Game Proposal: Games of Throne

CPSC 427 Video Game Programming

Team: Electric Artists

Toby Li (45239803)

Angela Li (35560911)

Ansh Gandhi (52839735)

Stephanie Song (70327564)

William Zhou (56572738)

Story & Setting:

In the chaotic and humorous kingdom of Astraeth, you are a spy on a mission to overthrow the reigning monarch. Starting from the lowest rank, working in the royal kitchens, your goal is to ascend the hierarchy by battling your way through various bosses until you claim the Iron Recliner Throne.

Your journey will involve mastering a short-range melee combat system, utilizing light and heavy attacks, dodges, and a unique combat resource called FLOW. You will also have to go around the maps. As you defeat enemies and bosses, you will unlock new weapons and abilities, with the ultimate goal of defeating the monarch and seizing control of the kingdom. And because you're a spy, you're used to learning things on the fly, allowing you to learn new spying skills.

Progression & Levels:

Each level in the game represents a step up the royal hierarchy, with increasingly difficult enemies and bosses that challenge your combat skills. The game's structure follows a progression from the kitchens to the throne room, with each boss guarding the next stage of your journey:

Royal Kitchen (First Level – Boss: The Grumpy Chef)
Castle Hallway (Second Level – Boss: Royal Guard)
Alchemist Chambers (Third Level – Boss: The Prince)
Throne Room (Final Level – Boss: The King)

As you advance, you unlock new tools, secret rooms with upgrades, and new weapons to aid you in battle.

Combat System Overview:

FLOW is the core combat resource that rewards skillful play, such as perfect dodges and combos.

1. Light Attacks:

Description: Quick, low damage strikes that can be chained into combos. Any light attack will give a small amount of FLOW if it hits an enemy.

2. Heavy Attacks:

Description: Slower, high-damage strikes that leave the player vulnerable during the charge-up.

Charge Time: Holding the right mouse button increases the FLOW (if it's not at max. already). Upon release of the button, the heavy attack is released.

Wide Arc: Heavy attacks can hit multiple enemies or break defenses.

FLOW Charge: When FLOW is full, it will be consumed to charge the heavy attack, making it significantly stronger (deals more damage).

3. Dodge Mechanism:

Description: Allows players to sidestep or roll to avoid enemy attacks.

Perfect Dodge: Dodging right before an attack increases FLOW and enables faster counterattacks.

4. Energy Mechanism:

Description: Sprinting, dodging, light attacks, and heavy attacks all consume energy. When the energy is empty, the player cannot use combo attacks or heavy attacks. When the player is not performing any of the aforementioned actions that consume energy, the energy will be rapidly restored.

No Energy: When you have no energy, you cannot sprint, perform combo attacks, and perform heavy attacks. But you can still dodge.

5. Player Abilities:

The player gains a new ability after beating a boss:

- chef: gain ability to be invisible for 5 seconds; 5 second cooldown.

- Knight: gain ability to teleport to the back of the enemy and perform a backstab that deals critical damage; 12 second cooldown.
- prince: gain ability to increase damage and attack speed for 10 seconds; 40 second cooldown.

The player can only use one ability at a time.

Weapon System:

Weapon 1: Dagger, Light Attack Damage & Speed Boost

These weapons focus on enhancing the Spy's light attacks, making them faster and more damaging with each level of rarity.

Attack Speed: 0.8s/attack (-0.2s/tier)

Attack Damage: 10 (+10/tier)

- Effect: increases light attack speed by **0.2s** and damage by **10+** for each tier.

Weapon 2: Sword, FLOW-Stacked Damage

These weapons focus on increasing the effectiveness of heavy attacks and FLOW-charged abilities.

Attack Speed: 1.5s/attack (-0.3s/-0.2s per tier)

Attack Damage: 20(+15/tier)

- Effect: light attacks charge FLOW 10% faster for each tier.
- Special Ability: heavy attacks deal 100% more damage for each tier.

Rarity:

Each weapon has three ranks, and the chances to pick up each tier's weapon in each map are scaled:

Map Level	Basic (Tier 1)	Rare (Tier 2)	Legendary (Tier 3)
Map 1	70%	25%	5%
Map 2	50%	40%	10%

Map 3	30%	50%	20%
Map 4	10%	50%	40%

Minions Design:

All minions share the same mechanics but feature different art designs. Their stats scale progressively across the maps.

