The Computational artifact I shall be researching is how multiplayer games sync their state. The entire idea of this artifact is that multiplayer games can be played together from different sources, or computers rather. If you made an input on one computer, how can the other computer read that one input? What about the other 98 people in your Fortnite game? In my artifact I describe in minimal detail how this is occured.

I researched as much as I could to find a good description of what is exactly happening in this idea, and I found a 2 part article describing in very exact detail. I then used the information given plus my previous knowledge of the matter and made up a quick script that I can use as a voice over for a video I could make. Using Adobe Premiere Pro, I was able to successfully, record my script, edit, analyze, and publish a video from scratch and have the video look and sound artistically pleasing to the average viewer.

This computing innovation has done basically one thing that is used in every single online multiplayer game today and is what makes actually playing these games accessible. If we didn't have this invention a lot of people would still be playing single-player games, by themselves, or local multiplayer games, unable to communicate or play or even interact with those in a different house/in a different place. There is not much to say that this innovation has done harm other than create what we call in the video game world, "Lag". The delay between your computer, to the server, to all other computers usually takes some time rather than a direct connection to a certain computer which is used to cut down this "Lag". But because of this, many competitive games have spawned and this "Lag" could be what keeps someone from winning. If someone has a better response time than one another, but both players execute their input at the exact same time, the person with the better response time wins which can lead to many unhappy people.

This innovation isn't very much harmful or hard to accomplish and fairly simple to grasp the idea of. In simple terms, the input is whatever input into the program, the output is your input being sent to the screens of yours, as well as everyone else in the program's screen, and the transformation of this data is done by sending the input then sends it to everyone at the same time. In more technical terms, the input is the exact input you have inputted into the program which is then sent to the main server where the program is being hosted on and that gets transformed into whatever the program reads that input as, and when that input is changed and read, it then sends all inputs to the output in every other computer running that one same program at the same time. The main data concern is when every single computer inputs something at once, the server has to read, transfer, then output those very commands to everyone at the server all at the same time which may overload the server or desynchronize the server with the players which ultimately messes everything up. One way to stop too much of such harm is increasing the amount of time that input has the read, comprehend, and output the information sent to the server which can lead to you inputting something and not getting a response as fast as usual.

The data my innovation uses is simple coding and binary set systematically to allow multiple people from multiple devices to interact with each other. This is also done in a simple, fair, and easy way so everyone can understand. The innovation consumes the input from each player of the game, that is then sent to the server to compute the information given at an equal rate which is then output to show that "kick" or "punch" that you inputted at the same time your

opponent saw that very same input. The main concern this has is the fact that too many people on one server can lead to too many inputs at once for a server which leads to server crashes, or messed up connection in general, which some people use maliciously in order to get their opponent in trouble so they can win themselves which is neither safe, fair, nor fun.

https://medium.com/@qingweilim/how-do-multiplayer-games-sync-their-state-part-1-ab72d6a54043
Qing Wei Lim. How do multiplayer games sync their state? Part 1. 12 March 2019. 10 May 2017.

https://medium.com/@qingweilim/how-do-multiplayer-game-sync-their-state-part-2-d746fa303950
Qing Wei Lim. How do multiplayer games sync their state? Part 2. 12 March 2019. 3 May 2017.

http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.11.2195&rep=rep1&type=pdf
Eric Cronin, Anthony R. Kurc, Burton Filstrup and Sugih Jamin. An Efficient Synchronization Mechanism for Mirrored Game Architectures (Extended Version). 18 March 2019. 28 January 2003.