PROJECT DOCUMENTATION

MAS WATTAD-318975539

The idea of the Project:

The idea of the project was to design a login page that is secure and user friendly. The login page provides practical cybersecurity features with an educational component. The login page is protected against SQL injections and includes additional security measures such as strong password requirements and encrypted password storage in the database. Additionally, the project also has an educational page and a quiz.

The goal of the project:

The goal of the project is to offer a secure and user-friendly login system that not only protects against common security threats like SQL injections but also enforces strong password policies and uses encrypted password storage. Besides offering security, the goal of the project is to educate users about security practices and security aspects of the page. The project aims to educate users on cybersecurity practices, focusing on raising awareness about social engineering techniques and promoting safe online behaviors which was done in the education page and with the quiz, the aim of project is also to educate the user through warnings on the screen or the practices that were imposed , such as a strong password with uppercase, lowercase, number, and letter password requirements.

Project Description:

Login Page Description:

if the user has already signed up and entered the correct username and password, there are directed to the youin page ,one this page there is a welcome message and a link to the education page and a link to go back to the login page If the username or password contains suspicious patterns (unusually long strings or characters commonly used in SQL injection attacks like --, ', or SQL keywords such as SELECT, INSERT, etc), the system will detect this as suspicious input. Upon entering the password, the user will receive a warning about suspicious input, ensuring that they are aware of potential security risks. If the user does not have an account, they can click the Signup Page link to register. This will redirect them to the Signup Page where they can create a new account with secure password requirements.

Signup page description:

The Signup Page allows users to create a new account by providing a username and password. Upon submission, the system checks if the username already exists in the database. If the username is taken, the user is notified with a message to choose a different username. The system enforces strong validation rules for both the username and password:

The username must be at least 5 characters long and no more than 20 characters.

The password must be at least 10 characters long and include at least one lowercase letter, one uppercase letter, one number, and one special character.

If the username or password fails any of these validation checks, the user is shown a specific error message, guiding them on how to correct the input. The Signup Page is protected against SQL injection attacks by filtering out suspicious characters and patterns (such as --, ', SQL keywords like SELECT, INSERT, DROP, etc.) in both the username and password fields. Once the user provides a valid username and password, the password is securely hashed using bcrypt before being stored in the database. This ensures that even if the database is compromised, the passwords remain secure.

If the registration is successful, the user is shown a message confirming that their account has been created, and they can now log in. There is also a button to generate a random password, as required. This feature is beneficial because it encourages users to create strong, unpredictable passwords, enhancing the overall security of their accounts. Additionally, there is a link to the Education Page, providing users with important information and educational resources related to security best practices.

Education page description:

The Education Page educates users on key security topics: Provides examples of phishing attacks (e.g., fake emails, suspicious links) and tips on how to recognize and avoid them.

Explains SQL injections, examples and prevention.

Guides users on creating strong passwords with a mix of letters, numbers, and special characters, and stresses the importance of unique passwords.

It also has a 5-question quiz to test users' understanding of the topics. Users receive a score showing how many answers they got right out of 5.

It also has a link to return to the Login Page after reviewing the content.

Technical Specifications:

For this project, I worked in Visual Studio Code while connected to a WSL (Windows Subsystem for Linux) Ubuntu environment.

Python (Flask): The web framework used to handle server-side logic, including routing, form processing, and interacting with the database.

SQLite: A lightweight database used for storing user data (like usernames and hashed passwords) and performing SQL queries.

Bcrypt: A library used to securely hash passwords before storing them in the database, ensuring sensitive data is kept secure.

SQL: Used within Flask to interact with the SQLite database to manage user accounts, authenticate users, and prevent SQL injection attacks.

HTML and CSS and JavaScript: Html was Used to create the structure of the web pages, such as the login page, signup page, education page, result page, yourin page and quiz page and CSS Used for styling the pages, making them visually appealing and ensuring a user-friendly design, JavaScript was used in the signup-html only, it was used to enable real-time password generation, enhancing interactivity on the signup page.

Project Structure:

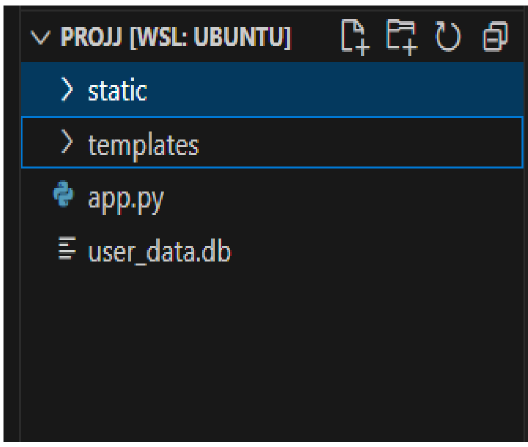
1. static folder: has picture files and style.css files.

