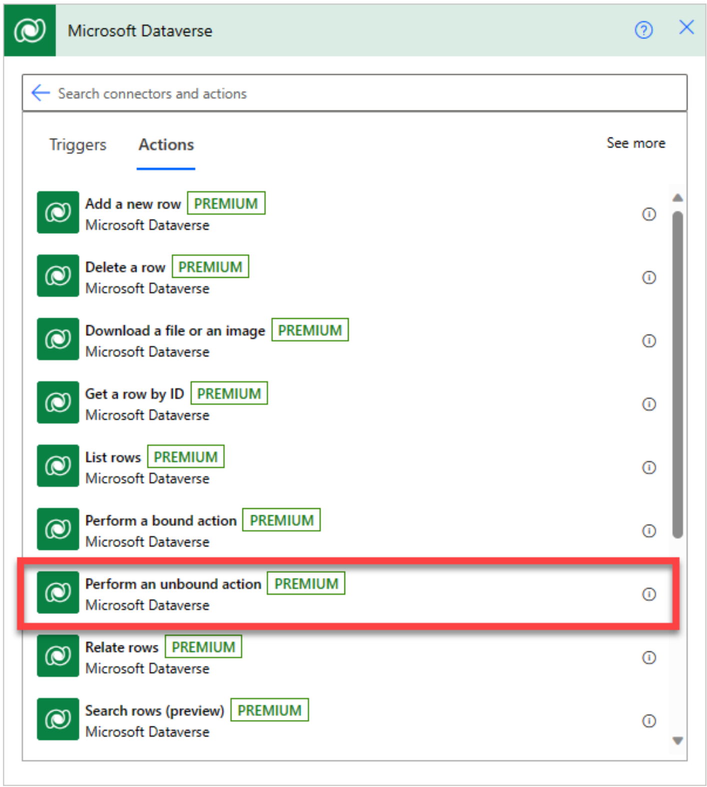
Process AI Builder form processing models with more than 300 fields in a Cloud Flow

1. Add the **Perform a bound action** from the Dataverse connector.



2. Put the following information on each field of the action.

Table name: AI Models

Action Name: Predict

Row id: The model id. (*You can get the model id form the URL in the Model Details Page in the Power Apps portal. For example: https://make.powerapps.com/environment/223537a6-254b-4fe5-910c-0fbd4e36a65d/aibuilder/models/e61cd551-8b1d-4757-9a99-80c3defe59ad. This is the model id to use*)

Version: 1.0

Request:

For a .jpeg image of the form:

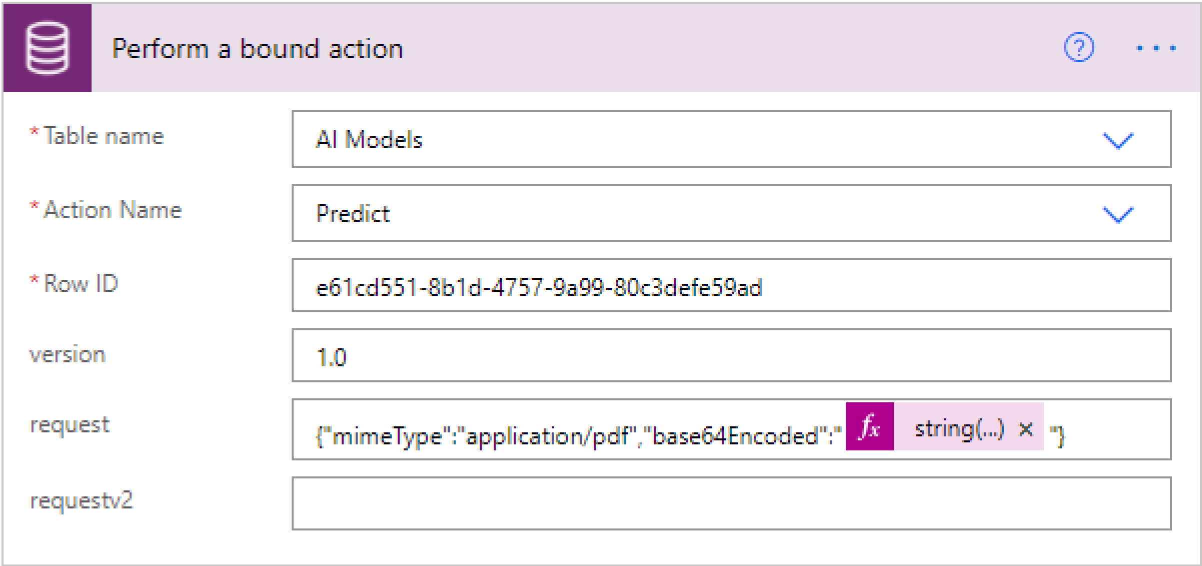
```
{
   "base64Encoded": "EXPRESSION",
   "mimeType": "image/jpeg"
}
```

