BAB 3
PELAKSANAAN KERJA MAGANG

3.1 Kedudukan dan Koordinasi

Dalam kerja magang ini, penulis menduduki posisi sebagai *frontend developer intern*. Kegiatan kerja magang dikoordinasi oleh Ibu Jesivinica Santoso selaku *Chief Executive Officer* dan *Supervisor*. Selain itu dalam pengembangan *wedding e-Invitation template* untuk Minyma E-Invitation, penulis melakukan koordinasi dengan Ibu Jesivinica Santoso selaku *Project Manager* untuk tim *frontend*, dan Bapak Daffa Grahana selaku *Leader* untuk tim UI/UX.

3.2 Tugas yang Dilakukan

Kerja magang diawali dengan memahami *requirements* yang diberikan oleh *supervisor* terkait *template* yang akan dikembangkan, dan juga memahami desain yang diberikan dari tim UI dan UX. Selama kerja magang berlangsung, tugas-tugas yang dilakukan adalah sebagai berikut:

1. Membuat *template wedding e-invitation* yang responsif dan akurat secara desain sesuai arahan klien dan tim UI/UX.
2. Membuat *template wedding e-invitation* secara per komponen berdasarkan arahan dari CTO yang bertujuan untuk meningkatkan skalabilitas.
3. Pembuatan animasi di dalam komponen menggunakan *library* seperti *Framer Motion*.
4. Penggunaan data *dummy* dari JSON untuk keperluan *testing* seandainya terdapat pengintegrasian API pada masa yang akan datang.
5. Pengintegrasian API untuk komponen *reservation*.
6. Pembuatan laman *guest management*.
7. Pembuatan laman CMS untuk *testing* skalabilitas dari *template*.

Adapun kontribusi utama dalam pelaksanaan magang ini yaitu pengembangan *template wedding e-invitation* sehingga diangkat oleh penulis sebagai bahan laporan kerja magang.

3.3 Uraian Pelaksanaan Magang

Kerja magang dilaksanakan selama 6 (enam) bulan atau 24 minggu dengan *timeline* kerja. Dapat ditunjukkan pada Tabel 3.1 dengan jadwal kerja magang sebagai berikut.

Tabel 3.1. Pekerjaan yang dilakukan tiap minggu selama pelaksanaan kerja magang

Minggu Ke -	Pekerjaan yang dilakukan
1	Mempelajari dasar dari Next.js.
2	Pembuatan *landing page* dan *company profile*.
3	Pengerjaan dasar dari laman *template*.
4	Pembuatan komponen dari laman *template* (*Hero*, *Couple*, pengerjaan *form* dan *testing* API).
5	Pembuatan komponen dari laman template (*Gift*, *Carousel*, *Footer*, *Reservation*) dan percobaan implementasi API YouTube.
6	Penambahan fitur *player* lagu dan pengintegrasian API YouTube.
7	Belajar mandiri.
8	Perombakan *template* mengikuti desain Figma.
9	Revisi *template* pada komponen *Location* dan komponen *Couple*.
10	Revisi *template* pada komponen *Photo Album* dan komponen *Couple*.
11	Belajar mandiri.
12	Pengerjaan *guest management* (1).
13	Revisi *template* pada komponen *Hero* dan *Reservation*.
14	Pengerjaan *guest management* (2)
15	Pengintegrasian API reservasi tahap pertama.
16	Pengintegrasian API reservasi tahap kedua beserta revisi.

Pada minggu pertama, merupakan tahapan dimana penulis mempelajari dasar dari Next.js. Penulis mengerjakan tutorial yang disediakan pada dokumentasi Next.js. Pada minggu kedua, sembari menunggu desain dari tim UI/UX, penulis dan setiap anggota dari tim *frontend* ditugaskan untuk membuat *landing page* dan *company profile* sesuai dengan kreativitas masing-masing. Pada minggu

ketiga, penulis melakukan pembelajaran mandiri sembari menunggu desain dari tim UI/UX. Pembelajaran yang dilakukan adalah percobaan pembuatan dari logika *templating* tersebut.

Pada minggu keempat, penulis menerima desain awal dari tim UI/UX untuk undangan pernikahan. Penulis pertama kali membuat komponen *Hero* terlebih dahulu, yang berisi foto dan nama dari *groom* dan *bride*, serta tanggal pernikahan. Setelah pengerjaan dari *Hero* selesai, penulis lanjut mengerjakan *Couple*, yang berisi foto dan nama lengkap *groom & bride* secara *individual*. Setelah itu, penulis mencoba membuat laman *form simple* untuk keperluan testing, penulis ingin mencoba apakah data pada *template* seperti nama, tanggal dan lokasi pernikahan dapat menggunakan *dummy* data yang diberikan oleh *input form* tersebut.

Pada minggu kelima, penulis melanjutkan pengerjaan *template* pada komponen *Gift*. Dimana section tersebut menampilkan nama bank dan nomor rekening dari *groom* untuk keperluan hadiah pernikahan. Setelah itu, penulis mengerjakan *Carousel* yang berisi foto-foto dari *groom & bride*. Selanjutnya penulis mengerjakan *Footer* dari undangan, yang berisi nama dan media sosial dari perusahaan. Lalu, penulis mengerjakan komponen *Reservation* versi awal dan mempelajari implementasi API YouTube untuk pemutaran lagu.

Pada minggu keenam, penulis melakukan pengerjaan pada bagian *music player*, *music player* akan aktif secara *default* ketika *user* memasuki undangan. *Player* berbentuk tombol bulat kecil pada kanan bawah komponen, yang jika ditekan akan menghentikan atau memutar lagu. Pada versi awal *music player*, lagu yang digunakan berasal dari data *local*. Pada versi berikutnya, lagu yang diputar berasal dari YouTube, hal tersebut dilakukan dengan menggunakan YouTube iFrame API.

Sembari menunggu revisi dari tim UI/UX, penulis menggunakan minggu ketujuh untuk belajar mandiri. Pada minggu kedelapan, tim UI/UX memberikan desain baru untuk undangan pernikahan klien. Terdapat revisi desain untuk menyesuaikan keinginan klien. Perubahan dilakukan hampir pada semua komponen yang ada, dari terdapat penambahan komponen seperti *Cover*, *photo album*, dan revisi menyeluruh pada section seperti *hero*, dan *location*. Lalu *couple section* juga dihilangkan, diganti dengan *groom & bride section*.

Pada minggu kesembilan, penulis melakukan revisi pada komponen *Location* dan komponen *Couple*. Revisi pada komponen *Location* berada pada card dari *location* yang tidak simetris. Dan revisi pada komponen *Couple* berada pada bagian nama dari orang tua *bride* yang perlu diganti serta spacing yang diperbaiki.

Pada minggu kesepuluh, penulis melakukan revisi pada komponen *Photo Album*, revisi dilakukan karena perubahan permintaan klien terhadap foto yang digunakan.

Sembari menunggu tugas dari *supervisor*, penulis melakukan pembelajaran mandiri untuk minggu kesebelas. Pada minggu keduabelas, penulis mendapatkan tugas di luar bagian *templating*. Penulis mendapatkan tugas berupa pembuatan laman manajemen untuk para tamu undangan. Laman tersebut digunakan untuk mengecek status kehadiran dari setiap tamu undangan, laman tersebut juga digunakan untuk melakukan *scan* QR yang sudah digenerate dan dimiliki oleh para tamu undangan. *Scan* dilakukan untuk mengkonfirmasi kehadiran para tamu secara *offline*.

Pada minggu ketigabelas, penulis mendapatkan tugas *urgent* yang berupa revisi pada laman *template*. Tepatnya pada komponen *Hero* dan *Reservation*. Karena terdapat perubahan konsep dari penyebaran undangan itu sendiri. Pada awalnya, setiap undangan memiliki nama yang unik sesuai dengan tamu yang diundang. Dikarenakan undangan pada awalnya ditujukan untuk setiap *user* secara *personalised*. Lalu konsep tersebut diubah menjadi semua *user* mendapatkan satu undangan yang sama. Tidak terdapat nama penerima lagi. Karena itu, komponen *Reservation* juga membutuhkan perubahan pada input form yang digunakan. Terdapat penambahan atribut nomor telepon.

Pada minggu keempatbelas, setelah penulis menyelesaikan revisi pada laman *template*, penulis kembali mengerjakan laman *attendance management*. Kali ini, penulis melakukan penambahan fitur dimana laman bisa membuka kamera untuk melakukan *scan QR Image* yang ada. Tetapi penulis belum sampai pada tahap dimana QR bisa dilakukan *scan* dan terhubung kepada bagian backend.

Pada minggu kelimabelas, penulis melakukan komunikasi kepada tim *backend* mengenai API yang dibutuhkan *user* untuk melakukan reservasi. Penulis memberikan list kepada *backend* yang berisi *requirement* dari input form yang digunakan beserta tipe datanya. API pertama yang selesai dibuat oleh tim *backend* adalah API *confirmation*, dimana API bertipe POST tersebut mengirim data yang disubmit oleh *user* untuk disimpan kedalam *database*. API tersebut juga memberikan respon berupa string QR, yang akan dikonversi menjadi gambar QR dan ditampilkan untuk *user*. Setelah API tersebut selesai, penulis melakukan integrasi pada komponen *Reservation* tersebut.

Pada minggu keenambelas, penulis melakukan integrasi untuk API kedua yaitu *check-phone*. API *check-phone* digunakan pada saat *user* selesai menginput nomor telepon. Setelah nomor telepon dikirim, maka API yang bertipe GET

tersebut akan mengecek *database*, apakah nomor telepon sudah teregistrasi (sudah pernah melakukan reservasi), ataupun belum. Jika belum, maka *user* akan mendapatkan input form tambahan, dimana terdapat jumlah hadirin dan nama yang perlu diisi. Jika *user* sudah pernah melakukan reservasi sebelumnya dan *user* memilih untuk menghadiri acara, maka laman akan menampilkan QR yang sudah digenerate, jika *user* memilih tidak menghadiri acara maka laman akan menampilkan pesan konfirmasi.

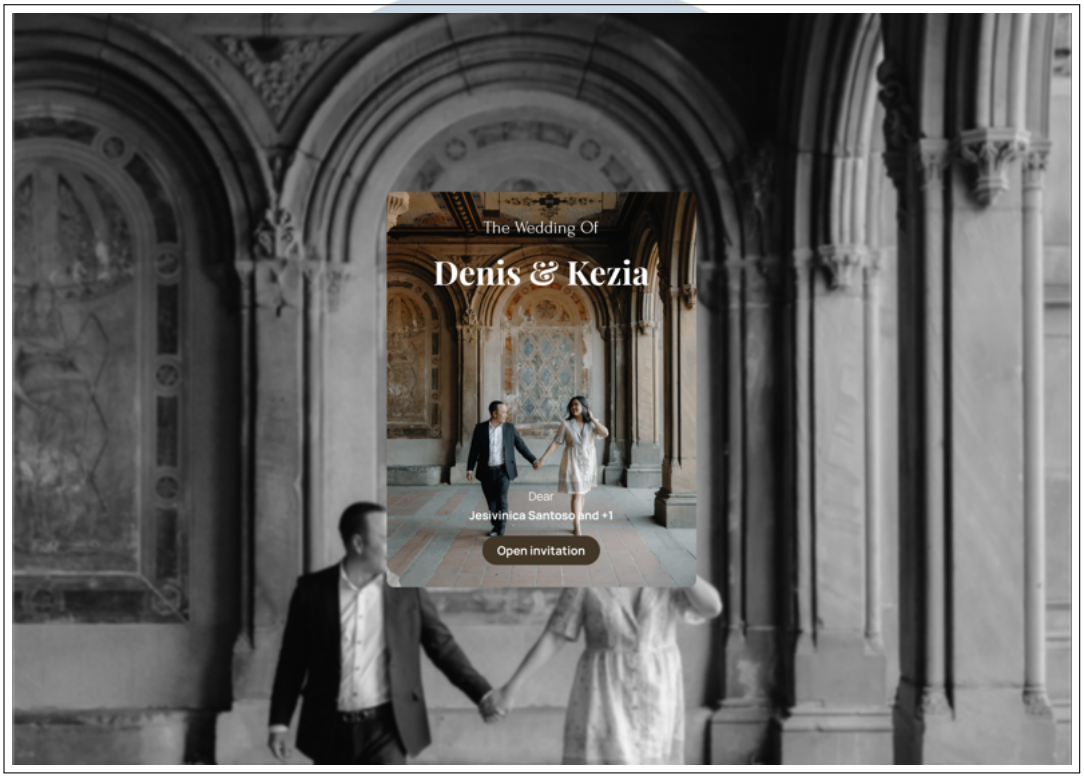
3.4 Analisis dan Perancangan Sistem

Proyek *template* dibagi menjadi dua bagian, yaitu undangan pra-revisi dan undangan final. Undangan pra-revisi merupakan versi awal undangan sebelum terdapat permintaan revisi dari klien. Sedangkan undangan final merupakan versi akhir dari undangan, dan merupakan versi undangan yang disebar kepada tamu undangan.

