Appendix E: Unit Tests

Below is a table of the unit tests included within this project.

The unit tests are associated with an appropriate requirement to allow for traceability, and the tests aim to check that the code works for any associated requirements. Not all requirements have associated tests, and vice versa - this is because some requirements cannot be explicitly unit tested, and some tests do not link up directly to a requirement, but are still needed to ensure the code functions as intended.

There is a criticality measure against each test, for both acceptance and unit tests - this is to represent how important the test is to the overall function of the code. Criticality is on a scale - high criticality means that if that test fails, the project will not function at all; low criticality means that if the test fails, the project will still mostly function as intended.

ID	Test Name	Purpose	Criticality	Class	Req ID	Result
1.01	testName	Verifies the name of a clue has been set correctly	Low	ClueUnitTest	5	Passed
1.02	testDescription	Verifies the description of the clue has been set correctly	Low	ClueUnitTest	5	Passed
1.03	testTileCoordinates	Verifies the location of the clue has been set as expected	High	ClueUnitTest	5.1.1	Passed
1.04	testEquality	Verifies that identical clues are considered equal	Medium	ClueUnitTest	5	Passed
1.05	testMurderWeapon	Verifies that a murder weapon has been chosen correctly	High	ClueUnitTest	5.1.4	Passed
1.06	testRedHerring	Verifies that Red Herrings have been chosen correctly	Low	ClueUnitTest	5	Passed
1.07	testFinishInteraction	Verifies that the game snapshot class keeps track of how many interactions the associated player has left in their current turn correctly	High	GameSnapshot Tests	2.1.5	Passed
1.08	testGetNPCs	Verifies that data stored on NPCs is stored and retrieved correctly	High	GameSnapshot Tests	3	Passed
1.09	testGetName	Verifies that the name stored and retrieved for the NPC is correct	Low	NPCUnitTests	3	Passed
1.10	testPersonality	Verifies that the personality stored and retrieved for the NPC is correct	Medium	NPCUnitTests	3	Passed

1.11	testInteractFindingClues	Verifies that the correct clue is correctly collected when interacted, also correctly altering the score	High	PlayerUnitTests	5.1.2	Passed
1.12	testPlayerName	Verifies that the playerName is stored and returned correctly	Low	PlayerUnitTests	2.1.5	Passed
1.13	testPlayerPersonality	Verifies that the players personality can be manipulated and stored correctly	Medium	PlayerUnitTests	2.1.1	Passed
1.14	doesPlayerMove	Verifies that the player is able to move correctly in all four cardinal directions	High	PlayerUnitTests	2.1.4	Passed
1.15	testCanAccuse	Verifies that the player is not able to accuse without evidence	Low	PlayerUnitTests	7.1.4	Passed
1.16	testScore	Verifies that the players score can be modified correctly	Medium	PlayerUnitTests	6.1.1	Passed
1.17	testPlayTime	Verifies that how long a player has played for is stored correctly	Low	PlayerUnitTests	6.1.2	Passed
1.18	testGetTransition	Verifies that the player transitions between rooms correctly	High	RoomUnitTests	2.1.4	Passed
1.19	testAddTransition	Verifies that new transitions are added correctly	High	RoomUnitTests	2.1.4	Passed
1.20	testWalkable	Verifies certain tiles are and aren’t walkable	Medium	RoomUnitTests	2.1.4	Passed
1.21	testTrigger	Verifies if a tile a trigger tile or not	High	RoomUnitTests	2.1.4	Passed
1.22	testMatRotation	Verifies that mats are rotated to the correct direction	Low	RoomUnitTests	2.14	Passed
1.23	testGenerateGame	Verifies that the scenario builder runs without generating an exception	High	ScenarioBuilder UnitTest	1	Passed