CSCE 606 - Vaccine Hesitancy Game: Final Report

Team roles:

- Product Owner: Benjamin Hawn
- Scrum Master: Catherine Shen
- Team Member: Bhogesh Maddirala
- Team Member: Liuyi Jin

Github repo: https://github.com/bhawn/CSCE_606_Project
Pivotal Tracker: https://www.pivotaltracker.com/n/projects/2536278
Deployment: https://vetmed.tamu.edu/peer/one-health
https://vetmed.tamu.edu/games/vaccine-hesitancy/GameDeploy/
Poster and Demo Video: https://vimeo.com/656379837/c5f520de05

Motivation and Summary:

Our customer, Dr. Walker, on behalf of the Partnership for Environmental Education & Rural Health (PEER) Program at Texas A&M's College of Veterinary Medicine & Biomedical Sciences (CVMBS), would like an educational game that raises awareness about the benefits of vaccines, particularly that of COVID-19, for those who are vaccine hesitant. A heavy emphasis was placed on making a vaccine-themed game fun and playable rather than going into detail of how vaccines work so that students will enjoy playing it while gaining the understanding of the benefits of vaccines. The stakeholders are the PEER program and the students and children who will play the game. We were given creative control on how to design the game.

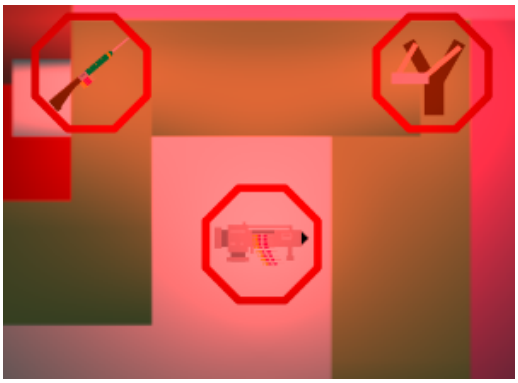
The game must use JavaScript or a JavaScript framework and be deployed to the PEER Program WordPress website by Samiksha Marne. We used the Phaser 3.0 JavaScript framework, worked in the AWS Cloud9 environment, deployed code to GitHub, deployed to Heroku for testing, and used Pivotal Tracker for recording and tracking our user stories. In this game, you move your player around the map to attack and avoid incoming swarms of enemy viruses. Killing the viruses may cause random items to drop. Collecting these vaccine boosts gives you a new weapon and attack boost that allows you to kill more viruses at once. This feature in the game helps show the users the benefits of getting the vaccine.

User Stories:

- **Completed with Low-fi UI**:
    - Set up poster to demo game (2 points)
    - Record a demo of the game (3 points)
    - Write a final report for the game (3 points)
    - Feature: Create Drop entities (1 point)

        As a Player
        I want to encounter drops (vaccines, spike proteins)
        So I can improve my attacks.

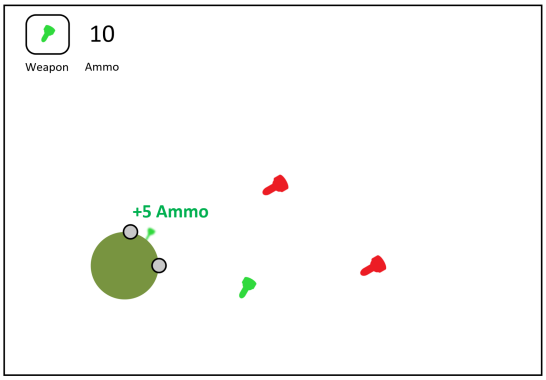
Design **Implemented**

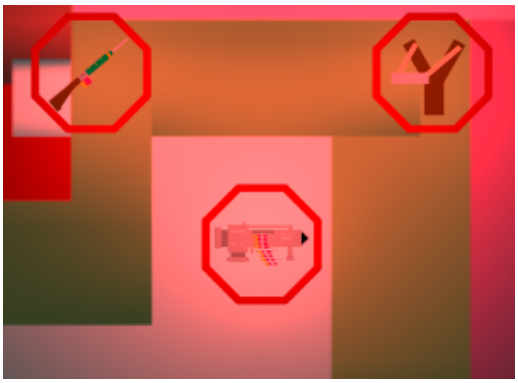


    - Feature: Increase amount of ammo player can shoot on vaccine drop pickup (2 points)

        As a Player
        I want more ammo
        So that I can keep using the same weapon

Design **Implemented**

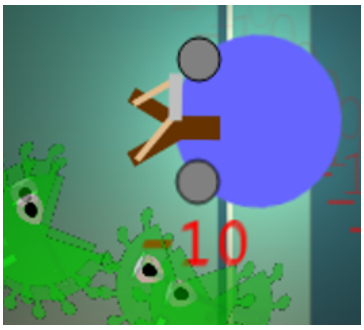


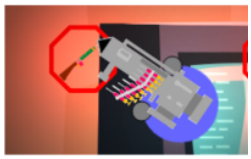


Pick up same weapon = more ammo

- Feature: Entity Collision (pick up drop, damage, etc) (2 points)
  As a Player
  I want to interact with other entities
  So that I can interact with different parts of the game.