3.4.1 Proyek Undangan Pra-Revisi

Terdapat beberapa perbedaan dari undangan pra-revisi dan undangan final. Perbedaan pertama ialah konsep desain yang dirombak secara total, dan perbedaan kedua ialah perbedaan konsep mengenai alur undangan secara keseluruhan pada versi pra-revisi dan versi final. Undangan pra-revisi memiliki flow penyebaran undangan yang *personalised* untuk setiap tamu undangan. Setiap tamu undangan akan memiliki undangan *individual* dengan nama tamu tersebut. Sedangkan pada undangan final, setiap tamu mendapatkan satu undangan universal yang sama. Undangan pra-revisi memiliki 10 komponen terpisah, yang terdiri dari:

A *Cover*



Gambar 3.1. Komponen *Cover* Pra-Revisi

Gambar 3.1 menunjukkan komponen pertama yang dibuat untuk *template* yaitu *cover*. *Cover* merupakan komponen pertama yang dilihat oleh *user* saat pertama kali membuka undangan. *Cover* terdiri dari gambar *background* yang menampilkan *groom & bride* dengan warna *grayscale*, lalu terdapat persegi kecil pada tengah komponen yang diisi dengan gambar *couple*, lengkap dengan nama *groom & bridenya*. Pada persegi kecil tersebut terdapat nama dari tamu undangan, dan sebuah tombol *call to action* yang bertuliskan *"Open Invitation"*. Jika tombol tersebut ditekan, maka *user* akan berpindah ke komponen selanjutnya, yaitu *Hero*.

B *Hero*

Gambar 3.2. Komponen *Hero* Pra-Revisi

Pada Gambar 3.2 merupakan komponen *Hero*. Komponen *Hero* menampilkan foto dari pasangan pernikahan, disertai informasi penting dari acara pernikahan seperti titel dari acara, pasangan, dan tanggal dari pernikahan. Penempatan konten pada komponen *Hero* ditujukan untuk menyampaikan gambaran identitas acara secara ringkas namun informatif, agar *user* dapat langsung memahami konteks dari acara yang mereka terima undangannya.

C *Couple*

Gambar 3.3. Komponen *Couple* Pra-Revisi

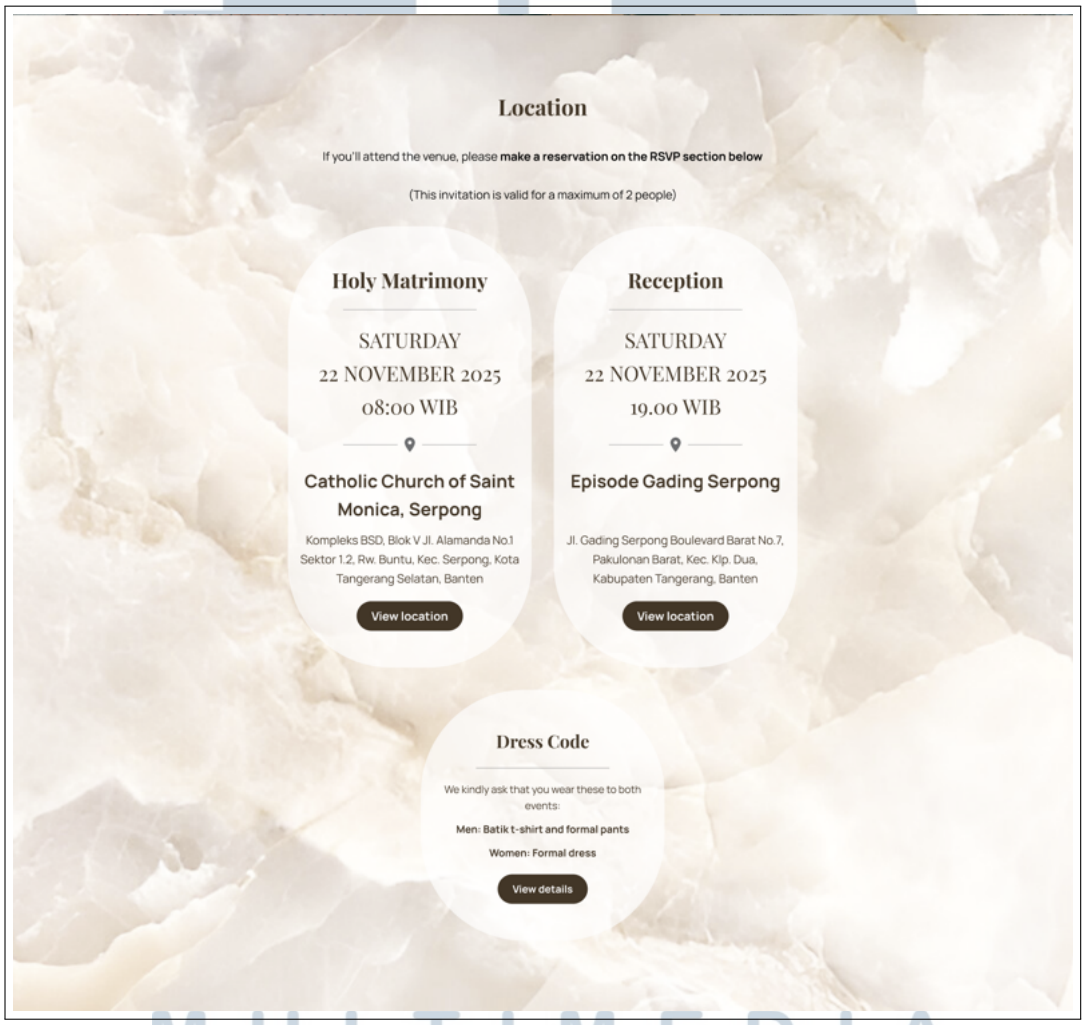
Komponen *Couple* pada Gambar 3.3 berperan sebagai elemen dalam undangan digital yang menyajikan informasi pribadi mengenai kedua mempelai kepada para tamu. Selain itu, masing-masing mempelai diperkenalkan secara terpisah melalui foto diri yang ditata secara simetris. Foto tersebut ditampilkan menggunakan *rounded corner* dan *border*. Terdapat juga nama lengkap dari pengantin pria dan wanita, disertai nama dari orang tua masing-masing pasangan.

D *Photo Album*

Gambar 3.4. Komponen *Photo Album* Pra-Revisi

Gambar 3.4 memperlihatkan komponen *Photo Album* yang dirancang untuk menampilkan kumpulan foto *prewedding* pasangan pengantin sebagai unsur estetika sekaligus narasi dalam undangan. Komponen ini memuat tiga foto vertikal statis dengan tinggi yang seragam, guna menciptakan keseimbangan visual serta menambah nuansa elegan dan personal.

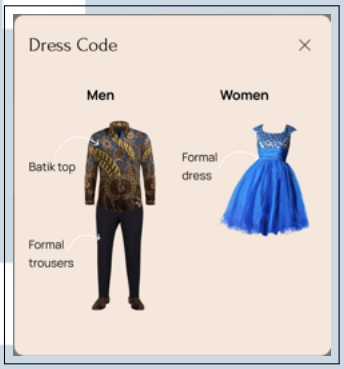
E *Location & Dresscode*



Gambar 3.5. Komponen *Location & Dresscode* Pra-Revisi

Pada Gambar 3.5 merupakan komponen *Location & Dresscode*. Komponen *Location & Dresscode* dibuka dengan himbauan mengenai keharusan reservasi jika tamu tersebut ingin datang ke acara pernikahan, lalu terdapat imbauan mengenai jumlah reservasi yang dibatasi untuk 2 orang. Setelah itu, terdapat 3 *rounded cards*. 2 *cards* pertama, berisi rincian dari lokasi acara akad dan resepsi. Rincian

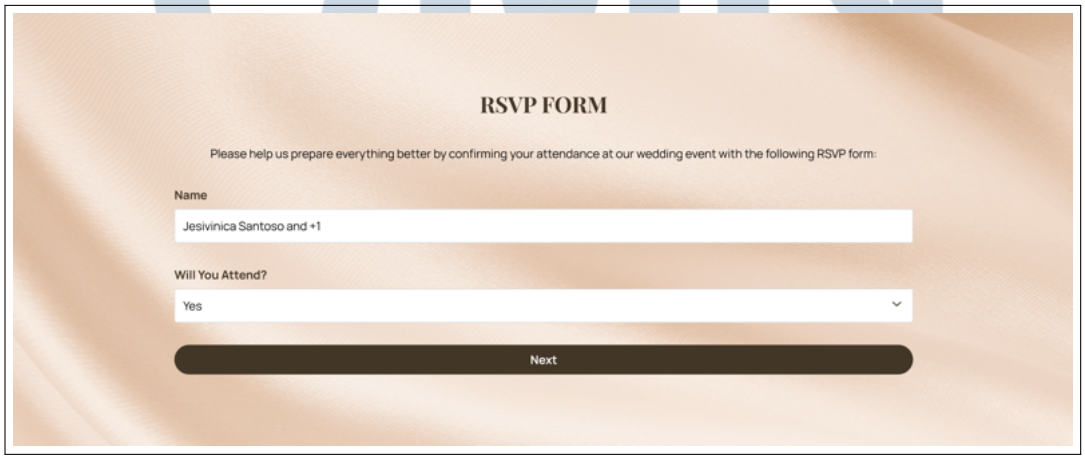
terdiri atas tanggal dari acara tersebut, nama gedung, dan alamat lengkap dari lokasi. Terdapat tombol *view location* yang jika ditekan, akan mengarahkan *user* kepada laman Google Maps dari lokasi acara. Pada *card* terakhir, berisi imbauan mengenai *dresscode* untuk acara tersebut dan terdapat tombol yang jika ditekan, akan memunculkan sebuah *pop-up* yang berisi gambaran mengenai *dresscode*. Hal tersebut ditunjukkan pada Gambar 3.6.



Gambar 3.6. *Pop-Up Dresscode* Pra-Revisi

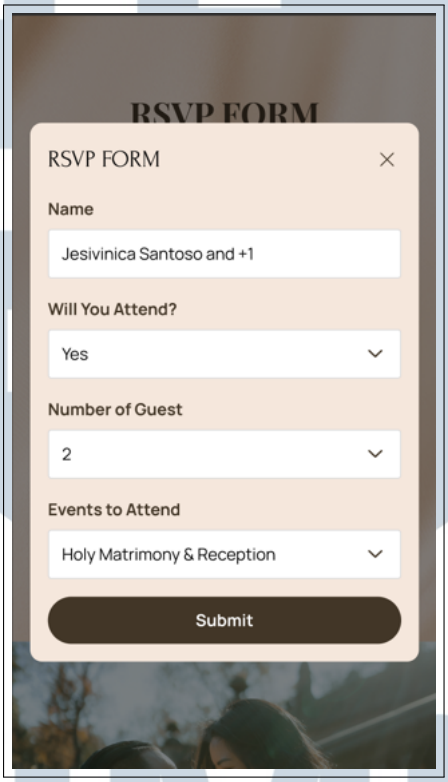
Gambar 3.6 menunjukkan *pop-up* jika tombol *"View Details"* pada card *dresscode* ditekan. *Pop-up* menampilkan busana yang dikenakan untuk tamu hadirin pria dan wanita. Saat ini, pop-up menampilkan himbauan untuk pria agar mengenakan batik sebagai atasan dan celana formal untuk bawahan. Wanita dihimbau untuk mengenakan pakaian formal.

F *Reservation*



Gambar 3.7. Komponen *Reservation* Pra-Revisi

Tepat dibawah komponen *Location & Dresscode*, terdapat komponen *reservation*. Seperti pada Gambar 3.7, komponen *Reservation* berisi form yang meminta nama dari *user*, dan status dari kehadiran *user* tersebut (*yes* atau *no*). Jika *user* memilih *yes*, maka *user* akan berpindah ke komponen selanjutnya yang ditampilkan pada Gambar 3.8.

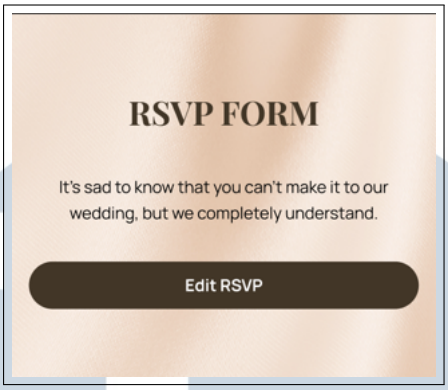


Gambar 3.8. Komponen *Reservation* Pra-Revisi jika *user* menghadiri

Gambar 3.8 menjelaskan bahwa form akan meminta *user* untuk melengkapi lebih lanjut detail kehadiran. *User* diminta untuk menginput jumlah tamu yang akan datang, tamu dibatasi sejumlah 2 sesuai permintaan klien. Setelah itu, *user* akan memilih acara yang didatangi, pilihan terdiri dari akad, resepsi, atau keduanya. Setelah itu, *user* akan diarahkan untuk mensubmit form tersebut, dan akan berpindah ke tahap selanjutnya yang akan ditampilkan pada Gambar 3.9.

