Datazora Technical Documentation

1. Introduction

Welcome to the **KIJI** for Salesforce. This document provides detailed information about the package, including installation instructions, configuration steps, key features, and troubleshooting tips.

Purpose

The purpose of this document is to guide administrators and developers in the effective use of the **KIJI** package.

Audience

This document is intended for Salesforce administrators, developers, and any users involved in the installation, configuration, and maintenance of the package.

2. Package Overview

Package Features

- Fetch and manage events from the Datazora source within Salesforce.
- Apply a customizable filter to view specific events based on tag criteria.
- Capture and store detailed logs error handling during events processing.

Components Overview

Apex Classes:

- **CalloutMock**: Simulates callouts for testing purposes, allowing you to test callout logic without making actual requests.
- **CalloutUtil**: A utility class for handling HTTP callouts and managing API interactions.
- **DatazoraDTO**: An Apex class used to encapsulate and manage data structures. This class helps in organizing and transferring data efficiently.
- **DatazoraUtil**: A utility class containing common methods used across the package, including data manipulation.
- **EventsListViewController**: The main controller class for handling requests related to the Events List View component.
- **EventsListViewSelector**: Contains logic for querying data based on the provided identifier.

- **EventsListViewService**: Provides services for data retrieval, transformation, and processing before displaying them in the UI.
- **Logger**: Manages logging operations, saving error and info logs to a custom Log object.
- **SecurityCheck**: Ensures that the user has the necessary permissions for CRUD operations before executing them.
- **TestDataBuilder**: Utility class for building test data structures, ensuring consistent and reliable unit tests.

Apex Test Classes:

- **CalloutUtilTest**: Tests the functionality of the CalloutUtil class, including handling HTTP requests and responses.
- **DatazoraUtilTest**: Ensures the correctness of data manipulation methods within the DatazoraUtil class.
- **EventsListViewControllerTest**: Verifies that the EventsListViewController handles events-related requests and interactions correctly.
- **EventsListViewSelectorTest**: Tests the querying logic implemented in the EventsListViewSelector class.
- **EventsListViewServiceTest**: Confirms that data retrieval, transformation, and processing performed by EventsListViewService are functioning as expected.
- **LoggerTest**: Assesses the logging operations and data storage performed by the Logger class.
- **SecurityCheckTest**: Tests the security checks implemented by the SecurityCheck class to ensure correct permissions are enforced.

Custom Metadata:

- **Datazora_Credentials__mdt**: Stores API credentials and endpoint URLs for external systems used by the package.
    - **Event_Service**: Stores credentials and configuration details used for accessing and retrieving event data from external services.
    - **Tags_Service**: Stores credentials and configuration details used for accessing and retrieving tags data from external services.

Custom Settings:

- **Datazora_Account_Fields_Payload__c**: Used to store the API names of Account fields that are sent in the payload to the Datazora service via API requests.

  Fields:

    - Field_Api_Name__c: Stores the API name of the Account field to be sent to the service.

Custom Labels:

- Categories **Datazora:** Store text values used in the Salesforce UI for the KIJI package, including localized text values for a consistent user experience across different languages.

Layouts:

- **Datazora_Credentials__mdt Layout**: Layout used to configure and display the Datazora Credentials metadata in the Salesforce UI.

Lightning Web Components:

- **eventsListView**: The main component responsible for displaying and filtering events on the record page.

Lightning Aura Application:

- **eventsListViewApp**: A Lightning Out app required for integration within the Visualforce component.

Visualforce Pages:

- **accountEventsListViewPage**: Uses to place events on an Account record page in the Salesforce Classic interface.
- **contactEventsListViewPage**: Uses to place events on a Contact record page in the Salesforce Classic interface.

Visualforce Components:

- **eventsListViewComponent:** Displays a list of events within the Salesforce Classic interface. This component is invoked on specified Visualforce pages.

Objects:

- **Log__c**: Custom object used for storing error and informational logs generated by the package.

  Fields:

  - Error_Location__c: Specifies the location in the code or process where the error occurred, helping identify the source of the issue.
  - Error_Message__c: Contains the message describing what went wrong.
  - Exception_Code__c: Stores a specific code or identifier related to the exception or error.
  - Stack_Trace__c: Provides a detailed stack trace for debugging, which can help trace the sequence of method calls leading to the error.

- **Datazora_Credentials__mdt**: Custom metadata type used to store API credentials and other configuration details.

  Fields:

  - MasterLabel: Store the name of the specific service (e.g., "Event Service" or "Tags Service") which the credentials relate to.
  - URL__c: The API endpoint or service URL associated with the credentials.
  - Authorization_Code__c: Stores the authorization code or API key required for authenticating with the service.

Permission Sets:

- **Datazora_Access_Admin**: Grants administrative users full access to configure and manage all features and settings of the KIJI package.
- **Datazora_Access_Client**: Provides standard users with access to view and interact with Datazora-related data and features, but limits administrative and configuration capabilities.

Remote Site Settings:

- **DatazoraApi**: Stores the endpoint URL for interacting with the Datazora API, used for package operations.

Tabs:

- **Log__c Tab**: Tab that provides administrators with access to view and manage logs stored in the Log__c object.

Translations:

- **ja:** Contains Japanese translations for the KIJI package labels, ensuring a user-friendly interface for Japanese-speaking users.

Test Coverage Details

To ensure comprehensive coverage of the package, the following table lists the key Apex classes along with their respective code coverage percentages.

Apex Class	Coverage Percentage, %
CalloutUtil	100

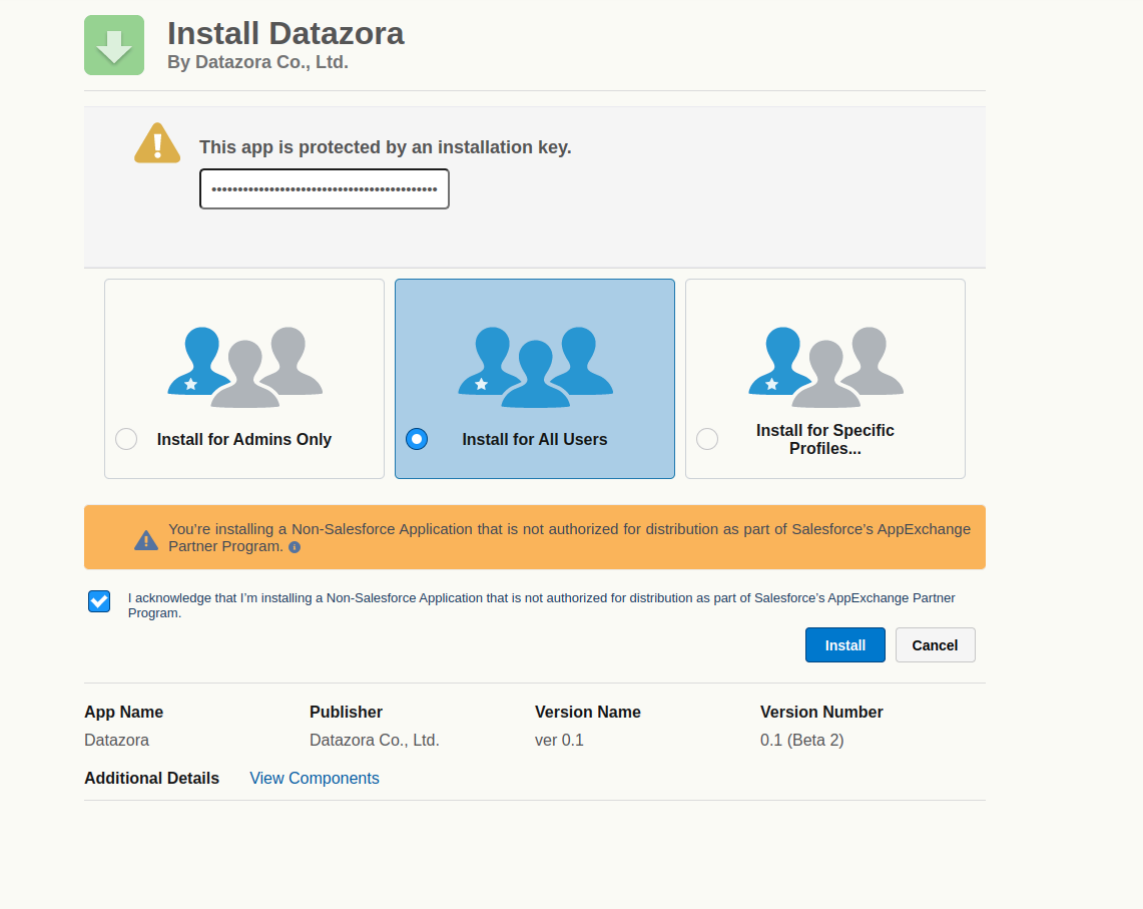
DatazoraDTO	100
DatazoraUtil	91
EventsListViewController	100
EventsListViewSelector	90
EventsListViewService	97
Logger	100
SecurityCheck	98

3. Installation Guide

Installation Steps

1. Visit the **Datazora** listing on Salesforce AppExchange.
2. Click *Get It Now*. Follow the prompts to install the package.

