TUGAS BESAR

SISTEM GAJI KARYAWAN DENGAN SIMULASI PROYEK DAN JAM KERJA DINAMIS DENGAN BAHASA *JAVA* PADA *VISUAL STUDIO CODE*

Nama Anggota kelompok 5:

1. Alicia Khairunnisya	(G1A025025)
2. Zora Ghaziyah Wanie	(G1A025027)
3. Andrian Revaldo	(G1A025029)

Nama Asisten Dosen :

1. Davi Sulaiman	(G1A022001)
2. Julia Mayang Sari	(G1A022010)
3. Evelyn Unike Aritonang	(G1A022024)
4. Reksi Hendra	(G1A022032)
5. Ade Irawan	(G1A022083)
6. Azzahra Faranisa	(G1A023010)
7. Lio Kusnata	(G1A023013)
8. Ahsan Nurfauzi	(G1A024002)
9. Gusti Revalion Putrada	(G1A024081)
10. Lantera Meusanah	

Dosen Pengampu :

1. Ir. Arie Vatresia, S.T., M.TI, Ph.D., IPP
2. Funny Farady Coastera, S.Kom, M.T.

PROGRAM STUDI INFORMATIKA

FAKULTAS TEKNIK

UNIVERSITAS BENGKULU

2025

KATA PENGANTAR

Puji dan syukur penulis panjatkan ke hadirat Tuhan Yang Maha Esa karena atas rahmat dan karunia-Nya, penulis dapat menyelesaikan laporan yang berjudul "SISTEM GAJI KARYAWAN DENGAN SIMULASI PROYEK DAN JAM KERJA DINAMIS DENGAN BAHASA JAVA PADA VISUAL CODE" tepat pada waktunya.

Laporan ini disusun sebagai salah satu bentuk penerapan materi pembelajaran dalam mata kuliah *Pemrograman*, khususnya dalam pemanfaatan bahasa pemrograman *Java* untuk membangun sistem berbasis logika perhitungan dan simulasi proyek. Penulis berharap laporan ini dapat memberikan gambaran mengenai bagaimana konsep dasar pemrograman, struktur data, serta logika algoritmik dapat diimplementasikan secara nyata dalam sistem penggajian yang bersifat dinamis dan berbasis proyek.

Ucapan terima kasih yang sebesar-besarnya penulis sampaikan kepada Abang Davi dan A bang Ade selaku asisten dosen yang telah banyak memberikan bimbingan, arahan, serta dukungan selama proses penyusunan laporan ini. Ucapan terima kasih juga penulis sampaikan kepada Ibu Arie dan Bapak Funny selaku dosen pengampu mata kuliah Pemrograman yang telah memberikan ilmu serta kesempatan bagi penulis untuk memahami mata kuliah ini.

Penulis menyadari bahwa laporan ini masih jauh dari sempurna. Oleh karena itu, penulis mengharapkan kritik dan saran yang membangun agar dapat menjadi bahan perbaikan di masa mendatang. Semoga laporan ini dapat memberikan manfaat bagi pembaca.

Bengkulu, 29 Oktober

Kelompok 5

DAFTAR ISI

DAFTAR GAMBAR

BAB I
PENDAHULUAN

A. LATAR BELAKANG

Sistem penggajian karyawan merupakan salah satu hal penting dalam pengelolaan sumber daya manusia di perusahaan. Penggajian berfungsi bukan hanya sebagai bentuk kompensasi finansial atas kinerja karyawan, tetapi juga sebagai sarana untuk menjaga kesejahteraan, loyalitas, serta meningkatkan motivasi kerja. Sistem penggajian yang disusun dengan prinsip akuntansi dan pengendalian internal yang baik dapat membantu organisasi dalam menjaga keakuratan data keuangan, mencegah kecurangan dan mendukung pengambilan keputusan yang lebih tepat (Heribertus Ary Setyadi, 2022).

Dalam perkembangan dunia kerja modern, sistem penggajian telah mengalami transformasi dari cara manual menuju sistem digital berbasis teknologi informasi. Penggunaan sistem informasi akuntansi penggajian membantu perusahaan untuk mengelola data karyawan, menghitung gaji, bonus, dan potongan dengan lebih cepat dan akurat. Penerapan sistem otomatis seperti ini juga mampu mengurangi kesalahan manusia serta memberikan efisiensi dalam proses administrasi keuangan. Penerapan sistem penggajian berbasis teknologi untuk meningkatkan proses pencatatan dan pelaporan keuangan (Heribertus Ary Setyadi, 2022).

Selain itu, pengembangan sistem penggajian kini juga mulai mempertimbangkan faktor kinerja dan jam kerja dinamis karyawan. Dalam konteks ini, setiap karyawan dapat menerima gaji berdasarkan total jam kerja aktual pada proyek tertentu, tingkat kesulitan pekerjaan, serta bonus atas penyelesaian proyek lebih awal. Pendekatan ini tidak hanya meningkatkan produktivitas tetapi juga mendorong motivasi kerja karena karyawan merasa dihargai sesuai kontribusinya. Bonus proyek dengan perhitungan berdasarkan tingkat kesulitan memberikan insentif yang adil dan proporsional terhadap beban kerja yang diterima.

Sistem penggajian modern juga perlu mempertimbangkan aspek perpajakan secara progresif agar perhitungan gaji bersih sesuai dengan ketentuan yang berlaku. Dengan menerapkan tarif pajak berlapis berdasarkan jumlah penghasilan, perusahaan dapat memastikan bahwa setiap karyawan membayar pajak sesuai dengan pendapatannya. Integrasi logika pajak progresif ini dalam sistem digital membuat perhitungan menjadi lebih objektif dan efisien.

Dalam pengembangan sistem, penggunaan bahasa pemrograman *Java* sangat relevan karena memiliki kemampuan dalam mengelola logika numerik kompleks dan *looping*. Melalui penerapan konsep *object-oriented programming* (OOP), sistem dapat dibangun secara modular dengan class seperti Proyek, Karyawan, dan Departemen yang masing-masing memiliki atribut serta metode perhitungan sendiri (Apriyanto Ghonzales, Akwan Sunoto, 2023).

B. RUMUSAN MASALAH

Berdasarkan latar belakang yang telah diuraikan, maka rumusan masalah dalam Sistem Gaji Karyawan dengan Simulasi Proyek dan Jam Kerja Dinamis ini adalah sebagai berikut:

1. Bagaimana merancang dan menerapkan sistem penggajian karyawan menggunakan bahasa pemrograman Java yang mampu menghitung gaji secara otomatis berdasarkan total jam kerja proyek mingguan, tingkat kesulitan proyek, dan status penyelesaian proyek?
2. Bagaimana sistem dapat menerapkan perhitungan bonus sebesar 5% dari total jam kerja dikalikan rata-rata tingkat kesulitan proyek serta pajak progresif sesuai dengan besaran penghasilan karyawan?
3. Bagaimana sistem dapat menampilkan laporan gaji seluruh karyawan, menentukan karyawan paling efisien, serta menerapkan prinsip *Object-Oriented Programming* (OOP) dengan penggunaan struktur kontrol seperti *looping* dan *if–else* agar proses perhitungan berjalan dinamis, efisien, dan terstruktur?

