SWINBURNE VIETNAM

HO CHI MINH CAMPUS

Computing Technology Inquiry Project – COS10026

Assignment 2 Report

INSTRUCTOR: DR. Eric Le

STUDENT NAMES: Nguyen Hoang Trung

STUDENT ID: SWS00638

HO CHI MINH CITY – April, 2024

WORD COUNT: 1795 EXCLUDING REFERENCES

I. Introduction

This report analyzes the development process and security enhancements implemented for our website in Assignment 2. It also highlights personal contributions, challenges faced, and lessons learned during the project.

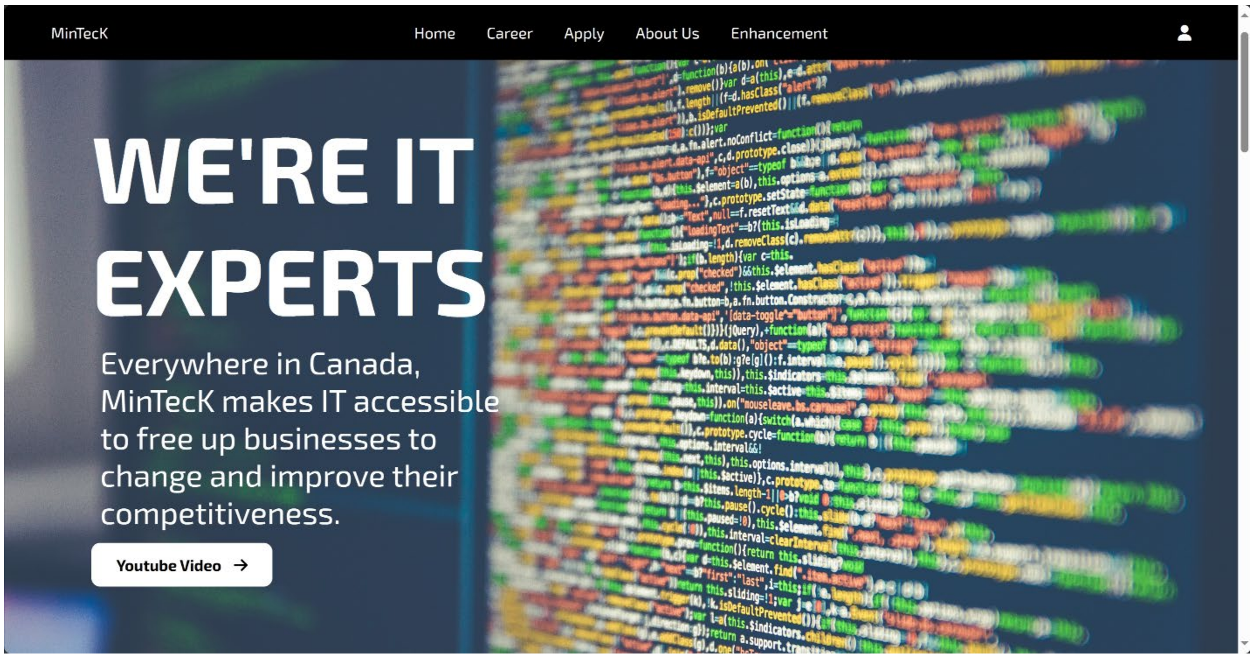
The report is structured as follows:

II. The Website

These websites are enhancements of the websites in Assignment 1. So our main topic for these websites is still making recruitment websites, with some more enhancements. We have different webpages:

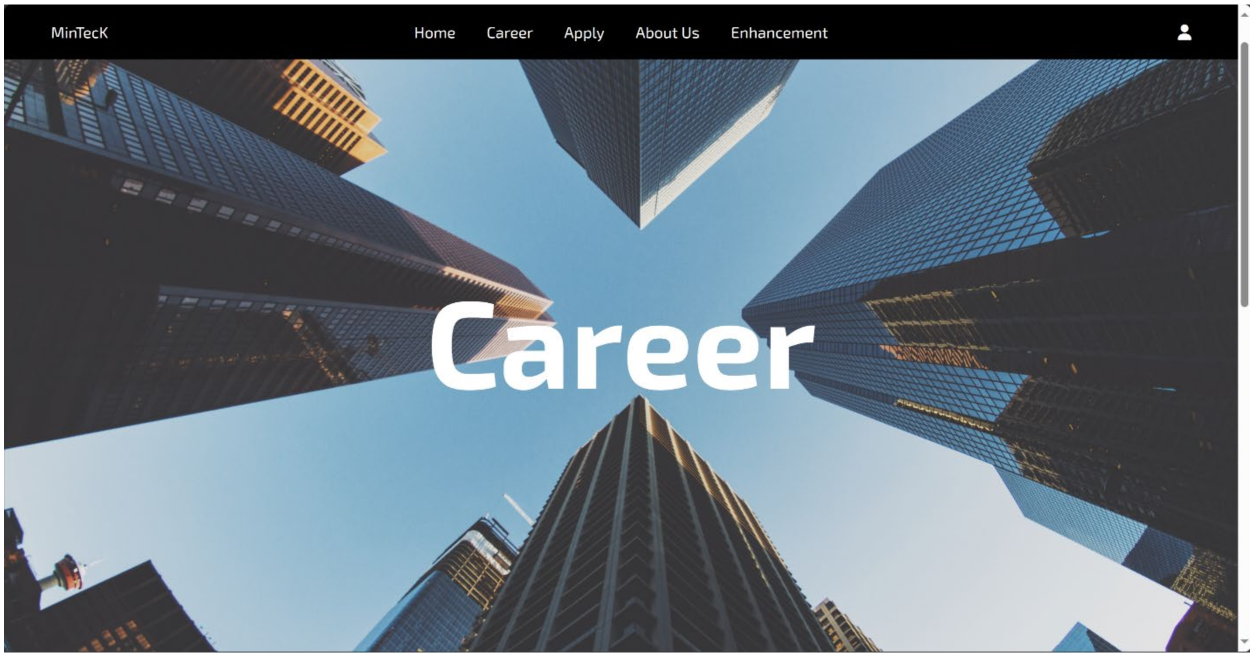
1. Homepage Webpage

Homepage is where you can find some basic information about our company and you can easily navigate through some important webpages.


