Last revision: 27 sept 2022

The purpose of this project is to give to *business users* a *no-code* tool for generating forms.

This project called *GFM Builder* is at prototype stage. It relies mainly on API wich are not documented, even if there are widely used in APEX export App files (before 22.1 version). This approach is clearly no supported, so this tool must be considered as a playground and used in a sandbox only.

The application GFM Builder is available from Github.

Target audience

GFM Builder addresses mainly companies intranet requirements but citizen portals needs as well.

A common use case is to trigger an administrative process based on informations provided by an employee or a citizen. ie: renewal contrat, ordering internal good, signaling problem in a district, etc.

This tool will leverage, in a next version, the new approval task process which can be embeded in a page.

Once these limitations given, let’s see the functionnal aspects.

Contenu Afficher

High level design

Persona and Role definitions

GFM Builder is dedicated to *business users*.
That means that we haven't to declare users as APEX developers. Users will be able to author and publish forms without any knowledge of Oracle APEX.

We consider the following roles:

- Regular APEX Roles
    - APEX Workspace Administrator
    - APEX Developer
    - APEX End User
- Application Roles brougth with the tool
    - GFM Builder Administrator
    - GFM Builder Developer
    - Final user (means an APEX End User or an anonymous user)

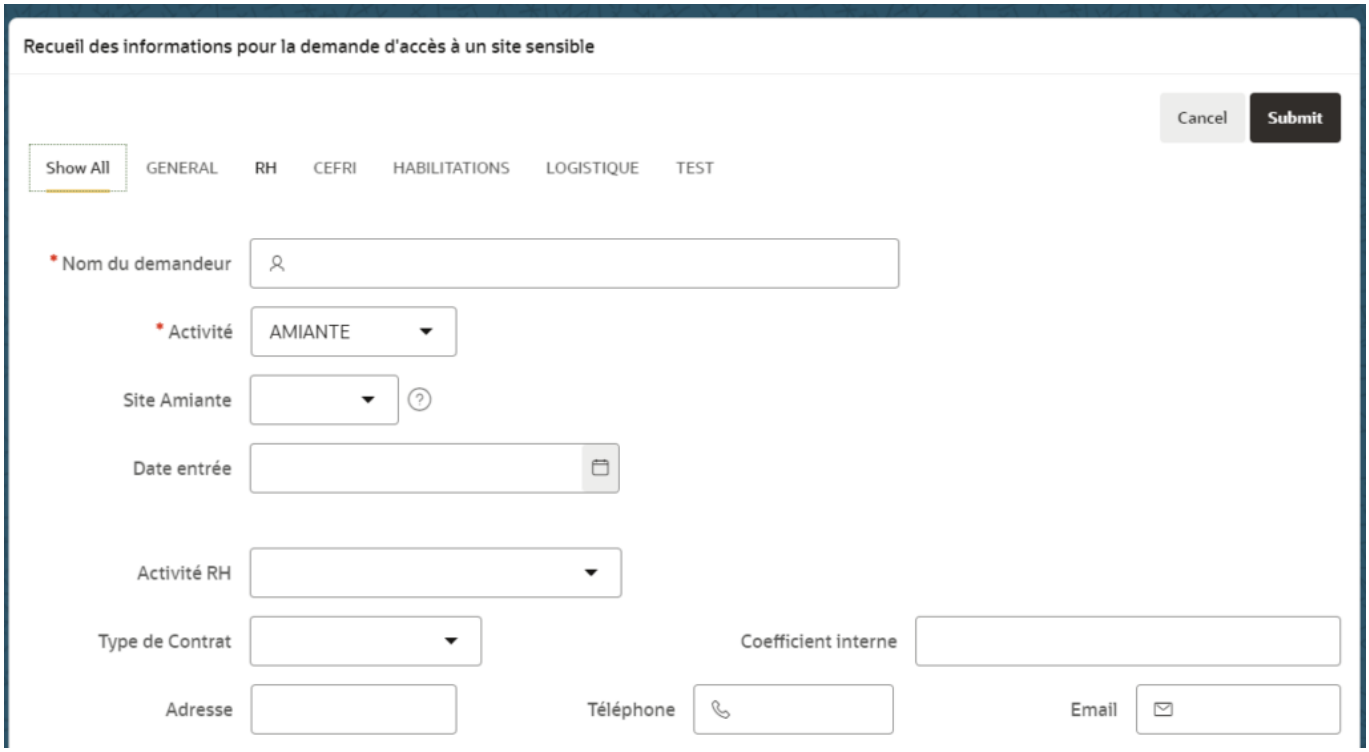
The GFM Builder Role is implemented as a *Contributor* Role in the APEX simplifed ACL model.

The GFM Builder Administrator role is maped on *Admin* Role

In order to build a form, the GFM Builder Developer role must be granted to an APEX End user.

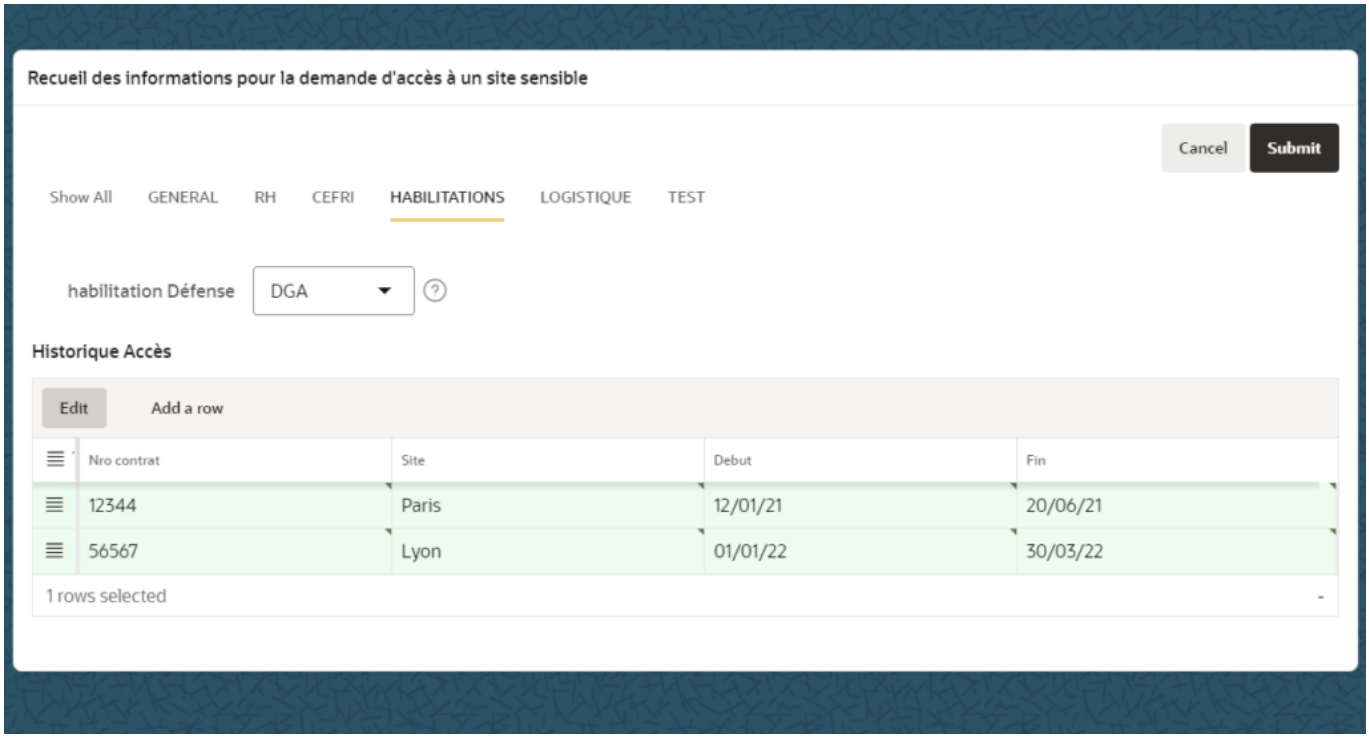
Generated Form Samples

Below are sample of forms generated with GFM Builder.



