

# BANanoVuetifyAD3



## Creating CRUD based BVAD3 WebApps

(for those in a hurry)

A Mashy eBook Project  
[mbanga.anele@gmail.com](mailto:mbanga.anele@gmail.com)  
+27 81 736 6739

## Welcome to the BVAD3 CRUD Builder

The CRUD builder is a source code builder that helps anyone to create CRUD based Vuetify WebApps using BANano + Vuetify + B4XTools. The resulting output for each compilation are the following:

1. Data Table
2. Dialog Box
3. Back-End functionality
4. Route Page Component

Terminology: Any reference to **BVAD3** for the rest of the documentation will refer to BANanoVuetifyAD3, and any reference to **CB** will mean CRUD Builder. Any reference to IndexedDB means BANanoSQL and vise versa.

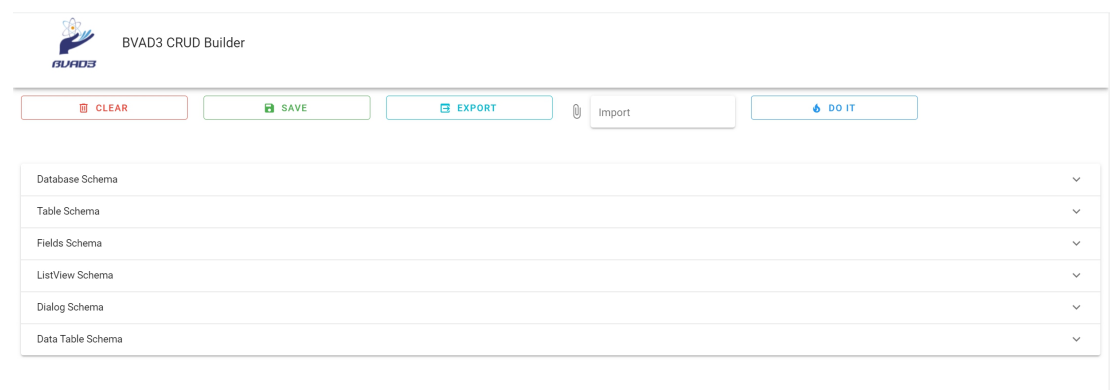
### What will you need to be successful with this CB?

All you need is your database engine type, the table schemas and you are ready to go. One table as a table, resulting in one route component page at a time for BVAD3.

### The Look and Feel of the CB

Figure 1 below shows the working screen of the BVAD3 CB, we will explain the sections as we go along.

Figure 1



This application has been purely developed with Vuetify (CDN). This application generates source code that can be used for your route components.

## 1. Databases

The supported databases by BVAD3 CRUD Builder are BANanoSQL (i.e. IndexedDB), MySQL, MSSQL, SQLite. With the exception of BANanoSQL, all other database connections uses PHP and the in built BANanoPHP functionality.

### 1.1 MySQL

To be able to use MySQL, the MySQL mysqli or rather mysql-nd (native drivers) should be activated in your PHP installation.

## 1.2 SQLite

To be able to use SQLite, the sqlite3 extension should be activated on your PHP installation.

## 1.3 MSSQL

To be able to use MSSQL, the MSSQL PDO drivers should be enabled for your PHP installation. Ensure you activate both -ts (thread safe) and -nts (not thread safe) options after you install the drivers.

## 1.4 BANanoSQL (IndexedDB)

BANanoSQL is a wrap of AlaSQL for BANano. As soon as you define a variable as BANanoSQL on your code, the alasql.min.js file will be added to your project assets.

Tip: For BANanoSQL, after you run your app for the first time after referencing BANanoSQL, you will have to close the app, click Sync on the Files tab and then re-run your app. If you don't do this your app will have an error.

## 2. The Design of the CB

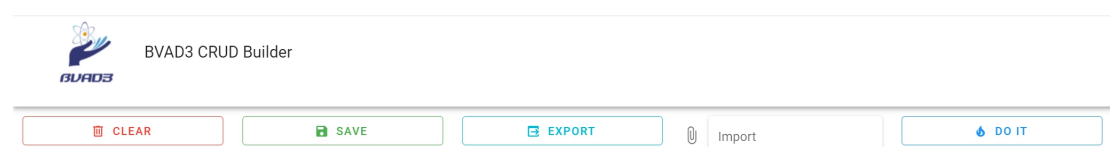
The CB is built using BANanoSQL (IndexedDB) as a back-end, meaning that the content is strictly temporal. You can however back it up / restore it via the interface.

