The project that our team is making is a turn-based strategy game, where you control your troops around the grid, the battlefield, and eliminate the enemies on the field. Parts of the game are inspired from the roleplaying game Final Fantasy Tactics. When making software, one must consider the ethical implications of the code, and what it will be used for. One major issue with our own project is the fact that it so closely resembles the actual game of Final Fantasy Tactics.

Plagiarism is a major problem facing all game developers. While the projects of a group of computer science students in a software design class are not extensive enough to run into any real legal issues, every video game developer should keep this in mind, along with ethical issues. As video games and their developers have slowly risen to the top of the entertainment industry, more emphasis to some of the effects video games have, particularly when the users are children. Many of these issues, such as violence in video games, have recently received a great deal of attention, with many activists pushing for increased censorship.

In our game, you are rewarded for killing your opponents, these rewards coming in the form of experience and gold. Both are necessary to progress further in your quest. This gameplay can be interpreted as promoting violence, even though a depiction of the player actually defeating the enemy is not shown. Instead, if you kill an enemy in combat, a gravestone replaces where the opponent “died.” While the aggression factor in our game is definitely lower than others, our game has a cartoonish aspect. This comical violence presented to a child that is predisposed to aggression could desensitize them and serve as a justification for future acts of violences.

Regarding the software itself, if any portion of the code is reused out of the original scope of the program, the code may not behave as the new user intends, unless the new user of the code
thoroughly understands how to properly use it. With this in mind, any software developer must make sure to explain what each portion of code does, in the form of comments. Another option would be to make some aspects of the program private, so that they cannot be modified by unaffiliated developers. By taking these measures, the potential risk involved with reusing code is mitigated.