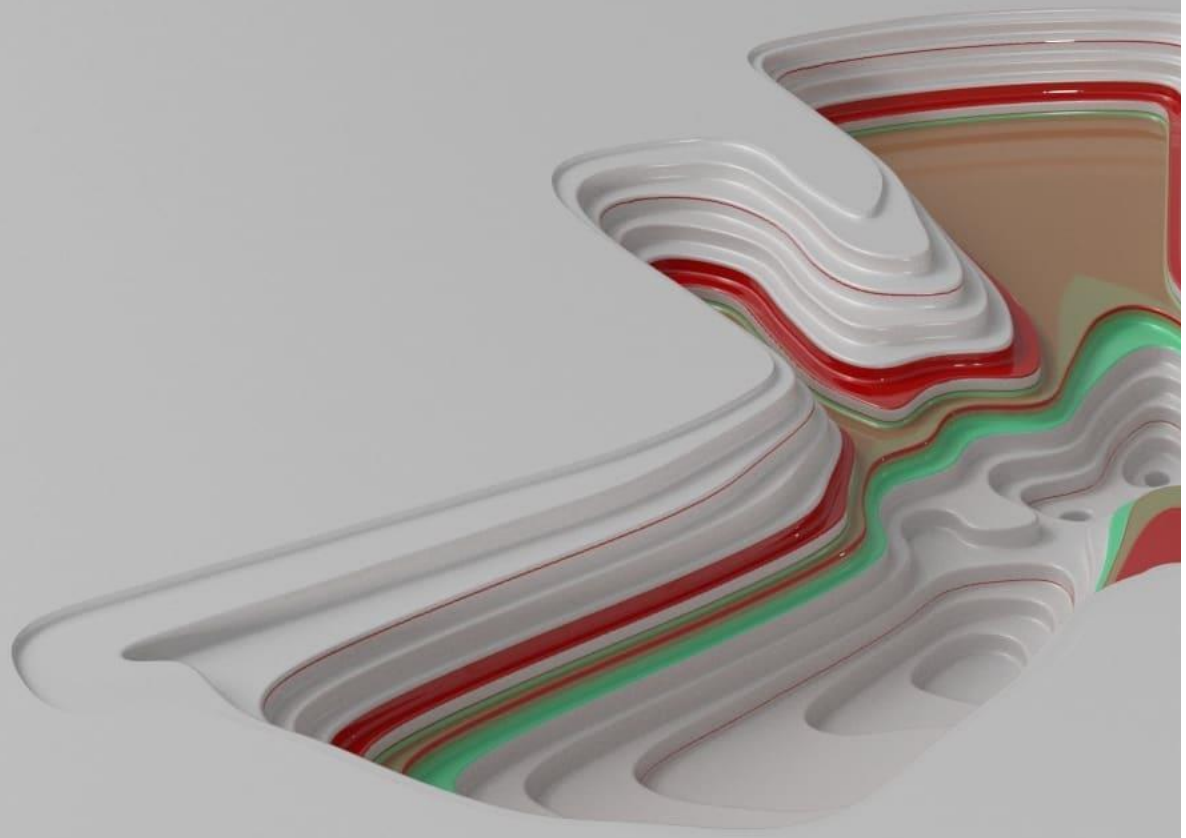


INNOQ Technology Day / 13.11.2023

Backstage

TAMMO VAN LESSEN
Principal Consultant

SASCHA SELZER
Senior Consultant

INNOQ

Agenda

- What is Backstage?
- Core Features
  - Catalog
  - Software Templates
  - Tech Docs
- Authentication
- Plugins

What is Backstage?

Born out of necessity

- "Where's the API for that service we're all supposed to be using?"
- "What version of that framework is everyone on?"
- "This service isn't responding, who owns it?"
- "I can't find documentation for anything!"

March 2022

WTH is Backstage?

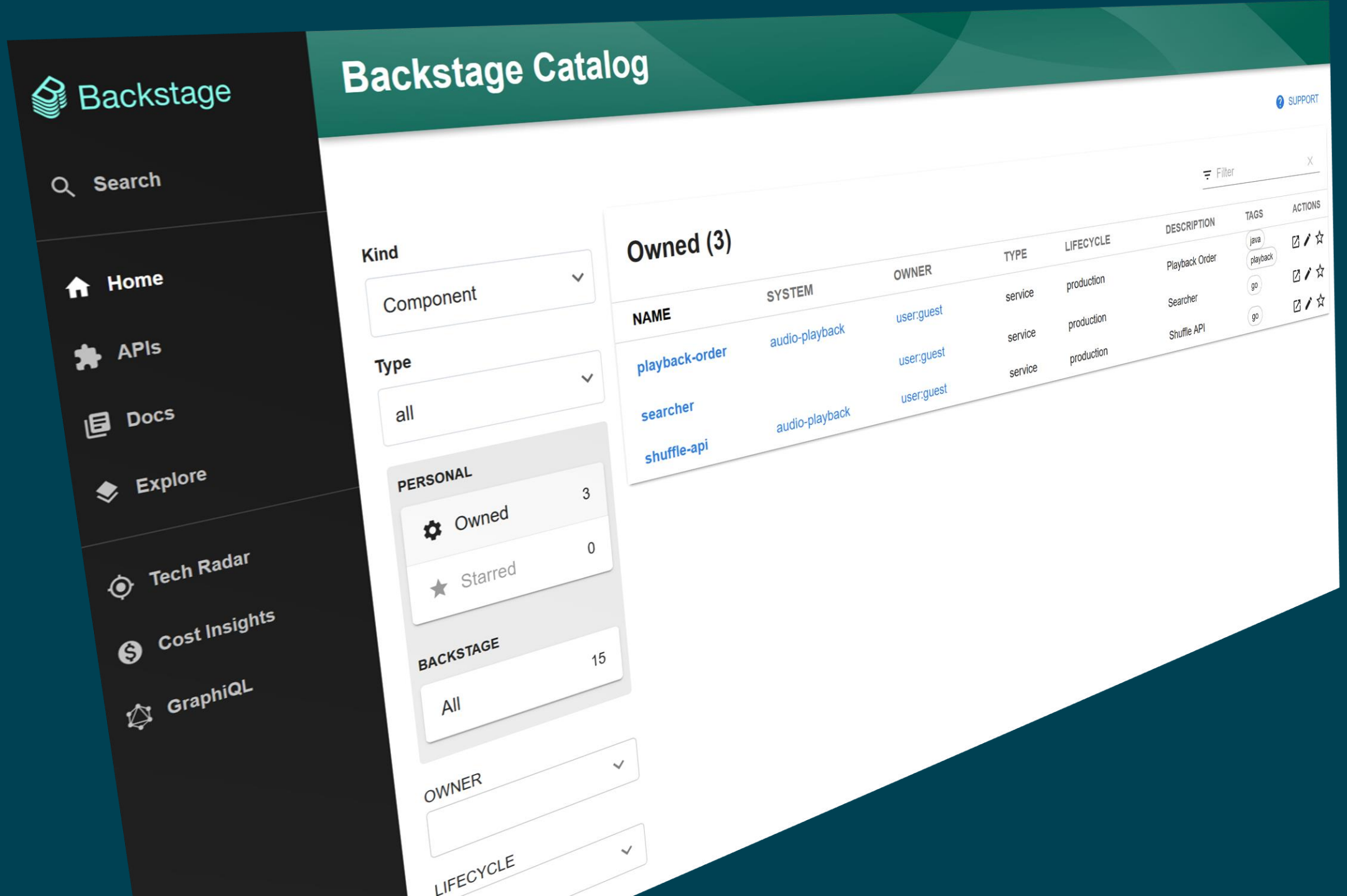
- Open platform for building developer portals
- Centralizes project/service information by crawling and collecting service definitions from different locations
- Aggregates them in a service catalog
- Integrate 3rd party services/content via plugins

- Everything is a plugin
- No hooks, just code

OOBE

- It's not a product, no docker file, no binary.
- It's a platform to run your very own developer portal from the beginning

- Still easy to start:


Backstage
Search
Home
APIs
Docs
Explore
Tech Radar
Cost Insights
GraphiQL
Backstage Catalog
SUPPORT
Filter
Kind
Component
Type
all
PERSONAL
Owned 3
Starred 0
BACKSTAGE
All 15
OWNER
LIFECYCLE
Owned (3)
NAME
SYSTEM
OWNER
TYPE
LIFECYCLE
DESCRIPTION
TAGS
ACTIONS
playback-order
audio-playback
user:guest
service
production
Playback Order
java
playback
searcher
user:guest
service
production
Searcher
go
shuffle-api
audio-playback
user:guest
service
production
Shuffle API
go

