BAB 3
PELAKSANAAN KERJA MAGANG

3.1 Posisi dan Koordinasi

1. Posisi

   Pada kegiatan magang di PT Winnicode Garuda Teknologi, penulis ditempatkan sebagai *Fullstack Developer Intern* yang bertanggung jawab dalam pengembangan *website* lowongan pekerjaan perusahaan. Dalam posisi ini, penulis memperoleh tugas langsung dari pembimbing lapangan untuk merancang dan membangun sistem secara menyeluruh, mulai dari merancang desain sistem, implementasi fitur inti *website*, hingga peningkatan keamanan dan efisiensi sistem.

   Struktur perusahaan yang sederhana membuat penulis bekerja secara mandiri dalam proses teknis pengembangan *website*, tanpa pembagian tim atau subdivisi khusus. Penulis berperan sebagai *developer* penuh yang menjalankan seluruh proses pengembangan dari tahap perancangan hingga penyelesaian.

2. Koordinasi

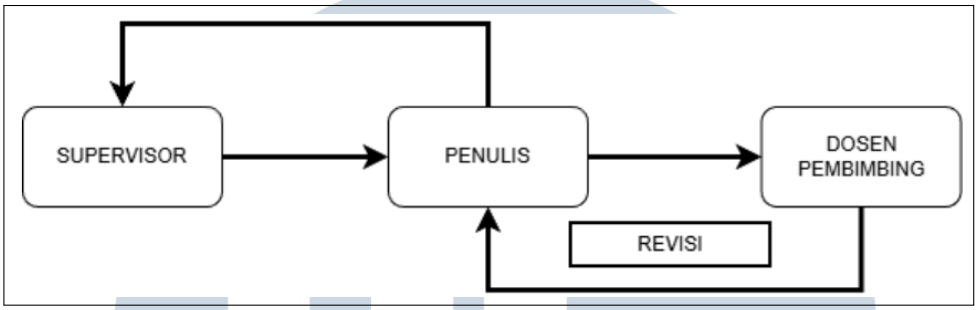
   Proses koordinasi kerja selama magang dilakukan melalui dua jalur yang berbeda sesuai dengan fungsi masing-masing pihak, yaitu *supervisor* perusahaan dan dosen pembimbing.

   (a) *supervisor* (pembimbing lapangan) *supervisor* memberikan tugas inti (dalam hal ini memberikan tugas untuk membuat *website* lowongan pekerjaan) dan juga menerima laporan progress mingguan dari penulis serta memberikan arahan tambahan untuk penulis dalam mengerjakan proyek.

   (b) Dosen Pembimbing Dosen pembimbing memberikan arahan terkait pembuatan dan penyusunan laporan magang yang menyangkut revisi dan bimbingan terhadap draft laporan.

   Dengan demikian, penulis melakukan pekerjaan teknis sebagaimana tugas yang diberikan oleh *supervisor* serta menyampaikan progress yang sudah dikerjakan, sementara penyusunan laporan dan revisi akademik dilakukan

bersama dosen pembimbing. Alur koordinasi pelaksanaan kerja magang dapat di visualisasikan seperti gambar 3.1 dibawah ini:



Gambar 3.1. Bagan alur koordinasi

3.2 Tugas yang Dilakukan

Linimasa selama mengerjakan proyek magang yang terdapat pada tabel 3.1

Tabel 3.1. Linimasa pengerjaan proyek

No.	Minggu ke-	Proyek	Keterangan
1	1	Perancangan proposal	Perencanaan konsep dari proyek ini, mulai dari alur *website*, tools yang digunakan, hingga tujuan proyek.
2	2–4	Gambaran proyek	Merancang *mockup* untuk memberikan gambaran mengenai *website*.
3	2–4	Laman *signup*	Implementasi halaman registrasi *user* maupun *employer*.
4	2–4	Halaman *login*	Implementasi halaman *login user* maupun *employer*.
5	2–4	Laman utama	Implementasi halaman utama (*dashboard*) sebagai *landing page*.
6	2–4	Laman *saved jobs* dan *applied jobs*	Implementasi fitur utama untuk *user*.
7	2–4	Halaman *job vacancy*	Implementasi fitur utama baik untuk *user* maupun *employer*.

No.	Minggu ke-	Proyek	Keterangan
8	2–4	Laman profil	Implementasi informasi pribadi *user* maupun *employer*.
9	5–8	Fitur *posting jobs* (*employer*)	Implementasi fitur utama untuk *employer*.
10	5–8	Fitur *my jobs* (*employer*)	Implementasi fitur untuk menampilkan pekerjaan yang sudah dibuat.
11	5–8	Laman *contact*	Implementasi halaman kontak untuk menampilkan informasi perusahaan.
12	5–8	Laman *about*	Implementasi halaman berupa informasi dan deskripsi singkat dari perusahaan tersebut.
13	9–15	Laman *backend*	Implementasi *backend* untuk kompleksitas dari *website*.
14	16	Finalisasi	Pengujian *website* dan penyusunan laporan akhir.

3.3 Uraian Pelaksanaan Kerja

Proyek utama yang dikembangkan adalah sebuah *website* lowongan pekerjaan yang dibagi menjadi dua bagian utama berdasarkan peran, yaitu *user* dan *employer*. *user* merupakan pihak yang dapat mengakses pekerjaan yang diposting, sementara *employer* memiliki akses untuk memposting lowongan pekerjaan yang nantinya akan bisa di akses oleh *user*. *employer* dapat menambah dan juga mengedit lowongan pekerjaan yang nantinya akan ditampilkan di halaman utama *user*.

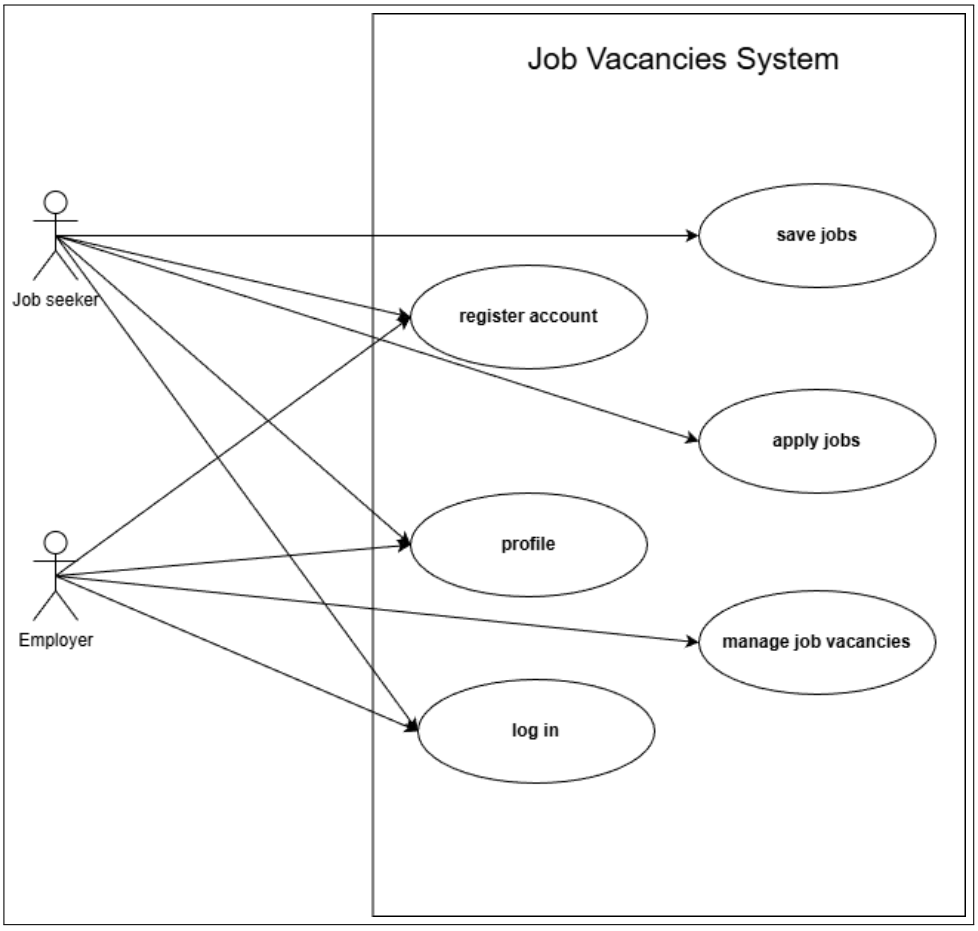
Proyek ini dilaksanakan secara mandiri dengan mengumpulkan progress proyek kepada *supervisor* melalui platform Google Classroom. Meskipun fleksibel, perancangan dan pengembangan proyek tetap menjadi kewajiban utama yang harus diselesaikan secara bertahap sesuai arahan dan standar yang ditetapkan oleh perusahaan.

