JACKA
Journal of Advanced Computer Knowledge and Algorithms
Vol. 2, No. 1, January, 2025, pp. 19-23
Journal homepage: http://ojs.unimal.ac.id/jacka

Implementation of the Secure Hashing Algorithm-512 (SHA-512) for Sign-Up Page Security in the KelasSeru Tutoring System

Cut Agusniar[1*], Ira Fazira[2], Laili Wahyunita[3]

[1,2] Universitas Malikussaleh, Indonesia
[3] Institut Agama Islam Negeri Palangkaraya, Indonesia

[*]Corresponding Author Email: cutagusniar@unimal.ac.id

Received: 29 December 2024
Revised: 31 December 2024
Accepted: 31 December 2024
Available online: 1 January 2025

Kata Kunci:
Autentifikasi Pengguna, Flask, Keamanan Sistem, Plaintext, Secure Hashing Algorithm-512 (SHA-512), Sistem Bimbingan Belajar

ABSTRAK

Keamanan Sistem untuk Autentifikasi pengguna pada proses *sign-up* menjadi aspek penting dalam melindungi data dari akses tidak sah. Penelitian ini bertujuan mengimplementasikan algoritma *Secure Hashing Algorithm*-512 (SHA-512) pada halaman *sign-up* sistem Bimbingan Belajar KelasSeru berbasis web dengan menggunakan Flask, untuk meningkatkan keamanan data pengguna, terutama kata sandi. SHA-512 dipilih karena kemampuannya menghasilkan *hash* sepanjang 512-bit yang tidak dapat dikembalikan ke bentuk aslinya, sehingga lebih tahan terhadap serangan siber seperti *bruteforce collision attack*. Metodologi penelitian mencakup pengembangan aplikasi berbasis Flask, validasi input, dan enkripsi kata sandi sebelum disimpan ke basis data. Proses enkripsi ini memastikan bahwa kata sandi tidak disimpan dalam bentuk *plaintext*, melainkan dalam bentuk hash yang sulit dipecahkan. Hasilnya menunjukkan bahwa SHA-512 efektif dalam menjaga kerahasiaan kata sandi dan meningkatkan keamanan sistem secara keseluruhan. Selain itu, website juga menampilan fitur tambahan serta tampilan halaman yang berbeda apabila yang melakukan login adalah admin atau peserta. Penelitian ini membuktikan bahwa penerapan SHA-512 pada halaman *sign up* dapat memberikan perlindungan yang signifikan terhadap ancaman siber dan memastikan data pengguna tetap aman, memberikan rasa nyaman dan kepercayaan bagi penggunanya.

Keywords:
Flask, Plaintext, Securte Hashing Algoritm-512(SHA-512), System Security, User Authentication, Tutoring System

ABSTRACT

Security for user authentication Security for user authentication in the sign-up process is an important aspect in protecting data from unauthorized access. This study aims to implement the Secure Hashing Algorithm-512 (SHA-512) algorithm on the sign-up page of the website-based KelasSeru tutoring system using Flask, to improve the security of user data, especially passwords. SHA-512 was chosen because of its ability to produce a 512-bit hash that cannot be returned to its original form, making it more resistant to cyber attacks such as bruteforce collision attacks. The research methodology includes developing a Flask-based application, validating input, and encrypting passwords before saving them to the database. This encryption process ensures that passwords are not stored in plaintext, but in hash form that is difficult to crack. The results show that SHA-512 is effective in maintaining password confidentiality and improving overall system security. In addition, the website also displays additional features and a different page display if the person logging in is an admin or users. This study proves that implementing SHA-512 on the sign up page can provide significant protection against cyber threats and ensure user data remains secure, providing a sense of comfort and trust for its users.

