|  |  |
| --- | --- |
| **Project Name: Project 1: Voting System Team#22** | |
| **Test Stage: Unit \_**✔**\_ System \_\_** | **Test Date: 11/18/19** |
| **Test Case ID#: VSC\_001** | **Name(s) of Testers: Berni D** |
| **Test Description:**   * Location: testing/voting\_system\_controller\_unittest.cc * Name: TEST\_F(VotingSystemControllerTest, InputFileName) Test: the function InputFileName return true if input is valid, return false if invalid.   **Run the test:**  UncommentTEST\_F(VotingSystemControllerTest, InputFileName) functions in voting\_system\_controller\_unittest.cc  Comment out other test if you want run only this test  **Terminal at:** testing/ $ make  **Terminal at:** testing/build/bin $ ./unittest   * ***Check terminal to see test pass or fail***   *Notes:*   * *copy opl.csv and cpl.csv from Project1/src/ => Project1/testing/build/bin if don’t have.* * *Make sure this line is uncommented in testing/makefile*   DEFINES += -DVOTING\_SYSTEM\_CONTROLLER\_TESTS |  |
| **Automated: yes\_\_\_ no \_**✔**\_\_** | **Indicate where are you storing the tests (what file) and the name of the method/functions being used.** |
| **Results: Pass \_\_\_**✔**\_\_ Fail\_\_\_\_\_\_\_\_** |  |
|  |  |
| **Preconditions for Test:** voting\_system\_controller is constructed. | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Step**  **#** | **Test Step**  **Description** | **Test**  **Data** | **Expected**  **Result** | **Actual**  **Result** | **Notes** |
| 1 | Run ./unittest at build/bin | Manually Enter opl.csv | true | true |  |
| 2 | Run voting\_system\_controller->InputFileName() | "" | true | true |  |
| 3 | Check result == true | Manually Enter opl.csv | true | true |  |
| 4 | Run voting\_system\_controller->InputFileName() | Manually Enter “hello”, then enter “Y”, then enter “opl.csv” | true | true |  |
| 5 | Check result == true |  |  |  |  |
| 6 | Run voting\_system\_controller->InputFileName() | Manually Enter “hello”, then enter “N”. | false | false |  |
| 7 | Check result == false |  | false | false |  |
| 8 |  |  |  |  |  |

**Post condition(s) for Test:**

file\_name is valid

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

|  |  |
| --- | --- |
| **Test Stage: Unit \_\_ System \_**✔**\_** | **Test Date: 11/18/19** |
| **Test Case ID#: VSC\_002** | **Name(s) of Testers: Berni D** |
| **Test Description:**   * Location: testing/voting\_system\_controller\_unittest.cc * Name: TEST\_F(VotingSystemControllerTest, Run) * Test: the running function of voting system controller (Run function is called in Main function)   **Run the test:**   * UncommentTEST\_F(VotingSystemControllerTest, Run)   functions in voting\_system\_controller\_unittest.cc.   * Comment out other tests if you want to run only this test.   **Terminal at:** testing/ $ make  **Terminal at:** testing/build/bin $ ./unittest   * ***Check terminal message to see test pass or fail***   *Notes:*   * *copy opl.csv and cpl.csv from Project1/testing / => Project1/testing/build/bin if don’t have.* * *Make sure this line is uncommented in testing/makefile*   DEFINES += -DVOTING\_SYSTEM\_CONTROLLER\_TESTS |  |
| **Automated: yes\_\_\_ no \_**✔**\_\_** | **Indicate where are you storing the tests (what file) and the name of the method/functions being used.** |
| **Results: Pass \_\_\_**✔**\_\_ Fail\_\_\_\_\_\_\_\_** |  |
|  |  |
| **Preconditions for Test:** voting\_system\_controller is constructed. | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Step**  **#** | **Test Step**  **Description** | **Test**  **Data** | **Expected**  **Result** | **Actual**  **Result** | **Notes** |
| 1 | Run ./unittest at testing/build/bin |  |  |  |  |
| 2 | OPL Run | Manually Enter opl.csv | audit\_OPL\_0.txt exists  share\_public\_OPL\_0.txt exists | audit\_OPL\_0.txt exists  share\_public\_OPL\_0.txt exists |  |
| 3 | CPL Run | Manually Enter cpl.csv | audit\_CPL\_0.txt exists  share\_public\_CPL\_0.txt exists | audit\_CPL\_0.txt exists  share\_public\_CPL\_0.txt exists |  |
| 4 | Select Exit option | Manually Enter option 2 | Program exit | Program exit |  |
| 5 |  |  |  |  |  |
| 6 |  |  |  |  |  |
| 7 |  |  |  |  |  |
| 8 |  |  |  |  |  |

**Post condition(s) for Test:**

There is no error occur.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

|  |  |
| --- | --- |
| **Test Stage: Unit \_\_ System \_**✔**\_** | **Test Date: 11/18/19** |
| **Test Case ID#: VSC\_003** | **Name(s) of Testers: Berni D** |
| **Test Description:**   * Location: testing/voting\_system\_controller\_unittest.cc * Name: TEST\_F(VotingSystemControllerTest, RunAuditProcess) * Test: Running audit process correspond to each voting type, OPL or CPL   **Run the test:**   * UncommentTEST\_F(VotingSystemControllerTest, RunAuditProcess) functions in voting\_system\_controller\_unittest.cc. * Comment out other tests if you want to run only this test.   **Terminal at:** testing/ $ make  **Terminal at:** testing/build/bin $ ./unittest   * ***Check terminal message to see test pass or fail***   *Notes:*   * *copy opl.csv and cpl.csv from Project1/testing / => Project1/testing/build/bin if don’t have.* * *Make sure this line is uncommented in testing/makefile*   DEFINES += -DVOTING\_SYSTEM\_CONTROLLER\_TESTS |  |
| **Automated: yes\_**✔**\_\_ no \_✔\_\_ Semimanual** | **Indicate where are you storing the tests (what file) and the name of the method/functions being used.** |
| **Results: Pass \_\_\_**✔**\_\_ Fail\_\_\_\_\_\_\_\_** |  |
|  |  |
| **Preconditions for Test:** voting\_system\_controller is constructed. | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Step**  **#** | **Test Step**  **Description** | **Test**  **Data** | **Expected**  **Result** | **Actual**  **Result** | **Notes** |
| 1 | Run ./unittest at testing/build/bin |  |  |  |  |
| 2 | OPL Run  voting\_system\_controller1->SetFileName("opl.csv"); | "opl.csv" file located in testing/build/bin | audit\_OPL\_0.txt exists  share\_public\_OPL\_0.txt exists | audit\_OPL\_0.txt exists  share\_public\_OPL\_0.txt exists |  |
| 3 | User manually check the display result in the screen |  | 3 winners, first is Pike | 3 winners, first is Pike |  |
| 4 | CPL Run  voting\_system\_controller1->SetFileName("cpl.csv"); | "cpl.csv" file located in testing/build/bin | audit\_CPL\_0.txt exists  share\_public\_CPL\_0.txt exists | audit\_CPL\_0.txt exists  share\_public\_CPL\_0.txt exists |  |
| 5 | User manually check the display result in the screen |  | 7 winners |  |  |
| 6 |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

**Post condition(s) for Test:**

There is no error occur.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

|  |  |
| --- | --- |
| **Test Stage: Unit \_\_ System \_**✔**\_** | **Test Date: 11/18/19** |
| **Test Case ID#: VSC\_004** | **Name(s) of Testers: Berni D** |
| **Test Description:**   * Location: testing/voting\_system\_controller\_unittest.cc * Name: TEST\_F(VotingSystemControllerTest, RunAuditProcess) * Test: Running audit process correspond to each voting type, OPL or CPL WITH TIE BREAKER   **Run the test: (several times to make sure it is random selection)**   * UncommentTEST\_F(VotingSystemControllerTest, RunAuditProcess) functions in voting\_system\_controller\_unittest.cc. * Comment out other tests if you want to run only this test.   **Terminal at:** testing/ $ make  **Terminal at:** testing/build/bin $ ./unittest   * ***Check terminal message to see test pass or fail***   *Notes:*   * *copy opl.csv, opl\_random.csv and cpl.csv from Project1/testing / => Project1/testing/build/bin if don’t have.* * *Make sure this line is uncommented in testing/makefile*   DEFINES += -DVOTING\_SYSTEM\_CONTROLLER\_TESTS |  |
| **Automated: yes\_**✔**\_\_ no \_✔\_\_ Semimanual** | **Indicate where are you storing the tests (what file) and the name of the method/functions being used.** |
| **Results: Pass \_\_\_\_\_ Fail\_\_\_\_✔\_\_\_\_** |  |
|  |  |
| **Preconditions for Test:** voting\_system\_controller is constructed. | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Step**  **#** | **Test Step**  **Description** | **Test**  **Data** | **Expected**  **Result** | **Actual**  **Result** | **Notes** |
| 1 | Run ./unittest at testing/build/bin |  |  |  |  |
| 2 | OPL Run  voting\_system\_controller1->SetFileName("opl\_random.csv"); | "opl\_random.csv" file located in testing/build/bin | audit\_OPL\_0.txt exists  share\_public\_OPL\_0.txt exists | audit\_OPL\_0.txt exists  share\_public\_OPL\_0.txt exists |  |
| 3 | User manually check the display result in the screen |  | 3 winners, first is Pike, second and third shoud be Foster, Borg or Smith | 3 winners, first is Pike, second and third shoud be Foster, Borg or Smith |  |
| 4 | **Run the test: (several times to make sure it is random selection). Repead step 1.** |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

