Dokumentenvorlage am Beispiel Angebot

Einleitung

Manchmal ist es sinnvoll von einer handvoll Dokumente eine Vorlage auswählen zu können, damit dem Anwendern die kontinuierliche Eingabe von immer den gleichen Datenpunkten erspart bleiben kann.

In diesem Beispiel ist eine Vorlage für den Dokumententyp Angebot (engl. Quotation) aufgeführt.

Planung

Wir legen einen neuen Dokumententyp namens "Angebotsvorlage" an. Dieser enthält genau die gleichen Felder, die das Angebot enthält minus die Felder, die im Anwendungsbeispiel nicht gebraucht werden. Minimal brauchen wir daher alle Pflichtfelder aus dem Dokumententyp "Angebot".

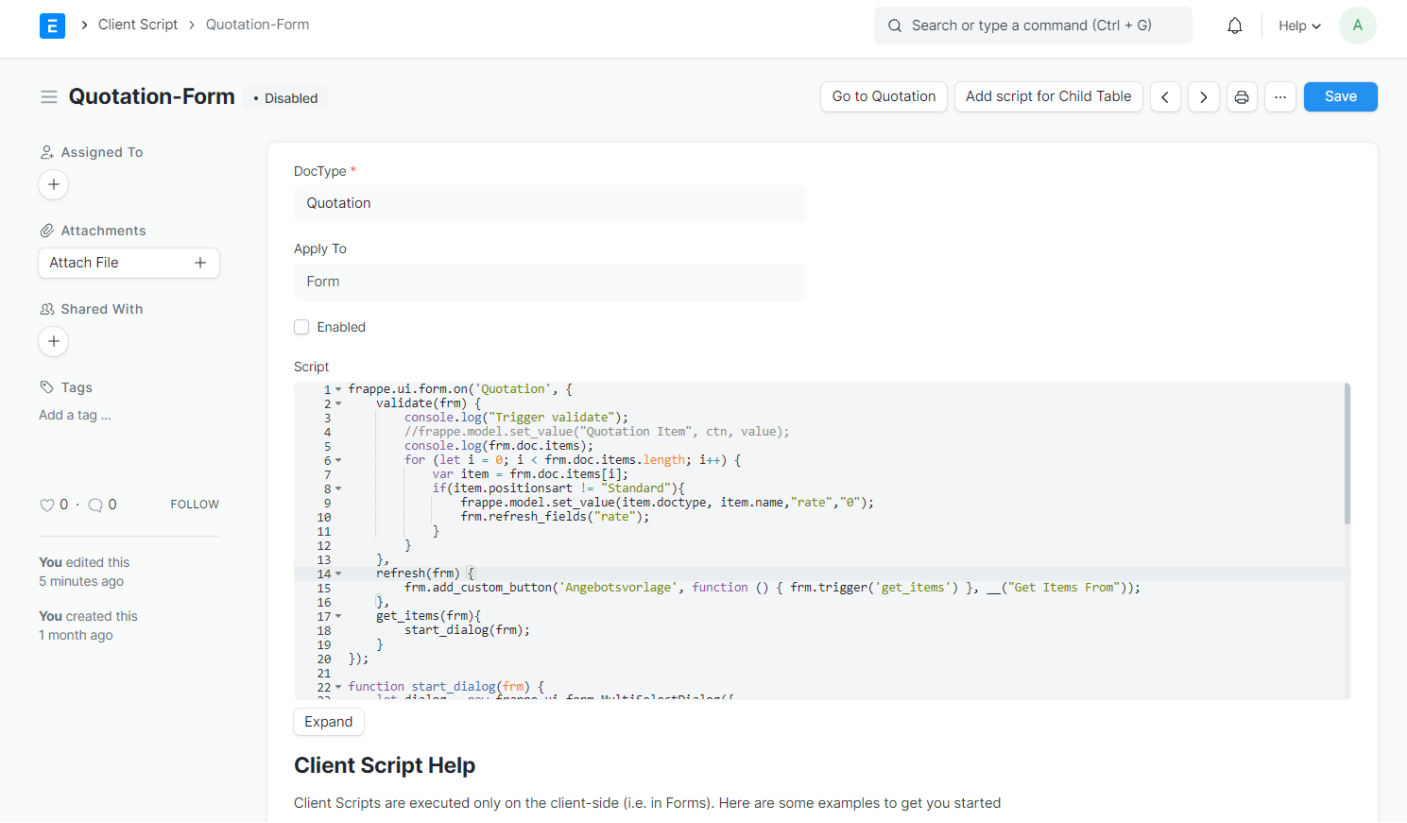
Nicht möglich ist es manchmal die Tabellen 1:1 mit zu nehmen. Im Angebot werden innerhalb der Tabelle 'items' die "Quotation Item" in mehrfachen Positionen gelistet. In unserem erstellen Dokumententyp "Angebotsvorlage" kann das "Quotation Item" zwar mit einer Tabelle eingebaut werden, ist aber nicht brauchbar. Grund sind ERPNExt Funktionen, die im Hintergrund ausgeführt wenn wir eine neue Zeile von Quotation Item in einer Quotation anlegen.

