Release 0.3

Description

The primary features in the third pilot release of DuraCloud are:

- Security
  - All DuraCloud applications now require authentication prior to performing write activities
  - Read activities on 'closed' spaces also require authentication, but 'open' spaces allow anonymous read access
  - See DuraCloud Security page for more information
- Sync Tool
  - Provides a command line utility for keeping DuraCloud content synchronized with the local file system
  - See Using the Sync Tool page for more information

Other improvements in the 0.3 release:

- Image Conversion Service
  - Adds an option to convert images to the (web standard) sRGB color space
  - Adds the capability to perform multiple conversions at once (providing the compute capacity is available) and provides more frequent activity feedback through the continual writing of the conversion output file
- DuraStore
  - Adds an option for users to provide MD5 checksum when adding content. This disables the in-transfer MD5 computation (providing improved performance) and compares the final MD5 computed by the storage provider with the user provided MD5.

For more details about specific changes in release 0.3, see the JIRA issue tracker.

Deliverables

1. Instance URLs
   - NYPL
     - https://nypl.duracloud.org/duradmin
     - https://nypl.duracloud.org/durastore
   - BHL
     - https://bhl.duracloud.org/duradmin
     - https://bhl.duracloud.org/durastore
   - WGBH
     - https://wgbh.duracloud.org/duradmin
     - https://wgbh.duracloud.org/durastore
2. REST API documentation
3. Store Client - Includes storeclient.jar and all dependent jars as well as Javadocs
4. Simple walk-through instructions for DurAdmin UI
5. JIRA Improvement and Bug tracker
   - Please add issues and bugs here as you find them

DuraCloud REST API (v0.3)

DuraCloud REST API methods:

- DuraStore
  - Get Stores
  - Get Spaces
  - Get Space
  - Get Space Metadata
  - Create Space
  - Set Space Metadata
  - Delete Space
  - Get Content
  - Get Content Metadata
  - Store Content
  - Set Content Metadata
  - Delete Content
- DuraService
  - Get Services
  - Get Service

- Get Deployed Service
- Get Deployed Service Properties
- Deploy Service
- Update Service Configuration
- UnDeploy Service

Release 0.3 includes security requirements for each of the methods below, see DuraCloud Security for more information

DuraStore

Purpose: DuraStore is the application through which DuraCloud manages storage. The DuraStore REST API provides access to storage by mediating the underlying storage provider APIs to allow access to multiple cloud storage options through a single API.

Store REST Methods

Get Stores

- Purpose: Provides a listing of available storage providers accounts (without credentials)
- Request: GET http://host:port/durastore/stores
- Parameters: None
- Response Code: 200 (on success)
- Response Body: XML similar to:

```
<storageProviderAccounts>
  <storageAcct isPrimary='true'>
    <id>1</id>
    <storageProviderType>AMAZON_S3</storageProviderType>
  </storageAcct>
  <storageAcct isPrimary="false">
    <id>2</id>
    <storageProviderType>RACKSPACE</storageProviderType>
  </storageAcct>
</storageProviderAccounts>
```

Space REST Methods

Get Spaces

- Purpose: Provides a listing of all of the spaces that a customer has created
- Request: GET http://host:port/durastore/spaces ? (storeID)
- Response Code: 200 (on success)
- Response Body: XML similar to:

```
<spaces>
  <space id="space1" />
  <space id="space2" />
</spaces>
```

Get Space

- Purpose: Provides a listing of the contents of a space along with space metadata
- Request: GET http://host:port/durastore/spaceID ? (storeID) (prefix) (maxResults) (marker)
  - storeID (optional) - ID of the content storage provider to query (default is primary store)
  - prefix (optional) - Only retrieve content ids with this prefix (default is all content ids)
  - maxResults (optional) - The maximum number of content IDs to return in the list (default is 1000)
  - marker (optional) - The content ID marking the last item in the previous set (default is the first set of ids)
- Response Code: 200 (on success)
- Response Body: XML similar to:

```
<space id="space1">
  <item>Image 1</item>
  <item>Image 2</item>
</space>
```

