PORT SUPPPORT ENGINEER ASSIGNMENT

Candidate: Asomugha Divine

Date: 11/08/2025

Table of Contents

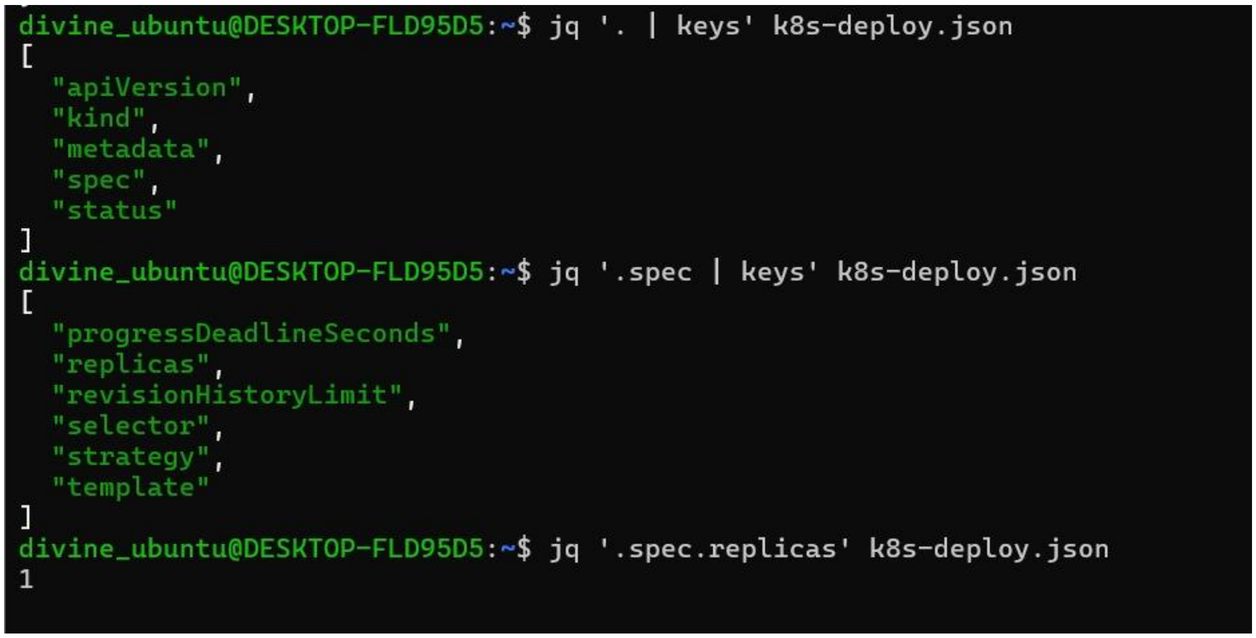
1. Exercise #1 – JQ Patterns
2. Exercise #2 – Jira & GitHub Integration
3. Exercise #3 – PR Count Scorecard
4. Exercise #4 – GitHub Workflow Troubleshooting

Exercise #1 – JQ Patterns.

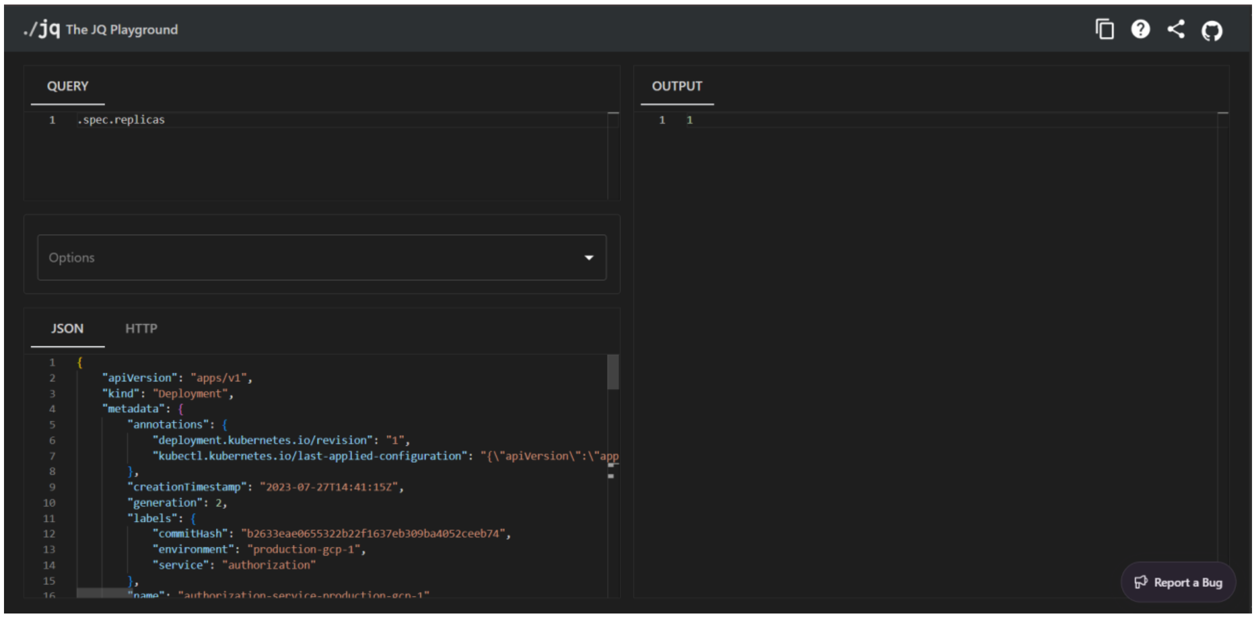
1(a) **Extract current replica count**

```.spec.replicas```

Explanation: `.spec.replicas` points to the integer value of the current replica count in a Kubernetes Deployment object.



```
divine_ubuntu@DESKTOP-FLD95D5:~$ jq '. | keys' k8s-deploy.json
[
  "apiVersion",
  "kind",
  "metadata",
  "spec",
  "status"
]
divine_ubuntu@DESKTOP-FLD95D5:~$ jq '.spec | keys' k8s-deploy.json
[
  "progressDeadlineSeconds",
  "replicas",
  "revisionHistoryLimit",
  "selector",
  "strategy",
  "template"
]
divine_ubuntu@DESKTOP-FLD95D5:~$ jq '.spec.replicas' k8s-deploy.json
1
```



```
.spec.replicas
```

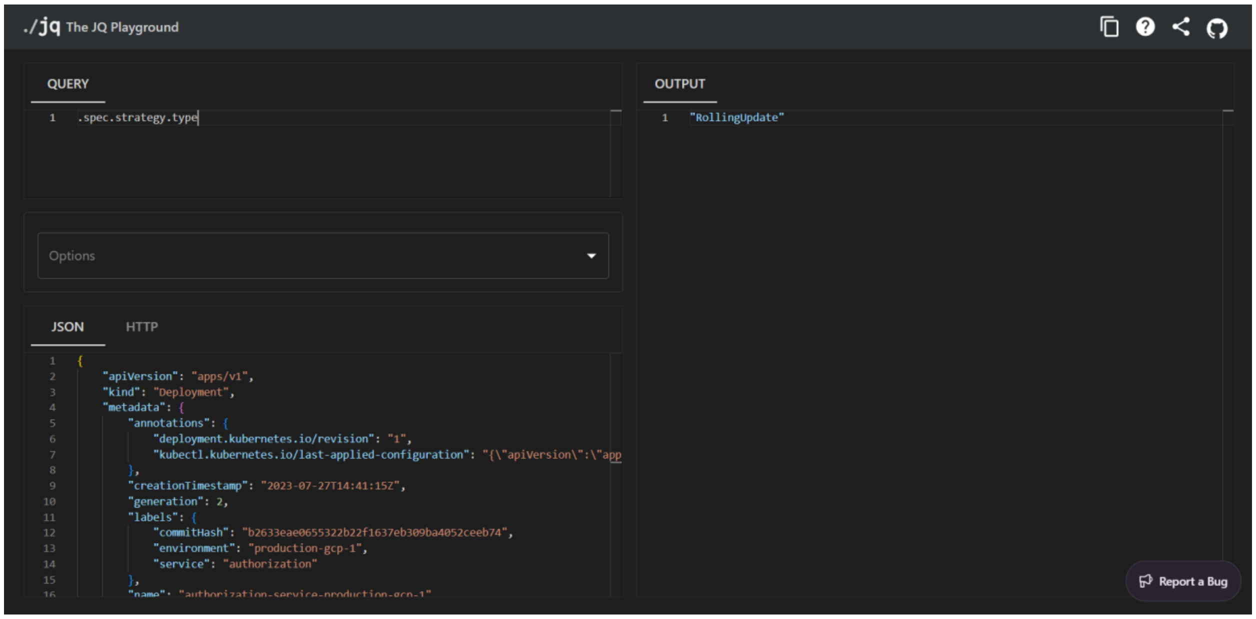
```
1
```

```
{
    "apiVersion": "apps/v1",
    "kind": "Deployment",
    "metadata": {
        "annotations": {
            "deployment.kubernetes.io/revision": "1",
            "kubectl.kubernetes.io/last-applied-configuration": "{\"apiVersion\":\"app
        },
        "creationTimestamp": "2023-07-27T14:41:15Z",
        "generation": 2,
        "labels": {
            "commitHash": "b2633eae0655322b22f1637eb309ba4052ceeb74",
            "environment": "production-gcp-1",
            "service": "authorization"
        },
        "name": "authorization-service-production-gcp-1"
```

1(b) **Extract deployment strategy**

```.spec.strategy.type```

Explanation: The deployment strategy type (e.g., `RollingUpdate` or `Recreate`) is found under `.spec.strategy.type`.



```
.spec.strategy.type
```

