

Forge External Integrations

Developer guide — for any external system connecting to Forge

What this integration does

An organization owner on **Forge** (forge.rhost.rw) has connected your system to specific repositories, using Forge's built-in self-service Integrations feature. This provisions a dedicated, least-privilege bot account that your system can use to create and update **issues** on those repositories through Forge's normal REST API — nothing more. This guide covers exactly what to build, regardless of what system you're integrating.

This is a **one-way** integration: your system creates and updates issues in Forge. Forge does not push changes back to your system automatically.

What you will receive from the Forge org owner

Before building anything, confirm you have all four of these — they should be handed to you out of band (not by email in plaintext):

API token	A single bearer token, scoped to <i>write:issue</i> + <i>read:repository</i> only. Shown to the org owner exactly once at creation — if it's lost, they must create a new integration and send you a new one.
Bot username	For reference only; you never need to send this yourself, only the token.
Repository list	One or more <i>{owner}/{repo}</i> pairs this token can act on. Any request to a repository not in this list is rejected.
Project ID(s)	A numeric ID per linked repository, used so created issues land in the right tracking board automatically (see below).

Authentication

Send the token in the **Authorization** header on every request, either form works:

```
Authorization: token <the-bot-token>

or

Authorization: Bearer <the-bot-token>
```

Any request outside the granted scope or repository list returns **403 Forbidden** by design — this is the access boundary working correctly, not an error to route around.

Creating an issue

POST https://forge.rhost.rw/api/v1/repos/{owner}/{repo}/issues

```
Authorization: token <the-bot-token>
Content-Type: application/json

{
  "title": "Short summary of the ticket",
  "body": "<!-- forge-integration:<slug>; external-id:<your-record-id> -->\nFull description here.",
  "projects": [42]
}
```

- **projects** — the linked repository's Project ID, provided by the org owner. Including it files the issue directly into that Project's board in the same call.
- **The response includes the created issue's number** (an integer, e.g. 137). **Store this against your own record id** — every future update uses this number, not any id internal to your system.
- A successful create returns **201 Created**. **422** usually means a validation problem (e.g. empty title). **403** means the token doesn't have access to that repository — double-check owner/repo against what was actually granted.

Updating or closing an issue

PATCH https://forge.rhost.rw/api/v1/repos/{owner}/{repo}/issues/{number}

```
Authorization: token <the-bot-token>
Content-Type: application/json

{
  "state": "closed"
}
```

{number} is the Forge issue number returned by the create call. state accepts "open" or "closed". The same endpoint also accepts title and body if you need to edit issue content later.

Correlation convention

Your own system is the source of truth for the id mapping. Store the Forge issue number returned at creation time, keyed by your own record id, and use that stored number for every future update call.

In addition, prefix every new issue's body with a hidden HTML comment, exactly as shown in the create example above:

```
<!-- forge-integration:<integration-slug>; external-id:<your-record-id> -->
```

This costs nothing and isn't required for the integration to function, but it means a human reading the issue directly in Forge — or a future reconciliation script on either side — can trace it back to your system's record without querying your database. The integration-slug is the lowercase, hyphenated form of the integration name the org owner chose (they will provide the exact value, shown on their Integrations page in Forge).

Security boundary — what the token cannot do

- Cannot push code, read repository contents or wikis, or see any repository outside the explicitly granted list.
- Cannot create, modify, or delete Projects, teams, webhooks, or any other organization-level resource.
- Cannot see or act on issues in any other repository, even within the same organization.
- If this token is ever compromised, its blast radius is limited to filing/editing issues in the granted repositories — nothing else in the organization is reachable with it. Store it the same way you'd store any other API secret: a secrets manager, never in source control or client-side code.

Questions or issues

For anything not covered here — different repositories, a lost token, or unexpected API behavior — go back to the Forge org owner who set up this integration; they can manage or recreate it from **Organization** → **Settings** → **Integrations** without any code changes on your side. Full reference documentation is also always available at <https://forge.rhost.rw/assets/help/index.html>.