BAB IV
ANALISIS DAN PERANCANGAN SISTEM

Bab ini menjelaskan hasil analisis dan perancangan Sistem Informasi Rekrutmen pada Career Development Center (CDC) Universitas Andalas. Pada tahap analisis, proses bisnis yang sedang berjalan dan yang diusulkan dimodelkan menggunakan Business Process *Model* and Notation (BPMN), dilanjutkan dengan analisis kebutuhan fungsional setiap aktor serta pemodelan *use case diagram* beserta deskripsi setiap *use case* dalam bentuk skenario. Pada tahap perancangan, sistem digambarkan melalui arsitektur aplikasi, *sequence diagram*, perancangan basis data yang terdiri atas *Entity Relationship Diagram* (ERD) dan struktur tabel, *class diagram*, serta perancangan antarmuka. Seluruh pemodelan dilakukan menggunakan Unified Modeling Language (UML).

4.1 Analisis Sistem

Tahapan analisis sistem memberikan gambaran bagaimana sistem yang sedang berjalan dan sistem yang diusulkan melalui tools BPMN, serta analisis kebutuhan fungsional, use case diagram, deskripsi tugas aktor, use case scenario dan sequence diagram yang dimodelkan menggunakan Unified Modeling Language (UML).

4.1.1 Analisis Proses Bisnis

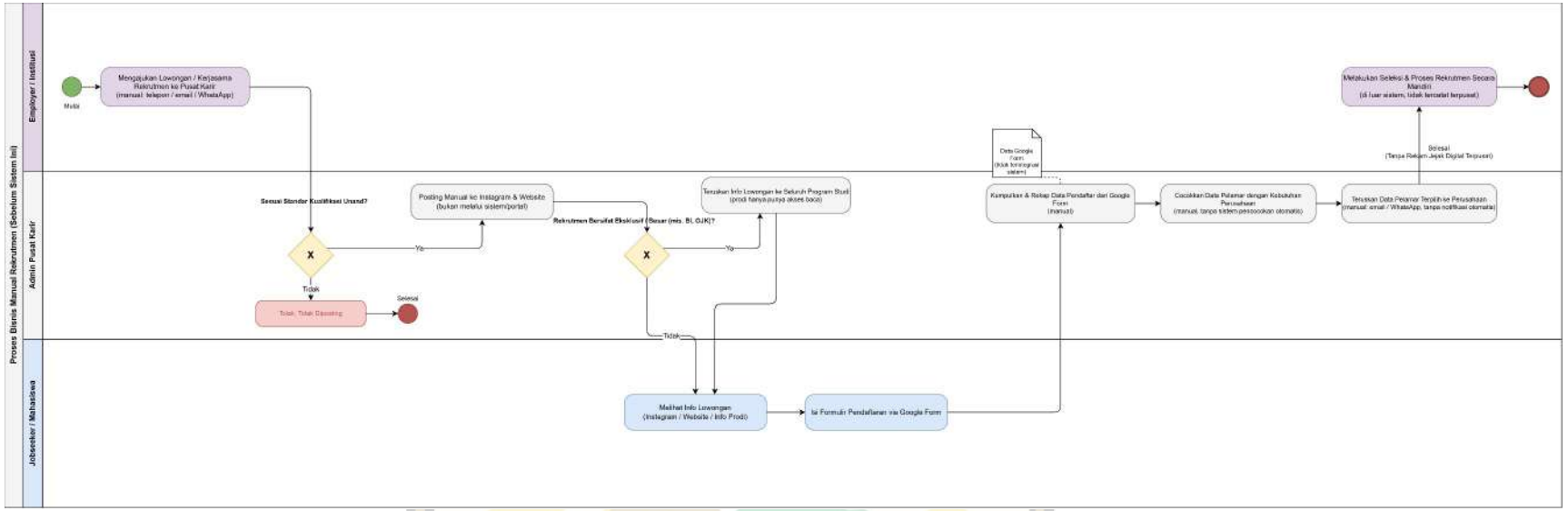
Analisis proses bisnis menggambarkan alur proses rekrutmen yang sedang berjalan pada CDC Unand serta proses yang diusulkan melalui Business Process *Model* and Notation (BPMN). BPMN disusun berdasarkan hasil wawancara dan analisis terhadap proses yang berjalan. Proses yang dimodelkan terdiri atas proses rekrutmen yang sedang berjalan, melamar pekerjaan, mengelola lowongan kerja, dan proses seleksi pelamar yang diusulkan.

Pemodelan proses bisnis pada penelitian ini menggunakan *Business Process Model and Notation* (BPMN) versi 2.0 (OMG, 2013). Setiap proses digambarkan dalam satu *pool* dengan *lane* untuk masing-masing peran, dimulai dari *start event* dan diakhiri pada *end event*, sedangkan setiap percabangan keputusan direpresentasikan menggunakan *gateway*, bukan dengan

mempercabangkan *sequence flow* secara langsung, sesuai kaidah pemodelan BPMN (Silver, 2011).

4.1.1.1 BPMN Proses Rekrutmen yang Sedang Berjalan

Proses rekrutmen yang sedang berjalan dilakukan secara manual oleh tiga aktor, yaitu *Employer*, *Admin* pusat karir, dan *Jobseeker*. BPMN proses rekrutmen yang sedang berjalan dapat dilihat pada Gambar 4.1. Berdasarkan gambar tersebut, penjelasan tahapan proses yang sedang berjalan adalah sebagai berikut:



Gambar 4.1 BPMN Proses Rekrutmen yang Sedang Berjalan

1. *Employer* mengajukan lowongan atau kerja sama rekrutmen kepada pusat karir melalui telepon, *email*, atau WhatsApp.

2. *Admin* memeriksa kesesuaian lowongan dengan standar kualifikasi.

   a. Jika tidak sesuai, pengajuan ditolak dan proses selesai.

   b. Jika sesuai, *Admin* memposting informasi lowongan secara manual ke Instagram dan website.

3. Jika rekrutmen bersifat eksklusif, *Admin* meneruskan informasi lowongan ke seluruh program studi.

4. *Jobseeker* melihat informasi lowongan, kemudian mengisi formulir pendaftaran melalui Google Form.

5. *Admin* merekap data pendaftar dan mencocokkannya dengan kebutuhan perusahaan secara manual.

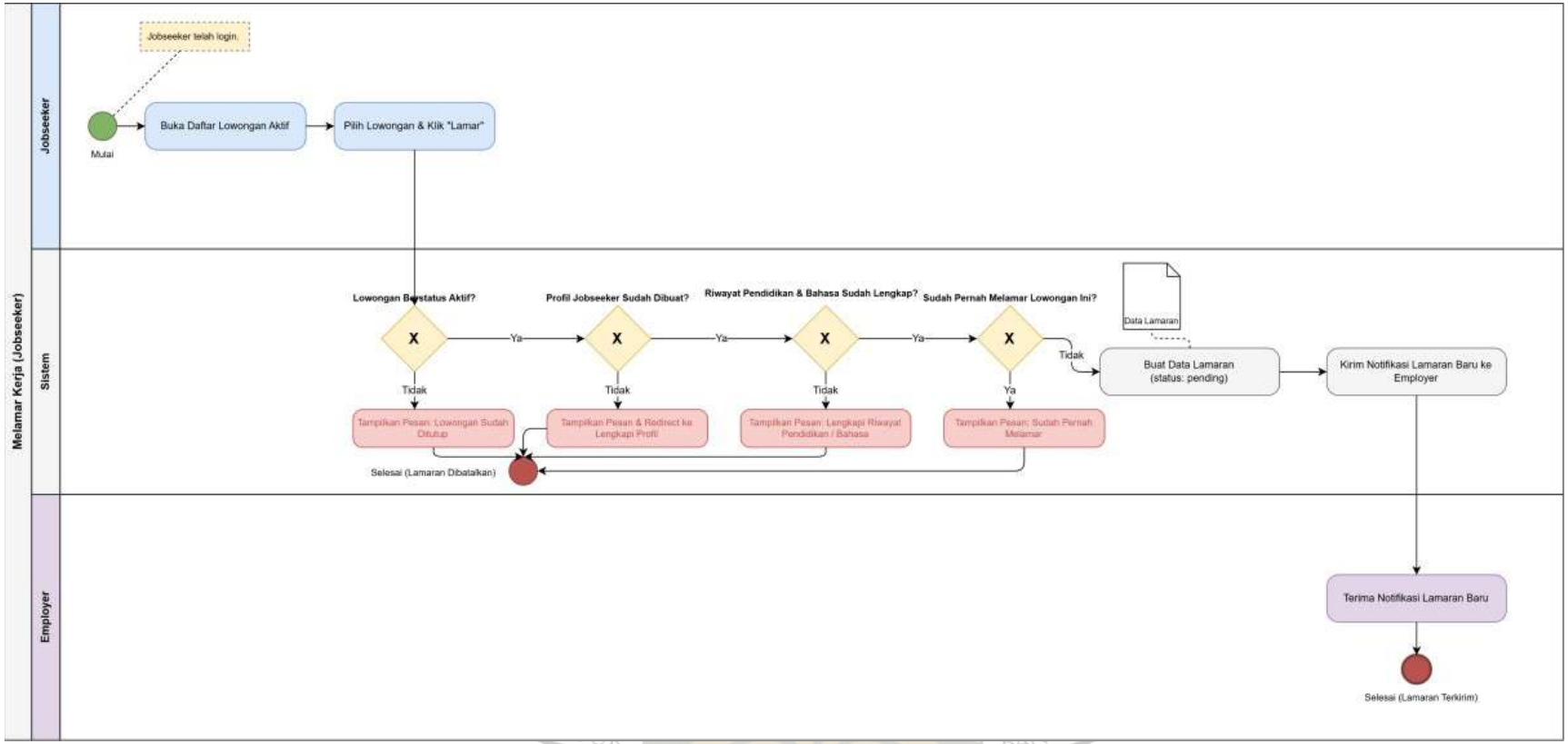
6. *Admin* meneruskan data pelamar terpilih kepada *Employer* melalui *email* atau WhatsApp.

7. *Employer* melakukan proses seleksi secara mandiri di luar sistem.

Proses tersebut tidak memiliki rekam jejak digital yang terpusat, sehingga kemajuan rekrutmen sulit dipantau baik oleh pusat karir maupun oleh pelamar.

4.1.1.2 BPMN Melamar Pekerjaan yang Diusulkan

Proses melamar pekerjaan yang diusulkan dilakukan oleh dua aktor, yaitu *Jobseeker* dan *Employer*, dengan validasi otomatis oleh sistem. BPMN yang diusulkan dapat dilihat pada Gambar 4.2. Penjelasan tahapannya adalah sebagai berikut:



Gambar 4.2 BPMN Melamar Pekerjaan yang Diusulkan

1. *Jobseeker* memilih lowongan dari daftar lowongan aktif dan menekan tombol lamar.

2. Sistem memeriksa status lowongan, kelengkapan profil, dan riwayat lamaran *Jobseeker*.

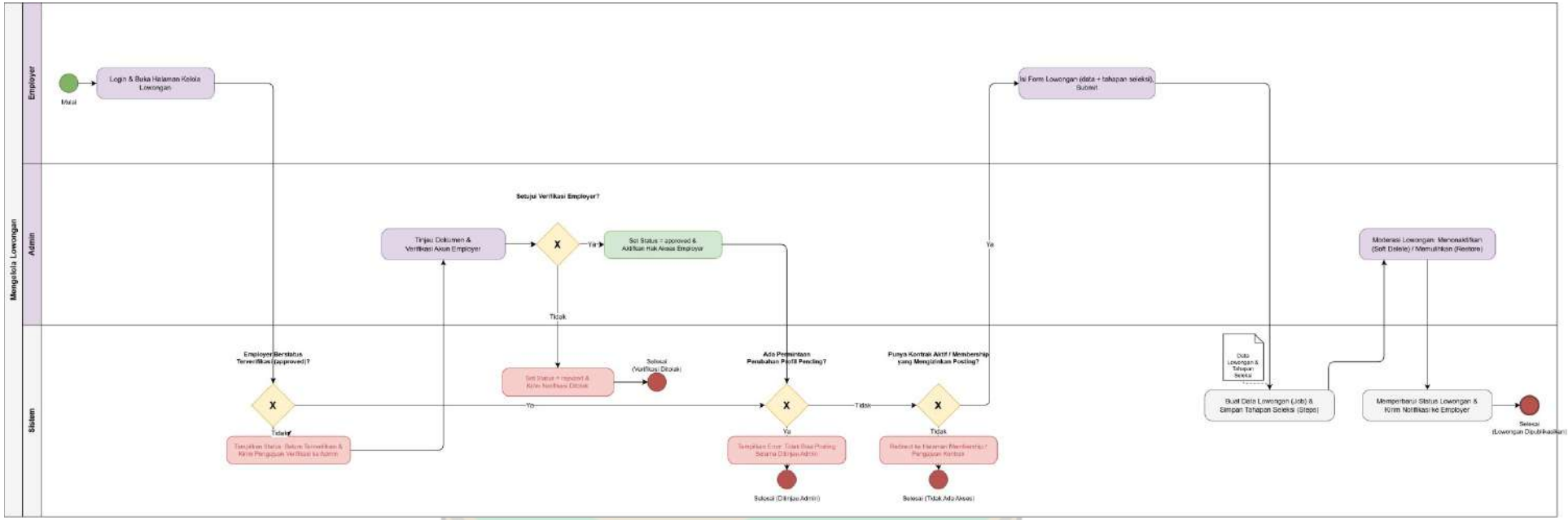
   a. Jika tidak memenuhi syarat, sistem menampilkan pesan sesuai kondisi dan proses berhenti.

   b. Jika memenuhi syarat, sistem menyimpan lamaran dengan status pending.

3. Sistem mengirimkan notifikasi lamaran baru kepada *Employer*.

4.1.1.3 BPMN Mengelola Lowongan Kerja yang Diusulkan

Proses mengelola lowongan kerja yang diusulkan dilakukan oleh *Employer* yang telah terverifikasi. BPMN yang diusulkan dapat dilihat pada Gambar 4.3. Penjelasan tahapannya adalah sebagai berikut:



Gambar 4.3 BPMN Mengelola Lowongan Kerja yang Diusulkan

1. *Employer* membuka halaman kelola lowongan.

2. Sistem memeriksa status verifikasi serta hak *posting Employer*, yaitu kontrak kerja sama atau *membership* yang aktif.

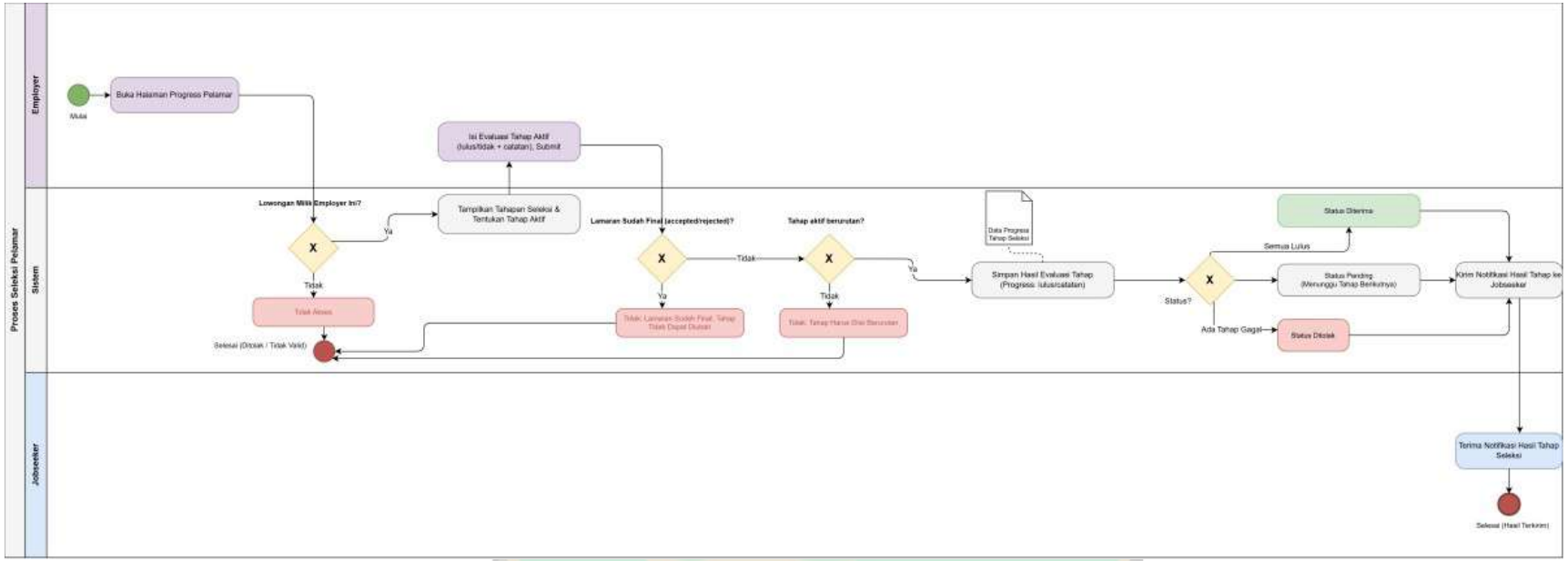
3. *Employer* mengisi data lowongan beserta susunan tahapan seleksinya.

4. Sistem menyimpan lowongan beserta urutan tahapan seleksi.

5. *Employer* dapat mengubah, menutup, atau membuka kembali lowongan. Perubahan tahapan seleksi terkunci apabila sudah ada pelamar dalam proses seleksi.

