## **Software Testing Documentation**

### **Comprehensive Database Tests**

Each of the following tests were conducted incrementally, starting with an empty database file. The success of one test determined the starting criteria for the next. All database tests were written in a single test file, data\_test.py, and utilizes data.py, file.py, and url.py.

**Author of data\_test.py:** Tim Yingling

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test #** | **Description** | **Preconditions / Input Values** | **Expected Output** | **Actual Output** | **P/F Criteria** | **Comments** |
| 1 | Test to see if the newly created database file has empty tables | Database file and tables for files and urls are created, use print\_all\_files and print\_all\_urls methods on empty tables. | Nothing is printed except a newline character after the program declares that it is printing the contents of each table | Nothing is printed except a newline character after the program declares that it is printing the contents of each table | Actual = Expected | Pass |
| 2 | Test to see if store\_file and store\_url methods insert the passed objects into an empty database correctly.  Store\_file is passed a File object and stores all of the member variables into the files table.  Store\_url is passed a URL object and stores all of the member variables into the urls table. | f1 = File(“Tim”, “Mark”, “Clement”, “Jenna”)  u1 = URL(  domain\_name = “Google”  )  store\_file(f1) and store\_url(u1) are called with an empty database | Two tuples displayed showing the contents of f1 and u1 respectively:  (‘Tim’, ‘Mark’, ‘Clemont’, ‘Jenna’, 0, 0, 0, 0, ‘’, ‘’)  (‘’, ‘’, ‘’, ‘’, 0, ‘’, ‘’, ‘’, ‘Google’, 0) | Two tuples displayed showing the contents of f1 and u1 respectively:  (‘Tim’, ‘Mark’, ‘Clemont’, ‘Jenna’, 0, 0, 0, 0, ‘’, ‘’)  (‘’, ‘’, ‘’, ‘’, 0, ‘’, ‘’, ‘’, ‘Google’, 0) | Actual = Expected | Pass |
| 3 | Test to see if search\_file and search\_url methods return None if the item is not found in the database.  Search\_file is passed a string and is checked against any of the hash types stored to find a match. Search\_url is passed a string that is checked against domain names stored to find a match. | Database has f1 and u1 in their respective tables.  string1 = “Biruk  string2 = “Amazon”  search\_file( string1)  and  search\_url( string2) are called on the database | Both functions return None and “File not found” and “URL not found” are displayed in the terminal | Both functions return None and “File not found” and “URL not found” are displayed in the terminal | Actual = Expected | Pass |
| 4 | Test to see if store\_file and store\_url methods insert the passed objects into an existing database correctly | f2 = File(“Biruk”, “Scott”, “Nick”,  “Kyle”)  u2 = URL( domain\_name = “Amazon”  )  store\_file(f2) and store\_url(u2) are called on the database. | Four tuples displayed showing the contents of f1, f2, u1, and u2 respectively:  (‘Tim’, ‘Mark’, ‘Clemont’, ‘Jenna’, 0, 0, 0, 0, ‘’, ‘’)  (‘Biruk’, ‘Scott’, ‘Nick’, ‘Kyle’, 0, 0, 0, 0, ‘’, ‘’)  (‘’, ‘’, ‘’, ‘’, 0, ‘’, ‘’, ‘’, ‘Amazon’, 0) | Four tuples displayed showing the contents of f1, f2, u1, and u2 respectively:  (‘Tim’, ‘Mark’, ‘Clemont’, ‘Jenna’, 0, 0, 0, 0, ‘’, ‘’)  (‘Biruk’, ‘Scott’, ‘Nick’, ‘Kyle’, 0, 0, 0, 0, ‘’, ‘’)  (‘’, ‘’, ‘’, ‘’, 0, ‘’, ‘’, ‘’, ‘Amazon’, 0) | Actual = Expected | Pass |
| 5 | Test to see if File and URL objects are returned correctly after calling search\_file and search\_url on items that exist in the tables | Database has f1, f2, u1, and u2 in their respective tables.  string1 = “Biruk  string2 = “Amazon”  search\_file( string1)  and  search\_url( string2) are called on the database | Search\_file( string1) returns File object similar to f2 and search\_url( string2) returns URL object similar to u2.  “File found, md5 hash = Biruk” and “URL found, domain name = Amazon” displayed to terminal | Search\_file( string1) returns File object similar to f2 and search\_url( string2) returns URL object similar to u2.  “File found, md5 hash = Biruk” and “URL found, domain name = Amazon” displayed to terminal | Actual = Expected | Pass |
| 6 | Test to see if search\_file will find an existing file using a different hash type than md5 which is the first hash type stored in the table | Database has f1, f2, u1, and u2 in their respective tables.  String = “Scott”  search\_file( String) called on the database | Search\_file( String) returns a File object similar to f2  “File found, md5 hash = Biruk” displayed to terminal | Search\_file( String) returns a File object similar to f2  “File found, md5 hash = Biruk” displayed to terminal | Actual = Expected | Pass |
| 7 | Test to see if search\_URL can pull from the database and get the appropriate engines and results | Database has f1, f2, u1, and u2 in their respective tables.  string = URL | search\_URL(string) returns File object with correct engines and results | search\_URL(string) returns File object with incorrect engines and results | Actual = Expected | Fail  For the URL class, the parameters “engines” and “results” needed to be switched because the database was retrieving the correct info but storing them in the wrong member variables |
| 8 | Test to see if database adds URLs that have the same domain but are different sites | Database has f1, f2, u1, and u2 in their respective tables.  database has site with same domain as passed-in URL  string = URL | search\_URL(string) returns None when the string has the same domain as another in the database but is a different site; store\_URL is called | search\_URL(string) returns a File object because it searches the database based on the domain which match between the passed-in string and the string in the database | Actual = Expected | Fail  Need to search database based on what the clean user input is and not domain name because different sites may have the same domain name, but they are technically different sites so they must be scanned and added separately |

