Emu War

by Emu War

By

Ryan Cooper, Kyle James, Sam Beckmann, Jan Sonmezer, Silver Hao

Project submitted in partial fulfillment of the requirements for
IGME-601: Game Development Processes

Rochester Institute of Technology

School of Interactive Games & Media

B. Thomas Golisano College of Computing and Information Sciences

Fall 2022

Instructions

Your project documentation should be professionally written, proofread and spellchecked, and organized for consistency. It should look like a single document from a unified team, and everyone on the team is responsible for it in its entirety.

Throughout all of your documentation, we expect you to write in well-formed, complete sentences and provide your rationale for all decisions. In other words, ***don't just tell us WHAT you are doing, tell us WHY****. This document should tell the story of what you made and how you made it. It should be interesting to read. To this end, everyone on the team should read and help edit all sections!*

Any content in this document based on others' prior work (related research, assets, games, etc.) must be cited appropriately using in-text citations in APA style + a full references section at the end of this document. See the References section for more notes.

Timeline

It is VERY tempting to focus on making the game and write everything up later. "It'll be fine. I just have to write it up." is a common sentiment. Don't succumb to this! Yes, the writing isn't the fun part - but it's a required element of this course.

Instead of a final hurdle to complete, think of this document as an evolving view into the internals of your team's objective, progress, and lessons learned. Use it as a communication tool internally and externally. *If you work on it slowly, writing when things are fresh in your head, you'll find that, before you know it, it's done and fairly painless. (You'll also get MUCH better feedback from stakeholders if they can read in small chunks instead of all at once.)*

For more about letting your writing evolve and preventing perfect from being the enemy of good/done, see: https://wrd.as.uky.edu/sites/default/files/1-Shitty%20First%20Drafts.pdf

Acknowledgments

Thank people here. Your faculty, external team members, peers, etc. No, this isn't a published paper, graduate thesis, etc., but all work we do is supported by others. It's good to learn how to acknowledge them officially. This doesn't need to be long, sappy, etc., but it should also be more than a list of names.
https://www.discoverphds.com/advice/doing/acknowledgements-for-thesis-and-dissertations is in the context of Ph.D. acknowledgments but has nice examples of appropriate tone and scope.

Thank you to Corrine and Professor Erika Mesh for their guidance and support on the project and our development process. Special thanks to our game design and development masters program cohort for playtesting and giving feedback on our game.

Executive Summary

Summarize your entire project in one page, essentially giving an overview of the entire document and project. If someone reads only this page, they should understand what you sought to achieve, how you achieved it, what problems you addressed, how well it worked, and what remains to be solved. You'll likely write this at the end of the year.

This is NOT just an elevator pitch/game summary. Think of it as an extended abstract in a research paper. It summarizes everything in this entire document. (You'll likely write it near the end of the project.)

Emu War is a 2D action-strategy game where you play as General Emunuel, leader of the emu horde, with the goal of devastating farmers' wheat crops. During the game, you must collect emus to grow your flock, eat wheat crops, and use stealth, evasion tactics, and player abilities to avoid and evade the hunters. Players should feel skillful in their tactics to outsmart the enemies and eat the wheat crops with their flock. This document is the record of our process of developing this game.

The overarching goal of the project was to get a better understanding of game development processes. As such, we spent most of the time investigating and delving into what works well for our group and how to make adjustments to the process in an ongoing project. The project's secondary goal was to design and create a fun game about a non-domestic animal with zany and dark humor undertones that we could show off as portfolio pieces.

Process-wise, we got into a great groove early on, thanks to establishing a solid foundation of roles, and found a process that worked well for each team member. We established an MVP that we thought would work for our set time limit of 3-4 hours per week and adjusted once we reached that MVP earlier than we expected. Thus, we found that our process was effective in helping us set and reach obtainable goals promptly.

We had minimal problems with development, and although some bugs were present in some Playtests, we worked through any development issue with relative ease. We mitigated the problems we addressed throughout the project with improvements to our process.

Due to the limited time that we had to explicitly work on the project's development, per project specifications, there is still plenty of room to polish and expand upon Emu War's experience. As is, Emu War is a minimal viable interaction prototype. Although we have designed many systems for the game, not everything is available in-game yet, but at the end of the experience, we have showcased the proof of concept that hints toward what a fully-fledged Emu War looks like.

Table of Contents.

This should update correctly whenever refreshed if you use the formats already defined consistently. (It'll all be in blue because these are links to sections. You can format to remove the color in your final version. No sense in doing it now because new sections will be blue every time you refresh.)

1 Introduction

This is where you introduce the game concept and give an overview of the project You can organize this section in a variety of ways depending on what works for your team. Some suggested subsections are below.

- *Some will write a single section to cover in a cohesive narrative.*
- *Other teams have chosen to break the introduction into key subsections. (Especially for teams tackling specific technical objectives as well. E.g. Game Context & Intent, Learning Objectives, Resulting Approach.)*

This is NOT your entire game design, but there should be enough here to provide context for Production, etc. even without having all of the game design details. Potentially interested stakeholders should also be able to read ONLY this section in order to decide if they want to read more.

Emu War is a 2D action-strategy game where you play as General Emunuel, leader of the emu horde, with the goal of devastating farmers' wheat crops. During the game, you must collect emus to grow your flock, eat wheat crops, and use stealth, evasion tactics, and player abilities to avoid and evade the hunters. Players should feel skillful in their tactics to outsmart the enemies and eat the wheat crops with their flock.

1.1 Synopsis

Introduce the game itself and your goals. This is effectively your elevator pitch. What is the intended overall player experience? Ensure that you give a tagline, summarize the gameplay game mechanics, and introduce what will make this game unique.

https://davidmullich.com/2018/06/25/an-actionable-game-design-document-template/ suggests describing an imaginary play session. "Create and name an imaginary player and then put yourself in their shoes playing the game for five minutes, beginning with him or her starting the game for the first time. Focus on the player's experience: what they're seeing, what they're hearing, what they're doing, and what they're thinking about. Any thoughts or feelings they have should be reflections of what is going on in the game.

- *High Concept: A one-sentence summary of the game's premise.*
- *Game Genre: First-person shooter, platformer, role-playing, real-time strategy game, etc.*
- *Setting: Medieval fantasy world, modern-day city, etc.*
- *Target Player: Interests, age, gender, casual vs. hardcore, etc.*
- *Play Value: What will make the game fun to play? Be specific and think in terms of the player experience rather than game features. Is it reality-based or fantasy-based? Is it predictable or full of surprises? Is it immersive? Is it a mindless pastime or challenging? Are there few or many rules? Do you build or destroy things? Do you cooperate with other players or is it a competition? Is it funny, scary, suspenseful, tranquil, cheerful, or gloomy? Is there a risk of losing progress, lives, or the entire game?*
- *Competition: What game(s) have a similar setting, genre, and/or mechanics?*
- *What's Unique: What makes your game different from similar games?*
- *Game Engine: Game Maker, Unity, Unreal?*

- **High Concept:** Play as General Emunuel and lead your flock of emus to devastate the wheat crops of Australian farmers to keep your flock fed and healthy, but take special care to avoid the Australian military.
- **Game Genre:** 2D, Action-strategy, Light stealth elements
- **Setting:** 1932 rural Australia
- **Target Player:** The game targets teens and older, especially those who enjoy the dark humor in programming such as Adult Swim as Emu War focuses on a zany but serious subject.
- **Play Value:** The game derives its fun from utilizing your strategies with the horde and abilities to avoid and evade the hunters while trying to keep your horde alive by decimating the wheat crops. The game is realistic, apart from the appearance

of General Emunuel and the hunters, focusing on the agility capabilities of emus and the real-life Emu War. There are minimal rules, only that you must survive and destroy the crops; the faster you do this, and the more emus you keep alive, the better the score will be. There is a risk of losing as the hunters may kill the player and the horde. The key play value is the zaniness of the situation mixed with dark humor.

- **Competition:**
  - *Emu War!* is highly fictional and focuses on warring emus and humans in a 3D open-world sandbox environment.
  - *The Great Emu War* is similar to our game but is a 3D third-person couch-competitive game, pitting players as hunters and emus against each other, where emus must survive, and hunters must hunt them down.
  - *The Great Emu War of 1932* is a 3D top-down shooter played from the perspective of a human trying to survive an onslaught of emus.
- **What's Unique:** What's unique about Emu War, in comparison to its competition, is that it's a 2D single-player game focused on the perspective of emus and utilizes minor stealth and evasion elements that the other competition does not. The key mechanics that set us apart from the competition is horde-throwing and the player's abilities.
- **Game Engine:** Unity, as many of us are comfortable with Unity, we want to focus more on the production process.

1.2 Expected Outcomes

What are the team's goals? This is where you summarize your MVP and also introduce any major team learning objectives or other goals/constraints that influence your MVP decision.

Our team would like to make an action-strategy game based on the engine of our choice, Unity. The three developers on the team agreed to Unity because we all understood how to use it properly. This engine allowed us to explore new concepts like raycasting and state machines. Additionally, Unreal is not designed to handle 2D games well, and not everyone on the team was familiar with GameMaker.

When it came to a game concerning the war topic, we came up with many typical genres, including battle chess, RTS, and tower defense. However, in our mind, the game should be fast-paced to present the high-speed moving of emus, as their frantic high-speed movement led the emus to "victory" against the Australian Army. So, we decided on the action-strategy genre with light stealth elements to stay closer to real-world events.

1.3 Background

Introduce the genre and your team's interest in it (i.e. why YOU want to make this game for this project). What makes that genre engaging? What are its strengths and weaknesses? Introduce any other key research that went into making the game.

The big catch of our game was the concept of the emu war at large. The concept of a military "losing" to a flock of birds is hilarious, even if it is not the whole picture. Due to

the more mature audience we intended to play our game, we did not need to severely alter the Emu War's story. Instead, it allowed us to lean into how ridiculous it is despite the fact that this was a “war” mandated by the Australian government.

Keeping with animal vs. man themes, we originally wanted to incorporate stealth elements into our game. However, after playtesting, we found that players had more fun trying to evade hunters quickly instead of planning their attacks. After all, these are giant birds; stealth is not their specialty.

The action scope of our gameplay allows for the quick gathering of a horde and quick movement speed similar to the emu’s rapid run speed. It's exhilarating to run around with your horde frantically around the Australian wastelands. At the same time, players should grow attached to their horde and take extra precautions to ensure that their flock stays alive.

1.4 Approach

Give a summary of the rest of the project and how different elements (production processes, game design details, aesthetics, technical design, etc.) all come together to support your goals.

The production processes the team created have provided great convenience to our cooperation and promoted the team’s overall working efficiency. Throughout the project, we applied Scrum to the best of our knowledge and used Jira as the task management tool. We create new stories during sprint planning, bring some of the backlogs to the board, and allocate different points to the stories based on their priorities. We follow the workflow shown below to move the stories among different columns to update the progress of each story and determine accomplishment by acceptance criteria.

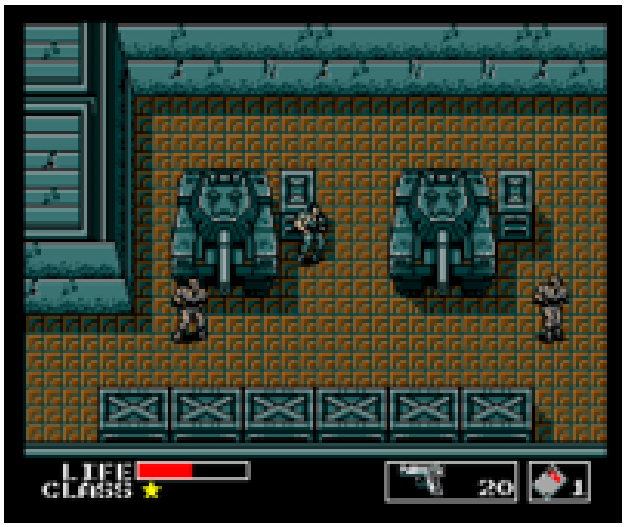
We use Discord as the basic contact tool to maintain good communication and GitHub as the version control tool to guarantee parallel development and coordinate works by our respective roles, whose details will be explained in production.

