Toloka

Toloka

WWW 2021 hands-on tutorial

Improving Web Ranking with Human-in-the-Loop: Methodology, Scalability, Evaluation

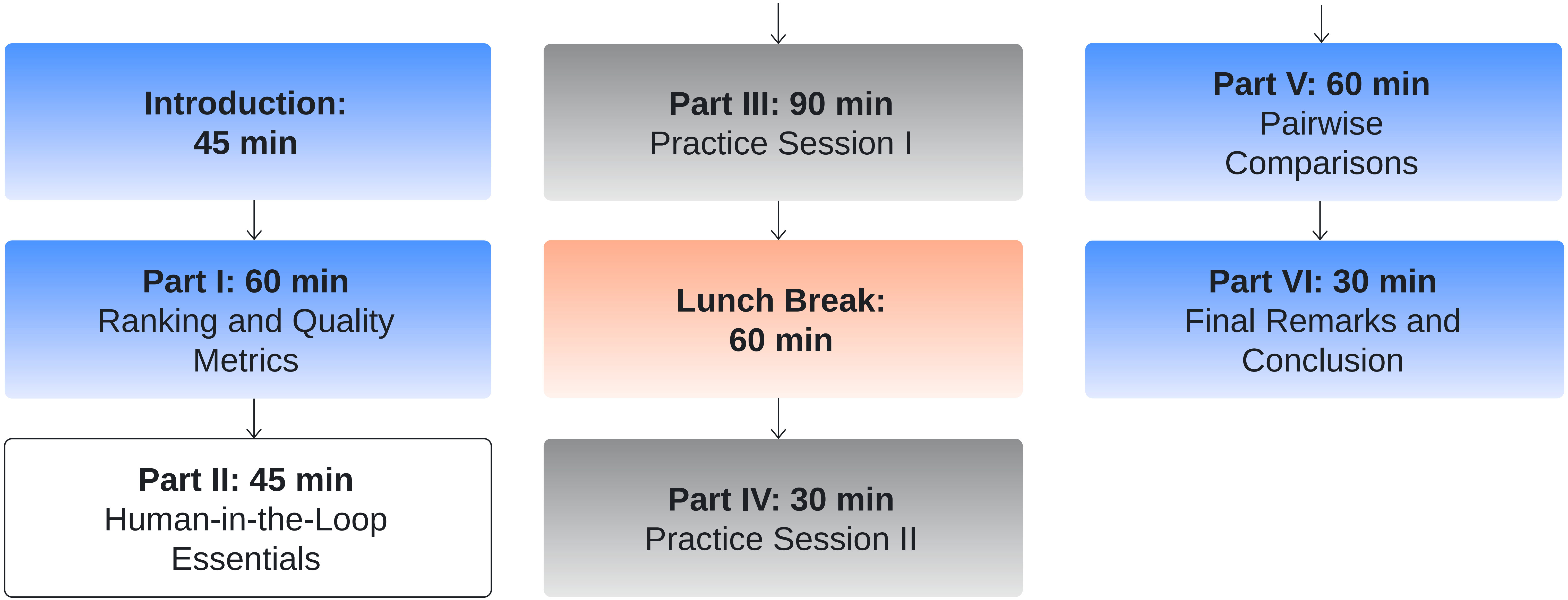
Alexey Drutsa, Dmitry Ustalov, Nikita Popov,
Daria Baidakova

