

Week 10, Part 1:

Cognitive Walkthrough



What is it?

- A method of **task-centred design**.
- Focuses on real, complete and representative tasks.
- Cognitive walkthrough is a method of task-centred evaluation.

What it's good for:

- Questioning assumptions about what users will be thinking
- Identifying controls that may be missing or hard to find
- Finding places that have inadequate feedback
- Suggesting difficulties users may have with labels or prompts

The purpose of cognitive walkthrough

- Focuses on problems that people using the interface will have when they first use it, without training
- Most effective if the designers can construct a mental picture of actual use

The caveat:

Cognitive walkthrough is not for evaluating the use of a system over time.

It can't tell us anything about how a person's skill will progress from a beginner to intermediate!

Cognitive walkthrough preparation:

1. A description or a **prototype** of the interface
2. A **task description** for a representative task
3. A complete **list of the actions** needed to complete the task
4. An idea of who the users will be and what kind of experience they'll bring to the job

Cognitive walkthrough procedure:

1. Define inputs
2. Get analysts
3. Step through action sequences
for each task
4. Record important information
5. Revise UI



1. Define the inputs.

Answer the following:

- Who are the users?
- What are the tasks?
- What are the action sequences for the tasks?

You'll also need a prototype to analyse (or at least a very good description!)



2. Get analysts.

You don't need actual users!

You can evaluate the interface by imagining the behaviour of an entire class of users, not one unique user.

A typical developer can perform this analysis.

(But they should have a knowledge of cognitive science to understand people's limitations)




3. Step through actions.

Will users know what to do?

Will users see how to do it?

Will users understand from feedback whether their actions are correct or not?



4. Record important information.

User knowledge (just before and just after action)

Assumptions about users

Side issues and design changes

Credible reasons for success/failure: Why would a user select or not select the correct action?



5. Revise the UI

If the user fails to select the right action:

- Eliminate that step
- Prompt them
- Change it so they know they can try the action
- If it's not obvious what to do, make it more obvious!



5. Revise the UI

If the user doesn't know which action is correct:

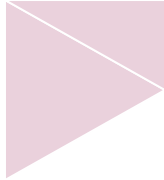
- Label it
- Check the sequence of actions - does it make sense?

If user can't tell where they are in the process:

- Give them feedback, say what happened

Let's try it!

forms.google.com



Our task:

Make a new form

Give it a title

Add three questions: One selector, one short answer, one scale

Change the background

Share a link to the form

Cognitive walkthrough: What is it good for?

Assesses the learnability
of an interface

Identifies specific
problems with design

Does not require users to
get involved

Cognitive walkthrough: What is it not good for?

Can't tell you how
beginners become
experts

Can only be used to
evaluate an interface as
it's used by a first-time
user

What kinds of problems can cognitive walkthrough find?

Severe problems - Fairly good, comparable to other techniques

Content-related problems - Comparable for Consistency, worse for Recurrence

Scope - Finds problems that are more specific rather than general

Week 10, Part 2:

Heuristic Evaluation

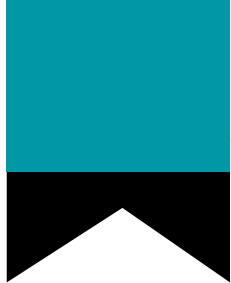


What is it?

- A system of evaluation for user interfaces that finds usability problems
- Developed by Jakob Nielsen and Rolf Molich in the early 1990s

What is it?

- It's a **fast** and **cheap** way to find usability problems.



heu•ris•tic (hyōō-rĭs'tĭk)

Of or relating to a usually speculative formulation serving as a **guide in the investigation** or solution of a problem

Of or constituting an educational method in which learning **takes place through discoveries that result from investigations** made by the student.

What is “usability”?

Ease of learning

Recall

Productivity

Minimal errors

High user satisfaction

Heuristic evaluation offers:

“Discount usability
engineering”



Everyone loves a discount.



