**PROGRAM APLIKAKASI**

**DATA KARYAWAN**



Oleh:

Nama : Melanie Anthikah Usrifan

Nim : 13020210147

Kelas : B2

Mata Kuliah : Pemrograman Berorientasi Objek

Dosen : Mardiyyah Hasnawi, S.Kom., M.T.

**PROGRAM STUDI TEKNIK INFORMATIKA**

**FAKULTAS ILMU KOMPUTER**

**UNIVERSITAS MUSLIM INDONESIA**

**MAKASSAR**

**2023**

**Daftar Isi**

BAB 1. Pendahuluan

1.1. Latar Belakang

1.2. Rumusan Masalah

1.3. Tujuan dan manfaat

BAB 2. Tinjauan Pustaka

2.1. Teori Singkat OOP

2.2. Teknik/Metode,Tools.

2.3. Solusi

BAB 3. Analilis dan Desain

3.1. Sistem Berjalan

3.2. Identifikasi

3.3. Analisis

3.4. Tahapan Pengembangan

BAB 4. Implementasi dan Pengujian

4.1. Capture

BAB 5. Penutup

5.1. Kesimpulan

Daftar Pustaka

Lampiran

Link Aplikasi dan Link Video Presentasi

**ABSTRAK**

Aplikasi “Data karyawan” adalah sebuah perangkat lunak yang dirancang untuk mengelola informasi dan data terkait karyawan dalam suatu organisasi atau perusahaan. Aplikasi ini bertujuan untuk menyediakan solusi yang efisien dalam manajemen data karyawan, termasuk data pribadi, data pekerjaan, dan data lainnya yang terkait. Aplikasi data karyawan ini memberikan kemudahan dalam melakukan pencatatan, pembaruan, dan penghapusan data karyawan secara terpusat. Dengan aplikasi ini, pengguna dapat dengan mudah mengakses informasi karyawan, seperti nama, alamat, nomor telepon, jabatan, gaji, riwayat pekerjaan, dan kualifikasi lainnya. Selain itu, aplikasi data karyawan juga dapat menyediakan fitur-fitur tambahan, seperti pelacakan absensi, pengaturan jadwal kerja, manajemen cuti, dan evaluasi kinerja karyawan. Hal ini membantu dalam mengoptimalkan proses manajemen sumber daya manusia dalam organisasi, meningkatkan efisiensi operasional, dan mempermudah pengambilan keputusan terkait karyawan.

Kata kunci : Aplikasi data karyawan, Manajemen data karyawan, Informasi karyawan, Platform digital.

***ABSTRACT***

The Employee Data Application is a software designed to manage and organize employee-related information within an organization. This application aims to provide an efficient solution for managing employee data, including personal details, job-related information, and other relevant data. The Employee Data Application offers a centralized platform for recording, updating, and deleting employee data. With this application, users can easily access employee information such as names, addresses, contact numbers, job positions, salaries, employment history, and qualifications. Furthermore, the application may provide additional features such as attendance tracking, work schedule management, leave management, and employee performance evaluation. These features help optimize human resource management processes, improve operational efficiency, and facilitate decision-making related to employees.

**BAB I**

**PENDAHULUAN**

* 1. **Latar Belakang**

Dalam lingkungan bisnis yang kompleks dan berkembang pesat, manajemen yang efektif terhadap karyawan menjadi kunci keberhasilan sebuah organisasi. Setiap organisasi membutuhkan informasi yang akurat, terkini, dan terstruktur mengenai karyawan mereka untuk pengambilan keputusan yang tepat dan efisien dalam berbagai aspek pengelolaan sumber daya manusia.

Namun, mengelola data karyawan secara manual dengan menggunakan dokumen fisik atau spreadsheet dapat menjadi tugas yang rumit, memakan waktu, dan rentan terhadap kesalahan. Oleh karena itu, diperlukan sebuah aplikasi data karyawan yang dapat mengotomatisasi dan menyederhanakan proses pengelolaan data karyawan.

