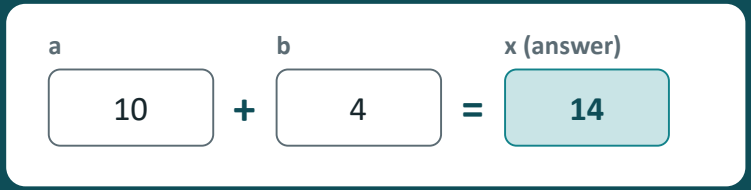
Formulas in Table Layouts

How a + b = x becomes one smart cell: the logic, the UI, and Insert vs Copy & paste



Form Builder · Design note · November 2026

The idea in one slide

A table section with a formula column

Item	Formula: a + b = x
Batch 1	a: 10 + b: 4 = x (answer): 14
Batch 2	a: 7 + b: 5 = x (answer): 12

3 fields, shown to the user as 1 cell

1

3 fields, 1 cell

a and b are number boxes, x is the answer. All three sit inside one table cell.

2

Live answer

x recalculates as the user types and cannot be edited.

3

Two ways to add it

Insert (click) or Copy & paste with an in-app clipboard.

Where we are today

Section layout	Add a formula	Live answer in preview
Inline, normal editor	**Works**	**Works**
Inline with an HTML table	**Blocked**	Only if the fields already exist
Table layout (drag-drop / CSV)	**Blocked**	Row 1 only
Grid	**Blocked**	Not applicable

Blocker 1: reload drops fields

A table is rebuilt from its CSV template on reload. A field with no cell of its own is lost.

Blocker 2: Add Row reads row 1

Copied rows rename fields to field_X_row_2, but the answer still reads row 1 inputs.

How the single cell works

What the user sees, what is stored, how it is drawn, saved and copied

What the user sees

Item	Formula: a + b = x
Batch 1	a: 10 + b: 4 = x (answer): 14
Batch 2	a: 2.5 + b: 1.5 = x (answer): 4

+ Add row

Each new row gets its own a, b and x

Input boxes

Number boxes for a and b, with the Integer or Decimal rule.

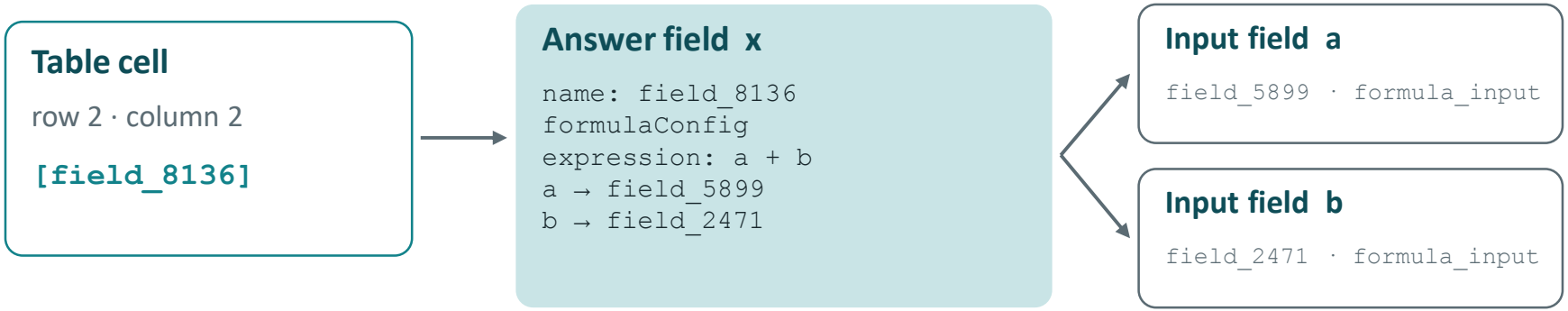
Operators

+ and = are plain text from the calculation, not fields.

Answer box

Read-only and tinted. Fills in live or shows an error.

What is stored behind the cell



CSV template saved for the table

```
Item,Formula: a + b = x
Batch 1,[field_8136]
```

Only the answer has a cell. The inputs are saved as fields and linked by name, so Laravel needs no change.

Render logic: drawing the cell

1

Find the cell

The table renderer reaches a cell whose field has formulaConfig.

2

Split it

formulaLayout("a+b") gives the pieces a, +, b, = and x.

3

Draw the boxes

Each key becomes its input field. Operators stay as text.

4

Calculate live

FormulaOutputInput watches a and b and shows the result.



Save and reload logic

On save

- serializeTableTemplate writes only [field_8136] into the cell.
- It skips fields marked formula_input, because they have no cell.
- The save payload still sends all 3 fields.
- Laravel stores formula_config as it does today.