Core Features

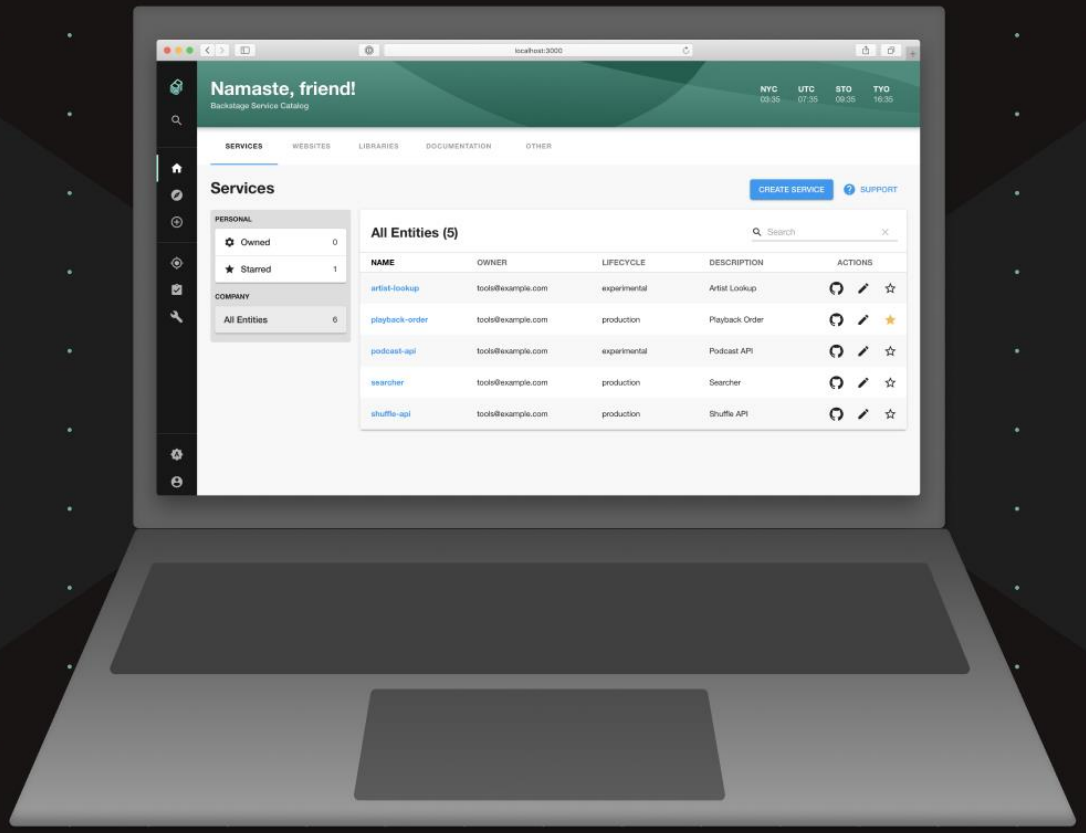
Five Core Pillars of Backstage

- **Software Catalog**
- Kubernetes
- **Software Templates**
- Backstage Search
- **Tech Docs**

Software Catalog

Backstage

```
9 lines (9 sloc)    182 Bytes

1 apiVersion: backstage.io/v1alpha1
2 kind: Component
3 metadata:
4     name: podcast-api
5     description: Podcast API
6 spec:
7     type: service
8     lifecycle: experimental
9     owner: tools@example
```