**Post condition(s) for Test:**

There is no error occur.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

|  |  |
| --- | --- |
| **Test Stage: Unit \_**✔**\_ System \_\_** | **Test Date: 11/18/19** |
| **Test Case ID#: OPL\_READ\_001** | **Name(s) of Testers: Berni D** |
| **Test Description:**   * Location: testing/opl\_readfile\_unittest.cc * Name: TEST\_F(OPLReadFileTest, ReadFile) * Test: Reading ballot file and store some value to the report variable   **Run the test:**   * UncommentTEST\_F(OPLReadFileTest, ReadFile) functions in opl\_readfile\_unittest.cc * Comment out other tests if you want to run only this test.   **Terminal at:** testing/ $ make  **Terminal at:** testing/build/bin $ ./unittest   * ***Check terminal message to see test pass or fail***   *Notes:*   * *copy opl.csv and cpl.csv from Project1/testing / => Project1/testing/build/bin if don’t have.* * *Make sure this line is uncommented in testing/makefile*   DEFINES += - DOPL\_READFILE\_TESTS \_TESTS |  |
| **Automated: yes\_**✔**\_\_ no \_\_\_** | **Indicate where are you storing the tests (what file) and the name of the method/functions being used.** |
| **Results: Pass \_\_\_**✔**\_\_ Fail\_\_\_\_\_\_\_\_** |  |
|  |  |
| **Preconditions for Test:** opl object is constructed. | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Step**  **#** | **Test Step**  **Description** | **Test**  **Data** | **Expected**  **Result** | **Actual**  **Result** | **Notes** |
| 1 | Run ./unittest at testing/build/bin |  |  |  |  |
| 2 | opl->ReadFile("opl.csv"); | "opl.csv" file located in testing/build/bin | report\_.GetNumTotalCandidates() = 6  report\_.GetNumBallots() = 9  report\_.GetNumTotalSeats() = 3 | report\_.GetNumTotalCandidates() = 6  report\_.GetNumBallots() = 9  report\_.GetNumTotalSeats() = 3 | There are more value need to be compare in the test. However, report is not consistent yet. Improve the report interface later. |
| 3 |  |  |  |  |  |
| 4 |  |  |  |  |  |
| 5 |  |  |  |  |  |
| 6 |  |  |  |  |  |
| 7 |  |  |  |  |  |
| 8 |  |  |  |  |  |

**Post condition(s) for Test:**

Successfully read the ballot info then store it to the report.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

|  |  |
| --- | --- |
| **Test Stage: Unit \_**✔**\_ System \_\_** | **Test Date: 11/18/19** |
| **Test Case ID#: CPL\_READ\_001** | **Name(s) of Testers: Berni D** |
| **Test Description:**   * Location: testing/cpl\_readfile\_unittest.cc * Name: TEST\_F(CPLReadFileTest, ReadFile) * Test: Reading ballot file and store some value to the report variable   **Run the test:**   * UncommentTEST\_F(CPLReadFileTest, ReadFile) functions in opl\_readfile\_unittest.cc * Comment out other tests if you want to run only this test.   **Terminal at:** testing/ $ make  **Terminal at:** testing/build/bin $ ./unittest   * ***Check terminal message to see test pass or fail***   *Notes:*   * *copy opl.csv and cpl.csv from Project1/testing / => Project1/testing/build/bin if don’t have.* * *Make sure this line is uncommented in testing/makefile*   DEFINES += - DCPL\_READFILE\_TESTS \_TESTS |  |
| **Automated: yes\_**✔**\_\_ no \_\_\_** | **Indicate where are you storing the tests (what file) and the name of the method/functions being used.** |
| **Results: Pass \_\_\_**✔**\_\_ Fail\_\_\_\_\_\_\_\_** |  |
|  |  |
| **Preconditions for Test:** opl object is constructed. | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Step**  **#** | **Test Step**  **Description** | **Test**  **Data** | **Expected**  **Result** | **Actual**  **Result** | **Notes** |
| 1 | Run ./unittest at testing/build/bin |  |  |  |  |
| 2 | cpl->ReadFile("cpl.csv"); | "cpl.csv" file located in testing/build/bin | report\_.GetNumTotalCandidates() = 16  report\_.GetNumBallots() = 65  report\_.GetNumTotalSeats() = 7 | report\_.GetNumTotalCandidates() = 16  report\_.GetNumBallots() = 65  report\_.GetNumTotalSeats() = 7 | There are more value need to be compare in the test. However, report is not consistent yet. Improve the report interface later. |
| 3 |  |  |  |  |  |
| 4 |  |  |  |  |  |
| 5 |  |  |  |  |  |
| 6 |  |  |  |  |  |
| 7 |  |  |  |  |  |
| 8 |  |  |  |  |  |

**Post condition(s) for Test:**

Successfully read the ballot info then store it to the report.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

|  |  |
| --- | --- |
| **Test Stage: Unit \_**✔**\_ System \_\_** | **Test Date: 11/18/19** |
| **Test Case ID#: VS\_001** | **Name(s) of Testers: Berni D** |
| **Test Description:**   * Location: testing/voting\_system\_unittest.cc * Name: TEST\_F(VotingSystemTest, RandomSelectWinner) * Testing the random method which will return as random array with input length   **Run the test:**  **Terminal at:** testing/ $ make  **Terminal at:** testing/build/bin $ ./unittest   * ***Check terminal to see test pass or fail***   *Notes:*   * *copy opl.csv and cpl.csv from Project1/src/ => Project1/testing/build/bin if don’t have.* * *Make sure this line is uncommented in testing/makefile*   DEFINES += -DVOTING\_SYSTEM\_TESTS |  |
| **Automated: yes\_**✔**\_\_ no \_\_\_** | **Indicate where are you storing the tests (what file) and the name of the method/functions being used.** |
| **Results: Pass \_\_\_**✔**\_\_ Fail\_\_\_\_\_\_\_\_** |  |
|  |  |
| **Preconditions for Test:** voting\_system are constructed. | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Step**  **#** | **Test Step**  **Description** | **Test**  **Data** | **Expected**  **Result** | **Actual**  **Result** | **Notes** |
| 1 | Run result= voting\_system->RandomSelectWinner(100) | Size of output: 100 | An array size = 100 | An array size = 100 |  |
| 2 | Check result.size() = = 100 | Size of output: 100 | An array size = 100 | An array size = 100 |  |
| 3 |  |  |  |  |  |
| 4 |  |  |  |  |  |
|  |  |  |  |  |  |

**Post condition(s) for Test:**

A random array with size = 100

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

|  |  |
| --- | --- |
| **Test Stage: Unit \_**✔**\_ System \_\_** | **Test Date: 11/18/19** |
| **Test Case ID#: VS\_002** | **Name(s) of Testers: Berni D** |
| **Test Description:**   * Location: testing/voting\_system\_unittest.cc * Name: TEST\_F(VotingSystemTest, SplitByComma) * Test: the function will split a string to vector<string> by comma delimiter (“,”)   **Run the test:**  **Terminal at:** testing/ $ make  **Terminal at:** testing/build/bin $ ./unittest   * ***Check terminal to see test pass or fail***   *Notes:*   * *copy opl.csv and cpl.csv from Project1/src/ => Project1/testing/build/bin if don’t have.* * *Make sure this line is uncommented in testing/makefile*   DEFINES += -DVOTING\_SYSTEM\_TESTS |  |
| **Automated: yes\_**✔**\_\_ no \_\_\_** | **Indicate where are you storing the tests (what file) and the name of the method/functions being used.** |
| **Results: Pass \_\_\_**✔**\_\_ Fail\_\_\_\_\_\_\_\_** |  |
|  |  |
| **Preconditions for Test:** voting\_system are constructed. | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Step**  **#** | **Test Step**  **Description** | **Test**  **Data** | **Expected**  **Result** | **Actual**  **Result** | **Notes** |
| 1 | Run result = voting\_system->SplitByComma("Mike,D,11"); | "Mike,D,11" | ["Mike”,”D”,”11"] | ["Mike”,”D”,”11"] |  |
| 2 | Check result.size() = = 3 | "Mike,D,11" | ["Mike”,”D”,”11"] | ["Mike”,”D”,”11"] |  |
| 3 | Check result[0] = = “Mike” |  |  |  |  |
| 4 | Check result[1] = = “D” |  |  |  |  |
|  | Check result[2] = = “11” |  |  |  |  |

**Post condition(s) for Test:**

A string array/vector = ["Mike”,”D”,”11"]

