Migration of Strapi from Self-Hosted to Strapi Cloud

Overview

This document outlines the steps followed and key considerations during the migration of self-hosted Strapi instance (v4.13.7) to Strapi Cloud. The migration was executed to leverage Strapi Cloud's built-in scalability, security, and ease of management, while ensuring all existing content, plugins, and customizations were preserved.

Pre-Migration Preparation

1. Ensure that the project is hosted in (GitHub/GitLab)

- Make sure that the latest updates are hosted in a repo GitHub

2. Backup Current Instance

- Created a complete backup of the existing Strapi instance, including:
    - Database (MySQL/PostgreSQL)
    - Files (Media library, configurations, etc.)
- Ensured that no content changes occurred during the migration window to avoid discrepancies.

3. Ensure compatibility with Strapi Cloud

- Ensure that Strapi version is 4.8.2 or higher
    - update Strapi to latest stable version (recommended)
- Ensure that Node engine is >=14.19.1 and <=20.x.x
    - Make sure to set in package.json

```
"engines": {
  "node": ">=14.19.1 <=20.x.x",
  "npm": ">=6.0.0"
 },
```

4. Make sure to provide the required configurations for migration

- Set the `TRANSFER_TOKEN_SALT` in .env file
    - E.g TRANSFER_TOKEN_SALT=HRQNLzb+9YFZlPKPHaDFhA==

- Configure *./config/admin.ts* and add this configuration

```
transfer: {
  token: {
    salt: env("TRANSFER_TOKEN_SALT"),
  },
},
```

- Add the origin of the Strapi cloud instance to the **strapi::cors** middleware in *./config/middlewares.ts*
  - Will obtain this origin later after deploying on Strapi cloud

- Make sure to apply config cync.
  - Go To settings/ config sync => import

Migration Process

1. Deploying to Strapi Cloud

- Set up the Strapi Cloud environment:
  - Created a new project on Strapi Cloud.
  - Configured environment variables (e.g., API keys, transfer token salt).

2. Database Migration

- Exported the existing database (MySQL/PostgreSQL) using appropriate tools
- Imported the data into self hosted Strapi instance
  - Ensured all relations and collections/tables were intact post-import.

3. Migrating Content, Assets, Links and configurations

- After deploying the code to Strapi cloud create a transfer token from **Settings / Transfer tokens => Add new transfer token** and give it a name and duration and push token type .
- After getting the transfer token run this command in CLI

```
 yarn strapi transfer --to <STRAPI_CLOUD_URL> --to-token <TRANSFER_TOKEN> --force --exclude
files
```

where :
STRAPI_CLOUD_URL : the URL of deployed Strapi cloud instance (e.g https://superb-talent-1152a5390d.strapiapp.com/admin)

TRANSFER_TOEKN : the transfer token that you get from the above step (e.g 3b4c01723abf84d8b2cfda2d5388bed1d760a272537dcd22c3d981e931c247009f61b52e31c0ef3b6a5bec56a511737adf0503eb46ef53a2f86eda2314070d7ad75552d3677198275fa2810c34e636a431a56819b89e55113114fa12b534fbd9b0ca86bd2c4613170c4ddcc5f5356464133dd1ce28496e277f95d29f7dea6585)

For example:

```
yarn strapi transfer --to https://superb-talent-1152a5390d.strapiapp.com/admin --to-token
3b4c01723abf84d8b2cfda2d5388bed1d760a272537dcd22c3d981e931c247009f61b52e31c0ef3b6a5bec56a511737
adf0503eb46ef53a2f86eda2314070d7ad75552d3677198275fa2810c34e636a431a56819b89e55113114fa12b534fb
d9b0ca86bd2c4613170c4ddcc5f5356464133dd1ce28496e277f95d29f7dea6585 --force --exclude files
```

Post-Migration Tasks

- **Change Domain name**
    - You can change domain name from **settings / domains / connect new domain In Strapi cloud settings**
    - But first you need to buy a domain name to set a DNS record with type **CNAME** to point to the target url of Strapi cloud instance

Conclusion

The migration to Strapi Cloud was successful with no data loss or major issues encountered. All content, customizations, and plugins are functioning as expected. Strapi Cloud offers improved performance, scalability, and ease of management, enhancing the overall efficiency of our content management system.