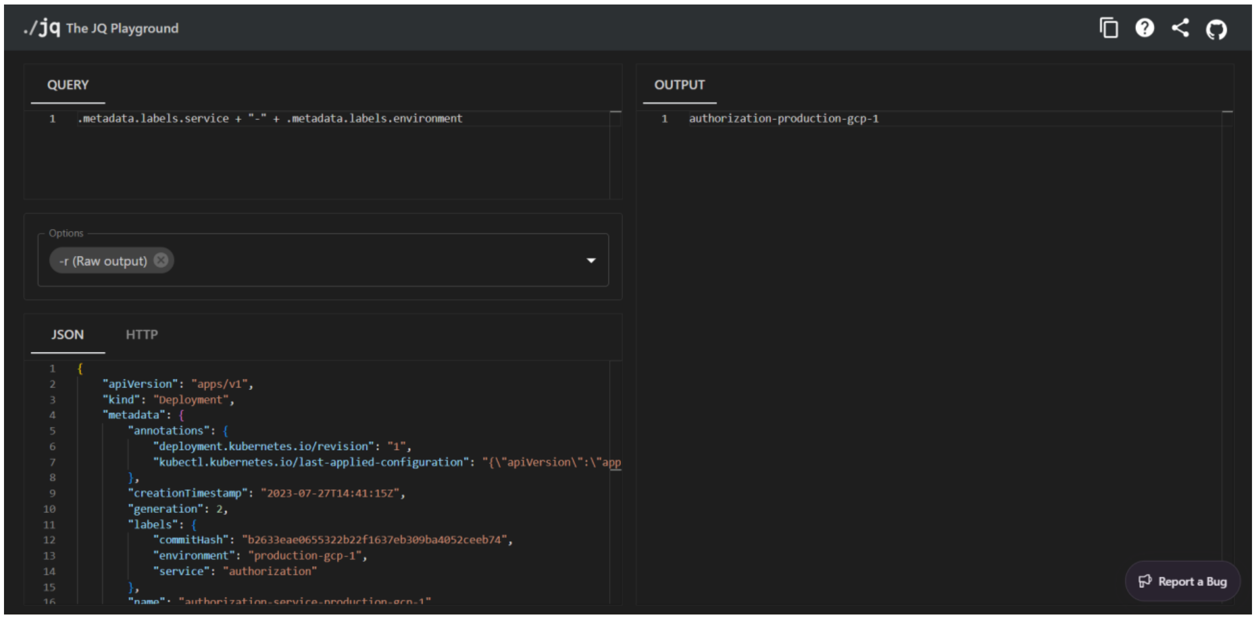
```
"RollingUpdate"
```

```
{
    "apiVersion": "apps/v1",
    "kind": "Deployment",
    "metadata": {
        "annotations": {
            "deployment.kubernetes.io/revision": "1",
            "kubectl.kubernetes.io/last-applied-configuration": "{\"apiVersion\":\"app
        },
        "creationTimestamp": "2023-07-27T14:41:15Z",
        "generation": 2,
        "labels": {
            "commitHash": "b2633eae0655322b22f1637eb309ba4052ceeb74",
            "environment": "production-gcp-1",
            "service": "authorization"
        },
        "name": "authorization-service-production-gcp-1"
```

1(c) **Extract the "service" label of the deployment concatenated with the "environment" label of the deployment, with a hyphen (-) in the middle.**

```.metadata.labels.service + "-" + .metadata.labels.environment```

Explanation: Accesses the two label values and concatenates them into a single string.



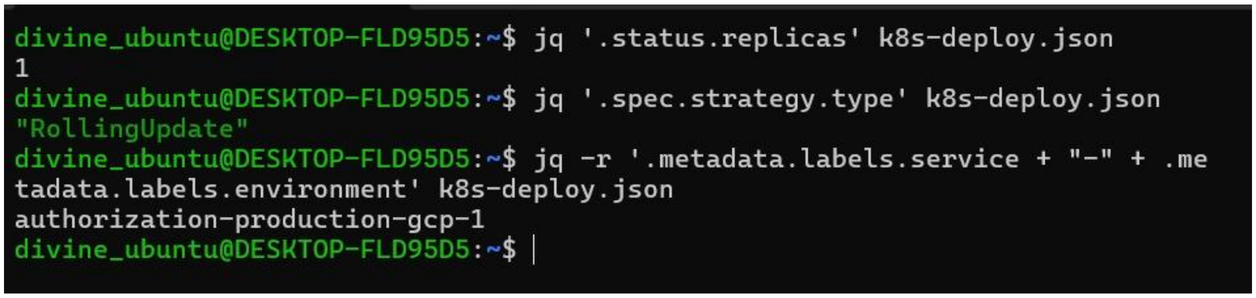
```
.metadata.labels.service + "-" + .metadata.labels.environment
```

-r (Raw output)

```
authorization-production-gcp-1
```

```
{
    "apiVersion": "apps/v1",
    "kind": "Deployment",
    "metadata": {
        "annotations": {
            "deployment.kubernetes.io/revision": "1",
            "kubectl.kubernetes.io/last-applied-configuration": "{\"apiVersion\":\"app
        },
        "creationTimestamp": "2023-07-27T14:41:15Z",
        "generation": 2,
        "labels": {
            "commitHash": "b2633eae0655322b22f1637eb309ba4052ceeb74",
            "environment": "production-gcp-1",
            "service": "authorization"
        },
        "name": "authorization-service-production-gcp-1"
```

Image proof from local pc

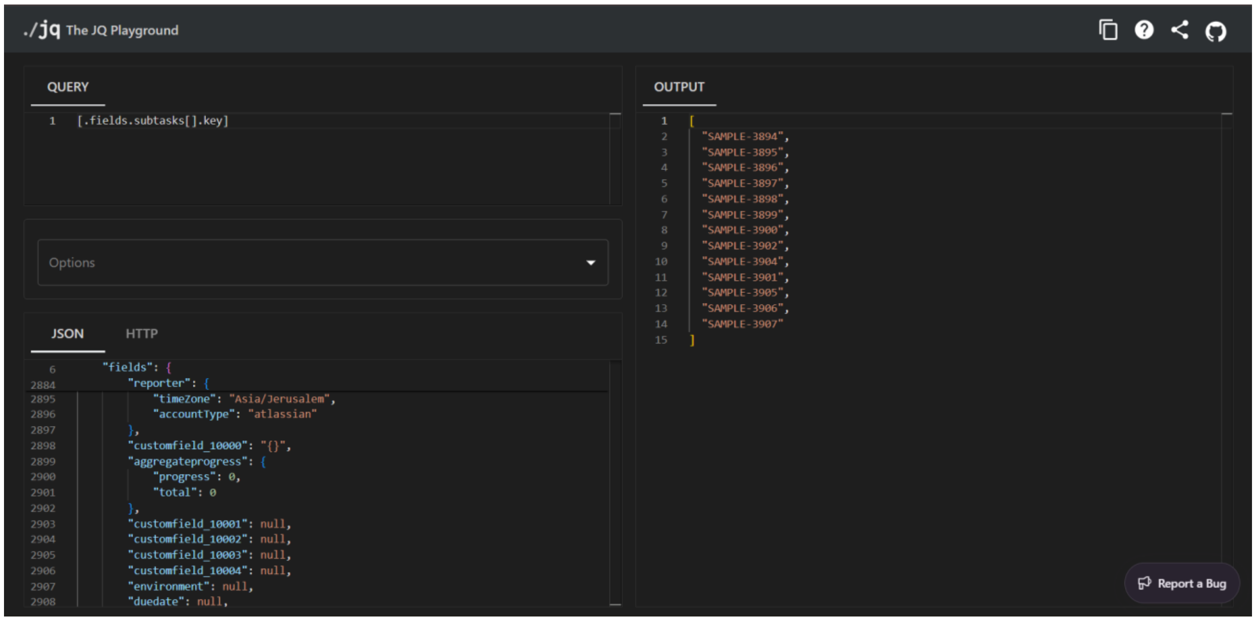


```
divine_ubuntu@DESKTOP-FLD95D5:~$ jq '.status.replicas' k8s-deploy.json
1
divine_ubuntu@DESKTOP-FLD95D5:~$ jq '.spec.strategy.type' k8s-deploy.json
"RollingUpdate"
divine_ubuntu@DESKTOP-FLD95D5:~$ jq -r '.metadata.labels.service + "-" + .me
tadata.labels.environment' k8s-deploy.json
authorization-production-gcp-1
divine_ubuntu@DESKTOP-FLD95D5:~$ 
```

2- Extract all the issue IDs (for example SAMPLE-123) for all subtasks, in an array.

