



# HYCU for GitHub

---

**R-Cloud Module Guide**

## Table of Contents

|  |    |
|--|----|
| About the module.....                                | 3  |
| Prerequisites.....                                   | 3  |
| Limitations .....                                    | 3  |
| Considerations.....                                  | 5  |
| Protecting data.....                                 | 6  |
| Registering a GitHub App .....                       | 6  |
| Generating a user access token for a GitHub App..... | 7  |
| Backing up data .....                                | 8  |
| Restoring data .....                                 | 10 |

# Copyright notice

© 2024 HYCU. All rights reserved.

This document contains proprietary information, which is protected by copyright. No part of this document may be photocopied, reproduced, distributed, transmitted, stored in a retrieval system, modified or translated to another language in any form by any means, without the prior written consent of HYCU.

# About the module

With the R-Cloud (formerly HYCU Protégé) module for GitHub, you can back up your SaaS application data securely and efficiently.

## Prerequisites

Before you add the module to R-Cloud as a source, the following prerequisites must be fulfilled:

- The GitHub organization name. Provide the name when adding the module as a source in R-Cloud.
- The GitHub application associated with the GitHub organization, and the user token linked to the GitHub application.
- Your GitHub application must be granted the necessary set of permissions. For details, see [Registering a GitHub App](#).


## Limitations

When adding the module and while protecting the related SaaS application, the following limitations apply:

- If the size of any individual commit in the repository is larger than 2 GB, restoring such commits is not possible. Even if the restore completes successfully, the restored data may not be complete due to the GitHub limitations.
- If a commit fails to be restored, its child commits, comments, and related releases will not be restored as well.
- During the restore, the releases require names with a unique tag for the repository. If a tag is unavailable or already associated with another release, the tag and the release cannot be restored.
- The created or modified timestamps of issues, pull requests, comments, and releases will not be restored.
- While wikis are included in the backup, their automatic restore is not possible due to the limitations of the GitHub API.

- GitHub pull requests cannot be restored as pull requests because of the API limitations. Instead, pull requests will be restored as issues. The pull request comments and other metadata will be added to the newly created issues. The following additional limitations apply:
  - The closed status messages cannot be restored.
  - Source file changes, checks, and commit information of a pull request cannot be restored.
  - Reviewers and development metadata fields cannot be restored.
  - References to a new issue inside the pull request comments cannot be restored.
  - Reactions cannot be restored.
- The language percentage metadata of a repository, calculated by GitHub, cannot be restored due to the API limitations.
- Repository information such as forks, watches, stars, and license information cannot be backed up or restored.
- The following properties of issues, issue comments, and commit comments cannot be restored:
  - The closed status message
  - References to a new issue inside the issue comments
  - Development metadata fields
  - The hidden property of issue comments
  - Reactions
- The attachments associated with issues, pull request, source comments, or body cannot be backed up or restored due to the API limitations. Instead, the module will be referencing those links from the old repository only.
- If the assigned users are contributors to the repository and not contributors to the organization, due to the API limitations, during a complete restore, the assignees cannot be directly restored. Instead, they will be added to the comments.
- The following properties of a project v2 cannot be backed up or restored:
  - Repository relationship during the repository level restore
  - Insights (Charts)
  - Iteration (Cannot be restored if the project is deleted)
  - Make template toggle
  - Project draft fields
  - Status update
  - Workflows

- Special characters, such as the double quote or escape, are not supported by the GitHub Projects API. Such characters will be replaced with the underscore.
- The Status field of a project v2 will be restored in a separate column called Restored status with project ID.
- Depending on whether the project you are restoring exists or not, the following will happen during the repository restore:
  - If the project exists: The issue and pull request mapping will be linked to the same project if the Restore project relations toggle is enabled.
  - If the project does not exist: The project will be newly created with its metadata and relations associated with the repository.
- Depending on whether the project you are restoring exists or not, the following will happen during the issue level/pull request level restore:
  - If the project exists: The issue and pull request mapping will be linked to the same project if the Restore project relations toggle is enabled.
  - If the project does not exist: The project relations will not be restored.
- Due to hierarchy limitations, the module does not support the Granular restore option for projects v2. The restore of a project v2 can only be performed at the repository level.
- The module does not support the GitHub enterprise server with a custom domain.

 **Note** GitHub imposes rate limits. Triggering parallel backups or restores of multiple repositories might extend the time required to complete the jobs.

## Considerations

Before you add the module as a source, consider the following:

- The repository restore might fail if the GitHub secret scanning is enabled for the organization and if the repository has tokens or secrets.
- If a release is tagged with a branch that does not exist or has been renamed, the restore of such a release might fail due to GitHub not updating the release metadata with the branch name changes.