Besides the **top section** of the App, there are 7 panels in the app broken into the following:

- A) Database Schema
- B) Table Schema
- A) Fields Schema
- B) List-View Schema
- C) Dialog Schema
- D) Data-Table Schema

### 2.1 Explaining the Top Section

Figure 2



Function	Description
Clear	Click this button when you want to clear the contents of the current project you are working on within the CB
Save	Save your current work within the CB
Export	Export your current project. This creates a JSON file that you can later import
Import	Click here to select the previously exported JSON CB file
Do It	Click here to create the CRUD source code for your route component page.

## 2.2 Creating a CB project

The CB works on a premise of a single table definition at one time. This means one needs to define the schema of a single table / page before continuing to the next one.

Before you create a CB project, first determine the database you will use, the table(s) you will define and their schemas.

### 2.2.1 Database Schema

The first part is to tell the CB which database you will use for your route component, when the section is expanded, one is able to specify this information.

*Figure 3*

Database Schema

Database Type ☒ MySQL ☐ MSSQL ☐ SQLite ☐ IndexedDB

Host Name / IP Address

Database Name

Username

Password

One needs to select the database engine / type they will use for this table. You are able to select MSSQL, MySQL, SQLite and IndexedDB (BANanoSQL).

Enter the hostname, database name, username and password. The host name is usually localhost or the actual ip address or domain name of your hosting server.

The corresponding part for this is then updating your config.php files in your BVAD project. These are the details for each config file you need to update per database you will use.

Database	Config File to Update	Required Drivers
MSSQL	mssqlconfig.php	MSSQL PDO Drivers

MySQL	mysqlconfig.php	MySQL
SQLite	None	Sqlite3

## 2.2.2 Table Schema

Figure 4

The next part is specifying the database table that you are dealing with. Here you specify the table name in the database, this also requires the primary key name including the name of the auto-increment key name.

Because the CB will create a route component for your BVAD3 project, you also need to specify the component details. Each component can be prefixed for uniqueness.

You are also prompted for a singular and plural name for your component. This information is used for your dialog box when adding / editing database table records.

## 2.2.3 Fields Schema

Figure 5

The table schema is the most important part of the CB. Here you define the fields in your table, their data types, the types of components that each field should use and some properties for the input components you will use.

### 2.2.3.1 Adding fields

To add fields for your table as specified in section 2.2.2, click the Add Fields button,



, this activates a text-area for you to enter all the fields for the table. You can separate the fields with a comma.

Figure 6

Add Fields - separate by ; or ,

Field Names

CANCEL

APPLY

Here you enter the field names separated by ; or , and press apply. This will add all the field names to the data-table as depicted above.

2.2.3.2 Updating Field Details

To activate the inline-editing, click on the field data in the row and column you want. This will activate an editor as depicted in Figure 7. You can then type the new content you need and press enter. At times these editors are text-areas and to persist your changes, you need to click the Ok or Cancel button.

Figure 7

Field Name	Title	VModel	Default	Max Len	Data Type
attrid	#			0	string
projectid	projectid			0	int
componentid	componentid	componentid		0	int

Updating Field Details using the right drawer

You can activate the right drawer for editing purposes when you click the Edit button relevant to the field name you want to update on the table. This is depicted in Figure 8 below.

Figure 8

Field Name

attrid

Title

#

VModel

attrid

Data Type

String

Component Type

TextField

Column Type

TextField

Row Pos

1

Column Pos

1

Default Value

Width

0

Height

0

Alt

Lazy Src

Yes Value

Yes

No Value

No

Avatar Size

0

Max Len

0

Data Source

Key

Value

☒ Active
 ☐ On Table

☒ Hide Details
 ☐ Dense

☐ Clearable
 ☐ Auto Focus

☐ Filled
 ☐ Shaped

Field Name

attrid

Title

#

VModel

attrid

Data Type

String

Component Type

TextField

Column Type

TextField

Row Pos

1

Column Pos

1

Default Value

Width

0

Height

0

Alt

Lazy Src

Yes Value

Yes

No Value

No

Avatar Size

0

Max Len

0

Data Source

Key

Value

☒ Active
 ☐ On Table

☒ Hide Details
 ☐ Dense

☐ Clearable
 ☐ Auto Focus

☐ Filled
 ☐ Shaped

Update the details of the fields and then click Save to update. Close the drawer by clicking Cancel at the bottom of it.

