## Use Case 1: Adding a resource

The user intends to add a resource to the wiki. Developers and managers may add resources to the wiki.

### Targeted Release

Initial Release

### Precondition

The user is a developer or manager. The user possesses an authentication token.

### Postcondition

A new resource has been added to the server.

### Trigger

The user sends an “Add Resource” request to the API.

### Process

1. Validate the authorization token with the authentication server.
   1. If authentication fails or the user does not provide a token: stop processing the request and return an error to the user.
2. Identify the system role possessed by the user.
   1. If the role is commentor or the user does not possess a role: stop processing the request and return an error to the user.
3. Sanitize and validate the inputs.
4. Add a new entry with given inputs to the database.
5. Add a resource history entry for the new resource.
6. Notify the user of the request’s completion.

## Use Case 2: Deleting a Resource

The user intends to delete a resource from the database. Only managers and the developer that created the resource may delete a resource.

### Targeted Release

Initial Release

### Precondition

The user is a manager or the developer that created the resource. The user possesses an authentication token. The user knows the ID of the resource to be deleted.

### Postcondition

The target resource has been removed from the server.

### Trigger

User sends a “Delete Resource” request to the API.

### Process

1. Validate the authorization token with the authentication server:
   1. If authentication fails or the user does not provide a token: stop processing the request and return an error to the user.
2. Identify the system role possessed by the user:
   1. If the user is a contributor: Only continue if the user id matches the creator id of the target resource.
   2. If the user is a commentor or does not have a role: stop processing the request and return an error to the user.
3. Sanitize and validate the ID.
4. Search the database for a resource that matches the ID:
   1. Resource found:
      1. Delete the resource.
      2. Update the resource history log.
      3. Notify the user of their request’s completion.
   2. Resource not found: Return an error to the user.

## Use Case 3: Listing all resources

The user intends to obtain a list of all resources in the library. Users with any authority level may obtain this list.

### Targeted Release

Initial Release

### Precondition

User possesses an authentication token.

### Postcondition

All resources in the database are returned to the user.

### Trigger

User sends a “List Resources” request to the API.

### Process

1. Validate the authorization token with the authentication server:
   1. If authentication fails or the user does not provide a token: stop processing the request and return an error to the user.
2. Identify the system role possessed by the user:
   1. If the user does not possess a role: stop processing the request and return an error to the user.
3. Query the database for a list of all resources.
4. Sanitize the data in the list of resources.
5. Return the list of resources to the user.

## Use Case 4: Commenting on a Resource

The user intends to comment on a resource. All users can comment on resources. These comments will be available as part of the resource when it is retrieved from the system.

### Targeted Release

Initial Release

### Precondition

The user possesses an authentication token. The user knows the ID of the resource they want to comment on.

### Postcondition

A new comment has been added to the target resource.

### Trigger

User sends a “Add Comment” request to the API.

### Process

1. Validate the authorization token with the authentication server:
   1. If authentication fails or no token is provided by the user: stop processing the request and return an error to the user.
2. Identify the system role possessed by the user.
   1. If the user has no role: stop processing the request and return an error to the user.
3. Sanitize and validate the inputs.
4. Search the database for a resource matching the target ID.
   1. If the resource is not found, stop processing the request and return an error to the user.
5. Create a new comment with the date, time, user ID of the current user, and provided comment text and add it to the target resource.
6. Notify the user of their request’s success.

## Use Case 5: Deleting a Comment

The user intends to delete a comment. Managers may delete any comment, but other users may only delete a comment if they created it.

### Targeted Release

Initial Release

### Precondition

The user possesses an authentication token. The user knows the ID of the comment they intend to delete. The user is a manager or created the target comment.

### Postcondition

The target comment has been removed from the locator.

### Trigger

User sends a “Delete Comment” request to the API.

### Process

1. Validate the authorization token with the authentication server:
   1. If authentication fails or no token is provided by the user: stop processing the request and return an error to the user.
2. Identify the system role possessed by the user.
   1. If the user is a contributor or commentor: only continue if the user ID matches the creator ID of the target comment.
   2. If the user has no role: stop processing the request and return an error to the user.
3. Sanitize and validate the inputs.
4. Search the database for a comment matching the target comment ID.
   1. If the comment exists:
      1. Delete the comment.
      2. Update the resource history log.
      3. Notify the user of their request’s success.
   2. If the comment does not exist: stop processing the request and return an error to the user.

## Use Case 6: Editing a Resource

The user intends to edit a single field of an existing resource in the database. They may edit resources they added themselves or, if they are a manager, resources created by other users. Note that the creator and creation date of a resource are not editable.

### Targeted Release

Upcoming Release

### Precondition

The user is either a manager or the creator of the target resource. The user possesses an authentication token. The user knows the ID of the targeted resource.