Based on the real history of the emu war, our game is thought to be fast-paced in the action-strategy genre, where gamers play the emu to steal food from farmers and survive the Australian military. For more thoughtful gameplay, we have decided on our player experience not just to rely on fast-paced action but the utilization of stealth as

well through avoiding undefeatable enemies on the level to reach the necessary collectibles. In addition, we've been learning technical design from many games like Right Click to Necromance, which helped to determine the original control mode of the emu flock. After changes to the final iteration of our movement, our goal has become to use the mouse exclusively for flocking to prevent the contradictions the early iterations of the mouse-controlled movement had with flocking.

(Right Click to Necromance, 2015)

Jan has taken the sole objective of creating art for the project. While he's not a dedicated artist, his specialty in 2D pixel art fits the visual aesthetics we have planned to accomplish for our project. Regarding the game's art style, we have been inspired by callbacks to other action-stealth games such as Metal Gear. Metal Gear offers a top-down point of view with a well-defined pixel grid to make the characters stand out from the static meshes, making it visually easier for the player to differentiate the elements within the level. We have followed this template to design our player character and level to provide a compact, traversable area, further reinforced by its visual art elements.



(Metal Gear, 1987)

The team is taking an overall well-woven style regarding our technical design. We chose this as we wanted all aspects of our game to feel meaningful and impactful. As a result, we wanted to ensure that all of our core subsystems interacted with each other subsystem on their own while also interacting when they are all together. This design allows us to create an overall more engaging and challenging experience for our players to enjoy.

2 Production

This chapter drills down on your current process. As it changes, update this!!.

Note: your analysis of how well the production process worked and changes you made along the way goes into the post-mortem later in this document.

2.1 Overview

Give a brief overview of the constraints you considered, and the approach you took when making production decisions for the project. This is NOT all of the production details in one place. Instead, think of it as defining the context within which your team will be/had to work, how that influenced you, and give some key highlights about what the rest of this chapter will describe in detail.

Given limited time (at least four hours per week for about three months) and manpower (five team members), as well as various risks of illness, merge conflicts, and so forth, the team is expected to make the best use of production processes learned from this course to overcome potential challenges and get project progress in control. As a team formed with the same interest in the game idea of Emu War, we hopefully have every team member contribute to the project with their expertise and set up good communication to help each other use their skill set to the best capacity without burning out anyone. Based on the analysis of resources and risk, we identified the planning and scope, including overall priorities, MVP, and stretch goals of the project. Moreover, we put forward the game development processes, including task management, version control, and asset pipelines used for production. We continuously summarize each release to see how we meet our planning and scope in each milestone and sprint.

2.2 Team Organization

2.2.1 Team Formation

How did you structure your team? What skills did you have available as a combined group and how did that influence your objectives and process? .... Core team and external team members - how did you form and then structure the team? Why?

Above all, our love of games and interest in the emu war gathered us together in the team formation class and became the cohesion of our team. Then we formed the group with our respective experiences in art, design, and programming that enabled us to make a 2D game of emu war. Based on individual specialties, different roles are assigned to each of us to build the development process of our project.

2.2.2 Core Team

Discuss the general/shared responsibilities and expectations for the core team as a whole and how you managed communications within the core team. Why did you choose this strategy and how did it evolve over time?

Every team member is expected to contribute to the team project with their knowledge and skills, learn from each other and grow up in the collaboration of the team project. To live up to the expectations, each team member needs to respect one another, be

prepared to engage in discussion and teamwork actively, and always keep in mind to call on each other whenever help is needed.

For effective team communication (immediate and beyond immediate), we have daily stand-ups with one weekly asynchronous meeting on Wednesday and one weekly synchronous meeting on Sunday. The Wednesday meeting deals with any concerns or formal code reviews, while the Sunday meeting is a general planning meeting to deal with the next steps in the milestone or any post-mortem work that needs to be done. We used Discord and Jira to deliver messages and share information with the team. With the development of our project, these communication tools will help improve involvement and efficiency of discussion and probably build up team knowledge management.

Then describe each team member's specific role(s).

Kyle James
Team Liaison, Programming

For each, why was this role important? How did this contribute to your overall production goals?

The role of the Team Liaison is vital to maintaining effective external and internal communication to ensure that everybody is on the same page, improving insight into production goals.

The Programming role is an overarching responsibility, assisting and programming various functions within the game to ensure we reach our goals for the game.

Samuel Beckmann
GitHub Master, Programming

For each, why was this role important? How did this contribute to your overall production goals?

The role of GitHub Master is important as it is a beacon of support for the team to rely on in issues relating to GitHub. This will contribute to our overall production goals as if we are unable to resolve said issues, it will severely hanker our overall production goals and progress.

Ryan Cooper
Programming, Code Architect

For each, why was this role important? How did this contribute to your overall production goals?

Created the structure for coding standards as well as disciplines of coding shared among the team. Good coding structure is important as it creates a unified system everyone can understand in our scripts. This entails easy understanding and quicker debugging as well as better uniformity. As a programmer, it's my job to use those standards and create the scripts needed for the game to run.

Jan Sonmezer
Designer, Art, Narrative

As designer, the main goal is to create and maintain the game's core goals with its inspiration through handling the game's aesthetics and story to coincide with its mechanics. Games are, at its core, a form of interactive entertainment, so it is integral to ensure that the project does not lose its identity while successfully constructing the bonds between art and code to produce a fun experience for the target audience.

Silver Hao
Level Design, Quality Assurance

For each, why was this role important? How did this contribute to your overall production goals?

Game design and development start with research. Research is such an important part of game development that it determines the success of ideas concerning gameplay, mechanism, aesthetic and all the other aspects of the game. As the researcher of our team, I'm to find evidence that affirms or negates the feasibility of each decision we make to direct everything on track. Game design and development end with testing. Likewise, testing is so important for game development that it assures the quality of the game. As the QA of our team, I'll perform acceptance testing with giving developers feedback on the tests to avoid anything off track.

2.3 Resource & Risk Analysis

What skills and roles will you need? Be open to additional team members, keeping in mind that you probably will have more work than you're expecting!

What technology will be required throughout the project, in terms of hardware, software, materials, etc? You might also want to include monetary costs if you will be working in the analog world and expect to do any actual production and purchasing.

Also DISCUSS the major risks your team will have/had to account for during the project. For each, cover:

- *Likelihood – What were the chances of this happening?*
- *Severity – What would the impact have been?*
- *Mitigation – How would/did you respond/minimize the impact of this?*

This is not simply a table with ratings. Discuss each and how it influenced your decisions and the project overall.

Risk	Likelihood	Severity	Mitigations
Burn Out	Medium	High	Keep morale high. Acknowledge that people may have issues keeping them from working.

Merge Conflicts	Medium	Medium	Make sure the team works on their branches for each task. Work on separate scenes for each task.
WebGL Build conflicts with our build	Medium	Medium/ Catastrophic	Test the WebGL build before the initial build date. Make any adjustments as needed.
Resource Production is slow	Low	Medium	Prioritize necessary assets. Divert more production to asset production if needed. Use placeholder assets when necessary.
Game-breaking bugs	Low	Catastrophic	Keep all noticeable bugs on your respective branch. Don't push anything if there are game-breaking bugs.
COVID-19, Other Illnesses, or other absences	Low	High	Take necessary measures to ensure our health. Inform the team if we do get sick and don't do anything to spread contagions to the team. However, with good communication and knowledge sharing, our team should face minimal impact from an unforeseen illness.
Playtest Results are negative against the game	Medium	High	Take criticism into mind while never straying away from criticism. Adjust gameplay as needed.
A member cannot attend a meeting	Medium	Low	Communicate anything necessary to the team. Keep the team informed if they are unable to attend, if possible.

Risk Handling Situations:

WebGL Build worked against us:

During Playtest 1: We did not plan for our build to work as differently as it did from WebGL to Unity. As such, the results of our playtest were slightly inaccurate since the movement of the Emu was not the same in the WebGL build as in the Unity Editor. We had to handle criticism of the movement with a grain of salt. In the future. We planned to have the build made well before the due date to avoid these conflicts.

Merge Conflicts:

Before our second playtest, we met the night before to compile our changes and make the build. However, little mistakes like two members of the team editing the same level

or the same prefab on different branches can result in headaches. We found this out the hard way. Compounded with the knowledge that the build had to be ready by tomorrow, the team struggled to get everything together. After we resolved the conflicts, we then had to make sure our changes worked with each other smoothly without any noticeable bugs. This process took significantly longer than we had hoped.

Sprint 6 Absences:

Although we reduced our workload in the final sprint, many unforeseen events challenged the production team. Due to a family emergency, Ryan had to take time off, and Jan had difficulties with foreign customs. This issue never hurt production despite losing crucial team members at the project's final leg. Communication was always keen, and we reported conflicts as soon as they arose. Because of this communication, we could finish strong despite so many setbacks.

2.4 Planning & Scope

2.4.1 Overall Priorities

Describe the overall priorities and rationale that guided your development priorities (this should add depth and context to the goals you introduced in Expected Outcomes). For example, for YOUR game, what is more critical: prototyping a variety of NPCs or doing 1, really well? Many levels or 1 representative level with a variety of mechanics? ...

At the beginning of our first meeting, we came up with an initial list of objectives we wanted to accomplish during the semester. In our discussion, we devised five objectives and arranged them by priority. The first was the ability to move the emu around. The player controller was the first task we created and was our top priority. Close behind was the terrain creation. This task was second in the priority queue but was planned to be done in tandem with the player movement. The third was arranging the hunter's AI and how they move around and interact with the player. The fourth objective we came up with was to implement an emu flocking system similar to the split function in Agar.io. Lastly, our final objective consisted of implementing our wheat, serving as the main collectible and goal for our player within the level. Ultimately, we achieved these goals much faster than we intended by sprint four. Underscoring a project like this was intentional due to our unique constraint of four hours a week. After we achieved our initial goals, we had the challenge of adding to what we already had. Adding in a scoring system and unique UI elements. New abilities for Emunel to use to offer players another mechanic to utilize within the level to achieve the goal of the game, aimed to further add to the fun factor.

We had set the process order of our priorities based on the mechanics and systems we wanted to share with our playtesters. The player character was naturally paramount for our first playable build, so we did its development quickly. Terrain followed later on thanks to the efficiency in our art asset creation to mitigate the use of placeholders as much as possible. Now that we had a working player character and a level for it to

transverse on, then came the implementation of our project's main challenge: The Hunter. After that was set, we devoted most of our resources to the last two goals of the flocking mechanic and the wheat collecting system to represent the player goals within the level. These five aspects were our key priorities because they were the essentials we decided on for our core game loop.

Avoid the hunters, lead your flock, collect the wheat, and repeat. That has been the designated game experience of our project since day one. Once we had implemented all those priorities, thanks to our well-organized process, we used our additional resources and time to design features that were not part of the core game loop but would still enhance it. That is when our priority shifted to scoring and UI elements to visually reinforce the game experience, ensuring that the interface reflected the information we wanted our player to be able to track. Finally, we added the player abilities to provide another method for player interactivity and agency in a way that supported the loop without deriving far from the main objective.

2.4.2 Minimum Viable Product

Given the priorities, what is/was your target MVP? This section WILL change over time as you refine the game design. That's fine. This is a living document!