1. Melee Minions

Attack Mode: Single-Target Slash or Thrust

Health: 100

Attack: 10

- **Description:** A close-range melee strike aimed at a single target. This could be a slash, thrust, or jab depending on the minion's weapon.
- **Behavior:** Melee minions approach the player, prioritizing a single, deliberate strike that deals moderate to heavy damage. The attack's speed and damage will scale with each level.

2. Ranged Minions

Attack Mode: Single-Target Projectile

Health: 50

Attack: 10

- **Description:** A ranged projectile aimed at a single target. Depending on the minion, this could be a thrown object (e.g., a piece of food, fireball, shadow bolt).
- **Behavior:** The minion focuses on a single player and fires a projectile in the play's direction. The speed and damage of the projectile will scale up as the game progresses.

Boss Design:

Chef:

- High damage but vulnerable. Basic damage: 100. Health: 350.
- Four attack ways go in turn with five seconds gap:

    - Tomato Frenzy: Shoot tomatoes in your way, each tomato deals 2x basic damage, if being hit by tomatoes in series, each one deals half of the damage than the previous one.
    - Pan Boomerang: Throw the pan in your way, and the pan flies back to the chef, deals 2.5x basic damage.
    - Dash towards you and slam the ground and stun the player for 1 second, while dealing 0.5x basic damage.

Knight:

- Low damage but tanky. Basic damage: 50. Health: 750.
- Three attack ways go in turn with eight seconds gap:
    - Dash to and attack the player when in range with the sword, each strike deals 1x basic damage. Each strike has a two seconds gap.
    - Hold the shield for three seconds, if the shield is hit by the player, the shield immediately breaks and deals 1.5x basic damage with a 0.5 second stun.
    - For three times, dash for a short distance in the direction of the player and swing the sword, every strike deals 2x basic damage. There's a 1.5 second gap between each dash. After the last strike, stick the sword to the ground, forming a field around him that deals 3x basic damage.

Prince:

- Mid damage and mid tanky. Basic damage: 75. Health: 400.
- If the player gets close to the prince, the prince swings his wand every 3 seconds dealing 1x damage.
- Every 15 seconds, the prince teleports behind the player and quickly fires a pulse that deals 3x damage.
- Every 20 seconds, the prince forms a hemisphere field around the player, the field fully constructs after 3 seconds, after which the player cannot go in or come out of the field. After the field is constructed, there are magic lasers inside the wall for 5 seconds, dealing 2x damage.
- When the prince’s health bar reaches 66% and 33%, the prince summons 5 minion spirits to follow and attack the player at the minion’s damage level. The spirits can be defeated with one hit.

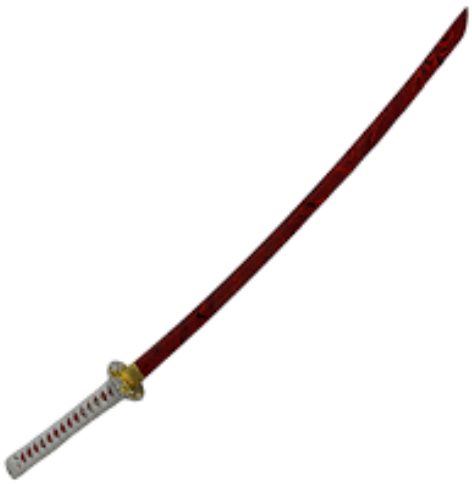
King:

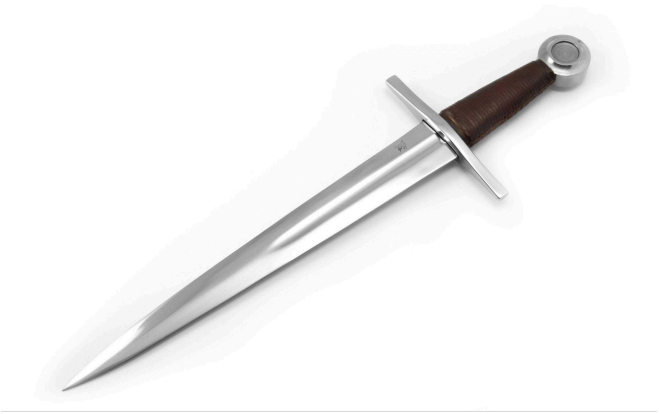
- Surprise! This final boss respawns with stronger abilities after being beaten. High attack damage and high health.
- First Phase:
    - Basic damage: 150. Health: 500.
    - Every 25 seconds, shoots a laser that spins around and deals 2x basic damage.
    - Every 20 seconds, dashes two times slowly towards the player and hits the player with his cane two times, dealing 2x basic damage.
    - Every 30 seconds, summon 2 soldiers that follow the player and deal 1x basic damage per hit. Soldiers have a health of 100 and damage of 10.
- Second Phase:

- Basic damage: 200. Health: 600.
- Gains a shield of 300 every 60 seconds.
- Every 25 seconds, make fire rain in circular areas that deals 3x basic damage upon stepping on it.
- Every 20 seconds, dashes towards the player's back very fast for three times, if he bypasses the player, it deals 3x basic damage.
- The king leaves a disguise at his original place and teleports to the player's back and uses his cane to deal 4x basic damage after 5 seconds. The player deals no damage when hitting the disguise.
- Upon the king's health bar reaching 66% and 33%, summon 5 soldiers that follow the player and deal 10 damage per hit. Soldiers have a health of 150. If the soldiers are not defeated after 20 seconds, the king empowers them, making them bigger and deal 2.3 basic damage per hit. If the soldiers are not defeated after 40 seconds, the king restores (the number of soldiers left) * 100 health.

Scenes:

katana & dagger:





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:

- Visual effects for weapon attacks, boss special attacks, player dodge, and on death animations. Implemented with shaders and animated sprites.
- Ability UI showing whether ability is ready, and its cooldown if it's not ready.
- Health and energy bar UI.
- Flow UI with a circular progress bar, once full indicates that Flow is full.
- Camera follows the player so the player does not go out of view.

Assets:

- Geometry of the environment, with walls to block off areas inaccessible to the player
- Invisible bounding boxes for every object and character in the world that are used for collision detection
- Sprites for:
    - environment (boxes / walls) to block movement
    - player
    - minions and their weapons
    - bosses and their weapons
    - player weapons
    - chest
    - fountain
    - boss ranged attack shockwaves
- Background music, unique to each level
- Sound effects on events such as hit, death, light attack, heavy attack, boss attack

Physics:

- Collision between player and the obstacles in the environment
- Collision between player and enemies, and enemies with each other
- Collision between ranged attack waves and the player - deals damage to the player
- Collision between player's weapon and enemies/enemy's weapon and player - deals damage
- Kinematics for the player and bosses. They have dashes that increase their momentum rapidly

Gameplay Logic:

- Player Controls (keyboard and mouse):
    - WASD for movement
    - Hold down Shift to sprint, which costs energy continuously
    - Space for dodging in the current direction of movement, player is invincible during the dodge

    - Mouse position controls where the player is facing, which controls where the attacks will deal damage
    - Left Click to light attack
    - Right Click to heavy attack; hold down right click to charge heavy attack, increasing Flow, when full it deals extra damage on hit
    - E for interaction; this includes opening chests and interact with fountain
    - Esc to exit the game
    - Player starts with 50 health and 50 energy
    - Sprinting takes 1 energy/sec
    - Each dodge takes 2 energy
    - Each light attack takes 3 energy
    - Heavy attack charging takes 5 energy/sec
- There will be chests and fountains around the world. They are guarded by a few minions. Only by defeating them can the player receive the benefits in the chests/fountains. They can interact using E to get the item in the chest.
    - Chests may contain one of the following items:
        - Increasing max health by 100
        - Increasing max energy by 10
        - A random weapon
    - Fountains restore 50% of the player's max health
- The player cannot interact with world elements including the chests and the fountain while in combat with any enemy
- When the boss of a level is defeated, the player gets teleported to the starting point in the next level's map.

AI:

- Enemies have an area of detection; when player enters the area of detection of the enemy or if the player attacks the enemy, the enemy will enter attack mode
    - during attack mode, the enemy will actively path towards the player to get within range of its own attack; then it will use one of its attacks (for regular enemies, there will only be 1-2 types of attack; for bosses, there will be more than 3 types of attacks)
- Bosses may choose the appropriate attack based on their own health, their relative positioning to the player, their distance to the player, and time. They have both short range and long range attacks in their arsenal.

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.

Combo attacks

Performing light attack consecutively will switch up the way the player attacks (wind up time, attack time, damage may be different); performing the 6th combo attack will trigger a finisher move that deals significantly higher damage. Dodging will keep the current combo. Taking damage or not attacking for a while resets the combo attack. The more combo you can hit, the more FLOW you can get per hit.

Impact on Gameplay if Included:

- It will increase the playability and skill expression for the player. It will make light attacks less repetitive-looking.

