­Prevent, Mitigate, and Recover (PMR) Insight

Collective Knowledge System (PICK)

Test plan

Version <2.1>

4/28/2020

Document Control

Approval

The Guidance Team and the customer shall approve this document.

Document Change Control

|  |  |
| --- | --- |
| Initial Release: | 1.0 |
| Current Release: |  |
| Indicator of Last Page in Document: | \* |
| Date of Last Review: | 4/28/2020 |
| Date of Next Review: | 4/18/2020 |
| Target Date for Next Update: | 4/22/2020 |

Distribution List

This following list of people shall receive a copy of this document every time a new version of this document becomes available:

Guidance Team Members:

Dr. Gates

Dr. Salamah

Dr. Roach

Steven Roach

Jake Lasley

Customer:

Dr. Oscar Perez

Vincent Fonseca

Herandy Denisse Vazquez

Baltazar Santaella

Florencia Larsen

Erick De Nava

Software Team Members:

Elizabeth Barragan

Jose A. Leon Cordero

Yamel E. Hernandez

Manuel Delgado

Change Summary

The following table details changes made between versions of this document

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Modifier | Description |
| 1.0 | 4/16/2020 | Manuel | Creation of Document & Document Control |
| 1.1 | 4/16/2020 | Elizabeth | Added a few database tests cases (Sec 3) and Test Suite DB3 (Sec 4) |
| 1.2 | 4/16/2020 | Yamel | Added section 1.5, added Ingestion test cases to section 3, & Ingestion Test Suit in Section 4 |
| 1.3 | 4/16/2020 | Antoine | Added tables 3 and 4 in section 3. |
| 1.4 | 4/16/2020 | Manuel | Added Vector Subsystem to section 3 as well as intro |
| 1.5 | 4/22/2020 | Elizabeth | Added Sec.1.1 ,1.2 |
| 1.6 | 4/27/2020 | Elizabeth | Updated Sec. 3: DB Test Suite,  Added DB Test 1, 2, Added DB Test 1,2 to the Appendix, Added Sec 1.6 |
| 1.7 | 4/27/2020 | Yamel | Added Sec 4 intro & deleted Sec 7 Other Sections |
| 1.8 | 4/28/2020 | Manuel | Added section 1.3 and test section Graph and Vector 4.9 to 4.19 |
| 1.9 | 4/28/2020 | Yamel | Added to the appendix for Ing1 & Ing 2 |
| 2.0 | 4/28/2020 | Antoine | Added 1.5, deleted table 3 in section 3, edited new table 3 in section 3, renamed section 4, added 4.12, and edited section 5 |
| 2.1 | 4/28/2020 | Elizabeth | Added DB Test 4-7, Added DB Test 4-5 in Appendix |

Note: The template presented in this document was taken from:

Donaldson, S., and S. Siegel, *Successful Software Development*. Upper Saddle River, NJ: Prentice Hall, 2001, pp. 321-323.

Note: The template presented in this document was taken from: Donaldson, S., and S. Siegel, *Successful Software Development*. Upper Saddle River, NJ: Prentice Hall, 2001, pp. 321-323 and modified by Humberto Mendoza and Steve Roach.

Supplementary information is from:

Pfleeger, S. *Software Engineering, Theory and Practice*. Upper Saddle River, NJ: Prentice Hall, 1998, p. 365.

Table of Contents

[Document Control ii](#_Toc39010694)

[Approval ii](#_Toc39010695)

[Document Change Control ii](#_Toc39010696)

[Distribution List ii](#_Toc39010697)

[Change Summary ii](#_Toc39010698)

[1. Introduction 1](#_Toc39010699)

[1.1. Purpose 1](#_Toc39010700)

[1.2. Scope 1](#_Toc39010701)

[1.3. System Overview 1](#_Toc39010702)

[1.4. Suspension and Exit Criteria 1](#_Toc39010703)

[1.5. Document Overview 2](#_Toc39010704)

[1.6. References 2](#_Toc39010705)

[2. Test Items and Features 3](#_Toc39010706)

[2.1. Components to be Tested 3](#_Toc39010707)

[2.2. Components not Implemented 3](#_Toc39010708)

[3. Testing Approach 4](#_Toc39010709)

[4. Test Breakdown 6](#_Toc39010710)

[4.1. Test DB 1 6](#_Toc39010711)

[4.2. Test DB 2 7](#_Toc39010712)

[4.3. Test DB 3 8](#_Toc39010713)

[4.4. Test DB 4 9](#_Toc39010714)

[4.5. Test DB 5 10](#_Toc39010715)

[4.6. Test DB 6 11](#_Toc39010716)

[4.7. Test DB 7 12](#_Toc39010717)

[4.8. Test Ing 1 13](#_Toc39010718)

[4.9. Test Ing 2 14](#_Toc39010719)

[4.10. Test Ing 3 15](#_Toc39010720)

[4.11. Test Ing 4 16](#_Toc39010721)

[4.12. Test EC 1 17](#_Toc39010722)

[4.13. Test GC1 18](#_Toc39010723)

[4.14. Test GC2 19](#_Toc39010724)

[4.15. Test GC3 20](#_Toc39010725)

[4.16. Test GC4 21](#_Toc39010726)

[4.17. Test GC5 21](#_Toc39010727)

[4.18. Test GC6 22](#_Toc39010728)

[4.19. Test GC7 22](#_Toc39010729)

[4.20. Test GC8 23](#_Toc39010730)

[4.21. Test VS1 24](#_Toc39010731)

[5. Test Schedule 26](#_Toc39010732)

[6. Appendix 28](#_Toc39010733)

# Introduction

## Purpose

The purpose of this document is to describe the kinds of tests that are ran in the system and provide guidance through specification/documentation for a tester to run tests accordingly for all parts of the system. In this test plan, unit testing will be performed in the subsystems with some of the subsystems integrated. Moreover, this document provides structure for organizing, scheduling, and managing of the tests ran in system. By providing analysis of the tests, this document allows the analysis of test coverage, identifies repetition of test, and improves test reproducibility.

## Scope

This test plan encompasses the release 1.0 including the following versions:

* V0.1: Executable GUI
* V0.2: Adding Ingestion Subsystem
* V0.3: Adding Vector/Graph Subsystem
* V0.4: Adding Database Subsystem

## System Overview

Testing will be done at system level, ensuring system integration was done correctly. Testing each feature as part of the whole system. Operations the system will contain are the ones that will be tested and are limited to the components implemented. Ingestion of log files into the Pick system, user graphical interface of the ability of the system, creation of a vector and its graph and finally permanent storage in the mongo database of any information inputted into the system

## Suspension and Exit Criteria

Testing for the PICK system is crucial in the eyes of the clients. Because it is a system that aims to optimize the goals of our clients, we define the following strict suspension and exit criteria for testing of the PICK system.