MinTecK
Home
Career
Apply
About Us
Enhancement
WE'RE IT EXPERTS
Everywhere in Canada, MinTecK makes IT accessible to free up businesses to change and improve their competitiveness.
Youtube Video

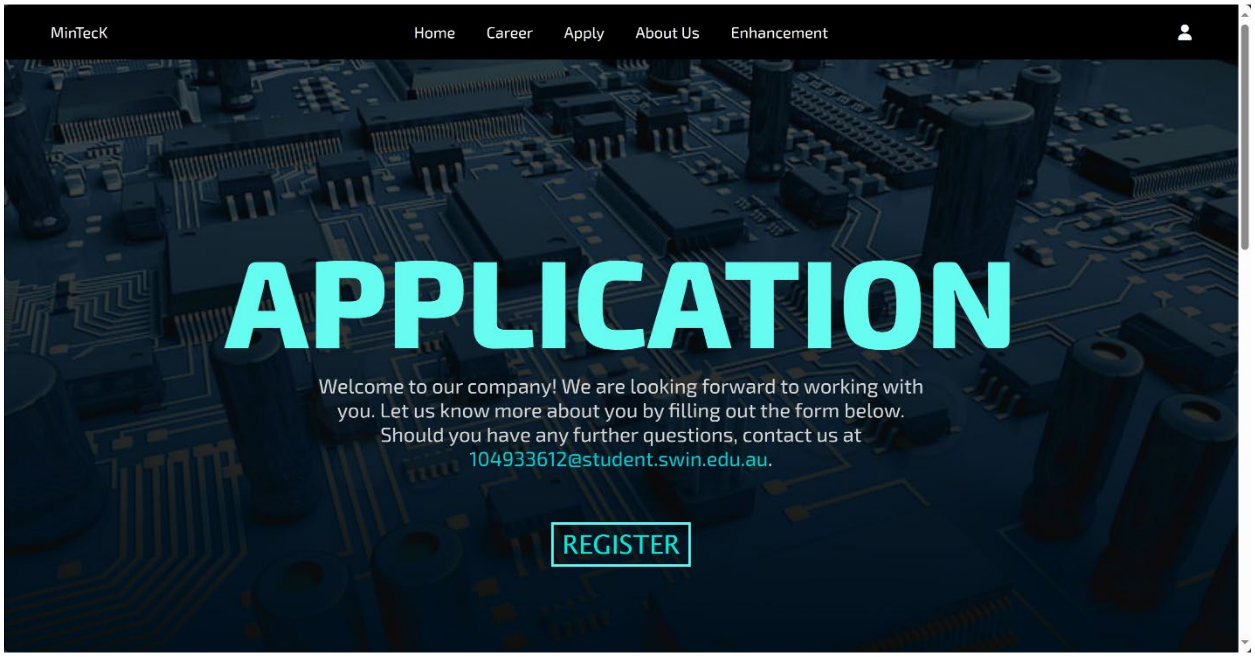
2. Career Webpage

This webpage explains to users what types of jobs we are recruiting. They can find the job's description for more information.



3. Apply Webpage

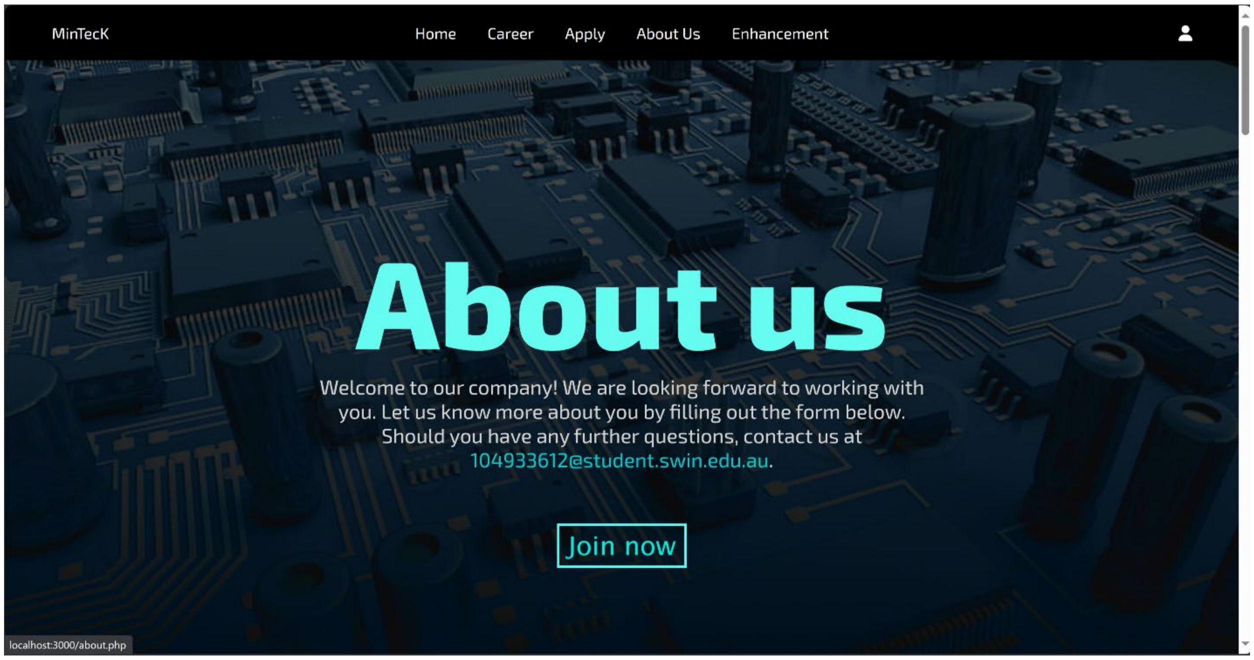
After the job description, there are some people looking forward to applying to our company. So, this is where you can provide us with some personal information.



