Jurnal Borneo Informatika & Teknik Komputer **p-ISSN:** 2808-8190
Vol. 4, No. 1, 2024. Hlm 08—14 **e-ISSN:** 2808-6600

IMPLEMENTASI *PREVENTIVE MAINTENANCE* PADA APLIKASI BERBASIS *WEBSITE* (STUDI KASUS KOSTQITA)

Ria Suci Nurhalizah[1], Ilham Sidik Saksena[2], Raden Bagus Bambang Sumantri[3], Riska Suryani[4]

[1,3,4]Program Studi Sistem Informasi Fakultas Sains & Teknologi Universitas Harapan Bangsa, Purwokerto, Jawa Tengah, Indonesia
[2]Program Studi Teknologi Informasi Fakultas Sains & Teknologi Universitas Harapan Bangsa, Purwokerto, Jawa Tengah, Indonesia
*E-mail : [*1]riascnr02@gmail.com*

Diterima	Direvisi	Disetujui
16-05-2024	20-05-2024	01-06-2024

Abstract *- Preventive maintenance is modifying software with the aim of detecting and correcting potential errors before they actually occur. On the KostQita website, software maintenance is carried out with the Preventive Maintenance system, where modifying the system is carried out on the source code in the Login process then testing errors to be corrected, where more than one account is selected, this can cause other big problems, it is necessary to add programs Mysqli_num_rows to prevent SQL Injection and so users cannot log in if more than one account is selected. The source code used is the programming language PHP and MYSQL as a database for data storage. so the results of this study are that the login process to the KostQita application using SQL Injection cannot be used and fails to log in then the system does not have an error.*

Keywords: Information System; Preventive Maintenance; PHP; MySQL

Abstrak - *Preventive maintenance* adalah memodifikasi perangkat lunak dengan tujuan untuk mendeteksi dan memperbaiki potensi kesalahan, sebelum kesalahan-kesalahan tersebut benar-benar terjadi. Pada website KostQita dilakukan perawatan perangkat lunak dengan sistem *Preventive Maintenance*, dimana memodifikasi sistem yang dilakukan pada *source code* pada proses *login* kemudian menguji kesalahan untuk diperbaiki, dimana akun yang terseleksi lebih dari satu, ini dapat menimbulkan masalah-masalah besar lainnya, maka perlu menambahkan program *Mysqli_num_rows* untuk mencegah SQL *Injection* dan agar *user* tidak bisa *login* jika ada lebih dari satu akun yang terseleksi. *Source code* yang digunakan yaitu bahasa pemrograman PHP dan MYSQL sebagai *database* untuk penyimpanan datanya. sehingga hasil dari penelitian ini yaitu proses *login* pada aplikasi KostQita menggunakan SQL *Injection* tidak dapat digunakan dan gagal untuk *login* kemudian sistem tidak error.

Kata Kunci: Sistem Informasi; *Preventive Maintenance*; PHP; MySQL

I. PENDAHULUAN

Perkembangan teknologi dan sistem informasi di indonesia kini semakin berjalan cepat dan semakin canggih. Dengan semakin berkembangnya berbagai bidang teknologi maka perlu adanya *maintenance* atau perawatan. *Maintenance* adalah suatu kegiatan yang dilakukan berulang agar memiliki kondisi yang sama seperti dengan keadaan awal (Junaidi et al., 2019). Manfaat adanya teknologi dan sistem informasi dapat dilihat di kehidupan sehari-hari, salah satunya dalam pemeliharaan perangkat lunak suatu aplikasi (WIYONO et al., 2022). Secara teori, pemeliharaan sebuah perangkat lunak dapat dilakukan dengan banyak metode: *perfective maintenance, preventive maintenance, corrective maintenance, adaptive maintenance*, dan lain lain (Nimas

Maharani et al., 2023). Salah satu metode pemeliharaannya yaitu dengan metode *preventive maintenance*. Dimana metode *preventive maintenance* adalah Memodifikasi Perangkat Lunak yang bertujuan untuk mendeteksi dan memperbaiki potensi-potensi kesalahan, sebelum kesalahan-kesalahan tersebut benar-benar terjadi. *Preventive maintenance* adalah kegiatan perawatan yang dilakukan sebelum komponen atau sistem mengalami kerusakan dan bertujuan untuk mencegah terjadinya kegagalan fungsi (Junaidi et al., 2019). *Preventive maintenance* juga dapat dikatakan sebuah kegiatan pemeliharaan secara berkala (Subekti et al., 2014).