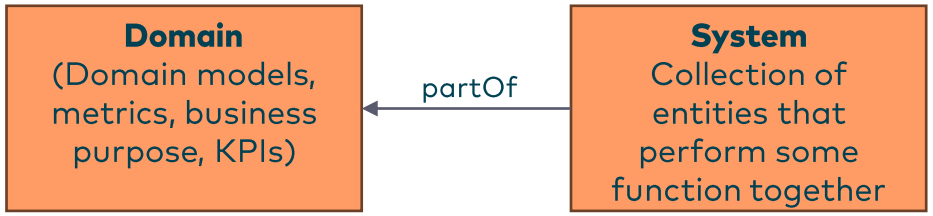


SERVICE DEFINITIONS

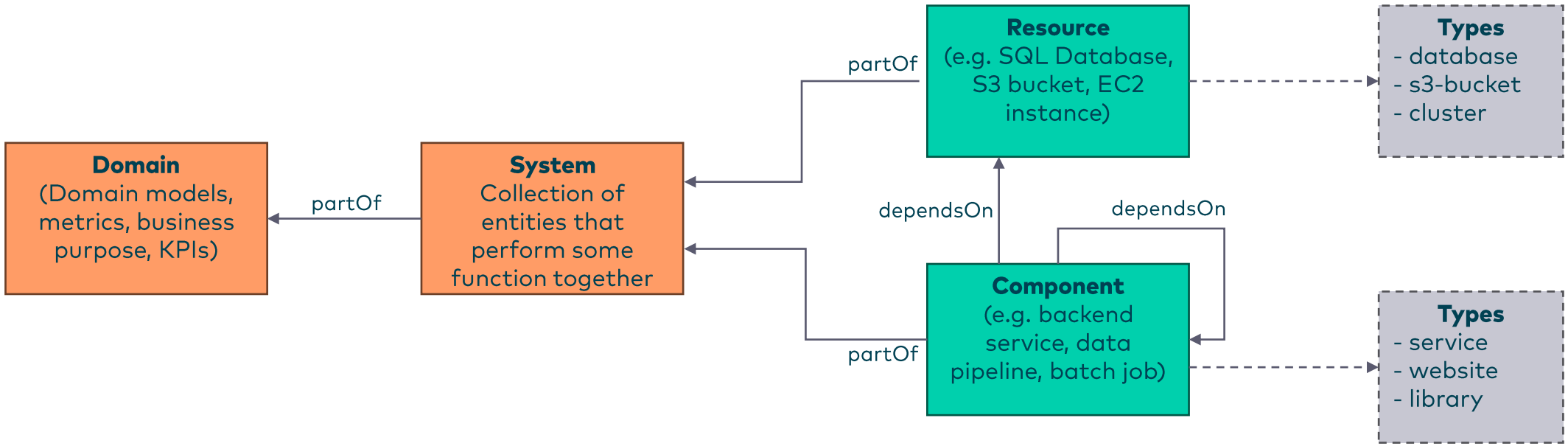
BACKSTAGE SERVICE CATALOG

INTEGRATED TOOLING VIA PLUGINS

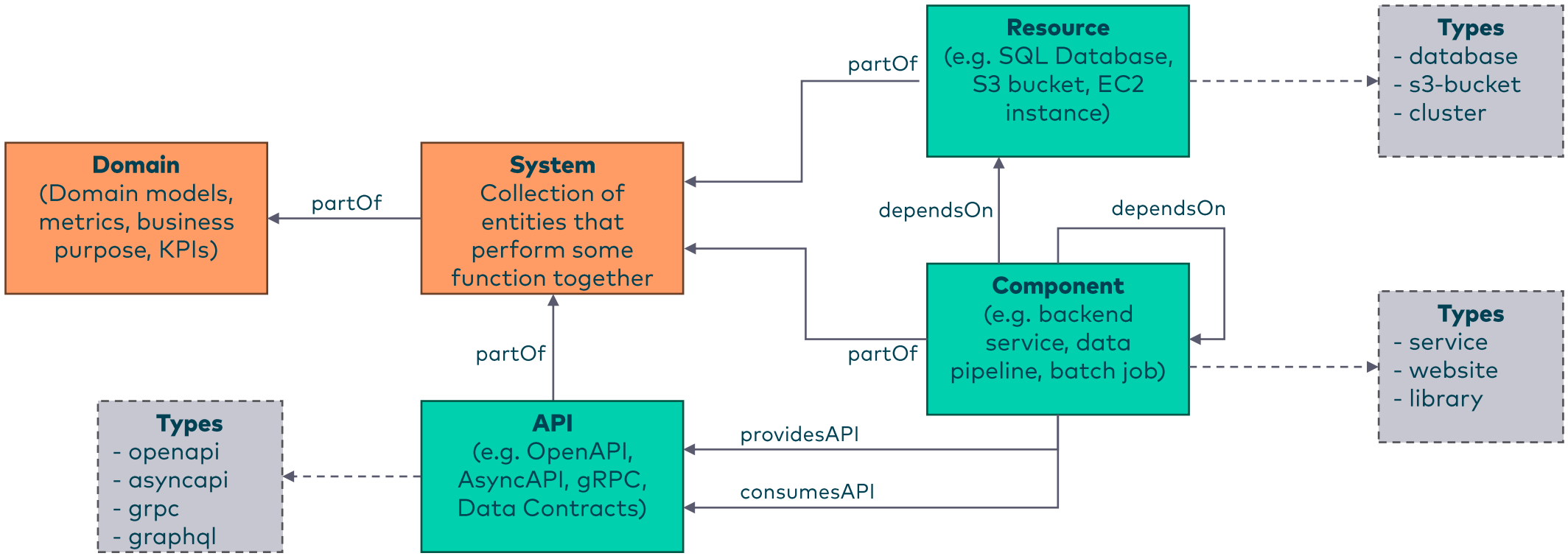
Catalog Entities



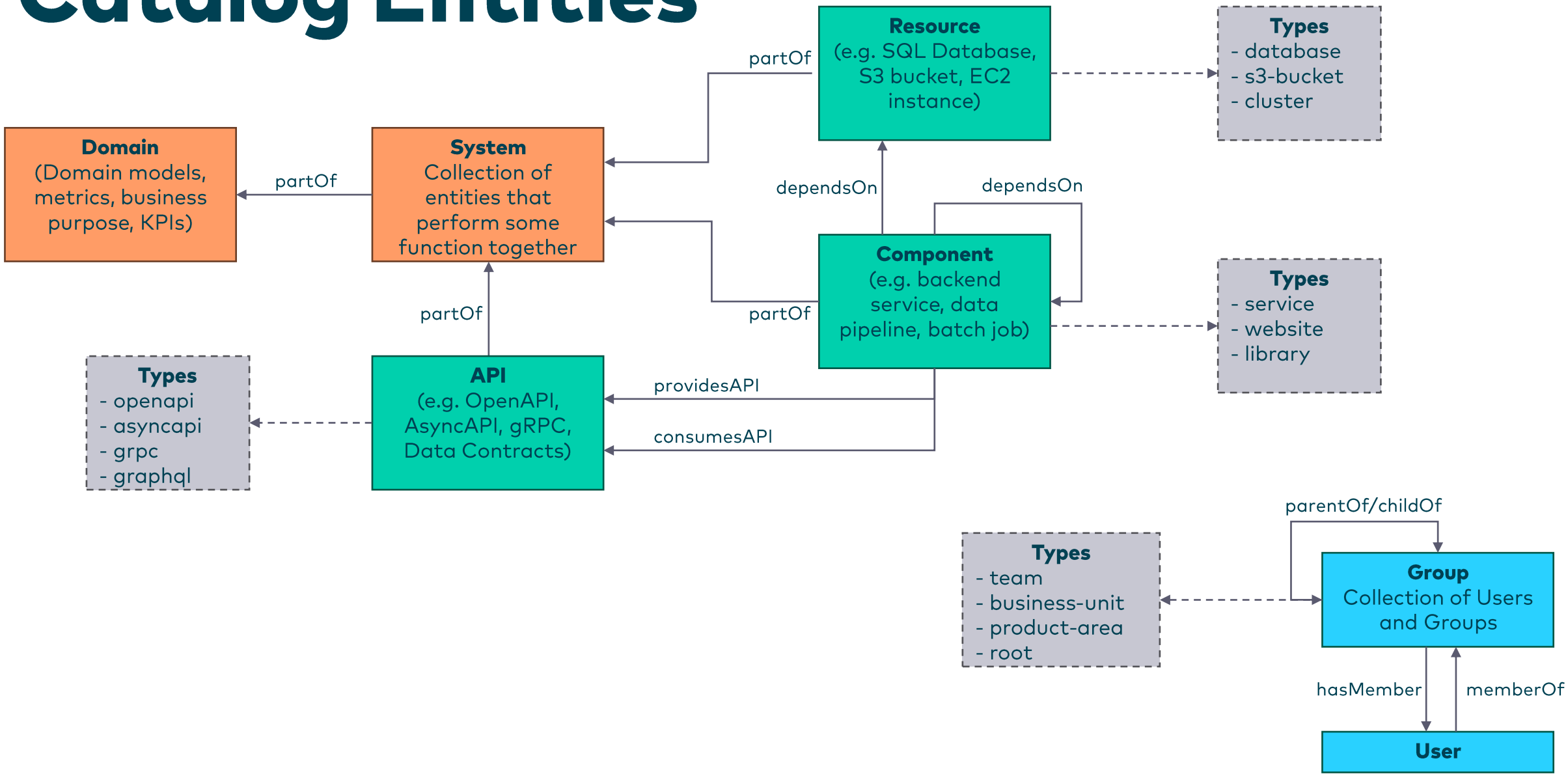
Catalog Entities



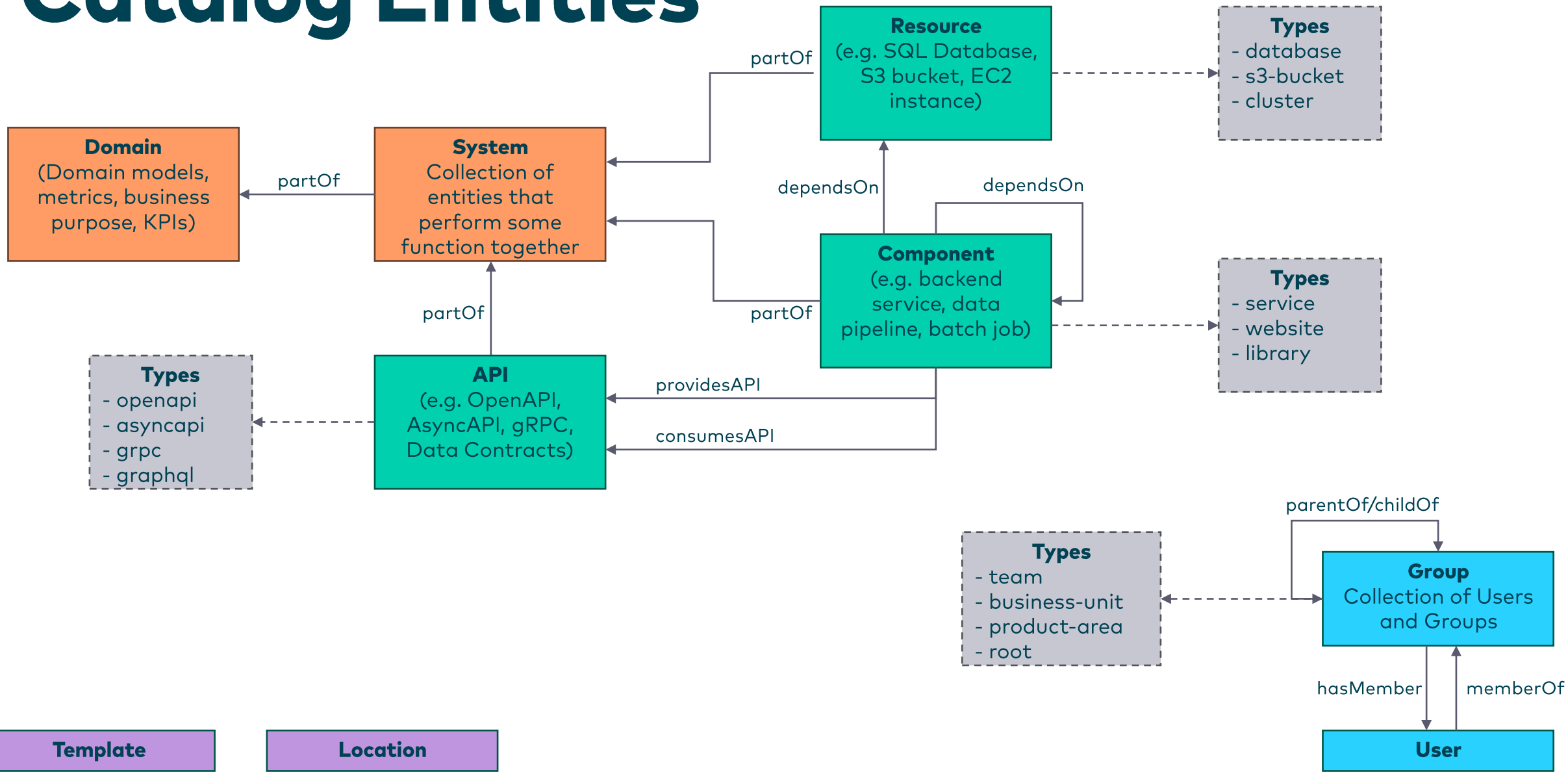
Catalog Entities



Catalog Entities



Catalog Entities



… are described in YAML

```yaml
apiVersion: backstage.io/v1alpha1
kind: Component
metadata:
  name: artist-web
  description: The place to be, for great artists
  labels:
    example.com/custom: custom_label_value
  annotations:
    example.com/service-discovery: artistweb
    circleci.com/project-slug: github/example-org/artist-website
  tags:
    - java
  links:
    - url: https://admin.example-org.com
      title: Admin Dashboard
      icon: dashboard
      type: admin-dashboard
spec:
  type: website
  lifecycle: production
  owner: artist-relations-team
  system: public-websites
```

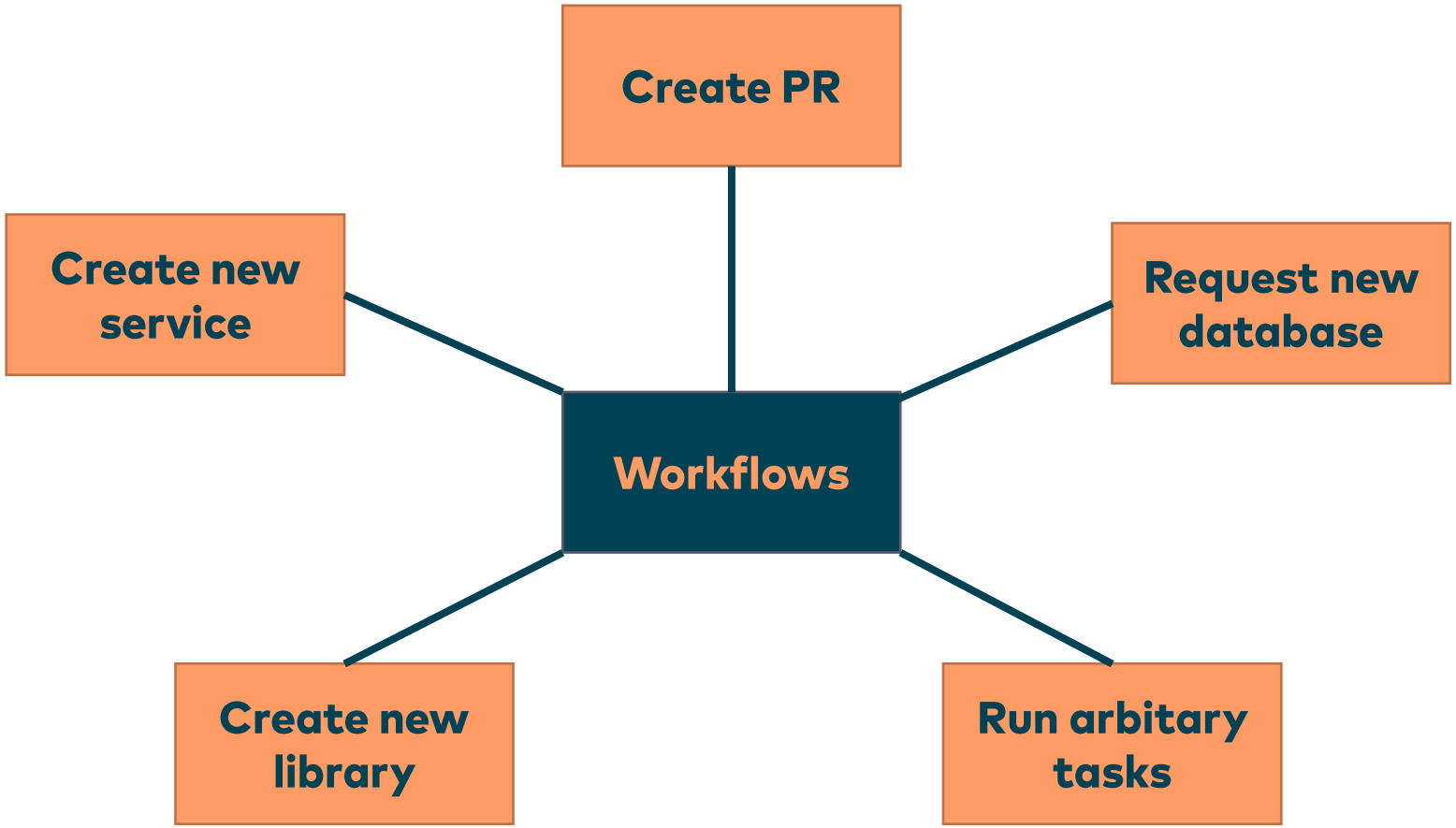
…and put into the repository as catalog-info.yaml next to the source code.
External entity providers are possible, too.

