Collabits Journal
Vol. 1 No. 3 September 2024 : 129-133
E-ISSN : 3046-6709, P-ISSN : 3062-8601
https://publikasi.mercubuana.ac.id/index.php/collabits

Implementasi Teknologi Enkripsi dalam Perancangan Chatbot Berbasis Gemini AI untuk Keamanan Data

Jericho Rayhan Maulana[1], Fajar Satrio Wicaksono[2], Bagus Banil Mustopa[3], Abdullah Al Manan[4], Mohammad Yusuf[5]
[1,2,3,4,5] Fakultas Ilmu Komputer, Universitas Mercu Buana, Jakarta, Indonesia

*Coressponden Author: 41523010025@student.mercubuana.ac.id

Abstract - Perkembangan teknologi informasi dan komunikasi telah mendorong inovasi seperti penggunaan chatbot, yang kini semakin populer dalam berbagai bidang. Namun, seiring meningkatnya penggunaan chatbot, isu keamanan data menjadi semakin krusial. Penelitian ini berfokus pada implementasi teknologi enkripsi dalam perancangan chatbot berbasis Gemini AI untuk meningkatkan keamanan data pengguna. Algoritma enkripsi yang digunakan adalah Advanced Encryption Standard (AES) dengan panjang kunci 256-bit. Proses enkripsi melibatkan inisialisasi kunci, enkripsi teks, dan dekripsi teks. Penelitian ini merancang aplikasi berbasis web menggunakan kerangka kerja Flask dan layanan Google Generative AI. Data pengguna dienkripsi dan disimpan dalam database, kemudian didekripsi saat dibutuhkan. Hasil penelitian menunjukkan bahwa penggunaan AES-256 memberikan tingkat keamanan yang tinggi tanpa mengurangi kinerja sistem secara signifikan. Implementasi enkripsi ini diharapkan dapat menjadi panduan praktis bagi pengembang untuk meningkatkan keamanan chatbot mereka.

Keywords :
Chatbot;
Keamanan Data;
Enkripsi;
Advanced Encryption Standard (AES);
Gemini AI, Flask;
Google Generative AI;
Python;
Cryptography;

Article History:
Received: 06-07-2024
Revised: 15-08-2024
Accepted: 20-09-2024

Article DOI : 10.22441/collabits.v1i3. 28459

1. INTRODUCTION

Perkembangan teknologi informasi dan komunikasi telah membawa perubahan signifikan dalam berbagai aspek kehidupan, termasuk di bidang bisnis, pendidikan, dan sosial. Salah satu inovasi yang semakin populer adalah penggunaan chatbot, yaitu program komputer yang dirancang untuk mensimulasikan percakapan manusia melalui pesan teks atau suara. Chatbot ini sering digunakan untuk memberikan layanan pelanggan, menjawab pertanyaan, dan bahkan melakukan transaksi. Namun, dengan meningkatnya penggunaan chatbot, isu keamanan data menjadi semakin krusial.

Dalam era digital saat ini, keamanan data merupakan salah satu aspek yang sangat penting. Kebocoran data dapat menimbulkan berbagai dampak negatif, seperti pencurian identitas, kerugian finansial, dan kerusakan reputasi. Oleh karena itu, implementasi teknologi enkripsi dalam sistem chatbot menjadi solusi yang relevan untuk melindungi data pengguna dari ancaman yang tidak diinginkan.

Penelitian ini berfokus pada implementasi teknologi enkripsi dalam perancangan chatbot berbasis Gemini AI. Gemini AI merupakan platform kecerdasan buatan yang memiliki kemampuan untuk belajar dari data dan beradaptasi dengan konteks percakapan. Dengan mengintegrasikan teknologi enkripsi, diharapkan chatbot ini dapat memberikan layanan yang lebih aman dan terpercaya bagi pengguna.

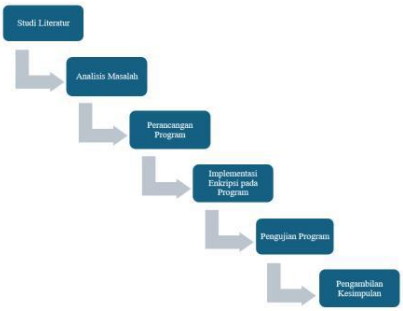
Adapun tujuan utama dari penelitian ini adalah untuk merancang dan mengimplementasikan chatbot berbasis Gemini AI dengan fitur keamanan data yang ditingkatkan melalui teknologi enkripsi. Penelitian ini juga akan menguji efektivitas enkripsi dalam melindungi data percakapan dan mengidentifikasi tantangan yang mungkin dihadapi dalam penerapannya.

