The Corporate Vulnerability Management System

Team members:

- Selin Deniz - 202011051
- Pelin Koz - 202011048
- Meleksu Özdoğan - 202011054
- Sena Bitirgen - 202011029

A.) Introduction of Program

Program Language: C++

Main Program's Name: Corporate Vulnerability Database Management System

Compiler: Visual Studio, Online gdb

B.)Design of Program

Purpose of The Program and Solution Design:

The Corporate Vulnerability Database System is a program designed to manage and provide information about previous vulnerabilities and attacks in a company, as well as information about new publicly released attacks. It can be accessed only by authorized personnel and requires users to authenticate themselves using a username and password. The program allows users to choose between three options: reading data about a specific vulnerability or attack, writing new data to the system, or exiting the program.

The program makes use of several approaches to achieve its goals, including

- **Polymorphism: The program uses polymorphism to allow different behavior depending on the type of user. For example, the** `write()` **function is implemented as a virtual function in the** `User` **class and is overridden in derived classes to provide different behavior based on the user's role.**

- **Overriding: As mentioned above, the** `write()` **function is implemented as a virtual function in the** `User` **class, and is overridden in derived classes to provide different behavior based on the user's role. This allows the program to provide different functionality depending on the type of user, without having to use conditional statements or other control structures.**

- **Overloading: The** `InvalidInputException` **class used in the program has an overloaded constructor that allows it to be initialized with either a string message or no message at all. This allows the user to customize the error message displayed in the event of invalid input, or to use a default error message if no message is provided.**

- **Abstraction: The program makes use of abstraction to hide the details of how certain tasks are performed, and instead provide a simplified interface for users to interact with. For example, the** `read()` **and** `write()` **functions provide a simple interface for retrieving and storing data, without exposing the details of how this data is stored or accessed.**

- **Encapsulation: The program uses encapsulation to protect the data stored in the** `User` **class from being accessed or modified by unauthorized users (not the class). This is achieved by declaring the data members of the** `User` **class as private and providing public accessor and mutator functions to allow authorized users to read and modify this data. This helps to ensure the integrity and security of the data stored in the system.**

- **Exception handling: The program makes use of exception handling to handle invalid input and other error conditions. This is implemented using the `try-throw statements and InvalidInputException class.**

In addition to the approaches mentioned above, the Corporate Vulnerability Database System also makes use of the following techniques:

- **File input/output: The program reads from and writes to files in order to store and retrieve data about vulnerabilities and attacks, verify the username, passwords, and mail addresses.**

- **Data validation: The program includes a number of checks to ensure that input data is in the correct format and meets other requirements. For example, the** `obtainCVE()` **function checks that the input string meets the required format for a CVE number, and the** `checkUser()` **function checks that the input username and password match those stored in the system.**

- **Data structures: The program uses a number of data structures to store and manipulate data.For example, the** `vector` **container is used to store information about publicly released attacks for reading purposes in the read() function.**

- **Regular expressions: The program makes use of regular expressions to validate the format of input data, such as the format of the mail address. This is implemented using the** `regex` **library for the 2FA process.**

- **Static variables: The program uses static variables to store data that is shared across all instances of a class. For example, the User class includes static variables for the CVE number and the mail of the user for the two-factor authentication process. These variables are shared across all instances of the User class, allowing them to be accessed and modified by any instance.**

Overall, the program uses a combination of these approaches to provide a functional and secure database system for managing and providing information about vulnerabilities and attacks.