Stop!
Demo Time

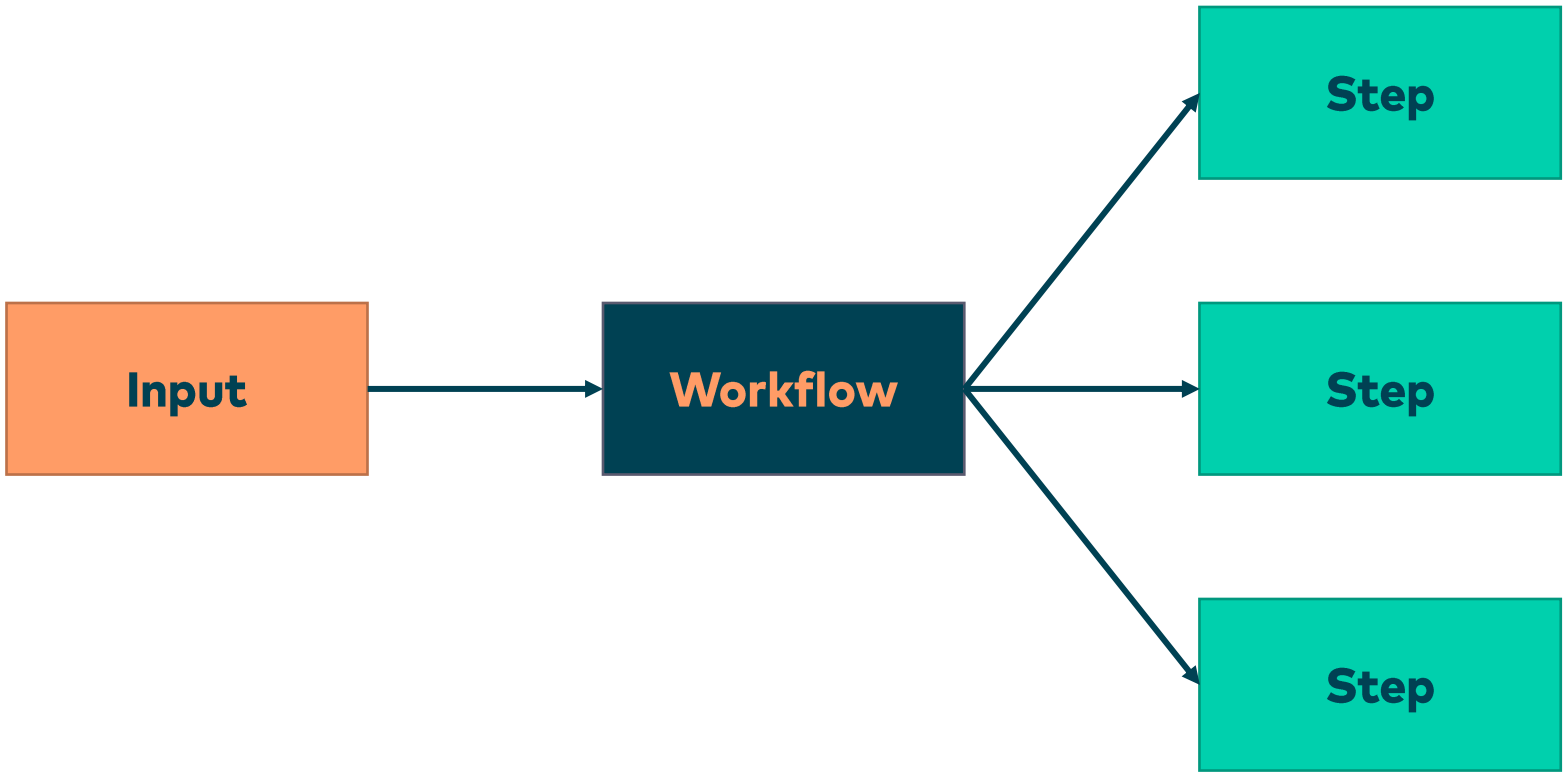
Software Templates

Automate common developer workflows

Software Templates



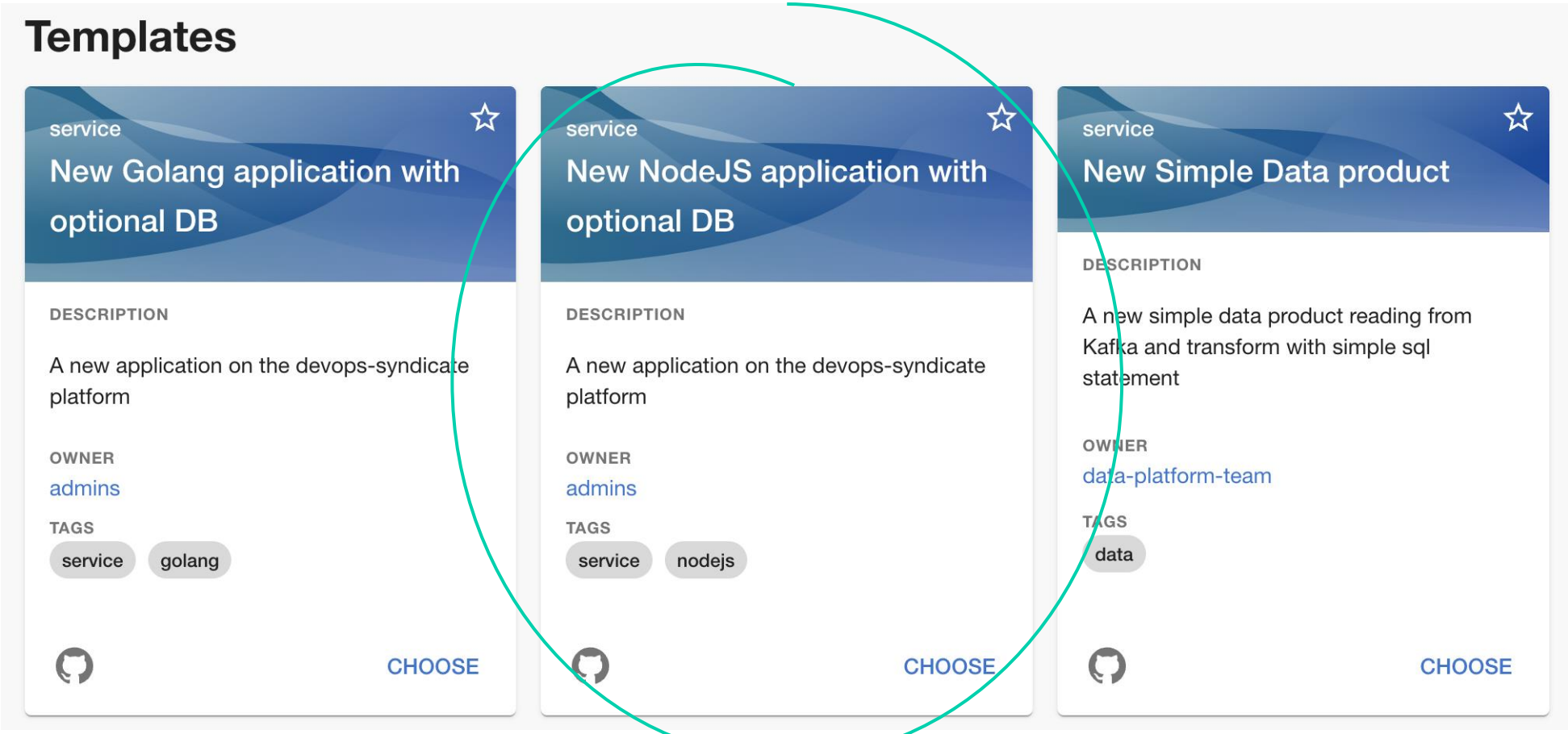
Software Templates



Software Template Resource

```yaml
apiVersion: scaffolder.backstage.io/v1beta3
kind: Template
metadata:
  name: nodejs-template
  title: New NodeJS application with optional DB
  description: A new application on the devops-syndicate platform
  tags: [service, nodejs]
spec:
  owner: admins
  type: service
  ...
```

Software Template Resource



```yaml
apiVersion: scaffolder.backstage.io/v1beta3
kind: Template
...
spec:
  ...
  parameters:
    - title: Basic information
      required:
        - name
        - owner
      properties:
        name:
          title: Name
          type: string
          description: Unique name of the component
          ui:autofocus: true
          ui:options:
            rows: 5
        owner:
          title: Owner
          type: string
          description: Owner of the component
          ui:field: OwnerPicker
          ui:options:
            allowedKinds:
            - Group
```

