Security Requirements and Use Case Test Plan

Team 1

Riley Dorough
Kayla Echols
Julia Wilkins
Brett Wolff

September 25th, 2021

Table of Contents

1. Introduction

1.1 Document Purpose

The purpose of this Security Requirements specification and Use Case Outline is to detail the requirements and constraints of the development project. Format derived from System Requirements Specification Template and System Test Specification Template by Sipantzi and Tucker (2021). Team One seeks to deliver the right product to the Central Virginia Liquefied Natural Gas (CVALNG) personnel and users with the utmost quality and satisfaction. Therefore, this document will serve as a binding repository of specific project features, requirements, and constraints. Furthermore, this document contains use cases and user documentation with supplemental diagrams to assist in understanding the software architecture and best practices. All of this is detailed below to ensure clarity of expectations for both the client and the development team.

1.2 Intended Audience

- Project Manager – The project managers will be responsible for facilitating requirements elicitation and negotiations with the client; this document will be produced, delivered, and updated by the project manager to this end.
- Development Team – The developers are responsible for adhering to this document to produce the correct product in a timely fashion according to customer expectations for features and performance.
- Client – The end user may use this document as an overview of features to expect and a reference for which helpful documentation is provided for guidance after project closeout.

1.3 Project Scope

The software in this document is a data historian used to read in the data from the OPTO22 hardware and provide a visual representation of the data for the Central Virginia Liquefied Natural Gas Incorporation. The data historian reads in various data that the

OPTO22 collects including the temperature and data from the switches. The project must use a SCADA framework to provide every requirement from the Request for Proposal checklist. The SCADA framework includes data acquisition, data communication, data presentation, and control. The users must be able to view the data from the data historian and access the servers to interact. The administrators must have access to control the database and servers.

1.4 Product Context

This project uses a SCADA framework to allow communication between the hardware and software used in this project and oversees the controls and operations. The OPTO22 hardware is read into a PLC and read into a database using SQL Server 2012. The data is then read from SQL Server 2012 into the data historian using the open historian software. The data historian then transfers the data into a readable presentation for the users and admin to view the data.

2. Security Requirements [James Dorough]

1. CVALNG Project shall meet modern LNG industry security standards
2. All records shall be kept private from all external parties except those with appropriate access to such records.
3. All records shall be kept private from all users except those with appropriate access to such records.
4. Network transactions shall be protected using SSL and HTTPS.
5. Users shall have restricted access to information as appropriate.
6. The system shall conform to applicable modern security requirements for data security.
7. The MVC SIS shall conform to any other applicable modern security requirements

3. Use Case Descriptions [Team 1]

Section 3 of the system requirements report details the use cases needed the section starts by stating the system's use cases and how each case applies to customer roles. Priorities will then be assigned to each use case as well as the cost of developing the case, how it benefits the system, and risks involved in implementing the case. Section 3 ends with a detailed description of each use case, and how each will be utilized by the whole system. These use case details will include which roles the use cases apply to, the case's post & preconditions, each main and exceptional workflow, and its priority. At the end of this section readers will understand what main actions this system will provide to the customer. The use cases focus on the STRIDE threat model.

Use Case	*Administrator*	*Standard User*	*Developer*	*Section #*
Log into Linux and Windows Server as Admin	✔			*3.1*
Log into Linux and Windows Server as Standard User		✔		*3.2*
Test the change data functionality as the admin account	✔			*3.3*
Users can view the data in the proper format	✔	✔		*3.4*
Changes that the admin users make can be viewed by everyone	✔	✔		*3.5*
Test the permissions for the user account		✔		*3.6*
Test the privileges for the admin account	✔			*3.7*
Checks machine navigation on different operating systems/ machines	✔			*3.8*
Manual testing of the code functionality			✔	*3.9*
Automated testing of the code			✔	*3.10*
Checks the communication between the various programs			✔	*3.11*

Tests the software for speed, response time, stability, scalability, and resource usage			✔	*3.12*
Checks to see if the device stored the input data in the database			✔	*3.13*
Receive Alerts for abnormal readings	✔	✔		*3.14*