Suspension Criteria

Upon failure of 5 critical test cases, the execution of testing will be suspended to address the failures. This definition is motivated by the nature of these tests – most critical tests have implications in various parts of the system, therefore if one critical test fails, the failure of another critical test is almost ensured and an indication of a major issue of the PICK system.

Exit Criteria

Testing of the PICK system is to be finalized based on a compound test pass rate. Of this compound test pass rate, all critical tests must pass, while 80% of the non-critical must pass. These criteria are motivated by the implemented features of the PICK system – critical features of the PICK system impact the efficiency of the Lethality, Survivability, and HSI Directorate (LSH) to conduct their analysis, and the inability of a critical feature to pass a test will have direct implications on efficiency and the performance of other system features. Though non-critical features of the PICK system may also impact efficiency of LSH in minor ways, potential failures do not impact the overall goal of the system or other features.

## Document Overview

This document outlines how we will implement and test our system. Section 2 will describe the test items and features that will be tested of those items, which will include a table or a class diagram for visualization. Section 3 will discuss how we will approach testing the items and features from section 2 in tabular format. The tables will list the type of test, it’s objective, and the level of criticality of the test. Section 4 will document the tests in a table per test. The table will have the test number, its status, title, approach, who performed the test, and when the test was concluded. Section 5 will list and describe the test schedule in a table, information on it will include a date, who will be performing the test and a short description. Section 6, the Appendix, will depict and illustrate outputs for analysis.

## References

[1] SDD\_team5.docx

[2] Team5SCM.docx

[3] 22c\_Test\_Plan.ppt

[4] SRS\_v7.docx

# Test Items and Features

In this section, components, classes, functions and features to be tested are defined. Additionally, we define those which are not to be tested.

## Components to be Tested

The following is a list of the components and their functions and features that will be tested:

* Real-World Objects
  + Data transformation
  + Instantiation
  + JSON representation
* Signal Controller
  + Signals handling of all models
* Database
  + Splunk
* Ingestion Subsystem
  + Splunk
  + Cleansing
    - Extra spaces
    - Non-Ascii Characters
  + Enforcement Action Report & Error Handling
* Graphical User Interface
  + Window frame adjustment
  + Button events
  + Input processing
  + Model representation
  + Error handling
  + Default security settings
* Vector Subsystem
  + Graph
  + Node Table

## Components not Implemented

The following is a list of features that will not be tested:

* Version Control
* Undo/Redo action
* Audio transcriber
* OCR
* Networking

# Testing Approach

The section demonstrates in a table format the types of test that will be performed on the system. The table provides a description for types of tests such as but not limited to exercising system functions, approximating operational use, or stressing the system to find out limitations. Each component or subsystem to be tested has a description of the type of test, test case identifier for each objective and its importance level.

Table 1: Database Test Suite

|  |  |  |
| --- | --- | --- |
| **TEST SUITE DATABASE** | | |
| **Description of Test Suite** | **This test suite has test cases for the functionality of the Database subsystem** | |
| **Test Case Identifier** | **Objective** | **Criticality** |
| DB 1 | Test the mapping/insertion between the Event Configuration object with the document schema design. | Critical |
| DB 2 | Test the mapping/insertion between the Vector object with the document schema design. | Critical |
| DB 3 | Test to update an Event Configuration object in the DB. | Critical |
| DB 4 | Test the pull/search of an Event Configuration object from the DB, reflecting it in the GUI. | Critical |
| DB 5 | Test the deletion of a Vector object from the DB. | Critical |
| DB 6 | Test the pull/search of a Vector object from the DB. | Critical |
| DB 7 | Test to update a Vector along its components in the DB. | Critical |

Table 2: Ingestion Test Suite

|  |  |  |
| --- | --- | --- |
| **TEST SUITE INGESTION** | | |
| **Description of Test Suite** | **This test suite will have appropriate test cases for the functionality of the ingestion subsystem and process** | |
| **Test Case Identifier** | **Objective** | **Criticality** |
| Ing 1 | Test for cleansing for additional spaces, lines, & binary data | Critical |
| Ing 2 | Test parsing in Splunk | Critical |
| Ing 3 | Test for error messages in enforcement action report & error class | Critical |
| Ing 4 | Test ingestion process to make sure log files get flagged for the state that they are in | Critical |

Table 3: Event Configuration Test Suite

|  |  |  |
| --- | --- | --- |
| **TEST SUITE EVENT CONFIGURATION** | | |
| **Description of Test Suite** | **This test suite defines test cases for probing data transformation and representation of the event configuration model** | |
| **Test Case Identifier** | **Objective** | **Criticality** |
| EC 1 | Test event configuration instantiation with user input data | Critical |

Table 4: Graph Component

|  |  |  |
| --- | --- | --- |
| **TEST SUITE GRAPH COMPONENT** | | |
| **Description of Test Suite** | **This test suite defines test cases for probing the creation and functionality of the Graph Component** | |
| **Test Case Identifier** | **Objective** | **Criticality** |
| GC1 | Test creation of a Graph | Critical |
| GC2 | Test creation of a Node | Critical |
| GC3 | Test creation of a Relationship | Critical |
| GC4 | Test functionality of movement for a Node | Normal |
| GC5 | Test functionality of movement for a Relationship | Normal |
| GC6 | Test deletion of a Node | Critical |
| GC7 | Test editing of a Node | Critical |
| GC8 | Test editing of a Relationship | Critical |

Table 5: Vector Subsystem Test Suite

|  |  |  |
| --- | --- | --- |
| **TEST SUITE VECTOR SUBSYSTEM** | | |
| **Description of Test Suite** | **This test suite defines test cases for probing the creation and functionality of the Vector Subsystem** | |
| **Test Case Identifier** | **Objective** | **Criticality** |
| VS1 | Test editing correlation between a Node and a Significant Log Entry | Critical |

# Test Breakdown

This section documents the tests performed on the system. Each subsection in here will encapsulate the type of test performed, who performed it, how they approached it, the method used, and nature of the test. These tests can be found from the test suites in section 3.

## Test DB 1

**Objective:** The objective of this test is to verify that the Event Configuration object will be mapped with the document schema design once an EC object is inserted.