Toloka

Part II
Human-in-the-Loop Essentials

Dmitry Ustalov,
Analyst/Software Developer

Tutorial Schedule



Introduction: 45 min

Part I: 60 min
Ranking and Quality Metrics

Part II: 45 min
Human-in-the-Loop Essentials

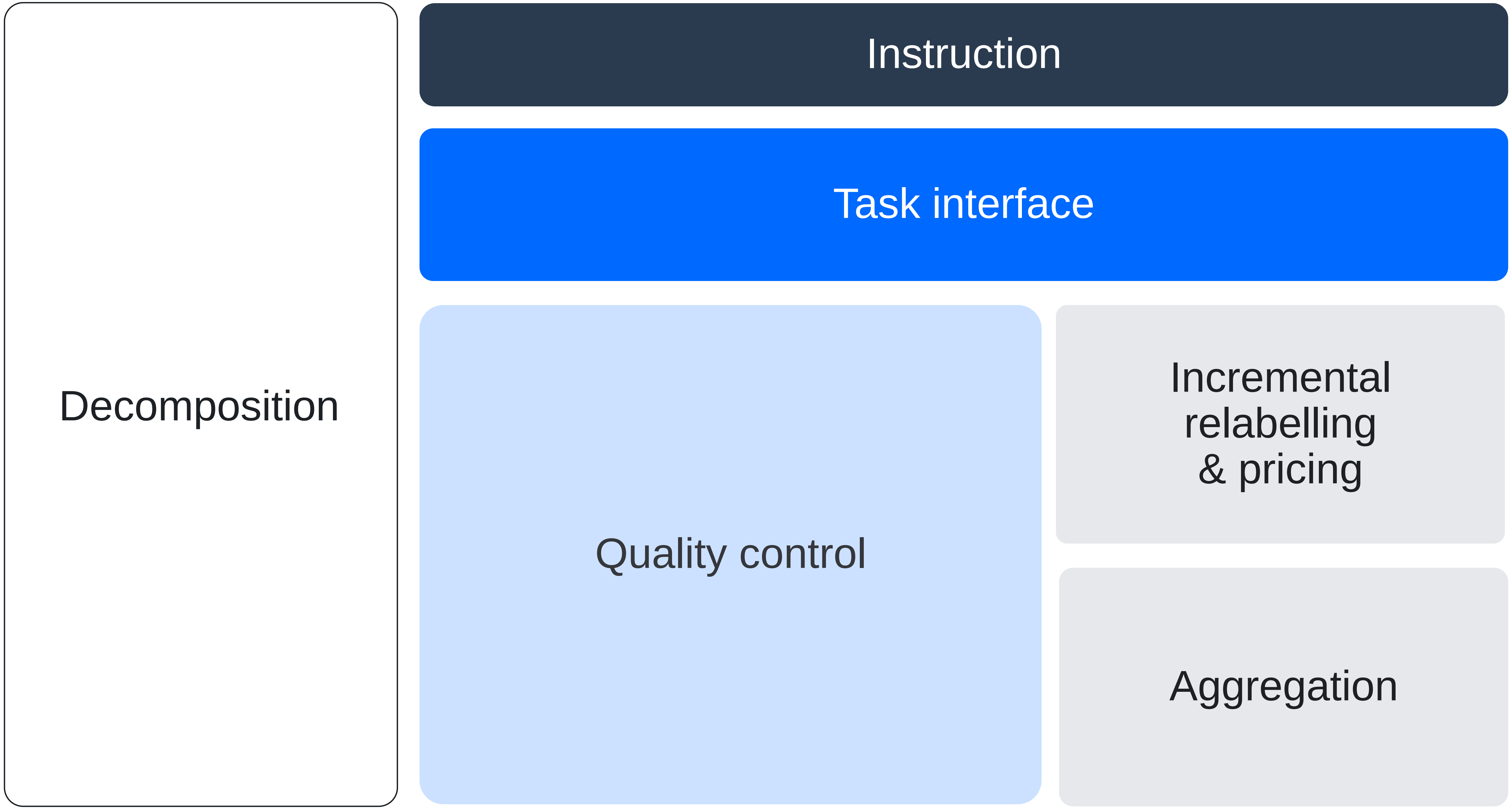
Part III: 90 min
Practice Session I

Lunch Break: 60 min

Part IV: 30 min
Practice Session II

Part V: 60 min
Pairwise Comparisons

Part VI: 30 min
Final Remarks and Conclusion



Decomposition

Instruction

Task interface

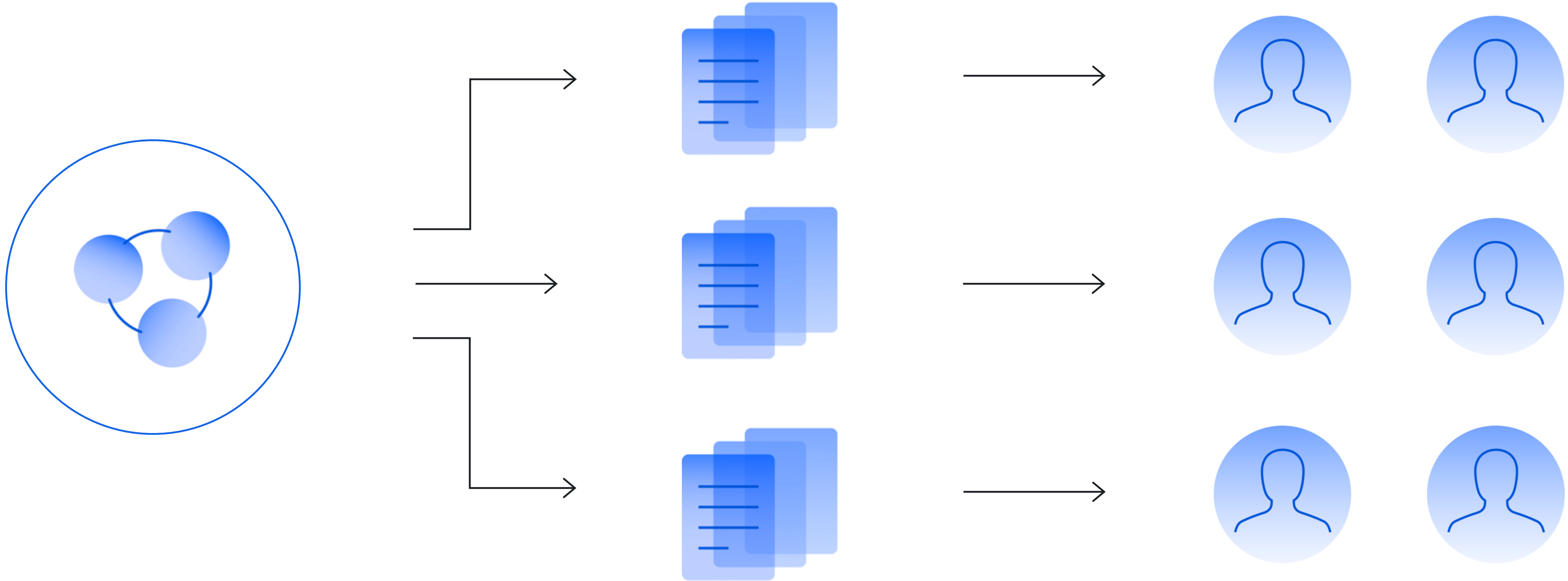
Quality control

Incremental relabelling & pricing

Aggregation

Decomposition

Decomposition



A big task

Projects with microtasks of different type

Cloud
of performers

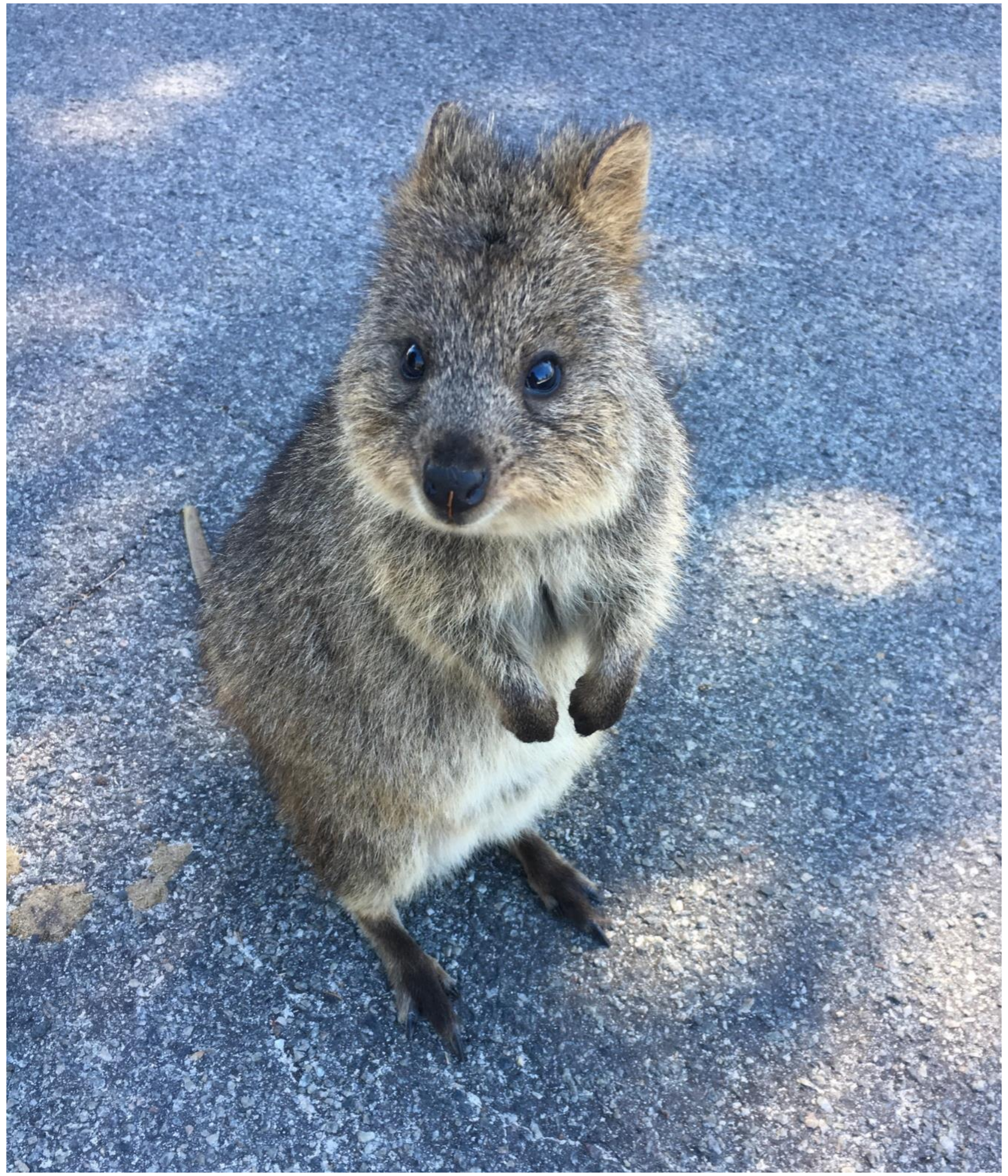
Decomposition: why?

- Performers are usually non-specialists in your specific task
- The simpler a single task is:
  - The more humans can perform your task
  - The easier its instruction
  - The better quality of performance
- A way to:
  - Distinguish tasks with different difficulty
  - Control and optimize pricing
  - Control quality by post verification

Decomposition: when?

- If
  - Your task requires an answer selected among more than 3–5 variants
  - Your task has a long instruction hard to read
- Then your task requires decomposition

Case of decomposition: a lot of questions



Bad practice: All questions in one task

What animal is on the photo?

- Cat
- Rabbit
- Bear
- Whale
- Koala
- None of the above

Is its tail visible?

- Yes
- No

Is it running?

- Yes
- No

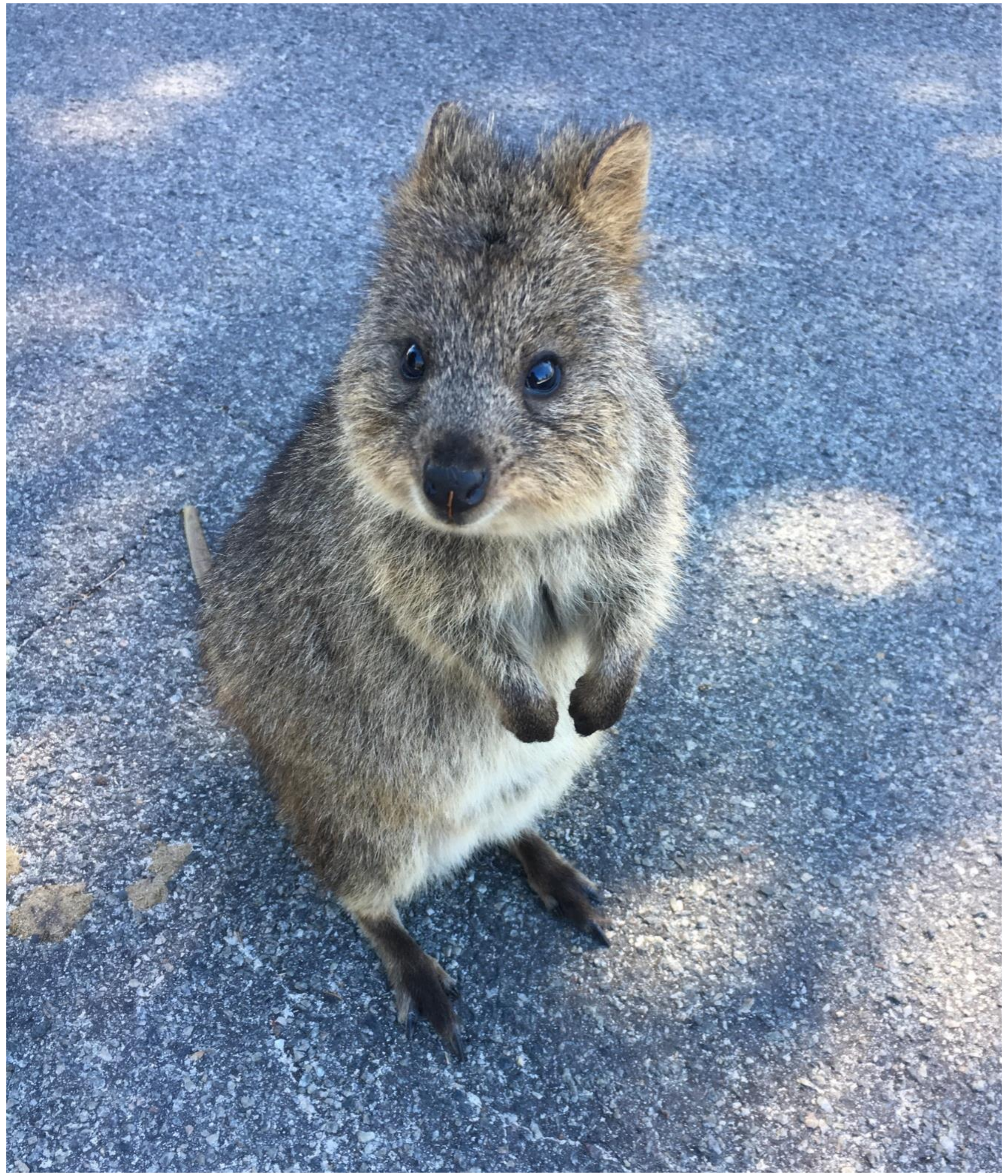
What color is it?

