Production II Project

Milestone Guide

(Jak Tiano, Ian Sartwell, Ryan Leslie, Shain Strother, Evan Schipellite, Tim House)

Prepared by: Evan Schipellite

Technical Synopsis

While this project may undergo several gameplay iterations over the course of the next few sprints, its overall direction should allow for a stable foundation to be built and coded from the beginning and onward. The game itself is based on recreating a 'party game' experience where four players, all utilizing controls and viewing the same screen, will eventually conflict with each other in two primary phases of gameplay. First, players will begin by drilling into one or more planets, thus allowing them to race against each other to collect robot parts. At this stage in the game, players are primarily seeking to at least locate a Torso, Arm pair, and set of Legs to continue to the next stage. After a time limit, in some manner players will combine the three types collected and then engage in combat. Using gravity systems surrounding the planets, the players will compete in a battle of elimination to dictate a winner. Obviously, numerous design and mechanic elements are still vague and in development, but from a technical perspective there are core pieces that can be developed while design undergoes testing and iterations to finalize the methods for gameplay.

Programming Requirements / Further Introduction

In terms of creating the foundation of the game, the project can be divided into building the two phases of gameplay. While the nature of gravity and rotational movement should be created first, afterwards it may be most beneficial for prototyping to iterate on a single phase. Therefore, building the basis for the combat system will yield the best results for presentation purposes. If the proposal is successful, and after enough of the combat features are present, it may then be possible to go about creating the initial drilling game phase in a manner that best translates to the second part of the game. However, for current purposes, personnel limitations, and overall project scope; only the conceptual design for the first part of the game needs to be developed and tested while the overall combat system is integrated into the project.

The game will be created in Unity, and will furthermore take place on a single screen in a partially 2D environment. 3D models will be utilized for the planets, robots, and other environment pieces, but the game will be played from a side view. In each corner of the screen the players should be able to visualize their hull percentage (health), their parts they have selected for the fighting scenario, as well as their potential cooldowns for abilities. At the current stage of design, players will have a simple set of controls that will be consistent through all stages of combat. Designed specifically for controllers, the left analog stick will be used for

moving, while the right will be available for aiming the player's weapon. The four triggers in the back of the controller will be used for jumping, Torso power, Leg power, and Arm power. In other words, each part collected will have its own designated button to activate its ability. For instance, grabbing a sword will allow the player to activate the arm button to swing, while grabbing a set of hover legs could allow them to hover for a short duration by pressing the leg button.

During the initial phase of the game, and likely to evolve and change over the duration of the project, players will not readily have means to combat, and instead will focus on movement through planets to collect items. However, in the second phase of the game, the mechanics present will be required to have much more universality and precise design to promote freedom of gameplay. Due to the rough design of 9-12 parts currently in progress, players may have several combinations of weapons and abilities, and each level should be prepared to accept various movements and combat scenarios that occur. For instance, a player could have a dash and hover ability, and these features should be acceptable in the level's physics structure.

Combat should allow players to freely run / rotation around the planet's center. However, players may also have access to other areas such as smaller planets, moons, ships, satellites, and platforms. Therefore, utilizing a jump or aerial mechanic, players should be able to access the planets orbit in order to increase their maneuverability, and they should also be able to break free from a planet's gravitational force. In the event of breaking free, players should be able to drift to other planets, and once they reach the edge of an area, they may appear on the reverse side in order to prevent them from permanently leaving the screen if they miss their landing opportunity. Projectiles should behave in a similar manner. If simply fired, projectiles should combine their initial velocity with the planet's gravitational force, thus causing them to fly around the planet. If shot at a more upwards angle, projectiles should also have the chance to also maintain orbit or break free from the gravitational force. This will allow players more freedom in attacking players that are not on a planet's surface.

Within 2 weeks this project should be able to convey a foundation for a basic combat system that allows 1-2 players to fight on 1-2 planets without many glaring glitches or bugs. Players may only be restricted to a few types of combat mechanics, but all additional mechanics designed should integrate easily into the physics structure. If this game continues onward, the additional 3 weeks can be used to continue advancing on additional mechanics, therefore stressing testing through QA sessions. From there, if this project continues forward, additional personnel will allow for more effective design, as well as the creation of the first phase of the game.