**Notes:** The Event Configuration (EC) object is not yet integrated with the GUI, unable to obtain user input, the EC attributes need to be given when the class is invoked.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test No.: DB1 | | | | Current Status: Passed | | |
| Test title: Test the mapping/insertion between the Event Configuration object with the DB document structure. | | | | | | |
| Testing approach: This test will be conducted by utilizing the Event Configuration class that will contain it’s given attributes, the mapping of the EC object will be reflected in the GUI Database tool, “Mongo Compass.” | | | | | | |
| STEP  1 | OPERATOR ACTION  Open up the command prompt, run the command “mongod” to connect to the MongoDB server.  Open up MongoDB Compass, click on “Connect” to be connected to the local host, select the “PICKDB” database. Select the “Event Configuration” Collection. | PURPOSE  Initial Condition | | | EXEPCTED RESULTS  The Event Configuration collection does not contain any documents. | COMMENTS  The “PICKDB” database along with the “Event Configuration” collection is created manually through the MongoDB Compass tool. |
| 2 | Run the “DBTest.py” script located in the directory “Source/Backend/Tests”  Run the function  “test\_ec\_add ()” | Running the function within the test script | | | The Event Configuration object is inserted as a document in the Event Configuration collection within the PICKDB database. An acknowledgment is printed in the command prompt | A default “\_id” attribute is added to the Event Configuration document, not used within the system. |
| 3 | Refer to the “Mongo Compass” tool and refresh the “Event Configuration” collection, view the structure of the inserted Event Configuration document. | Observing the insertion/mapping  of the object in the DB. | | | The Event Configuration object is inserted and mapped as the document schema design inside the “Event Configuration” collection. | The document schema design can be referenced back in the SDD Sec.4.1. |
| Concluding Remarks:  The mapping of the document schema design facilitates pulling the object from the DB and load it into the system. Reference the Appendix (Sec.7) to see the procedure and results of the test. | | | | | | |
| Testing Team:  Elizabeth Barragan | | | Date Completed:  04/09/20 | | | |

## Test DB 2

**Objective:** The objective of this test is to verify that the Vector object will be mapped with the document schema design once a Vector object is inserted.

**Notes:** The Vector object is not yet integrated with the GUI, unable to obtain user input, the Vector attributes need to be given when the class is invoked.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test No.: DB2 | | | | Current Status: Passed | | |
| Test title: Test the mapping/insertion between the Vector object with the DB document structure. | | | | | | |
| Testing approach: This test will be conducted by utilizing the Vector, Graph, Node, Node Visibility, Relationship and Significant Log Entry classes that will contain their given attributes. The mapping of the Vector object will be reflected in the GUI Database tool, “Mongo Compass.” | | | | | | |
| STEP  1 | OPERATOR ACTION  Open up the command prompt, run the command “mongod” to connect to the MongoDB server.  Open up MongoDB Compass, be connected to the DB local host, select the “PICKDB” database. Select the “Vector” Collection. | PURPOSE  Initial Condition | | | EXEPCTED RESULTS  The Vector  collection does not contain any documents. | COMMENTS  Assumption that the tester does not have the MongoDB server running.  The “PICKDB” database along with the “Vector” collection is created manually through the MongoDB Compass tool. |
| 2 | Run the “DBTest.py” script located in the directory “Source/Backend/Data/”  Run the function  “test\_vector\_add ()” | Running the function within the test script | | | The Vector object is inserted as a document in the Vector collection within the “PICKDB” database. An acknowledgment is printed in the command prompt | A default “\_id” attribute is added to the Vector document, not used within the system. |
| 3 | Refer back to the “Mongo Compass” tool and refresh the “Vector” collection, view the structure of the inserted Vector document. | Observing the insertion/mapping  of the object in the DB. | | | The Vector object is inserted and mapped as the document schema design inside the “Vector” collection. | The document schema design can be referenced back in the SDD Sec.4.1. |
| Concluding Remarks:  The mapping of the embedded document design will facilitate pulling multiple objects from the DB and load them into the system all at once. Reference the Appendix (Sec.7) to see the procedure and results of the test. | | | | | | |
| Testing Team:  Elizabeth Barragan | | | Date Completed:  04/11/20 | | | |

## Test DB 3

**Objective:** The objective of this test is to verify if one or many attributes are updated in the Event Configuration object the change will be reflected in the document within the DB.

**Notes:** The Event Configuration object is not yet integrated with the GUI, unable to obtain user input, the Event Configuration attributes need to be given wherever the class is invoked.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test No.: DB 3 | | | | Current Status: Pending | | |
| Test title: Update one or many attributes of an Event Configuration document | | | | | | |
| Testing approach: This test will be conducted by utilizing the Event Configuration class that will contain it’s given attributes, updating one or many of the object’s attribute will be reflected in the database GUI tool. | | | | | | |
| STEP  1 | OPERATOR ACTION  Open up MongoDB Compass, be connected to the DB local host, select the “PICKDB” database. Select the “Event Configuration” Collection. | PURPOSE  Initial Condition | | | EXEPCTED RESULTS  An “Event Configuration” document exists within the Event Configuration Collection. | COMMENTS  Assumption: The DB Mongo server is already running.  The “PICKDB” database along with the “Event Configuration” collection has been already created before running this test. |
| 2 | Go to the “DBTest.py” script located in the directory “Source/Backend/Data/”  and modify one or multiple parameters given within the “test\_ec\_update ()” function then run the function. | Running the function within the test script | | | The attribute changes are updated within the Event Configuration document in the DB. |  |
|  | Refer back to the “Mongo Compass” tool and refresh the “Event Configuration” collection. | Observing the updates. | | | The update(s) is reflected in MongoDB compass in Event Configuration document. |  |
| Concluding Remarks:  **HAS NOT BEEN TESTED** | | | | | | |
| Testing Team:  Elizabeth Barragan | | | Date Completed:  TBD | | | |

## Test DB 4

**Objective:** The objective of this test is to test the pull/search of an Event Configuration object from the DB, reflecting it in the GUI.

**Notes:** The Event Configuration object is not yet integrated with the GUI, unable to obtain user input, the Event Configuration will pull the attributes from an Event Configuration document in the DB.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test No.: DB 4 | | | | Current Status: Failed | | |
| Test title: Test the pull/search of an Event Configuration object from the DB | | | | | | |
| Testing approach: This test will be conducted by utilizing the Event Configuration class that will contain it’s given attributes; the results are reflected in the Event Configuration Window. | | | | | | |
| STEP  1 | OPERATOR ACTION  Open up MongoDB Compass, be connected to the DB local host, select the “PICKDB” database. Select the “Event Configuration” Collection. | PURPOSE  Initial Condition | | | EXEPCTED RESULTS    One or many “Event Configuration” documents exists within the Event Configuration Collection. | COMMENTS  Assumption: The DB Mongo server is already running. |
| 2 | Go to the “GraphicalUserInterface.py” script located in the directory “Source/Frontend/”  Run the script. | Running the Graphical User Interface | | | The Graphical User Interface Main Window opens up. | The script is invoked within the main method in the same script. |
| 3 | Click on the “File 🡪 Project” button on the Menu tool bar within the Main Window | Accessing the Event Configuration Window | | | The Event Configuration Window pops up and the information pulled from the DB is reflected in the Window | To be able to modify what vector to search and pull from the DB, this will need to be specified in the Graphical user interface script. |
| Concluding Remarks:  The “Event Start Timestamp”, “Event End Timestamp” and “Connection Status” does not display the information pulled from the DB. Reference the Appendix (Sec. 7) to see the procedure and results of the test. | | | | | | |
| Testing Team:  Elizabeth Barragan | | | Date Completed:  04/27/20 | | | |

