Design Antarmuka Pengguna

Program Studi S1 Sistem Informasi Stikom Surabaya

User Interface Design

User Interface (UI) **Design** focuses on anticipating **what users might need to do**

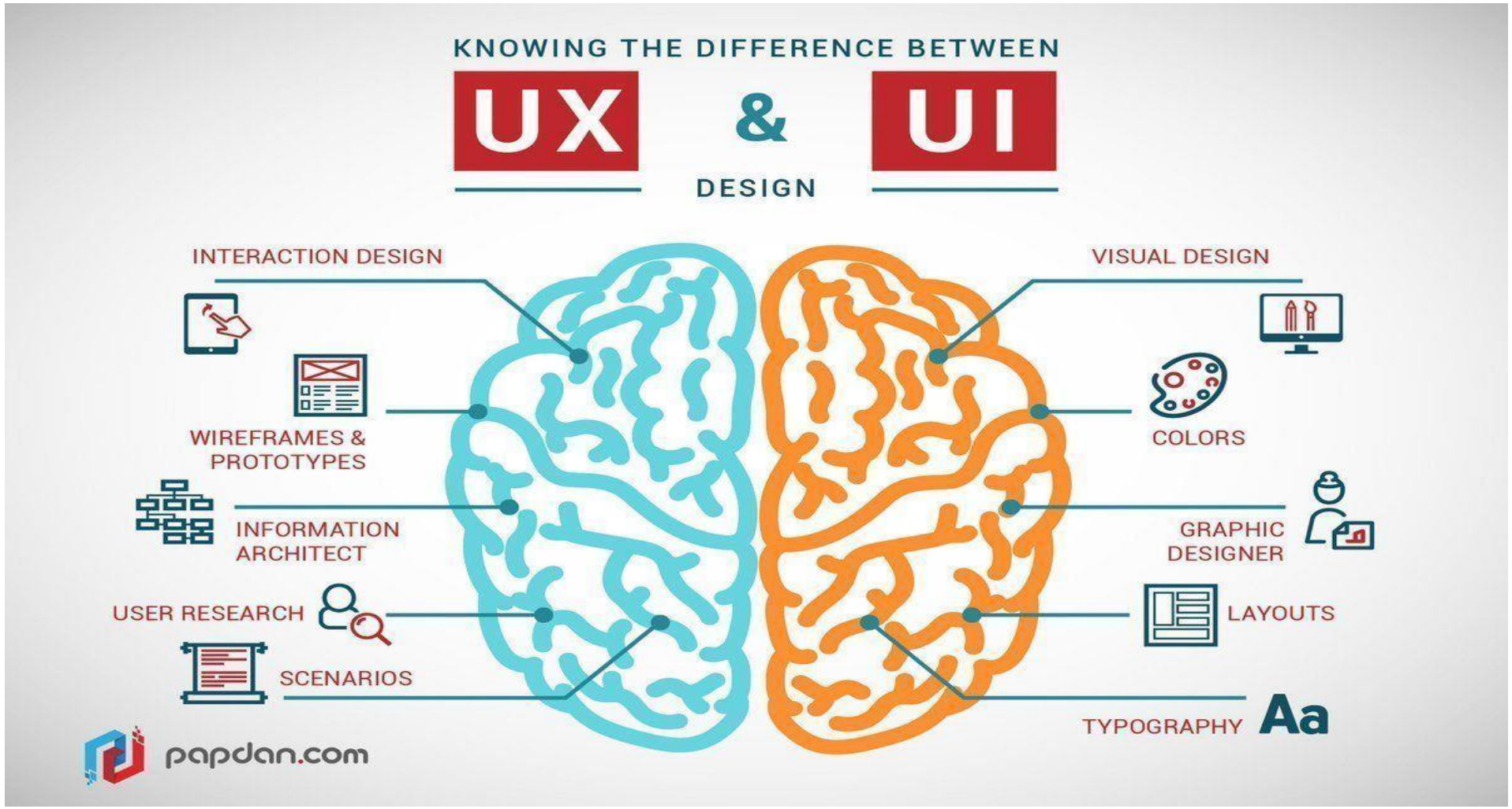
Design of websites, computers, appliances, machines, mobile communication devices, and software applications with the focus on the **user's experience (UX)** and **interaction**.

