Testing Plan

For the entities assigned to a team member, that team member will also create tests.

Battle

BATTLE CASES	
Test battle without weapon	Create battle cases and ensure that each case behaves as expected. A battle takes place when the Player and an enemy are in the same cell at any point within a single tick. The conditions for a battle occurring are the same regardless of whether the player moves onto the same tile as the enemy, or vice versa.
Test battle without weapon against spiders	
Test battle without weapon against mercenary	
Test bribing mercenary	
Test battle with invincibility potion	
Test player dies	
Test multiple battles	

WEAPONS AND DEFENCE ITEMS	
Test battle with sword	Assert that weapon and defence items increase attack / defence with correct multipliers.
Test battle with bow	
Test battle with shield	
Test battle with bow and sword	
Test battle with bow, sword and shield	
Test duplicate weapon	
Test weapon durability	
Test Midnight Armour	

Bosses

HYDRA	
Test Hydra is initialised	Hydras are generally considered to be a special creatures similar to Zombies. Hydras are limited by the same movement constraints as Zombies. When a hydra is attacked by the player, there is a certain chance that its health will increase rather than decrease by the given amount, as two heads have grown back when one is cut off.
Test Hydra health gain: increase rate 0	
Test Hydra health gain: increase amount 0	
Test Hydra health regain	

ASSASSIN

Test Assassin is initialised	Assassins are exceptionally powerful mercenaries which deal significantly more damage. When bribing an Assassin, there is a certain chance that the bribe will fail; the gold will be wasted and the Assassin will remain hostile. Assassins are also capable of seeing and moving towards the Player when they are invisible, if they are within a certain radius.
Test Assassin bribe: rate 1	
Test Player lose treasure if Assassin not bribed	
Test Assassin bribed: fail rate 99	
Test Assassin bribed successful	

Buildable

INVALID ARGUMENT EXCEPTION	
Test Invalid Argument Exception: non bow/ shield/sceptre/midnight_armour string	Invalid actions should throw Invalid Argument Exceptions
Test Invalid Argument Exception: item cannot be built	

SCEPTRE	
Test Player can build Sceptre with wood, key and sunstone	Can be crafted with (1 wood OR 2 arrows) + (1 key OR 1 treasure) + (1 sun stone). A character with a sceptre does not need to bribe mercenaries or assassins to become allies, as they can use the sceptre to control their minds without any distance constraint. But the effects only last for a certain number of ticks.
Test Player can build Sceptre with arrows, treasure and sunstone	
Test Player can build Sceptre with wood and two sun stones	
Test Player can build Sceptre with wood and two sun stones and does not use arrows	

MIDNIGHT ARMOUR	
Test Player cannot build midnight armour if zombie toast is present	Can be crafted with (1 sword + 1 sun stone) if there are no zombies currently in the dungeon. Midnight armour provides extra attack damage as well as protection, and it lasts forever.
Test Player builds midnight armour	

BUILDABLE ENTITIES	
Test Player build input entity	Some entities can be built using a 'recipe' by the player, where entities are combined to form more complex and useful entities. Once a buildable item has been constructed, it is stored in a player's inventory. For all buildable entities, once the item is constructed the materials used in that construction have been consumed and disappear from the player's inventory.
Test Player build shield with key	
Test Player can build shield with sun stone and sun stone is kept	
Test buildable response contains bow, shield, midnight_armour and sceptre	

Collectable

CASES	
Test item removed from entity response, added to inventory	Items that can be added to a player's inventory
Test one Key in inventory	
Test initialised Sun Stone can be collected	

Goals

GOALS	
Test Player reaches exit goal	Basic goals are: • Getting to an exit; • Destroying a certain number of enemies (or more) AND all spawners; • Having a boulder on all floor switches; • Collecting a certain number of treasure items (or more); More complex goals can be built by logically composing goals.
Test enemy goal simple: single enemy	
Test treasure goal simple: single treasure	
Test boulder goal simple: single switch	
Test destroy spawner AND destroy all enemies enemies goal	
Test OR goal	

Intractable

ILLEGAL ARGUMENT EXCEPTION	
Test entity id does not exist	Actions that throw illegal argument exceptions
Test entity is not intractable entity	

