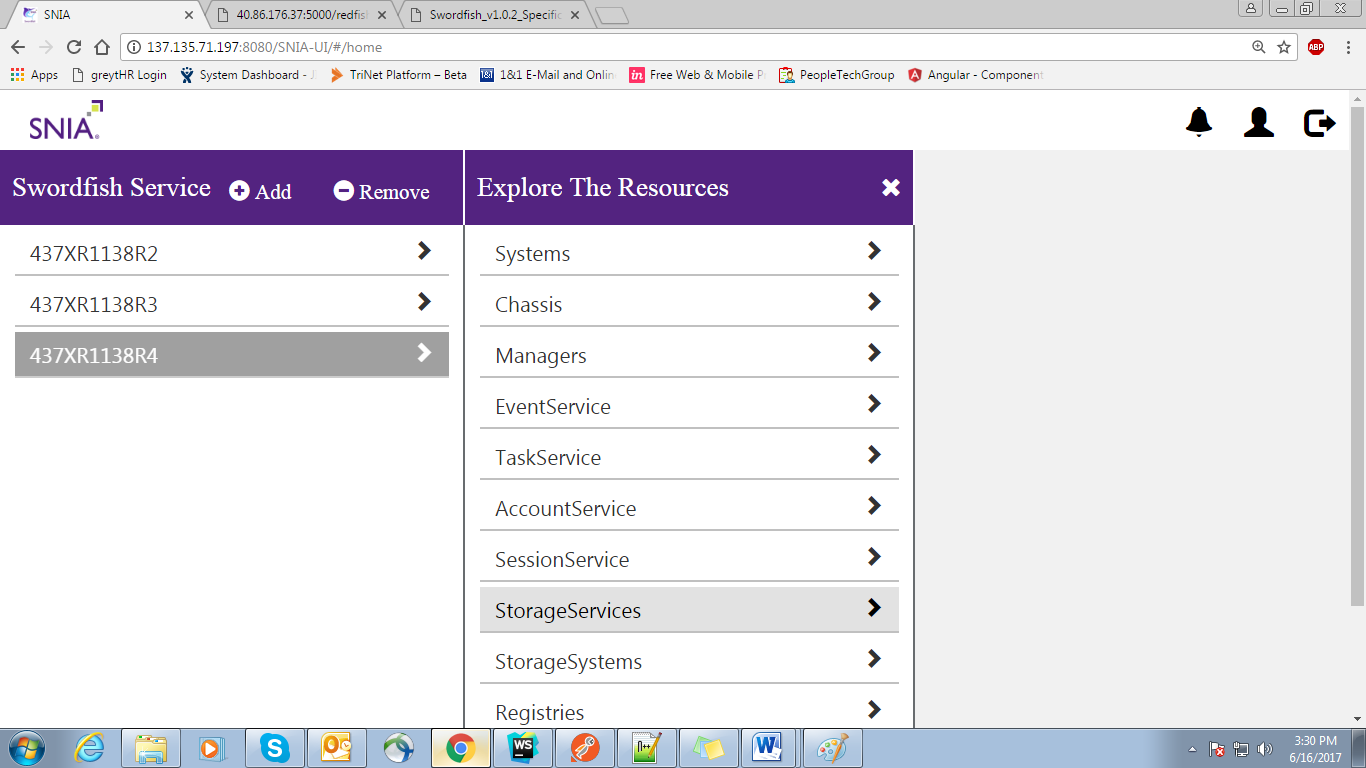
**SNIA UI Functionality**

**1. View**

Each Service has several Endpoints, which further may have endpoints, OData and properties.

The Following Example illustrates the Service(437XR1138R3) and it’s endpoints shown below “Explore The Resources “ blade.

UI View:



Emulator View :

{

"@odata.type": "#ServiceRoot.1.0.0.ServiceRoot",

"ServiceVersion": "1.0.0",

"Name": "Root Service",

"Links": {

"Systems": {

"@odata.id": "/redfish/v1/Systems"

},

"Chassis": {

"@odata.id": "/redfish/v1/Chassis"

},

"Managers": {

"@odata.id": "/redfish/v1/Managers"

},

"EventService": {

"@odata.id": "/redfish/v1/EventService"

},

"TaskService": {

"@odata.id": "/redfish/v1/TaskService"

},

"AccountService": {

"@odata.id": "/redfish/v1/AccountService"

},

"SessionService": {

"@odata.id": "/redfish/v1/SessionService"

},

"StorageServices": {

"@odata.id": "/redfish/v1/StorageServices"

},

"StorageSystems": {

"@odata.id": "/redfish/v1/StorageSystems"

},

"Registries": {

"@odata.id": "/redfish/v1/Registries"

}

},

"UUID": "018ee5b4-e55f-48f6-a55f-1343ee7bba78",

"@odata.id": "/redfish/v1/",

"@odata.context": "/redfish/v1/$metadata#ServiceRoot",

"Id": "RootService"

}

The @odata.id indicates the child endpoint to the existing service.

**StorageServices (**/redfish/v1/StorageServices**):**

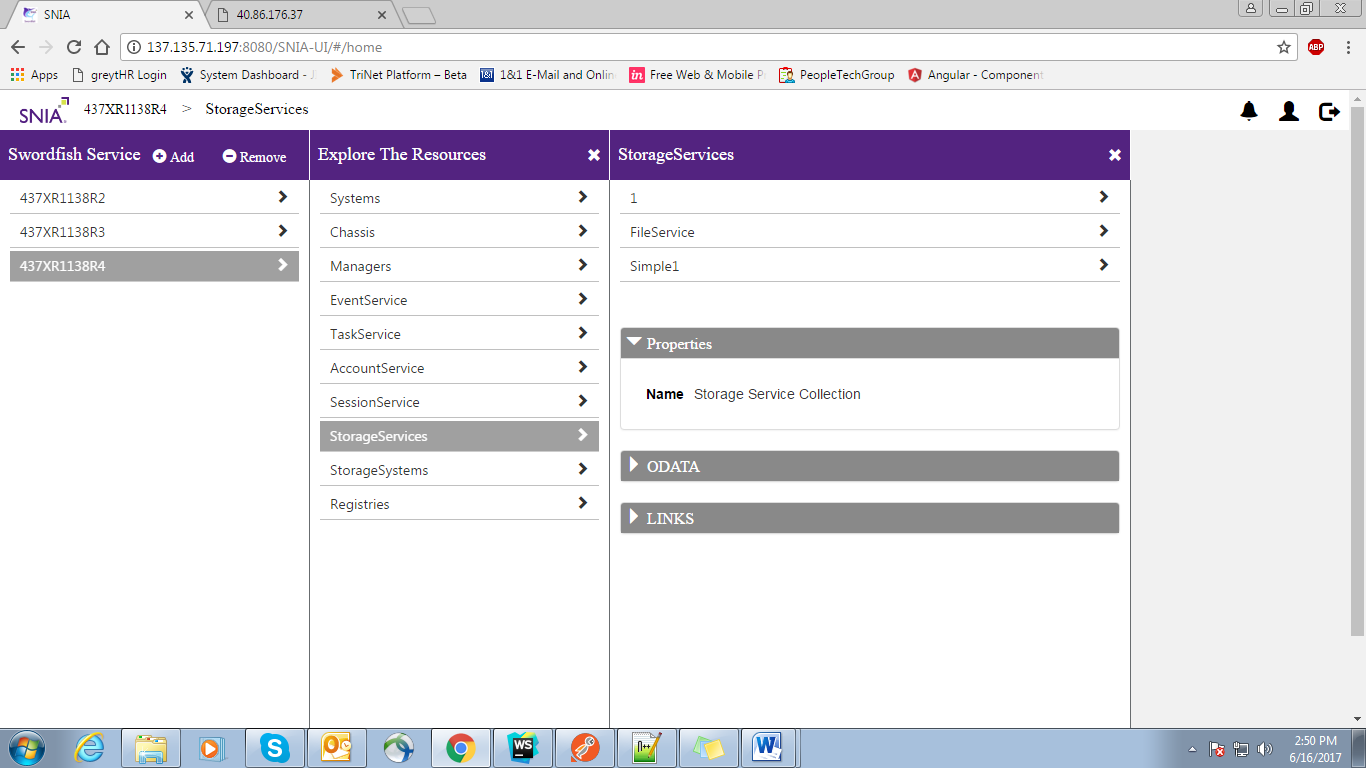
Each StorageService resource represents the resources and behaviors supported by that storage service.