For the minimum viable product, we intended to have the player control General Emunel to steal crops from the hunters' watch. Emunel can pick up spare emus to add to his flock as he progresses. If a hunter catches an emu within their line of sight, they will pursue the flock attempting to hunt them down. The win condition is based on eating all of the wheat in each level. Our base movement and perspective were completed by the end of milestone one. However, we decided to strictly move the player controls to the keyboard instead of the mouse. The decision to move away from the mouse-based movement control had a lot to do with feedback we received on playtests that stated that it was too difficult to use the mouse to throw the horde and move the player simultaneously. We also redesigned our level throughout the semester until we found a format we liked in a linear layout to an open area. While we initially planned for three levels, we cut back to one well-polished level that was bigger in size. The hunter was polished over the semester, but the base functionality was ready by our playtest in milestone 2. From there, we furthered polishing through raycast vision and better awareness. The horde mechanic was the last of our main objectives. We planned to get it implemented before the end of milestone three. Still, we completed the implementation at the end of milestone two. Implementing the Horde mechanic allowed us to add a level of tactical choices and agency for the player in which they want to traverse our game. This mechanic allows the player to decide if they want to amass a large number of emus in their horde, allowing them to walk around with a large "meat shield" of the horde around them, mostly preventing them from taking damage. While also allowing the player to decide if they want, they can throw a member of their Horde away as a distraction to potentially save more lives of the Horde itself.

2.4.3 Stretch Goals

List any features that you wanted to accomplish, but weren't deemed essential. (Effectively future work section). In the postmortem, close the loop by discussing more about how your concept evolved and priorities changed (& why!)

We intentionally under-scoped our plans to ensure that our MVP was everything we sought to be. That being said, we had plenty of stretch goals down the line. One idea was a more complex way for the hunters to spot the Emus with a hearing system. If the emu made a certain level of noise, the hunter would notice the emu in the same way as walking into their line of sight and begin to pursue it. The player could use tall grass to hide from the hunters instead of staying safe behind a wall. We could have implemented other environments, such as water or swamps, to add variety to our terrain. To complement this, we wanted to add a nav mesh to prevent the hunters from traversing these environments. Navmesh would have also allowed us to enhance the hunters' patrol cycles further. Finally, we would have liked to give the players more abilities with the option to pick and choose what they wanted to bring to the level.

2.5 Game Development Process

2.5.1 Overview

Describe your overall development process from both strategic (the overall model: overall workflow, etc.) and tactical (how you enacted your process strategy: task management procedures, etc.) perspectives. Be specific and include your rationale for these approaches and their impact on the final product quality.

Our overall development process involves using task management systems such as Jira to manage the card system, where we can assign tasks and manage the project's overall backlog throughout the semester. We also used GitHub to manage our version control and the project's main code base.

Our process follows a two-week sprint, two-sprint milestone time dynamic. Before each sprint starts, the team establishes the most important and pressing tasks that need to be covered in the upcoming sprint. After discussing this, the team assigns the card associated with each task to the corresponding member of the team. Throughout the sprint, once a task is completed, the completing member of the team will request a review of their assignment, which will allow the team to verify the quality of the completed task. If the task is deemed finished, that team member will push their changes and start the next most important task they can complete. If not deemed finished, the member will continue working on it with the feedback in mind, then requests another review when finished. Towards the final days of the sprint, the team members will meet to have a retrospective meeting and a review of the sprint as a whole. This process allows the team to go over what they were able to accomplish in the sprint and what they learned from the sprint, which will allow them to take these aspects into account for the next sprint.

2.5.2 Task Management

Given the priorities and target MVP + your team organization and communication strategies, how are you tracking what needs to be done? How do you know who is doing what, when? How are dependencies managed and progress assessed? Describe this from both strategic (the overall model: overall workflow, etc.) and tactical (how you enacted your process strategy: task management procedures, etc.) perspectives. Be specific and include your rationale for these approaches and their impact on the final product quality and how your MVP and risks motivated your process decisions. (Use an appendix for detailed task management procedures, etc. as needed to keep this section from getting cluttered.)

For our task management, we are using Jira as a tool to manage our overall task flow. We are using Jira to make backlog items and task cards to hold each task and their status (to be done, done, under review, backlog).

As much of the project's work is done outside of meeting times, the team is utilizing a Jira board, as it will allow the team to manage and monitor the tasks each team member is completing. Each task will have its own story (job or user) and acceptance criteria. Including the story will allow the team to ensure that the intended effect is met by completing the task. Including the acceptance criteria allows the team to give the person completing the task a baseline of what is expected from this task, which the team will compare to when the task is completed. The task board is repeatedly updated and reviewed during every sprint meeting and weekly meeting to ensure that the tasks are completed at the necessary rate to stay on task for the sprint goals. Using a tool like Jira, the team can monitor the progress of the tasks and what will be needed in the upcoming sprints asynchronously, which will aid the team more.

2.5.3 Version Control

How did you manage the source code and other artifacts for the project? Be specific! Pull requests, continuous integration, ...

We are managing the source code for this project through GitHub. We will manage the git repository by having different feature branches, specifically on AI, Dev, and main, to name a few. We will also make more branches as needed, at which point we will update this section. We will manage art assets by keeping a copy of each asset in Google Drive while also being kept in the GitHub repository. We are also instantiating a rule for source control that every commit must accurately describe what was done in that push.

We will merge our branches only once a code review has been completed and confirmed that the merge should not break anything in the branch being merged to. We will most commonly do this, especially with merging to main before a build; however, merging to dev will happen much more often when a new feature is completed.

We ended up utilizing feature branches effectively to mitigate overlap and merge conflicts. This process was one aspect of version control that, as a team, we did very well, as we communicated effectively on top of using our feature branches to make sure to get our work done. This system helped us work effectively and efficiently as a team without worrying about losing work done by careless version control usage.

2.5.4 Asset Pipeline

Summarize how you store, manage, and incorporate assets into your game. Include any strategies for coordinating with external team members here as well. Add details into the Assets chapter later in this document!

We store our assets in multiple locations; however, after an incident where we lost access to the raw versions of our files, we are now moving to store all assets (raw and finished) in Google Drive. To ensure everyone has equal access to the assets in the project, we are storing them in our Google Drive. On top of this, we are storing our actual assets in the game in their respective locations of animation, materials, etc. Each folder has a sub-folder if necessary to ensure that the main folders are not overcrowded (i.e., in animations, there are Hunter, Horde, and Player with all their respective animations in it). We are incorporating the assets into the game and storing them in our GitHub repository folders. Before assets are added to the project in Unity and the GitHub repository, the asset goes under review and is approved by the team.

2.6 Release Summary

This is NOT a detailed project plan! That belongs in your task management system. This is a VERBAL summary of your major milestones/releases -- the ones that are completed and the ones that you have planned.

2.6.1 Release 1/Movement & Terrain - 9/22/2022

For each, copy/paste this subsection and describe each major build/milestone, when it occurred, what your priorities were for it, and the game elements completed for it. -- Not week by week - significant points in development: major pitches, pivots that you made, end of the semester, major playtests/submissions, ...

This is not just a list of what was done, but what did you learn with that release and how did it influence the rest of your milestones/progress? If there was a playtest associated with this release, describe it in Playtesting.

We prepared this build intending to get feedback from playtesters on our initial movement and camera mechanics and the aesthetics of our main character. This build was our first, and we completed the initial animation of our main character, the movement, the terrain, and the camera moving in the landscape. The build itself did not generate any significant pivots; however, the playtest associated with the build garnered great feedback from our playtesters that influenced our future direction with the project. See the analysis from the associated playtest in the playtest section.

2.6.2 Release 2/ Level and Hunter AI - 10/13/2022

We created this build in time for the second playtest. As our art goes, we improved our animation for Emunnel for his walk to look more natural and less sickly from advice from Professor Mesh. We also added the grid tiles for our floors and walls to begin building a level. We created a large level to highlight Emunuel's movement. The level was created using the tile assets and featured the ability to eat wheat and the first draft of the hunter AI. The wheat crops are the win condition scattered across the level for the player to eat them all. As the player stands on the crop, its health bar will go down. We implemented the first edition of the Hunter with a basic state machine. They will move from patrolling to shooting based on whether or not it sees the emu through their

vision cone. In a similar fashion to build one, the playtesters provided feedback on the current state of the build. Many of their suggestions were planned to be carried over to the next sprint. See the analysis from the associated playtest in the playtest section.

2.6.3 Release 3/ Abilities, Scoring, and Hoard - 11/10/2022

We created this build for the third playtest. Since our last playtest, we have worked on the hunters' directional movement and shooting animations. Though we have not been able to implement the animation, the creation of the assets is now complete. This process has taught us that implementing some ideas can be more difficult than creating them. Within the playtest build, however, we managed to successfully introduce our wheat model, as well as its collecting animation. We have improved our level design through playtester feedback, teaching us that a small space meaningfully laid out is more effective than a larger, broader concept.

The scoring system is also integrated, which rewards the player with a number based on how much wheat they have collected, how many emus they have in their hoard, how much health they have remaining, and how much time they have spent completing the level. The horde system aims to provide players with a resource to collect aside from wheat, but also one they can spend. Players can shoot emus from their hoard to distract Hunters, making the AI focus fire them instead. We have also implemented player abilities to accommodate the need for player agency from our feedback. There are two abilities with unlimited uses, limited only by cooldowns. The first ability temporarily increases General Emunuel's movement speed. The second ability shows the direction of the nearest collectible Wheat. The players can activate these abilities by pressing their designated buttons, allowing them to use the abilities when they choose to do so. See the analysis from the associated playtest in the playtest section.

2.6.4 Release 4/ Animations, Raycasting, Movement - 12/1/2022

For each, copy/paste this subsection and describe each major build/milestone, when it occurred, what your priorities were for it, and the game elements completed for it. -- Not week by week - significant points in development: major pitches, pivots that you made, end of the semester, major playtests/submissions, ...

This is not just a list of what was done, but what did you learn with that release and how did it influence the rest of your milestones/progress? If there was a playtest associated with this release, describe it in Playtesting.

For the fourth release, we started to shift our focus of development to polishing our gameplay and addressing feedback from Playtest 3. With the polish, we cleaned up our UI, added a couple of new menus to help the players understand the game better, and some new animations, along with raycasting for our hunter enemies. Regarding addressing feedback from Playtest 3, we shifted our movement from mouse-based to keyboard-based movement due to the horde-throwing mechanic's necessity to use the mouse. We also constructed our final build of the level for the game with feedback from that playtest. This build ultimately tied together our MVP (MVI) for the final milestone of the semester. See the analysis from the associated playtest in the playtest section.

3 Market and Competitive Analysis

This is the research you did to determine HOW you'll achieve the objectives you laid out in prior sections (i.e. your overall concept as described in the intro + MVP priorities led to the research here....)

For each area, you should do the following:

- *Compare the common (and uncommon) games in this "space." Ideally, you should have a table that notes gameplay and mechanics.*
- *Now that you have noted every canonical game in that "space," discuss how your game is innovative in the genre*

Add subsections as needed to break this up in a relevant way to support your game!

3.1 Overview

Give a summary of this chapter (i.e. enough for a reader to know what subsections are coming and make a decision about which they want to read for more details)

The market and competitive analysis introduce who our audience is and how we target the audience and what the genre of this game should be and how we determine on the genre. Based on the genre and audience, we researched the market and got inspired by other games to perfect our game ideas.

3.2 Audience

Why did you target the audience you did? How does the game design meet their needs? -- From the perspective of attributes of the audience and your game. Define the audience itself and how to address their interests/needs using relevant references.

For our audience, we want players who are in their late teens and above. War is a serious topic that can get gritty, but our retelling of Emu War necessarily isn't. As gun violence and animal abuse are undeniably topics regarding the politics of our game, our take on it is much more like the cartoon shows of Adult Swim, where sitcoms with heavy topics are lightened through colorful and funny cartoon animations. So we want our players to be mature enough to acknowledge war but humorous enough to have fun without undermining it.

