dengan nilai 87,5% (sangat memuaskan). Hasil skor pengujian dimensi *ease of use* adalah 93,6% (sangat mudah digunakan).
	Perbedaan Penelitian	Penelitan ini melakukan analisis UI/UX website dengan studi kasus *ecommerce* dan merancang UI/UX dengan menggunakan metode *Task Centered System Design (TCSD).* Perbedaannya, penelitian ini menggunakan *storyboard* pada tahap *user-centered requirement analysis*.
2.	Judul	*Design Of Property Sales Information System PT. Quality Property Indonesia*
	Tahun	2020
	Peneliti	Ferry Sudarto, Mulyati, Eka Purnama Harahap, dan Fira Arbaimaniar Nurul
	Metode	*SWOT analysis, UML*
	Hasil	Sebuah sistem informasi penjuakan properti berbasis web yang menyediakan layanan dan informasi kepada pelanggan serta membantu proses pencatatan data pelanggan, data unit rumah, dan laporan penjualan.
	Perbedaan Penelitian	Penelitian ini merancang sebuah sistem informasi penjualan properti berbasis web, namun tanpa menerapkan metode perancangan UI/UX.

Tabel 2.3 Penelitian terdahulu (lanjutan)

No.	Penelitian Terdahulu	
3.	Judul	Penerapan Metode TCSD Untuk Analisis dan Perancangan UI/UX pada E-Learning SMAN 1 Sidoarjo
	Tahun	2021
	Peneliti	Muhamad Nizar Taufani
	Metode	*Task Centered System Design (TCSD),* Evaluasi Heuristik
	Hasil	Metode TCSD mampu mengidentifikasi kebutuhan task pengguna *elearning* dan memberikan rekomendasi UI yang mendukung pembelajaran daring. Hasil dari *wireframe testing* mendapatkan nilai indeks rata-rata 84% yang berarti setiap halaman sudah sesuai *task* setiap *user*.
	Perbedaan Penelitian	Penelitian ini menganalisis dan membuat desain UI/UX menggunakan metode TCSD dengan studi kasus sistem *e-learning* berbasis web. Perbedaannya, penelitian ini menggunakan metode evaluasi heuristik untuk tahap pengujian.

Berdasarkan tinjauan pustaka pada penelitian terdahulu, terdapat beberapa perbedaan dan juga persamaan pada dalam penelitian yang akan dilaksanakan, yaitu penelitian ini mengembangkan sebuah desain prototipe aplikasi penjualan rumah dengan menerapkan salah satu metode perancangan UI/UX , *Task-Centered System Design*, menggunakan UML pada proses perancangan alur sistem, dan menggunakan *USE Questionnaire* untuk pengujian hasil rancangan.