```[.fields.subtasks[].key]```

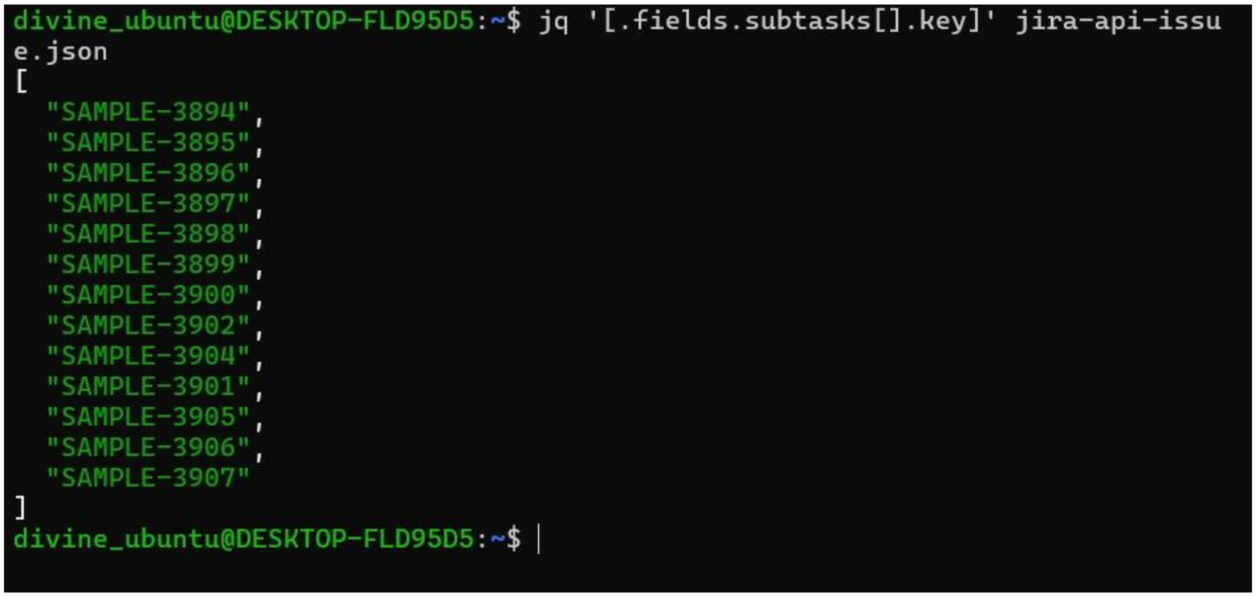
Explanation: `.fields.subtasks` is an array of subtasks; selecting `.key` from each returns their IDs. Wrapping with `[...]` ensures an array output:



```
[.fields.subtasks[].key]
```

```
    "fields": {
        "reporter": {
            "timeZone": "Asia/Jerusalem",
            "accountType": "atlassian"
        },
        "customfield_10000": "{}",
        "aggregateprogress": {
            "progress": 0,
            "total": 0
        },
        "customfield_10001": null,
        "customfield_10002": null,
        "customfield_10003": null,
        "customfield_10004": null,
        "environment": null,
        "duedate": null,
```

```
[
  "SAMPLE-3894",
  "SAMPLE-3895",
  "SAMPLE-3896",
  "SAMPLE-3897",
  "SAMPLE-3898",
  "SAMPLE-3899",
  "SAMPLE-3900",
  "SAMPLE-3902",
  "SAMPLE-3904",
  "SAMPLE-3901",
  "SAMPLE-3905",
  "SAMPLE-3906",
  "SAMPLE-3907"
]
```



```
divine_ubuntu@DESKTOP-FLD95D5:~$ jq '[.fields.subtasks[].key]' jira-api-issu
e.json
[
  "SAMPLE-3894",
  "SAMPLE-3895",
  "SAMPLE-3896",
  "SAMPLE-3897",
  "SAMPLE-3898",
  "SAMPLE-3899",
  "SAMPLE-3900",
  "SAMPLE-3902",
  "SAMPLE-3904",
  "SAMPLE-3901",
  "SAMPLE-3905",
  "SAMPLE-3906",
  "SAMPLE-3907"
]
divine_ubuntu@DESKTOP-FLD95D5:~$
```

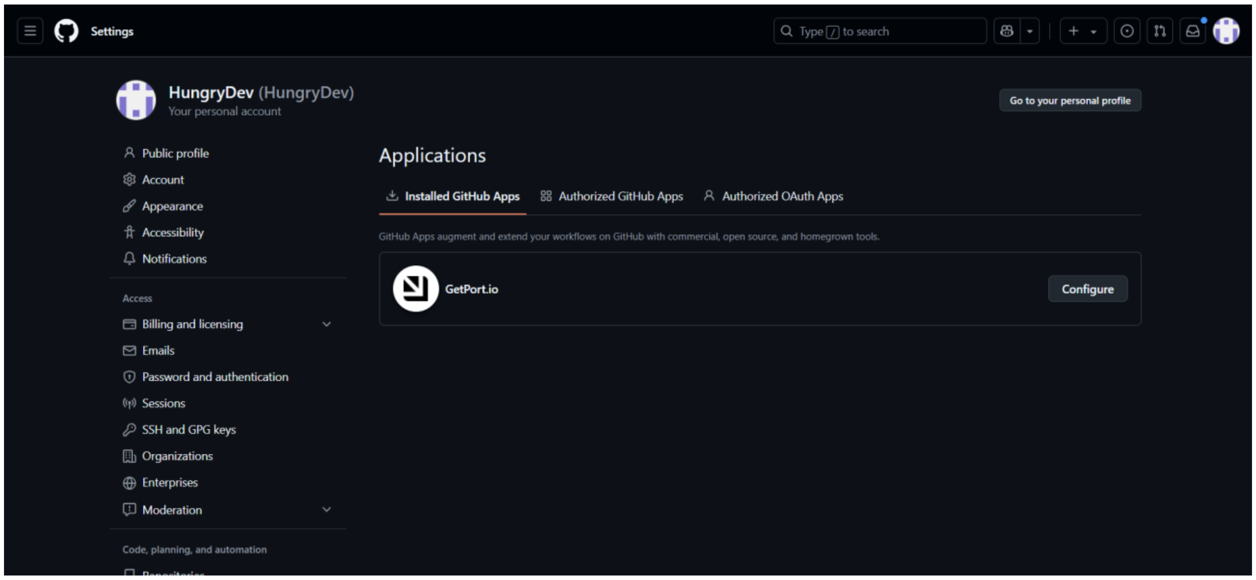
Exercise #2 – Jira & GitHub Integration.

Steps Taken:

1. Signed Up to Port with this email. divine+port@solmateai.site

2. Installed Port's GitHub app and authorized repos.

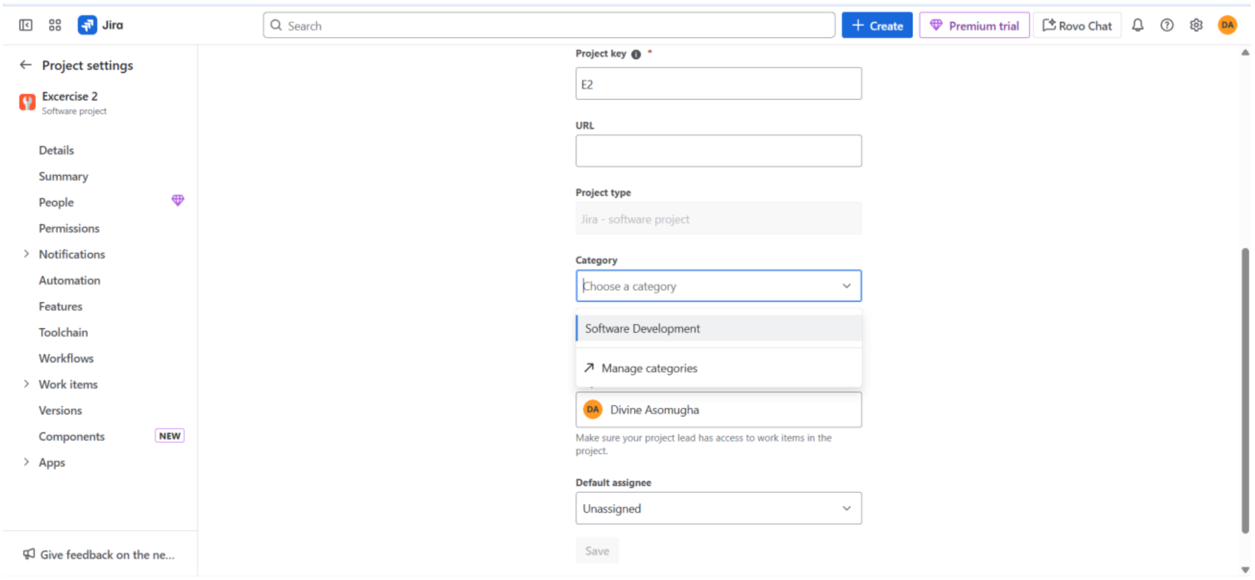
Image below shows the installed getport github application



GitHub Apps augment and extend your workflows on GitHub with commercial, open source, and homegrown tools.

3. Created Jira project (Excercise2) with Scrum template (company-managed).

Images below shows the created Jira project



Project key *: E2

URL:

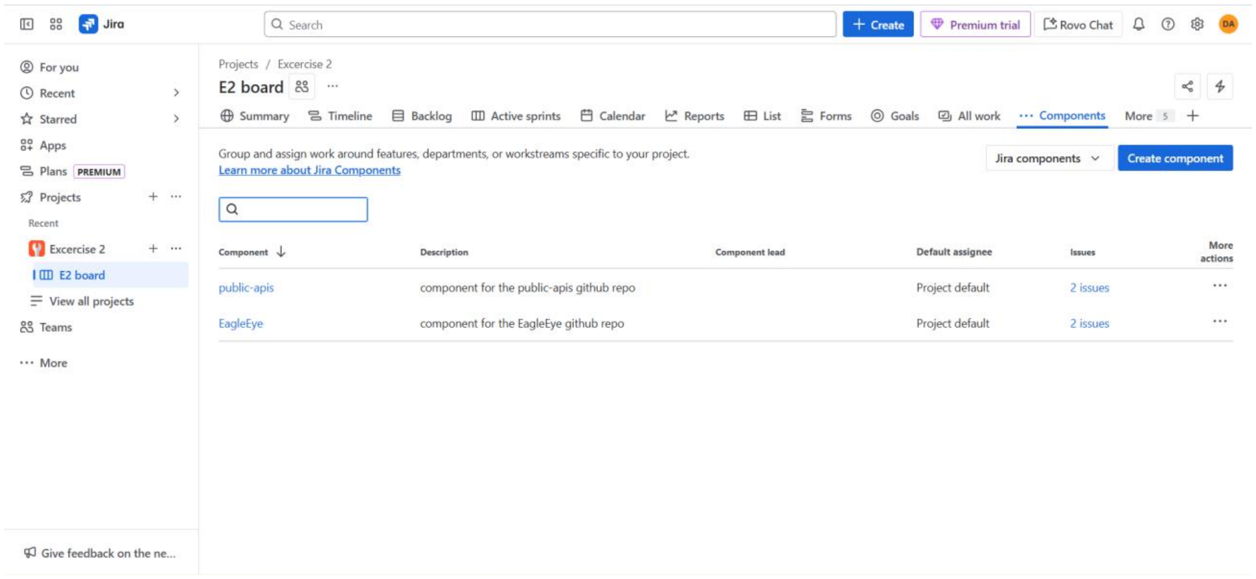
Project type: Jira - software project

Category: Choose a category
- Software Development
- Manage categories

Divine Asomugha

Make sure your project lead has access to work items in the project.

