

# Fermat Navi - TSE (Table Structure Edit)

## VERSION

Version 1.0 (Version 0.1 was completed between March 2016 and April 2016 - This version only enables any kind of user that can login in [fermat.org](http://fermat.org) to add, modify and delete *layers*)

## TEAM

Isaias Taborda, Luis Campo, Simón Oroño

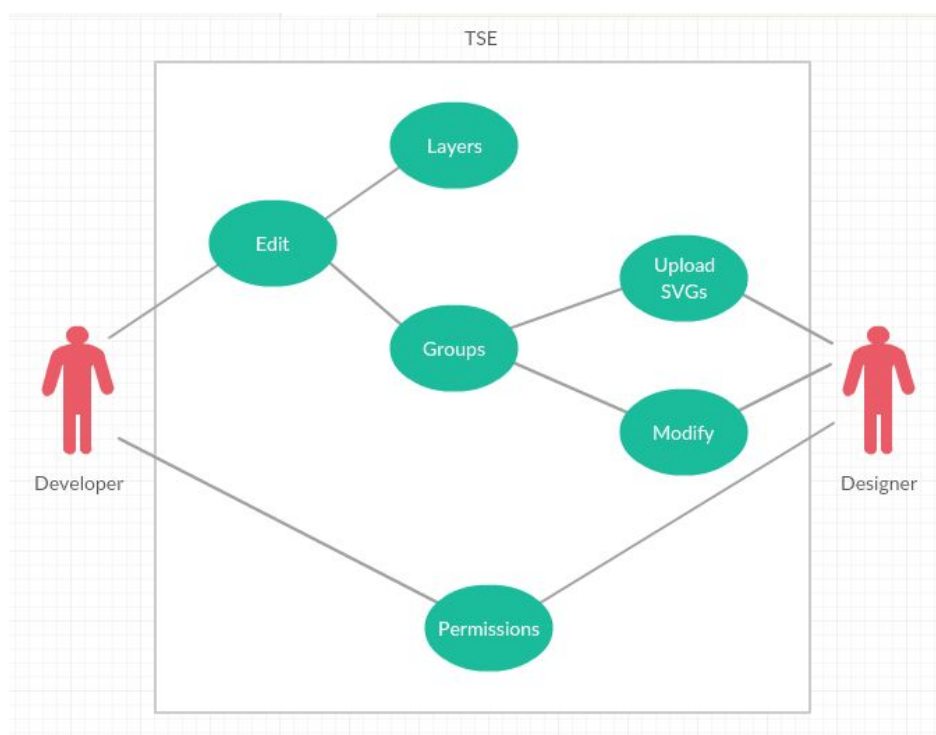
## ABSTRACT

Since it's development, [fermat.org](http://fermat.org) has shown the ecosystem of components in the Fermat network this goes from the classification of *platforms* and *superlayers* down to the description of *tiles* and the visualization of their *workflows*. With the development of Fermat ORG Editable users can add, modify and delete (the word *edit* will be used from now on to refer to these actions) components and workflows, however, to create either a component or a workflow, a *group* is needed (be that a platform or a superlayer).

A component is classified in a *group* and a *layer* according to its functionality while a workflow is classified only by a *group*. As of now, Fermat Navi Developers are in charge of the manipulation of said these structures, only requiring a small amount of data (and a couple of images in the case of a group) which is given when it is

approved. As it is right now, Fermat Developers only have access to the current TSE; while Fermat Designers send the SVGs needed to a Fermat Navi Developer, who manually transforms said images to the required file formats, who in turn, upload the new images to the web server.

This project aims to cut down the middleman by creating TSE access for groups to both Fermat Developers and Fermat Designers: Developers will be have access to a group's information while Designers will be in charge of uploading the SVGs which will be transformed and uploaded to the web server automatically. In order to guarantee more security, a permission system will be implemented in the server since an user can still try to manipulate the TSE through the server's Api. Both type of users will be able to give other users access to certain edit features, as long as the given user does not have said access and is the same type of user:



# SCOPE

## 1. Structure permission system

A permission system will be implemented in order to enhance security in the current table structure edition as well as screen development and logic structures to allow an user (Developer or Designer) to give permissions to another user of the same type. Once the user logs in, the main view will consist on a list which contains the users of the same type (either Developer or Designer) to whom he/she may give his/her permissions available (permissions in a green background are **Available** while those in a red background are **Not Available**), using the **Modify** button next to the corresponding user. The button **Back** will redirect the user back to the TSE management:

Fermat TSE			Back
Layer: <span>Add</span> <span>Edit</span> <span>Delete</span> Superlayer: <span>Add</span> <span>Edit</span> <span>Delete</span> Platform: <span>Add</span> <span>Edit</span> <span>Delete</span>			
Username	Name	Actions	
emmanuelcolina	Emmanuel Colina	<span>Modify</span>	
campol	Luis Campo	<span>Modify</span>	
miguelcldn	Miguel Celedon	<span>Modify</span>	
ricardo460	Ricardo Delgado	<span>Modify</span>	

As the logged-in user clicks that button, a form consisting on a series of check buttons, arranged according to the type of permission, will be displayed and the user will select which permissions he/she will give and use the **Submit** button:

Fermat TSE

Layer
 

☐ Add
 ☐ Edit
 ☐ Delete

SuperLayer
 

☐ Add
 ☐ Edit
 ☐ Delete

Platform
 

☐ Add
 ☐ Edit
 ☐ Delete

Submit
Cancel

The only check buttons available will be the ones matching the permissions of the logged user. Said user may use the **Cancel** button to clear the form and return to the user list.

Size: 50%


## 2. Layer management

Screen development and logic structures to allow a Developer type user to create, modify and delete layers. Once the user logs in, the main view will consist on a list which contains all the layers in the table structure. The user may add, modify or delete layers using the corresponding buttons (**Add, Modify & Delete**). A drop down button will allow the user to switch between layer management and group management (platform or superlayer) and the button **Permissions** will redirect the user to the structure permission system:



Name	Lang	Superlayer	Order	Actions
Core	Java		0	<button>Modify</button> <button>Delete</button>
Niche Wallet	Java-android		1	<button>Modify</button> <button>Delete</button>
Reference Wallet	Java-android		2	<button>Modify</button> <button>Delete</button>
Sub App	Java-android		3	<button>Modify</button> <button>Delete</button>
Desktop	Java-android		4	<button>Modify</button> <button>Delete</button>
Engine	Java		5	<button>Modify</button> <button>Delete</button>

To add a layer, the user will be shown a form to fill with the layer's name, select which of the available programming languages, whether it is in a superlayer or not and the relative position within all the layers in the table structure:


Fermat TSE
Permissions
Layer
Add

**Name:**


**Lang:**

**In Superlayer:**

**Position:**

Submit
Cancel

To modify a layer, the same form used to add layers will be shown filled with the information from the layer to be modified and the current relative position will be indicated:


Fermat TSE
Permissions
Layer
Add

**Name:**

**Lang:**

**Currently:** Below - Niche Wallet (In no Superlayer)

**In Superlayer:**

**Position:**

Submit
Cancel

If the user tries to delete a layer, a prompt will show a warning and ask for confirmation:

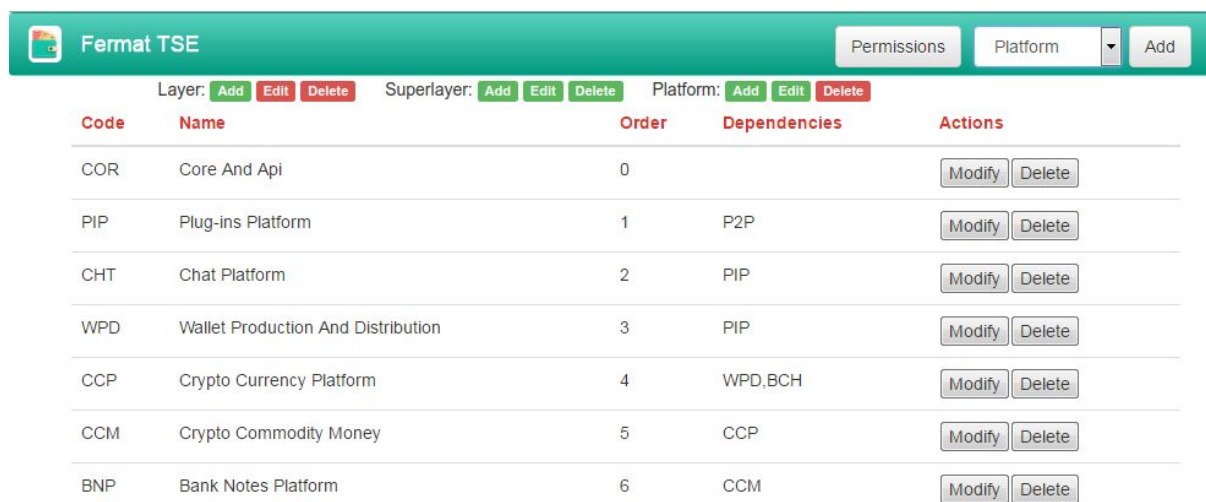
Are you sure you want to remove the layer: Core? (Removing this layer will delete all of it's associated components)