- White
- Black
- Brown
- Red
- Other

Where is it situated?

- On the grass
- On a tree
- On a road
- It is flying
- None of the above

Case of decomposition: a lot of questions



Good practice: Each question in a separate task

What animal is on the photo?

- Cat
- Rabbit
- Bear
- Whale
- Koala
- None of the above

Is its tail visible?

- Yes
- No

Is it running?

- Yes
- No

What color is it?

- White
- Black
- Brown
- Red
- Other

Where is it situated?

- On the grass
- On a tree
- On a road
- It is flying
- None of the above

Case of decomposition: need to verify answers

The task: Highlight all koalas on the photo

Problem: **highlighting can be done in different ways**

Hence, it is difficult to make:

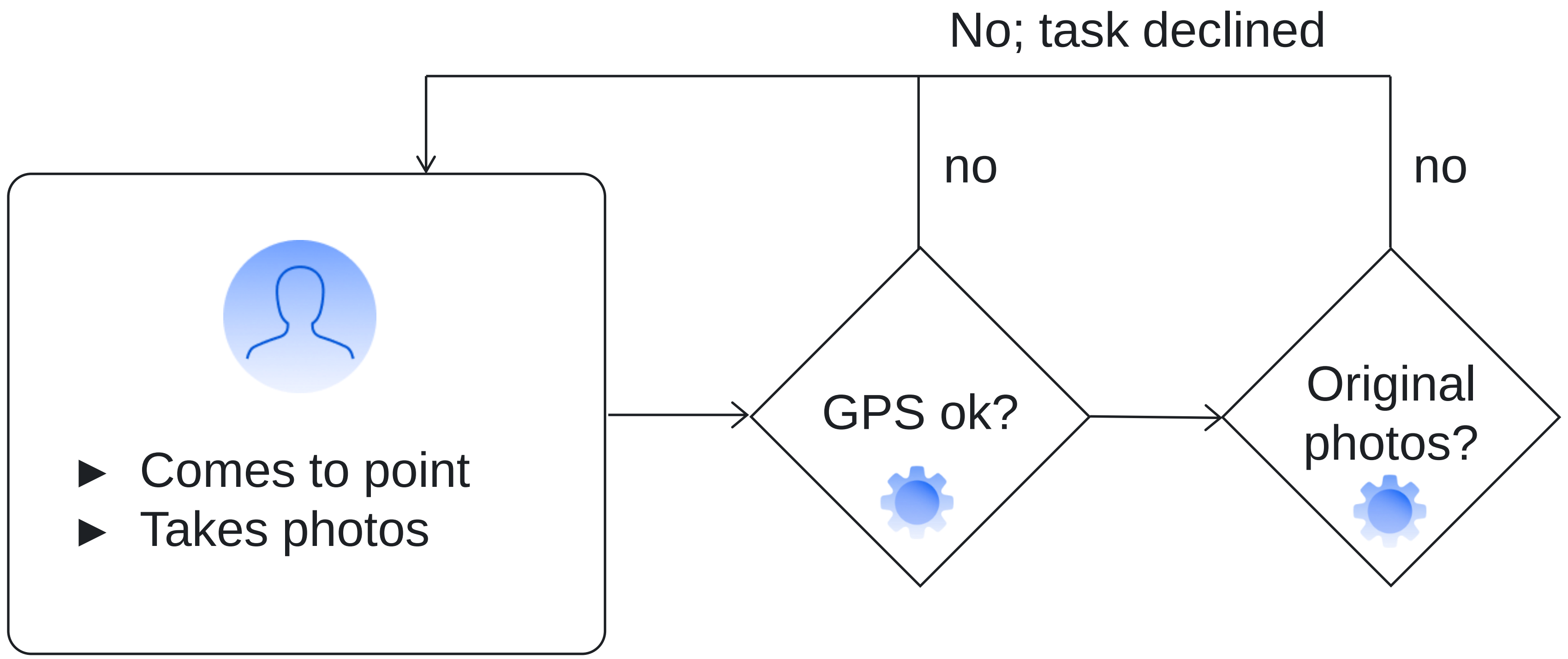
- Comparison with control answers
- Aggregation of answers from different performers

A good solution
A task for another performer:
Is the highlighting of all koalas made correctly?

Real example: decomposition for an offline data collection task

- Comes to point
- Takes photos

Final result:
verified business



No; task declined

no

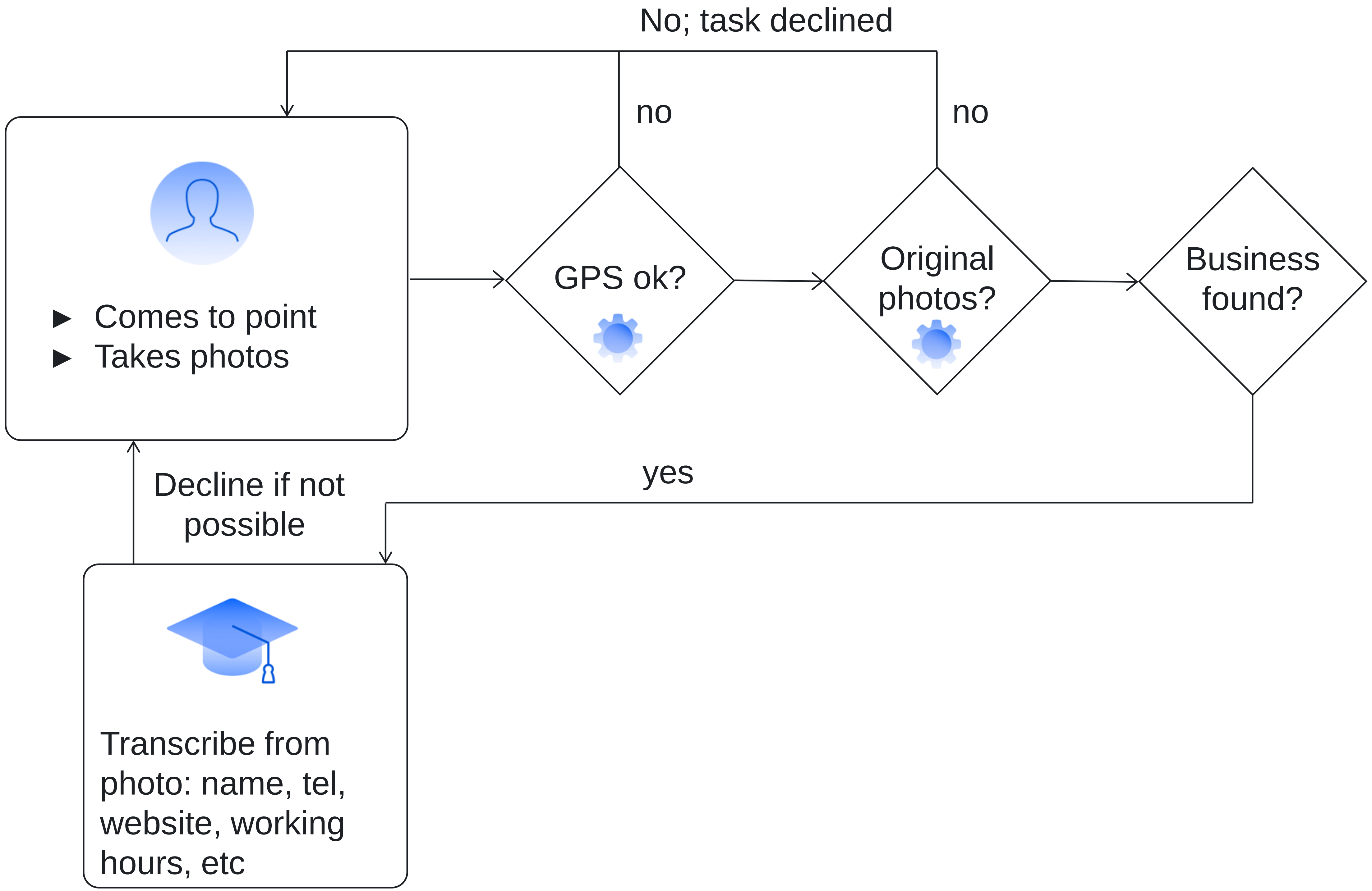
no

- Comes to point
- Takes photos

GPS ok?