Aplikasi data karyawan bertujuan untuk menyediakan platform terpusat yang memungkinkan pengelolaan informasi karyawan secara efisien, akurat, dan aman.

* 1. **Rumusan Masalah**

Dalam pembuatan Aplikasi Data Karyawan, masalah yang perlu diidentifikasi adalah bagaimana?

* 1. **Tujuan dan Manfaat**

Aplikasi data karyawan dirancang untuk mengotomatisasi dan menyederhanakan proses pengelolaan data karyawan, sehingga mengurangi ketergantungan pada proses manual yang memakan waktu dan rentan terhadap kesalahan. mengurangi waktu dan usaha yang dibutuhkan untuk mengelola informasi karyawan, sehingga meningkatkan efisiensi operasional organisasi.

**BAB II**

**TINJAUAN PUSTAKA**

* 1. **Teori Singkat OOP**

OOP (Object-Oriented Programming) adalah paradigma pemrograman yang berfokus pada konsep objek, yang menggabungkan data (atribut) dan fungsi (metode) yang berhubungan dalam satu kesatuan yang disebut objek. Berikut ini adalah teori singkat tentang OOP:

Objek : Objek adalah representasi konkret dari suatu entitas dalam dunia nyata.

Objek memiliki atribut yang menyimpan data dan metode yang menggambarkan perilaku atau tindakan yang dapat dilakukan oleh objek tersebut.

Encapsulation (Pengkapsulan) : Encapsulation adalah mekanisme untuk menggabungkan data dan metode dalam satu unit yang disebut kelas. Konsep ini menerapkan prinsip informasi tersembunyi (data hiding) dengan menyediakan akses terkontrol ke data melalui metode publik.

Inheritance (Pewarisan) : Inheritance adalah konsep yang memungkinkan sebuah kelas untuk mewarisi sifat-sifat (atribut dan metode) dari kelas lain yang lebih umum atau dikenal sebagai kelas induk atau superclass. Dengan inheritance, kelas turunan atau subclass dapat memperluas atau mengubah perilaku kelas induk.

Polymorphism (Polimorfisme) : Polymorphism adalah kemampuan objek untuk memiliki banyak bentuk atau perilaku yang berbeda. Polymorphism memungkinkan penggunaan metode dengan nama yang sama pada objek yang berbeda, sehingga memungkinkan fleksibilitas dan modularitas dalam desain program.

Abstraction (Abstraksi) : Abstraction adalah proses penyederhanaan kompleksitas sistem dengan menyembunyikan detail yang tidak relevan dan hanya menyoroti fitur-fitur penting. Abstraksi memungkinkan pembuatan kelas abstrak yang hanya mendefinisikan kerangka dasar dan metode yang umum untuk kelas-kelas turunannya.

Class (Kelas) : Class adalah blueprint atau cetak biru yang digunakan untuk membuat objek. Class mendefinisikan atribut dan metode yang dimiliki oleh objek dan merupakan entitas utama dalam pemrograman berbasis OOP.

OOP memungkinkan pengorganisasian kode yang lebih terstruktur, modular, dan mudah dipelihara. Dengan menggunakan konsep OOP, pengembang dapat membagi program menjadi unit yang lebih kecil dan saling terkait, mempermudah pengembangan, pemeliharaan, dan penggunaan kembali kode.

* 1. **Teknik/Metode,Tools,dan Solusi**

Dalam pemrograman berbasis OOP, berikut adalah beberapa permasalahan umum, teknik/metode, tools, dan solusi terkait beserta sumber referensi dan data-data yang tepat dan valid:

Permasalahan : Pengelolaan Kode yang Bersih dan Mudah Dipelihara

Solusi : Menerapkan praktik-praktik pemrograman yang baik seperti Single Responsibility Principle (SRP), Don't Repeat Yourself (DRY), dan Separation of Concerns(SoC). Menggunakan refactoring untuk memperbaiki dan menyederhanakan kode yang kompleks.

