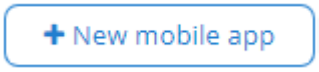
AppSheet - How to create an application? Development guide

AppSheet - How to create an application? Development guide

Go to AppSheet, log in and go to "My Apps". Then click on 

and follow the AppSheet steps.

You can use the following development guide or open the Solvay App Catalog to find new ideas.

Elements of AppSheet

This tool has different elements which you have to use when you want to make an app.

Data tables

The first step for your data to become an Appsheet app is to be added as a table to the template of the app. These tables are simply “models” of the rows and columns of the Spreadsheet sheets. For this reason, when you make your sheet you have to think that column headers are very important because these define the structure of the table. Another interesting point to take in care in this process are changes in rows, while columns are always the same, rows can change as needed by adding, deleting or updating new data. Appsheet allows you to add as many tables as you need.

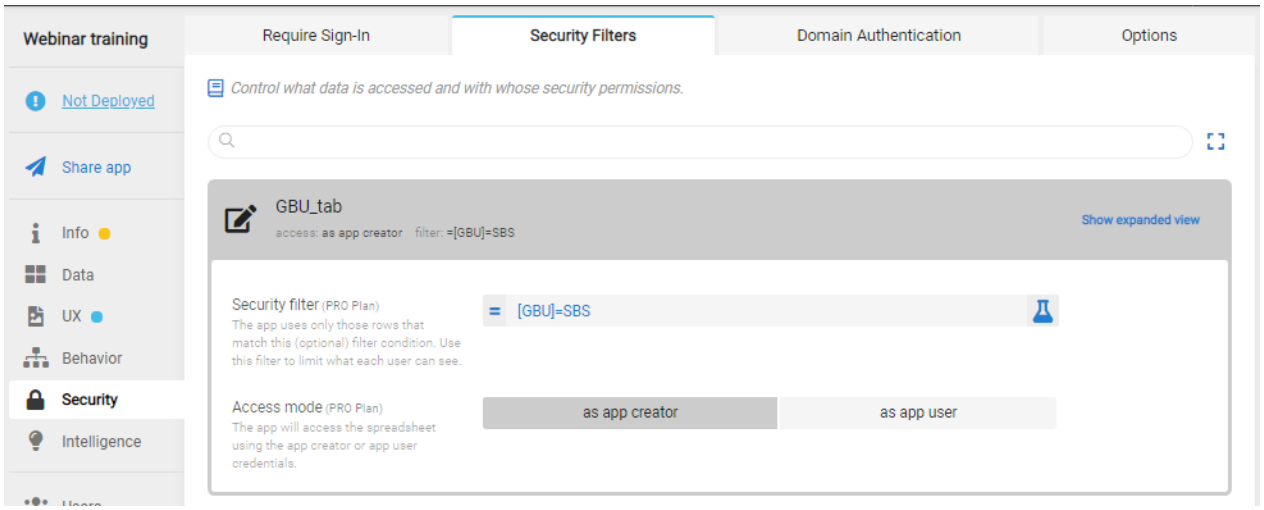
KEY	NAME	LAST NAME	EMAIL	PICTURE	FUNCTION	GBU	SITE
APIwDS31	Matteo	Menghetti	matteo.menghetti@solvay.com	WEBINAR TRAINING_Images/APIwDS31.PICTURE.022501.jpg	IS digital analyst	SBS	Lyon EPA
6dziYnnl	Francis	Boulu	francis.boulu@solvay.com	WEBINAR TRAINING_Images/6dziYnnl.PICTURE.022746.png	IS digital analyst	Industrial Functic	NOH
mtna3bnA	Pascal	Hierckens	pascal.hierckens@solvay.com		IS digital analyst	Industrial Functic	NOH
uFPiGZJ1	Diego	Ballarin	diego.ballarin@solvay.com		Change agent	Specialty Polyme	Spinetta
H6EQ001W	Christophe	Mallevaey	Christophe.Mallevaey@solvay.com		SAP expert	Industrial Functic	BELLE ETOILE
8AZcdkzb	Travis	martin	Travis.martin@solvay.com			Technology Solutions	
RU6uOTF1	Pedro	Falcao	pedro.falcao@solvay.com	WEBINAR TRAINING_Images/RU6uOTF1.PICTURE.051359.jpg	DEVELOPER	SBS	Lisbon

The rows have different categories, these can be primary keys, required, used only for read, labels, hidden. All tables must have a primary key because it is the column which identificate each row. The definition of the key is not unique, in the example below I the column *KEY, EMAIL* can be a good key. It is only important that its value its always different from a row to another.

