



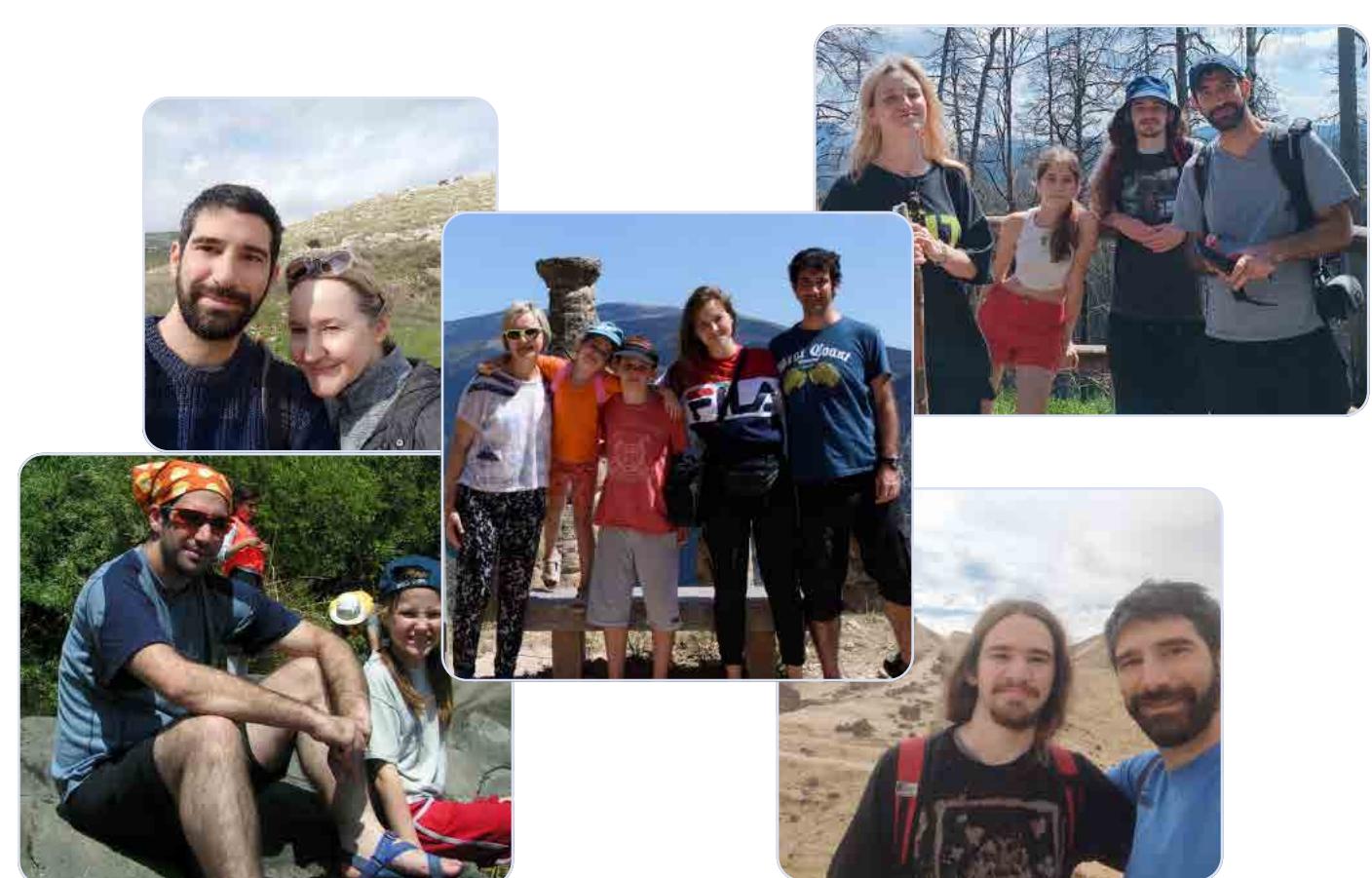
Hi, I'm **Yogev Raved**

Building end-to-end user experiences for complex systems

About me

What makes me tick

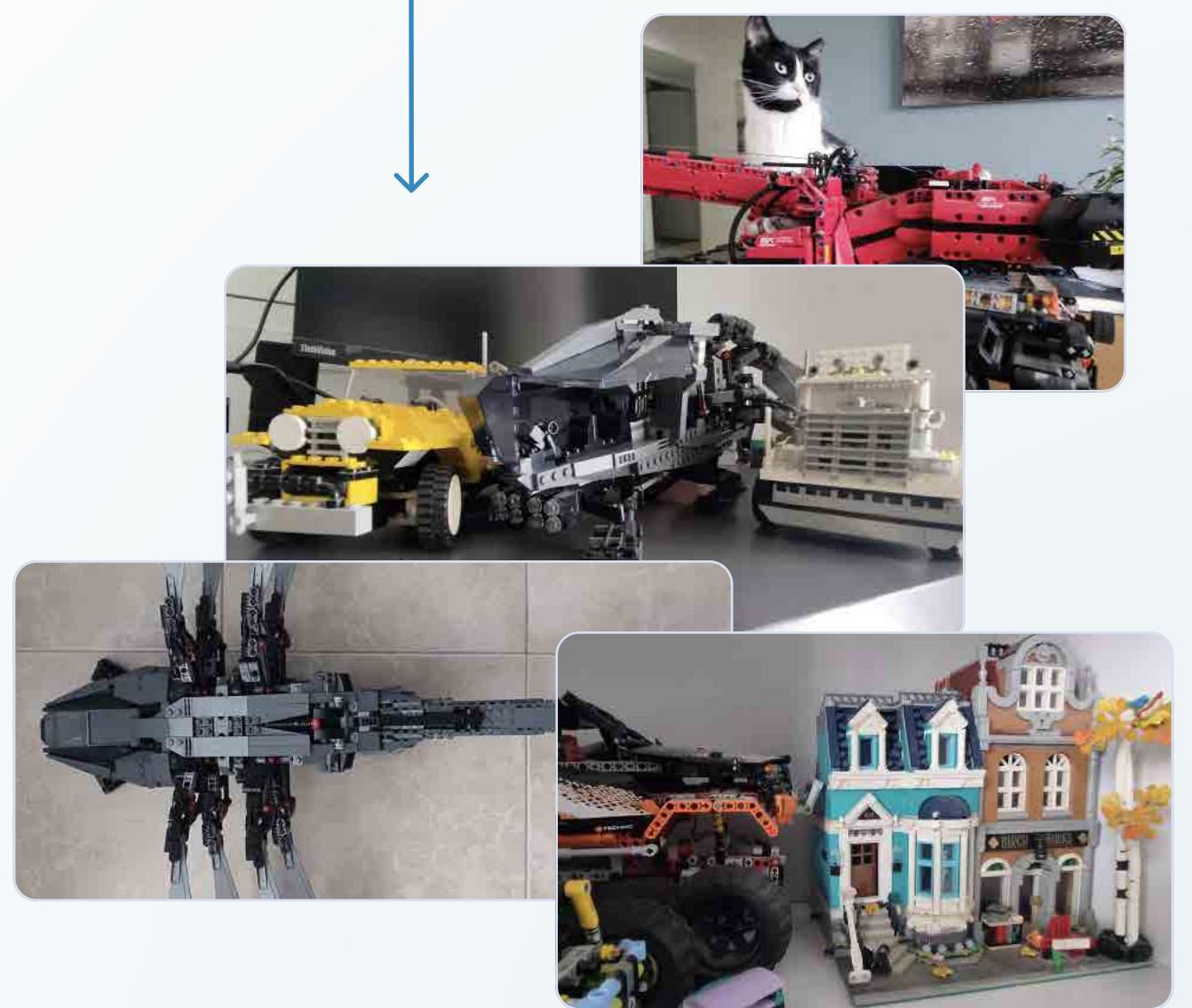
Dragging my family
to hiking trips



Thousands of wasted hours
on city building games



Geeking out
with Lego

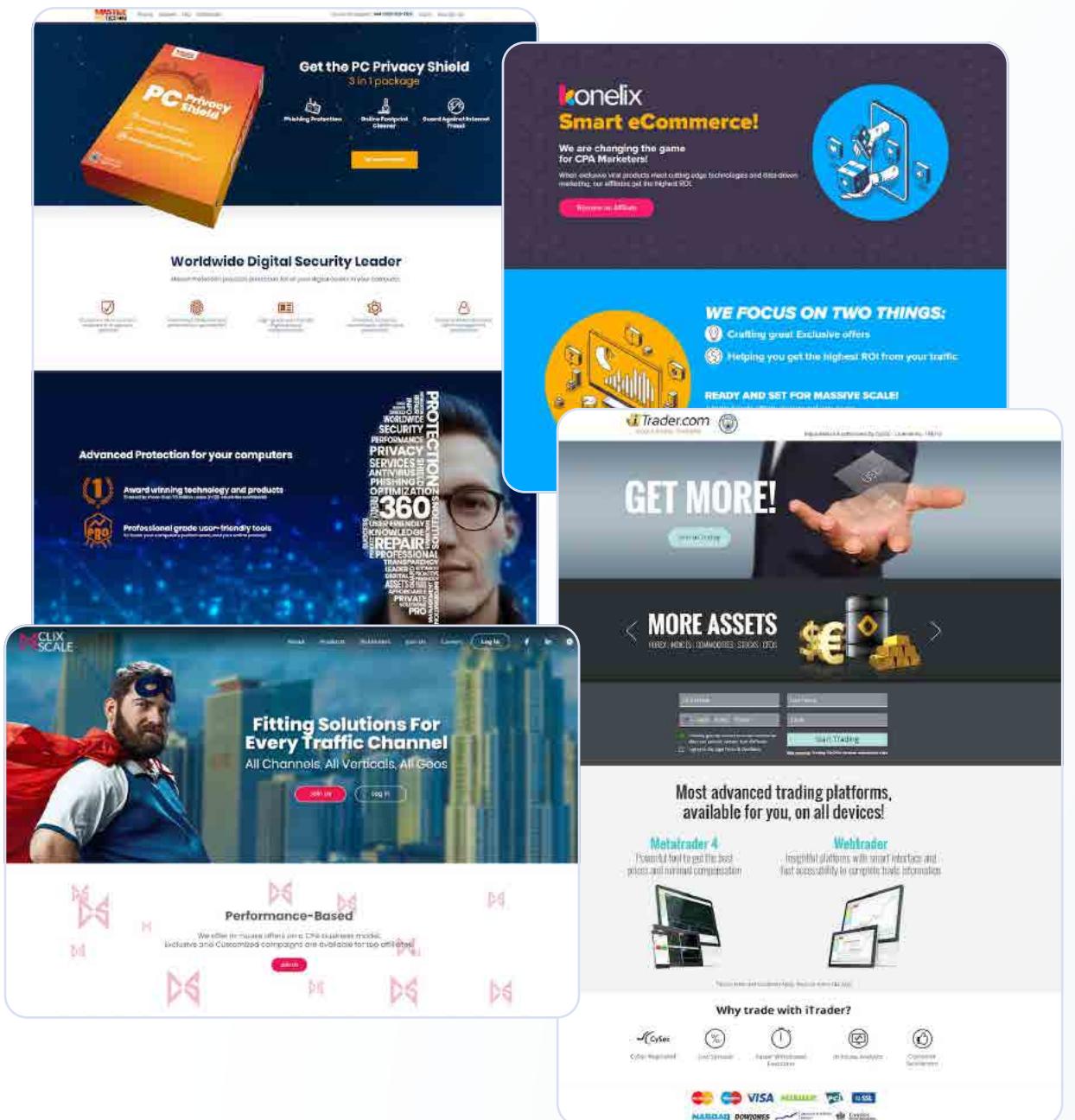


My design career path

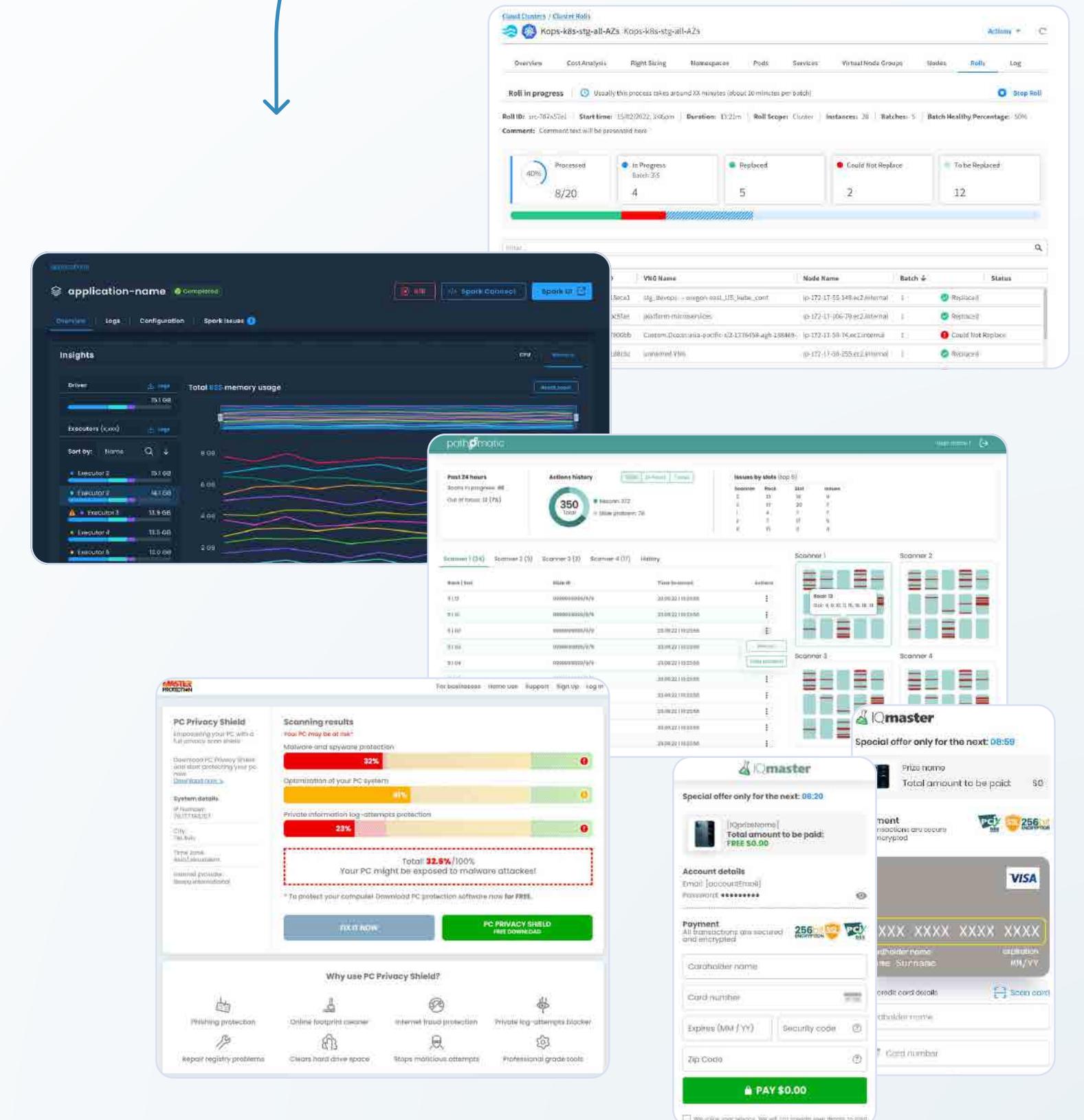
Graphic designer



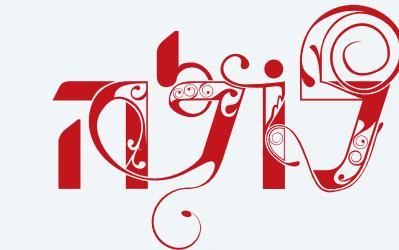
Web designer & developer



Product designer



Some logos



Every design has a story



Config-templates design revamp



Marketing campaigns management tool

The image displays a collage of five screenshots from a digital marketing dashboard, likely from a platform like SEMrush or similar, showing campaign performance and landing page examples.