Welcome to our company! We are looking forward to working with you. Let us know more about you by filling out the form below. Should you have any further questions, contact us at 104933612@student.swin.edu.au.

4. About Us Webpage

To find out more about us, MinTecK, this is where you can do that. It includes founder information.

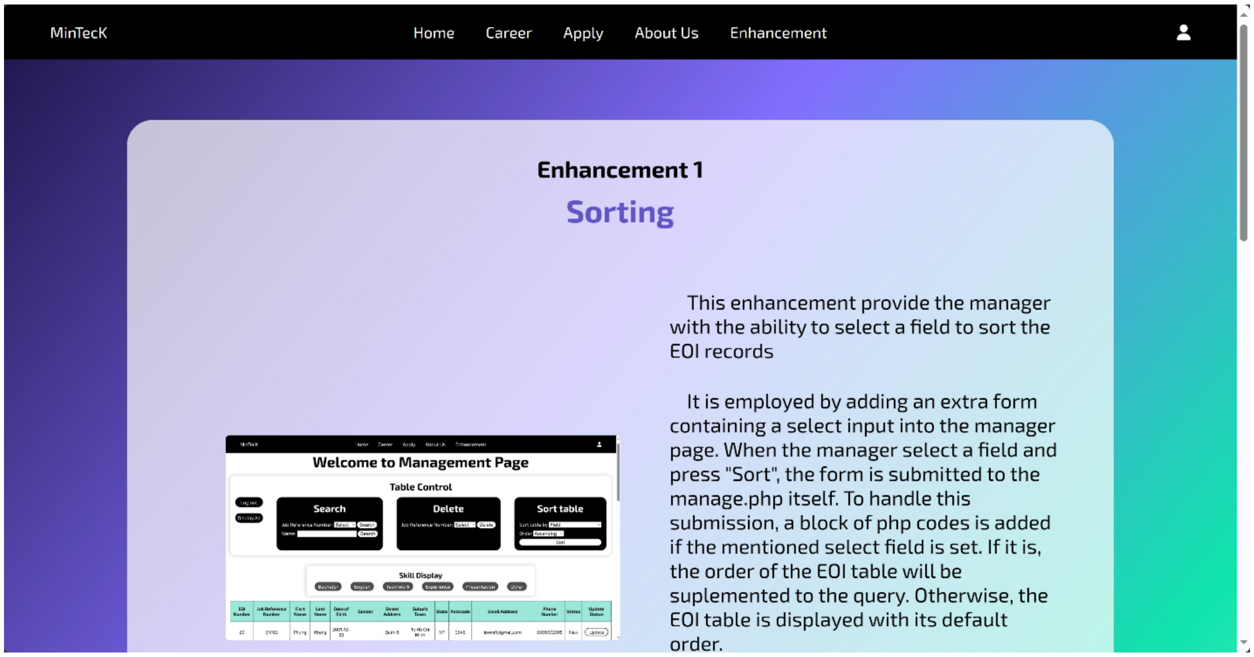


About us

Welcome to our company! We are looking forward to working with you. Let us know more about you by filling out the form below. Should you have any further questions, contact us at 104933612@student.swin.edu.au.

5. Enhancement Webpage

To make the website more functional, we must have some enhancements. This is where you can see the enhancements that we have applied and navigate to them.



Enhancement 1

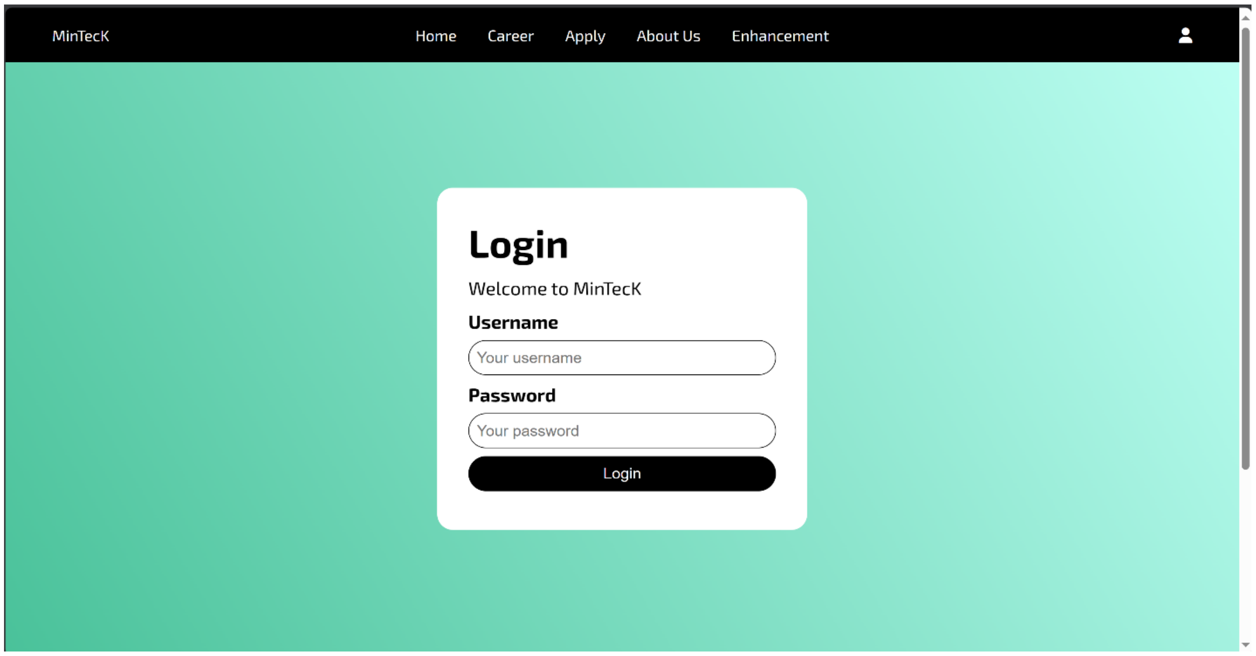
Sorting

This enhancement provide the manager with the ability to select a field to sort the EOI records

It is employed by adding an extra form containing a select input into the manager page. When the manager select a field and press "Sort", the form is submitted to the manage.php itself. To handle this submission, a block of php codes is added if the mentioned select field is set. If it is, the order of the EOI table will be suplemented to the query. Otherwise, the EOI table is displayed with its default order.