### 2.3.3.3 Deleting a Field

You can delete a field by clicking the delete button. If you want to change a field, you can do it inline or click the edit button to activate the right drawer.

### 2.3.3.4 The ToolBar

Close to the Add fields buttons are other buttons to perform various actions.



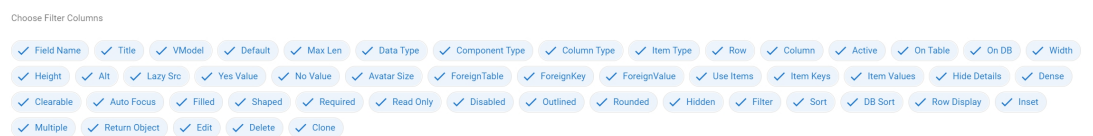
These have the following meanings:

Button	Description of Action
	Clear sorts, groups etc
	Apply column headers - See 2.2.5
	Limit the table columns to: Field Name, Title, VModel, Default, Max Len, Data Type, Component Type, Column Type, Item Type, Row, Column, Active, On Table, On DB, Hidden, Edit, Delete, Clone.
	Limit the table columns to: Field Name, Title, Component Type, Active, On Table, On DB, Hide Details, Dense, Required, Read Only, Disabled, Clearable, Auto Focus, Outlined, Filled, Shaped, Rounded, Hidden, Filter, Sort, DB Sort, Row Display, Inset, Multiple, Return Object, Edit, Delete Clone
	Limit the table columns to: Field Name, Title, Component Type, Alt, Width, Height, Avatar Size, Yes Value, No Value, Foreign Table, Foreign Key, Foreign Value, Use Items, Item Keys, Item Values, Edit, Delete, Clone
	Show all column header

### 2.3.3.5 Table Columns

You can also filter all the columns in the table to display only the ones you need by selecting them from the filter as depicted on Figure 9. When done selecting your field options, click the apply filter button to make your changes.

Figure 9



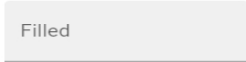
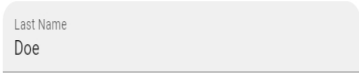
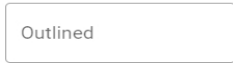
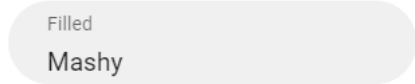

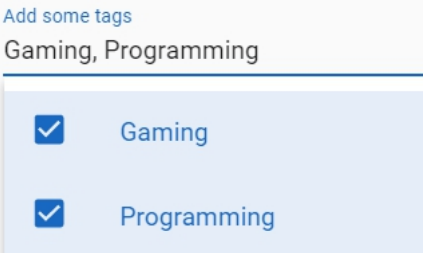
### 2.2.5.1 The meaning and purpose for each column in the CB table columns

Column Name	Purpose
Field Name	This is the name of the actual field name in your backend. It is recommended to make this lowercase.
Title	This will appear on the label for your component.



