UCLA COMPUTER SCIENCE FIELD NOTES

CS 188: HCI

Human-Computer Interaction

Human-centered research, interaction design, prototyping, evaluation, accessibility, and responsible product decisions

How these notes are written. I wrote each chapter the way I wish the material had been explained the first time: the real idea, the point where I usually get stuck, the simplest mental model that still stays honest, and a worked decision instead of a polished answer appearing from nowhere. Every chapter closes with practice and a fully written solution. If a sentence sounds impressive but does not help me solve or explain something, it does not belong here.

Daniel Mastick

Curriculum map, working notes, practice, and solutions

Course map

Week	Core thread	What should be solid by the end
1	**Human-centered problem framing**	Explain the model, apply it to a concrete case, and defend the relevant correctness or design decision.
2	**Qualitative research and synthesis**	Explain the model, apply it to a concrete case, and defend the relevant correctness or design decision.
3	**Models of action and cognition**	Explain the model, apply it to a concrete case, and defend the relevant correctness or design decision.
4	**Information architecture and interaction flows**	Explain the model, apply it to a concrete case, and defend the relevant correctness or design decision.
5	**Prototyping and design systems**	Explain the model, apply it to a concrete case, and defend the relevant correctness or design decision.
6	**Usability evaluation**	Explain the model, apply it to a concrete case, and defend the relevant correctness or design decision.
7	**Accessibility and inclusive interaction**	Explain the model, apply it to a concrete case, and defend the relevant correctness or design decision.
8	**Experiments, ethics, and product judgment**	Explain the model, apply it to a concrete case, and defend the relevant correctness or design decision.
9	**Integration workshop**	Connect at least three chapters in one end-to-end problem and explain where their contracts meet.
10	**Synthesis and project review**	Audit assumptions and explain a complete solution from interface to failure handling.

Scope anchor: https://catalog.registrar.ucla.edu/course/2025/COMSCI188. The sequence is aligned to the official catalog description and expanded into a practical ten-week study plan.

How I would use this packet

Treat each numbered section as one chapter. Read the formal explanation, pause at the student interpretation, and see whether the easy version still says the same thing. Rework the example without looking, then cover the solution in the chapter practice box. The cumulative set at the end is deliberately mixed: knowledge becomes durable when I have to choose the tool instead of being told which chapter I am in.

Contents

CHAPTER 1 Human-centered problem framing

The idea

HCI studies the fit among people, tasks, technologies, and contexts. A design problem should identify users, goals, constraints, current workarounds, and harms without jumping immediately to a feature.

Rebuild the chapter from first principles

The claim I need to own is this: HCI studies the fit among people, tasks, technologies, and contexts. A design problem should identify users, goals, constraints, current workarounds, and harms without jumping immediately to a feature. I do not count that as learned just because the sentence looks familiar. I should be able to name the objects, state what changes and what stays fixed, recover the rule from the definitions, and explain why the result is allowed.

The move

My working sequence

1. **Translate.** Rewrite the prompt in the language of human-centered problem framing. List the known quantities, the unknown, the units or types, and any hidden constraint.
2. **Choose.** State the theorem, model, algorithm, accounting rule, or physical law before substituting values. This is where I make the assumption visible.
3. **Execute.** Work one justified step at a time and keep intermediate state readable enough that I can audit it later.
4. **Verify.** Check a boundary case, sign, unit, invariant, order of magnitude, or alternative derivation. Then translate the result back into an ordinary sentence.

Details I would refuse to skip

- **Definitions:** I should be able to define every technical noun in the chapter without using that same noun in the definition.
- **Assumptions:** I write the condition that makes the method legal beside the method itself. A correct procedure under the wrong assumptions is still a wrong answer.
- **Representation:** I move freely among words, equations or code, a diagram, and a small numerical example. If one representation disagrees with another, that disagreement is useful evidence.
- **Failure modes:** I keep the nearest counterexample in mind: the boundary where the rule changes, the invariant breaks, or the model stops matching reality.
- **Communication:** My final answer names what the result means and what it does not establish. I do not let notation do the explanatory work for me.

Sanity check

Closed-note check. Can I teach human-centered problem framing on a blank page, derive or reconstruct its main rule, work a fresh example, and diagnose a deliberately broken solution? If I can only recognize the finished work, I am not done.

How this chapter connects

I read this chapter as the bridge from the course foundations to qualitative research and synthesis. The earlier material supplies the language and constraints; this chapter adds a new reasoning move; the later material asks me to use that move without being told its name. That connection matters because exams and real projects mix chapters on purpose.

What I need to be able to do

For this chapter, I should be able to define the objects and state involved, state the governing invariant or contract, trace a small example without hand-waving, and explain which assumption makes the method valid. I should also be able to compare this model with the nearest alternative and say what failure would make me switch approaches.

Deep notes

- **Model.** HCI studies the fit among people, tasks, technologies, and contexts.
- **Mechanism.** A design problem should identify users, goals, constraints, current workarounds, and harms without jumping immediately to a feature.
- **Boundary.** The useful test is whether I can apply human-centered problem framing to a new case and explain the assumption that makes the answer valid.
- **Decision test.** I should be able to say when human-centered problem framing is the right model, when its assumptions fail, and what evidence would change my choice.

How I actually think about it

I write the user's job in verbs before I draw a screen.

The easy version

HCI studies the fit among people, tasks, technologies, and contexts.