C. TUJUAN

Tujuan dari pembuatan proyek Sistem Gaji Karyawan dengan Simulasi Proyek dan Jam Kerja Dinamis ini adalah sebagai berikut:

1. Untuk merancang dan menerapkan sistem penggajian karyawan berbasis bahasa pemrograman Java yang dapat menghitung gaji secara otomatis berdasarkan jam kerja proyek mingguan, tingkat kesulitan proyek, serta status penyelesaian proyek.
2. Untuk menerapkan perhitungan bonus sebesar 5% dari total jam kerja dikalikan rata-rata tingkat kesulitan proyek serta pajak progresif berdasarkan besaran penghasilan guna menghasilkan perhitungan yang akurat dan adil.
3. Untuk mengembangkan sistem yang mampu menampilkan laporan gaji karyawan, menentukan karyawan paling efisien, serta menerapkan prinsip *Object-Oriented Programming* (OOP) agar program lebih terstruktur dan mudah dikembangkan.

D. MANFAAT

Adapun manfaat yang diharapkan dari pengembangan sistem ini adalah sebagai berikut:

1. Bagi perusahaan, sistem ini dapat membantu mempercepat dan mempermudah proses perhitungan gaji secara otomatis, akurat, dan transparan.
2. Bagi karyawan, sistem ini memberikan perhitungan gaji yang lebih adil karena mempertimbangkan tingkat kesulitan proyek, jam kerja, dan bonus kinerja.
3. Bagi mahasiswa, proyek ini menjadi sarana penerapan langsung konsep *Object-Oriented Programming* (OOP) dalam pengembangan aplikasi berbasis Java.

BAB II
LANDASAN TEORI

Sistem penggajian merupakan rangkaian proses yang mencakup pencatatan, perhitungan, dan pelaporan kompensasi bagi karyawan. Fungsi utama sistem penggajian adalah memastikan ketepatan waktu pembayaran, akurasi perhitungan gaji, serta kepatuhan terhadap peraturan perpajakan. Peralihan dari penggajian manual ke sistem terkomputerisasi bertujuan menekan risiko human *error*, mempercepat proses administrasi, dan meningkatkan transparansi proses penggajian sehingga pengelolaan sumber daya manusia menjadi lebih profesional. Studi kasus pada koperasi menunjukkan bahwa implementasi sistem informasi penggajian yang terkomputerisasi mampu memperkuat pengendalian internal dan meningkatkan akurasi pencatatan transaksi pengupahan (Wasundhari & Pasek, 2024).

Perancangan sistem informasi penggajian umumnya mengikuti siklus pengembangan perangkat lunak (*mis*. *waterfall* atau metode iteratif) dan mengintegrasikan modul-modul penting seperti data karyawan, absensi, perhitungan upah, potongan, dan pelaporan. Banyak penelitian pengembangan aplikasi penggajian pada konteks industri lokal menggambarkan implementasi berbasis web dengan basis data terpusat untuk memudahkan akses, pemeliharaan, dan integrasi dengan modul lain seperti kehadiran dan akuntansi. Rancangan yang modular (*mis*. kelas Proyek, Karyawan, Departemen) memudahkan pengujian dan perluasan fitur, termasuk simulasi perhitungan gaji berdasarkan berbagai skenario proyek. Implementasi yang baik juga memasukkan validasi *input* dan mekanisme audit trail untuk menjamin integritas data (Tayang & Yuliawan, 2023).

Akurasi perhitungan upah sangat bergantung pada catatan jam kerja yang akurat. Sistem absensi digital modern (*mobile*/GPS/*selfie* atau perangkat biometrik) memungkinkan pelacakan jam kerja *real-time*, pengelolaan *shift*, pencatatan cuti, dan perhitungan lembur sehingga mendukung perhitungan gaji yang bersifat dinamis. Integrasi modul absensi dengan modul penggajian mengurangi perhitungan manual dan memudahkan perhitungan total jam kerja mingguan per proyek, yang menjadi dasar bagi perhitungan gaji serta bonus berbasis jam kerja. Penelitian tentang optimasi sistem absensi dan manajemen kehadiran menegaskan pentingnya integrasi ini untuk mengurangi kecurangan absensi dan meningkatkan reliabilitas data jam kerja (Ramansyah & Rosyid, 2025).

Kompensasi berbasis kinerja (*performance-based compensation*) dan pemberian bonus merupakan instrumen penting untuk meningkatkan motivasi dan produktivitas karyawan. Skema bonus yang mempertimbangkan faktor seperti total jam kerja, penyelesaian proyek lebih awal, dan tingkat kesulitan proyek memberikan insentif yang lebih adil dan responsif terhadap

kontribusi nyata karyawan. Literatur empiris lokal menunjukkan hubungan positif antara desain kompensasi yang adil dan peningkatan kinerja karyawan; oleh karena itu desain formula bonus (mis. bonus = 5% × total jam × tingkat kesulitan rata-rata) perlu didukung data kehadiran dan hasil proyek yang dapat diaudit. Hal ini juga mendorong perancangan antarmuka sistem yang transparan sehingga karyawan dapat melihat bagaimana gaji dan bonus dihitung (Desshyfa et al., 2024).

Perhitungan gaji bersih harus memperhitungkan potongan pajak penghasilan (PPh pasal 21) sesuai peraturan yang berlaku. Peraturan Pemerintah Nomor 58 Tahun 2023 memperkenalkan mekanisme tarif efektif rata-rata (TER) yang berdampak pada cara pemotongan PPh 21 bulanan. Oleh karena itu, sistem penggajian modern wajib mengimplementasikan logika pajak yang sesuai termasuk skema progresif dan ketentuan ambang agar perhitungan pemotongan menjadi akurat dan patuh hukum. Analisis lokal terhadap implementasi PP 58/2023 menegaskan perlunya pembaruan modul perpajakan dalam aplikasi penggajian agar perusahaan tidak melakukan kesalahan pemotongan yang berisiko sanksi administrasi (Andjeliartini & Setyadi, 2024).

Pengendalian internal pada sistem penggajian mencakup pemisahan tugas (*segregation of duties*), proteksi data, otorisasi pembayaran, rekonsiliasi gaji, serta audit trail. Penelitian pada koperasi yang sudah mengimplementasikan sistem informasi akuntansi penggajian menunjukkan bahwa kontrol internal yang disokong oleh sistem terkomputerisasi mengurangi kesalahan perhitungan dan mendeteksi anomali yang berpotensi sebagai kecurangan. Oleh karena itu, desain sistem harus memasukkan mekanisme log audit, hak akses berbasis peran, dan laporan rekonsiliasi gaji untuk memudahkan pemeriksaan internal serta eksternal (Wasundhari & Pasek, 2024).

