Understanding the user

Understanding the User: Beyond Demographics

Great product teams don't start with "what should we build?"
They start with: **"who is this for, what are they struggling with, and why do they care?"**

---

1. Types of Understanding You Need

You must understand the user on **4 levels**:

1. **Goals** – What they want to achieve (long-term and in context of your product)
2. **Behaviors** – What they do now, not what they say they do
3. **Pain Points** – What frustrates them, slows them down, or causes them to quit
4. **Mental Models** – How they *think* things work, whether correct or not

---

2. Tactical Methods for Understanding

Let's break down key techniques you can use.

a) 1-on-1 User Interviews

Your goal is to get stories, not opinions.

Ask things like:

- "Tell me about the last time you tried to solve this problem."
- "What happened next?"

- “What was frustrating?”
- “What would have made it easier?”

Avoid asking:

- “Would you use this?”
- “Do you like this idea?”
  Because people lie to be polite.

---

b) User Shadowing or Self-Journaling

If possible, **watch them in their environment**.
If not, ask them to log what they do for a few days (voice notes, photos, screenshots).
Behavior beats speculation.

---

c) Surveys (careful here)

Use for patterns at scale.
Good questions:

- Rank these frustrations from most to least annoying
- In the last month, how many times did you try to solve [problem]?

Bad questions:

- “Do you care about health?” → Everyone says yes
- “Would you use a nutrition tracker?” → Too vague

---

d) Behavioral Analytics (after launch)

Once product is live, look at:

- Where people drop off
- Which features get repeated use
- Time spent, depth of use, return frequency

---

3. Patterns = Personas

Once you've gathered insights, **group them by patterns**, not just quotes.
Example pattern:

- Users who think about health only when they feel discomfort
- Users who want advice, not just tracking
- Users who hate typing after meals

From these patterns, you shape personas.

Proto-Persona: "Time-Starved Tracker"

- **Name**: Mira (or Mark), 42 years old
- **Profession**: Office worker or busy parent
- **Tech Skill**: Basic (uses apps but avoids complex ones)
- **Behavior**: Thinks about health when they feel pain or gain weight
- **Goal**:
    - Passively track nutrient intake without manual input
    - Stay on target with daily/weekly/monthly goals
    - Receive insights ("You're low on magnesium this week") and course corrections
- **Frustrations**:
    - Manual logging
    - Generic advice
    - Lack of feedback loops
- **Constraints**:
    - Low motivation
    - Poor consistency

    - Doesn't enjoy tracking

- **What wins them**:
    - Text-based input ("I ate chicken curry and rice")

Personas: Priorities, Ranking, and Testing

---

1. What people prioritize (and why it matters)

Users *don't* care about features. They care about **outcomes** — often emotional ones.
To design well, you need to understand *what they truly care about*, not what they say.

In UX, user priorities usually fall into these common categories:

Priority Type	Examples	What It Tells You
Functional	"I want to see my nutrient totals"	The job to be done — core utility
Emotional	"I want to feel I'm doing something right"	Triggers trust, confidence, motivation
Social	"I want to share progress with my trainer"	Adds accountability, connection
Effortless	"I don't want to think too much"	UX must be frictionless and fast
Aesthetic	"It should look clean and modern"	Drives perception of trust, quality

---

2. How people rank priorities

People rarely tell you straight. You need to observe **trade-offs** they make.
Use **forced-choice questions**, **scenario ranking**, or **observation**.

Tactics:

a) Scenario Ranking

Ask:

> "Imagine you had to choose between these two versions of an app:
> A) Very accurate but takes effort

B) Not perfect but super fast
Which would you use daily?”

They reveal their values by choosing.

b) Time-to-action trade-off

Ask:

> “How long are you willing to spend logging a meal each day?”

If they say “20 seconds or I won’t do it,” you know effortlessness beats accuracy for them.

c) Problem stack

Ask:

> “Rank these frustrations from most to least annoying:
>
> - Forgetting to log
> - Not knowing if I’m doing well
> - Not seeing progress
> - Manually typing food”

Now you see which pain is most urgent to solve first.