In this case, we choosed to display fields grouped by sections (GENERAL, RH, ..)

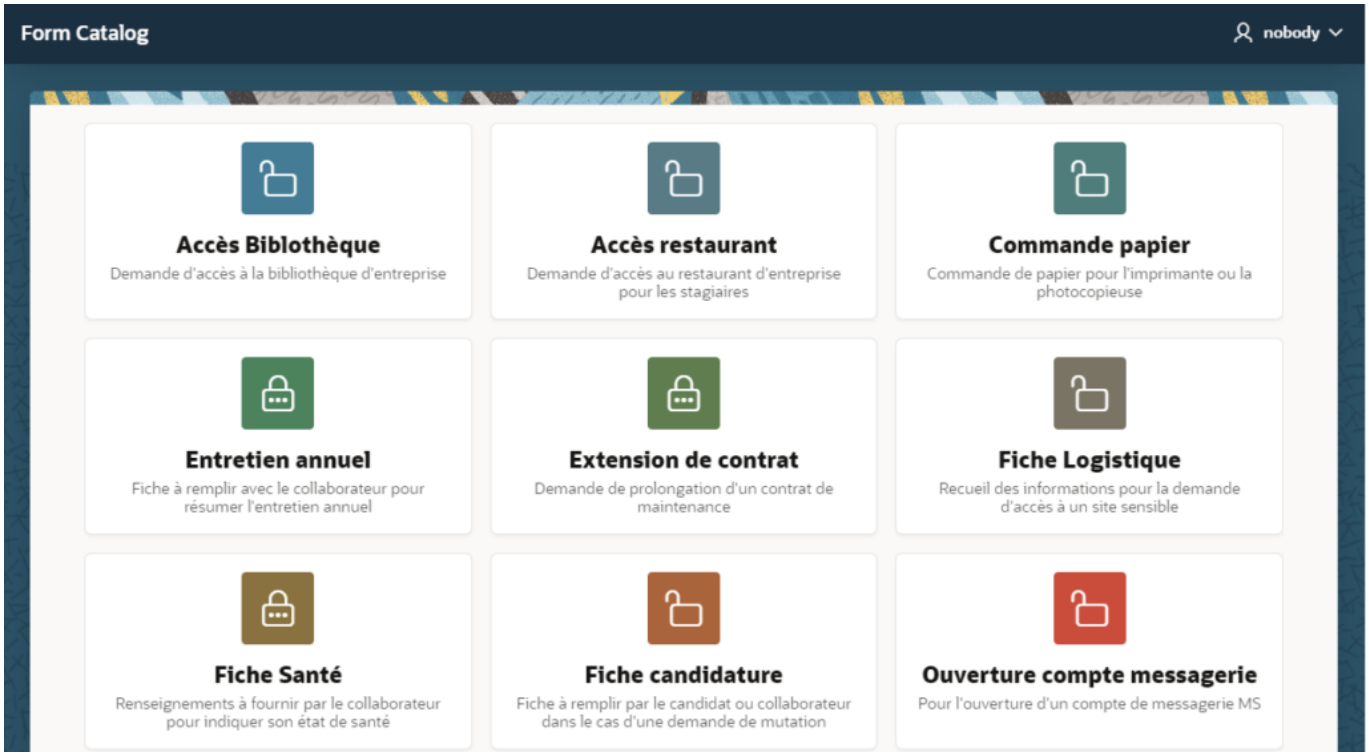
A form can embed a detail array for multivalued informations. In this case, it’s a historic of entries.



Using a detail array inside a form

Apps Store

A sample catalog application (GFM Store) is provided as an exemple. Each time a form is tagged as *Published*, it will be displayed in this catalog.

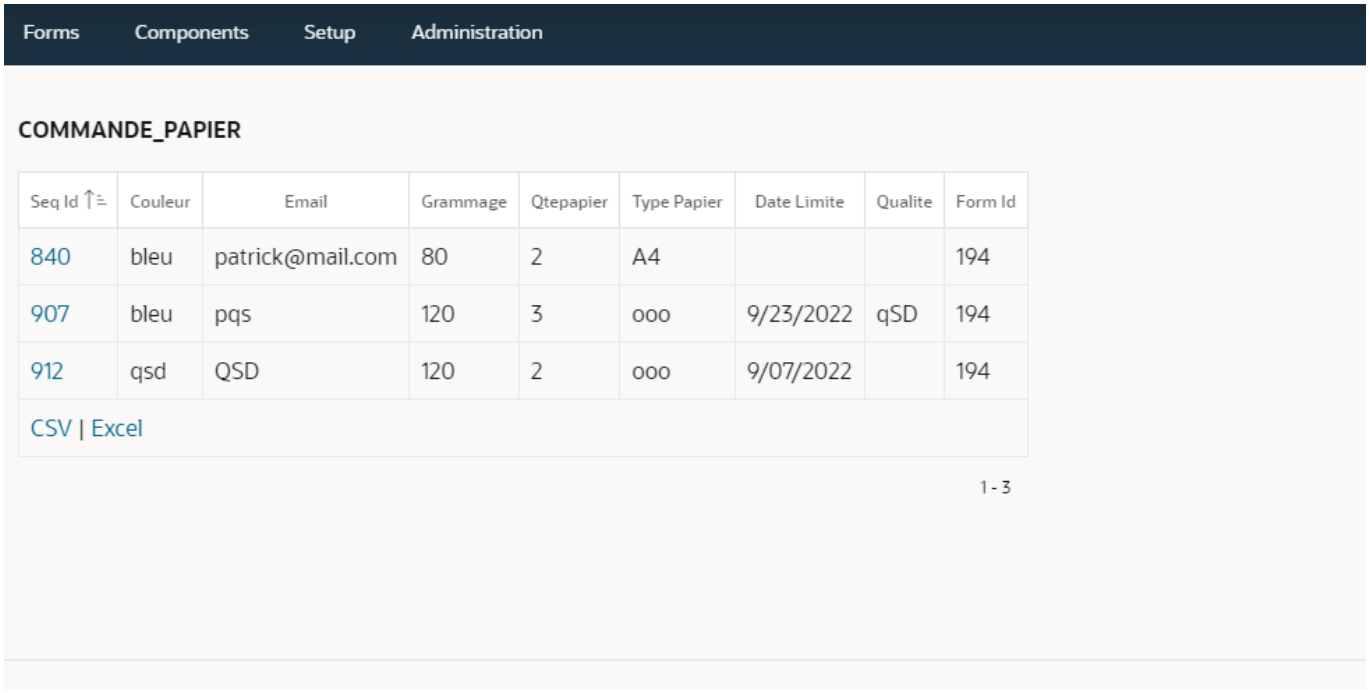


Data collecting

All data are stored in a unique table GFM_DATA.

The data can be displayed by GFM Builder Admin and Forms owners.

Final users will not be able to display data. We suppose that a process will send data to the user, by any feedback way, email for instance. It’s not in the scope of the project.



Simplified report displaying data collected through a form

Form design

The design process is mainly declarative. It’s a matter of designing the items which constiture the form.

Instead of creating a data set then building the form (in in standard way), we create there both the data set and the form in the same user interface.