Original photos?

Final result:
verified business



No; task declined

no

no

- Comes to point
- Takes photos

GPS ok?

Original photos?

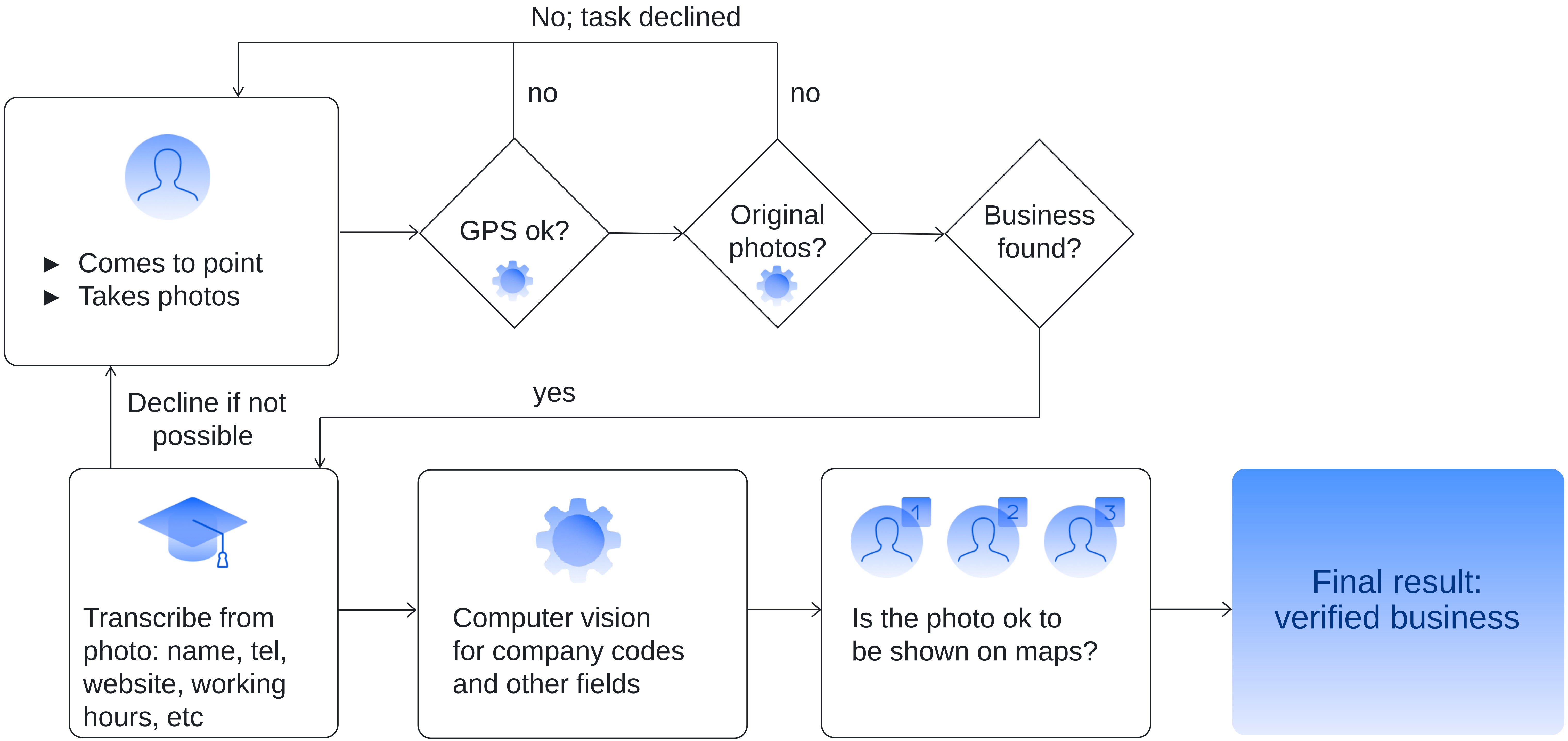
Business found?

yes

Decline if not possible

Transcribe from photo: name, tel, website, working hours, etc

Final result:
verified business



No; task declined

no

no

- Comes to point
- Takes photos

GPS ok?

Original photos?

Business found?

yes

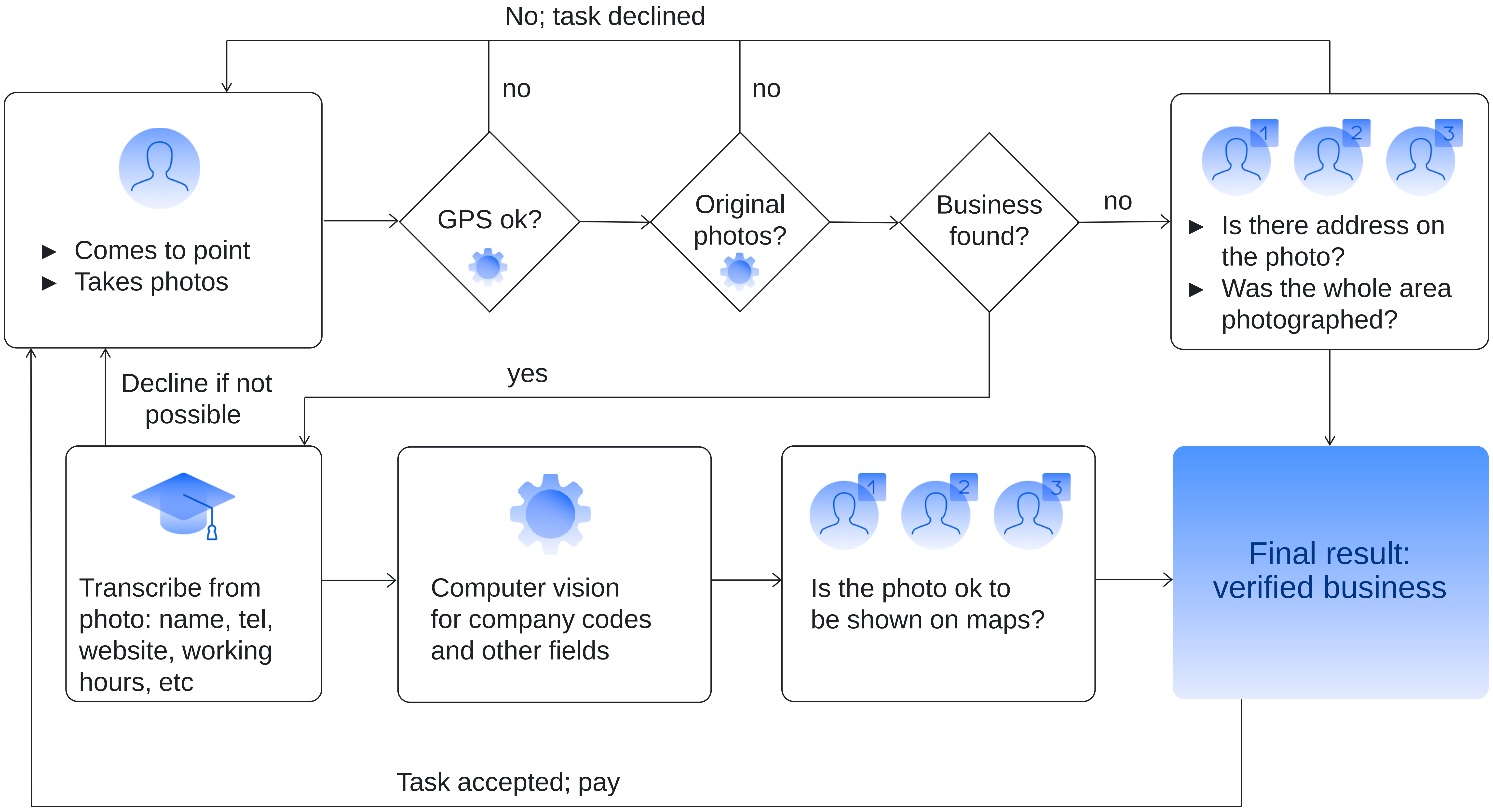
Decline if not possible

Transcribe from photo: name, tel, website, working hours, etc

Computer vision for company codes and other fields

Is the photo ok to be shown on maps?

