

## BETA TEST PLAN – Haddock

Haddock is a modern and innovative Platform as a Service (PaaS) solution, designed to simplify the deployment of applications by relying on containers.

### 1. Core Functionalities

#### 1.1 Onboarding

Feature Name	Description	Priority (High/Medium/Low)
Installer	Linux installer used to setup Haddock on the host machine.	High
Github Application Setup	Onboarding step to configure Github application, used for oAuth authentication.	High
Admin signup	Onboarding step to create the first admin user.	High
Main DNS configuration	Onboarding step to the main domain name, used to access the Haddock dashboard and deployed applications.	High
Other DNS configuration	Onboarding step to configure additional domain names for applications.	Medium

#### 1.2 Authorizations

Feature Name	Description	Priority (High/Medium/Low)
oAuth Authorization	A way for users to use their Github account to authenticate with Haddock and use their repositories.	High

Feature Name	Description	Priority (High/Medium/Low)
Personal Access Tokens	A way for users to authenticate with Haddock using a personal access token and use their repositories.	Medium
Deploy Key	A way for users to authenticate with Haddock using a deploy key and use their repositories.	Medium
Revocation	Ability to revoke access to their repositories from Haddock using a specific method.	Low

### 1.3 Application Deployment

Feature Name	Description	Priority (High/Medium/Low)
Repository selection	Ability to list with auto-complete the repositories of the user through the selected authorization method.	High
Branch selection	Ability to select the branch to deploy from the selected repository.	High
Virtual Machine configuration	Ability to select the virtual machine configuration for the application. (RAM, Disk space, CPUs)	High

### 1.4 Application Management

Feature Name	Description	Priority (High/Medium/Low)
Topological view	A graphical representation of the deployed applications and their dependencies within a graph.	High
Service details	Detailed information about the deployed services, including status, image used. . .	High
Project settings	Section to manage the settings of the deployed application.	High
Network binding	Interactive reverse proxy configuration to bind the application services to a specific domain name.	High
Application execution	Ability to start, stop, and restart the deployed application.	High
Services execution	Ability to start, stop, and restart the services within the application.	Medium

### 1.5 Application Monitoring

Feature Name	Description	Priority (High/Medium/Low)
Application status	Real-time execution status of the application	High
Services status	Real-time execution status of the services within the application	Medium
Application logs	Overview of the logs of a deployed application.	Medium
Application metrics	Ability to view the metrics of a deployed application.	Medium

Feature Name	Description	Priority (High/Medium/Low)
Environment variables	Registry of the environment variables used by the deployed application.	Medium

## 1.6 Infrastructure and platform management

Feature Name	Description	Priority (High/Medium/Low)
User management	Ability to manage users and their roles within Haddock.	High
Dashboard overview	Overview of deployed applications on the host machine.	High
Github application management	Ability to manage the Github application used for oAuth authentication.	Medium
Download user data	Ability to download the user data from Haddock.	Low

## 2. Beta Testing Scenarios

### 2.1 User Roles

Role Name	Description
System Administrator	Responsible for setting up and configuring Haddock on the host machine.
Admin	Responsible for managing users, applications, and configurations within Haddock.
Developer	Casual user that wants to manage applications using Haddock.

### Scenario 2: Application installation

- **Role Involved:** System administrator
- **Objective:** To verify the successful installation of the Haddock application on a Debian 12 machine using Vagrant as the provider.

- **Preconditions:**
  - A Debian 12 machine is set up and running.
  - The system administrator has access to the terminal with appropriate permissions.
- **Test Steps:**
  1. Open the terminal on the Debian 12 machine.
  2. Run the installation command provided in the Haddock documentation.
  3. When prompted, select yes for installing “vagrant” as the provider for the installation.
  4. Monitor the installation process and wait until it completes successfully.
- **Expected Outcome:**
  - The installer should exit automatically upon completion.
  - The installer should display all installation steps with checkmarks indicating successful completion of each step.
  - The installer should output a URL for accessing the installed Haddock application.

### Scenario 3: Onboarding: Setup github application

- **Role Involved:** System administrator
- **Objective:** Verify the ability to configure and set up a GitHub application during the onboarding process.
- **Preconditions:**
  - The Haddock installation process must have been successfully completed.
  - The system administrator must have access to the onboarding panel link generated after installation success.
  - The administrator must have a GitHub account with sufficient permissions to create and manage applications.
- **Test Steps:**
  1. Access the onboarding panel via the link provided after the Haddock installation was successfully completed.
  2. On the onboarding panel, click on the “App Configuration” button.
  3. Navigate to the GitHub website and create a new GitHub application by filling out the required fields (e.g., application name, homepage url, callback URL provided by Haddock).
  4. Obtain the required credentials (e.g., client ID and client secret) from the GitHub application configuration.

5. Return to the onboarding panel, and input the obtained GitHub application credentials into the corresponding fields on Haddock.
6. Save and confirm the configuration.
7. Log out of Haddock and attempt to access the system again to check for redirection behavior. ??

- **Expected Outcome:**

- After saving the GitHub application configuration, the setup process completes successfully without errors.
- The “Login with Github” button is prominently visible.

#### Scenario 4: Onboarding: Register via GitHub oAuth

- **Role Involved:** System administrator

- **Objective:** Verify that the system administrator can register and log in to the application using GitHub oAuth.

- **Preconditions:**

- The system administrator must have a valid GitHub account.
- A GitHub application must be set up as described in the “Onboarding: Setup GitHub application” section.
- The application must have the GitHub oAuth integration enabled and accessible.

- **Test Steps:**

1. Click the “Login with GitHub” button.
2. In the GitHub oAuth modal, log in to GitHub (if not already logged in).
3. Grant access to the requested permissions for the GitHub application by clicking the “Authorize” button.
4. Allow time for the system to complete the authentication process.

- **Expected Outcome:**

- The system administrator should be redirected to the “Domain names” page within the application.
- If permissions are denied in the GitHub modal, the system should handle the error gracefully and display an appropriate message.