Sprint 1a: Risk Expectations

Required Components

- Technical Risk Assessment
  - During this week, the programmer documentation should be created and properly set-up on Chile. While only one programmer may currently be working on this project, this document should be designed to not only provide an area for scheduling and reflection, but it should also be presented in a manner that will be easy enough to read for additional programmers, as well as useful in its manner of dividing up iterative Sprints and distributing tasks each week. (This should also be related to the Product Owner for constructive adjustments to the documentation's structure).
- Physics Tests / Creation
  - Before any combat is created, basic physics tests should be conducted to test the flexibility of the system. A basic prototype for planets and player rotation has already been built, and therefore the next phase involves testing the limits of the system and improving its flexibility.
  - Without projectiles or combat, Players should be able to easily navigate around and between planets using their jump.
  - Gravitational forces should also be comfortable and make sense for the player.
- Basic Player Architecture
  - Players will need to be designed in a manner that allows them to accept a torso, leg, and arm component.
  - This means that the Player class may be responsible for undergoing creation with the addition of a torso, leg, and arm. Polymorphism should allow for the Player to utilize these parts without the need to own specific types.
- Combat
  - Basic project shooting could be useful for further testing the physics system.

Optional Components

- Melee combat
  - An introduction to a melee arm piece would be effective in assisting testing balance between ranged and melee pieces.

Risk Assessments

- Technical Risk Assessment (**Easy**)

    - Not much more is required aside from simply setting down the plans for the first few sprints. Writing may take a while, but the process will ultimately serve to streamline ideas and concerns about the upcoming milestone.
- Physics Test / Creation (**Medium)**
    - This ultimately depends on the universality of the physics abilities already in place. It is likely that new features involving orbiting and drifting will still need to be implemented. These may take time to develop, as well as time to add to the current system.
- Basic Player Architecture (**Medium)**
    - This should just be a matter of creating a polymorphic structure for the torso, arms, and legs. Programmatically this feature should be simple, but it will also involve syncing the project art pieces with the intended ability, which may take some design.
- Combat (**Medium**)
    - Combat will be more time-consuming than difficult. It will involve possibly creating health system and projectile art. Furthermore, projectiles will need to undergo testing in order to work properly, but previous ventures have shown that this is more than possible.
- Melee Combat (**Easy**)
    - Seemingly easy, but due to the sheer amount of work, this may need to wait until another sprint. It would be nice for testing, but would require testing in itself in order to design in a fun and appropriate manner.

Sprint 1a: Review

All intended factors, although not bug free, have been completed and properly implemented for this sprint. There are multiple issues with regards to fixing gravity bugs and tuning mechanics, but those will likely become a goal for a later sprint as controls and gameplay continues to undergo design iterations. Players at the current stage could move freely around the map while shooting projectiles at each other. Projectiles would properly extend for a duration, and the force of getting hit would allow for gameplay additions in terms of repelling players away. This aspect was received well, although the force of the weapon should be lowered and projectiles should be destroyed on any collision. Melee combat was never implemented, mainly becomes various complexities relating to it and controls was uncovered toward the end of the sprint, but since it was considered optional, this isn’t an issue for the current moment.

Sprint 1b: Risk Expectations

One of the important features to remember going onward is the necessity for testing gameplay with the addition of mechanics. Initial testing procedures should be based around the aspect of managing the actual bugs and large-scale problems resulting from the activation, use, and combination of various mechanics. However, eventually once mechanics become more grounded, this testing will turn to the power balancing of mechanics. Also, for this sprint, orientation and gravity glitches will need to be examined before any additional implementation of mechanics can be conducted. This is mainly due to the fact that the player needs a way to flip their orientation, and current implementations conflict with the gravity components. The player also has some issues with regards to colliding incorrectly with gravity pieces. Furthermore, it is recommended that the majority of testing be conducted utilizing controllers from now on, as the keyboard controllers are becoming messy and the controllers will resolve a lot of issues if utilized now.

Required Components

- Technical Risk Assessment
  - Just an update to discuss this week's issues.
- Physics Fixes
  - A number of issues have surfaced with regards to gravity including: orientation, jumping, and rotating.
  - This could take a bit to fix depending on how it conflicts with other pieces of code.
- Controller Support
  - Controller implementation should be conducted during this sprint. This means that the current 2 test players should be managed by controller input.
  - This will involve axis conversion.
  - This could also conflict with forces for movement.
- Sword Mechanic
  - A basic sword mechanic should be added.
  - The player should be able to swing in an arc.
  - Enemies within the slash should be damaged and encounter a knock-back force.
- Combat (Health)
  - Players should have available health bars.
  - Upon reaching 0, the game should simply reset for prototype purposes.
  - Weapons should damage.
    - Projectiles may currently be designed to deal 50% to the owner if hit.

Optional Components

- Additional Mechanic
  - Depending on time available, an additional mechanic could be investigated.
  - Focus on smoothing out bugs should be emphasized over this feature.

