**Test Plan:**

**Level Load:**

* **Preconditions:**
  + None
* **Steps:**

1. Launch the game (./TIMELOCK)

* **Expected Result:**
  + The decelerate level is loaded. This should contain the following elements:
    - Background (2 background layers + foreground chains)
    - Boundaries (thin 1px border)
    - Ground/Ceiling (single png)
    - Midground elements (platforms, crates, small chains, spikes, door) loaded from json parser
    - Camera created
    - Player

**Movement:**

* **Preconditions:**
  + The game has been launched, and the decelerate tutorial level has been loaded. (Create player and create camera must have been called)
* **Steps:**

1. Press the **A key**
2. Press the **D key**
3. Press the **W key**
4. Hold the **A key**, then hold the **D key**
5. Hold the **A key**, then press **W key** while holding **A key**
6. Hold the **A key**, then press **W key** while releasing **A key**
7. Hold the **W key,** then press the **A key**

* **Expected Result:**
  + The character will move. The following movements should happen:

1. The character moves left until the **A key** is released. They will accelerate at PLAYER\_WALK\_ACCELERATION px/s until they reach PLAYER\_MAX\_WALKING\_SPEED.
2. The character moves right until the **D key** is released. They will accelerate at PLAYER\_WALK\_ACCELERATION px/s until they reach PLAYER\_MAX\_WALKING\_SPEED.
3. The player jumps repeatedly. Holding the **W key** will cause the player to repeatedly jump. Pressing the **W** key in the air will not cause the player to jump higher. They will move upwards at JUMP\_VELOCITYpx/s, and be slowed down by GRAVITY px/s.
4. The player will begin moving left, and once the D key is pressed, they will quickly change directions and begin moving right.
5. The player will move according to (1) until the **W key** is pressed. Once the **W key** is pressed, the player will move according to (3) while maintaining leftward velocity. (horizontal distance covered while in-air is dependent on how long A key held before jumping (maximized when max walking speed achieved before jumping)
6. The player will move according to (1) until the **W key** is pressed. Once the **W key** is pressed, the player will move according to (3) while maintaining leftward velocity. Once the **A key** is released, the player will halt their horizontal movement and continue with the vertical motion until they reach the ground.
7. The player will move according to (3) until the **A key** is pressed. Once the **A key** is pressed, the player will move according to (1) mid-air until the player reaches the ground. Displacement to the left of their original jump position is directly related to how quickly the A key is pressed (highest displacement when pressed soon after jump)

**Game State:**

* **Preconditions:**
  + The game has been launched, and the level has been loaded. (Create player and create camera must have been called)
* **Steps:**

1. Press the **R** key
2. Press the **1** key
3. Press the **2** key

* **Expected result:**
  + After step (1) the level should restart. This means
    - All states will return to the original state of the level. Any moving platforms will reset, and any physics objects will return to their initial position.
    - If time control is active, it will reset to normal. All time controllable objects will return to their original velocity.
  + On step (3), the decelerate tutorial will load, and the player will be placed at the start of the level.
  + On step (4), the accelerate tutorial will load, and the player will be placed at the start of the level.

**\*\*\* NOTE: the time control tests do not cover cooldowns at the moment as we have set this to be very low since M1 playtesting. Cooldowns will be more relevant when we get into more detailed level design.**

**Player Motion (Coyote Jump)**

* **Preconditions:**
  + The game has been launched, and the level has been loaded. (Create player and create camera must have been called)
  + There must be a height difference in the scene; ideally should be a reachable static platform or a pit on the ground;
* **Steps:**

1. Walk along the platform/ground until falling off the ground;
2. While falling down and within JUMPING\_VALID\_TIME\_MS, press **W** key;

* **Expected Results:**
  + Should see the player able to jump despite being in air.