### Postcondition

The targeted resource has been updated.

### Trigger

User sends an “Edit Resource” request to the API.

### Process

1. Validate the authorization token with the authentication server:
   1. If authentication fails or no token is provided by the user: stop processing the request and return an error to the user.
2. Identify the system role possessed by the user.
   1. If the user is a contributor: only continue if the user ID of the current user matches the creator ID of the target resource.
   2. If the user is a commentor or possesses no role: stop processing the request and return an error to the user.
3. Sanitize and validate the inputs.
4. Search the database for the targeted resource by ID:
   1. If the resource does not exist, stop processing the request and return an error to the user.
5. Check that the targeted field can be edited:
   1. If field cannot be edited, stop processing the request and return an error to the user.
6. Update the targeted field with the given data.
7. Update the resource history log.
8. Notify the user of their request’s success.

## Use Case 7: Editing a Comment

The user intends to edit a comment. Users may only edit comments that they created.

### Targeted Release

Upcoming Release

### Precondition

The user possesses an authentication token. The user is a created the target comment. The user knows the ID of the target comment.

### Postcondition

A new resource has been added to the server.

### Trigger

User sends an “Edit Comment” request to the API.

### Process

1. Validate the authorization token with the authentication server:
   1. If authentication fails or the user does not provide a token: stop processing the request and return an error to the user.
2. Identify the system role possessed by the user.
   1. If the user has a role: only continue if the user ID of the user matches the creator ID of the target comment.
   2. If the user has no role: stop processing the request and return an error to the user.
3. Sanitize and validate the inputs.
4. Search for the comment matching the target comment ID in the database.
   1. If the comment does not exist: stop processing the request and return an error to the user.

## Use Case 8: Searching for a Resource by Tags

The user intends to obtain a list of all resources matching a tag. Users of any authority level may obtain this list.

### Targeted Release

Upcoming Release

### Precondition

The user possesses an authentication token.

### Postcondition

User has obtained a list of all resources matching the tag.

### Trigger

User sends a “Search” request with the tag parameter to the API.

### Process

1. Validate the authorization token with the authentication server:
   1. If authentication fails or the user provides no token: stop processing the request and return an error to the user.
2. Identify the system role possessed by the user.
   1. If the user has no role: stop processing the request and return an error to the user.
3. Sanitize and validate the inputs.
4. Query the database for all resources matching the tag.
5. Sanitize data in the output list of resources.
6. Return the list of resources.

## Use Case 9: Searching for a Resource by Creator

The user intends to search for a resource by creator. Users of any authority level may obtain this list.

### Targeted Release

Upcoming Release

### Precondition

The user possesses an authentication token. The user knows the creator ID that they are searching for.

### Postcondition

The user obtains a list of all resources created by the employee with the given user ID.

### Trigger

User sends a “Search” request with the creator ID parameter to the API.

### Process

1. Validate the authorization token with the authentication server:
   1. If authentication fails or no token is provided: stop processing the request and return an error to the user.
2. Identify the system role possessed by the user.
   1. If the user has no role: stop processing the request and return an error to the user.
3. Sanitize and validate the inputs.
4. Query the database for all resources matching the target user ID.
5. Sanitize data in the output list of resources.
6. Return the list of resources.

## Use Case 10: Searching for a Resource by Text

The user intends to search in the wiki by text content. Text content search will find all resources where the Title, URL, or description contains the search string.

### Targeted Release

Upcoming Release

### Precondition

The user possesses an authentication token.

### Postcondition

The user obtains a list of all resources matching the search string.

### Trigger

User sends a “Search” request with a text search string to the API.

### Process

1. Validate the authorization token with the authentication server:
   1. If authentication fails or the user provides no token: stop processing the request and return an error to the user.
2. Identify the system role possessed by the user.
   1. If the user has no role: stop processing the request and return an error to the user.
3. Sanitize and validate the inputs.
4. Run a text search on the database using the search string.
5. Sanitize the output list of resources.
6. Return the list of resources.

## Use Case 11: Requesting the history of a resource

The user intends to request the history of a resource. Only managers may request the history of a resource.

### Targeted Release

Upcoming Release

### Precondition

User possesses an authentication token. The user is a manager. The user knows the ID of the target resource.

### Postcondition

The user receives a history of the events that have happened to the target resource.

### Trigger

User sends a “Resource History” request to the API.

### Process

1. Validate authorization token with authentication server:
   1. If authentication fails or the user provides no token: stop processing the request and return an error to the user.
2. Identify the system role possessed by the user:
   1. Contributor, commentor, or no role: stop processing the request and return an error to the user.
3. Sanitize and validate the inputs.
4. Search the database for a resource history record matching the