Software Requirements Specification

for

COD3BR3AKR

Version 1.1 approved

Prepared by Abdul Alqahtani, Alex Calis, Jacob (Shijie) Liu

Hood College
Department of CS & IT
Frederick Maryland

10/18/2018

Table of Contents

Revision History

Name	Date	Reason For Changes	Version
A. Calis, A. Alqahtani, J. Liu,	10-08-18	Initial Release	1.0
A.Calis	12-20-18	Updates per instructor's corrections	1.1

1.Introduction

1.1.Purpose

This Software Requirements Specification (SRS) document explains the requirements for the product named “COD3BR3KR” (also referred to as “CodeBreaker”, “CodeBrekr” or “Product”). The Software Version is 1.0 with release number 1. This is a stand-alone software system that will run on a Windows OS. The scope of the SRS covers the entire product, which is a single software application.

1.2.Document Conventions

This document is written with Arial font using size 11. There is no highlighting, italicize, bold, or other text formatting to highlight or add special significance to the meaning of the text.

1.3.Intended Audience and Reading Suggestions

This document is intended for multiple audiences, such as the end customer (and user), developers, testers, and documentation writers.

The following subsections (1.4 and 1.5) will explain Product Scope and References.

The sections after that explain the Overall Description of the product (including Perspective, Functions, User Classes and Characteristics, Operating Environment, Design and Implementation Constraints, User Documentation, and Assumptions and Dependencies), the External Interface Requirements (including User, Hardware, Software, and Communications interfaces), System Features (explaining lower level details for each feature), Nonfunctional Requirements, and Other Requirements. Finally, the end of the document includes Appendixes A (Glossary), B (Analysis Models), and C (To Be Determined List).

The suggested reading sequence for all audiences is to follow the document from the beginning to the end, without skipping or jumping around in sections. This way the reader can understand the product scope at a high level (Section 2) before diving into more detailed requirements and features (Sections 3, 4, 5 and 6). The Appendixes may be helpful resources for understanding the product better, and can be viewed at any point while reading this SRS.

1.4.Product Scope

Codebreaker is a stand-alone software encryption-decryption system in which a user can either input plaintext data (via string or file input) to be encrypted to ciphertext, or input ciphertext (via string or file input) to be decrypted to plaintext. It can be used as a method to protect user data using cryptographic technologies.

1.5.References

The SRS does not reference any external documents or web addresses.

2.Overall Description

2.1.Product Perspective

The CodeBreaker is a new, self-contained product that runs on a Windows OS.

