Encryption Recommendation for SMEs

Functional Specification

by

Kewin Skuza

Student ID: C00237361

Project Supervisor: Christopher Staff

Date: 25th April 2022

Abstract

Every project requires a foundation to build on. The purpose of a functional specification is to describe what the project is about and my plans for the development process. This application is going to view the data supplied and recommend which fields are sensitive and should be encrypted. I will use an Amazon Web Services service named Macie to do the main chunk of detection. More smaller functions will be developed to guide it. Data will be display in a desktop application GUI developed using Tkinter. The results and application should be self-explanatory and easy to understand so that people of varied technical background can use it.

Table Of Contents

Contents

The Product

I am developing an application that will analyze data inside a database. Upon analysis the application will recommend which columns should be encrypted. The encryption recommendation should be fully compliant with GDPR and compare data pairs to make sure that all Personal Identifiable Information is found. The application will be a standalone system that is not web-based. I will attempt to integrate a machine learning algorithm that will automatically detect all that sensitive information without the need for rule-based patterns that take a long time to develop and will not encompass the full scope of sensitive information. All information that should be encrypted will be displayed on screen in a neat, columnar manner that corresponds to the way it is displayed in the database. Alongside the recommendations I will propose best practices for encryption and hashing to promote data security and make sure that all information will not be compromised when a system breach occurs.

Functionality

Core Functions

- The machine learning algorithm should be educated appropriately to generate reliable results.
- The application should be easy to operate and self-explanatory.
- The user should be able to include database files in the application to be processed.
- All recommendations should be displayed with appropriate headings in a tidy manner.
- The application will be a standalone system and not a web application.

Optional Functions

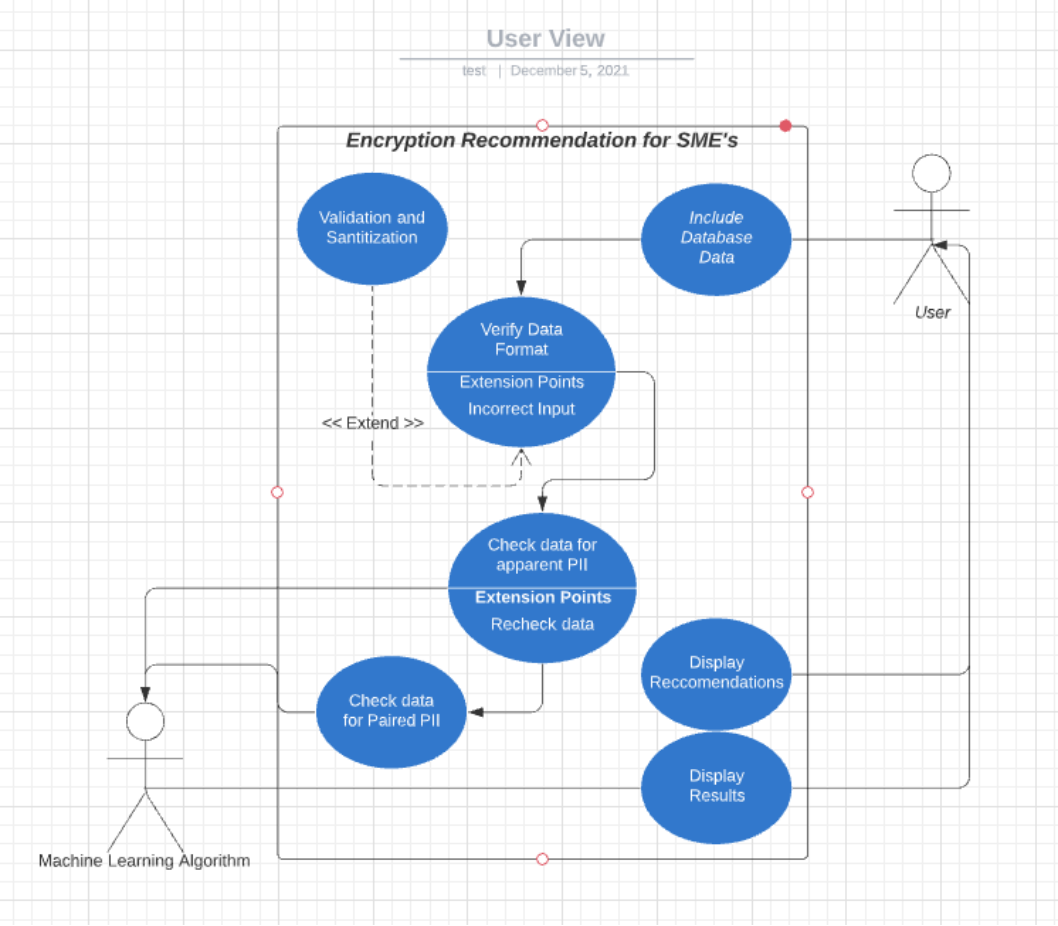
- The recommended fields should be stored in a generated file and should be downloadable by the user.
- A drag and drop functionality could be added to include user files.
- The output recommendation files could be encrypted when stored in the application environment so when an attacker gets ahold of those files no confidential data gets out.