Pengujian perangkat lunak merupakan aspek penting dalam suatu sistem informasi agar dapat mengetahui error pada perangkat lunak tersebut. Perangkat lunak memiliki peran penting untuk menghasilkan perangkat lunak yang berkualitas dengan memiliki beberapa karakteristik diantaranya yaitu ketepatan, tidak rancu, kelengkapan, konsisten dan lain sebagainya. (Enda & Siahaan, 2018)

Dalam website KostQita perlu adanya sebuah modifikasi dan perbaikan karena aplikasi KostQita memiiki potensi terdapat kesalahan yang dapat menimbulkan kesalahan kesalahan lainnya. Salah satunya pada fitur *login* yang memiliki potensi kesalahan dimana jika *user* salah memasukan *username* dan *password* tetapi tetap dapat *login*. Maka dari itu, perlu adanya pemeliharaan perangkat lunak dengan metode *preventive maintenance* agar dapat memperbaiki berbagai kesalahan. Dengan adanya *preventive maintenance*, monitoring dan pemeliharaan perangkat lunak dapat lebih mudah sehingga website dapat berjalan dengan lancar.

II. METODE PENELITIAN

1. Analisis

Tujuan dari analisis sistem yaitu untuk pembuatan list entity, pembuatan *Entity Relationship Diagram* (ERD) dan pembuatan *Data Flow Diagram*(DFD) (Situmorang & Asbari, 2022). Untuk mencari celah terdapatnya kesalahan pada sebuah perangkat lunak maka analisis perlu dilakukan, dalam proses pemeliharaan perangkat lunak preventive ini ada tiga hal yang penulis lakukan, yaitu:

a. Menganalisis kemungkinan terjadinya kesalahan pada aplikasi KostQita.
b. Memodifikasi sistem untuk menguji kesalahan.
c. Memodifikasi dan melakukan perbaikan kesalahan yang ditemukan pada sistem.

2. Modifikasi Perangkat Lunak

Dari hasil analisis yang didapat dan telah ditemukan adanya kemungkinan kesalahan pada sistem, maka tahap selanjutnya adalah memodifikasi sistem tersebut agar dapat diuji dengan detail apakah kemungkinan kesalahan yang dianalisis sebelumnya benar-benar terjadi dan dapat menimbulkan kesalahan lainnya timbul dan menjadi lebih besar.

3. Pengujian Kesalahan

Setelah sistem dimodifikasi, selanjutnya dapat diakukan pengujian untuk melihat kesalahan yang ditimbulkan, proses pengujian ini dilakukan berdasarkan hasil dari analisis yang telah dilakukan, terkait cara teknis yang dilakukan, inputan yang dimaksud pada proses analisis, dan hasil kesalahan yang dilakukan saat analisis apakah sesuai dengan apa yang dihasilkan pada proses pengujian ini (Choiri et al., 2011).

