=PDFxFill User Guide

📄PDFxFill – Advanced PDF Form Filling and Text replacement Engine for Salesforce

Welcome to the PDFxFill User Guide. This comprehensive guide will help you understand and effectively use the PDFxFill package for template-based PDF filling in Salesforce.

Table of Contents

1. Introduction

PDFxFill is a Salesforce package that enables users to fill and replace text in PDF documents using Salesforce record data. It allows administrators to create reusable PDF templates and map Salesforce fields to specific positions within the PDF. Users can get the filled documents directly from records, automation flows, or custom integrations. Generated PDFs are automatically saved in Salesforce Files for easy access and sharing.

2. Installation

Follow the steps below to install and configure the PDFxFill: Form-Fill package in your Salesforce organization.

Step 1: Install the PDFxFill: Form-Fill Package

1. Obtain the **PDFxFill Form-Fill Managed Package installation link**.
2. Open the installation link in your browser.
3. Login to your **Salesforce organization**.
4. The **Package Installation Page** will appear.
5. Review the package components.
6. Click **Install**.

Choose one of the following installation options:

- Install for Admins Only (Recommended)
- Install for All Users
- Install for Specific Profiles

7. Click **Install**.
8. Wait for the installation to be completed.

Once installation is finished, the **PDFxFill application and components** will be available in your Salesforce org.

Step 2: Open PDFxFill Application

3. Post Installation

To start using the PDF filling features, you must first configure External (Named) Credentials, register your organization, and generate an API key.

The API key must be added to the External Credential to enable secure communication between Salesforce and the PDFxFill PDF filling service.

Step 1: Permission Set for External Named Credentials Access

Important: External Named Credentials contain sensitive authentication information and require specific permissions to view or edit. The standard PDFxFill_Admin permission set does not include External Named Credentials management permissions. You must create a separate permission set and assign it to administrators and users who need to configure the API key in External Named Credentials.

Why This Permission Set is Required

External Named Credentials store authentication details (API keys, tokens, etc.) for external services.

Salesforce restricts access to these sensitive configurations

Only users with "Manage Named Credentials" and "External Credential Principal Access" permissions can view or edit them

This ensures security by limiting who can modify external service credentials

Step-by-Step Guide to Create External Named Credentials Permission Set

Step 1: Navigate to Permission Sets

1. Log in to Salesforce as a System Administrator
2. Click on the Setup gear icon (⚙) in the top right corner
3. In the Quick Find search box, type **"Permission Sets"**
4. Click on Permission Sets under Users

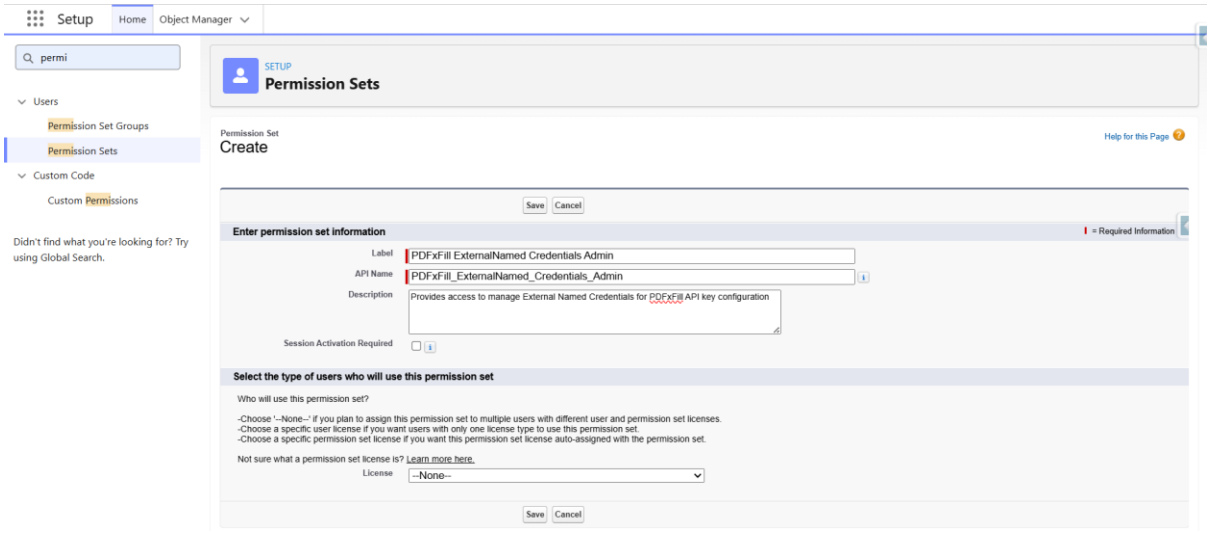
Step 2: Create New Permission Set

Click the New Permission Set button

Fill in the permission set details:

- **Label:** PDFxFill External Named Credentials Admin
- **API Name:** PDFxFill_External_Named_Credentials_Admin (auto populated)
- **Description:** Provides access to manage External Named Credentials for PDFxFill API key configuration.
- **License:** None

