**Table of Content**

* [Huly Platform](https://github.com/hcengineering/platform/tree/develop#huly-platform)
  + [About](https://github.com/hcengineering/platform/tree/develop#about)
  + [Self-Hosting](https://github.com/hcengineering/platform/tree/develop#self-hosting)
  + [Activity](https://github.com/hcengineering/platform/tree/develop#activity)
  + [Table of Content](https://github.com/hcengineering/platform/tree/develop#table-of-content)
  + [Pre-requisites](https://github.com/hcengineering/platform/tree/develop#pre-requisites)
  + [Verification](https://github.com/hcengineering/platform/tree/develop#verification)
  + [Installation](https://github.com/hcengineering/platform/tree/develop#installation)
  + [Build and run](https://github.com/hcengineering/platform/tree/develop#build-and-run)
  + [Run in development mode](https://github.com/hcengineering/platform/tree/develop#run-in-development-mode)
  + [Update project structure and database](https://github.com/hcengineering/platform/tree/develop#update-project-structure-and-database)
  + [Troubleshooting](https://github.com/hcengineering/platform/tree/develop#troubleshooting)
  + [Build & Watch](https://github.com/hcengineering/platform/tree/develop#build--watch)
  + [Tests](https://github.com/hcengineering/platform/tree/develop#tests)
    - [Unit tests](https://github.com/hcengineering/platform/tree/develop#unit-tests)
    - [UI tests](https://github.com/hcengineering/platform/tree/develop#ui-tests)
  + [Package publishing](https://github.com/hcengineering/platform/tree/develop#package-publishing)
  + [Additional testing](https://github.com/hcengineering/platform/tree/develop#additional-testing)

**Pre-requisites**

* Before proceeding, ensure that your system meets the following requirements:
  + [Node.js](https://nodejs.org/en/download/) (v20.11.0 is required)
  + [Docker](https://docs.docker.com/get-docker/)
  + [Docker Compose](https://docs.docker.com/compose/install/)

**Verification**

To verify the installation, perform the following checks in your terminal:

* Ensure that the docker commands are available:
* docker --version

docker compose version

**Fast start**

sh ./scripts/fast-start.sh

**Installation**

You need Microsoft's [rush](https://rushjs.io/) to install application.

1. Install Rush globally using the command:

npm install -g @microsoft/rush

1. Navigate to the repository root and run the following commands:
2. rush install

rush build

Alternatively, you can just execute:

sh ./scripts/presetup-rush.sh

**Build and run**

Development environment setup requires Docker to be installed on system.

Support is available for both amd64 and arm64 containers on Linux and macOS.

cd ./dev/

rush build # Will build all the required packages.

# rush rebuild # could be used to omit build cache.

rush bundle # Will prepare bundles.

rush package # Will build all webpack packages.

rush validate # Will validate all sources with typescript and generate d.ts files required for ts-node execution.

rush svelte-check # Optional. svelte files validation using svelte-check.

rush docker:build # Will build Docker containers for all applications in the local Docker environment.

rush docker:up # Will set up all the containers

Be aware rush docker:build will automatically execute all required phases like build, bundle, package.

Alternatively, you can just execute:

sh ./scripts/build.sh

By default, Docker volumes named dev\_db, dev\_elastic, and dev\_files will be created for the MongoDB, Elasticsearch, and MinIO instances.

Before you can begin, you need to create a workspace and an account and associate it with the workspace.

cd ./tool # dev/tool in the repository root

rushx run-local create-workspace ws1 -w DevWorkspace # Create workspace

rushx run-local create-account user1 -p 1234 -f John -l Appleseed # Create account

rushx run-local configure ws1 --list --enable '\*' # Enable all modules, even if they are not yet intended to be used by a wide audience.

rushx run-local assign-workspace user1 ws1 # Assign workspace to user.

rushx run-local confirm-email user1 # To allow the creation of additional test workspaces.

Alternatively, you can just execute:

sh ./scripts/create-workspace.sh

Add the following line to your /etc/hosts file

127.0.0.1 host.docker.internal

Accessing the URL [http://host.docker.internal:8087](http://host.docker.internal:8087/) will lead you to the app in development mode.

Limitations:

* Local installation does not support sending emails, thus disabling functionalities such as password recovery and email notifications.

**Run in development mode**

Development mode allows for live reloading and a smoother development process.

cd dev/prod

rush validate

rushx dev-server

Then go to [http://localhost:8080](http://localhost:8080/)

Click on "Login with password" link on the bottom of the right panel and use the following login credentials:

Email: user1

Password: 1234

Workspace: ws1

**Update project structure and database**

If the project's structure is updated, it may be necessary to relink and rebuild the projects.

rush update

rush build

It may also be necessary to upgrade the running database.

cd ./dev/tool

rushx upgrade -f

**Troubleshooting**

If a build fails, but the code is correct, try to delete the [build cache](https://rushjs.io/pages/maintainer/build_cache/) and retry.

# from the project root

rm -rf common/temp/build-cache

**Build & Watch**

For development purpose rush build:watch action could be used.

It includes build and validate phases in watch mode.

**Tests**

**Unit tests**

rush test # To execute all tests

rushx test # For individual test execution inside a package directory

**UI tests**

cd ./tests

rush build

rush bundle

rush docker:build

## creates test Docker containers and sets up test database

./prepare.sh

## runs UI tests

rushx uitest

To execute tests in the development environment, please follow these steps:

cd ./tests

./create-local.sh ## use ./restore-local.sh if you only want to restore the workspace to a predefined initial state for sanity.

cd ./sanity

rushx dev-uitest # To execute all tests against the development environment.

rushx dev-debug -g 'pattern' # To execute tests in debug mode with only the matching test pattern.

**Package publishing**

node ./common/scripts/bump.js -p projectName