Applied Information Technology And Computer Science Vol. 4 No. 1 (2023) 124-138

AITCS

Homepage: http://publisher.uthm.edu.my/periodicals/index.php/aitcs
e-ISSN :2773-5141

E-EvidenceNotes: An Implementation of Online Digital Evidence Taking Notes Tool with Data Retention

Nurul Hidayah Hassan[1], **Nurul Hidayah Ab Rahman**[1]*

[1]Faculty of Computer Science and Information Technology,
Universiti Tun Hussein Onn Malaysia (UTHM), Parit Raja, 86400, MALAYSIA

*Corresponding Author Designation

DOI: https://doi.org/10.30880/aitcs.2023.04.01.008
Received 18 September 2022; Accepted 27 May 2023; Available online 30 June 2023

Abstract: There are many techniques of evidence note-taking in an investigation. However, a few issues arise in the handwriting method for digital evidence note-taking such as the alteration made by outsiders, no digital attachments and the risk of information being stolen. Therefore, digital evidence taking notes tool with data retention policy is developed to address these issues. Waterfall Model is used as the methodology to develop e-EvidenceNotes consisting of five phases and the development of the project's prototype is using PHP, MySQL and Visual Studio Code. This tool has simplified the investigators manage their case notes which have the log in module, notebook and team notebook module, activity logs as well as data retention definition.

Keywords: Digital Evidence, Taking Notes Tool, Data Retention Policy

1. Introduction

Taking notes is the best practice to document any procedure when investigating and generating a thorough report. The tools required for the manual technique of evidence note-taking in an investigation are reachable by merely using a pad of paper and a pen. To ensure that the investigator's notes can be presented to the courts for scrutiny, it is important that the notes remain unaltered and that no notes are missing by not removing any pages from notes, leaving blank spaces, overwriting, or writing in margins.

Nevertheless, the traditional pen-and-paper method of taking notes is ineffective for gathering evidence found in digital files from electronic sources. The issues that arise are the handwriting method for digital evidence note-taking is easy can be altered by outsiders as it does not provide a timestamp for every alteration, making it hard for investigators to track it down. Next, it has no digital attachments, which the digital evidence can be any sort of digital file from electronic sources and this issue will make it harder for people to look up full evidence simply from the notes. The risks of lost or stolen information from an excessive backlog of paper files [1] might arise particularly in complex investigations that

*Corresponding author: hidayahar@uthm.edu.my

require the use of more paper when taking evidence notes which then make it difficult for them to manage the numerous files that have no duration of storage. The objective of this e-EvidenceNotes is:

i. To design web-based digital evidence taking notes tool with data retention approach.
ii. To develop web-based digital evidence taking notes tool with a data retention approach.
iii. To test and evaluate the developed system from system functionalities and users testing.

The scope for this system will be divided into two, which are the user scope and the system scope. As for user scope, the system enables users to log into the system, manage the case notebooks whether it is personal or with team and securely edit the case notes together. There will be end-users that can access the system and log in to the system by their own account. For the system's scope, there will be modules to guide this project toward its objectives which are log in, registration, my notebook, team notebook, case notes, data retention definition, activity logs and manage user account. At the end of this project, e-EvidenceNotes tool will be able to implement digital evidence taking notes tool concept by computerizing the manual system which can simplify the users to manage the investigation notes. In addition to that, the system will organize and save the data storage as well as ensure the security of stored information by integrating the data retention policy into the system.

2. Related Work

2.1 Digital Forensics Phase

Digital forensics has existed for as long as computers have stored data that could be used as evidence [2]. It is a procedure to analyze digital objects and develop and test theories that may be used in court as evidence in investigations and legal proceedings. The objective of the digital forensics phases is to recover, analyze, and preserve computer and associated documents so that the investigating agency may submit them as evidence in a court of law. Other than that, it is to generate a computer forensic report that provides a complete report on the investigation process as well as to preserve the evidence by following the chain of custody. To be accepted in a court of law, digital evidence must be handled in a very precise way that eliminates the possibility of cybercriminals tampering with the evidence. Therefore, digital forensics consists of the process of identification, preservation, analysis, documentation of computer evidence and presentation that can be used in court.

2.2 Digital Forensics Reporting Phase

The digital forensics reporting phase is the phase where a report will be generated and display the digital evidence discovered as well as the techniques, allowing anybody who reads the reports to see how conclusions arrived. Typically, reports must be tailored to whoever will eventually read them, whether it is an attorney, a client, or an investigator. There are two components in this reporting phase which are contemporaneous notes and Scientific Working Group on Digital Evidence (SWGDE) Requirements.

Creating contemporaneous notes needs to consider the form and integrity. The minimum requirements for form and integrity are that digital contemporaneous notes be treated as a main note-type wherever available [3]. Following that, integrity checks must be presented, and timestamp information must be validated as accurate, and timestamps for the beginning, end, and each note entry must be included. For SWDGE requirements, proper documentation is vital in providing investigators with the ability to recreate the forensic process and results as digital evidence, since they may be questioned in a court of law or other official procedures [4].

2.3 Data Retention Policy

A data retention policy is a documented process for storing records in a specific period for compliance or regulatory purposes. It is the fundamental guideline for managing the data and it helps to identify the purpose of the data, regulations that must be fulfilled, length of time that data should be kept and how

it should be archived or deleted when the time comes. The objective of the policy is to guarantee that essential records and files are protected and kept while the records that have no value or are no longer needed by an organization are disposed of at the appropriate time. Many risks are minimized by retention policies, including lost or stolen information, an excessive backlog of paper files, a waste of time and space when maintaining records internally [1]. Organizations commonly develop their own data retention policies while ensuring that those policies comply with or exceed all relevant data retention requirements [5].

2.4 Comparison with Existing System

In this study, the existing systems of digital evidence taking notes tools are being compared to learn more about the relevant aspects and to examine the security mechanism that should be enhanced and implemented.

