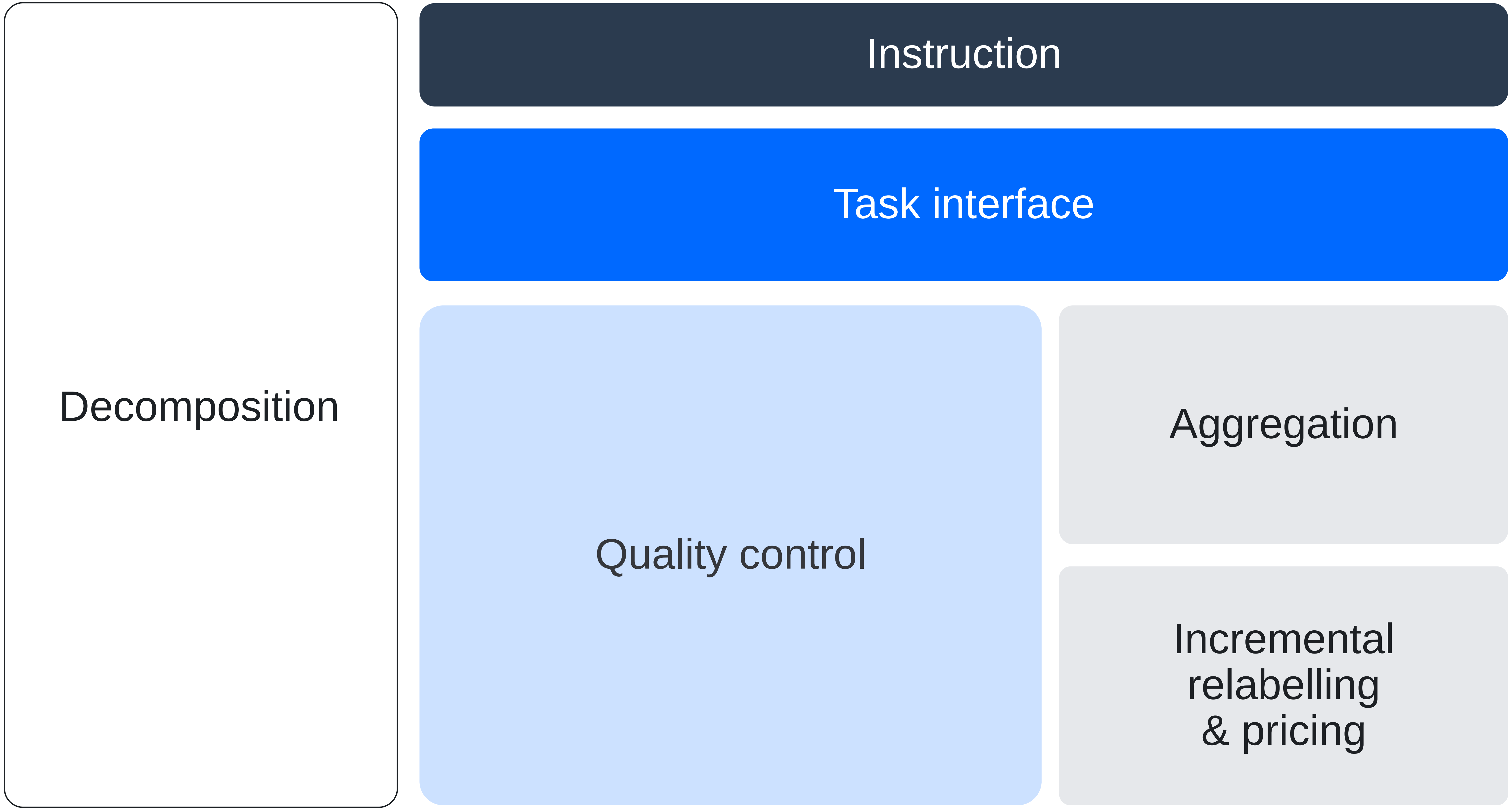
Part I

Main components of data collection via crowdsourcing

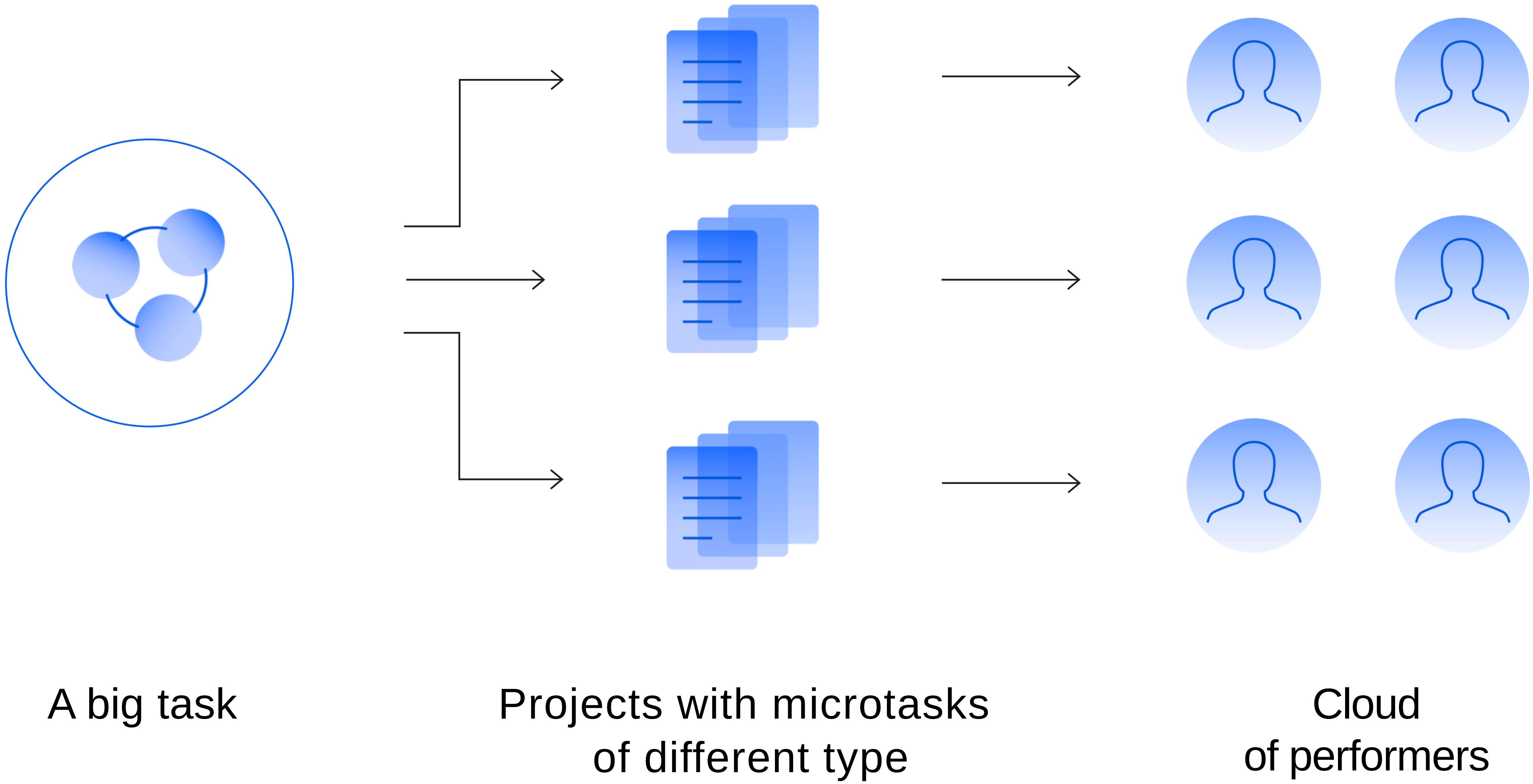
Alexey Drutsa,
Head of Efficiency and Growth Division

Main components for effective crowdsourcing


Instruction
Task interface
Decomposition
Quality control
Aggregation
Incremental relabelling & pricing

Decomposition

Decomposition



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
- Dog
- 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
- Dog
- 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

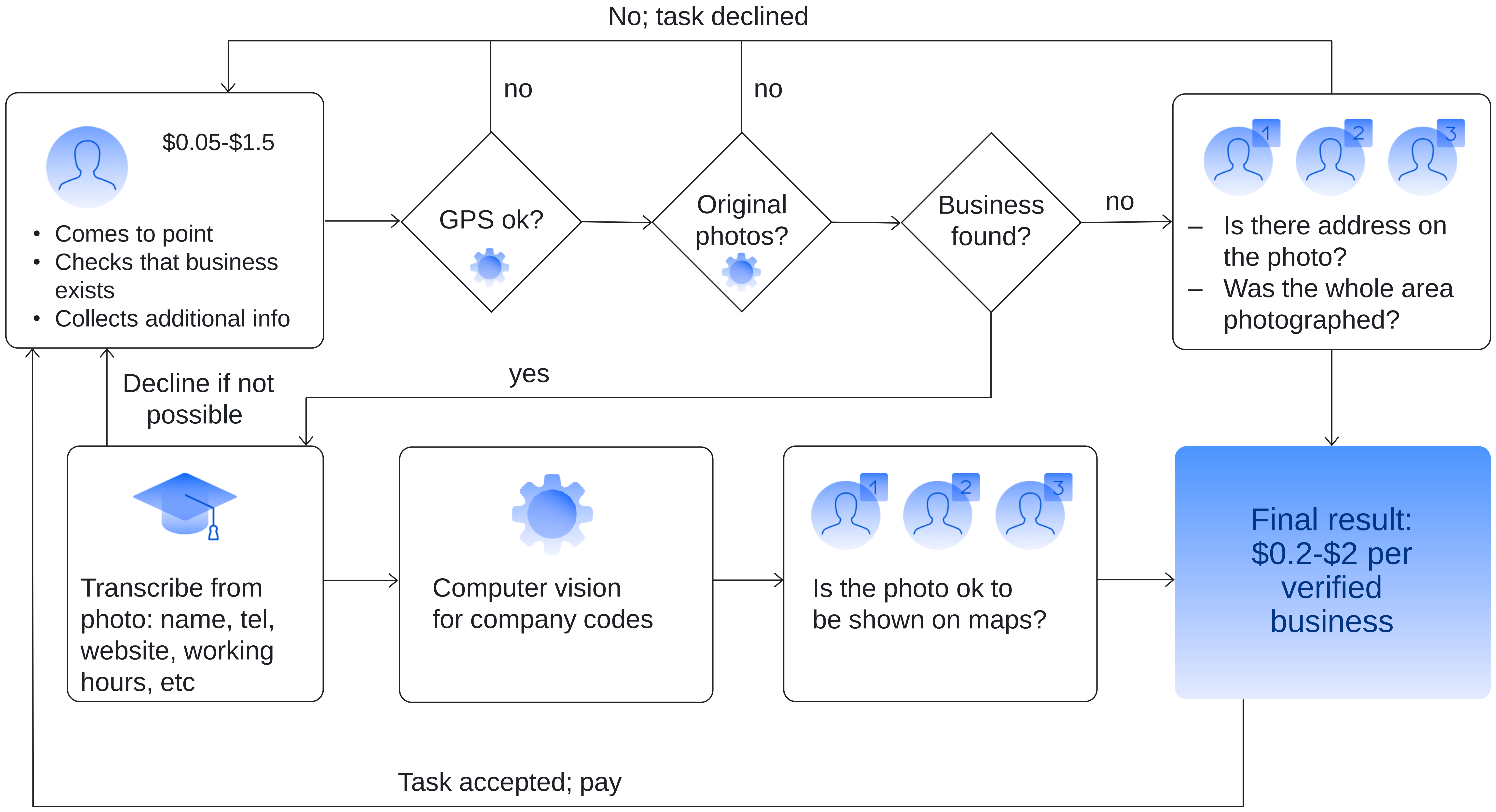
Problem: highlighting can be done in different ways

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 a field survey


No; task declined
no
no
$0.05-$1.5
Comes to point
Checks that business exists
Collects additional info
GPS ok?
Original photos?
Business found?
no
Is there address on the photo?
Was the whole area photographed?
yes
Decline if not possible
Transcribe from photo: name, tel, website, working hours, etc
Computer vision for company codes
Is the photo ok to be shown on maps?
Final result: $0.2-$2 per 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”
- In the interface: add a button “do not know” to find this rare case

Instruction ambiguity for a rare case: example

Is this cat white?

Yes

No

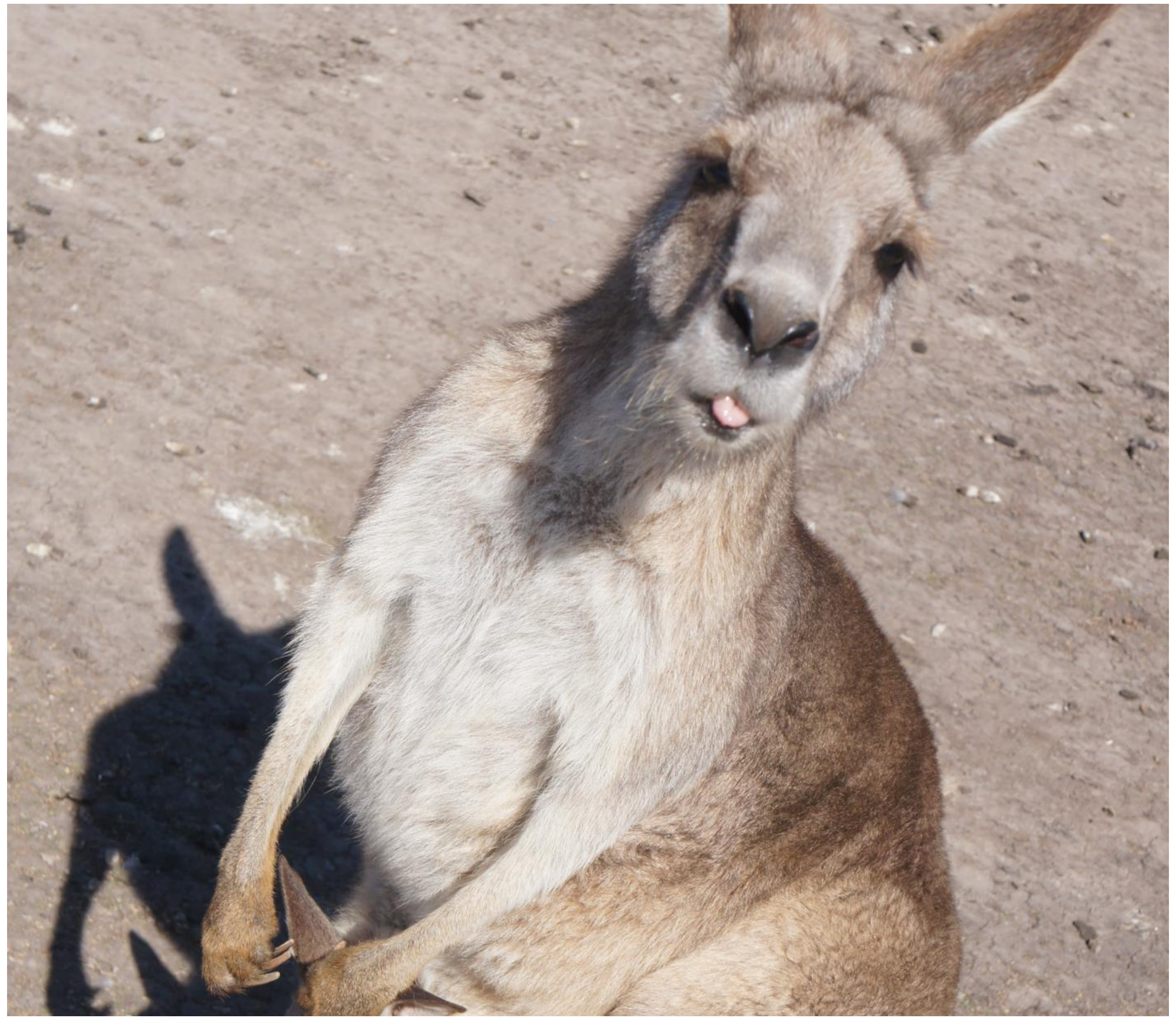
Rare case: many cats

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

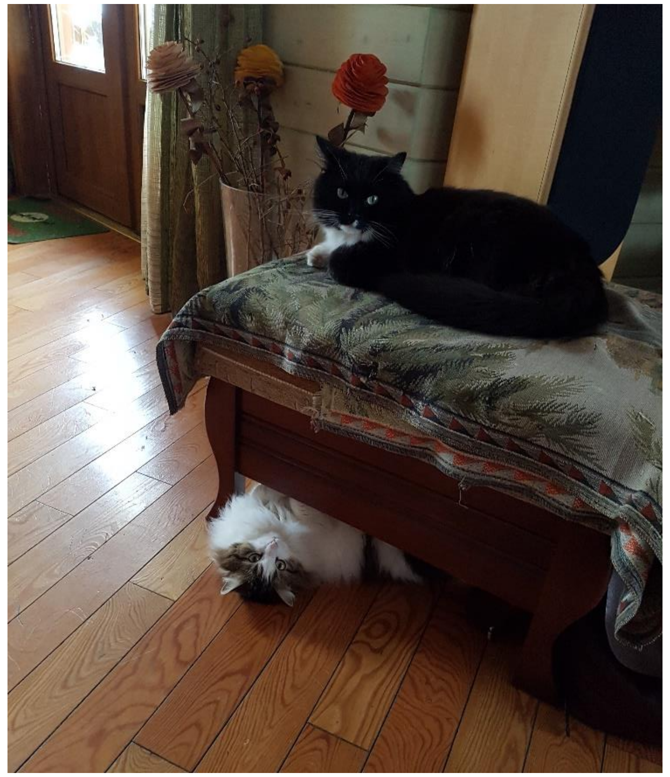
Rare case: image has not been shown

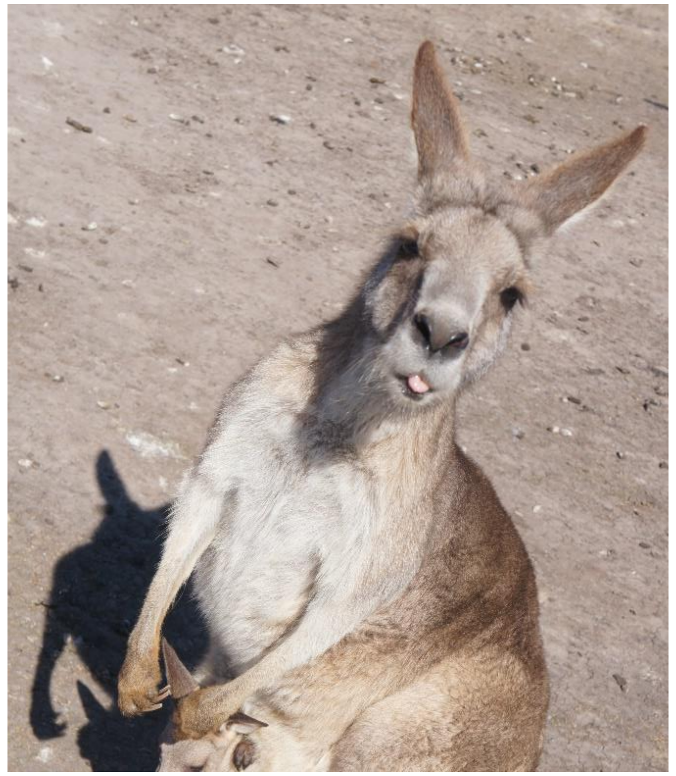
Instruction ambiguity for a rare case: example

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 depending on quality

Will be discussed in Part V

Tricks to remove bots and cheaters

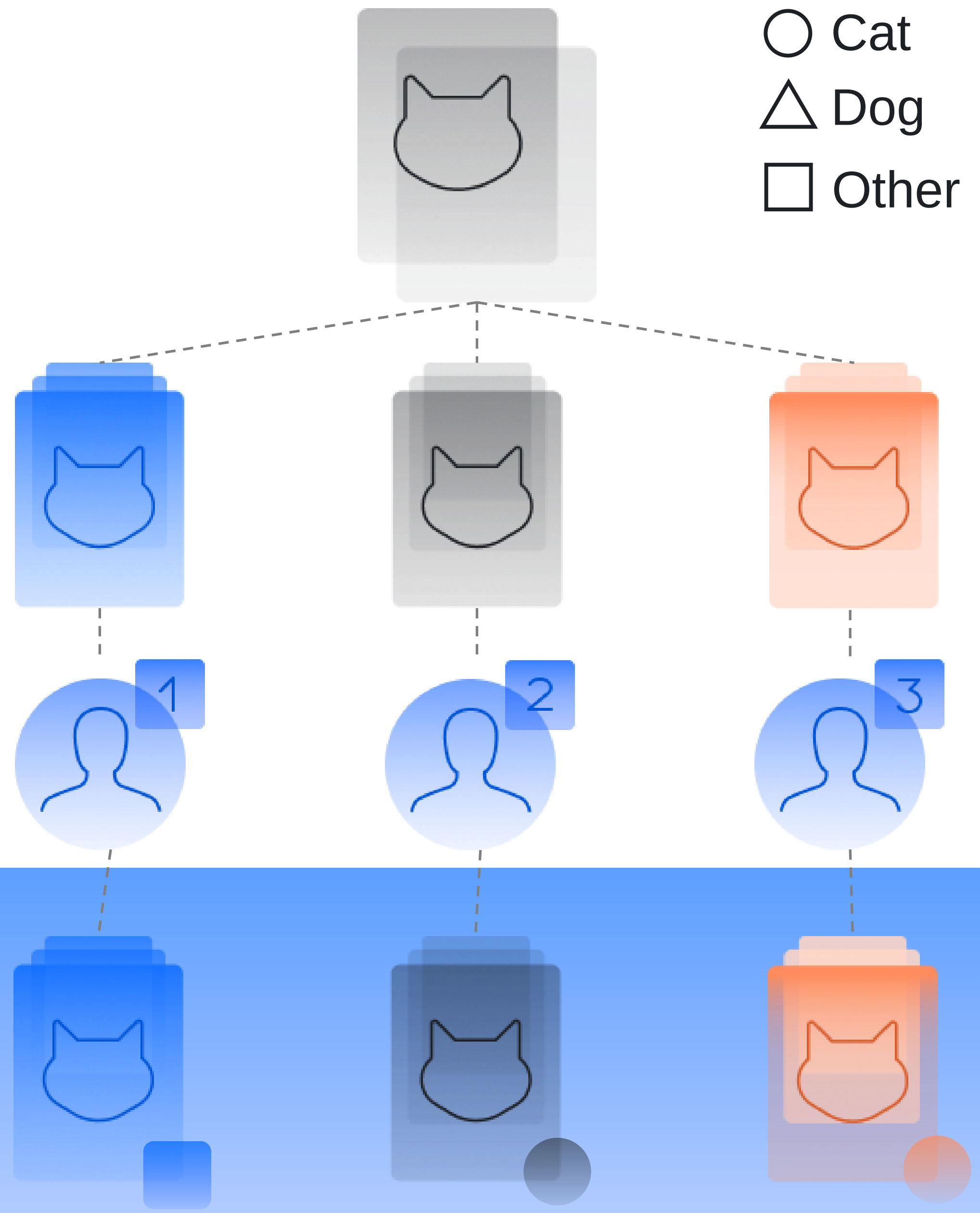
- Control fast responses
- Check whether a link has been visited
- Check whether a video has been played
- etc

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)

Consensus between performers



Upload multiple copies of each object to label

Workers assign noisy labels to objects

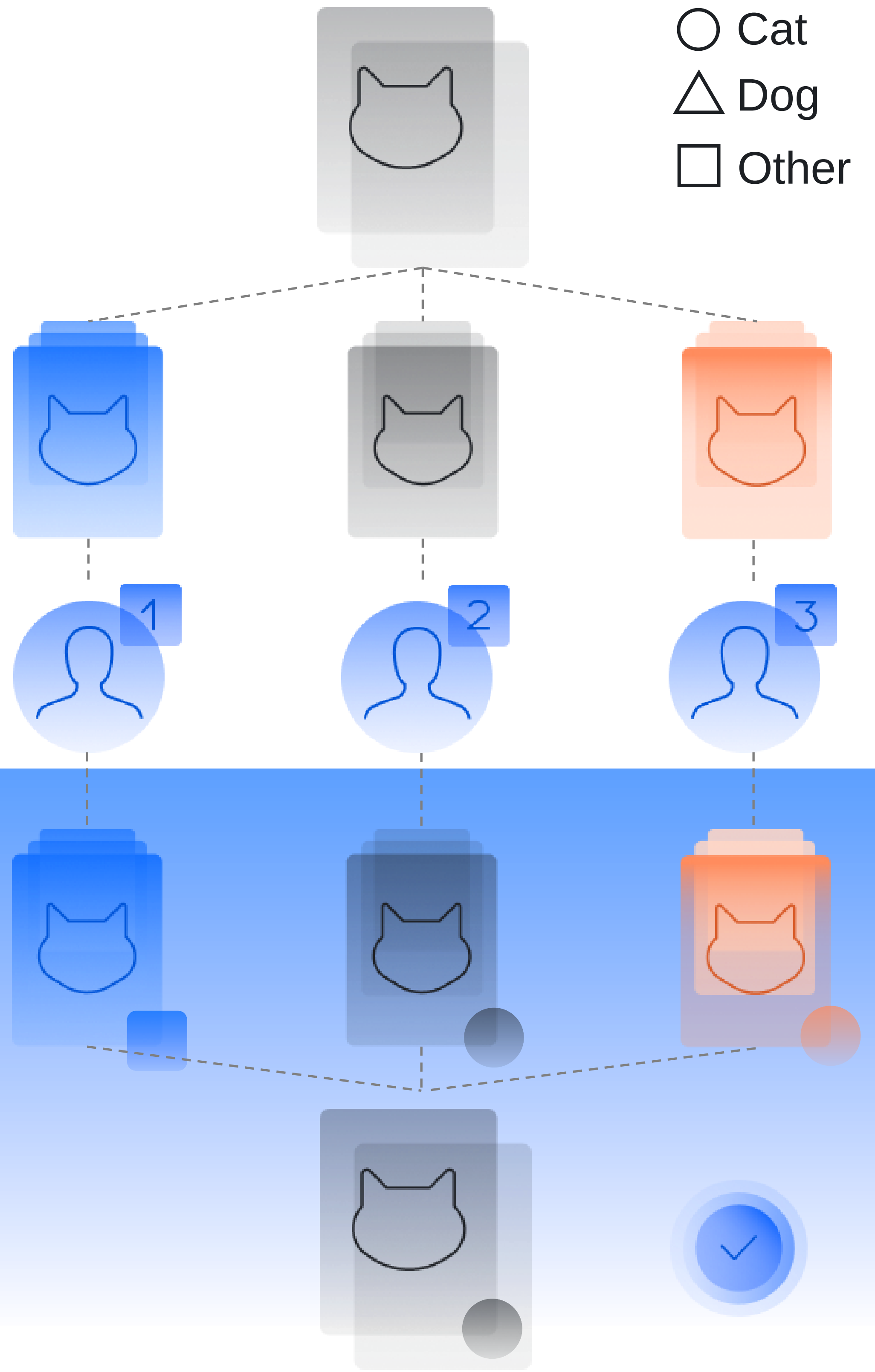
Derive quality from consensus

Works well only if most workers have good quality

Will be discussed in Part V

Aggregation

Aggregation



Upload multiple copies of each object to label

Workers assign noisy labels to objects

Aggregate multiple labels into a more reliable one

The simplest way:

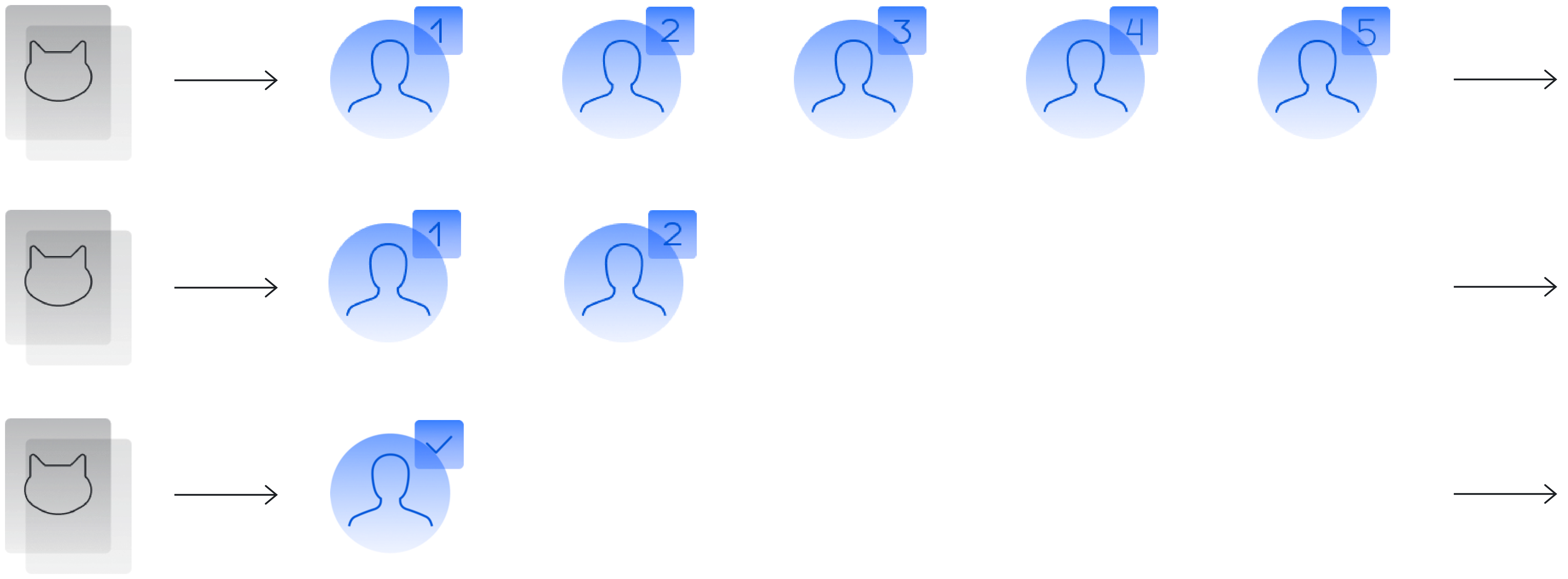
- Assign the most popular answer (Majority Vote)
- There are more sophisticated methods

Will be discussed in Part V

Incremental relabelling & Pricing

Incremental relabelling

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

Will be discussed in Part V

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

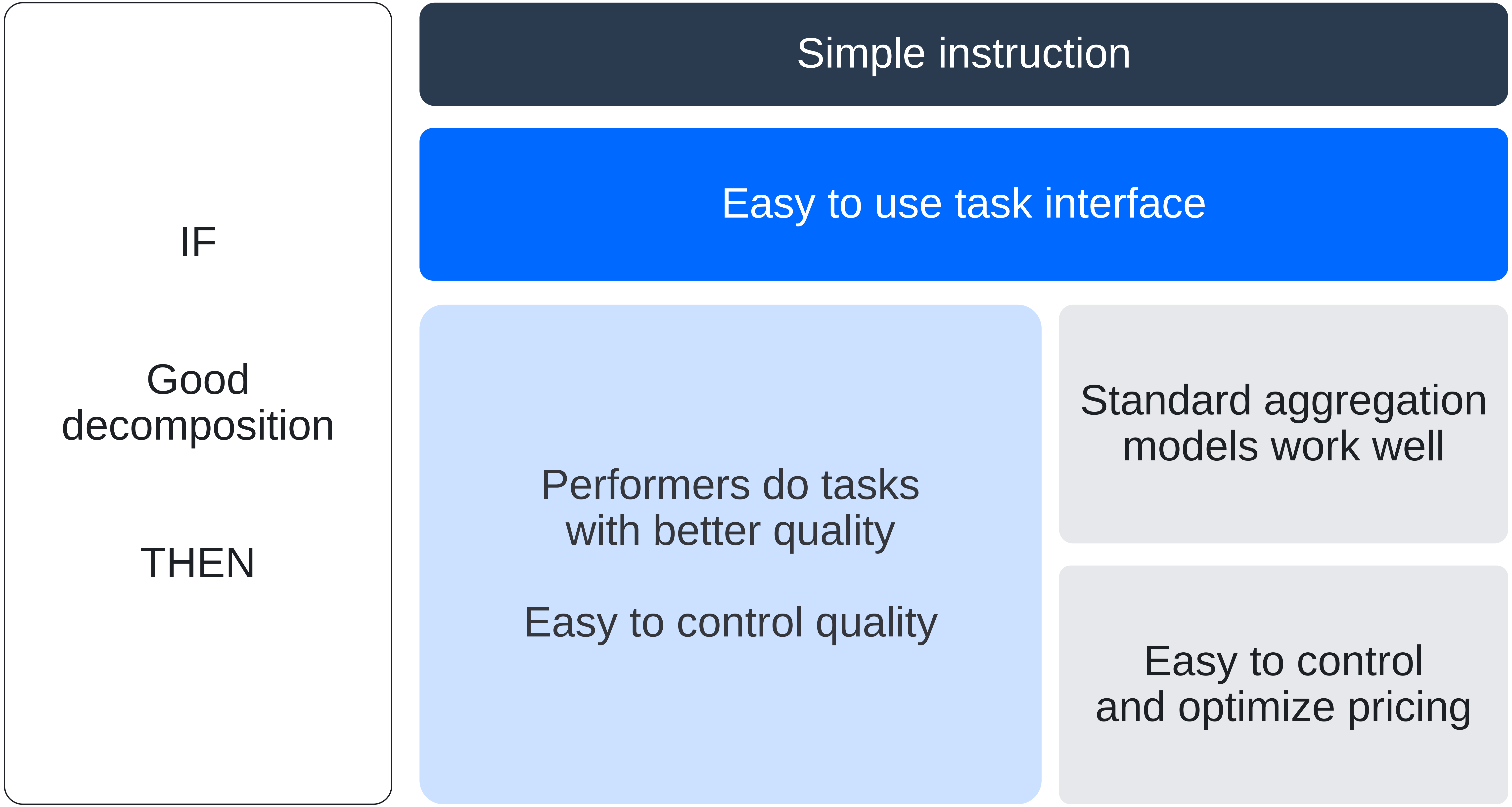
- The lower supply of performers is (e.g. due to specific skills), the higher price
- How quickly do you need accomplished tasks (latency)?

Result quality

- Incentivize better performance by a quality-dependent price

Will be discussed in Part V

Good decomposition is the key to success


Simple instruction
Easy to use task interface
IF
Good decomposition
THEN
Performers do tasks with better quality
Easy to control quality
Standard aggregation models work well
Easy to control and optimize pricing

Part III

Introduction to Toloka for requesters

Evfrosiniya Zerminova,
Head of Data Analysis and Research Group

Key types of instances in Toloka

Project

- Defines the structure of tasks
- Defines how to perform them

Configure in a project

- Input and output data types
- Task interface
- Task instruction

Pool

- Is a batch of tasks
- Defines access of performers

Configure in a pool

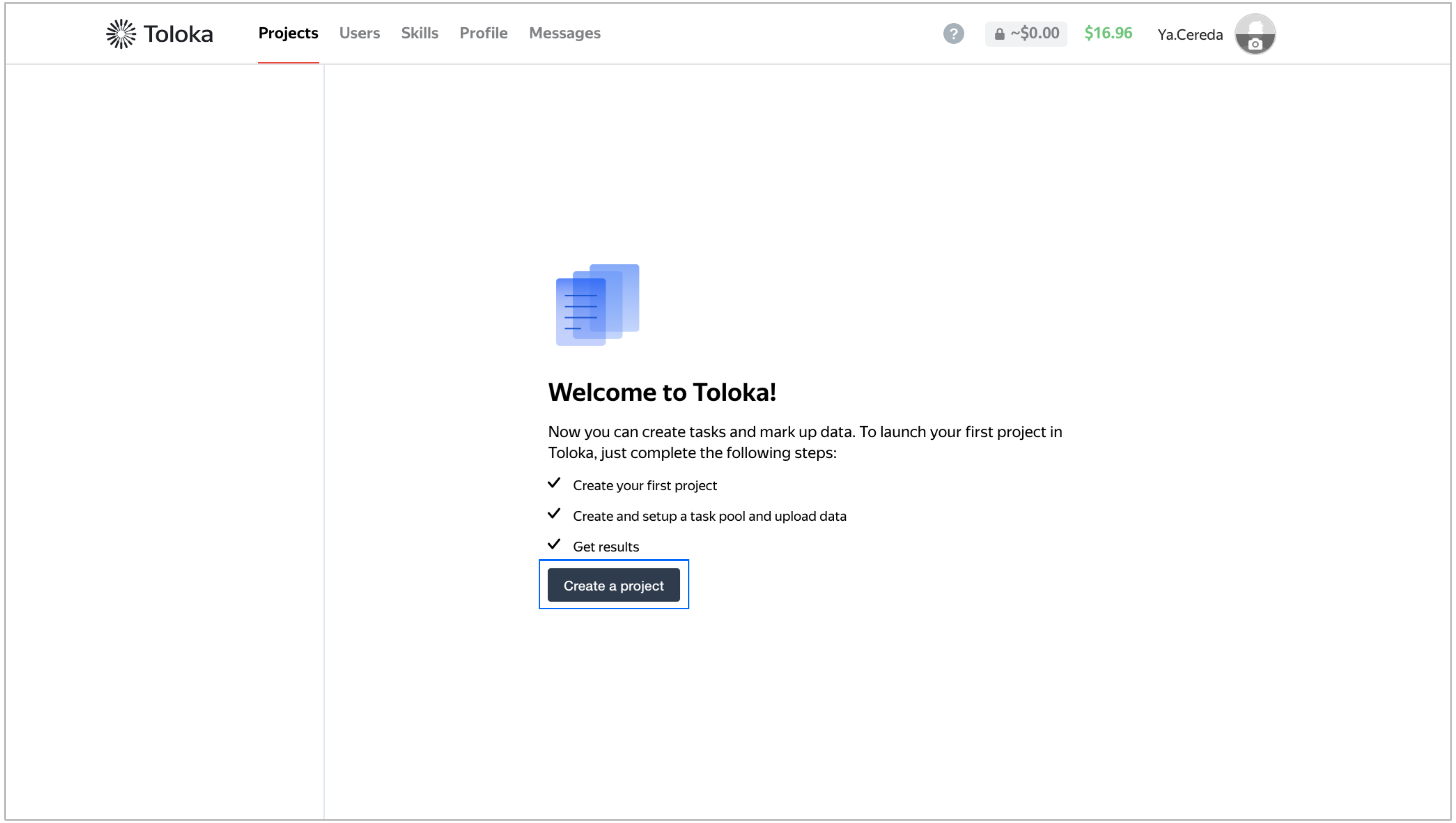
- Performer filters
- Quality control mechanisms
- Overlap settings

Task

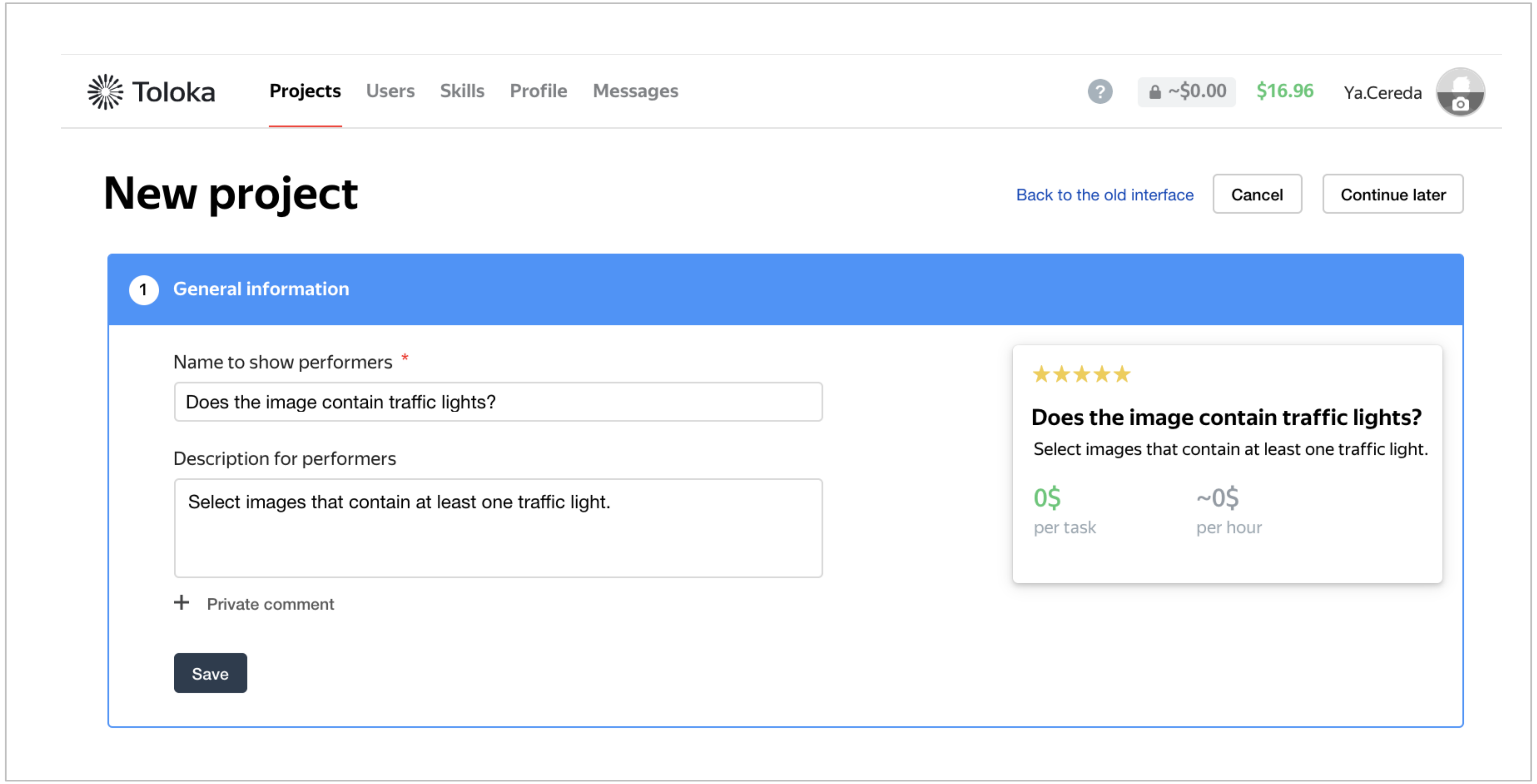
- A particular input data
- Results for it from performers