2. templates: this folder has all the html files like:

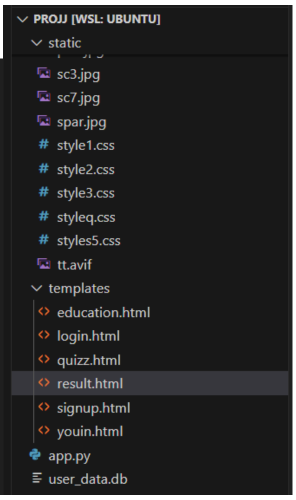
login. html, education.html, signup.html, quizz.html, result.html, youin.html.

3. file app.py: The main entry point for your Flask application.

4. file user_data.db: which has the usernames and the passwords.



```
PROJJ [WSL: UBUNTU]
  > static
  > templates
  app.py
  user_data.db
```



```
PROJJ [WSL: UBUNTU]
  static
    sc3.jpg
    sc7.jpg
    spar.jpg
    style1.css
    style2.css
    style3.css
    styleq.css
    styles5.css
    tt.avif
  templates
    education.html
    login.html
    quizz.html
    result.html
    signup.html
    youin.html
  app.py
  user_data.db
```

Problems I have dealt with:

When I initially designed the project, I didn't implement password encryption, and the users in the database had plain, not hashed passwords. After adding the encryption feature, I encountered an issue where users with normal passwords would cause an error when logging in. Regarding SQL injection prevention, any input that could potentially be part of an SQL injection triggers a warning, preventing such inputs from being processed. Even if the warning system didn't exist, SQL injections would still be blocked due to the input validation and parameterized queries, ensuring that the database remains secure and protected from malicious queries. Throughout this project, I focused heavily on design, ensuring that the user experience was engaging and relevant. It was not easy finding background images that were both fun and relevant to this project. I also faced several design challenges since I had no prior experience, and it required a lot of trial and error to achieve the desired result. Many aspects of the project were new to me, and I had to do a lot of self-studying to learn new technologies and concepts along the way.

Strengths of the Project:

1.The project has multiple Security Features for example it implements measures to prevent SQL injections and enforces secure password policies with validation and encryption, ensuring user data is well-protected.

2.It includes an educational component that raises awareness about phishing, SQL injections, and strong passwords, making the project both practical and informative.

3.The project design is Interactive it Features quiz and real-time warnings for suspicious inputs which actively engage users and make security concepts more relatable.

4.The project has User-Friendly Features like a random password generator that simplifies creating secure passwords, and clear error messages that guide users in meeting security requirements.

Weaknesses of the project:

From a security perspective, the project incorporates several important security features, such as SQL injection prevention, password encryption using bcrypt, and strong password policies. However, there is still room for improvement in securing login pages, particularly for more robust and large-scale implementations. For instance, implementing multi-factor authentication (MFA) would provide an additional layer of protection, and using HTTPS would ensure that all data in transit is encrypted and secure.

This project is designed with small databases in mind, making it a practical starting point for understanding and applying basic security principles in login systems.

Did I reach my goal with this project:

Yes, I believe I successfully achieved the end goal of this project. I developed a secure and user-friendly login system that not only addresses the critical threat of SQL injection but also goes beyond the initial work plan by incorporating additional security measures. These include features like a password generator and enforcing strong password policies to enhance user security.

Furthermore, the project effectively tackles the educational aspect of cybersecurity, particularly social engineering. The education page does an excellent job of explaining and providing practical examples of phishing attempts, highlighting how psychological manipulation is used to deceive users. It also introduces users to the broader cybersecurity landscape, educating them on various threats and the precautions they can take to stay secure.

Overall, the project successfully combines security and education, delivering a comprehensive system that protects users while increasing their awareness of cybersecurity best practices.

Conclusion

Summary

The project successfully delivered a secure and user-friendly login system, preventing SQL injection through input validation and parameterized queries. It ensured password security with strong policies and encryption and educated users on social engineering threats via an interactive education page. Additional features, like a password generator and suspicious input warnings, enhanced the system beyond its initial scope.

Reflections:

In this project, I have gained significant knowledge about cybersecurity and the various vulnerabilities that websites aim to protect against. For instance, I now understand the importance of requiring strong passwords that include uppercase and lowercase letters, numbers, and symbols, as well as how these measures enhance security. Additionally, I learned about phishing attempts, their characteristics, and how to identify and avoid them. Not long ago, I successfully recognized and avoided a phishing attempt to steal my credit card credentials.

This project also marked my first experience with many of the technologies I used, making it a valuable learning process both in terms of technical skills and practical application.

Sources Used During the Project:

I learned and got ideas from the following sources:

- Flask Basics for Login Page:

https://pythonspot.com/login-authentication-with-flask/ .

- HTML and CSS Tutorials:

https://www.youtube.com/watch?v=p1GmFCGuVjw.

https://www.youtube.com/watch?v=wsTv9y931o8

- SQL injections:

https://portswigger.net/web-security/sql-injection

- phishing:

https://www.phishing.org/what-is-phishing

for styling I used icons from: Boxicons : Premium web friendly icons for free

I also watched numerous videos on YouTube to deepen my understanding of various topics relevant to the project.

How to Open and Set Up the Project:

1.Extract and Save:

Extract the zipped project file and save it in the Ubuntu folder. Open the project in Visual Studio Code and ensure it is connected to WSL.