Hasil dari penelitian ini diharapkan dapat memberikan kontribusi signifikan dalam pengembangan teknologi chatbot yang lebih aman dan dapat diandalkan, serta memberikan panduan praktis bagi pengembang yang ingin mengimplementasikan teknologi enkripsi dalam sistem chatbot mereka.

2. METHODOLOGY

Pada penelitian ini, algoritma enkripsi yang digunakan adalah Advanced Encryption Standard (AES) dengan panjang kunci 256-bit. Proses penelitian dilakukan dalam beberapa tahap, ditampilkan pada Gambar 1.1 di bawah:

Gambar 1.1 Tahapan Penelitian



Pada proses implementasi Enkripsi pada program, proses Enkripsi dan Dekripsi akan berjalan pada proses berikut :

1) **Inisialisasi Kunci dan Algoritma**
   Membuat kunci enkripsi menggunakan modul Fernet dari pustaka cryptography di Python serta melakukan Inisialisasi algoritma AES dengan kunci yang telah dibuat.
2) **Enkripsi Teks**
   Mengonversi teks pertanyaan menjadi format bytes dan mengenkripsi teks pertanyaan menggunakan kunci AES.
3) **Dekripsi Teks**
   Mengonversi teks terenkripsi kembali menjadi format bytes dan mendekripsi teks menggunakan kunci yang sama.

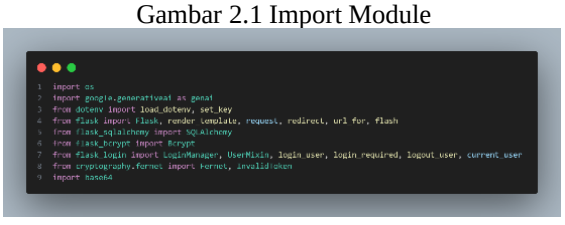
3. RESULTS AND DISCUSSION

Penelitian ini merancang sebuah aplikasi berbasis web yang mengintegrasikan layanan Generative AI dari Google, menggunakan kerangka kerja Flask, dan mengimplementasikan keamanan data dengan enkripsi melalui library cryptography. Kode yang diimplementasikan pada penelitian ini terdiri dari beberapa komponen utama yang mendukung fungsionalitas sistem.

Berikut penjelasan detail mengenai setiap bagian kode:

a) Import Module yang Dipakai

Gambar 2.1 Import Module



```
import os
import google.generativeai as genai
from dotenv import load_dotenv, set_key
from flask import Flask, render_template, request, redirect, url_for, flash
from flask_sqlalchemy import SQLAlchemy
from flask_bcrypt import Bcrypt
from flask_login import LoginManager, UserMixin, login_user, login_required, logout_user, current_user
from cryptography.fernet import Fernet, InvalidToken
import base64
```

Bagian ini mengimpor berbagai modul yang dibutuhkan untuk aplikasi, termasuk modul untuk memuat variabel lingkungan, mengatur Flask, dan melakukan enkripsi.

b) Konfigurasi dan Inisialisasi

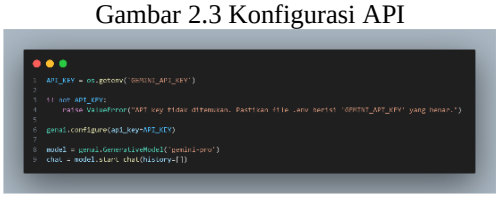
Gambar 2.2 Konfigurasi dan Inisialisasi

```
# Path ke file .env Anda
dotenv_path = r'C:\xampp\htdocs\IB2 Pemrograman Lanjut\IB2 PEMLAN\.env'

# Load environment variables
load_dotenv(dotenv_path)

# Configure Flask app
app = Flask(__name__)
app.config['SECRET_KEY'] = os.getenv('FLASK_SECRET_KEY', 'your_secret_key')
app.config['SQLALCHEMY_DATABASE_URI'] = 'mysql+mysqlconnector://root:@localhost/ai'

# Initialize extensions
db = SQLAlchemy(app)
bcrypt = Bcrypt(app)
login_manager = LoginManager(app)
login_manager.login_view = 'login'
```

Bagian ini mengkonfigurasi aplikasi Flask dengan menetapkan SECRET_KEY dan URI database untuk SQLAlchemy. Selain itu, dilakukan inisialisasi ekstensi Flask lainnya seperti Bcrypt untuk hashing password, LoginManager untuk manajemen login, dan konfigurasi API Google Generative AI dengan kunci API yang dimuat dari variabel lingkungan.

