**PRAKTIKUM REKAYASA PERANGKAT LUNAK 2**



|  |
| --- |
| **MANUAL BOOK**  ***“Judul Aplikasi”* Menggunakan**  **Netbeans Nama Anggota :**  **Bonaventura Dimas Sakti Karunia 50421290**  **Haziq Duha Zainul Ihsan 51421599**  **Senopatyi Khalifa Zalatin S. 514**21405    **Kelas : 4IA14**  **Fakultas** **: Teknologi Industri**  **Jurusan** **: Teknik Informatika**  **PJ : *aji*** |

**Ditulis Guna Melengkapi Sebagian Syarat Praktikum Pemrograman Web Jenjang S1**

**Universitas Gunadarma**

**2019**

**BAB I**

**PENDAHULUAN**

# 1.1 Latar Belakang

Program ini dirancang untuk mengelola data karyawan dalam sebuah perusahaan dengan menggunakan arsitektur berbasis **Spring Boot**. Mengelola data karyawan merupakan aspek penting dalam administrasi perusahaan, terutama untuk memastikan informasi seperti identitas karyawan, jabatan, dan gaji tercatat dengan baik. Dengan adanya program ini, diharapkan dapat meningkatkan efisiensi dan akurasi dalam manajemen data karyawan, menggantikan proses manual yang rentan terhadap kesalahan.

Program ini juga mendukung operasi **CRUD (Create, Read, Update, Delete)**, sehingga mempermudah administrator untuk menambah, mengedit, atau menghapus data karyawan secara langsung melalui antarmuka. Komponen seperti **ModelTabelKaryawan** diintegrasikan untuk mempermudah representasi data dalam bentuk tabel pada aplikasi desktop berbasis Java Swing, sehingga program ini tidak hanya cocok untuk backend tetapi juga memberikan dukungan antarmuka pengguna.

# 1.2 Tujuan

Tujuan dari dibuatnya program ini adalah untuk menyediakan sistem pengelolaan data karyawan yang terpusat dan efisien, sehingga memudahkan administrator dalam melakukan operasi seperti penambahan, pembaruan, penghapusan, dan pengambilan data karyawan. Program ini dirancang untuk meningkatkan keakuratan data melalui penyimpanan yang terintegrasi dengan database menggunakan, serta memastikan pengolahan data dilakukan secara otomatis dan terstruktur. Selain itu, program ini mendukung visualisasi data dalam bentuk tabel melalui antarmuka desktop berbasis Java Swing, sehingga mempermudah pengguna dalam mengelola informasi karyawan. Dengan arsitektur yang modular, program ini juga dirancang agar mudah dikembangkan lebih lanjut untuk menambahkan fitur tambahan sesuai kebutuhan perusahaan.

**BAB II**

**PEMBAHASAN**

# 2.1 NetBeans

NetBeans dijelaskan sebagai serambi pengembangan perangkat lunak (IDE) yang ditulis menggunakan Java. IDE ini mempermudah pengembangan aplikasi dengan menyediakan alat yang mendukung proses coding, debugging, dan implementasi aplikasi berbasis Java.

 **Fungsi NetBeans**:

* Membantu pengembang dalam menulis, mengedit, dan debug kode.
* Menyediakan fitur seperti autocomplete, debugging tools, dan integrasi dengan sistem versioning seperti Git.

 **Kelebihan**:

* Open-source dan gratis digunakan.
* Memiliki komunitas yang besar untuk dukungan.
* Mendukung pengembangan aplikasi berbasis desktop, web, dan mobile.

**2.2 Spring**

Spring diidentifikasi sebagai framework untuk pengembangan aplikasi Java, khususnya pada aplikasi berbasis web. Spring menawarkan berbagai modul seperti Spring Boot untuk mempermudah konfigurasi dan Spring Data untuk manipulasi database.

 **Tujuan Spring**:

* Mengurangi kompleksitas pengembangan aplikasi.
* Memberikan arsitektur modular melalui dependency injection dan pengelolaan aplikasi berbasis POJO (Plain Old Java Object).

