**Project Description** [2.5 pts]: The name of the term project and a short description of what it will be.

This project will be called BattleShip!, but I believe I’ll be changing the name to something more original and add my own twist to this game. As mentioned, I’ll be recreating Battleship with 112 graphics, but it’ll be with 2.5/3D graphics, as well as 3 distinct types of AI and a 2 player mode.

**Competitive Analysis** [2.5 pts]: A 1-2 paragraph analysis of similar projects you've seen online, and how your project will be similar or different to those.

Battleship itself is not a new concept to the Internet or otherwise, there are plenty of battleship games out there, especially online ones to play with friends. What is not as common, however, is AI. Battleship AI is a lot harder to code or figure out due to the nature of the game. Unlike Chess, for example, there is little to no strategy. How players play is mostly dependent on luck. But I believe I can make an intelligent AI that searches systematically as I’ll explain below.

**Structural Plan** [2.5 pts]: A structural plan for how the finalized project will be organized in different functions, files and/or objects.

Just in one file. I’ll have functions that draw the different menus, mousePressed detectors for buttons, and then grid drawing functions. There will be a ship object that gets placed and has a row,col stored within it. Currently in this zipfile, I understand there is also tp0 and tp0 easy AI. These are just versions of my battleship game played in the terminal instead, one with 2 player and one with an AI. I’ll be assimilating these later into the main file as my final product.

**Algorithmic Plan** [2.5 pts]: A detailed algorithmic plan for how you will approach the trickiest part of the project. Be sure to clearly highlight which part(s) of your project are algorithmically most complex and include details of the algorithm(s) you are using in those cases.

There are three different ways to code Battleship AI. The simplest way is to just make it shoot randomly. The next way is to have it shoot nearby ships of one that you’ve already shot to find another one. The final way to have it learn to shoot based upon where the ship can possibly be as it keeps shooting. I’ll be using an expectimax algorithm.

**Timeline Plan** [2.5 pts]: A timeline for when you intend to complete the major features of the project.

I plan to finish the simplest AI(random shooting) and the next more difficult AI(random + shooting around ships you’ve already shot) by tp2. I will also add better graphics, animations, and images especially for the buttons and background.

**Version Control Plan** [1.5 pts]: A short description **and image** demonstrating how you are using version control to back up your code. Notes:

**You must back up your code somehow!!!**

**Your backups must not be on your computer** (ideally, store them in the cloud)

I’ll simply upload my files onto google drive every hour or so.

Graphical user interface, application

Description automatically generated

**Module List** [1 pts]: A list of all external modules/hardware/technologies you are planning to use in your project. Note that any such modules must be approved by a tech demo. If you are not planning to use any additional modules, that's okay, just say so!

No additional modules.

**TP3 Update**

I did not add 3D graphics or 2.5D graphics, instead I added a 4th machine learning AI that would play based off how you play. Also, the third AI is not expectimax, but rather the Monte Carlo method.