Worked example

A useful way to work this chapter is to make the smallest concrete instance, write the state or contract explicitly, and then scale the reasoning only after that instance behaves correctly. The worked question below makes that reasoning concrete: Turn 'build a study app' into a researchable problem. The conclusion is Specify which students, what study task, where breakdowns occur, how success is observed, and which constraints matter; the app is only one possible response.

Common miss

The common mistake is to memorize the visible procedure while dropping its precondition. I write the assumption beside the method and test one boundary case before trusting the result.

Solved chapter practice

Practice. Turn 'build a study app' into a researchable problem.
Solution. Specify which students, what study task, where breakdowns occur, how success is observed, and which constraints matter; the app is only one possible response.

Practice 2. Give a short oral explanation of human-centered problem framing that includes its model, one assumption, one failure mode, and one comparison with a neighboring method.
Solution. A strong answer begins with the model in the blue box, names the assumption that licenses the method, uses the common miss above as the failure mode, and chooses the comparison according to the decision test in the deep notes. The goal is not one memorized sentence; it is a four-part explanation I can reproduce on a new example.

Mastery practice A. Reconstruct human-centered problem framing from a blank page. Include the central definition, the governing rule, one required assumption, and one boundary case.
Solution. Start from the blue idea box, but rewrite it in my own words. Identify the objects and state, name the rule before using it, copy the relevant assumption from the details audit, and choose a boundary where that assumption becomes equality or fails. A complete solution explains why each part belongs; a list of vocabulary is not enough.

Mastery practice B. A classmate gets a polished-looking answer to a human-centered problem framing problem but never states a model or checks the result. Write the shortest useful review of that solution.
Solution. I would ask four concrete questions: What exactly are the inputs and desired output? Which rule licenses the main step? Which assumption could invalidate it? Which independent check supports the conclusion? If the answer cannot survive those four questions, I treat the numerical or verbal result as unverified.

CHAPTER 2 Qualitative research and synthesis

The idea

Interviews, contextual inquiry, observation, diary studies, and artifact analysis reveal practices and meanings. Sampling, consent, leading questions, researcher interpretation, and thematic coding shape the evidence.

Rebuild the chapter from first principles

The claim I need to own is this: Interviews, contextual inquiry, observation, diary studies, and artifact analysis reveal practices and meanings. Sampling, consent, leading questions, researcher interpretation, and thematic coding shape the evidence. I do not count that as learned just because the sentence looks familiar. I should be able to name the objects, state what changes and what stays fixed, recover the rule

from the definitions, and explain why the result is allowed.

The move

My working sequence

1. **Translate.** Rewrite the prompt in the language of qualitative research and synthesis. List the known quantities, the unknown, the units or types, and any hidden constraint.
2. **Choose.** State the theorem, model, algorithm, accounting rule, or physical law before substituting values. This is where I make the assumption visible.
3. **Execute.** Work one justified step at a time and keep intermediate state readable enough that I can audit it later.
4. **Verify.** Check a boundary case, sign, unit, invariant, order of magnitude, or alternative derivation. Then translate the result back into an ordinary sentence.

Details I would refuse to skip

- **Definitions:** I should be able to define every technical noun in the chapter without using that same noun in the definition.
- **Assumptions:** I write the condition that makes the method legal beside the method itself. A correct procedure under the wrong assumptions is still a wrong answer.
- **Representation:** I move freely among words, equations or code, a diagram, and a small numerical example. If one representation disagrees with another, that disagreement is useful evidence.
- **Failure modes:** I keep the nearest counterexample in mind: the boundary where the rule changes, the invariant breaks, or the model stops matching reality.
- **Communication:** My final answer names what the result means and what it does not establish. I do not let notation do the explanatory work for me.

Sanity check

Closed-note check. Can I teach qualitative research and synthesis on a blank page, derive or reconstruct its main rule, work a fresh example, and diagnose a deliberately broken solution? If I can only recognize the finished work, I am not done.

How this chapter connects

I read this chapter as the bridge from human-centered problem framing to models of action and cognition. The earlier material supplies the language and constraints; this chapter adds a new reasoning move; the later material asks me to use that move without being told its name. That connection matters because exams and real projects mix chapters on purpose.

What I need to be able to do

For this chapter, I should be able to define the objects and state involved, state the governing invariant or contract, trace a small example without hand-waving, and explain which assumption makes the method

valid. I should also be able to compare this model with the nearest alternative and say what failure would make me switch approaches.

Deep notes

- **Model.** Interviews, contextual inquiry, observation, diary studies, and artifact analysis reveal practices and meanings.
- **Mechanism.** Sampling, consent, leading questions, researcher interpretation, and thematic coding shape the evidence.
- **Boundary.** The useful test is whether I can apply qualitative research and synthesis to a new case and explain the assumption that makes the answer valid.
- **Decision test.** I should be able to say when qualitative research and synthesis is the right model, when its assumptions fail, and what evidence would change my choice.

How I actually think about it

I ask for the last concrete time something happened, not what people imagine they usually do.

The easy version

Interviews, contextual inquiry, observation, diary studies, and artifact analysis reveal practices and meanings.

Worked example

A useful way to work this chapter is to make the smallest concrete instance, write the state or contract explicitly, and then scale the reasoning only after that instance behaves correctly. The worked question below makes that reasoning concrete: Why avoid 'Would you use this feature?' The conclusion is Hypothetical agreement is cheap and socially biased. Ask about current behavior, pain, tradeoffs, and observed workarounds, then test an actual prototype.