4.1.1.4 BPMN Proses Seleksi Pelamar yang Diusulkan

Proses seleksi pelamar yang diusulkan dilakukan oleh *Employer* terhadap lamaran yang masuk, dengan hasil yang tersampaikan otomatis kepada *Jobseeker*. BPMN yang diusulkan dapat dilihat pada Gambar 4.4. Penjelasan tahapannya adalah sebagai berikut:



Gambar 4.4 BPMN Proses Seleksi Pelamar yang Diusulkan

1. *Employer* membuka halaman *progress* pelamar, kemudian sistem menampilkan tahapan seleksi beserta tahap yang sedang aktif.

2. *Employer* mengisi hasil evaluasi berupa lulus atau tidak lulus beserta catatan pada tahap aktif.

3. Sistem memvalidasi urutan tahap dan memastikan lamaran belum berstatus final.

4. Sistem menyimpan hasil evaluasi dan menyinkronkan status lamaran: diterima apabila seluruh tahap lulus, ditolak apabila terdapat tahap yang gagal, atau tetap berjalan apabila masih ada tahap berikutnya.

5. Sistem mengirimkan notifikasi hasil tahapan kepada *Jobseeker*.

4.1.2 Analisis Kebutuhan Fungsional

Berdasarkan analisis proses bisnis dan wawancara yang telah dilakukan, sistem ini melibatkan empat aktor, yaitu Guest, *Jobseeker*, *Employer*, dan *Admin*. Guest merupakan pengunjung yang belum masuk ke sistem, *Jobseeker* merupakan mahasiswa atau alumni pencari kerja, *Employer* merupakan perusahaan yang membuka lowongan, sedangkan *Admin* merupakan pengelola platform dari pihak pusat karir. Kebutuhan fungsional setiap aktor dijabarkan sebagai berikut:

a. Kebutuhan Fungsional Guest

1. Guest dapat melihat halaman utama.

2. Guest dapat melihat daftar lowongan aktif.

3. Guest dapat membaca artikel yang dipublikasikan.

4. Guest dapat melakukan registrasi akun, login, dan reset kata sandi (lupa password).

b. Kebutuhan Fungsional *Jobseeker*

1. *Jobseeker* dapat melihat *dashboard* beserta statistik lamaran.

2. *Jobseeker* dapat mengelola profil, riwayat pendidikan, dan bahasa.

3. Jobseeker dapat melihat daftar dan detail lowongan aktif.

4. *Jobseeker* dapat melamar pekerjaan.

5. *Jobseeker* dapat melihat daftar lamaran beserta *progress* seleksinya.

6. *Jobseeker* dapat mengunduh CV dalam bentuk PDF.

7. *Jobseeker* dapat mendaftar ke *job fair* dan memperoleh QR code personal.

8. *Jobseeker* dapat melakukan check-in kehadiran pada *job fair*.

9. *Jobseeker* dapat memindai *booth employer* untuk masuk antrian.

10. *Jobseeker* dapat melihat notifikasi.

c. Kebutuhan Fungsional *Employer*

1. *Employer* dapat mengajukan data verifikasi perusahaan.

2. *Employer* dapat mengelola profil perusahaan, termasuk mengajukan permintaan perubahan data legal.

3. *Employer* dapat mengelola lowongan beserta tahapan seleksinya.

4. *Employer* dapat melihat pelamar serta melihat dan mengunduh CV pelamar.

5. *Employer* dapat mengevaluasi pelamar pada setiap tahapan seleksi.

6. *Employer* dapat mendaftarkan lowongan ke *job fair* serta mengelola *booth* dan antrian.

7. *Employer* dapat membeli paket *membership* dan melakukan pembayaran manual.

8. *Employer* dapat memposting artikel.

9. *Employer* dapat melihat notifikasi.

d. Kebutuhan Fungsional *Admin*

1. *Admin* dapat melihat *dashboard* backoffice.

2. *Admin* dapat mengelola data pengguna.

3. *Admin* dapat memverifikasi *employer* yang mendaftar.

4. *Admin* dapat meninjau permintaan perubahan data *employer*.

5. *Admin* dapat mengelola lowongan dan *job fair*, termasuk menyetujui pendaftaran lowongan ke *job fair*.

6. *Admin* dapat meninjau artikel sebelum dipublikasikan.

7. *Admin* dapat mengelola paket *membership*, kontrak mitra kerja, dan rekening bank.

8. *Admin* dapat memverifikasi pembayaran manual *membership*.

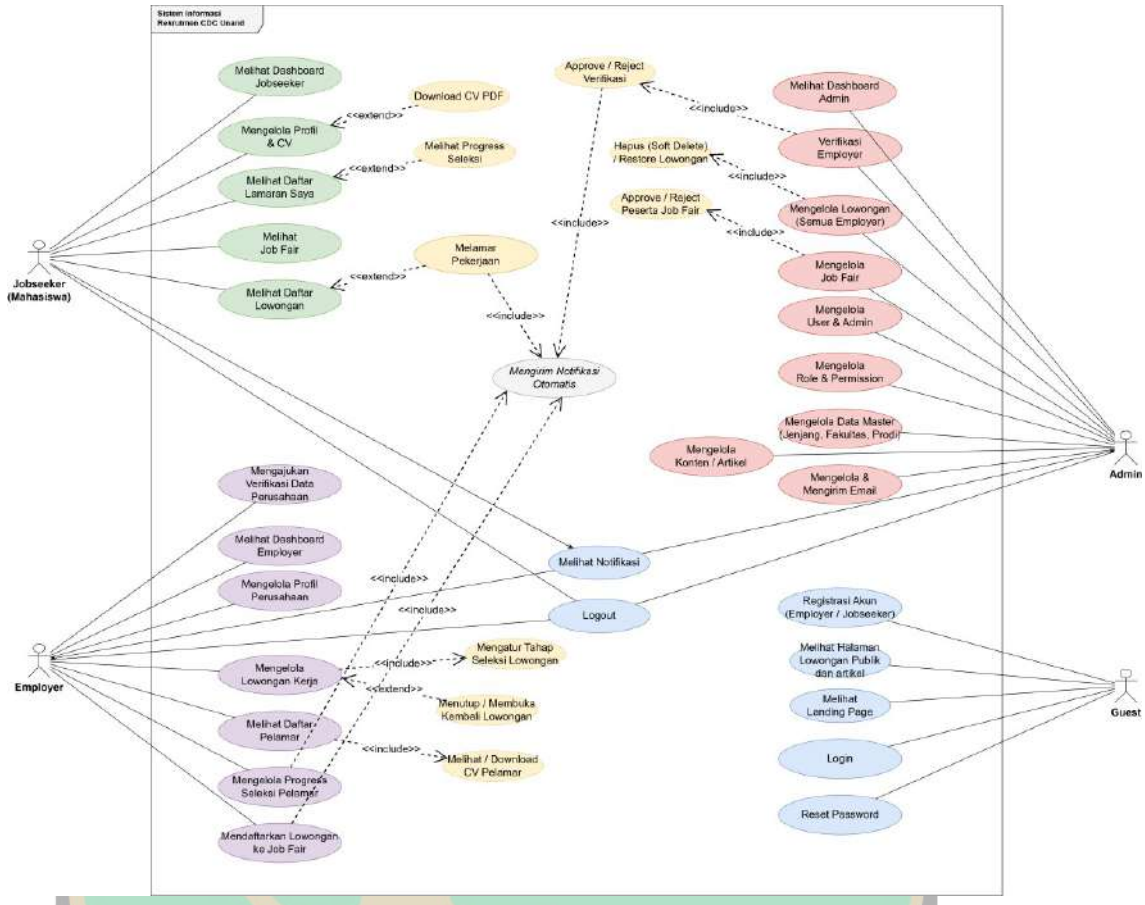
9. Admin dapat melihat analitik rekrutmen berupa statistik lamaran dan corong tahap seleksi.

4.1.3 Use Case Diagram

Use case diagram menggambarkan interaksi antara aktor dengan fungsional yang disediakan sistem. Berdasarkan analisis kebutuhan yang telah dilakukan, Sistem Informasi Rekrutmen ini melibatkan empat aktor, yaitu *Admin*, *Employer*, *Jobseeker*, dan Guest (pengunjung umum). *Use case diagram* sistem dapat dilihat pada Gambar 4.5.

Sistem terbagi menjadi dua bagian, yaitu backoffice dan frontoffice. Backoffice diakses oleh *Admin* dan difokuskan pada pengelolaan data serta operasional platform, seperti verifikasi *Employer*, pengelolaan *job fair*, verifikasi pembayaran *membership*, dan pengelolaan artikel. Sementara itu, frontoffice diakses oleh *Employer*, *Jobseeker*, dan Guest. *Employer* dapat mengelola lowongan, meninjau pelamar, serta mengikuti *job fair*; *Jobseeker* dapat menelusuri dan melamar lowongan serta mengelola profil dan CV; sedangkan

Guest dapat melihat halaman utama dan daftar lowongan yang tersedia tanpa perlu masuk ke sistem.



Gambar 4.5 Use Case Diagram Sistem Informasi Rekrutmen CDC Unand

4.1.3.1 Use Case Scenario

Use case scenario menjelaskan urutan langkah yang menggambarkan alur interaksi antara aktor dan sistem untuk mencapai suatu tujuan. Setiap skenario terdiri atas kondisi awal (entry condition), skenario normal yang memuat interaksi utama antara aktor dan sistem, serta skenario alternatif yang menangani kondisi ketika alur normal tidak terpenuhi. *Use case scenario* yang dijelaskan pada subbab ini yaitu *login*, pembayaran *membership*, mengelola lowongan kerja, melamar pekerjaan, mengelola *progress* seleksi pelamar, dan mendaftarkan lowongan ke *job fair*.

Skenario *login* merupakan alur yang dilakukan oleh Guest (calon *Employer*, *Jobseeker*, atau *Admin*) untuk masuk ke sistem sesuai perannya. *Use case scenario login* dapat dilihat pada Tabel 4.1.

Tabel 4.1 Use Case Scenario Login

Use Case Name	*Login*	
Actor	*Guest* (calon *Employer* / *Jobseeker* / *Admin*)	
Entry Condition	Aktor belum *login* dan sudah memiliki akun dengan peran tertentu.	
Scenario (Normal)	**Aktor**	**Sistem**
	1. Aktor membuka halaman *login* dan mengisi *email* serta *password*.	
		2. Sistem memvalidasi kredensial yang dimasukkan.
		3. Sistem mengarahkan Aktor ke *dashboard* sesuai perannya (*Admin*, *Employer*, atau *Jobseeker*).
Alternative	1. Jika *email* atau *password* salah, Sistem menampilkan pesan kesalahan	

Scenario	dan mengembalikan Aktor ke halaman *login*.
	2. Jika *Employer* sudah *login* tetapi belum terverifikasi, Sistem mengarahkan ke halaman status verifikasi, bukan ke *dashboard*.

Skenario pembayaran *membership* merupakan alur yang dilakukan oleh *Employer* untuk mengaktifkan paket *membership* melalui pembayaran manual yang diverifikasi oleh *Admin*. *Use case scenario* pembayaran *membership* dapat dilihat pada Tabel 4.2.

Tabel 4.2 Use Case Scenario Pembayaran Membership

Use Case Name	Pembayaran *Membership*	
Actor	*Employer*, *Admin*	
Entry Condition	Aktor (*Employer*) sudah *login* dan memilih paket *membership* yang tersedia.	
Scenario (Normal)	**Aktor**	**Sistem**
	1. Aktor (*Employer*) memilih paket *membership* dan metode pembayaran manual.	
	2. Aktor mengunggah bukti transfer sesuai rekening tujuan.	
		3. Sistem menyimpan pengajuan pembayaran dengan status menunggu verifikasi.
	4. Aktor (*Admin*) memeriksa dan menyetujui bukti transfer.	

		5. Sistem mengaktifkan *membership Employer* dan mengirim notifikasi persetujuan.
Alternative Scenario	1. Jika *Employer* masih memiliki kontrak aktif atau pembayaran yang belum diverifikasi, Sistem menolak pengajuan baru.	
	2. Jika bukti transfer tidak valid, Sistem menampilkan pesan validasi.	
	3. Jika *Admin* menolak pembayaran, Sistem mengirim notifikasi penolakan beserta alasannya kepada *Employer*.	

Skenario mengelola lowongan kerja merupakan alur yang dilakukan oleh *Employer* untuk membuat, mengubah, serta menutup dan membuka kembali lowongan beserta tahapan seleksinya. *Use case scenario* mengelola lowongan kerja dapat dilihat pada Tabel 4.3.

Tabel 4.3 Use Case Scenario Mengelola Lowongan Kerja

Use Case Name	Mengelola Lowongan Kerja	
Actor	*Employer*	
Entry Condition	Aktor sudah *login*, terverifikasi, dan berada di halaman kelola lowongan.	
Scenario (Normal)	**Aktor**	**Sistem**
	1. Aktor membuat lowongan baru beserta tahapan seleksinya.	
		2. Sistem menyimpan dan menampilkan lowongan pada daftar lowongan *Employer*.
	3. Aktor mengedit, menutup, atau membuka kembali lowongan yang	

	sudah ada.	
		4. Sistem memperbarui data lowongan sesuai perubahan yang dilakukan.
Alternative Scenario	1. Jika *Employer* belum terverifikasi atau tidak memiliki hak *posting*, Sistem menolak dan mengarahkan ke halaman terkait.	
	2. Jika lowongan yang diedit sudah memiliki pelamar dalam proses seleksi, Sistem mengunci perubahan tahapan seleksi.	

Skenario melamar pekerjaan merupakan alur yang dilakukan oleh *Jobseeker* untuk mengirim lamaran pada sebuah lowongan setelah kelengkapan profil terpenuhi. *Use case scenario* melamar pekerjaan dapat dilihat pada Tabel 4.4.

Tabel 4.4 Use Case Scenario Melamar Pekerjaan

Use Case Name	Melamar Pekerjaan	
Actor	*Jobseeker*	
Entry Condition	Aktor sudah *login* sebagai *Jobseeker* dan berada di halaman daftar lowongan.	
Scenario (Normal)	**Aktor**	**Sistem**
	1. Aktor memilih lowongan dari daftar lowongan aktif dan menekan tombol Lamar.	
		2. Sistem memeriksa kelengkapan profil dan riwayat lamaran Aktor.

		3. Sistem menyimpan lamaran dan mengirim notifikasi ke *Employer*.
Alternative Scenario	1. Jika lowongan sudah ditutup, Sistem menampilkan pesan bahwa lowongan tidak lagi menerima lamaran.	
	2. Jika profil belum lengkap, Sistem meminta Aktor melengkapi profil terlebih dahulu.	
	3. Jika Aktor sudah pernah melamar lowongan yang sama, Sistem menolak pengiriman ulang.	

Skenario mengelola *progress* seleksi pelamar merupakan alur yang dilakukan oleh *Employer* untuk mengevaluasi pelamar pada tiap tahap seleksi hingga status lamaran ditentukan. *Use case scenario* mengelola *progress* seleksi pelamar dapat dilihat pada Tabel 4.5.

Tabel 4.5 Use Case Scenario Mengelola Progress Seleksi Pelamar

Use Case Name	Mengelola *Progress* Seleksi Pelamar	
Actor	*Employer*	
Entry Condition	Aktor sudah *login* dan membuka halaman *progress* salah satu pelamar pada lowongan miliknya.	
Scenario (Normal)	**Aktor**	**Sistem**
	1. Aktor membuka *progress* salah satu pelamar dan mengisi hasil evaluasi tahap aktif.	
		2. Sistem menyimpan hasil evaluasi tersebut.

		3. Sistem memperbarui status lamaran (diterima, ditolak, atau masih berjalan) berdasarkan hasil seluruh tahapan.
		4. Sistem mengirim notifikasi hasil tahap kepada *Jobseeker*.
Alternative Scenario	1. Jika lamaran sudah berstatus final, Sistem menolak perubahan lebih lanjut.	
	2. Jika Aktor mengisi tahap yang bukan urutannya, Sistem menolak dan meminta tahap diisi berurutan.	

Skenario mendaftarkan lowongan ke *job fair* merupakan alur yang dilakukan oleh *Employer* untuk mendaftarkan lowongan miliknya ke sebuah *job fair*, yang kemudian menunggu persetujuan *Admin*. *Use case scenario* mendaftarkan lowongan ke *job fair* dapat dilihat pada Tabel 4.6.

Tabel 4.6 Use Case Scenario Mendaftarkan Lowongan ke Job Fair

Use Case Name	Mendaftarkan Lowongan ke *Job Fair*	
Actor	*Employer*	
Entry Condition	Aktor sudah *login* dan terverifikasi, serta *job fair* yang dituju berstatus aktif.	
Scenario (Normal)	**Aktor**	**Sistem**
	1. Aktor memilih *job fair* aktif dan salah satu lowongan miliknya untuk didaftarkan.	
		2. Sistem memvalidasi kelayakan pendaftaran (jadwal, kuota, dan

		status *job fair*).
		3. Sistem menyimpan pendaftaran dengan status menunggu persetujuan *admin* dan mengirim notifikasi ke *Admin*.
Alternative Scenario	1. Jika *job fair* belum aktif, sudah dimulai, atau kuota penuh, Sistem menampilkan pesan sesuai kondisi dan menolak pendaftaran.	
	2. Jika lowongan sudah terdaftar pada *job fair* yang sama, Sistem menolak pendaftaran ulang.	