2.4.1 CaseNotes

CaseNotes [6] is a single lightweight application software that runs on the Microsoft Windows platform and allows incident responders, forensic analysts, and examiners of all disciplines to securely record their contemporaneous notes electronically. It does not have a login and register section which makes this system is for the personal use of taking notes tool for the investigator. The configuration information of CaseNotes system is stored in a user-editable text-based.xml file, and it allows the users to run many copies of CaseNotes at the same time. However, as CaseNotes is not a multi-user application, each analyst should keep their own notes in their Case File, even if they are working on the same investigation case. It employs a secure 'write-once, read-many' method of capturing and storing case note data, as well as a comprehensive audit trail of a case note data entry and metadata modifications in a self-contained log.

2.4.2 Monolith Forensics

Monolith Notes [7] is a free desktop application that allows users to create forensic case notes. The case note's function does allow for picture formatting and pasting, and the notes are filterable, sortable, and associated with a specific case. Monolith Forensics does not have a login or registration section; therefore, this system is just for the investigator's private use as a note-taking tool only. It offers a user-friendly interface that allows users to quickly locate the system's capabilities. This free application has several features, including the ability to add a case, filter cases by case lead, status, type, and open date. Each note can also be edited or deleted, and the note can be added as much as the users wish. Users can export the notes to a Microsoft Word document which then can be edited, printed, or converted to a PDF document.

The results of the comparison between the existing systems and the proposed system are important in determining the similarities and differences, which have been summarized in Table 1.

Table 1: Comparative Summary of Existing System and Proposed System

Features / System	CaseNotes	Monolith Forensics	e-EvidenceNotes
Log In	X	X	√
Generate Report	X	√	√
Data Retention Policy Applied	X	X	√
Media attachments (images, video, audio)	X	√	√

3. Methodology/Framework

The Waterfall Model is a sequential software development methodology in which progress flows gradually, much like a waterfall, through all project phases [8]. Each of the phases in the waterfall model is reliant on the deliverables of the preceding one and corresponds to task specialization. Since the design used to develop the system, e-EvidenceNotes is based on a structured approach using the PHP programming language, the Waterfall model is relevant to be applied in this project.

3.1 Requirements Phase

The requirements phase is where all potential requirements are identified and analyzed for the proposed system, e-EvidenceNotes. The first task of this phase is identifying the system and the scope of the proposed system which the digital evidence taking notes tool is intended for the investigator to take notes during any investigation especially in cybercrime. The information regarding the data retention policy model was identified by conducting research and analysis from trusted online sources and journal papers that are related to the tool that is going to be developed. The information about evidence taking notes was also collected from the same sources.

A comparison of the existing tools has been made to study the common methods used in developing digital evidence taking notes tool and to understand the requirements for each of the systems. There were two existing systems that were selected to observe and analyze the functions, features and methods used and all the information collected from the comparisons is applied to the proposed digital evidence taking notes tool. From the tasks that have been done, the functional requirements, non-functional requirements as well as user requirements were coming out with the discussion that has been held with the supervisor.

3.2 Design Phase

The purpose of this phase is to study the requirement specifications from the previous phase and prepare the system design. In this phase, system design helps in designing the algorithm of the e-EvidenceNotes system. The tool that has been used is Lucidchart application to produce the system's flowchart as to design the flow of the system from the process of users register and log into the system, manage case notes inside the personal notebooks and team notebooks, assign people to the team, check the activity logs and the output of the system which is the report from case notes.

The wireframe for the proposed system's interfaces which include the login page, registration page, list of notebooks page, case notes, activity log and the generated report of the notes are being designed during this phase using Microsoft Word. The user interface design principles of the interface were inspired by the existing systems that were analyzed during requirement phase which met the user requirements.

The diagrams to present the database design of the system were also done in this phase. Figure 1 shows the context diagram, Figure 2 shows the Data Flow Diagram (DFD) and Figure 3 depicts the ERD of the system e-EvidenceNotes.

