Validation Testing

Last edited by **Priya Ashok Sardhara** 1 day ago

Validation Testing

Validation testing ensures that the software **fulfills the functional and non-functional requirements** set out in the original project specification and team-defined requirements documentation. Each test case below is tied directly to a project requirement and is designed to confirm that the implemented feature behaves correctly during real usage.

Test Case 1 : Pet Stats Decay Over Time

Test Case Name	**ValidateStatDecay**
Test Case Description	Ensure that a pet's fullness, sleep, and happiness stats decrease over time while the game is running
Test Steps	1. Launch the game and select a new pet 2. Leave the game running without using any commands for 2–3 minutes 3. Observe stat changes in the sidebar
Pre-Requisites	Game must be running; player must have selected a pet
Expected Results	Fullness, sleep, and happiness values decrease at regular intervals
Test Category	Validation Test
Requirement	3.1.6 (Vital Statistics)
Automation	No (Manual)
Date Run	March 31, 2025
Pass/Fail	Pass
Test Results	All three stats decreased after idle time
Remarks	Stat decay rates matched the expected values defined for the selected pet type

Test Case 2 : Pet State Changes Trigger Correct Sprites

Test Case Name	**ValidateSpriteChange**
Test Case Description	Verify that the pet's sprite image updates when its state changes (e.g., hungry, angry, sleeping)
Test Steps	1. Reduce a stat below a threshold (e.g., hunger < 30%) 2. Observe the pet's displayed image
Pre-Requisites	Game must be running; stat levels must be manually reduced or allowed to decay
Expected Results	Pet sprite changes to the correct image (e.g., hungry sprite)
Test Category	Validation Test
Requirement	3.1.10 (Pet Sprite)
Automation	No

Date Run	March 31, 2025
Pass/Fail	Pass
Test Results	Hungry, angry, and sleeping sprites all displayed correctly during test
Remarks	Sound effects also matched pet state

Test Case 3: Save and Load Pet State

Test Case Name	**ValidateSaveLoad**
Test Case Description	Ensure that the save/load system stores and restores the full game state
Test Steps	1. Play the game and modify stats or inventory 2. Save the game 3. Exit and restart the application 4. Load the saved game
Pre-Requisites	Must have an available save slot
Expected Results	Pet state, stats, inventory, and score match saved values
Test Category	Validation Test
Requirement	3.1.5 (Save/Load Game State)
Automation	No
Date Run	March 31, 2025
Pass/Fail	Pass
Test Results	Pet state and inventory restored correctly
Remarks	Tested across multiple save slots for completeness

Test Case 4: Command Availability Based on Pet State

Test Case Name	**ValidateCommandLocking**
Test Case Description	Ensure commands are disabled when the pet state restricts them (e.g., Dead or Sleeping)
Test Steps	1. Reduce health to 0 (pet dies) 2. Attempt to click a command button (e.g., Feed or Play) 3. Repeat for Sleeping state
Pre-Requisites	Pet state must be Dead or Sleeping
Expected Results	Command buttons are visually disabled and non-functional
Test Category	Validation Test
Requirement	3.1.7 (Command Restrictions)

Automation	No
Date Run	March 31, 2025
Pass/Fail	Pass
Test Results	Buttons were grayed out and non-clickable
Remarks	Visual feedback matches wireframe requirements

Test Case 5: Parental Control Restriction Enforced

Test Case Name	**ValidateParentalRestriction**
Test Case Description	Ensure the game enforces the configured playtime window
Test Steps	1. Launch the game outside of allowed hours 2. Attempt to start a new game or load an existing game
Pre-Requisites	Parental control must be configured to restrict access during the current time
Expected Results	Access is denied and a message is shown
Test Category	Validation Test
Requirement	3.1.11 (Parental Controls)
Automation	No
Date Run	March 31, 2025
Pass/Fail	Pass
Test Results	Game access blocked as expected with proper error message
Remarks	Verified restriction for both new and saved games