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

|  |  |
| --- | --- |
| **Test Stage: Unit \_**✔**\_ System \_\_** | **Test Date: 11/18/19** |
| **Test Case ID#: VS\_003** | **Name(s) of Testers: Berni D** |
| **Test Description:**   * Location: testing/voting\_system\_unittest.cc * Name: TEST\_F(VotingSystemTest, SplitByComma) * Test: the function will split a string to vector<string> by comma delimiter (“,”)   **Run the test:**  **Terminal at:** testing/ $ make  **Terminal at:** testing/build/bin $ ./unittest   * ***Check terminal to see test pass or fail***   *Notes:*   * *copy opl.csv and cpl.csv from Project1/src/ => Project1/testing/build/bin if don’t have.* * *Make sure this line is uncommented in testing/makefile*   DEFINES += -DVOTING\_SYSTEM\_TESTS |  |
| **Automated: yes\_**✔**\_\_ no \_\_\_** | **Indicate where are you storing the tests (what file) and the name of the method/functions being used.** |
| **Results: Pass \_\_\_**✔**\_\_ Fail\_\_\_\_\_\_\_\_** |  |
|  |  |
| **Preconditions for Test:** voting\_system are constructed. | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Step**  **#** | **Test Step**  **Description** | **Test**  **Data** | **Expected**  **Result** | **Actual**  **Result** | **Notes** |
| 1 | Run result = voting\_system->SplitByComma(""); | "" | [""] | [""] |  |
| 2 | Check result.size() = = 1 | "" | [""] | [""] |  |
| 3 | Check result[0] = = “” | "" | [""] | [""] |  |
| 4 |  |  |  |  |  |
|  |  |  |  |  |  |

**Post condition(s) for Test:**

A string array/vector = [“”]

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

|  |  |
| --- | --- |
| **Test Stage: Unit X System \_\_** | **Test Date: 12/6/19** |
| **Test Case ID#: Report\_UT\_1** | **Name(s) of Testers:** OPLExportsFileCreatedTests |
| **Test Description: After Report class call function ExportToAuditFile(), ExportToPublicFile(), and ExportToShortReport() for OPL in the setup of this test, it creates three file names according to the function call and contain all necessary information for different files.**  **How to** **Run the test:**  **Terminal at:** Project2/testing $ make clean; make all  **Terminal at:** Project2/testing/build/bin $ ./unittest   * ***Check terminal to see test pass or fail***   *Notes:*   * *copy opl.csv from Project2/src/ => Project2/testing/build/bin if doesn’t have.* * *Make sure this line is uncommented in testing/Makefile*   DEFINES += -DREPORT\_UNIT\_TESTS |  |
| **Automated: Automatic** | **Indicate where are you storing the tests (what file) and the name of the method/functions being used.** |
| **Results: Pass X Fail\_\_\_\_\_\_\_\_** |  |
|  |  |
| **Preconditions for Test: Project compile successfully. The functions of exist\_file(), ExportToAuditFile(), ExportToPublicFile(), and ExportToShortReport() get call and run successfully.** | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Step**  **#** | **Test Step**  **Description** | **Test**  **Data** | **Expected**  **Result** | **Actual**  **Result** | **Notes** |
| 1 | Automatic test run for the result of calling function exists\_file() | opl.csv | true | true |  |
| 2 | Automatic test run for the result of call function ExportToAuditFile() in setup of this Test | Audit\_opl.txt | true | true |  |
| 3 | Automatic test run for the result of call function ExportToPublicFile() in setup of this Test | Share\_opl.txt | true | true |  |
| 4 | Automatic test run for the result of call function ExportToShortReport() in setup of this Test | Short\_report\_opl.txt | true | true |  |

**Post condition(s) for Test:** Files opl.csv, Audit\_opl.txt, Share\_opl.txt, and Short\_report\_opl.txt exist in the Project2/testing directory.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

|  |  |
| --- | --- |
| **Test Stage: Unit X System \_\_** | **Test Date: 12/6/19** |
| **Test Case ID#: Report\_UT\_2** | **Name(s) of Testers:** CPLExportsFileCreatedTests |
| **Test Description: After Report class call function ExportToAuditFile(), ExportToPublicFile(), and ExportToShortReport() for CPL in the setup of this test, it creates three file names according to the function call and contain all necessary information for different files.**  **Run the test:**  **Terminal at:** Project2/testing $ make clean; make all  **Terminal at:** Project2/testing/build/bin $ ./unittest   * ***Check terminal to see test pass or fail***   *Notes:*   * *copy cpl.csv from Project2/src/ => Project2/testing/build/bin if doesn’t have.* * *Make sure this line is uncommented in testing/Makefile*   DEFINES += -DREPORT\_UNIT\_TESTS |  |
| **Automated: Automatic** | **Indicate where are you storing the tests (what file) and the name of the method/functions being used.** |
| **Results: Pass X Fail\_\_\_\_\_\_\_\_** |  |
|  |  |
| **Preconditions for Test: Project compile successfully. The functions of exist\_file(), ExportToAuditFile(), ExportToPublicFile(), and ExportToShortReport() get call and run successfully.** | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Step**  **#** | **Test Step**  **Description** | **Test**  **Data** | **Expected**  **Result** | **Actual**  **Result** | **Notes** |
| 1 | Automatic test run for the result of calling function exists\_file() | cpl.csv | true | true |  |
| 2 | Automatic test run for the result of call function ExportToAuditFile() in setup of this Test | Audit\_cpl.txt | true | true |  |
| 3 | Automatic test run for the result of call function ExportToPublicFile() in setup of this Test | Share\_cpl.txt | true | true |  |
| 4 | Automatic test run for the result of call function ExportToShortReport() in setup of this Test | Short\_report\_cpl.txt | true | true |  |

**Post condition(s) for Test:** Files cpl.csv, Audit\_cpl.txt, Share\_cpl.txt, and Short\_report\_cpl.txt exist in the Project2/testing directory.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

|  |  |
| --- | --- |
| **Test Stage: Unit X System \_\_** | **Test Date: 12/6/19** |
| **Test Case ID#: Report\_UT\_3** | **Name(s) of Testers:** ExtractFileNameFunctionTests |
| **Test Description: This test will test the ability to extract only the necessary part of the file name and uses it to name the output file of each run of the voting system. For example, “opl.csv” will return “opl”.**    **Run the test:**  **Terminal at:** Project2/testing $ make clean; make all  **Terminal at:** Project2/testing/build/bin $ ./unittest  ***Check terminal to see test pass or fail***  *Notes:*   * *copy opl.csv and cpl.csv from Project2/src/ => Project2/testing/build/bin if doesn’t have.* * *Make sure this line is uncommented in testing/Makefile*   DEFINES += -DREPORT\_UNIT\_TESTS |  |
| **Automated: Automatic** | **Indicate where are you storing the tests (what file) and the name of the method/functions being used.** |
| **Results: Pass X Fail\_\_\_\_\_\_\_\_** |  |
|  |  |
| **Preconditions for Test: Project compile successfully. The function of** ExtractFileName() **gets call and run successfully.** | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Step**  **#** | **Test Step**  **Description** | **Test**  **Data** | **Expected**  **Result** | **Actual**  **Result** | **Notes** |
| 1 | Automatic test run for the result of calling function ExtractFileName() on OPL data input | opl.csv | true | true |  |
| 2 | Automatic test run for the result of calling function ExportFileName() for CPL data input | cpl.csv | true | true |  |

**Post condition(s) for Test:** N/A

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

|  |  |
| --- | --- |
| **Project Name: Project 1: Voting System Team#22** | |
| **Test Stage: Unit X System \_\_** | **Test Date: 12/6/19** |
| **Test Case ID#: Report\_UT\_4** | **Name(s) of Testers:** FindLongestStringFunctionTests |
| **Test Description: This test will test the ability to find the maximum length of concatenate name of all Candidate for each party.**    **Run the test:**  **Terminal at:** Project2/testing $ make clean; make all  **Terminal at:** Project2/testing/build/bin $ ./unittest  ***Check terminal to see test pass or fail***  *Notes:*   * *Make sure this line is uncommented in testing/Makefile*   DEFINES += -DREPORT\_UNIT\_TESTS |  |
| **Automated: Automatic** | **Indicate where are you storing the tests (what file) and the name of the method/functions being used.** |
| **Results: Pass X Fail\_\_\_\_\_\_\_\_** |  |
|  |  |
| **Preconditions for Test: Project compile successfully. A Party vector is created correctly format.** | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Step**  **#** | **Test Step**  **Description** | **Test**  **Data** | **Expected**  **Result** | **Actual**  **Result** | **Notes** |
| 1 | Automatic test run for the result of calling function FindLongestCPLCandidateName() on OPL data input | party\_vec (a Party class vector that contain all Candidate, you can find out more detail about this vector in the setup part of this unittest) | true | true |  |
| 2 | Automatic test run for the result of calling function FindLongestCPLCandidateName()  for CPL data input | party\_vec (a Party class vector that contain all Candidate, you can find out more detail about this vector in the setup part of this unittest) | true | true |  |