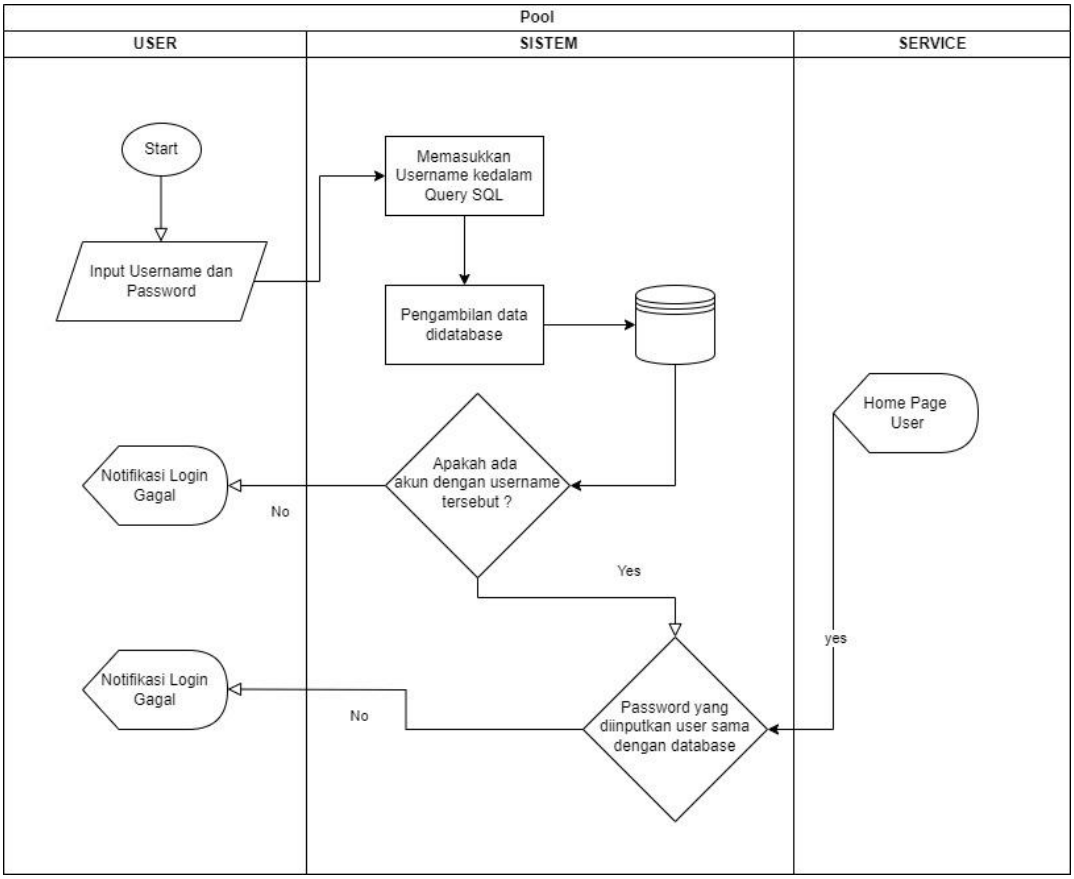
4. Perbaikan Kesalahan

Proses terakhir yang perlu dilakukan untuk melengkapi langkah pemeliharaan perangkat lunak secara *preventive* adalah dengan melakukan perbaikan, perbaikan yang dimaksud mengacu pada hasil pengujian kesalahan yang didapatkan. Perbaikan mengharuskan kesalahan yang terjadi sebelumnya teratasi sepenuhnya, sehingga kesalahan-kesalahan lain yang lebih besar tidak muncul dan menyebabkan kerugian (Rachman Khadafi, 2021).