Design for Programming:

- Pseudocode:

1. Obtain a password and username from the user for authentication

1.1 If there is no match ask the user until three attempts are made
- Exit if there is still no match

2. Obtain the preference of the user (read/write)

2.1 If the preference is READING from the database
- Obtain the related CVE Number
- Read related data
- Ask the user to write, exit, or read again
  - If the user chooses to exit, direct the user-related website in the case of approval to learn more about the newly released attack

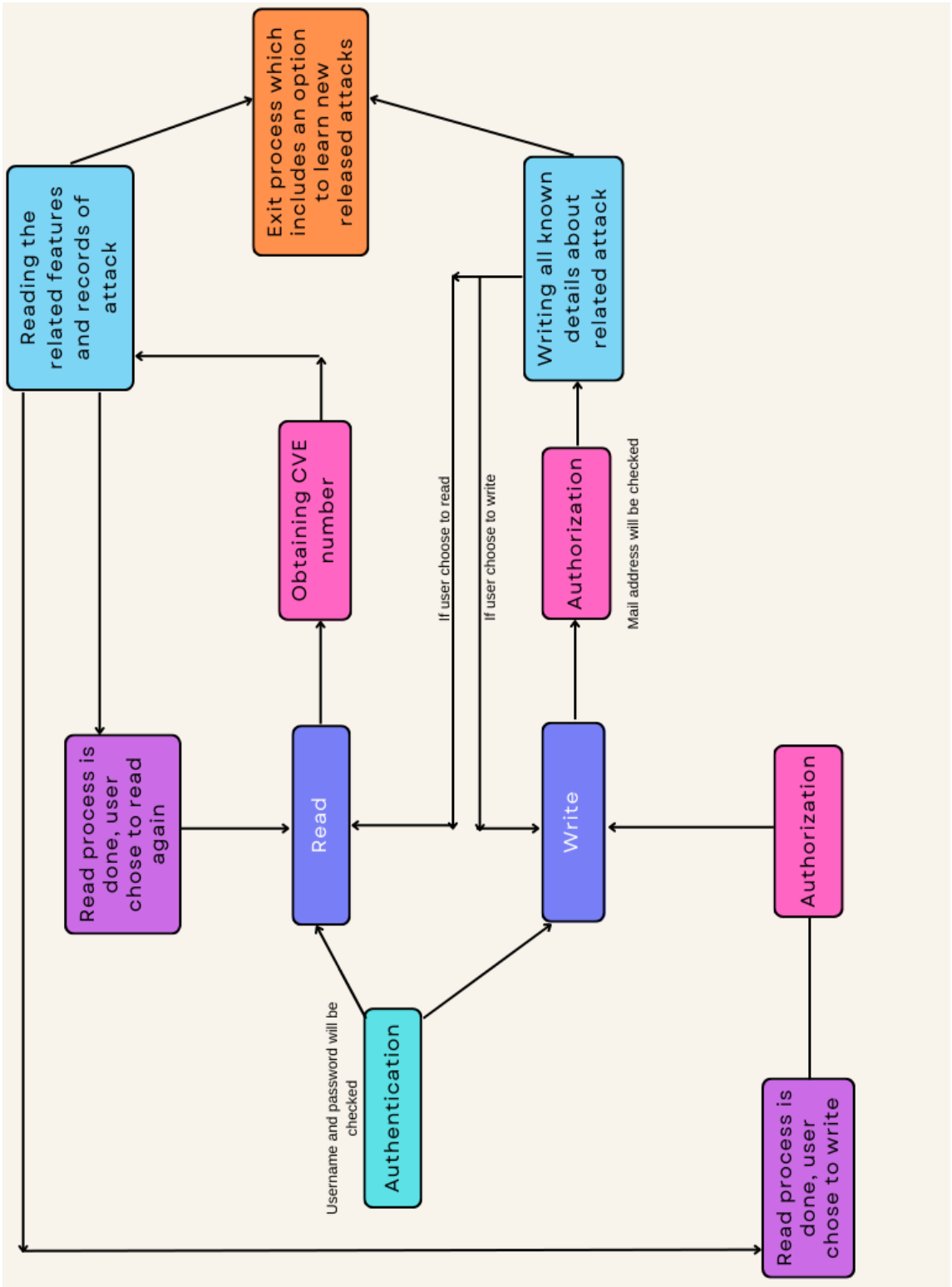
2.2 If the preference is WRITING on the database