On reload

- parseTableTemplate rebuilds the cells from the CSV.
- It finds field_8136 and reads its formulaConfig.
- It re-attaches field_5899 and field_2471 as formula_input.
- Without this step the inputs would be dropped.

Net effect: only the builder changes. The database and Laravel stay as they are.

Add Row logic

Today (bug)

Row	Answer field	Its inputs
1	`field_8136`	`field_5899, field_2471`
2	`field_8136_row_2`	**`field_5899, field_2471`**

Reads row 1 inputs

With the fix

Row	Answer field	Its inputs
1	`field_8136`	`field_5899, field_2471`
2	`field_8136_row_2`	**`field_5899_row_2, field_2471_row_2`**

Reads its own row

Rule

When a row is copied, rename the answer's formulaConfig.inputs with the same _row_n suffix as the cells.

Two ways to add a formula

Insert (click) and Copy & paste (in-app clipboard)

Method 1: Insert (click)

1. **Select a cell**

   Click an empty input cell in the table.

2. **Click Σ in the table toolbar**

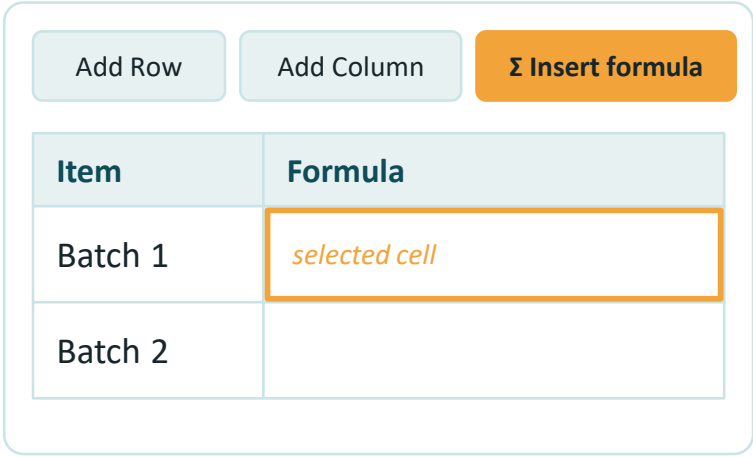
   Opens the formula picker for this section.

3. **Pick a formula**

   For example a + b = x.

4. **The cell is created**

   Three fields are added; the cell shows a + b = x.



Works everywhere

No clipboard: same on localhost, HTTP and HTTPS.

Method 2: Copy & paste

1

Pick in Σ

The formula is kept as the pending item.

2

"Ready to paste"

A chip shows the item. Esc cancels it.

3

Click, Ctrl+V

The paste event fires on HTTP and HTTPS.

4

Builder inserts

It uses its own item, not the real clipboard.

Why an in-app clipboard instead of the real one

- navigator.clipboard only exists on HTTPS or localhost.
- The old execCommand fallback fails inside the MUI dialog because of its focus trap.
- Clear the pending item on Esc, after one use, or when the user copies something else.

Pros and cons

Insert (click)

Recommended

Pros

- Works on localhost, HTTP and HTTPS
- Target cell is explicit, no guessing
- Least code, easiest to test

Cons

- Needs a Σ button in each table toolbar
- User must select a cell first

Copy & paste

Pros

- One global Σ button for everything
- Familiar Ctrl+V for users
- Same flow for symbols and formulas

Cons

- Can hijack a real paste unless cleared
- Some browsers skip paste on empty clipboard

Rules and edge cases

Cell already used

Only an empty input cell can take a formula.

Deleting

Deleting the formula cell also deletes its two inputs.

Moving the cell

Dragging moves the answer cell; inputs follow by name.

CSV mode

The template holds [field_8136]; inputs come back from formulaConfig.

Integer or Decimal

Each input box gets the rule from the formula.

CI4 product app

Its table renderer must also draw and calculate the cell.

Implementation checklist

Area	Change	Where
Toolbar	Σ Insert formula button + selected cell	`builder/index.tsx`
Create fields	Answer in the cell, inputs as formula_input	`builder/index.tsx`
Render	Boxes + operators + live answer in one cell	`renderTableInputCell`
Save	Serializer skips formula_input; payload keeps them	`serializeTableTemplate`
Reload	Re-attach inputs from formulaConfig	`parseTableTemplate (x2)`
Add Row	Rename formulaConfig.inputs per row	`dynamicField copy`
Paste	Method 2 only: allow table cells, use pending item	`paste handler`
Backend / CI4	Laravel: none, verify one save. CI4: draw the cell	`FormController / CI4`

Recommendation

Build Insert first, add in-app Copy & paste later as a convenience, and serve the app over HTTPS anyway.

1 Insert

2 In-app paste

3 HTTPS