Project:
creation & configuration

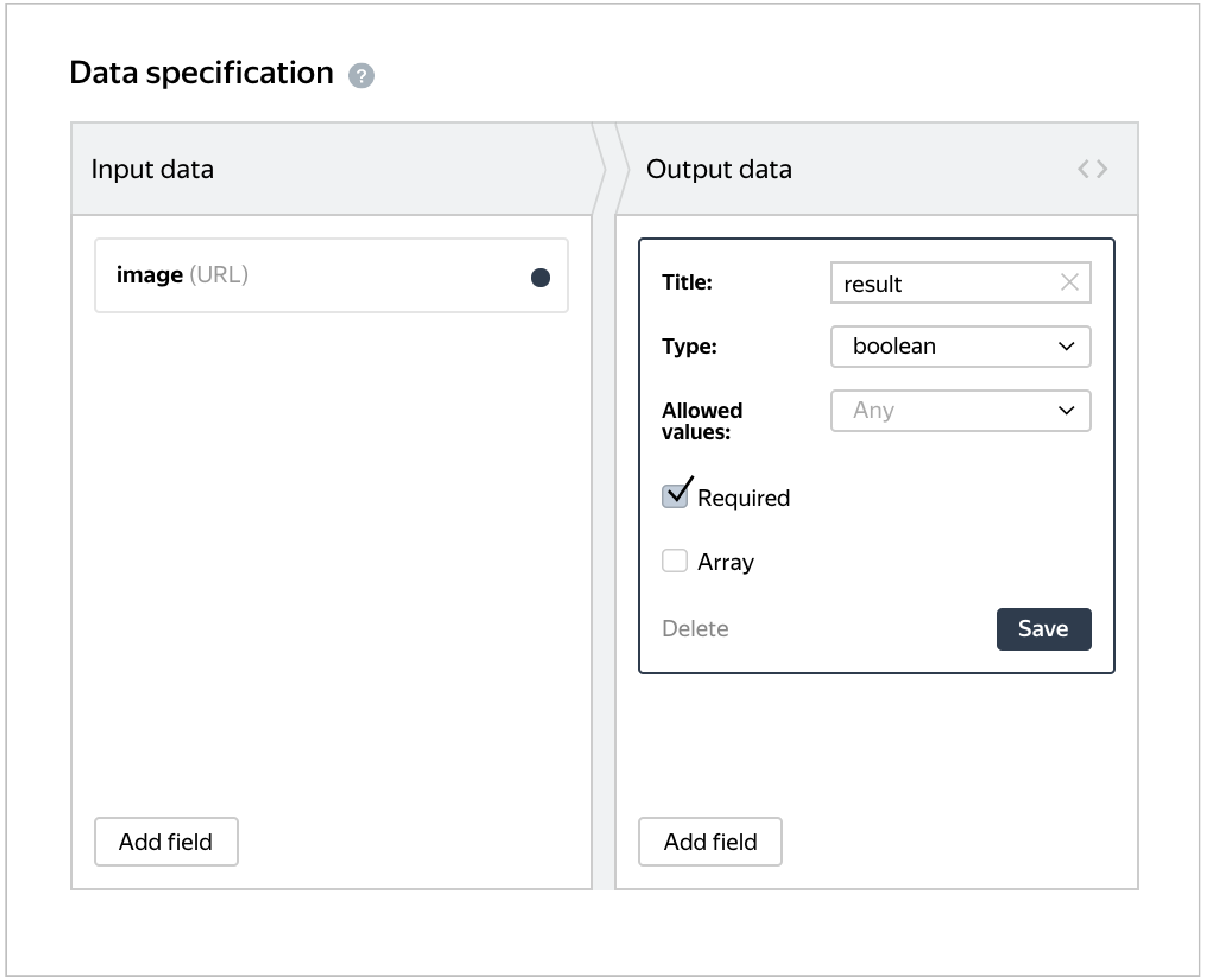
Project: creation



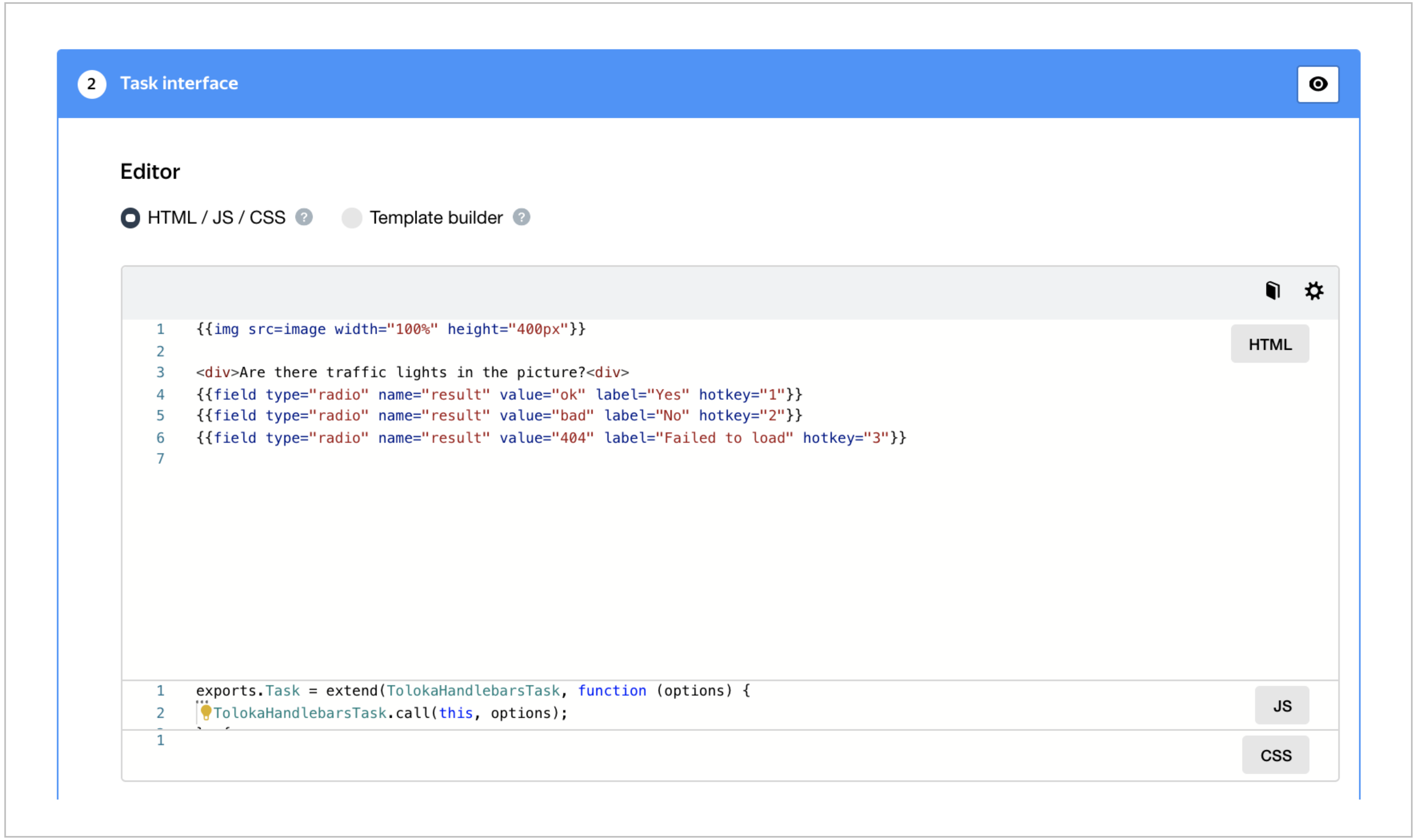
Project: configure name and instruction



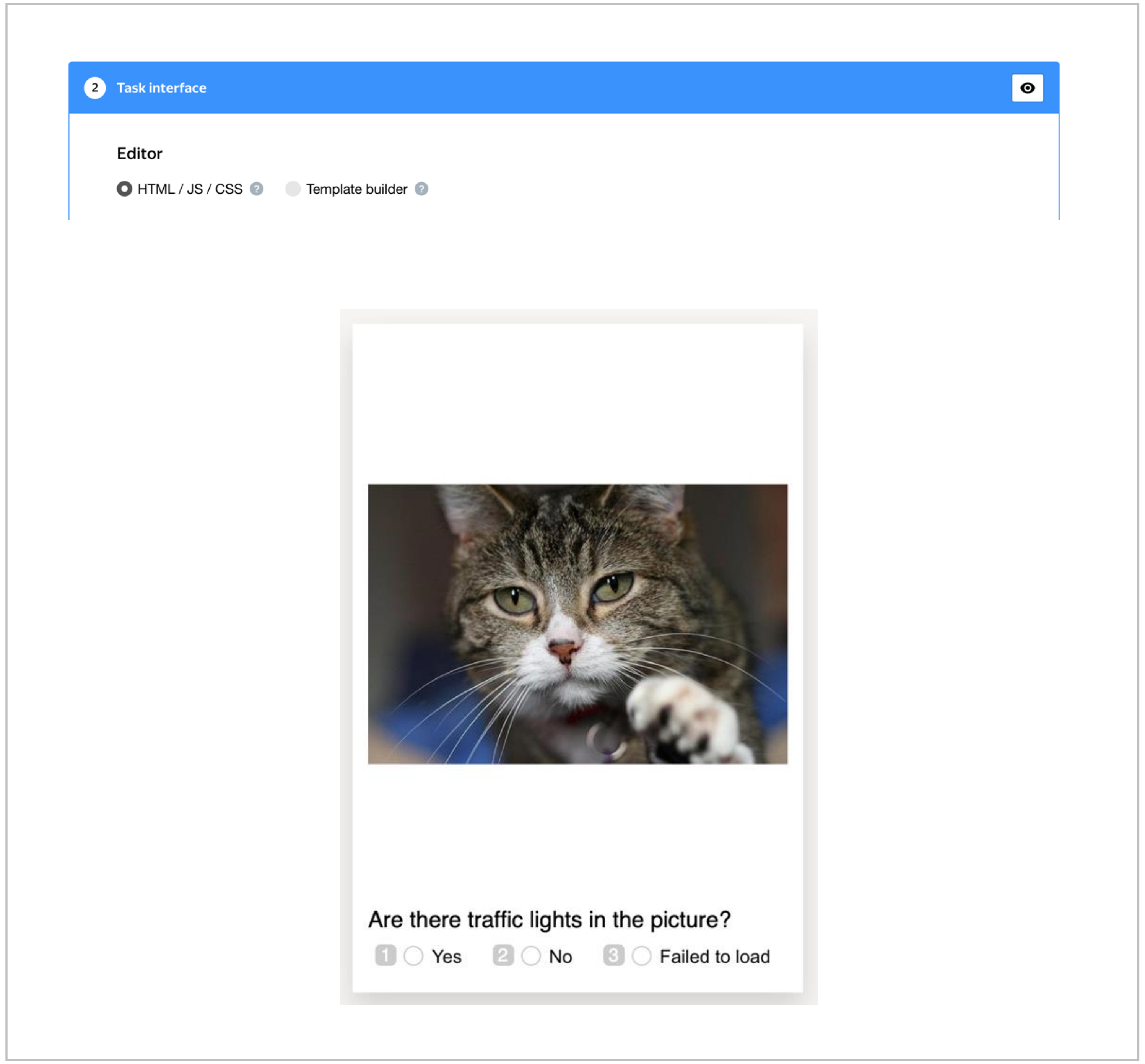
Project: configure in/out data types



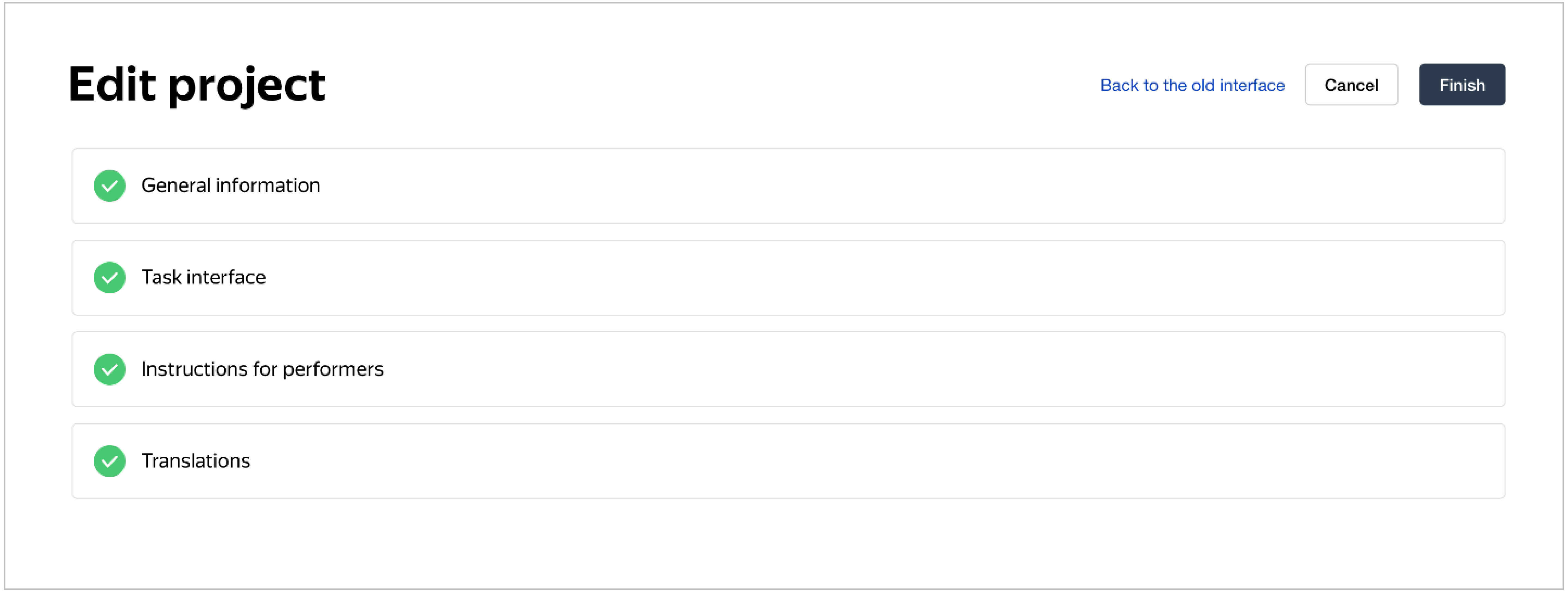
Project: configure interface



Project: use the task preview to see how it works

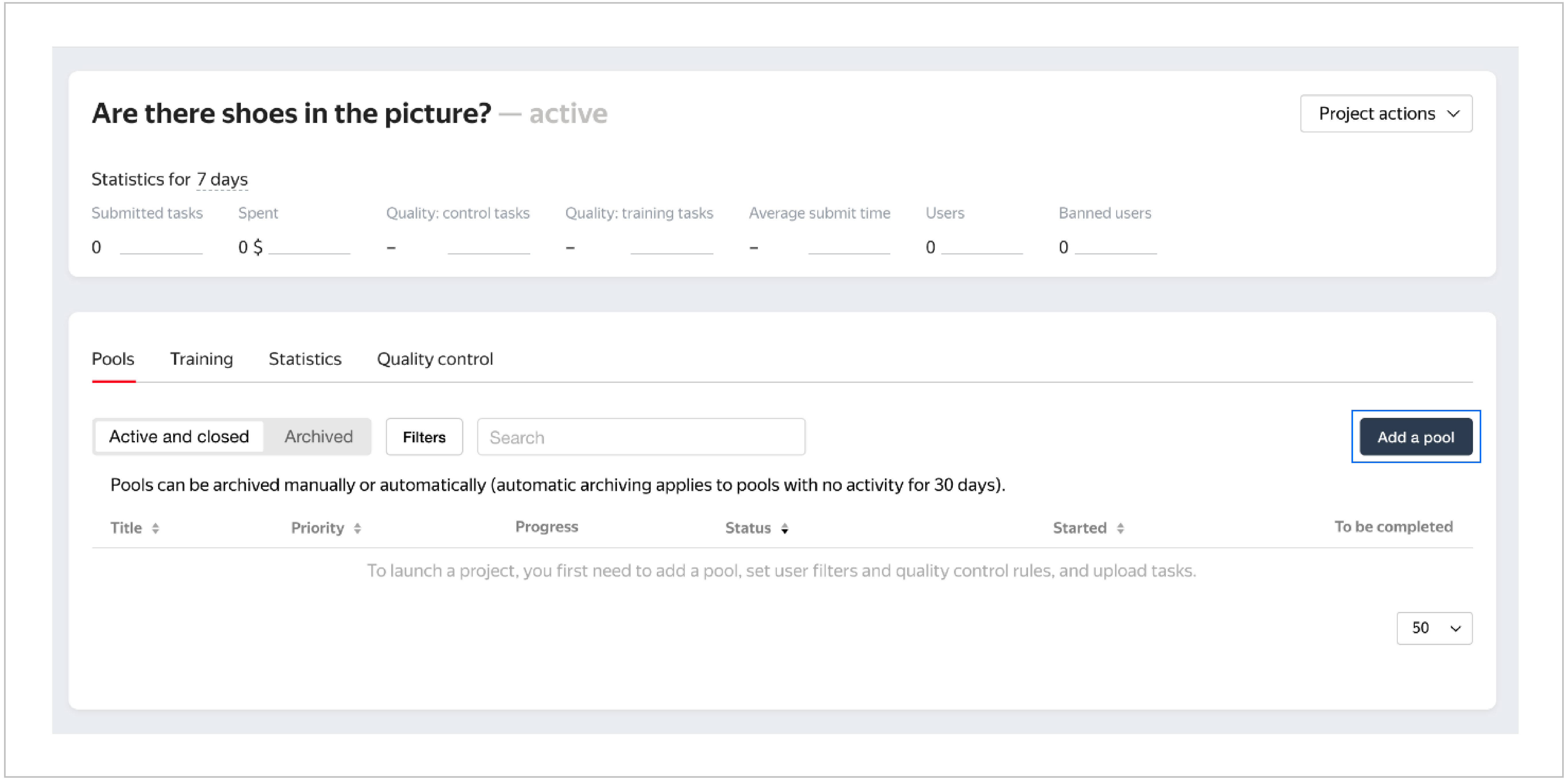


Project: saving

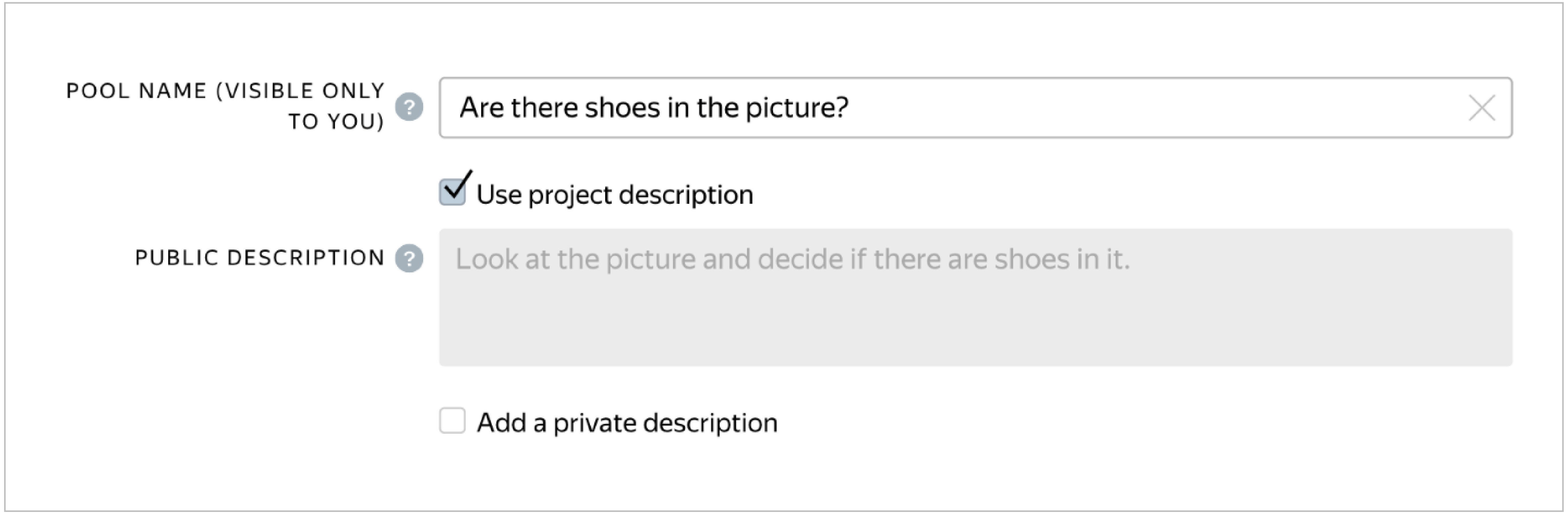


Pool:
creation & configuration

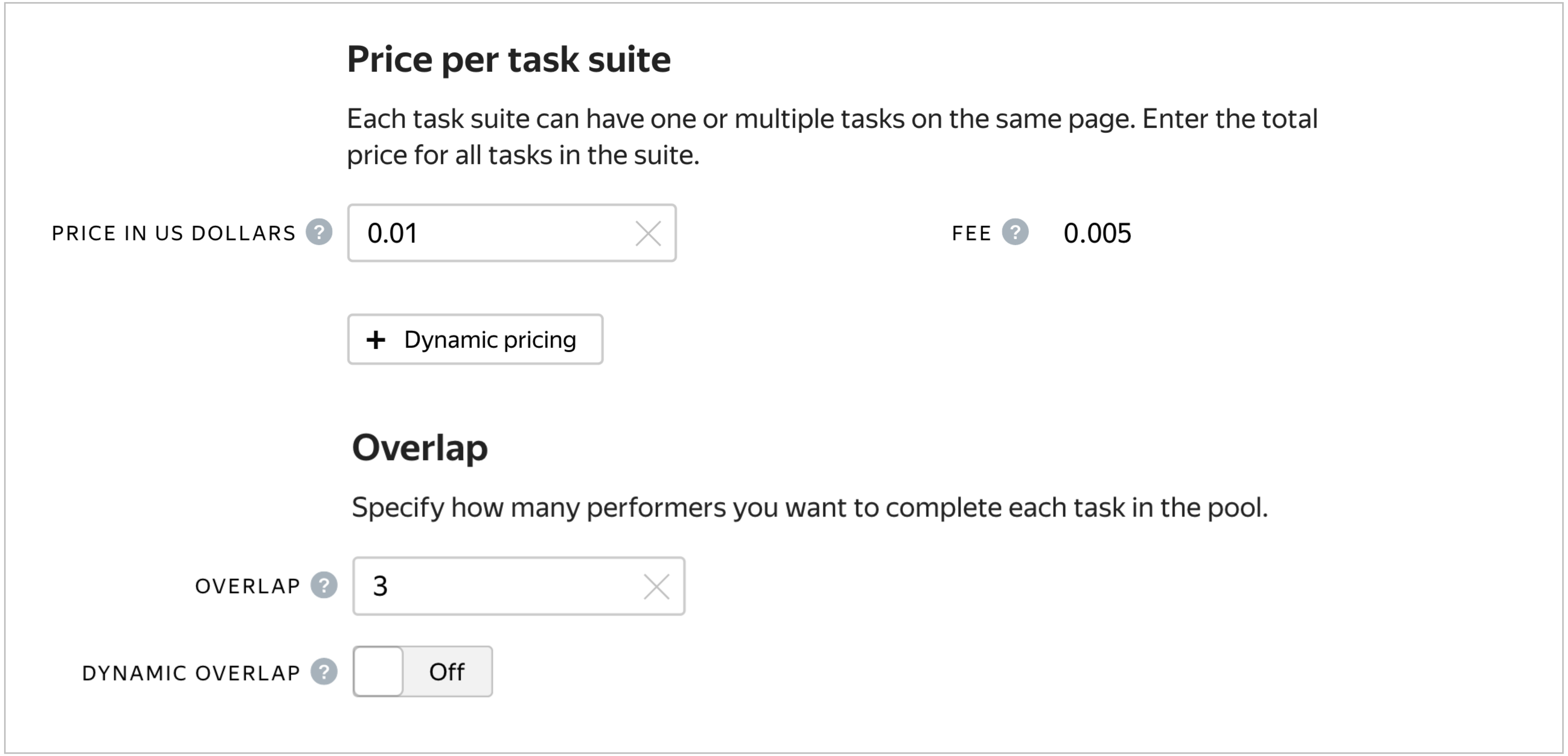
Pool: creation



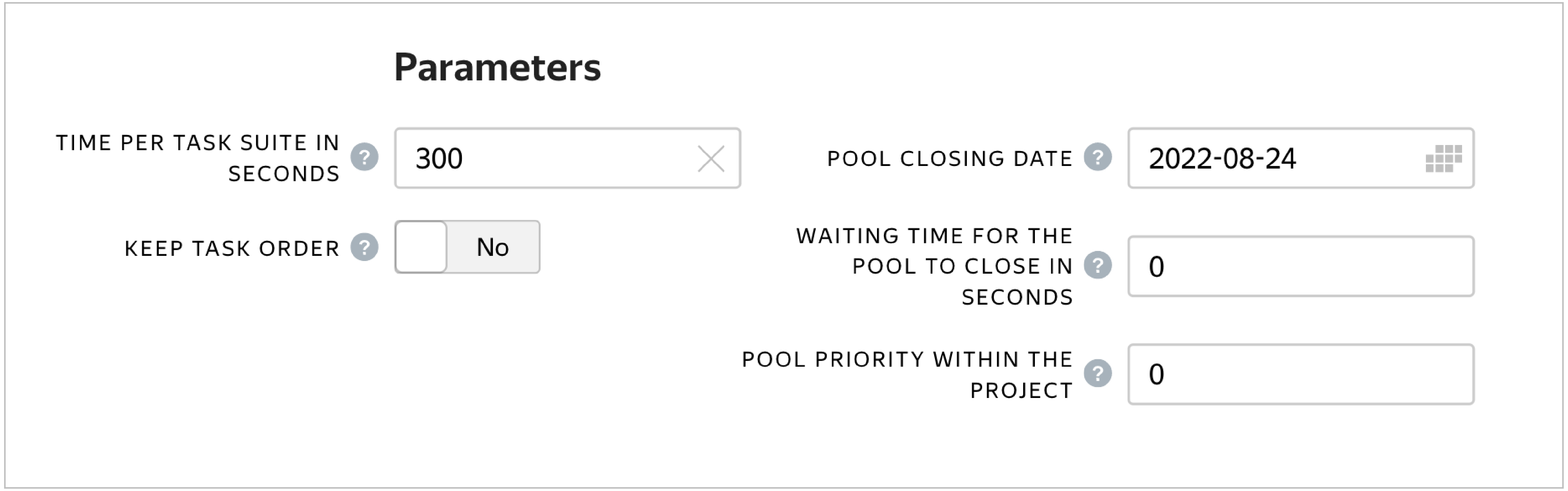
Pool: configure name and description



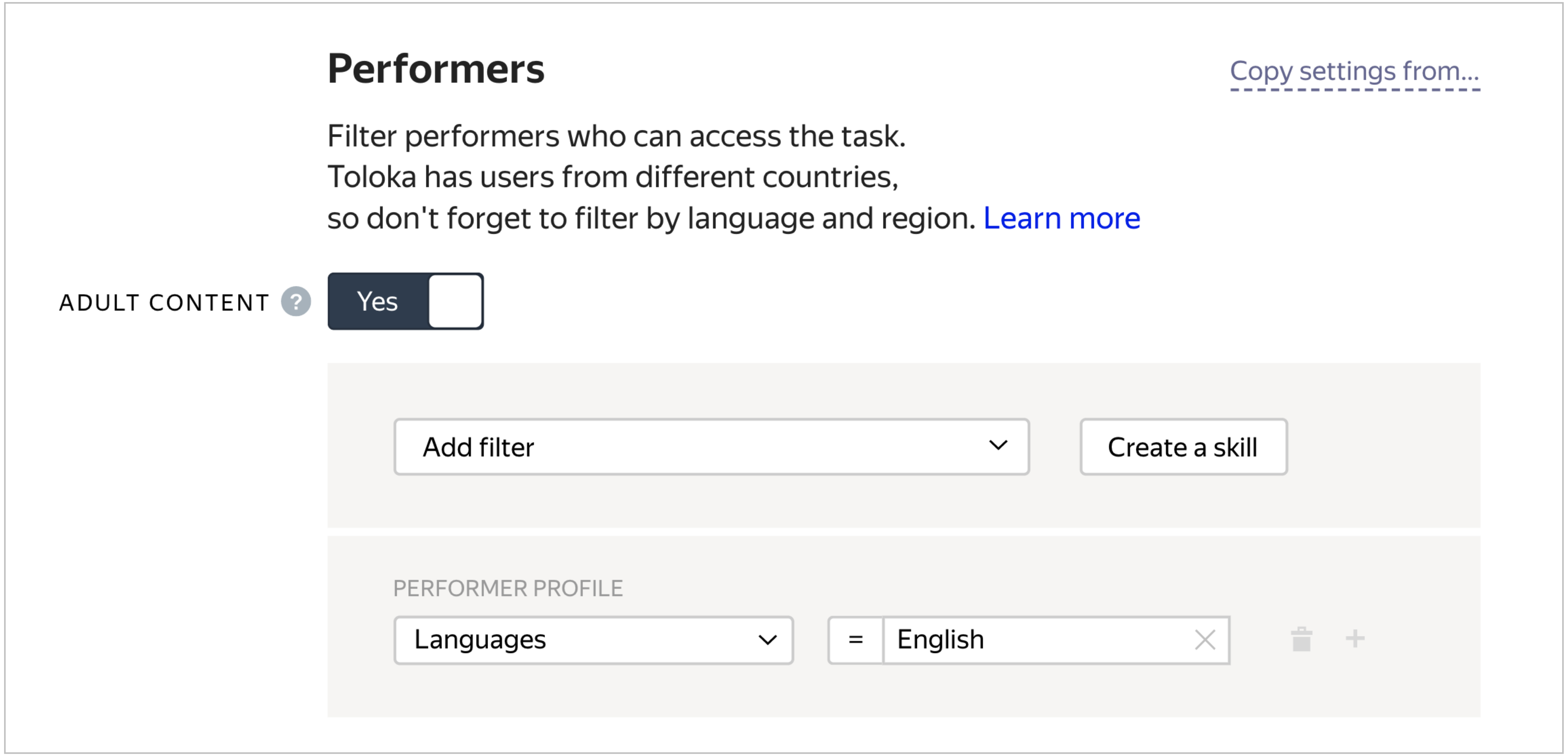
Pool: configure pricing and overlap



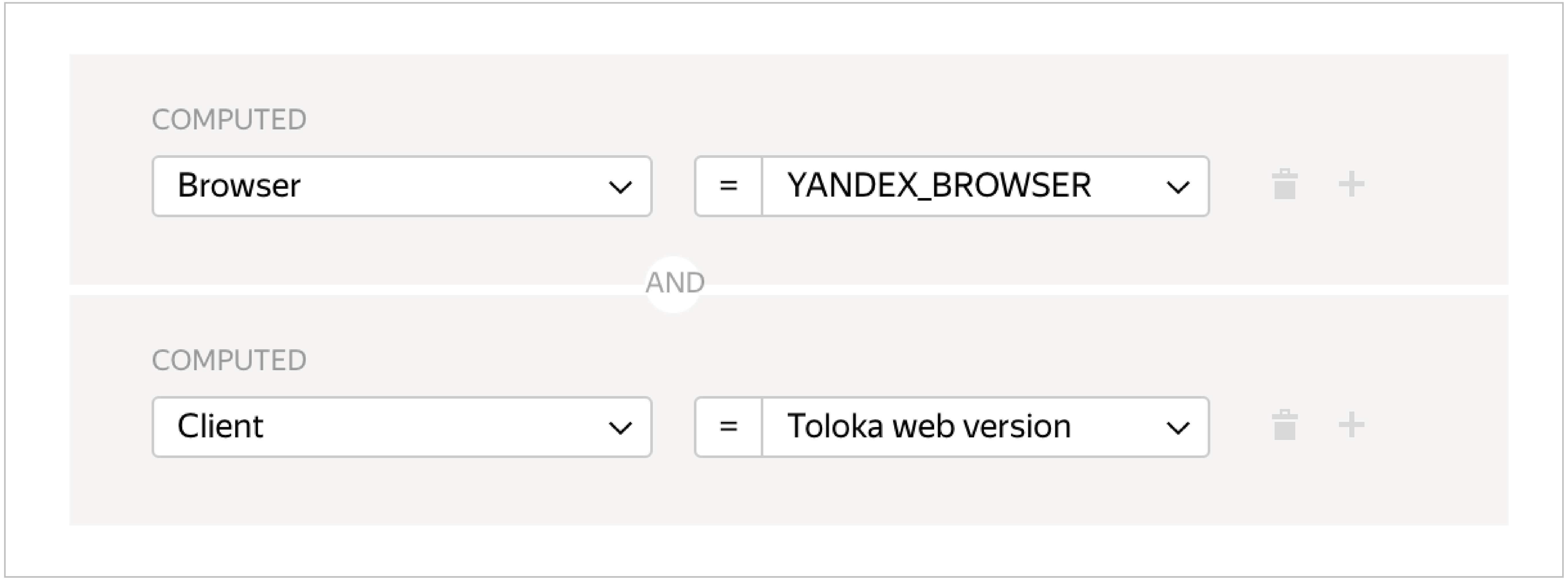
Pool: configure timing and post verification



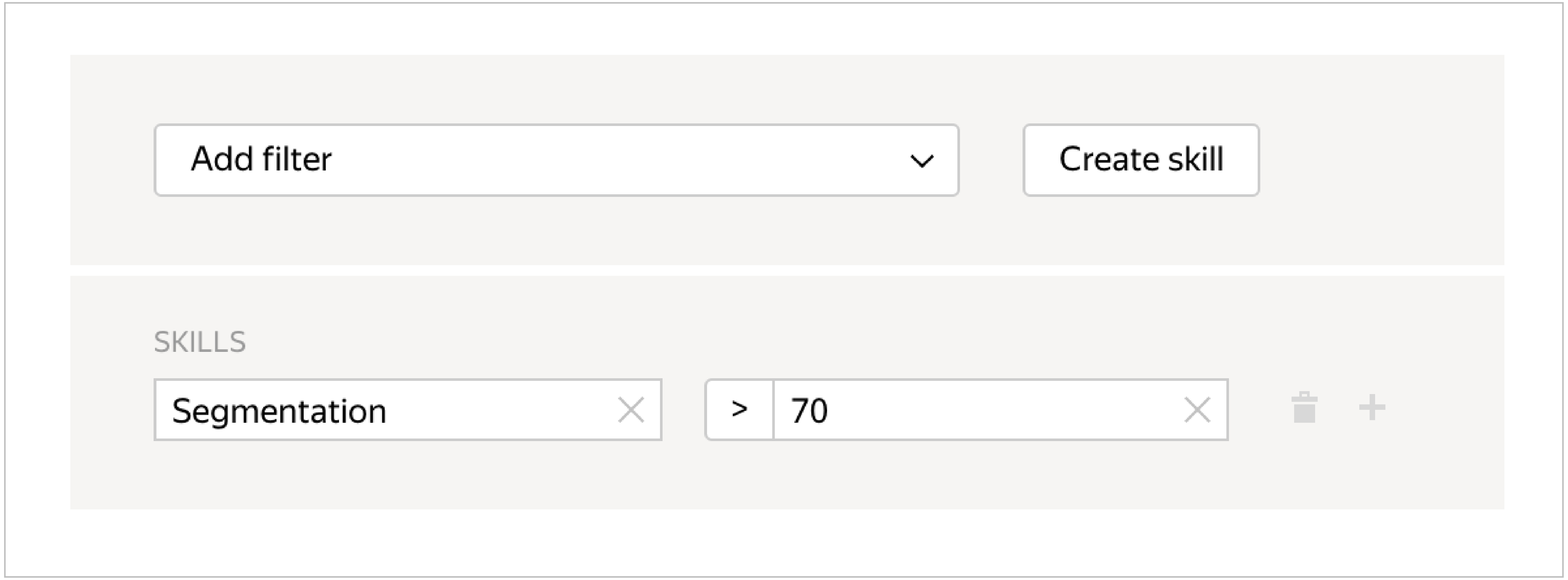
Pool: filter performers by their profiles



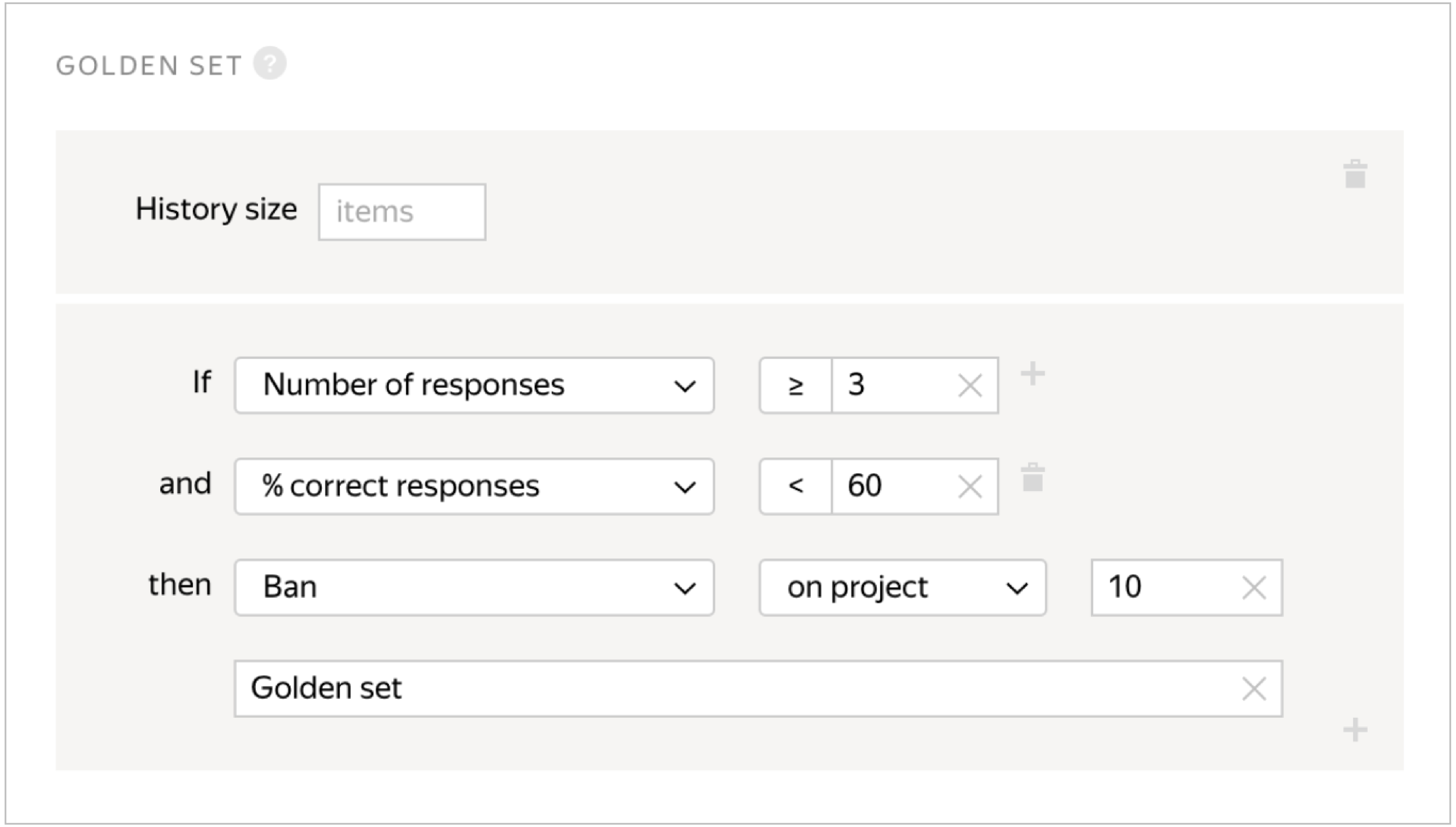
Pool: filter performers by computed properties