Table 3.1 – Use Cases by Actor

3.1 *Log into Linux and Windows Server as Admin*

Log into Linux and Windows Server as Admin	**Description and Priority** ● Purpose: Tests if the administrator account can access the software with the proper permissions on both Linux and Windows operating systems ● Role Usage: The administrator account must be able to access the software effectively on multiple operating systems ● Priority (9): The rest of the system relies on this use case being operational ● Benefit (9): Brings value to the system by enabling the admins to enter the system on different operating systems ● Cost (0): Critical, System cannot be accessed without functionality ● Risk (4): Common entry point for many digital attacks. Authentication is important to keep the confidentiality of the system.
Precondition(s)	● System must be functional and active ● User must have an internet connection ● User must have the correct terminal and OS settings ● User must be able to locate program's entry point ● User has the correct credentials to enter system

	● The software is downloaded onto system ● User has credentials to log on to software
Main Flow of Event(s)	● User Action: Power on/ activate system ● System Action: System boots and loads log in screen ● User Action: Select fields and enter valid credentials, then opens software to test accessibility ● System Action: Validates credentials and proper permissions for the admin user
Postcondition(s)	● User is in main desktop or home directory ● User can issue appropriate commands ● User can open the software ● User can edit data if given permission to
Exceptional Flow of Event(s)	• User Action: Inputs invalid credentials • System Action: Reject login and display non detailed invalid credentials notice (Do not give out info about valid users or passwords)

3.2 *Log into Linux and Windows Server as Standard User*

Log into Linux and Windows Server as Standard User	**Description and Priority** ● Purpose: Tests if the user account can access the software with the proper permissions and restrictions on both Linux and Windows operating systems ● Role Usage: The user account must be able to access the software effectively on multiple operating systems ● Priority (9): The rest of the system relies on this use case being operational ● Benefit (9): Brings value to the system by enabling the users to enter the system on different operating systems ● Cost (0): Critical, System cannot be accessed without functionality ● Risk (4): Common entry point for many digital attacks. Authentication is important to keep the confidentiality of the system.
Precondition(s)	● System must be functional and active

	● User must have an internet connection ● User must have the correct terminal and OS settings ● User must be able to locate program's entry point ● User has the correct credentials to enter system ● The software is downloaded onto system ● User has credentials to log on to software
Main Flow of Event(s)	● User Action: Power on/ activate system ● System Action: System boots and loads log in screen ● User Action: Select fields and enter valid credentials, then opens software to test accessibility ● System Action: Validates credentials and proper permissions for the user
Postcondition(s)	● User is in main desktop or home directory ● User can issue appropriate commands ● User can open the software ● User can edit data if given permission to
Exceptional Flow of Event(s)	• User Action: Inputs invalid credentials • System Action: Reject login and display non detailed invalid credentials notice (Do not give out info about valid users or passwords)

3.3 *Test the change data functionality as the admin account*

Test the change data functionality as the admin account	**Description and Priority** ● Purpose: Check to make sure the GUI works properly and allows the admin to change data as permissions allow ● Role Usage: Changing the data will be useful for the admin ● Priority (7): If the data cannot be changed by proper accounts there is the risk of data integrity ● Benefit (9): Keeps data up to date by the admin ● Cost (0): Standard practice for all admin accounts when necessary ● Risk (7): Proper security must be maintained to restrict non authorized users from having access to make changes
Precondition(s)	● System must be functional and active

	● Admin must have an internet connection ● Admin must have server access and credentials ● Admin must have access to a CLI and/or GUI based user control application ● Admin has the correct and current credentials to enter software ● Admin has access to change the data within the software
Main Flow of Event(s)	● User Action: Logs into server ● System Action: Standby for user action ● User Action: Opens software and enters credentials ● System Action: Standby for user action ● User Action: Admin uses GUI to edit the data within the software ● System Action: Effectively changes the data to what the admin entered
Postcondition(s)	● The data is effectively changed
Exceptional Flow of Event(s)	• User Action: Enters incorrect credentials • System action: Error message appears on the screen and prompts user to try again

3.4 *Users can view the data in the proper format*

Users can view the data in the proper format	**Description and Priority** ● Purpose: Check to make sure that the data can be viewed by the users and admin in the proper format ● Role Usage: Viewing the data properly will help users read the data correctly ● Priority (8): If users cannot view the data properly, the data coming in will be messy and hard to read ● Benefit (9): Gives the software Grafana purpose by allowing users to view the data ● Cost (0): Standard practice for software functionality ● Risk (7): If the data is not being displayed in the correct format the data integrity could be lost.