## Test DB 5

**Objective:** The objective of this test is to delete a vector object from the DB

**Notes:** The Graph/Vector object is not yet integrated with the GUI, unable to obtain the attributes from the Graph along its nodes and relationship. The Vector contains dummy data to test the deletion of the object.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test No.: DB 5 | | | | Current Status: Passed | | |
| Test title: Test the deletion of a Vector object from the DB. | | | | | | |
| Testing approach: This test will be conducted by utilizing the Vector class that will contain its dummy attributes; the results are reflected in the “MongoDB Compass” tool | | | | | | |
| STEP  1 | OPERATOR ACTION  Open up “MongoDB Compass “tool, be connected to the DB local host, select the “PICKDB” database. Select the “Vector” Collection. | PURPOSE  Initial Condition | | | EXEPCTED RESULTS    One or many “Vectors” documents exists within the Vector Collection. | COMMENTS  Assumption: The DB Mongo server is already running. |
| 2 | Go to the “DBTest.py” script located in the directory “Source/Backend/Data/”  and run the “test\_vector\_delete ()” function | Running the function within the test script. | | | The Vector that matches the specified search criteria within the test function will be deleted from the Vector collection in the DB. |  |
| 3 | Refer back to the “Mongo Compass” tool and refresh the “Vector” collection. | Observing the deletion | | | The Vector document within the “Vector” collection is deleted. |  |
| Concluding Remarks:  The Vector document along with its embedded sub documents (Graph, Nodes, Relationships, Icon, etc.) and references to other collections are deleted. Reference the Appendix (Sec. 7) to see the procedure and results of the test. | | | | | | |
| Testing Team:  Elizabeth Barragan | | | Date Completed:  04/27/20 | | | |

## Test DB 6

**Objective:** The objective of this test is to test the pull/search of a Vector object from the DB, reflecting it in the GUI.

**Notes:** The Vector, Graph, Nodes, Relationships objects are not yet integrated with the GUI, unable to obtain attributes input, the Vector attributes need to be given wherever the class is invoked.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test No.: DB 6 | | | | Current Status: Pending | | |
| Test title: Test the pull/search of a Vector object from the DB | | | | | | |
| Testing approach: This test will be conducted by utilizing the Vector class that will contain it’s given attributes; the results are reflected in the Vector Window, Graph. | | | | | | |
| STEP  1 | OPERATOR ACTION  Open up “MongoDB Compass, be connected to the DB local host, select the “PICKDB” database. Select the “Vector” Collection. | PURPOSE  Initial Condition | | | EXEPCTED RESULTS    One or many “Vectors” documents exists within the Vector Collection. | COMMENTS  Assumption: The DB Mongo server is already running. |
| 2 | Go to the “GraphicalUserInterface.py” script located in the directory “Source/Frontend/”  Run the script. | Running the Graphical User Interface | | | The Graphical User Interface Main Window opens up and the Graph is Visible | The script is invoked within the main method in the same script. |
| 3 | Click on the “File 🡪 Project” button on the Menu within the Main Window. Click on the Vector tab. | Accessing the Vector Window | | | The Vector information pulled from the DB is reflected in the Vector Window. | To be able to modify what vector to search and pull from the DB, this will need to be specified in the Graphical user interface script. |
| Concluding Remarks:  **HAS NOT BEEN TESTED** | | | | | | |
| Testing Team:  Elizabeth Barragan | | | Date Completed:  TBD | | | |

## Test DB 7

**Objective:** The objective of this test is to verify if one or many attributes within any of the Vector, Graph, Node, Relationships are updated the changes will be reflected in the Vector document within the DB.

**Notes:** The Vector, Graph, Nodes, Relationships objects are not yet integrated with the GUI, unable to obtain attributes input, the Vector, Graph, Nodes, Relationships attributes need to be given wherever the class is invoked.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test No.: DB 7 | | | | Current Status: Pending | | |
| Test title: Update one or many attributes of a Vector, Graph, Node, Relationship object in the DB. | | | | | | |
| Testing approach: This test will be conducted by utilizing the Vector class that will contain it’s given attributes; the results are reflected in the Vector document, and or Graph, Nodes, Relationship sub documents in the DB. | | | | | | |
| STEP  1 | OPERATOR ACTION  Open up MongoDB Compass, be connected to the DB local host, select the “PICKDB” database. Select the “Vector” collection. | PURPOSE  Initial Condition | | | EXEPCTED RESULTS  One or many “Vectors” documents exists within the Vector Collection. | COMMENTS  Assumption: The DB Mongo server is already running. |
| 2 | Go to the “DBTest.py” script located in the directory “Source/Backend/Data/”  and modify one or multiple parameters given within the “test\_vector\_update ()” function then run the function. | Running the function within the script | | | The attribute changes are updated within the Vector document and or sub documents within the “Vector” collection in the DB. |  |
|  | Refer back to the “Mongo Compass” tool and refresh the “Vector” collection. | Observing the updates. | | | The update(s) are reflected in MongoDB compass in the Vector document. |  |
| Concluding Remarks:  **HAS NOT BEEN TESTED** | | | | | | |
| Testing Team:  Elizabeth Barragan | | | Date Completed:  TBD | | | |

## Test Ing 1

**Objective:** The objective of this test is to confirm whether the system performs cleansing properly and if not to determine which functionality of it fails.