**Post condition(s) for Test:** N/A

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

|  |  |
| --- | --- |
| **Test Stage: Unit X System \_\_** | **Test Date: 12/6/19** |
| **Test Case ID#: Report\_UT\_5** | **Name(s) of Testers:** Display Manual Test |
| **Test Description: This test will test the out of each voting system run. It prints out in the terminal all related information of each voting run.**    **Run the test:**  **Terminal at:** Project2/testing $ make clean; make all  **Terminal at:** Project2/testing/build/bin $ ./unittest  ***Check terminal to see test pass or fail***  *Notes:*   * *copy opl.csv and cpl.csv from Project2/src/ => Project2/testing/build/bin if doesn’t have.* * *Make sure this line is uncommented in testing/Makefile*   DEFINES += -DREPORT\_UNIT\_TESTS |  |
| **Automated: Manually** | **Indicate where are you storing the tests (what file) and the name of the method/functions being used.** |
| **Results: Pass X Fail\_\_\_\_\_\_\_\_** |  |
|  |  |
| **Preconditions for Test: Project compile successfully. A Party vector is created correctly format.** | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Step**  **#** | **Test Step**  **Description** | **Test**  **Data** | **Expected**  **Result** | **Actual**  **Result** | **Notes** |
| 1 | Run ./unittest | opl.csv and cpl.csv | N/A | N/A |  |
| 2 | Check for Voting Type is OPL | opl.csv | true | true |  |
| 3 | Check for Number of Total Candidates = 4 | opl.csv | true | true |  |
| 4 | Check for Party D Candidates List have Davis, Victor | opl.csv | true | true |  |
| 5 | Check for Party D Candidates List have Frank, Janifer | opl.csv | true | true |  |
| 6 | Check for Number of Total Seats = 2 | opl.csv | true | true |  |
| 7 | Check for Number of Ballots = 10 | opl.csv | true | true |  |
| 8 | Check for Ballot ID list = 1 2 3 4 5 6 7 8 9 10 | opl.csv | true | true |  |
| 9 | For table, Check for Candidate Victor belong to Party D won with total of 3 votes | opl.csv | true | true |  |
| 10 | For table, Check for Candidate Frank belong to Party R won with total of 4 votes | opl.csv | true | true |  |
| 11 | Check for Voting Type is OPL | cpl.csv | true | true |  |
| 12 | Check for Number of Total Candidates = 4 | cpl.csv | true | true |  |
| 13 | Check for Party D Candidates List have Davis, Victor | cpl.csv | true | true |  |
| 14 | Check for Party D Candidates List have Frank, Janifer | cpl.csv | true | true |  |
| 15 | Check for Number of Total Seats = 2 | cpl.csv | true | true |  |
| 16 | Check for Number of Ballots = 10 | cpl.csv | true | true |  |
| 17 | Check for Ballot ID list = 1 2 3 4 5 6 7 8 9 10 | cpl.csv | true | true |  |
| 18 | For table, Check for Party D won 1 seat belong to Victor with 3 votes | cpl.csv | true | true |  |
| 19 | For table, Check for Party R won 1 seat belong to Frank with 4 votes | cpl.csv | true | true |  |

**Post condition(s) for Test:** N/A

|  |  |
| --- | --- |
| **Project Name: Project 2: Voting System Team#22** | |
| **Test Stage: Unit \_**✔**\_ System \_\_** | **Test Date: 12/07/19** |
| **Test Case ID#: cpl\_Test\_01** | **Name(s) of Testers: Team #22** |
| **Test Description:**   * Location: Project2/testing/cpl\_readfile\_unittest.cc * Name: TEST\_F(CPLReadFileTest, ReadFile) * Test: the function ReadFile in cpl.cc to check if the file read correctly.   **Run the test:**  UncommentTEST\_F(CPLReadFileTest, ReadFile) functions in cpl\_readfile\_unittest.cc  Comment out others test if you want to run only this test  **Terminal at:** Project2/testing/ $ make  **Terminal at:** testing/build/bin $ ./unittest   * ***Check terminal to see test pass or fail***   *Notes:*   * *copy opl.csv and cpl.csv from Project1/src/ => Project1/testing/build/bin if don’t have.* * *Make sure this line is uncommented in testing/makefile* |  |
| **Automated: yes\_\_\_ no \_**✔**\_\_** | **Indicate where are you storing the tests (what file) and the name of the method/functions being used.** |
| **Results: Pass \_\_\_\_\_ Fail\_\_\_\_**✔**\_\_\_\_** |  |
|  |  |
| **Preconditions for Test:** Testing the file reading the file in cpl.csv correctly | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Step**  **#** | **Test Step**  **Description** | **Test**  **Data** | **Expected**  **Result** | **Actual**  **Result** | **Notes** |
| 1 | Project2/testing/ | “cpl.csv” |  |  |  |
| 2 | Run  make | “cpl.csv” |  |  |  |
| 3 | ./build/bin/unittest | “cpl.csv” |  |  |  |
| 4 | Check the total number of candidates | “cpl.csv” | 17 | 16 |  |
| 5 | Check the total number of ballots | “cpl.csv” | 100 | 65 |  |
| 6 | Check the total number of seats | “cpl.csv” | 10 | 7 |  |
| 7 |  |  |  |  |  |
| 8 |  |  |  |  |  |

**Post condition(s) for Test: |**

Reading the cpl file incorrectly

|  |  |
| --- | --- |
| **Project Name: Project 2: Voting System Team#22** | |
| **Test Stage: Unit \_**✔**\_ System \_\_** | **Test Date: 12/07/19** |
| **Test Case ID#: opl\_Test\_01** | **Name(s) of Testers: Team#22** |
| **Test Description:**   * Location: Project2/testing/opl\_readfile\_unittest.cc * Name: TEST\_F(OPLReadFileTest, ReadFile) * Test: the function ReadFile in opl.cc to check if the file read correctly.   **Run the test:**  UncommentTEST\_F(OPLReadFileTest, ReadFile) functions in opl\_readfile\_unittest.cc  Comment out others test if you want to run only this test  **Terminal at:** Project2/testing/ $ make  **Terminal at:** testing/build/bin $ ./unittest   * ***Check terminal to see test pass or fail***   *Notes:*   * *copy opl.csv and cpl.csv from Project1/src/ => Project1/testing/build/bin if don’t have.* * *Make sure this line is uncommented in testing/makefile* |  |
| **Automated: yes\_\_\_ no \_**✔**\_\_** | **Indicate where are you storing the tests (what file) and the name of the method/functions being used.** |
| **Results: Pass \_\_\_\_\_ Fail\_\_\_\_**✔**\_\_\_\_** |  |
|  |  |
| **Preconditions for Test:** Testing the file reading the file in opl.csv correctly | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Step**  **#** | **Test Step**  **Description** | **Test**  **Data** | **Expected**  **Result** | **Actual**  **Result** | **Notes** |
| 1 | Project2/testing/ | “opl.csv” |  |  |  |
| 2 | Run  make | “opl.csv” |  |  |  |
| 3 | ./build/bin/unittest | “opl.csv” |  |  |  |
| 4 | Check the total number of candidates | “opl.csv” | 7 | 6 |  |
| 5 | Check the total number of ballots | “opl.csv” | 100 | 9 |  |
| 6 | Check the total number of seats | “opl.csv” | 5 | 3 |  |
| 7 |  |  |  |  |  |
| 8 |  |  |  |  |  |

**Post condition(s) for Test:**

Reading the opl file incorrectly

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

|  |  |
| --- | --- |
| **Test Stage: Unit \_**✔**\_ System \_\_** | **Test Date: 12/07/19** |
| **Test Case ID#: Party\_UT\_001** | **Name(s) of Testers: Danet Chheng** |
| **Test Description:**   * Location: testing/party\_UT.cc * Name: TEST\_F(PartyTests, AddCandidateTests) * Test: Adding all candidates in the list of candidate for each party.   **Run the test:**   * UncommentTEST\_F(PartyTests, AddCandidateTests) functions in party\_UT.cc * Comment out other tests if you want to run only this test.   **Terminal at:** testing/ $ make  **Terminal at:** testing/build/bin $ ./unittest   * ***Check terminal message to see test pass or fail***   *Notes:*   * *copy opl.csv and cpl.csv from Project1/testing / => Project1/testing/build/bin if don’t have.* * *Make sure this line is uncommented in testing/makefile*   DEFINES += -DPARTY\_TESTS |  |
| **Automated: yes\_**✔**\_\_ no \_\_\_** | **Indicate where are you storing the tests (what file) and the name of the method/functions being used.** |
| **Results: Pass \_\_\_**✔**\_\_ Fail\_\_\_\_\_\_\_\_** |  |
|  |  |
| **Preconditions for Test: T**esting add candidate to the list of candidate for each party | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Step**  **#** | **Test Step**  **Description** | **Test**  **Data** | **Expected**  **Result** | **Actual**  **Result** | **Notes** |
| 1 | Run ./unittest at testing/build/bin |  |  |  |  |
| 2 | party->AddCandidate(\*cand1); |  | “Benjamin” | “Benjamin” |  |
| 3 | party->AddCandidate(\*cand2); |  | “Sam” | “Sam” |  |
| 4 | party->AddCandidate(\*cand3); |  | “Steve” | “Steve” |  |
| 5 | party->AddCandidate(\*cand4); |  | “Alice” | “Alice” |  |
| 6 |  |  |  |  |  |
| 7 |  |  |  |  |  |
| 8 |  |  |  |  |  |