It also provides the client an efficient means to search across all StorageService resources, regardless of which ComputerSystem is supporting the service

The Storage Services has

1. Properties that describes about the Basic Information about Storage Systems such as Id, Name etc.
2. OData explains about the Context and type of the Endpoint.
3. Links specifies the child endpoints for the existing Endpoint.

UI View:



Emulator View:

{

"@Redfish.Copyright": "Copyright 2015-2016 SNIA. All rights reserved.",

"@odata.context": "/redfish/v1/$metadata#StorageService.StorageService",

"@odata.id": "/redfish/v1/StorageSystems/Simple/StorageServices",

"@odata.type": "#StorageServiceCollection.1.0.0.StorageServiceCollection",

"Members": [

{

"@odata.id": "/redfish/v1/StorageServices/1"

},

{

"@odata.id": "/redfish/v1/StorageServices/FileService"

},

{

"@odata.id": "/redfish/v1/StorageServices/Simple1"

}

],

"Members@odata.count": 3,

"Name": "Storage Service Collection"

}

The @odata.id indicates the child endpoint to the existing Endpoint

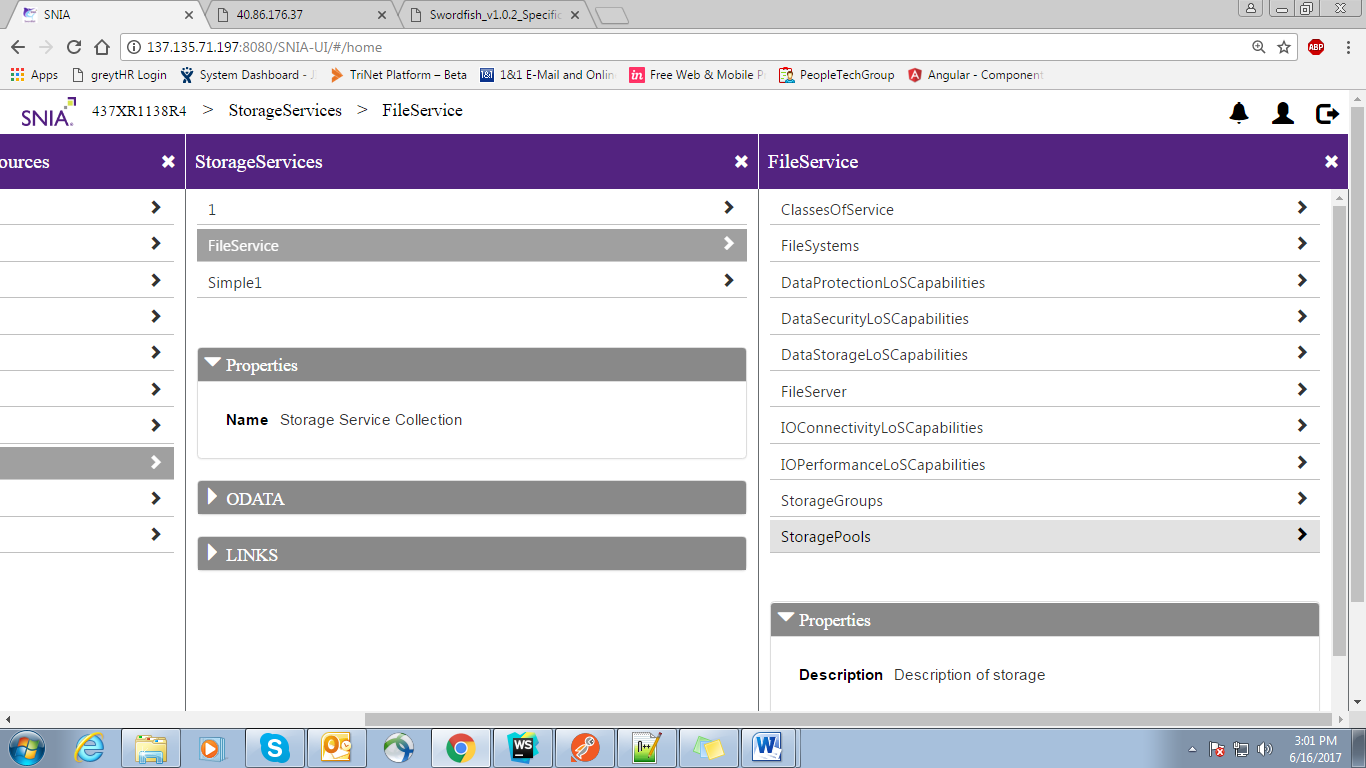
In the Above example the StorageServices has 3 Endpoints

1. 1
2. FileService
3. Simple1

**FileService:**

FileService is an endpoint which act as a child to StorageServices.

UI View:



Emulator view:

{

"@Redfish.Copyright": "Copyright 2014-2016 SNIA. All rights reserved.",

"@odata.context": "/redfish/v1/$metadata#StorageService.StorageService",

"@odata.id": "/redfish/v1/StorageServices/FileService",

"@odata.type": "#StorageService.1.0.0.StorageService",

"ClassesOfService": {

"@odata.id": "/redfish/v1/StorageServices/FileService/ClassesOfService"

},

"Description": "Description of storage",

"FileSystems": {

"@odata.id": "/redfish/v1/StorageServices/FileService/FileSystems"

},

"Id": "1",

"Links": {

"DataProtectionLoSCapabilities": {

"@odata.id": "/redfish/v1/StorageServices/FileService/DataProtectionLoSCapabilities"

},

"DataSecurityLoSCapabilities": {

"@odata.id": "/redfish/v1/StorageServices/FileService/DataSecurityLoSCapabilities"

},

"DataStorageLoSCapabilities": {

"@odata.id": "/redfish/v1/StorageServices/FileService/DataStorageLoSCapabilities"

},

"HostingSystem": {

"@odata.id": "/redfish/v1/StorageSystems/FileServer"

},

"IOConnectivityLoSCapabilities": {

"@odata.id": "/redfish/v1/StorageServices/FileService/IOConnectivityLoSCapabilities"