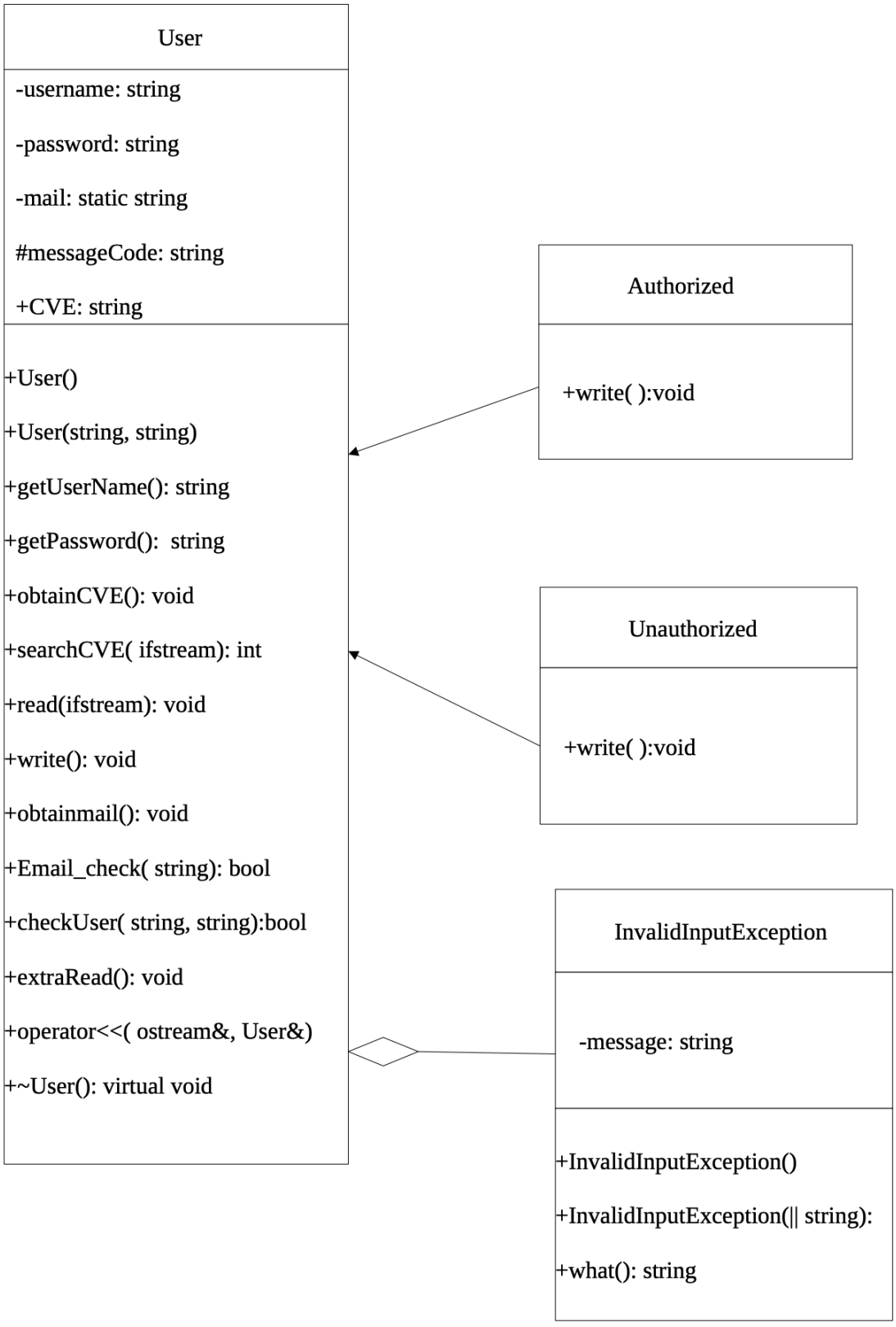
- Apply the appropriate method for 2FA (two-factor-authentication)

2.2.1 If the user has no access to write on the system
- Print the message “permission is denied”
- Ask the user to read or exit
  - If the user chooses to exit, direct the user-related website in the case of approval to learn more about the newly released attack

2.2.2 Otherwise
- User writes related data about the new entry
- Ask the user to read, exit, or write again
- If the user chooses to exit, direct the user-related website in the case of approval to learn more about the newly released attack.