Default assignee: Unassigned


Projects / Excercise 2

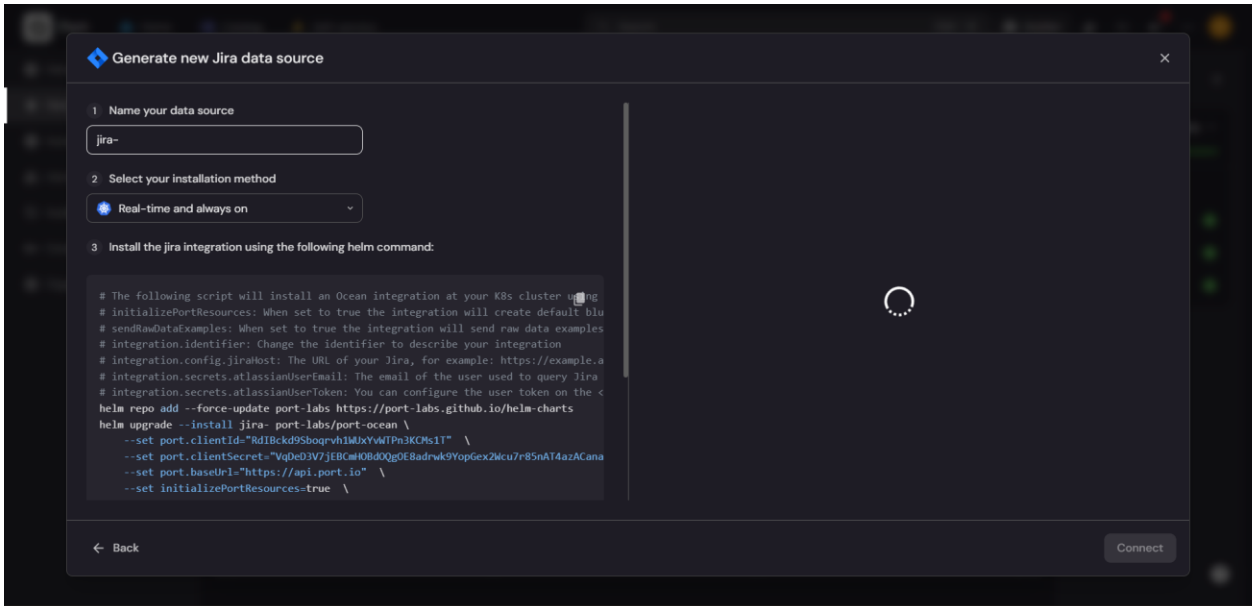
E2 board

Group and assign work around features, departments, or workstreams specific to your project.
Learn more about Jira Components

Component	Description	Component lead	Default assignee	Issues	More actions
public-apis	component for the public-apis github repo		Project default	2 issues	···
EagleEye	component for the EagleEye github repo		Project default	2 issues	···

4. Installed Port Ocean Integration for Jira via Helm.

Image below shows the installation instruction for Port Ocean Integration for Jira


Generate new Jira data source

1 Name your data source: jira-

2 Select your installation method: Real-time and always on

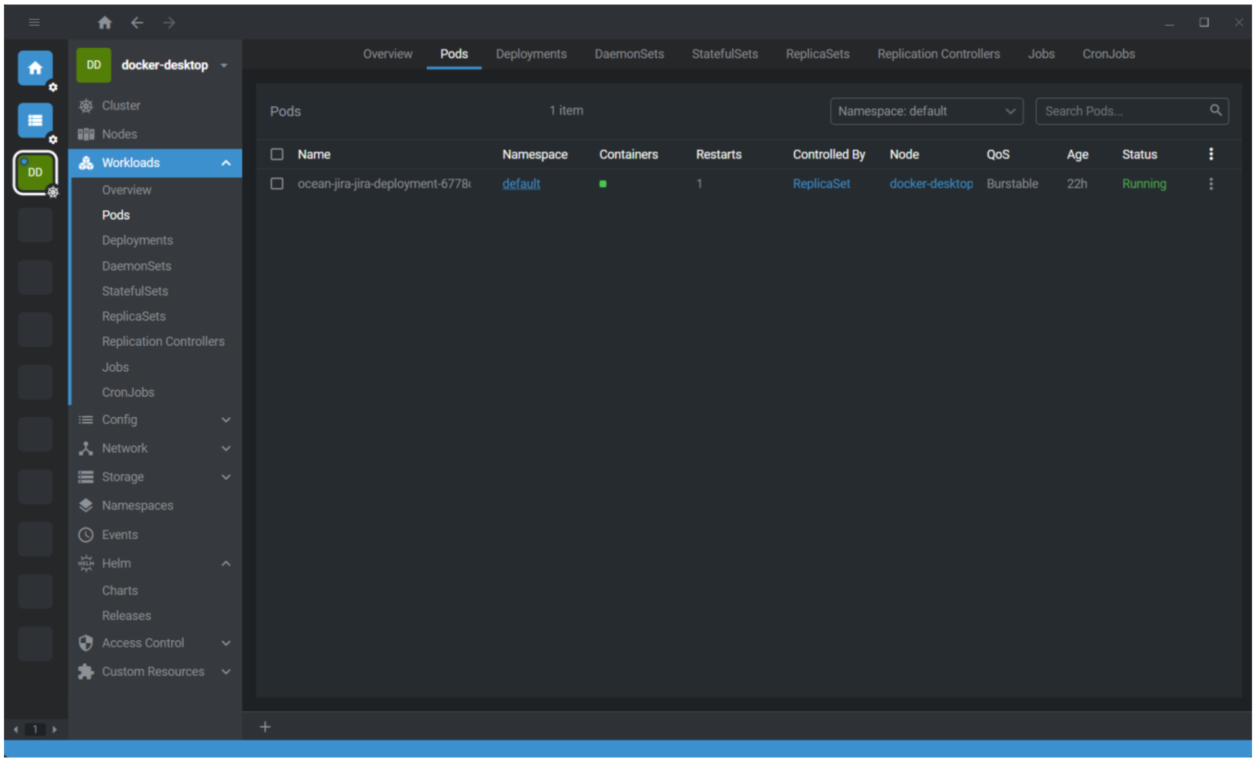
3 Install the jira integration using the following helm command:

```
# The following script will install an Ocean integration at your K8s cluster using
# initializePortResources: When set to true the integration will create default blu
# sendRawDataExamples: When set to true the integration will send raw data examples
# integration.identifier: Change the identifier to describe your integration
# integration.config.jiraHost: The URL of your Jira, for example: https://example.a
# integration.secrets.atlassianUserEmail: The email of the user used to query Jira
# integration.secrets.atlassianUserToken: You can configure the user token on the <
helm repo add --force-update port-labs https://port-labs.github.io/helm-charts
helm upgrade --install jira- port-labs/port-ocean \
    --set port.clientId="RdIBckd9Sboqrvh1WUxYvWTPn3KCMs1T"  \
    --set port.clientSecret="VqDeD3V7jEBCmHOBdOQgOE8adrwk9YopGex2Wcu7r85nAT4azACana
    --set port.baseUrl="https://api.port.io"  \
    --set initializePortResources=true  \
```

Image below shows the helm command for the Port Ocean Integrator deployment on Kubernetes

```
helm repo add --force-update port-labs https://port-labs.github.io/helm-charts
helm upgrade --install jira port-labs/port-ocean \
        --set port.clientId="RdIBckd9Sboqrvh1WUxYvWTPn3KCMs1T"  \
        --set port.clientSecret="VqDeD3V7jEBCmHOBdOQgOE8adrwk9YopGex2Wcu7r85nAT4azACanaaThOTvh03x"  \
        --set port.baseUrl="https://api.port.io"  \
        --set initializePortResources=true  \
        --set sendRawDataExamples=true  \
        --set scheduledResyncInterval=360  \
        --set integration.identifier="jira"  \
        --set integration.type="jira"  \
        --set integration.eventListener.type="POLLING"  \
        --set integration.config.jiraHost="https://hyperflox.atlassian.net"  \
        --set integration.secrets.atlassianUserEmail="divine@solmateai.site"  \
        --set integration.secrets.atlassianUserToken="ATATT3xFfGF01nvRG-dzzI__3ejl4B-
io4uyW2wSafciECz_MDZMb2q5V4ygaKl_bneffffedddddddddddddddddddddddddddddddddddddddd5eWDxonJxNTouJ2eZQ1VzaPn5JjhjuFqOSqE9dQPm87vbiVWk9SEBD4f
v4CMEk3-pWsPMTJYkT4Xw=F210E987"
```

