# USAGE GUIDE FOR PeNDeS

## Petri Net Definition:

A Petri net is defined as a triple (P, T, F) where:

● P is a finite set of *places*

● T is a finite set of *transitions* (P ∩ T = ∅)

● F ⊆ (P x T) ∪ (T x P) is a set of *arcs* (flow relation) {for short we will write f(p→t) to

describe an arc that connect transition t to place p}

Additionally, *marking* M ∈ P → Z\* is a function that assigns a non-negative integer to every

place and represents the state of the net (some definitions call this a marked petri net). M(p)

denotes the marking of place p. *Inplaces* of a transition (\*t) is a set of places where each

element of a set is connected to the transition (the place is the source of the arc and the

transition is the destination). Conversely, *outplaces* of a transition (t\*) is a set of places that are

connected to the transition by *arcs* where the places are the destinations and the transition is

the source.

The following definitions cover how the petri net progress from one marking to another:

● t ∈ T is *enabled* if ∀ p ∈ P | ∃ f(p→t) ∈ F M(p) > 0 - for all *inplaces* of the transition

(that are connected to the transition via an incoming arc) the amount of tokens at the

place is non zero

● *Firing* an enabled transition decreases the amount of tokens on all *inplaces* with one and

increases the amount of token in all *outplaces* of the transition by one.

**Classifications**

We recognize the following features of a petri net:

● *Free-choice petri net* - if the intersection of the inplaces sets of two transitions are not

empty, then the two transitions should be the same (or in short, each transition has its

own unique set if *inplaces* )

● *State machine* - a petri net is a state machine if every transition has exactly one *inplace*

and one *outplace* .

● *Marked graph* - a petri net is a marked graph if every place has exactly one out transition

and one in transition.

● *Workflow net* - a petri net is a workflow net if it has exactly one source place s where \*s

= ∅, one sink place o where o\* = ∅, and every x ∈ P ∪ T is on a path from s to o.

## Typical use cases:

Petri Net use cases include Behaviorial analysis of system like state machines , Communication protocol verification etc.

* The usage of the petri net Studio is explained below. First go through the examples and select the one that suits your design and use it as existing seed.
* The studio provides designing the petri nets using places and transitions and Arcs
* Once the design is done we can run the classifier to classify petrinets into 4 categories like free choice, state machine, marked graph and workflow types.
* We can visualize using the plotView visualizer
* Current visualizer version only provides static visualization and does not provide firing and other dynamic visualization which may be updated in future.

## Installation for PeNDeS:

Use google for Help to install the below prerequisite software for PeNDeS:

1) install node

2) install mongo db

3) install git

4) install python

## B. Install the following node modules:

npm install -g webgme-cli

mkdir CS6388-01-PeNDeS

Git clone the repository

https://github.com/nithinkumar030/CS6388-01-PeNDeS

cd CS6388-01-PeNDeS

Install the following node modules:

npm install

npm install --save lodash

npm install --save jointjs

python -m pip install webgme-bindings

npm install webgme-bindings --save

Try this for vscode errors during bindings --save

npm install --global windows-build-tools --vs2015

npm config set msvs\_version 2015

Finally start the WebGME server:

node ./app.js

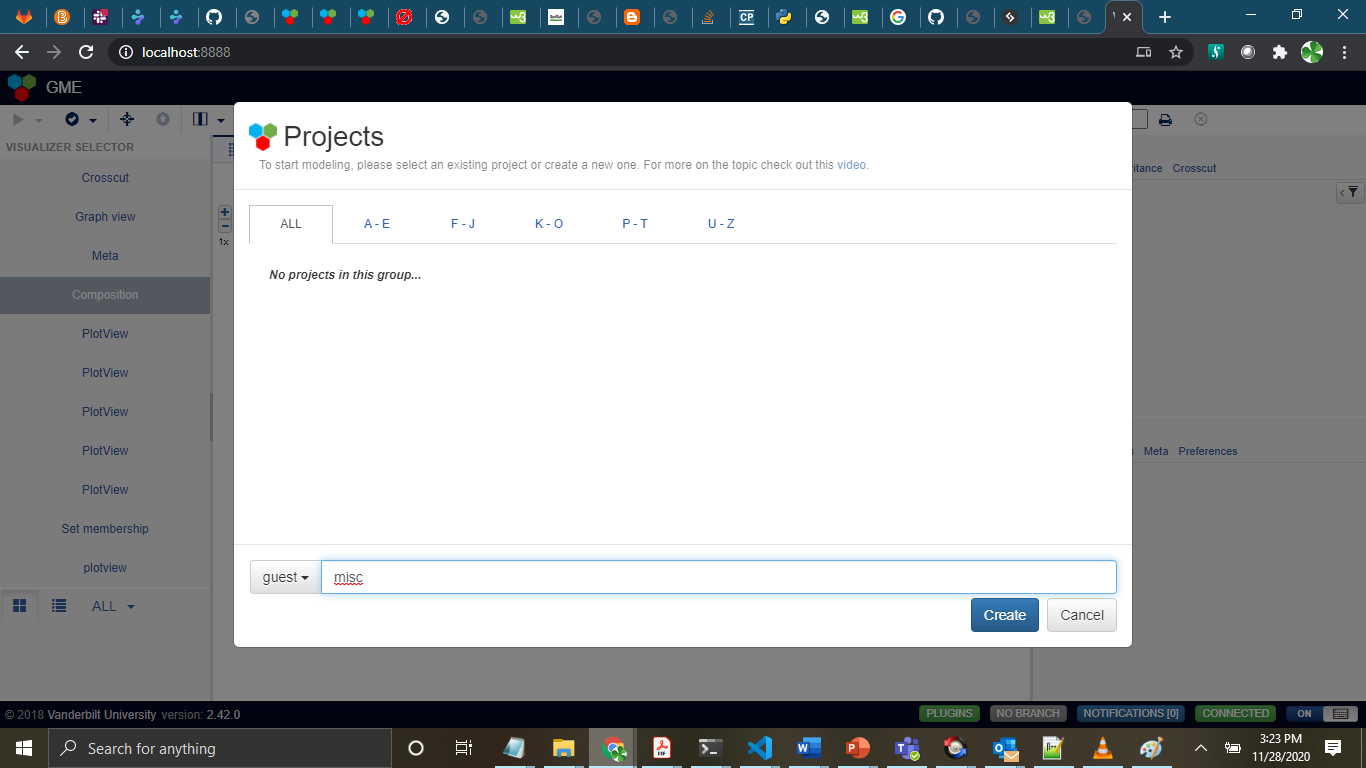
Use a browser and open using below link to open the design studio:

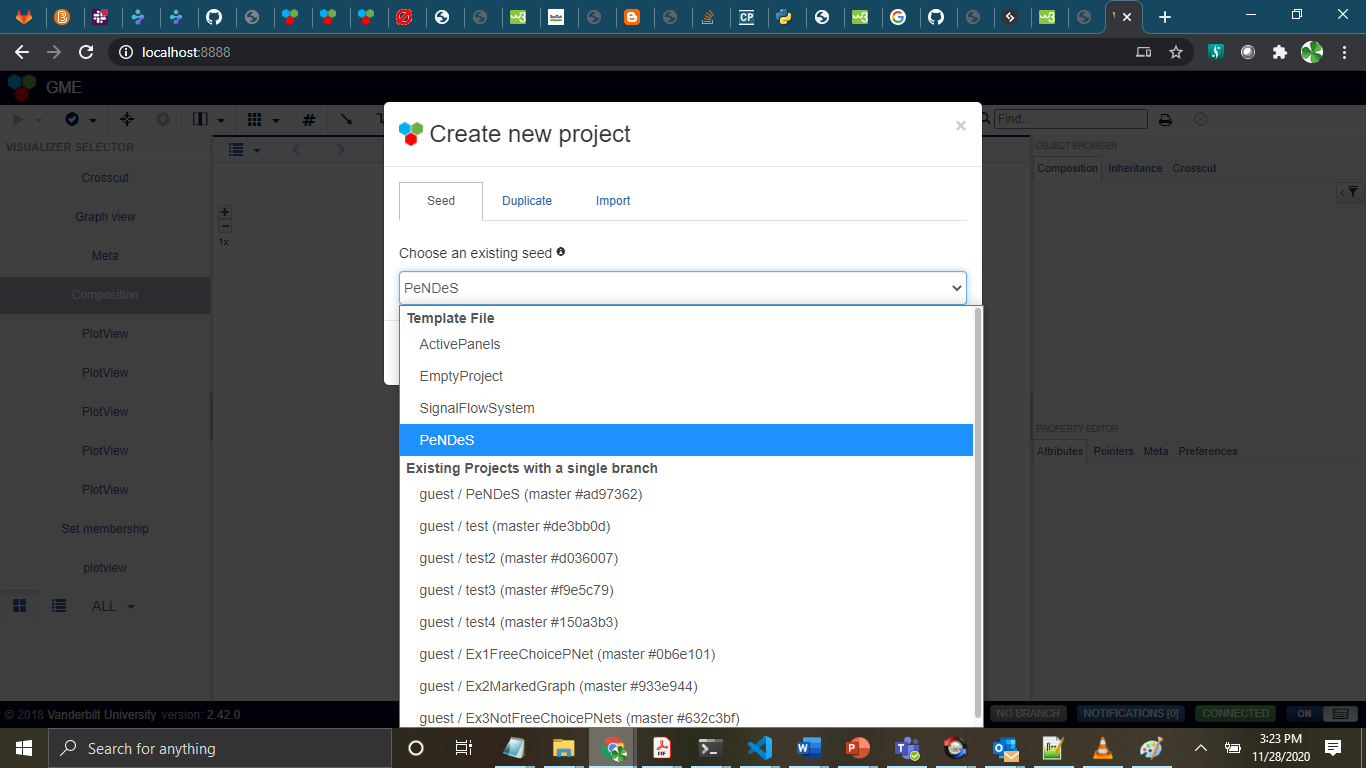
http://localhost:8888/

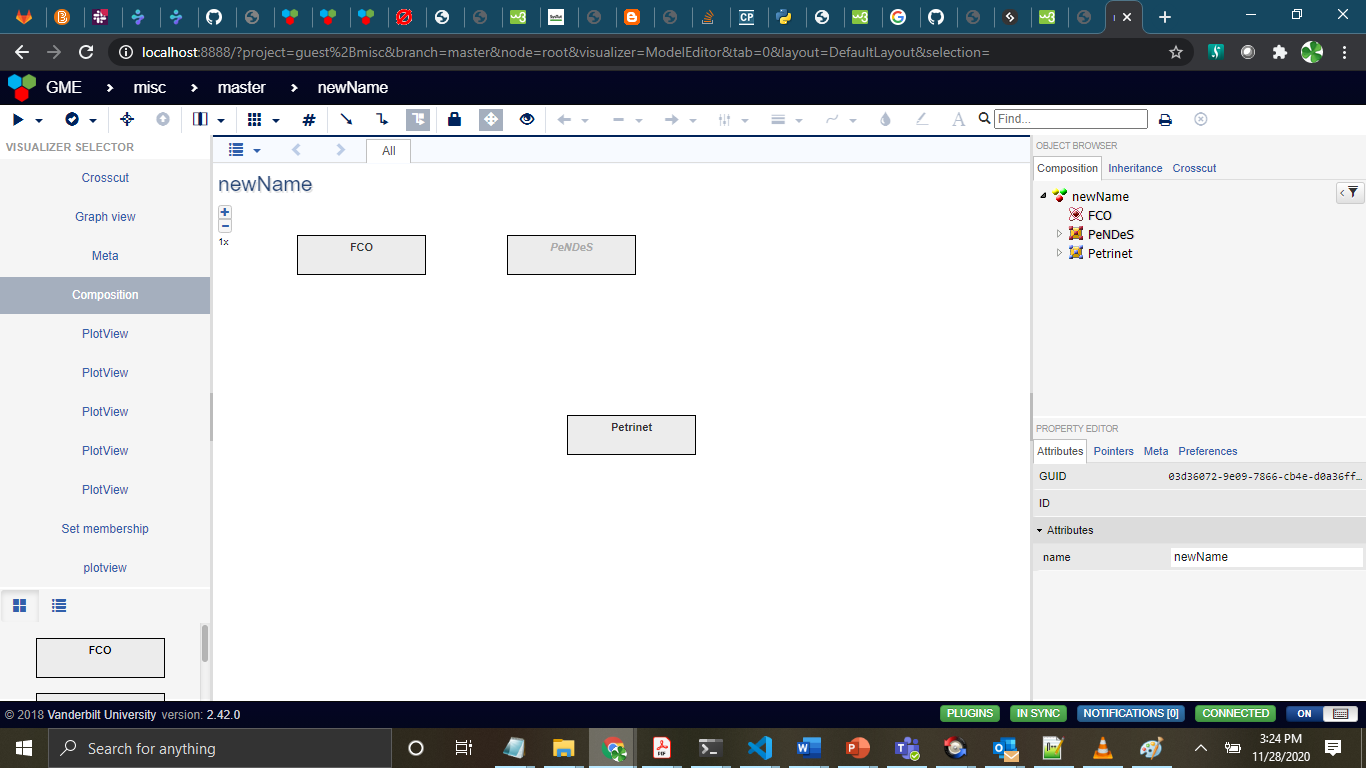
# Design studio Usage:

Click on the GME icon and open the project meu

You can create a new project : Type the name and use the PeNDes as seed.