1. INTRODUCTION

Di era digitalisasi saat ini khususnya pengembangan aplikasi web keamanan informasi menjadi salah satu bagian penting yang harus diperhatikan, terutama dalam sistem yang melibatkan autentikasi pengguna. Halaman *sign up* ataupun pendaftaran akun merupakan pintu gerbang pertama dalam menjaga integritas data pengguna agar terhindar dari akses yang tidak sah. Oleh sebab itu, data yang paling sensitif seperti kata sandi (*password*), harus disimpan dan dikelola dengan cara yang aman untuk mencegah terjadinya pencurian atau kebocoran data yang dapat melanggar privasi. Seiring dengan berkembangnya serangan siber yang semakin kompleks, organisasi dan pengelola aplikasi web harus memperkuat upaya pencegahan terhadap serangan yang berfokus pada kata sandi untuk menjaga integritas dan kerahasiaan data [1].

Salah satu metode yang sering digunakan untuk menjaga keamanan kata sandi adalah dengan menerapkan algoritme *hashing*. *Secure Hash Algorithm* (SHA) merupakan fungsi *Hash* yang bersifat "tidak dapat diubah kembali" menjadi pesan semula (satu arah) yang akan menghasilkan sebuah *checksum* atau *fingerprint* dari data tersebut [2]. *Algoritme*

hashing berfungsi untuk mengubah kata sandi menjadi serangkaian kode yang unik dan sulit dipecahkan, sehingga meskipun data yang tersimpan berhasil diakses oleh pihak yang tidak berwenang, kata sandi asli tidak dapat diungkap dengan mudah. Algoritma *hash* adalah metode enkripsi yang mengubah teks menjadi serangkaian karakter acak dengan panjang yang tetap, dan merupakan enkripsi satu arah sehingga hasilnya tidak dapat dikembalikan ke bentuk aslinya. Salah satu algoritma *hashing* yang umum digunakan adalah *Secure Hash Algorithm* 1 (SHA-1), yang berfungsi untuk memastikan integritas data[3]. SHA-512 adalah salah satu fungsi *hash* yang dikembangkan dari algoritme SHA-1, yang memiliki kemampuan untuk mengokonversi *input string* dengan panjang variatif menjadi *output string* dengan panjang tetap, yaitu 512 bit.

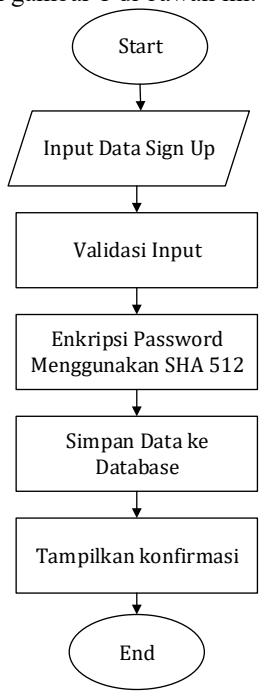
Algoritma ini memiliki beberapa karakteristik utama, antara lain: 1) Sifat *requirement*, yang memungkinkan *hash* untuk dihitung dengan mudah dari *input*, 2) Sifat *oneway function*, di mana meskipun mudah untuk menghasilkan *hash*, sulit untuk mengembalikannya ke input asli, dan 3) Sifat *collision free*, yang menjamin tidak ada dua *input* berbeda yang dapat menghasilkan nilai *hash* yang sama, sehingga melindungi dari risiko pemalsuan data [4]. Salah satu mekanisme kerja kriptografi pada algoritma SHA-512 adalah menerima *input* berupa pesan dengan ukuran variabel dan menghasilkan ringkasan pesan (*message digest*) yang memiliki panjang tetap sebesar 512 bit [5]. Algoritme ini bekerja mirip dengan SHA-256, namun menggunakan ukuran blok dan proses transformasi yang lebih besar. SHA-512 memproses data dalam blok berukuran 1024-bit, dua kali lebih besar dibandingkan dengan SHA-256. Ukuran ini meningkatkan tingkat keamanan dan membuatnya lebih tahan terhadap serangan kriptografis yang lebih canggih. Hasil *hash* dari SHA-512 ditampilkan sebagai rangkaian angka dalam format heksadesimal [6].