2. install Python:

Open your terminal and install Python 3.13.0 on your laptop

sudo apt update

sudo apt install python3.13

3.Install the following libraries by running:

pip install flask bcrypt flask-wtf flask-sqlalchemy

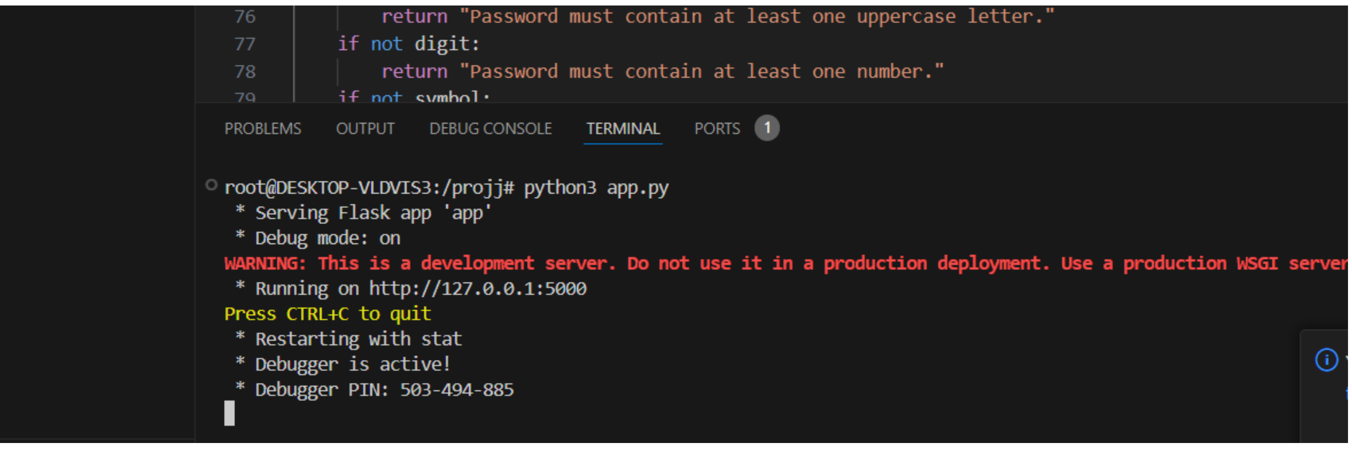
4.Ensure that SQLite is installed on your system. If not, install it with:

sudo apt install sqlite3

5.Start the Flask development server by typing:

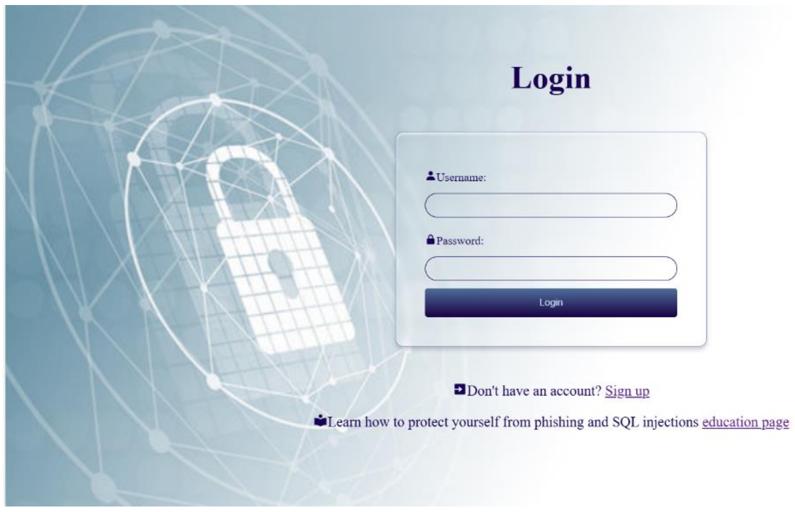
python3 app.py

The terminal will display a local URL (ex: http://127.0.0.1:5000). Open this URL in your web browser to access the application.



```
        return "Password must contain at least one uppercase letter."
    if not digit:
        return "Password must contain at least one number."
    if not symbol:
```

```
root@DESKTOP-VLDVIS3:/projj# python3 app.py
 * Serving Flask app 'app'
 * Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server
 * Running on http://127.0.0.1:5000
Press CTRL+C to quit
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 503-494-885
```