- Response Headers: All available space metadata, example:

```
  x-dura-meta-space-count=65
  x-dura-meta-space-access=OPEN
  x-dura-meta-space-created=Mon, 01 Jan 2000 08:00:00 EST
  x-dura-meta-custom-metadata=Custom Metadata Value
```

Get Space Metadata

- Purpose: Provides all space metadata
- Request: HEAD http://host:port/durastore/spaceID ? (storeID)
- Response Code: 200 (on success)
- Response Headers: Same as for Get space (above)

Create Space

- Purpose: Creates a new space
- Request: PUT http://host:port/durastore/spaceID ? (storeID)
- Request Headers: Metadata about the space, example:

```
  x-dura-meta-space-access=OPEN
  x-dura-meta-custom-metadata=Custom Metadata Value
```

- Response Code: 201 (on success)
- Response Headers: Location of the new space (i.e. the URL used to create the space), example:

```
  Location=http://myhost:8080/durastore/space1
```

Set Space Metadata

- Purpose: Updates the metadata associated with a space
- Request: POST http://host:port/durastore/spaceID ? (storeID)
- Request Headers: Same as Create space (above)
- Response Code: 200 (on success)
- Response Body: "Space $spaceID updated successfully" (on success)

Delete Space

- Purpose: Deletes a space
- Request: DELETE http://host:port/durastore/spaceID ? (storeID)
- Response Code: 200 (on success)
- Response Body: "Space $spaceID deleted successfully" (on success)

Content REST Methods

Get Content

- Purpose: Retrieves a piece of content along with its metadata
- Request: GET http://host:port/durastore/spaceID/contentID ? (storeID) (attachment)
    - if attachment param value is true, a Content-Disposition header is included with the response
- Response Code: 200 (on success)
- Response Body: The content stream
- Response Headers: All available content metadata, example:

```
Content-Type=text/plain
Content-Length=5732
Content-MD5=345670923478509747383920 2
ETag=345670923478509747383920 2
x-dura-meta-content-name=Testing Content
x-dura-meta-content-owner=JSmith
```

Get Content Metadata

- Purpose: Retrieves the metadata of a piece of content without the content itself
- Request: HEAD http://host:port/durastore/spaceID/contentID ? (storeID)
- Response Code: 200 (on success)
- Response Headers: Same as Get content (above)

Store Content

- Purpose: Adds a piece of content to the store
- Request: PUT http://host:port/durastore/spaceID/contentID ? (storeID)
- Request Body: Content to be added
- Request Headers: Metadata about the content, example:

```
Content-Type=text/plain
Content-MD5=4cd56e137a93a1accb43c5d32f4afffb
x-dura-meta-content-name=Testing Content
x-dura-meta-content-owner=JSmith
```

    - Note that when the optional Content-MD5 header is included, the final checksum of the stored file is compared against the MD5 value included in the header to ensure that the file was stored correctly. If the header is not included, an MD5 checksum is computed as the file is transferred to storage, and that value is used in the final comparison.
- Response Code: 201 (on success)
- Response Headers: Location of the new content (i.e. the URL used to create the content), example:

```
Location=http://myhost:8080/durastore/space1/content1
```

Set Content Metadata

- Purpose: Updates the metadata associated with a piece of content
- Request: POST http://host:port/durastore/spaceID/contentID ? (storeID)
- Request Headers: Same as Store content (above)
- Response Code: 200 (on success)
- Response Body: "Content $contentID updated successfully"

Delete Content

- Purpose: Removes a piece of content from the store
- Request: DELETE http://host:port/durastore/spaceID/contentID ? (storeID)
- Response Code: 200 (on success)
- Response Body: "Content $contentID deleted successfully"

DuraService

Purpose: DuraService is the application through which DuraCloud manages services. The DuraService REST API provides the means by which services available in the DuraCloud service repository are deployed, configured, and undeployed.