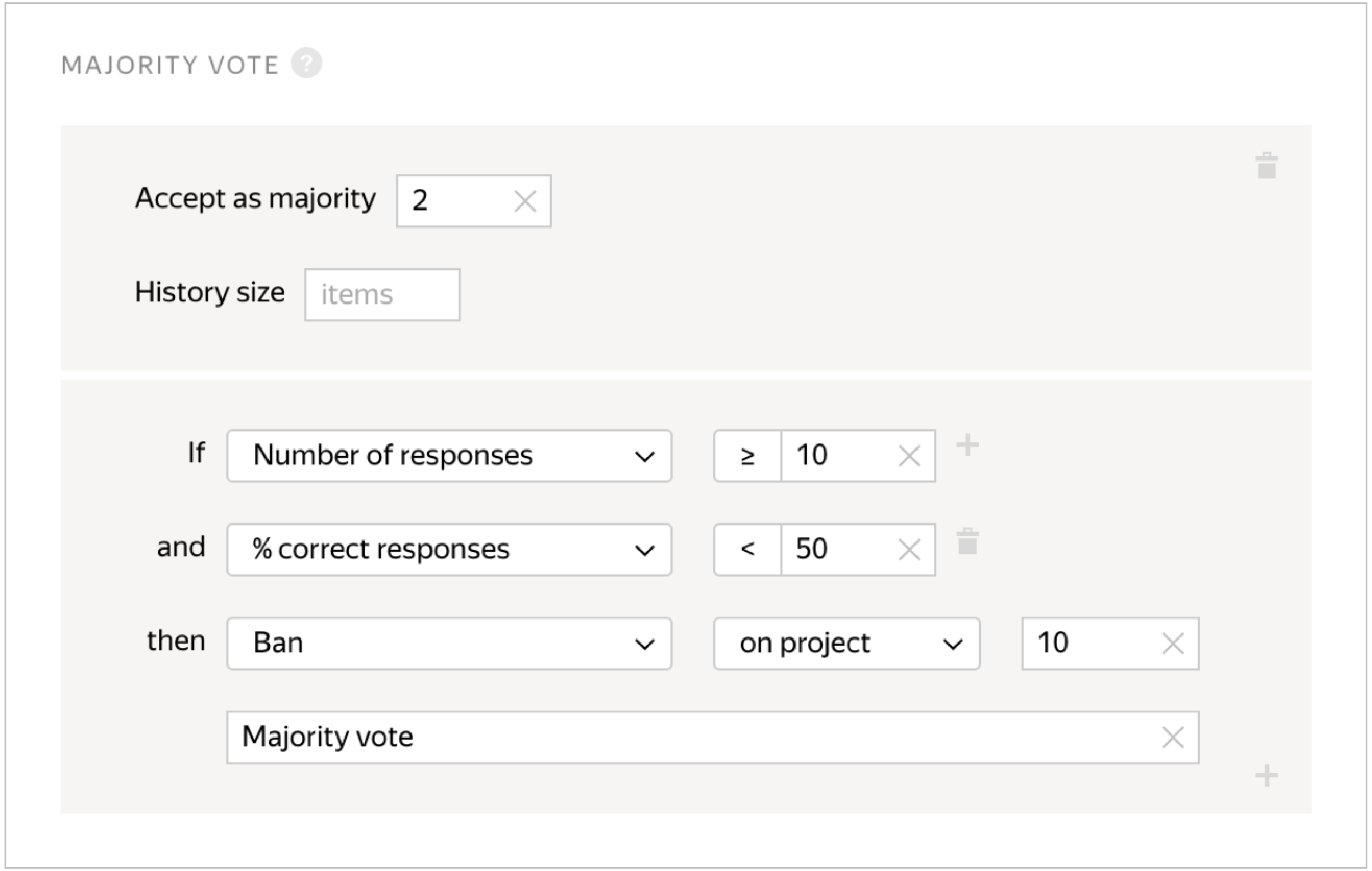
Pool: filter performers by custom skills



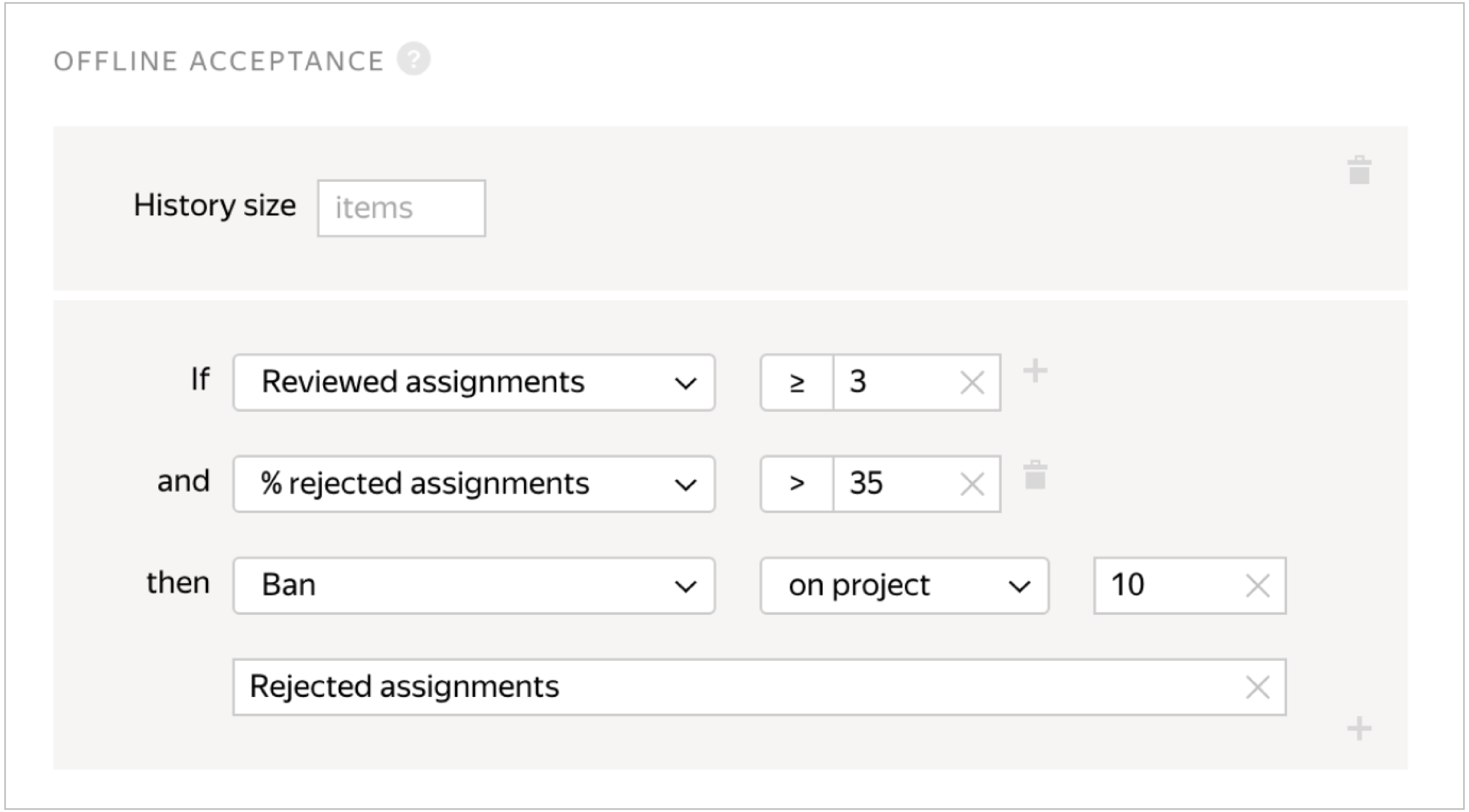
Pool: control quality via a rule on golden set



Pool: control quality via a rule on majority vote

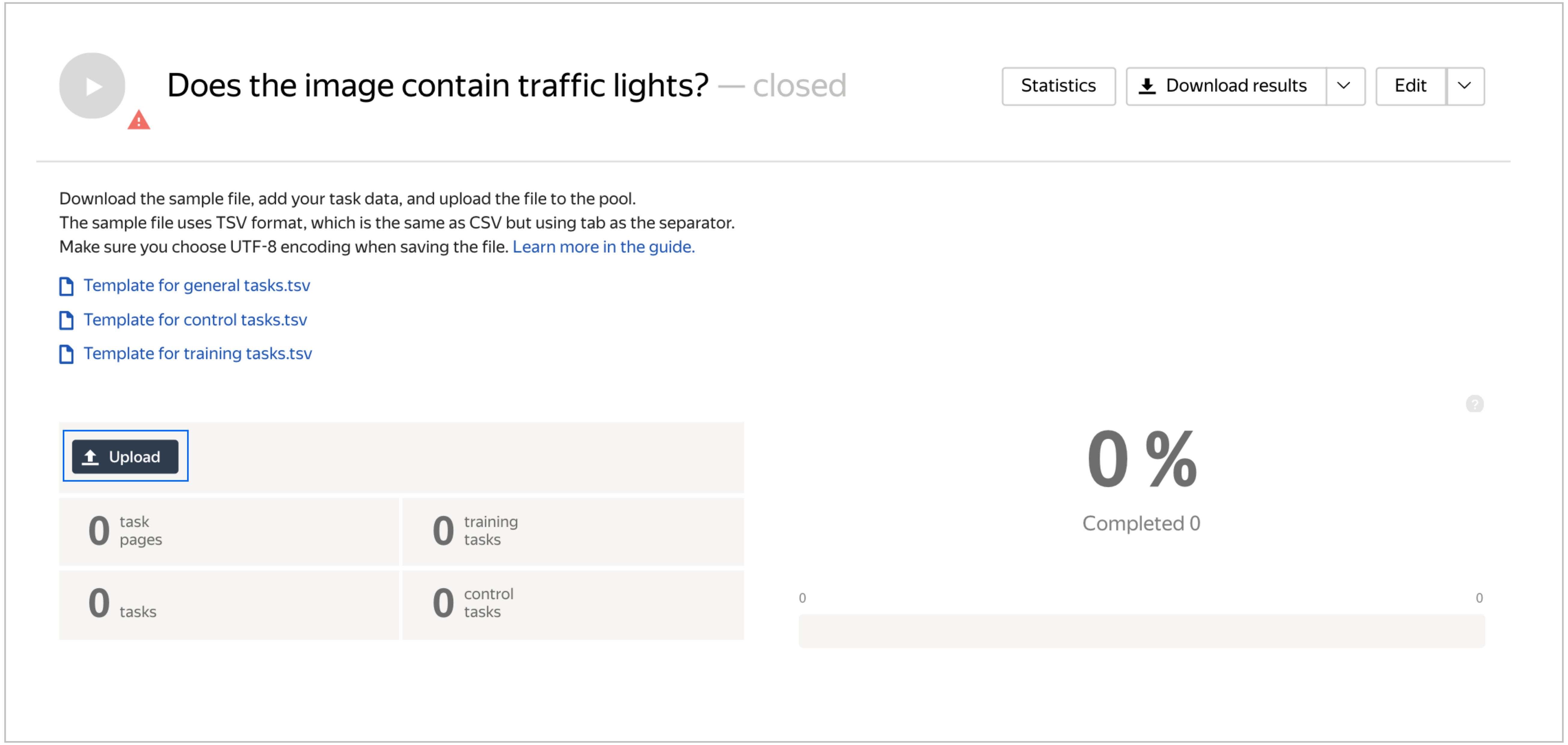


Pool: control quality via a rule on post verification

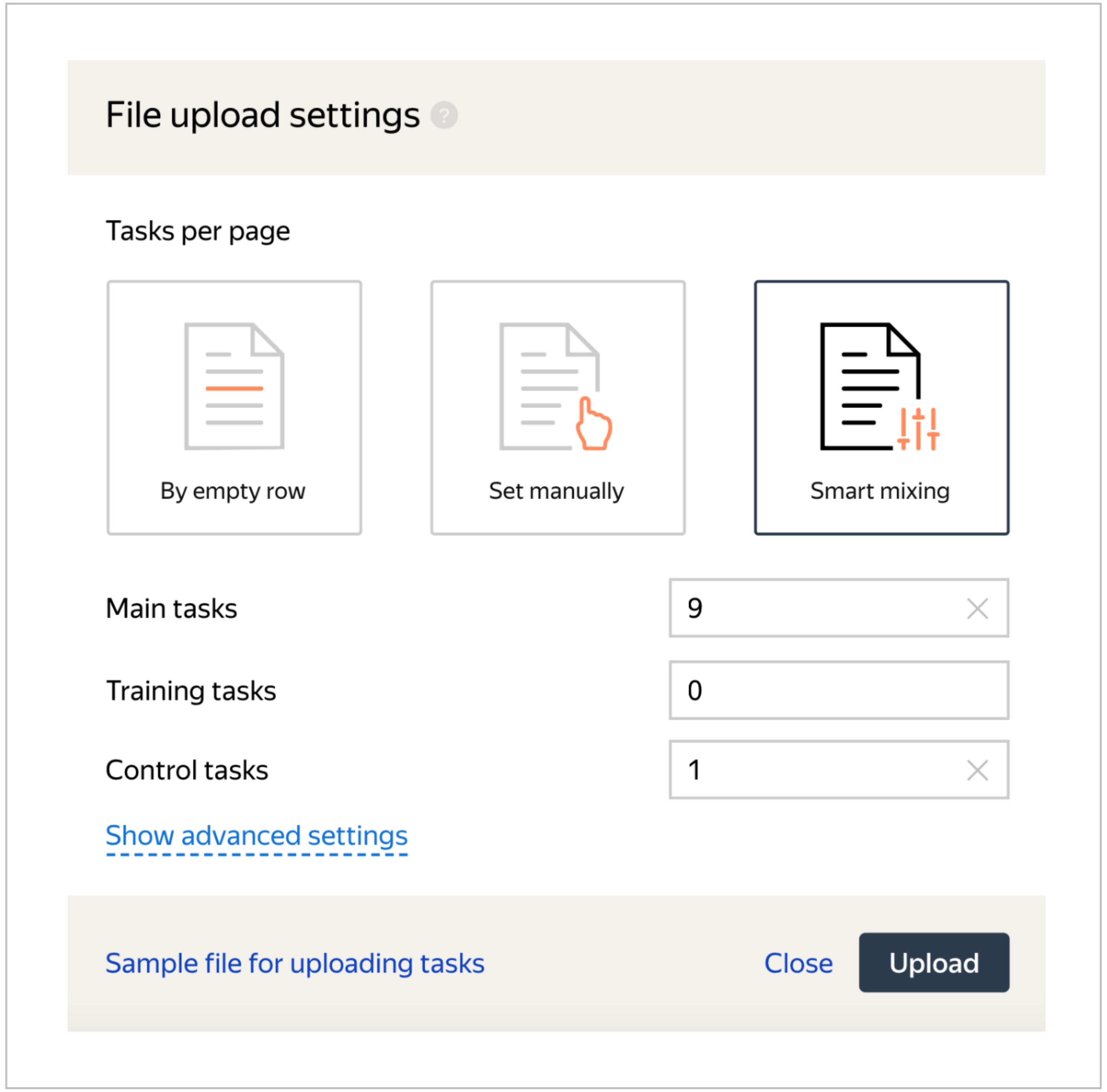


Task uploading & golden set creation

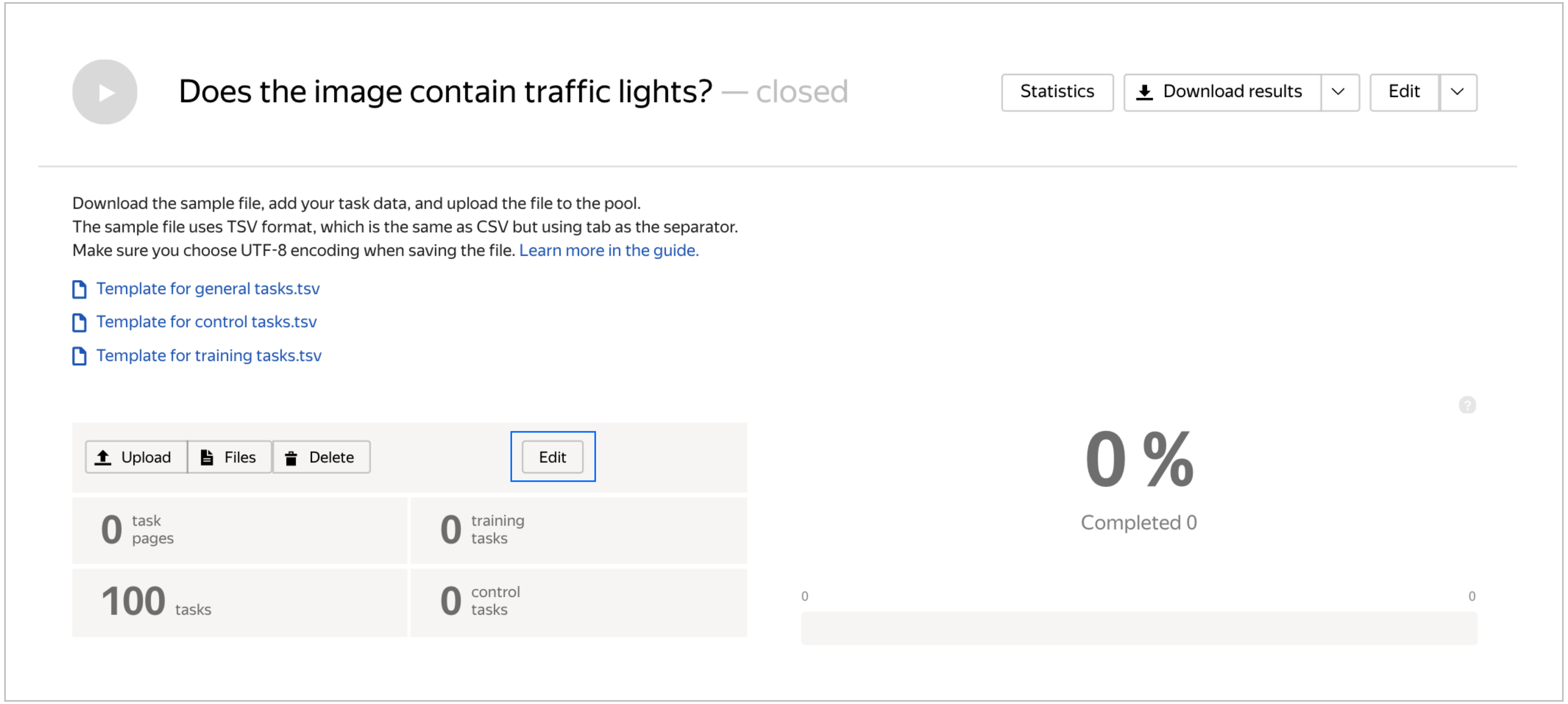
Task: uploading



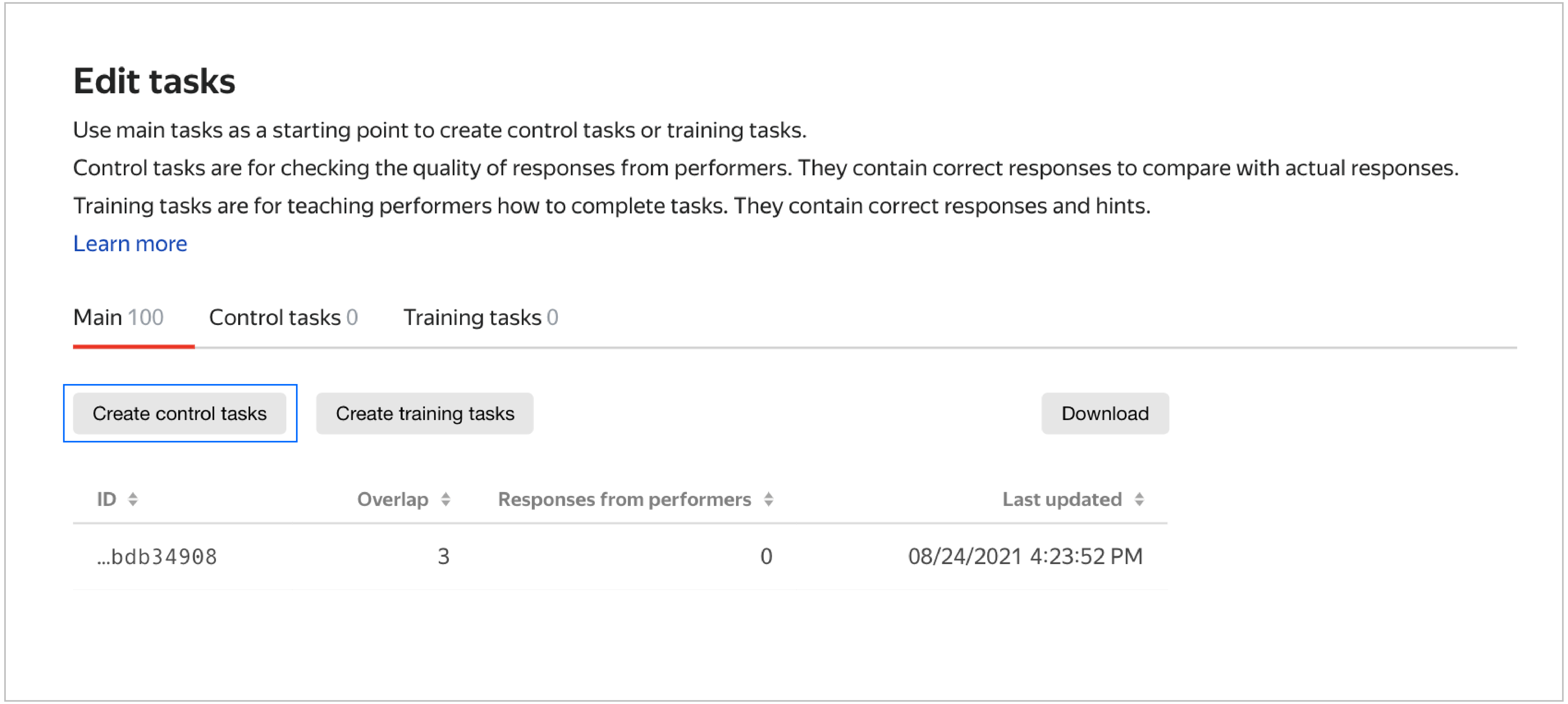
Task: uploading



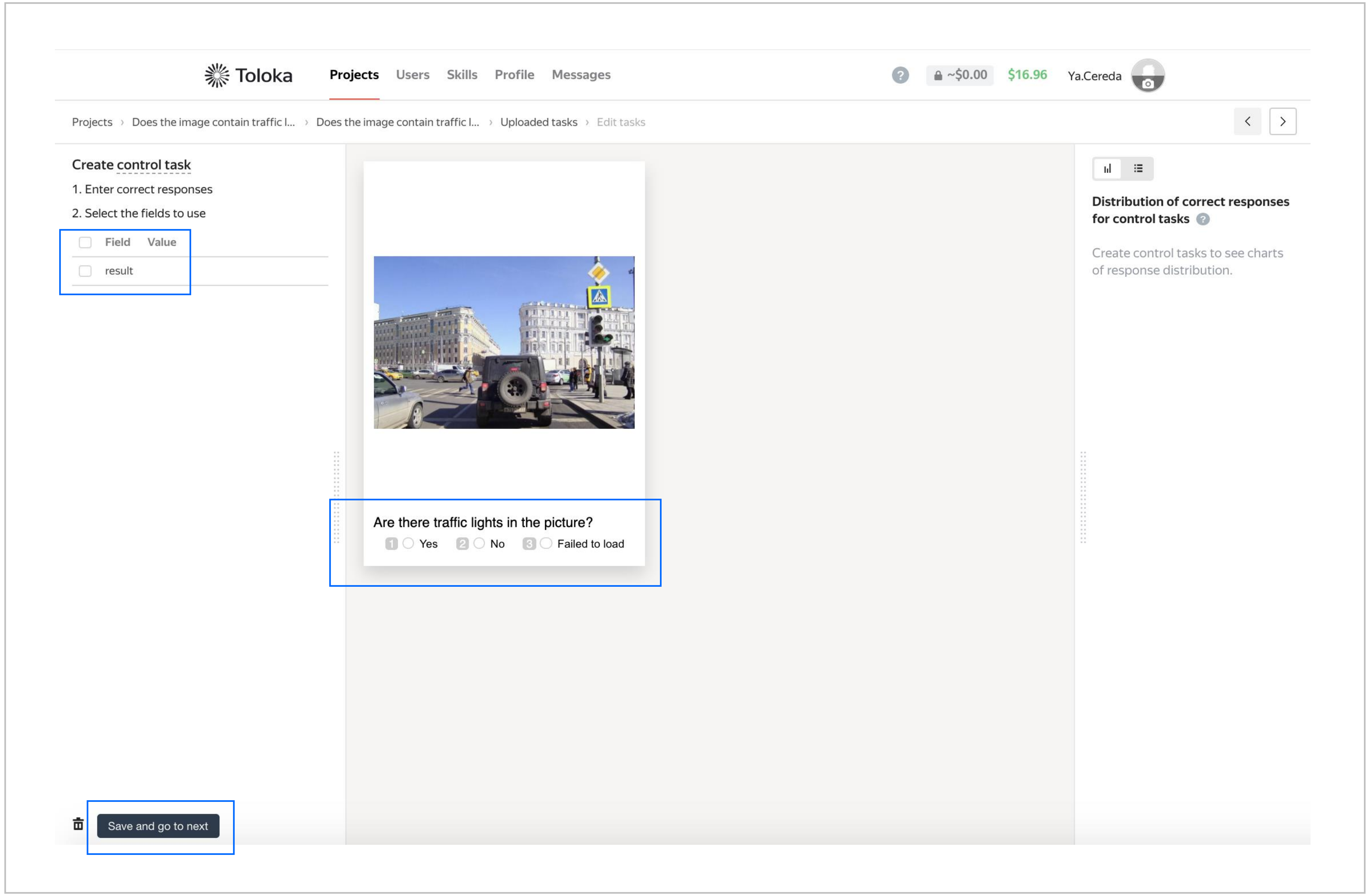
Task: edit for creation of control tasks



Task: control task creation

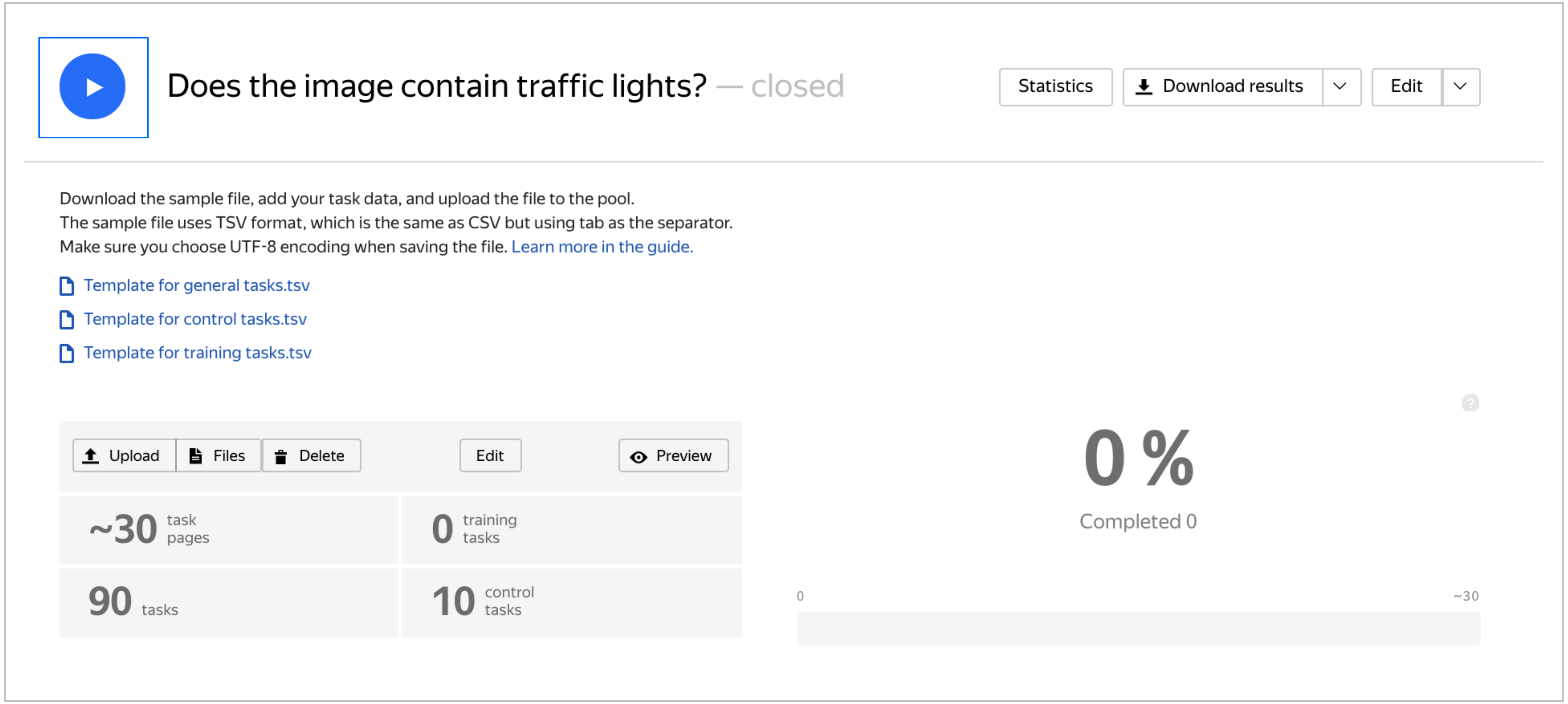


Task: create a control task by answer selection

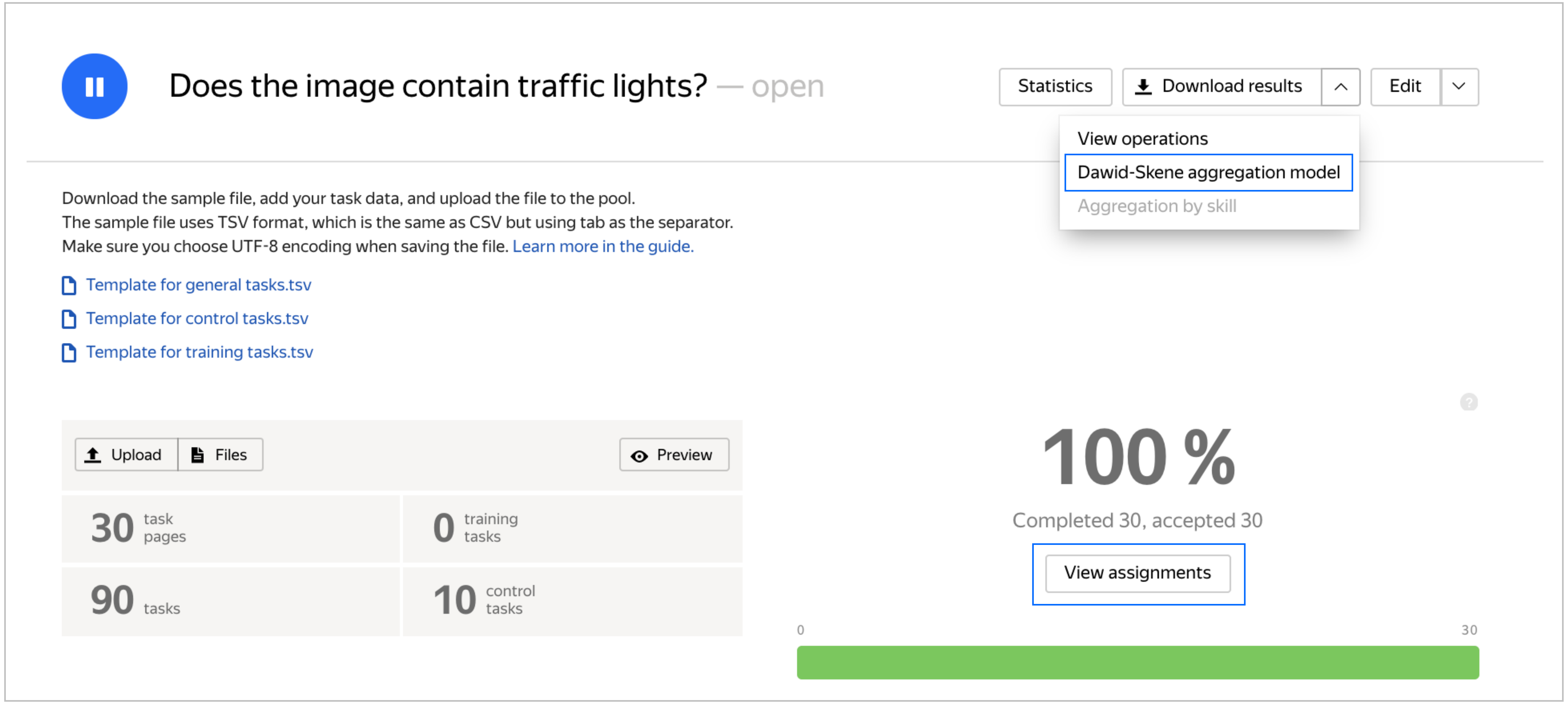


Run the pool & result downloading

Pool running



View and aggregate the results of your tasks



Monitor the statistics on how tasks are performed

Part IV

Setting up and running label collection projects

Olga Megorskaya,
CEO

What you need for the practice session

We are starting the practice session

We give you a card with information and links to:

- A step-by-step instruction to configure and run our crowd projects
- A video version of this instruction
- A dataset of photos that should be labeled
- Login+Password to sign in Yandex.Toloka as a requester

We also provide several copies of a printed version of the instruction

Did everybody receive this card?

Requester account that you received

You have Login+Password to sign in Toloka as a requester

The same account is given for several participants (a group)

- So, you can divide work on the project configuration within this group
- Or, each member of a group may work individually and create the whole pipeline by her/himself

Sign into Toloka as a requester

1. Go to https://toloka.ai

2. Click on “Sign in” in the top-right corner

3. Use received Login+Password to sign in

Requester account that you received

You have Login+Password to sign in Toloka as a requester

The account of this requester has money

► So, you will run your tasks on real crowd performers!

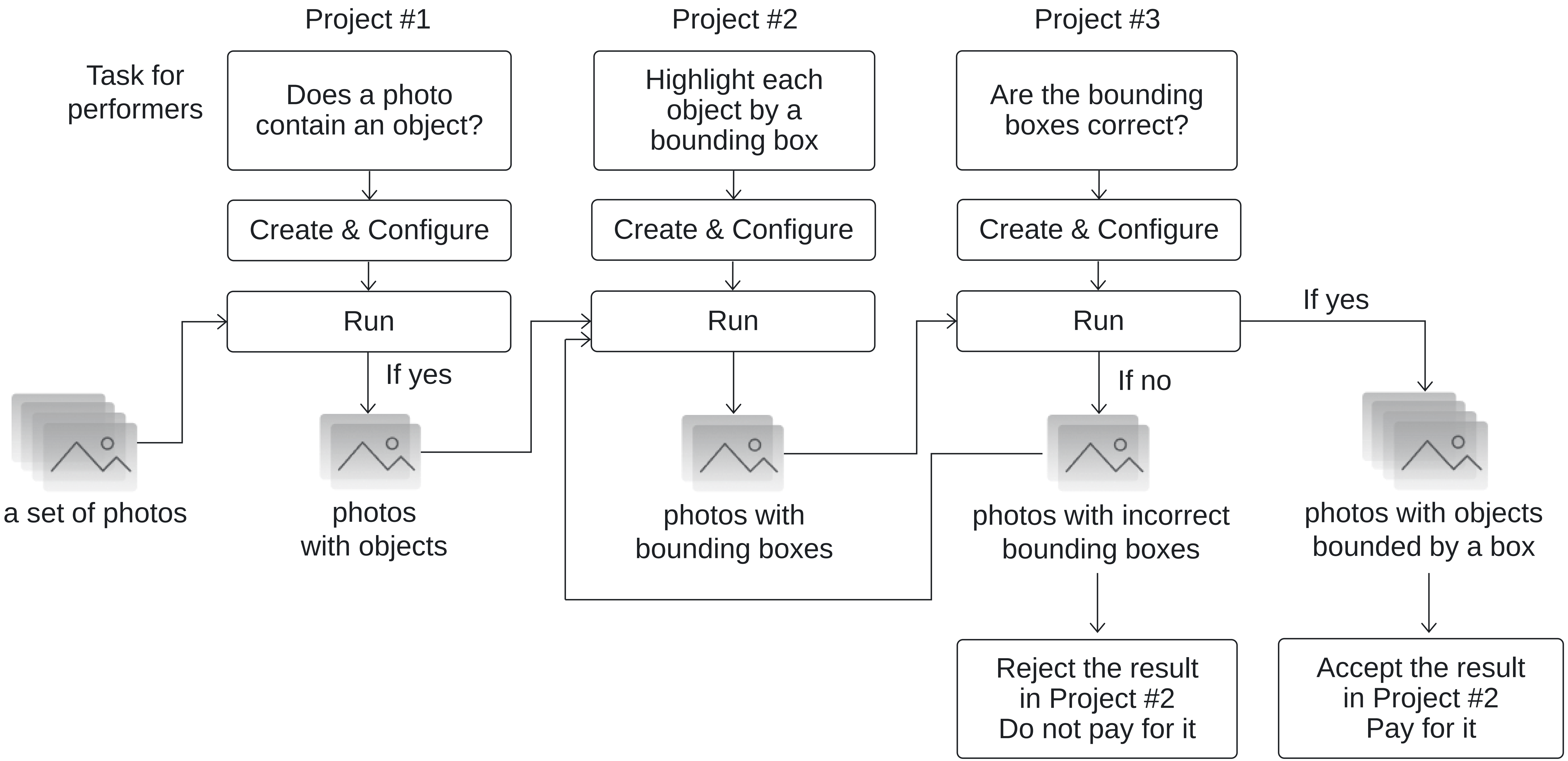
Practice: creating a real crowdsourcing pipeline

Now we will create a real simplified crowdsourcing pipeline

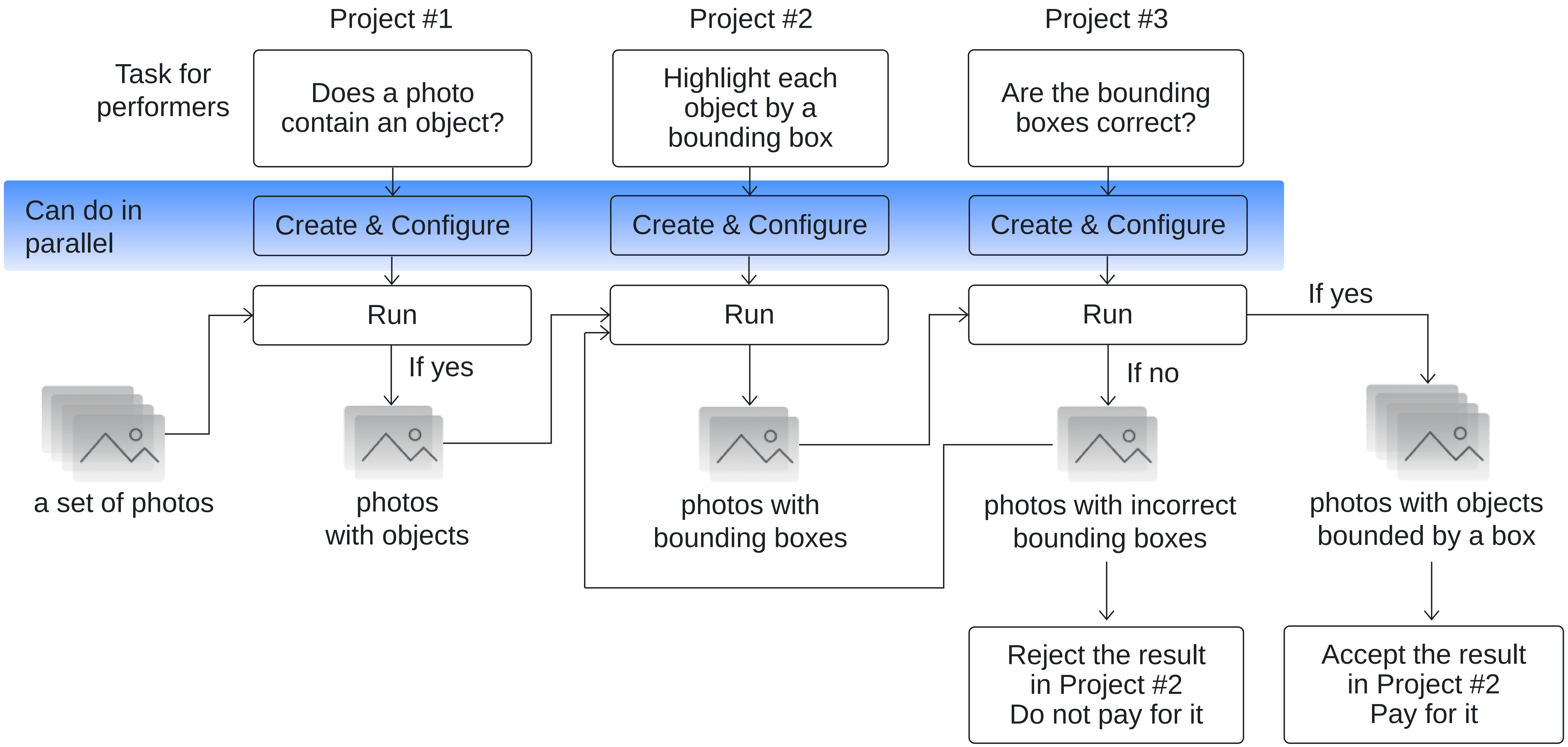
To simplify the task, we ask you to:

- Highlight **one type** of objects
- Choose any type of object you want to highlight. For example, **Traffic signs**
- Use **Bounding Boxes**

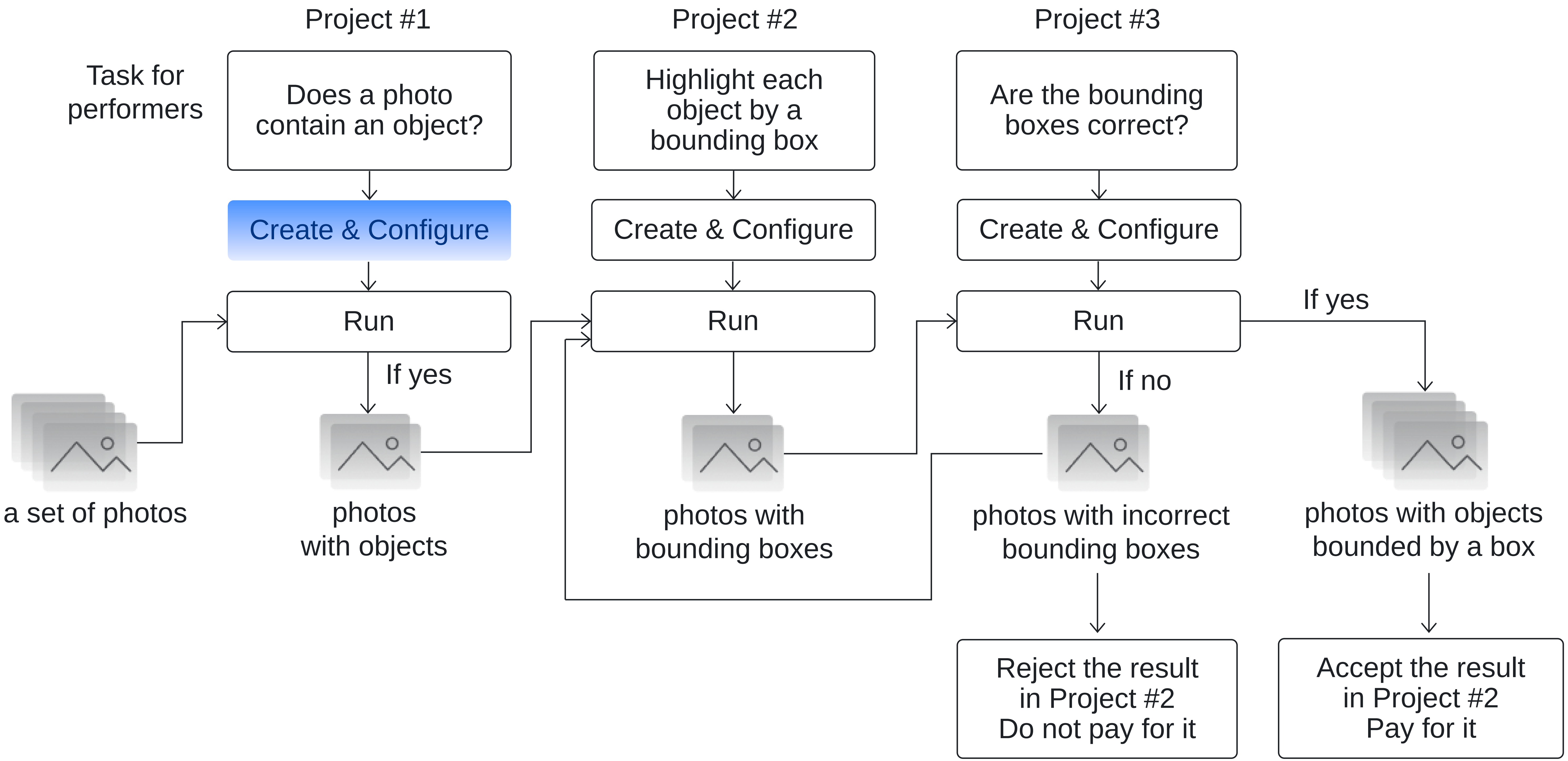
Reminder: we implement and run our pipeline



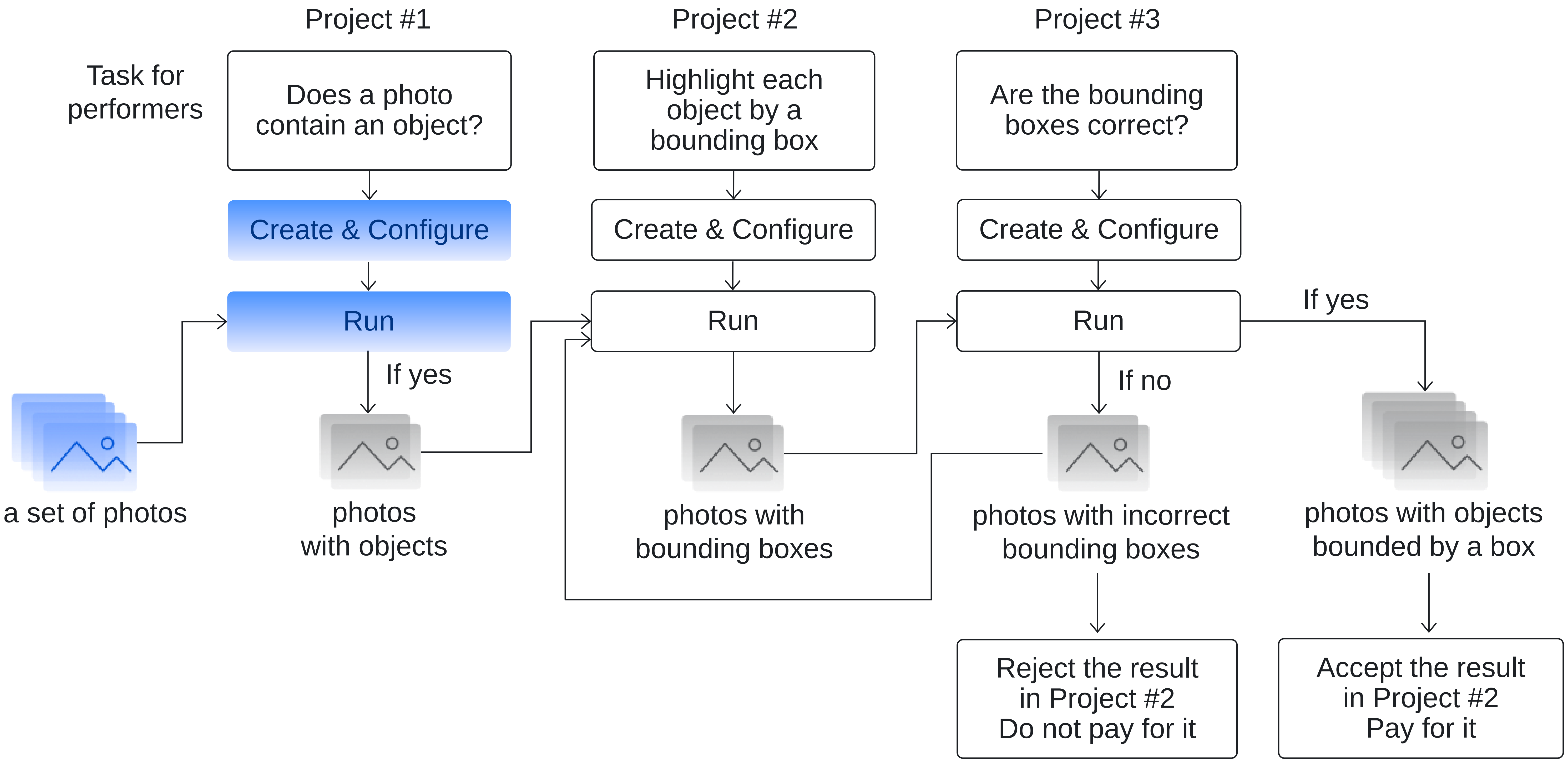
You can divide work within a participant group



Step #1

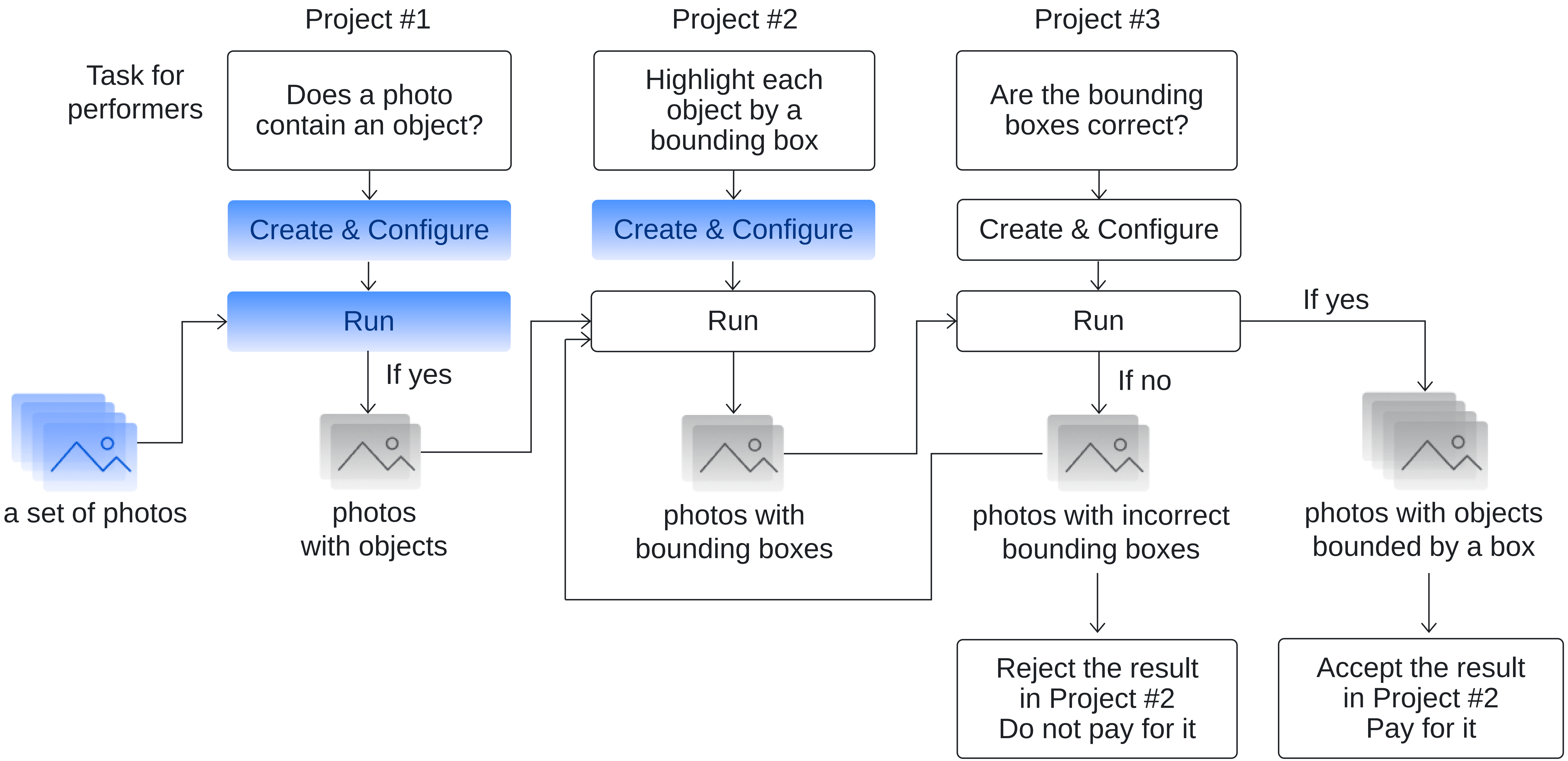

Project #1
Project #2
Project #3
Task for performers
Does a photo contain an object?
Highlight each object by a bounding box
Are the bounding boxes correct?
Create & Configure
Create & Configure
Create & Configure
Run
Run
Run
If yes
If no
If yes
a set of photos
photos with objects
photos with bounding boxes
photos with incorrect bounding boxes
photos with objects bounded by a box
Reject the result in Project #2 Do not pay for it
Accept the result in Project #2 Pay for it

Step #2

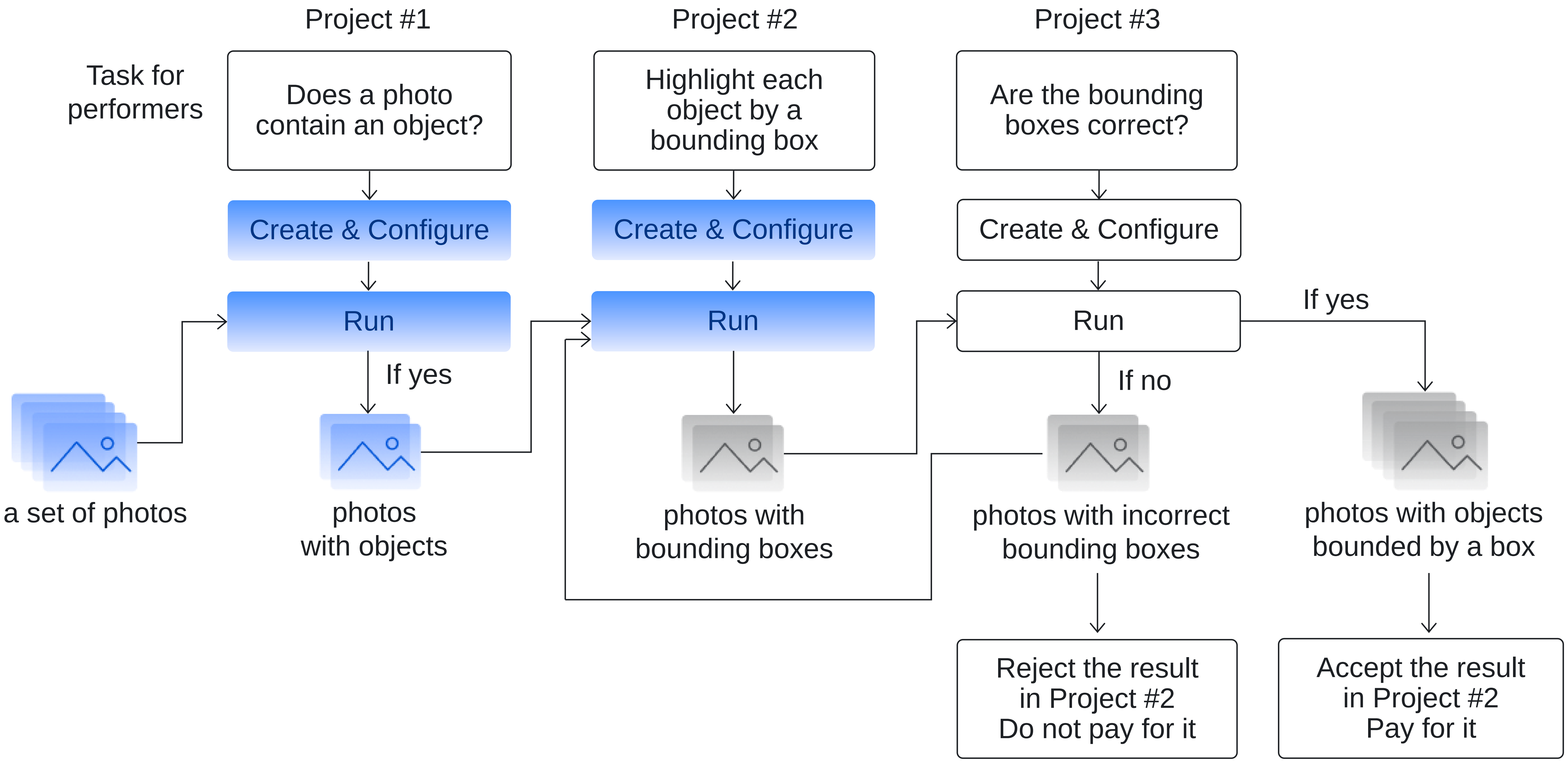


Project #1
Project #2
Project #3
Task for performers
Does a photo contain an object?
Highlight each object by a bounding box
Are the bounding boxes correct?
Create & Configure
Create & Configure
Create & Configure
Run
Run
Run
If yes
If yes
If no
a set of photos
photos with objects
photos with bounding boxes
photos with incorrect bounding boxes
photos with objects bounded by a box
Reject the result in Project #2 Do not pay for it
Accept the result in Project #2 Pay for it

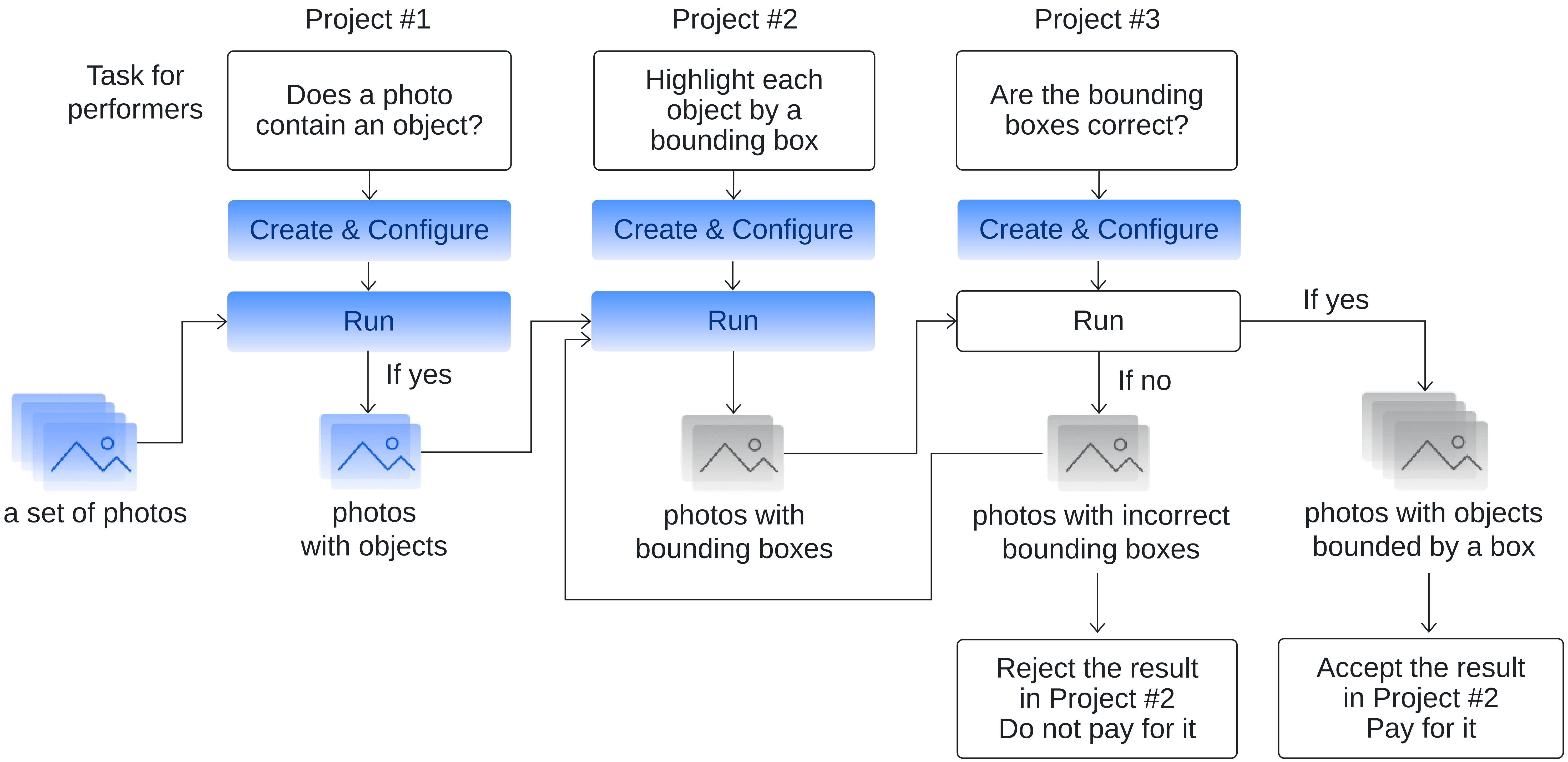
Step #3


Project #1
Project #2
Project #3
Task for performers
Does a photo contain an object?
Highlight each object by a bounding box
Are the bounding boxes correct?
Create & Configure
Create & Configure
Create & Configure
Run
Run
Run
If yes
If yes
If no
a set of photos
photos with objects
photos with bounding boxes
photos with incorrect bounding boxes
photos with objects bounded by a box
Reject the result in Project #2 Do not pay for it
Accept the result in Project #2 Pay for it

Step #4

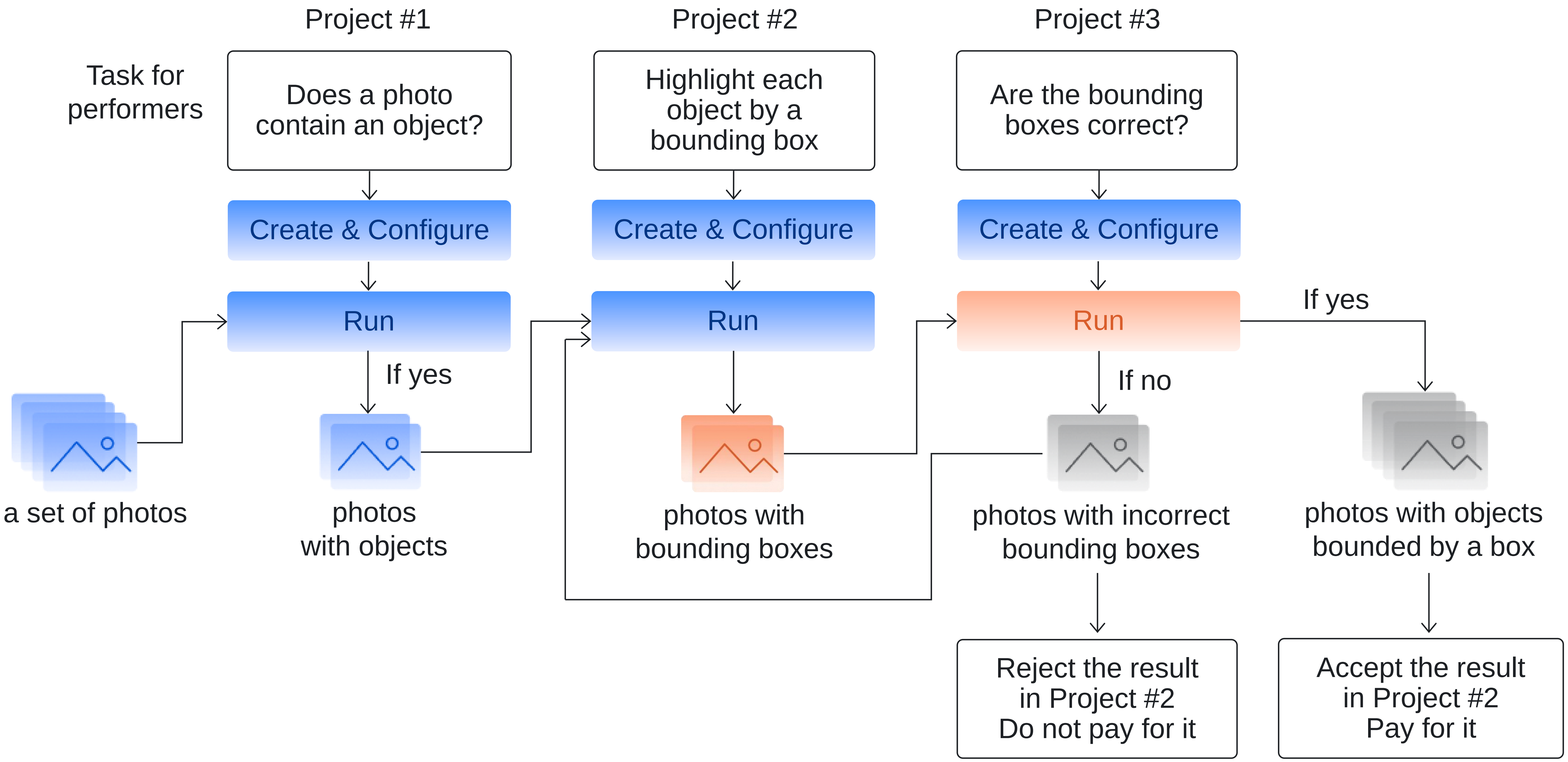

Project #1
Project #2
Project #3
Task for performers
Does a photo contain an object?
Highlight each object by a bounding box
Are the bounding boxes correct?
Create & Configure
Create & Configure
Create & Configure
Run
Run
Run
If yes
If yes
If no
a set of photos
photos with objects
photos with bounding boxes
photos with incorrect bounding boxes
photos with objects bounded by a box
Reject the result in Project #2 Do not pay for it
Accept the result in Project #2 Pay for it

Step #5

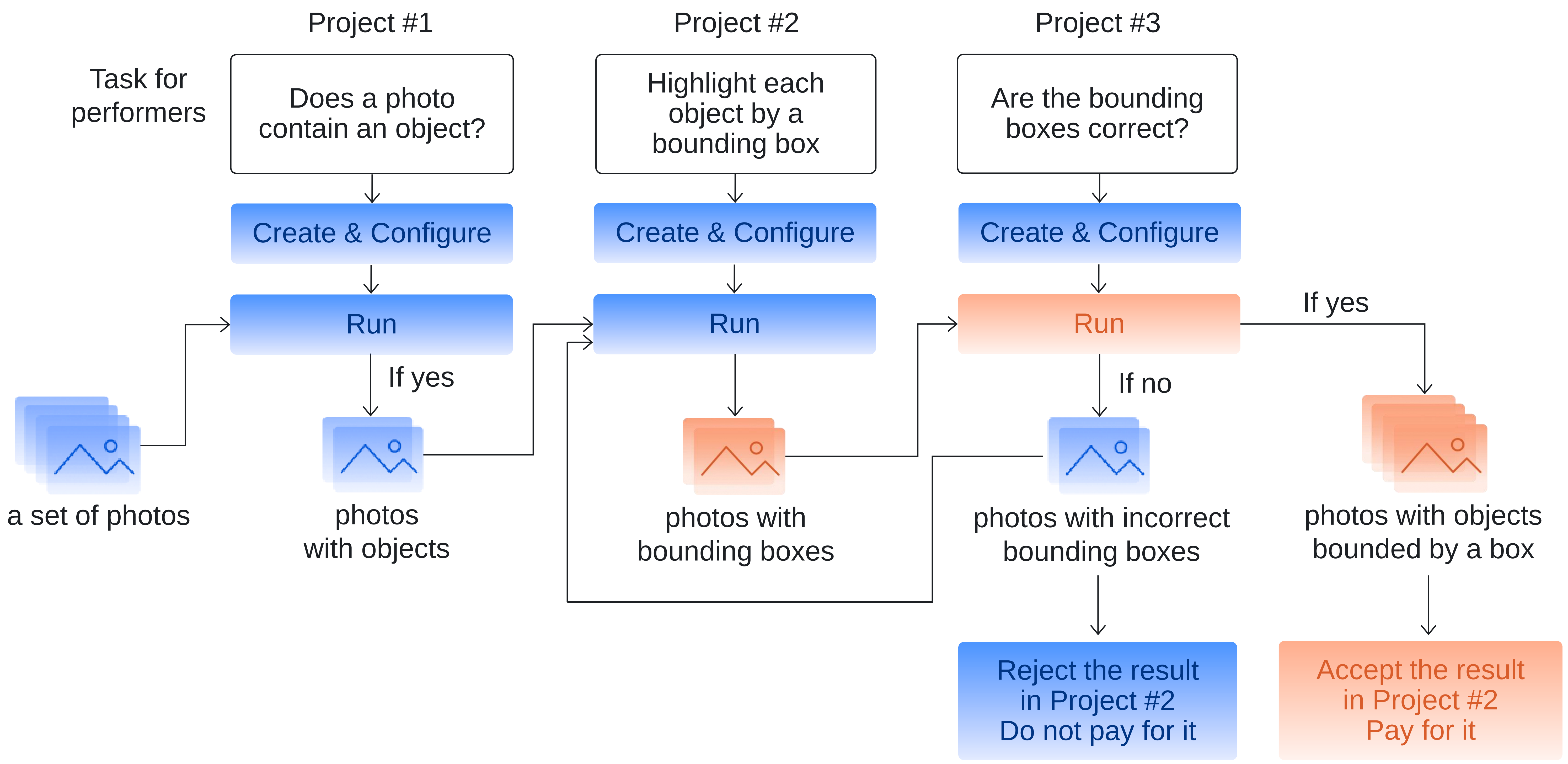


Project #1
Project #2
Project #3
Task for performers
Does a photo contain an object?
Highlight each object by a bounding box
Are the bounding boxes correct?
Create & Configure
Create & Configure
Create & Configure
Run
Run
Run
If yes
If yes
If no
a set of photos
photos with objects
photos with bounding boxes
photos with incorrect bounding boxes
photos with objects bounded by a box
Reject the result in Project #2 Do not pay for it
Accept the result in Project #2 Pay for it

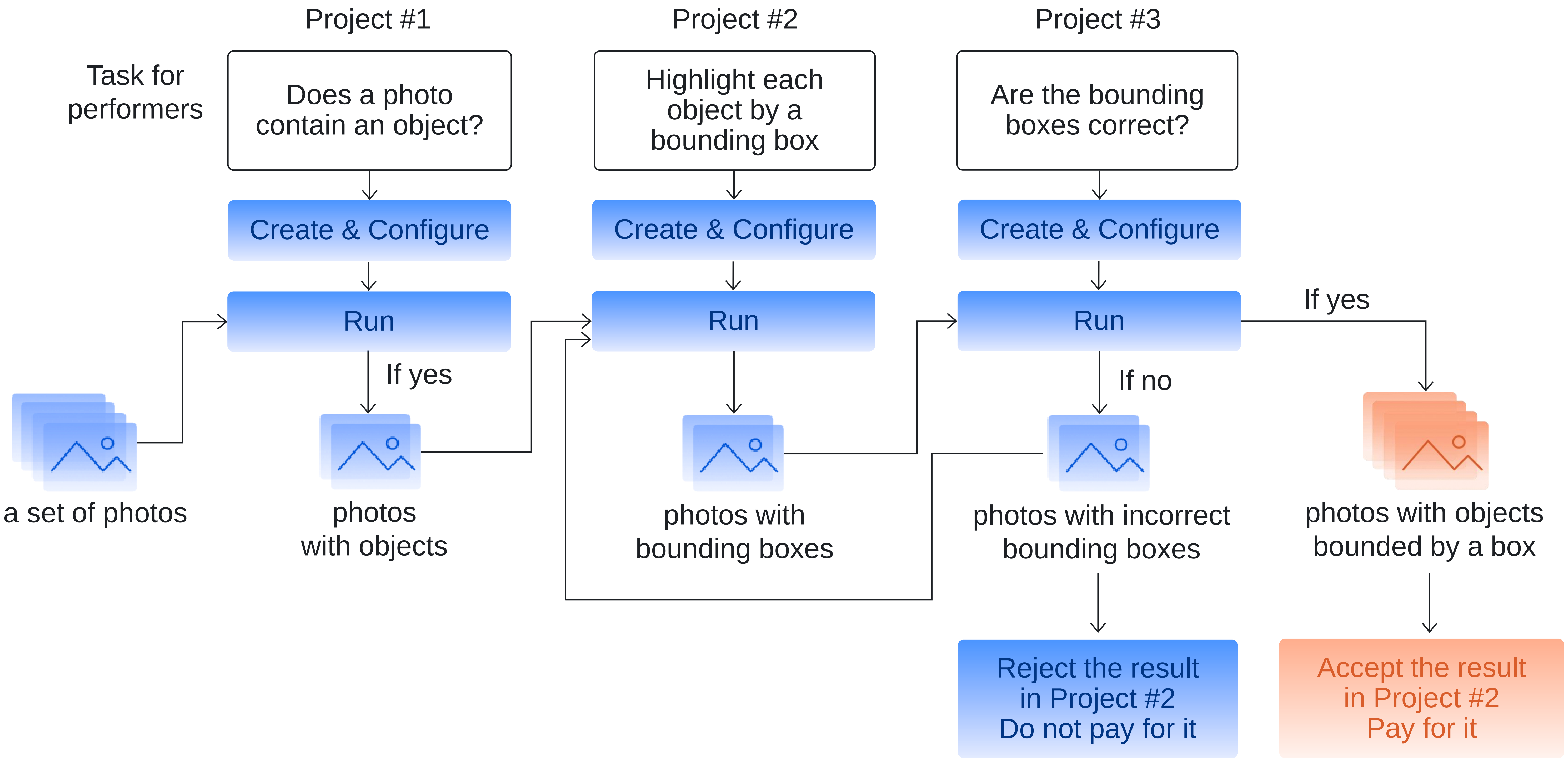
Step #6


Project #1
Project #2
Project #3
Task for performers
Does a photo contain an object?
Highlight each object by a bounding box
Are the bounding boxes correct?
Create & Configure
Create & Configure
Create & Configure
Run
Run
Run
If yes
If yes
If no
a set of photos
photos with objects
photos with bounding boxes
photos with incorrect bounding boxes
photos with objects bounded by a box
Reject the result in Project #2 Do not pay for it
Accept the result in Project #2 Pay for it

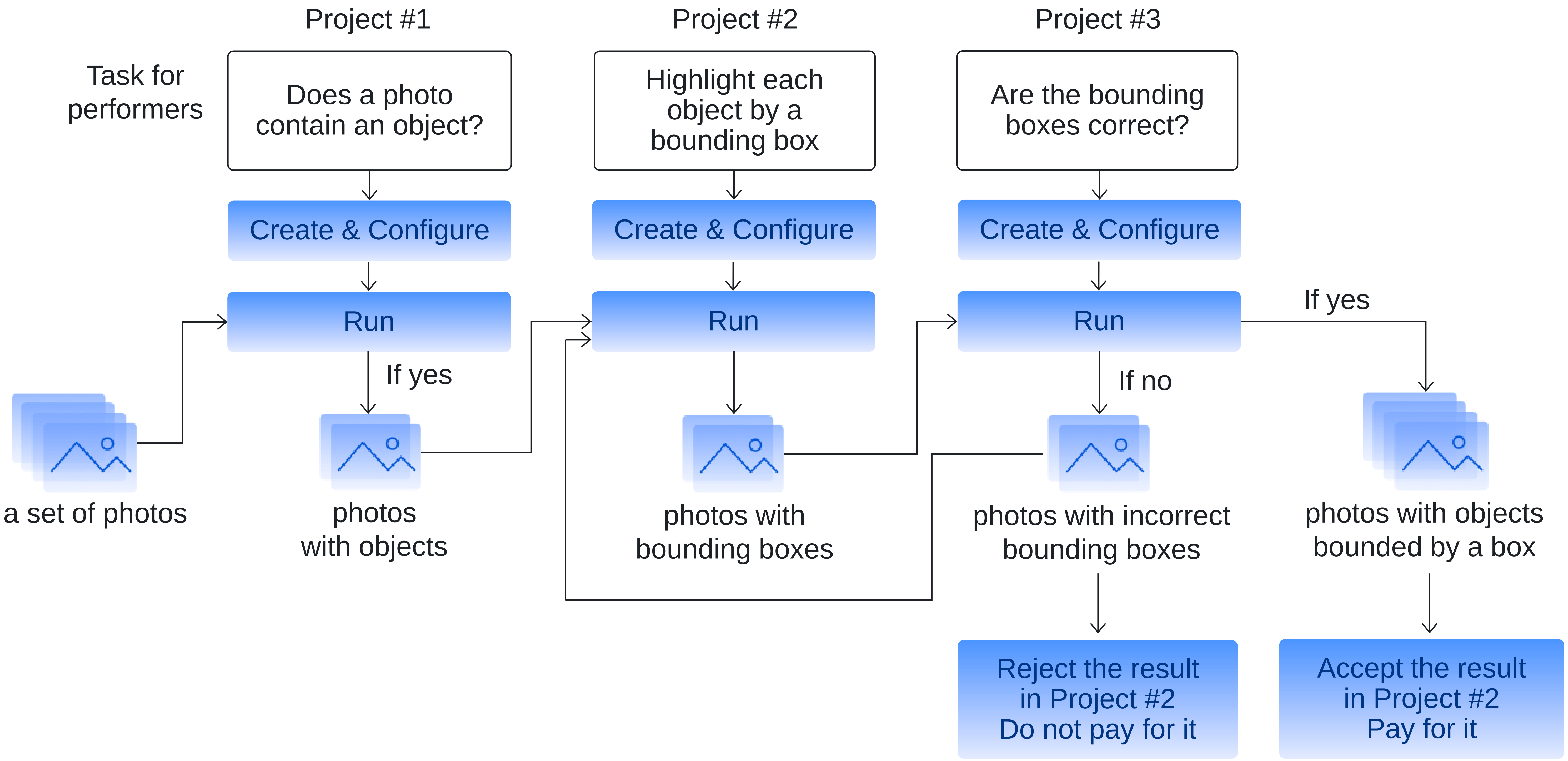
Step #7


Project #1
Project #2
Project #3
Task for performers
Does a photo contain an object?
Highlight each object by a bounding box
Are the bounding boxes correct?
Create & Configure
Create & Configure
Create & Configure
Run
Run
Run
If yes
If yes
If no
a set of photos
photos with objects
photos with bounding boxes
photos with incorrect bounding boxes
photos with objects bounded by a box
Reject the result in Project #2
Do not pay for it
Accept the result in Project #2
Pay for it

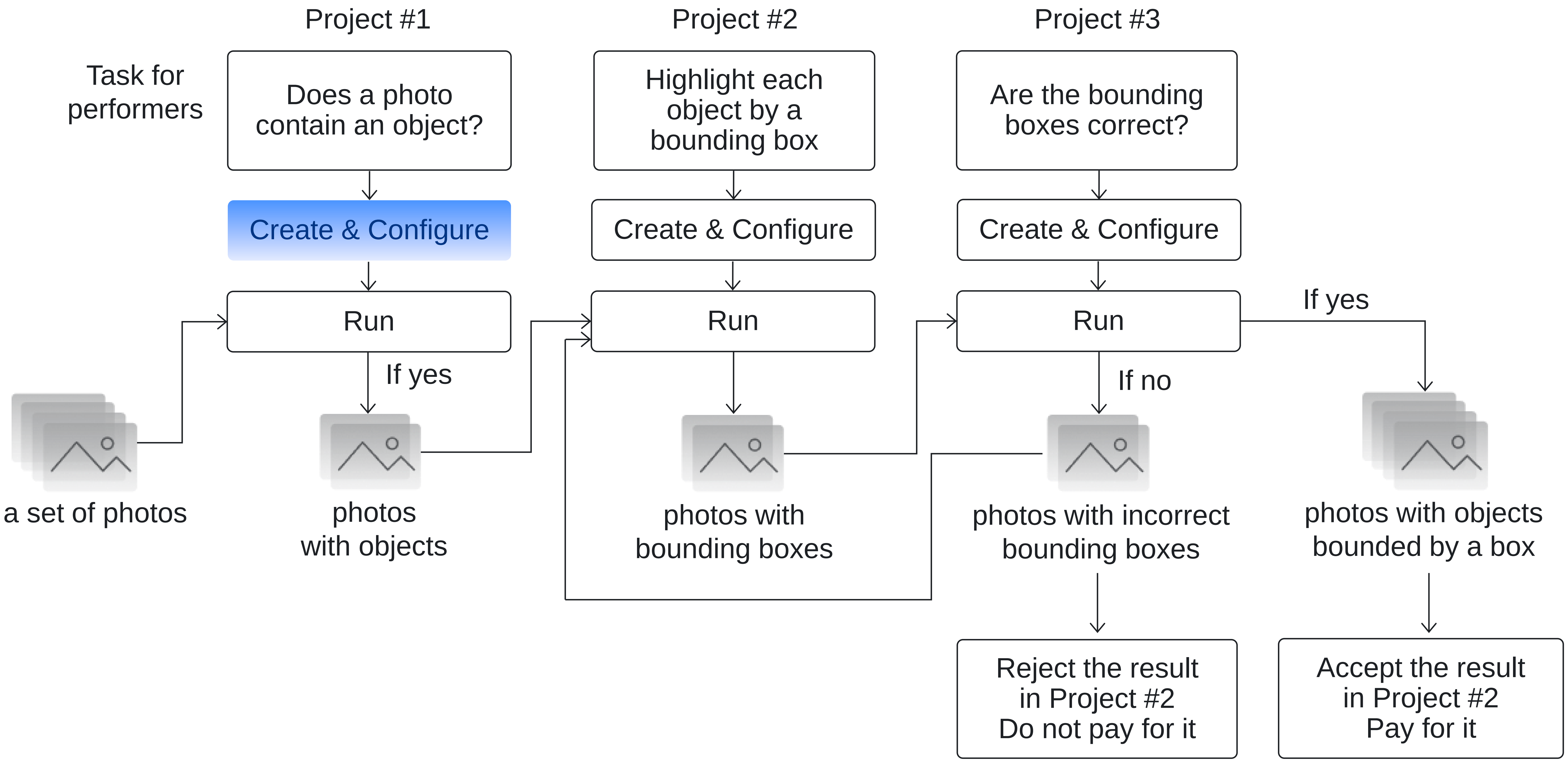
Step #6 (repeat until none rejected in Project #2)



