NovaNector Services pvt ltd.

Secure Encryption and Decryption System: Implementation and Analysis

Submitted by: Sandeep Mewara

Code:

CREATE TABLE encrypted_data (

id INT AUTO_INCREMENT PRIMARY KEY,

encrypted_text TEXT NOT NULL,

encryption_key VARCHAR(255) NOT NULL

);

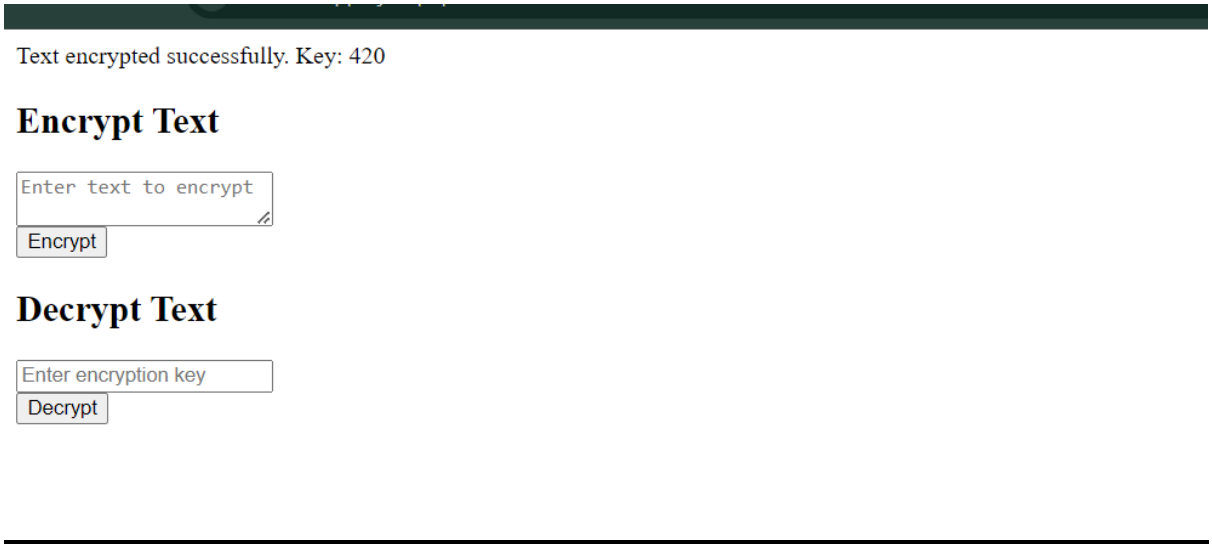
```php
<?php
// MySQL Database Connection
$servername = "localhost";
$username = "root";
$password = "";
$database = "encrypt";

// Create connection
$conn = new mysqli($servername, $username, $password, $database);

// Check connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}

// Function to generate a random encryption key
function generateEncryptionKey() {
    return rand(0, 999); // Generates a random integer
}

// Function to generate a random IV (Initialization Vector)
function generateIV() {
    return openssl_random_pseudo_bytes(16);
}

// Function to encrypt text
function encryptText($text, $key) {
    $iv = generateIV();
    $encrypted = openssl_encrypt($text, "aes-256-cbc", $key, 0, $iv);
    return base64_encode($iv . $encrypted); // Concatenate IV and encrypted
text
}
```

