KFC Single Sign-On Portal

Automated Application Synchronization

Test Plan

CECS 491B, Sec 11

April 17, 2019

Krystal Leon

013986607

Revision History

Date	Version	Description
4/17/19	1.0	First draft.

Table of Contents

1. Introduction

This document is made in order to understand the methods used for testing the automated application synchronization feature of the KFC Single-Sign-On application portal. This feature includes the functionalities for individual applications to register to the portal, publish themselves into the portal, request a new API key, and delete themselves from the portal.

2. Scope

The following describes the methods used to test each functionality.

2.1 Unit Testing

The functions called from the Application Manager that validate the HTTP request entries will be tested.

2.1.1 Functions To Be Tested.

- *bool IsValidStringLength(string title, int length)*
  - Validates that the title input is not null and that the title length is less than the inputted length of characters. If the title meets these requirements, then True is returned. If not, then False is returned.
- *bool IsValidEmail(string email)*
  - Validates that the email input is not null and that it is a properly formatted email. If the email meets these requirements, the True is returned. If not, then False is returned.
- *bool IsValidUrl(string url, ref Uri urlResult)*
  - Validates that the url input is not null and that it is a properly formatted url address. If the url meets these requirements, then True is returned. If not, then False is returned.
- *bool IsValidExtension(Uri imageUrl, string ex)*
  - Validates that the url input is not null and that the path extension is the inputted file type, ex. If the url meets these requirements, then True is returned. If not, then False is returned.
- *bool IsValidDimensions(Uri imgUrl, int width, int height)*
  - Validates that the image url input is not null and that dimensions are no more than the inputted width and height. If the url meets these requirements, then True is returned. If not, then False is returned

2.1.2 Test Scenarios

- *IsValidStringLength_Pass_ReturnTrue()*
    - If a string is passed that is not null and less than the inputted length of characters, then return True.
- *IsValidStringLength_Fail_ReturnFalse()*
    - If a string is passed that is not null and greater than the inputted length of characters, then return False.
- *IsValidString_Fail_NullValueReturnsFalse()*
    - If a null string is passed, then return False.
- *IsValidEmail_Pass_ReturnTrue()*
    - If an email is passed that is not null and in a proper email format, then return True.
- *IsValidEmail_Fail_ReturnFalse()*
    - If an email is passed that is not null and not in a valid email format, then return False.
- *IsValidEmail_Fail_NullValueReturnsFalse()*
    - If a null email is passed, then return False.
- *IsValidUrl_Pass_ReturnTrue()*
    - If a url is passed that is not null and in a proper url format, then return True.
- *IsValidUrl_Fail_ReturnFalse()*
    - If a title is passed that is not null and not in a proper url format, then return False.
- *IsValidUrl_Fail_NullValueReturnsFalse()*
    - If a null url is passed, then return False.
- *IsValidImageExtension_Pass_ReturnTrue()*
    - If a url is passed that is not null and whose path is the inputted file extension, then return True.
- *IsValidImageExtension_Fail_ReturnFalse()*
    - If a url is passed that is not null and not the inputted file extension, then return False.
- *IsValidImageExtension_Fail_NullValueReturnsFalse()*
    - If a null url is passed, then return False.
- *IsValidDimensions_Pass_ReturnTrue()*
    - If an image url is passed that is not null and no more than the inputted width and height of pixels, then return True.
- *IsValidDimensions_Fail_ReturnFalse()*

    - If an image url is passed that is not null and greater than the inputted width and height of pixels, then return False.
- *IsValidDimensions_Fail_NullValueReturnsFalse()*
    - If a null image url is passed, then return False.

2.2 Integration Testing

The Application service functions and ApiKey service functions will be tested. These functions depend on the Application Repository, the ApiKey Repository, and the Database.

2.2.1 Application Services

2.2.1.1 Functions To Be Tested

- *Application CreateApplication(DatabaseContext _db, Application app)*
    - Creates a new Application entry in the Application table.
- *Application DeleteApplication(DatabaseContext _db, Guid Id)*
    - Deletes an existing Application entry in the Application table.
- *Application GetApplication(DatabaseContext _db, Guid Id)*
    - Retrieves an existing Application entry in the Application table.
- *Application GetApplication(DatabaseContext _db, string title, string email)*
    - Retrieves an existing Application entry in the Application table.
- *Application UpdateApplication(DatabaseContext _db, Application app)*
    - Updates an existing Application entry in the Application table.