Risk Assessments

- Technical Risk Assessment (**Easy**)
  - Simply requires time to conduct the update procedure.
- Physics Fixes (**Hard)**
  - This may take some time due to the fact that a number of minor problems exist within the current system. Correcting most of these problems may only take a few changes and tests, but some may take a bit to properly resolve without further hard-coding the system.
- Controller Support (**Easy)**
  - The hardest part here will be acquiring at least 2 controllers. Implementing the buttons shouldn't be too difficult, and will simply involve axis conversion.
- Sword Mechanic (**Hard**)
  - While ideas are available, the sword mechanic will likely undergo a number of changes until it is perfect. For the current moment, it should display a striking movement and damage / knock-back the player(s) hit. This may take some time to properly create the striking effect.
- Combat Health (**Easy**)
  - Merely a matter of noting the player's health at the start, showing a GUI bar, and deducting it when hit. The drawing of the GUI bar may take some time to learn how to do in Unity. Furthermore, Health should be based on player pieces as well, and the health bar could reflect that.
- Additional Mechanic (**Medium**)
  - Since it is not a requirement, this isn't an actual impediment this sprint. However, if time allots, it may take a decent amount of time to implement another feature, especially if it is a non-arm mechanic.

Sprint 1b: Review

This sprint wound up taking a bit more time than initially expected. Various issues arose during the process, notably relating to the controller support, and therefore more time was allotted to deal with these issues. As a result, the additional mechanic was not created, and some of the features implemented / tested should be further improved and reviewed during the next sprint. Initially, it was assumed that the way Unity handled controllers would be viewed, by the Input Manager, as a standardized process. However, this was not the case, and furthermore Unity views each controller as its own separate entity, thus making it difficult to bound buttons properly without restricting the controller types. Due to limited resources, and the reality that the

game may be built on the Ouya, Playstation, or Xbox, it became imperative to create a solution that would at least account for the differences between the three console controllers.

A very straightforward, but painfully hardcoded solution would be to allow the game to dictate which controllers were in use, and then create two different button schemes for Xbox / PS3 controllers. However, even this became an issue due to the fact that some Xbox / PS3 controllers were viewed by Unity as 3rd party sources, and therefore could not be read in the same manner. For instance, a PS3 controller in use had its arcade axis stick and D-Pad swapped, which caused Unity to read in all of the inputs from different sources. While we might have further limited our expectations to certain official controller types, a 3rd party InputManager became available to use from Patrick Hogan called InControl.

This InputManager, while taking a while to setup and install, allowed us to continue develop since it essentially served to create a standardized set of inputs for all controllers. It should be noted that the Xbox controller treats the back triggers as a single axis, thus making it difficult to simply swap between reading in PS3 and Xbox input. However, InControl served to solve all of these issues by creating a wrapper to standardize all input. Furthermore, the InputManager provides useful functionality to test values, bools, and vectors relating to buttons and different axis locations. Time was allotted to account for unknown controllers and register them within the system, but the InputManager would save us time later down the road.

Other than that, the Psychics system would take a bit of time to work with projectiles and the addition of the sword mechanic. These features would require further development as we go on, but at the moment we do have a multiplayer functionality with decreasing health and gun / sword combat. The camera still has some issues, and the weapons require hit registration to assist the players. The sword may also need to be adjusted so that it works a bit between with rotation and orientation. Orientation wound up being another problem due to the fact that the players orientation would have to account for the rotation of its other mechanics and parts. With that said, some additional time was spent finalizing absolute movement for the rotation of the arms, and this created a specific case for the rotation of the sword mechanic.

Sprint 2: Risk Expectations

Due to the complications during the previous week, the best route for this sprint would involve examining the refactoring of code and improving current features. Afterwards, it is best to create a number of new mechanics that can be further tested and balanced in the upcoming week. Factors such as the camera, weapon powers, and movement have become noticeable issues. Therefore, during the process that code is cleaned up, these features should be fixed and improved to allow for further testing. Movement should be designed based on absolute direction. In other words, the player should attempt to move in the direction held on the axis controller. Due to the potential for the current InputManager to provide Vector returns on an axis, this shouldn't be too difficult to detect. However, it will involve Vector math to detect movement within an angle. So, if the player attempts to move to the right, the game should register that movement within a 20-30 degree angle of the precise direction.

Camera should be redesigned a bit. Currently it attempts to keep all players on the screen, but the manner in which is does so hinders gameplay. The camera should remain at a fixed height during gameplay, and all players should be kept within few. However, an exception to this rule involves players that fly off the planets. While the players should be viewable within tolerance, it would hinder other players if the camera zooms out too far. Therefore, if a player goes beyond a certain threshold, the camera should thereafter assume the threshold point as the player's location. This should create an effect where the camera follows all players until they leave the screen limits, thus continuing to stay zoomed out at the max until they return. Future iterations may provide UI components to show where the player has flown off to.