Pendekatan berbasis *Object-Oriented Programming* (OOP) sangat sesuai untuk merepresentasikan entitas-entitas nyata seperti Proyek, Karyawan, dan Departemen. Setiap kelas dapat menyimpan atribut relevan (mis. nama, tingkatKesulitan, jamKerja, selesai) dan menyediakan metode perhitungan (hitungTotalJam, hitungBonus, hitungPajak, hitungGajiBersih). Pemodelan ini memudahkan simulasi skenario (mis. beberapa proyek per karyawan, variasi tingkat kesulitan, kasus bonus penyelesaian awal), serta memungkinkan penggunaan struktur data (*List*) dan algoritma iteratif untuk mengagregasi jam kerja mingguan. Rancangan yang bersih juga memudahkan pengujian unit (unit testing) dan pemeliharaan kode. Implementasi berbasis Java atau platform web modern direkomendasikan untuk skalabilitas dan interoperabilitas (Andjeliartini & Setyadi, 2024). Memberikan bonus kepada karyawan atas penyelesaian proyek dengan cepat dan atas tingkat kesulitan yang lebih tinggi merupakan

strategi kompensasi yang relevan. Model yang sering digunakan adalah menetapkan formula bonus yang terkait langsung dengan jam kerja aktual dan tingkat kesulitan tugas. Misalnya, bonus = 5% × total jam kerja × tingkat kesulitan rata-rata. Beberapa studi menunjukkan bahwa penggunaan bonus yang spesifik dan transparan dapat meningkatkan kinerja individu serta mempercepat penyelesaian tugas. Dengan demikian, sistem penggajian yang menggabungkan jam kerja proyek dan bonus berbasis kesulitan proyek berarti perusahaan menyesuaikan kompensasi (Yuyu Ruhayu, Andi Rustam, Andi Lukman , Rustan, Andi Arifwangsa Adiningrat, 2024).

Dalam lingkungan kerja modern, karyawan sering ditugaskan ke lebih dari satu proyek dalam satu minggu. Maka, sistem harus mampu menghitung total jam kerja mingguan berdasarkan semua proyek yang dikerjakan. Selain menghitung total jam, sistem dapat menambahkan modul efisiensi untuk menentukan siapa yang paling efisien misalnya menyelesaikan jam kerja lebih sedikit namun menyelesaikan proyek lebih cepat atau dengan hasil yang lebih baik. Studi manajemen SDM lokal menekankan bahwa efisiensi karyawan bukan hanya berdasar jam kerja lebih sedikit, tetapi juga hasil yang dicapai (Kukuh Yulianto & Dety Mulyanti, 2023).

Potongan pajak penghasilan karyawan adalah bagian yang tidak bisa diabaikan dalam sistem penggajian. Model pajak progresif mensyaratkan bahwa tarif potongan meningkat seiring dengan kenaikan penghasilan. Misalnya, gaji bruto di bawah Rp 5.000.000 dikenakan 10 %, antara Rp 5.000.000 hingga Rp 10.000.000 dikenakan 15 %, dan di atas Rp 10.000.000 dikenakan 20 %. Integrasi logika ini dengan modul gaji dan bonus memastikan bahwa gaji bersih yang diterima karyawan sudah memperhitungkan pajak secara benar dan tetap sesuai dengan kontribusi kerja mereka.

Kepuasan kerja karyawan dipengaruhi oleh persepsi keadilan dalam penggajian serta transparansi sistem. Ketika karyawan memahami bahwa jam kerja yang dicatat benar, bonus dihitung adil berdasarkan tingkat kesulitan, dan pajak dipotong sesuai regulasi, maka rasa keadilan dan kepercayaan terhadap organisasi meningkat.

Pengembangan sistem penggajian modern sering menggunakan bahasa pemrograman yang mendukung OOP (*Object-Oriented Programming*), seperti Java. Dengan struktur seperti kelas Proyek, Karyawan, dan Departemen, sistem dapat mengelola data secara modular dan fleksibel. Misalnya, daftar proyek disimpan dalam struktur list (*List*), dan metode seperti hitungTotalJam(), hitungBonus(), dan hitungGajiBersih() memudahkan penggunaan logika dan kondisi *if–else* untuk penanganan skenario yang berbeda. Studi rekayasa perangkat lunak yang fokus pada aplikasi kepegawaian menegaskan pentingnya desain yang modular, mudah

diperluas, dan teruji secara unit agar sistem penggajian dapat berkembang seiring kebutuhan organisasi (Eka Melani Majid & Gunawan Prayitno, 2025).

Dalam pengembangan sistem penggajian yang modern, tidak cukup hanya menghitung jam kerja atau gaji pokok saja, melainkan sistem harus mampu mengakomodasi berbagai skenario kerja dinamis yang terjadi dalam praktik bisnis modern. Misalnya, karyawan dapat ditugaskan pada lebih dari satu proyek dalam satu periode mingguan. Oleh karena itu, sistem yang dirancang harus mampu melakukan agregasi total jam kerja mingguan berdasarkan seluruh proyek yang dikerjakan, lalu mengolah data tersebut menjadi *input* bagi modul perhitungan kompensasi. Temuan empiris menunjukkan bahwa fleksibilitas jam kerja dan pola kerja yang dinamis mampu mempengaruhi produktivitas karyawan secara signifikan (Siska Ayudia Adiyanti, 2024).

Selain total jam kerja, mekanisme pengukuran efisiensi menjadi semakin penting dalam konteks kerja berbasis proyek. Efisiensi di sini bukan hanya ditandai oleh jam kerja yang sedikit, tetapi juga oleh hasil atau *output* yang dicapai dalam periode kerja tersebut. Dengan kata lain, seorang karyawan yang menyelesaikan proyek dalam waktu lebih cepat dengan hasil berkualitas dapat dianggap lebih efisien dibandingkan yang hanya menghabiskan banyak jam kerja namun dengan hasil yang biasa saja. Hal ini memperluas paradigma pengukuran kinerja di luar sekadar kuantitas jam kerja menjadi kualitas hasil kerja (Kukuh Yulianto & Dety Mulyanti, 2023).

Karena itu, sistem penggajian berbasis proyek yang baik perlu menyertakan laporan efisiensi sebagai bagian dari keluaran sistem. Laporan tersebut dapat berupa perbandingan jam kerja vs. jumlah proyek atau tingkat penyelesaian, serta metrik seperti “jumlah proyek selesai per jam kerja” atau “rasio tingkat kesulitan terhadap waktu penyelesaian”. Dengan adanya laporan seperti ini, manajemen dapat lebih mudah mengidentifikasi karyawan yang paling efisien, menentukan insentif yang sesuai, dan mengoptimalkan alokasi tugas (Tayang & Yuliawan, 2023).

Lebih jauh, pencatatan data kehadiran dan jam kerja secara otomatis menggunakan sistem absensi digital (misalnya biometrik, GPS, *self-check*) menjadi fondasi penting bagi keakuratan perhitungan jam kerja dinamis. Tanpa data jam kerja yang akurat dan *up-to-date*, modul penggajian dan bonus akan rentan terhadap kesalahan atau manipulasi. Sebuah penelitian pada institusi pendidikan menegaskan bahwa penggunaan sistem penggajian berbasis web dan *database* terpusat dapat meningkatkan efisiensi dan akurasi rekap gaji. Dengan demikian, sistem ini tidak hanya mempercepat pengolahan data, tetapi juga memperkuat kepercayaan antara karyawan dan pihak manajemen. (Randi & Yulef Dian, 2025).