- Campaign 1 (Top Left):** A landing page for "Award Winning Online Protection". It features a box for "PC Doctor" software, a "FREE" offer, and a "START YOUR TEST" button. The dashboard summary shows:
 - Campaign name:** 1001 - Campaign name
 - Clicks:** 6.97K
 - Spend:** 19K
 - CR:** 1.27%
 - Languages:** EN, ES, DE, AR, JP, FR, TR
- Campaign 2 (Top Middle):** An IQ test landing page titled "Find Out How Smart Are You!". It includes an "IQ Score Distribution" chart and a "START YOUR IQ TEST" button. The dashboard summary shows:
 - Campaign name:** 1001 - Campaign name
 - Clicks:** 6.97K
 - Spend:** 19K
 - CR:** 1.27%
 - Languages:** EN, ES, DE, AR, JP, FR, TR
- Campaign 3 (Top Right):** A landing page for "Someone might be looking at you from your Camera". It features a camera icon and a "PROTECT YOUR PRIVACY" button. The dashboard summary shows:
 - Campaign name:** 1001 - Campaign name
 - Clicks:** 6.97K
 - Spend:** 19K
 - CR:** 1.27%
 - Languages:** EN, ES, DE, AR, JP, FR, TR
- Campaign 4 (Bottom Left):** A landing page for "FREE AND UNLIMITED". It features a download progress bar and a "Download" button. The dashboard summary shows:
 - Campaign name:** 1001 - Campaign name
 - Clicks:** 6.97K
 - Spend:** 19K
 - CR:** 1.27%
 - Languages:** EN, ES, DE, AR, JP, FR, TR
- Campaign 5 (Bottom Middle):** A landing page for "Protect Your Privacy". It features a download progress bar and a "Download" button. The dashboard summary shows:
 - Campaign name:** 1001 - Campaign name
 - Clicks:** 6.97K
 - Spend:** 19K
 - CR:** 1.27%
 - Languages:** EN, ES, DE, AR, JP, FR, TR



Config-templates design revamp

Role
UX Designer
and Researcher

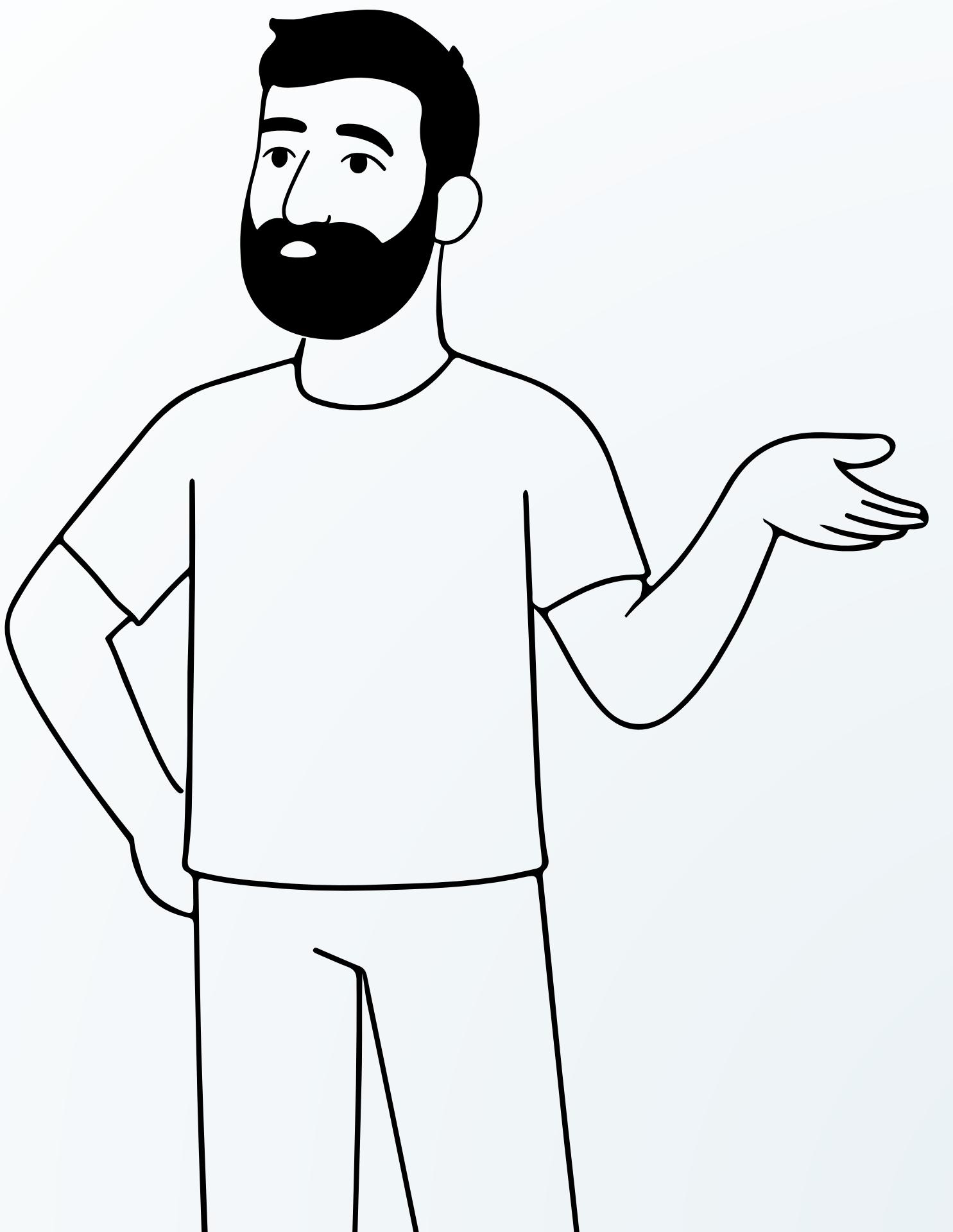
Teammates
Product Manager,
fullstack engineers

Timeline
9 month

What the #*% is it?

What are Config-templates?

- Data engineers use Apache Spark to run big-data applications.
Each type of application needs its' own configuration (in a JSON format).
- The Config-templates page is a workspace within the Spot console for storing and managing them.



What are Config-templates?

How do users use them?

- Select a template and copy its' code
- Paste it in Apache Spark environment and run the app
- Come back to Spot's console to view the app's progress, stats and issues

Motivation for the redesign

Improving usability issues of the current design:

- Template ID is visible only for the selected template.
- Viewing few amount of templates - **only 3 on small screens!** makes it hard to browse between templates.
- Adding more info to each template such as last editor, number of versions and tags will make each card much more overloaded.

The screenshot shows the 'Configuration Templates' section of the Spot by NetApp interface. On the left, there's a sidebar with links like 'Ocean For Spark', 'Applications', 'Jobs', 'Clusters', and 'Workspaces'. The 'Configuration Templates' link is highlighted. The main area has a header 'Configuration Templates' and filters for 'Cluster' (set to 'All (18)') and 'Authors' (set to 'All (48)'). Below is a table with five rows of template cards. Each card contains the template name, cluster name, author, and a 'Template ID' column. To the right of the table is a vertical panel displaying a JSON configuration file with lines 1 through 10 visible. The JSON content includes details about a Python template with sparkVersion 3.2.1 and dynamic allocation configurations.

Template ID	alex-ofas	Cluster Name	alex-ofas-cluster-15	Author	Alex Tarasov ...
alex-ofas	asteroid	Cluster Name	asteroid	Author	Sigmar Karl
	awendlinger-notebooks	Cluster Name	awendlinger-tobys	Author	Antoine Wen...
	basic-notebook	Cluster Name	clement-aws-dev-2	Author	Clement Rez...
	buggun				

```
1 {  
2   "type": "Python",  
3   "sparkVersion": "3.2.1",  
4   "sparkConf": {  
5     "spark.dynamicAllocation.enabled": "true",  
6     "spark.dynamicAllocation.maxExecutors": "10",  
7     "spark.dynamicAllocation.minExecutors": "0",  
8     "spark.dynamicAllocation.initialExecutors": "  
9   }  
10 }
```

The goal

What do we want to achieve?



Reduce time spent on troubleshooting

Increase user satisfaction



Reduce support calls

Save time and money for the customers and for our organization



More users per team

Increase product value



Help selling the product to more clients

More revenue to our organization



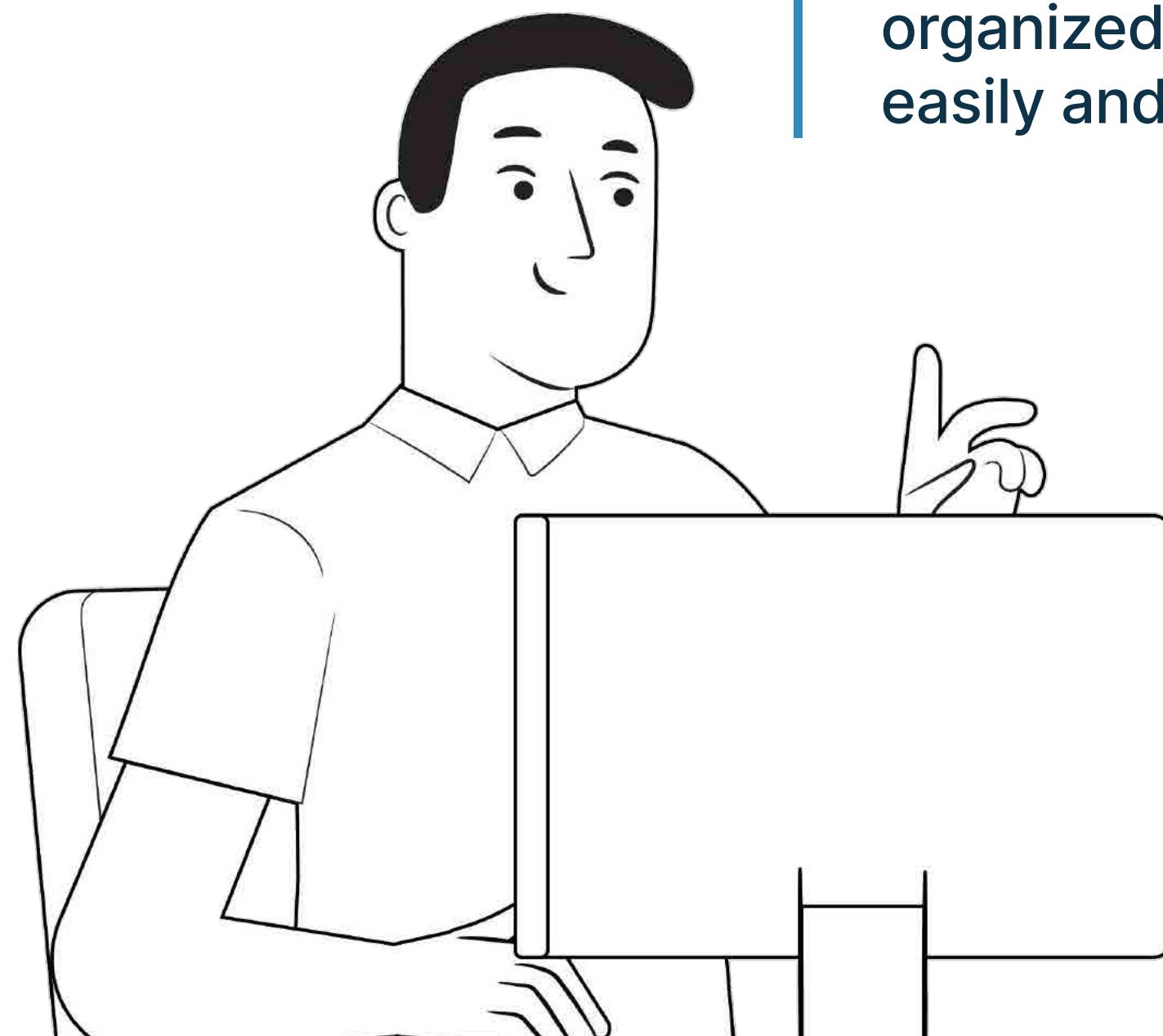
Research

Who are the users?

Data engineer

Designs, builds, and optimizes data pipelines on Apache Spark, ensuring efficient data ingestion, transformation, and processing for analytics and machine learning applications.

“I need things to be very organized so I can find what I need easily and focus on my tasks”



Data Platform Engineer (DevOps)

Develops and maintains the infrastructure for running Apache Spark applications, managing scalability, performance, and reliability to support seamless data operations.

“I need an option to view what changes were made in the templates over time and the reasons they were made”



Reaching out to the users

Why?

- To make sure our vision for the product is relevant
- Validate assumptions

How?

- FullStory
- Surveys
- Interviews

Reaching out to the users

Surveys

- Open-ended and closed questions
- Features rating and ideas ranking
- Gained qualitative and quantitative insights

Please rate how much you find useful these following features:

1 Useless 2 Somewhat useful 3 Useful 4 Extremely useful

	Comparing 2 different configuration templates	History, change log and revert options	Ability to add tags on configuration templates	Have starter templates when creating a new template	Be able to edit a configuration templates from the app page
Customer A	3	4	4	3	3
Customer B	3	4	4	3	3
Customer C	3	4	3	4	2
Customer D	4	3	2	3	3
Customer E	3	3	4	3	2
Customer F	4	4	4	4	4
Average	3.3	3.6	3.5	3.3	2.8

Reaching out to the users

Customer & Stakeholder interviews

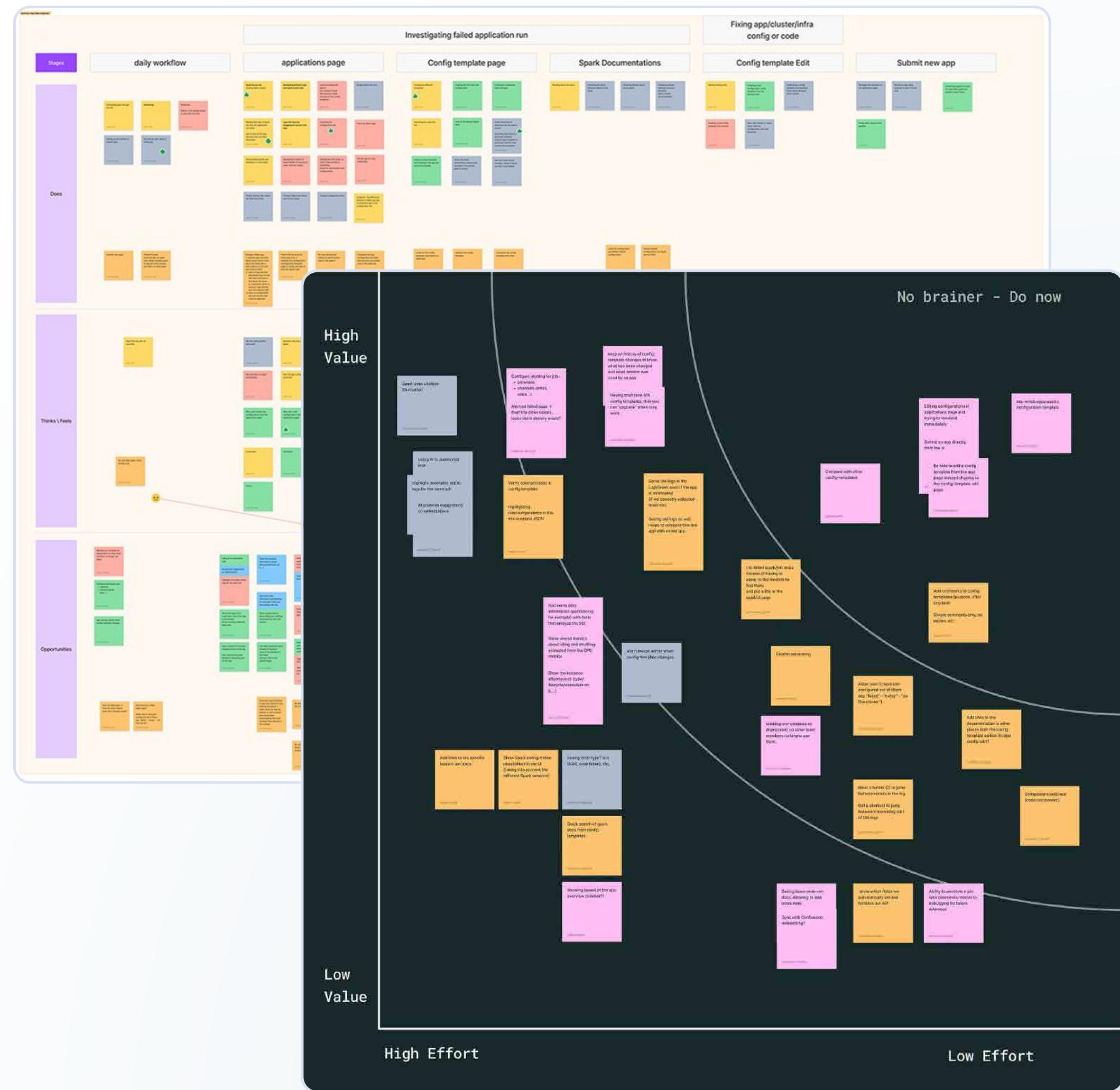
- Explored their daily workflow for creating and maintaining config-templates
- Examined integration with existing tools and processes
- Assessed work aspects both within and outside our product
- Analyzed current usage patterns and desired capabilities
- Studied user interactions and workflows
- Uncovered key pain points and frictions
- Identified short-term and long-term objectives



Research

User Journey Mapping

- Collaborative workshops with cross-functional teams
- Identified key interaction points and friction areas
- Prioritized problem spaces
- Created alignment on solution requirements



Key Research Findings

"As a data engineer I want to:

- Filter and sort templates based on various criteria so that I can focus on relevant items.
- Organize and filter config-templates more effectively.
- Group templates by different attributes so that I can better organize my view.
- Perform bulk actions (like editing tags or deleting) on multiple templates at once so that I can manage them more efficiently.
- Compare templates side by side so that I can easily identify differences for debugging."