3.3.1 Perancangan Proposal dan Tahapan Awal

Tahap awal pengembangan difokuskan pada pembuatan frontend dari *website* utama yang berfungsi menampilkan berbagai data berupa daftar pekerjaan yang tersimpan di dalam *localstorage*. Dengan metode ini, pengembangan *frontend* hingga ke *backend* dan *database* dapat berjalan lebih terstruktur. *supervisor* memberikan tugas untuk membuat proposal sebagai dasar pelaksanaan magang. Proposal mencakup tujuan sistem, *scope* proyek, kebutuhan fitur, serta target akhir yang dapat digunakan untuk pengerjaan dan penyusunan proyek. Menggunakan Microsoft Word untuk pembuatan dan penyusunan proposal dan aplikasi Whatsapp untuk koordinasi dengan *supervisor*.

3.3.2 Gambaran Proyek

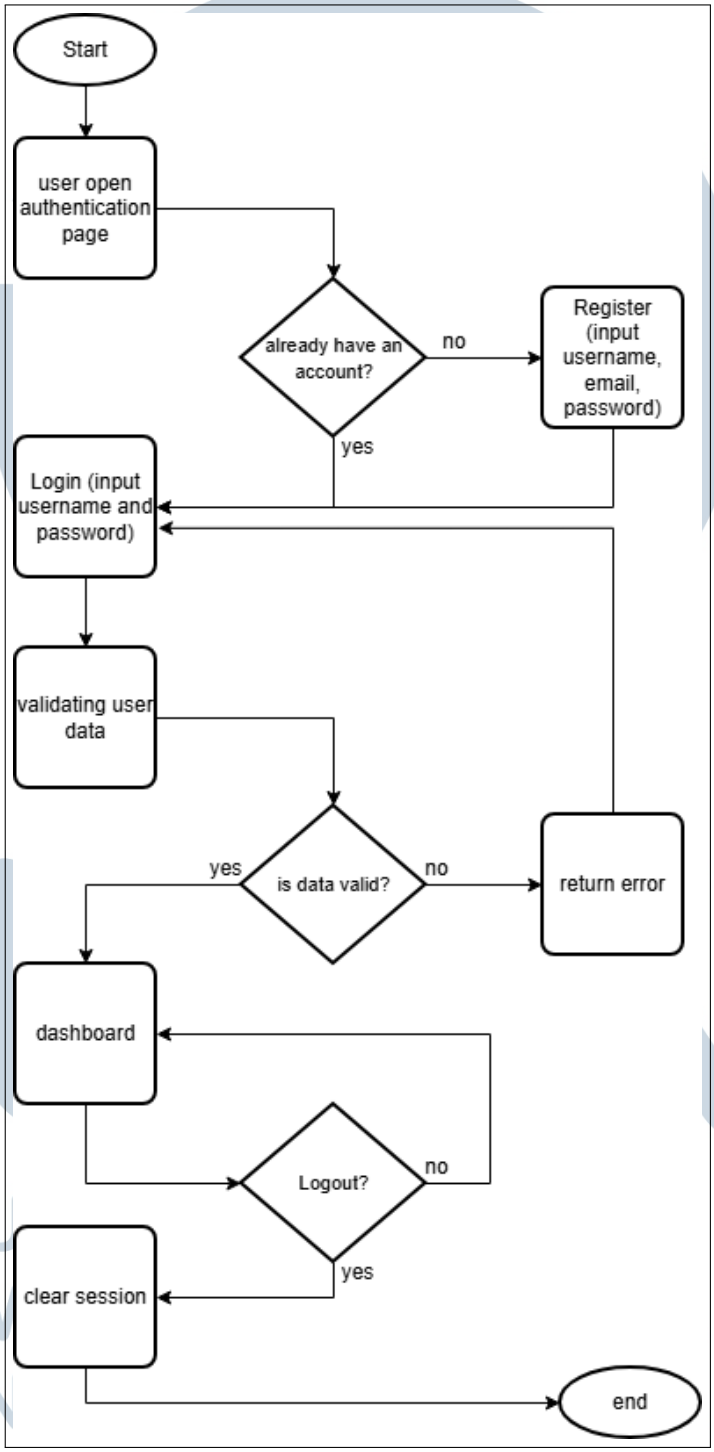
Pada gambar 3.2 *use case diagram* menggambarkan interaksi antara aktor dengan sistem *Job Vacancies System*. Sistem ini memiliki dua aktor utama, yaitu *Job Seeker* dan *Employer*, yang masing-masing memiliki hak akses dan fitur yang berbeda sesuai dengan perannya. *Use case diagram* ini menunjukkan bahwa sistem lowongan pekerjaan dirancang dengan konsep *role-based access*, di mana setiap aktor hanya dapat mengakses fitur sesuai dengan perannya. *job seeker* difokuskan pada aktivitas pencarian dan pelamaran pekerjaan, sedangkan *employer* difokuskan pada pengelolaan lowongan dan kandidat. Dengan pemisahan peran ini, sistem dapat berjalan secara terstruktur, aman, dan sesuai dengan kebutuhan masing-masing pengguna.



Gambar 3.2. *Use case diagram*

Berdasarkan gambar 3.3 yang tertera dibawah, *flowchart* modul autentikasi menjelaskan urutan proses akses *user* ke dalam sistem, mulai dari pembuatan akun hingga keluarnya *user* dari aplikasi. Proses dimulai ketika *user* mengunjungi halaman autentikasi. Jika *user* belum memiliki akun, sistem akan mengarahkan *user* ke halaman pendaftaran. Pada tahap tersebut, *user* diminta mengisi informasi akun seperti *username* dan *password*. Data yang telah dimasukkan kemudian disimpan dalam *database*. Setelah proses registrasi selesai, *user* kembali ke halaman *login* untuk melakukan proses autentikasi. Jika *user* sudah memiliki akun, sistem akan memproses *login* dengan membandingkan *username* dan *password* yang dimasukkan dengan data yang tersimpan di *database*. Apabila data tidak cocok, sistem menampilkan pesan kesalahan. Namun, jika data valid, sistem akan membuat sesi autentikasi berupa *token*, lalu mengarahkan *user* ke halaman utama sesuai dengan peran (*role*) yang dimiliki, yaitu *user* atau *employer*. Selama sesi autentikasi masih aktif, *user* dapat mengakses semua fitur yang diizinkan oleh

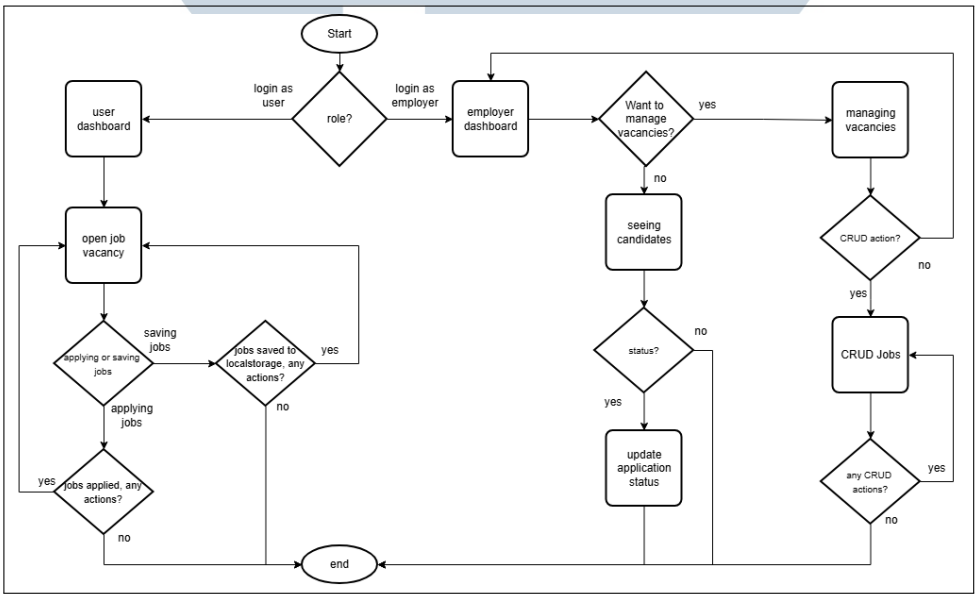
sistem. Ketika *user* memilih untuk *logout*, sistem akan mengakhiri sesi autentikasi, sehingga akses ke sistem terputus dan *user* kembali ke halaman awal.