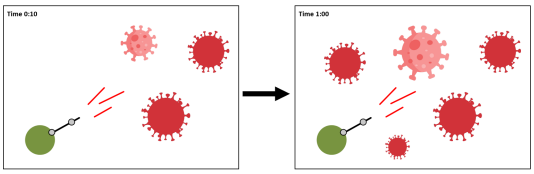
Design **Implemented**

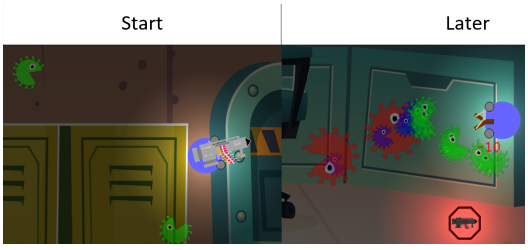




- Feature: Enemy Spawn Progression (2 points)
  As a Player
  I want harder and more enemies as I progress
  So that I can feel accomplished if I win.

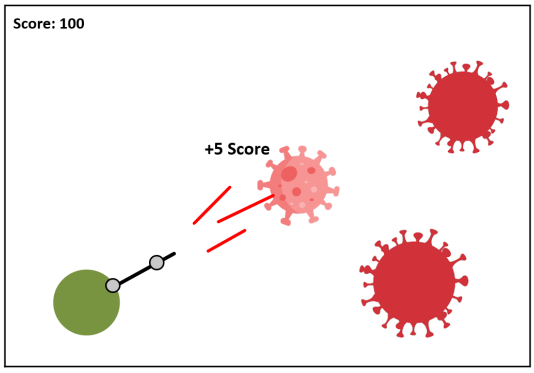
Design **Implemented**

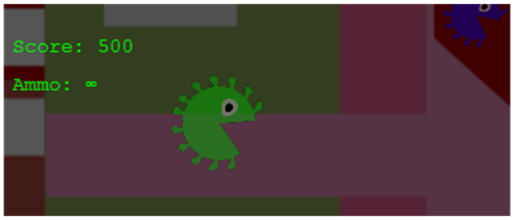


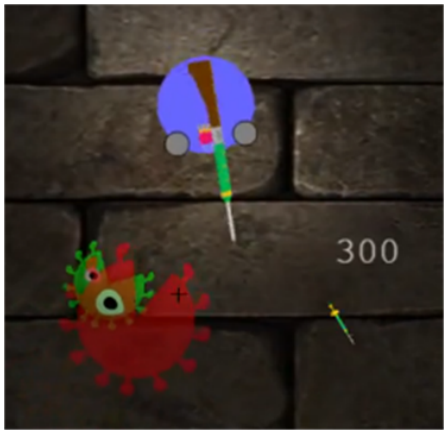


- Feature: Score (2 points)
  As a Player
  I want to see a game score
  So I can see how well I'm doing and compare with friends.

Design **Implemented**

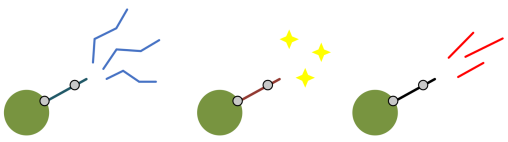


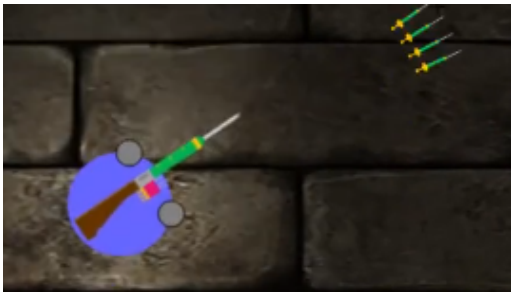


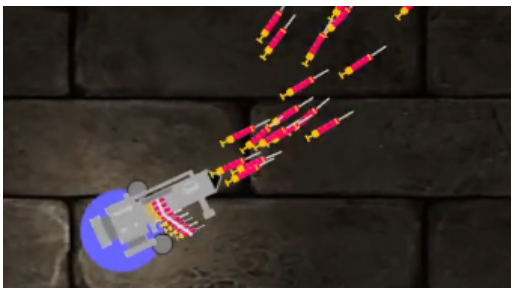


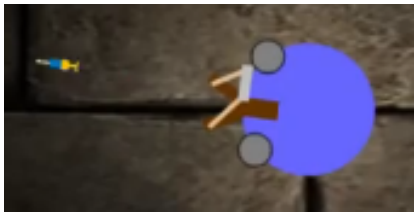
- Feature: Weapon Variety (2 points)
  As a Player
  I want there to be multiple weapons
  So that there is variety in the game