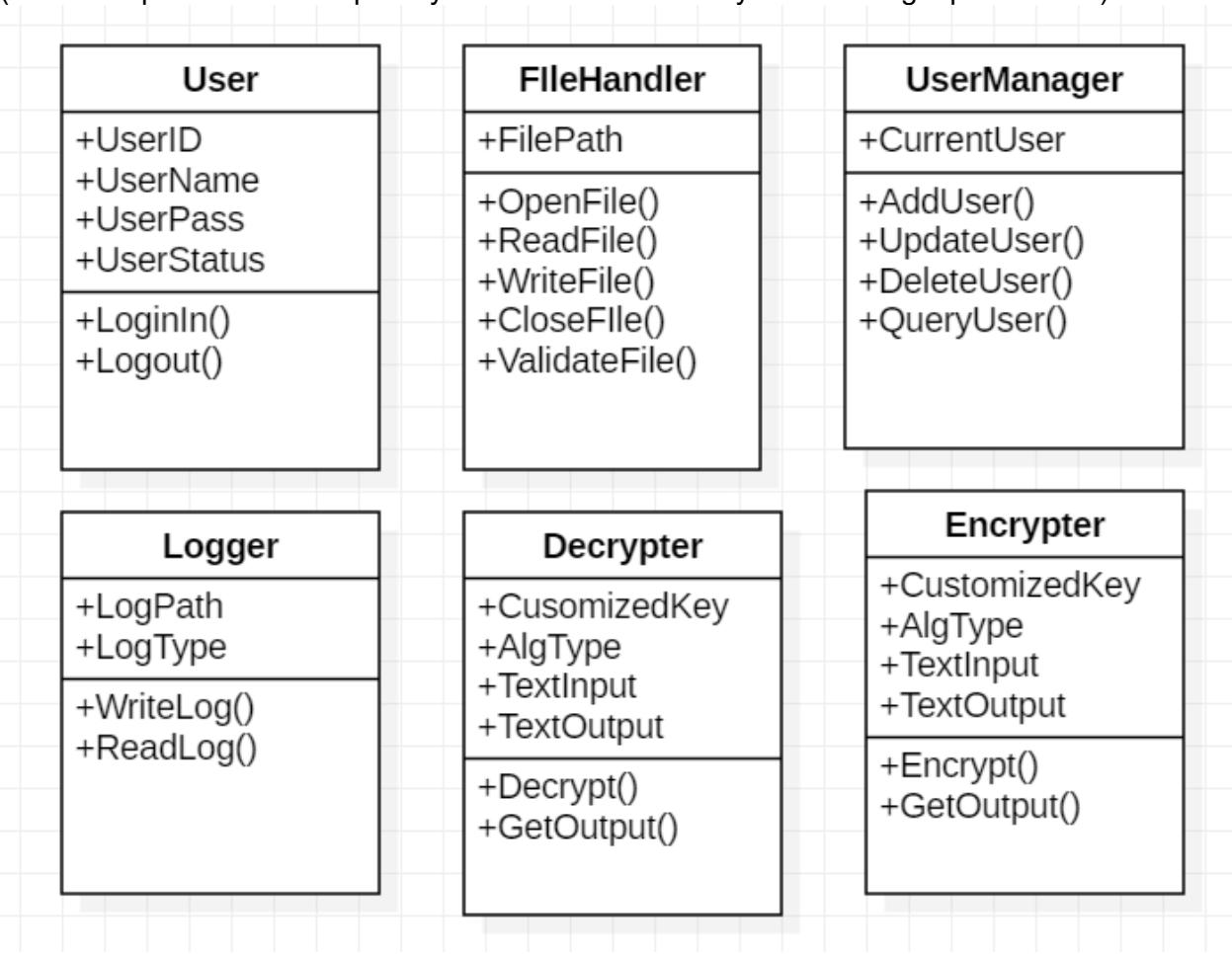
2.2.Product Functions

The main functions of the product are:

1. Provide Authentication mechanisms for logging in to the system
2. Encrypt user data and output the result
3. Decrypt user data and output the result
4. Provide an interface for managing users of the system
5. Provide an interface for viewing logs for user activity and error reporting

2.3.User Classes and Characteristics

The following is a diagram of the user classes of the module. Within each of these classes, there are the class properties (which are preceded with a plus symbol) and class methods (which are preceded with a plus symbol and succeeded by a left and right parenthesis)



- The User class holds the information of a User and assigns a user a unique ID. A user has an ID, user name, password, and status (active or inactive).
- Upon initializing via a file path, the FileHandler class can be used to open, read, write, close, and validate the file.
- The UserManager class provides the user the ability to add, update, delete or query a user profile.
- The Logger class provides a method to write and read log reports.
- Finally, the Encrypter and Decrypter classes are used to encrypt and decrypt data, respectively, using a customized key, with a specific algorithm (such as AES or Triple-DES) with methods to encrypt (for the Encrypter class), decrypt (for the Decrypter class) and get output (for both classes).

2.4.Operating Environment

The CodeBreaker will operate on Windows OS running on a General-Purpose Computer (GPC). The product does not depend on other dedicated software or hardware components or applications in order to function correctly.

2.5.Design and Implementation Constraints

The system shall be implemented using the limitations/constraints of Microsoft Visual Studio.