(But what does this mean?)



IT'S CHEAP!

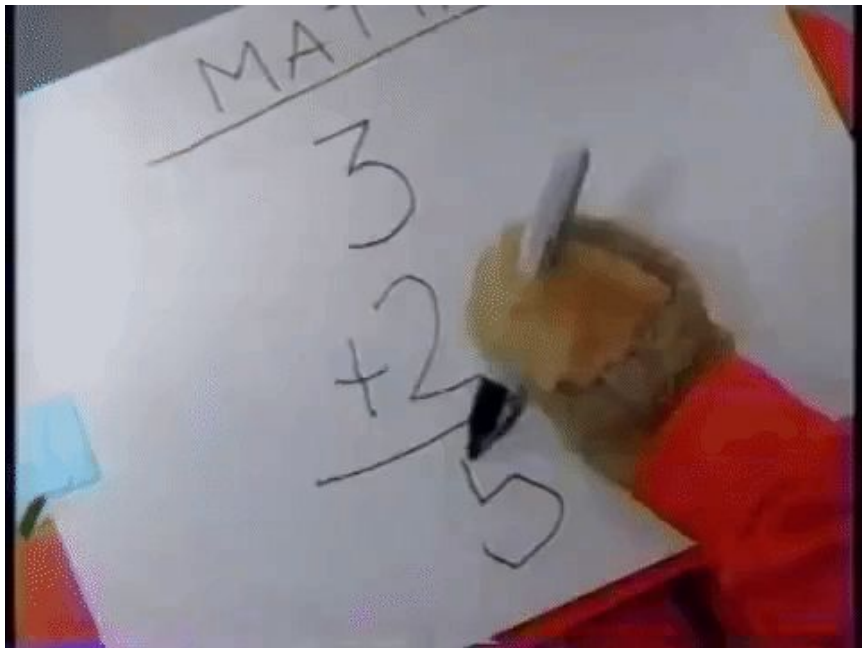
No special labs needed!

If you're careful, it's
really really cheap!



IT'S FAST!

You can do the whole
process in about a day!




IT'S EASY!

So easy anyone can
learn it! You don't need to
be an expert!

Heuristic evaluation sounds great!



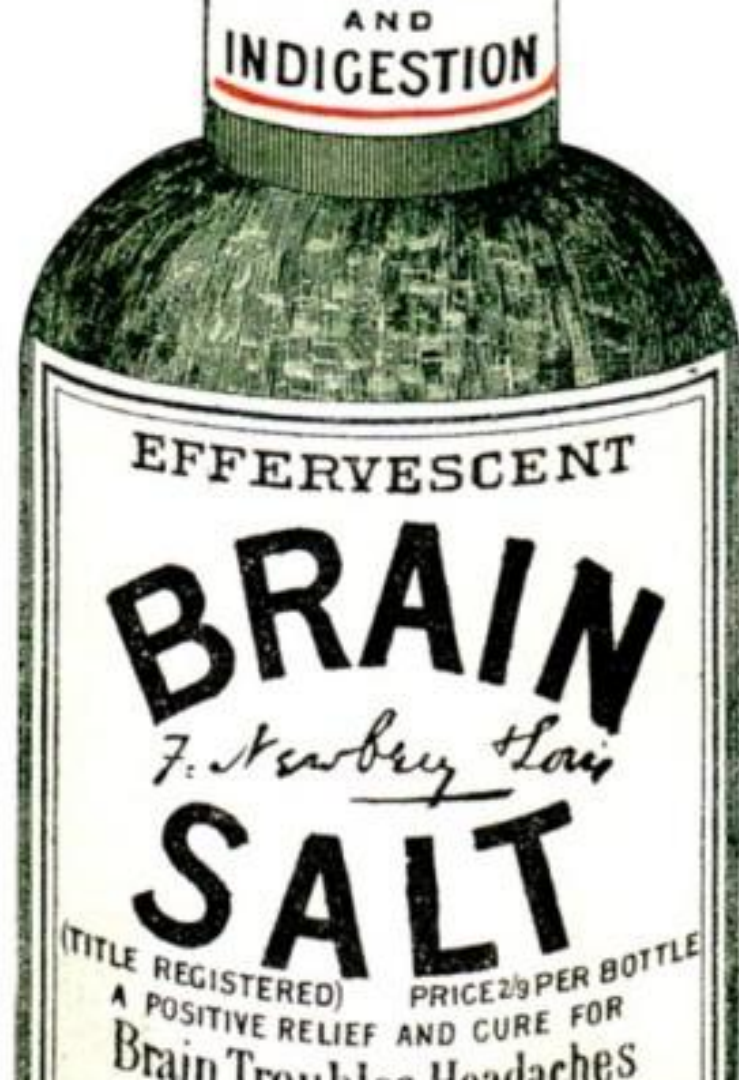
A man with a mustache and glasses, wearing a dark suit and a wide-brimmed hat, holds a bottle of Snake Oil. He is pointing at the label with his right index finger. The background is a solid dark purple color.

