

Microsoft Power Platform ローコード開発 [活用] 入門

Chapter 13 「案件管理アプリ③」 参考資料

更新：2022 年 8 月 24 日

見積書を作成する（書籍本体 Chapter13-2 参考資料）

見積書を作成するアプリの概要は図 13-1 のようになります。見積書は見積書作成画面で作成します。見積書作成画面内にはアイコンが 3 つあり、保存アイコンを押すと保存用ダイアログが、フォルダのアイコンを押すと見積書を開くダイアログが、印刷アイコンを押すと印刷プレビューが開きます。

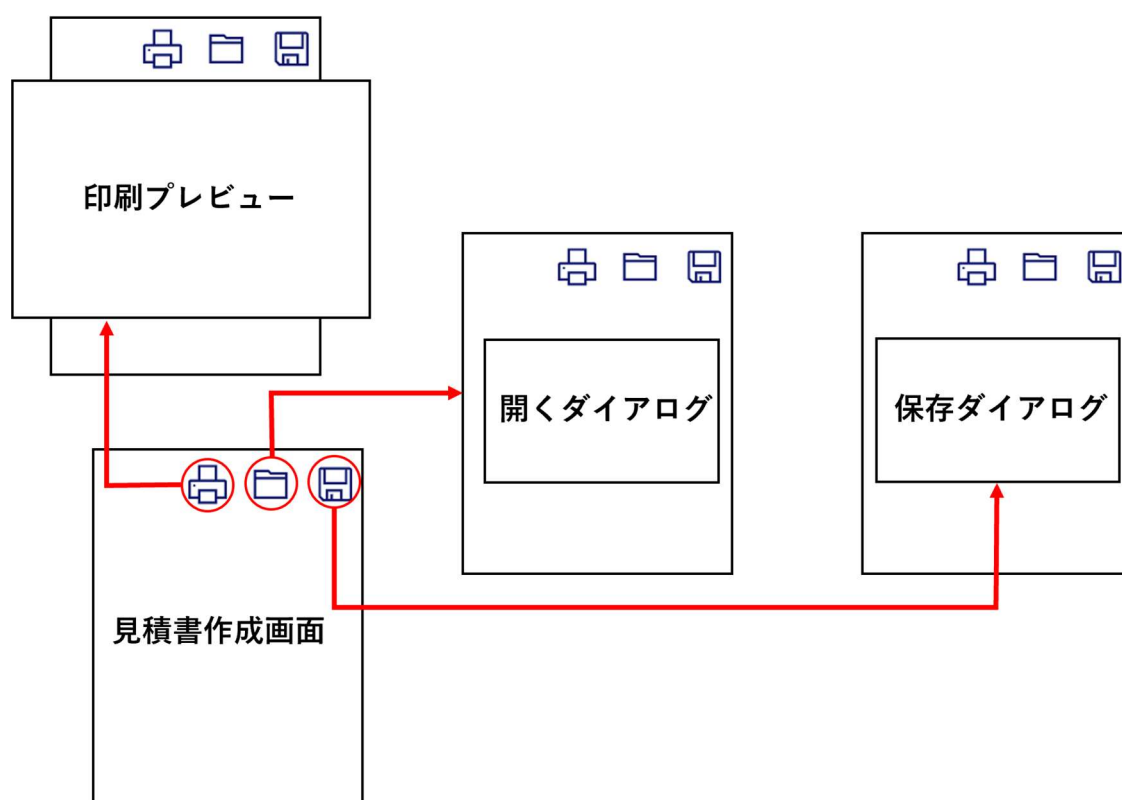


図 13-1

以下のような手順で作成していきます。

1. Dataverse にテーブルを作成する
2. キャンバスアプリの作成とテーブル接続
3. 入力フォームの実装
4. 保存機能の実装
5. 呼び出し機能の実装
6. 印刷機能の実装

1. Dataverse にテーブルを作成する

使用するテーブルの作成

図 13-2 のように作成します。

- ・案件テーブル(trnOpportunityTable)……案件の情報を格納するテーブル。こちらの案件テーブルの作成手順は書籍本体の Chapter11 で紹介されています
- ・見積書テーブル(trnEstimate)……見積書の宛名や条件などの情報を保管するテーブル
- ・見積書詳細テーブル(trnEstimateDetail)……見積書の内訳を格納するテーブル

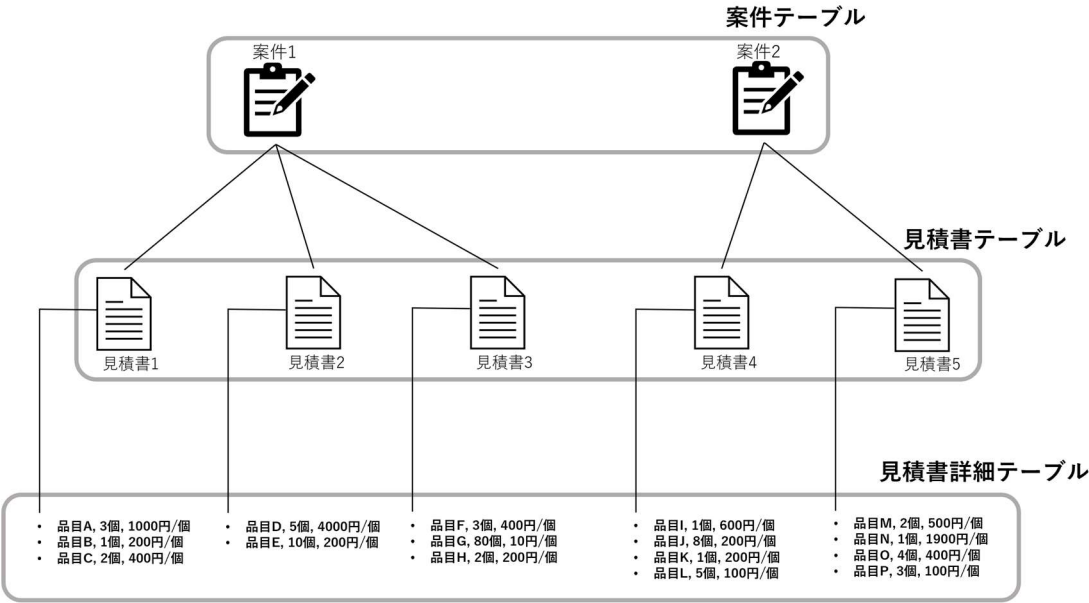


図 13-2

列の追加

表 13-1、13-2 のように列を追加します。

表 13-1 : 見積書テーブル(trnEstimate)

表示名	スキーマ名	データの種別	書式	必須	関連テーブル
タイトル[プライマリ名]	title	一行テキスト	テキスト	必須なビジネス	-
件名	subject	一行テキスト	テキスト	任意	-
宛名	address	一行テキスト	テキスト	任意	-
納期	deadline	日付と時刻	日付のみ	任意	-
支払条件	paymentTerms	一行テキスト	テキスト	任意	-
有効期限	expirationDate	日付と時刻	日付のみ	任意	-
担当者	personInCharge	一行テキスト	テキスト	任意	-
案件	project	検索	-	任意	案件テーブル

表 13-2 : 見積書詳細テーブル(trnEstimateDetail)

表示名	スキーマ名	データの種類	書式	必須	関連テーブル
品目[プライマリ名の列]	item	一行テキスト	テキスト	必須なビジネス	-
数量	quantity	整数	なし	任意	-
単位	unit	一行テキスト	テキスト	任意	-
単価	unitPrice	整数	なし	任意	-
行番号	index	整数	なし	任意	-
見積書	estimate	検索	-	任意	見積書テーブル

2. キャンバスアプリの作成とテーブル接続

キャンバスアプリの画面からテーブルに対してデータの登録や読み取りの操作を行うためには、キャンバスアプリとそのテーブルが接続されている必要があります。そのため、まずキャンバスアプリを作成し、前節で作成したテーブルを接続します。

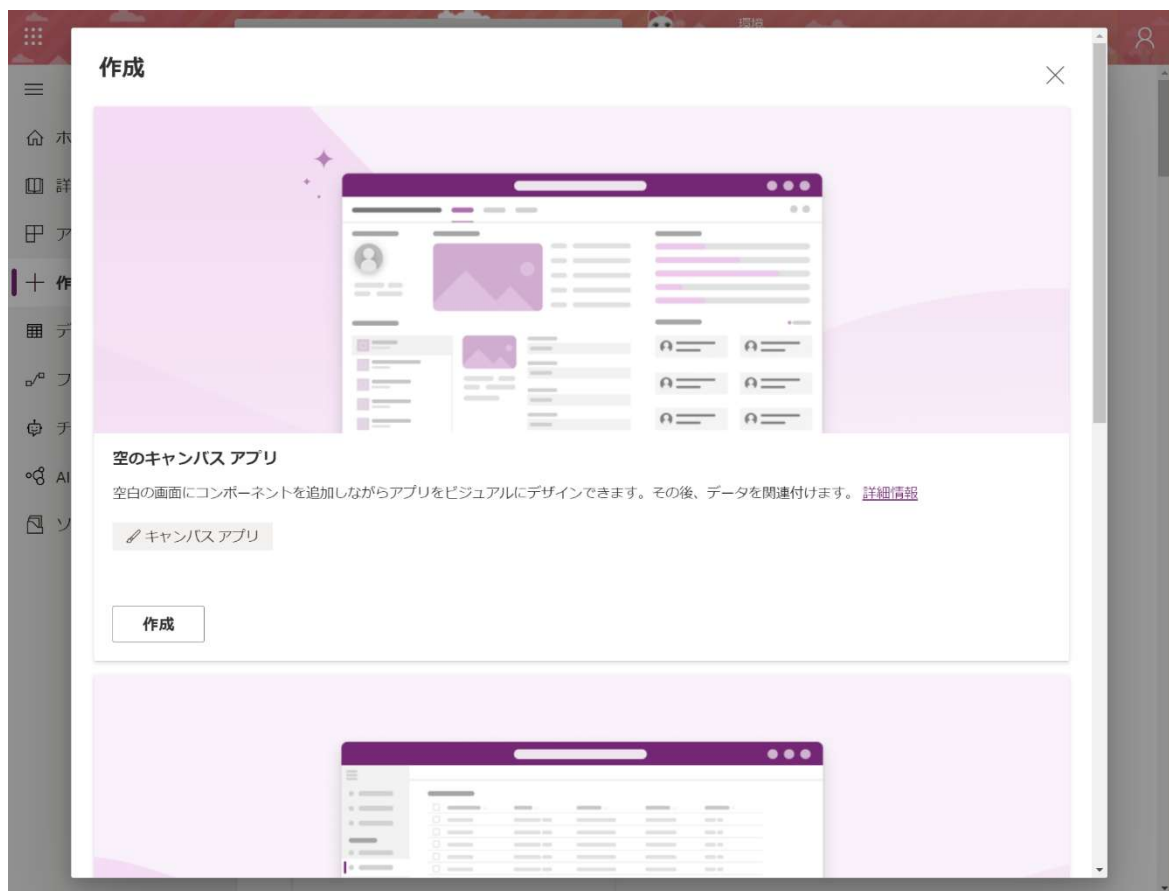
キャンバスアプリの作成

Power Apps メーカーポータルにアクセスし、左のタブの「+作成」を選択します（[画面 13-1](#)）。



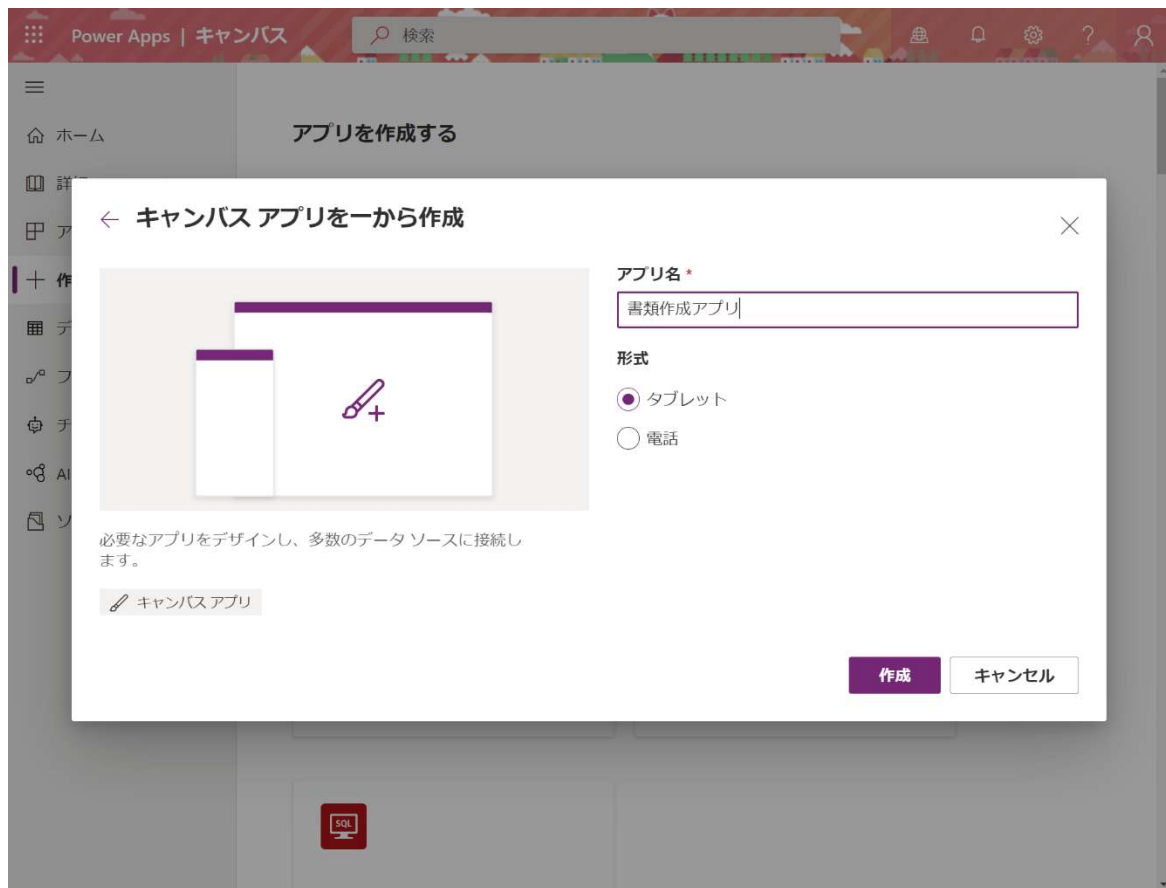
画面 13-1

空のキャンバスアプリの「作成」を選択します（画面 13-2）。



画面 13-2

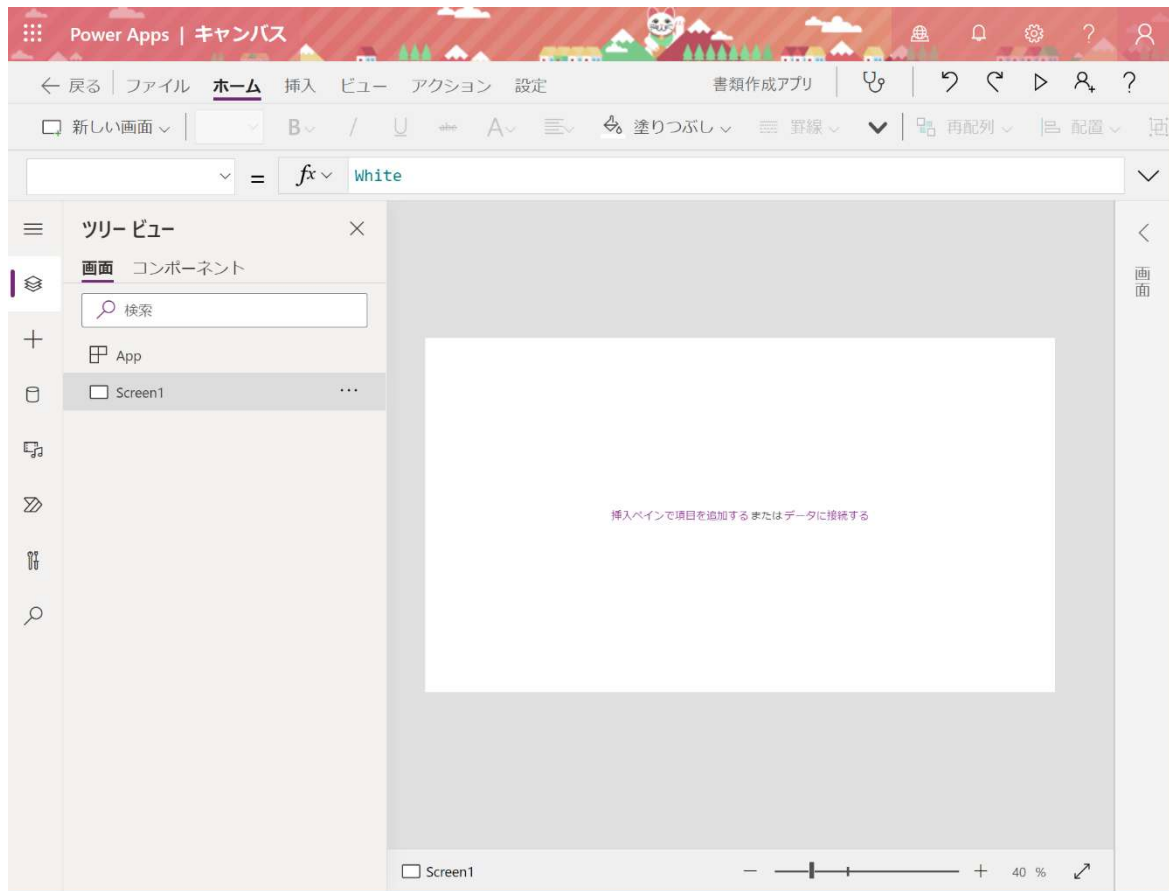
アプリ名を「書類作成アプリ」と入力し、[保存] をクリックします (画面 13-3)。



画面 13-3

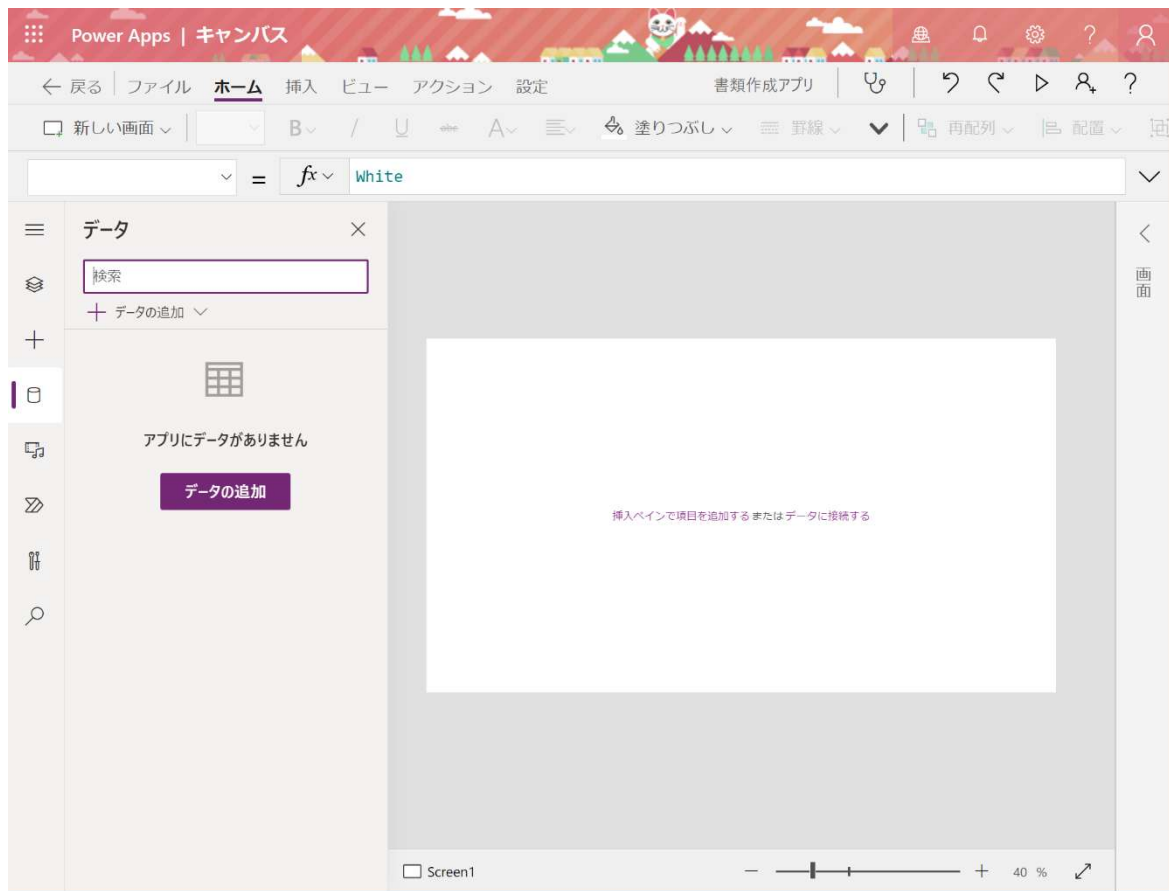
テーブルの接続

書類作成アプリを開き、一番左のタブのデータベースのアイコンを選択します（画面 13-4）。



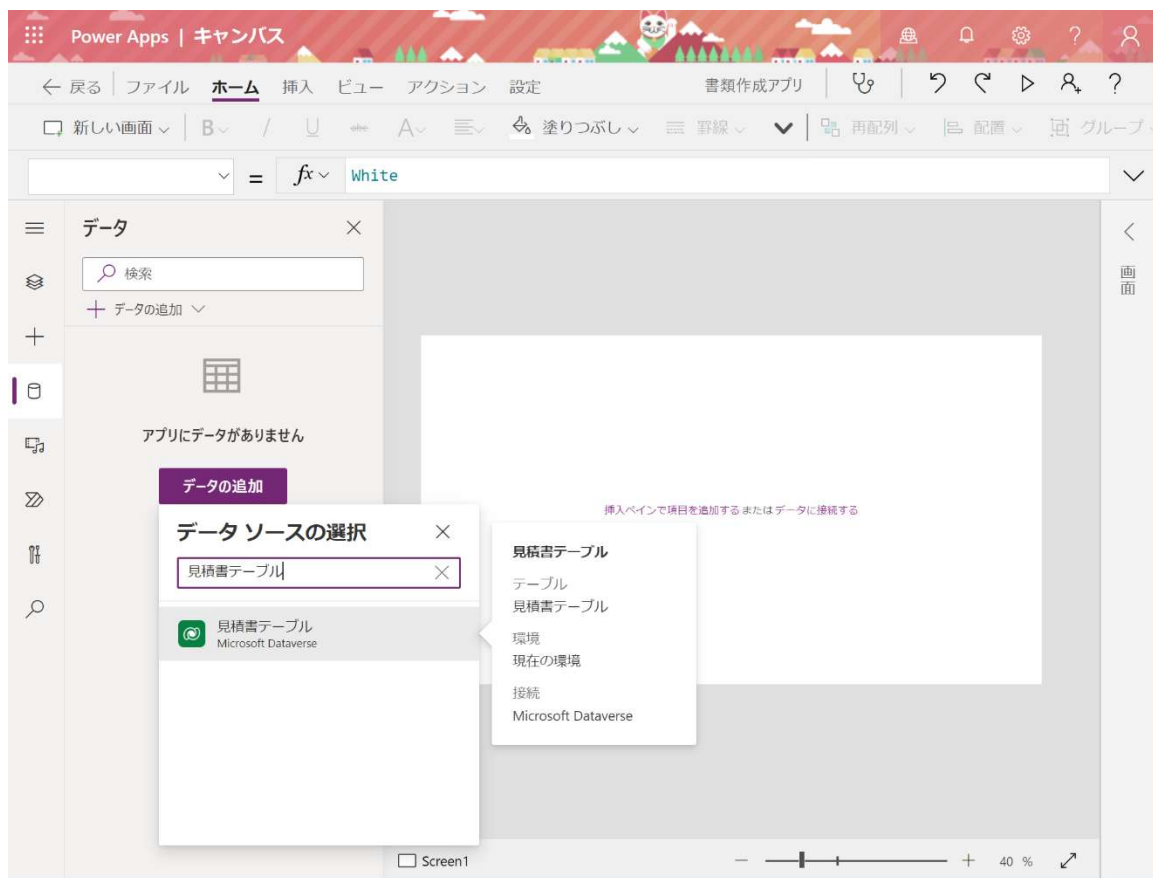
画面 13-4

[データの追加] を選択します。



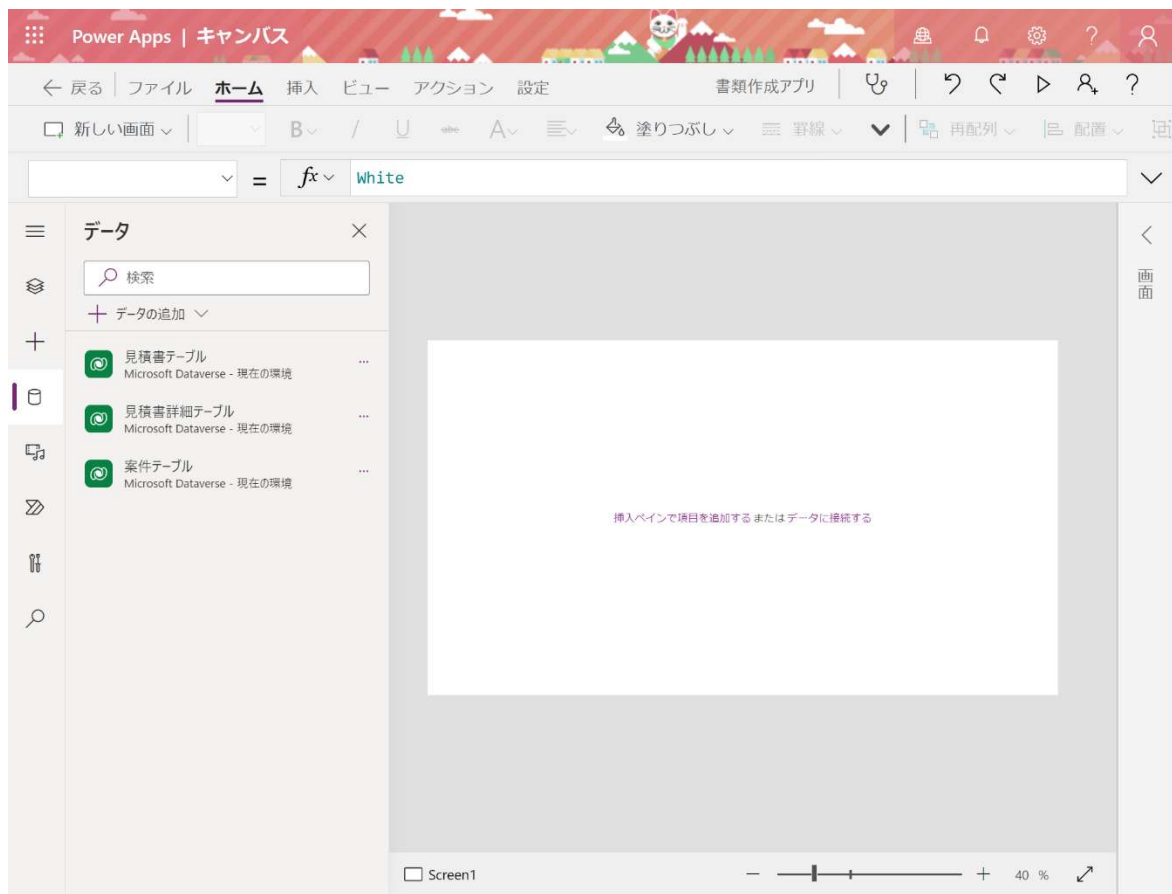
画面 13-5

「見積書テーブル」を選択します（画面 13-6）。



画面 13-6

これでキャンバスアプリと見積書テーブルが接続されます。同様の手順で見積書詳細テーブルと案件テーブルを接続します（画面 13-7）。

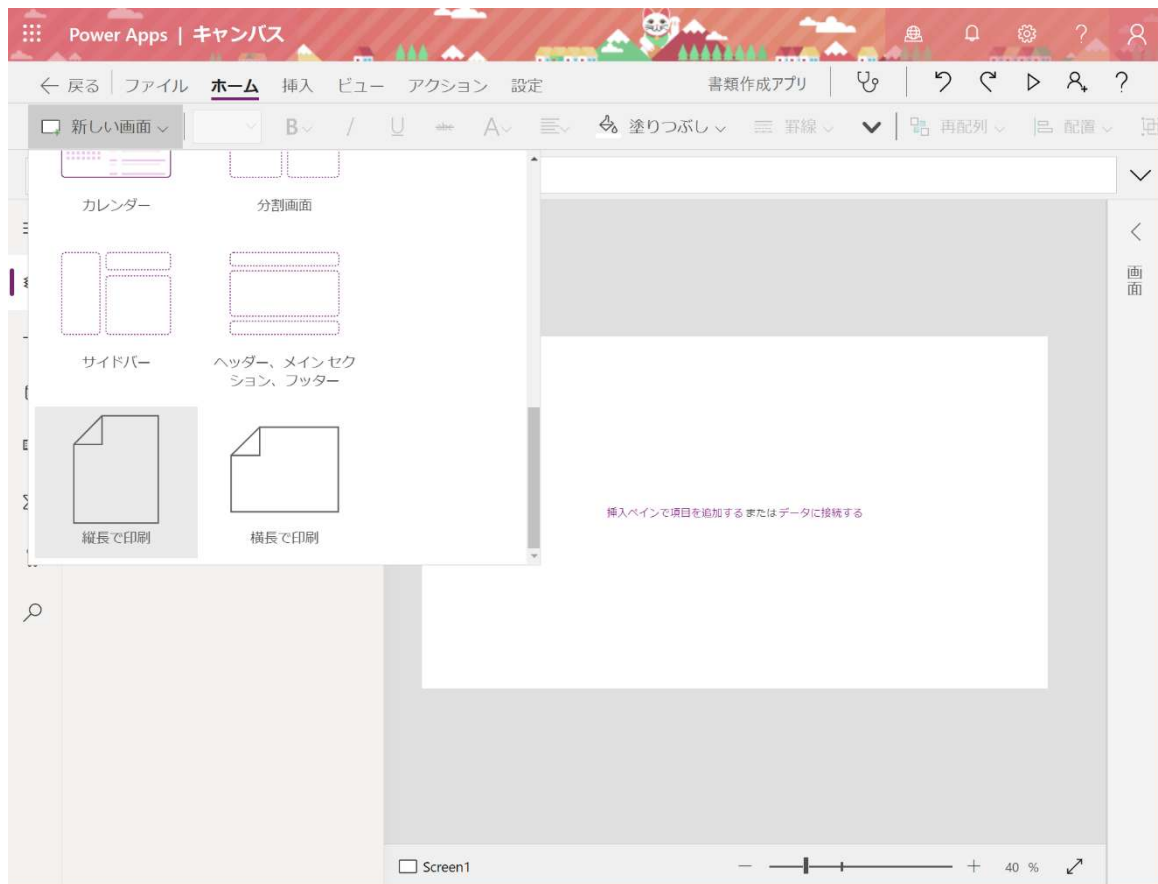


画面 13-7

3. 入力フォームの実装

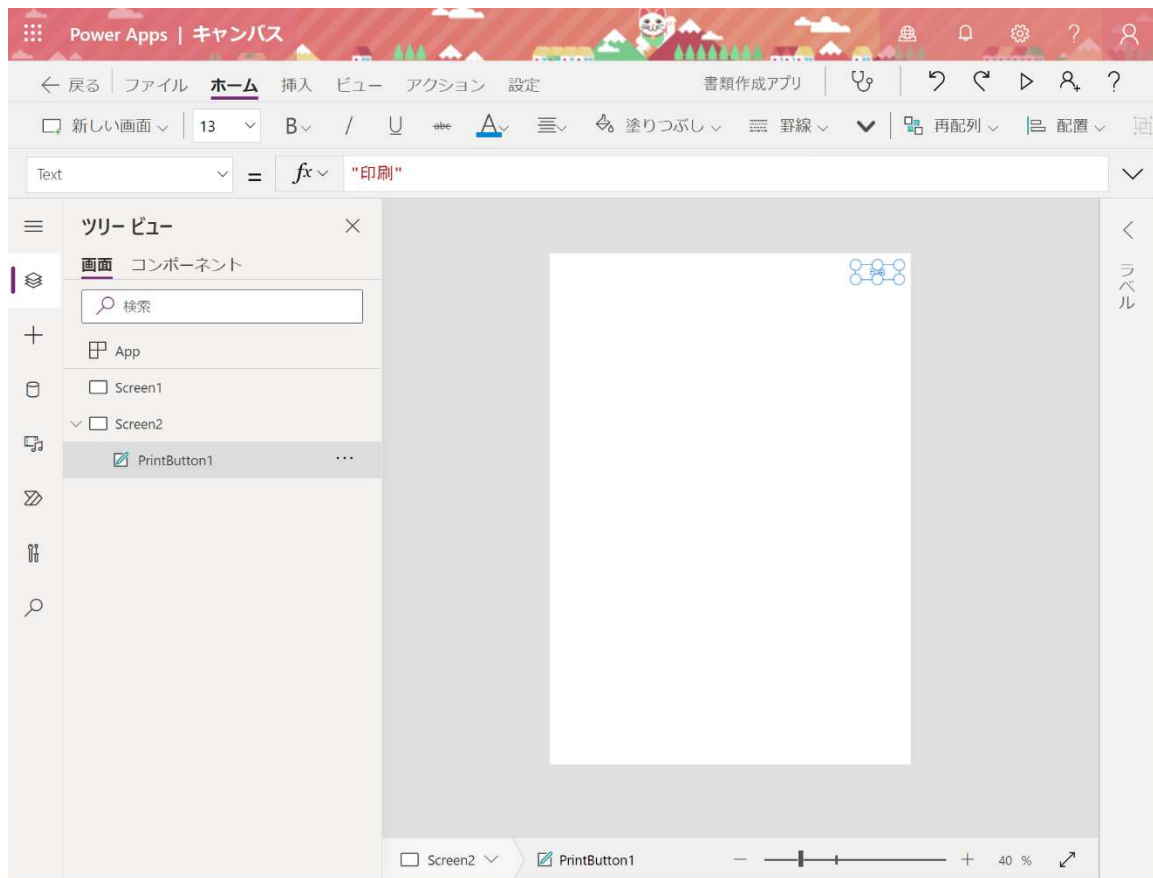
新しい画面の追加

「新しい画面」⇒「縦長で印刷」を選択します（画面 13-8）。



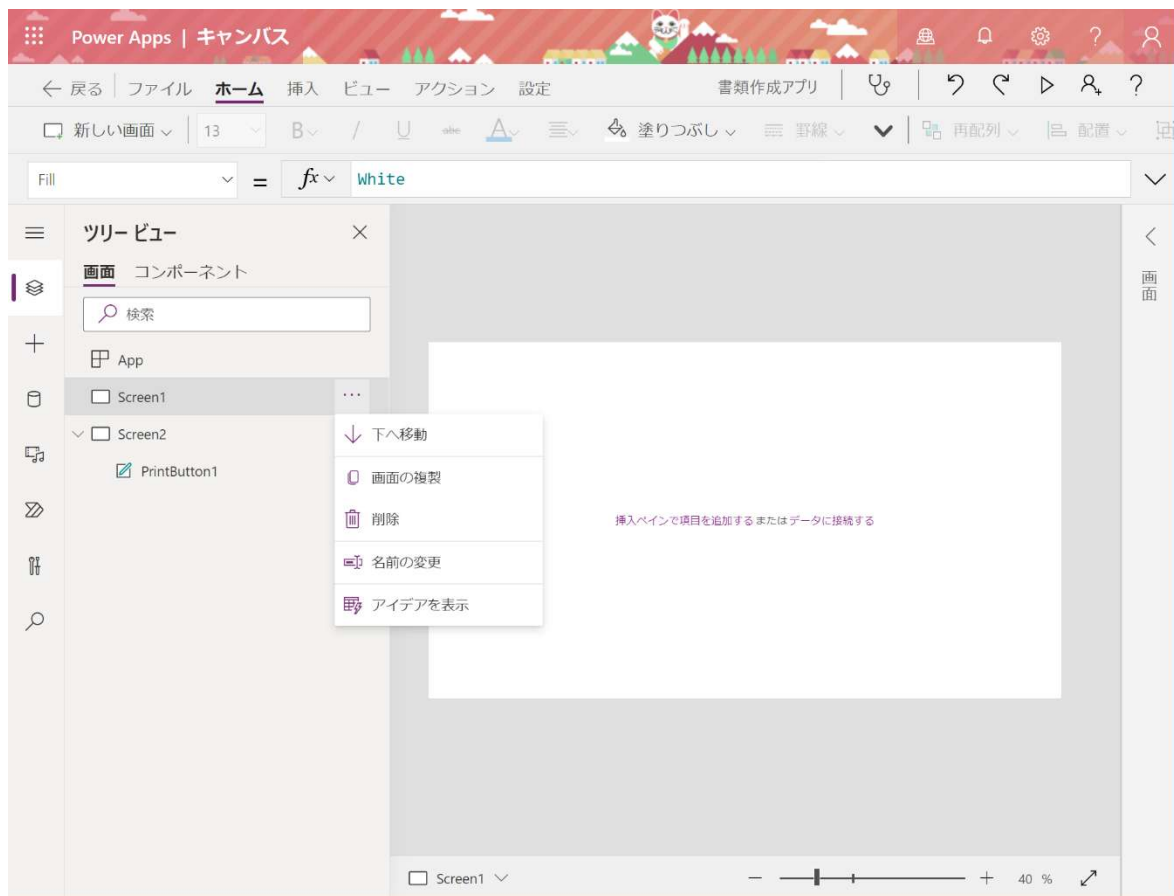
画面 13-8

縦長の画面が作成されます。「印刷」と表示されているラベルは削除します（画面 13-9）。



画面 13-9

左のツリービューを見ると、この時点で画面が二つあることが確認できます。Screen1 は今回使用しないので削除します（画面 13-10）。



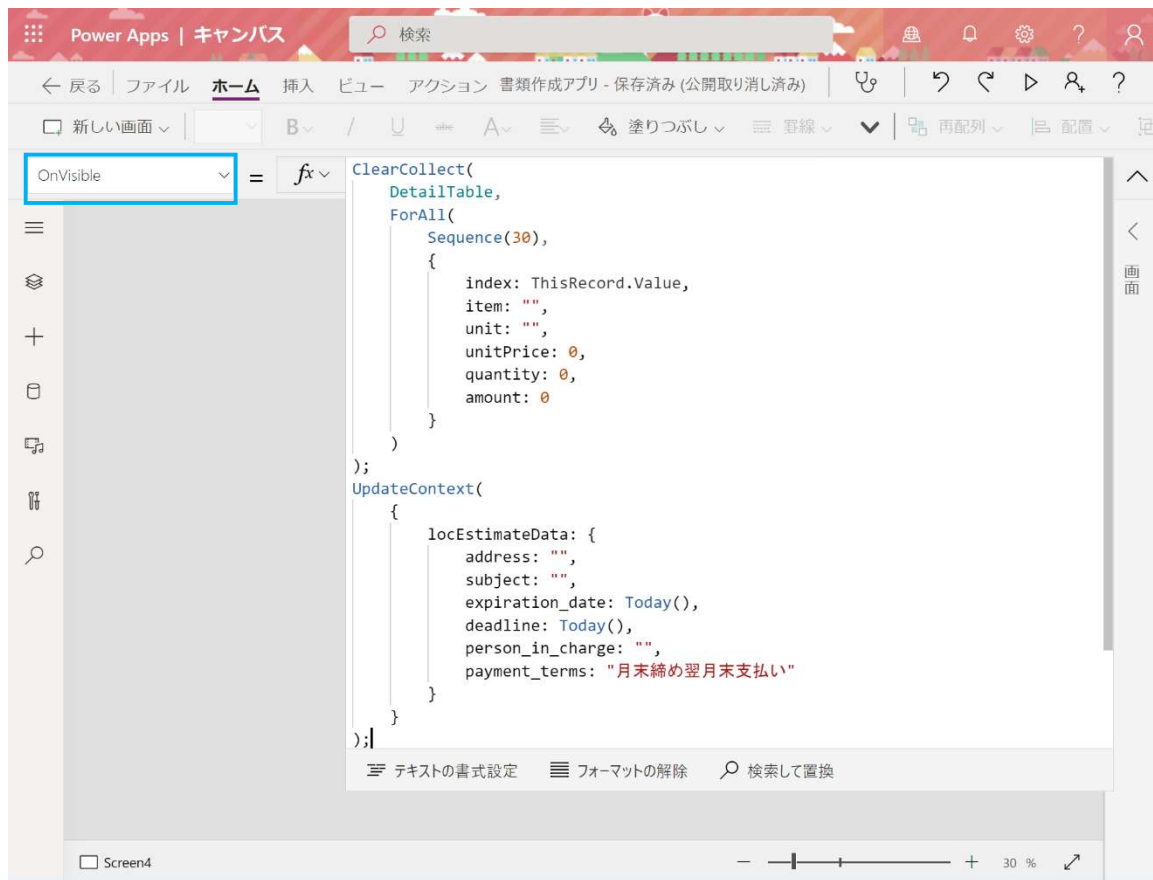
画面 13-10

画面起動時の初期化設定 (OnVisible プロパティの設定)

画面の OnVisible プロパティは画面を開いたときの動作を定義します。ツリービューから [Screen2] ⇒ [OnVisible] を選択し、以下の内容を入力します(画面 13-11)。コード入力時は、[関数の入力補助] フォルダに格納されている<Chapter13-2_パラメータシート.xlsx>をご参照ください。

ここでは、見積書の内訳の初期値を格納する DetailTable というコレクションと、宛名や件名などの初期値を格納する locEstimateData というレコード型のローカル変数を定義します。

```
ClearCollect(
    DetailTable,
    ForAll(
        Sequence(30),
        {
            index: ThisRecord.Value,
            item: "",
            unit: "",
            unitPrice: 0,
            quantity: 0,
            amount: 0
        }
    )
);
UpdateContext(
    {
        locEstimateData: {
            address: "",
            subject: "",
            expiration_date: Today(),
            deadline: Today(),
            person_in_charge: "",
            payment_terms: "月末締め翌月末支払い"
        }
    }
);
```