2.2.1.2 Test Scenarios

- *CreateApplication_Pass_ReturnApp*
    - Creates a non-existing Application entry in the Application table and returns the new Application.
- *CreateApplication_Fail_ExistingAppShouldReturnNull*
    - Attempts to create an existing Application entry in the Application table. Entry is not created and null is returned.
- *CreateApplication_Fail_MissingFieldsShouldThrowException*
    - Attempts to create a non-existing Application entry in the Application table with missing required fields. Entry is not created, a DbEntityValidationException is thrown, and null is returned.
- *CreateApplication_Fail_NullShouldReturnNull*
    - Attempts to create a non-existing Application entry in the Application table with null values. Entry is not created and null is returned.

- *DeleteApplication_Pass_ReturnApp*
    - Deletes an existing Application entry in the Application table and returns the deleted Application.
- *DeleteApplication_Fail_NonExistingAppShouldReturnNull*
    - Attempts to delete a non-existing Application entry in the Application table. Application is not deleted and returns null.
- *UpdateApplication_Pass_ReturnApp*
    - Updates an existing Application entry in the Application table and returns the updated Application.
- *UpdateApplication_Fail_NonExistingAppShouldThrowException*
    - Attempts to update a non-existing Application entry in the Application table. Application is not updated throws and exception.
- *UpdateApplication_Fail_NullShouldReturnNullReferenceException*
    - Attempts to update an existing Application entry in the Application table with null values. Entry is not updated and null is returned.
- *GetApplicationById_Pass_ReturnApp*
    - Retrieves an existing Application entry in the Application table and returns the found Application.
- *GetApplicationById_Fail_NonExistingAppShouldReturnNull*
    - Attempts to retrieve a non-existing Application entry in the Application table. Application is not found and returns null.
- *GetApplicationByTitleEmail_Pass_ReturnApp*
    - Retrieves an existing Application entry in the Application table and returns the found Application.
- *GetApplicationByTitleEmail_Fail_NonExistingAppShouldReturnNull*
    - Attempts to retrieve a non-existing Application entry in the Application table. Application is not found and returns null.
- *GetApplicationByTitleEmail_Fail_NullValuesReturnNull*
    - Attempts to retrieve an existing Application entry in the Application table with null values. Application is not found and returns null.

2.2.1.3 Test Data

- *new Application()*
    - Id = Guid.NewGuid(),
    - Title = "KFC App",
    - LaunchUrl = "https://kfc.com",
    - Email = "kfc@email.com",
    - UserDeletionUrl = "https://kfc.com/delete",
    - LogoUrl = "https://kfc.com/logo.png",

    - Description = "A KFC app"
    - SharedSecretKey = Guid.NewGuid().ToString("N")
    - HealthCheckUrl = "https://kfc.com/health

2.2.2 ApiKey Services

2.2.2.1 Functions To Be Tested

- *ApiKey CreateKey(DatabaseContext _db, ApiKey key)*
    - Creates a new ApiKey entry in the ApiKey table.
- *ApiKey DeleteKey(DatabaseContext _db, Guid Id)*
    - Deletes an existing ApiKey entry in the ApiKey table.
- *ApiKey GetKey(DatabaseContext _db, Guid Id)*
    - Retrieves an existing ApiKey entry in the ApiKey table.
- *ApiKey GetKey(DatabaseContext _db, string key)*
    - Retrieves an existing ApiKey entry in the ApiKey table.
- *ApiKey UpdateKey(DatabaseContext _db, ApiKey key)*
    - Updates an existing ApiKey entry in the ApiKey table.