KEY	NAME	LAST NAME	EMAIL	PICT
APlwDS31	Matteo	Menghetti	matteo.menghetti@solvay.com	WEB
6dziYnnl	Francis	Boulu	francis.boulu@solvay.com	WEB
mtna3bnA	Pascal	Hierckens	pascal.hierckens@solvay.com	
uFPiGZJ1	Diego	Ballarin	diego.ballarin@solvay.com	
H6EQ001W	Christophe	Mallevaey	Christophe.Mallevaey@solvay.com	
8AZcdkzb	Travis	martin	Travis.martin@solvay.com	
RU6uOTF1	Pedro	Falcao	pedro.falcao@solvay.com	WEB

Security filters

If you want to protect some data of your tables, the best form is using Security Filters. A security filter is an expression that is evaluated on each row in a table and returns true or false. If it is true, the row is included in the app and if it is false, the row is discarded.

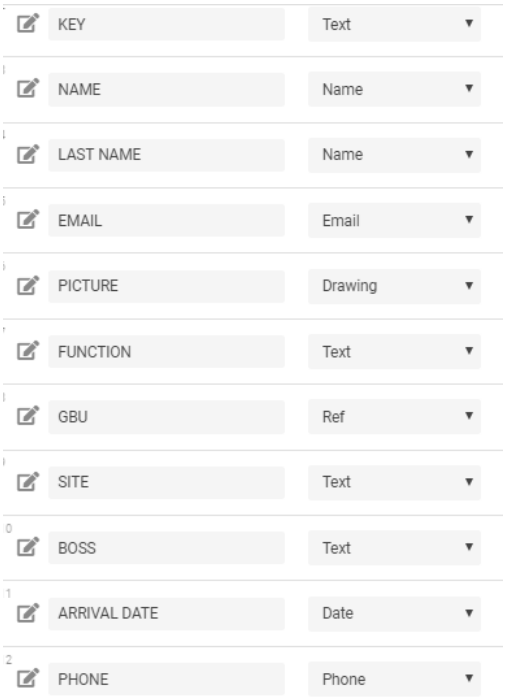


Column Types

AppSheet has a lot of column types which help you to differentiate the type of information you are using on each column. These types can be organized in the following groups:

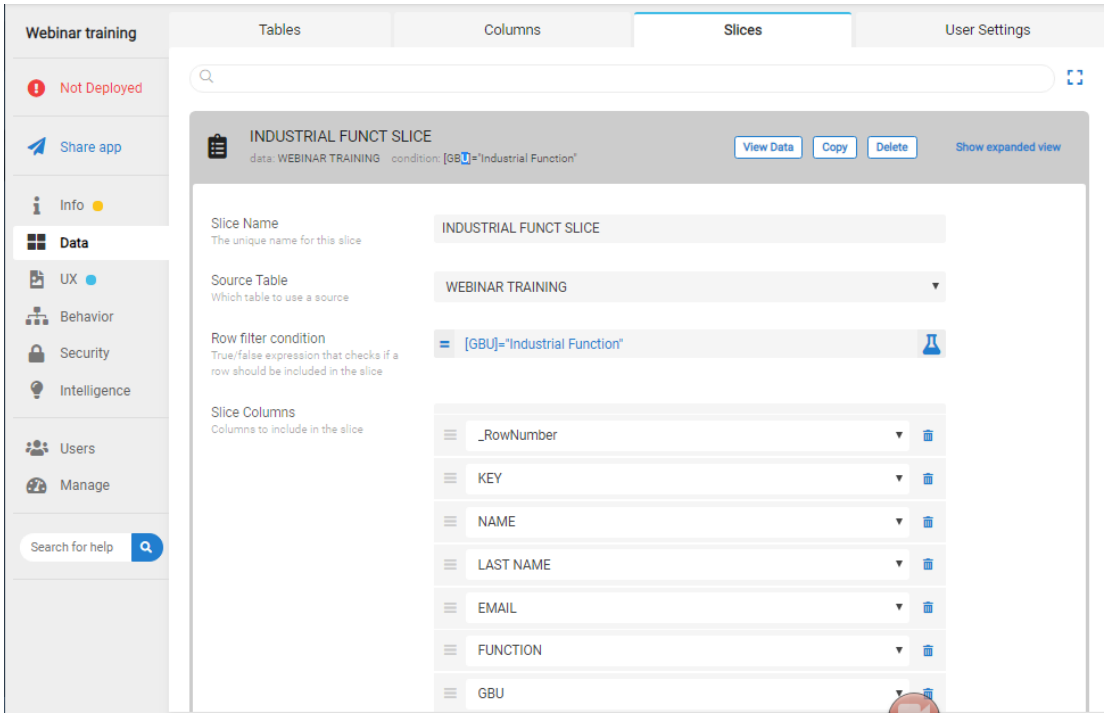
- Numeric types (Number, Decimal, Price, Percent)
- Temporal types (Date, Time, DateTime, Duration)
- Change types (Change Timestamp, ChangeCounter, ChangeLocation): are using in apps in which is important to record a timestamp or increment a counter automatically in a row when changes are made to other columns, and even _values_ within the columns.
- Enumerated types (Yes/No, Enum, EnumList, Ref, Color, Progress)
    - You can use the type Ref to do references between tables. It can improve a lot the structure of your app
- Communication types (Phone, Email)
- Geographic types (Address, LatLong)
- Content types (Show, Image, Drawing, Thumbnail, Signature, File)

- Other types (URL, App, MultiColumnKey)