Design **Implemented**



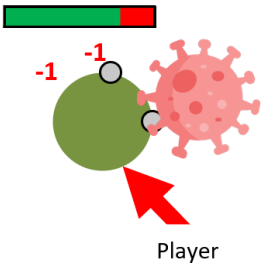




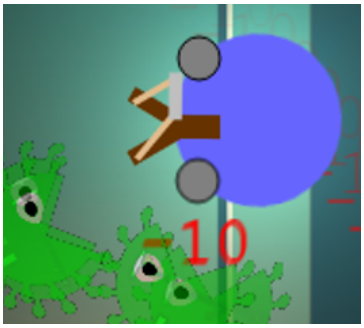


- Feature: Create player HP mechanic (2 points)
  As Player
  I want an HP mechanic
  So that I can see how close I am to failing

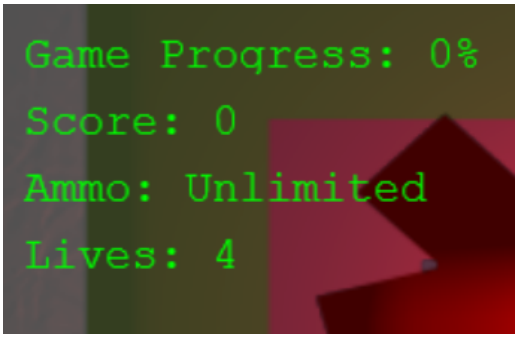
Design



Implemented



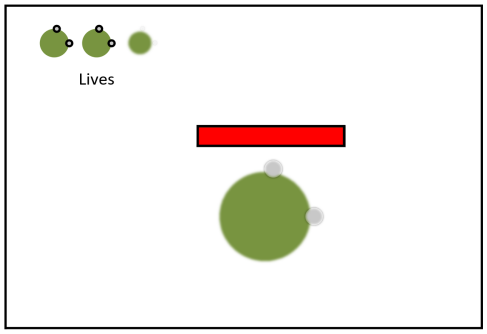
HP was transitioned into lives. From the above to below. HP wasn't as fun.

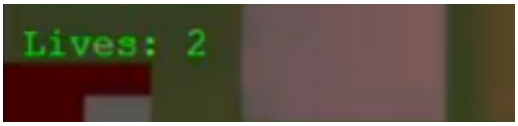


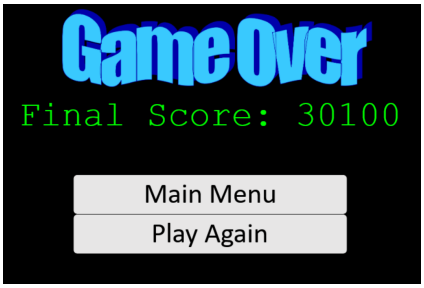
- Feature: Win Page (2 points)
  As a Player
  I want to see a win page
  So I know when I win the game

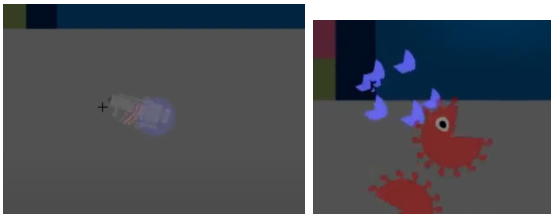
- Feature: Death and/or Respawn (2 points)
  As a Player
  I want there to be a consequence in failing
  So that I can feel accomplished in winning

Design **Implemented**



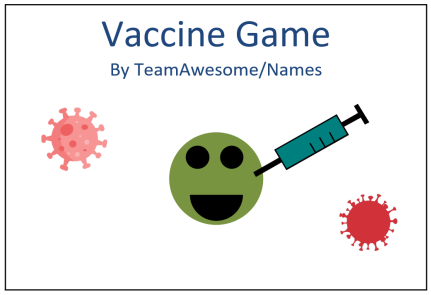






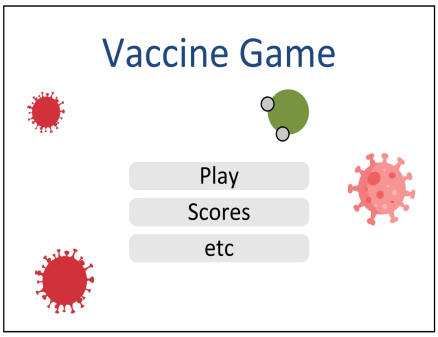
- Feature: (Low Priority) Loading/Credits Screen (2 credits)
  As a developer
  I want there to be a loading/credits screen
  So that I can load assets or let people know that I worked on this.