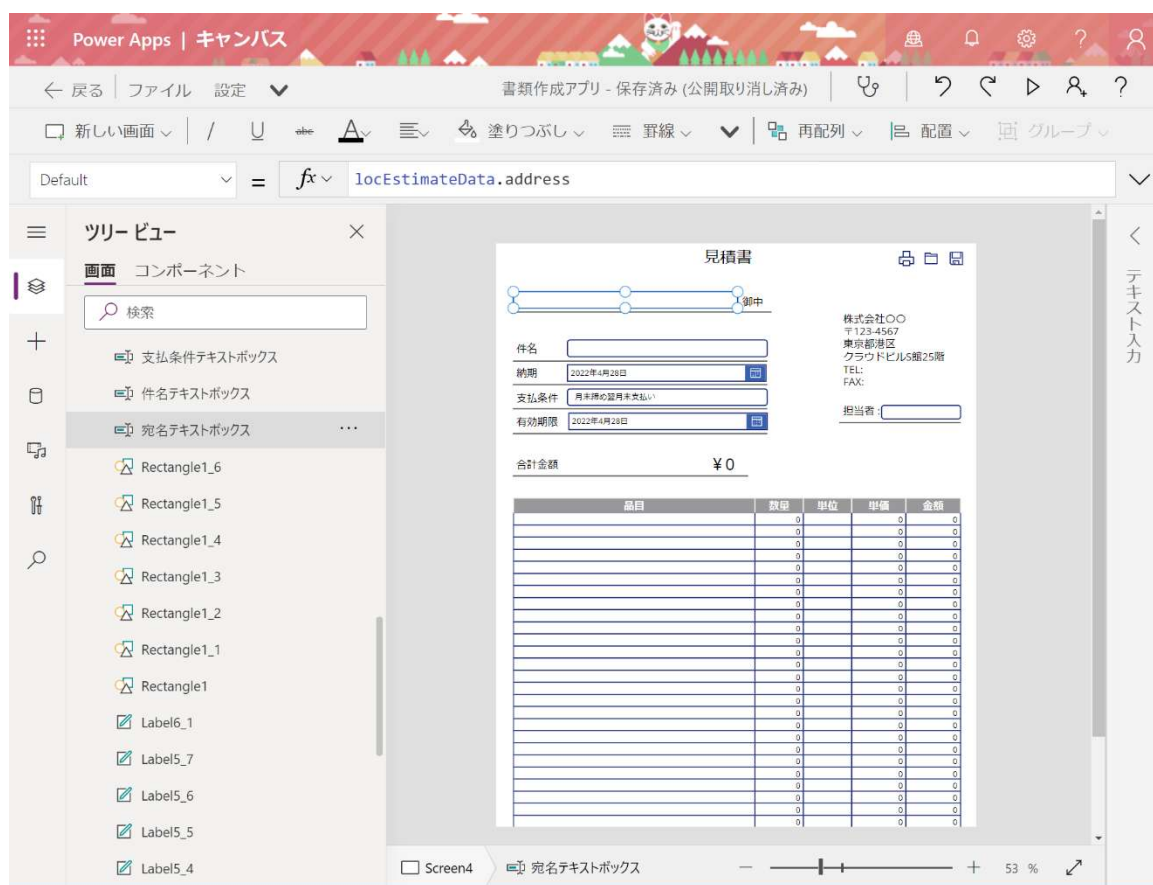
画面 13-11

コントロールの配置 & プロパティの設定

追加した画面にコントロールを配置します。便宜上、[画面 13-12](#) で示すように A、B、C のセクションに分割しています。

説明の中で「〇〇のコントロール名」と表記している部分については、該当するコントロールの名前を入力してください。コントロールの名前を確認したい場合、[ツリービュー] から対象のコントロールを選択します。

画面 13-13 の場合、選択されているコントロールの名前は「宛名テキストボックス」であることが確認できます。



画面 13-13

■ セクション A

- ① ラベル … “見積書” のテキストを表示するラベル。[Text] プロパティを選択⇒ “見積書” と入力。
- ② テキスト入力 … 宛名の入力欄。[Default] プロパティ⇒

- locEstimateData.address と入力。ツリービューからコントロール名を“宛名テキストボックス”に変更します。
- ③ ラベル … “御中”のテキストを表示するラベル。[Text] プロパティを選択⇒“御中”と入力。
 - ④ 四角形 … 四角形アイコンを利用して一本線を作ります。
[Height] プロパティを選択⇒1 と入力。
 - ⑤ ラベル … “件名”のテキストを表示。[Text] プロパティを選択⇒「件名」と入力。
 - ⑥ テキスト入力 … 件名の入力欄。[Default] プロパティ⇒locEstimateData.subject と入力。ツリービューからコントロール名を“件名テキストボックス”に変更します。
 - ⑦ 四角形 … ④と同様
 - ⑧ ラベル … “納期”のテキストを表示。[Text] プロパティを選択⇒“納期”と入力。
 - ⑨ 日付の選択 … 納期を選択する [日付の選択コントロール] を挿入します。カレンダー形式で日付を選択することができます。
[DefaultDate] プロパティ⇒locEstimateData.deadline と入力。ツリービューからコントロール名を“納期ドロップダウン”に変更します。
 - ⑩ 四角形 … ④と同様
 - ⑪ ラベル … “支払条件”のテキストを表示するラベル。[Text] プロパティを選択⇒“支払条件”と入力。
 - ⑫ テキスト入力 … 支払条件の入力欄。[Default] プロパティ⇒locEstimateData.payment_terms と入力。ツリービューからコントロール名を“支払条件テキストボックス”に変更します。
 - ⑬ 四角形 … ④と同様
 - ⑭ ラベル … “有効期限”のテキストを表示するラベル。[Text] プロパティを選択⇒“有効期限”と入力。
 - ⑮ 日付の選択 … 有効期限を選択する [日付の選択コントロール]。[DefaultDate] プロパティ⇒locEstimateData.expiration_date と入力。ツリービューからコントロール名を“有効

期限ドロップダウン”に変更します。

- ⑫ 四角形 … ④と同様
- ⑬ ラベル … “合計金額”のテキストを表示するラベル。[Text] プロパティを選択⇒“合計金額”と入力。
- ⑭ ラベル … 合計金額の値を表示するラベル。
- ⑮ 四角形 … ④と同様
- ⑯ ラベル … 会社住所などの情報のテキストを表示するラベル。
[Text] プロパティを選択⇒会社情報を入力。
- ⑰ ラベル … “担当者”のテキストを表示するラベル。[Text] プロパティを選択⇒“担当者”と入力。
- ⑱ テキスト入力 … 担当者の入力欄。[Default] プロパティ⇒locEstimateData. person_in_charge と入力。ツリービューからコントロール名を“担当者テキストボックス”に変更します。
- ⑲ 四角形 … ④と同様

■ セクションB

- ① アイコン(印刷) … 印刷する時に押すアイコン
- ② アイコン(フォルダ) … ファイルを開く時に押すアイコン
- ③ アイコン(保存) … 保存する時に押すアイコン

■ セクションC

- ① ラベル … “品目”のテキストを表示するラベル。[Text] プロパティを選択⇒“品目”と入力。[Align] プロパティ⇒Align.Center と入力。①～⑤は表の列名として使用するため、背景の色([Fill] プロパティ)や文字の色([Color] プロパティ)を適宜変更する。
- ② ラベル … “数量”のテキストを表示するラベル。[Text] プロパティを選択⇒“数量”と入力。[Align] プロパティ⇒Align.Center と入力。
- ③ ラベル … “単位”のテキストを表示するラベル。[Text] プロパティを選択⇒“単位”と入力。[Align] プロパティ⇒

Align.Center と入力。

- ④ ラベル … “単価” のテキストを表示するラベル。[Text] プロパティを選択⇒ “単価” と入力。[Align] プロパティ⇒ Align.Center と入力。
- ⑤ ラベル … “金額” のテキストを表示するラベル。[Text] プロパティを選択⇒ “金額” と入力。[Align] プロパティ⇒ Align.Center と入力。
- ⑥ ギャラリー … 見積書の詳細を表示する縦方向のギャラリー。
[挿入] ⇒ [ギャラリー] ⇒ [縦方向(空)] で追加。[Items] プロパティを選択⇒ DataTable と入力。ツリービューからコントロール名を「摘要欄 Gallery」に変更します。
- ⑦ テキスト入力 … 品目の入力欄。⑦～⑪は⑥のギャラリー内に設置します。ギャラリー内にコントロールを配置する手順は図 13-3 の画像の通りです。

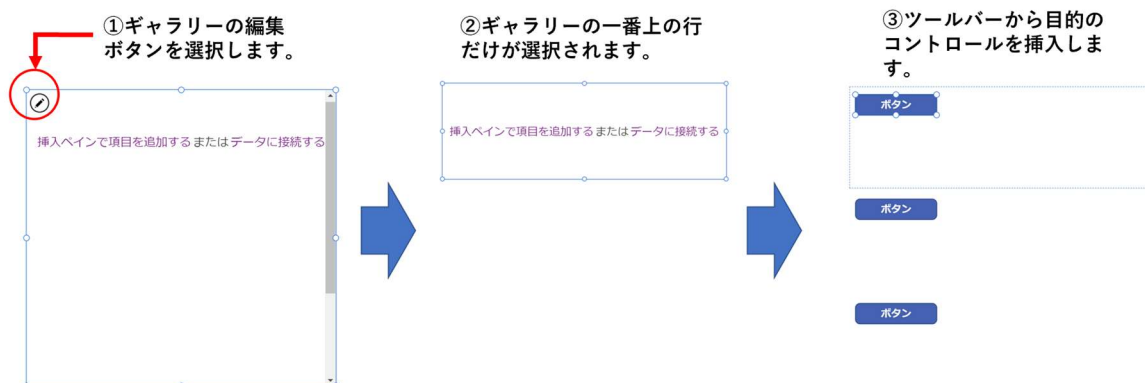
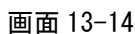


図 13-3

コントロールを配置したら、[Default] プロパティ⇒ ThisItem.item と入力。[BorderThickness] プロパティ⇒ 0 と入力。[Align] プロパティ⇒ Align.Right と入力。ツリービューからコントロール名を“品目 TextInput”に変更します。

30 行の入力欄がギャラリーに収まるように適宜コントロールのサイズを調整してください。

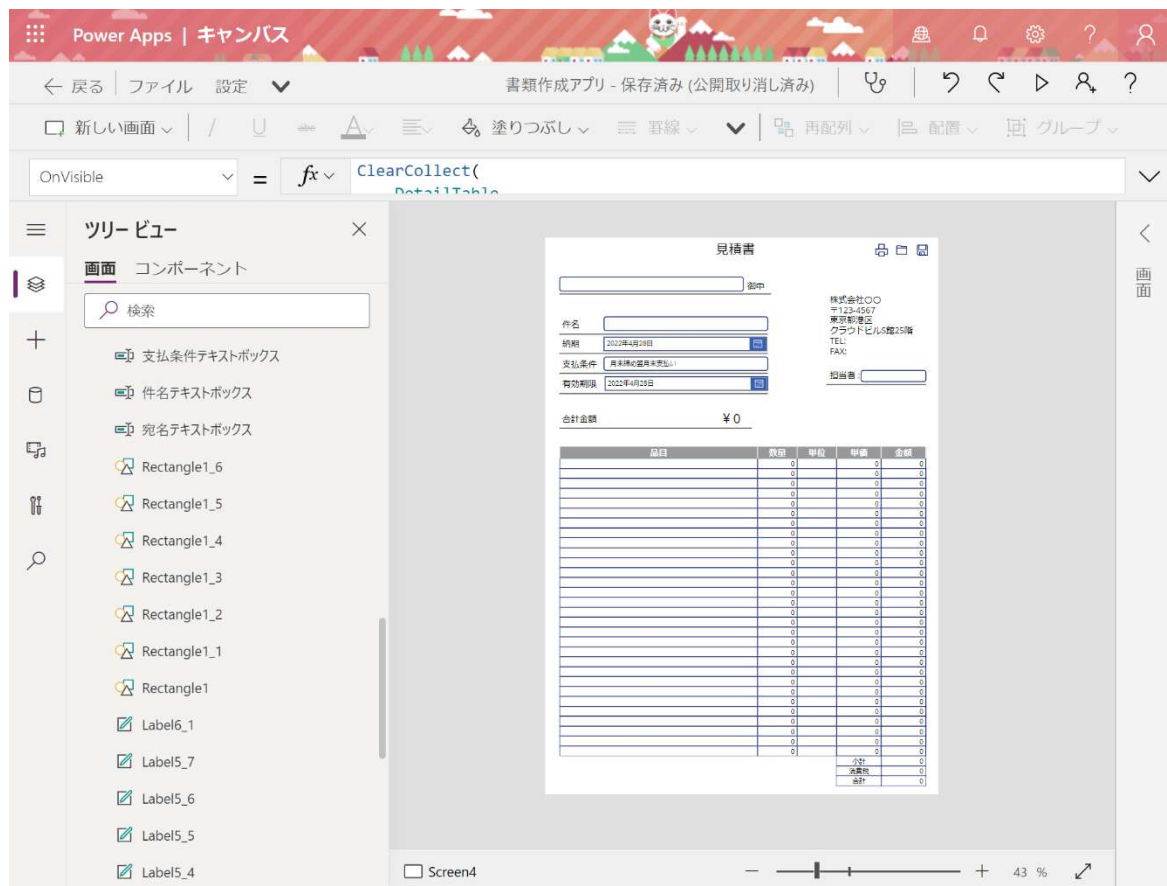
- ⑧ テキスト入力 … 数量の入力欄。[Default] プロパティ⇒ ThisItem.quantity と入力。[BorderThickness] プロパティ⇒ 0 と入力。[Align] プロパティ⇒ Align.Right と入力。ツリービューからコントロール名を“数量 TextInput”に変更します。
- ⑨ テキスト入力 … 単位の入力欄。[Default] プロパティ⇒ ThisItem.unit と入力。[BorderThickness] プロパティ⇒ 0 と入力。[Align] プロパティ⇒ Align.Right と入力。ツリービューからコントロール名を“単価 TextInput”に変更します。
- ⑩ テキスト入力 … 単価の入力欄。[Default] プロパティ⇒ ThisItem.unitPrice と入力。[BorderThickness] プロパティ⇒ 0 と入力。[Align] プロパティ⇒ Align.Right と入力。ツリービューからコントロール名を“単位 TextInput”に変更します。
- ⑪ テキスト入力 … 金額の入力欄。[Default] プロパティ⇒ [⑧のコントロール名].Text*[⑩のコントロール名].Text と入力。
[BorderThickness] プロパティを選択⇒ 0 と入力。[Align] プロパティ⇒ Align.Right と入力。ツリービューからコントロール名を“金額 TextInput”に変更します。
- ⑫ ラベル … “小計”のテキストを表示するラベル。[Text] プロパティを選択⇒ “小計” 入力。
- ⑬ ラベル … “消費税”のテキストを表示するラベル。[Text] プロパティを選択⇒ “消費税” と入力。
- ⑭ ラベル … “合計”のテキストを表示するラベル。[Text] プロパティを選択⇒ “合計” と入力。
- ⑮ ラベル … 小計の値を表示するラベル。[Text] プロパティを選択⇒ Sum([⑥のコントロール名].AllItems. [⑪のコントロール名] , [⑪のコントロール名].Text) を入力 (画面 13-14)。
[Align] プロパティ⇒ Align.Right と入力。ツリービューからコントロール名を“小計ラベル”に変更します。



- ⑩ ラベル … 消費税の値を表示するラベル。[Text] プロパティを選択⇒RoundDown([⑤のコントロール名].Text*0.10 , 0)を入力。[Align] プロパティ⇒Align.Right と入力。ツリービューからコントロール名を“消費税ラベル”に変更します。
- ⑪ ラベル … 合計の値を表示するラベル。[Text] プロパティを選択⇒[⑤のコントロール名].Text + [⑩のコントロール名].Text と入力。[Align] プロパティ⇒Align.Right と入力。ツリービューからコントロール名を“合計ラベル”に変更します。

A-⑱の合計金額

ここまでの手順で画面 13-15 のような画面が作成されています。最後に A-⑱の合計金額の値を表示します。[Text] プロパティを選択⇒“¥” & [C-⑰のコントロール名].Text と入力します。



画面 13-15

4. 保存機能の実装

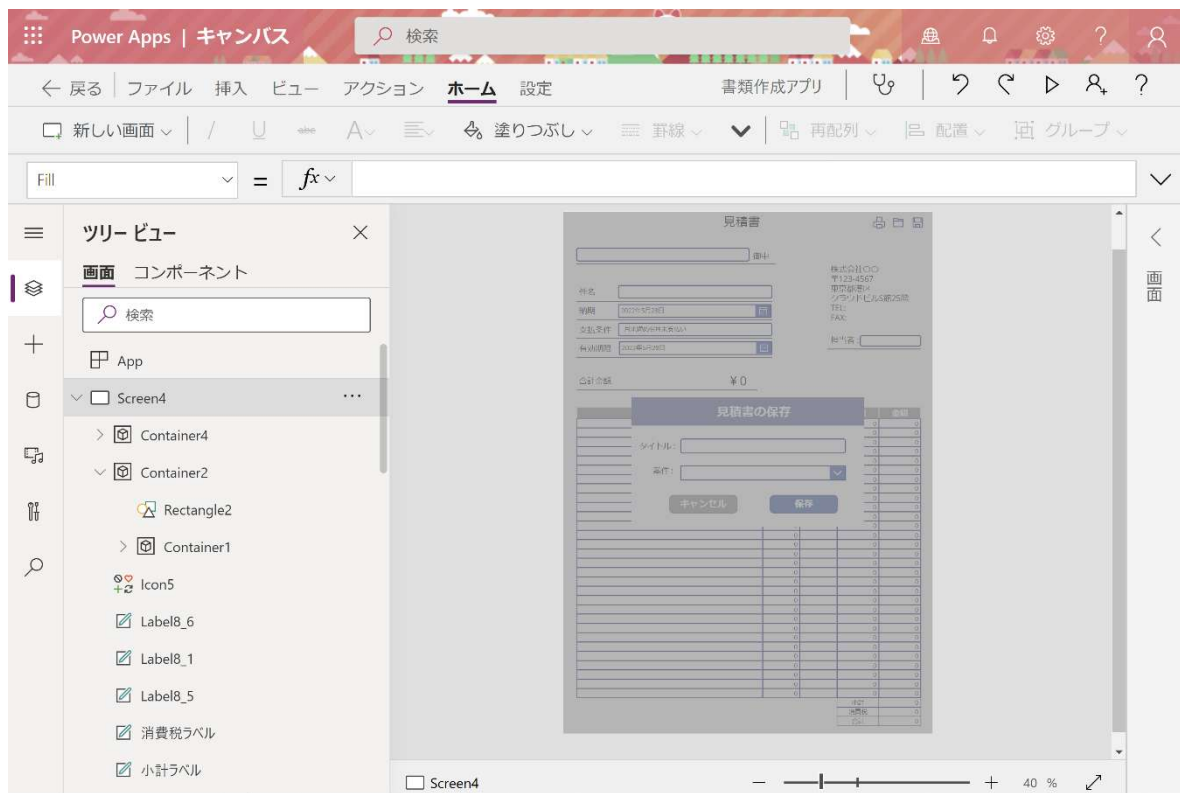
保存ダイアログの作成

作成した見積書を保存するためのダイアログを作成します。ここでは新たに画面を追加するのではなく、セクション A、B、C を作成した画面内にコントロールを配置し、最終的に画面 13-16 のようなダイアログを作成します。

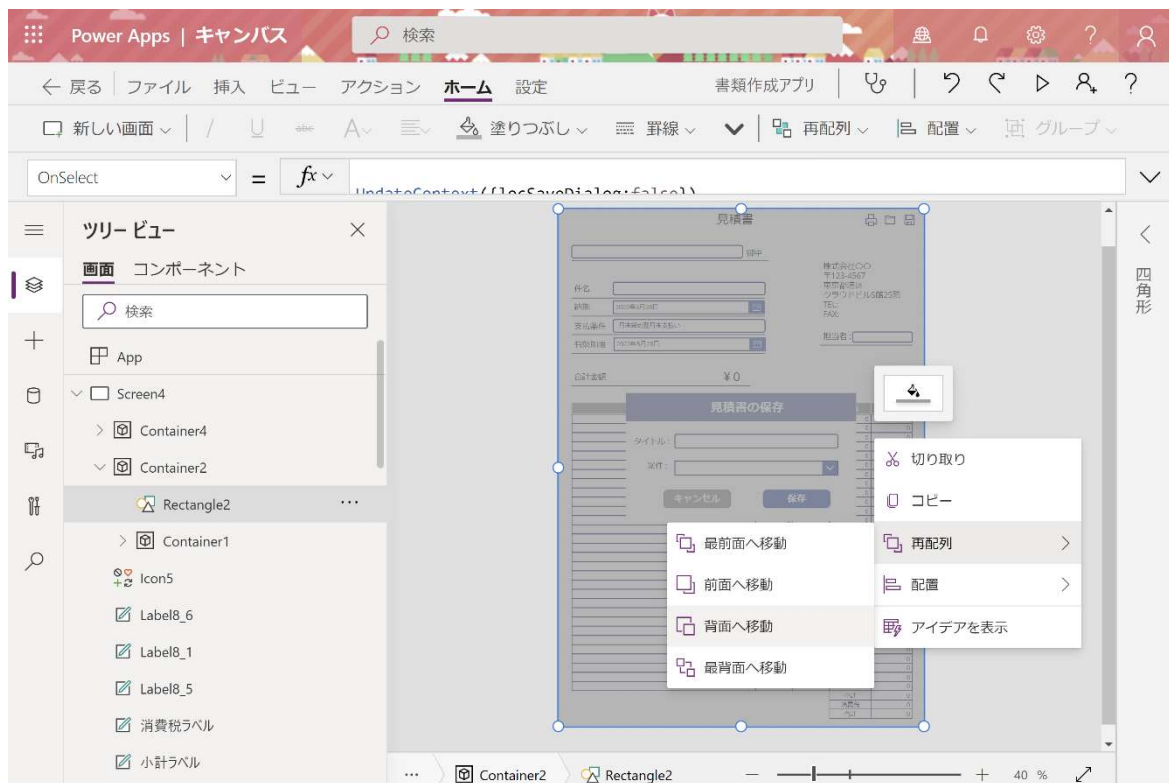
画面 13-16

■ セクションD

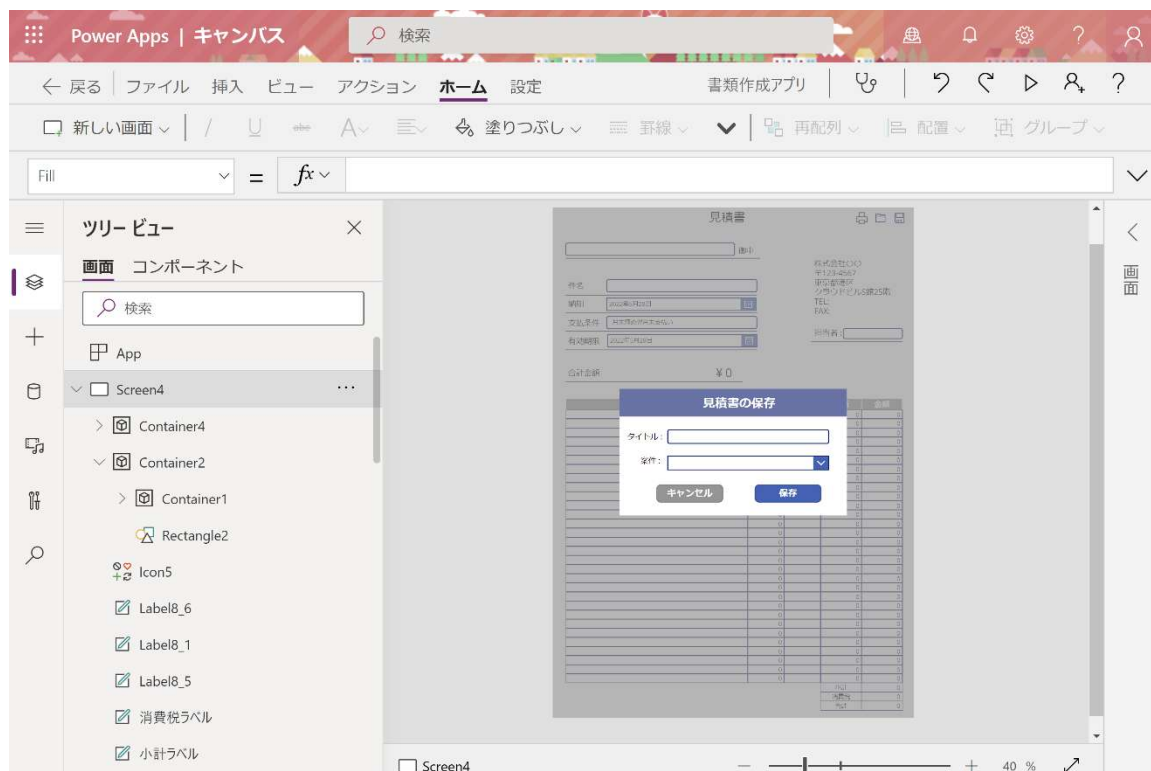
- ① ラベル … ダイアログの土台として使うラベル。[Text] プロパティを選択⇒ “” と入力。[Fill] プロパティを選択⇒White と入力。
- ② ラベル … “タイトル：”のテキストを表示するラベル。[Text] プロパティを選択⇒ “タイトル：” と入力。
- ③ テキスト入力 … タイトルの入力欄。ツリービューからコントロール名を「タイトルテキストボックス」に変更します。
- ④ ラベル … “案件”のテキストを表示するラベル。[Text] プロパティを選択⇒ “案件：” と入力。
- ⑤ ドロップダウン … 見積書に紐づける案件を選択するドロップダウン。[Items] プロパティを選択⇒案件管理テーブルと入力。ツリービューからコントロール名を「案件ドロップダウン」に変更します。
- ⑥ ボタン … キャンセルボタン。[Text] プロパティを入力⇒ “キャンセル” と入力。背景色を適宜変更する。([Fill] プロパティ)
- ⑦ ボタン … 保存ボタン。[Text] プロパティを入力⇒ “保存” と入力。
- ⑧ ラベル … “見積書の保存”のテキストを表示するラベル。
[Text] プロパティを入力⇒ “見積書の保存” と入力。適宜背景の色([Fill] プロパティ)を変える。
- ⑨ 四角形 … ダイアログの背景として表示する。[Fill] プロパティを選択⇒RGBA(149, 149, 149, 0.7)を入力。ここまでの手順通りにダイアログの背景を配置した場合、[画面 13-17](#) のようにこの背景が一番前に来てしまいます。このような場合はコントロールを右クリック⇒ [再配列] ⇒ [背面へ移動] を選択することでコントロールの表示順序を変更します([画面 13-18](#)、[13-19](#))。



画面 13-17



画面 13-18



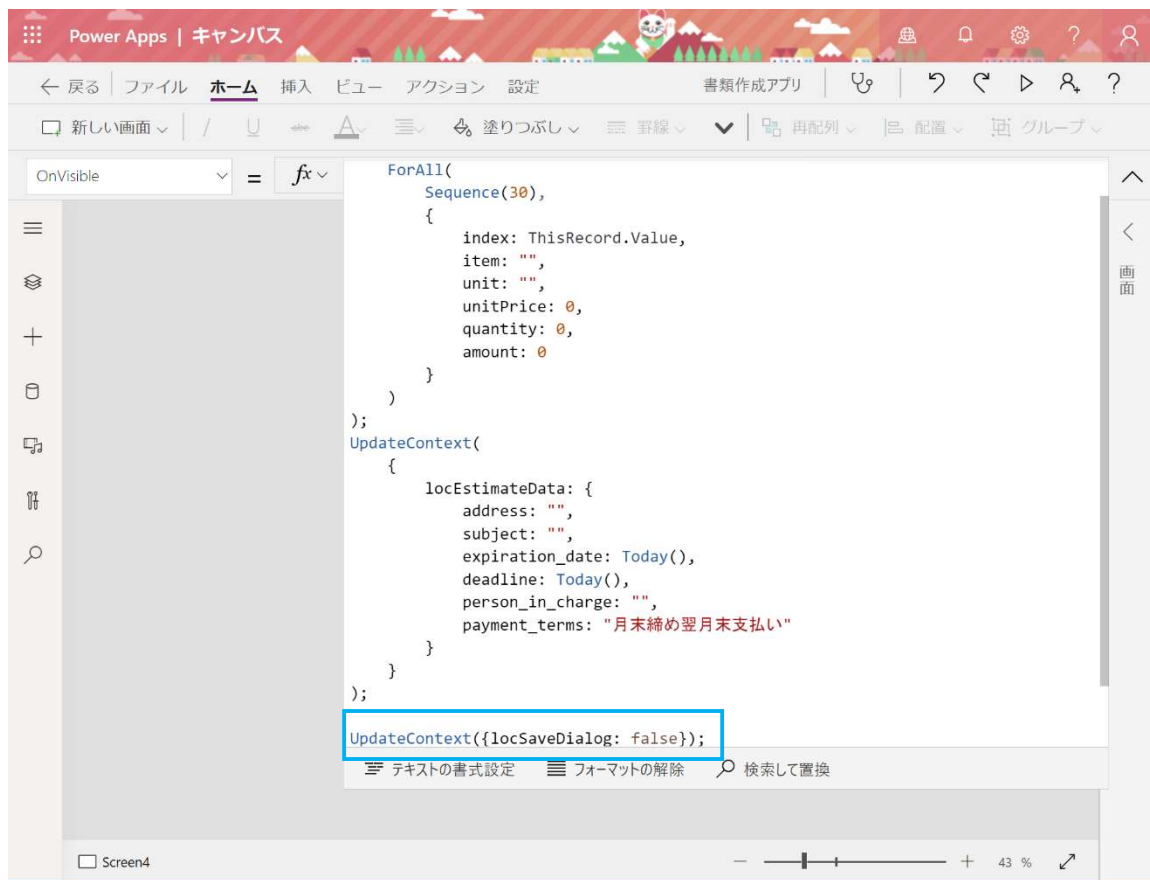
画面 13-19

ボタンを押してダイアログを開閉する

保存アイコンをクリックするとダイアログが開く、ダイアログが開いた状態で背景画面をクリックするとダイアログが閉じる、のような動作を設定します。これは [Visible] プロパティを利用することで実現できます。

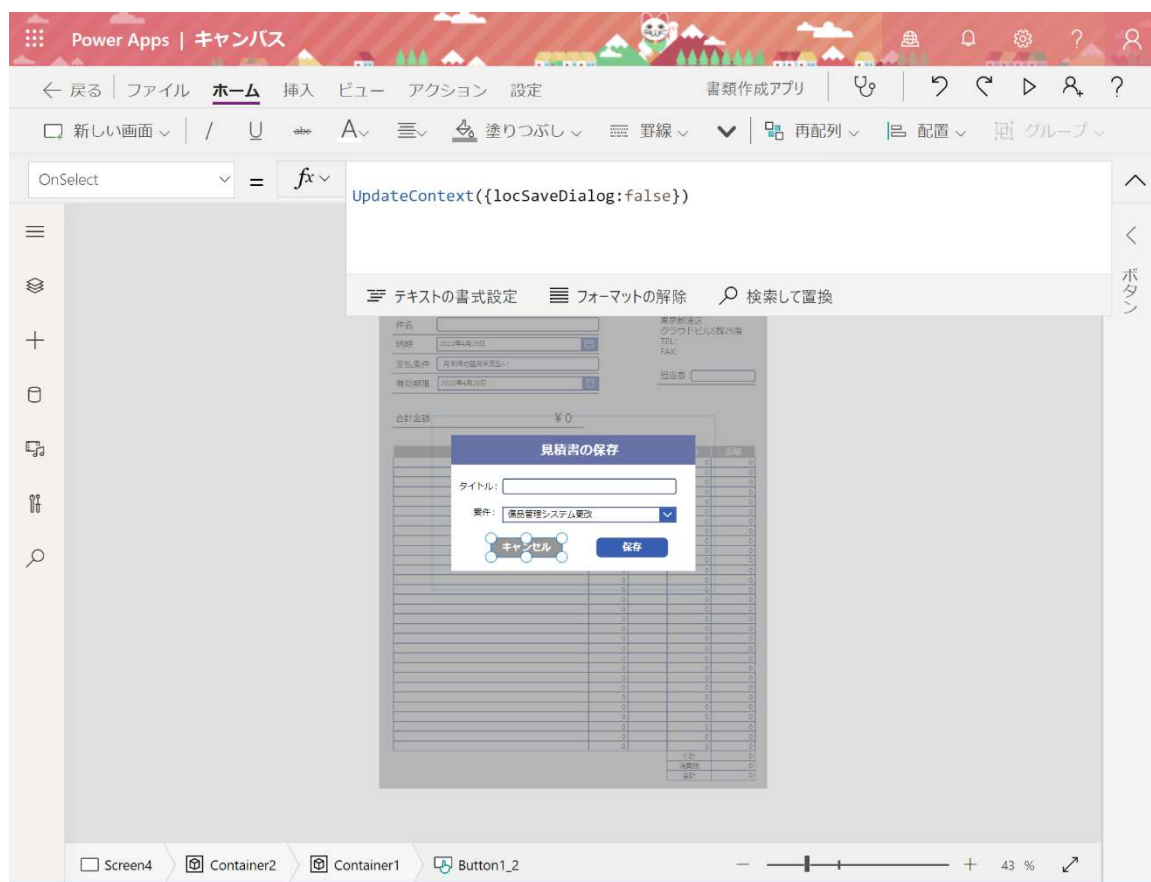
[Visible] プロパティが true の場合は画面上に表示され、false の場合は非表示になります。

この [Visible] プロパティに Bool 型(true/false のいずれかを保持するデータ型)の変数を設定します。画面の [OnVisible] プロパティを選択⇒ `UpdateContext({locSaveDialog: false});`を追加します (画面 13-20)。この状態でボタンのクリックをすると、ダイアログの表示/非表示を切り替えることができます。



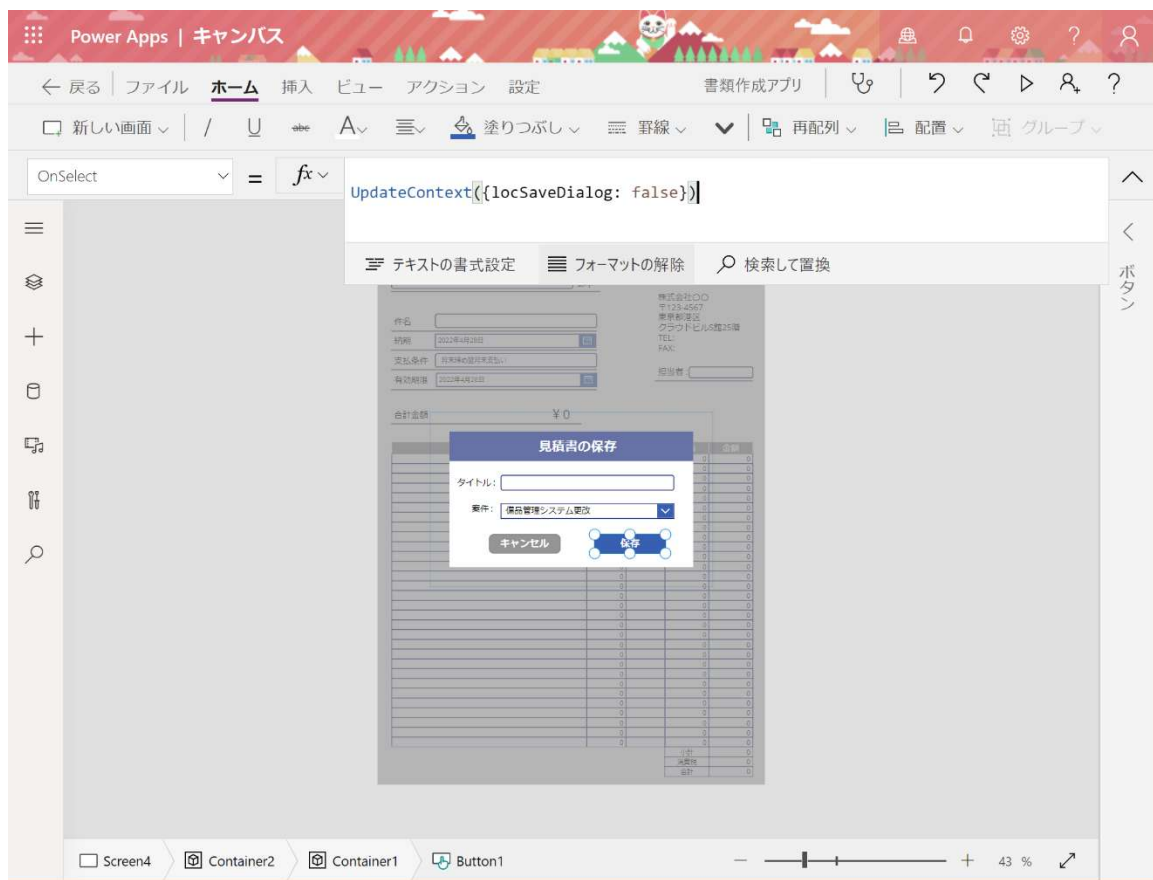
画面 13-20

⑥のキャンセルボタンを選択⇒[OnSelect]プロパティを選択⇒
`UpdateContext({locSaveDialog:false})`を入力します (画面 13-21)。

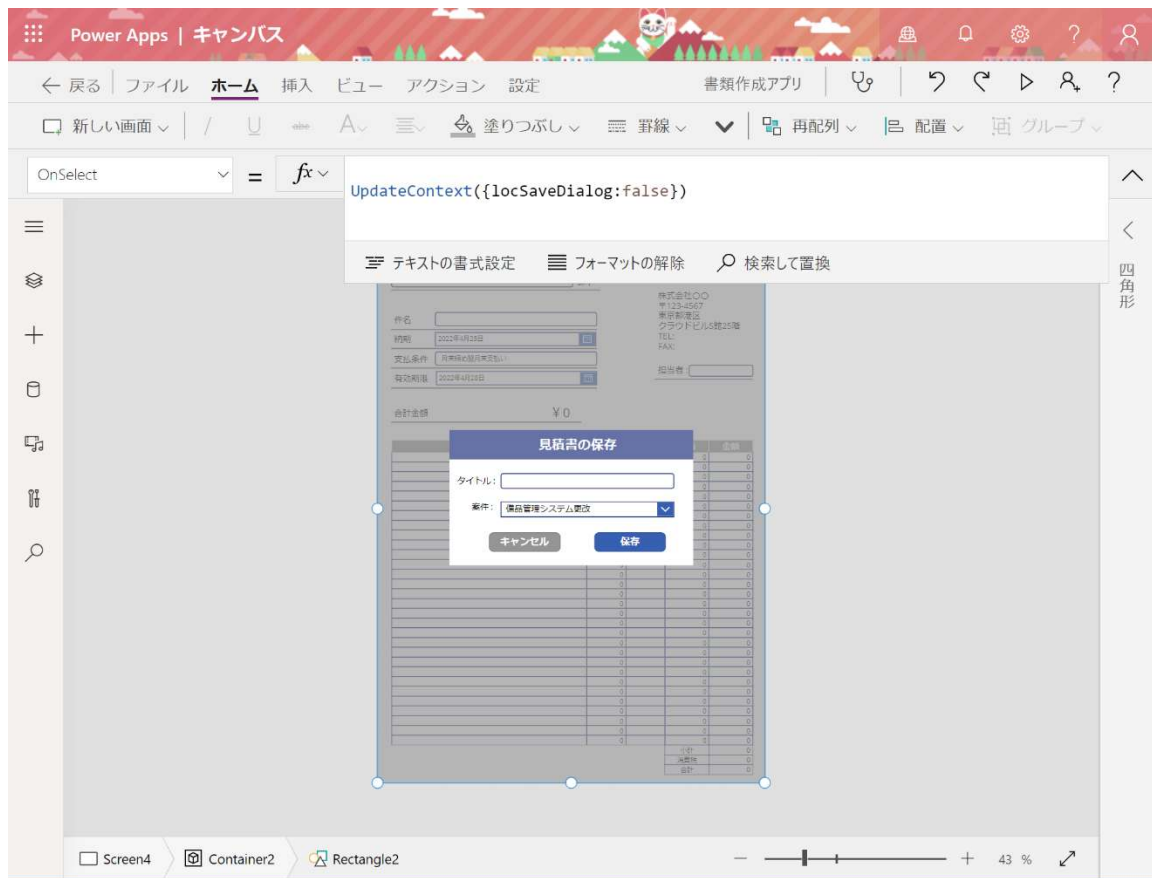


画面 13-21

同様に、⑦の保存ボタンと⑨の背景の [OnSelect] プロパティにも `UpdateContext({locSaveDialog: false});`を追加します (画面 13-22、13-23)。



画面 13-22



画面 13-23

次に、①～⑨のすべてのコントロールの [Visible] プロパティに locSaveDialog を入力します。

この時点でキャンセルボタンを押すと、ダイアログが閉じます。

ダイアログを閉じたら、B-③の保存アイコンを選択⇒ [OnSelect] プロパティを選択⇒ UpdateContext({locSaveDialog:true}) を入力します。

これで保存アイコンを押すと、ダイアログが開きます。

保存機能の実装

ツリービューから保存ボタンのコントロールを選択⇒ [OnSelect] プロパティを選択⇒既に入力されているコードの前の行に以下のコードを追加

します。コード入力時は、[関数の入力補助] フォルダに格納されている<Chapter13-2_パラメータシート.xlsx>をご参照ください。[〇〇のコントロール名] となっている部分是对应するコントロール名を入力してください (画面 13-24、13-25)。