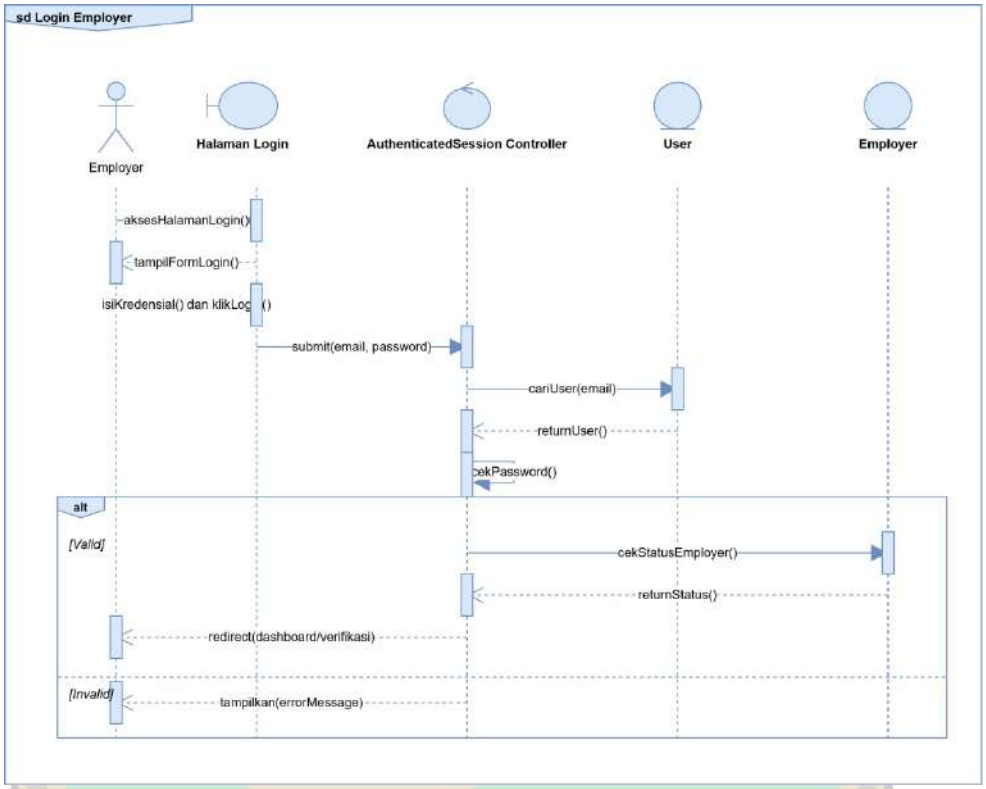
4.1.4 Sequence Diagram

Sequence diagram digunakan untuk menggambarkan interaksi antar objek dan logika prosedural dalam sistem berdasarkan urutan waktu. Diagram ini memperlihatkan pesan yang dipertukarkan antara aktor, antarmuka, *controller*, dan *model* selama sebuah fungsional dijalankan. Sistem ini dibangun menggunakan *framework* Laravel yang menerapkan arsitektur *Model*-View-*Controller* (MVC), sehingga setiap permintaan aktor diteruskan melalui *controller* yang kemudian berinteraksi dengan *model* untuk mengakses basis data. *Sequence diagram* yang dijelaskan pada subbab ini yaitu *login employer*, pembayaran *membership*, mengelola lowongan kerja, melamar pekerjaan, evaluasi tahapan seleksi, dan *job fair*.

4.1.4.1 Sequence Diagram Login Employer

Proses *login employer* dijalankan melalui class AuthenticatedSessionController. *Employer* terlebih dahulu membuka halaman *login*, kemudian sistem menampilkan form *login*. *Employer* memasukkan *email* dan *password* lalu menekan tombol masuk, dan data tersebut diteruskan ke *controller*. *Controller* memvalidasi kelengkapan masukan, kemudian mencocokkan kredensial terhadap *model* User. Jika kredensial cocok, *controller*

membaca peran pengguna dari atribut user_type dan mengarahkan *Employer* ke *dashboard employer*. Sebaliknya, jika kredensial tidak cocok, *controller* mengembalikan *Employer* ke halaman *login* beserta pesan kesalahan. *Sequence diagram login employer* dapat dilihat pada Gambar 4.6.

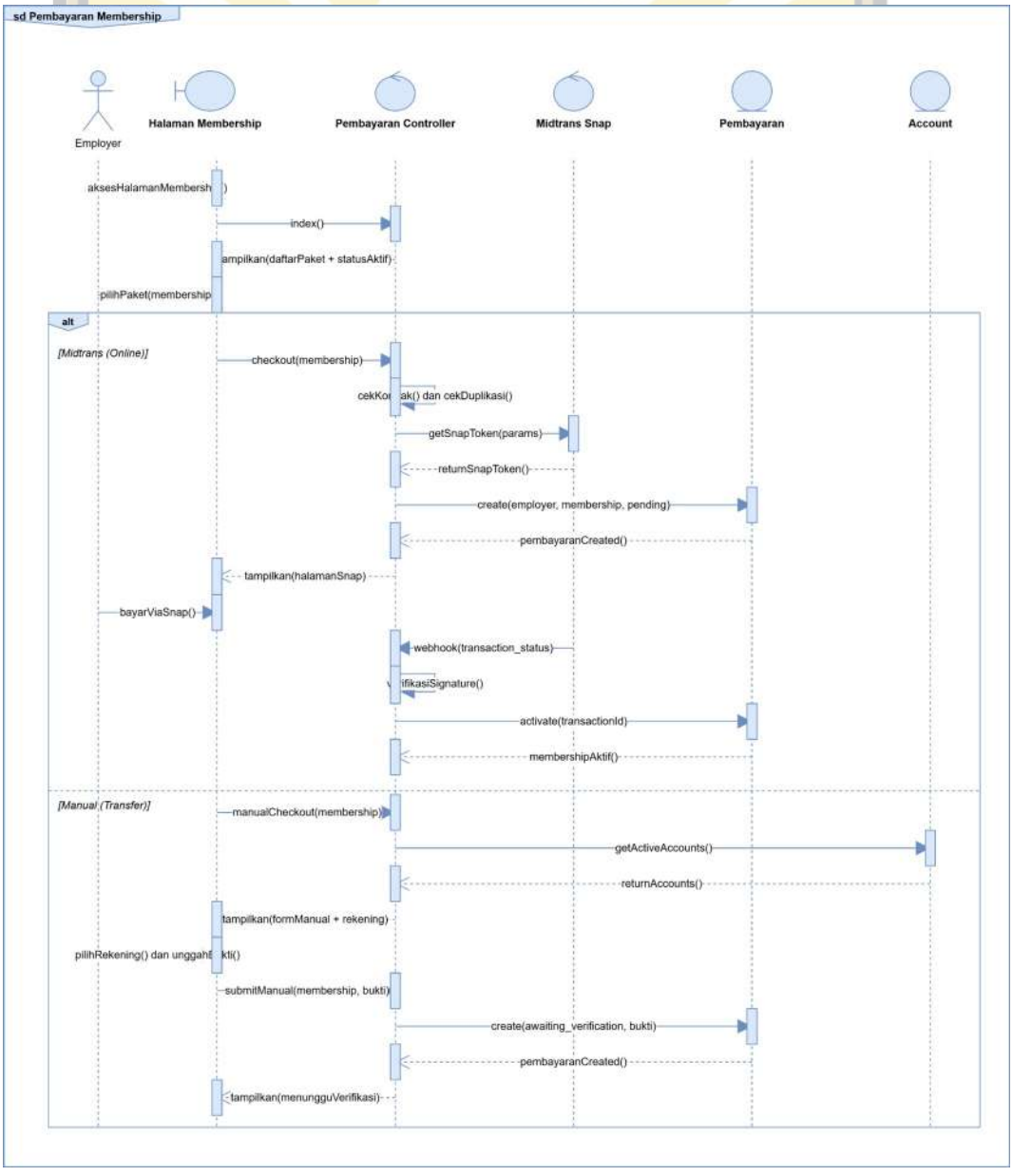


Gambar 4.6 Sequence Diagram Login Employer

4.1.4.2 Sequence Diagram Pembayaran Membership

Proses pembayaran *membership* dijalankan melalui class PembayaranController. *Employer* membuka halaman *membership*, kemudian *controller* mengambil daftar paket dari *model Membership* serta daftar rekening tujuan dari *model* Account untuk ditampilkan. *Employer* memilih paket, mengunggah bukti transfer, lalu menekan tombol kirim. Permintaan diteruskan ke *controller* yang memvalidasi masukan dan menyimpan data pembayaran ke *model* Pembayaran dengan status menunggu verifikasi, kemudian menampilkan pesan bahwa bukti transfer berhasil diunggah.

Selanjutnya, *Admin* membuka halaman verifikasi pembayaran dan meninjau bukti transfer yang diambil *controller* dari *model* Pembayaran. Apabila *Admin* menyetujui, *controller* mengubah status pembayaran menjadi lunas dan mengaktifkan *membership* pada data *Employer* sesuai durasi paket, lalu mengirimkan notifikasi melalui *model* Notification. Apabila ditolak, *controller* mengubah status menjadi gagal dan mengirimkan notifikasi penolakan beserta catatan kepada *Employer*. *Sequence diagram* pembayaran *membership* dapat dilihat pada Gambar 4.7.

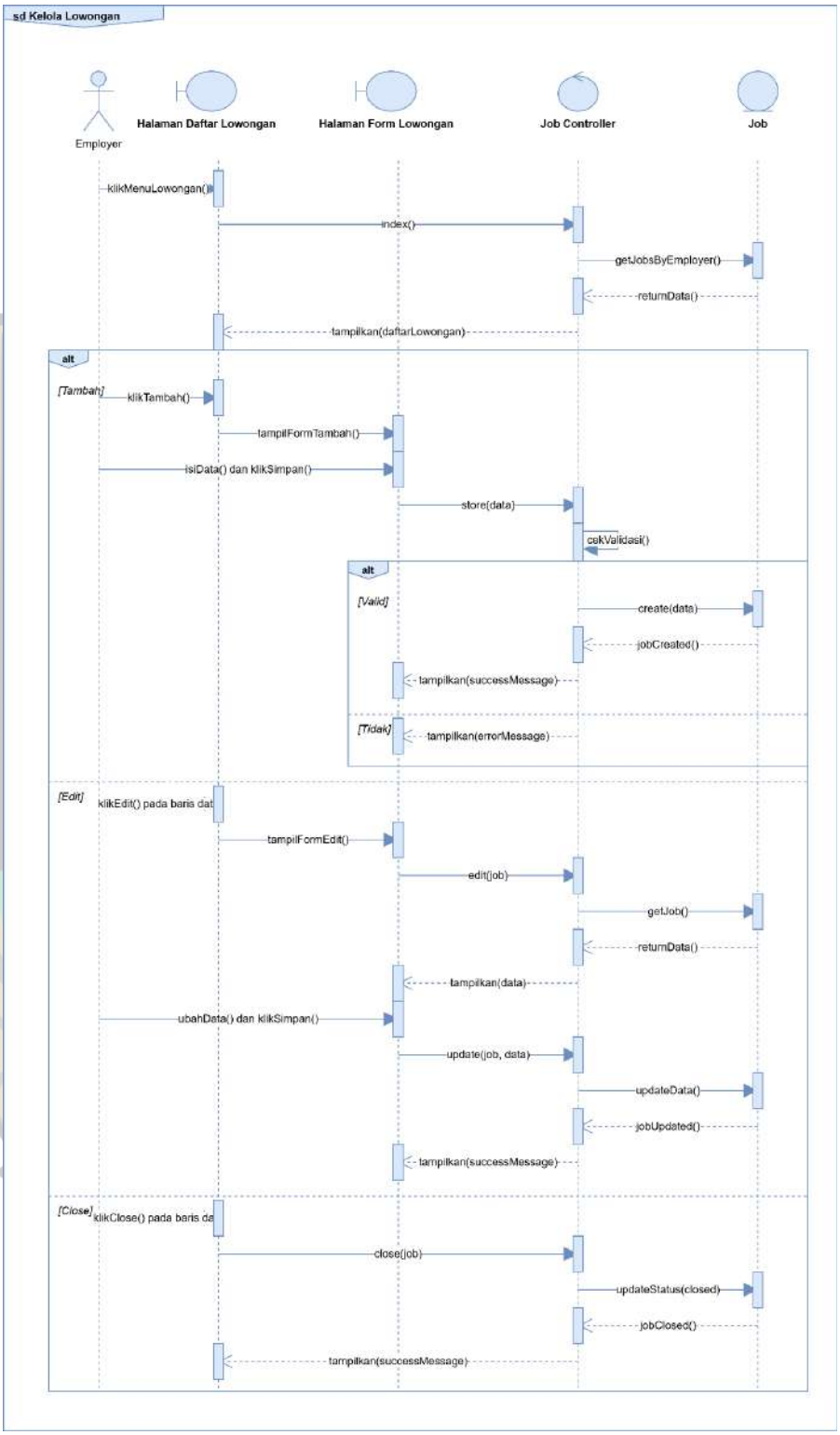


Gambar 4.7 Sequence Diagram Pembayaran Membership

4.1.4.3 Sequence Diagram Mengelola Lowongan Kerja

Proses mengelola lowongan kerja dijalankan melalui class JobController. *Employer* membuka halaman kelola lowongan, kemudian *controller* mengambil data lowongan milik *Employer* dari *model* Job untuk ditampilkan. Untuk membuat lowongan baru, *Employer* mengisi form data lowongan beserta tahapan seleksi lalu menekan tombol simpan. *Controller* memvalidasi masukan, menyimpan data lowongan ke *model* Job, kemudian menyimpan setiap tahapan seleksi ke *model* Step yang merujuk pada *model* Proses.

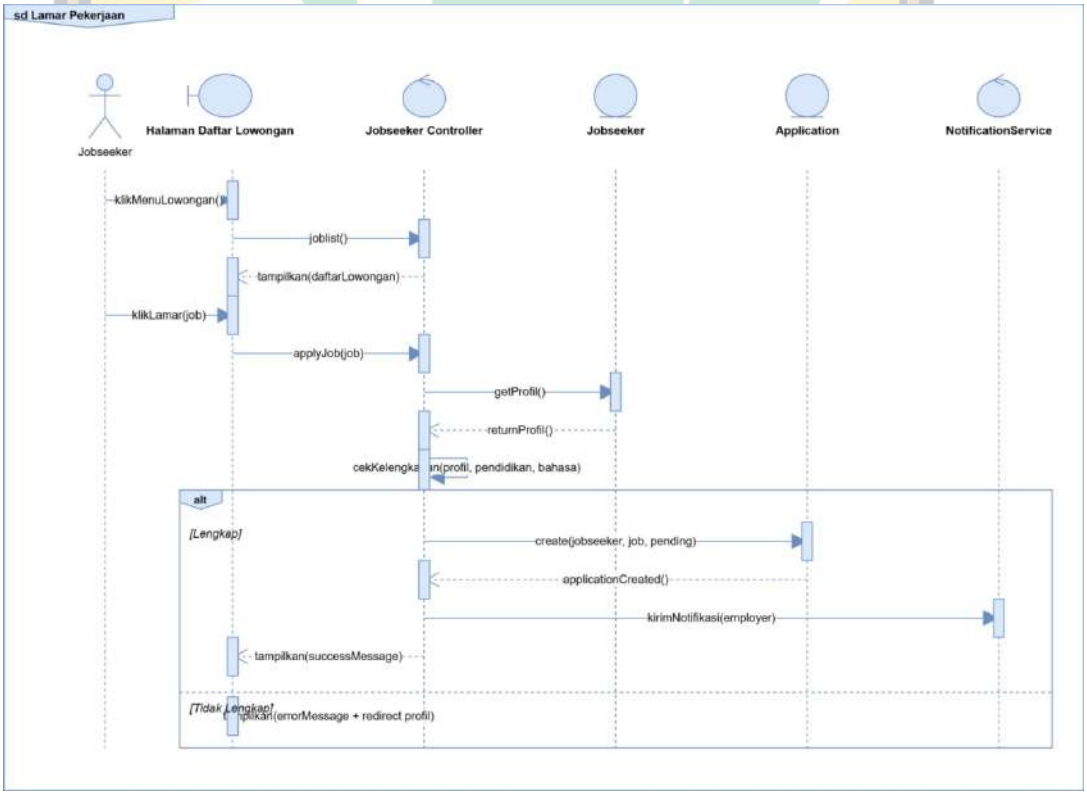
Untuk mengubah lowongan, *Employer* membuka form edit yang datanya diambil *controller* dari *model* Job, memperbarui data, lalu menyimpannya sehingga *controller* memperbarui data pada *model* Job. Selain itu, *Employer* dapat menutup atau membuka kembali lowongan, di mana permintaan tersebut diteruskan ke *controller* untuk memperbarui atribut status pada *model* Job. *Sequence diagram* mengelola lowongan kerja dapat dilihat pada Gambar 4.8.



