kalmstrom.com Business Solutions

PnP List Generator

User Guide

Design SharePoint lists once, and create them as PnP PowerShell, Site Script, PnP XML, JSON or a Power Automate flow

SharePoint Online, Subscription Edition, 2019, 2016 and 2013

Contents

1 **Introduction** 3

1.1 Key Capabilities 4

2 **Getting Started** 5

2.1 Interface Overview 5

2.2 Your First List 7

3 **Designing Lists** 9

3.1 Supported Field Types 9

3.2 List Card Anatomy 12

3.3 Field Settings 13

3.4 Lookup Relationships 15

3.5 Canvas Controls & Context Menu 16

4 **List Settings & Content Types** 18

4.1 Content Types 19

5 **Views & Aggregations** 21

5.1 Aggregations & JSON Formatting 22

6 **Managing Data** 23

6.1 CSV & Excel Import 24

7 **Generating Output** 25

7.1 Output Formats 25

7.2 Feature Matrix 29

7.3 Power Automate ZIP Import 30

7.4 REST API Import 30

8 **Templates & Shortcuts** 32

CHAPTER 1

Introduction

> If you have ever provisioned a SharePoint site by hand-writing PnP PowerShell, debugging a malformed Provisioning XML, or copying field definitions between four lists, you know the cost: one typo in an `InternalName` breaks the whole script, and the script you write for one tenant rarely runs cleanly on the next.
>
> PnP List Generator replaces that workflow with a single canvas. Drag a Choice field, name it once, and ship the same schema as PnP PowerShell, Site Script, PnP Provisioning XML, a Power Automate solution, and a portable JSON backup – regenerated from the same source whenever the design changes. Free, browser-only, no install.

1.1 Key Capabilities

Visual canvas

Drag field types onto list cards and arrange lists spatially. The relationship between Orders and Customers is something you see, not something you remember from a 400-line script.

Five output formats from one design

JSON, PnP PowerShell, Site Script, PnP Provisioning XML, and Power Automate. Add a column on Monday, regenerate the PowerShell on Tuesday, regenerate the Site Script on Wednesday – they stay in sync because they come from the same canvas.

Reverse-engineer existing sites

Inherited a tenant nobody documented? Point PnP List Generator's **REST Import** at the site, paste the JSON response, and every list, field, and lookup lands on the canvas – ready to document, modify, or redeploy elsewhere (see §7.4).

Auto-prefixed internal names

Set an **App Code** once and every internal name gets the prefix automatically. Eliminates the most common SharePoint deployment bug – collisions and typos in `InternalName` – before they reach production.

Sample data & CSV / Excel import

Paste 200 rows from Excel, point at a CSV, or type directly into the data grid. Data is included in PowerShell and Power Automate output, so dev environments come up usable on day one.

Templates & undo / redo

Load built-in templates (E-commerce, Project Management, CRM, Inventory) or save your own. 50-step undo / redo means you can refactor without fear.

CHAPTER 2

Getting Started

> In the next two pages you'll go from an empty canvas to a working PnP PowerShell script – about ninety seconds, no install required. The first section maps out the interface; the second walks the seven clicks that make a list real.

2.1 Interface Overview

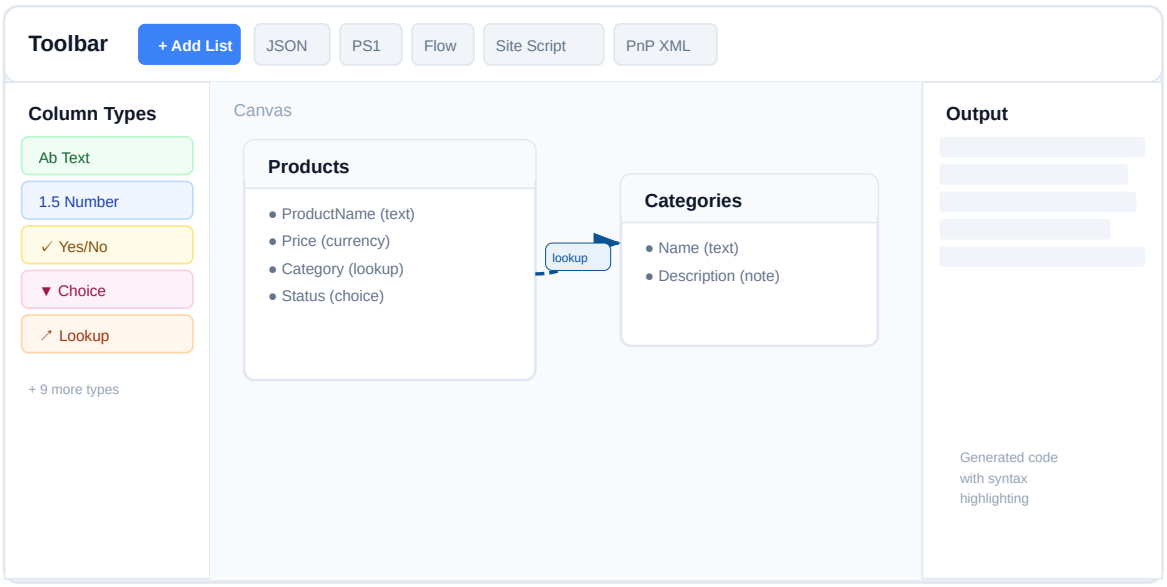


Figure 1 – Application layout: toolbar, sidebar, canvas, and output panel

The interface is divided into four main areas:

Area	Location	Purpose
Toolbar	Top bar	App Code field, Add List button, output format buttons, search box, undo/redo
Field Palette	Left sidebar	14 draggable field types (Text, Number, Choice, Lookup, etc.)
Canvas	Center	Visual workspace with draggable list cards and relationship lines
Output Sidebar	Right (on demand)	Generated code with syntax highlighting, copy and download buttons

When you first open the application, the canvas shows an empty state with quick-start buttons and tips.

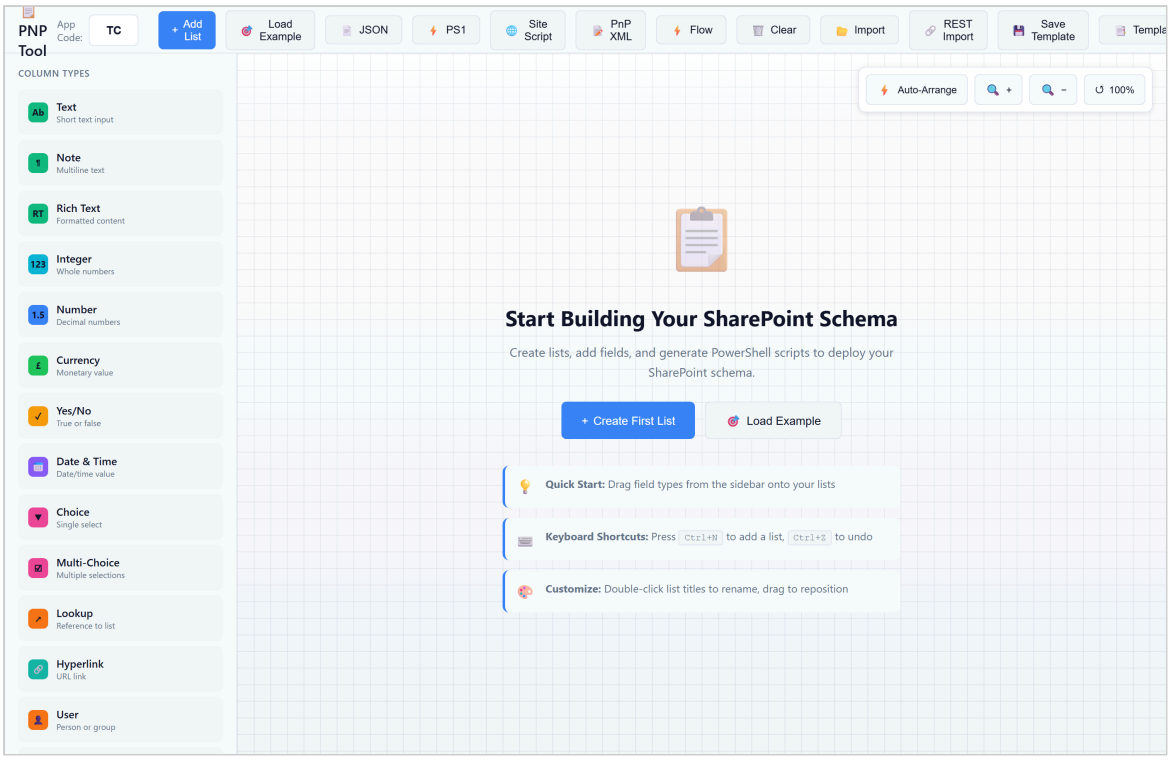


Figure 2 – The empty canvas with quick-start buttons and tips

Click **Load Example** to populate the canvas with a complete 7-list e-commerce schema, including lookup relationships, content types, views, and sample data.