User Groups

The user groups will include Small to Medium Businesses, meaning that there cannot be much technical terminology. Most of the people using the application will require simple and concise explanations to the information displayed and an easy and intuitive way to traverse and use the application.
The results may be given to a system administrator so they can be encrypted. This means that the results should be formatted correctly to ease the implementation of the required encryption.
An IT Sector Manager might want to view the data beforehand to make sure that the recommendations are of satisfactory quality. The recommendations should not include technical terminology and should be displayed in a neat manner to accelerate the approval process.
The chief of the Security Operations Center (SOC) might view the data and recommendations to see if all vulnerabilities are covered.

Diagrams

Use-Case Diagram



Use Case Descriptions

1. Include Database Data

Interaction:

User

Description:

- User enters the application
- Display box asks user to enter database information
- Application takes in database information

Results:

The application takes the user data in to be processed.

2. Verify Data Format

Interaction:

Application

Description:

- Data is received
- Check if format is as expected
- Attempt to modify format to make it usable

Results:

If data supplied is correct then it will be sent to the machine learning algorithm to be classified. If the data supplied is incorrect it will be sent to the validation and sanitization functions.

3. Validation and Sanitization

Interaction:

Application

Description:

- Unsafe/unformatted data received

- Check if data format is correct
- Validate and sanitize the input

Results:

The data is validated/sanitized and safe to use or display an error message alerting the user of incorrect input.

4. Check for Apparent PII

Interaction:

Machine Learning Algorithm

Description:

- Validated data received
- Show data to machine learning algorithm
- Machine Learning algorithm decides on which data is classified as PII

Results:

Apparent PII is found and saved to the output. Data is resubmitted for further investigation.

5. Check for Paired PII

Interaction:

Machine Learning Algorithm

Description:

- Data is resubmitted to the machine learning algorithm
- Machine learning algorithm compares different data clusters to each other to see if any PII is formed

Results:

The rest of paired PII is detected and sent to the output and recommendations section.

6. Display Recommendations

Interaction:

Application

Description:

- The function receives results from the machine learning algorithm
- System displays recommendations to the user

Results:

The application displays recommendations on how to deal with sensitive information, for example which encryption algorithms to use and how to implement them correctly.

7. Display Results

Interaction:

Application

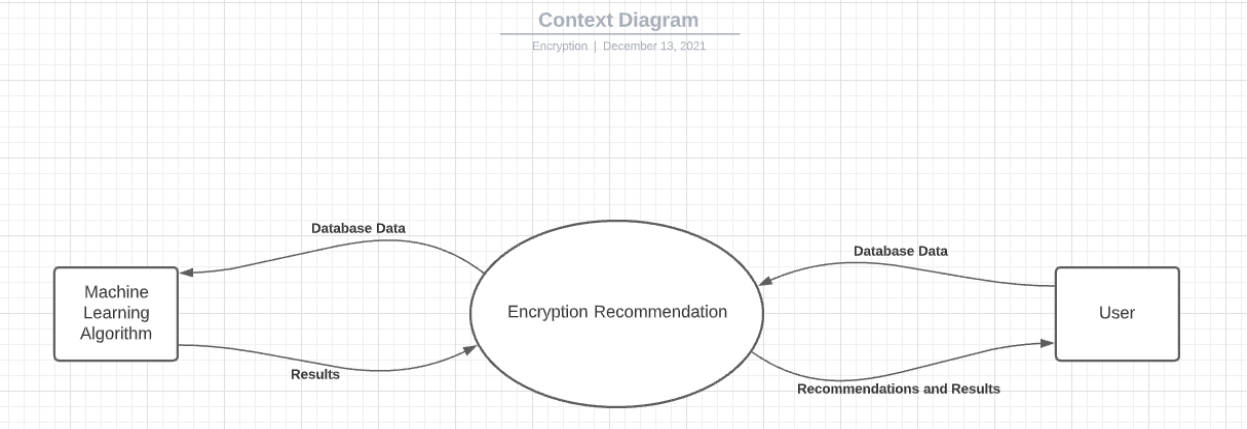
Description:

- The application receives the results from the machine learning algorithm
- The results are displayed to the user

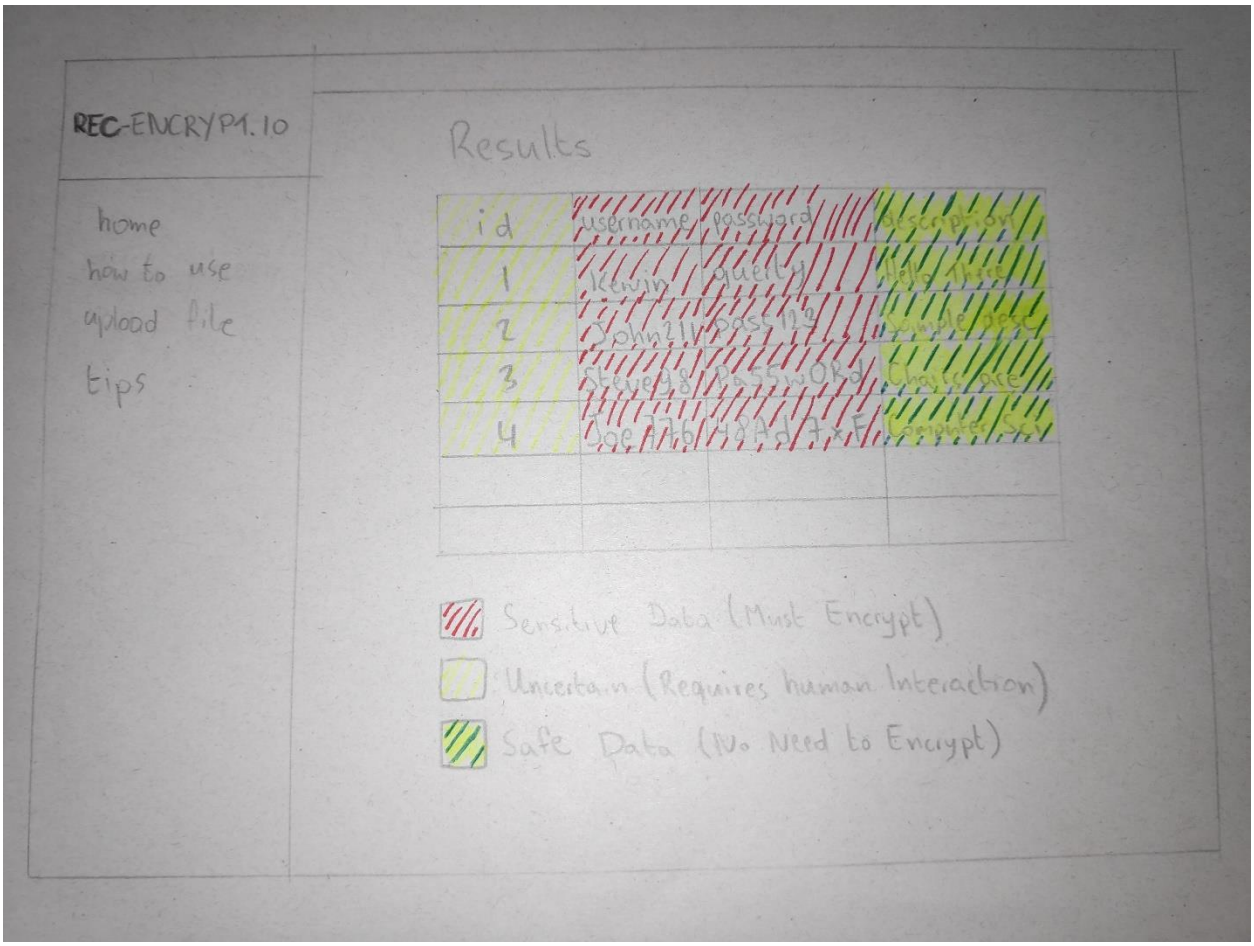
Results:

The application displays the sensitive information/PII that needs to be encrypted to the user.