Column Name	Purpose
VModel	The state variable name to bind the value of the component to.
Default	The default value of the field on the component
Max Len	<p>What is the counter value for this component. As an example:</p> <div> <div>Limit exceeded</div> <div>California is a state in the western United States</div> <div>50 / 25</div> </div>
Data Type	Choose between string, double, integer, blob
Component Type	<p>Which component should be generated. Choose between Paragraph, H6, H1, H2, H3, H4, H5, TextField, TextArea, TimePicker, DatePicker, FileInput, FileInputImage, Money, Thousands, Password, ComboBox, AutoComplete, Select, Avatar, Avatar1, CheckBox, Switch, RadioGroup, Image, Image1, Telephone, Slider, ChipAvatar, ChipGroup, Rating, ProgressCircular, ProgressLinear.</p> <p>In your input dialog, the field name will be shown depending on the component you have chosen.</p> <p>Explore the Vuetify Documentation for more details on how these components work and see our example documentation about these.</p>
Column Type	<p>This applies to the records listing of your table in the data-table and how that field will be displayed on the data-table. Choose between Normal, TextArea, TextField, IconView, Chip, Switch, Action, Image, CheckBox, Time, Money, AvatarImg, Rating, ProgressCircular, ProgressLinear, DataColumn, DateTimeColumn, NumberColumn, ButtonColumn, LinkColumn, ComboBox, AutoComplete, Select.</p> <p>Explore the Vuetify Tables fore more details and see our documentation on examples of tables.</p>
Item Type	<p>This applies to what the field will be displayed as when a ListView is used. This option generates source code specific to list views. Choose between Key, Icon, IconColor, RightChip, RightChipColor, Avatar, Title, SubTitle, SubTitle1, SubTitle2, SubTitle3, SubTitle4", RightIcon, RightIconColor, RightText, RightRating, URL, LeftIcon, LeftIconColor, LeftCheckBox, LeftSwitch, RightSwitch, RightCheckBox, AvatarIcon, AvatarIconColor, RightAvatar, RightAvatarIcon, RightAvatarIconColor</p>
Row	Where the component will be placed on the input dialog
Column	Where the component will be placed on the input dialog
Active	When checked, it means this field can be used for design. In-active fields are ignored by the code generator.
On Table	When checked, the field should appear as a column in the data table listing

Column Name	Purpose
On DB	When checked, this field will be used for v-model / value binding
Width	The width of the image
Height	The height of the image
Alt	Alt for the image
Lazy Src	Related to images
Yes Value	The yes-value for Switch/CheckBox
No Value	The no-value for Switch/CheckBox
Avatar Size	Obvious
Foreign Table	The name of the foreign table to use as a data-source for this field. Foreign Table, Foreign Key and Foreign Values are used for ChipGroups, ComboBox, AutoComplete, Select, RadioGroup
Foreign Key	The name of the foreign field in that foreign table to use for this field
Foreign Value	The name of the foreign field in that foreign table to use for this field.
Use Items	If you want to specify your own source for data sources, you can check this on
Item Keys	The values of the keys to be used from the data-source object list
Item Values	<p>The values of the items matching to the item keys to be used for the data source object list. For example</p> <div> <input checked="" type="checkbox"/> Boolean,String,Int         <input type="checkbox"/> Boolean,String,Int       </div> <p>In the selection I want to use Boolean, String and Int, an own defined data source.</p>
Hide Details	Hide the section to show error messages
Dense	Shrink the height of the component
Clearable	<div> <input type="text" value="Hey!"/> <span>×</span> </div> <p>Must have the X to clear the content</p>
Auto Focus	Receives the auto-focus
Filled	

Column Name	Purpose
	
Shaped	
Required	This is a required field. When turned on, rules are added to the source code to check if content is specified on the input dialog.
Read Only	Allow/Do not allow data entry.
Disabled	Mark the element as disabled
Outlined	
Rounded	
Hidden	Apply the v-show directive to hide the component on the page.
Filter	Should show on the column filters on the data-table and should be filterable on the data table.
Sort	Should be able to be sorted by on the data-table.
DB Sort	This field will be applied when sorting from the database.
Row Display	Uses for Radio Group to show radios as a row instead of a column.
Inset	
Multiple	
Return Object	Return the selected object and not just the key, useful with combos, selects, auto-completes

Most of the input components share the same properties, so you can design a nice UX based on the choices that you make here.

#### 2.2.4 List-View Schema

For now lets skip this, its currently not used as I added a shortcut to the code using the “Item Type” column.

2.2.5 Dialog Schema

Figure 11

Dialog Schema

Focus On

attname

Display Field

attname

Dialog Width

800

Full Screen on Mobile

Lazy Validation

This part contains some additional settings for your input dialog. One specifies the field name for the component to focus on. The display field is used during deletes and the app will read and display and show that field for your delete confirmations.

Then you specify the width of the dialog box and whether the dialog should be full screen on module devices.