As for gameplay and feel, we want our audience to be interested in pixel art action adventures, lite stealth projects, Agar.io, and other humorous avian projects such as the Untitled Goose Game. As a web game, we target a broader, casual audience of players over a dedicated and experienced fanbase.

3.3 Genre

Similar to the audience - how did you choose this genre, what evidence/reference supports this decision, etc.?

Throughout the development process, the genre we intended for our game changed over time. Originally we wanted a stealth game with strategic elements closer

to the original Metal Gear. However, we eventually changed our spec to include action and minor stealth elements. Eventually, after seeing players prefer gung-ho tactics over stealth in our playtests, we realized we should forgo stealth as one of our main selling points. So we decided to finalize our genre as action strategy instead of pure strategy or stealth-based gameplay.

3.4 Market Influences

Now that you know who you want to target and what they want, what defines this space in the market - ESA reports, other games, ... How is our game like this and how is it unique.

The nature of the Emu War that took place in Australia is a ridiculous event closer to a Monty Python sketch than a real-world event. As such, we choose not to take the art styles or theme of the game too seriously. One game we look towards is Untitled Goose Game, where a single goose must do everything in his power to annoy the town. We wish to carry these ridiculous themes while balancing the mature themes of combat and death. We imagine this game akin to a show on Adult Swim. Aiming to be gory, brutal, and occasionally immature while tackling some darker themes.

3.5 Conclusions

How will your game leverage this information and how will it be unique?

In terms of market research, we find that games concerning wars are too heavy, and games about emu war are thought of funny topics. We'd like to find a balance to close the gap between heaviness and absurdity. In our mind, emus are just seeking survival without the intention to aggress human beings. So, our game portrays emus as a united horde faced with a powerful human army. Furthermore, we render the whole story legendary by picking one of the emus, Emunuel, to lead the horde, which is a classic narrative technique used in many movies like The Lion King. We expect this game to deliver a sense of responsibility, ethnic unity, and pursuit of life to the player.

4 Game Design

You may want to repeat the way in which your game is innovative here. The final game design document (GDD) is a written explanation of how the proposed game actually works. The GDD is "living" specification of your game—another team could make the same game using your GDD. This should be the complete design document. At the beginning of this section, you should give a short summary of the entire design for those that don't want to read the gory details of the game design.

Add concept mockups and eventually game screenshots throughout this section.

This is the big picture – not just what you implemented. Include WHY you made these decisions, not just a report on what the game is. This is not a user manual! Technical and functional will not grab Weez's heartstrings. What's the hook?!

Explain your game to potential players, investors, etc. – NOT to developers. Yes, describe your mechanics, but do it in the context of the game concept, theme, etc. All of this should be specific to YOUR game. For example, for a 2d platformer, this section shouldn't be applicable to any platform, just YOURS.

Details about how mechanics are implemented, etc. go in technical design. This is about objectives of mechanics, level design, etc.

THIS is where you provide the details about how your game will achieve the goals laid out in the introduction.

4.1 Game Overview

Early in the project, you should define the core concept that will drive your design activities for the rest of the semester. Make sure to cover what will make this game fun!

What does the player control (an avatar, puzzle blocks, ...), and what is their goal and motivation? What is the "win" condition/level? What NPCs are present in the game? How does the player interact with them, and what is the result of these interactions?

What is the context for the game? Main story/characters? Setting?

*Much of this should have already been covered lightly in the Introduction (so that the reader had context when reading the Production and Analysis chapters). Here is where you expand on that and paint a **full** picture of what it will like to play your game. The rest of this chapter then details how the design will help you achieve those goals.*

The player controls an emu who can lead a horde of emus to explore a large open level where hunters patrol to kill them. The player is expected to find other emus and eat wheat crops while keeping as many of the flock alive. The more emus and wheat crops the player collects in the shortest possible time, the higher scores the player will achieve. To walk through the game, the player must collect all the wheat in the level while keeping at least one emu alive.

The NPCs in the game are the Australian military, which the player must try to avoid; otherwise, the player will lose emus in the horde or the leader's health by being shot by the Australians. When the leader of the emus loses all the health, the game is over.

The context of the game takes place in 1932 rural Australia, inspired by the real-life Emu War (Banerji, 2016). This game tells the story of General Emunuel, who was pushed out of his fields of crops and gathered his horde to reclaim the glorious crops of wheat that were easier to steal before.

4.2 Related Work

Discuss all the background work you did for the design and aesthetics. Appropriate papers and games should be cited here.

Examples of background for this section would be the use of flow theory for the game flow as well as motivational research in order to better motivate individuals to play a game.

This section should be about the research you did as a team to support specific elements of the game: how specific mechanics work, usability issues, etc. I.e. you didn't pull your game design or implementation out of thin air - you based it on prior work (other games, other research, etc.).

How did you dig deep to solve core problems in your game design and development? (balance, mechanics, rhythm motivations, ...)

Summarize all of that background research here AND cite as needed in the detailed design discussions throughout this chapter.

Our related work research comprised three key factors: the Emu War's history, Australia's landscape, and Emu agility. For the history of the Emu War, we have resorted to reading from several sources to get a better grasp of what happened during this humorously odd piece of history and the details within it. These articles have provided us with information on not only the equipment of the military sent to deal with the emus but observations on the emus as well. One thing that stood out to us was the mention of emu packs having leaders to deal with the military threat, which led to the creation of General Emunuel.

After our protagonist was set, we worked on getting it right. We have watched several videos on Youtube to analyze and replicate the movement of emus over the medium of our game. We initially struggled with the movement of the emus within our game due to their abnormal anatomy. Still, the more data we have collected through video documentation of these unique avians, the more we established a smooth movement for our player character and flock. Once we felt confident with the look and feel of our emu through the feedback of our playtesters, we then focused on the terrain on which our characters would walk.

This part was the most comfortable to research and implement. Instead of a direct work, we have used Pinterest to go through numerous shots of the Australian Great Victoria Desert landscape. With each terrain item, our goal was to be as authentic as possible. Our rocks are red and dry, our sand desolate but smooth, and our grass is shrubby and darker than the average plantation.

These three aspects allowed our team to stay true to the design and aesthetics of the geography and history we decided to inspire the game from, providing a player experience that is loyal to its roots. In addition, we still utilized creative freedom by basing the design of our game on the research we had done to add the touch of humor and uniqueness we sought to have for our game's overall mood.

Sources for Related Work:

The Great Emu War of 1932: Stranger than Fiction!

What Was the Emu War?

Great Emu War: How Australia Started a War against Flightless Birds – and Lost

Old Man Emu Walking

Emus with Dr. Dave

4.3 Narrative Design

If applicable, provide an overview as to the grander narrative content of the game: who, what, where, when, why, and how? In what ways does the player interact with this narrative? Is that separate from how the player's character experiences the narrative? Is the narrative static, or dynamic? Additionally, explain the reasoning behind your narrative decisions. (Add an appendix with the full narrative details if needed.)

Emunuel was once a simple farmer. Just tending to his flock, collecting the food that was generously grown by the humans specifically for them. But that all changed when humans with firesticks showed up. They pushed Emunuel and his flock from the fields of wheat that other humans had prepared for them, forcing them to find food elsewhere. The flock starved until they could not take it anymore. A solution had to be found. The year is 1932, in the Campion District of Western Australia. General Emunuel has gathered his horde with one objective in mind: reclaim the glorious crops of wheat that were easier to steal before. Against the destructive firesticks of their enemies, emus have started to utilize guerilla tactics to return to the fields of wheat to feed their young and raise an army. The Emu War has begun, and Emunuel is here to win.

As the protagonist's controller, the player's interaction with the narrative is built around the mechanics of avoiding the soldiers, leading their flock, and utilizing their skills to retrieve wheat from the fields. As a stretch goal, dialogue can be implemented to further reinforce the dark comedy of this bizarre piece of history, immersing the player not only in the real Emu War but also in the fictional world we have created through it. For simplicity, we aim for the narrative to be static. You are an Emu leading a flock against the human military to gather crops, which we can later develop to have more dynamic elements, such as character development for Emunuel. At the basis of our narrative, we want our player to find the setting funny, as Emu War is perceived more as a meme than a tragedy, so it would be integral for the game's narrative decisions to remain loyal to that media representation for audience expectations.

4.4 Gameplay

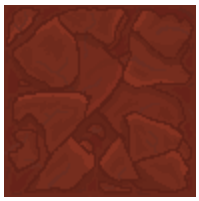
How many players are supported? What are the general controls/mechanics? Is it turn-based or real-time? How does the player progress through the game? Add/adjust sections as needed to appropriately describe the entire player experience.

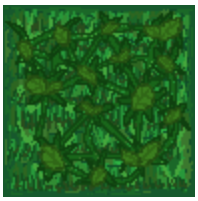
The gameplay consists of a real-time single-player experience with movement and horde mechanics controlled by the keyboard and mouse. To aid the player and allow them full tactical control, they can use the mouse to point a member of the Horde to run to a position by clicking the left mouse button. The player moves using WASD, and the Horde tracks the player's movement and follows it, surrounding the hunter. For additional agency, the player can gain a quick burst of speed or point out where the nearest crop will be. The player progresses through the game by clearing levels of the wheat crop while keeping at least the main character alive. There is the incentive to gain a bigger horde, as the player gains points from collecting each member of the Horde and also gains, in essence, a meat shield surrounding them.

4.5 Aesthetics

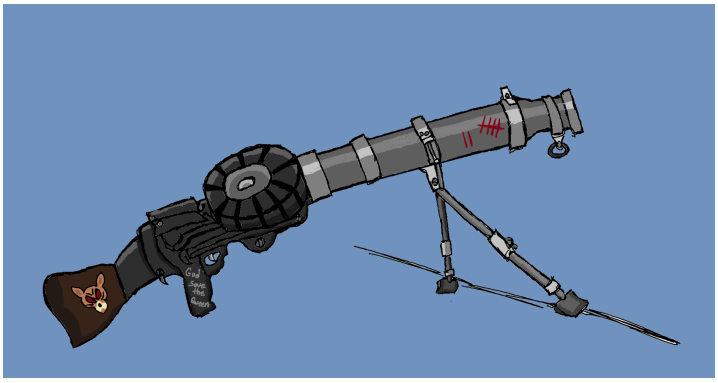
Mood, feel, inspirations, how it fits with the game, etc. PLEASE use visuals, etc. to SHOW the aesthetics. Let us see what you're making - screenshots and a few sentences about how this shows how these aesthetics support the game design.

For our game, we have wanted to utilize pixel art to create a mood that is serious in tone but goofy in presentation. For our environment, we researched Australian geography and Emu habitat to get a sense of what the terrain of our levels would look like, and we have designed them based on that analysis.

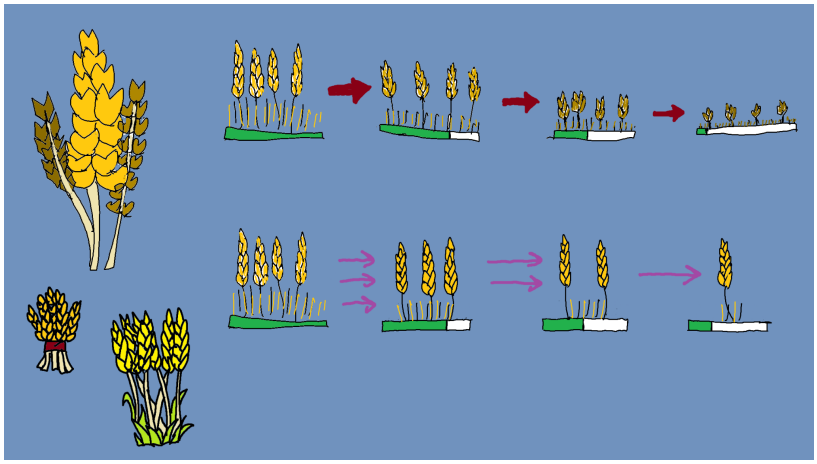




As for our enemies, we went for a type that we call “The Hunter.” Hunters are directly inspired by the Australian uniform standards during World War I, equipped with the Lewis Machine Guns that they used during the real Australian Emu War. Through creative freedom, we have also given them gas masks worn during WWI to make them look scarier and less human as the main antagonist of our game.