#### Scenario 5: Onboarding: Register via email

- **Role Involved:** System administrator

- **Objective:** Verify that a system administrator can successfully register via email and be redirected to the “Domain names” page.

- **Preconditions:**

- The application is up and running.
- The system administrator has access to the signup page.

- **Test Steps:**
  1. Open the onboarding signup page.
  2. Enter a valid user name in the “Name” input field.
  3. Enter a valid email address in the “E-mail” input field.
  4. Enter a valid password in the “Password” input field.
  5. Click on the “Signup” button.
- **Expected Outcome:**
  - The system administrator should be successfully registered.
  - The system should redirect the administrator to the “Domain names” page within the application.

### Scenario 6: Onboarding: Setup primary domain

- **Role Involved:** System Administrator
- **Objective:** To test the functionality of setting up a primary domain during the onboarding process.
- **Preconditions:**
  - The system administrator should have access to the onboarding platform.
  - The system administrator must own and have control over a valid domain name.
  - The system administrator should have access to their domain provider’s interface for DNS configuration.
- **Test Steps:**
  1. Open the onboarding onboarding Domain names page.
  2. Select the “Setup main domain” section.
  3. In the “Domain name” input, write an owned valid domain name and click on the “Confirm” button.
  4. Note down the three generated DNS Bind inputs (primary, wildcard, challenge).
  5. Use these DNS Bind inputs to set up the domain on the provider’s interface.
  6. Click on the “Refresh” button to check the progression of the setup. The checkmark should update according to the configuration on the provider’s interface.
  7. Repeat step 6 until all steps are marked by a checkmark.
  8. Once all steps are completed, the “Next” button should be enabled.
  9. Click on the “Save” button to save the primary domain name.
- **Expected Outcome:**
  - After confirming the domain at the first step, 3 bind values (primary, wildcard, challenge) must be generated.

- The checkmark should update on refresh button click, according to the configuration on the provider’s interface.
- The “Next” button must be disabled until all steps are completed.
- The “Save” button must allow you to setup a second domain name after the first one is saved.
- No error messages or unexpected behavior should occur during the test.

### Scenario 7: Onboarding: Setup secondary domain

- **Role Involved:** System Administrator
- **Objective:** To verify the functionality of setting up a secondary domain in the onboarding process.
- **Preconditions:**
  - The system administrator should have access to the onboarding platform.
  - The system administrator must have already set up a primary domain.
  - The system administrator must own and have control over a valid domain name (different than the domain used as Primary Domain).
  - The system administrator should have access to their domain provider’s interface for DNS configuration.
- **Test Steps:**
  1. Still on the onboarding onboarding “Domain names” page.
  2. Locate and click the “Setup secondary domain” section.
  3. Enter a new, unique domain name in the “Domain name” field that differs from the primary domain and click on the “Confirm” button.
  4. Note down the three generated DNS Bind inputs (primary, wildcard, challenge).
  5. Use these DNS Bind inputs to set up the domain on the provider’s interface.
  6. Click on the “Refresh” button to check the progression of the setup. The checkmark should update according to the configuration on the provider’s interface.
  7. Repeat step 6 until all steps are marked by a checkmark (note that this time, Primary Bind is optional).
  8. Click on the “Save” button to save the primary domain name.
- **Expected Outcome:**
  - After confirming the domain at the first step, 3 bind values (primary, wildcard, challenge) must be generated.
  - The checkmark should update on refresh button click, according to the configuration on the provider’s interface.

- No error messages or unexpected behavior should occur during the test.

#### Scenario 8: Inviting a new user

- **Role Involved:** Admin
- **Objective:** To test the functionality of inviting a new user to the platform.
- **Preconditions:**
  - A valid email address for the invited user
  - Be logged in with an active admin account
- **Test Steps:**
  1. Open the settings drawer by clicking on the purple stack icon at the top-right corner of the dashboard.
  2. Unfold the “Users” sub-menu and click on it to view the list of existing users.
  3. Press the “Invite User” button to invite a new user.
  4. Inside the “Invite User” modal, enter the user’s email.
  5. Press the “Send invitation button” to invite the user.
- **Expected Outcome:**
  - A toast message appears at the bottom of the screen informing you that the user has been invited successfully.
  - The invited user appears in the list of users with a “invited” Role.

#### Scenario 9: Setting the github application configuration from the settings drawer

- **Role Involved:** Admin
- **Objective:** Test the functionality of setting the GitHub application configuration from the settings drawer.
- **Preconditions:**
  - The system administrator did not set the GitHub application configuration yet.
  - The administrator is logged in to the application.
  - A valid GitHub client ID and secret exist.
- **Test Steps:**
  1. Open the settings drawer by clicking on the purple stack icon at the top-right corner of the screen.
  2. Unfold the “GitHub Application” sub-menu.
  3. Fill in the “Github Client ID” and “Github Client Secret” fields with valid information.
  4. Press the “Update” button to save your changes.
- **Expected Outcome:**
  - A toast message appears at the bottom of the screen to inform you that the configuration was updated successfully, indicating a successful setting of the GitHub application configuration.

#### Scenario 10: Editing the github application configuration

- **Role Involved:** Administrator
- **Objective:** Test the functionality of editing the GitHub application configuration.
- **Preconditions:**
  - The administrator is logged in to the application.
  - A valid GitHub client ID and secret exist.
- **Test Steps:**
  1. Open the settings drawer by clicking on the purple stack icon at the top-right corner of the screen.
  2. Unfold the “GitHub Application” sub-menu.
  3. Fill in the new “Github Client ID” and “Github Client Secret” fields with valid information.
  4. Press the “Update” button to save your changes.
- **Expected Outcome:**
  - A toast message appears at the bottom of the screen to inform you that the configuration was updated successfully, indicating a successful edit of the GitHub application configuration.