III. HASIL DAN PEMBAHASAN

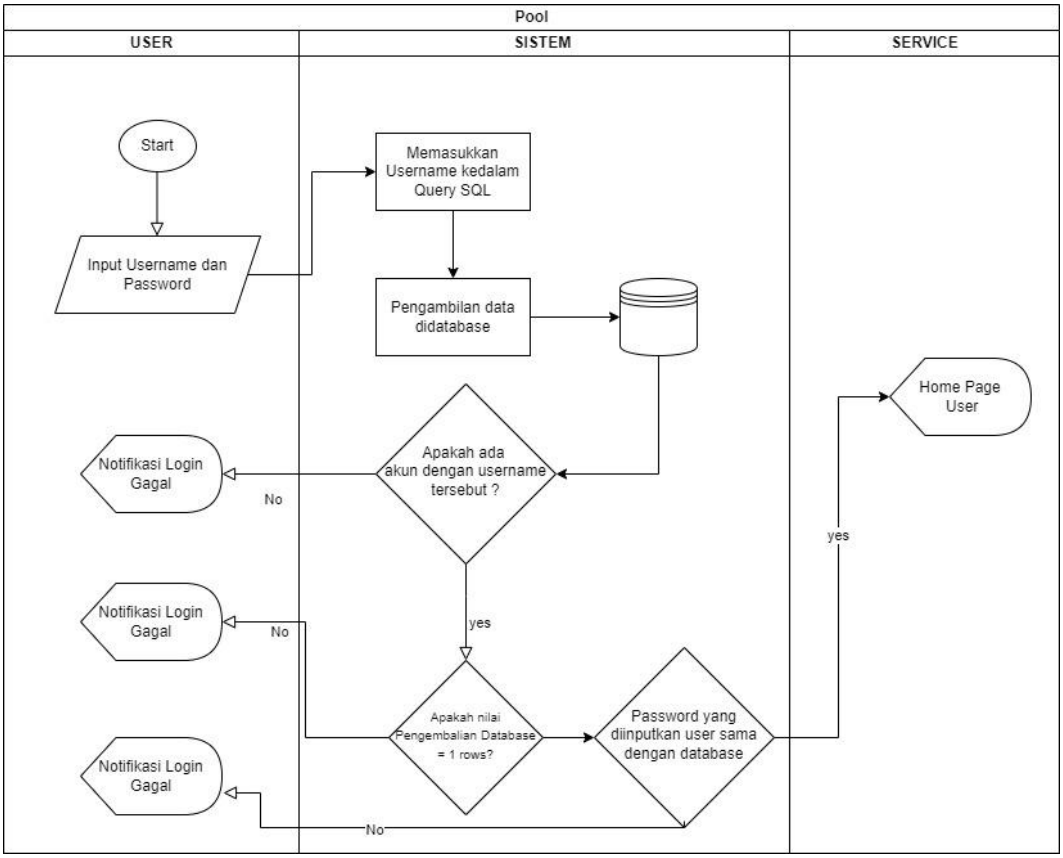
1. Analisis

Mengacu pada kejahatan *cyber security* yang marak terjadi, penulis melakukan analisis pada aplikasi website KostQita dan mengindikasi adanya celah terjadinya kesalahan pada fitur *login*. Pada fitur ini memungkinkan terjadinya kesalahan *login* jika *user* menginputkan *Query* SQL tertentu pada *username* dan *password*, Tindakan ini disebut sebagai kejahatan *cyber security* SQL *Injection*, jika user berhasil *login* menggunakan SQL *Injection* maka dapat menimbulkan kesalahan-kesalahan lain nya timbul dan bahkan terjadi kejahatan tertentu seperti pencurian data, pemalsuan data dan lain-lain, juga dapat menimbulkan kerugian yang signifikan pada pihak developer maupun pemilik asli akun tersebut. Untuk melakukan *preventive maintenance* maka yang dilakukan terlebih dahulu adalah memperbarui diagram aktifitas pada proses *login*. *Create Activity Diagram* yaitu fitur dalam pembuatan *activity* diagram dan komponennya seperti *initial* dan *final node, decision, join,* serta konektor yang meliputi generalisasi, agregasi, komposisi, dan asosiasi (Hariansyah & Saragih, 2021). Activity Diagram yang saat ini diterapkan



Gambar 1. Activity Diagram

Pada diagram aktifitas diatas dijelaskan bahwa username dan password yang di inputkan user akan diproses oleh sistem, *username* akan dimasukan ke dalam *Query* SQL *Select*, kemudian *password* yang dimasukkan akan *dienkripsi* terlebih dahulu, kemudian hasil *enkripsi* tersebut dicocokkan dengan *enkripsi password* yang ada di *database* sesuai dengan *username* yang telah terseleksi tadi, apabila *password* nya sama dengan yang ada di *database*, maka *user* dapat *login*. Jika SQL *Injection* diterapkan pada proses ini maka *user* tanpa memasukkan *username* yang benar pun akan tetap dapat *login*.