2.2.2.2 Test Scenarios

- *CreateKey_Pass_ReturnKey*
    - Creates a non-existing ApiKey entry in the ApiKey table and returns the new ApiKey .
- *CreateKey_Fail_ExistingKeyShouldReturnNull*
    - Attempts to create an existing AppliApiKey cation entry in the ApiKey table. Entry is not created and null is returned.
- *CreateKey_Fail_NullValuesShouldReturnNullReferenceException*
    - Attempts to create a non-existing ApiKey entry in the ApiKey table with null values. Entry is not created and NullReferenceException is thrown.
- *DeleteKey_Pass_ReturnKey*
    - Deletes an existing ApiKey entry in the ApiKey table and returns the deleted ApiKey .
- *DeleteKey_Fail_NonExistingKeyShouldReturnNull*
    - Attempts to delete a non-existing ApiKey entry in the ApiKey table. Application is not deleted and returns null.
- *UpdateKey_Pass_ReturnKey*
    - Updates an existing ApiKey entry in the ApiKey table and returns the updated ApiKey .
- *UpdateKey_Fail_NonExistingKeyShouldThrowException*

    - Attempts to update a non-existing ApiKey entry in the ApiKey table. Application is not updated and an exception is thrown.
- *UpdateKey_Fail_NullShouldReturnNull*
    - Attempts to update an existing ApiKey entry in the ApiKey table with null values. Entry is not updated and null is returned.
- *GetApiKeyById_Pass_ReturnKey*
    - Retrieves an existing ApiKey entry in the ApiKey table and returns the found ApiKey .
- *GetApiKeyById_Fail_NonExistingKeyShouldReturnNull*
    - Attempts to retrieve a non-existing ApiKey entry in the Application table. ApiKey is not found and returns null.
- *GetApiKeyByKey_Pass_ReturnKey*
    - Retrieves an existing ApiKey entry in the ApiKey table and returns the found ApiKey .
- *GetApiKeyByKey_Fail_NonExistingKeyShouldReturnNull*
    - Attempts to retrieve a non-existing ApiKey entry in the ApiKey table. ApiKey is not found and returns null.
- *GetApiKeyByKey_Fail_NullValuesReturnNull*
    - Attempts to retrieve an existing ApiKey entry in the ApiKey table with null values. ApiKey is not found and returns null.

2.2.2.3 Test Data

- *new ApiKey()*
    - Id = Guid.NewGuid(),
    - Key = Guid.NewGuid().ToString("N"),
    - ApplicationId = app.Id,
    - IsUsed = false

2.3 End to End Testing

Puppeteer was used to simulate the user flow of the application registration, api key generation, and application deletion functionalities. Screenshots are made throughout to document the flow.

2.3.1 App Registration Test Scenarios

- *Valid Registration()*
    - All valid inputs should successfully register an application and return a success message, and api key, a shared secret key, and the application id.
- *Reject submission with empty values*
    - Trying to register with blank fields should give an error.
- *Reject submission with invalid email*
    - Trying to register with an invalid email format should give an error that prevents submission.
- *Reject submission with invalid launch url*
    - Trying to register with an invalid launch url format should give an error.
- *Reject submission with invalid health check url*
    - Trying to register with an invalid health check url format should give an error.
- *Reject submission with invalid user deletion url*
    - Trying to register with an invalid user deletion url format should give an error.

2.3.2 App Deletion Test Scenarios

- *Valid Deletion()*
    - All valid inputs should successfully delete an application and return a success message.
- *Reject submission with empty values*
    - Trying to delete with blank fields should give an error.
- *Reject submission with invalid email*
    - Trying to delete with an invalid email format should give an error that prevents submission.
- *Reject values of a non-existing application*
    - Trying to delete with the title and email of a non existing application should give an error.

2.3.3 Key Generation Test Scenarios

- *Valid Key Generation*
    - All valid inputs should successfully delete an application and return a success message.
- *Reject submission with empty values*
    - Trying to delete with blank fields should give an error.
- *Reject submission with invalid email*
    - Trying to delete with an invalid email format should give an error that prevents submission.
- *Reject values of a non-existing application*
    - Trying to delete with the title and email of a non existing application should give an error.

2.3.4 Test Data

- *appTitle* = 'My Application';
- *appEmail* = 'app@email.com';
- *appLaunchUrl* = 'https://app.com';
- *appDeleteUrl* = 'https://app.com/delete';
- *appHealthCheckUrl* = 'https://app.com/health';

2.4 Performance Testing

Not Available.

2.5 Penetration Testing

Not Available.