**Time Control (Decelerate - Activate):**

* **Preconditions:**
  + The game has been launched, and the level has been loaded. (Create player and create camera must have been called)
  + There must be a time controllable object in the level (ie. a moving platform)
* **Steps:**

1. Press the **-** key
2. Wait DECELERATION\_DURATION\_MS (currently 10s)

* **Expected Results:**
  + On step 1, the time decelerate ability will activate. The following should happen:
    - The blue vignette screen shader will fade in over DECELERATION\_EMERGE\_MS (currently 150ms)
    - Objects with the time controllable component will transition smoothly to slow their velocity down over DECELERATION\_EMERGE\_MS (currently 150ms)
      * Precisely, they should move at DECELERATE\_FACTOR x the original speed. (currently 0.2x)
    - Objects will remain slowed down for the duration DECELERATION\_DURATION\_MS (currently 10s)
      * After when the ability deactivates, the objects will smoothly transition to normal speed and the screen shader will fade out again over DECELERATION\_EMERGE\_MS
    - Harmful objects (projectiles shaded with red effect) will be shaded with a green effect for the duration of DECELERATION\_DURATION\_MS
      * Harmful objects in decelerate mode cannot hurt the player
      * Once the DECELERATION\_EMERGE\_MS has ended, harmful projectiles will turn red again

**Time Control (Accelerate - Activate):**

* **Preconditions:**
  + The game has been launched, and the level has been loaded. (Create player and create camera must have been called)
  + There must be a time controllable object in the level (ie. a moving platform)
* **Steps:**

1. Press the **+/=** key
2. Wait ACCELERATION\_DURATION\_MS (currently 10s)

* **Expected Results:**
  + On step 1, the time accelerate ability will activate. The following should happen:
    - The red vignette screen shader will fade in over ACCELERATION\_EMERGE\_MS (currently 150ms)
    - Objects with the time controllable component will transition smoothly to speed their velocity down over ACCELERATION\_EMERGE\_MS (currently 150ms)
      * Precisely, they should move at ACCELERATE\_FACTOR x the original speed. (currently 2x)
    - Objects will remain slowed down for the duration ACCELERATE\_DURATION\_MS (currently 10s)
      * After when the ability deactivates, the objects will smoothly transition to normal speed again and the screen shader will fade out over ACCELERATION\_EMERGE\_MS

**Time Control (Decelerate - Deactivate):**

* **Preconditions:**
  + The game has been launched, and the level has been loaded. (Create player and create camera must have been called)
  + There must be a time controllable object in the level (ie. a moving platform)
* **Steps:**

1. Press the **-** key
2. Before DECELERATION\_DURATION\_MS has elapsed, press the **-** key again

* **Expected Results:**
  + On step 1, the time decelerate ability will activate. Please see the decelerate activate test for the expected behaviour.
  + On step 2, the time decelerate ability will deactivate. The following should happen:
    - The objects will smoothly transition to normal speed again and the screen shader will fade out over DECELERATION\_EMERGE\_MS

**Time Control (Accelerate - Deactivate):**

* **Preconditions:**
  + The game has been launched, and the level has been loaded. (Create player and create camera must have been called)
  + There must be a time controllable object in the level (ie. a moving platform)
* **Steps:**

1. Press the **+/=** key
2. Before ACCELERATE\_DURATION\_MS has elapsed, press the **+/=** key again

* **Expected Results:**
  + On step 1, the time accelerate ability will activate. Please see the accelerate activate test for expected behaviour.
  + On step 2, the time decelerate ability will deactivate. The following should happen:
    - The objects will smoothly transition to normal speed again and the screen shader will fade out over ACCELERATION\_EMERGE\_MS

**Time Control (Decelerate to Accelerate):**

* **Preconditions:**
  + The game has been launched, and the level has been loaded. (Create player and create camera must have been called)
  + There must be a time controllable object in the level (ie. a moving platform)
* **Steps:**

1. Press the **-** key
2. Before DECELERATION\_DURATION\_MS has elapsed, press the **+/=** key

* **Expected Results:**
  + On step 1, the time decelerate ability will activate. Please see the decelerate activate test for expected behaviour.
  + On step 2, the time accelerate ability will activate. Please see the accelerate activate test for expected behaviour.

