Basic Testing of Use Cases

Identify all the tests in your Validation Testing Strategy - this is a set of test cases based on your Use Cases that you will apply at the end of Gizmo ball development to demonstrate that it meets the original requirements. These should be described in a style that a third party can understand and perform on your system. For example, it will include a set of tests to demonstrate that "Add Gizmo" functions correctly.

Use Case: Start  
  
Check if player’s click on button “Start” triggers the code tied to it. Board should then change from build mode into play mode (maybe have some super basic variable for checking board mode). Check if the game board is now displaying in the GUI. Scan through GUI and ensure that gizmos and ball are displayed in the correct position (specified through pre-set board OR build mode map).

Use Case: Stop  
  
Check if player’s click on button “Stop” triggers code tied to it. Board should be in run mode AND running, for the STOP command to continue so check that the board has met these conditions. After these conditions have been met, the gameplay should halt (check that the board is in run mode BUT NOT running). Gameplay should have stopped and key presses should not affect the state of the game.

**RUN MODE**  
Use Case: Load Model (from save // game in progress)  
  
Check if player’s click on button “Load Model” triggers code tied to it. Board should be in run mode in order to continue with loading a previous save. Check if file selected is in the correct format (correct file extension). Saved game state should appear in the GUI. User should be able to continue from saved state (so maybe change Start > Continue after a map has been loaded).

Use Case: Reload Model  
  
Check if player’s click on button “Reload Model” triggers code tied to it. Current map should then be “refreshed” (map should reload from pre-built map or user made map). Check if the correct map has been loaded (check gizmo locations and compare to saved gizmo locations).

Use Case: Save Model  
  
Check if player’s click on button “Save Model” triggers code tied to it. User should be in build mode and at least 4 gizmo’s (some progress) should have been placed in build mode (just some basic counter to keep track?) in order for this code to continue. After file is saved check if file exits and if it does continue, if not ERROR.

Use Case: Build Game-play  
  
Check if player’s click on button “Build Mode” triggers code tied to it. Board should then change from Run mode into Build mode.

Use Case: Quit  
  
Check if player’s click on button “Quit” triggers code tied to it. Check game has stopped and program closes (print something in console).

Use Case: Add Gizmo  
  
Check if player’s click on button “Add Gizmo” triggers code tied to it. (Button only available in build mode so no need to check if in build mode?). Check if a gizmo is added to the board in the correct position (where the user clicked).

**Use Case: Delete Object**Check if player’s click on button “Delete Object” triggers code tied to it.  
Used to enter a sort of “delete mode”. User clicks button then clicks on an object they want to delete. Exits the “delete mode” after one deletion to prevent accidental deletes. Check if there is something on the board to delete (check if not empty). Targeted object should be deleted, check that there is no object in the position of where the user clicked.

Use Case: Add Absorber  
  
Check if player’s click on button “Add Absorber” triggers code tied to it. Absorber should be added where the user specified (check where user clicked and that the object added is an absorber). Also check if there is already an object in the space clicked, if there is, do not add an absorber.

Use Case: Add Flipper  
  
Check if player’s click on button “Add Flipper” triggers code tied to it. Flipper should be added where the user specified (check where user clicked and that the object added is a flipper). Also check if there is already an object in the space clicked, if there is, do not add a flipper.

Use Case: Key Connect  
  
Check if player’s click on button “Bind Key” triggers code tied to it. User then clicks on a flipper or absorber and a window will pop up. Check if the object clicked is valid. User then enters a VALID key to associate with the object. Then make sure that the key is bound to the correct flipper or absorber. Key bind that has just been made by the user should then be added to the current game.

Use Case: Key Disconnect  
  
Check if player’s click on button “Unbind Key” triggers code tied to it. User clicks on an object and the program should unbind any key associated with it. Check that the object now has 0 keys bound to it.

Use Case: Add Ball  
  
Check if player’s click on button “Add Ball” triggers code tied to it. Ball should be added where the user specified (check where user clicked and that the object added is a ball). Also check if there is already an object in the space clicked, if there is, do not add an absorber. **Ensure that there are no other balls on the board, if there are, delete the balls and the check again to make sure board has no balls. (Do this on button click?)**

Use Case: Rotate  
  
Check if player’s click on button “Rotate” triggers code tied to it. User will then click on an object. Check if object is there, if yes, continue. Code should then rotate the object 90 degrees. Check that the object has been rotated (basic variable that is updated after rotate?).

Use Case: Clear Board  
  
Check if player’s click on button “Clear Board” triggers code tied to it. Runs the code. Check that there is nothing on the board after clear has been run. **(Only available in run mode?)**

Use Case: Move  
  
Check if player’s click on button “Move” triggers code tied to it. Check that board is not empty. User clicks on an object, then on the preferred location of where they would like to move it. Check that the user has clicked on a valid object (not empty). After object has been moved check that it is in the correct position by comparing mouse click location to current object position.

Use Case: Connect  
  
Check if player’s click on button “Connect gizmo” triggers code tied to it. Check that board is not empty. User clicks on a gizmo and then another, these are now connected. Check that the 2 gizmos are connected / linked together. Exit Connect mode after 1 link is created to minimize accidental links.

Use Case: Disconnect  
  
Check if player’s click on button “Disconnect gizmo” triggers code tied to it. Check that board is not empty and that a current link / connection between 2 gizmos exists. User clicks on a linked gizmo and then the associated gizmo, these are no longer connected. Check that the 2 gizmos are no longer connected / linked together. Exit disconnect mode after 1 link is been deleted to minimize accidental disconnects.

Use Case: Change Friction   
  
Check if slider “Change Friction” triggers code tied to it. User chooses value on a slider. Check that the number displayed on the GUI is the same as the value for friction.

Use Case: Change Gravity   
  
Check if slider “Change Gravity” triggers code tied to it. User chooses value on a slider. Check that the number displayed on the GUI is the same as the value for gravity.