2.2.6 Data-Table Schema

Figure 12

Data Table Schema

Data-Table Preferences

Dense

PDF

Excel

Search

Add New

Clear Sort

Filter

Refresh

Back

Edit

Delete

Clone

Print

Save

Cancel

Download

Menu

Show Select

Single Select

Show Group By

Must Sort

Multi Sort

Sort By

Items per Page

This final section of the CV is applicable to the data-table. The data-table has a title, toolbar buttons and action buttons on the columns. We can control what appears on the data-table and the events it should have here.

We have already specified the sorts when we were defining the fields, you can ignore that here. You can also indicate the number of items per page. Lets clarify the data-table settings.

Settings	Description
Dense	The data table should be dense
Excel	Add an excel export button to the data-table toolbar
Search	The data-table has search functionality
Add New	Add an add new button to the toolbar
Clear Sort	Add a clear sort button to the toolbar
Filter	Add a filter button to the toolbar
Refresh	Add a refresh button to the toolbar
Back	Add a back button to the toolbar
Edit	Add an edit action button per row
Delete	Add a delete action button per row
Clone	Add a clone action button per row
Print	Add a print action button per row
Save	Add a save action button per row
Cancel	Add a cancel action button per row
Download	Add a download action button per row
Menu	Add a menu action button per row
Show Select	Show a select all option on the data-table
Single Select	The selection mode should be for each record
Show Group By	Show the group by option on the data-table
Must Sort	The records in the data-table can be sorted
Multi Sort	The columns can be sorted with more than 1 column

### 3. Generating BVAD3 CRUD source code

When you are done specifying the criteria for your table. You can then click this button



For the source code for the route component page to be created. This is generated and downloaded by the CB. The generated code will vary depending on what you want to achieve. We placed a little snippet here as per Figure 13.

NB: In case an error is raised, close and re-run the CB website, it will open up with your previous work.

*Figure 13.*

```
attributes (1).txt X  Untitled-3  <v-app id="inspire">  Untitled-2  <div id="placeholder">  Untitled-1
C: > Users > mbang > Downloads > attributes (1).txt
1 CREATE TABLE IF NOT EXISTS `attributes` (`attrid` VARCHAR(255) NOT NULL PRIMARY KEY AUTO
2
3 'static code module
4 Sub Process_Globals
5 Public vuetify As VuetifyApp
6 Public attributes As VueComponent
7 Public path As String
8 Public name As String = "attributes"
9 Private Banano As BANano
10 Private dtAttributes As VueTable
11 Private dlgAttributes As VueElement
12 private contAttribute As VueElement
13 Private msgBox As VueElement
14 Private Mode As String
15 Private foreignProjects As Map
16 Private foreignComponents As Map
17 Private foreignAttrtypedes As Map
18 Private foreignOnconditionds As Map
19
20 End Sub
21
22 Sub Initialize
23 'establish a reference to the app
24 vuetify = pgIndex.vuetify
25 ''initialize the component
26 attributes.Initialize(Me, name)
27 attributes.vuetify = vuetify
28 path = attributes.path
29 '
30 'setting up permissions
31 attributes.SetData("attributes", attributes.NewMap)
32 ''add a msgbox dialog for this page
33 msgBox = attributes.AddMsgBox(True, 500, "success", "error")
34 'add a container to hold the attributes
35     contAttribute = vuetify.AddContainer(Me, attributes.Here, "contAttribute", True)
36     contAttribute.Blurred = "!attributes.see"
37     contAttribute.AddRows1.AddColumnns12
38     contAttribute.BuildGrid
39     '
40     attributes.BindVueElement(contAttribute)
41 '
42 CreateAttributesTable
43 CreateAttributesDialog
```

The first line usually is the SQL query string to create your table and the rest is the code that you can copy and paste on your code module. To use this code, on your BVAD3 project.

### 3.1 Using the generated CRUD source code in your project

From an existing BVAD3 project you are working on...

1. Click Project > Add New Module > Code Module. Type the name of your code module e.g. ViewContacts
2. Copy and Paste the all the code from 'static code module.' to your new code module
3. Ensure you add the page on pgIndex.AddPages e.g. ViewContacts.Initialize

4. Ensure you have a link to the page on the `pgIndex.BuildNavigationDrawer` code

NB: You might have to remove the code to create the database table from the generated source.

If you don't have an existing project, you can create a new project using the BVAD3 (1) template.

Enjoy!