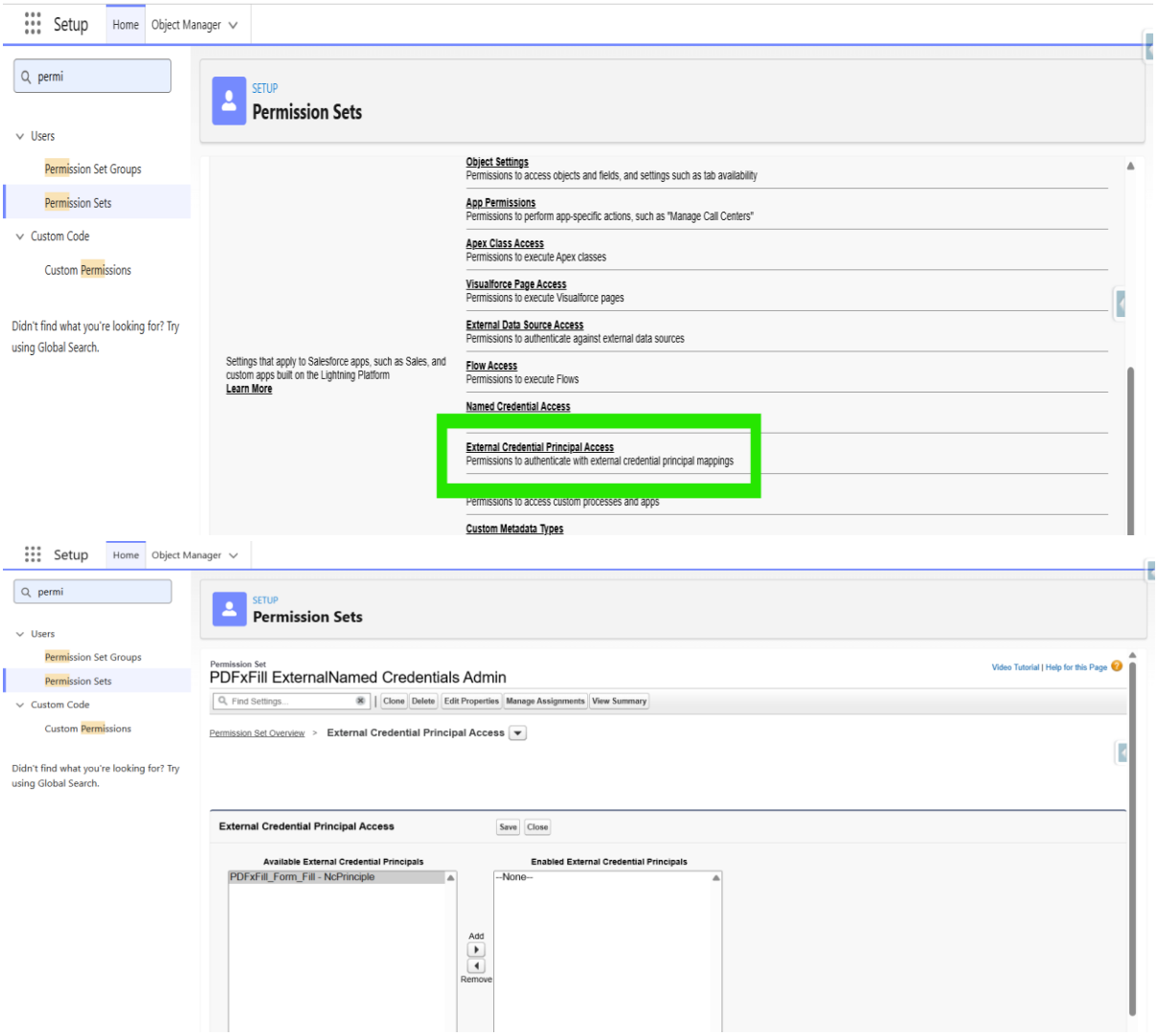
Click **Save**



Step 3: Add External Credential Principal

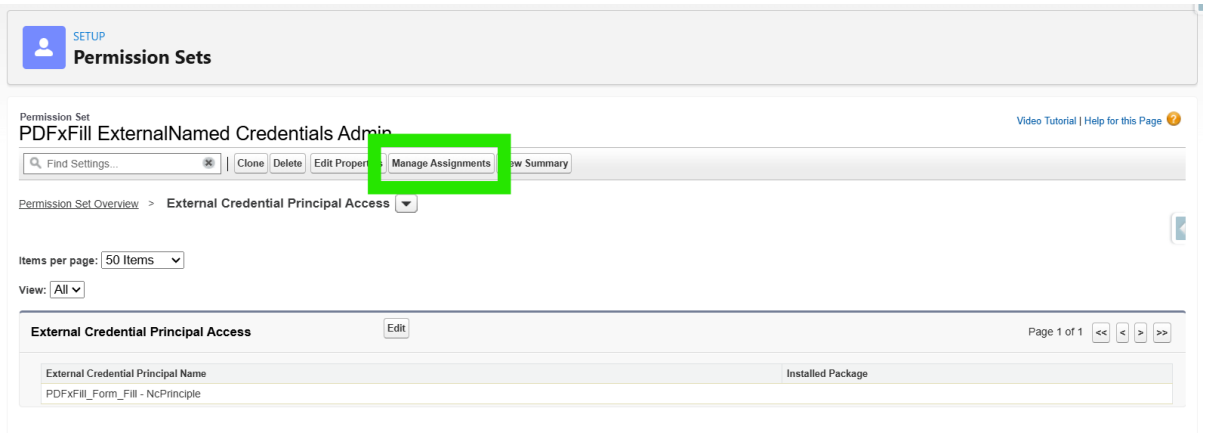
1. Click **Edit**
2. In **Available External Credential Principals**
   - Select: PDFxFill_Form_Fill - NcPrinciple
3. Click **Add →**
4. Move it to **Enabled External Credential Principals**

5. Click **Save**



Step 4: Assign Permission Set to Yourself (Admin)

1. From the Permission Set detail page, click **Manage Assignments**
2. Click **Add Assignments**
3. Search for your username or email
4. Check the box next to your name
5. Click **Assign**
6. Click **Done**



Step 5: Assign Permission Set to Other Users

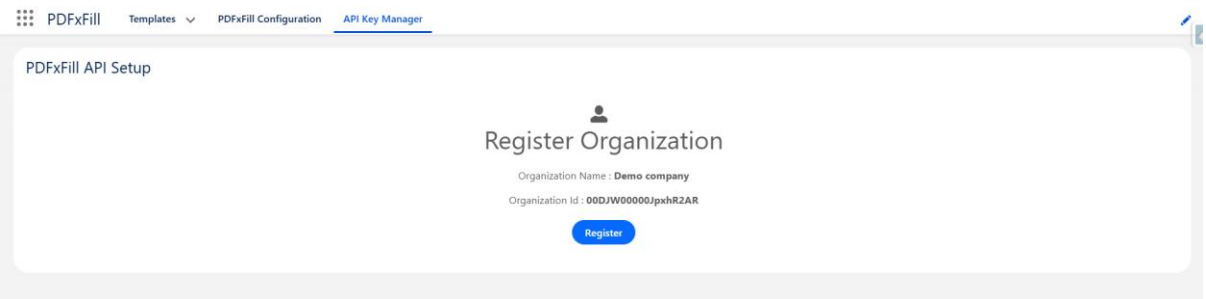
1. From the Permission Set detail page, click **Manage Assignments**
2. Click **Add Assignments**
3. Search for the users who need access (you can search by name, email, or profile)
4. Check the boxes next to the users who should have External Named Credentials access
5. Click **Assign**
6. Click **Done**

Step 2: Open API Key Manager

1. Open **App Launcher**.
2. Search for **PDFxFill**.
3. Open the **PDFxFill App**.
4. Click **API Key Manager**.

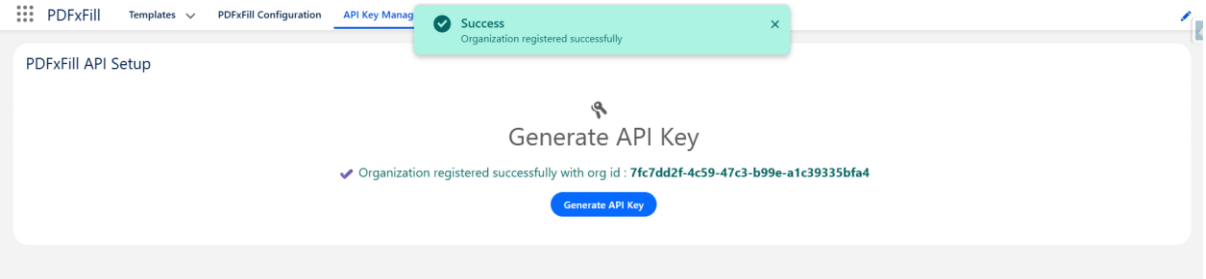
Step 3: Register Organization

1. Click **Register Organization**.



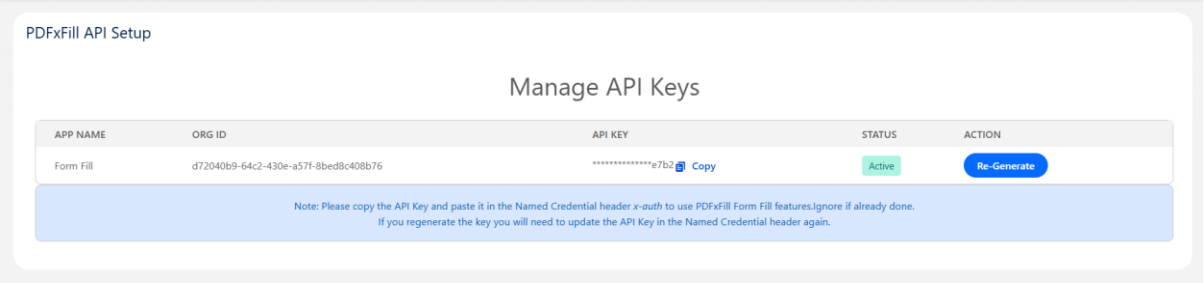
2. Salesforce sends a request to the PDFxFill service.