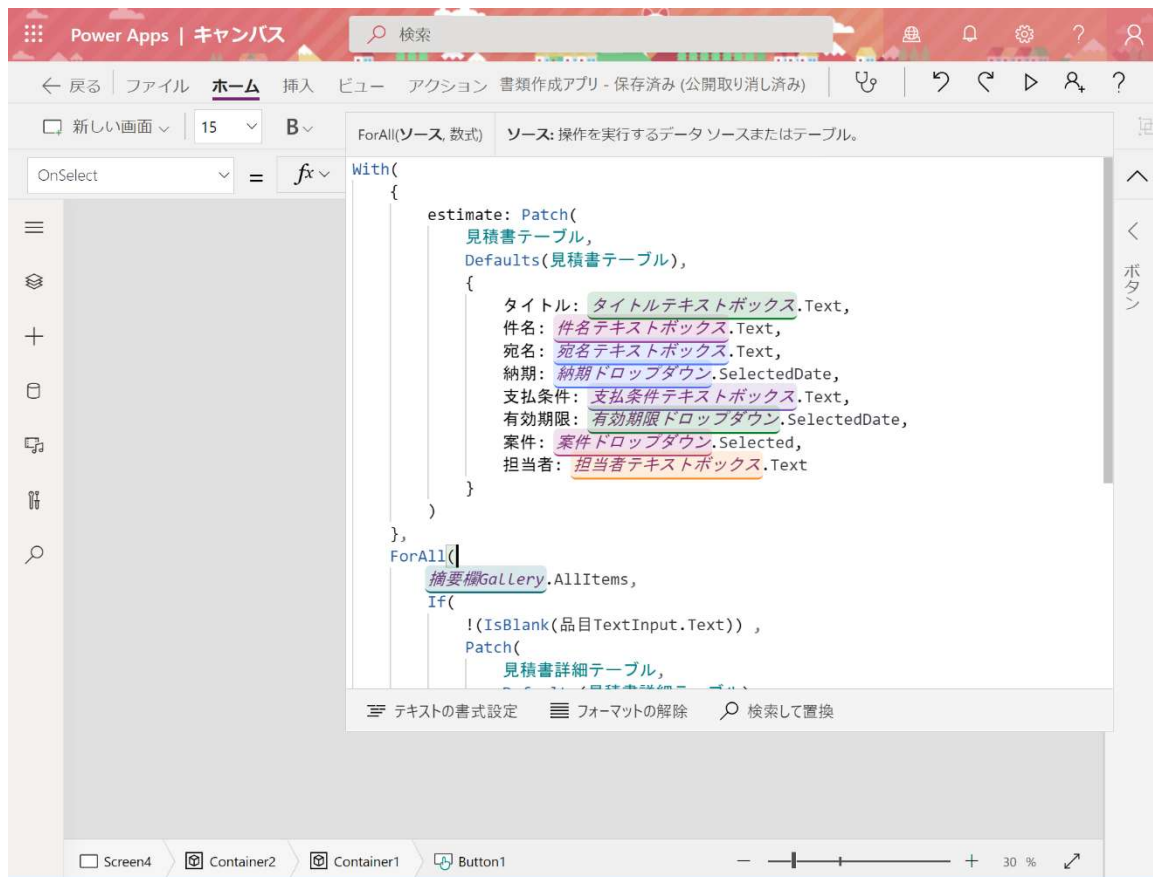
```
With(
{
    estimate: Patch(
        見積書テーブル,
        Defaults(見積書テーブル),
        {
            タイトル: [D-③のコントロール名].Text,
            件名: [A-⑥のコントロール名].Text,
            宛名: [A-②のコントロール名].Text,
            納期: [A-⑨のコントロール名].SelectedDate,
            支払条件: [A-⑫のコントロール名].Text,
            有効期限: [A-⑮のコントロール名].SelectedDate,
            案件: [D-⑤のコントロール名].Selected,
            担当者: [A-⑳のコントロール名].Text
        }
    )
},
ForAll(
    [C-⑥のコントロール名].AllItems,
    If(
        !(IsBlank(品目 TextInput.Text)),
        Patch(
            見積書詳細テーブル,
            Defaults(見積書詳細テーブル),
            {
                品目: [C-⑦のコントロール名].Text,
                単価: Value([C-⑩のコントロール名].Text),
                数量: Value([C-⑧のコントロール名].Text),
                単位: [C-⑨のコントロール名].Text,
                行番号: index,
                見積書: estimate
            }
        )
    )
})
```



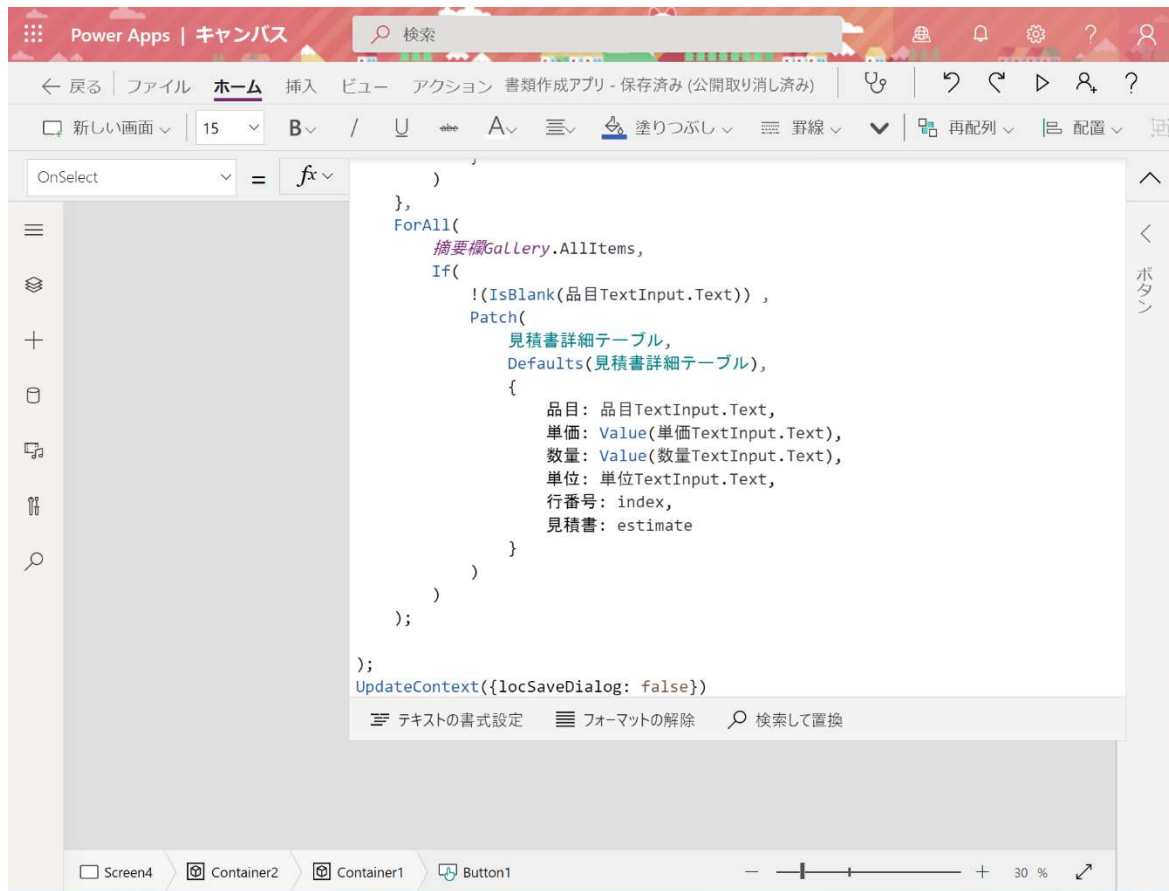
```

    )
);
);

```



画面 13-24



画面 13-25

5. 呼び出し機能の実装

見積書を開くダイアログの作成

保存ダイアログと同様に、同じ画面内にコントロールを配置し、画面 13-26 のような保存した見積書を開くためのダイアログを作成します。



見積書

御中

株式会社〇〇
〒123-4567
東京都港区
クラウドビルS館25階
TEL:
FAX:

担当者:

件名

納期 2022年4月28日

支払条件 月末締め翌月末支払い

有効期限 2022年4月28日

合計金額 ￥0

見積書を開く

	金額
テスト1	0
見積書A	0
見積書B	0
見積書C	0
見積書D	0
見積書E	0
見積書F	0
見積書G	0
見積書H	0
見積書I	0
見積書J	0
見積書K	0
見積書L	0
見積書M	0
見積書N	0
見積書O	0
見積書P	0
見積書Q	0
見積書R	0
見積書S	0
見積書T	0
見積書U	0
見積書V	0
見積書W	0
見積書X	0
見積書Y	0
見積書Z	0
小計	0
消費税	0
合計	0

キャンセル
開く

画面 13-26

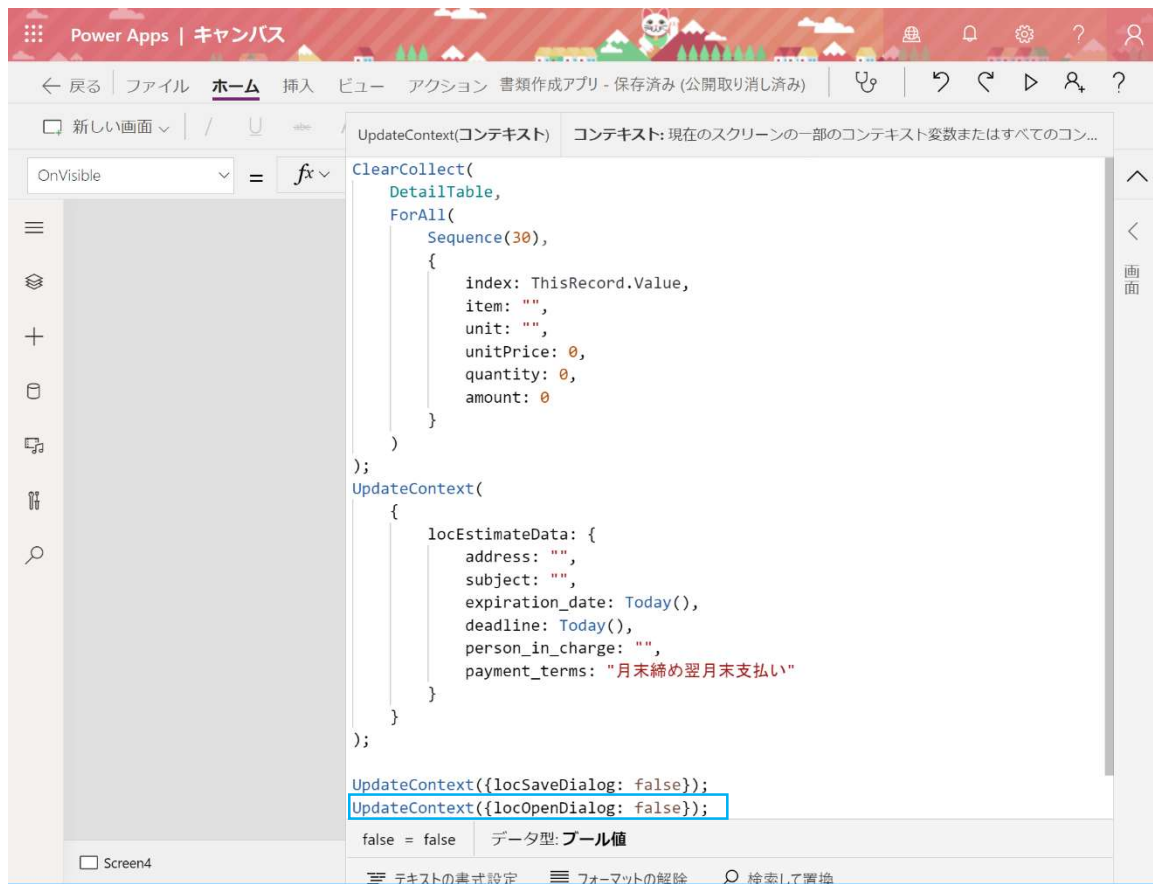
■ セクション E

- ① ラベル … ダイアログの土台として使うラベル。[Text] プロパティを選択⇒“”と入力。[Fill] プロパティを選択⇒White と入力。
- ② ギャラリー … 開きたい見積書の一覧を表示するギャラリー。
[Items] プロパティを選択⇒見積書テーブルと入力。
[TemplateFill] プロパティ⇒If(ThisItem.IsSelected, RGBA(149, 149, 149, 1))と入力。[BorderThickness] プロパティ⇒1 と入力し枠線をつける。ツリービューからコントロール名を「見積書一覧 Gallery」に変更します。
- ③ ラベル … 見積書のタイトルを表示するラベル。②のギャラリー内に配置する。[Text] プロパティを入力⇒ThisItem. タイトルと入力。
- ④ ボタン … キャンセルボタン。[Text] プロパティを入力⇒“キャンセル”と入力。
- ⑤ ボタン 開くボタン。[Text] プロパティを入力⇒“開く”と入力。
- ⑥ ラベル … “見積書を開く”のテキストを表示するラベル。
[Text] プロパティを入力⇒“見積書を開く”と入力。背景色を適宜変更する。([Fill] プロパティ)
- ⑦ 四角形 … ダイアログの背景として表示する。[Fill] プロパティを選択⇒RGBA(149, 149, 149, 0.7)を入力。ここでも D-⑨でやったように、コントロールを右クリック⇒[再配列] ⇒[背面へ移動] を選択します。

ボタンを押してダイアログを開閉する

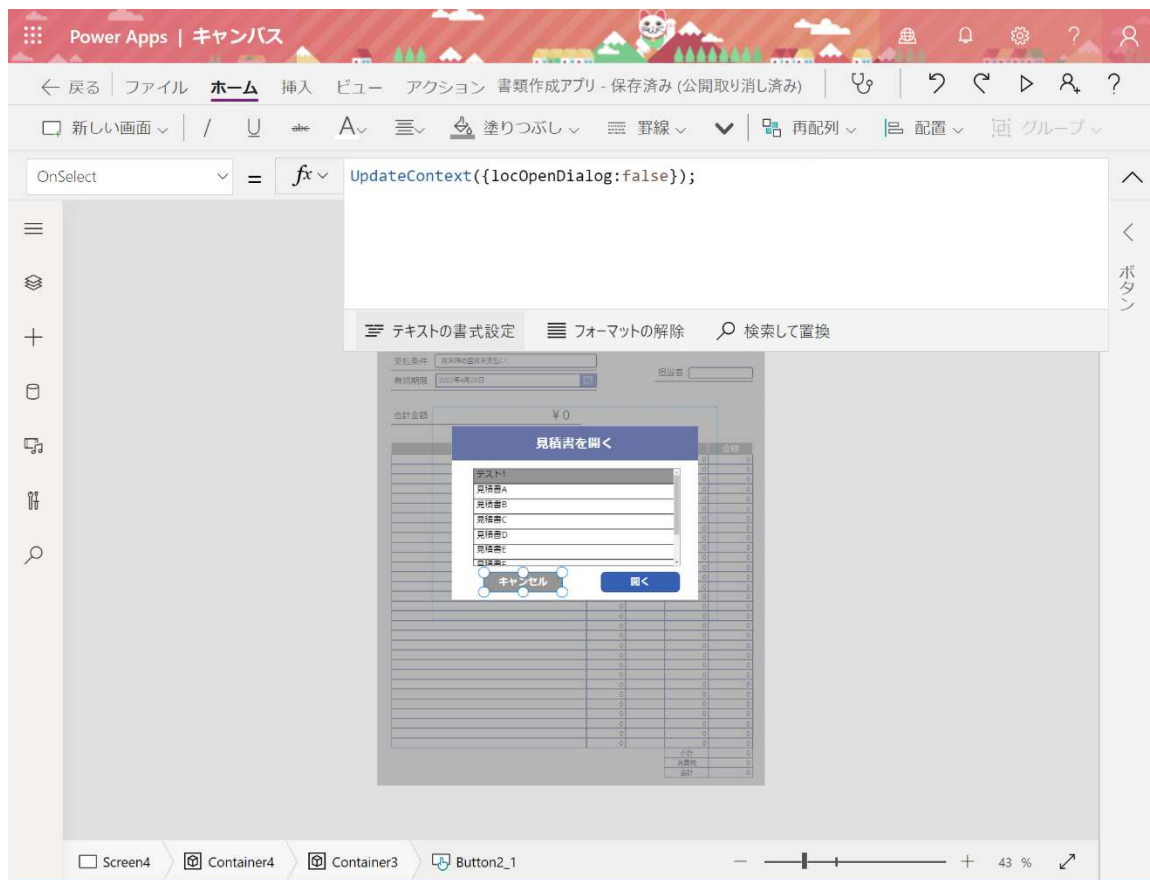
保存ダイアログと同様に、Bool 型変数を使って表示/非表示の切り替えを実装します。

画面の [OnVisible] プロパティを選択⇒UpdateContext({locOpenDialog: false});を追加します (画面 13-27)。



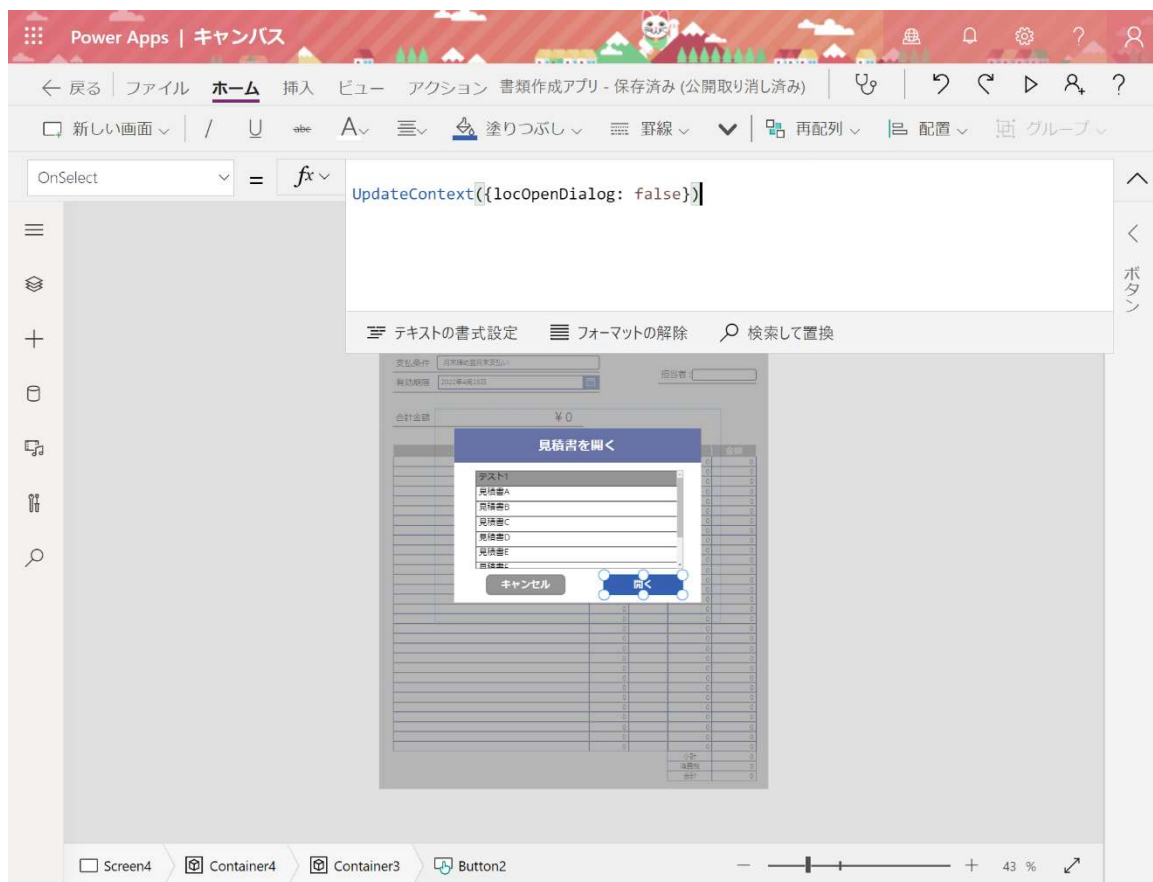
画面 13-27

④のキャンセルボタンを選択⇒[OnSelect]プロパティを選択⇒
UpdateContext({locOpenDialog: false});を追加します (画面 13-28)。

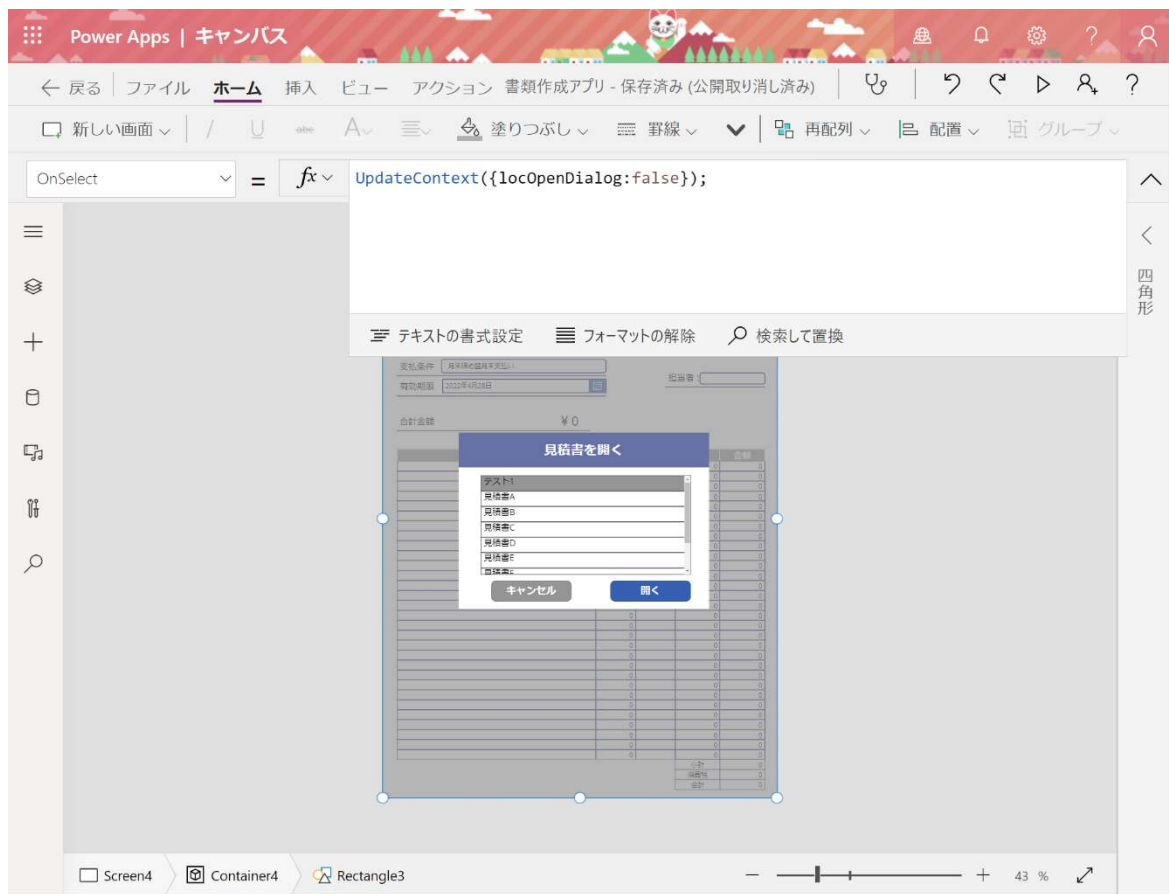


画面 13-28

同様に、⑤の開くボタンと⑦の背景の [OnSelect] プロパティにも `UpdateContext({locOpenDialog: false});` を追加します (画面 13-29、13-30)。



画面 13-29



画面 13-30

次に、①～⑦のすべてのコントロールの [Visible] プロパティに locOpenDialog を入力します。

この時点でキャンセルボタンを押すと、ダイアログが閉じます。

ダイアログを閉じたら、B-②のフォルダアイコンを選択⇒ [OnSelect] プロパティを選択⇒ UpdateContext({locOpenDialog:true}) を入力します。

これでフォルダアイコンを押すと、ダイアログが開きます。

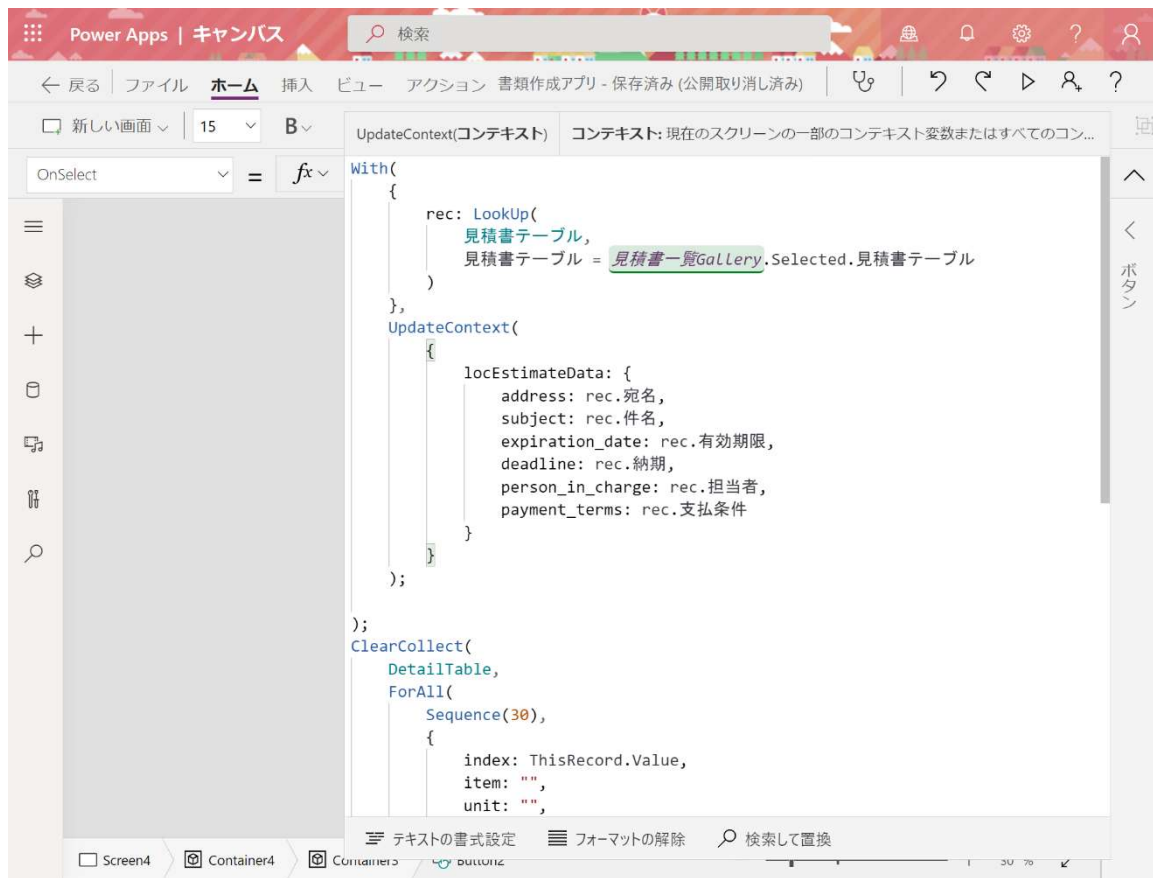
呼び出し機能の実装

ツリービューから開くボタンのコントロールを選択⇒ [OnSelect] プロ

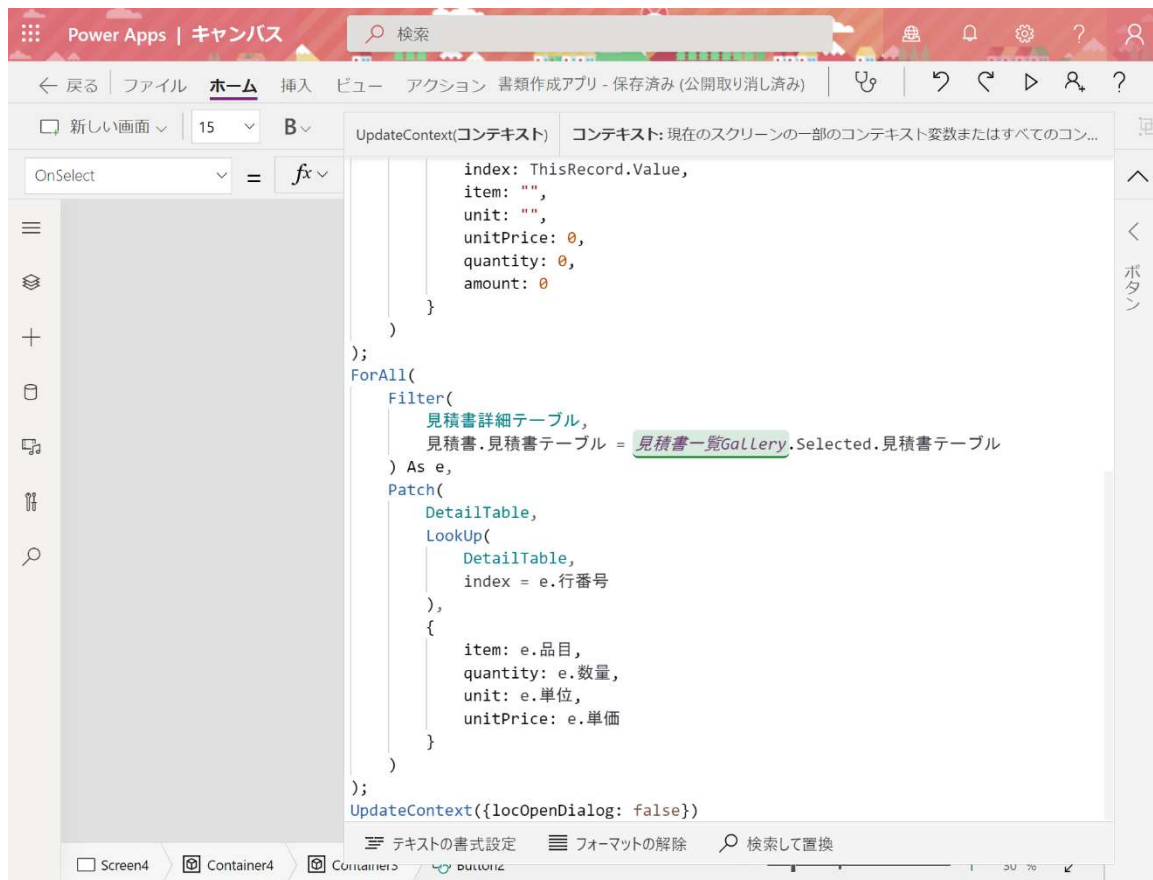
パティを選択⇒既に入力されているコードの前の行に以下のコードを追加します（画面 13-31、13-32）。コード入力時は、[関数の入力補助] フォルダに格納されている<Chapter13-2_パラメータシート.xlsx>をご参照ください。[〇〇のコントロール名] となっている部分是对应するコントロール名を入力してください。

```
With(
    {
        rec: LookUp(
            見積書テーブル,
            見積書テーブル = [E-②のコントロール名].Selected.見積書テーブル
        )
    },
    UpdateContext(
        {
            locEstimateData: {
                address: rec.宛名,
                subject: rec.件名,
                expiration_date: rec.有効期限,
                deadline: rec.納期,
                person_in_charge: rec.担当者,
                payment_terms: rec.支払条件
            }
        }
    );
);
ClearCollect(
    DetailTable,
    ForAll(
        Sequence(30),
        {
            index: ThisRecord.Value,
            item: "",
            unit: "",
            unitPrice: 0,
            quantity: 0,
            amount: 0
        }
    )
);
```

```
    }  
  )  
);  
ForAll(  
  Filter(  
    見積書詳細テーブル,  
    見積書.見積書テーブル = [E-②のコントロール名].Selected.見積書テーブル  
  ) As e,  
  Patch(  
    DetailTable,  
    LookUp(  
      DetailTable,  
      index = e.行番号  
    ),  
    {  
      item: e.品目,  
      quantity: e.数量,  
      unit: e.単位,  
      unitPrice: e.単価  
    }  
  )  
);
```



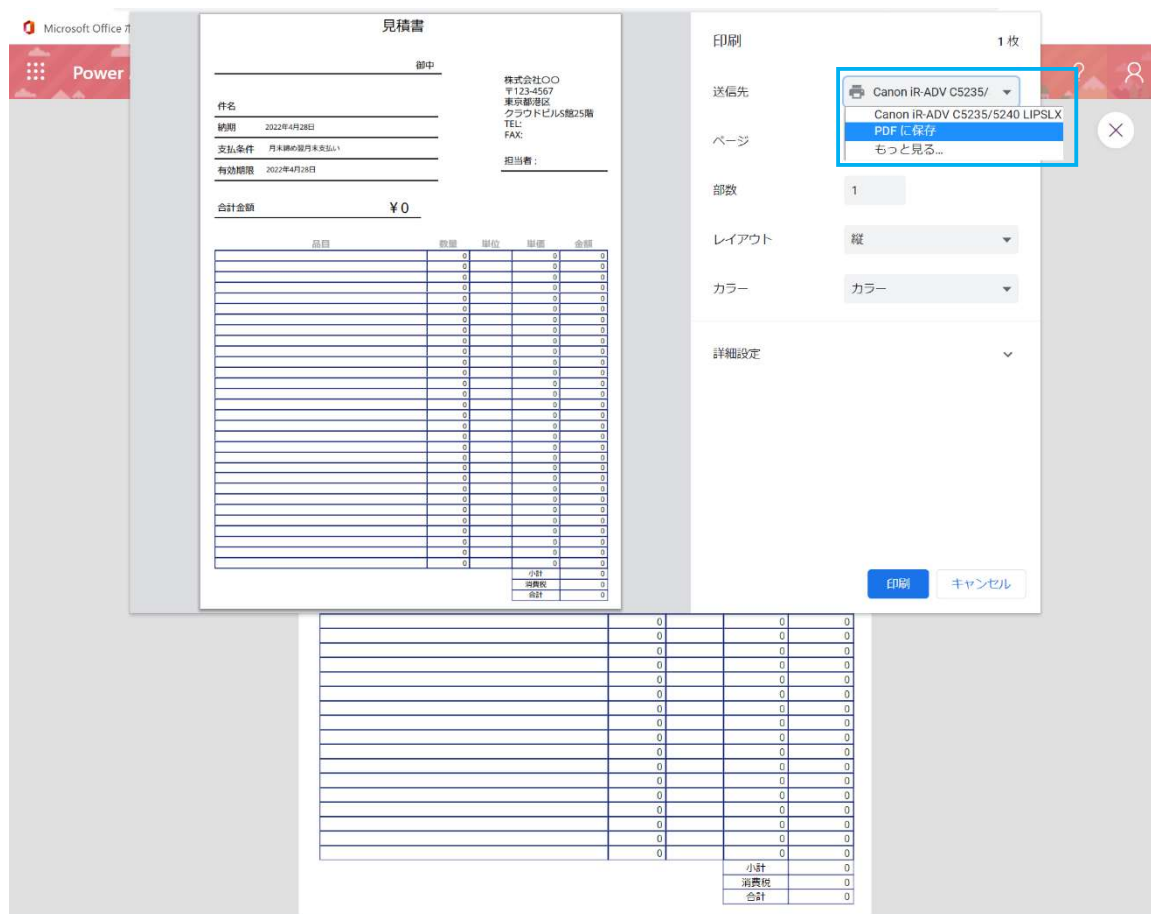
画面 13-31



画面 13-32

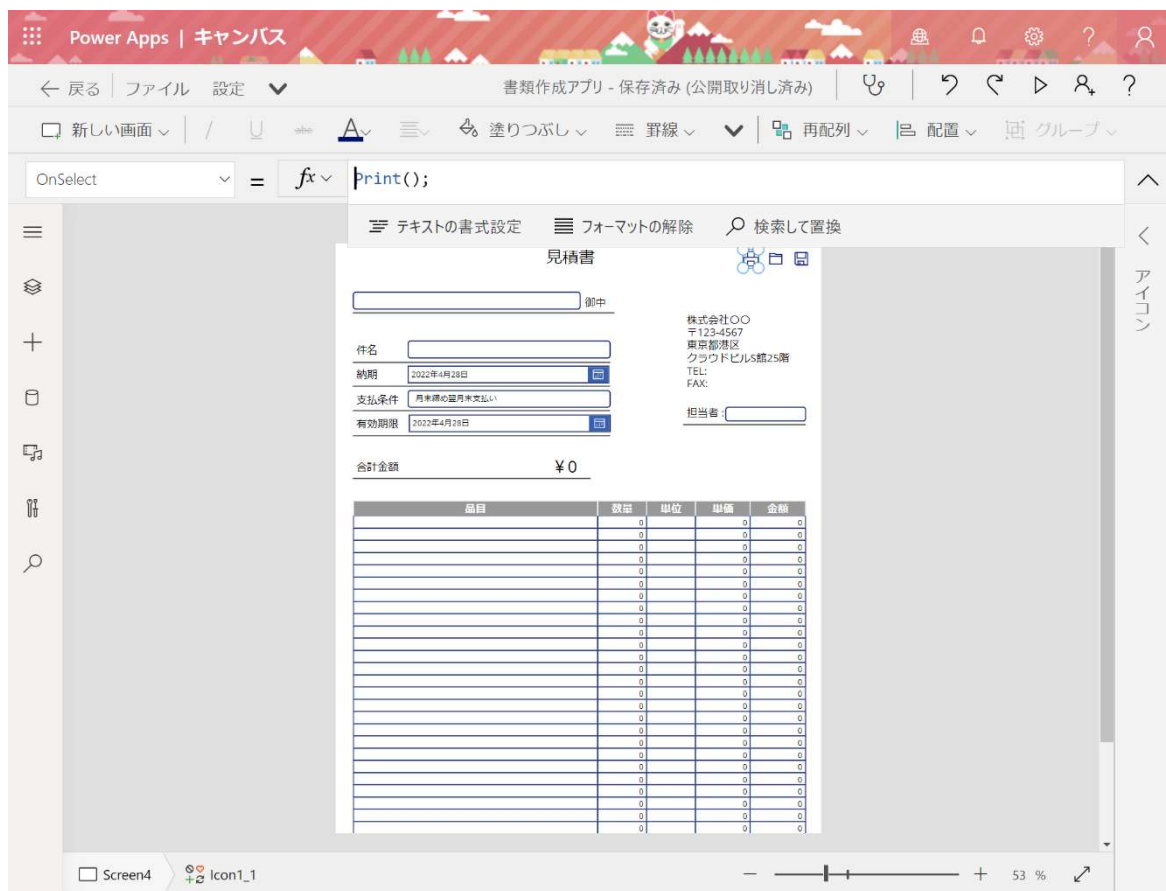
6. 印刷機能の実装

Power Apps には `Print()` という関数が用意されており、`Print` 関数で画面の印刷や PDF 出力ができます。(画面 13-33)。詳細は書籍本体 Chapter9 の Column 「キャンバスアプリに印刷機能を搭載する」をご参照ください。



画面 13-33

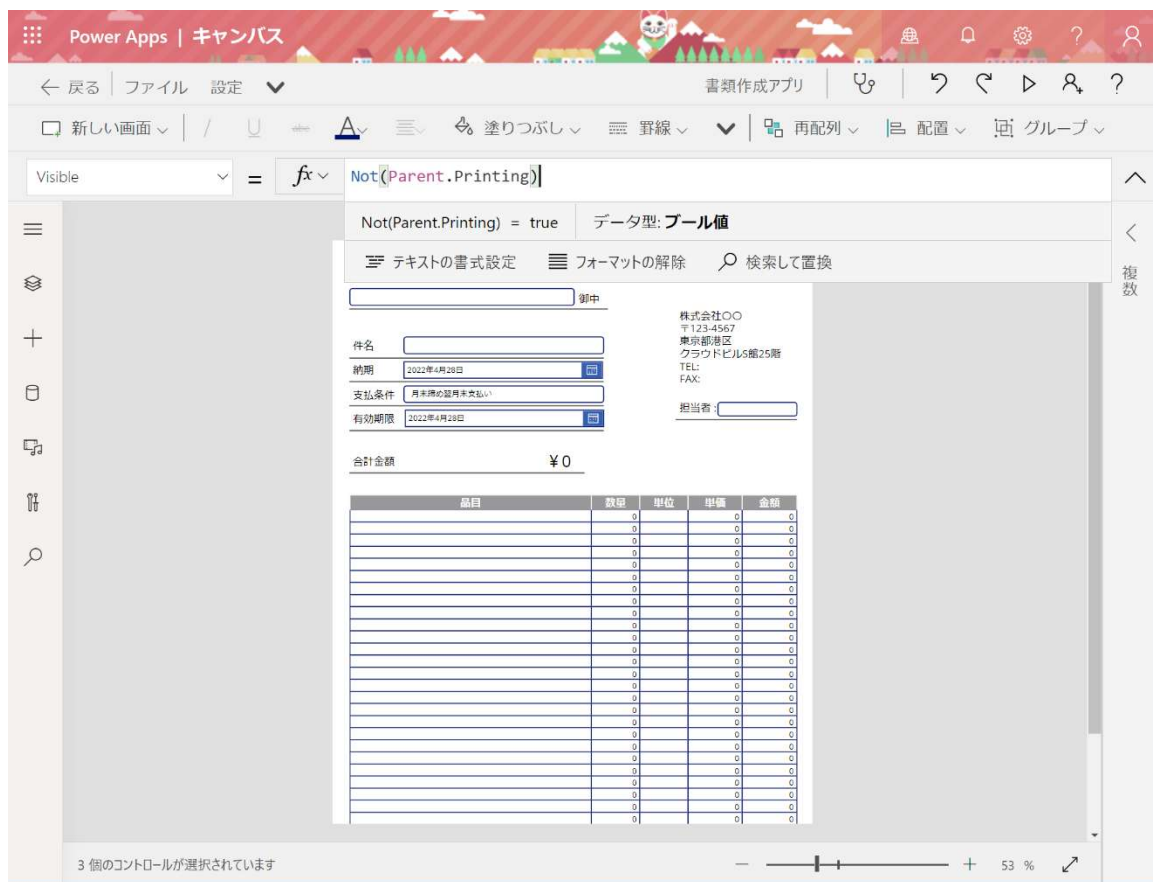
B-①の印刷アイコンを選択⇒ [OnSelect] プロパティを選択⇒Print ()を入力します (画面 13-34)。