Aus diesem Grund bauen wir einen zweiten Dokumententyp "Angebotsvorlage Item". Dieser enthält genau die gleichen Felder, die das "Quotation Item" enthält minus die Felder, die im jeweiligen Anwendungsbeispiel nicht gebraucht werden.

Sobald ein Dokument vom Typ "Angebotsvorlage" erstellt werden kann ist dieses zu speichern. Wir stellen im nächsten Schritt ein neues "Angebot" indem wir die Vorlage öffnen und folgend der Anleitung am Ende der Seite.

Skript

Das Skript wird als Client Skript eingebunden und beruht auf dem Doctype "Quotation".



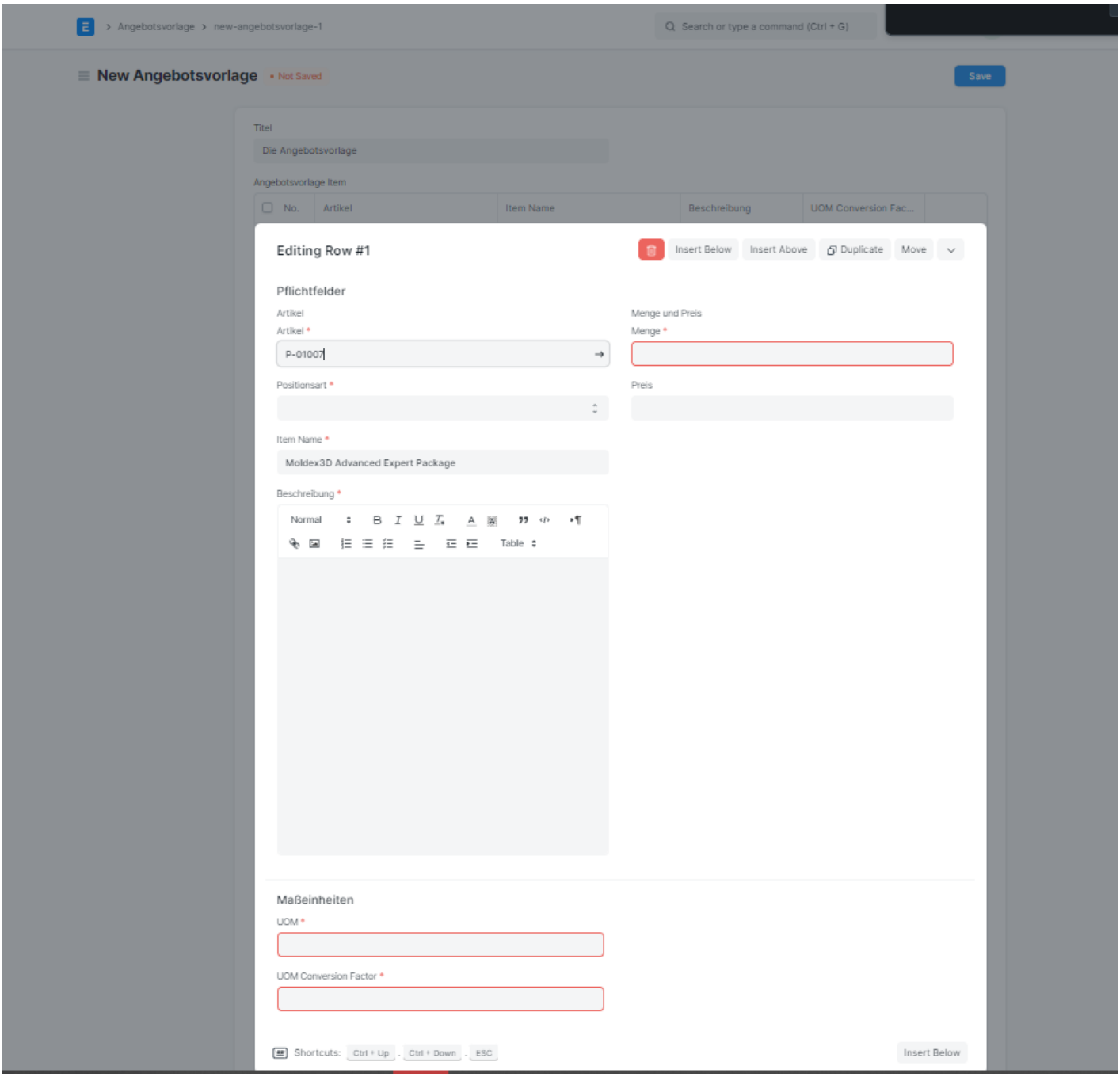

```
// The fetch-from fields
var fields = [
  "item_code",
  "item_name",
  "positionsart",
  "description",
  "qty",
  "uom",
  "rate"];


frappe.ui.form.on('Quotation', {
```

```
	refresh(frm) {
	    frm.add_custom_button('Angebotsvorlage', function () { frm.trigger('get_items') }, __("Get Items From"));
	},
	get_items(frm){
	    start_dialog(frm);
	}
});

function start_dialog(frm) {
	let dialog = new frappe.ui.form.MultiSelectDialog({

		// Read carefully and adjust parameters
		doctype: "Angebotsvorlage", // Doctype we want to pick up
		target: frm,
		setters: {
			// MultiDialog Filterfields
			// customer: frm.doc.customer,
		},
		date_field: "creation", // "modified", "creation", ...
		get_query() {
			// MultiDialog Listfilter
			return {
				filters: {  }
			};
		},
		action(selections) {
		    for(var n = 0; n < selections.length; n++){
		        var name = selections[n];
		        frappe.db.get_doc("Angebotsvorlage", name) // Again, the Doctype we want to pick up
                .then(doc => {
                    // Remove the first empty element of the table
                    if(!('item_code' in frm.get_field("items").grid.grid_rows[0].doc)){
                        frm.get_field("items").grid.grid_rows[0].remove();
                    }

                    // Run through all items of the template quotation
                    for(var n = 0; n < doc.angebotsvorlage_item.length; n++){
                        // Declare variables and add table row
                        var item=doc.angebotsvorlage_item[n];
```

```
                var row=frm.add_child("items"); // Zeile anlegen
                frm.refresh_fields("items"); // Refresh Tabelle

                // Copy-Paste Operation
                for(var m = 0; m < fields.length; m++){
                    frm.get_field("items").grid.grid_rows[n+1].doc[fields[m]] = item[fields[m]];
                    frm.get_field("items").grid.grid_rows[n+1].refresh_field(fields[m]);
                }
                frm.refresh_fields("items"); // Refresh Tabelle
            }
        });
□□    }
□□}
□});
}
```