**Notes:** Ingestion subsystem is not completely implemented yet. Practice data must be created specifically to point out the functionality of cleansing. At least 3 files must be created for this test. Test duration should be within 5 minutes.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test No.: Ing 1 | | | | Current Status: Passed | | |
| Test title: Test for cleansing for additional spaces, lines, & binary data. | | | | | | |
| Testing approach: This test will be performed by setting up practice data to ingest into the system | | | | | | |
| STEP | OPERATOR ACTION  Begin test by creating practice data and by saving it in the uncleansed directory  Have one file to have extra lines, another to have extra spaces, and the last one to have binary data | PURPOSE  Initial condition to properly test | | | EXEPCTED RESULTS  The system accepts the dummy files and begins process, so the user can make sure that conditions are met for cleansing | COMMENTS |
|  | Start the cleansing process by running the system | This is so you can make sure the system will accept the files and begin cleansing | | | The system begins the cleansing process without errors | If errors are found in this phase, the system doesn’t accept any files at all and is considered a failure. |
|  | After the system begins cleansing it lets the user know it is done cleansing. At this stage the user needs to check Cleansed directory made from the cleansing process. | This is to make sure that the system properly cleanses the files, and you know exactly which functionality lacks. | | | The system cleanses the files without any issues, and this can be seen in the cleansed folder. | An error occurs here, the file with the extra lines is cleansed properly along with the one with extra spacing, along with non-ascii words erased. |
| Concluding Remarks:  Test passes by removing white spaces and the non ascii characters. | | | | | | |
| Testing Team:  Yamel Hernandez | | | Date Completed:  04/26/20 | | | |

## Test Ing 2

**Objective:** The objective of this test is to confirm whether the system performs parsing through splunk properly and if not to determine which functionality of it fails.

**Notes:** Ingestion subsystem is not completely implemented yet. Practice data must be created specifically to point out the functionality of parsing through splunk.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test No.: Ing 2 | | | | Current Status: Passed | | |
| Test title: Test Parsing in Splunk | | | | | | |
| Testing approach: This test will be performed by setting up practice data to ingest into the system | | | | | | |
| STEP | OPERATOR ACTION  Begin test by creating practice data and by saving it in the uncleansed directory | PURPOSE  Initial condition to properly test | | | EXEPCTED RESULTS  The system accepts the dummy files and begins process, so the user can make sure that conditions are met for cleansing to be sent to splunk. | COMMENTS |
|  | Start the process by starting the software. | This is so you can make sure the system will accept the files | | | The system begins the cleansing process without errors and files are then sent to splunk | If errors are found in this phase, the system doesn’t accept any files at all and is considered a failure. |
|  | After the system begins the system lets the user know, and prompts the user to input their credentials for splunk | This is to make sure that the system properly accepts the files into splunk | | | The system takes the files from the cleansed folder and uploads them to splunk. |  |
| Concluding Remarks:  Test passes by parsing properly the files. | | | | | | |
| Testing Team:  Yamel Hernandez | | | Date Completed:  4/20/20 | | | |

## Test Ing 3

**Objective:** The objective of this test is to confirm whether the system creates an enforcement action report properly and if not to determine which functionality of it fails.

**Notes:** Ingestion subsystem is not completely implemented yet. Practice data must be created specifically to point out the functionality of the report to point out errors.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test No.: Ing 3 | | | | Current Status: Pending | | |
| Test title: Test for error messages in enforcement action report & error class | | | | | | |
| Testing approach: This test will be performed by setting up practice data to ingest into the system | | | | | | |
| STEP | OPERATOR ACTION  Begin test by creating practice data and by saving it in the uncleansed directory  Have some files with errors and know where the errors are located and what the errors are. | PURPOSE  Initial condition to properly test | | | EXEPCTED RESULTS  The system accepts the dummy files and begins process, so the user can make sure that conditions are met to create the enforcement action report. | COMMENTS |
|  | Start the process by running the system | This is so you can make sure the system will accept the files and begin the initial steps | | | The system finds an error or errors. | If errors are not found in this phase, it means the enforcement action report isn’t properly working and it needs more work done to the class. |
|  | Once the system finds errors, a report is generated and is shown to the user. | This is to make sure that the enforcement action report is producing proper output for the files that have errors. | | | The system generates a report for which files have errors, where the errors are and what the errors are. |  |
| Concluding Remarks: | | | | | | |
| Testing Team:  Yamel Hernandez | | | Date Completed:  TBD | | | |

## Test Ing 4

**Objective:** The objective of this test is to confirm whether the system flags log files properly when ingesting.

**Notes:** Ingestion subsystem is not completely implemented yet. Test duration should be within 5 minutes.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test No.: Ing 4 | | | | Current Status: Pending | | |
| Test title: Test ingestion process to make sure log files get flagged for the state that they are in. | | | | | | |
| Testing approach: This test will be performed by setting up practice data to ingest into the system | | | | | | |
| STEP | OPERATOR ACTION  Begin test by creating practice data and by saving it in the uncleansed directory  Create one file so it can be flagged uncleansed, another cleansed, and so on. | PURPOSE  Initial condition to properly test | | | EXEPCTED RESULTS  The system accepts the dummy files and begins process, so the user can make sure that conditions are met. | COMMENTS |
|  | Start the process by running the system. | This is so you can make sure the system will accept the files and to make sure all flagging components work. | | | The system runs and rejects one file (uncleansed), creates an enforcement action report, other files get flagged cleansed, and so on for the validation & ingestion status. |  |
|  | After the system begins the process, we can check the flagging of the files in the log file class | This is to make sure that the system flags entries properly. | | | The system flags the files properly. | If no files are flagged then that means the log file class is not properly flagging the files. |
| Concluding Remarks: | | | | | | |
| Testing Team:  Yamel Hernandez | | | Date Completed:  TBD | | | |

## Test EC 1

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test No.: EC 1 | | | | Current Status: Pending | | |
| Test title: Test event configuration instantiation with user input data | | | | | | |
| Testing approach: This test will be performed by initiating a new project, which requires the user to input event configuration information directly. | | | | | | |
| STEP | OPERATOR ACTION  Initialize the PICK system. | PURPOSE  We must initialize the system to be prompted to begin a new project | | | EXEPCTED RESULTS  System initializes, main window pops up alongside with a pop-up window prompting the user to continue or create a new project | COMMENTS |
|  | Select option to “create new project” | This sets the environment for the user to input initial project information | | | Pop-up window progresses to next stage in project initialization |  |
|  | Provide event configuration data as follows:   * Event description: “Test” * Event start timestamp: “10:10 10/10/10 AM” * Event end timestamp: 11:11 11/11/11 PM” * Root directory: “C:/Events/10-10-10” * Red directory: “C:/Events/10-10-10/Red” * White directory: “C:/Events/10-10-10/White” * Blue directory: “C:/Events/10-10-10/Blue” | This initializes event configuration attributes to test parameters | | | Event configuration object is created and stored in database, along with generated attributes “Lead” set to TRUE, “Lead’s IP Address” set to the current machine’s IP address, and “Connections Established” set to 0. |  |
|  | Select option to begin ingestion | This option sets up the project and begins ingestion | | | The user has access to the main window |  |
|  | On the menu bar, select “File”, then select “Project” | This step opens the project information menu, which displays all the information for the current project. | | | This window defaults to the event configuration information; therefore, the tester should see the previously input test parameters in step 3, along with generated parameters mentioned in the expected results of step 3. |  |
| Concluding Remarks: This test signifies a successful recording, storing, and displaying of event configuration information that is input by a user. | | | | | | |
| Testing Team:  Jose Antoine Leon Cordero | | | Date Completed:  TBD | | | |