Gambar 4.8 Sequence Diagram Mengelola Lowongan Kerja

4.1.4.4 Sequence Diagram Melamar Pekerjaan

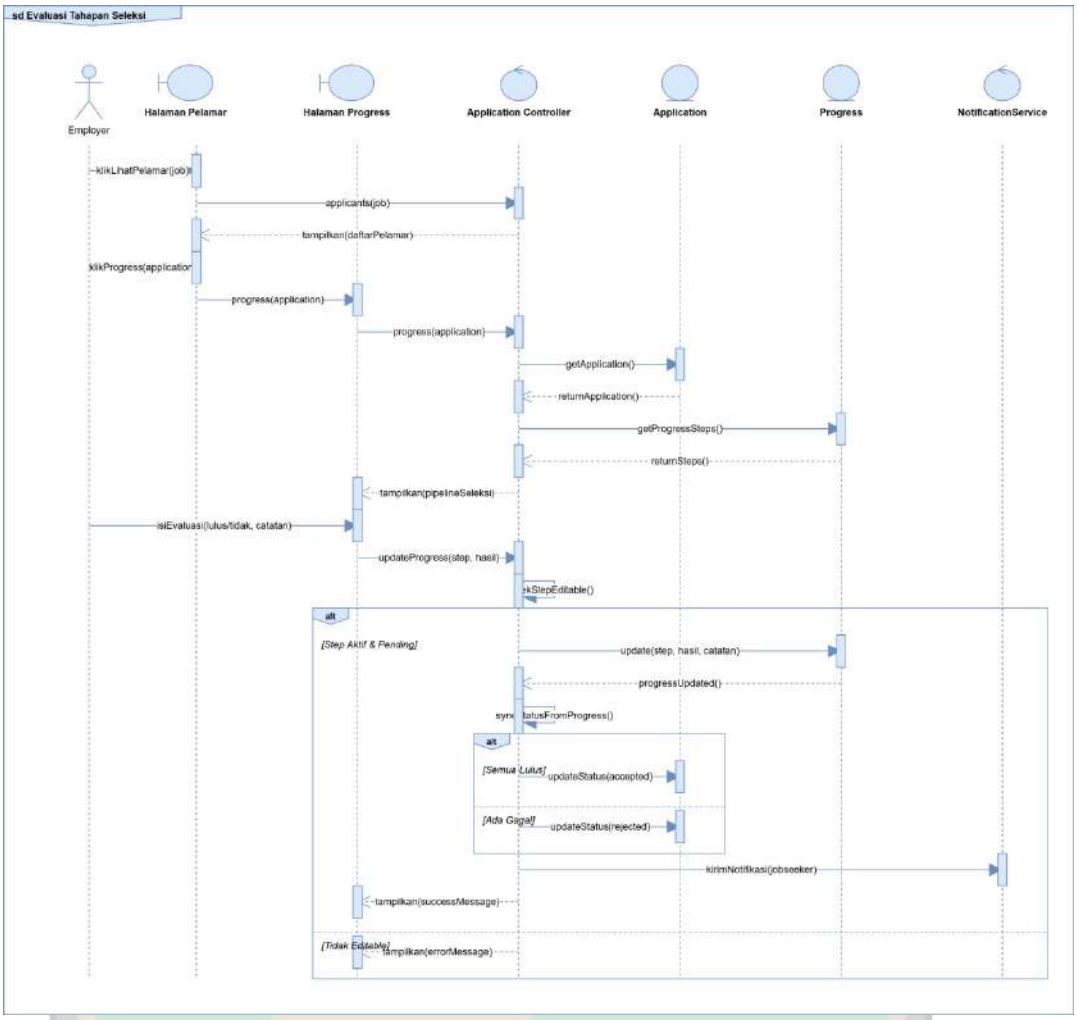
Proses melamar pekerjaan dijalankan melalui class JobseekerController. *Jobseeker* membuka daftar lowongan aktif yang diambil *controller* dari *model* Job, memilih salah satu lowongan, lalu menekan tombol lamar. Permintaan diteruskan ke *controller* yang terlebih dahulu memeriksa kelengkapan profil *Jobseeker* melalui *model* RiwayatPendidikan dan Bahasa, serta memeriksa apakah lamaran yang sama sudah pernah dibuat pada *model* Application. Jika profil belum lengkap atau lamaran sudah pernah dibuat, *controller* menampilkan pesan yang sesuai. Sebaliknya, jika seluruh syarat terpenuhi, *controller* menyimpan lamaran baru ke *model* Application dengan status pending, kemudian mengirimkan notifikasi lamaran baru kepada *Employer* melalui *model* Notification. *Sequence diagram* melamar pekerjaan dapat dilihat pada Gambar 4.9.



Gambar 4.9 Sequence Diagram Melamar Pekerjaan

4.1.4.5 Sequence Diagram Evaluasi Tahapan Seleksi

Proses evaluasi tahapan seleksi dijalankan melalui class ApplicationController. *Employer* membuka halaman *progress* salah satu pelamar, kemudian *controller* mengambil tahapan seleksi dari *model* Step beserta hasil evaluasi sebelumnya dari *model Progress* untuk ditampilkan. *Employer* mengisi hasil evaluasi berupa lulus atau tidak lulus beserta catatan pada tahap yang sedang aktif, lalu menyimpannya. *Controller* memvalidasi bahwa tahap yang diisi merupakan tahap aktif dan lamaran belum berstatus final, kemudian menyimpan hasil ke *model Progress*. Setelah itu, *controller* menyinkronkan status lamaran pada *model* Application berdasarkan hasil seluruh tahapan, yaitu diterima apabila semua tahap lulus, ditolak apabila terdapat tahap yang gagal, atau tetap berjalan apabila masih ada tahap berikutnya. Terakhir, *controller* mengirimkan notifikasi hasil kepada *Jobseeker* melalui *model* Notification. *Sequence diagram* evaluasi tahapan seleksi dapat dilihat pada Gambar 4.10.

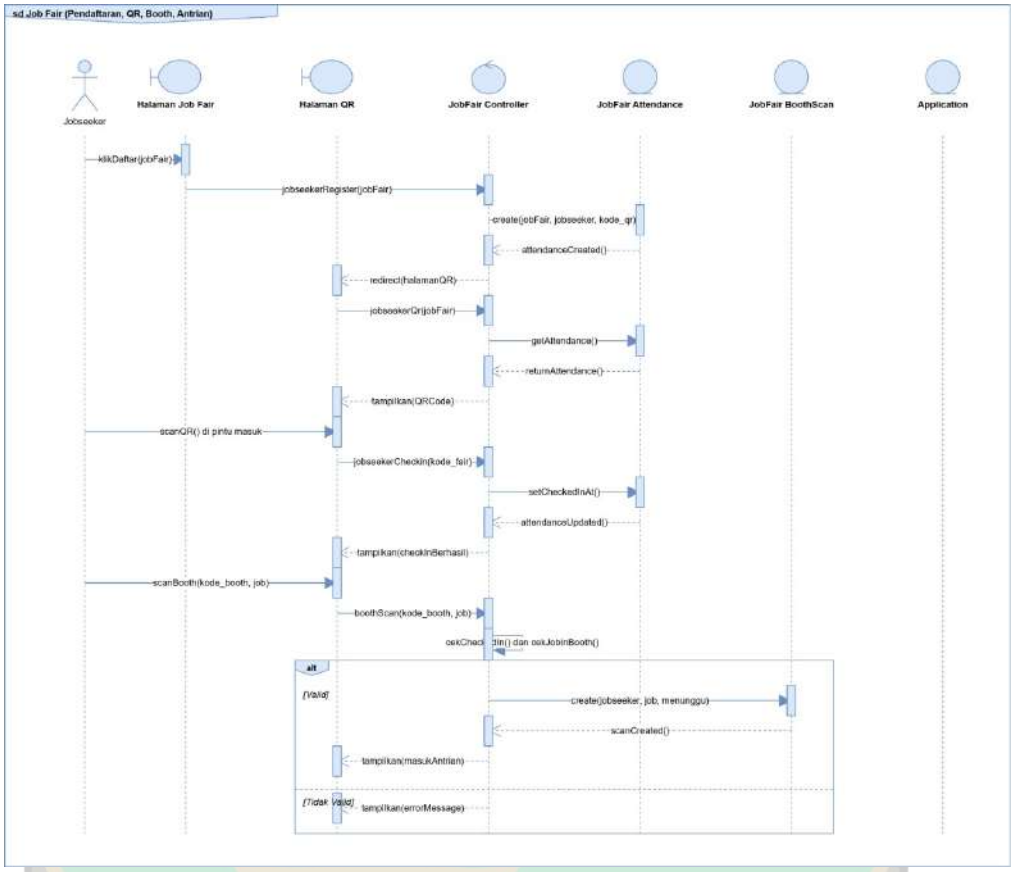


Gambar 4.10 Sequence Diagram Evaluasi Tahapan Seleksi

4.1.4.6 Sequence Diagram Job Fair

Proses mendaftarkan lowongan ke *job fair* dijalankan melalui class JobFairController. *Employer* membuka daftar *job fair* aktif yang diambil *controller* dari *model* JobFair, memilih sebuah *job fair*, lalu memilih salah satu lowongan miliknya untuk didaftarkan. Permintaan diteruskan ke *controller* yang memvalidasi kelayakan pendaftaran, meliputi status *job fair* yang masih aktif, jadwal pelaksanaan yang belum dimulai, dan ketersediaan kuota. Jika salah satu syarat tidak terpenuhi, *controller* menampilkan pesan yang sesuai. Sebaliknya, jika seluruh syarat terpenuhi, *controller* menyimpan pendaftaran ke *model* JobFairJob dengan status menunggu persetujuan, kemudian mengirimkan

notifikasi kepada *Admin* melalui *model* Notification. *Sequence diagram job fair* dapat dilihat pada Gambar 4.11.



Gambar 4.11 Sequence Diagram Job Fair

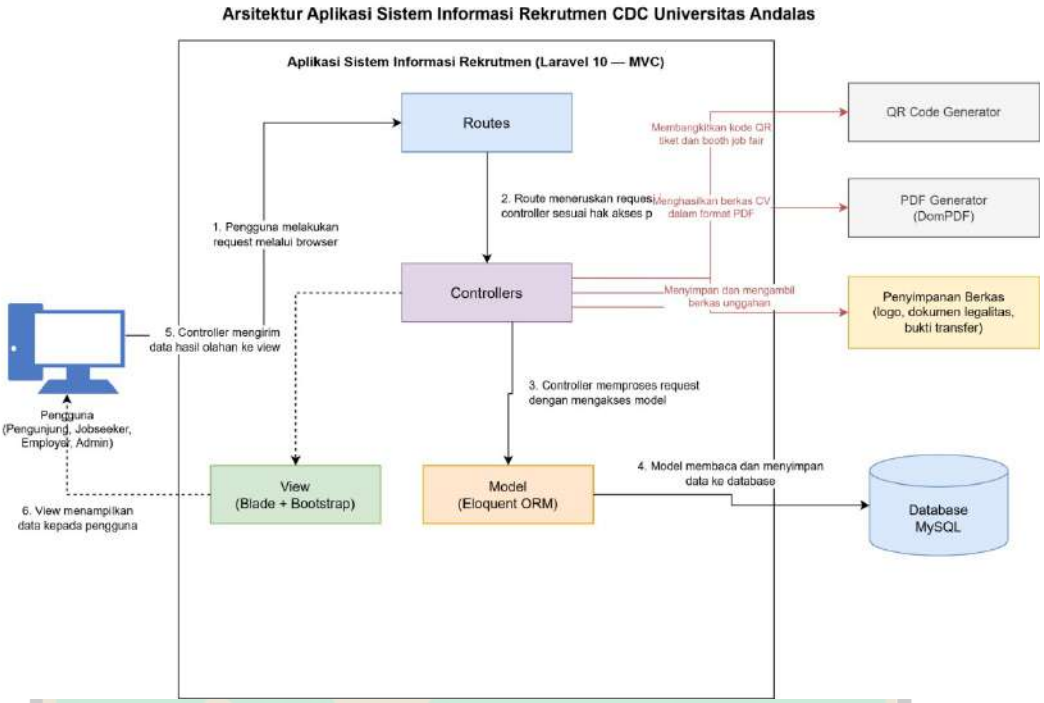
4.2 Perancangan Sistem

Subbab ini membahas rancangan sistem yang mencakup arsitektur aplikasi, class diagram, ERD, dan rancangan antarmuka.

4.2.1 Rancangan Arsitektur Aplikasi

Sistem dibangun berbasis web menggunakan *framework* Laravel yang menerapkan arsitektur *Model-View-Controller* (MVC). Permintaan dari browser pengguna diterima melalui routing yang meneruskannya ke *controller*. *Controller* menjalankan logika aplikasi dan berinteraksi dengan *model* untuk mengakses basis data MySQL, kemudian meneruskan data yang telah diolah ke *view* berbasis Blade untuk ditampilkan kembali kepada pengguna. Selain itu, sistem

memanfaatkan pustaka pendukung, di antaranya pembangkit *QR code* untuk kebutuhan *job fair* serta pembangkit dokumen PDF untuk pencetakan CV. Arsitektur aplikasi dapat dilihat pada Gambar 4.12.



Gambar 4.12 Arsitektur Aplikasi Sistem Informasi Rekrutmen

Sistem Informasi Rekrutmen pada *CDC* Universitas Andalas dibangun sebagai aplikasi *web* menggunakan *framework* Laravel dengan arsitektur *Model-View-Controller* (MVC). Saat pengguna melakukan *request* melalui browser, route meneruskan permintaan tersebut ke *controller* sesuai hak akses pengguna. Controller kemudian mengolah permintaan dengan mengakses *model*, dan *model* mengambil atau menyimpan data pada basis data MySQL lalu mengembalikannya kepada *controller*. Data yang telah diolah dikirimkan ke *view* berbasis Blade untuk ditampilkan kepada pengguna. Selain itu, *controller* memanfaatkan pustaka pembangkit *QR code* untuk tiket dan *booth job fair*, *library* PDF untuk mencetak CV, serta penyimpanan berkas untuk dokumen yang diunggah pengguna.

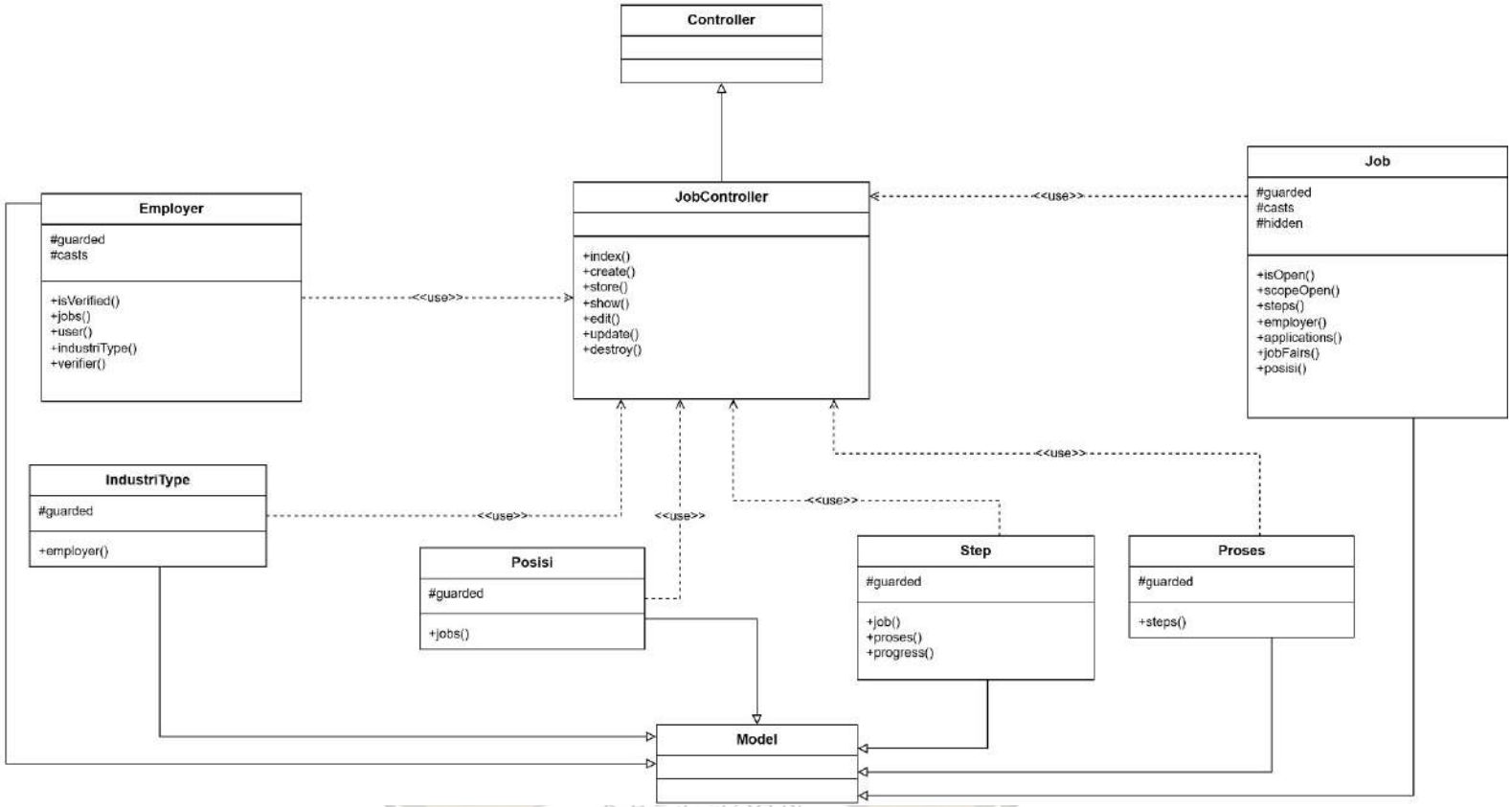
4.2.2 Rancangan Class Diagram

Class diagram menggambarkan kelas-kelas penyusun sistem yang meliputi atribut, metode, dan relasi antar kelas. Sistem ini menggunakan *framework* Laravel dengan arsitektur *Model*-View-*Controller* (MVC), sehingga kelas terbagi menjadi kelas *controller* yang menangani logika permintaan dan kelas *model* yang mewakili tabel pada basis data. Setiap kelas *controller* merupakan turunan (generalisasi) dari kelas *Controller* bawaan Laravel, sedangkan setiap kelas *model* merupakan turunan dari kelas *Model*. *Class diagram* yang dijelaskan pada subbab ini yaitu mengelola lowongan kerja, mengelola lamaran, mengelola *job fair*, dan verifikasi *employer*.