# Protecting data

R-Cloud starts protecting your GitHub data after you complete the following tasks:

1. Register the GitHub app. For instructions, see [Registering a GitHub App](#).
2. Generate the user access token for the GitHub app. For instructions, see [Generating a user access token for a GitHub App](#).
3. Add the module as a source to R-Cloud. For instructions, see *HYCU R-Cloud Help*.

**Note** When adding the module as a source, provide the GitHub organization name and the user token generated in the previous two steps.

4. Assign a policy to the related SaaS application. For instructions, see *HYCU R-Cloud Help*.

**Important** The backup and restore options for the following files are not enabled by default: LFS files and release asset files. If you plan to backup and restore these files, enable the options from the backup and restore configuration.

## Registering a GitHub App

1. Create a new GitHub App. For instructions, see the GitHub documentation on [Registering a GitHub App](#).
2. There are three additional steps to the standard GitHub app registering procedure:
  - a. Fill in all the required inputs to create a GitHub App and enter an optional value for the Callback URL.
  - b. Disable the Expire user authorization tokens option. This ensures uninterrupted backup and restore using the user authorization tokens.
  - c. Under **Permissions**, select the following:
    - Repository permissions:
      - Administration – Read and write
      - Contents – Read and write
      - Issues – Read and write
      - Metadata – Read only

- Pull requests – Read and write
  - Projects – Read and write
  - Organization permissions:
    - Members – Read only
    - Projects – Read and write
3. After creating the GitHub App, install the app for the organization. For instructions, see the GitHub documentation on [Installing your own GitHub App](#).

**ⓘ Important** During the installation, select all repositories or the repositories that you want to list in R-Cloud to be backed up.

## Generating a user access token for a GitHub App

1. After registering and installing the GitHub App for the organization, navigate to the General section from the sidebar.
2. Generate a Client Secret by clicking the Generate a new client secret button. Copy the generated secret to a secure location, as it is viewable only once.
3. Copy the Client Id and update the {client\_id} in the URL below. Open the URL in any browser. You will be redirected to an authorization screen. Authorize the request.  
`https://github.com/login/oauth/authorize?client_id={client_id}&state=abcdefg`
4. After the authorization, the control will be redirected back to the callback URL configured during the GitHub App creation. Copy the code field from the URL. This code is needed to generate the user authorization token.

**📄 Note** The code field is valid for one-time usage. If you encounter any errors during the token generation process, regenerate the code.

5. To generate the user authorization token from the code, use any script or tools, and call the below defined POST API. After triggering this API call, GitHub will provide a JSON response that includes the access token. Use this token, to configure the source in R-Cloud.

URL: `https://github.com/login/oauth/access_token`



Method: POST

Request body (JSON):

```
{  
  "grant_type": "code",  
  "client_id": "{client_id}",  
  "code": "{code}",  
  "client_secret": "{client_secret}"  
}
```

Replace the placeholders {client\_id}, {code}, and {client\_secret} before making the API call.

## Backing up data

After adding the module, all the repositories in the specified organization, their issues, comments, releases, and pull requests will be automatically detected.

The supported objects are:

- Repository Source
  - refs
  - branches
  - commits
  - tags
  - objects
  - logs
  - LFS Files
- Commit Comments
  - comment text
  - creation date
  - creator
- Labels
  - name
  - description
  - color
  - associated issues

- Milestones
  - status
  - name
  - description
  - due date
  - associated issues
- Issues
  - title
  - description
  - creation date
  - creator
  - status
  - comments (without reactions)
  - assignee
  - assigned labels
  - assigned milestones
  - project relations as comments
- Pull Requests
  - title
  - description
  - creation date
  - creator
  - status
  - review comments (without reactions)
  - assignee
  - assigned labels
  - assigned milestones
  - project relations as comments
- Releases
  - tag name
  - name
  - target commit
  - body
  - creation date
  - creator
  - metadata (pre-release, draft, latest)
  - release assets

- Projects v2
  - readme
  - short description
  - private/public metadata
  - columns and rows (including custom columns)
- Wikis

For details on how to configure backups for your SaaS application, see *HYCU R-Cloud Help*.

## Restoring data

R-Cloud allows you to restore the protected GitHub data at the following levels:

- Repository
- Repository Issue
- Repository Pull Request
- Repository Release
- Repository Release Asset

For details on how to configure the restore for your SaaS application, see *HYCU R-Cloud Help*.

# Provide feedback

For any suggestions and comments regarding this product or its documentation, send us an e-mail to:

[info@hycu.com](mailto:info@hycu.com)

We will be glad to hear from you!

