CMSC 447

Software Test Description (STD)

[1 Scope 3](#_Toc432633399)

[1.1 Identification 3](#_Toc432633400)

[1.2 System overview 3](#_Toc432633401)

[1.3 Document overview 3](#_Toc432633402)

[2 Referenced documents 3](#_Toc432633403)

[3 Test preparations 3](#_Toc432633404)

[3.1 (Project-unique identifier of a test) 3](#_Toc432633405)

[3.1.1 Hardware preparation 3](#_Toc432633406)

[3.1.2 Software preparation 4](#_Toc432633407)

[3.1.3 Other pre-test preparations 4](#_Toc432633408)

[4 Test descriptions 4](#_Toc432633409)

[4.1 (Project-unique identifier of a test) 4](#_Toc432633410)

[4.1.1 (Project-unique identifier of a test case) 4](#_Toc432633411)

[5 Requirements traceability 6](#_Toc432633412)

[6 Notes 7](#_Toc432633413)

[A. Appendixes 8](#_Toc432633414)

# Scope

This section shall be divided into the following paragraphs.

## Identification

This paragraph shall contain a full identification of the system and the software to which this document applies, including, as applicable, identification number(s), title(s), abbreviation(s), version number(s), and release number(s).

This project is to be a web application which when implemented will allow a user to input preferences into a questionnaire and top locations will then be outputted onto a map. When first accessing the webpage, the user will begin with the “login page.” From the login page, a user can access a “create an account page” or after authentication access their “homepage.” The homepage will allow the user to modify their account information from a “modify account page,” as well as allow the user to see the “map” and access the “questionnaire,” which are defined below.

The “questionnaire” refers to the portion of the project where the user inputs these preferences. This will be accessed from the homepage into a “questionnaire page.” The “ratings” will refer to the numerical input from a user in which they determine which statistics have a higher preference. The “statistics” refer to the set of data that our project will be accessing.

The “map” refers to the portion of the project which is the visualization of data. The map is initially displayed from the homepage, but will initially be empty, and be modified as the user takes the questionnaire. After the user completes the questionnaire the map is referenced to be an “updated map” with the continuing definitions being components of the map. The map may also have markers which will be referred to as “pins.” The pins will allow the user to see an image of the location referred to as the “pin image.” In some cases where an image is not available, or if there is additional implementation, a pin may allow the user to see descriptive information on the location which will be referred to as the “pin information.”

## System overview

This paragraph shall briefly state the purpose of the system and the software to which this document applies. It shall describe the general nature of the system and software; summarize the history of system development, operation, and maintenance; identify the project sponsor, acquirer, user, developer, and support agencies; identify current and planned operating sites; and list other relevant documents.

The goal of this project is to create a web application which will allow a user to create an account, and from such account be able to evaluate different statistical values such that a list of locations within the United States will be outputted to the user which correlate to the user input. With the list, a map should be presented with pins that correlate to the location and can then be further expanded to output a location image or description.

We expect to use a database for maintaining username and password data. Along with this we expect to be using third party software for our output. This would include the use of GoogleMaps for our map output, and possibly Flickr for our image outputs, and Wikipedia for a description output. Further third-party software will be expanded in this section as seems fit throughout the project.

Because the nature of this project is in the scope of a classroom project, there will not be long term maintenance of the project and will be run on a local machine. Throughout this document the term sponsor will refer to John Winder who is the group’s client throughout the project. Similarly, the group refers to the group of developers working on the project including Matthew Hearn, Aaron Lewis, Alex Rochford, Cathy Poore, Ben Kittner, and Steven Heckman. The project will refer to the software and documentation created for this assignment.

All project development will be done through GitHub and then developer preference for development environments, debuggers, ect. The web portion of the application will be run using an Angular Framework on a localhost.

## Document overview

This paragraph shall summarize the purpose and contents of this document and shall describe any security or privacy considerations associated with its use.