4.2.2.1 Class Diagram Mengelola Lowongan Kerja

Proses mengelola lowongan kerja dijalankan melalui *class* JobController, sebuah kelas yang bertanggung jawab untuk mengimplementasikan logika pemrograman yang terlibat dalam interaksi dengan *database*. *Class* JobController merupakan extend dari *Class* Controller. Pada class ini, menggunakan enam *class* yang diimplementasikan sebagai *extend* dari kelas model diantaranya Employer, Job, IndustriType, Posisi, Step, dan Proses. *Class* Job menyimpan data lowongan dan terhubung dengan *class* Employer sebagai pemilik lowongan. *Class* IndustriType dan Posisi digunakan sebagai data acuan bidang dan posisi pekerjaan, sedangkan *class* Step dan Proses digunakan untuk menyimpan tahapan seleksi pada lowongan. *Class* diagram mengelola lowongan kerja dapat dilihat pada Gambar 4.13.

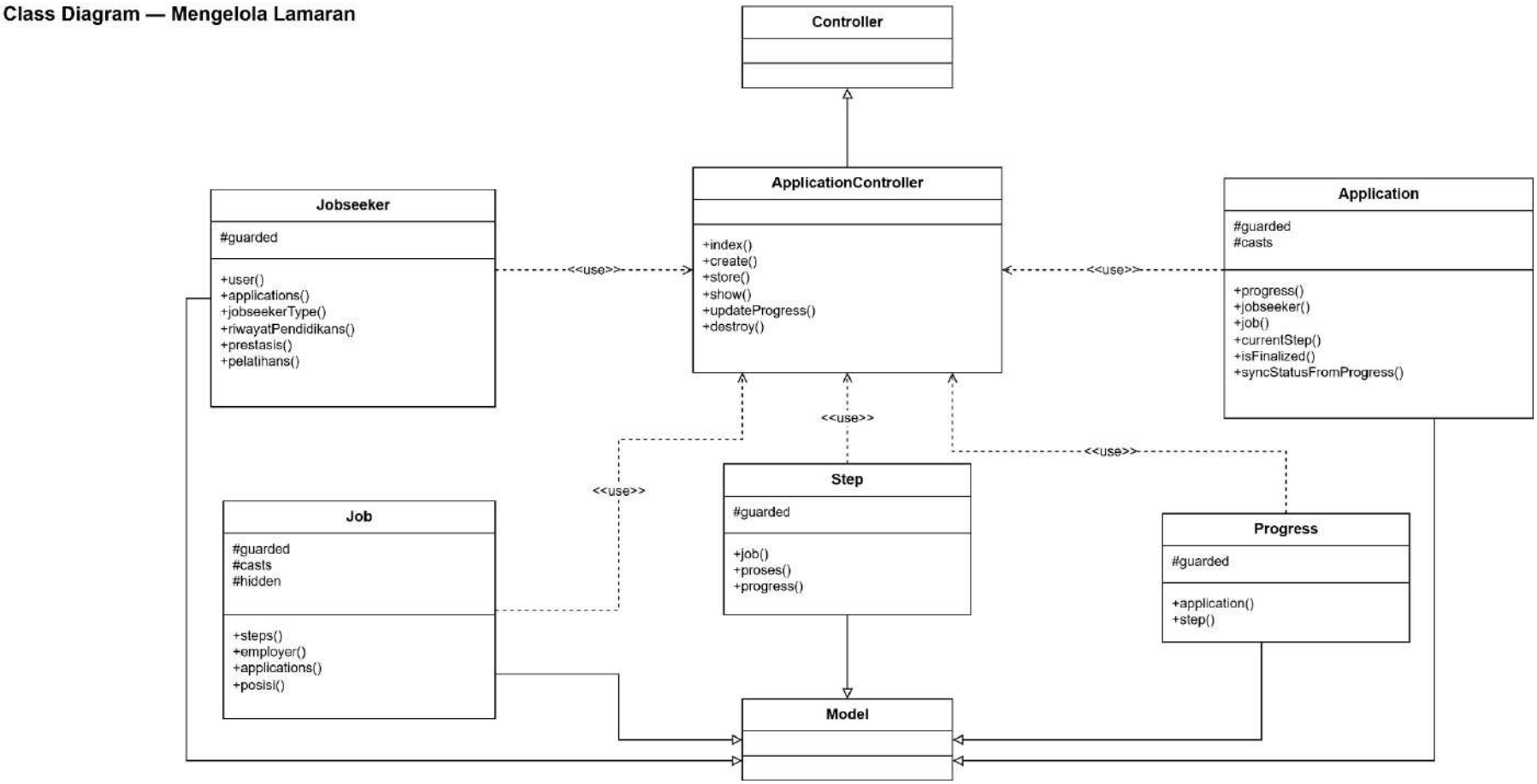
Class Diagram — Mengelola Lowongan Kerja



Gambar 4.13 Class Diagram Mengelola Lowongan Kerja

4.2.2.2 Class Diagram Mengelola Lamaran

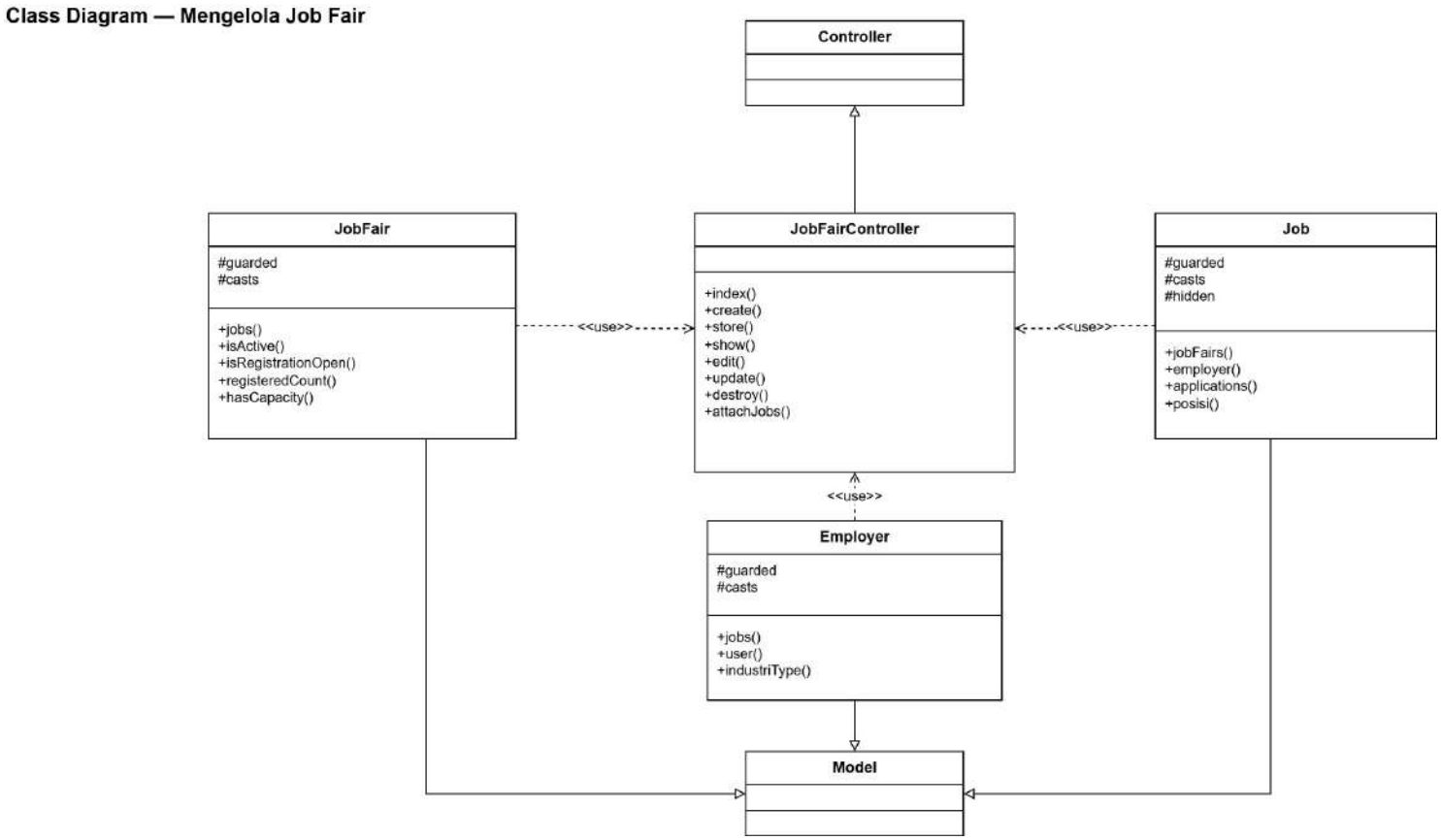
Proses mengelola lamaran dijalankan melalui *class* ApplicationController, sebuah kelas yang bertanggung jawab untuk mengimplementasikan logika pemrograman yang terlibat dalam interaksi dengan *database*. *Class* ApplicationController merupakan *extend* dari *Class* Controller. Pada *class* ini, menggunakan lima *class* yang diimplementasikan sebagai *extend* dari kelas *model* diantaranya Jobseeker, Application, Job, Step, dan Progress. *Class* Application menghubungkan pelamar yang disimpan pada *class* Jobseeker dengan lowongan yang disimpan pada *class* Job. *Class* Step dan Progress digunakan untuk menyimpan tahapan seleksi beserta hasil evaluasinya, sehingga status lamaran dapat diperbarui sesuai hasil evaluasi tersebut. *Class* diagram mengelola lamaran dapat dilihat pada Gambar 4.14.



Gambar 4.14 Class Diagram Mengelola Lamaran

4.2.2.3 Class Diagram Mengelola Job Fair

Proses mengelola *job fair* dijalankan melalui *class* JobFairController, sebuah kelas yang bertanggung jawab untuk mengimplementasikan logika pemrograman yang terlibat dalam interaksi dengan *database*. *Class* JobFairController merupakan *extend* dari *Class* Controller. Pada class ini, menggunakan tiga *class* yang diimplementasikan sebagai *extend* dari kelas *model* diantaranya JobFair, Job, dan Employer. *Class* JobFair menyimpan data acara *job fair* beserta status dan kuota pendaftarannya. *Class* Job digunakan untuk menyimpan lowongan yang didaftarkan, sedangkan *class* Employer digunakan untuk menyimpan data perusahaan pemilik lowongan yang mendaftar. *Class* diagram mengelola job fair dapat dilihat pada Gambar 4.15.

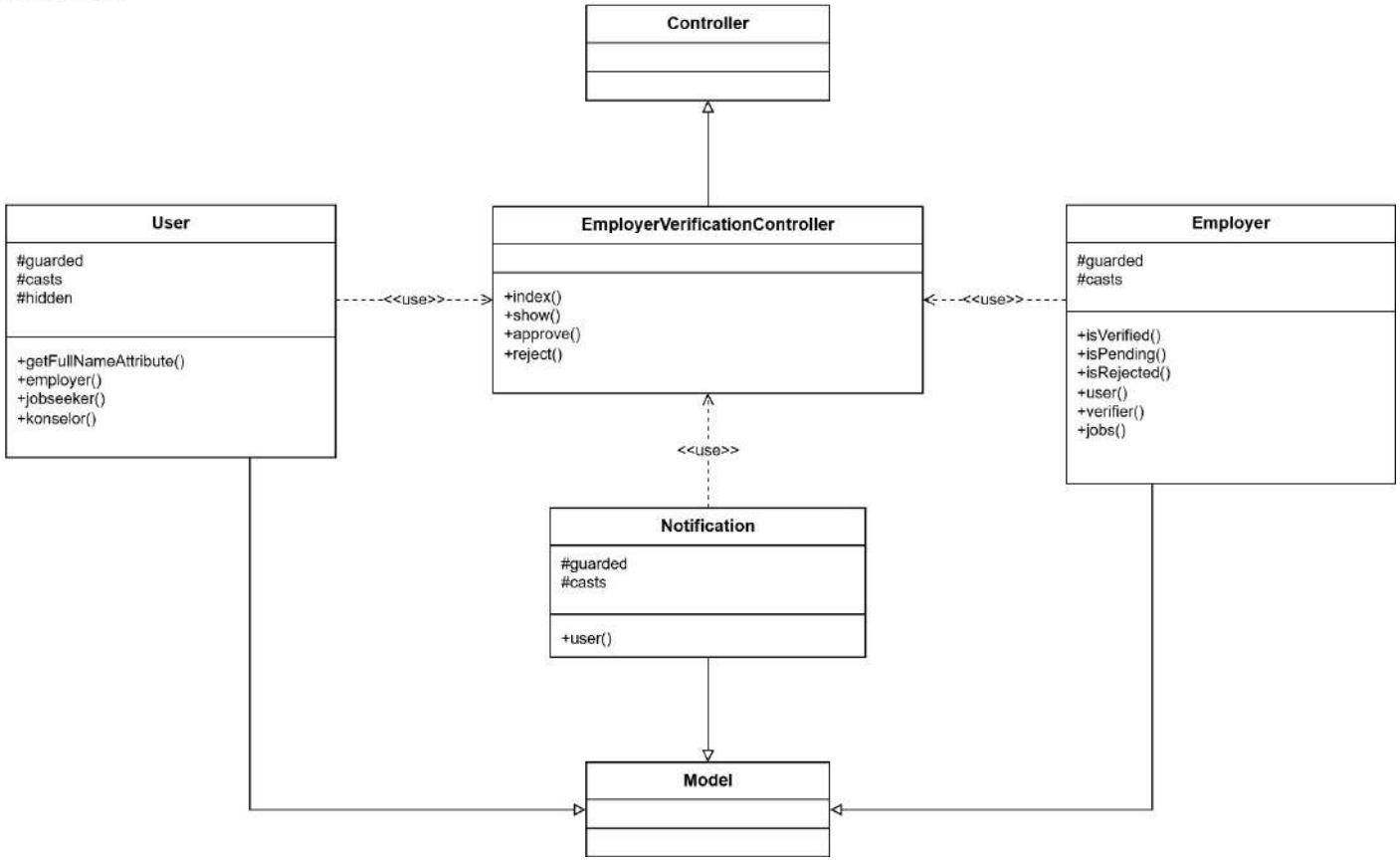


Gambar 4.15 Class Diagram Mengelola Job Fair

4.2.2.4 Class Diagram Verifikasi Employer

Proses verifikasi *employer* dijalankan melalui *class* EmployerVerificationController, sebuah kelas yang bertanggung jawab untuk mengimplementasikan logika pemrograman yang terlibat dalam interaksi dengan *database*. *Class* EmployerVerificationController merupakan *extend* dari *Class* Controller. Pada *class* ini, menggunakan tiga *class* yang diimplementasikan sebagai *extend* dari kelas *model* diantaranya Employer, User, dan Notification. *Class* Employer menyimpan data perusahaan beserta status verifikasinya, sedangkan *class* User menyimpan data akun pengguna yang terhubung dengan employer tersebut. *Class* Notification digunakan untuk menyimpan pemberitahuan yang dikirimkan kepada *employer* ketika status verifikasinya berubah. *Class* diagram verifikasi *employer* dapat dilihat pada Gambar 4.16.

Class Diagram — Verifikasi Employer



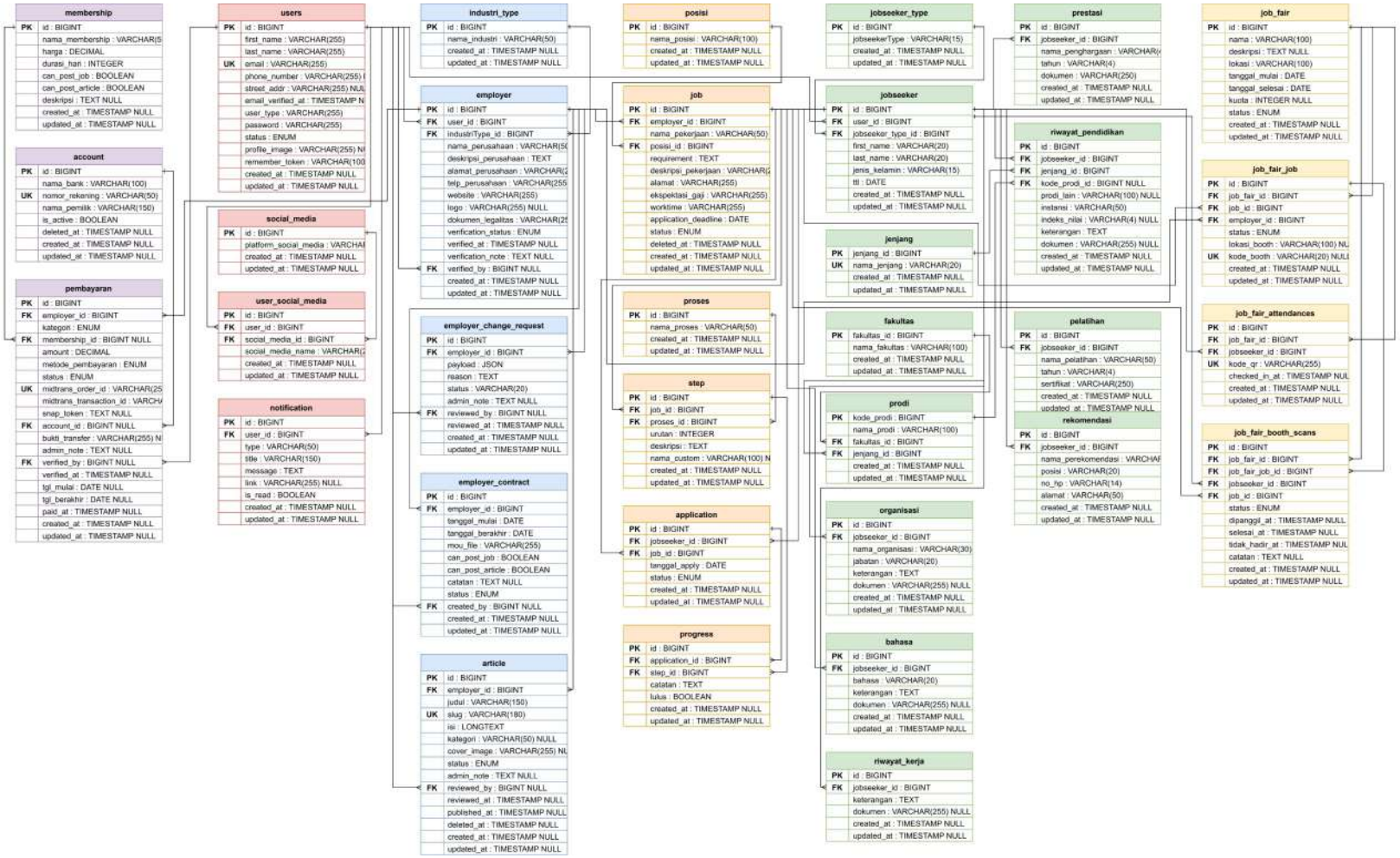
Gambar 4.16 Class Diagram Verifikasi Employer

4.2.3 ERD

Basis data merupakan kumpulan data yang terorganisir dan disimpan secara sistematis sehingga dapat diakses serta dikelola menggunakan perangkat lunak basis data. Pada perancangan basis data, keterhubungan antar entitas digambarkan melalui *Entity Relationship Diagram* (ERD). ERD memodelkan entitas, atribut, serta relasi antar entitas yang menjadi dasar pembentukan tabel pada basis data.