Gambar 3.3. Alur autentikasi

Berdasarkan gambar 3.4 di bawah ini, *flowchart* modul lamaran pekerjaan

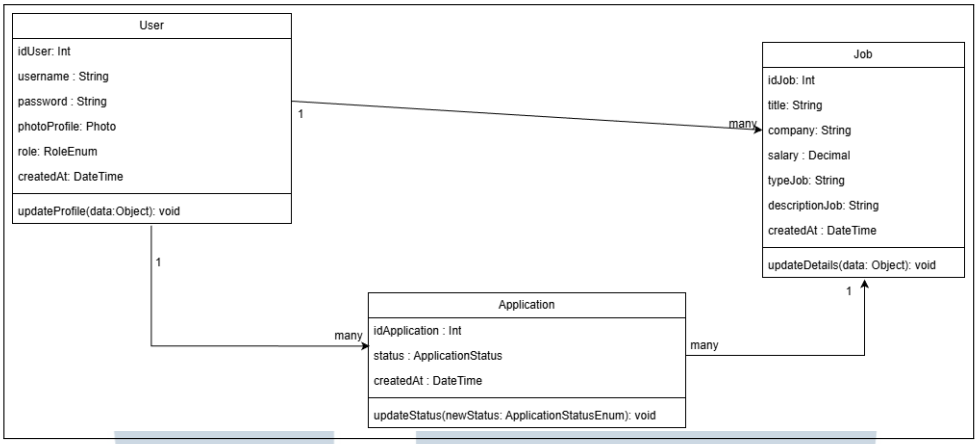
menunjukkan urutan interaksi antara *user* sebagai pelamar dan *employer* sebagai pengelola rekrutmen. Proses dimulai dengan memilih *role* sebagai *user* atau *employer* yang kemudian masuk ke *dashboard* masing-masing *role*. Selanjutnya, *user* memiliki fitur untuk memilih salah satu lowongan dan melakukan tindakan menyimpan atau melamar lowongan. Ketika tindakan tersebut dilakukan, sistem akan menyimpan informasi ke dalam *database* dan *local storage*. Setelah melakukan proses pelamaran atau menyimpan lowongan dan tidak ingin melanjutkan proses pendaftaran, maka alur tersebut berakhir. Setelah data lamaran disimpan, *employer* dapat mengakses halaman kandidat untuk melihat daftar pelamar yang mengajukan diri pada lowongan tertentu. Pada tahap ini, *employer* memiliki hak untuk memperbarui status lamaran, seperti diterima, ditolak, atau sedang dalam proses seleksi. Alur ini menampilkan proses rekrutmen yang terintegrasi, mulai dari pengajuan lamaran oleh *user* hingga pengelolaan kandidat oleh *employer* dalam satu sistem yang terstruktur.



Gambar 3.4. Alur *job vacancy*

Pada sisi *employer*, proses dimulai dengan diberikan pilihan, yaitu melihat *applicants* atau melakukan *CRUD jobs* (*Create, Read, Update, Delete*). Jika memilih *CRUD*, maka *employer* dapat menambahkan lowongan, melihat lowongan, memperbarui lowongan, dan menghapus lowongan. Namun, jika *employer* memilih fitur *seeing candidates*, maka *employer* dapat memperbarui status lamaran dan mengirimkan kembali informasi tersebut ke akun pelamar. Alur ini memastikan bahwa semua proses pengelolaan lowongan pekerjaan dilakukan secara terkontrol,

terstruktur, dan hanya dapat diakses oleh pihak yang memiliki wewenang.



Gambar 3.5. *Class Diagram*

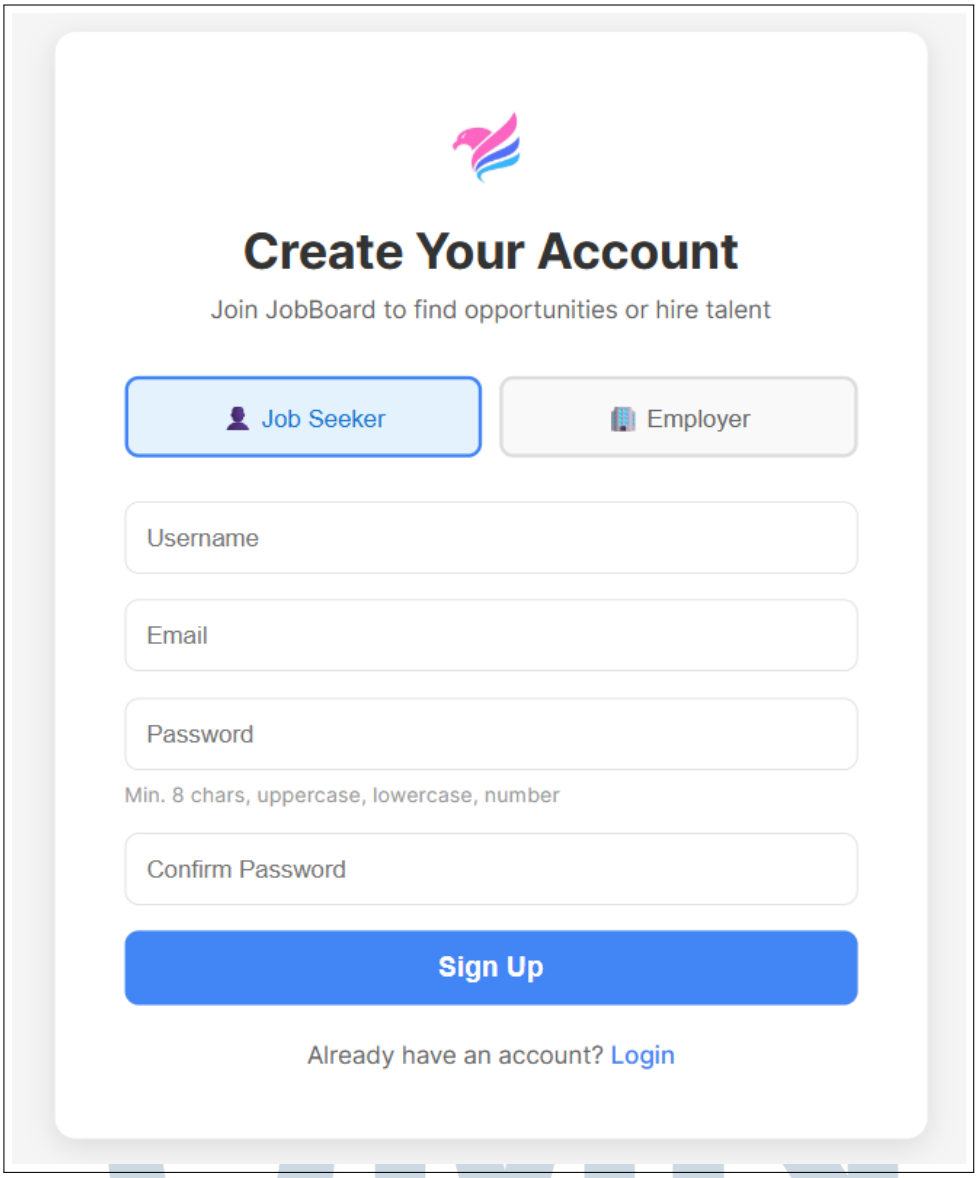
Pada Gambar 3.5, menunjukkan rancangan struktur data statis dari sistem lowongan pekerjaan dalam bentuk *class diagram*, yang terdiri dari tiga entitas utama yang saling berelasi. Entitas pertama adalah kelas *user*, yang berfungsi untuk mengelola seluruh data pengguna sistem, termasuk atribut penting seperti kredensial *login* (*username, password*) dan atribut role untuk membedakan jenis pengguna. Entitas kedua adalah kelas *job*, yang merepresentasikan detail data dari setiap lowongan pekerjaan yang dipublikasikan, mencakup informasi seperti judul posisi, nama perusahaan, dan gaji yang ditawarkan. Entitas ketiga adalah kelas *application*, yang digunakan untuk mencatat setiap lamaran kerja yang diajukan, dengan atribut seperti status untuk melacak progres lamaran tersebut.Diagram ini juga memperlihatkan bagaimana ketiga entitas tersebut berinteraksi melalui relasi asosiasi yang jelas:

- Terdapat hubungan satu-ke-banyak (1-to-*) antara *user* dan *job*, yang menunjukkan bahwa satu pengguna (yang berperan sebagai pemberi kerja atau *employer*) dapat memposting banyak lowongan pekerjaan.
- Terdapat hubungan satu-ke-banyak (1-to-*) antara *user* dan *application*, yang mengartikan bahwa satu pengguna (yang berperan sebagai pelamar) dapat mengirimkan banyak lamaran.
- Terdapat hubungan satu-ke-banyak (1-to-*) antara *job* dan *application*, yang menjelaskan bahwa satu lowongan pekerjaan spesifik dapat menerima banyak lamaran dari berbagai pengguna.

Struktur ini memastikan integritas pada data sistem, di mana setiap aktivitas lamaran (*Application*) pasti terhubung secara langsung dengan satu pelamar (*User*) yang mengajukannya dan satu lowongan (*Job*) yang dituju.

3.3.3 Laman *signup*

Laman *website* yang meliputi dua peran (*multi-role*) yaitu *user* dan *employer*. Fitur untuk *user* meliputi *dashboard*, *login page*, *register page*, *saving jobs, applying jobs*, dan *logout*; sementara fitur untuk *employer* meliputi *dashboard*, *login page*, *register page, posting jobs, logout*. Seluruh komponen UI difokuskan untuk kemudahan akses, konsistensi *layout*, pemilihan warna profesional, serta kejelasan struktur informasi. Tampilan halaman *register* dapat dilihat pada gambar 3.6 dibawah ini.