Also, this week should not be focused on balancing, as balancing will become much more effective when other mechanics are involved. Players will not typically be wielding one weapon, but three different abilities. Therefore, a leg and torso mechanic should be created this sprint. These will allow players additional abilities to test, as well as further convey how planet movement will be conducted during gameplay. The results from these new mechanics should provide the foundation for balancing all of the mechanics as more are improved, balanced, and created in the near future.

Required Components

- Technical Risk Assessment
  - Just an update to discuss this week's issues.
  - Might take a bit more to set out the information and review from last week.
- Refactoring
  - Camera needs to be improved so it no longer bases itself on height.
    - Bounds should be set for players past certain distances from the center.
  - Code should be cleaned up and organized for simpler changes later.
- Absolute Movement

        - The player should move based on absolute movement
            - This means that vector input should be read and translated to the world
            - This should smooth gameplay and allow players to feel as though their characters move in the exact direction the axis faces
    - Health Bar
        - Health bars should be visible above players
            - This should show the percentage of their health remaining
            - Future iterations should convey the actual amount
                - Possibly placed on the screen HUD
                - Could also be conveyed by divisions
    - Torso Mechanic
        - A torso mechanic should be added this week
            - Meetings will determine which one
            - Likely it will be the Body Slam torso to provide movement testing
    - Leg Mechanic
        - A leg mechanic should be added this week
            - Meetings will determine which one
            - Likely it will be the ground-pound leg mechanic
    - Weapon adjustments
        - Weapons should be tested in order to ensure they are presentable
        - Collisions and forces should properly register
    - Art Implementation
        - Any art available should be implemented if time allots

Optional Components

- Additional Mechanics
    - Depending on time available, other mechanics can be added to further gameplay variations.

Risk Assessments

- Technical Risk Assessment (**Easy**)
    - Simply requires time to conduct the update procedure.
- Refactoring (**Medium)**
    - The basic components may be easy, but adjusting the camera may take some work. It should be easy to improve, but difficult to prefect. Just note that this may involve further testing and design going onward.
- Absolute Movement (**Medium)**
    - Getting the Vector input should be easy, but translating it to world space and detecting movement may prove to be challenging. Vector math must be used to detect the movement within a tolerance.

- Health Bar (**Medium**)
  - This may involve creating a plane, coloring it, and then proceeding to create a red background and placing it behind. Then, the plane can be scaled and moved to represent the ratio between current and max health. The logic is known, but the creation in Unity and the development based on the strange rotations around the world may take a bit of additional work.
- Torso and Leg Mechanic (**Medium**)
  - These two features can be lumped together to prevent redundancy during evaluation. Both mechanics are wildcards in the sense that, the time available may determine the complexity of the mechanics created. Therefore, it will be possible that during creation, the simplest mechanics for both features will be created initially. This should make both tasks easy, but unknowns relating to the ability use could create further issues. It's best to finish other tasks before tackling these to properly evaluate the time available.
- Weapon Adjustments (**Easy**)
  - This matters more for the upcoming sprint. But, the idea is that any major issues relating to weapons should be resolved if possible. This could just relate to blaring power differences between weapons.
- Art Implementation (**Medium**)
  - While this should be easy, it depends on what art is available.
- Additional Mechanics (**Medium**)
  - Not difficult due to the fact that it's not required, but the more mechanics available for testing, the better. It would be nice to available some variations for legs and torso to improve presentation / testing purposes.

Sprint 2: Review

There were a number of obstacles during this sprint, similar to the previous iteration. All objectives were completed, but perhaps additional time / debugging was applied in order to meet expectations. It's hard to say whether these issues will rise or diminish with the project duration, as many of the refactoring components this week, despite having their issues, did provide a solid ground for testing once completed. During the next sprint, further time should be spent improving the nature of the gravity and movement around planets. The implementation of the torso and leg mechanics has provided better ground for balancing and testing the process of maneuvering around the planets and controlling the orbit system. These features, along with the Rocket Launcher, should undergo further development as we go onward. Now that teams have merged, and this project has essentially passed the initial Greenlight phase, there should be a larger team in terms of programming / testing. This should allow for the simultaneous development of new features while additional aspects are improved and balanced on the side.