画面 13-34

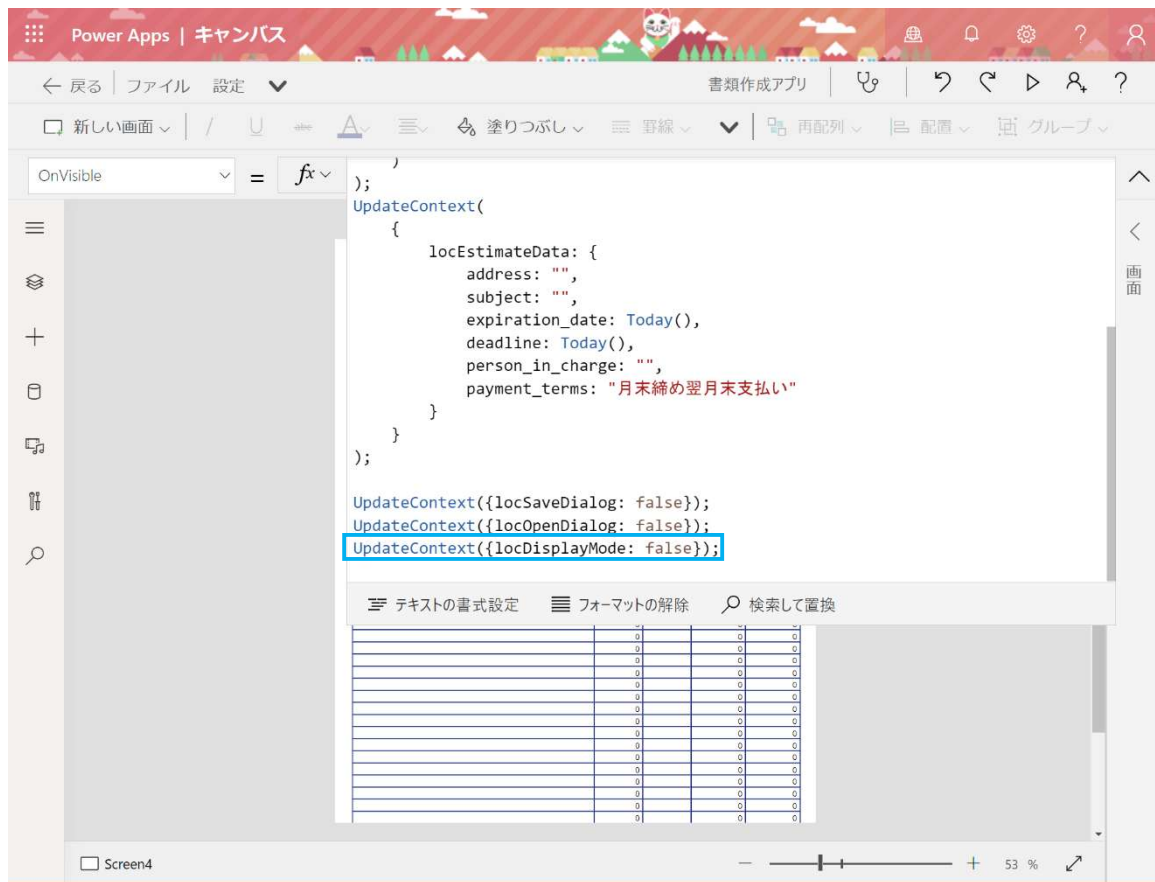
B-①、B-②、B-③のそれぞれの [Visible] プロパティに Not (Parent. Printing) を入力します (画面 13-35)。

これによって印刷のプレビューにこれらのボタンが非表示になります。



画面 13-35

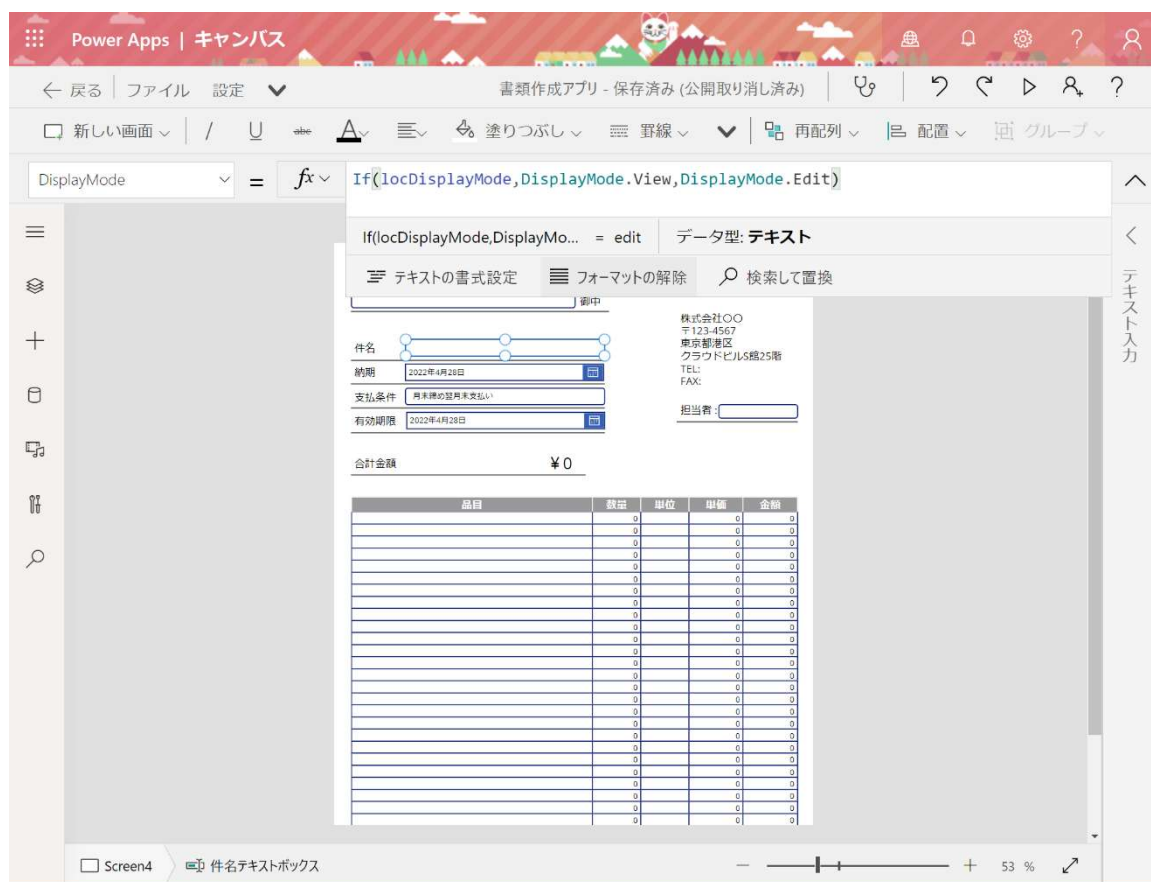
画面の [OnVisible] プロパティを選択⇒UpdateContext({locDisplay Mode: false});を追加します (画面 13-36)。



画面 13-36

A-②、A-⑥、A-⑨、A-⑫、A-⑮、A-⑳の [DisplayMode] プロパティに以下のコードを入力します (画面 13-37)。

```
If(locDisplayMode, DisplayMode.View, DisplayMode.Edit)
```

画面 13-37

B-①の印刷アイコンを選択⇒ [OnSelect] プロパティを以下のように変更します。

```
UpdateContext({locDisplayMode: true});
Print();
UpdateContext({locDisplayMode: false});
```

このようにすることで、印刷アイコンを押したときにテキスト入力欄の表示形式が編集モードからビューモードになります (画面 13-38)。

[illegible]

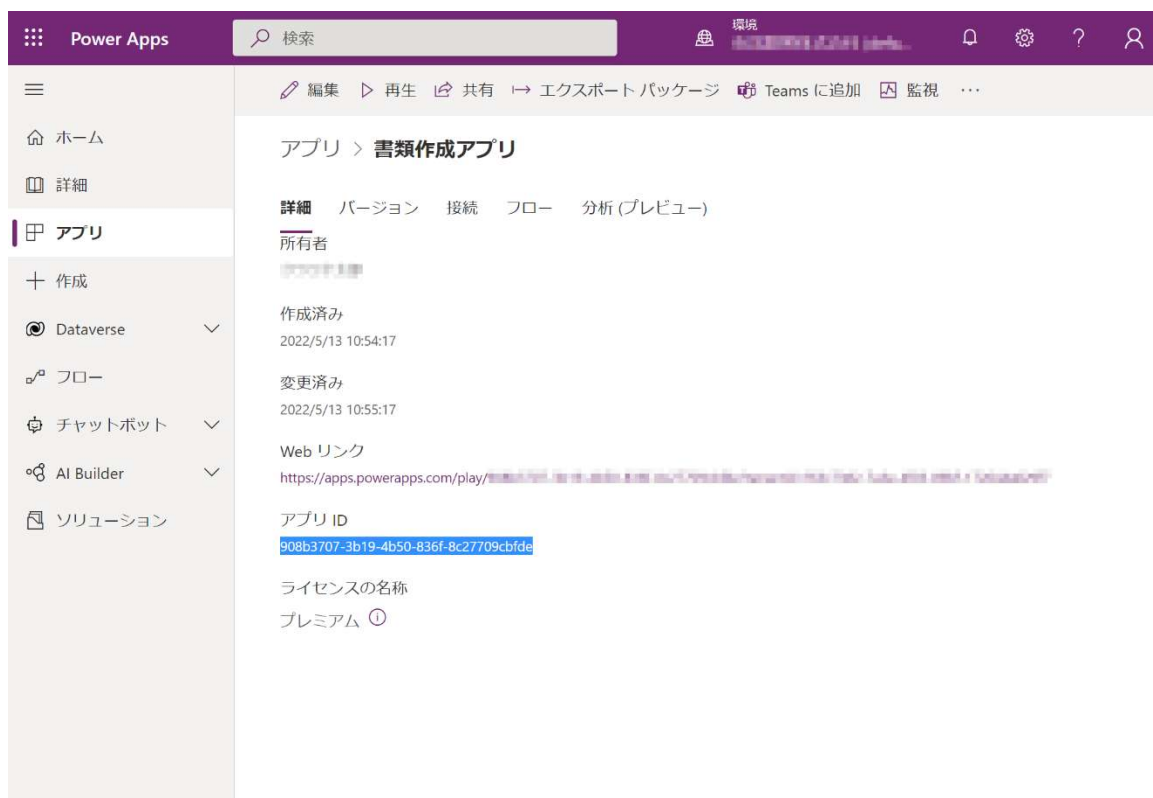
画面 13-38

案件管理アプリと見積作成アプリの連携設定（書籍本体 Chapter13-3 参考資料）

モデル駆動型アプリ内のコマンドバー（旧リボン）にキャンバスアプリの呼び出しボタンを作る方法

Power Apps メーカーポータルから呼び出し対象のキャンバスアプリのアプリ ID(Power Apps キャンバスアプリを識別、管理する情報)を確認します。

[アプリ] ⇒ [書類作成アプリ] ⇒ [...] ⇒ [詳細] で、キャンバスアプリの詳細画面を表示します。詳細画面に表示されたアプリ ID を控えます（画面 13-39）。

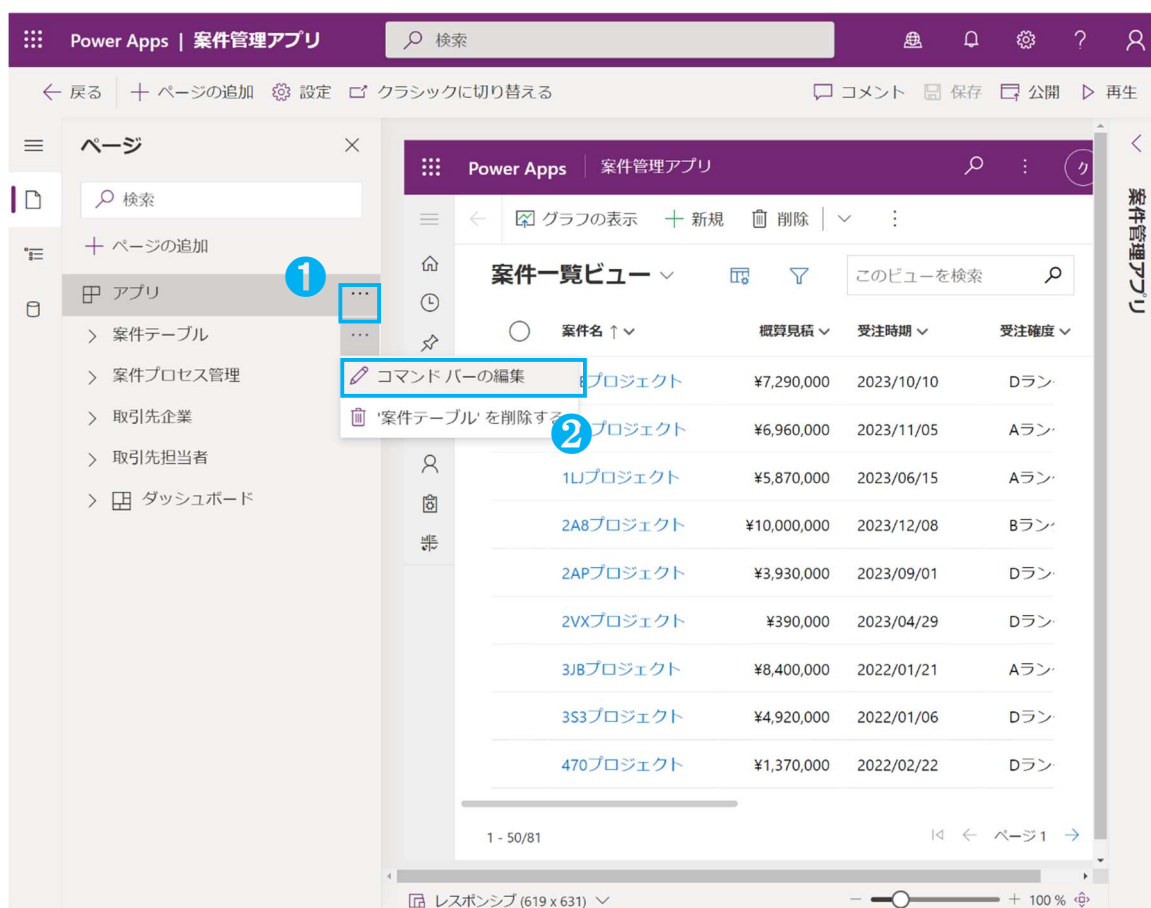


画面 13-39

次は、呼び出し元のモデル駆動型アプリを編集します。

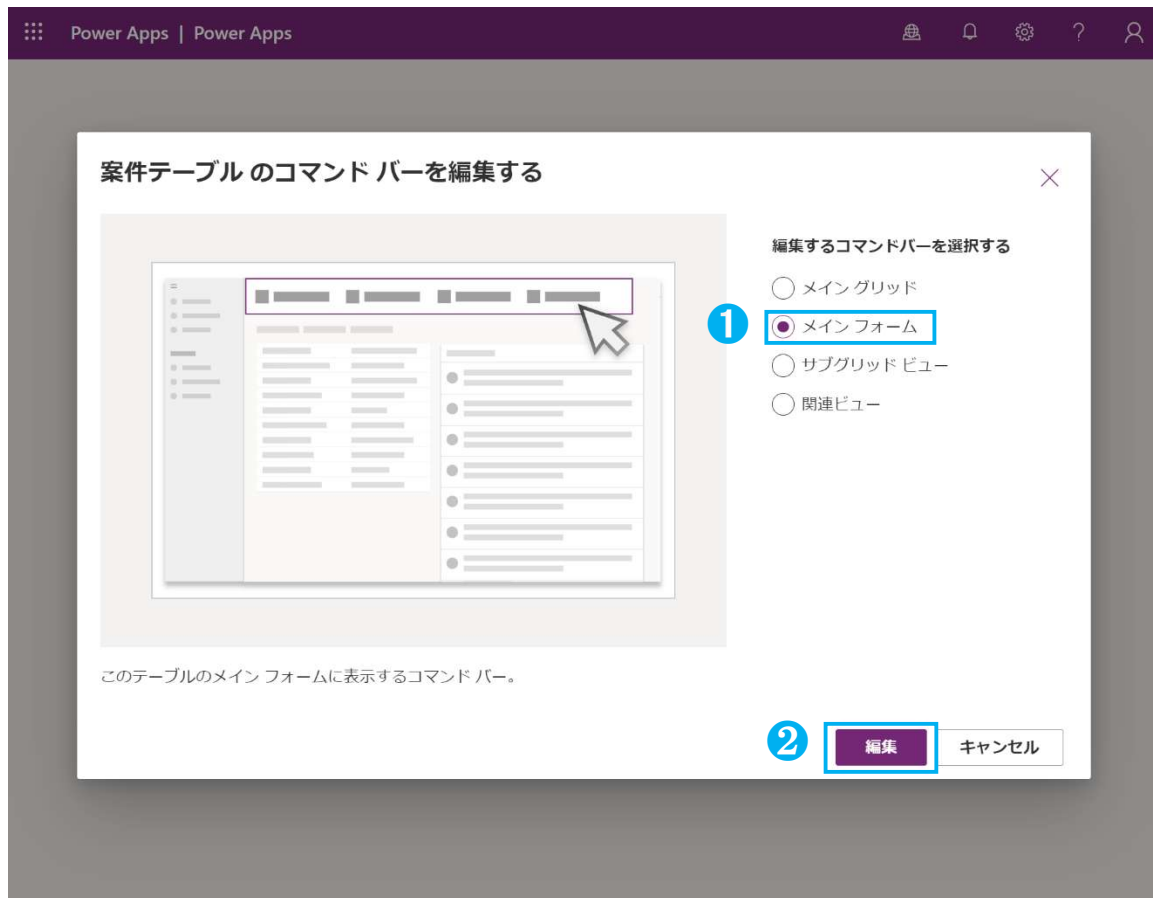
[アプリ] ⇒ [案件管理アプリ] ⇒ [...] ⇒ [編集] で、モデル駆動型アプリを開き、コマンドバーを編集します。

[案件テーブル] ⇒ [...] ⇒ [コマンドバーの編集] をクリックします (画面 13-40)。



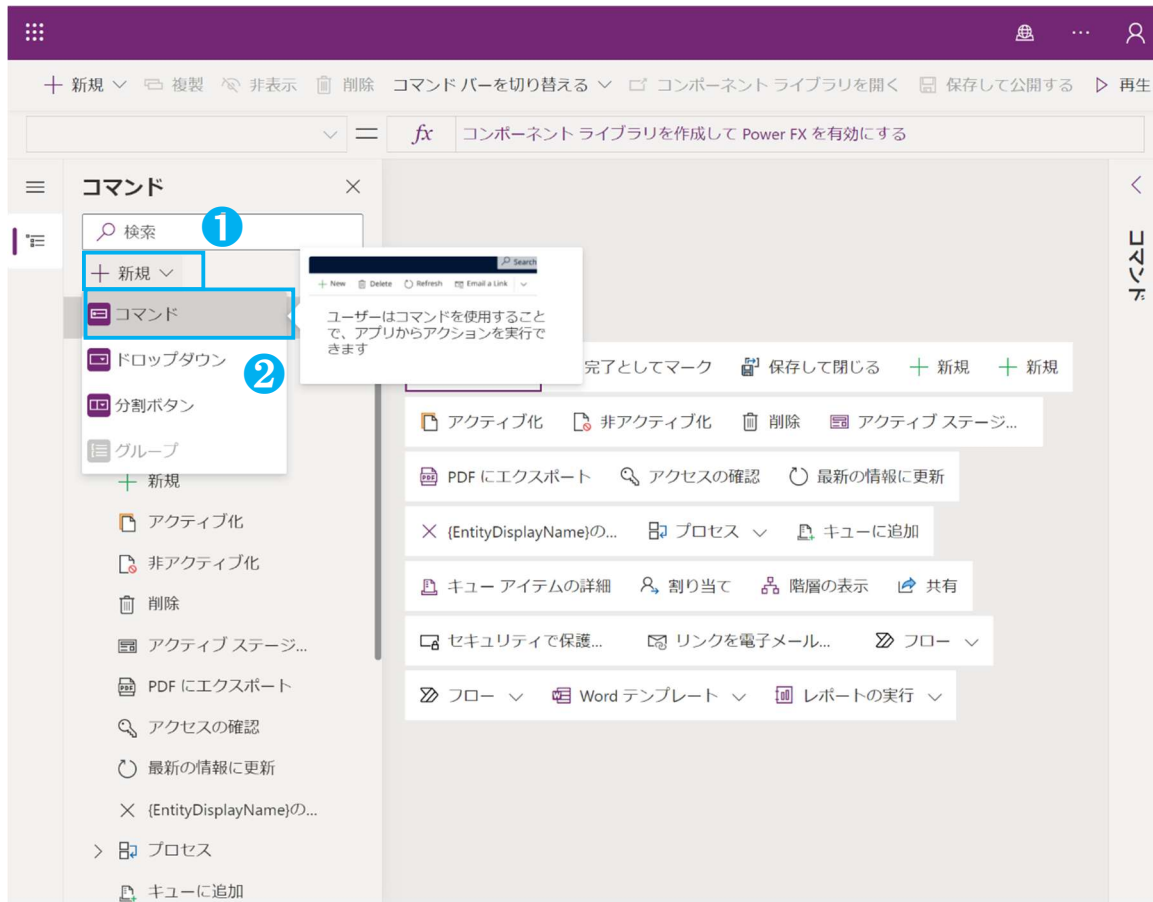
画面 13-40

今回はフォーム上のコマンドバーからキャンバスアプリを呼び出す設定をします。[メインフォーム] ⇒ [編集] をクリックします (画面 13-41)。



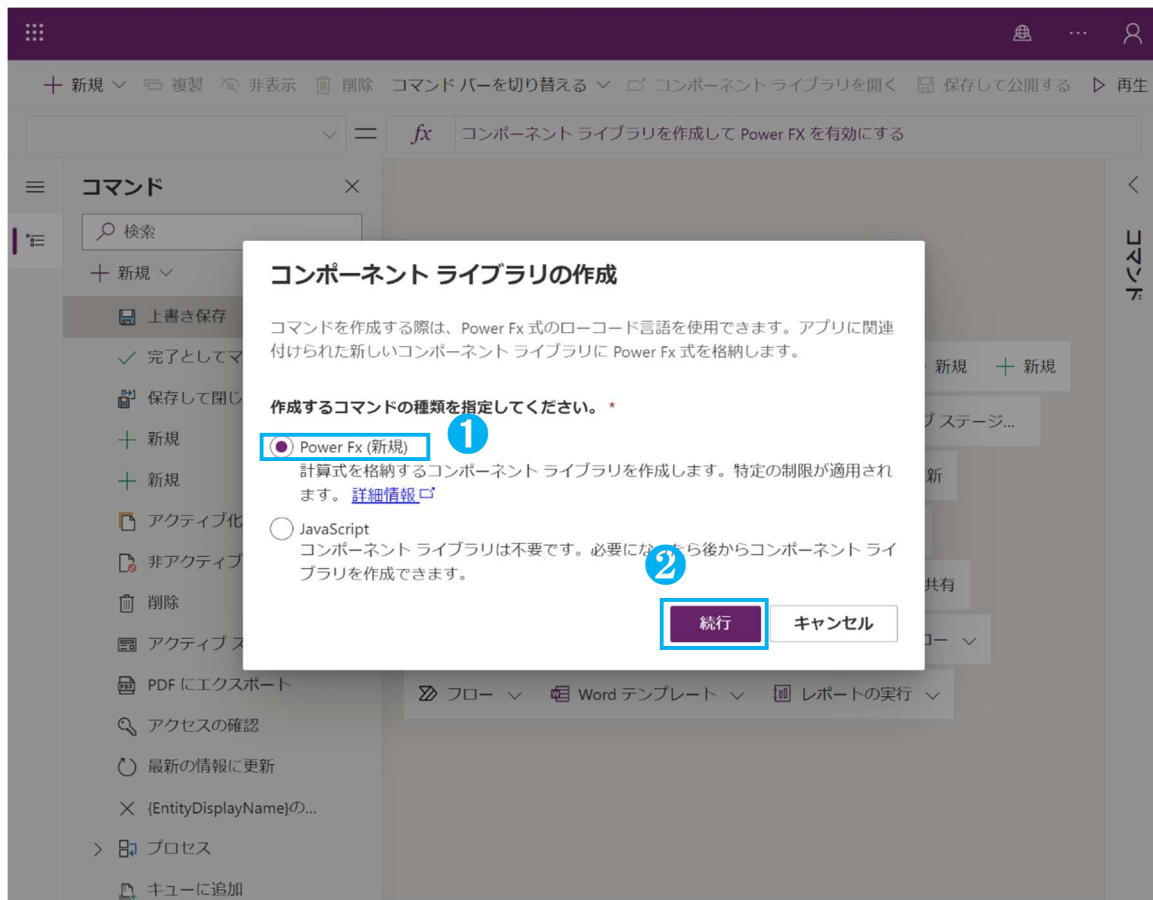
画面 13-41

[+新規] ⇒ [コマンド] をクリックします (画面 13-42)。



画面 13-42

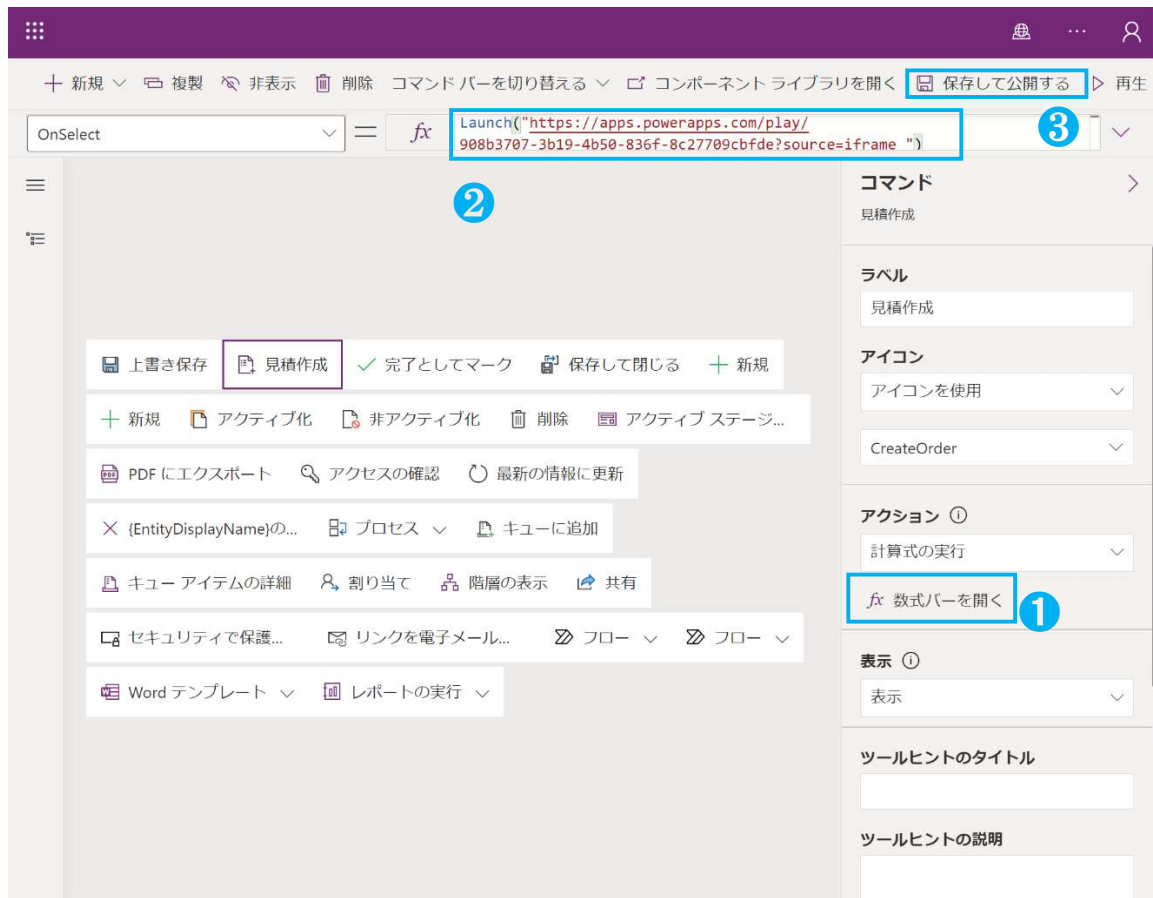
[Power Fx(新規)] ⇒ [続行] をクリックします (画面 13-43)。



画面 13-43

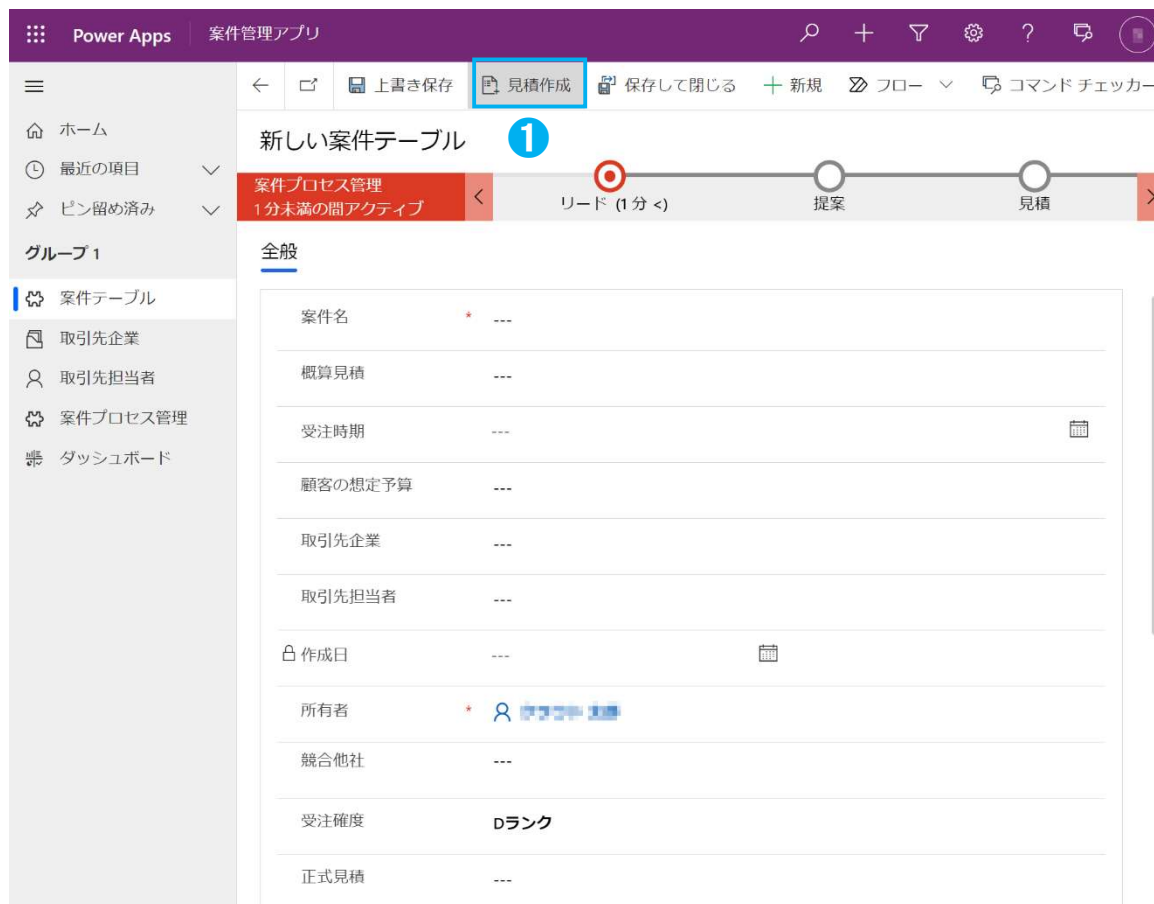
追加されたコマンドバー(初期表示は NewCommand)を編集します。次のとおりに設定し、
[保存して公開する] ⇒ [再生] をクリックします (画面 13-44)。

- ・ [ラベル] : [見積作成]
- ・ [アイコン] : [アイコンを使用] - [任意のアイコン]
- ・ [数式バーを開く] : [OnSelect] - [Launch("https://apps.powerapps.com/play/
【呼び出すキャンバスアプリのアプリ ID】?source=iframe ")]



画面 13-44

[案件テーブル] ⇒ [+新規] ⇒ [見積作成] をクリックします (画面 13-45)。



画面 13-45

キャンバスアプリ(書類作成アプリ)が表示されました (画面 13-46)。

コマンドバーからキャンバスアプリを呼び出すことで、グラフィカルな表示、入力を提供するキャンバスアプリとモデル駆動型アプリを連携させることができます。ビューやフォームなど利用シーンに合わせて、必要なコマンドバーの設置、別アプリの呼び出し設定を行い、より利便性の高いアプリに進化させることができます。

[illegible]

画面 13-46

見積書を Excel ファイルに出力する(書籍本体 Chapter13-4 参考資料)

この方法を使えば今まで手入力で行っていた Excel データ入力の自動化だけでなく、入力ミスも防げて作業品質も確保できます。

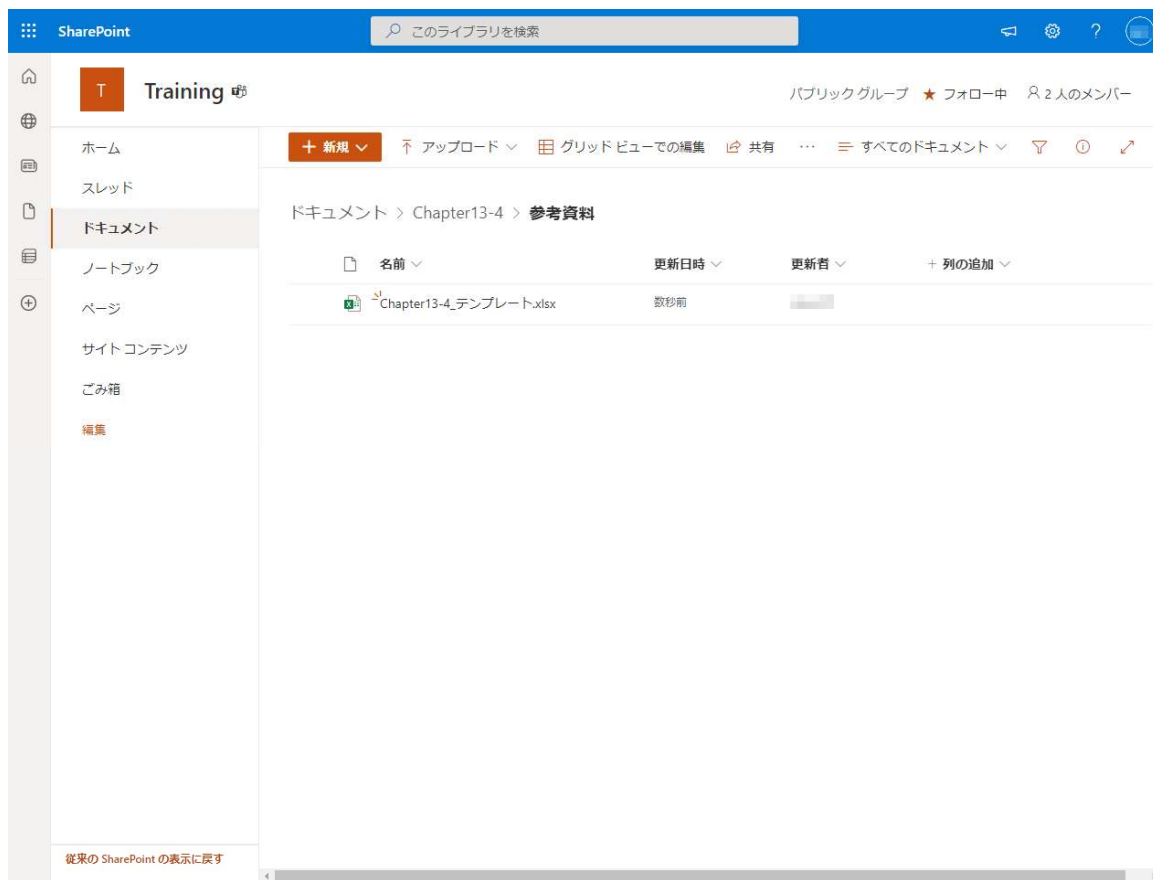
また、Chapter13-2 で作成したキャンバスアプリは PDF 出力が可能です。取引先によっては個別見積で独自の見積項目を作成する場合があります。そのようなニーズに応えるために、変更可能な書類出力の手段があると利便性が高まります。

以下のような手順で作成していきます。

1. SharePoint サイトの書類テンプレートを別フォルダにコピーする
2. Excel テーブルに [行の削除] を実行する
3. Power Automate でクラウドフローを作成する
4. 見積書を作成するアプリと手順 3. のクラウドフローを接続する
5. 実行してテストする

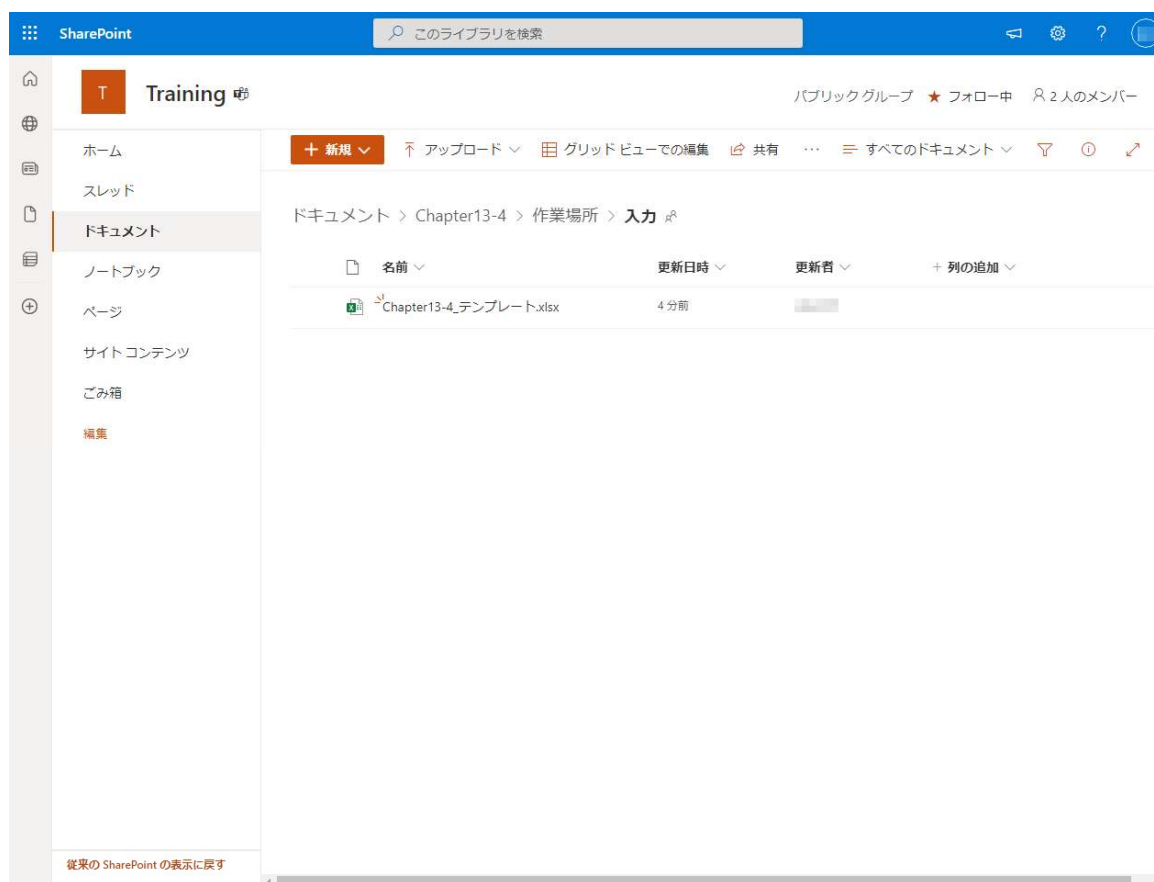
1. SharePoint サイトの書類テンプレートを別フォルダにコピーする

書籍本体の Chapter3 で作成した SharePoint サイト [Training] を利用します。[参考資料] フォルダに見積テンプレート<Chapter13-4_テンプレート.xlsx>が格納されています (画面 13-47)。