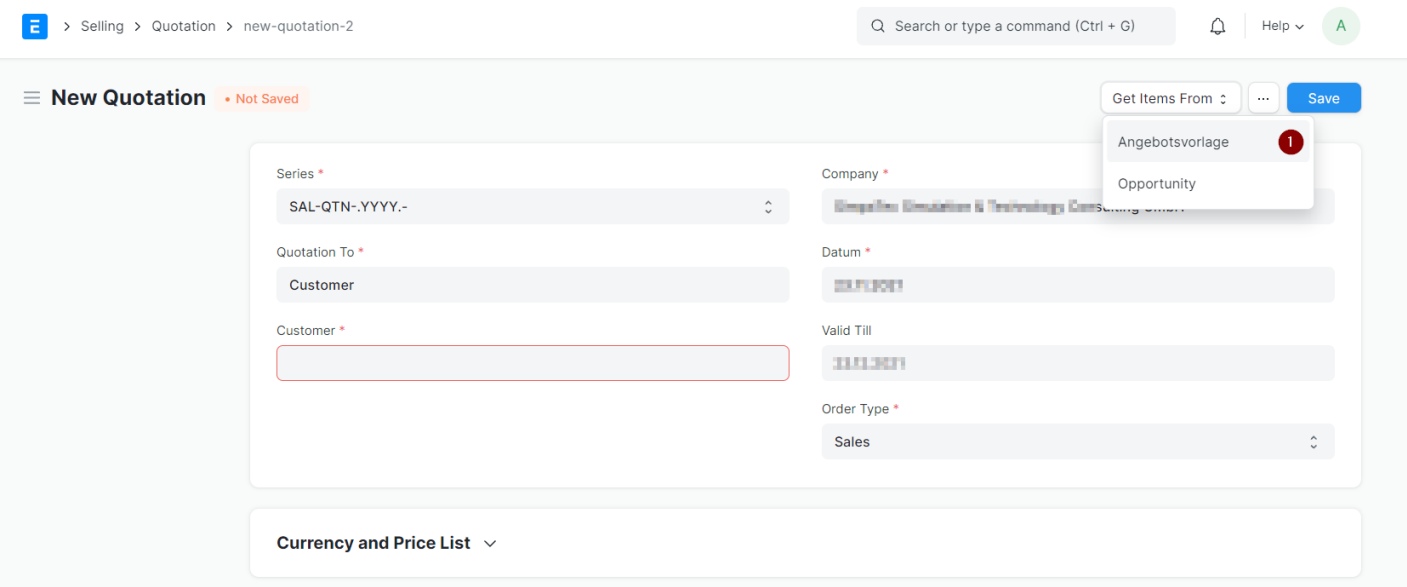
Entwurf der Angebotsvorlage

Der Entwurf enthält alle Pflichtfelder des "Quotation Items".