---

3. How to ask about priorities in interviews

Avoid asking “What matters most to you?” That’s too abstract.
Instead, ask **situation-based questions**:

- “Think about the last app you used for tracking. What made you stop using it?”
- “What’s more frustrating — not seeing your progress, or having to log every meal?”
- “Tell me about a time when you felt you were doing well with your nutrition. What made you feel that way?”

You’re listening for pain, motivation, and success moments.

---

4. How to test your design against their priorities

This is key. You don't test if the design *looks good*. You test if it delivers on their *priorities*.

a) Setup usability tests focused on user goals
Say:

> "Your goal is to stay under 50g of sugar today — can you figure out how close you are?"

Watch:

- Where they hesitate
- What they miss
- What frustrates them

b) Post-task reflection
Ask:

- "Was that what you expected?"
- "Was anything annoying?"
- "Would you do this every day? Why or why not?"

c) Prioritization feedback loop
After testing, ask:

> "Did that help you with what's most important to you — or was it just extra noise?"

If the feature doesn't help their top priorities, it's fluff — cut it or improve it.

Summary Checklist: Testing Design Against User Priorities

- Do I know my user's *top 2 priorities*?
- Have I seen them choose one feature over another?

- Did I observe frustration or flow in high-priority moments?
- Did my test confirm we’re solving the thing that matters most?

User stories

User Stories — The DNA of Product Thinking

INTRODUCTION: Why User Stories Matter

Most product teams fail not because they build poorly — but because they build *the wrong thing*.
User stories are the **language of intent**. They're how we capture the *why* behind a feature before writing a single line of code.

At a startup, you're operating in chaos. Everyone's guessing.
User stories are your anchor — they translate chaos into clarity.

DEFINITION

A user story is not a requirement. It is a **short, structured statement of need from the user's point of view**.
It typically follows this format:

> **As a [type of user], I want to [take some action] so that I can [achieve some goal].**

Simple, but deep.

This structure forces you to:

- Name the *real user* — not "admin" or "customer," but something human
- Describe the *action* — not vague wishes, but concrete behavior
- Describe the *value* — the "so that I can" is the most important part. It's where motivation lives.

EXAMPLES: Shallow vs. Real

Bad story:

> As a user, I want to log in.

This is a task, not a story. It doesn't tell you *why* the login matters.

Better:

> As a returning user, I want to access my saved data so I don't have to re-enter everything every time.

Now we understand motivation: convenience, retention, saving effort.

Another:

> As a busy professional, I want to text what I ate so I can stay on track without disrupting my day.

That's actionable. That defines the design. That aligns with business value.

LEVELS OF USER STORIES

Think in 3 layers:

1. **Epic**
    - Broad objective (e.g., "Manage my health")
2. **User Story**
    - One action + one value (e.g., "Text a meal and get nutrients")
3. **Sub-tasks or Acceptance Criteria**
    - What the dev/designer must actually build
    - (e.g., “Parse meal text into macros”, “Send feedback message”)

GOOD USER STORIES ARE TESTABLE

That's the difference between a story and a wish.

A good user story leads directly to a question:

> “Can the user now achieve this goal with the feature we built?”
> If not — your story is incomplete, or your solution missed the point.

HOW TO WRITE GREAT USER STORIES

1. **Start with real interviews or behavior**
    - Never write in isolation
    - Listen for user quotes that match the story structure
    - Stories come from the field, not your head
2. **Avoid vague roles**
    - “User” is too generic
    - Say: “new user,” “power user,” “busy parent,” “athlete recovering from injury”
3. **Ground in context**
    - Add small context notes (e.g., “usually happens at lunch break”)

HOW TO USE STORIES IN A TEAM

- As a **PM**, use stories to write product specs
- As a **UX designer**, use stories to shape flows and interface
- As a **dev**, use stories to scope and build features
- As a **team**, use stories in backlog grooming, sprint planning, design reviews

---

EXERCISE FOR YOU

Pick a feature of Nutripal you want to build.
Now write:

1. **1 Epic**
2. **2 User Stories under that epic**
3. **2 Acceptance criteria for one of the stories**

Answers:
1. **Epic:** I want to track my nutrition to achieve my eating habit goals
2. **User Story 1:** "as office worker, after my lunch break, i want to write it in natural language to ai what i ate and get my meal and nutrients logged"
User Story 2: "As a CTO of a small company, i want to take a picture of my breakfast to get it logged and get advice on what to eat at lunch"
3. **Acceptance criteria 1:** getting nutrients info from meal description
Acceptance criteria 2: analysing nutrients based on picture, Creating meal advice based on logged food and goals

Epic Review

> *"I want to track my nutrition to achieve my eating habit goals."*

✔ **Clear intent**
✔ Broad enough for multiple user stories
✏ Minor note: consider making it active — “Track my nutrition so I can improve my eating habits.” Makes it more outcome-driven.

User Story 1

> *"As an office worker, after my lunch break, I want to write in natural language to AI what I ate and get my meal and nutrients logged."*

✔ Specific user (not just “user”)
✔ Includes time/context (“after my lunch break”) — good for behavior modeling
✔ Outcome is clear: logs meal, gets nutrient info
✏ Optional refinement: "so that I can stay on track without disrupting my workday" — adds value layer.

User Story 2

> *"As a CTO of a small company, I want to take a picture of my breakfast to get it logged and get advice on what to eat at lunch."*

✔ Very specific persona — nice
✔ Combined tracking + proactive advice
✔ This moves from passive tracking to an intelligent recommendation system — great vision
✏ Suggest: break into two stories later (tracking from photo vs. receiving advice), but keep as one for now if you're MVP-scoping.

Acceptance Criteria

1. **Getting nutrients from meal description**
   ✔ Direct, testable
   ✔ Can be broken into subcriteria later (accuracy threshold, confirmation by user)

2. **Analyzing nutrients from picture + creating advice**
   ✔ Ambitious but valid
   ✏ This combines computer vision + goal-aware reasoning. Likely two separate subsystems.
   → You could write this as:

   - "Given a breakfast photo, system estimates nutrient profile with at least 70% confidence"
   - "System offers one lunch recommendation aligned with user's current nutrient gap and stated goal (e.g. weight loss)"

Customer Journey Mapping

INTRO

You don't build great products by designing isolated screens. You build great products by **mapping how real people experience the world over time** — their thoughts, feelings, and struggles.

The **customer journey map** is your zoomed-out system lens. It's how you stop optimizing for pixels and start designing for *moments that matter*.

---

1. WHAT A CUSTOMER JOURNEY MAP *ACTUALLY* IS

Not a deliverable. Not a pretty chart.
It's a **thinking tool**. A journey map is your visual hypothesis of what the user goes through — emotionally, mentally, and behaviorally — to get from *problem* to *resolution*.

In a startup, it helps you:

- Align product and design around user pain
- Prioritize what to build next based on experience gaps
- Spot where you're bleeding users — not in code, but in moments

You are not designing screens. You are designing *the arc of experience*.

---

2. CORE STRUCTURE OF A JOURNEY MAP

You always ask:
"What is my user trying to achieve, and what do they go through to get there?"

We break it down into **5 core dimensions**:

Stage	Goal	Action	Emotion	Friction Point	Opportunity

Let’s walk through each.

a) Stage

A logical phase of the experience.
Examples: Discover the product → Sign up → First use → Daily routine → Drop-off or loyalty

This defines the flow of time in the experience.

b) User Goal

What the user *wants* to achieve in this stage — not what you want them to do.

> In sign-up, your goal is conversions.
> But *their* goal might be to “quickly try it without commitment.”

User-first thinking starts here.

c) Action

What they actually do — both *inside* your product and around it.
Includes things like “searched on Google,” “checked App Store reviews,” “asked a friend,” “opened app, closed it after 2 seconds.”

Journey maps aren't limited to your UI.

d) Emotion

How do they feel here?