画面 13-47

こちらの見積テンプレート<Chapter13-4_テンプレート.xlsx>を〔作業場所〕⇒〔入力〕フォルダにコピーします（[画面 13-48](#)）。



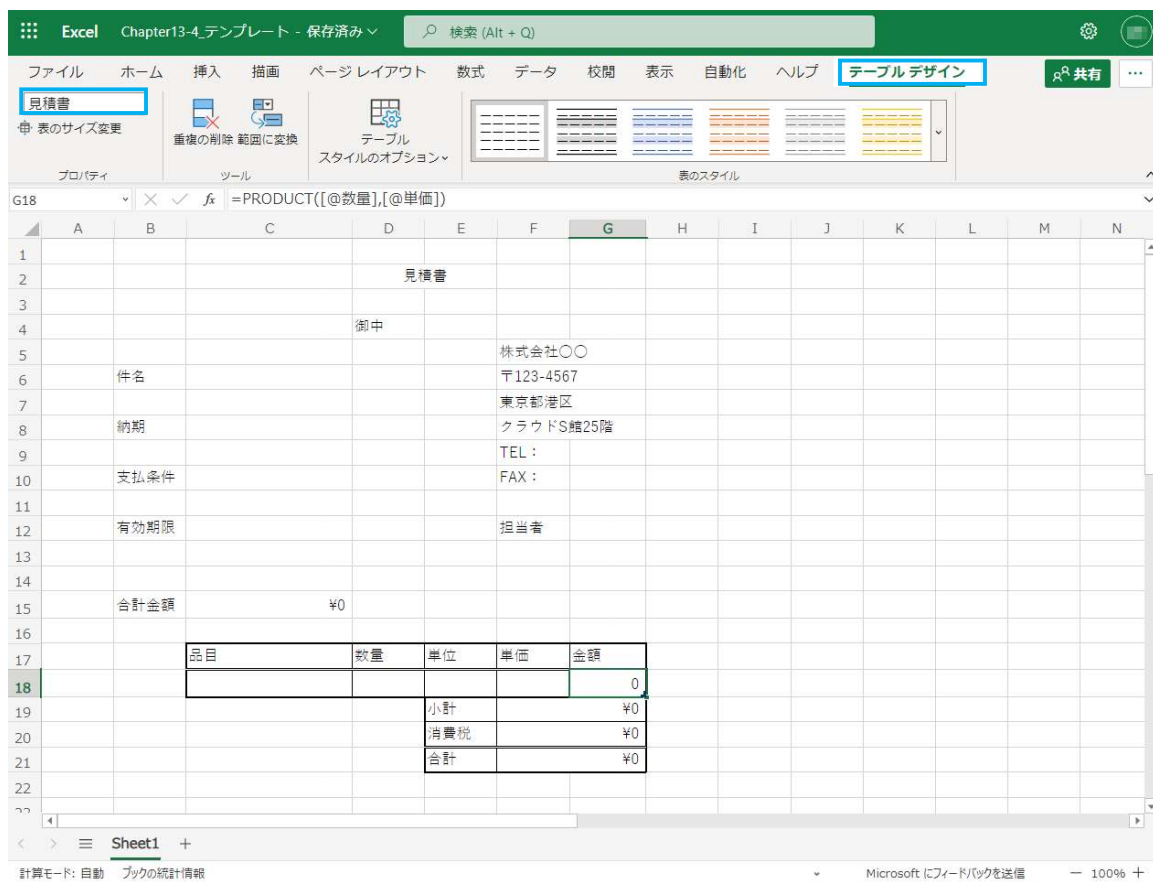
画面 13-48

2. Excel テーブルに [行の削除] を実行する

手順 1. で [入力] フォルダにコピーした見積テンプレート<Chapter13-4_テンプレート.xlsx>を開きます。

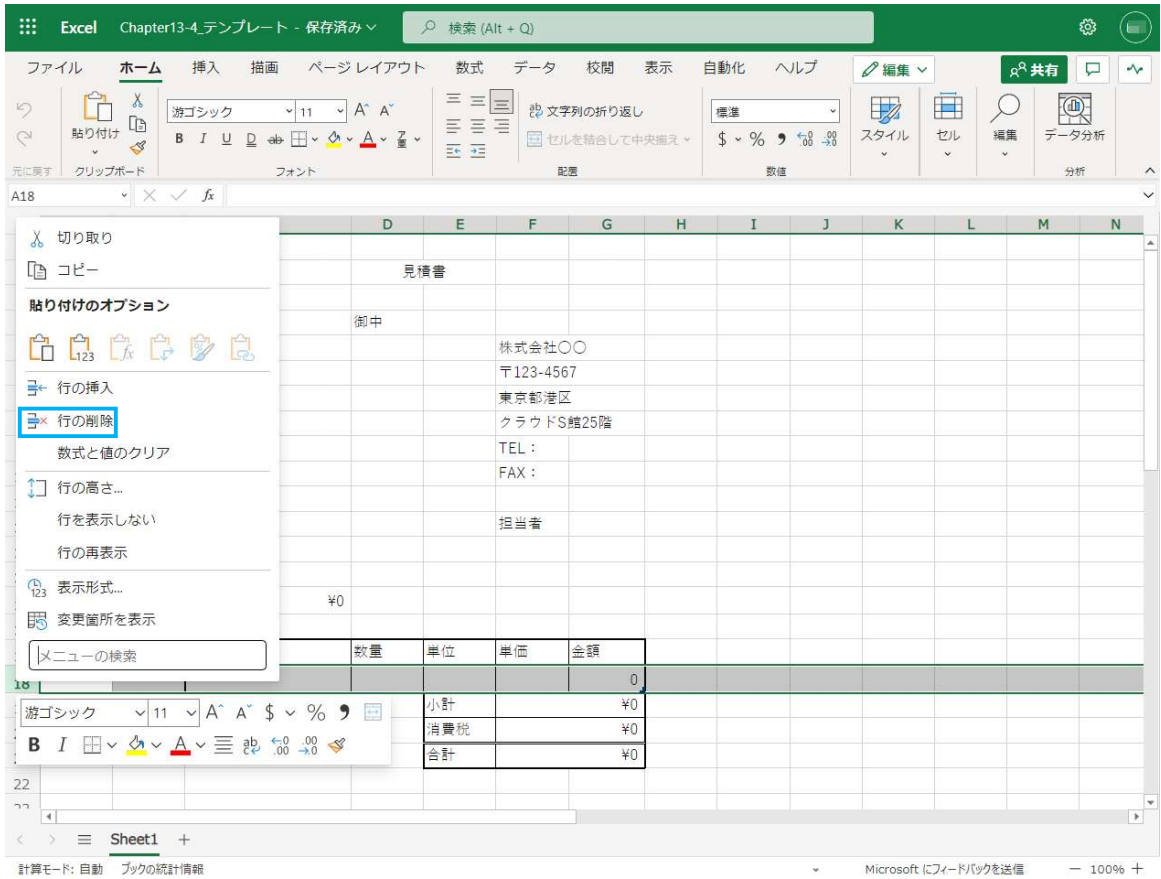
[テーブルデザイン] タブを選択して、[見積書] テーブルの [金額] [小計] [消費税] [合計] [合計金額] セルの各数式バーに数式が入力されていることを確認します (画面 13-49)。

後述の手順 3. で作成する Power Automate のクラウドフローを実行することで、[数量] [単価] セルの値が挿入されて、その値をもとに [金額] [小計] [消費税] [合計] [合計金額] セルの値が自動計算されます。

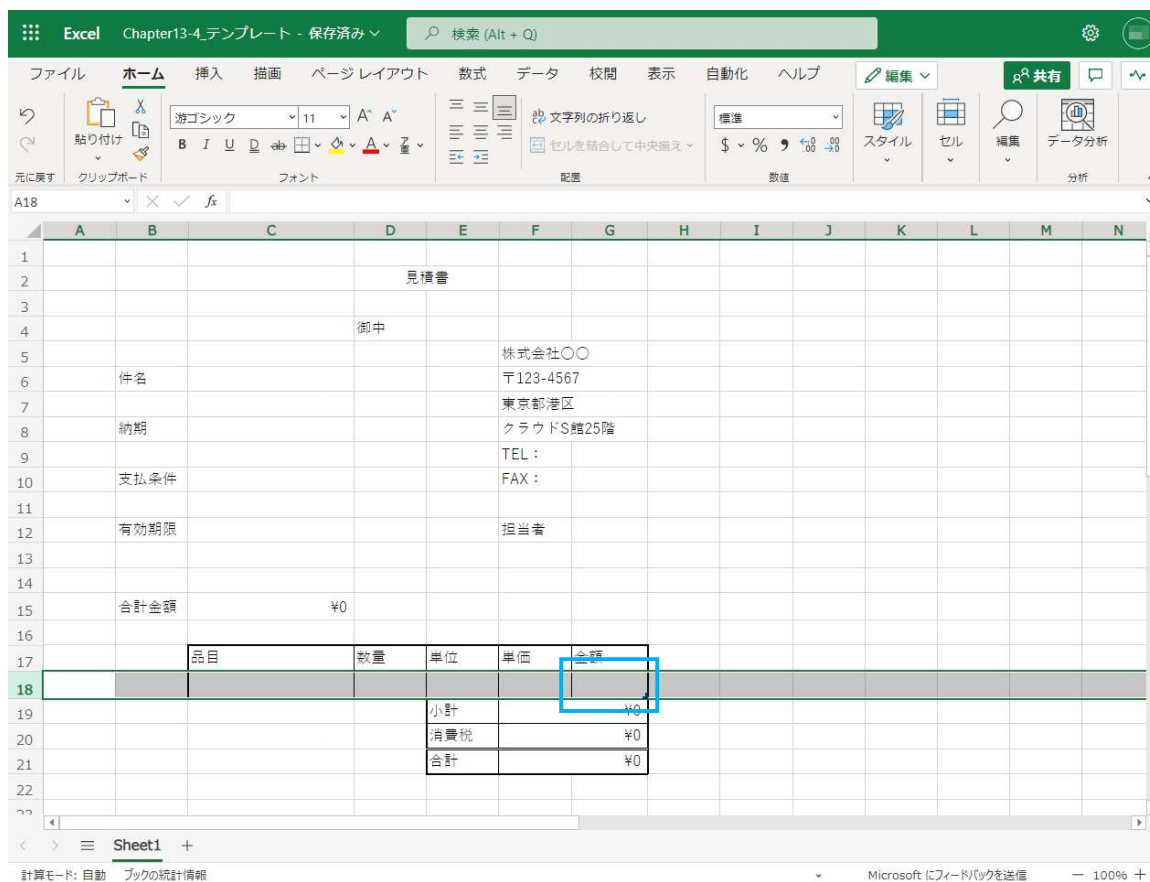


画面 13-49

「見積書」テーブルの1行目の行番号を右クリックで選択し、「[行の削除]」を実行して「金額」セルが空白になることを確認します（画面 13-50、13-51）。



画面 13-50

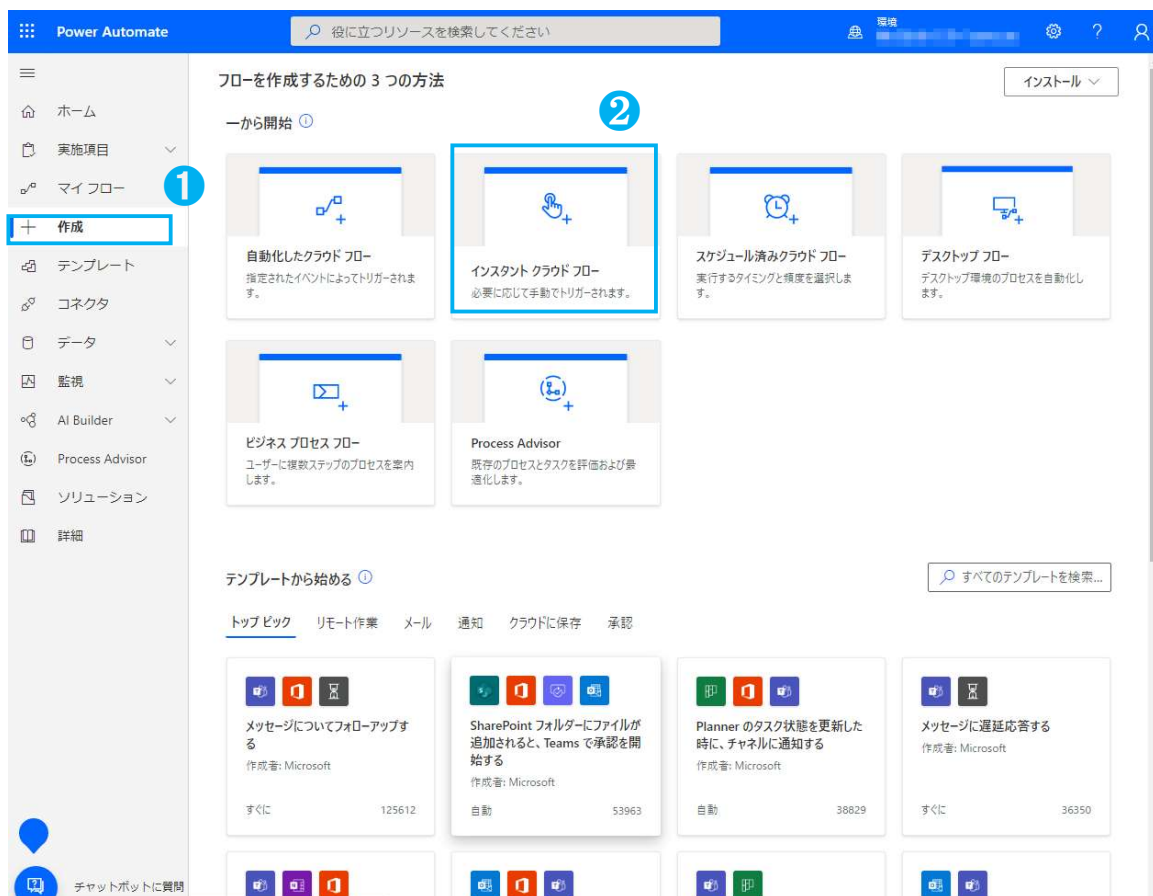


画面 13-51

こちらの「[行の削除]」を実行する理由については、「関数の入力補助」フォルダに格納されている<Chapter13-4_パラメータシート.xlsx>の「行の削除」をご参照ください。

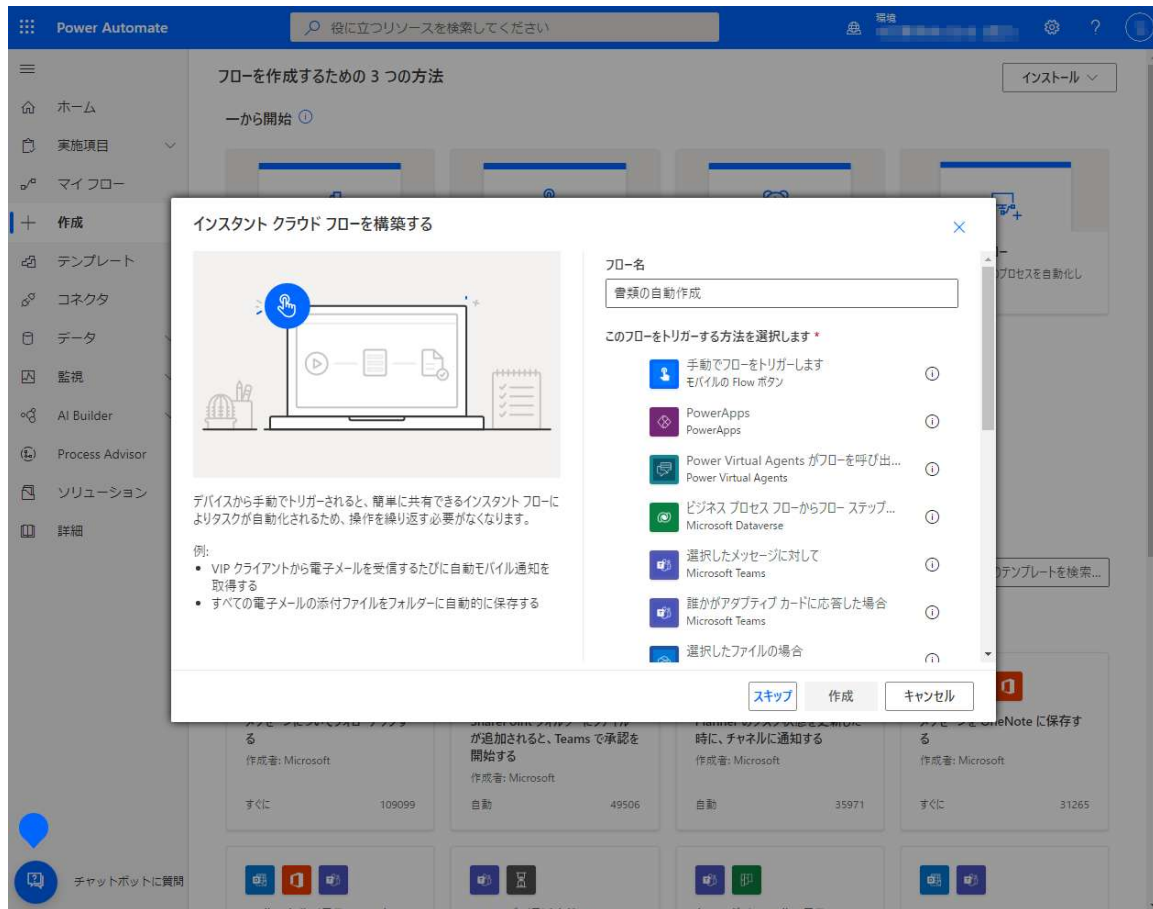
3. Power Automate でクラウドフローを作成する

[+作成]⇒[インスタントクラウドフロー]を選択します(画面 13-52)。



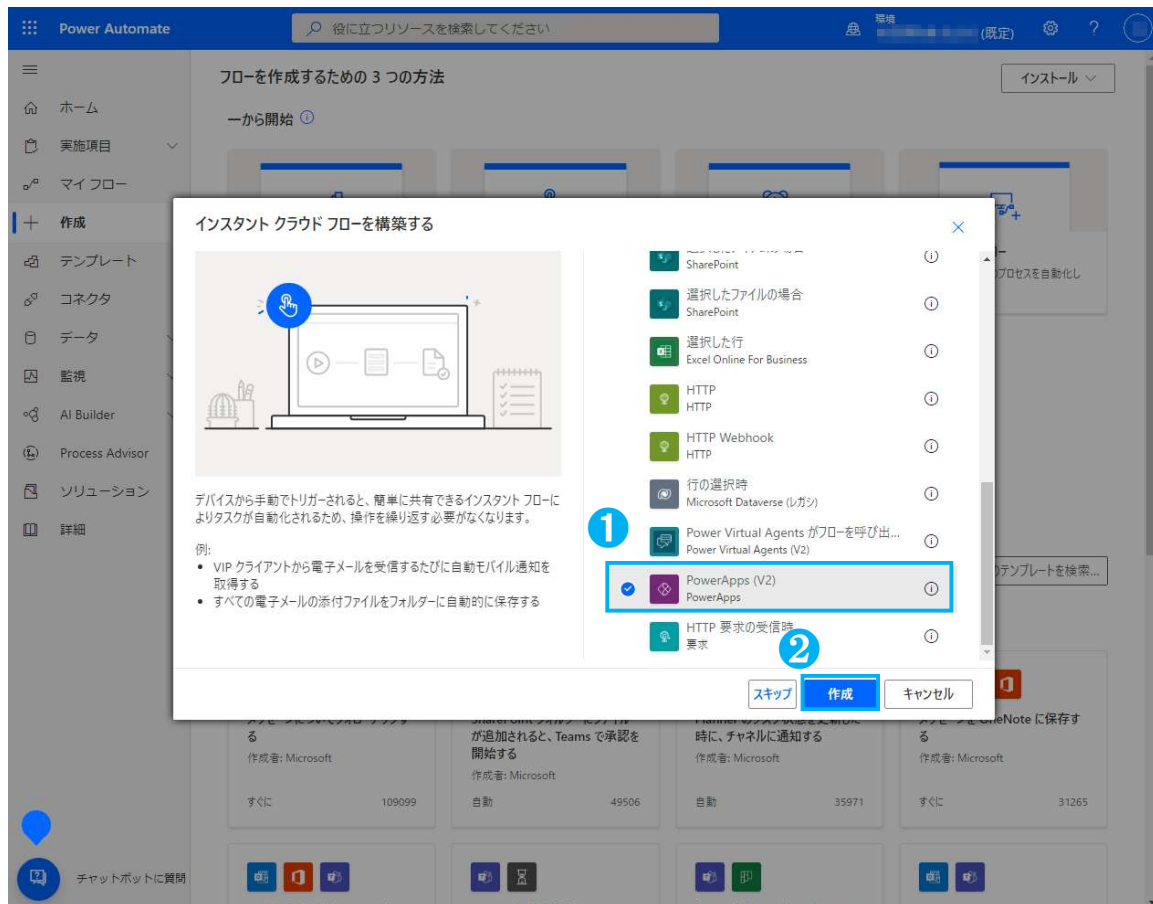
画面 13-52

フロー名とトリガーを指定します。フロー名は、[書類の自動作成] と入力します（画面 13-53）。



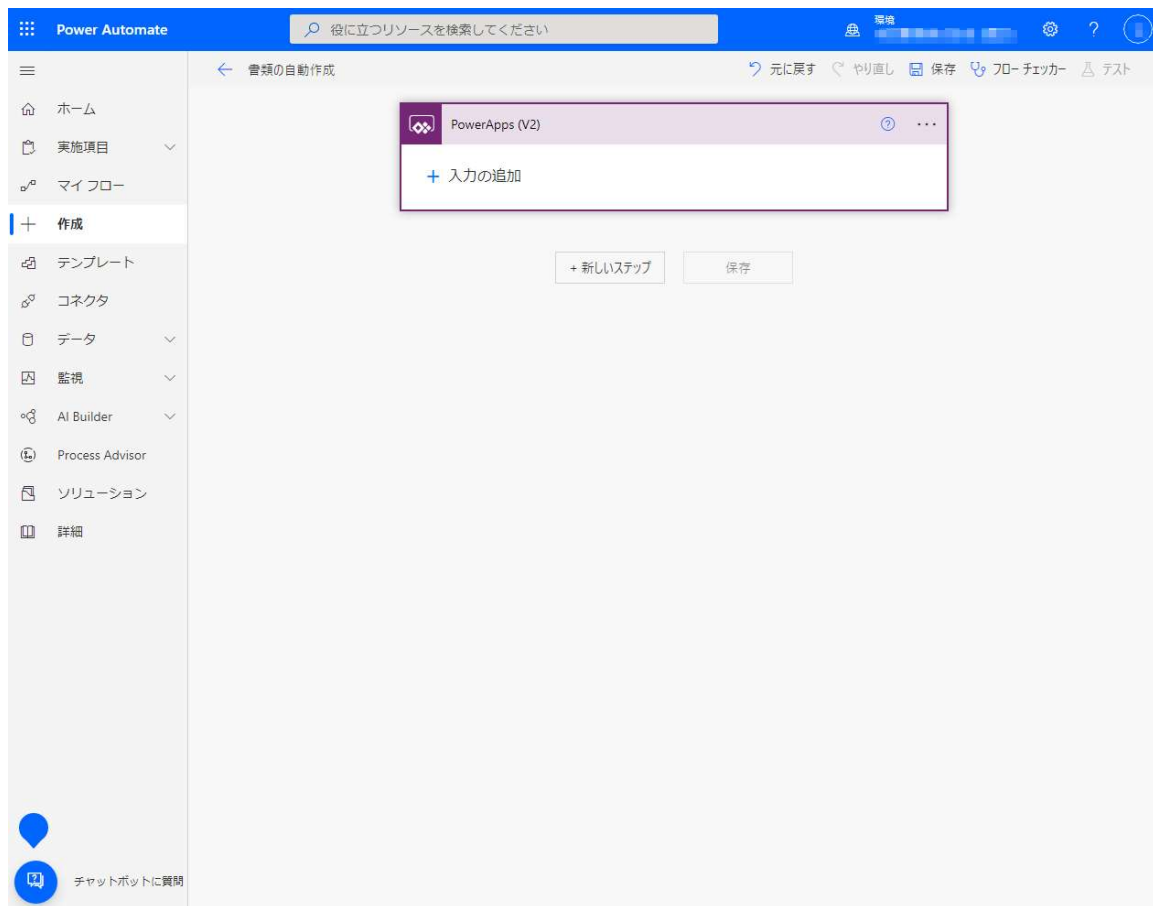
画面 13-53

トリガーは、[Power Apps (V2)] ⇒ [作成] を選択します (画面 13-54)。

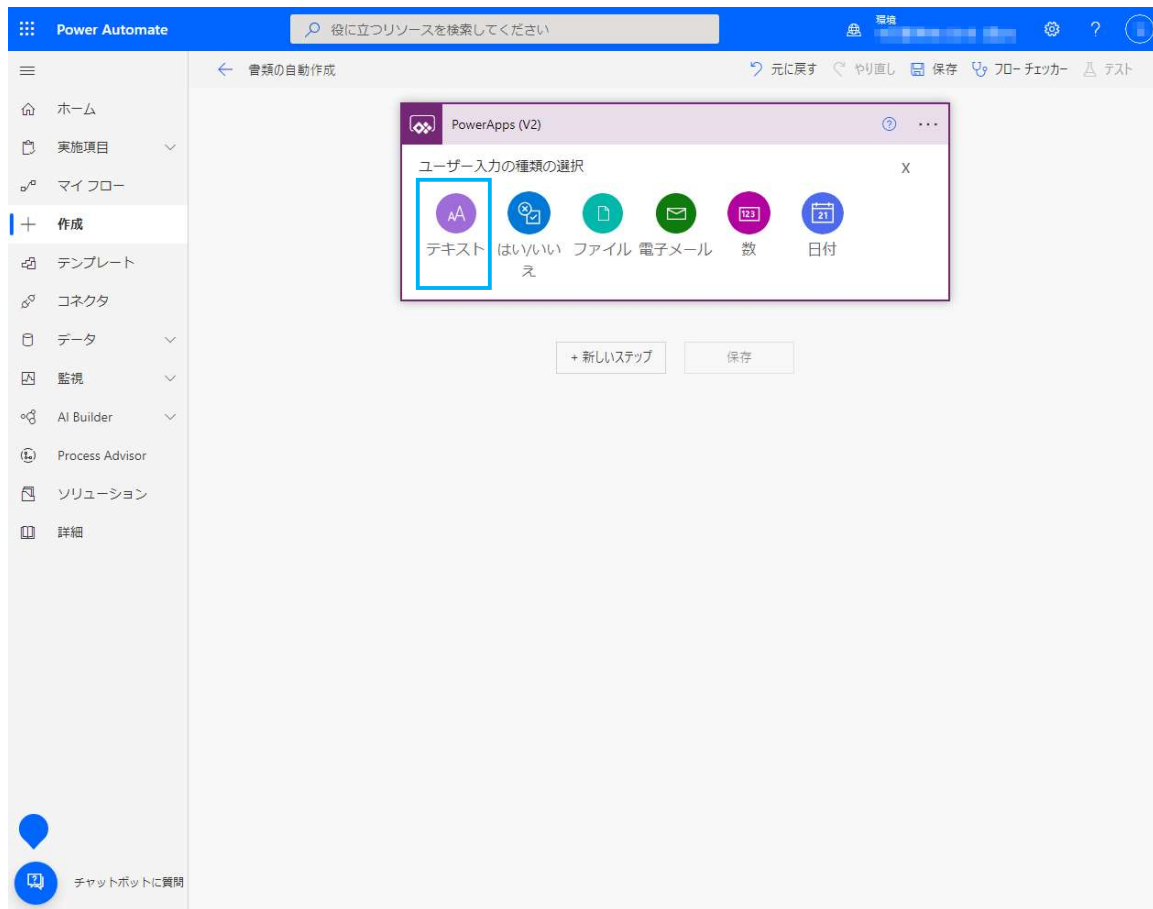


画面 13-54

Power Automate 側で Power Apps の値を受け取る設定をします。[+入力
の追加] ⇒ユーザー入力の種類の選択 [テキスト] を選択します (画面 13-
55、13-56)。

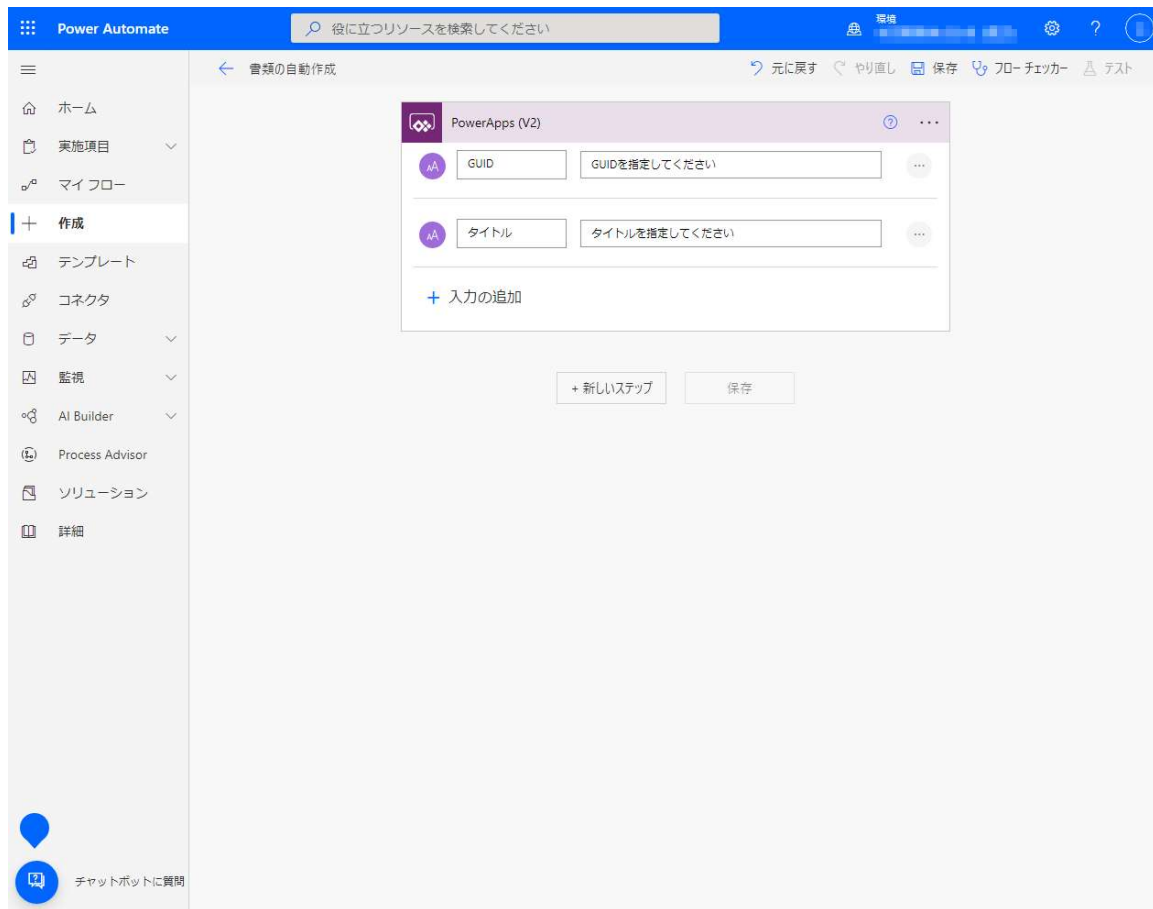


画面 13-55



画面 13-56

左側に受け取る値の変数名・右側に説明文を入力します（画面 13-57）。

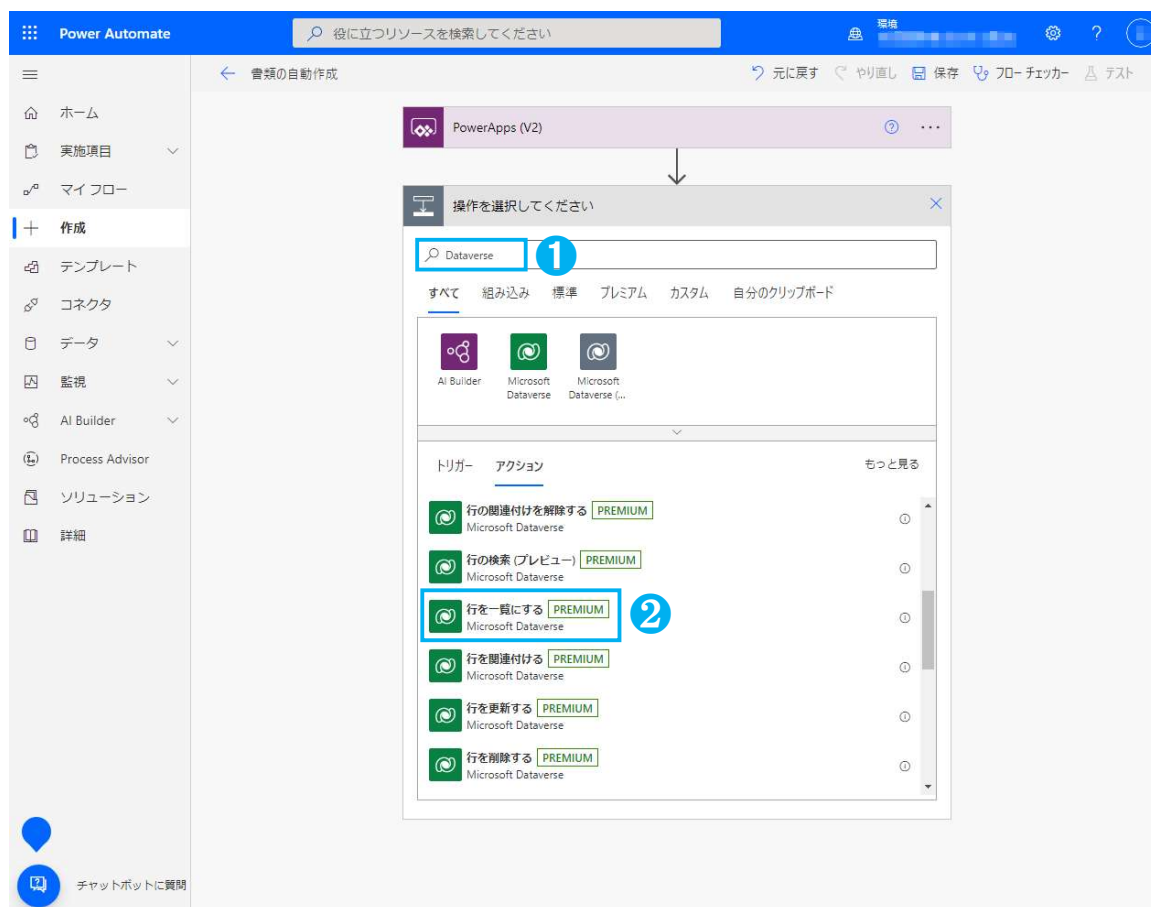


画面 13-57

トリガーの次はアクションを設定します。

まず、見積書テーブルと見積書詳細テーブルのデータを取得します。

「+新しいステップ」をクリック後、検索バーに「Dataverse」と入力します。表示されたアクション一覧から Microsoft Dataverse の「行を一覧にする」を選択します（画面 13-58）。



画面 13-58

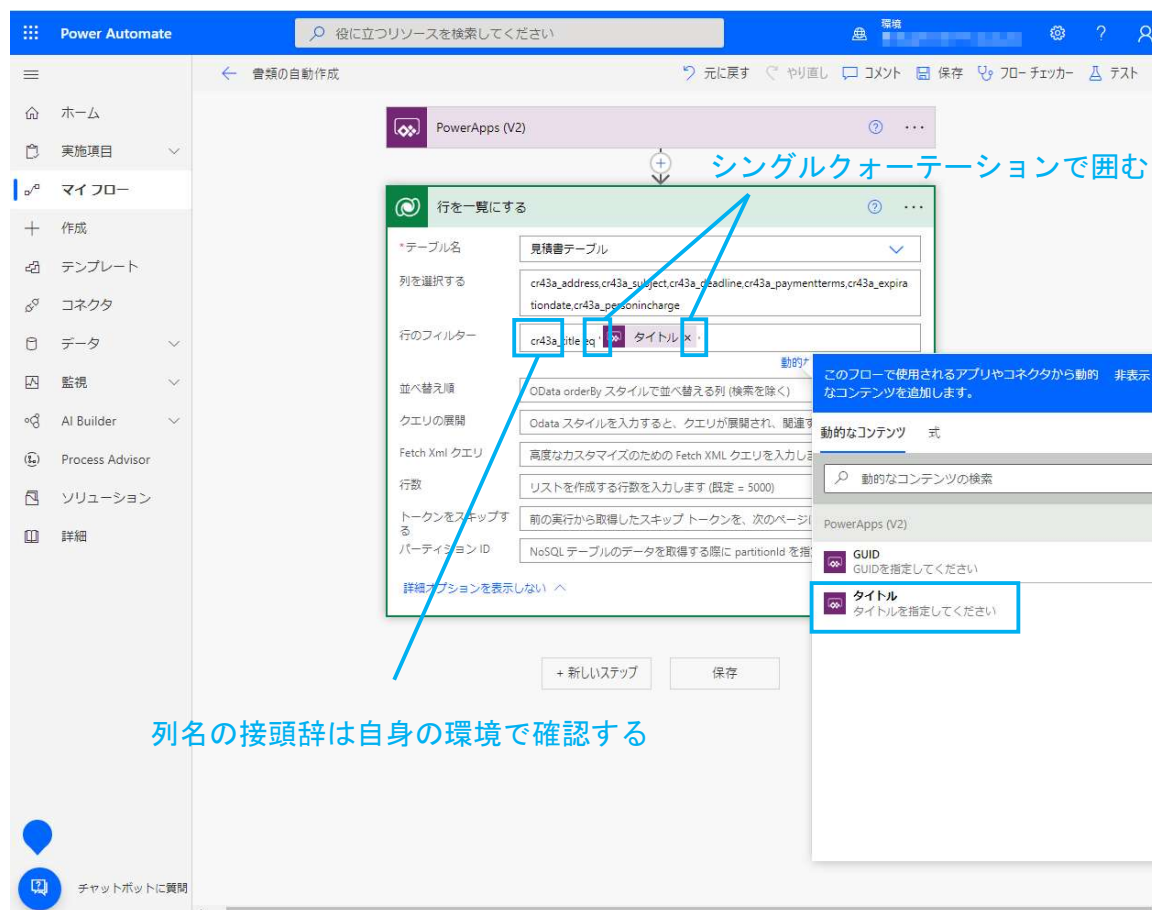
テーブル名は、[見積書テーブル] を選択します。

「列を選択する」は、取得するテーブルの列名を全て半角小文字で入力し、カンマで列を区切ります。

取得する列は、[宛名(address)] [件名(subject)] [納期(deadline)] [支払条件(paymentterms)] [有効期限(expirationdate)] [担当者(personincharge)] です。

「列を選択する」の記述方法については、[関数の入力補助] フォルダに格納されている<Chapter13-4_パラメータシート.xlsx>の<接頭辞の確認・列を選択する>をご参照ください。

行のフィルターは、取得する条件式を入力します。条件式の左辺には列名を入力します。条件式の右辺には「動的なコンテンツ」画面から「Power Apps (V2)」⇒「タイトル」を選択して挿入します。「タイトル」をシングルクォーテーションで囲むのを忘れないでください（画面 13-59）。



画面 13-59

※今回使用している環境では列名の接頭辞が「cr43a」となっていますが、こちらの接頭辞は環境によって異なります。ご自身の環境に表示されている接頭辞を記述してください。

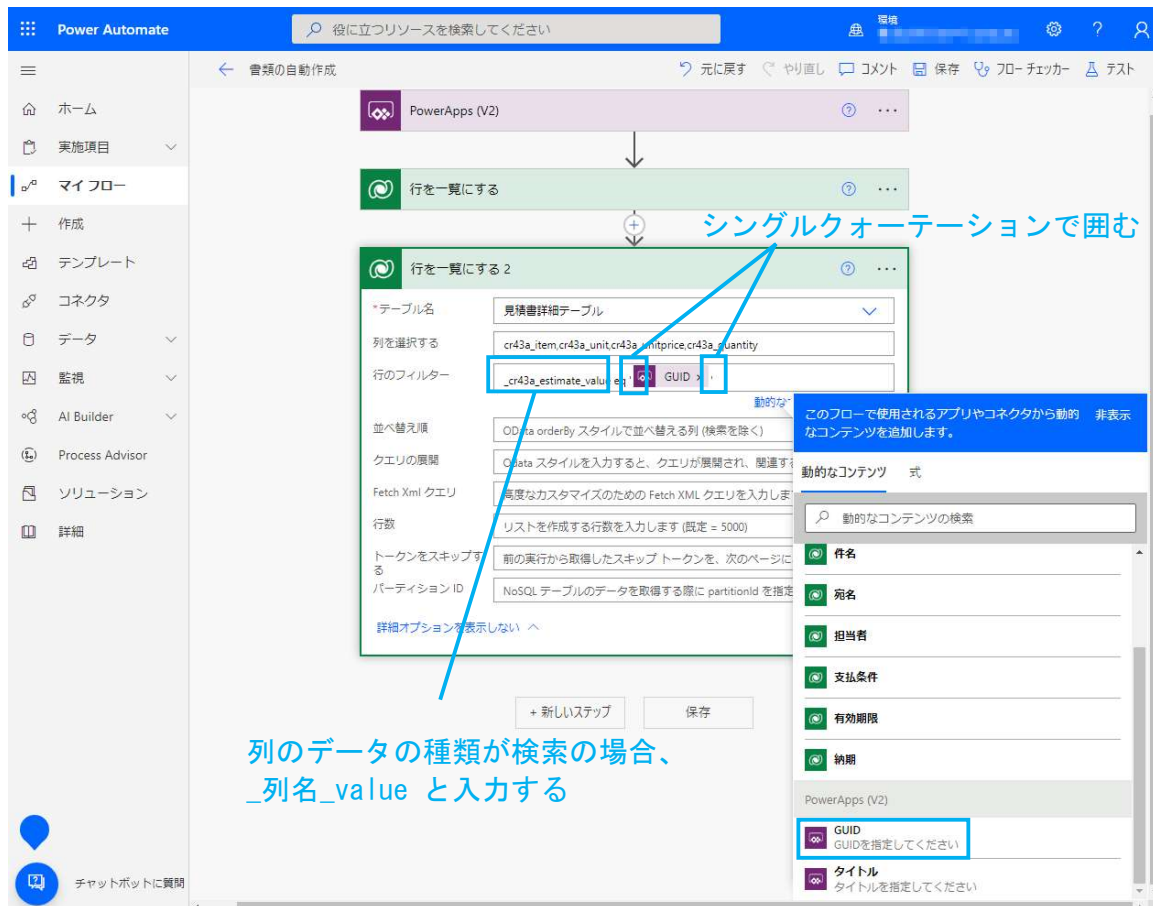
接頭辞の確認方法と行のフィルターの記述方法については、「関数の入力

補助] フォルダに格納されている<Chapter13-4_パラメータシート.xlsx>の<接頭辞の確認・行のフィルター>をご参照ください。

同様の手順で、見積書詳細テーブルのデータを取得します。

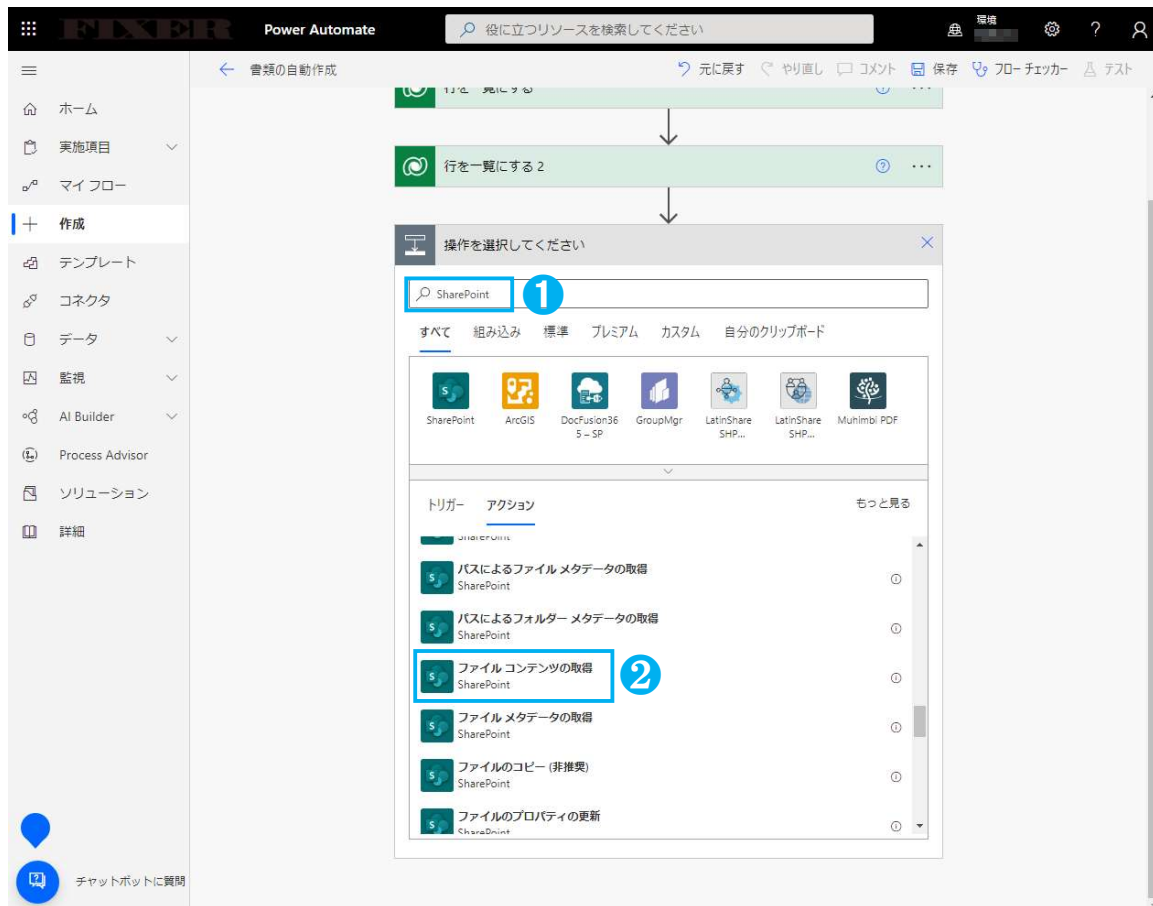
取得する列は、[品目(item)][単位(unit)][単価(unitprice)][数量(quantity)]です。