Penelitian sebelumnya mengenai pengamanan proses *login web* yang menggunakan SHA-512 berjudul *"Analysis of Secure Hash Algorithm (SHA) 512 for Encryption Process on Web Based Application."* Penelitian ini membahas kelemahan algoritme MD5 yang digunakan untuk enkripsi kata sandi, yang rentan terhadap serangan *Collision Attack*. Untuk mengatasi hal ini, mereka mengusulkan penggunaan algoritme SHA-512, yang lebih handal dan tahan terhadap serangan kriptografi. Hasilnya adalah peningkatan keamanan *login* melalui penggunaan SHA-512, terbukti lebih tahan terhadap *brute force* dan mendapatkan penerimaan pengguna sebesar 86% melalui UAT [7]. Namun, penelitian ini hanya berfokus pada enkripsi *login*, tanpa membahas fitur keamanan lainnya. Penelitian kali ini akan mengembangkan lebih lanjut dengan menambahkan fitur keamanan yang lebih komprehensif.

Flask adalah sebuah kerangka kerja aplikasi web [8]. *Flask* merupakan sebuah *microframework* yang dikembangkan menggunakan bahasa pemrograman *Python*, dirancang untuk memudahkan pengembang dalam membangun aplikasi web yang terstruktur dan mudah dikelola [9]. *Framework* ini digunakan untuk mempercepat pengembangan aplikasi, karena *Flask* telah menyediakan berbagai pustaka dan kumpulan kode siap pakai, sehingga memungkinkan pembuatan aplikasi web tanpa harus memulai dari nol [10]. *Flask* memiliki tiga dependensi utama: subsistem yang disediakan oleh *Werkzeug*, dukungan template yang menggunakan *Jinja2*, dan integrasi *command-line* yang didukung oleh *Click* [11]. Dalam pengembangan keamanan halaman *sign up* pada Sistem Informasi Bimbel KelasSeru berbasis *Flask*, penerapan algoritma SHA-512 dapat memberikan jaminan keamanan tambahan, khususnya dalam menjaga kerahasiaan data pengguna di server. Dengan memadukan *Flask* sebagai *framework* dan SHA-512 sebagai metode *hashing*, sistem pendaftaran yang dibangun diharapkan mampu menghadirkan tingkat keamanan yang memadai terhadap ancaman siber yang terus berkembang. Penelitian ini bertujuan untuk menerapkan algoritma *hashing* SHA-512 pada halaman *sign up* Sistem Informasi Bimbel berbasis *Flask* serta mengevaluasi tingkat keamanan yang dihasilkannya.

2. RESEARCH METHODS

Adapun *flowchart* sistem yang digunakan dalam penelitian ini dapat dilihat dari gambar 1 di bawah ini.



Gambar 1. *Flowchart* Penelitian

Berikut adalah penjelasan dari *flowchart* di atas :

1. Start

Proses *sign-up* ini dimulai ketika pengguna mengakses halaman pendaftaran pada website yang dibuat menggunakan *microframwork Flask*.

2. Input Data Sign Up

Pada halaman input data *sign up* akun, pengguna memasukkan informasi yang diperlukan untuk pendaftaran akun seperti yang diminta pada halaman web, yaitu nama pengguna, kata sandi, dan konfirmasi kata sandi.

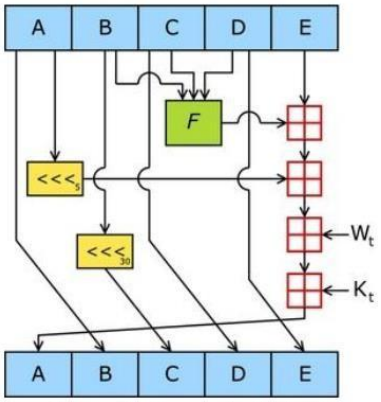
3. Validasi *Input*

Setelah data dimasukkan, sistem melakukan validasi untuk memastikan bahwa semua kolom yang wajib diisi telah lengkap dan tidak ada kolom yang kosong serta memastikan bahwa format data yang diberikan sesuai dengan aturan yang diminta.