Insights

1. Users strongly desired version control and change history
2. Template organization was a major pain point: Need for logical grouping and categorization and no way to track template purposes, and uses per application
3. Search and filtering capabilities were crucial for large template libraries

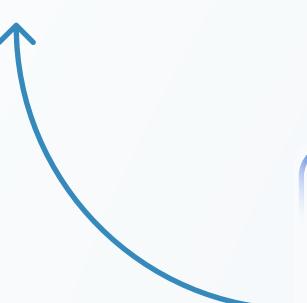


Ideation

Design solution

So, we had to do some prioritization based on development capabilities

- Add version history and changelog option
- Tagging system - This feature did not exist in the Spot console, and deciding on its behavior was challenging
- Advanced search and filtering
- Template comparison tools
- Applications running on config-templates list
- Presets for templates with common configurations
- Bulk actions for template management



The one page improvement task evolved into an enhanced design and development project...

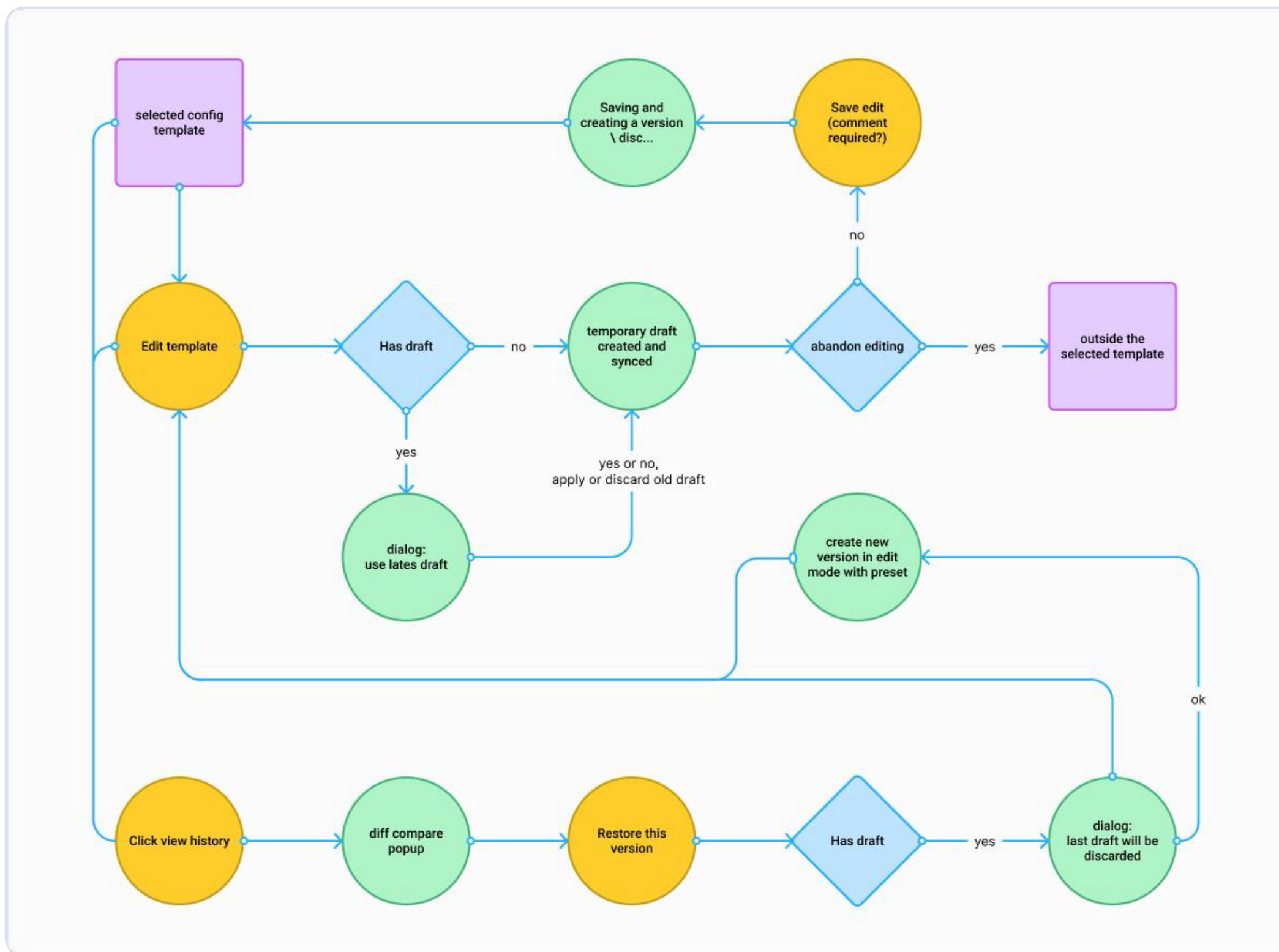
Design process

How should we approach each feature's design

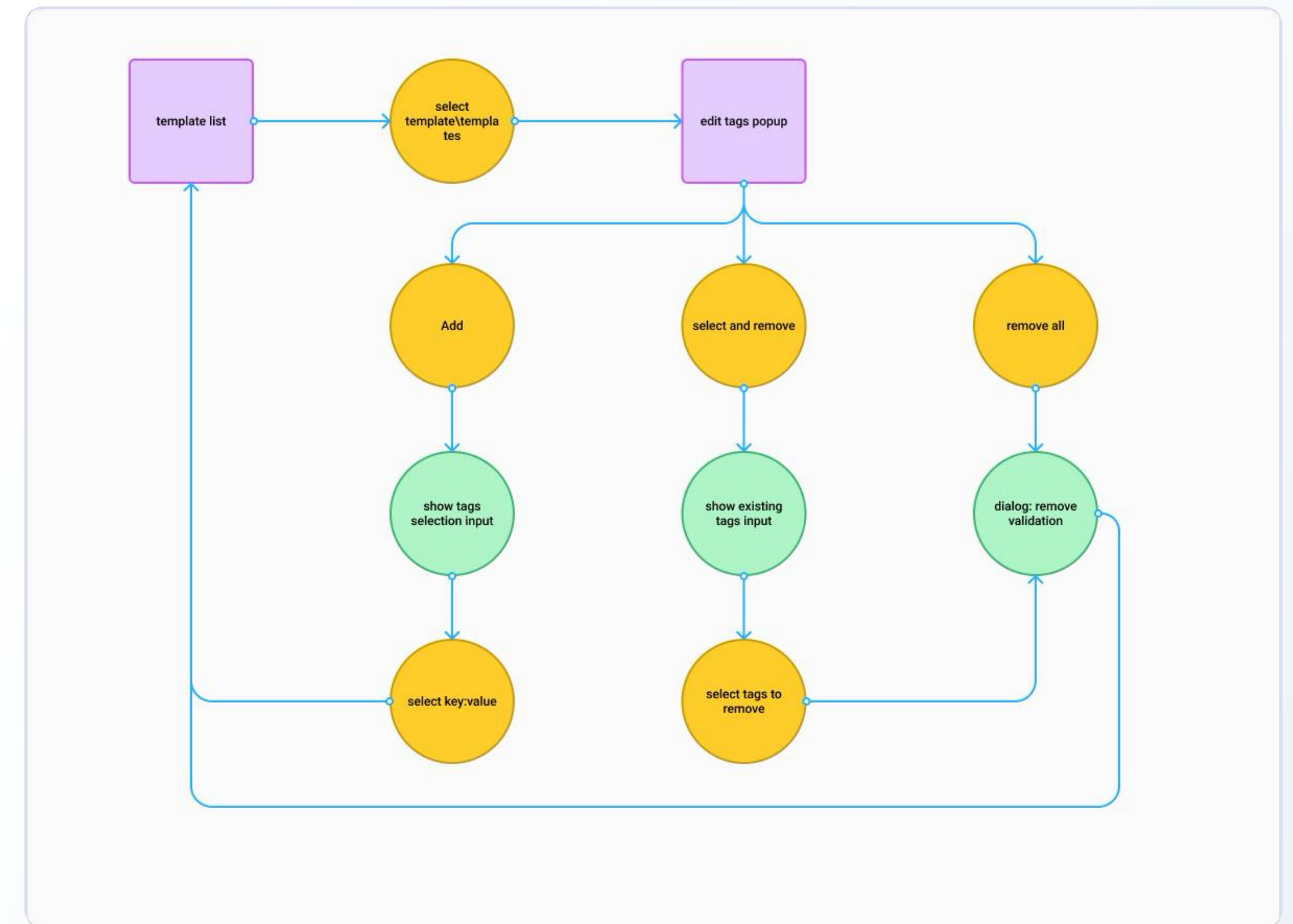
- Analyzed existing design system components
- Researched applicable design patterns
- Should we add a draft mechanism for unfinished editing
- Should commenting on a changelog be mandatory
- Mechanism of the tagging system
- Low fidelity wireframes on the page layout
- Original Cards layout vs a new table design
- Working with tags
- History view - compare and changelog
- Presets - custom and built in use cases

User flows

Editing



Tagging Process



Exploring layout directions

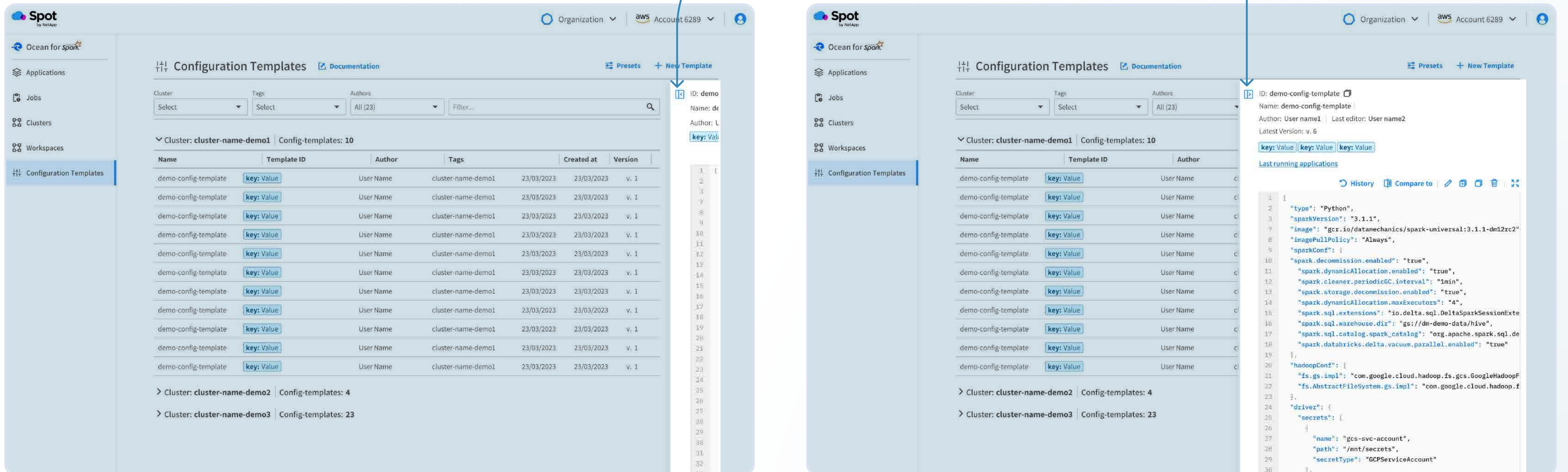
- The new features were added to the cards design.
Several attempts and iterations of various layouts were made, such as wider cards to hold more data, an expanding panel and a table.
- Several users' interviews and testing were conducted with internal users and with customers.
- The testers were presented with the various layouts, and were asked to perform tasks such as filtering, sorting and performing bulk actions.

The figure consists of six screenshots of a software application's 'Configuration Templates' section, illustrating various layout directions:

- Screenshot 1:** Shows a card-based layout where each template is represented by a card with its name, type, and a snippet of configuration code.
- Screenshot 2:** Shows a card-based layout similar to the first one, but with a different color scheme and slightly different card design.
- Screenshot 3:** Shows a card-based layout with a more complex card design, possibly including a summary table or a larger preview area.
- Screenshot 4:** Shows a table-based layout where templates are listed in a grid. The columns include Name, Template ID, Author, Label, Created at, and Version.
- Screenshot 5:** Shows a table-based layout similar to the fourth one, but with a different set of columns and a more detailed configuration snippet displayed.
- Screenshot 6:** Shows a table-based layout with a very detailed configuration snippet displayed in a large text area, likely a modal or a expanded view of a specific template.

Exploring layout directions

Expanded panel



By default: The template's content is collapsed

Clicking the expand icon
expand the panel over
the config-templates list

User research

Exploring layout directions

Original cards

The screenshot shows a card-based interface for managing configuration templates. At the top, there are filters for Tags (Select), Cluster (All (2)), Authors (All (23)), and a search bar. Below the filters, a section titled "Latest Version: v. 6" displays a configuration file with line numbers. To the left of the file, there are four collapsed card components:

- ID: be-basic-notebook-stg**
Author: Samantha Smith
Name: be-basic-notebook-stg
Cluster: demo-cluster-config-template
Env: Staging Team: Backend Project: Flying Toothbrush
- ID: spark-connect-dev**
Author: Woody Evered
Cluster: cluster-config-template
Env: Dev Team: Frontend Project: Hail Mary
- ID: be-basic-notebook-pr**
Author: Allana Jerome
Cluster: Cluster-name-1
Env: Prod Team: Backend Project: Flying Toothbrush
- ID: HM-basic-notebook-1**
Author: Woody Evered
Cluster: cluster-config-template
key: Staging Team: Infra Team: Backend Project: Hail Mary

Below the cards is a large code editor window showing the JSON configuration file.

Table layout

The screenshot shows a table-based interface for managing configuration templates. At the top, there are buttons for "Show Tags" (selected), "Select 2 templates for comparison", "Compare selected", "Edit tags", and a search bar. Below the search bar is a "Filter..." input and a "Group by:" dropdown set to "Default".