UI brings together concepts from **interaction design, visual design, and information architecture**

UI design is process talking the way a machine works and translating it into way a person thinks

Everythings steems from knowing your users, including understanding their goals, skills, preferences, and tendencies

User Experience (UX) vs User Interface (UI)



General Principles Of UI Design

1. User Compatibility
2. Product Compatibility
3. Task Compatibility
4. Work Flow Compatibility
5. Consistency
6. Familiarity
7. Simplicity
8. Direct Manipulation
9. Control
10. WYSIWYG
11. Flexibility
12. Responsiveness
13. Invisible Technology
14. Robustness
15. Protection
16. Ease of Learning
17. Ease of use

User Compatibility

- Seorang perancang sistem harus benar-benar paham tentang pengetahuan, cara berpikir dan cara menerima informasi dari *user*sehingga sistem yang nantinya akan digunakan oleh *user*dapat membuat *user*lebih produktif.
- Oleh karena itu sebuah software seolah-olah mengenal usernya, mengenal karakteristik usernya, dari sifat sampai kebiasaan manusia secara umum.
- Hal tersebut harus terpikirkan oleh desainer dan tidak dianjurkan merancang antarmuka dengan didasarkan pada dirinya sendiri
- Survey adalah hal yang paling tepat

Product Compatibility

- Seringkali sebuah aplikasi menghasilkan hasil yang berbeda dengan sistem manual atau sistem yang ada.
- Hal tersebut sangat tidak diharapkan dari perusahaan karena dengan adanya aplikasi software diharapkan dapat menjaga produk yang dihasilkan dan dihasilkan produk yang jauh lebih baik.
- **Contoh** : aplikasi sistem melalui antarmuka diharapkan menghasilkan report/laporan serta informasi yang detail dan akurat dibandingkan dengan sistem manual.

Task Compatibility

- Rancanglah *interface* sistem sesuai dengan tugas dari *user*, jangan sampai *user* kesulitan untuk menggunakannya, karena hal ini dapat menyebabkan aplikasi yang kita buat tidak akan terpakai dan akhirnya tidak dapat membantu pekerjaan / tugas *user*
- Sebisa mungkin user tidak dihadapkan dengan kondisi memilih dan berpikir, tapi user dihadapkan dengan pilihan yang mudah dan proses berpikir dari tugas-tugas user dipindahkan dalam aplikasi melalui antarmuka.
- Contoh : User hanya klik setup, tekan tombol next, next, next, finish, ok untuk menginstal suatu sotfware.

Work Flow Compatibility

- Sebuah aplikasi sistem sudah pasti mengapdopsi sistem manualnya dan didalamnya tentunya terdapat urutan kerja dalam menyelesaikan pekerjaan.
- Dalam sebuah aplikasi, software engineer harus memikirkan berbagai runutan-rununtan pekerjaan yang ada pada sebuah sistem.
- Jangan sampai user mengalami kesulitan dalam menyelesaikan pekerjaannya karena user mengalami kebingungan ketika urutan pekerjaan yang ada pada sistem manual tidak ditemukan pada software yang dihadapinya.
- Selain itu user jangan dibingungkan dengan pilihan-pilihan menu yang terlalu banyak dan semestinya menu-menu merupakan urutan dari runutan pekerjaan.
- Sehingga dengan workflow compatibility dapat membantu seorang user dalam mempercepat pekerjaannya.

Consistency

- Prinsip ini sudah jelas, bahwa sistem harus konsisten terhadap fungsionalitas / kegunaan dari sistem tersebut. Contoh sederhananya adalah ketika *user*menekan tombol "save" maka proses yang terjadi adalah penyimpanan bukan hapus data.
- Hal itu didasarkan pada karakteristik manusia yang mempunyai pemikiran yang menggunakan analogi serta kemampuan manusia dalam hal memprediksi.
- Contoh : keseragaman tampilan toolbar pada Word, Excell, PowerPoint, Access hampir sama.

Familiarity

- Sifat manusia mudah mengingat dengan hal-hal yang sudah sering dilihatnya/didapatkannya. Secara singkat disebut dengan familiar.
- Antarmuka sebisa mungkin didesain sesuai dengan antarmuka pada umumnya, dari segi tata letak, model, dsb.
- Hal ini dapat membantu user cepat berinteraksi dengan sisem melalui antarmuka yang familiar bagi user.
- Gunakanlah konsep, terminologi dan pengaturannya yang mudah dipahami oleh *user* Seperti ikon atau gambar "Recycle Bin" pada Sistem Operasi Windows, ini membuktikan bahwa fokus*user*terhadap gambar tersebut adalah *file-file*yang sudah dihapus sebelumnya.

Simplicity

- Kesederhanaan perlu diperhatikan pada saat membangun antarmuka.
- Tidak selamanya antarmuka yang memiliki menu banyak adalah antarmuka yang baik.
- Kesederhanaan disini lebih berarti sebagai hal yang ringkas dan tidak terlalu berbelit.
- User akan merasa jengah dan bosan jika pernyataan, pertanyaan dan menu bahkan informasi yang dihasilkan terlalu panjang dan berbelit.
- User lebih menyukai hal-hal yang bersifat sederhana tetapi mempunyai kekuatan/bobot.