2.6.User Documentation

The product will be delivered with the following documentation:

1. Installation Guide –how a user or admin installs the system on a GPC running a Windows OS.
2. User's Guide (with screenshots) –how a user operates the system and what are the main features and functionalities.
3. Developer's Guide – how the system is build, the architecture, and the framework. The guide should be able to explain the system to any developer, regardless on experience with the system.

2.7.Assumptions and Dependencies

The CodeBreaker may be delivered using two third-party components, in which the components are assumed to function as expected. The specific components are not yet known, but once identified, the developers will integrate the components, and explain the components in the Developer document.

The CodeBreaker may also be delivered using a component developed by another software development team which the CodeBreaker developers may work closely together with. The specific component is not yet known, but once identified, the developers will integrate the component, and explain the component in the Developer document.

3.External Interface Requirements

3.1.User Interfaces

The Graphical User Interface (GUI) shall be implemented with windows form application controls.

The first step is going to be to log into the system. The user interface will provide the ability to enter a user name and password to be authenticated. If authentication is successful, the interface will provide access to the services provided by the system, such as encryption, decryption, user management, and accessing to log report.

Once logged in, the interface will have four tabs at the top the screen, which will allow the user to use different features of the program.

The first tab is called “Encryption”, which will have two methods to input plaintext data to be encrypted: 1) String input field; and 2) a file input browse button. Once the input is logged, user shall be able to click on the Encryption button.

The second tab is called “Decryption”, which will have two methods to input cyphertext data to be decrypted: 1) String input field; and 2) a file input browse button. Once the input is logged, user shall be able to click on the Decryption button.

The third tab called “User Management”, which allows the user to manage the full list of system users. The user shall be able to Create, Read, Update and Delete users.

The fourth tab allows the user to view system logs, including error, activity, and status logs.

3.2.Hardware Interfaces

Not applicable as the system does not interact with external hardware.

3.3.Software Interfaces

The system shall use Windows OS Windows 7 or above to run the application and Microsoft .NET Framework version 3.5 or above.

3.4.Communications Interfaces

Not applicable as the system does not communicate with external components.

4.System Features

4.1.Accept Plaintext Data, Encrypt the Data and Output the Result

4.1.1 Description and Priority

A user shall be able to input a plaintext data string and the system will encrypt the data using a single algorithm, then output the ciphertext to the user. This is of High priority.

4.1.2 Stimulus/Response Sequences

The user will input the plaintext data in only one of two ways: 1) Via a text file; or 2) Via manually entered string input. Then, the system will encrypt this plaintext input using a single encryption algorithm and output the results to the user in only one of two ways: 1) Via a text file; or 2) Via a system view that displays the ciphertext data to the user.

4.1.3 Functional Requirements

REQ-1: The user shall be able to manually input plaintext data as a string that is no more than 300 characters in length. The system shall support US-ASCII symbols as string input. The system shall reject input that is not a string or that is not part of the US-ASCII symbol set before the operation is performed. There is no limit to character combinations, as long as they fall within the symbol set of US-ASCII and are no more than 300 characters in length.

REQ-2: The user shall be able to manually input plaintext data as a text file. Other, non-text files shall be rejected by the system before the file is loaded onto the system. The system shall support US-ASCII symbols as string input and reject strings that are not part of this symbol set before the operation is performed. There is no limit on the size of the file (such as 500KB, 3 MB, etc.). There shall not be a limit on the file extension name, as long as the file is a text file. The system shall only accept one file input at a time, not multiple files. There is no limit to character combinations, as long as they fall within the symbol set of US-ASCII.

REQ-3: The user shall input the encryption key (if required for encryption on a specific algorithm). The system will use this key and plaintext data to encrypt the plaintext data for output.

REQ-4: The user shall have the option to choose which encryption algorithm to use for encrypting the plaintext data. For example, the user might have the choice between either the AES-128 ECB or Triple-DES ECB for data encryption.

REQ-5: The system shall implement at least two encryption algorithms if the algorithms are considered “strong” per the client definition. If the algorithms are too weak to be considered “strong” per this definition, then the system shall implement at least three encryption algorithms. This requirement is directly related to REQ-20.