**Time Control (Accelerate to Decelerate):**

* **Preconditions:**
  + The game has been launched, and the level has been loaded. (Create player and create camera must have been called)
  + There must be a time controllable object in the level (ie. a moving platform)
* **Steps:**

1. Press the **+/=** key
2. Before ACCELERATION\_DURATION\_MS has elapsed, press the **-** key

* **Expected Results:**
  + On step 1, the time accelerate ability will activate. Please see the accelerate activate test for expected behaviour.
  + On step 2, the time decelerate ability will activate. Please see the decelerate activate test for expected behaviour.

**The following two tests should be run with time accelerate and decelerate in different combinations. The expected results are the same, but context dependent on the speed of the platform when time is sped up / slowed down:**

**Physics (Player -> Static Platform)**

* **Preconditions:**
  + The game has been launched, and the level has been loaded. (Create player and create camera must have been called)
  + There is a static platform below the spawn point of the player.
  + There is a static and tall platform to the right of the player, touching the horizontal platform
* **Steps:**

1. Wait until the player lands on the platform
2. Jump a few times
3. Move to the edge of the platform (left)
4. Run to the right into the vertical platform
5. Hold the right run button to continue running into the vertical platform
6. Run and jump to the right into the vertical platform

* **Expected Behaviour:**

1. The player will land on the platform and not pass through
2. After jumping, the player will land on the platform and not pass through
3. The player will remain perpendicular to the floor of the platform until both feet of the player are off the platform (ie no contact between the two)
4. The player will collide with the wall and stop running.
5. Even when holding the run button, they will not pass through
6. Even when running and jumping into the platform, they will not pass through

**Physics (Player -> Horizontal Moving Platform)**

* **Preconditions:**
  + The game has been launched, and the level has been loaded. (Create player and create camera must have been called)
  + There is a static platform below the spawn point of the player.
  + There is a fast moving horizontal platform, and a slow moving one.
* **Steps:**

1. Run and jump onto the slow moving horizontal platform
2. Jump off of the slow moving platform.
3. Run and jump onto the fast moving horizontal platform
4. Jump off of the fast moving platform.

* **Expected Results:**

1. When landing on either platform, the player will accelerate to match the velocity of the platform slowly. The player should slide a little bit on top of the platform before reaching the desired speed.
2. When the player jumps off of the platforms, they should maintain their velocity and slowly come to a stop
   1. (in the air, slowing down by AIR\_RESISTANCE px/s, or if landing on another platform slowing down by STATIC\_FRICTION px/s, flings are hard to control so they are not explicit tests, but make sure to try until you get both cases).
   2. The fast platform should fling the player.

**Physics (Player -> Vertical Moving Platform)**

* **Preconditions:**
  + The game has been launched, and the level has been loaded. (Create player and create camera must have been called)
  + There is a static platform below the spawn point of the player.
  + There is a fast moving vertical platform, and a slow moving one.
* **Steps:**

1. Run and jump onto the slow vertical moving platform
2. Jump off of the slow moving platform.
3. Run and jump onto the fast vertical moving platform
4. Jump off of the fast moving platform.

* **Expected Results:**

1. When landing on either platform, the player will accelerate to match the velocity of the platform slowly.
   1. If the vertical platform is moving up, it will push the player up with it
   2. If the vertical platform is moving down, the player will fall at the speed of gravity (if the platform is moving faster than PLAYER\_MAX\_FALLING\_SPEED, the player will lose contact with the platform.
2. When the player jumps off of the platforms, they should maintain their velocity and slowly come to a stop
3. (in the air, slowing down by AIR\_RESISTANCE px/s, or if landing on another platform slowing down by STATIC\_FRICTION px/s, flings are hard to control so they are not explicit tests, but make sure to try until you get both cases).
4. The fast platform should fling the player.