4. Enkripsi *Password* Menggunakan SHA-512

Setelah validasi dilakukan, kata sandi pengguna akan dienkripsi menggunakan algoritma *hashing* SHA-512. Proses enkripsi SHA 512 dilakukan dengan mengubah kata sandi menjadi *string* acak dengan panjang tetap yang tidak dapat dikembalikan ke bentuk aslinya. Tujuannya adalah untuk meningkatkan keamanan data pengguna, terutama terhadap potensi serangan yang mencoba mencuri kata sandi.

Fungsi *hash* SHA 512 adalah fungsi yang menghasilkan ukuran 512-bit dan panjang blok 1024 bit yang digali pesan. Cara kerja algoritma kriptografi SHA 512 adalah menerima input dalam bentuk pesan dengan panjang atau ukuran berapapun dan akan menghasilkan intisari pesan yang memiliki panjang tetap 512 bit [12]. Untuk lebih jelas perhatikan gambar 2 berikut.



Gambar 2. Ilustrasi Kerja / Pembuatan Pesan Intisari SHA-512

Cara kerja pembuatan *message digest* dengan algoritma SHA-512 adalah sebagai berikut:

1. Penambahan Bit

Proses pertama adalah menambahkan pesan dengan sejumlah bit tambahan sehingga panjang pesan (dalam bit) menjadi kongruen dengan 890 mod 1024. Hal yang perlu diingat adalah angka 1024 muncul karena algoritma SHA-512 memproses pesan dalam blok berukuran 1024 bit. Jika ada pesan dengan panjang 24 bit, pesan tersebut masih akan ditambahkan dengan bit tambahan. Pesan tersebut akan ditambahkan dengan 896 - (24 + 1) = 871 bit. Jadi, panjang bit tambahan berada di antara 1 hingga 896. Selain itu, perlu diperhatikan bahwa bit tambahan terdiri dari satu bit bernilai 1 diikuti dengan sisa bit bernilai 0.

2. Penambahan Nilai Panjang Pesan

Proses selanjutnya adalah menambahkan pesan kembali dengan 128bit yang menyatakan panjang pesan asli. Jika panjang pesan lebih besar dari 2^128, maka panjang pesan tersebut diambil dalam modulo 2^128. Dengan kata lain, jika awalnya panjang pesan sama dengan K bit, maka 128bit ditambahkan dengan K modulo 2^128. Jadi, setelah proses kedua ini dilakukan, panjang pesan sekarang menjadi 1024 bit.

3. Inisialisasi Nilai *Hash*

Dalam algoritma SHA-512, nilai *hash* H(0) terdiri dari 8 kata dengan 64 bit dalam notasi heksadesimal seperti yang ditunjukkan pada tabel 1.

Tabel 1. Notasi Heksadesimal SHA 512

Buffer	**Nilai awal**
A	6a09e667f3bcc908
B	bb67ae8584caa73b
C	3c6ef372fe94f82b
D	a54ff53a5f1d36f1
E	510e527fade682d1
F	9b05688c2b3e6c1f
G	1f83d9abfb41bd6b
H	5be0cd19137e2179

5. **Simpan Data ke Database**

Kata sandi yang telah dienkripsikan akan disimpan di dalam database MySQL, termasuk data pengguna Setelah kata sandi. Pada tahap ini, hanya hasil enkripsi yang disimpan, bukan kata sandi asli.

6. **Tampilkan Konfirmasi**

Setelah data berhasil disimpan, sistem akan menampilkan konfirmasi kepada pengguna bahwa pendaftaran telah berhasil.

7. **End**

Proses *sign-up* diselesaikan dan pengguna dapat melanjutkan untuk menggunakan fitur lainnya dalam laman web.

Secure Hashing Algorithm – 512 (SHA-512) juga bisa di implementasikan di bidang *Computer Vision* yang bisa diterapkan untuk *Face Recognition*[13], SHA-512 juga bisa digunakan untuk melindungi keamanan data siswa pada Platform pembelajaran[14], Penelitian Terkini, Algoritma SHA-512 diintegrasikan dengan pengenalan sidik jari yang di ekstraksi menjadi fitur yang unik, dan data dimasukkan kedalam algoritma SHA-512 sehingga meningkatkan efisiensi berbasis sidik jari sekaligus meningkatkan keamanan[15].