## Test GC1

**Objective:** Verify creation of Graph with GC1

**Notes:** It is a component test to verify correct implantation of a graph.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test No.: GC1 | | | | Current Status: Pending | | |
| Test title: Testing for creation of a Graph | | | | | | |
| Testing approach: Testing creation of Graph with one correct input parameters and two graphs that won’t generate because of incorrect parameters past into construct.  Environment: PyCharm Professional 2019.3.2  Runtime Version: 11.0.5+10-b520.30 amd64  VM: OpenJDK 64-bit Server VM by JetBrains s.r.o  With pyQt5 == 5.14.1  Test cannot run without PyQt5 installed at least version 5.14.1, test is critical, Graph is one of the most important aspects. | | | | | | |
| STEP  1  2  3 | OPERATOR ACTION  Run “VectorTest.py” found inside “Test” directory inside of “Graph” directory. (Refer to Appendix)  Click on the “+” sign tab above the tabular view of significant log entries.  Enter “Graph1” on the name section click save. | PURPOSE  Testing creation of graph for each vector.  Generate different graphs.  Create a different vector in order to create a graph. | | | EXEPCTED RESULTS  An instance of the PICK system will open on the main window.  A pop-window asking for name and optional description will show.  A new vector will be added to the vector tabs and a new graph will be shown inside the vector. | COMMENTS |
| Concluding Remarks: Has not been tested yet because it has not been completed. | | | | | | |
| Testing Team:  Manuel | | | Date Completed: TBA | | | |

## Test GC2

**Objective:** Verify feature creation of nodes from log entries works properly.

**Notes:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test No.: GC2 | | | | Current Status: Pending | | |
| Test title: Test creation of a node | | | | | | |
| Testing approach: Testing Graph component of a vector | | | | | | |
| STEP  1  2 | OPERATOR ACTION  Run “VectorTest.py” found inside “Test” directory inside of “Graph” directory. (Refer to Appendix)  Right click over log entry with Log Entry Name of “Log Entry 4” and select add to current vector. | PURPOSE  Testing creation of a node with predefined log entry data.  Test creation of a node from a log entry. | | | EXEPCTED RESULTS  An instance of the PICK system will open on the main window with five predefined log entries in the log entry section and one predefined vector “Vector1”.  A node with name of “Log Entry 4” will be created inside the graph of “Vector 1”. | COMMENTS |
| Concluding Remarks: Has not been tested yet because it has not been completed. | | | | | | |
| Testing Team:  Manuel Delgado | | | Date Completed: TBA | | | |

## Test GC3

**Objective:** Verify creation of a relationship between nodes in a graph is working properly.

**Notes:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test No.: GC3 | | | | Current Status: Pending | | |
| Test title: Test creation of a relationship | | | | | | |
| Testing approach: | | | | | | |
| STEP  1  2 | OPERATOR ACTION  Run “VectorTest.py” found inside “Test” directory inside of “Graph” directory. (Refer to Appendix)  From “Vector1” graph click and drag from node with name “Node2” to node with name “Node3” and release. | PURPOSE  Test relationship creation  Create new relationship graphically. | | | EXEPCTED RESULTS  An instance of the PICK system will open on the main window with three predefined significant log entries in the tabular view in the predefined vector “Vector1”.  Once the mouse is released a new relationship will have been created between node “Node2” to “Node3”. | COMMENTS |
| Concluding Remarks: Has not been tested yet because it has not been completed. | | | | | | |
| Testing Team:  Manuel Delgado | | | Date Completed: TBA | | | |

## Test GC4

**Objective:** Verify the graph contains correct functionality for the nodes displayed.

**Notes:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test No.: GC4 | | | | Current Status: Pending | | |
| Test title: Test functionality of movement for a node | | | | | | |
| Testing approach: | | | | | | |
| STEP  1  2 | OPERATOR ACTION  Run “VectorTest.py” found inside “Test” directory inside of “Graph” directory. (Refer to Appendix)  Inside vector “Vector1” graph click and hold on node “Node3” and drag around the graph. | PURPOSE  Test movement of nodes on a graph.  Test node movement. | | | EXEPCTED RESULTS  An instance of the PICK system will open on the main window with three predefined nodes in the graph of predefined vector “Vector1”.  “Node3” will move around the graph without any problems. | COMMENTS |
| Concluding Remarks: Has not been tested yet because it has not been completed. | | | | | | |
| Testing Team:  Manuel Delgado | | | Date Completed: TBA | | | |

## Test GC5

**Objective:** Verify that moving a node in the graph with a relationship already created moves correctly.

**Notes:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test No.: GC5 | | | | Current Status: Pending | | |
| Test title: Test functionality of movement for a relationship | | | | | | |
| Testing approach: | | | | | | |
| STEP  1  2 | OPERATOR ACTION  Run “VectorTest.py” found inside “Test” directory inside of “Graph” directory. (Refer to Appedndix)  Click and hold node “Node2” inside vector “Vector1” graph and move around. | PURPOSE  Test the relationships predefined between nodes are working without any mistakes.  Test movement of a node with a relationship already attached to it works properly. | | | EXEPCTED RESULTS  An instance of the PICK system will open on the main window with three predefined nodes in the graph of predefined vector “Vector1”.  Relationships attached to node “Node2” will move without any problems. | COMMENTS |
| Concluding Remarks: Has not been tested yet because it has not been completed. | | | | | | |
| Testing Team:  Manuel Delgado | | | Date Completed: TBA | | | |

## Test GC6

**Objective:** Verify deletion of a node deletes the node from the table and all relevant information connected to that node and does not do anything else.