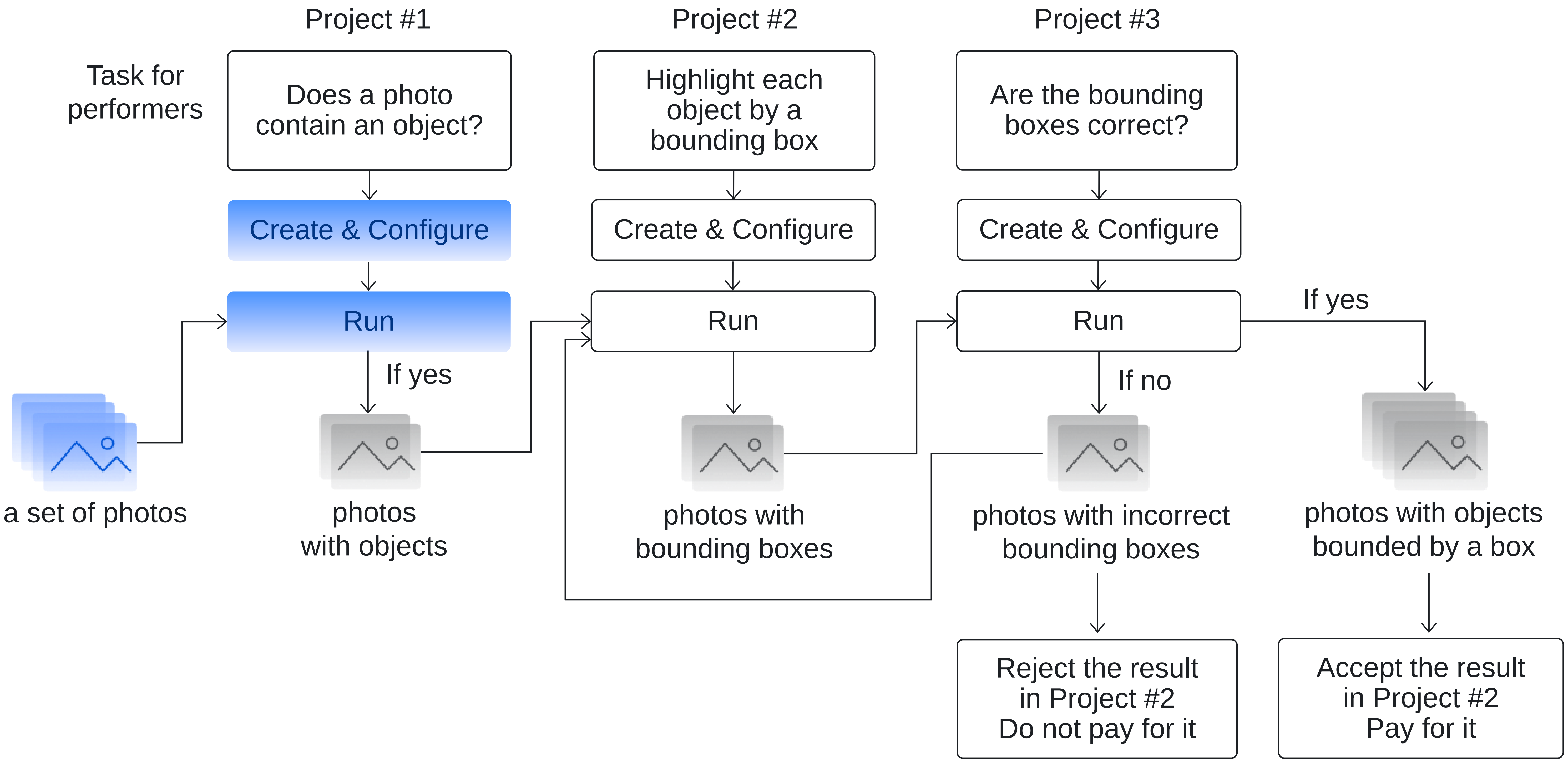
Step #7 (repeat until none rejected in Project #2)



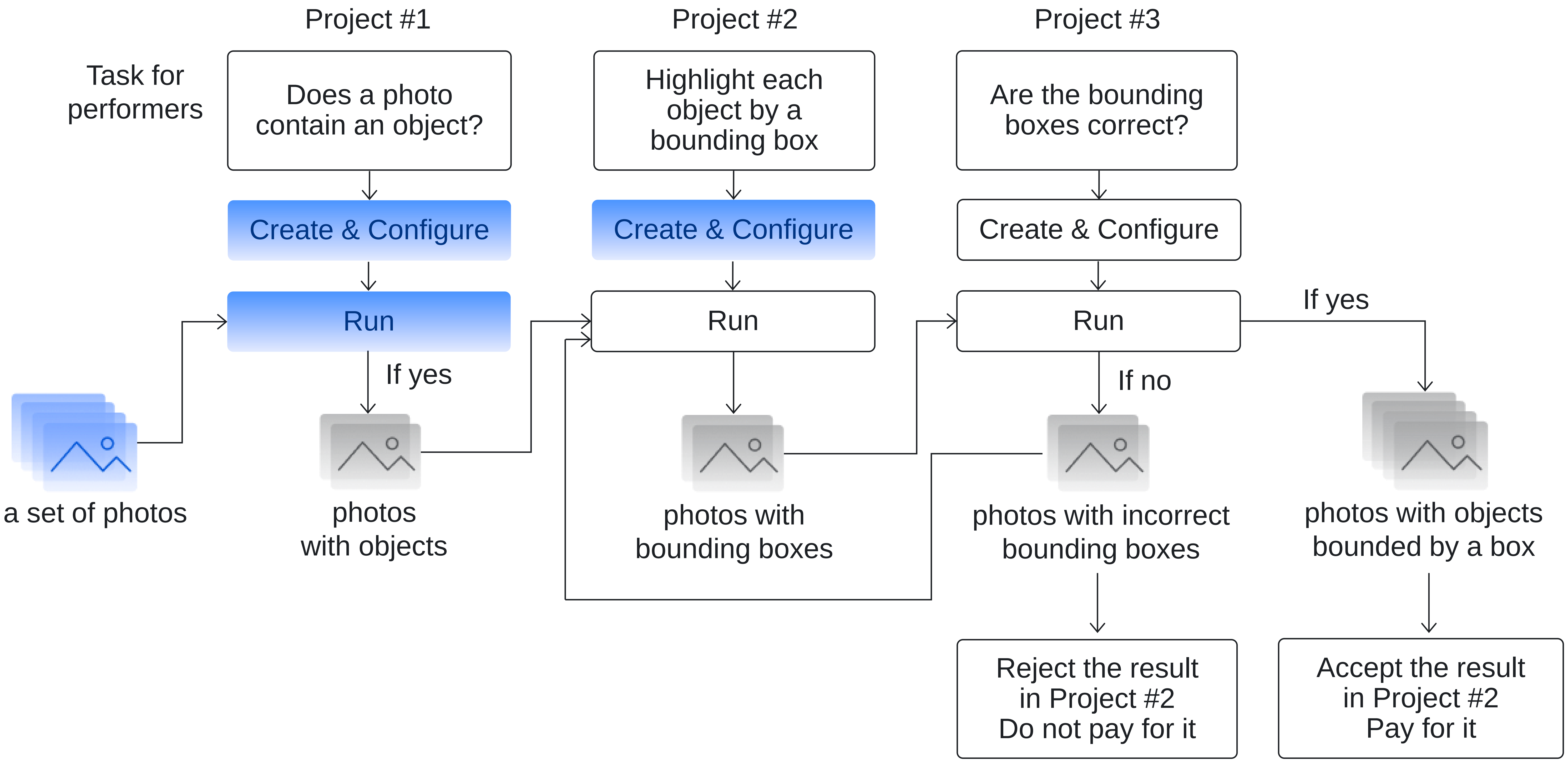
Most of us are at this step



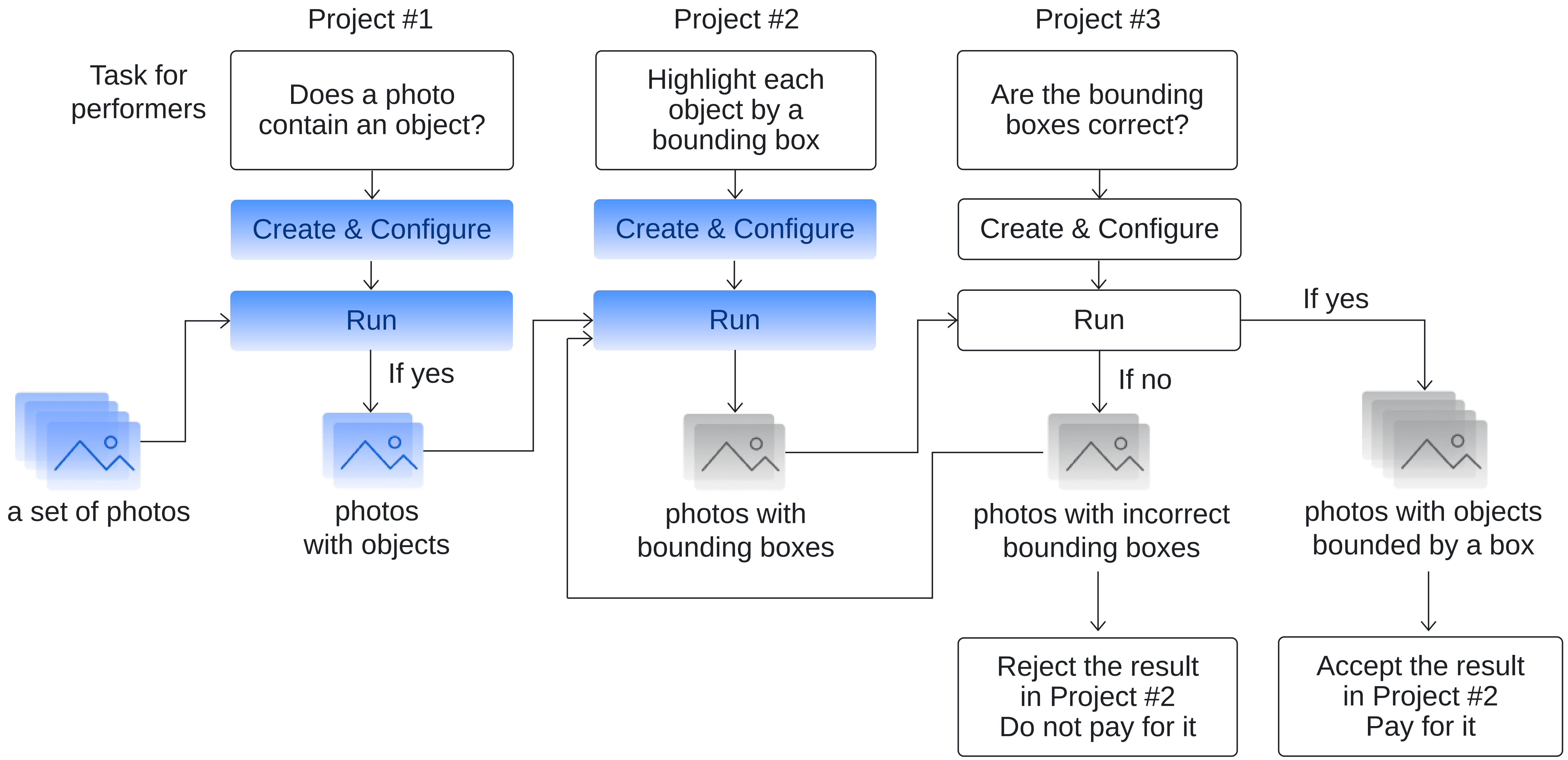
Most of us are at this step



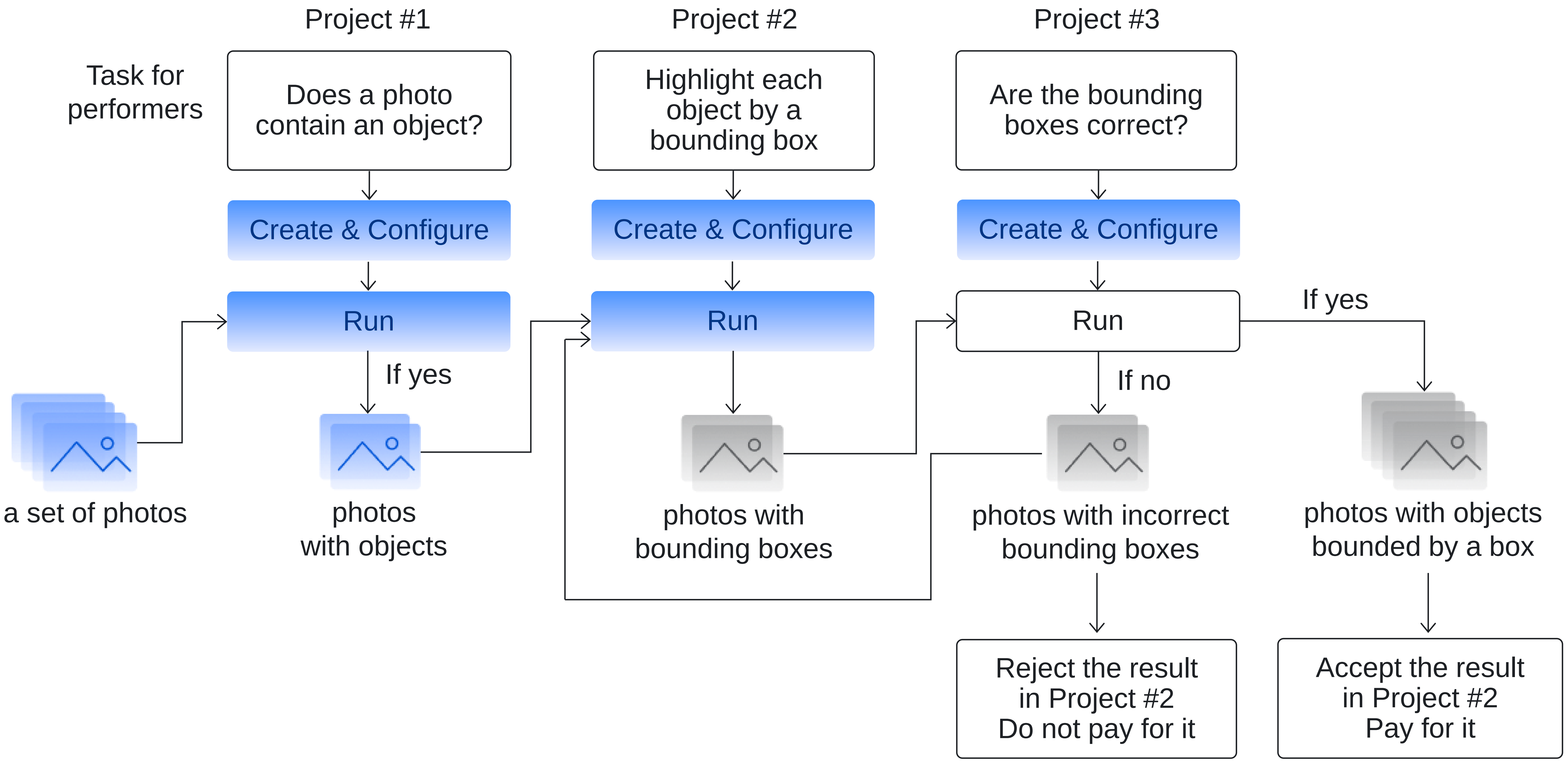
Most of us are at this step



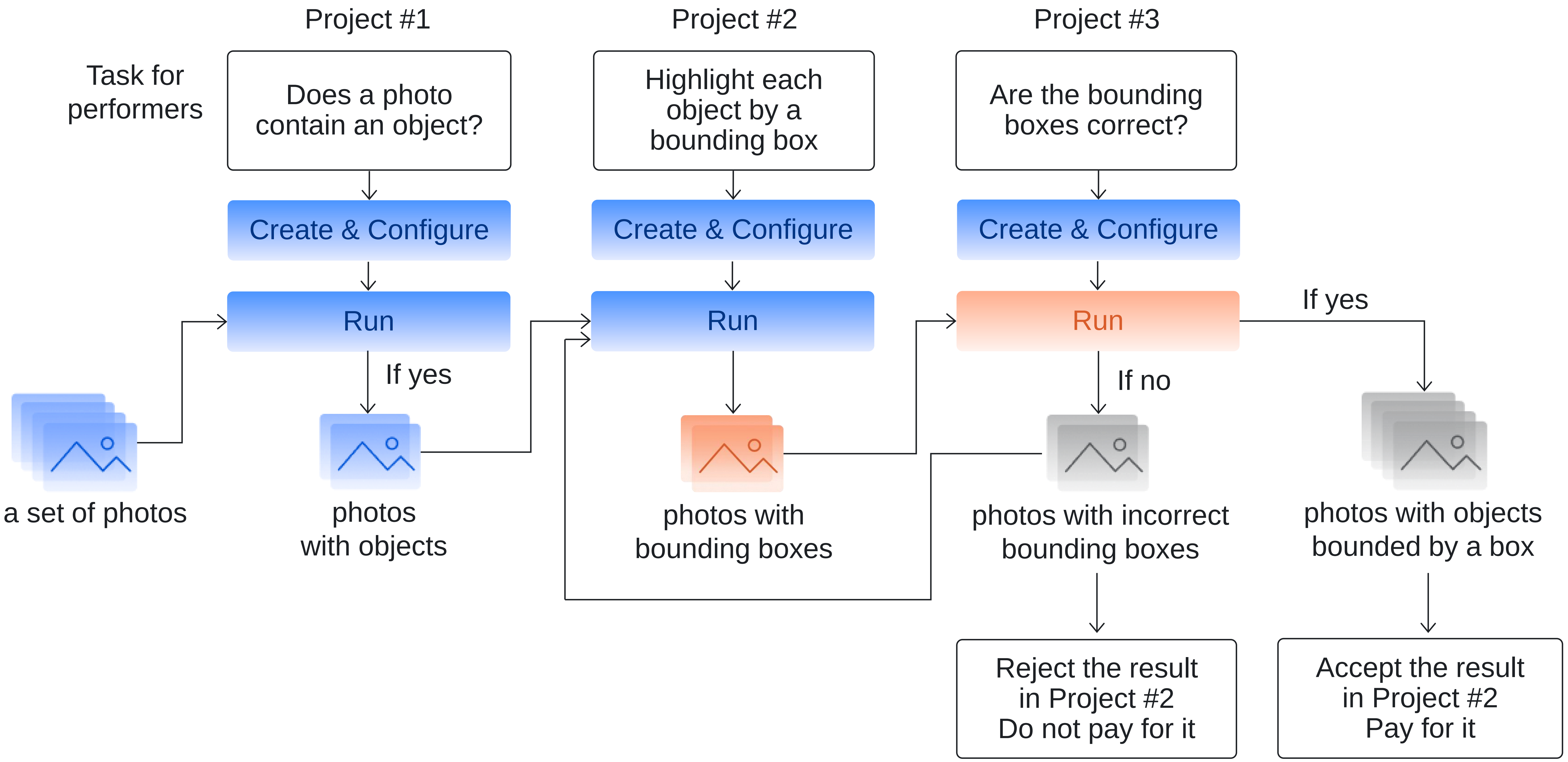
Most of us are at this step



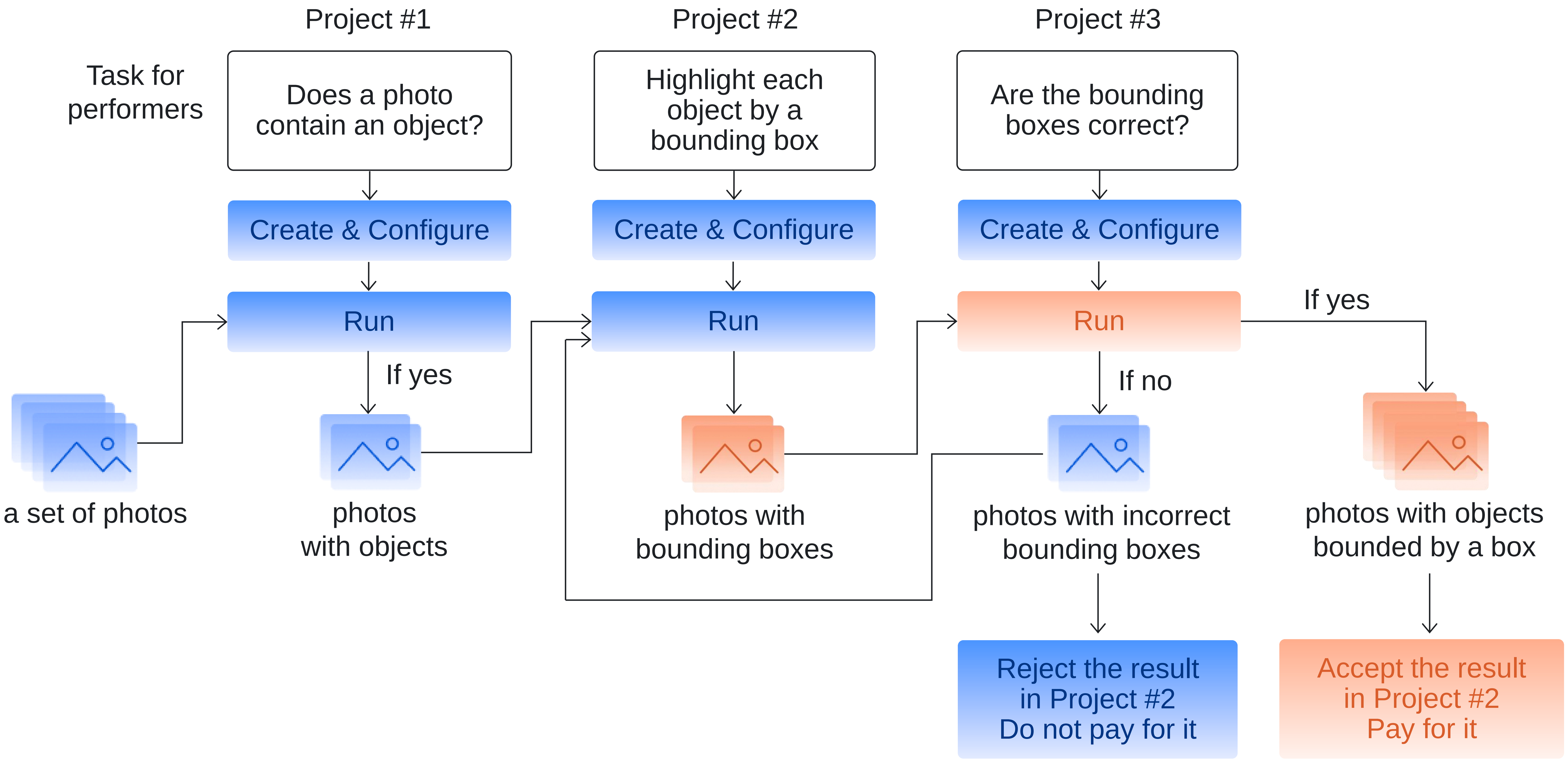
Most of us are at this step



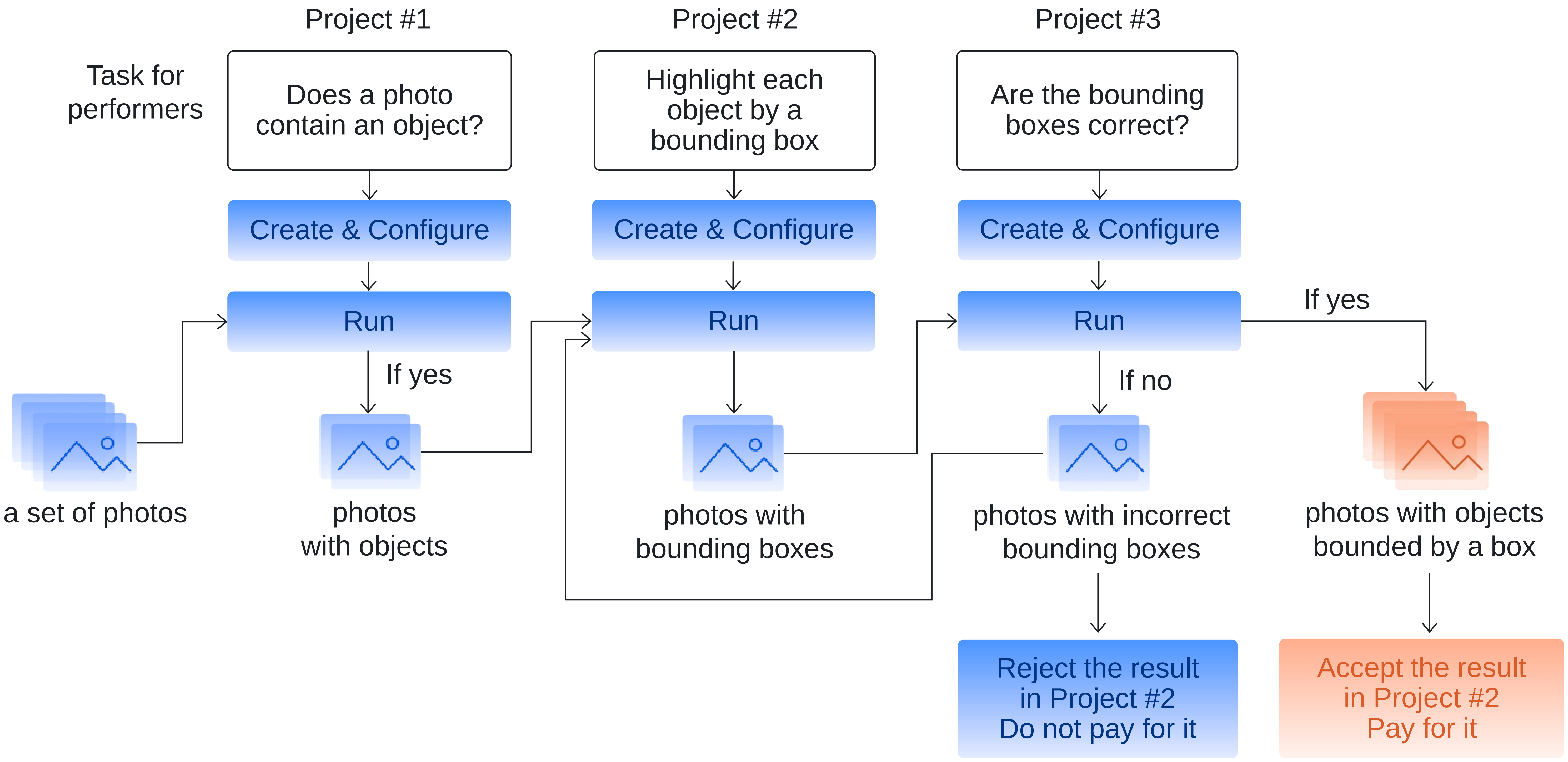
Most of us are at this step



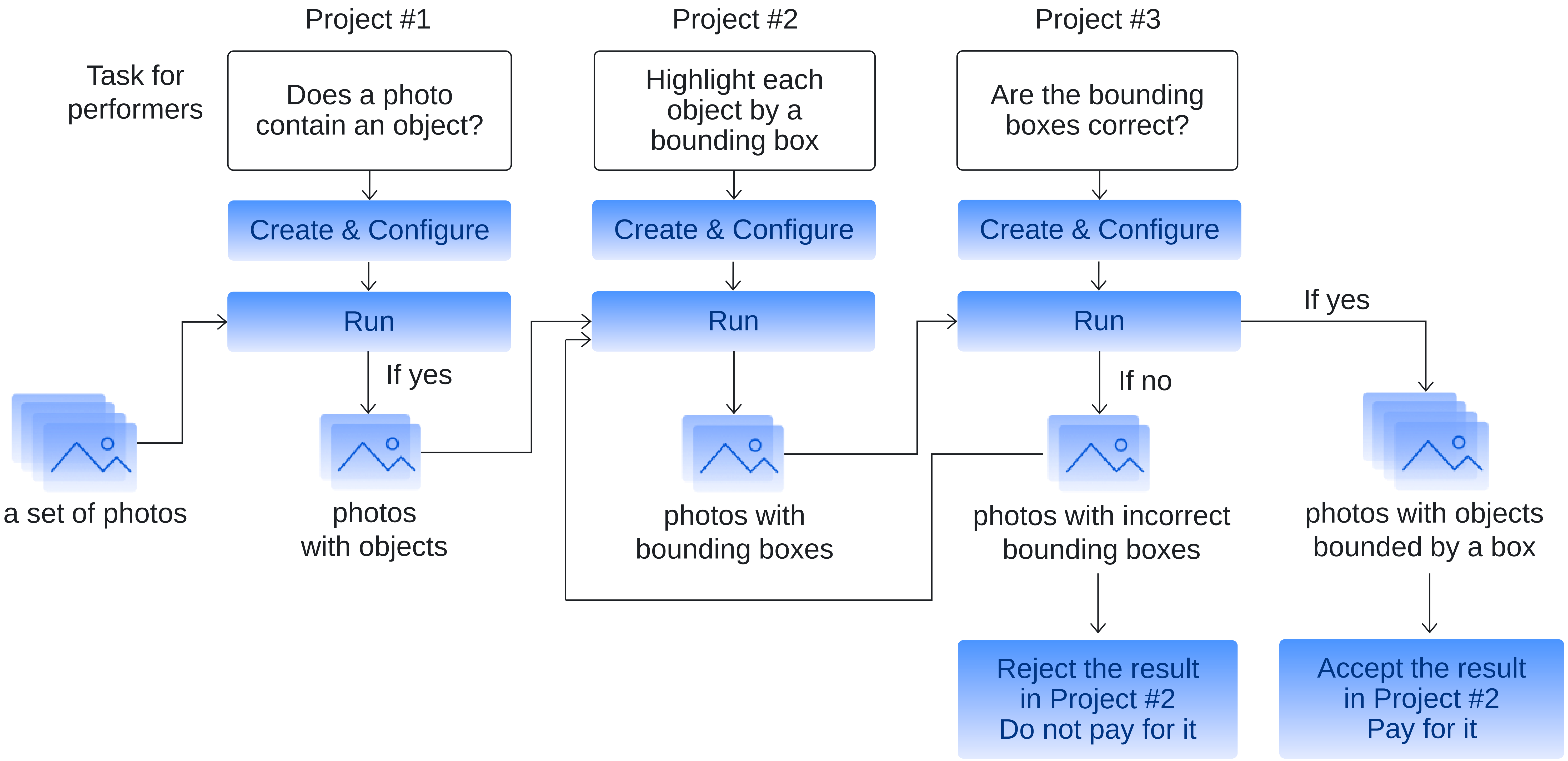
Most of us are at this step



Most of us are at this step



Most of us are at this step



Part V

Theory on efficient aggregation, incremental relabeling, and pricing

Valentina Fedorova,
Researcher

Project 1: Filter images

Does the image contain traffic signs?

Yes

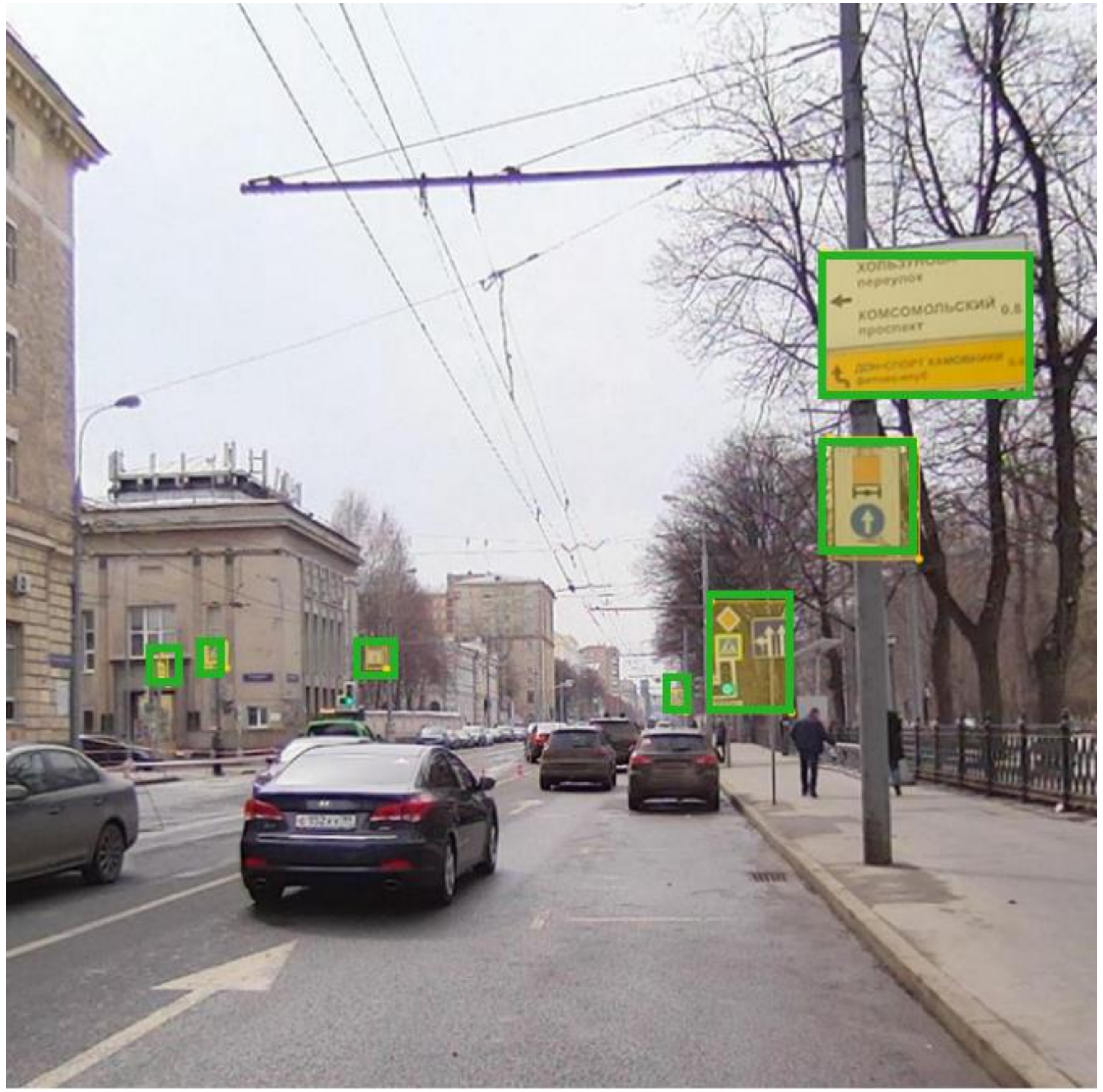
No

Project 3: Verification

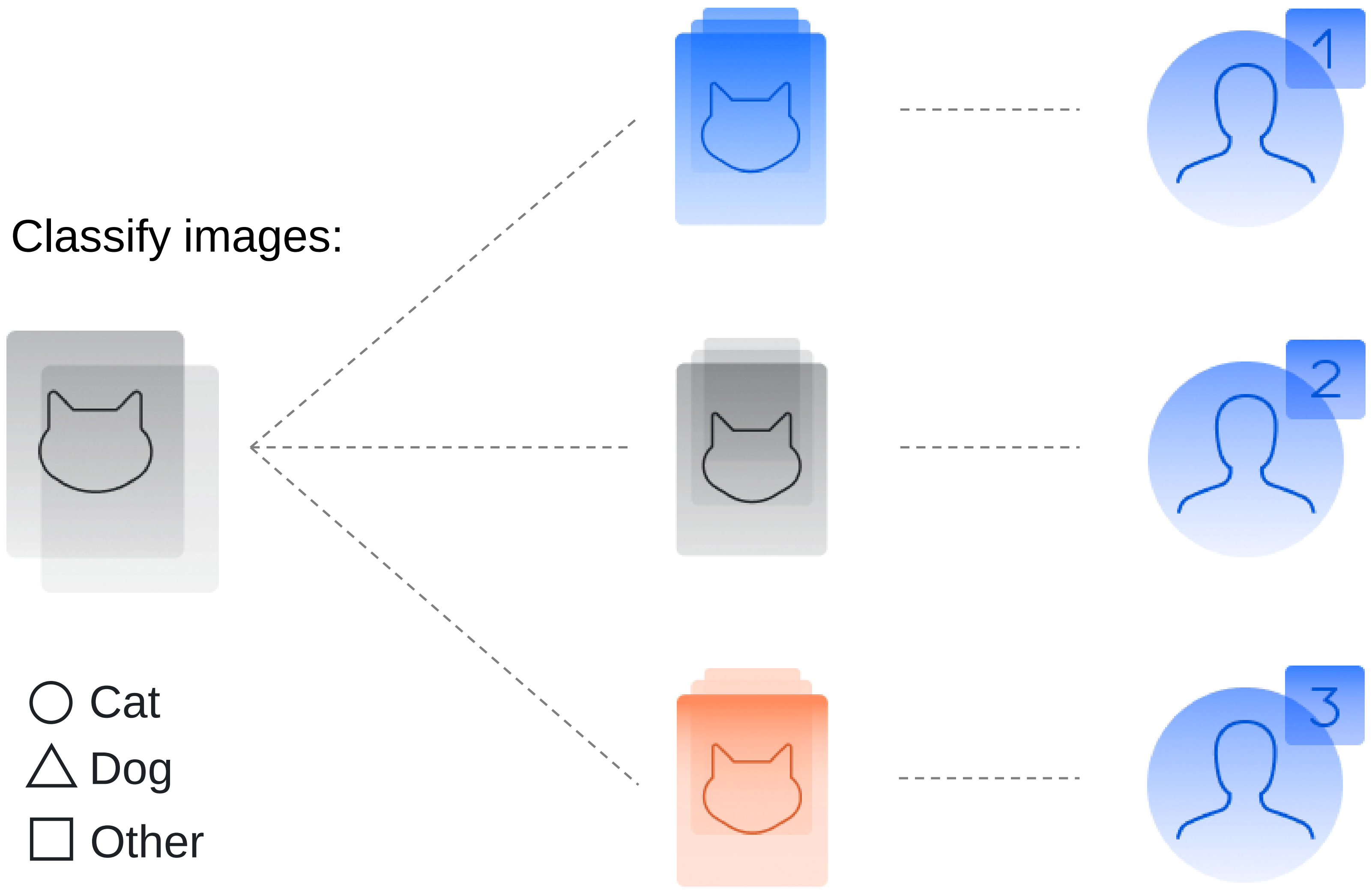
Are the bounding boxes correct?