The main area is a table with columns: Template ID, Name, Author, and Last Editor. The table contains eight rows, each representing a configuration template with its details and a "key: Value" button. To the right of the table, the "Latest Version: v. 6" configuration file is displayed in its entirety, showing a large JSON object with many nested properties and values.

Exploring layout directions

New features on cards layout

The screenshot displays the Configuration Templates interface in two states, connected by a large blue curved arrow.

Left Card (Initial State): Shows a single template selected. A blue arrow points from the text "Selecting a template opens a modal with options" to the modal overlay on the template card. The modal contains buttons for "Edit tags", "Compare To", and "Delete".

Right Card (Final State): Shows two templates selected. A blue arrow points from the text "Selecting 2 items enables comparing" to the "Compare selected" button at the bottom of the card. The card also includes buttons for "Edit Tags" and "Delete Template(s)".

Template Details:

```
Latest Version: v. 6
Group by: Default
History Compare To Edit tags
ID: be-basic-notebook-stg
Author: Samantha Smith
Name: be-basic-notebook-stg
Cluster: demo-cluster-config-template
Env: Staging Team: Backend Project: Flying Toothbrush
ID: spark-connect-dev
Author: Woody Evered
Cluster: cluster-config-template
Env: Dev Team: Frontend Project: Hail Mary
ID: be-basic-notebook-pr
Author: Allana Jerome
Cluster: Cluster-name-1
Env: Prod Team: Backend Project: Flying Toothbrush
ID: HM-basic-notebook-1
Author: Woody Evered
Cluster: cluster-config-template
key: Staging Team: Infra Project: Hail Mary
```

Selected Templates (Right Card):

```
Latest Version: v. 6
Group by: Default
History Compare To Edit tags
ID: be-basic-notebook-stg
Author: Samantha Smith
Name: be-basic-notebook-stg
Cluster: demo-cluster-config-template
Env: Staging Team: Backend Project: Flying Toothbrush
ID: spark-connect-dev
Author: Woody Evered
Cluster: cluster-config-template
Env: Dev Team: Frontend Project: Hail Mary
ID: be-basic-notebook-pr
Author: Allana Jerome
Cluster: Cluster-name-1
Env: Prod Team: Backend Project: Flying Toothbrush
ID: HM-basic-notebook-1
Author: Woody Evered
Cluster: cluster-config-template
key: Staging Team: Infra Project: Hail Mary
```

Buttons (Right Card):

- Clear Selection
- Edit Tags
- Delete Template(s)
- Compare selected
- Select 2 templates for comparison

Cards design

User testing insights

Cards design - Actions popup

Issue:

Placement of bulk actions buttons outside this area could be confusing.

Solution:

Actions popup - Triggered by checking an item.

Pros:

Saves space on the page.

Cons:

No indication for the actions without checking an item.

The screenshot shows a user interface for managing configuration templates. At the top, there are filters for Tags (Select), Cluster (All (2)), and Authors (All (23)). Below these are buttons for Presets and New Template. The main area displays a list of templates, each with a checkbox, ID, author, name, cluster, and environment details. An 'Actions' button is visible at the bottom right of each card. To the right of the list, a code editor shows the JSON configuration for the selected template, which includes fields like type, sparkVersion, image, and sparkConf. A sidebar on the right shows 'Last running applications' with icons for different services.

```
1 "type": "Python",
2 "sparkVersion": "3.1.1",
3 "image": "gcr.io/datamechanics/spark-universal:3.1.1-dm12rc2",
4 "imagePullPolicy": "Always",
5 "sparkConf": [
6     "spark.decommission.enabled": "true",
7     "spark.dynamicAllocation.enabled": "true",
8     "spark.cleaner.periodicGC.interval": "1min",
9     "spark.storage.decommission.enabled": "true",
10    "spark.dynamicAllocation.maxExecutors": "4",
11    "spark.sql.extensions": "io.delta.sql.DeltaSparkSessionExtension",
12    "spark.sql.warehouse.dir": "gs://ds-demo-data/hive",
13    "spark.sql.catalog.spark_catalog": "org.apache.spark.sql.delta.catalog.DeltaCatalog",
14    "spark.databricks.delta.vacuum.parallel.enabled": "true"
15 ],
16 "hadoopConf": [
17     "fs.gs.impl": "com.google.cloud.hadoop.fs.gcs.GoogleHadoopFileSystem",
18     "fs.AbstractFileSystem.gs.impl": "com.google.cloud.hadoop.fs.gcs.GoogleHadoopFS"
19 ],
20 "driver": [
21     {
22         "secrets": [
23             {
24                 "name": "gcs-svc-account",
25                 "path": "/mnt/secrets"
26             }
27         ]
28     }
29 ]
```

Cards design

User testing insights

Strength:

- Intuitive grouping by attribute - Tags, Author, Cluster.
- Template view is wide from the start.

Weaknesses:

- Sorting is limited.
- Only a few templates are on screen.
- Added more info to each template makes the cards much more overloaded and harder to browse.

Example: Grouped by Tag "Env", collapsed and expanded

Checking an item triggers actions popup (compare, edit tags, delete), that can be performed in bulk

"Show more" icon for high amount tags

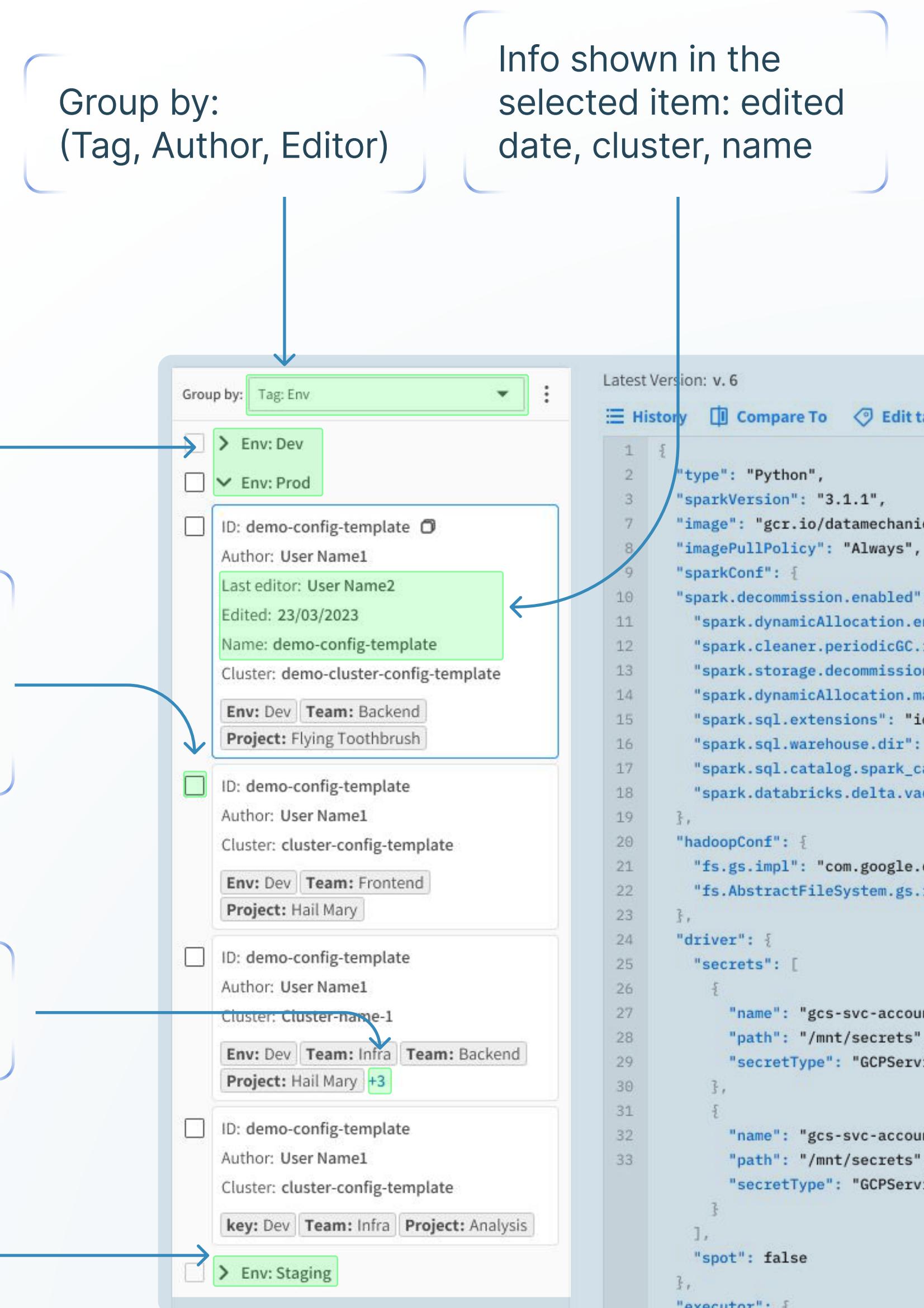


Table design

User testing insights

Configuration Templates [Documentation](#)

Filter... Search icon

Show Tags | Select 2 templates for comparison Compare selected Edit tags Delete

Drag column headings here to group Clusters or Authors

Template ID	Name	Author	Last Editor	Cluster	Created	Last	Actions
Config-template id	Config-template-name	User Name 1	User Name 2	cluster-name-demo1	23/03/2023	23/03/2023	key: Value key: Value key: Value
Config-template id	Config-template-name	User Name 1	User Name 2	cluster-name-demo1	23/03/2023	23/03/2023	key: Value key: Value key: Value key: Value
Config-template id	Config-template-name	User Name 1	User Name 2	cluster-name-demo1	23/03/2023	23/03/2023	key: Value key: Value key: Value
Config-template id	Config-template-name	User Name 1	User Name 2	cluster-name-demo1	23/03/2023	23/03/2023	key: Value key: Value key: Value
Config-template id	Config-template-name	User Name 1	User Name 2	cluster-name-demo1	23/03/2023	23/03/2023	key: Value key: Value key: Value key: Value
Config-template id	Config-template-name	User Name 1	User Name 2	cluster-name-demo1	23/03/2023	23/03/2023	key: Value key: Value key: Value
Config-template id	Config-template-name	User Name 1	User Name 2	cluster-name-demo1	23/03/2023	23/03/2023	key: Value key: Value key: Value key: Value
Config-template id	Config-template-name	User Name 1	User Name 2	cluster-name-demo1	23/03/2023	23/03/2023	key: Value key: Value key: Value
Config-template id	Config-template-name	User Name 1	User Name 2	cluster-name-demo1	23/03/2023	23/03/2023	key: Value key: Value key: Value key: Value
Config-template id	Config-template-name	User Name 1	User Name 2	cluster-name-demo1	23/03/2023	23/03/2023	key: Value key: Value key: Value
Config-template id	Config-template-name	User Name 1	User Name 2	cluster-name-demo1	23/03/2023	23/03/2023	key: Value

Total: 23

Presets + New Template

Latest Version: v. 6 Last running applications

History Compare To Edit tags Edit Copy Delete More

```
1 {  
2   "type": "Python",  
3   "sparkVersion": "3.1.1",  
4   "image": "gcr.io/datamechanics/spark-universal:3.1.1-dm12rc2",  
5   "imagePullPolicy": "Always",  
6   "sparkConf": {}  
7   "spark.decommission.enabled": "true",  
8   "spark.dynamicAllocation.enabled": "true",  
9   "spark.cleaner.periodicGC.interval": "1min",  
10  "spark.storage.decommission.enabled": "true",  
11  "spark.dynamicAllocation.maxExecutors": "4",  
12  "spark.sql.extensions": "io.delta.sql.DeltaSparkSessionExtension",  
13  "spark.sql.warehouse.dir": "gs://dm-demo-data/hive",  
14  "spark.sql.catalog.spark_catalog": "org.apache.spark.sql.delta.catalog.DeltaCatalog",  
15  "spark.databricks.delta.vacuum.parallel.enabled": "true"  
16 },  
17 "hadoopConf": {}  
18   "fs.gs.impl": "com.google.cloud.hadoop.fs.gcs.GoogleHadoopFileSystem",  
19   "fs.AbstractFileSystem.gs.impl": "com.google.cloud.hadoop.fs.gcs.GoogleHadoopAbstractFileSystem",  
20 },  
21 "driver": {}  
22   "secrets": [  
23     {  
24       "name": "gcs-svc-account",  
25       "path": "/mnt/secrets",  
26       "secretType": "GCPServiceAccount"  
27     },  
28     {  
29       "name": "gcs-svc-account",  
30       "path": "/mnt/secrets",  
31       "secretType": "GCPServiceAccount"  
32     },  
33   ],  
34   "spot": false  
35 },  
36 "executor": {}  
37   "secrets": [  
38     {  
39       "name": "gcs-svc-account",  
40       "path": "/mnt/secrets",  
41       "secretType": "GCPServiceAccount"  
42     },  
43   ]  
44 }  
45 }
```

Table design

User testing insights

Group by column header - Tag, Author, Editor

Bulk selection for actions - compare, edit tags, delete

Total amount of templates

Tags row allow showing many items

Easy sorting, filtering and grouping

Expand the preview to full screen

Custom columns to arrange and to filter data

User testing insights

Strength:

- Sortable by multiple parameters (ID, name, cluster, author, date).
- No need to hide config-template info.
- Easier template browsing.

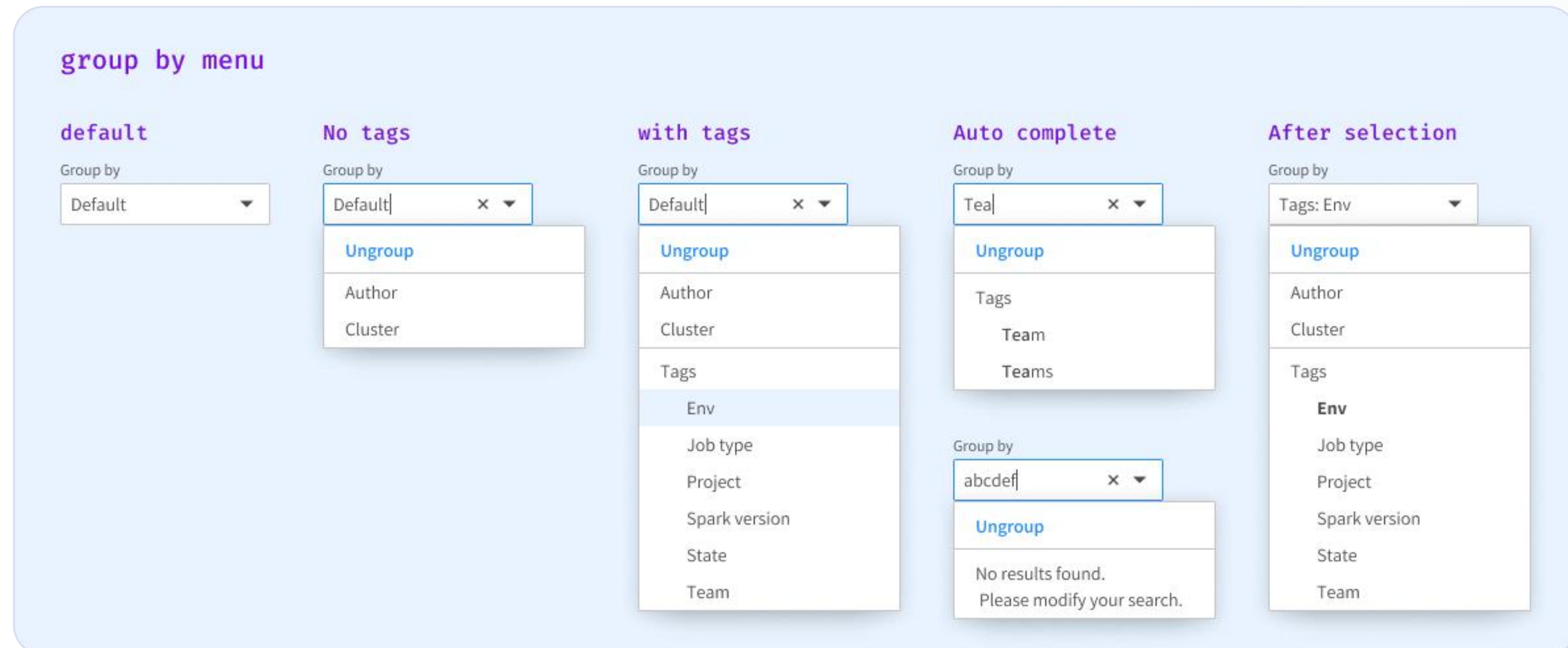
Weaknesses:

- Less initial space for templates (solved with full-screen option).
- Grouping limited to column headers, missing tag-based grouping (design system).

