|  |  |
| --- | --- |
| **Project Name: Project 1: Voting System Team4** | |
| **Test Stage: Unit \_√\_ System \_\_** | **Test Date: Nov 17, 2019** |
| **Test Case ID#: UC\_001** | **Name(s) of Testers: Jerry Nie, Yanjun Cui** |
| **Test Description:**  **Test users register into the system, and the register account and password will be saved into userinfo.txt file for future log in.** |  |
| **Automated: yes\_\_\_ no \_\_** **√ \_** | **Indicate where are you storing the tests (what file) and the name of the method/functions being used.**  **Generated user information file “userinfo.txt” saves in src.**  **Function used is: register** |
| **Results: Pass \_\_** **√ \_\_\_ Fail\_\_\_\_\_\_\_\_** |  |
|  |  |
| **Preconditions for Test:**  **User opens the system and select “r” for registration.** | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Step**  **#** | **Test Step**  **Description** | **Test**  **Data** | **Expected**  **Result** | **Actual**  **Result** | **Notes** |
| 1 | Test can we register successfully.  Input a valid username and password, test if it could be generated into userinfo file. If we could detect the correct username and password, then return true. Otherwise, return false. | Username: "HanzhangWu", Password: "OneShotForSeven"  userinfo.txt | Test1 pass. | Test1 pass. |  |
| 2 | Test whether the test system can prevent registrations that do not meet the requirements.  Input invalid username and password, test whether it will be written to the user information file. Then keep input invalid password, until all five times are ran out of. If we could detect the invalid username and password, then return false. Otherwise, return true. | userinfo.txt  invalid password: “OneShotForSevenSeconds” | Test2 pass. | Test2 pass. |  |
| 3 | Test whether the test system can prevent registrations that do not meet the requirements.  Input invalid username and password at first, test whether it will be written to the user information file. Then input valid password, test if the correct password could be stored.  If we can not detect the invalid username and password and we can detect the correct username with password, then return false. Otherwise, return true. | userinfo.txt  Password: "OneShotForSeven" | Test3 pass. | Test3 pass. |  |
| 4 |  |  |  |  |  |
|  |  |  |  |  |  |

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

User will log into the system, and can upload election file.

|  |  |
| --- | --- |
| **Project Name: Project 1: Voting System Team4** | |
| **Test Stage: Unit \_**√**\_ System \_\_** | **Test Date: Nov 17, 2019** |
| **Test Case ID#: TC\_002** | **Name(s) of Testers: Yanjun Cui, Jerry Nie** |
| **Test Description:**  **Test whether user logs into the system successfully.** |  |
| **Automated: yes\_\_\_ no \_\_\_**√ **\_** | **Indicate where are you storing the tests (what file) and the name of the method/functions being used.**  **Userinfo.txt saves in src directory.**  **Function: register, logout, login** |
| **Results: Pass \_\_\_\_**√ **\_\_ Fail\_\_\_\_\_\_\_\_** |  |
|  |  |
| **Preconditions for Test:**  **User runs the system.** | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Step**  **#** | **Test Step**  **Description** | **Test**  **Data** | **Expected**  **Result** | **Actual**  **Result** | **Notes** |
| 1 | Test whether user could login successfully.  If succeed, then return true. Otherwise, return false. | Username: "HanzhangWu";  password : "OneShotForSeven" | Test1 pass | Test1 pass |  |
| 2 | Test if login function could block incorrect password. Register first, then login by incorrect password five times. Test if we could not login, then return true. Otherwise, return false. | Username: "HanzhangWu";  password : "OneShotForSeven"  userinfo.txt  and some wrong passwords. | Test2 pass | Tes2 pass |  |
| 3 | Test if login function could block incorrect password and pass the correct password. Register first, then login by incorrect password for one time. Then use correct password to login. Test if we could login, then return true. Otherwise, return false. | Username: "HanzhangWu";  password : "OneShotForSeven"  userinfo.txt  and some wrong passwords. | Test3 pass | Test3 pass |  |
| 4 |  |  |  |  |  |
|  |  |  |  |  |  |