Test Case 6 : Stat Increases After Command Use

Test Case Name	**ValidateStatIncrease_CommandUse**
Test Case Description	Ensure that using commands like "Feed" or "Play" increases the corresponding stat (Fullness or Happiness)
Test Steps	1. Launch the game and select a pet 2. Note the current value of the "Hunger" or "Happiness" stat 3. Click the corresponding command (e.g., “Feed” or “Play”) 4. Observe any changes to the stat value
Pre-Requisites	Pet must be in a state where the command is available (not dead or sleeping)
Expected Results	The stat increases by the expected amount after the command is used
Test Category	Validation Test

Requirement	3.1.7 (Commands)
Automation	No
Date Run	March 31, 2025
Pass/Fail	Pass
Test Results	Stat increased as expected based on pet type
Remarks	Effect confirmed through sidebar stat change

Test Case 7 : Inventory Item Count Decreases When Used

Test Case Name	**ValidateInventoryUse**
Test Case Description	Verify that the number of items in inventory decreases when a command uses one (e.g., food or gift).
Test Steps	1. Launch the game and select a pet 2. Check inventory quantity for a specific item (e.g., food) 3. Use the related command (e.g., “Feed”) 4. Check the inventory again
Pre-Requisites	Inventory must contain at least one usable item
Expected Results	Inventory item count decreases by 1
Test Category	Validation Test
Requirement	3.1.8 (Inventory System)
Automation	No
Date Run	March 31, 2025
Pass/Fail	Pass
Test Results	Item quantity updated correctly
Remarks	No duplicate updates; item not used if command blocked

Test Case 8 : Game Over Triggers on Health = 0

Test Case Name	**ValidateParentalRestriction**
Test Case Description	Ensure that when the pet’s health reaches 0, the Game Over screen is triggered
Test Steps	1. Allow the pet’s stats to decay naturally until health reaches 0 2. Observe if the Game Over screen is displayed
Pre-Requisites	Pet stats must be sufficiently low to reduce health over time
Expected Results	Game transitions to the Game Over screen with relevant messaging

Test Category	Validation Test
Requirement	3.1.6 (Vital Stats), 3.1.7 (Commands), 3.1.11 (State Changes)
Automation	No
Date Run	March 31, 2025
Pass/Fail	Pass
Test Results	Game Over screen was triggered and commands were locked
Remarks	Proper UI feedback and sprite displayed

Test Case 9 : Sound Feedback Plays on Interaction

Test Case Name	**ValidateSoundFeedback**
Test Case Description	Verify that sound effects play correctly when using pet interaction commands
Test Steps	1. Launch the game 2. Click any command (Feed, Play, Exercise, etc.) 3. Listen for sound output
Pre-Requisites	Sound files must exist and be mapped to commands
Expected Results	Correct sound plays with corresponding action
Test Category	Validation Test
Requirement	3.1.13 (Sound)
Automation	No
Date Run	March 31, 2025
Pass/Fail	Pass
Test Results	All expected sound effects played on command use
Remarks	Volume consistent; fallback to silent state handled

Test Case 10 : Score Increases with Positive Actions

Test Case Name	**ValidateScoreSystem**
Test Case Description	Ensure that the score increases when a valid command is executed successfully
Test Steps	1. Observe the initial score 2. Use a valid command (e.g., Feed) 3. Check for score increment
Pre-Requisites	Command must be available, and pet must not be dead or sleeping

Expected Results	Score increases according to the action
Test Category	Validation Test
Requirement	3.1.9 (Scoring)
Automation	No
Date Run	March 31, 2025
Pass/Fail	Pass
Test Results	Score updated by correct amount.
Remarks	Points vary per command and pet state

Test Case 11 : Save Confirmation Message Displays

Test Case Name	**ValidateSaveFeedback**
Test Case Description	Ensure that a save confirmation message is displayed when the user saves the game
Test Steps	1. Click the “Save” button 2. Observe the UI for feedback message or confirmation box
Pre-Requisites	Save feature must be implemented and accessible
Expected Results	A message confirms the game was saved
Test Category	Validation Test
Requirement	3.1.5 (Save/Load Game)
Automation	No
Date Run	March 31, 2025
Pass/Fail	Pass
Test Results	“Game Saved” message appeared briefly
Remarks	Message disappears after a few seconds; also appears after autosave