User testing insights

Grouping solution

Instead of the design system's grouping option I switched to a simple “Group by” selection





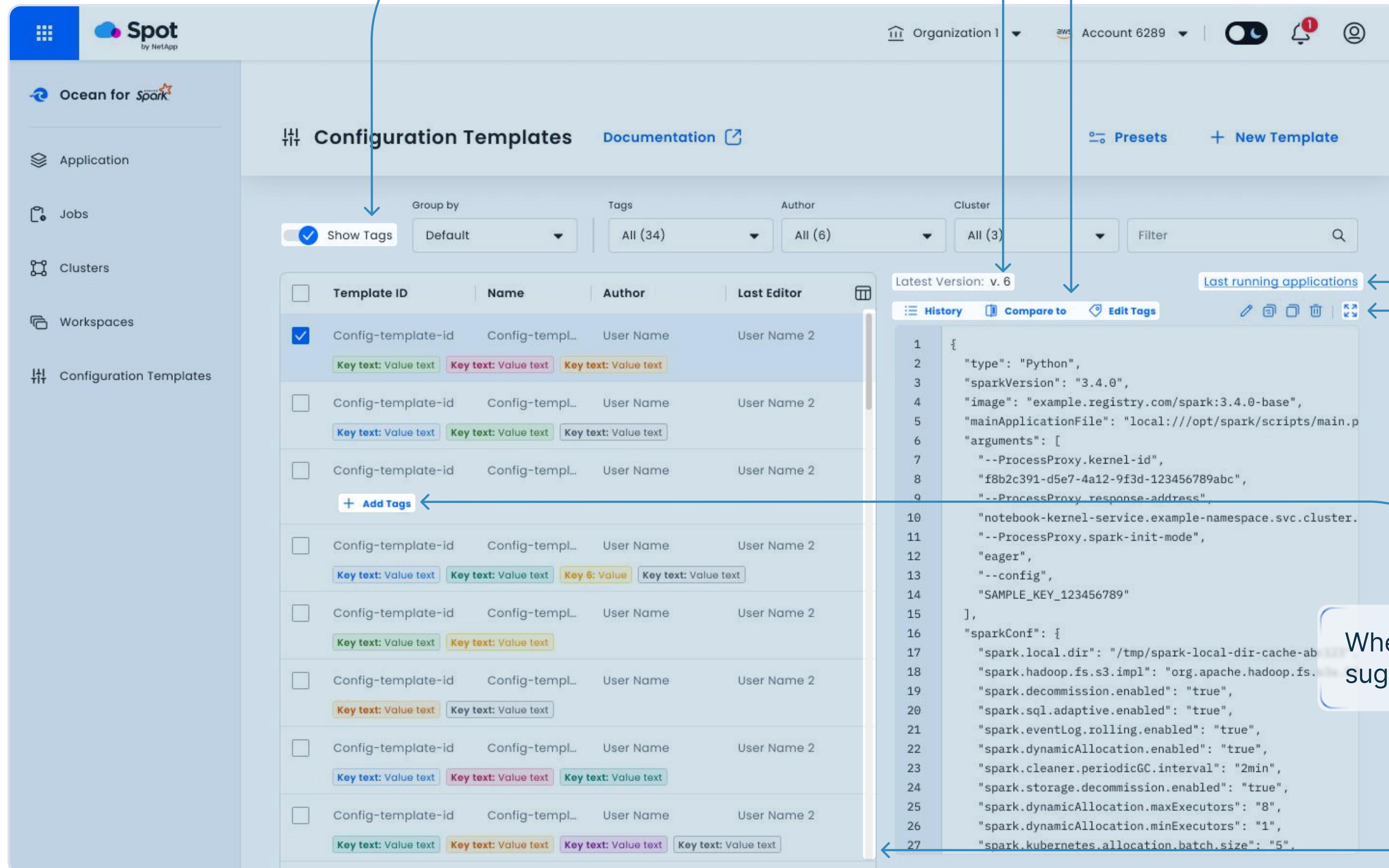
Final design & New UI

Main page

The screenshot shows the Spot by NetApp interface. The left sidebar includes links for Ocean for Spark, Application, Jobs, Clusters, Workspaces, and Configuration Templates. The main area is titled "Configuration Templates" and displays a table of templates. The table columns are Template ID, Name, Author, and Last Editor. A "Show Tags" button is checked, and dropdown menus show Tags (All (34)), Author (All (6)), and Cluster (All (3)). A search bar and a "Presets" button are also present. On the right, a preview window shows the latest version (v. 6) of a template, with code starting with:

```
1 {  
2   "type": "Python",  
3   "sparkVersion": "3.4.0",  
4   "image": "example.registry.com/spark:3.4.0-base",  
5   "mainApplicationFile": "local:///opt/spark/scripts/main.p  
6   "arguments": [  
7     "--ProcessProxy.kernel-id",  
8     "f8b2c391-d5e7-4a12-9f3d-123456789abc",  
9     "--ProcessProxy.response-address",  
10    "notebook-kernel-service.example-namespace.svc.cluster.  
11    --ProcessProxy.spark-init-mode",  
12    "eager",  
13    "--config",  
14    "SAMPLE_KEY_123456789"  
15  ],  
16  "sparkConf": {  
17    "spark.local.dir": "/tmp/spark-local-dir-cache-abc123",  
18    "spark.hadoop.fs.s3.impl": "org.apache.hadoop.fs.s3a.S3  
19    "spark.decommission.enabled": "true",  
20    "spark.sql.adaptive.enabled": "true",  
21    "spark.eventLog.rolling.enabled": "true",  
22    "spark.dynamicAllocation.enabled": "true",  
23    "spark.cleaner.periodicGC.interval": "2min",  
24    "spark.storage.decommission.enabled": "true",  
25    "spark.dynamicAllocation.maxExecutors": "8",  
26    "spark.dynamicAllocation.minExecutors": "1",  
27    "spark.kubernetes.allocation.batch.size": "5",
```

Main page



Option to hide the tags row

Version No.

- Selected item actions

Template run apps list

Expand to full screen

When no tags, the user is suggested to add some

Infinite scroll

Empty state

Configuration Templates Documentation ↗ Presets + New Template



Configuration templates are stored fragments of Spark application configuration, for when you need to share a large default configuration between multiple applications, or to avoid storing configurations on your side. [Learn more](#)

You can create template from a Preset or start from scratch.

Presets + New Template

Panel update on selection

Enabled when two items are selected

The diagram illustrates a user interface flow for managing configuration templates. It consists of two main panels:

- Left Panel:** A list of configuration templates. One template is selected (indicated by a checked checkbox). The selected template's details are displayed in a modal overlay titled "Latest Version: v. 6". The modal shows the JSON configuration code for the selected template.
- Right Panel:** A list of configuration templates. Two templates are selected (indicated by checked checkboxes). The selected templates' details are displayed in a modal overlay titled "Last running applications". The modal shows the JSON configuration code for the selected templates.

A large blue arrow originates from the selected item in the left panel's list and points to the selected items in the right panel's list, indicating that the right panel's content is updated based on the selection made in the left panel.

Left Panel Details:

- Template ID: Config-template-id
- Name: Config-templ...
- Author: User Name
- Last Editor: User Name 2
- Cluster: cluster-name-demo
- Key text: Value text
- Key text: Value text
- Key text: Value text

Right Panel Details:

- Template ID: Config-template-id
- Name: Config-templ...
- Author: User Name
- Last Editor: User Name 2
- Cluster: cluster-name-demo
- Key text: Value text
- Key text: Value text
- Key text: Value text

Modal Content (Left Panel):

```
1 {
  "type": "Python",
  "sparkVersion": "3.4.0",
  "image": "example.registry.com/spark:3.4.0-base",
  "mainApplicationFile": "local:///opt/spark/scripts/main.py",
  "arguments": [
    "-ProcessProxy.kernel-id",
    "f8b2c391-d5e7-4a12-9f3d-123456789abc",
    "-ProcessProxy.response-address",
    "notebook-kernel-service.example-namespace.svc.cluster.",
    "-ProcessProxy.spark-init-mode",
    "eager",
    "--config",
    "SAMPLE_KEY_123456789"
  ],
  "sparkConf": {
    "spark.local.dir": "/tmp/spark-local-dir-cache-abc123",
    "spark.hadoop.fs.s3.impl": "org.apache.hadoop.fs.s3a.S3",
    "spark.decommission.enabled": "true",
    "spark.sql.adaptive.enabled": "true",
    "spark.eventLog.rolling.enabled": "true",
    "spark.dynamicAllocation.enabled": "true",
    "spark.cleaner.periodicGC.interval": "2min",
    "spark.storage.decommission.enabled": "true",
    "spark.dynamicAllocation.maxExecutors": "8",
    "spark.dynamicAllocation.minExecutors": "1",
    "spark.kubernetes.allocation.batch.size": "5",
    "spark.kubernetes.driver.requestTimeout": "25000",
    "spark.dynamicAllocation.initialExecutors": "2",
    "spark.sql.execution.arrow.spark.enabled": "true",
    "spark.kubernetes.driver.connectionTimeout": "25000",
    "spark.sql.execution.arrow.pyspark.enabled": "true",
    "spark.storage.decommission.rddBlocks.enabled": "true",
    "spark.dynamicAllocation.executorAllocationRatio": "0.5",
    "spark.dynamicAllocation.shuffleTracking.enabled": "true",
    "spark.storage.decommission.shuffleBlocks.enabled": "true",
    "spark.kubernetes.allocation.driver.readinessTimeout": "25000",
    "spark.dynamicAllocation.sustainedSchedulerBacklogTimeout": "10000",
    "spark.hadoop.mapreduce.fileoutputcommitter.algorithm.version": "1"
}
```

Total: 14

Modal Content (Right Panel):

```
1 {
  "type": "Python",
  "sparkVersion": "3.4.0",
  "image": "example.registry.com/spark:3.4.0-base",
  "mainApplicationFile": "local:///opt/spark/scripts/main.py",
  "arguments": [
    "-ProcessProxy.kernel-id",
    "f8b2c391-d5e7-4a12-9f3d-123456789abc",
    "-ProcessProxy.response-address",
    "notebook-kernel-service.example-namespace.svc.cluster.",
    "-ProcessProxy.spark-init-mode",
    "eager",
    "--config",
    "SAMPLE_KEY_123456789"
  ],
  "sparkConf": {
    "spark.local.dir": "/tmp/spark-local-dir-cache-abc123",
    "spark.hadoop.fs.s3.impl": "org.apache.hadoop.fs.s3a.S3",
    "spark.decommission.enabled": "true",
    "spark.sql.adaptive.enabled": "true",
    "spark.eventLog.rolling.enabled": "true",
    "spark.dynamicAllocation.enabled": "true",
    "spark.cleaner.periodicGC.interval": "2min",
    "spark.storage.decommission.enabled": "true",
    "spark.dynamicAllocation.maxExecutors": "8",
    "spark.dynamicAllocation.minExecutors": "1",
    "spark.kubernetes.allocation.batch.size": "5",
    "spark.kubernetes.driver.requestTimeout": "25000",
    "spark.dynamicAllocation.initialExecutors": "2",
    "spark.sql.execution.arrow.spark.enabled": "true",
    "spark.kubernetes.driver.connectionTimeout": "25000",
    "spark.sql.execution.arrow.pyspark.enabled": "true",
    "spark.storage.decommission.rddBlocks.enabled": "true",
    "spark.dynamicAllocation.executorAllocationRatio": "0.5",
    "spark.dynamicAllocation.shuffleTracking.enabled": "true",
    "spark.storage.decommission.shuffleBlocks.enabled": "true",
    "spark.kubernetes.allocation.driver.readinessTimeout": "25000",
    "spark.dynamicAllocation.sustainedSchedulerBacklogTimeout": "10000",
    "spark.hadoop.mapreduce.fileoutputcommitter.algorithm.version": "1"
}
```

Total: 14

Right Panel Buttons:

- Compare selected
- Edit tags
- Delete templates

Right Panel Icons:

- Two overlapping blue tags icon

New template

The image displays three sequential screenshots of a 'New Config-template' dialog box, illustrating the workflow for creating a new configuration template.

Screenshot 1: Start from a preset

This screen shows the initial step of creating a new template. It includes:

- A section titled "Start from a preset" with two radio button options: "Scenarios" (selected) and "Custom Presets".
- A dropdown menu labeled "Select a preset".
- A section titled "Duplicate a config-template" with a dropdown menu labeled "Select a Config-template".
- A section titled "Start from scratch".
- Buttons at the bottom: "Cancel" (outline) and "Continue" (solid blue).

Screenshot 2: Progressing through the steps

The user has selected "Scenarios" and chosen a preset. The "Select a Config-template" dropdown is now highlighted in blue, indicating it is the active or next step.

Screenshot 3: Final step - Start from scratch

The "Select a Config-template" dropdown is no longer visible, suggesting it has been completed. The "Start from scratch" section is now highlighted in blue, indicating it is the final step in this sequence.

Presets

Configuration Templates / Presets

Presets

Search a preset

New Template

Presets are reusable code snippets to base your new config-templates on. You can select a scenario to start from or create your own.

+ New Custom Preset

Scenarios

- Basic Spark configuration
- Spark Connect
- Spark Streaming
- DBT Integration
- Notebooks

1 {
2 "type": "Python",
3 "sparkVersion": "3.1.1"
4 }
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20

+ New Preset

Edit

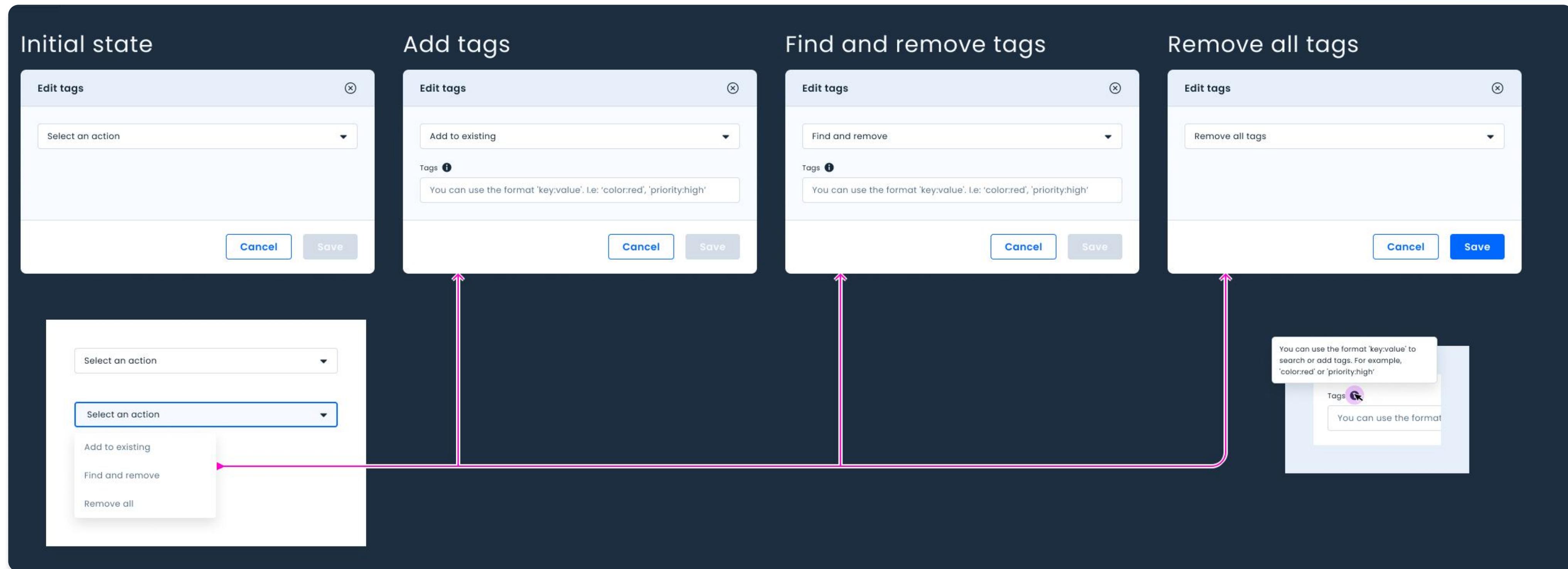
Create a template from this preset

User created preset specific for their needs

Common use-cases to aid selecting the right starter template

The screenshot shows a user interface for managing Configuration Templates, specifically focusing on Presets. At the top, there's a navigation bar with 'Configuration Templates / Presets'. Below it, a title 'Presets' with a subtitle 'Presets are reusable code snippets to base your new config-templates on.' and a note about selecting scenarios or creating custom ones. A 'New Custom Preset' button is highlighted with a blue arrow. To the right is a code editor window showing a JSON template for a Python configuration with a sparkVersion of 3.1.1. The code editor has lines numbered 1 through 20. Above the code editor are buttons for 'Edit', 'Copy', 'Share', and 'Delete'. A callout bubble 'Create a template from this preset' points to the code editor area. On the left, a sidebar lists five 'Scenarios': 'Basic Spark configuration', 'Spark Connect', 'Spark Streaming', 'DBT Integration', and 'Notebooks', each with a short description. A callout bubble 'User created preset specific for their needs' points to the 'Presets' title. Another callout bubble 'Common use-cases to aid selecting the right starter template' points to the 'Scenarios' list.