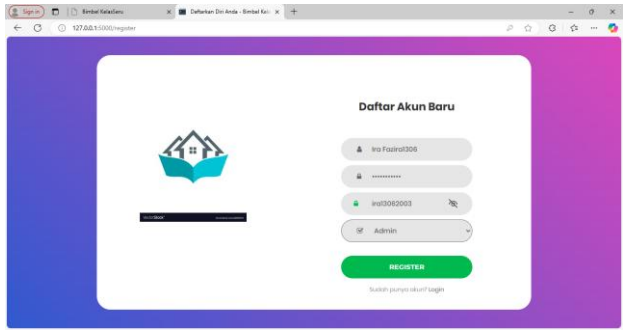
3. RESULT AND DISCUSSION

Cara menyusun Penerapan algoritma SHA-512 dilakukan menggunakan bahasa pemrograman *python* yang memanfaatkan *microframwork flask* untuk mengenkripsikan password yang didaftarkan pada saat *sign up* akun. Algoritma *hashing* SHA-512 menghasilkan *output hash* sepanjang 512-bit yang sangat sulit dipecahkan. Dalam implementasi ini, pengguna memasukkan kata sandi saat melakukan registrasi, dan kata sandi tersebut dienkripsi sebelum disimpan ke basis data.

Pada tampilan dan interaksi web, digunakan bahasa pemrograman HTML dan CSS untuk desain tampilan halaman web. Backend dari sistem yang dibangun menggunakan Flask, Setiap halaman web dihubungkan melalui *routing*, yang memastikan bahwa setiap URL diarahkan ke fungsi yang sesuai.

Dalam penelitian ini enkripsi SHA-512 ditempatkan di *route /login* untuk memverifikasi kata sandi yang di input pengguna. Sebelum diarahkan ke halaman pendaftaran akun, pengguna akan diarahkan ke laman beranda sistem informasi Bimbel terlebih dahulu. Berikut ini adalah tampilan dari laman beranda sistem informasi bimbel KelasSeru.

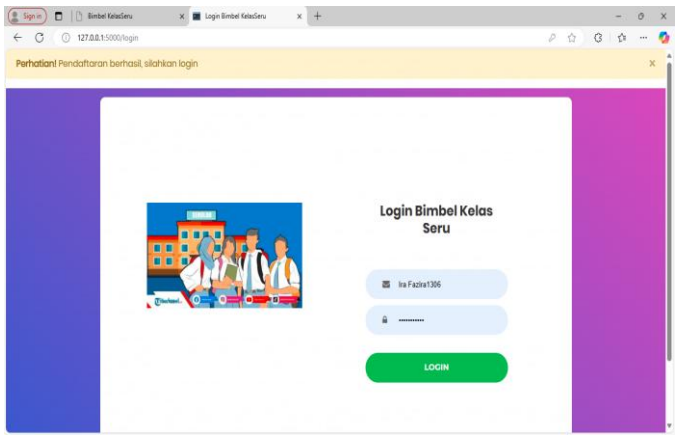
Melalui halaman beranda, pengguna bisa langsung melakukan pendaftaran akun. Sistem mengambil *hash* kata sandi dari basis data, kemudian membandingkannya dengan *hash* dari kata sandi yang dimasukkan saat login. Jika *hash* cocok, pengguna akan berhasil login, dan sesi login akan dimulai. Jika tidak cocok, sistem akan menolak akses. Berikut ini adalah tampilan awal awal dari laman login.