#### Scenario 11: Downloading a user’s personal data

- **Role Involved:** Admin
- **Objective:** Test the functionality of downloading a user’s personal data from the system.
- **Preconditions:**
  - The user is logged in to their admin account.
  - The settings drawer is open.
  - There are existing users in the system.
- **Test Steps:**
  1. Unfold the “Users” menu.
  2. Select the desired test user with the “Member” role by clicking the three purple dots at the start of their row.
  3. Click the “Download personal data” option.
- **Expected Outcome:**
  - A JSON file is downloaded to the admin’s machine.
  - The downloaded JSON file should contain all the necessary personal data of the selected user (e.g., name, email, roles, etc.).
  - The file format should be valid JSON and the content inside should not have any syntax errors.

#### Scenario 12: Delete an authorization method

- **Role Involved:** Developer
- **Objective:** Test the functionality of deleting an authorization method in the application settings.
- **Preconditions:**

- The user is logged in to their developer account.
- The settings drawer is open.
- There is at least one existing authorization method already set up.
- **Test Steps:**
  1. Unfold the “Authorizations” menu.
  2. Click on the red bin icon next to an existing authorization method.
- **Expected Outcome:**
  - The authorization method is removed from the list and no longer visible in the Authorizations section.
  - A prompt or message indicating successful deletion of the authorization method appears.

### Scenario 13: Deactivate a user

- **Role Involved:** Admin
- **Objective:** Test deactivation of a user’s account
- **Preconditions:**
  - The user is logged in to their admin account.
  - The settings drawer is open.
  - A test user with “Member” role and active status exists in the system.
- **Test Steps:**
  1. Unfold the “Users” menu.
  2. Select the desired test user with the “Member” role by clicking the three purple dots at the start of their row.
  3. Click the “Deactivate” option to deactivate the selected user’s account.
- **Expected Outcome:**
  - The selected user’s status in the list should change from green checked circle to a red circle with a dash, indicating that their account has been deactivated.
  - Attempting to connect as the deactivated user should result in an error.

### Scenario 14: Activate a user

- **Role Involved:** Admin
- **Objective:** Test activation of a user’s account
- **Preconditions:**
  - The user is logged in to their admin account.
  - The settings drawer is open.
  - A test user with “Member” role and inactive status exists in the system.
- **Test Steps:**
  1. Unfold the “Users” menu.
  2. Select the desired test user with the “Member” role by clicking the three purple dots at the start of their row.
  3. Click the “Activate” option to activate the selected user’s account.

- **Expected Outcome:**
  - The selected user’s status in the list should change from red circle with a dash to a green checked circle, indicating that their account has been activated.
  - Attempting to connect as the activated user should succeed.

#### Scenario 15: Change a user’s password

- **Role Involved:** Admin
- **Objective:** Test changing a user’s password
- **Preconditions:**
  - The user is logged in to their admin account.
  - The settings drawer is open.
  - A test user with “Member” role and active status exists in the system.
- **Test Steps:**
  1. Unfold the “Users” menu.
  2. Select the desired test user with the “Member” role by clicking the three purple dots at the start of their row.
  3. Click the “Reset Password” option to open the password change dialog.
  4. A dialog prompts you to select a new password for the user.
  5. Click the “Change Password” button.
- **Expected Outcome:**
  - Attempting to connect as the selected user with the old password should throw an error.
  - Attempting to connect as the selected user with the new password should work as a normal login.

#### Scenario 16: Delete a user

- **Role Involved:** Admin
- **Objective:** Test deletion of a user’s account
- **Preconditions:**
  - The user is logged in to their admin account.
  - The settings drawer is open.
  - A test user with “Member” role and active status exists in the system.
- **Test Steps:**
  1. Unfold the “Users” menu.
  2. Select the desired test user with the “Member” role by clicking the three purple dots at the start of their row.
  3. Click the “Delete” option to delete the selected user’s account.
- **Expected Outcome:**
  - The selected user should disappear from the user list.
  - Attempting to connect as the deactivated user should throw an error.

#### Scenario 17: Logout

- **Role Involved:** Developer
- **Objective:** Test the logout feature
- **Preconditions:**
  - The user is logged in to their account.
  - The settings drawer is open.
- **Test Steps:**
  1. Click the “Logout” button at the bottom of the drawer.
- **Expected Outcome:**
  - The user should be redirected to the login page and lose access to the dashboard and other account-specific features.

#### Scenario 18: Adding a new authorization method - OAuth

- **Role Involved:** Developer
- **Objective:** Test the functionality of adding a new OAuth authorization in the application.
- **Preconditions:**
  - The user is logged in to their developer account.
  - The settings drawer is open.
- **Test Steps:**
  1. Unfold the “Authorizations” menu.
  2. Press the “Add authorization” button to initiate the creation process for a new authorization method.
  3. Fill in the desired label for the new authorization method.
  4. Select the “OAuth” radio button.
  5. Click the “Login with Github” button.
  6. You are redirected to Github’s OAuth setup page, on which you should approve any requests to add a new OAuth connection.
- **Expected Outcome:**
  - A toast message appears at the bottom of the screen to inform the user that the new authorization has been created successfully.
  - The newly added authorization is displayed in the “Authorizations” table within the previous sub-menu inside the settings drawer.
  - The new authorization method also becomes available within the “Deploy a project” modal, allowing users to choose it for their projects.

#### Scenario 19: Adding a new authorization method - Personal Access Token

- **Role Involved:** Developer
- **Objective:** Test the functionality of adding a new Personal Access Token authorization in the application.
- **Preconditions:**
  - The user is logged in to their developer account.
  - The settings drawer is open.
- **Test Steps:**

1. Unfold the “Authorizations” menu.
  2. Press the “Add authorization” button to initiate the creation process for a new authorization method.
  3. Fill in the desired label for the new authorization method.
  4. Select the “Personal Access Token” radio button.
  5. Fill in your desired personal access token in the text field.
  6. Press the “Confirm” button to complete the flow.
- **Expected Outcome:**
    - A toast message appears at the bottom of the screen to inform the user that the new authorization has been created successfully.
    - The newly added authorization is displayed in the “Authorizations” table within the previous sub-menu inside the settings drawer.
    - The new authorization method also becomes available within the “Deploy a project” modal, allowing users to choose it for their projects.