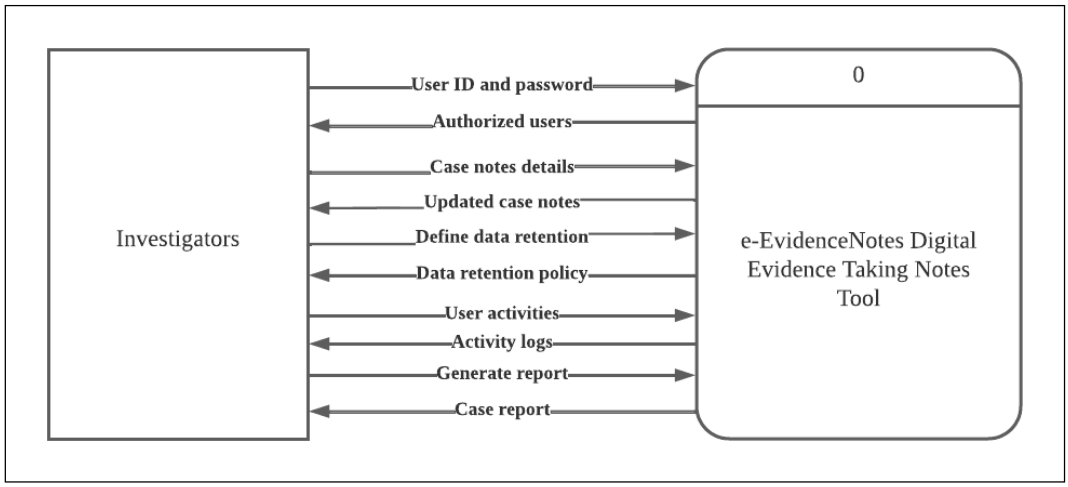

Figure 1: Context Diagram of e-EvidenceNotes

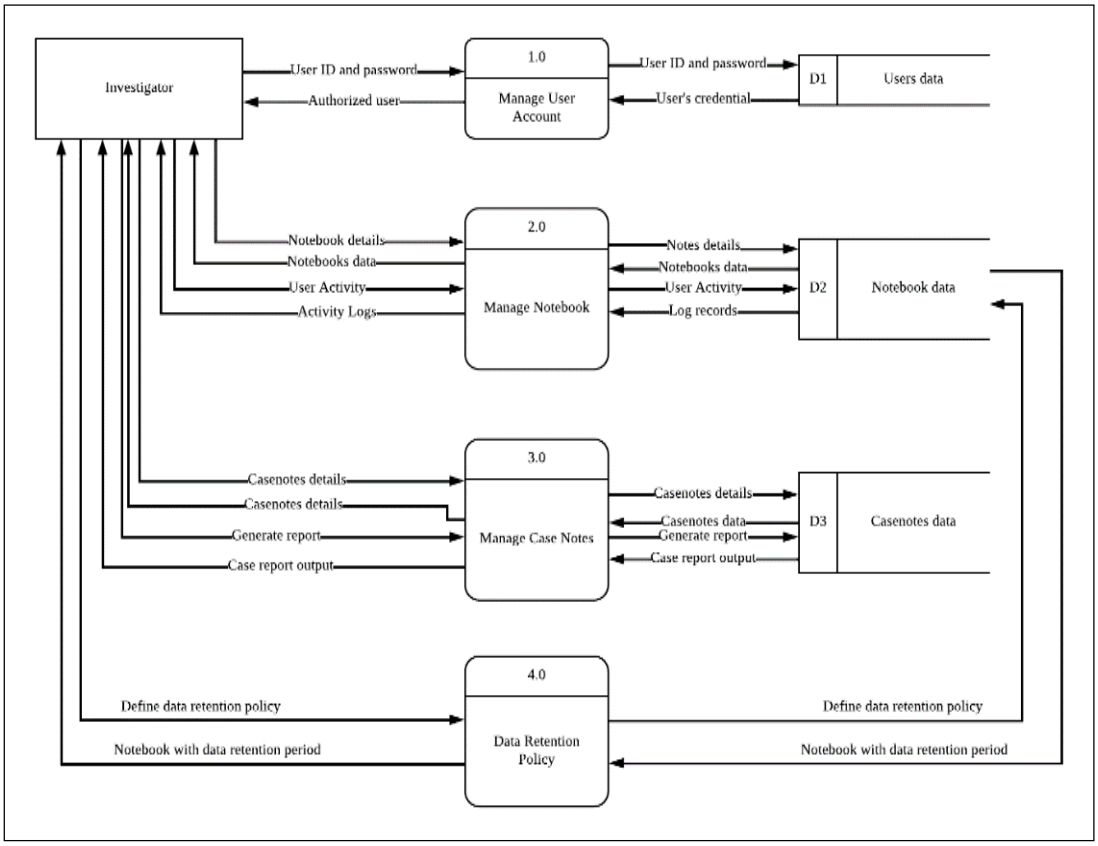

Figure 2: DFD of e-EvidenceNotes

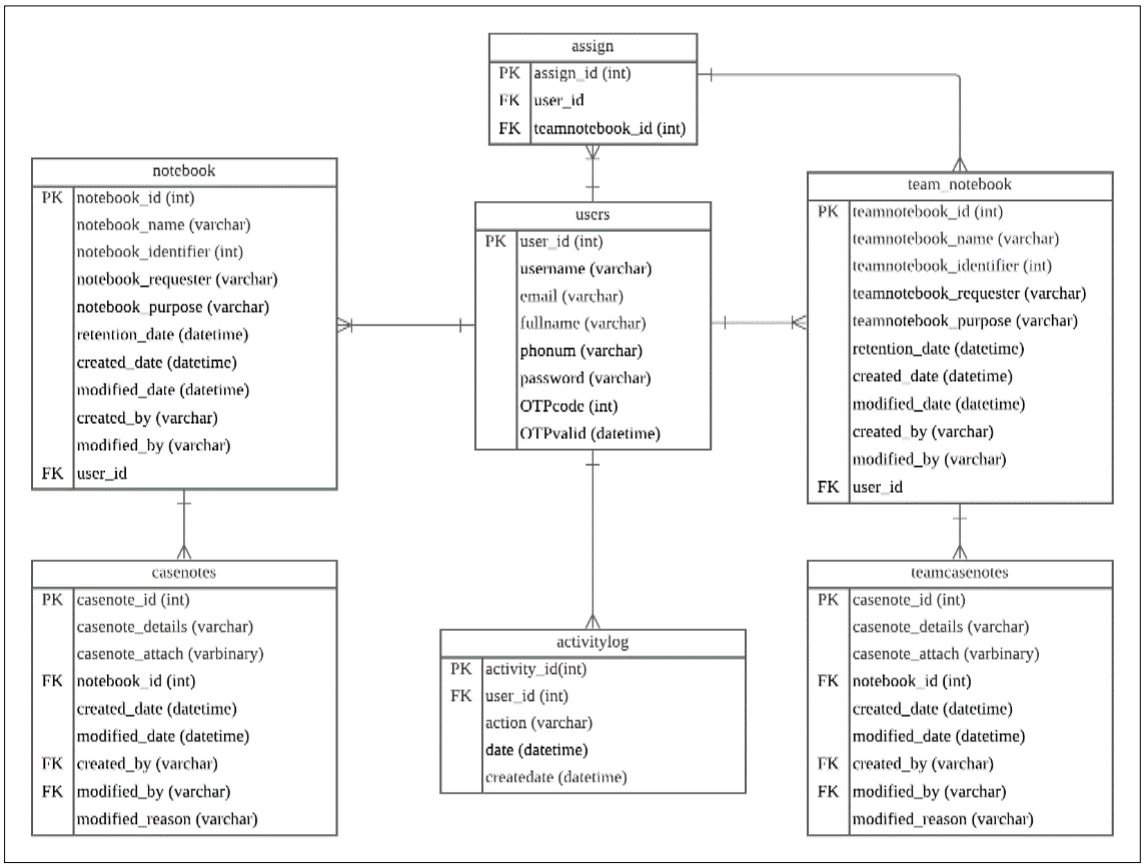

Figure 3: ERD of e-EvidenceNotes

3.3 Implementation Phase

This phase is where the actual code of the e-EvidenceNotes were written with inputs from the system design with implementation requirements. Several programming languages, including HTML and JavaScript, are used to develop the proposed system. The programming platform used to code the system is Visual Studio Code 1.62.3 and the connection of the coding with the system's database involves PHP using XAMPP Control Panel.