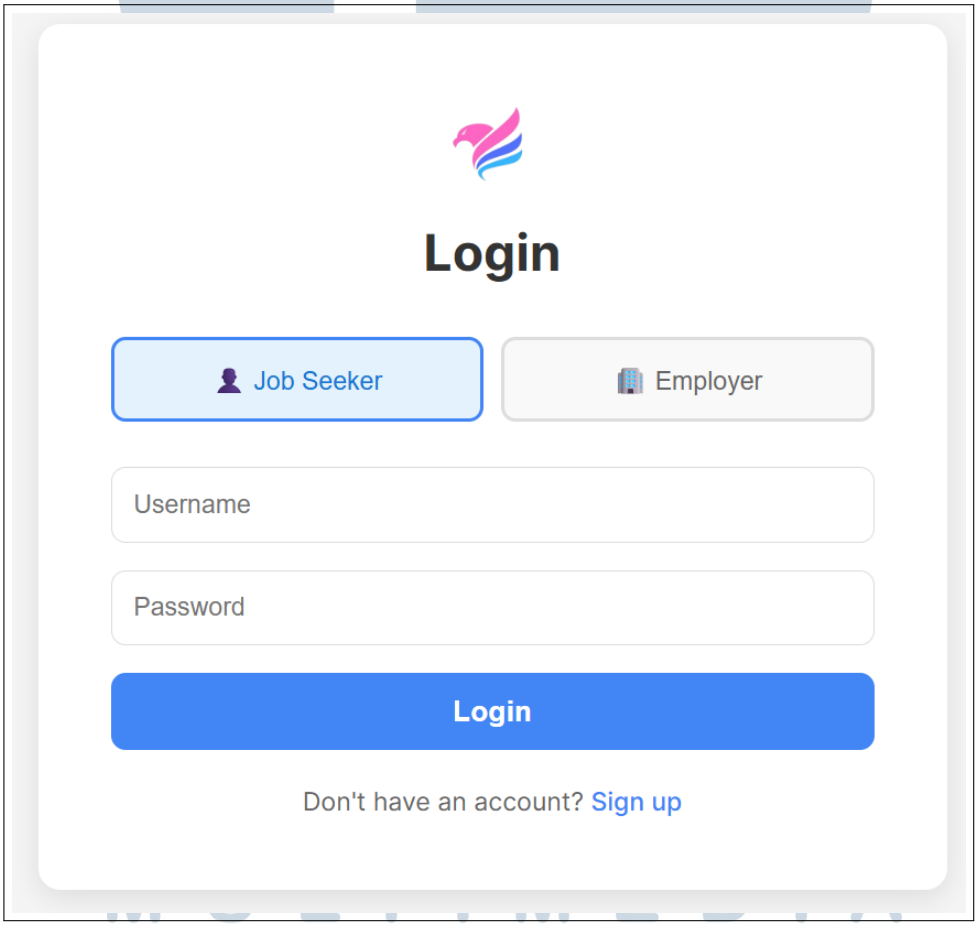


Gambar 3.6. Laman *Register*

Pada tampilan halaman *register* diatas, menunjukkan *field* untuk melakukan *registrasi* akun. *User* maupun *employer* yang ingin *registrasi* akun wajib mengisi *username, email*, dan *password* agar terbaca di sistem. Namun jika sudah memiliki akun, maka baik *user* maupun *employer* bisa melakukan proses *login* sesuai dengan data yang sudah diinput.

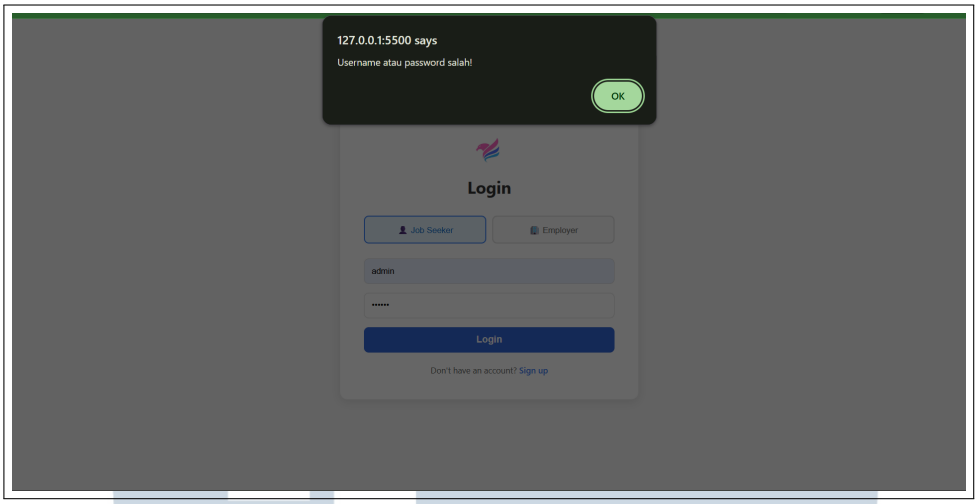
3.3.4 Laman *login*

Pada tampilan *login* yang terdapat pada gambar 3.7, *user* maupun *employer* perlu mengisi *username* dan *password* untuk mengakses *website*. Tujuannya untuk memberikan akses cepat dan langsung bagi *user* maupun *employer* yang telah terdaftar untuk masuk ke *website*. Halaman *login* merupakan tampilan pertama kali saat akan mengakses *website*. Halaman ini berfungsi untuk melakukan verifikasi data dengan memasukkan *username* dan *password* sebelum memberikan akses ke fitur-fitur dalam sistem. Selain sebagai pintu masuk utama, halaman *login* juga berperan penting dalam menjaga keamanan data dan memastikan hanya yang terdaftar yang dapat mengakses ke dalam *website*.



Gambar 3.7. Laman *Login*

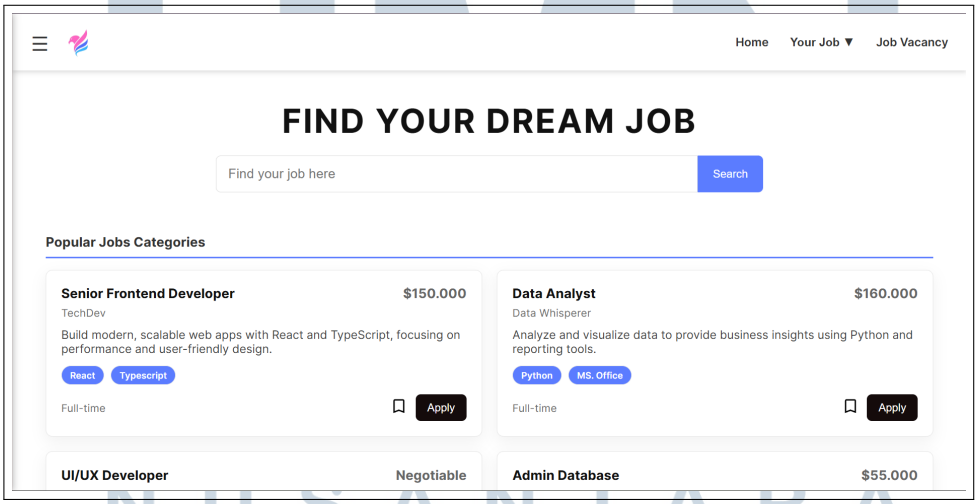
Apabila terdapat kesalahan pada *username* atau *password*, maka sistem akan menampilkan *alert* seperti pada gambar 3.8 dan mewajibkan untuk memasukkan nilai atau data yang benar.



Gambar 3.8. *Alert* ketika *username* dan *password* salah

3.3.5 Laman Utama

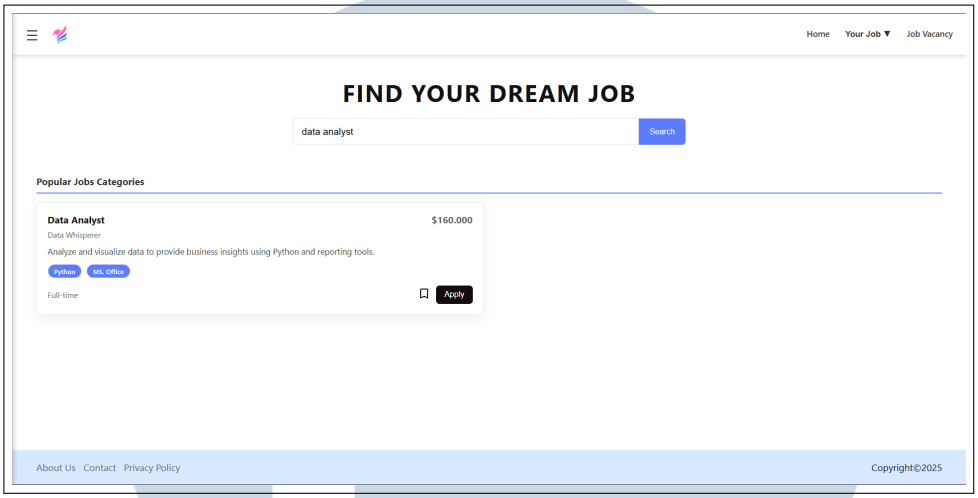
Pada gambar 3.9 yang menampilkan halaman utama untuk *user* yang menampilkan pekerjaan-pekerjaan yang tersedia. Tipe pekerjaan bervariasi, mulai dari jenis pekerjaan seperti *full-time, part-time,* dan *freelance*; kemampuan apa yang diminta dari pekerjaan tersebut seperti Python, PHP, dan sebagainya; hingga gaji yang bervariasi. Bagian ini dirancang secara responsif agar user dapat langsung menemukan apa yang dicari.



