KING'S OWN INSTITUTE*

Success in Higher Education

ASSIGNMENT COVERSHEET

Family Name: Ahnaf Given Name: Shakil

Student Number: 20031759 Lecturer's Name: Sajad Ghatrehsamani sir

Subject Name: ICT711 Programming and Algorithms

Assignment Title: Algorithms and GUI

Declaration

(This declaration must be completed by the student or the assignment will not be marked.)
I certify the following:

- I have read and understood the *Student Academic Misconduct Policy*
- This assignment is my own work based on my personal study and or research.
- I have acknowledged all material and sources used in the preparation of this assignment including any material generated in the course of my employment.
- The assignment has not previously been submitted for assessment.
- I have not copied in part or in whole or otherwise plagiarised the work of other students.
- I have read and I understand the criteria used for assessment.
- The assignment is within the word and page limits specified in the unit outline.
- The use of any material in this assignment does not infringe the intellectual property / copyright of a third party.
- I understand that this assignment may undergo electronic detection for plagiarism, and an anonymous copy of the assignment may be retained on the database and used to make comparisons with other assignments in future.
- By completing this coversheet in full and submitting this assignment electronically, I am bound by the conditions of the KOI's *Student Academic Misconduct Policy* and the declaration on this coversheet.

Ahnaf Shakil
Signature

27 / 01/ 2026
Date

Assignment Receipt

Family Name: Ahnaf Given Name:(s) Shakil

Student Number: 20031759 Lecturer's Name: Sajad Ghatrehsamani sir

Subject Name: ICT711 Programming and Algorithms

Assignment Title: Algorithms and GUI

Ahnaf Shakil
Signature

27 / 01/ 2026
Date

Table of Contents

1. Project Overview

1.1 System Description:

Here I have choosen to make a text based and GUI based software “Cyber Security Incident Management System”. This is a Java based program that provides a means for creating, managing and storing reports of cyber security incidents. With this software, cyber security experts can log incidents reported by customers, monitor the level of vulnerability and the current status of these incidents and store information on a continuous basis through the use of files. The system also has two user friendly interfaces a text interface and a graphical interface to give users flexibility when selecting which interface they wish to use.

1.2 Objectives:

The overall objectives of this project is to:

1) Provide a simple to use cyber security incident reporting system that is built using the Java programming language,

2) Allow for all the CRUD (create, read, update, delete) operations associated with incidents to be completed by using this program,

3) Demonstrate the principles of object oriented programming,

4) Create a means for preserving data using file handling,

5) Enable users to interact with the system via the text interface or the graphical user interface.

1.3 Key Features:

1) Two interfaces (console and graphical),

2) Ability to add, edit, delete, search, and view incidents,

3) All incidents will have a unique ID,

4) Data for incidents will be saved permanently in a text file,

5) A modern GUI will be created based on aesthetic appeal and custom layouts and styles,

6) Classes for system functionality will be structured in a way that promotes reuse of code and modularity.

2. User Interface Implementation

2.1 Text Based Interface:

The text based interface is implemented through the ConsoleUI class. It provides a menu driven interaction where users can:

1) Add new incidents,

2) View all incidents,

3) Search incidents by ID,

4) Exit and save data (Together).

User input is handled using the Scanner class with validation to prevent invalid numeric entries. This interface is simple, lightweight, and suitable for quick testing or environments without GUI support.

2.2 Graphical User Interface (GUI):

The Graphical User Interface (GUI) implemented in the GUIFrame class makes use of the Java Swing technology. The GUI is designed with cyber security visual effects. The following features are included in the GUI with the java Swing technology:

1) A custom CircularLayout to navigate through the Circular Button Icons,

2) PopuUp Dialogs to Add and Edit Incidents and view a previous incident or Delete an incident,

3) Controlled user input by allowing users to indicate Severity and Status of incidents via Combo Boxes along with a Scrollable Text Box for a description of the Incident,

4) An improved overall user experience and

5) A more professional and interactive method of interacting with the System compared to the Console interaction.