**Physics (Player -> Object):**

* **Preconditions:**
  + The game has been launched, and the level has been loaded. (Create player and create camera must have been called)
  + There is a static platform below the spawn point of the player.
  + There is a physics object spawning directly above the player
* **Steps:**

1. Do not move and allow the block to land on your head.
2. Jump with the block on your head.
3. Walk so that the physics object falls off your head.
4. Walk into the physics object, and continue walking
5. Jump on top of the physics object.
6. Push the physics object off the platform.

* **Expected Results:**

1. The block will land on top of the player, it will not pass through or move to the ground, remaining on their head
2. The block will move up with the player and fall with the same gravity acceleration.
3. The block will move slightly with the player, but slide off pretty easily
4. The block will be pushed by the player. The player will not phase through the block.
5. The player will land on top of the block
6. The object will fall off the platform with gravity.

**Physics (Player -> Cannon Bolt):**

* **Preconditions:**
  + The game has been launched, and the level has been loaded. (Create player and create camera must have been called)
  + The player is within trigger distance to a cannon
* **Steps:**

1. Do not move and wait for the projectile to hit you
2. Do not move, press **-** key and wait for the projectile to hit you
3. Do not move, wait for projectile to be fired, press **-** key while projectile is mid-air

* **Expected Results:**

1. The projectile will be fired from the cannon with a red-effect shader. Upon collision the player will “die” - this will trigger the shader effect which will “close” the screen by turning it black, and then “reopen” the screen. The player will be respawned at the last checkpoint when the screen returns.
2. The projectile will be fired from the cannon with a green-effect shader. Upon the collision the player will collide and move slightly with the object. The player will not phase through the object
3. Before deceleration activates, the projectile has red-effect shader. Afterwards, same as expected result for step (2)

**Physics (Player -> Gear):**

* **Preconditions:**
  + The game has been launched, and the level has been loaded. (Create player and create camera must have been called)
  + At least one gear is created and reachable to the player.
* **Steps:**

1. Jump onto the gear.
2. Try to walk between two teeth of the gear.

* **Expected Results:**

1. The player should be able to interact with (i.e., can be supported by) the gear properly; the collision box and the visual representation of the gear has discrepancy within a tolerable range;
2. The player can be carried by the gear and orbit around its centre.
3. The player can apply momentum to the gear if it is not fixed. Depending on the gear’s mass and moment of inertia, it will be harder/easier to roll.

**Physics (Multiple Gears):**

* **Preconditions:**
  + The game has been launched, and the level has been loaded. (Create player and create camera must have been called)
  + Multiple Gears are created in the scene, properly joined to each other.
* **Steps:**

1. Observe.
2. Step between 2 gears that are rotating inwards.

* **Expected Results:**

1. The gear system should be properly driven by any gear that offers initial motive; the rotation directions are reasonable;
2. The player is squeezed between the 2 gears but not killed; they will be carried downwards and eventually fall from the 2 gears.

**Physics (Physical Object Rotation):**

* **Preconditions:**
  + The game has been launched, and the level has been loaded. (Create player and create camera must have been called)
  + At least one physical object is created; for testing purposes, objects with large torque or easy to rotate (e.g., spike ball) are preferred.
  + The object is spawned unsupported by any platforms or physical objects.
* **Steps:**

1. Observe.

* **Expected Results:**

1. The object should start to rotate around its pivot point when coming to contact with at least one supporting object; its motion makes physical sense.
2. If the object’s center of mass falls outside of the points supporting it, it will fall over due to gravity.

