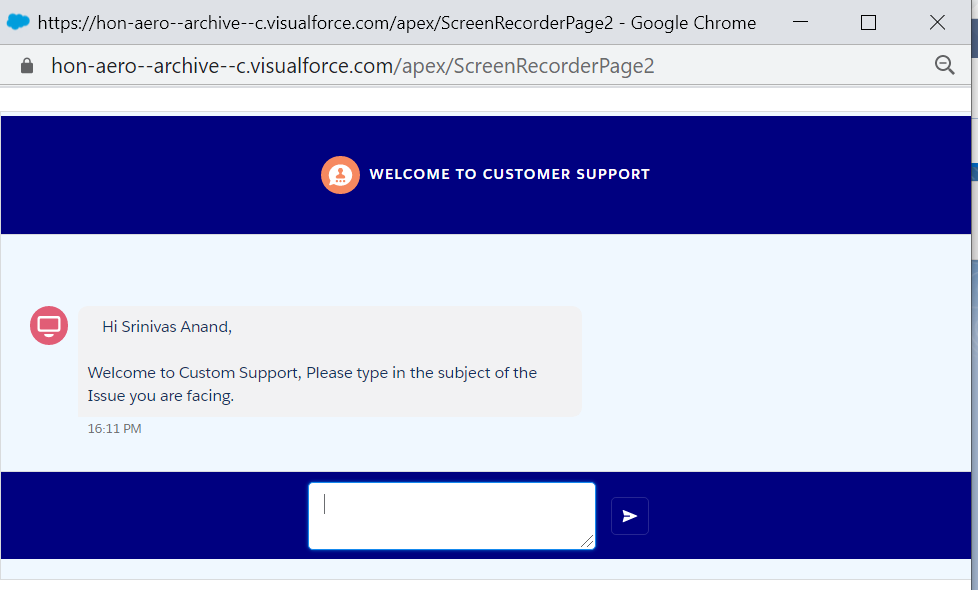
**Screen Recorder KT Document**

**Lightning URL -** <https://hon-aero--archive--c.visualforce.com/apex/ScreenRecorderPage2>

**Classic URL -** <https://hon-aero--archive--c.visualforce.com/apex/ScreenRecorderPage>

Used for allowing salesforce users to create service-related incidents in Salesforce or Service Now based along with the debug logs getting captured and optional screen recording.



User can click on the ‘Customer Support App’ lightning tab and open the chat application which will ask users to enter the issue details.

**Components used:**

**VF Page**:

ScreenRecorderPage2 – contains the screen recording logic along with lightning out to host the screenRecorderChatApplication which has aura:dependency on the screenRecorderChatComponent

Commnuncation between VF page and the lightning comonents is done using application events.

**Lightning Component:**

ScreenRecorderComponent – Used in custom tab to popup the ScreenRecorderPage2 VF page in classic context.

captureLogEvent – Application event triggered when log capture or record screen is selected. Handled by vf page and screenRecorderChatComponent

chatElementComponent – creates each chat element dynamically from the parent screenRecorderChatComponent

chatSearchComponent - used to search for assignment group / queue for assigning the case/ incident

chatSearchResults – shows search results

clearValueEvent – event triggered when search selection is cleared

getPromiseLightning – contains a helper method to call server-side controller. All components extend this component.

oSelectedValueEvent – event triggered when assignment group is selected

radioSelectEvent – event triggered when case creation environment is selected (salesforce/ service now)

screenRecorderChatApplication – used in vf page and contains lightning out dependency on screenRecorderChatComponent

screenRecorderChatComponent – component that hosts the chat layout. The chat elements are created here dynamically. Textual input and file input are provided based on conditions.

**Classes**:

clearRecorderDataBatch – Batch class to clear saved files from library for specified time frame in days.

logDeleteQueueable – Deletes debug logs and trace flags.

DebugLogControllerMock – Mocking Responses for callouts.

customMetadataService – Used to access the custom metadata in the test classes. custom metadata can’t be queried from test classes and it will return an empty list if we call a method that is querying a custom metadata.

To avoid this we have created a class that has a private Map<string,List<sobject>> and a method called getQuery.

The getQuery method is called with the query in the controller.

Ex: **customMetadataService.getQuery('SELECT id, MasterLabel,DeveloperName**

**FROM Custom\_support\_Instruction\_\_mdt ‘);**

This method will actually query the records when not in Test Context and returns a list of sobjects.

When we call from test class it will return the value contained in the map. So in the test class we need to do the below:

**customMetadataService.bigObjectQueryMap.put('SELECT id, MasterLabel,DeveloperName**

**FROM Custom\_support\_Instruction\_\_mdt ‘, metadataList );**

Where metadatalist is the list of custom\_support\_Instruction\_\_mdt record instances that we want to be returned in the controller when the query executes.

screenRecorderChatController

Controller for Screen Recorder Application to get metadata and create records in salesforce and service now.

Methods:

customMetaWrapper – Wrapper class used to send the below details to application.