The system functionality and security were also implemented in this phase. The user interface and database design, logic and requirements outlined in previous phases are implemented and linked the system with the database.

3.4 Testing Phase

The objective of this phase is to test all the units built during the implementation phase and merge them into a system to ensure that the proposed system works as intended. To identify the improvements that can be made to ensure the proposed system is error-free, two forms of testing will be performed: user testing and user acceptance testing.

Unit testing is the testing of individual system modules or components that comprise the system [9]. This testing is carried out for each module of the system to see the units are working as intended. Next, the user acceptance testing is conducted by the end user to verify or accept the software system before deploying it to the production environment [10]. The testing was planned to be executed through a Google Form for potential users to give the feedback from the test plan of the system.

3.5 Maintenance Phase

This phase is performed on the e-EvidenceNotes web-based system that has been developed to improve, update, and enhance the end product so that it remains corrective and adaptable perpetually. If there are any concerns in the user environment, this may include the distribution of patch updates or the release of new versions. If there are any flaws in the developed digital evidence taking notes tool, they will be addressed as soon as possible so that the user may use the system without any issue.

4. Results and Discussion

4.1 Function Testing

Table 2 to Table 4 depict the function testing that is used to verify the system's functionality. The activities will be described in the list below, and the outcomes will be the real results provided by the system. The built system passes all the test plans, with actual outcomes matching those predicted:

Table 2: Function Testing Table for Digital Evidence Taking Notes

No	Checklist Statement	Result
1	Users want to fill in the details of the case notes.	Success
2	Users want to attach file in case note.	Success
3	Users want to see the audit logs of case's update activities.	Success
4	Users want to generate report from the case notes.	Success

Table 3: Function Testing Table for Data Retention

No	Checklist Statement	Result
1	Users can define the period for data retention on a case notebook.	Success
2	The case will be disposed within the period that has been defined from the users.	Success

Table 4: Function Testing Table for Security

No	Checklist Statement	Result
1	The error message not direct indicate which part of the authentication data is incorrect when users log in.	Success
2	Input validation is enforced in required input fields.	Success
3	The password complexity is requiring users to use alphanumeric and special character when creating password.	Success
4	The password length policy is enforced with minimum eight characters.	Success
5	The password should be hashed using Bcrypt hashing function in user database.	Success
6	All the case notes recorded will be timestamped.	Success
7	The notebook should be deleted when it passed its retention period.	Success
8	Multifactor authentication using password and OTP through email before users log in.	Success

4.2 Implementation of Data Retention Policy

The data retention policy has been implemented in e-EvidenceNotes system as a function, as shown in Figure 4 that has been applied in login function. When users log in, the retention() function will be executed automatically. It will capture the current time of users logged with the default time zone set in the source code to compare with the retention date that has been set on each of the notebooks. If the

retention_date of a notebook is less than or equal to the current time, the query of deleting the notebook will be executed and will delete it automatically from the database.

```php
function retention() {
    $db = mysqli_connect('localhost', 'root', '', 'evidencenotes');
    date_default_timezone_set("Asia/Kuala_Lumpur");
    $now = time();

    $today = date('Y-m-d H:i:s', $now);

    $query = "DELETE FROM notebook WHERE retention_date <= ? ";
    $stmt = $db->prepare($query);
    $stmt->bind_param('s', $today);
    $stmt->execute();

    $query2 = "DELETE FROM team_notebook WHERE retention_date <= ? ";
    $stmt2 = $db->prepare($query2);
    $stmt2->bind_param('s', $today);
    $stmt2->execute();
}

?>
```

Figure 4: Implementation of retention() Function

The data retention policy has been applied to the modal window of creating new notebook which the person needs to fill in all of the required information as shown in Figure 5 and the retention date will requires users to choose the date and time from the calendar to set when the notebook will be deleted automatically by the system with the function of retention().

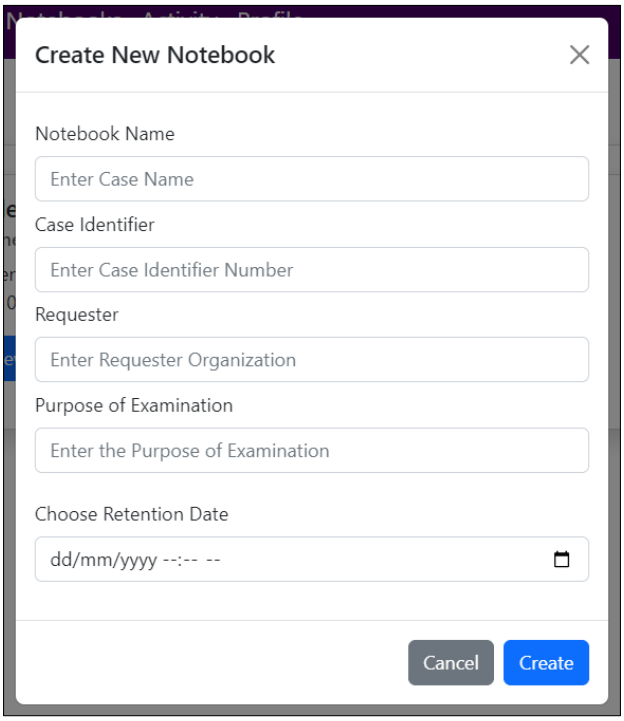

Figure 5: Implementation of Data Retention Policy in Create New Notebook

When the notebook has been created, the notebook will be shown in the List of Notebooks as shown in Figure 6. As the retention date has passed the date and time set by the user, the notebook will automatically disappear from the List of Notebooks.