Tagging system



</> Ready for dev :)

Main page

Config-templates page

table

Toggle tags

Groups

Templates selection

template panel

filter

Scroll behaviour

Screen sizes

dark theme

Below the fold

Edit template

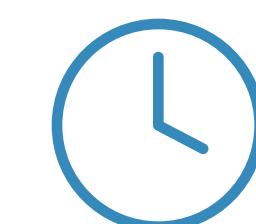
Edit templates page

Tags edit popup

Add tags

Remove tags

So, how did we do?



Time saving

Reduced troubleshooting and config-template creation time

Monitored via Pendo



User satisfaction

More users per team will start working with the config-template

Monitored via Solution Architects



Efficiency

Reduced the amount of unused config-template per customer

Monitored via BI tools



Selling point

Helping tool for the sales team to recruit new clients over competitors

Monitored via sales reports



Marketing campaigns monitoring tool

Role	Teammates	Timeline
Product Designer	Product Managers, fullstack developers	5 month

The goal

Some context

- A web application designed to simplify marketing campaign management with a user-friendly interface.
- It streamlines workflows and enhances collaboration, making campaign monitoring more efficient.
- It had started as an inner product and expanded to be used with more than 30 partners.

The goal

What do we want to achieve?



Real-time performance tracking

Support crucial decision making



Easy campaign selection

Better organization saves time



Clear communication channels

Reduces frictions among business partners



Transparent reporting mechanisms

Financial teams can monitor more closely



Research

Who are the users?

Advertising and Marketing Managers

Responsible for driving online sales through digital marketing campaigns, optimizing customer retention, analyzing data, and collaborating with cross-functional teams.

“I need seamless tools to track performance, personalize campaigns, and scale efficiently while optimizing ad spend”

Marketing affiliate manager

A strategic affiliate manager focused on partnerships, conversions, and commission optimization. Seeks reliable tracking, high-performing affiliates, and scalable growth.

“I need accurate tracking, strong relationships, and the right incentives to drive consistent, high-quality traffic”

Reaching out to the users

Interview and surveys

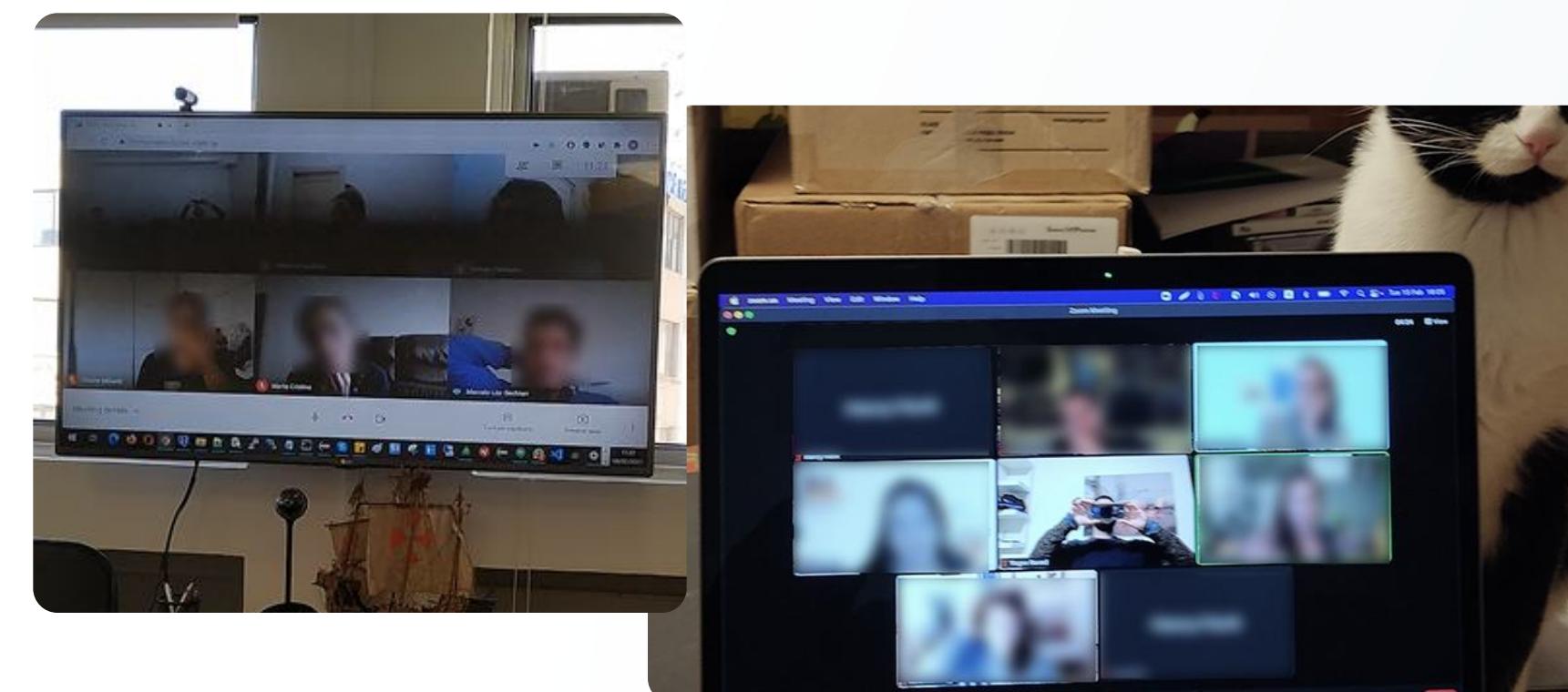
Surveys

We composed a series of questions, sent it to partners and shared it within marketing and affiliates communities.

This survey is designed to gather insights about their experience with marketing campaign tools. We aimed to understand their needs, preferred tools, and challenges when working with partners.

Interviews

We interviewed several affiliate managers, to gather insights on their experiences, challenges, and needs when working with marketing campaign management tools. The goal was to identify key pain points, preferred features, and opportunities for improvement to enhance usability and efficiency.



Key Research Findings

**“As an marketing manager,
I want to:**

- Improve affiliate engagement.
- Ensure transparent reporting and analytics.
- Track and compare performance to monitor spending.
- Verify traffic quality

**“As an affiliate manager,
I want to:**

- Optimize campaigns with real-time data.
- Ensure fair and accurate payments.
- Enable direct contact with advertisers for faster issue resolution.
- Integrate multiple tracking tools into one dashboard.

Key Research Findings

Insights

1. Most tools in the market fit less to medium\small networks.
2. Simplify multi-channel campaign tracking.
3. Streamline affiliate onboarding.
4. Increase visibility into campaign metrics.
5. Improve communication workflows.



Ideation & exploration

Design solution

- UI including campaigns list and their data
- Performance dashboard with real-time stats
- Responsive design for all devices
- Embedding them within the emerging financial platform we had worked on

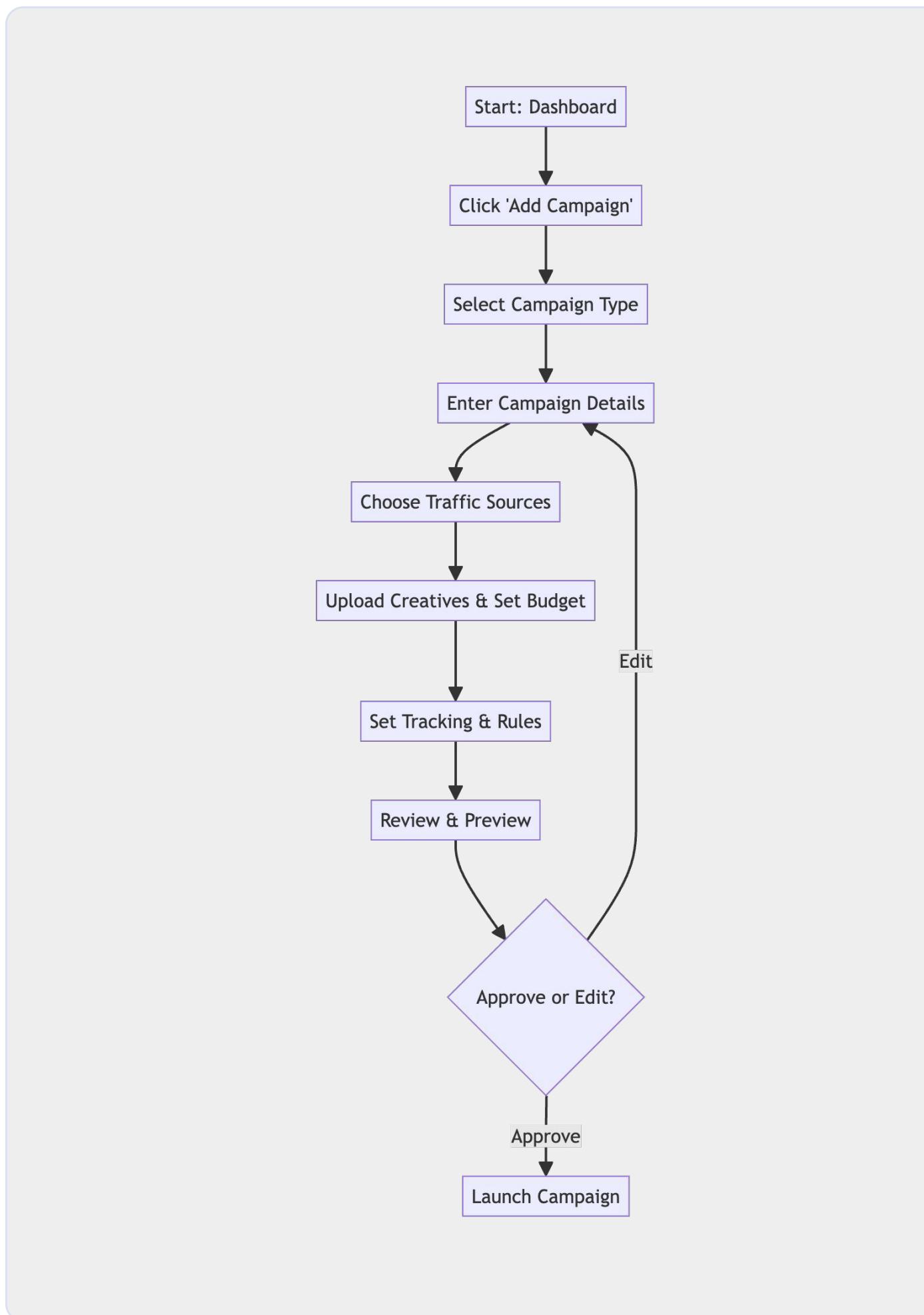
Design process

- Information Architecture
- Designed an intuitive dashboard layout
- Developed clear navigation for campaign selection and management
- Wireframing & Prototyping
- Created low-fidelity wireframes
- Developed interactive prototypes for user testing
- Iterative design based on user feedback
- Design system and UI
- Developed a design system from scratch and implemented it over the wireframes