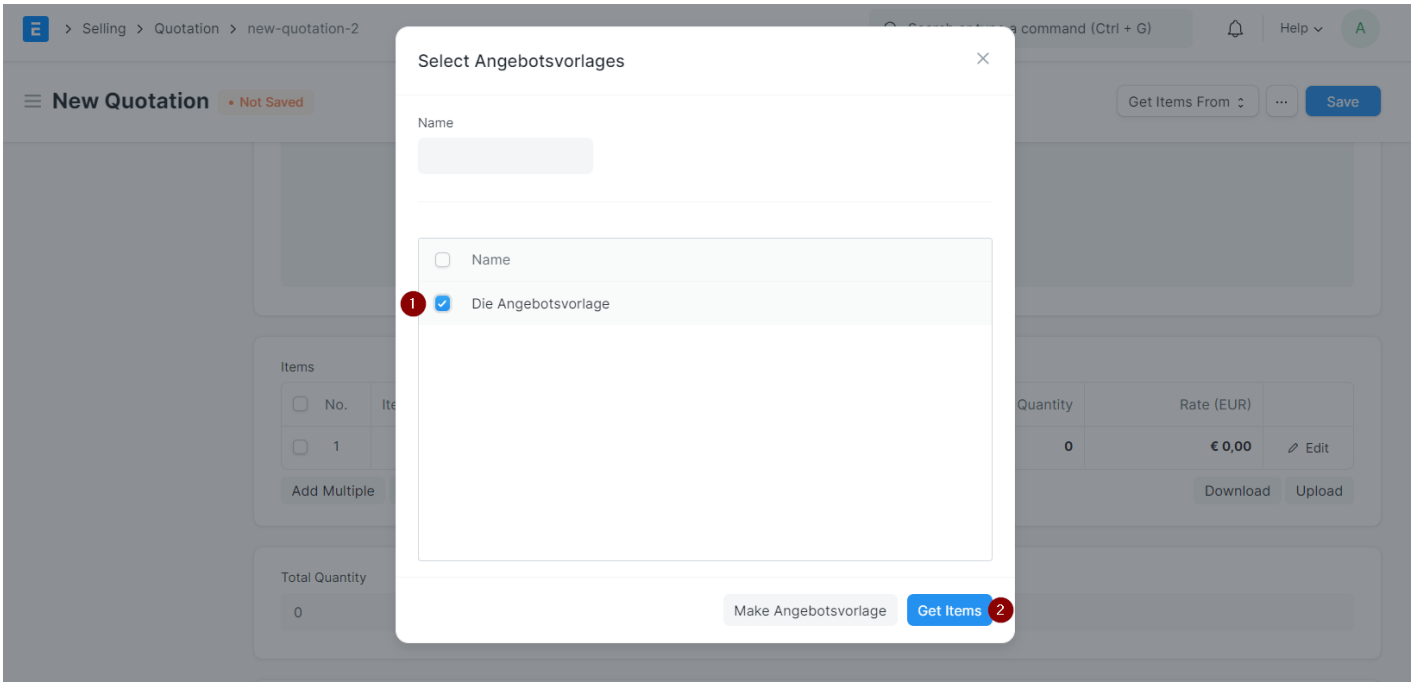


Anleitung

Wir erstellen eine neues Angebot und betätigen den 'Angebotsvorlage'-Button (1) im Dropdownmenü 'Get Items From'.



Im folgenden MultiSelect-Dialog wählen wir die gewünschte Angebotsvorlage aus und selektieren diese in der Mehrfachauswahl (1) und bestätigen mit (2).



Danach erscheinen die Positionen aus der Angebotsvorlage im Angebot.

> Bekannter Bug: Der Dialog funktioniert nur einmalig. Bei mehrfachem 'Get Items From'->'Angebotsvorlage' werden Leerzeilen in der Positionstabelle eingefügt.

Weitere Ausbaustufen

Anstelle des Client Script, der von uns entwickelt wurde, kann eventuell die ERPNext eigene Funktion in Zeile 95 in folgendem Code-Blob verwendet werden.

https://github.com/frappe/erpnext/blob/develop/erpnext/selling/doctype/quotation/quotation.js

Die Funktion an Position (1) ist eine oft verwendete Ressource und kann wiederverwendet werden. Die Aufgerufene Methode (2) ist spezifisch um ein Angebot zu erstellen. Das ganze ist als Client Skript anzulegen.

```
if (this.frm.doc.docstatus===0) {
    this.frm.add_custom_button(__('Opportunity'),
        function() {
            erpnext.utils.map_current_doc({
                method: "erpnext.crm.doctype.opportunity.opportunity.make_quotation",
                source_doctype: "Opportunity",
                target: me.frm,
                setters: [
                    {
                        label: "Party",
                        fieldname: "party_name",
                        fieldtype: "Link",
                        options: me.frm.doc.quotation_to,
                        default: me.frm.doc.party_name || undefined
                    },
                    {
                        label: "Opportunity Type",
                        fieldname: "opportunity_type",
                        fieldtype: "Link",
                        options: "Opportunity Type",
                        default: me.frm.doc.order_type || undefined
                    }
                ],
                get_query_filters: {
                    status: ["not in", ["Lost", "Closed"]],
                    company: me.frm.doc.company
                }
            })
        }, __("Get Items From"), "btn-default");
}
```

Die Setters können wahrscheinlich im ersten Schritt leer gelassen werden, genauso wie die Filter.

```
this.frm.add_custom_button(__('Opportunity'),
□□□□function() {
```