 **Fitur Utama**:

* **Spring Boot**: Mempermudah konfigurasi aplikasi berbasis Spring.
* **Spring Data**: Mengelola koneksi dan manipulasi database.
* **Spring MVC**: Membantu pengembangan aplikasi berbasis web dengan arsitektur Model-View-Controller.

 **Kelebihan**:

* Fleksibel dan dapat diintegrasikan dengan framework lain.
* Dokumentasi yang lengkap dan komunitas besar.

# 2.3 Hibernate

Hibernate dijelaskan sebagai fitur sistem operasi yang terkait dengan pengelolaan status memori. Namun, kemungkinan terdapat miskonsepsi dalam penulisan karena Hibernate sebenarnya adalah framework untuk Object Relational Mapping (ORM) yang digunakan untuk mempermudah akses dan pengelolaan database di aplikasi berbasis Java.

## 2.3.1.1 Fungsi Hibernate

Fungsi Hibernate adalah untuk menyimpan aktivitas memori dan memberikan opsi untuk melanjutkan pekerjaan tanpa kehilangan data. Ini lebih terkait dengan sistem hibernasi pada perangkat laptop, bukan framework Hibernate.

 Pada fitur sistem operasi, hibernate memungkinkan menyimpan status aktivitas komputer ke disk sehingga dapat dilanjutkan setelah dinyalakan kembali.

 Biasanya digunakan untuk laptop agar dapat menghemat daya.

## 2.3.1.2 Manfaat Hibernate

Ada beberapa manfaat hibernate yaitu :

 Menyimpan data atau pekerjaan saat perangkat mati.

 Meminimalkan risiko kehilangan data.

 Memungkinkan pengguna melanjutkan pekerjaan dengan cepat setelah dinyalakan kembali.

**2.3.2 Kerugian dan Masalah Hibernate**

Kerugian hibernate yaitu;

 Waktu proses hibernate bisa lebih lama dibandingkan sleep mode.

 Membutuhkan ruang disk yang cukup besar untuk menyimpan aktivitas memori.

 Tidak cocok untuk perangkat dengan kapasitas penyimpanan kecil.

# 2.4 ORM(Object Relational Mapping)

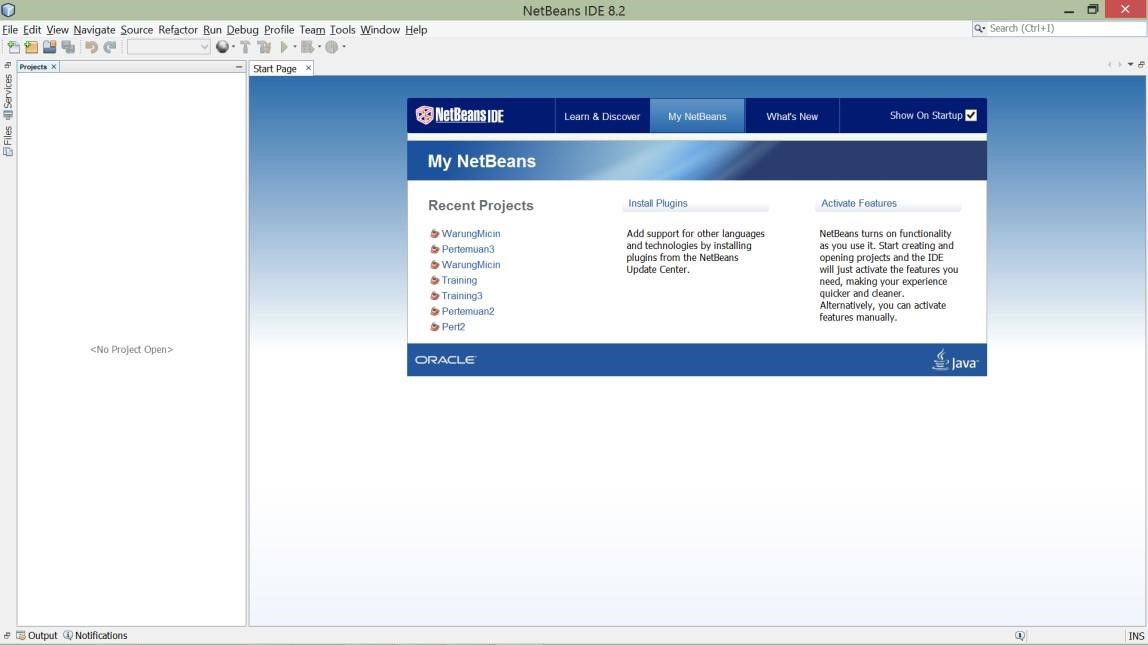
ORM adalah metode pemrograman yang digunakan untuk menjembatani dunia objek dalam bahasa pemrograman berorientasi objek (OOP) dengan struktur tabel di database relasional.

