1. the name of your game, your names, and your CSc-165 section number(s)
   1. Duum, Aaron Goodlund (section 1) and Sheridan Lynch (section 2)
2. at least one image (screenshot) showing a typical scene from your game being played
3. instructions for compiling and running your game, including the network server
   1. run buildTAGE.bat, compile.bat, runServer.bat, then run.bat to join into the server that people can join
4. how to play your game, including what things happen and how the scoring works
   1. you move along 2 axes and rotate, able to fire out arrows along the red line at any point
5. what player controls are available (what all keyboard/gamepad buttons do, etc.)
   1. Keyboard
      1. Move: WASD
      2. Rotate: Left and Right arrow keys
      3. Shoot: Space bar
      4. Change view: V
      5. Change character: C
      6. Toggle Flashlight: F
   2. Controller:
      1. Move: Left Stick
      2. Rotate: Right Stick
      3. Shoot:
      4. Change View:
      5. Change Character:
      6. Toggle Flashlight:
6. a description of the lighting used in your game, including any lights that turn on/off
   1. there’s a dim ambient light and 3 colored spotlights pointing down from the jellyfish in the scene
   2. the player avatar also has a spotlight that works as a flashlight in front of them
7. a brief summary of any changes (or none) that you made to the network protocol
   1. A Message object was created that contains the basic data types sent by the client
8. a list of changes and additions that you made to TAGE
   1. Pitch, Yaw, OrbitCamera3D, and Networking were all required in earlier milestones
   2. Message.java
   3. HUDElement/HUDmanager
      1. Modularized HUD elements so text could be more easily placed on the screen, including having more than 2 elements maximum
9. a statement indicating the (1) genre, (2) theme, (3) dimensionality, and (4) activities utilized in your game (see week 1 notes -- chapter 00 -- for examples)
   1. Genre: Twin Stick shooter
   2. Theme: Underwater
   3. Dimensionality: 2-2.5D
   4. Activities: Combat
10. an explanation of where (in the game, not the code) each project requirement is visible
    1. External Models
       1. Every moving model, aside from the generic ghost, were created by us
    2. Networked Multi-Player
       1. Player ghosts are visible when active
    3. Skybox and Terrain
       1. Both are best visible from the flat view that happens after pressing V
    4. Lights
       1. Red, Green, and Yellow spotlights are visible along the lower left, lower middle, and upper right of the visible terrain
       2. The light directly in front of the avatar can be toggled on and off with the F key
    5. HUD
       1. At the bottom left of the screen is the player’s health, which counts to 0 when they get hit
    6. 3D Sound
       1. A dead enemy sits near the center of the stage that emits a constant bubbling noise
       2. When shooting there is a sfx for shooting a crossbow
    7. Hierarchical SceneGraph
       1. The crossbow and laser are both child objects of the avatar
    8. Animation
       1. Using the movement commands as the diver causes its walk animation to play
11. A list of the requirements that you weren’t able to get working
    1. NPCs
    2. Physics
12. any technique you used in your game that goes beyond the requirements
    1. PositionalColor is a render state that causes a model with a null texture to have its colors based on the rasterized position of its vertices in the model space
13. the contributions of each team member, including who designed which model(s)
    1. Aaron:
       1. Movement actions
       2. manualCrystal
       3. PanCameraAction
       4. ShootAction
       5. ToggleFlashLightAction
       6. ChangeCharacterAction
       7. Message.java
       8. Unused nodeControllers
          1. BobController
          2. RollController
       9. Changes to RenderObjectStandard, RenderStates, and the standard fragment shader for positional color
       10. HUDmanager
       11. HUDElement
    2. Sheridan:
       1. Heightmapping
       2. Physics
       3. FollowPlayer
       4. NPC’s/AI
       5. Assisted with
          1. ProtocolClient
          2. Audio
       6. Unused Functions
          1. calculateAvatarCollision
          2. getGhostShape
          3. getGhostTex
14. a list of assets that you created yourself (models, textures, heightmap, etc.), and items obtained that were distributed in CSc-155 or CSc-165
    1. CSC155/165
       1. Ice.jpg
       2. dolphinHighPoly.obj
       3. dolphinLowPoly.obj
    2. Aaron:
       1. Crossbow\_empty/crossbow\_loaded.obj
       2. Diver.obj + skeleton, mesh, and animation
       3. Jellyfish.obj
       4. Spear.obj
       5. ULPD.obj
       6. “Unda da sea” skybox folder
       7. Diver\_UV.png
       8. Heightmap map.png
       9. Oiter.png
       10. Sand\_watery.png
       11. Sand.png
       12. ULPDuv.png
       13. Unused textures
           1. Custom\_mouse\_test.png
       14. Mouse reticle.png
    3. Sheridan:
       1. PufferFish\_Angry.obj
       2. PufferFish\_Calm.obj
       3. Flipped.obj
       4. PufferFish\_Angry\_Spiney.png
       5. PufferFish\_Angry\_SpineyAlt.png
       6. PufferFish\_Angry\_Spineless.png
       7. grass.jpg
       8. hills.jpg
15. Source and evidence of permission for any item (models, textures, etc.) not listed in #14
    1. sound\_ahead\_\_bubbles\_low\_4.wav
       1. <https://freesound.org/people/sound_ahead/sounds/567455/>
       2. Attribution 3.0: "You are free to share (to copy, distribute and transmit) and to remix (to adapt and modify) as long as you credit the author of the sound."
          1. Sound\_ahead is the account name of the author
    2. 752207\_\_dude\_x-soundlab\_\_crossbow-fire-vii.wav
       1. <https://freesound.org/people/DUDE_X-SoundLab/sounds/752207/>
       2. Creative Commons 0: "You can copy, modify, distribute and perform the sound, even for commercial purposes, all without the need of asking permission to the author."
          1. Dude\_x-soundlab is the account name of the author
16. which RVR-5029 lab machines (at least two – it’s networked!) on which your program was tested and is known to work correctly on
    1. ECS-MYST, ECS-PACMAN