**Software Requirements and Design Document**

**For**

**Group Tanx**

Iteration 1

**Authors**:

Blake Antone

Andrew Thrash

Nolan Monahan

Alexander Gentry

# Overview (5 points)

Our goal for the project is to develop a browser based reinterpretation of the Wii play game Tanks that uses mouse and keyboard for controls.Essentially, we are creating a point-and-click bullet-hell-esque game featuring one user completing different levels based on the objective given to the player. We are approximating a 10 stage game with new map designs, new enemies, and differing objectives from the original game. Some features we are considering adding to our game include power ups to the tanks such as: speed boost, increased fire rate, auto aim, etc., enemies that vary from the original game’s, better controls that do not include Wii motion tracking, other additions meant to improve gameplay and give our spin to the game.

# Functional Requirements (10 points)

1. Application Opens and Runs: Ensure that our project can actually run on start up correctly. Must show things like Game Menu, Stage Selection Menu, the Levels itself, and other functions that we put into our project

Priority: high

1. Stages: Create multiple stages that allow users to traverse, complete, and play against AI opponents. These stages will contain walls so things on the map won’t go off screen and also to provide some challenge. Levels must be traversable and easily viewable to the viewer that contains clear sight lines for users to see the AI.

Priority: high

1. User Tank: Allow users to control a tank that can move and shoot around the map. Will allow users to use their keyboard to move the tank (WASD) and use their cursor to control the direction in which the tank fires.

Priority: high

1. AI Tanks: Create multiple different AI opponents to provide a challenge to users when playing levels. There will be different AI tanks that will have different abilities and will progressively get more abilities as the game goes on. For example, the 1st/tutorial stage may have a tank that doesn’t move or shoot and die on one hit so users can get used to the simple functions of the game. Then, the later levels may feature AI that can move faster than the user, have projectiles at higher velocities, take more than one shot to hit, etc.

Priority: high

1. Projectiles: The user tank and most AI tanks must be able to shoot projectiles on command. Projectiles must kill, regardless of where they come from, on impact of a tank. They will also only take a certain number of bounces before they disappear and tanks will only have a limited amount of projectiles on screen that the tank that shot that projectile can have.

Priority: med

1. Game Menu: Will present the user with the option to play the game or quit. Should the Player choose “play game”, the app will switch to a stage selection menu. Otherwise it will quit.

Priority: med

1. Stage Selection Menu: Will present the user with an option to choose any one of the 10 stages and a back to game menu option. If a stage other than stage one is selected, the user will be prompted to enter a stage code. The stage codes are presented at the end of each stage so no stage can be played without beating the prior stage.

Priority: med

1. Tank Selection Menu: Add a customizable feature before starting a level that allows users to customize parts of their user tank. For example, be able to create a tank that is red instead of the default color or even select a different sprite for the tank, like a turtle. This will save the users options for when they are playing multiple levels and keep the same customization across those levels.

Priority: low

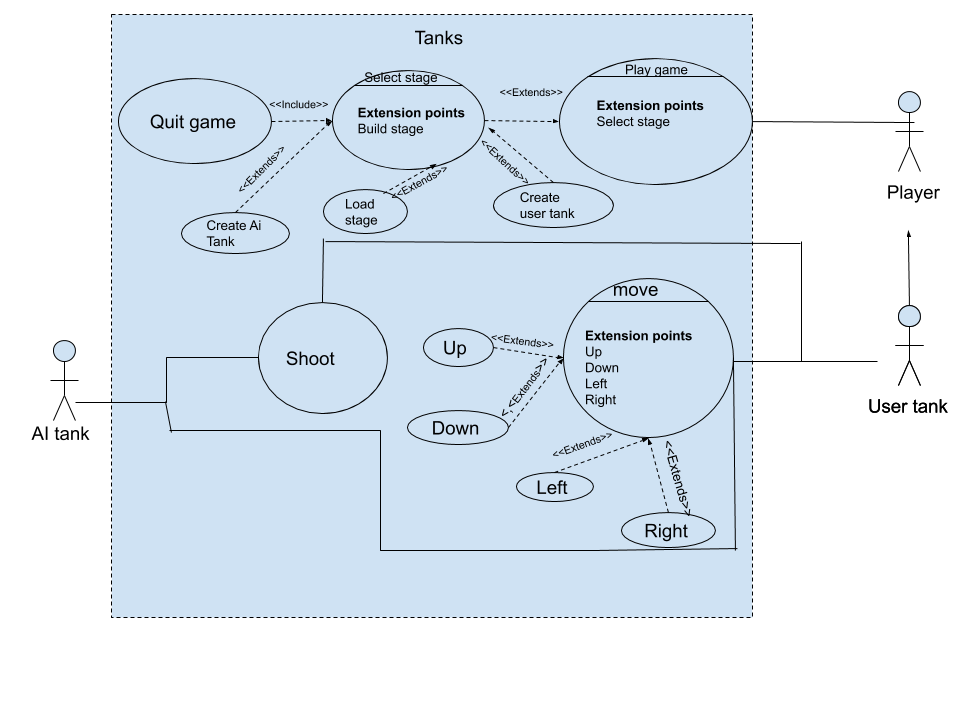
1. Mines: Allow both users and AI tanks to place mines that will take a certain amount of time for the mine to detonate. Must include a reasonable blast radius and time to detonate and also must ensure that AI tanks avoid running into their own mines regularly.