**Post condition(s) for Test:**

Successfully add all the candidates in the list of candidates for each party.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

|  |  |
| --- | --- |
| **Test Stage: Unit \_**✔**\_ System \_\_** | **Test Date: 12/07/19** |
| **Test Case ID#: Party\_UT\_002** | **Name(s) of Testers: Danet Chheng** |
| **Test Description:**   * Location: testing/party\_UT.cc * Name: TEST\_F(PartyTests, AddBallotIdTests) * Test: Adding all ballot identifications in the list of ballot id for each party.   **Run the test:**   * UncommentTEST\_F(PartyTests, AddBallotIdTests) functions in party\_UT.cc * Comment out other tests if you want to run only this test.   **Terminal at:** testing/ $ make  **Terminal at:** testing/build/bin $ ./unittest   * ***Check terminal message to see test pass or fail***   *Notes:*   * *copy opl.csv and cpl.csv from Project1/testing / => Project1/testing/build/bin if don’t have.* * *Make sure this line is uncommented in testing/makefile*   DEFINES += -DPARTY\_TESTS |  |
| **Automated: yes\_**✔**\_\_ no \_\_\_** | **Indicate where are you storing the tests (what file) and the name of the method/functions being used.** |
| **Results: Pass \_\_\_**✔**\_\_ Fail\_\_\_\_\_\_\_\_** |  |
|  |  |
| **Preconditions for Test: T**esting add ballot identification to the list of ballot id for each party | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Step**  **#** | **Test Step**  **Description** | **Test**  **Data** | **Expected**  **Result** | **Actual**  **Result** | **Notes** |
| 1 | Run ./unittest at testing/build/bin |  |  |  |  |
| 2 | party->AddBallotId(1); |  | 1 | 1 |  |
| 3 | party->AddBallotId(2); |  | 2 | 2 |  |
| 4 | party->AddBallotId(3); |  | 3 | 3 |  |
| 5 | party->AddBallotId(4); |  | 4 | 4 |  |
| 6 |  |  |  |  |  |
| 7 |  |  |  |  |  |
| 8 |  |  |  |  |  |

**Post condition(s) for Test:**

Successfully add all the ballot identifications in the list of ballots id for each party.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*PROJECT 2\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

|  |
| --- |
| **PBI & Task Description: PBI 3 - Fix ExportToFile() for OPL Bug, include votes a candidate or party had, order of ballot of being received** |
| **Team Member(s) Responsible: Khoa Tran** |
| **Inputs:**   * opl.csv in /Project2/Input |
| **Tests:**   * (opl.csv) Test for successful export a file name audit\_result.txt in Project2/build/bin with a voting type of OPL (manually) |
| **Output:**   * ExportToFile() execute successfully and a file audit\_result.txt created in Project2/build/bin with a voting type of OPL |
| **Passed or Failed:** Passed |
| **Date: 12/08/2019** |

|  |
| --- |
| **PBI & Task Description: PBI 4 - Fix ExportToFile() for CPL Bug, include votes a candidate or party had, order of ballot of being received** |
| **Team Member(s) Responsible: Khoa Tran** |
| **Inputs:**   * cpl.csv in /Project2/Input |
| **Tests:**   * (cpl.csv) Test for successful export a file name audit\_result.txt in Project2/build/bin with a voting type of CPL (manually) |
| **Output:**   * ExportToFile() execute successfully and a file audit\_result.txt created in Project2/build/bin with a voting type of CPL |
| **Passed or Failed:** Passed |
| **Date: 12/08/2019** |

|  |
| --- |
| **PBI & Task Description: PBI 5 - Report Display Bug - Task: Manual Test the output of Display function in the terminal for OPL** |
| **Team Member(s) Responsible: Khoa Tran** |
| **Inputs:** A party vector that including four candidates (this can be found in the Project2/testing file report\_UT.cc in the Setup() function) |
| **Tests:**   * Test 1: For a section output on the terminal * Test 2: For Voting Type * Test 3: For Number of Total Candidates * Test 4: For Number of Total Seats * Test 5: For Number of Ballots * Test 6: For A PARTY/SEAT RECIEVE section with information of party, number seats win, and total votes * Test 7: For OPL REPORT WINNERS table that contains Candidate Name, Party Name, Number of Vote print out with correct information after audit processed. |
| **Output: ALL OK.**   * Test 1: A table section printed out in terminal * Test 2: Expected OPL (OK) * Test 3: Expected 4 (OK) * Test 4: Expected 2 (OK) * Test 5: Expected 10 (OK) * Test 6: Expected   D - 1 seats - 3 total votes  R - 1 seats - 4 total votes  Actual:  D - 1 seats - 0 total votes (FAILED)  R - 1 seats - 0 total votes (FAILED   * Test 7: Expected a table  |  |  |  | | --- | --- | --- | | Candidate Name | Party Name | Number of Vote | | Victor | D | 3 | | Frank | R | 4 | |
| **Passed or Failed: Failed** |
| **Date: 12/07/2019** |
| **Task Created: Fix Display() for OPL with the correct total number of votes for Winner** |

|  |
| --- |
| **PBI & Task Description: PBI 5 - Report Display Bug - Task: Fix Display() for OPL with the correct total number of votes for Winner and Manual RE-Test the output of Display() in the terminal for OPL** |
| **Team Member(s) Responsible: Khoa Tran** |
| **Inputs:** A party vector that including four candidates (this can be found in the Project2/testing file report\_UT.cc in the Setup() function) |
| **Tests:**   * Test 1: For a section output on the terminal * Test 2: For Voting Type * Test 3: For Number of Total Candidates * Test 4: For Number of Total Seats * Test 5: For Number of Ballots * Test 6: For A PARTY/SEAT RECIEVE section with information of party, number seats win, and total votes * Test 7: For OPL REPORT WINNERS table that contains Candidate Name, Party Name, Number of Vote print out with correct information after audit processed. |
| **Output: ALL OK.**   * Test 1: A table section printed out in terminal * Test 2: Expected OPL (OK) * Test 3: Expected 4 (OK) * Test 4: Expected 2 (OK) * Test 5: Expected 10 (OK) * Test 6: Expected   D - 1 seats - 3 total votes  R - 1 seats - 4 total votes  Actual:  D - 1 seats - 3 total votes (PASSED)  R - 1 seats - 4 total votes (PASSED)   * Test 7: Expected a table  |  |  |  | | --- | --- | --- | | Candidate Name | Party Name | Number of Vote | | Victor | D | 3 | | Frank | R | 4 | |
| **Passed or Failed: Passed** |
| **Date: 12/08/2019** |

|  |
| --- |
| **PBI & Task Description: PBI 6 - Report Display Bug - Task: Manual Test the output of Display function in the terminal for CPL** |
| **Team Member(s) Responsible: Khoa Tran** |
| **Inputs:** A party vector that including four candidates (this can be found in the Project2/testing file report\_UT.cc in the Setup() function) |
| **Tests:**   * Test 1: For a section output on the terminal * Test 2: For Voting Type * Test 3: For Number of Total Candidates * Test 4: For Number of Total Seats * Test 5: For Number of Ballot * Test 6: For A PARTY/SEAT RECIEVE section with information of party, number seats win, and total votes * Test 7: For CPL REPORT Winners table contains Party, Number of Seats, Winners Name, Number of Vote print out with correct information after audit processed. |
| **Output: ALL OK.**   * Test 1: A table section printed out in terminal * Test 2: Expected CPL (OK) * Test 3: Expected 4 (OK) * Test 4: Expected 2 (OK) * Test 5: Expected 10 (OK) * Test 6: Expected:   D - 1 seats - 3 total votes  R - 1 seats - 4 total votes  Actual:  D - 1 seats - 0 total votes (FAILED)  R - 1 seats - 0 total votes (FAILED)   * Test 7: OK  |  |  |  |  | | --- | --- | --- | --- | | Party | Number of Seats | Winners Name | Number of Vote | | D | 1 | Victor | 3 | | R | 1 | Frank | 4 | |
| **Passed or Failed: Failed** |
| **Date: 12/07/2019** |
| **Task Create: Fix Display() for CPL with the correct total number of votes for Winner** |