6. Log In Webpage

The log-in webpage is built for managers. And it has a functionality that checks login attempts. You are only allowed to enter three times, and after that, the form will automatically lock for 10 seconds.



Login

Welcome to MinTecK

Username

Your username

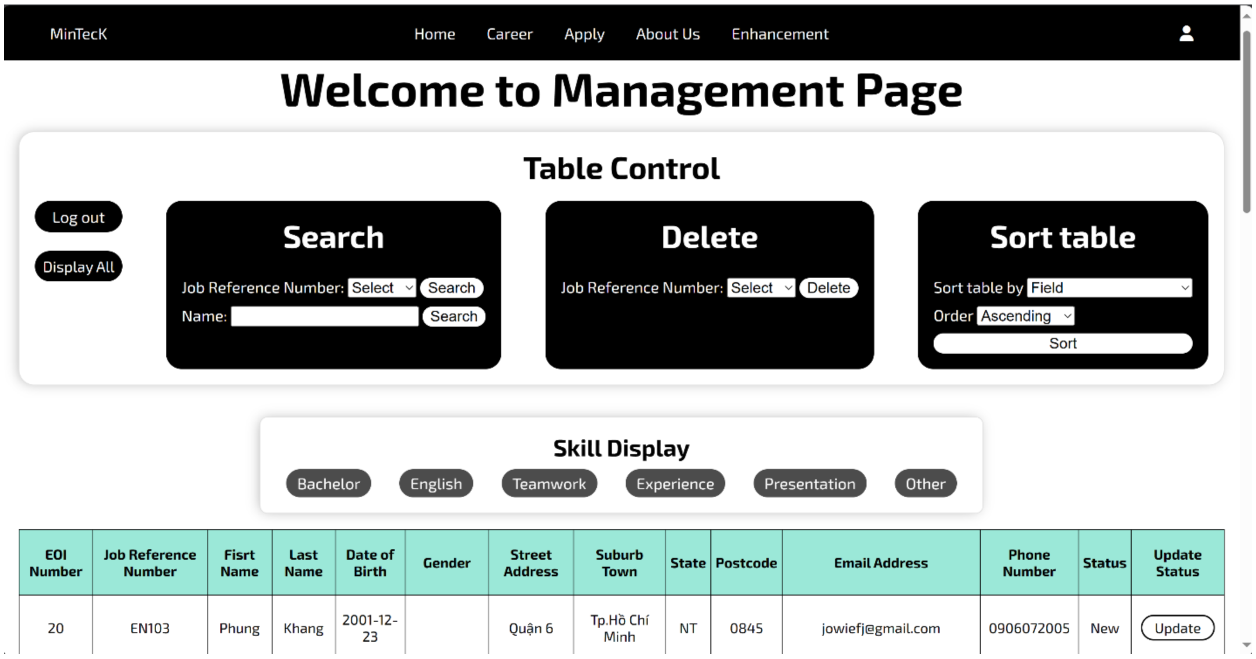
Password

Your password

Login

7. Manage Webpage

After logging in successfully, you will move to the webpage for the manager, where you can see the users' data through a table. As a manager, you can manipulate the table by searching for names, deleting jobs, or sorting the table by a field.



Welcome to Management Page

Table Control

Search

Job Reference Number: Select Search

Name: Search

Delete

Job Reference Number: Select Delete

Sort table

Sort table by Field

Order Ascending

Sort

Skill Display

Bachelor English Teamwork Experience Presentation Other

EOI Number	Job Reference Number	Fisrt Name	Last Name	Date of Birth	Gender	Street Address	Suburb Town	State	Postcode	Email Address	Phone Number	Status	Update Status
20	EN103	Phung	Khang	2001-12-23		Quận 6	Tp.Hồ Chí Minh	NT	0845	jowiefj@gmail.com	0906072005	New	Update

- **Enhancements**

To enhance our websites, we created a login website for managers. This login website has been implemented with some enhanced functions, such as login attempts and preventing users from entering directly into manage.php. We used $_SESSION to store the username and password, so the website can check whether the $_SESSION is true or false to make sure that nobody can directly enter manage.php through the URL. And in manage.php, we also applied one enhancement, which is sorting data in the table. The manager can choose one of the fields in the table to sort the data.