3. Wait for the success message.



Step 4: Generate API Key

1. Click **Generate API Key**.
2. The system generates a unique API key for the organization.
3. The generated API key will appear on the screen.



This API key is required for all PDF filling requests.

Step 5: Copy API Key and Update External Credentials

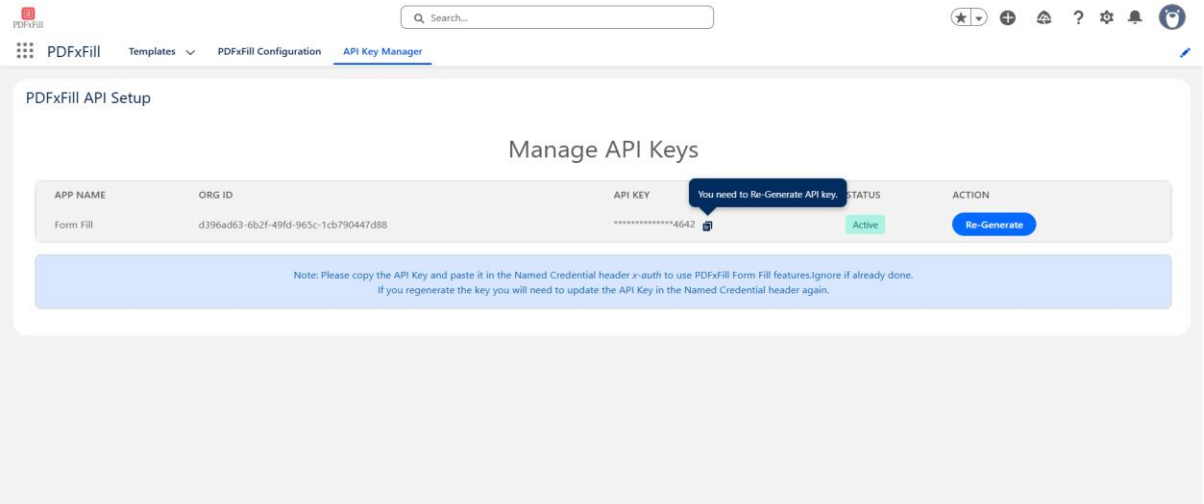
1. In the **API Key Manager**, locate the generated API key in the table.
2. Click the **Copy** button next to the masked API key
   (The full key will be copied to your clipboard.)

Important:

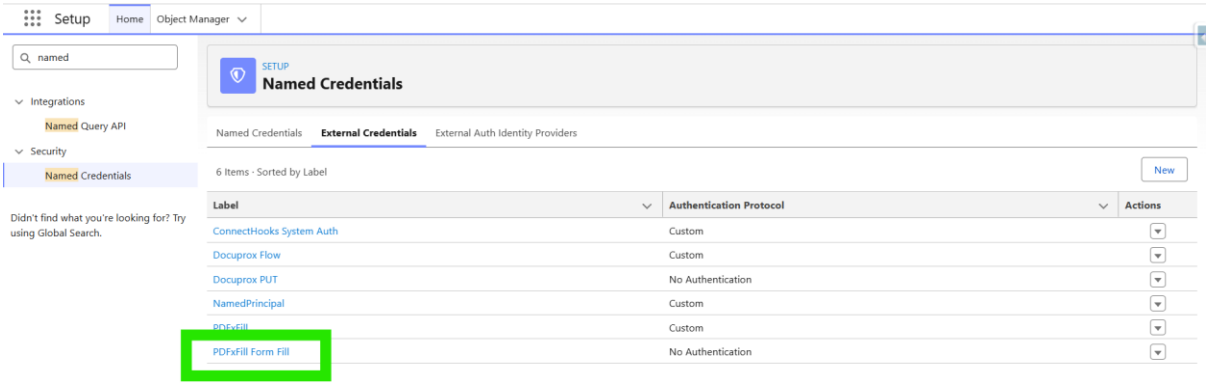
The API key can only be copied before refreshing the page. If the page is refreshed, the API key cannot be copied. When hovering over the copy button, the system displays the message:

"You need to generate a new API key."

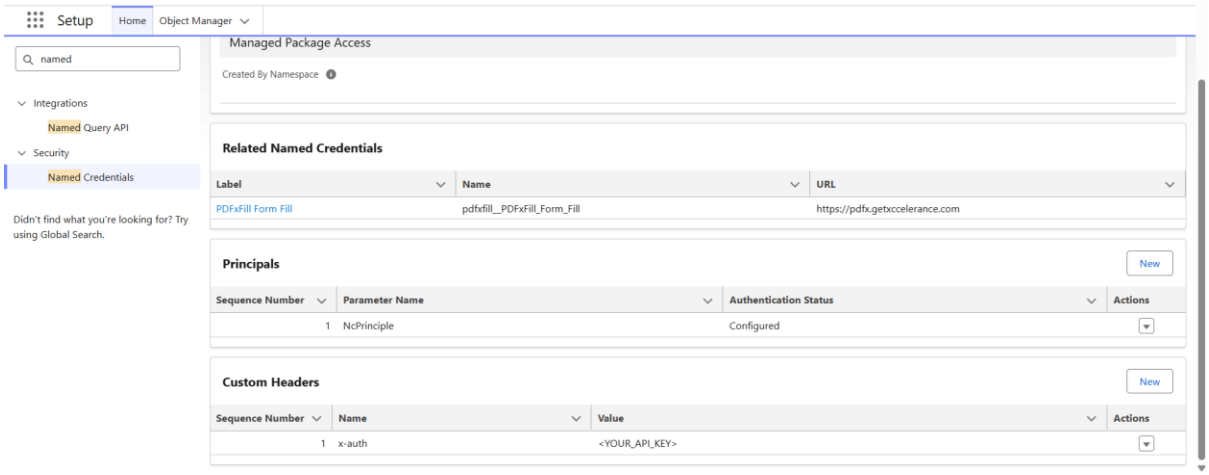
In such cases, you must click the **Re-Generate** button to generate a new API key.