Next, our collectible: Wheat. The art style for this asset came easily. We simply aimed for it to represent the grain accurately. What we discussed in detail, however, was the animation of the wheat as it is being collected. The first concept proposed is to reduce the size of wheat on the bush as it is being collected. The second concept kept the wheat size the same but reduced the amount as it gets collected. We ultimately decided to follow the latter option for our final design.



Finally, the UI icons for our player abilities. We wanted to provide on-theme representation visually, hinting at what the abilities do. The boot with wings provides a temporary speed increase to the player character, while the compass shows the direction of the nearest collectible. Just like the equipment of the Hunter model, the style of these two icons was also directly inspired by WWI gear to fit into our mood.

4.6 User Interface

What are the major game modes? How does the player get information about the game status? Visual, audio, controls (not details, but what can the user control? Timing? Precision? ...) -- things that matter in terms of how the player interacts with the game. How is player onboarding accomplished? Feedback to the player as well - interfaces go two ways!

There is only one game mode for the game that involves the player collecting emus into their horde, eating wheat, and avoiding and evading hunters. There are three game states, the main menu, which includes the controls menu and credits scene, the game, and the game over screen. The player gets information about the game status by performing some action to move to the next state. In the main menu, they click a button that says “play” to start the game. In the game state, they have information about their health, how many emus they have in their horde, and how much wheat they have collected out of the wheat in the level; they also have an interface that informs them of their player abilities. Once the player either collects all of the wheat or dies, the player enters the game over menu, displaying their score and win or loss condition.

While in the game, the player will receive visual cues when eating the wheat with an animation that shows it disappearing (getting eaten), along with the health bar for the wheat. There are also visual cues for hunters shooting and their line-of-sight with animated bullets and a ray-casted vision field. The player abilities also get visual cues with animated timers that show how long they have to recover the ability and when it is in use; the player controls when they activate each ability.

Onboarding for the game gets accomplished through the tutorial sequence and controls screen, allowing the player to get the hang of the basic mechanics early on.

4.7 Onboarding

How is the player introduced to the objectives, mechanics, etc. IN GAME?

As a smaller project, our plan for Emu War is to present the game's core idea from the start by bringing the player the features they will need to get accustomed to as soon as possible. There are three main objectives in the Emu War.

First, the player will learn how to collect wheat in the game. In the controls menu, the player will learn that the game's objective is to collect all wheat in each level. In the introductory section of the level, the player must walk into a wheat crop and see the interaction between the emu and crop before moving on to the rest of the level. Second, the player has to amass and maintain their flock. The player will find out that having more emus in the flock is beneficial through the user interface to stay informed about the size of their flock, hinting that it is an integral value to the gameplay. There is also another piece of wheat after amassing the first members of your flock to show that they eat wheat faster with a flock. Lastly, the player must avoid confrontation with the military, as they lack the tools to match their machine guns. The game will introduce the player to this idea from their first interaction with a soldier. The game will directly or contextually prompt them to stay away from the enemy. If they get detected, they will most likely get shot, thus learning the threat level of the encounter.

Avoid the military to keep the emus in your flock alive, and have as many in your flock as possible to gather more wheat. Through this simple yet effective design, Emu War has an easy-to-learn experience that should not require excessive user interfacing or direct information to introduce the core mechanics to the audience.

5 Assets

Provide the readers a high-level overview of the look-and-feel of the art and music that you developed for the game. Since the complete asset list might be quite large, move it to an appendix and provide just an overview in this chapter. Use the space here to provide examples (& sources!) and address how the assets serve your desired game design goals.

This isn't design choices (that goes above in game design). Grouping and constraints with asset choices, how you organized them, sources, etc. A non-technical overview of your asset pipeline would be appropriate here. But not ALL your assets go here. Give a sampling.

Erika suggests using this to segue to the technical design. You game design set detailed objectives. Assets fill in how you'll achieve some of those. The technical design will describe in more detail how those assets were used to implement the game design.

STILL NO BULLETED LISTS HERE! Provide context for how you made decisions about these. It provides background for readers, but also sets up how to make decisions when collecting assets for future development. Imagine what you would tell a content creator about expectations for creating assets to fit the game design.

***How** did you approach/address soundtrack, audio feedback, character animations, environment, feedback animation, UI -- those are the basic buckets.*

Our game relied on two key factors for its art assets. First, every asset has to be pixelated with 96x96 resolution. This detail was paramount in achieving a consistent look to our game, ensuring that the product was visually appealing and complete. Our player character, enemies, wheat, terrain, and ability icons have all been completed following this rule. Since our lead artist had done most of the assets, we avoided variations between art styles. Second, all our assets had to remain visually loyal to our source material, the Emu War. We wanted our characters, General Emunuel and Hunter, to have their style mainly inspired by their real-life parallels with a comedic twist. Emunuel is an Emu but wears military gear. The Hunter uses Australian WWI gear but has also been embellished with a painted gas mask for characterization. Additionally, our terrain has purely been created with direct inspiration from the Australian landscape.

We created our assets based on prioritization of importance. Our player character General Emunuel came first, followed by terrain to fill our first playtest. Later we added the Hunter and its numerous directional walking & shooting animations to have a visually smooth experience for our only enemy. Finally, we revised our terrain and added the icons for our UI elements to finish the look. Throughout this process, we have learned that directional animations are quite hard to implement when the game follows a 2.5D camera perspective, making Hunter's animations take far longer than intended. We had to distribute the workload to several consecutive sprints to make up for it, which delayed the implementation of other assets and canceled our plans to add non-terrain props into our environment.

As none of the main members had past professional experience with sound design, we have decided not to invest considerable time and effort into creating art assets. For this semester, we have avoided implementing audio assets, leaving our game to be mute. For future development, we would consider music to serve our game's theme and ambiance. Additionally, sound effects would reinforce the minor stealth elements of our player experience through auditory triggers when detected and using abilities.

6 Technical Architecture

This is your technical design document. No code should be in the document unless there is something incredibly clever/core to the implementation of the game.

Please do NOT go overboard. This is NOT an onboarding doc for new developers - it's your overall architecture and why you made the decisions you did. *Reference individual research, external references, etc. as needed. LOTS of pictures!*

Focus on models that might be helpful and the level of discussion that helps readers know how to interpret the model. Super detailed models are generally useless. Make sure the level of detail matches the depth of discussion in each section. Lots of small models are much more useful.

Even in technical design, the language here should be in the context of your game! Avoid jargon, etc. As said by a past IGME-601 student, ***frame everything in the context of "Architecture in Service of Gameplay":***

> *I want to see how each team considered the architecture of the game and how it could be best designed to deliver gameplay that lent itself to the larger vision of the game.*
> - *Discuss the evolution of the code and how the design decisions influenced gameplay.*
> - *When the reasoning for decisions is discussed, explain how this architecture is best suited to make the game fun for players and show off their skillsets vs solely referencing ease of development and implementation (which is still very important!)*
>
> *...*
> *Without this how the design choices should best translate to technical design aren't obvious, so knowing how best to implement any given mechanic seems like it might be a personal judgment call rather than something that would be derived from the game design.*

Focusing on how the technical aspects are in service of the game itself:
1. Motivates why this technical work is important.
2. Keeps the non-technical members of the audience engaged (and even the technical ones - because tech alone is really dry.

6.1 Overview & Approach

Discuss, in plain terms, your overall approach to technical design for the game. Think of this as a summary of the rest of this section. You should probably include a diagram of major subsystems and any other important information needed to understand the structure of your technical design. Address how its design serves your desired game design goals. Make sure to cover fundamental choices you made in the development process of the game. Start with a big picture view and then organize/describe your technical approaches and decisions.

The core aspect of our game will revolve around our three main subsystems: the player, their horde, and the hunter. The Player is one of our main subsystems, and it will interact with the Hunter, as the player will try to avoid the Hunters as much as possible to survive and remove their risk of death. The Player will also interact with the Horde, as the Player will be able to collect the members of the Horde by running into them. They will also be able to throw the Horde members out to distract Hunters to avoid being shot. The Hunter will interact with the Horde, as they will try to shoot and kill said Horde with the same motivation that they are attempting to kill the player.

6.2 Related Work

Add other subsections to discuss all the background work you did for the technical design and implementation. Appropriate papers and games should be cited here.

This section should be about the research you did as a team to support specific elements of the game. I.e. you didn't pull your game design or implementation out of thin air - you based it on prior work (other games, other research, etc.). You'll also likely have references to related work throughout the various subsystem sections. This section is for the higher-level, major technical related research you did before going into detailed design and implementation.

6.3 Emu (Player)

*Add sections/subsections as needed to describe the various elements/subsystems *and* how they interact with each other + how they helped you meet the objectives of the game. What were the major decisions you made for each subsystem? Remember, the focus is on architecture in service of gameplay. This should provide context for technical readers and be interesting to non-technical readers (even if some of the details are more than they need).*

Movement

The player can take control of the emu and move it with keyboard controls WASD. While moving, the camera moves with the emu with a small offset to allow for more fluid movement in the game view. This movement scheme evolved from the original, where the emu constantly moved toward the cursor and would stand still if the cursor collided with the emu.

Speed Ability

The Speed Ability interacts with the movement system. When the player uses the Speed Ability, the Movement speed increases for a set amount of time; once that time is up, the Movement speed returns to normal. The Speed Ability has a cooldown before the player can use it again.

Wheat Sense Ability

The Wheat Sense Ability interacts with the Wheat system. When the player uses the Wheat Sense Ability, a signifier appears near the emu, pointing the player toward the nearest Wheat crop. The Wheat Sense Ability only finds and calculates the angle to the nearest crop when active to save overhead. The signifier only appears briefly, hiding once the set time is up. The Wheat Sense Ability has a cooldown before the player can use it again.

6.4 Hunter

*Add sections/subsections as needed to describe the various elements/subsystems *and* how they interact with each other + how they helped you meet the objectives of the game. What were the major decisions you made for each subsystem? Remember, the focus is on architecture in service of gameplay. This should provide context for technical readers and be interesting to non-technical readers (even if some of the details are more than they need).*

Object Pooler:

The object pooler is a way to create a set number of destroyable objects that can be recycled after a large batch is created on run time. While this was only implemented for the bullets, the script had functionality for any game object to be pooled. The object pooler is a dictionary that holds a queue of GameObjects stored by a key. Using this pooler, we can call upon already existing objects and activate them instead of creating them in the instance. Additionally, instead of destroying objects once they have fulfilled their purpose, they can be deactivated and reused.

Vision Script:

The basis of this script is based on a tutorial provided by the YouTube channel Code Monkey. The vision is based on a raycast field of view from the player's perspective. A set number of rays are fired out at set intervals. If and where the raycasts hit provides Vector3s used to construct a mesh. Raycasts allow this mesh to be created and respond to its environment. For example, if the ray collides with a wall, the mesh will respond to this object and not go through it. (Code Monkey, 2019)

The secondary function of these rays is to check if an emu, player, or horde is detected within the vision cone. If one of these elements is found, a signal is sent to the main script to begin shooting.

Peripheral Script:

This script is attached to a hidden circle collider surrounding the hunter. If the player or a horde member collides with the circle, the same logic will occur had the player entered the vision mesh.

Movement Script

This script handles how the hunter moves around in the world and is responsible for handling responses from the vision script to transition into shooting.