Test Case 12 : UI Loads Correctly from JSON

Test Case Name	**ValidateJSONLayoutLoading**
Test Case Description	Ensure that UI screens load correctly based on JSON layout files
Test Steps	1. Launch the game 2. Navigate through screens (Main Menu, GamePlay, Save, Tutorial) 3. Confirm layout is consistent with wireframes
Pre-Requisites	JSON layout files must be present in correct directory

Expected Results	UI renders as designed; no missing components
Test Category	Validation Test
Requirement	3.1.2–3.1.4 (UI Screens)
Automation	No
Date Run	March 31, 2025
Pass/Fail	Pass
Test Results	Screens match wireframe layout (not fully)
Remarks	Few UI crashes and missing images

Test Case 13 : Revival from Parental Controls

Test Case Name	**ValidatePetRevival**
Test Case Description	Ensure that a pet can be revived via the Parental Controls screen
Test Steps	1. Allow the pet to die (health reaches 0) 2. Navigate to Parental Controls 3. Click the “Revive Pet” button 4. Return to the game screen
Pre-Requisites	Parental controls must be enabled and accessible
Expected Results	Pet is brought back to life; stats reset or restored
Test Category	Validation Test
Requirement	3.1.11 (Parental Controls)
Automation	No
Date Run	March 31, 2025
Pass/Fail	Pass
Test Results	Pet revived and gameplay resumed
Remarks	Revival limited to once per save slot

Test Case 14 : Tutorial Screens Show in Order

Test Case Name	**ValidateTutorialNavigation**
Test Case Description	Ensure that all 8 tutorial screens display in proper sequence and are navigable
Test Steps	1. Open the Tutorial screen from the Main Menu 2. Use “Next” and “Back” buttons to navigate all 8 screens

Pre-Requisites	Tutorial image and JSON files must be present
Expected Results	All screens load correctly in sequence; buttons function properly
Test Category	Validation Test
Requirement	3.1.3 (Tutorial)
Automation	No
Date Run	March 31, 2025
Pass/Fail	Pass
Test Results	All 8 tutorial screens navigated successfully
Remarks	Screen transitions were smooth; no lag or crashes

Test Case 15 : Blocked Command Does Not Play Sound

Test Case Name	ValidateBlockedCommandSoundDoesNotPlay
Test Case Description	Ensure no sound plays when an unavailable (disabled) command button is brought to the left end of bar
Test Steps	1. Let the pet enter a state (e.g., Sleeping or Dead) that disables commands 2. Attempt to click a disabled command button (e.g., Feed) 3. Observe any sound output
Pre-Requisites	Pet must be in a state where commands are disabled
Expected Results	No sound plays when the command is disabled
Test Category	Validation Test
Requirement	3.1.7 (commands), 3.1.10 (Pet Sprite)
Automation	No
Date Run	March 31, 2025
Pass/Fail	Pass
Test Results	No audio was triggered on disabled commands
Remarks	Feedback remained silent as expected

Test Case 16 : Invalid Pin Entry Shows Error

Test Case Name	ValidateInvalidPINEntry
Test Case Description	Ensure the system displays an error when an incorrect PIN is entered in the Parental Controls screen
Test Steps	1. Navigate to the Parental Controls screen 2. Enter an incorrect PIN (e.g., 0000)

Pre-Requisites	Parental Controls must be enabled
Expected Results	An error message is displayed indicating invalid PIN
Test Category	Validation Test
Requirement	3.1.11 (Parental Controls)
Automation	No
Date Run	March 31, 2025
Pass/Fail	Pass
Test Results	Error message appeared immediately
Remarks	Access was blocked as intended