Gambar 3.9. Komponen *Reservation* Pra-Revisi saat *user* selesai melakukan reservasi

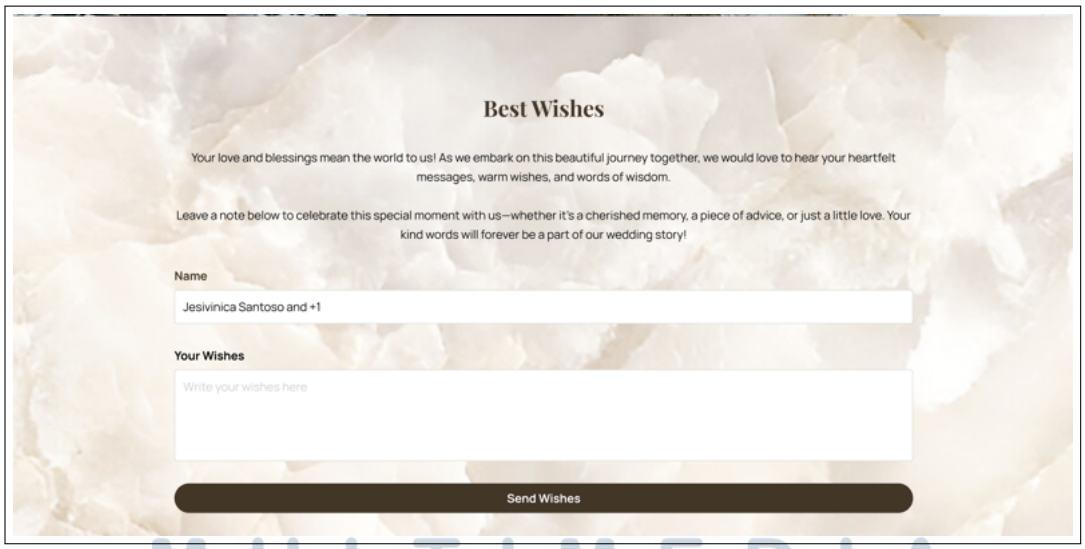
Ditunjukkan pada Gambar 3.9, jika *user* sudah mensubmit reservasi, maka laman akan menampilkan *QR code* yang berisi status reservasi dari *user* tersebut. *QR* akan digunakan saat *user* menghadiri acara pada hari H untuk memvalidasi kedatangan dari tamu. Terdapat tombol unduh yang digunakan untuk mengunduh QR tersebut, dan terdapat tombol edit RSVP yang akan mengarahkan *user* ke laman reservasi awal seperti pada Gambar 3.10.



Gambar 3.10. Komponen *Reservation* Pra-Revisi jika *user* tidak menghadiri

Gambar 3.10 menunjukkan tampilan laman jika *user* tidak menghadiri acara. Jika pada laman awal *Reservation*, *user* memilih untuk tidak menghadiri acara, maka laman akan menampilkan pesan dan tombol edit RSVP yang dapat digunakan untuk merubah status kehadiran. Jika tombol edit RSVP ditekan, maka *user* akan berpindah ke laman reservasi awal seperti pada Gambar 3.11.

G *Best Wishes*



Gambar 3.11. Komponen *Form Input Best Wishes*

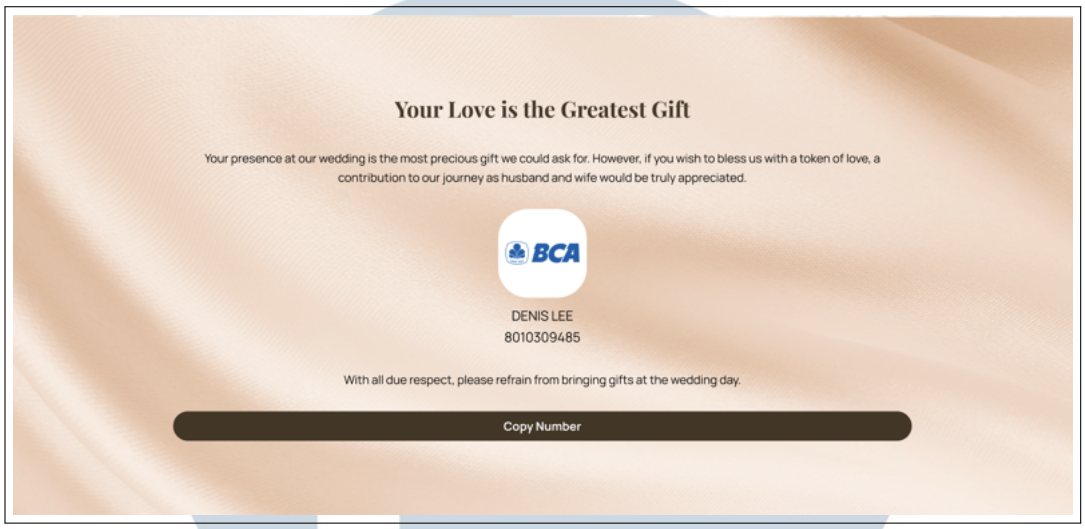
Gambar 3.11 menunjukan komponen *Best Wishes*. Komponen *Best Wishes* merupakan laman dimana *user* dapat memberikan pesan kepada pasangan pengantin. Laman tersebut berisi form yang meminta input nama dari *user*, serta pesan dari *user*. Jika *user* sudah selesai menginput pesan, *user* akan diarahkan

untuk mengirim pesan tersebut. Pesan yang dikirim oleh *user* akan ditampilkan bersama pesan-pesan lainnya yang dikirim tamu undangan lain.

Gambar 3.12. Komponen Pesan *Best Wishes*

Seperti yang ditunjukkan Gambar 3.12, setiap entri komentar menampilkan informasi penting berupa nama pengirim yang jelas teridentifikasi dan pesan yang ditujukan kepada pasangan pengantin. Laman komentar berada persis dibawah input form pesan tersebut. Untuk mengantisipasi volume pesan yang berpotensi sangat besar, terutama pada pernikahan dengan jumlah tamu yang banyak, sistem *pagination* telah diimplementasikan dengan navigasi halaman yang intuitif (Previous, 1, 2, 3, ..., Next), memungkinkan *user* untuk menjelajahi seluruh koleksi pesan tanpa mengalami masalah *loading* yang berlebihan atau antarmuka yang terlalu padat.

H *Gift*



Gambar 3.13. Komponen *Gift* Pra-Revisi

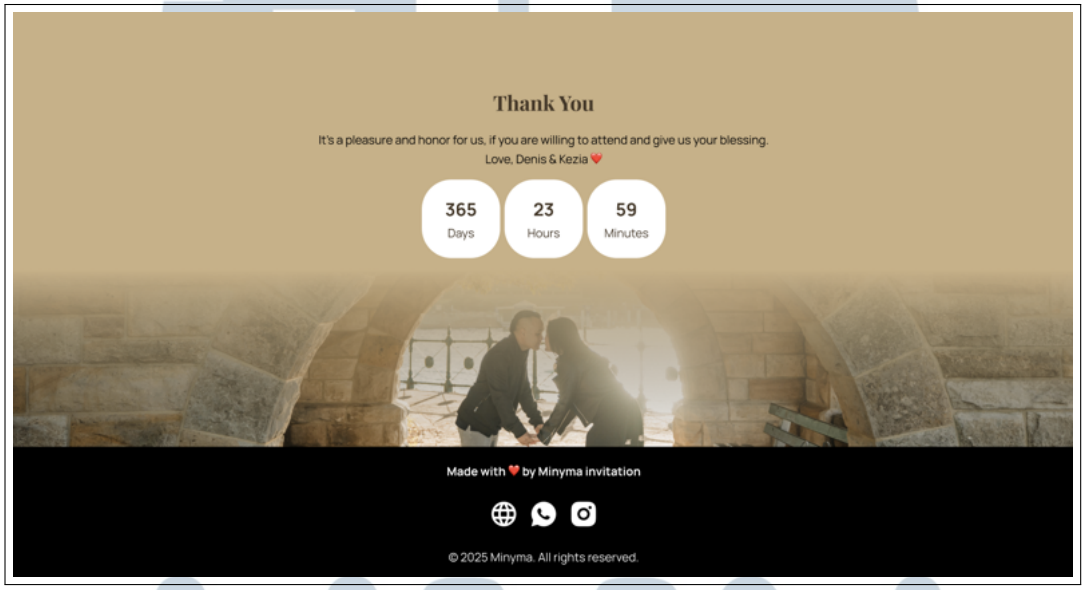
Gambar 3.13 menunjukkan komponen *Gift*. Komponen *Gift* ditujukan untuk *user* yang ingin memberikan hadiah kepada pasangan pengantin. Laman diawali dengan pesan sambutan, lalu dilanjutkan dengan logo bank dari rekening mempelai pria, lengkap dengan nama pemilik rekening dan nomor rekening. Lalu terdapat tombol "*Copy Number*" yang memungkinkan *user* untuk menyalin nama pemilik rekening dan nomor rekening.

I *Carousel*

Gambar 3.14. Komponen *Carousel*

Gambar 3.14 memperlihatkan komponen *Carousel*, sebuah elemen interaktif yang memungkinkan pengguna menelusuri dokumentasi foto *prewedding* pasangan pengantin dengan cara yang lebih dinamis. Fitur ini menampilkan total sepuluh gambar yang bergulir secara otomatis dari kanan ke kiri, memberikan kesan visual yang hidup serta pengalaman yang imersif dan berkesinambungan bagi pengguna.

J *Thank You & Footer*



Gambar 3.15. Komponen *Thank You* & Komponen *Footer* Pra-Revisi

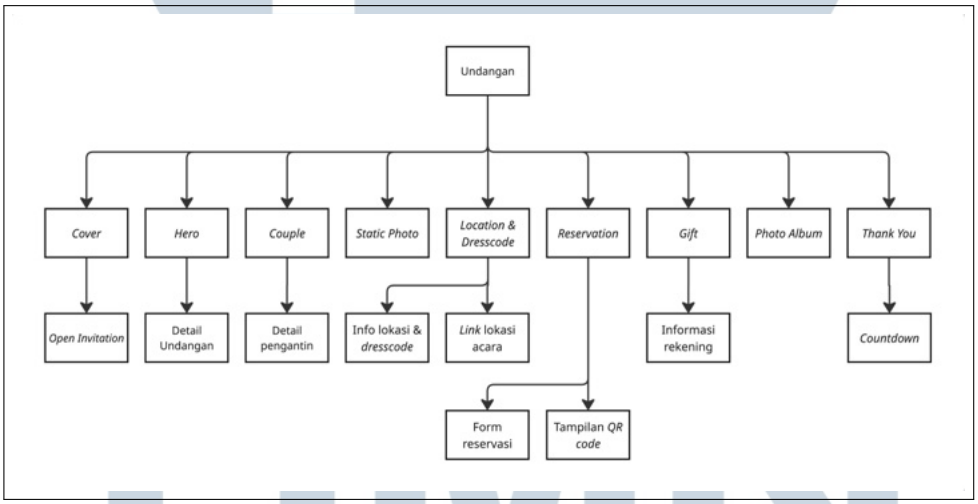
Pada Gambar 3.15 ditunjukkan komponen *Thank You & Footer*. Komponen *Thank You* menampilkan ucapan terima kasih dari pasangan pengantin disertai dengan *countdown* untuk acara pernikahan. Komponen tersebut juga memiliki latar belakang berupa foto dari pasangan disertai dengan gradasi coklat keemasan. Pada bagian penutup halaman, terdapat komponen *Footer* yang memuat informasi mengenai pihak penyedia jasa undangan digital. Selain itu, ditampilkan pula ikon-ikon yang mengarahkan ke media sosial seperti situs web, WhatsApp, dan Instagram, serta mencantumkan keterangan hak cipta tahun 2025 sebagai upaya memperkuat identitas merek dan legalitas layanan yang disediakan.

3.4.2 Proyek Undangan Final

Undangan final merupakan undangan yang sudah mendapatkan approval final dari klien dari segi desain. Dan undangan final juga merupakan versi

akhir undangan yang ditujukan sebagai undangan yang akan disebar kepada para *user*. Terdapat beberapa perbedaan dari undangan final jika dibandingkan dengan undangan pra-revisi. Dari segi desain, hampir setiap komponen diubah secara visual mengikuti kemauan klien dan arahan dari tim UI/UX, terdapat juga komponen yang dihilangkan seperti *Best Wishes* dan *Carousel*. Dari segi teknikalitas, terdapat perubahan mengenai tata cara penyebaran undangan, jika pada undangan pra-revisi setiap tamu undangan akan memiliki undangan individual dengan nama tamu tersebut, pada undangan final, setiap tamu mendapatkan satu undangan universal yang sama. Hal tersebut ditujukan agar mengurangi kerumitan dari undangan itu sendiri, dan ubahan tersebut dilakukan berdasarkan arahan dari CTO dan CEO.