},

"IOPerformanceLoSCapabilities": {

"@odata.id": "/redfish/v1/StorageServices/FileService/IOPerformanceLoSCapabilities"

}

},

"Name": "My Storage Service",

"Oem": {},

"Status": {

"Health": "OK",

"State": "Enabled"

},

"StorageGroups": {

"@odata.id": "/redfish/v1/StorageServices/FileService/StorageGroups"

},

"StoragePools": {

"@odata.id": "/redfish/v1/StorageServices/FileService/StoragePools"

}

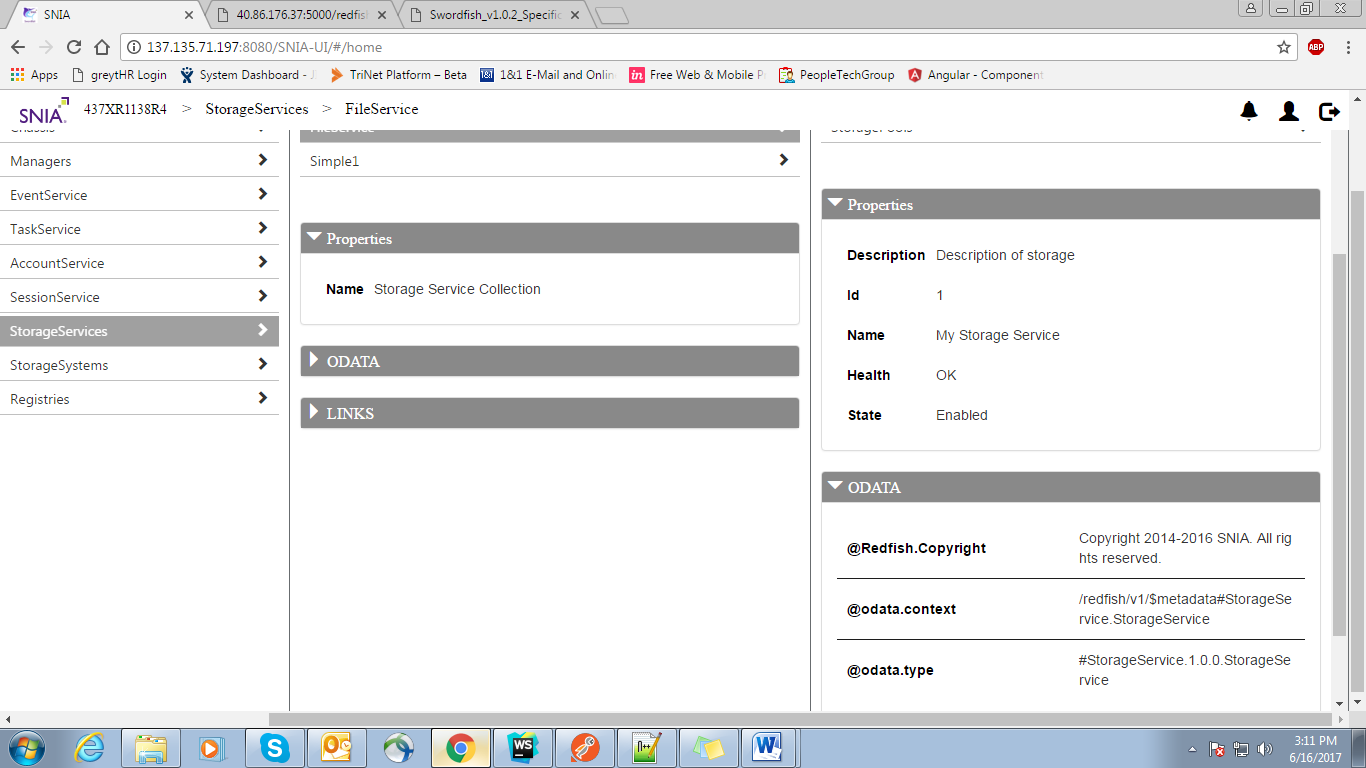
}

The @odata.id indicates the child endpoint to the existing Endpoint

FileService provides a path to know about the system by allowing to access it’s Child endpoints

1. ClassesOfService
2. FileSystems
3. DataProtectionLoSCapabilities
4. DataSecurityLoSCapabilities
5. DataStorageLoSCapabilities
6. FileServer
7. IOConnectivityLoSCapabilities
8. IOPerformanceLoSCapabilities
9. StorageGroups
10. StoragePools

The Basic information regarding FileService can be viewed in the Properties Section and additional information can be viewed in the OData Section.



**ClassesOfService (**"/redfish/v1/StorageServices/FileService/ClassesOfService"**) :**

A reference to a resource collection that specifies the set supported ClassOfService resources.

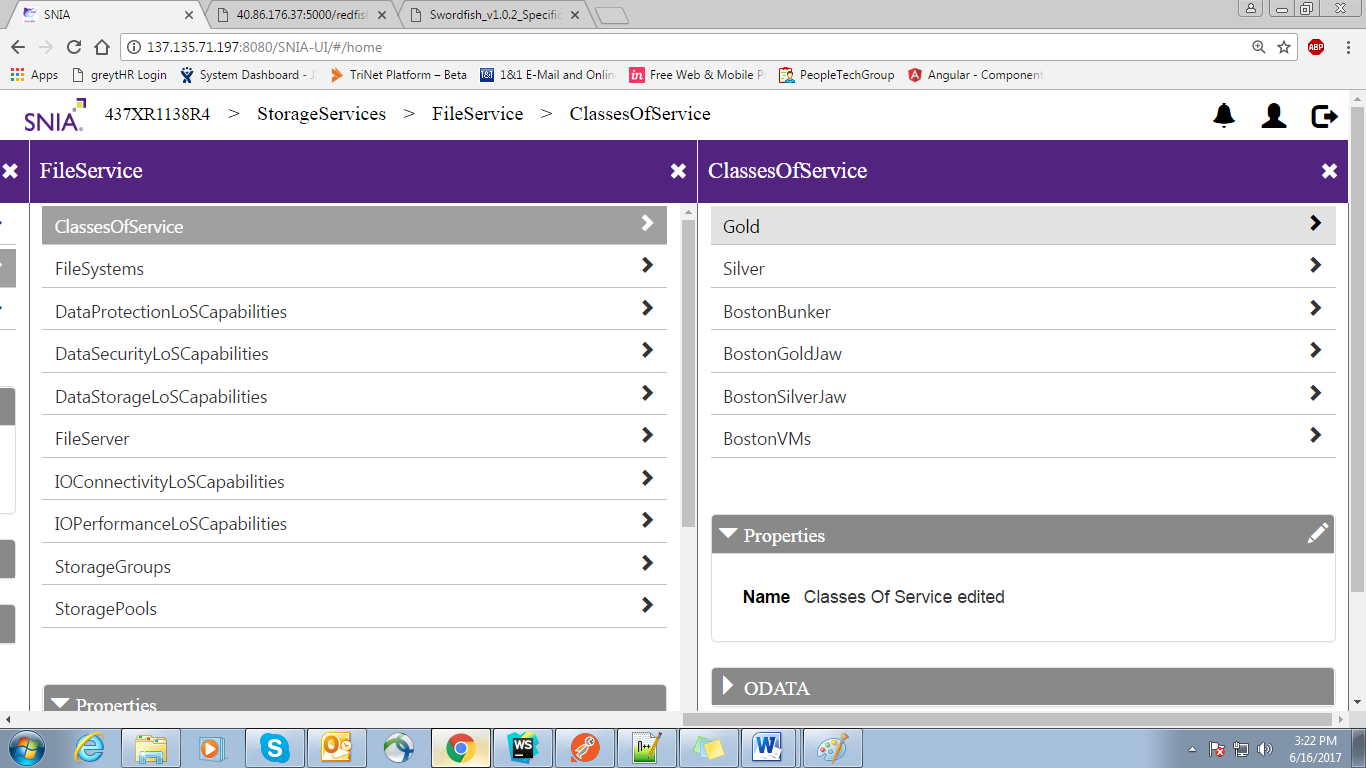
The value of each enty in the array shall reference a ClassOfService supported by this service.