For a .pdf document of the form:

```
{
   "base64Encoded": "EXPRESSION",
   "mimeType": "application/pdf"
}
```

In the case of a manual trigger, replace EXPRESSION with the following expression: string(triggerBody()?['file']?['contentBytes'])

Depending on which connector the file comes from, the expression will need to be enclosed by base64() instead of string().



Perform a bound action

* Table name	AI Models
* Action Name	Predict
* Row ID	e61cd551-8b1d-4757-9a99-80c3defe59ad
version	1.0
request	{"mimeType":"application/pdf","base64Encoded":" *fx* string(...) × "}
requestv2	

3. Select **Test** on the upper right, select **I'll perform the trigger action**, and then select Save & Test.

4. Select a document that can be processed by your trained form processing model and then select **Run flow**.

5. Copy the results from **PredictResponse response**.

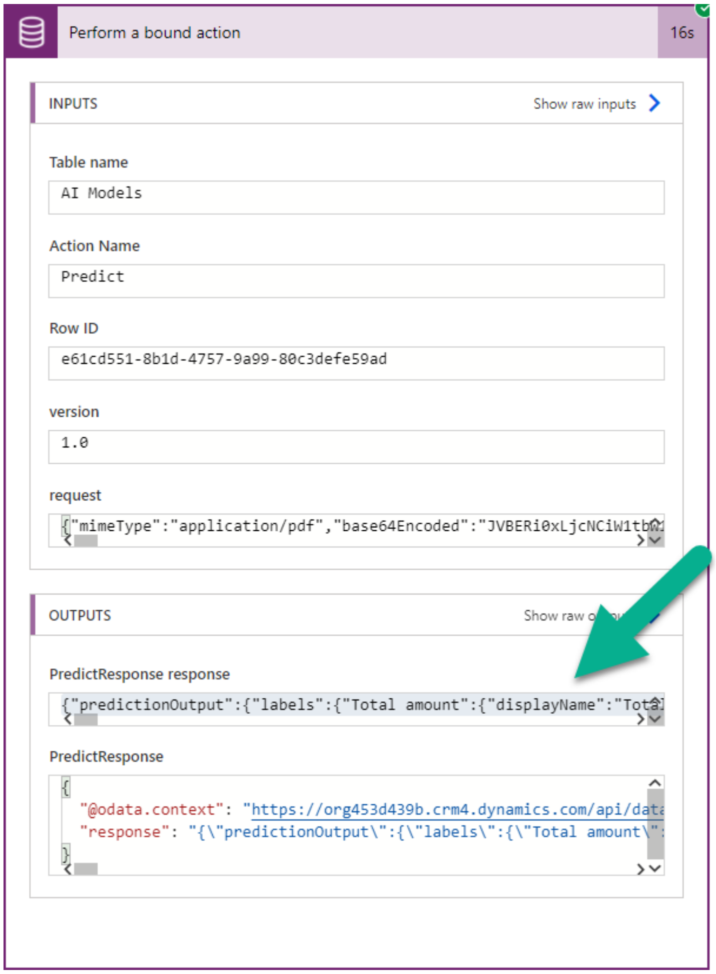


Table name

```
AI Models
```

Action Name

```
Predict
```

Row ID

```
e61cd551-8b1d-4757-9a99-80c3defe59ad
```

version

```
1.0
```

request

```
{"mimeType":"application/pdf","base64Encoded":"JVBERi0xLjcNCiW1tbW
```