3. System Architecture

3.1 Class Design:

The system is divided into modules defining the different roles within the system,

Main: Represents the start of the application where the user selects the UI type,

Incident: Data Model for the Incident,

IncidentManager: Handles data logic and stores incident related data,

ConsoleUI: Handles text based user interaction,

GUIFrame: handles graphical user interaction,

CircularLayout: A layout manager to implement circular button positioning. By dividing the modules in this manner it makes it easier to update and enhance each module.

3.2 Design Patterns Used:

Model View Controller (MVC) (Partial)

Model: Incident

Controller: IncidentManager

View: ConsoleUI and GUIFrame

Single Responsibility Principle: Here each class performs a specific role.

Encapsulation: Basically it's private fields with public getters and setters.

4. Sorting and Searching Implementation

4.1 Searching Algorithm:

To locate Incidents by their ID, we utilize a linear search algorithm via the searchById() method inside the IncidentManager.

The algorithm justifies that:

1) The current dataset size is relatively small,

2) The ease of reading and understanding takes priority,

3) The linear search is sufficient for academic purposes and small applications.

4.2 Sorting:

The current implementation will not allow the user to define an explicit sort order for the display of incidents, however, the order in which they were added will continue to be utilized as the display order (last in the first out). This design allows the program to remain simplistic while satisfying the requirement of the assignment.

5. Algorithm Discussion & Complexity

Operation	Algorithm	Time Complexity
Search by ID	Linear Search	O(n)
Add Incident	Append to ArrayList	O(1)
Delete Incident	RemoveIf (Linear Scan)	O(n)
Edit Incident	Search + Update	O(n)
Load from File	Sequential Read	O(n)
Save to File	Sequential Write	O(n)

The algorithms selected for this project are efficient for small to moderate size data sets and also fit well within the scope of the project.

6. Testing and Debugging

6.1 Test Cases:

Test Case	Expected Result	Actual Result
Add new incident	Incident saved successfully	Passed
Duplicate ID	Warning message shown	Passed
Search existing ID	Correct record displayed	Passed
Search invalid ID	Not found message	Passed
Save to file	Data written to text file	Passed

Load from file	Incidents loaded at startup	Passed

6.2 Debugging:

Common problems that were discovered while developing the project:

1) Errors encountered from improper specification of file paths that affect the ability to persist data,

2) Problems with aligning the layout of the GUI,

3) Problems validating user input to ensure the data entered into numeric fields is indeed numeric,

4) Problems compiling files that utilize the CircularLayout class,

5) Problems were resolved through systematic debugging via console logging and exception management.

7. Reflection

7.1 Challenges Faced:

1) Creating an attractive, user friendly graphical user interface using the Swing package,

2) Develop persistence of your data through the correct file path and storage mechanism selections,

3) Validate all user input from both interfaces,

4) Use the user defined layout manager you created for this project,

5) As my VS Code system changes daily so I had to made strong coding decision.

7.2 Problem Solving Approach:

The identified obstacles were solved using the following methods:

1) Breaking the program up into smaller modules.

2) Testing the program features in a stepwise manner.

3) Following the principles of object-oriented design to maintain a good, clean program.

4) Utilizing a stepwise method of debugging rather than modifying both components simultaneously.

7.3 Lessons I Learned:

1) The importance of splitting up tasks and responsibilities,

2) The benefits of combining the Graphical User Interface (GUI) with the Command Line (CLI),

3) The effective use of file handling in Java,

4) More detailed knowledge of swing component creation and the layout of those components,

5) Better methods of debugging and testing code,

6) Day by day coding skills.

8. Conclusion

Overall, this project has shown the creation of an Incident Management System for Cyber Security with Java using object oriented programming principles as well as object file handling through both GUI and CLI applications. The system has satisfied all functional specifications and will allow for future improvements such as sorting, database connections and implementing secure user authentication.