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

User can upload the file.

|  |  |
| --- | --- |
| **Project Name: Project 1: Voting System Team4** | |
| **Test Stage: Unit \_√\_ System \_\_** | **Test Date: Nov 17, 2019** |
| **Test Case ID#: TC\_003** | **Name(s) of Testers: Jerry Nie, Yanjun Cui** |
| **Test Description:**  **After generating results, to ensure information safety, user can log out the system.** |  |
| **Automated: yes\_\_ no \_\_** **√ \_** | **Indicate where are you storing the tests (what file) and the name of the method/functions being used.**  **Function: logout** |
| **Results: Pass \_\_\_** **√ \_\_ Fail\_\_\_\_\_\_\_\_** |  |
|  |  |
| **Preconditions for Test:**  **Election algorithm finished, and the file user wanted is generated successfully.** | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Step**  **#** | **Test Step**  **Description** | **Test**  **Data** | **Expected**  **Result** | **Actual**  **Result** | **Notes** |
| 1 | Test can user logout if the loginStatus is already false.  If user already logs out, they cannot log out again. | None | Test1 pass | Test1 pass |  |
| 2 | Test can user logout successfully if user already logged in the system. | None | Test2 pass | Test2 pass |  |
| 3 |  |  |  |  |  |
| 4 |  |  |  |  |  |
|  |  |  |  |  |  |

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

User logs out the system. The system exits.

|  |  |
| --- | --- |
| **Project Name: Project 1: Voting System Team4** | |
| **Test Stage: Unit \_√\_ System \_\_** | **Test Date: Nov. 17, 2019** |
| **Test Case ID#: UC\_004** | **Name(s) of Testers: Jerry Nie, Yanjun Cui** |
| **Test Description:**  **Test whether the winner list of the election can be shown on screen correctly.** |  |
| **Automated: yes\_\_\_ no \_\_** **√ \_** | **Indicate where are you storing the tests (what file) and the name of the method/functions being used.**  **Test file saves in ../testing/OPLexample.csv.**  **Methods are showWinner and openListRanking.** |
| **Results: Pass \_\_\_** **√ \_\_ Fail\_\_\_\_\_\_\_\_** |  |
|  |  |
| **Preconditions for Test:**  **The election algorithm finished successfully.** | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Step**  **#** | **Test Step**  **Description** | **Test**  **Data** | **Expected**  **Result** | **Actual**  **Result** | **Notes** |
| 1 | Test whether the function can find the winner correctly. If the showWinner function returns the same name of the real winner, then test function return true. Otherwise, return false. | ../testing/OPLexample.csv | Test1 pass. | Test1 pass. |  |
| 2 | Suppose we already knew what winner’s name is, test whether the function can find the winner correctly. | ../testing/OPLexample.csv | Test2 pass. | Test2 pass. |  |
| 3 |  |  |  |  |  |
| 4 |  |  |  |  |  |

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

User can choose what other operations (sending to media or generating the audit file) they want to do to the result and log out.

|  |  |
| --- | --- |
| **Project Name: Project 1: Voting System Team4** | |
| **Test Stage: Unit \_√\_ System \_\_** | **Test Date: Nov 17, 2019** |
| **Test Case ID#: TC\_005** | **Name(s) of Testers: Yanjun Cui, Jerry Nie** |
| **Test Description:**  **Test whether the system can show open party list election result correctly to screen.** |  |
| **Automated: yes\_\_\_ no \_** **√ \_\_** | **Indicate where are you storing the tests (what file) and the name of the method/functions being used.**  **OPLexample.csv file stored in testing directory**  **Function: openListRanking** |
| **Results: Pass \_\_** **√ \_\_\_ Fail\_\_\_\_\_\_\_\_** |  |
|  |  |
| **Preconditions for Test:**  **OPL election algorithm runs and finished.** | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Step**  **#** | **Test Step**  **Description** | **Test**  **Data** | **Expected**  **Result** | **Actual**  **Result** | **Notes** |
| 1 | Input a valid file. If test function gets the same results as expected, then return true. | ../testing/OPLexample1.csv | Test1 pass. | Test1 pass. |  |
| 2 | Input an invalid file. If error shows and variables don't obtain a value, then return true. | ../fdds | Test2 pass. | Test2 pass. |  |
| 3 |  |  |  |  |  |
| 4 |  |  |  |  |  |
|  |  |  |  |  |  |