Gambar 2. Activity Diagram Baru

Diagram aktifitas yang baru menambahkan ketentuan dimana jika akun yang terpilih di *database* apabila nilai pengembalian dari *database* sama dengan satu baris maka *user* diperbolehkan *login*, jika akun yang terpilih dari *database* adalah nol atau lebih dari satu maka dapat dipastikan *user* tersebut menggunakan SQL *Injection* saat *login*, maka *user* tidak diperbolehkan *login*.

2. Modifikasi Perangkat Lunak

Perangkat lunak perlu dimodifikasi terlebih dahulu agar dapat melakukan pengujian kesalahan, modifikasi dilakukan pada *source code* diproses *Login*, *Source Code* ini menggunakan Bahasa pemrograman PHP dan MYSQL

```php
$uname = $_POST['uname'];
$pass = $_POST['pass'];
$result = mysqli_query($conn, "SELECT * FROM tb_pemilik_kost WHERE uname = '$uname'");

    $rows = mysqli_fetch_assoc($result);
    if(password_verify($pass, $rows['pass'])) {
        $_SESSION[$rows['id_pemilik']] = true;

        header('location: ibukost/ibuview.php?ibuID='.$rows['id_pemilik']);
    } else {
        header('location: masuk.php?failed=true');
    }
```

Gambar 3. Source Code Proses Login

Gambar diatas merupakan *source code* proses *login*, dimana *username* yang di inputkan *user* ditangkap dan dimasukkan ke dalam variable *$uname*, dan *password* dimasukkan ke dalam variabel *$pass*, selanjutnya *$uname* dimasukan ke *Query* SQL *Select* yang terdapat pada variabel

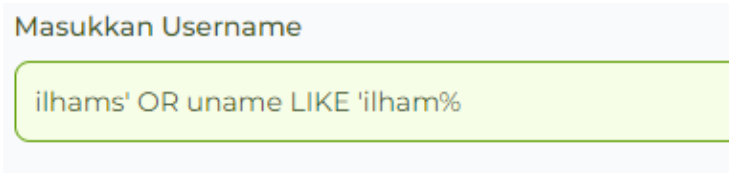
$result, kemudian jika *password* yang dimasukan sesuai dengan *password* yang ada di *database* pada akun yang telah terseleksi, maka *user* diperbolehkan untuk *login*.

```
while($rows = mysqli_fetch_assoc($result)) {
    echo "Selected = ".$rows['uname']."<br>";
}
die();
```

Gambar 4. Modifikasi Source Code

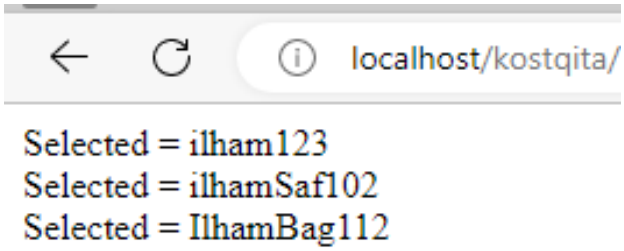
Kemudian *source code* dimodifikasi, untuk mengetahui berapa banyak akun yang terseleksi, jika *user* menggunakan SQL *Injection* maka akan ada lebih dari satu akun yang terseleksi. Setelah dilakukan modifikasi tahap selanjutnya adalah menguji program tersebut dengan SQL *Injection* untuk mengetahui kesalahan yang terjadi.

3. Pengujian Kesalahan



Gambar 5. Form Login Username

Pada form *login* di kolom *username* akan dilakukan pengujian dengan menginputkan *"ilhams' OR uname LIKE 'ilham%"* setelah itu dapat dilihat hasil dari eksekusi program adalah sebagai berikut :



Gambar 6. Hasil Eksekusi Program

Dari hasil diatas dapat disimpulkan bahwa dengan *Query* SQL pada form input dapat menimbulkan masalah, dimana akun yang terseleksi lebih dari satu, ini dapat menimbulkan masalah-masalah besar lainnya. Tahap selanjutnya setelah mengetahui kesalahan yang terjadi adalah memperbaiki program agar akun yang terseleksi hanya diperbolehkan satu, jika nol atau lebih dari satu maka *user* tidak diperbolehkan untuk *login*.