Design **Implemented**



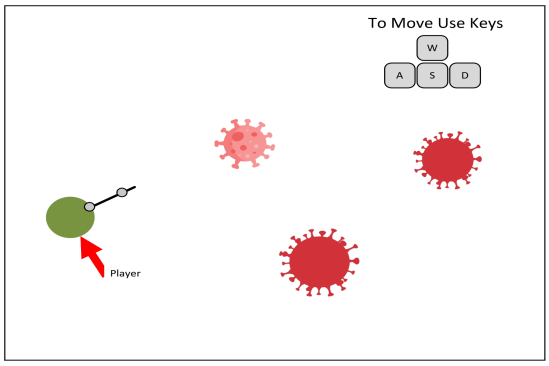
- Feature: (Low Priority) Main Menu Screen (2 credits)
  As a Player
  I want a Main Menu Screen
  So that I can play the game when I choose

Design **Implemented**



- Feature: (Low Priority) In Game Instructions (e.g. WASD to move/Defeat the viruses!) (1 point)
    As a Player
    I need instructions
    So I know how and why to play.

Design

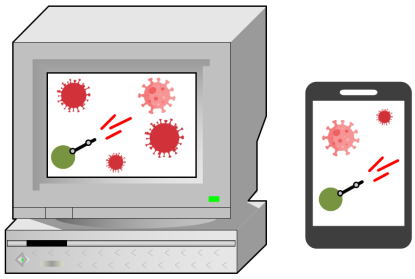


Implemented

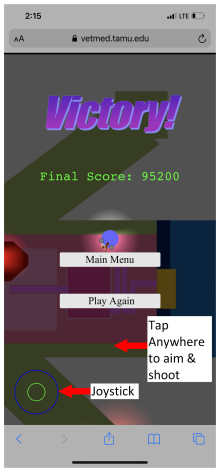
(Desktop instructions not shown on mobile)

- Feature: (Low Priority) Browser compatibility (2 points)
    As a user of a old phone model
    I want the game to work
    So I can play it

Design

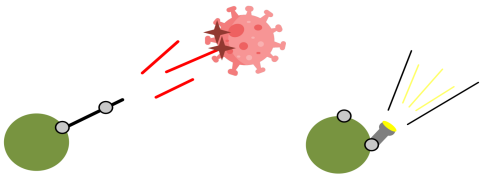


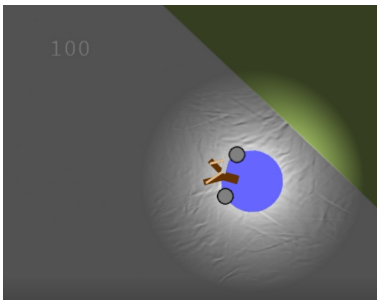
Implemented

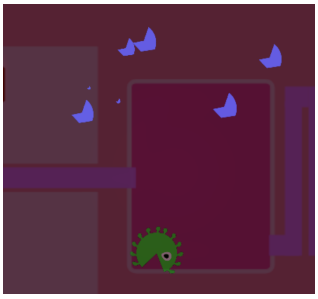


- Feature: (Low Priority) Effects (Lights, particles on bullet hit, etc..) (2 points)
    As a player
    I want the game to have fancy effects
    So that the game looks nice

Design **Implemented**

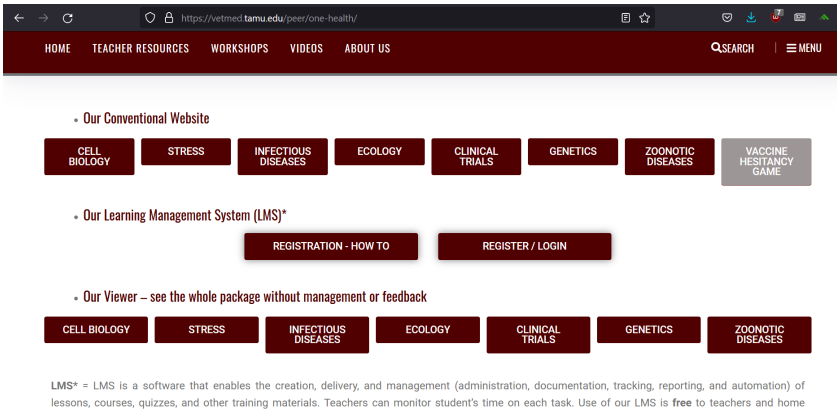




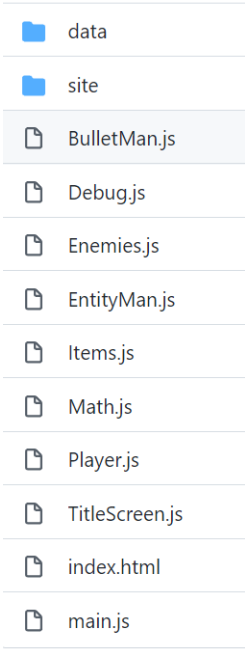


- Feature: Preparation for production site (getting sent code sent over & setup) (2 points)