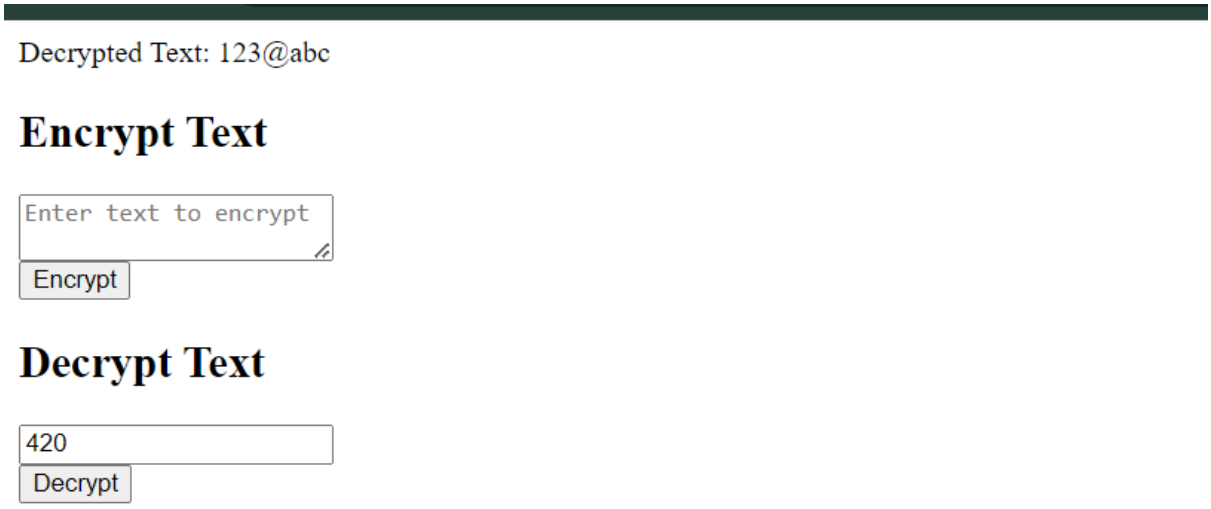
```php
// Function to decrypt text
function decryptText($encrypted_text, $key) {
    $data = base64_decode($encrypted_text);
    $iv = substr($data, 0, 16); // Extract IV from the first 16 bytes
    $encrypted = substr($data, 16); // Extract encrypted text
    return openssl_decrypt($encrypted, "aes-256-cbc", $key, 0, $iv);
}

// Encrypt button clicked
if (isset($_POST['encrypt'])) {
    $plaintext = $_POST['plaintext'];
    $encryption_key = generateEncryptionKey();
    $encrypted_text = encryptText($plaintext, $encryption_key);

    // Store encrypted text and key in the database
    $sql = "INSERT INTO encrypted_data (encrypted_text, encryption_key) VALUES
('$encrypted_text', '$encryption_key')";
    if ($conn->query($sql) === TRUE) {
        echo "Text encrypted successfully. Key: $encryption_key";
    } else {
        echo "Error: " . $sql . "<br>" . $conn->error;
    }
}

// Decrypt button clicked
if (isset($_POST['decrypt'])) {
    $encryption_key = $_POST['encryption_key'];
    $sql = "SELECT encrypted_text FROM encrypted_data WHERE encryption_key =
'$encryption_key'";
    $result = $conn->query($sql);

    if ($result->num_rows > 0) {
        $row = $result->fetch_assoc();
        $encrypted_text = $row['encrypted_text'];
        $decrypted_text = decryptText($encrypted_text, $encryption_key);
        echo "Decrypted Text: $decrypted_text";
    } else {
        echo "No data found for the provided key.";
    }
}

$conn->close();
?>

<!DOCTYPE html>
<html>
<head>
```

```
    <title>Encryption and Decryption System</title>
</head>
<body>
    <h2>Encrypt Text</h2>
    <form method="post">
        <textarea name="plaintext" placeholder="Enter text to
encrypt"></textarea><br>
        <button type="submit" name="encrypt">Encrypt</button>
    </form>

    <h2>Decrypt Text</h2>
    <form method="post">
        <input type="number" name="encryption_key" placeholder="Enter
encryption key"><br>
        <button type="submit" name="decrypt">Decrypt</button>
    </form>
</body>
</html>
```

Output:

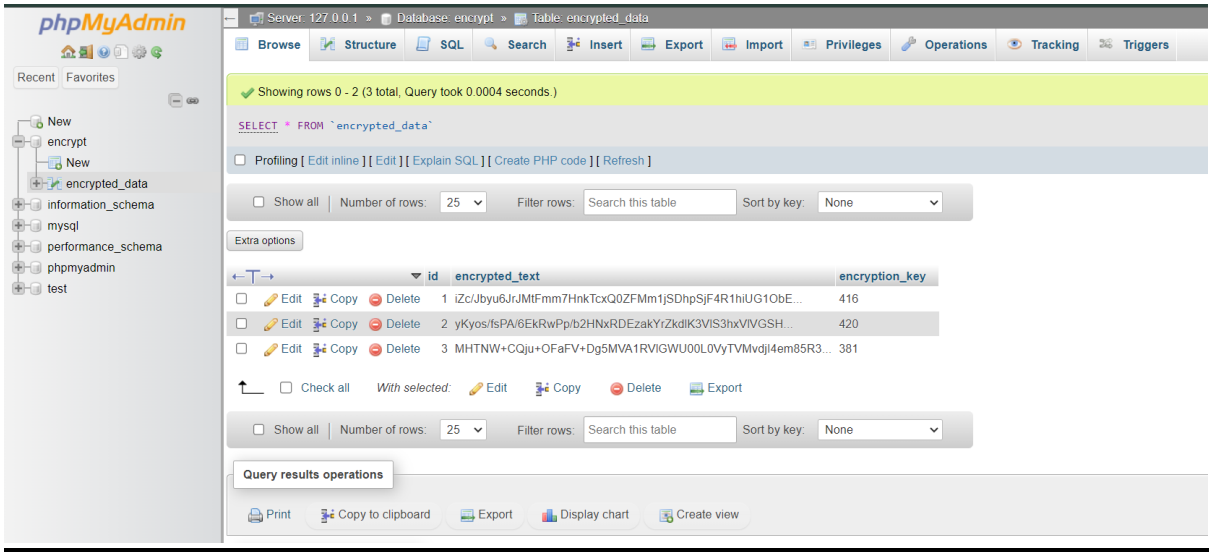
NovaNector Services pvt ltd.





Data in Encryption Form:

NovaNector Services pvt ltd.



- Introduction:
  - Introducing the project: "Creating a Secure Encryption and Decryption System Using PHP and MySQL."
  - Explaining why encryption matters for protecting sensitive data in websites and apps.

- Methodology:
  - Choosing AES-256-CBC encryption for strong security.
  - Using openssl_random_pseudo_bytes() to create random Initialization Vectors (IVs) for extra protection.
  - Explaining how PHP's OpenSSL functions handle encryption and decryption.

- Implementation:
  - Explaining how we built the encryption and decryption features in PHP.
  - Describing the roles of encryption keys and IVs in keeping data safe.
  - Sharing code snippets to show how the system interacts with the database.

- Database Design:
  - Showing the MySQL database layout we created for storing encrypted data and keys.
  - Explaining why it's important to keep keys separate from data for security.

- User Interface:
  - Detailing the simple forms we made for users to encrypt and decrypt text.
  - Explaining how users can input their text and see the encrypted or decrypted result.

NovaNector Services pvt ltd.

- Security Considerations:
  - Highlighting the security features we implemented, like AES-256 encryption and random IV generation.
  - Discussing potential risks such as SQL injection and how we guarded against them.

- Performance Evaluation:
  - Sharing how fast and efficient our encryption and decryption system is.
  - Comparing its performance with other methods to see how well it holds up.

- Recommendations:
  - Offering suggestions for improving security, like using HTTPS and validating user inputs.
  - Ideas for future enhancements such as adding more authentication options or improving error handling.

- Conclusion:
  - Summarizing what we learned and accomplished with our secure encryption and decryption system.
  - Emphasizing the importance of data security in today's digital world.

- References:
  - Listing the sources we used for learning about encryption, PHP, and MySQL.