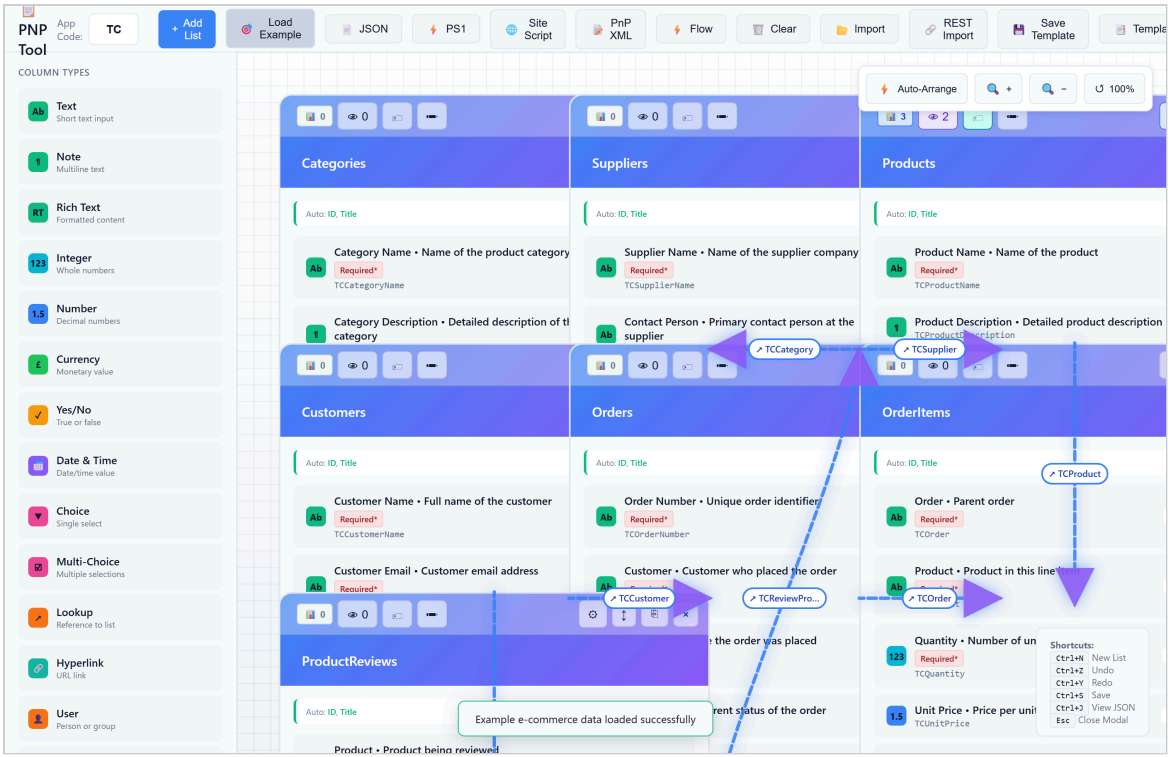


Figure 3 – The canvas with the full e-commerce example loaded

2.2 Your First List

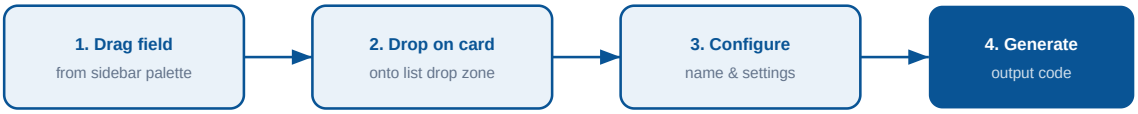


Figure 4 – The drag-and-drop workflow

1. Enter an **App Code** in the top-left input (e.g. `TC`). It is prepended to every internal field name. The input accepts up to two characters and is auto-uppercased – type `my`, you'll see `MY`.
2. Click **Add List** on the toolbar and choose **Generic List** or a library type.
3. Double-click the list title on the card and type a name (e.g. “Products”).
4. Drag a field type from the left palette (e.g. “Text”) and drop it onto the list card.
5. Name the field in the dialog that appears (e.g. “Product Name”). The internal name is auto-generated with the App Code prefix.
6. Click the **gear icon** on the field for advanced settings – form visibility, validation, custom formatting.
7. Click a format button on the toolbar (**JSON**, **PS1**, **Flow**, etc.) to generate output.

> **That's the whole loop.** Every time you add a field, rename a list, or change a validation formula, click **PS1** again – the entire deployment script regenerates. No diffing, no patching, no forgotten field.

> **Shortcuts:** `Ctrl+N` adds a list, `Ctrl+J` shows the JSON output, `Shift+?` opens the full shortcut list. Work is auto-saved to your browser every second – `Ctrl+S` just makes the save explicit.

CHAPTER 3

Designing Lists

> Each list on the canvas represents a SharePoint list. Fields are added by dragging from the palette, and every property can be configured through settings panels.
>
> *A note on terminology:* SharePoint's UI uses both **field** and **column** for the same thing. We use them interchangeably; you'll see **Add Column** on the dialog and **Field Settings** on the gear-icon panel because that's what the underlying tool labels them.

3.1 Supported Field Types

Every field type carries its SharePoint type, internal-name prefix, validation rules, form visibility, and column-formatting JSON together – so when you regenerate the deployment script, none of those settings ever fall off the back of the truck.

Type	SP Type	Key Options
Text	Text	Max length (255 default)
Note	Note	Number of lines, append-only
Rich Text	Note (RichText)	Full HTML editing
Integer	Integer	None
Number	Number	Min, max, decimals, percentage
Currency	Currency	Min, max, decimals
Yes/No	Boolean	Default value
Date & Time	DateTime	Date only or date+time, friendly format
Choice	Choice	Options list, fill-in choice
Multi-Choice	MultiChoice	Options list, fill-in choice
Lookup	Lookup	Target list, multi-value (display column is always Title)
Hyperlink	URL	None
Person	User	Multi-value, selection mode/group
Geolocation	Geolocation	GPS coordinates (limited support in modern lists)

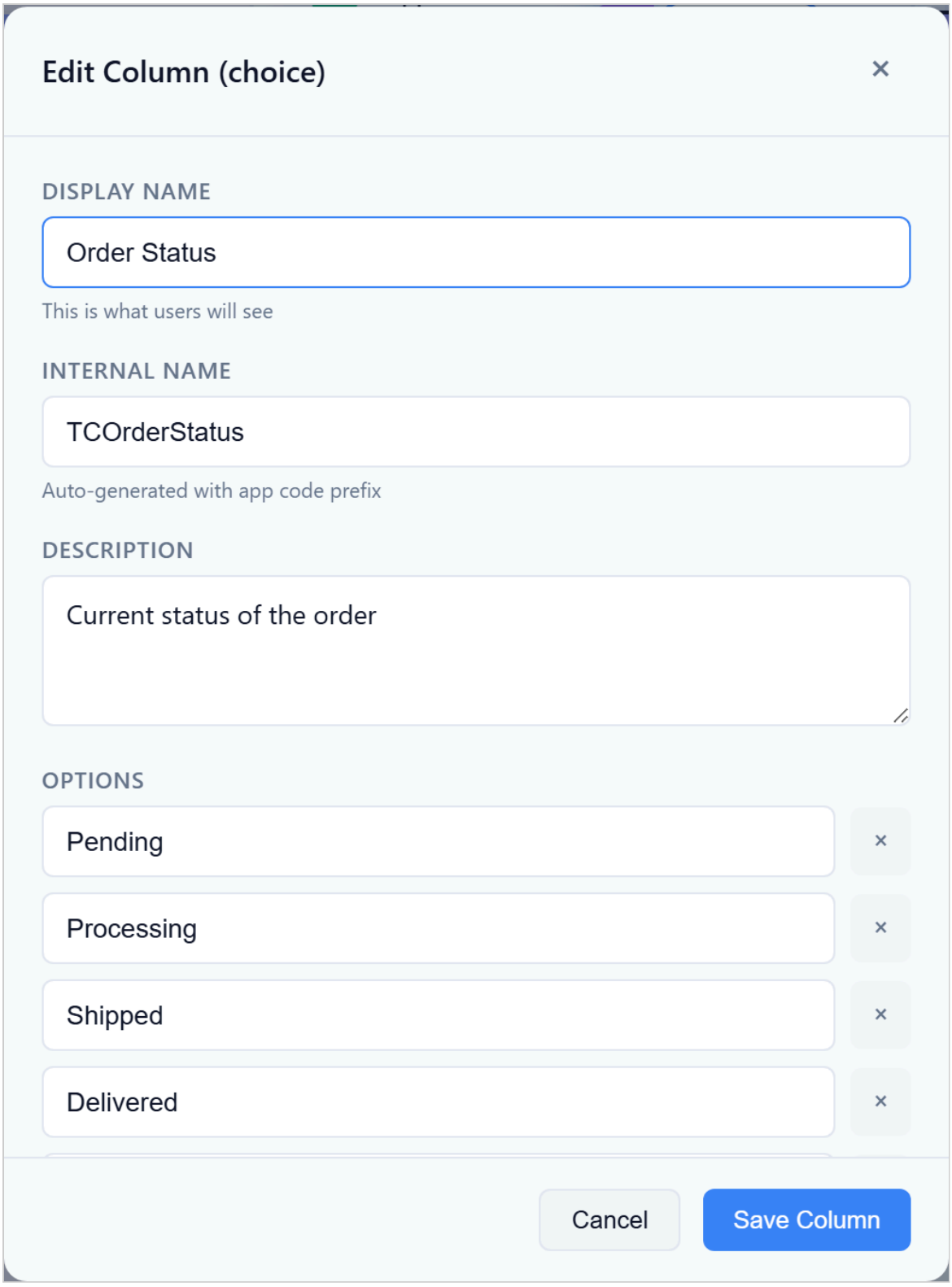


Figure 5 – The Add Column dialog showing a Choice field with options

3.2 List Card Anatomy

Each list card displays the list name, toolbar buttons, fields, and views. Below is the Products card from the e-commerce example:

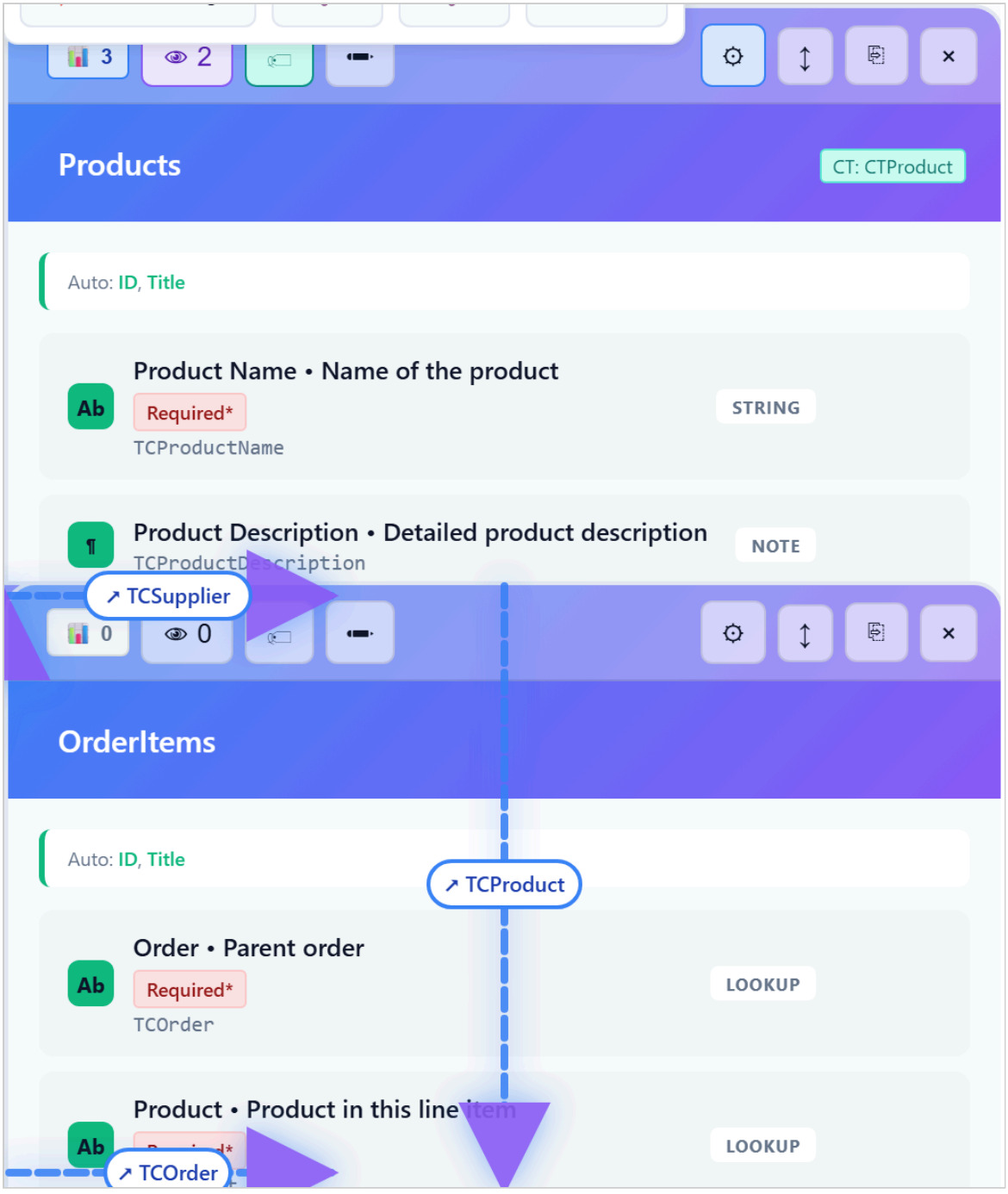


Figure 6 – A list card showing toolbar, fields with type badges, and views

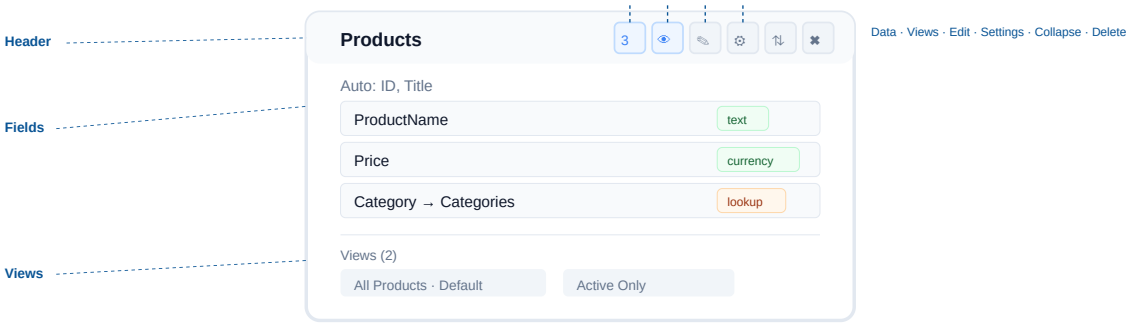


Figure 7 – Card anatomy: header toolbar, field rows with type badges, and views section

3.3 Field Settings

Click the **gear icon** on any field to open its advanced settings – form visibility, validation formulas, JSON column formatting, unique constraints, and every type-specific option (Number of Lines for Note, Allow Multiple for Lookup, Min/Max for Number, Selection Mode for User, etc.). All of these travel with the field into every generated output. In raw PnP PowerShell each one would require a separate cmdlet call after the field is created; here they are applied in the right order automatically.

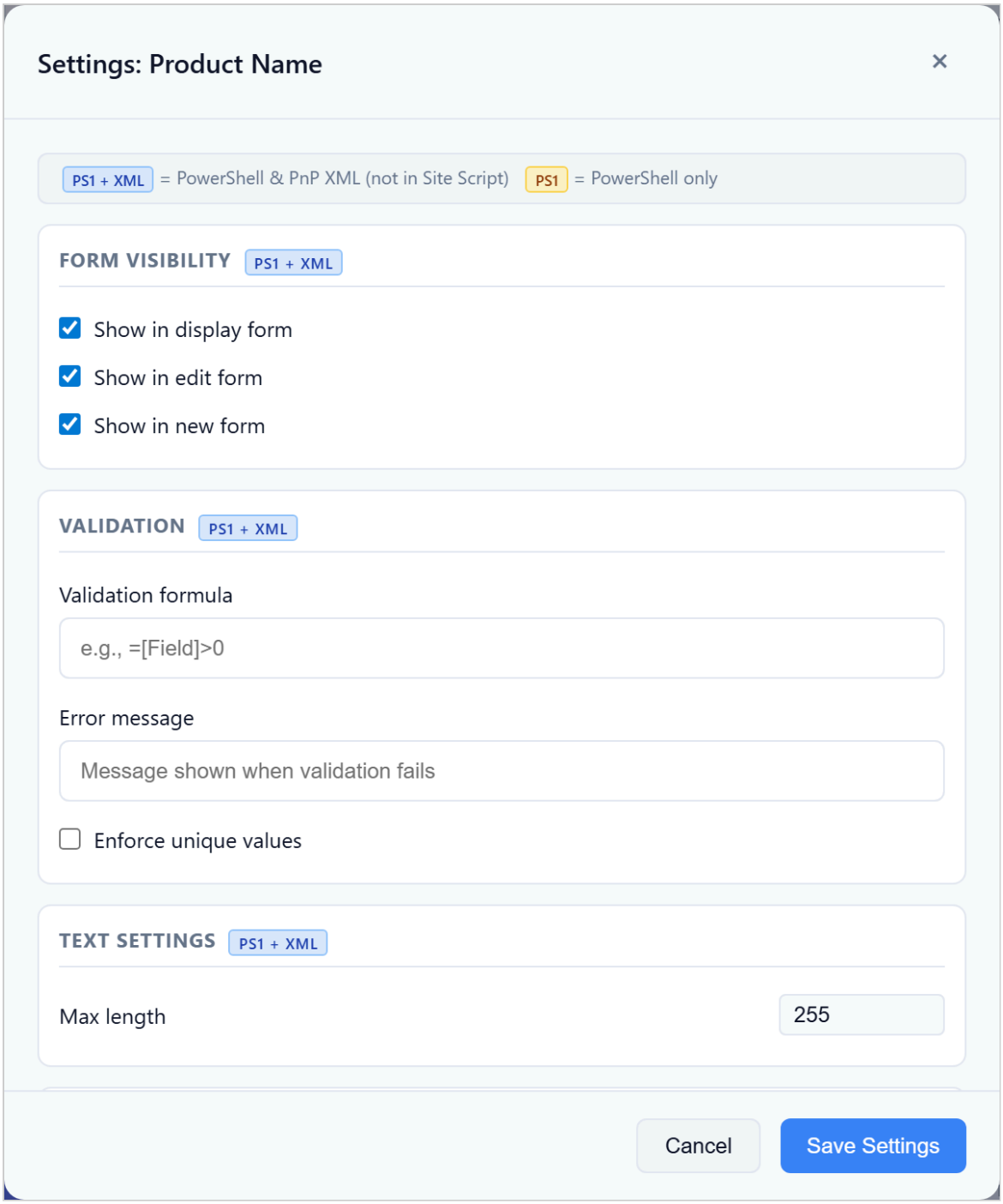


Figure 8 – Advanced field settings: form visibility, validation, and formatting

Setting	Description	Output
Show in Display Form	Show this field when viewing an item	PS1 + XML
Show in Edit Form	Show this field when editing	PS1 + XML
Show in New Form	Show this field when creating new items	PS1 + XML
Custom Formatter	JSON column formatting definition	PS1
Validation Formula	Formula to validate field values	PS1 + XML
Validation Message	Message shown when validation fails	PS1 + XML
Enforce Unique	Prevent duplicate values	PS1 + XML

> **Tip:** The Field Settings panel is where you'll spend most of your tuning time. Custom Formatter accepts the same JSON SharePoint expects under *Format this column* – paste an existing formatter and it ships with the field in every regeneration, instead of living in a Gist that has to be reapplied per tenant.

3.4 Lookup Relationships

Lookup fields create a link between two lists. The canvas draws the relationship line so you can see, before you deploy, which list points where – the bug-hunt that normally only surfaces after `Add-PnPField -Type Lookup` fails because the target list was created in the wrong order. PnP List Generator also orders list creation in the generated PowerShell so lookup targets always exist first.