Sprint 3: Risk Expectations

While all features were implemented accordingly last iteration, a number of obstacles were encountered that required extra time to be devoted to some of the tasks planned. As a result, some of these features will require additional development and testing during this sprint. There are also two other external influences that will impact progress during this sprint. First, the initial Greenlight phase has passed, meaning that teams have already merged. Originally, plans for this sprint would have been based around the presentation of the concept, but now that there are no official requirements for the upcoming iteration, there should be more time to explore minor fixes and balances for gameplay. Second, as a result of the merge, there is now additional assistance with regards to programming, design, and art. Particularly for programming, it should now be possible to complete tasks with more perspectives and insights into development, as well as divide work flow in order to better allow for the advancement of implemented features.

While there is no official presentation due anymore, the goals for this week may only slightly change from the initial intentions. In order to assist the process of integrating new members into the project, the tasks created this week should be based on tweaks, improvements, and balances to allow the other programmer to get a better understanding of the current structure of the project. This may also come with the process of refactoring with newly suggested ideas that may serve to clean up the code further. In this sense, tasks should be divided accordingly to properly explore the development of the current mechanics. While all the foundations have been created, some of the mechanics contain drastic problems, and many of them require additional effects to distinguish themselves from other mechanics. For instance, the machine gun mechanic and rocket launcher mechanic should both be developed down separate paths. The machine gun should fire faster, deal less damage, contain weaker bullets, and undergo an overheating process that limits continuous fire. The rocket launcher should fire slow, but powerful, shots that interact with gravity in such a way that players can utilize the fall of rockets to target players on the downfall. The other mechanics will undergo similar develops that will be described below.

This week may mostly be focused on the improvement of current mechanics, which should be done in a manner that allows for the full exploration and universality of the mechanics. This means that aspects such as hit registration and constant QA testing will be applied to further make the game fast, fun, and engaging for players. On top of this, two other primary features should be improved. Currently, gravitational effects and aerial movement are poorly implemented. Players can somewhat fight the effects of movement in the air, which can create some frustrating gameplay. In order to fix this, the programmers will need to work together to develop the best methods that work with the gravitational movement. There is no set solution to this process, but ideas currently relate to increasing the downward fall and limiting a player’s directional influence in the air. These aspects should probably be restricted to the use of abilities. Also, cooldown bars, at least prototype versions, should be created this week in order to properly show when players can utilize their abilities.

Required Components

- Technical Risk Assessment
  - Just an update to discuss this week's issues.
- Orbit Movement Smoothing (Evan / Tim)
  - Improve the gameplay relating to the orbit system
  - Gravity may be stronger when falling into planets
  - Players should not be able to fight orbital movement as much
  - This should allow for players to feel the effects of knock-backs longer
    - Unless an ability is used, players should be limited in directional movement when in the air
- Cooldown Bars (Evan)
  - One bar, split into 3 pieces
  - Create 3 different cooldown bars below the Health bar
    - Colors may need to be tested
    - These colors should connect / relate somehow to the legs, torso, arms
- Rocket Launcher Mechanic Improvements (Tim)
  - Projectiles should fall faster
  - Projectiles should explode (Knock-back in area)
  - Projectile arc / constant force?
  - Should destroy projectiles / not be destroyed by them
  - Projectile should have health
- Sword Mechanic Improvements (Evan / Tim)
  - Balance force components
  - Hit registrations
  - Currently doesn't always hit when in close range
    - Test and fix this if viable
- Machine Gun Projectile Improvements (Tim)
  - Smaller/ tighter fire rate
  - Reload
    - Overheat function
    - When not firing, reduces heat value
- Body Slam Mechanic Improvements (Evan)
  - Currently self-damage
  - Too much force?
- Dynamic Bounding Box (Evan)
  - Players should set Collider based on itself
    - Height based on Torso + Legs
    - Width based on Torso / Legs

        - 2 Colliders for Torso / Leg

Optional Components

- Additional Mechanics (Evan / Tim)
    - Depending on time available, other mechanics can be added to further gameplay variations.