#### Scenario 20: Adding a new authorization method - Deploy Key

- **Role Involved:** Developer
- **Objective:** Test the functionality of adding a new Deploy Key authorization in the application.
- **Preconditions:**
  - The user is logged in to their developer account.
  - The settings drawer is open.
- **Test Steps:**
  1. Unfold the “Authorizations” menu.
  2. Press the “Add authorization” button to initiate the creation process for a new authorization method.
  3. Fill in the desired label for the new authorization method.
  4. Select the “Deploy Key” radio button.
  5. Fill in your desired deploy key in the text area.
  6. Press the “Confirm” button to complete the flow.
- **Expected Outcome:**
  - A toast message appears at the bottom of the screen to inform the user that the new authorization has been created successfully.
  - The newly added authorization is displayed in the “Authorizations” table within the previous sub-menu inside the settings drawer.
  - The new authorization method also becomes available within the “Deploy a project” modal, allowing users to choose it for their projects.

#### Scenario 21: Project setup

- **Role Involved:** Developer
- **Objective:** Test the functionality of creating and deploying a new project within the platform.
- **Preconditions:**
  - The user is logged in to their developer account.

- The dashboard page is open.
- **Test Steps:**
  1. Click on the “Deploy a project” button on the top-right corner of the page.
  2. Select the desired authorization from the configured authorizations.
  3. Select a GitHub repository from the repositories available to the authorization method.
  4. Select a branch to deploy from.
  5. Enter your Docker Compose file’s path inside the repository.
  6. Move to the next step.
  7. Use the sliders to set your desired allocations for CPUs, Memory and Disk.
  8. Press the “Create” button
- **Expected Outcome:**
  - A new project card should appear in the dashboard’s project list.
  - After a short wait, the project’s status should become “Started” or “Error”.

#### **Scenario 22: Deploying a project from a public repository without an authorization method**

- **Role Involved:** Developer
- **Objective:** Test deploying a public repository without any authorization method setup
- **Preconditions:**
  - The user is logged in to their developer account.
  - The dashboard page is open.
- **Test Steps:**
  1. Click on the “Deploy a project” button on the top-right corner of the page.
  2. Select the “N/A” authorization method.
  3. Fill in your desired public Github repository name.
  4. Fill in your desired branch to deploy from.
  5. Enter your Docker Compose file’s path inside the repository.
  6. Move to the next step.
  7. Use the sliders to set your desired allocations for CPUs, Memory and Disk.
  8. Press the “Create” button
- **Expected Outcome:**
  - A new project card should appear in the dashboard’s project list.
  - After a short wait, the project’s status should become “Started” or “Error”.

#### **Scenario 23: User display preferences in topology view**

- **Role Involved:** Developer

- **Objective:** To test the functionality of displaying user preferences in the topology view for a Haddock project.
- **Preconditions:**
  - The developer should be logged into the Haddock platform and have access to the specified project.
  - That project should have a compose file inside that defines at least one service in it.
- **Test Steps:**
  1. Navigate to the dashboard page.
  2. Locate the desired project on the list.
  3. Click on the project name to open its details page.
  4. Locate the Topology tab (default one).
  5. Manipulate the nodes (services) by clicking and dragging them to rearrange them in a preferred layout within the topology view.
  6. Test the functionality of showing or hiding connections between services by toggling this option on or off, respectively.
  7. Refresh the page to verify the changes made.
- **Expected Outcome:**
  - Upon refreshing the page, the rearranged nodes (services) should be displayed in the preferred layout selected by the developer.
  - If the option to display connections was toggled off, there should be no visible lines connecting the services. Conversely, if the option to display connections was toggled on, all connections between the services should be clearly visible.

#### Scenario 24: Edit project informations

- **Role Involved:** Developer
- **Objective:** To test the functionality of editing project information on the Haddock platform.
- **Preconditions:**
  - A developer account is set up and logged in.
  - The project to be edited is already deployed on Haddock.
- **Test Steps:**
  1. Navigate to the dashboard.
  2. Locate the project to be edited from the list and click on it.
  3. In the newly opened page, navigate to the settings tab.
  4. Click on the “Edit this Project” button that appears.
  5. Update the name and description fields with new values.
  6. Click the ‘Edit this Project’ button to save the changes.
- **Expected Outcome:**
  - A toast message should be displayed on the bottom right side of the screen confirming the successful update of the project information.
  - The updated name should be reflected in the header of the project page.
  - The updated name and description should appear in the list of projects

on the dashboard.

#### Scenario 25: Edit an existing project's authorization method

- **Role Involved:** Developer
- **Objective:** Test updating a project's authorization method.
- **Preconditions:**
  - The user is logged in to their developer account.
  - The dashboard page is open.
  - There is an existing deployed project available.
- **Test Steps:**
  1. On the dashboard, select an existing project from among the list of projects displayed.
  2. Move to the “Settings” tab on the project details page by clicking on it.
  3. Press the “Edit this project” button located at the top right corner of the settings page.
- **Expected Outcome:**
  - A new toast message is generated once the edit button is pressed, confirming that the authorization method was updated successfully.

#### Scenario 26: Monitoring a deployed project

- **Role Involved:** Developer
- **Objective:** Test the ability to monitor CPU, Memory, and Disk Usages for a deployed project, as well as view the project's docker logs.
- **Preconditions:**
  - The user is logged in to their developer account.
  - The project's details page should be open.
- **Test Steps:**
  1. Click on the “Monitoring” tab to move to that section.
  2. Check that the CPU, Memory, and Disk Usages are displayed as a percentage ring chart and a detailed histogram for each metric.
  3. Verify that the charts are updated in real-time to reflect the current status of the project.
  4. Scroll down to view the project's docker logs at the bottom of the page.
- **Expected Outcome:**
  - The project details page and the “Monitoring” tab should be accessible for the selected project.
  - The CPU, Memory, and Disk Usages are displayed as a percentage ring chart and a histogram for each metric.
  - The charts should update in real-time to reflect the current status of the project.
  - The project's docker logs can be viewed at the bottom of the page.