Image below shows the Port Ocean Integrator pod running in Kubernetes

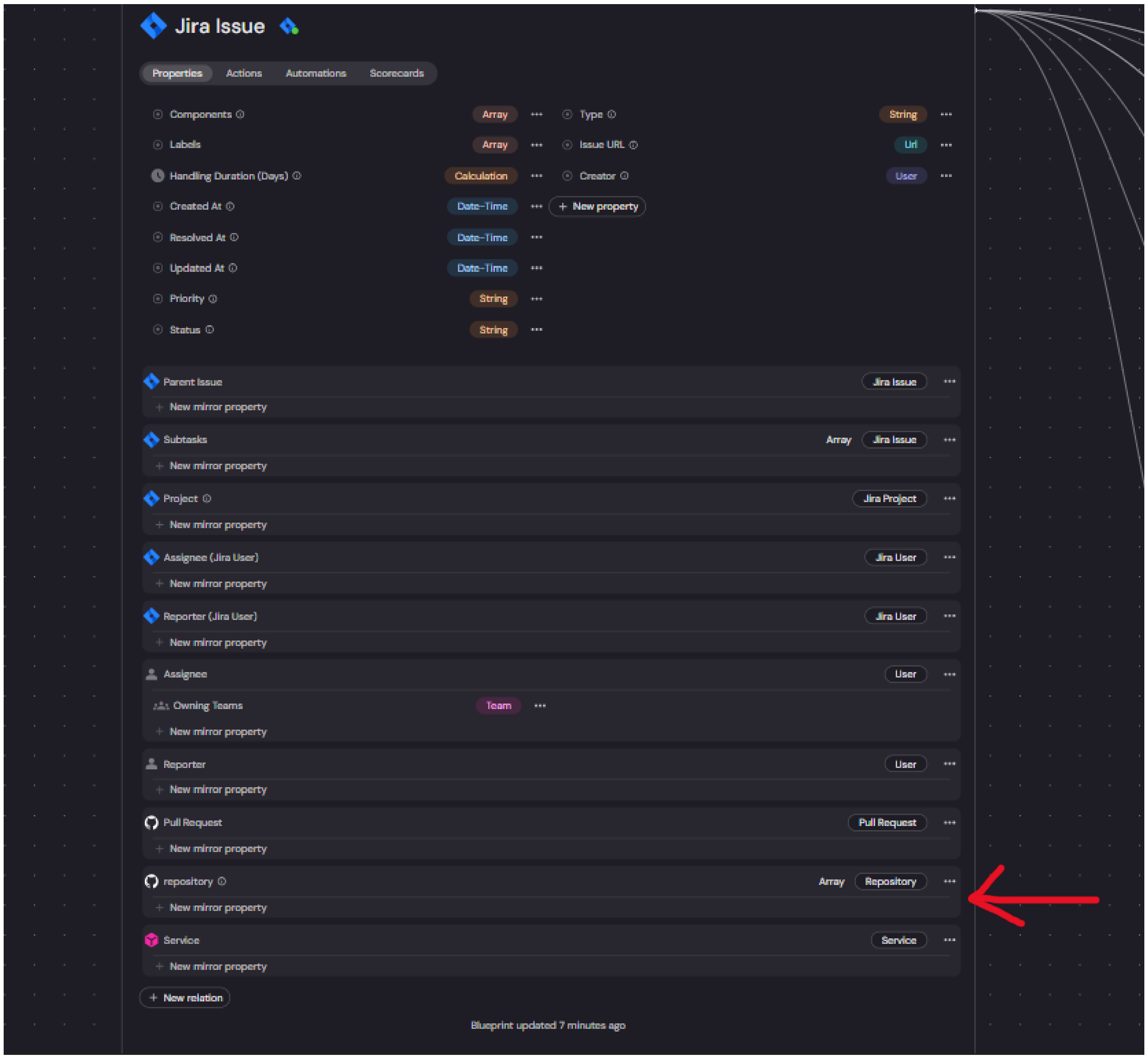


Pods 1 item

Name	Namespace	Containers	Restarts	Controlled By	Node	QoS	Age	Status
ocean-jira-jira-deployment-6778	default	🟩	1	ReplicaSet	docker-desktop	Burstable	22h	Running

5. Added relation Jira Issue → Repository.

Image below shows the added relation btw Jira issues and GitHub repositories


Jira Issue

Properties · Actions · Automations · Scorecards

Components	Array		Type	String
Labels	Array		Issue URL	Url
Handling Duration (Days)	Calculation		Creator	User
Created At	Date-Time		+ New property	
Resolved At	Date-Time			
Updated At	Date-Time			
Priority	String			
Status	String			

Parent Issue		Jira Issue
Subtasks	Array	Jira Issue
Project		Jira Project
Assignee (Jira User)		Jira User
Reporter (Jira User)		Jira User
Assignee		User
Owning Teams	Team	
Reporter		User
Pull Request		Pull Request
repository	Array	Repository
Service		Service

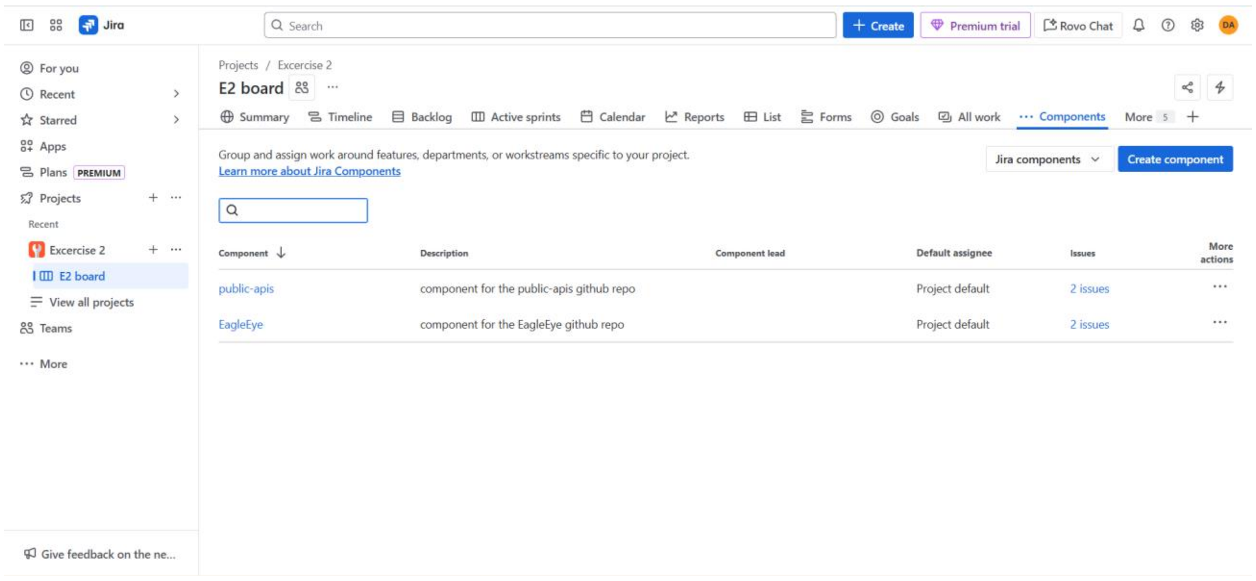
+ New mirror property

+ New relation

Blueprint updated 7 minutes ago

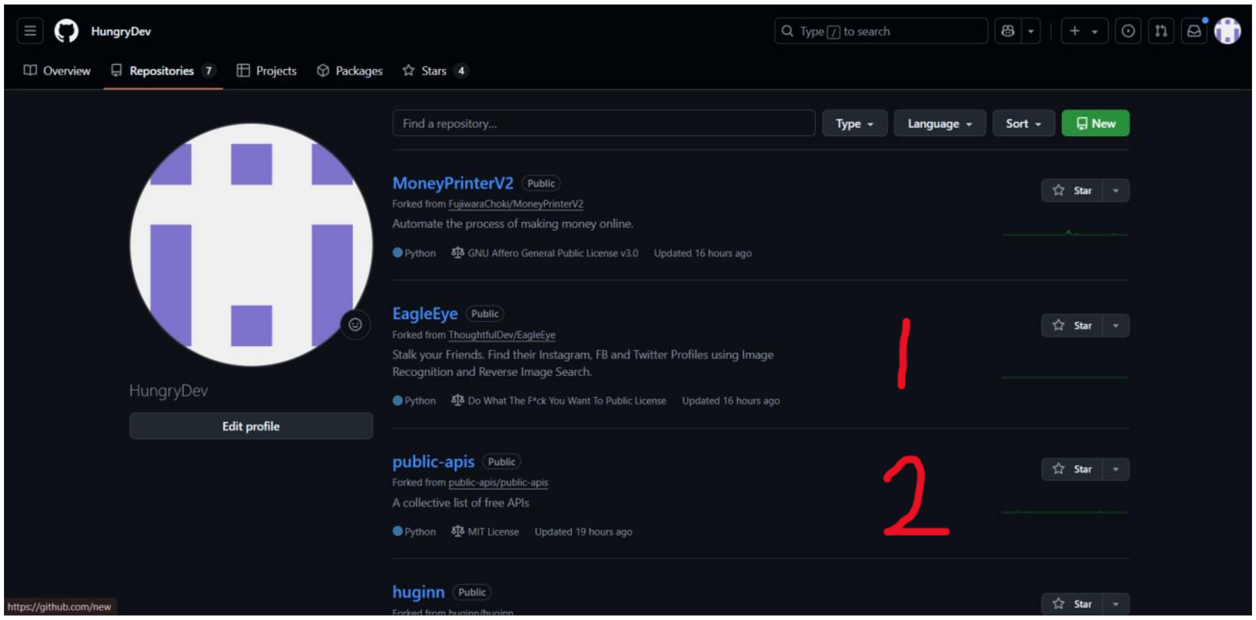
6. Created matching Jira components for repos.