1. List of metadata for chat instructions
2. Current user details along with the user id from target environment to be used as the caller
3. Flags to indicate running a job for deleting trace flag when already one is present for same user
4. Flag to indicate running a job for deleting logs when it has crossed the limit of 300 MB

RemoteSiteWrapper – used to return name value pairs of groups or queues from salesforce or target env

startRequest – Returns a continuation which makes callout to target environment to get the mapped current user id and to check if the logs are exceeded or trace flag for the running user already exists. Called from screenRecorderChatComponent.

processResponse – callback method for continuation which returns the customMetaWrapper object.

getIdFromResponse – gets log Ids and trace flag Ids from response. Called from chatElementComponent

setTraceFlag – used to set trace flag for the current user Called from chatElementComponent

getDebugLogs – get debug logs for the current user (only latest 20 logs) Called from chatElementComponent

deleteTraceFlag – deletes trace flag for the user Called from chatElementComponent

getLogBody – gets log body using debug log Id and saves it in contentversion. Called from a list of promises from chatElementComponent using promise.all.

deleteLog – deletes debug log.

saveChunk, appendToFile, saveTheFile – used to save uploaded recorded using chunking of the file. Called from sceenRecorderChatComponent. Called from screenRecorderChatComponent.

createCase – Creates case in salesforce and contentDocumentLinks to the files for the created case. Called from screenRecorderChatComponent.

sendToSnow – Creates incident in target environment. Called from screenRecorderChatComponent.

sendToSnowGetGroup – Gets name value pair for groups in target environment. Called from chatSearchComponent.

createCaseGetGroup – Gets name value pair for groups in salesforce. Called from screenRecorderChatComponent.

**Objects:**

Screen\_Recorder\_settings\_\_c custom setting:

Used for API settings and default values for case/ incident creation

|  |  |
| --- | --- |
| **Field Name** | **Purpose** |
| Name | Unique key to get the setting using name from controllers |
| fieldValue\_\_c | Contains value of the default setting/ endpoint etc |
| isAPI\_\_c | Is used for identifying default values while creating ticket in target environment. Name of the records are prefixed by ‘API\_\_’ |
| isSFDC\_\_c | Is used for identifying default values while creating ticket in SFDC |

Customer\_Support\_Instructions\_\_c custom metadata:

Stores the chat instructions, Notes and used to configure the actions to be taken at each instruction

|  |  |
| --- | --- |
| Field Name | Purpose |
| Instructions\_\_c | Instructions to be appeared on chat |
| Note\_\_c | Note that appears in bold in the chat instruction |
| fieldname\_\_c | Maps the field names in target environment and SFDC where the reply has to go in for the current instruction while creating the case/ incident.  **Format stored: fieldname=sfdcFieldName;TargetFieldName;**  OR  If radio\_\_c=true  {  “Salesforce” : **SFDCFieldName**,  “Target”: **TargetFieldName**  } |
| Order\_\_c | Order in which the instructions should appear |
| Radio\_\_c | If true for any record, an environment selection button will be displayed at the instruction |
| SearchBox\_\_c | If true for any record, a group selection search box is displayed at the instruction |
| Capture | If true for any record, 3 buttons ‘Create Case/ Incident Only’ ,’ Capture logs Only ‘and ‘Capture Logs with Screen Recording’ are created at the instruction. |

**VF Page application: - same functionalities as above in classic**

VF page : ScreenRecorderPage

Controller : DebugLogController – has the same functionalities as the lightning version except the chat.