Basis data Sistem Informasi Rekrutmen ini dirancang untuk menyimpan data pengguna beserta perannya, data perusahaan (*employer*) dan lowongan yang diposting, data pelamar (*jobseeker*) beserta lamaran dan tahapan seleksinya, data *job fair* beserta kehadiran dan interaksi *booth*, serta data *membership* dan pembayaran. Entitas-entitas tersebut saling terhubung, misalnya satu *employer* dapat memiliki banyak lowongan, satu lowongan dapat menerima banyak lamaran, dan satu lamaran memiliki catatan *progress* pada setiap tahap seleksinya.

Basis data pada segmen rekrutmen ini di antaranya terdiri atas tabel users, employer, jobseeker, job, posisi, application, proses, step, progress, job_fair, job_fair_job, job_fair_attendances, notification, dan job_fair_booth_scans, serta tabel master data akademik berupa jenjang, fakultas, dan prodi yang menjadi acuan bagi data riwayat pendidikan pelamar. ERD Sistem Informasi Rekrutmen dapat dilihat pada Gambar 4.17.



Gambar 4.17 Entity Relationship Diagram (ERD) Sistem Informasi Pusat Karir

4.2.3.1 Perancangan Struktur Tabel

Struktur tabel merupakan gambaran dari setiap tabel beserta atribut dan relasinya, yang mencakup *primary key*, *foreign key*, dan tipe data dari masing-masing atribut. Berikut dijelaskan struktur tabel-tabel utama pada segmen rekrutmen Sistem Informasi Rekrutmen.

Tabel *users* menyimpan data akun seluruh pengguna sistem. Kolom *user_type* membedakan peran akun sebagai admin, *employer*, atau *jobseeker*, sedangkan *status* menandai keaktifan akun. Tabel ini menjadi acuan bagi tabel *employer* dan *jobseeker* melalui kolom *user_id*. Kolom *email* bersifat unik sehingga tidak ada dua akun dengan alamat surel yang sama, sedangkan *password* disimpan dalam bentuk terenkripsi. Kolom *profile_image* menyimpan nama berkas foto profil pengguna.

Tabel 4.7 Struktur Tabel users

Nama Attribute	Tipe Data	Keterangan
id	Big Integer	PK, Auto Increment
first_name	Varchar(255)	
last_name	Varchar(255)	
email	Varchar(255)	Unique
phone_number	Varchar(255)	Nullable
street_addr	Varchar(255)	Nullable
email_verified_at	Timestamp	Nullable
user_type	Varchar(255)	Default admin
password	Varchar(255)	
status	Enum	Default pending
profile_image	Varchar(255)	Nullable
remember_token	Varchar(100)	Nullable

created_at	Timestamp	
updated_at	Timestamp	

Tabel *employer* menyimpan data perusahaan beserta status verifikasinya. Kolom *verification_status*, *verified_at*, *verified_by*, dan *verification_note* mencatat hasil peninjauan admin, sedangkan *dokumen_legalitas* dan *logo* menyimpan berkas yang diunggah saat pengajuan. Kolom *user_id* menghubungkan perusahaan dengan akun penggunanya, sedangkan *industriType_id* merujuk pada tabel *industri_type* sebagai data acuan bidang usaha. Perusahaan hanya dapat memasang lowongan setelah berstatus terverifikasi.

Tabel 4.8 Struktur Tabel employer

Nama Attribute	**Tipe Data**	**Keterangan**
id	Big Integer	PK, Auto Increment
user_id	Big Integer	FK
nama_perusahaan	Varchar(50)	
deskripsi_perusahaan	Text	
industriType_id	Big Integer	FK ke industri_type
created_at	Timestamp	
updated_at	Timestamp	
alamat_perusahaan	Varchar(255)	
telp_perusahaan	Varchar(255)	
website	Varchar(255)	
verification_status	Enum	Default pending
verified_at	Timestamp	Nullable
verification_note	Text	Nullable

verified_by	Big Integer	FK ke users, Nullable
logo	Varchar(255)	Nullable
dokumen_legalitas	Varchar(255)	Nullable

Tabel *jobseeker* menyimpan data pelamar yang terhubung ke akun pengguna melalui *user_id*. Data pelengkap profil seperti riwayat pendidikan, bahasa, dan pengalaman disimpan pada tabel tersendiri agar setiap pelamar dapat memiliki lebih dari satu data. Kolom *jobseekerType_id* membedakan pelamar dari kalangan mahasiswa dan alumni Universitas Andalas dengan pelamar umum. Data pada tabel ini menjadi acuan pemeriksaan kelengkapan profil sebelum pelamar diizinkan melamar.

Tabel 4.9 Struktur Tabel jobseeker

Nama Attribute	**Tipe Data**	**Keterangan**
id	Big Integer	PK, Auto Increment
user_id	Big Integer	FK
first_name	Varchar(20)	
last_name	Varchar(20)	
jenis_kelamin	Varchar(15)	
ttl	Date	
jobseeker_type_id	Big Integer	FK ke jobseeker_type
created_at	Timestamp	
updated_at	Timestamp	

Tabel *job* menyimpan data lowongan yang dipasang *employer*. Kolom *status* menentukan lowongan masih dibuka atau ditutup, *application_deadline* menandai batas pelamaran, dan *deleted_at* dipakai untuk penghapusan sementara agar lamaran lama tetap tersimpan. Kolom *employer_id* menghubungkan

lowongan dengan perusahaan pemiliknya, sedangkan *posisi_id* merujuk data acuan posisi pekerjaan. Kolom *requirement* dan *deskripsi_pekerjaan* menyimpan teks panjang yang ditampilkan pada halaman detail lowongan.

Tabel 4.10 Struktur Tabel job

Nama Attribute	Tipe Data	Keterangan
id	Big Integer	PK, Auto Increment
employer_id	Big Integer	FK ke employer
nama_pekerjaan	Varchar(50)	
requirement	Text	
posisi_id	Big Integer	FK ke posisi
created_at	Timestamp	
updated_at	Timestamp	
deskripsi_pekerjaan	Varchar(255)	
alamat	Varchar(255)	
ekspektasi_gaji	Varchar(255)	
worktime	Varchar(255)	
application_deadline	Date	
deleted_at	Timestamp	Nullable
status	Enum	Default active

Tabel *application* menyimpan lamaran yang dikirim pelamar pada sebuah lowongan. Kolom *jobseeker_id* dan *job_id* menghubungkan keduanya, sedangkan *status* menandai lamaran masih diproses, diterima, atau ditolak. Kolom *tanggal_apply* mencatat waktu pengiriman lamaran. Status pada tabel ini tidak diubah secara manual, melainkan disesuaikan sistem berdasarkan hasil evaluasi seluruh tahapan seleksi pada tabel *progress*.

Tabel 4.11 Struktur Tabel application

Nama Attribute	Tipe Data	Keterangan
id	Big Integer	PK, Auto Increment
jobseeker_id	Big Integer	FK ke jobseeker
tanggal_apply	Date	
created_at	Timestamp	
updated_at	Timestamp	
job_id	Big Integer	FK ke job
status	Enum	Default pending

Tabel *proses* menyimpan data master jenis tahapan seleksi, misalnya seleksi administrasi, tes tertulis, atau wawancara. Data ini dipakai berulang oleh banyak lowongan melalui tabel *step*. Dengan memisahkan jenis tahapan ke dalam tabel tersendiri, penamaan tahapan menjadi seragam untuk seluruh lowongan dan admin dapat menambah jenis tahapan baru tanpa mengubah data lowongan yang sudah ada.

Tabel 4.12 Struktur Tabel proses

Nama Attribute	Tipe Data	Keterangan
id	Big Integer	PK, Auto Increment
nama_proses	Varchar(50)	
created_at	Timestamp	
updated_at	Timestamp	

Tabel *step* menyimpan tahapan seleksi milik sebuah lowongan beserta urutannya. Kolom *job_id* menghubungkan tahapan ke lowongan dan *proses_id* menentukan jenis tahapannya, sehingga setiap lowongan dapat memiliki susunan tahapan yang berbeda. Kolom *urutan* menentukan posisi tahapan dalam rangkaian

seleksi sehingga sistem dapat memastikan evaluasi hanya diisi secara berurutan. Kolom *deskripsi* memuat keterangan tambahan yang ditampilkan kepada pelamar.

Tabel 4.13 Struktur Tabel step

Nama Attribute	Tipe Data	Keterangan
id	Big Integer	PK, Auto Increment
job_id	Big Integer	FK ke job
proses_id	Big Integer	FK ke proses
deskripsi	Text	
nama_custom	Varchar(100)	Nullable
created_at	Timestamp	
updated_at	Timestamp	
urutan	Integer	Default 1

Tabel *progress* menyimpan hasil evaluasi pelamar pada setiap tahapan seleksi. Kolom *application_id* dan *step_id* menghubungkan hasil dengan lamaran dan tahapan, sedangkan *lulus* menjadi dasar pembaruan status lamaran secara otomatis. Kolom *catatan* menyimpan keterangan yang diberikan *employer* pada setiap tahapan. Satu lamaran dapat memiliki beberapa baris pada tabel ini sesuai jumlah tahapan seleksi yang telah dinilai.

Tabel 4.14 Struktur Tabel progress

Nama Attribute	Tipe Data	Keterangan
id	Big Integer	PK, Auto Increment
application_id	Big Integer	FK ke application
step_id	Big Integer	FK ke step
catatan	Text	
lulus	Boolean	

created_at	Timestamp	
updated_at	Timestamp	

Tabel *job_fair* menyimpan data acara *job fair* beserta jadwal dan kuota pesertanya. Kolom tanggal dipakai untuk menentukan apakah pendaftaran masih dibuka, dan kuota membatasi jumlah lowongan yang dapat didaftarkan. Kolom *lokasi* memuat tempat penyelenggaraan acara dan *status* menandai acara masih berlangsung atau telah berakhir. Tabel ini menjadi induk bagi pendaftaran lowongan, kehadiran peserta, dan antrian *booth*.

Tabel 4.15 Struktur Tabel job_fair

Nama Attribute	**Tipe Data**	**Keterangan**
id	Big Integer	PK, Auto Increment
nama	Varchar(100)	
deskripsi	Text	Nullable
lokasi	Varchar(100)	
tanggal_mulai	Date	
tanggal_selesai	Date	
status	Enum	Default draft
created_at	Timestamp	
updated_at	Timestamp	
kuota	Integer	Nullable

Tabel *job_fair_job* menyimpan pendaftaran lowongan ke sebuah acara *job fair*. Kolom *status* mencatat persetujuan admin, sedangkan data *booth* dipakai sebagai acuan pemindaian kode QR pada saat acara berlangsung. Kolom *job_fair_id* dan *job_id* menghubungkan acara dengan lowongan peserta,

sedangkan *employer_id* memudahkan penelusuran pemilik *booth* saat acara berlangsung tanpa perlu menelusuri tabel lowongan.

Tabel 4.16 Struktur Tabel job_fair_job

Nama Attribute	Tipe Data	Keterangan
id	Big Integer	PK, Auto Increment
job_fair_id	Big Integer	FK ke job_fair
job_id	Big Integer	FK ke job
employer_id	Big Integer	FK ke employer
status	Enum	Default pending
created_at	Timestamp	
updated_at	Timestamp	
lokasi_booth	Varchar(255)	Nullable
kode_booth	Varchar(255)	Nullable, Unique

Tabel *job_fair_attendances* menyimpan registrasi dan kehadiran pelamar pada acara *job fair*. Kolom *kode_qr* menjadi tiket peserta dan kolom waktu *check-in* menandai kehadiran pada lokasi acara. Kolom *job_fair_id* dan *jobseeker_id* memastikan satu pelamar hanya memiliki satu tiket untuk setiap acara. Status kehadiran pada tabel ini menjadi syarat sebelum pelamar dapat memindai kode *booth*.

Tabel 4.17 Struktur Tabel job_fair_attendances

Nama Attribute	Tipe Data	Keterangan
id	Big Integer	PK, Auto Increment
job_fair_id	Big Integer	FK ke job_fair
jobseeker_id	Big Integer	FK ke jobseeker
kode_qr	Varchar(255)	Unique

checked_in_at	Timestamp	Nullable
created_at	Timestamp	
updated_at	Timestamp	

Tabel *notification* menyimpan pemberitahuan yang dikirim sistem kepada pengguna. Kolom *type* menandai jenis kejadian, *link* mengarahkan ke halaman terkait, dan *is_read* menandai notifikasi telah dibaca. Kolom *user_id* menentukan penerima pemberitahuan sehingga setiap pengguna hanya melihat notifikasi miliknya sendiri. Notifikasi dibuat sistem secara otomatis ketika terjadi perubahan status pada lamaran, verifikasi, maupun pembayaran.

Tabel 4.18 Struktur Tabel notification

Nama Attribute	**Tipe Data**	**Keterangan**
id	Big Integer	PK, Auto Increment
user_id	Big Integer	FK ke users
type	Varchar(50)	
title	Varchar(150)	
message	Text	
link	Varchar(255)	Nullable
is_read	Boolean	Default false
created_at	Timestamp	
updated_at	Timestamp	

Tabel *membership* menyimpan paket berlangganan yang dapat dibeli *employer* beserta harga dan masa berlakunya. Paket ini menjadi acuan pada saat *employer* melakukan pembayaran. Kolom harga dan durasi menentukan biaya serta lama masa aktif paket. Data paket dikelola admin melalui *backoffice* sehingga penambahan atau perubahan paket tidak memerlukan perubahan pada program.

Tabel 4.19 Struktur Tabel membership

Nama Attribute	Tipe Data	Keterangan
id	Big Integer	PK, Auto Increment
nama_membership	Varchar(50)	
harga	Decimal(10,2)	
durasi_hari	Integer	Default 365
can_post_job	Boolean	Default false
can_post_article	Boolean	Default false
deskripsi	Text	Nullable
created_at	Timestamp	
updated_at	Timestamp	

Tabel *pembayaran* menyimpan transaksi pembelian *membership* oleh *employer*. Kolom status dan waktu verifikasi mencatat hasil peninjauan admin, sedangkan bukti transfer menyimpan berkas yang diunggah pada pembayaran manual. Kolom *employer_id* dan *membership_id* menghubungkan transaksi dengan perusahaan dan paket yang dipilih, sedangkan *account_id* menunjukkan rekening tujuan transfer. Masa aktif dihitung sejak pembayaran disetujui admin.

Tabel 4.20 Struktur Tabel pembayaran

Nama Attribute	Tipe Data	Keterangan
id	Big Integer	PK, Auto Increment
employer_id	Big Integer	FK ke employer
kategori	Enum	Default membership
membership_id	Big Integer	FK ke membership, Nullable
employer_event_id	Big Integer	FK, Nullable
amount	Decimal(10,2)	
status	Enum	Default pending
midtrans_order_id	Varchar(255)	FK, Nullable, Unique

midtrans_transaction_id	Varchar(255)	FK, Nullable
snap_token	Text	Nullable
tgl_mulai	Date	Nullable
tgl_berakhir	Date	Nullable
paid_at	Timestamp	Nullable
created_at	Timestamp	
updated_at	Timestamp	
metode_pembayaran	Enum	Default midtrans
account_id	Big Integer	FK ke account, Nullable
bukti_transfer	Varchar(255)	Nullable
admin_note	Text	Nullable
verified_by	Big Integer	FK ke users, Nullable
verified_at	Timestamp	Nullable

Tabel *job_fair_booth_scans* menyimpan antrian dan interaksi pelamar pada *booth employer*. Kolom *status* menandai tahap antrian dari menunggu, dipanggil, hingga selesai, beserta waktu setiap perpindahannya. Kolom *job_fair_job_id* menghubungkan antrian dengan *booth* lowongan tertentu, sedangkan *job_id* mencatat posisi yang diminati pelamar. Setelah interaksi selesai, sistem membentuk lamaran secara otomatis dari data ini.