Final result: verified business


No; task declined

no

no

GPS ok?

Original photos?

Business found?

no

- Comes to point
- Takes photos

- Is there address on the photo?
- Was the whole area photographed?

Decline if not possible

yes

Transcribe from photo: name, tel, website, working hours, etc

Computer vision for company codes and other fields

Is the photo ok to be shown on maps?

Final result: verified business

Task accepted; pay

Instruction

Instruction: a typical structure

- Goal of the task to be done
- Interface description
- Algorithm of required actions
- Examples of good and bad answers
- Algorithm and examples for rare cases
- Reference materials

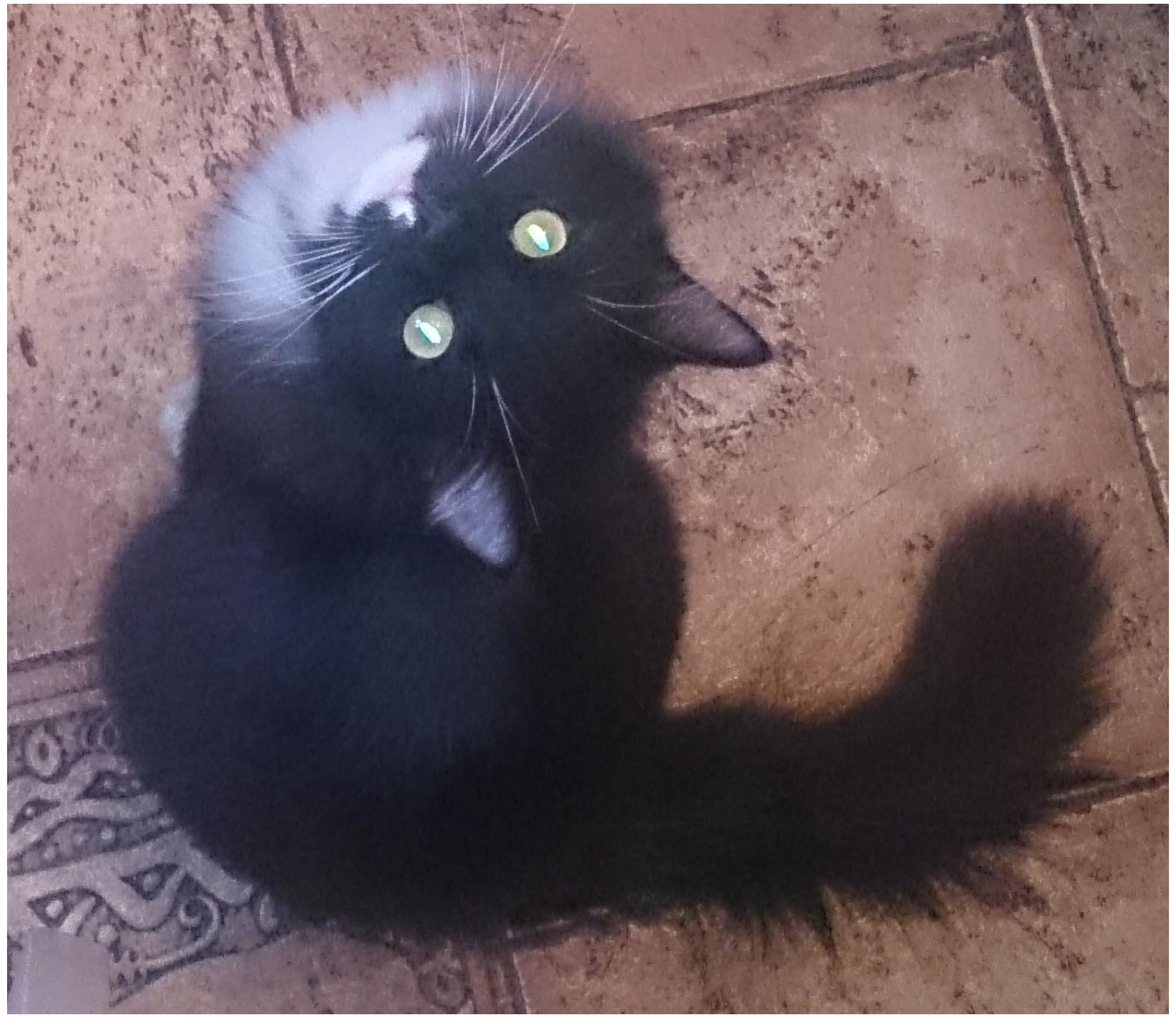
↑ Most pitfalls are here

Instruction ambiguity for a rare case: example

Is this cat white?

Yes

No



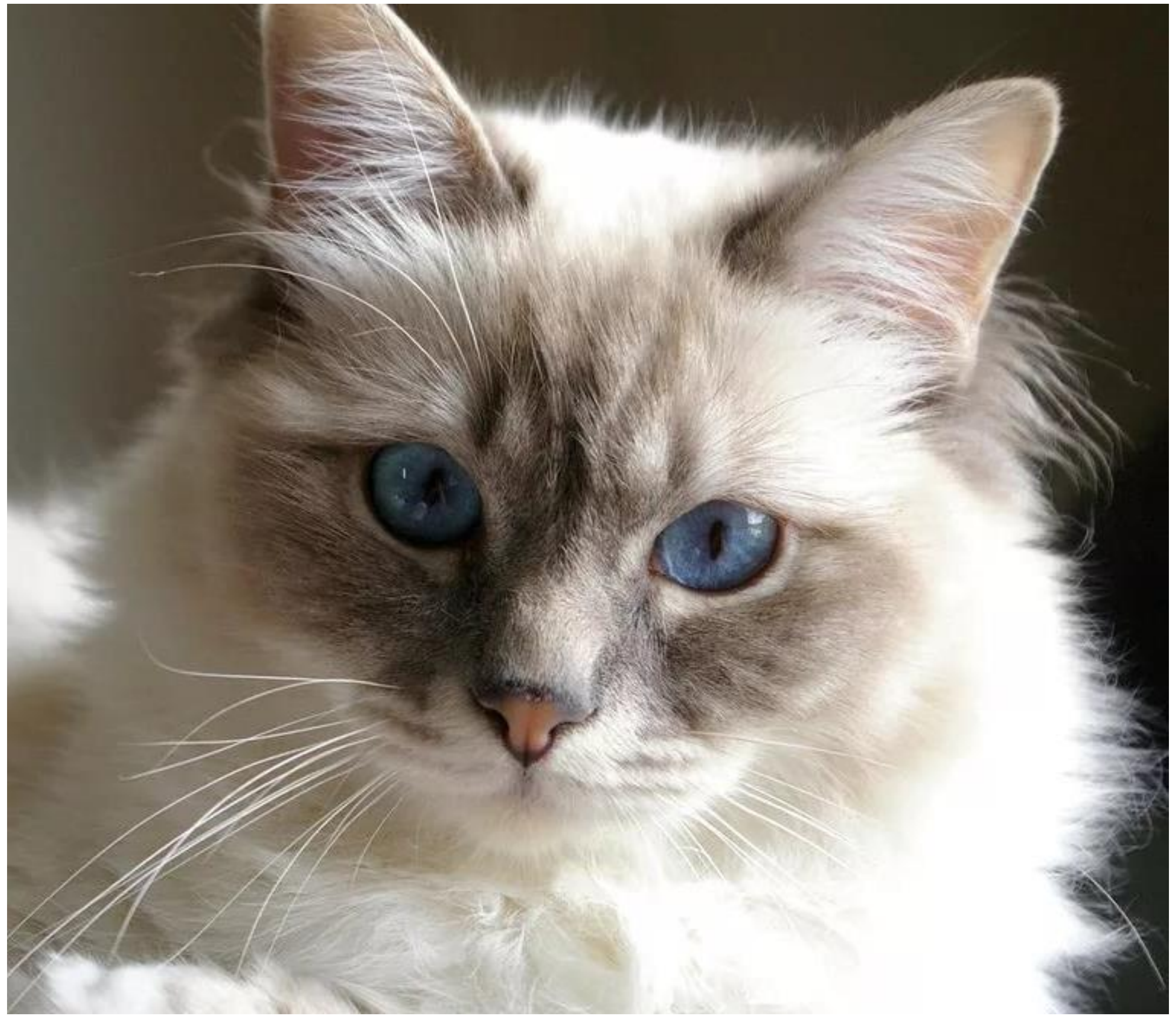
OK: the answer and the task seem clear

Instruction ambiguity for a rare case: example

Is this cat white?