New NodeJS application with optional DB

1 Basic information

Name*

Unique name of the component

Owner*

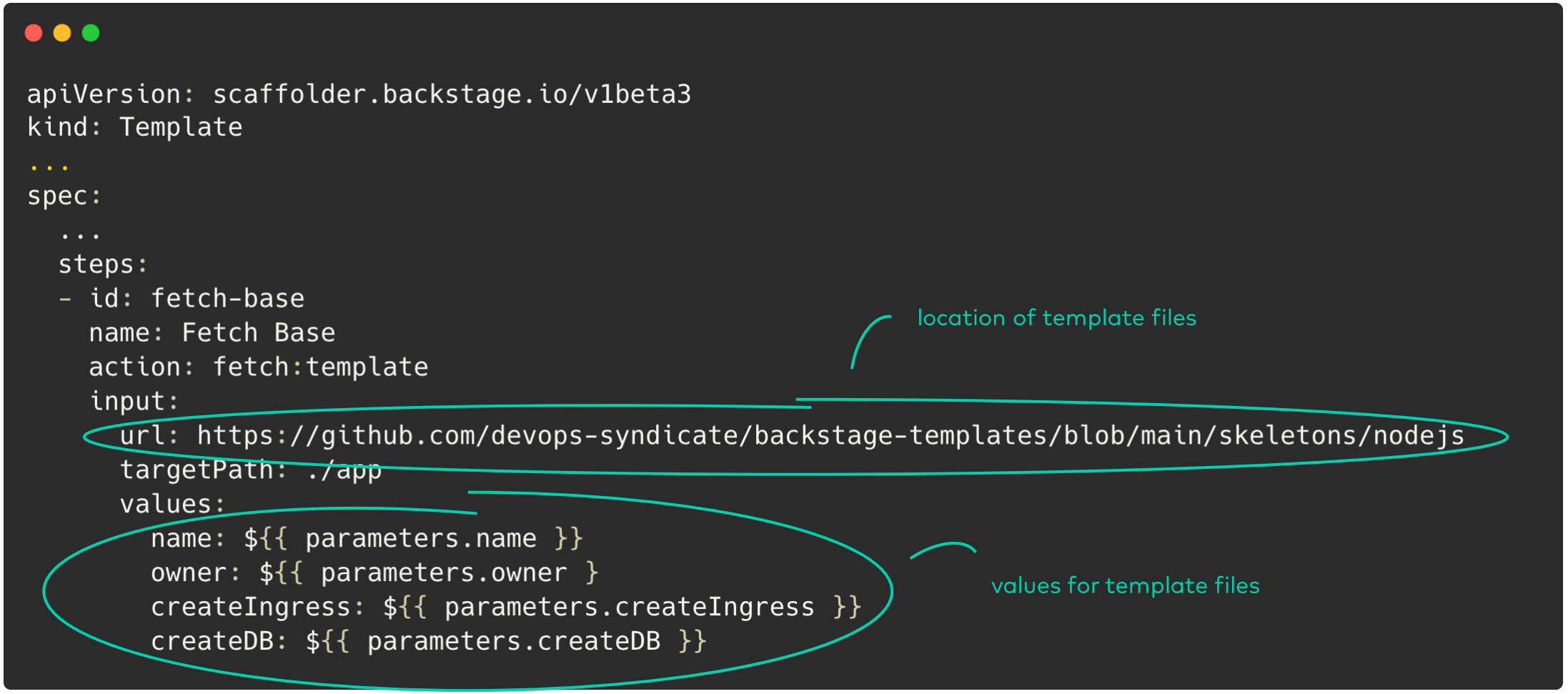
Owner of the component

BACK NEXT STEP

2 Location

3 Ingress

4 Database


apiVersion: scaffolder.backstage.io/v1beta3
kind: Template
...
spec:
...
steps:
- id: fetch-base
name: Fetch Base
action: fetch:template
input:
url: https://github.com/devops-syndicate/backstage-templates/blob/main/skeletons/nodejs
targetPath: ./app
values:
name: ${{ parameters.name }}
owner: ${{ parameters.owner }}
createIngress: ${{ parameters.createIngress }}
createDB: ${{ parameters.createDB }}
location of template files
values for template files

Template files

```yaml
apiVersion: backstage.io/v1alpha1
kind: Component
metadata:
  name: "${{ values.name }}"
  annotations:
    github.com/project-slug: "devops-syndicate/${{ values.name }}"
    argocd/app-name: ${{ values.name }}
    backstage.io/techdocs-ref: dir:.
  links:
    - title: Link to ArgoCD
      url: http://argo-cd.127.0.0.1.nip.io/applications/${{ values.name }}
spec:
  type: service
  owner: ${{ values.owner }}
  lifecycle: production
  providesApis:
    - ${{ values.name }}-api
```

usage of values in template files


Fetch RDS Database Template
Skipped
Open PR for new database
Skipped
Fetch Base
2 seconds
Publish
5 seconds
Register Service
1 second
Register API
0 seconds
Open Github Repository
Open Catalog
START OVER
Search
1 2023-04-16T17:02:38.289Z Beginning step Register API
2 2023-04-16T17:02:38.292Z info: Registering https://github.com/devops-syndicate/app-d/tree/main/api/api.yaml in the cata
3 2023-04-16T17:02:38.726Z Finished step Register API

COMPONENT — SERVICE

app-d ☆

Owner team-a

Lifecycle production

OVERVIEW | CI/CD | API | DEPENDENCIES | DOCS

About

VIEW SOURCE

VIEW TECHDOCS

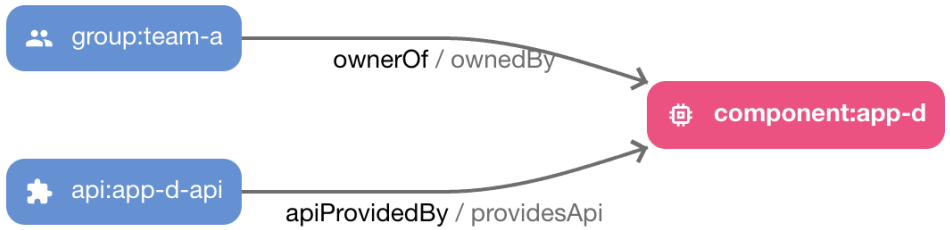
DESCRIPTION

No description

OWNER	SYSTEM	TYPE
team-a	**No System**	**service**

LIFECYCLE	TAGS
production	**No Tags**

Relations



View graph →

Default actions

- Load skeletons of code
- Template in some variables
- Publish template to some location
- Delete/Change files in workspace
- Register new component in backstage
- Add label to pull requests

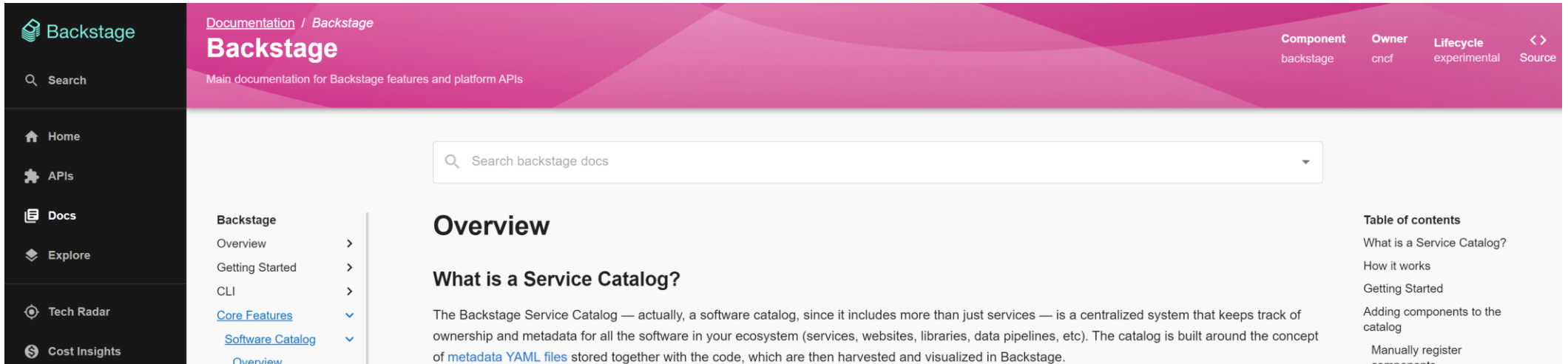
Custom action

```ts
import { createTemplateAction } from '@backstage/plugin-scaffolder-node';
import fs from 'fs-extra';
import { z } from 'zod';

export const createNewFileAction = () => {
  return createTemplateAction({
    id: 'acme:file:create',
    schema: {
      input: z.object({
        contents: z.string().describe('The contents of the file'),
        filename: z
          .string()
          .describe('The filename of the file that will be created'),
      }),
    },

    async handler(ctx) {
      await fs.outputFile(
        `${ctx.workspacePath}/${ctx.input.filename}`,
        ctx.input.contents,
      );
    },
  });
};
```

Tech Docs

Docs as Code

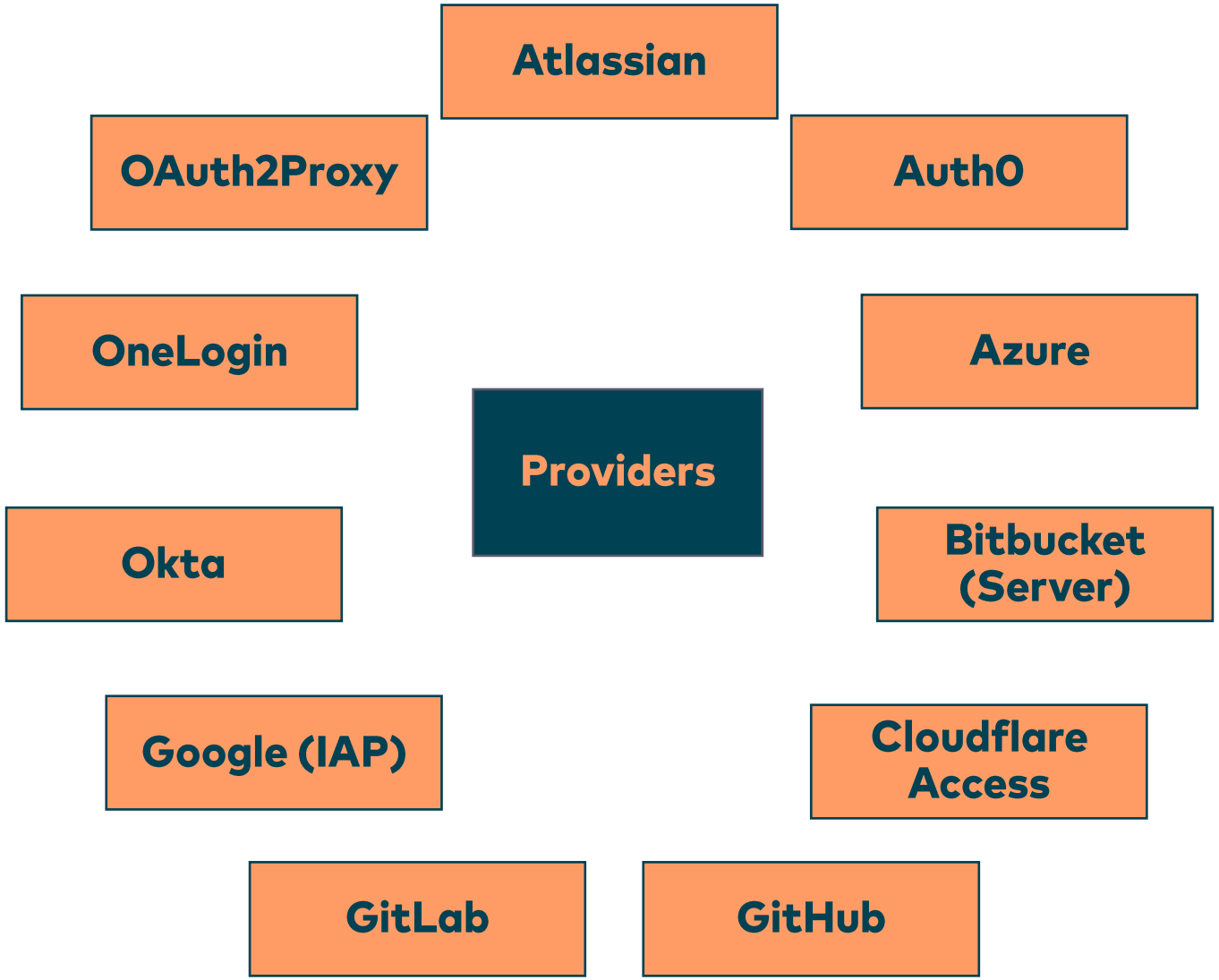
- MkDocs based Static Site Generator integrated into Backstage
- Docker-based Builds, easy setup
- Ingests docs into integrated search framework
- Layout is customizable