**SNAKE
OIL**

For Rheumatism
For Spasmodic
For Building Blood

But hang on.
There's something you
should know.

IT IS NOT
A CURE-ALL.



**Heuristic
evaluation is
not a cure-all.**

**Heuristic
evaluation is
not a cure-all.**

But it sure is
useful
when you're
looking for a
fast and cheap
way to find
usability
problems.



(Heuristic evaluation is not a cure-all.)



How it's done

- Pre-evaluation training
- Evaluate
- Collate the results
- Rate severity
- Feed this back into the design



Evaluate

A group of 3-5 evaluators:

- Evaluate according to ten **heuristics**

The 10 heuristics (or usability principles):

1. Visibility of system status

The system should keep the user informed, with appropriate feedback.

2. Match between system and real world.

System should use language and symbols familiar to the target user. Should be logical and orderly (according to the user).

3. User control and freedom

The user will often do something wrong and need a way to go back.

4. Consistency and standards.

User should not have to wonder if words, symbols and so on mean the same thing. Interface should follow the conventions on the platform.

5. Error prevention.

Good error messages ... or
good design that prevents
errors from occurring in the first
place.

6. Recognition rather than recall.

Make objects, options, etc visible. The user shouldn't have to try to remember one part of the interface to use another.

7. Flexibility and efficiency of use.

Can some actions be sped up, so the user can reduce the time it takes to use the interface once they're good at it? Allow users to tailor frequent actions.