It allows to edit the existing properties Values.

This Collection exhibits the further in depth information about the services and consists the following endpoints as the children

1. Gold
2. Silver
3. BostonBunker
4. BostonGoldJaw
5. BostonSilverJaw
6. BostonVMs

UI-View:



Emulator View:

{

"Members": [

{

"@odata.id": "/redfish/v1/StorageServices/FileService/ClassesOfService/Gold"

},

{

"@odata.id": "/redfish/v1/StorageServices/FileService/ClassesOfService/Silver"

},

{

"@odata.id": "/redfish/v1/StorageServices/FileService/ClassesOfService/BostonBunker"

},

{

"@odata.id": "/redfish/v1/StorageServices/FileService/ClassesOfService/BostonGoldJaw"

},

{

"@odata.id": "/redfish/v1/StorageServices/FileService/ClassesOfService/BostonSilverJaw"

},

{

"@odata.id": "/redfish/v1/StorageServices/FileService/ClassesOfService/BostonVMs"

}

],

"Name": "Classes Of Service edited",

"Permissions": [

{

"Read": "True"

},

{

"Write": "True"

}

]

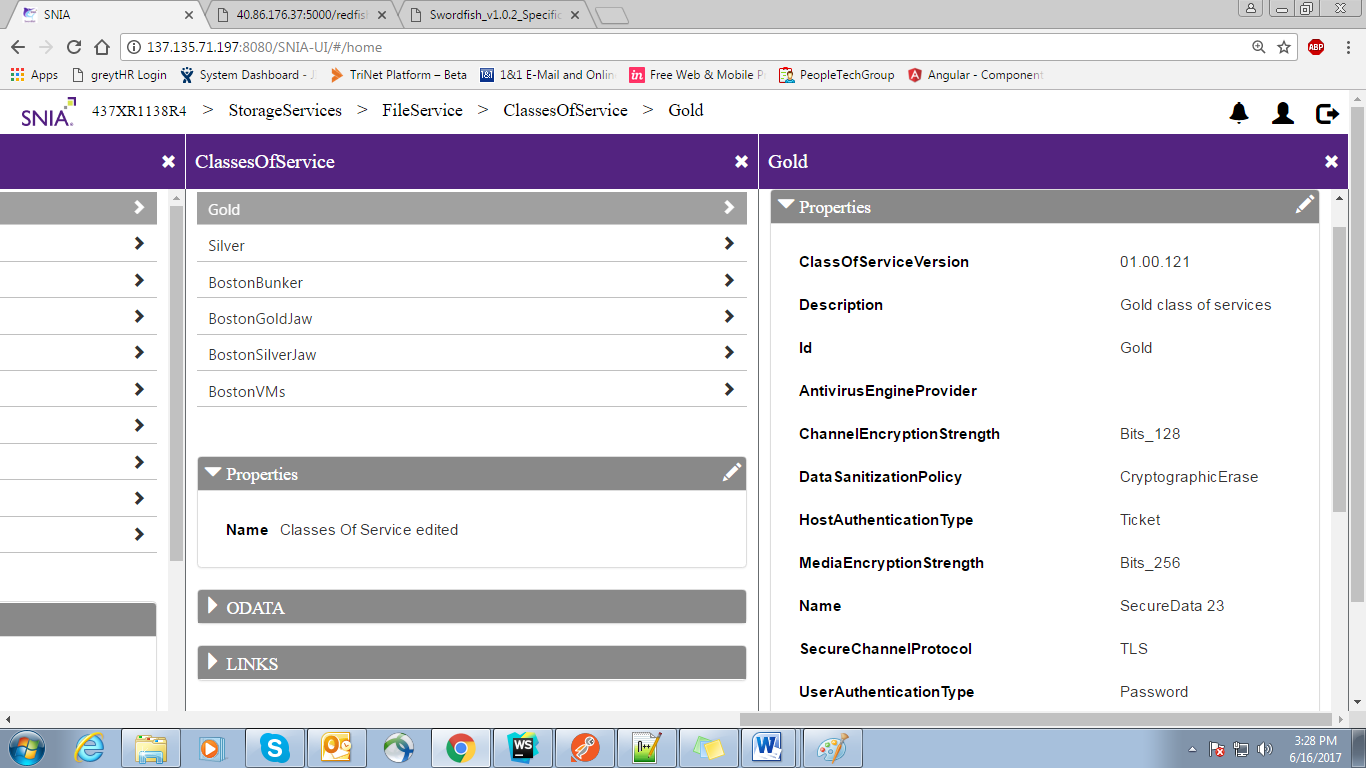
}

Based On the Write Permission of EndPoint , client can modify the properties of the Endpoint as well as the it’s children Endpoint values.

**Gold**:

Illustrates the basic properties of the system such as Version, Description ,Id etc

UI-View:



The edit property allows to modify the property values.

Emulator View:

{

"ClassOfServiceVersion": "01.00.121",

"Description": "Gold class of services",

"Id": "Gold",

"LinesOfService": {

"DataProtectionLineOfService": [],

"DataSecurityLineOfService": {

"AntivirusEngineProvider": null,

"AntivirusScanPolicies": [],

"ChannelEncryptionStrength": "Bits\_128",

"DataSanitizationPolicy": "CryptographicErase",

"HostAuthenticationType": "Ticket",

"MediaEncryptionStrength": "Bits\_256",

"Name": "SecureData 23",

"SecureChannelProtocol": "TLS",

"UserAuthenticationType": "Password"

},

"DataStorageLineOfService": {

"Name": "HA-Thin for AntiVirus",

"ProvisioningPolicy": "Thin",

"RecoveryTimeObjective": 0,

"SpaceEfficient": true

},

"IOConnectivityLineOfService": {

"AccessProtocol": "FC",

"MaxSupportedIoOperationsPerSecond": null,

"Name": "FiberChannel"

},

"IOPerformanceLineOfService": {

"AverageIoOperationLatencyMicroseconds": 5000,

"IOWorkload": {

"Name": "Duplicon:OLTP"

},

"IoOperationsPerSecondIsLimitedBoolean": "false",

"MaxIoOperationsPerSecondPerTerabyte": 133,

"Name": "Heavy-OLTP-HDD",

"SamplePeriod": "PT1M"

}

},

"Name": "Gold changed"

}

**2. EDIT (For an EndPoint):**

Based on the Write Permissions from the Emulator, Endpoints property values can be modified

**Emulator View:**

"Permissions": [

{

"Read": "True"