Common miss

The common mistake is to memorize the visible procedure while dropping its precondition. I write the assumption beside the method and test one boundary case before trusting the result.

Solved chapter practice

Practice. Why avoid 'Would you use this feature?'
Solution. Hypothetical agreement is cheap and socially biased. Ask about current behavior, pain, tradeoffs, and observed workarounds, then test an actual prototype.

Practice 2. Give a short oral explanation of qualitative research and synthesis that includes its model, one assumption, one failure mode, and one comparison with a neighboring method.
Solution. A strong answer begins with the model in the blue box, names the assumption that licenses the method, uses the common miss above as the failure mode, and chooses the comparison according to the decision test in the deep notes. The goal is not one memorized sentence; it is a

four-part explanation I can reproduce on a new example.

Mastery practice A. Reconstruct qualitative research and synthesis from a blank page. Include the central definition, the governing rule, one required assumption, and one boundary case.
Solution. Start from the blue idea box, but rewrite it in my own words. Identify the objects and state, name the rule before using it, copy the relevant assumption from the details audit, and choose a boundary where that assumption becomes equality or fails. A complete solution explains why each part belongs; a list of vocabulary is not enough.

Mastery practice B. A classmate gets a polished-looking answer to a qualitative research and synthesis problem but never states a model or checks the result. Write the shortest useful review of that solution.
Solution. I would ask four concrete questions: What exactly are the inputs and desired output? Which rule licenses the main step? Which assumption could invalidate it? Which independent check supports the conclusion? If the answer cannot survive those four questions, I treat the numerical or verbal result as unverified.

CHAPTER 3 Models of action and cognition

The idea

Gulf of execution and evaluation, mental models, recognition versus recall, cognitive load, attention, and memory explain interaction costs. Models guide hypotheses but do not replace research with real users.

Rebuild the chapter from first principles

The claim I need to own is this: Gulf of execution and evaluation, mental models, recognition versus recall, cognitive load, attention, and memory explain interaction costs. Models guide hypotheses but do not replace research with real users. I do not count that as learned just because the sentence looks familiar. I should be able to name the objects, state what changes and what stays fixed, recover the rule from the definitions, and explain why the result is allowed.

The move

My working sequence

1. **Translate.** Rewrite the prompt in the language of models of action and cognition. List the known quantities, the unknown, the units or types, and any hidden constraint.
2. **Choose.** State the theorem, model, algorithm, accounting rule, or physical law before substituting values. This is where I make the assumption visible.
3. **Execute.** Work one justified step at a time and keep intermediate state readable enough that I can audit it later.
4. **Verify.** Check a boundary case, sign, unit, invariant, order of magnitude, or alternative derivation. Then translate the result back into an ordinary sentence.

Details I would refuse to skip

- **Definitions:** I should be able to define every technical noun in the chapter without using that same noun in the definition.
- **Assumptions:** I write the condition that makes the method legal beside the method itself. A correct procedure under the wrong assumptions is still a wrong answer.
- **Representation:** I move freely among words, equations or code, a diagram, and a small numerical example. If one representation disagrees with another, that disagreement is useful evidence.
- **Failure modes:** I keep the nearest counterexample in mind: the boundary where the rule changes, the invariant breaks, or the model stops matching reality.
- **Communication:** My final answer names what the result means and what it does not establish. I do not let notation do the explanatory work for me.

Sanity check

Closed-note check. Can I teach models of action and cognition on a blank page, derive or reconstruct its main rule, work a fresh example, and diagnose a deliberately broken solution? If I can only recognize the finished work, I am not done.

How this chapter connects

I read this chapter as the bridge from qualitative research and synthesis to information architecture and interaction flows. The earlier material supplies the language and constraints; this chapter adds a new reasoning move; the later material asks me to use that move without being told its name. That connection matters because exams and real projects mix chapters on purpose.

What I need to be able to do

For this chapter, I should be able to define the objects and state involved, state the governing invariant or contract, trace a small example without hand-waving, and explain which assumption makes the method valid. I should also be able to compare this model with the nearest alternative and say what failure would make me switch approaches.

Deep notes

- **Model.** Gulf of execution and evaluation, mental models, recognition versus recall, cognitive load, attention, and memory explain interaction costs.
- **Mechanism.** Models guide hypotheses but do not replace research with real users.
- **Boundary.** The useful test is whether I can apply models of action and cognition to a new case and explain the assumption that makes the answer valid.
- **Decision test.** I should be able to say when models of action and cognition is the right model, when its assumptions fail, and what evidence would change my choice.

How I actually think about it

When an interface feels confusing, I ask whether the next action is visible and whether the result explains what happened.

The easy version

Gulf of execution and evaluation, mental models, recognition versus recall, cognitive load, attention, and memory explain interaction costs.

Worked example

A useful way to work this chapter is to make the smallest concrete instance, write the state or contract explicitly, and then scale the reasoning only after that instance behaves correctly. The worked question below makes that reasoning concrete: Apply recognition over recall to command discovery. The conclusion is Expose relevant commands in menus, search, or contextual actions with labels and shortcuts; do not require users to remember hidden syntax before they can proceed.

Common miss

The common mistake is to memorize the visible procedure while dropping its precondition. I write the assumption beside the method and test one boundary case before trusting the result.

Solved chapter practice

Practice. Apply recognition over recall to command discovery.
Solution. Expose relevant commands in menus, search, or contextual actions with labels and shortcuts; do not require users to remember hidden syntax before they can proceed.