III. Security of the Website

1. Brute Force Attack

a. Definition

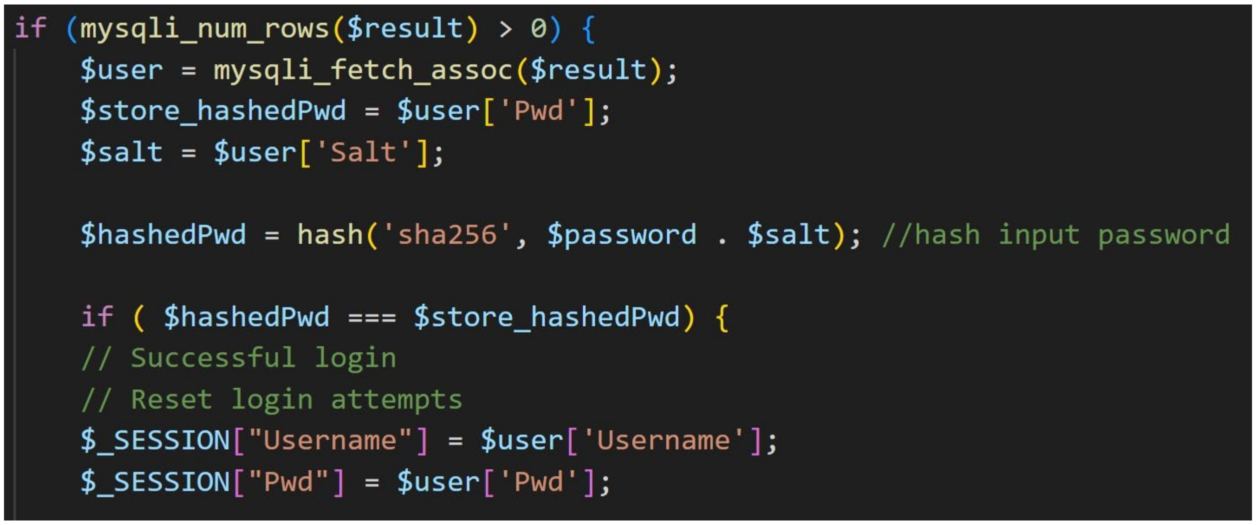
Brute Force Attack is a common tactic used by hackers to gain unauthorized access to systems or accounts by systematically trying all possible combinations of passwords or login credentials (Fortinet, 2023). This method relies on the sheer volume of attempts rather than exploiting system vulnerabilities, making it a popular threat to websites, applications, and databases. According to Externetworks (2023), hackers often use automated scripts to generate various combinations of usernames and passwords, from simple words to alphanumeric characters, until the correct login information is found. Although real-world passwords are usually more complicated, how simple it is for hackers to carry out brute force attacks emphasizes how important it is to have strong security measures in place to protect against them. Therefore, it is crucial for every website or application to prepare for and defend against Brute Force Attacks.

b. How to prevent

As mentioned previously, there are different methods of Brute Force Attack, so security for websites is necessary. These are some security solutions that I have researched:

- ***Password Hashing***

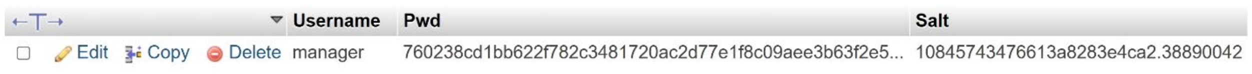
Hashing in this context refers to “chopping something into small pieces” and making it into a mess. So, when applied to computing, it can be understood closely like that. This method will encrypt passwords from plaintext into complicated strings by using different letters, numbers, or algorithms. The reason this method is secure is because it is one-way, which means that you partly cannot reverse the hash action. But even though the hashing algorithm is stronger, it can still be easily cracked using Brute Force attack. Therefore, usually, updating the algorithm and following some password policies are still encouraged.



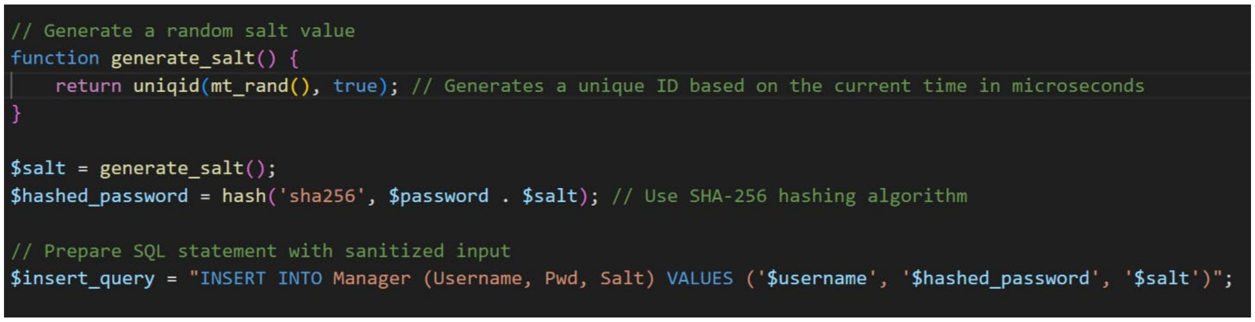
```
if (mysqli_num_rows($result) > 0) {
    $user = mysqli_fetch_assoc($result);
    $store_hashedPwd = $user['Pwd'];
    $salt = $user['Salt'];

    $hashedPwd = hash('sha256', $password . $salt); //hash input password

    if ( $hashedPwd === $store_hashedPwd) {
    // Successful login
    // Reset login attempts
    $_SESSION["Username"] = $user['Username'];
    $_SESSION["Pwd"] = $user['Pwd'];
```



	Username	Pwd	Salt
Edit Copy Delete	manager	760238cd1bb622f782c3481720ac2d77e1f8c09aee3b63f2e5...	10845743476613a8283e4ca2.38890042