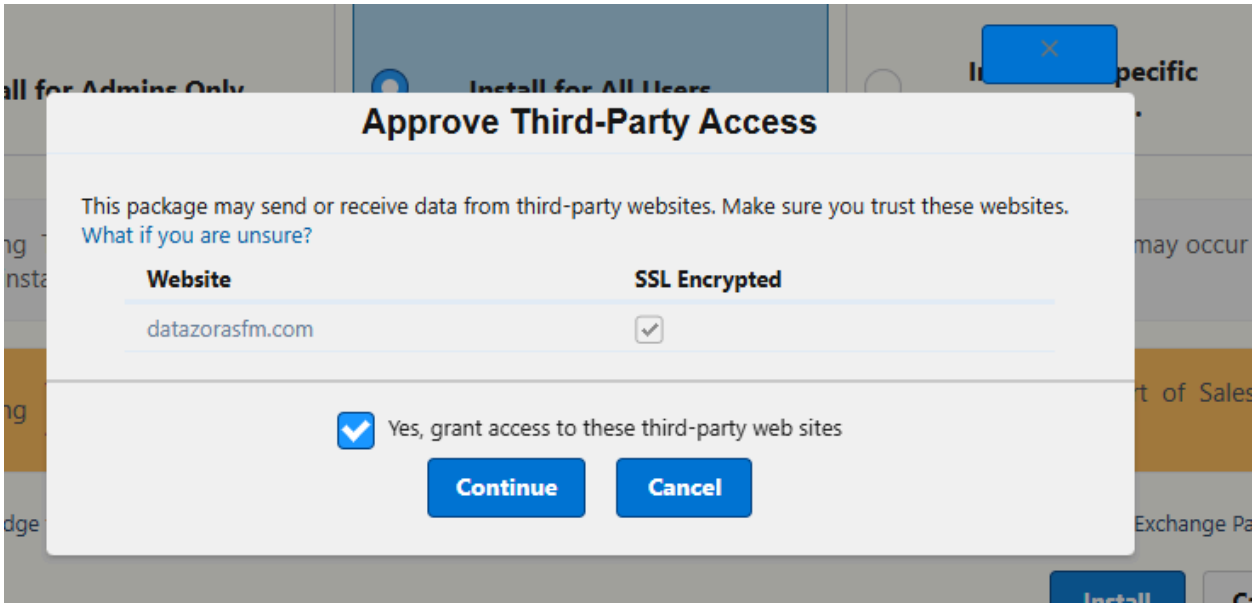
3. Choose Installation Type. Select either "Install for Admins Only," "Install for All Users," or "Install for Specific Profiles."



Enter the installation key:
c2FsZXNmb3JjZV9taXp1aG9zYzo1RUlzOUdlUVFqdjlmaDhi

Follow the on-screen instructions and click **Install**.

4. Approve any third-party access requests, if applicable. Click **Continue**.



5. Wait for the installation to complete. You will receive a confirmation email.

Post-Installation Steps

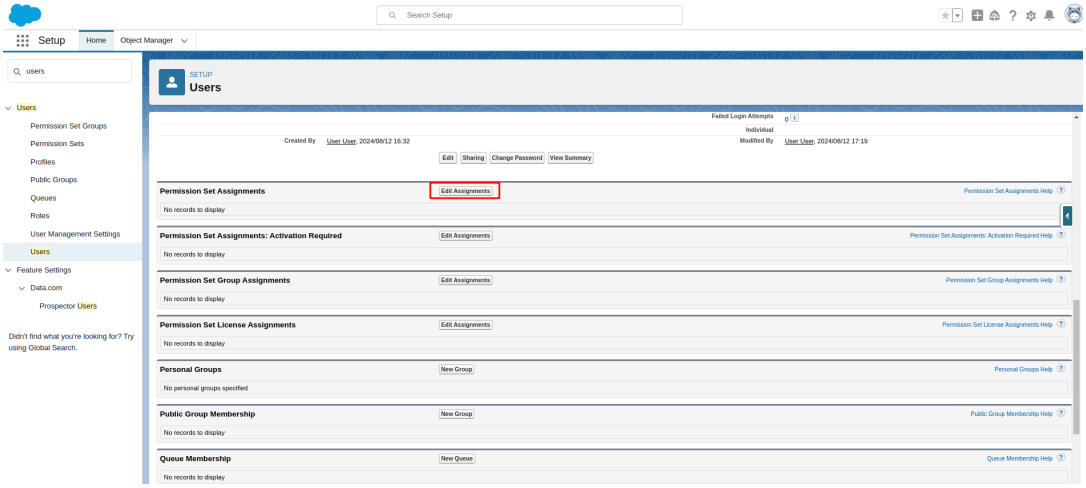
- Assign the necessary permissions sets to the appropriate users.

  Datazora_Access_Admin: Provides full administrative access and should be assigned to **admin** users.

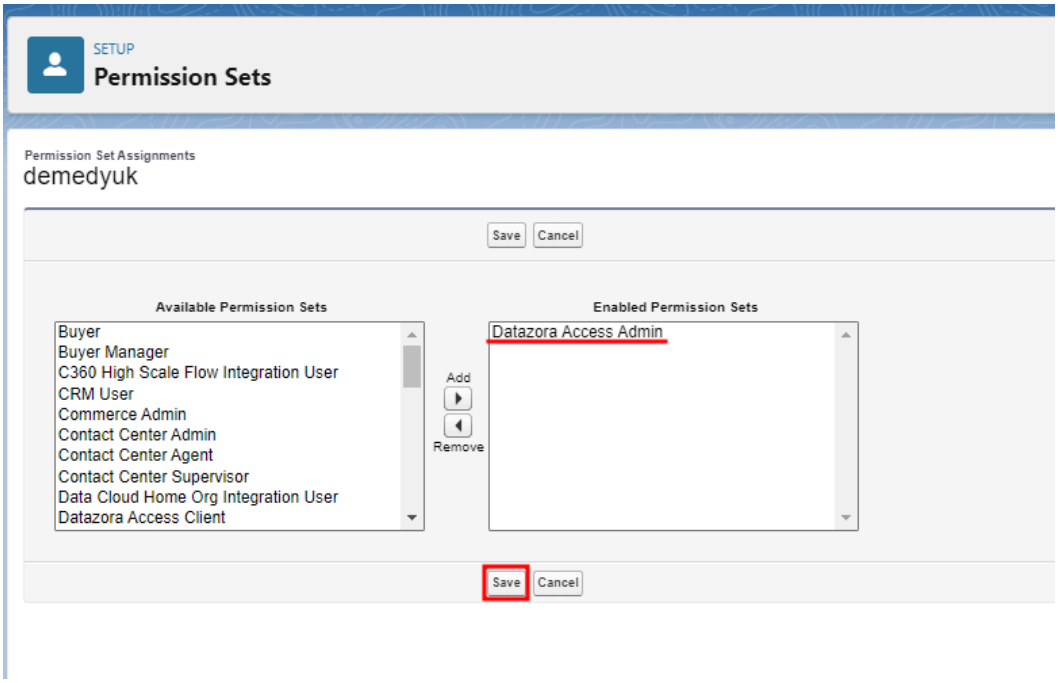
  Datazora_Access_Client: Provides limited access for end users or clients.

  Setup steps:
  - Go to the Setup and navigate to ‘Users’.
  - Click on the user who needs the permission set.

- In the ‘Permission Set Assignments’ section, click on the ‘Edit Assignments’ button.

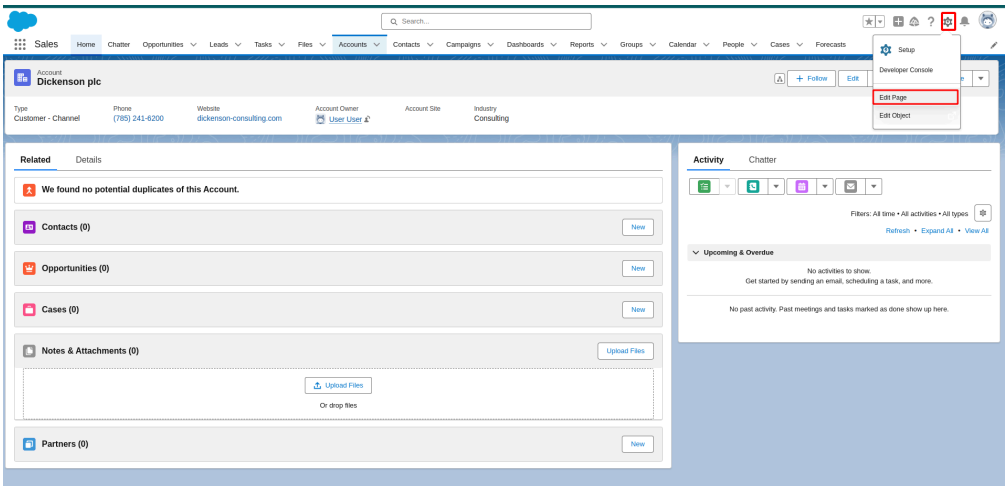


- Move the necessary permission set to the ‘Enabled Permission Sets’ column and click Save.

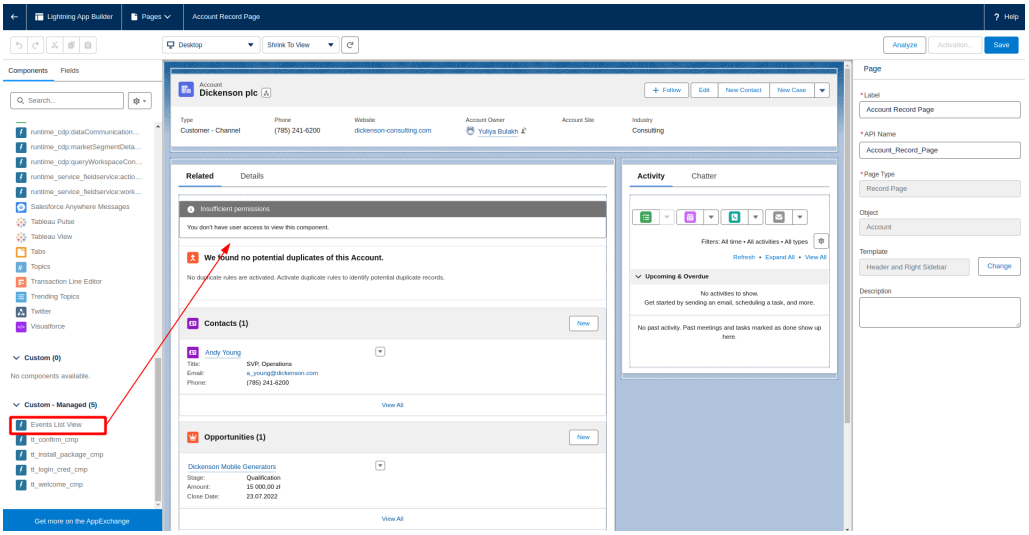


- Add Events to Record Page Layout
  - **Navigate to the Record Page**.
    - Go to the desired record page, such as Account or Contact by selecting the appropriate tab in Salesforce.
    - Click on the gear icon in the top-right corner of the screen.

- From the dropdown menu, select **Edit Page**. This will open the Lightning App Builder in edit mode, allowing you to customize the page layout.