### **Comprehensive Frontend Blackbox Tests**

Each of the following tests were conducted incrementally and independently. Each feature on the front-end was tested thoroughly with their respective output described below.

**Author of test:** Mark Biegel

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test #** | **Description** | **Input Values** | **Expected Output** | **Actual Output** | **P/F Criteria** | **Comments** |
| **1** | Front-end accounts for all types of hash and URL malicious status including clean, suspicious, and malicious | A URL and hash of the following malicious statuses: clean, suspicious, and malicious | A checkmark for a clean hash  A caution sign for suspicious hash  An x-mark for a malicious hash | A checkmark for a clean hash  An x-mark for a malicious hash | Actual = Expected | Fail  VirusTotal returned a “suspicious” flag, but the frontend doesn’t account for it. |
| 2 | Front-end account s for invalid hash or URL and a query limit of 500 new queries per day | Non-urls, non-hashes, or more than 500 new queries | Rendered “invalid\_request.html” | Rendered “invalid\_request.html” | Actual = Expected | Pass |
| 3 | Front-end redirects to “help.html” when “Help” button is clicked | “Help” button on “homescreen.html” is clicked | Rendered “help.html” | Rendered “help.html” | Actual = Expected | Pass |
| 4 | Front-end redirects to “about.html” when “About” button is clicked | “About” button on “homescreen.html” is clicked | Rendered “about.html” | Rendered “about.html” | Actual = Expected | Pass |
| 5 | Front-end redirects to “homepage.html” whenever the Threat Detector logo is clicked except for on the homepage | Logo icon on any .html page is clicked | Rendered “homepage.html” | Rendered “homepage.html” | Actual = Expected | Pass |
| 6 | Front-end redirects to the second page html templates when “Search” button is clicked on the homepage or on the second page templates. | Search button is clicked on homepage or second page templates | Rendered “second\_page\_hash\_malware\_negative”, “second\_page\_hash\_malware\_positive”,  “second\_page\_URL\_malware\_negative”,  “second\_page\_URL\_malware\_positive” | Rendered “second\_page\_hash\_malware\_negative”, “second\_page\_hash\_malware\_positive”,  “second\_page\_URL\_malware\_negative”,  “second\_page\_URL\_malware\_positive” | Actual = Expected | Pass |
| 7 | Front-end displays the Detection tab when click on either of the second page templates | Detection button is clicked on the second page templates | Detection tab is displayed with the list of vendors and detection status | Detection tab is displayed with the list of vendors and detection status | Actual = Expected | Pass |
| 8 | Front-end displays the Metadata tab for a URL or hash when click on either of the second page templates | Metadata button is clicked on the second page templates | Metadata tab is displayed with the list of statistics and values for either a URL or hash | Metadata tab is displayed with the list of statistics and values for either a URL or hash | Actual = Expected | Pass |
| 9 | Front-end displays the Malicious Status tab when click on either of the second page templates | Malicious Status button is clicked on the second page templates | Malicious Status tab is displayed with a description of the malicious status for either a URL or hash | Malicious Status tab is displayed with a description of the malicious status for either a URL or hash | Actual = Expected | Pass |