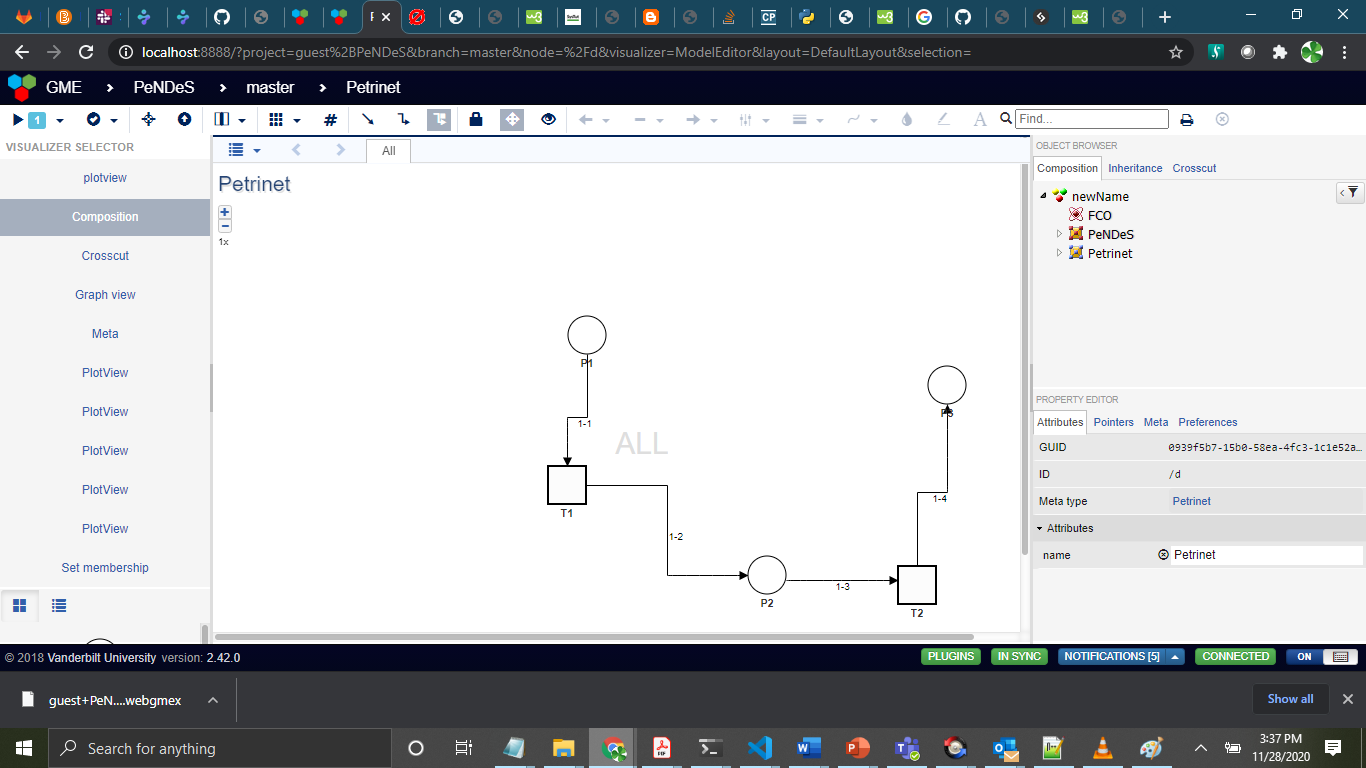


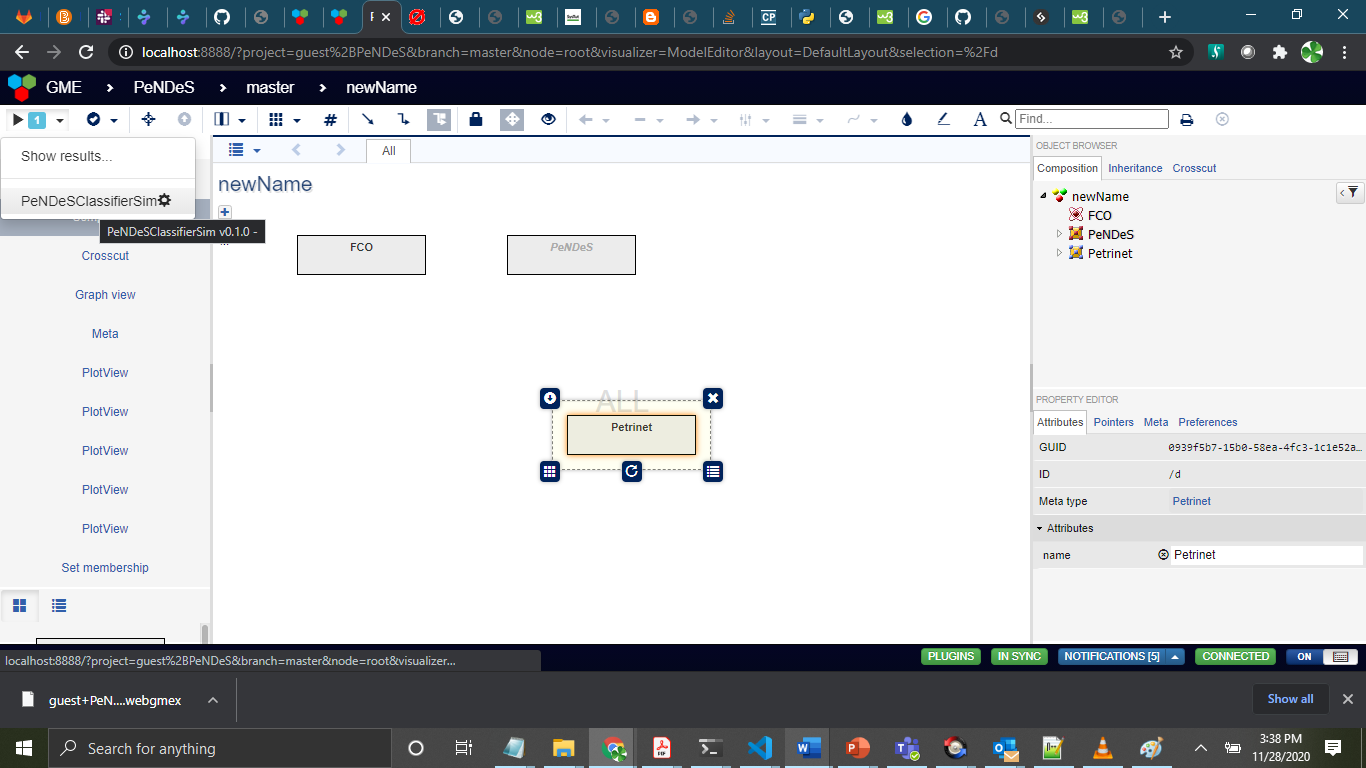


Landing page: Click on the Petrinet icon and

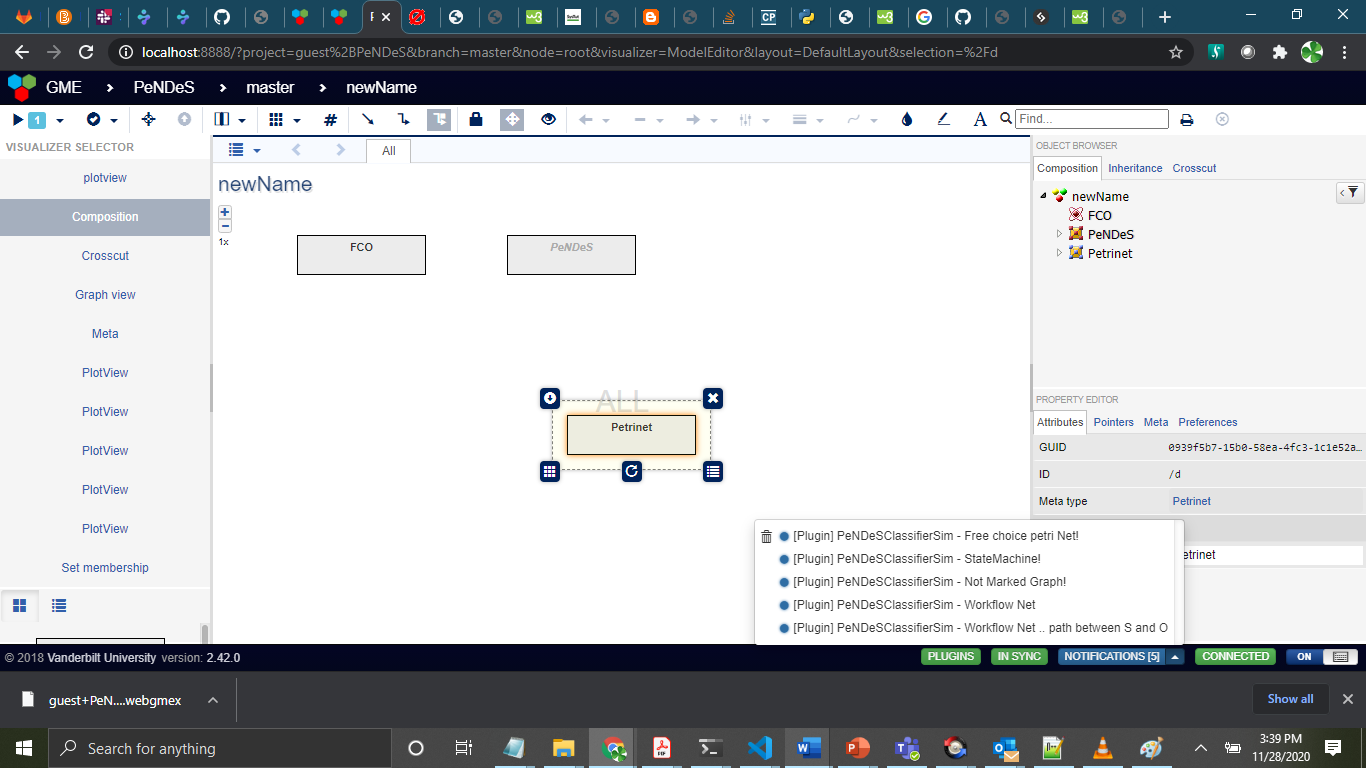
This is the default petrinet editor. You can customize editing the nodes with different names