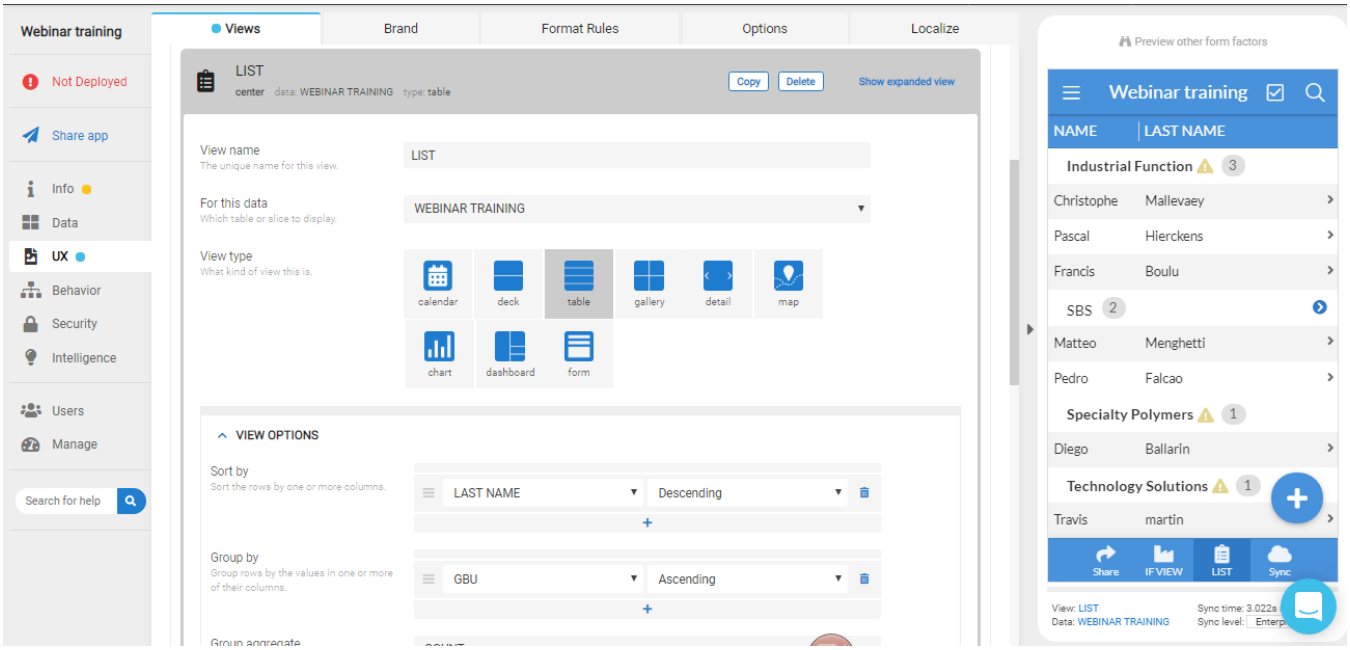
Slices

Slice is like a sub set of rows from a table, and can be used in the same place as a table but they haven't security filters. The slice below gather together all employees who belong to the Industrial Function



Views

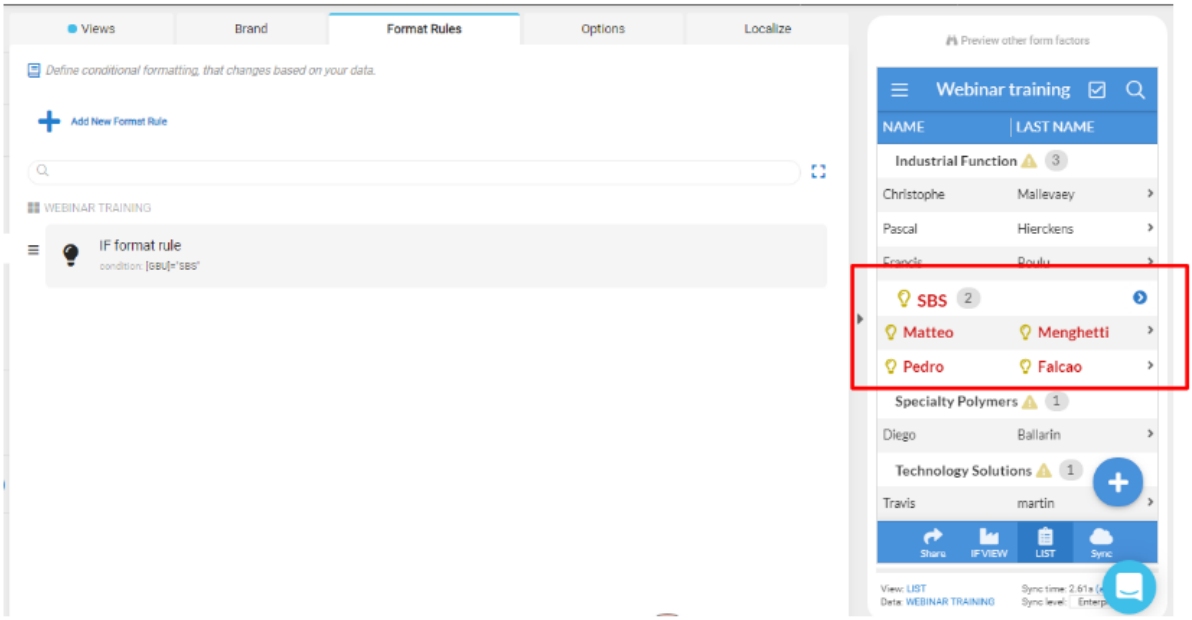
The next step, it is to build views for your tables. Views are the tabs that you will see in your app. There are several types, some of them like Deck and Table summarize the data as a scrollable list of rows, however Gallery lets you show the content like a summary with images, others like Chart or Map are more specific. One of the most interesting is Form because it allows you to make registers directly. Also, you can choose the names of your views, their icons, when they are showing...