Precondition(s)	● System must be functional and active ● User must have internet access ● User must have server access ● User has the correct and current credentials to enter system ● User can access the software to view the data
Main Flow of Event(s)	● User Action: Logs into account ● System Action: Standby for user action ● User Action: Opens software to view the data ● System Action: Prompt user to enter user information ● System Action: Confirms credentials are accurate ● System Action: Allows user to view the data
Postcondition(s)	● The data can be viewed from the software in the proper format with the correct data
Exceptional Flow of Event(s)	• User Action: Issues wrong command or opens wrong window • System action: Exception control begins and notifies of invalid command or opens selected window (not intended window) • User Action: Enters incorrect credentials • System Action: Prompts user to reenter credentials

3.5 *Changes that the Admin users make can be viewed by everyone*

Changes that the admin users make can be viewed by everyone	**Description and Priority** ● Purpose: The data that is changed by the admin users can be viewed by all users ● Role Usage: All users should be able to view the changed data ● Priority (7): If data needs to be altered by an admin for whatever reason, the changed data needs to be viewable by all users ● Benefit (9): Allows the modified data to be seen by all users in order to get accurate data ● Cost (5): The data should be viewable on the software ● Risk (7): The data could be entered incorrectly, or not be the updated data could not be viewable by other users risking data integrity

Precondition(s)	● System must be functional and active ● Admin must have an internet connection ● Admin must have access to change the data ● Admin has information to change the data on the software ● User and admin must be able to log in with correct credentials ● User and admin must have access to the software to view the data
Main Flow of Event(s)	● User Action: Logs into account ● System Action: Standby for user action ● User Action: Opens the software and admin changes data ● System Action: Successfully saves the data that was altered by the admin account ● User Action: Logs into a different account ● System Action: Standby for user action ● User Action: Opens the software and confirms that the changes to the data were saved and are viewable ● System Action: Standby for next user action
Postcondition(s)	● System shows updated data on the software to all users with access
Exceptional Flow of Event(s)	• User Action: Inputs incorrect data in the system • System action: Saves the altered data input by the admin • User Action: User logs in and views the incorrect data

3.6 Test the permissions for the user account

Test the permissions for the user account	**Description and Priority** ● Purpose: Enable the standard user accounts to have the proper permissions ● Role Usage: Standard users must be able to view the data but should not be able to write data to the system ● Priority (9): If the standard user needs to view the data they should be able to view it promptly ● Benefit (9): Allows the standard user to view the data and use it when they need to ● Cost (5): Software must be accessible to the user

	● Risk (9): Standard user accounts should not be allowed to change the data
Precondition(s)	● System must be functional and active ● Standard user must have an internet connection ● Standard user has the correct and current credentials to enter system ● Standard user has proper permissions put in place by the developers
Main Flow of Event(s)	● User Action: Logs into account ● System Action: Standby for user action ● User Action: Standard user opens the software to view the data ● System Action: Prompts user to log in ● User Action: Enters the correct credentials for the standard user account ● System Action: Standby for next user action ● User Action: User attempts to change the data ● System Action: An error message appears that the standard user does not have permission to do that action
Postcondition(s)	● The standard user can view the data but cannot change the data
Exceptional Flow of Event(s)	• User Action: Attempts to change the data • System action: The system allows the standard user to alter the data • User Action: Inserts data to the database without the proper permission • User Action: Inserts invalid credentials • System Action: An error message is shown to the screen

3.7 Test the privileges for the admin account

	Description and Priority
Test the privileges for the admin account	● Purpose: Enable the admin accounts to have the proper permissions ● Role Usage: Admin users must be able to view the data and should be able to write data to the system

	● Priority (9): If the admin user needs to view the data or alter the data they should be able to do so promptly ● Benefit (9): Allows the admin user to view the data and change it when they need to ● Cost (5): Software must be accessible to the user ● Risk (9): The admin accounts have the ability to change the data and could input incorrect data
Precondition(s)	● System must be functional and active ● User must have an internet connection ● User must have server access ● Admin user has the correct and current credentials to enter system ● User account must exist ● Proper permissions are already put in place by the developers
Main Flow of Event(s)	● User Action: Logs into account ● System Action: Standby for user action ● User Action: Admin user opens the software to view the data ● System Action: Prompts user to log in ● User Action: Enters the correct credentials for the admin user account ● System Action: Standby for next user action ● User Action: User attempts to change the data ● System Action: Allows changes to be made and saves the changes
Postcondition(s)	● The admin user can view the data and can change the data
Exceptional Flow of Event(s)	• User Action: Attempts to change the data with incorrect data • System action: The system allows the admin user to alter the data with the incorrect data • User Action: Inserts data to the database • User Action: Inserts invalid credentials • System Action: An error message is shown to the screen

3.8 Checks machine navigation on different operating systems/machines

Checks machine	**Description and Priority**

navigation on different operating systems/ machines	● Purpose: Tests the navigation for the software on different operating systems ● Role Usage: This uses case will only be performed by a system admin ● Priority (9): The software should be fully operational on multiple operating systems ● Benefit (9): Allows users to work on the system or view the data on the system from multiple operating systems ● Cost (3): Navigation should be a basic feature on the software ● Risk (9): There could be different vulnerabilities within the software on the different operating systems
Precondition(s)	● System must be functional and active ● User must have an internet connection ● User must have server access ● User must be logged into a server ● User account must exist
Main Flow of Event(s)	● User Action: Logs onto Windows Server system ● System Action: Authenticates User ● User Action: Open Server Manager ● System Action: Open software ● User Action: Can navigate through the data effectively ● User Action: Logs into Ubuntu Server ● Repeats the steps above
Postcondition(s)	● Machine navigation is effective on multiple operating systems
Exceptional Flow of Event(s)	• User Action: Logs into the Ubuntu server and attempts to open software • System Action: Does not allow the user to open software • User Action: Enters incorrect credentials into an operating system • System Action: Cannot log into the operating system

3.9 Manual testing of the code functionality

Manual testing of the code functionality	**Description and Priority** ● Purpose: Tests the functionality of the code for the database ● Role Usage: This user case will be tested by the developer ● Priority (9): In order for the software to properly input correct data from the hardware, the code must be correctly written for the database ● Benefit (8): Allows the data to be read from the hardware to the database ● Cost (8): The developer must have a strong understanding of the coding language (SQL) ● Risk (9): Developer could crash the system with poorly written code or code in a way that allows hackers to access the system
Precondition(s)	● System must be functional and active ● Developer must have an internet connection ● Developer must have access to the database code ● Developer must understand the SQL code
Main Flow of Event(s)	● User Action: Opens the SQL code ● System Action: Launches SQL code ● User Action: Developer runs tests to make sure the code functions properly ● System Action: The code runs as the developer types in the commands
Postcondition(s)	● The code runs efficiently and effectively.
Exceptional Flow of Event(s)	• User Action: Opens SQL • System Action: Prompts for log in credentials • User Action: Inputs invalid username & password • System Action: Prompts User for username & password again • System Action: After so many attempts display an error message. • User Action: Opens SQL and enters credentials • System Action: Opens the code for SQL

	• User Action: Runs the code to test functionality • System Action: Prints an error message that the code doesn't work

3.10 Automated testing of the code

Automated testing of the code	**Description and Priority** ● Purpose: Tests the functionality of the code for the database using an open source automated testing ● Role Usage: This user case will be tested by the developer ● Priority (9): In order for the software to properly input correct data from the hardware, the code must be correctly written for the database ● Benefit (8): Allows the data to be read from the hardware to the database ● Cost (8): The open source code must be downloaded on the system and work effectively to make sure the code has no errors ● Risk (9): The developer could code in a way that allows hackers to access the system
Precondition(s)	● System must be functional and active ● Developer must have an internet connection ● Developer must have access to the database code ● The open-source software must be downloaded, accessible, and compatible with SQL
Main Flow of Event(s)	● User Action: Opens the SQL code ● System Action: Launches SQL code ● User Action: Opens open-source automated code review ● System Action: Launches open-source automated code review ● User Action: Developer runs tests using the automated code review ● System Action: The code runs
Postcondition(s)	● The code runs efficiently and effectively.
Exceptional Flow of Event(s)	• User Action: Opens automated code review • System Action: Launches automated code review

	• User Action: Opens SQL and enters credentials • System Action: Opens the code for SQL • User Action: Runs the code to test functionality • System Action: Prints an error message that the code doesn't work

3.11 Checks the communication between the various programs

Checks the communication between the various programs	**Description and Priority** ● Purpose: Tests the communication between the various software ● Role Usage: This will be tested by the developer to allow communication between the different software ● Priority (9): For any of the data to be viewed or edited by users, the software must be able to effectively communicate to each other ● Benefit (9): By establishing the connection between the various software, the data can be used as needed ● Cost (7): Requires compatibility between the various software being used ● Risk (9): If one software is not compatible with another, the data will not be accessible
Precondition(s)	● System must be functional and active ● The different software must be downloaded and connected to each other ● Developer must have an internet connection ● Developer must have server access
Main Flow of Event(s)	● User Action: Developer logs into server and opens the various software ● System Action: Opens the software requested ● User Action: Checks the raw data, the database data, the open Historian, and the software to present the data to verify that it is consistent ● System Action: Allows user to switch between software

Postcondition(s)	● All the data is the same on each software to show that the programs communicate effectively
Exceptional Flow of Event(s)	• User Action: Opens software to view data input • System Action: Opens software as requested • User Action: Checks the data consistency • System Action: Shows different data in each software • System Action: The software fails to communicate to one another effectively

3.12 Tests the software for speed, response time, stability, and resource usage

Tests the software for speed, response time, stability, scalability, and resource usage	**Description and Priority** ● Purpose: Tests the software for various performance tests to get the most use out of the software ● Role Usage: The software will be used by both the standard user and the admin and fast and reliable software is important financially ● Priority (7): For the software to be used long term it must be fast and stable ● Benefit (9): The purpose of the software running smoothly is to keep users from delaying in getting their work done ● Cost (4): Software speed is important ● Risk (6): The tasks of the workers using the program could be delayed based on how quickly the response time is for the software being used.
Precondition(s)	● System must be functional and active ● Developer must have an internet connection ● System must have software downloaded and ready to use ● Developer must have credentials to get into system
Main Flow of Event(s)	● User Action: Logs into the server ● System Action: Verifies credentials ● User Action: Opens software and begins running the program ● System Action: Runs software ● User Action: Runs tests such as timing how long the software takes to run

Postcondition(s)	● The developer gets the results from the software speed, response time, stability, and resource usage
Exceptional Flow of Event(s)	• User Action: Attempts to open software • System Action: The software cannot be accessed because it was downloaded properly

3.13 Checks to see if the device stored the input data in the database

Checks to see if the device stored the input data in the database	**Description and Priority** ● Purpose: Keeps a backup of all the output data in a database ● Role Usage: Each user has access to view this database in case something goes wrong, and the data does not match. ● Priority (9): A backup in case it needs to be used ● Benefit (9): Allows the users and admins to view the data after it has been put through the software ● Cost (7): The database must be set up properly to read in the data from the data historian ● Risk (9): The device must store the data properly in the database, data integrity is at risk
Precondition(s)	● System must be functional and active ● Developer must have an internet connection ● Developer must have access to the database ● Developer must have active credentials ● Developer must have access to the software
Main Flow of Event(s)	● User Action: Logs into server ● System Actions: Checks credentials awaits further action ● User Action: Opens the database that stores the input data from the device ● System Action: Opens database ● User Action: Checks database to make sure the data is accurate to the device
Postcondition(s)	● The data is successfully entered into the database.
Exceptional Flow of Event(s)	• User Action: Opens the database to view the data • System Action: The database cannot be opened because it is not working properly

	• User Action: Opens database to view data • User Action: Compares data to the device and database and notices differences between the data

3.14 Receive alerts for abnormal readings

Receive Alerts for abnormal readings	**Description and Priority** ● Purpose: Allows user to receive email and text notifications in the event of abnormal readings ● Role Usage: The “Log In” user case will be used by the standard user and the system administrator ● Priority (9): The purpose of the system is to monitor and if the data cannot be viewed then the system serves no purpose. ● Benefit (9): Brings value to the system by enabling the users to enter the system ● Cost (5): Entry portal is not resource intensive ● Risk (9): User case will hinder the production of other elements if not completed first
Precondition(s)	● System must be functional and active ● User must have an internet connection ● User must have a working email address and cellphone number.
Main Flow of Event(s)	● System Action: Detection program notices abnormal data ● System Action: Sends email and text alert saying what type of data and from what sensor
Postcondition(s)	● If abnormal data reading is not resolved in 1 hours another alert will be sent out.
Exceptional Flow of Event(s)	• If messages are not able to be sent, then the system will continue to push messages until they are received.

4. Test Case Descriptions [Brett Wolff]

Test Case #	Test Scenario	Use Case ID	Steps	Test Data	Expected Results
1	Functionality Test Case: Testing whether various users (standard user and administrator) can log into different operating systems.	3.1/3.2	1. Log into system different times as various users 2. Open software to view data 3. Log into software as the various users 4. Test whether software works effectively and can be viewed on different operating systems 5. Tests to make sure each user has the proper permissions to keep confidentiality and authentication	User email: User1@cvalng.com Password: Password1 Admin email: Admin1@cvalng.com Password: Admin1	Each user should be able to log on from different operating systems
2	User Interface Test Case: The log in GUI works effectively and the GUI for the changes that the admin user can make.	3.1/3.2/3.3	1. Log into the system as various users at different times (admin and user) 2. Open software to test the login page GUI 3. As the admin, check that the changes that can be made work properly	User email: User1@cvalng.com Password: Password1 Admin email: Admin1@cvalng.com Password: Admin1	The GUI should work properly and allow the user to enter in credentials on the main login page. The admin account should be able to effectively edit the data within the GUI.
3	Unit Test Case: The developer tests the	3.9/3.10	1. Tester should activate both tools to automatically run through the code	User email: User1@cvalng.com Password:	The code review should find close to no errors that

	code using both automated and manual testing techniques.		and the tools to manually run through the code 2. Tester then runs the automatic test. 3. Tester saves report of results of automatic test, then starts manual test to verify automatic test results.	Password1 Admin email: Admin1@cvalng.com Password: Admin1	would hinder the use, and the manual test should show the same results as the automatic.
4	Integration Test Case	3.11	1. Tester logs into the database and the network. 2. Tester sends logs into the database. 3. Tester follows the test logs and verifies accuracy and integrity of flow.	User email: User1@cvalng.com Password: Password1 Admin email: Admin1@cvalng.com Password: Admin1	The data should be communicated cleanly and as is, with no man in the middle intercepting the data in transit.
5	Performance Test Case	3.12	1. Tester sets up a large-scale data for input. 2. Tester sends data to the systems 3. Tester creates a report based on the speed and accuracy the software was able to read and place data into database.	User email: User1@cvalng.com Password: Password1 Admin email: Admin1@cvalng.com Password: Admin1	Software would be easily able to handle as much data as is needed. Should the need for more arise, the software should be able to be upgraded to handle a larger amount easily.
6	Security Test Case	3.11/3.14	1. Tester sets up live test for constant data supervision. 2. Tester sets up monitoring software	User email: User1@cvalng.com Password: Password1 Admin email:	Data should have full integrity. There should be no leakage of data. If there

			and reporting software. 3. Tester runs a live test and saves reports.	Admin1@cvalng.com Password: Admin1	are any errors, there should be an alert for the proper people to deal with.
7	Database Test Case	3.3/3.4/3.13	1. Tester logs into database as both a standard user and admin user. 2. Tester runs through tester outlined in Table 3.1	User email: User1@cvalng.com Password: Password1 Admin email: Admin1@cvalng.com Password: Admin1	Proper errors should be shown when trying to do unauthorized actions, and no data should be shown if user does not have the proper credentials. Database should be able to properly save and hold data.
8	Usability Test Case	3.1-3.8	1. Tester follows normal use scenarios.	User email: User1@cvalng.com Password: Password1 Admin email: Admin1@cvalng.com Password: Admin1	The software should be able to easily handle everyday tasks. Errors should be clear and concise and not give away any data.

5. **Testing Plan** [James Dorough]

All tests are to be completed with clear documentation. Each test case should be understood with its intent, controlling parameters, and output. Test results which show vulnerabilities or pose any form of security risk as seen in the STRIDE threat model are to be flagged and prioritized for system adaptation. Version control is crucial here to ensure that older systems do not pass security risks down to subsequent generations. Careful

observation of second order effects should be implemented to understand how altering an aspect of the system can impact testing of another aspect of the system.

References

Sipantzi, T., Tucker, R. (2021). System Requirements Specification Template. CSIS 471: Software Development.