Yes

No



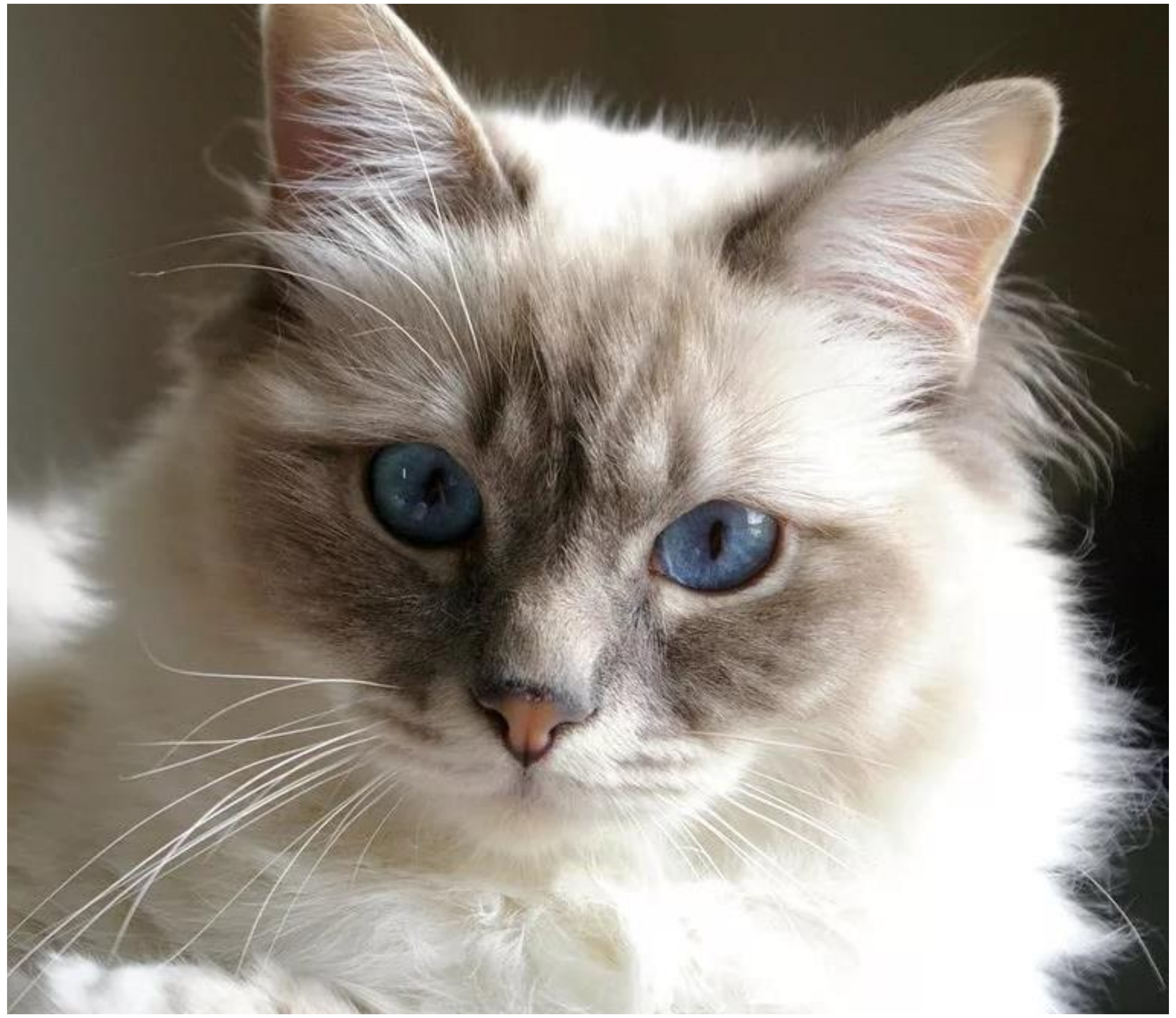
What is the correct answer?

Instruction ambiguity for a rare case: example

Is this cat white?

Yes

No



How to fix

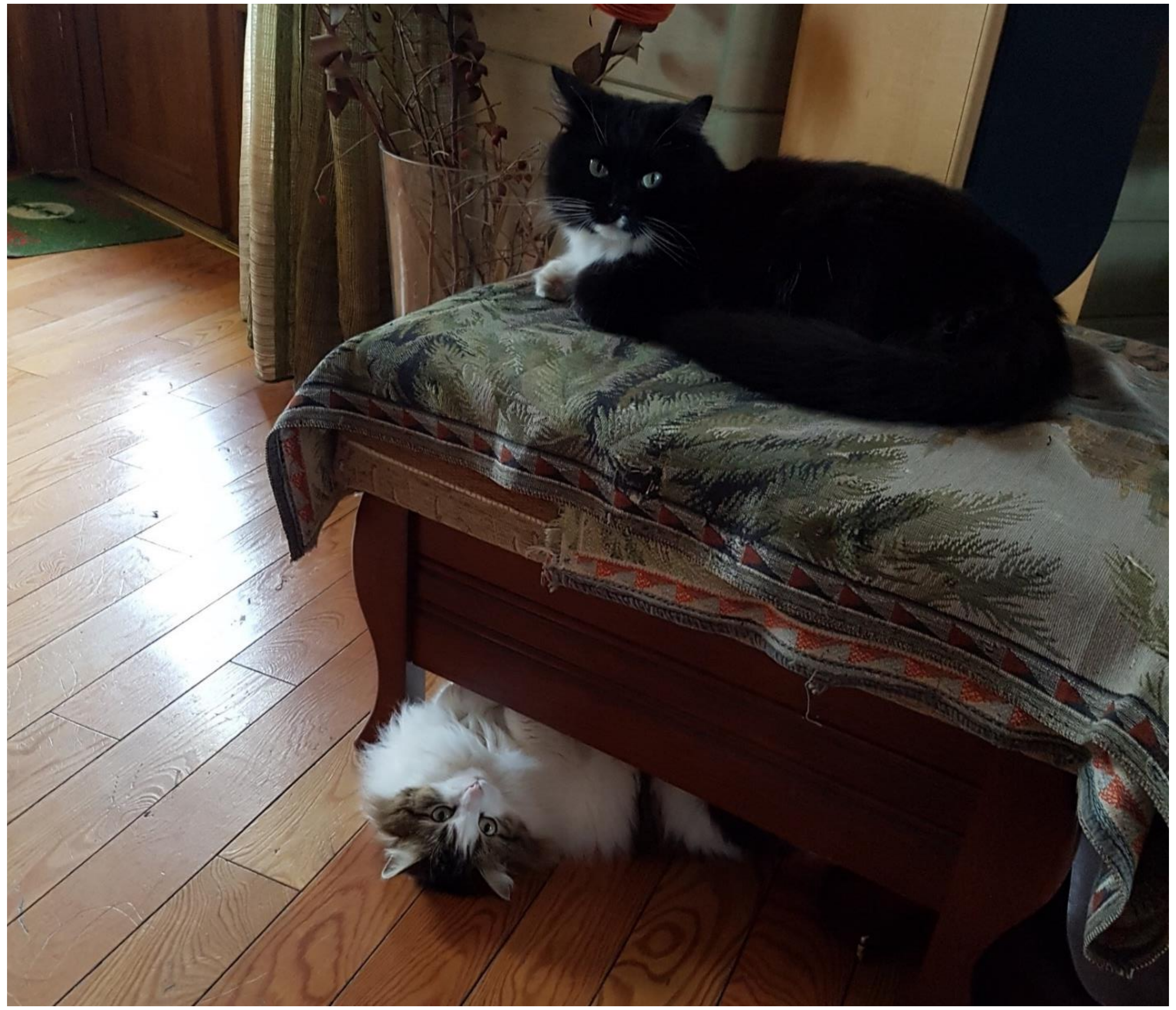
– In the instruction: clarify what you mean under “a white cat”

Instruction ambiguity for a rare case: example

Is this cat white?