The purpose of this document is to describe the testing procedure that is to be implemented for the system previously described. This will include the procedures for setting up a test environment, the testing cases, and descriptions of the actual tests. Each test will be testing a different unique feature within the system. These will be broken into unit, integration, system, and acceptance tests. Unit test will be testing individual functions within the system. Unit tests will be testing the individual functions within the system. These will be written by the developer of the function. Automated test should also be included within a test driver unique to each programming language. This will be done to allow for ease of regression testing. Integration testing will be done after completion of each CSCI component. Automation of the integration testing should have a unique driver for each CSCI component. Similarly, after completion of each CSCI component, a regression test should be done for the previously developed CSCI components. Finally, the system test should be done after completion of the last CSCI component is developed, tested, and completion of all regression testing. Automated system test should have a unique test driver. Finally, after completion of the system test then acceptance test should be done with the project client.

For transparency, the following section will include any updates made to this document:

Version 1.0: This is the completion of the original document.

Version 1.1: Updated spelling and grammar throughout the document. Also updated test to include test for the avatar requirement which was updated as of our 4/23 meeting with the client. This also included an update to the setup which now includes installation information for Python, Visual C++, and MYSQL.

# Referenced documents

This section shall list the number, title, revision, and date of all documents referenced in this document. This section shall also identify the source for all documents not available through normal Government stocking activities.

The following documents may be referenced throughout this document:

SDP- Software Development Plan

SRS- Software Requirements Specification

SDD- Software Design Description

# Test preparations

This section shall be divided into the following paragraphs. Safety precautions, marked by WARNING or CAUTION, and security and privacy considerations shall be included as applicable.

## (Project-unique identifier of a test)

This paragraph shall identify a test by project-unique identifier, shall provide a brief description, and shall be divided into the following subparagraphs. When the information required duplicates information previously specified for another test, that information may be referenced rather than repeated.

### Hardware preparation

This paragraph shall describe the procedures necessary to prepare the hardware for the test. Reference may be made to published operating manuals for these procedures. The following shall be provided, as applicable:

* + - 1. The specific hardware to be used, identified by name and, if applicable, number
      2. Any switch settings and cabling necessary to connect the hardware
      3. One or more diagrams to show hardware, interconnecting control, and data paths
      4. Step-by-step instructions for placing the hardware in a state of readiness

This system is purely a software system. As a result, there is no hardware preparation outside of a required internet connection setup.

### Software preparation

This paragraph shall describe the procedures necessary to prepare the item(s) under test and any related software, including data, for the test. Reference may be made to published software manuals for these procedures. The following information shall be provided, as applicable:

* + - 1. The specific software to be used in the test
      2. The storage medium of the item(s) under test (e.g., magnetic tape, diskette)
      3. The storage medium of any related software (e.g., simulators, test drivers, databases)
      4. Instructions for loading the software, including required sequence
      5. Instructions for software initialization common to more than one test case

The following items will need to be installed and setup for this system: Python3, Visual C++, and MYSQL. Additionally, there will be a description for how to start the web application. Each will then have a further description on how to install below.

To run the backend code Python 3 will need to be installed. This will just require running the python-3.6.5-amd64.exe. This can be found in the Programs directory. In the first stage of the installation, you will need to choose the “Add to Path” checkbox option. Further instructions for the Python Installation can be found within the SETUP README file found in the home directory.

Most machines already have an installation of Microsoft Visual C++. To verify if your machine has Microsoft Visual C++ installed simply search for the application in your application search bar. If Microsoft Visual C++ is not installed on the machine simply run the vc\_redist.x64.exe found in the Programs directory. Further instruction for the Microsoft Visual C++ Installation can be found within the SETUP README file found in the home directory.

MYSQL will be required to run the login database in the frontend, as well as the Census Bureau database in the backend. This install file is called mysql-installer-web-community-8.0.11.0.exe which can be found in the Programs directory. This will require a custom install of MYSQL. The following will need to be installed during the process.