- Algorithm Chart and UML diagram:




User
-username: string
-password: string
-mail: static string
#messageCode: string
+CVE: string
+User()
+User(string, string)
+getUserName(): string
+getPassword(): string
+obtainCVE(): void
+searchCVE(ifstream): int
+read(ifstream): void
+write(): void
+obtainmail(): void
+Email_check(string): bool
+checkUser(string, string):bool
+extraRead(): void
+operator<<(ostream&, User&)
+~User(): virtual void
Authorized
+write( ):void
Unauthorized
+write( ):void
InvalidInputException
-message: string
+InvalidInputException()
+InvalidInputException(|| string):
+what(): string

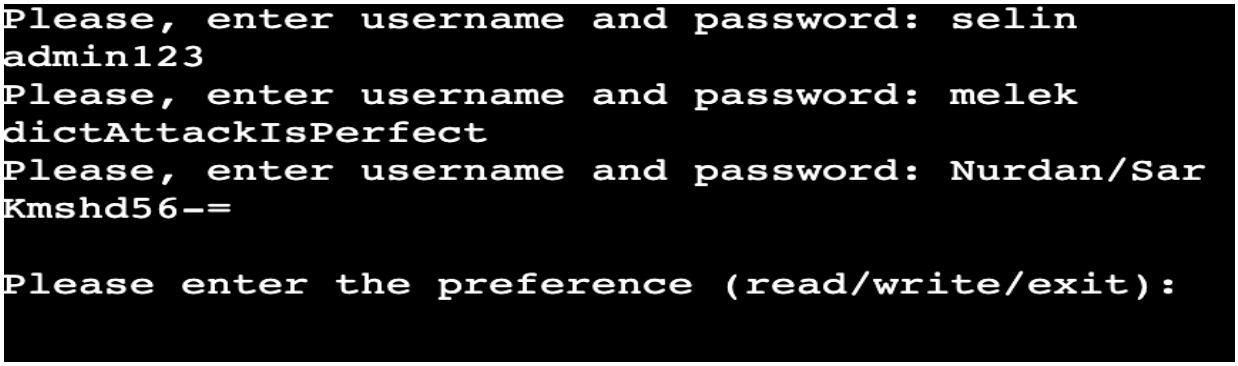
- Testing Results:

—----------LOG IN—----------

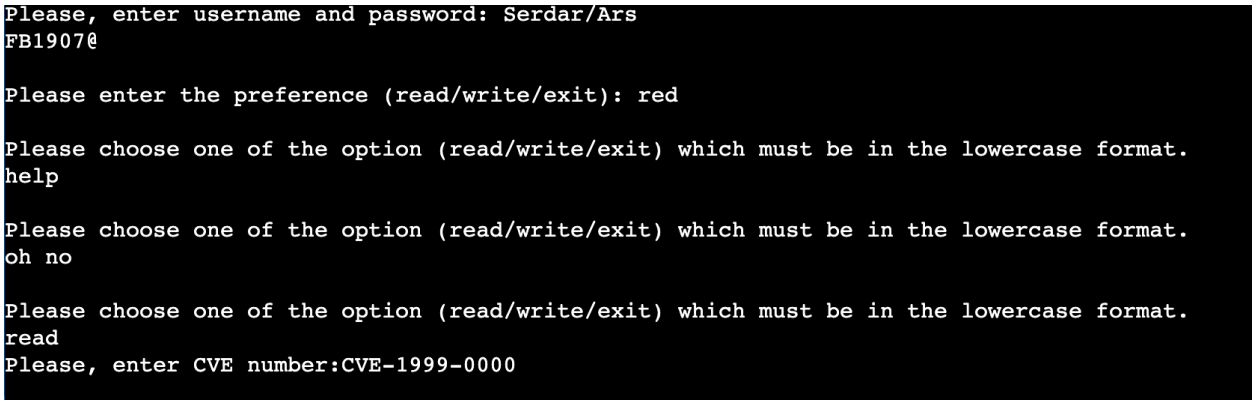
```
Please, enter username and password: Selin
admin123
Please, enter username and password: Melek
password123
Please, enter username and password: Sena
mypassword


...Program finished with exit code 0
Press ENTER to exit console.
```

```
Please, enter username and password: Serdar/Ars
H*9zXyzm-

Please enter the preference (read/write/exit):
```

```
Please, enter username and password: selin
mypassword
Please, enter username and password: AhmetT/Cos
Nm8*9_-Kl

Please enter the preference (read/write/exit):
```