A *Sitemap Undangan*

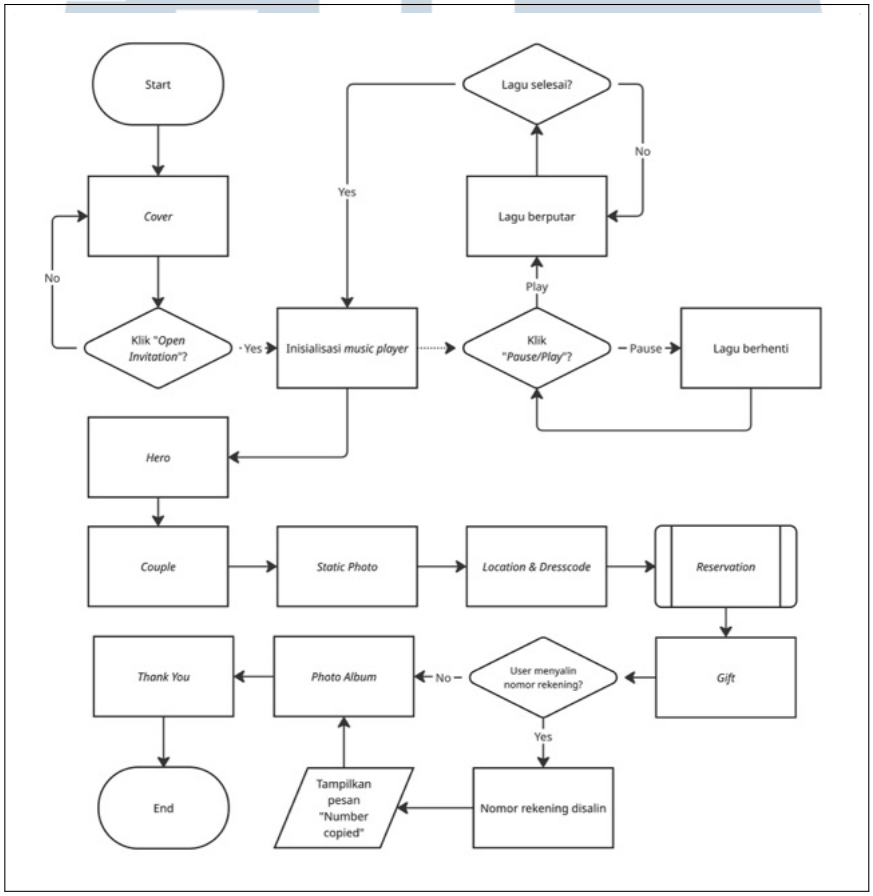


Gambar 3.16. *Sitemap* Undangan

Gambar 3.16 menunjukkan *sitemap* dari undangan. Alur dimulai dari halaman *Cover*, yang menampilkan nama dan foto pengantin, serta dilengkapi tombol untuk membuka undangan. Setelah masuk, *user* akan disambut oleh halaman *Hero* yang menyoroti nama acara dan nama kedua mempelai. Selanjutnya, halaman *Couple* menyajikan informasi lebih detail mengenai mempelai pria dan wanita, termasuk nama orang tua pengantin. Lalu terdapat halaman *Static Photo* yang menampilkan sebuah foto pengantin. Informasi inti acara kemudian dirinci pada bagian *Location & Dresscode*, yang mencakup detail untuk prosesi *Holy Matrimony* (akad), resepsi, serta aturan berbusana (*Dress Code*). Bagian interaktif utama adalah *Reservation*, di mana tamu dapat mengisi formulir RSVP dan akan

mendapatkan tampilan QR *Code* sebagai bukti konfirmasi kehadiran. Setelah itu, terdapat bagian *Gift* yang berisi informasi nomor rekening dan pesan dari pengantin. Undangan ini juga dilengkapi dengan *Photo Album* yang menampilkan enam foto pengantin. Sebagai penutup, pada bagian akhir terdapat halaman *Thank You* yang berisi ucapan terima kasih serta sebuah *countdown timer* menuju hari acara.

B *Flowchart Undangan*



Gambar 3.17. *Flowchart* Keseluruhan Undangan

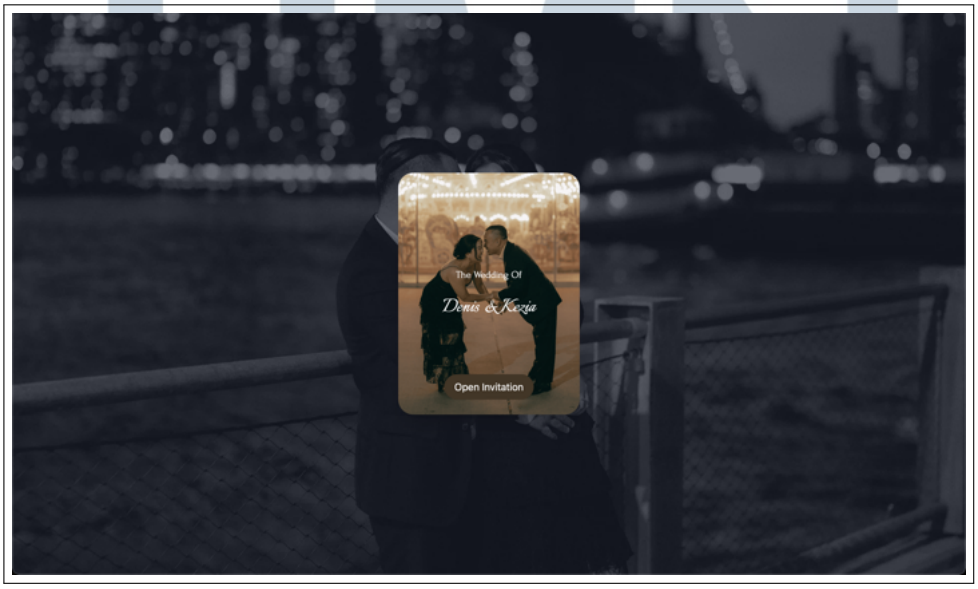
Gambar 3.17 menunjukkan alur keseluruhan undangan. Alur dimulai saat *user* membuka undangan dan disambut dengan halaman *Cover*. Terdapat kondisi di mana *user* harus menekan tombol "*Open Invitation*" untuk dapat melihat isi undangan. Jika tombol tidak ditekan, tampilan akan tetap berada di halaman *Cover*. Setelah menekan tombol, sistem akan melakukan inisialisasi music player dan secara bersamaan mulai menampilkan konten undangan. Secara paralel, lagu akan mulai berputar. *User* memiliki kontrol untuk menekan tombol "*Pause/Play*".

Jika *user* menekan ”*Pause*”, lagu akan berhenti, dan jika menekan ”*Play*”, lagu akan kembali berputar. Konten undangan kemudian ditampilkan secara berurutan, dimulai dari bagian *Hero*, diikuti oleh bagian *Couple*, *Static Photo*, dan informasi mengenai *Location & Dresscode*. Selanjutnya, ditampilkan bagian *Reservation*, di mana alur untuk melakukan reservasi kehadiran akan berjalan. Setelah bagian reservasi, terdapat bagian *Gift*. Pada bagian ini, sistem akan memeriksa apakah *user* menyalin nomor rekening. Jika ya, maka akan ada konfirmasi bahwa nomor rekening telah disalin. Alur kemudian berlanjut dengan menampilkan *Photo Album* dan diakhiri dengan halaman *Thank You*.

C Hasil Implementasi Undangan Final

Pada bagian selanjutnya akan disajikan penjelasan yang komprehensif dan detail mengenai hasil implementasi dari proyek pengembangan yang telah berhasil dicapai selama periode magang. Paparan ini dimaksudkan untuk memberikan pemahaman yang lebih komprehensif tentang tahapan kerja, fungsionalitas, serta output final dari aplikasi undangan yang telah dibuat, dengan tujuan menunjukkan kontribusi konkret yang telah diberikan sepanjang kegiatan magang.

C.1 *Cover*



Gambar 3.18. Komponen *Cover* Undangan Final

Pada komponen *Cover* di undangan final, terdapat perubahan *minor* yang ditunjukkan pada Gambar 3.18. Perubahan terdapat pada gambar latar yang digunakan. Foto latar menggunakan gambar monokromatik (hitam-putih) berskala penuh yang menampilkan siluet pasangan dengan pemandangan kota di malam hari. Serta terdapat perubahan pada nama tamu undangan yang dihilangkan. Nama tamu undangan dihilangkan atas dasar pertimbangan kompleksitas. Walaupun begitu, fungsionalitas komponen tetaplah sama. Jika *user* mengklik tombol "*Open Invitation*", maka *user* akan membuka undangan secara lengkap dan akan berpindah ke laman selanjutnya yaitu *Hero*.

```
export default function InvitationCover({ onOpenInvitation, player
    , setIsPlaying }) {
  const [isOpened, setIsOpened] = useState(false);

  useEffect(() => {
    // Play music when Cover component mounts
    if (player) {
      player.playVideo();
      setIsPlaying(true);
    }
  }, [player]); // Only run when player is available

  const handleOpenInvitation = () => {
    setIsOpened(true);
    onOpenInvitation();

    if (player) {
      player.playVideo();
      setIsPlaying(true);
    }
  };

  return (
    <div className="relative w-full h-screen flex items-center
     justify-center bg-gray-900 z-0">
      {!isOpened ? (
        <>
          <div className="absolute inset-0">
            <Image
              src="/images/Template4/Cover/backcover.png"
              alt="Background"
              layout="fill"
```

```
31                 objectFit="cover"
32                 className="opacity-40 grayscale"
33               />
34             </div>
```

Kode 3.1: Kode Komponen *Cover*

Kode 3.1 mendefinisikan komponen React bernama *InvitationCover* yang berfungsi untuk menampilkan *cover* undangan. Komponen ini menerima beberapa *props* dari komponen induknya, yaitu *onOpenInvitation* sebagai fungsi yang akan dijalankan saat undangan dibuka, *player* sebagai objek untuk mengontrol pemutar musik, dan *setIsPlaying* sebagai fungsi untuk mengubah status pemutaran musik. Secara internal, komponen menggunakan *hook useState* untuk mengelola *state isOpened* yang memeriksa apakah *user* sudah menekan tombol "*Open Invitation*". Terdapat juga hook *useEffect* yang memastikan musik dapat langsung diputar ketika komponen berhasil dimuat dan player telah siap. Fungsi utamanya, *handleOpenInvitation*, akan dipicu ketika *user* menekan tombol "*Open Invitation*", yang kemudian akan mengubah *state isOpened* menjadi *true*, memanggil fungsi *onOpenInvitation*, dan secara eksplisit memulai pemutaran musik. Secara visual, kode ini me-*render* sebuah *div* yang menampilkan gambar latar belakang dengan efek *opacity-40* dan *grayscale* yang hanya akan muncul jika undangan belum dibuka, dikontrol oleh *state isOpened*.

C.2 *Hero*

Gambar 3.19. Komponen *Hero* Undangan Final

Pada komponen *Hero* yang ditunjukkan pada Gambar 3.19, terdapat perubahan visual, dimulai dari foto latar yang berubah. Lalu informasi dari pasangan pengantin yang semula berada pada tengah laman, diubah menjadi pada kiri laman. Juga terdapat perubahan pada font yang dipakai untuk nama pasangan pengantin. Selain itu, terdapat penambahan fitur tombol *music player* pada kanan bawah laman. Tombol berfungsi untuk menghentikan atau memulai lagu. Lagu akan terputar secara *default* jika *user* memasuki komponen *Hero*. Jika *user* melakukan *scrolldown*, *user* akan beralih ke laman *Couple*.

```
export function Hero({ groomFullName, brideFullName, weddingDate
    }) {
  const date = new Date(weddingDate);
  const formattedDate = `${date.toLocaleDateString("en-US",
    { weekday: "long" })},
    ${date.getDate().toString().padStart(2, "0")}
    ${date.toLocaleDateString("en-US", { month: "long" })}
    ${date.getFullYear()}`;

  return (
    <div className="relative h-screen w-full">
      <motion.div
```

```
        initial={{ opacity: 0 }}
        whileInView={{ opacity: 1 }}
        transition={{ duration: 1.5 }}
        viewport={{ once: true }}
        className="absolute inset-0 w-full h-full bg-[url('/images/Template4/Hero/wedding.png')] bg-cover bg-center"
      >
        <motion.div
          initial={{ opacity: 0 }}
          whileInView={{ opacity: 1 }}
          transition={{ duration: 2, delay: 0.5 }}
          viewport={{ once: true }}
          className="absolute inset-0 bg-black bg-opacity-30"
        />
      </motion.div>
```

Kode 3.2: Kode Komponen *Hero*

Kode 3.2 mendefinisikan komponen *Hero*. Komponen ini menerima *groomFullName*, *brideFullName*, dan *weddingDate* sebagai *props* untuk menampilkan informasi pasangan dan tanggal pernikahan secara dinamis. Di dalam komponen, tanggal yang diterima melalui prop *weddingDate* akan diformat ulang menggunakan *toLocaleDateString* untuk mendapatkan format yang lebih mudah dibaca oleh *user*. Untuk aspek visual, komponen ini menggunakan library Framer Motion (motion.div) untuk memberikan efek animasi *fade-in* saat komponen masuk ke dalam *viewport*. Animasi ini diterapkan pada gambar latar belakang dan lapisan (*overlay*) hitam transparan yang muncul sesaat setelahnya untuk memberikan kontras pada teks. *Styling* utama, termasuk gambar latar, diatur menggunakan kelas dari Tailwind CSS, seperti *bg-cover* dan *bg-center*, untuk memastikan tampilan yang responsif di berbagai perangkat.