Gambar 3. Halaman Registrasi

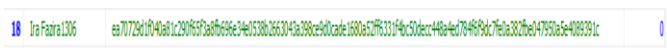
Proses enkripsi kata sandi dalam sistem terjadi saat pengguna melakukan pendaftaran akun baru yang dilakukan pada form registrasi dengan mengisi informasi seperti *username*, kata sandi, dan konfirmasi kata sandi serta pendaftaran akun sebagai admin atau peserta. Setelah data diinput, sistem melakukan validasi untuk memastikan bahwa semua kolom telah terisi dengan benar dan sesuai format yang diminta. Hal ini bertujuan untuk mencegah kesalahan input dan menjaga integritas data. Jika semua data valid, sistem kemudian melanjutkan ke proses enkripsi kata sandi menggunakan algoritma SHA-512. Algoritma ini mengubah kata sandi pengguna menjadi deretan karakter unik sepanjang 512-bit yang tidak dapat dibalik kembali ke bentuk aslinya.

Hashing menggunakan SHA-512 memberikan tingkat keamanan yang tinggi karena sulit untuk diretas atau direkayasa balik, sehingga kata sandi asli pengguna terlindungi meskipun database terancam. Setelah kata sandi di enkripsi, data pengguna seperti *username* dan *hash* dari kata sandi tersebut disimpan ke dalam basis data dengan aman serta peserta yang login sebagai admin akan disimpan dalam *database* sebagai 0 yang menandakan dia admin dan 1 untuk pengguna yang mendaftarkan akun sebagai peserta. Pada tahap ini, pengguna akan menerima konfirmasi bahwa pendaftaran akun telah berhasil, memberikan sinyal kepada pengguna bahwa proses registrasi telah selesai dengan baik. Sistem kemudian dapat melanjutkan ke tahap login, di mana enkripsi ini kembali digunakan untuk memverifikasi kata sandi yang dimasukkan pengguna dengan yang tersimpan dalam bentuk *hash* di database. Setelah proses pendaftaran akun baru menampilkan pesan konfirmasi, Berikut ini adalah halaman login yang bisa diakses setelah proses pendaftaran selesai.



Gambar 4. Halaman Login

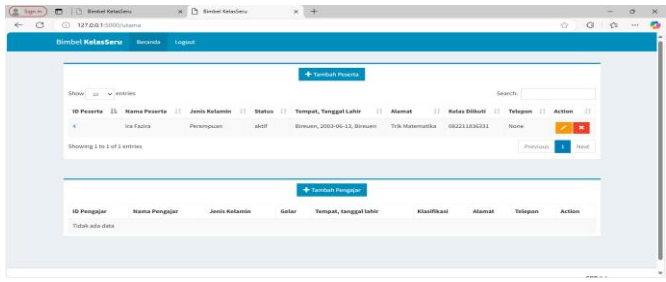
Setelah proses *hashing* selesai, nilai *hash* hasil SHA-512 (512-bit atau 128 karakter dalam format heksadesimal) akan disimpan di dalam database. Dengan penerapan SHA-512, database tidak lagi menyimpan kata sandi dalam bentuk *plaintext*, melainkan dalam bentuk *ciphertext* yang telah dihasilkan melalui proses hashing dengan algoritma SHA-512. Berikut ini kata sandi yang disimpan dalam database.



Gambar 5. Hasil Kata Sandi Tersimpan

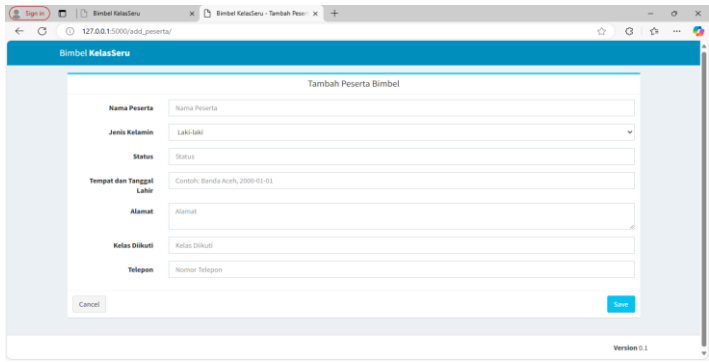
Pengguna dapat melakukan *sign-up* kembali dengan menggunakan *username* dan *password* yang sudah didaftarkan sebelumnya. Pengguna diarahkan langsung ke halaman *home* dari *website* yang menampilkan langsung biodata mahasiswa.