Practice 2. Give a short oral explanation of models of action and cognition that includes its model, one assumption, one failure mode, and one comparison with a neighboring method.
Solution. A strong answer begins with the model in the blue box, names the assumption that licenses the method, uses the common miss above as the failure mode, and chooses the comparison according to the decision test in the deep notes. The goal is not one memorized sentence; it is a four-part explanation I can reproduce on a new example.

Mastery practice A. Reconstruct models of action and cognition from a blank page. Include the central definition, the governing rule, one required assumption, and one boundary case.
Solution. Start from the blue idea box, but rewrite it in my own words. Identify the objects and state, name the rule before using it, copy the relevant assumption from the details audit, and choose a boundary where that assumption becomes equality or fails. A complete solution explains why each part belongs; a list of vocabulary is not enough.

Mastery practice B. A classmate gets a polished-looking answer to a models of action and cognition problem but never states a model or checks the result. Write the shortest useful review of that solution.
Solution. I would ask four concrete questions: What exactly are the inputs and desired output? Which rule licenses the main step? Which assumption could invalidate it? Which independent check supports the conclusion? If the answer cannot survive those four questions, I treat the numerical or verbal result as unverified.

CHAPTER 4 Information architecture and interaction flows

The idea

Navigation, hierarchy, labels, search, progressive disclosure, state transitions, error recovery, and feedback organize a product. Task flows should include alternate, empty, loading, permission, offline, and failure states.

Rebuild the chapter from first principles

The claim I need to own is this: Navigation, hierarchy, labels, search, progressive disclosure, state transitions, error recovery, and feedback organize a product. Task flows should include alternate, empty, loading, permission, offline, and failure states. I do not count that as learned just because the sentence looks familiar. I should be able to name the objects, state what changes and what stays fixed, recover the rule from the definitions, and explain why the result is allowed.

The move

My working sequence

1. **Translate.** Rewrite the prompt in the language of information architecture and interaction flows. List the known quantities, the unknown, the units or types, and any hidden constraint.
2. **Choose.** State the theorem, model, algorithm, accounting rule, or physical law before substituting values. This is where I make the assumption visible.
3. **Execute.** Work one justified step at a time and keep intermediate state readable enough that I can audit it later.
4. **Verify.** Check a boundary case, sign, unit, invariant, order of magnitude, or alternative derivation. Then translate the result back into an ordinary sentence.

Details I would refuse to skip

- **Definitions:** I should be able to define every technical noun in the chapter without using that same noun in the definition.
- **Assumptions:** I write the condition that makes the method legal beside the method itself. A correct procedure under the wrong assumptions is still a wrong answer.
- **Representation:** I move freely among words, equations or code, a diagram, and a small numerical example. If one representation disagrees with another, that disagreement is useful evidence.
- **Failure modes:** I keep the nearest counterexample in mind: the boundary where the rule changes, the invariant breaks, or the model stops matching reality.
- **Communication:** My final answer names what the result means and what it does not establish. I do not let notation do the explanatory work for me.

Sanity check

Closed-note check. Can I teach information architecture and interaction flows on a blank page, derive or reconstruct its main rule, work a fresh example, and diagnose a deliberately broken solution? If I can only recognize the finished work, I am not done.

How this chapter connects

I read this chapter as the bridge from models of action and cognition to prototyping and design systems. The earlier material supplies the language and constraints; this chapter adds a new reasoning move; the later material asks me to use that move without being told its name. That connection matters because exams and real projects mix chapters on purpose.

What I need to be able to do

For this chapter, I should be able to define the objects and state involved, state the governing invariant or contract, trace a small example without hand-waving, and explain which assumption makes the method valid. I should also be able to compare this model with the nearest alternative and say what failure would make me switch approaches.

Deep notes

- **Model.** Navigation, hierarchy, labels, search, progressive disclosure, state transitions, error recovery, and feedback organize a product.
- **Mechanism.** Task flows should include alternate, empty, loading, permission, offline, and failure states.
- **Boundary.** The useful test is whether I can apply information architecture and interaction flows to a new case and explain the assumption that makes the answer valid.
- **Decision test.** I should be able to say when information architecture and interaction flows is the right model, when its assumptions fail, and what evidence would change my choice.

How I actually think about it

I design the unhappy path beside the happy path because real products spend a lot of time there.

The easy version

Navigation, hierarchy, labels, search, progressive disclosure, state transitions, error recovery, and feedback organize a product.

Worked example

A useful way to work this chapter is to make the smallest concrete instance, write the state or contract explicitly, and then scale the reasoning only after that instance behaves correctly. The worked question below makes that reasoning concrete: Improve a destructive delete flow. The conclusion is Use clear object naming, consequences, undo where feasible, a confirmation

proportional to harm, immediate feedback, and recovery rather than a generic 'Are you sure?' alone.

Common miss

The common mistake is to memorize the visible procedure while dropping its precondition. I write the assumption beside the method and test one boundary case before trusting the result.

Solved chapter practice

Practice. Improve a destructive delete flow.
Solution. Use clear object naming, consequences, undo where feasible, a confirmation proportional to harm, immediate feedback, and recovery rather than a generic 'Are you sure?' alone.

Practice 2. Give a short oral explanation of information architecture and interaction flows that includes its model, one assumption, one failure mode, and one comparison with a neighboring method.
Solution. A strong answer begins with the model in the blue box, names the assumption that licenses the method, uses the common miss above as the failure mode, and chooses the comparison according to the decision test in the deep notes. The goal is not one memorized sentence; it is a four-part explanation I can reproduce on a new example.