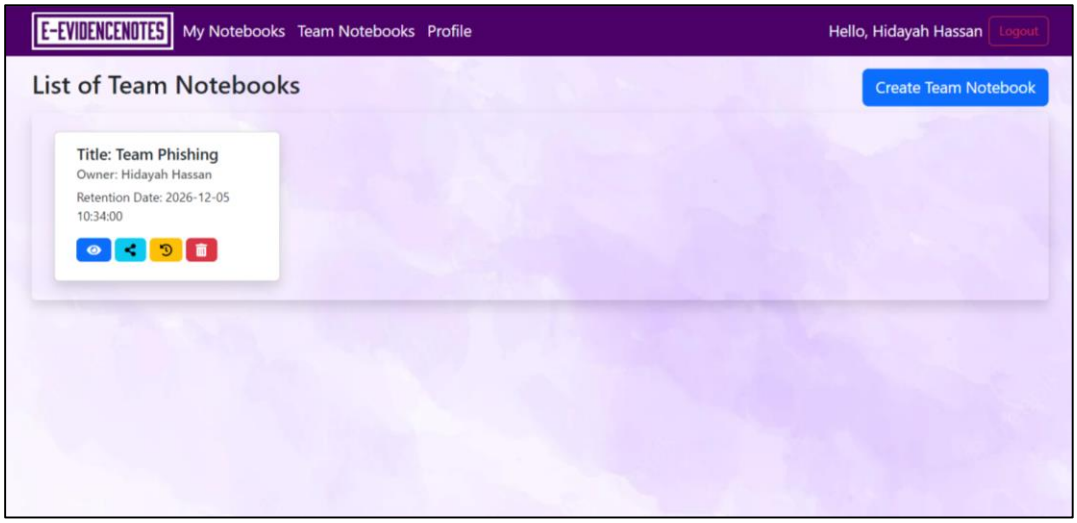

Figure 6: Implementation of Data Retention Policy in Create New Notebook

4.3 Implementation of Input Validation

In module of user registration, input validation has been implemented as shown in Figure 7 to ensure that only properly formed data enters the system's process, preventing flawed data from remaining in the database and causing downstream components to fail [11].

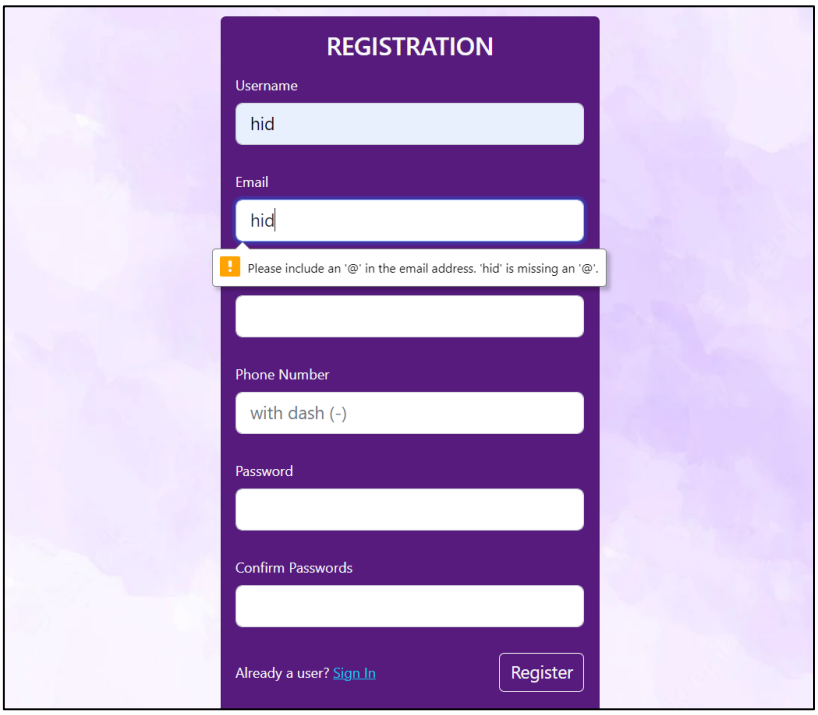

Figure 7: Implementation of Input Validation in User Registration

The input validation in the source code consists of a valid email with a '@' in the email address, phone number that matched its format, password requirements and all the fields are required to be filled as shown in Figure 8.

```
<h1 class="h3 mb-3 font-weight-normal text-center text-light">REGISTRATION</h1>
<div class=" position-relative" style="margin-bottom:30px">
    <label for="emailInput" class="form-label text-light">Username</label>
    <input class="form-control form-control-lg" name="username" type="text"
        value="<?php echo $username; ?>" required>
</div>

<div class=" position-relative" style="margin-bottom:30px">
    <label for="emailInput" class="form-label text-light">Email</label>
    <input class="form-control form-control-lg" name="email" type="email"
        value="<?php echo $email; ?>" required>
</div>

<div class=" position-relative" style="margin-bottom:30px">
    <label for="emailInput" class="form-label text-light">Full Name</label>
    <input class="form-control form-control-lg" type="text" name="fullname"
        value="<?php echo $fullname; ?>"required>
</div>

<div class=" position-relative" style="margin-bottom:30px">
    <label for="emailInput" class="form-label text-light">Phone Number</label>
    <input class="form-control form-control-lg" type="tel" name="phonum"
        placeholder="with dash (-)" pattern="[0-9]{3}-[0-9]{7}" value="<?php echo $phonum; ?>"
        required>
</div>

<div class=" position-relative" style="margin-bottom:30px">
    <label for="emailInput" class="form-label text-light">Password</label>
    <input class="form-control form-control-lg" type="password" id="psw" name="password_1"
        pattern="(?=.*\d)(?=.*[a-z])(?=.*[A-Z]).{8,}"
        title="Must contain at least one number and one uppercase and lowercase letter, and at least 8 or more characters"
        required>
</div>

<div class=" position-relative" style="margin-bottom:30px">
    <label for="emailInput" class="form-label text-light">Confirm Passwords</label>
    <input class="form-control form-control-lg" type="password" name="password_2" required>
```