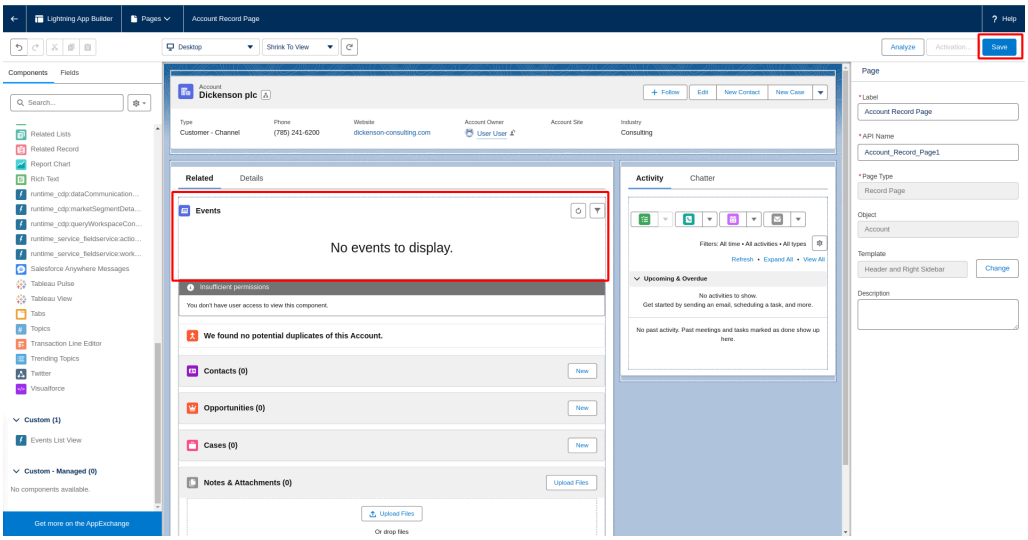


- **Locate the Event List View Component.**
  - On the left side of the Lightning App Builder, scroll down to the **Custom - Managed** section and drag and drop the ‘Event List View’ component into the desired area of the page layout.



- **Save and Activate.**

- Once the component is placed and configured, click **Save** in the top-right corner.



- If this layout is new, you may be prompted to activate it. Choose the appropriate activation settings based on your organization's requirements, such as setting the layout for specific profiles, record types, or making it the default.

- **Check the component on the page**.
  - Go back to the record page to ensure the **Event List View** component is displayed correctly and functioning as expected.

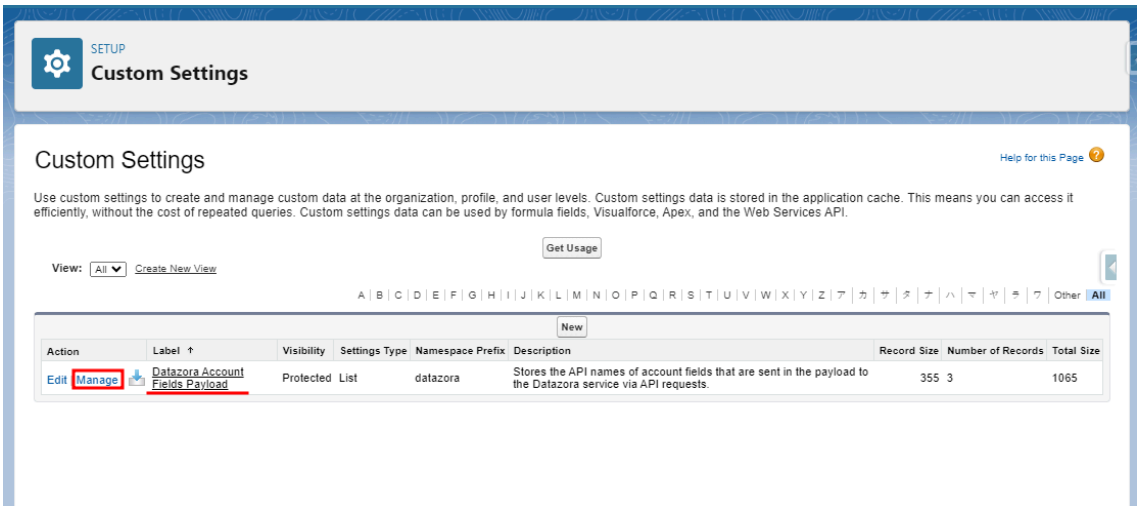
4. Configuration Guide

- **Populate a Datazora Account Fields Payload Custom Setting**.

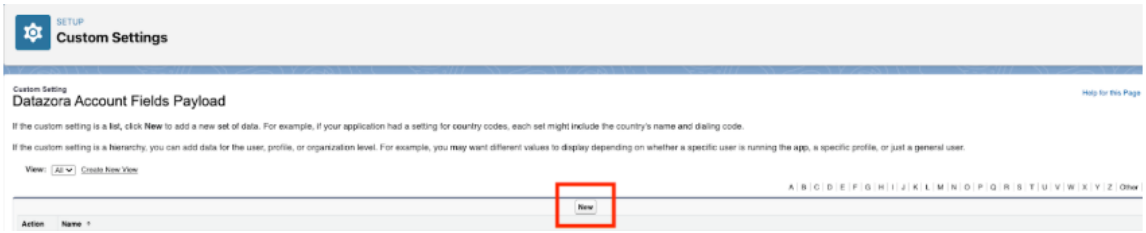
  Populating a Datazora Account Fields Payload custom setting is crucial for ensuring that the correct API names of account fields are sent to the Datazora service during API requests. This configuration allows for seamless data integration and ensures that the necessary account information is accurately transmitted, facilitating effective events processing and management.

  - Go to Setup and navigate to 'Custom Settings'.

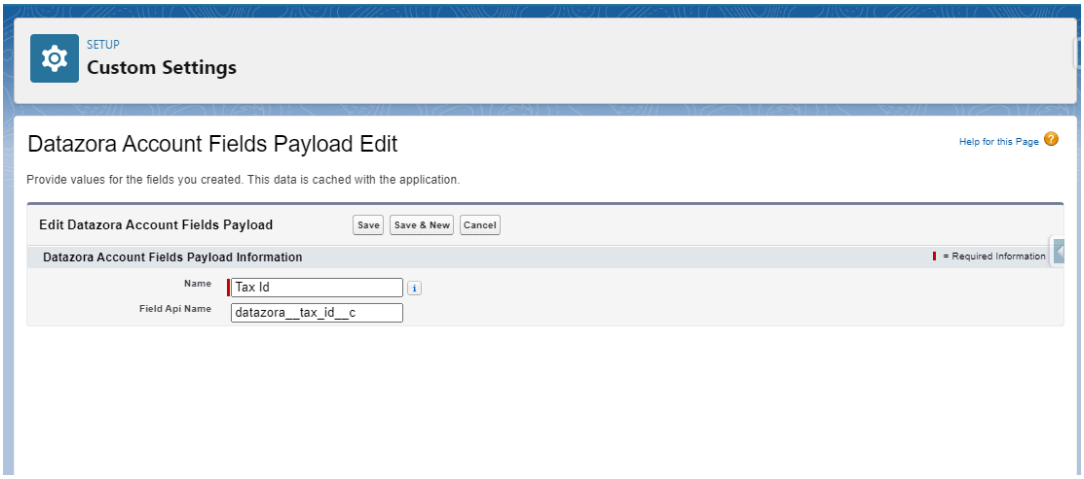
- Find *Datazora Account Fields Payload* in the list of custom settings and click **Manage**.



- Click New to create a new custom setting record.



- In the Field API Name field, enter the API name of the Account field that will be sent to the Datazora service.



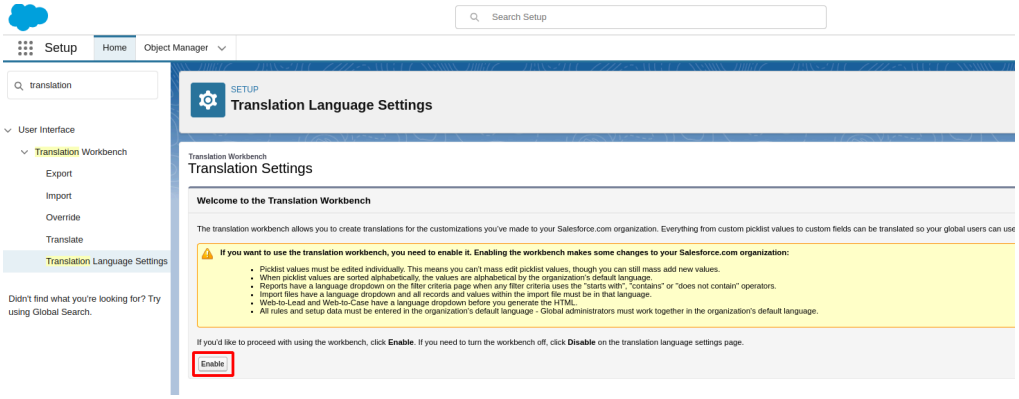
Note: Ensure that the Field API Name matches a valid API name from the Account object in your Salesforce instance, and that users who will be using this custom setting have read access to these fields.

- Click Save. Repeat the steps for any additional fields you need to configure.

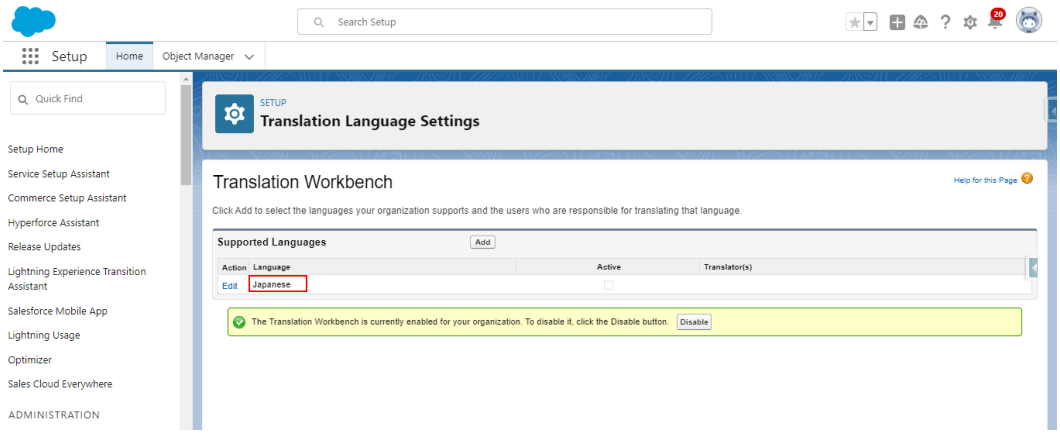
- **Adding Translations to Custom Labels.**

By following these steps, users can ensure that all custom labels used within the component are properly translated to meet the needs of different language users.