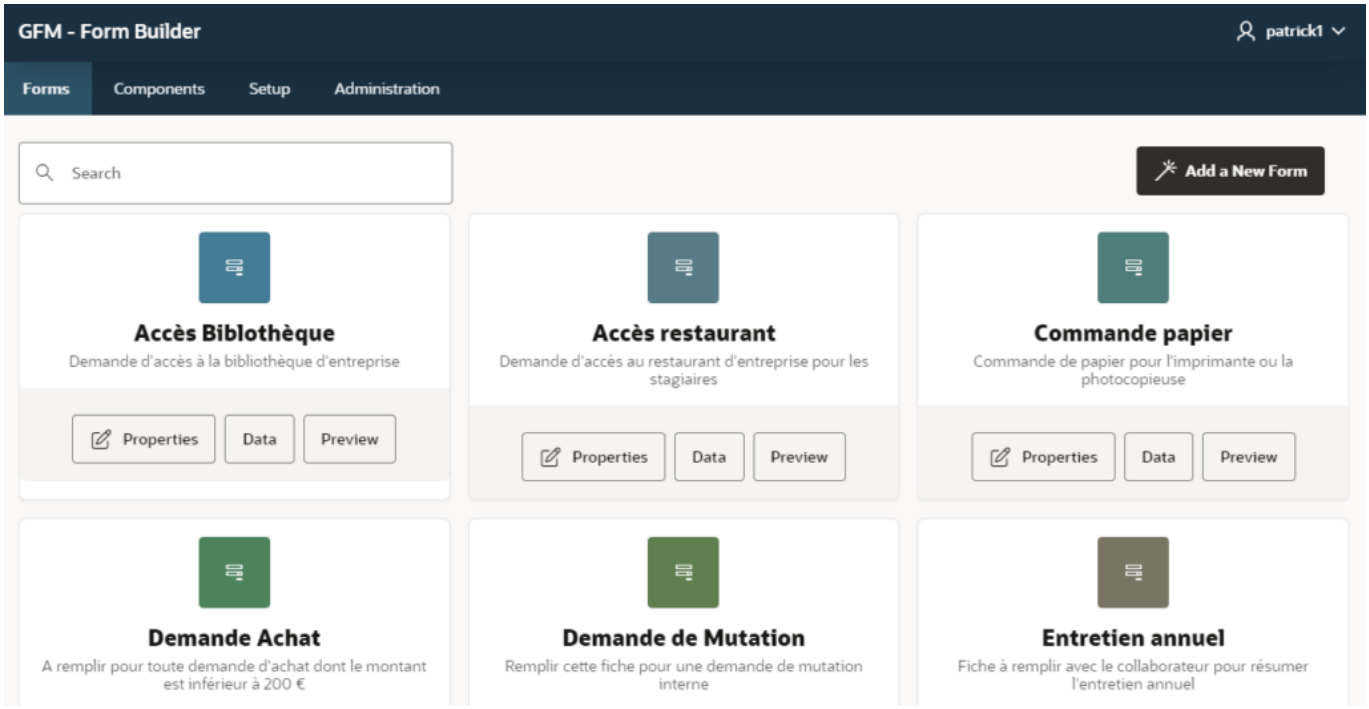
A form is comprised of items. An items is either a simple attribute (like text Field, date) or a multivalued attribute. This second kind of items is implemented as an Interactive Grid. This technical choice provides more flexibility during fields filling.

Items must be pick up from a predifined collection of components which are:

- Basic items

Text,Number, Date, email, Inline List (List of values declared inline)

- Predifined List of values
  These lists are created in a dedicated page called ‘Components’, either by an administrator or the user himself.
- Grids
  It’a an existing form wich is declared as been usable as a grid inside an other form. ie: Historic of positions, list of contracts, etc.



Reusing existing form definitions

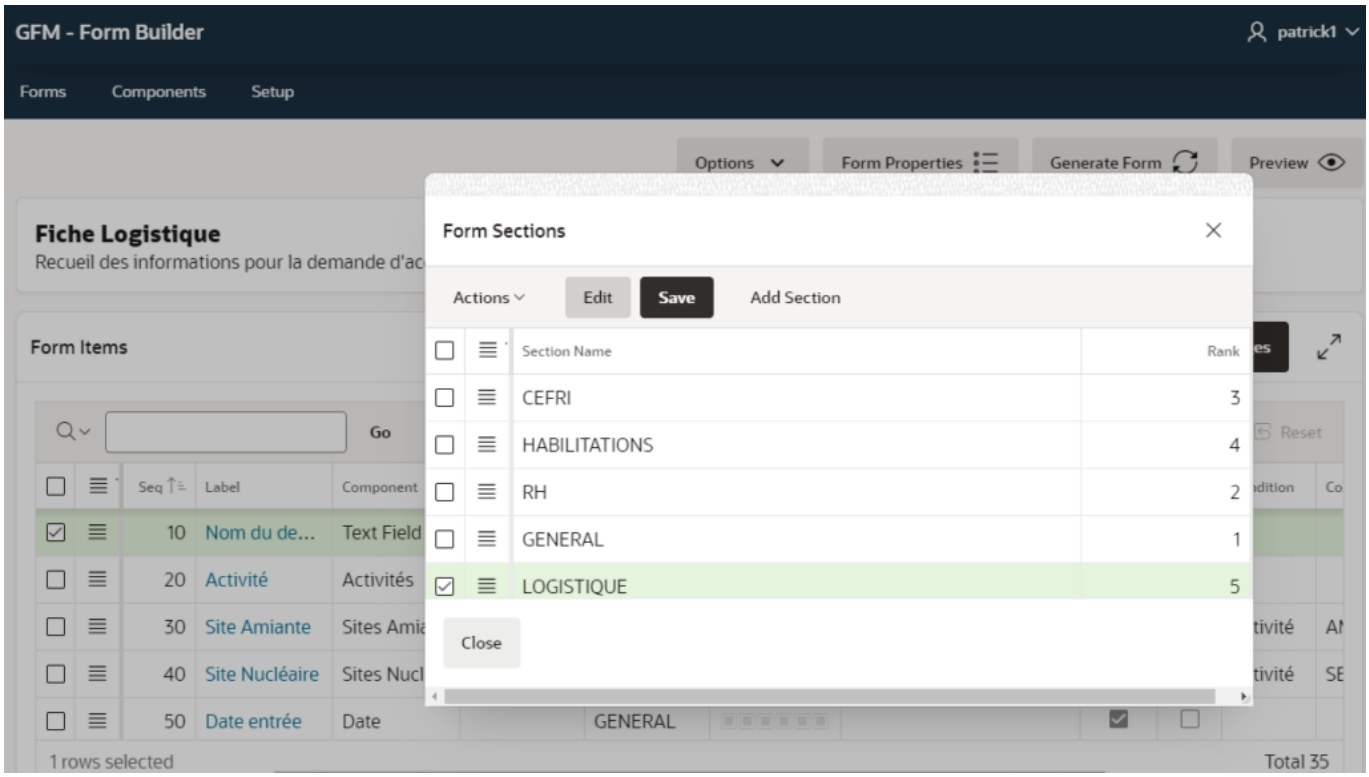
It’s possible to copy all items of an existing form into an other form.

- Inline declaration
  value and label are given the same value.
- Specification declaration in component module
- External declaration

per declaring the name of the external table or view.

Sections

A section is a logical group of items which will be rendered as either a tab or a paragraph, depending a preference.