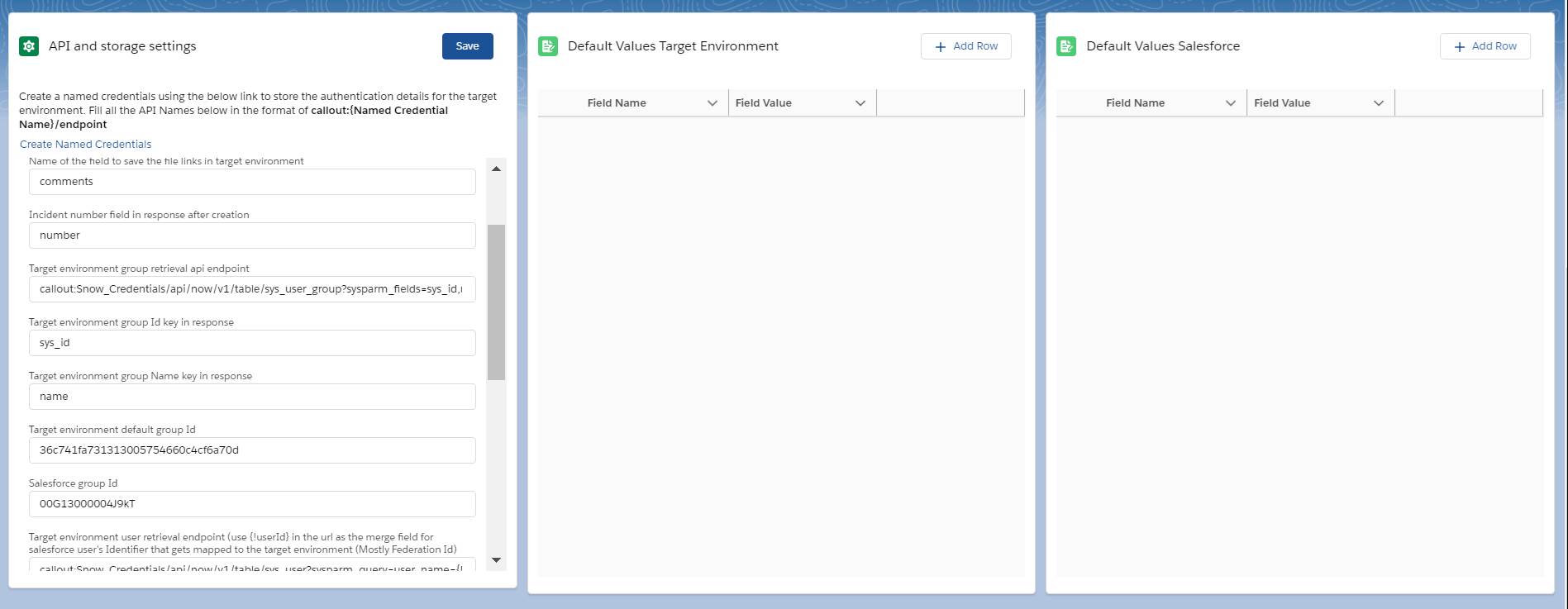
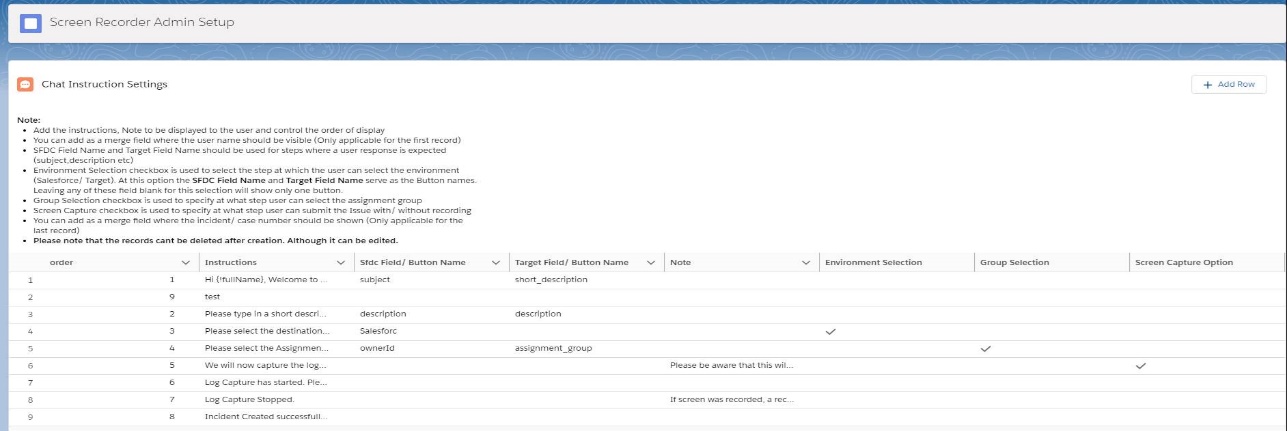
**Test Classes:**

DebugLogControllerTest

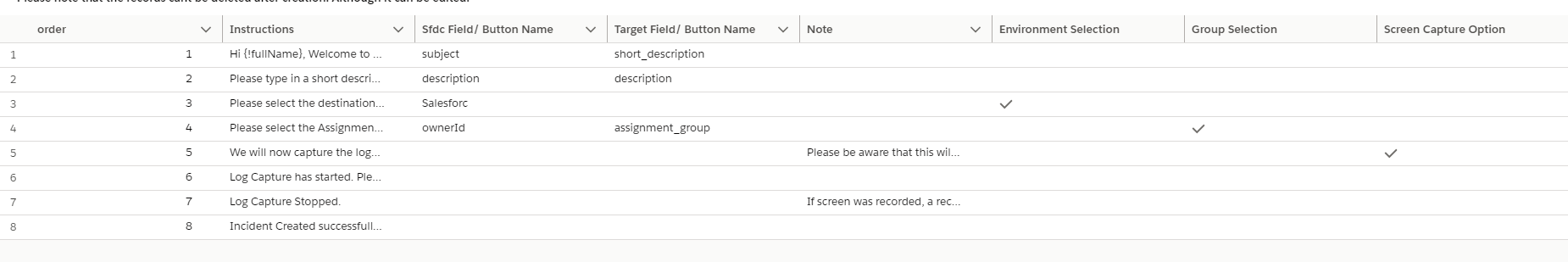
ScreenRecorderCSControllerTest

ScreenRecorderChatControllerTest

**Admin Screen:**



**Chat Instruction setting: (Customer support Instructions Metadata Object)**



1. The first component will be the chat instruction setting where the users can enter Instructions, Note, order of display to the users.
2. admins can add as a merge field where the user’s full name should be visible (Only applicable for the first record)
3. SFDC Field Name and Target Field Name should be used for steps where a user response is expected (subject,description etc) – This is stored in the field ‘fieldName\_\_c’ of the customer support instructions metadata.