|  |
| --- |
| **PBI & Task Description: PBI 6 - Report Display Bug - Task: Fix Display() for CPL with the correct total number of votes for Winner and Manual RE-Test the output of Display() in the terminal for CPL** |
| **Team Member(s) Responsible: Khoa Tran** |
| **Inputs:** A party vector that including four candidates (this can be found in the Project2/testing file report\_UT.cc in the Setup() function) |
| **Tests:**   * Test 1: For a section output on the terminal * Test 2: For Voting Type * Test 3: For Number of Total Candidates * Test 4: For Number of Total Seats * Test 5: For Number of Ballot * Test 6: For A PARTY/SEAT RECIEVE section with information of party, number seats win, and total votes * Test 7: For CPL REPORT Winners table contains Party, Number of Seats, Winners Name, Number of Vote print out with correct information after audit processed. |
| **Output: ALL OK.**   * Test 1: A table section printed out in terminal * Test 2: Expected CPL (OK) * Test 3: Expected 4 (OK) * Test 4: Expected 2 (OK) * Test 5: Expected 10 (OK) * Test 6: Expected:   D - 1 seats - 3 total votes  R - 1 seats - 4 total votes  Actual:  D - 1 seats - 3 total votes (PASS )  R - 1 seats - 4 total votes (PASS)   * Test 7: OK  |  |  |  |  | | --- | --- | --- | --- | | Party | Number of Seats | Winners Name | Number of Vote | | D | 1 | Victor | 3 | | R | 1 | Frank | 4 | |
| **Passed or Failed: Passed.** |
| **Date: 12/08/2019** |

|  |
| --- |
| **PBI & Task Description: PBI 5: Report Class Bug - Task: Change the Display Format - OPL display (Manual)** |
| **Team Member(s) Responsible: Khoa Tran** |
| **Inputs:** A party vector that including four candidates (this can be found in the Project2/testing file report\_UT.cc in the Setup() function) |
| **Tests:**   * Test 1: For a section output in the terminal * Test 2: For format of table, a clean display table with a good space between column that allows user to read the content easily. * Test 3: For format of table, everything will align to the left of table |
| **Output: ALL OKAY**   * Test 1: A table section printed out in terminal * Test 2: OK. Expected a table look like:   Candidate Name Party Name Number of Vote  Victor D 3  Frank R 4     * Test 3: OK. Expected a table with all columns align to the left   Candidate Name Party Name Number of Vote  Victor D 3  Frank R 4 |
| **Passed or Failed: Passed** |
| **Date: 12/10/2019** |

|  |
| --- |
| **PBI & Task Description: PBI 6: Report Class Bug - Task: Change the Display Format - CPL display (Manual)** |
| **Team Member(s) Responsible: Khoa Tran** |
| **Inputs:** A party vector that including four candidates (this can be found in the Project2/testing file report\_UT.cc in the Setup() function) |
| **Tests:**   * Test 1: For a section output in the terminal * Test 2: For format of table, a clean display table with a good space between column that allows user to read the content easily. * Test 3: For format of table, everything will align to the left of table |
| **Output: ALL OKAY**   * Test 1: A table section printed out in terminal * Test 2: OK. Expected a table look like:   Party Number of Seats Winners Name Number of Vote  D 1 Victor 3  R 1 Frank 4     * Test 3: OK. Expected a table with all columns align to the left   Party Number of Seats Winners Name Number of Vote  D 1 Victor 3  R 1 Frank 4 |
| **Passed or Failed: Passed** |
| **Date: 12/10/2019** |

|  |
| --- |
| **PBI & Task Description: PBI 7 – Unittest for adding candidate into the list of candidates for party class (party\_UT\_001)** |
| **Team Member(s) Responsible: Danet Chheng** |
| **Inputs:** A candidate vector that including four candidates (this can be found in the Project2/testing file party\_UT.cc in the Setup() function) |
| **Tests:**  TEST\_F(PartyTests, AddCandidateTests)   * Test1: Expect “Benjamin” is added into the list of candidates for their party * Test2: Expect “Sam” is added into the list of candidates for their party * Test3: Expect “Steve” is added into the list of candidates for their party * Test4: Expect “Alice” is added into the list of candidates for their party |
| **Output: ALL OK.**   * Test1: Expect “Benjamin” is added: OK * Test2: Expect “Sam” is added: OK * Test3: Expect “Steve” is added: OK * Test4: Expect “Alice” is added: OK |
| **Passed or Failed: Passed.** |
| **Date: 12/07/2019** |

Before:

|  |
| --- |
| **PBI & Task Description: PBI 7 – Unittest for adding ballot Id into the list of ballot id for party class (party\_UT\_002)** |
| **Team Member(s) Responsible: Danet Chheng** |
| **Inputs:** An integer vector for list\_ballot\_id\_ including four list\_ballot\_id\_ (this can be found in the Project2/testing file party\_UT.cc in the Setup() function) |
| **Tests:**  TEST\_F(PartyTests, AddBallotIdTests)   * Test1: Expect 1 is added into the list of ballot id * Test2: Expect 2 is added into the list of ballot id * Test3: Expect 3 is added into the list of ballot id * Test4: Expect 4 is added into the list of ballot id |
| **Output: Errors**   * Test1: Error to add the ballot id * Test2: Error to add the ballot id * Test3: Error to add the ballot id * Test4: Error to add the ballot id |
| **Passed or Failed: Failed** |
| **Date: 12/05/2019** |

After:

|  |
| --- |
| **PBI & Task Description: PBI 7 – Unittest for adding ballot Id into the list of ballot id for party class (party\_UT\_002)** |
| **Team Member(s) Responsible: Danet Chheng** |
| **Inputs:** An integer vector for list\_ballot\_id\_ including four list\_ballot\_id\_ (this can be found in the Project2/testing file party\_UT.cc in the Setup() function) |
| **Tests:**  TEST\_F(PartyTests, AddBallotIdTests)   * Test1: Expect 1 is added into the list of ballot id * Test2: Expect 2 is added into the list of ballot id * Test3: Expect 3 is added into the list of ballot id * Test4: Expect 4 is added into the list of ballot id |
| **Output: ALL OK.**   * Test1: Expect 1 is added: OK * Test2: Expect 2 is added: OK * Test3: Expect 3 is added: OK * Test4: Expect 4 is added: OK |
| **Passed or Failed: Passed.** |
| **Date: 12/07/2019** |

**PBI1 : BUG #1**

* **CASE1**: Test for OPL regular cases: (include tie party, tie candidate, or both)

|  |
| --- |
| The PBI, the Task Description (from Sprint Log) with Unique Testing Number: **BUG-1-Case1-01**  Checking audit opl regular case (no tie happened) |
| Team Member(s) Responsible: Berni D |
| Inputs:   * opl.csv * opl\_regular\_1.csv * opl\_regular\_2.csv |
| Tests:   * Test for successful display/export/sharing report (manually) * Test for successful the correct result list of winners (manually) * Result list of winner is in specific order (manually) |
| Outputs:   * OK display/export/sharing report * OK print correctly The result list of winners * ERR the order result list of winner |
| Passed or Failed: Failed |
| Date: 12/01/19 |
| Task Create: Fix Audit() OPL correct order |

|  |
| --- |
| The PBI, the Task Description (from Sprint Log) with Unique Testing Number: **BUG-1-Case1-02**  Checking audit opl regular case (no tie happened) |
| Team Member(s) Responsible: Berni D |
| Inputs:   * opl.csv * opl\_regular\_1.csv * opl\_regular\_2.csv |
| Tests:   * Test for successful display/export/sharing report (manually) * Test for successful the correct result list of winners (manually) * Result list of winner is in specific order (manually) |
| Outputs:   * OK display/export/sharing report * OK print correctly The result list of winners * OK the order result list of winner |
| Passed or Failed: Passed |
| Date: 12/05/19 |
| Task Completed. |

(Retest even though its passed)

|  |
| --- |
| The PBI, the Task Description (from Sprint Log) with Unique Testing Number: **BUG-1-Case1-03**  Checking audit opl regular case (no tie happened) |
| Team Member(s) Responsible: Berni D |
| Inputs:   * opl.csv * opl\_regular\_1.csv * opl\_regular\_2.csv |
| Tests:   * Test for successful display/export/sharing report (manually) * Test for successful the correct result list of winners (manually) * Result list of winner is in specific order (manually) |
| Outputs:   * OK display/export/sharing report * OK print correctly The result list of winners * OK the order result list of winner |
| Passed or Failed: Passed |
| Date: 12/10/19 |