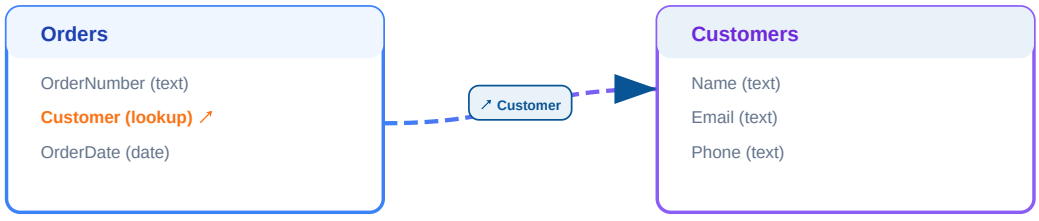


Figure 9 – Lookup relationship: Orders.Customer points to Customers list

1. Drag the **Lookup** field type onto the source list.
2. In the Add Column dialog, pick the **Target List** from the dropdown. The displayed column is the target list's **Title**.
3. Optionally open Field Settings (gear icon) and enable **Allow Multiple Values**.

The canvas draws a connecting line between the two lists when the lookup is set. Hovering over the line highlights both cards.

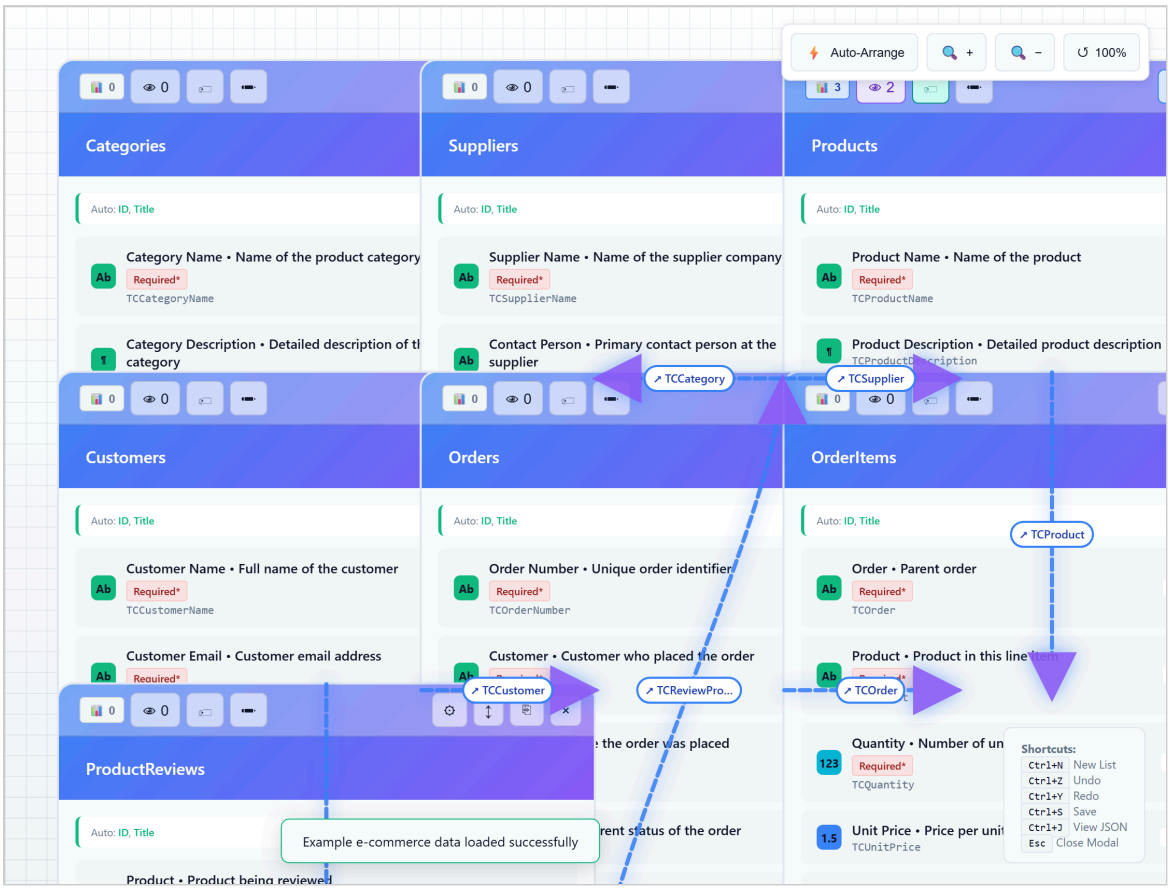


Figure 10 – Canvas showing multiple lookup relationships between lists

3.5 Canvas Controls & Context Menu

Action	How
Move a list	Drag the list card header
Zoom in / out	Click the zoom buttons in the canvas toolbar
Auto-arrange	Click **Auto-Arrange** in the canvas toolbar
Collapse a list	Click the ⇅ button on the card header
Edit fields in bulk	Click the pencil icon on a card – opens a tabular editor for all fields at once
Re-import a saved JSON	**Import** in the toolbar – paste a previously exported JSON to round-trip a design
Wipe the canvas	**Clear** in the toolbar (asks for confirmation)

Right-click any list card to open its context menu:

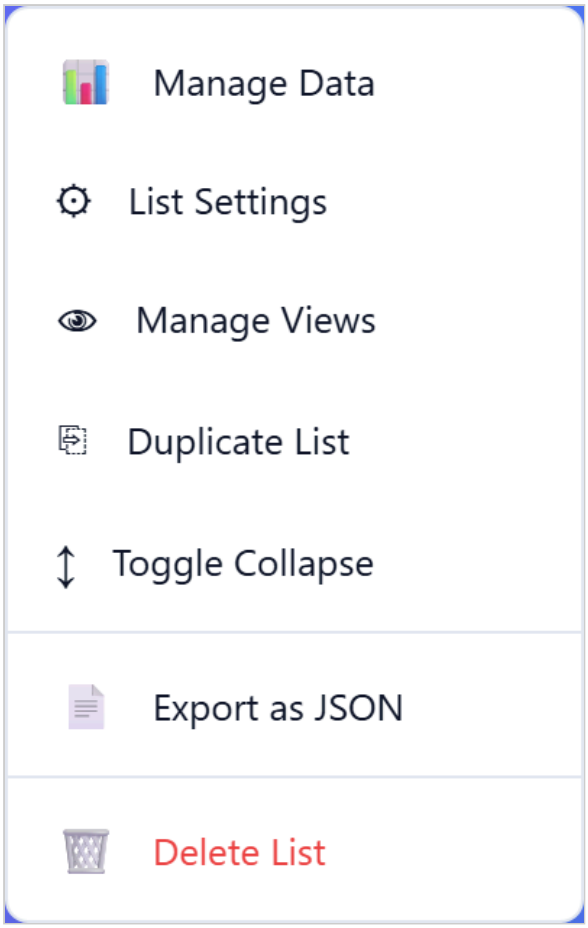


Figure 11 – Right-click context menu on a list card

Menu item	What it does
Manage Data	Open the data grid for this list
List Settings	Configure versioning, security, content types
Manage Views	Add, edit, or remove list views
Duplicate List	Clone the list with all its fields and settings
Export as JSON	Download this single list as a JSON file
Delete List	Remove the list (asks for confirmation)

> **Tip:** The canvas auto-saves to your browser's local storage every second – close the tab, reopen later, work continues. **Ctrl+Z** / **Ctrl+Y** support 50 undo steps, so you can refactor without fear.

CHAPTER 4

List Settings & Content Types

> Each list can be configured with SharePoint settings for versioning, security, content types, and more. Click the **gear icon** on the card header or right-click → **List Settings**.

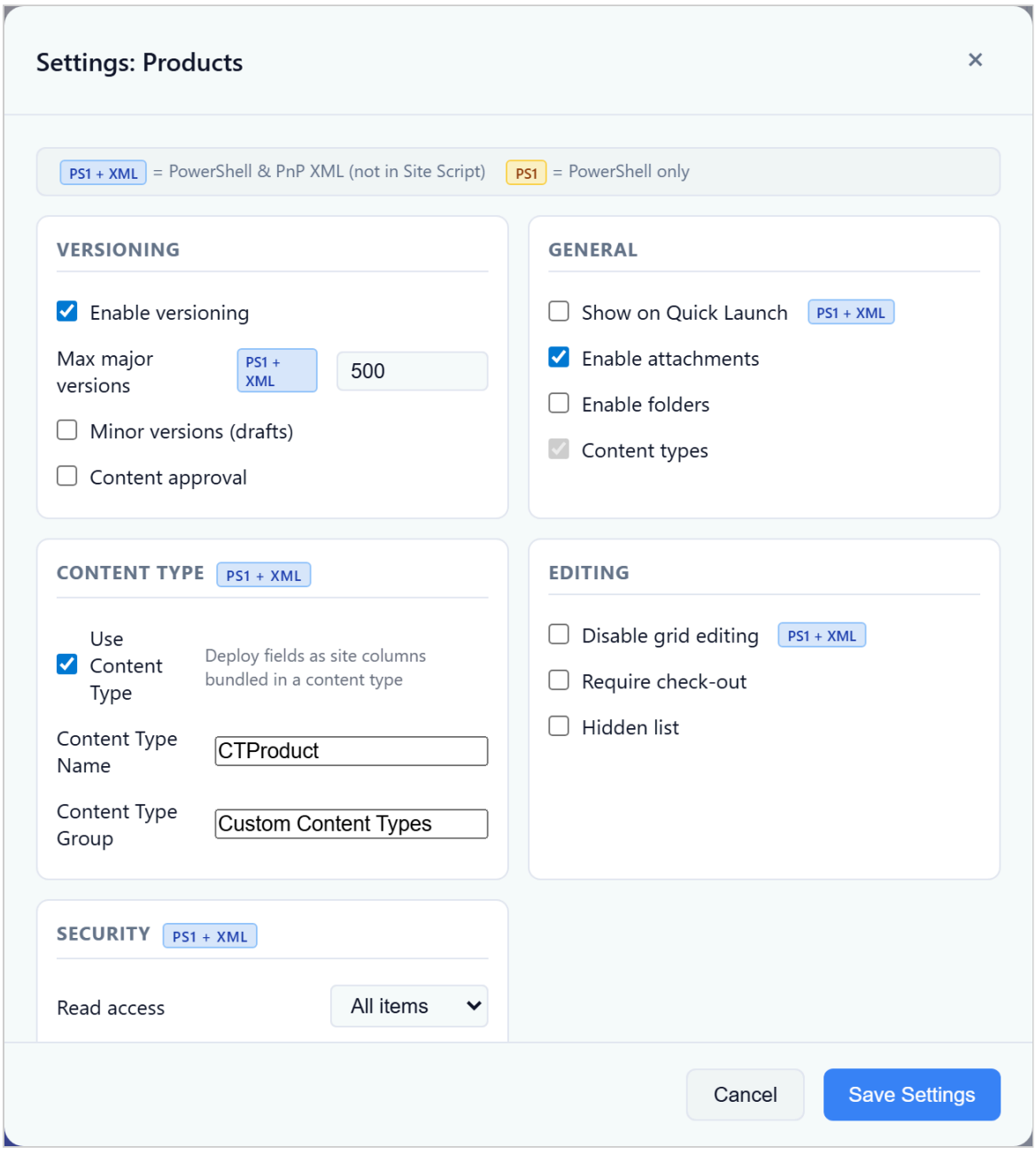


Figure 12 – List Settings dialog with versioning, content type, editing, and security options