Context Diagram



Sample Design



Metrics

Benchmark	Description
PII detection	The algorithm should contain a minimum amount of false negative answers where non-sensitive data is recommended for encryption.
PII detection	The algorithm should contain a minimum amount of false positive answers where sensitive data is not encrypted.
Feedback	The system should approximate the amount of time required to encrypt the data it recommends.
Reliability	The results should be 100% compliant with GDPR.
Security	The Machine Learning algorithm should not be accessible by unqualified personnel.
Security	Any results that will be stored on a file should be encrypted using a secure algorithm.
Usability	The recommendations and results are easy to understand.
Usability	Wherever the application requires any kind of input there should be a prompt to ask for such.
Usability	The application is intuitive by design.
Errors	The application handles errors appropriately by displaying a helpful message to the user. For example, Incorrect Input try again or Please provide a database.
Error Tolerance	Any error that comes up should not interfere with the user's experience.
Result Accuracy	The algorithm's recommendations should match the recommendations of a cybersecurity professional.

FURPS+

Functionality

- The application should allow end users to upload data to the system
- The application should detect sensitive information and output it to screen or file for the user to view
- The application should recommend best ways to secure the sensitive information detected through encryption and appropriate storage methods

Usability

- The user interface of the application should be self-explanatory to allow users of varied technical background to use it
- The application should be compatible with the windows operating system

Reliability

- The application should be designed so that when the data supplied is incorrect some validation methods will be deployed and in the worst-case scenario the end user will be given an error message
- The application should be secure against cyber attacks
- The machine learning algorithm should be trained to give reliable results, extra training data can be entered to further increase the accuracy of the system

Performance

- The machine learning approach is designed to make the application as efficient as possible
- The application should calculate the time complexity required to encrypt the supplied vulnerable data
- A time estimation should be displayed when the system is processing the supplied data

Supportability

- Testing of the application should be easy to do and performed with dummy data instead of actual data to protect user privacy
- The use of a machine learning algorithm allows the application to adapt to the data supplied meaning that it will make its own attempt to classify it

(+) Security

- The recommendations should include up to date techniques to secure the system
- If any application output is stored on the computer it should be encrypted before storage to prevent privacy breaches
- If a login functionality will be implemented the login information stored must be encrypted and passwords hashed inside a SQL database

Inspiration

While conducting my research I found out that all current solutions are too complex for the general user, with either too many features or the lack of a user-friendly GUI. In the past I've been on the receiving end of data breaches. Sensitive data of mine has been leaked online. This led to me being targeted by scam calls/texts and fishing campaigns. Data breaches similar to the one I was a victim of could be prevented in the future through the use of security automation tools, leaving less room for human error. I believe that companies should become more aware of their obligation to protect their users' data. If companies could automatically identify information that requires proper and secure storage it would empower them to make better decisions in the future in regard to data security.

Project Plan

Plan	Due Date
Research Manual	26th November 2021
Testing libraries to be used in project	9th December 2021
Presentation I	16th December 2021
Functional Specification	17th December 2021
Project Plan	17th December 2021
Research Project Design	20th December 2021
Research Python Stand Alone Applications (Tkinter)	22nd December 2021
Design Manual	13th January 2022
Presentation II	28th January 2022
Create Application Skeleton (Tkinter)	7th February 2022
Build function that allows JSON file input	12th February 2022
Make validation and formatting functions for user data	22nd February 2022
Familiarize with AWS syntax	2nd March 2022
Use AWS functions to interact with the service	12th March 2022
Investigate results given by the Macie algorithm and implement a results page	18th March 2022
Parse Macie results into something easier to work with	23rd March 2022

Create additional pattern-based detection for password fields and home addresses	28th March 2022
Create a page to display results and recommendations	31st March 2022
Test the application core functions	4th April 2022
Final error handling	8th April 2022
Implement login functionality	14th April 2022
Test application optional functions	18th April 2022
Final Report	21st April 2022
Final Product	25th April 2022
Project Website	25th April 2022

Tools Used

Software

- Python
- AWS console
    - Amazon Macie
    - Amazon s3
    - Amazon sts
    - Amazon IAM
- Urllib3
- Boto3
- Json
- Tkinter
- pycryptodome

Hardware

- Windows Machine
    - Running Services
    - Running Application

DECLARATION

*I declare that all material in this submission e.g. thesis/essay/project/assignment is entirely my/our own work except where duly acknowledged.

*I have cited the sources of all quotations, paraphrases, summaries of information, tables, diagrams or other material; including software and other electronic media in which intellectual property rights may reside.

*I have provided a complete bibliography of all works and sources used in the preparation of this submission.

*I understand that failure to comply with the Institute's regulations governing plagiarism constitutes a serious offence.

Student Name: (Printed) Kewin Skuza

Student Number(s): c00237361

Signature(s): 

Date: 25th April 2022