I improved the prompt injections by adding line cuts instead of giving the AI one whole paragraph of instructions so he can get a clear idea of each instructions

Added the rules logic in the rest of the prompts so it takes in consideration the rules all the time and not just when adding something on the canvas, which improved Dynotate’s response a lot, it’s a lot more accurate now.

To create a benchmark of the same format as the Mars game for the other project, I started writing rules like like “a cat should be next to a box” similar to what we have been doing to test the rule logic quickly.. But then I thought I should just use Arabi’s game rules like we planned to do and wrote the rules down in an excel sheet this way:

A screenshot of a game

Description automatically generated

And in another sheet, I wrote a list of inputs that either directly or indirectly contradicts/impacts the initial rules. Also added some examples of inputs that should not go against the game rules(see yellow highlighted rows) to test the AI in various ways.

A screenshot of a computer

Description automatically generated

**Dynotate test?**

First, in one input I told the AI that I’m going to explain a game and gave it the rules of the game as it is in the first excel sheet but added “\n” to make it easier for the AI to understand. This was the output:

A screenshot of a game

Description automatically generated

It seems to be adding rules in general in a good way but not storing the details, for example, yes the player can take one action per round but it didn’t store the specific actions they can take.

A screenshot of a phone

Description automatically generated

Dynotate is still limited to the DEFINE, ADD, RULE logics, so I shouldn’t expect it to output in the terminal or in the app that I’m contradicting my initial rules.. I guess this should be improved next..?