},

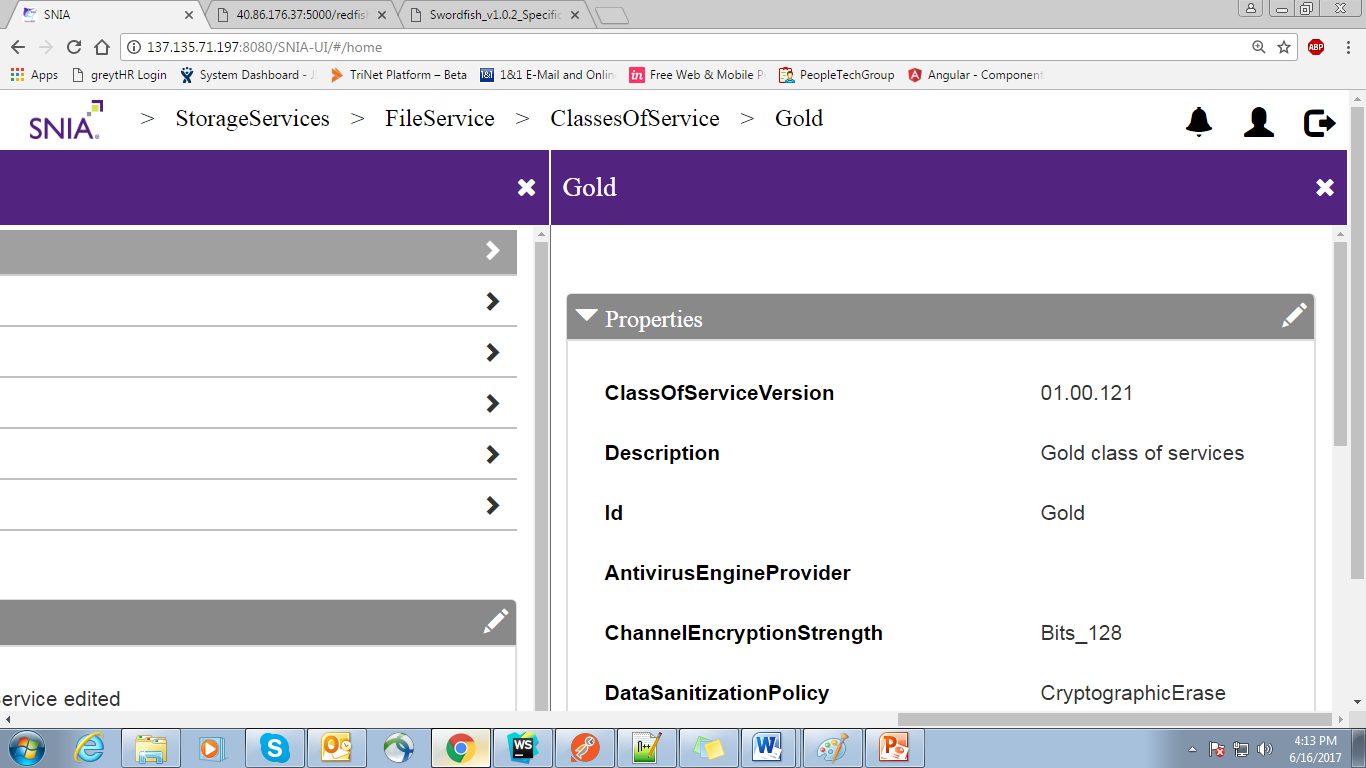
{

"Write": "True"

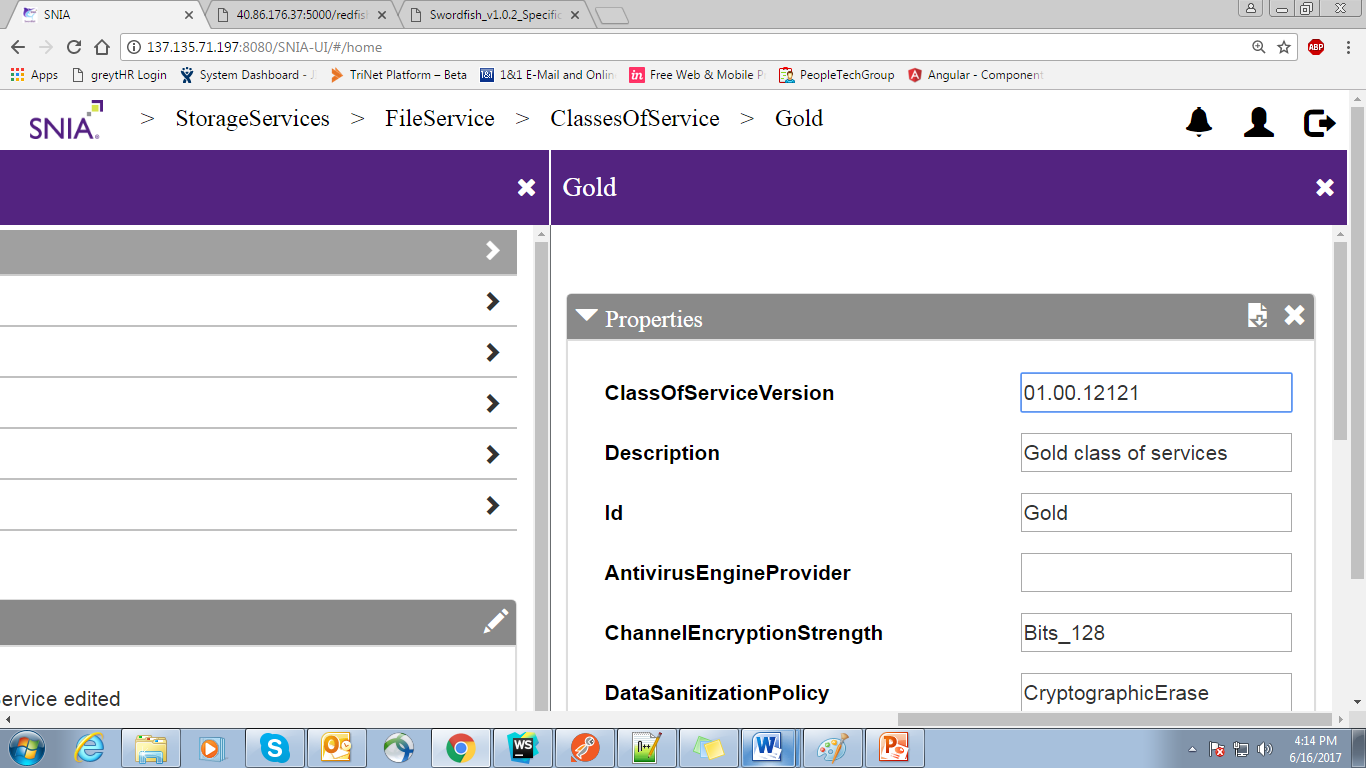
}

]