### Scenario 27: Service View

- **Role Involved:** Developer
- **Objective:** Verify that the Service View of a deployed Haddock project correctly displays all services in a React Flow.
- **Preconditions:**
  - The developer should be logged into the Haddock platform and have access to the specified project.
  - That project should have a compose file inside that defines services in it.
- **Test Steps:**
  1. Navigate to the dashboard page.
  2. Locate the desired project on the list.
  3. Click on the project name to open its details page.
  4. Locate the Topology tab (default one).
  5. Verify that the React Flow visualization is loaded on the Topology tab
- **Expected Outcome:**
  - The entire list of services from the Compose file should be displayed in the Flow representation.
  - Each service node should be clickable and provide additional information when clicked.

### Scenario 28: Service Status

- **Role Involved:** Developer
- **Objective:** To verify that the Service Status is correctly displayed for a deployed Haddock project with at least one service.
- **Preconditions:**
  - The developer should be logged into the Haddock platform and have access to the specified project.
  - That project should have a compose file inside that defines at least one service in it.
- **Test Steps:**
  1. Navigate to the dashboard page.
  2. Locate the desired project on the list.
  3. Click on the project name to open its details page.
  4. Locate the Topology tab (default one).
  5. Click on the selected service node.
  6. Ensure that the drawer on the right side of the screen is opened to display the details of the chosen service
- **Expected Outcome:**
  - The drawer should open on the right side of the screen, revealing the details of the selected service.
  - The status of the service should be displayed in the drawer.
  - The image of the service should be visible in the drawer.

### Scenario 29: Service Configuration: environment variable

- **Role Involved:** Developer
- **Objective:** Testing that environment variables defined in a Compose file for a Haddock project service are correctly displayed in the “Environment Variables” section on the configuration page.
- **Preconditions:**
  - The developer should be logged into the Haddock platform and have access to the specified project.
  - That project should have a compose file inside that defines at least one service in it.
  - The service should have environment variables defined in the compose file.
- **Test Steps:**
  1. Navigate to the dashboard page.
  2. Locate the desired project on the list.
  3. Click on the project name to open its details page.
  4. Locate the Topology tab (default one).
  5. Click on the selected service node.
  6. In the drawer, navigate to and open the configuration tab.
- **Expected Outcome:**
  - The environment variable defined in the Compose file should be correctly displayed under the “Environment Variables” section in the configuration page.
  - The displayed value should match the actual value defined within the Compose file.

### Scenario 30: Service Configuration: depends on

- **Role Involved:** Developer
- **Objective:** Verify that the “depends\_on” services listed in the compose for a specific service are correctly displayed on the configuration page of the Haddock project.
- **Preconditions:**
  - The developer should be logged into the Haddock platform and have access to the specified project.
  - That project should have a compose file inside that defines at least one service in it.
  - The service should have depends\_on defined in the compose file.
- **Test Steps:**
  1. Navigate to the dashboard page.
  2. Locate the desired project on the list.
  3. Click on the project name to open its details page.
  4. Locate the Topology tab (default one).
  5. Click on the selected service node.
  6. In the drawer, navigate to and open the configuration tab.

7. Locate the “depends on” section on the configuration page.
- **Expected Outcome:**
    - All the services listed as “depends\_on” in the compose for that service should be displayed correctly under the “depends on” section on the configuration page.

### Scenario 31: Service Configuration: ressource limits

- **Role Involved:** Developer
- **Objective:** Test the display of resource limits in the service configuration for a deployed Haddock project.
- **Preconditions:**
  - The developer should be logged into the Haddock platform and have access to the specified project.
  - That project should have a compose file inside that defines at least one service in it.
  - The service should have resource limits defined in the compose file.
- **Test Steps:**
  1. Navigate to the dashboard page.
  2. Locate the desired project on the list.
  3. Click on the project name to open its details page.
  4. Locate the Topology tab (default one).
  5. Click on the selected service node.
  6. In the drawer, navigate to and open the configuration tab.
  7. Locate the “resource limits” section in the configuration page.
- **Expected Outcome:**
  - The resource limits (memory, CPU, etc.) specified in the service’s Compose file should be correctly displayed within the resource limits section of the service’s configuration settings.

### Scenario 32: Service Configuration: User

- **Role Involved:** Developer
- **Objective:** Test the functionality of displaying user information in the Service Configuration tab for a given service within a deployed Haddock project.
- **Preconditions:**
  - The developer should be logged into the Haddock platform and have access to the specified project.
  - That project should have a compose file inside that defines at least one service in it.
  - The service should be configured to have user defined with UID and GID.
- **Test Steps:**
  1. Navigate to the dashboard page.
  2. Locate the desired project on the list.

3. Click on the project name to open its details page.
  4. Locate the Topology tab (default one).
  5. Click on the selected service node.
  6. In the drawer, navigate to and open the configuration tab.
  7. Locate the “user” section on the configuration page.
- **Expected Outcome:**
    - The user information provided in the Compose file (UID, GID) is correctly displayed in the Service Configuration’s User section.

### Scenario 33: Service network: Ports list

- **Role Involved:** Developer
- **Objective:** Test the display of all ports defined in the compose file under the “Ports” section for a targeted service.
- **Preconditions:**
  - The developer should be logged into the Haddock platform and have access to the specified project.
  - That project should have a compose file inside that defines at least one service in it.
  - The compose file should define ports for that specific service.
- **Test Steps:**
  1. Navigate to the dashboard page.
  2. Locate the desired project on the list.
  3. Click on the project name to open its details page.
  4. Locate the Topology tab (default one).
  5. Click on the selected service node.
  6. In the drawer, navigate to and open the Network tab.
  7. Locate the “Ports” section.
- **Expected Outcome:**
  - The “Ports” section should be visible in the Networks tab for the selected service.
  - The list of all the ports defined in the compose file should be displayed correctly in the “Ports” section.