Images below show the components and the repositories with matching names



Group and assign work around features, departments, or workstreams specific to your project.
Learn more about Jira Components

Component	Description	Component lead	Default assignee	Issues	More actions
public-apis	component for the public-apis github repo		Project default	2 issues	···
EagleEye	component for the EagleEye github repo		Project default	2 issues	···



MoneyPrinterV2 Public
Forked from FujiwaraChoki/MoneyPrinterV2
Automate the process of making money online.
Python GNU Affero General Public License v3.0 Updated 16 hours ago

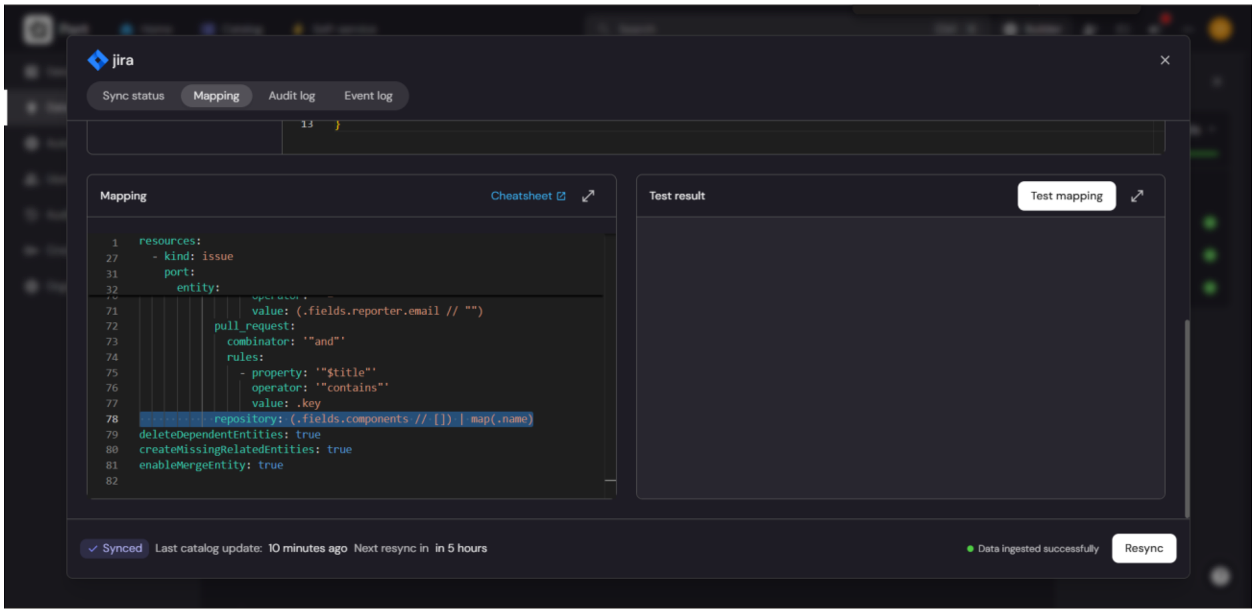
EagleEye Public
Forked from ThoughtfulDev/EagleEye
Stalk your Friends. Find their Instagram, FB and Twitter Profiles using Image Recognition and Reverse Image Search.
Python Do What The F*ck You Want To Public License Updated 16 hours ago

public-apis Public
Forked from public-apis/public-apis
A collective list of free APIs
Python MIT License Updated 19 hours ago

huginn Public

7. Updated Jira integration mapping.

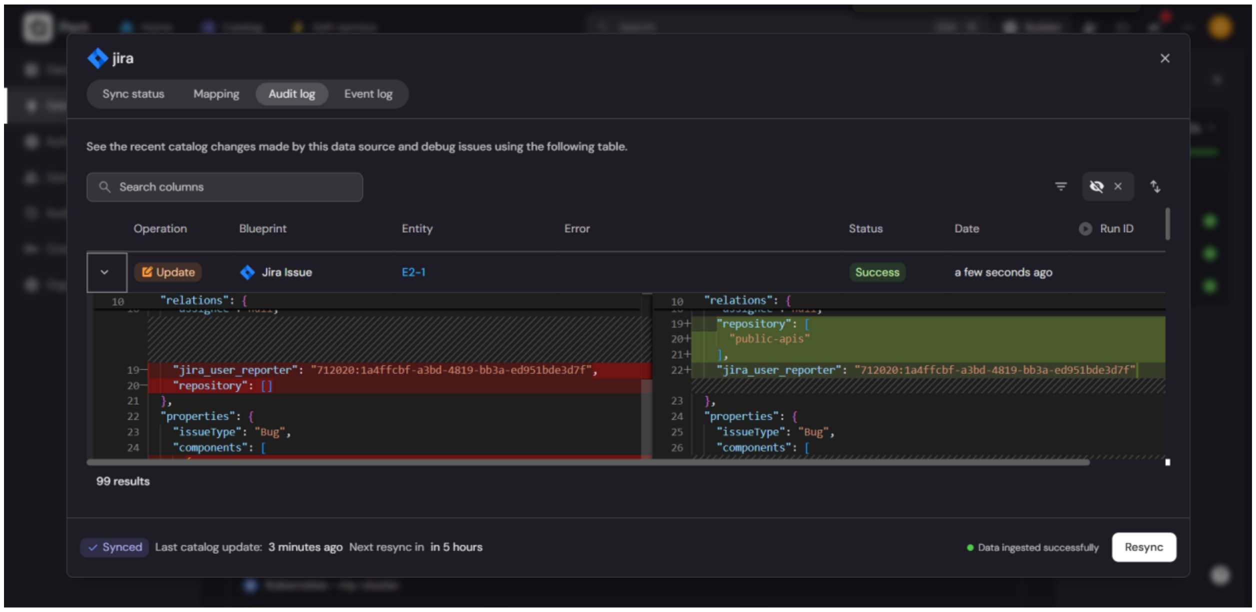
Image below shows the Jira issue updated mapping



```
resources:
  - kind: issue
    port:
      entity:
                  value: (.fields.reporter.email // "")
              pull_request:
                combinator: '"and"'
                rules:
                  - property: '"$title"'
                    operator: '"contains"'
                    value: .key
              repository: (.fields.components // []) | map(.name)
deleteDependentEntities: true
createMissingRelatedEntities: true
enableMergeEntity: true
```

Synced
Last catalog update: 10 minutes ago Next resync in in 5 hours
Data ingested successfully
Resync

Image below shows the Audit Log to show that the mapping works



See the recent catalog changes made by this data source and debug issues using the following table.

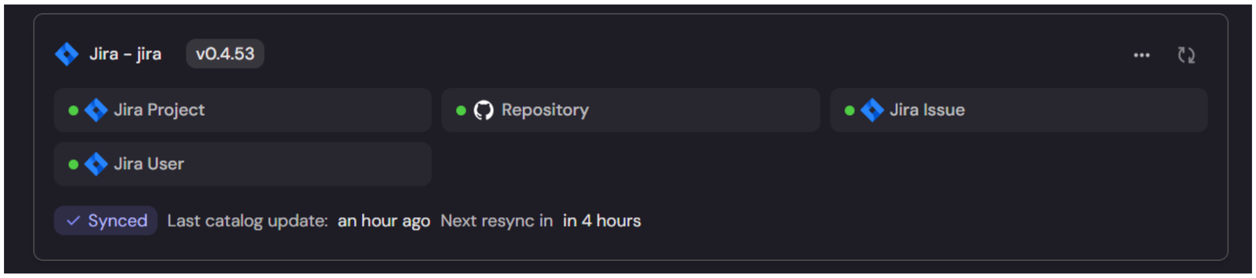
Operation	Blueprint	Entity	Error	Status	Date	Run ID
Update	Jira Issue	E2-1		Success	a few seconds ago	

```
  "relations": {
-   "jira_user_reporter": "712020:1a4ffcbf-a3bd-4819-bb3a-ed951bde3d7f",
-   "repository": []
  },
  "properties": {
    "issueType": "Bug",
    "components": [
```

```
  "relations": {
+   "repository": [
+     "public-apis"
+   ],
+   "jira_user_reporter": "712020:1a4ffcbf-a3bd-4819-bb3a-ed951bde3d7f"
  },
  "properties": {
    "issueType": "Bug",
    "components": [
```

99 results

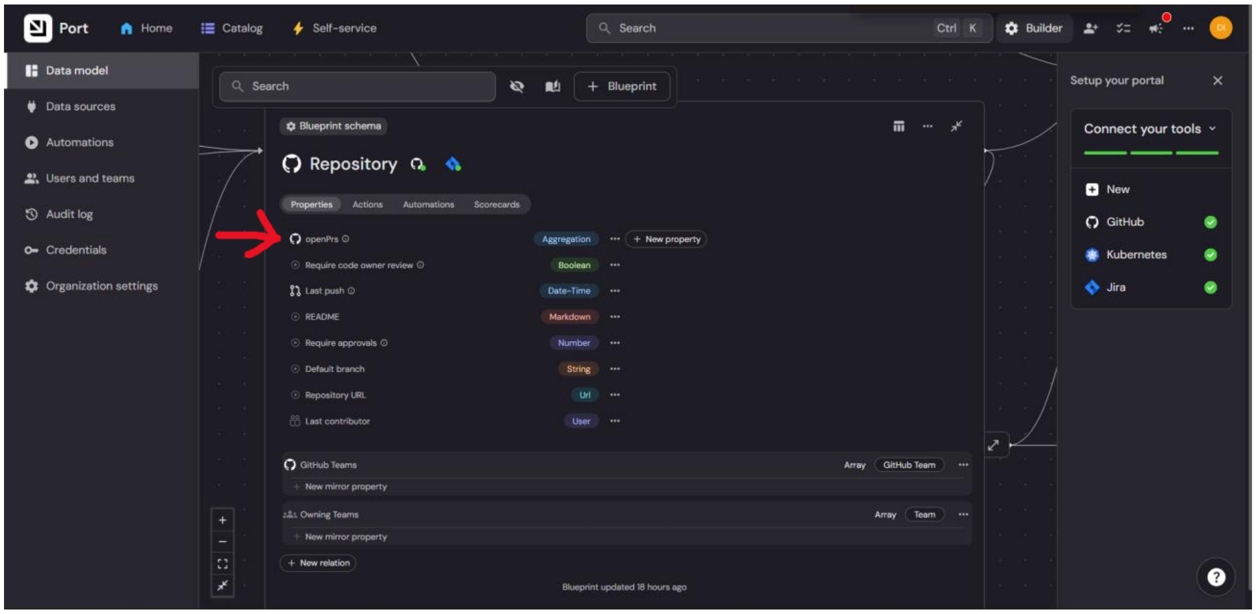
Synced Last catalog update: 3 minutes ago Next resync in in 5 hours
Data ingested successfully