Berikut ini adalah halaman *home* yang ditampilkan setelah dilakukan *sign-up* oleh admin.



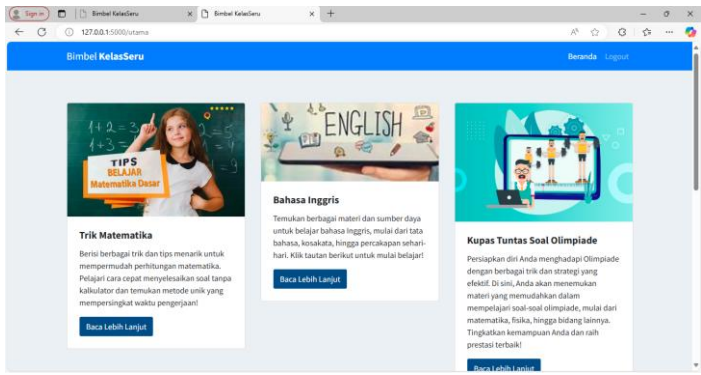
Gambar 6. Halaman Utama

Setelah melakukan *sign-up* dan pengguna diarahkan ke halaman utama, pada halaman ini admin dapat menambahkan peserta baru dari kelas bimbel ataupun menambahkan pengajar. Berikut ini adalah *form* tambah data peserta.



Gambar 7. Halaman Tambah Peserta

Admin dapat menambahkan data peserta maupun data pengajar baru dan dapat melihat hasil tambah data di halaman utama sistem infromasi. Website akan menampilkan tampilan yang berbeda apabila yang melakukan proses login adalah peserta dan bukan admin. Berikut ini adalah tampilan yang ada pada halaman utama jika melakukan login sebagai peserta.



Gambar 8. Halaman Peserta

Pada halaman peserta tersebut peserta dapat melihat berbagai materi dari bimbel KelasSeru dengan cara klik baca lebih lanjut dan peserta akan di arahkan langsung ke tampilan teori dari berbagai kelas yang dipilih.

4. CONCLUSION

Penelitian ini berhasil mengimplementasikan algoritme SHA-512 pada halaman *sign-up* berbasis Flask, yang terbukti mampu memberikan perlindungan yang signifikan terhadap ancaman siber. SHA-512 menawarkan tingkat keamanan yang

lebih tinggi dibandingkan algoritme *hashing* lainnya dengan menghasilkan *hash* yang sulit dipecahkan, bahkan dalam situasi database yang diretas. Dengan demikian, sistem ini memastikan kerahasiaan data pengguna terjaga. Selain keamanan, penelitian ini juga menambahkan fitur tambahan dan tampilan halaman website yang berbeda untuk login peserta dan admin. Penerapan lebih lanjut dapat menggabungkan mekanisme keamanan tambahan untuk menghadapi serangan siber yang lebih kompleks di masa mendatang.

REFERENCES

[1] M. A. Ramadhan, Arpinda, D. Saputra, and D. I. Mulyana, "Pencegahan Serangan Berbasis Kata Sandi: Studi Komprehensif Tentang Implementasi Hash Pada Aplikasi Web," vol. 7, no. 3, pp. 920–925, 2024.

[2] R. Pamungkas and F. W. Z. Zaney, "Penerapan Hashing SHA1 dan Algoritma Asimetris RSA untuk Keamanan Data pada Sistem Informasi berbasis Web," *Res. J. Comput. Inf. Syst. Technol. Manag.*, vol. 4, no. 1, p. 84, 2021, doi: 10.25273/research.v4i1.9099.

[3] A. I. Irawan, I. H. Santoso, and M. Rahayu, "Implementasi Sistem Keamanan Presensi Berbasis Kode QR Menggunakan Algoritma RSA dan Hash," vol. 13, pp. 53–59, 2024.