Setting	Description	Output
Enable Versioning	Track changes with version history	All
Max major versions	Maximum number of major versions retained	PS1 + XML
Minor Versions	Track draft versions (0.1, 0.2, etc.)	All
Content Approval	Require approval before items become visible to readers	All
Show on Quick Launch	Add the list to the site's left navigation	PS1 + XML
Attachments	Allow file attachments on items	PS1 + XML
Folders	Allow creating folders in the list	All
Read / Write Security	Limit users to seeing or editing only items they created	PS1 + XML
Experience	Force Classic, force Modern, or let SharePoint decide	PS1 + XML
Disable Grid Editing	Prevent inline Quick Edit	PS1 + XML
Require check-out	Require check-out before editing	All
Hidden	Hide from site navigation and contents	All

4.1 Content Types

Content types deploy fields as reusable **site columns** bundled in a content type, rather than as list-scoped columns – the recommended approach when the same columns are reused across multiple lists or sites.

In raw PnP PowerShell, switching a list from list-scoped fields to a content-type-based deployment means rewriting the script: remove the list-field cmdlets, add `Add-PnPField` at site scope, then `New-PnPContentType`, `Add-PnPFieldToContentType`, `Add-PnPContentTypeToList`, in that exact order. Tick one box here and the generator does it for you, and re-does it on every regeneration.

1. Open **List Settings** for the list.
2. In the **General** section, tick **Content Types** – this exposes the Content Type section.
3. In the **Content Type** section, tick **Use Content Type**.
4. Optionally set a **Content Type Name** (defaults to “CT{ListName}”) and **Group**.
5. Click **Save Settings**.

The generated output will then:

- Create site columns at the site level
- Create a content type and add the site columns to it
- Bind the content type to the list

> **Important:** Content type binding changes how SharePoint validates new items. When using Power Automate output, data insertion is automatically ordered *before* CT binding to avoid validation errors.

> **Note:** The `PS1 + XML` badge in the List Settings dialog indicates which settings are included in PowerShell and PnP XML output but not in Site Script. Settings without a badge are included in all output formats.

CHAPTER 5

Views & Aggregations

> Views control how list data is displayed. Each list can have multiple views with different columns, filters, and aggregations.

Click the **views badge** on a card header or right-click → **Manage Views**.

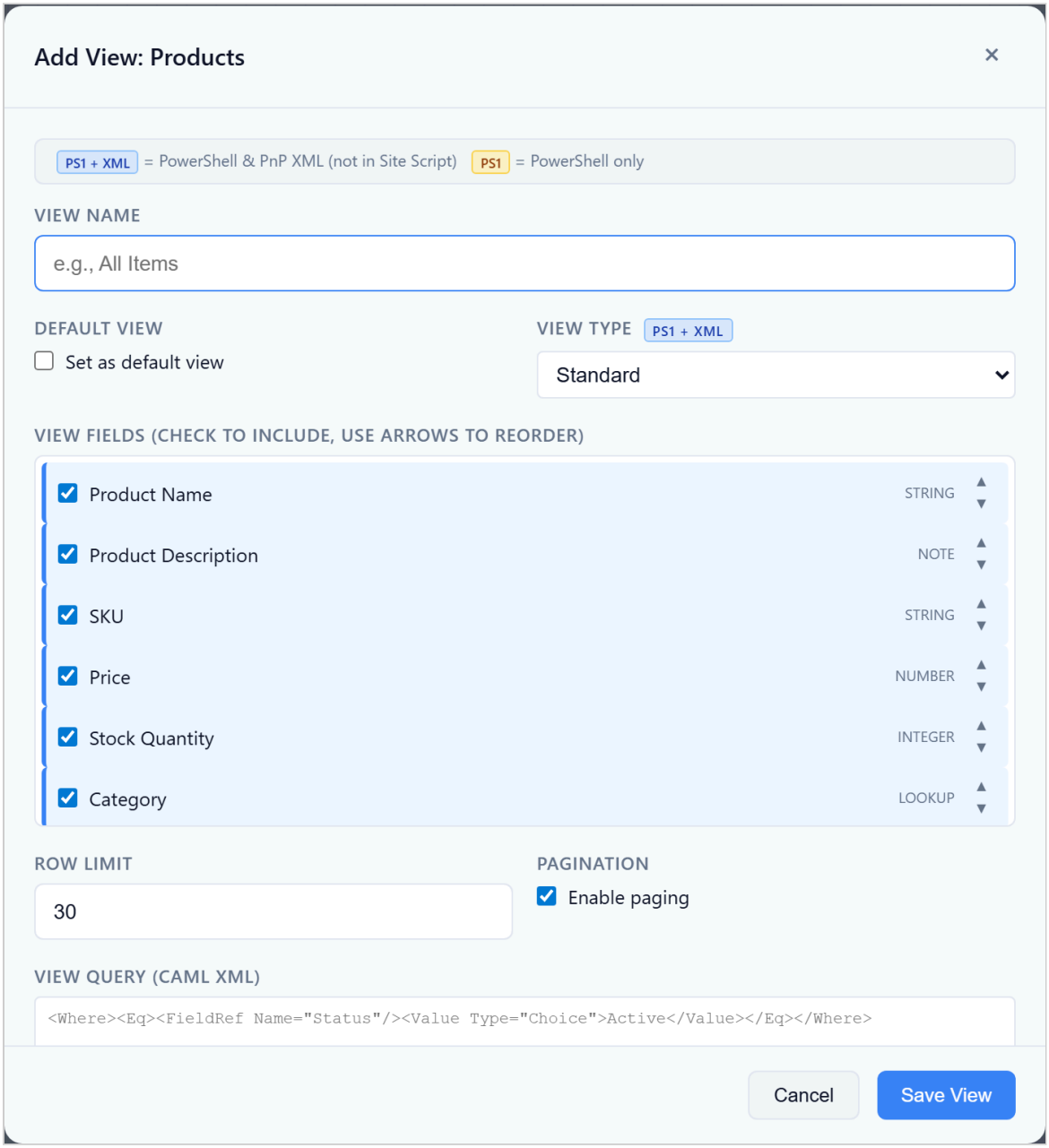


Figure 13 – The View Editor with field selection, type, and pagination settings

1. Click **+ Add View** to create a new view.
2. Set the **view name**, **type** (Standard, Datasheet, or Calendar), and optionally mark it as the **default view**.
3. Select which **fields** to include using the checkboxes. Use the arrows to reorder.
4. Set **Row Limit** and enable **Paging** for large lists.
5. Add a **CAML Query** to filter or sort items (optional).
6. Set **Scope** (Default, RecursiveAll, or FilesOnly).

5.1 Aggregations & JSON Formatting

Aggregations display summary values (totals, averages, counts) at the bottom of a view column. Add them in the View Editor under the **Aggregations** section.

Function	Description	Applies to
COUNT	Number of items	Any column
SUM	Total of all values	Number, Currency
AVG	Average value	Number, Currency
MIN	Smallest value	Number, Currency, Date
MAX	Largest value	Number, Currency, Date
STD	Standard deviation	Number, Currency
VAR	Variance	Number, Currency

Custom JSON Formatting

Paste a SharePoint List Formatting JSON definition into the **View Formatting JSON** field in the view editor – same JSON SharePoint expects under *Format current view*. (This is the view-level equivalent of the field-level **Custom Formatter** in §3.3.) View-formatting JSON normally lives in a Gist somewhere and gets pasted into the modern UI by hand on every tenant; store it once here and it ships with every regenerated PowerShell script.

> **Note:** Aggregations are included in **PS1 + XML** output. View formatting JSON is included in **PS1** output only.

CHAPTER 6

Managing Data

> Sample data travels with the schema into PowerShell, PnP XML, and Power Automate output. Use it to ship demo content to a customer tenant, seed lookup target lists so a dev environment is usable on day one, or build a 200-row test dataset once and re-deploy it every time you blow away the dev site.

Click the **data badge** on a card header (shows the row count) or right-click → **Manage Data**.