BAB III

PEMBAHASAN

STUDI KASUS 3: SISTEM GAJI KARYAWAN DENGAN SIMULASI PROYEK & JAM KERJA DINAMIS

Deskripsi:

Perusahaan ingin menghitung gaji karyawan berdasarkan jam kerja proyek mingguan, bonus proyek selesai lebih awal, dan pajak progresif. Setiap proyek memiliki tingkat kesulitan (1–5) yang memengaruhi bonus.

Struktur Program:

1. **Class Proyek**
   a. Atribut: nama (String), tingkatKesulitan (int), jamKerja (int), selesai (boolean).
2. **Class Karyawan**
   a. Atribut: nama (String), List daftarProyek, gajiPerJam (double).
   b. Method:
   a. hitungTotalJam(), hitungBonus() bonus = 5% * total jam * tingkat kesulitan rata-rata.
   b. hitungPajak() pajak progresif (10% < 5jt, 15% < 10jt, 20% > 10jt).
   c. hitungGajiBersih().
3. **Class Departemen**
   a. Atribut: List tim.
   b. Method: tampilkanLaporanGaji() dan cari karyawan paling efisien.
4. **Class MainGaji**
   a. Input proyek per karyawan, lalu tampilkan gaji bersih. b. Gunakan looping, if–else, dan logika numerik kompleks.

***Printscreen Source code* 1.1 :**

Proyek.java > ...

```java
public class Proyek {
    private String nama;
    private int tingkatKesulitan;
    private int jamKerja;
    private boolean selesai;
    public Proyek(String nama, int tingkatKesulitan, int jamKerja, boolean selesai) {
        this.nama = nama;
        this.tingkatKesulitan = tingkatKesulitan;
        this.jamKerja = jamKerja;
        this.selesai = selesai;
    }
    public String getNama() {
        return nama;
    }
    public int getTingkatKesulitan() {
        return tingkatKesulitan;
    }
    public int getJamKerja() {
        return jamKerja;
    }
    public boolean isSelesai() {
        return selesai;
    }
}
```

Gambar 1 *Source Code Class* Proyek

***Source Code* 1.1 :**

```
public class Proyek {
    private String nama;
    private int tingkatKesulitan;
    private int jamKerja;
    private boolean selesai;
    public Proyek(String nama, int tingkatKesulitan, int jamKerja, boolean selesai) {
        this.nama = nama;
        this.tingkatKesulitan = tingkatKesulitan;
        this.jamKerja = jamKerja;
        this.selesai = selesai;
    }
    public String getNama() {
        return nama;
    }
    public int getTingkatKesulitan() {
        return tingkatKesulitan;
    }
    public int getJamKerja() {
        return jamKerja;
    }
    public boolean isSelesai() {
        return selesai;
    }
}
```

Pembahasan *Source Code* 1.1:

Pertama kita buat class proyek yang berfungsi sebagai menyimpan nama, Tingkat kesulitan, jam kerja, dan berapa banyaknya proyek yang selesai. Kita membuat class encapsulation untuk menyimpan parameternya public class Proyek {private String nama;private int tingkatKesulitan; private int jamKerja; private boolean selesai; class ini hanya dapat digunakan atau dipanggil dengan menggunakan getter. public Proyek(String nama, int tingkatKesulitan, int jamKerja, boolean selesai) { this.nama = nama; this.tingkatKesulitan = tingkatKesulitan; this.jamKerja = jamKerja; this.selesai = selesai; } method khusus yang dipanggil saat membuat objek baru. public String getNama() { return nama; } public int

getTingkatKesulitan() { return tingkatKesulitan; } public int getJamKerja() { return jamKerja; } public boolean isSelesai() { return selesai; } ini adalah getter, method untuk mengakses nilai variabel private.

***Printscreen Source code* 1.2:**

```
import java.util.ArrayList;
import java.util.List;
public class Karyawan {
    private String nama;
    private List<Proyek> daftarProyek;
    private double gajiPerJam;
    public Karyawan(String nama, double gajiPerJam) {
        this.nama = nama;
        this.gajiPerJam = gajiPerJam;
        this.daftarProyek = new ArrayList<>();
    }
    public void tambahProyek(Proyek p) {
        daftarProyek.add(p);
    }
    public int hitungTotalJam() {
        int total = 0;
        for (Proyek p : daftarProyek) {
            total += p.getJamKerja();
        }
        return total;
    }
     public double hitungBonus() {
        int totalKesulitan = 0;
        int proyekSelesai = 0;
        int totalProyek = daftarProyek.size();
        int totalJam = hitungTotalJam();
```

```
    public double hitungGajiKotor() {
        return hitungTotalJam() * gajiPerJam + hitungBonus();
    }
    public double hitungPajak(double gajiKotor) {
        if (gajiKotor < 5000000) {
            return 0.10 * gajiKotor;
        } else if (gajiKotor < 10000000) {
            return 0.15 * gajiKotor;
        } else {
            return 0.20 * gajiKotor;
        }
    }
    public double hitungGajiBersih() {
        double gajiKotor = hitungGajiKotor();
        double pajak = hitungPajak(gajiKotor);
        return gajiKotor - pajak;
    }
    public String getNama() {
        return nama;
    }
    public double getEfisiensi() {
        int selesai = 0;
        for (Proyek p : daftarProyek) {
            if (p.isSelesai()) selesai++;
        }
        return (double) selesai / hitungTotalJam();
    }
     public void tampilkanData() {
        System.out.println("Nama Karyawan  : " + nama);
        System.out.println("Total Jam      : " + hitungTotalJam());
        System.out.println("Bonus          : Rp" + hitungBonus());
        System.out.println("Gaji Kotor     : Rp" + hitungGajiKotor());
        System.out.println("Gaji Bersih    : Rp" + hitungGajiBersih());
        System.out.println(x:"----------------------------------------");
    }
}
```

Gambar 2 *Source Code Class* Karyawan

***Source Code* 1.2 :**

```
import java.util.ArrayList;
import java.util.List;
public class Karyawan {
    private String nama;
    private List<Proyek> daftarProyek;
    private double gajiPerJam;
    public Karyawan(String nama, double gajiPerJam) {
        this.nama = nama;
        this.gajiPerJam = gajiPerJam;
        this.daftarProyek = new ArrayList<>();
    }
    public void tambahProyek(Proyek p) {
        daftarProyek.add(p);
    }
    public int hitungTotalJam() {
        int total = 0;
        for (Proyek p : daftarProyek) {
            total += p.getJamKerja();
        }
        return total;
    }
     public double hitungBonus() {
        int totalKesulitan = 0;
        int proyekSelesai = 0;
        int totalProyek = daftarProyek.size();
        int totalJam = hitungTotalJam();
        for (Proyek p : daftarProyek) {
            totalKesulitan += p.getTingkatKesulitan();
            if (p.isSelesai()) proyekSelesai++;
        }
        if (proyekSelesai < totalProyek) {
            return 0;
        }
```

```
    double rataKesulitan = (double) totalKesulitan / totalProyek;
    double bonus = 0.05 * totalJam * rataKesulitan;
    return bonus * 10000;
}
public double hitungGajiKotor() {
    return hitungTotalJam() * gajiPerJam + hitungBonus();
}
public double hitungPajak(double gajiKotor) {
    if (gajiKotor < 5000000) {
        return 0.10 * gajiKotor;
    } else if (gajiKotor < 10000000) {
        return 0.15 * gajiKotor;
    } else {
        return 0.20 * gajiKotor;
    }
}
public double hitungGajiBersih() {
    double gajiKotor = hitungGajiKotor();
    double pajak = hitungPajak(gajiKotor);
    return gajiKotor - pajak;
}
public String getNama() {
    return nama;
}
public double getEfisiensi() {
    int selesai = 0;
    for (Proyek p : daftarProyek) {
        if (p.isSelesai()) selesai++;
    }
    return (double) selesai / hitungTotalJam();
}

 public void tampilkanData() {
    System.out.println("Nama Karyawan  : " + nama);
```

```
        System.out.println("Total Jam      : " + hitungTotalJam());
        System.out.println("Bonus          : Rp" + hitungBonus());
        System.out.println("Gaji Kotor     : Rp" + hitungGajiKotor());
        System.out.println("Gaji Bersih    : Rp" + hitungGajiBersih());
        System.out.println("-----------------------------------------");
    }
}
```