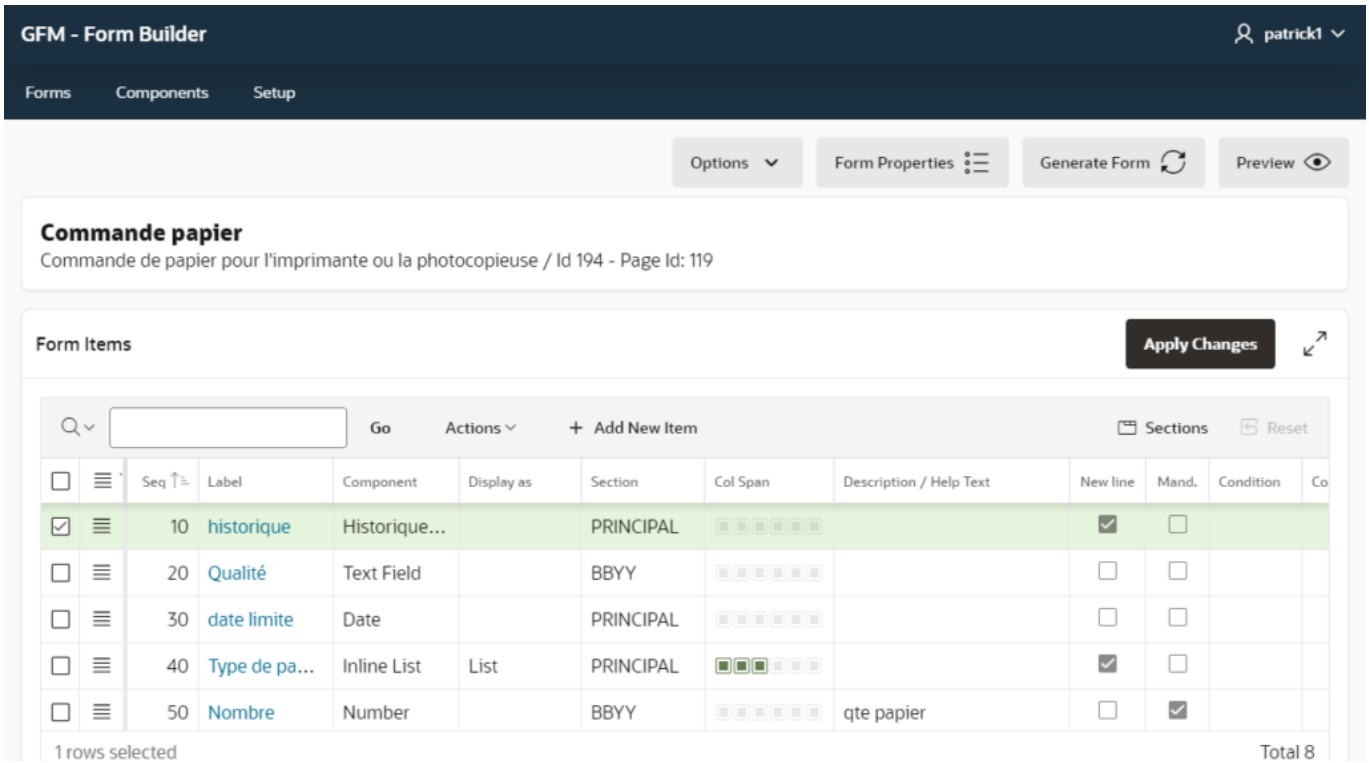
Fields are spreaded in sub-regions if there are two sections at least referenced by items. Navigation is provided by tabs, either with regular tabs or tabs and a first overall tab.