Yes

No



Labeling data with crowdsourcing



- How to choose a reliable label?
- How many workers per object?
- How much to pay to workers?
- …

Evaluation of labeling approaches

vs

Accuracy

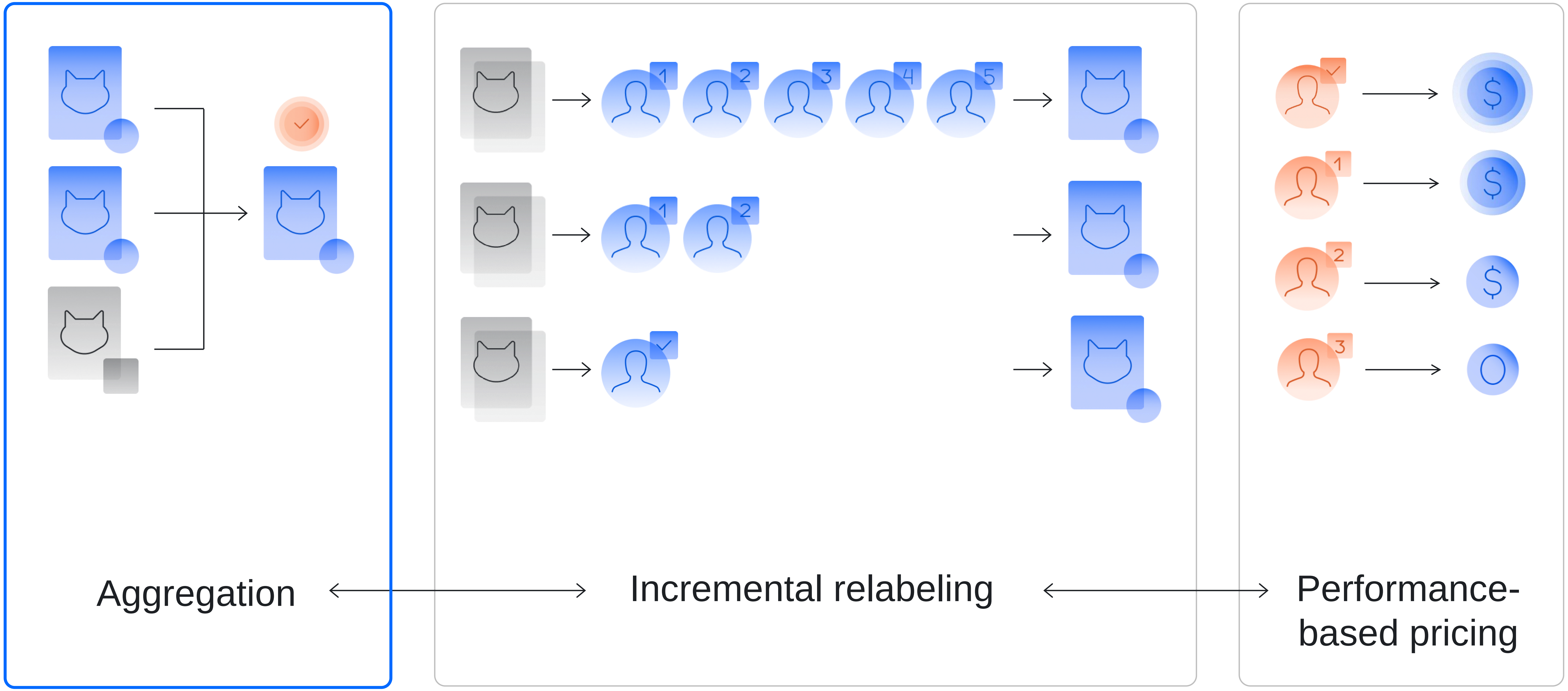
Cost

- Labels with a maximal level of accuracy for a **given budget**

  or

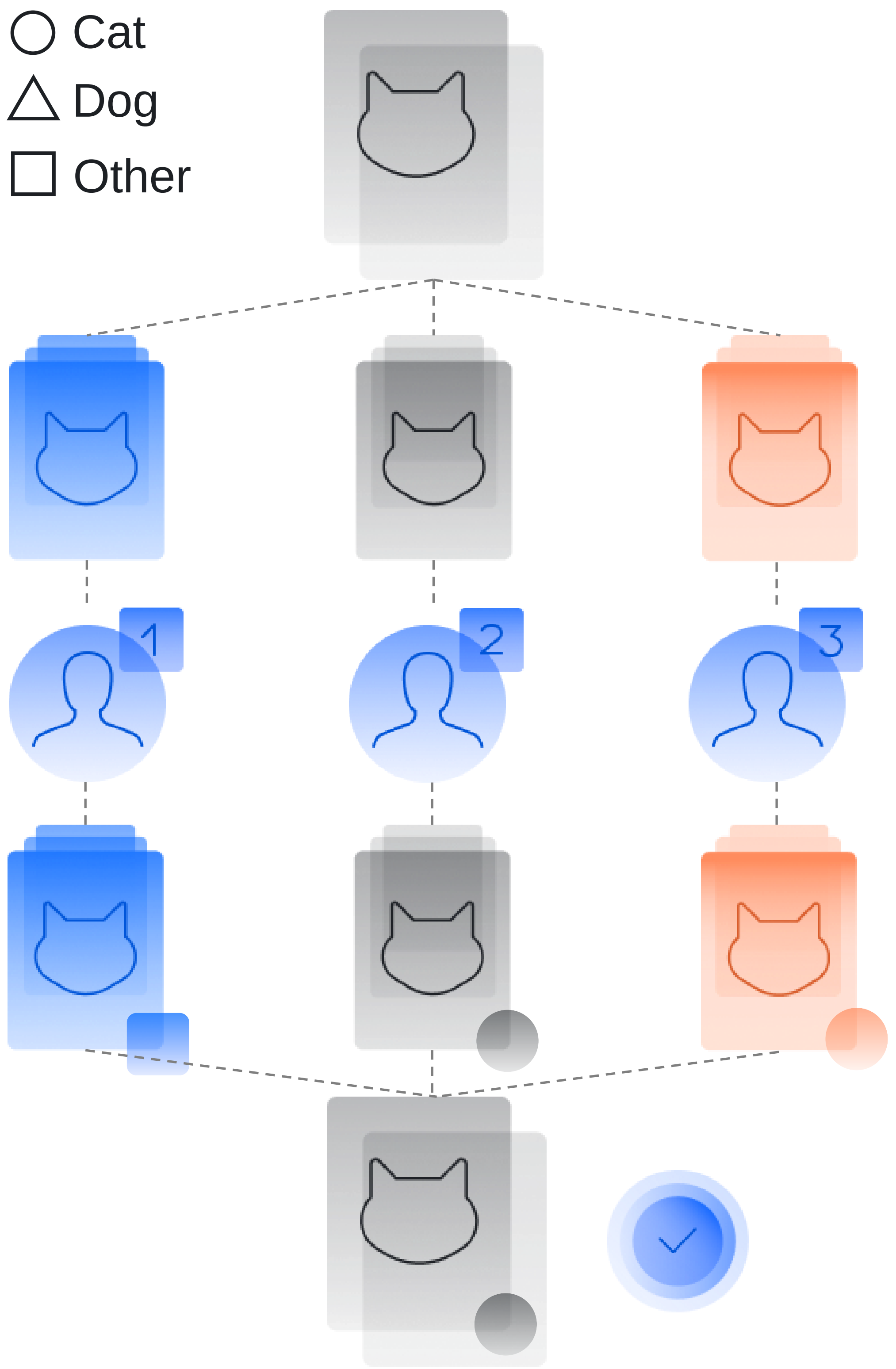
- Labels of a chosen accuracy level for a **minimal budget**

Key components of labeling with crowds



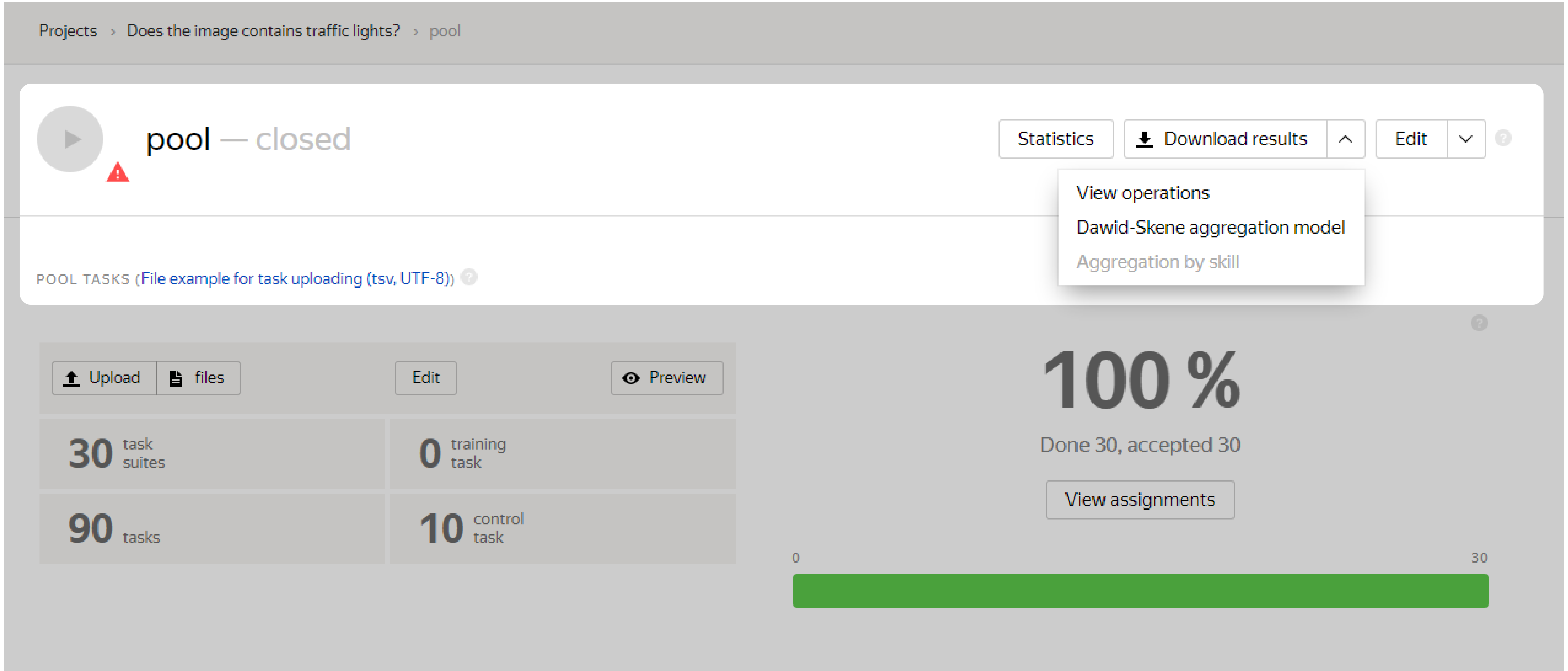
Aggregation

Labeling data with crowds



- Classify images
- Upload multiple copies of each object to label
- Workers assign noisy labels to objects
- Aggregate multiple labels for each object into a more reliable one

Process results

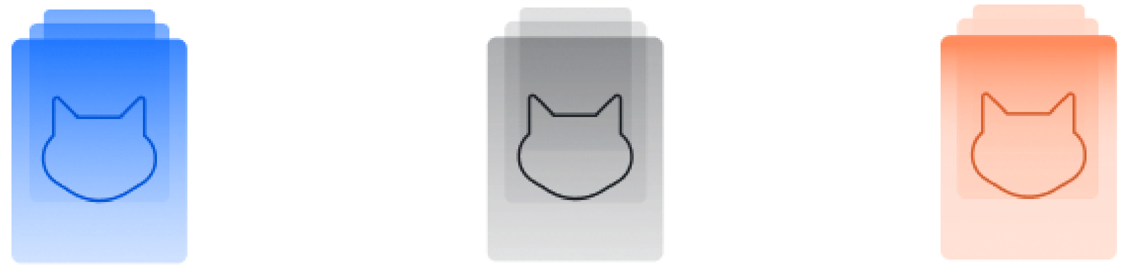


Notation

- Categories $k \in \{1,\ldots,K\}$. E.g.:

○ Cat   △ Dog   ☐ Other

- Objects $j \in \{1,\ldots,J\}$. E.g.:

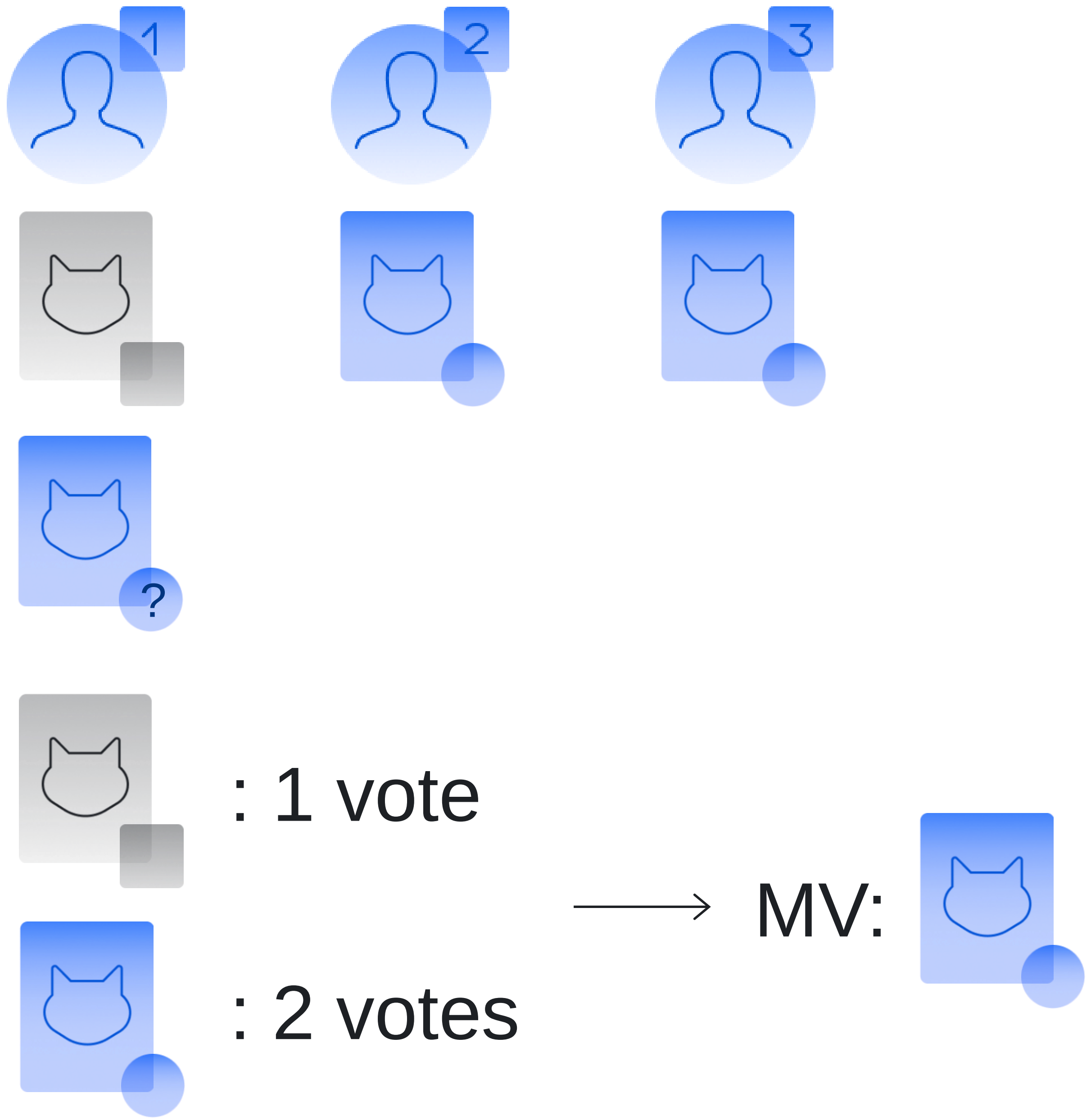


- Workers: $w \in \{1,\ldots,W\}$. E.g.:

  - $W\_j \subseteq \{1,\ldots,W\}$ — workers labeled object j

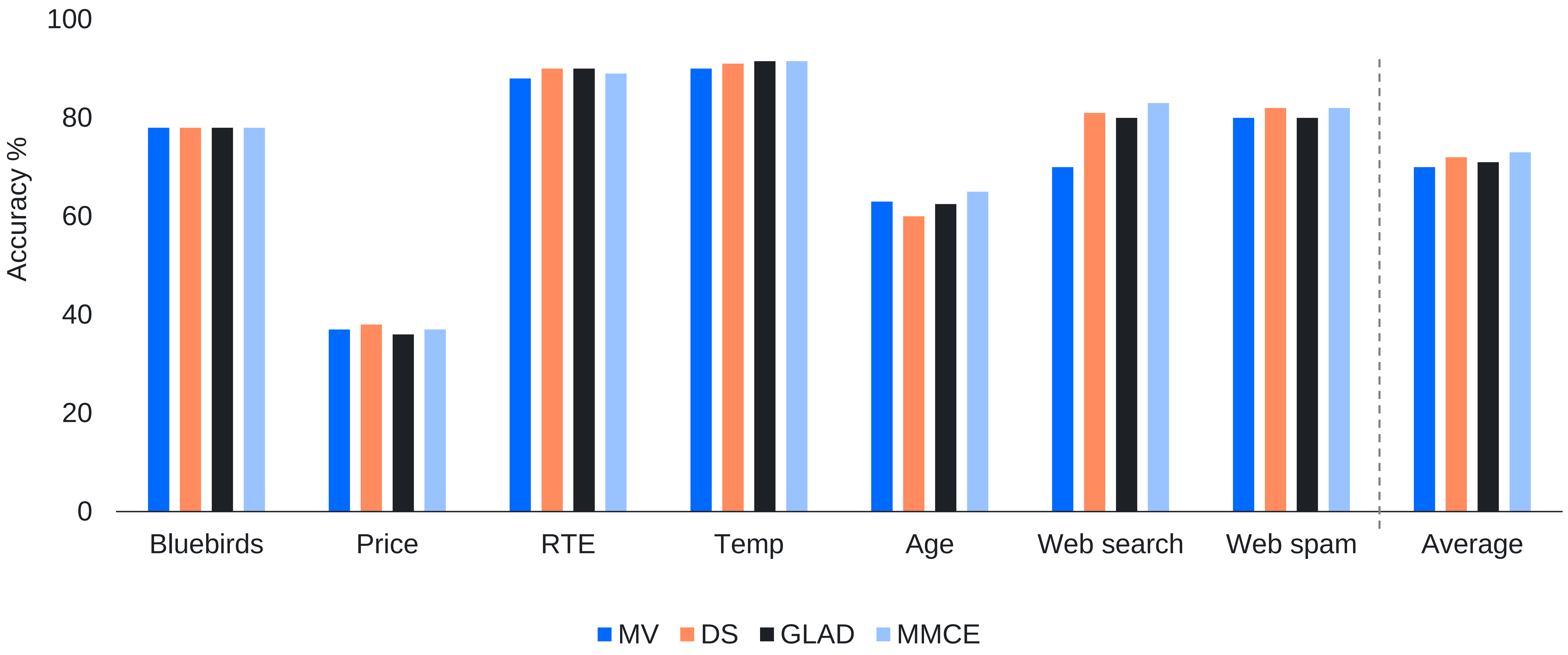
The simplest aggregation: Majority Vote (MV)

- The problem of aggregation:
  - Observe noisy labels
    $y = \{y_j^w \mid j = 1, \dots, J \text{ and } w = 1, \dots, W\}$
  - Recover true labels $z = \{z_j \mid j = 1, \dots, J\}$
- A straightforward solution:



$\hat{z}_j^{MV} = \arg\max_{y=1,\dots,K} \sum_{w \in W_j} \delta(y = y_j^w)$, where $\delta(A) = 1$ if A is true and 0 otherwise

Performance of MV vs other methods

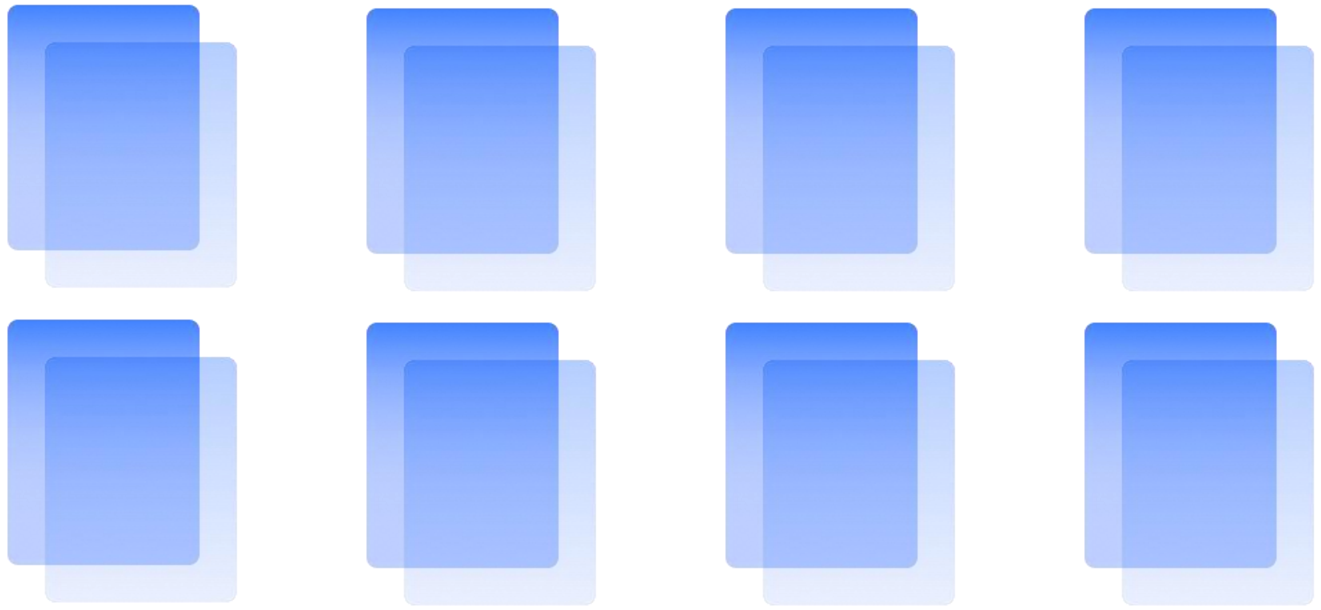


Zhou D. et al. Regularized minimax conditional entropy for crowdsourcing. 2015

Properties of MV

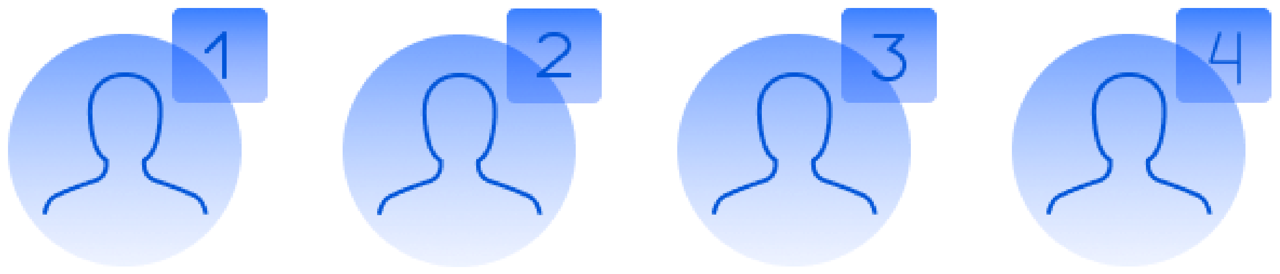
All workers are treated similarly

All objects are treated similarly



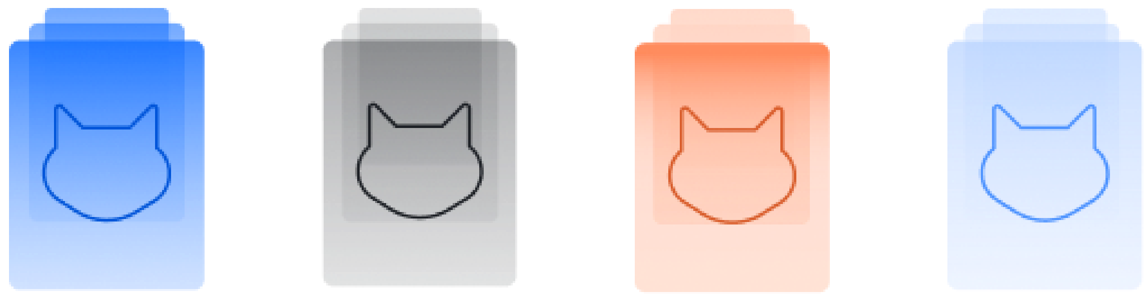
Advanced aggregation: workers and objects

Parameterize expertise of workers by e^{w}



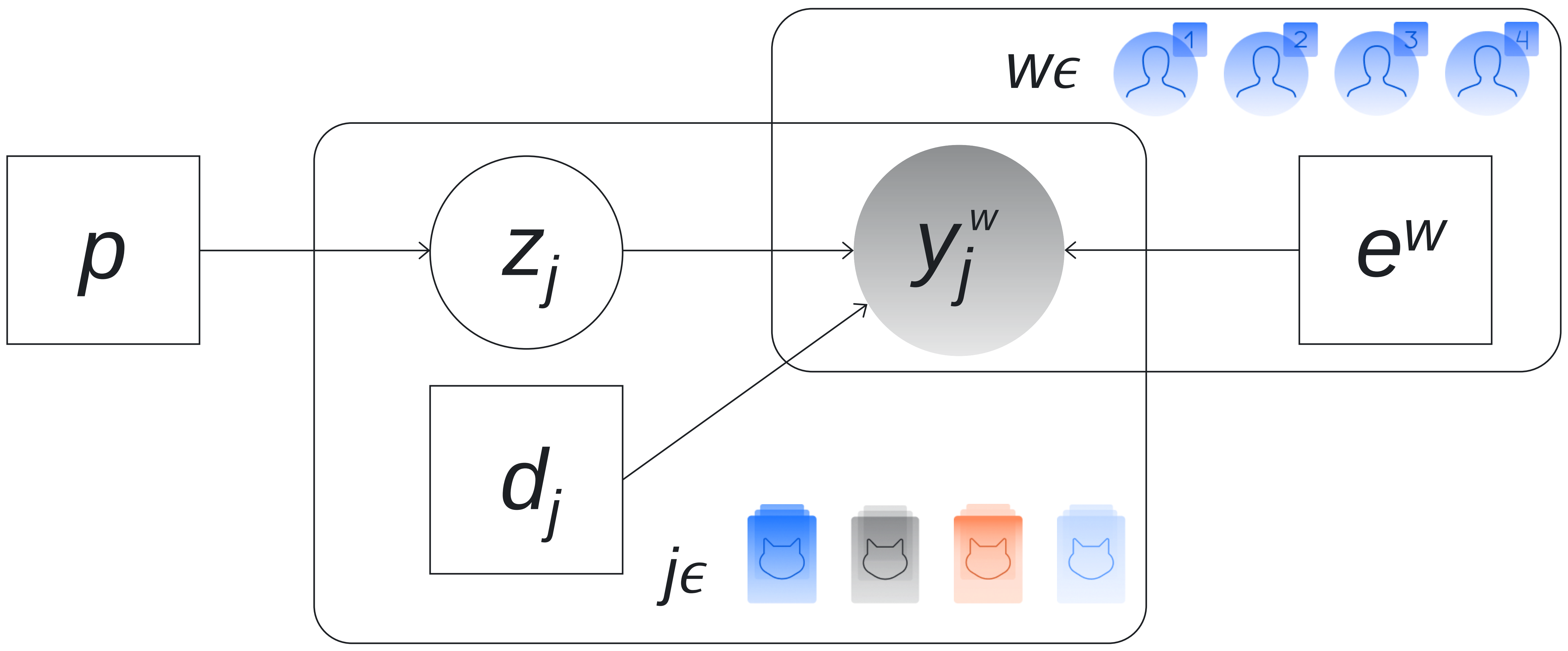
e^{w_1} e^{w_2} e^{w_3} e^{w_4}

Parameterize difficulty of objects by d_j

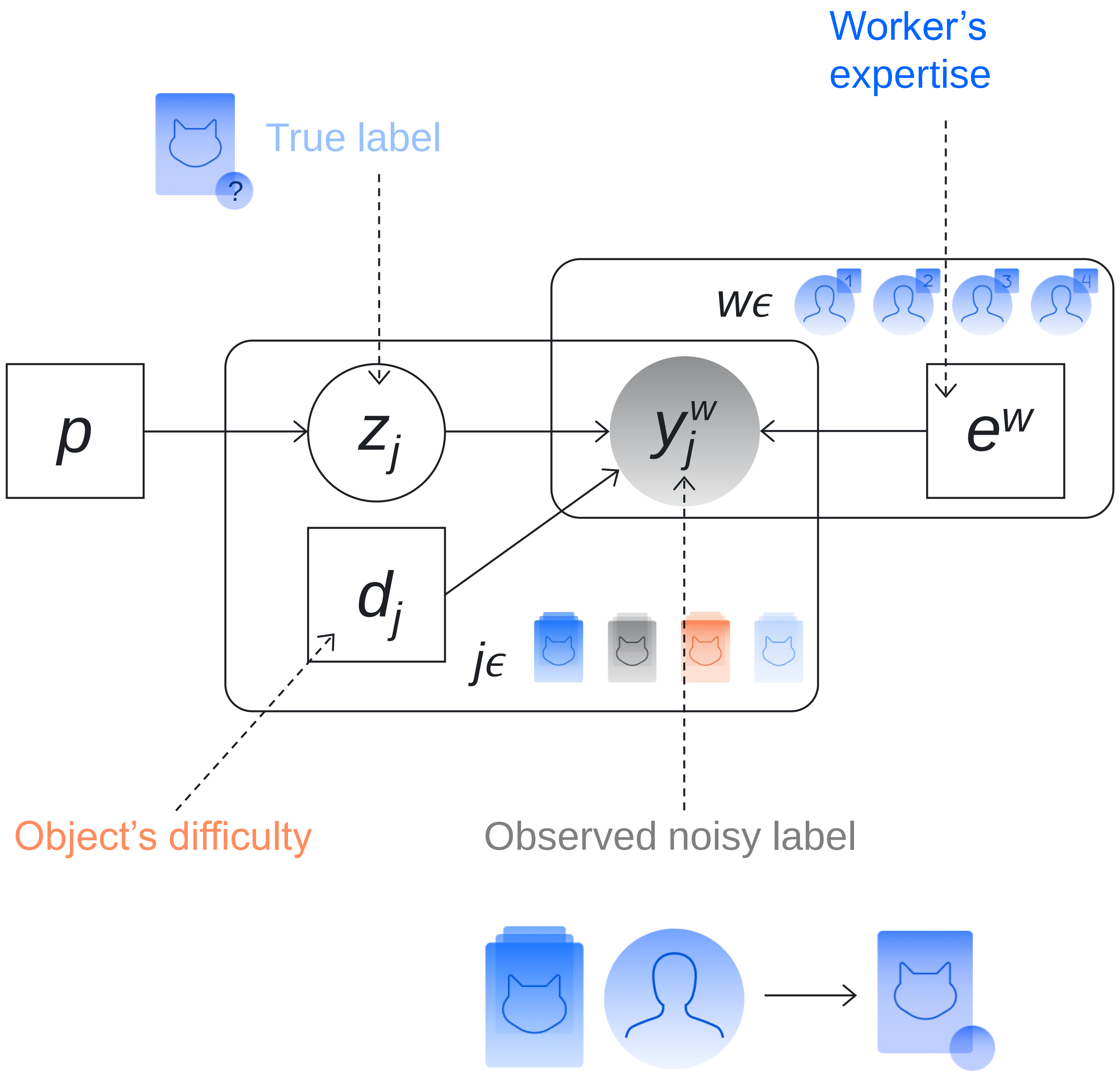


d_{j_1} d_{j_1} d_{j_1} d_{j_1}

Advanced aggregation: latent label models

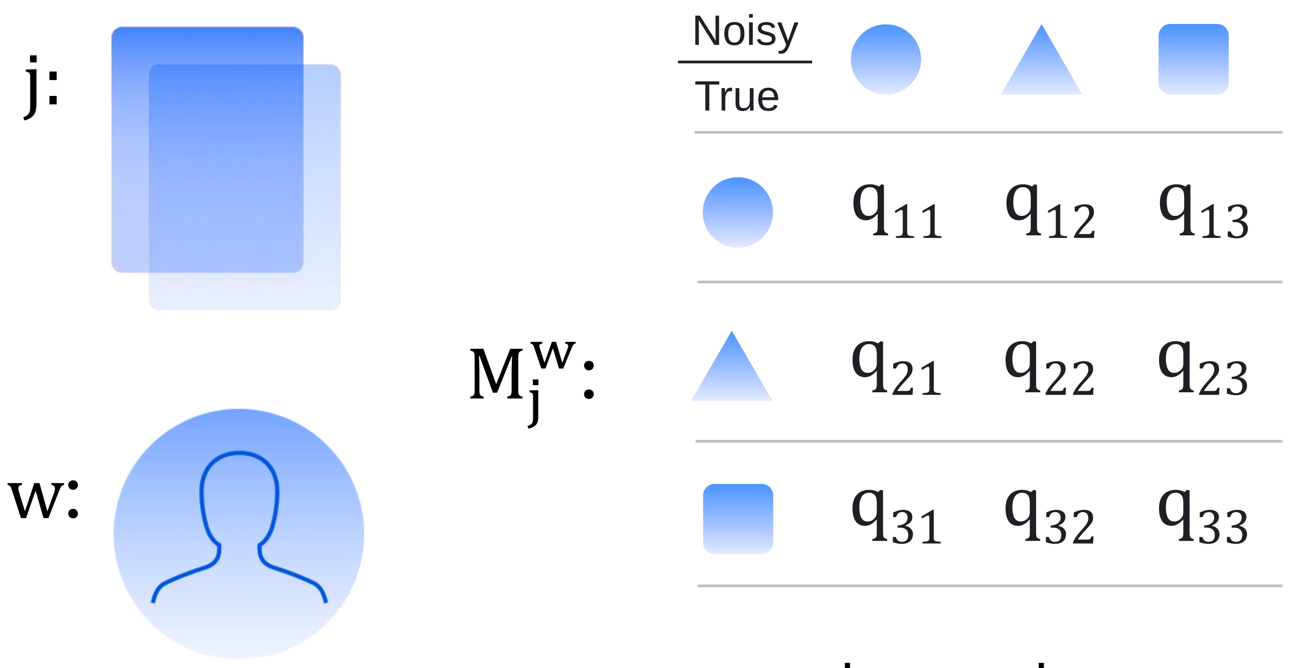


Latent label models: noisy label model



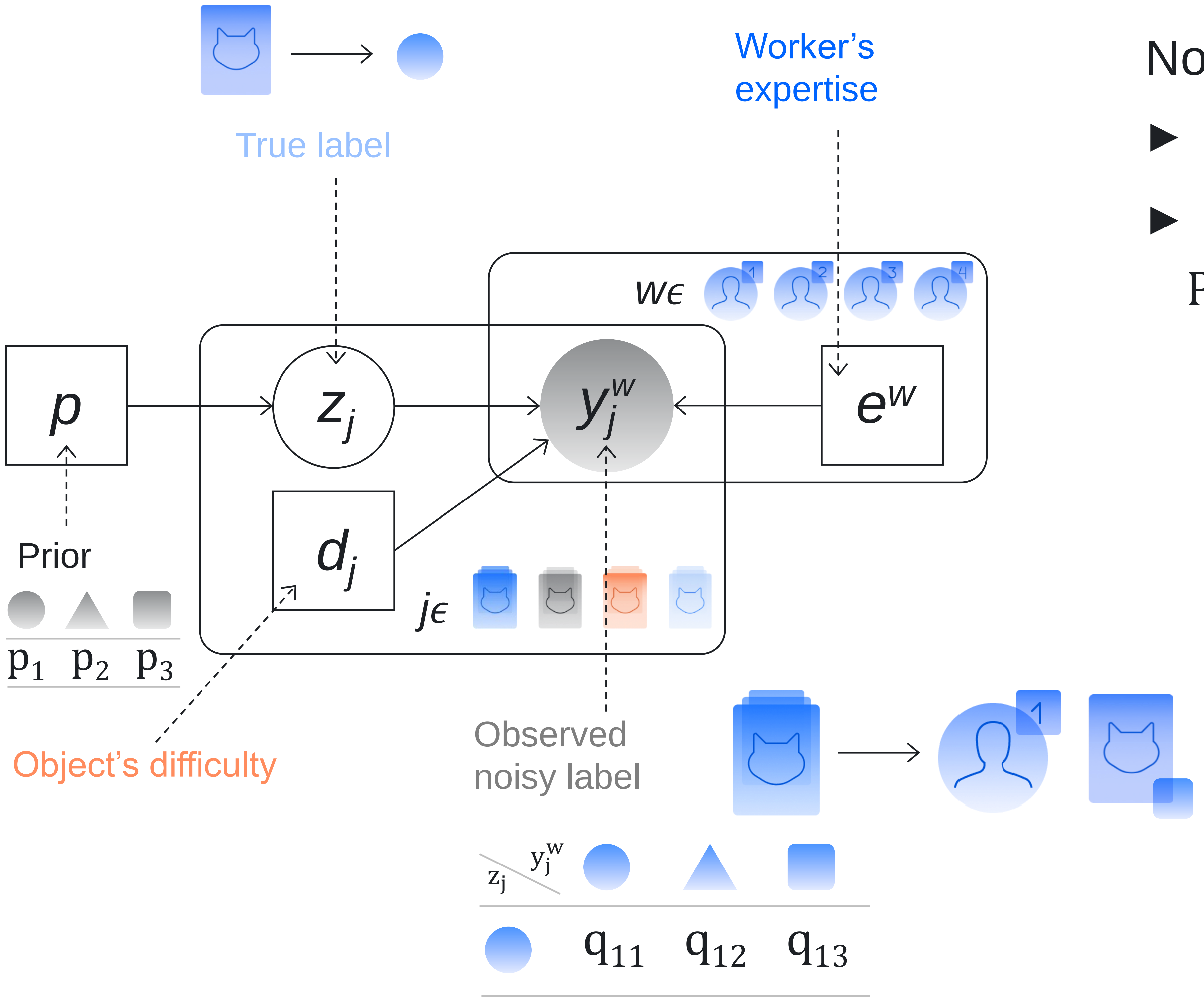
A noisy label model $M_j^w = M(e^w, d_j)$ is a matrix of size $K \times K$ with elements

$$M_j^w[c,k] = \Pr\left(Y_j^w = k \mid Z_j = c\right)$$



$q_{c1} + q_{c2} + q_{c3} = 1$
for each c

Latent label models: generative process



Noisy labels generation:

- Sample z_j from a distribution $P_Z(p)$
- Sample y_j^w from a distribution $P_Y(M_j^w[z_j,\cdot])$

In multiclassification, a standard choice for $P_Z(\cdot)$ and $P_Y(\cdot)$ is a Multinomial distribution Mult($\cdot$)

Latent label models: parameters optimization

- Assumption: y_j^w is cond. independent of everything else given z_j, d_j, e^w
- The likelihood of y and z under the latent label model:

Observed noisy label

$$L\Big(\underbrace{\{z_j\}_{j=1}^{J}}_{\text{Latent true label}}, \underbrace{p, \{d_j\}_{j=1}^{J}, \{e^w\}_{w=1}^{W}}_{\text{Latent parameters}}\Big) = \underbrace{\prod_{j \in J} \sum_{z_j \in \{1,\dots,K\}} \Pr(z_j | p) \prod_{w \in W_j} \Pr(y_j^w | z_j, d_j, e^w)}_{\text{Likelihood of noisy and true labels for object } j}$$

- Estimate parameters and true labels by maximizing $L(\dots)$

Latent label models: EM algorithm

- Maximization of the expectation of log-likelihood (LL)*

$$\mathbb{E}_{z} \log \Pr(y, z) = \sum_{j \in J} \sum_{z_j \in \{1,\ldots,K\}} \Pr(z_j | p) \log \prod_{w \in W_j} \Pr(z_j | p) \Pr(y_j^w | z_j, d_j, e^w)$$

- **E-step:** Use Bayes' theorem for posterior distribution of $\hat{z}$ given p, d, e:

$$\hat{z}_j[c] = \Pr(Z_j = c | y, p, d, e) \propto \Pr(Z_j = c | p) \prod_{w \in W_j} \Pr(y_j^w | Z_j = c, d_j, e^w)$$

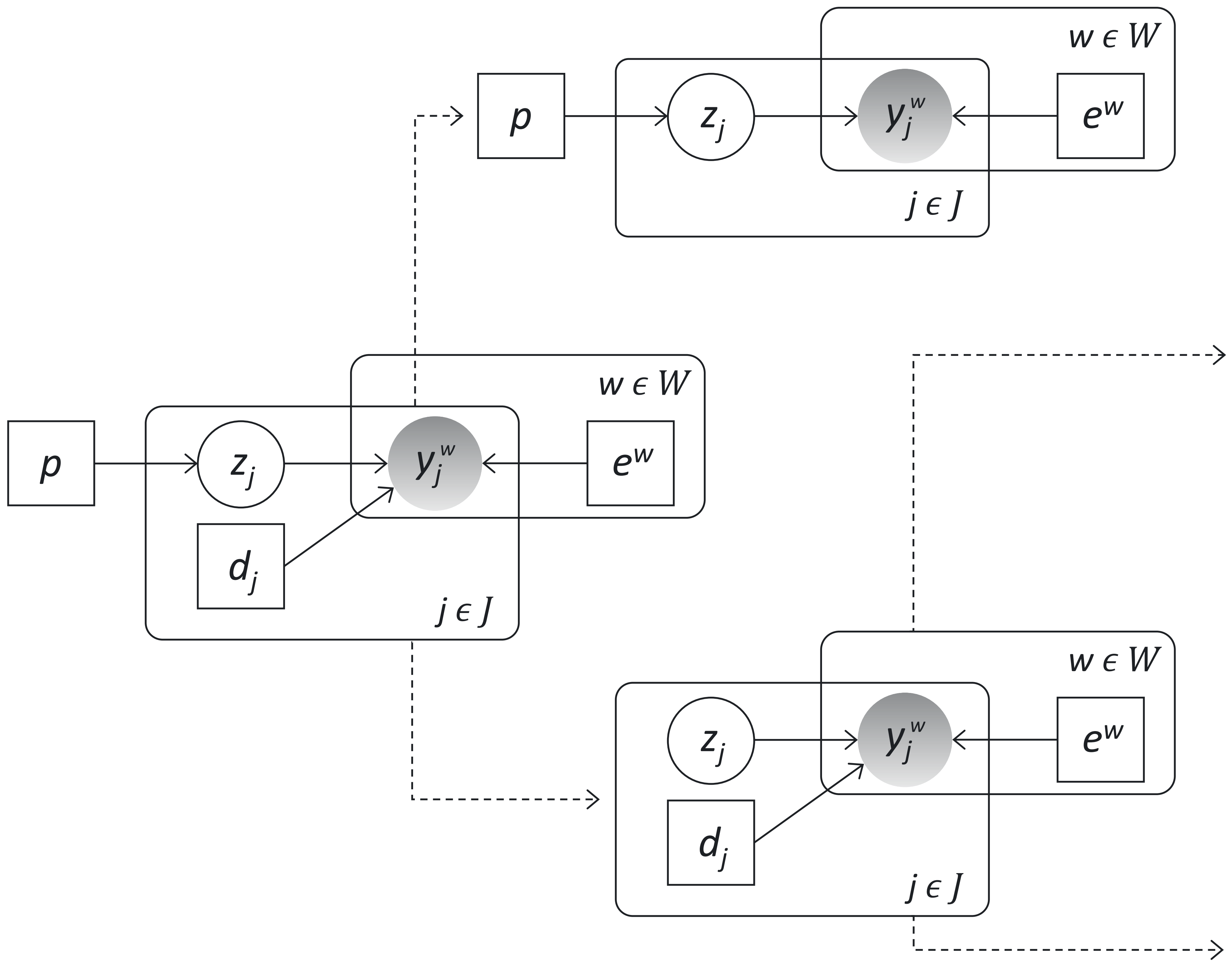
- **M-step:** Maximize the expectation of LL with respect to the posterior distribution of $\hat{z}$:

$$(p, d, e) = \operatorname{argmax} \mathbb{E}_{\hat{z}} \log \Pr(z_j | p) \prod_{w \in W_j} \Pr(y_j^w | z_j, d_j, e^w)$$