Exercise #3 – PR Count Scorecard

1. **Added the aggregation property named openPrs**

Image below shows the added Aggregation property

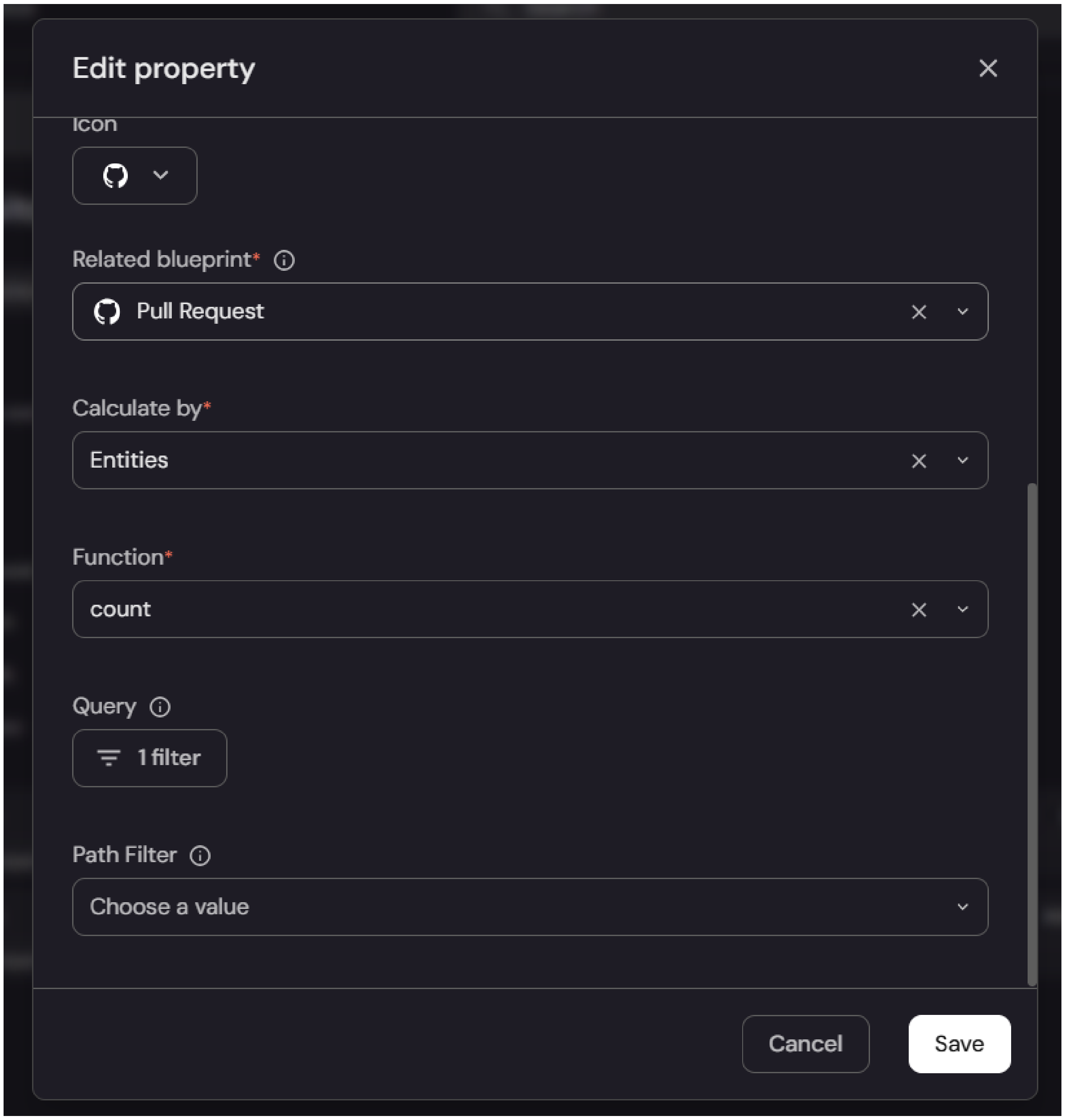


openPrs	Aggregation
Require code owner review	Boolean
Last push	Date-Time
README	Markdown
Require approvals	Number
Default branch	String
Repository URL	Url
Last contributor	User

GitHub Teams	Array	GitHub Team
Owning Teams	Array	Team

Blueprint updated 18 hours ago

Images below show the steps for the Aggregation property setup



Edit property

Icon

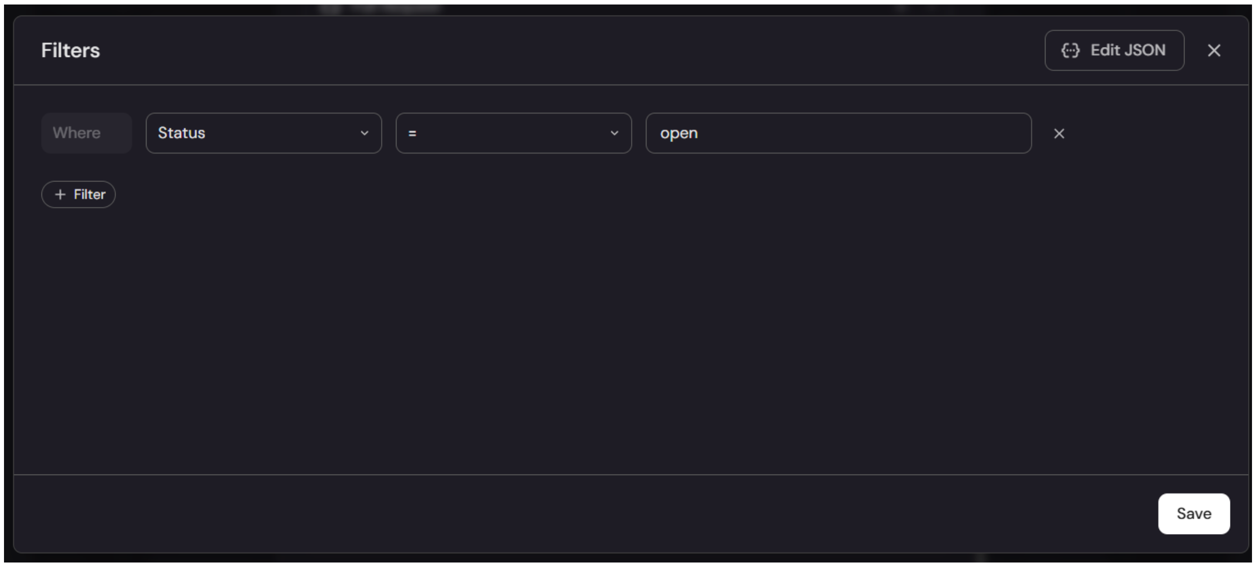
Related blueprint*: Pull Request

Calculate by*: Entities

Function*: count

Query: 1 filter

Path Filter: Choose a value


Filters

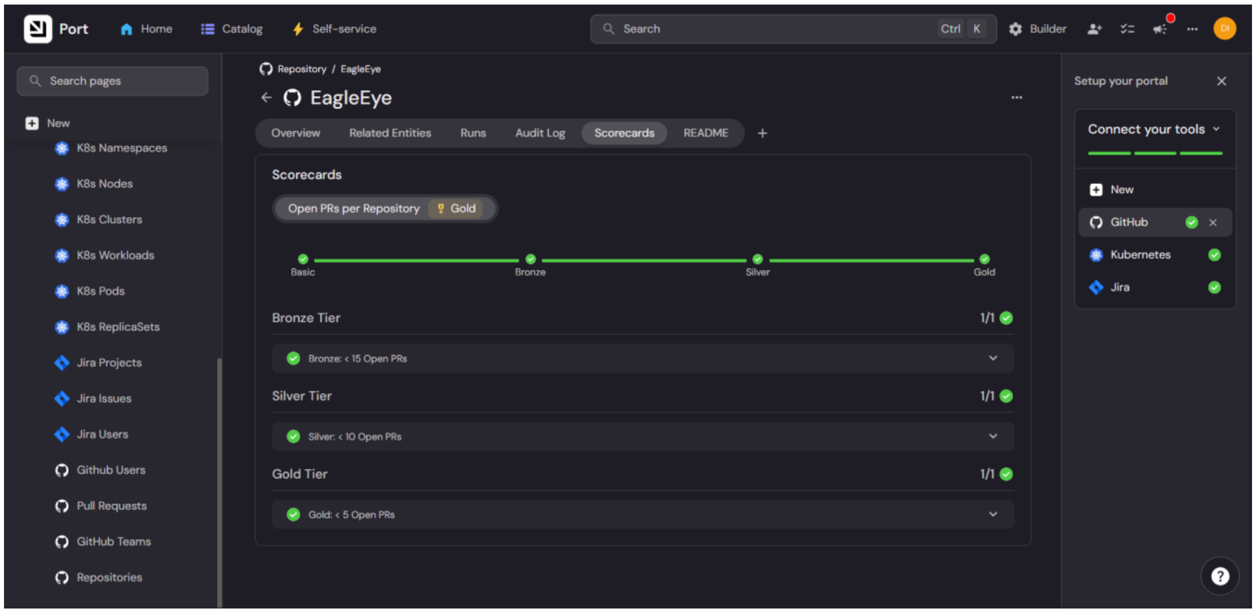
Where Status = open

2. **Added the scorecard logic to the Repository blueprint**

```json
{
  "identifier": "open_prs_scorecard",
  "title": "Open PRs per Repository",
  "levels": [
    {
      "color": "paleBlue",
      "title": "Basic"
    },
    {
      "color": "bronze",
      "title": "Bronze"
    },
    {
      "color": "silver",
      "title": "Silver"
    },
    {
      "color": "gold",
      "title": "Gold"
    }
  ],
  "rules": [
    {
      "identifier": "open_prs_gold",
      "title": "Gold: < 5 Open PRs",
      "level": "Gold",
      "query": {
        "combinator": "and",
        "conditions": [
          {
            "operator": "<",
            "property": "open_prs",
            "value": 5
          }
        ]
      }
    },
    {
      "identifier": "open_prs_silver",
      "title": "Silver: < 10 Open PRs",
      "level": "Silver",
      "query": {
        "combinator": "and",
        "conditions": [
          {
            "operator": "<",
            "property": "open_prs",
            "value": 10
          }
        ]
      }
    },
    {
      "identifier": "open_prs_bronze",
      "title": "Bronze: < 15 Open PRs",
      "level": "Bronze",
      "query": {
        "combinator": "and",
        "conditions": [
          {
            "operator": "<",
            "property": "open_prs",
            "value": 15
          }
        ]
      }
    }
  ]
```

```
}
```
` ` `