Stop!
Demo Time

Authentication

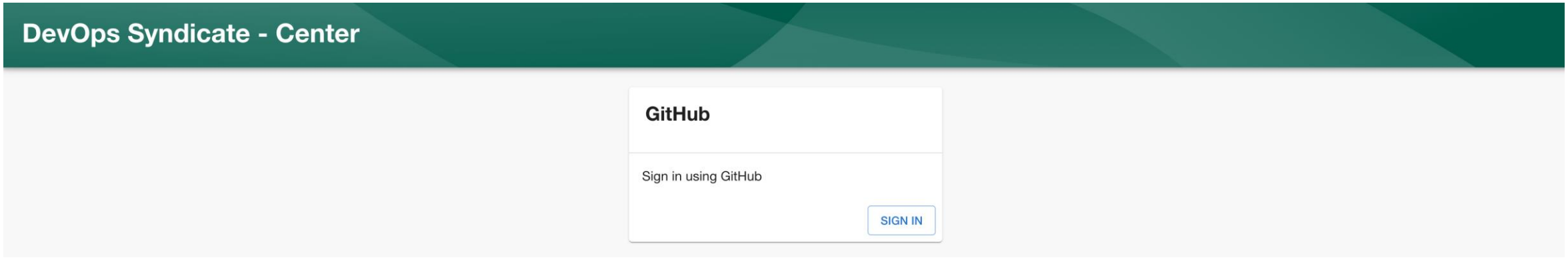
Authentication Providers



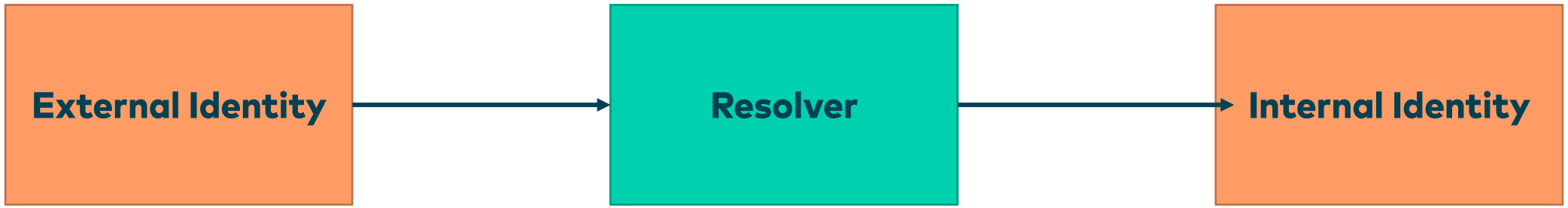
Login Page

```
const app = createApp({
  components: {
    SignInPage: props => (
      <SignInPage
        {...props}
        auto
        provider={{
          id: 'github-auth-provider',
          title: 'GitHub',
          message: 'Sign in using GitHub',
          apiRef: githubAuthApiRef,
        }}
      />
    ),
  },
  ...
});
```

Login Page


DevOps Syndicate - Center
GitHub
Sign in using GitHub
SIGN IN

Identity Resolver



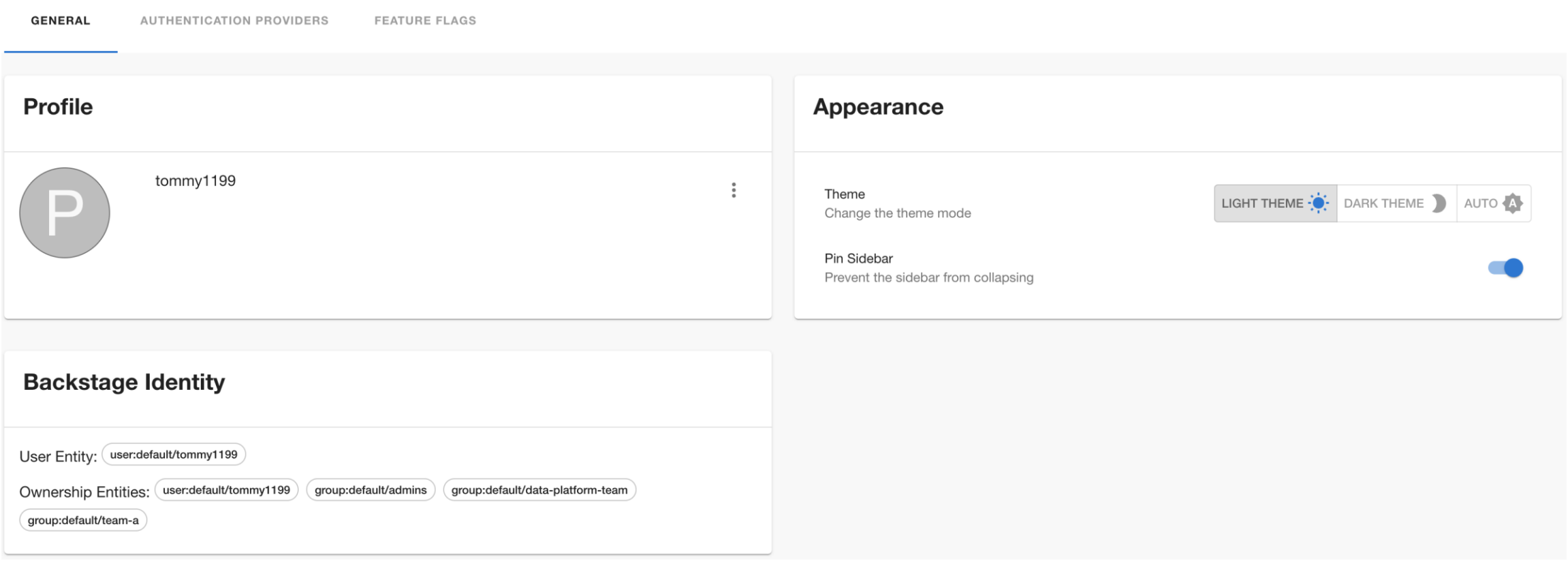
Simple Identity Resolver

```
signIn: {
  resolver(_, ctx) {
    const userRef = 'user:default/guest'
    return ctx.issueToken({
      claims: {
        sub: userRef,
        ent: [userRef],
      },
    }),
  },
},
```

Github Identity Resolver

```
signIn: {
   resolver: providers.github.resolvers.usernameMatchingUserEntityName(),
},
```