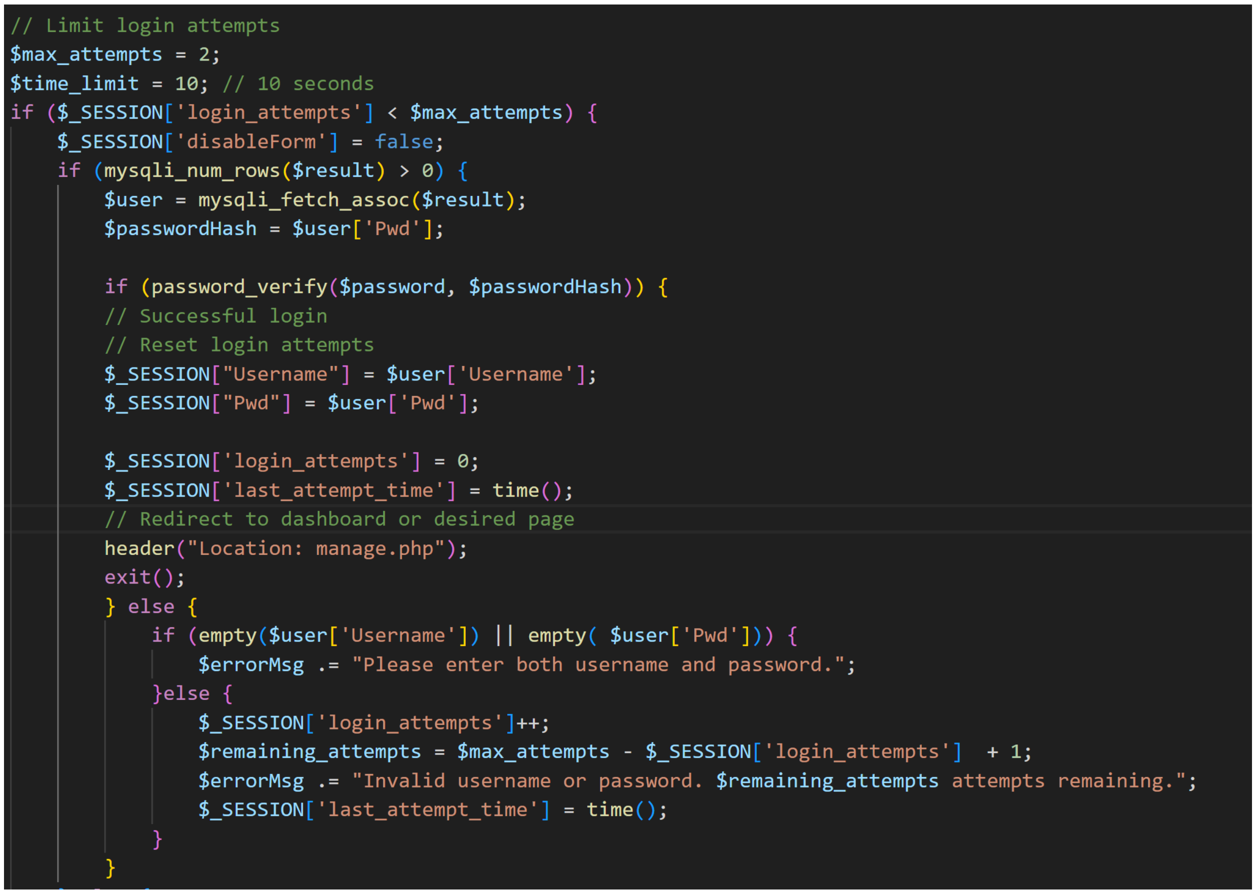
```
// Generate a random salt value
function generate_salt() {
    return uniqid(mt_rand(), true); // Generates a unique ID based on the current time in microseconds
}

$salt = generate_salt();
$hashed_password = hash('sha256', $password . $salt); // Use SHA-256 hashing algorithm

// Prepare SQL statement with sanitized input
$insert_query = "INSERT INTO Manager (Username, Pwd, Salt) VALUES ('$username', '$hashed_password', '$salt')";
```

- ***Limit Login Attempts:***

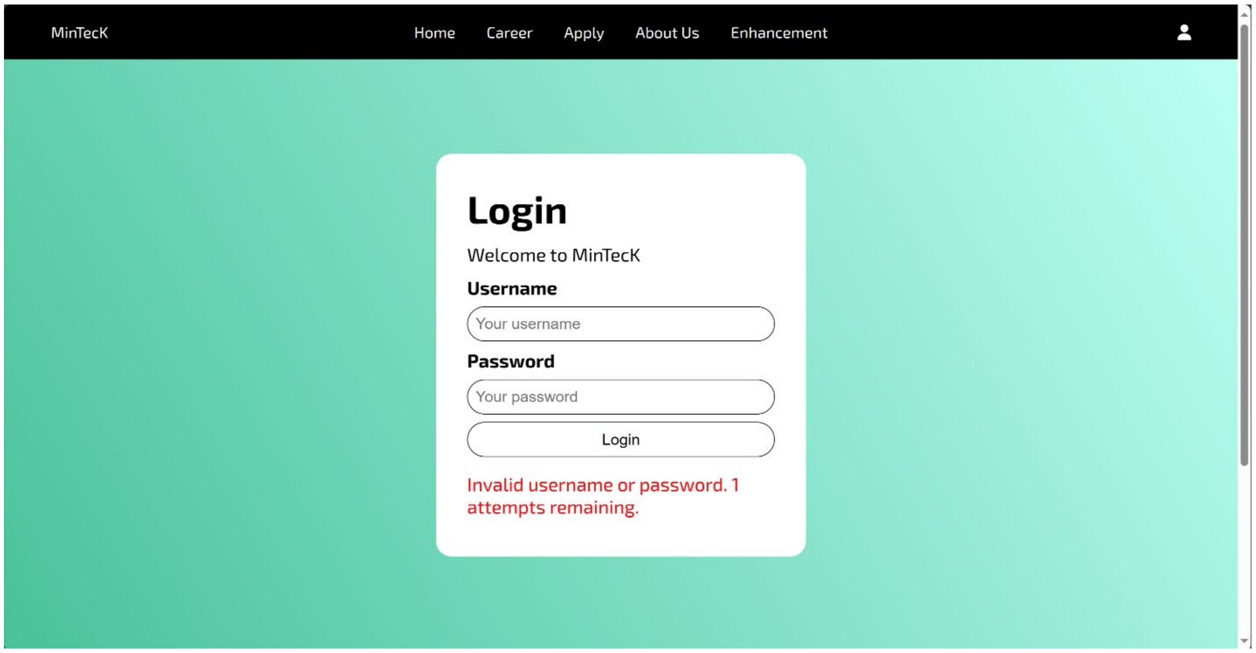
One good solution for security, which I've included in my website enhancements, is implementing account lockout policies. Brute force attacks automatically generate a high volume of login attempts, making it crucial to limit these attempts. For instance, after three unsuccessful login attempts, the login form could be locked for three minutes. Additionally, increasing the lockout duration for subsequent failed attempts can further deter attackers. While I have successfully applied this measure to my assignment, I faced challenges in adjusting the lockout duration based on the number of failed attempts.



```
// Limit login attempts
$max_attempts = 2;
$time_limit = 10; // 10 seconds
if ($_SESSION['login_attempts'] < $max_attempts) {
    $_SESSION['disableForm'] = false;
    if (mysqli_num_rows($result) > 0) {
        $user = mysqli_fetch_assoc($result);
        $passwordHash = $user['Pwd'];

        if (password_verify($password, $passwordHash)) {
        // Successful login
        // Reset login attempts
        $_SESSION["Username"] = $user['Username'];
        $_SESSION["Pwd"] = $user['Pwd'];

        $_SESSION['login_attempts'] = 0;
        $_SESSION['last_attempt_time'] = time();
        // Redirect to dashboard or desired page
        header("Location: manage.php");
        exit();
        } else {
            if (empty($user['Username']) || empty( $user['Pwd'])) {
                $errorMsg .= "Please enter both username and password.";
            }else {
                $_SESSION['login_attempts']++;
                $remaining_attempts = $max_attempts - $_SESSION['login_attempts']  + 1;
                $errorMsg .= "Invalid username or password. $remaining_attempts attempts remaining.";
                $_SESSION['last_attempt_time'] = time();
            }
        }
```