Pembahasan *Source Code* 1.2 :

Kemudian kita buat class Karyawan yang berfungsi menyimpan proyek yang dikerjakan oleh karyawan, untuk menghitung bonus, gaji kotor, dan gaji bersih. private String nama; private List<Proyek> daftarProyek; private double gajiPerJam; adalah kelas encapsulation yang berfungsi menyimpan input scanner. private List<Proyek> daftarProyek; adalah daftar yang menyimpan proyek-proyek karyawan. this.daftarProyek = new ArrayList<>(); kegunaannya adalah membuat list kosong untuk menyimpan proyek. public void tambahProyek(Proyek p) { daftarProyek.add(p); } Tujuannya adalah agar setiap objek Karyawan bisa menambahkan proyek baru ke daftar proyek yang ia kerjakan. public int hitungTotalJam() { int total = 0; for (Proyek p : daftarProyek) { total += p.getJamKerja(); } return total; } berfungsi Menjumlahkan semua jam kerja dari setiap proyek. for (Proyek p : daftarProyek) { totalKesulitan += p.getTingkatKesulitan(); if (p.isSelesai()) proyekSelesai++; } berfungsi menjumlahkan semua kesulitan proyek dan menjumlahkan berapa proyek yang selesai. if (proyekSelesai < totalProyek) { return 0; } berfungsi untuk apakah karyawan mendapatkan bonus atau tidak, jika 2 proyek selesai kedua-duanya maka mendapatkan bonus tetapi jika hanya 1 proyek yang selesai, karyawan tidak mendapatkan bonus sama sekali. double rataKesulitan = (double) totalKesulitan / totalProyek; double bonus = 0.05 * totalJam * rataKesulitan; return bonus * 10000; berfungsi menghitung bonus. public double hitungGajiKotor() { return hitungTotalJam() * gajiPerJam + hitungBonus(); } berfungsi menjumlahkan gaji kotor. public double hitungPajak(double gajiKotor) { if (gajiKotor < 5000000) { return 0.10 * gajiKotor; } else if (gajiKotor < 10000000) { return 0.15 * gajiKotor; } else { return 0.20 * gajiKotor; } } untuk menghitung pajak dengan menggunakan syarat-syarat yang telah ditentukan. public double hitungGajiBersih() { double gajiKotor = hitungGajiKotor(); double pajak = hitungPajak(gajiKotor); return gajiKotor - pajak; } untuk menghitung gaji bersih. public double getEfisiensi() { int selesai = 0; for (Proyek p : daftarProyek) { if (p.isSelesai()) selesai++; } return (double) selesai / hitungTotalJam(); } berfungsi mengukur banyak proyek selesai dibandingkan total jam kerja. Semakin besar nilainya, semakin efisiensi karyawan tersebut.

***Printscreen Source Code* 1.3 :**

Gambar 3 *Source Code Class* Departemen

***Source Code* 1.3 :**

```
import java.util.ArrayList;
import java.util.List;
public class Departemen {
   private List<Karyawan> tim;
   public Departemen() {
      tim = new ArrayList<>();
   }
   public void tambahKaryawan(Karyawan k) {
      tim.add(k);
   }
   public void tampilkanLaporanGaji() {
      System.out.println("\n=== LAPORAN GAJI KARYAWAN ===");
      for (Karyawan k : tim) {
         k.tampilkanData();
      }
```

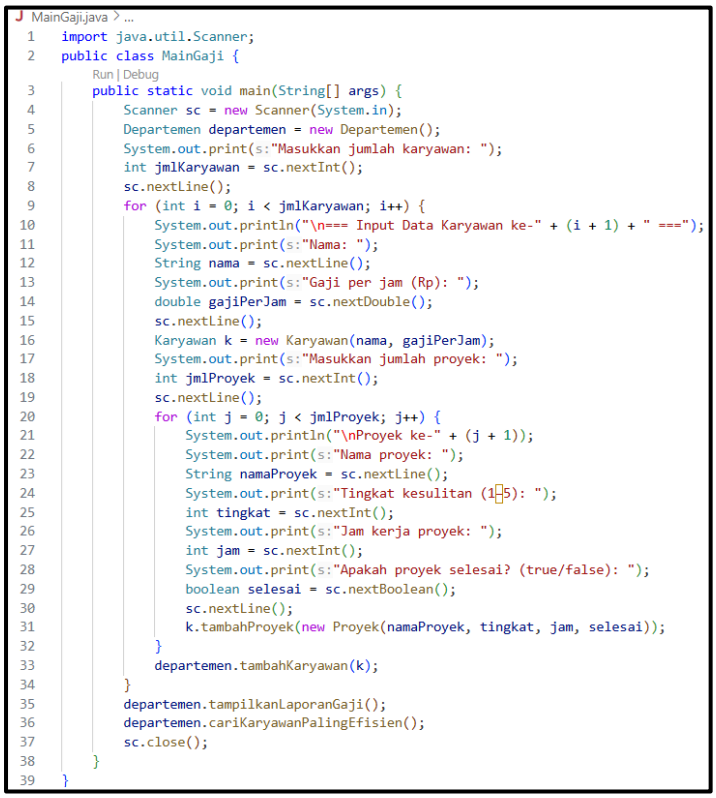
```
    }
    public void cariKaryawanPalingEfisien() {
        Karyawan efisien = null;
        double maxEf = 0;
        for (Karyawan k : tim) {
            if (k.getEfisiensi() > maxEf) {
                maxEf = k.getEfisiensi();
                efisien = k;
            }
        }

        if (efisien != null) {
            System.out.println("Karyawan paling efisien: " + efisien.getNama() +
                               " (Efisiensi: " + String.format("%.4f", maxEf) + ")");
        }
    }
}
```

Pembahasan *Source Code* 1.3 :

Kemudian kita buat class Departemen yang berfungsi menyimpan jumlah karyawan, menampilkan gaji, dan mencari efisiensi karyawan. private List<Karyawan> tim; public Departemen() { tim = new ArrayList<>(); }public void tambahKaryawan(Karyawan k) { tim.add(k); } berfungsi untuk menyimpan jumlah karyawan dan menambahkan jumlah karyawan. public void tampilkanLaporanGaji() { System.out.println("\n=== LAPORAN GAJI KARYAWAN ==="); for (Karyawan k : tim) { k.tampilkanData(); } } berfungsi untuk menampilkan data jumlah gaji karyawan, setiap karyawan akan memunculkan hasil yang beda sesuai dengan input-an scanner. public void cariKaryawanPalingEfisien() { Karyawan efisien = null; double maxEf = 0; for (Karyawan k : tim) { if (k.getEfisiensi() > maxEf) { maxEf = k.getEfisiensi(); efisien = k; } } berfungsi untuk mencari karyawan yang paling efisien dalam bekerja. if (efisien != null) { System.out.println("Karyawan paling efisien: " + efisien.getNama() + " (Efisiensi: " + String.format("%.4f", maxEf) + ")"); } Setelah perulangan selesai, program memeriksa apakah efisien tidak kosong (null). Ini memastikan bahwa setidaknya ada satu karyawan dalam daftar. Jika ada, maka akan menampilkan nama karyawan paling efisien dan nilai efisiensinya ke layar. String.format("%.4f", maxEf) menampilkan nilai efisiensi dengan 4 angka di belakang koma, agar lebih rapi.