Risk Assessments

- Technical Risk Assessment (**Easy**)
    - Simply requires time to conduct the update procedure. From now on, a meeting to discuss the tasks for the TRA should be done with both programmers present. The meeting will mainly be done in order to scope the week's tasks, but this shouldn't present any forms of difficult for the creation of the weekly iteration.
- Orbit Movement Smoothing (**Hard**)
    - Orbital movement may need to undergo several tests and changes before it is complete. In that sense, it may not even be possible to finish it during this current iteration without extensive testing. The manner in which players interact with the gravitational forces upon jumping or being knocked off planets should be believable and understandable. There is not specific implementation to resolve this issue, so it will take both programmers to effectively improve and validate.
- Cooldown Bars (**Easy**)
    - While easy, these should merely be prototypes for future artistic developments. The cooldown bars should mainly serve to convey when weapons can be used again, therefore assisting balancing going forward. Since the health bar scaling, positioning, and rotation have already been created and tested, this should simply be a matter of duplicating the process and adding additional bars to convey these features.
- Rocket Launcher Mechanic Improvements (**Medium**)
    - This is mostly a matter of taking the current rocket launcher design and creating additions that will allow for the concept to be conveyed more readily to the player. Projectiles should probably have their own set of health and damage to projectiles. This will allow rockets to be able to destroy machine gun bullets, or other future projectiles, without blowing up themselves. On the contrary, a player with a machine gun should feel the reward of shooting rocket launcher projectiles, as eventually they can be destroyed with enough shots. The rocket launcher should also undergo a faster downfall that makes it feel more like an actual rocket undergoing self-force.
- Sword Mechanic Improvements (**Medium**)
    - Mainly, the force and bug issues with the sword need to be resolved. Currently, when in close range, the sword sometimes doesn't work, so this issue would need to be

investigated. However, balancing of the forces, damage, and usability of the sword should be looked at. It might be work increasing the overall sword itself.

- Machine Gun Projectile Improvements (**Medium**)
  - Similar to the rocket launcher, projectiles should have health and damage to projectiles to allow them to conflict with other types of projectiles in the air. Other than that, adding the overheating feature is the only aspect currently required for this task.
- Body Slam Mechanic Improvements (**Medium**)
  - Since this mechanic was made toward the end of the last iteration, there are a number of features relating to it that are probably ridden with bugs. Sometimes to player is damaged by their own use of the body slam, so further development should be conducted to properly register damages and forces.
- Dynamic Bounding Box (**Easy**)
  - Ideally, the player's colliders should be broken into two parts. First, the torso should be considered a collider. Second, its legs should be considered a collider. Currently, a constant collider has been created, but upon creation, the mech should create and managed its own colliders.
- Additional Mechanics (**Medium**)
  - It's probably best to simply deal with the improvements of other features this week. But, to allow the new members to understand how mechanics are added, a new feature could be started in order to convey a better understanding of the code structure.

Sprint 3: Review

While all tasks were tackled, some of the progress in a few of the improvement tasks will definitely need further iteration going forward. This is partially due to the lack of time available, as a result of external demands, but more importantly, the necessity for continued work has surfaced due to the amount of QA testing conducted during this sprint. QA testing has allowed for a flow of suggestions, as well as the notification of bugs, which should now provide the programming team with a substantial base of possible options that will need to be prioritized. While there are numerous improvements we can provide for the game, the reasonable time to do so in the upcoming sprints will limit which tasks are chosen, as well as the order in which they are completed.

Overall movement, orbit, jumping, and gameplay have made tremendous strides. Now, along with fixing up some glitches, a toggle-switch for gravity may be applied and tested. The focus of this mechanic is to not give a player a double-jump, but instead allow them to have more control moving around a planet, and by tapping the jump again in the air, they can opt to undergo the influence of a neighboring planet if within range. This should allow for players to move around planets freely without the hesitation of jumping due to the fact that they would normally be immediately pulled away upon touching another planet's orbit colliders.

All of the ability mechanics have become a bit tighter, although a few may require further exploration. The rocket launcher may need increased fall speed and 2-3 bounces upon landing before exploding. The machine gun, while currently viewed as an overheating weapon, will be redesign to represent a burst fire weapon. All projectile weapons should have slight homing capability, making them easier to aim by restricting precision when firing. Players should still have the window to dodge projectiles, but in the fast-nature of the game, players were constantly finding their projectiles missing only by a slight amount, even when their opponents were not moving.

Other features, such as camera adjustments and collision issues should be examined further. Next iteration is preceded by a week break, but it will be likely that the programming team will be conducting a small iteration within that timespan to further develop some of these features.

Sprint 3.5: Risk Expectations

The iteration for this week may largely depend on the time available for the programming team. Technically, this is considered as a break week, but the programmers are interested in continuing to develop some of the features of the project, even if the tasks are a bit more limited. As a result, this week should be focused on implementing a lot of the previous sprint's suggestions for QA testing, setting variables for editors use, as well as potentially adding a few new mechanics or beginning development of the inventory selection screen.

Currently, the machine gun and rocket launcher should undergo further development in order to reflect the suggestions made during the previous iteration. The spider legs may also undergo a few changes in order to make them more impactful due to the increased gravity supplied in the last iteration. Setting variables for editor use should just be a matter of selecting features to be present in gameplay for the designers to adjust while testing. This has been limited due to the design pattern relating to the individual mech pieces, but it should be possible now that all the pieces are set and the variables are finalized. Furthermore, this week, depending on time, could allow for another torso / leg mechanics to be development for more randomness in gameplay. But, this week could also allow the programmers to begin developing the UI for the inventory selection screen.