**Format stored: fieldname=sfdcFieldName;TargetFieldName;**

If any one is left blank the semicolons are retained and the values are left blank.

**;TargetFieldName; or sfdcFieldName;;**

1. Group Selection checkbox is used to specify at what step user can select the assignment group. Here the fieldname\_\_c field will store the values of the fieldNames in Salesforce or target org where the assignment of Case/ Incident is specified in the previously mentioned format.

For Salesforce and service as target environment this will hold: **ownerId;assignment\_group;**

If only salesforce is needed then use ownerId;;

1. Environment Selection checkbox is used to select the step at which the user can select the environment (Salesforce/ Target). At this option the SFDC Field Name and Target Field Name serve as the Button names. Leaving any of these field blank for this selection will show only one button. This is stored in the field ‘fieldName\_\_c’ of the customer support instructions metadata.

Format stored : {

“Salesforce” : **SFDCFieldName**,

“Target”: **TargetFieldName**

}

If any one is left blank the JSON is created based on the values entered.

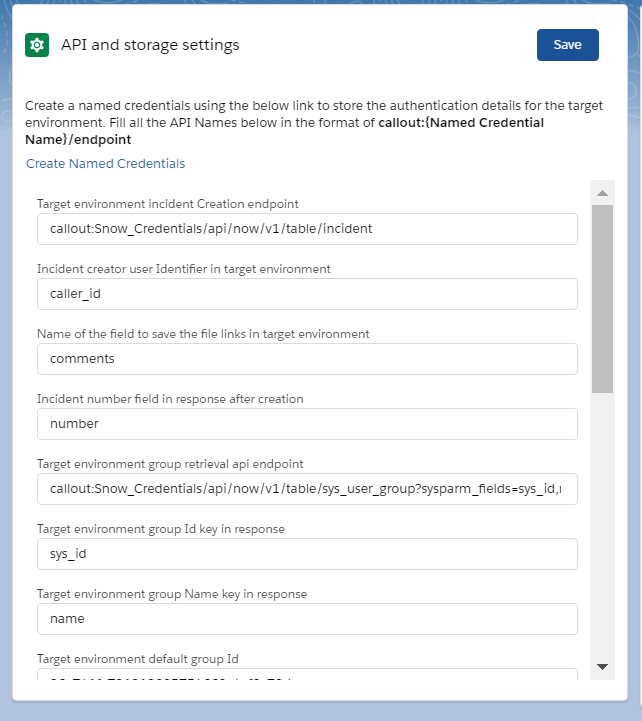
{

“Salesforce” : **SFDCFieldName**

}

1. Screen Capture checkbox is used to specify at what step user can submit the Issue with/ without recording.
2. You can add as a merge field where the incident/ case number should be shown (Only applicable for the last record)
3. Please note that the records can’t be deleted after creation. Although it can be edited.