***Printscreen Source Code* 1.4 :**




```java
import java.util.Scanner;
public class MainGaji {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        Departemen departemen = new Departemen();
        System.out.print("Masukkan jumlah karyawan: ");
        int jmlKaryawan = sc.nextInt();
        sc.nextLine();
        for (int i = 0; i < jmlKaryawan; i++) {
            System.out.println("\n=== Input Data Karyawan ke-" + (i + 1) + " ===");
            System.out.print("Nama: ");
            String nama = sc.nextLine();
            System.out.print("Gaji per jam (Rp): ");
            double gajiPerJam = sc.nextDouble();
            sc.nextLine();
            Karyawan k = new Karyawan(nama, gajiPerJam);
            System.out.print("Masukkan jumlah proyek: ");
            int jmlProyek = sc.nextInt();
            sc.nextLine();
            for (int j = 0; j < jmlProyek; j++) {
                System.out.println("\nProyek ke-" + (j + 1));
                System.out.print("Nama proyek: ");
                String namaProyek = sc.nextLine();
                System.out.print("Tingkat kesulitan (1-5): ");
                int tingkat = sc.nextInt();
                System.out.print("Jam kerja proyek: ");
                int jam = sc.nextInt();
                System.out.print("Apakah proyek selesai? (true/false): ");
                boolean selesai = sc.nextBoolean();
                sc.nextLine();
                k.tambahProyek(new Proyek(namaProyek, tingkat, jam, selesai));
            }
            departemen.tambahKaryawan(k);
        }
        departemen.tampilkanLaporanGaji();
        departemen.cariKaryawanPalingEfisien();
        sc.close();
    }
}
```

Gambar 4 *Sorce Code Class* MainGaji

***Source Code* 1.4:**

```
import java.util.Scanner;
public class MainGaji {
   public static void main(String[] args) {
      Scanner sc = new Scanner(System.in);
      Departemen departemen = new Departemen();
      System.out.print("Masukkan jumlah karyawan: ");
      int jmlKaryawan = sc.nextInt();
      sc.nextLine();
      for (int i = 0; i < jmlKaryawan; i++) {
         System.out.println("\n=== Input Data Karyawan ke-" + (i + 1) + " ===");
         System.out.print("Nama: ");
         String nama = sc.nextLine();
         System.out.print("Gaji per jam (Rp): ");
         double gajiPerJam = sc.nextDouble();
         sc.nextLine();
         Karyawan k = new Karyawan(nama, gajiPerJam);
```

```
            System.out.print("Masukkan jumlah proyek: ");
            int jmlProyek = sc.nextInt();
            sc.nextLine();
            for (int j = 0; j < jmlProyek; j++) {
                System.out.println("\nProyek ke-" + (j + 1));
                System.out.print("Nama proyek: ");
                String namaProyek = sc.nextLine();
                System.out.print("Tingkat kesulitan (1–5): ");
                int tingkat = sc.nextInt();
                System.out.print("Jam kerja proyek: ");
                int jam = sc.nextInt();
                System.out.print("Apakah proyek selesai? (true/false): ");
                boolean selesai = sc.nextBoolean();
                sc.nextLine();
                k.tambahProyek(new Proyek(namaProyek, tingkat, jam, selesai));
            }
            departemen.tambahKaryawan(k);
        }
        departemen.tampilkanLaporanGaji();
        departemen.cariKaryawanPalingEfisien();
        sc.close();
    }
}
```

Pembahasan *Source Code* 1.4 :

Yang terakhir adalah class MainGaji yang berfungsi sebagai tempat Membuat objek Departemen, Meminta input dari pengguna lewat Scanner, Memasukkan data karyawan & proyek, Menampilkan laporan, dan mencari karyawan paling efisien. Terdapat beberapa code ini int jmlKaryawan = sc.nextInt(); pada class, ini berfungsi sebagai penggunaan scanner untuk memasukkan input-an, dan String nama = sc.nextLine(); befungsi membersihkan newline setelah membaca input-an. System.out.print("Nama: "); berfungsi tempat memasukkan nama. for (int i = 0; i < jmlKaryawan; i++) .{ System.out.println("\n=== Input Data Karyawan ke-" + (i + 1) + " ==="); Ini adalah perulangan for. jmlProyek adalah jumlah proyek yang dimasukkan pengguna sebelumnya. Misal user mengetik 3, maka perulangan akan berjalan 3 kali. Variabel j digunakan sebagai nomor urut proyek (mulai dari 0 sampai jumlah proyek - 1). Jadi program

akan input proyek ke-1, ke-2, ke-3, dan seterusnya. System.out.print("Gaji per jam (Rp): "); double gajiPerJam = sc.nextDouble(); sebagai tempat untuk mmemasukkan input gaji, setelah dimasukkan inputnya akan disimpan pada class karyawan. k.tambahProyek(new Proyek(namaProyek, tingkat, jam, selesai)); Di sini berfungsi membuat objek proyek baru berdasarkan data yang sudah dimasukkan user. new Proyek(...) memanggil constructor dari kelas Proyek untuk membuat proyek. Lalu k.tambahProyek(...) akan menambahkan proyek tersebut ke daftar proyek milik karyawan k. Setelah semua proyek untuk karyawan tersebut selesai diinput menggunakan departemen.tambahKaryawan(k);,karyawan k (beserta semua proyeknya) ditambahkan ke daftar karyawan di Departemen. System.out.print("Masukkan jumlah proyek: "); tempat memasukkan input proyek yang diinginkan oleh pengguna, setelah dibaca jumlah proyek tersebut disimpan ke dalam class Karyawan. departemen.tampilkanLaporanGaji(); berfungsi untuk menampilkan hasil laporan gaji, program memanngil Kembali dari class Proyek, kemudian ke class Departemen, dan baru ke class MainGaji. System.out.print("Apakah proyek selesai? (true/false): "); boolean selesai = sc.nextBoolean(); ini tempat sebagai masukkan apakah proyeknya selesai atau tidak, setelah itu dimasukkan ke class karyawan, kemudian di cari mana karyawan yang paling efisien. departemen.cariKaryawanPalingEfisien(); untuk menampilkan karyawan paling efisien.