Mastery practice A. Reconstruct information architecture and interaction flows from a blank page. Include the central definition, the governing rule, one required assumption, and one boundary case.
Solution. Start from the blue idea box, but rewrite it in my own words. Identify the objects and state, name the rule before using it, copy the relevant assumption from the details audit, and choose a boundary where that assumption becomes equality or fails. A complete solution explains why each part belongs; a list of vocabulary is not enough.

Mastery practice B. A classmate gets a polished-looking answer to a information architecture and interaction flows problem but never states a model or checks the result. Write the shortest useful review of that solution.
Solution. I would ask four concrete questions: What exactly are the inputs and desired output? Which rule licenses the main step? Which assumption could invalidate it? Which independent check supports the conclusion? If the answer cannot survive those four questions, I treat the numerical or verbal result as unverified.

CHAPTER 5 Prototyping and design systems

The idea

Sketches explore structure cheaply; wireframes test flow; high-fidelity prototypes test detailed interaction and perception. Components, tokens, states, and content rules create consistency without preventing context-specific choices.

Rebuild the chapter from first principles

The claim I need to own is this: Sketches explore structure cheaply; wireframes test flow; high-fidelity prototypes test detailed interaction and perception. Components, tokens, states, and content rules create consistency without preventing context-specific choices. I do not count that as learned just because the sentence looks familiar. I should be able to name the objects, state what changes and what stays fixed, recover the rule from the definitions, and explain why the result is allowed.

The move

My working sequence

1. **Translate.** Rewrite the prompt in the language of prototyping and design systems. List the known quantities, the unknown, the units or types, and any hidden constraint.
2. **Choose.** State the theorem, model, algorithm, accounting rule, or physical law before substituting values. This is where I make the assumption visible.
3. **Execute.** Work one justified step at a time and keep intermediate state readable enough that I can audit it later.
4. **Verify.** Check a boundary case, sign, unit, invariant, order of magnitude, or alternative derivation. Then translate the result back into an ordinary sentence.

Details I would refuse to skip

- **Definitions:** I should be able to define every technical noun in the chapter without using that same noun in the definition.
- **Assumptions:** I write the condition that makes the method legal beside the method itself. A correct procedure under the wrong assumptions is still a wrong answer.
- **Representation:** I move freely among words, equations or code, a diagram, and a small numerical example. If one representation disagrees with another, that disagreement is useful evidence.
- **Failure modes:** I keep the nearest counterexample in mind: the boundary where the rule changes, the invariant breaks, or the model stops matching reality.
- **Communication:** My final answer names what the result means and what it does not establish. I do not let notation do the explanatory work for me.

Sanity check

Closed-note check. Can I teach prototyping and design systems on a blank page, derive or reconstruct its main rule, work a fresh example, and diagnose a deliberately broken solution? If I can only recognize the finished work, I am not done.

How this chapter connects

I read this chapter as the bridge from information architecture and interaction flows to usability evaluation. The earlier material supplies the language and constraints; this chapter adds a new reasoning move; the later material asks me to use that move without being told its name. That connection matters because exams and real projects mix chapters on purpose.

What I need to be able to do

For this chapter, I should be able to define the objects and state involved, state the governing invariant or contract, trace a small example without hand-waving, and explain which assumption makes the method valid. I should also be able to compare this model with the nearest alternative and say what failure would make me switch approaches.

Deep notes

- **Model.** Sketches explore structure cheaply; wireframes test flow; high-fidelity prototypes test detailed interaction and perception.
- **Mechanism.** Components, tokens, states, and content rules create consistency without preventing context-specific choices.
- **Boundary.** The useful test is whether I can apply prototyping and design systems to a new case and explain the assumption that makes the answer valid.
- **Decision test.** I should be able to say when prototyping and design systems is the right model, when its assumptions fail, and what evidence would change my choice.

How I actually think about it

I match prototype fidelity to the question; polished pixels too early make weak ideas expensive to discard.

The easy version

Sketches explore structure cheaply; wireframes test flow; high-fidelity prototypes test detailed interaction and perception.

Worked example

A useful way to work this chapter is to make the smallest concrete instance, write the state or contract explicitly, and then scale the reasoning only after that instance behaves correctly. The worked question below makes that reasoning concrete: Which prototype tests checkout navigation? The conclusion is Use a clickable low- or medium-fidelity flow with realistic labels and data. Visual polish is unnecessary unless trust or brand perception is the research question.

Common miss

The common mistake is to memorize the visible procedure while dropping its precondition. I write the assumption beside the method and test one boundary case before trusting the result.

Solved chapter practice

Practice. Which prototype tests checkout navigation?
Solution. Use a clickable low- or medium-fidelity flow with realistic labels and data. Visual polish is unnecessary unless trust or brand perception is the research question.

Practice 2. Give a short oral explanation of prototyping and design systems that includes its

model, one assumption, one failure mode, and one comparison with a neighboring method.
Solution. A strong answer begins with the model in the blue box, names the assumption that licenses the method, uses the common miss above as the failure mode, and chooses the comparison according to the decision test in the deep notes. The goal is not one memorized sentence; it is a four-part explanation I can reproduce on a new example.

Mastery practice A. Reconstruct prototyping and design systems from a blank page. Include the central definition, the governing rule, one required assumption, and one boundary case.
Solution. Start from the blue idea box, but rewrite it in my own words. Identify the objects and state, name the rule before using it, copy the relevant assumption from the details audit, and choose a boundary where that assumption becomes equality or fails. A complete solution explains why each part belongs; a list of vocabulary is not enough.