Figure 8: Source Code of Input Validation Implementation

4.4 Implementation of Password Hashing (using Bcrypt)

Figure 9 shows the implementation of bcrypt hashing function which allows to create a password security infrastructure that grows with computer power and salts every password hash. The added computational effort helps guard against dictionary and brute force assaults by slowing down the attack [12].

```
// register user if there are no errors in the form
if (count($errors) == 0) {
    $password = password_hash($password_1, PASSWORD_BCRYPT);//hash the password before saving in the database

    {
        $query = "INSERT INTO users (username, email, fullname, phonum, password)
                  VALUES('$username', '$email', '$fullname', '$phonum', '$password')";
        mysqli_query($db, $query);
        $_SESSION['success']  = "New user successfully registered!!";

        // get id of the created user
        $logged_in_user_id = mysqli_insert_id($db);

        $_SESSION['user'] = getUserById($logged_in_user_id); // put logged in user in session
        $_SESSION['success']  = "You are now logged in";
                    header('location: login.php');
    }
}
```

Figure 9: Implementation of Bcrypt Hashing

4.5 Implementation of One Time Password (OTP) for Authentication

The OTP has been implemented in the system as the multifactor authentication in this e-EvidenceNotes rather than only using the password to verify the user to log in. The OTP verification page as shown in Figure 10 will appear after the user enters the correct credentials and successfully logs in.

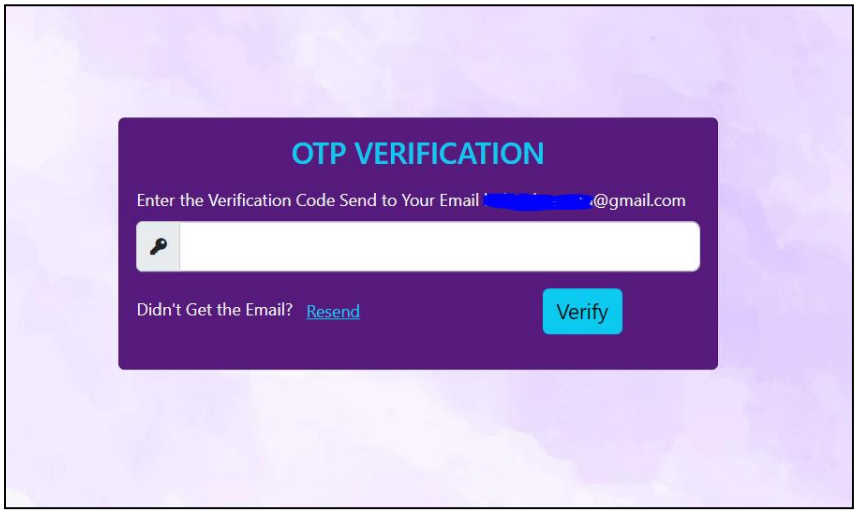

Figure 10: Implementation of One Time Password (OTP)

The function otpverify() has been applied in OTP verification page that will be called inside the login() function as shown in Figure 11. Firstly, the system will verify whether users log in with the correct username and password. If the credentials are correct, it will proceed to the OTP verification page or else it will go back to the login page and the OTP is failed to send.

```php
function login(){
    global $db;
    include('retention.php');
    $username = e($_POST['username']);
    $password = $_POST['password'];

    $query = "SELECT user_id, email as email, username, password as hash FROM users WHERE username = ? ";
    $stmt = $db->prepare($query);
    $stmt->bind_param('s', $username);
    $stmt->execute();
    $stmt->bind_result($id, $email, $username, $hash);
    $stmt->store_result();

    if ($stmt->num_rows == 1) { //To check if the row exists
        if ($stmt->fetch()) { //fetching the contents of the row
            if(password_verify($password, $hash)){
                if(mailer($email,$id,$db)){
                    $_SESSION['otpwrong'] = 0;
                    $_SESSION['login'] = getUserById($id);
                    retention();
                    header("Location: otp.php");
                }
                else{

                    session_destroy();
                    header("Location: login.php");
                    echo "OTP SEND FAILED!";
                }
            }
            else {
                echo "Wrong Password!";
```

Figure 11: Implementation of OTP Verification Inside login() Function

The otpverify() function inside the otp.php is as shown in Figure 12. Users need to enter the verification code sent to their email. The OTP must be valid and if it is expired, the 'otpwrong'=1 will print out the message of "The Code has Expired". If the OTP entered by users match as they received in the email, users will successfully log in and will be redirected to homepage and if it is incorrect, 'otpwrong'=2 will print out the message of the OTP is wrong and will stay on the OTP Verification page.

```php
function otpverify(){
    $otpcodeInput = $_POST['otpcode'];
    $otpcode = $_SESSION['login']['OTPcode'];
    $otpvalid = $_SESSION['login']['OTPvalid'];
    $id = $_SESSION['login']['user_id'];


    $now = time();
    $datetimenow = date('Y-m-d H:i:s', $now);

    if( $datetimenow > $otpvalid ){

        $_SESSION['otpwrong'] = 1;
        header("Location: otp.php");
    }
    else{

        if($otpcodeInput == $otpcode){

            $_SESSION['user'] = $_SESSION['login'];
            unset($_SESSION['login']);
            $id = $_SESSION['user']['user_id'];

            header("Location: home.php");
        }
        else{

            $_SESSION['otpwrong'] = 2;
            header("Location: otp.php");

        }
    }
```

Figure 12: Implementation of otpverify() Function

4.6 Implementation of Timestamp

This e-EvidenceNotes implements the timestamp in the system especially in the case notes module which needs it as it must be recorded the exact time and date when someone create or modify the case notes in Team Notebook module as depicts in Figure 13 to maintain the integrity of users and the system.