**API and storage settings (Screen Recorder Settings Custom Setting Object)**

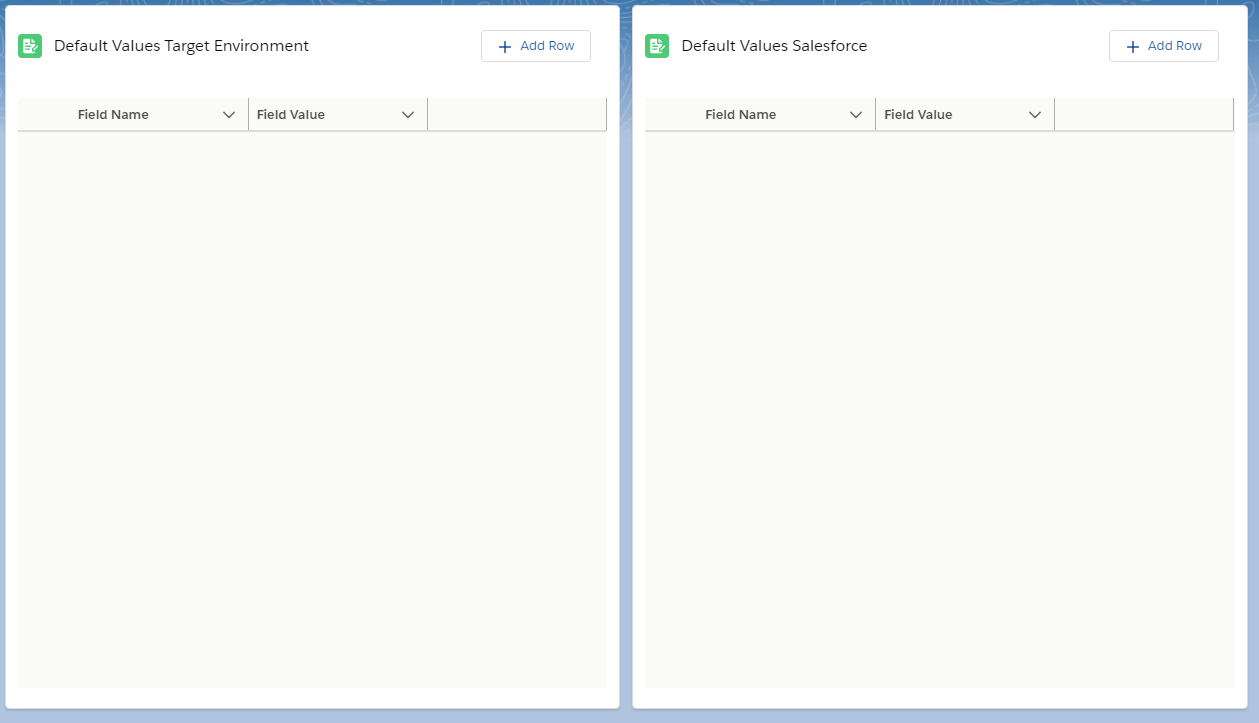


1. This will store the values of the API and storage related settings in the custom setting ‘Screen Recorder Setting’ as a (name, value) pair.
2. The component name here is apiSettingCreationComponent where the input field will hold the custom setting record name in the aura:id attribute and value in the value attribute. Value is got in the handleChange method.

**<lightning:input class="slds-m-around--small" aura:id="API\_incidentNumber" label="{!v.incNumber}" onchange="{!c.handleChange}"/>**

1. For all endpoints the API endpoints are to be added in the format of **callout:{Named Credential Name}/endpoint**
2. For the endpoint which is used to get the group names from target environment, we need to use the {!userId} merge field in the url to replace the user identifier from Salesforce user which has been mapped to the target environment. (Usually the federation Id)
3. The descriptions of each field and what it is used for is mentioned in the lightning component.

**Default Target and Salesforce Environment Values: (Screen Recorder Settings Custom Setting Object)**



1. This is used to add the default values for case/ Incident creation (Type, configuration Item etc).
2. For Salesforce default values, the custom settings will have isSFDC\_c =true and fieldValue\_\_c will hold the default value and Name will hold the field name.
3. For Target Environment default values, the custom settings will have isAPI\_c =true and fieldValue\_\_c will hold the default value and Name will hold the field **name prefixed by ‘API\_’**.

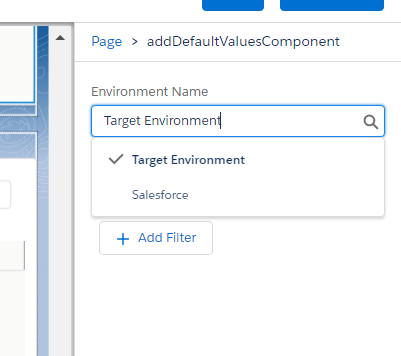
**Admin Screen Components:**

**LightningComponents:**

apiSettingCreationComponent – for API and Storage setting creation.

addDefaultValuesCompoent - for default values configuration for Salesforce and target environment.

Same component is used twice on the Lightning Page with a design attribute ‘environmentName’ passed from the Page.



Create\_Update\_Metadata – used to add chat related Instructions.

**ApexClass:**

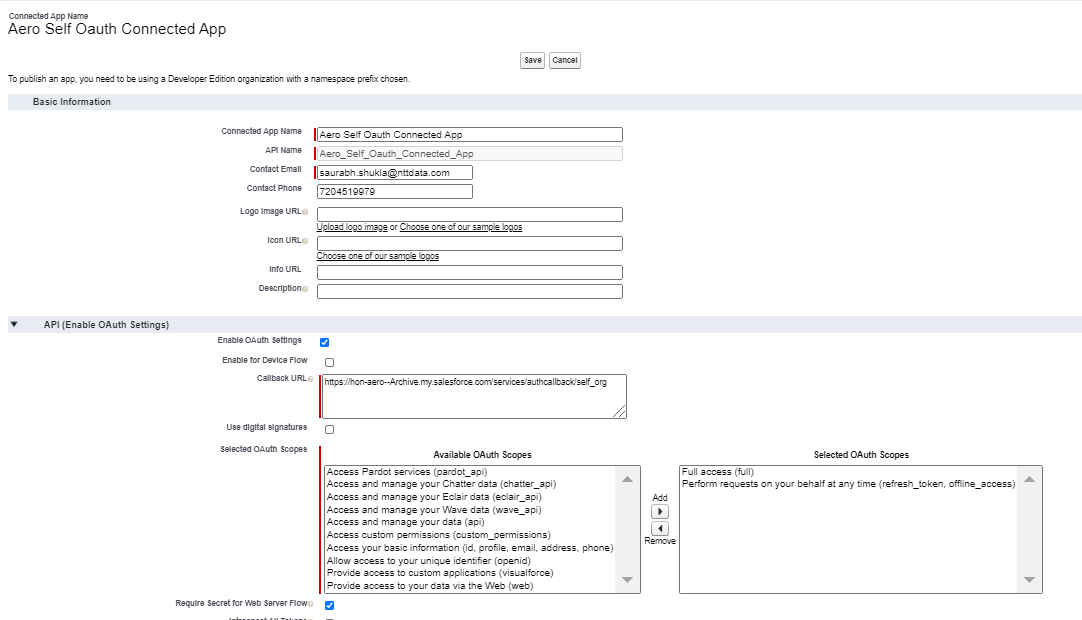
AddingMetadata- creating chat related instructions

ScreenRecorderCustomSettingController – creating API setting and default values.