Github Identity Resolver



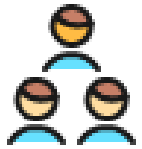
Stop!
Demo Time

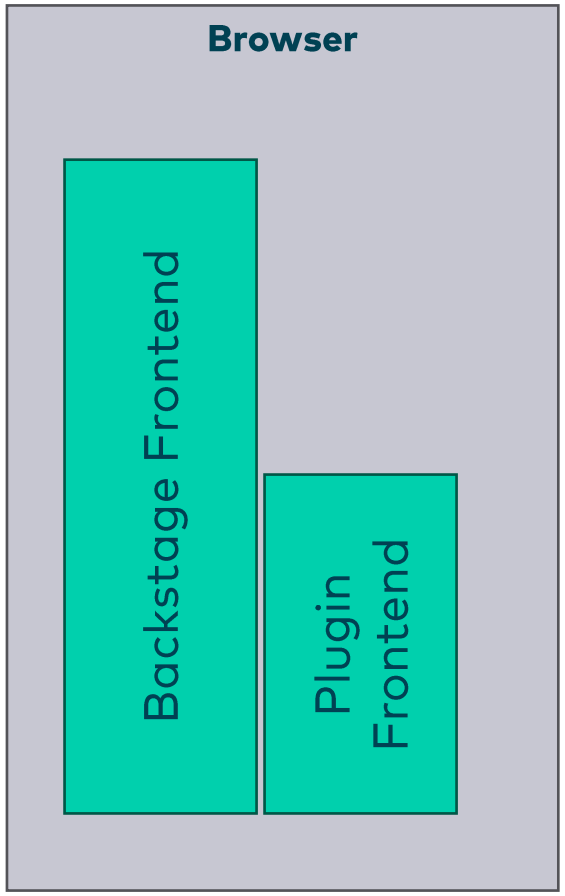
Plugins

Extensibility API

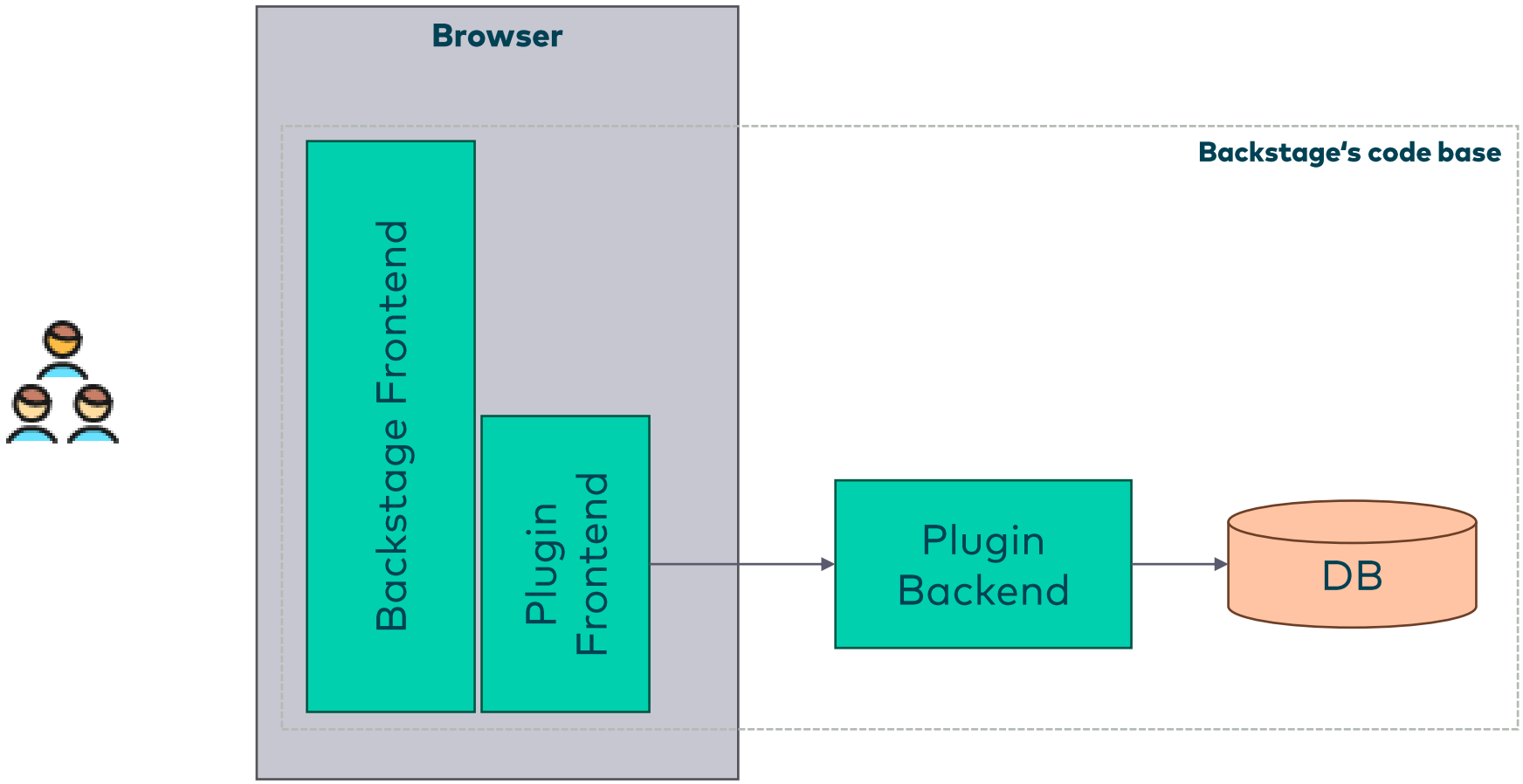
- Access to Entities
- Access to Logged-In User info
- Access to the catalog
- Access to configuration
- Own UI components
- To communicate with external systems, use the proxy-backend to avoid CORS
- Can be a full page plugin or integrated in the software catalog as tile or tab