- Ensure Translation Workbench is Enabled: Before adding translations, make sure that the Translation Workbench is enabled in your Salesforce org. To do this, go to **Setup** and navigate to **Translation Language Settings**. Click Enable.

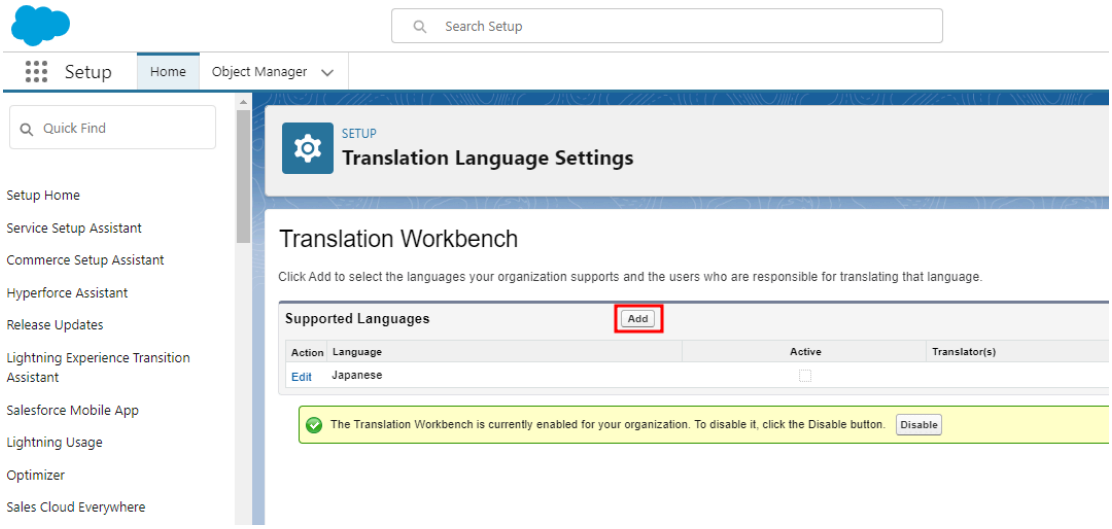


- The **Japanese** translation is already included in the package.

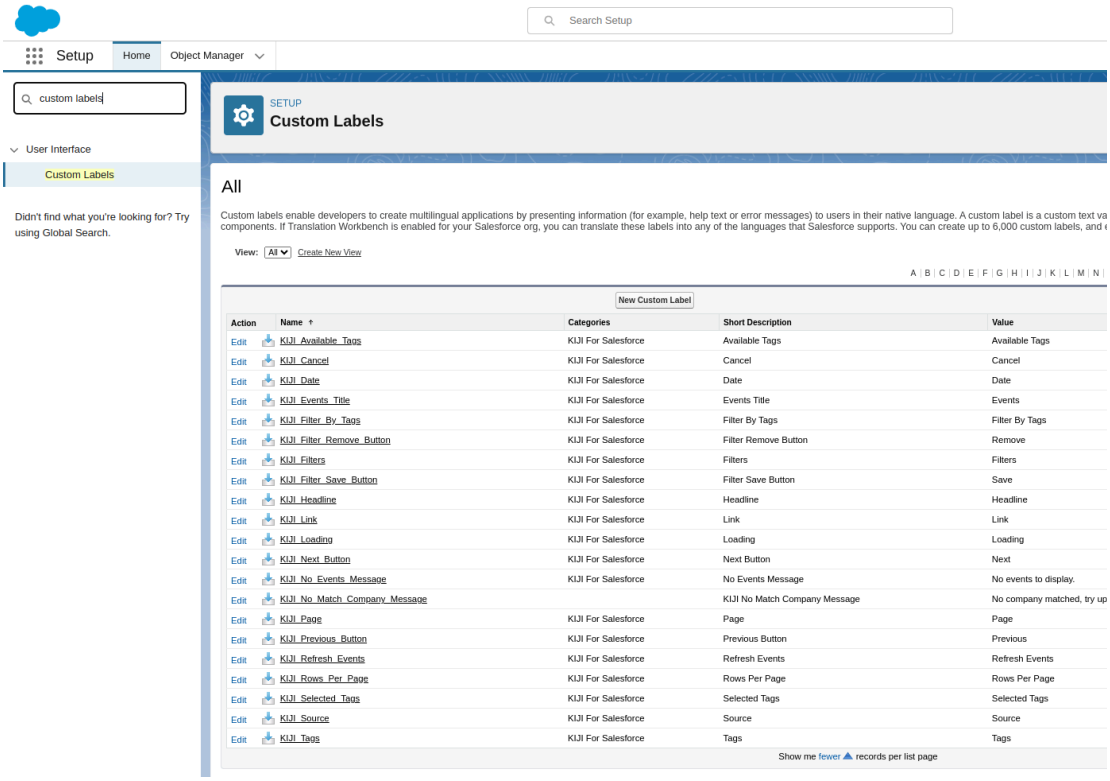


- Add Languages to Translation Workbench: Ensure that the languages you want to translate your custom labels into are added to the Translation Workbench. You can do this by clicking **Add** in the **Translation Language Settings** and adding

the desired languages.

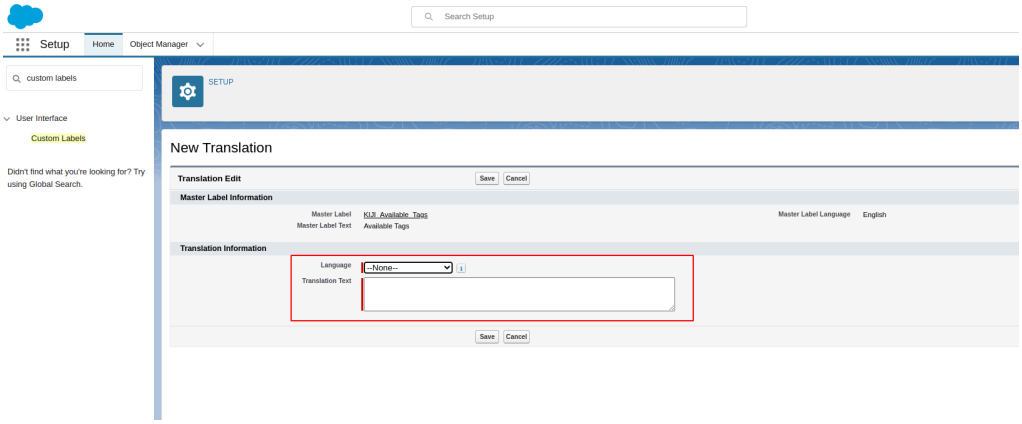


- Navigate to Custom Labels: Go to Setup and click on the Custom Labels.



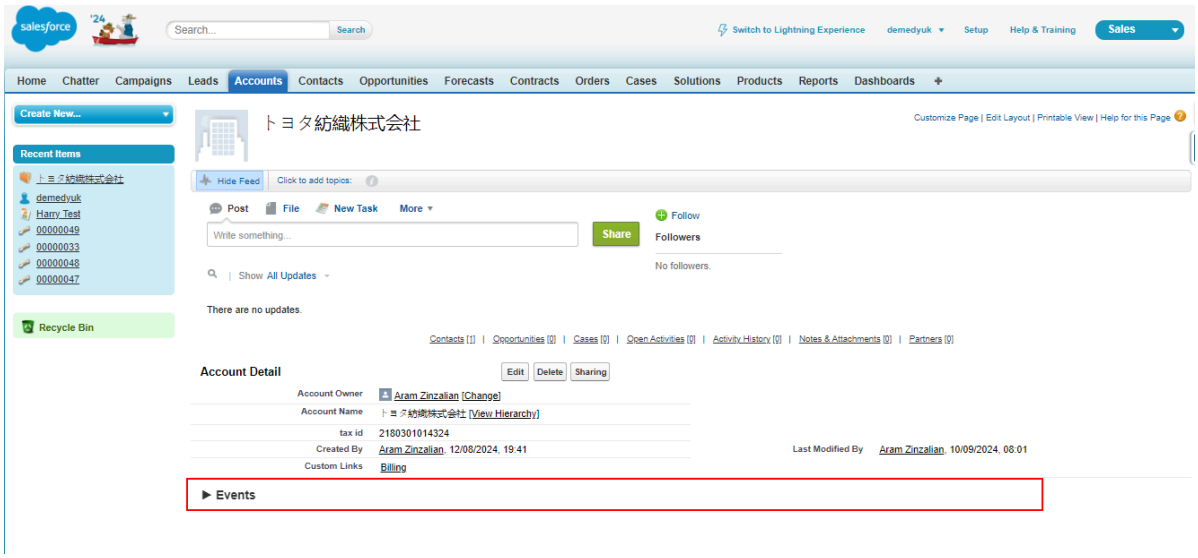
- Find and select the custom label you wish to translate. Click on the label name to view details.
- Add Translations:
  - Click **New Local Translation** or **Edit** for an existing translation.
  - Enter the translated value for the desired language in the provided text box.

- Ensure that you select the appropriate language from the dropdown menu.

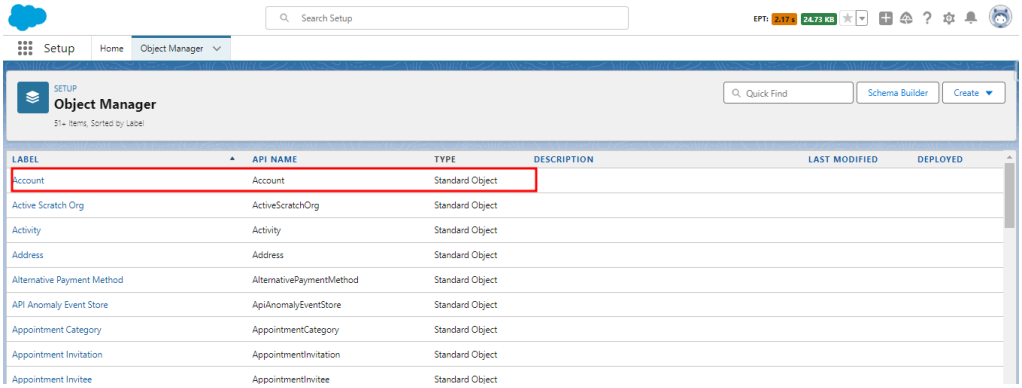


- Save Changes: Click Save to apply the translations.