Impact if Skipped:

- The light attacks will all look and feel the same. There wouldn't be variation in the damage and FLOW.

Proposed Alternative:

- We can randomize the damage and FLOW gained by light attacks, which will reduce the repetition of light attacks by a bit.

Random map generation

We can generate the entire map of all the levels randomly, ensuring the paths all lead to something (chest, fountain, enemies) and place the items/enemies randomly in appropriate places.

Impact on Gameplay if Included:

- Including random map generation will significantly enhance the replayability of the game. Players will have a fresh and unique experience each time they play, preventing them from memorizing layouts and encouraging adaptive strategies.

Impact if Skipped:

- Without random map generation, the game would rely on pre-designed, static levels that remain the same for every playthrough.

Proposed Alternative:

- If random map generation is skipped, an alternative would be to create multiple handcrafted level designs that rotate between playthroughs. This would offer some level of variety without the complexity of procedural generation. Additionally, a map modifier system could be added, which would randomly change certain aspects of each static map (such as enemy placement, obstacles) to introduce variability, though on a smaller scale compared to full random generation.

Devices:

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

We only support PC and Mac, with keyboard and mouse input devices. We do not provide controller or Linux support. For how inputs make to in-game controls, see Gameplay Logic section in the Technical Elements.

Tools:

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

GLFW - for interacting with user input (pressing of keys and mouse input)

https://github.com/eug/awesome-opengl?tab=readme-ov-file

SDL - low level interface to connect to hardware (for audio output when there is a collision in the game, attacks land on player or when player successfully dodges boss attakc)

imgui - for creating graphical user interfaces (gui showing user inventory of weapons)

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.

As mentioned in the TWA, we will be following agile methodology. Assign points to each task and make sure the team members get an even split of tasks. We will vote on the number of task points required for each task/story together.

1 task point - trivial change (few lines of code in one place)

2 task points - code changes in multiple places

3 task points - standard new feature

5 task points - major new feature with complex design

During weekly team meetings, we should look at all tasks needed to be done this week and split the total task points evenly among all team members.

During the middle of the week, team members should update everyone with their progress on the assigned tasks and seek help if they feel they can't finish the task by the end of the week.

We will track deadlines and tasks in GitHub.

While we have roles, we will also be doing pair programming to help each other if they struggle with implementation.

Roles for each person (and why they are assigned the roles):

1) Combat System, SFX & Music – **William**: With prior game development experience, William will take on the responsibility of designing and implementing the combat system which is central to the game. His music production experience also makes him suitable for creating sounds and music for the game.

2) Map, Chest & Fountain Interactions, Inventory System, SFX & Music – **Stephanie**: Stephanie's passion for fantasy games makes her the perfect candidate to manage the map design, interactions with chests and fountains, and the development of the inventory system. Stephanie can also contribute to the sounds and music of the game with her prior music experience.

3) Bosses and Minion AI – **Ansh**: Ansh will be responsible for the design and implementation of AI systems for both bosses and minions. His strong interest in AI design and pathfinding algorithms makes him eager to explore and develop these key gameplay components.

4) Weapon Implementation and Physics System – **Toby**: With a strong background in calculus, Toby will be responsible for the implementation of the physics system and weapon mechanics. His expertise will ensure accurate and realistic weapon behavior.

5) UI and Asset Creation – **Angela**: Angela's prior experienabice with design, coupled with her eye for aesthetics, will allow her to create an intuitive and visually appealing user interface and game assets, ensuring the game is both functional and engaging.

Development Plan:

Milestone 1: Basic Game Setup (Weeks 1-2)

Week 1 Goals:

- Map & Level Layout: Design the basic map for the Royal Kitchen using OpenGL and GLFW. The goal is to render a simple map layout where the player can navigate without passing through doors or obstacles.
- Player Movement & Input Handling: Implement WASD movement and dodging with an invincibility window. The player should be able to move in all directions and dodge using the spacebar, ensuring the movement and dodge work smoothly without passing through walls.
- Collision Detection (Player vs. Environment): Implement collision detection between the player and map boundaries using OpenGL and custom collision logic. The player should be prevented from moving through walls or obstacles, and the collision system should stop the player from moving when they hit a wall.