Teknik/Metode : Refactoring, Code Review, dan Code Smells Detection.

Tools: IDE (Integrated Development Environment) yang menyediakan fitur refactoring seperti IntelliJ IDEA, Eclipse, atau Visual Studio. Linter seperti ESLint atau SonarQube untuk menganalisis kode dan mendeteksi code smells.

Referensi : Martin Fowler. "Refactoring: Improving the Design of Existing Code." Addison-Wesley Professional, 1999.

Robert C. Martin. "Clean Code: A Handbook of Agile Software Craftsmanship." Prentice Hall, 2008.

Permasalahan: Pengujian dan Validasi Perangkat Lunak

Solusi: Menerapkan praktik pengujian perangkat lunak yang komprehensif seperti Test-Driven Development (TDD) dan Behavior-Driven Development (BDD). Menggunakan teknik pengujian seperti unit testing, integrasi testing, dan acceptance testing.

Teknik/Metode: TDD, BDD, dan Continuous Integration/Continuous Deployment (CI/CD).

Tools: JUnit, NUnit, pytest (framework pengujian unit), Selenium atau Cypress (framework pengujian integrasi), Cucumber (framework BDD), dan Jenkins atau Travis CI (tools CI/CD).

Referensi:

Kent Beck. "Test Driven Development: By Example." Addison-Wesley Professional, 2002.

Robert C. Martin. "Clean Agile: Back to Basics." Prentice Hall, 2019.

Permasalahan: Pengelolaan Versi dan Kolaborasi Tim

Solusi: Menggunakan sistem pengendalian versi (version control system) untuk melacak perubahan dan mengelola kode secara kolaboratif. Menggunakan alur kerja pengembangan yang efisien seperti Gitflow atau Agile Scrum.

Teknik/Metode: Git, Gitflow Workflow, dan Continuous Integration/Continuous Deployment (CI/CD).

Tools: Git (sistem pengendalian versi), GitHub, GitLab, atau Bitbucket (platform kolaborasi dan hosting repository), dan Jenkins atau Travis CI (tools CI/CD).

Referensi:

Scott Chacon, Ben Straub. "Pro Git." Apress, 2014.

Vincent Driessen. "A successful Git branching model." [Online]. Available: <https://nvie.com/posts/a-successful-git-branching-model/>.

Permasalahan: Optimasi Kinerja dan Skalabilitas

Solusi: Menerapkan teknik optimasi kinerja seperti caching, penggunaan algoritma dan struktur data yang efisien, pengoptimalan database, dan scaling horizontal/vertikal.

Teknik/Metode: Caching, indexing, query optimization, load balancing, dan horizontal/vertical scaling.

**BAB III**

**ANALISIS DAN DESAIN**

* 1. **Sistem Berjalan**

dibuat berdasarkan kebutuhan yang telah diidentifikasi. Struktur data yang efisien dirancang untuk menyimpan informasi karyawan. Diagram ER (Entity-Relationship) dan skema database digunakan untuk menggambarkan hubungan antara entitas dalam sistem seperti karyawan, jabatan, atau departemen.

Identifikasi dan Analisis Pengguna : Makukan wawancara, survei, atau pengamatan langsung untuk memahami kebutuhan dan tujuan pengguna. Identifikasi masalah yang mereka hadapi dalam mengelola data karyawan, tugas-tugas khusus yang mereka lakukan menggunakan aplikasi, dan manfaat yang diharapkan dari penggunaan aplikasi data karyawan.

* 1. **Identifikasi**

Desain Antarmuka Pengguna : Rancang antarmuka pengguna yang intuitif dan responsif. perlu dilakukan pengujian dan iterasi berulang untuk memastikan kelayakan, keterbacaan, dan kegunaan antarmuka. Penggunaan prinsip desain yang baik, seperti prinsip kejelasan, konsistensi, dan kegunaan, akan meningkatkan pengalaman pengguna dan efektivitas aplikasi.