Mastery practice B. A classmate gets a polished-looking answer to a prototyping and design systems problem but never states a model or checks the result. Write the shortest useful review of that solution.
Solution. I would ask four concrete questions: What exactly are the inputs and desired output? Which rule licenses the main step? Which assumption could invalidate it? Which independent check supports the conclusion? If the answer cannot survive those four questions, I treat the numerical or verbal result as unverified.

CHAPTER 6 Usability evaluation

The idea

Formative evaluation improves a design; summative evaluation estimates performance. Think-aloud tests reveal breakdowns, while time, errors, completion, SUS-style ratings, and qualitative evidence answer different questions.

Rebuild the chapter from first principles

The claim I need to own is this: Formative evaluation improves a design; summative evaluation estimates performance. Think-aloud tests reveal breakdowns, while time, errors, completion, SUS-style ratings, and qualitative evidence answer different questions. I do not count that as learned just because the sentence looks familiar. I should be able to name the objects, state what changes and what stays fixed, recover the rule from the definitions, and explain why the result is allowed.

The move

My working sequence

1. **Translate.** Rewrite the prompt in the language of usability evaluation. List the known quantities, the unknown, the units or types, and any hidden constraint.
2. **Choose.** State the theorem, model, algorithm, accounting rule, or physical law before substituting values. This is where I make the assumption visible.

3. **Execute.** Work one justified step at a time and keep intermediate state readable enough that I can audit it later.
4. **Verify.** Check a boundary case, sign, unit, invariant, order of magnitude, or alternative derivation. Then translate the result back into an ordinary sentence.

Details I would refuse to skip

- **Definitions:** I should be able to define every technical noun in the chapter without using that same noun in the definition.
- **Assumptions:** I write the condition that makes the method legal beside the method itself. A correct procedure under the wrong assumptions is still a wrong answer.
- **Representation:** I move freely among words, equations or code, a diagram, and a small numerical example. If one representation disagrees with another, that disagreement is useful evidence.
- **Failure modes:** I keep the nearest counterexample in mind: the boundary where the rule changes, the invariant breaks, or the model stops matching reality.
- **Communication:** My final answer names what the result means and what it does not establish. I do not let notation do the explanatory work for me.

Sanity check

Closed-note check. Can I teach usability evaluation on a blank page, derive or reconstruct its main rule, work a fresh example, and diagnose a deliberately broken solution? If I can only recognize the finished work, I am not done.

How this chapter connects

I read this chapter as the bridge from prototyping and design systems to accessibility and inclusive interaction. The earlier material supplies the language and constraints; this chapter adds a new reasoning move; the later material asks me to use that move without being told its name. That connection matters because exams and real projects mix chapters on purpose.

What I need to be able to do

For this chapter, I should be able to define the objects and state involved, state the governing invariant or contract, trace a small example without hand-waving, and explain which assumption makes the method valid. I should also be able to compare this model with the nearest alternative and say what failure would make me switch approaches.

Deep notes

- **Model.** Formative evaluation improves a design; summative evaluation estimates performance.
- **Mechanism.** Think-aloud tests reveal breakdowns, while time, errors, completion, SUS-style ratings, and qualitative evidence answer different questions.

- **Boundary.** The useful test is whether I can apply usability evaluation to a new case and explain the assumption that makes the answer valid.
- **Decision test.** I should be able to say when usability evaluation is the right model, when its assumptions fail, and what evidence would change my choice.

How I actually think about it

I define the failure signal before the session so I do not reinterpret every struggle as success.

The easy version

Formative evaluation improves a design; summative evaluation estimates performance.

Worked example

A useful way to work this chapter is to make the smallest concrete instance, write the state or contract explicitly, and then scale the reasoning only after that instance behaves correctly. The worked question below makes that reasoning concrete: Design a five-user formative test. The conclusion is Recruit relevant participants, give neutral scenario tasks, avoid coaching, record completion and breakdowns, debrief, group severity and frequency, revise, and test again.

Common miss

The common mistake is to memorize the visible procedure while dropping its precondition. I write the assumption beside the method and test one boundary case before trusting the result.

Solved chapter practice

Practice. Design a five-user formative test.
Solution. Recruit relevant participants, give neutral scenario tasks, avoid coaching, record completion and breakdowns, debrief, group severity and frequency, revise, and test again.

Practice 2. Give a short oral explanation of usability evaluation that includes its model, one assumption, one failure mode, and one comparison with a neighboring method.
Solution. A strong answer begins with the model in the blue box, names the assumption that licenses the method, uses the common miss above as the failure mode, and chooses the comparison according to the decision test in the deep notes. The goal is not one memorized sentence; it is a four-part explanation I can reproduce on a new example.

Mastery practice A. Reconstruct usability evaluation from a blank page. Include the central definition, the governing rule, one required assumption, and one boundary case.
Solution. Start from the blue idea box, but rewrite it in my own words. Identify the objects and state, name the rule before using it, copy the relevant assumption from the details audit, and choose a boundary where that assumption becomes equality or fails. A complete solution explains why each part belongs; a list of vocabulary is not enough.

Mastery practice B. A classmate gets a polished-looking answer to a usability evaluation problem but never states a model or checks the result. Write the shortest useful review of that solution.
Solution. I would ask four concrete questions: What exactly are the inputs and desired output? Which rule licenses the main step? Which assumption could invalidate it? Which independent

check supports the conclusion? If the answer cannot survive those four questions, I treat the numerical or verbal result as unverified.