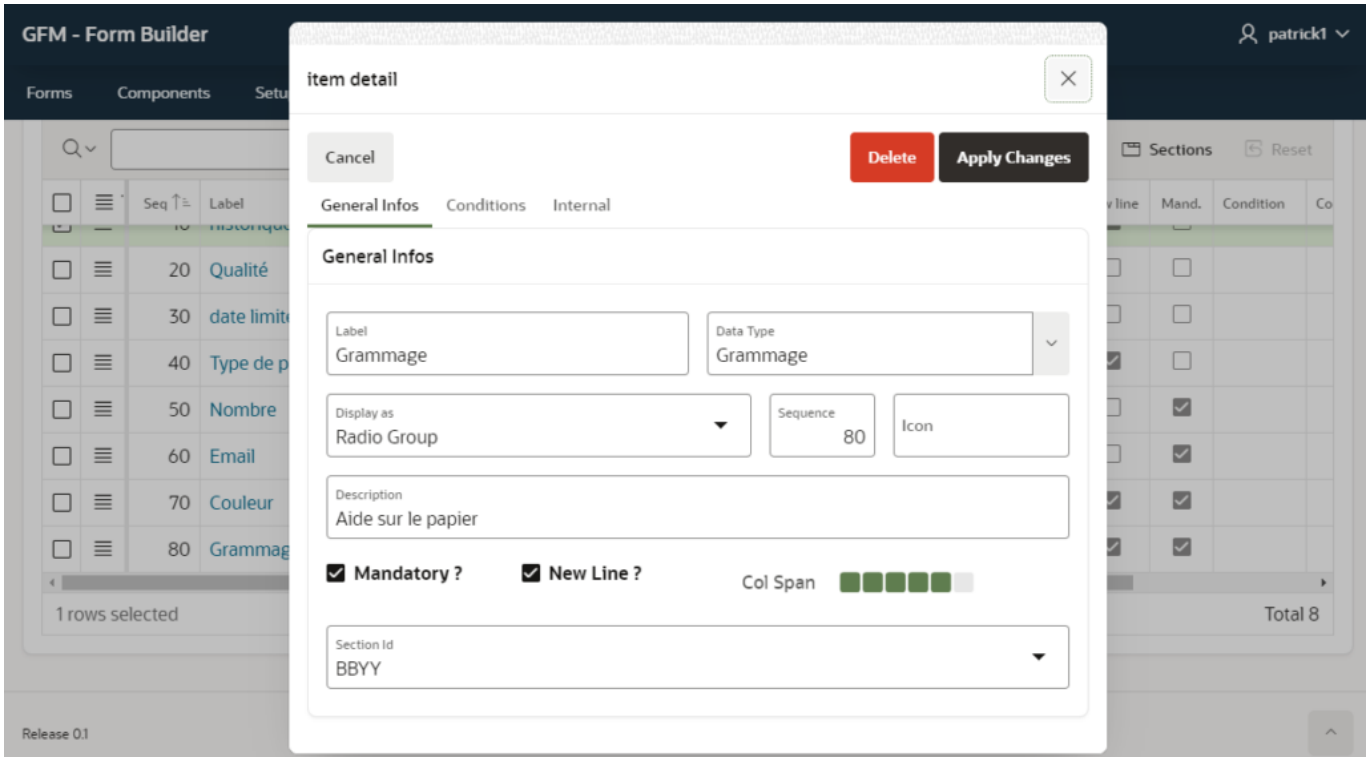


Example of navigation with preference $REGION_SECTION =DISPLAY_SELECTOR

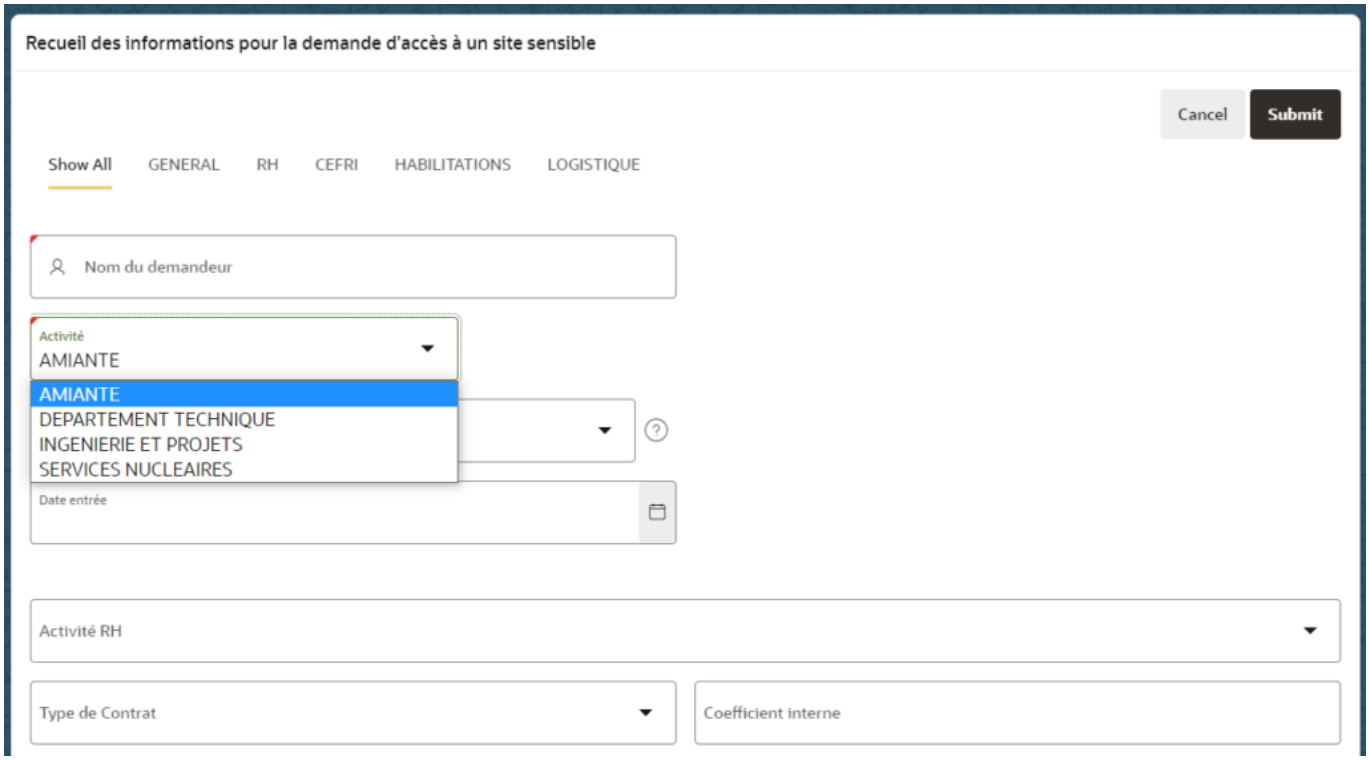
User Interface

The user interface is very classic with very minimum usage of Drag n Drop. This is a weakness if we compare with other no-code tools on the market. But on an other hand, the tool priviliges flexibility in rendering, that is not the case with « Google Forms », for instance.
At the minimum, it’s possible by Drag’n Drop to re-order displayed items.

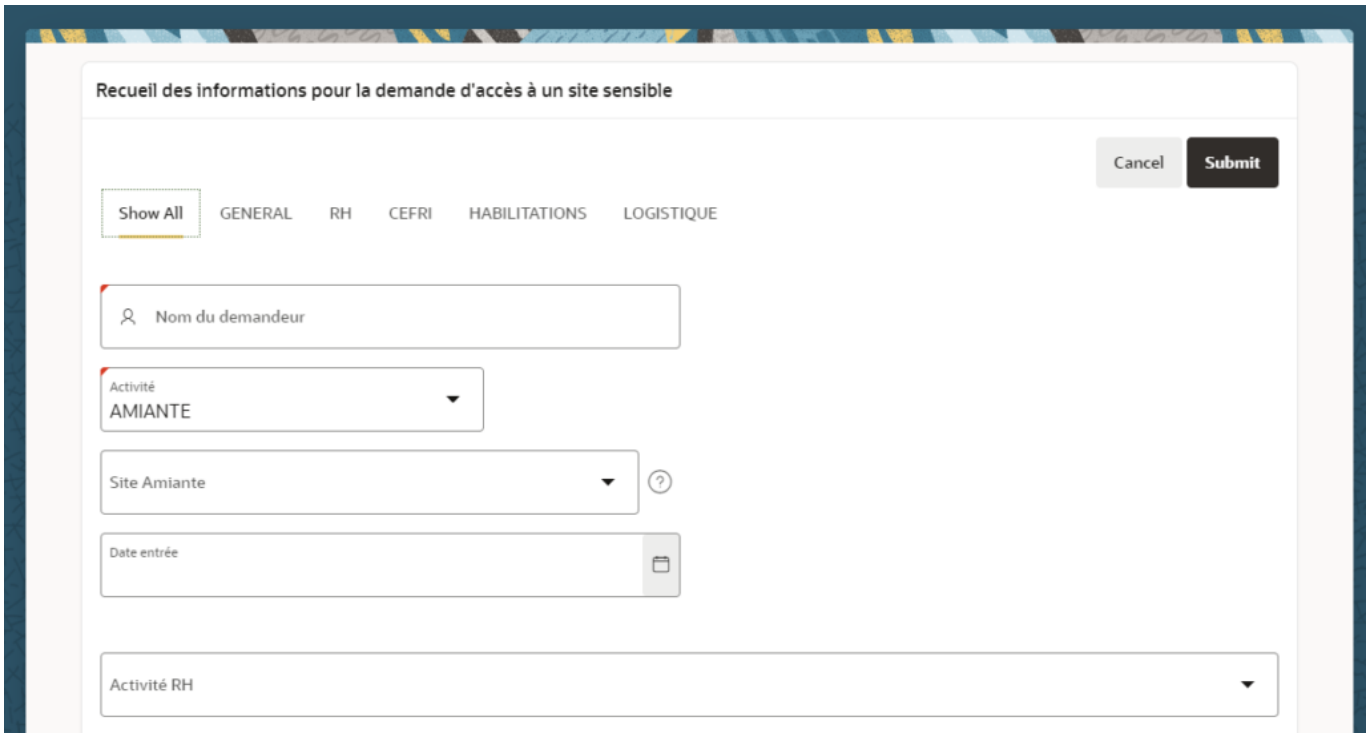



GFM - Form Builder
patrick1
Forms
Components
item detail
Cancel
Delete
Apply Changes
General Infos
Conditions
Internal
Label
Grammage
Data Type
Grammage
Display as
Radio Group
Sequence
80
Icon
Description
Aide sur le papier
Mandatory ?
New Line ?
Col Span
Section Id
BBYY
Sections
Reset
Seq
Label
Mand.
Condition
20 Qualité
30 date limit
40 Type de p
50 Nombre
60 Email
70 Couleur
80 Grammag
1 rows selected
Total 8
Release 0.1

Rendering


Recueil des informations pour la demande d'accès à un site sensible
Cancel
Submit
Show All
GENERAL
RH
CEFRI
HABILITATIONS
LOGISTIQUE
Nom du demandeur
Activité
AMIANTE
AMIANTE
DEPARTEMENT TECHNIQUE
INGENIERIE ET PROJETS
SERVICES NUCLEAIRES
Date entrée
Activité RH
Type de Contrat
Coefficient interne