Service REST Methods

Get Services

- Purpose: Retrieves a listing of services, along with their configuration options
- Request: GET http://host:port/duraservice/services ? (show)
    - Parameter options for show (optional)
        1. available (default) - Includes only services which have not been deployed but are available for deployment
        2. deployed - Includes only services which have been deployed and started
- Response Code: 200 (on success)
- Response Body: XML list of services (see service config xsd)

Get Service

- Purpose: Retrieves information about a particular service including description, configuration options, and all deployments
- Request: GET http://host:port/duraservice/services/serviceID
- Response Code: 200 (on success)
- Response Body: XML service (see service config xsd)

Get Deployed Service

- Purpose: Retrieves information about a deployed service including description, configuration options, and a single deployment indicating the configuration options in use
- Request: GET http://host:port/duraservice/services/serviceID/deploymentID
- Response Code: 200 (on success)
- Response Body: XML service (see service config xsd)

Get Deployed Service Properties

- Purpose: Retrieves the runtime properties of a deployed service
- Request: GET http://host:port/duraservice/services/serviceID/deploymentID/properties
- Response Code: 200 (on success)
- Response Body: XML service (simple xml Map serialization)

Deploy Service

- Purpose: Deploys and starts an available service
- Request: PUT http://host:port/duraservice/services/serviceID ? (serviceHost)
    - Parameter value for serviceHost (optional) should indicate the services host on which the service should be deployed. Default is the primary customer host.
- Request Body: XML user configuration indicating the config selections for the service (see user config portion of service config xsd)
- Response Code: 201 (on success)
- Response Header: Location header indicates the URL at which information about the deployed service can be retrieved (the URL for a get deployed service call) which includes the deploymentID

Update Service Configuration

- Purpose: Updates the configuration of a deployed service
- Request: POST http://host:port/duraservice/services/serviceID/deploymentID
- Request Body: Updated XML user configuration indicating the config selections for the service (see user config portion of service config xsd)
- Response Code: 200 (on success)

UnDeploy Service

- Purpose: Stops and Undeploys a deployed service
- Request: DELETE http://host:port/duraservice/services/serviceID/deploymentID
- Response Code: 200 (on success)

Using the Sync Tool (v0.3)

Introduction

The Sync Tool is a utility which was created in order to provide a simple way to move files from a local file system to DuraCloud and subsequently keep the files in DuraCloud synchronized with those on the local system.

Download

Download the sync tool here.

How the Sync Tool Works

- When you run the Sync Tool for the first time, you must include DuraCloud connection information (host, port, username, password) as well as the space where you would like all of your files stored. You must also provide a list of directories which will be synced to DuraCloud and a directory for the Sync Tool to use for its own backups.
- When the Sync Tool starts up, it will look through all of the files in each of the local sync directories and add them to its internal queue for processing. Each of those files will then be written to your DuraCloud space. As this initial write is happening a listener is set up to watch for any file changes within each of the sync directories. As a change occurs (a file is added, updated, or deleted), that change is added to the queue, and the appropriate action is taken to make the DuraCloud space consistent with the local file (i.e. the file is either written to the space or deleted from the space.)
- You can stop the Sync Tool at any time by typing 'x' or 'exit' on the command line where it is running. It will stop all listeners, complete any file transfers that are in progress, and close down.
- When you restart the Sync Tool, if you point it at the same backup directory, it will pick up where it left off. While the Sync Tool is running, it is constantly writing backups of its internal queue, so it first reads the most current backup and begins processing the files there. It then scans the sync directories to see if there are any files which have been added or updated since the last backup, and it also pulls a list of files from the DuraCloud space and scans that list to see if any local files have been deleted. Any changes detected are added to the internal queue, and the Sync Tool continues to run as usual.

Operational notes

- Security
  - Version 0.3 of DuraCloud includes a requirement that all data must be transported over SSL. In order to connect to your DuraCloud instance, the Sync Tool needs to have access to the DuraCloud SSL certificate. See the page on DuraCloud Security for more information about including the DuraCloud certificate in your local truststore.
  - If you add the DuraCloud certificate to a truststore which is not picked up by your Java installation, you may need to specify its location on the command line. To do this, add the parameter

    ```
    -Djavax.net.ssl.trustStore=<path-to-truststore>
    ```

    prior to the -jar parameter when starting the Sync Tool. You will, of course, have to replace <path-to-truststore> with the path to the truststore in which the DuraCloud certificate is stored.

    - An error similar to this:

      ```
      Caused by: javax.net.ssl.SSLHandshakeException:
      sun.security.validator.ValidatorException:
      PKIX path building failed:
      sun.security.provider.certpath.SunCertPathBuilderException:
      unable to find valid certification path to requested target
      ```

      when attempting to run the Sync Tool indicates that the you'll need to include the explicit path to your truststore.