C.3 *Couple*

Gambar 3.20. Komponen *Couple* Undangan Final

Pada Gambar 3.20 merupakan komponen *Couple*. Komponen *Couple* ini berfungsi sebagai inti perkenalan pasangan pengantin, di mana seluruh informasi disajikan di atas latar belakang bertekstur marmer mewah berwarna krem. Terdapat perubahan menyeluruh pada komponen *Couple*. Foto individual dari masing-masing pengantin dihilangkan secara menyeluruh. Perubahan hanya menyisakan pesan dari pasangan pengantin untuk *user*, dan nama lengkap dari pasangan pengantin beserta nama orang tua pengantin dengan tata letak yang simetris.

```
export function CoupleSection({
  groomFirstName,
  groomLastName,
  brideFirstName,
  brideLastName,
  groomFather,
  groomMother,
  brideFather,
  brideMother,
}) {
  return (
    <div className="min-h-screen flex items-center justify-center
    px-4 sm:px-8 md:px-12 lg:px-20 xl:px-32 bg-[url('/images/
```

```
    Template4/CoupleSection/weddingbackground.png')] bg-cover bg-
    center">
13      <motion.div
14        initial={{ opacity: 0, y: 50 }}
15        animate={{ opacity: 1, y: 0 }}
16        transition={{ duration: 1.5 }}
17        className="bg-[url('/images/Template4/CoupleSection/
    shutterstock.png')] bg-cover bg-center shadow-lg rounded-[90px]
     md:rounded-[80px] lg:rounded-[150px] p-6 sm:p-10 md:p-12 lg:p
    -16 xl:p-20 text-center flex flex-col justify-center w-full max
    -w-[1100px]"
18      >
```

Kode 3.3: Kode Komponen *Couple*

Kode 3.3 mendefinisikan *CoupleSection*, yaitu komponen yang menampilkan detail informasi dari kedua mempelai beserta nama orang tua pengantin. Komponen ini menerima serangkaian props yang berisi data nama lengkap kedua mempelai dan nama orang tua masing-masing. Strukturnya terdiri dari div berlapis, di mana div terluar menggunakan sebuah gambar latar, dan di dalamnya terdapat div lain yang berfungsi sebagai "*card*" utama dengan latar belakang dan efek bayangan (*shadow-lg*) tersendiri untuk menciptakan tampilan yang elegan. *Card* informasi utama ini dianimasikan menggunakan Framer Motion agar muncul dengan efek transisi dari bawah ke atas sambil perlahan menjadi terlihat. Dari segi *layout*, komponen dirancang agar sepenuhnya responsif menggunakan properti *flexbox* dan *padding* yang adaptif untuk berbagai ukuran layar.

C.4 *Static Photo*

Gambar 3.21. Komponen *Static Photo* Undangan Final

Terdapat penambahan komponen baru pada undangan final, yaitu *Static Photo* yang dapat dilihat pada Gambar 3.21. Komponen tersebut hanya terdiri dari satu gambar statis yang menampilkan foto *prewedding* dari pasangan pengantin. Penempatan komponen *Static Photo* ini strategis dalam alur navigasi undangan, memberikan jeda visual antara informasi tekstual dan menciptakan *emotional connection* yang lebih kuat antara pasangan pengantin dengan para tamu undangan.

```
export default function StaticPhoto() {
    return (
        <div className="h-full w-full">
            <img
                src="/images/Template4/PhotoCarousel/staticPhoto2.
    jpg"
                alt="Photo 1"
                className="w-auto h-auto object-cover"
            />
        </div>
    );
}
```

Kode 3.4: Kode Komponen *Static Photo*

Kode 3.4 mendefinisikan komponen StaticPhoto yang memiliki tujuan yang sangat spesifik dan sederhana. Fungsinya hanyalah untuk menampilkan sebuah gambar statis tanpa adanya logika atau state yang kompleks. Komponen ini hanya mengembalikan sebuah div yang membungkus elemen img. Alamat gambar (src) pada kode ini bersifat statis atau di-*hardcode*, yang berarti gambar tersebut tidak dinamis dan akan selalu sama. Kelas *w-auto h-auto object-cover* digunakan untuk memastikan gambar ditampilkan dengan proporsional tanpa mengalami distorsi visual.

C.5 *Location & Dresscode*

Gambar 3.22. Komponen *Location & Dresscode* Undangan Final

Teruntuk laman *Location & Dresscode* yang dapat dilihat pada 3.22, terdapat perubahan pada desain *card* dan *card* individual untuk *dresscode* yang dihilangkan. Terdapat penambahan ikon gereja yang merepresentasikan lokasi dari akad dan ikon kue pernikahan yang merepresentasikan resepsi pernikahan. Aturan *dresscode* dipindahkan kedalam *card reception*. Fungsionalitas tombol tidaklah berubah, tombol "*View Location*" yang jika ditekan oleh *user*, tetap akan membuka laman Google Maps dari lokasi tersebut. Jika *user* melakukan *scrolldown*, *user* akan beralih ke komponen *Reservation*.

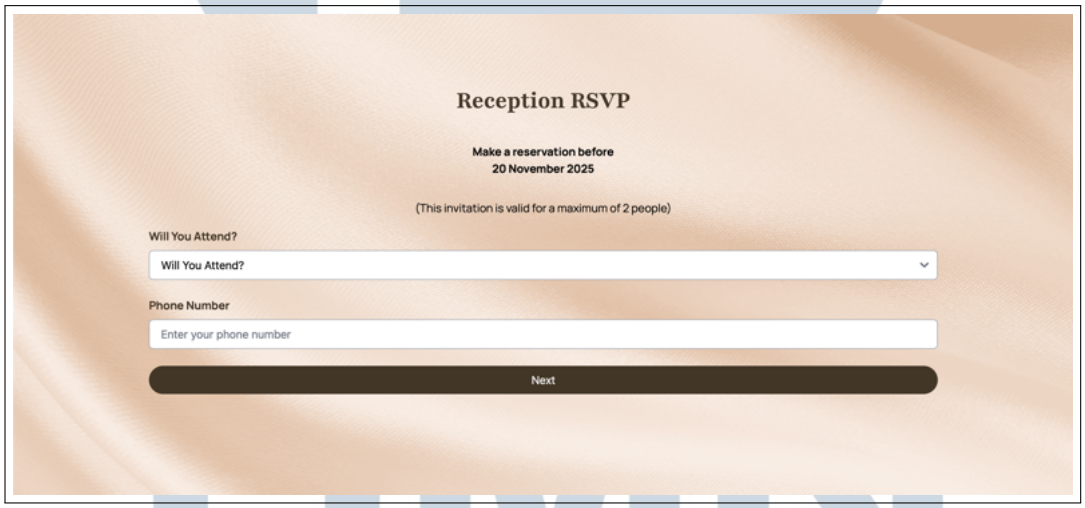
```
export default function LocationSection() {
  return (
    <div
      className="bg-cover bg-center py-16 px-6"
```

```
    style={{
      backgroundImage:
        "url('/images/Template4/CoupleSection/weddingbackground.
png')",
    }}
  >
    <div className="flex flex-col md:flex-row justify-center
items-center gap-6">
      <motion.div
        className="w-[360px] h-[600px] text-center p-6 bg-[url
('/images/Template4/CoupleSection/shutterstock.png')] bg-
opacity-90 shadow-lg flex flex-col justify-center"
        style={{ borderRadius: "100px" }}
        initial={{ opacity: 0, y: 50 }}
        whileInView={{ opacity: 1, y: 0 }}
        transition={{
          duration: 0.8,
          delay: 0.3,
          ease: "easeOut"
        }}
        viewport={{ once: true }}
      >
        <img
          src="/images/Template4/Location/church.png"
          alt="Holy Matrimony Icon"
          className="w-20 h-24 mx-auto mb-4"
        />
        <h3 className={`text-2xl mb-3 font-bold text-custom-
primaryBrown ${playfair700.className}`}>
          <AnimatedText text="Holy Matrimony" delay={0.5} />
        </h3>
        <p className={`mt-1 text-xl text-custom-primaryBrown ${
playfair400.className}`}>SATURDAY</p>
        <p className={`text-xl text-custom-primaryBrown ${
playfair400.className}`}>22 NOVEMBER 2025</p>
        <p className={`text-xl text-custom-primaryBrown ${
playfair400.className}`}>08:30 WIB</p>
        <div className="flex justify-center items-center mb-3">
        <img src="/images/Template4/Location/divider.svg" alt="
Decorative Divider" className="h-5 mt-4" />
        </div>
```

Kode 3.5: Kode Komponen *Location & Dresscode*

Kode 3.5 mendefinisikan komponen *Location & Dresscode* yang menampilkan detail jadwal dan lokasi acara, seperti akad dan resepsi. Strukturnya menggunakan *flexbox* yang membuatnya dapat beradaptasi dari tumpukan vertikal di layar kecil menjadi barisan horizontal di layar yang lebih besar. Setiap kartu informasi di dalamnya dianimasikan secara individual menggunakan motion.div, di mana animasi hanya berjalan sekali saat kartu masuk ke dalam pandangan dengan sedikit jeda waktu. Seperti komponen lainnya, styling dilakukan dengan latar belakang berlapis dan kartu yang diberi gaya spesifik. Di dalamnya, terdapat elemen-elemen seperti ikon, teks acara, dan penggunaan komponen kustom *AnimatedText* yang menandakan bahwa teks judul memiliki animasi tersendiri.

