

Implementation of DES Encryption Algorithm

Supervisor

Dr. Narendran Rajagopalan,

Associate Professor,

NITPY.

Assignment by,

Guguloth Praveen

(CS22B1015)

Introduction:

In today's world, information security plays a critical role in protecting data from unauthorized access and tampering. One of the earliest and most well-known encryption techniques is the Data Encryption Standard (DES), a symmetric-key algorithm for encrypting and decrypting information.

This project presents a simple web application that demonstrates the DES encryption and decryption process using Python (Flask framework) for backend logic and HTML/CSS for the frontend interface.

Users can input a message, provide an 8-character key, and choose whether to encrypt or decrypt their message, helping them understand how classical encryption algorithms work in practice.

Implementation:

```
app.py X
app.py > ...
1
2 from flask import Flask, render_template, request
3 import pyDes
4
5 app = Flask(__name__)
6
7 def des_encrypt(data, key):
8     des = pyDes.des(key, pyDes.ECB, padmode=pyDes.PAD_PKCS5)
9     encrypted_data = des.encrypt(data)
10    return encrypted_data.hex() # Return as hex string for display
11
12 def des_decrypt(data, key):
13     des = pyDes.des(key, pyDes.ECB, padmode=pyDes.PAD_PKCS5)
14     decrypted_data = des.decrypt(bytes.fromhex(data))
15     return decrypted_data.decode()
16
17 @app.route('/', methods=['GET'])
18 def home():
19     return render_template('index.html')
20
21 @app.route('/process', methods=['POST'])
22 def process():
23     message = request.form['message']
24     key = request.form['key']
25     operation = request.form['operation']
26     result = ''
27
28     if len(key) != 8:
29         result = "Key must be exactly 8 characters long."
30     else:
31         try:
32             if operation == 'encrypt':
33                 result = des_encrypt(message, key)
34             else:
35                 result = des_decrypt(message, key)
36         except Exception as e:
37             result = f"Error: {str(e)}"
38
39     return render_template('index.html', result=result)
40
41 if __name__ == '__main__':
42     app.run(debug=True)
43
```

```
app.py index.html X
templates > index.html > ...
1
2 <!DOCTYPE html>
3 <html lang="en">
4 <head>
5     <meta charset="UTF-8">
6     <title>DES Encryption App</title>
7     <link rel="stylesheet" href="{{ url_for('static', filename='style.css') }}">
8 </head>
9 <body>
10     <div class="container">
11         <h1>DES Encryption/Decryption</h1>
12         <form method="POST" action="/process">
13             <textarea name="message" placeholder="Enter your message here" required></textarea><br>
14             <input type="text" name="key" placeholder="Enter 8-character key" maxlength="8" required><br>
15             <select name="operation" required>
16                 <option value="encrypt">Encrypt</option>
17                 <option value="decrypt">Decrypt</option>
18             </select><br>
19             <button type="submit">Submit</button>
20         </form>
21
22         {% if result %}
23         <div class="result">
24             <h2>Result:</h2>
25             <p>{{ result }}</p>
26         </div>
27         {% endif %}
28     </div>
29 </body>
30 </html>
31
```

app.py index.html # style.css

static > # style.css > body

```
1  body {
2      font-family: Arial, sans-serif;
3      background-color: #f5f5f5;
4      text-align: center;
5      padding-top: 50px;
6  }
7
8  .container {
9      background-color: white;
10     padding: 30px;
11     border-radius: 10px;
12     width: 400px;
13     margin: auto;
14     box-shadow: 0px 0px 10px gray;
15 }
16
17 textarea {
18     width: 90%;
19     height: 100px;
20     margin-bottom: 10px;
21 }
22
23 input, select, button {
24     width: 90%;
25     margin-bottom: 10px;
26     padding: 10px;
27 }
28
29 .result {
30     margin-top: 20px;
31     background-color: #e0ffe0;
32     padding: 10px;
33     border-radius: 8px;
34 }
35
```

Output:

DES Encryption/ Decryption

86b60a31d9f477b0

password

Decrypt



Submit

Result:

Praveen

DES Encryption/ Decryption

Praveen

password

Encrypt

Submit

Result:

86b60a31d9f477b0

Explanation:

This project is built with three main components:

1. Frontend (HTML/CSS)

- A clean, simple webpage is designed to accept:
 - The **message** to encrypt or decrypt

- An **8-character key** (as required by DES)
- The operation: **Encrypt** or **Decrypt**
- It provides a user-friendly interface styled with CSS to make the interaction smooth.

2. Backend (Python Flask)

- **Flask**, a lightweight Python web framework, handles the form submission.
- After receiving the input, it uses the **pyDes** library to perform DES encryption or decryption:
 - **Encryption**: Converts plain text into a secure hex string.
 - **Decryption**: Converts the hex string back into readable plain text.
- The backend then sends the result back to the frontend for display.

3. Encryption Logic

- The DES algorithm requires a **fixed 8-character key**.
- Messages are padded automatically to match DES block size requirements.
- Outputs are presented in a readable **hexadecimal format** after encryption, making it easier for users to copy and store.

Conclusion:

This project successfully demonstrates the implementation of a classic DES encryption system inside a web application, combining web development (HTML, CSS) with cybersecurity techniques (encryption algorithms).

It helps users interactively understand how messages can be encrypted and decrypted using symmetric-key cryptography.

While DES is no longer recommended for high-security systems due to its shorter key length and vulnerability to brute-force attacks, it remains a fundamental learning tool for cryptography basics.

This project serves as a foundation for further explorations into more advanced encryption algorithms like AES or RSA and introduces how web applications can be used for real-time security operations.