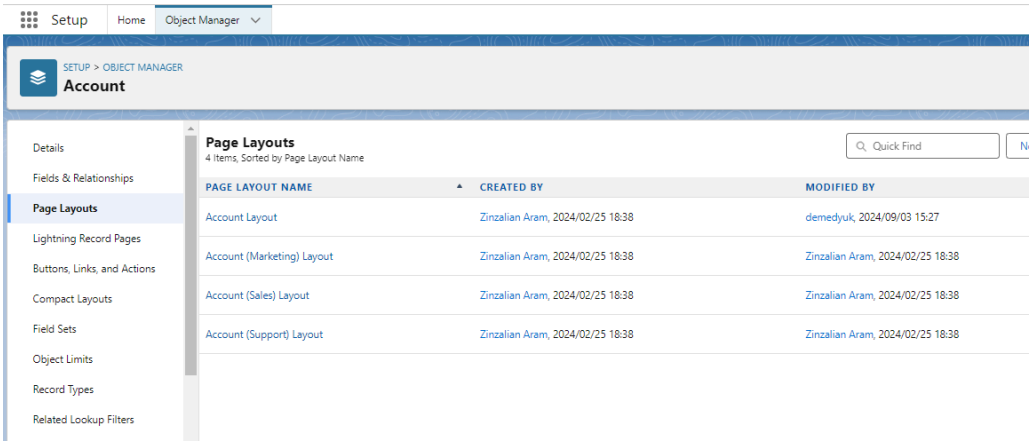
- **Add Events Component to the Page in Classic View.**



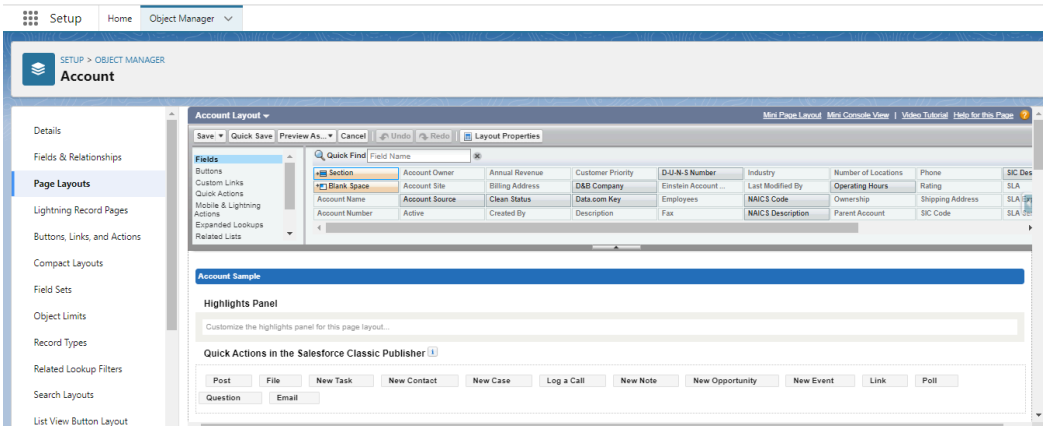
- Go to Page Layout: Go to Setup and click on Object Manager. Find and select the object you want to customize (Account or Contact).



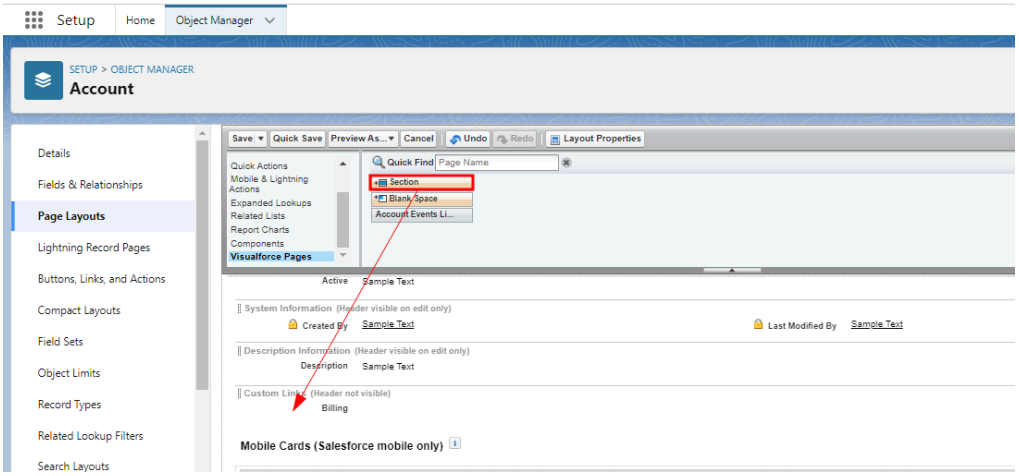
- In the left sidebar, find and click on **Page Layouts**.



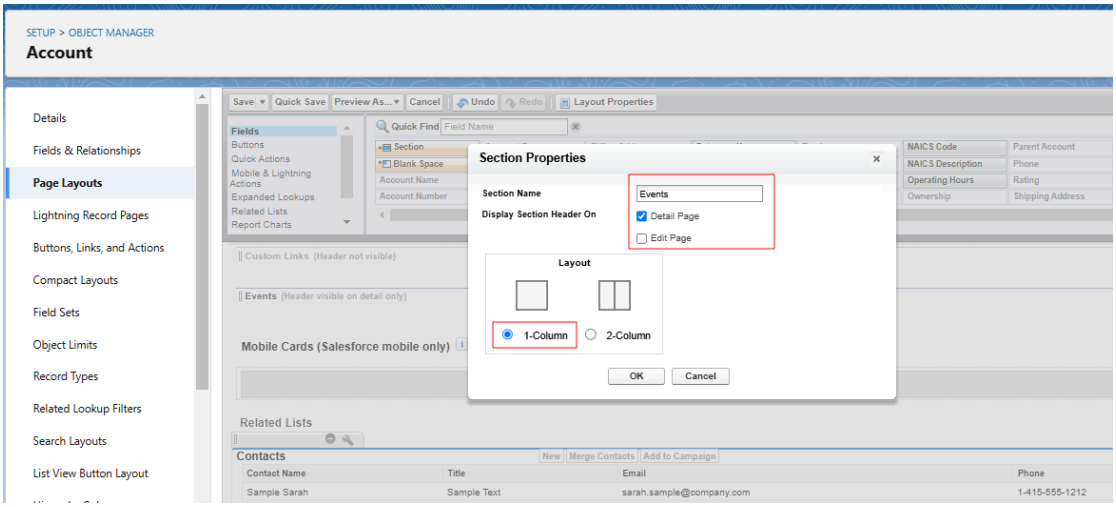
- Click on the name of the page layout you want to modify. This will open the page layout editor.



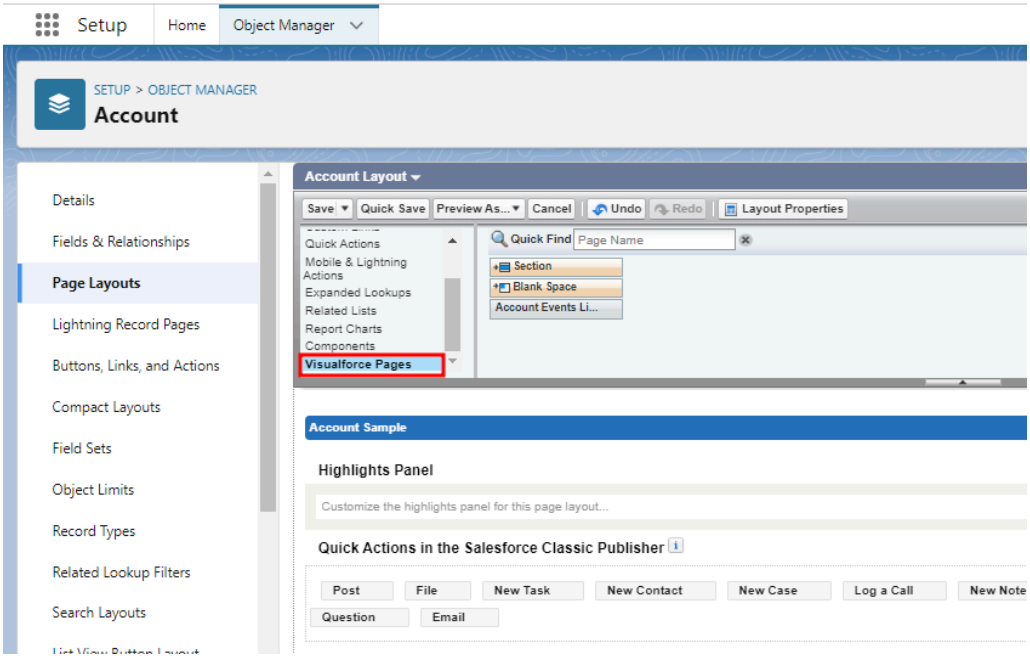
- Create a Section on Page Layout: Drag the **Section** component from the palette onto the layout in place where you want to see the component.



- Configure the new section's properties (e.g., label and layout) as on the screen and click Save.