Gambar 3.9. Laman *dashboard* (*user*)

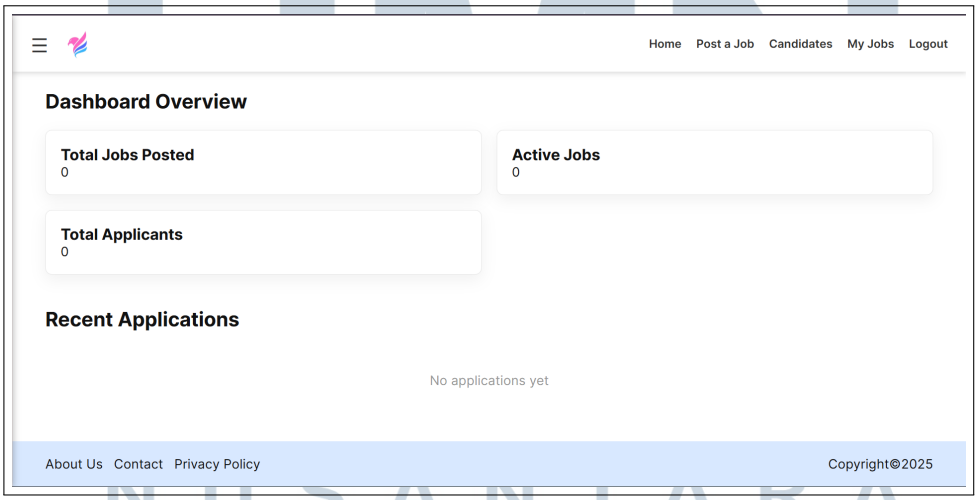
Pada gambar 3.10, *user* dapat mencari pekerjaan sesuai dengan preferensi masing-masing. Setelah *user* melakukan pencarian, maka akan muncul hasil yang

relevan dengan kata kunci yang dimasukkan. Hasil yang diterima akan ditampilkan di *dashboard user* secara *real-time*.



Gambar 3.10. Laman pencarian pekerjaan

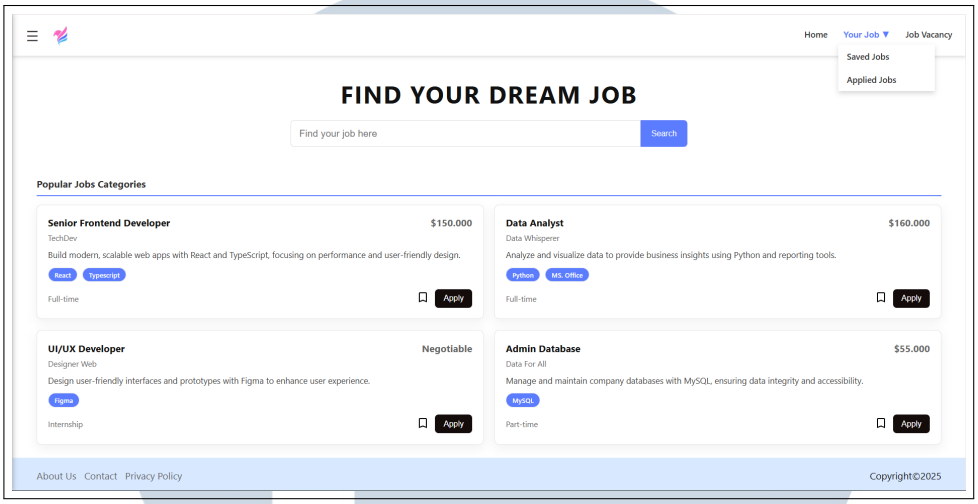
Pada gambar 3.11, menampilkan *dashboard* untuk *employer* yang berisi total pekerjaan yang diposting (*total job posted*), pekerjaan yang masih aktif (*active jobs*), jumlah pelamar (*total apllicants*), dan lamaran terbaru (*recent applications*). Dirancang lebih sederhana agar tujuan dari *website* nya jelas dan tidak membingungkan. Ketika *employer* memposting pekerjaan, maka jumlah dari *total jobs posted* dan *active jobs* akan bertambah secara *real-time*.



Gambar 3.11. Laman *dashboard* (*employer*)

Pada gambar 3.12 menampilkan *dropdown* yang berisi pekerjaan yang sudah disimpan oleh *user* dan pekerjaan yang sudah berhasil di-*apply* oleh *user*. Desain

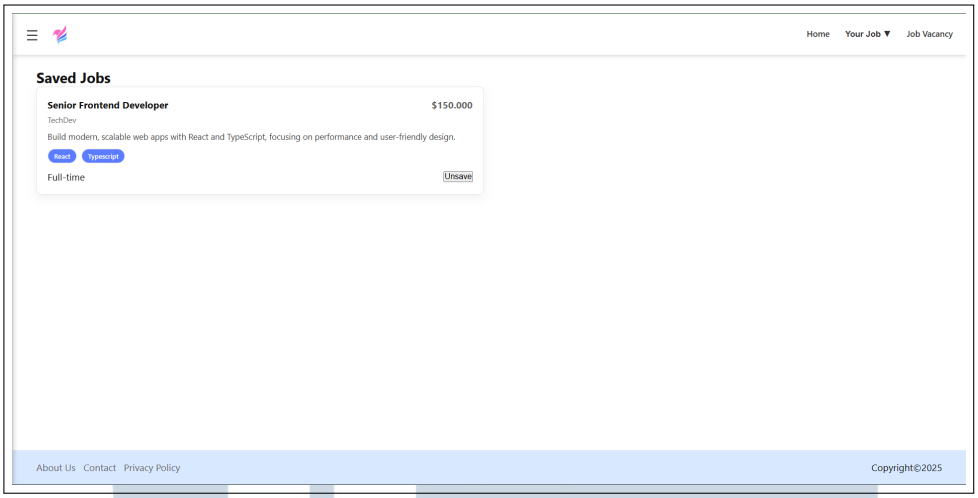
dropdown menampilkan keefektifan *layout* karena memiliki kemiripan dalam fitur sehingga lebih baik.



Gambar 3.12. *Dropdown* untuk *saved jobs* dan *applied jobs*

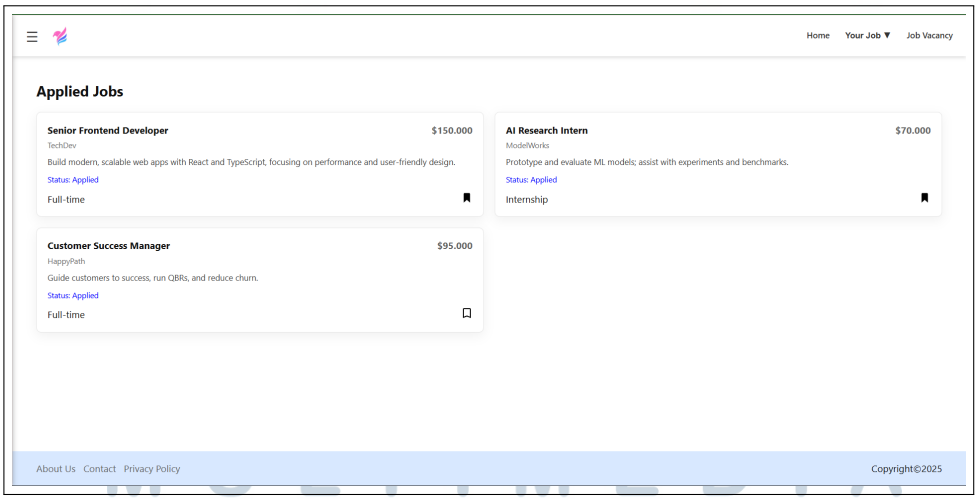
3.3.6 Laman *Saved Jobs* dan *Applied Jobs*

Pada fitur *saved jobs* yang ditunjukkan pada gambar 3.13, user dapat menyimpan lowongan pekerjaan yang diminati namun belum dapat dilamar karena belum memenuhi persyaratan tertentu, seperti keterampilan (*skills*) yang dibutuhkan. Fitur ini memungkinkan *user* untuk mengakses kembali lowongan tersebut di kemudian hari tanpa harus melakukan pencarian ulang, sehingga proses perencanaan dan persiapan karier menjadi lebih terstruktur dan efisien. *User* juga bisa untuk menghapus daftar pekerjaan yang disimpan dan menyimpan pekerjaan yang lain.