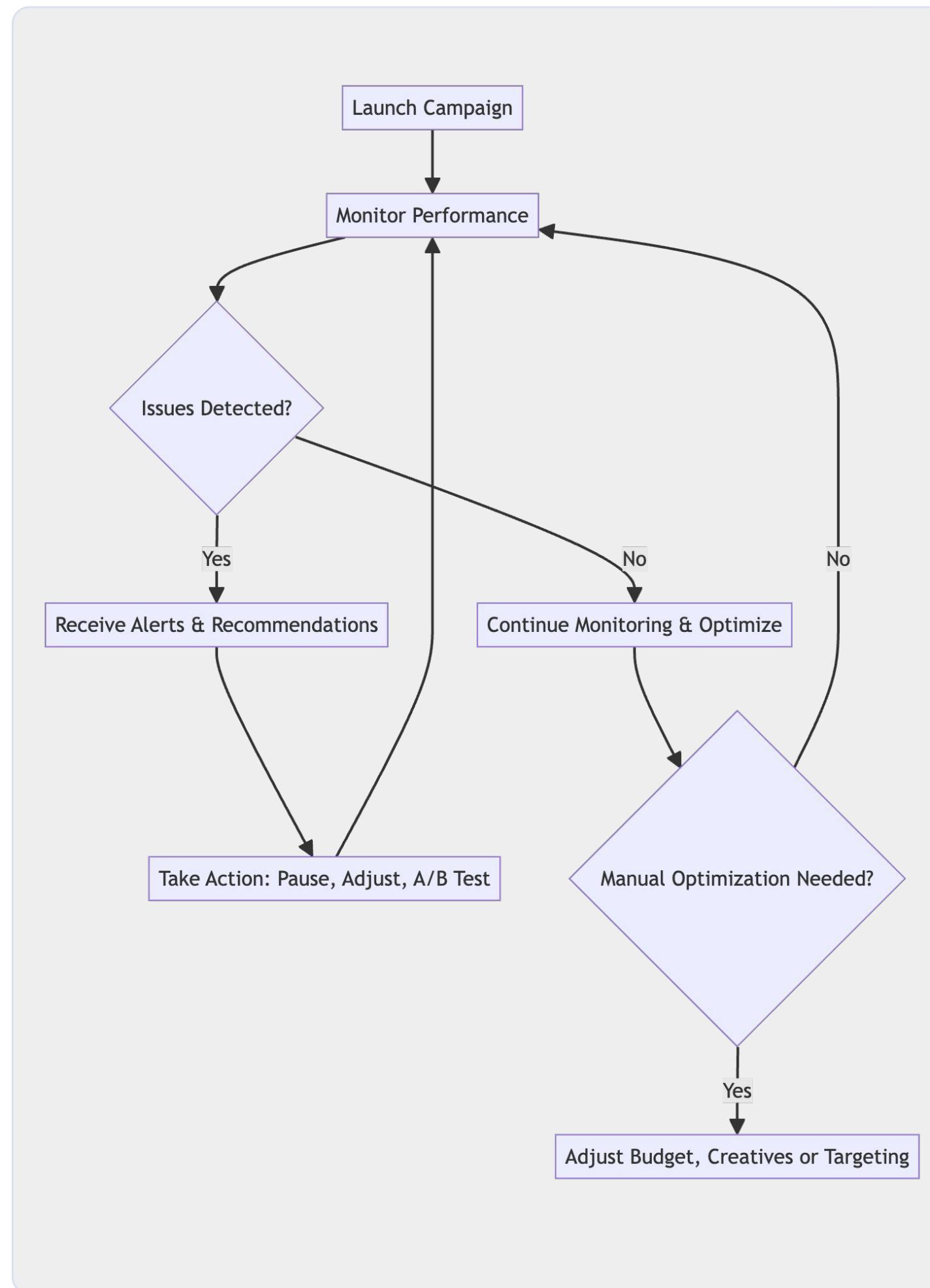


User flows

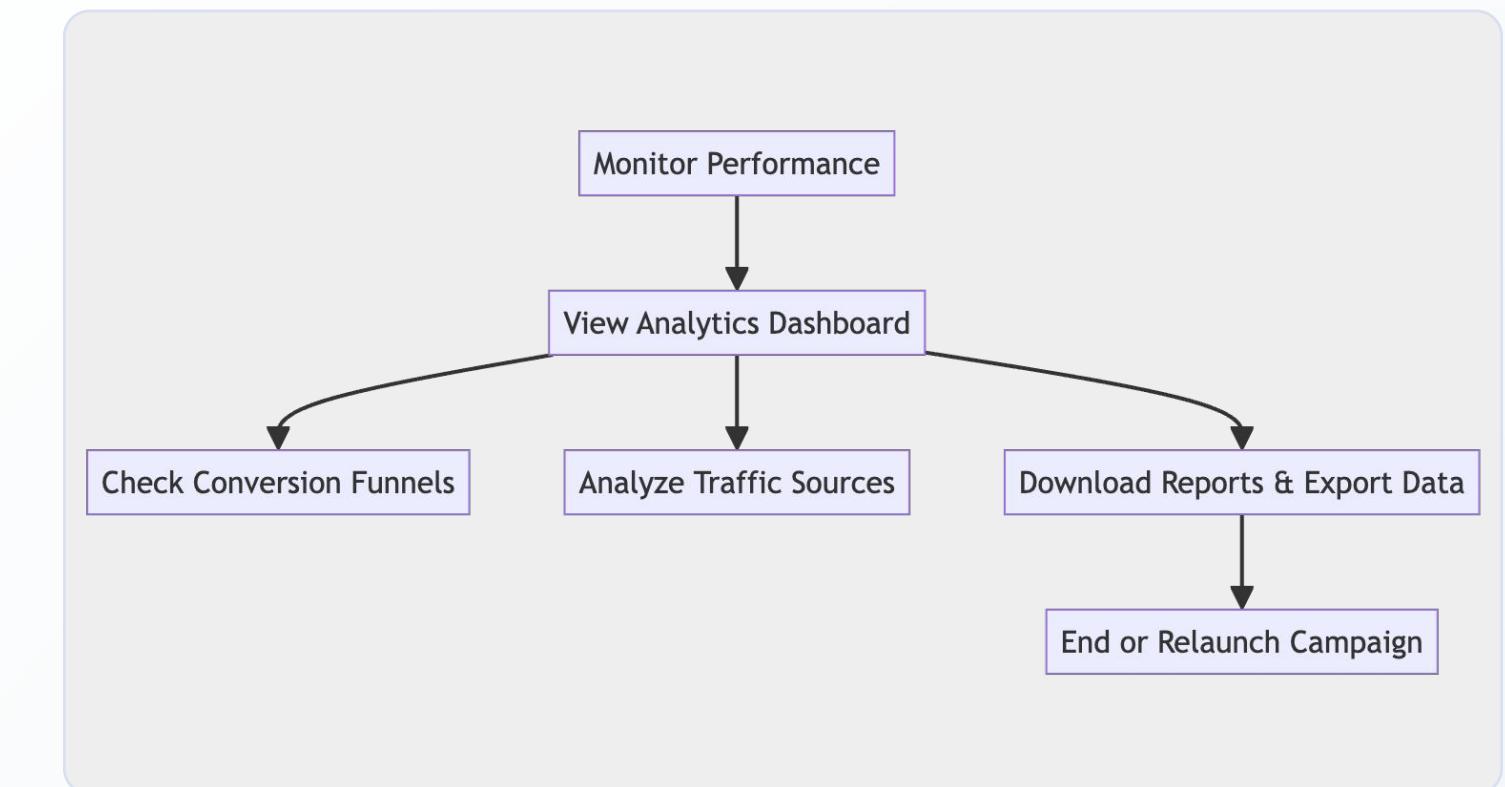
Create campaign



Edit



Monitoring - main



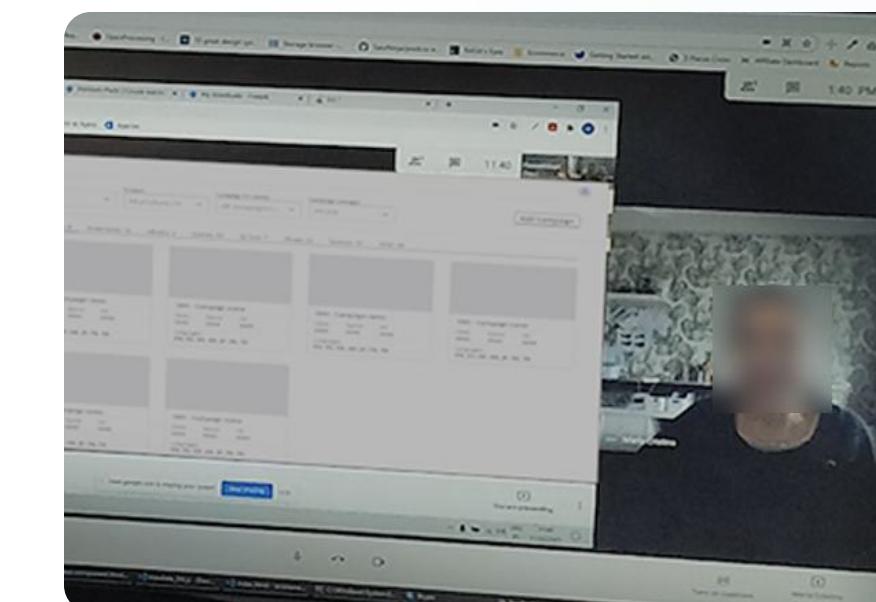
User testing for grouping

Expanded sections

The screenshot shows a user interface for campaign management. On the left, there's a sidebar with 'Campaigns' selected. Under 'Management', it lists 'Games: 12' and 'Security: 6'. Below these are four collapsed card-like components, each labeled '1001 - Campaign name' and showing placeholder data for Clicks, Spend, CR, and Languages. An 'Add Campaign' button is located in the top right corner.

Tabs division

This screenshot shows a similar user interface but with a tab-based navigation at the top. The tabs are 'Management', 'Affiliates', and 'Billing'. Below the tabs, there are four tabs representing different campaign types: 'Security: 6', 'Downloads: 14', 'eBooks: 2', and 'Games: 30'. Each tab has its own set of four collapsed card components, each labeled '1001 - Campaign name' and showing placeholder data for Clicks, Spend, CR, and Languages. An 'Add Campaign' button is located in the top right corner.



Some exploration

The collage consists of six distinct interface snippets:

- Top Left:** A product management or catalog interface titled "Marketing E-commerce Management". It shows a list of products under the category "Gadgets", specifically "M-Grade smart watch". Each item has a thumbnail, ID, type, stock level, and tags (Watch, Wearable, Smartwatch, luxury). Buttons for "Edit Product" and "Delete" are present.
- Top Middle:** An "Edit account" dialog box. It includes fields for "Edit account name" (with placeholder "Cap") and "Comments". Two sections for "Cap - Credit Card A" and "Cap - Credit Card B" show ranges from \$13.00 to \$72,000, with "USD 10,000" selected for both. Buttons for "Cancel" and "Save" are at the bottom.
- Top Right:** A dashboard titled "● Active". It displays metrics for "Credit Card A" (24.5% | 50(70)k), "Credit Card B" (51.2% | 50(20)k), and "Volume" (8k). Below these are two red progress bars labeled "Chase after silly colored fish toys around the house Dont wait for the storm to pass". A note at the bottom says "Bank: Bank name Connector: Connector name www.account-domain.com".
- Middle Left:** A marketing campaign performance dashboard for "1001 - Campaign name stats". It shows weekly data from "01/11/2020 - 31/11/2020". Key metrics include Clicks: 1.27M, Sales: 19.IK, Rev: 72.3K, CR: 1.4%, and a funnel data chart showing impressions, clicks, leads, and billing/conversions.
- Middle Center:** A grid-based interface showing multiple active bank accounts. Each card includes the bank name, domain, and specific transaction details like Visa CB, Mastercard CB, and volume. A note below each card states: "Summary test of the printing and typesetting industry. Lorem ipsum has been standard."
- Middle Right:** A detailed view of a single bank account. It lists various transaction types (Visa CB, Mastercard CB, etc.) with their respective counts and volumes. A note at the bottom says: "Summary test of the printing and typesetting industry. Lorem ipsum has been standard."



Final design

I deliver!

Campaigns list

CLIX
SCALE

Marketing Management

Armstrong Media

JC

[Campaigns](#)

[Analytics](#)

[Affiliates](#)

Group by

Default

Product

All products (7)

Campaign ID \ Name

All campaigns...

Campaign manager

All (23)

Clicks

Spends

CR

[+ Add Campaign](#)

1001 - Campaign name

Clicks: 6.97K Spend: 19K CR: 1.27%

Languages:
EN, ES, DE, AR, JP, FR, TR

1001 - Campaign name

Clicks: 6.97K Spend: 19K CR: 1.27%

Languages:
EN, ES, DE, AR, JP, FR, TR

1001 - Campaign name

Clicks: 6.97K Spend: 19K CR: 1.27%

Languages:
EN, ES, DE, AR, JP, FR, TR

1001 - Campaign name

Clicks: 6.97K Spend: 19K CR: 1.27%

Languages:
EN, ES, DE, AR, JP, FR, TR

1001 - Campaign name

Clicks: 6.97K Spend: 19K CR: 1.27%

Languages:
EN, ES, DE, AR, JP, FR, TR

1001 - Campaign name

Clicks: 6.97K Spend: 19K CR: 1.27%

Languages:
EN, ES, DE, AR, JP, FR, TR

1001 - Campaign name

Clicks: 6.97K Spend: 19K CR: 1.27%

Languages:
EN, ES, DE, AR, JP, FR, TR

1001 - Campaign name

Clicks: 6.97K Spend: 19K CR: 1.27%

Languages:
EN, ES, DE, AR, JP, FR, TR

I deliver!

Campaigns list

Group by State, Product and Campaign manager

The screenshot displays a marketing management platform with a top navigation bar featuring 'Marketing Management' and 'Armstrong Media' with a 'JC' logo. On the left sidebar, there are links for 'Campaigns', 'Analytics', and 'Affiliates'. The main area shows a grid of campaign preview cards, each containing a thumbnail, a campaign name ('1001 - Campaign name'), and performance metrics ('Clicks: 6.97K', 'Spend: 19K', 'CR: 1.27%'). The cards also show language details ('Languages: EN, ES, DE, AR, JP, FR, TR'). A pink arrow points from the 'Product' dropdown in the top search bar to the first card. Another pink arrow points from the 'Add Campaign' button to the top right.

Filtering and sorting mechanism

Campaign actions

The image displays three screenshots of a software application interface, likely a campaign manager, illustrating a tooltip feature for campaign actions.

- Screenshot 1:** Shows the main campaign overview page for "1001 - Campaign name". It includes metrics: Clicks: 6.97K, Spend: 19K, CR: 1.27%, and language support: EN, ES, DE, AR, JP, FR, TR.
- Screenshot 2:** Shows the same campaign page with a mouse cursor hovering over the campaign name "1001 - Campaign name". A tooltip appears, displaying the same metrics and language information, along with two action buttons: "Stats" and "Edit".
- Screenshot 3:** Shows the tooltip from Screenshot 2 enlarged, highlighting the "Edit" button. A pink arrow points from the text "Actions are exposed while hovering" to this tooltip.