Aceptar
Cancelar

Size: 15%

### 3. Group management

Screen development and logic structures to allow a certain edition features depending on the type of user: Developers will be able to create a group without its icon and modify or delete it and Designers will be able to create a group with only its icon, name and code and modify a group's icon. Once the user uses the drop down button, mentioned in **Layer management**, the list shown will be updated accordingly to the group selected (either platform or superlayer). The view will follow the same pattern used for the buttons displayed:



Code	Name	Order	Dependencies	Actions
COR	Core And Api	0		<button>Modify</button> <button>Delete</button>
PIP	Plug-ins Platform	1	P2P	<button>Modify</button> <button>Delete</button>
CHT	Chat Platform	2	PIP	<button>Modify</button> <button>Delete</button>
WPD	Wallet Production And Distribution	3	PIP	<button>Modify</button> <button>Delete</button>
CCP	Crypto Currency Platform	4	WPD,BCH	<button>Modify</button> <button>Delete</button>
CCM	Crypto Commodity Money	5	CCP	<button>Modify</button> <button>Delete</button>
BNP	Bank Notes Platform	6	CCM	<button>Modify</button> <button>Delete</button>

To add a layer two forms will be made. If the user is a Developer, he/she will be shown a form to fill with the group's code and name, select which of the available groups in the table structure are dependencies and the relative position within all the group type (either platforms or superlayers) in the table structure:

**Fermat TSE** Permissions Platform Add

**Code**

**Name**

**Position:**

Before

**Dependencies:**

- COR - Core And Api
- PIP - Plug-ins Platform
- CHT - Chat Platform
- WPD - Wallet Production And Distribution
- COP - Core Open Platform

Submit Cancel

If the user is a Designer he/she will be shown a form to fill with the group's code and name and select the SVG files that correspond to the group being added:

**Fermat TSE**

**Code**

**Name**

**Header SVG**

Choose File No file chosen

**Icon SVG**

Choose File No file chosen

Submit Cancel

Once a Developer or Designer adds a group, an user of the other type will need to fill the rest of the information needed to completely add the group. As the group will be featured in the group's list, the second user will only need to use the **Modify** button and fill its corresponding form. This option will also be available for any group to be able to modify any kind of information a certain type of user can. If the logged-in user tries to delete a group, a prompt will show a warning and ask for confirmation:

Are you sure you want to remove the platform: Crypto Commodity Money? (Removing this layer will delete all of it's associated components)

Aceptar

Cancelar

Size: 35%

# EVALUATION

The following requirements will be evaluated during the beta testing:

## 1. Structure permission system

- All users will have a permission code established.
- All users will have be classified as either a Developer or a Designer.
- Users will be able to visualize which structures can they edit.
- Users will be able to give permissions to another user of the same type.
- An user will be able to visualize the users of the same type to whom he/she may give permissions.
- An user will only be able to give the exact same permissions he/she has.

## 2. Layer management

- Visualize all the existing layers.
- Add new layers if the user has enough permission to do so.
- Modify existing layers if the user has enough permission to do so.
- Delete existing layers if the user has enough permission to do so.
- Show an error message if an user tries to perform an action for which he/she has not enough permissions.

## 3. Group management

- Visualize all the existing groups (platforms & superlayers).



- Add new groups without their icon if the user is a Developer and has enough permission to do so.
- Add new groups with only their icon, name and code if the user is a Designer and has enough permission to do so.
- Modify a group's information if the user is a Developer and has enough permission to do so.
- Modify a group's icon if the user is a Designer and has enough permission to do so.
- Delete existing groups if the user has enough permission to do so.
- Show an error message if an user tries to perform an action for which he/she has not enough permissions.