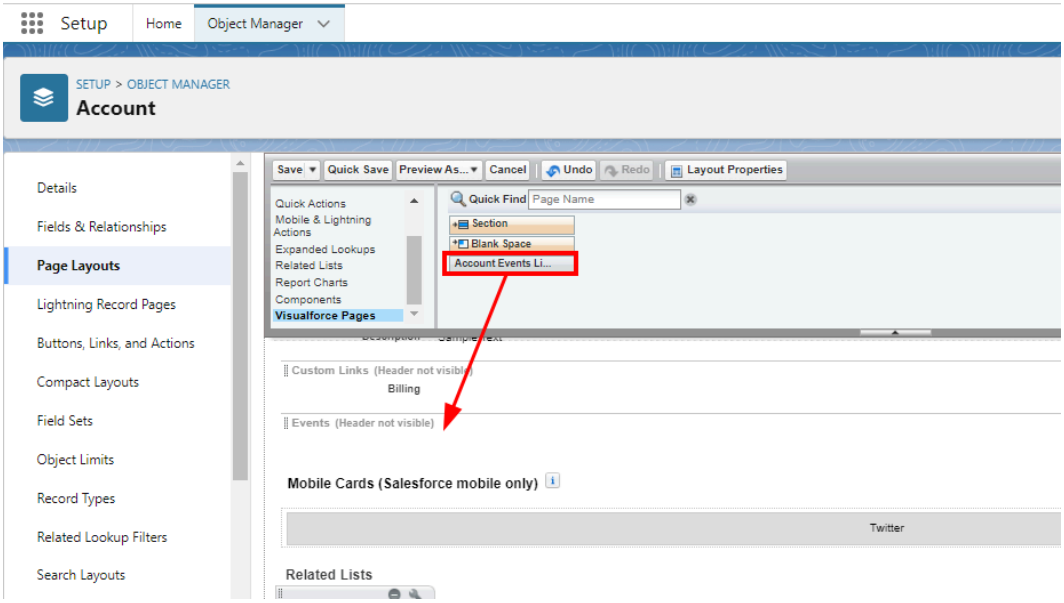


- In the page layout editor, find the **Visualforce Pages** section in the palette on the top of the layout editor.



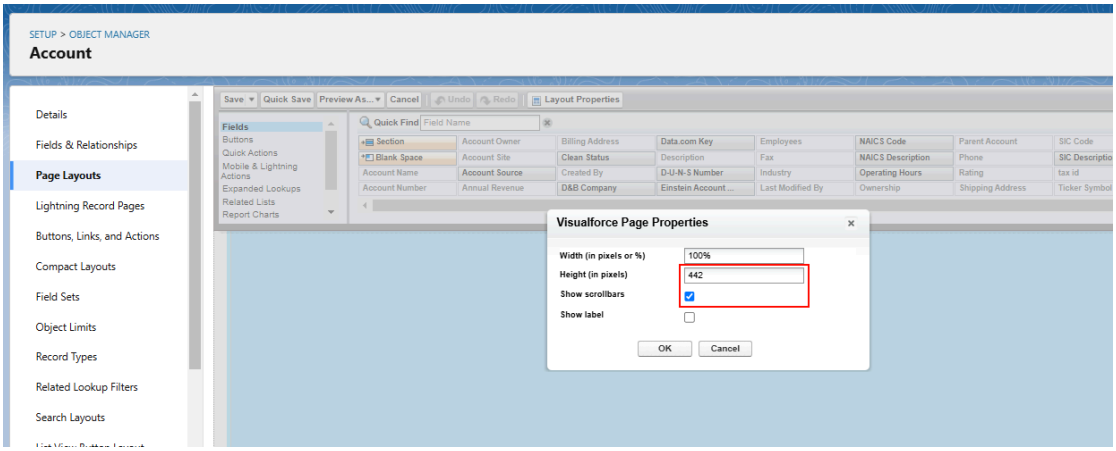
- Drag the appropriate Visualforce Page component from the palette onto the desired section of the page layout.

  For Account layouts, choose “Account Events List View Page”.
  For Contact layouts, choose “Contact Events List View Page”.

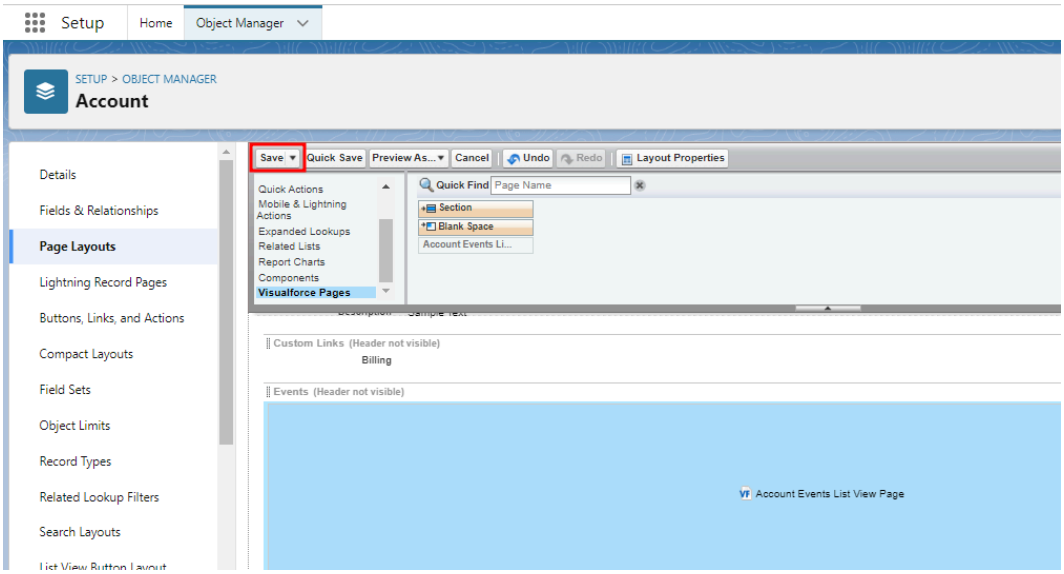


- After placing the Visualforce page component on the layout, click on the component to configure it. Replace *Height (in pixels)* setting with **442**. Select the

Show Scrollbars checkbox. Click OK.



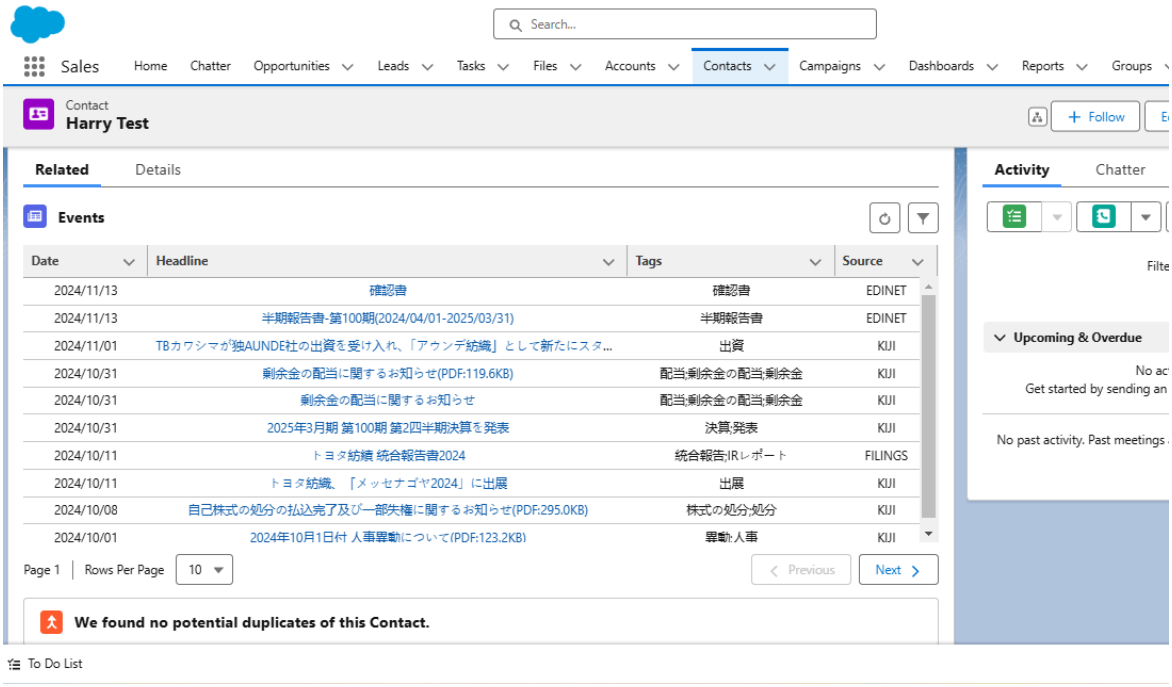
- Save the Layout: After configuring the Visualforce page, click **Save** to apply the changes to the page layout.



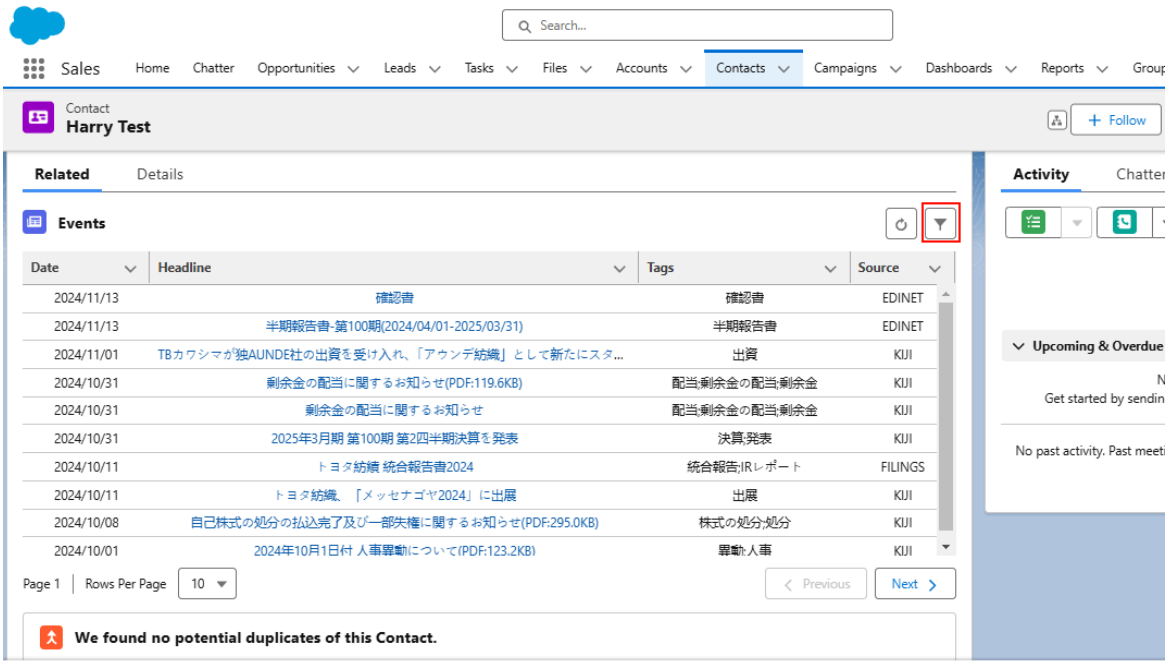
5. Usage Guide

- **Navigating to the Component**:
  - Users can access the **Event List View** component directly from the Account or Contact record pages where it has been added. Simply navigate to one of these records, and the component will be visible in the designated section of the page layout.
  - To use this component on a **Contact** record, ensure that the Contact is associated with a **Parent Account**. The component relies on the Account information to retrieve and display relevant events from an external system.

