|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Black Box Testing Suite** | | | | | |
| **Test Type** | Random Testing and Equivalence Partitioning | | | | |
| **Test Description** | Using the user interface for the years functionality | | | | |
| **Tester Name** | Neth Botheju | | | | |
| **Used Operating System** | MacOS Monterey version 12.2 | | | | |
| **Test Cases** | | | | | |
| **Case No.** | **Details** | **Input** | **Output** | **Expected Output** | **Pass or Fail** |
| **3.1** | Attempts to click the << and >> | Mouse click on  << and then the >> twice | Display updated with new dates and month box increased by 1 increment | Display should update with new dates and year box should’ve increased by 1 increment | Fail |
| **3.2** | Attempt to manually enter a year value into the drop down box | Click drop down box and type = {2002, 2022, 1949, 2051} | Cannot get cursor to select the year to manually change | The box should appear with the insertion point to type the value | Fail |
| **3.3.1** | Attempt to select a valid input on the boundary of the requirement | Year = 1950 | System does not have the year in the drop down box as an option to select | The system should update to the selected year and display the month | Fail |
| **3.3.2** | Attempt to select a valid input within the boundary | Year = 2022 | System allows user to select the year and updates the display accordingly | The system should have the year as an option and update to the selected year and display the month | Passed |
| **3.3.3** | Attempt to select an invalid year outside of the boundary | Year = 1951 | The year is not an option in the drop down box to select | The system should not display the year in its drop down box | Passed |
| **Rationale** | | | | | |
| **Case No.** | **Why this black box strategy was used and justified.** | | | | |
| **3.1** | Random testing was the chosen strategy for this particular suite because of the various inputs needed for this requirement. As there are two buttons that update the display, both were tested at the same time which minimised the amount of test cases that needed to be done. This test was most suitable as just because one button works, doesn’t guarantee the other one works as intended hence both should be checked.  **Input Domain**  Clicking of intended buttons using the user’s cursor  **Test inputs**  Inputs = {clicking the << and >> buttons}  **How the system executes**  System will  refresh the UI display with the relevant information depending on which arrow button was clicked  **Results**  The system should have moved to the following year and updated the year box but instead it navigates through the months. This appeals to user cognitive thinking and strives for consistency as most other calendar based applications allow the user to navigate through the months in a similar format.  Hence this is incorrect results in terms of the requirements but should be correct according to user interface standards and ease of use. | | | | |
| **3.2** | The specification stating just because one input works doesn’t guarantee they all work also apply for this testing suite as well. Hence the use of random testing which inputs random values both within and out of the bounds given which test the ability for the system to throw an error as well. Since this test only checks the requirement that the dropdown box can be typed into, there is no need to partition inputs with separate outputs since it only checks the functionality.  **Input Domain**  Any integer between 1900 and 2100  **Test inputs**  Inputs = {clicking on the drop down box and manually enter 2002}  **How the system executes**  The system should update the display and show the specific month of that manually entered year.  **Results**  The system does not allow the user to manually enter a value into the drop down box in the tested Operating System of MacOS. This test expectedly passes on a Windows OS but the requirement entails that the program should run appropriately on Windows, MacOS, Linux and all other similar operating systems hence fails this requirement. | | | | |
| **3.3** | Equivalence Partitioning was used for this particular suite to minimise the amount of testing that needs to be done to check the valid years. This also allows testing to see if the system will throw errors when invalid years are attempted to be selected. It makes sure to cover all possible outcomes and input partitions so that the testing is thorough with checking all viable options the user can select.  **Input Variable: Years**  EC1: any year between [1950, 2050] - checks all the valid years that the system should allow to show up on the drop down  EC2: any year [1950 > or 2050 <] - checks invalid years that the system should not let the user select  **Output Variable: Update System Display**  EC3: updates system - the program should display the chosen month of the year that is selected from the drop down  EC4: does not allow to select - the year should not be displayed as an option  **Output Variable: Error**  EC5: None  EC6: Invalid input or Error sound (MacOS)  **Test Cases:**  TC 3.3.1: Year = 1950, Output = updates system, error = None  TC 3.3.2: Year = 2022, Output = updates system, error = None  TC 3.3.3: Year = 1951, Output = does not allow to select, error = Error sound  The system displayed the month of the selected year from the drop down box for inputs that were able to be selected but as explored in TC 1.2, some values despite being in the requirement range does not appear to be selected in the drop down box. | | | | |

|  |  |  |
| --- | --- | --- |
| Test case | Output | Comment |
| 3.1 | Original:    After execution: | The month moved from 8 to 9 on the drop down box instead of the years |
| 3.2 |  | Drop down box doesn’t allow to type |
| 3.3.1 |  | The drop down box does not go to any year before 2000 |
| 3.3.2 |  | The input year 2022 is available to select |
| 3.3.3 |  | The selection does not display an option beyond the 2050 boundary |