- Excited? Anxious? Impatient?
- Confused? Relieved? Impressed?

Map these honestly. Emotions shape behavior more than UI does.

e) Friction

Where does the user get stuck, confused, annoyed, or lose momentum?

Examples:

- “Too many steps before I see value”
- “I’m not sure what this button does”
- “I don’t see the benefit of tracking every day”

This is gold. This is where you design.

f) Opportunity

This is your call.

- What can you simplify?
- What feedback loop can you build?
- Where can you delight, nudge, or coach?

This is where UX stops being reactive and becomes strategic.

---

3. EXAMPLE WALKTHROUGH — MIRA USING NUTRIPAL

Let’s simulate what Mira’s journey might look like. I’ll narrate this like a story while mapping it:

Stage 1: Discovery

- **Goal**: “Find something that helps me eat better without extra work.”
- **Action**: Googles “easy food tracker,” sees an Instagram ad
- **Emotion**: Hopeful but skeptical
- **Friction**: Doesn’t trust new apps, burned by MyFitnessPal before
- **Opportunity**: Build trust instantly — social proof, testimonials, “No typing needed” promise

---

Stage 2: Sign-Up

- **Goal**: Try it without too much commitment
- **Action**: Installs app, skips data entry
- **Emotion**: Curious, cautious
- **Friction**: Long onboarding, required form fields
- **Opportunity**: One-tap onboarding, optional setup later, instant demo mode

---

Stage 3: First Log

- **Goal**: Try logging a meal quickly
- **Action**: Texts “chicken curry with rice”
- **Emotion**: Interested, waiting to be impressed
- **Friction**: Delay in response, doesn’t get clear nutrient breakdown
- **Opportunity**: Instant feedback, “Great choice! Here’s what we got from that meal…” plus a tip

---

Stage 4: Daily Use

- **Goal**: Stay on track with minimal input
- **Action**: Logs food 1–2 times/day, skims advice
- **Emotion**: Routine forming, low effort
- **Friction**: No visual sense of progress, gets bored
- **Opportunity**: Streaks, nudges, “You’ve logged 5 days in a row!” or “You’re low on fiber today — want a tip?”

Stage 5: Progress Review or Drop-Off

- **Goal**: See if effort is paying off
- **Action**: Opens dashboard
- **Emotion**: Curious or disappointed
- **Friction**: Stats are too complex or too vague
- **Opportunity**: Show one key win (“You’ve eaten 28 balanced meals in 2 weeks!”), simple trend lines, shareable insights

---

4. WHEN TO BUILD A JOURNEY MAP

- **Before design**: to model what kind of experience you need to create
- **After user interviews**: to align patterns you heard into structured phases
- **After MVP launch**: to diagnose where users drop and why

---

5. HOW TO MAKE IT STICK

Don’t keep it in a doc.
Print it, whiteboard it, put it in Figma — and update it as you learn more.

Best practice: pair every **user story** with the **stage of the journey** it belongs to. This makes sure you’re solving the right problems at the right time.

Final Thought

Good product design doesn’t just ask:
“What feature should we build?”

It asks:
“Where in the user’s journey is the pain, and how can we remove it, support it, or turn it into momentum?”

That’s how you build products people keep using.

Conducting UX Research: Seeing with the User's Eyes

INTRO

Every pixel you design is either a guess or an answer.
UX research is how you **turn guesses into evidence**.

It's not a phase. It's a loop — a way of working. In a startup, it's your edge.

When founders guess, when competitors guess, you **learn**. You observe real users and let them show you what matters.

---

1. WHAT IS UX RESEARCH?

UX research is the process of **understanding users' needs, behaviors, and frustrations** so you can design solutions that work in the real world.

It's **not about opinions**. It's about:

- Real behavior
- Real emotion
- Real context

UX research asks:

> *What is this person trying to do, and what gets in their way?*

---

2. TYPES OF UX RESEARCH (LOW-BUDGET STARTUP VERSION)

You don’t need labs, panels, or fancy tools. You need curiosity and structure.