Format Rules

Format Rules transform the way information is displayed throughout your apps. You can do many changes in the format of your text: color, type, size... Moreover, format rules allow you to add icons next to the rows.

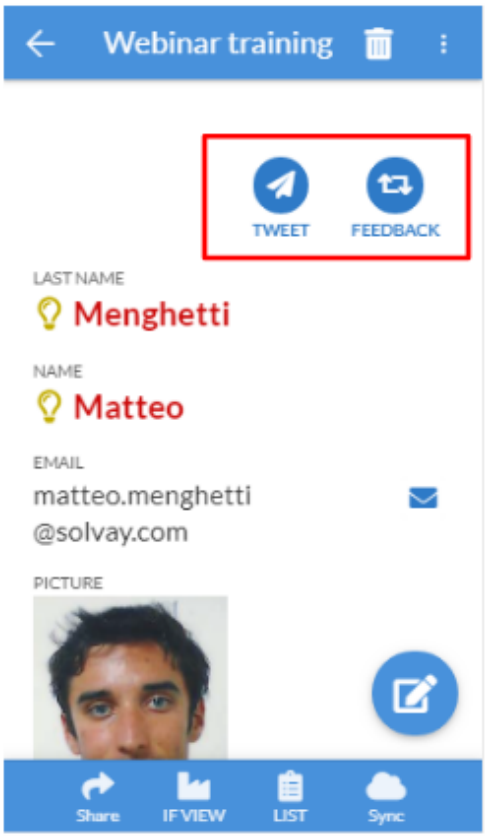
The same app can have different format rules.



Behavior

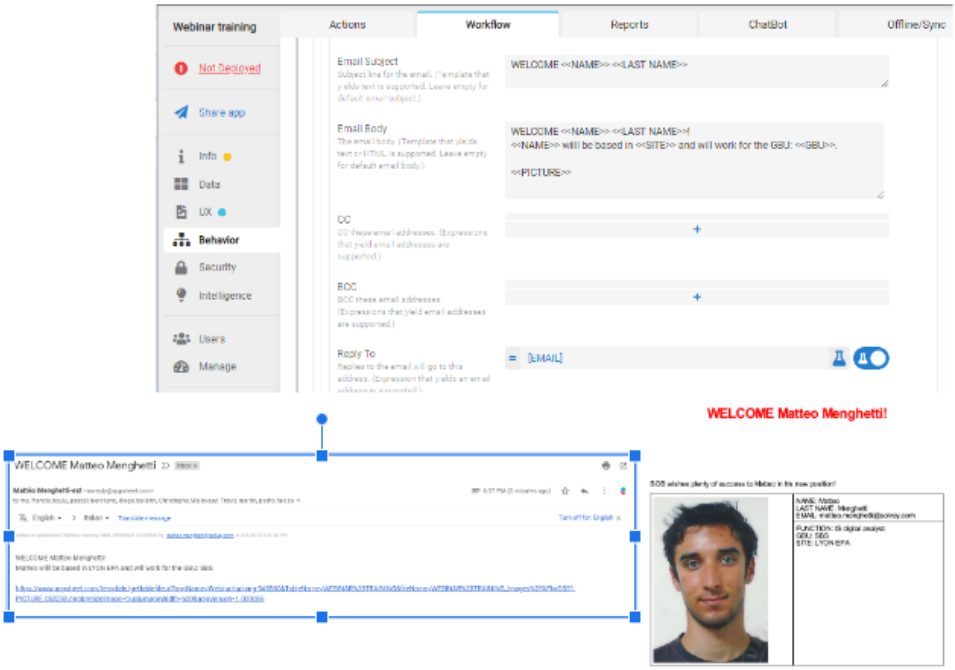
Actions

Actions are mini-tasks that can modify data in your app. They have different uses: set or update the value of a field on the row, open a different AppSheet app or go to another view in the current app, copies the current row and opens it in a form view.