* **Contoh ORM**: Hibernate, JPA, dan MyBatis.
* **Manfaat ORM**:
  + Mengurangi kompleksitas pengelolaan database.
  + Membantu pengembang fokus pada logika aplikasi tanpa memikirkan query SQL manual.
  + Mempermudah migrasi database dengan menyesuaikan perubahan model objek.
* **Konsep Dasar**:
  + Setiap tabel database direpresentasikan sebagai kelas.
  + Setiap kolom tabel direpresentasikan sebagai atribut dalam kelas.
  + Relasi antar tabel (seperti one-to-many, many-to-many) direpresentasikan sebagai hubungan antar objek.

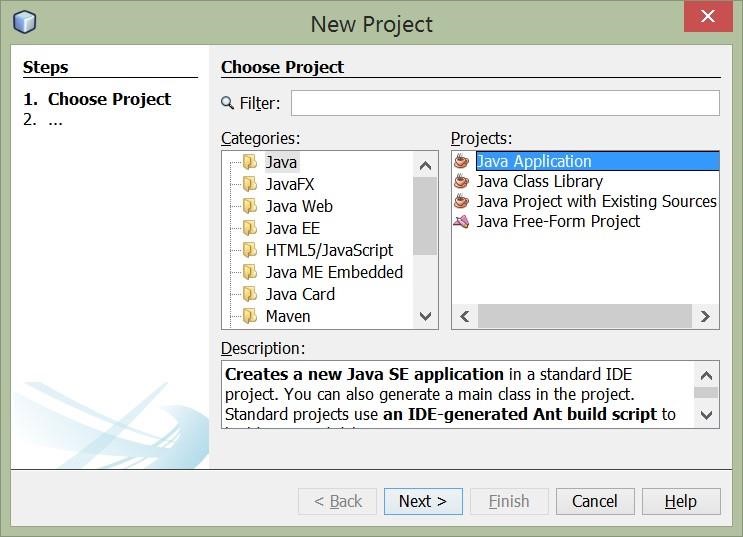
**BAB III**

**ANALISA DAN PERANCANGAN**

Project yang akan kita buat yaitu bertema Manajemen Data Karyawan, denga nama Aplikasi Sistem Informasi Karyawan. Dibawah ini merupakan tampilan awal pada Netbeans.



Langkah selanjutnya click “ new project “, pilih Java Application terus Next sperti gambar dibawah ini.



File Ujian\_bvn.java

|  |
| --- |
| /\*  \* Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license  \*/  package me.boven;  import me.boven.controller.KaryawanController;  import me.boven.service.KaryawanService;  import me.boven.view.KaryawanView;  import org.springframework.boot.ApplicationArguments;  import org.springframework.boot.ApplicationRunner;  import org.springframework.boot.SpringApplication;  import org.springframework.boot.autoconfigure.SpringBootApplication;  import org.springframework.beans.factory.annotation.Autowired;  import org.springframework.context.ApplicationContext;  /\*\*  \*  \* @author WINDOWS 10  \*/  @SpringBootApplication  public class Ujian\_bvn implements ApplicationRunner {  @Autowired  private KaryawanService karyawanService;  public static void main(String[] args) {  System.setProperty("java.awt.headless", "false"); // Disable headless mode  // Start the Spring application and get the application context  ApplicationContext context = SpringApplication.run(Ujian\_bvn.class, args);  // Instantiate the view and inject the controller manually  KaryawanController controller = context.getBean(KaryawanController.class);  KaryawanView mahasiswaView = new KaryawanView(controller);  mahasiswaView.setVisible(true);  }  @Override  public void run(ApplicationArguments args) throws Exception {  }  } |