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

User can do next operations, such as select generate audit file or send to media.

|  |  |
| --- | --- |
| **Project Name: Project 1: Voting System Team4** | |
| **Test Stage: Unit \_√\_ System \_\_** | **Test Date: Nov 18, 2019** |
| **Test Case ID#: TC\_006** | **Name(s) of Testers: Jerry Nie, Yanjun Cui** |
| **Test Description:**  **Test whether open party list algorithm can generate correct results.** |  |
| **Automated: yes\_\_\_ no \_** **√ \_\_** | **Indicate where are you storing the tests (what file) and the name of the method/functions being used.**  **Files used: OPLexample.csv, OPLexample2.csv under testing directory.**  **Funtions: openListRanking** |
| **Results: Pass \_\_** **√ \_\_\_ Fail\_\_\_\_\_\_\_\_** |  |
|  |  |
| **Preconditions for Test:**  **The file is successfully uploaded and recognized as an open party list election.** | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Step**  **#** | **Test Step**  **Description** | **Test**  **Data** | **Expected**  **Result** | **Actual**  **Result** | **Notes** |
| 1 | Test can we run a OPL ranking successfully.  Input a valid OPL CSV file, test if the voting system could be generated. | Valid path: “OPLexample.csv” | Test1 pass. | Test1 pass. |  |
| 2 | T est can we run a OPL ranking successfully.  Input a valid OPL CSV file, test if the voting system could be generated. | Valid path: “OPLexample2.csv” | Test2 pass. | Test2 pass. |  |
| 3 | Input an invalid file, test if the voting system could throw an exception. | Invalid path: “aaa” | Test3 pass. | Test3 pass. |  |
| 4 |  |  |  |  |  |
|  |  |  |  |  |  |

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

The open party list election result is generated, user can do next operation, such as generating audit file or send result to media.

|  |  |
| --- | --- |
| **Project Name: Project 1: Voting System Team4** | |
| **Test Stage: Unit \_√\_ System \_\_** | **Test Date: Nov 17. 2019** |
| **Test Case ID#: TC\_007** | **Name(s) of Testers: Jerry Nie, Yanjun Cui** |
| **Test Description:**  **Test whether the file is uploaded successfully.** |  |
| **Automated: yes\_\_\_ no \_\_** **√ \_** | **Indicate where are you storing the tests (what file) and the name of the method/functions being used.**  **Test code is stored in src directory**  **Used function: selectFile** |
| **Results: Pass \_** **√ \_\_\_\_ Fail\_\_\_\_\_\_\_\_** |  |
|  |  |
| **Preconditions for Test:**  **User already logs in the system.** | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Step**  **#** | **Test Step**  **Description** | **Test**  **Data** | **Expected**  **Result** | **Actual**  **Result** | **Notes** |
| 1 | Input an invaild file. | Invalid file: “invalid” | Test1 fail. | Test1 fail. |  |
| 2 | Input a valid file | “/testing/OPLexample.csv” | Test2 pass. | Test2 pass. |  |
| 3 |  |  |  |  |  |
| 4 |  |  |  |  |  |
|  |  |  |  |  |  |

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

Open party list algorithm or closed party list algorithm will run.