Edit the default Petri Nets to create your custom one

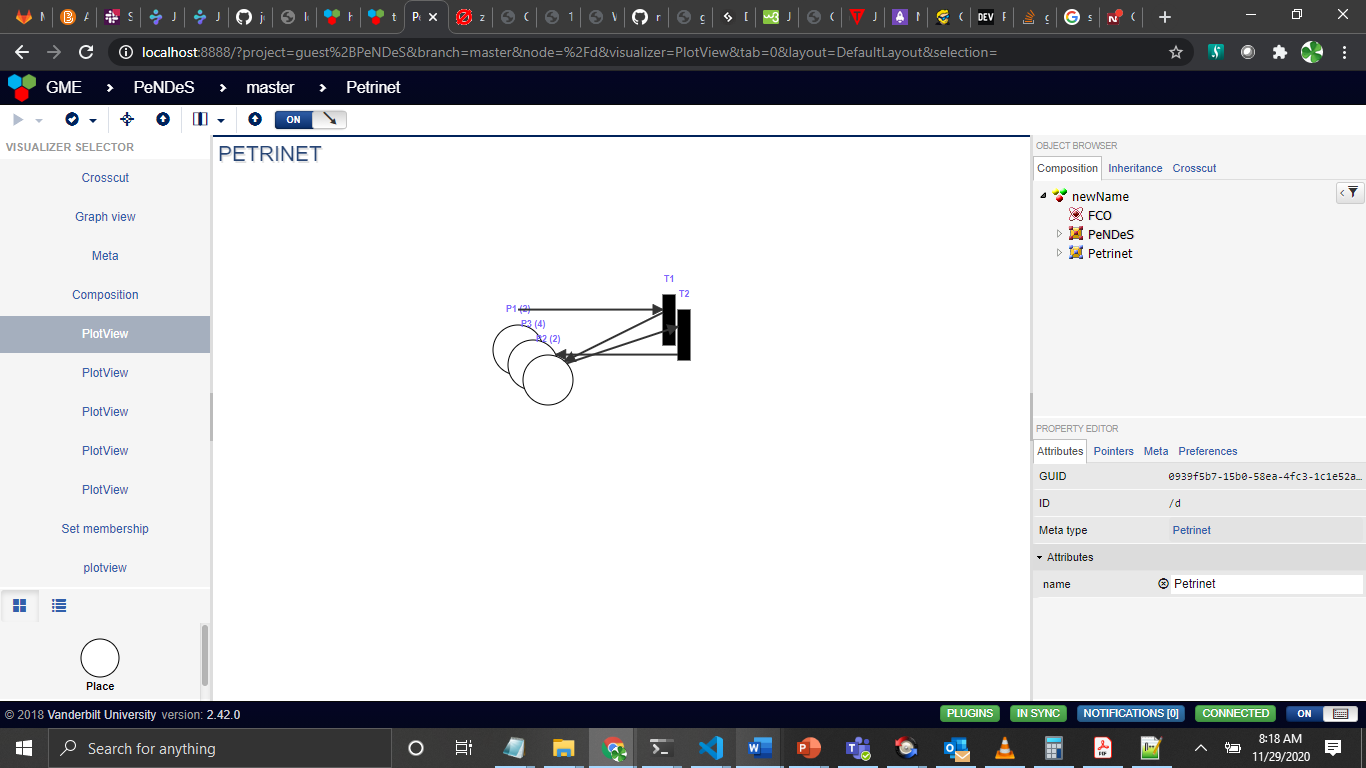


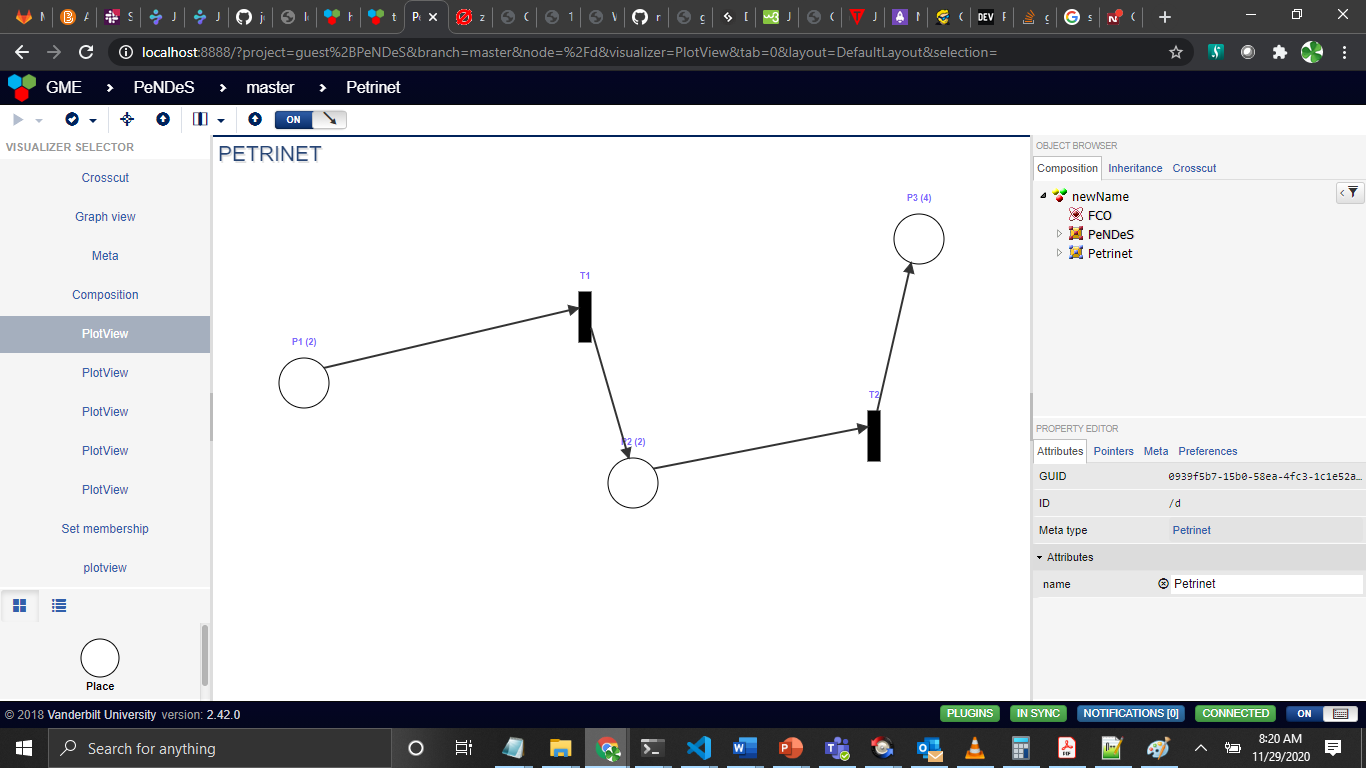


Run the petri net classifier to understand the type it belongs



Result of the petri net classification as notifications

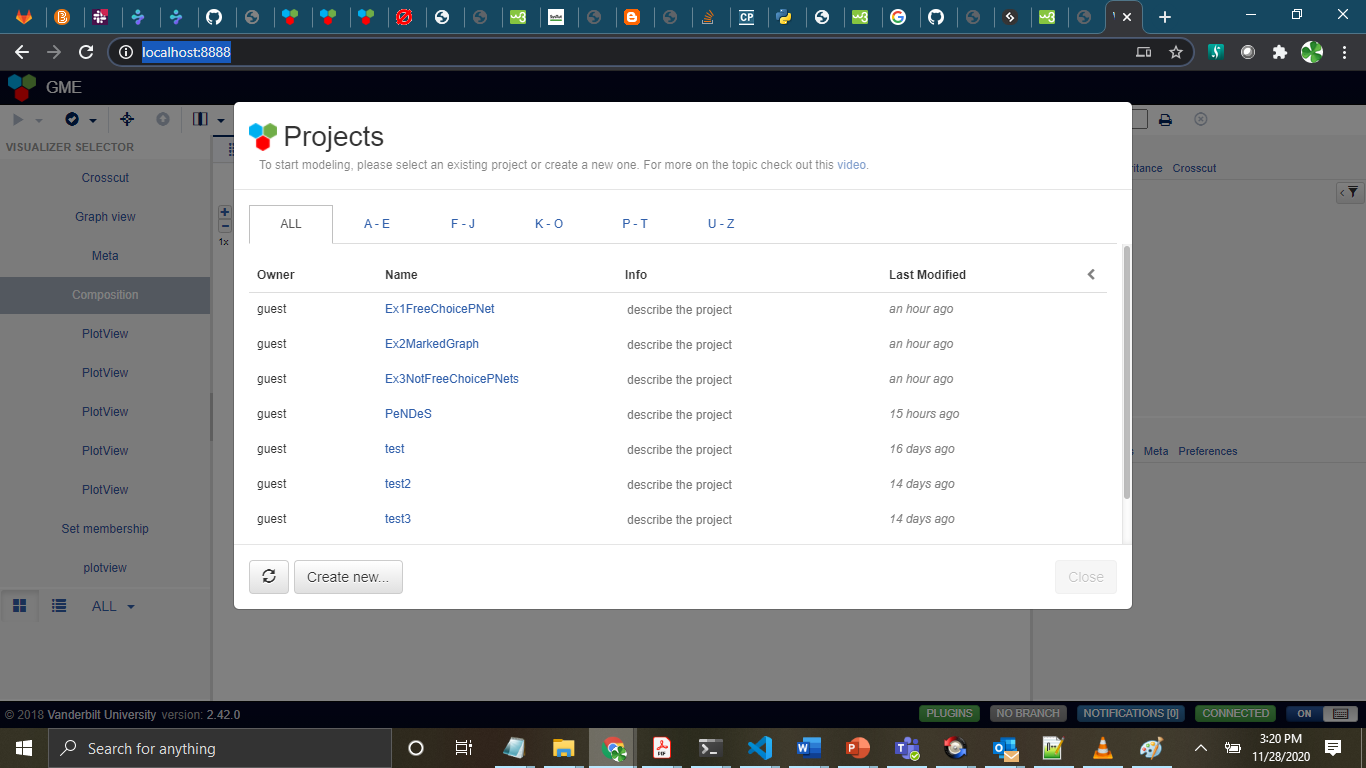


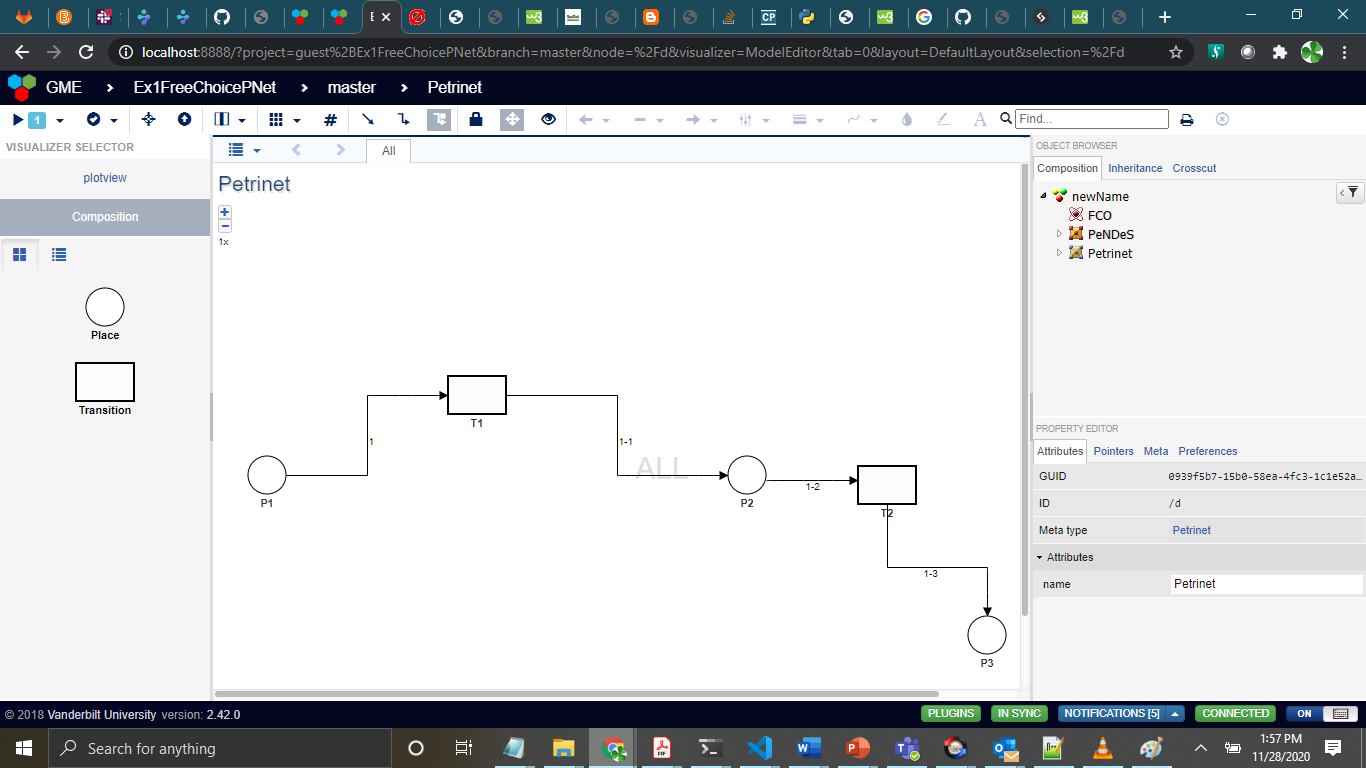