Required Components

- Technical Risk Assessment (Evan)
  - Just an update to discuss this week's issues.
- Projectile Base Class (Tim / Evan)
  - Similar features in one class
  - Slight homing w/ variable
- Machine Gun Improvements (Tim)
  - Finalize Burst-Fire
  - Trail-render
  - Little influence by gravity
    - Maybe not at all
- Rocket Launcher Improvements (Tim)
  - Faster fall
  - Bounce (2-3)
  - Explosion (Bigger than projectile)
- Camera Adjustments (Evan)
  - Lerp on Player Death
  - Deal with no players
  - Fix glitches
- Orbit Toggle (Evan)

        - Jump again to move to another planet
    - Spider Legs Improvements (Evan)
        - Redesign impact in gameplay
        - Adding / Subtracting movement from parts
    - Inventory Selection Start-Up (Tim)
        - Create basic UI elements
            - Show Mech
            - Have Selection with All Current Parts
            - Show Health / Speed
            - Color (Not required)
                - Not allow players to have the same color
                - Allow for dynamic / determined

Optional Components

- Additional Mechanics (Evan / Tim)
    - Depending on time available, other mechanics can be added to further gameplay variations.
    - Torso / Leg

Risk Assessments

- Technical Risk Assessment (**Easy**)
    - Simply requires time to conduct the update procedure. From now on, a meeting to discuss the tasks for the TRA should be done with both programmers present. The meeting will mainly be done in order to scope the week's tasks, but this shouldn't present any forms of difficult for the creation of the weekly iteration.
- Additional Mechanics (**Medium**)
    - This is probably a good optional task to have available, since most of the tasks may be limited to prevent them from taking up too much time. In the event that all the other features are completed, a torso / leg mechanic should begin development to allow for further testing possibilities.

Sprint 3.5: Review

All of the required features, except for the Spider legs improvements, were completed during this partial iteration. For the most part, this iteration was limited and based upon the development of current mechanics, as well as a few fixes for various problems. The creation of the basis for the inventory system was also started in order to provide the Designers with a system to begin tweaking and comparing their Game UIs alongside. The Spider legs, due to the changes in gravity forces, will likely wait until an official iteration to discuss the redesign for its features.

Sprint 4: Risk Expectations

This iteration will likely be based in two separate areas. First, further development for the inventory system, mostly centering on the process of connecting it with the second stage of the game, should be enacted for testing purposes. Therefore, the inventory should be completed enough that it can acquire all information from the separate parts, and it display all connected users at the start of the game. From that point, after selecting all of the pieces, the game should begin with players obtaining their corresponding parts. This should mostly be a matter of continuing the foundation created during the previous iteration, as well as researching how to transfer information to the next scene.

Second, further Mech pieces should be implemented wherever possible. Initially, this week was going to be based around creating the first phase of gameplay, however, due to limited time remaining, it is best to go about implementing more aspects for the Designers to tune in the upcoming weeks. Therefore, aside from other tunings to previous objects and fixes, a few other components should be added to allow for the Mech selection screen to be much more diverse. During the next iteration, it is likely that one of the programmers will be entirely tasked with creating and maintaining the first phase of gameplay, so it's important that every part has at least two variations and the Mech selection screen works accordingly.

Required Components

- Technical Risk Assessment (Evan)
  - Just an update to discuss this week's issues.
- Inventory Selection Screen (Tim)
  - Mostly just further iterations
  - Connect the Inventory Scene with the Gameplay Scene
    - This will need to be able to move to the appropriate level scene in the future
    - Pass information regarding Player Pieces / Amount of Players
- Machine Gun Tuning (Tim)
  - Tweak the Machine Gun
  - Quantity of Bullets fires over variable time period
  - Larger bullets, more forces, more damage
- Spider Leg Redesign (Evan)
  - Redesign the Spider legs
  - Stronger impact on landing
  - Faster downfall
  - Stun target?
- Additional Leg and Torso Piece (Evan)
  - These should provide more variation in gameplay

Optional Components

- First Phase of Gameplay (Evan / Tim)
  - While this does not need to be completed, research on how to set-up some of its aspects and connect it to the Part selection screen could be conducted depending on the time remaining. More details will be listed in the following iteration.