* **CASE2:** Test for OPL random cases:

|  |
| --- |
| The PBI, the Task Description (from Sprint Log) with Unique Testing Number: **BUG-1-Case2-01**  Checking audit opl when tie happened |
| Team Member(s) Responsible: Berni D |
| Inputs:   * Opl\_random\_1.csv * Opl\_random\_2.csv |
| Tests:   * Test for successful display/export/sharing report (manually) * Test for successful the correct result list of winners (manually) * Result list of winner is in specific order (manually) * Random party when tie * Random candidate when tie |
| Outputs:   * OK display/export/sharing report * ERR print correctly The result list of winners * ERR the order result list of winner * OK Random party when tie * ERR random candidate when tie * ERR random when both party and candidate are tie |
| Passed or Failed: Failed |
| Date: 11/25/19 |
| Task Create: Fix Audit() OPL when tie occur, make sure Audit() are fully tested |

|  |
| --- |
| The PBI, the Task Description (from Sprint Log) with Unique Testing Number: **BUG-1-Case2-02**  Checking audit opl when tie happened |
| Team Member(s) Responsible: Berni D |
| Inputs:   * Opl\_random\_1.csv * Opl\_random\_2.csv |
| Tests:   * Test for successful display/export/sharing report (manually) * Test for successful the correct result list of winners (manually) * Result list of winner is in specific order (manually) * Random party when tie * Random candidate when tie |
| Outputs:   * OK display/export/sharing report * ERR print correctly The result list of winners * ERR the order result list of winner * OK Random party when tie * ERR candidate when tie * ERR random when both party and candidate are tie |
| Passed or Failed: Failed |
| Date: 11/30/19 |
| Tasks: Still Fixing Audit() OPL when tie occur, make sure Audit() are fully tested |

|  |
| --- |
| The PBI, the Task Description (from Sprint Log) with Unique Testing Number: **BUG-1-Case2-03**  Checking audit opl when tie happened |
| Team Member(s) Responsible: Berni D |
| Inputs:   * Opl\_random\_1.csv * Opl\_random\_2.csv * Opl\_random\_3.csv |
| Tests:   * Test for successful display/export/sharing report (manually) * Test for successful the correct result list of winners (manually) * Result list of winner is in specific order (manually) * Random party when tie * Random candidate when tie * Random when both candidate and party are tied |
| Outputs:   * OK display/export/sharing report * OK print correctly The result list of winners * ERR the order result list of winner * OK Random party when tie * OK candidate when tie * ERR random when both party and candidate are tie |
| Passed or Failed: Failed |
| Date: 12/08/19 |
| Task Create: None, there is a task already for that. |

|  |
| --- |
| The PBI, the Task Description (from Sprint Log) with Unique Testing Number: **BUG-1-Case2-04**  Checking audit opl when tie happened |
| Team Member(s) Responsible: Berni D |
| Inputs:   * Opl\_random\_1.csv * Opl\_random\_2.csv * Opl\_random\_3.csv |
| Tests:   * Test for successful display/export/sharing report (manually) * Test for successful the correct result list of winners (manually + auto test) * Result list of winner is in specific order (manually + auto test) * Random party when tie * Random candidate when tie |
| Outputs:   * OK display/export/sharing report * OK print correctly The result list of winners * OK the order result list of winner * OK Random winners * OK Random party when tie * OK candidate when tie |
| Passed or Failed: Passed |
| Date: 12/09/19 |
| Task Create: None. Task completed |

Unit test and system/integration test are provided in Testing folder

**PBI1 : BUG #1 COMPLETED.**

**PBI1 : BUG #2**

* **CASE1**: Test for CPL regular cases:

|  |
| --- |
| The PBI, the Task Description (from Sprint Log) with Unique Testing Number: **BUG-2-Case1-01**  Checking audit cpl regular case (no tie happened) |
| Team Member(s) Responsible: Berni D |
| Inputs:   * cpl.csv * cpl\_regular\_1.csv * cpl\_regular\_2.csv |
| Tests:   * Test for successful display/export/sharing report (manually) * Test for successful the correct result list of winners (manually) * Result list of winner is in specific order (manually) |
| Outputs:   * OK display/export/sharing report * ERR print correctly The result list of winners * ERR the order result list of winners * Weird pointer exception somewhere in code |
| Passed or Failed: Failed |
| Date: 11/24/19 |
| Task Create: Refactor Audit() CPL, fix weird pointer exception |

|  |
| --- |
| The PBI, the Task Description (from Sprint Log) with Unique Testing Number: **BUG-2-Case1-02**  Checking audit cpl regular case (no tie happened) |
| Team Member(s) Responsible: Berni D |
| Inputs:   * cpl.csv * cpl\_regular\_1.csv * cpl\_regular\_2.csv |
| Tests:   * Test for successful display/export/sharing report (manually) * Test for successful the correct result list of winners (manually) * Result list of winner is in specific order (manually) |
| Outputs:   * OK display/export/sharing report * ERR print correctly The result list of winners * ERR the order result list of winners * Weird pointer exception somewhere in code |
| Passed or Failed: Failed |
| Date: 11/30/19 |
| Task Create: None, already have a task |

|  |
| --- |
| The PBI, the Task Description (from Sprint Log) with Unique Testing Number: **BUG-2-Case1-03**  Checking audit cpl regular case (no tie happened) |
| Team Member(s) Responsible: Berni D |
| Inputs:   * cpl.csv * cpl\_regular\_1.csv * cpl\_regular\_2.csv |
| Tests:   * Test for successful display/export/sharing report (manually) * Test for successful the correct result list of winners (manually) * Result list of winner is in specific order (manually) |
| Outputs:   * OK display/export/sharing report * ERR print correctly The result list of winners * ERR the order result list of winners * Weird pointer exception somewhere in code |
| Passed or Failed: Failed |
| Date: 12/03/19 |
| Task Create: None, already have a task |

|  |
| --- |
| The PBI, the Task Description (from Sprint Log) with Unique Testing Number: **BUG-2-Case1-04**  Checking audit cpl regular case (no tie happened) |
| Team Member(s) Responsible: Berni D |
| Inputs:   * cpl.csv * cpl\_regular\_1.csv * cpl\_regular\_2.csv |
| Tests:   * Test for successful display/export/sharing report (manually) * Test for successful the correct result list of winners (manually) * Result list of winner is in specific order (manually) |
| Outputs:   * OK display/export/sharing report * OK print correctly The result list of winners * OK the order result list of winners * NO weird pointer exception somewhere in code |
| Passed or Failed: PASSED |
| Date: 12/09/19 |
| Task Create: None. Task completed |

* **CASE2:** Test for CPL random cases:

|  |
| --- |
| The PBI, the Task Description (from Sprint Log) with Unique Testing Number: **BUG-2-Case2-01**  Checking audit cpl when tie happened |
| Team Member(s) Responsible: Berni D |
| Inputs:   * cpl\_random\_1.csv * cpl\_random\_2.csv * cpl\_random\_3.csv |
| Tests:   * Test for successful display/export/sharing report (manually) * Test for successful the correct result list of winners (manually) * Result list of winner is in specific order (manually) * Random party when tie (manually) |
| Outputs:   * OK display/export/sharing report * ERR print correctly The result list of winners * ERR the order result list of winner * Weird pointer exception somewhere in code * ERR Random party when tie |
| Passed or Failed: Failed |
| Date: 11/24/19 |
| Task Create: Fix Audit() CPL when tie occur, make sure Audit() are fully tested |

|  |
| --- |
| The PBI, the Task Description (from Sprint Log) with Unique Testing Number: **BUG-2-Case2-02**  Checking audit cpl when tie happened |
| Team Member(s) Responsible: Berni D |
| Inputs:   * cpl\_random\_1.csv * cpl\_random\_2.csv * cpl\_random\_3.csv |
| Tests:   * Test for successful display/export/sharing report (manually) * Test for successful the correct result list of winners (manually) * Result list of winner is in specific order (manually) * Random party when tie (manually) |
| Outputs:   * OK display/export/sharing report * ERR print correctly The result list of winners * ERR the order result list of winner * Weird pointer exception somewhere in code * ERR Random party when tie |
| Passed or Failed: Failed |
| Date: 11/30/19 |
| Task Create: Fix Audit() OPL when tie occur, make sure Audit() are fully tested |

|  |
| --- |
| The PBI, the Task Description (from Sprint Log) with Unique Testing Number: **BUG-2-Case2-03**  Checking audit cpl when tie happened |
| Team Member(s) Responsible: Berni D |
| Inputs:   * cpl\_random\_1.csv * cpl\_random\_2.csv * cpl\_random\_3.csv |
| Tests:   * Test for successful display/export/sharing report (manually) * Test for successful the correct result list of winners (manually) * Result list of winner is in specific order (manually) * Random party when tie (manually) |
| Outputs:   * OK display/export/sharing report * ERR print correctly The result list of winners * ERR the order result list of winner * Weird pointer exception somewhere in code * ERR Random party when tie |
| Passed or Failed: Failed |
| Date: 12/03/19 |
| Task Create: Fix Audit() OPL when tie occur, make sure Audit() are fully tested |

|  |
| --- |
| The PBI, the Task Description (from Sprint Log) with Unique Testing Number: **BUG-2-Case2-04**  Checking audit cpl when tie happened |
| Team Member(s) Responsible: Berni D |
| Inputs:   * cpl\_random\_1.csv * cpl\_random\_2.csv * cpl\_random\_3.csv |
| Tests:   * Test for successful display/export/sharing report (manually) * Test for successful the correct result list of winners (manually) * Result list of winner is in specific order (manually) * Random party when tie (manually) |
| Outputs:   * OK display/export/sharing report * OK print correctly The result list of winners * OK the order result list of winner * NO weird pointer exception somewhere in code * OK Random party when tie |
| Passed or Failed: Passed |
| Date: 12/10/19 |
| Task Create: None. Task completed |

**PBI1 : BUG #2 COMPLETED.**

**PBI1 : BUG #7**

**This is tasks to provide all possible test cases to cover all potential situations. It includes all missed test from Project 1 (Unit and Integration Test) – No need to provide test log for this. As long as the tests compile and run in the machine.**

