

Game Proposal: Bnuuy's Ship

CPSC 427 – Video Game Programming

Team: Bnuuy Savers

Brian Moniaga 27894880

Chinmay Bhansali 61816229

Clare Pan 95483459

Dayshaun Lee 76084086

Lily Zhang 86004801

Story:

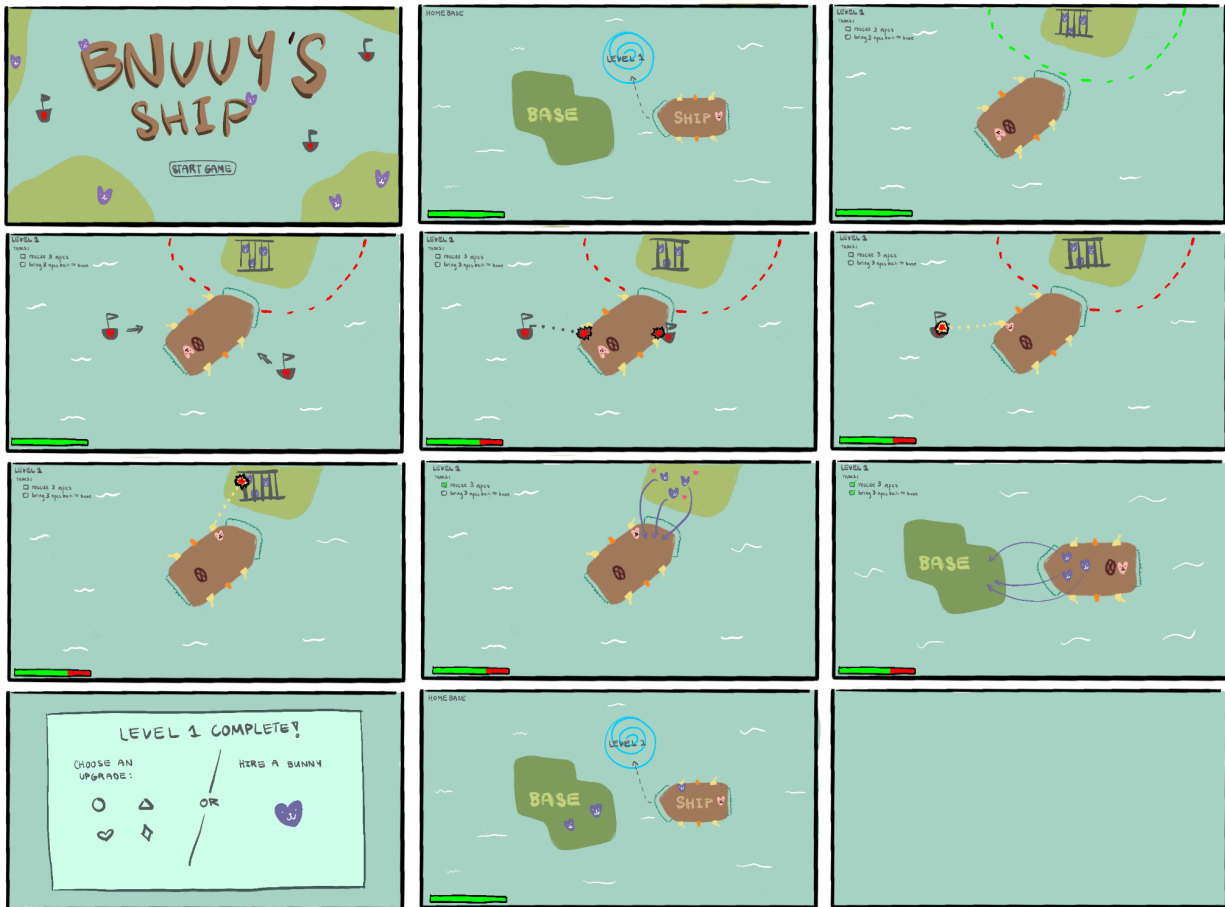
Briefly describe the overall game structure with a possible background story or motivation. Focus on the gameplay elements of the game over the background story:

Our game sets in a flooded world where you captain the Bnuuy Fleet searching for bunnies among scattered islands while rescuing them from pirates.