c) API dan Model AI

Gambar 2.3 Konfigurasi API

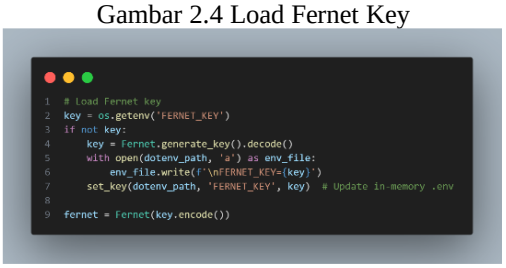


```
API_KEY = os.getenv('GEMINI_API_KEY')

if not API_KEY:
    raise ValueError("API key tidak ditemukan. Pastikan file .env berisi 'GEMINI_API_KEY' yang benar.")

genai.configure(api_key=API_KEY)

model = genai.GenerativeModel('gemini-pro')
chat = model.start_chat(history=[])
```

Konfigurasi API Google Generative AI dengan kunci API yang dimuat dari variabel lingkungan.

d) Enkripsi dan Dekripsi

Gambar 2.4 Load Fernet Key



```
# Load Fernet key
key = os.getenv('FERNET_KEY')
if not key:
    key = Fernet.generate_key().decode()
    with open(dotenv_path, 'a') as env_file:
        env_file.write(f'\nFERNET_KEY={key}')
    set_key(dotenv_path, 'FERNET_KEY', key)  # Update in-memory .env

fernet = Fernet(key.encode())
```

Memuat atau menghasilkan kunci enkripsi dengan Fernet. Menyimpan kunci enkripsi ke file .env jika belum ada.

e) API Gemini Key dan Fernet Key

Gambar 2.5 API Gemini dan Fernet

```
GEMINI_API_KEY = AIzaSyAUMlS58p84R8cj3inByatMG9kJcNVk1C0
FERNET_KEY='D9t8n4K3y0iXfHQcwpxYhlUVfSadVIy6HgGihrfOkEo='
```

API Gemini Key dan juga Fernet Key

f) Model Pengguna dan Riwayat Chat

Gambar 2.6 Pendefinisian Model Pengguna dan Riwayat Chat

Model pengguna (User) dan riwayat chat (ChatHistory) didefinisikan menggunakan SQLAlchemy ORM. Setiap pengguna memiliki username, password yang di-hash, dan relasi dengan riwayat chat.

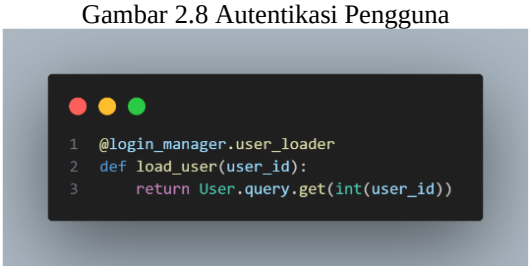
g) Fungsi Enkripsi dan Dekripsi

Gambar 2.7 Fungsi Enkripsi dan Dekripsi

Fungsi encrypt_text dan decrypt_text digunakan untuk mengenkripsi dan mendekripsi data teks menggunakan kunci yang dihasilkan oleh library cryptography.

h) Fungsi Autentikasi Pengguna

Gambar 2.8 Autentikasi Pengguna



Fungsi load_user digunakan oleh Flask-Login untuk memuat pengguna berdasarkan ID mereka dari database.

i) Fungsi Format Response

Gambar 2.9 Fungsi Format Response

Fungsi ini digunakan untuk memformat respons teks agar lebih mudah dibaca dengan membatasi jumlah kata per baris.

j) Rute dan Tampilan

Gambar 2.10 Rute Halaman Utama (Registrasi, Login dan Logout)

Gambar 2.11 Rute Halaman Utama (Home, Creator dan Article)

Bagian ini mencakup rute untuk halaman utama, registrasi, login, logout, halaman pengguna, dan rute untuk berinteraksi dengan bot AI. Data input pengguna disimpan terenkripsi di database, kemudian didekripsi saat ditampilkan kembali.

k) Rute untuk Interaksi Chat

Gambar 2.12 Rute Interaksi Chat

Bagian ini menangani rute untuk mengirim chat ke model generatif dan menyimpan riwayat chat terenkripsi ke database.

l) Menjalankan Aplikasi

Gambar 2.13 Menjalankan Aplikasi

```
if __name__ == '__main__':
    with app.app_context():
        db.create_all()
    app.run(debug=True)
```

Pada bagian akhir, database diinisialisasi dan aplikasi Flask dijalankan dalam mode debug.