Don't have an account? Sign up

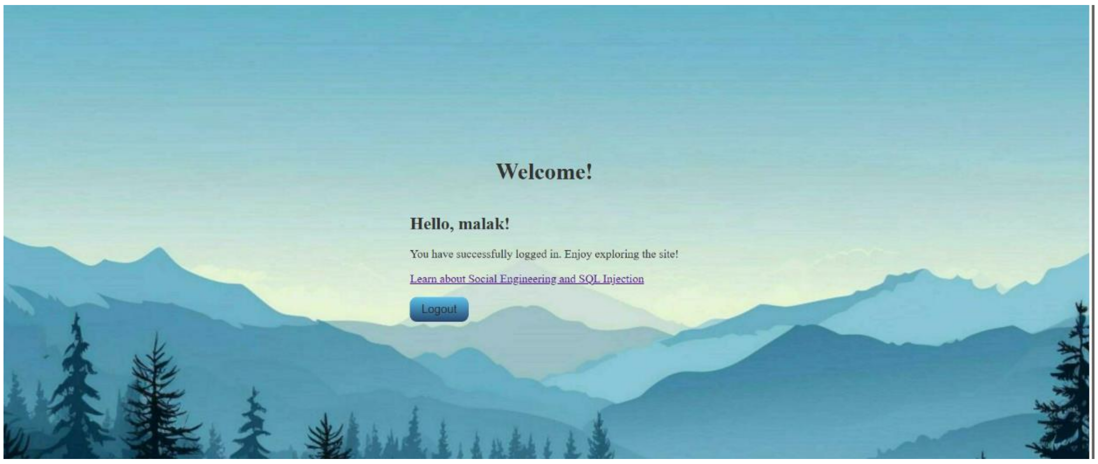
Learn how to protect yourself from phishing and SQL injections education page

This is the main page that appears when opening the project.

Try a username and password of a user that has signed up for example:

Username: Malak   password: Malak2210!!

You will be directed to:



Welcome!

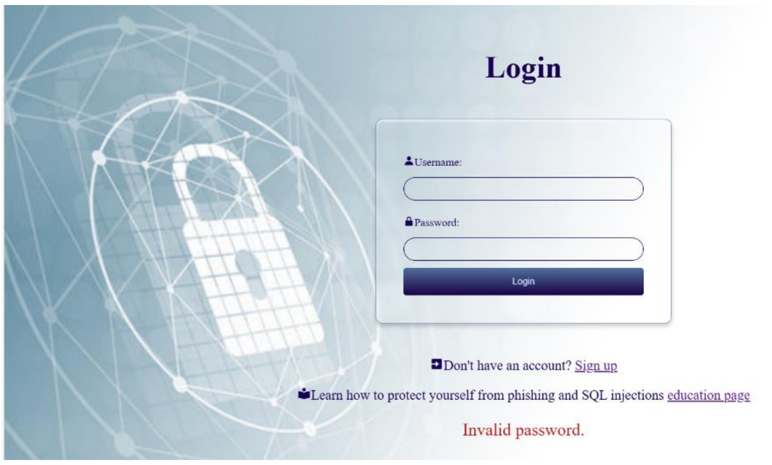
Hello, malak!

You have successfully logged in. Enjoy exploring the site!

Learn about Social Engineering and SQL Injection

If you enter the wrong password, you will get invalid password for example

username: Malak   password: malak2211!!:

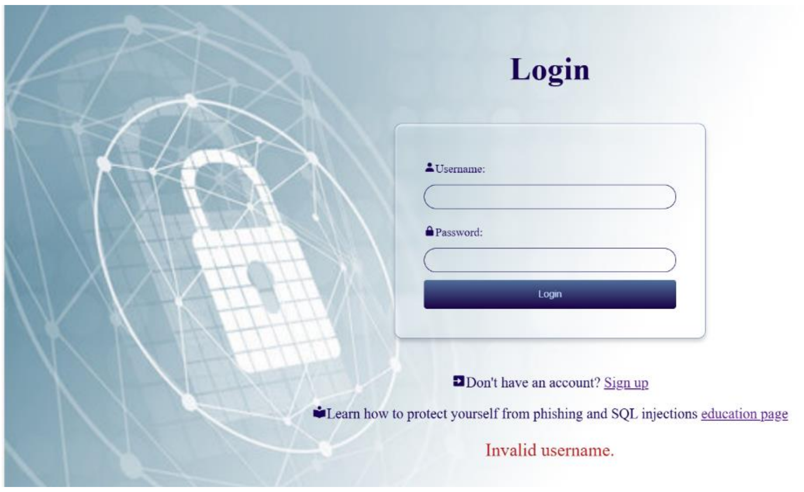


Don't have an account? Sign up

Learn how to protect yourself from phishing and SQL injections education page

Invalid password.

If you enter a username that is not in the databases, you get:

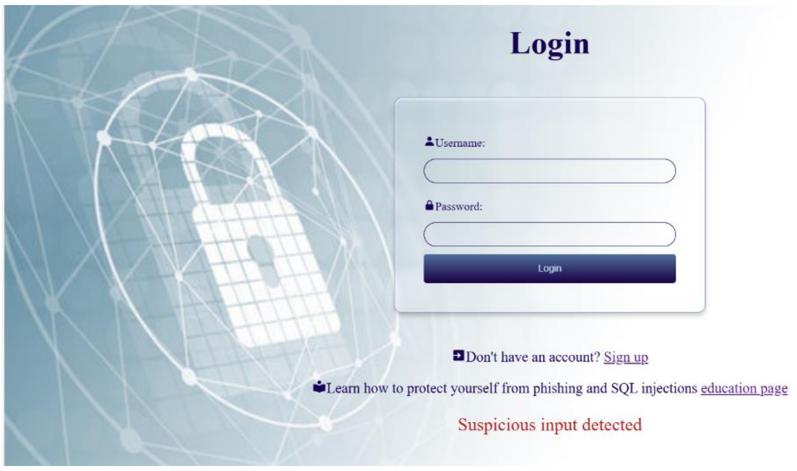


Don't have an account? Sign up

Learn how to protect yourself from phishing and SQL injections education page

Invalid username.