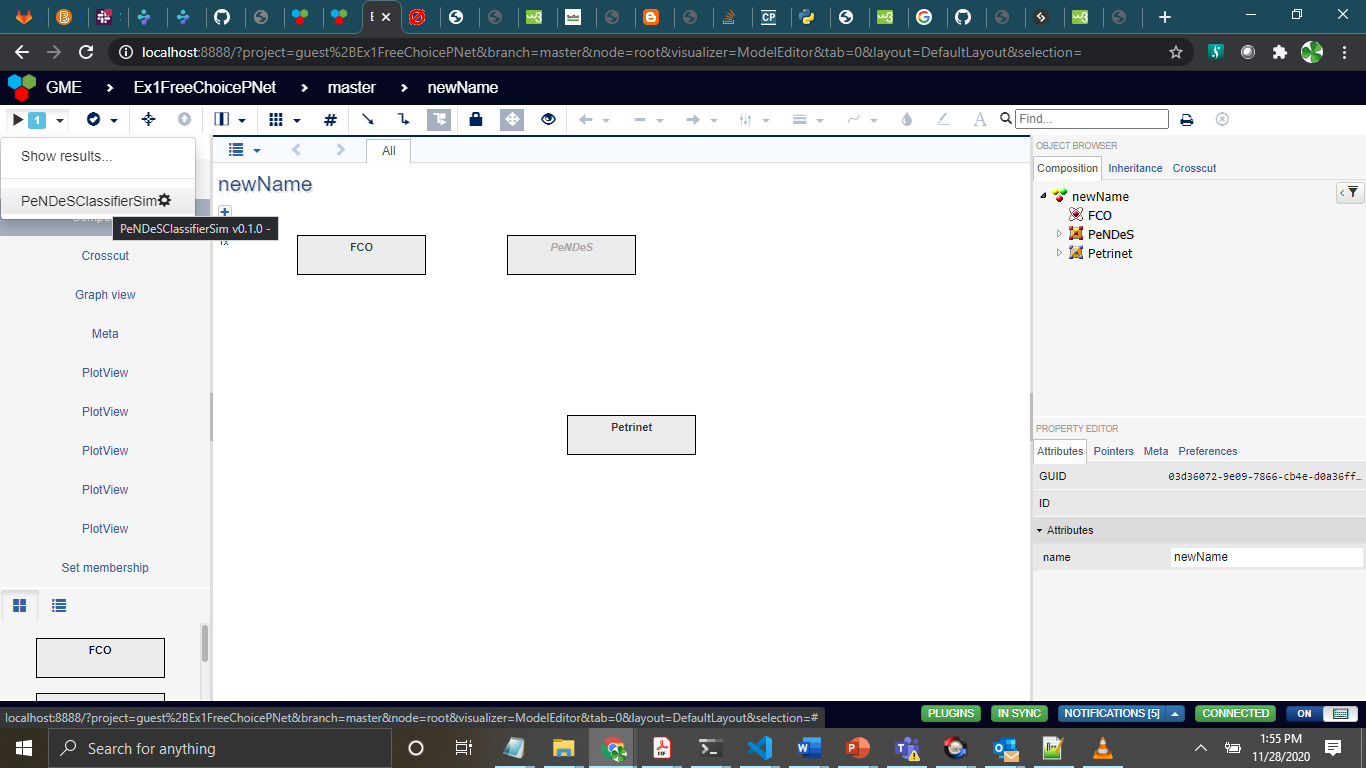
**Note: You can interact with the petri net contents by dragging the elements so that view becomes more realistic petri net according to user convenience. The parenthesis value in Places denotes markings.**

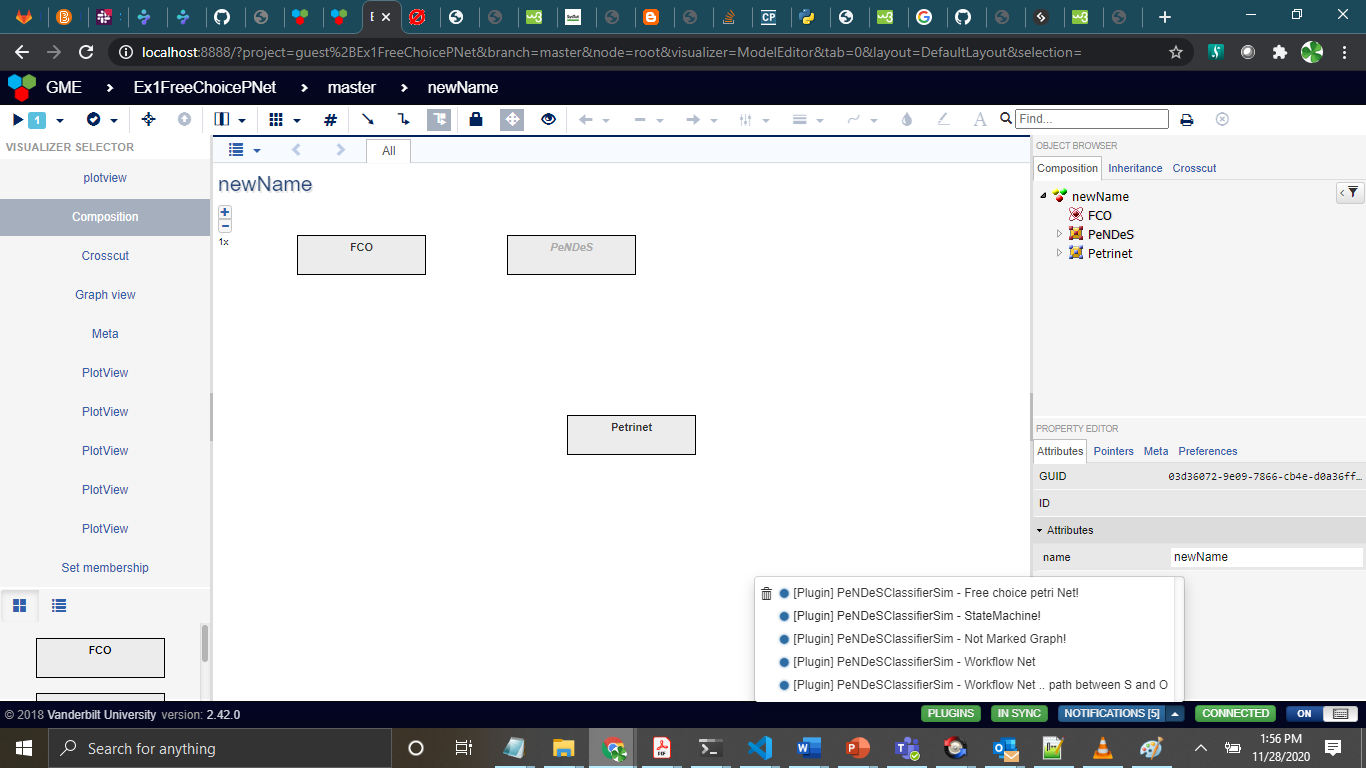
## Next : There are some examples of petri net types and its usage explantions :