Here are **four core methods** you can use as a startup PM/UX designer:

A) User Interviews

Qualitative gold.

Goal: Uncover motivation, problems, and decision-making patterns

Structure:

- Warm-up: "Tell me about your day" (context)
- Trigger: "Last time you tried to do [X], what happened?"
- Deep dive: "What did you feel frustrated with? What helped?"
- Reflection: "What would have made it easier?"

Tips:

- Shut up and let them talk
- Dig into stories, not statements
- Always record (with permission)

B) Usability Testing

Your best weapon once you have a design.

Goal: Observe how users interact with your interface — where they struggle or hesitate

How it works:

1. Prepare 3–5 realistic tasks (“Log a meal,” “Check your fiber intake”)
2. Ask users to think aloud

3. Watch, don't guide

Focus on:

- Confusion
- Misclicks
- Time to success
- "I don't know what this does"

Test with just 5 users — 80% of problems will appear

C) Surveys

Use with caution. Great for patterns, bad for depth.

Goal: Quantify trends, rank problems, validate ideas

Good questions:

- "What tools do you currently use to track nutrition?"
- "What's your #1 frustration with food logging?"
- "How often do you abandon tracking?"

Avoid:

- Yes/no
- Hypotheticals
- Leading questions

D) Behavioral Analytics (after launch)

Your map of what users actually do.

Use tools like: Mixpanel, Amplitude, Hotjar, custom logs
Watch for:

- Drop-off points (onboarding, log streaks)
- Time-on-task
- Feature abandonment
- Rage clicks / UI dead zones

Analytics show **what happened**, interviews show **why**.

Use both.

---

3. HOW TO PLAN UX RESEARCH

In a startup, you're always short on time.
You need a **lean, focused research plan**.

Use this structure:

Step	Question
1. Research Goal	What do we need to learn right now?
2. Method	Interview, test, survey, analytics?
3. Users	Who will give the most relevant insight?
4. Script	5–7 open-ended questions or 3 clear tasks
5. Output	What decisions will this help us make?

---

4. ANALYZING AND SYNTHESIZING FINDINGS

Once you talk to users or run tests — don't just quote them. **Group patterns. Extract signals.**

Use sticky notes, Miro, Notion — and cluster by:

- Problems they mention
- Desired outcomes
- Workarounds they use
- Emotions (frustration, relief, confusion)

From those clusters, derive:

- Priorities
- Feature direction
- UX friction zones

---

5. WHAT MAKES UX RESEARCH *REAL*

Great researchers:

- Look for *behavior*, not words
- Observe *emotion*, not just task success
- See the user as a **human in context**, not a "test subject"

You're not proving your design works. You're **watching how it fails** — so you can make it better.

PRACTICAL EXERCISE (Your Turn)

Let's say you want to research Mira's daily logging experience with Nutripal.

Write a **quick UX research plan** by answering these:

1. What's your research goal?
2. What method will you use?
3. Who will you talk to (describe user type)?
4. What are 3 questions or tasks you'll ask?
5. What do you want to learn to make a product decision?

Designing Intuitive Interfaces

INTRO: WHAT IS "INTUITIVE" DESIGN?

When we say a design is “intuitive,” we often mean:

> *“I didn’t have to think about it. I just knew what to do.”*

But intuition is not magic.
It’s the result of:

- Familiar patterns
- Predictable feedback
- Cognitive simplicity
- Emotional alignment

The user isn’t guessing. They’re recognizing.
Great design doesn’t teach the user — it **respects what they already know**.

---

1. THE PSYCHOLOGY OF INTUITIVE DESIGN

Your users bring **mental models** with them — unconscious expectations of how things should behave.
Mental models come from:

- Everyday life (switches, knobs, checklists)
- Other apps (iOS, Android, web UI norms)
- Cultural habits (reading direction, icons, colors)

If your design matches their mental model, it feels intuitive.
If it fights it, it feels broken.

Design intuition = alignment between your interface and their mental model.

---

2. THE PRINCIPLES OF INTUITIVE DESIGN

Let's break it into five core principles. These are non-negotiable.