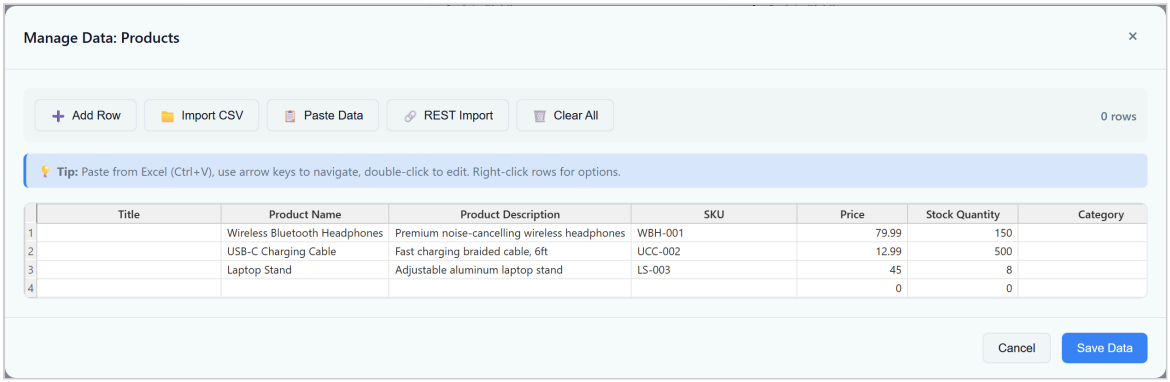


Figure 14 – The data grid with sample rows and import tools

Data Grid

The grid displays all fields as columns. Click a cell to edit its value. The toolbar provides buttons to add rows, import CSV, paste data, import from REST API, and clear all data.

- **Choice** fields appear as dropdown selectors.
- **Multi-Choice** fields accept semicolon-separated values (e.g. `Red;Blue`).
- **Boolean** fields appear as checkboxes.
- **Number / Currency** fields appear as number inputs.
- **Lookup** fields take the target item's numeric ID (not its display text).

6.1 CSV & Excel Import

CSV Import

1. Click **Import CSV** in the data grid toolbar.
2. Select a CSV file. A preview appears with detected rows and columns.
3. Toggle **First row is header** if needed.
4. Map CSV columns to list fields using the dropdowns.
5. Review the data preview (first 5 rows), then click **Import Data**.

Paste from Excel

Copy a range of cells from Excel (including headers), then click **Paste Data** in the data grid toolbar. PnP List Generator matches column headers to field display names automatically.

You can also use `Ctrl+V` while the data grid is open. Use `Tab`, `Enter`, and arrow keys to navigate between cells.

REST API Import

Click **REST Import** in the data grid to import data from a live SharePoint list. This uses the same REST API workflow described in §7.4.

> **Tip:** Data rows are included in PowerShell (`Add-PnPListItem`) and Power Automate (as list item creation actions) output. JSON export always includes data. Site Script output does not support data.

> **Note:** Lookup field values in data rows should use the numeric item ID of the target item, not the display text.

CHAPTER 7

Generating Output

> Click an output format button in the toolbar to generate deployment code. The output sidebar opens with syntax-highlighted code plus copy and download buttons.
>
> The same canvas powers all five generators. Add a column on Monday, regenerate the PowerShell on Tuesday, regenerate the Site Script for the hub-site team on Wednesday – they stay in sync because they come from the same source of truth.

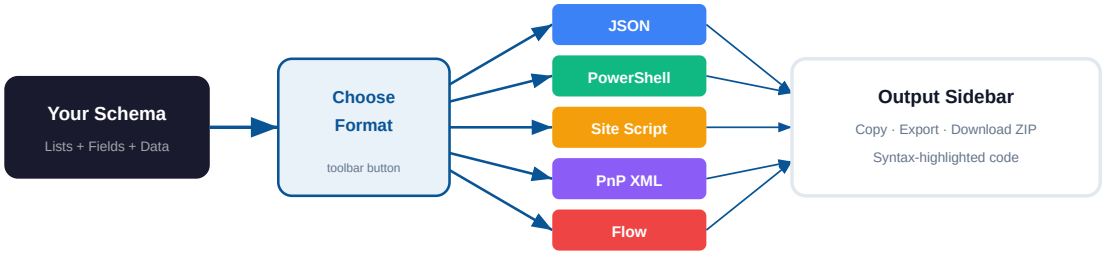


Figure 15 – Output generation workflow

7.1 Output Formats

Five buttons, five different jobs. Pick the one that matches who is going to deploy:

Format	Pick this when…
JSON	You want to back up a design, hand it to a colleague, or version-control it in Git alongside the rest of the project. Re-importable via the toolbar's **Import** button.
PnP PowerShell	You have tenant-admin rights and want a re-runnable script – the developer / consultant default.
Site Script	You need the schema applied as a Site Design from the SharePoint Admin Center, so site owners can self-serve provisioning without scripts.
PnP XML	You're publishing a Provisioning Template to a corporate template gallery, or applying schemas via `Apply-PnPProvisioningTemplate` in CI/CD.
Power Automate	The person deploying has no PowerShell, no admin rights, and no patience – they import a ZIP, hit Run, and the lists appear.

Each format has its own page in §7.1.1–7.1.5 below; the full feature matrix is in §7.2.

7.1.1 JSON Output

The JSON format exports the complete schema – lists, fields, settings, views, and data – in a single file. Use it to back up a design, version-control it in Git, hand it to a colleague, or load it back into PnP List Generator via the toolbar's **Import** button to round-trip a design.

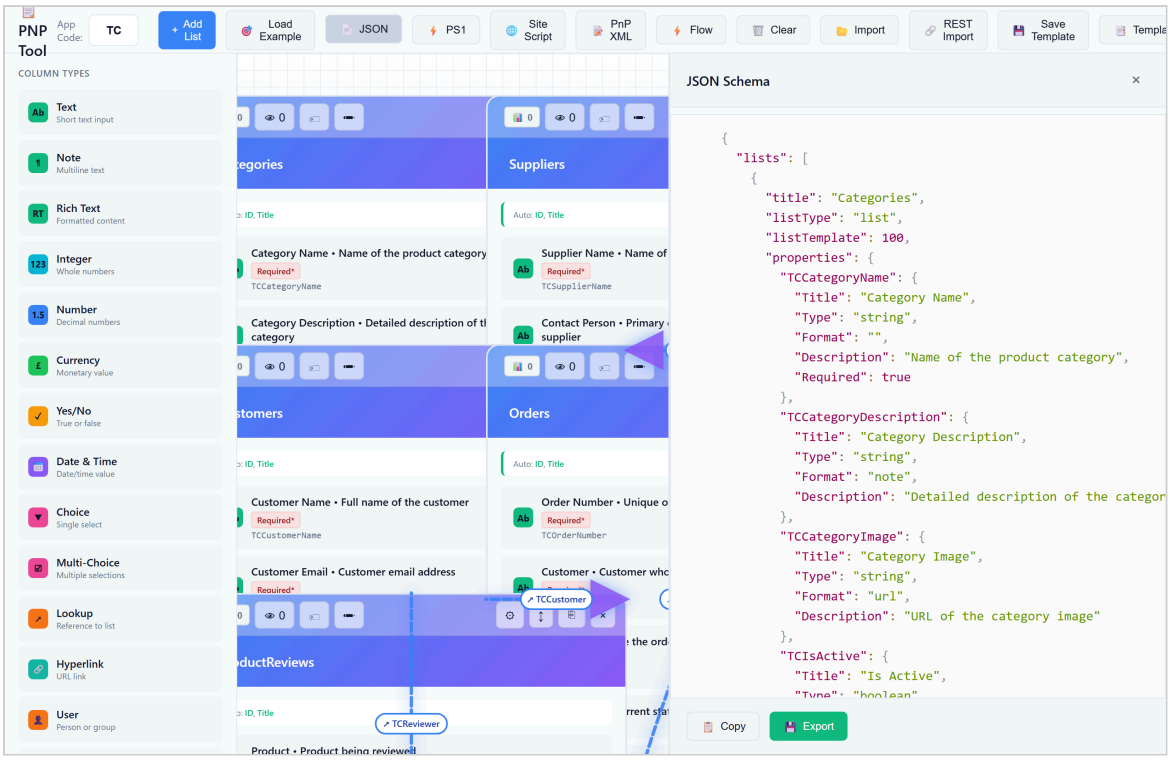


Figure 16 – JSON schema output with syntax highlighting

7.1.2 PowerShell Output

Generates a **PnP PowerShell** script (`.ps1`) that provisions every list, field, content type, view, and data row. The script uses `Connect-PnPOnline`, `New-PnPList`, `Add-PnPField`, `Add-PnPListItem`, and related cmdlets, and orders list creation so lookup targets always exist before the lists that reference them.

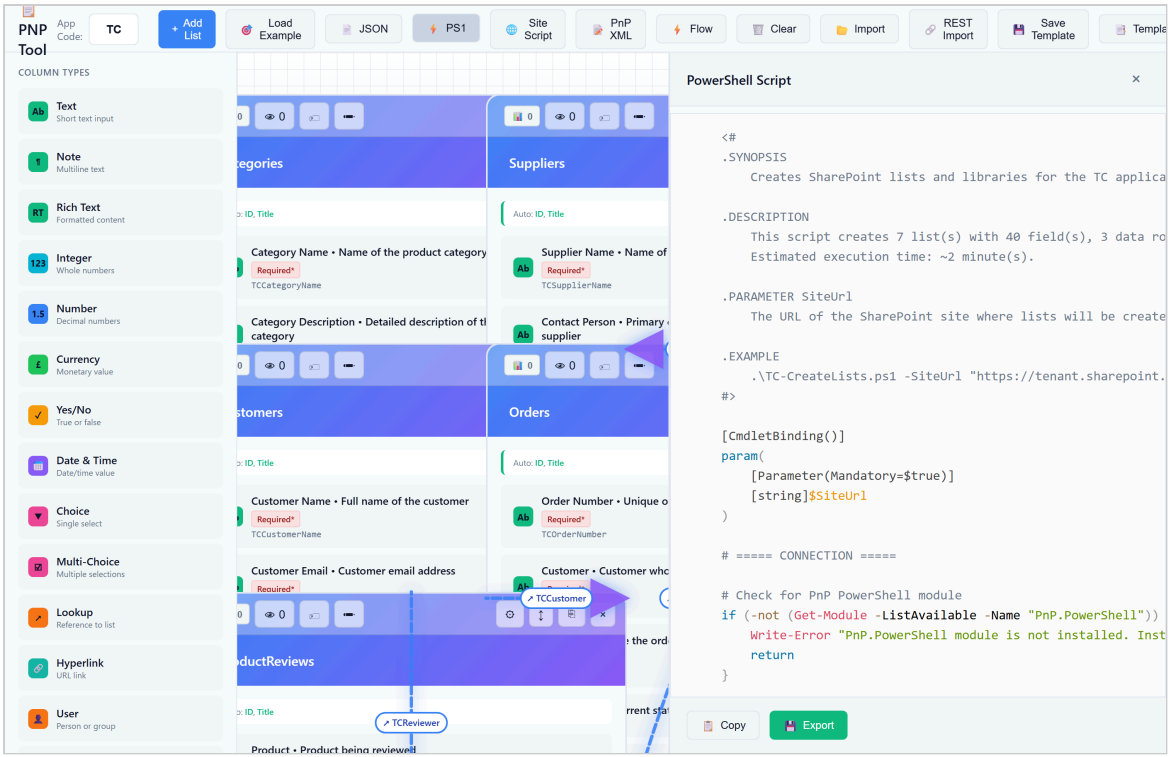


Figure 17 – PnP PowerShell deployment script

7.1.3 Site Script Output

Generates a SharePoint **Site Script** JSON. SharePoint caps a single Site Script at 20 actions; PnP List Generator auto-splits larger schemas into Part A, Part B, etc. and offers a **Download All** button – so the 20-action ceiling stops being a problem you have to solve by hand.