Direct Manipulation

- User berharap aplikasi yang dihadapinya mempunyai media atau tools yang dapat digunakan untuk melakukan perubahan pada antarmuka tersebut.
- User ingin sekali aplikasi yang dihadapannya bisa disesuaikan dengan kebutuhan, sifat dan karakteristik user tersebut. Selain itu, sifat dari user yang suka merubah atau mempunyai rasa bosan.
- Contoh : tampilan warna sesuai keinginan (misal pink) pada window bisa dirubah melalui desktop properties, tampilan skin winamp bisa dirubah, dll.

Control

- Prinsip control ini berkenaan dengan sifat user yang mempunyai tingkat konsentrasi yang berubah-ubah. Hal itu akan sangat mengganggu proses berjalannya sistem.
- Kejadian salah ketik atau salah entry merupakan hal yang biasa bagi seorang user. Akan tetapi hal itu akan dapat mengganggu sistem dan akan berakibat sangat fatal karena salah memasukkan data 1 digit/1 karakter saja informasi yang dihasilkan sangat dimungkinkan salah.
- Oleh karena itu software engineer haruslah merancang suatu kondisi yang mampu mengatasi dan menanggulangi hal-hal seperti itu.
- Contoh : “illegal command”, “can’t recognize input” sebagai portal jika terjadi kesalahan.

WYSIWYG

- WYSIWYG = what you see is what you get = apa yang didapat adalah apa yang dilihatnya.
- Contoh : apa yang tercetak di printer merupakan informasi yang terkumpul dari data-data yang terlihat di layar monitor pada saat mencari data.
- Hal ini juga perlu menjadi perhatian software engineer pada saat membangun antarmuka.
- Informasi yang dicari/diinginkan harus sesuai dengan usaha dari user pada saat mencari data dan juga harus sesuai dengan data yang ada pada aplikasi sistem (software).
- Jika sistem mempunyai informasi yang lebih dari yang diinginkan user, hendaknya dibuat pilihan (optional) sesuai dengan keinginan user. Bisa jadi yang berlebihan itu justru tidak diinginkan user.
- Yang mendasar disini adalah harus sesuai dengan kemauan dan pilihan dari user.

Flexibility

- Fleksibel merupakan bentuk dari dari solusi pada saat menyelesaikan masalah.
- Prinsip ini merupakan prinsip yang sangat penting bagi *user*dengan keterbatasan fisik. Ini berarti mengijinkan banyak kontrol dari *user*yang mendukung untuk menggunakan aplikasi yang kita rancang dan mampu mengakomodir kemampuan *user*yang lain.
- Seperti aplikasi yang dapat didukung oleh perangkat lain (*mouse, keyboard, joystick, trackball)*

Responsiveness

- Sistem harus selalu merespon dengan cepat apa yang di *inputk*an oleh *user* Seperti menampilkan *Progress Bar*
- Selain teknologi komputer semakin maju sesuai dengan tuntutan kebutuhan manusia, software yang dibangun pun harus mempunyai reaksi tanggap yang cepat. Hal ini didasari pada sifat manusia yang semakin dinamis / tidak mau menunggu.

Invisible Technology

- Secara umum, user mempunyai keingintahuan sebuah kecanggihan dari aplikasi yang digunakannya. Untuk itu aplikasi yang dibuat hendaknya mempunyai kelebihan yang tersembunyi. Bisa saja kelebihan itu berhubungan dengan sistem yang melingkupinya atau bisa saja kecanggihan atau kelebihan itu tidak ada hubungannya.
- Contoh : sebuah aplikasi mempunyai voice recognize sebagai media inputan, pengolah kata yang dilengkapi dengan language translator

Robustness

- Interaksi manusia dan komputer (pembangunan antarmuka) yang baik dapat berupa frase-frase menu atau error handling yang sopan.
- Kata yang digunakan harus dalam kondisi bersahabat sehingga nuansa user friendly akan dapat dirasakan oleh user selama menggunakan sistem .
- Contoh yang kurang baik : YOU FALSE !!!, BAD FILES !!!, FLOPPY ERROR, dsb. Akan lebih baik jika BAD COMMAND OR FILES NAMES, DISK DRIVE NOT READY,dll.

Protection

- Suasana nyaman perlu diciptakan oleh software engineer di antarmuka yang dibangunnya.
- Nyaman disini adalah suasana dimana user akan betah dan tidak menemui suasana kacau ketika user salah memasukkan data atau salah eksekusi.
- Seorang user akan tetap merasa nyaman ketika dia melakukan kesalahan, misal ketika user melakukan deleting atau menghapus files tanpa sengaja tidaklah menjadi kekacauan yang berarti karena misal ada recovery tools seperti undo, recycle bin, dll atau "are you sure…."
- Proteksi disini lebih menjaga kenyamanan user ketika menggunakan aplikasi sistem khususnya data-data berupa file.

Ease of Learning and Ease of Use

- Kemudahan dalam mengoperasikan software hanya dengan memandangi atau belajar beberapa jam saja.
- Kemudahan dalam memahami icon, menu-menu, alur data software, dsb.
- Sesudah mempelajari, user dengan mudah dan cepat menggunakan software tersebut. Jika sudah memahami tentunya akan membantu proses menjalankan sistem dengan cepat dan baik.

Desain Output :

- Internal output
- Eksternal Output
- Turnaround Output

Desain Output :

- **Internal output** digunakan untuk para pemilik dan pengguna sistem dalam sebuah perusahaan. Output internal mendukung operasi bisnis sehari-hari atau pengawasan manajemen dan pengambilan keputusan.
- Eksternal Output
- Turnaround Output

Desain Output :

- **Internal output** digunakan untuk para pemilik dan pengguna sistem dalam sebuah perusahaan. Output internal mendukung operasi bisnis sehari-hari atau pengawasan manajemen dan pengambilan keputusan.
  - Detailed Report,
  - Summary Report,
  - Exception Report
- Eksternal Output
- Turnaround Output

Desain Output :

- **Internal output** digunakan untuk para pemilik dan pengguna sistem dalam sebuah perusahaan. Output internal mendukung operasi bisnis sehari-hari atau pengawasan manajemen dan pengambilan keputusan.
  - Detailed Report,
  - Summary Report,
  - Exception Report
- **Eksternal Output** bersifat keluar organisasi. Output ini ditujukan kepada konsumen, pemasok, mitra bisnis dan badan pemerinta han. Output eksternal menyimpulkan dan melaporkan transaksi bisnis. Contoh faktur, nota pembelian, jadwal kursus, tiket pesawat, tagihan telepon dan lain sebagainya.
- **Turnaround Output**

Desain Output :

- **Internal output** digunakan untuk para pemilik dan pengguna sistem dalam sebuah perusahaan. Output internal mendukung operasi bisnis sehari-hari atau pengawasan manajemen dan pengambilan keputusan.
  - Detailed Report,
  - Summary Report,
  - Exception Report
- **Eksternal Output** bersifat keluar organisasi. Output ini ditujukan kepada konsumen, pemasok, mitra bisnis dan badan pemerinta han. Output eksternal menyimpulkan dan melaporkan transaksi bisnis. Contoh faktur, nota pembelian, jadwal kursus, tiket pesawat, tagihan telepon dan lain sebagainya.
- **Turnaround Output** adalah output eksternal yang akhirnya masuk kembali ke dalam sistem sebagai input. Contoh tagihan telepon yang hasil pembayaran pelanggan menjadi inputnya.

DESAIN INPUT

prinsip-prinsip desain input yang harus diikuti:

- Dapatkan hanya data variabel, jangan memasukkan data konstan. Misalnya pada input sales order, maka kita membutuhkan part number dari seluruh bagian yang akan dipesan, tetapi kita tidak perlu menginput part descriptions untuk bagian-bagian tersebut.
- Jangan meng-capture data yang dapat dikalkulasi atau dihitung dengan menggunakan program komputer.
- Gunakan kode untuk atribut yang tepat.

Kontrol Internal – Data Editing untuk Input

- Jumlah input harus diawasi
- Data harus valid. Terdapat dua tipe kesalahan yg dapat terjadi pada data : Kesalahan data entry dan penyimpanan data invalid.

Proses Desain Input

1. Mengidentifikasikan input system dan memberikan prasyarat logika
2. Memilih control GUI yg sesuai
3. Mendesign, memvalidasi dan mengetes input
4. Mendesign source document

Masalah dalam DAP

- Terlalu banyak jargon
- Design tidak jelas
- Tidak mampu membedakan antara tindakan pilihan
- Pendekatan pemesacahan masalah yang tidak konsisten
- Design tidak konsisten

Solusi:

- Pahami pengguna dan tugas mereka
- Libatkan pengguna dalam design antarmuka
- Uji Sistem pd pengguna actual
- Lakukan design Iterative

Petunjuk Human Engineering

- Pengguna tahu yg hrs dilakukan selanjutnya
- Pesan, perintah atau informasi harus jelas
- Gunakan atribut tampilan yang hemat
- Nilai field yg salah dan pembenaran nilai hrs ditentukan
- Antisipasi kesalahan. Misalnya “Data akan dihapus?”
- Jika error, pengguna hrs nya tidak bisa lanjut
- Jika pengguna melakukan hal yg parah, keyboard hrs terkunci

Tone dan Terminologi Dialog

1. Jangan menggunakan Jargon komputer
2. Gunakan istilah yg sederhana
3. Penggunaan terminology hrs konsisten
4. Perintah: gunakan kata kerja yg tepat
5. Hindari penggunaan singkatan