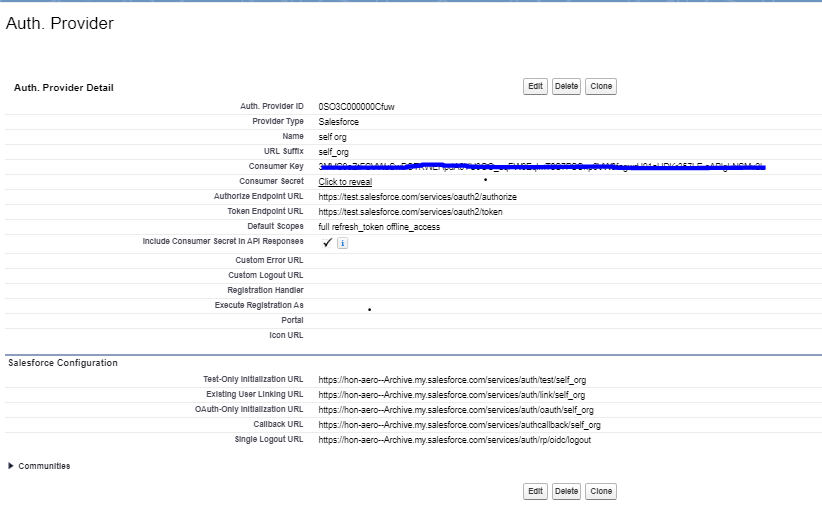
Create\_Update\_Metadata – create/ edit metadata deployment callback class

**Post Installation steps:**

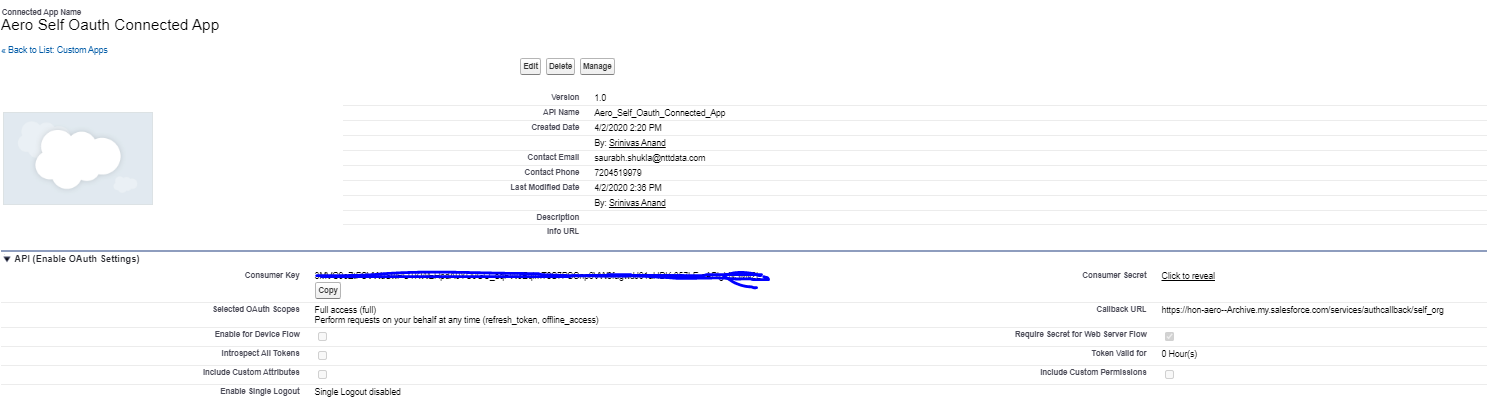
1. To access tooling API from the same org using a admin user for creating trace flag, getting logs etc we need to create a connected app to the same org. Use the admin screen to configure the chat instructions, api endpoints and other settings.
2. Create a connected app in the org with oauth enabled.