3. Navigate to **Setup** in Salesforce.
4. Search for **Named Credentials** and click on it.
5. Click on the **External Credential tab**.
6. Find and open the **PDFxFill Form Fill External credential.**



7. Go to the **Authentication Parameters / Custom Headers** section and click **Edit**.
8. Locate the **`x-auth` header parameter**.
9. Replace the placeholder value **<YOUR_API_KEY>** with the copied API key.
10. Click **Save**.



Note: This API key is required for all PDF filling requests. If the API key is regenerated, you must update the value in the **External Credential** to ensure uninterrupted integration.

4. Creating Templates

Templates define the structure of the PDF document and the Salesforce object whose data will populate the PDF.

Templates can be created in two ways:

1. From the **Templates Tab**
2. From the **PDFxFill Configuration Tab**

Method 1: Create Template from Templates Tab

Step 1: Open Templates Tab

1. Open the **PDFxFill App**.
2. Click the **Templates** tab.

The Templates page displays all existing templates.

Step 2: Create a New Template

1. Click **New**.
2. The **New Template window** will appear.

Step 3: Enter Template Information

- ### Template Name

  Enter a unique template name.

  Example: Account Application Form

- ### Target Object

  Select the Salesforce object whose data will populate the PDF.

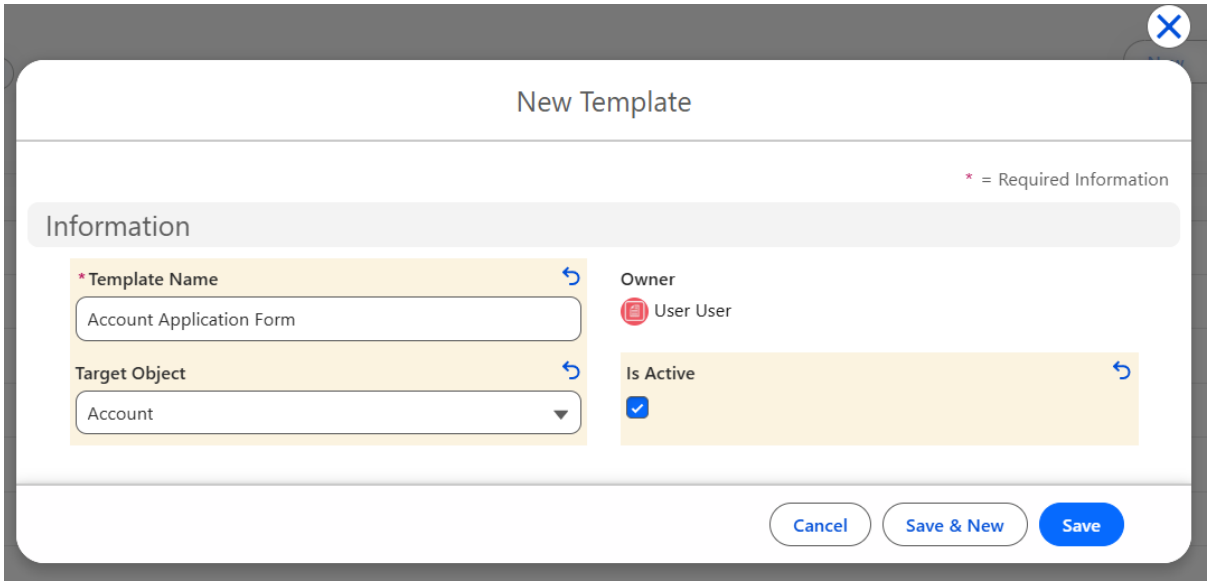
  Example: Account

- ### Owner

  Displays the user creating the template.

- ### Is Active

  If checked, the template will be available for generating PDFs.

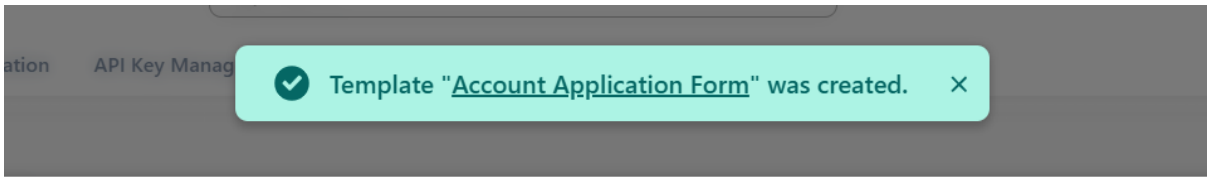


Step 4: Save Template

After entering the required details, click **Save**.

Once the template is saved, Salesforce will open the **Template Record Page**.

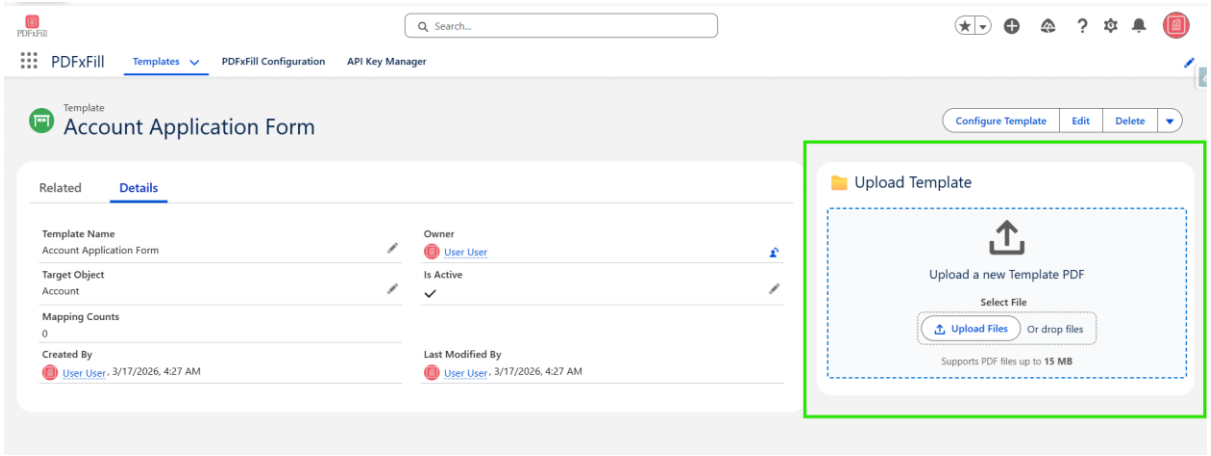
A success message will appear at the top of the screen confirming that the template has been created.



Step 5: Upload Template PDF

After saving the template, you must upload the **PDF template file** that will be used for PDF filling.

On the **Template Record Page**, you will see an **Upload Template section on the right side of the screen**.



Steps to Upload the Template

1. Navigate to the **Upload Template** section on the right side of the template record.
2. Click **Upload Files**.
3. Select the **PDF template file** from your system.
4. Click **Upload**.

You can also **drag and drop the PDF file** into the upload area.

File Requirements

- File format: **PDF**
- Maximum file size: **15 MB**

After uploading the file, the template PDF will be stored and used for **field mapping configuration**.

Next Step: Configure Template

After uploading the template PDF, click **Configure Template**.

Once you click **Configure Template**, the **PDF Mapping Interface** will open.

To continue configuring the template and map Salesforce fields to the PDF, follow the instructions in the **PDF Mapping Interface section**.