**Notes:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test No.: GC6 | | | | Current Status: Pending | | |
| Test title: Test deletion of a node | | | | | | |
| Testing approach: | | | | | | |
| STEP  1  2 | OPERATOR ACTION  Run “VectorTest.py” found inside “Test” directory inside of “Graph” directory. (Refer to Appedndix)  Right click over node “Node1” in graph of vector “Vector1” and select delete. | PURPOSE  Test all the correct information is deleted when a node is deleted.  Delete node | | | EXEPCTED RESULTS  An instance of the PICK system will open on the main window with three predefined nodes in the graph of predefined vector “Vector1”.  Node “Node1” is deleted and relationship to node “Node2” is also deleted. | COMMENTS |
| Concluding Remarks: Has not been tested yet because it has not been completed. | | | | | | |
| Testing Team:  Manuel Delgado | | | Date Completed: TBA | | | |

## Test GC7

**Objective:** Verify information editing of a nodes content is handled, saved and displayed correctly.

**Notes:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test No.: GC7 | | | | Current Status: Pending | | |
| Test title: Test editing of a node | | | | | | |
| Testing approach: | | | | | | |
| STEP  1  2  3 | OPERATOR ACTION  Run “VectorTest.py” found inside “Test” directory inside of “Graph” directory. (Refer to Appedndix)  Right click over node “Node1” and select edit.  Click on Log Reference editable information and delete. Type in “I changed this” and click save. | PURPOSE  Test editing of a node already existent in a graph.  Edit node information.  Test editing of anode works correctly | | | EXEPCTED RESULTS  An instance of the PICK system will open on the main window with three predefined nodes in the graph of predefined vector “Vector1”.  A pop-up window with node “Node1” information will be shown to and editable.  Pop-window will disappear and node “Node1” on graph will now contain “I changed this” in the Log Entry Reference section | COMMENTS |
| Concluding Remarks: Has not been tested yet because it has not been completed. | | | | | | |
| Testing Team:  Manuel Delgado | | | Date Completed: TBA | | | |

## Test GC8

**Objective:** Verify editing of an already existing relationship is done correctly.

**Notes:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test No.: GC8 | | | | Current Status: Pending | | |
| Test title: Test editing of a relationship | | | | | | |
| Testing approach: | | | | | | |
| STEP  1  2  3 | OPERATOR ACTION  Run “VectorTest.py” found inside “Test” directory inside of “Graph” directory. (Refer to Appendix)  Right click over relationship between node “Node1” and node “Node2” and select edit.  Click on relationship and delete label “Relationship1” and type in “Changed Relationship” | PURPOSE  Test editing of a predefined relationship between nodes.  Edit predefined relationship between nodes.  Verify edit was updated graphically. | | | EXEPCTED RESULTS  An instance of the PICK system will open on the main window with three predefined nodes in the graph of predefined vector “Vector1”.  A pop-window will be shown with the node “Node1” information.  Pop-window will be closed and edited information will be displayed on the graph window. | COMMENTS |
| Concluding Remarks: Has not been tested yet because it has not been completed. | | | | | | |
| Testing Team:  Manuel Delgado | | | Date Completed: TBA | | | |

## Test VS1

**Objective:** Test both table and node are being updated simultaneously when information is being edited in either form.

**Notes:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test No.: VS1 | | | | Current Status: Pending | | |
| Test title: Test editing correlation between a node and a significant log entry | | | | | | |
| Testing approach: | | | | | | |
| STEP  1  2  3 | OPERATOR ACTION  Run “VectorTest.py” found inside “Test” directory inside of “Graph” directory. (Refer to Appedndix)  Double click on node “Node1” event type.  Delete “EventType1” and type “I changed this” | PURPOSE  Verify updating information on a table updated information on that node in the graph. | | | EXEPCTED RESULTS  An instance of the PICK system will open on the main window with three predefined significant log entries in the tabular view in the predefined vector “Vector1”.  Information will be highlighted blue.  Node “Node1” on graph will reflect change on event type | COMMENTS |
| Concluding Remarks: Has not been tested yet because it has not been completed. | | | | | | |
| Testing Team:  Manuel Delgado | | | Date Completed: TBA | | | |

# Test Schedule

This section demonstrates the schedule of the tests that were ran, the individual that ran the tests, and the tests that will be ran in the future.

|  |  |  |
| --- | --- | --- |
| **Task and date** | **People** | **Description** |
| DB Test 1  04/09/20 | Elizabeth | Test the mapping/ insertion of the Event Configuration object. |
| DB Test 2  04/11/20 | Elizabeth | Test the mapping/ insertion of the Vector object. |
| DB Test 3  04/30/20 | Elizabeth | Test to update an Event Configuration object in the DB. |
| DB Test 4  04/11/20 | Elizabeth | Test the pulling/searching of the Event Configuration object. |
| DB Test 5  04/27/20 | Elizabeth | Test the deletion of a vector object. |
| DB Test 6  04/30/20 | Elizabeth | Test the pull/search of a Vector object from the DB. |
| DB Test 7  04/03/20 | Elizabeth | Test to update a Vector along its components in the DB. |
| Ing 2  04/20/20 | Yamel | Test Parsing in Splunk |
| Ing 1  04/26/20 | Yamel | Test for cleansing for additional spaces, lines, & binary data |
| Ing 3  04/30/20 | Yamel | Test for error messages in enforcement action report & error class |
| Ing 4  04/30/20 | Yamel | Test ingestion process to make sure log files get flagged for the state that they are in. |
| EC 1  04/30/20 | Antoine | Test event configuration instantiation with user input data |
| GC1  4/29/20 | Manuel | Testing creation of a graph for a vector. |
| GC2  4/29/20 | Manuel | Testing creation of a node. |
| GC3  4/30/20 | Manuel | Test creation of a relationship. |
| GC4  4/29/20 | Manuel | Test functionality of movement for a node. |
| GC5  4/30/20 | Manuel | Test functionality of movement for a relationship. |
| GC6  4/30/20 | Manuel | Test deletion of a node. |
| GC7  4/30/20 | Manuel | Test editing of a node. |
| GC8  4/30/20 | Manuel | Test editing of a relationship. |
| VS1  5/1/20 | Manuel | Test editing correlation between a node and a significant log entry. |