|  |  |
| --- | --- |
| **Project Name: Project 1: Voting System Team4** | |
| **Test Stage: Unit \_√\_ System \_\_** | **Test Date: Nov 18, 2019** |
| **Test Case ID#: TC\_008** | **Name(s) of Testers: Jerry Nie, Yanjun Cui** |
| **Test Description:**  **Test whether close party list algorithm can generate correct results.** |  |
| **Automated: yes\_\_\_ no \_** **√ \_\_** | **Indicate where are you storing the tests (what file) and the name of the method/functions being used.**  **Files used: CPLexample.csv under testing directory.**  **Funtions: closeListRanking, displayResultcpl** |
| **Results: Pass \_\_\_\_ Fail\_\_\_\_** **√ \_\_\_\_** |  |
|  |  |
| **Preconditions for Test:**  **The file is successfully uploaded and recognized as a close party list election.** | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Step**  **#** | **Test Step**  **Description** | **Test**  **Data** | **Expected**  **Result** | **Actual**  **Result** | **Notes** |
| 1 | Test can we run a OPL ranking successfully.  Input a valid OPL CSV file, test if the voting system could be generated. | Valid path: “OPLexample.csv” | Test1 pass. | Test1 fail. |  |
| 2 | Input an invalid file, test if the voting system could throw an exception. | Invalid path: “aaa” | Test2 pass. | Test2 pass. |  |
| 3 |  |  |  |  |  |
| 4 |  |  |  |  |  |
|  |  |  |  |  |  |

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

The open party list election result is generated, user can do next operation, such as generating audit file or send result to media.

|  |  |
| --- | --- |
| **Project Name: Project 1: Voting System Team4** | |
| **Test Stage: Unit \_√\_ System \_\_** | **Test Date: Nov 18 2019** |
| **Test Case ID#: UC\_009** | **Name(s) of Testers: Jerry Nie, Yanjun Cui** |
| **Test Description:**  **Test if the audit file generated successfully.** |  |
| **Automated: yes\_\_\_ no \_\_** **√ \_** | **Indicate where are you storing the tests (what file) and the name of the method/functions being used.**  **Generated audit file “auditopl.txt” saves in src.**  **Used file: OPLexample2.csv under testing directory.**  **Function used is: openListRanking, auditfileopl** |
| **Results: Pass \_\_** **√ \_\_\_ Fail\_\_\_\_\_\_\_\_** |  |
|  |  |
| **Preconditions for Test:**  **The election algorithm finished and user choose to generate audit file.** | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Step**  **#** | **Test Step**  **Description** | **Test**  **Data** | **Expected**  **Result** | **Actual**  **Result** | **Notes** |
| 1 | Test can we generate audit file successfully.  If we could detect the audit file, then return true. Otherwise, return false. | ../testing/OPLexample2.csv  auditopl.txt | Test1 pass. | Test2 pass |  |
| 2 | Test can we generate throw an exception successfully. | ./testing/OPLexample2.csv | Test2 pass. | Test2 pass | Before test, we must block the function working correctly, which needs to be finished outside of code. |
| 3 |  |  |  |  |  |
| 4 |  |  |  |  |  |
|  |  |  |  |  |  |

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

User can log out the system.

|  |  |
| --- | --- |
| **Project Name: Project 1: Voting System Team4** | |
| **Test Stage: Unit \_√\_ System \_\_** | **Test Date: Nov 18 2019** |
| **Test Case ID#: UC\_0010** | **Name(s) of Testers: Jerry Nie, Yanjun Cui** |
| **Test Description:**  **Test if the sending-to-media file for Open party list election generated successfully.** |  |
| **Automated: yes\_\_\_ no \_\_** **√ \_** | **Indicate where are you storing the tests (what file) and the name of the method/functions being used.**  **Generated audit file “mediaopl.txt” saves in src.**  **Used file:**  **Function used is: OPLexample2.csv under testing directory** |
| **Results: Pass \_\_** **√ \_\_\_ Fail\_\_\_\_\_\_\_\_** |  |
|  |  |
| **Preconditions for Test:**  **The election algorithm finished, and user choose to generate sending to media file.** | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Step**  **#** | **Test Step**  **Description** | **Test**  **Data** | **Expected**  **Result** | **Actual**  **Result** | **Notes** |
| 1 | Test can we generate sending to meida file successfully.  If we could detect the sending to media file, then return true. Otherwise, return false. | ../testing/OPLexample2.csv  mediaopl.txt | Test1 pass. | Test2 pass |  |
| 2 | Test can we generate throw an exception successfully. | ../testing/OPLexample2.csv | Test2 pass. | Test2 pass | Before test, we must block the function working correctly, which needs to be finished outside of code. |
| 3 |  |  |  |  |  |
| 4 |  |  |  |  |  |
|  |  |  |  |  |  |