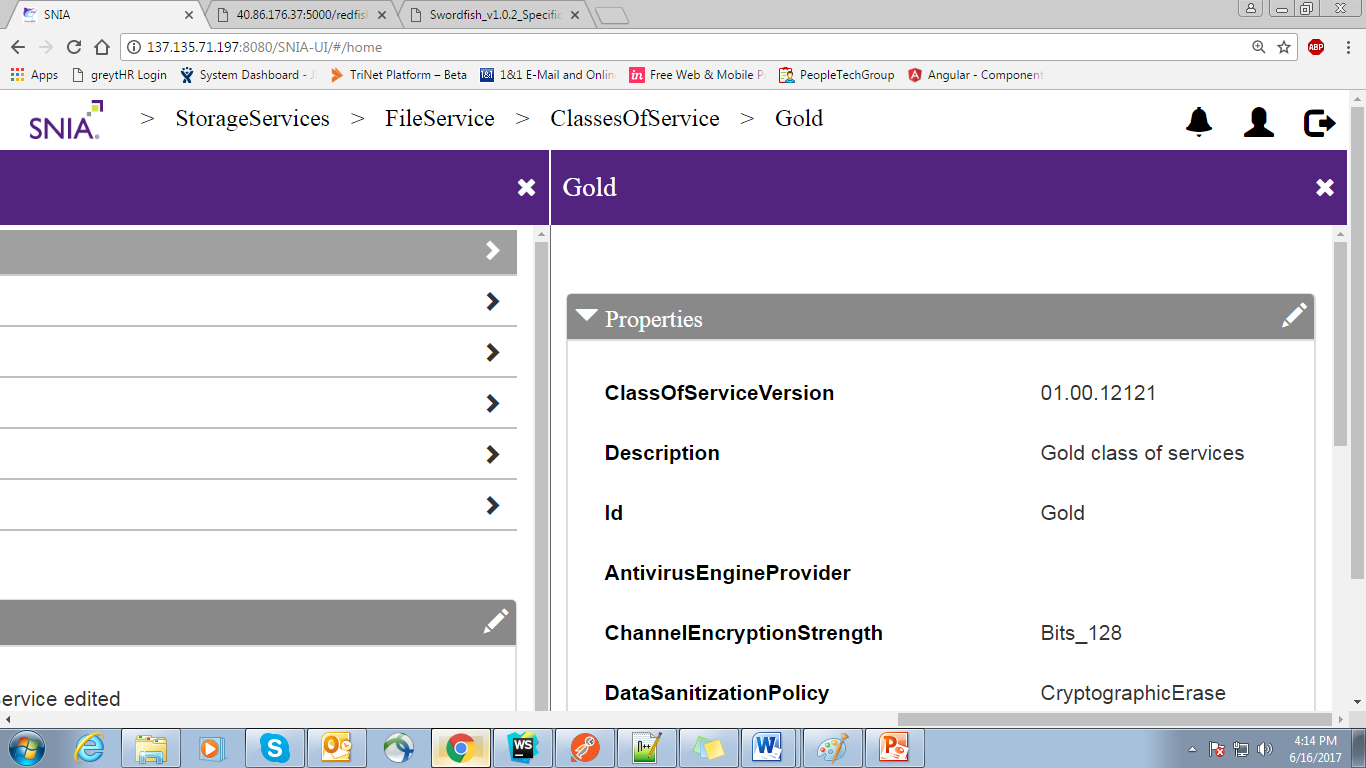
UI-view:



On Edit :



On Save



In the Emulator After Edit functionality :

{

"ClassOfServiceVersion": "01.00.12121",

"Description": "Gold class of services",

"Id": "Gold",

"LinesOfService": {

"DataProtectionLineOfService": [],

"DataSecurityLineOfService": {

"AntivirusEngineProvider": null,

"AntivirusScanPolicies": [],

"ChannelEncryptionStrength": "Bits\_128",

"DataSanitizationPolicy": "CryptographicErase",

"HostAuthenticationType": "Ticket",

"MediaEncryptionStrength": "Bits\_256",

"Name": "SecureData 23",

"SecureChannelProtocol": "TLS",

"UserAuthenticationType": "Password"

},

"DataStorageLineOfService": {

"Name": "HA-Thin for AntiVirus",

"ProvisioningPolicy": "Thin",

"RecoveryTimeObjective": 0,

"SpaceEfficient": true

},

"IOConnectivityLineOfService": {

"AccessProtocol": "FC",

"MaxSupportedIoOperationsPerSecond": null,

"Name": "FiberChannel"

},

"IOPerformanceLineOfService": {

"AverageIoOperationLatencyMicroseconds": 5000,

"IOWorkload": {

"Name": "Duplicon:OLTP"

},

"IoOperationsPerSecondIsLimitedBoolean": "false",

"MaxIoOperationsPerSecondPerTerabyte": 133,

"Name": "Heavy-OLTP-HDD",

"SamplePeriod": "PT1M"