Test Case 17 : All Commands Blocked During Sleep

Test Case Name	ValidateMultipleCommandsBlockedDuringSleep
Test Case Description	Ensure all interactive commands are disabled when the pet is sleeping
Test Steps	1. Use the Sleep command to put the pet to sleep 2. Try to click other commands like Feed, Play, etc
Pre-Requisites	Pet must be sleeping
Expected Results	All commands except 'Wake Up' are disabled.
Test Category	Validation Test
Requirement	3.1.6 (Vital Statistics and Rules), 3.1.7 (Commands)
Automation	No
Date Run	March 31, 2025
Pass/Fail	Pass
Test Results	Buttons were disabled during sleep state
Remarks	Game behaved correctly according to rules

Test Case 18 : Inventory Limit Handling

Test Case Name	ValidateInventoryLimitHandling
Test Case Description	Ensure the inventory UI handles full or zero-count inventory states gracefully
Test Steps	1. Use all items in inventory until count is 0 2. Try to use a command requiring an item 3. Observe UI behaviour

Pre-Requisites	Inventory must reach 0 items
Expected Results	UI prevents use and displays warning
Test Category	Validation Test
Requirement	3.1.8 (Inventory System)
Automation	No
Date Run	March 31, 2025
Pass/Fail	Pass
Test Results	Use was blocked; visual warning shown
Remarks	No item underflow occurred

Test Case 19 : Sound Toggle Works Correctly

Test Case Name	ValidateSoundToggleFunctionality
Test Case Description	Ensure that enabling/disabling sound in settings toggles all audio output
Test Steps	1. Go to Settings screen 2. Disable sound 3. Trigger a command or background event 4. Re-enable sound and repeat
Pre-Requisites	Sound must be functional
Expected Results	No sound when disabled; normal audio resumes when enabled
Test Category	Validation Test
Requirement	3.1.12 (Housekeeping and Error Handling)
Automation	No
Date Run	March 31, 2025
Pass/Fail	Pass
Test Results	Toggle worked correctly
Remarks	Sound responded immediately to toggle

Test Case 20 : Start Overflow and Underflow Prevention

Test Case Name	ValidateStateOverflowPrevention
Test Case Description	Ensure stats cannot exceed 100 or drop below 0 due to rapid input or save/load edge cases

Test Steps	1. Use commands repeatedly or load a manipulated save file 2. Observe stat values
Pre-Requisites	Access to stat boundaries or save files
Expected Results	Stat values stay within 0 to 100
Test Category	Validation Test
Requirement	3.1.6 (Vital Statistics and Rules)
Automation	No
Date Run	March 31, 2025
Pass/Fail	Pass
Test Results	No overflow or underflow occurred
Remarks	Game safely clamped all values

Test Case 21 : UI Responsiveness Under Load

Test Case Name	ValidateUIResponsivenessUnderLoad
Test Case Description	Ensure that the UI remains responsive under heavy or rapid user input
Test Steps	1. Rapidly click buttons and navigate screens 2. Monitor any freezing or lag
Pre-Requisites	Run on standard Windows system
Expected Results	No UI lag or crashes observed
Test Category	Validation Test
Requirement	3.2 (Non-functional)
Automation	No
Date Run	March 31, 2025
Pass/Fail	Pass
Test Results	UI remained smooth and usable
Remarks	No rendering or responsiveness issues

Test Case 22 : Exiting Tutorial Returns to Main Menu

Test Case Name	ValidateTutorialExitReturnsToMainMenu
Test Case Description	Ensure that exiting the tutorial screen returns the user to the main menu

Test Steps	1. Open the tutorial from the main menu 2. Click 'Exit' on any tutorial screen
Pre-Requisites	Tutorial JSON and images must be loaded
Expected Results	User is redirected back to Main Menu screen
Test Category	Validation Test
Requirement	3.1.3 (Instructions and Tutorial)
Automation	No
Date Run	March 31, 2025
Pass/Fail	Pass
Test Results	Exit button worked as expected
Remarks	No navigation errors or screen glitches