The script rotates through a set of waypoints to travel towards. What movement the hunter takes depends on whether they are in the moving or rotating state. In the moving state, the hunter moves toward the current waypoint. After they reach the destination, the next waypoint is cycled, and the hunter enters the rotating state. In the

rotating state, the hunter looks towards the next waypoint over a short period. After this time expires, the moving state occurs again. If, at any point in this process, the player or a horde emu is detected by either the vision script or the peripheral script, then the hunter enters the shooting state. This state's logic is handled in a separate script.

Shooting Script

This script is called upon once the player has entered the shooting state. In this state, the hunter rotates to look directly at the entity (player or horde emu) closest to the hunter. After a brief pause, the hunter will fire five bullets. The bullets have been created at run time from the object pooler. After the bullets have been fired, the script checks if any emu is still in sight. If they are, the shooting process begins again; otherwise, the hunter returns to the rotating state and reorients itself to the next waypoint.

6.5 Wheat

*Add sections/subsections as needed to describe the various elements/subsystems *and* how they interact with each other + how they helped you meet the objectives of the game. What were the major decisions you made for each subsystem? Remember, the focus is on architecture in service of gameplay. This should provide context for technical readers and be interesting to non-technical readers (even if some of the details are more than they need).*

Wheat is a vital component of the game; eating all of it will progress the player to the next level or the end of the game. Wheat interacts with three systems: the player, the wheat sense ability, and the horde.

While colliding with the player, the health of the wheat diminishes at a rate of 1, plus a boost if there are emus in the flock. This decision promotes the player collecting other emus and colliding with the wheat.

One of the subsystems of the player, the wheat sense ability, also interacts with the wheat. The wheat sense ability gets all the wheat in the level and points toward the nearest wheat. This decision helps players find all of the wheat in the level so that they may progress without searching the entirety of the level for one or two small pieces of wheat.

If the horde collides with the wheat, the health of the wheat diminishes at a rate of 0.1, plus a boost for each emu in the flock. The horde's rate is much lower than the player's because the wheat tracks all collisions with it. Therefore, if the player and a horde emu collide with the wheat, there's already a 1.1 rate applied to the diminishment of the wheat. As the horde gets larger, there are additional collisions along with a considerable boost for each emu in the flock.

Wheat Health Bar

The wheat health bar shows the player how much health the wheat has left. It starts hidden, but once the wheat takes any damage, it will appear for the player above the associated wheat. As damage gets done to the wheat, the health bar gets smaller, until

eventually, the wheat has 0 health left, where the wheat and the health bar get destroyed.

6.6 Horde

*Add sections/subsections as needed to describe the various elements/subsystems *and* how they interact with each other + how they helped you meet the objectives of the game. What were the major decisions you made for each subsystem? Remember, the focus is on architecture in service of gameplay. This should provide context for technical readers and be interesting to non-technical readers (even if some of the details are more than they need).*

Horde and the hoarding mechanic is one of the four main mechanics we wanted to have in the game at the bare minimum. The Horde is one of the two main playstyles we wanted to include in the game: either sneaking around the hunters and collecting wheat on your own or collecting and amassing an enormous amount of Emus to decimate the wheat. The Horde will interact with the player, emus, wheat, speed-up ability, and the hunter.

When an individual member of the Horde who is uncollected collides with the player or an already collected emu, they will become a member of the Horde and follow the player throughout the level. We made this decision so the player could collect a group of followers to join them, collect more emus, and eat wheat faster.

The Horde will impact how the wheat itself is collected for each Emu added to the horde, and it will increase the Players own consumption rate to 1 + 0.025 * the size of the horde. While each Emu in the horde will be consuming wheat at 0.1 + 0.025 * the horde size. The reason behind making these numbers different is that we want the wheat not to be reduced at too quickly of a speed if it is simply the horde eating the wheat, but we still want them to actively eat the wheat, as otherwise, it wouldn't make sense.

The Player's speed boost ability also interacts with the Horde, as it will cause the rest of the flock to speed up with the player. This decision was made so that it doesn't force the player to leave their Horde behind when using this ability. However, after some feedback from the pitches, we are considering how this will impact the Horde moving forward.

The Horde itself will interact with the Hunters, and their AI as the Hunter will hunt the Horde and other Emus who are not in the Horde regardless of if they are in the Horde or not. This logic also plays into the Player/Horde ability, which allows the Player to shoot out a member of the Horde in a targeted direction, thus allowing them to use this to potentially distract a hunter, letting the Player and or the rest of the Horde get away successfully. The Horde will also interact with the Hunter, as when it gets shot, it will die. This element incentivizes the player also to collect Emus for the Horde, as they can block shots that might hurt or kill the Player.

The Horde is also implemented into our main score element, adding 50 to the score per Emu you collect in your Horde.

6.7 Score

*Add sections/subsections as needed to describe the various elements/subsystems *and* how they interact with each other + how they helped you meet the objectives of the game. What were the major decisions you made for each subsystem? Remember, the focus is on architecture in service of gameplay. This should provide context for technical readers and be interesting to non-technical readers (even if some of the details are more than they need).*

The score is the main system that challenges the player to walk through the level, collect as many emus and wheat crops as possible, and suffer from minimum loss of the horde in the shortest possible time, which gives the player a tense experience of seeking survival for the horde.

For each wheat crop collected, the player will get 100 points. For each emu collected, the player will get 50 points.

In addition, we give bonus points rather than a penalty to those excellent leaders who keep alive and collect all the wheat crops for their great performance faced with the army and leadership of the horde:

5000 bonus points during 00:00 - 01:59;

4000 bonus points during 02:00 - 02:59;

3000 bonus points during 03:00 - 03:59;

2000 bonus points during 04:00 - 04:59;

1000 bonus points during 05:00 - 05:59;

No bonus points since 06:00;

7 Playtesting

With all of the previous chapters presented, a question remains. Does your game actually address the goals you set for the team? You need to explain how you playtested your game. Keep all of this focused on product knowledge -- this isn't about any general research. It's about YOUR game.

7.1 Overview & Approach

What were the overarching questions and challenges you needed to address with all playtests? What specific aspects of your game were always going to need more playtesting? What made this hard? ...

3C's (character, control, and camera) are the main overarching challenges we face with all playtests (Pluralsight, 2014). Since Emu War is a top-down 2D game, players' game experience greatly depends on whether they're provided with a proper movement of

the character, a comfortable way to control the character with the mouse and camera followed smoothly. To address 3C's, every time we iterate our game, we need more playtesting on the movement of our character with adjustment of the cursor-following control and the camera's speed. 3C's are hard because they're involved with many aspects of the game, like gameplay and level design. For example, control by dragging and clicking give players more time to think about their decisions, while if we want the game to be fast-paced, we need cursor-following control to realize the real-time movement of our character by mouse.

7.2 Playtest 1/Movement & Terrain - 9/22/2022

Add a new subsection for each playtest... There should be a matching subsection in Release Summary describing what was completed for each playtest. If you link to it, you can reduce redundancy here.

What were the objectives? How did you prepare? What info did you collect? What were the results? How did this influence game design, development, priorities, and the objectives of your next playtest?

Give your focus group questions and results that you obtained. If the datasheets for your results are too long, put them in an appendix. After obtaining results from an experiment, there should be a discussion of how this affected the design of the game. What do the playtest results "say" about the game?

The objectives of the playtest were to get feedback from our playtesters on the main character's movement control and how the camera movement feels. However, we also fielded feedback about the main character's appearance and animation. Therefore, we prepared by building terrain and some items that, although not interactable at the moment, could help the playtester understand the value of moving around with the main character.

The results of the playtest were mainly concerned with the camera movement being too slow to keep up with our main character. However, comments regarding the movement seemed optimistic, reinforcing that it felt natural to move with mouse control and that collisions functioned adequately. One remark regarding potentially removing or changing the cursor prompted further discussion about the implications that might have on our game. In terms of the camera movement, many of our playtesters expressed concerns about the main character leaving the central area of the screen and how it might affect interacting with our stealth elements in the future. On the other hand, many of the playtesters enjoyed the aesthetic of the main character. Still, they wished there were more animation frames for different walking directions and commented on the legs' "broken" look.

The results of the playtest strongly influenced camera movement and keeping the player closer to the center of the screen during gameplay. We also reflected on the ideas presented regarding changing and expanding upon the main character's animation. Many of the playtesters expressed the wish to interact with the wheat bundles on the ground to get a better sense of movement and include some kind of object to avoid to get an idea of how stealth might feel. These comments sparked further discussion on collisions with wheat and how we want our NPC enemy AI to function and build both into our next sprint.

Datasheets from this playtest are included in the appendices section.

7.3 Playtest 2/Wheat, Hunters, & Level - 10/13/2022

Add a new subsection for each playtest... There should be a matching subsection in Release Summary describing what was completed for each playtest. If you link to it, you can reduce redundancy here.

What were the objectives? How did you prepare? What info did you collect? What were the results? How did this influence game design, development, priorities, and the objectives of your next playtest?

Give your focus group questions and results that you obtained. If the datasheets for your results are too long, put them in an appendix. After obtaining results from an experiment, there should be a discussion of how this affected the design of the game. What do the playtest results "say" about the game?

Many team members consider this build the "real first playtest" since we presented more than just movement. Our initial goals for this playtest were to determine how people felt about the hunter AI and the level design and victory conditions. We wanted to ensure the level felt fun and dynamic and played well with the hunter and emus.

Overall the playtest showed us that our design needed to be more intentional and compact. A playtester referred to it as "Looking around in Metroid for the item I need." The AI could be frustrating to deal with at moments, and many players did not understand the meaning of the vision cone. One gripe we received from playtesters was that we included the hoarding in our UI, but we did not implement the mechanic yet. This implementation caused some confusion and a note for the team that nothing should be hinted at that is not implemented. Overall, nothing in the playtest shocked us, and the feedback proved valuable.

Datasheets from this playtest are included in the appendices section.

7.4 Playtest 3/Abilities, Hoard, & Score - 11/10/2022

Add a new subsection for each playtest... There should be a matching subsection in Release Summary describing what was completed for each playtest. If you link to it, you can reduce redundancy here.

What were the objectives? How did you prepare? What info did you collect? What were the results? How did this influence game design, development, priorities, and the objectives of your next playtest?

Give your focus group questions and results that you obtained. If the datasheets for your results are too long, put them in an appendix. After obtaining results from an experiment, there should be a discussion of how this affected the design of the game. What do the playtest results "say" about the game?

After our last playtest, we used our time efficiently to prioritize what needed to get done and how to get it done. We quickly changed the behavior of our Hunters to ensure that they are more challenging yet also more predictable to avoid confusion from our players. We have also adjusted our Level layout based on the feedback of our playtesters. They considered the original level far too large and directionless, so we have reduced its size and placed game objects within it more meaningfully placed. Thanks to how quickly we responded to that feedback, we got to work on the horde system, and the scoring for this playtest. Outside our scope, we have also decided to add abilities to the game to increase player agency aside from the hoard.

Our updated Hunter and Level, however, was well received. It was clear that the changes we had made through feedback significantly improved our play experience. Furthermore, integrating the abilities opened a new design space for our playtesters to meddle with. Though the playtesters highly endorsed the idea of abilities, some considered them too strong. Our user interface was also a recurring discussion during the playtest. Though currently only a placeholder, the user interface we have and the feedback we got from it has provided us with ample data to design our final interface according to those suggestions.

One thing we were admittedly shocked about was the bugs. There were a few issues with our new features which we had not successfully caught during our internal playtesting. The scoring system did not keep track of the player properly. It did keep track initially, but a bug resulted in the final score being inaccurate. Additionally, there are several improvements to be made to the hoard. We observed that our playtesters did not use them the way we intended. Instead of a resource to be collected and used, our playtesters have instead only collected the hoard and avoided spending them. We will be looking back into their design space to discuss how we can motivate our players to use them as intended.