Gambar 3.13. Fitur *saved jobs*

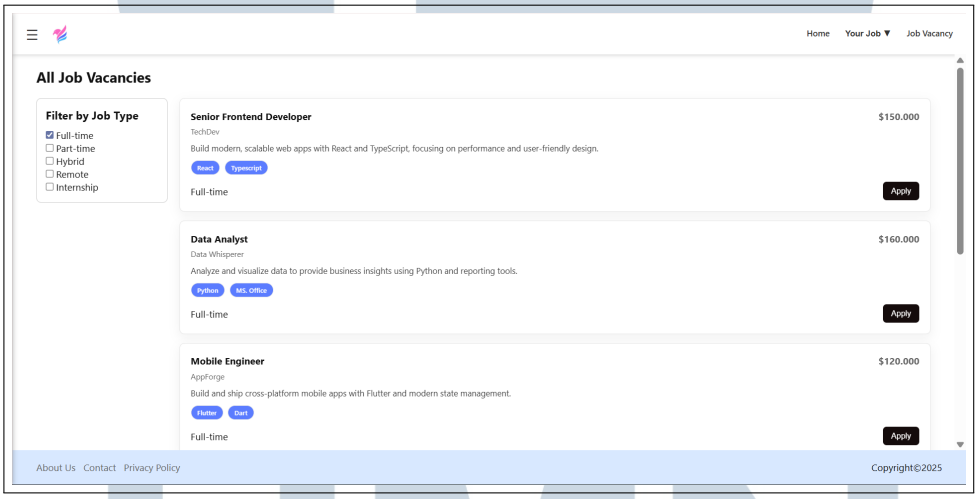
Pada fitur *applied jobs* yang ditunjukkan pada gambar 3.14, sistem menampilkan daftar pekerjaan yang telah dilamar oleh *user*. Fitur ini berfungsi untuk membantu *user* memantau status lamaran yang telah diajukan, sehingga *user* dapat mengetahui progres seleksi tanpa harus mengingat atau mencari ulang lowongan yang sama. Dengan adanya *applied jobs*, user dapat mengelola riwayat lamaran secara terstruktur, menghindari pengiriman lamaran ganda, serta memiliki referensi pekerjaan yang sedang atau telah diproses oleh perusahaan.



Gambar 3.14. Fitur *applied jobs*

3.3.7 Laman *Job Vacancy*

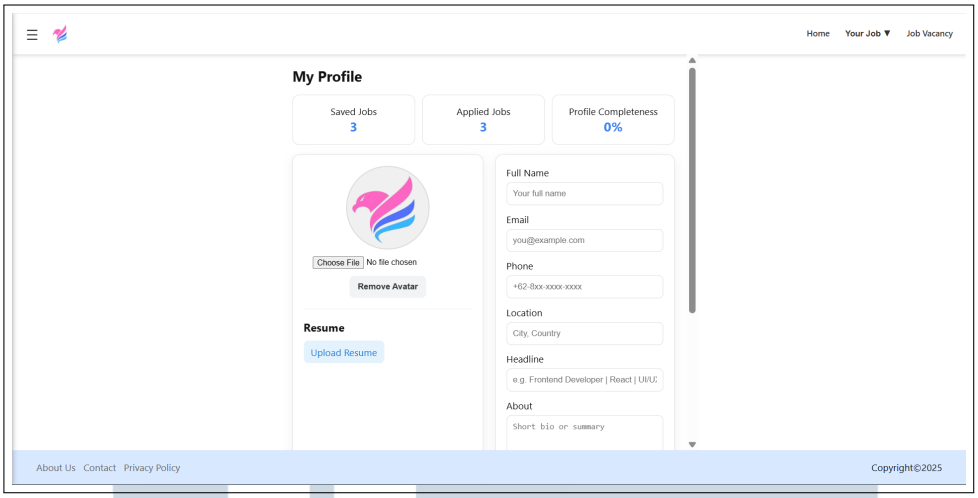
Pada laman job vacancy yang ditampilkan pada gambar 3.15 sistem menyediakan daftar lowongan pekerjaan yang tersedia dari berbagai perusahaan. Halaman ini mirip dengan bagian *dashboard*, namun yang membedakan adalah fitur *filter* berdasarkan jenis pekerjaan seperti *full-time, part-time, hybrid, remote*, dan *internship* sehingga memudahkan *user* dalam menyesuaikan pencarian pekerjaan sesuai dengan preferensi dan kebutuhan. *User* dapat langsung melamar pekerjaan melalui tombol *apply* yang tersedia pada setiap lowongan. Dengan adanya fitur ini, proses pencarian dan pelamaran kerja menjadi lebih terstruktur, efisien, dan mudah diakses oleh user.



Gambar 3.15. Fitur *job vacancy*

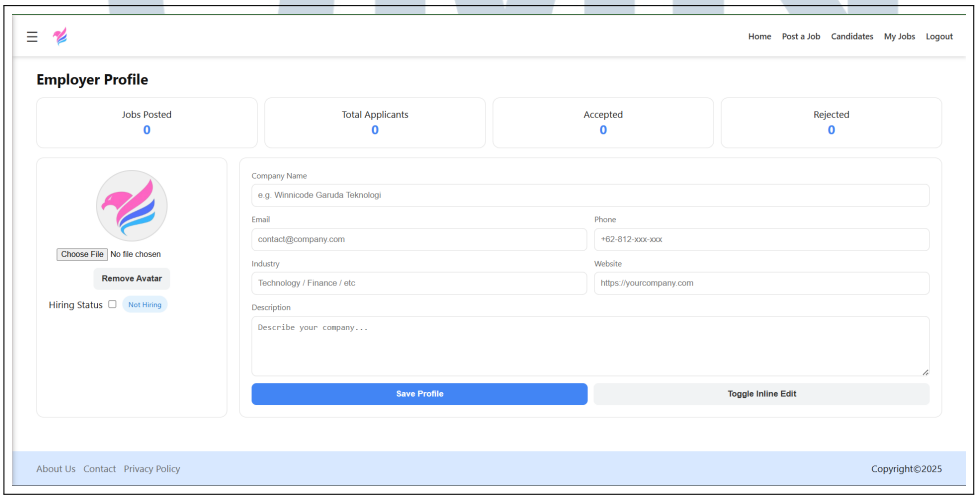
3.3.8 Laman Profil

Pada halaman profil yang ditunjukkan pada gambar 3.16, sistem menyediakan fasilitas bagi *user* untuk mengelola data pribadi yang digunakan dalam proses pencarian kerja. Halaman ini menampilkan informasi dasar user seperti foto profil, nama lengkap, *email*, nomor telepon, lokasi, serta *headline*. Selain itu, *user* juga dapat mengunggah *resume* dalam bentuk *file* sebagai dokumen pendukung saat melamar pekerjaan. Pada bagian atas halaman, sistem menampilkan ringkasan aktivitas *user* berupa presentase kelengkapan profil, pekerjaan yang disimpan dan pekerjaan yang sudah di-apply. Dengan adanya fitur ini, *user* dapat memastikan bahwa informasi yang dimiliki selalu diperbarui sehingga meningkatkan peluang untuk diterima pada lowongan pekerjaan yang dilamar.



Gambar 3.16. Fitur profil (*user*)

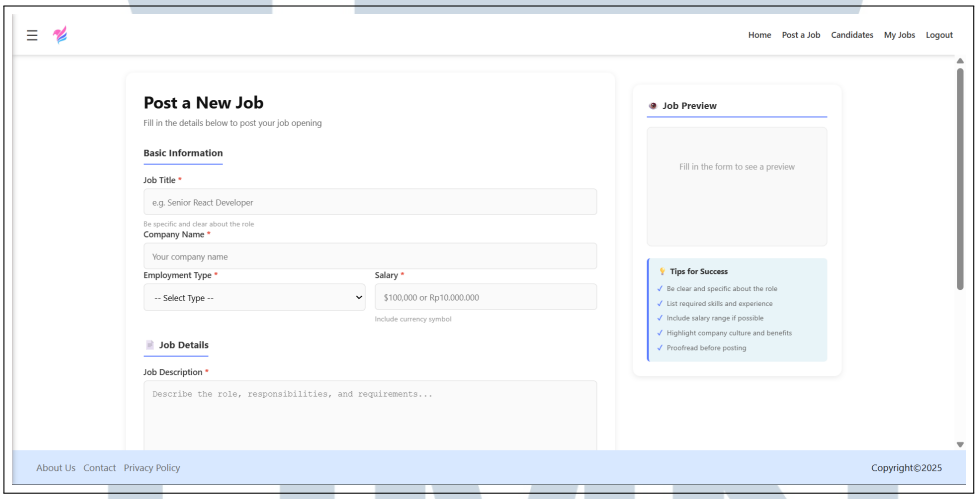
Pada *role employer* juga memiliki fitur profil sendiri seperti yang ada pada gambar 3.17. Pada halaman ini, *employer* dapat mengelola dan memperbarui data perusahaan secara efektif. Halaman ini menampilkan ringkasan aktivitas perusahaan, seperti jumlah lowongan yang telah diposting, total pelamar, jumlah pelamar yang diterima, dan jumlah pelamar yang ditolak. Selain itu, perusahaan dapat mengatur identitas perusahaan yang meliputi logo, nama perusahaan, *email*, nomor telepon, industri tempat perusahaan, *website*, serta deskripsi singkat mengenai perusahaan. Tersedia pula status *hiring* untuk menandai apakah perusahaan sedang membuka lowongan atau tidak. Seluruh perubahan data dapat disimpan melalui tombol *save profile*, sehingga informasi perusahaan yang ditampilkan kepada pelamar selalu akurat dan terkini.