1. MySQL Server

2. MySQL Workbench

3. MySQL Shell

4. MySQL Router

5. Connector/Python (3.6) 8.0.11

For help finding the hidden installation requirements directions are below:

1. MySQL Server is under the MySQL Servers folder

2. MySQL Workbench, Shell, Router are under the Applications folder

3. Python connector is in the MySQL Connectors folder. Make sure to get the Python 3.6 x64 connector

From here leave all settings as except the root password should be the default setting. Set the password to be "password" (without the quotation marks.) When asked for the Authentication Method, check the Legacy Authentication Method. Finally, open the program MySQL Workbench and double click on "Local instance MySQL Router and do the following:

1) When asked for the root password, enter "password" and click connect

2) Click on the button in the toolbar that looks like two cylinders stacked on top of each other (If you hover over it should say create new schema) and call the new schema "project”. Further instructions for the MYSQL installation can be found within the SETUP README file.

Next is how to configure the backend environment. This will first require changing the PowerShell execution policy. To change the PowerShell execution policy. This can be done by opening. PowerShell as Administrator and run the command:

"Set-ExecutionPolicy RemoteSigned"

Next to set up the virtual environment to run python code. This can be done with the following steps:

1) In PowerShell run "pip install virtualenv"

2) Then in the Group2Project folder, run "virtualenv venv/"

3) Then run "venv/Scripts/activate"

4) then run "pip install -r Code/requirements/base.txt"

5) then run "python Code/manage.py migrate"

Python code can now be executed using the command:

"python Code/manage.py runserver". Further instructions can be found in the SETUP README file.

This setup is for a windows environment. First, in a browser go to: https://nodejs.org/dist/v8.11.1/node-v8.11.1-x64.msi

Next, install node js 8.1.1 with the default options. From the root directory, enter the frontend directory. Open a Windows PowerShell by holding shift and right clicking. From the list of options, click the “open PowerShell window here.” From here run the following commands:

npm install

npm run

This first command will install the dependencies for the system. The second command will open the site. Afterwards the website may be reached from localhost:4200. Further instructions on the setup process can be found within the SETUP README.

### Other pre-test preparations

This paragraph shall describe any other pre-test personnel actions, preparations, or procedures necessary to perform the test.

This project will require the use of a MYSQL database. The database name should be known as ‘TEAM2.’ This database will store a single table with the name of ‘USERINFO.’ This table will have fields ‘username’, ‘password’, and ‘about me.’ The username and password will be stored as VARCHAR(255) data type. The about me will be stored as a TEXT data type.

\*\*\*\*\*\*UPDATE THESE WITH SCHEMAS FOR BOTH DATABASES\*\*\*\*\*

# Test descriptions

This section shall be divided into the following paragraphs. Safety precautions, marked by WARNING or CAUTION, and security and privacy considerations shall be included as applicable.

## (Project-unique identifier of a test)

This paragraph shall identify a test by project-unique identifier and shall be divided into the following subparagraphs. When the required information duplicates information previously provided, that information may be referenced rather than repeated.

### (Project-unique identifier of a test case)

This paragraph shall identify a test case by project-unique identifier, state its purpose, and provide a brief description. The following subparagraphs shall provide a detailed description of the test case.

The following are the test that need to be done for the defined completion of this project as described within the system description. These tests will address those of the CSCI components and the overall system, and will not go into the individual unit tests. For information on the unit test, description will be given upon their development within the automated testing code. The test described here will be done to verify that the designs described within the SDD are completed as described. Test 1-4 are tests corresponding to the individual CSCI components within the system. Test 5 will verify that the CSCI components integrate into the system.

Test 1 will refer to verification of the login and create account pages portion of the system. Test 1.a will verify that the user cannot enter a NULL value for the username field. Test 1.b will verify that the user cannot enter a NULL value for the password field. Test 1.c will verify that the username entered is unique within the database. Test 1.d will verify that when the user enters ‘good’ input, which is input that passes tests 1.a-1.c, then the data is submitted into the login database. Test 1.e will verify that in the case of bad input from 1.a-c then the user receives an error message. Test 1.f will verify that the user can choose and avatar from the crate account page. These test all assume an initial state within the login page portion of the system. Verification that this page can be reached will be done in a later test. Test 1 will be viewed as a ‘pass’ if Tests 1.a-1.f are passed.