**PBI1 : BUG #7 COMPLETED.**

**PBI2 : #8**

**Case1: Testing input/open file button for opl csv**

|  |
| --- |
| The PBI, the Task Description (from Sprint Log) with Unique Testing Number:  **PBI-8-Case1-01 (manual)**  **Design GUI (.xml or using Graphical Library) for File Search, Open** |
| Team Member(s) Responsible: Berni D |
| Inputs:   * opl.csv which is a csv file |
| Tests:   * “Open File” Button On Click * Test for successful Select a file using GUI * Test for successful Select (.csv) using GUI |
| Outputs:   * There is no “Open File” Button |
| Passed or Failed: Failed |
| Date: 12/01/19 |

|  |
| --- |
| The PBI, the Task Description (from Sprint Log) with Unique Testing Number:  **PBI-8-Case1-02 (manual)**  **Design GUI (.xml or using Graphical Library) for File Search, Open** |
| Team Member(s) Responsible: Berni D |
| Inputs:   * opl.csv which is a csv file |
| Tests:   * “Open File” Button On Click * Test for successful Select a file using GUI * Test for successful Select (.csv) using GUI |
| Outputs:   * OK select a file * OK The path file of opl.csv are showed in “File Selected” textbox * OK The button is clickable * OK allow select only (.csv) |
| Passed or Failed: Passed |
| Date: 12/07/19 |

**Case2: Testing input/open file button for cpl csv**

|  |
| --- |
| The PBI, the Task Description (from Sprint Log) with Unique Testing Number:  **PBI-8-Case2-01 (manual)**  **Design GUI (.xml or using Graphical Library) for File Search** |
| Team Member(s) Responsible: Berni D |
| Inputs:   * cpl.csv which is a csv file |
| Tests:   * “Open File” Button On Click * Test for successful Select a file using GUI * Test for ability to search for a file name * Test for ability to cancel selection or searching * Test for successful Select (.csv) using GUI |
| Outputs:   * OK select a file * OK can search for file in machine * OK The path file of opl.csv are showed in “File Selected” textbox * OK The button is clickable * OK Can cancel selection * OK allow select only (.csv) |
| Passed or Failed: Passed |
| Date: 12/07/19 |

**Case3: Testing audit button**

|  |
| --- |
| The PBI, the Task Description (from Sprint Log) with Unique Testing Number:  **PBI-8-Case3-01 (manual)**  **Design GUI (.xml or using Graphical Library) for Audit** |
| Team Member(s) Responsible: Berni D |
| Inputs:   * cpl.csv which is a csv file Or Opl.csv |
| Tests:   * “Run Audit” Button On Click * Test for successful run the audit process (require open a file first) |
| Outputs:   * OK The button is clickable * OK Running the Audit process * OK Result is printed in screen * OK Report is saved on default file |
| Passed or Failed: Passed |
| Date: 12/07/19 |

**PBI2 : #8 COMPLETED.**

**PBI3 : #9**

**Case1: Testing Export Audit button**

|  |
| --- |
| The PBI, the Task Description (from Sprint Log) with Unique Testing Number:  **PBI-9-Case1-01 (manual)**  **Design GUI (.xml or using Graphical Library) for Exporting** |
| Team Member(s) Responsible: Berni D |
| Inputs:   * Any csv file |
| Tests:   * “Export Audit” Button On Click * Test report successful saved at the location we specify * Test for ability to cancel selection |
| Outputs:  There is no “Export Audit” Button |
| Passed or Failed: Failed |
| Date: 12/05/19 |

|  |
| --- |
| The PBI, the Task Description (from Sprint Log) with Unique Testing Number:  **PBI-9-Case1-02 (manual)**  **Design GUI (.xml or using Graphical Library) for Exporting** |
| Team Member(s) Responsible: Berni D |
| Inputs:   * Any csv file |
| Tests:   * “Export Audit” Button On Click * Test report successful saved at the location we specify * Test for ability to cancel selection |
| Outputs:   * OK The button is clickable * OK Can select a file for saving. * OK the file select or typing can be only .txt * OK Can type the file name * OK Can cancel selection * OK Report is saved at the location we specify |
| Passed or Failed: Passed |
| Date: 12/10/19 |

**Case2: Testing Export Sharing button**

|  |
| --- |
| The PBI, the Task Description (from Sprint Log) with Unique Testing Number:  **PBI-9-Case2-01 (manual)**  Design GUI (.xml or using Graphical Library) for Exporting |
| Team Member(s) Responsible: Berni D |
| Inputs:   * Any csv file |
| Tests:   * “Export Sharing” Button On Click * Test sharing report successful saved at the location we specify * Test for ability to cancel selection |
| Outputs:   * There is no “Export Sharing” Button |
| Passed or Failed: Failed |
| Date: 12/105/19 |

|  |
| --- |
| The PBI, the Task Description (from Sprint Log) with Unique Testing Number:  **PBI-9-Case2-02 (manual)**  **Design GUI (.xml or using Graphical Library) for Exporting** |
| Team Member(s) Responsible: Berni D |
| Inputs:   * Any csv file |
| Tests:   * “Export Sharing” Button On Click * Test sharing report successful saved at the location we specify * Test for ability to cancel selection |
| Outputs:   * OK The button is clickable * OK Can select a file for saving. * OK the file selects or typing can be only .txt * OK Can type the file name * OK Can cancel selection * OK Sharing Report is saved at the location we specify |
| Passed or Failed: Passed |
| Date: 12/10/19 |

**Case5: Testing RESET button**

|  |
| --- |
| The PBI, the Task Description (from Sprint Log) with Unique Testing Number:  **PBI-8-Case5-01 (manual)**  **Design GUI (.xml or using Graphical Library) for Exporting** |
| Team Member(s) Responsible: Berni D |
| Inputs:   * Any csv file |
| Tests:   * “Reset” Button On Click * Reset All variables in voting system * Reset All textbox status in GUI |
| Outputs:   * OK The button is clickable * OK Reset status of the text box in GUI * OK Reset All variables in voting system * OK Other functions set as original values after resetting |
| Passed or Failed: Passed |
| Date: 12/10/19 |

**PBI3 : #9 COMPLETED.**

**This is tasks to provide all possible test cases to cover all potential situations. It includes all missed test from Project 1 (Unit and Integration Test) – No need to provide test log for this. As long as the tests compile and run in the machine.**

|  |
| --- |
| The PBI, the Task Description (from Sprint Log) with Unique Testing Number: **Efficiency\_Case1**  Time Audit() and check if it exceeds 1 minute for large amount of ballots |
| Team Member(s) Responsible: Hon Kwan Sin |
| Inputs:   * opl\_large\_1.csv |
| Tests:   * Test list of winner party is of correct size * Test if Audit() runs in under 1 minute for 500,000 ballots. |
| Outputs:   * OK Correct size of reading and auditing the data * OK runs in under a second |
| Passed or Failed: Pass |
| Date: 12/13/19 |
| Task Create: None |

|  |
| --- |
| The PBI, the Task Description (from Sprint Log) with Unique Testing Number: **Efficiency\_Case2**  Time Audit() and check if it exceeds 1 minute for large amount of ballots |
| Team Member(s) Responsible: Hon Kwan Sin |
| Inputs:   * cpl\_large\_1.csv |
| Tests:   * Test list of winner party is of correct size * Test if Audit() runs in under 1 minute for 500,000 ballots. |
| Outputs:   * OK Correct size of reading and auditing the data * OK runs in under a second |
| Passed or Failed: Pass |
| Date: 12/13/19 |
| Task Create: None |

|  |
| --- |
| The PBI, the Task Description (from Sprint Log) with Unique Testing Number: **Tiebreaker\_Case1**  Check if tiebreaker is correct and is random |
| Team Member(s) Responsible: Hon Kwan Sin |
| Inputs:   * opl\_tiebreaker\_1.csv |
| Tests:   * Test if list of winners are of correct size * Test if parties get the correct amount of seats when a tie occur * Test if tiebreaker is truly random for when parties are tied * Test if tiebreaker is truly random for when candidates are tied at the same time |
| Outputs:   * OK Correct Audit * OK Correct amount of seats * OK Tiebreaker is random when parties are tied * OK Tiebreaker is random when candidates are tied * OK Tiebreaker is random when both happens at the same time |
| Passed or Failed: Pass |
| Date: 12/13/19 |
| Task Create: None |

|  |
| --- |
| The PBI, the Task Description (from Sprint Log) with Unique Testing Number: **Tiebreaker\_Case2**  Check if tiebreaker is correct and is random |
| Team Member(s) Responsible: Hon Kwan Sin |
| Inputs:   * cpl\_tiebreaker\_1.csv |
| Tests:   * Test if list of winners are of correct size * Test if parties get the correct amount of seats when a tie occur * Test if tiebreaker is truly random for when parties are tied |
| Outputs:   * OK Correct Audit * OK Correct amount of seats * OK Tiebreaker is random when parties are tied |
| Passed or Failed: Pass |
| Date: 12/13/19 |
| Task Create: None |