On the login page, the user has 3 attempts to write an invalid username or password, in the case of a successful login process preference is obtained, otherwise, the program is terminated.



After the successful login process, the user has to enter 3 options available which are sensitive upper-lowercase (read/write/exit). Preference is obtained until the user obeys the rules of the expected format.

——————————————————READ——————————————————

```
Please, enter username and password: Serdar/Ars
FB1907@

Please enter the preference (read/write/exit): read
Please, enter CVE number:cVe-1000-1000

Expected format of CVE number didn't provided, please try again.
The string must start with 'CVE' prefix.

CVE_abcd-abcd

Expected format of CVE number didn't provided, please try again.
The string must include '-' character after 'CVE' prefix.

CVE-ab-1234

Expected format of CVE number didn't provided, please try again.
The string must include 4 digits after the first '-' character.

CVE-1999-0001
```

In the case of the read option being chosen, obtained CVE number must provide the expected format requirements. The process is repeated until requirements are provided.

```
Please, enter username and password: Ser/
Ars/

Please enter the preference (read/write/exit): read
Please, enter CVE number:CVE-1907-1907
Status:Hopefully to finals
Known affected configurations:Galatasaray
Known name:
"lefter çıktı sahaya
topu dikti havaya
bunu gören cimbomlar
başladı ağlamaya"

Please enter the preference (read/write/exit): read
Please, enter CVE number:CVE-2098-0009
Given CVE number couldn't found in the system.
```

In the case of a valid CVE number entry, related pieces of information are shown to the user. The preference-asking process is repeated until the user decides to exit.