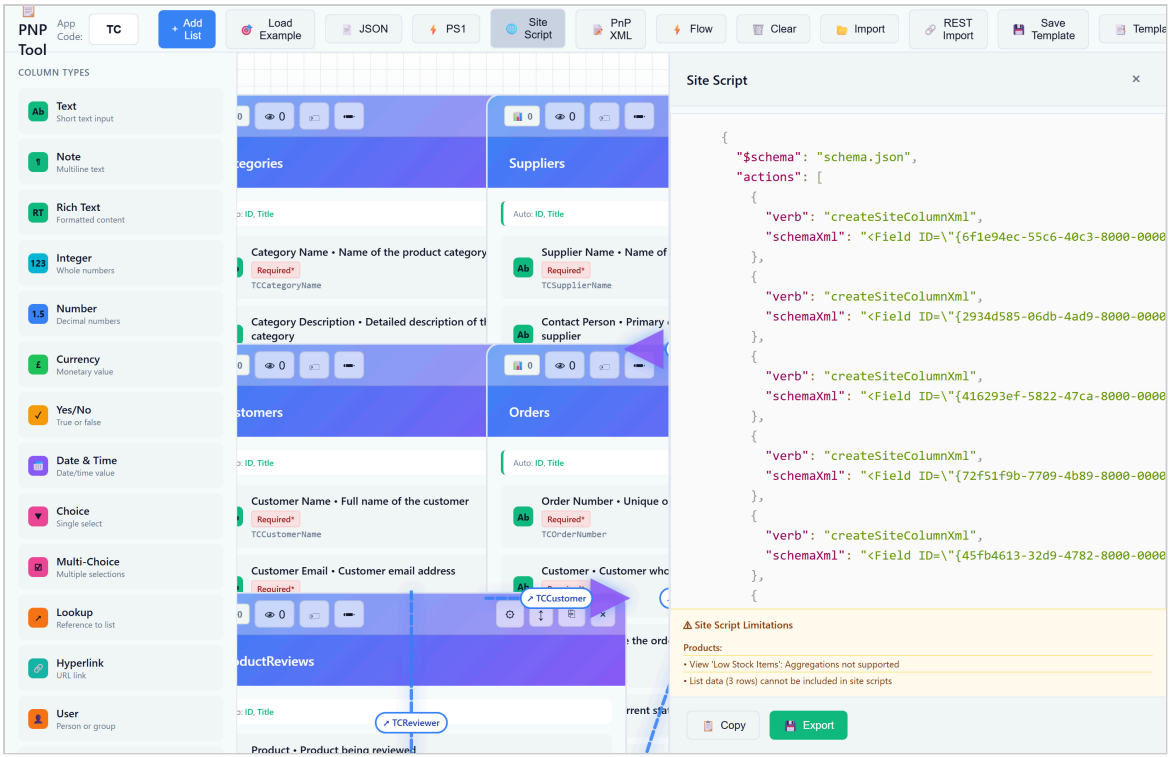


Figure 18 – Site Script output sidebar

7.1.4 Power Automate Output

Generates a **Power Automate cloud flow** definition that provisions lists and fields via the SharePoint REST API. Includes both a **JSON preview** for inspection and a downloadable **solution ZIP** for one-click import. The solution name is derived from your App Code (e.g. App Code `TC` → solution `TCCreateSharePointLists`).

7.1.5 PnP Provisioning XML Output

Generates a **PnP Provisioning Template** (`.xml`) covering every list, content type, site column, view, and data row. Apply with `Apply-PnPProvisioningTemplate -Path schema.xml`, or upload to a corporate template gallery so site owners can self-apply the schema. Carries the most settings of any format – security, versioning, and field-level options that Site Script and Flow can't represent.

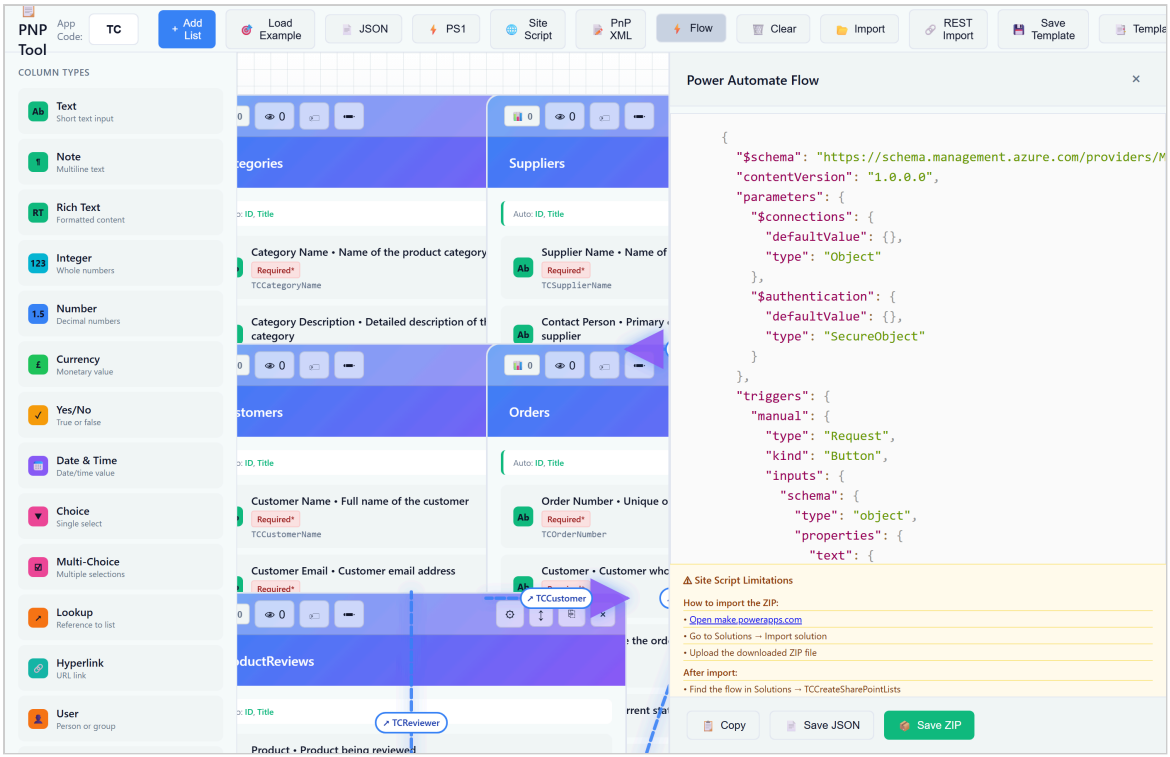


Figure 19 – Power Automate flow with Save JSON and Save ZIP buttons

7.2 Feature Matrix

What each generator can and cannot include in its output:

Feature	JSON	PowerShell	Site Script	PnP XML	Flow
Lists & fields	✓	✓	✓	✓	✓
Content types	✓	✓	Site columns only	✓	✓
Views	✓	✓	✓	✓	✓
Sample data	✓	✓	–	✓	✓
List settings	✓	✓	Partial	✓	Partial
Field settings	✓	✓	–	✓	Partial
Security settings	✓	✓	–	✓	–
Downloadable file	.json	.ps1	.json	.xml	.json or .zip

> **Note:** “Partial” for Power Automate field settings means unique constraints, indexing, and user-selection mode are emitted; full validation formulas and column formatting are not.

7.3 Power Automate ZIP Import



Figure 20 – Power Automate solution import workflow

1. Click the **Flow** button in the toolbar to generate the flow.
2. Click **Save ZIP** in the output sidebar.
3. Open **make.powerapps.com** and navigate to **Solutions**.
4. Click **Import solution** and upload the downloaded ZIP file.
5. On the connection mapping screen, sign in with your SharePoint account.
6. After import, find the flow in the solution and click **Edit**.
7. Verify the SharePoint connection is assigned, click **Save**, then click **Turn on**.
8. Run the flow manually and enter your SharePoint site URL when prompted.

> **Important:** Do *not* use **My Flows** → **Import** – that route only handles the older package format and will fail for the solution ZIP. Always use **Solutions** → **Import solution**.

> **Note:** The solution name is derived from your App Code (App Code `TC` → solution `TCCreateSharePointLists`). The flow is imported in the **Off** state – turn it on after the connection is mapped.

7.4 REST API Import

Inherited a tenant with thirty lists nobody documented? Migrating a legacy site to a new tenant? Point PnP List Generator's REST Import at the site, paste the JSON response, and every list – fields, types, lookups – lands on the canvas. From there you can document it, redeploy a copy to staging, or convert it into a Site Script that other site owners can self-apply.

The workflow has three stages – generate the URL, fetch the JSON, paste it back – broken down step-by-step below.