}

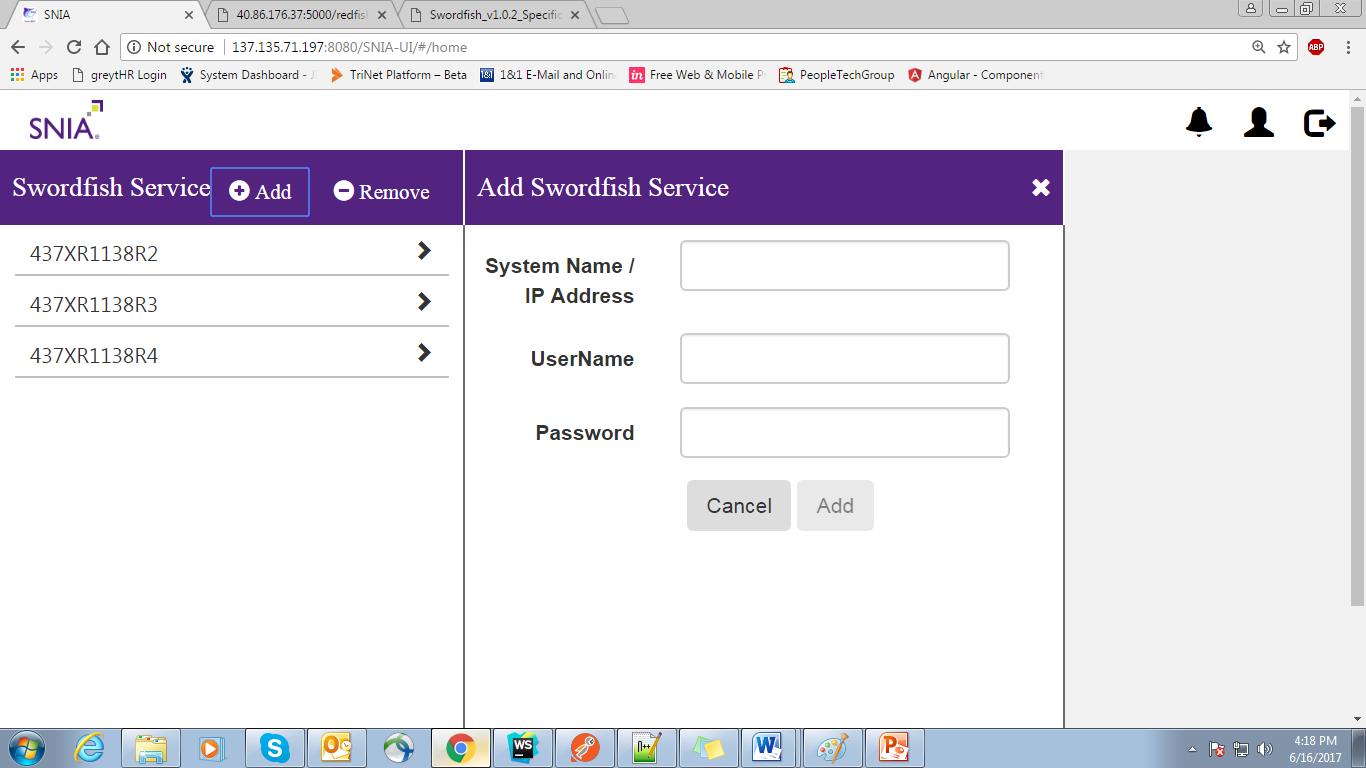
},

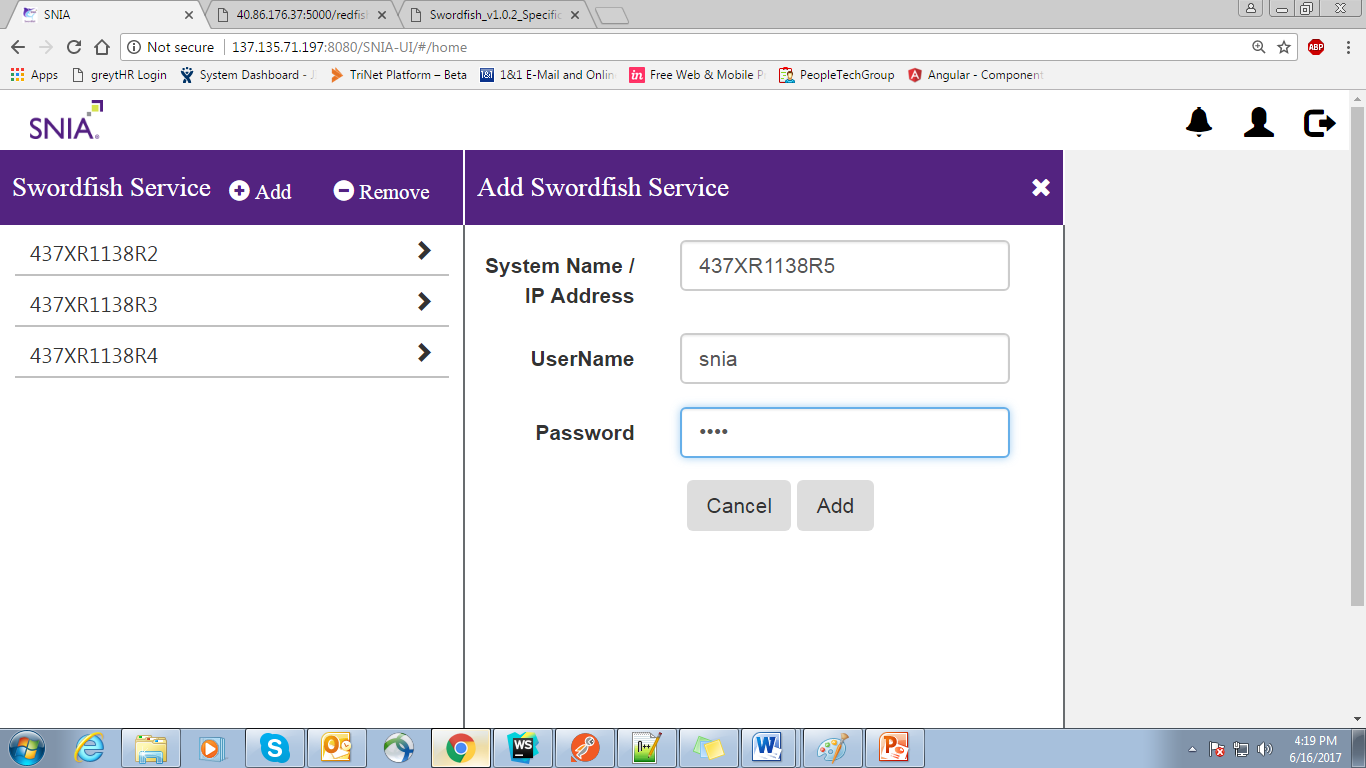
"Name": "Gold changed"