PredictResponse response

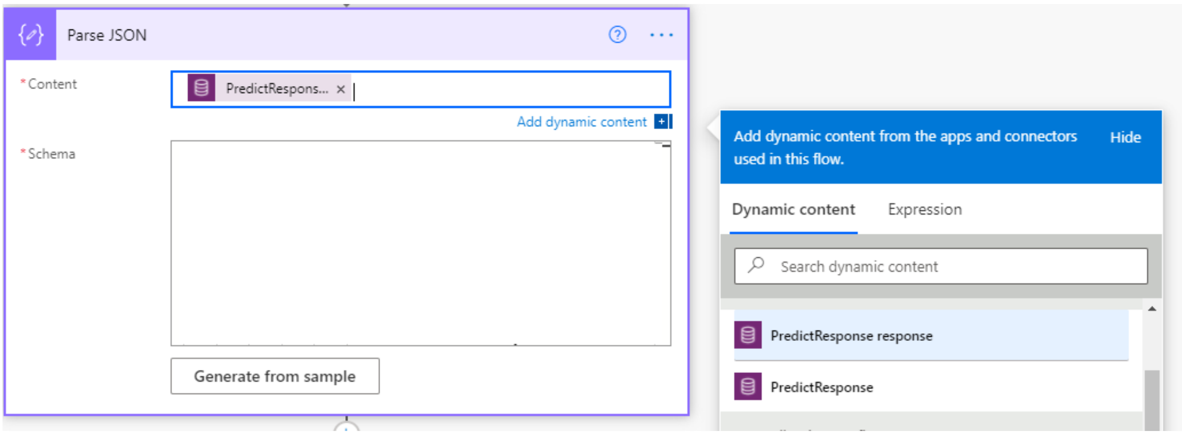
```
{"predictionOutput":{"labels":{"Total amount":{"displayName":"Total
```

PredictResponse

```
{
  "@odata.context": "https://org453d439b.crm4.dynamics.com/api/data
  "response": "{\"predictionOutput\":{\"labels\":{\"Total amount\":
}
```