行のフィルターで列名のデータの種類の種類が[検索]の場合、列名の先頭に[_ (半角アンダーバー)]・末尾に[_ (半角アンダーバー)value]と入力します。条件式の右辺には[動的なコンテンツ]画面から[Power Apps(V2)]⇒[GUID]を選択して挿入します。[GUID]をシングルクォーテーションで囲むのを忘れないでください(画面 13-60)。



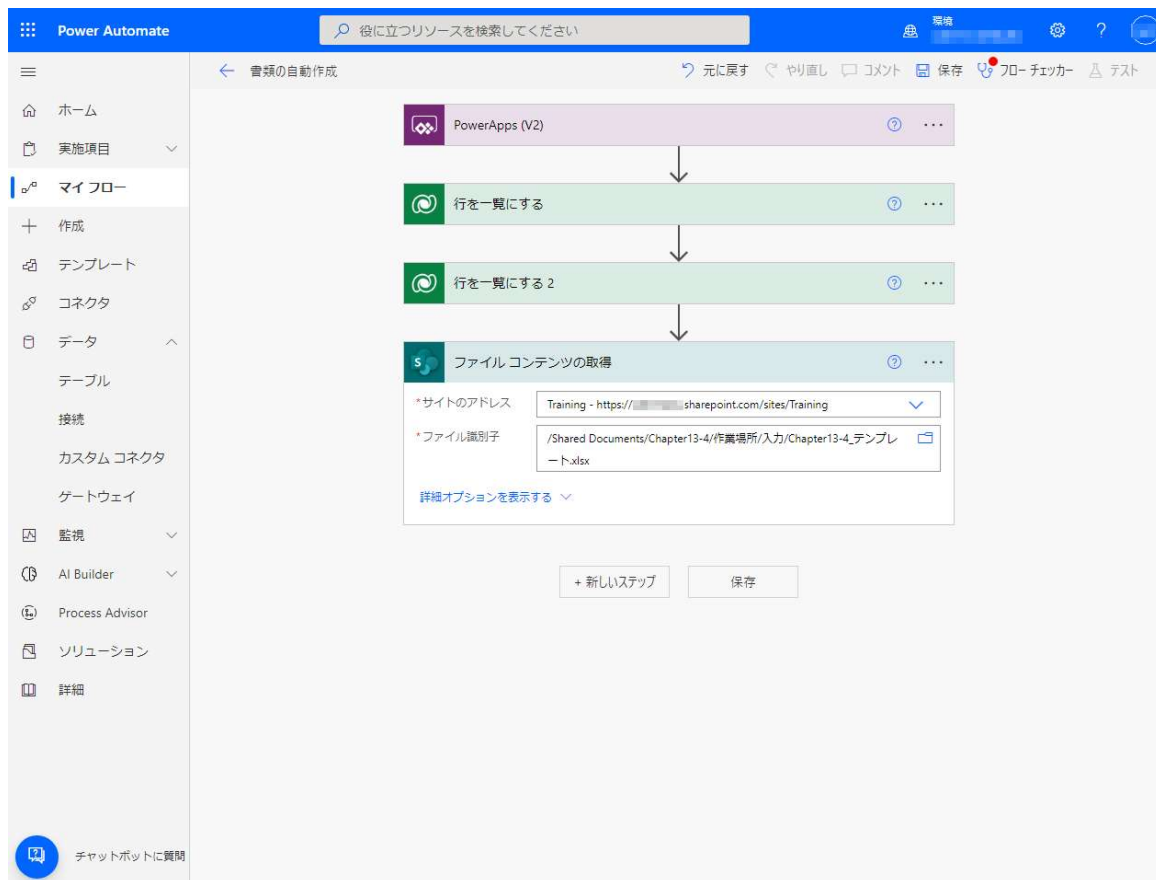
画面 13-60

次は、検索バーに [SharePoint] と入力します。表示されたアクション一覧から SharePoint の [ファイル コンテンツの取得] を選択します (画面 13-61)。



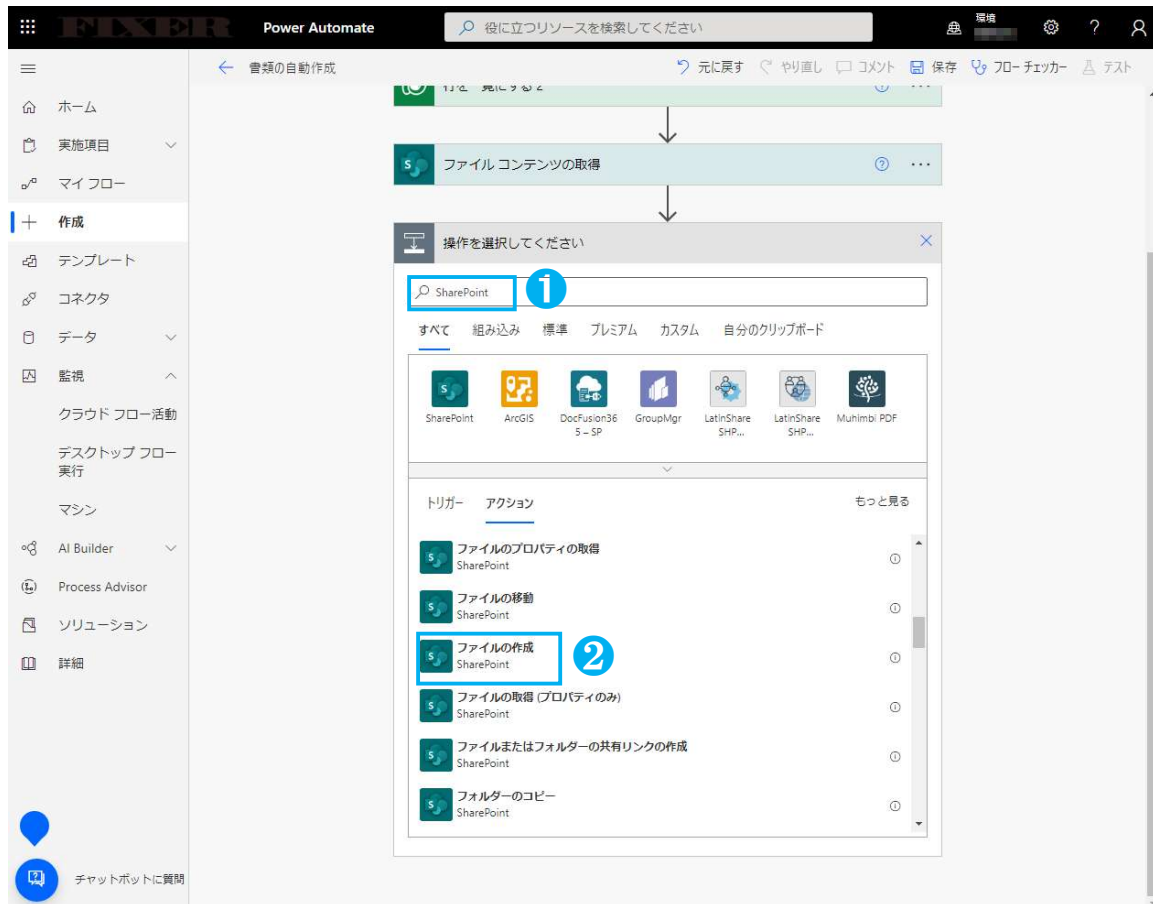
画面 13-61

サイトのアドレスは、表示される URL 一覧から [Training] を選択します。ファイル識別子は、[] を選択し、[Shared Documents] ⇒ [Chapter13-4] ⇒ [作業場所] ⇒ [入力] ⇒ [Chapter13-4_テンプレート.xlsx] を順番に選択します。こちらの手順でテンプレートファイルのデータを取得できます (画面 13-62)。



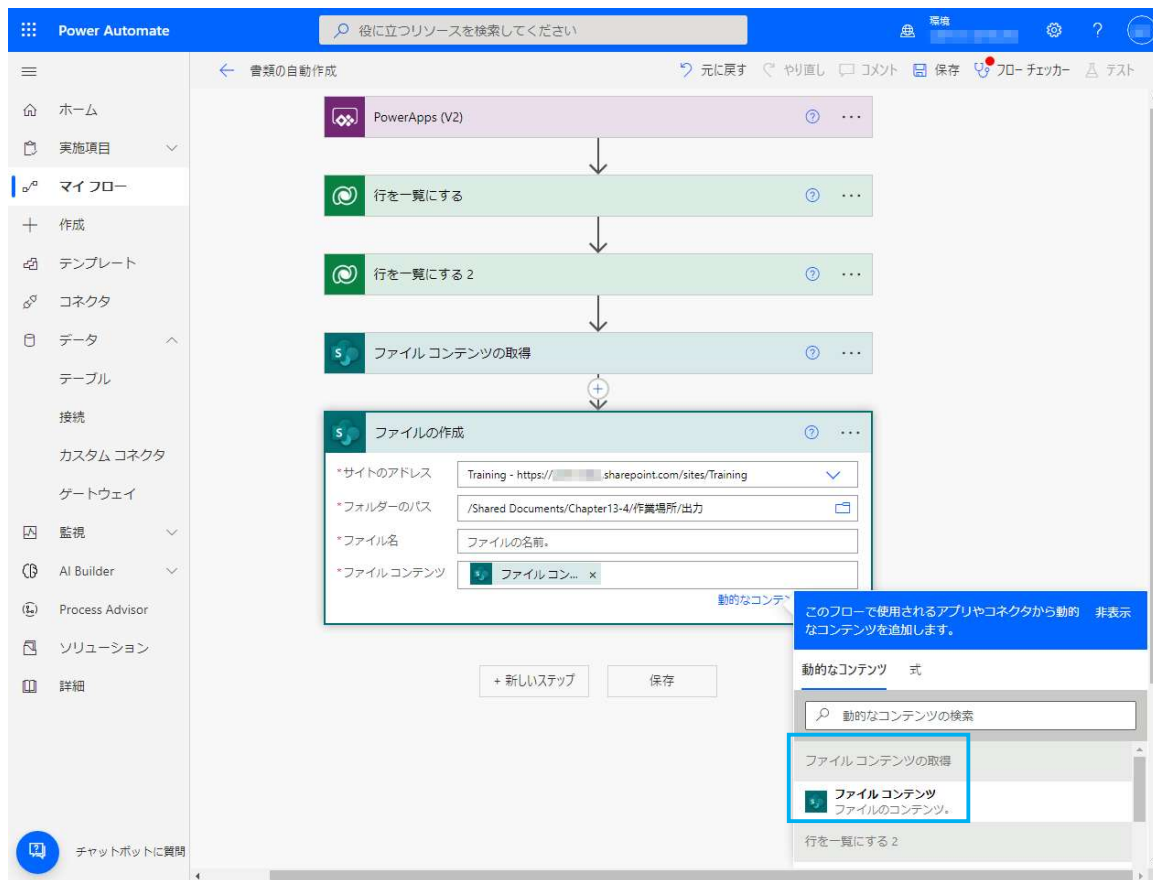
画面 13-62

検索バーに [SharePoint] と入力します。表示されたアクション一覧から SharePoint の [ファイルの作成] を選択します (画面 13-63)。



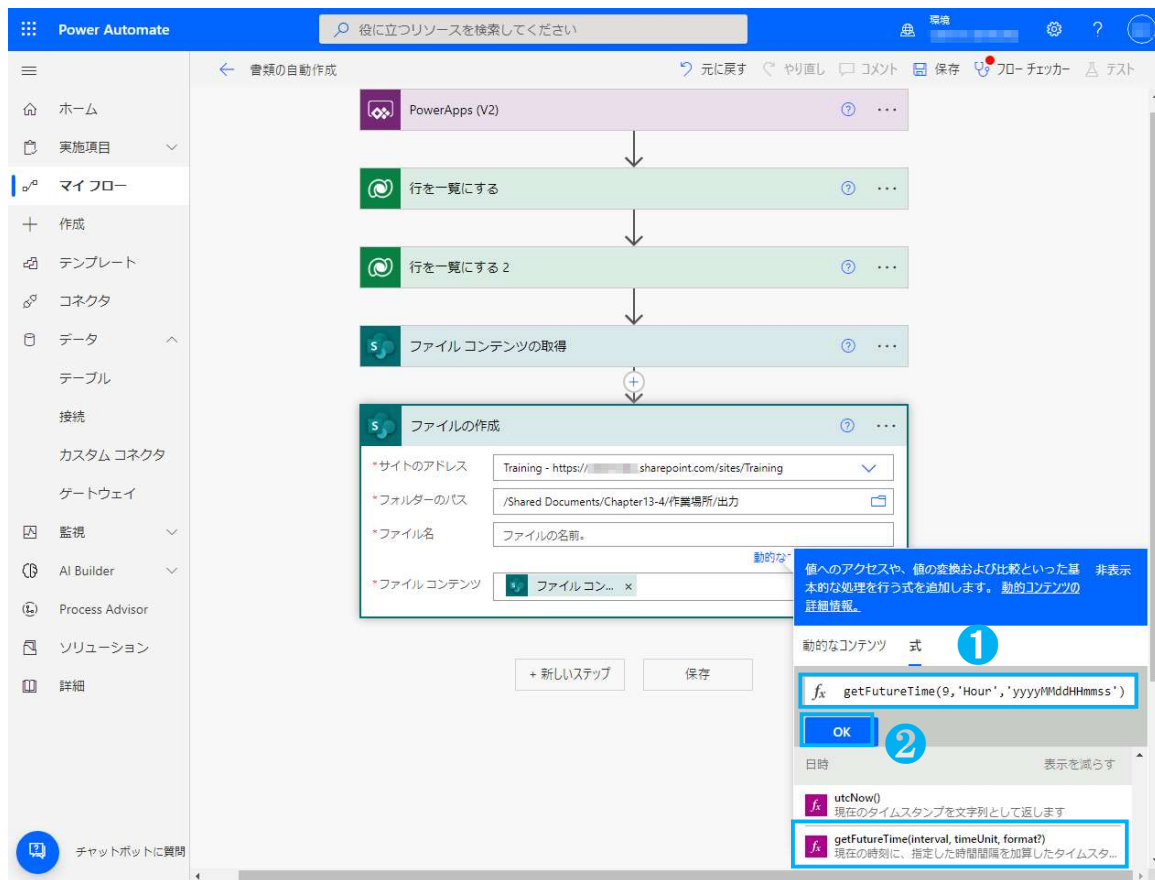
画面 13-63

サイトのアドレスは、表示される URL 一覧から [Training] を選択します。フォルダーのパスは、[] を選択し、[Shared Documents]⇒[Chapter13-4] ⇒ [作業場所] ⇒ [出力] を順番に選択します。ファイルコンテンツは、[動的なコンテンツ] 画面から [ファイルコンテンツの取得] ⇒ [ファイルコンテンツ] を選択して挿入します（画面 13-64）。



画面 13-64

ファイル名は、[動的なコンテンツ] 画面の式タブに移動し、[日時] ⇒ [getFutureTime] を選択します。getFutureTime の第 1 引数 [9]・第 2 引数 [‘Hour’]・第 3 引数 [‘yyyyMMddHHmmss’] と入力して、[OK] ボタンを選択します (画面 13-65)。

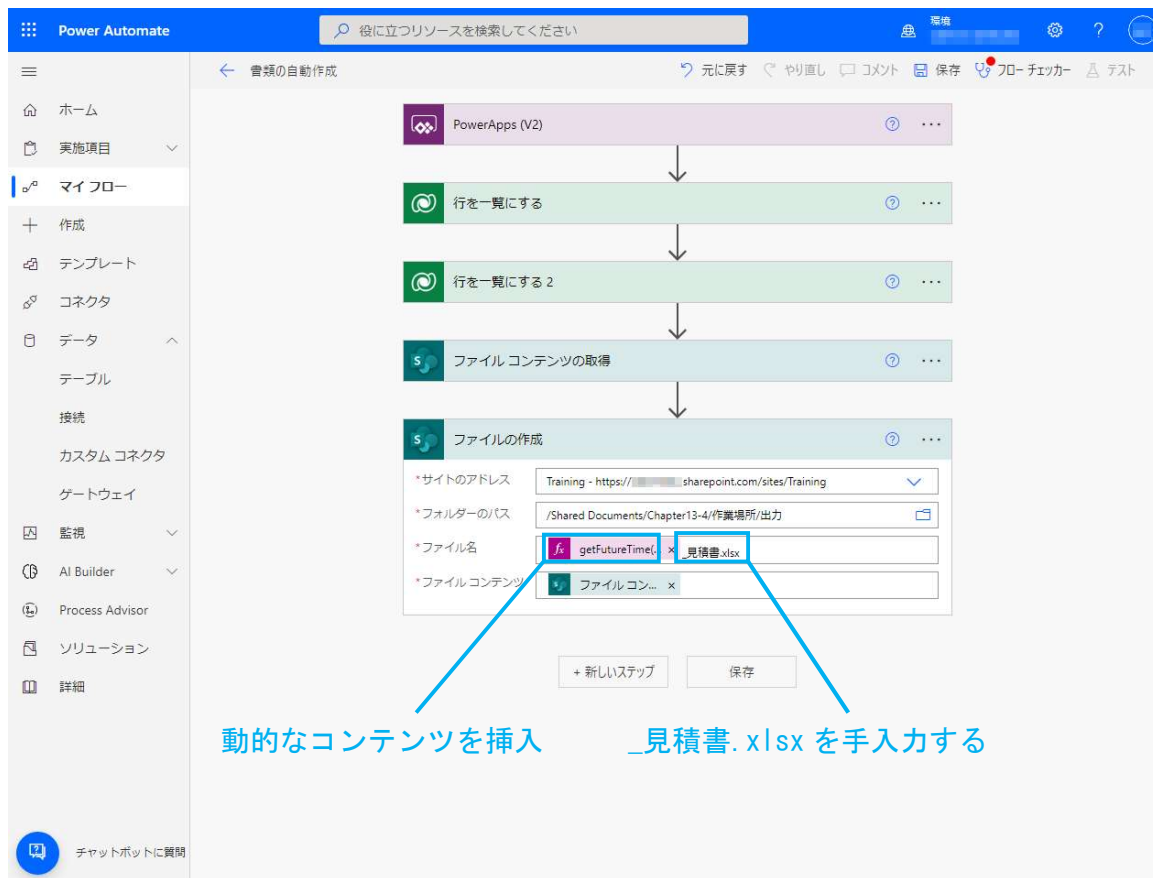


画面 13-65

また、ファイル名を<日付_見積書.xlsx>の形式にするので、末尾に「_見積書.xlsx」を入力します（画面 13-66）。

こちらの手順で「入力」フォルダに格納されている<Chapter13-4_テンプレート.xlsx>を「出力」フォルダに<日付_見積書.xlsx>というファイル名でコピーできます。

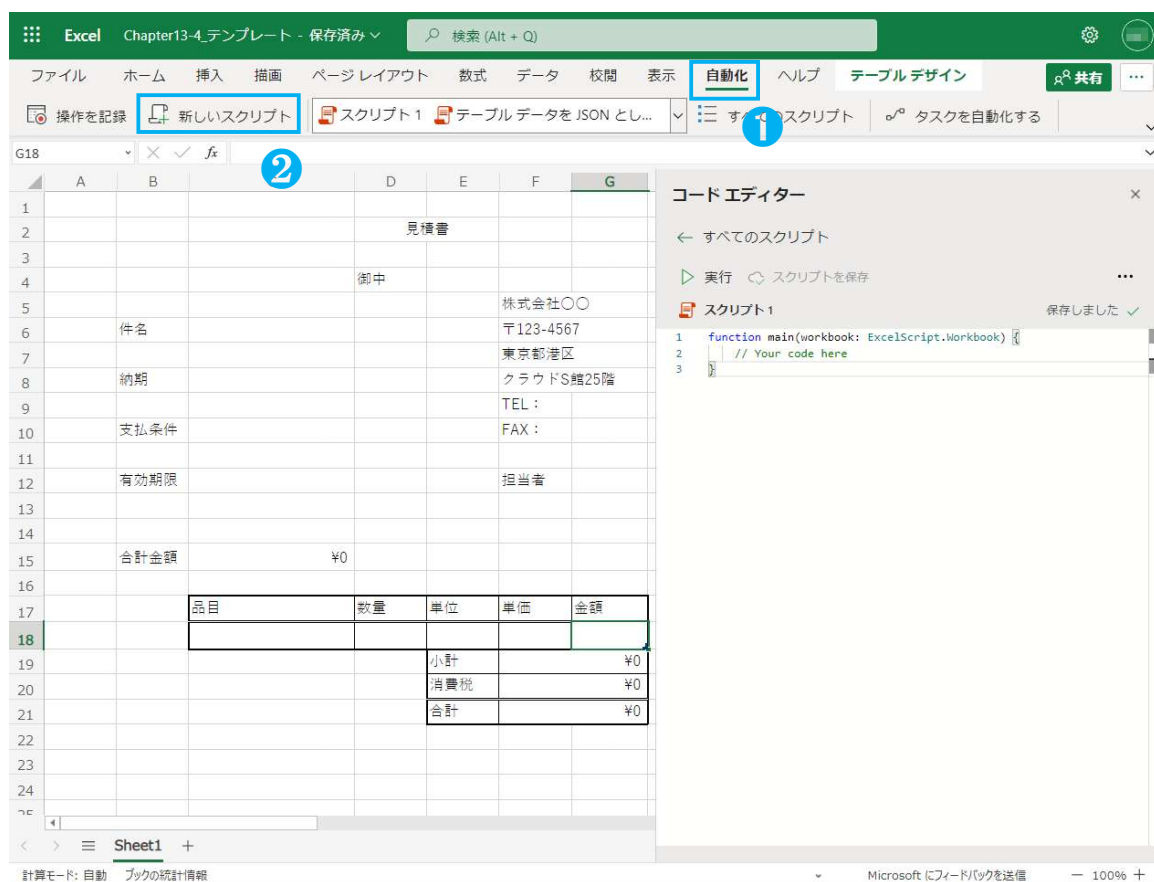
こちらのアクション設定をする理由は、見積テンプレートを複製して何度も使いまわせるようにするためです。見積テンプレートに直接書き込む処理をせずに済みます。



画面 13-66

次は、Office スクリプトで先ほどコピーした<日付_見積書.xlsx>の特定のセルにデータを挿入するアクションを作成します。

SharePoint サイト [Training] の「入力」フォルダに格納された<Chapter13-4_テンプレート.xlsx>を開き、[自動化] タブ⇒[新しいスクリプト] を選択します (画面 13-67)。



画面 13-67

〔関数の入力補助〕フォルダに格納されている<Chapter13-4_パラメータシート.xlsx>の<Office スクリプトコード>を参照して、コードエディターにスクリプトコードを記述します。また、スクリプトの名前を〔見積書スクリプト〕と分かりやすい名称に変更します（画面 13-68）。

各行のコードの意味は以下のとおりです。

- 1 行目：Power Automate から受け取ったデータを変数に格納する。
- 2 行目：現在、起動中のワークシートを `selectedSheet` という変数に格納する。
- 3-8 行目：`selectedSheet(起動中のワークシート).getRange(セルの場`

所).setValue(変数名)

フローを実行すると、以下の動作になります。

Power Automate から受け取った「ABC 銀行」というデータを変数名 [address] に格納する。現在起動中の Sheet1 のセル [C4] に「ABC 銀行」を挿入する。

The screenshot shows the Microsoft Excel interface with a form and a script editor. The form contains the following data:

件名	株式会社〇〇
納期	〒123-4567
支払条件	東京都港区
有効期限	クラウドS館25階
合計金額	TEL :
	FAX :
	担当者
	¥0

The script editor shows the following code:

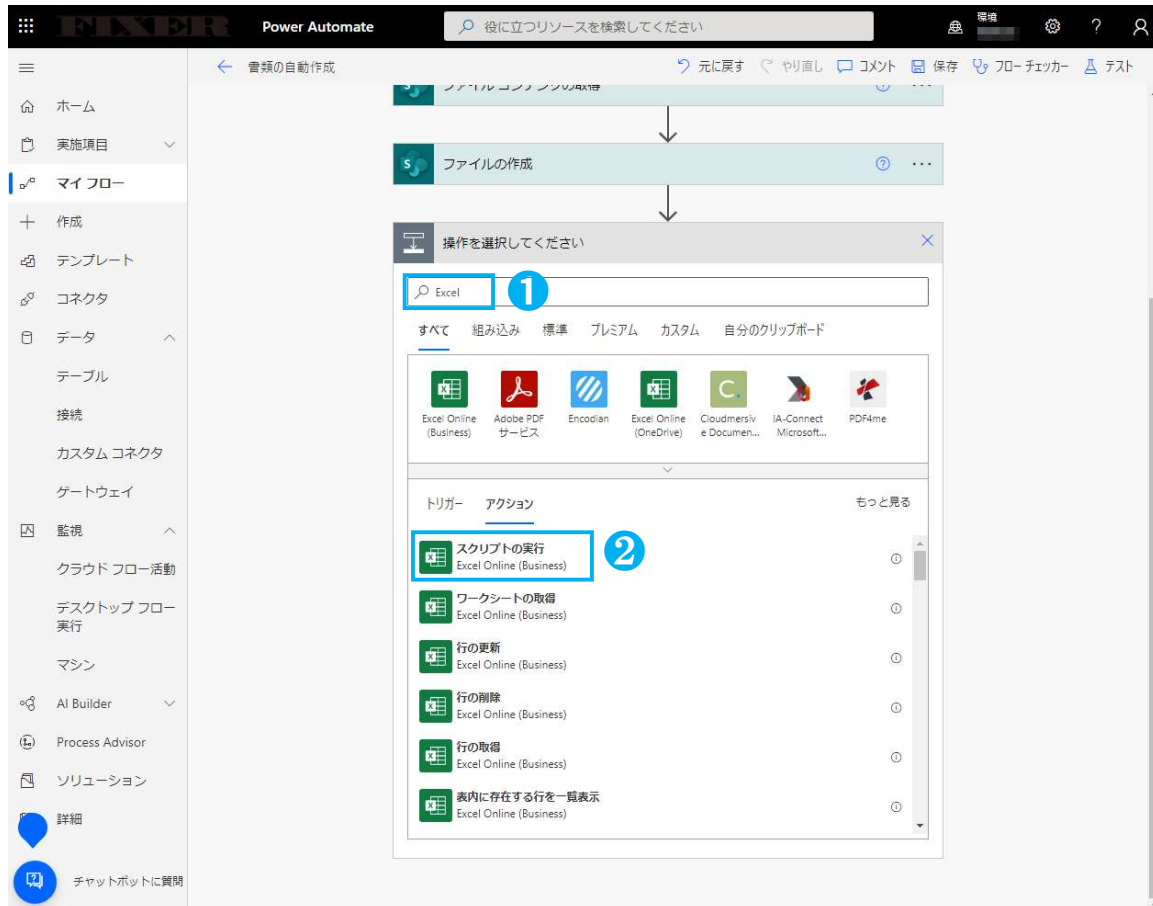
```

function main(workbook: ExcelScript.Workbook, address: string,
  subject: string, deadline: string, paymentTerms: string,
  expirationDate: string, personInCharge: string) {
  let selectedSheet = workbook.getActiveWorksheet();
  selectedSheet.getRange("C4").setValue(address);
  selectedSheet.getRange("C6").setValue(subject);
  selectedSheet.getRange("C8").setValue(deadline);
  selectedSheet.getRange("C10").setValue(paymentTerms);
  selectedSheet.getRange("C12").setValue(expirationDate);
  selectedSheet.getRange("G12").setValue(personInCharge);
}

```

画面 13-68

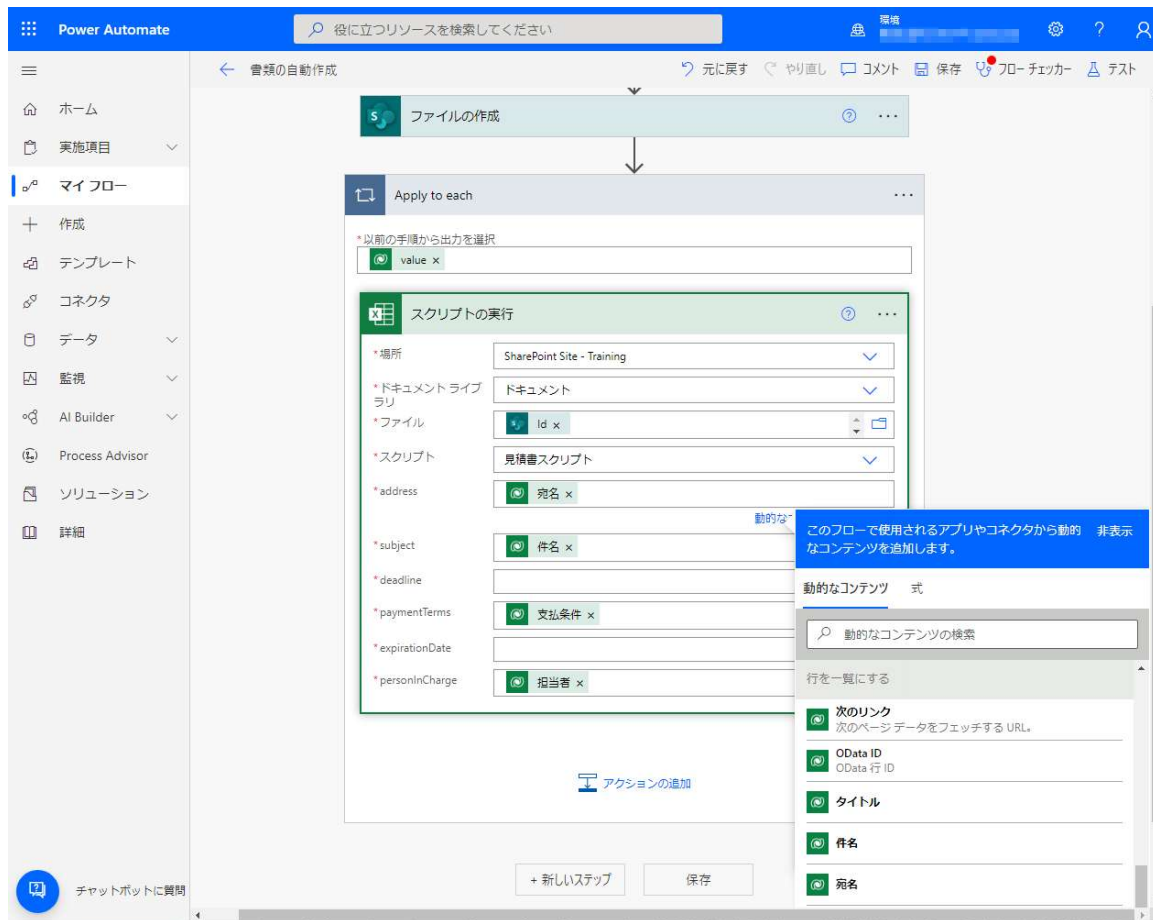
検索バーに [Excel] と入力します。表示されたアクション一覧から Excel Online (Business) の [スクリプトの実行] を選択します (画面 13-69)。



画面 13-69

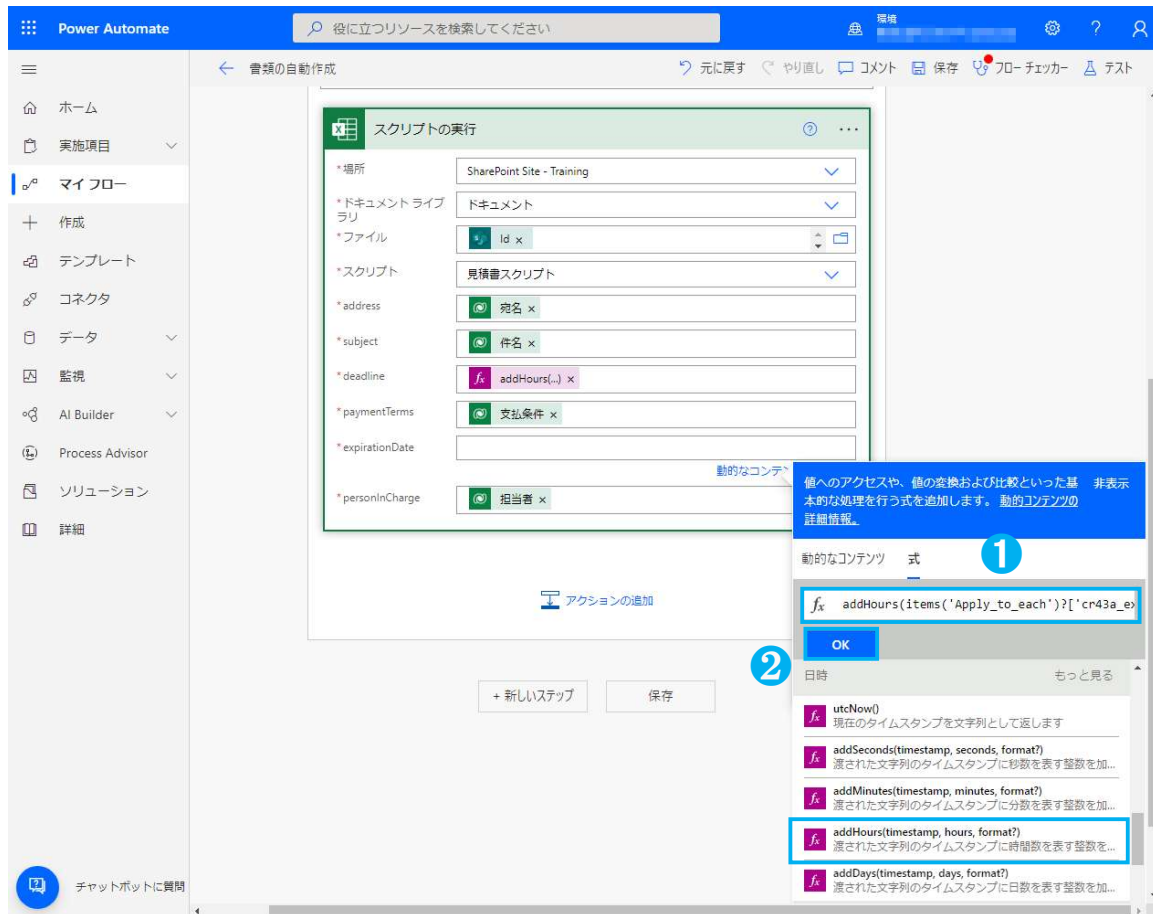
場所は、表示される一覧から [SharePoint Site - Training] を選択します。ドキュメントライブラリは、[ドキュメント] を選択します。ファイルは、[動的なコンテンツ] 画面から [ファイルの作成] ⇒ [Id(ファイルまたはフォルダーの一意の ID)] を選択して挿入します。スクリプトは、表示される一覧から [見積書スクリプト] を選択します。

Office スクリプトで設定されている変数名が一覧に表示されます。各変数名に該当する値を [動的なコンテンツ] 画面から選択します (画面 13-70)。



画面 13-70

[deadline] [expirationDate] の項目には、9 時間足して日本時間に変換する処理が必要です。[動的なコンテンツ] 画面の式タブに移動し、[日時] ⇒ [addHours] を選択します。addHours の第 1 引数 [items('Apply_to_each')? ['列名']]・第 2 引数 [9]・第 3 引数 ['yyyy 年 M 月 d 日'] と入力します。[OK] ボタンを選択します (画面 13-71)。

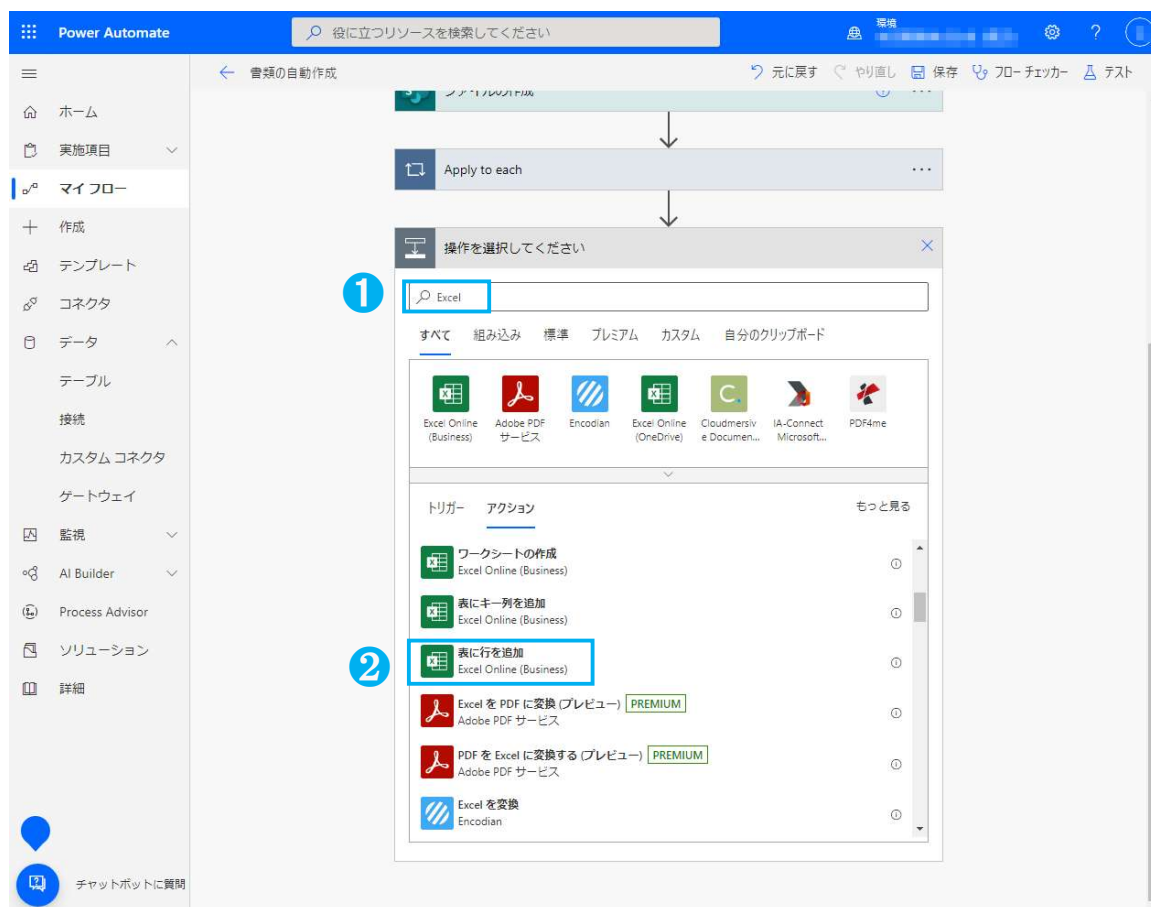


画面 13-71

addHours 関数はタイムスタンプに時間数を加算する関数です。第 1 引数にタイムスタンプ、第 2 引数に加算する単位の数、第 3 引数に書式フォーマット “yyyy-MM-ddTHH:mm:ss” を指定します。第 3 引数は省略可能です。