### Scenario 34: Service Network: Network list

- **Role Involved:** Developer
- **Objective:** Testing that the defined networks in a Haddock project’s compose file are displayed under the Networks tab of the targeted service.
- **Preconditions:**
  - The developer should be logged into the Haddock platform and have access to the specified project.
  - That project should have a compose file inside that defines at least one service in it.
  - The compose file should define networks for that specific service.
- **Test Steps:**

1. Navigate to the dashboard page.
  2. Locate the desired project on the list.
  3. Click on the project name to open its details page.
  4. Locate the Topology tab (default one).
  5. Click on the selected service node.
  6. In the drawer, navigate to and open the Network tab.
  7. Locate the “Networks” section.
- **Expected Outcome:**
    - The “Networks” section should be visible in the Networks tab for the selected service.
    - The list of all the networks defined in the compose file should be displayed correctly in the “Networks” section.

### Scenario 35: Service network redirections: No subdomain

- **Role Involved:** Developer
- **Objective:** To test the functionality of creating a network redirection without a subdomain in a Haddock project.
- **Preconditions:**
  - The developer should be logged into the Haddock platform and have access to the specified project.
  - That project should have a compose file inside that defines at least one service in it.
  - The compose file should define ports for that specific service.
  - At least two domains must be pre-configured in the system to use in the redirection.
- **Test Steps:**
  1. Navigate to the dashboard page.
  2. Locate the desired project on the list.
  3. Click on the project name to open its details page.
  4. Locate the Topology tab (default one).
  5. Click on the selected service node.
  6. In the drawer, navigate to and open the Network tab.
  7. Locate the Redirections section.
  8. Click on the “Add redirection” button.
  9. Choose a port from the available options in the dialog box.
  10. Leave the subdomain field empty and fill in one of the pre-configured domains.
  11. Click on the “Create” button to save the redirection settings.
- **Expected Outcome:**
  - It must not be possible to choose the primary domain.
  - A notification message should be displayed, confirming that the redirection has been created successfully.
  - The newly created redirection should appear in the list of network redirections with the specified port and domain.
  - By accessing the Internet using the filled-in domain, you should be

able to reach the targeted service via the created network redirection.

### Scenario 36: Service Networks redirections: With subdomain

- **Role Involved:** Developer
- **Objective:** To test the functionality of creating a network redirection with a subdomain in a Haddock project.
- **Preconditions:**
  - The developer should be logged into the Haddock platform and have access to the specified project.
  - That project should have a compose file inside that defines at least one service in it.
  - The compose file should define ports for that specific service.
  - One or more domains should be pre-configured in the system to use in the redirection.
- **Test Steps:**
  1. Navigate to the dashboard page.
  2. Locate the desired project on the list.
  3. Click on the project name to open its details page.
  4. Locate the Topology tab (default one).
  5. Click on the selected service node.
  6. In the drawer, navigate to and open the Network tab.
  7. Locate the Redirections section.
  8. Click on the “Add redirection” button.
  9. Choose a port from the available options in the dialog box.
  10. Fill in the subdomain field with a valid subdomain.
  11. Click on the “Create” button to save the redirection settings.
- **Expected Outcome:**
  - A notification message should be displayed, confirming that the redirection has been created successfully.
  - The newly created redirection should appear in the list of network redirections with the specified port and full domain.
  - By accessing the Internet using the filled-in domain, you should be able to reach the targeted service via the created network redirection.

### Scenario 37: Service Network redirection: Delete redirection

- **Role Involved:** Developer
- **Objective:** Test the successful deletion of a service network redirection in a Haddock project.
- **Preconditions:**
  - The developer should be logged into the Haddock platform and have access to the specified project.
  - That project should have a compose file inside that defines at least one service in it.
  - The service already has an active redirection.

- **Test Steps:**
  1. Navigate to the dashboard page.
  2. Locate the desired project on the list.
  3. Click on the project name to open its details page.
  4. Locate the Topology tab (default one).
  5. Click on the selected service node.
  6. In the drawer, navigate to and open the Network tab.
  7. Locate the Redirections section.
  8. Click on the trash button next to the selected redirection.
  9. Click on the delete button in the dialog.
- **Expected Outcome:**
  - A notification should be displayed saying that the redirection has been successfully deleted.
  - The deleted redirection should no longer appear in the Redirections list.
  - It should no longer be possible to access the service on the internet using the previous domain associated with the deleted redirection.

#### Scenario 38: Start a service

- **Role Involved:** Developer
- **Objective:** Test the functionality to start a service in a deployed Haddock project.
- **Preconditions:**
  - The developer should be logged into the Haddock platform and have access to the specified project.
  - That project should have a compose file inside that defines at least one service in it.
  - The service should be in a stopped or error state.
- **Test Steps:**
  1. Navigate to the dashboard page.
  2. Locate the desired project on the list.
  3. Click on the project name to open its details page.
  4. Locate the Topology tab (default one).
  5. Click on the selected service node.
  6. In the drawer, navigate to and open the Status tab (default one).
  7. Identify the 3 dots action button in the “Current Status” section and click on it.
  8. Click on the “Start” button that appears after clicking the 3 dots action button.
- **Expected Outcome:**
  - The service should transition from a stopped or error state to a running state.
  - A notification should be displayed, confirming that the service is now running.

### Scenario 39: Restart service

- **Role Involved:** Developer
- **Objective:** Test the functionality to restart a service in a deployed Haddock project.
- **Preconditions:**
  - The developer should be logged into the Haddock platform and have access to the specified project.
  - That project should have a compose file inside that defines at least one service in it.
  - The service should be in a running state.
- **Test Steps:**
  1. Navigate to the dashboard page.
  2. Locate the desired project on the list.
  3. Click on the project name to open its details page.
  4. Locate the Topology tab (default one).
  5. Click on the selected service node.
  6. In the drawer, navigate to and open the Status tab (default one).
  7. Identify the 3 dots action button in the “Current Status” section and click on it.
  8. Click on the “Restart” button that appears after clicking the 3 dots action button.
- **Expected Outcome:**
  - The selected service should restart successfully.
  - A notification should be displayed confirming that the service has been restarted.