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

User can log out the system.

|  |  |
| --- | --- |
| **Project Name: Project 1: Voting System Team4** | |
| **Test Stage: Unit \_√\_ System \_\_** | **Test Date: Nov 18 2019** |
| **Test Case ID#: UC\_011** | **Name(s) of Testers: Jerry Nie, Yanjun Cui** |
| **Test Description:**  **Test if the sending-to-media file for Close party list election generated successfully.** |  |
| **Automated: yes\_\_\_ no \_\_** **√ \_** | **Indicate where are you storing the tests (what file) and the name of the method/functions being used.**  **Generated audit file “mediacpl.txt” saves in src.**  **Used file: CPLexample.csv under testing directory**  **Function used is: closeListRanking** |
| **Results: Pass \_\_** **√ \_\_\_ Fail\_\_\_\_\_\_\_\_** |  |
|  |  |
| **Preconditions for Test:**  **The election algorithm finished, and user choose to generate sending to media file.** | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Step**  **#** | **Test Step**  **Description** | **Test**  **Data** | **Expected**  **Result** | **Actual**  **Result** | **Notes** |
| 1 | Test if the failure case generated successfully. | ../testing/CPLexample.csv | Test1 pass. | Test2 pass |  |
| 2 | Test can we generate sending to meida file successfully.  If we could detect the sending to media file, then return true. Otherwise, return false. | ../testing/CPLexample.csv  mediacpl.txt | Test2 pass. | Test2 pass. |  |
| 3 |  |  |  |  |  |
| 4 |  |  |  |  |  |
|  |  |  |  |  |  |

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

User can log out the system.

|  |  |
| --- | --- |
| **Project Name:  Project 1: Voting System                                                                           Team4** | |
| **Test Stage:   Unit \_\_   System \_√\_** | **Test Date:  Nov 18, 2019** |
| **Test Case ID#:  ST\_001** | **Name(s) of Testers:  Yanjun Cui** |
| **Test Description:**  **Test whether system runs open party election correctly from log in / register, then generate audit file, show winners and send the results to media, then log out.** |  |
| **Automated:   yes\_\_\_ no \_\_√\_** | **Indicate where are you storing the tests (what file) and the name of the method/functions being used.**  **Function used: Expected closeListRanking**  **File used: OPLexample.csv under testing directory** |
| **Results:   Pass \_\_\_\_\_\_       Fail\_\_\_√\_\_\_\_\_** |  |
|  |  |
| **Preconditions for Test:**  **Run “javac voting\_system.java”, and then “java voting\_system” at terminal.** | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Step**  **#** | **Test Step**  **Description** | **Test**  **Data** | **Expected**  **Result** | **Actual**  **Result** | **Notes** |
| **1** | **Register into the system.** | **r** | **“Your are now in the registration page**  **Please set your username:”** | **“Your are now in the registration page**  **Please set your username:”** |  |
| **2** | **Input user name and password.** | **Username: 123**  **password: 123456** | **“Your registration was successful!**  **You have been redirected and is now logged in!**  **Please enter the name of the csv file:”** | **“Your registration was successful!**  **You have been redirected and is now logged in!**  **Please enter the name of the csv file:”** |  |
| **3** | **Upload file.** | **../testing/OPLexample.csv** | **“The type of voting is open list ranking**  **The number of seats is 3**  **The number of ballots is 10**  **The number of candidates is 6**  **The winner is Foster who is from D and gets 3 ballots.**  **The audit file has been successfully generated!**  **Enter 1: Show winners**  **Enter 2: Get audit file**  **Enter 3: Send to media**  **Enter 4: Log out”** | **“The type of voting is open list ranking**  **The number of seats is 3**  **The number of ballots is 10**  **The number of candidates is 6**  **The winner is Foster who is from D and gets 3 ballots.**  **The audit file has been successfully generated!**  **Enter 1: Show winners**  **Enter 2: Get audit file**  **Enter 3: Send to media**  **Enter 4: Log out”** |  |
| **4** | **Input “2”** | **2** | **“The audit file has been successfully generated!”** | **“The audit file has been successfully generated!”** | **File “auditopl.txt” generated under src directory.** |
| **5** | **Press “enter”**  **Automatic logged out and exit the system.** |  |  |  | **In here, the system automatically logged out, so we cannot do other operations. If we want to send the results to media, we need to run the system again.** |