Analisis Kebutuhan Fungsional:

Kebutuhan fungsional yang harus dilakukan oleh Aplikasi Data Karyawan, Pengguna harus dapat membuat, mengedit, dan menghapus profil karyawan. Informasi yang dapat dimasukkan termasuk nama, alamat, tanggal lahir, nomor telepon, jabatan, departemen, dan informasi kontak lainnya.

Analisis Kebutuhan Non-Fungsional:

Kebutuhan non-fungsional yang harus dipenuhi oleh aplikasi Data Karyawan ini adalah , menjaga kerahasiaan dan integritas data karyawan, menerapkan mekanisme autentikasi yang kuat. Menentukan persyaratan khusus untuk setiap kebutuhan non-fungsional, misalnya perlindungan terhadap serangan keamanan seperti peretasan atau malware.

Analisis Prototyping:

Analisis prototyping dalam aplikasi data karyawan melibatkan pembuatan model atau versi awal dari aplikasi yang berfungsi untuk menguji dan mendapatkan umpan balik dari pengguna sebelum mengembangkan versi final. Tujuan dari prototyping adalah untuk mengidentifikasi kebutuhan dan keinginan pengguna, memvalidasi desain dan fungsionalitas, serta mempercepat proses pengembangan.

prototyping adalah proses iteratif, dan prototipe awal mungkin berubah dan berkembang seiring dengan berjalannya waktu. Prototyping membantu memastikan bahwa aplikasi data karyawan yang akhir nantinya memenuhi kebutuhan pengguna dengan lebih baik.

* 1. **Tahapan Pengembangan**

Analisis Kebutuhan:

Identifikasi kebutuhan fungsional dan non-fungsional dari aplikasi Data Karyawan melibatkan identifikasi dan pemahaman terhadap kebutuhan pengguna serta fungsi yang diinginkan dari aplikasi tersebut, bertujuan untuk memastikan bahwa aplikasi data karyawan dapat memenuhi kebutuhan pengguna secara efektif dan efisien, serta memberikan nilai tambah dalam pengelolaan data karyawan dalam organisasi.

Desain:

* Rancang desain aplikasi Data Karyawan berdasarkan kebutuhan yang telah diidentifikasi. Rancangan desain aplikasi mencakup tabel-tabel seperti "Karyawan" untuk menyimpan informasi pribadi karyawan, "Departemen" untuk mengelola departemen dalam perusahaan, "Jabatan" untuk mengelola jabatan karyawan, dan lain-lain.

Implementasi:

Implementasikan desain yang telah dirancang ke dalam kode program.

Gunakan bahasa pemrograman seperti Java sesuai kebutuhan.

Buat kelas-kelas, fungsi-fungsi, dan modul-modul yang dibutuhkan dalam aplikasi.

Pengujian:

Lakukan pengujian aplikasi untuk memastikan bahwa semua fitur berfungsi dengan baik. Uji fungsionalitas, kesalahan, dan kinerja aplikasi.

Identifikasi dan perbaiki bug atau masalah yang ditemukan selama pengujian.

Pemeliharaan:

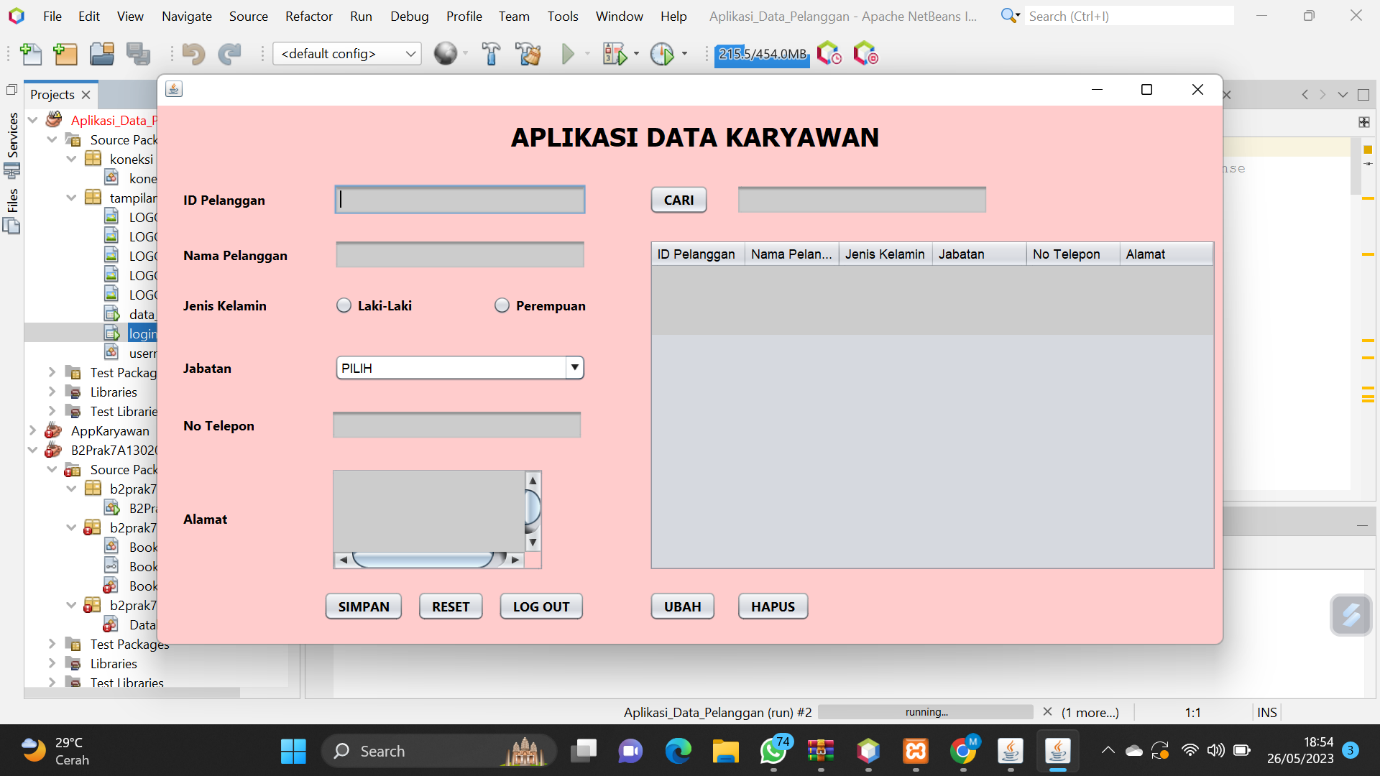
Setelah pengujian selesai dan aplikasi sudah berjalan, lakukan pemeliharaan rutin untuk menjaga dan memperbaiki aplikasi jika diperlukan.

Tangani masalah yang muncul, perbarui fitur, dan tanggapi umpan balik dari pengguna.