C.6 *Reservation*



Gambar 3.23. Komponen *Reservation* Undangan Final

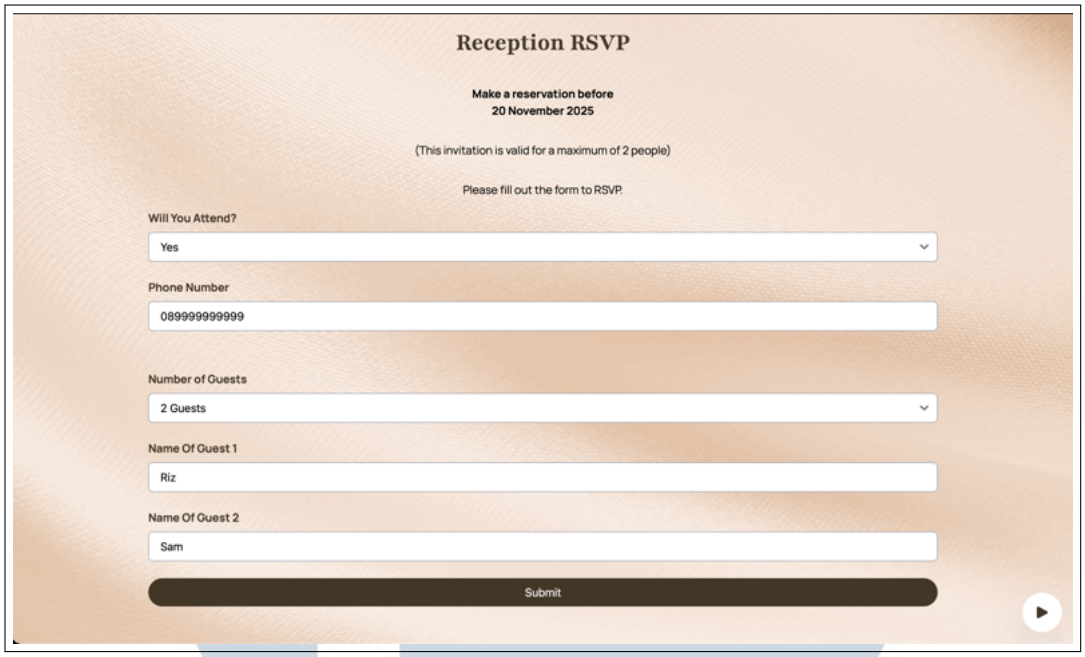
Pada komponen *Reservation* undangan final, terdapat sedikit perubahan pada form input. Perubahan terlihat pada Gambar 3.23. Yang jika sebelumnya form meminta nama dan status kehadiran dari tamu undangan, maka untuk form undangan final, input yang diminta pertama kali adalah status kehadiran dan nomor telepon. Nomor telepon digunakan sebagai bentuk registrasi bertujuan untuk memudahkan *user* untuk melakukan *scan attendance*. Dimana setiap undangan, hanya memerlukan satu QR. Jika *user* membawa *companion*, maka QR yang digunakan tetaplah satu. Setelah *user* mengisi kehadiran dengan "*Yes*" dan nomor telepon. Maka *user* akan berpindah ke laman selanjutnya yang ditunjukkan pada Gambar 3.24.

```
{!successContentVisible && !(apiStatusMessage && apiStatusMessage.toLowerCase().includes("maximum")) && !showSadMessage && (
        <motion.div className="w-full px-6 sm:px-16 md:px-32 lg:px-48" initial={{ opacity: 0, y: 30 }} whileInView={{ opacity: 1, y: 0 }} transition={{ duration: 0.8, delay: 0.5 }} viewport={{ once: true }}>
          <div className="mb-6">
            <motion.label className={`block text-left text-custom-primaryBrown mb-2 ${manrope700.className}`} initial={{ opacity: 0, x: -20 }} whileInView={{ opacity: 1, x: 0 }} transition={{ duration: 0.5, delay: 0.7 }} viewport={{ once: true }}>
              Will You Attend?
            </motion.label>
            <motion.select name="willAttend" value={formData.willAttend} onChange={handleInputChange} className="w-full px-4 py-2 border rounded-md border-gray-400 focus:outline-none focus:ring-2 focus:ring-gray-500 bg-white" initial={{ opacity: 0, x: -20 }} whileInView={{ opacity: 1, x: 0 }} transition={{ duration: 0.5, delay: 1 }} viewport={{ once: true }}>
              <option value="">Will You Attend?</option>
              <option value="yes">Yes</option>
              <option value="no">No</option>
            </motion.select>
          </div>

          <div className="mb-6">
            <motion.label className={`block text-left text-custom-primaryBrown mb-2 ${manrope700.className}`} initial={{ opacity: 0, x: -20 }} whileInView={{ opacity: 1, x: 0 }} transition={{ duration: 0.5, delay: 0.9 }} viewport={{ once: true }}>
              Phone Number
            </motion.label>
            <motion.input type="text" name="phoneNumber" value={formData.phoneNumber} onChange={handleInputChange} placeholder="Enter your phone number" className="w-full px-4 py-2 border rounded-md border-gray-400 focus:outline-none focus:ring-2 focus:ring-gray-500" initial={{ opacity: 0, x: -20 }} whileInView={{ opacity: 1, x: 0 }} transition={{ duration: 0.5, delay: 1 }} viewport={{ once: true }} />
            {phoneNumberError && (
```

```
                <p className="mt-2 text-red-600 text-sm">{
phoneNumberError}</p>
              )}
            </div>

            {!showGuestForm && (
              <motion.button
                onClick={handleSubmit}
                disabled={!formData.willAttend || isSubmitting}
                className="w-full rounded-full bg-custom-
primaryBrown text-white py-2 hover:bg-amber-900 transition
duration-300 disabled:opacity-50 disabled:cursor-not-allowed"
                initial={{ opacity: 0, y: 20 }}
                whileInView={{ opacity: 1, y: 0 }}
                transition={{ duration: 0.5, delay: 1.1 }}
                viewport={{ once: true }}
                whileHover={{ scale: 1.02 }}
                whileTap={{ scale: 0.98 }}
              >
                {isSubmitting ? "Next" : (formData.willAttend ===
"no" ? "Submit" : "Next")}
              </motion.button>
            )}
          </motion.div>
        )}
```

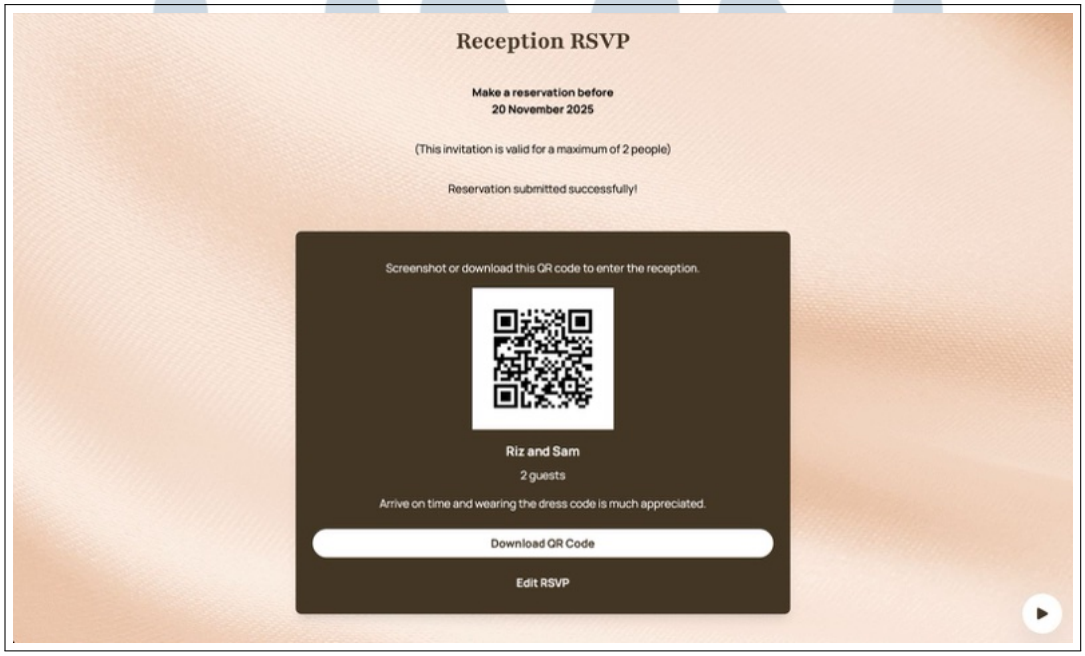
Kode 3.6: Kode Komponen *Reservation*

Kode 3.6 merupakan bagian dari logika formulir reservasi, khususnya untuk tampilan awal sebelum *user* mengirimkan respons. Tampilan formulir ini dikontrol oleh *conditional rendering*, yang berarti ia hanya akan muncul jika kondisi tertentu terpenuhi, seperti belum adanya status sukses atau pesan error. Formulir ini terdiri dari sebuah *dropdown* (select) untuk memilih kehadiran dan sebuah input untuk nomor telepon, yang nilainya terhubung ke state *formData*. Setiap elemen formulir, mulai dari label hingga tombol, diberi animasi menggunakan motion dengan jeda waktu yang bertingkat, menciptakan efek visual di mana elemen-elemen tersebut muncul satu per satu. Logika tombol juga dibuat dinamis, di mana tombol tidak dapat diklik sebelum *user* memilih status kehadiran, dan teksnya dapat berubah sesuai pilihan *user*.



Gambar 3.24. Laman *Reservation* jika *user* menghadiri acara

Gambar 3.24 menunjukkan komponen *Reservation* jika *user* memilih untuk hadir. *User* diwajibkan untuk mengisi lengkap form input tersebut, form tambahan terdiri atas jumlah hadirin yang datang beserta namanya. Jika *user* sudah selesai mengisi form tersebut, maka *user* bisa berpindah ke laman selanjutnya yang ditunjukkan pada Gambar 3.25



Gambar 3.25. QR pada laman *Reservation*

Gambar 3.25 menunjukkan hasil akhir dari komponen *Reservation* jika *user* memilih untuk menghadiri acara. Terdapat QR, nama tamu undangan, dan jumlah hadirin yang ditampilkan, QR digunakan saat tamu undangan menghadiri acara secara langsung. Cara penggunaan QR adalah *user* menunjukkan QR kepada petugas yang ada, petugas akan melakukan *scan* QR tersebut. Saat QR berhasil dilakukan *scan*, maka tamu undangan akan tervalidasi kehadirannya. Selain itu terdapat tombol untuk mengunduh gambar QR, dan tombol untuk mengedit reservasi. Yang jika ditekan, akan mengembalikan *user* ke laman awal *Reservation* seperti pada Gambar 3.23.

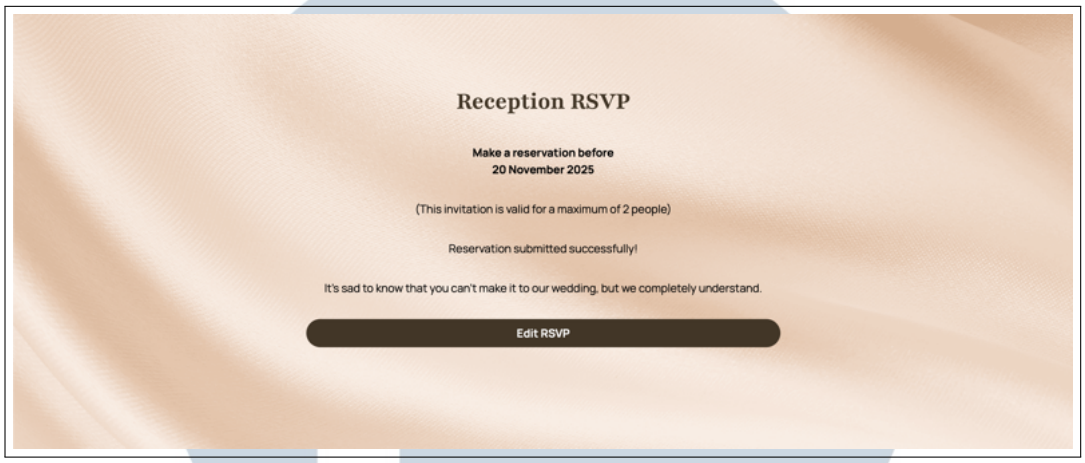
```
{showSuccessMessage && (
            <div className="fixed inset-0 bg-black bg-opacity-50 flex items-center justify-center z-50">
              <div className="bg-custom-primaryBrown text-white p-6 rounded-lg shadow-lg w-[90%] sm:w-[400px] lg:w-[700px] relative">
                {/* Header Section */}
                <div className="flex items-center justify-between mb-4">
                  <h1 className={`text-2xl text-white ${forum.className}`}>
                    Reception RSVP Form
                  </h1>
                  <button
                    onClick={() => setShowSuccessMessage(false)}
                    className="text-white text-lg font-bold hover:text-gray-300"
                  >
                    &times;
                  </button>
                </div>

                {/* Content */}
                <p className={`text-sm sm:text-base mb-4 mt-4 text-center ${manrope400.className}`}>
                  Screenshot or download this image to enter the reception.
                </p>
                {qrCodeDataUrl ? (
                  <img
                    src={qrCodeDataUrl}
                    alt="Reception Entry QR Code"
```

```
25                     className="mx-auto mb-4 w-[200px] h-[200px]
   object-contain"
26                   />
27                 ) : (
28                   <p className={`text-sm sm:text-base mb-4 text-
   center ${manrope400.className}`}>Generating QR Code...</p>
29                 )}
30                 <p className={`text-lg font-bold mb-2 text-center
   ${manrope700.className}`}>
31                   {formData.guestNames.slice(0, parseInt(formData.
   numberOfGuests)).join(" and ")}
32                 </p>
33                 <p className={`text-sm sm:text-base mb-4 text-
   center ${manrope400.className}`}>
34                   {formData.numberOfGuests} guest{formData.
   numberOfGuests > 1 ? "s" : ""}
35                 </p>
36                 <p className={`text-sm sm:text-base mb-4 text-
   center ${manrope400.className}`}>
37                   Arrive on time and wearing the dress code is
   much appreciated.
38                 </p>
39                 <button
40                   onClick={() => downloadQRCode()}
41                   className={`w-full rounded-full bg-white text-
   custom-primaryBrown py-2 hover:bg-gray-200 transition duration
   -300 mt-2 ${manrope700.className}`}
42                 >
43                   Download QR Code
44                 </button>
45               </div>
46             </div>
47           )}
```

Kode 3.7: Kode Komponen *Reservation*

Kode 3.7 menampilkan sebuah *modal* atau *pop-up* yang berisi QR *code* setelah *user* berhasil melakukan reservasi. Tampilan modal ini juga dikontrol oleh *conditional rendering* dan hanya akan muncul jika state *showSuccessMessage* bernilai *true*. Strukturnya menggunakan kelas *fixed inset-0* untuk menciptakan *overlay* yang menutupi seluruh layar. Di dalamnya, gambar QR *code* ditampilkan secara dinamis dari *qrCodeDataUrl*. Jika data QR belum siap, sebuah pesan penanda akan ditampilkan. Modal ini juga menyajikan informasi relevan seperti