If suspicious input is entered for example username: delete1 password: ajjjnqienijef

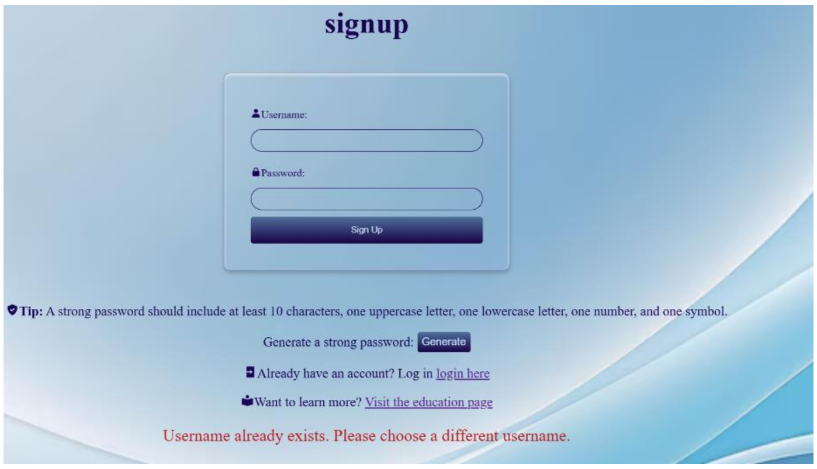


Don't have an account? Sign up

Learn how to protect yourself from phishing and SQL injections education page

Suspicious input detected

If username already exists:



Tip: A strong password should include at least 10 characters, one uppercase letter, one lowercase letter, one number, and one symbol.

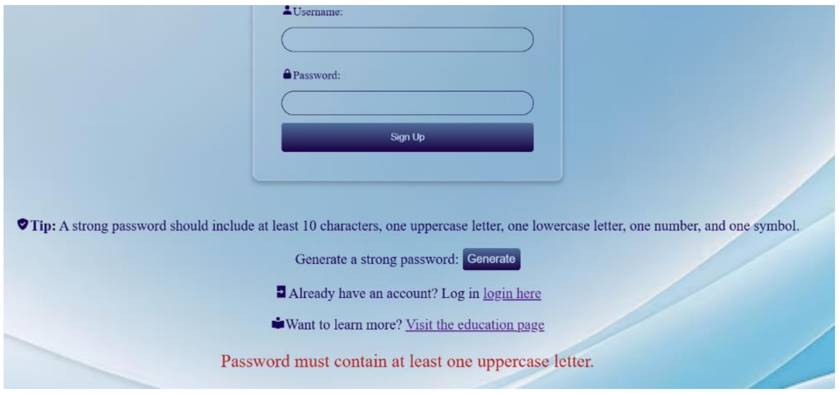
Generate a strong password: Generate

Already have an account? Log in login here

Want to learn more? Visit the education page

Username already exists. Please choose a different username.

If the password does not have at least one uppercase letter and one lower case letter one number or one symbol you will get warning accordingly:



Tip: A strong password should include at least 10 characters, one uppercase letter, one lowercase letter, one number, and one symbol.

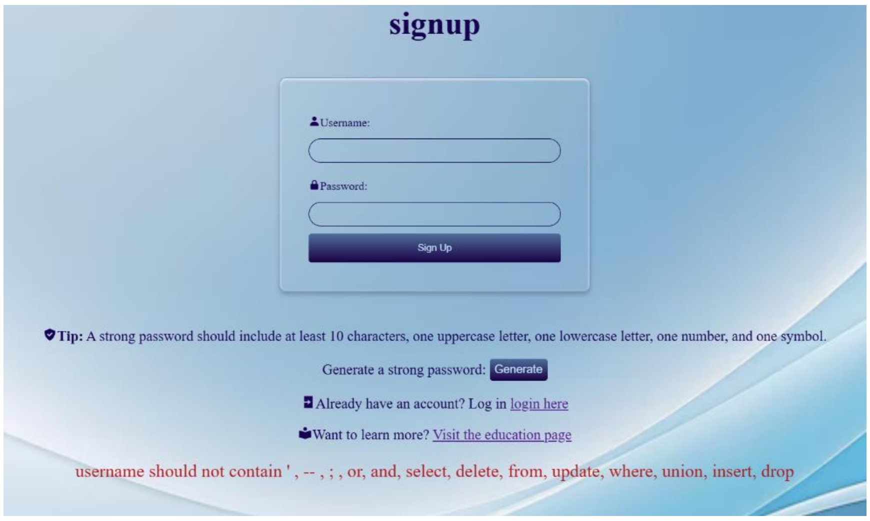
Generate a strong password: Generate

Already have an account? Log in login here

Want to learn more? Visit the education page

Password must contain at least one uppercase letter.