Actions are exposed while hovering: Links to stats page and edit campaign

Upper head title

Empty state

CLIX SCALE

Marketing Management

Armstrong Media JC

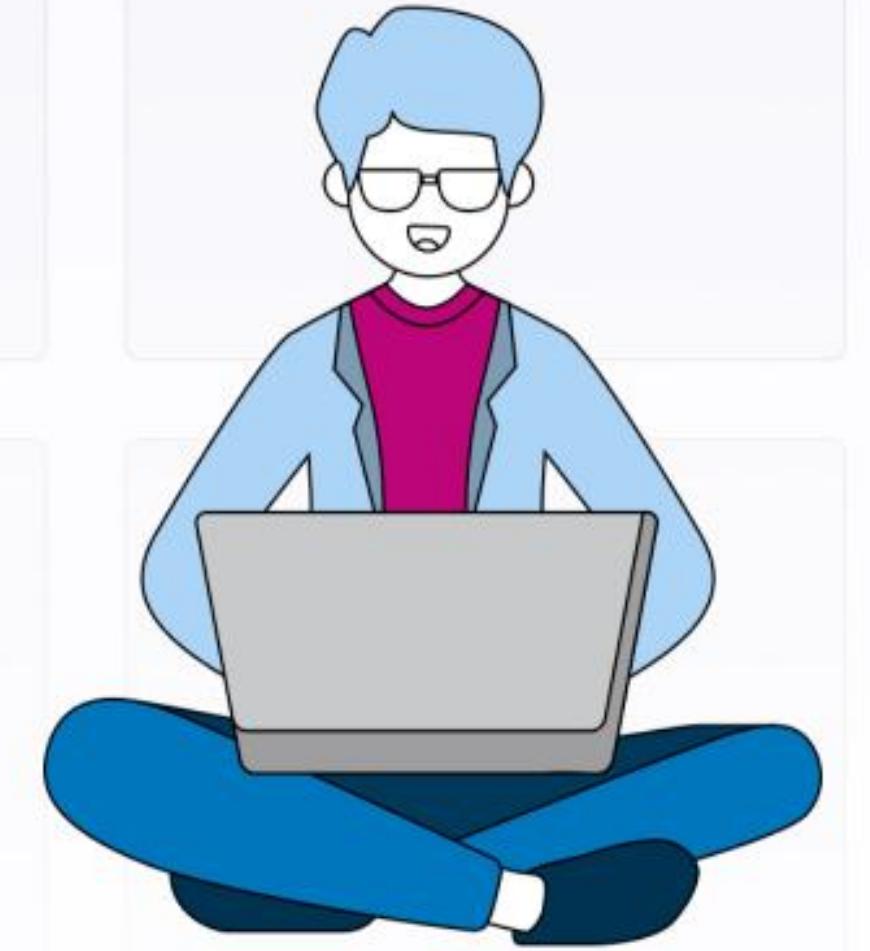
Campaigns

Analytics

Affiliates

Group by Default Product Campaign ID \ Name Campaign manager Clicks Spends CR

+ Add Campaign

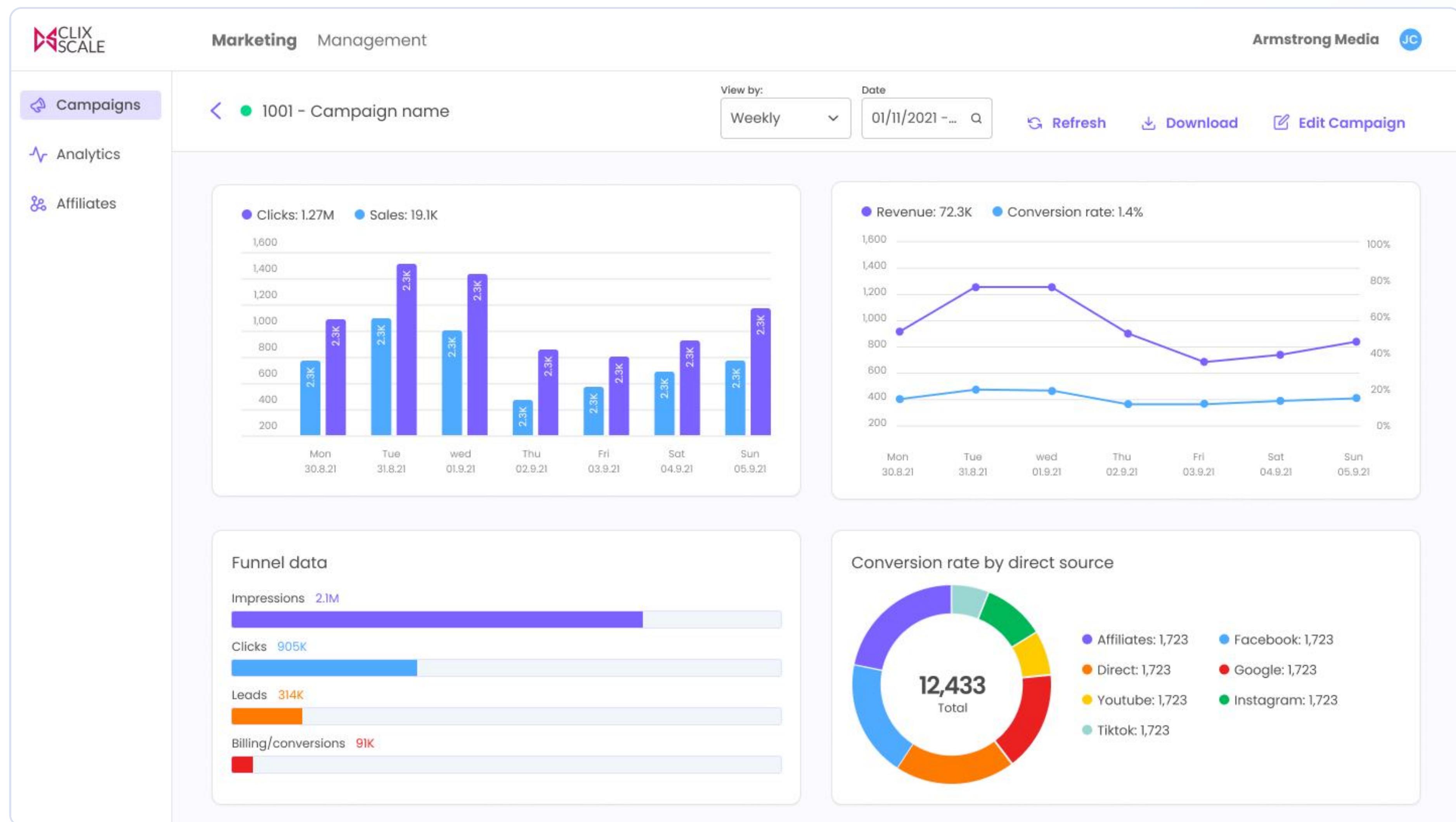


It's time to create your first campaign!

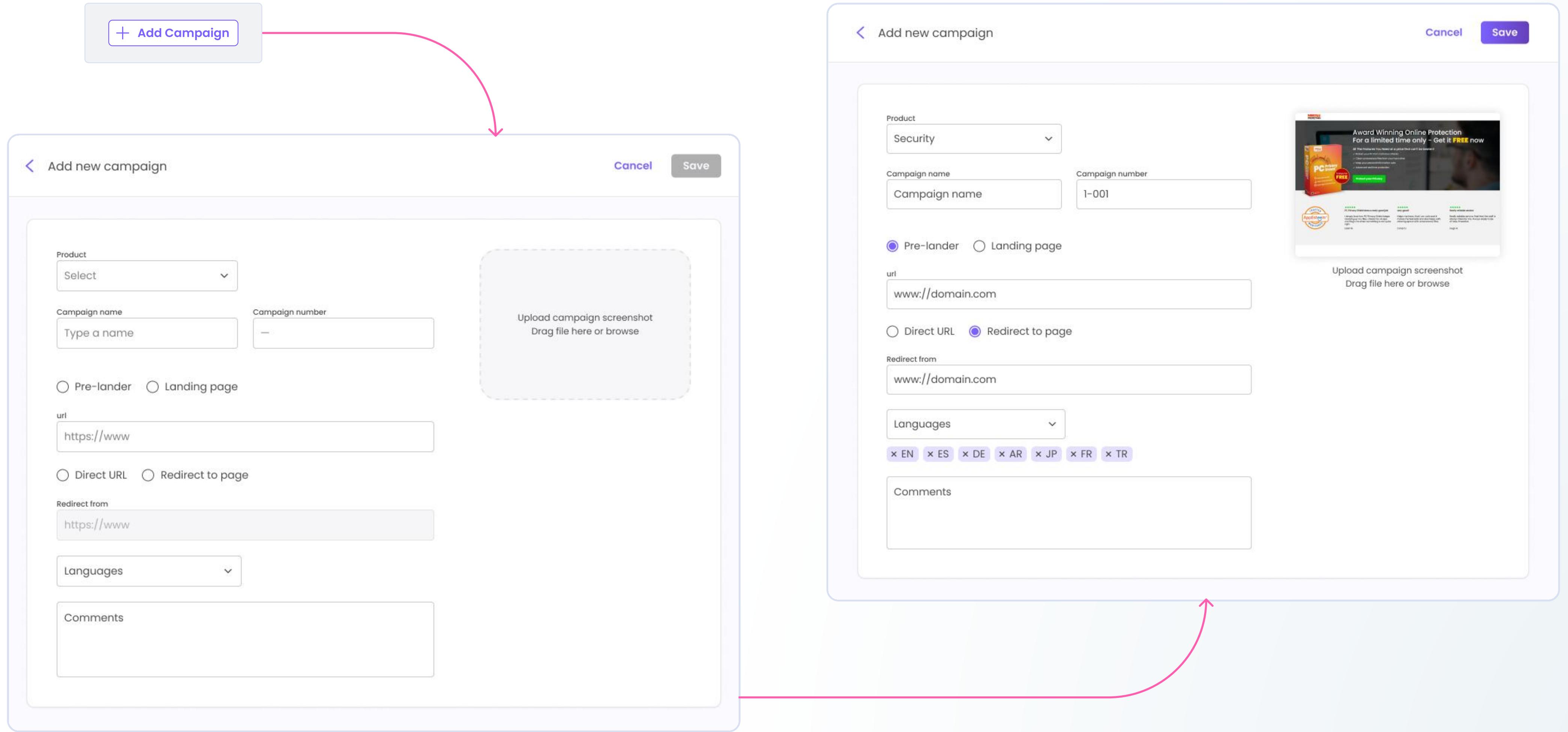
+ Add Campaign

Edit campaign

Campaign stats



New campaign flow



Additional screens

Financial management

CLIX SCALE

Marketing **Management**

Campaigns Analytics Affiliates

Group by Banks Connectors Threat Status

Armstrong Media JC

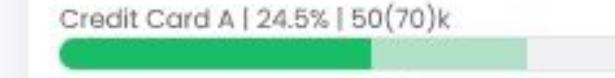
Active

Bank: Bank name
Connector: Connector name
[www.account-domain.com](#)

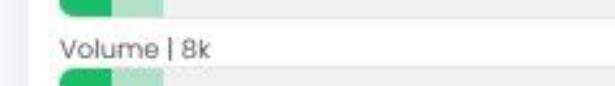
Credit Card A | 24.5% | 50(70)k 90Tr



Credit Card B | 51.2% | 50(20)k 40Tr



Volume | 8k 10K



Chase after silly colored fish toys around the house Dont wait for the storm to pass

Active

Bank: Bank name
Connector: Connector name
[www.account-domain.com](#)

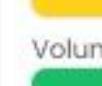
Credit Card A | 24.5% | 50(70)k 90Tr



Credit Card B | 51.2% | 50(20)k 40Tr



Volume | 8k 10K



Chase after silly colored fish toys around the house Dont wait for the storm to pass

Active

Bank: Bank name
Connector: Connector name
[www.account-domain.com](#)

Credit Card A | 24.5% | 50(70)k 90Tr



Credit Card B | 51.2% | 50(20)k 40Tr



Volume | 8k 10K



Chase after silly colored fish toys around the house Dont wait for the storm to pass

Active

Bank: Bank name
Connector: Connector name
[www.account-domain.com](#)

Credit Card A | 24.5% | 50(70)k 90Tr



Credit Card B | 51.2% | 50(20)k 40Tr



Volume | 8k 10K

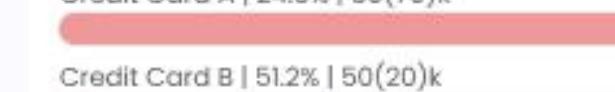


Chase after silly colored fish toys around the house Dont wait for the storm to pass

Paused

Bank: Bank name
Connector: Connector name
[www.account-domain.com](#)

Credit Card A | 24.5% | 50(70)k 90Tr



Credit Card B | 51.2% | 50(20)k 40Tr



Volume | 8k 10K

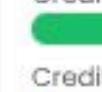


Chase after silly colored fish toys around the house Dont wait for the storm to pass

Active

Bank: Bank name
Connector: Connector name
[www.account-domain.com](#)

Credit Card A | 24.5% | 50(70)k 90Tr



Credit Card B | 51.2% | 50(20)k 40Tr



Volume | 8k 10K



Chase after silly colored fish toys around the house Dont wait for the storm to pass

Active

Bank: Bank name
Connector: Connector name
[www.account-domain.com](#)

Credit Card A | 24.5% | 50(70)k 90Tr



Credit Card B | 51.2% | 50(20)k 40Tr



Volume | 8k 10K



Chase after silly colored fish toys around the house Dont wait for the storm to pass

Active

Bank: Bank name
Connector: Connector name
[www.account-domain.com](#)

Credit Card A | 24.5% | 50(70)k 90Tr



Credit Card B | 51.2% | 50(20)k 40Tr



Volume | 8k 10K

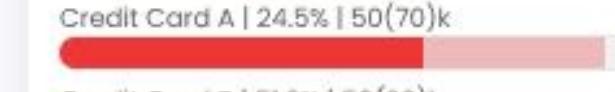


Chase after silly colored fish toys around the house Dont wait for the storm to pass

Active

Bank: Bank name
Connector: Connector name
[www.account-domain.com](#)

Credit Card A | 24.5% | 50(70)k 90Tr



Credit Card B | 51.2% | 50(20)k 40Tr



Volume | 8k 10K



Active

Bank: Bank name
Connector: Connector name
[www.account-domain.com](#)

Credit Card A | 24.5% | 50(70)k 90Tr



Credit Card B | 51.2% | 50(20)k 40Tr



Volume | 8k 10K



Active

Bank: Bank name
Connector: Connector name
[www.account-domain.com](#)

Credit Card A | 24.5% | 50(70)k 90Tr



Credit Card B | 51.2% | 50(20)k 40Tr



Volume | 8k 10K



Active

Bank: Bank name
Connector: Connector name
[www.account-domain.com](#)

Credit Card A | 24.5% | 50(70)k 90Tr



Credit Card B | 51.2% | 50(20)k 40Tr



Volume | 8k 10K



Additional screens

Financial management

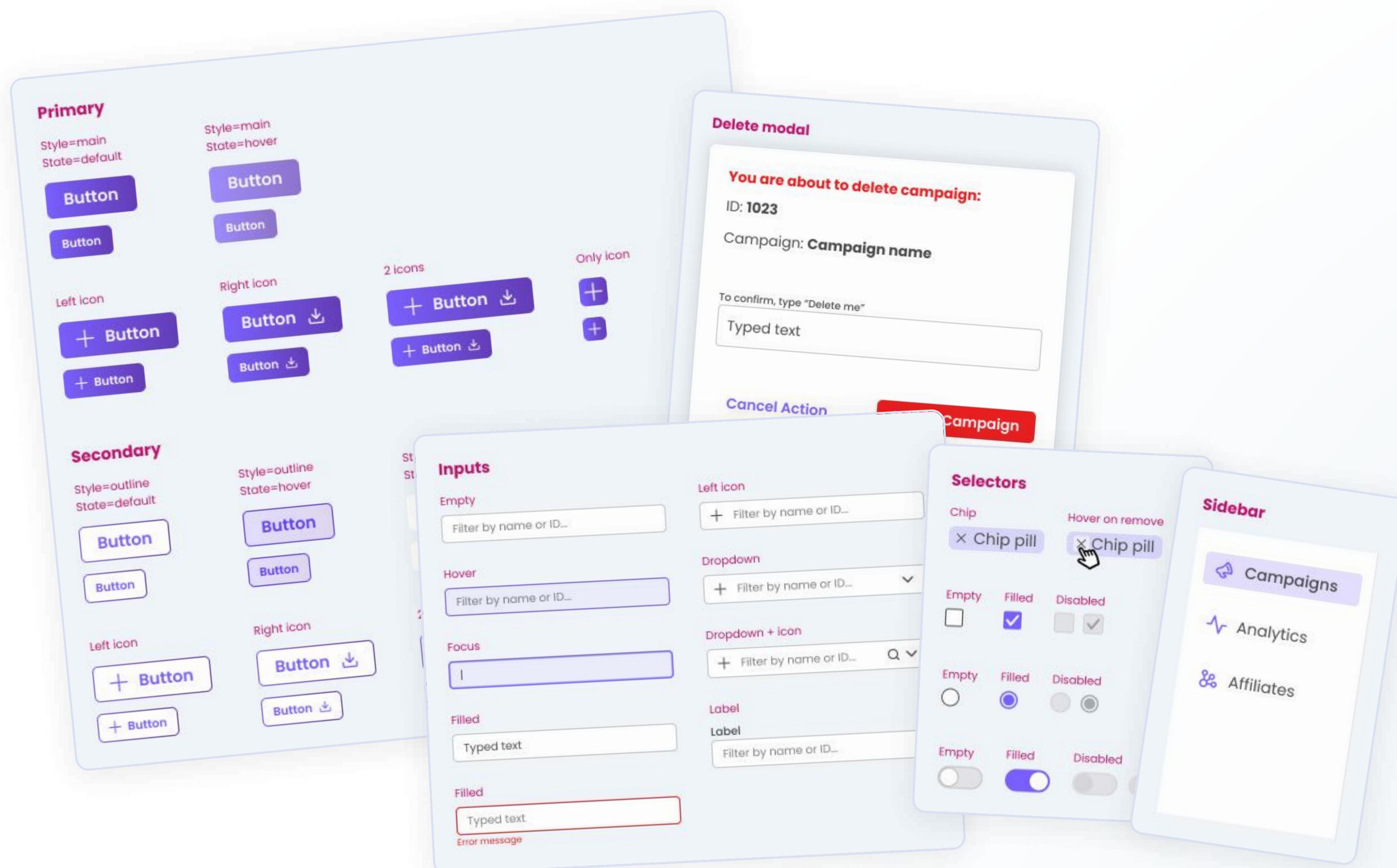
The screenshot displays a digital marketing management platform with a top navigation bar for Marketing and Management. On the left, there's a sidebar with Campaigns, Analytics, and Affiliates sections. The main area shows a list of accounts grouped by status (Active, Paused). Each account entry includes fields for Bank, Connector, and various performance metrics like Credit Card A, Credit Card B, and Volume, each represented by a progress bar.

A central modal window titled "Edit: Account Name" is open, showing the current account name as "Cap - Credit Card A". It features a toggle switch for "Active" status (which is active), dropdown menus for "Change connector" (set to "Select connector") and "Replace domain" (set to "Select domain"), and two sliders for "Cap - Credit Card A" and "Cap - Credit Card B", both set to "USD 100,000". Below these sliders is a "Comments" text area. At the bottom of the modal are "Cancel" and "Save" buttons.

Additional screens

Financial management - compact view

Design system



Thank You :)