



**ArchivesSpace**

a community served by ✦Lyrasis

# **Creating Actionable Feature Requests**

**March 26, 2024**



# Actionable

- Short
- Unambiguous
- Complete



- [illegible]





# Behavior Scenarios

- User – ArchivesSpace interaction
- Given / When / Then - Gherkin
  - **Given** some initial state
  - **When** the user makes an action
  - **Then** the system responds with an outcome



# An example

**Scenario:** As a staff member **I want to** log-in **so that** I can access restricted functionality

**Given** a user has already created a staff account

**And** the user is not logged in

**When** the user browses the login form

**And** the user submits their credentials

**Then** the user is successfully logged in



# Gherkin Keywords

- **Feature:** As a [persona], I want to [do something], so that [an outcome is achieved]
- **Background:** context
- **Rule:** a group of scenarios
- **Scenario:** Summary of the behavior
- **Given:** initial state
- **When:** User actions
- **Then:** expected outcome
- **And / But:** additional steps





# Given / When / Then order

- One scenario - one behavior
  - One When-Then pair
- Ordered steps:
  - Given
  - When
  - Then



# Step phrasing

- Steps are ordered
- Detailed
- Use third person
- Subject-predicate
- Present tense
- Declarative not imperative







# Real world example

- As an Archivist, I want to identify an Agent linked to a (Resource | Digital Object | Accession | Resource Component | Digital Object Component) as "Primary"
- Only relevant to MARCXML exports
- Not to appear on the PUI
- <https://archivesspace.atlassian.net/browse/ANW-504>



# Feature declaration

**As an** Archivist,

**I want to** mark none or exactly one agent link of a Resource | Digital Object | Accession | Resource Component | Digital Object Component as "Primary",

**so that** the primary agent appears as a 1xx field in MARCXML exports.



**ArchivesSpace**

a community served by **+**Lyrasis



# Background

The primary agent should appear as a 1xx field in MARCXML exports.

It should not be viewable anywhere in the PUI.

On the SUI only viewable on the edit and show views of a Resource.



# Scenario 1

**Scenario:** As a staff User, I want to mark exactly one agent as primary on a Resource so that it appears as primary on the edit view of the SUI

**Given** the user opens the edit form of a resource on SUI

**And** the resource has three agent links (two creators and one subject)

**And** the second agent link is already marked as primary

**When** the user marks the first agent as primary

**Then** then second agent does not appear as primary anymore

**And** the first agent appears marked as primary



# Scenario 2

**Scenario:** As a staff user, I want to see the primary agent of a resource, on the show view of the resource on the SUI, so that it becomes clear that there is a primary agent defined for that resource.

**Given** a staff user marks a specific agent link as primary on a resource

**When** the user opens the show view for that resource on the SUI

**Then** the agent link is marked as primary



# Scenario 3

**Scenario:** As an Archivist, I want to see a 1xx field for the primary agent in the MARCXML export of a resource

**Given** the user opens the edit form of a resource on SUI

**And** the resource has three agent links (two creators and one subject)

**And** the second agent link is already marked as primary

**When** the user downloads a MARCXML export of that resource

**Then** a 1xx field appears in the MARCXML for the primary agent link



# Scenario Outline

**Scenario Outline:** As a staff user, I want to open a list of associated entities directly from an accession view

**Given** a staff user opens a "<view type>" view of an accession

**When** the user selects "<entity type>" from a "Related" dropdown

**Then** a list of related "<entity type>" entries appears

## Examples:

View Type	Entity type
show	Resources
show	Top Containers
show	Accessions
edit	Resources
edit	Top Containers
edit	Accessions





# Wiki Pages updated

ArchivesSpace

Home

Recent

Spaces

Teams

Apps

Templates

Create

Search

Share

Following

ArchivesSpace / Contribute to ArchivesSpace... / How to Request a New Feature 

UNPUBLISHED CHANGES

## How to Request a New Feature

Owned by BradleyW (Unlicensed) ---  
Last updated: Feb 29, 2024 by Thomas Dimopoulos • 2 min read • 134 people viewed • Legacy editor

(If you want to report a bug, see [How to Report a Bug](#).)

The ArchivesSpace community encourages requests for new features to be added to the ArchivesSpace application. New features can increase the application's suitability for supporting management of archival materials, as well as optimize the experience of the application's users.