Yes

No



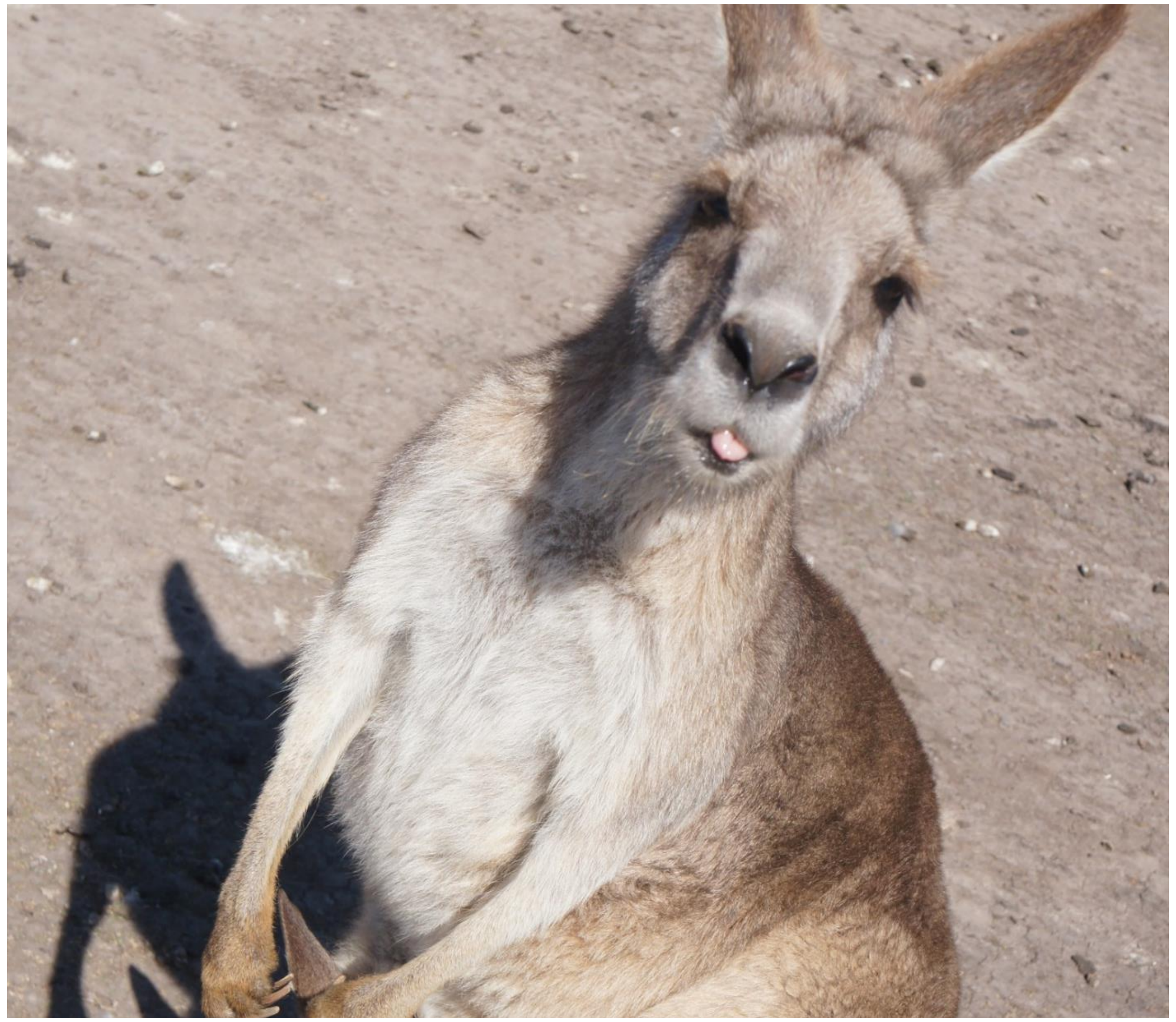
Rare case: many cats

Instruction ambiguity for a rare case: example

Is this cat white?

Yes

No



Rare case: not a cat

Instruction ambiguity for a rare case: example

Is this cat white?

Yes

No

404: Cannot download the image

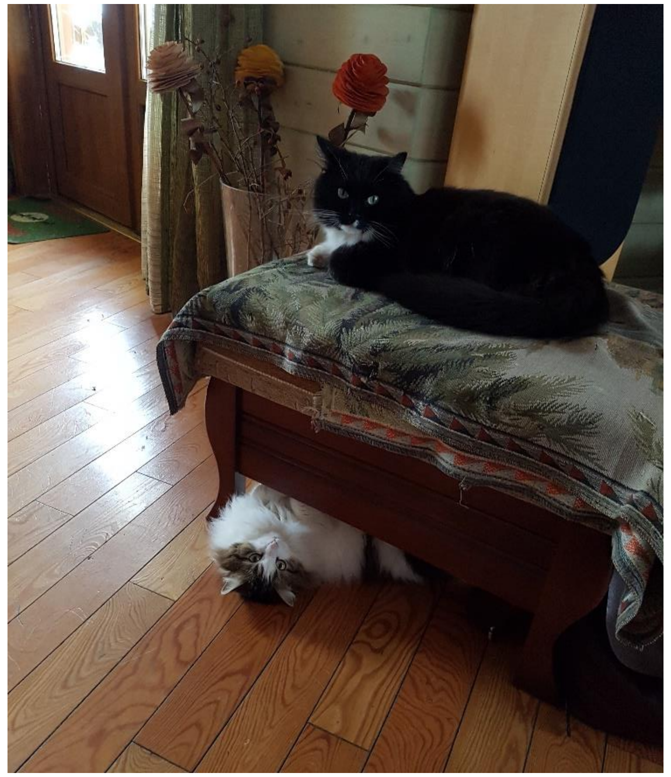
Rare case: image has not been shown

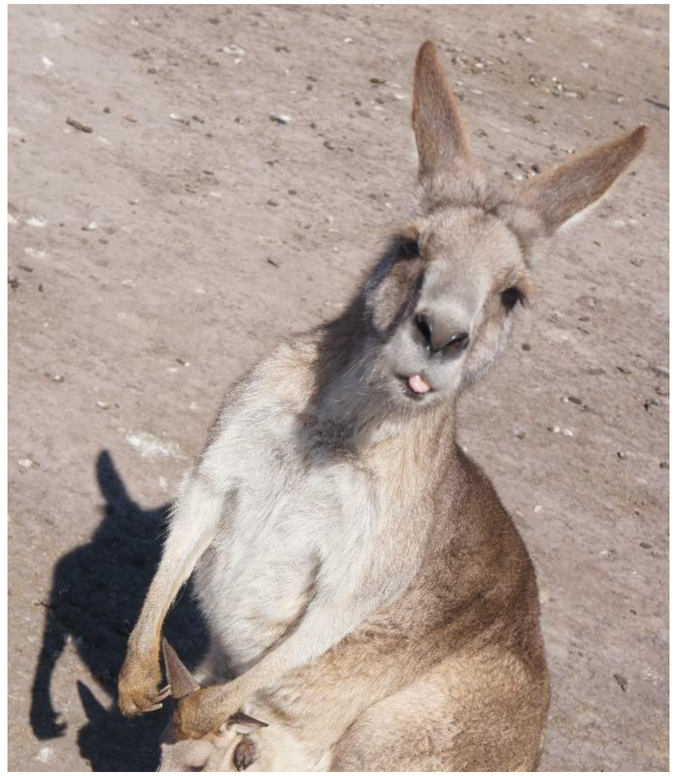
Instruction ambiguity for a rare case: example

Is this cat white?

Yes

No





404: Cannot download the image

It is difficult to predict situations of any kind, but you can:

- In the instruction: clarify what should be done in a non-standard situation
- In the interface: add a text field to allow a performer to report the case

Task interface

Task interface: summary on best practices

For faster performance:

- Hot key combinations for checkboxes / radio buttons / buttons
- Reduce navigation to third-party sites
- Effective composition of a task template
- Optimal position of tasks on a page

For better quality and less errors:

- Dynamic interface (show/hide input controls depending on user actions)
- Adaptive interface (good view for any device and screen resolution)
- Always test your interface (template testing)
- Dynamic validation of input data (e.g. a text is less than 3 words)

Quality control

Quality control

“Before” task performance

- ► Selection of performers
- ► Well-designed instruction

“Within” task performance

- ► Golden set (aka honey pots)
- ► Well-designed interface
- ► Motivation (e.g. performance-based pricing)
- ► Tricks to remove bots and cheaters (e.g. quick answers)

“After” task performance

- ► Post verification (acceptance)
- ► Consensus between performers and result aggregation

Selection of performers

- ► Filter by static properties (e.g. education, languages, citizenship, etc.)
- ► Filter by computed properties (e.g. browser, region by phone/IP, etc.)
- ► Filter by skills
  - To select proper specialization
  - To control quality level on your tasks
  - To get performers with best quality on past projects
- ► Educate to perform your tasks
  - Use training tasks to show how to perform tasks
  - Use exam tasks to evaluate education level

Golden set (aka honey pots)

Tasks with known correct answer shown to performers to evaluate their quality

- ▶ Distribution of answers in golden set = distribution in whole set of tasks
- ▶ But should contain rare answer variants with higher frequency
- ▶ Refresh your set of honey pots regularly to avoid bots and cheating
- ▶ Automatic golden set generation via performers:
  - Tasks with answers of high confidence
  - (e.g. aggregation of answers from a large number of performers)

↑

Best practices

Motivation

- Bonuses for a good quality within a period
- Gamification (e.g. achievements, leader boards, etc.)
- Price depends on quality

Tricks to remove bots and cheaters

- Control fast responses
- Check whether a link has been visited
- Check whether a video has been played
- etc.