Printscreen Output Code 1.5 :

```
Masukkan jumlah karyawan: 2

=== Input Data Karyawan ke-1 ===
Nama: Zora
Gaji per jam (Rp): 300.000
Masukkan jumlah proyek: 2

Proyek ke-1
Nama proyek: a
Tingkat kesulitan (1-5): 3
Jam kerja proyek: 18
Apakah proyek selesai? (true/false): false

Proyek ke-2
Nama proyek: b
Tingkat kesulitan (1-5): 2
Jam kerja proyek: 12
Apakah proyek selesai? (true/false): true

=== Input Data Karyawan ke-2 ===
Nama: Andre
Gaji per jam (Rp): 300.000
Masukkan jumlah proyek: 1

Proyek ke-1
Nama proyek: x
Tingkat kesulitan (1-5): 5
Jam kerja proyek: 20
Apakah proyek selesai? (true/false): true

=== LAPORAN GAJI KARYAWAN ===
Nama Karyawan  : Zora
Total Jam      : 30
Bonus          : Rp0.0
Gaji Kotor     : Rp9000000.0
Gaji Bersih    : Rp7650000.0
-----------------------------------------
Nama Karyawan  : Andre
Total Jam      : 20
Bonus          : Rp50000.0
Gaji Kotor     : Rp6050000.0
Gaji Bersih    : Rp5142500.0
-----------------------------------------
Karyawan paling efisien: Andre (Efisiensi: 0,0500)
PS C:\Users\alici\Tubes>
```

Gambar 5 *Output Code*

Pembahasan *Output Code* 1.5 :

Program ini merupakan sistem pengelolaan gaji dan efisiensi karyawan dalam suatu departemen. Ketika program ini dijalankan dengan menggunakan terminal, dan menghasilkan output seperti ada pada gambar. Itu dikarenakan pada awalnya, pengguna memasukkan jumlah karyawan sebanyak 2, yaitu Zora dan Andre. Setelah itu, untuk setiap karyawan, program meminta input berupa nama, gaji per jam, jumlah proyek, serta data tiap proyek seperti nama proyek, tingkat kesulitan (1–5), jam kerja proyek, dan status proyek (selesai atau tidak).

Untuk Zora, terdapat dua proyek: proyek pertama memiliki tingkat kesulitan 3 dan durasi 18 jam namun belum selesai, sedangkan proyek kedua memiliki tingkat kesulitan 2 dengan durasi 12 jam dan sudah selesai. Total jam kerja Zora adalah 30 jam. Karena proyek pertama belum selesai, Zora tidak mendapat bonus (terlihat dari "Bonus: Rp0"), sehingga gaji kotornya adalah hasil dari 30 jam × Rp300.000 = Rp9.000.000. Setelah dikurangi pajak 15%, gaji bersih Zora menjadi Rp7.650.000.

Sementara itu, Andre hanya mengerjakan satu proyek dengan tingkat kesulitan 5, durasi 20 jam, dan proyek tersebut selesai. Total jam kerja Andre adalah 20 jam, dan karena proyeknya selesai serta memiliki tingkat kesulitan tinggi, Andre mendapatkan bonus Rp500.000. Gaji kotornya adalah 20 jam × Rp300.000 + bonus = Rp6.500.000, dan setelah dipotong pajak 15%, gaji bersihnya menjadi Rp5.142.500.

Pada akhir program, sistem menampilkan laporan gaji untuk masing-masing karyawan, termasuk total jam kerja, bonus, gaji kotor, dan gaji bersih. Terakhir, fungsi cariKaryawanPalingEfisien() menghitung tingkat efisiensi setiap karyawan berdasarkan perbandingan antara total proyek selesai, tingkat kesulitan, dan jam kerja. Hasilnya menunjukkan Zora memiliki efisien (0,0333) dan Andre memiliki efisiensi (0,0500), bisa dilihat bahwa Andre mempunyai efisien yang lebih tinggi dari zora, sehingga ia dinyatakan sebagai karyawan paling efisien.

BAB IV
KESIMPULAN DAN SARAN

A. KESIMPULAN

Secara kesimpulan, sistem penggajian modern berperan penting dalam memastikan proses pembayaran gaji berlangsung akurat, cepat, dan transparan. Peralihan dari metode manual ke sistem terkomputerisasi membantu mengurangi kesalahan manusia, mempercepat administrasi, dan mempermudah pengawasan internal. Dengan adanya modul-modul seperti data karyawan, absensi, perhitungan upah, bonus, dan pajak, sistem ini mampu mengelola seluruh proses penggajian secara terintegrasi.

Penerapan sistem absensi digital yang terhubung langsung dengan modul penggajian juga menjadi faktor penting dalam menjaga keakuratan data jam kerja. Dengan teknologi biometrik, GPS, atau absensi berbasis web, data kehadiran karyawan dapat tercatat secara -+dan otomatis. Integrasi ini membantu menghitung total jam kerja, bonus berbasis proyek, hingga potongan pajak progresif secara tepat. Selain itu, skema kompensasi yang mempertimbangkan tingkat kesulitan proyek dan kinerja karyawan turut meningkatkan rasa keadilan dan motivasi dalam bekerja, karena karyawan dapat melihat hubungan langsung antara hasil kerja mereka dan imbalan yang diterima.

Dari sisi pengembangan perangkat lunak, pendekatan berbasis *Object-Oriented Programming* (OOP) seperti penggunaan kelas Proyek, Karyawan, dan Departemen membuat sistem lebih fleksibel dan mudah dikembangkan. Setiap bagian sistem dapat diperbarui tanpa mengganggu keseluruhan program, sehingga cocok digunakan dalam lingkungan kerja yang dinamis dan berbasis proyek. Dengan desain yang modular dan validasi data yang kuat, sistem penggajian tidak hanya menjadi alat administrasi, tetapi juga sarana evaluasi kinerja dan efisiensi karyawan.

B. SARAN

Bagi mahasiswa, khususnya yang mempelajari bidang sistem informasi atau teknik informatika, penting untuk memahami bahwa pengembangan sistem penggajian bukan hanya soal membuat program yang bisa menghitung gaji, tetapi juga bagaimana sistem tersebut mampu mencerminkan proses bisnis secara efisien dan akurat. Mahasiswa disarankan untuk memperdalam pemahaman tentang konsep *Object-Oriented Programming* (OOP), logika perhitungan numerik, serta integrasi data antar modul seperti absensi, pajak, dan bonus. Selain itu, penting juga untuk belajar tentang prinsip keamanan data dan pengendalian internal agar sistem yang dibuat tidak hanya berfungsi dengan baik, tetapi juga aman dan dapat dipercaya dalam dunia kerja.

DAFTAR PUSTAKA

Andjeliartini, N. I., & Setyadi, B. (2024). Analisis Perubahan PPh 21 Pasca UU HPP dan PP 58/2023 untuk Wajib Pajak Pribadi. *Jurnal Neraca: Jurnal Pendidikan Dan Ilmu Ekonomi Akuntansi*, *8*(2), 149–162. https://doi.org/10.31851/neraca.v8i2.16591

Apriyanto Ghonzales, Akwan Sunoto, H. (2023). *Perancangan Sistem Informasi Penggajian Karyawan Pada PT. Tunas Utama Alumindo Jambi*.