If username contains suspicious input:


signup

Tip: A strong password should include at least 10 characters, one uppercase letter, one lowercase letter, one number, and one symbol.

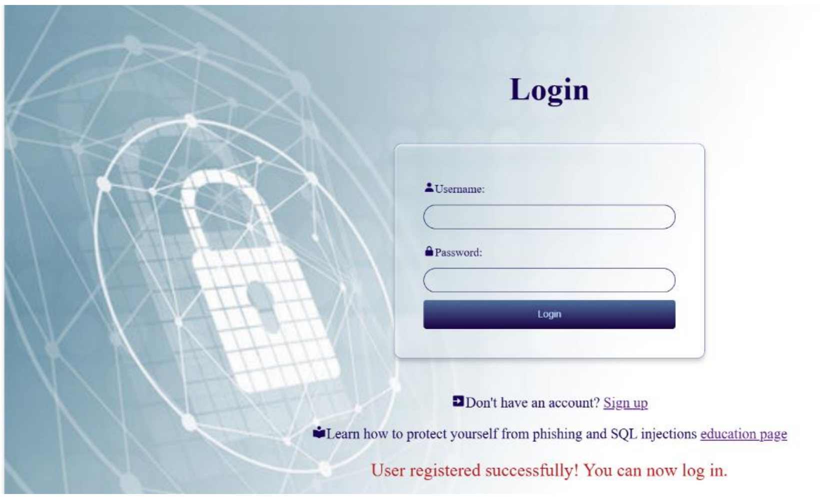
Generate a strong password: Generate

Already have an account? Log in login here

Want to learn more? Visit the education page

username should not contain ' , -- , ; , or, and, select, delete, from, update, where, union, insert, drop

If the username is not used before and does not contain suspicious input and the password is a strong password (has at least 10 characters and contains uppercase letter small case letter number and symbol):

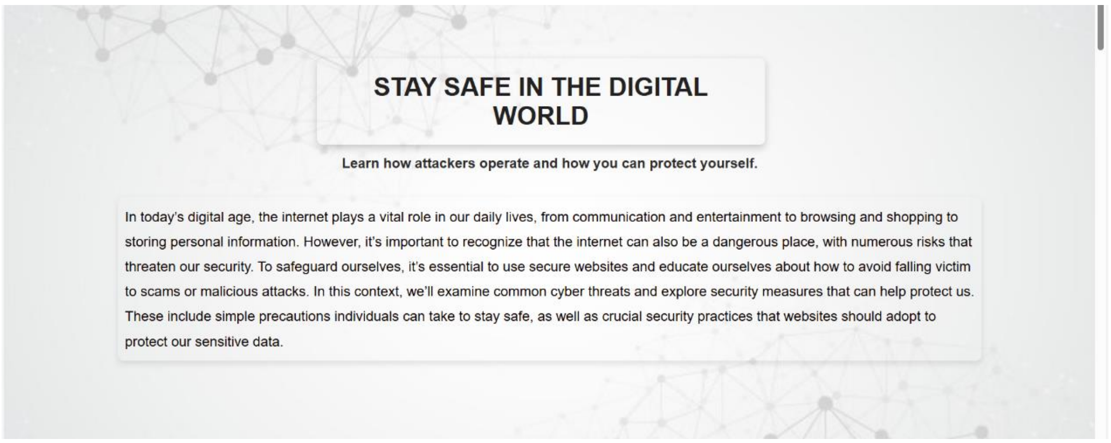

Login

Don't have an account? Sign up

Learn how to protect yourself from phishing and SQL injections education page

User registered successfully! You can now log in.

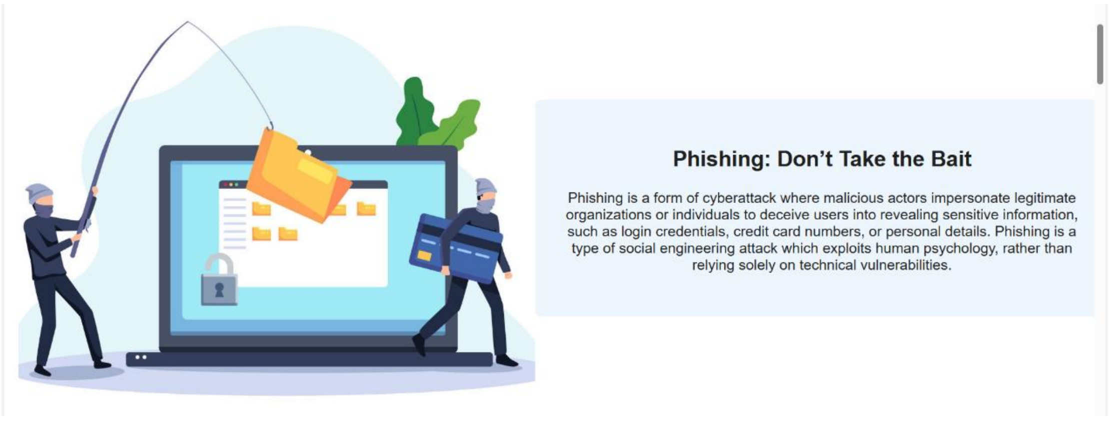
The education page:



STAY SAFE IN THE DIGITAL WORLD

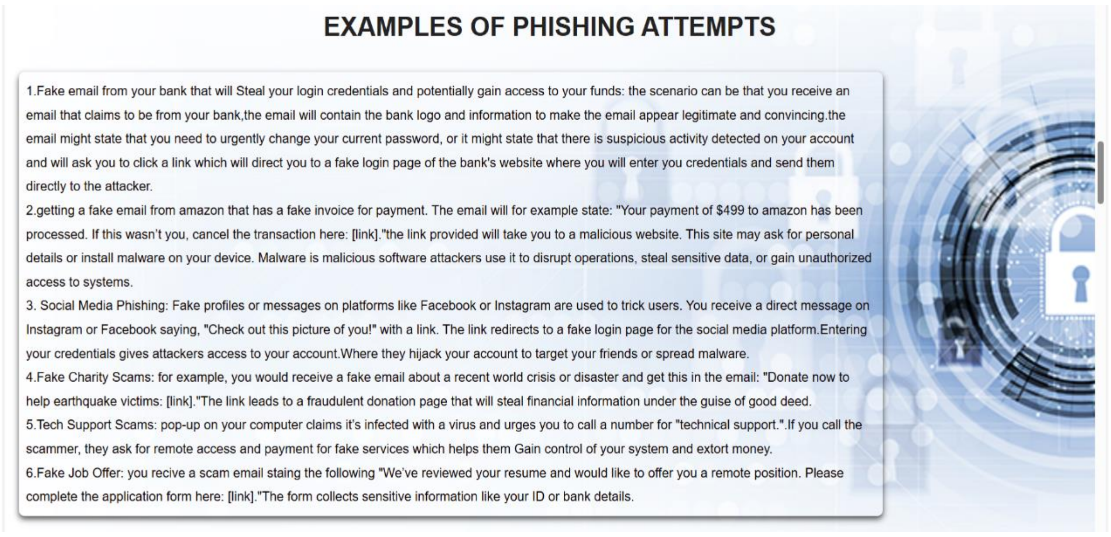
Learn how attackers operate and how you can protect yourself.

In today's digital age, the internet plays a vital role in our daily lives, from communication and entertainment to browsing and shopping to storing personal information. However, it's important to recognize that the internet can also be a dangerous place, with numerous risks that threaten our security. To safeguard ourselves, it's essential to use secure websites and educate ourselves about how to avoid falling victim to scams or malicious attacks. In this context, we'll examine common cyber threats and explore security measures that can help protect us. These include simple precautions individuals can take to stay safe, as well as crucial security practices that websites should adopt to protect our sensitive data.



Phishing: Don't Take the Bait

Phishing is a form of cyberattack where malicious actors impersonate legitimate organizations or individuals to deceive users into revealing sensitive information, such as login credentials, credit card numbers, or personal details. Phishing is a type of social engineering attack which exploits human psychology, rather than relying solely on technical vulnerabilities.