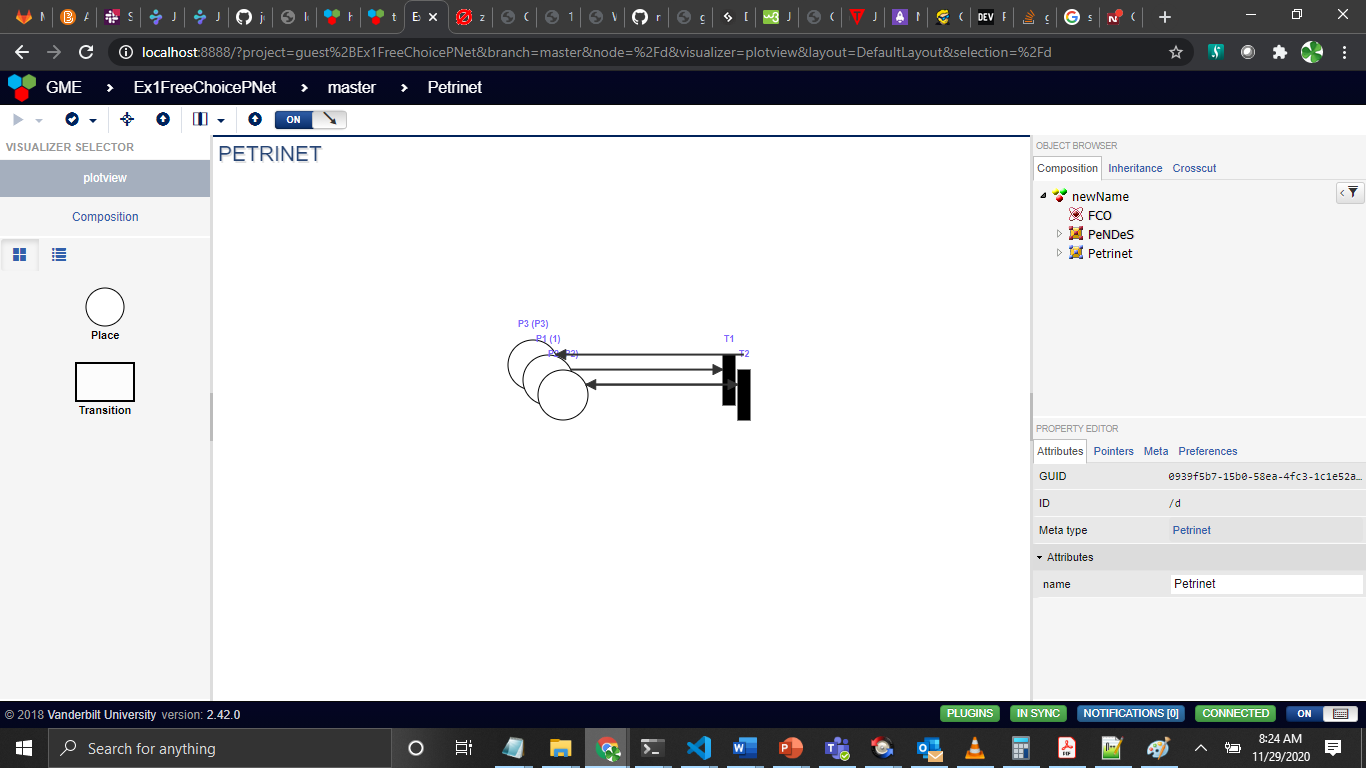
1. FreeChoice Petrinet

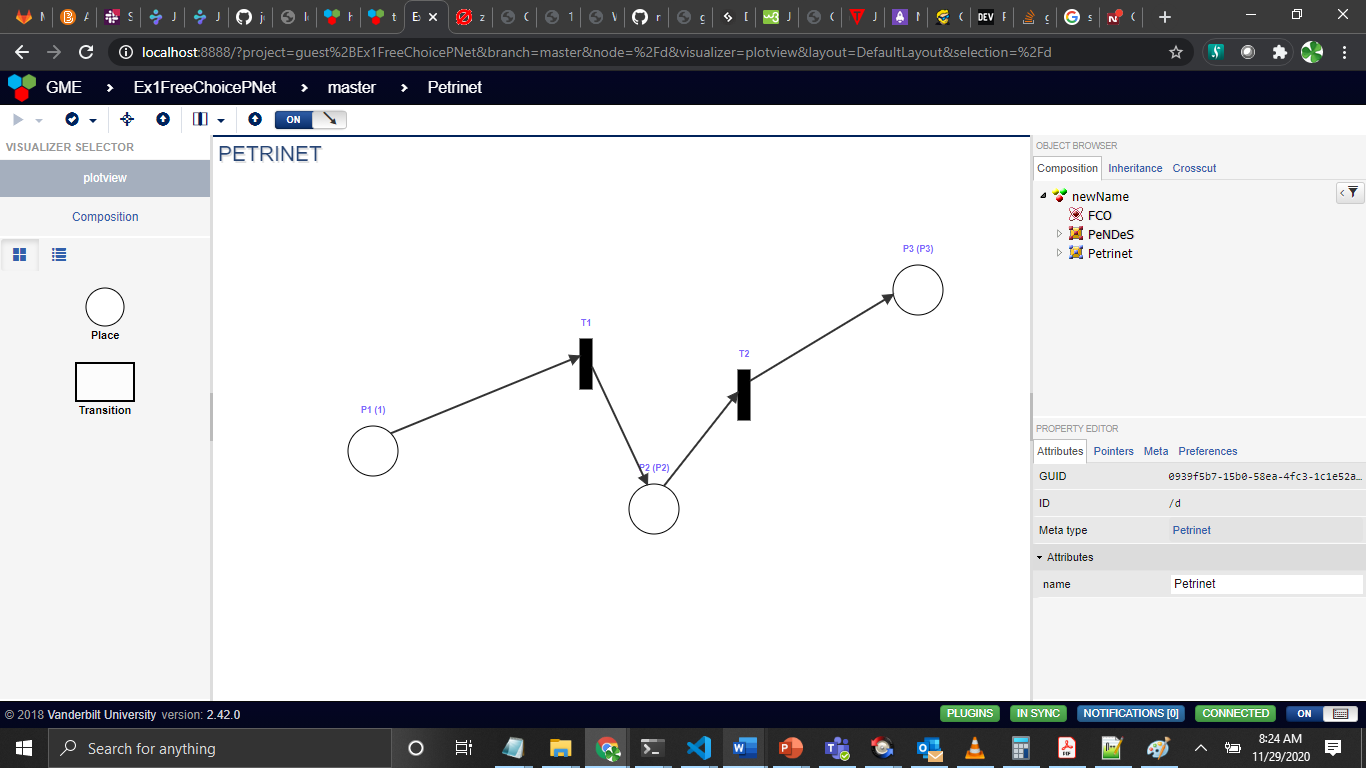






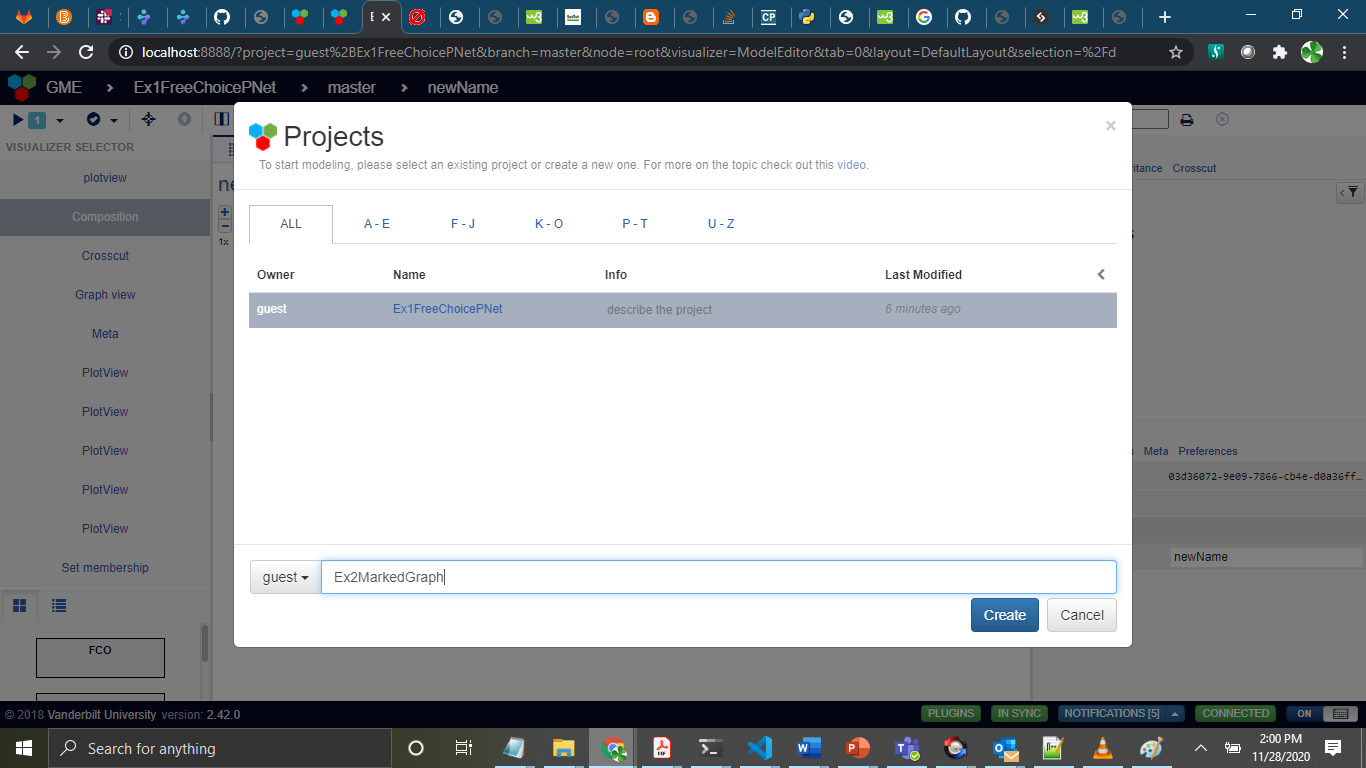


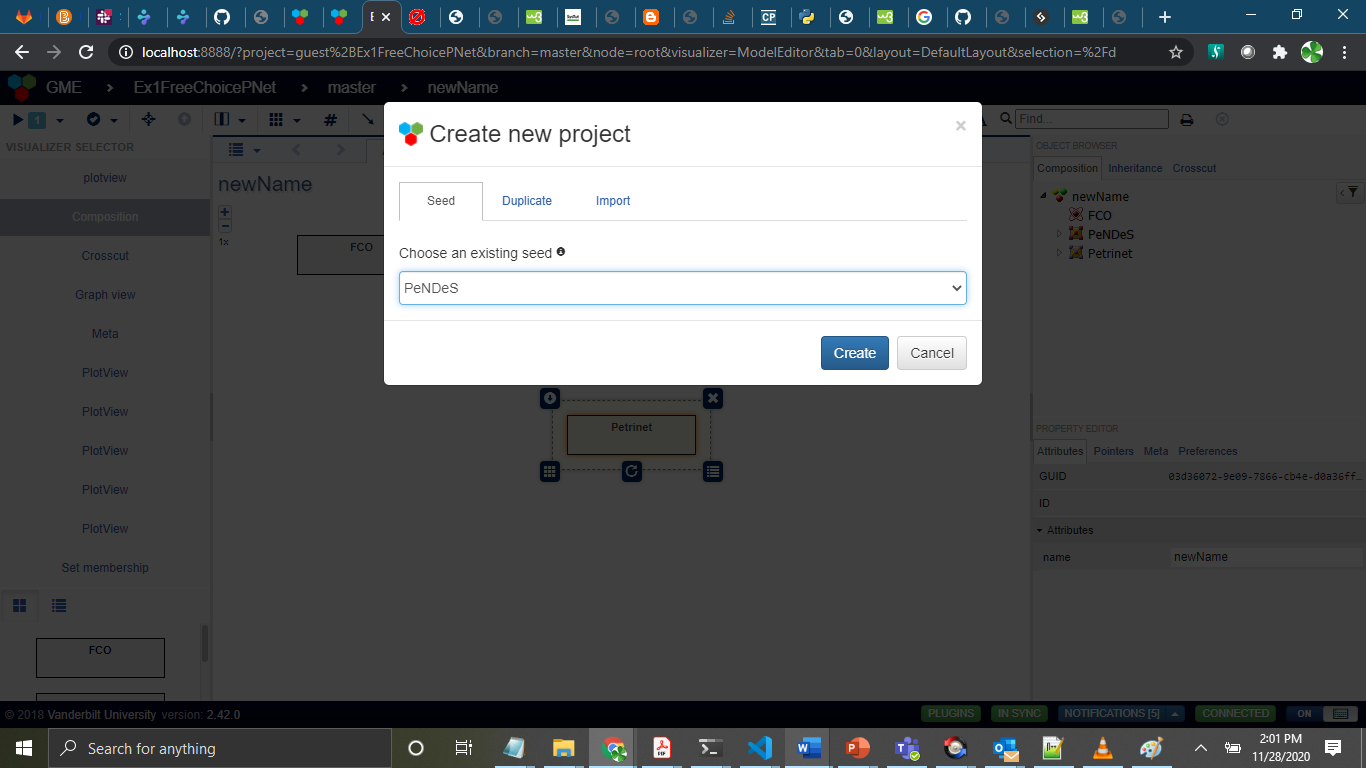