**Camera:**

* **Preconditions:**
  + The game has been launched, and the level has been loaded. (Create player and create camera must have been called)
  + There is a static platform below the spawn point of the player.
* **Steps:**

1. Move left/right
2. Jump

* **Expected results:**
  + The camera should follow the player, catching up to its position, while positioning the player mildly to the left/right of the screen center depending on their direction (more screen space is left for the player’s forward direction);
  + If the player jumps fast enough (launched by a platform), they may slightly leave the camera’s FOV. It will catch up when the player stops/slows down;
  + When reaching the boundaries of the scene, the camera should be fixed to include most of the scenes on the screen, and no longer tracing the player;

**Render (Parallax):**

* **Preconditions:**
  + The game has been launched, and the level has been loaded. (Create player and create camera must have been called)
  + There is a static platform below the spawn point of the player.
* **Steps:**

1. Move in any direction (up, down, left , right)

* **Expected results:**
  + Each background layer should move **opposite** to the player’s direction of movement according to the following relative speeds (layers ordered in increasing distance to camera):
    - Foreground (chain) should move faster than SOP (speed of player)
    - Midground should move at the same speed SOP
    - Gold/metal background should move slightly slower than SOP
    - Brown/gear background should move the slowest in comparison to SOP

**Render (Particle System):**

* **Preconditions:**
  + The game has been launched, and the level has been loaded. (Create player and create camera must have been called)
* **Steps:**

1. Run until the player reaches max speed.

* **Expected results:**
  + Grey square dusts should start to summon beneath the player when the player is running fast enough;

**Boss Fight (Boss System: Fight initiation):**

* Preconditions:
  + The boss level has been loaded
* Steps:
  + 1. Run to the right until the player comes close to the boss
* Expected results:
  + The boss remains idle until the player is close to it, the boss then starts moving towards the player, initiating the boss fight

**Boss Fight (Boss System: General flow):**

* Preconditions:
  + The boss fight has initiated and neither the player nor the boss is dead
* Steps:
  + 1. Run around the boss room (running across the boss is also fine), using deceleration is also permitted (do not acceleration, as it is being deprecated)
  + 2. Dodge different boss attacks by jumping, running or using deceleration
* Expected results:
  + When the boss is walking on the floor, collision between the player and the boss does not result in player death
  + When the boss has walked for 5 seconds, it will use a random number to choose from a list of allowed attacks. These allowed attacks are determined based on its proximity to the player.
  + Every 30 seconds, the boss enters the exhausted state, which is the only time period the player can damage the boss
  + The boss’s velocity is based on its distance from the player (faster if player is far away, slower if close by, and basically not moving when it shares the same x position as the player)

**Boss Fight (Boss System: Regular and Fast Projectile Attack):**

* Preconditions:
  + The boss fight has initiated and neither the player nor the boss is dead
  + The boss has walked for 5 seconds and the 30-second timer is not up yet
  + The player is sufficiently far away from the boss
  + The boss raises only one of its arms in the direction of the player
* Steps:
  + Note: it might be difficult to do all of the following in one go, you will likely need to test it through several rounds of regular or fast projectile attack
  + 1. Do not move and dodge the first projectile
  + 2. Jump
  + 3. Activate deceleration
  + 4. Run to the other side of the boss
  + 4. Deactivate deceleration and get hit by a projectile
* Expected results:
  + A projectile will be created based on a timer, for a total of 4 projectiles in the direction of the player
  + The fast and regular projectile attacks are identical except the fast version uses a faster velocity for the projectiles
  + The player does not die if they dodge the projectiles
  + The projectiles are always created based on the player’s y position and position relative to the boss
  + The boss always points in the direction of the player and will change its pointing direction if the player runs to the other side (for example, running from left to right of the boss will cause the boss to point to the left first before pointing to the right)
  + When deceleration is active, the projectiles travel at slower speed and the player does not die if they collide.
  + If deceleration is not active and the player is hit by a projectile, the player dies