Tabel 4.21 Struktur Tabel job_fair_booth_scans

Nama Attribute	Tipe Data	Keterangan
id	Big Integer	PK, Auto Increment
job_fair_id	Big Integer	FK ke job_fair
job_fair_job_id	Big Integer	FK
jobseeker_id	Big Integer	FK ke jobseeker
job_id	Big Integer	FK ke job
status	Enum	Default menunggu
dipanggil_at	Timestamp	Nullable
selesai_at	Timestamp	Nullable
tidak_hadir_at	Timestamp	Nullable
catatan	Text	Nullable
created_at	Timestamp	
updated_at	Timestamp	

Tabel fakultas menyimpan data master fakultas. Tabel ini menjadi acuan bagi tabel prodi melalui kolom fakultas_id.

Tabel 4.22 Struktur Tabel fakultas

Nama Attribute	Tipe Data	Keterangan
fakultas_id	Big Integer	PK, Auto Increment
nama_fakultas	Varchar(100)	
created_at	Timestamp	
updated_at	Timestamp	

Tabel jenjang menyimpan data master jenjang pendidikan, misalnya SMA, D3, dan S1. Tabel ini menjadi acuan bagi tabel prodi dan riwayat_pendidikan melalui kolom jenjang_id.

Tabel 4.23 Struktur Tabel jenjang

Nama Attribute	Tipe Data	Keterangan
jenjang_id	Big Integer	PK, Auto Increment
nama_jenjang	Varchar(20)	Unique
created_at	Timestamp	
updated_at	Timestamp	

Tabel prodi menyimpan data master program studi. Setiap baris merujuk ke fakultas dan jenjang melalui kolom fakultas_id dan jenjang_id, serta menjadi acuan bagi tabel riwayat_pendidikan melalui kolom kode_prodi_id.

Tabel 4.24 Struktur Tabel prodi

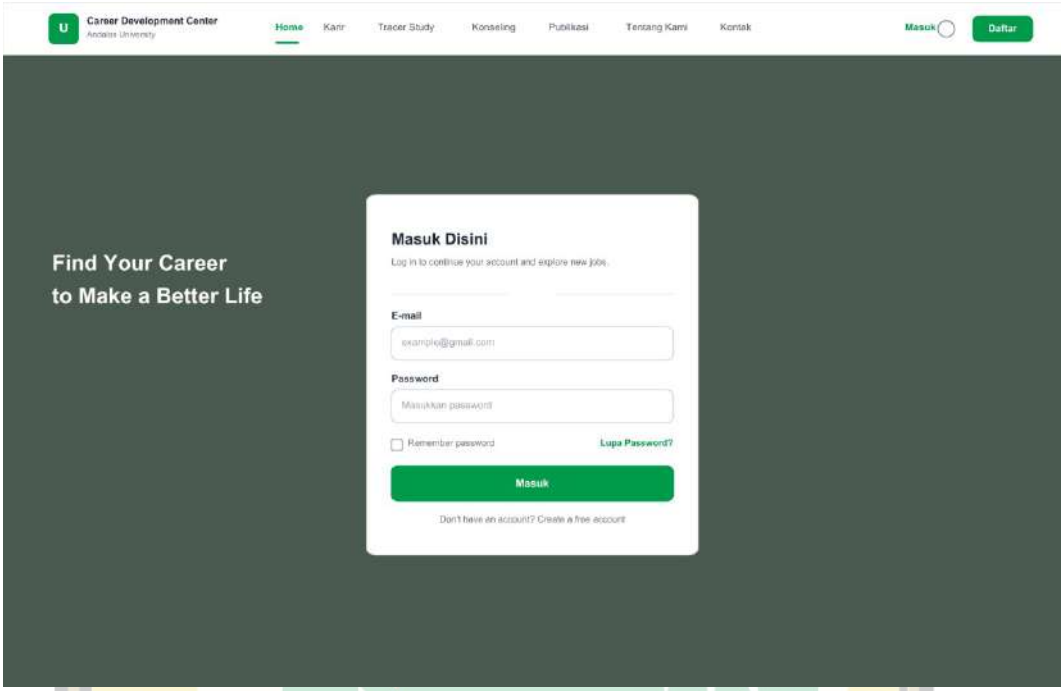
Nama Attribute	Tipe Data	Keterangan
kode_prodi	Big Integer	PK
nama_prodi	Varchar(100)	
fakultas_id	Big Integer	FK ke fakultas
jenjang_id	Big Integer	FK ke jenjang
created_at	Timestamp	
updated_at	Timestamp	

4.2.4 Perancangan Antarmuka

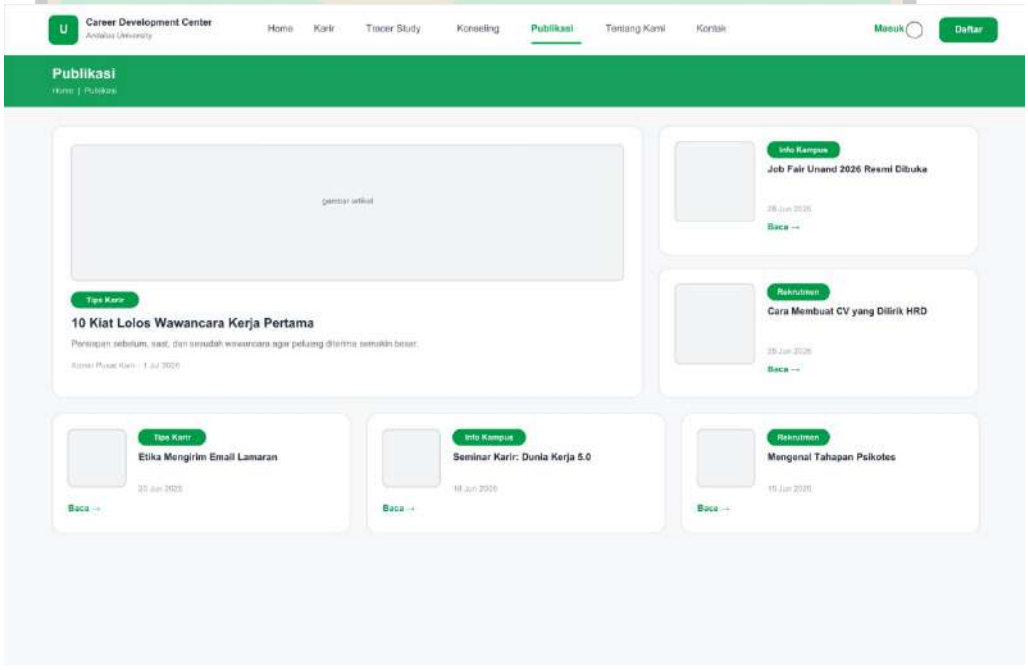
Antarmuka merupakan elemen yang memfasilitasi interaksi antara pengguna dan sistem melalui tampilan yang mudah dipahami, sehingga pengguna dapat mengoperasikan sistem dengan lebih mudah dan efektif. Berdasarkan tingkat kedetailannya (fidelity), rancangan antarmuka terbagi atas low fidelity yang berupa kerangka dasar tanpa detail visual dan high fidelity yang mendekati tampilan akhir aplikasi. Rancangan antarmuka pada penelitian ini dibuat dalam bentuk high fidelity mengikuti tema tampilan aplikasi dan digunakan sebagai acuan dalam implementasi tampilan sistem. Rancangan dikelompokkan berdasarkan pengguna sebagai berikut.

4.2.4.1 Antarmuka Umum

Antarmuka umum dapat diakses tanpa masuk ke sistem. Rancangan halaman *login* ditampilkan dalam bentuk modal yang memuat pilihan masuk melalui akun media sosial serta isian *email* dan *password*, sedangkan rancangan halaman publikasi menampilkan artikel utama beserta daftar artikel lainnya dalam bentuk kartu berkategori. Rancangan antarmuka umum dapat dilihat pada Gambar 4.18 dan Gambar 4.19.



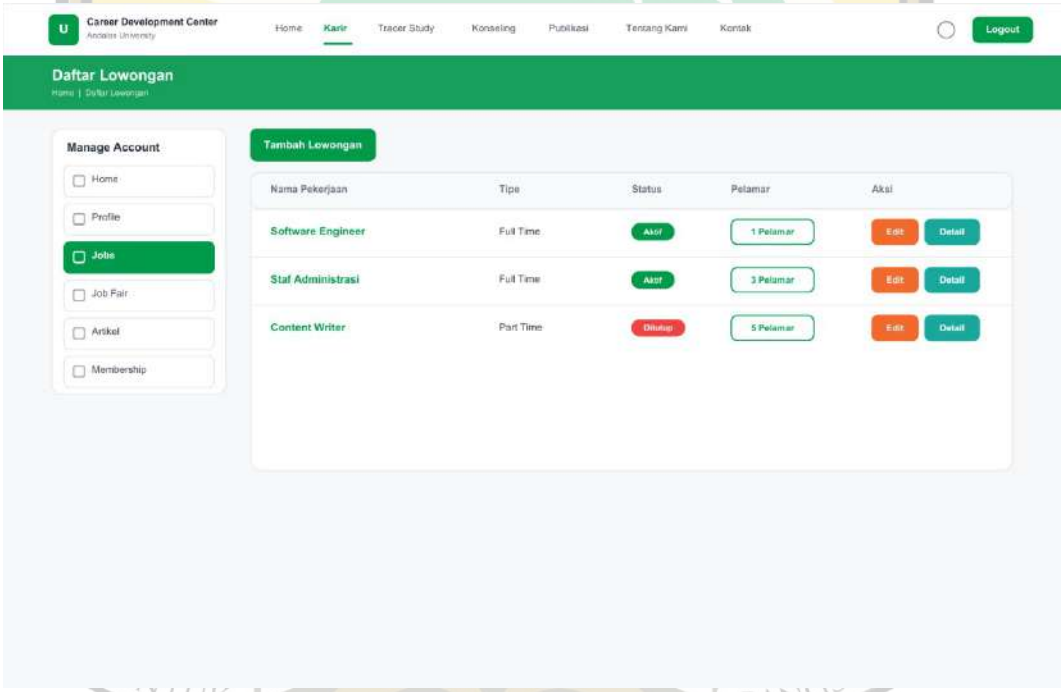
Gambar 4.18 Rancangan Antarmuka Login



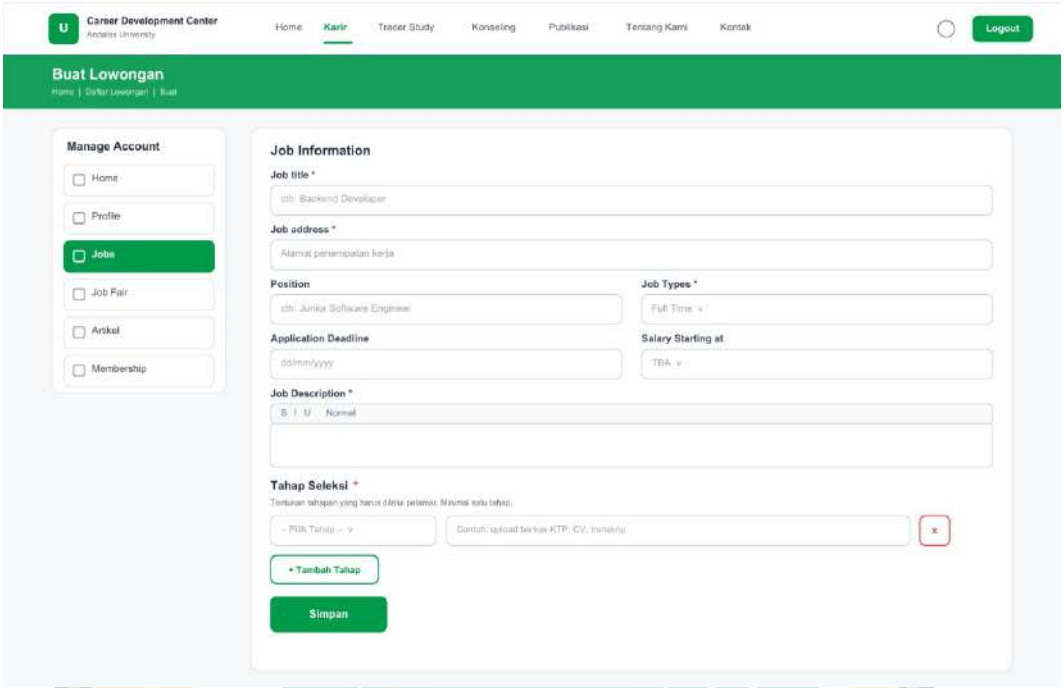
Gambar 4.19 Rancangan Antarmuka Publikasi Artikel

4.2.4.2 Antarmuka Employer

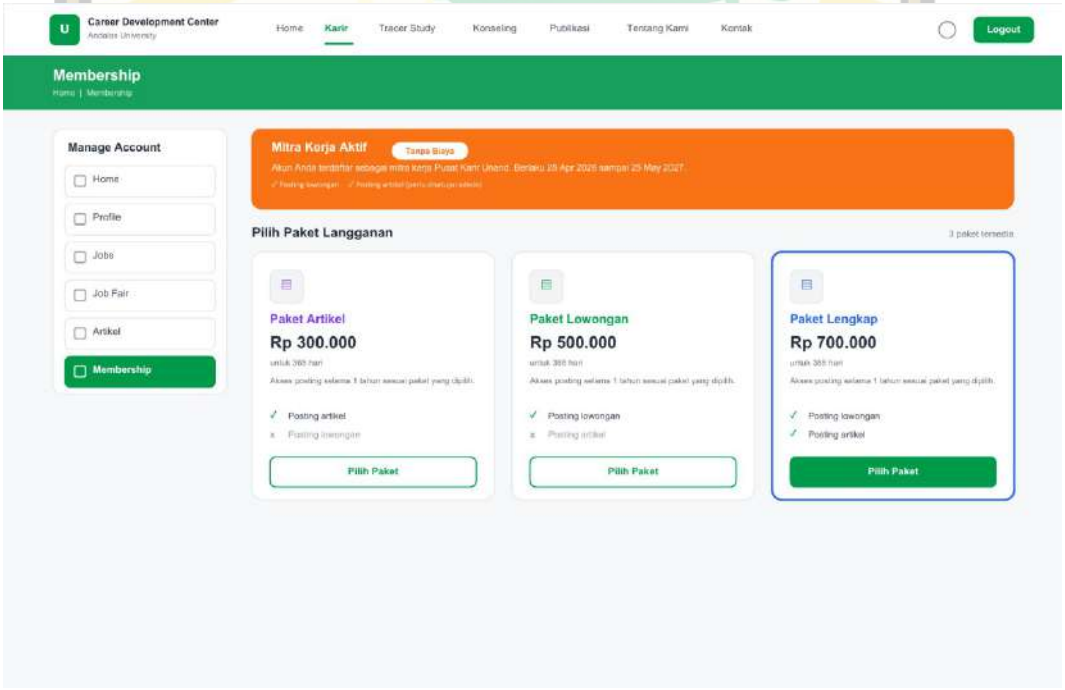
Antarmuka *employer* terdiri atas halaman daftar lowongan yang menampilkan tabel lowongan beserta status dan tombol aksi; halaman buat lowongan yang memuat form data lowongan beserta penyusunan urutan tahapan seleksi; halaman *membership* yang menampilkan pilihan paket langganan beserta form pembayaran manual dengan unggah bukti transfer; halaman *progress* seleksi yang menampilkan urutan tahapan beserta form evaluasi hasil; serta halaman antrian *booth* yang menampilkan kode QR *booth* dan daftar *jobseeker* yang dikelompokkan menurut status antrian. Rancangan antarmuka *employer* dapat dilihat pada Gambar 4.20 sampai Gambar 4.24.