### Scenario 40: Stop a service

- **Role Involved:** Developer
- **Objective:** Test the functionality to stop a service in a deployed Haddock project.
- **Preconditions:**
  - The developer should be logged into the Haddock platform and have access to the specified project.
  - That project should have a compose file inside that defines at least one service in it.
  - The service should be in a running state.
- **Test Steps:**
  1. Navigate to the dashboard page.
  2. Locate the desired project on the list.
  3. Click on the project name to open its details page.
  4. Locate the Topology tab (default one).
  5. Click on the selected service node.
  6. In the drawer, navigate to and open the Status tab (default one).
  7. Identify the 3 dots action button in the “Current Status” section and

click on it.

8. Click on the “Stop” button that appears after clicking the 3 dots action button.

- **Expected Outcome:**
- The service should now be in a stopped state.
- A notification should be displayed, confirming that the service is now stopped.

#### Scenario 41: Display project environment variable

- **Role Involved:** Developer
- **Objective:** Test the functionality to display project environment variables in a Haddock-deployed project.
- **Preconditions:**
  - A project has been deployed using Haddock with a .env file containing some environment variables.
  - The developer should be logged into the Haddock platform and have access to the specified project.
- **Test Steps:**
  1. Navigate to the dashboard page.
  2. Locate the desired project on the list.
  3. Click on the project name to open its details page.
  4. Go to the “Settings” tab within the project page.
  5. Locate and expand the “Manage Environment Variables” section.
- **Expected Outcome:**
  - Upon clicking on the “Manage Environment Variables,” the developer should see all the environment variables defined in the .env file associated with the tested project.

#### Scenario 42: Delete environment variable

- **Role Involved:** Developer
- **Objective:** Test the ability to delete an environment variable in a deployed project on the platform.
- **Preconditions:**
  - A Haddock-deployed project is accessible and active.
  - The project has at least one environment variable set.
- **Test Steps:**
  1. Navigate to the dashboard page.
  2. Locate the desired project on the list.
  3. Click on the project name to open its details page.
  4. Go to the “Settings” tab within the project page.
  5. Locate and expand the “Manage Environment Variables” section.
  6. Identify the environment variable to be deleted.
  7. Click on the “Delete” button associated with the selected environment variable.

- **Expected Outcome:**
  - A notification should appear on the right bottom side of the screen confirming that the action was successful.
  - The deleted environment variable should no longer be visible in the list of environment variables under the “Manage Environment Variables” section.

#### Scenario 43: Add an environment variable

- **Role Involved:** Developer
- **Objective:** To add an environment variable to a deployed Haddock project and verify that it appears in the list and within the VM environment.
- **Preconditions:**
  - A valid Haddock account is set up.
- **Test Steps:**
  1. Navigate to the dashboard page.
  2. Locate the desired project on the list.
  3. Click on the project name to open its details page.
  4. Go to the “Settings” tab within the project page.
  5. Locate and expand the “Manage Environment Variables” section.
  6. Fill out the form by providing a key and value for the new environment variable, ensuring it is not marked as secret.
  7. Click on the “Add” button to submit the new environment variable
  8. Stop and Start the project to apply the modifications.
- **Expected Outcome:**
  - The new environment variable should be created successfully.
  - The newly added environment variable should be displayed in the list within the “Manage environment variables” section.
  - The newly added environment variable should be successfully applied and accessible in the VM environment when running the project.
  - Since it’s not secret, the value of the variable should clearly be displayed within the list without any obfuscation or masking.

#### Scenario 44: Add a secret environment variable

- **Role Involved:** Developer
- **Objective:** Test the functionality of creating a secret environment variable for a deployed Haddock project.
- **Preconditions:**
  - A valid Haddock account is set up.
- **Test Steps:**
  1. Navigate to the dashboard page.
  2. Locate the desired project on the list.
  3. Click on the project name to open its details page.
  4. Go to the “Settings” tab within the project page.
  5. Locate and expand the “Manage Environment Variables” section.

6. Fill out the form by providing a key and value for the new environment variable.
  7. Check the box marked 'Secret'.
  8. Click on the 'Add' button to create the new environment variable.
  9. Stop and Start the project to apply the modifications.
- **Expected Outcome:**
    - The new environment variable should be created successfully.
    - The newly added environment variable should be successfully applied and accessible in the VM environment when running the project.
    - The newly added environment variable should be displayed in the list within the "Manage environment variables" section.
    - Since the variable is secret, the value should not be visible within the project settings page.

#### Scenario 45: Edit non secret environment variable

- **Role Involved:** Developer
- **Objective:** To test the functionality of editing a non-secret environment variable in the Haddock project.
- **Preconditions:**
  - The developer should be logged into the Haddock platform and have access to the specified project.
- **Test Steps:**
  1. Navigate to the dashboard page.
  2. Locate the desired project on the list.
  3. Click on the project name to open its details page.
  4. Go to the "Settings" tab within the project page.
  5. Locate and expand the "Manage Environment Variables" section.
  6. Identify the environment variable that will be edited.
  7. Replace both the key and the value of the selected environment variable with new values.
  8. Click the 'Save' button to save the changes
- **Expected Outcome:**
  - The edited environment variable should be updated on the list.
  - The updated environment variable should be reflected in the VM environment for the deployed Haddock project.