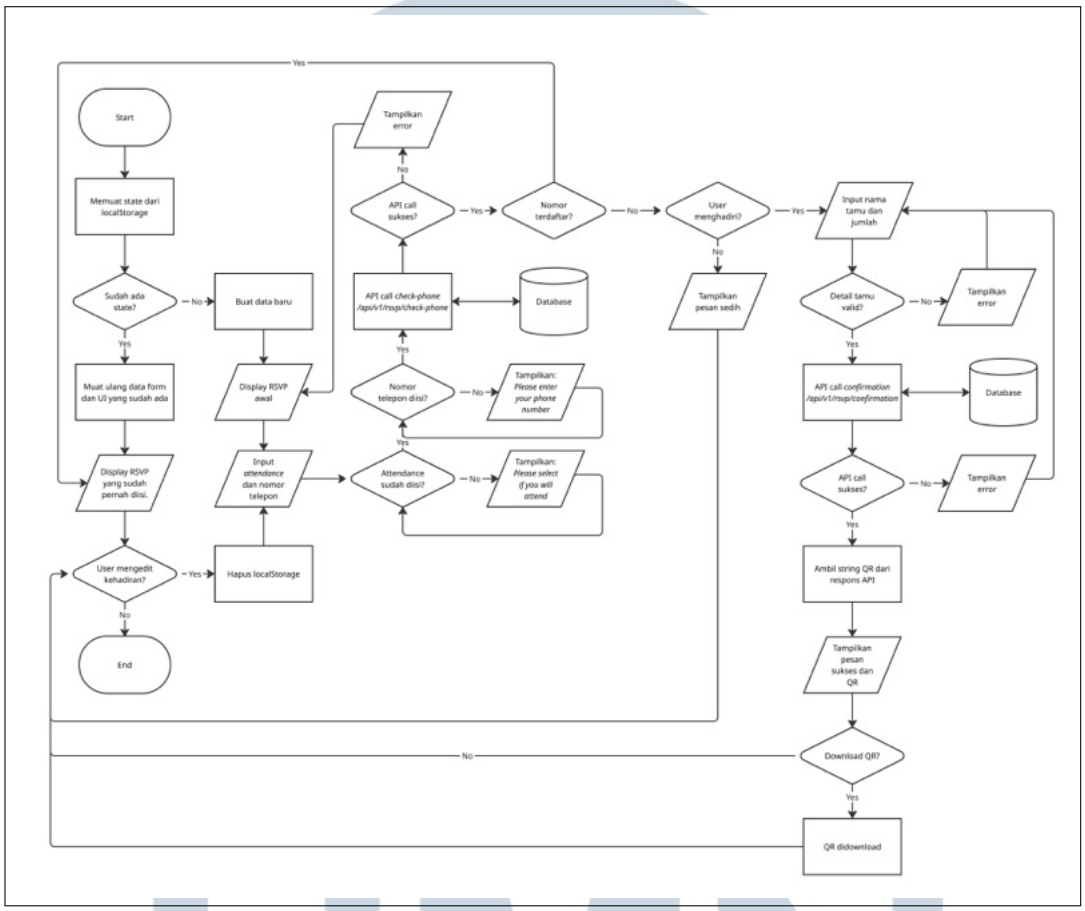
nama tamu dan jumlah hadirin yang diambil dari state *formData*. Terakhir, user diberikan *aksi* berupa tombol untuk mengunduh gambar QR *code* tersebut ke perangkatnya.



Gambar 3.26. Laman *Reservation* jika *user* tidak menghadiri acara

Gambar 3.26 menunjukkan komponen *Reservation* jika tamu undangan tidak menghadiri acara. Laman berisi pesan yang menyayangkan ketidakhadiran tamu undangan, dilengkapi dengan tombol untuk mengedit reservasi.

C.6.1 *Flowchart Reservation*



Gambar 3.27. Flowchart *Reservation*

Gambar 3.27 menunjukkan alur untuk reservasi. Reservasi dimulai dengan memuat data dari *localStorage* untuk memeriksa apakah sudah ada sesi reservasi sebelumnya. Jika data state ditemukan (sudah pernah mengisi), maka sistem akan menampilkan form yang sudah diisi beserta tampilan QR atau pesan sedih. Jika tidak terdapat data *state* yang ditemukan (*user* baru atau sesi baru), maka sistem akan menampilkan form RSVP awal. Setelah itu, *user* diminta untuk menginput status kehadiran (”*Yes*” atau ”*No*”) dan nomor telepon. Sistem akan memvalidasi apakah kedua input tersebut sudah diisi. Jika *user* memilih untuk ”*No*”, sebuah *API call* yang dijalankan menggunakan *check-phone* akan dijalankan untuk mencatat status ketidakhadirannya. API *check-phone* terdapat pada tabel 3.2. Setelah itu, akan ditampilkan pesan konfirmasi bahwa tamu tidak dapat hadir (”pesan sedih”), dan *user* akan dikembalikan ke form RSVP awal. Jika *user* memilih ”*Yes*”, sistem akan melakukan *API call* lagi untuk memeriksa apakah nomor telepon tersebut

terdaftar. Jika *user* sudah terdaftar (menandakan *user* sudah pernah mengisi reservasi), maka sistem akan menampilkan form yang sudah diisi beserta tampilan QR. Jika *user* belum terdaftar, maka *user* akan diarahkan untuk mengisi form detail tamu, yang mencakup nama dan jumlah tamu. Jika form detail tamu terkonfirmasi *valid*, maka akan ada *API call* menggunakan API *confirmation* yang terdapat pada tabel 3.3. Setelah *API call* dilakukan maka sistem akan mengambil string QR dari respons API tersebut dan menampilkannya untuk *user*. *User* diberikan pilihan untuk mendownload QR dan pilihan untuk mengubah kehadiran tamu. Jika tamu memilih untuk mengubah kehadiran, maka sistem akan menghapus data yang sudah ada dan mengarahkan user untuk mengisi ulang form.

C.6.2 Struktur Tabel API *Reservation*

API yang digunakan pada kerja magang ini berasal dari Minyma API. API diimplementasikan pada komponen *Reservation*. Berikut adalah struktur tabel-tabel *payload* JSON API yang digunakan. Pada *flowchart* yang ditunjukkan pada Gambar 3.27, digunakan API *check-phone*. Payload secara rinci ditunjukkan pada Tabel 3.2. API digunakan untuk memeriksa apakah nomor telepon yang digunakan *user* sudah terdaftar. API akan menggunakan *eventId* dan *phoneNumber* sebagai payload, setelah API dipanggil, maka API akan mengirim respon berupa data yang menunjukkan apakah *phoneNumber* sudah teregistrasi dan apakah *user* sudah melakukan reservasi. Tabel API *check-phone* secara rinci dapat dilihat pada Tabel 3.2.

Tabel 3.2. Tabel Payload API *Check-Phone*

Metode:	GET
Endpoint:	/api/v1/rsvp/check-phone
Deskripsi:	API berfungsi untuk memverifikasi apakah phone number terdaftar (*registered*).
Header:	-
Body:	-

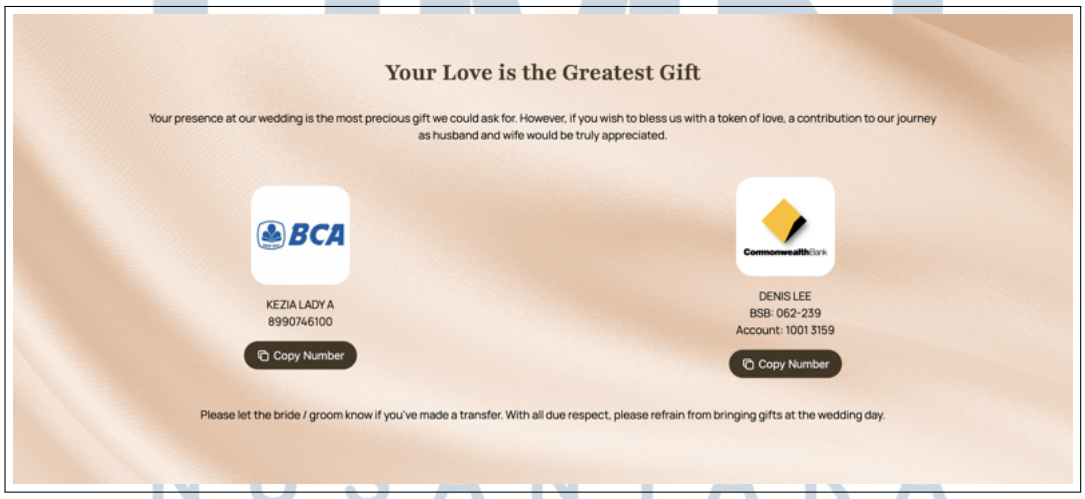
Tabel 3.3 menunjukkan *payload* dari API *confirmation* yang digunakan untuk mengirim data reservasi *user* pada *database*. API akan mengirim data yang telah diinput oleh *user*, seperti nomor telepon, jumlah hadirin, dan nama hadirin. Jika *user* belum pernah melakukan registrasi sebelumnya, maka data *user* akan

dikirim dan disimpan pada *database*. Setelah itu, *user* akan mendapat respon *string QR* yang dikonversi menjadi *image*. Jika *user* sudah pernah melakukan registrasi sebelumnya, maka laman akan menampilkan kode QR yang sudah dibuat sebelumnya.

Tabel 3.3. Tabel Payload API *Confirmation*

Method:	GET	
Endpoint:	/api/v1/rsvp/confirmation	
Description:	Berfungsi untuk menyimpan data reservasi ke dalam *database*. Serta mengembalikan respons QR *string*.	
Header	Applications/json	Mengirim data dalam format JSON.
Body	eventId	ID unik dari acara undangan.
	Phone	Nomor telepon *user*
	isAttending	Status kehadiran dari *user*
	attendingCount	Jumlah tamu yang akan hadir
	name	Nama dari tamu yang akan hadir
	isRegistered	Registrasi nomor telepon *user*
	isRsvp	Konfirmasi kehadiran *user*

C.7 *Gift*



Gambar 3.28. Komponen *Gift* Undangan Final

Gambar 3.28 menunjukkan sedikit perubahan pada komponen *Gift*, perubahan hanya terdapat pada jumlah rekening yang ditampilkan menjadi 2

rekening, yang menampilkan detail rekening dari masing-masing mempelai dan dilengkapi tombol fungsional "*Copy Number*" untuk meminimalisir kesalahan dan memudahkan proses transfer.

```
return (
    <div className="bg-[url('/images/Template4/GiftSection/
    GiftBackground.png')] h-auto pb-16 sm:h-full sm:pb-0 bg-cover
    bg-center">
      <motion.div
        className="text-center items-center flex flex-col justify-
    center pb:32 sm:pb-16"
        initial={{ opacity: 0 }}
        whileInView={{ opacity: 1 }}
        transition={{ duration: 1 }}
        viewport={{ once: true }}
      >
        <motion.h1
          className={`text-4xl font-serif text-custom-primaryBrown
     pt-16 pb-8 px-12 ${playfair700.className}`}
        >
          <AnimatedText text="Your Love is the Greatest Gift"
    delay={0.2} />
        </motion.h1>

        <motion.p
          className={`px-6 mb-4 sm:px-48 ${manrope400.className}`}
          initial={{ opacity: 0, y: 20 }}
          whileInView={{ opacity: 1, y: 0 }}
          transition={{ duration: 0.8, delay: 0.4 }}
          viewport={{ once: true }}
        >
          Your presence at our wedding is the most precious gift
    we could ask
          for. However, if you wish to bless us with a token of
    love, a
          contribution to our journey as husband and wife would be
     truly
          appreciated.
        </motion.p>
```

Kode 3.8: Kode Komponen *Gift*

Kode 3.8 mendefinisikan komponen *Gift* yang menampilkan bagian rekening pasangan, di mana sebuah div utama berfungsi sebagai wadah dengan

gambar latar. Di dalamnya, seluruh konten dibungkus oleh *motion.div* dari Framer Motion yang memberikan efek *fade-in* saat komponen masuk ke area pandang *user* ketika di-*scroll*. Judul pada komponen ini menggunakan komponen kustom *AnimatedText* untuk menciptakan animasi teks yang lebih mendetail. Sementara itu, paragraf deskripsinya dianimasikan agar muncul dengan efek *fade-in* dan sedikit bergerak ke atas. Penggunaan jeda waktu (*delay*) pada animasi paragraf menciptakan efek visual yang dinamis dan bertingkat.

C.8 *Photo Album*

Gambar 3.29. Komponen *Photo Album*

Terdapat penambahan komponen *Photo Album* yang ditunjukkan pada Gambar 3.29, *Photo album* merupakan komponen baru yang hanya terdapat pada undangan final. *Photo album* berfungsi untuk memberikan sentuhan yang lebih personal dan visual bagi para tamu. *Photo album* terdiri dari 2 baris dan 3 kolom gambar yang dengan total menampilkan 6 gambar. Setiap gambar di dalamnya bersifat statis, dirancang bukan sebagai galeri interaktif.