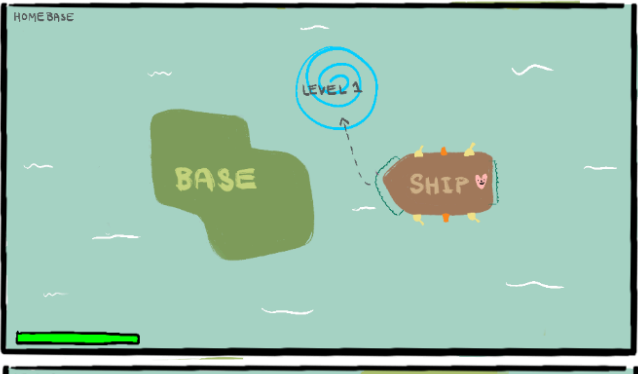
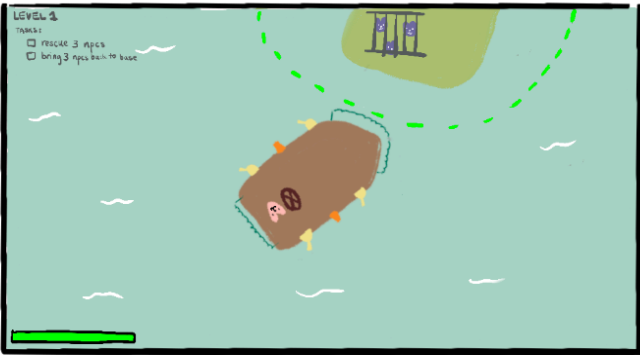
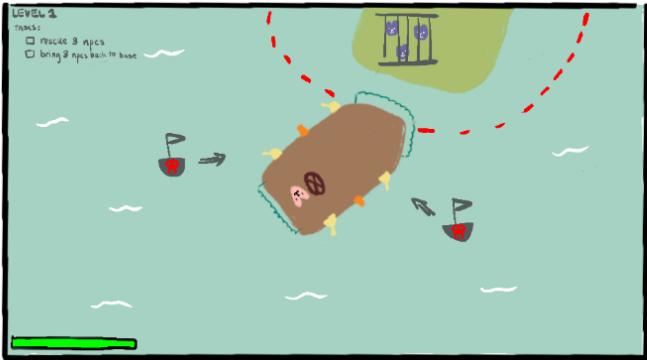
Bnuuy must navigate through unknown waters and defeat pirates to rescue the bunnies and bring them back to the base island. As you bring back more bunnies, you will be able to upgrade your ship and weapons to defeat more powerful enemies.

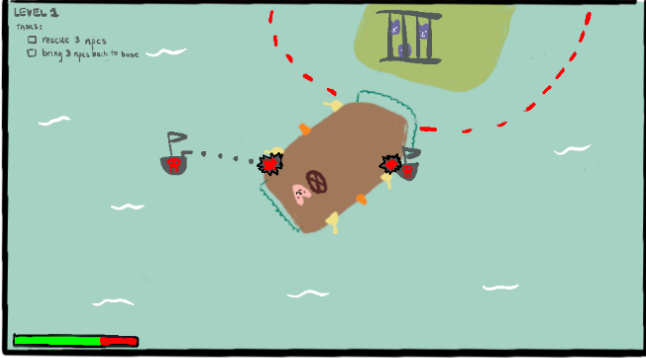

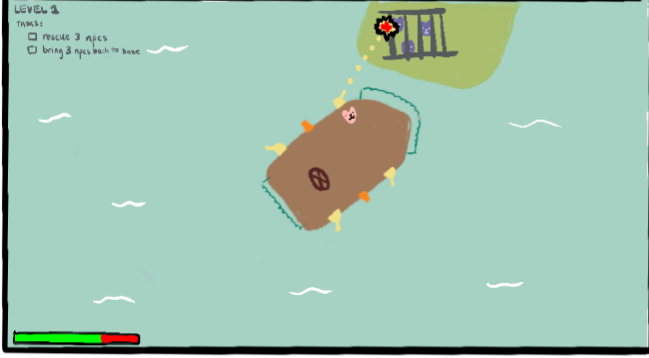
Scenes:

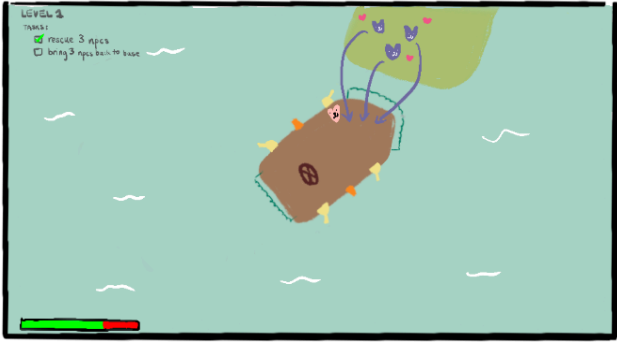
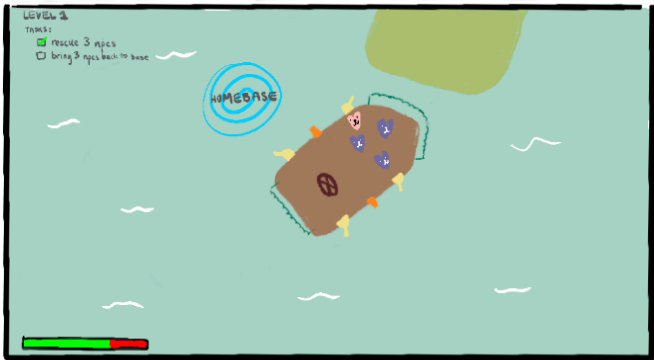
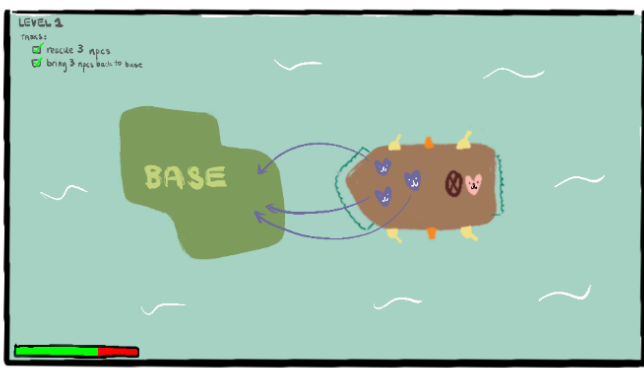
Produce basic, yet descriptive, sketches of the major game states (screens or scenes). These should be consistent with the game design elements, and help you assess the amount of work to be done. These should clearly show how players will interact with the game and what the outcomes of their interactions will be. For example, jumping onto platforms, shooting projectiles, enemy pathfinding or 'seeing' the player. This section is meant to demonstrate how the game will play and feeds into the technical and advanced technical element sections below. If taking inspiration from other games, you can include annotated screenshots that capture the game play elements you are planning to copy.


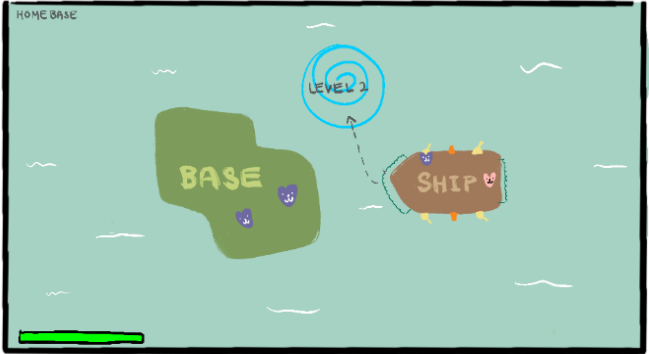


Scene #	Description
1.	<ul style="list-style-type: none"> - Starting game screen
2.	<ul style="list-style-type: none"> - Game starts near the home base - Player on the ship and can control the ship

	<ul style="list-style-type: none"> - Player can use the steering station and move the ship to the blue whirlpool to begin the current level
<p>3.</p> 	<ul style="list-style-type: none"> - The level's task will be displayed on the top left corner <ul style="list-style-type: none"> - Rescue X bunnies - Bring X bunnies back to base - Player will move the ship closer to island with bunnies in order to rescue them
<p>4.</p> 	<ul style="list-style-type: none"> - When the ship is within the island's radar, enemy ships will spawn (swarm behaviour) - The enemies will follow player's ship (simple path finding)
<p>5.</p>	<ul style="list-style-type: none"> - Depending on the enemy's behaviour, they will try to attack and damage player's ship (Enemy group behaviour; Cooperative Planning) <ul style="list-style-type: none"> - Enemy can shoot bullets at player's ship