- Restarting
  - You can perform a restart of the Sync Tool by using the -c command line option to point to the Sync Tool configuration file, which is written into the backup directory (named synctool.config)
  - If you would like the Sync Tool to perform a clean start rather than a restart (i.e. you would like it to compare all files in the sync directories to DuraCloud) you will need to either point it to a new backup directory, or clear out the existing backup location.
  - The Sync Tool will perform a clean start (not a restart) if the list of sync directories is not the same as the previous run. This is to ensure that all files in all sync directories are processed properly.
- Collisions
  - The Sync Tool allows you to sync multiple local directories into a single space within DuraCloud. Because of this, there is the possibility of file naming collisions, where two local files resolve to the same DuraCloud ID. If this happens, one file will be overwritten by the other. There are a few ways to ensure that this does not occur:
    - Ensure that the top level files and directories within the set of sync directories do not have overlapping names.
    - Sync only a single directory to a space. You can run multiple copies of the Sync Tool, each over a single local directory, syncing to its own DuraCloud space.
- Backup Directory - these files and directories can be found in the backup directory (specified using the -b command line parameter)
  - Config Files
    - When the Sync Tool starts up, it writes the list of parameters and values provided by the user on startup to a file called

synctool.config in the backup directory. This file can be used to restart the Sync Tool, using the -c parameter to point to the file's location. You can also restart the Sync Tool by indicating the same set of options as used originally. The -c parameter is for convenience only and is not required in any circumstance. Note that this file is overwritten each time the Sync Tool is run with a different set of parameters, so you may choose to copy the file elsewhere (or give it a new name) if you would like to keep a copy of a particular configuration set.
    - You may also see a file named synctool.config.bak in the backup directory which is used to compare against the current config in order to determine if a restart is possible. In order for a restart to occur the list of sync directories (-d parameter) must be the same as the previous execution of the tool, and there must be at least one changed list backup (see below.)
- Changed List Directory
    - While the Sync Tool is running it is constantly updating the list of files which have been changed (when starting the first time, this includes all files in the directories that need to be synced). In order to allow the Sync Tool to restart after it has been stopped, this list of files is continually backed up into the *changedList* directory. There is no reason to edit these files, but you may choose to delete the *changedList* directory along with the config files mentioned above to ensure that the Sync Tool does not attempt to perform a restart.
- Logs Directory
    - Information about what the Sync Tool is doing while it is running can be found in the sync-tool.log file. It is a good idea to monitor this file for errors and warnings as this information is not printed to the console.
    - The duracloud.log file is useful for application debugging when the information in the sync-tool.log file is insufficient to understand a problem.

Prerequisites

- You must have Java version 6 or above installed on your local system. If Java is not installed, you will need to download and install it. To determine if the correct version of Java is installed, open a terminal or command prompt and enter

```
java -version
```

  The version displayed should be 1.6.0 or above. If running this command generates an error, Java is likely not installed.

- You must have downloaded the Sync Tool. It is available as a link near the top of this page.

Starting the Sync Tool

- To run the Sync Tool, open a terminal or command prompt and navigate to the directory where the Sync Tool is located
- To display the help for the Sync Tool, run

```
java -jar synctool-0.3-driver.jar
```

- When running the Sync Tool for the first time, you will need to use these options:

Short Option	Long Option	Argument Expected	Required	Description	Default Value (if optional)
-h	--host	Yes	Yes	The host address of the DuraCloud DuraStore application	
-p	--port	Yes	No	The port of the DuraCloud DuraStore application	8080
-s	--space	Yes	Yes	The ID of the DuraCloud space where content will be stored	
-u	--username	Yes	Yes	The username necessary to perform writes to DuraStore	
-w	--password	Yes	Yes	The password necessary to perform writes to DuraStore	
-d	--sync-dirs	Yes	Yes	A list of the directory paths to monitor and sync with DuraCloud. If multiple directories are included in this list, they should be separated by a space.	
-b	--backup-dir	Yes	Yes	The state of the sync tool is persisted to this directory	
-f	--poll-frequency	Yes	No	The time (in ms) to wait between each poll of the sync-dirs	10000 (10 seconds)
-t	--threads	Yes	No	The number of threads in the pool used to manage file transfers	3
-m	--max-file-size	Yes	No	The maximum size of a stored file in GB (value must be between 1 and 5), larger files will be split into pieces	1