Test 2 will refer to the modify account page portion of the system. These tests should mirror those of the create account portion of the project. Test 2.a will verify that the newly entered username is not NULL. Test 2.b will verify that a newly entered password is not NULL. Test 2.c will verify that a newly entered username is unique from previously entered usernames within the login database. Test 2.d will verify that when the user enters ‘good’ input, which is input that passes tests 2.a-2.c, then the login database is updated with the new user information. Test 2.e will verify that if tests 2.a-c fail that the user receives an error message. Test 2.f will verify that the user may modify their avatar. These test all assume an initial state within the modify account portion of the system. Verification that this page can be reached will be done in a later test. Test 2 will be viewed as a ‘pass’ if Tests 2.a-2.e are passed.

Test 3 will refer to the questionnaire page portion of the system. Test 3.a will verify that there are at least 5 statistics that a user can rate. Test 3.b will verify that the user has entered a rating for each of the statistics. Test 3.c will verify that a list will output with 10 different locations. This shall be verified with a list output. Test 3.c will assume tests 3.a-b have been passed. Test 3.d will verify that the different statistical input will present different outputs to the user. Test 3.d will assume tests 3.a-3.c have been passed. Test 3.e should verify that data is being accessed from the correct inputs of the Census Bureau dataset. Test 3 will assume that the user can reach the questionnaire page of the system. Verification that this page can be reached will be done in a later test. Tests 3 will be viewed as a ‘pass’ if Tests 3.a-3.e are passed.

Test 4 will refer to the map output and homepage portions of the system. Test 4.a will verify that the user is presented with a list of their personal top choices. Test 4.b will verify that the list presented in 4.a has 10 unique locations. Test 4.c will verify that the user is presented with a map of the United States. Test 4.d will verify that the map has pins corresponding to the list in 4.a. Test 4.e will verify that the pins in 4.d may be clicked. Test 4.f will verify that the pins in 4.d, when clicked will output an image of the location. Test 4.g will verify that the pins in 4.d, when click will output a description of the location. Test 4.h will verify that the user’s avatar is presented on the homepage. Test 4.i will test the administrative permissions of the accounts. This will be broken into two separate tests. Test 4.i.1 will test if a user can look up another specific user by their username. Test 4.i.2 will test if an administrator account can loop up all users. Test 4 will assume that the user can reach the homepage. Test 4 will also assume that the list of locations was created by the questionnaire. Within testing, this list may be hardcoded with locations in the same format that they would be outputted from Test 3. This output is assumed to be verified through the tests checked in Test 3.a-c. Testing that the homepage will be done in a later test. Test 4 will be viewed as a ‘pass’ if Tests 4.a-g are passed.

Test 5 will test the integration of the CSCI components tested within Test 1-4. Test 5.a will verify that the user may reach the login page. Test 5.b will verify that the user may reach the create account page. Test 5.c will verify that the user may reach the homepage. Test 5.d will verify that the user may reach the modify account page. Test 5.d will verify the user may reach the questionnaire page. Test 5.e will verify that completion of the questionnaire page redirects the user to the homepage. Test 5.f will verify the output of the questionnaire is a list type. Test 5.g will verify that the map is originally presented on the homepage. Test 5 will be viewed as a pass if Tests 5.a-g are passed.