	<ul style="list-style-type: none"> - Enemy can bash their ship against the player's ship, resulting in both ships taking damage - Player's ship's health bar will change based on the amount of damage taken <ul style="list-style-type: none"> - If health is depleted, game over. Player will return back to the home base with their last saved point (before the current level began)
<p>6.</p> 	<ul style="list-style-type: none"> - The player can exit the steering station and use the turret or canon station to shoot down the enemy <ul style="list-style-type: none"> - The player can only use one station at a time
<p>7.</p> 	<ul style="list-style-type: none"> - To rescue the bunnies, the player must shoot and destroy the cage that is trapping the bunnies.
<p>8.</p>	<ul style="list-style-type: none"> - Once the cage is destroyed, the bunnies can move onto the player's ship

	
<p>9.</p> 	<ul style="list-style-type: none"> - Once X number of bunnies has been rescued, a blue whirlpool will appear - If the player moves the ship towards the blue whirlpool, it will take them back to the home base
<p>10.</p> 	<ul style="list-style-type: none"> - When back at the home base, player can finish the level by moving the rescued bunnies to the home base island.
<p>11.</p>	<ul style="list-style-type: none"> - After the level has been completed, depending on the conditions, player may have the option to: <ul style="list-style-type: none"> - Upgrade a component

	<p>of the ship OR</p> <ul style="list-style-type: none"> - Recruit a bunny to help out on the ship (cooperative planning)
<p>12.</p> 	<ul style="list-style-type: none"> - Player can continue the game by moving to the next level <ul style="list-style-type: none"> - Ship health will be restored

Technical Elements:

Identify how the game satisfies the core technical requirements: rendering; geometric/sprite/other assets; 2D geometry manipulation (transformation, collisions, etc.); gameplay logic/AI, physics.

Rendering

- Pixel art style
- 2D top down
- Animated sprites for characters

Assets

- We are planning to use assets found online (pixel art) and create our base ship sprites using the same theme
- Royalty-free music for different phases in the game

2D Geometry Manipulation

- we need the Ship to move around
- we need the Ship module connection and placement system
- We need precise collision detection for projectiles
- as well as the Collision between the ship and enemies/islands

Game AI

- Crew members:
 - The crew members station themselves on the remaining spots
 - The crew members aim weapons at the nearest enemies to their weapon
- Enemy behaviour:
 - The enemies will path straight toward the ship, and attack once they are in range

Physics

- Projectile trajectories for cannons and other weapons
- Ship movement and player movement within the ship

Advanced Technical Elements:

List the more advanced and additional technical elements you intend to include in the game prioritized on likelihood of inclusion. Describe the impact on the gameplay in the event of skipping each of the features and propose an alternative.

Graphics

- Simple rendering effects (Basic)
 - Priority: High
- Particle system.
 - Priority: High
- 2D Dynamic Shadow.
 - Priority: Low

Physics & Simulation:

- Basic physics (Basic)
 - Priority: High
 - If skipped, our game will not work.
- Complex prescribed motion (Basic)
 - Priority: High
 - If skipped, we can perhaps use a simple linear motion instead.
- Precise collisions
 - Priority: high
 - If skipped, use simple calculation for collision such as distance threshold between ship and island
- Articulated Motion
 - Priority: low
 - A nice-to-have feature for one of our upgrades. But if this feature is skipped, we simply won't have this upgrade.

Artificial Intelligence

- Simple path finding (Basic)

- Priority: High
- If skipped enemy can't find the player
- Swarm behaviour
 - Priority: medium
 - If skipped, then enemies will usually not attack as a swarm.
- Enemy group behaviour; Cooperative Planning
 - Priority: medium
 - If skipped, then enemies will not attack as a swarm.
- Cooperative Planning
 - Priority: high
 - Need the bunnies to aim when they are put at canon stations

Software Engineering

- Reloadability (basic)
 - Priority: High
 - If skipped, player will have to beat all the levels in one instance
- External integration (basic)
 - Priority: High
 - If skipped, we will have to implement a lot of tools from scratch.