Desshyfa, R. T., Maudri, F. R., & Wardhani, V. P. (2024). Peran Kinerja Karyawan Dalam Meningkatkan Motivasi Kerja dan Kompensasi di Lingkungan Perusahaan. *Jurnal Penelitian Ilmiah Multidisiplin*, *8*(12), 689–695.

Eka Melani Majid, & Gunawan Prayitno. (2025). Implementasi Sistem Informasi Penggajian Digital Berbasis Java-MySQL untuk Optimalisasi Administrasi Tenaga Honorer: Studi Kasus Dinas Perdagangan Kabupaten Nabire. *Jurnal Teknologi Dan Informatika*, *3*(1), 136–149. https://doi.org/10.70539/jti.v3i1.58

Heribertus Ary Setyadi, S. (2022). *Sistem Informasi Manajemen Kehadiran Dan Jam Kerja Karyawan Untuk Kelengkapan Perhitungan Gaji Karyawan*.

Kukuh Yulianto, & Dety Mulyanti. (2023). Pengaruh Jam Kerja Terhadap Jam Absensi Dan Produktivitas Karyawan: Studi Kasus Salah Satu Karyawan PT. XYZ. *Jurnal Ekonomi Manajemen Akuntansi*, *29*(1), 48–53. https://doi.org/10.59725/ema.v29i1.64

Ramansyah, T., & Rosyid, H. (2025). Optimasi Sistem Informasi Absensi Berbasis Website pada Dinas Komunikasi dan Informatika Kabupaten Lamongan. *Digital Transformation Technology*, *4*(2), 1024–1031. https://doi.org/10.47709/digitech.v4i2.5156

Randi, & Yulef Dian. (2025). Pengembangan Sistem Informasi Penggajian Karyawan Berbasis Web Untuk Meningkatkan Efisiensi Di PT Satu Nusa Lintas Persada. *JEKIN - Jurnal Teknik Informatika*, *5*(2), 551–561. https://doi.org/10.58794/jekin.v5i2.1417

Siska Ayudia Adiyanti, R. M. S. (2024). Pengaruh Fleksibilitas Jam Kerja dan Gaji terhadap Produktivitas Kerja para Pekerja Remote. *Remik: Riset Dan E-Jurnal Manajemen Informatika Komputer*, *8*, 66–72.

Tayang, N., & Yuliawan, K. (2023). Perancangan Sistem Informasi Penggajian Karyawan Menggunakan Php Dan Mysql Pada Pt Amp. *Journal of Information System Management (JOISM)*, *5*(1), 14–18. https://doi.org/10.24076/joism.2023v5i1.1126

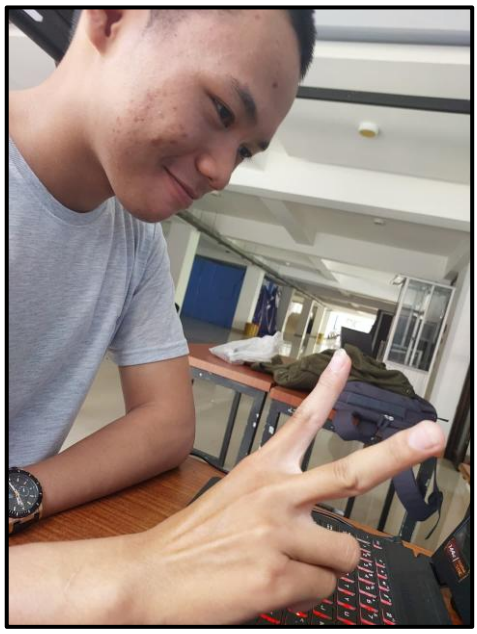
Wasundhari, P. D., & Pasek, N. S. (2024). Analisis Sistem Informasi Akuntansi Penggajian Karyawan Dalam Upaya Pengendalian Internal Pada Koperasi Simpan Pinjam Winasa Sari. *Jurnal Ilmiah Akuntansi Dan Humanika*, *14*(3), 376–384. https://doi.org/10.23887/jiah.v14i3.83846

Yuyu Ruhayu, Andi Rustam, Andi Lukman , Rustan4, Andi Arifwangsa Adiningrat, F. F. T. (2024). *Implementation of bonus scheme in improving employee work productivity in sea transportation companies*. *7*.

LAMPIRAN

Gambar 6 Dokumentasi Kerja 1

Gambar 7 Dokumentasi Kerja 2



Gambar 8 Anak sholeh, Rajin, Baik, dan Pintar

PEMBAGIAN *JOB DESCRIPTION* KELOMPOK 5

1. **Alicia Khairunnisya (Ketua Kelompok)**
    - Memastikan tiap anggota kelompok mengerjakan tugasnya dan mengingatkan ke anggota kelompok bagian tugas-tugas mereka.
    - Membagi tugas dalam pengerjaan laporan, PPT, mengatur jadwal kerja kelompok, dan memperbaiki kesalahan dalam penulisan laporan.
    - Membuat kata pengantar, daftar isi, latar belakang, landasan teori, dan daftar pustaka.
    - Mengatur format penulisan dalam laporan.
2. **Zora Ghaziyah Wanie**
    - Membuat kode tiap studi kasus, mulai dari kode Proyek, Karyawan, Departemen, dan MainGaji.
    - Membuat *source code*, penjelasan *souce code*, *print screen output*, dan penjelasan *output*.
    - Membuat daftar gambar
    - Membuat PPT presentasi.
3. **Andrian Revaldo**
    - Membuat border dan *watermark* laporan.
    - Membuat bagian rumusan masalah, tujuan, manfaat, dan lampiran.
    - Membuat halaman pada laporan.
    - Membuat PPT presentasi (Penanggung jawab).

KEMENTERIAN PENDIDIKAN TINGGI, SAINS DAN TEKNOLOGI
UNIVERSITAS BENGKULU
FAKULTAS TEKNIK
PROGRAM STUDI INFORMATIKA
Jl. Wr. Supratman Kandang Limun, Bengkulu Bengkulu 38371 A
Telp: (0736) 344087, 22105 - 227

LEMBAR ASISTENSI
TUGAS BESAR PROYEK PENGANTAR PEMROGRAMAN

Nama Anggota Kelompok	: 1. Alicia Khairunnisya	(G1A025025)
	2. Zora Ghaziyah Wanie	(G1A025027)
	3. Andrian Revaldo	(G1A025029)
Dosen	: 1. Funny Farady Coastera, S.Kom, M.T	
	2. Ir. Arie Vatresia, S.T.,M.TI, Ph.D., IPP.	
Asisten	: 1. Davi Sulaiman	(G1A022001)
	2. Julia Mayang Sari	(G1A022010)
	3. Evelyn Unike Aritonang	(G1A022024)
	4. Reksi Hendra	(G1A022032)
	5. Ade Irawan	(G1A022083)
	6. Azzahra Faranisa	(G1A023010)
	7. Lio Kusnata	(G1A023013)
	8. Ahsan Nurfauzi	(G1A024002)
	9. Gusti Revalion Putrada	(G1A024081)
	10. Lantera Meusanah	(G1A024105)

Laporan Praktikum	Catatan dan Tanda Tangan
Laporan Tugas Besar	