Login

Welcome to MinTecK

Username

Password

Invalid username or password. 1 attempts remaining.

2. SQL Injection

a. Definition

SQL Injection is a type of cyberattack where attackers take advantage of vulnerabilities in input fields inside applications to enter and exhaust unauthorized SQL commands (PortSwinger, n.d.). By inserting specially crafted SQL code, attackers can manipulate the applications' databases, steal personal information, or even take control of the whole system.

b. How to prevent

To safeguard against SQL Injection attacks, developers should use prepared statements with parameterized queries. Prepared statements separate SQL commands from user input data, preventing attackers from injecting malicious SQL code. In PHP, developers can use the mysqli_prepare function to create prepared statements. This function prepares an SQL query with placeholders for parameters, which are then bound to user input data before execution. By ensuring that user input is handled as data rather than executable code, this method reduces the possibility of SQL Injection attacks. Developers can improve application security and shield critical data from modification or unauthorized access by incorporating prepared statements.

```php
$username = sanitise_input($_POST['Username']);
$password = sanitise_input($_POST['Password']);

// Use prepared statement to prevent SQL injection
$stmt = mysqli_prepare($conn, "SELECT * FROM Manager WHERE Username = ? LIMIT 1");
mysqli_stmt_bind_param($stmt, "s", $username);
mysqli_stmt_execute($stmt);
$result = mysqli_stmt_get_result($stmt);
```

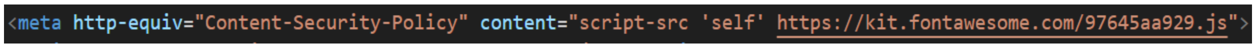
3. Cross Site Scripting (XSS)

a. Definition

One of the most common types of attacks is cross-site scripting. This is accomplished by inserting suspicious codes—most frequently Javascript—into reliable websites in order to get access to databases, steal user cookies, and carry out other serious threats. Cross Site Scripting (XSS) is very common on websites that do not perform enough data sanitization and user input validation (What Is Cross Site Scripting (XSS) and How Does It Work? | Synopsys, n.d.).

b. How to prevent

These factors make it crucial to use methods that check for illegal script injection in order to prevent cross-site scripting attacks (XSS). According to Mozilla (2019), the Content Security Policy restricts the execution of cross-site scripting on a website, thereby mitigating its effects. In this instance, the domain of the web page itself and Font Awesome (since we are utilizing their symbols) are websites from which scripts can be run.



```
<meta http-equiv="Content-Security-Policy" content="script-src 'self' https://kit.fontawesome.com/97645aa929.js">
```

4. Cross-site request forgery (CSRF)

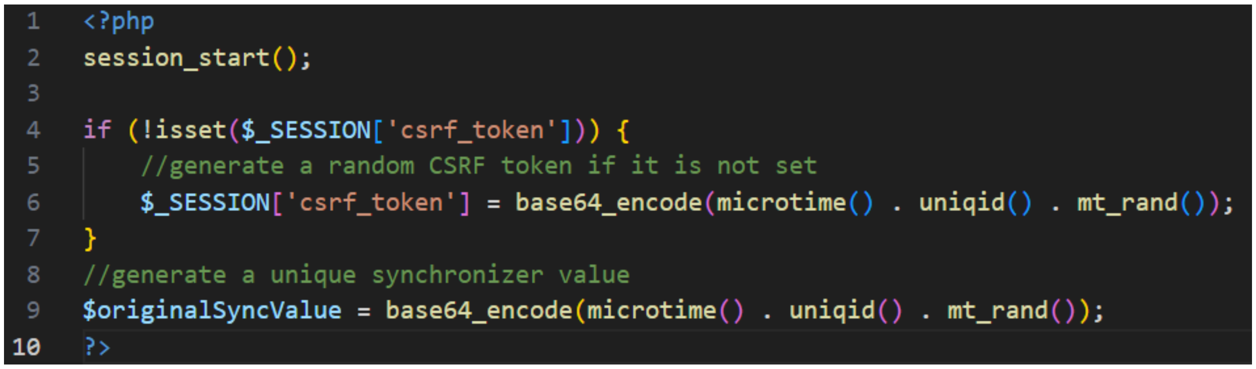
a. Definition

CSRF attacks, by definition, compel a user's browser to deliver a spoof HTTP request to a website. The application identifies this request as qualified since it includes the session cookie and authentication information of authorized users (Conklin & Robinson, n.d.).

b. How ro prevent

According to the Cross-Site Request Forgery Prevention - OWASP Cheat Sheet Series (2012), the Synchronizer Token Pattern is a widely used and efficacious technique to prevent cross-site request forgeries. In order for the web application to differentiate between authorized users and attackers, this method necessitates the generation of a random ID and its inclusion in the users' HTTP request. Below is an example of how I applied the Synchronizer Token Pattern to the apply.php and processEOI.php files.