Very Minimal Decoration



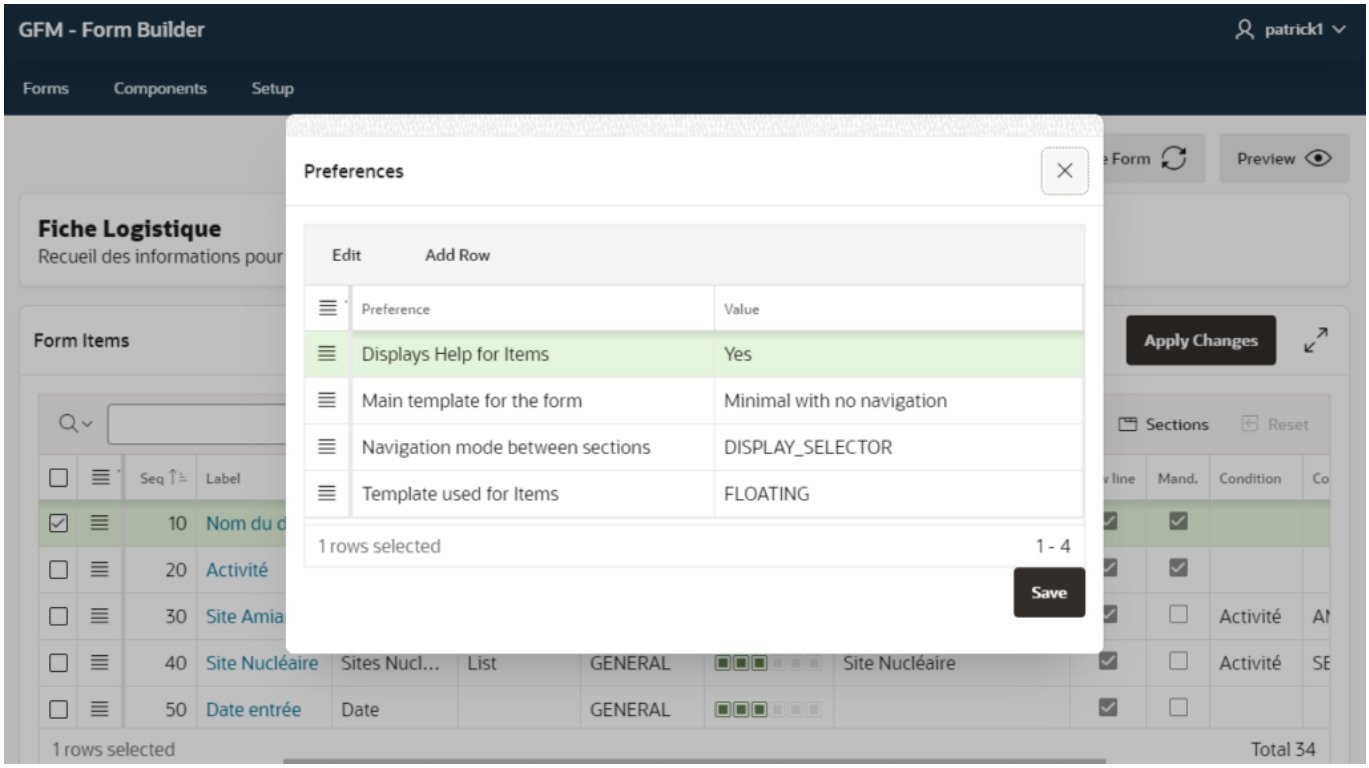
Minimal decoration

Generation options

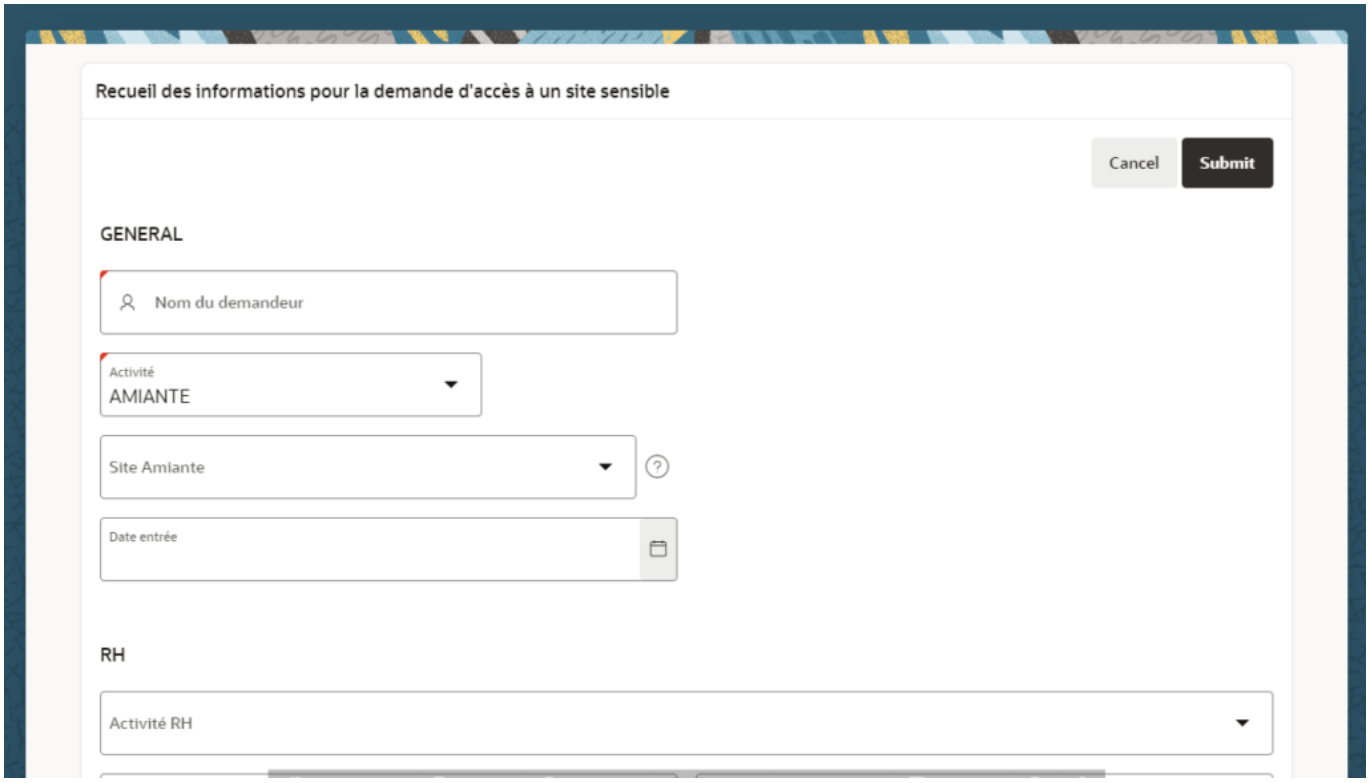
a set of preferences drives the generation process. We have reused APEX terminology in order to be in line with the regular Developer Interface.

For the moment, there are a few preferences:

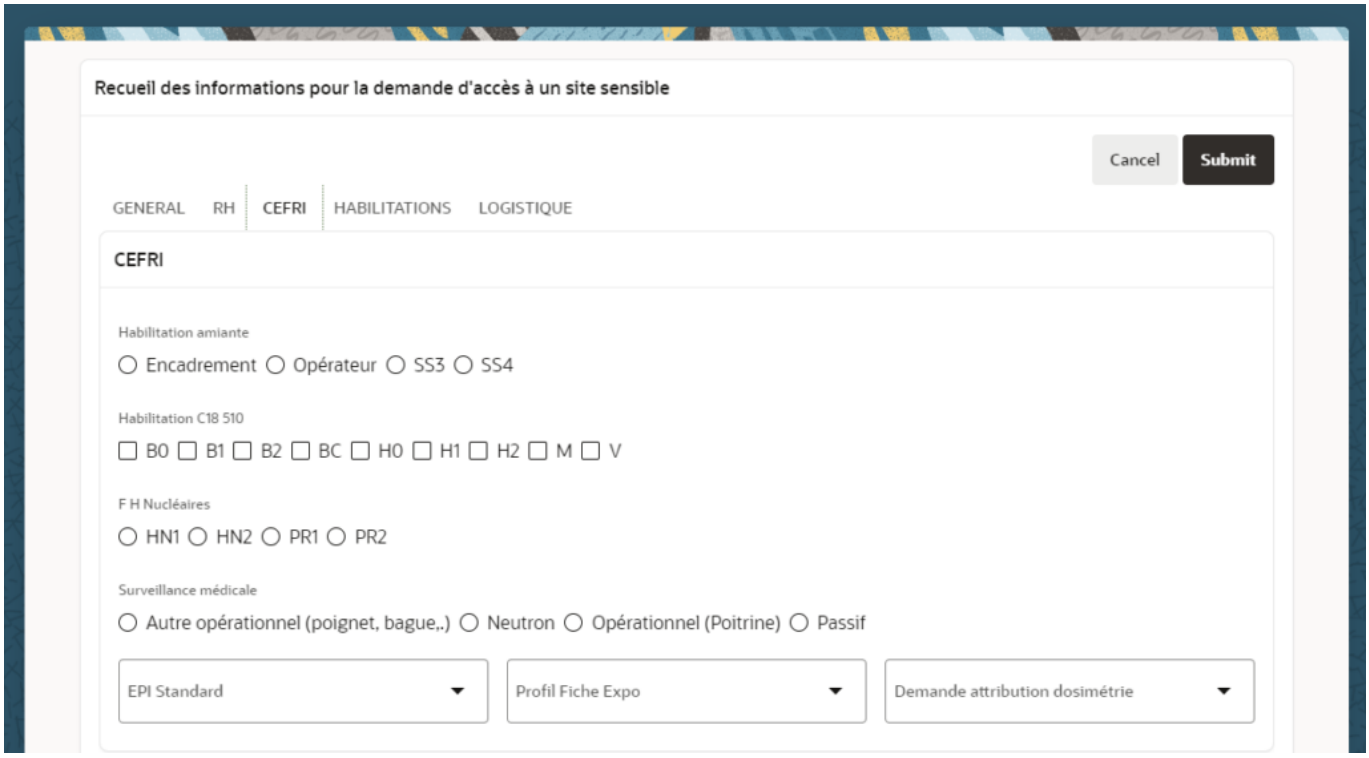
- $ITEM_TEMPLATE (FLOATING, ABOVE, BESIDE)
- $ITEM_HELP (Y,N)
- $PAGE_TEMPLATE (MINIMAL_NO_NAVIGATION)
- $REGION_SECTION (STANDARD, TABS_CONTAINER,DISPLAY_SELECTOR



Below are some variants depending on « Navigation between sections » preference



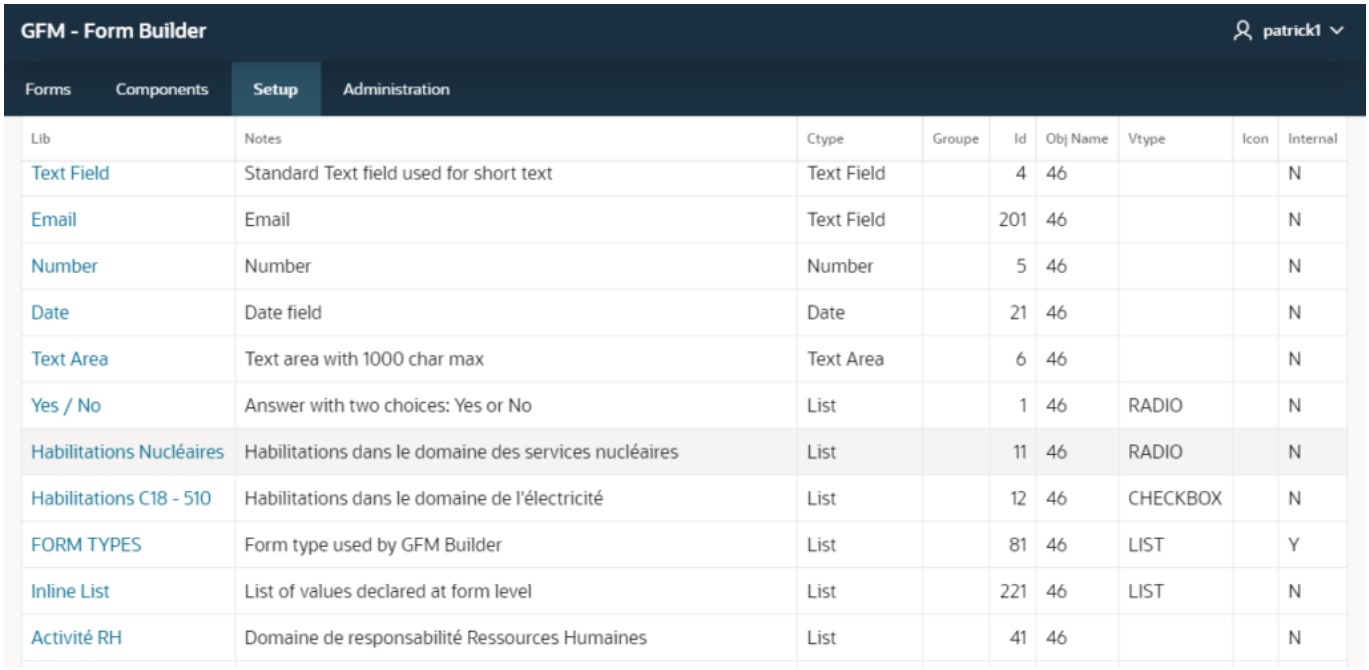
Section in a standard rendering



Sections rendered with Tabs

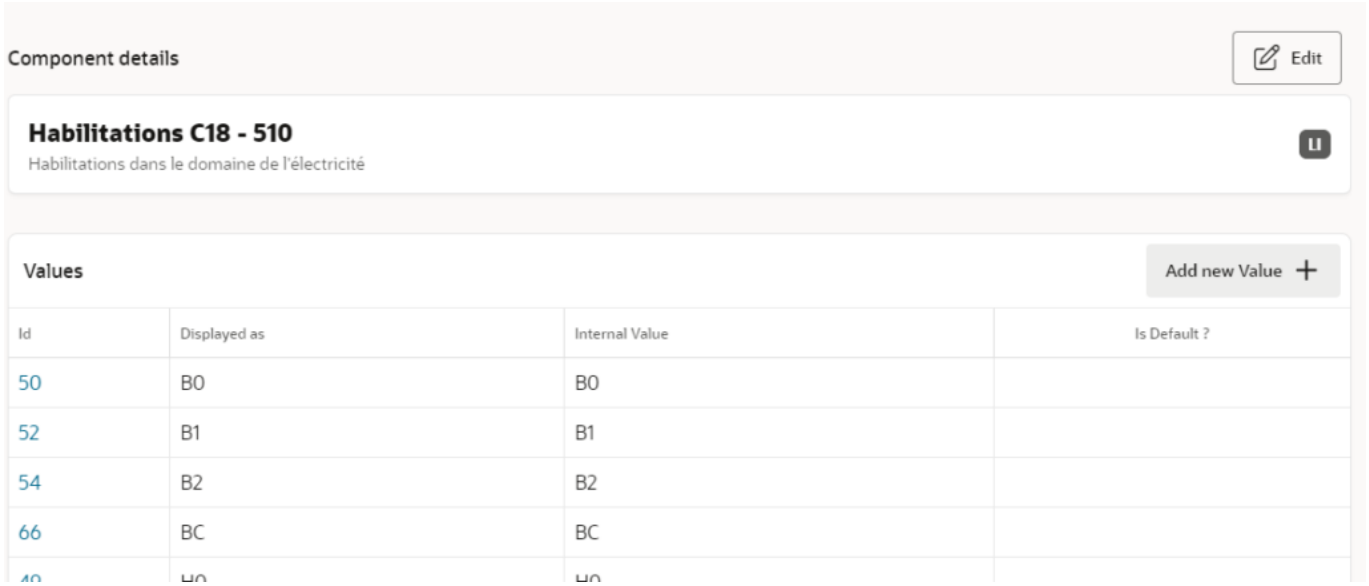
List of Values

The creation of list of values is done by creating a new component with a LIST type. Then, a second page is used for entering the values.