CHAPTER 7 Accessibility and inclusive interaction

The idea

Accessibility covers perceivability, operability, understandability, and robustness across vision, hearing, motor, cognitive, language, and situational differences. Semantic structure, keyboard access, focus, contrast, captions, alternatives, and plain content are foundational.

Rebuild the chapter from first principles

The claim I need to own is this: Accessibility covers perceivability, operability, understandability, and robustness across vision, hearing, motor, cognitive, language, and situational differences. Semantic structure, keyboard access, focus, contrast, captions, alternatives, and plain content are foundational. I do not count that as learned just because the sentence looks familiar. I should be able to name the objects, state what changes and what stays fixed, recover the rule from the definitions, and explain why the result is allowed.

The move

My working sequence

1. **Translate.** Rewrite the prompt in the language of accessibility and inclusive interaction. List the known quantities, the unknown, the units or types, and any hidden constraint.
2. **Choose.** State the theorem, model, algorithm, accounting rule, or physical law before substituting values. This is where I make the assumption visible.
3. **Execute.** Work one justified step at a time and keep intermediate state readable enough that I can audit it later.
4. **Verify.** Check a boundary case, sign, unit, invariant, order of magnitude, or alternative derivation. Then translate the result back into an ordinary sentence.

Details I would refuse to skip

- **Definitions:** I should be able to define every technical noun in the chapter without using that same noun in the definition.
- **Assumptions:** I write the condition that makes the method legal beside the method itself. A correct procedure under the wrong assumptions is still a wrong answer.
- **Representation:** I move freely among words, equations or code, a diagram, and a small numerical example. If one representation disagrees with another, that disagreement is useful evidence.
- **Failure modes:** I keep the nearest counterexample in mind: the boundary where the rule changes,

the invariant breaks, or the model stops matching reality.

- **Communication:** My final answer names what the result means and what it does not establish. I do not let notation do the explanatory work for me.

Sanity check

Closed-note check. Can I teach accessibility and inclusive interaction on a blank page, derive or reconstruct its main rule, work a fresh example, and diagnose a deliberately broken solution? If I can only recognize the finished work, I am not done.

How this chapter connects

I read this chapter as the bridge from usability evaluation to experiments, ethics, and product judgment. The earlier material supplies the language and constraints; this chapter adds a new reasoning move; the later material asks me to use that move without being told its name. That connection matters because exams and real projects mix chapters on purpose.

What I need to be able to do

For this chapter, I should be able to define the objects and state involved, state the governing invariant or contract, trace a small example without hand-waving, and explain which assumption makes the method valid. I should also be able to compare this model with the nearest alternative and say what failure would make me switch approaches.

Deep notes

- **Model.** Accessibility covers perceivability, operability, understandability, and robustness across vision, hearing, motor, cognitive, language, and situational differences.
- **Mechanism.** Semantic structure, keyboard access, focus, contrast, captions, alternatives, and plain content are foundational.
- **Boundary.** The useful test is whether I can apply accessibility and inclusive interaction to a new case and explain the assumption that makes the answer valid.
- **Decision test.** I should be able to say when accessibility and inclusive interaction is the right model, when its assumptions fail, and what evidence would change my choice.

How I actually think about it

I test with keyboard and screen-reader semantics before calling accessibility a final checklist.

The easy version

Accessibility covers perceivability, operability, understandability, and robustness across vision, hearing, motor, cognitive, language, and situational differences.

Worked example

A useful way to work this chapter is to make the smallest concrete instance, write the state or contract explicitly, and then scale the reasoning only after that instance behaves correctly. The worked question below makes that reasoning concrete: Why is placeholder text not a label? The conclusion is It disappears on input, often has weak contrast, and may not provide a persistent accessible name. Use an explicit associated label and separate help or error text.

Common miss

The common mistake is to memorize the visible procedure while dropping its precondition. I write the assumption beside the method and test one boundary case before trusting the result.

Solved chapter practice

Practice. Why is placeholder text not a label?
Solution. It disappears on input, often has weak contrast, and may not provide a persistent accessible name. Use an explicit associated label and separate help or error text.

Practice 2. Give a short oral explanation of accessibility and inclusive interaction that includes its model, one assumption, one failure mode, and one comparison with a neighboring method.
Solution. A strong answer begins with the model in the blue box, names the assumption that licenses the method, uses the common miss above as the failure mode, and chooses the comparison according to the decision test in the deep notes. The goal is not one memorized sentence; it is a four-part explanation I can reproduce on a new example.

Mastery practice A. Reconstruct accessibility and inclusive interaction from a blank page. Include the central definition, the governing rule, one required assumption, and one boundary case.
Solution. Start from the blue idea box, but rewrite it in my own words. Identify the objects and state, name the rule before using it, copy the relevant assumption from the details audit, and choose a boundary where that assumption becomes equality or fails. A complete solution explains why each part belongs; a list of vocabulary is not enough.

Mastery practice B. A classmate gets a polished-looking answer to a accessibility and inclusive interaction problem but never states a model or checks the result. Write the shortest useful review of that solution.
Solution. I would ask four concrete questions: What exactly are the inputs and desired output? Which rule licenses the main step? Which assumption could invalidate it? Which independent check supports the conclusion? If the answer cannot survive those four questions, I treat the numerical or verbal result as unverified.