Workflows

Workflows allow you to automatically invoke actions when an events occur. They support the following actions: customize and send email, customize and sent SMS messages and invoke cloud services.



Expressions

Expressions may be used in various AppSheet features to customize app behavior and provide your users with advanced functionality. When you want to make an expression you have to write values with the following forms: constants using "double quotes" except for numeric values and column names using [square brackets]. These are the built-in functions of Appsheet:

- NOW(): for the current DateTime
- TODAY(): for the current Date
- TIMENOW(): for the current Time
- HERE(): for LatLong of the current Location
- USERNAME(): for the Name of the current user
- USEREMAIL(): for the Email of the current user
- UNIQUEID(): to create unique Text values for keys
- ISBLANK(<expression>): to check if an expression is empty
- CONCATENATE(<text-expression1>, <text-expression2>, ...): to combine two or more text values
- LEN(<text-expression>): to get the length of a text value
- IF(<condition>,<then-expression>,<else-expression>)
- IFS(<condition1>,<then-expression1>, <condition2>, <then-expression2>, ...): to provide a sequence of condition-value pairs that are evaluated left-to-right until one of the conditions is true.
- SWITCH(<expression>, <value1>, <result1>, <value2>, <result2>, ..., <default_result>): to choose one of the results based on the value of the expression. It is a simpler version of IFS.
- ORDERBY(<List of Ref expression>, <column1>, <direction1>, <column2>, <direction2>, ...): to sort a list of references. The first argument must yield a list of references, i.e. a list of the keys of the records to sort. This is followed by one or more pairs indicating a column name to order by, and its ordering direction. The value TRUE indicates "Descending" order. The value FALSE indicates ascending order. If the data should be ordered by just one column (which is the common case), the ordering direction may be omitted and defaults to FALSE (so the rows are sorted in ascending order). For example, OrderBy([Related Orders [Customer Name]], [Order Date], FALSE)

- SECOND(duration): for the Second component of a Specific Time
- MINUTE(duration): for the Minute component of a Specific Time
- HOUR(duration): for the Hour component of a Specific Time
- DAY(date): for the Day of the Month
- WEEKDAY(date): for the Day Number from a Date
- WEEKNUM(date,[type]): for the Week Number from a Date
- MONTH(date): for the Month Number from a Date
- YEAR(date): for the Year from a Date

Appsheet has these arithmetic operators to build arithmetic expressions:

- Add: +
- Subtract: -
- Multiply: *
- Divide: /

Besides, Appsheet contains other operators:

- Equals: =
- Not Equals: <>
- Greater Than: >
- Greater Than or Equals: >=
- Less Than: <
- Less Than or Equals: <=
- AND(expr1, expr2, ...) : returns true if all the sub-expressions are true
- OR(expr1, expr2, ...): : returns true if any of the sub-expressions are true
- NOT(expr): returns true if the sub-expression is false and vice-versa
- ISBLANK({*}) returns true if an expression is empty (ISBLANK)
- CONTAINS({text_1},{text_2}): returns true if text_1 contains text_2
- IN({*},{List}): returns true if a value is in a list

Structure

Menu

In the majority of cases, to do a menu is the best starting point.

Tables

First of all, you need a table that contains the options you can access through your menu. It structure should be similar to this:

Menu	Link	Image	order	ROLE
Agenda	#control=Agenda	icons/IND_Agenda2_240x240_OCT17.png	1	PARTICIPANT, SPEAKER
Who's who	#control=People	icons/IND_WhoIsWho_240x240_OCT17.png	2	PARTICIPANT, SPEAKER
Maps	#control=Maps	icons/IND_Map_240x240_OCT17.png	3	
Feed	#control=FeedView	icons/LiveFeed.png	4	
Feedback	#control=FeedBackForm	icons/IND_Vote_240x240_OCT17.png	5	PARTICIPANT
My contacts	#control=My contacts	icons/IND_Community_240x240_OCT17.png	6	ORGANIZER

- ID
  - Description: Number of each row
  - Type: Number
  - Primary key
  - Hidden
  - Required
  - Initial value: [_RowNumber]-1

- Menu
  - Description: Name of each option

    - Type: Text
    - Required

- App
    - Description: Address in the app of each option. You need use this expression “#control=Name of the view”
    - Type: App
    - Required

- Image
    - Description: Icon of each option
    - Type: Image
    - Label
    - Required

- Role
    - Description: Role of the users who can show each option. If this field is without text, all users will show this option.
    - Type: Ref (to the next table)
    - Hidden
    - Required

Then it is necessary make another table that relate each role with users (in this case using their email). It will help you to restrict the access to the registers.