File KaryawanController.java

|  |
| --- |
| /\*  \* Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license  \* Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template  \*/  package me.boven.controller;  import org.springframework.beans.factory.annotation.Autowired;  import org.springframework.web.bind.annotation.\*;  import me.boven.model.ModelKaryawan;  import me.boven.service.KaryawanService;  import java.util.List;  import org.springframework.stereotype.Controller;  @Controller  public class KaryawanController {  @Autowired  private KaryawanService karyawanService;  // Add new Mahasiswa  public String addKaryawan(@RequestBody ModelKaryawan krw) {  karyawanService.addK(krw);  return "Karyawan added successfully";  }  // Get Mahasiswa by ID  public ModelKaryawan getKaryawan(@PathVariable int id) {  return karyawanService.getK(id);  }  // Update Mahasiswa  public String updateKaryawan(@RequestBody ModelKaryawan krw) {  karyawanService.updateK(krw);  return "Karyawan updated successfully";  }  // Delete Mahasiswa by ID  public String deleteMahasiswa(@PathVariable int id) {  karyawanService.deleteK(id);  return "Mahasiswa deleted successfully";  }  // Get all Mahasiswa  public List<ModelKaryawan> getAllK() {  return karyawanService.getAllK();  }  } |

File ModelKaryawan.java

|  |
| --- |
| /\*  \* Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license  \* Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template  \*/  package me.boven.model;  import jakarta.persistence.\*;  /\*\*  \*  \* @author Lenovo  \*/  @Entity  @Table(name = "karyawan")  public class ModelKaryawan {  @Id  @GeneratedValue(strategy = GenerationType.IDENTITY)  @Column(name = "ID")  private int id;  @Column(name = "Nama", nullable = false, length = 8)  private String nama;  @Column(name = "Jabatan", nullable = false, length = 50)  private String jabatan;  @Column(name = "Gaji")  private int gaji;  //Buka code -> generate -> constructor -> all  //code -> generate -> setter and getter -> All  //Baru buat anotasi @ nya      // constructor kosong untuk ORM  public ModelKaryawan() {  }    //constructor ada isinya untuk dipanggil di aplikasi  public ModelKaryawan(int id, String nama, String jabatan, int gaji) {  this.id = id;  this.nama = nama;  this.jabatan = jabatan;  this.gaji = gaji;  }  public int getId() {  return id;  }  public void setId(int id) {  this.id = id;  }    public String getNama() {  return nama;  }  public void setNama(String nama) {  this.nama = nama;  }  public String getJabatan() {  return jabatan;  }  public void setJabatan(String jabatan) {  this.jabatan = jabatan;  }  public float getGaji() {  return gaji;  }  } |

File ModelTabelKaryawan.java

|  |
| --- |
| /\*  \* Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license  \* Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template  \*/  package me.boven.model;  import javax.swing.table.AbstractTableModel;  import java.util.List;  public class ModelTabelKaryawan extends AbstractTableModel{  private List<ModelKaryawan> karyawanList;  private String[] columnNames = {"ID", "NAMA", "JABATAN", "GAJI"};  public ModelTabelKaryawan(List<ModelKaryawan> karyawanList) {  this.karyawanList = karyawanList;  }  @Override  public int getRowCount() {  return karyawanList.size(); // Jumlah baris sesuai dengan jumlah data mahasiswa  }  @Override  public int getColumnCount() {  return columnNames.length; // Jumlah kolom sesuai dengan jumlah elemen dalam columnNames  }  @Override  public Object getValueAt(int rowIndex, int columnIndex) {  ModelKaryawan karyawan = karyawanList.get(rowIndex);  switch (columnIndex) {  case 0:  return karyawan.getId();  case 1:  return karyawan.getNama();  case 2:  return karyawan.getJabatan();  case 3:  return karyawan.getGaji();  default:  return null;  }  }  @Override  public String getColumnName(int column) {  return columnNames[column]; // Mengatur nama kolom  }  @Override  public boolean isCellEditable(int rowIndex, int columnIndex) {  return false; // Semua sel tidak dapat diedit  }  // Method untuk menambahkan atau memodifikasi data, jika dibutuhkan  public void setKaryawanList(List<ModelKaryawan> karyawanList) {  this.karyawanList = karyawanList;  fireTableDataChanged(); // Memberitahu JTable bahwa data telah berubah  }  } |