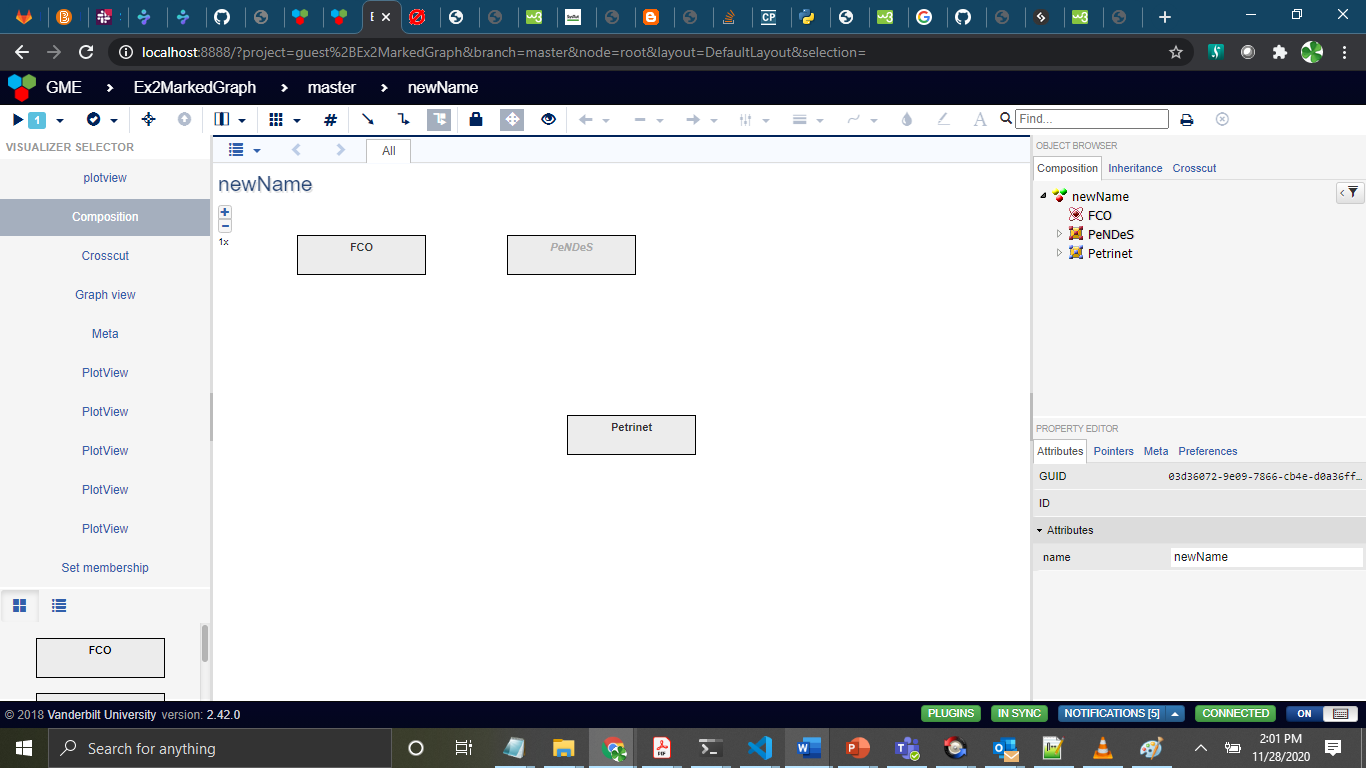


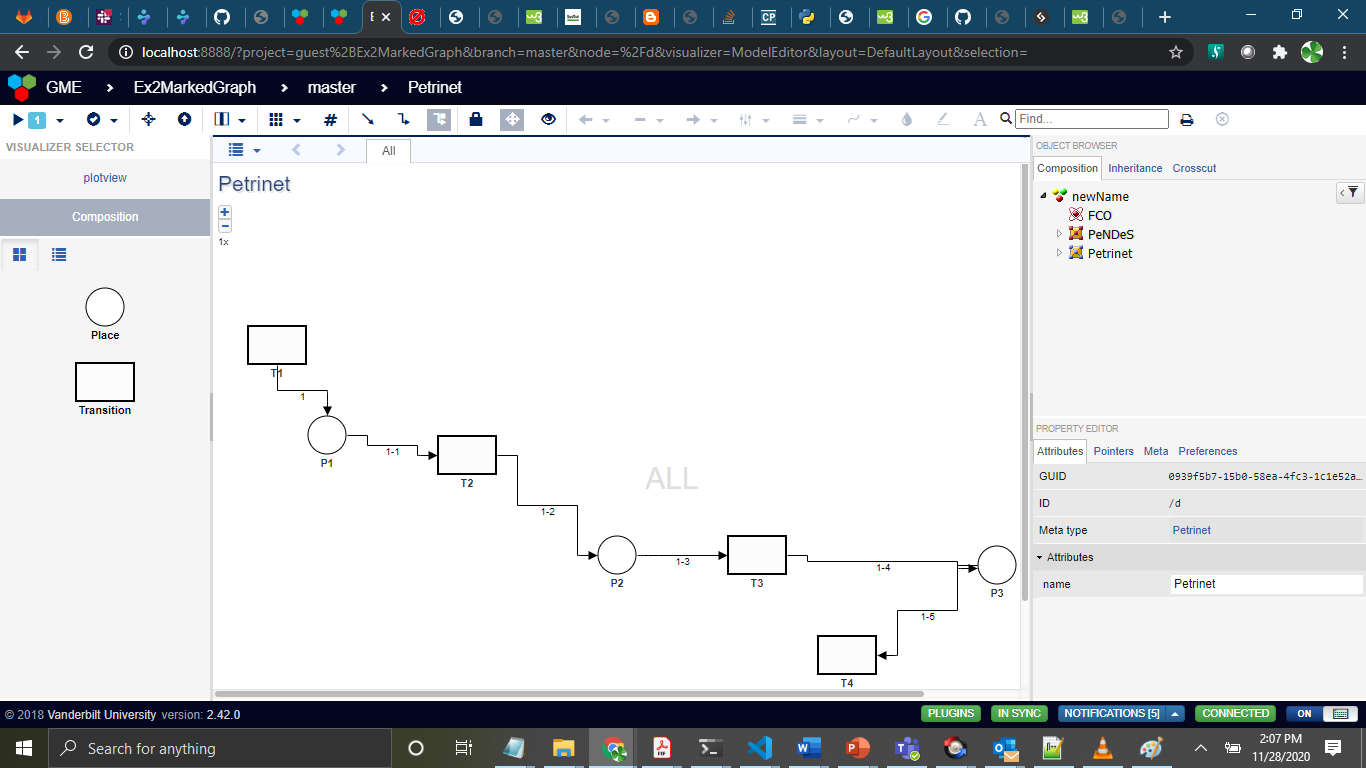
Note: After placing the elements properly in accordance with user convenience.

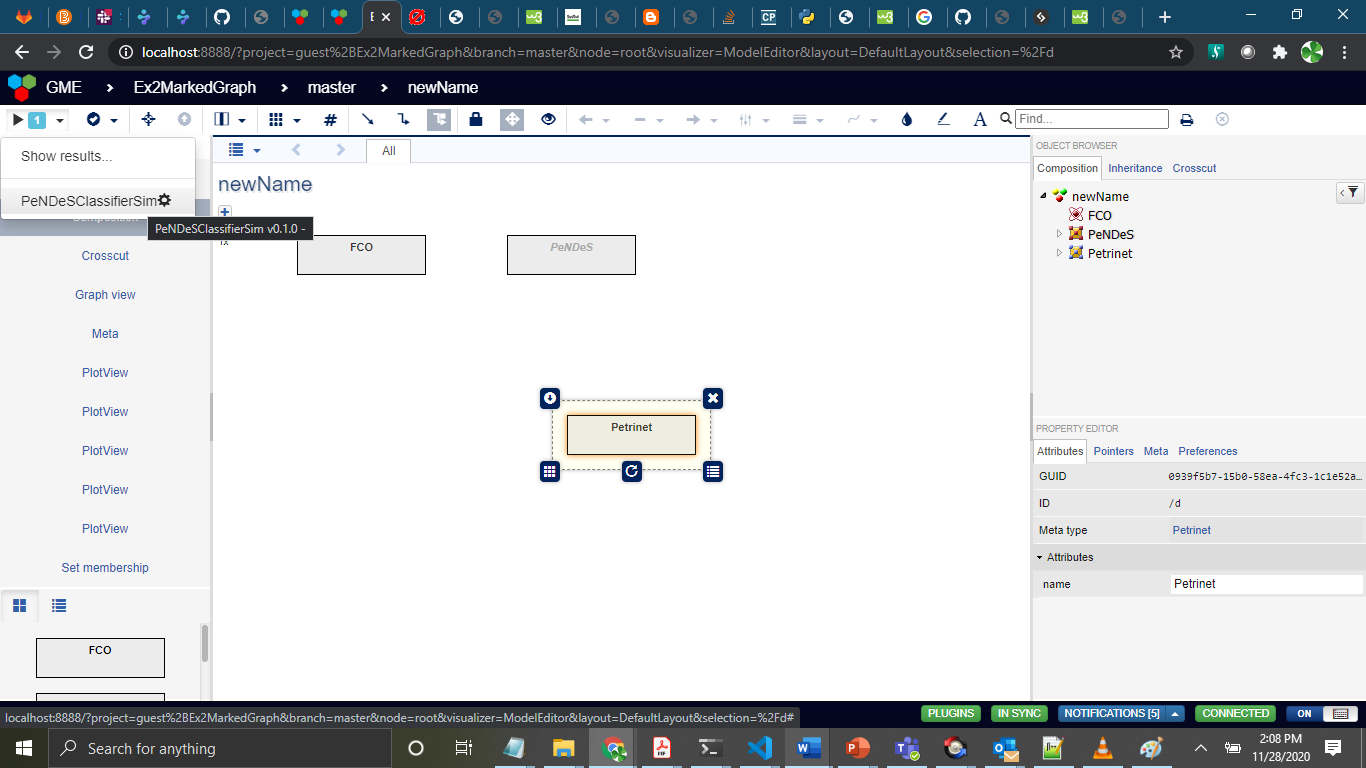
1. Marked Graph petrinet:

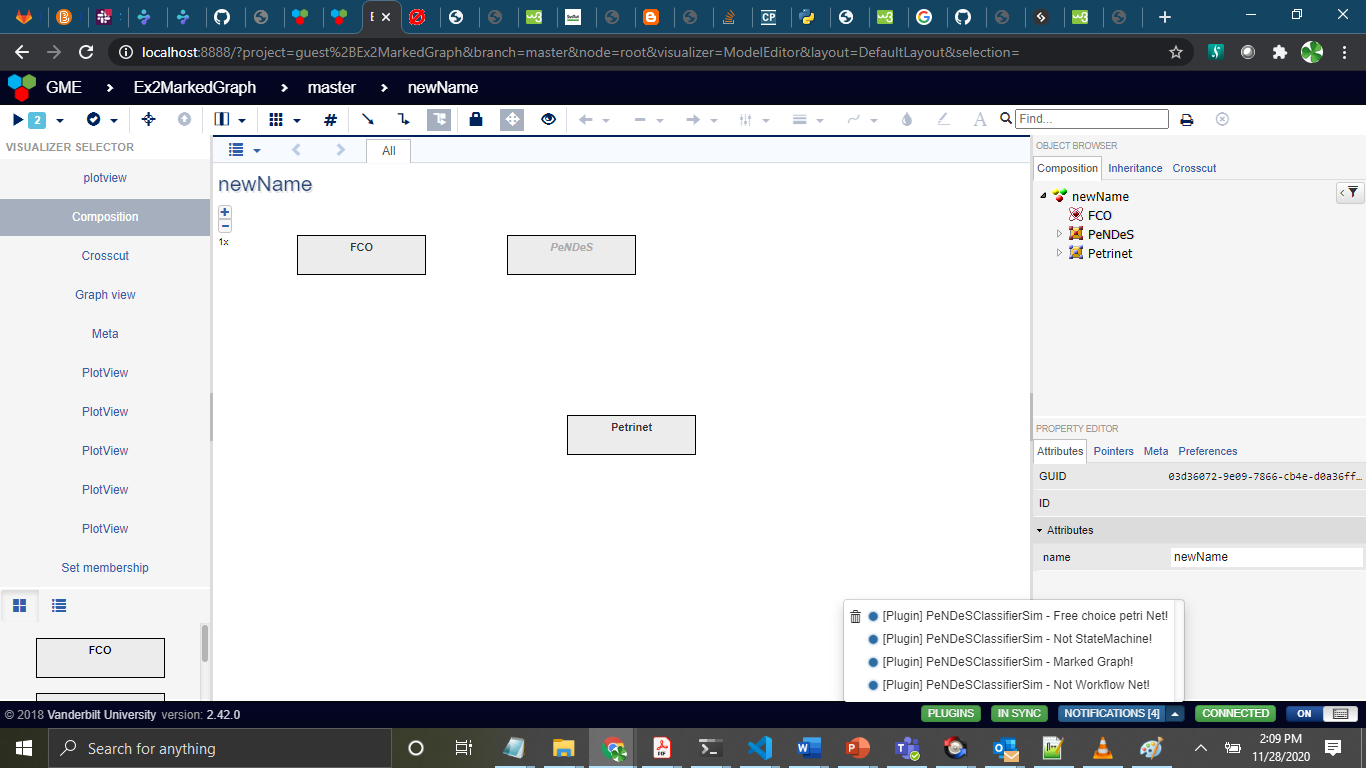


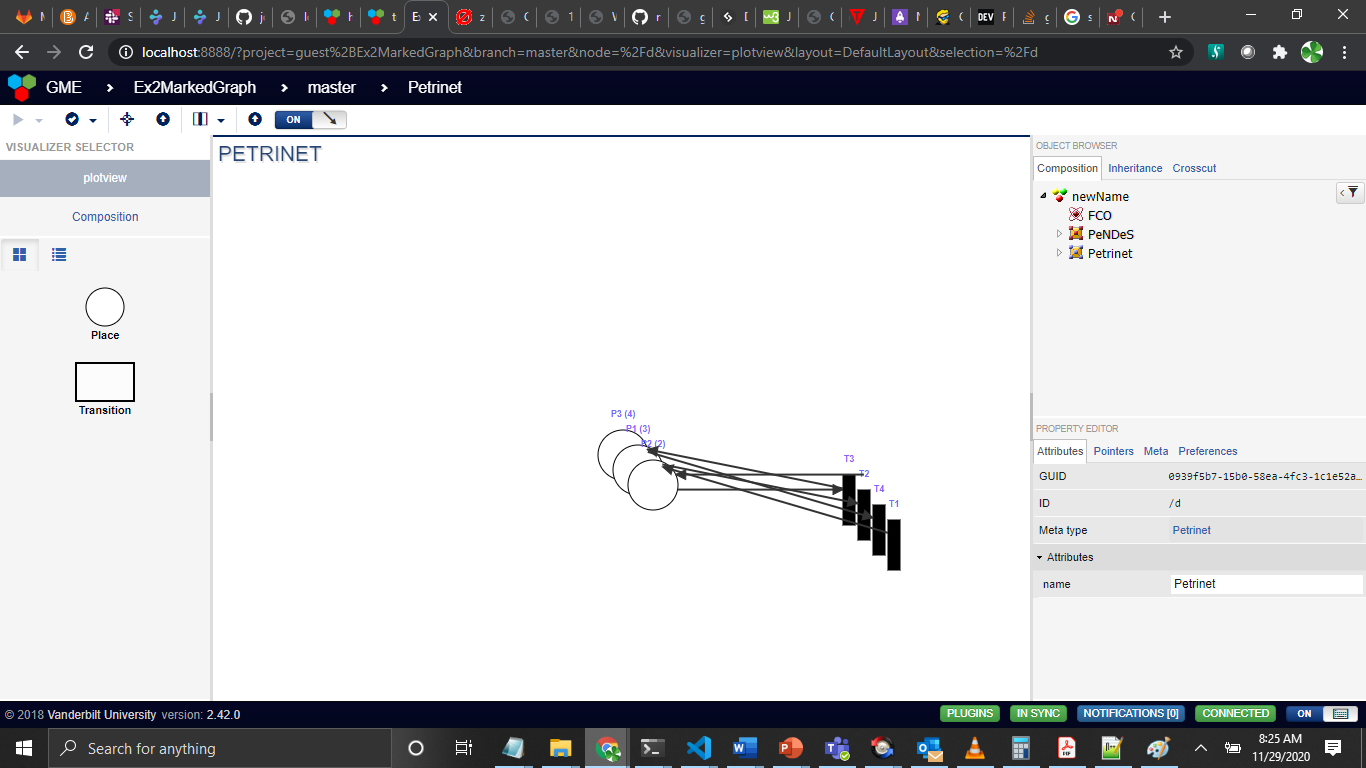


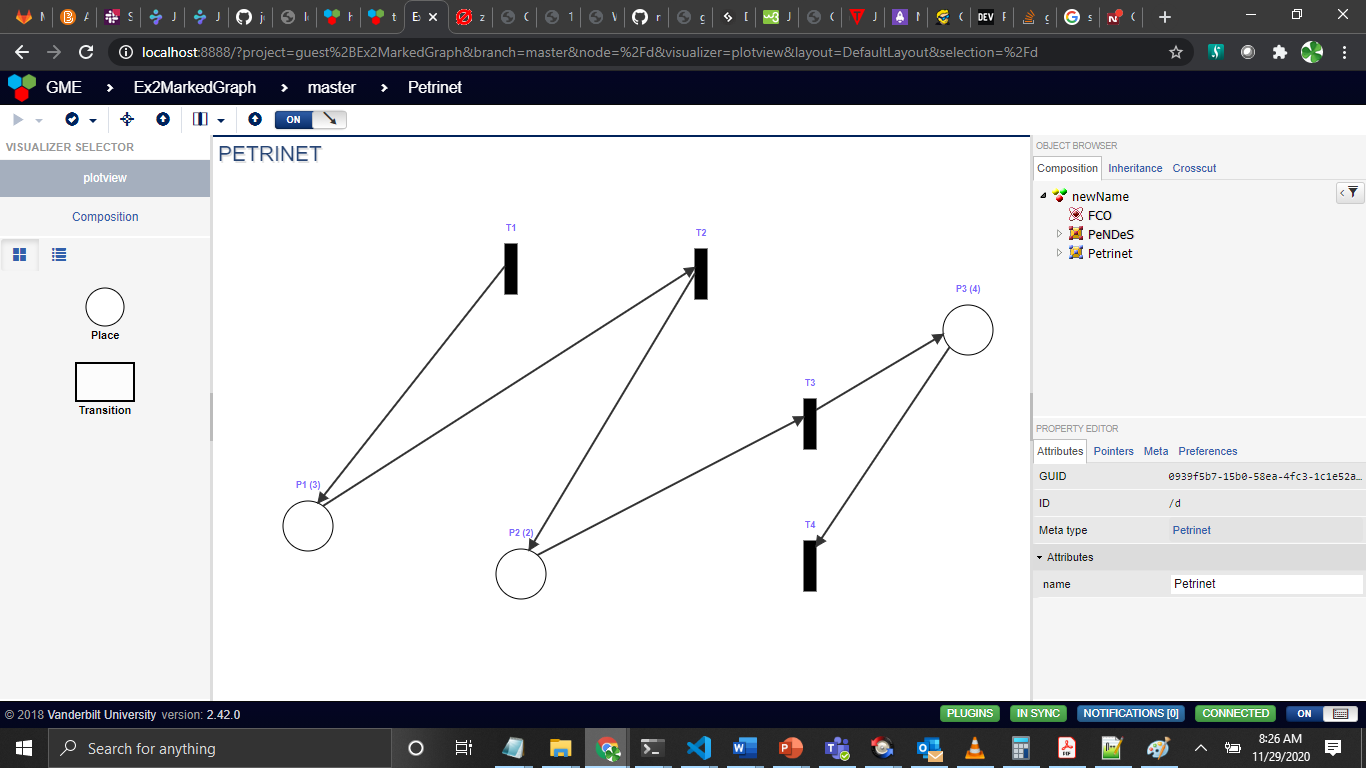






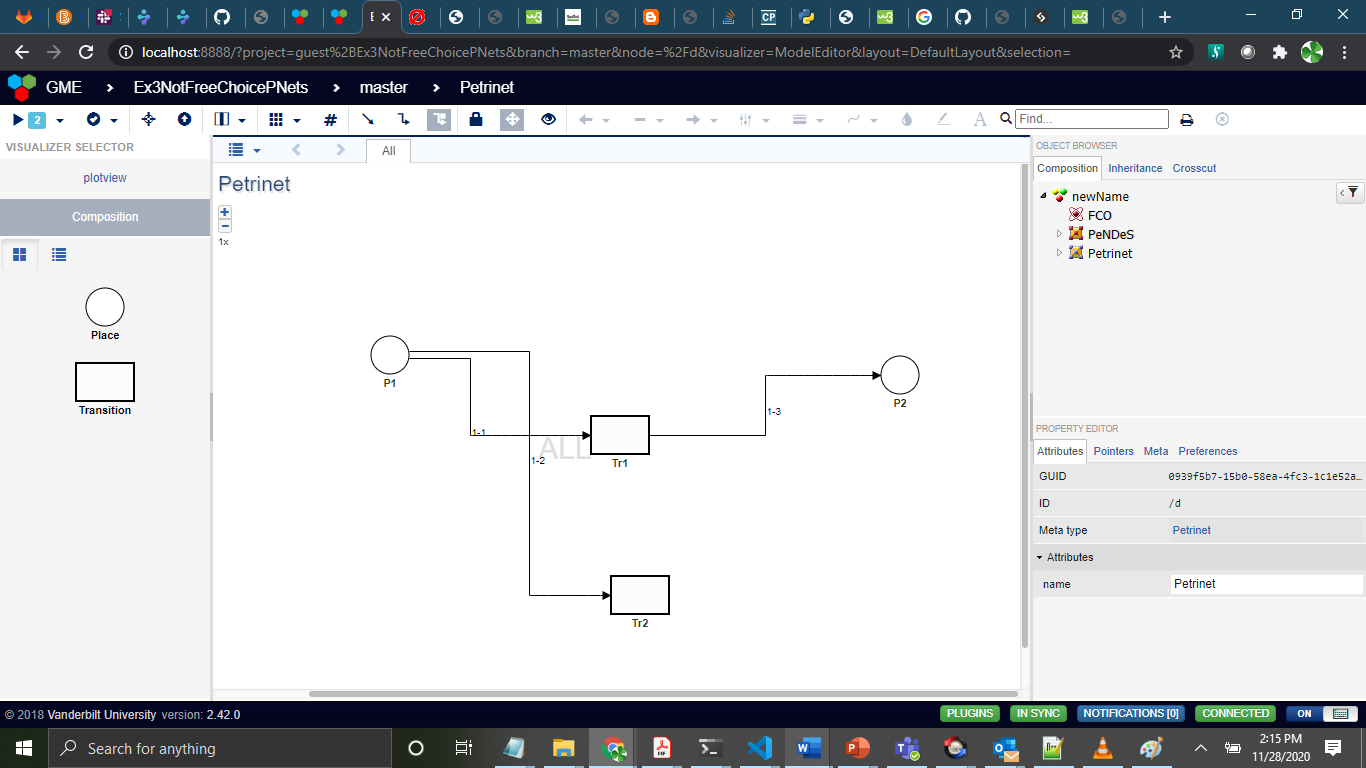


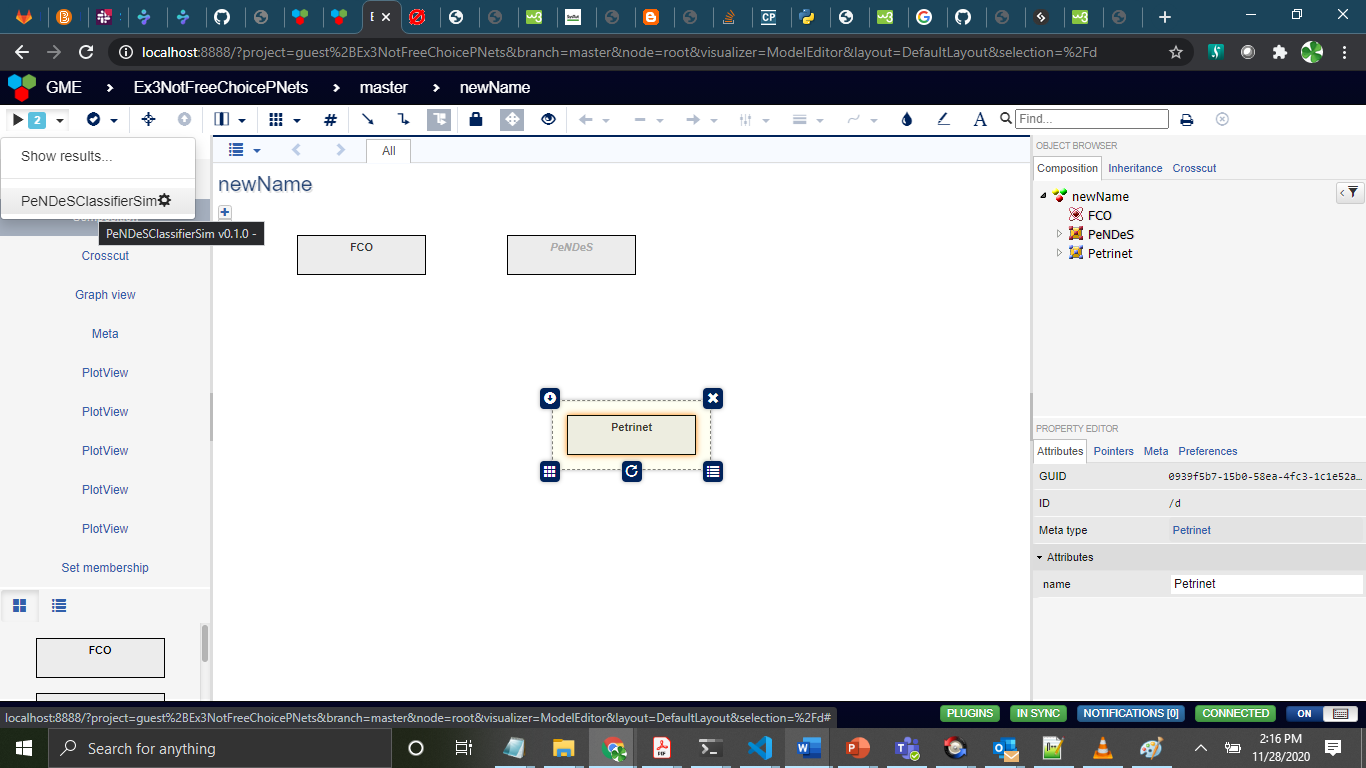