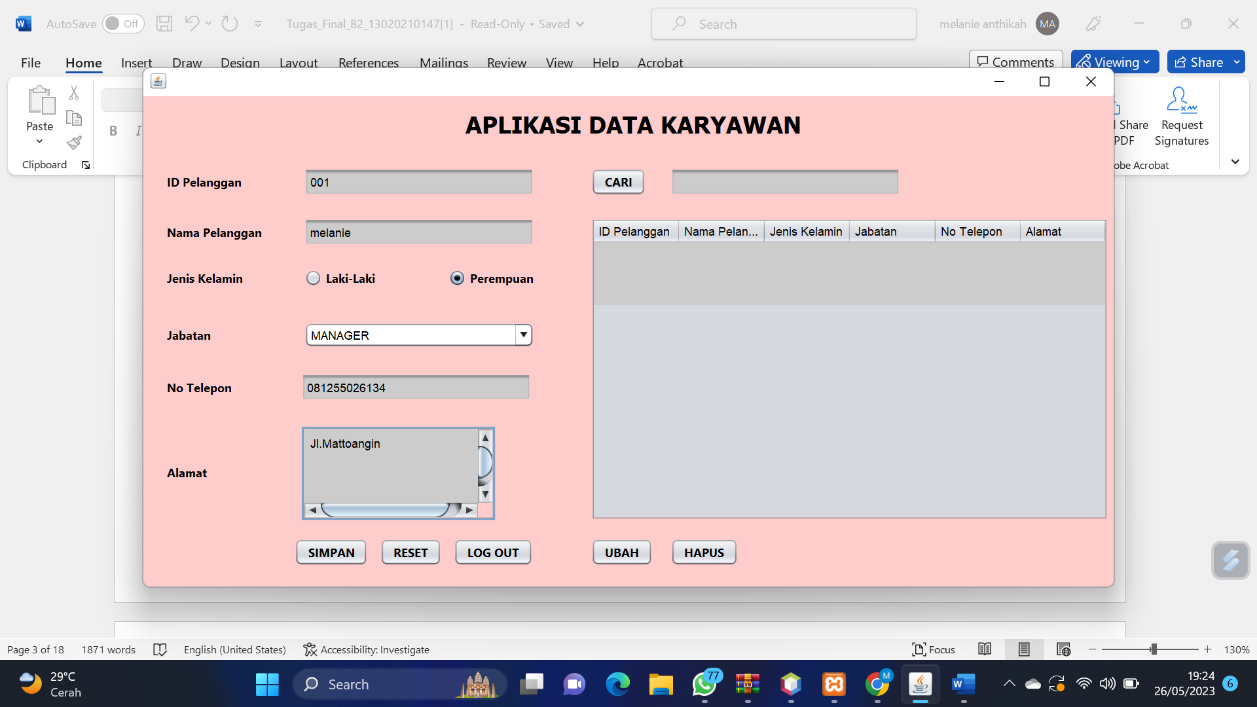
**BAB IV**

**IMPLEMENTASI DAN PENGUJIAN**

* 1. **Tampilan**
* Tampilan awal



* Mengisi form id, nama, jenis kelamin, jabatan, no telepon, alamat.



**BAB V**

**PENUTUP**

* 1. **Kesimpulan**

Saya menyimpulkan bahwa pembuatan aplikasi Data Karyawan, langkah-langkah analisis dan desain yang cermat sangat penting untuk menghasilkan aplikasi yang efisien, user-friendly, dan sesuai dengan kebutuhan pengguna. Dengan adanya aplikasi data karyawan, manajemen sumber daya manusia atau departemen personalia dapat dengan mudah mengelola informasi karyawan secara efisien dan akurat.

**DAFTAR PUSTAKA**

Deitel, P., & Deitel, H. (2017). Java: How to Program (Early Objects). Pearson.

Horstmann, C. S., & Cornell, G. (2018). Core Java Volume I--Fundamentals (11th Edition). Prentice Hall.

Freeman, E., Robson, E., Bates, B., & Sierra, K. (2019). Head First Java (2nd Edition). O'Reilly Media.

Downs, R. (2014). JavaFX 8 Introduction by Example (2nd Edition). Apress.

McDowell, J. (2019). Java Coding Interview Pocket Book. CareerCup.

Rischpater, R. (2018). The Definitive Guide to SWT and JFace. Apress.

Määttä, M. (2019). Swing and JavaFX: What's New, What's the Same?. Apress.

Liguori, R., & Liguori, W. (2018). Java 9 Cookbook. Packt Publishing.

Eckel, B. (2006). Thinking in Java (4th Edition). Prentice Hall.

Cadenhead, R., & Lemay, L. (2018). Java in 24 Hours, Sams Teach Yourself (8thn Edition). Sams Publishing.

**LAMPIRAN**

Link aplikasi:

Link video persentase: