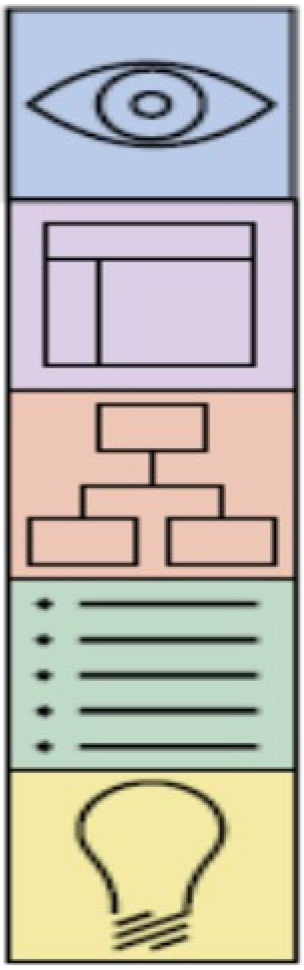
Surface Plane (Looks)
Skeleton Plane (Layout)
Structure Plane (Flow)
Scope Plane (What)
Strategy Plane (Why)



Strategy Plane — “Why are we building this?”

This is the **foundation** of UX.
It defines the **goals** of the product — both for the **business** and for the **user**.

Purpose:

- Understand **what the organization wants** to achieve.
- Understand **what users need** from the product.

Key Components:

- **Product Objectives** (Business goals)
- **User Needs** (Problems users face)
- **Brand Identity** (How users perceive you)
- **Success Metrics** (How you measure success)

Example:

For a **food delivery app**:

- Business goal: Increase daily orders
- User need: Quick and easy ordering
- Success metric: Reduce order time from 5 min → 2 min

Scope Plane — “What will it do?”

After defining the *why*, we define the *what*.
This plane lists **all the features and content** your product will have.

Purpose:

- Turn strategy into **clear requirements**.
- Decide **which features** to include and which to skip (for now).

Key Components:

- **Functional Specifications** → What the system will *do*
- **Content Requirements** → What the system will *contain*

Example:

In our food app:

- Features: search restaurants, order tracking, in-app payment
- Not included (yet): dine-in reservations, reviews

Structure Plane — “How will it work?”

Now that we know what’s inside, we design **how everything connects**.
It’s about the **organization** and **flow** of information or actions.

Two main areas:

- **Interaction Design** → How users perform actions (buttons, clicks, gestures)
- **Information Architecture** → How information is arranged (menus, categories)

Example:

- When the user taps “Order History”, it should show past orders in a clear list.
- Categories like “Pizza ”, “Burgers ”, “Desserts ” are structured for easy browsing.

Skeleton Plane — “How will it look on screen?”

This is where we design **layout and navigation** — the “blueprint” of screens.
We create **wireframes** (simple sketches of page layouts).

Key Focus:

- **Interface Design** – placement of buttons, input boxes, etc.
- **Navigation Design** – how users move between pages
- **Information Design** – presenting content clearly

Example:

- The “Order Now” button should be large and at the bottom for thumb reach
- The logo stays on top for brand visibility

Surface Plane — “What will it look and feel like?”

Finally, the **visual design layer** — the *appearance* of the product.
This is what the user actually sees and interacts with.

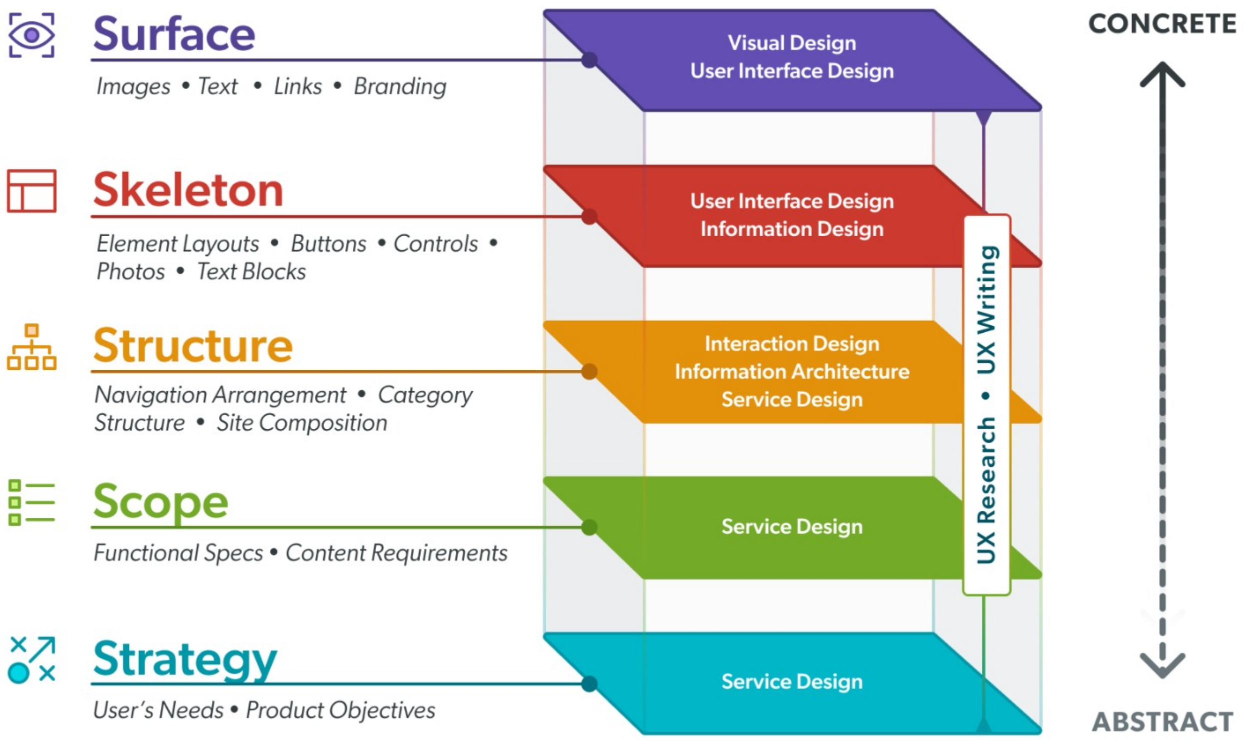
Focus Areas:

- **Visual Design / Sensory Design** (colors, fonts, icons, contrast)
- **Consistency** (uniform look across screens)
- **Accessibility** (readable text, color contrast)

Example:

- Using a bright orange "Place Order" button for visibility
- Smooth animations during checkout
- Consistent typography and colors

Plane	Focus	Key Question	Example (Food App)
Strategy	Why	What do users & business want?	Fast ordering for students
Scope	What	What features/content will we include?	Search, order tracking
Structure	How (flow)	How do users move through it?	Browse → Order → Pay
Skeleton	Layout	How will screens be arranged?	Button, menu placement
Surface	Look & Feel	What will it look like?	Colors, icons, typography



The Strategy Plane — "Why are we building this?"

Simple Idea:

Before you design *anything*, you must know **why the product exists** and **who it is for**.
That's what the **Strategy Plane** defines — it's the *foundation* for all UX design decisions.

It answers two main questions:

1. **What do we (the organization) want to achieve?**

2. **What do users need and expect from us?**

When both align — you get a successful user experience.

Example:

Let’s take our food delivery app

Aspect	Example
Business Goal	Increase online orders by 50% this year
User Need	Easily find affordable food and track delivery
Strategy Outcome	Create an app that’s fast, reliable, and simple for students

So, if the company wants **more sales** and users want **less waiting time**,
→ UX strategy focuses on **speed + simplicity.**

Key Components of the Strategy Plane

Product Objectives

These are the *goals* the product must achieve for the organization.

In simple terms:

“What do we want our product to do for our company?”

Examples:

- A bank app → Reduce branch visits by 40%.
- A college website → Increase admission inquiries.
- A news app → Grow subscribers by 25%.

These objectives keep the team focused on measurable success.

Business Goals

These are **strategic, internal** goals that align with the company’s mission and revenue.

They describe *why* the company wants to make this product.

Examples:

- Make ₹1 crore in online sales this year.
- Launch into 10 new cities by next quarter.
- Build brand reputation among tech users.

Business goals = Company’s big-picture targets.

Brand Identity

Brand identity = **how users perceive your company**
It’s not just your logo or colors — it’s the *feeling* users get when interacting with your product.

Example:

Brand	User Perception
Apple	Premium, creative
Swiggy	Fast, friendly, reliable
Government Portal	Trustworthy, official

Consistency in color, tone, typography, and message strengthens the user’s trust.

Success Metrics (a.k.a. KPIs)

How will you know your product is successful?
That’s what **Success Metrics** tell you — they are measurable indicators of your goals.

Examples:

Goal	Success Metric
Increase orders	No. of orders/day
Improve usability	Drop in user complaints
Build loyalty	Repeat customer rate

“If you can measure it, you can improve it.”

User Needs

Users’ needs are the **problems they want solved**.
To build great UX, you must see things from their eyes

Example:

For a student using a food app:

- Need: “I want cheap food near my hostel.”
- Pain point: “Delivery is slow; I can’t track it.”
- Expectation: “Show real-time tracking and student discounts.”

UX designers focus on *what makes users’ lives easier and happier.*

User Segmentation

We can’t design for everyone — so we divide users into **groups (segments)** based on shared traits.

Segmentation Criteria:

- **Demographic:** age, gender, income, education
- **Psychographic:** interests, attitudes, lifestyle
- **Behavioral:** how often they use apps, tech comfort

- **Geographic:** location-based habits

Example:

Segment	Example
Students	Use offers & fast delivery
Working Professionals	Prefer office lunch combos
Families	Order on weekends in bulk

This helps design features and content for *each type* of user.

Usability & User Research

Before finalizing the product idea, we must **research users** to understand:

- What they do
- How they behave
- What frustrates them

Common Research Methods:

Method	What it does
Surveys	Ask user opinions in bulk
Interviews	Get detailed insights
Observation / Field Study	Watch real users use the product
User Testing	Let users test your prototype
Task Analysis	Study steps users take to complete a task

"Don't assume — research!"

Creating Personas

After research, UX designers build **Personas** —
fictional characters that represent real user groups .

Each persona includes:

- Name and photo
- Background
- Goals and frustrations
- Preferred features

Example:

Name: Priya, 21, College Student
Goal: Wants cheap food fast
Pain Point: Slow checkout, unclear delivery time
Preferred Device: Mobile
Motivation: Save time between classes

Designers keep these personas visible during development to ensure *user-centered decisions.*

Concept	Meaning	Example
Product Objectives	What company wants	“Increase orders”
Business Goals	Big internal targets	“Grow market 30%”
Brand Identity	Company’s personality	“Friendly & fast”
Success Metrics	How success is measured	“More repeat users”
User Needs	What users want	“Quick & cheap food”
User Segmentation	Divide audience	“Students, professionals”
User Research	Study users	Surveys, testing
Personas	Fictional user models	“Priya, 21, foodie student”

The Scope Plane — “What will it do?”

Simple Meaning:

The **Scope Plane** takes the *big goals* from the **Strategy Plane** (why we’re building it) and turns them into a *clear list of features and content* (what we’re building).

Think of it as writing your product’s “to-do list.”

Example (Food Delivery App):

Type	Example
Functional Scope	“Users can search restaurants, add to cart, and track delivery.”
Content Scope	“The app will show restaurant menus, photos, and user reviews.”

So, **scope = what your app will contain and do.**

Why Defining the Scope Matters

Defining scope is like planning before you build a house —
you decide which rooms (features) to include and which to skip for now.

Without a clear scope:

- Teams get confused about priorities
- Developers keep adding “cool” features that don’t fit goals
- Deadlines and budgets go out of control

Two Main Sides of the Scope Plane

The scope plane has two key parts:

1. **Functional Specifications**

2. **Content Requirements**

Let’s explore both

Functional Specifications

This defines **what the system will do —**
the features, interactions, and behaviors.

It’s like a **feature blueprint** for developers.

Example:

For our food app:

- Login & registration
- Search for restaurants
- Order tracking
- Payment gateway integration
- Table booking (maybe in future)

Each feature is described *clearly* so that all team members know what’s included.

Why It’s Important:

- It removes **ambiguity** — everyone knows what’s being built.
- Helps evaluate **new feature ideas** later (“Does it fit the scope?”).
- Prevents **scope creep** — adding too many features mid-project.

Example in Simple Terms:

If you’re building a **music app**,
Functional specs may include:

- Play/pause songs
- Add to playlist
- Download offline
- Show lyrics

Not included (yet):

- Live radio streaming
- AI-based recommendations

This gives the team a clear **feature boundary**.

Content Requirements

While functional specs talk about what the system *does*,
content requirements describe what the system *shows* —
the text, images, videos, or documents that appear inside it.

Example:

For a **food delivery app**:

- Menu items with prices
- Restaurant descriptions
- Food photos
- Customer reviews
- Help center FAQs

Each piece of content is **planned, created, and maintained** intentionally.

Key Points:

- Identify **types of content** (text, video, images, etc.)
- Estimate **size/quantity** (e.g., 500 product photos, 300-word descriptions)
- Assign **responsibility** (who creates and updates it)
- Decide **format and purpose** (don't confuse *how* it looks with *why* it's needed)

Example of Content Planning:

Content Type	Purpose	Source
Menu Photos	Attract customers	Restaurant uploads
Delivery Icons	Guide users visually	Design team
FAQs	Support users	Customer care

The Structure Plane — "How will it work?"

Simple Meaning

After we decide **what** the product will contain (Scope Plane),
the **Structure Plane** decides **how users will move through it** and **how the system will behave**.

It answers:

> “How will users get from Point A → Point B logically and smoothly?”

This plane focuses on **flow**, **organization**, and **behavior** of the system.

The Structure Plane Has Two Parts:

Interaction Design (for functionality-heavy products)

Information Architecture (IA) (for content-heavy products)

Let’s break them down

Interaction Design — “How will the system react to the user?”

Interaction Design defines:

- What happens when the user **clicks**, **swipes**, **scrolls**, or **submits**
- The **steps** the user takes to finish tasks
- How the system **responds** (success, error, loading, etc.)

Goal:

Make every action *predictable, smooth, efficient,* and *meaningful*.

Example (Food Delivery App):

User action → System reaction

User Action	System Behavior
Tap “Add to Cart”	Item slides into cart, quantity updates
Tap “Track Order”	Show live delivery map
Enter wrong OTP	Show clear error message + retry option
Payment successful	Show confirmation animation

This makes the app feel **logical** and **easy to use**.

Interaction Design Fixes:

- Confusing button behavior
- Too many steps to complete an action

- Poor error messages (e.g., “Null input exception”)
- Missing feedback (nothing happens when user taps)

Information Architecture (IA) — “How is content organized?”

IA focuses on arranging **content** so users can *find what they need quickly*.

Goal:

Organize information in a clear, understandable structure like a library .

Example (E-commerce Website):

Good IA organizes items into:

- Electronics
    - Mobiles
    - Laptops
    - Accessories
- Clothing
    - Men
    - Women
    - Kids

If the categories are messy, users get lost → **bad UX**.

IA Helps With:

- Menus and submenus
- Categories and labels
- Search structure
- Content hierarchy

Summary Table for the Structure Plane

Concept	Meaning	Example
Interaction Design	How users act & how system reacts	Add to cart → update instantly

Concept	Meaning	Example
Information Architecture	How content is organized	Categories like Pizza / Burgers

Why the Structure Plane Is Important

Because it decides:

- How **easy or difficult** it is for users to complete tasks
- Whether users will **stay or leave** your app
- How **logical and predictable** the journey feels

Even if your UI looks beautiful, if the structure is confusing → user quits!

Super Simple Analogy

Think of **Structure Plane** as the **floor plan of a house**:

- Scope Plane says: "We need 3 bedrooms, 1 kitchen."
- Structure Plane says: "Bedroom 1 here, kitchen next to hall, bathroom in between."

Without a good floor plan → people get lost inside the house.

The Skeleton Plane — "How will everything be arranged on the screen?"

Simple Meaning

The **Skeleton Plane** turns the abstract structure (flows, IA, interactions) into a **layout blueprint** for every screen.

It answers:

> "Where should each button, menu, image, and text block be placed?"

This is where **wireframes** are created — simple black-and-white sketches showing *placement*, not colors or style.

The Skeleton Plane Has 3 Main Parts

Interface Design

Navigation Design

Information Design

Let’s break them down

Interface Design — “How do users interact with the system?”

This defines the **arrangement of functional elements**:

- Buttons
- Forms
- Sliders
- Input fields
- Icons

Goal:

Make interactions easy, fast, and user-friendly.

Example:

On a **checkout page**:

- “Place Order” button must be big and visible
- Address fields must be organized clearly
- Payment options must be grouped logically

Bad interface design = tiny buttons, confusing layout, misplaced features.

Navigation Design — “How will users move around?”

Navigation design creates:

- Menus

- Tabs
- Sidebars
- Breadcrumbs
- Links

Goal:
Help users understand *where they are*, *where they can go*, and *how to get back*.

Example:

For a food app:

- Bottom navigation: Home | Search | Orders | Profile
- Category links on top: Pizza | Biryani | Desserts
- Back button must always be visible

Good navigation = users never feel lost.

Information Design — “How should information be presented for clear understanding?”

This decides:

- What information appears first
- How data is grouped
- How text, icons, and visuals support clarity

Goal:
Present information in a way that is **clear**, **logical**, and **quick to understand**.

Example:

On a product page:

- Price should be large and bold
- Image on top
- Description below

- Reviews at the bottom

Good information design reduces cognitive load for the user.

Putting It All Together (Simple Example)

Let's imagine the **Skeleton Plane** for a “Product Page”.

Structure Plane (previous layer):

- User → Category → Product → Add to cart flow is decided.

Skeleton Plane (current layer):

Now we decide layout:

```
[ Product Image ]
[ Product Title ]
[ Price ] [ Ratings ]
[ Description ]
[ Add to Cart Button ]
```

No colors, no styling — just placement and hierarchy.

Why the Skeleton Plane Is Important

Because:

- It prepares designers and developers for **what goes where**
- It prevents confusion later during UI design
- It ensures usability before applying colors and styles
- It supports clear, consistent layout across all screens

Bad skeleton → beautiful design with horrible usability

Good skeleton → strong foundation for a smooth UI

Simple Analogy

Think of the Skeleton Plane like a **blueprint of a house**:

- Where is the door?
- Where are windows?
- Where is the kitchen?

The architect doesn’t paint colors here — just the layout.

That’s exactly what wireframes do for apps.

The Surface Plane — “What will it look and feel like?”

Simple Meaning

The **Surface Plane** is the final layer where the user actually *sees and interacts* with the product.

Everything below it (strategy → scope → structure → skeleton) becomes **visual design** here.

It answers:

> “What should the product look like visually so the experience is clear, consistent, and enjoyable?”

This includes:

- Colors
- Typography (fonts)
- Buttons
- Spacing
- Icons
- Images
- Contrast
- Visual consistency

It defines the **sensory experience** of the product.

Key Parts of the Surface Plane

Sensory Design

Making Sense of the Senses

Contrast & Uniformity

Internal & External Consistency

Color Palettes & Typography

Design Comps & Style Guides

Let’s break them down in simple words

Sensory Design — “Appealing to the senses”

This includes:

- How things **look** (visual design)
- How they **feel** (micro-animations, feedback)
- Sound cues (optional in apps)

The primary goal:

> Make the user’s interaction *pleasant and intuitive*.

Examples:

- A **green** button means success (Place Order).
- A **red** color indicates errors.
- A vibrating animation shows wrong OTP.
- High-quality food images attract buyers.

Making Sense of the Senses

This means using visuals to *communicate meaning*.

Principle:

> Users should immediately understand what an element means by looking at it.

Examples:

- A trash bin icon = delete
- A heart icon = save/favorite
- Bold text = important
- Light text = secondary info

Users don’t read manuals — visual cues must guide them naturally.

Contrast & Uniformity

Contrast

Used to highlight important elements.

Example:

- Bright orange “Order Now” button stands out.
- Dark text on white for readability.

Uniformity

Consistency of design across screens.

Example:

- All headings use the same font size
- All primary buttons use the same color
- Margins and spacing follow a pattern

Uniformity improves professionalism and predictability.

Internal & External Consistency

Internal Consistency

Consistency *within* the product.

Example:

- All error messages look the same
- Navigation stays in the same position across screens

External Consistency

Consistency with *other apps or platforms* users already know.

Example:

- A “hamburger menu” icon works similarly everywhere
- A “shopping cart” icon universally means checkout

This reduces learning time and boosts usability.

Color Palettes & Typography

Color Palette

Defines:

- Primary colors
- Secondary colors
- Accent colors
- Error/success colors

Color influences emotion:

- Blue = trust
- Green = success
- Red = danger
- Orange = energy

Typography

Deals with:

- Font family
- Font size hierarchy
- Line spacing
- Readability

Good typography = easier reading + better focus.

Design Comps & Style Guides

Design Comps (Design Compositions)

High-fidelity visual mockups that show exactly how the app will look.
This is the *final visual reference* for developers.

Style Guides

A rulebook that explains:

- Colors
- Fonts
- Buttons
- Icons

- Spacing
- UX patterns

Helps maintain consistency throughout the project.

Example: Food Delivery App (Final Look)

If lower planes handled logic and layout, the Surface Plane decides:

- Button color = Orange
- Fonts = Inter, 16px for body, 24px for headings
- Icons = Rounded style
- Background = White
- Images = High-resolution food photos
- Animation = Smooth "Add to Cart" bounce

Everything becomes polished and user-friendly.

Why the Surface Plane Is Important

Because **first impressions matter**.

Good visual design:

- Attracts users
- Builds trust
- Makes interaction smoother
- Reduces cognitive load
- Enhances brand identity

Even if structure and skeleton are perfect, bad visual design = poor UX

UNIT–II: Elements of UX Design

1. Interaction Design (from Structure Plane)

This is one of the most important exam topics.

To help you visualize, here's an Interaction Design example:

What is Interaction Design?

Interaction Design (IxD) focuses on:

> "How users interact with the system and how the system responds back."

It defines the **behavior** of the product.

If UX was a conversation, Interaction Design decides **how smooth the conversation feels**.

Interaction Design Decides:

User Actions

What users *do*:

- Click
- Tap
- Swipe
- Enter text
- Drag & drop

System Responses

How the system reacts:

- Shows a message
- Loads new screen
- Highlights errors
- Confirms success

Task Flows

Step-by-step paths users take to finish goals.

Example:
Add to cart → Checkout → Pay → Track order

Example: Food Delivery App

User Action	System Reaction
Tap “Add to Cart”	Item slides to cart + number increases
Enter wrong OTP	“Incorrect OTP, try again” + shake animation
Swipe left on item	Shows “Remove” option
Tap “Track Order”	Opens live map with rider location

Good interaction design makes the app feel **alive**, **predictable**, and **friendly**.

Goals of Interaction Design

Make tasks **easy and quick**
Reduce user errors
Provide immediate feedback
Make actions predictable
Ensure the experience feels smooth and enjoyable

Key Principles of Interaction Design

1. Feedback

Every user action should produce *visible or audible* feedback.

Example:

- Button turns grey when tapped
- Loader spins while data loads
- Vibrate on wrong password

---

2. Affordance

A design element should clearly show how it is meant to be used.

Example:

- Buttons look clickable
- Sliders look draggable
- Text fields look editable

---

3. Constraints

Guide users by preventing mistakes.

Example:

- “Submit” disabled until all fields are filled
- Incorrect email format → red highlight

---

4. Visibility

Important elements must be clearly seen.

Example:

- “Order Now” should be large and noticeable
- Hidden items frustrate users

---

5. Consistency

Same actions should always produce same results.

Example:

- Back button always returns one screen
- Swiping left always reveals delete options

---

6. Mapping

The control should match the result.

Example:

- Swipe right = move item right
- Up arrow = scroll up

Good mapping reduces confusion.

Why Interaction Design Matters?

Because users judge apps based on how they *feel*:

- Smooth app → great UX
- Laggy, confusing app → instant uninstall

Even beautiful UI fails if interaction design is bad.

2. Conceptual Models (Structure Plane)

What is a Conceptual Model? (Super Simple Definition)

A **Conceptual Model** is the *mental picture* that users have about **how your system works**.

It answers:

> "How does the user THINK this system operates?"

If the system works the way users *expect*, the experience feels smooth.
If not, users get confused → bad UX.

Example 1: Real-Life Conceptual Model

When you turn a **door knob**, you *expect* the door to open.
This expectation = **conceptual model**.

If a door looks like it should be pushed but actually must be pulled → confusion.

This same thing happens in apps and websites.

Example 2: Food Delivery App

User's Conceptual Model:

- Tap food → see details
- Add to cart → cart increases
- Checkout → payment
- Track order → map opens

If your app follows this natural flow → good conceptual design.

But if your app mixes steps (payment comes before choosing address), users get confused.

Why Conceptual Models Are Important

Because users don’t read manuals.
They depend on **intuition** and **previous experiences**.

Good conceptual models:
Reduce learning curve
Make app feel natural
Lower error rates
Improve user confidence
Increase usability

Types of Conceptual Models in UX

Object-Based Models

Users think of elements as *objects* they can interact with.

Example:
Cart → you put items into it, just like real life.

Task-Based Models

Focuses on users completing tasks step-by-step.

Example:
Login → Search → Order → Pay.

Metaphors

Use real-world similarities to explain digital actions.

Examples:

- “Trash Bin” icon for deleting
- “Folder” icon for storing files
- “Magnifying glass” for search

These metaphors make apps understandable instantly.

Good Conceptual Model = Predictable System

Users must *predict* what happens next.

For example:

- Tapping the back arrow should always go to the previous screen
- Swiping left should reveal options
- A red color should always indicate error

If an app breaks these expectations → users lose trust.

What Happens When Conceptual Models Fail?

Users get lost
They don't understand how to complete tasks
They click randomly
They abandon the product

Example:
If tapping "Order History" accidentally opens "Profile Settings", the conceptual model is broken.

One-Line Summary:

A Conceptual Model is the user's mental understanding of how the system works; good UX matches the user's expectations so the app feels natural and easy.

3. Error Handling

Error Handling = **How a system prevents, detects, and helps users recover from errors.**

Simple Definition

Error handling ensures the user doesn't get stuck, confused, or frustrated when something goes wrong.

Good error handling:

- Prevents errors
- Helps users understand what went wrong

- Shows how to fix it
- Reduces frustration

Types of Errors

User Errors (Slips & Mistakes)

When the user does something wrong by accident.

Examples:

- Entering wrong password
- Leaving a required field empty
- Typing an invalid email

System must help them fix it easily.

System Errors

These come from the backend or server.

Examples:

- Server down
- Payment gateway not responding
- Slow internet

System must show a meaningful message.

Principles of Good Error Handling

1. Prevent Errors Before They Happen

Best error is the one that never occurs.

Examples:

- Disable “Submit” until form is complete
- Auto-fill user info
- Show email format hint

---

2. Clear, Human-Friendly Error Messages

Avoid:
"Error 500"
"Invalid input exception"

Use:
"Please enter a valid 10-digit phone number."
"Your internet connection seems slow. Try again."

---

3. Highlight the Error Location

Mark the problematic field in **red** and show tips to fix it.

---

4. Offer a Way to Recover

Examples:

- Retry button
- Undo action
- "Try Again" for failed payments

---

5. Avoid Blaming Language

"You entered a wrong password"
"Incorrect password. Try again."

---

6. Use Confirmations to Reduce Errors

Example:
"Are you sure you want to delete this item?"

Why Error Handling Matters?

Because:

- Users hate feeling stupid
- Errors increase abandonment rate
- Clear error recovery creates better trust

Good apps help users. Bad apps punish them.

---

4. Information Architecture (IA)

Now let’s move to the second topic:
Information Architecture — how content is arranged and structured.

Simple Definition

> **Information Architecture (IA) is the practice of organizing, structuring, and labeling content so users can find what they need easily.**

Think of IA as the **map of your app or website**.

IA Helps Solve These Questions:

- Where should content go?
- What pages should exist?
- How should information be grouped?
- What labels should menus use?
- How will users navigate between sections?

IA Components

Organization Structure

How content is grouped.

Examples:

- Categories & Subcategories
- Alphabetical lists
- Chronological lists
- Topic-based grouping

Labeling System

Clear, understandable names for menu items and buttons.

Examples:

- “Order History” instead of “Previous Transactions”

- “Settings” instead of “User Preferences Panel”

Navigation System

How users move across the site.

Examples:

- Header menu
- Sidebar
- Breadcrumbs
- Tabs

Search System

Helps users find content quickly.

Examples:

- Search bar
- Filters
- Sort options

Examples of Information Architecture

Example 1: E-commerce Website

Categories:

- Electronics
    - Phones
    - Laptops
    - Accessories
- Fashion
    - Men
    - Women
    - Kids

Navigation:

- Home
- Categories
- Cart
- Profile

This helps users quickly find what they want.

Example 2: Food Delivery App

Menu Structure:

- Pizza
- Biryani
- Snacks
- Beverages
- Desserts

Navigation Elements:

- Home
- Search
- Orders
- Offers
- Profile

Without IA → everything becomes chaotic and confusing.

Why Is IA Important?

Because:

- Users leave when they can’t find what they want
- Good IA reduces cognitive load
- Users feel confident and in control
- Helps create smooth navigation and structure

Simple Analogy

IA is like arranging books in a library.

If books are placed randomly:
Nobody can find anything

If books are grouped by topic:
Easy, quick, satisfying experience

That’s exactly what IA does for websites and apps.

One-Line Summaries

Error Handling:

Helps users prevent and recover from mistakes with clear messages and guidance.

Information Architecture:

Organizes content logically so users can find what they need with ease.

Skeleton Plane:

Interface Design

Simple Meaning:

Interface Design decides **where interactive elements will be placed**, and how the user will physically interact with the system.

It deals with:

- Buttons
- Text fields
- Icons
- Toggles
- Sliders
- Dropdowns
- Pop-ups

These are called **interface elements**.

Goals of Interface Design:

- Make interactions **easy and natural**
- Ensure important items are **visible**
- Reduce user confusion
- Minimize effort to complete tasks

Example — Food Delivery App

Interface design would decide:

- “Add to Cart” button must be large and near thumb reach
- Star rating icons appear under the restaurant name
- Filters appear at the top
- Search bar is always visible

Bad interface = user gets confused
Good interface = user finishes task easily

Navigation Design

Simple Meaning:

Navigation Design decides **how users move from one screen/page to another**.

If Interface Design is *what* elements exist,
Navigation Design is *how to move around them*.

Types of Navigation:

- **Primary Navigation** → Bottom bar or top bar
- **Secondary Navigation** → Tabs, submenus
- **Breadcrumbs** → “Home > Electronics > Mobiles”
- **Sidebars**
- **Hamburger menu**
- **Back button behavior**

Example — Food App Navigation

Your app may have:

Bottom Navigation Bar:
Home | Search | Cart | Orders | Profile

Top Navigation Tabs:
Pizza | Biryani | Burgers | Desserts

Breadcrumbs:
Home > Restaurant > Menu > Checkout

Back Button:
Always takes user to previous step logically.

Good Navigation Design:

- Users never feel "lost"
- Movement between screens is **predictable**
- Frequently used areas are easy to reach

Information Design

Simple Meaning:

Information Design is about **presenting information clearly**, so users understand it easily and quickly.

It decides:

- What information appears first
- What is highlighted
- What is grouped together
- How text, images, and symbols are arranged

Goals of Information Design:

- Reduce user reading effort
- Improve comprehension
- Show important things first

- Use hierarchy (big → small)

Example — Product Page Information Layout:

A clear information design may show:

1. Product Image
2. Product Name
3. Price
4. Rating
5. Description
6. Add to Cart button

If description is shown at the top and price is hidden → bad information design.

Tools Used:

- Headings
- Subheadings
- Bullet points
- Icons
- Grouping + spacing
- Visual hierarchy

Wireframes

Simple Meaning:

> Wireframes are **simple, low-fidelity layouts** that show the placement of elements without colors or styling.

They are the blueprint of your app.

Wireframes show:

- Where the images go
- Where buttons go
- How screens connect

- What content appears where

Purpose of Wireframes:

- To plan layout
- To test usability early
- To discuss changes before UI design
- To avoid expensive redesigns later

Wireframes → Skeleton
UI Design → Surface

Example Wireframes (Food App)

Home Screen Wireframe:

```
[Search Bar]
[Banner Image]
[Categories Row]
[Restaurant List]
[Bottom Navigation]
```

Checkout Wireframe:

```
[Address Section]
[Order Summary]
[Payment Options]
[Place Order Button]
```

Types of Wireframes:

- **Low Fidelity** → simple boxes, black/white (quick sketches)
- **Mid Fidelity** → more details but no final colors
- **High Fidelity** → close to final design (often overlaps with surface plane)

SUMMARY TABLE (Skeleton Plane)

Topic	Meaning	Example
Interface Design	Placement of interactive UI elements	Button positions, text fields
Navigation Design	How users move across screens	Bottom bar, menus, back button
Information Design	Presenting info clearly and logically	Headings, grouping, hierarchy
Wireframes	Blueprint layout of screens	Sketch of home, product, checkout

SURFACE PLANE (Topmost Plane of UX)

This is where all design decisions become *visible*.
It answers:

> “How will the product look and feel to the user?”