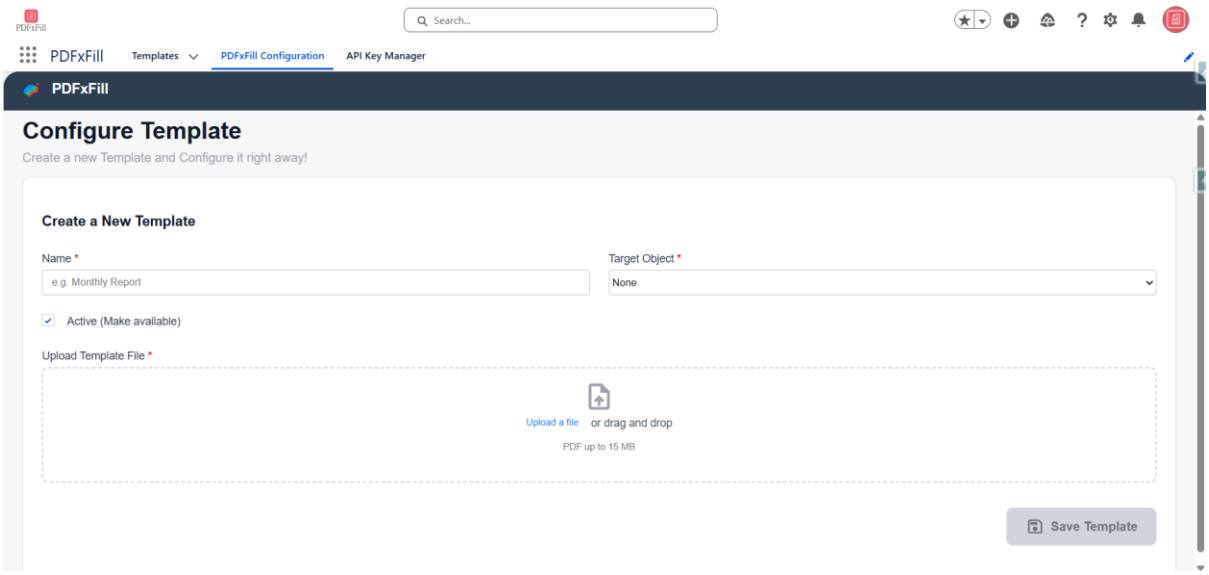
Method 2: Create Template from PDFxFill Configuration

This method allows users to create the template and configure mappings in a single workflow.

Step 1: Open PDFxFill Configuration

1. Open **PDFxFill App**.
2. Click **PDFxFill Configuration** tab.

The **Configure Template page** will appear.



Step 2: Enter Template Details

Name

Enter the template name.

Example: Home Loan Form

Target Object

Select the Salesforce object that will populate the PDF.

Active (Make available)

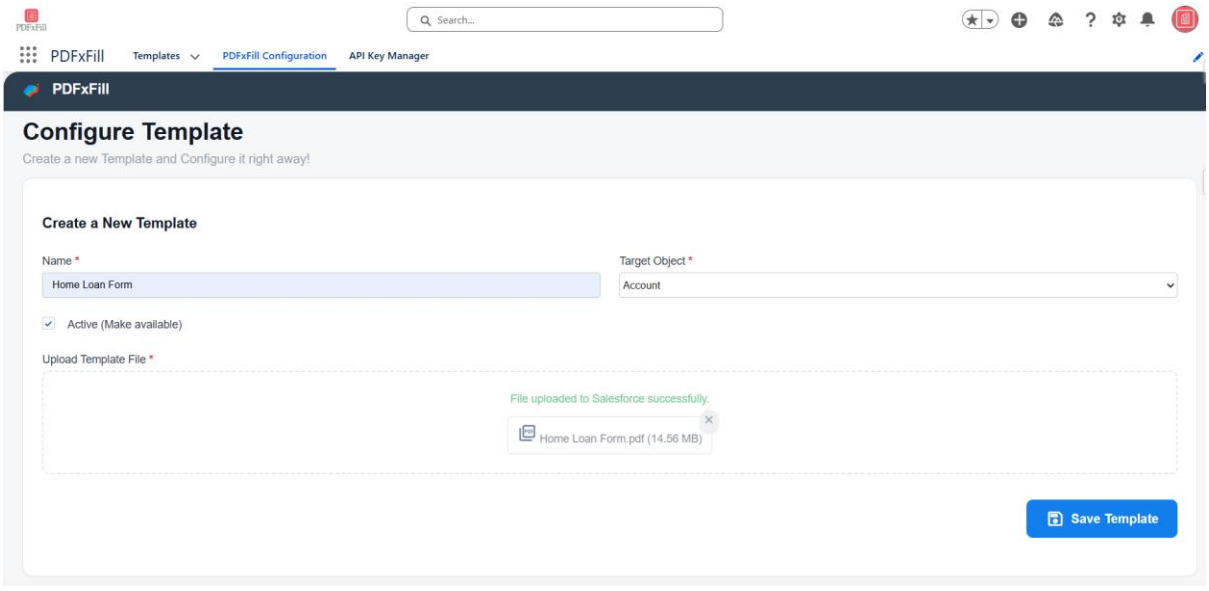
Enable this option to make the template available to users.

Step 3: Upload Template File

1. Click **Upload a file**.
2. Select the PDF template from your system.

Requirements:

- File format: **PDF**
- Maximum size: **15MB**



Step 4: Save Template

Click **Save Template**.

The system will open the **PDF Mapping Interface**.

To continue the configuration process, refer to the **PDF Mapping Interface** section and follow the next steps.

5. PDF mapping interface

After uploading the template, the system displays the **PDF mapping interface**.

The interface contains two sections:

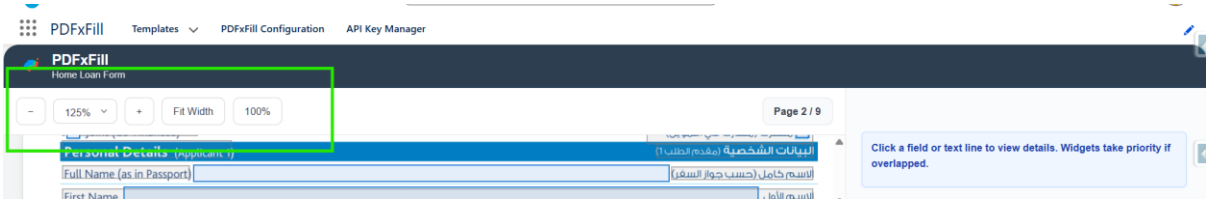
PDF Viewer (Left Panel)

The left side displays the uploaded PDF.