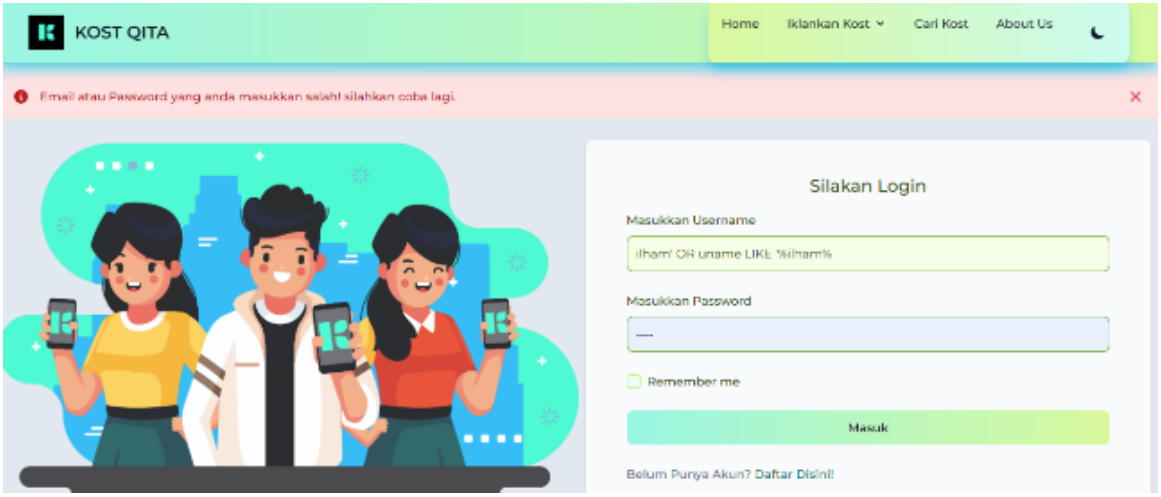
4. Perbaikan Kesalahan

Dari masalah yang dihasilkan saat pengujian, maka harus segera diatasi dengan menambahkan program untuk tidak memperbolehkan user untuk *login* jika ada lebih dari satu akun yang terseleksi.

```
$result = mysqli_query($conn, "SELECT * FROM
if (mysqli_num_rows($result) === 1 ) {
    echo "1 Akun Seected";
else {
 echo "0 atau lebih dari 1 akun ter-select";
 die();
 header('location: masuk.php?failed=true');
```

Gambar 7. Perbaikan Kesalahan

Setelah kode program ditambahkan *mysqli_num_rows* maka dapat mencegah terjadinya kesalahan "*Select Account*" meskipun *query LIKE* tetap dieksekusi, tetapi jika nilai pengambilan ya tidak sama dengan 1 baris atau akun yang terseleksi tidak sama dengan satu, maka *user* otomatis dikembalikan ke halaman *login* dan gagal melakukan *login*. Selain itu juga masih ada proses verifikasi *password* yang mengharuskan *password* benar dan sesuai dengan *password* dari *username* yang telah terseleksi. Langkah selanjutnya setelah kode program ditambahkan adalah melakukan pengujian ulang menggunakan SQL *Injection* dengan inputan *"ilhams' OR uname LIKE 'ilham%"*.



Gambar 8. Hasil Perbaikan

User akan dikembalikan ke halaman *login* dan mendapat notifikasi bahwa *login* gagal, dapat disimpulkan bahwa *preventive maintenance* ini berhasil, karena *login* pada aplikasi KostQita menggunakan *SQL Injection* tidak dapat digunakan lagi dan gagal untuk *login*.