The topics in the Surface Plane:

Sensory Design
Making Sense of the Senses
Contrast and Uniformity
Internal & External Consistency
Color Palettes and Typography
Design Comps and Style Guides

Let's explain each one clearly

Sensory Design

Simple Meaning:

Sensory Design deals with the **visual, auditory, and tactile (touch/feedback)** experience of the interface.

It involves:

- Visual elements (colors, shapes, images)
- Micro-animations
- Sounds (notification chime)
- Vibrations (haptic feedback)

Examples:

- A smooth animation when adding food to cart
- Error vibration when password is wrong
- A bell notification sound
- Eye-pleasing spacing and layout

Good sensory design makes the experience feel **polished, modern, and delightful**.

Making Sense of the Senses

Simple Meaning:

Use visual cues that help users understand things *instantly*, without reading long text.

The product should “explain itself” through design.

Examples:

- A delete icon =
- A green tick = success
- A red border = error
- A disabled button is greyed out
- A slider visually suggests moving left-right

If users can **guess** what to do without reading instructions → great UX.

Contrast and Uniformity

A. Contrast

Contrast helps highlight important elements.

Examples:

- “Order Now” button in bright orange
- Black text on white background for readability
- Bold headings vs. light body text

Good contrast = clarity, readability, and focus.

B. Uniformity (Consistency in appearance)

Uniformity ensures similar elements look similar everywhere.

Examples:

- All primary buttons use the same color
- Same font style and size for headings
- Same card layout for restaurants

Uniformity builds **predictability** and makes the interface look professional.

Internal & External Consistency

A. Internal Consistency

Consistency **within your own product**.

Examples:

- Back button always top-left
- Same icon style across screens
- Same error message style everywhere

If one screen has rounded buttons and another has sharp edges → bad internal consistency.

B. External Consistency

Consistency with **common design standards** users already know.

Examples:

- Shopping cart icon for checkout (universal meaning)
- Hamburger menu symbol for navigation
- Blue underline for clickable links

External consistency reduces the learning curve.

Color Palettes and Typography

A. Color Palette

A color palette decides:

- Primary color (main brand color)
- Secondary color
- Accent color
- Error color (usually red)
- Success color (usually green)

Examples:

- Swiggy → Orange + White
- Netflix → Red + Black
- Google → Multi-color

Good color choices:

- Improve emotion
- Strengthen brand identity
- Increase readability
- Provide accessibility

B. Typography

Typography controls:

- Font family (Roboto, Inter, etc.)
- Font sizes
- Line spacing
- Text hierarchy (H1 > H2 > body text)

Examples:

- Headings in bold 24px
- Body text in 16px
- Light grey for secondary information

Good typography makes reading effortless.

Design Comps and Style Guides

A. Design Comps (Design Compositions)

These are **high-fidelity visual designs** — the final version the developer will build.

Design comps include:

- Final colors
- Final icons
- Final spacing

- Final layout

They look almost exactly like the finished app.

B. Style Guides

A **Style Guide** is a rulebook that defines:

- Colors (with hex codes)
- Fonts and sizes
- Button styles
- Icon styles
- Spacing rules
- Grid system

Purpose:

- Maintain visual consistency
- Help all designers/developers stay aligned
- Speed up future updates

Example Style Guide Elements:

- Primary Button: Orange (#FF7A00), 16px text, rounded corners
- Headline Font: 24px, Bold
- Body Text: 16px, Regular
- Error Color: Red (#FF3333)

SUPER SUMMARY TABLE (Surface Plane)

Topic	Meaning	Example
Sensory Design	Appeals to senses using visuals, sound, animation	Cart animation, vibration on error
Making Sense of the Senses	Visual cues that communicate meaning	Trash icon for delete
Contrast	Highlight important elements	Bright CTA button
Uniformity	Keep look consistent	Same button style everywhere
Internal Consistency	Consistency within product	Same icon style

Topic	Meaning	Example
External Consistency	Follows common standards	Cart icon = checkout
Color Palettes	Defines color system	Orange theme (Swiggy)
Typography	Fonts and text hierarchy	24px bold heading
Design Comps	Final UI mockups	High-fidelity screens
Style Guides	Rules for colors, fonts, components	Branding guideline document

ONE-LINE TAKEAWAYS

- **Sensory Design** → Makes experience beautiful and interactive
- **Making Sense of Senses** → Users understand without reading
- **Contrast** → Shows what is important
- **Uniformity** → Makes design predictable
- **Internal & External Consistency** → Builds trust and reduces confusion
- **Color & Typography** → Defines visual language
- **Style Guides & Comps** → Ensure final UI is consistent and professional