Gambar 3.17. Fitur profil (*employer*)

3.3.9 Fitur *Posting Jobs*(*employer*)

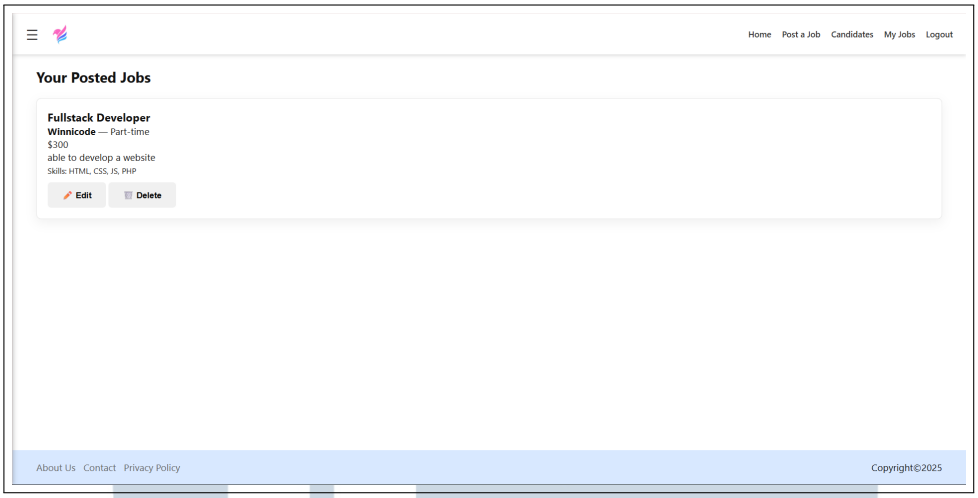
Fitur *posting jobs* pada gambar 3.18 digunakan oleh perusahaan untuk menambahkan lowongan pekerjaan baru ke dalam sistem. Pada halaman ini, perusahaan diwajibkan mengisi informasi dasar pekerjaan seperti judul pekerjaan, nama perusahaan, tipe pekerjaan, dan gaji. Selanjutnya, perusahaan dapat menambahkan detail pekerjaan berupa deskripsi tugas dan keterampilan yang dibutuhkan. Sistem juga menyediakan tampilan job preview secara *real-time* sehingga perusahaan dapat melihat apa saja yang sekiranya bisa disempurnakan sebelum dipublikasikan. Dengan adanya fitur ini, proses pembuatan dan publikasi lowongan kerja menjadi lebih terstruktur, jelas, dan meminimalkan kesalahan informasi.



Gambar 3.18. Fitur *posting jobs*

3.3.10 Fitur My Jobs (*employer*)

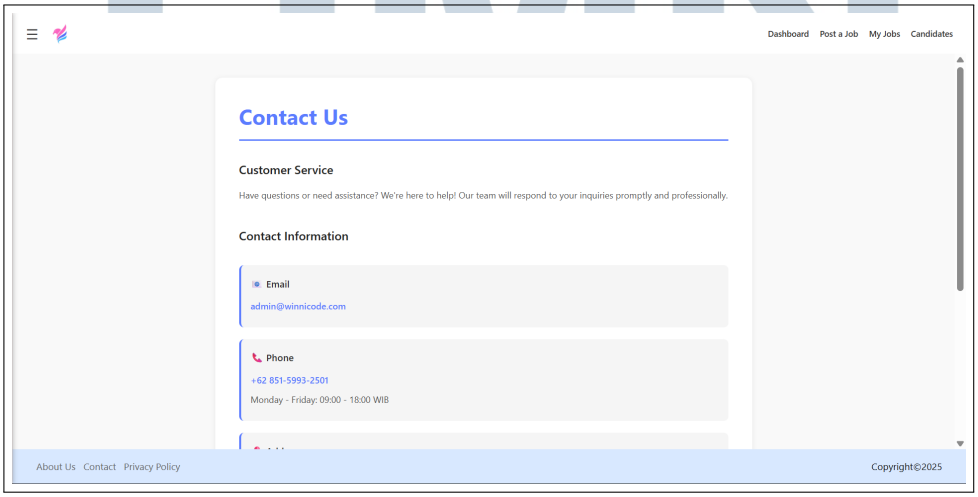
Pada gambar 3.19 menampilkan fitur employer untuk melihat pekerjaan apa yang selama ini sudah dibuat. Pada halaman ini, setiap lowongan ditampilkan secara ringkas, mencakup posisi, nama perusahaan, tipe pekerjaan, gaji, deskripsi singkat pekerjaan, serta daftar keterampilan yang dibutuhkan. Perusahaan juga diberikan kontrol penuh untuk mengelola lowongan tersebut melalui tombol *edit* untuk memperbarui informasi pekerjaan dan *delete* untuk menghapus lowongan yang sudah tidak dibutuhkan. Dengan adanya fitur ini, perusahaan dapat memantau dan mengelola lowongan kerja secara efisien dalam satu halaman.



Gambar 3.19. Fitur *my jobs*

3.3.11 Laman *Contact*

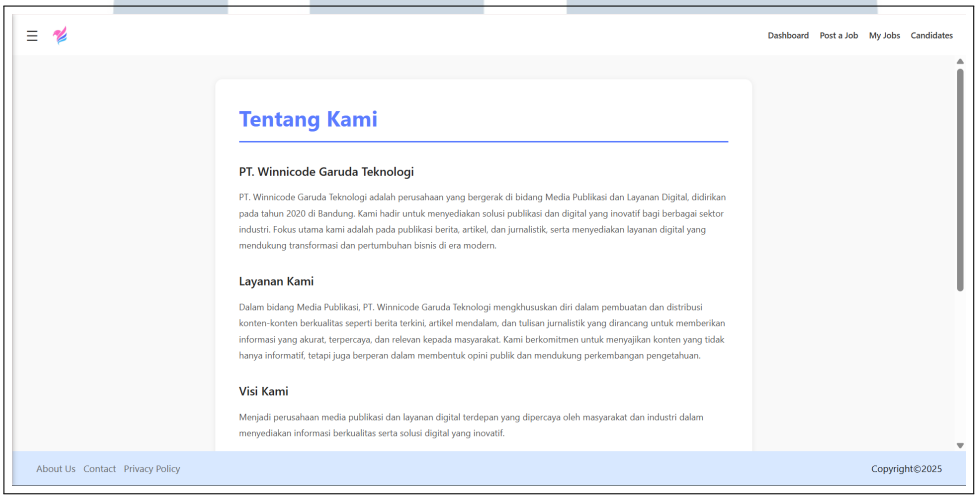
Pada Gambar 3.20 dibawah, ditunjukkan halaman *Contact Us* yang berfungsi sebagai sarana komunikasi antara user dan pengelola aplikasi. Halaman ini menyediakan informasi kontak resmi perusahaan seperti email, nomor yang bisa dihubungi, serta jam operasional layanan. Fitur ini memungkinkan user baik user maupun *employer*, untuk menyampaikan pertanyaan, kendala teknis, maupun kebutuhan bantuan secara langsung. Dengan adanya halaman ini, aplikasi tidak hanya berfokus pada fungsi pencarian dan penyediaan lowongan kerja, tetapi juga memberikan dukungan layanan yang profesional dan responsif



Gambar 3.20. Fitur *contact*

3.3.12 Laman *About*

Pada gambar 3.21 diperlihatkan halaman About Us yang menjelaskan profil dan identitas perusahaan pengelola aplikasi. Halaman ini memuat informasi mengenai latar belakang perusahaan, bidang usaha, layanan yang disediakan, serta visi perusahaan. Informasi ini bertujuan untuk membangun kepercayaan user dengan memberikan gambaran yang jelas mengenai pihak yang bertanggung jawab atas aplikasi. Selain itu, halaman ini juga memperkuat kredibilitas platform sebagai media digital yang profesional dan terpercaya.