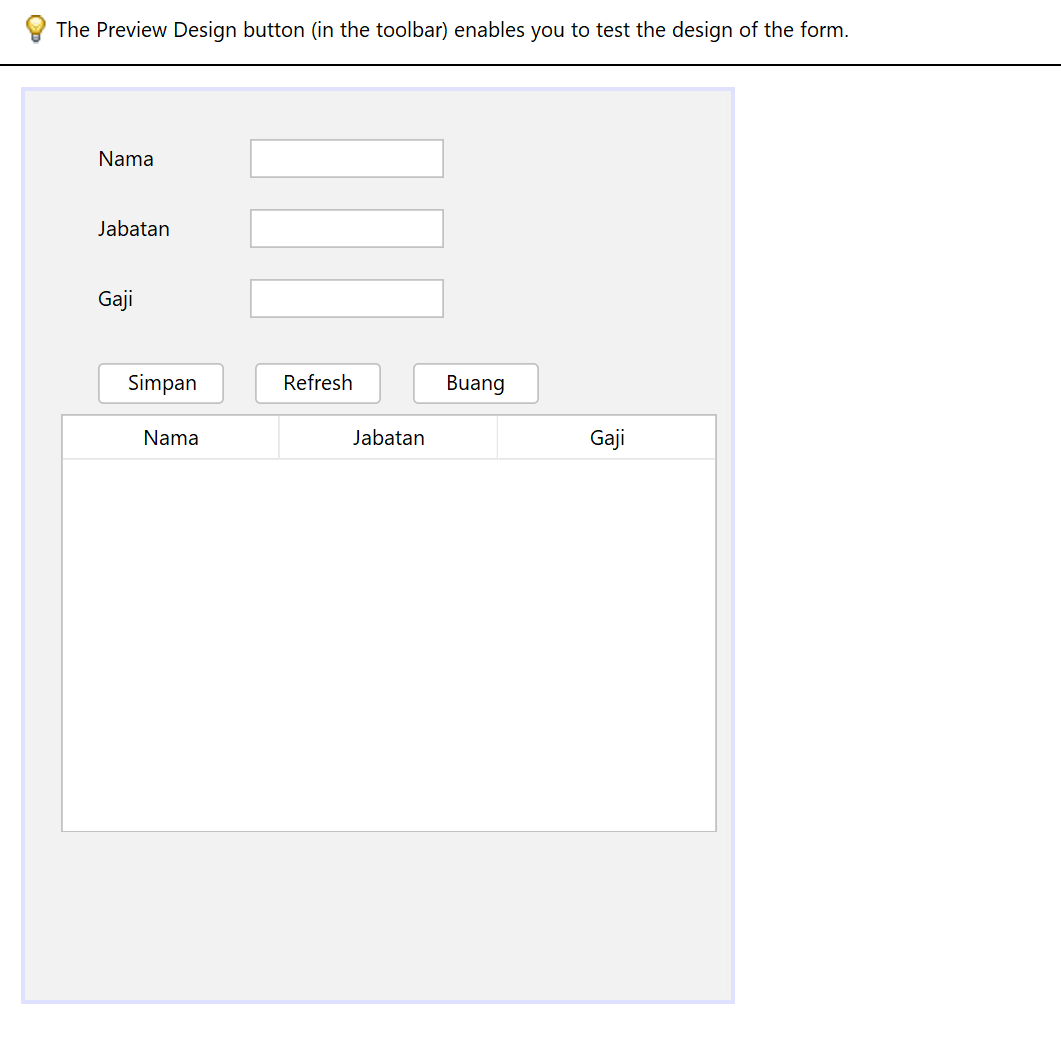
File KaryawanRepository.java

|  |
| --- |
| /\*  \* Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license  \* Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template  \*/  package me.boven.repository;  import me.boven.model.ModelKaryawan;  import org.springframework.data.jpa.repository.JpaRepository;  import org.springframework.stereotype.Repository;  /\*\*  \*  \* @author WINDOWS 10  \*/  @Repository  public interface KaryawanRepository extends JpaRepository<ModelKaryawan, Integer> {    } |

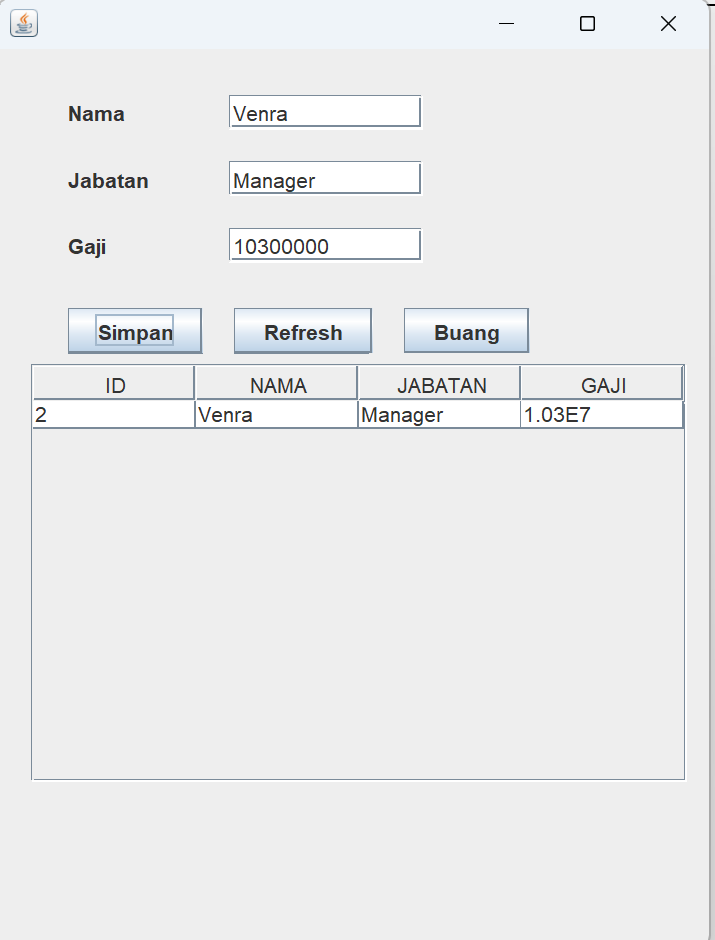
File KaryawanService.java

|  |
| --- |
| /\*  \* Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license  \* Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template  \*/  package me.boven.service;  import me.boven.model.ModelKaryawan;  import me.boven.repository.KaryawanRepository;  import java.util.List;  import org.springframework.beans.factory.annotation.Autowired;  import org.springframework.stereotype.Service;  /\*\*  \*  \* @author WINDOWS 10  \*/  @Service  public class KaryawanService {    @Autowired  private KaryawanRepository repository;  public void addK(ModelKaryawan krw) {  repository.save(krw);  }  public ModelKaryawan getK(int id) {  return repository.findById(id).orElse(null);  }  public void updateK(ModelKaryawan krw) {  repository.save(krw);  }  public void deleteK(int id) {  repository.deleteById(id);  }  public List<ModelKaryawan> getAllK() {  return repository.findAll();  }  } |

File KaryawanView.java



Output Program :