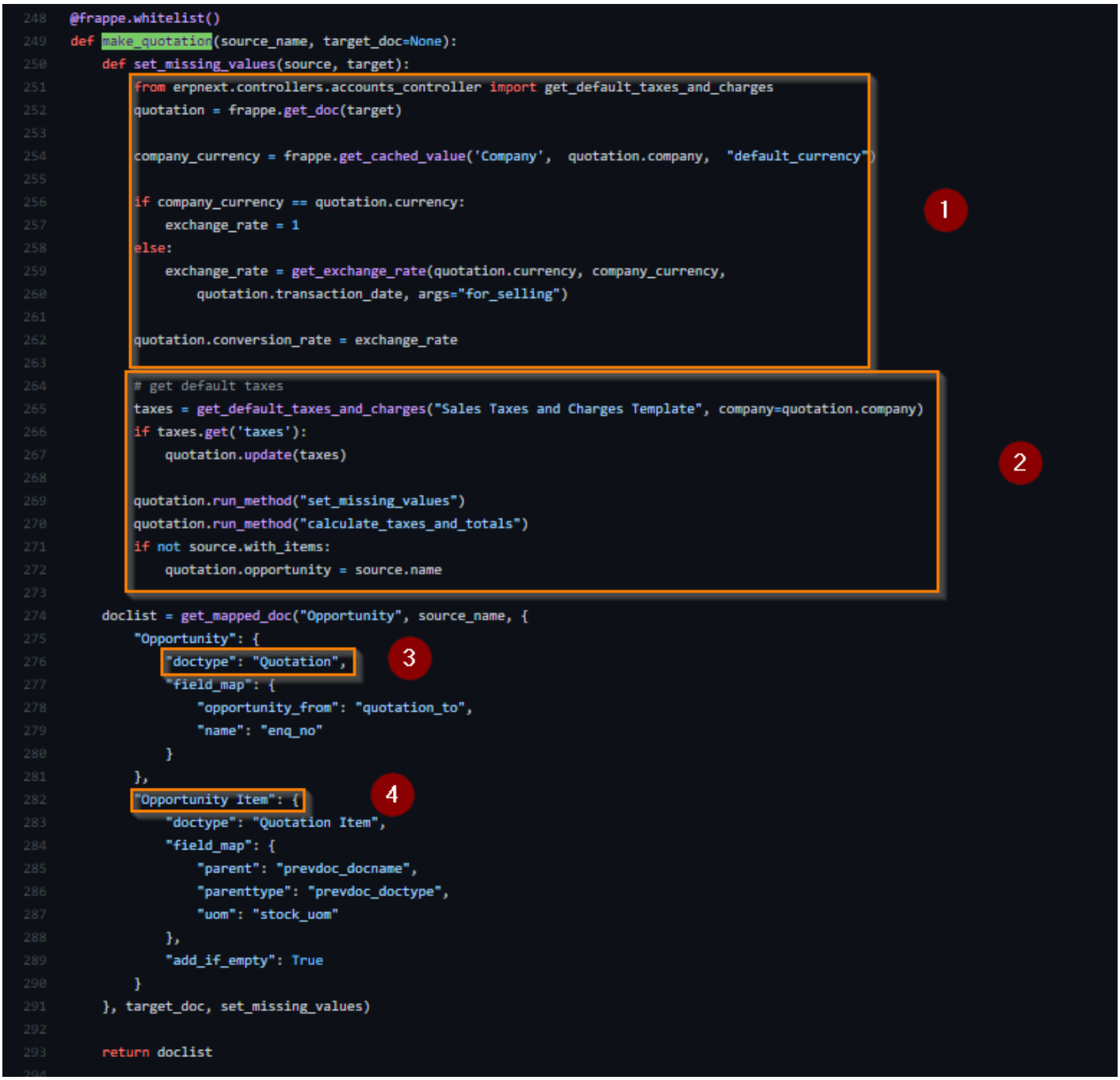
```
□□□□□erpnext.utils.map_current_doc({
□□□□□□method: "erpnext.crm.doctype.opportunity.opportunity.make_quotation",
□□□□□□source_doctype: "Opportunity",
□□□□□□target: me.frm,
□□□□□□setters: [
□□□□□□],
□□□□□□get_query_filters: {
□□□□□□}
□□□□□})
□□□□}, __("Get Items From"), "btn-default");
```

Die in der Funktion aufgerufene whitelist-Methode make_quotation beinhaltet den Backend-Skript im Codeblob

https://github.com/frappe/erpnext/blob/6954dd6329f2af0db88413fe933c5e1a111bcb9b/erpnext/crm/doctype/opportunity/opportunity.py