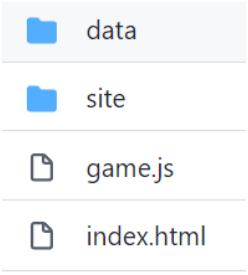
  As a player
  I want to be able to find the game online
  So that I can play the game



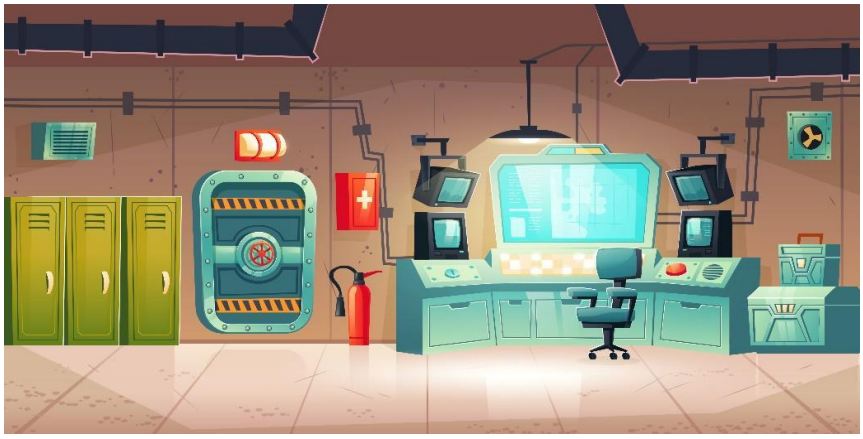
Development files



Deploy files



- Feature: Temporary Asset Replacement (2 points)
  As a developer
  I want textures that match the theme and are free use
  So the game is themed and the assets can be used.

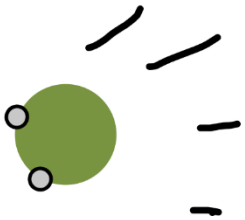


- Write iteration report 3 (2 points)
- Write iteration report 2 (2 points)

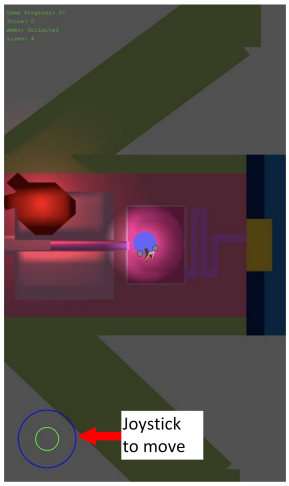
- Feature: Player Entity Movement (1 point)

  As a Player
  I want to move my character.
  So I can explore or dodge enemies in the game.

Design



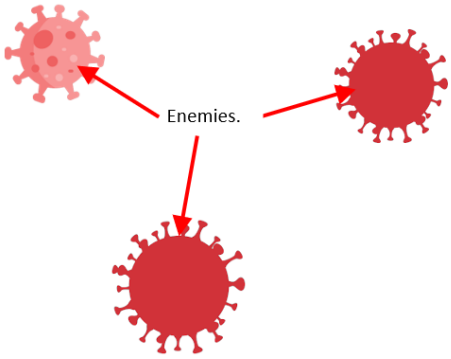
Implemented



- Feature: Create Enemy Entity (1 point)

  As a Player
  I want to encounter enemies (viruses)
  So that I can attack or run away from them.

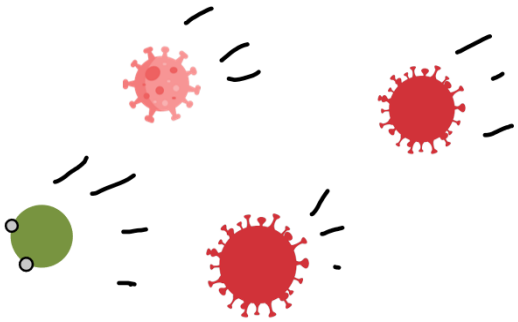
Design

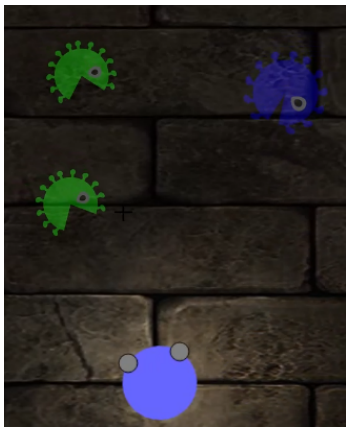


Implemented

- Feature: Enemy Entity Movement (1 point)
  As a Player
  I want enemies to move towards me
  So that there is something I need to overcome to win.