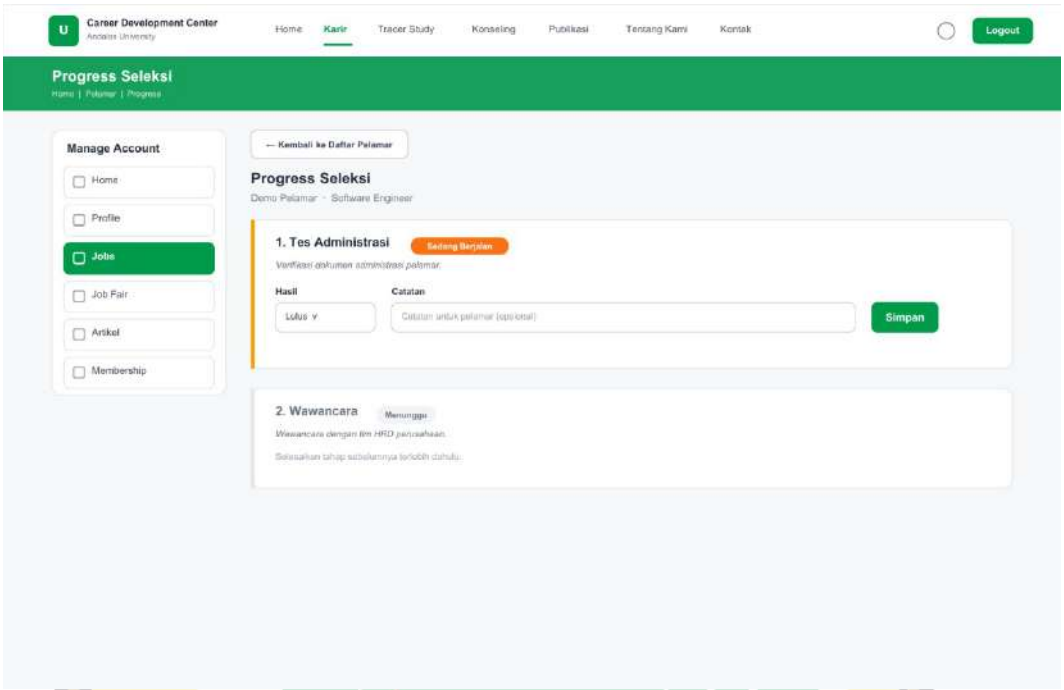
Gambar 4.20 Rancangan Antarmuka Daftar Lowongan



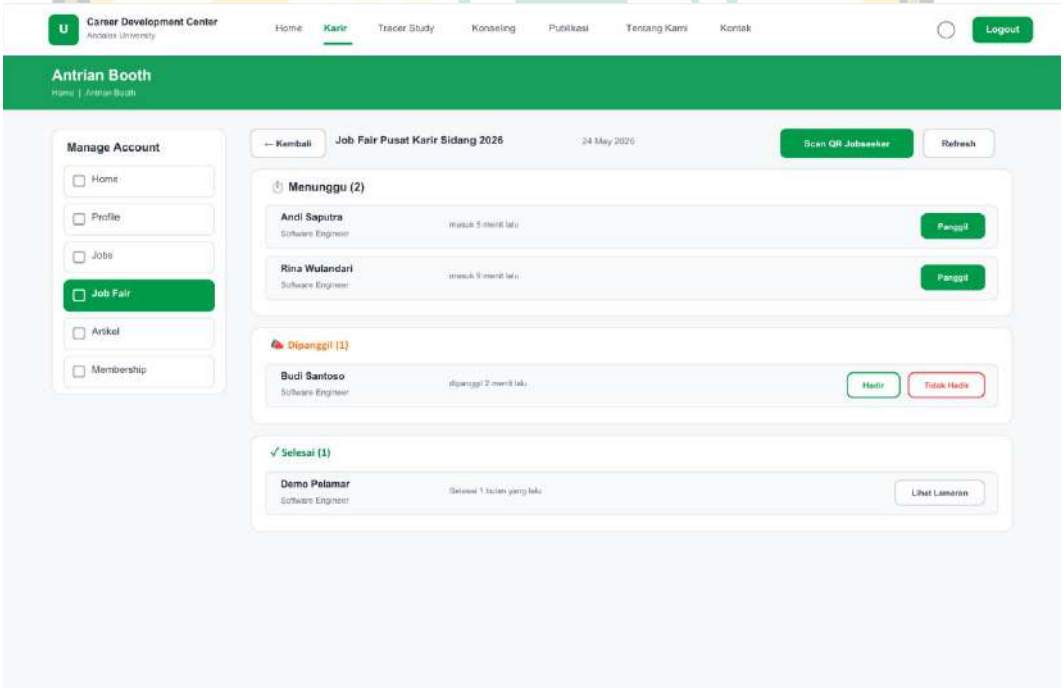
Gambar 4.21 Rancangan Antarmuka Buat Lowongan



Gambar 4.22 Rancangan Antarmuka Membership dan Pembayaran



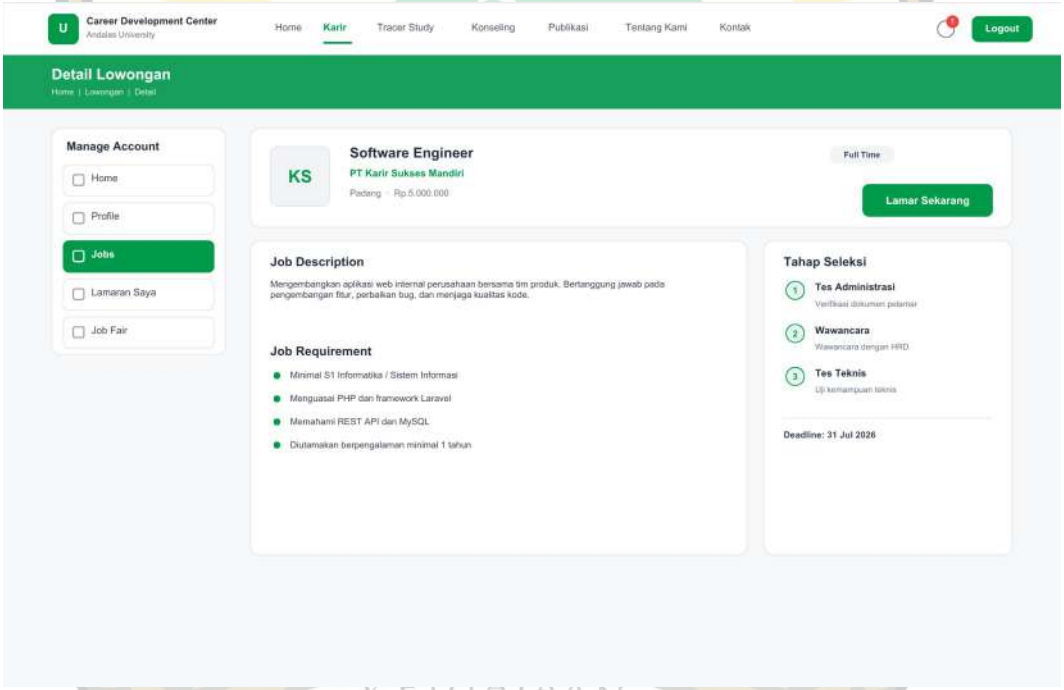
Gambar 4.23 Rancangan Antarmuka Progress Seleksi



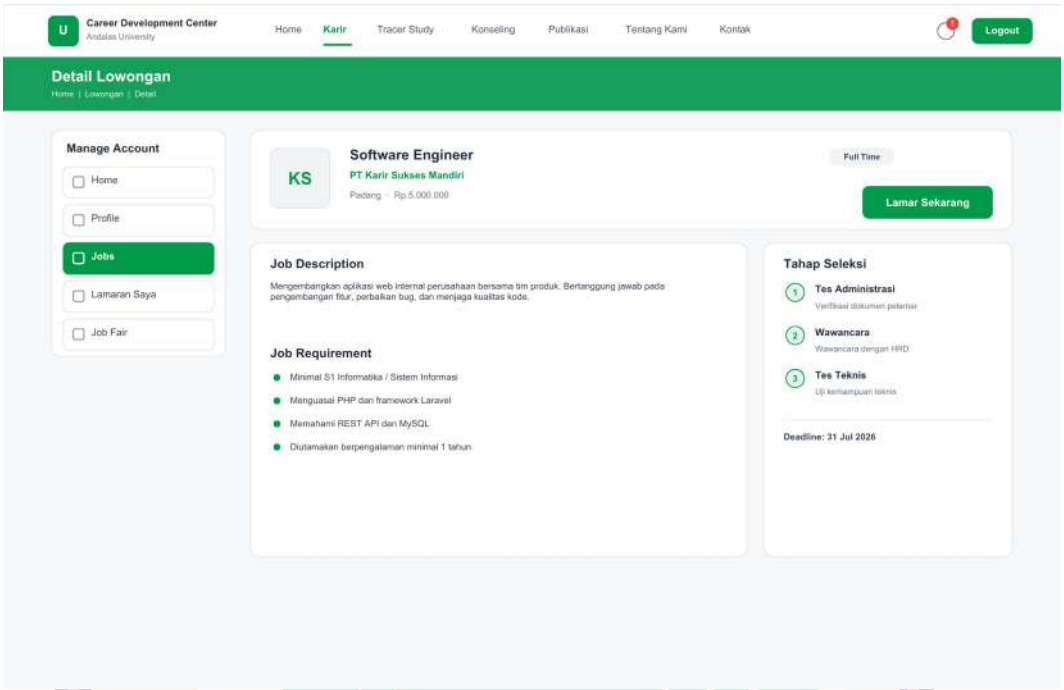
Gambar 4.24 Rancangan Antarmuka Antrian Booth Job Fair

4.2.4.3 Antarmuka Jobseeker

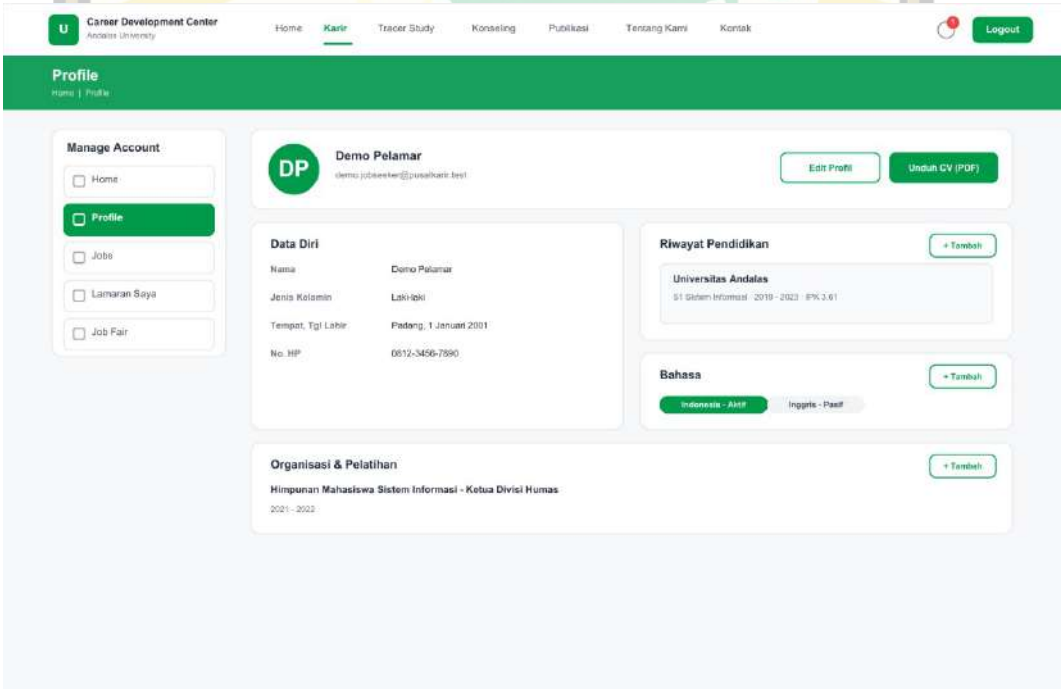
Antarmuka *jobseeker* terdiri atas halaman daftar lowongan yang menampilkan kartu lowongan beserta tombol lamar; halaman detail lowongan yang memuat deskripsi pekerjaan, persyaratan, dan tahapan seleksi; halaman profil yang menampilkan data diri, riwayat pendidikan, dan bahasa beserta tombol unduh CV; serta halaman tiket *job fair* yang menampilkan QR code personal, status check-in, form masuk antrian *booth*, dan daftar *booth* yang telah dikunjungi. Rancangan antarmuka *jobseeker* dapat dilihat pada Gambar 4.25 sampai Gambar 4.28.



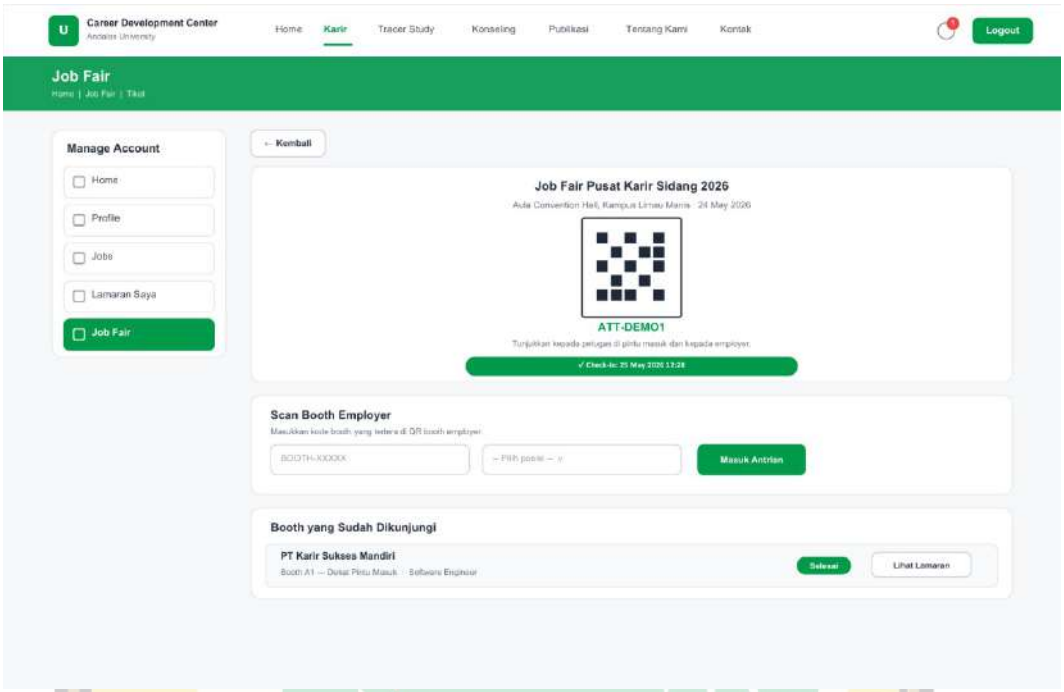
Gambar 4.25 Rancangan Antarmuka Daftar Lowongan Jobseeker



Gambar 4.26 Rancangan Antarmuka Detail Lowongan



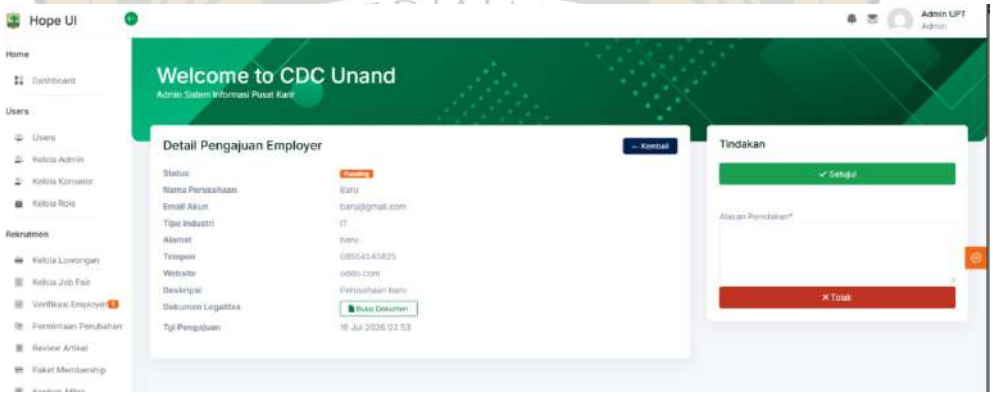
Gambar 4.27 Rancangan Antarmuka Profil Jobseeker



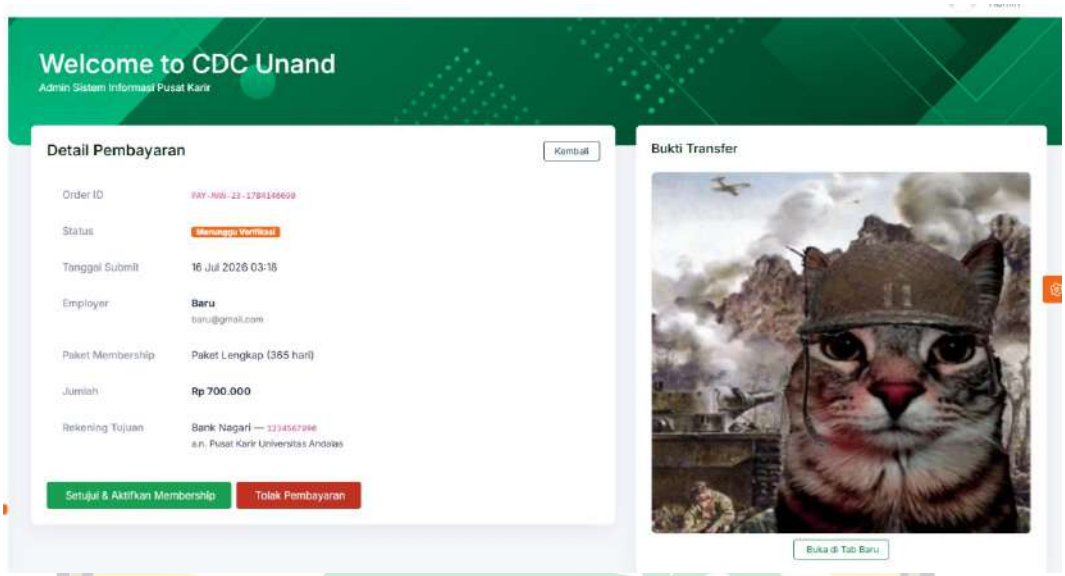
Gambar 4.28 Rancangan Antarmuka Tiket QR Job Fair

4.2.4.4 Antarmuka Backoffice Admin

Antarmuka backoffice digunakan *Admin* untuk memproses verifikasi. Rancangan halaman verifikasi *employer* menampilkan daftar pengajuan beserta detail perusahaan, dokumen legalitas, dan tombol persetujuan atau penolakan, sedangkan rancangan halaman verifikasi pembayaran menampilkan daftar pembayaran yang menunggu beserta pratinjau bukti transfer. Rancangan antarmuka backoffice dapat dilihat pada Gambar 4.29 dan Gambar 4.30.



Gambar 4.29 Rancangan Antarmuka Verifikasi Employer



Gambar 4.30 Rancangan Antarmuka Verifikasi Pembayaran