Week 2 Goals:

- Basic Combat System: Implement light and heavy attacks with basic damage calculations. The player should be able to perform both attack types, with damage and visual feedback shown on successful hits.
- AI & Pathfinding for Minions: Implement basic enemy movement and pathfinding. Minions should follow the player using grid-based pathfinding and stop when within range of their attacks.
- FLOW System: Set up the FLOW meter, which increases on perfect dodges and is consumed by heavy attacks. FLOW will be used to enhance heavy attacks and provide visual feedback.
- Collision Detection (Player vs. Minions): Detect collisions between the player's attacks and minions using custom collision logic.

Milestone 2: Playability and Boss Fight (Weeks 3-4)

Week 1 Goals:

- Boss AI (Grumpy Chef): Implement the Grumpy Chef's attack patterns with custom AI logic. The boss should use all attacks, respond to player movements, and leave counterattack openings. Each attack should have its respective cooldown.
- Health System: Implement health bars for both the player and enemies using ImGui and OpenGL. These health bars should appear on the HUD and update in real-time as damage is taken.
- Collision Detection (Weapons vs. Boss): Ensure player attacks damage the Grumpy Chef using custom collision logic. The boss should respond visually when hit, showing feedback such as flashing or health reduction.
- HUD Implementation: Implement the HUD, which displays player health, FLOW meter, and attack status using ImGui. The HUD should accurately reflect the player's health, FLOW, and abilities in real-time.

Week 2 Goals:

- Refine Combat System: Add the ability to charge heavy attacks, along with visual cues and audio feedback. The heavy attack should deal more damage based on how long the button is held, providing feedback when fully charged.
- Refine Boss AI (Grumpy Chef): Adjust AI patterns and scale difficulty based on health depletion, ensuring that the boss's attacks are properly timed and balanced.
- Collision Detection (Boss Attacks): Ensure the Grumpy Chef's attacks damage the player when in range. The boss's attacks should deal appropriate damage, and visual effects should be triggered upon impact.

Milestone 3: Level 2 Development (Weeks 5-6)

Week 1 Goals:

- Map & Level Layout (Castle Hallway): Design and build the Castle Hallway map using OpenGL and custom level design logic. The map should be fully rendered, with obstacles and paths that the player can navigate without glitches.

- AI & Pathfinding for Royal Guard: Implement AI for the Royal Guard using custom AI logic and pathfinding algorithms. The Guard should be able to attack, block with a shield, and perform a shield charge. It should respond dynamically to the player, switching between attacking and defending.
- Minion AI (Castle Sentries): Implement AI for the Castle Sentries. The sentries should engage the player, using their attacks and reacting to player movements.

Week 2 Goals:

- Combat System (Advanced Mechanics): Add advanced combat mechanics like parrying and perfect dodges. Players should be able to parry incoming attacks, and successful parries should negate damage while generating additional FLOW. Perfect dodges should also generate FLOW.
- AI & Pathfinding Refinement: Fine-tune the AI behaviors for the Royal Guard and minions, ensuring that they behave dynamically and intelligently, without getting stuck or ignoring the player.
- HUD Enhancements: Add visual and audio cues for boss attacks using ImGui. These cues should indicate when the Royal Guard or other bosses are about to perform special attacks, providing the player with warnings to react in time.

Milestone 4: Final Level Setup and Game Polishing (Weeks 7-8)

Week 1 Goals:

- Map & Level Layout (Royal Chambers): Build the Royal Chambers level, complete with traps and hidden rooms using OpenGL and custom level design logic. The map should be fully functional, with interactive traps that activate and hidden rooms that can be discovered.
- AI for the Prince: Implement AI for the Prince, focusing on rapid attacks, and counterattacks.

Week 2 Goals:

- Map & Level Layout (Throne Room): Finalize the design for the Throne Room, complete with environmental hazards using OpenGL. The map should include obstacles that pose a challenge during the final boss fight.
- AI for the King: Implement the AI for the final boss (King/Queen), incorporating magic-based attacks and scepter strikes. The King/Queen should switch their attacks, leaving brief openings for the player to strike.
- Polishing & Bug Fixing: Refine all systems and fix major bugs using debugging tools and testing frameworks. By the end of this milestone, no gameplay-breaking bugs should remain, and the game should be fully playable without critical issues.

Scope:

If unexpected events cause the project to be delayed. To ensure we can still deliver a complete and polished experience.

- Instead of trying to develop 4 levels, we plan to focus on the first two levels and with associated bosses for those levels
- Reduce the variation of Bosses' attacks, implement just a couple types of attack for bosses across all levels, instead of them having different kinds of attacks.

This reduction in scope will prevent us from overcommitting and allow us to adjust based on progress, providing a clear path forward even if development challenges arise.