Post verification (acceptance)

A performer gets money only if his answer is accepted

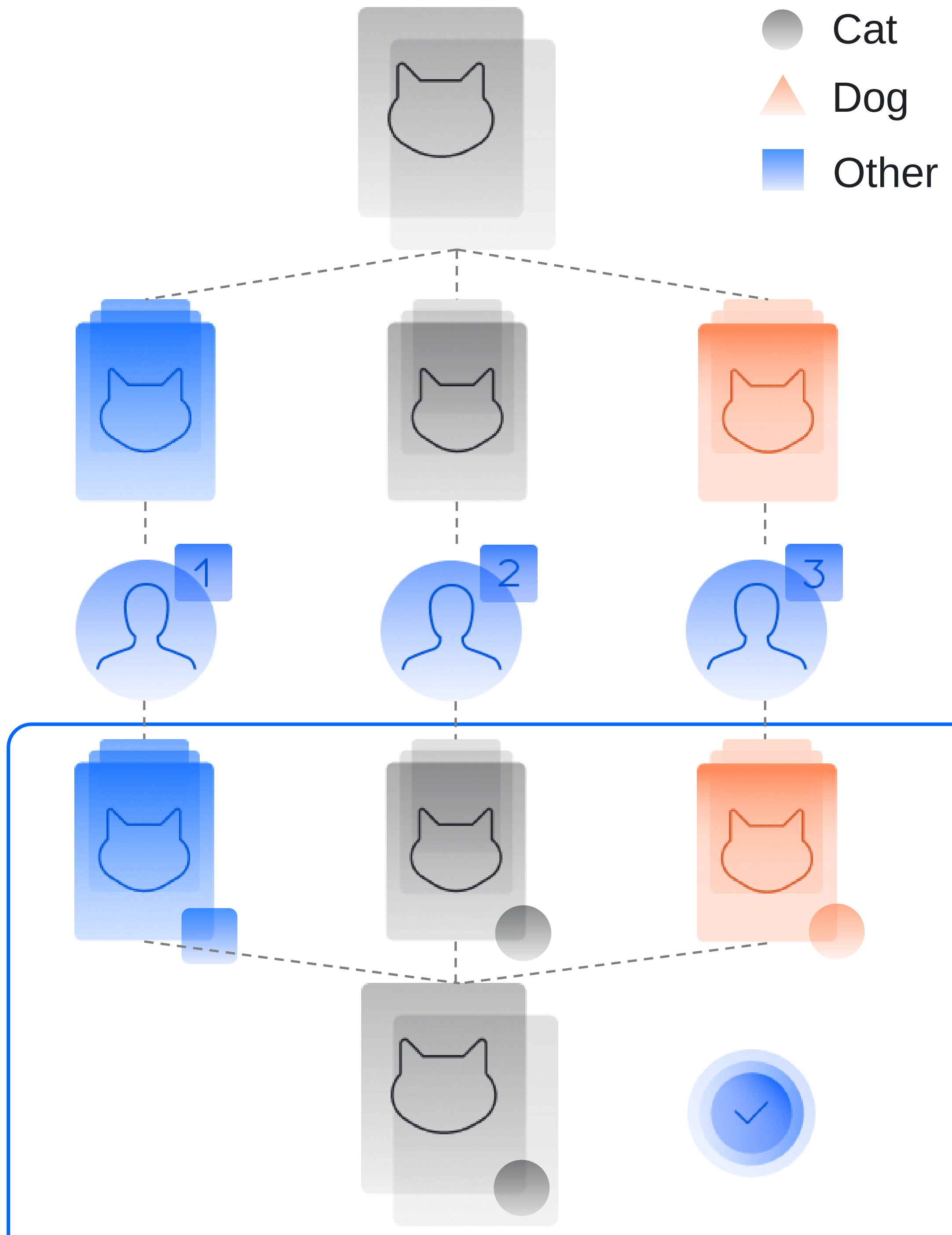
- Is used when a task is sophisticated
  (neither golden set nor consensus models work)
- Can be performed on your own, but

You can use other crowd performers via a task of different type

- Thus, you deal with hierarchy of projects
  (you apply decomposition)

Aggregation

Aggregation



Upload multiple copies of each object to label

Performers assign noisy labels to objects

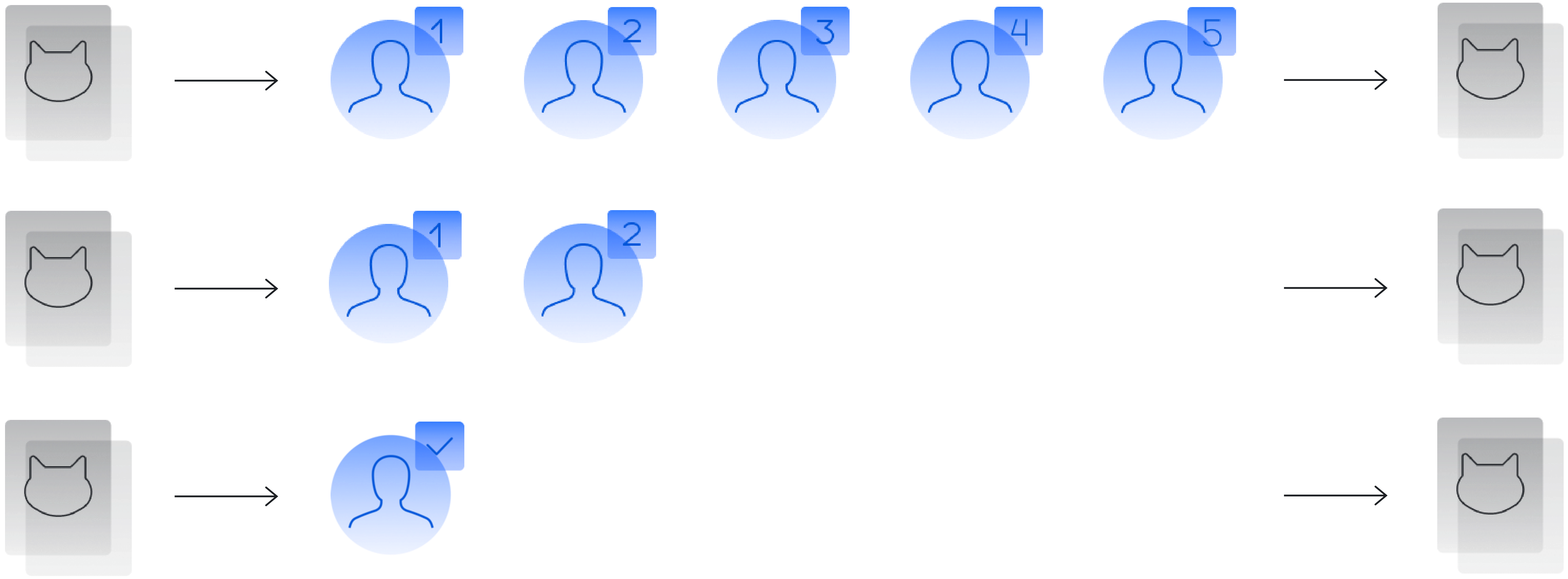
Aggregate multiple labels into a more reliable one

Will be discussed in Part V

Incremental Relabeling & Pricing

Incremental relabeling

Obtain aggregated labels of a desired quality level using a fewer number of noisy labels



Several unknown performers

A few performers with known good quality

One expert with high quality

Pricing depends on

Task design

- Payment is made per a batch of microtasks (aka a task suite)
- Time required to perform a task: control hourly wage

Market economy aspects

- The lower supply of performers is (e.g. due to specific skills), the higher price
- How quickly do you need the accomplished tasks (latency)?

Result quality

- Incentivize better performance with a quality-dependent price

IF

Good decomposition

THEN

Simple instruction

Easy to use task interface

Performers do tasks with better quality

Easy to control quality

Standard aggregation models work well

Easy to control and optimize pricing

Thank you!
Questions?

Dmitry Ustalov

Analyst/Software Developer

✉ dustalov@yandex-team.ru

🌐 https://research.yandex.com/tutorials/crowd/www-2021