**8.
Aesthetic
and
minimalist
design.**

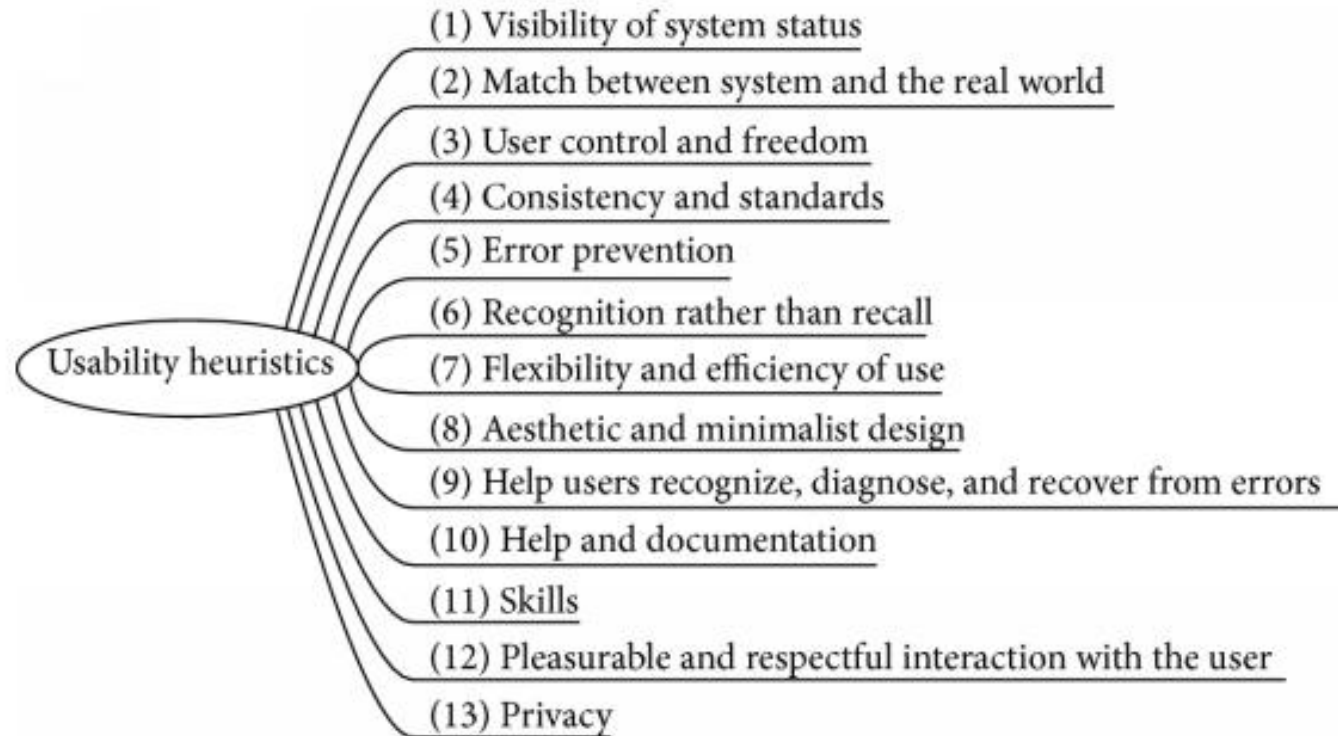
No extraneous information or text, because all extraneous elements distract from the relevant information.

**9.
Help users
recognise
and recover
from errors.**

If something goes wrong, can the user recover?

10. Help and documenta- tion

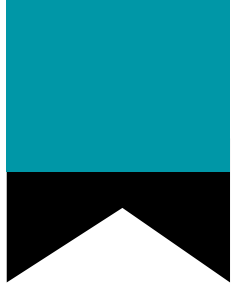
Ideally the system should be so well designed that it doesn't need documentation, but in case it does, it should be present and easy to find.



Heuristic Evaluation on Mobile Interfaces: A New Checklist

Gómez, Caballero, Sevillano (2014)

The severity ratings



Severity ratings

These are ways of prioritising or giving weight to problems.

We assign severity based on the **frequency**, **impact**, and **persistence** of the problems.

frequency

impact

persistence

0

There is **no violation**, or this criteria doesn't
apply to this system.

1

This problem is **cosmetic**. You could fix it if you want, but the system is still perfectly usable if it's still there.

2

Minor violation. Definitely an issue, but is a low priority.

3

Major usability problem. This needs to be fixed
ASAP!

4











CATASTROPHE!

Can't release without fixing this. It's URGENT.

So let's do it.

QMPlus.

10 Usability Heuristics

-  Visibility of system status
-  Match between system and the real world
-  User control and freedom
-  Consistency and standards
-  Error prevention
-  Recognition rather than recall
-  Flexibility and efficiency of use
-  Aesthetic and minimalist design
-  Helps users recognise, diagnose, and recover from errors
-  Help and documentation

**What violations (at
what severity) did
you find?**

Extremely.