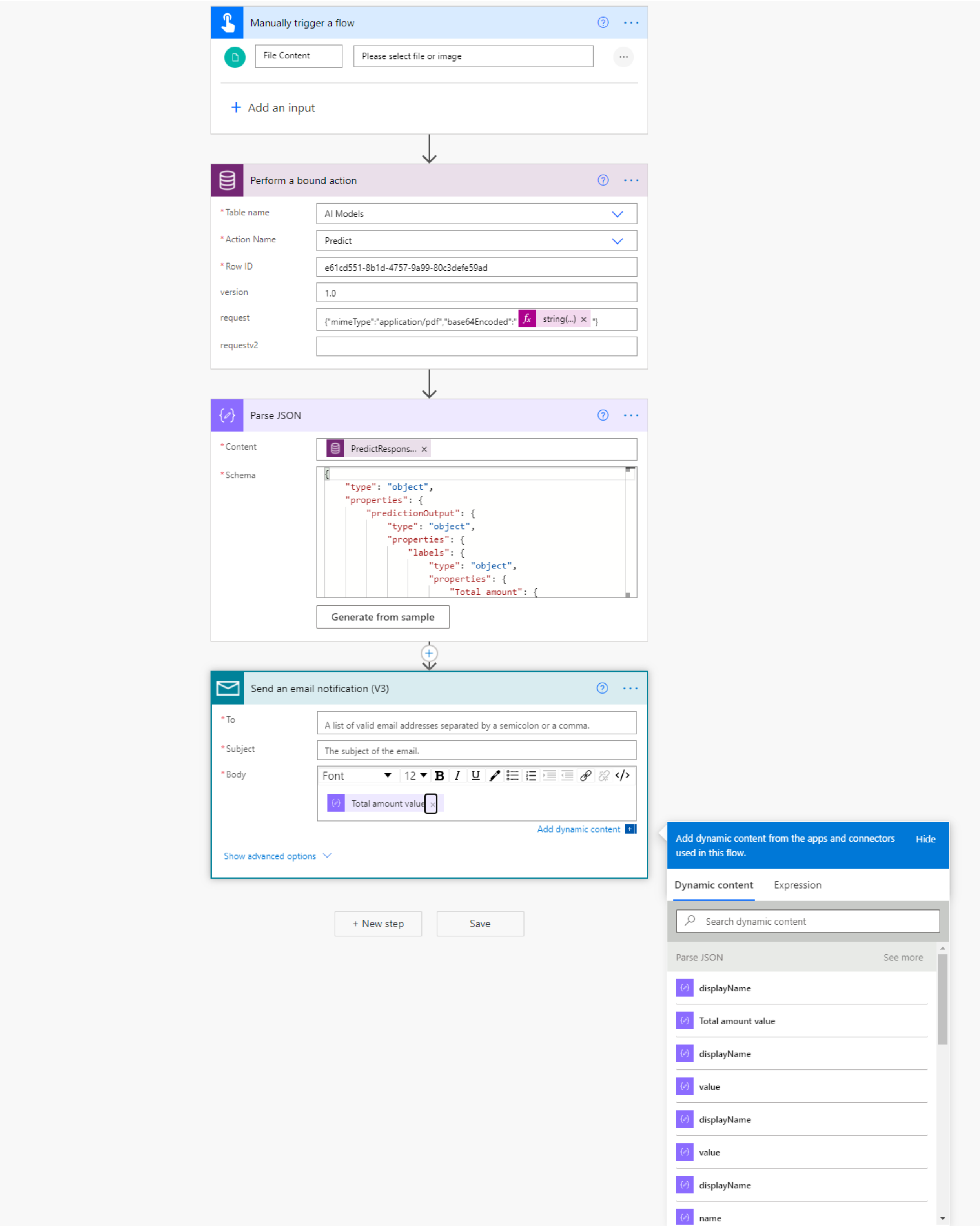
6. Go back to edit the flow editor. Select **+ New step**, search for Parse JSON, and then select **Parse JSON – Data Operations** from the list of actions.

7. In the Parse JSON action, next to **Content**, select **PredictResponse response** from the Dynamic content pane.



8. Click on **Generate from sample** and paste what you copied on step 5, and then select **Done**.

9. Now you can use the output of the form processing model in subsequent actions in Power Automate.



Manually trigger a flow

File Content | Please select file or image

+ Add an input

Perform a bound action

*Table name: AI Models

*Action Name: Predict

*Row ID: e61cd551-8b1d-4757-9a99-80c3defe59ad

version: 1.0

request: {"mimeType":"application/pdf","base64Encoded":" *fx* string(...) × "}

requestv2:

Parse JSON

*Content: PredictRespons... ×

*Schema:

```
{
    "type": "object",
    "properties": {
        "predictionOutput": {
            "type": "object",
            "properties": {
                "labels": {
                    "type": "object",
                    "properties": {
                        "Total amount": {
```

Generate from sample

Send an email notification (V3)

*To: A list of valid email addresses separated by a semicolon or a comma.

*Subject: The subject of the email.