IV. KESIMPULAN

Simpulan pada implementasi *preventive maintenance* pada aplikasi website KostQita adalah fitur *login* yang ada perlu untuk diperbaiki yaitu dengan menggunakan metode *preventive maintenance*, dimana terdapat kesalahan pada fitur *login* jika *user* menggunakan SQL *Injection* maka akan menjadikan kesalahan-kesalahan lainnya muncul dan dapat menyebabkan kerugian yang berdampak besar jika terus dibiarkan. *Preventive maintenance* dilakukan dengan memodifikasi program dan menambahkan *Query* SQL *Mysqli_num_rows* untuk mencegah SQL *Injection* tetap berjalan dan sistem tidak error. Maka dari itu, saran yang penulis dapat sampaikan untuk penelitian selanjutnya, melakukan *preventive maintenance* pada fitur *login* dengan menambahkan kode program lainnya, bukan hanya *mysqli_num_rows* tetapi juga dengan pencegahan karakter khusus seperti spasi, tanda petik, dan tanda persen pada form *login* di kolom *username*, atau dapat dilakukan dengan cara yang lain.

REFERENSI

Choiri, M., Santoso, P. B., & Rahman, A. (2015, April). Rancang Bangun Software Sistim Informasi Preventive Maintenance Untuk Industri Kecil Menengah. In *Proceeding Seminar Nasional Teknik Industri & Kongres BKSTI VI, Medan, diperoleh dari http://arifindustri. lecture. ub. ac. id/files/2014/01/Research_007. pdf, diakses tanggal* (Vol. 5).

Enda, D., & Siahaan, D. (2018). Rekomendasi Perbaikan Pernyataan Kebutuhan yang Rancu dalam Spesifikasi Kebutuhan Perangkat Lunak Menggunakan Teknik Berbasis Aturan. *Jurnal Teknologi Informasi Dan Ilmu Komputer*, *5*(2), 207–216. https://doi.org/10.25126/jtiik.201852627

Hariansyah, & Saragih, S. P. (2021). Rancang Bangun Sistem Informasi Preventive Maintenance Berbasis Web Pada Perusahaan Manufaktur. *Jurnal Comasie*, *04*(04), 107–116.

Junaidi, A., Gunawan, I., Rizal, S., & Teknik Mesin Politeknik Negeri Sriwijaya Jl Srijaya Negara Bukit Besar Palembang, J. (2019). Pembuatan Sistim Preventive Maintenance Pada Bengkel Produksi Politeknik Negeri Sriwijaya Berbasis Aplikasi. *Jurnal Austenit*, *11*(1), 16–20. https://jurnal.polsri.ac.id/index.php/austenit/article/view/1801/865

Nimas Maharani, C., Darwis, D., Penulis, N., Dedi, K. :, & Submitted, D. (2023). Analisis Perbandingan Kualitas Perangkat Lunak Pada Website Perguruan Tinggi Menggunakan Metode Webqual, Apache J-Meter, Dan Web Server Stress Tool. *Jurnal Teknologi Dan Sistem Informasi*, *4*(1), 34–41.

Rachman Khadafi, W. (2021). Rancang Bangin Aplikasi Check Sheet Preventive Maintenance Plant BCHI Menggunakan Progressive Web Application. *Jurnal Instrumentasi Dan Teknologi Informatika (JITI)*, *2*(2), 2746–7635.

Situmorang, H., & Asbari, M. (2022). Design of Web-Based Information System for Preventive Maintenance on Forklifts with Fuzzy Logic Method at PT Henkel Footwear Speciality and Adhesives. *UJoST-Universal Journal of Science and Technology*, *1*(1), 28–35.

Subekti, M., Lukman, Indrawan, donny, & Putra, G. (2014). *2199-Article Text-6243-1-10-20170425*. *5*, 625–635.

Wiyono, N., Riyanto, ., & Rejeki, A. S. (2022). Perancangan Sistem Informasi Preventive Maintenance Berbasis Web Pada Pt Macroprima Panganutama. *Insan Pembangunan Sistem Informasi Dan Komputer (IPSIKOM)*, *9*(2), 93–101. https://doi.org/10.58217/ipsikom.v9i2.206