# TERMS AND CONDITIONS

1. The team agrees that the implementation project has two stages: functionality and beta testing.
2. The team understands and accepts that the functionality will be considered done when all the features described in the scope of this agreement are completed and tested in an alpha stage.
3. The team understand and accepts that the implementation of functionality must follow the Fermat's technical guidelines.
4. Component architecture and workflows are created in the Analytics System and the Interfaces in the API library of the platform involved.
5. The team understands and accepts that implementation will be evaluated by the @bounty-program-team and will include GUI/UX design checking, functionality test and review of Fermat's technical guidelines compliance.
6. The team understands and accepts that there is only one free review for functionality and one for Fermat's technical guidelines compliance. The following

reviews will cost the team 25% of the related bounty each in case the first one wasn't approved.

7. The team agrees to complete the implementation on the following conditions:

- **Implementation bounty:** The functionality will be 70% of the total bounty. This bounty will be awarded to the development team when the @bounty-program-team considers that the functionality delivered is done.
- **Implementation due date:** All the features will be finished before **08/06/2016**. If on this date the development team does not deliver the agreed functionality, it will lose the implementation bounty.
- **Implementation collateral deposit:** The team agrees to deposit the amount of **\$1.250** paid in tokens in favor of the Fermat Foundation, as a collateral to be lost if this part project is not approved before the due date.
- **Implementation margin:** No more penalties apply 7 calendar days after implementation due date.
- **Implementation penalty:** 5% of the implementation bounty for each calendar day that elapses after the implementation margin without formal acceptance from the @bounty-program-team. This penalty will be paid by the development team from its savings to the Fermat Foundation. If savings are not enough it will be deducted from their cash salaries.

8. The team understands and accepts that beta testing will be conducted by the @beta-testing-team.

9. The team understands and accepts that criteria to pass beta testing are:

- A. No bug issues on beta testing due date,
- B. Or no bug issues in a period of three (3) consecutive calendar days before the due date,
- C. Or no answer from @beta-testing-team about solved bugs for (6) consecutive days before the due date.

10. The team agrees to complete the beta testing on the following conditions:

- **Beta testing bounty:** The beta testing bounty will be a fixed 30% of the total bounty. It could be awarded to the development team if it passes the beta testing on time or by @beta-testing-team if it fails. Also, it implies that development team will not get this bounty unless it succeeds in the beta testing process.
- **Beta testing due date:** Beta testing will be passed before **23/06/2016**. If on this date the development team does not pass the beta testing, it will lose the beta testing bounty, which will be automatically awarded to the @beta-testing-team.
- **Beta testing collateral deposit:** The team agrees to deposit the amount of **\$500** in Fermat tokens in favor of the Fermat Foundation, as a collateral to be lost if this part of the project is not approved before the due date.
- **Beta testing margin:** No more penalties are applied **7** calendar days after the beta testing due date.
- **Beta testing penalty:** **3%** of the bounty for each calendar day that elapses after the due date without formal passing through beta testing. This penalty will be paid by the development team from the implementation bounty previously awarded or its savings to the beta testing team. If savings are not enough it will be deducted from their cash salaries.

## TOTAL BOUNTY

The total amount of the bounty in Fermat tokens for this project is **\$5.000 (\$3.500 for implementation + \$1.500 for the beta testing)**.

## DISTRIBUTION OF BOUNTY BY CONTRIBUTOR

Isaias Taborda: 33.33%

Luis Campo: 33.33%

Simón Oroño: 33.33%

**Total amount deposit: \$1.750**

## **SUMMARY**

<b>Implementation due date</b>	<b>08/06/2016</b>
<b>Implementation collateral</b>	<b>\$1.250,00</b>
<b>Implementation margin (days)</b>	<b>7</b>
<b>Implementation penalty (%/day)</b>	<b>5</b>
<b>Functionality review (attempts-25%)</b>	<b>1</b>
<b>Technical review (attempts-25%)</b>	<b>1</b>
<b>Implementation bounty (\$)</b>	<b>\$3.500,00</b>
<b>Beta testing due date</b>	<b>23/06/2016</b>
<b>Beta testing collateral (\$)</b>	<b>\$500,00</b>
<b>Beta testing margin (days)</b>	<b>7</b>
<b>Beta testing penalty (%/day)</b>	<b>3</b>
<b>Beta testing bounty (\$)</b>	<b>\$1.500,00</b>
<b>Total bounty (\$)</b>	<b>\$5.000,00</b>