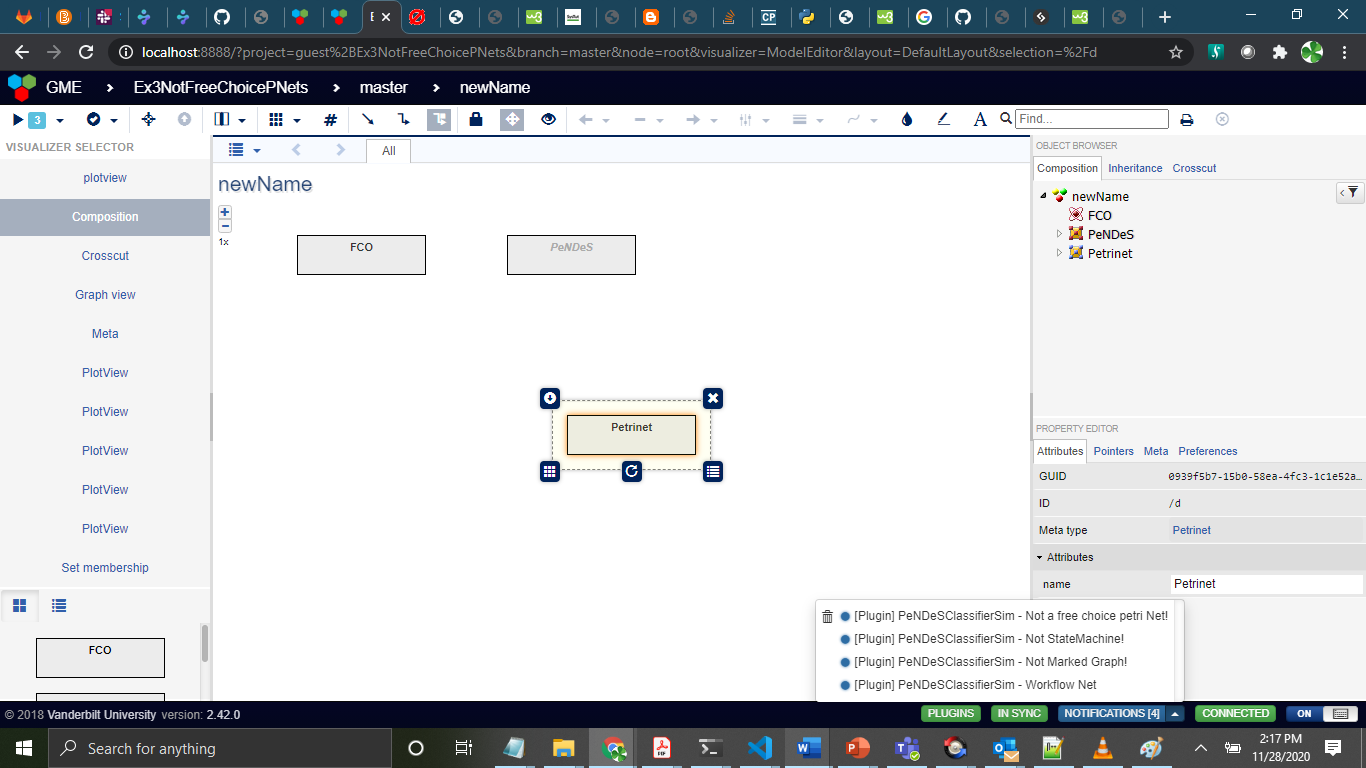


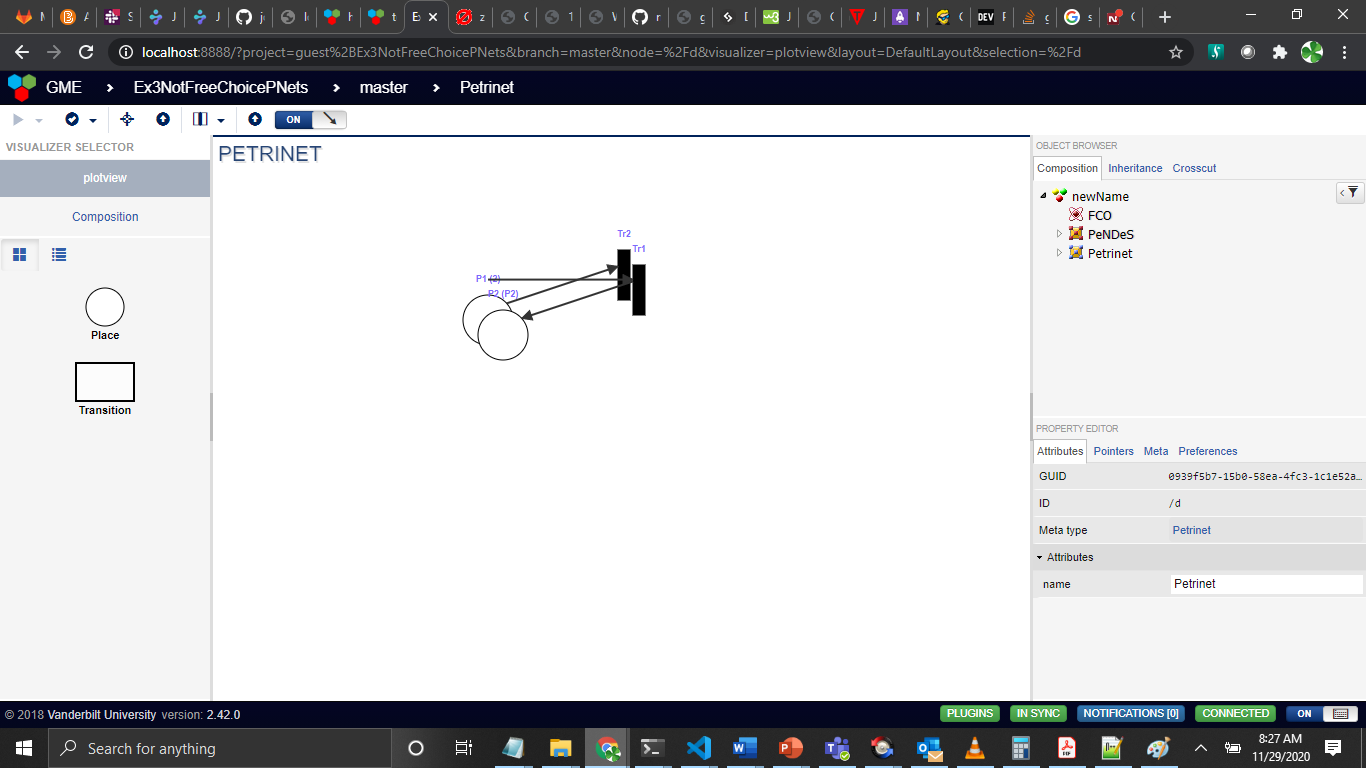
Note : After disentanglement of elements

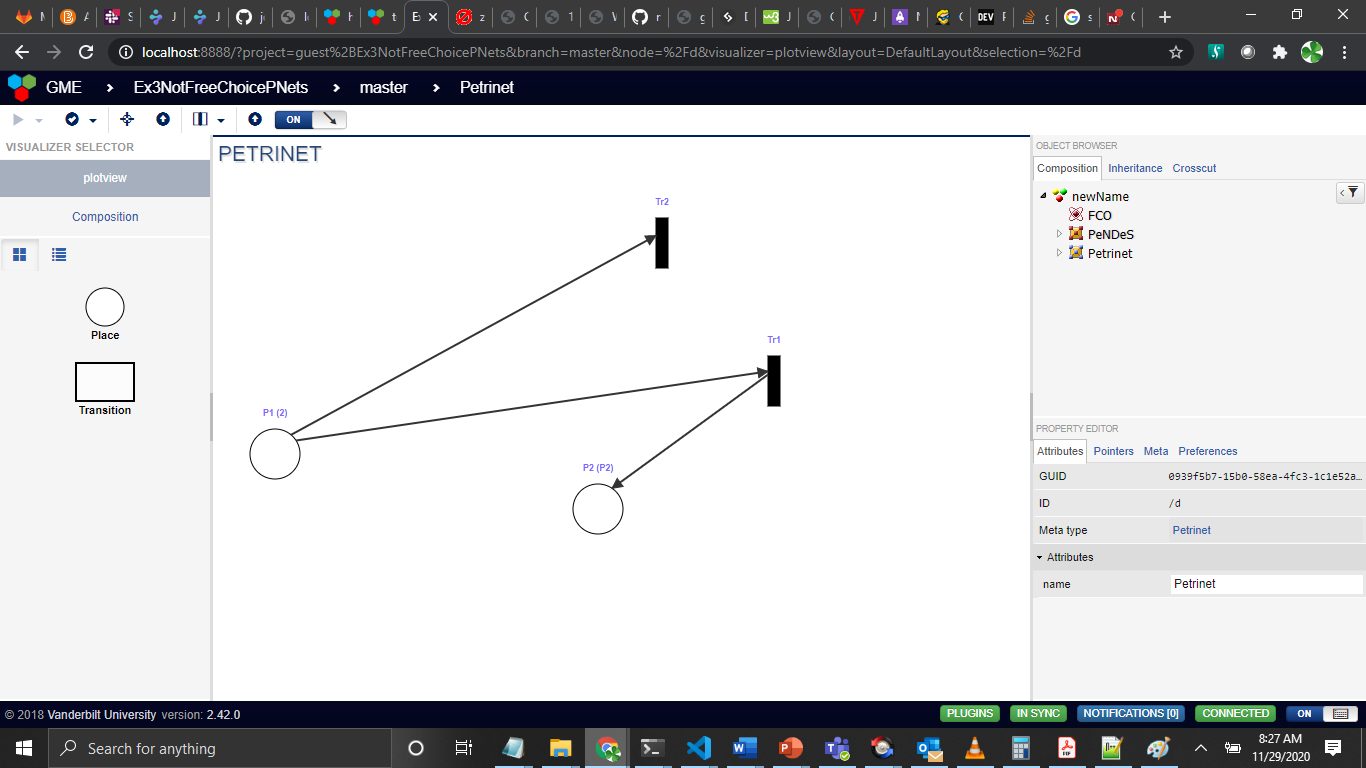
3.Not a FreeChoice Petrinet:











Note : After disentanglements of elements