次は、見積書テンプレートの [見積書] テーブルにデータを挿入します。

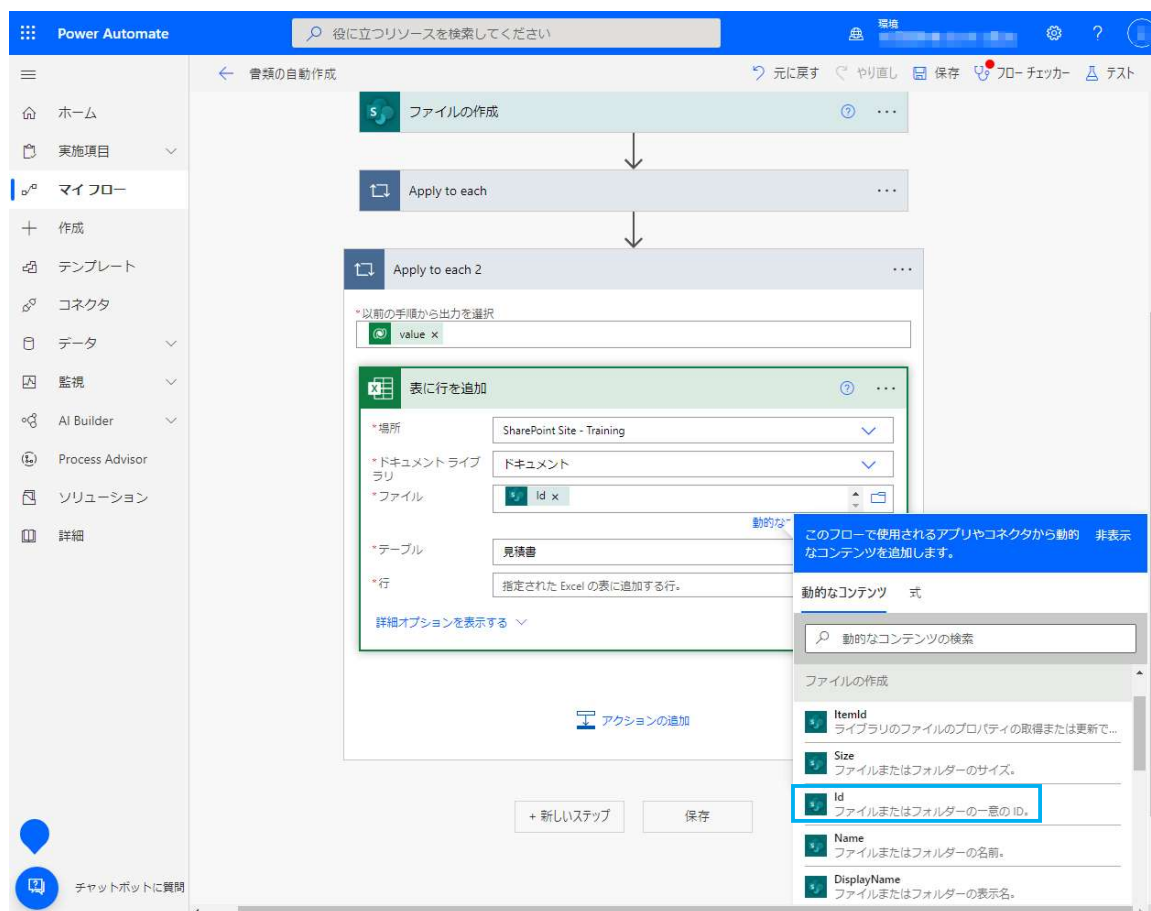
検索バーに [Excel] と入力します。表示されたアクション一覧から Excel Online (Business) の [表に行を追加] を選択します (画面 13-72)。



画面 13-72

場所は、表示される一覧から [SharePoint Site - Training] を選択します。ドキュメントライブラリは、[ドキュメント] を選択します。

ファイルは、[動的なコンテンツ] 画面から [ファイルの作成] ⇒ [Id(ファイルまたはフォルダーの一意の ID)] を選択して挿入します。テーブルは、[カスタム値の入力] を選択し、[見積書] と入力します (画面 13-73)。

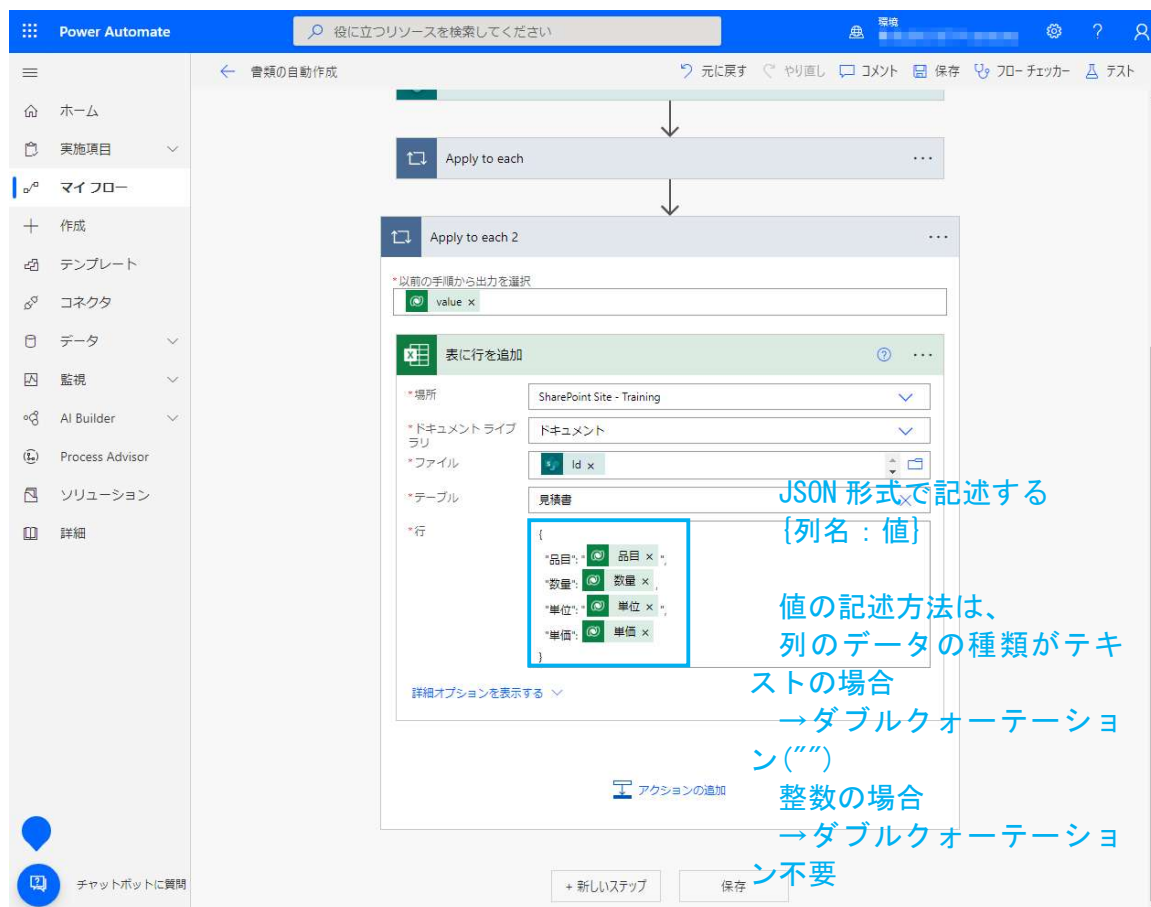


画面 13-73

行は、JSON 形式で入力していきます。{} の中に列名と値をコロンで区切って入力します。列名はダブルクォーテーションで囲む必要があります。また、列名と値の組み合わせを複数入力する場合は、カンマで区切ります。JSON の記述方法については、[関数の入力補助] フォルダに格納されている〈Chapter13-4_パラメータシート.xlsx〉の〈表に行を追加する〉をご参照ください。

また、「JSON」についての詳細は本書 Column 「JSON と仲良くなろう！」で解説しています。JSON を深く学習したい方はそちらもご覧ください。

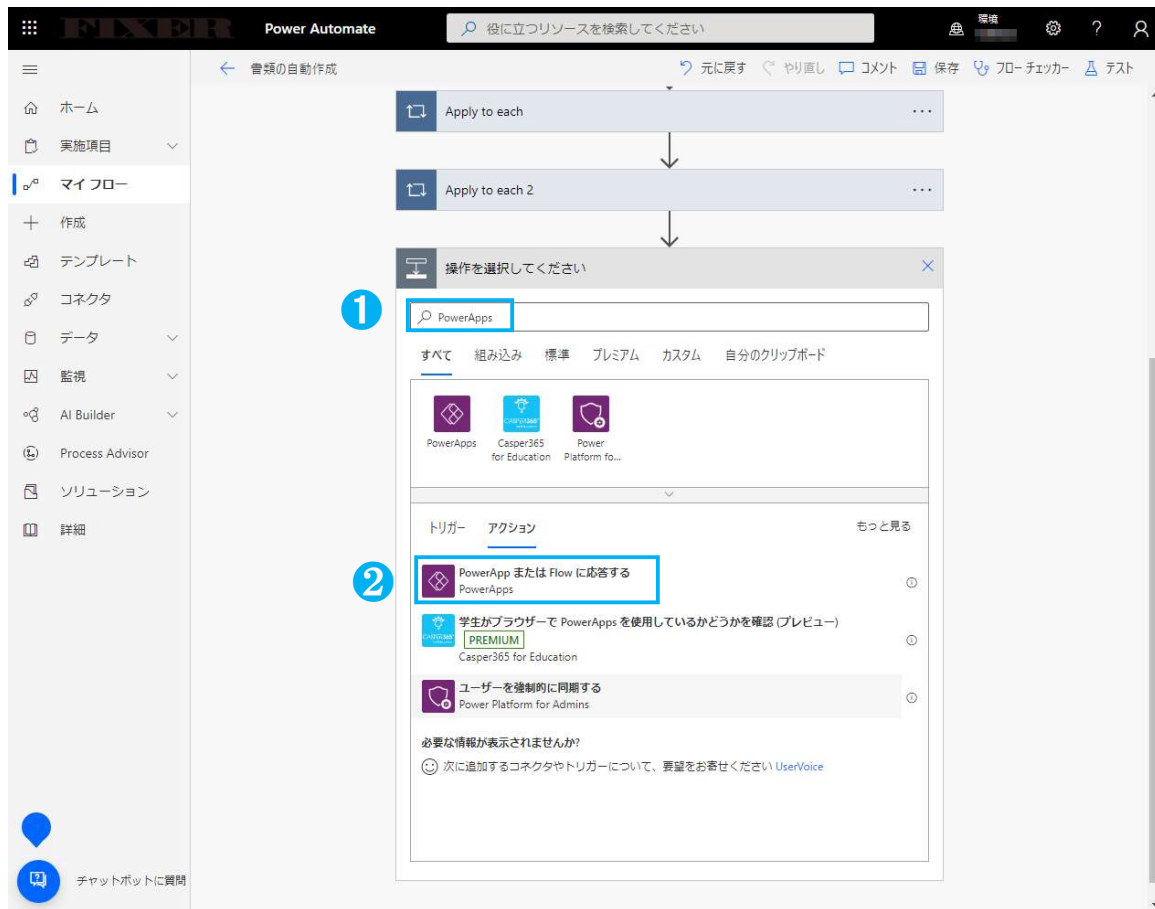
今回の場合、値は[動的なコンテンツ]画面から選択します(画面 13-74)。



画面 13-74

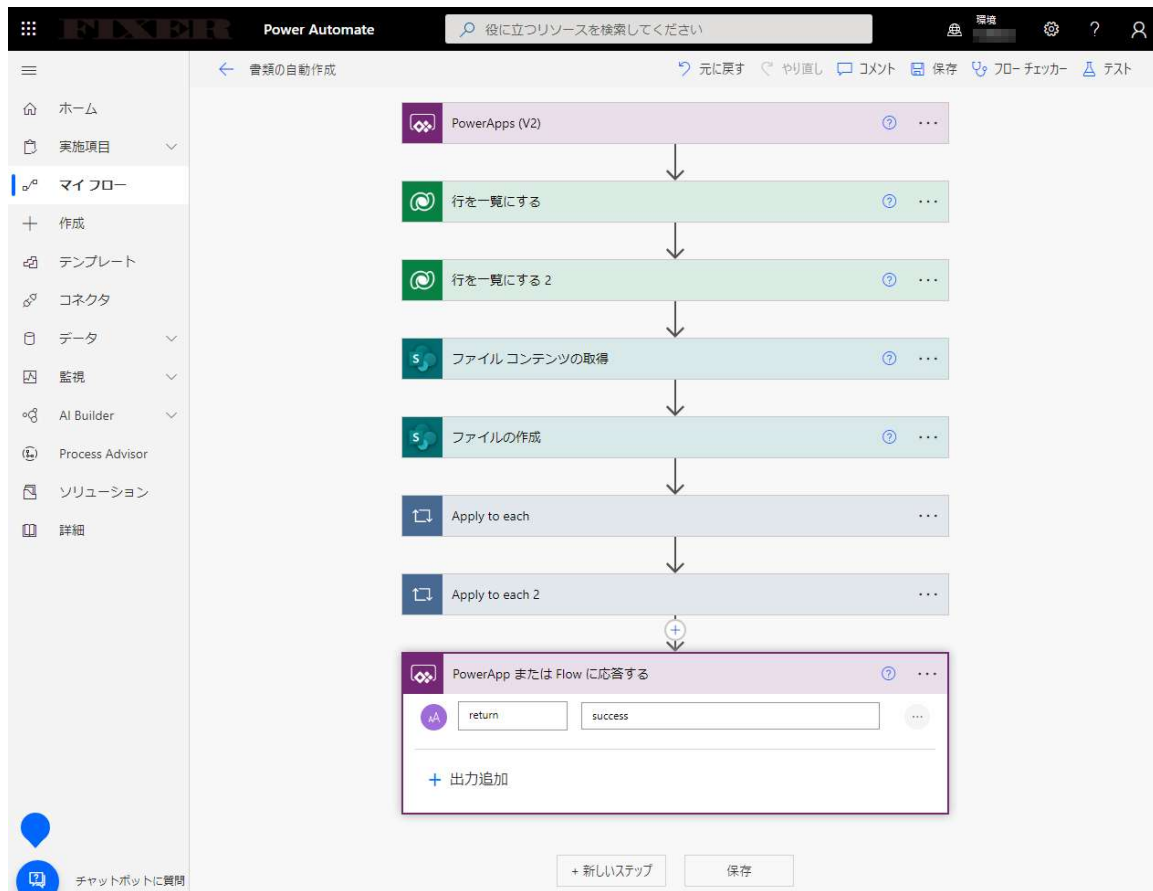
最後に、クラウドフローの Excel 出力処理の終了後、キャンバスアプリに値を渡します。

検索バーに [PowerApps] と入力します。表示されたアクション一覧から Power Apps の [PowerApp または Flow に応答する] を選択します (画面 13-75)。



画面 13-75

[+出力追加] ⇒出力の種類を選択 [テキスト] を選択し、左側に渡す変数名・右側に値を入力します。今回の場合、クラウドフローの Excel 出力処理の終了後、キャンバスアプリに [success] という値を渡します (画面 13-76)。

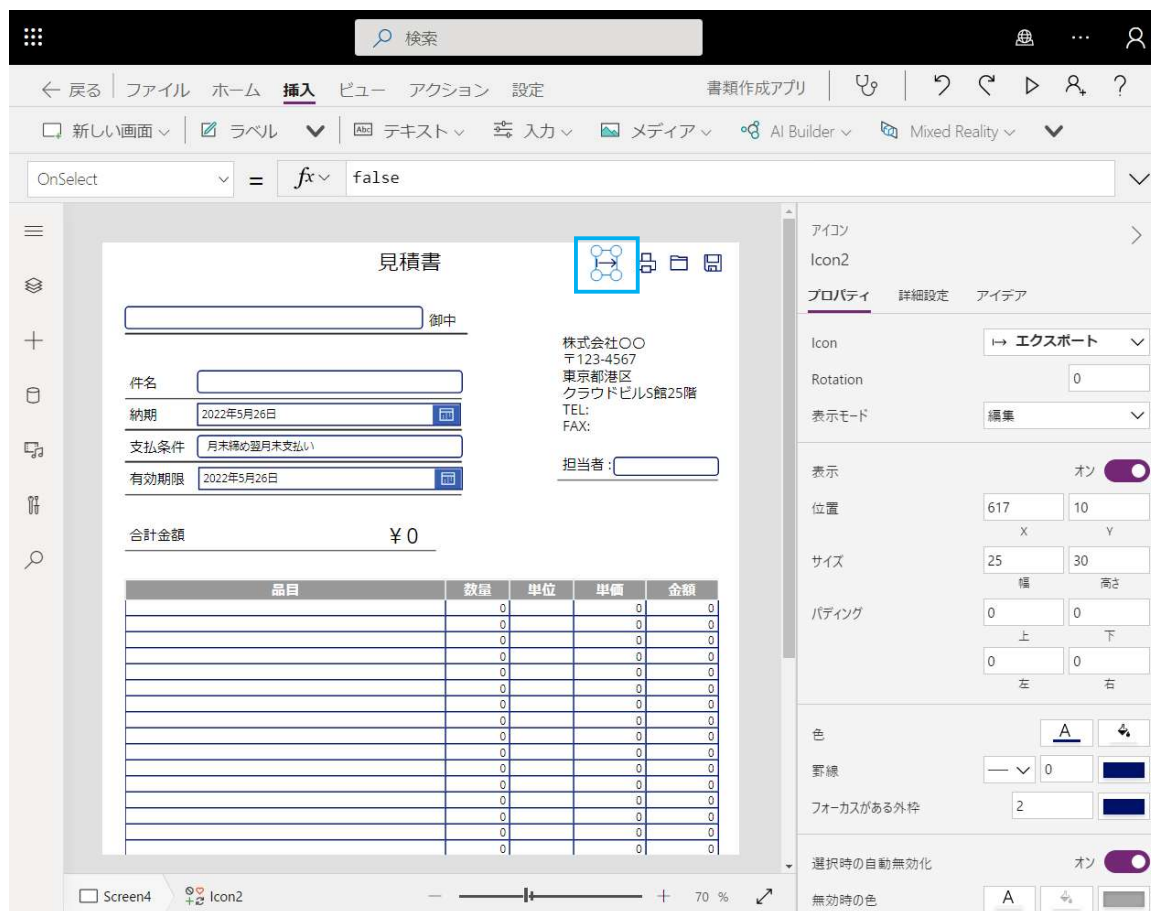


画面 13-76

以上で、アクションの設定は完成です。

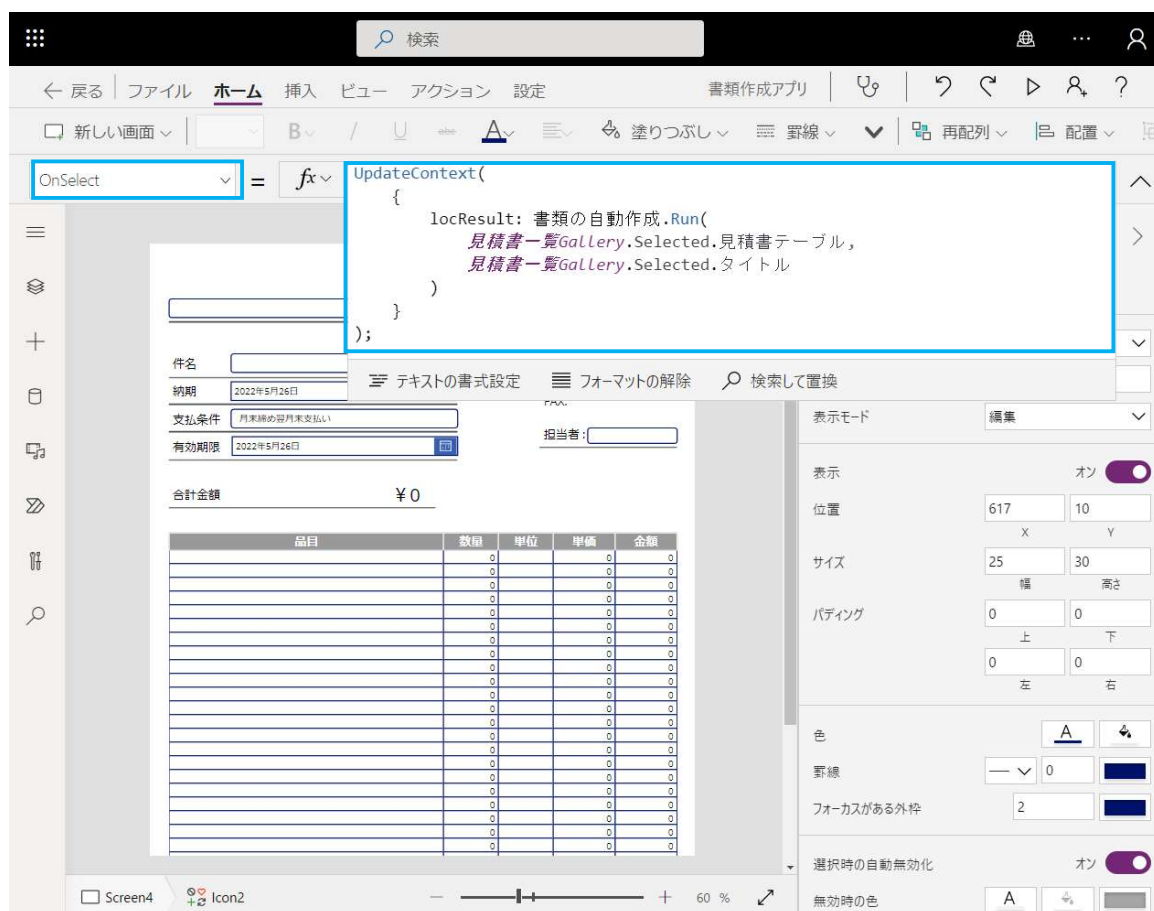
4. 見積書を作成するアプリと手順 3. のクラウドフローを接続する

Chapter13-2 で作成した見積書を作成するアプリを開き、アイコンを追加します（画面 13-77）。



画面 13-77

左側のナビゲーションエリアの「Power Automate アイコン」を選択します。「+フローの追加」から手順 3. で作成したクラウドフロー「書類の自動作成」を選択します（画面 13-78）。

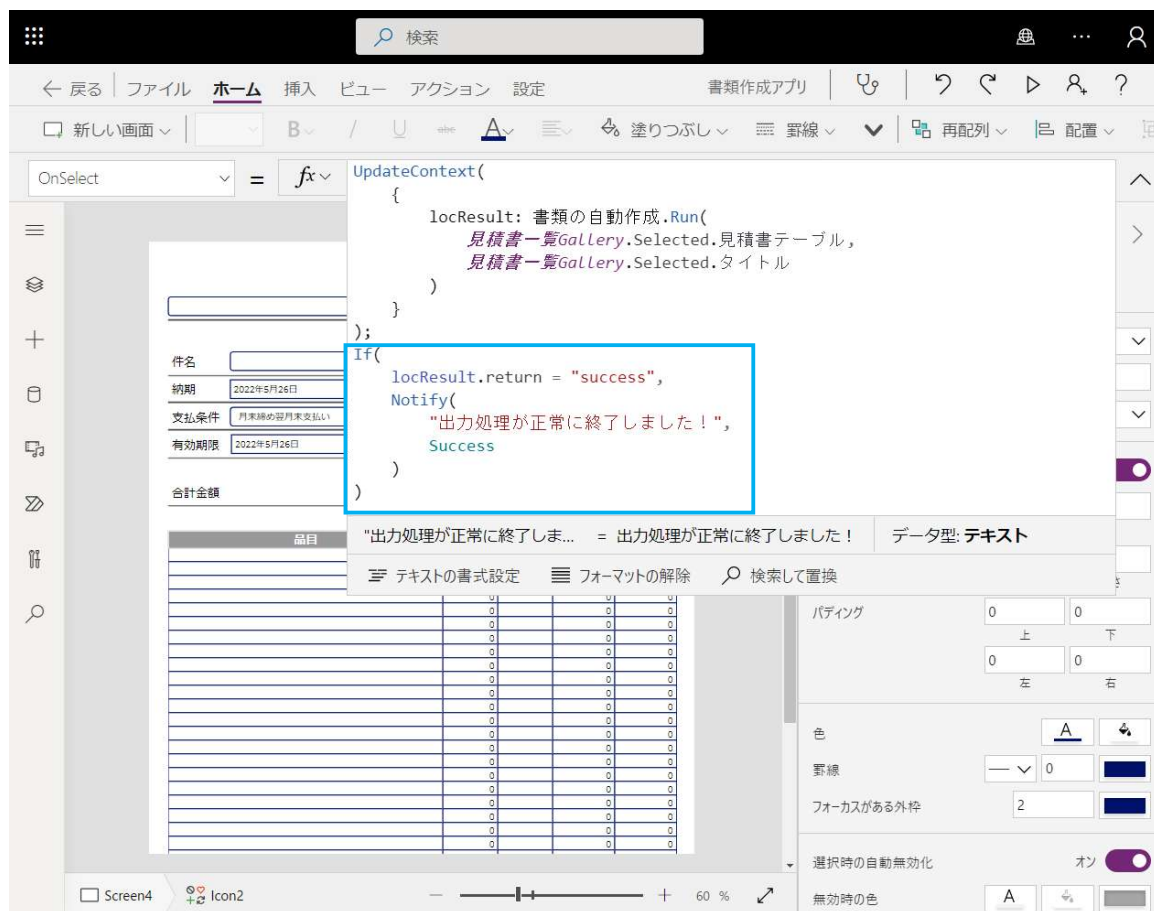


画面 13-79

Excel 書類の出力処理が完了したことを利用者に通知するために、Notify 関数でダイアログのメッセージを表示します。

キャンバスアプリの関数の入力については、[関数の入力補助] フォルダに格納されている<Chapter13-4_パラメータシート.xlsx>の<キャンバスアプリ>をご参照ください。

こちらの手順でキャンバスアプリとクラウドフローを接続することができます (画面 13-80)。



画面 13-80

Notify 関数は画面の上部にダイアログのメッセージを表示する関数です。

第 1 引数に表示するメッセージ、第 2 引数にメッセージの種類” Error, Information, Success, Warning”、第 3 引数に表示するミリ秒数を指定します。第 3 引数の規定値は 10 秒(10000 ミリ秒)です。

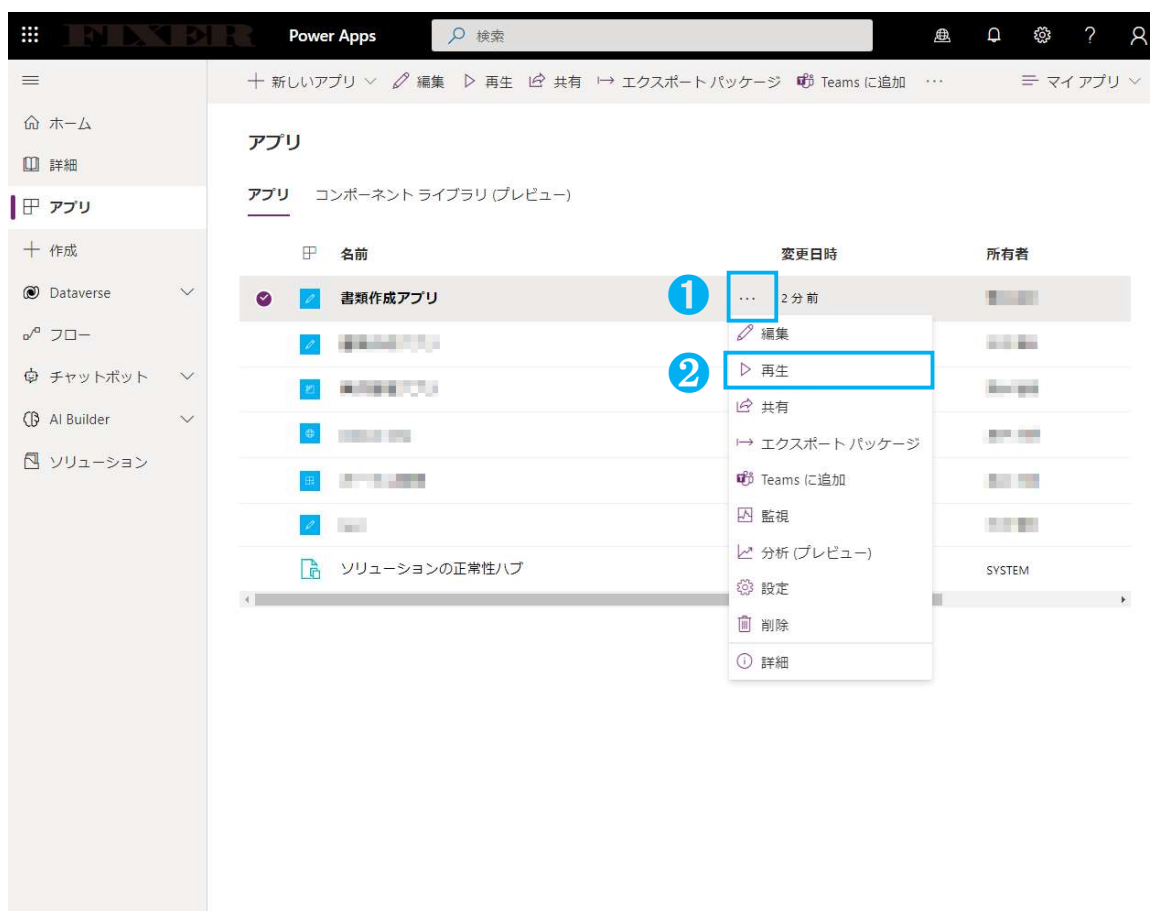
[Ctrl + S] または [ファイル] ⇒ [保存] で保存終了後、編集内容を反映したキャンバスアプリを実行するには [公開] を選択します。保存と公開の手順については、書籍本体の Chapter 3-4「キャンバスアプリの公開と共有」をご参照ください。

以上で、キャンバスアプリとクラウドフローの接続の設定は完了です。

5. 実行してテストする

キャンバスアプリの保存と公開が終了したら、実際に書類作成アプリを起動して、見積書を Excel に出力してみましょう。

Power Apps メーカーポータルにアクセスし、[⋮] ⇒ [再生] を選択します (画面 13-81)。



画面 13-81

出力したい書類を選択し、アイコンを選択します。2、3 分程経過すると、画面上部に「出力処理が正常に終了しました！」とダイアログのメッセー

ジが表示されます（画面 13-82）。

FIXER Power Apps | 書類作成アプリ

見積書

ABC銀行 御中

株式会社〇〇
〒123-4567
東京都港区
クラウドビル5階25階
TEL:
FAX:

件名: IPO上場支援

納期: 2022年4月1日

支払条件: 現金

有効期限: 2022年4月29日

担当者: テスト太郎

合計金額: ￥627000

品名	金額
AAA	10000
BBB	60000
CCC	500000
見積書A	0
見積書B	0
見積書C	0
見積書D	0
見積書E	0
見積書F	0
見積書G	0
見積書H	0
見積書I	0
見積書J	0
見積書K	0
見積書L	0
見積書M	0
見積書N	0
見積書O	0
見積書P	0
見積書Q	0
見積書R	0
見積書S	0
見積書T	0
見積書U	0
見積書V	0
見積書W	0
見積書X	0
見積書Y	0
見積書Z	0

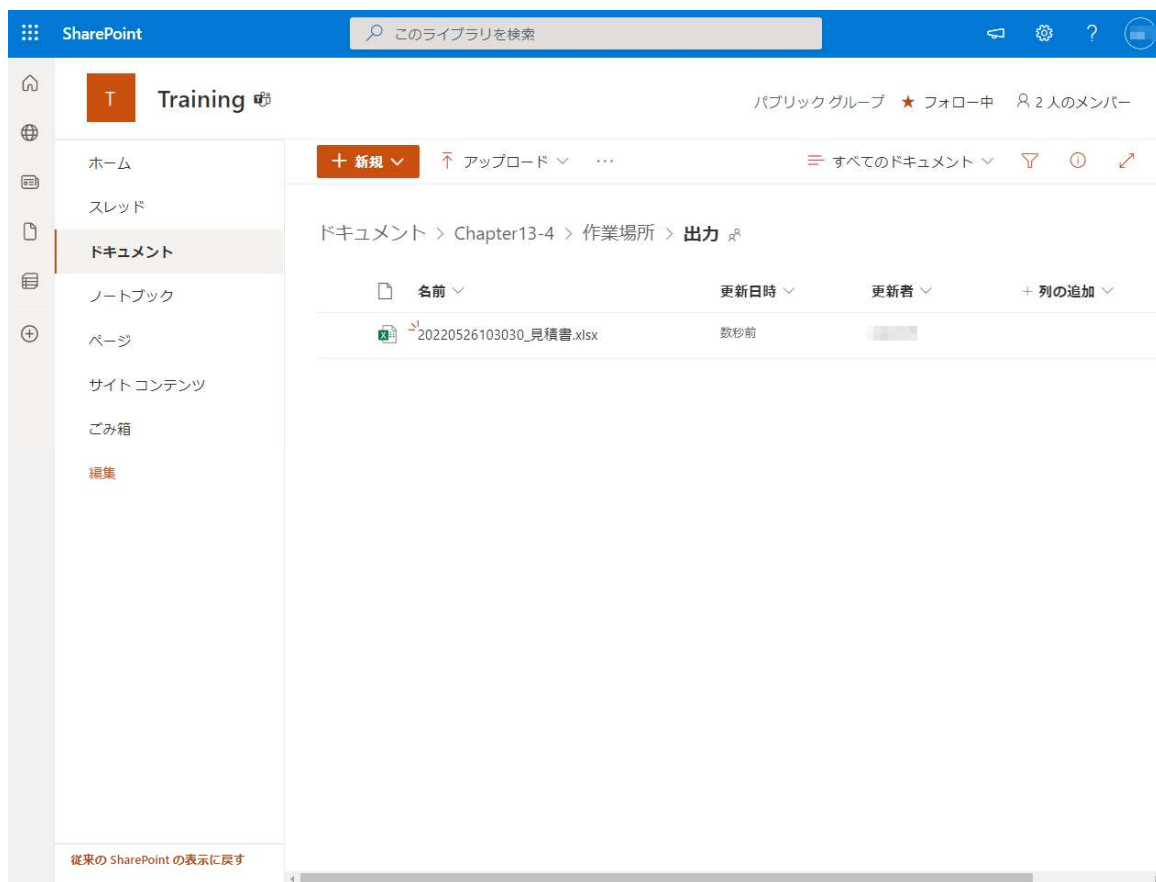
見積書を開く

見積書A
見積書B
見積書C
見積書D
見積書E
見積書F
見積書G

キャンセル 開く

画面 13-82

SharePoint サイト [Training] に移動し、[出力] フォルダの中身を確認すると、<日付_見積書.xlsx>の形式で見積書が Excel に出力されています（画面 13-83、13-84）。



画面 13-83

Excel 20220526103030_見積書 - 保存済み

検索 (Alt + Q)

ファイル ホーム 挿入 描画 ページレイアウト 数式 データ 校閲 表示 自動化 ヘルプ

編集 ユーザー設定

F23 =PRODUCT(SUM(見積書[金額]),1.1)

	A	B	C	D	E	F	G	H	I	J	K	L	M	N
1														
2					見積書									
3														
4			ABC銀行	御中										
5						株式会社〇〇								
6		件名	IPO上場支援			〒123-4567								
7						東京都港区								
8		納期	2022年4月1日			クラウドS館25階								
9						TEL :								
10		支払条件	現金			FAX :								
11														
12		有効期限	2022年4月29日			担当者	テスト太郎							
13														
14														
15		合計金額	¥627,000											
16														
17			品目	数量	単位	単価	金額							
18			AAA	1	個	10,000	10,000							
19			BBB	3	個	20,000	60,000							
20			CCC	10	個	50,000	500,000							
21					小計		¥570,000							
22					消費税		¥57,000							
23					合計		¥627,000							
24														
25														
26														

Sheet1

計算モード: 自動 ブックの統計情報

Microsoft にフィードバックを送信 100%

画面 13-84

この Power Automate のクラウドフローでファイルのコピーや作成、Excel へのデータ入力処理、等の業務の自動化をすることができます。

見積書や請求書等の書類作成をする場合、ボタンをクリックするだけで書類作成が自動化されるので、わざわざ手作業で1つ1つデータ入力をするといった面倒な作業がなくなります。ぜひ、ご自身の大切な時間を奪っている面倒な雑務を自動化し、本当にやりたい仕事に集中しましょう。

Column JSON と仲良くなろう！

本 Column では、JSON について紹介します。解説の流れは次のとおりです。

1. JSON とは
2. JSON の記述方法
3. データ型
4. JSON の解析アクションを使ってみよう
5. まとめ

1. JSON とは

JSON とは JavaScript Object Notation の略で、ジェイソンと呼びます。構造化データを表現するための形式の一つで、データの送受信に使われることが多いです。

2. JSON の記述方法

基本的な記述方法は非常にシンプルで、以下の規則に従います。{} の中にキーと値をコロンで区切って記述します。

```
{"key": "value"}
```

また、複数記述の場合は、カンマで区切ります。

```
{  
  "key1": "value1",  
  "key2": "value2"  
}
```

3. データ型

データ型はデータの種類のことで、JSON は以下のデータ型に対応しています。

文字列

ダブルクォーテーションで囲む必要があります。

```
{"name": "太郎"}
```

数値

ダブルクォーテーションで囲みません。ダブルクォーテーションで囲むと文字列扱いになります。

```
{"age": 20}
```

Bool

true または false で指定します。

```
{"smoking": false}
```

配列

[]で囲む必要があります。配列内の要素はカンマで区切って複数記述します。

```
{"food": ["りんご", "みかん", "ばなな"] }
```

オブジェクト

キーと値の組み合わせからなっているデータです。

```
{"name": "太郎", "age": 20}
```

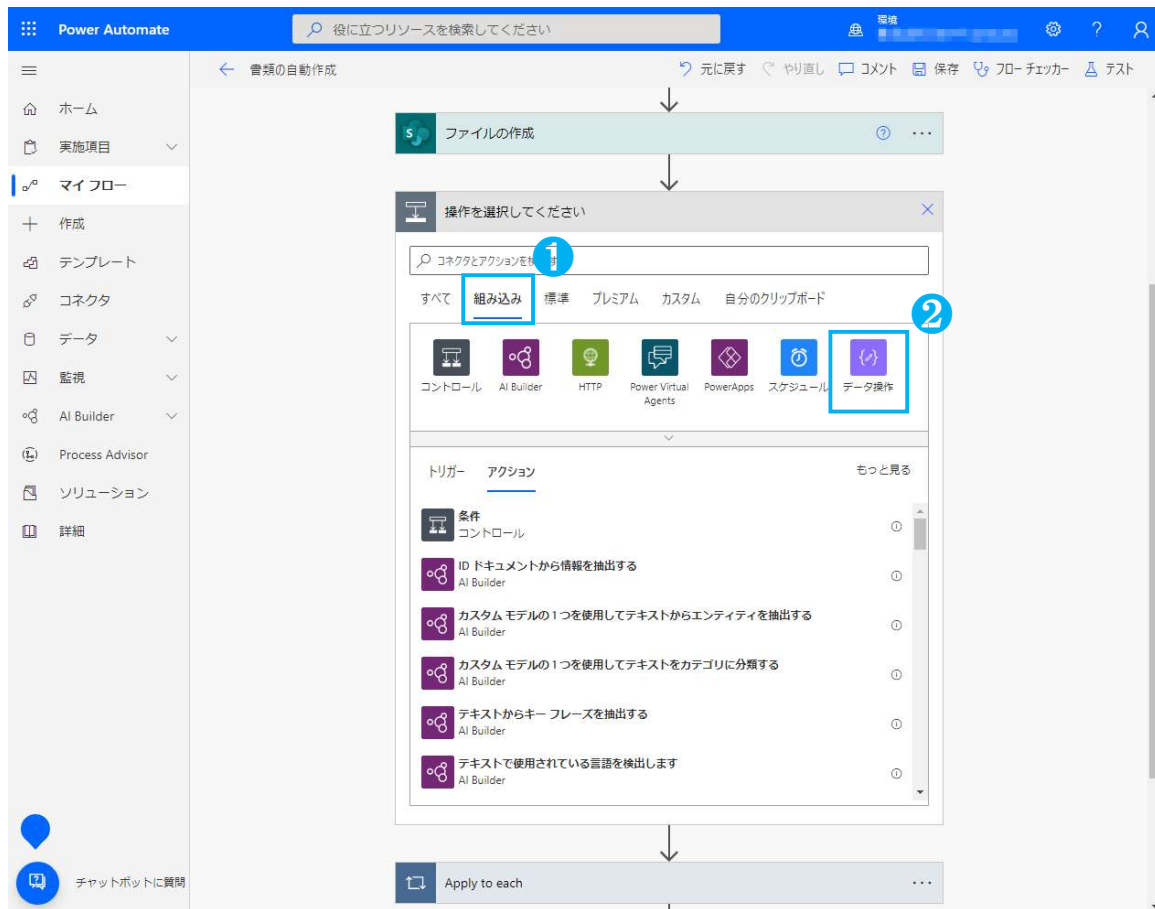
がオブジェクトです。

以下は、太郎さんの要素1 (オブジェクト) と花子さんの要素2 (オブジェクト) が配列になっています。

```
[
  {
    /// 要素1
    "name": "太郎",
    "age": 20,
    "job": "エンジニア"
  },
  {
    /// 要素2
    "name": "花子",
    "age": 18,
    "job": "デザイナー"
  }
]
```

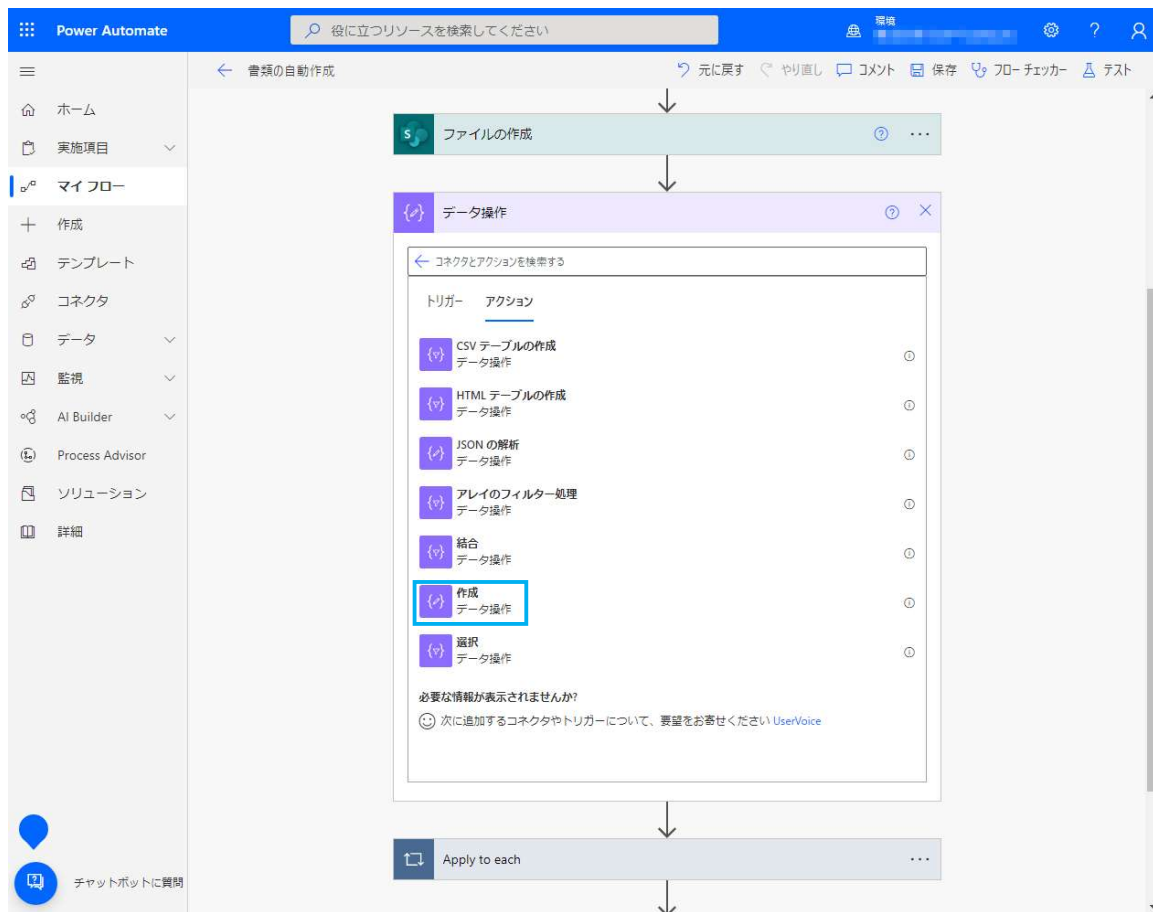
システムから取得したデータは、JSON 形式になっています。実際に、見積書テーブルのデータの中身を確認してみましょう。