Image below shows the scorecard for the repository EagleEye with 1 open PR



Port | Home | Catalog | Self-service | Search Ctrl K | Builder

Search pages

New
- K8s Namespaces
- K8s Nodes
- K8s Clusters
- K8s Workloads
- K8s Pods
- K8s ReplicaSets
- Jira Projects
- Jira Issues
- Jira Users
- Github Users
- Pull Requests
- GitHub Teams
- Repositories

Repository / EagleEye

EagleEye

Overview | Related Entities | Runs | Audit Log | Scorecards | README

Scorecards

Open PRs per Repository — Gold

Basic — Bronze — Silver — Gold

Bronze Tier 1/1 ✓
- ✓ Bronze: < 15 Open PRs

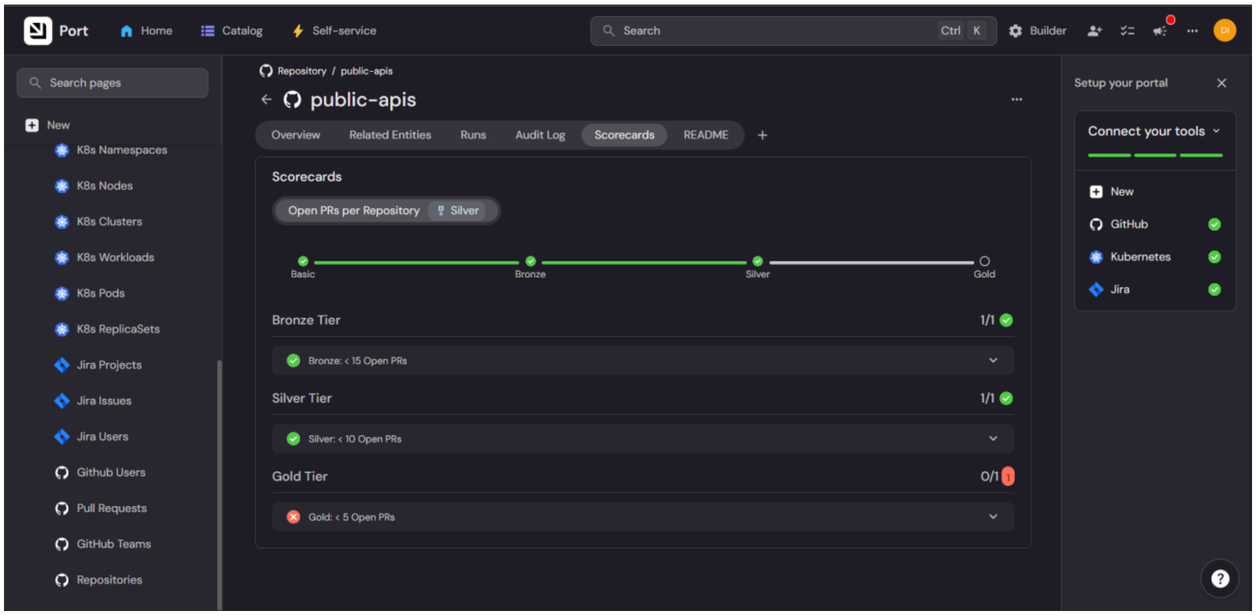
Silver Tier 1/1 ✓
- ✓ Silver: < 10 Open PRs

Gold Tier 1/1 ✓
- ✓ Gold: < 5 Open PRs

Setup your portal

Connect your tools
- New
- GitHub ✓
- Kubernetes ✓
- Jira ✓

Image below shows the scorecard for the repository public-apis with 1 open PR



Repository / public-apis

public-apis

Overview | Related Entities | Runs | Audit Log | Scorecards | README

Scorecards

Open PRs per Repository · Silver

Basic — Bronze — Silver — Gold

Bronze Tier 1/1 ✓

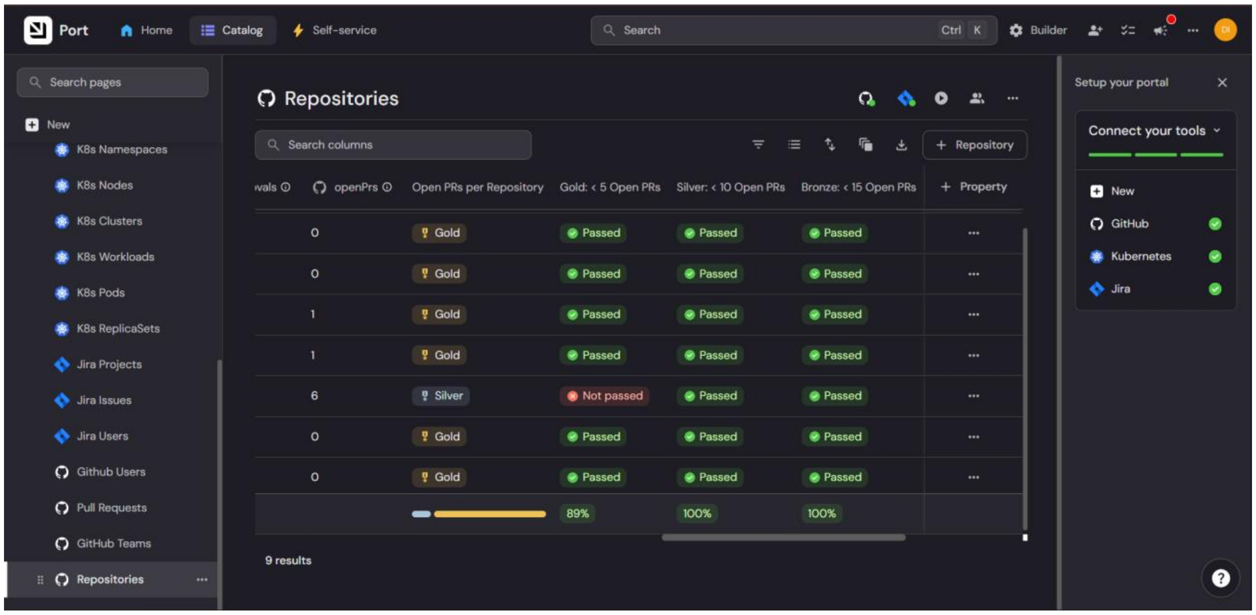
✓ Bronze: < 15 Open PRs

Silver Tier 1/1 ✓

✓ Silver: < 10 Open PRs

Gold Tier 0/1 !

✗ Gold: < 5 Open PRs



Repositories

openPrs	Open PRs per Repository	Gold: < 5 Open PRs	Silver: < 10 Open PRs	Bronze: < 15 Open PRs
0	Gold	Passed	Passed	Passed
0	Gold	Passed	Passed	Passed
1	Gold	Passed	Passed	Passed
1	Gold	Passed	Passed	Passed
6	Silver	Not passed	Passed	Passed
0	Gold	Passed	Passed	Passed
0	Gold	Passed	Passed	Passed
		89%	100%	100%

9 results

Exercise #4 – GitHub Workflow Troubleshooting.

Step 1. Verify Action Backend Configuration

Organization, Repository, and Workflow Name:
Double-check that the self-service action in Port is configured with the correct GitHub organization, repository, and workflow file name. Any typo or mismatch will most likely prevent the workflow from being triggered. This is a common cause and will result in the action being stuck in progress with no activity in GitHub.

Workflow Location:
Ensure the workflow file exists in the .github/workflows/ directory of the specified repository and branch.

Step 2. GitHub App Installation

App Installation:
Confirm that Port's GitHub App is installed in the correct GitHub organization and on the relevant repository.

Permissions:
The GitHub App must have the necessary permissions (Actions, Pull Requests, Contents, etc.) and be installed on the repository where the workflow resides.

Step 3. Workflow Trigger Configuration

workflow_dispatch Trigger:
The workflow must be configured to use the workflow_dispatch trigger. Without this, it cannot be triggered remotely by Port.

Branch Existence:
If specifying a branch via the ref key, customer has to ensure the workflow file exists in the default branch as well, due to GitHub's requirements.

Step 4. Secrets and Authentication

Port Credentials:
Ensure the GitHub repository contains the correct PORT_CLIENT_ID and PORT_CLIENT_SECRET as secrets, if required by your workflow.

GitHub Token:
If using a webhook invocation, verify that the GitHub token used has the necessary permissions and is correctly referenced in Port's secrets.

Step 5. Logs and Error Messages

Check Port UI and Logs:
Look for any error messages in the Port UI or logs. Sometimes, errors such as "Got 404 while addressing github api, repo or workflow not found" indicate misconfiguration of org, repo, or workflow names.

GitHub Workflow Runs:
Check the Actions tab in the GitHub repository to see if any workflow runs were triggered or if there are any failed runs.

Step 6. Additional Checks

Action Organization/Repository:
If using templates or guides, ensure you have replaced all placeholder values (e.g., <GITHUB_ORG>, <GITHUB_REPO>) with your actual organization and repository names.

Secrets in Port:
Make sure all required secrets are set in Port (e.g., GitHub tokens, Port credentials).