Design **Implemented**

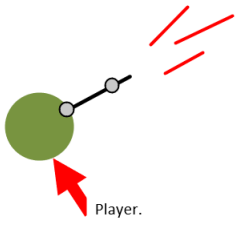


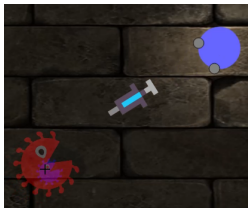


(Again, there is movement)

- Feature: Create Attack Methods (2 points)
  As a Player
  I want to shoot antibodies/tcells/etc at viruses
  So that I can attack and destroy them.

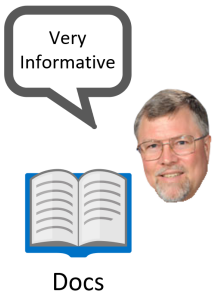
Design **Implemented**



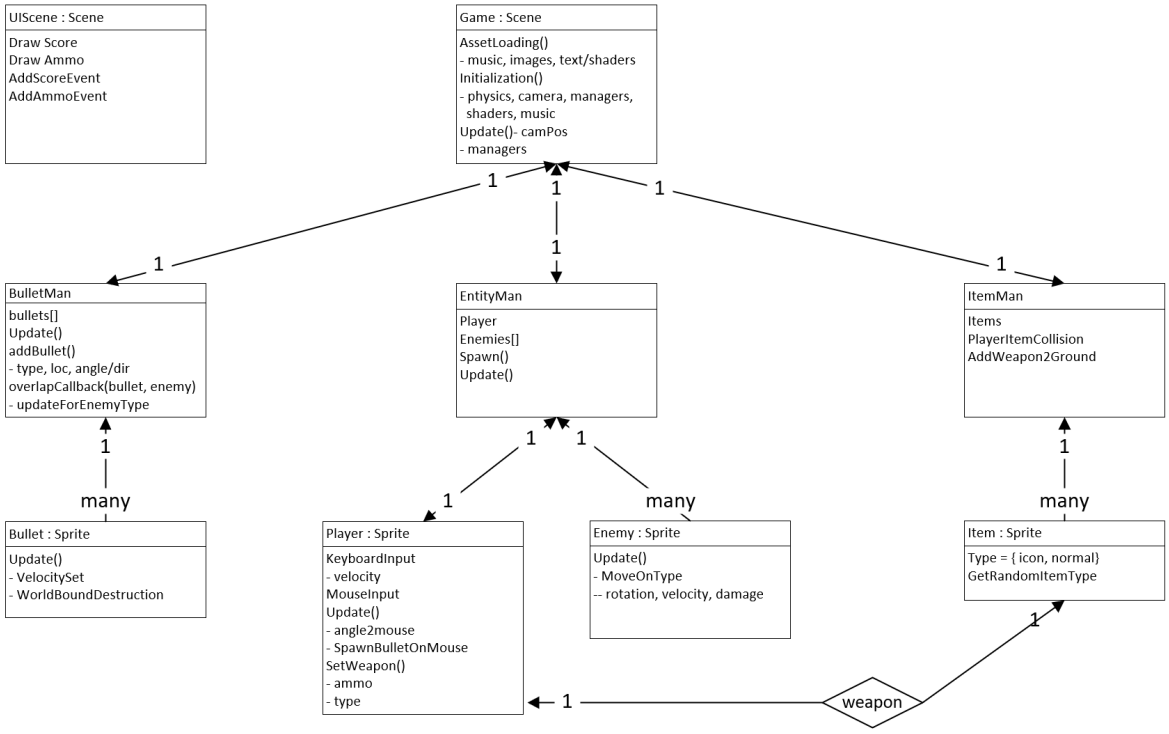


- Write iteration report 0 (2 points)
- Write iteration report 1 (2 points)
- Finalize idea for a vaccine game (1 point)
    - Discussed, refined, created multiple of, and got approval for a finalized idea of a vaccine themed game where the focus should be on the game aspect over education.
- Discussed, tested, and finalized a choice for a JavaScript game framework. (1 point)
    - As a developer,
      We want to choose a JavaScript framework
      So that we could build our vaccine game

- **Not complete**:
    - Feature: Game and Phaser framework documentation (2 points)
      As a developer
      I want to document how the game works
      So that in the future, developers can improve the game