[組み込み] タブ⇒ [データ操作] を選択します (画面 13-85)。

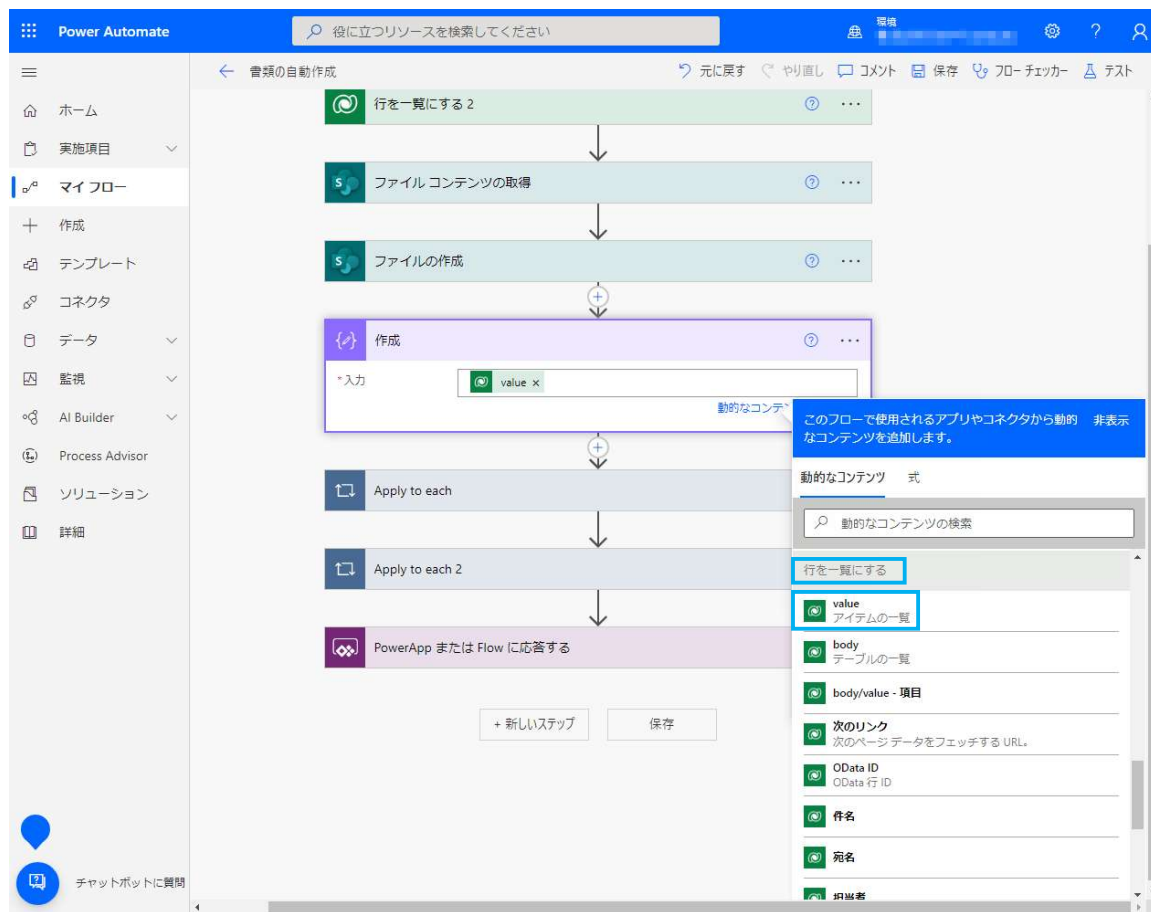


画面 13-85

表示されたアクション一覧から「作成」を選択し、[動的なコンテンツ] 画面から「行を一覧にする」⇒「value」を選択して挿入します（画面 13-86、13-87）。



画面 13-86



画面 13-87

試しに、クラウドフローを実行します（画面 13-88）。実行終了後、実行履歴から「作成」アクションの中身を確認すると、見積書 A のデータが JSON 形式で取得されています。

Power Automate

役に立つリソースを検索してください

編集 共有 名前をつけて保存 削除 実行 コピーの送信 テンプレートとして送信する エクスポート フローチェッカー

フロー > 書類の自動作成

詳細

フロー: 書類の自動作成

所有者: クラウド太郎

状況: オン

作成済み: 5月30日 12:54

変更日時: 5月30日 13:17

種類: すぐに

プラン: フローを実行するユーザー

28 日間の実行履歴

開始	時間	状況
5月30日 13:17 (45 秒 前)	00:00:20	成功

すべての実行

接続

Microsoft Dataverse

Excel Online (Business)

SharePoint アクセス許可

所有者: クラウド太郎

プロセス分析情報 (プレビュー) フローを改善

フローを改善

分析に必要なデータを準備しています。数分後に再度確認し、フローを分析してください。

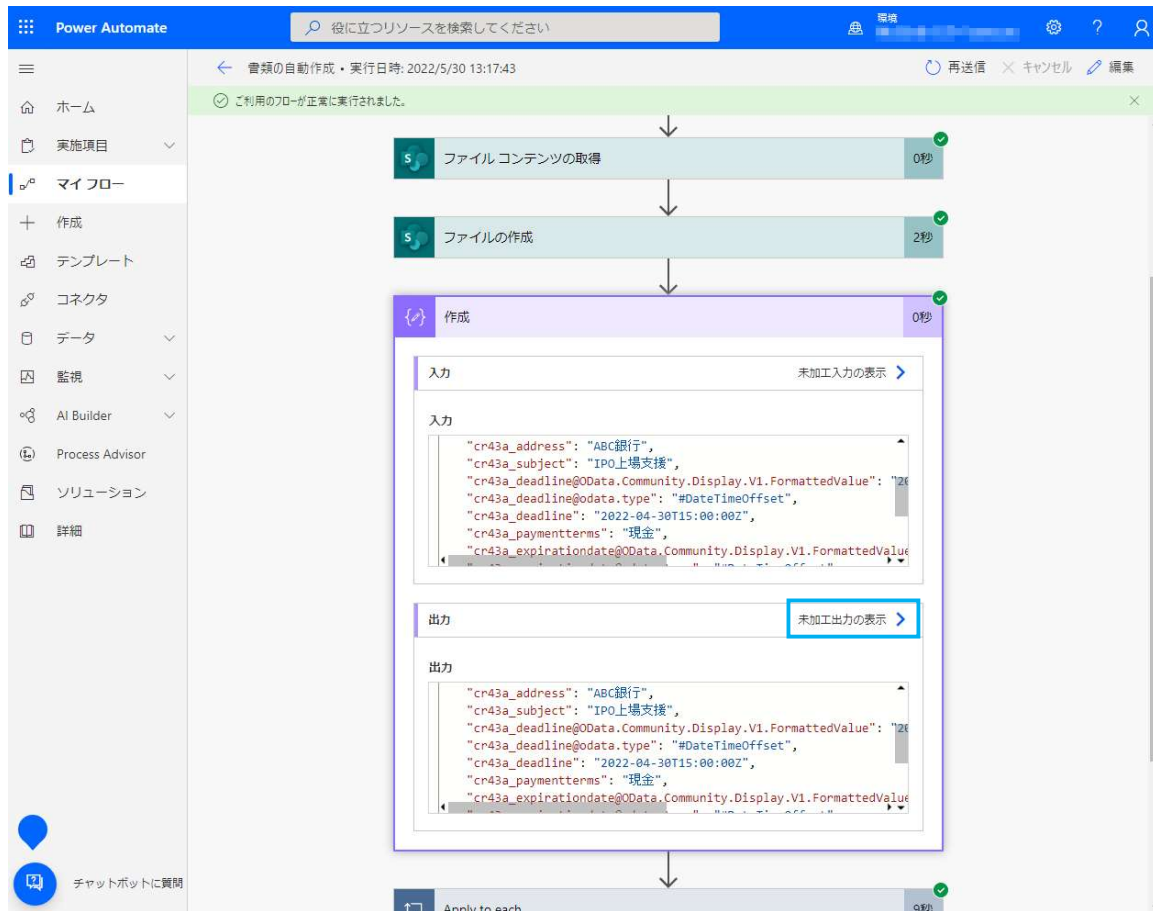
実行のみのユーザー

フローは、他のユーザーと共有されていません。

チャットボットに質問

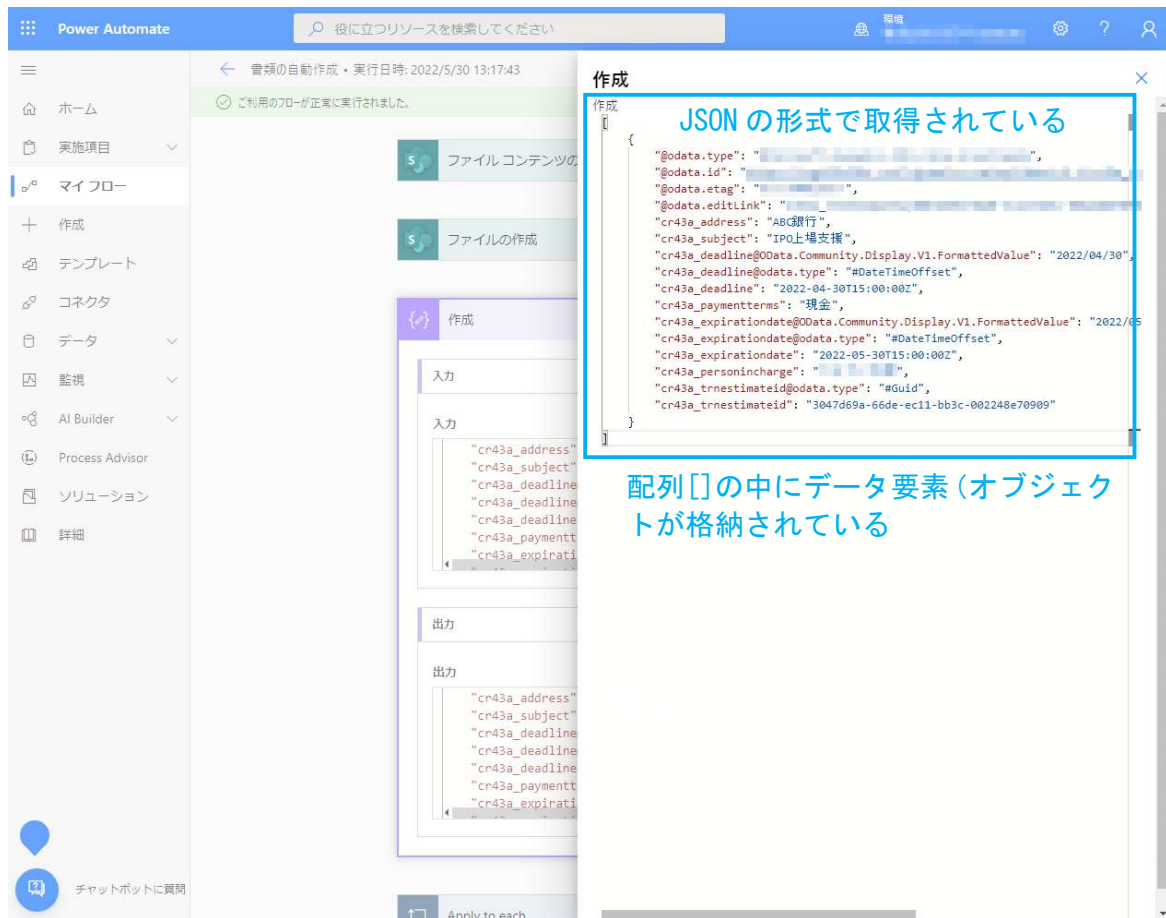
画面 13-88

[未加工出力の表示 >] を選択すると、拡大表示されて見やすくなります (画面 13-89)。



画面 13-89

値の中身を確認してみると、データが JSON 形式で取得されています（画面 13-90）。

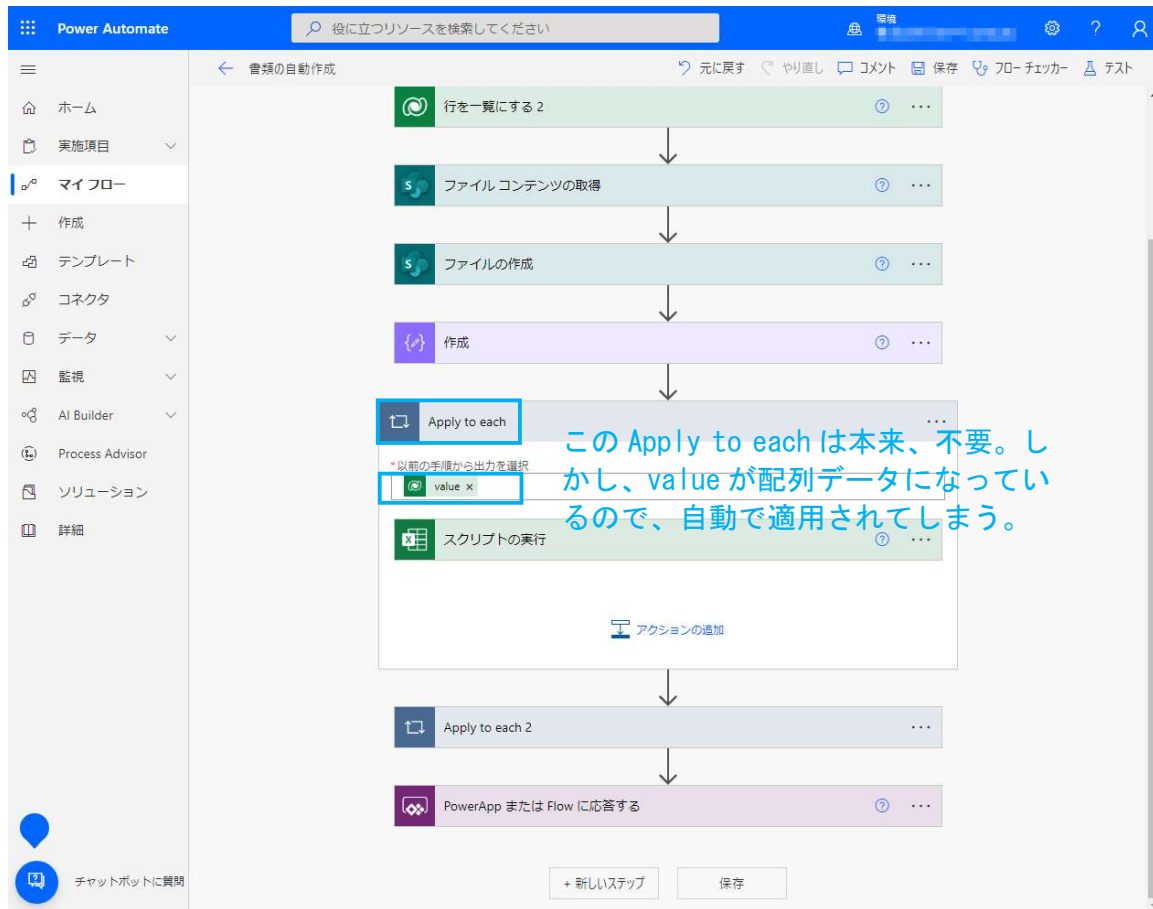


画面 13-90

4. JSON の解析] アクションを使ってみよう

[スクリプトの実行] アクションで [Apply to each] が自動で適用されています。

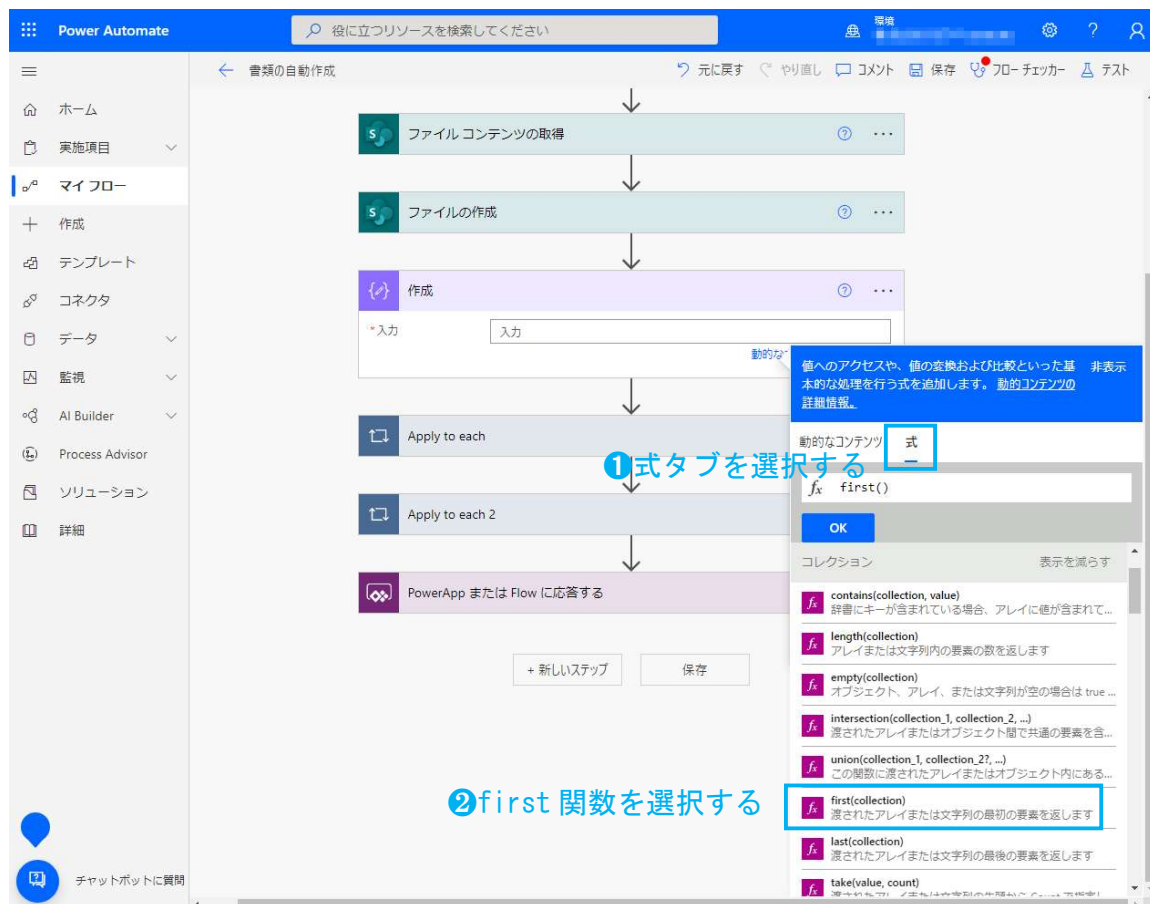
今回は [見積書テーブル] の [タイトル] 列に同じ値は入力されない想定なので、[行のフィルター] で必ずデータは 1 行のみに絞り込まれます。つまり、この自動で適用された [Apply to each] は本来、不要なものです。単一のデータにループ処理をする必要はないからです (画面 18-91)。



画面 13-91

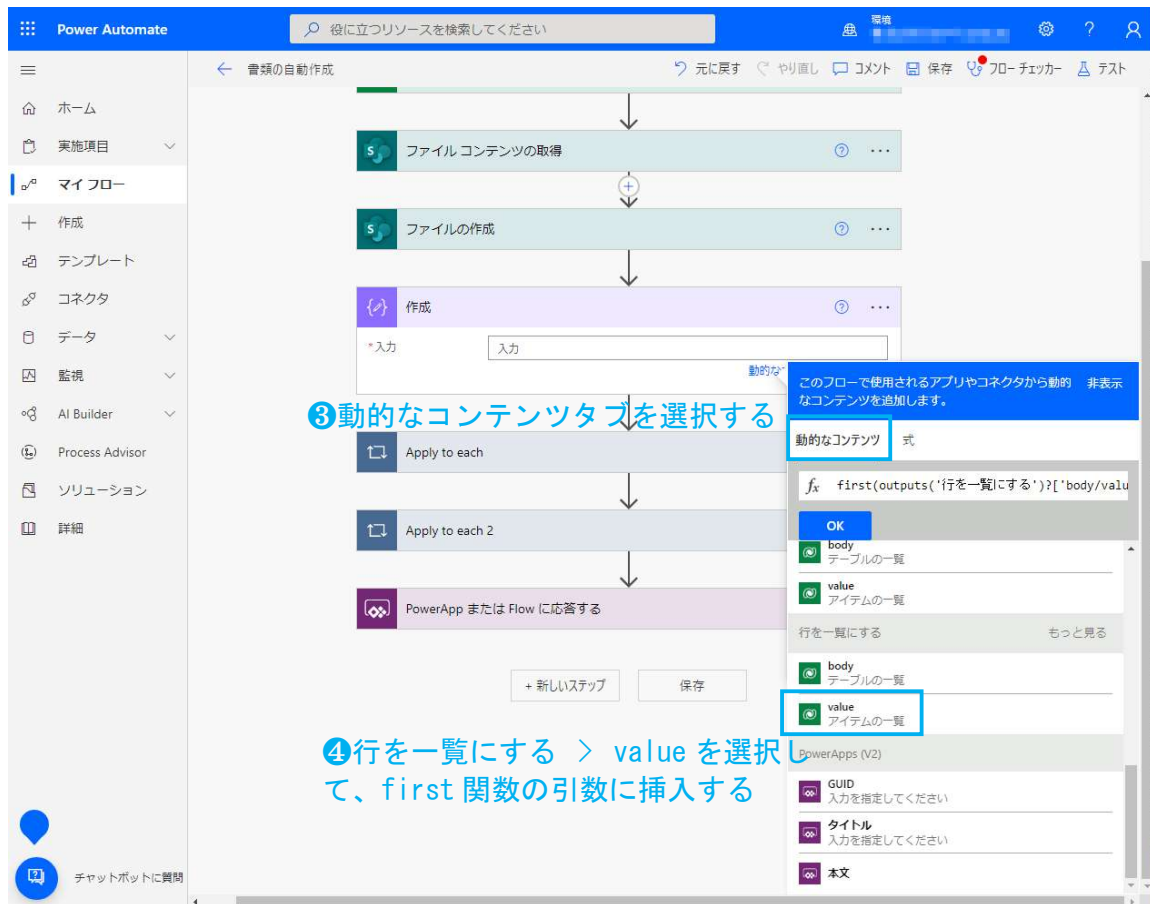
「Apply to each」が自動で適用されてしまうのは、データを1件だけ取得できていても、取得結果が配列になっているのが原因です。画面 18-90を確認すると、配列形式でデータを取得しています。このような場合は、first関数を利用します。

「動的なコンテンツ」画面の式タブを選択し、[コレクション]⇒[first]を選択します（画面 13-92）。



画面 13-92

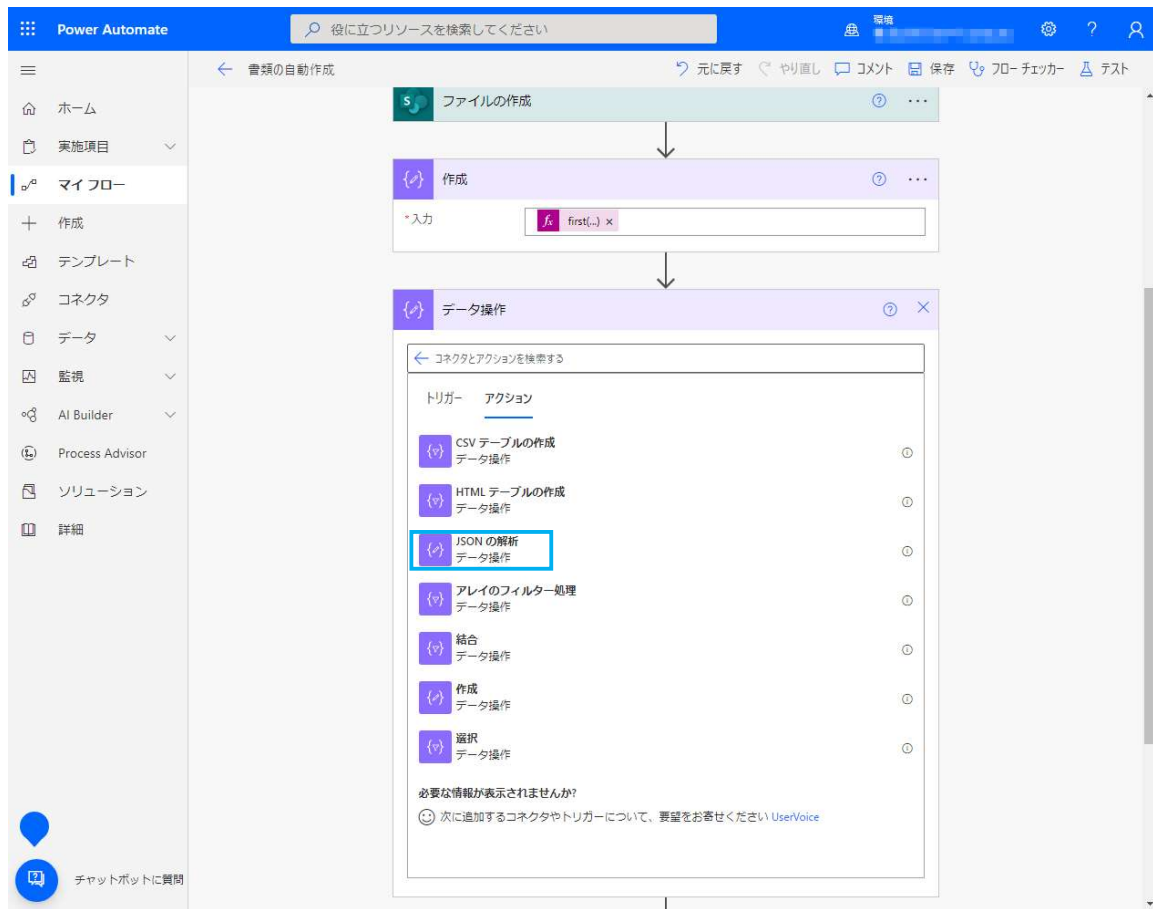
[動的なコンテンツ] タブに移動し、[value] を選択して、first 関数の引数に挿入します (画面 13-93)。



画面 13-93

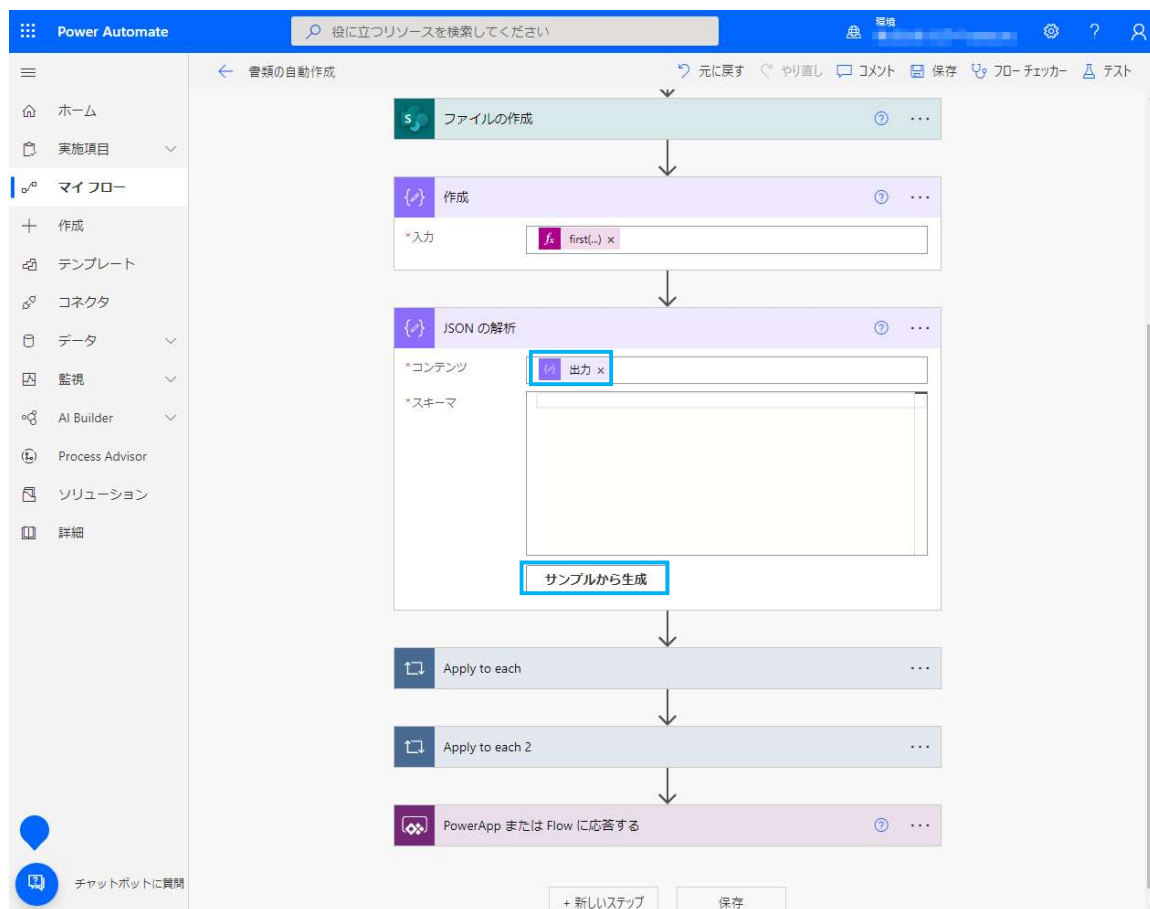
こちらの手順で不要な Apply to each の適用を回避することができます。しかし、このままでは扱いづらく、[作成] アクションで取得した見積書データの中身を抽出したいです。このような場合に [JSON の解析] アクションを利用します。

同様に、[データ操作] の表示されたアクション一覧から [JSON の解析] を選択します（画面 13-94）。



画面 13-94

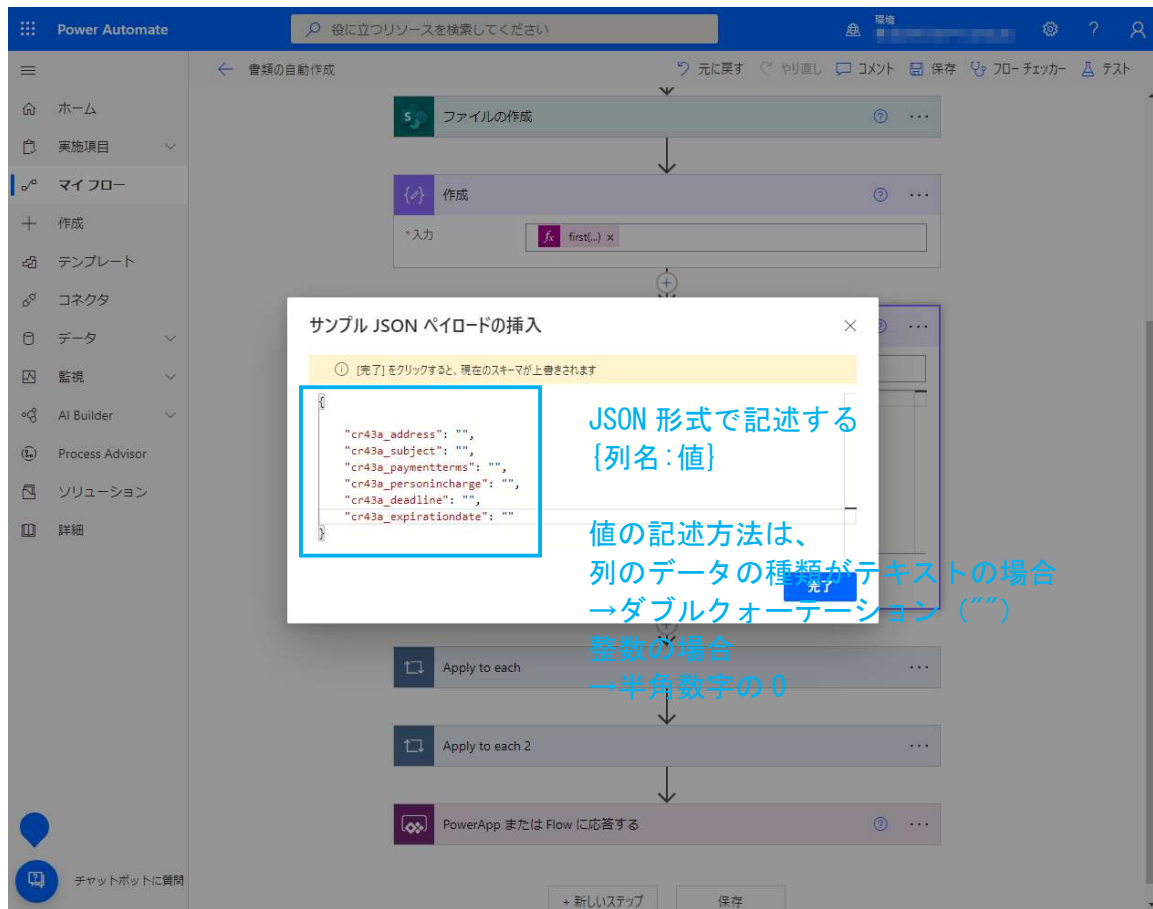
コンテンツは、[動的なコンテンツ] 画面から [作成] ⇒ [出力] を選択して挿入します。スキーマは、[サンプルから生成] を選択します (画面 13-95)。



画面 13-95

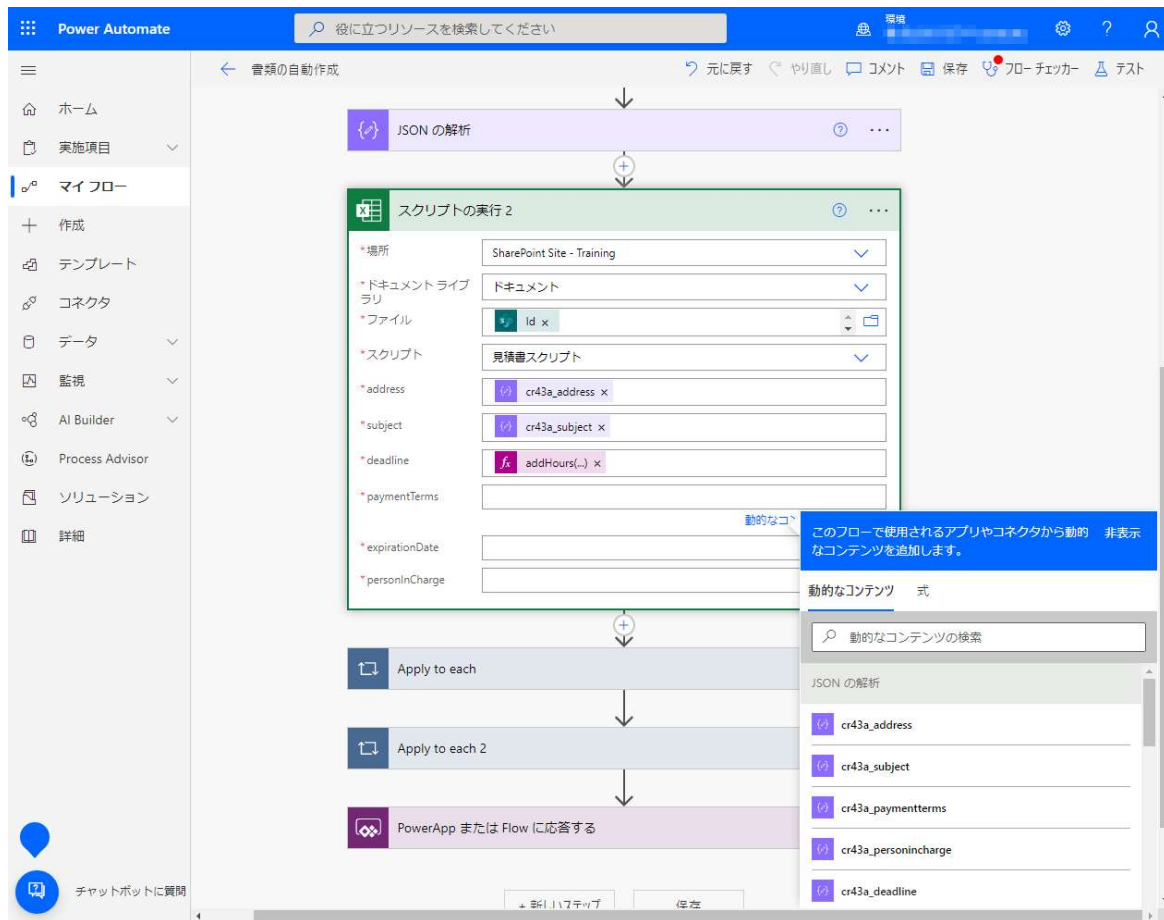
[サンプル JSON ペイロードの挿入] が表示されます。

こちらに取得したい見積書データを JSON で記述し、[完了] を選択します (画面 13-96)。詳細は [関数の入力補助] フォルダに格納されている <Column_パラメータシート.xlsx>の<接頭辞の確認・JSON の解析>をご参照ください。



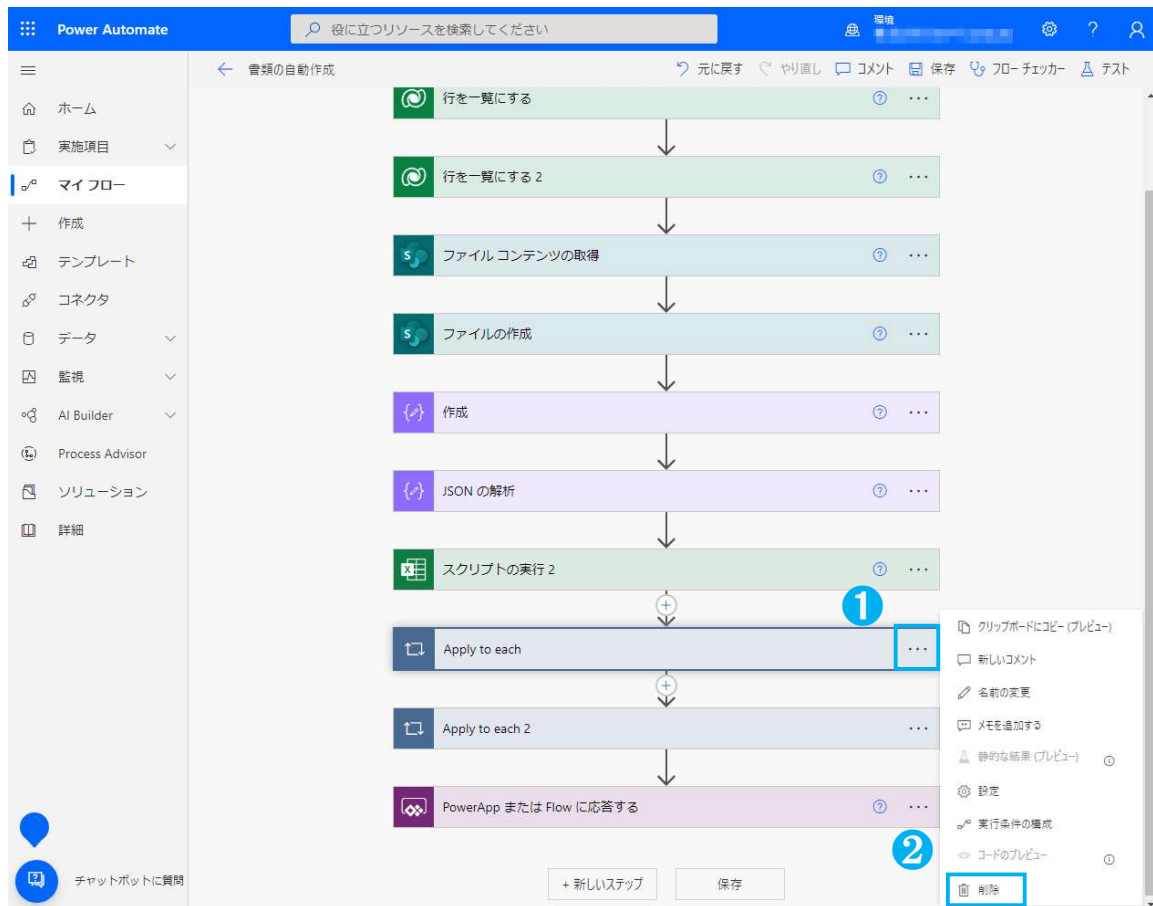
画面 13-96

あとは通常通りの手順で「スクリプトの実行」アクションを設定し、「動的なコンテンツ」画面から「JSON の解析」アクションで取得した見積書データを挿入します（画面 13-97）。



画面 13-97

Apply to each を含む [スクリプトの実行] アクションは不要なので削除しておきましょう (画面 13-98)。



画面 13-98

5. まとめ

first 関数で不要な Apply to each の適用を回避し、JSON の解析でデータを取得することで、よりシンプルなクラウドフローを作成することができます。

JSON と聞くと、難しそうな専門用語と認識して、最初はなんとなく身構えてしまうかもしれません。試しに、たくさん [JSON の解析] アクションを使うことで、このアクションの便利さが自然と理解できるようになります。[JSON の解析] アクションを使って JSON と仲良くなりましょう！