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

**Winner shows while running the election algorithm, and an audit file generated.**

|  |  |
| --- | --- |
| **Project Name:  Project 1: Voting System                                                                           Team4** | |
| **Test Stage:   Unit \_\_   System \_√\_** | **Test Date:  Nov 18, 2019** |
| **Test Case ID#:  ST\_002** | **Name(s) of Testers:  Yanjun Cui** |
| **Test Description:**  **Test whether system runs closed party election correctly from log in / register, then generate audit file, show winners and send the results to media, then log out.** |  |
| **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:**  **Run “javac voting\_system.java”, and then “java voting\_system” at terminal.** | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Step**  **#** | **Test Step**  **Description** | **Test**  **Data** | **Expected**  **Result** | **Actual**  **Result** | **Notes** |
| **1** | **Register into the system.** | **r** | **“Your are now in the registration page**  **Please set your username:”** | **“Your are now in the registration page**  **Please set your username:”** |  |
| **2** | **Input user name and password.** | **Username: 123**  **password: 123456** | **“Your registration was successful!**  **You have been redirected and is now logged in!**  **Please enter the name of the csv file:”** | **“Your registration was successful!**  **You have been redirected and is now logged in!**  **Please enter the name of the csv file:”** |  |
| **3** | **Upload file.** | **../testing/CPLexample/.csv** | **“The audit file has been successfully generated!**  **The type of voting is close list ranking**  **The number of parties is 4**  **The number of seats is 7**  **The number of candidates is 16**  **The number of ballots is 13**  **Party G has 4 votes and 3 seats. In party G, Jones, Smith, Lewis got elected.**  **Party I has 1 vote and 1 seat. In party I, Perez got elected.**  **Party D has 3 votes and 1 seat. In party D, Pike got elected.**  **Party R has 5 votes and 2 seats. In party R, Deutsch, Wong got elected.**  **Enter 1: Send to media**  **Enter 2: Log out”** | **“The audit file has been successfully generated!**  **The type of voting is close list ranking**  **The number of parties is 4**  **The number of seats is 7**  **The number of candidates is 16**  **The number of ballots is 13**  **Party G has 4 votes and 2 seats. In party G, Jones, Smith got elected.**  **Party I has 1 vote and 0 seat. In party I, no one got selected.**  **Party D has 3 votes and 2 seat. In party D, Pike and Foster got elected.**  **Party R has 5 votes and 3 seats. In party R, Deutsch, Wong and Walters got elected.**  **Enter 1: Send to media**  **Enter 2: Log out”** | **Audit.txt generated under src directory.** |
| **4** | **Generate audit file.** | **1** | **“The media file has been successfully generated!”** | **“The media file has been successfully generated!”** | **Mediacpl.txt generated under src firectory.** |
| **5** | **Log out** | **2** | **“You are now logged out!”** | **“You are now logged out!”** |  |