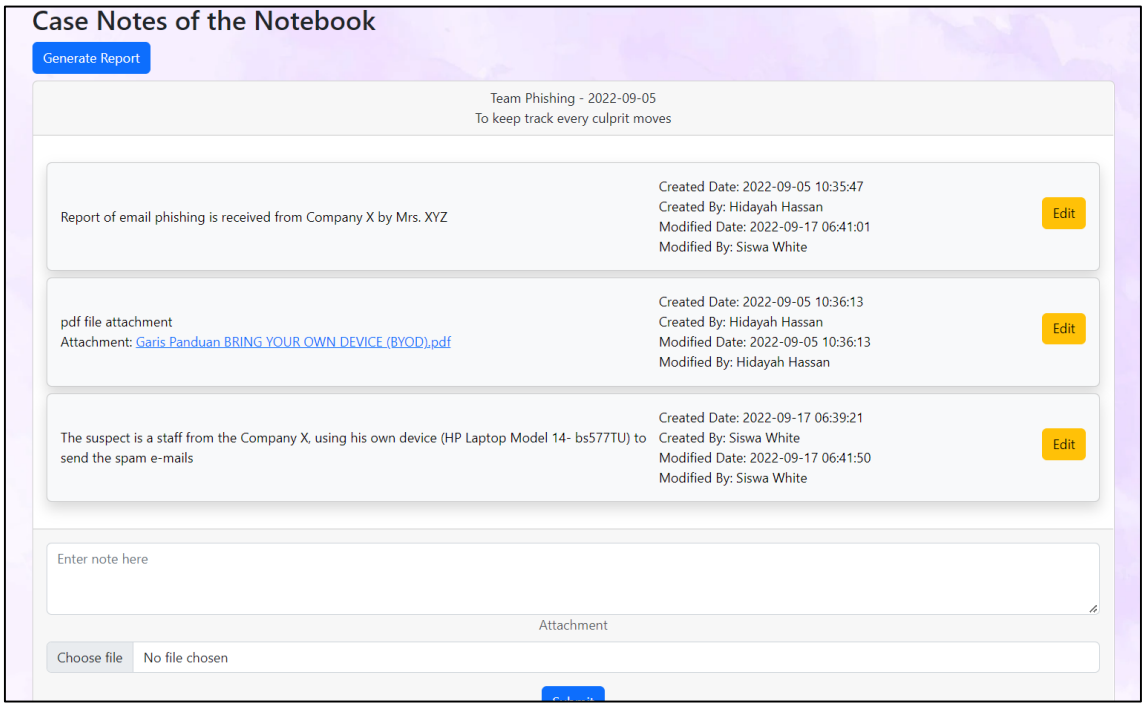

Figure 13: Implementation of Timestamp in Case Notes

From Figure 14, the function addcasenotes() has been applied with the timestamp which collects the exact time and date of the case notes details, created date and modified date to be shown in the Case Notes page.

```php
function addcasenotes(){
    global $db;
    global $now;
    $notebook_id = $_POST['notebook_id'];
    $casenote_detail = $_POST['casenote_details'];
    $createddate = date('Y-m-d H:i:s', $now);
    $modifieddate = date('Y-m-d H:i:s', $now);
    $createdby = $_SESSION['user']['user_id'];
    $modifiedby = $_SESSION['user']['user_id'];

    if($_SESSION['notebook']['type'] == "teamnotebook"){

        $stmt = $db->prepare("INSERT INTO teamcasenotes (teamcasenote_details,teamnotebook_id,created_date,modified_date,created_by,modified_by)
        VALUES (?,?,?,?,?,?)");
        $stmt->bind_param("sissii", $casenote_detail, $notebook_id,$createddate,$modifieddate,$createdby,$modifiedby);
        $stmt->execute();
    }
    else if ($_SESSION['notebook']['type'] == "notebook"){

        $stmt = $db->prepare("INSERT INTO casenotes (casenote_details,notebook_id,created_date,modified_date,created_by,modified_by)
        VALUES (?,?,?,?,?,?)");
        $stmt->bind_param("sissii", $casenote_detail, $notebook_id,$createddate,$modifieddate,$createdby,$modifiedby);
        $stmt->execute();
    }
```

Figure 14: Implementation of addcasenotes() function

The timestamp that is related with the case notes module will also be implemented in the Activity Log of the Team Notebook as shown in Figure 15 and can be accessed by all of the members that has been assigned to the team.

Activity Log

#	User	Activity	Reason	Timestamp
1	Hidayah Hassan	Created Team Notebook 'Team Phishing'		2022-09-05 10:34:59
2	Hidayah Hassan	Assigned 'Hidayah Hassan'		2022-09-05 10:35:01
3	Hidayah Hassan	Created Team Casenote 'test'		2022-09-05 10:35:47
4	Hidayah Hassan	Created Team Casenote 'pdf file attachment'		2022-09-05 10:36:13
5	Hidayah Hassan	Uploaded 'Garis Panduan BRING YOUR OWN DEVICE (BYOD).pdf' to Team Casenote 'pdf file attachment'		2022-09-05 10:36:15
6	Siswa White	Assigned 'Siswa White'		2022-09-17 06:28:06
7	Siswa White	Created Team Casenote 'The suspect is using his own device (HP Laptop Model 14- bs577TU) to send the		2022-09-17 06:39:21
8	Siswa White	Modified Team Casenote 'test' to 'Report of email phishing is received from Company X by Mrs. XYZ'	Detailed information	2022-09-17 06:41:01
9	Siswa White	Modified Team Casenote 'The suspect is using his own device (HP Laptop Model 14- bs577TU) to send th	Detailed information of suspect	2022-09-17 06:41:50

Figure 15: Implementation of Timestamp in Activity Log

4.7 Implementation of Strong Password Policy

As shown in Figure 16, e-EvidenceNotes has implemented strong password policy in registration module to ensure that users will enter strong password with the length must be equal to or greater than 8 characters, must include lowercase letters, uppercase letters and numbers and if it does not comply with the minimum requirements of the password policy then it will be invalid and show the users the pop up error message.