**Boss Fight (Boss System: Delayed Projectile Attack):**

* Preconditions:
  + The boss fight has initiated and neither the player nor the boss is dead
  + The boss has walked for 5 seconds and the 30-second timer is not up yet
  + The player is at medium distance or close to the boss
  + The boss raises both arms in the air
* Steps:
  + 1. Do not move and activate deceleration when the leftmost projectile starts flying towards the player
  + 2. Run around the room and wait before the second projectile starts flying towards the player
  + 3. Deactivate deceleration and get hit by a projectile
* Expected results:
  + 3 projectile will be created in the air and the player cannot reach those spots even when jumping
  + The projectiles do not fly towards the player right away
  + The 3 projectiles start flying towards the player at different times
  + The direction in which the projectile travels is based on the position of the player
  + When deceleration is active, the projectiles travel at slower speed and the player does not die if they collide.
  + If deceleration is not active and the player is hit by a projectile, the player dies
  + Once the projectiles are created, the boss starts moving again

**Boss Fight (Boss System: Dash Attack):**

* Preconditions:
  + The boss fight has initiated and neither the player nor the boss is dead
  + The boss fight has reached phase 2 (below 80% total health, aka after 1 hit from the player)
  + The boss has walked for 5 seconds and the 30-second timer is not up yet
  + The player is far away from the boss
  + The boss raises both arms in the air
* Steps:
  + Note: this will likely take several tries
  + 1. Activate deceleration
  + 2. Let the boss run past the player
  + 3. Deactivate deceleration
  + 4. Let the boss collide with the player during the dash attack
* Expected results:
  + The boss has fire it its eyes during the dash attack
  + When decelerated, the boss runs more slowly and collision with the player does not result in player death
  + The boss will run past the player if the player either uses deceleration or jumps over the boss at the right time. The boss will continue running until it hits the a boundary (either the left or right tunnel entrance) and changes direction

**Boss Fight (Boss System: Ground Slam Attack):**

* Preconditions:
  + The boss fight has initiated and neither the player nor the boss is dead
  + The boss has walked for 5 seconds and the 30-second timer is not up yet
  + The player is at medium distance or close to the boss
  + The boss fight has reached phase 2 (below 80% total health, aka after 1 hit from the player)
  + The boss briefly stops
* Steps:
  + 1. Do not move for 2 seconds, and start running.
  + 2. Activate deceleration and observe the boss action
  + 3. While deceleration is active, try to collide with the boss as it is coming down or hitting the ground
  + 3. Deactivate deceleration and get hit by the boss when it’s coming down or stand within the impact area when the boss hits on the floor
* Expected results:
  + The boss first starts rising into the air
  + The boss then follows the player for some time before descending towards the floor, and finally hitting the floor, killing the player if deceleration is not active and the player is standing inside the impact area or being hit by it (either when it is on the floor or when it is coming down to the floor).
  + The boss repeats the attack for a total of 3 times. Each time, it follows the player for a different amount of time before descending to the floor.
  + When deceleration is active, colliding with the boss or standing inside the impact area does not result in player death and the boss moves more slowly.

**Boss Fight (Boss System: Exhaustion State and Hitting the Snooze Button):**

* Preconditions:
  + The boss fight has initiated and neither the player nor the boss is dead
  + The 30-second timer has been up
  + The boss is not in an attack state
* Steps:
  + 1. Do not move for a few seconds.
  + 2. Run and jump to collide with the snooze button above the boss.
* Expected results:
  + Upon entering the exhausted state, the boss stops moving and a different rendering is shown. A snooze button is also rendered above the boss.
  + If the player successfully collides with the snooze button during this period, the boss will take some damage and enter the damaged state, then the recover state before starting to move and attack again.

**Door:**

* Preconditions:
  + The player is within a level that is preceding a future level (i.e. either the deceleration tutorial or deceleration level)
  + The player is at the end of the level at a door.
* Steps:
  + Run into the door
* Expected results:
  + Player should be teleported to the next level
  + All entities of previous level should be destroyed, and all entities of the next level should be parsed and initialized in-game.