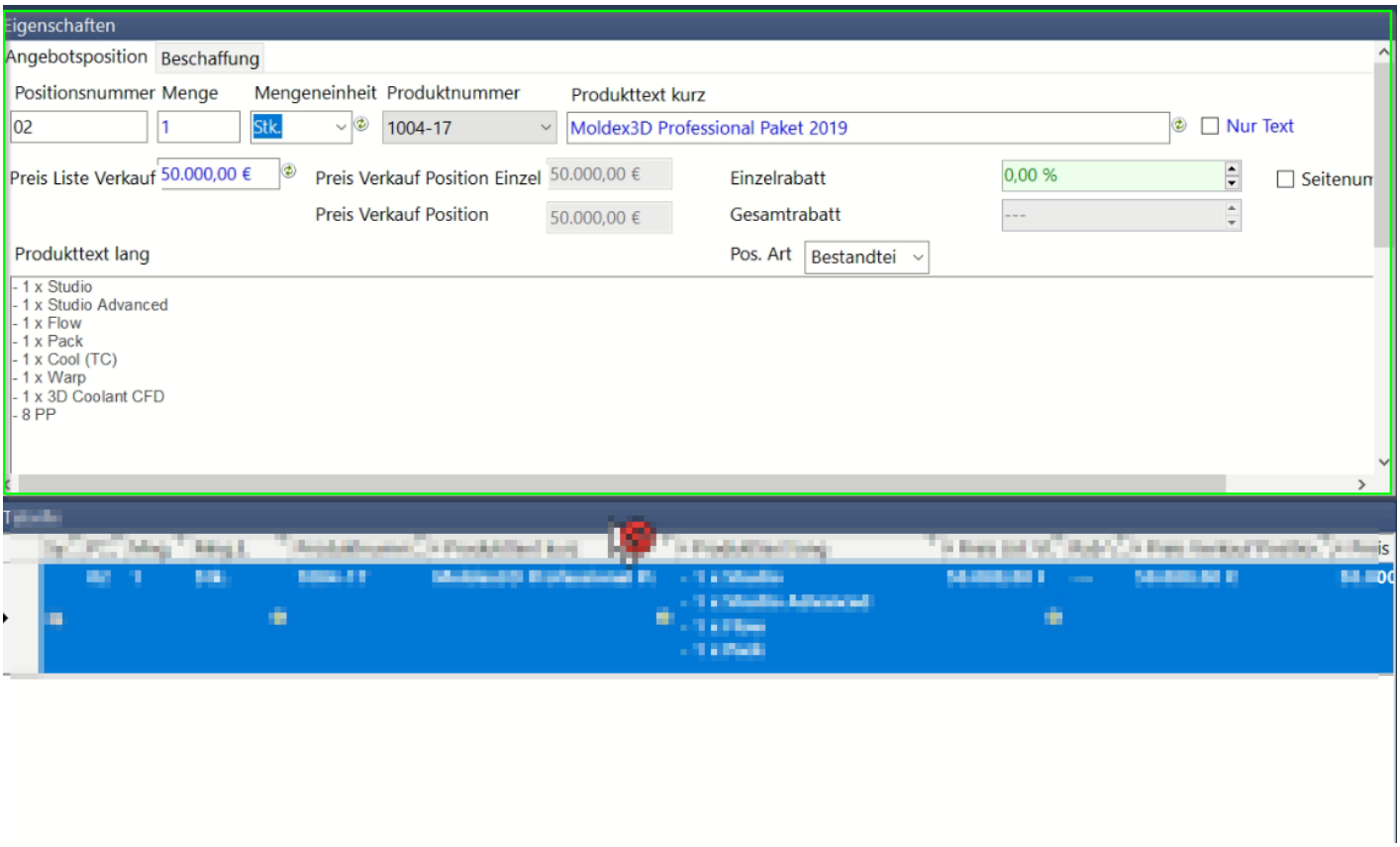
Dieser Blob kann wahrscheinlich gecopied, gepasted werden. Codeblob an Position (1) kann ignoriert oder gelöscht werden. Codeblob an Position (2) auch.

Die Zeilen (3) und (4) müssen angepasst werden. Das ganze ist als gewhitelistetes Server Skript anzulegen.



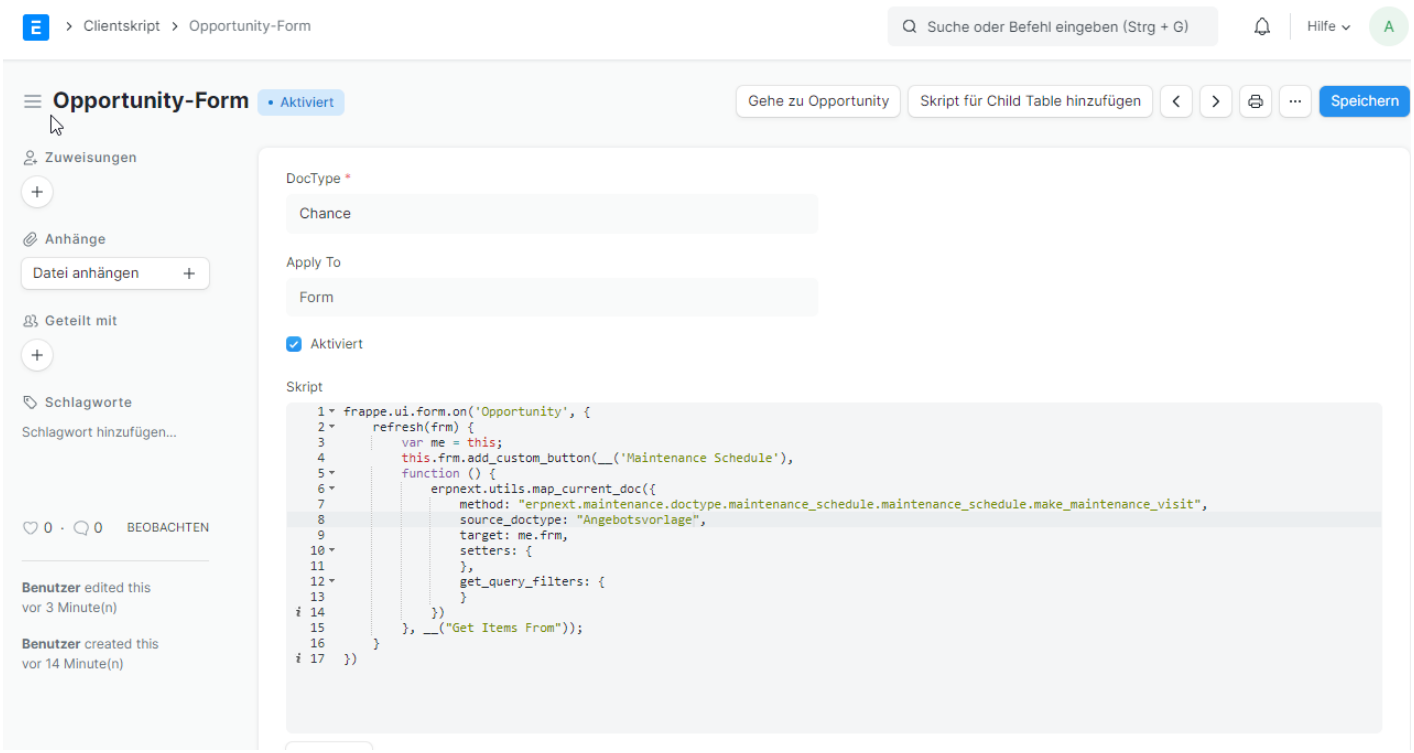
Abstimmung mit Herr Steffen Paul 26.11.2021