Datasheets from this playtest are included in the appendices section.

7.5 Playtest 4/Animations, Raycasting, Movement - 12/1/2022

Add a new subsection for each playtest... There should be a matching subsection in Release Summary describing what was completed for each playtest. If you link to it, you can reduce redundancy here.

What were the objectives? How did you prepare? What info did you collect? What were the results? How did this influence game design, development, priorities, and the objectives of your next playtest?

Give your focus group questions and results that you obtained. If the datasheets for your results are too long, put them in an appendix. After obtaining results from an experiment, there should be a discussion of how this affected the design of the game. What do the playtest results "say" about the game?

The objectives of this playtest were to get feedback on our new animations, the hunter raycasting ability, and the new keyboard-based movement controls; however, we also fielded feedback about the game as a whole in its minimal viable interaction state. We prepared the minimal viable interaction for the playtest, allowing the playtesters to see the game in its entirety in its current state and give feedback on everything they thought necessary. Our methodology focused on using a feedback form and tracking what playtesters were saying and experiencing during the playtest.

The feedback results stated that they liked the new hunter animations, although there were sometimes some bugs with the orientation of the animations. For raycasting, many of the playtesters seemed to enjoy the improved field of vision. However, there were some comments regarding the field of view being too big, a distaste for the cone stopping at the player's location, and some bugs where the ray-casted cone would not

appear. For the movement, players found it much more intuitive to use WASD controls, but there were a couple of comments about the ability to move faster while moving diagonally. With the additional comments available for the MVI, some players were confused about the genre of the game as they felt it was more action-based but still required a lot of stealth due to the lack of healing ability, meaning that they had to play more cautiously to avoid losing. There was also a lot of distaste for sacrificing the horde for the player's success.

Moving forward in development, we know we have to fix some of the animations for both the player and hunter for improved quality of life in the game. The major issues, though, are that raycasting needs some adjustment in size and visibility and that we also need to fix the diagonal movement bug. For design, we'll need to consider how many stealth elements we want to maintain and how much more we want to dive into action strategy by considering items such as health packs.

Datasheets from this playtest are included in the appendices section.

7.6 Future Work

Knowing that the game likely isn't "done", what would be the priority for future playtests?

For the coming Playtest 1, we prioritized testing the basic cursor-moving control and camera.

For the coming Playtest 2, we'd like to test our first level, the basic AI and winning conditions.

For the coming Playtest 3, we'll perfect our hunter AI, polish the level, and add power to the emus. We want to pay more attention to these during the playtest.

For the coming Playtest 4, we're to have our final level, raycasting, which makes the hunter more intelligent, and new keyboard control. We're seeking feedback on these priorities.

In the future, we would like to get more feedback on new levels and adjusted bugs from the feedback we received in Playtest 4, ensuring that we're reaching our desired play values.

8 Postmortem

Every research report has two final sections: Conclusions and Recommendations/Future Work. This document merges these sections into the post mortem, which is a common industry practice. In the last section, you discussed how the playtest results confirmed or denied what did or did not work about your game. This section extends that discussion by reflecting on why those responses happened for your audience and your team.

As part of this discussion, include what you learned (e.g., how to improve the team dynamics, the game, and the processes). Everything in this section should consider your ENTIRE project, both semesters, from multiple perspectives: game design, production process, technical design, implementation, ….

For each, cover: What went well? What didn't? What impact did this have on the game and project as a whole? What did you learn as a result?

This is NOT about criticizing your entire project. It's a candid discussion of strengths, weaknesses, and lessons learned -- Have you reflected on the entire experience? Do you understand what went well and what didn't, and, more importantly, do you understand WHY that happened. This is the story of your journey with some hindsight, but without beating yourselves up.

Write to your past selves. What would you have wanted to know from past teams so you could learn from their experiences?

8.1 Overview

Give an "executive summary" of the entire postmortem chapter.

The focus of creating *Emu War* was experiential design in learning about game development processes. We originally planned on creating a stealth and action-strategy game but pivoted when we learned that players favored the action-strategy elements more; pivots of this variety were common throughout our development process, utilizing playtests as the rationale. We will discuss how and why we made these adjustments in our design and how the project's various components informed said design. Lastly, we will go over what future work might look like for *Emu War* and what we would do to improve it if we were to continue working on it.

8.2 Reflections

Look back to what you wrote for each section above - you should probably be touching on all of the major topics/decisions you made and reflect on the effectiveness of those. Add/break apart subsections as needed!

8.2.1 Production

Including team dynamics, etc. -- there IS a way to professionally discuss issues here. Talk to us about how! Remember, this will be read by others! Content here needs to be from a team/process perspective, not about individuals!

Team Organization

At the beginning of this semester, we formed the team based on our common interest in the game idea. We feel proud of our idea, and every team member played different roles with our respective expertise and contributed to the game with great passion. However, when we look back, there were two main issues with the team formation.

One is the risky distribution of manpower. Our team has Jan as the only artist, but he cannot show up with the team toward the end for some travel reasons. Fortunately, the

project didn’t need any new assets at this stage. However, in case of such unexpected situations happened to him earlier, the asset production might not meet sprint MVPs.

The other is the divergent skills. Silver, our level designer, is not as familiar with working with Unity as with Unreal. It took the team some time to find a way to incorporate him into the project work.

In conclusion, we learned that team formation should consider common interests and a balanced workforce to decrease the risk of unexpected time away.

Planning and Scope

Given limited resources and manpower, we set five clear priorities and successfully finished our game in the final. We feel like the five overall priorities always motivate the team toward our MVPs.

We’re glad that we managed to meet our semester MVP set at the beginning of this semester, which is credited to each team member's contribution and continuous improvement in the processes of the whole team. Even so, the process of achieving sprint and milestone goals is not easy. We found that we struggled with some sprint goals while some others were easy. So, we think we should have planned sprints and milestone goals more smoothly.

Unfortunately, we don’t have enough time to polish our game further to achieve the stretch goals. Most of the stretch goals would become our future work. It’s probably because we ignored the time spent on debugging each build. We realize we should consider debugging as stories though they might not come from our backlogs (because we don’t know there will be bugs until we come across them).

Game Development Process

During the project, we enact processes concerning communication, task management, version control, and so forth. We used Discord, Jira, and GitHub throughout the project.

We notice that communication might be the most crucial process for our team. Effective communication helps our team form a shared vision, deliver ideas among team members and build a positive team environment.

We used Discord throughout the project as the main communication tool because we often needed to communicate online for immediate feedback. However, we were sick of communicating with more and more information in the channel. Hence, we created more channels based on different topics, which classified different types of messages so that we could easily find the information we needed.

Though we can get down most things online, we felt in-person meetings could provide us with a better synchronous working environment and exceptional face-to-face

communication. So, we have two weekly in-person meetings on Wednesday night, when we usually wrap up, and Sunday at noon, when we usually have retrospectives and sprint planning.

We've used Jira throughout the project to implement task management. In the early stage, when there were few stories, we were not concerned about the priorities, but later we learned to assign different points to stories to figure out what we needed.

We've used GitHub throughout the project to carry out version control, by which we can do respective work. We set up a dev branch to avoid any branch directly merging into the main branch. In addition, we set up several branches to develop new features. Version control served well for our project. Even so, we still learned two lessons.

One is not merging until the last moment because we don't know how many bugs there might be. One night, all team members struggled with debugging to build until about 2 AM. From then on, we brought forward code view from Wednesday night to Monday night.

The other is keeping the habit of committing. During the last build, our level designer lost his stash by accident and could not recover the levels on the repository due to the lack of commitment.

8.2.2 Game Design & Aesthetics

Be sure to discuss more about how your concept evolved and priorities changed (& why!)

Gameplay

When the team first assembled, we initially agreed to four base mechanics; solid player movement, a well-designed map, hunters that accurately react to the player, and the ability to horde other emus. Based on our discussions with other teams, our initial planning was heavily underscoped. This underscoping was due to the unique time constraint that we could only work four hours a week on the project. Ultimately, this low production load allowed us to accomplish our objectives.

The initial pitch of the real-world emu war fed into our project's initial design and tone due to the ridiculous nature of the failed attack. However, the genre of our game changed over time. Initially, we thought stealth would play a big factor in our gameplay. Our initial pitches of Emu War dictate the game as a stealth game. However, over time we moved away from this concept. One factor was based on our playtests. Playtesters were more keen to go in aggressively to collect wheat crops than play cautiously and avoid the hunters. We have some theories on why stealth was such an unpopular playstyle, but it mostly fell on our production cycle. The elements of gameplay that highlighted being stealthier, like tall grass the player can hide in, or abilities to blind the hunters, were a low priority on our backlog. With the satisfaction rate of the players already high and the lack of production in stealth-based elements, we decided to

remove stealth from our pitch and gameplay. By the end of the first milestone, we redesigned the pitch of our game to describe Emu War as an action-strategy game with minor stealth elements. Concluding milestone two, we removed any reference to stealth altogether, settling on action strategy as our genre.

The base gameplay of our player changed drastically throughout the production cycle. Much like the genre for our game, the basis of these changes was based on playtesting. At our first playtest, Kyle designed our movement scheme to revolve around using the mouse to guide the player in a 2D environment. The control scheme was simple to use, and players had fun controlling the player. However, this movement scheme also presented some bugs. If the player held the mouse close enough to the dead zone, Emunuel would twitch, beginning his movement cycle one frame and ending it the next so long as the mouse was there. We intended this exploit to be a bug we can fix in the upcoming milestones. The control scheme worked fine for a while, but the playtest in the middle of milestone three forced us to reconsider. In playtest three, we introduced many new abilities to the player. These abilities included throwing members of your horde, a speed boost, and an indicator to the nearest wheat crop. The horde throwing ability was bound to the left mouse click. The other two player abilities were bound to the 1 and 2 buttons on the keyboard, respectively. These new abilities highlighted the flaws of the mouse movement system. Not only was the deadbox issue still present, but players also had difficulty throwing emus while trying to run away from the hunters. After discussion, we decided to move the base movement to keyboard control since mouse movement presented too many problems to fix before we finished our production. We were able to switch player controls to the keyboard successfully. This switch allowed easier use of the emu throwing and the two other player abilities already assigned to the keyboard.

The hunters were our antagonists and the necessary force to counter the player. Ryan was the one in charge of implementing this feature. The first rendition of the hunter was ready by the second sprint. The hunter could move towards a list of waypoints and rotate to the next one when needed. A cone hitbox was indicated by a png attached to the hunter to show its vision. When they detected the emu, it would fire in a five-round burst, similar to the Australian guns. This feature was the base-level functionality and could be improved drastically. The common consensus was that the hunters were too dumb. Players can run behind them and not be detected. If the emu walks out of the hunter's range, they will do nothing to pursue it. The second playtest was to add complexity to the hunters. Before the third playtest, the plan was to give the hunters a detection circle around them, have them keep the emu in a line of sight, and change how they detected the players from a base collision polygon to a raycasted mesh. The first two objectives were accomplished in time, but the raycast was too bugged to present. However, we fixed the bugs before the final playtest. Some bugs persist, with the mesh occasionally not appearing on screen in the WebGL build. However, the hunter's complexity provided a good opponent to the player.

The ability to control a horde to help you against the hunters was the last of the main tasks we sought to complete during the production cycle. This idea came from our

discussion of how the emus could have won against the hunters in the first place. The general observations were that the emus were too big and fast to kill easily and that the hunters were overwhelmed. Our initial intention with our design was to be able to split a chunk of the horde and fire it in another direction while still having control over it, similar to Agar.IO. However, as the base gameplay developed, we realized implementing this mechanic would contradict what we had already set up since we had already made a single-player model in Emunuel. As such, we modified the ability to throw emus by tossing a single emu in a designated direction.