#### Scenario 46: Change environment variable from non secret to secret

- **Role Involved:** Developer
- **Objective:** Test the functionality of changing an environment variable from non-secret to secret within a project.
- **Preconditions:**
  - The developer should be logged into the Haddock platform and have access to the specified project.
- **Test Steps:**

1. Navigate to the dashboard page.
  2. Locate the desired project on the list.
  3. Click on the project name to open its details page.
  4. Go to the “Settings” tab within the project page.
  5. Locate and expand the “Manage Environment Variables” section.
  6. Check on the secret state for the variable you want to edit.
  7. Save the updated environment variable settings.
- **Expected Outcome:**
    - The environment variable should be saved and updated as a secret variable.
    - The value of the environment variable shouldn’t be displayed anymore.
    - The checkbox “Secret” should be checked.

#### Scenario 47: Edit secret environment variable

- **Role Involved:** Developer
- **Objective:** Test the functionality of editing a secret environment variable in a deployed Haddock project.
- **Preconditions:**
  - The developer should be logged into the Haddock platform and have access to the specified project.
- **Test Steps:**
  1. Navigate to the dashboard page.
  2. Locate the desired project on the list.
  3. Click on the project name to open its details page.
  4. Go to the “Settings” tab within the project page.
  5. Locate and expand the “Manage Environment Variables” section.
  6. Identify the secret environment variable that will be edited.
  7. Update the key and value of the variable as desired.
  8. Click the ‘Save’ button to save the changes
- **Expected Outcome:**
  - The key and value of the identified secret environment variable should be updated with the new values provided by the developer.
  - Since the variable is secret, the updated value cannot be viewed. However, the key and the fact that it is a secret variable should remain visible.
  - Since the variable is secret, the “Secret” checkbox should be checked and cannot be changed.

#### Scenario 48: Start a Project

- **Role Involved:** Developer
- **Objective:** Test the functionality of starting a Haddock project that is either stopped or in error.
- **Preconditions:**
  - A user account with appropriate permissions to start a project.

- The project to be started is already deployed on Haddock.
- **Test Steps:**
  1. Navigate to the dashboard page.
  2. Locate the desired project on the list, which is currently either stopped or in an error state.
  3. Click on the project name to open its details page.
  4. Find and click the “Start” button located near the status of the project.
- **Expected Outcome:**
  - A notification message should appear on the bottom right side of the screen saying that the project is now starting.
  - The project status should change to “Starting”.
  - After a short delay, the project status should change to “Running”. This indicates that the feature works correctly and the project has successfully started or resumed.

#### Scenario 49: Stop Project

- **Role Involved:** Developer
- **Objective:** To verify that the ‘Stop Project’ functionality stops the project and all associated services.
- **Preconditions:**
  - A developer account is set up and logged in.
  - The project to be stopped is already deployed and ‘Running’ on Haddock.
- **Test Steps:**
  1. Navigate to the dashboard page.
  2. Select the desired ‘Running’ state project from the list.
  3. Click on the ‘Stop Project’ button.
  4. In the modal that appears, click on the ‘Stop Project’ button again to confirm the action.
- **Expected Outcome:**
  - A notification with a success message about stopping the project should appear on the bottom right side of the screen.
  - The selected project’s state should change from ‘Running’ to ‘Stopped’.
  - All related services associated with the project should be stopped and their status updated accordingly.

#### Scenario 50: Update Project

- **Role Involved:** Developer
- **Objective:** To verify that a Haddock project can be updated from its GitHub source, and the changes are reflected in the deployed application.
- **Preconditions:**
  - A Haddock project is already deployed and accessible.
  - The project’s source code resides on a GitHub repository.

- Developer has necessary access rights to make commits on the GitHub repository.
- The deployment system is configured to pull updates from GitHub automatically or manually.
- **Test Steps:**
  1. Navigate to the GitHub repository hosting the Haddock project’s source code.
  2. Make a commit containing desired changes, ensuring that the changes are relevant to the project and do not break any existing functionality.
  3. Navigate to the dashboard page.
  4. Locate the desired project on the list
  5. Click on the project name to open its details page.
  6. Find and click the “Update” button located near the status of the project.
  7. Confirm the update by clicking on the appropriate modal or dialog box, as prompted by the system.
- **Expected Outcome:**
  - A notification message should appear on the bottom right side of the screen saying that the update process has started.
  - The project will begin to recompile and restart, incorporating the new modifications from the committed changes in the GitHub repository.
  - Once the project is fully updated, it should now be running with the new modifications applied.

#### Scenario 51: Recreate Project

- **Role Involved:** Developer
- **Objective:** To verify the functionality of recreating a stopped or error Haddock project from its page.
- **Preconditions:**
  - A Haddock project is already deployed and accessible.
  - The project is currently in a stopped or error state.
  - A user account with appropriate permissions to recreate the project.
- **Test Steps:**
  1. Navigate to the dashboard page.
  2. Locate the desired project on the list, which is currently either stopped or in an error state.
  3. Click on the project name to open its details page.
  4. Find and click the “Recreate” button located near the status of the project.
  5. Confirm the recreation action in the appearing modal.
- **Expected Outcome:**
  - A notification appears on the right bottom side of the screen, confirming the start of project recreation process.
  - The project status should change to “Starting”.
  - After a short delay, the project status should change to “Running”.

This indicates that the feature works correctly and the project has successfully been recreated.

#### **Scenario 52: Delete project**

- **Role Involved:** Developer
- **Objective:** To verify that a project can be successfully deleted and the user is redirected to the dashboard page where the deleted project is no longer displayed.
- **Preconditions:**
  - A developer account is set up and logged in.
  - The project to be deleted is already deployed on Haddock.
- **Test Steps:**
  1. Navigate to the Dashboard page.
  2. Identify the desired project on the dashboard.
  3. Click on the project to access its details page.
  4. Locate and click on the 'Settings' tab.
  5. Find the 'Delete Project' button, then click on it.
  6. Wait for 3 seconds after the modal appears.
  7. Click on the 'Delete Project' button in the modal.
- **Expected Outcome:**
  - The developer should be redirected to the Dashboard page.
  - The deleted project should no longer appear on the Dashboard.
  - There should be a confirmation message indicating that the project has been successfully deleted.