Die Felder im Kasten Eigenschaften sollten vom Angebot bis hin zur Ausgangsrechnung durchgezogen werden.

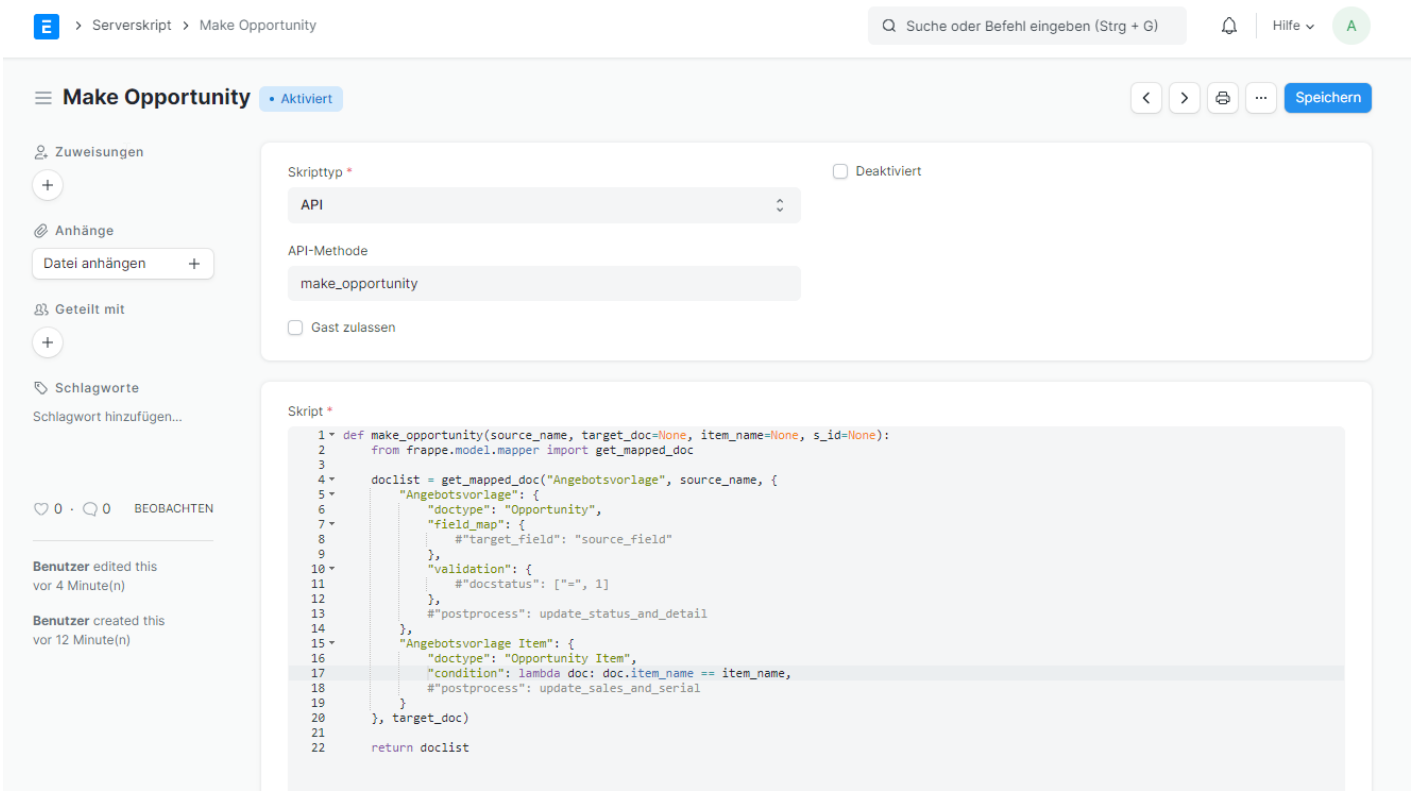


Versuch die Ausbaustufe der Angebotsvorlage auf die Opportunity anzuwenden

Clientscript



Serverscript



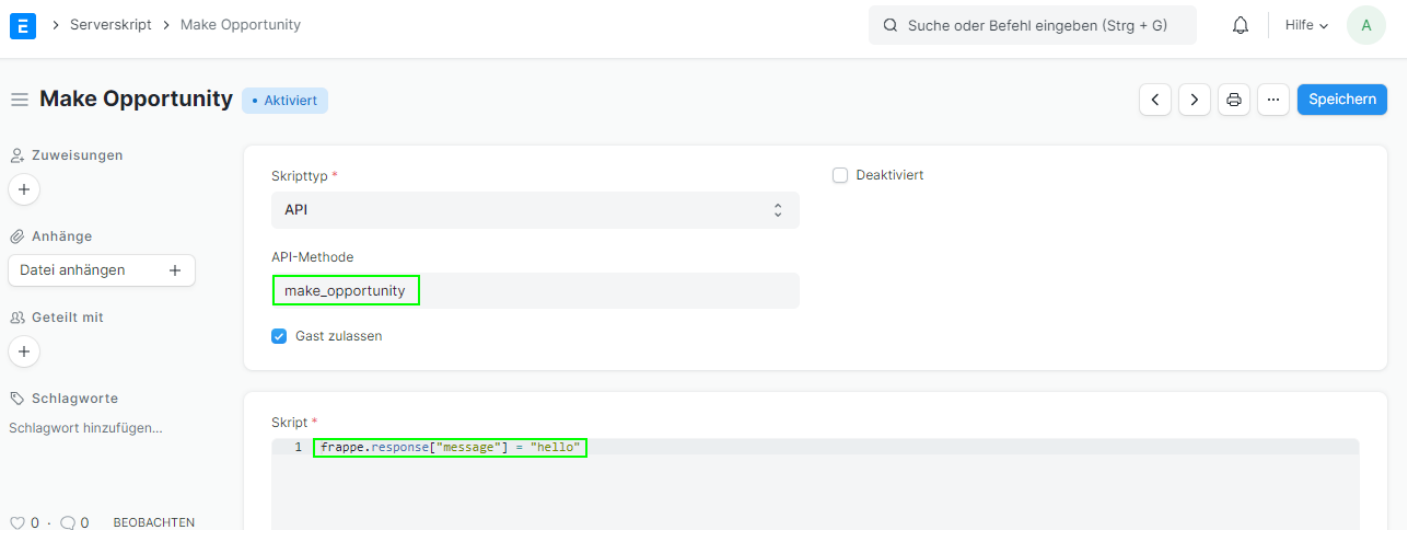
Ergebnisse

Auf diesem Weg kann eine Childtable nicht übertragen werden. Jedoch war es möglich einen aus zwei Komponenten bestehenden Front-End / Back-End Skript zu erstellen.

Client Skript auf Opportunity

```
frappe.ui.form.on('Opportunity', {
	refresh(frm) {

		frm.add_custom_button(
			'Angebotsvorlage',
			frappe.call({
				method: "make_opportunity",
				args: {
					'doctype': 'Item'
				},
				callback: function(r){
					console.log(r);
				},
			}),
			__("Get Items From"));
	}
})
```

Back End Skript vom Typ API



Warum funktioniert dies nicht mit der Angebotsvorlage?

Der Client Script ruft hierbei nicht die Funktion frappe.call(...) (wie in dem letzten Beispiel), sondern die Funktion frappe.model.mapper.get_mapped_doc(...) (vorletztes Beispiel) auf. Dieser

Funktionsaufruf ist so wie es aussieht nicht von einem Client Script aufrufbar. Sehr schade, aber leider nicht abänderbar.

Weitere Erkenntnisse

Server Script haben ein Sicherheitslayer, welches nur eine Teilmenge der in Python üblichen Methoden & Funktionen aufzurufen:

https://docs.erpnext.com/docs/v13/user/manual/en/customize-erpnext/server-script#23-api-scripts

Weitere Dokumentation

https://frappeframework.com/docs/v13/user/en/desk/scripting/server-script

https://frappeframework.com/docs/v13/user/en/guides/basics/frappe_ajax_call#calling-standard-api

Robuste finale Version

Die Einbau- und Anbringungsanleitung findet sich im ERPNext Benutzerhandbuch und kann auf andere Doctypes angewandt werden. Siehe https://doku.phamos.eu/books/erpnext-benutzerhandbuch/page/copy-childtable-get-items-from

---

Version #22
Erstellt: 19 November 2021 12:46:38
Zuletzt aktualisiert: 23 November 2023 14:39:25