  - Analytical solutions
  - Gradient descent

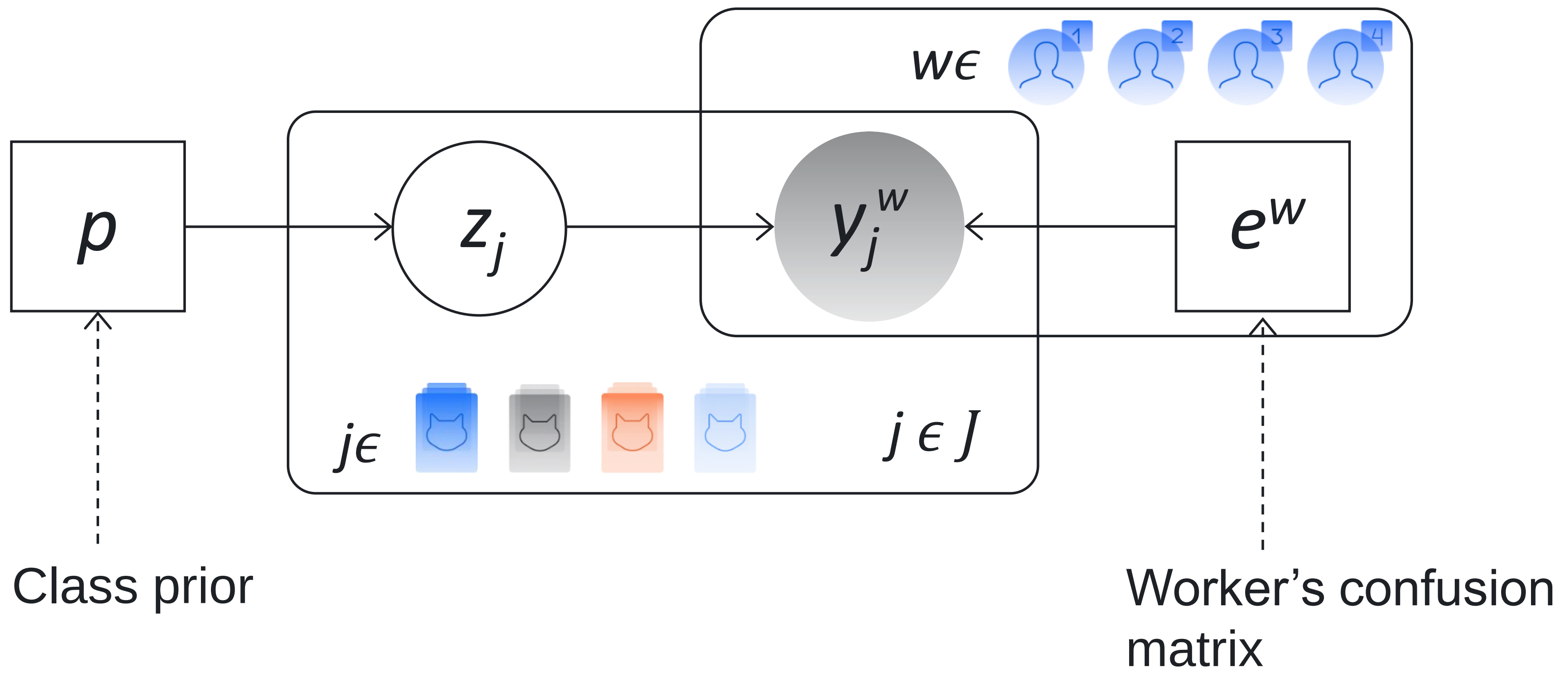
* it is a lower bound on LL of $\mathbf{y}$ and $\mathbf{z}$

Latent label model (LLM): special cases



- Dawid and Skene model (DS):
  - Categories are different
  - Objects are **similar**
  - Workers are different
- Generative model of labels, abilities, and difficulties (GLAD):
  - Categories are **similar**
  - Objects are different
  - Workers are different
- Minimax conditional entropy model (MMCE):
  - Categories are different
  - Objects are different
  - Workers are different

Dawid and Skene model (DS)



LLM with parameters:

- p — vector of length K: $p[i] = \Pr(Z = c)$
- e^w — matrix of size $K \times K$: $e^w[c, k] = \Pr(Y^w = k | Z = c)$
- Model:
  - Z_j ~Mult(p)
  - y_j^w ~Mult$\left(e^w[z_j, \cdot]\right)$

DS: parameters optimization

- **E-step:**

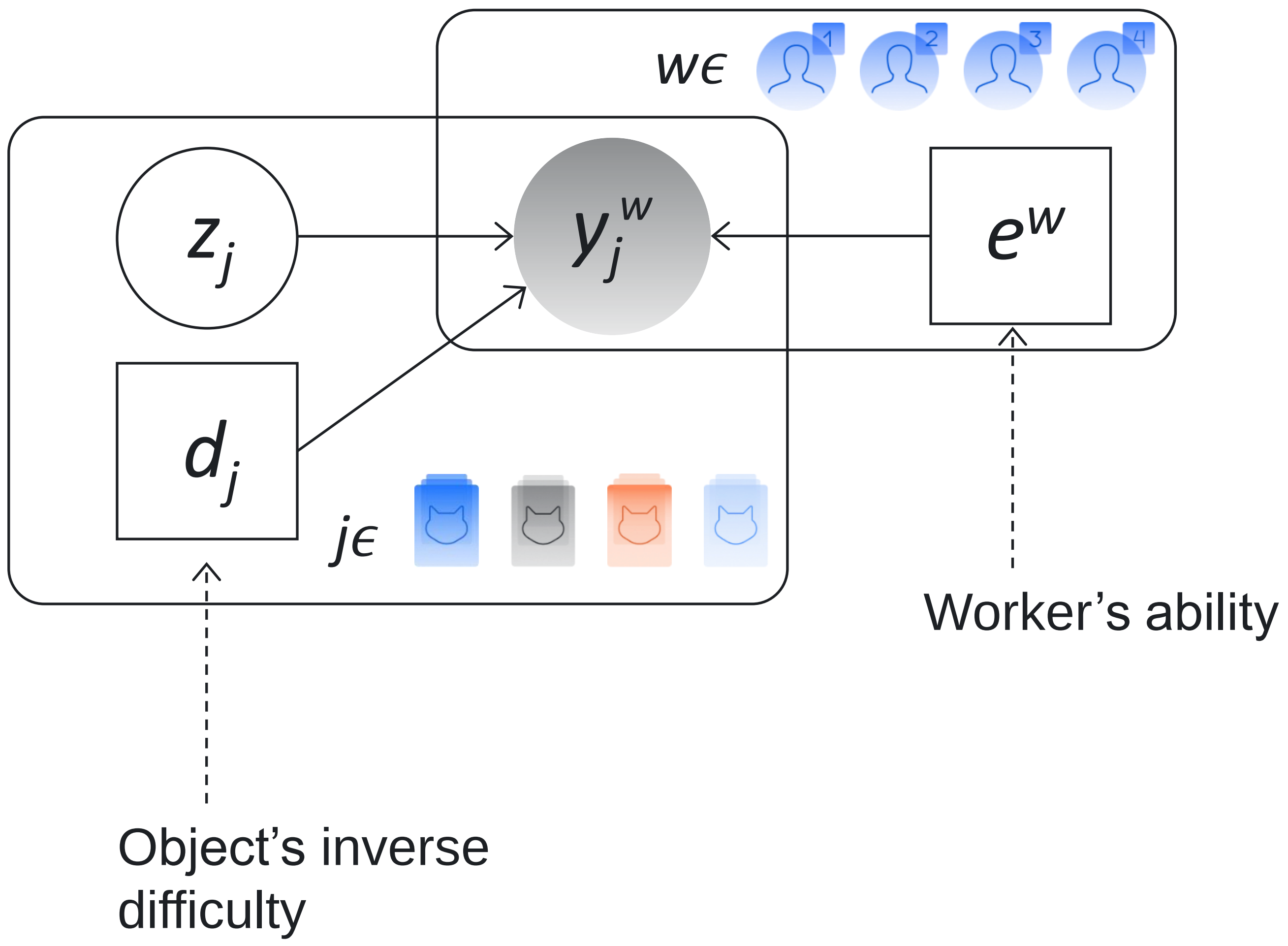
$$\hat{z}_j[c] = \frac{p[c] \; \prod_{w \in W_j} e^w\left[c, y_j^w\right]}{\sum_k p[k] \; \prod_{w \in W_j} e^w\left[k, y_j^w\right]}, \qquad c = 1, \dots, K$$

- **M-step:** Analytical solution

$$e^w[c, k] = \frac{\sum_{j \in J} \hat{z}_j[c] \delta\left(y_j^w = k\right)}{\sum_{q=1}^{K} \sum_{j \in J} \hat{z}_j[c] \delta\left(y_j^w = q\right)}, \qquad k, c = 1, \dots, K$$

$$p[c] = \frac{\sum_{j \in J} \hat{z}_j[c]}{J}, \qquad c = 1, \dots, K$$

Generative model of Labels, Abilities, and Difficulties (GLAD)



LLM with parameters:

- Scalar $d_j \in (0, \infty)$
- Scalar $e^w \in (-\infty, \infty)$
- Model:

$$\Pr\left(Y_j^w = k | Z_j = c\right) = \begin{cases} a(w,j), & c = k \\ \dfrac{1 - a(w,j)}{K - 1}, & c \neq k \end{cases}$$

$$\text{where } a(w,j) = \frac{1}{1 + \exp(-e^w d_j)}$$

GLAD: parameters optimization

- Let $a(w,j) = \frac{1}{1+\exp(-e^{w} d_j)}$ and $P(z_j)$ be a predefined prior (e.g., $P(z_j) = {}^{1}/_{K}$)

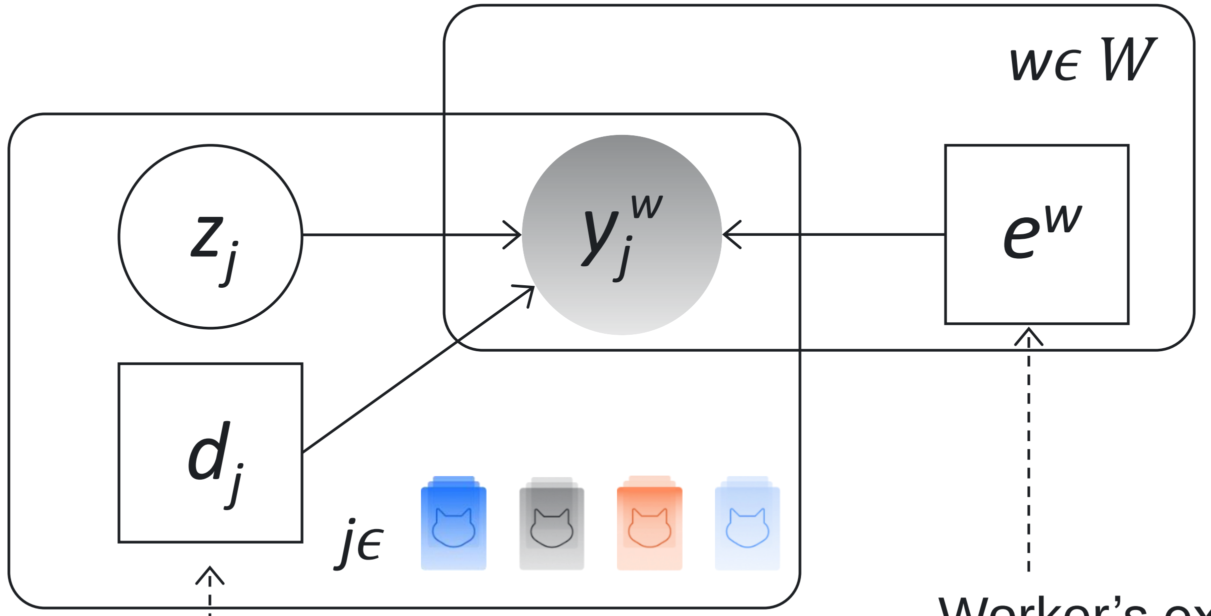
- **E-step:**

$$\hat{z}_j\,[c] \propto P(Z_j = c) \prod_{w \in W_j} a(w,j)^{\delta\left(y_j^w = c\right)} \left(\frac{1 - a(w,j)}{K-1}\right)^{\delta\left(y_j^w \neq c\right)}, \quad c = 1, \dots, K$$

- **M-step:** estimate (d, e) for given $\hat{z}$ using gradient descent

$$(d^t, e^t) = \operatorname{argmax} \sum_{j \in J} \left[\mathbb{E}_{\hat{z}_j} \log P(z_j) + \sum_{w \in W_j} \mathbb{E}_{\hat{z}_j} \log \Pr\left(y_j^w \middle| z_j\right) \right]$$

MiniMax Conditional Entropy model (MMCE)



Worker's expertise matrix

Object's confusability matrix

- LLM with parameters:
  - d_j — matrix of size $K \times K$
  - e^w — matrix of size $K \times K$
  - Noisy label model*

$$\Pr\left(Y_j^w = k | Z_j = c\right) = \exp\left(d_j[c,k] + e^w[c,k]\right)$$

*The model was derived by minimizing the maximum conditional entropy of observed labels

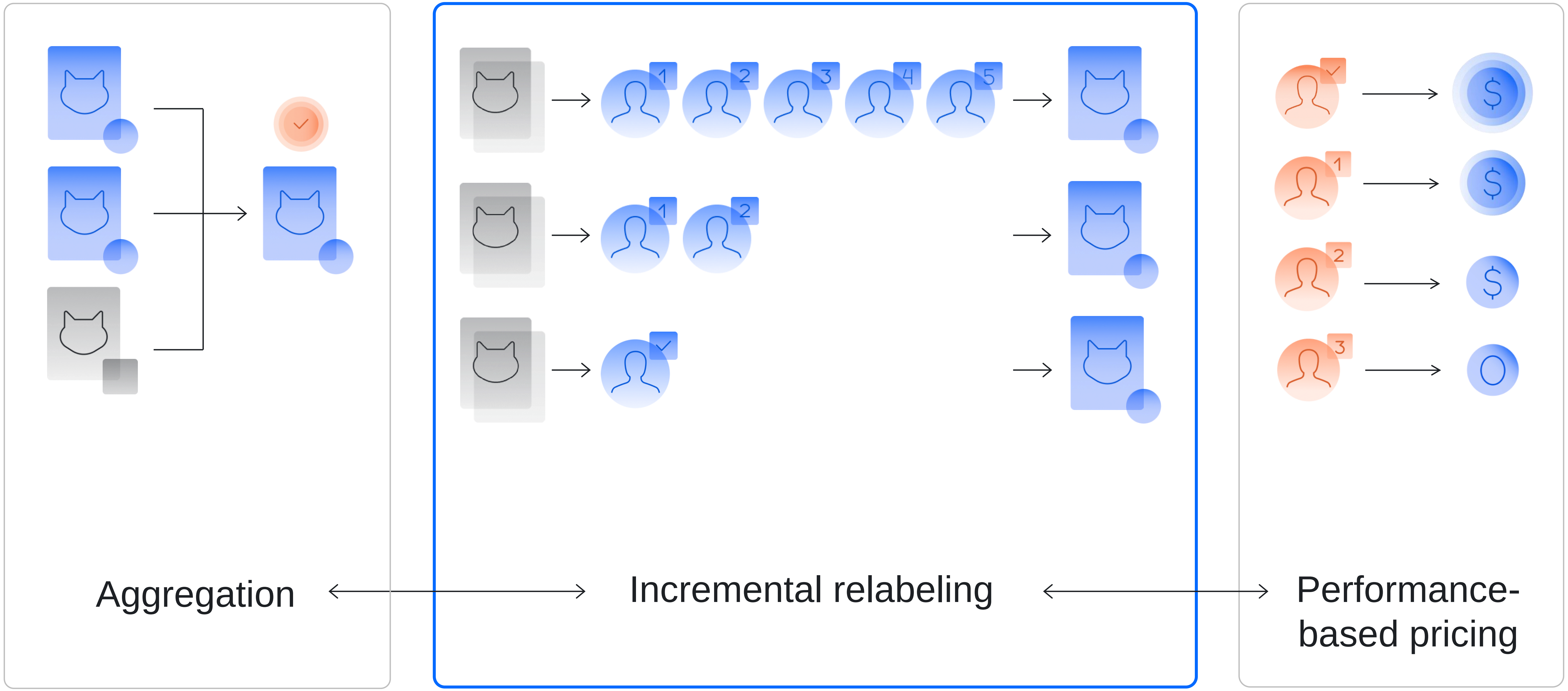
$$\min_Q \max_P - \sum_{\substack{j \in J \\ c \in \{1,\dots,K\}}} Q\left(Z_j = c\right) \sum_{\substack{w \in W \\ k \in \{1,\dots,K\}}} P\left(Y_j^w = k | Z_j = c\right) \log P\left(Y_j^w = k | Z_j = c\right)$$

Zhou et al., Learning from the Wisdom of Crowds by Minimax Entropy, 2012

Summary of aggregation methods

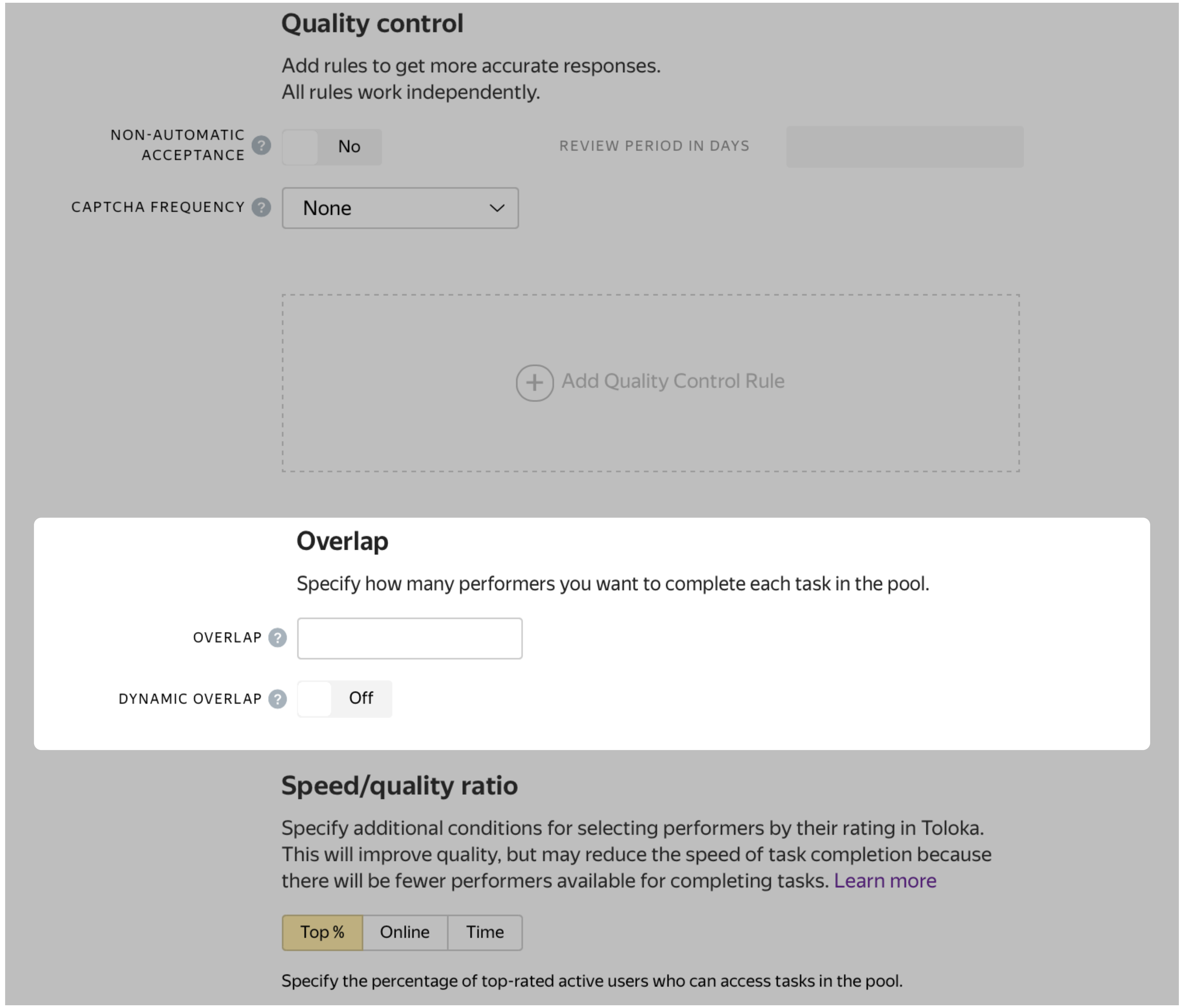
	MV	DS	GLAD	MME
Categories (K)				
Objects (J)				
Workers (W)	1 1 1	1 2 3	1 2 3	1 2 3
Number of parameters	0	$WK^2 + K$	$W + J$	$(W + J)K^2$

Key components of labeling with crowds



Incremental relabeling
aka dynamic overlap

Pool settings: dynamic overlap



Incremental relabeling problem

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

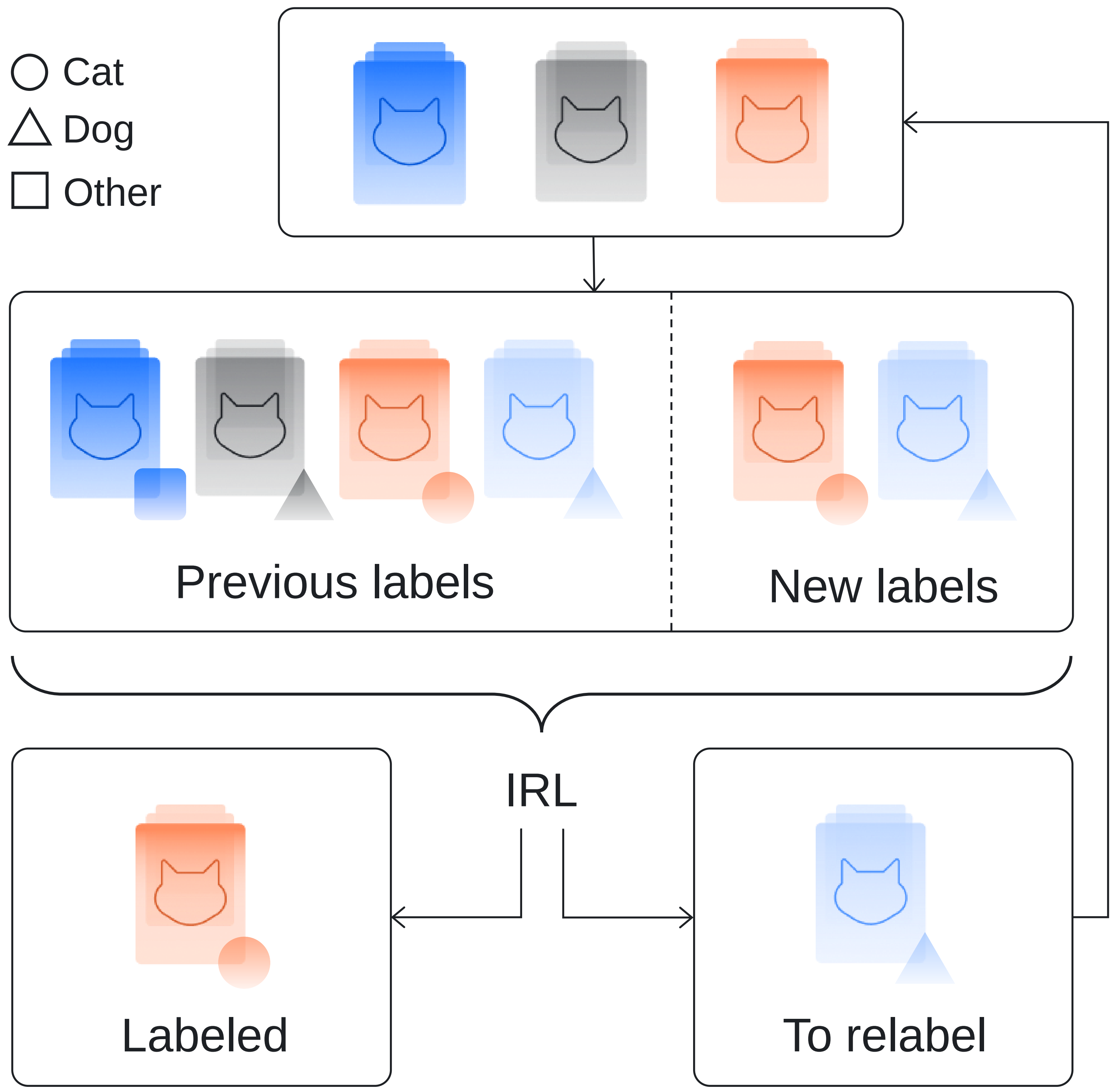
Incremental relabeling scheme (IRL)

Request 1 label for each object

In real time IRL algorithm receives:
(1) previously accumulated labels
(2) new labels

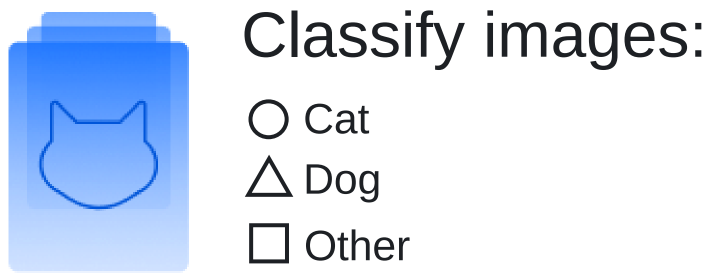
Decides:
(1) which objects are labeled
(2) which objects to relabel

Repeat until all tasks are labeled

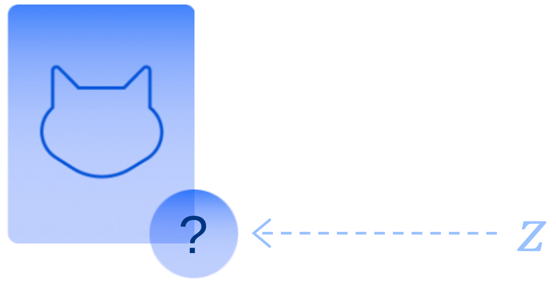


Notations

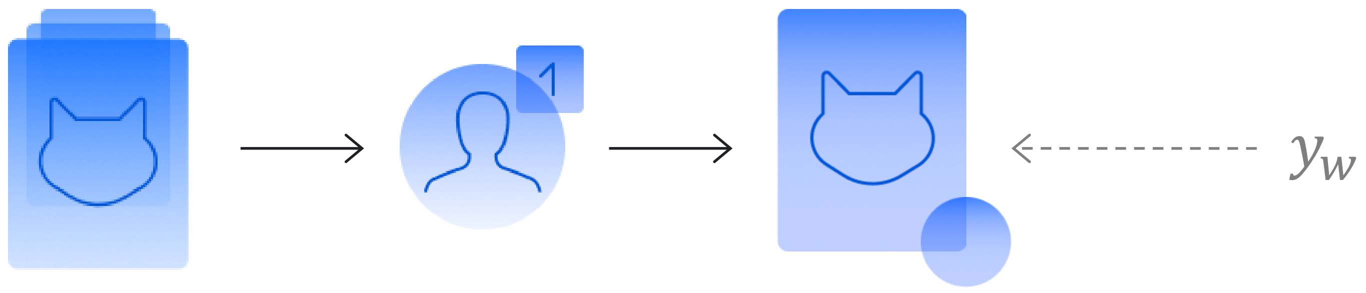
- Consider one object



- $z \in \{1, \dots, K\}$ — latent true label



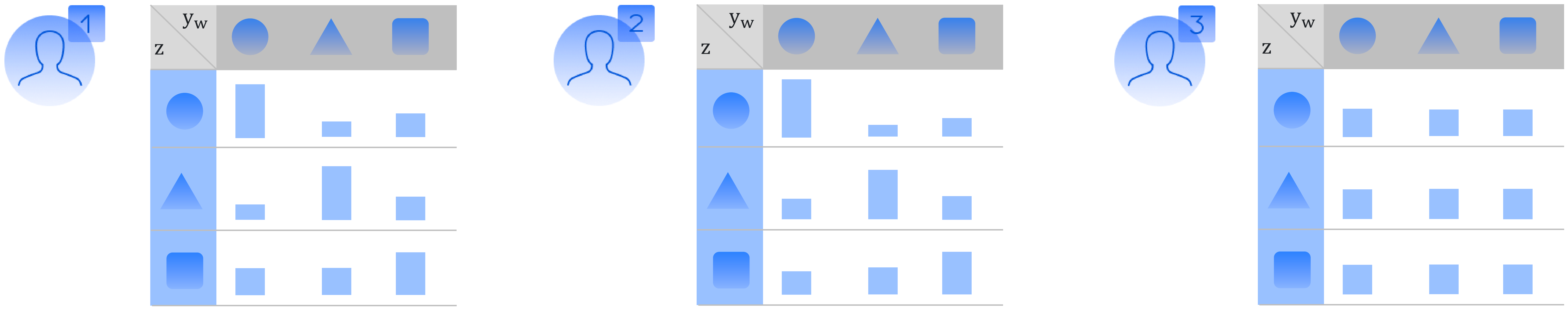
- $y_w \in \{1, \dots, K\}$ — observed noisy label from worker w:



Notations

- Noisy label model for worker w:

$$M_w \in [0,1]^{K\times K}: \Pr(Y_w = k | Z = c) = M_w[c,k]$$

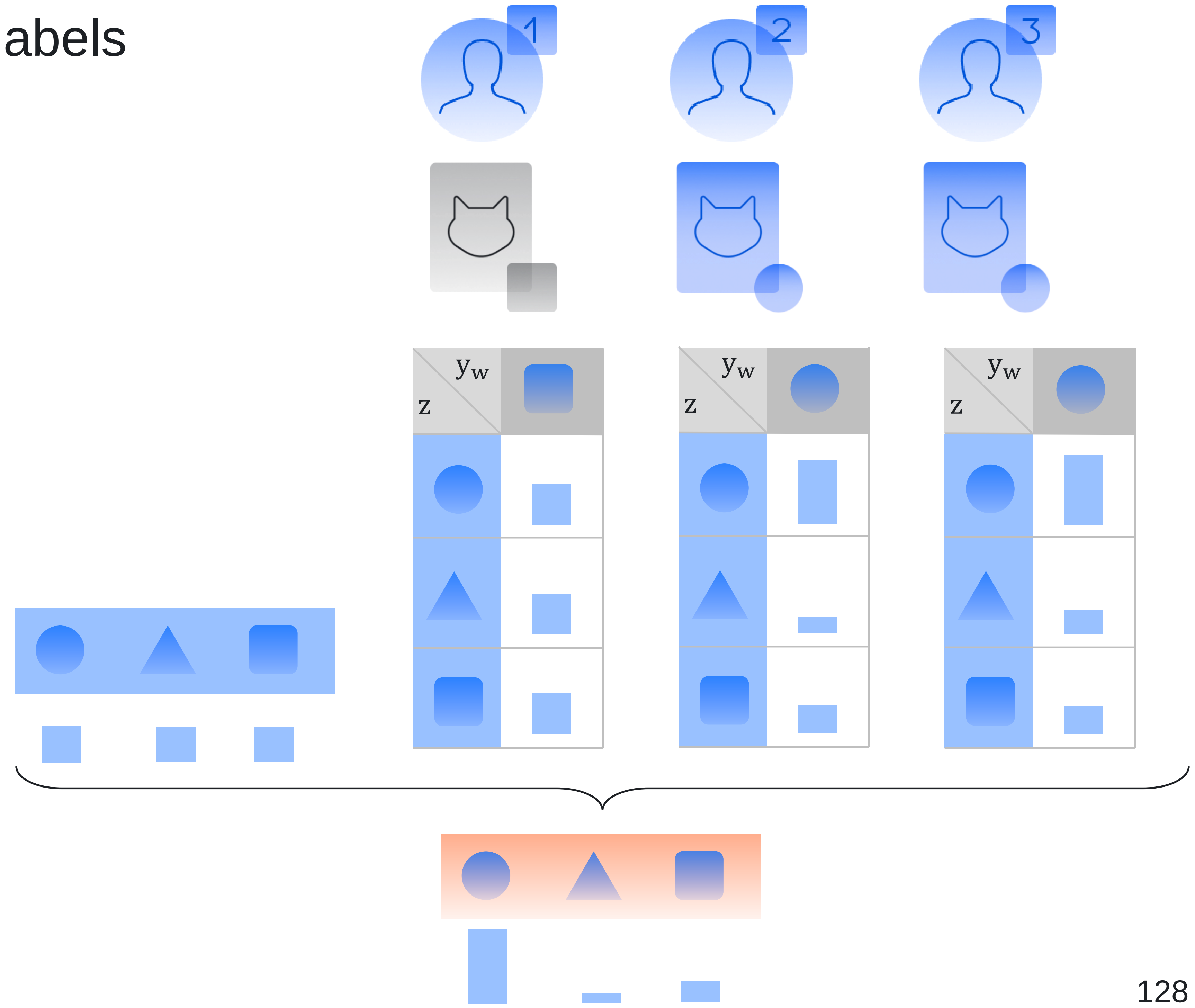


- Prior distribution: $\Pr(Z = k) = p_k$

Posterior distribution

- $\{y_{w_1}, \dots, y_{w_n}\}$ — accumulated noisy labels for the object
- Using Bayes rule:

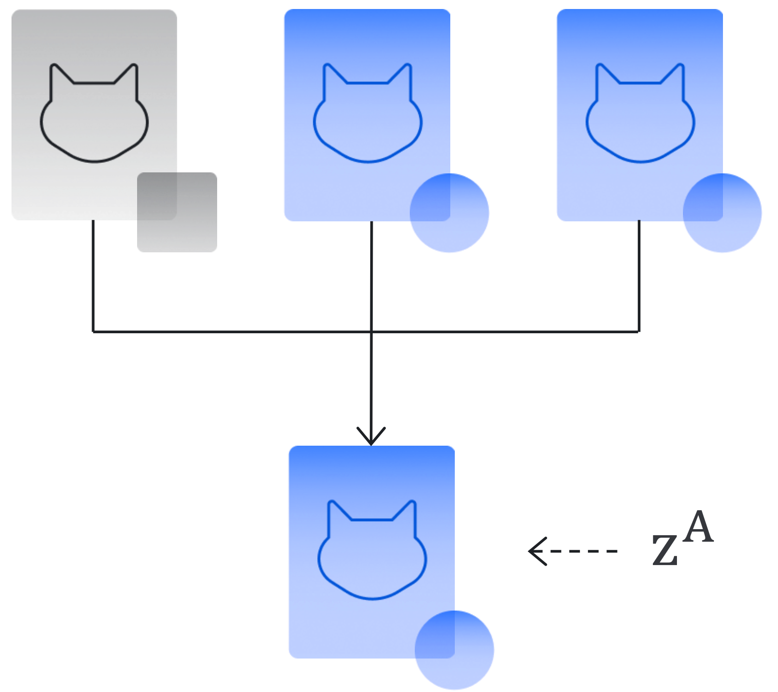
$$\Pr(Z = k | \{y_{w_1}, \dots, y_{w_n}\}) = \frac{\Pr(Z = k)\Pr(\{y_{w_1}, \dots, y_{w_n}\} | Z = k)}{\Pr(\{y_{w_1}, \dots, y_{w_n}\})} = \frac{p_k \prod_{i=1}^{n} M_{w_i}[k, y_{w_i}]}{\sum_{t=1}^{K} p_t \prod_{i=1}^{n} M_{w_i}[t, y_{w_i}]}$$



Expected accuracy of aggregated labels

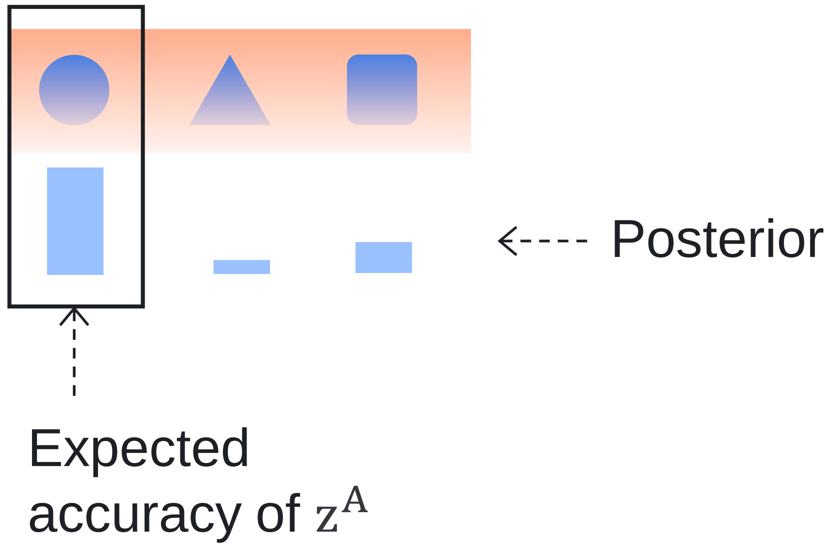
- Let A be an aggregation model, e.g. MV, DS, GLAD,…
- Denote aggregated label $z^A = A(\{y_{w_1}, \dots, y_{w_n}\})$
- Expected accuracy of aggregated labels given noisy labels is

$$E(\delta(z = z^A) | \{y_{w_1}, \dots, y_{w_n}\}) = \Pr(z = z^A | \{y_{w_1}, \dots, y_{w_n}\})$$



- Stop labeling if $E(\delta(z = z^A) | \{y_{w_1}, \dots, y_{w_n}\}) \geq C$

Parameter



Incremental relabelling algorithm

Input: $\cup_{t=1}^{T-1} Y^t$ — previous labels till step T

Y^T — new labels

Output: R — objects to relabel

For each object j with a label in Y^T: ← Object with a new label

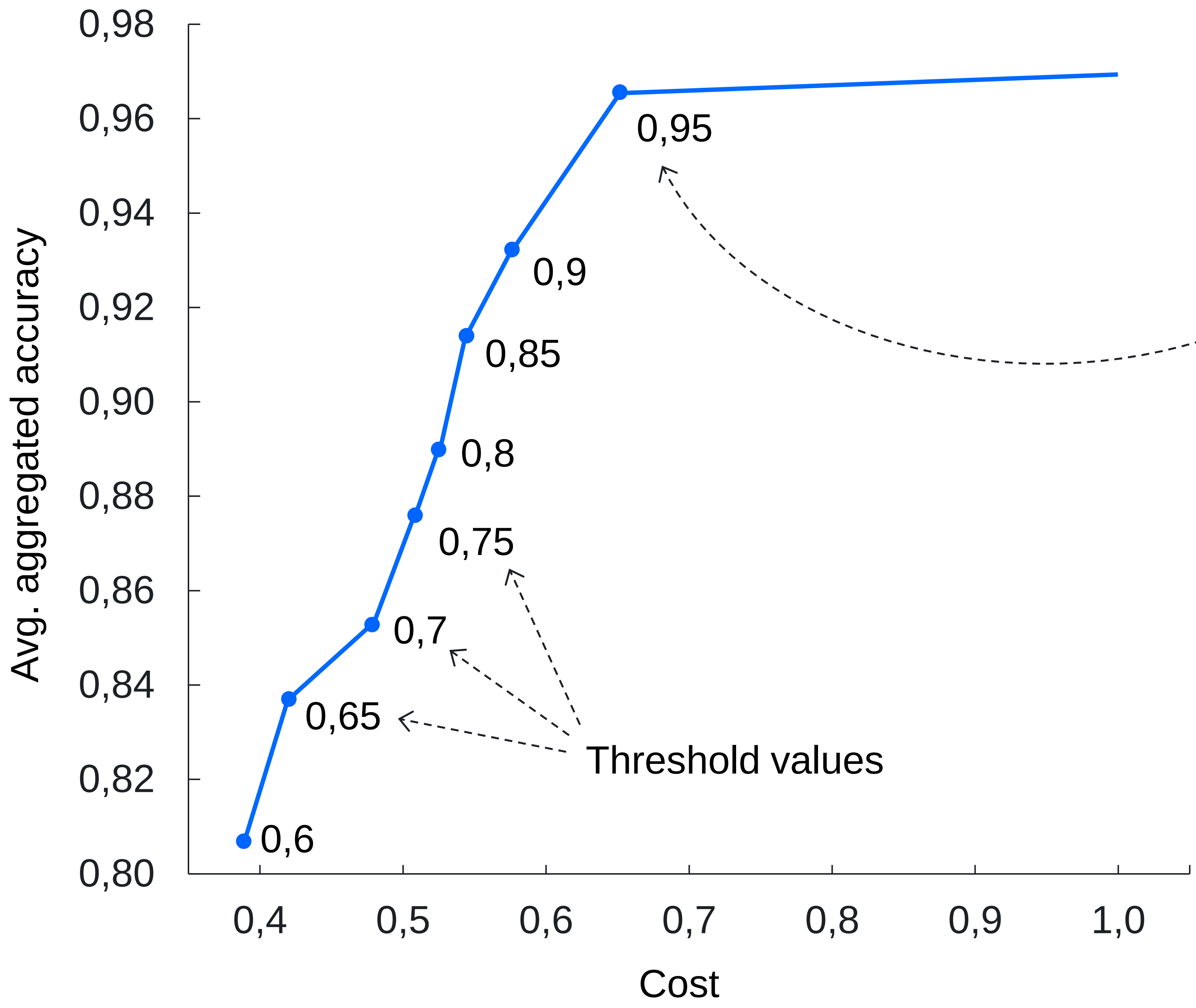
$z_j^M = M(\cup_{t=1}^{T} Y^t)$ ← Current aggregated label