Level Design

In the beginning, we considered making several levels. We continuously collected feedback from our audience and iterated the level design of our game after each playtest.

In our first playtest, there was no level. We had only been developing the level for a week and had no time to plan out a level. We only had an aesthetically pleasing room because Ryan pulled assets from another project he had previously worked on.

The team was much more prepared when the second playtest came along. Silver was acclimated to the grid system in Unity and was ready to fulfill the needs as the team's level designer. The level presented was a maze with many nooks and tunnels for the player to navigate. We spread the hunters and wheat crops throughout the entirety of the maze. While we were proud of Silver's work, the playtesters were not so kind. Silver intended for the beginning of the level to guide the player to the first crop while dodging the hunter; not many playtesters followed that path. Additionally, the crops were so spread out players got frustrated looking for everything. One playtester said, "it feels like I'm exploring for the powerup I need in Metroid." Clearly, the level needed refinement to guide the player better while enforcing the game's concepts onto them.

The third playtest scrapped the maze idea in favor of separate quadrants of a distinct theme. The first section was linear to teach the player about collecting wheat crops and the importance of collecting horde members. Afterward, the separate quadrants were open with a different environment. One problem with this design was that the final section was intended to be swamp land, but we did not have time to create the assets and functionality of these tiles. Players found this new level much better in design, as they thought our placements were intentional. The addition of the wheat detection ability also helped players find any loose crops that escaped them. However, they thought there were too many hunters at the end of the level, which unfairly increased the difficulty.

The final level provided a good balance between all previous level renditions. This edition also featured an initial onboarding section like the prior. However, this level had a different open layout that allowed for greater exploration. We toned down the number of hunters to make the game less difficult. Additionally, Silver found a way to manipulate the hunter's waypoints so they would stay in one spot but rotate in different

cardinal directions. This feature led to a much more dynamic and polished level that provided enough difficulty but was still fun and engaging.

Aesthetics

Overall, our aesthetic decisions and art style came down to how we wanted the game to feel and how we wanted to make the player feel as a result. As our game is meant to be something similar to the flash games of the early 2000s, we wanted to go for the pixel art style to set the feel of the world similar to that of the games found on miniclip.com and Adult Swim. Once we decided on this, we wanted to go with a 2.5D perspective on the game so that the player could see enough information while still creating a connection with Emunuel. Once we made these choices, we knew we would have to decide how graphic we wanted the game to be because we were handling a war. After deciding to want it to be played by those around the Teen-ish age, we decided not to make the game too dark and mature. However, we did not want the game to feel as if it was meant and the art was for children; thus, we wanted to make the player, hunter, and horde all have a primarily darker and more natural color palette for their designs. This palette allows us to reinforce the idea of the seriousness and tone of the game to the player. Once this was decided, we knew we had to get the overall terrain of the Australian outback correct. This focus caused us to ensure our three main types of terrain properly matched what was in the Outback. We wanted to honor the red rock formations found in the Outback, so we decided to use these as our bounds for our walls. On top of this, we also wanted to honor the fact that the Outback is primarily sand-based; however, as we are focusing on where the Hunters are growing wheat, thus there would be more grass, especially where we would end up placing our wheat blocks.

8.2.3 Technical Design

Not just what you did, but how these technical decisions impacted the game and project overall!

Our overall technical design focused on intertwined and important mechanics. As a team, we did not want to have our core mechanics not interact with each other. This decision resulted from wanting all aspects of the game to feel important to the player. The decision resulted in us as a team making sure to frontload these tasks (i.e., the player, the hunter, the horde) so that we could get these elements functioning and working together in front of the player as soon as possible. This plan allowed us to gain influential feedback from the playtests that would eventually help correct the course of our hunters and the controls the player uses to move. Through the feedback of our players, we were able to make these interactions not only feel more impactful but also engaging for the players. Also, by having these interactions so intertwined and thus starting them so early, we could hammer away at them and allow ourselves to sophisticate them multiple times to make them as best as we could be in the time we were given. In all, through the intertwined interactions of our player, hunter, and horde, we created an overall engaging experience for our players and ourselves.

8.2.4 Playtesting

Reflect on how you conducted playtesting, not only the results from playtest.

Initially, our playtesting methodology could have been more sound. In Playtest 1, we explained a lot of things to our playtesters and had discussions with them during the playtest about the game, what we planned to implement, and why certain things were the way that they were. Because of our unintentionally defensive mindset, we may have ignored suggestions because we assumed that our future features would deal with said issues. We also utilized a questionnaire that had too many questions for our audience to answer. We learned more about test planning between playtests one and two, including methodology, data collection, research questions, and the effective use of a questionnaire.

In Playtest 2, we focused more on tracking our playtesters' intangibles and on a handful of more specific research questions. We shortened our questionnaire and received more relevant feedback on what we wanted input; there would be three pointed required questions to get feedback on our research questions, followed by one catch-all where the playtester may discuss what they pleased about our game. We still had some dialogue between us and the playtesters that may have resulted in defense of some gameplay mechanics, which may have blinded us from other issues. Overall, this was the turning point in the project, where we found better ways to construct our methodologies and utilize questionnaires.

In Playtest 3 and Playtest 4, we continued to focus on the intangible aspects of our playtesters, including where they got frustrated and what they seemed to enjoy unknowingly. We were more hands-off, waiting until the playtester was going to fill out the form to inform them of aspects of the game we wanted feedback on that they ignored. This methodology helped us understand that crucial elements of our game weren't apparent to our playtesters, and we had to make pivots accordingly. We still utilized short questionnaires at the end of the playtest to get feedback in writing about what our determined research questions were for that playtest. After they completed the questionnaire, we would have discussions with the playtesters to wrap up the session and discuss elements of the game that they commented on to get further feedback.

Overall, our methodology for playtesting evolved from our less rigid playtests at the beginning of the project. We found much more valuable feedback on what we were focusing on because of it. Moving forward, each team member will continue to organize their playtests similarly due to the effectiveness and value of these results.

8.3 Conclusions

Discuss, overall, how the project went. State what you (or someone else) would (or should) do next to improve the game AND your process if you were to continue this project.

As a team, we would like to say that the project was successful, as it allowed us to learn more about working in Unity and with processes and with each other as a team. For the most part, this was the first time the team had worked together, so there was a lot of initial work to ensure we all understood how the rest of the team worked and how we could thus work best together. We eventually met the goals we set for ourselves for our project as we wrapped up our semester, and this came as a result of the processes we followed throughout the semester. The team faced hardships, but we persevered due to the well-structured processes we had enacted and clear communication. Something that both we and we would expect someone to say on which to improve our processes would be to make sure to have more stand-ups, try and dissolve knowledge silos, and do more documentation synchronously. We would expect these things, as these are what we found to be what we wanted to improve if we were to continue this project. Towards our final milestone and sprint, we found that the structured stand-ups we had been following began to fall apart. As a team, this impacted us a bit and caused our production to waver slightly; thus, we wanted to make sure if we were continuing the project, we would want to keep the process more high-ceremony regardless. On top of this, we wanted to dissolve knowledge and asset silos, as this was something that did impact us when unforeseen circumstances would arise and take a team member out of commission at times. While these were all unpredictable, one way we believe we would be able to mitigate this in the future is to have raw files and rationale in a shared space like Google Drive while also having the team somewhat overlap with each other so we can flex over to support them if needed comfortably. Finally, as a team, we recognized that we needed to do more documentation synchronously, as it would allow us to stay ahead on the documentation itself and communicate better about certain aspects that needed the full team's attention.

On top of this, as a team, we recognized that there were some things that we still wanted to do to improve the game; however, we, unfortunately, did not have the time for it. A lot of these things were more polishing up existing portions of the game; however, there were more aspects that we would have liked to introduce, like the tall grass mechanic, potentially another style of hunters, and most importantly, more levels, so the game feels more engaging and challenging for the player. These were all aspects of the game that we viewed as stretch goals that, if we had the time, we would enact as they would help bring the game to the next level and elevate it.

8.4 Future Work

Finally, assess the project's readiness to move into production and document what would be required to do this. This section should, in effect, act as a mini-business plan. You can organize this section however you see fit (I want to make it engaging and structured appropriately for the current status of your game). However, you must cover:

- *How are you going to evaluate and measure your future progress?*
- *What are the risks associated with your project and what are your plans to mitigate these risks?*
- *What resources do you need to continue? (Consider this from a people, skill, hardware & software perspective.)*
- *What major milestones are you planning and what is a rough timeline in which you think these can be accomplished (assuming access to the required resources)?*
- *What are your long-term plans for the project?*
- *What is your plan for distribution and marketing?*

If you really intend to never touch this again, don't just bail on this section. Explain why.

Emu War was an experiential design project focused on learning about game development processes. For the game to be considered for the market, the game would have to go quite a long way, even when measured against other flash-style games. Though the team does not believe that *Emu War* needs to become a marketed game or publicly brandished, we are happy to keep *Emu War* as this experiential design project useful for a portfolio and discussion of game development processes. On top of this, we hope that those who read our documentation can learn about the value process has on creating a good experience.

As previously mentioned, there are currently no plans to develop *Emu War* further, as we feel it fits the bill for a portfolio piece in its current state. While *Emu War* is starting to come into its action-strategy identity, and the team enjoyed working on it with one another, we feel that we are ready to move on to other projects.

If the team were to continue working on the game, we would have to deal with the risk of transitioning further into the action-strategy genre while properly managing and balancing the minor stealth elements. We feel that, in terms of resources, the only thing that prevented moving the project further was time, as we were limited to four hours of development per week; our team makeup was ideal for the development of this project. We would also expand the project's scope to provide more mechanics for players to participate in. Delving more into the action-strategy genre than we had while still including minor stealth elements in an expanding scope should be enough to build a marketable flash-style game around.

9 References

More on Citation:

Give credit to ideas that are not yours and the games that influenced you in ALL written work. Use APA style, which is standard in the Learning Sciences disciplines.

Recommendation: use Zotero free bibliographic software to automate your citations. It allows you to store citations for future reference and automatically reformats to different styles. Note that Zotero APA still requires capitalization correction to "sentence case" as opposed to "title case" - https://www.zotero.org/support/google_docs

Use the following page for learning in-text citations and how to write a references section:
https://owl.purdue.edu/owl/research_and_citation/apa_style/apa_style_introduction.html

For games, cite designer (as author), year, and publisher using APA style – choose one of these:

- *http://askus.library.wwu.edu/faq/116850*
- *http://moonflowerdragon.blogspot.com/2010/03/how-to-cite-game-in-apa-style.html*
- *https://www.bibguru.com/g/apa-video-game-citation/*

If it is not your original idea, you must cite!

Banerji, U. (2016, March 21). In 1932, Australia started an 'emu war'-and lost. Atlas Obscura. Retrieved from https://www.atlasobscura.com/articles/the-great-emu-war-australia

Character, controls, camera: The 3cs of game development. Pluralsight. (2014, September 22). Retrieved from https://www.pluralsight.com/blog/film-games/character-controls-camera-3cs-game-development

Code Monkey (2019, October 13).Field of View Effect in Unity (Line of Sight, View Cone) [Video]. YouTube. https://www.youtube.com/watch?v=CSeUMTaNFYk

10 Appendices

Asset details, pitch Q&A notes, playtest feedback details, etc. If you don't want to copy/paste large amounts of data here (e.g. raw survey results), you can link to it elsewhere, but it MUST be in a publicly accessible location with a permanent link, NOT a Google Drive. Hosted from a public GitHub repo or blog works well.

Playtest Feedback Results #1: Emu War - Playtest Survey #1 (Responses)

Playtest Feedback Results #2: Emu War - Playtest Survey #2 (Responses)

Playtest Feedback Results #3: Emu War - Playtest Survey #3 (Responses)

Playtest Feedback Results #4: Emu War - Playtest Survey #4 (Responses)