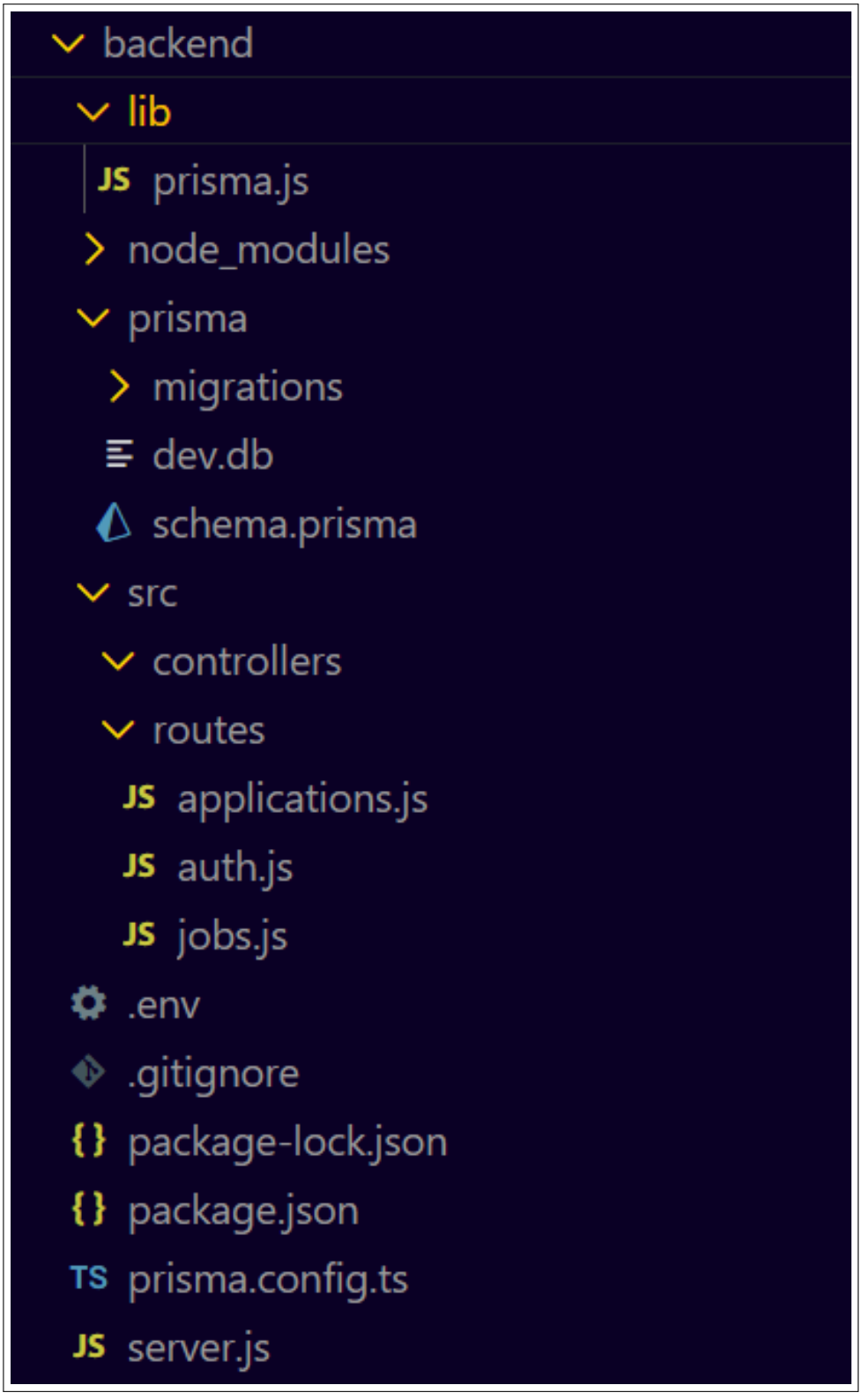


Gambar 3.21. Fitur *about*

3.3.13 Laman *Backend*

Struktur backend pada gambar 3.22 menunjukkan arsitektur aplikasi server yang dibangun menggunakan Node.js, Express, dan Prisma sebagai ORM. Folder lib berisi file prisma.js yang berfungsi sebagai inisialisasi dan konfigurasi koneksi Prisma Client ke database. Folder prisma menyimpan konfigurasi utama database, termasuk file schema.prisma sebagai definisi skema tabel, relasi, dan tipe data, serta folder migrations yang mencatat riwayat perubahan struktur database. File dev.db digunakan sebagai database development berbasis SQLite. Seluruh logika utama aplikasi ditempatkan di dalam folder src, yang dipisahkan secara modular ke dalam folder controllers untuk logika bisnis dan routes untuk pendefinisian endpoint API (Application Programming Interface) seperti auth.js, jobs.js, dan applications.js. Pemisahan ini bertujuan agar kode lebih terstruktur, mudah dipelihara, dan scalable. File .env digunakan untuk menyimpan variabel lingkungan sensitif, sedangkan

server.js berperan sebagai entry point aplikasi backend.



Gambar 3.22. Struktur *backend*

Gambar 3.23 dibawah menunjukkan implementasi file server.js yang berfungsi sebagai pusat konfigurasi Express server. Aplikasi dimulai dengan mengimpor library utama seperti express dan cors, serta mengimpor route khusus pekerjaan dari file jobs.js. Middleware cors() digunakan untuk mengatur Cross-Origin Resource Sharing agar frontend dapat berkomunikasi dengan backend, sementara json() digunakan untuk memproses request body dalam format JSON (JavaScript Object Notation). Selanjutnya, route /api/jobs dihubungkan dengan jobRoutes sebagai pengatur seluruh proses pengambilan dan pengelolaan data lowongan pekerjaan. Terakhir, server dijalankan pada port 3000 (http://localhost:3000) dan menampilkan pesan di console sebagai indikator bahwa server telah aktif dan siap menerima request dari sisi client.

```
import express, { json } from "express";
import cors from "cors";

import jobRoutes from "./src/routes/jobs.js";

const app = express();
app.use(cors());
app.use(json());

// health check (BIAR GA "Cannot GET /")
app.get("/", (req, res) => {
  res.send("🔥 API is running");
});

app.use("/api/jobs", jobRoutes);

app.listen(3000, () => {
  console.log("🔥 Server running on http://localhost:3000");
});
```

Gambar 3.23. Implementasi *server.js*

3.3.14 Pengujian Sistem (*Blackbox*)

Tabel 3.2. Hasil Pengujian Fungsional Sistem

No	Fitur	Skenario	Input	Output yang Diharapkan	Hasil
1	Login	Pengguna memasukkan data login yang valid	Username dan password valid	Sistem berhasil melakukan autentikasi dan menampilkan halaman utama	Valid
2	Login	Pengguna memasukkan password yang salah	Username valid, password salah	Sistem menampilkan pesan kesalahan (*alert*)	Valid
3	Registrasi	Pengguna melakukan pendaftaran akun baru	Username, email, dan password	Data akun tersimpan dan pengguna diarahkan ke halaman login	Valid
4	Logout	Pengguna melakukan proses keluar dari sistem	Klik tombol logout	Sistem mengakhiri sesi dan kembali ke halaman awal	Valid

3.4 Kendala dan Solusi yang Ditemukan

Selama proses pengembangan *website* lowongan pekerjaan, terdapat beberapa kendala yang ditemukan selama proses pengembangan sistem.

1. Kendala
    - Koordinasi pengembangan sistem

      Proses pengembangan sistem dilakukan secara daring sehingga koordinasi teknis terkait kebutuhan sistem, alur pengembangan, dan penyesuaian fitur tidak selalu dapat dilakukan secara langsung. Kondisi ini berdampak pada perlunya penyesuaian ulang terhadap beberapa bagian sistem agar sesuai dengan kebutuhan yang diharapkan.
    - Kompleksitas pengembangan backend

Pada tahap pengembangan backend, terdapat kendala dalam mengimplementasikan alur sistem yang melibatkan autentikasi pengguna, pengelolaan data lowongan, serta proses penyimpanan dan pengambilan data dari database. Hal ini disebabkan oleh kompleksitas logika sistem backend yang membutuhkan pemahaman struktur data dan alur proses yang terintegrasi.

2. Solusi

- Penyesuaian alur pengembangan sistem

  Dilakukan penyesuaian terhadap alur pengembangan dengan menyusun flowchart dan struktur sistem yang lebih terorganisir agar proses implementasi fitur dapat dilakukan secara bertahap dan terarah.

- Pendalaman teknis pengembangan backend

  Pengembangan backend dilakukan dengan memanfaatkan berbagai referensi teknis seperti dokumentasi resmi, tutorial pengembangan, serta eksplorasi langsung melalui proses implementasi dan pengujian sistem. Pendekatan ini membantu dalam memahami alur kerja backend serta integrasi antara frontend dan database.