|  |  |
| --- | --- |
| **Test ID#** | 1 |
| **Test level** | System (1 of 5) |
| **Quality criterion / attribute** | External and Internal Quality: Reliability |
| **Description of test** | A new user attempts to register a user that already exists |
| **Requirements reference** | Usability |
| **Steps of the test case** | 1. A new user runs the install.py script to begin installation. 2. A test enters ‘testUser’, a username known to exist as the user name they would like to register. 3. A user enters any matching password twice to advance registration. |
| **Expected outcome** | The user receives a message stating that the user name they requested already exists and the user is allowed to start the registration process over again. |
| **Actual test outcome** | Expected behavior happens. |
| **Pass/Fail** | Pass |
| **Defect(s)** | Standardized failure messages do not exist. |

|  |  |
| --- | --- |
| **Test ID#** | 2 |
| **Test level** | System (2 of 5) |
| **Quality criterion / attribute** | External and Internal Quality: Portability |
| **Description of test** | A new user is able to complete install and setup on a Raspberry Zero instead of a Raspberry Pi B+ |
| **Requirements reference** | Usability |
| **Steps of the test case** | 1. A new user downloads and extracts the Smart Gardens piDir zip file to their home directory. 2. The user runs the install.py script using the following command: *sudo python setup.py* 3. The user completes the user registration 4. The user completes Garden and sensor setup 5. The user completes the job scheduler setup |
| **Expected outcome** | The user will be able to complete the install of the Smart Gardens software regardless of the Pi Model (A, B, 2, 3, or Zero) so long as their OS is Raspbian. |
| **Actual test outcome** | Installation on Raspberry Pi Zero succeeded. |
| **Pass/Fail** | Pass |
| **Defect(s)** | N/A |

|  |  |
| --- | --- |
| **Test ID#** | 3 |
| **Test level** | User |
| **Quality criterion / attribute** | Quality in Use: Efficiency |
| **Description of test** | A user will be able to leave the SmartGarden system unmonitored for at least two weeks without having the intervene, restart the software or hardware, and have daily uploads completed without fail |
| **Requirements reference** | Usability |
| **Steps of the test case** | 1. Complete the setup process in a stable location with a constant power source and good wifi. 2. Register a test account, setup a garden with sensors, and schedule twice a day readings. 3. Leave the system running without intervention for 2 weeks 4. Check that readings were taken each day. |
| **Expected outcome** | There will be 28 garden readings over the 14-day period with no user interaction. |
| **Actual test outcome** | All readings were taken at the appropriate times. |
| **Pass/Fail** | Pass |
| **Defect(s)** | N/A |

|  |  |
| --- | --- |
| **Test ID#** | 4 |
| **Test level** | Integration |
| **Quality criterion / attribute** | External and Internal Quality: Functionality |
| **Description of test** | Configuration settings can be downloaded remotely |
| **Requirements reference** | Usability |
| **Steps of the test case** | 1. An administrator changes the configText field in the login table. 2. An administrator changes the configTime field to reflect the time changed. 3. The user’s system attempts to take a reading 4. System compares remote configTime and downloads new configText 5. System stores the new configuration as config.json 6. System loads the new configuration and takes the readings with the new configuration. |
| **Expected outcome** | The system will update the config.json and display a line that a new configuration has been found. |
| **Actual test outcome** | The new configuration was detected, the config.json was updated, and the display line was printed. |
| **Pass/Fail** | Pass |
| **Defect(s)** | N/A |

|  |  |
| --- | --- |
| **Test ID#** | 5 |
| **Test level** | Unit |
| **Quality criterion / attribute** | External and Internal Quality: Functionality |
| **Description of test** | The system will be able to support more than one Garden as part of a single configuration. |
| **Requirements reference** | Scalability |
| **Steps of the test case** | 1. The user will add a complete garden with at least one of each sensor. 2. The user will then add a second garden with at least one of each sensor. 3. The user will run the takeReadings job. |
| **Expected outcome** | The readings for both gardens will be completed with no errors. |
| **Actual test outcome** | Both gardens were added and worked normally |
| **Pass/Fail** | Pass |
| **Defect(s)** | N/A |