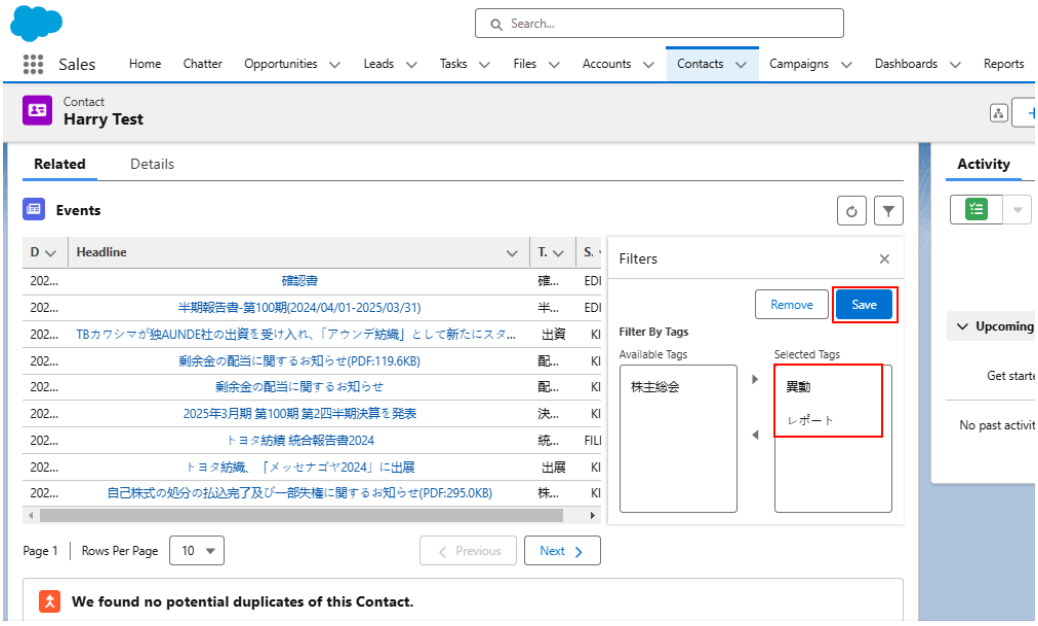
- The component displays a list of events associated with the selected Account or Contact (linked to an Account).



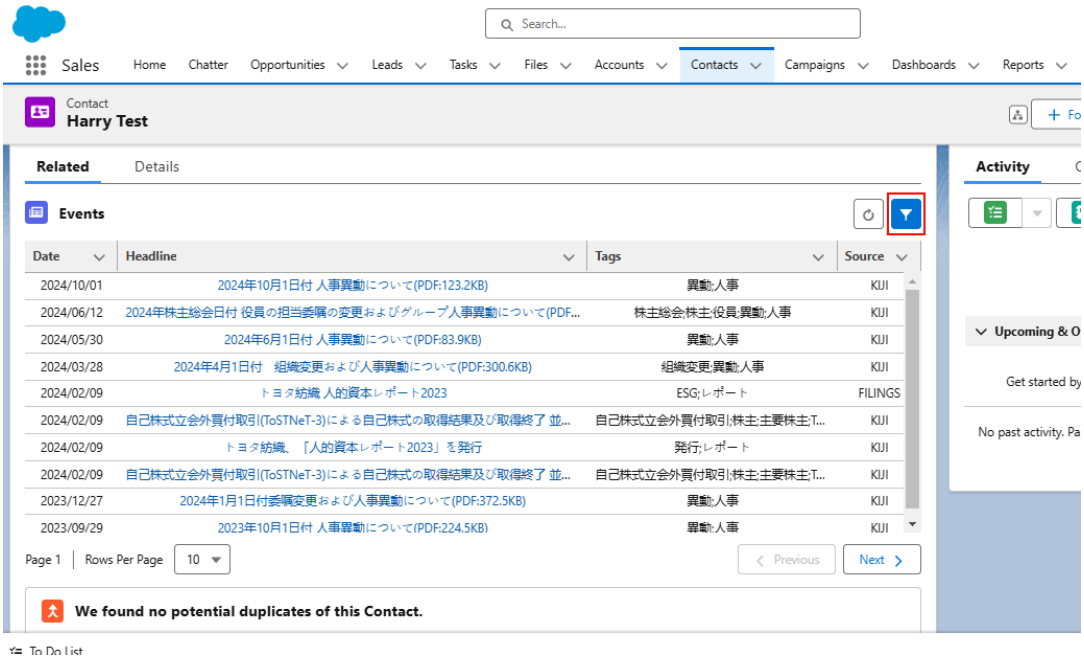
- **Using the Event Filter**:
  - **View Filter Panel**: To open the filter panel, click on the Filter icon-button. To open the filter panel, click on the Filter button. This will allow you to apply filters to the displayed events.



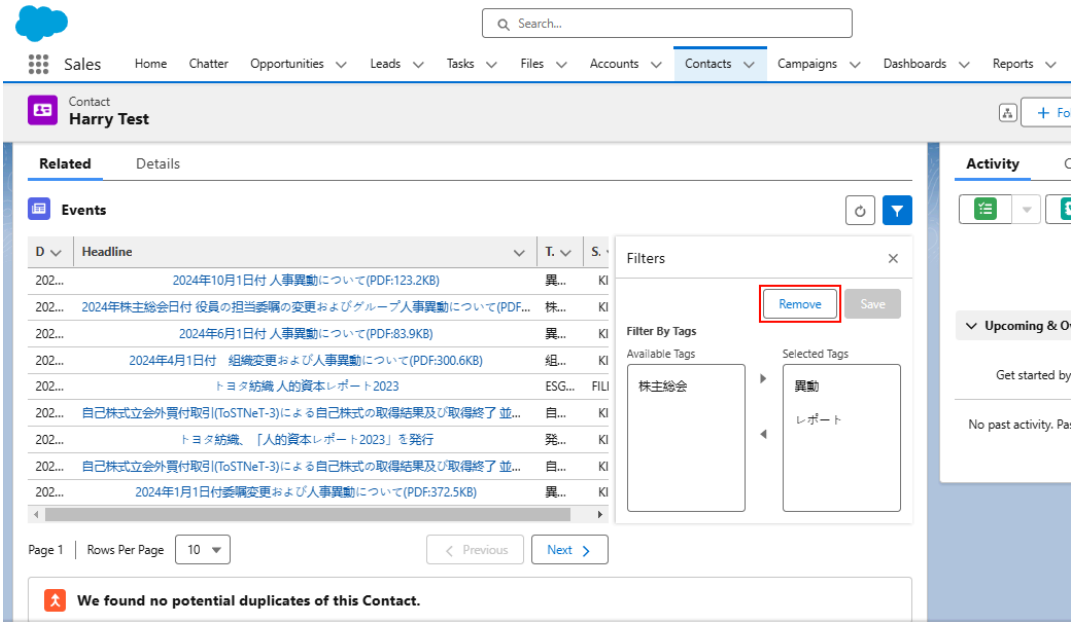
- **Tag Filters**: Currently, the filter feature supports filtering by tags.
  - **Activate Filter**: After selecting the desired tags, click the **Save** button to apply the filter and view the filtered events.



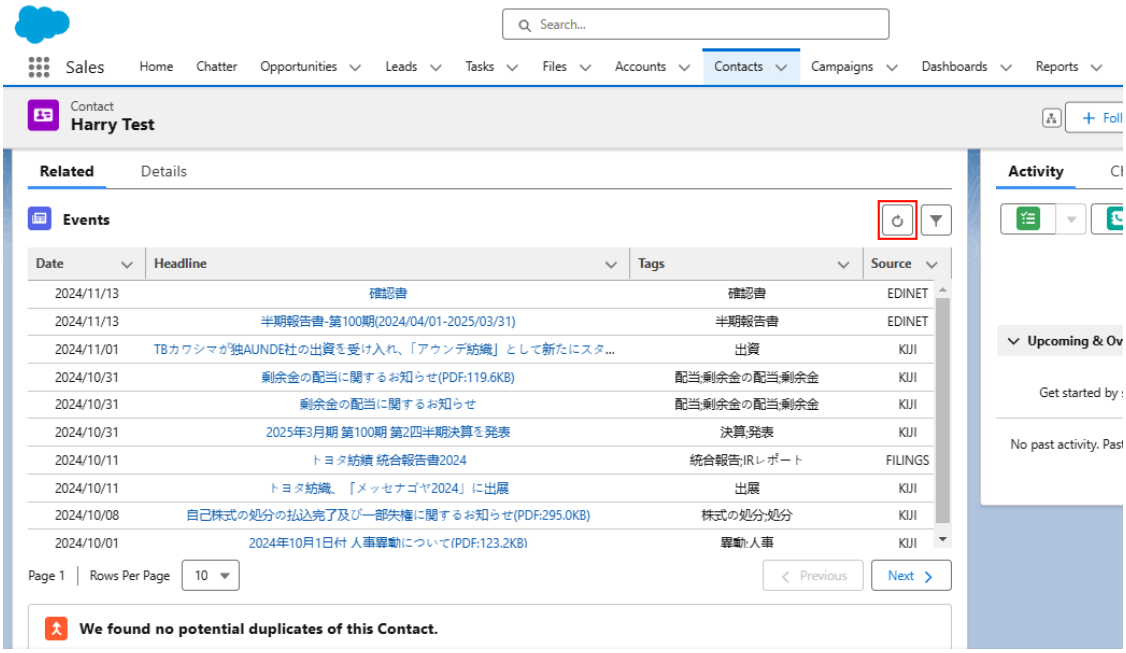
  - **Filter Icon Indicator:** When a filter is applied, the filter icon in the component will change to a branded color to indicate that a filter is currently active.