REQ-6: Upon user request, the system shall be able to output the ciphertext data (after encryption is performed) to a text file available to the user. The data in the output

file shall only include content of the encryption (i.e. the ciphertext), but nothing else (i.e. user data, key, error log, etc.).

REQ-7: Upon user request, the system shall be able to output the ciphertext data (after encryption is performed) to a system view available to the user within the application. The data in this view shall only include content of the encryption (i.e. the ciphertext), but nothing else (i.e. user data, key, error log, etc.).

4.2.Accept Ciphertext Data, Decrypt the Data and Output the Result

4.2.1 Description and Priority

A user shall be able to input a ciphertext data string and the system will decrypt the data using an algorithm, then output the plaintext to the user. This is of High priority.

4.2.2 Stimulus/Response Sequences

The user will input the ciphertext data in only one of two ways: 1) Via a text file; or 2) Via manually entered string input. Then, the system will decrypt this ciphertext data using one decryption algorithm and output the results to the user in only one of two ways: 1) Via a text file; or 2) Via system view that displays the plaintext data to the user.

4.2.3 Functional Requirements

REQ-8: The user shall be able to manually input ciphertext data as a string that is no more than 300 characters in length. The system shall support US-ASCII symbols as string input. The system shall reject input that is not a string or that is not part of the US-ASCII symbol set before the operation is performed. There is no limit to character combinations, as long as they fall within the symbol set of US-ASCII and are no more than 300 characters in length.

REQ-9: The user shall be able to manually input ciphertext data as a text file. Other, non-text files shall be rejected by the system before the file is loaded onto the system. The system shall support US-ASCII symbols as string input and reject strings that are not part of this symbol set before the operation is performed. There is no limit on the size of the file (such as 500KB, 3 MB, etc.). There shall not be a limit on the file extension name, as long as the file is a text file. The system shall only accept one file input at a time, not multiple files. There is no limit to character combinations, as long as they fall within the symbol set of US-ASCII.

REQ-10: The user shall input the decryption key (if required for decryption), The system will use this key and ciphertext data to decrypt the ciphertext data for output.

REQ-11: The user shall have the option to choose which decryption algorithm to use for decrypting the ciphertext data. For example, the user might have the choice between either the AES-128 ECB or Triple-DES ECB for data decryption.

REQ-12: The system shall implement at least two decryption algorithms if the algorithms are considered “strong” per the client definition. If the algorithms are too weak to

be considered “strong” per this definition, then the system shall implement at least three decryption algorithms. This requirement is directly related to REQ-20.

REQ-13: Upon user request, the system shall be able to output the plaintext data (after decryption is performed) to a text file available to the user. The data in the output file shall only include content of the decryption (i.e. the plaintext), but nothing else (i.e. user data, key, error log, etc.).

REQ-14: Upon user request, the system shall be able to output the plaintext data (after decryption is performed) to a system view available to the user within the application. The data in this view shall only include content of the decryption (i.e. the plaintext), but nothing else (i.e. user data, key, error log, etc.).

4.3.Authentication

4.3.1 Description and Priority

The system shall require user authentication before providing access to the system. This is of High priority.

4.3.2 Stimulus/Response Sequences

The user shall log in to the system before gaining access to any system features, services, or data. This shall be done via a user name and user password.

4.3.3 Functional Requirements

REQ-15: The system shall require authentication via a username and password before allowing access to the system resources, features, or services. There is no limit or restraints on username or passwords (such as length, strength, character set, etc.). There is also no limit on failed attempts, meaning a user can fail authentication any number of times without getting locked out or require password resetting. There is no requirement for system verification, besides a username and password.

REQ-16: The system does not need to support concurrent users, but does need to support multiple, uniquely identified users.

REQ-17: There is no requirement for the system to enforce access control. In other words, every user identity will have the same, full access to the system.

REQ-18: OPTIONAL As an optional requirement, the system shall provide a password reset option.