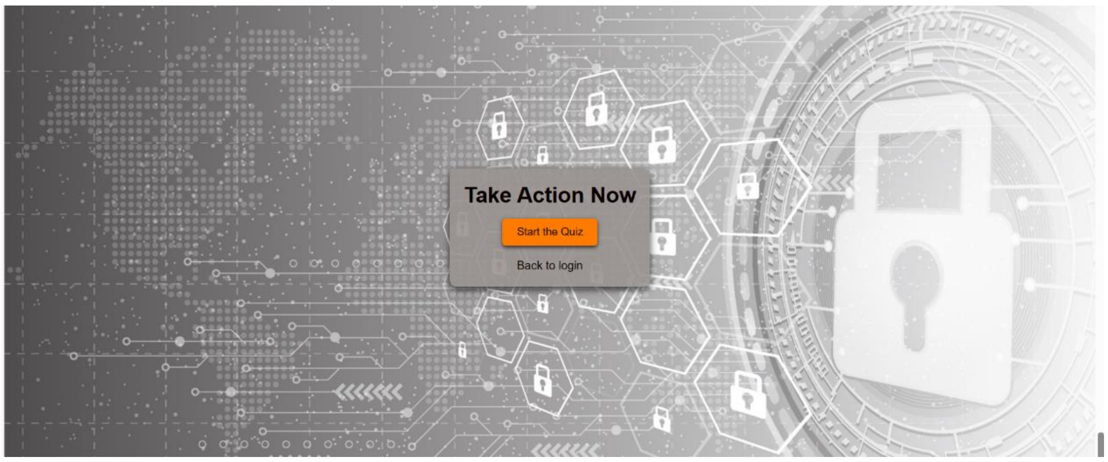


EXAMPLES OF PHISHING ATTEMPTS

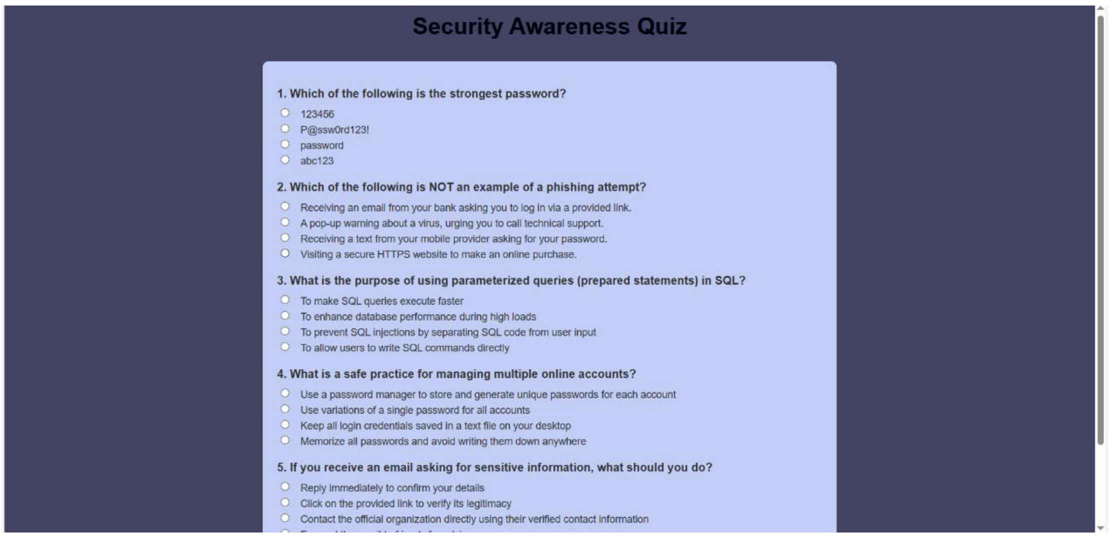
1.Fake email from your bank that will Steal your login credentials and potentially gain access to your funds: the scenario can be that you receive an email that claims to be from your bank,the email will contain the bank logo and information to make the email appear legitimate and convincing.the email might state that you need to urgently change your current password, or it might state that there is suspicious activity detected on your account and will ask you to click a link which will direct you to a fake login page of the bank's website where you will enter you credentials and send them directly to the attacker.
2.getting a fake email from amazon that has a fake invoice for payment. The email will for example state: "Your payment of $499 to amazon has been processed. If this wasn't you, cancel the transaction here: [link]."the link provided will take you to a malicious website. This site may ask for personal details or install malware on your device. Malware is malicious software attackers use it to disrupt operations, steal sensitive data, or gain unauthorized access to systems.
3. Social Media Phishing: Fake profiles or messages on platforms like Facebook or Instagram are used to trick users. You receive a direct message on Instagram or Facebook saying, "Check out this picture of you!" with a link. The link redirects to a fake login page for the social media platform.Entering your credentials gives attackers access to your account.Where they hijack your account to target your friends or spread malware.
4.Fake Charity Scams: for example, you would receive a fake email about a recent world crisis or disaster and get this in the email: "Donate now to help earthquake victims: [link]."The link leads to a fraudulent donation page that will steal financial information under the guise of good deed.
5.Tech Support Scams: pop-up on your computer claims it's infected with a virus and urges you to call a number for "technical support.".If you call the scammer, they ask for remote access and payment for fake services which helps them Gain control of your system and extort money.
6.Fake Job Offer: you recive a scam email staing the following "We've reviewed your resume and would like to offer you a remote position. Please complete the application form here: [link]."The form collects sensitive information like your ID or bank details.

And there is more ……….

At the end you can take a quiz or return to the login page:



Quiz page:



Security Awareness Quiz

1. Which of the following is the strongest password?

○ 123456
○ P@ssw0rd123!
○ password
○ abc123

2. Which of the following is NOT an example of a phishing attempt?

○ Receiving an email from your bank asking you to log in via a provided link.
○ A pop-up warning about a virus, urging you to call technical support.
○ Receiving a text from your mobile provider asking for your password.
○ Visiting a secure HTTPS website to make an online purchase.

3. What is the purpose of using parameterized queries (prepared statements) in SQL?

○ To make SQL queries execute faster
○ To enhance database performance during high loads
○ To prevent SQL injections by separating SQL code from user input
○ To allow users to write SQL commands directly

4. What is a safe practice for managing multiple online accounts?

○ Use a password manager to store and generate unique passwords for each account
○ Use variations of a single password for all accounts
○ Keep all login credentials saved in a text file on your desktop
○ Memorize all passwords and avoid writing them down anywhere

5. If you receive an email asking for sensitive information, what should you do?

○ Reply immediately to confirm your details
○ Click on the provided link to verify its legitimacy
○ Contact the official organization directly using their verified contact information

Note: the users I have in the database and their password are:

1.username: usercs password: M,|3d/d9t?$

2.username:user12 password: WkIS*<4|5]m

3.username:user34 password: L?-61ffP&+b28j

4.username: malak password: Malak2210!!

4.username: khaled password: Khaled2210!!

6.username: noor1 password: Noor12210!!

7. username: mas52 password: Mas522210!!

8.username: noore password: Aa22!adojind