-x	--sync-deletes	No	No	Indicates that deletes performed on files within the sync directories should also be performed on those files in DuraCloud; if this option is not included all deletes are ignored	Not set

- When the Sync Tool runs, it creates a backup of your configuration in the backup directory that you specify. When running the tool again, you can make use of this file to keep from having to re-enter all of the options specified on the initial run. In this case you need only a single option:

Short Option	Long Option	Argument Expected	Required	Description
-c	--config-file	Yes	Yes	Read configuration from this file (a file containing the most recently used configuration can be found in the backup-dir, named synctool.config)

- An example for running the Sync Tool

```
java -jar synctool-0.3-driver.jar -b C:\tools\synctool\backup -d C:\files\important -f 2000 -h
test.duracloud.org -p 443 -s important-dir-backup -t 5 -u myname -w mypassword
```

Runtime commands

- While the Sync Tool is running, these commands are available. Just type them on the command line where the tool is running.

Short Command	Long Command	Description
x	exit	Tells the Sync Tool to end its activity and close
c	config	Prints the configuration of the Sync Tool (the same information is printed at startup)
s	status	Prints the current status of the Sync Tool
l <Level>	N/A	Changes the log level to <Level> (may be any of DEBUG, INFO, WARN, ERROR)
h	help	Prints the runtime command help

DuraCloud Security (v0.3)

Overview

The security approach is divided into two distinct spheres of responsibility

1. Channel security (encryption)
2. Application security (AuthN / AuthZ)

The configuration of any given user compute instance will consist of an Apache HttpServer layered on top of Tomcat.

1. Apache HttpServer
    - All requests will come through Apache on port 443 (https) of the instance
    - The requests will internally be unencrypted, where encryption exists, and redirected to tomcat as open text
2. Tomcat
    - A defined set of resource endpoints will require AuthN and AuthZ
    - Spring-security is being leveraged to wire AuthN and AuthZ across relevant resources

Channel Security Implementation

1. Apache HttpServer is configured to require all requests to the three DuraCloud web applications (/duradmin, /durastore, and /duraservice) go over https.
2. Below are the https enforcement rules configured in Apache

```
###
# ensure 'duradmin' uses https
###

RewriteCond %{REQUEST_URI} /duradmin
RewriteCond %{SERVER_PORT} !^443$
RewriteRule ^(.*)$ https://%{SERVER_NAME}$1 [R=301,L]

###
# require https for 'durastore' & 'duraservice' for external requests
###

RewriteCond %{REQUEST_URI} ^(/durastore|/duraservice)
RewriteCond %{SERVER_PORT} !^443$
RewriteCond %{SERVER_NAME} !^localhost$
RewriteCond %{SERVER_NAME} !^127.0.0.1$
RewriteCond %{REMOTE_HOST} !^127.0.0.1$
RewriteCond ${local-ip-map:%{REMOTE_HOST}} !^localhost$
RewriteRule ^(.*)$ https://%{SERVER_NAME}$1 [R=301,L]
```

Security Certificates

1. When accessing /duradmin for the first time, the user will need to accept the self-signed DuraCloud security certificate.
2. When accessing DuraCloud via client-side java applications, such as the synctool, it is necessary to have the security cert available in your local truststore.
    a. Get the cert
        - Run the following application (binary | source)

```
java -jar install-cert.jar <host>
```

        - Note: where 'host' is something like 'nypl.duracloud.org'
        - Enter '1' to save the cert locally as '*jssecacerts*'
    b. Add cert to truststore
        - Depending on your local system, one (or both) of the below should allow the application to use the downloaded cert
            - Option A: Specify the cert location as a jvm variable

```
java -Djavax.net.ssl.trustStore=<location to jssecacerts> <your-application>
```

            - Option B: Add the cert (*jssecacerts*) to the truststore

```
cp jssecacerts <JAVA_HOME>/jre/lib/security
```