4.4.User Management Interface

4.4.1 Description and Priority

The system shall provide a management user application interface, giving the user the ability to add, remove, or edit users of the system. This is of High priority.

4.4.2 Stimulus/Response Sequences

Once logged in, the user shall have the option to view the user management interface for managing user roles.

4.4.3 Functional Requirements

REQ-19: The user shall be able to view, within the application, a User Management interface, in which the user can add, remove, and/or edit user roles.

4.5.Cryptographic Functionality and Limitations

4.5.1 Description and Priority

The system shall provide pre-defined options on how to use the cryptographic functionality of the system. This is of High priority.

4.5.2 Stimulus/Response Sequences

Once logged in, the user shall be given limited cryptographic selections and workflow options, which the system shall enforce.

4.5.3 Functional Requirements

REQ-20: The system shall support at least three “simple” algorithm options, or two “strong” algorithm options. Examples of a simple algorithm is a character (or bit) shift or swap. Examples of a strong algorithm are industry recognized cryptographic algorithms such as AES, Triple-DES, etc. Strong algorithms may be something besides just industry recognized algorithms, but it has to be complex enough not to be a straightforward one-operation encryption/decryption algorithm.

REQ-21: The system shall support both encryption and decryption operations for each encryption algorithm implemented by the system.

REQ-22: The user shall only be allowed to perform encryption or decryption, but not both, for a given use case scenario. Therefore, while the system shall support both encryption and decryption for a particular algorithm, the system shall only allow a single encryption or decryption operation that results in an output, per use case scenario.

REQ-23: The user shall only be allowed to choose a single algorithm per use case scenario. The system shall restrict picking multiple algorithms for the same cryptographic operation.

4.6.System and Error Logs

4.6.1 Description and Priority

The system shall track and store system activities and error log, available to the user.

4.6.2 Stimulus/Response Sequences

Once logged in, the user shall be given the option to view the system activities and error logs or previous and current events which the system has tracked automatically.

4.6.3 Functional Requirements

REQ-24: The system shall track an activity log of all users of the system. This includes, for each user, activities such as logged in status (successful or not), which algorithm was used during the cryptographic operation (AES or Triple-DES for example), what type of conversion (encrypt or decrypt), what input/output options were used (file or raw string input), was the cryptographic function successful or not, logged out status (successful or not). This log does not include contents (such as encryption or decryption data results). In addition, timestamps are required for all events mentioned above. Failed activity events (such as logged in attempts, wrong algorithm chosen, unsuccessful cryptographic operation, etc.) result in an “errors” rather than “statuses” which results from successful activities.

REQ-25: The system shall implement a status log for each algorithm. That is, the log shall show the time taken for an algorithm to complete the cryptographic operation using a specific algorithm, key, and input. For example, the log would supply the following info: time it took for an AES algorithm to complete the encrypt operation using key “1234” and input “hello world”.

REQ-26: The system shall log activity while performing tasks in real time. For example, if the system crashes unexpectedly, the log file(s) shall provide evidence as to what processes completed successfully or which may have failed, and what activity has completed up to the point of failure. Therefore, an implicit requirement for the system is to save logs after each event is recorded.

REQ-27: The timestamps in all log files shall granular to the liking of the client. For example, time between algorithm operations may be in micro or nanoseconds, when time of user login may be in date down to seconds only.

REQ-28: The system shall provide a location to view logs. That is, activity, status, and error logs shall be stored and available for viewing, such as saved to a log file.

5.Other Nonfunctional Requirements

5.1.Performance Requirements

REQ-29: The system shall operate under a Windows OS.

REQ-30: The system shall calculate the time consumed for each encryption/decryption and report it on the user interface.

REQ-31: The system shall log events as to indicate system performance, such as user activities, system status, and error logs.

5.2.Safety Requirements

REQ-32: The system shall provide warnings to the user interface upon application being accidentally closed during an operation.

5.3.Security Requirements

REQ-33: The system shall require user to authenticate before gaining access to any system features besides log in and reset password (if implemented).

