AdminStore Web Service Design

Version 1.0

# Description

AdminStore is Web Service to facilitate application and project administration functionality, user authentication and authorization.

# Scope

AdminStore is primary web service to encapsulate all functionality directly and indirectly related to application and project administration, user authentication and authorization. AdminStore Web Service will contain all user records, privileges and other user and application settings and properties. The following design principles are used to limit the scope of the AdminStore Web Service for the current version:

1. IIS service supports PUT and DELETE HTTP methods, alternatively X-HTTP-Method-Override header to be used in conjunction with POST method.

# Interface

## Endpoint

Service will be hosted in the path /svc/adminstore

All methods except for sign-in expect to receive header Token to identify user session. Token is used to authorize user for an action being requested.

## Methods

POST /standard-authentication  
- where {id} is string containing 32 alphanumerical characters of unique identifier (GUID) of the file information to retrieve

Method for downloading file information (FileId, StoredDate, FileName, ContentType, file size).

Returns file details.

Returns status **200 OK** if no issue is detected.  
Returns status **401 Unauthorized** if session cannot be found or user is not permitted the action requested.  
Returns status **403 Bad Request** if GUID is unrecognized.  
Returns status **406 Not Acceptable** if Token is missing or malformed.  
Returns status **404 Not Found** if file is not found.  
Returns status **500 Internal Server Error** if error:  
- database connection failed;  
- database connection established but table Files not found.

Example: HEAD /svc/adminstore/files/ee40d62d883d4eecb095f78883b69d63

POST /federated-authentication  
- where {dp} is base64-encoded string containing URI of the blueprint web application destination URI.

Method for processing SAML response from Identity Provider. The URI of this method with encoded destination URI should be included in SAML request.

Returns status **200 OK** if no issue is detected.  
Returns status **401 Unauthorized** if session cannot be found or user is not permitted the action requested.  
Returns status **403 Bad Request** if GUID is unrecognized  
Returns status **406 Not Acceptable** if Token is missing or malformed.  
Returns status **404 Not Found** if file is not found.  
Returns status **500 Internal Server Error** if error:  
- database connection failed;  
- database connection established but table Files not found.

Example: GET /svc/adminstore/files/ee40d62d883d4eecb095f78883b69d63

### POST /files

Method for uploading file into the application.

Returns string containing 32 alphanumerical characters of unique identifier (GUID) of the posted file.

Returns status **200 OK** if no issue is detected.  
Returns status **401 Unauthorized** if session cannot be found or user is not permitted the action requested.  
Returns status **406 Not Acceptable** if Token is missing or malformed.  
Returns status **500 Internal Server Error** if error:  
- database connection failed;  
- database connection established but table Files not found.

DELETE /files/{id}  
- where {id} is string containing 32 alphanumerical characters of unique identifier (GUID) of the file to delete

Method for removal of previously uploaded file.

Returns string containing 32 alphanumerical characters of unique identifier (GUID) of the deleted file.

Returns status **200 OK** if no issue is detected.  
Returns status **401 Unauthorized** if session cannot be found or user is not permitted the action requested.  
Returns status **403 Bad Request** if GUID is unrecognized  
Returns status **406 Not Acceptable** if Token is missing or malformed.  
Returns status **404 Not Found** if file is not found.  
Returns status **500 Internal Server Error** if error:  
- database connection failed;  
- database connection established but table Files not found.

Example: DELETE /svc/adminstore/files/ee40d62d883d4eecb095f78883b69d63

GET /status  
Method to return current status of AdminStore Web Service.

Returns status **200 OK** if no issue is detected.  
Returns status **401 Unauthorized** if session cannot be found or user is not permitted the action requested.  
Returns status **406 Not Acceptable** if Token is missing or malformed.  
Returns status **500 Internal Server Error** if error:  
- database connection failed;  
- database connection established but table Files not found.

# Technology

AdminStore Web Service will be implemented using ASP.NET Web API/C#. Database will be Microsoft SQL Server 2012.

# Behavior

## Methods

HEAD /files/{id}  
Method checks if database table contains the record identified by id provided as a parameter. If record is found then file record (Id, Stored, Name, ContentType) is returned as an object. If record is not found then attempt is made to retrieve FILESTREAM information from legacy file storage. Return value is cached.

GET /files/{id}  
Method checks if database table contains the record identified by id provided as a parameter. If record is found then file is returned as file stream. If record is not found then attempt is made to retrieve FILESTREAM from legacy file storage. Return value is cached.

### POST /files

Method creates new database table record using file stream provided as a parameter. File identifier is generated in the database using NEWSEQUENTIALID() and returned as a 32 alphanumerical character string. File name is passed as parameter / header and content type is taken from header.

DELETE /files/{id}  
Method checks if database table contains the record identified by id provided as a parameter. If record is found then the record is deleted asynchronously. Otherwise asynchronous attempt is made to delete the FILESTREAM from legacy file storage. Returns file identifier as it was supplied to allow for piping on the client side.

GET /status  
Method tries to execute SELECT COUNT(\*) FROM [FILES]; to test database connection.

# Storage

Records to be stored in AdminStore.Files (database AdminStore, table Files)

## Columns

FileId : uniqueidentifier, PK, default = NEWSEQUENTIALID()  
StoredTime : datetime  
FileName : varchar(256)  
FileType : varchar(64)  
FileContent : varbinary(max)