Before requesting a new feature,

- Upgrade to the latest [release](#) of the application and make sure the feature you want to request is not already present in the application, and
- Review feature requests logged in the [ArchivesSpace development project](#) to see if the feature has already been requested and, perhaps, prioritized for development.

Once you have determined the Feature does not already exist in the application

- Log in to your JIRA account (You will need to [create a JIRA account](#) to submit a ticket).
- Access JIRA and select **Create** issue (Select ArchivesSpace New Workflow as the project).

An actionable feature request requires the following:

- Issue Type Identified as "New Feature"**
- Summary description of the feature requested in the format of "As [role] I request [feature]"** (The roles typically used in ArchivesSpace are System Administrator, Repository Manager, Project Archivist, Archivist, Advanced Data Entry Staff, Beginning Data Entry Staff, and Repository Patron)

**Examples:**

As a repository manager, I would like to link event records to name records  
As a project manager, I would like the ability to declare default values comprehensively through the ArchivesSpace application  
As a repository patron, I would like to be able to view a broad range of digital content within the ArchivesSpace application
- A detailed description of the desired functionality**

Feature requests that include [detailed Behavior Scenarios](#) or [Use Cases](#) can be more easily reviewed and are more likely to be prioritized for development since these use cases provide more detailed information about the requested feature and improve the likelihood of a successful outcome in development.

Feature requests that include use cases that follow the [Given/When/Then format](#) or that include detailed specifications and descriptions are more likely to be selected for development.
- Attached files, screenshots, or specifications that amplify the feature request.**

Upon submission of the ticket, the Development Prioritization Council or members of the ArchivesSpace program team will contact the submitter and all other commenters directly via the ticket.

Ticket submitters and all other individuals interested in the development of the feature request are encouraged to remain vigilant of the ticket and respond to all requests for information. Tickets that have a status of "awaiting more information" are unable to be prioritized or implemented until questions are resolved.

+ Add label

Related pages

Behavior Scenarios

ArchivesSpace

Often read together

How to Report a Bug

ArchivesSpace

Organised together


2024-02-28 TAC/UAC Joint Meeting

Advisory Councils

Often read together

Be the first to add a reaction

Write a comment...



ArchivesSpace


a community served by +Lyrasis





# Wiki Pages updated

## Behavior Scenarios

 Owned by Thimios Dimopoulos ...  
Last updated: 21 minutes ago • 6 min read • 👁 15 people viewed

One or more Behavior Scenarios (also known as Use Cases or Usage Scenarios) are encouraged in a [Feature Request](#) in order to describe the requirements in an unambiguous and compact manner.

Feature requests that include detailed Behavior Scenarios or Use Cases can be more easily reviewed and are more likely to be prioritized for development since these use cases provide more detailed information about the requested feature and improve the likelihood of a successful outcome in development.

### Gherkin or Given / When / Then language

Gherkin, also known as Given / When / Then syntax, is a language for writing Behavior Scenarios. Gherkin is a plain-text language with a simple structure and is intended to be easy enough to learn for non-programmers while allowing the level of specificity required by developers to successfully develop a feature.

Gherkin scenarios are meant to be short and sound like plain English. Each scenario has the following structure:

- Given some initial state
- When an action is taken
- Then an outcome is expected

Here is an example usage scenario in Given / When / Then syntax:

```
1 Feature: As a non-logged in user, I want to login, so that I can access restricted
2   functionality
3
4 Background: As the staff area of ArchivesSpace includes functions that need to be
5   available to staff personnel only, users should go through a login process
6   before accessing it.
```



# Find Out More

- Books and Tutorials:
  - <https://www.manning.com/books/writing-great-specifications>
  - <https://automationpanda.com/2017/01/30/bdd-101-writing-good-gherkin/>
- On our Wiki:
  - <https://archivesspace.atlassian.net/wiki/spaces/ADC/pages/19202060/How+to+Request+a+New+Feature>
  - <https://archivesspace.atlassian.net/wiki/spaces/ADC/pages/3593306113/Behavior+Scenarios>

Get in touch: [ArchivesSpaceHome@lyrasis.org](mailto:ArchivesSpaceHome@lyrasis.org)