Application Security Implementation

The basic AuthN flow is as follows

1. User requests secured resource
2. If credentials not in request
    - response 401
3. Spring AuthenticationProvider performs AuthN
    a. AuthProvider asks UserDetailsService for GrantedAuthorities for given Principal
    b. notes
        i. DuraCloud provides custom UserDetailsService implementation to return UserDetails of requesting Principal
        ii. AbstractSecurityInterceptor permanently caches user AuthN decisions by default
4. Authentication object and "configuration attributes" are passed to AccessDecisionManager for AuthZ

Security Servlet Filters

DuraCloud leverages Spring's mechanism for wiring AuthN/Z into an application across servlet url patterns.
The following access rules are placed across the durastore and duraservice REST-APIs:

Store REST Methods

Action	Role
Get Stores	ROLE_USER
Get Spaces	ROLE_ANONYMOUS if space 'open', else ROLE_USER
Get Space	ROLE_ANONYMOUS if space 'open', else ROLE_USER
Get Space Metadata	ROLE_ANONYMOUS if space 'open', else ROLE_USER
Create Space	ROLE_USER
Set Space Metadata	ROLE_USER
Delete Space	ROLE_USER
Get Content	ROLE_ANONYMOUS if space 'open', else ROLE_USER
Get Content Metadata	ROLE_ANONYMOUS if space 'open', else ROLE_USER
Store Content	ROLE_USER
Set Content Metadata	ROLE_USER
Delete Content	ROLE_USER

Service REST Methods

Action	Role
Get Services	ROLE_USER
Get Service	ROLE_USER
Get Deployed Service	ROLE_USER
Get Deployed Service Properties	ROLE_USER
Deploy Service	ROLE_USER
Update Service Configuration	ROLE_USER
UnDeploy Service	ROLE_USER

Roles

The fixed set of users/roles listed below are provided in DuraCloud. Each role in the list below represents a super set of the privileges of those above it.

1. ROLE_ANONYMOUS
    - no username/password
2. ROLE_USER
    - user created by DuraCloud-account admin
3. ROLE_ADMIN
    - owner of DuraCloud-account
4. ROLE_SYSTEM
    - internal user for delegation requests
5. ROLE_ROOT
    - DuraSpace personnel

User Management

1. In order for the administrator of a DuraCloud account to manage new users, an initial user with ROLE_ADMIN privileges is provided at start-up:
    - username: admin

- password: changeme

Note: *Ultimately, the management of users will take place through the DuraCloud.org website (where users initially create accounts). User management is exposed in Duradmin in the upper-righthand corner of the console, but the usernames/passwords edited here are not persisted.*
This means that when the application is migrated to the next release, those details will need to be re-entered/re-created.

DurAdmin (v0.3)

Introduction to DurAdmin

The DuraCloud Administrator (DurAdmin) is a web interface designed to provide simple interaction with DuraCloud. DurAdmin sits on top of two DuraCloud applications, DuraStore and DuraService, which handle the content storage and the service deployment portions of DuraCloud, respectively. In this third point release, DurAdmin will be interacting with DuraStore (allowing you to view, update, and delete Spaces and Content) as well as with DuraService (allowing you to deploy several different Services).

Definitions

- **Space** - A top-level container in which content files are stored. Spaces are DuraCloud's single level of file organization. Spaces in DuraCloud are similar in nature to Buckets in Amazon S3 and Containers in the Rackspace Cloud.
- **Content** - A file stored in DuraCloud. Each content item in DuraCloud is stored in a space. The name of a content item can indicate a hierarchical structure (e.g. folder1/folder2/myfile.txt) but DuraCloud treats it simply as a file within a space.
- **Service** - An application or action that, when deployed, will allow for some operation to take place on stored content. Some services are dependent on other services to run properly.

Using the DurAdmin Interface

To use DurAdmin, start by opening a web browser and entering the address provided for your pilot organization. (This should be similar to https://organization.duracloud.org/duradmin/). You will be prompted to enter a username and password before being redirected to the DurAdmin main page. Once you successfully login, you will see a welcome page with a small amount of information.