Lib	Notes	Ctype	Groupe	Id	Obj Name	Vtype	Icon	Internal
Text Field	Standard Text field used for short text	Text Field		4	46			N
Email	Email	Text Field		201	46			N
Number	Number	Number		5	46			N
Date	Date field	Date		21	46			N
Text Area	Text area with 1000 char max	Text Area		6	46			N
Yes / No	Answer with two choices: Yes or No	List		1	46	RADIO		N
Habilitations Nucléaires	Habilitations dans le domaine des services nucléaires	List		11	46	RADIO		N
Habilitations C18 - 510	Habilitations dans le domaine de l'électricité	List		12	46	CHECKBOX		N
FORM TYPES	Form type used by GFM Builder	List		81	46	LIST		Y
Inline List	List of values declared at form level	List		221	46	LIST		N
Activité RH	Domaine de responsabilité Ressources Humaines	List		41	46			N

List of components (The predefined lists are tagged List)



Id	Displayed as	Internal Value	Is Default ?
50	B0	B0	
52	B1	B1	
54	B2	B2	
66	BC	BC	

Entering the values for a List of Values component.

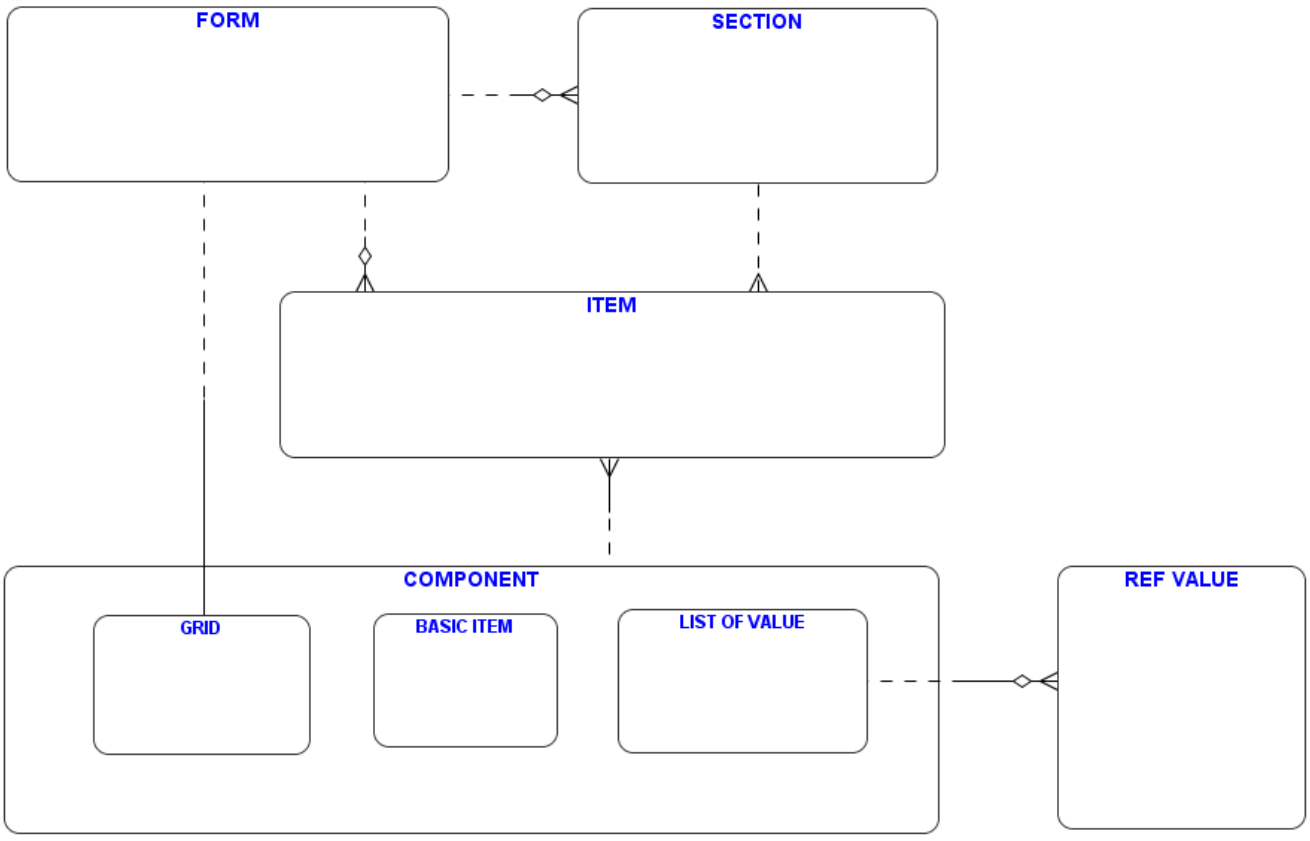
Conditionnal displaying

An item can be displayed or not depending the value of an other item. If the driving item is based upon a List, the list of values will be proposed for

choosing a specific value which will trigger displaying of slave item.

Technical design

Data Model



Implementation

- A data model for the form/Sections/Items, preferences and components
- A main data store (GFM_DATA), designed in the same way that the apex_collection table. There are 50 items at maximum.
- As many views there are forms. Each view is given an *instead* trigger.
- For each form, a specific view maps columns on given columns of the

gfm_data table.

Generating process

- Re-generate specific view
- Re-generate specific Trigger
- Generate the corresponding Page in a target Application.

The target application must belong to an arbitrary Application Group called ‘GFM Apps’ and must be choosed in the setup page. The ‘GFM Apps’ is automatically created during installation of GFM Store application.

List of Values

All the data are stored in a dedicated table: GFM_LOV.

Objects are named in Oracle database with GFM_ prefix

Lifecycle

If an item is renamed, there are no impact in the mapping

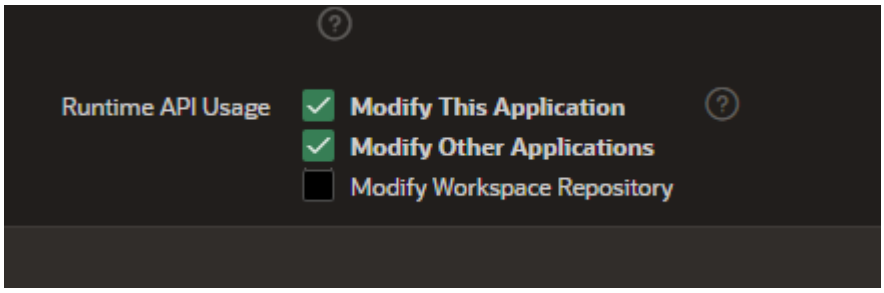
If an item is dropped, the mapped column in the GFM_DATA table is preserved, and no other item won’t be mapped on it. The reason it that we want avoid to display old data with the new item.
An option menu allow to reset data and column mapping.

Security

Before form are deployed, they must by set « Published » state.

A form can be accessible either in a anonymous mode (no need to be authenticated) or in an authenticated mode.

because GFM Builder relies intensively on internal APEX API, some important flags have been set:



A compagnon Application called ‘GFM Store’ is provided as an example of an Apps store.

Appendices

Non documented APIs used in this project

- wwv_flow_api.create_page
- wwv_flow_api.create_page_plug
- wwv_flow_api.create_page_button
- wwv_flow_api.create_page_branch
- wwv_flow_api.create_page_item
- wwv_flow_api.create_page_process
- wwv_flow_api.create_page_da_event
- wwv_flow_api.create_page_da_action
- wwv_flow_api.create_region_column
- wwv_flow_api.create_interactive_grid

- wwv_flow_api.create_ig_report_column

APEX export API have been rewritten in 22.1 (wwv_flow_imp). That means that the previous API are not garanteed to be working in the future, and therefore neither is this tool GFM Builder.

Plugins used in the project

- FOEX Drag n Drop
- FOEX Redirect and Message Actions

Author

- 

  Patrick

  GPM Factory