Figure 23 – REST API import workflow

1. **Sign in to SharePoint** in your browser first – the URL you paste in step 3 will use your existing session cookies to authenticate.
2. Click **REST Import** on the toolbar and enter your SharePoint site URL.
3. PnP List Generator generates the REST API endpoint URL. **Copy** it and paste it into your browser's address bar.
4. The browser returns a JSON response. Select all (`Ctrl+A`) and copy (`Ctrl+C`).
5. Paste the JSON back into PnP List Generator. A preview shows the detected lists and fields.
6. Click **Import Lists** to add them to the canvas.

> **Tip:** REST API import works for both list schemas (field definitions) and list data (item rows). Use the toolbar **REST Import** button for schemas, or the data grid **REST Import** button for data.

CHAPTER 8

Templates & Shortcuts

> Templates turn the second project of the same kind into a thirty-second job. Save your standard "project workspace" design once; load it on every new client tenant. Keyboard shortcuts trim the rest.

Built-in Templates

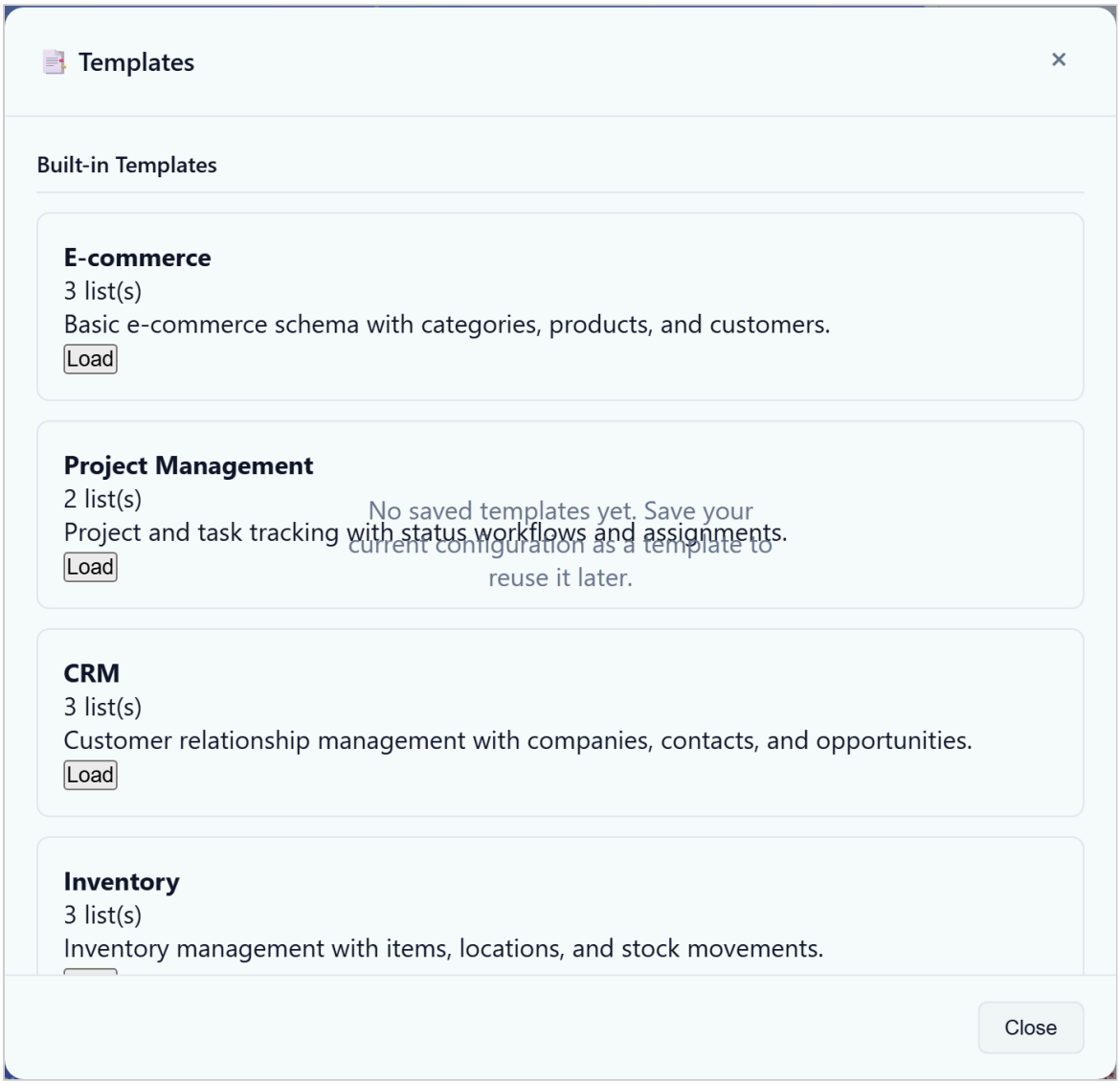


Figure 21 – Template browser with built-in and user-saved templates

Template	Lists	Description
E-commerce (Full Example)	7	Products, Categories, Suppliers, Orders, Order Items, Customers, Reviews – with content types, views, sample data
E-commerce	3	Products, Categories, Suppliers with lookups
Project Management	2	Projects and Tasks with status tracking
CRM	3	Contacts, Companies, Deals pipeline
Inventory	3	Items, Warehouses, Stock movements

The **Load Example** toolbar button is a shortcut to the *E-commerce (Full Example)* template above – useful as a guided tour of every feature in five seconds.

Click **Save Template** to store your current design. Saved templates live in your browser's local storage and appear under **My Templates**. Clearing browser data removes them, and they don't sync across browsers – for portable backups, use **JSON** export instead (see §7.1.1).

Keyboard Shortcuts

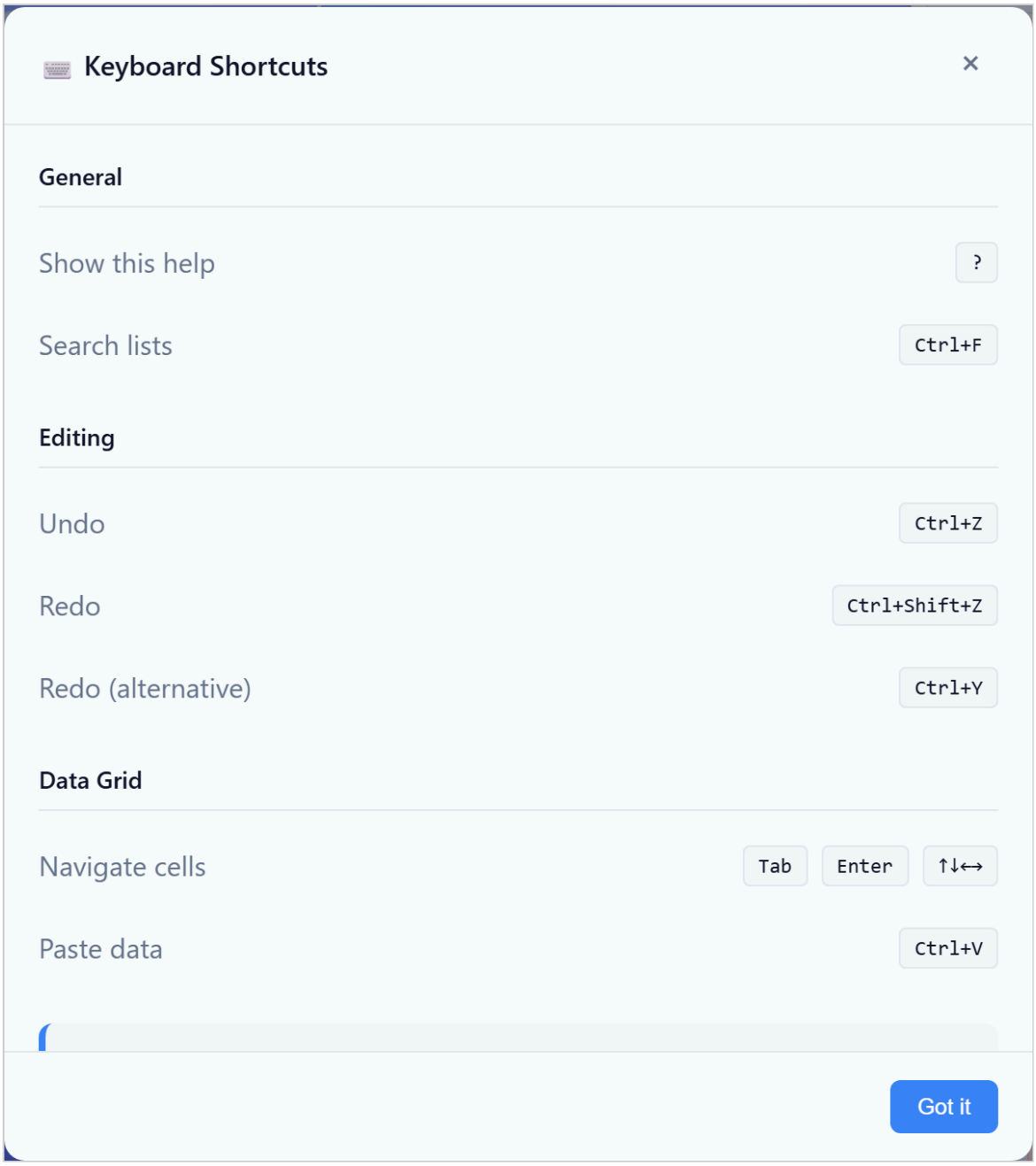


Figure 22 – Keyboard shortcuts reference

Shortcut	Action
`Ctrl+N`	Add a new list
`Ctrl+Z`	Undo (50-step history)
`Ctrl+Y` / `Ctrl+Shift+Z`	Redo
`Ctrl+J`	View JSON schema
`Ctrl+S`	Save (the canvas also auto-saves every second)
`Shift+?`	Show keyboard shortcuts
`Escape`	Close any modal or sidebar
`Tab` / `Enter` / Arrows	Navigate data grid cells
`Ctrl+V`	Paste data into grid