# Appendix

**[Test DB 1]**

**A close up of text on a black background

Description automatically generated**

**Figure 1.1.** Starting the MongoDB Server

A screenshot of a social media post

Description automatically generated

**Figure 1.2.** Connecting to the DB Local Host in MongoDB Compass

A screenshot of a cell phone

Description automatically generated

**Figure 1.3.** Accessing the Event Configuration Collection in MongoDB Compass

A screenshot of a cell phone

Description automatically generated

**Figure 1.4.** Running the test script function

A picture containing knife, bird

Description automatically generated

**Figure 1.5.** Acknowledgment of object inserted

A screenshot of a cell phone

Description automatically generated

**Figure 1.6.** Event Configuration document structure in MongoDB Compass

**[Test DB 2]**

**A close up of text on a black background

Description automatically generated**

**Figure 2.1.** Starting the MongoDB Server

**A screenshot of a social media post

Description automatically generated**

**Figure 2.2.** Connecting to the DB Local Host in MongoDB Compass

**A screenshot of a cell phone

Description automatically generated**

**Figure 2.3.** Accessing the Vector Collection in MongoDB Compass

**A picture containing knife

Description automatically generated**

**Figure 2.4.** Running the test script function

**A screenshot of a cell phone

Description automatically generated**

**Figure 2.5.** Acknowledgment of object inserted

**A screenshot of a social media post

Description automatically generated**

**Figure 2.6.** Vector document structure in MongoDB Compass

**[Test DB 4]**

A screenshot of a cell phone

Description automatically generated

**Figure 4.1.** Accessing the Event Configuration Collection in MongoDB Compass

**A screenshot of a cell phone

Description automatically generated**

**Figure 4.2.** Running the test script function

**A picture containing bird

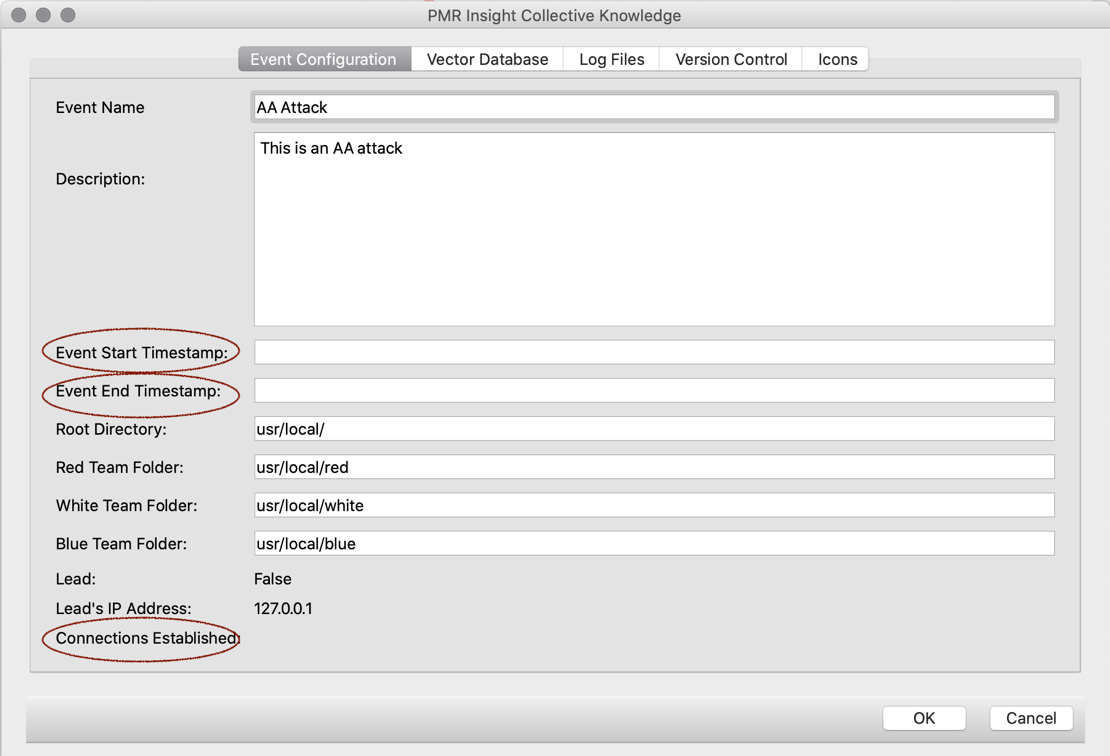
Description automatically generated**

**Figure 4.3.** Acknowledgment of object pulled from DB with the search parameters

**A screenshot of a cell phone

Description automatically generated**

**Figure 4.4** Tool Bar Menu button selection

****

**Figure 4.5.** Event Configuration Window displaying the pulled object’s attributes from the DB

****

**Figure 4.6.** Event Configuration document attributes in MongoDB Compass

**[Test DB 5]**

**A screenshot of a cell phone

Description automatically generated**

**Figure 5.1.** Vector document in MongoDB Compass

**A picture containing knife

Description automatically generated**

**Figure 5.2.** Running the test script function

**A screenshot of a cell phone

Description automatically generated**

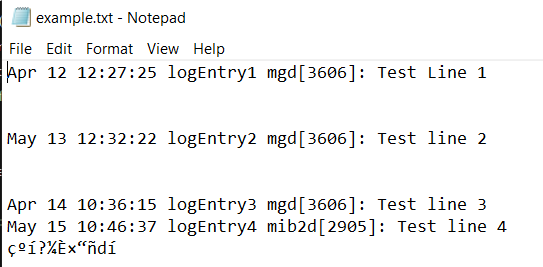
**Figure 5.3.** Acknowledgment of the deletion of the object

**A screenshot of a cell phone

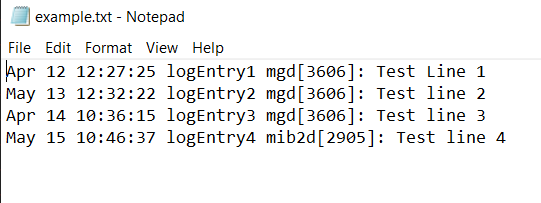
Description automatically generated**

**Figure 5.4.** Vector document deletion in MongoDB Compass

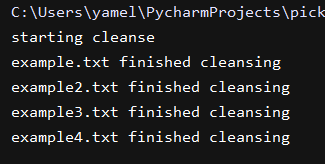
**[Test Ing 1]**

****

**Uncleansed Data with Non-Ascii Characters**

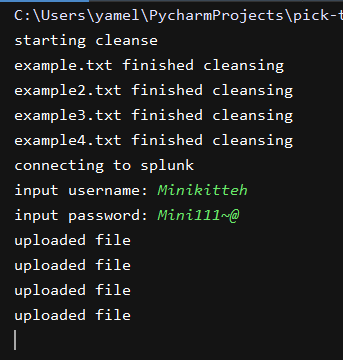
****

**Cleansed Data**

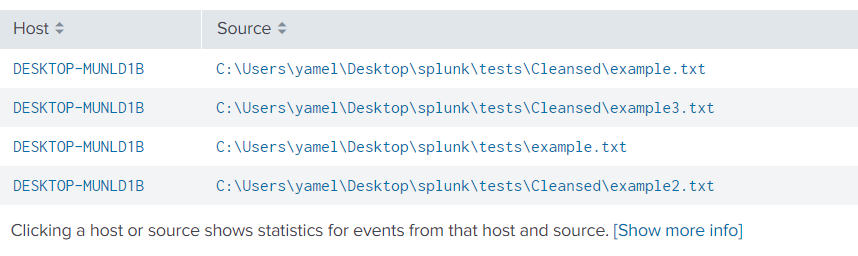
****

**Terminal Output**

**[Test Ing 2]**

****

**Terminal Input & Output for Splunk**

****

**Files on Splunk**

\*