INVALID ACTION EXCEPTION	
Test Player does not have a weapon	Actions that throw illegal action exceptions
Test Player has weapon but is not cardinally adjacent	
Test Player breaks spawner	
Test Player does not have enough gold to bribe	
Test Player is not within range of mercenary bribe radius	

PLAYER INTERACTIONS	
Test Player within range and has enough treasure to bribe	refers to if the entity can receive interaction updates from frontend, which only pertains to mercenaries and zombie toast spawners. When mercenaries become allies, they are no longer interactable.
Test Player bribes mercenary: loses treasure	

Test Player bribes mercenary with Sceptre	
Test Player fights assassin after mind control duration stops	

Movement

PLAYER	
Test Player can move in each direction	Testing all Player movements behave as expected
Test Player can move through doors with a matching key	
Test Player cannot move through doors without a key	
Test Player cannot move over walls	

MOVING ENTITIES	
Test Spider moves in circle direction	Testing that moving entities are behaving as expected
Test Zombie moves and only moves in adjacent tiles	
Test Spider movement when obstructed	

STRATEGY	
Test fleeing strategy	Testing that strategies result in the correct moment of moving entities
Test stalking strategy	

STATIC ENTITIES	
Test Boulder-switch interaction	Testing that when moving entities interact with static entities, these interactions behave as expected.
Test Invalid boulder movement	
Test Switch-bomb interaction	
Test valid portal interaction	
Test invalid portal interaction	

New Game

VALIDITY CHECKS	
Test IllegalArgumentException on illegal Dungeon Name	Assert that all set up arguments and actions are valid.

Test IllegalArgumentException on illegal Config Name	
Test no exception thrown with valid dungeon name and config name	
Test JSON file with only single player in it and gives correct output on EntiityInfoResponse	

INITIALISATION	
Test static entities are initialised	Test that entities are initialised with a new game.
Test moving entities are initialised	
Test collectable entities are initialised	

Persistance

PERSISTANCE	
Test load game throws IllegalArgumentException if file does not exist	Test that game state is saved in a local persistence layer.
Test simple save and load with player	
Test Player collects sword, no longer has item after reload	
Test if Player reloads from battle, the enemy remains after reload	
Test if Player builds item, item should be lost after reload	
Test if Player breaks spawner, spawner should be back after reload	
Test if Zombies spawn, should be gone after reload	
Test multiple Games saved are shown in list	

Spawn

SPAWNING ENTITIES	
Test Spider Spawn	Test that spoonable entities are behaving as expected.
Test Zombie Spawn	

Time

ILLEGAL ARGUMENT	
Test IllegalArgumentException if ticks is <= 0	Throw illegal argument exception if ticks are invalid.
Test IllegalArgumentException if ticks exceeds total number of ticks	

TIME TURNER AND PORTAL	
Test Test time turner can be collected	If the player has collected a time turner, then two rewind buttons will appear on the frontend. When clicked, these buttons move the state of the game back one tick and 5 ticks respectively and "transport" the current player back to those game states in a time travelling fashion. If a player travels through a time travelling portal, they end up on the same square as the portal, except the dungeon state is that of 30 ticks previously. If less than 30 ticks have passed, then the dungeon state is simply the initial dungeon state.
Test Test time travelling portal initialised	
Test Items are back in environment after 1 tick rewind	
Test Items are back after 5 ticks	

Time Travel

TIME TRAVEL RULES	
Test controller saves previous ticks	When a character has time travelled, either by the rewind buttons or via a time travelling portal: • Their 'older self' still exists in the dungeon as its own entity. If they encounter their older self and **either** are carrying a sun stone or are wearing midnight armour, or they are invisible, then nothing happens. If not, then a battle ensues. • Their 'older self' should take the same path as was taken initially, and unless they encounter their 'current self' (they character being controlled), should eventually travel through the time portal and disappear from the map. • The player's inventory persists across time travelling. This means that if a player picks up a sword then travels through a time portal, the sword remains in their inventory as well as being back on the map available to pick up.
Test time travelling with portal	
Test time travelling for 5 ticks	

Use Item

USE ITEM	
Test IllegalArgumentException is thrown: input item id not usable item	Test that illegal argument exception is thrown when usable item arguments are not valid.

Test InvalidActionException is thrown: Player uses item not in inventory	
Test usable items	