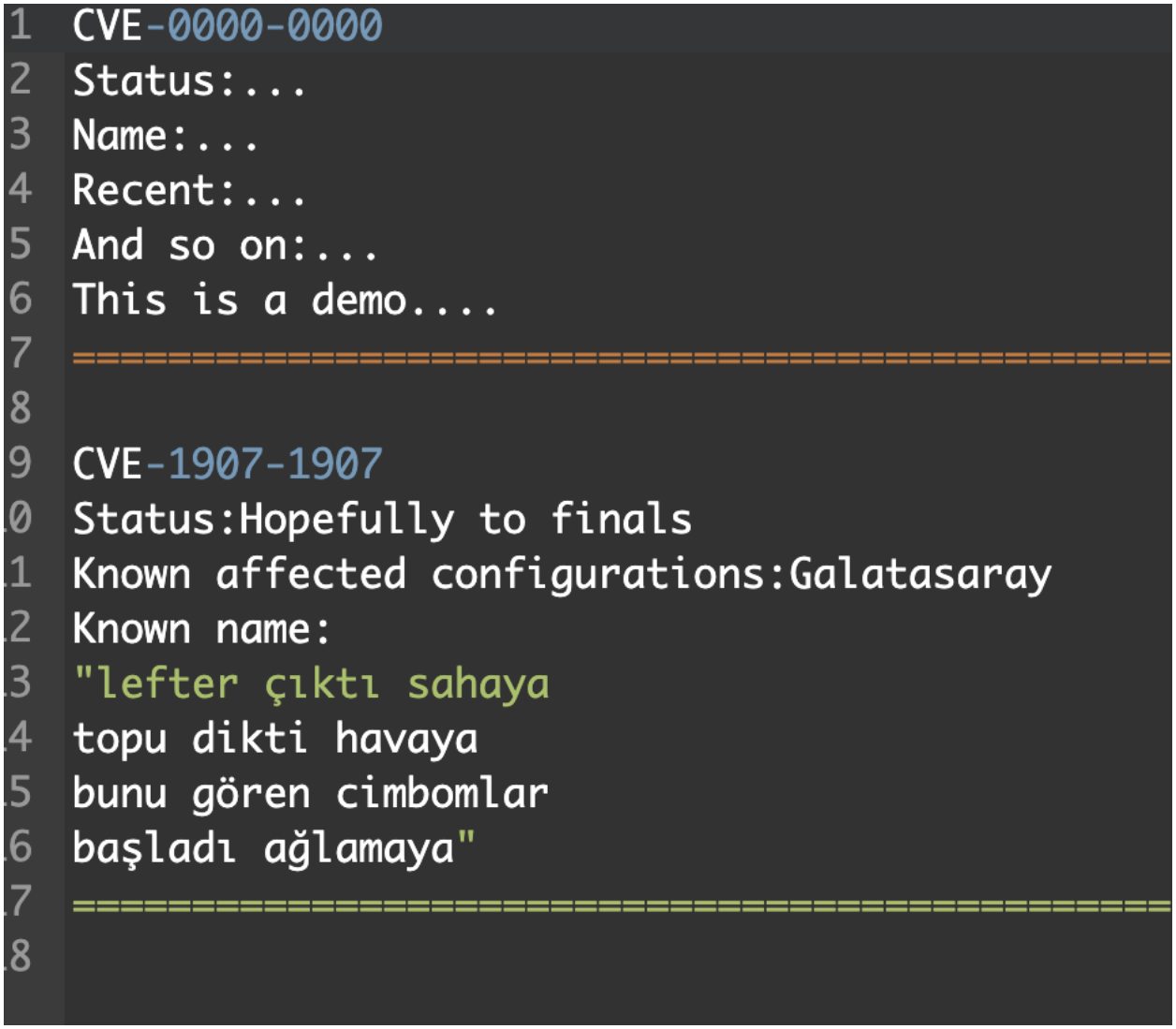
——----------WRITE——----------

```
1 CVE-0000-0000
2 Status:...
3 Name:...
4 Recent:...
5 And so on:...
6 This is a demo....
7 ==========
8
```

File before the writing process.

```
Please enter the preference (read/write/exit): write
Please, enter your Email-Id:
sarslan@cankaya.edu.tr
User  with mail address sarslan@cankaya.edu.tr is entering the system
When you are finished typing, please press enter and type done.
Please enter the records of only one at a time with respect to CVE number
CVE-1907-1907
Status:Hopefully to finals
Known affected configurations:Galatasaray
Known name:
"lefter çıktı sahaya
topu dikti havaya
bunu gören cimbomlar
başladı ağlamaya"
Done

This program succesfully stored in the file.
Please enter the preference (read/write/exit): exit

Would you like to be informed about new released attacks?no

You chose to exit, program is terminated.
```

In the event of an Authorized user entering the system to write.