A. Clarity Over Cleverness

> "If it looks cool but causes hesitation, it's wrong."

- Use clear, familiar labels ("Log Meal" beats "Fuel Up")
- Prioritize legibility, not artistry
- Don't invent navigation metaphors unless absolutely necessary

---

B. One Action Per Screen

> "Cognitive overload kills intuition."

- Give users *one primary decision* per view
- Reduce scanning and choice paralysis
- Group related elements together (Gestalt law of proximity)

C. Consistency

> "Intuition is pattern recognition. Break the pattern, break the brain."

- Keep navigation, buttons, and interaction behavior the same across the product
- Use consistent icon language, visual hierarchy, and layout grids

- Don’t surprise your user unless you’re delighting them — not confusing them

---

D. Immediate Feedback

> “The brain needs confirmation that the action worked.”

- Every tap, click, swipe should respond — visually or audibly
- Use subtle animation, microcopy, haptics
- Examples: Button changes color on tap, text appears with a fade, a checkmark flashes briefly after logging a meal

---

E. Affordances and Signifiers

> “Show me what I can do, not just what it is.”

- **Affordance** = What the object *can* do
  (e.g., a raised button looks pressable)
- **Signifier** = Clue that invites the action
  (e.g., underline = link, trash icon = delete)

Design must communicate function *visually*.

---

3. HOW TO TEST IF A DESIGN IS INTUITIVE

You don’t ask “Do you like it?”
You ask: **Can the user do what they’re supposed to do without thinking too hard?**

Here’s how to test it:

A. First-Click Test

Ask:

> “Where would you click first to [log a meal]?”

- If they hesitate or ask questions, you have a problem
- Measure accuracy: If 80% of users click the right element on first try, it’s intuitive
- Use paper mockups, Figma, or live product

B. 5-Second Test

Show a screen for 5 seconds. Then hide it.

Ask:

> “What was that screen about?”
> “What could you do on it?”
> “What would you do next?”

This tests **clarity and hierarchy** — if users can’t understand the purpose instantly, you’re too abstract.

C. Silent Usability Test

Tell user:

> “Please complete this task. I won’t help or guide.”

Watch:

- Where they pause
- Where they backtrack
- Where they guess

Confusion = cognitive load = anti-intuition.

D. Expectations Test

Ask:

> "What do you think will happen if you tap that?"

Then have them do it.

If the result **matches their expectation**, your design is intuitive. If not, redesign.

---

4. EXAMPLES FROM REAL PRODUCTS

- **Apple Health**: Extremely clean, but suffers from unclear hierarchy. Users often don't know where to go. Looks intuitive, isn't.
- **Instagram Stories**: Simple interaction (tap to skip, swipe to exit). Highly intuitive because the interaction is taught through repetition.
- **Notion**: Complex tool, but uses consistent design patterns, clear iconography, and layered discovery — becomes intuitive over time.

---

5. INTUITION IS A GRADIENT

Some things are intuitive on **first use**
Others become intuitive through **repetition** (muscle memory, predictability)

You must design for both:

- First-use clarity
- Long-term mastery

---

6. HOW TO IMPROVE INTUITION IN A PRODUCT

- Use **progressive disclosure** (show complexity only when needed)

- Apply **Jakob's Law**: Users prefer your product to work like products they already know
- Build **error forgiveness** (undo, confirmation, no punishment for trying things)
- Use **empty states** and **tooltips** to guide early users without cluttering the UI for advanced ones

---

7. INTUITIVE DESIGN IN MVPs

Even in MVPs, design intuition is not optional.
You get one chance to earn trust. If the first experience is confusing, they're gone.

- Default to known UI patterns unless you *must* invent
- Always test with fresh eyes before shipping
- Assume no one will read instructions

Final Reflection

Intuitive design is about **removing friction between the user's intent and the outcome.**

If your product makes people stop and think, not because it's interesting, but because it's *unclear* — you've failed.

Your job is to make every step feel like:

> "Of course. That's exactly what I expected."