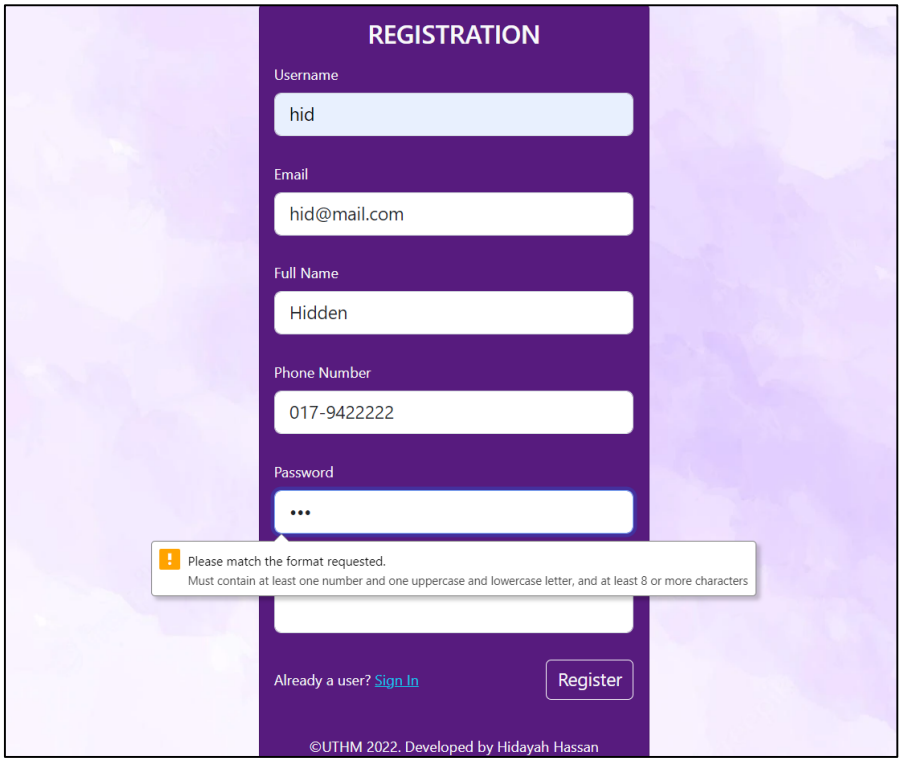

Figure 16: Implementation of Strong Password Policy in Registration Page

The strong password policy implementation in the source code as depicted in Figure 17 consists of validation for each of the requirements, that is the use of lowercase letters, capital letters, numbers, and the password's length. If any of the requirements are not complied with, it will be invalid and will pop up the error message.

```
myInput.onkeyup = function() {
    // Validate lowercase letters
    var lowerCaseLetters = /[a-z]/g;
    if (myInput.value.match(lowerCaseLetters)) {
        letter.classList.remove("invalid");
        letter.classList.add("valid");
    } else {
        letter.classList.remove("valid");
        letter.classList.add("invalid");
    }

    // Validate capital letters
    var upperCaseLetters = /[A-Z]/g;
    if (myInput.value.match(upperCaseLetters)) {
        capital.classList.remove("invalid");
        capital.classList.add("valid");
    } else {
        capital.classList.remove("valid");
        capital.classList.add("invalid");
    }

    // Validate numbers
    var numbers = /[0-9]/g;
    if (myInput.value.match(numbers)) {
        number.classList.remove("invalid");
        number.classList.add("valid");
    } else {
        number.classList.remove("valid");
        number.classList.add("invalid");
    }

    // Validate length
    if (myInput.value.length >= 8) {
        length.classList.remove("invalid");
        length.classList.add("valid");
    } else {
        length.classList.remove("valid");
```

Figure 17: Source Code of Strong Password Policy Implementation

5. Conclusion

The development of e-EvidenceNotes may help more investigators to carry out note taking activity while conducting investigations, particularly those involving cybercrime. Using the system, the user can detect any suspicious activities made by someone on the notes from the timestamp of altered notes. Moreover, the system will be very helpful in organizing and saving the data storage as well as ensuring the security of stored information from data breaches and liabilities.

Acknowledgment

The authors would like to thank the Faculty of Computer Science and Information Technology, UniversitiTun Hussein Onn Malaysia for its support.

References

[1] Augusta Data Storage. 2022. Why an Effective Records Retention Policy is Crucial for Your Business - Augusta Data Storage. [online] Available at: <https://www.augustadatastorage.com/effective-records-retention-policy-crucial-business/>

[2] Kaur, R., & Kaur, A. (2012). Digital forensics. International Journal of Computer Applications, 50(5).

[3] Horsman, G. (2021). Contemporaneous notes for digital forensic examinations. Forensic Science International: Digital Investigation, 37, 301173. https://doi.org/10.1016/j.fsidi.2021.301173

[4] 2018-11-20 SWGDE Requirements for Report Writing in Digital and Multimedia Forensics. SWGDE. (n.d.). from https://www.swgde.org/documents/published-complete-listing

[5] What is a data retention policy? Definition and related FAQs. (2021, March 25). Druva. https://www.druva.com/glossary/what-is-a-data-retention-policy-definition-and-related-faqs/

[6] First Response. 2022. Casenotes. [online] Available at: <https://first-response.co.uk/casenotes/>.

[7] Monolith Forensics. 2022. Monolith Forensics - Digital Forensics Case Management. [online] Available at: <https://monolithforensics.com/>

[8] Product development: Waterfall model: Software development methodology: Entrepreneur's toolkit. MaRS Startup Toolkit. (2019, August 7), from https://learn.marsdd.com/article/product-development-the-waterfall-methodology-model-in-software-development/

[9] Hamilton, T. (2021, December 11). Unit testing tutorial: What is, types, tools & test example. Guru99, from https://www.guru99.com/unit-testing-guide.html

[10] Hamilton, T. (2021, October 8). What is user acceptance testing (UAT)? with examples. Guru99, from https://www.guru99.com/user-acceptance-testing.html

[11] Cheatsheetseries.owasp.org. 2022. Input Validation - OWASP Cheat Sheet Series. [online] Available at: <https://cheatsheetseries.owasp.org/cheatsheets/Input_Validation_Cheat_Sheet.html>

[12] Auth0 - Blog. 2022. Hashing in Action: Understanding bcrypt. [online] Available at: <https://auth0.com/blog/hashing-in-action-understanding-bcrypt/>