Risk Assessments

- Technical Risk Assessment (**Easy**)
  - Just an update, mainly a matter of meeting to discuss the progress from the last iteration and what amounts of time programmers may have available to continue working.
- Inventory Selection Screen (**Medium**)
  - While transferring information shouldn't be too difficult, it will still require some amount of research to effectively move to the appropriate level and initialize the Player Mechs. Other aspects, such as UI visuals may also prove difficult, but are not as crucial at the moment in comparison to simply creating the Menu process.
- Machine Gun Tuning (**Easy**)
  - Shooting all projectiles over a specified amount of time is the only aspect that should require new Code Units. Otherwise, this is just a matter of tweaking the available values.
- Spider Legs Redesign (**Easy**)
  - While a few new aspects may need to be tested, this is largely just a matter of updating this mechanic to work better in the new gravity environment. Now that players are no longer 'floaty' this mechanic should be altered to provide more downward strength and a reliable impact.
- Additional Leg and Torso Piece (**Medium**)
  - While time consuming to add the new features, this shouldn't prevent any large-scale conflicts with other components in the project. Each piece should be based on the design document specifications, and should provide new mechanics that interact effectively in the gravity-based environment.
- First Phase of Gameplay (**Medium**)
  - The creation of this foundation is best left to take up the majority of time in the upcoming week, but if time is available some outlines for the architecture could be created. This is mostly a matter of creating a simple 1-4 player collection game that will transition to the inventory selection screen.

Sprint 4: Review

Timing due to external work further limited the testing and design of the mechanics, but since the focus of this week was mainly to integrate new material for testing, this shouldn't be an issue. The weapons now have all undergone a number of tuning processes that have assisted in balancing the flow of gameplay. The inventory system has further improved, as the functionality

now allows for monitoring of controllers connected and players to begin each round with. Two new mechanic parts were also added, and although may require additional balancing tweaks, still represent decent progress as the remaining parts to create total about six to seven for the remaining milestones. Within the last three iterations before Alpha, it should be possible to implement all game features in time, even if the first phase is discarded to allow for better polishing during beta.

Sprint 5: Risk Expectations

Due to the impending milestone requirements, this iteration will primarily concentrate of documentation from the programmer perspective. While this document has been consistently updated, its presence has been more of a method of generating adaptable milestone for the programmer team. It has proved useful for listing tasks for each iteration, discussing details, and depicting possible risks, but the milestone requires are defined set of technical details and risk assessments for the entirety of the project. Therefore, while the Lead Programmer handles the documentation, other programming work may focus on simply improving the inventory system, adjusting the cooldown display, and potentially adding a new mech part.

Required Components

- Milestone Guide (Technical Risk Assessment) (Evan)
  - Just an update to discuss this week's issues.
- Technical Risk Assessment and Technical Synopsis (Evan)
  - Essentially, these two documents represents two sides of the same coin
  - Both should incorporate a list of all required game features
  - Technical Synopsis
    - Provides specifics for each item listed for the available game features
    - Details platform requirements
  - Technical Risk Assessment
    - Conveys possible risks for each item
- Inventory Improvement (Tim)
  - Simply a matter of providing more functionality
    - Detect controller connect / disconnect?
    - Shows mech on screen
    - Gameplay returns to inventory selection on end
- Weapon tweaking (Tim)
  - General balancing with regards to current weapons and particle effects
- Cooldown Display (Tim)
  - Remove the old cooldown display
  - Potentially test new designs if time allots
- Additional Mechanic (Evan)
  - Add a new mechanic for designers to test

Risk Assessments

- Milestone Guide (**Easy**)
  - Nothing new, mainly just an update to convey this week's milestone requirements.
- Technical Risk Assessment and Technical Synopsis (**Medium**)

    - Formatting is somewhat of an issue. Scarce information is available with regards to the structure of both documents. Research will be conducted, but it will mainly be a matter of formatting the documents in a manner that details all of the feature information in a readable format.
- Inventory Improvement (**Medium**)
    - There may have been hesitation to immediately tackle the issues surrounding controller disconnect / reconnect and model view, but now that the basis is established time should be available to implement these aspects.
- Weapon Tweaking (**Easy**)
    - Other aspects such as presentation and documentation take priority during this iteration. Therefore, whatever can be done to balance weapons can be enacted, but otherwise this isn't a huge concern.
- Cooldown Display (**Easy**)
    - At the very minimum, removal shouldn't be a problem. Testing new designs could take a bit depending on the complexity, but this isn't strictly required.
- Additional Mechanic (**Medium**)
    - The mechanic itself doesn't need to be completely balanced, however the challenge surrounds the fact that the mechanic shouldn't risk game-breaking possibilities before the milestone end. Therefore, while a simple prototype mechanic is fine, caution must be exerted to ensure that it can still be presented.

Sprint 5: Review