#### Requirements addressed

This paragraph shall identify the CSCI or system requirements addressed by the test case. (Alternatively, this information may be provided in [5.a.)](#_bookmark1)

This section will serve to map the requirements described in the SRS table to the test numbers described above. This will be done through a table mapping requirement number to test number.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Number | Requirement Summary | Test Number | Group Initials | Client Initials |
| 3.1 | State Requirements | 1-5 |  |  |
| 3.1.a | Login page | 1,5 |  |  |
| 3.1.a.1 | Access account creation | 5a,5b |  |  |
| 3.1.a.2 | Username input | 1.a |  |  |
| 3.1.a.3 | Password input | 1.b |  |  |
| 3.1.a.4 | Authentication | 1.d |  |  |
| 3.1.a.5 | Admin Permissions | 4.i.1-2 |  |  |
| 3.1.b | CAP | 5.b |  |  |
| 3.1.b.1 | Create username | 1.a |  |  |
| 3.1.b.2 | Unique username | 1.c |  |  |
| 3.1.b.3 | Create Password | 1.b |  |  |
| 3.1.b.4 | About me | 1.d |  |  |
| 3.1.b.5 | Avatar | 1.f |  |  |
| 3.1.c | Homepage | 5.c |  |  |
| 3.1.c.1 | Original Map | 5.g |  |  |
| 3.1.c.2 | Access modify account page | 5.b |  |  |
| 3.1.c.3 | Access questionnaire | 5.d |  |  |
| 3.1.c.4 | Updated Map | 4.c-f |  |  |
| 3.1.c.5 | Avatar Display | 4.h |  |  |
| 3.1.d | Modify Account Page | 2.a-e |  |  |
| 3.1.d.1 | Change username | 2.a |  |  |
| 3.1.d.2 | Change password | 2.b |  |  |
| 3.1.d.3 | Change about me | 2.d |  |  |
| 3.1.d.4 | Change Avatar | 2.f |  |  |
| 3.1.e | Questionnaire | 3.a-e |  |  |
| 3.1.e.1 | Display Statistics | 3.a |  |  |
| 3.1.e.2 | Rate statistics | 3.b |  |  |
| 3.1.e.3 | Updated Map Algorithm | 3.d,4,5 |  |  |
| 3.1.e.3.a | List top 10 | 3.a |  |  |
| 3.1.e.3.b | Pin Information | 4.f,4.g |  |  |
| 3.1.e.3.c | Multiple Questionnaires | 5.e |  |  |
| 3.2 | Capability Requirements | 1-5 |  |  |
| 3.2.a | Login Page | 1 |  |  |
| 3.2.a.1 | Successful login | 1.b |  |  |
| 3.2.a.2 | Unsuccessful login | 1.e |  |  |
| 3.2.b | Create Account | 1.d |  |  |
| 3.2.b.1 | Unique username | 1.c |  |  |
| 3.2.b.2 | Non-unique username | 1.c |  |  |
| 3.2.b.3 | Empty Username | 1.a |  |  |
| 3.2.b.4 | Empty Password | 1.b |  |  |
| 3.2.c | Homepage | 5.c |  |  |
| 3.2.c.1 | Empty Map | 5.g |  |  |
| 3.2.c.2 | Top 10 | 4.a |  |  |
| 3.2.c.3 | Pin Info | 4.f,4.g |  |  |
| 3.2.d | Modify Account | 2.a-e |  |  |
| 3.2.d.1 | Empty Username | 2.a |  |  |
| 3.2.d.2 | Unique Username | 2.c |  |  |
| 3.2.d.3 | Empty Password | 2.b |  |  |
| 3.2.e | Questionnaire | 3.a-e |  |  |
| 3.2.e.1 | Display Statistics | 3.a |  |  |
| 3.2.e.2 | Empirical Statistics | 3.a |  |  |
| 3.2.e.3 | Statistical Priority | 3.b |  |  |
| 3.2.e.4 | Varied Output | 3.d |  |  |
| 3.2.e.5 | Updated Map | 4.d |  |  |
| 3.5 | Database Requirements | 1 |  |  |
| 3.5.a | Database usernames | 1.d |  |  |
| 3.5.b | Database passwords | 1.d |  |  |
| 3.5.c | Database Avatar | 1.d, f |  |  |

#### Prerequisite conditions

This paragraph shall identify any prerequisite conditions that must be established prior to performing the test case. The following considerations shall be discussed, as applicable:

* + - * 1. Hardware and software configuration
        2. Flags, initial breakpoints, pointers, control parameters, or initial data to be set/reset prior to test commencement
        3. Preset hardware conditions or electrical states necessary to run the test case
        4. Initial conditions to be used in making timing measurements
        5. Conditioning of the simulated environment
        6. Other special conditions peculiar to the test case

Prerequisite testing should be completed with Test 5 and with the setup readme instructions. However, these will be described more explicitly here. The entire system will be running on a localhost:4200 after the setup is done. For an abnormal IP setup, the 4200 portion may be required to be modified based on the setup of the local machine. This modification will be stated after the npm run command. The following are the page dependencies for navigating the system. The create account page is dependent on access from the login page. The homepage is dependent on access from the login page. This homepage will have an empty map output. The questionnaire page is dependent on access from the homepage. The questionnaire upon completion should redirect the user to the homepage with the updated map output. The map pin image and descriptions depend on user completion of the questionnaire.

#### Test inputs

This paragraph shall describe the test inputs necessary for the test case. The following shall be provided, as applicable:

1. Name, purpose, and description (e.g., range of values, accuracy) of each test input
2. Source of the test input and the method to be used for selecting the test input
3. Whether the test input is real or simulated
4. Time or event sequence of test input
5. The manner in which the input data will be controlled to:

Test the item(s) with a minimum/reasonable number of data types and values

Exercise the item(s) with a range of valid data types and values that test for overload, saturation, and other "worst case" effects

Exercise the item(s) with invalid data types and values to test for appropriate handling of irregular inputs

Permit retesting, if necessary

The following describe the inputs for the tests. Any test which is testing NULL input should test an empty string, as well as no input. Validation testing of user input should also test values larger than the buffer described within the database. For example, the username has a varchar of size 255. A username of 256 should then be tested. To test modification to a database, automation testing should user a sql SELECT query and validate the value within the database. An entire profile should be verified by placing the user information into a list and validating the list of user information. Verification of the database information should use correctly formatted data whose format has been verified through the previous tests. Testing uniqueness of a value within a database should also be done through a sql SELECT query and validating that the new value is unique within the established data. This will require having sample data within the database for testing purposes. Any data visualization testing is assumed to not be automated. The testing of data visualization should be done through access to the web environment following the procedures described in the software environment setup.

#### Expected test results

This paragraph shall identify all expected test results for the test case. Both intermediate and final test results shall be provided, as applicable.

Automated testing should simply output a pass or fail to the developers. This is assumed to be done through the system’s standard output. Failed outputs should print a failure message as well as information pertaining to the test. This should include the function being tested, as well as a corresponding test number. Passing outputs should simply output a pass and the test being done. If a component is tested using multiple automated test, Boolean values should be used with each test, which will then be compared to validate the component. If a single test fails, then the component is to be viewed as failing.

Data visualization testing should be described within a test.txt file. These tests should be simple yes/no or true/false test which will be written out corresponding to the test described in section 3 of this document. The test.txt file should just be a description of the testing procedure for what a developer is looking to confirm is valid output. These test results should contain a global Boolean value within the automated testing, so that after visualization testing is concluded, automated testing can output a pass/fail value for system components.

#### Criteria for evaluating results

This paragraph shall identify the criteria to be used for evaluating the intermediate and final results of the test case. For each test result, the following information shall be provided, as applicable:

* + - * 1. The range or accuracy over which an output can vary and still be acceptable
        2. Minimum number of combinations or alternatives of input and output conditions that constitute an acceptable test result
        3. Maximum/minimum allowable test duration, in terms of time or number of events
        4. Maximum number of interrupts, halts, or other system breaks that may occur
        5. Allowable severity of processing errors
        6. Conditions under which the result is inconclusive and re-testing is to be performed
        7. Conditions under which the outputs are to be interpreted as indicating irregularities in input test data, in the test database/data files, or in test procedures
        8. Allowable indications of the control, status, and results of the test and the readiness for the next test case (may be output of auxiliary test software)
        9. Additional criteria not mentioned above.

Within this system all tests will be outputting a simple pass or fail result. The individual tests will be determining the procedure for what constitutes a pass or fail within the context of that test. The only test which will have ambiguity will be the tests which verify requirement 3.2.e.4: The questionnaire page shall yield different outputs for different user inputs. This test will be required to test at minimum a test for maximizing each statistical input. This would be 5 separate inputs. Each of these outputs shall be unique implying that none of the 5-outputted list should be equivalent. This can be tested by using a simple if(lista == listb == …) statement using all 5 lists. If at any point a test causes a website error, ie 404, then the entire test is to be determined to be an error. These errors should be determined to be noted as ‘severe.’ An error which is computational in nature should be determined to be ‘moderate.’ Severe errors should not be permitted at all. Moderate error should be reviewed for the implication of allowing such error to persist. If the error causes only a unnoticeable slowing of website performance it will be determined to be ‘allowable.’ If the error cause noticeable slowing of the website performance it should be determined to be ‘moderate.’ If the error causes non-usability of the website the error should be determined to be ‘severe.’

#### Test procedure

This paragraph shall define the test procedure for the test case. The test procedure shall be defined as a series of individually numbered steps listed sequentially in the order in which the steps are to be performed. For convenience in document maintenance, the test procedures may be included as an appendix and referenced in this paragraph. The appropriate level of detail in each test procedure depends on the type of software being tested. For some software, each keystroke may be a separate test procedure step; for most software, each step may include a logically related series of keystrokes or other actions. The appropriate level of detail is the level at which it is useful to specify expected results and compare them to actual results. The following shall be provided for each test procedure, as applicable:

1. Test operator actions and equipment operation required for each step, including commands, as applicable, to:
2. Initiate the test case and apply test inputs
3. Inspect test conditions
4. Perform interim evaluations of test results
5. Record data
6. Halt or interrupt the test case
7. Request data dumps or other aids, if needed
8. Modify the database/data files
9. Repeat the test case if unsuccessful
10. Apply alternate modes as required by the test case
11. Terminate the test case
12. Expected result and evaluation criteria for each step
13. If the test case addresses multiple requirements, identification of which test procedure step(s) address which requirements. (Alternatively, this information may be provided in [5.)](#_bookmark1)
14. Actions to follow in the event of a program stop or indicated error, such as:
15. Recording of critical data from indicators for reference purposes
16. Halting or pausing time-sensitive test-support software and test apparatus
17. Collection of system and operator records of test results
18. Procedures to be used to reduce and analyze test results to accomplish the following, as applicable:

Detect whether an output has been produced

Identify media and location of data produced by the test case

Evaluate output as a basis for continuation of test sequence

Evaluate test output against required output

This will describe the testing procedure for a developer. The tester should begin be setting up the test environment following the procedure in the setup readme file. The developer should then begin with any data visualization testing. This will be done to set the global Booleans within the automated test. All data visualization test should be initialized to ‘False’ unless a developer manually modifies the test to be passed, where they may then be set to ‘True.’ After completion of the data visualization testing, then the developer may proceed to run the automated tests. This should be done via a single automated test which calls any other automation tests. The final print statements of any automation test should include a description of the testing results. This should include a pass or fail for each of the CSCI components and the overall system.

#### Assumptions and constraints

This paragraph shall identify any assumptions made and constraints or limitations imposed in the description of the test case due to system or test conditions, such as limitations on timing, interfaces, equipment, personnel, and database/data files. If waivers or exceptions to specified limits and parameters are approved, they shall be identified and this paragraph shall address their effects and impacts upon the test case.

All assumptions for the test environment have been described within the tests.

# Requirements traceability

This paragraph shall contain:

1. Traceability from each test case in this STD to the system or CSCI requirements it addresses. If a test case addresses multiple requirements, traceability from each set of test procedure steps to the requirement(s) addressed. (Alternatively, this traceability may be provided in [4.x.y.1.)](#_bookmark0)
2. Traceability from each system or CSCI requirement covered by this STD to the test case(s) that address it. For CSCI testing, traceability from each CSCI requirement in the CSCI’s Software Requirements Specification (SRS) and associated Interface Requirements Specifications (IRSs). For system testing, traceability from each system requirement in the system’s System/Subsystem Specification (SSS) and associated IRSs. If a test case addresses multiple requirements, the traceability shall indicate the particular test procedure steps that address each requirement.

This was described in 4.1.1.1.

# Notes

This section shall contain any general information that aids in understanding this document (e.g., background information, glossary, rationale). This section shall include an alphabetical listing of all acronyms, abbreviations, and their meanings as used in this document and a list of any terms and definitions needed to understand this document.

# Appendixes

Appendixes may be used to provide information published separately for convenience in document maintenance (e.g., charts, classified data). As applicable, each appendix shall be referenced in the main body of the document where the data would normally have been provided. Appendixes may be bound as separate documents for ease in handling. Appendixes shall be lettered alphabetically (A, B, etc.).

DESCRIPTION/PURPOSE

The Software Test Description (STD) describes the test preparations, test cases, and test procedures to be used to perform qualification testing of a Computer Software Configuration Item (CSCI) or a software system or subsystem.

The STD enables the acquirer to assess the adequacy of the qualification testing to be performed.

APPLICATION/INTERRELATIONSHIP

Portions of this plan may be bound separately if this approach enhances their usability. Examples include plans for software configuration management and software quality assurance.

The Contract Data Requirements List (CDRL) should specify whether deliverable data are to be delivered on paper or electronic media; are to be in a given electronic form (such as ASCII, CALS, or compatible with a specified word processor or other support software); may be delivered in developer format rather than in the format specified herein; and may reside in a computer-aided software engineering (CASE) or other automated tool rather than in the form of a traditional document.

PREPARATION INSTRUCTIONS

General instructions.

a. Automated techniques. Use of automated techniques is encouraged. The term "document" in this means a collection of data regardless of its medium.

b. Alternate presentation styles. Diagrams, tables, matrices, and other presentation styles are acceptable substitutes for text when data required can be made more readable using these styles.

c. Title page or identifier. The document shall include a title page containing, as applicable: document number; volume number; version/revision indicator; security markings or other restrictions on the handling of the document; date; document title; name, abbreviation, and any other identifier for the system, subsystem, or item to which the document applies; contract number; CDRL item number; organization for which the document has been prepared; name and address of the preparing organization; and distribution statement. For data in a database or other alternative form, this information shall be included on external and internal labels or by equivalent identification methods.

d. Table of contents. The document shall contain a table of contents providing the number, title, and page number of each titled paragraph, figure, table, and appendix. For data in a database or other alternative form, this information shall consist of an internal or external table of contents containing pointers to, or instructions for accessing, each paragraph, figure, table, and appendix or their equivalents.

e. Page numbering/labeling. Each page shall contain a unique page number and display the document number, including version, volume, and date, as applicable. For data in a database or other alternative form, files, screens, or other entities shall be assigned names or numbers in such a way that desired data can be indexed and accessed.

f. Response to tailoring instructions. If a paragraph is tailored out of this document, the resulting document shall contain the corresponding paragraph number and title, followed by "This paragraph has been tailored out." For data in a database or other alternative form, this representation need occur only in the table of contents or equivalent.

g. Multiple paragraphs and subparagraphs. Any section, paragraph, or subparagraph in this DID may be written as multiple paragraphs or subparagraphs to enhance readability.

h. Standard data descriptions. If a data description required by this document has been published in a standard data element dictionary specified in the contract, reference to an entry in that dictionary is preferred over including the description itself.

i. Substitution of existing documents. Commercial or other existing documents, including other project plans, may be substituted for all or part of the document if they contain the required data.