Frontend Integration

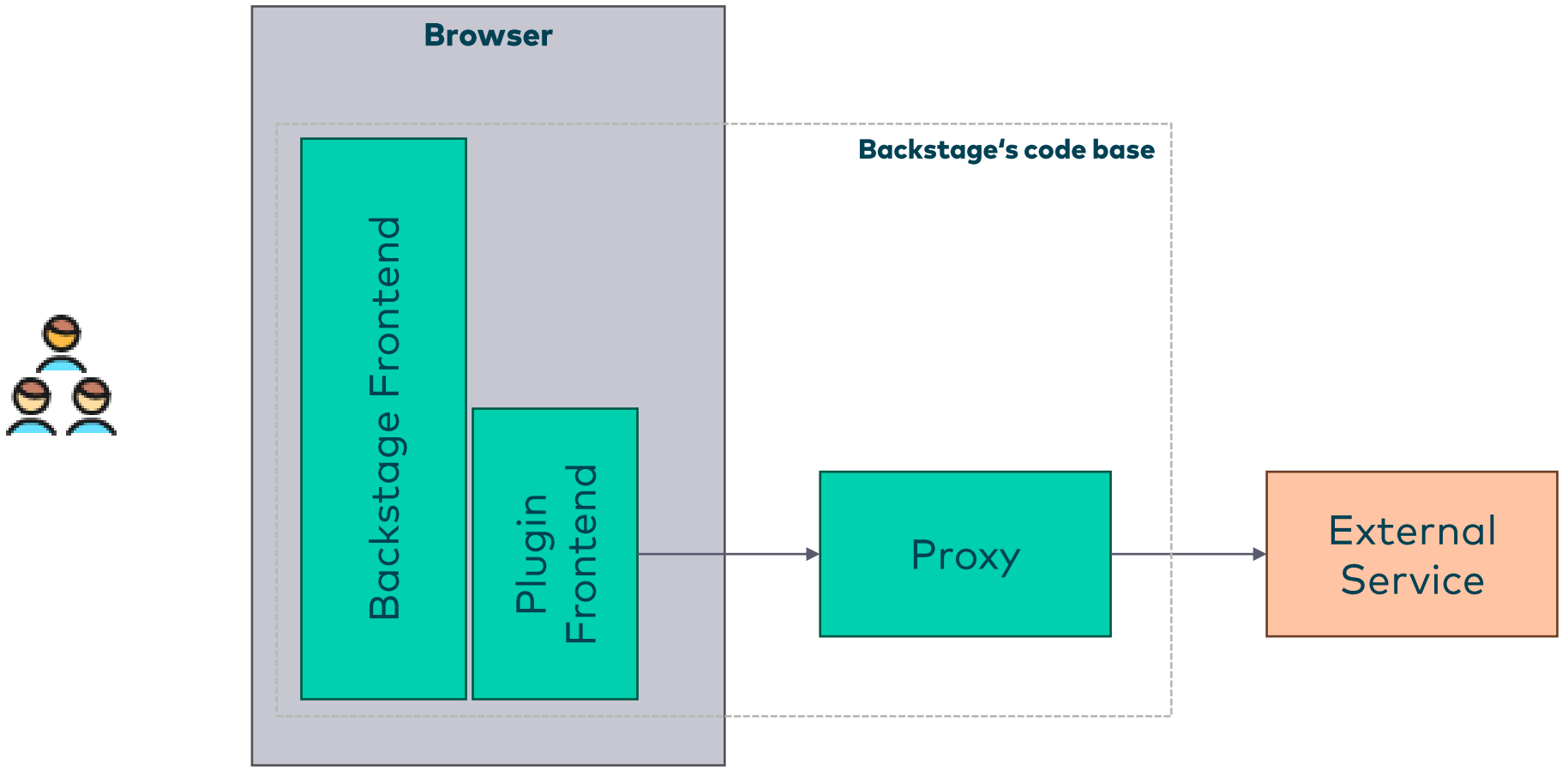




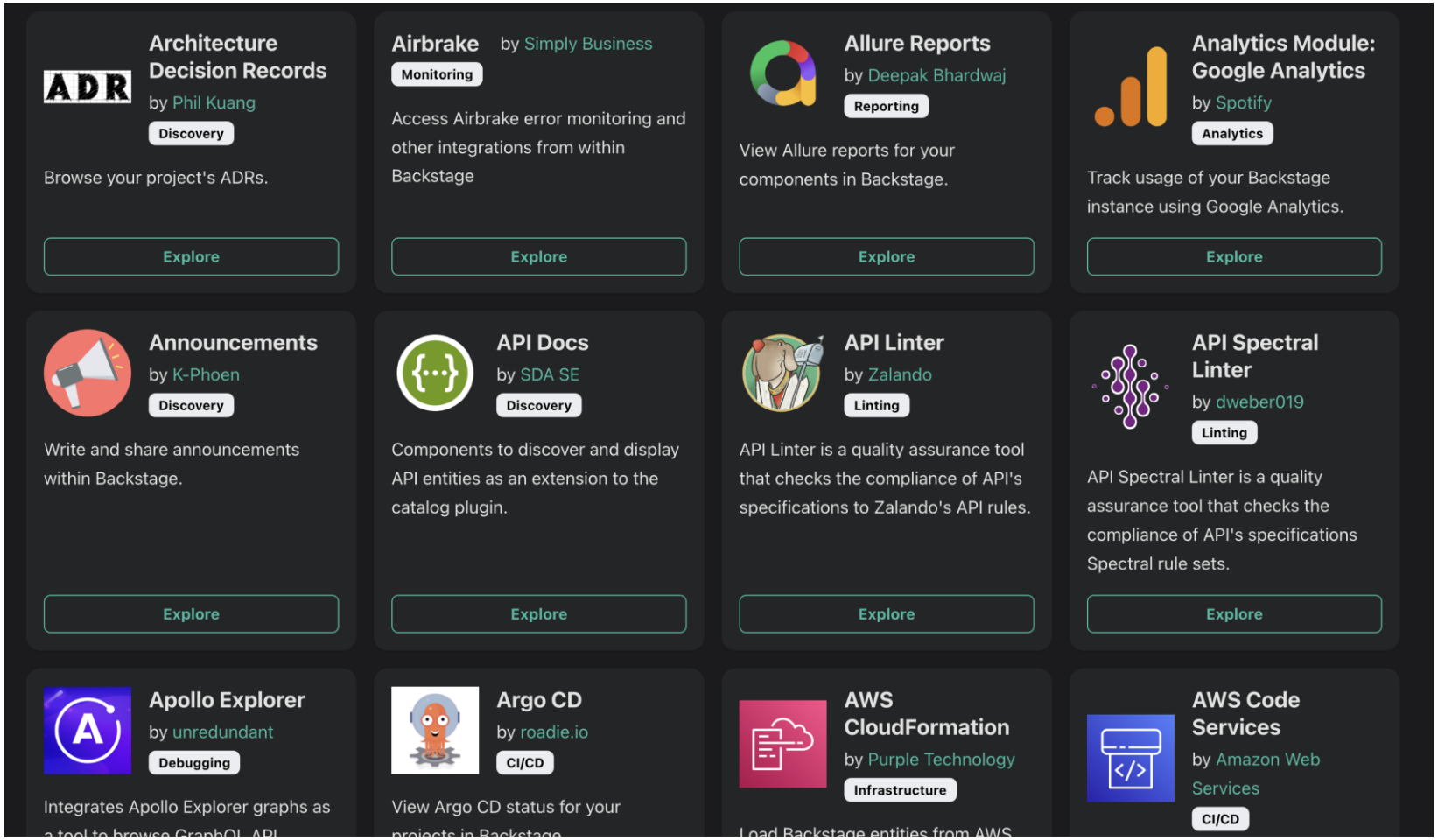
Integration with own Backend



Integration via Proxy

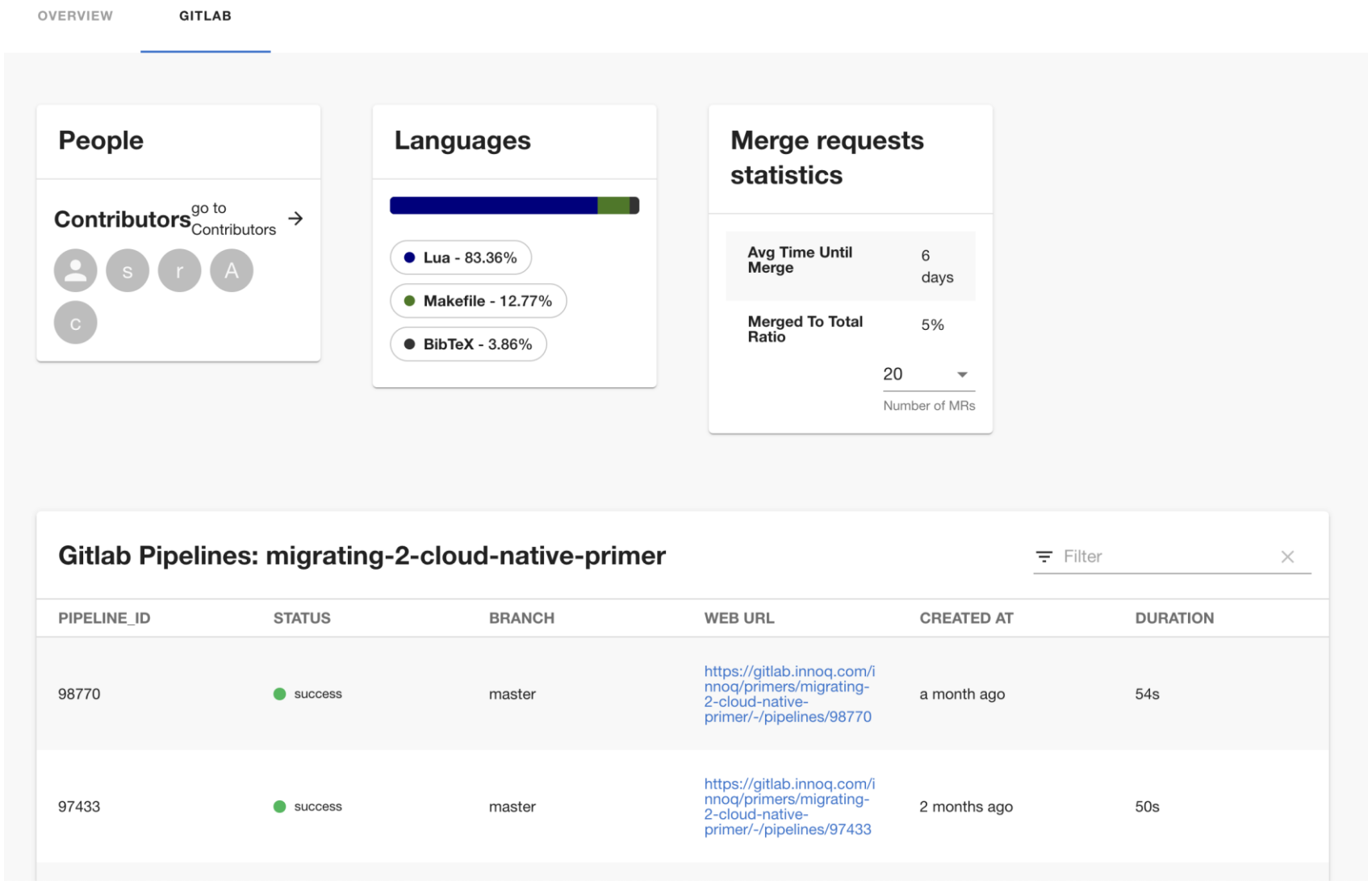


Plugin Marketplace

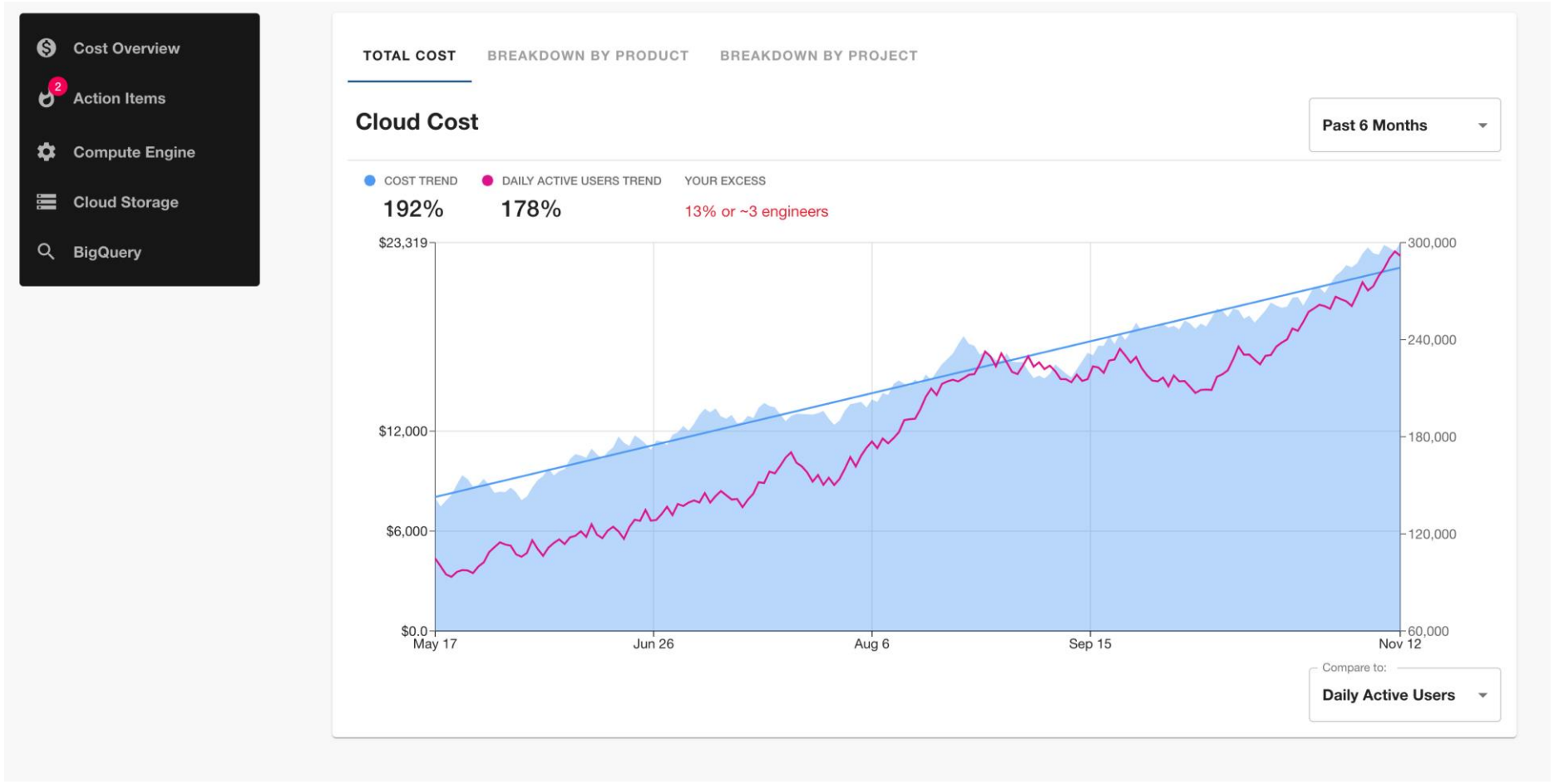


https://backstage.io/plugins

Gitlab Plugin



Cost Insights Plugin



Conclusions

Conclusions

- Backstage provides a decent framework to easily build and run developer portals within your organization
- Extensible catalog meta model
- Extensible tooling via Plugins
- Single entrypoint of knowledge without the downsides of a central catalog repository

Danke! Fragen?

INNOQ
www.innoq.com

Tammo van Lessen
tammo.van-lessen@innoq.com
@taval

Sascha Selzer
sascha.selzer@innoq.com
@tommy1199

innoQ Deutschland GmbH

Krischerstr. 100
40789 Monheim
+49 2173 3366-0

Ohlauer Str. 43
10999 Berlin

Ludwigstr. 180E
63067 Offenbach

Kreuzstr. 16
80331 München

Hermannstrasse 13
20095 Hamburg

Erftstr. 15-17
50672 Köln

Königstorgraben 11
90402 Nürnberg