A. Penjelasan Web

Kode HTML dan CSS yang disajikan di bawah ini membentuk antarmuka pengguna untuk aplikasi bot chat berbasis web yang menggunakan Bootstrap dan Font Awesome untuk desain yang responsif dan modern. Di sini duraikan struktur dan fungsi dari setiap bagian kode, serta bagaimana elemen-elemen ini saling berinteraksi untuk menciptakan pengalaman pengguna yang intuitif dan menarik.

1. Kolom Kiri (Navigasi dan Riwayat)

Gambar 3.1 Navigasi Bar

```
<div class="col-lg-2 left-side">
        <h3>Santa Crusher.AI</h3>
        <form class="d-flex mb-3" role="search">
          <input
            class="form-control me-2"
            type="search"
            placeholder="Search"
            aria-label="Search"
          />
          <button class="btn btn-outline-success" type="submit">
            Search
          </button>
        </form>
        <div class="wrapper">
          <h5>Dashboard</h5>
          <ul>
            <li>
              <a href="/"> <i class="fa-solid fa-house"></i> Home </a>
            </li>
            <li>
              <a href="creator.html">
                <div class="text-start">
                  <i class="fa-solid fa-users"></i> Creator
                </div>
              </a>
            </li>
            <li>
              <a href="article.html">
                <i class="fa-solid fa-newspaper"></i> Article
              </a>
            </li>
            <li>
              <a href="/logout">
                <i class="fa-solid fa-sign-out-alt"></i> Logout
              </a>
            </li>
          </ul>
        </div>
        <hr class="mt-4" />
        <h5>History</h5>
        <div class="history container text-dark emphasis">
          <div class="d-flex align-items-start">
            <div class="vertical-line"></div>
            <div
              class="history-list"
              style="overflow-y: auto; scroll-behavior: smooth"
            >
              <ul>
                {% for history in user_input_history %}
                <li>{{ history }}</li>
                {% endfor %}
              </ul>
            </div>
          </div>
        </div>
        <hr class="mt-4" />
        <div class="copyright">
          <p>&copy; 2023 Santa Crusher.AI. All rights reserved.</p>
        </div>
      </div>
```

Menyediakan bagian untuk navigasi (Dashboard, Home, Creator, Article, Logout), dan riwayat percakapan dengan bot yang ditampilkan menggunakan loop templating {% for history in user_input_history %}...{% endfor %}.

2. Kolom Konten Utama

Gambar 3.2 Kolom Konten Utama (Informasi Pengguna)

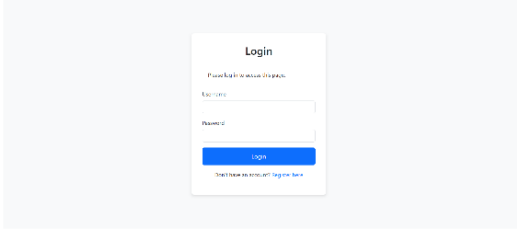
```
<div class="col-lg-10">
        <div class="row">
          <div class="col-lg-3">
            <div class="user-info">
              <div class="circle"></div>
              <p class="mt-0">{{ username }}</p>
            </div>
          </div>
          <hr class="mt-4" />
        </div>
        <form class="d-flex" role="search" action="/chat" method="POST">
          <input
            class="form-control me-2 rounded-input"
            type="search"
            placeholder="Ask me anything..."
            aria-label="Search"
            name="question"
          />
          <button
            class="btn btn-outline-success rounded-button"
            type="submit"
          >
            Submit
          </button>
        </form>
        <hr class="border-top" />
        {% if user_input %}
        <p><strong>You:</strong> {{ user_input }}</p>
        <h4>Answer</h4>
        <p><strong>SantaCrusher.AI:</strong></p>
        <p>{{ bot_response|safe }}</p>
        {% endif %}
      </div>
```

Menampilkan informasi pengguna, formulir input untuk mengajukan pertanyaan kepada bot, dan bagian untuk menampilkan input pengguna serta respons dari bot.