Users can:

- Navigate pages
- Zoom in/out
- Select fields inside the PDF

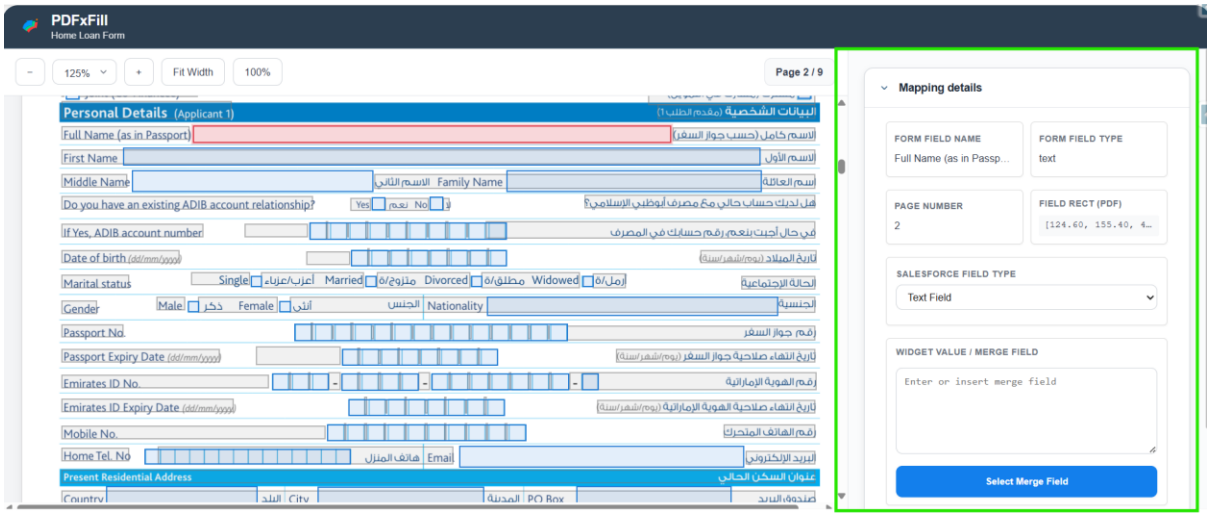
When a field is selected, the system captures its position.



Mapping Details Panel

When you select a field or text area inside the PDF template, the **Mapping Details panel** appears on the right side of the screen.

This panel allows you to configure how Salesforce data will be inserted into the selected location in the PDF.



Form Field Name

Displays the name of the field selected in the PDF.

Example:

Full Name (as in Passport)

This helps identify which PDF field is currently being configured.

Form Field Type

Displays the type of field detected in the PDF template.

Example:

Text

This information is automatically detected from the PDF.

Page Number

Displays the page number where the selected field exists in the PDF document.

This helps identify the exact page location of the field inside multi-page PDF templates.

Field Rect (PDF)

This shows the **rectangle coordinates of the selected field inside the PDF**.

These coordinates represent the exact position where the Salesforce data will be inserted during PDF filling.

The rectangle values are automatically captured when you click a field in the PDF viewer.

Users do not need to modify these values manually.

Salesforce Field Type

This option defines the **type of Salesforce data that will populate the selected PDF field**.

Select the field type based on the type of value you want to insert.

Common field types include:

- **Text Field**

  Use this when the value is being inserted into text.

  Examples: - Name - Address - Email - Account Name - City - Passport Number

  Example merge field: {!Account.Name}

- **Date**

  Use this when the value represents a **date field from Salesforce**.

  Examples: - Date of Birth - Contract Start Date - Created Date - Application Date

  Example merge field: {!Account.CreatedDate}

  The system will automatically format and insert the date into the PDF.

- **Boolean**

  Use this option for **true/false values or checkbox fields**.

  Examples: - Active - Approved - Verified

  Boolean fields are typically used when the PDF contains **checkbox-style inputs**.

  Example merge field: {!Account.IsActive__c}

  If the value is **true**, the checkbox will be marked in the generated PDF.

Widget Value / Merge Field

This field stores the **Salesforce merge field that will populate the selected PDF field**.

After selecting a field in the PDF:

1. The **Page Number** and **Field Rect values** will automatically appear.
2. You must specify which Salesforce field value should be inserted.

To do this:

1. Click **Select Merge Field**.
2. Choose the Salesforce field whose value you want to display in the PDF.
3. The merge field will be automatically inserted into the **Widget Value / Merge Field** box.

Example: {!Account.Name}

During PDF filling, this merge field will be replaced with the actual Salesforce record value.

Selecting Fields from Related Objects

The **Select Merge Field** option also allows you to access fields from related or child objects.

For example:

- Account fields
- Related Contact fields
- Custom object fields

This allows users to display **data from related records inside the generated PDF**.

Replacing Static Text in the PDF

You can also replace **existing text in the PDF template**.

To replace text:

1. Click the text element in the PDF viewer.
2. The mapping details for that text will appear on the right side.
3. In the **Text Value / Merge Field**, enter the new text value or merge field.

Example:

Original PDF text: Home Finance Application

You can replace it with: Loan Application Form

or use a merge field such as: {!Account.Name}

4. Click **Save Mapping**.

   The selected text will be replaced with the specified value when the PDF is generated.

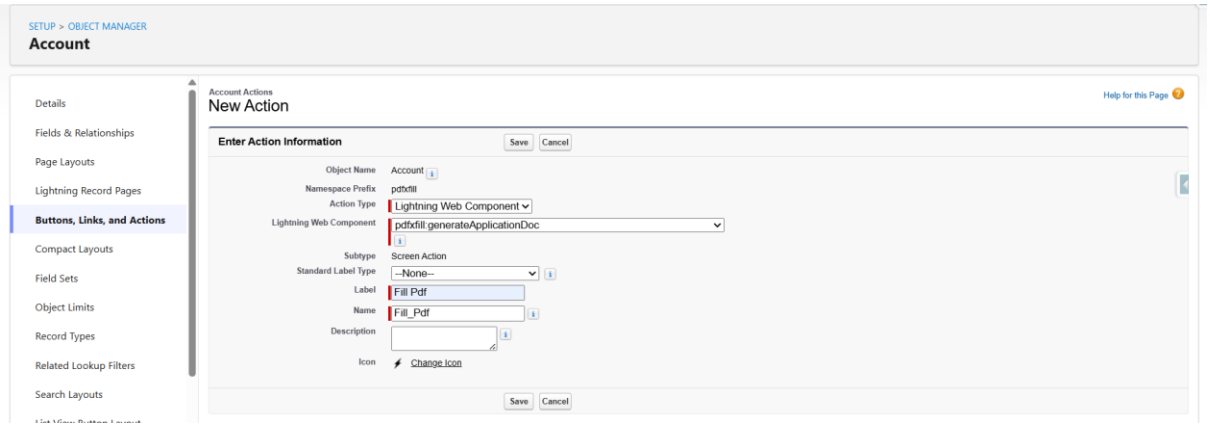
6. Filling PDF from Salesforce Record

To allow users to generate PDFs from records, create a **Quick Action button**.