[4] R. Rizki and S. Mulyati, "Implementasi One Time Password Menggunakan Algoritma SHA-512 Pada Aplikasi Penagihan Hutang PT. XHT," *Edumatic J. Pendidik. Inform.*, vol. 4, no. 1, pp. 111–120, 2020, doi: 10.29408/edumatic.v4i1.2158.

[5] D. P. Purba, "Analisa Dan Perbandingan Algoritma Whirpool Dan Sha-512 Dalam Penyandian Data Gambar," *Bull. Artif. Intell.*, vol. 1, no. 1, pp. 8–12, 2022, doi: 10.62866/buai.v1i1.2.

[6] M. A. Fadhillah, L. Mulyarahim, and K. Nadira, "Algoritme Hashing Sha-512 Pada Sistem Halaman Sign Up Java," *TRIPLE A J. Pendidik. Teknol. Inf.*, vol. 2, no. 1, pp. 27–34, 2023.

[7] M. Sumagita and I. Riadi, "Analysis of Secure Hash Algorithm (SHA) 512 for Encryption Process on Web Based Application," *Int. J. Cyber-Security Digit. Forensics*, vol. 7, no. 4, pp. 373–381, 2018, [Online]. Available: https://www.researchgate.net/publication/327392778

[8] T. Chandra Harita, R. Kridalukmana, and D. Eridani, "Pengembangan Aplikasi Analisis Sentimen Terhadap Brand Berbasis Web Menggunakan Kerangka Kerja Flask Web-Based Sentiment Analysis Application Development Using Flask Framework," *J. Tek. Komput.*, vol. 1, no. 2, pp. 36–40, 2022, doi: 10.14710/jtk.v1i2.36307.

[9] H. L. Walingkas and P. O. N. Saian, "Penerapan Framework Flask pada Pembangunan Sistem Informasi Pemasok Barang," *J. JTIK (Jurnal Teknol. Inf. dan Komunikasi)*, vol. 7, no. 2, pp. 227–234, 2023, doi: 10.35870/jtik.v7i2.729.

[10] C. Wijayanto and Y. A. Susetyo, "Implementasi Flask Framework Pada Pembangunan Aplikasi Sistem Informasi Helpdesk (SIH)," *JIPI (Jurnal Ilm. Penelit. dan Pembelajaran Inform.*, vol. 7, no. 3, pp. 858–868, 2022, doi: 10.29100/jipi.v7i3.3161.

[11] A. C. Darmawan and L. Iswari, "Pengembangan Aplikasi Berbasis Web dengan Python Flask untuk Klasifikasi Data Menggunakan Metode Decision Tree C4.5," *J. Pendidik. dan Konseling*, vol. 4, no. 5, pp. 5351–5362, 2022.

[12] Rasyada, N. (2022). SHA-512 Algorithm on Json Web Token for Restful Web Service-Based Authentication. *Journal of Applied Data Sciences*, *3*(1), 33-43.

[13] Vankadara, A., Myneni, V., Pendyala, H., & Vadlamudi, D. (2023, April). Enhancing Encryption Mechanisms using SHA-512 for user Authentication through Password & Face Recognition. In *2023 International Conference on Inventive Computation Technologies (ICICT)* (pp. 1086-1095). IEEE.

[14] Sheketa, V., Pasieka, M., Serman, T., Pasieka, N., Chupakhina, S., & Krul, L. (2021, September). System Analysis and Example of Using SHA-512 Hash Functions to Protect Students' Personal Data on Educational Platforms. In *2021 IEEE 16th International Conference on Computer Sciences and Information Technologies (CSIT)* (Vol. 2, pp. 132-136). IEEE.

[15] Gupta, A., Banakar, N., Kumar, C., Aryan, M., & Purushotham, U. (2024, April). Design and Implementation of an Efficient Fingerprint Authentication Algorithm using SHA-512. In *2024 Third International Conference on Distributed Computing and Electrical Circuits and Electronics (ICDCECE)* (pp. 1-7). IEEE.