REQ-34: The system shall not have any restrictions on password length or password characters chosen when setting the password.

5.4.Software Quality Attributes

Reliability:

REQ-35: The system shall be reasonably reliable, in that is it assumed to be available and functioning correctly at any given time.

Usability:

REQ-36: The system shall be presented with a Graphical User Interface that is user friendly and easy to navigate.

REQ-37: Upon selecting an algorithm for encryption or decryption that requires a secure key, the system shall give the user the ability provide the customized secure key.

Robustness:

REQ-38: The system shall provide user error messages upon invalid data input entries, such as non-valid text file, non-ASCII character input, failed operations (such as cryptographic failures or navigation errors), or other errors that will guide the user to correct errors rather than crash the system.

Maintainability:

REQ-39: The system shall log all the user management activities, which includes creating, removing, updating.
For example:
2018-10-12 18:30:25.250 – NOTICE: Jacob added a new user “Mark”.

REQ-40: The system shall log activities that are being performed during normal operation for all users, and the log entry shall include timestamp, username, activity, and result of activity. For example,
2018-10-13 13:30:25.250 – NOTICE: Joe performed a successful encryption of a file.
2018-10-14 09:40:12.230 – ERROR: Jack failed to Login in the system due to invalid password.
2018-10-14 09:40:12.230 – NOTICE: John successfully Logged In to the system.

Portability:

N/A

5.5.Business Rules

REQ-41: The system shall enforce user to have username and password before using the system features.

REQ-42: Any user in the system has the ability to add, delete and update user accounts.

REQ-43: All active users shall have same operation permissions within the system.

6.Other Requirements

Capacity Requirements:

REQ-44: The system shall only support one operation at a time. In other words, encryption and decryption shall not be running at the same time.

REQ-45: While processing file input, only one file is supported at a time, multiple files selection shall be disabled.

Documentation Requirements:

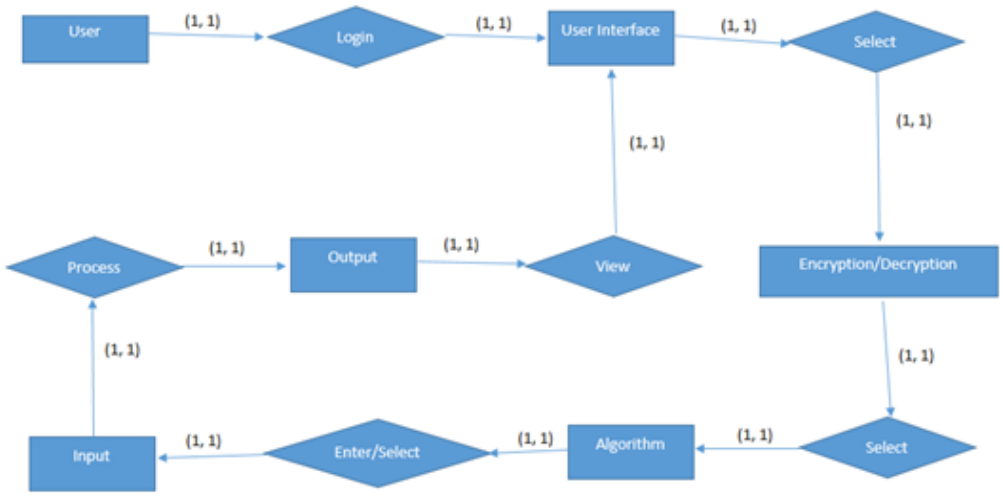
REQ-46: The system shall provide a user guide and developers guide along with the installation instructions as part of the system package.

Appendix A: Glossary

Secure Key: A key needed while performing encryption or decryption as prerequisite.

Appendix B: Analysis Models

Entity-Relationship Diagrams (Encrypt/Decrypt):



Appendix C: To Be Determined List

There is no list of the TBD (to be determined) references that remain in the SRS so they can be tracked to closure.