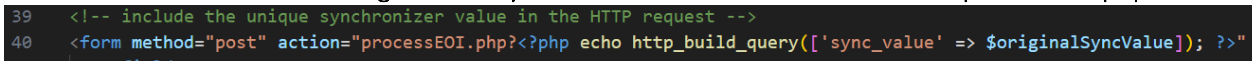
i. A session is started, and a CSRF token and an initial synchronization value are randomly generated. I acknowledge that those values should be cryptographically generated to enhance security. However, that would mean using libraries and extensions, which is out of this assignment's scope. Therefore, a combination of alternative functions is used to achieve a similar outcome to some extent.



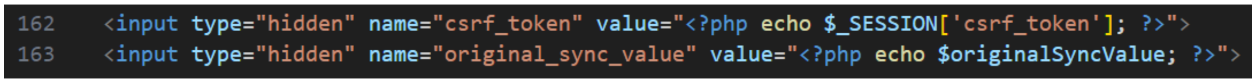
```php
<?php
session_start();

if (!isset($_SESSION['csrf_token'])) {
    //generate a random CSRF token if it is not set
    $_SESSION['csrf_token'] = base64_encode(microtime() . uniqid() . mt_rand());
}
//generate a unique synchronizer value
$originalSyncValue = base64_encode(microtime() . uniqid() . mt_rand());
?>
```

ii. A pair of key - value (containing the original synchronization value) is appended to the header of the HTTP request. Hidden input fields are also inserted into the form to pass the value of the CSRF token and the original synchronization value to processEOI.php.



```html
<!-- include the unique synchronizer value in the HTTP request -->
<form method="post" action="processEOI.php?<?php echo http_build_query(['sync_value' => $originalSyncValue]); ?>"
```



```html
<input type="hidden" name="csrf_token" value="<?php echo $_SESSION['csrf_token']; ?>">
<input type="hidden" name="original_sync_value" value="<?php echo $originalSyncValue; ?>">
```

iii. The file processEOI.php decodes these values and checks if the CSRF token and Synchronizer token are from the authorized user.

IV. Contribution

This is a group assignment, so we decided to divide the workload in half. I focused on developing *manage.php, update.php, delete.php, enhancement1.php*, and *logout.php* as well. And each other of us did one enhancement for the website. Although a report is individual work, we have researched together to find the best solutions for website security to make sure both of us could gain knowledge of this field. Throughout assignment 2, we faced many challenges, such as debugging code or finding the same signal to solve the problem. To overcome these challenges, we held regular meetings to review code and discuss solutions, fostering efficient problem-solving and deeper project understanding. In general, my contributions encompassed both individual tasks and collaborative efforts, reflecting a commitment to achieving our goals through effective teamwork.

V. Refection and Discussion

After this assignment, my teammate and I developed vital cooperation and communication skills because we understood how important they were to reaching our goals. We created an environment of open communication and assistance, utilizing one another's advantages to overcome obstacles and produce outcomes. We not only achieved our objectives through this method, but we also developed a deeper comprehension of productive teamwork. Additionally, this assignment demonstrated how crucial cybersecurity is to online development. In order to protect user data from potential dangers, we carried out an in-depth study and put strong security mechanisms in place. This experience emphasized the need for continuous vigilance and proactive measures to mitigate risks and ensure the integrity of our digital solutions.

Furthermore, by promoting lively debates on a range of web development topics, the project promoted a culture of critical thinking and intellectual rigor. We discussed subjects like database optimization, responsive design, and user experience improvement, which helped our team members have compelling and clear conversations. As demonstrated by our teamwork, cybersecurity efforts, and in-depth conversations on various aspects of web development, our project path, in essence, represents a deep commitment to reflection and continuous progress.

VI. Conclusion

In conclusion, this report highlights the effort to enhance the website's functionality and security. Through the implementation of some enhancements, we have researched and applied defenses against potential cyber threats. Our project experience has provided valuable insights into teamwork and communication. Moving forward, continued vigilance and proactive measures will be crucial to maintaining the website's security and functionality. Overall, this project has been a valuable learning experience, and we are confident in applying these lessons to future endeavors.

VII. References

Arias, D. (2018, April 25). *Hashing Passwords: One-Way Road to Security*. Auth0 - Blog. https://auth0.com/blog/hashing-passwords-one-way-road-to-security/

Becker, J. (2014, August 24). *What is Password Hashing (and How Does It Work)?* Make Tech Easier. https://www.maketecheasier.com/what-is-password-hashing/

Externetworks. (2023, June 20). *What is a Brute Force Attack? How They Work and Ways to Prevent*. Learning Center. https://www.extnoc.com/learn/security/brute-force-attack

Fortinet. (2023). *What is Brute Force Attack? | Definition, Types & How It Works*. Fortinet. https://www.fortinet.com/resources/cyberglossary/brute-force-attack

Ingalls, S. (2022, December 27). *How to Prevent SQL Injection in 5 Steps | eSecurity Planet*. ESecurityPlanet. https://www.esecurityplanet.com/threats/how-to-prevent-sql-injection-attacks/#:~:text=5%20Key%20Methods%20to%20Prevent%20SQL%20Injection%20Attacks

Mozilla. (2019, March 20). *Content Security Policy (CSP)*. MDN Web Docs. https://developer.mozilla.org/en-US/docs/Web/HTTP/CSP

OWASP. (2012). *Cross-Site Request Forgery Prevention · OWASP Cheat Sheet Series*. Owasp.org. https://cheatsheetseries.owasp.org/cheatsheets/Cross-Site_Request_Forgery_Prevention_Cheat_Sheet.html

OWASP. (2021). *SQL Injection Prevention · OWASP Cheat Sheet Series*. Owasp.org; Owasp. https://cheatsheetseries.owasp.org/cheatsheets/SQL_Injection_Prevention_Cheat_Sheet.html

PortSwigger. (n.d.). *What is SQL Injection? Tutorial & Examples*. Portswigger.net; PortSwigger. https://portswigger.net/web-security/sql-injection

What Is Cross Site Scripting (XSS) and How Does It Work? | Synopsys. (n.d.). Www.synopsys.com. https://www.synopsys.com/glossary/what-is-cross-site-scripting.html#:~:text=Definition