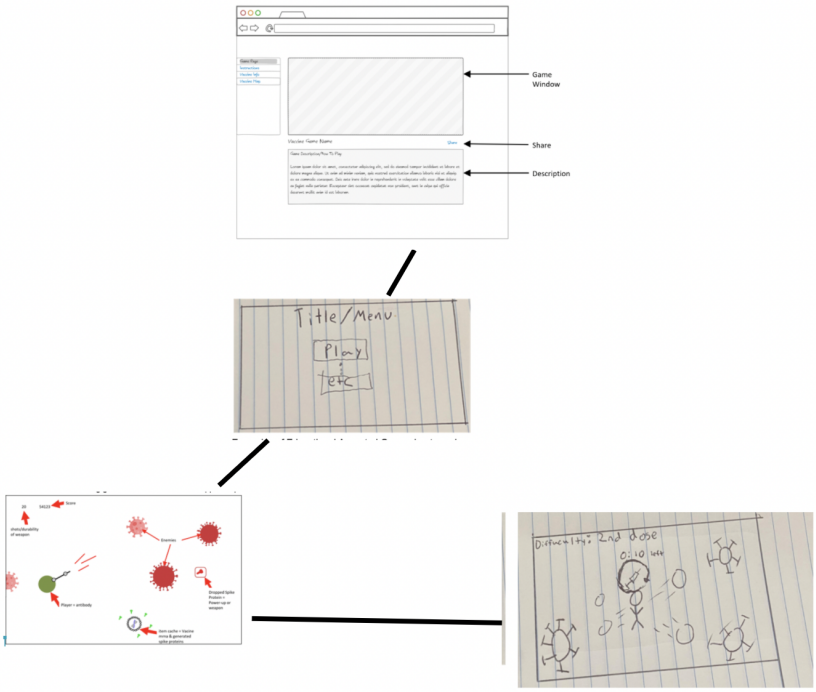


Design Diagram:



StoryBoard:

- The initial game mockup and storyboard.



Incomplete User Stories:

If you were not able to implement all the stories that you initially chose for this iteration, please list which ones and why not.

- Time constraints and changing game requirements limited the amount of stories we could implement. The user story from last iteration that was not able to be implemented this iteration was the Phaser framework documentation inside the repository. We managed to cover the behaviors of the game's expecture features and the game's mechanics in the test.md file, but we couldn't add more information about the phaser framework, however Phaser has a lot of useful documentation on how it works and its implementation online. Therefore, most of the questions would be answered easily online.

Scrum Iterations:

- There were no changes to the scrum master and product owner roles.
- Iteration 0: Met with Dr. Walker, explored game ideas, created mockups, created potential user stories for Iteration 1.
- Iteration 1:
    - Accomplished: Finalized the idea for the vaccine game, chose a JavaScript framework, and wrote iteration reports 0 and 1.
    - Points completed: 6
- Iteration 2:
    - Accomplished: Created player and enemy virus entities and their movements, showed short video demo of game to Dr. Walker, and wrote Iteration report 2.
    - Points completed: 7
- Iteration 3:
    - Accomplished: Added vaccine weapon drop entities that increases the amount of ammo a player can shoot at once, added entity collisions to pick up items and cause damage, added increasing difficulty of enemy viruses to kill as game progresses, and added score to keep track of points. Showed an updated demo of the game to Dr. Walker to share with VetMed faculty.
    - Points completed: 13
- Final:
    - Accomplished: Added dying and respawning, player hp/lives, effects, win and lose screens, credits screen, main screen, relevant open-sourced background and sound effects, check game compatibility on multiple browsers for desktop and mobile, and game instructions on main screen. We had one last meeting with Dr. Walker, made some adjustments, then sent the game to Samiksha Marne to deploy to the PEER website.
    - Points completed: 27

Customer Meeting Dates:

- (10/28/21) Exchanged messages over email with customer, Dr. Walker.
    - Send a document listing possible game ideas in order to get feedback and approval to begin work towards our finalized project idea.
    - Discussed that while the game ideas are fine, we need to be able to have the game done, tested, and deployed by the end of the semester. I.e. Try to make a simple game before trying to make anything fancy.
- (10/22/21-10/24/21) Exchanged series of emails with the web developer, Samiksha Marne, of the website where our completed project will be deployed to.
    - Discussed, if there would be any html or javascript constraints we would be under and where our finalized game would be hosted in order to account for existing html elements.
- (11/16/21) Showed a short video demo of the game to Dr. Walker. The hypodermic needles shoot at viruses as they converge onto the player, along with player movement and how covid cells attack you. Sample background music and attack audio was also included.
    - Demo displayed player and enemy virus entities with the player shooting
    - Discussed feedback on demo about improvements and next steps. Priority is to have basic game mechanics working--player shoots and kills viruses to achieve high score until player's HP bar is depleted, player can pick up vaccine item drops to increase the amount of ammo it shoots, randomize virus spawn locations, have background and sound related to theme--then focus on browser and mobile compatibilities.
    - Dr. Walker will share the video clip demo with VetMed faculty.
- (12/01/21) Sent an updated video demo of current progress to Dr. Walker for him to share with VetMed faculty.
- (12/08/21) Showed a demo video of the game to Dr. Walker to get final feedback before deploying.
    - Discussed feedback, testing, and where the game will be deployed to on PEER website. It was recommended to implement a win screen after scoring a certain threshold of points. Features would have to be tested manually.

