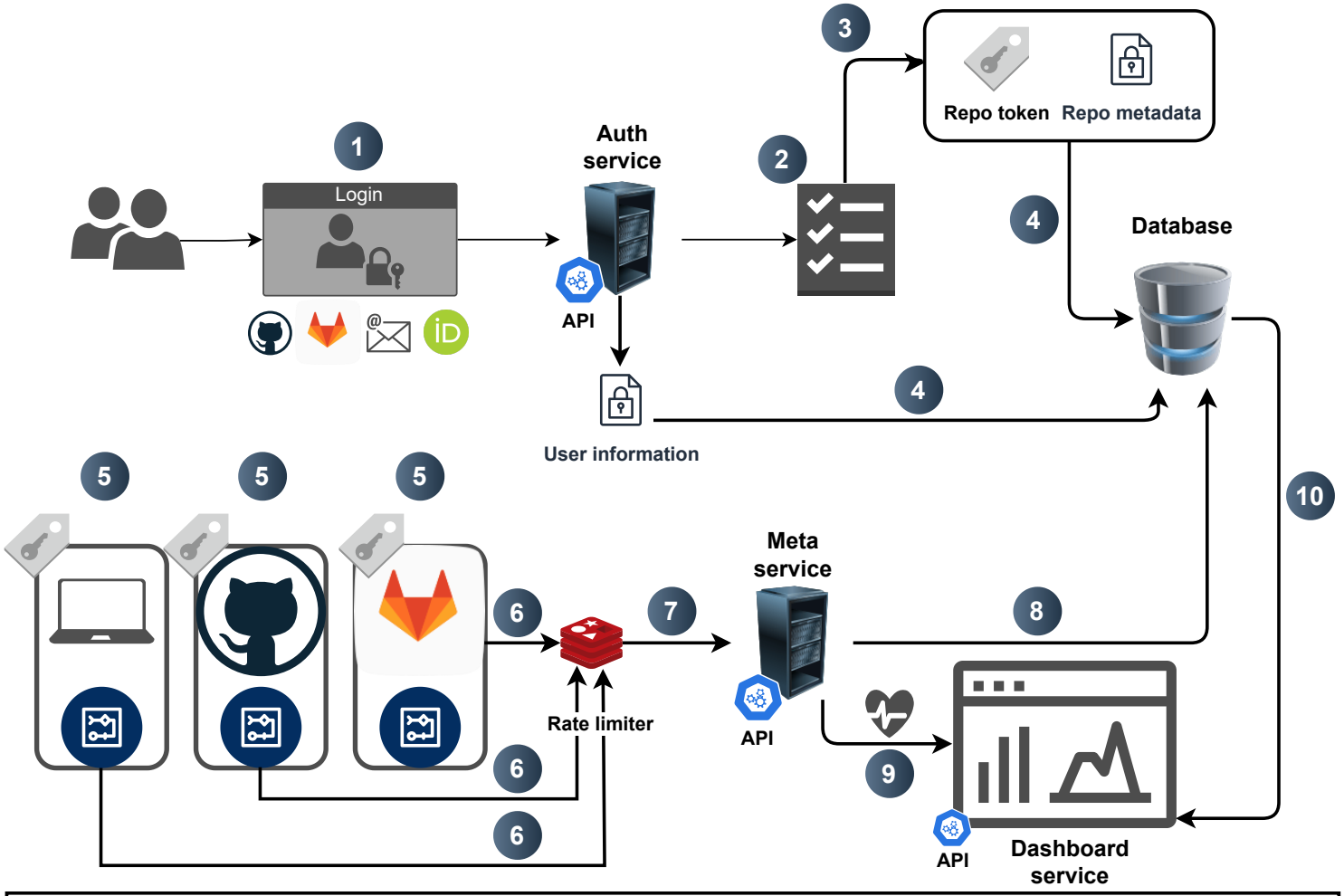
EVERSE System Design

Version 0.2.1

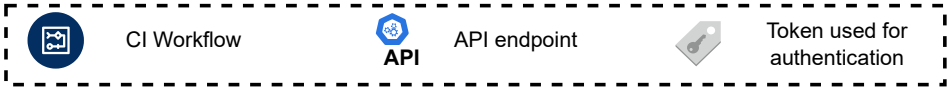


Storyline

1. User Authentication
Users log in to the authentication service using their GitHub, GitLab, email, ORCID, or other supported credentials.
2. Repository Access
After logging in, users enable access to the desired code repositories:

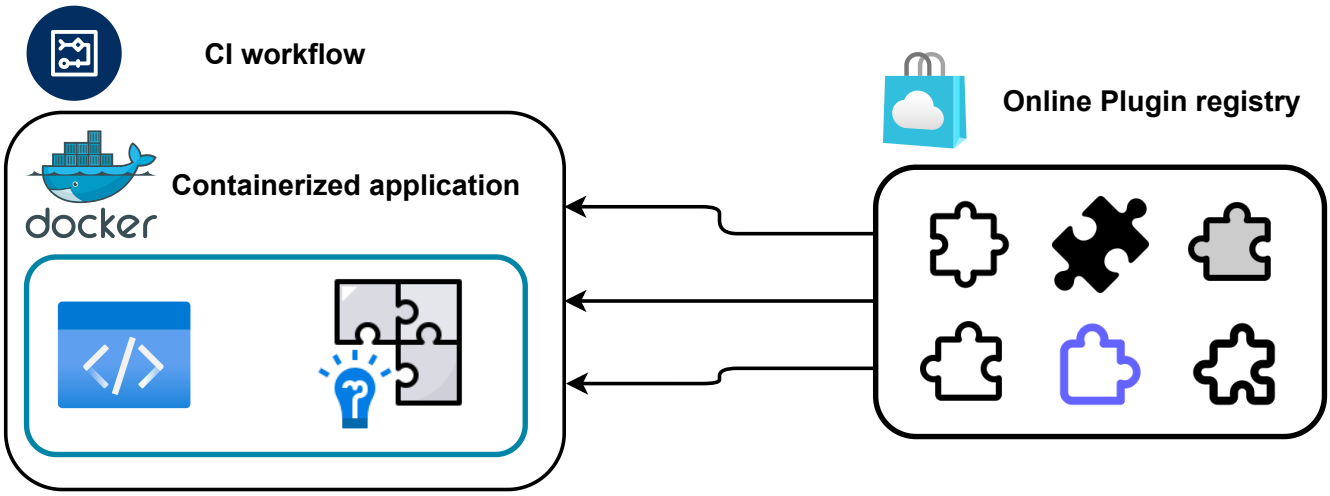
- If logged in with GitHub or GitLab, users can select repositories from a pre-populated list.
- If logged in with email or other credentials, users must manually add the repositories they wish to enable.

3. Token Generation
The authentication service generates a unique token for each repository, which will be used for subsequent authenticated workflows.
4. Data Storage
The authentication service stores the user information, tokens, and repository metadata in the database.
5. Compliance Assessment
A compliance assessment workflow is triggered, either running locally or as part of a continuous integration (CI) workflow. The workflow pulls required plugins (e.g., tests for various metrics) from an online plugin registry. It executes the assessment, generating an output that will be pushed to the meta service.
6. Rate Limiting
A rate limiter service monitors and limits the number of requests made by each workflow, ensuring fair use and preventing abuse.
7. Authenticated Output Submission
The workflow uses the previously generated token (from step 3) to authenticate with the meta service and submits the assessment output.
8. Data Storage in Meta Service
The meta service stores the submitted output in its database.
9. Notification of the dashboard
The meta service notifies the dashboard service of the new update through a websocket connection.
10. Dashboard Update
The dashboard service retrieves the updated data from the database and refreshes the dashboard with the latest information.



Metrics Workflow Design

Version 0.2.1



Main Application

- Containerized application which can be used in workflows (CI) or run on own servers or locally
- It pulls and uses selected plugins from the online registry

Plugins (tests)

- Two possible implementations
  - Python plugins
  - Docker containers
- One plugin per metric

Schema example for metrics

Version 0.2.1

```
[
          {
            "authors": {
              "name": "maintainer name",
              "type": [
                "individual",
                "organization"
              ],
              "orcid": "https://orcid.org/0000-0000-0000-0000",
              "email": "someone@somewhere.com"
            },
            "metric": {
              "name": "indicator name",
              "description": "a short description of metric",
              "tags": [
                "tag_1",
                "tag_2"
              ],
              "type": [
                "findable",
                "accessible",
                "interoperable",
                "reusable",
                "quality",
                "other"
              ],
              "release-date": "09-10-2024",
              "version":"0.0.1",
              "doi": "10.5281/zenodo.0000000",
              "software": {
                "cff": "https://github.com/citation-file-format/citation-file-format/blob/main/CITATION.cff"
              }
            }
          }
        ]
```

Full schema: https://gist.github.com/fdiblen/b8eddda1a6dfb40c83c46e2fc41fbf1c