Step 1: Create Fill PDF Button

1. Go to **Setup**
2. Open **Object Manager**
3. Select your object (Example: Account)
4. Click **Buttons, Links, and Actions**
5. Click **New Action**

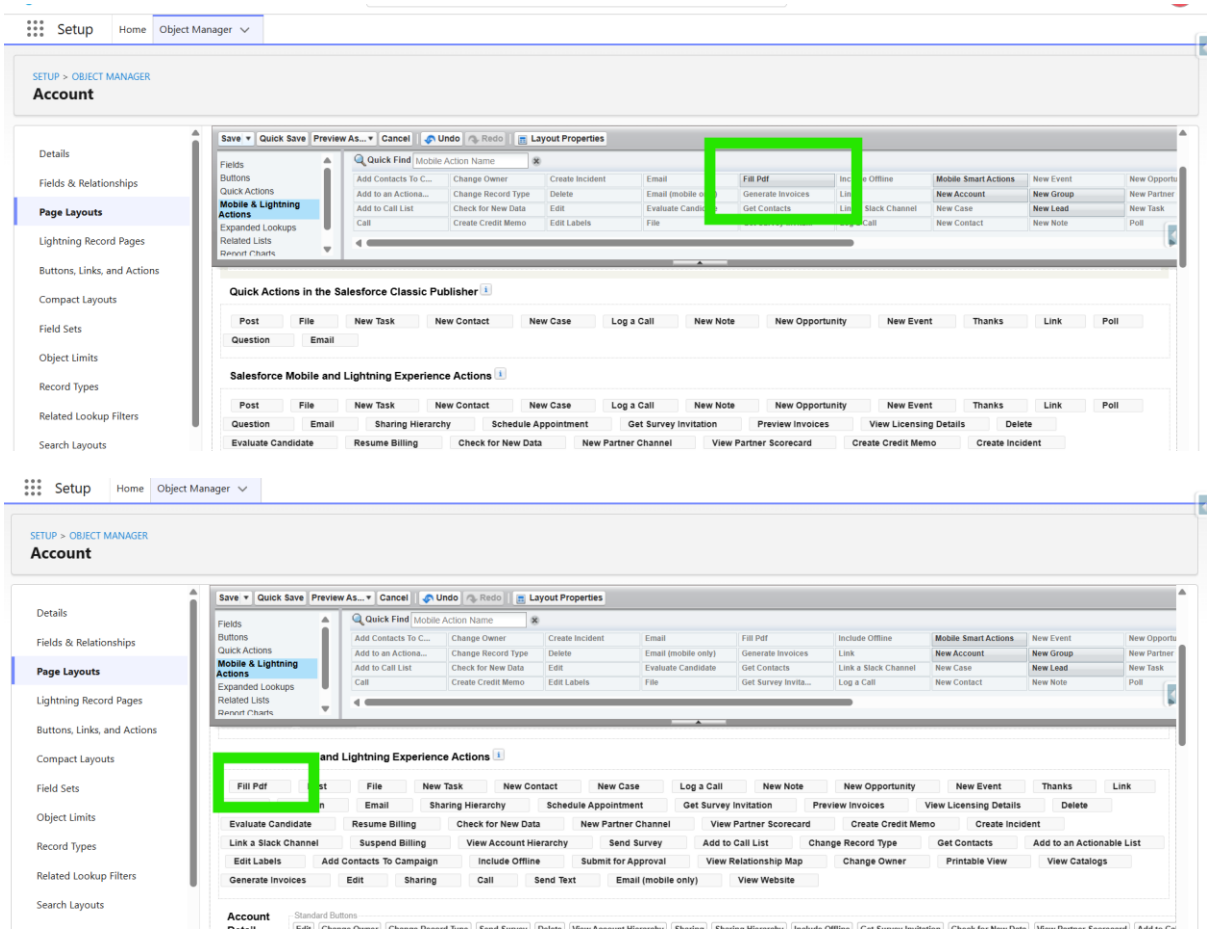
Configure:



Field	Value
Action Type	Lightning Web Component
Lightning Web Component	Pdfxfill:generateApplicationDoc
Subtype	Screen Action
Label	<YOUR_BUTTON_NAME>

Click **Save**

Step 2: Add Button to Page Layout

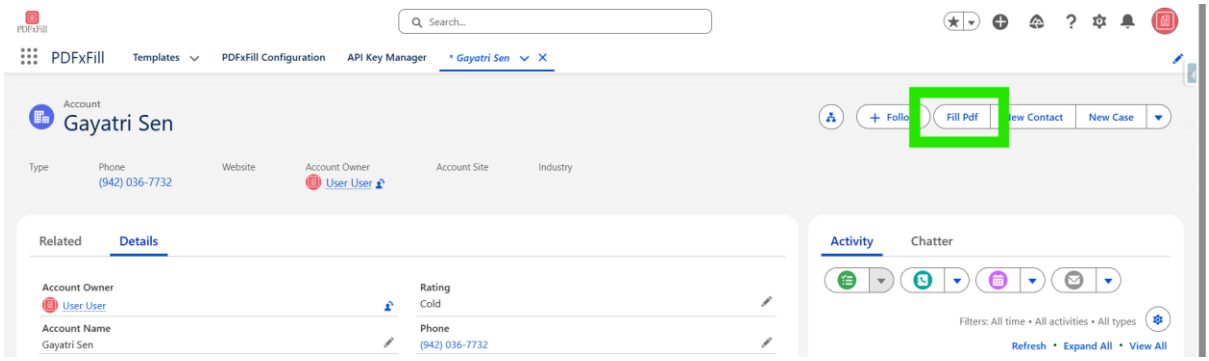


1. Go to **Page Layouts**
2. Edit the layout
3. Drag **Fill PDF** action to Salesforce Mobile and Lightning Experience Actions