- **Remove Filter**: Use the **Remove** button to clear the filters and display the full list of events again.



- **Refresh Events**:
  - The component includes a **Refresh** button that allows users to manually refresh the event list. Clicking this button will reload the event data from the external system, ensuring that the latest events are displayed.
  - If a filter is applied, refreshing will reload the event data with the current filters still applied. This ensures that you continue to see the filtered events, reflecting any updates from the external system.



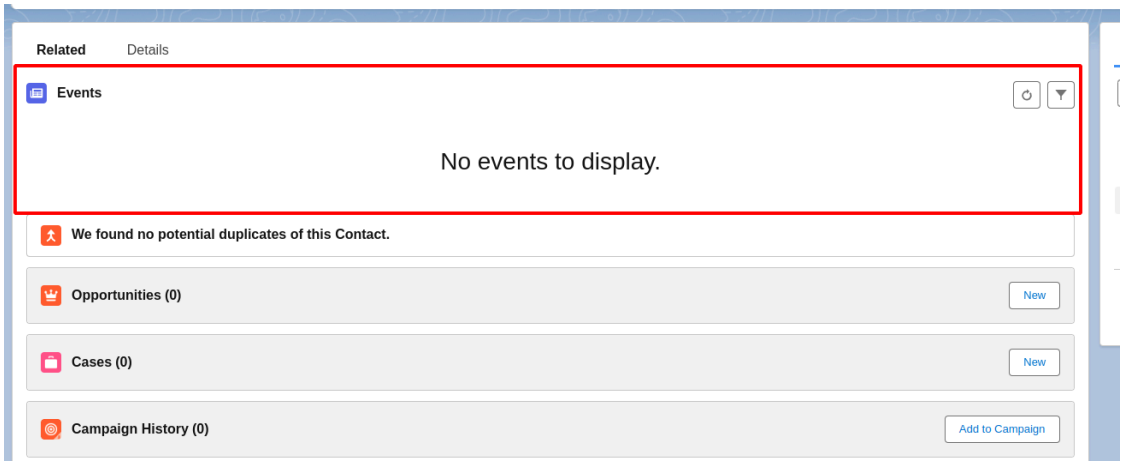
Best Practices

- **Maximize Use of Tag Filters**:
  - Encourage users to regularly use the tag filters to quickly access the most relevant events.
  - Remember to click **Save** on the filter panel to apply filters, monitor the brand-colored filter icon for active filters, and use **Remove** to clear them.
- **Ensure Contact-Account Association**:
  - Always associate Contacts with a Parent Account to ensure that events are properly retrieved and displayed. Without this association, the component cannot fetch event data.
- **Regularly Refresh Event Data**:
  - Use the **Refresh** button regularly to ensure that you are viewing the most up-to-date event information from the external system. If a filter is active, the refreshed data will respect the current filter settings.
  - Event data also refreshes automatically when the page is reloaded.

6. Troubleshooting

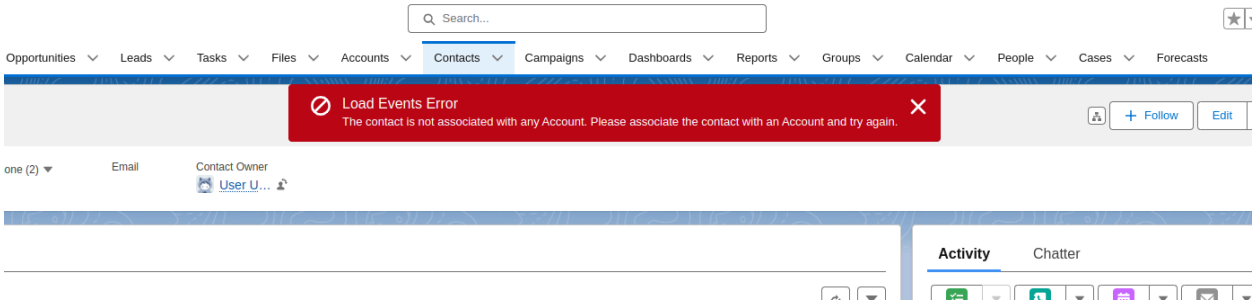
Common Issues

- **No Events Displayed**

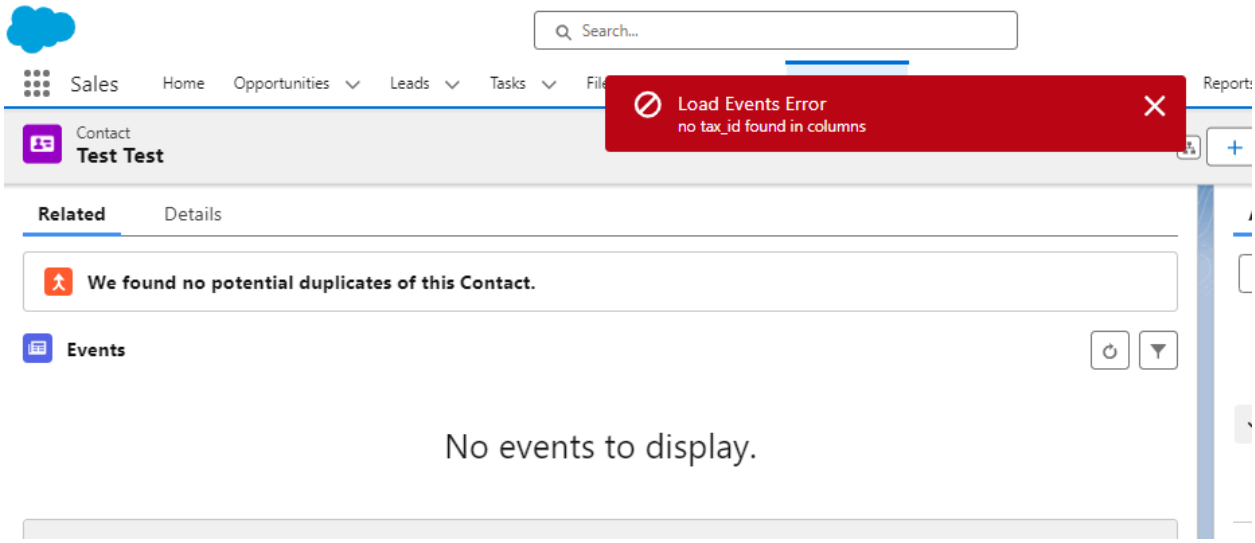


  - **Description**: The component is not showing any events.
  - **Solution**:
    1. Verify that the Contact is correctly associated with a Parent Account.
    2. Ensure that the Account has events linked to it in the external system.
    3. Check your filter settings, they may be too restrictive.
    4. Review the **Log** records for detailed error information if needed (admin access required).

- **The contact is not associated with any Account. Please associate the contact with an Account and try again.**

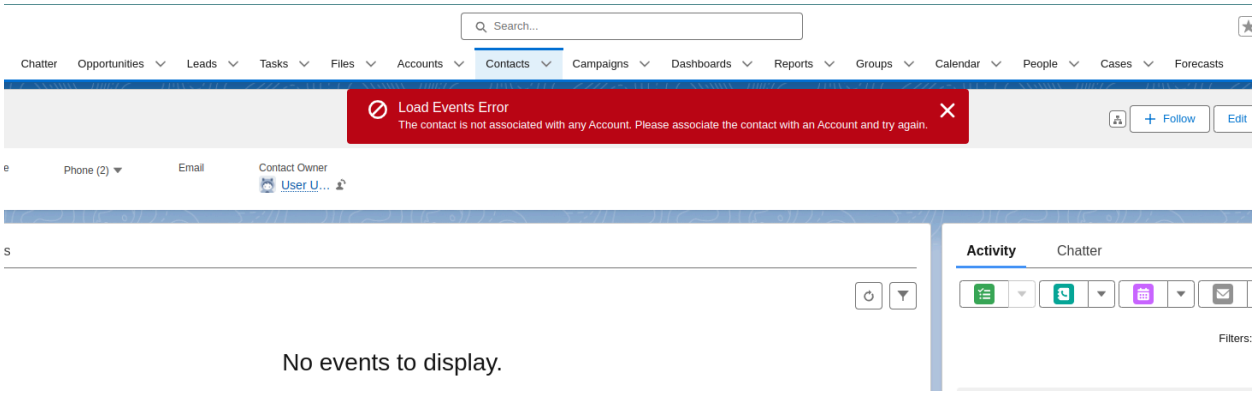


  - **Description:** This error occurs when a contact is not linked to any account, which is necessary for retrieving events.
  - **Solution**:
    1. Ensure that the contact is properly associated with a parent account.
    2. Update the contact's record to link it with an account and try again.
    3. If the issue persists, consult the **Log** object for more detailed error diagnostics (admin access required).
- **No tax_Id found in the columns.**



  - **Description:** This error occurs when a request is made from an account or a contact associated with an account that lacks the required tax_id field.
  - **Solution**:
    - Ensure that the tax_id field is correctly added to the Datazora Account Fields Payload Custom Setting.
    - Verify that the tax_id field exists on the Account object, and ensure it is populated with a valid value on the associated account record.
    - Update the tax id on the account record accordingly and refresh the events.
    - If the issue persists, check the **Log** object for more detailed error diagnostics (admin access required).

- **Error Toasts with Various Messages**



  - **Description**: Users may encounter error toasts with different messages indicating issues such as access errors, system errors, or code errors.
  - **Solution**: Review the error message for specific details about the issue. Common solutions include checking user permissions, verifying API access, or reviewing the code for any bugs. If the issue persists, consult the Log object for more details.

Error Tracking

For detailed tracking and analysis of errors encountered with this component, use the **Log** custom object. This object is available to Salesforce administrators and allows for a detailed view of error records.

- **How to Use the Log Object**:
  1. **View Errors**: Navigate to the **Log** object records in Salesforce to review error entries.
  2. **Analyze Issues**: Examine error records to understand the nature of the issues encountered. Look for specific error messages or codes that can help identify the problem.
  3. **Resolve Issues**: Use the information from the Log to address and resolve issues. If further assistance is required, include details from the Log when contacting support.

7. Updates and Release Notes

Version 1.0

- **Initial Release**: This is the first version of the package. It includes the foundational features for managing and displaying events, along with basic filtering functionality.