User Management

In order for the administrator of a DuraCloud account to manage new users, an initial user with administrative privileges is provided on start-up (username: admin and password: changeme). Clicking on ***Admin*** located in the upper right will navigate you to the DurAdmin User Management page. From this page you can:

- Add a new user
  - To create a new user click on the ***Add*** button in the middle of the page. This will give you the ability to enter the ***Username*** and ***Password*** for the new user. Simply click ***Add*** again to confirm the new user's credentials.
- Remove a user
  - To remove a user click on the ***Remove*** button in the middle of the page. You will be prompted to confirm user deletion from a pop-up window after clicking the remove button; simply click ***OK*** to confirm or ***Cancel*** to return to the User Management screen. Please note that this operation is irreversible.
- Modify an existing user
  - To modify an existing user's credentials, click on the ***Modify*** button in the middle of the page. You will then have the option to ***Reset Password***. Simply type the new password and then click ***Modify*** for the change to take effect. To cancel user modification, simply click anywhere outside of the ***Modify*** box.

Spaces

Clicking on the ***Spaces*** tab near the top left will bring you to the "Spaces" page. This page lists the spaces that have been created in your account along with each space's metadata. Hovering over each space will give you the option to add a content item to the space or delete the space entirely as well as all associated content items. From this page you can:

- Create a new space
  - To create a space use the ***Add Space*** link in the upper left corner of the spaces page. This will give you the opportunity to enter the ***Space Id*** and to select the ***Space Access***. A space can be either OPEN or CLOSED; an OPEN space allows for direct anonymous read-access to the content in a space through the underlying storage provider; a CLOSED space allows only an authenticated user to access the content.
- View an existing space
  - View a space by clicking on the space name in the list.
- Add content to an existing space
  - Adding content can be accomplished either on the spaces page or in the space details page. To add content from the spaces page, click on the ***Add Content Item*** link in the space hover menu.
- Delete a space
  - Delete a space by clicking on the ***Remove*** link in the hover menu. Be warned, this is a recursive delete, all of the content items

contained in the space will be removed as part of this activity. *Deleting a space CANNOT be undone.*

From the "Spaces" page you can navigate to any one space by clicking on the space name in the list. This will take you to the space details page that will provide you with a listing of all of the content items stored in the space along with each content item's metadata. As on the spaces page, hovering over each content item will give you the options to view the details of the content item, download the content item, or delete the content item. From this page you can:

- Space Functions
    - Add/delete space metadata
        - From the space details bar on the left side of the screen, you can add metadata about the space by clicking on the **+** icon next to the word Metadata.
        - To delete metadata that has been added, simply hover over the metadata and then click the ***x*** icon.
    - Add/delete space tags
        - From the space details bar on the left side of the screen, you can also add tags about the space by clicking on the **+** icon next to the word Tags. To add more than one tag at a time, enter each word separated by the "|" symbol (pipe character).
        - To delete tags that have been added, simply hover over the tag that you wish to delete and then click the ***x*** icon.
    - Add a content item to the space
        - Add a content item by selecting the ***Add Content Item*** link in the upper right corner. This allows you to select a file and upload it into your space. By default the name of the file will be used as the file's ID, but you can override that value by filling in the ***Content ID*** field. By default the system will attempt to discover the correct MIME type for your file based on known extensions, but you can ensure the correct value by entering it in the ***MIME Type*** field.
    - Change the space access property
        - Change the access property of a space by selecting either the ***Open Space*** link (available if the space is currently closed) or the ***Close Space*** link (available if the space is currently open) in the upper right corner.
    - Delete the space
        - Delete the space by clicking on the ***Remove*** link in the top right corner. Please note that this is a recursive delete so all of the content items contained within this space will be removed as part of this activity. *Deleting a space CANNOT be undone.*
    - Refresh the space
        - Refresh the space's information as well as all the content item list by clicking on the ***Refresh*** link in the top right corner.
- Content Functions
    - View/edit a content item
        - View more details about a content item by selecting the name of the item or clicking ***Details*** in the hover menu.
        - Selecting the name of a content item will take you to the content details page that provides a detailed view of that item. All of the item's metadata is listed, some of which can be updated. From this page you can:
            - Add/delete content metadata
                - From the content details bar on the left side of the screen, you can add metadata about the content item by clicking on the **+** icon next to the word Metadata.
                - To delete metadata that has been added, simply hover over the metadata and then click the ***x*** icon.
            - Add/delete content tags
                - From the content details bar on the left side of the screen, you can also add tags about the content item by clicking the **+** icon next to the word Tags. To add more than one tag at a time, enter each word separated by the "|" symbol (pipe character).
                - To delete tags that have been added, simply hover over the tag that you wish to delete and then click the ***x*** icon.
            - Edit content MIME Type
                - Again, from the content details bar on the left side of the screen, you can edit the MIME type associated with the content item by including it in the text field and clicking ***Update***. You can update the MIME type as many times as you wish.
            - Add a content item
                - Add a new content item by selecting the ***Add Content Item*** link in the upper-right menu bar. See above for more details.
            - View a content item
                - When the J2K Service has been deployed, image files (files with a MIME type beginning with image/) will be viewable via the J2K viewer. A thumbnail of the image will appear in the box to the right of the content metadata. Clicking on either the thumbnail image or the ***View*** link in the upper-right menu bar will open the viewer. Note that images do not need to be in the J2K format to be viewed, but it will require more time for the thumbnail view and the viewer to display the image. Also note that the larger the file, the longer it will take for the image to render in both the thumbnail view and the viewer.
            - Download a content item
                - Download a file by selecting the ***Download*** link in the upper-right menu bar.
            - Delete a content item
                - Delete a content item by selecting the ***Remove*** link in the upper-right menu bar. *Deleting a content item CANNOT be undone.*
            - Refresh a content item
                - Refresh a content item's information by selecting the ***Refresh*** link in the upper-right menu bar.
    - Download a content item
        - Download a content item by selecting the ***Download*** link in the hover menu.
    - Delete a content item
        - Delete a content item by selecting the ***Remove*** link in the hover menu. *Deleting a content item CANNOT be undone.*

Services

Clicking on the ***Services*** tab near the top left will bring you to the "Services" page. By default, this page will list the services that are available to be deployed (if no services have been deployed) or the services that have been deployed (if at least one service has been deployed). To navigate between the two options, simply toggle the ***Deployed*** or ***Available*** tabs located in the top right of the screen. From this page you can:

- View/reconfigure/undeploy deployed services
    - The services that have been deployed are listed on the ***Deployed*** services page. To view more information about a deployed service, click the service name on the left side of the page. This will bring up a detailed description box to the right of the service.
    - Please note that certain services allow you to reconfigure the deployment. If this is an option, the ***Reconfigure*** link will appear on the right side of the page.
    - To refresh a service, simply click the ***Refresh*** button.
    - To undeploy a service, simply click the ***Undeploy*** button.
- View available services
    - The services that are available to be deployed as well as a brief description of the service are listed on the ***Available*** services page.
    - To deploy a service, simply click on the ***deploy*** link. Please note that for this release, the service can only be deployed to the primary service instance.

Storage Providers

To navigate between the available storage providers, use the dropdown menu in the top right of the screen. For this release, the default service provider is Amazon S3 and the only other storage provider option is Rackspace.

Available Services

- **Image Conversion Service** - The Image Conversion service provides a simple way to convert image files from one format to another. A space is selected from which image files will be read and converted to the chosen format. The converted image files will be stored in the destination space along with a file which details the results of the conversion process. Note that the ImageMagick service must be deployed prior to using the Image Conversion service.
- **ImageMagick Service** - The ImageMagick service deploys the ImageMagick application which allows other services to take advantage of its features.
- **J2K Service** - The J2K service deploys an instance of the Adore Djatoka web application which provides for serving and viewing JPEG2000 images.
- **Replication Service** - The Replication service provides a simple mechanism for synchronizing your content between two storage providers. A running replication service will listen for updates which occur in one store and duplicate those activities in another store.
    - For this release the replication service
        - will only start replicating after you turn on the service (nothing that exists prior to the service being deployed will be replicated)
        - will only replicate content upon initial upload
- **Web App Utility Service** - The Web App Utility service coordinates the (de)installation and startup/shutdown of Apache Tomcat instances that are created to run web application services that are deployed externally to the hosting OSGi container.