BDD/TDD Process:

- Tests are usually written before implementation, but for a game that was built with the Phaser framework with evolving implementations dependent on customer input, we could not actually use TDD. We had to have users manually test and document the features in the game because normal testing frameworks (e.g. Jasmine, Mocha) do not work with Phaser. Thus, less TDD and more BDD was used in testing.
- Testing is essential in making the game work as implemented; however since it is not compatible with TDD, we had to manually test each element and use mostly BDD, then document it in the test.md file. Testing by playing the game ensures that the features work as implemented, makes finding glitches faster and helps gain knowledge on possible improvements to the game.

Configuration Management Approach:

- Discuss your configuration management approach. Did you need to do any spikes? How many branches and releases did you have?

  We used GitHub and AWS Cloud9 to manage our code and allow real time collaboration without having to worry about installing dependencies and other installation issues. After collaborating on code in Cloud9, it gets pushed to GitHub. We did have to use some spikes since there were times when we could not create specific user stories to tackle a problem. Once we learned what needed to be done, we were able to break a problem or feature down into user stories. We had four branches and one release.

Issues:

- Discuss any issues you had in the production release process to Heroku. Describe any issues you had using AWS Cloud9 and GitHub and other tools.

  We have not had any issues with the production release process to Heroku, AWS Cloud9, or GitHub. It took a while to adjust and become familiar with using Phaser. When testing, since we used Phaser, we could not use frameworks like Jasmine and Mocha for BDD, so we relied on manual testing.

Tools:

- Describe the other tools/GEMs you used, such as CodeClimate, or SimpleCov, and their benefits.

  Phaser 3 is an open-sourced JavaScript framework for making 2-D games. It is beginner-friendly and has many tutorials and a large active community. Also, it supports WebGL and Canvas and if WebGL is unavailable, it switches to the other. A disadvantage is that the code structure may not be cohesive.

Repo Contents:

- Make a separate section discussing your repo contents and the process, scripts, etc., you use to deploy your code. Make very sure that everything you need to deploy your code is in the repo. We have had problems with legacy projects missing libraries. We will verify that everything is in the repo.
    - Repo Contents:
        - src/data:
            - The data folder contains any and all assets that need to be loaded by the game in order to work. This includes graphics (gfx),

sounds/music (sfx), and effects (fx). We also include a text file listing sources of all assets and licenses they fall under which may be required by license.

- src/site:
    - The site folder contains three framework related scripts required for the framework to work properly. PhaserLightFix.js Is an expanded version of the phaser 3 framework used for debugging. PhaserLightFix.min.js is a minified version we use for deployment. The joystick script contains the plugin needed on mobile to add the joystick for movement. If we had any css files this is where we would place them also.
- src/*.js
    - This folder contains the majority of our work done for the development version of the game. index.html is almost blank beyond some css formatting and bringing together the scripts in order to work. main.js hosts the main game scene and is also the file where we configure and first start phaser. Note that part of phaser arcade (velocity) is in debug mode, this is due to a fix we patched into the phaser framework that draws a single pixel in the top left hand corner of the screen. For some reason this fixes the issues phaser has with lights disappearing (culled) when they are not supposed to be. The debug for velocity has been moved to under the same debug for body instead. The rest of the files are intuitively named. Player.js contains player code, enemy.js contains enemy code, and Entity.js brings these two entities under one roof for handling.
    - If you are in the deploy branch we combine all the js files into one file (excluding phaser/plugins), remove exports and imports, and use uglifyjs to minify and compress the game. Make sure to switch to the minified version of phaser to use before you deploy the game to the vetmed site.

- Running the game: In order to run the game locally you will need to run a webserver of your choosing to serve the files. There are plenty of command line tools available that will do this for you. This is **NOT** anything specific to our game the same will be true for any HTML project that's larger than a single standalone html file. Through Cloud9 you can open up the index.html click preview, preview index.html. I personally found this a little slow and prone to issues compared to using a small server to serve the files.

        - If you really want to use the same tool as we did we used https://github.com/juj/emrun. This is not what you would normally choose to serve your files. I personally use this as it is part of another framework I am used to and helps with that particular framework. It will work the same as any other tool though.
            - python3 emrun.py --hostname 127.0.0.1 or 0.0.0.0 --port 8080 --no_browser index.html
    - Before deploying to the vetmed website, you are required to test that the game works on a wordpress website. This can either be done online through a tool of your choice or serverpress if you wish to run it locally. We used serverpress for testing our deployment as it was recommended.
        - https://www.youtube.com/watch?v=9sRlqzBmn0g
            - It's pretty simple to do. Create a site in serverpress, then through your files into a folder at the root location of your site on your computer and navigate to that folder under the site.
            - The video above shows you all you need to know after getting serverpress setup.
    - In order to actually deploy the game to vetmed site once tested, put the files into a zip folder and send them to whoever is in charge of the site development. The site dev may change between classes so ask your customer (professor) to put you in contact with this person.