**Step 5: Test and Refine the Solution (Debug and Verify)**

|  |  |  |  |
| --- | --- | --- | --- |
| **Sample Scenario** | **Input** | **Expected Output** | **Logic** |
| Food bin is empty | Bin\_full = 0 | Red\_light is on | Ok |
| Feeding time | Bin\_full = 1 **and** Bowl = 0 Schedule\_time = Real-time | Servo ON for Feeding\_time seconds | Ok |
| Pet eats as expected | Bin\_full = 1 **and** Bowl = 0 Schedule\_time ≠ Real-time | No Action  Until next feeding time | Ok |
| Pet does not eat food (Bin is full **and**  Bowl is full) | Bin\_full = 1 **and** Bowl = 1 Schedule\_time + 15min  >  Real-time | Red\_light is on | Ok |

As the table shows, the logic turns on the warning light whenever the food bin is empty or the pet doesn't eat. If the pet eats as it should, the system takes no action until the next feeding time.

There is room for improvement by using load cells instead of micro switches under the food bin and feeding bowl. This will allow for improvements in the code that can make the control more precise and customized by the type of animal. Furthermore, it will also be possible to track the animal's feeding history, which can be used to compare its feeding habits over time.

**GitHub repository link**

The link for my repository is show bellow

<https://github.com/KennySousa224/pet-feeder-project>

**References**

* Lipovski, G. J. (1999). *Introduction to microcontrollers: Architecture, programming, and interfacing of the Motorola 6812*. Academic Press.