$c_j = E(z_j = z_j^M | \cup_{t=1}^{T} Y^t)$ ← Expected accuracy for the current aggregated label

If $c_j < c$, then $R = R \cup j$

Parameter: c — threshold for expected accuracy

Threshold in IRL: cost – accuracy trade-off



- Optimal threshold $c = 0.95$
- A higher c does not increase accuracy
- Saving ≈ 35% of noisy labels

How to obtain a cost-accuracy plot

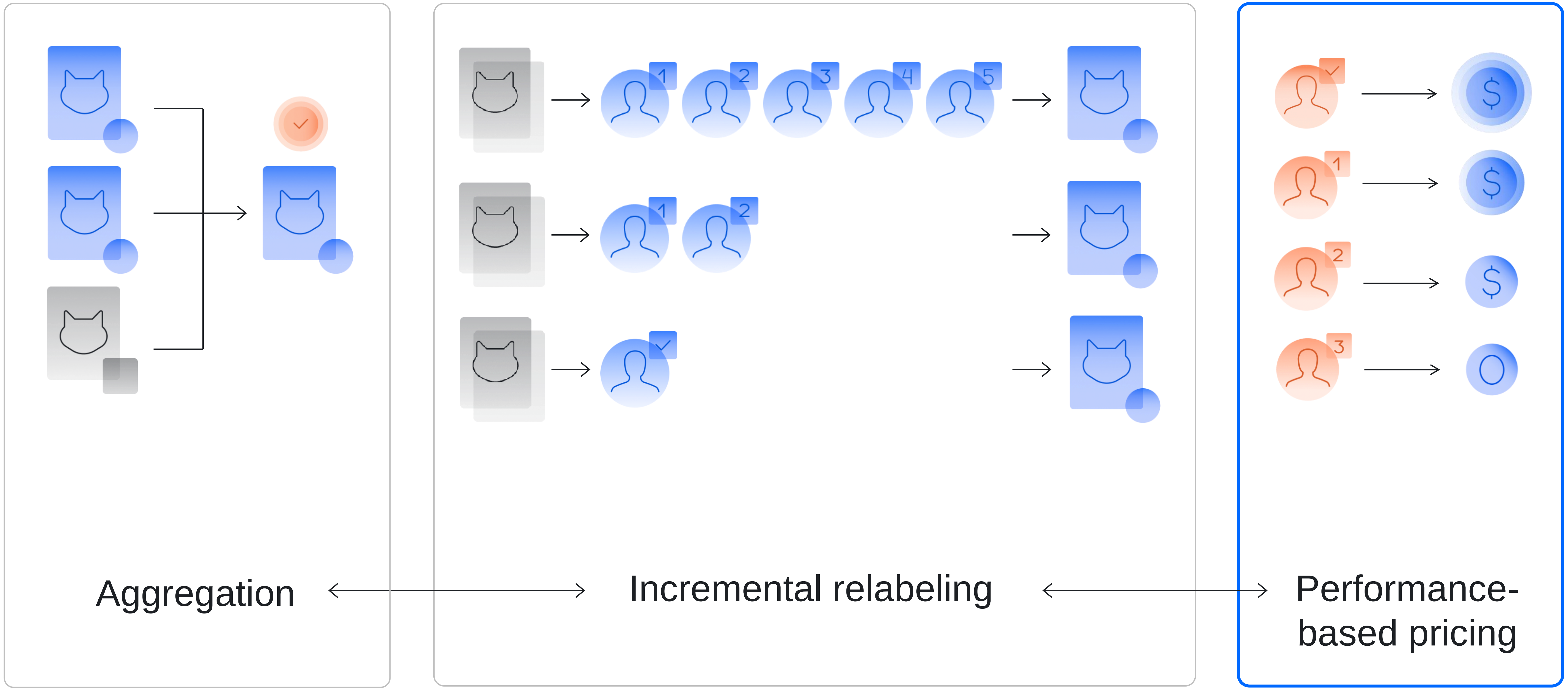
Data for the plot:

- Label a pool of objects with a redundant overlap (e.g., 10)
- Obtain ground truth labels for the objects (e.g., expert labels or MV labels)

Simulate IRL with different thresholds using the data:

- For each threshold c from a grid $0 < c_0 < \ldots < c_m \leq 1$
- Repeat N times:
  1. Shuffle noisy labels and fix the order of labels
  2. Draw labels sequentially and test the IRL condition after each label
  3. Once the IRL condition for an object is met, discard unused labels for the object
  4. When all objects are labelled calculate
     - accuracy of aggregated labels
     - cost as the fraction of used noisy labels
- Average N values of aggregated accuracy and N values of cost for each value of threshold c

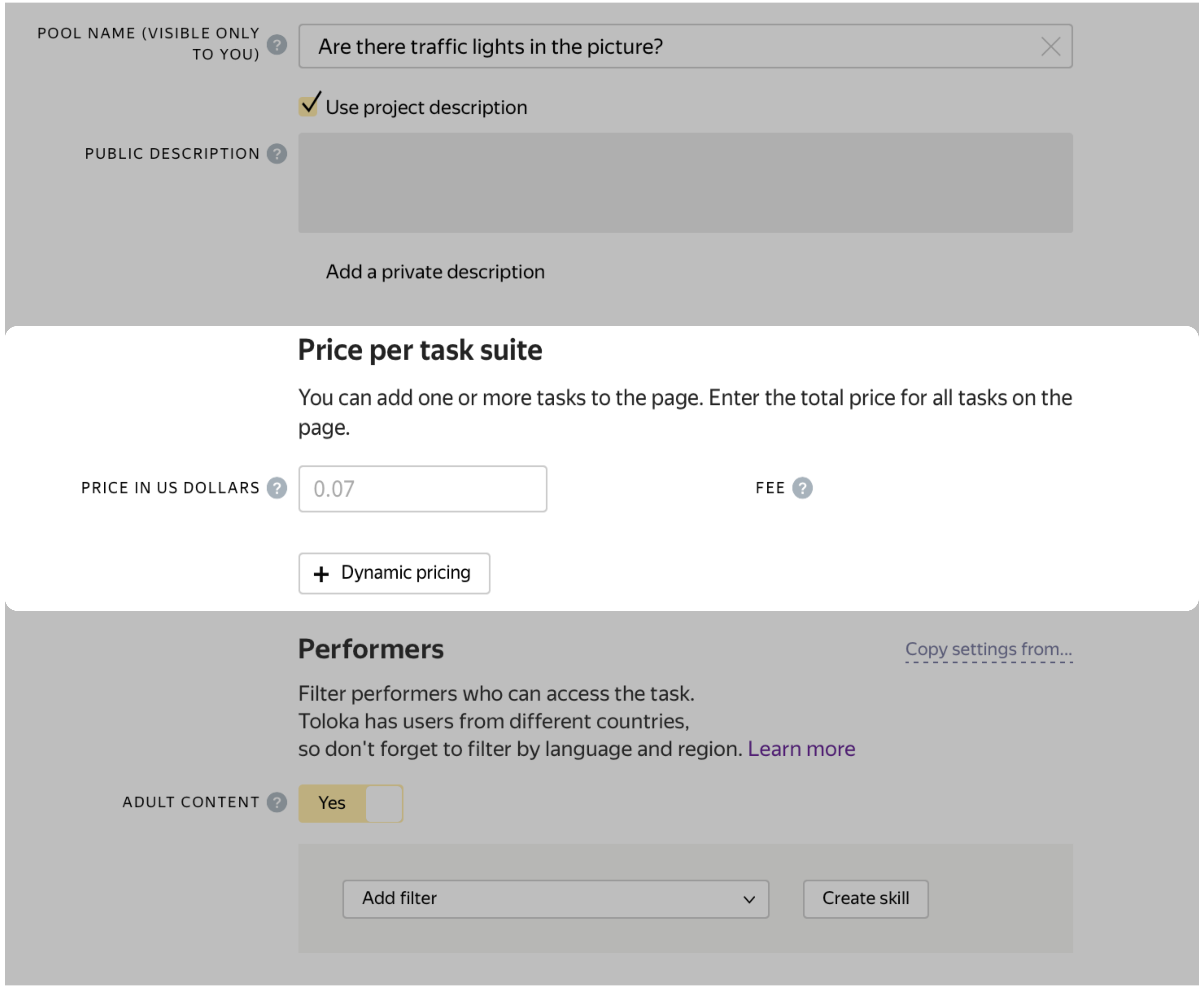
Key components of labeling with crowds



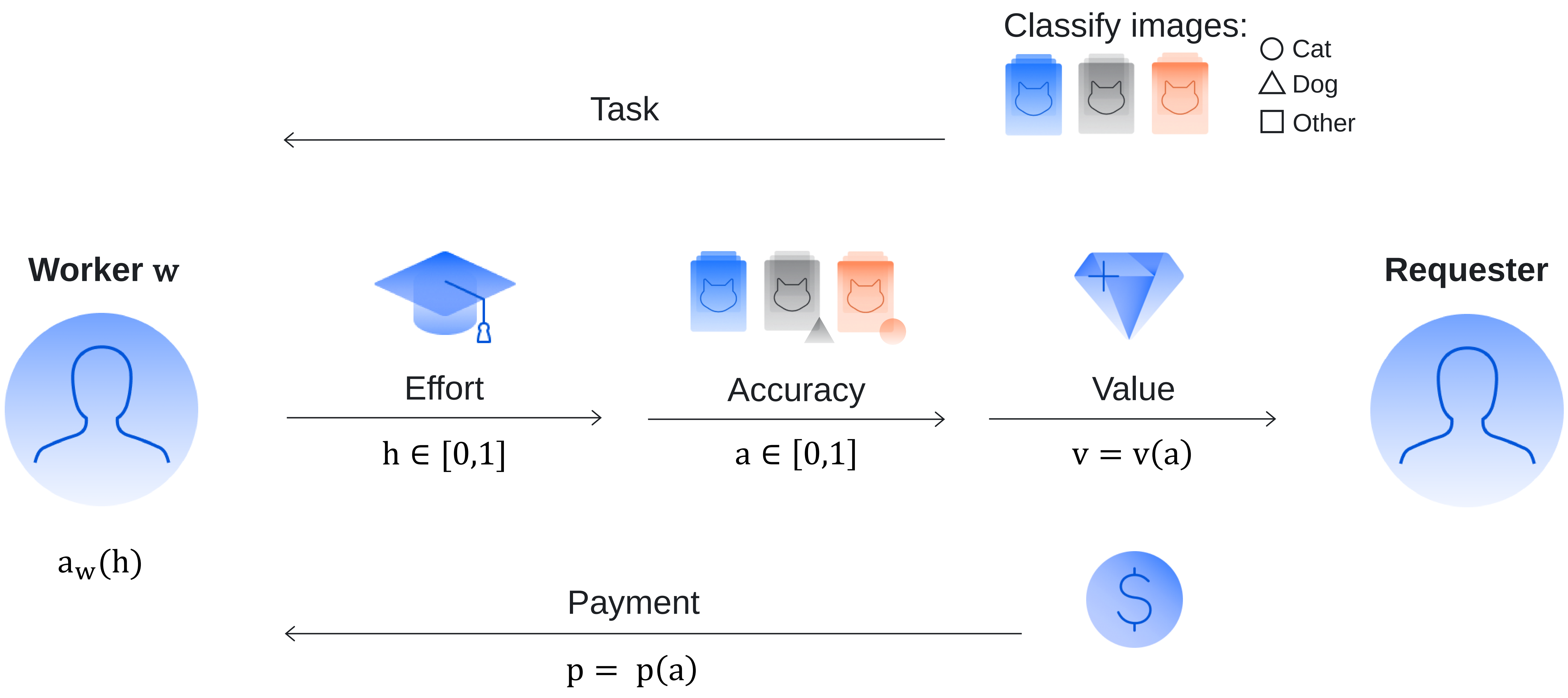
Performance-based pricing

aka dynamic pricing

Pool settings: dynamic pricing

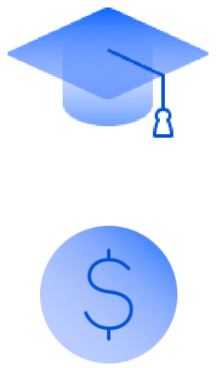


Labeling as a game: notation



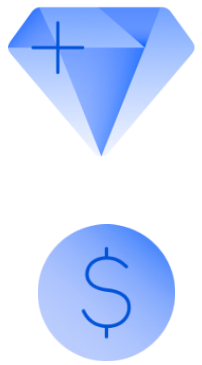
Labeling as a game: formalization

- Each worker w chooses a level of effort h for labeling object to maximize earnings per unit of spent effort:



$$\frac{p\big(a_w(h)\big)}{h} \to \max_{h \geq 0}$$

- The requester chooses a pricing p(a) to minimize payments per unit of obtained value



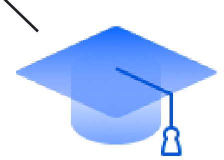
$$\frac{v(a\ )}{p(a)} \to \max_{a \in [0,1]}$$

Labeling as a game: incentive compatible pricing

- Assume $a_w(h)$ is a linear function of h:

$$a_w(h) = c_1 h + c_0$$

Accuracy



Theorem: the requester and workers maximize their utility simultaneously if the pricing $p(a)$ for each label is proportional to its accuracy a

Performance-based pricing in practice: settings

- Price p for the level of accuracy a_0 : $\Pr(\hat{z} = z) \geq a_0$ E.g.:

p = 0.3$

a_0 = 0.99

- $\hat{q}_w = \Pr(y^w = z)$ — estimated quality level of worker w, e.g. the fraction of correct labels for golden set (GS):

5 correct GS among 10
$\hat{q}_w = 0.5$

16 correct GS among 20
$\hat{q}_w = 0.8$

100 correct GS among 100
$\hat{q}_w = 1$

Performance-based pricing in practice: settings

- Aggregation $\hat{z}_j^{wMV} = \arg\max_{y=1,\ldots,K} \sum_{w \in W_j} \hat{q}_w \delta(y = y_j^w)$

- IRL algorithm is based on the expected accuracy of $\hat{z}_j^{wMV}$

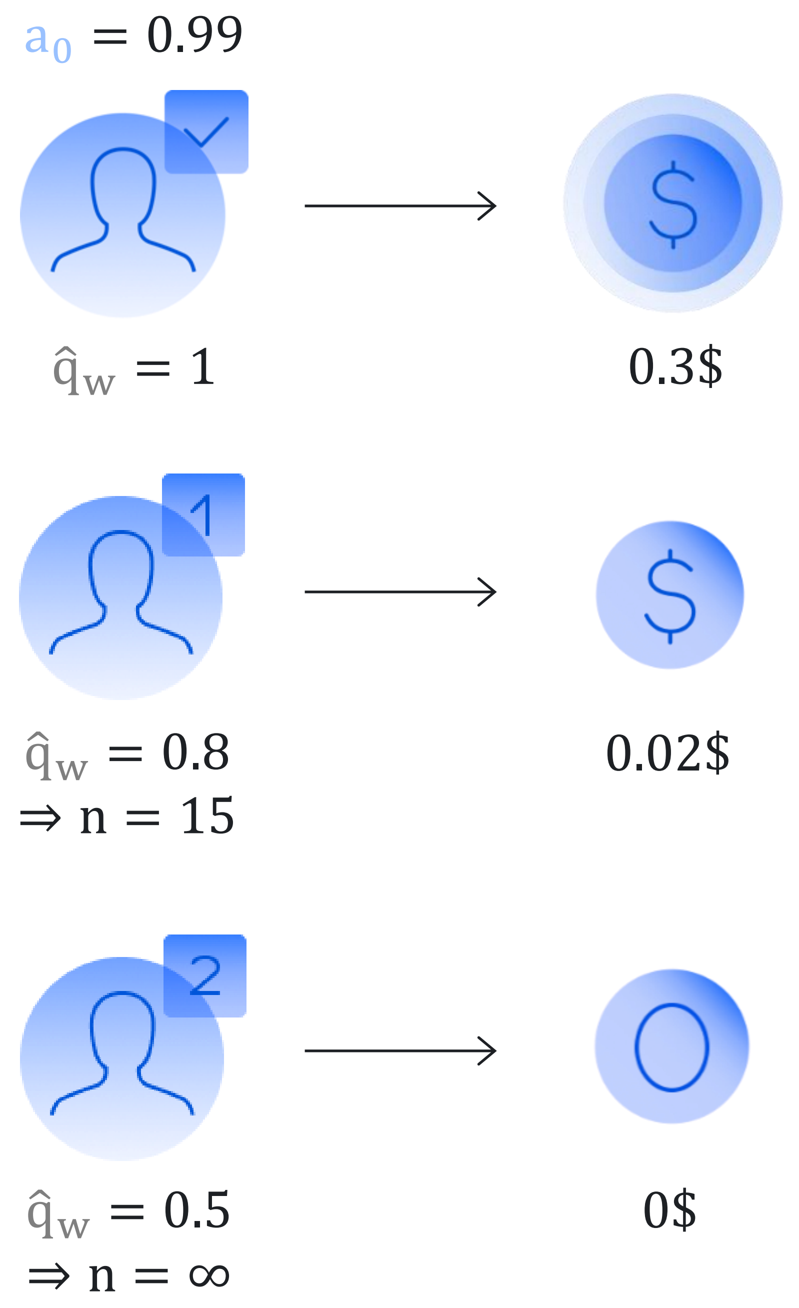
Performance-based pricing in practice

- Pricing rules

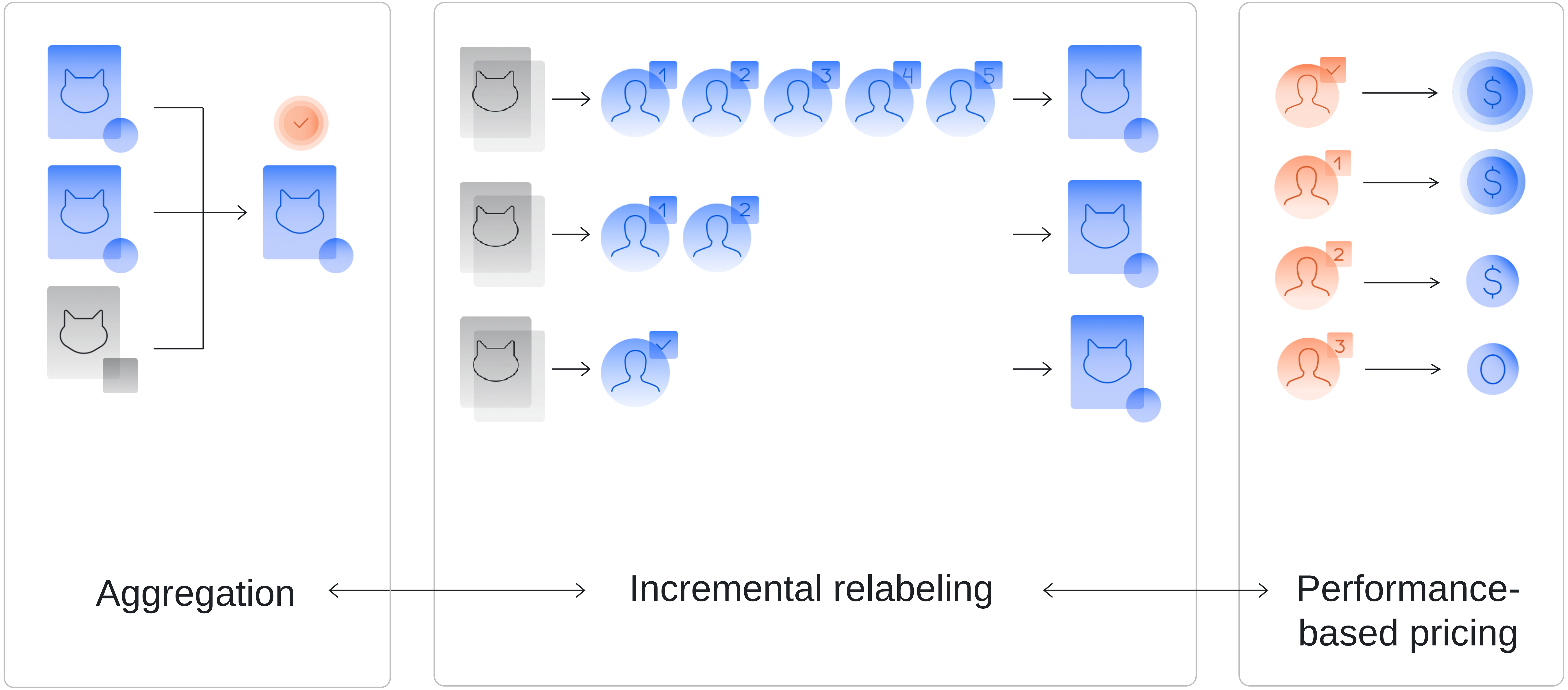
1. If $\hat{q}_w \geq a_0$, then the price is p
2. Else find n:

$$\underbrace{\sum_{k=0}^{n/2}\binom{n}{k}\hat{q}_w^{n-k}(1-\hat{q}_w)^k}_{\text{Expected accuracy for MV}} \geq a_0$$

The price is p/n



Key components of labeling with crowds

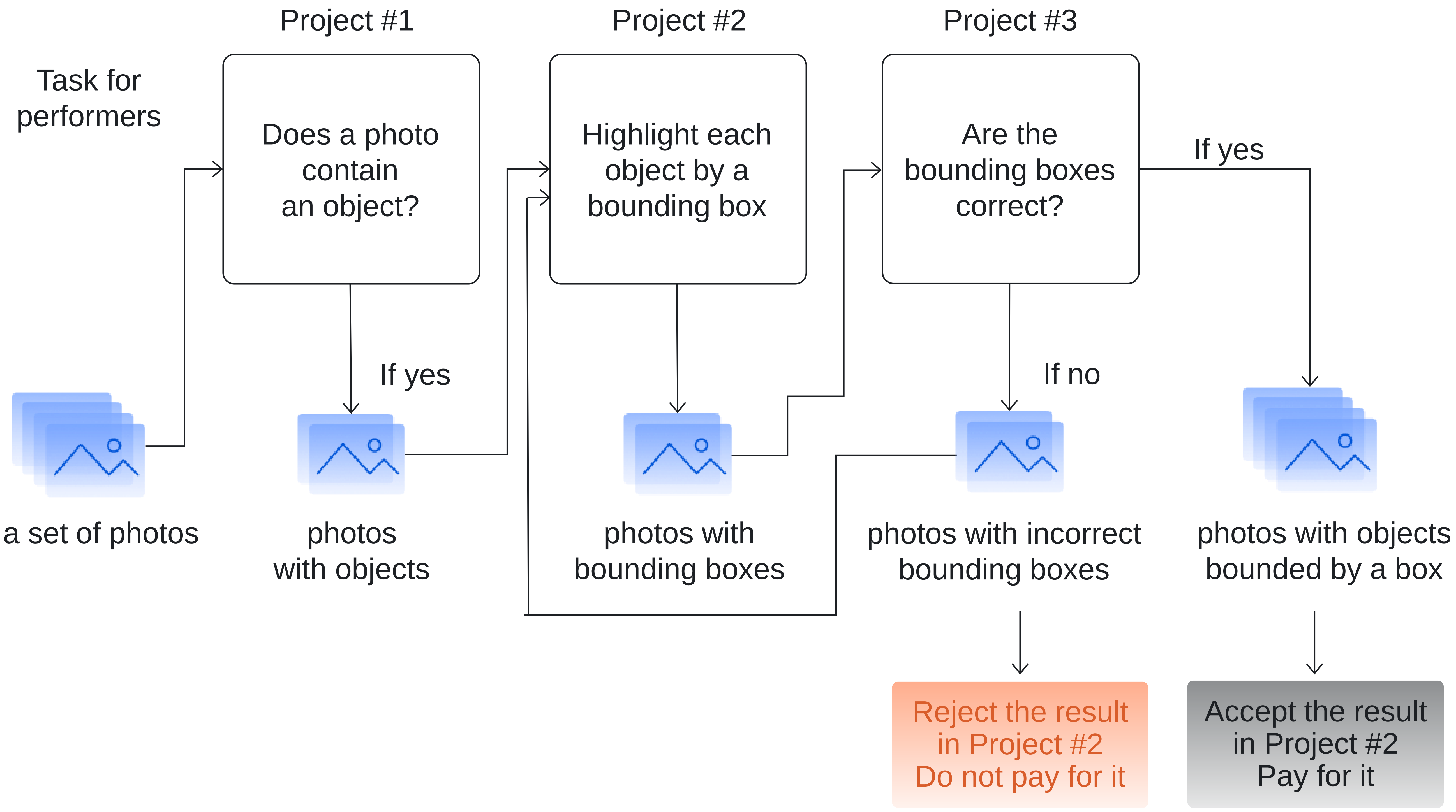


Part VI

Discussion of results from the projects. Conclusions

Olga Megorskaya,
CEO

Reminder: we implemented the pipeline



Project #1: Filter out photos without objects

Task

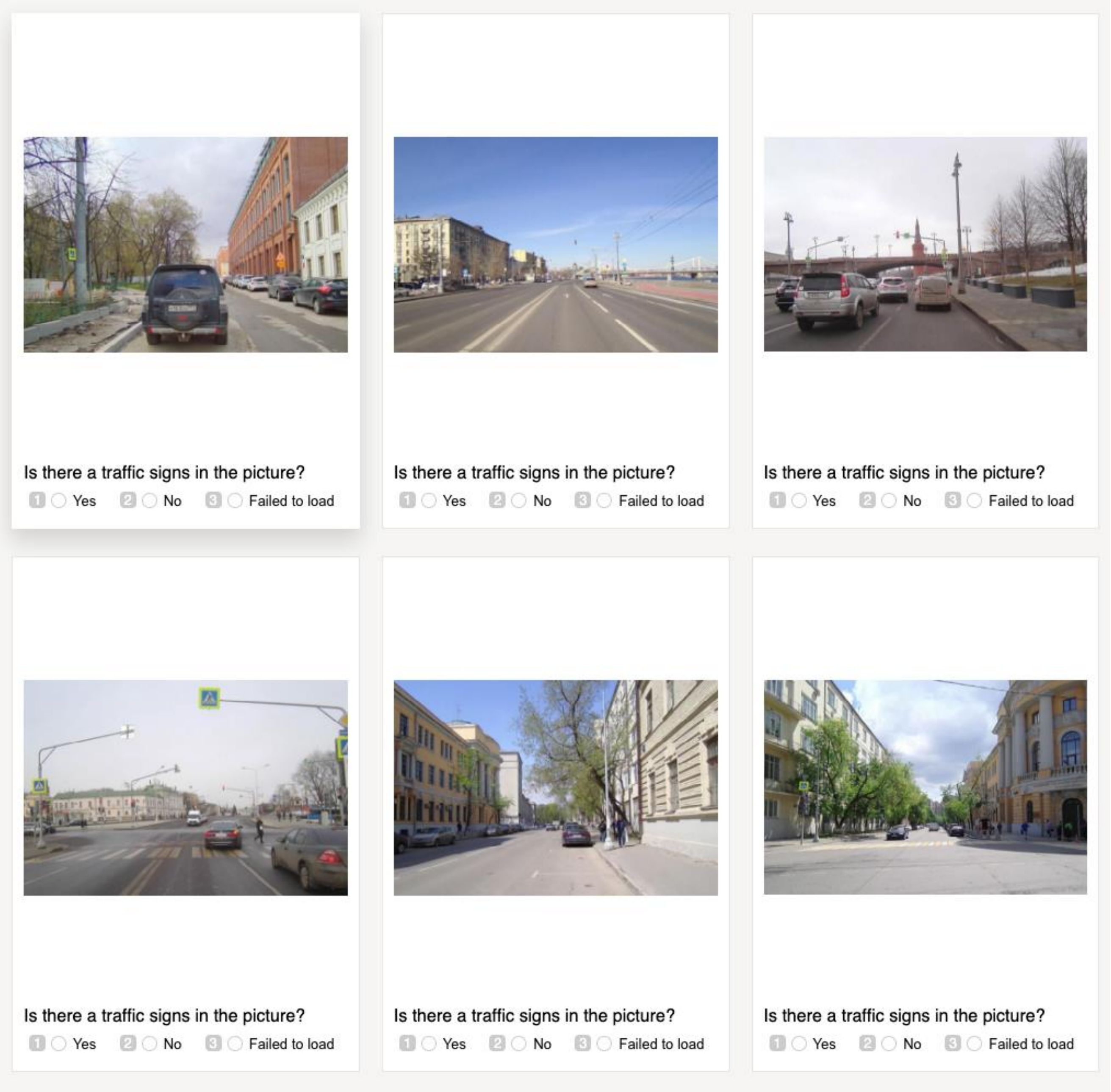
► Does a photo contain objects of desired type?

Our results:

► 100 photos evaluated

► Within 4 min on real performers

► Cost: $0.3 + Toloka fee



Project #2: Highlighting objects by rectangles

Task

- ► Highlight each object of desired type by a bounding box

Our results:

- ► 67 photos processed
- ► Within 5.5 min on real performers
- ► Cost: $0.67 + Toloka fee

Project #3: Accept correct bounding boxes

Task

- Are the objects of desired type highlighted by the bounding boxes correctly?

Our results:

- 90 photos evaluated
- Within 5 min on real performers
- Cost: $0.36 + Toloka fee

Statistics over the whole pipeline

- 100 photos processed to highlight desired objects
- Within 14.5 min on real performers
- Total cost: $1.33 + Toloka fee
- Quality of the final result (via manual assessment):
  - Recall: 90% (measured on results of Project #1)
  - Precision: 86% (measured on results of Project #2)

Afterparty: upgrade your pipeline

To obtain more comprehensive data

- Use Polygons instead of Bounding boxes
- Highlight more object types

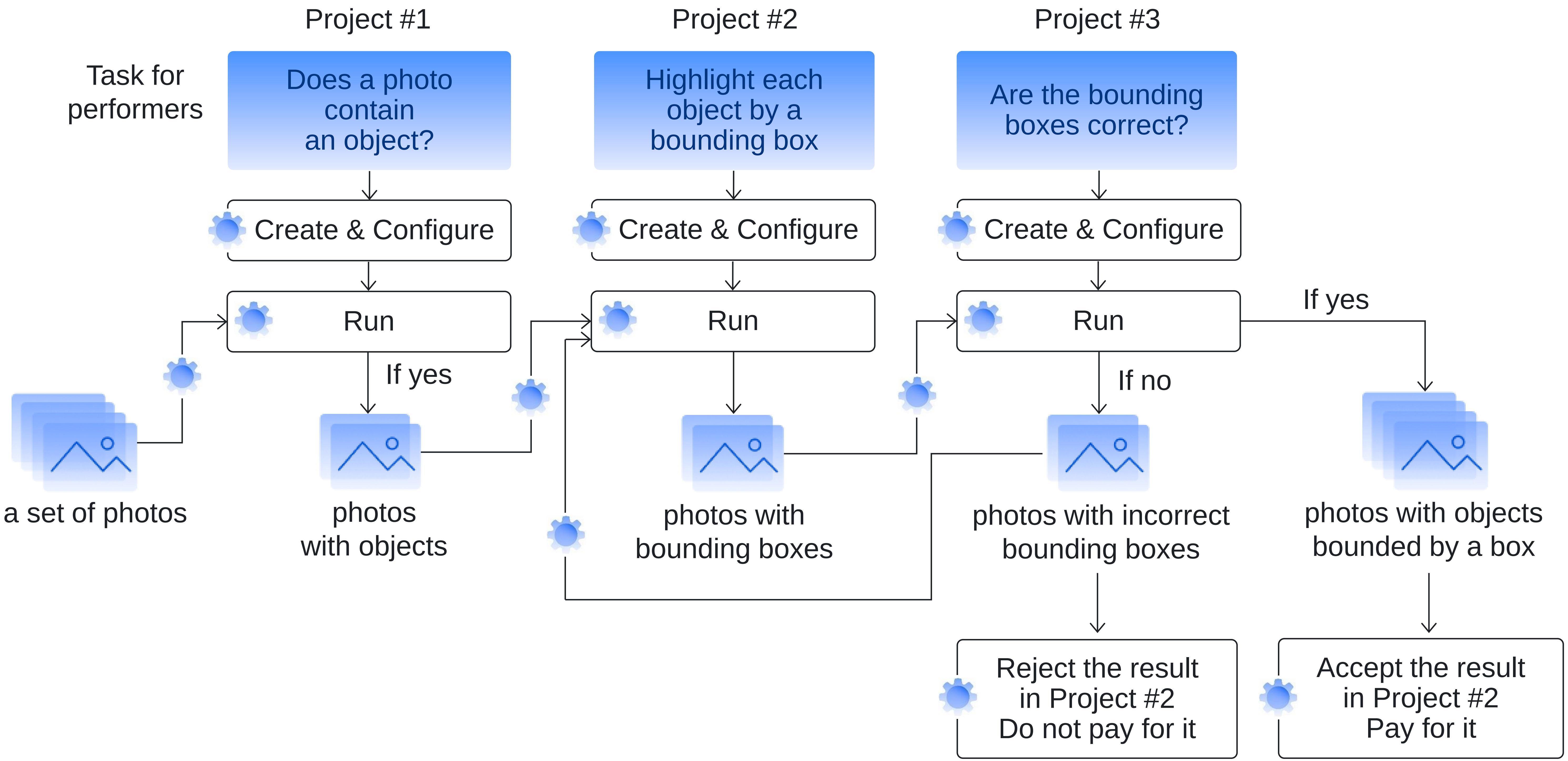
To reduce costs

- Use incremental relabeling aka Dynamic overlap

To improve quality

- Use dynamic pricing
- Add more Golden Sets and hints
- Experiment with aggregation methods
- Add training for workers

API of Toloka



API of Toloka

Allows you to automate all steps of our pipeline

- Discover at: https://yandex.com/dev/toloka/

Crowdsource all types of data

Search Relevance

Moderation

Generation of content

Computer vision

Speech Technologies

References: Aggregation

1. Dawid, A. P. and Skene, A. M, Maximum likelihood estimation of observer error-rates using the EM algorithm, Applied statistics 1979
2. Whitehill, J., Wu, T., Bergsma, J., Movellan, J. R, Ruvolo, P. L, Whose vote should count more: Optimal integration of labels from labelers of unknown expertise}, NIPS 2009
3. Zhou, D., Liu, Q., Platt, J. C, Meek, C., Shah, N. B, Regularized minimax conditional entropy for crowdsourcing, arXiv preprint 2015
4. Raykar, V. C, Yu, S., Zhao, L. H, Valadez, G. H., Florin, C., Bogoni, L., Moy, L., Learning from crowds, JMLR 2010
5. Snow, R., O'Connor, B., Jurafsky, D., Ng, A. Y, Cheap and fast---but is it good?: evaluating non-expert annotations for natural language tasks, EMNLP 2008
6. Ruvolo, P., Whitehill, J.,Movellan, J. R, Exploiting Commonality and Interaction Effects in Crowdsourcing Tasks Using Latent Factor Models, NIPS ’13 Workshop on Crowdsourcing: Theory, Algorithms and Applications
7. Faridani, S. and Buscher, G., LabelBoost: An Ensemble Model for Ground Truth Inference Using Boosted Trees, HCOMP 2013
8. Welinder, P., Branson, S., Perona, P., Belongie, S. J , The multidimensional wisdom of crowds, NIPS 2010
9. Jin, Y., Carman, M., Kim, D., Xie, L., Leveraging Side Information to Improve Label Quality Control in Crowd-Sourcing, HCOMP 2017
10. Imamura, H., Sato, I., Sugiyama, M., Analysis of Minimax Error Rate for Crowdsourcing and Its Application to Worker Clustering Model, arXiv preprint 2018

References: Aggregation

11. Sheshadri, A. and Lease, M., Square: A benchmark for research on computing crowd consensus, HCOMP 2013
12. Kim, H. and Ghahramani, Z., Bayesian classifier combination, AISTATS 2012
13. Venanzi, M., Guiver, J., Kazai, G., Kohli, P., Shokouhi, M., Community-based bayesian aggregation models for crowdsourcing, WWW2014
14. Vuurens, J., de Vries, A. P, Eickhoff, C., How much spam can you take? an analysis of crowdsourcing results to increase accuracy, SIGIR Workshop CIR 2011
15. Chen, X. and Bennett, P. N and Collins-Thompson, K. and Horvitz, E., Pairwise ranking aggregation in a crowdsourced setting, WSDM 2013
16. Liu, C. and Wang, Y., Truelabel+ confusions: A spectrum of probabilistic models in analyzing multiple ratings, ICML 2012

References: Incremental relabeling & Pricing

17. Ipeirotis, P. G and Provost, F. and Sheng, V. S and Wang, J., Repeated labeling using multiple noisy labelers, KDD 2014
18. Abraham, I., Alonso, O., Kandylas, V., Patel, R., Shelford, S., Slivkins, A., How many workers to ask?: Adaptive exploration for collecting high quality labels, SIGIR 2016
19. Ertekin, S., Hirsh, H., Rudin, C., Learning to predict the wisdom of crowds, arXiv preprint 2012
20. Lin, C. H, Mausam, M., Weld, D. S, To Re(label), or Not To Re(label), HCOMP 2014
21. Zhao, L., Sukthankar, G., Sukthankar, R., Incremental relabeling for active learning with noisy crowdsourced annotations, PASSAT/SocialCom 2011
22. Wang, J., Ipeirotis, P. G, Provost, F., Quality-based pricing for crowdsourced workers, working paper, 2013
23. Cheng, J., Teevan, J., Bernstein, M. S, Measuring crowdsourcing effort with error-time curves, CHI 2015
24. Ho, C., Slivkins, A., Suri, S., Vaughan, J. W., Incentivizing high quality crowdwork, WWW 2015
25. Difallah, D. E., Catasta, M., Demartini, G., Cudr`e-Mauroux, P., Scaling-up the crowd: Micro-task pricing schemes for worker retention and latency improvement, HCOMP 2014
26. Yin, M., Chen, Y., Sun, Y., The effects of performance-contingent financial incentives in online labor markets, AI 2013
27. Shah, N., Zhou, D., Peres, Y., Approval voting and incentives in crowdsourcing, ICML 2015

[26] Shah, N. and Zhou, D., No oops, you won’t do it again: Mechanisms for self-correction in crowdsourcing, ICML 2016

References: Tutorials

27. Crowdsourcing: Beyond Label Generation, Vaughan, J. W. KDD 2017
28. Crowd-Powered Data Mining, Li, G., Wang, J., Fan, J., Zheng, Y., Chai, C., KDD 2018
29. Social Spam Campaigns Social Spam, Campaigns, Misinformation and Crowdturfing, Lee, K., Caverlee, J., Pu, C., WWW2014
30. From Complex Object Exploration to Complex Crowdsourcing, Amer-Yahia, S., Roy, S.B., WWW 2015
31. Crowdsourced Data Management: Overview and Challenges, Li, G., Zheng, Y., Fan, J., Wang, J., Cheng, R, SIGMOD 2017
32. Spatial Crowdsourcing: Challenges, Techniques, and Applications, Tong, Y., Chen, L., Shahab, C., VLDB 2016
33. Truth Discovery and Crowdsourcing Aggregation: A Unified Perspective, Gao, J., Li, Q., Zhao, B., Fan, W., Han, J., VLDB 2015
34. Data-Driven Crowdsourcing: Management, Mining, and Applications, Chen, L., Lee, D., Milo, T., ICDE 15