CHAPTER 8 Experiments, ethics, and product judgment

The idea

A/B tests estimate causal effects when assignment, metrics, sample size, duration, interference, and stopping rules are sound. Ethics includes consent, manipulation, dark patterns, privacy, representational harm, and who benefits.

Rebuild the chapter from first principles

The claim I need to own is this: A/B tests estimate causal effects when assignment, metrics, sample size, duration, interference, and stopping rules are sound. Ethics includes consent, manipulation, dark patterns, privacy, representational harm, and who benefits. I do not count that as learned just because the sentence looks familiar. I should be able to name the objects, state what changes and what stays fixed, recover the rule from the definitions, and explain why the result is allowed.

The move

My working sequence

1. **Translate.** Rewrite the prompt in the language of experiments, ethics, and product judgment. List the known quantities, the unknown, the units or types, and any hidden constraint.
2. **Choose.** State the theorem, model, algorithm, accounting rule, or physical law before substituting values. This is where I make the assumption visible.
3. **Execute.** Work one justified step at a time and keep intermediate state readable enough that I can audit it later.
4. **Verify.** Check a boundary case, sign, unit, invariant, order of magnitude, or alternative derivation. Then translate the result back into an ordinary sentence.

Details I would refuse to skip

- **Definitions:** I should be able to define every technical noun in the chapter without using that same noun in the definition.
- **Assumptions:** I write the condition that makes the method legal beside the method itself. A correct procedure under the wrong assumptions is still a wrong answer.
- **Representation:** I move freely among words, equations or code, a diagram, and a small numerical example. If one representation disagrees with another, that disagreement is useful evidence.
- **Failure modes:** I keep the nearest counterexample in mind: the boundary where the rule changes, the invariant breaks, or the model stops matching reality.
- **Communication:** My final answer names what the result means and what it does not establish. I do not let notation do the explanatory work for me.

Sanity check

Closed-note check. Can I teach experiments, ethics, and product judgment on a blank page, derive or reconstruct its main rule, work a fresh example, and diagnose a deliberately broken solution? If I can only recognize the finished work, I am not done.

How this chapter connects

I read this chapter as the bridge from accessibility and inclusive interaction to the cumulative course model. The earlier material supplies the language and constraints; this chapter adds a new reasoning move; the later material asks me to use that move without being told its name. That connection matters because exams and real projects mix chapters on purpose.

What I need to be able to do

For this chapter, I should be able to define the objects and state involved, state the governing invariant or contract, trace a small example without hand-waving, and explain which assumption makes the method valid. I should also be able to compare this model with the nearest alternative and say what failure would make me switch approaches.

Deep notes

- **Model.** A/B tests estimate causal effects when assignment, metrics, sample size, duration, interference, and stopping rules are sound.
- **Mechanism.** Ethics includes consent, manipulation, dark patterns, privacy, representational harm, and who benefits.
- **Boundary.** The useful test is whether I can apply experiments, ethics, and product judgment to a new case and explain the assumption that makes the answer valid.
- **Decision test.** I should be able to say when experiments, ethics, and product judgment is the right model, when its assumptions fail, and what evidence would change my choice.

How I actually think about it

A metric increase is not automatically a product improvement; I ask what behavior created it and who paid the cost.

The easy version

A/B tests estimate causal effects when assignment, metrics, sample size, duration, interference, and stopping rules are sound.

Worked example

A useful way to work this chapter is to make the smallest concrete instance, write the state or contract explicitly, and then scale the reasoning only after that instance behaves correctly. The worked question below makes that reasoning concrete: Why is click-through an incomplete success

metric? The conclusion is It can reward confusion or manipulation. Pair it with task success, retention, satisfaction, error, accessibility, and harm guardrails tied to the actual user goal.

Common miss

The common mistake is to memorize the visible procedure while dropping its precondition. I write the assumption beside the method and test one boundary case before trusting the result.

Solved chapter practice

Practice. Why is click-through an incomplete success metric?
Solution. It can reward confusion or manipulation. Pair it with task success, retention, satisfaction, error, accessibility, and harm guardrails tied to the actual user goal.

Practice 2. Give a short oral explanation of experiments, ethics, and product judgment that includes its model, one assumption, one failure mode, and one comparison with a neighboring method.
Solution. A strong answer begins with the model in the blue box, names the assumption that licenses the method, uses the common miss above as the failure mode, and chooses the comparison according to the decision test in the deep notes. The goal is not one memorized sentence; it is a four-part explanation I can reproduce on a new example.

Mastery practice A. Reconstruct experiments, ethics, and product judgment from a blank page. Include the central definition, the governing rule, one required assumption, and one boundary case.
Solution. Start from the blue idea box, but rewrite it in my own words. Identify the objects and state, name the rule before using it, copy the relevant assumption from the details audit, and choose a boundary where that assumption becomes equality or fails. A complete solution explains why each part belongs; a list of vocabulary is not enough.

Mastery practice B. A classmate gets a polished-looking answer to a experiments, ethics, and product judgment problem but never states a model or checks the result. Write the shortest useful review of that solution.
Solution. I would ask four concrete questions: What exactly are the inputs and desired output? Which rule licenses the main step? Which assumption could invalidate it? Which independent check supports the conclusion? If the answer cannot survive those four questions, I treat the numerical or verbal result as unverified.

Final study check

I should be able to close this packet and reconstruct the core models, not merely recognize their names. My final check is to explain one complete problem aloud: what the inputs mean, which invariant or contract controls the work, why the method is legal, what it costs, how it can fail, and what evidence would convince another engineer.