*Body: Total amount value ×

Add dynamic content from the apps and connectors used in this flow. Hide

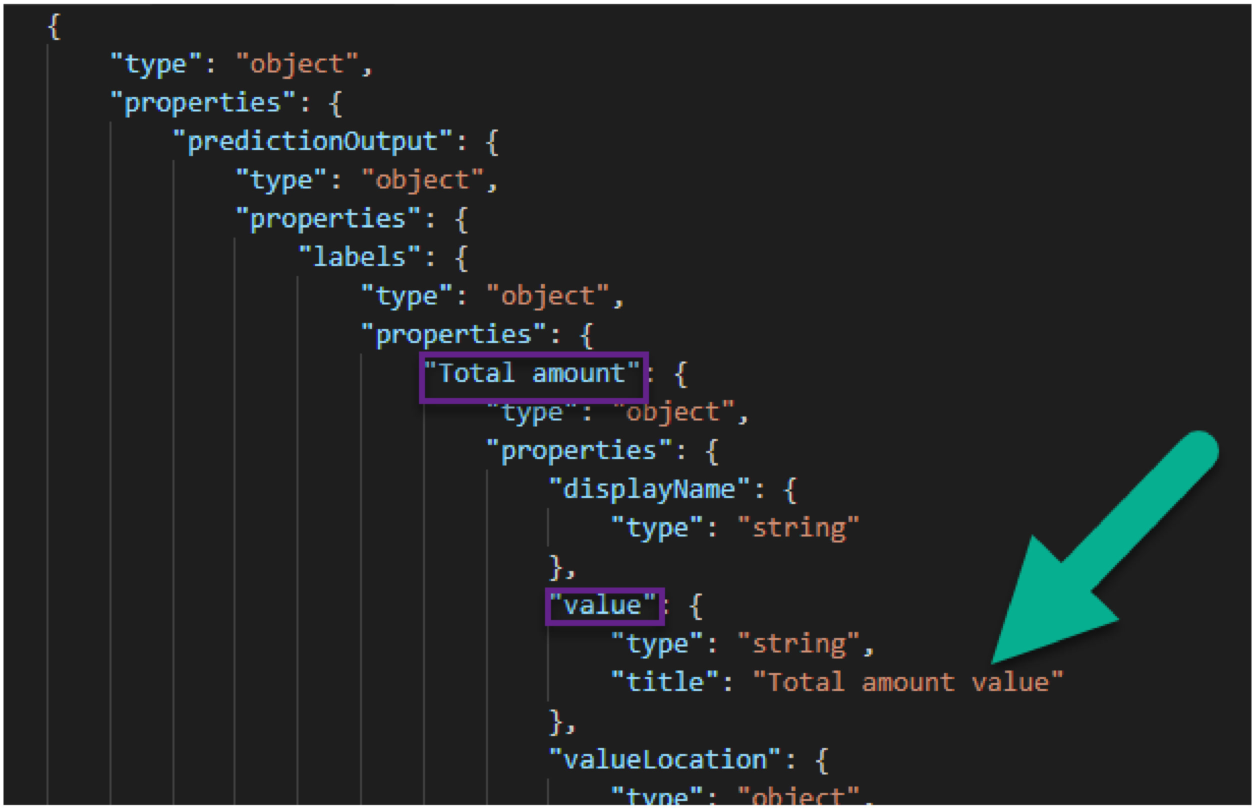
Dynamic content | Expression

Search dynamic content

Parse JSON — See more

- displayName
- Total amount value
- displayName
- value
- displayName
- value
- displayName
- name

Note. All fields will be displayed with the name “value” on the dynamic content pane. To put a more explanatory name, modify the JSON schema that was generated from the Parse JSON step. Put a title attribute like seen in the following screenshot for each field.



```
{
    "type": "object",
    "properties": {
        "predictionOutput": {
            "type": "object",
            "properties": {
                "labels": {
                    "type": "object",
                    "properties": {
                        "Total amount": {
                            "type": "object",
                            "properties": {
                                "displayName": {
                                    "type": "string"
                                },
                                "value": {
                                    "type": "string",
                                    "title": "Total amount value"
                                },
                                "valueLocation": {
                                    "type": "object",
```