# UNDO FUNCTIONALITY

The undo functionality in the game can be implemented using the MVC design pattern with the following components-

1) **Model –** The model code files show that it is responsible for maintaining the state of the game and tracks the moves made by the players. It ensures that the moves are legal and that the player inputs are valid in context with the game design and rules. One way that it enables undo functionality is to keep track of the latest move made so far at every stage. This can be implemented with the help of a variable that represents the move and a Boolean variable which states whether undo function can be used or not. The latest move is assigned to the variable and the Boolean variable is marked true. If the undo option is triggered, the state of the game is also pushed one step earlier. The change made by the undo function also appears on the view as that particular block of the grid becomes empty. The Boolean variable is marked false indicating that undo function cannot be used now. Then the next move is assigned to the variable and Boolean becomes true again. However, the function should only be used once as undoing once again removes the move of the second player which should be avoided. The functions in the model code files can be updated by activating the undo function call after the first move is made. The function is deactivated when it's utilized to reverse one move. It will activate once again when a new move is made.

2) **View -** The view displays the state of the game at all stages to the players. For the undo function, a button can be shown in the UI which is usable only at the stages mentioned in the model component description above. It should not be clickable in other scenarios. When the undo button is clicked, the request is sent to the controller and it enables the activation of the undo function. Once successful, it is deactivated again and the undo button becomes unclickable until the next move is made. The code to create the Undo button and its use can be implemented in the RowGameGUI function that is used to create the whole layout.

3) **Controller –** The controller is meant to process the input from the user and updates the model and view components. When the undo button is clicked on the UI by the player, the call is sent to the controller from View. Once the model performs the undo function, the controller updates the new state of the game and notifies the View to make changes in the GUI to show the new state to the players. If the undo function is not activated, then the controller notifies View to disable the Undo

button so that it is not being used at the GUI. In the numerous if else-if statements in the controller code, the undo function call can be added which only works if the undo request was received from View. It can be added at the reset game function with initially it being disabled before the first move is made.

To summarize, the Model keeps track of the latest move and the current state of the game. The View presents a button for the Undo function and enables/disables it based on the particular situation. And the Controller deals with the user input and notifies the View and Model with appropriate function calls depending on the case.