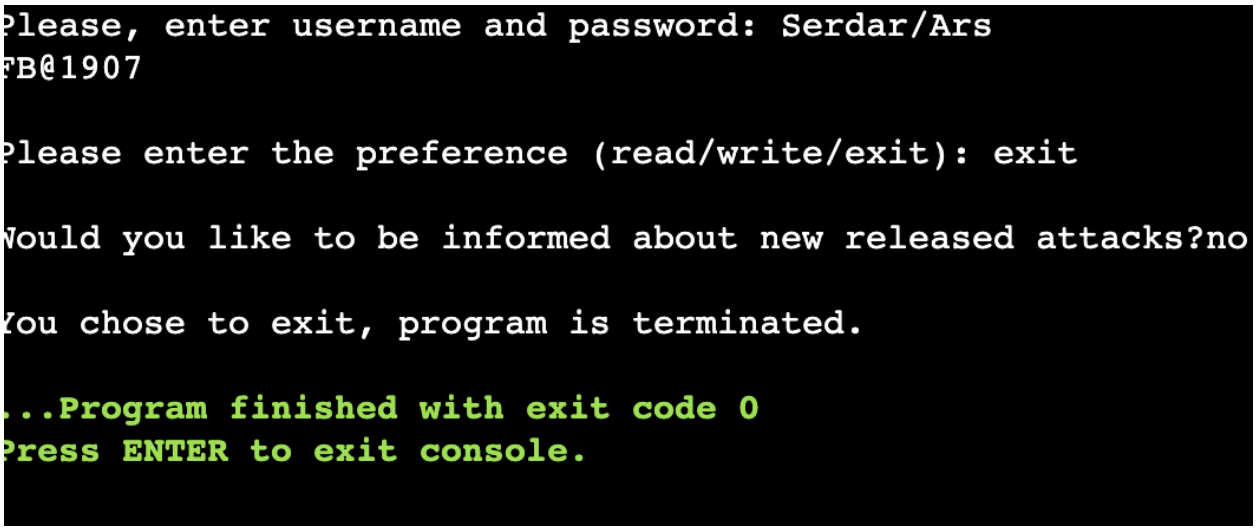
File after the writing process.

```
Please, enter username and password: Ser/
Ars/

Please enter the preference (read/write/exit): write
Please, enter your Email-Id:
CimBomBom@backTohome

Invalid mail address is inputted, please try again.

Cimbobom@whereIshome

Invalid mail address is inputted, please try again.

Cimbombom@home..

Permission denied for Unauthorized writer!!

Please enter the preference (read/write/exit): write

Permission denied for Unauthorized writer!!

Please enter the preference (read/write/exit):
```

In the case of Unauthorized user entry which is not verified by the 2FA method and used all rights to enter (3), aborted in the system when trying to write on the database.

——————————————————EXIT——————————————————

```
Please, enter username and password: Serdar/Ars
FB@1907

Please enter the preference (read/write/exit): exit

Would you like to be informed about new released attacks?yes
You chose to read beyond about new released attacks and methods.
Please visit the website via provided URL: https://attack.mitre.org/resources/updates/

You chose to exit, program is terminated.

...Program finished with exit code 0
Press ENTER to exit console.
```



In the case of the exit option being chosen, preference about reading beyond is obtained from the user, and in a positive case user directed related website, otherwise, the program is terminated.

C.)Conclusion

Shortcomings:

One potential issue is that the program only allows the user 3 attempts to enter the correct username and password for authentication. While this is a security measure to prevent unauthorized access to the system, it may also make it more difficult for legitimate users to access the system if they forget their login credentials.

Another potential issue is that the program does not offer any mechanism for recovering forgotten login credentials. This means that if a user forgets their username or password, they will not be able to access the system until they are able to reset their login information through some other means.

Finally, the program does not currently have any mechanism for tracking or managing user activity within the system. This means that it is not possible to see which users are accessing which records or what changes they are making to the data within the system. This could make it more difficult to identify any potential security issues or unauthorized access to the system.

How much it serves to purpose:

Overall, the Corporate Vulnerability Database System appears to effectively serve its purpose of managing and providing detailed information about previous vulnerabilities and attacks within the company. The program implements a number of approaches to ensure the security and integrity of the data, such as encapsulation and polymorphism, and also provides a user-friendly interface for authorized personnel to access and modify this information.

Result/Comment/Suggestions

In conclusion, the Corporate Vulnerability Database System successfully serves its purpose of providing authorized personnel with detailed information about previous vulnerabilities and attacks, as well as offering the option to read about new publicly released attacks. However, there are some areas that could be improved in future iterations of the program. For example, the program currently only allows for 3 attempts at entering a correct username and password before terminating. It may be beneficial to implement additional security measures, such as a lockout system or the option to reset a forgotten password, to further protect the system and the data it stores. Additionally, the program currently does not have a system in place for tracking user activity or monitoring for any suspicious behavior. Implementing such a feature could help to further secure the system and ensure that it is being used appropriately. Overall, the Corporate Vulnerability Database System is a valuable tool for managing and safeguarding the company's data, but there is always room for improvement to enhance its functionality and security.