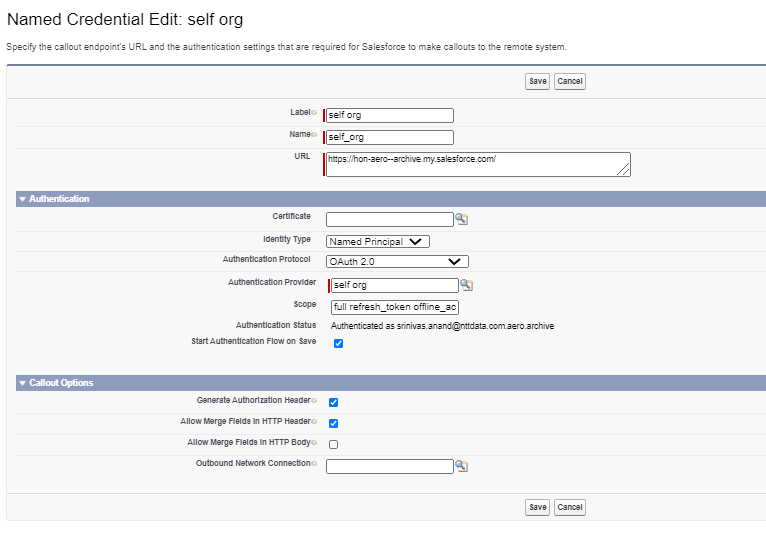
1. Create an auth provider in the same org with the client id and client secret of the same org. Make sure you add the same scopes in the connected app, auth provider and named credentials.



1. Copy the login url from the auth provider related list and use it in the connected app callback url.



1. Create a named credential named ‘self\_org’ and use the Named principal authentication and oauth 2.0 option.
2. Lookup the auth provider that was created and use the same in the named credential.
3. Allow merge fields in the Http header, generate authorization header, start authentication flow on save should be checked.



1. Authorize the named credential on save.

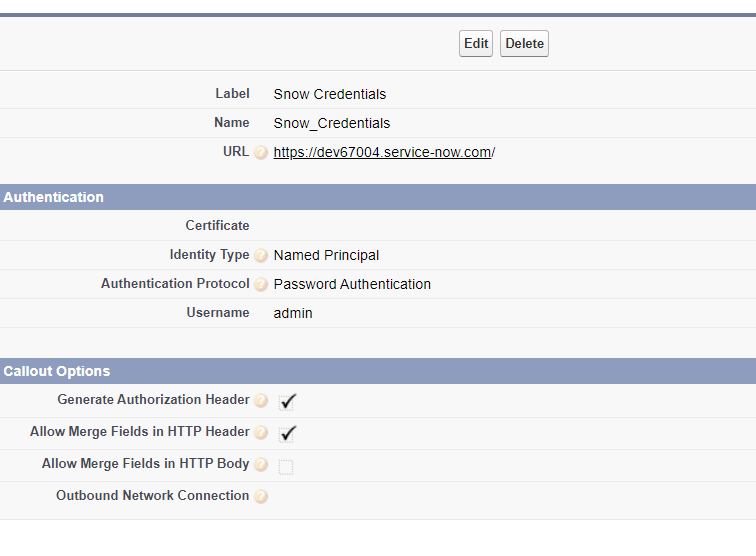
**Named Credentials:**

Create a named credential of your choice of authentication to connect to the target environment where the incidents need to be created.

Below screenshot shows the named credentials created for SNOW integration with username password authentication.

While specifying endpoints in the Admin tool or custom settings, ensure you type the endpoints in the below format:

*callout:{Named Credential Name}/endpoint*



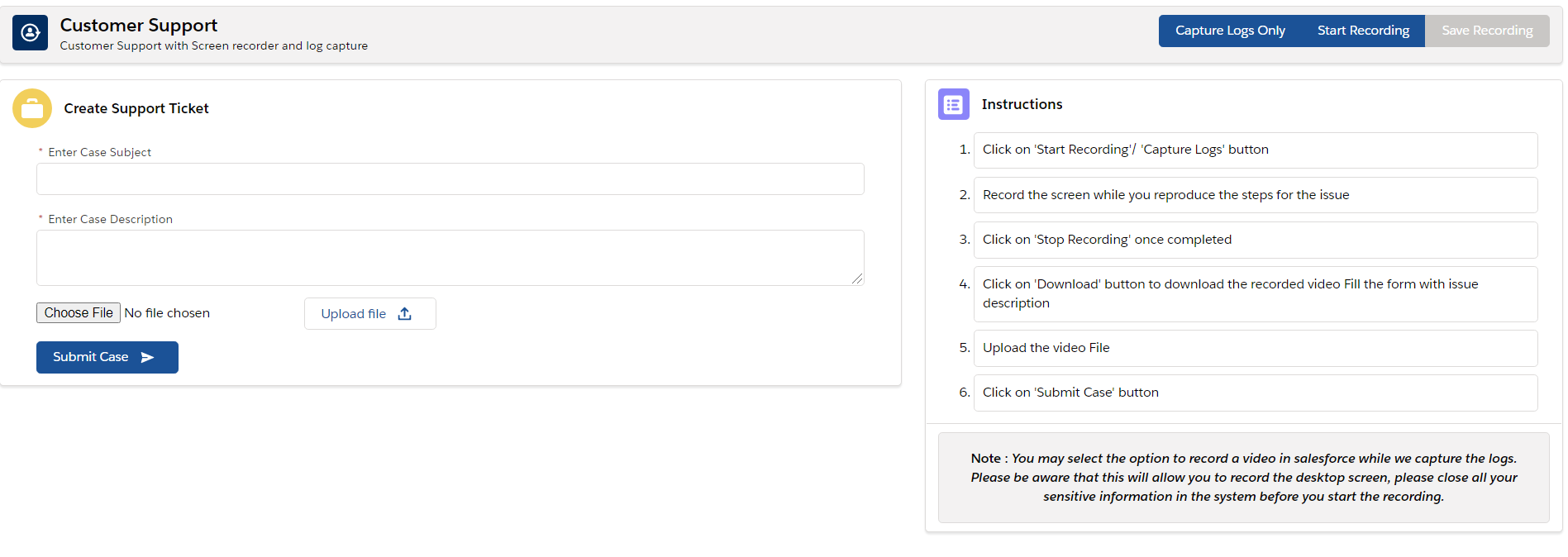
**Classic Version of Screen Recorder:**

For Orgs which are not classic enabled they can go to the custom Link ‘Contact Support’ to open the VF page ScreenRecorderPage.

Functionalities are similar to the lightning Page.

1. Click on 'Start Recording'/ 'Capture Logs' button
2. Record the screen while you reproduce the steps for the issue
3. Click on 'Stop Recording' once completed
4. Click on 'Download' button to download the recorded video Fill the form with issue description
5. Upload the video File
6. Click on 'Submit Case' button

**Note : *You may select the option to record a video in salesforce while we capture the logs. Please be aware that this will allow you to record the desktop screen, please close all your sensitive information in the system before you start the recording.***



**Classic version custom setting names:**

Go to ‘screen recorder settings’ custom setting and create records with below Names and fill the desired values in the field\_value\_\_c field.

[API\_Caller\_Id\_Field](https://hon-aero--archive.my.salesforce.com/aIL3C0000008OVw?setupid=CustomSettings) – Incident creator user Identifier in target environment

[API\_creation\_endpoint](https://hon-aero--archive.my.salesforce.com/aIL3C0000008OW9?setupid=CustomSettings) - Target environment incident Creation endpoint with the named credential.

[API\_incidentNumber](https://hon-aero--archive.my.salesforce.com/aIL3C0000008OW6?setupid=CustomSettings) – incident number identifier from the json response for incident creation

[debugLevelId](https://hon-aero--archive.my.salesforce.com/aIL3C0000008OW4?setupid=CustomSettings)- default debug level Id for capturing logs

[file\_deletion\_Interval](https://hon-aero--archive.my.salesforce.com/aIL3C0000008OWB?setupid=CustomSettings) – log and recorded video file deletion interval

[FLDVF\_description\_field](https://hon-aero--archive.my.salesforce.com/aIL3C0000008Ob5?setupid=CustomSettings) – Description field on the incident for target environment

[FLDVF\_subject\_field](https://hon-aero--archive.my.salesforce.com/aIL3C0000008ObA?setupid=CustomSettings) – subject field on the incident for target environment

[libraryName](https://hon-aero--archive.my.salesforce.com/aIL3C0000008OVy?setupid=CustomSettings) – Library api name where the recordings and files need to be stored

[recorded\_video\_alert\_note](https://hon-aero--archive.my.salesforce.com/aIL3C0000008OWA?setupid=CustomSettings) – alert note to be added in the description field of the incident regarding the file deletion interval

[targetEnvUserMappingField](https://hon-aero--archive.my.salesforce.com/aIL3C0000008OWD?setupid=CustomSettings) – user mapping field name which will be used to map the user record from salesforce to target environment (usually federationidentifier/ employee id)

**Default values for incident creation:**

For Default fields to be added use the same custom setting object and create fields in the below format.

‘API\_’ + field name where default values need to be added.

Add the default values in the field\_value\_\_c in the custom setting record. Check the isAPI\_\_c checkbox to true.

**Instructions:**

Go to ‘Customer Support Instructions’ custom metadata and create a record with the DeveloperName=’Instructions’.

Fill the Instructions in the Instructions\_\_c custom field in separate lines. This will be shown as bullet points in the page.

Fill the Note field if required which will be displayed in bold after the instructions.

**Batch scheduling:**

To clear the recorded video files and logs from documents fill in the deletion interval from the admin page or create a record with Name file\_deletion\_interval in ‘screen recorder settings’ custom settings and fill in the number of days for which the records are deleted.

Ex: field\_value=14 will delete the files which were created 2 weeks ago and if the job is scheduled to run everyday then it will delete the files everyday that were created 2 weeks ago from that date.

Batch class name: clearRecorderDataBatch

Can be scheduled from apex scheduling window.

**Permission Sets:**

Screen recorder Admin Permission – Assigned to the admin to use the ‘Screen Recorder Admin’ Application and the screen Recorder application.

Screen recorder User Permission – Assigned to the any user who uses the screen Recorder application.

**Library (Content Workspace):**

Create a library to store the files related to the application (logs and recordings) and give Author permissions to all internal users and ‘Workspace administrator’ Permissions to all admins.