ROLE	MEMBERS
ORGANIZER	john.doe@solvay.com
PARTICIPANT	brian.smith@solvay.com, alice@solvay.com
SPEAKER	luke@solvay.com, ann@solvay.com

- Role
    - Description: Name of different roles
    - Type: Text
    - Primary key
    - Label
    - Required
- Members
    - Description: E-mail of the different users who have each role.
    - Type: Email

In Appsheet, the first table must be reference the following table. Moreover, both tables are configured as“Only-read”.

View

It is necessary only one view, it uses the data of the first table and it type is “gallery”.

Security filter

The next step is to use a security filter for the first table to select which rows can be displayed to each user. You can use this formula:

blocked URL

Access

When you are going to create a new tab, you must have clear who will have access to it registers.These are the most common cases and their solutions:

- All users can add registers, however each user only can show his/her registers which can update
    - The column “Role” in it row has to be empty
    - It is possible to update and to add registers in the table
    - It view show the data of a slice with a filter condition like this: [User]=USEREMAIL() . In this slice is possible to add and to update registers.
- Someone can add, update and show all registers
    - The column “Role” in it row has to be empty
    - It is possible to update and to add registers in the table
    - It view show the data of the table
- All users can show all registers, but they only can add and update their own registers
    - The column “Role” in it row has to be empty
    - It is possible to update and to add registers in the table
    - This option require two views
        - First includes him/her own registers
            - The data source of this view is a slice with a filter condition like this: [User]=USEREMAIL() . In this slice is possible to update and to add registers
        - The other includes all registers
            - The data source of this view is a slice in which it is only possible to read registers
- Someone can add and update his/her own registers, moreover some users can show all registers
    - It is possible to update and to add registers in the table
    - This option require two views
        - First includes him/her own registers
            - It is necessary include the role of users who only have access to his/her own registers in the column “Role” of the row of this view
            - The data source of this view is a slice with a filter condition like this: [User]=USEREMAIL() . In this slice is possible to update registers
        - The other includes all registers
            - It is necessary include the role of users who have access to all registers in the column “Role” of the row of this view
            - The data source of this view is a Slice e in which it is only possible to read registers

Interesting webinars

First contact

What is the function of AppSheet?:

https://www.youtube.com/watch?v=AvE_QEOtijA&index=3&list=PLZ81nepkT97KAehtQ_Y__bNJX3qZaStvt

Webinar about the basic steps for make a new app:

https://www.youtube.com/watch?v=OG-TFY18xmU

Column types:

https://www.youtube.com/watch?v=FPxxhQXbejg

https://www.youtube.com/watch?v=v_TQGe9gnLs

https://www.youtube.com/watch?v=n7BzR33b-e8

https://www.youtube.com/watch?v=j_z_S7Glj9I

Webinar concerning filter data (Slices and Security filters):

https://www.youtube.com/watch?v=tkJhVgvj1ak

More advanced users

Connect different tables with reference type:

https://www.youtube.com/watch?v=iCKkK0zWiow

Make a menu using Valid_If:

https://www.youtube.com/watch?v=WA6Ku8ta2MU

Webinar about several things that you can use in AppSheet:

https://www.youtube.com/watch?v=HinyDIw2E1I

Webinar about the meaning of the different possible expressions:

https://www.youtube.com/watch?v=AkrtzKm9RTQ

Interesting to go deeper in the use of actions

https://www.youtube.com/watch?v=Ome1ZFm7rYo