Priority: low

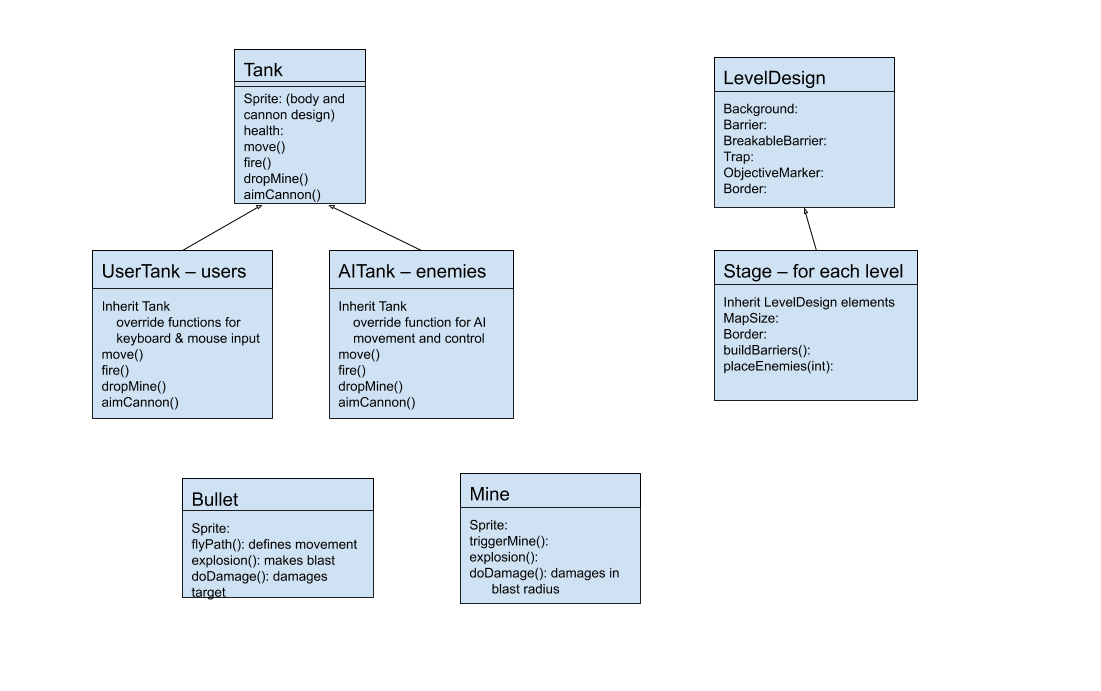
# Non-functional Requirements (10 points)

1. The software must run on any HTML5 supported Web Browser
2. A user will not be able to access a level until the previous level has been completed
3. The levels and menus load within a reasonable amount of time
4. The gameplay runs relatively smooth
5. Levels/menu screens are killed once the player switches to a new screen, ensure previous screens are not still running in the background

# Use Case Diagram (10 points)

**

# Class Diagram and/or Sequence Diagrams (15 points)

**

# Operating Environment (5 points)

Our software will operate on any desktop computer with any operating system given that the user has an HTML5 supported web browser installed, and it will rely on the Phaser 3 framework for gameplay implementation.

# Assumptions and Dependencies (5 points)

We’re assuming that there will not be any major issues implemented into the Phaser framework during the duration of our project as the entire implementation of our project is dependent on the functionality of this framework. We also assume users are running our project on a desktop that can support a browser game and a system that can support a local server. We also will assume that XAMPP has no framework updates or problems when hosting our project on a local server.