|  |  |
| --- | --- |
| **Project Name: Project 1: Voting System Team4** | |
| **Test Stage: Unit \_√\_ System \_\_** | **Test Date: Dec 6, 2019** |
| **Test Case ID#: UC\_012** | **Name(s) of Testers: Yangjiawen Xu, Hanzhang Wu** |
| **Test Description:**  **Test graphical interface work properly and users search through the file structure and select a file if needed.** |  |
| **Automated: yes\_\_\_ no \_\_** **√ \_** | **Indicate where are you storing the tests (what file) and the name of the method/functions being used.**  **Function: csv\_search** |
| **Results: Pass \_\_** **√ \_\_\_ Fail\_\_\_\_\_\_\_\_** |  |
|  |  |
| **Preconditions for Test:**  **User runs the system and select “r” for registration and set username and password.** | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Step**  **#** | **Test Step**  **Description** | **Test**  **Data** | **Expected**  **Result** | **Actual**  **Result** | **Notes** |
| 1 | Test whether user can select the exact csv file through graphical interface.  User can type in “2” and then click “go to” to go to the testing directory. Then select “0” and then click “select”. | Enter “2” Then enter “0” | The csv file has been selected! | The csv file has been selected! | Test 1 pass |
| 2 | Test whether user will be given a message to inform he/she that his/her action is wrong.  Enter the number that is a file but click “go to” | Enter 4 which is a file | The file selected is not a directory! | The file selected is not a directory! | Test 2 pass |
| 3 | Test whether user will be given a message to inform he/she that his/her action is wrong.  Enter the number that is a directory but click “select” | Enter 6 which is a directory | The file selected is not a csv file! | The file selected is not a csv file! | Test 3 pass |

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

The algorithm is run, and user can be prompted to select the results form.

|  |  |
| --- | --- |
| **Project Name: Project 1: Voting System Team4** | |
| **Test Stage: Unit \_√\_ System \_\_** | **Test Date: Dec 6, 2019** |
| **Test Case ID#: UC\_013** | **Name(s) of Testers: Yangjiawen Xu, Yanjun Cui** |
| **Test Description:**  **Test graphical interface work properly and users can rename and save the file after the audit file is generated.** |  |
| **Automated: yes\_\_\_ no \_\_** **√ \_** | **Indicate where are you storing the tests (what file) and the name of the method/functions being used.**  **Function: myjframe1** |
| **Results: Pass \_\_** **√ \_\_\_ Fail\_\_\_\_\_\_\_\_** |  |
|  |  |
| **Preconditions for Test:**  **User logs in the system, select the csv file and the algorithm is run. User can then choose the results form.** | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Step**  **#** | **Test Step**  **Description** | **Test**  **Data** | **Expected**  **Result** | **Actual**  **Result** | **Notes** |
| 1 | Test whether user can rename and save the generated audit file through graphical interface.  User can type in the desired file name and save the file to the desired folder. | aaa  Desktop | Audit file is successfully generated and saved, you can do other operations.  The file appears at my Desktop. | Audit file is successfully generated and saved, you can do other operations.  The file appears at my Desktop. | Test 1 pass |
| 2 |  |  |  |  |  |
| 3 |  |  |  |  |  |

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

The audit file is generated and saved to the desired folder with desired name. The user can choose other result forms.

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

**Winner shows while running the election algorithm, and an audit file and media file generated**

**Project Name:**  The project #, name of your system, and the team#

**Test Stage:** Indicate whether it is a unit test or a system test.

**Test Date:**  The date the test was performed.

**Test Case ID#:**  A unique ID is required. Decide on a naming convention and use numbering. Example: Ballot\_Shuffle\_1

**Name(s) of Testers:** List the names of anyone involved in running this test case.

**Test Description:**  Describe briefly the test objective.

**Automated:**  Indicate if the test is completely automated or being checked manually. (If you have methods running the tests and checking results, select “yes”. If you are manually checking results, indicate manual by selecting the “no.”)

**Results:** Indicate if the test passed or failed.

**Step #:** You will be listing the test steps in order. This number is the step number in the process.

**Test Step Description:** Details of the test step.

**Test Data:** What the test data will be for this step. Be clear on what the input data will be. If using a specific file, be clear on the name.

**Expected Result:** What result are you expecting from the program component or system.

**Actual Result:** What result were returned based on the test.

**Post condition for Test:** What will be true after the test has been run? Has the state of the system changed in any way?

**Notes:** Comments and notesfor you and your team members.