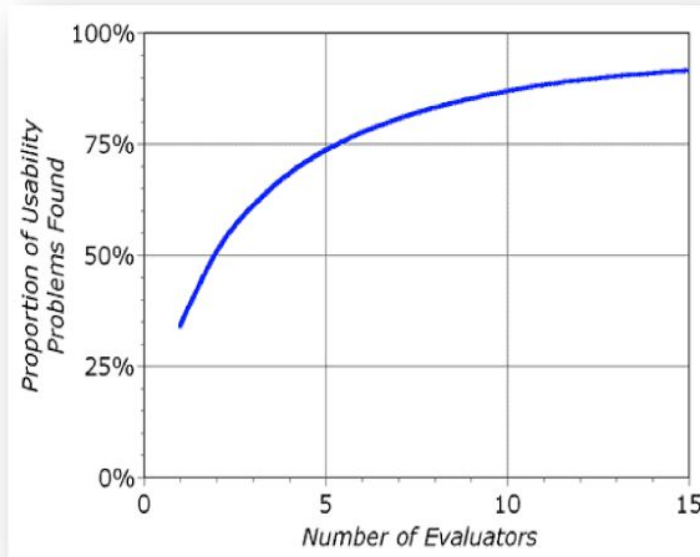
Nielsen calculated that for a cost of \$10,500, the benefit was \$500,000.

A single evaluator gets poor results (found 35% of problems), but 3-5 evaluators found 75% of problems.

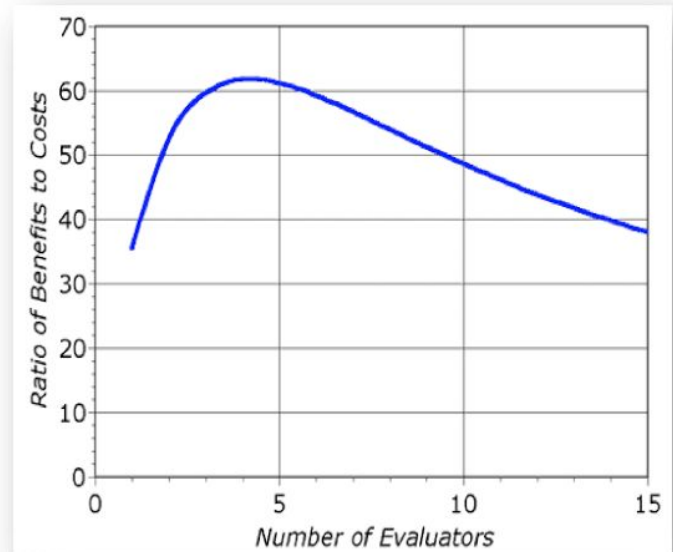
“If we get great results with 3-5 evaluators, just imagine the results we’d get with lots more evaluators!”

... except that's not how it works.

problems found



benefits / cost



The good part:

It's cheap, fast, and easy; doesn't require experts; huge benefits when compared to costs; finds 75% of problems.

The drawbacks

1. It's not task-focused.

The drawbacks

2. It doesn't use actual people.

The drawbacks

3. It isn't rigorous.

In summary:

1. Heuristic evaluation is a cheap, fast and easy way to find usability problems.

In summary:

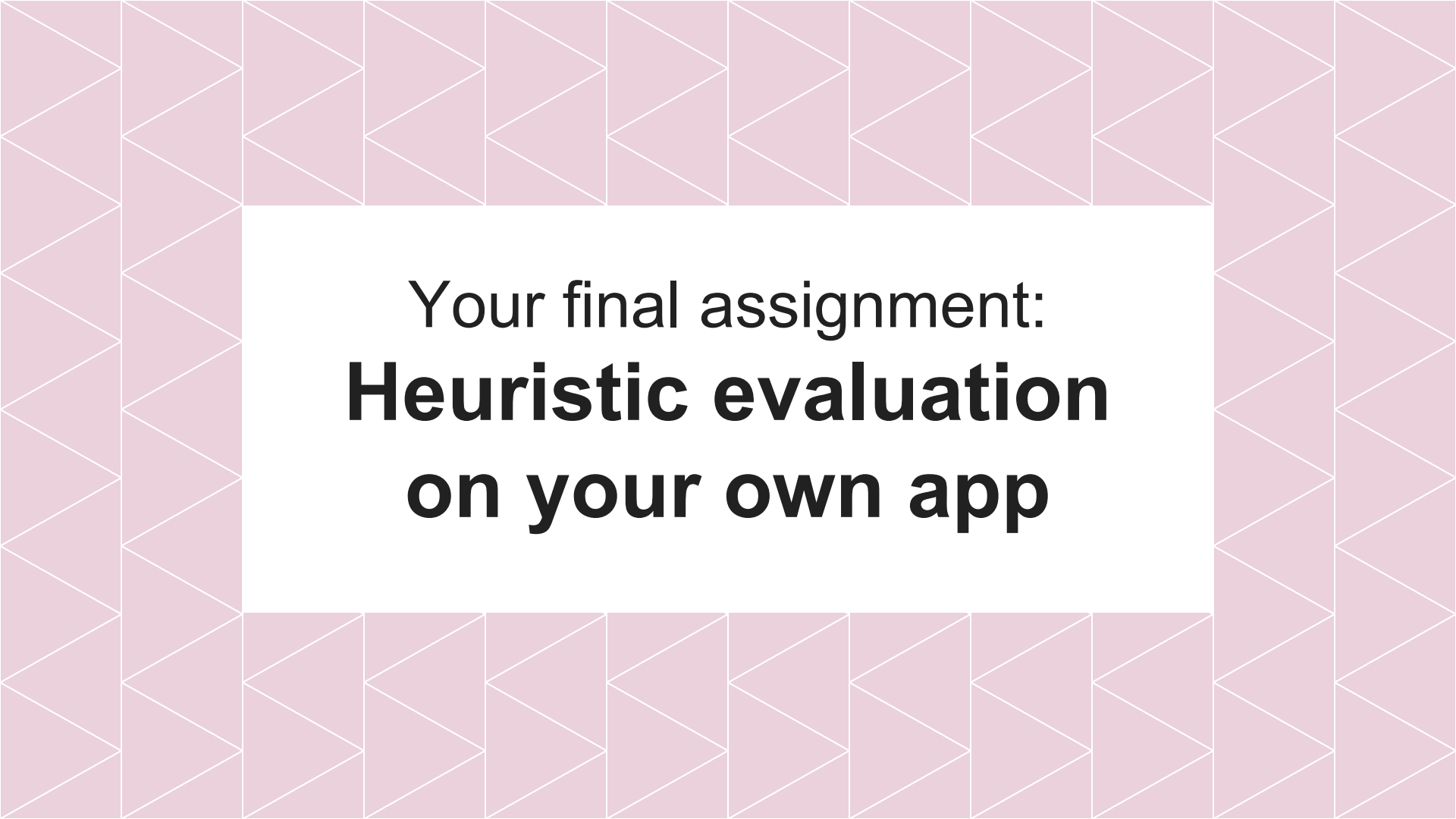
2. Even though it sounds great it's not a cure-all, and should be used appropriately.

In summary:

3. It's done by 3-5 evaluators looking at an interface according to 10 heuristics, and assigning severity ratings to violations.

In summary:

4. Then, the evaluations are used to produce a re-design!

The background of the slide is a light pink color with a repeating geometric pattern of white lines forming triangles and diamonds. A white rectangular area is centered on the slide, containing the text.

Your final assignment:
Heuristic evaluation
on your own app

Start with your own app.

You will use the app you handed in -
it doesn't matter what condition it's
in!

Your group is the evaluators.

You need 3-5 evaluators ... how convenient, that's the number in your group.

**Perform the evaluation,
collate the results.**

Present your results in an organised and cohesive way. Don't forget severity ratings! Screenshots help.

Based on the results, make suggestions for redesign.

Show us your suggestions so we understand them (screenshots!)