**BAB IV**

**PENUTUP**

**4.1 Kesimpulan**

Dari hasil pembahasan yang telah dijelaskan sebelumnya, dapat disimpulkan bahwa aplikasi yang dibangun menggunakan Spring Boot dan teknologi terkait, seperti Hibernate dan JPA, memberikan banyak kemudahan dalam pengembangan aplikasi berbasis Java, terutama dalam konteks pengelolaan data karyawan di sebuah perusahaan. Berikut adalah beberapa kesimpulan penting yang dapat diambil:

1. **Spring Boot Mempermudah Pengembangan Aplikasi**  
   Dengan menggunakan Spring Boot, proses pengembangan aplikasi menjadi lebih sederhana dan cepat. Spring Boot menyembunyikan banyak konfigurasi yang kompleks dan memungkinkan pengembang fokus pada logika aplikasi. Adanya fitur **dependency injection** dan **auto-configuration** memungkinkan aplikasi berjalan dengan lebih efisien dan mudah dikembangkan.
2. **Penggunaan Hibernate untuk ORM (Object-Relational Mapping)**  
   Hibernate sebagai framework ORM sangat mendukung dalam pengelolaan hubungan antara objek dalam Java dengan data dalam database relasional. Dengan Hibernate, pengembang tidak perlu lagi menulis query SQL manual untuk operasi dasar seperti **CRUD** (Create, Read, Update, Delete), yang sangat mengurangi kompleksitas pengelolaan database. Hibernate juga mempermudah dalam mengelola hubungan antar objek, seperti **one-to-many** atau **many-to-many**, tanpa perlu menulis kode SQL yang rumit.
3. **Spring Data JPA**  
   Penggunaan **Spring Data JPA** memungkinkan aplikasi untuk mengelola entitas data karyawan dengan mudah dan efisien. Spring Data menyediakan antarmuka yang dapat digunakan untuk berinteraksi dengan database, memungkinkan pengelolaan data melalui **Repository**. Selain itu, JPA memberikan dukungan untuk fitur-fitur seperti lazy loading dan transaksi database yang lebih baik.
4. **Model-View-Controller (MVC)**  
   Arsitektur **Model-View-Controller** yang diimplementasikan dalam aplikasi ini memisahkan antara logika aplikasi (Model), antarmuka pengguna (View), dan kontrol alur aplikasi (Controller). Ini membuat kode lebih terstruktur, mudah dipelihara, dan memudahkan pengembangan lebih lanjut. Aplikasi ini memanfaatkan Spring MVC untuk memfasilitasi pengembangan aplikasi berbasis web dengan pemisahan yang jelas antara komponen.
5. **Penggunaan GUI untuk Visualisasi Data**  
   Untuk meningkatkan pengalaman pengguna, aplikasi ini menggunakan antarmuka grafis berbasis Java Swing untuk menampilkan data karyawan dalam bentuk tabel. Dengan menggunakan **ModelTabelKaryawan**, data yang diambil dari database dapat ditampilkan dalam tampilan tabel yang interaktif. Hal ini memberikan kemudahan bagi pengguna untuk melihat dan berinteraksi dengan data karyawan.
6. **Manfaat dan Kelebihan Spring**  
   Spring sebagai framework memiliki banyak kelebihan, seperti fleksibilitas, skalabilitas, dan dukungan untuk berbagai jenis aplikasi. Selain itu, Spring menyediakan banyak modul tambahan yang memudahkan pengembangan aplikasi berbasis Java. Kemudahan integrasi dengan teknologi lain dan dokumentasi yang lengkap membuat Spring sangat populer di kalangan pengembang aplikasi.

Secara keseluruhan, aplikasi yang dibangun dengan Spring Boot, Hibernate, dan teknologi terkait lainnya memberikan solusi yang efektif dan efisien dalam mengelola data karyawan. Dengan menggunakan arsitektur yang terstruktur dan pemisahan yang jelas antara komponen aplikasi, pengembangan dan pemeliharaan aplikasi menjadi lebih mudah dan cepat. Keuntungan lainnya adalah aplikasi ini juga dapat dengan mudah dikembangkan lebih lanjut untuk menambahkan fitur-fitur baru yang dibutuhkan oleh pengguna.

# 4.2 Saran