UI & IO

- Camera controls (basic)
 - Priority: High
 - If skipped, camera cannot follow the ship and the game will have to be one very zoomed out screen
- Audio feedback (basic)
 - Priority: High
 - If skipped, the game will be boring and have no audio

Quality & UX

- Basic integrated assets (basic)
 - Priority: High
 - If skipped, our game will be super ugly and unattractive
- Game balance (basic)
 - Priority: Medium
 - If skipped, our game might feel too hard or too easy.
- Numerous sophisticated integrated assets
 - Priority: Low
 - If skipped, our game feels less engaging.
- Story elements
 - Priority: Low
 - If skipped, our game has less personality to the characters

Devices:

Explain which input devices you plan on supporting and how they map to in-game controls.

Keyboard (Player): W (Up), A (Left), S (Down), D (Right), Space (Use/exit a station)

Keyboard (Ship): W (Forward), A (Turn left), D (Turn right), S (Stop / Move backwards)

Mouse: aim weapons, left click or hold to shoot

Tools:

Specify and motivate the libraries and tools that you plan on using except for C/C++ and OpenGL.

- EnTT for ECS
- Aseprite, Krita, Procreate for assets

Team management:

Identify how you will assign and track tasks and describe the internal deadlines and policies you will use to meet the goals of each milestone.

- Notion board to track tasks
- Aim for one meaningful contribution per member per week
- Setup additional meetings if necessary
- Follow the TWA

Development Plan:

Provide a list of tasks that your team will work on for each of the weekly deadlines. Account for some testing time and potential delays, as well as describing alternative options (plan B). Include all the major features you plan on implementing (no code).

Milestone 1: Skeletal Game

It should incorporate basic rendering, input-driven response, 2D motion, correct basic collision handling, event-driven/random response, and a minimal set of assets.

Week 1:

- Main Menu, and game scene (scene manager)
- Map
 - Island (survivor's island)
 - Island base
 - survivor on island
- Player Movement
- Ship Movement
- Basic Rendering
- Moving camera
- Input system (input device)

Week 2

- Saving bunnies mechanic (Rescue and dropoff bunny, but not station them. As well as shooting the cage that holds a bunny)
- A weapon in the ship (functionality)
- Health bar

- Enemies (stationary)
- Basic Rendering

Milestone 2: Minimal Playability

For this milestone, you should continue to support **all required** skeletal game features and fix any game-stopping bugs (P0 and P1). You should augment those features with core gameplay logic, incorporate additional assets and play elements that allow for non-repetitive gameplay, fix all known bugs, and perform playability testing.

Week 1

- Enemy pathfinding (week 1-2)
- Enemy spawning logic (week 1-2)
- Ship module system (week 1-2)
- Level system, and Basic ship upgrade (week 1-2)
- Animated sprites
- Player Collision
- Ship collision handling (colliding with islands)

Week 2

- Serialization (Saving levels)
- Bunny gets to use stations
- Audio assets
- More Rendering

Milestone 3: Playability

For this milestone, you should have a completely playable game, continue to support all required features from prior milestones, and have fixed the majority of your latent bugs. In addition, you should **now also include advanced features** such as detailed geometry, non-linear motion, and time-stepping based physics. Your game should provide robust continuous play with no memory leaks, crashes or glitches.

Key to this milestone is that you test the playability of all new features and ensure alignment with your team's game development plan. You are encouraged to ask people outside your team to play test your game, so you can fine tune your gameplay and interface.

Week 1

- Complex enemies (Boids enemy, new enemy, etc.) (Week 1-2)
- Advanced ship upgrades (different weapon kinds, shields, etc.) (Week 1-2)

- Particle effects
- Bunnies automatically use weapons
- UI (Pause Menu, Death Menu)

Week 2

- More rendering (bunnies working on island base)
- More assets (different weapons, sound effects, etc.)
- Cutscene (Dialogues when bunnies get rescued, etc.)

Milestone 4: Final Game

The final version of your game should support robust and continuous gameplay and integrate advanced game elements. It should be self-contained, i.e., players should be able to play the game with no outside help or instructions other than those provided by the game itself. You should implement one or more additional advanced gameplay feature (see below) and you are welcome to incorporate advanced features using third-party libraries (verify with TA). Your final video will demonstrate your final game, highlight the advanced features, and provide a review of your game's development and evolution.

Key to this final milestone is that you test the playability of all new features and ensure alignment with your team's game development plan. You are encouraged to ask people outside your team to play test your game, so you can balance and fine tune your gameplay and interface.

Week 1

- Game balancing
- Narratives
- Code efficiency (Optimize game performance)
- More levels
- Additional assets refinements

Week 2

- Make the final video
- Test and fix any bug left