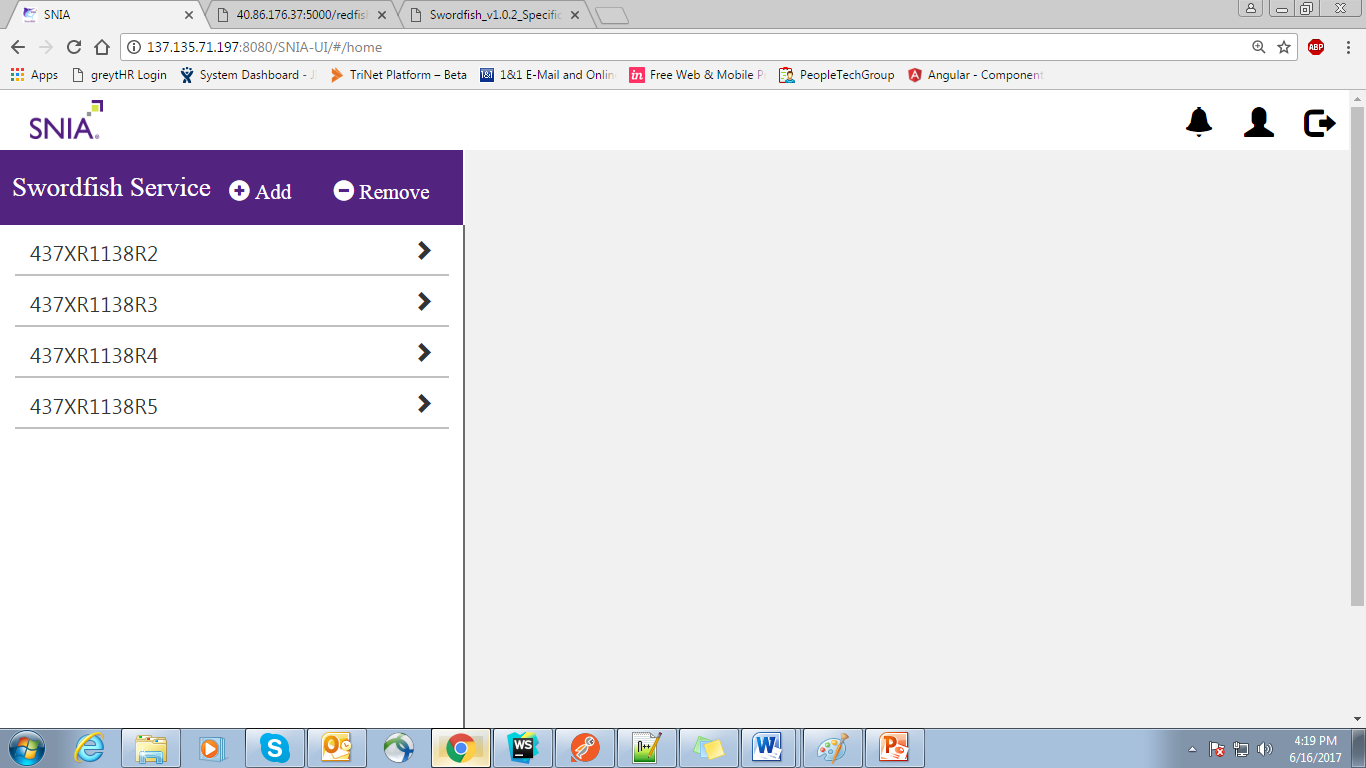
}

**3 . Add A Service**

Client can be able to a new Service





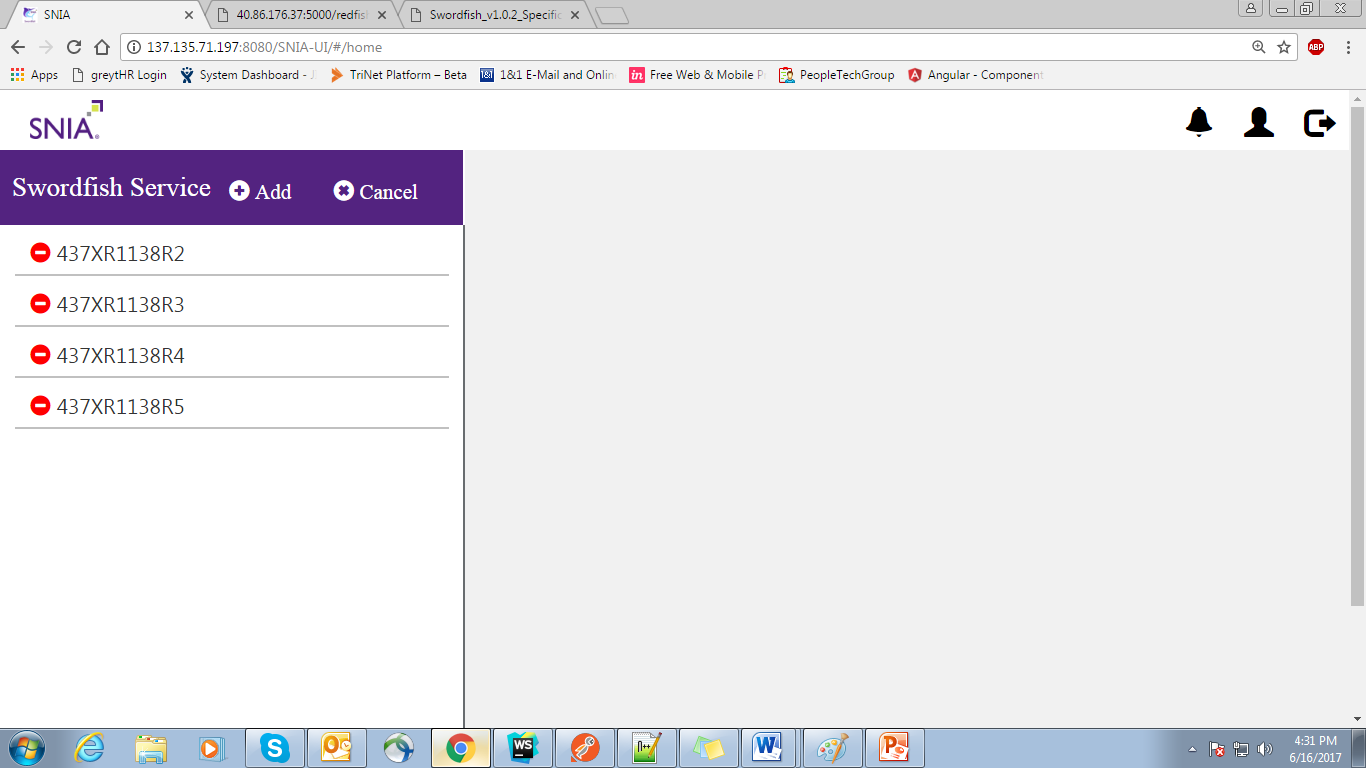


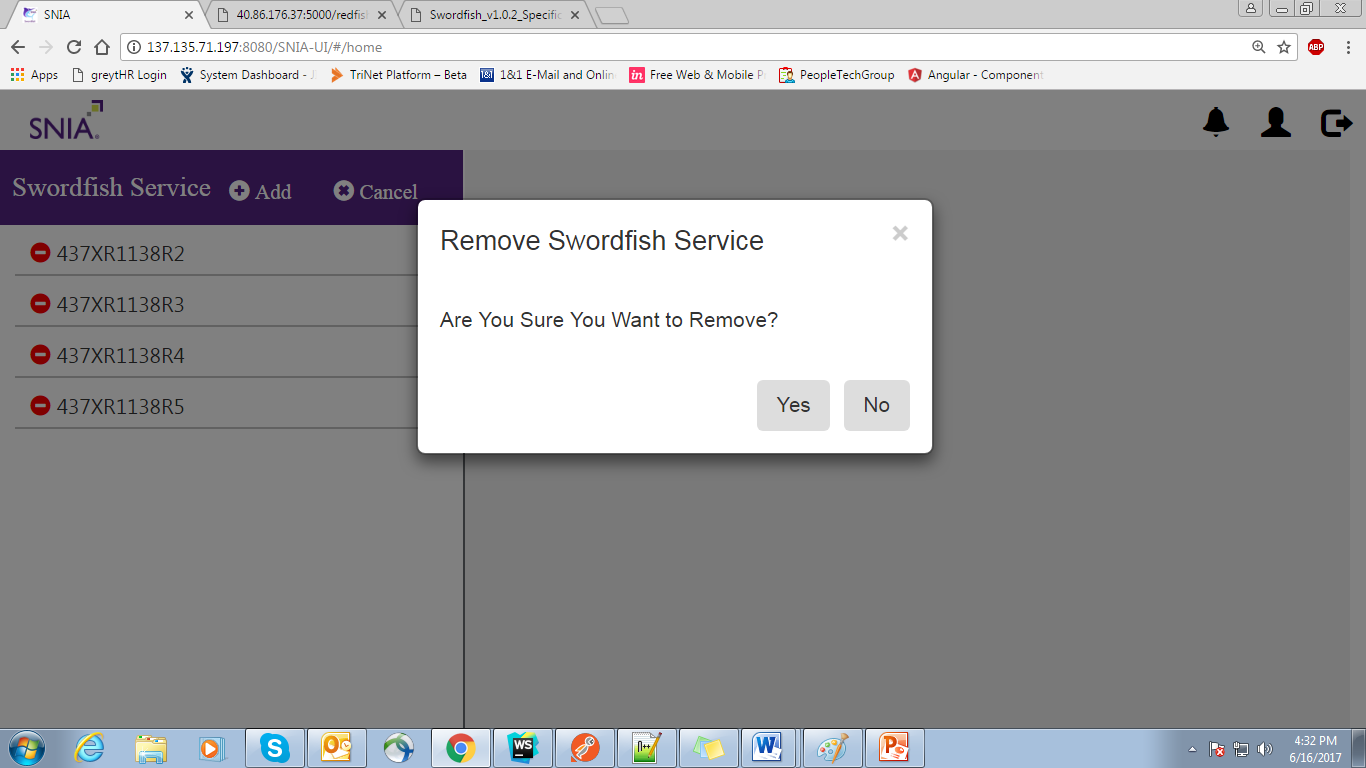
The newly added service has the inbuilt collection structure which will be created from Emulator structure

**4 . Remove A Service**

A delete option Will be provided to remove the collection structure related to the service

**UI-View:**





Service Will be removed if the client wishes to remove it by clicking ‘YES’.

UI-View after delete