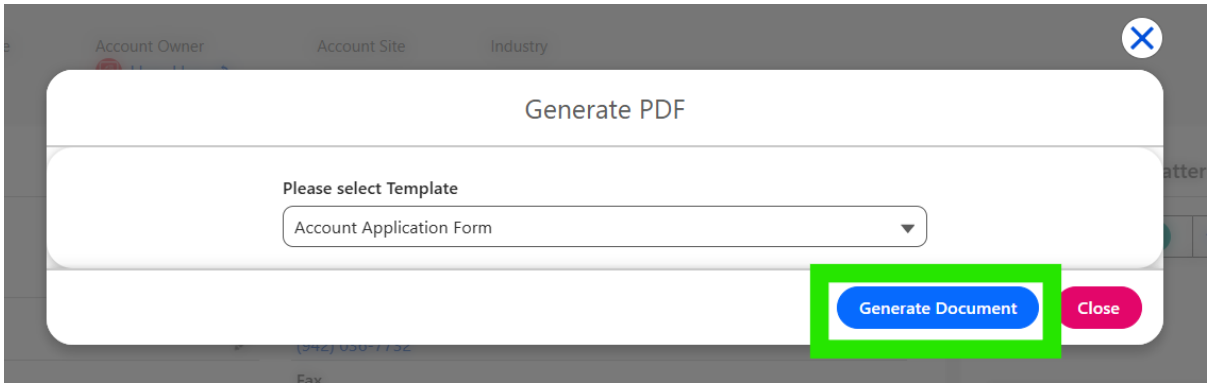
Click **Save**

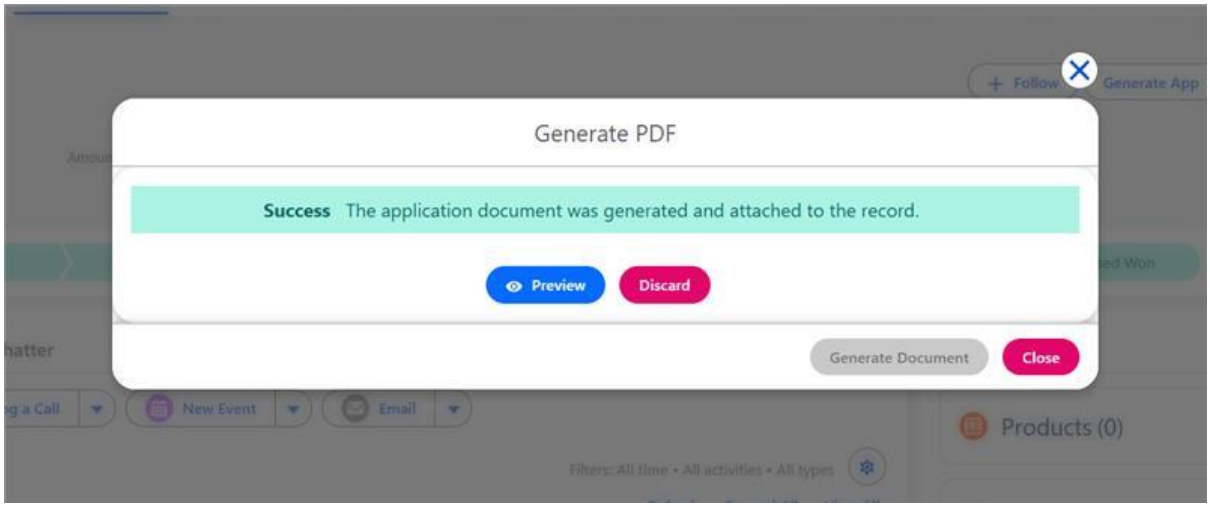
Step 3: Generate PDF

1. Open a record (Account / Opportunity etc.)
2. Click **Fill PDF**



3. Select template (if prompted)
4. Click **Generate Document** Button.





The generated PDF will automatically be saved in: Record → Files Section

7. Global Classes in PDFxFill Package

This Table provides details on all global classes in the PDFxFill Salesforce package. These classes are designed to be accessible externally (e.g., from other packages or orgs) and provide the core functionality for PDF form filling using the PDFxFill service. Each class includes a description, global methods, parameters, return types, and usage notes.

1. PDFxFillController

Description: This is the primary controller class for the PDFxFill service. It serves as the main entry point for filling PDF forms, handling the preparation of request data and invoking the PDF fill service.

Global Methods:

- global static Id pdfFillService(Id recordId, Id templateId)
    - **Parameters**:
        - recordId (Id): The Salesforce record ID for which the PDF needs to be generated.
        - templateId (Id): The ID of the PDF template used for generation.
    - **Return Type**: Id - The ID of the inserted ContentDocument (the filled PDF).
    - **Description**: Central method that prepares JSON payload and calls the PDFxFill service. Automatically attaches the filled PDF to the record.

Usage Example:

```
Id contentDocId = PDFxFillController.pdfFillService('001XXXXXXXXXXXXX', '00DXXXXXXXXXXXXX');
```

2. PDFxFillService

Description: This is the primary service class for making API callouts to the PDFxFill service. It handles the HTTP requests and response processing for PDF generation.

Global Methods:

- global static Id fillPDF(String mapping, Id recordId, Id templateId, Boolean attachToRecord)
    - **Parameters**:

- ♣ mapping (String): The JSON string representing the data mapping for PDF fields.
- ♣ recordId (Id): The Salesforce record ID.
- ♣ templateId (Id): The ID of the PDF template.
- ♣ attachToRecord (Boolean): Whether to attach the filled PDF to the record.

- **Return Type**: **Id** - The ID of the inserted ContentDocumentLink.
- **Description**: Makes the API callout to PDFxFill, processes the response, and optionally attaches the PDF to the record. Includes logging for debugging.

Usage Example:

```
String jsonMapping = '{"field1": "value1"}';
Id contentDocId = PDFxFillService.fillPDF(jsonMapping, '001XXXXXXXXXXXXX', '00DXXXXXXXXXXXXX', true);
```

3.PDFxFillJSONBuilder

Description: This class is responsible for building the JSON payload required for the PDFxFill service callout. It prepares data mappings from Salesforce records and templates.

Global Methods:

- global static String prepareJSON(Id recordId, Id templateId)
  - **Parameters:**
    - recordId (Id): The Salesforce record ID.
    - templateId (Id): The ID of the PDF template.
  - **Return Type**: **String** - The serialized JSON string for the PDFxFill callout.
  - **Description:** Prepares and serializes the JSON payload based on record and template data. Returns an empty string if mapping fails.

Usage Example:

```
String jsonPayload = PDFxFillJSONBuilder.prepareJSON('001XXXXXXXXXXXXX', '00DXXXXXXXXXXXXX');
```

4.PDFxFillJSONValidator

Description: This class validates JSON data against a predefined JSON schema stored in Salesforce Static Resources. It ensures the data conforms to the expected structure before sending to PDFxFill.

Global Methods:

- global static Boolean validateJsonAgainstSchema(String jsonString)
  - **Parameters**:
    - ♣ jsonString (String): The JSON string to validate.
  - **Return Type**: **Id** - Boolean - True if valid, False otherwise.
  - **Description**: Validates the JSON string against the default schema from custom settings.

- global static Boolean validateJsonAgainstSchema(List<Object> data)
  - **Parameters**:
    - ♣ data (List<Object>): The data list to validate.
  - **Return Type**: **Id** - Boolean - True if valid, False otherwise.
  - **Description**: Validates the data list against the default schema.

- global static Boolean validateJsonAgainstSchema(String jsonString, String schemaJson)
  - **Parameters**:
    - ♣ jsonString (String): The JSON string to validate.
    - ♣ schemaJson (String): The JSON schema string.
  - **Return Type**: **Id** - Boolean - True if valid, False otherwise.
  - **Description**: Validates the JSON string against a custom schema.

- global static Boolean validateJsonAgainstSchema(List<Object> data, String schemaJson)
  - **Parameters**:
    - ♣ data (List<Object>): The data list to validate.
    - ♣ schemaJson (String): The JSON schema string.
  - **Return Type**: **Id** - Boolean - True if valid, False otherwise.
  - **Description**: Validates the data list against a custom schema.

Usage Example:

```
Boolean isValid = PDFxFillJSONValidator.validateJsonAgainstSchema('{"field": "value"}');
```

8. Troubleshooting

API Key Not Generated

Possible causes:

- Organization not registered
- Callout failure
- Invalid system key

Solution:

- Re-register organization
- Check callout settings
- Review debug logs

PDF Not Generating

Possible causes:

- Template not configured
- Missing field mappings
- API service unavailable

Solution:

- Verify template configuration
- Check field mappings
- Ensure API key is active

Data Not Appearing in PDF

Possible causes:

- Incorrect field mapping
- Field permission issues
- Incorrect rectangle position

Solution:

- Verify mapping configuration
- Check field-level security
- Validate rectangle coordinates

Connect With Support Team

Email : pdfxfill.support@xccelerance.com

Phone Number : **+91-9039042301**