B. Langkah – Langkah Penggunaan Aplikasi

a) Login

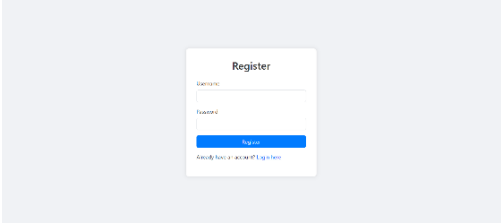
Gambar 4.1 Halaman Login



Jika sudah memilik akun, Pengguna bisa langsung Login

[1] Register

Gambar 4.2 Halaman Registrasi

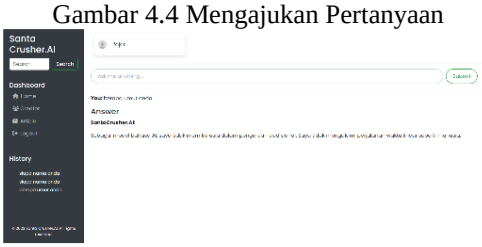


Jika belum memiliki akun, Pengguna akan diarahkan untuk melakukan registrasi. Jika sudah registrasi maka data akan tersimpan di Database MySQL dengan password ter hashing. Ditampilkan pada Gambar 4.3 di bawah ini.

Gambar 4.3 Penyimpanan pada Database setelah Registrasi

[2] Pengguna Mencoba Bertanya ke Chatbot AI

Gambar 4.4 Mengajukan Pertanyaan



Pertanyaan akan tersimpan di Database MySQL dalam bentuk enkripsi. Ditampilkan pada Gambar 4.5 di bawah ini.

Gambar 4.5 Penyimpanan pada Database setelah Bertanya

4. CONCLUSION

Penggunaan algoritma AES-256 untuk enkripsi Chatbot AI memberikan tingkat keamanan yang tinggi karena panjang kunci yang besar. Algoritma AES telah diakui sebagai standar enkripsi yang kuat dan digunakan secara luas dalam berbagai aplikasi keamanan.

Dari segi kinerja, proses enkripsi dan dekripsi dengan AES relatif cepat dan efisien, sehingga tidak akan memperlambat sistem pada proses Bertanya dan Menjawab di Chatbot AI secara signifikan. Namun, penggunaan enkripsi dapat menambah beban komputasi, terutama jika jumlah pertanyaan yang dienkripsi sangat banyak.

Implementasi enkripsi menggunakan pustaka cryptography di Python memudahkan proses enkripsi dan dekripsi dengan menyediakan antarmuka yang sederhana dan intuitif. Kode yang digunakan untuk enkripsi dan dekripsi juga cukup ringkas dan mudah dipahami, sehingga memudahkan pengembang dalam mengintegrasikan fitur ini ke dalam sistem yang ada.

Keandalan sistem juga terjaga dengan baik, karena menggunakan kunci yang sama untuk enkripsi dan dekripsi menjaga integritas data. Proses enkripsi ini memastikan bahwa hanya pihak yang memiliki kunci yang dapat membaca teks asli, sehingga kerahasiaan dan keamanan data Pengguna bisa terjaga.

REFERENCE

[1] Aprizaldi Aprizald, Mhd Arief Hasan and Setiawan, D. (2023). Aplikasi Keamanan Data Berbasis Web Menggunakan Algoritma AES 128 Untuk Enkripsi Dan Dekripsi Data. Jurnal Teknik Informatika, [online] 2(2), pp.85–95. doi:https://doi.org/10.58794/jekin.v2i2.225.

[2] Melenia Bayu Aryanto, Muhlis Tahir, Irma, S., Zuda Nuril Mustofa, Qurrotun Ainiyah and Shelviatus Sundoro (2023). Implementasi Enkrip Dan Dekrip File Menggunakan Metode Advance Encryption Standard (AES-128). Jurnal Ilmiah Sistem Informasi dan Ilmu Komputer, [online] 3(1), pp.89–104. doi:https://doi.org/10.55606/juisik.v3i1.434.

[3] Diana Musabbihah (2018). Perencanaan Sistem Keamanan Pada Jaringan Komunikasi ITS (Intelligent Transport System) Antara OBU dan TMC Server. Tugas Akhir - Departemen Teknik Elektro, Fakultas Teknologi Elektro, Institut Teknologi Sepuluh Nopember, Surabaya. pp. 1-102.

[4] Adminlp2m (2024). Mengungkap Rahasia Algoritma Fernet dalam Enkripsi – Lembaga Penelitian dan Pengabdian Masyarakat. [online] Lembaga Penelitian dan Pengabdian Masyarakat. Available at: https://lp2m.uma.ac.id/2024/02/05/mengungkap-rahasia-algoritma-fernet-dalam-enkripsi/ [Accessed 19 Jul. 2024].