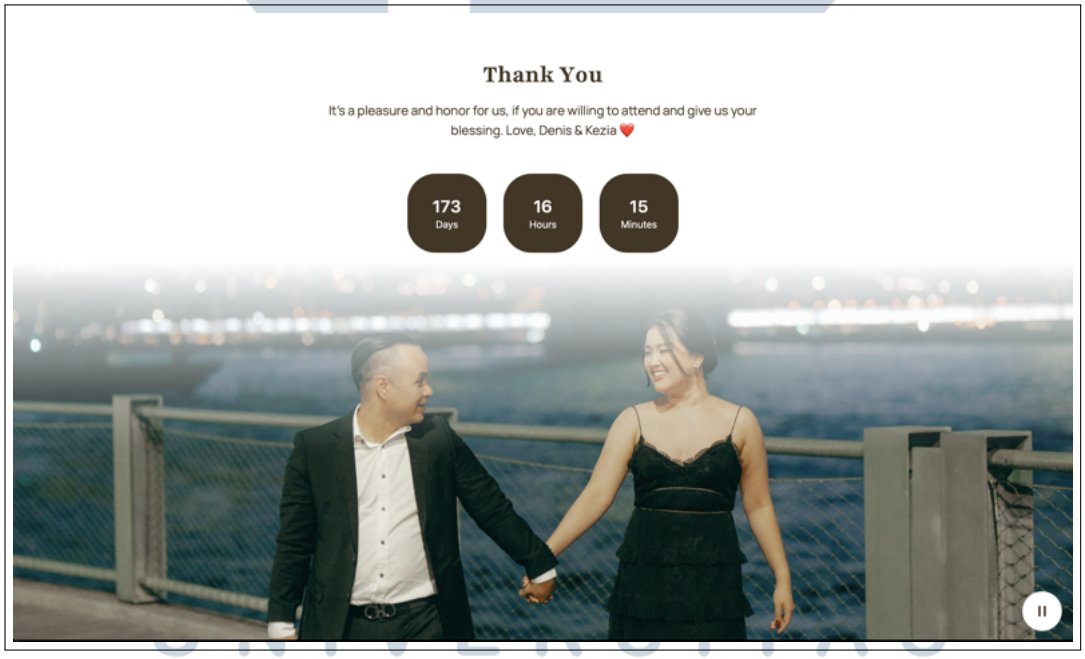
```
export default function photoAlbum() {
  return (
      <div className="h-full w-full">
          <img
              src="/images/Template4/photoAlbum/photo.jpg"
```

```
                alt="Photo 1"
                className="w-auto h-auto object-cover"
            />
        </div>
    );
}
```

Kode 3.9: Kode Komponen *Photo Album*

Kode 3.9 menunjukkan kode untuk komponen *Photo Album*. Kode ini sangatlah ringkas dan fungsional. Komponen ini didefinisikan untuk menampilkan enam buah gambar. Kelas *w-auto h-auto object-cover* diterapkan untuk memastikan gambar dapat menyesuaikan diri dengan wadahnya secara proporsional tanpa merusak rasio aspek aslinya.

C.9 *Thank You*



Gambar 3.30. Komponen *Thank You*

Gambar 3.30 menunjukkan komponen *Thank You*. Tidak terdapat perubahan yang berarti teruntuk komponen *Thank You*, perubahan hanya terdapat pada pada warna gradien yang diubah. Yang semula berwarna coklat, kemudian diganti menjadi warna putih. Terdapat perubahan lain pada foto latar yang digunakan. Hal tersebut sesuai mengikuti kemauan klien dan arahan dari tim UI/UX.

```
export default function ThankYou({ weddingDate }) {
    const containerRef = useRef(null);
    const isContainerInView = useInView(containerRef, { once: true
    , amount: 0.2 });

    const titleRef = useRef(null);
    const isTitleInView = useInView(titleRef, { once: true, amount
    : 0.5 });

    const textRef = useRef(null);
    const isTextInView = useInView(textRef, { once: true, amount:
    0.5 });

    const timerRef = useRef(null);
    const isTimerInView = useInView(timerRef, { once: true, amount
    : 0.5 });

    return (
        <div className="relative h-[70vh] md:h-screen" ref={
    containerRef}>
            <motion.div
              className="absolute inset-0 bg-white"
              initial={{ opacity: 0 }}
              animate={isContainerInView ? { opacity: 1 } : {
    opacity: 0 }}
              transition={{ duration: 1 }}
            />
            <motion.div
                className="absolute inset-0 bg-cover bg-center"
                style={{
                    backgroundImage: "url('/images/Template4/
    ThankYou/Background2.jpeg')",
                    maskImage: "linear-gradient(to bottom,
    transparent 40%, black 70%)",
                    WebkitMaskImage: "linear-gradient(to bottom,
    transparent 40%, black 70%)"
                }}
                initial={{ opacity: 0 }}
                animate={isContainerInView ? { opacity: 1 } : {
    opacity: 0 }}
                transition={{ duration: 1, delay: 0.5 }}
            />
```

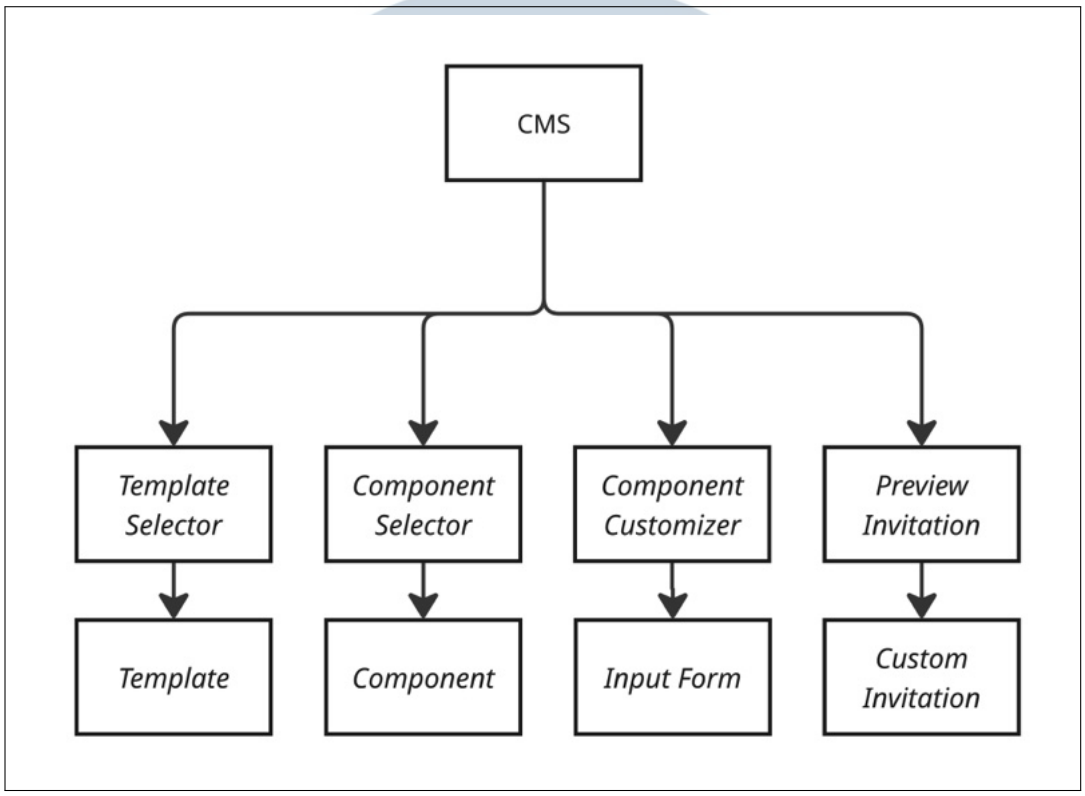
Kode 3.10: Kode Komponen *Thank You*

Kode 3.10 mendefinisikan komponen *ThankYou* yang menjadi bagian penutup dari keseluruhan undangan. Komponen ini menggunakan pendekatan animasi dengan hook *useRef* dan *useInView* dari Framer Motion. Hal ini memungkinkan animasi dipicu secara presisi hanya ketika elemen spesifik masuk ke dalam *viewport user*. Terdapat penggunaan *maskImage* dengan gradien linear untuk menciptakan efek di mana gambar latar belakang seolah-olah memudar dan menyatu dengan mulus ke latar belakang di atasnya. Animasi pada komponen ini juga dibuat berlapis, di mana latar belakang putih muncul lebih dulu, diikuti oleh gambar latar, sehingga menciptakan efek kedalaman yang dinamis.

3.4.3 Proyek Laman CMS Undangan

Laman *content management system* atau yang biasa disebut laman CMS merupakan sebuah platform yang memungkinkan *user* membuat, mengelola, dan memodifikasi konten website. Pada kerja magang ini, laman CMS ditujukan untuk *user* jika ingin membuat undangan dari *template* yang ada. CMS juga ditujukan untuk *user* yang ingin mengkustomisasi *template* berdasarkan keinginan *user* seperti mengubah gambar, ataupun text dari *template* tersebut. Selain itu, laman CMS digunakan oleh penulis untuk mengubah undangan yang semula semi *hardcoded* menjadi dinamis dan layak untuk dijadikan *template*. Laman CMS dibagi menjadi 4 halaman utama. Yaitu laman *template selector*, laman *component selector*, laman *component customizer*, dan laman *preview invitation*.

A Sitemap CMS

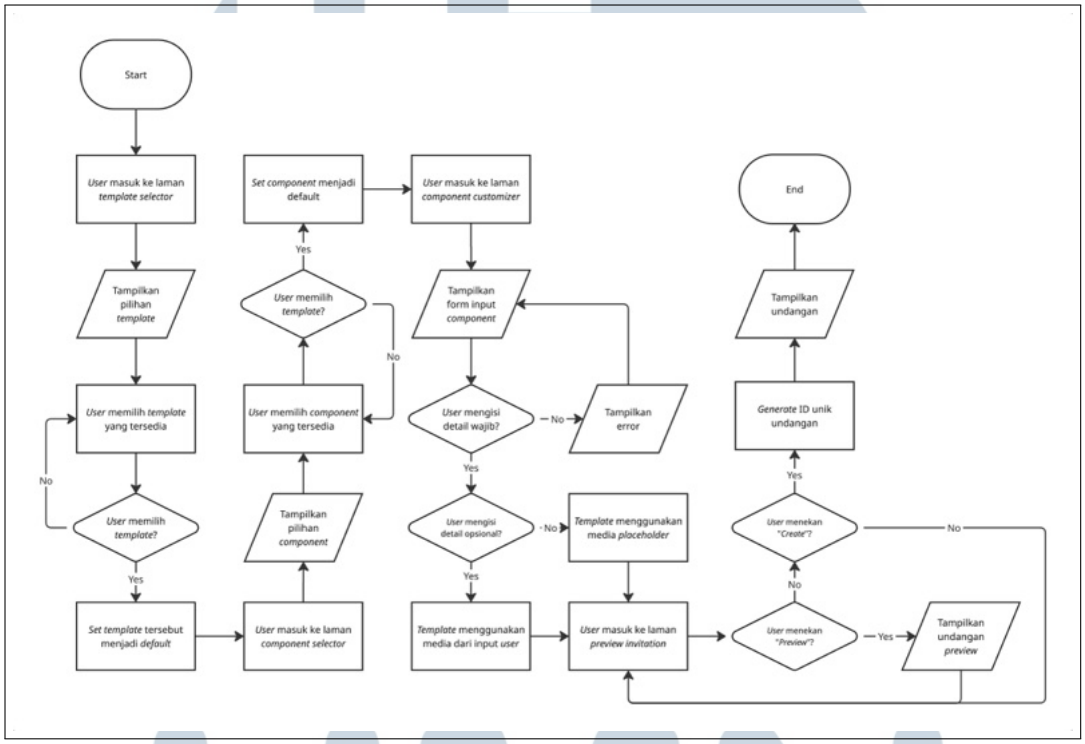


Gambar 3.31. *Sitemap* dari CMS

Gambar 3.31 menunjukkan *sitemap* dari laman *Content Management System* (CMS) yang dirancang untuk membuat undangan digital dan mengkustomisasi. Laman CMS dibagi menjadi empat fungsi inti. Alur pertama adalah *template selector*, di mana *user* memulai proses dengan memilih *template* yang disediakan. Hasil dari tahap ini adalah sebuah *template* yang akan menjadi fondasi undangan. Selanjutnya, *user* beralih ke laman *component selector*. Di sini, *user* dapat memilih bagian-bagian spesifik (seperti *Cover*, *Hero*, *Gift*, dll.) yang ingin disertakan, yang kemudian menghasilkan sekumpulan *component* terpilih untuk undangan. Selanjutnya, laman *component customizer* menyediakan serangkaian *input form* yang memungkinkan *user* untuk mengisi atau mengubah konten—seperti teks dan gambar—pada setiap komponen yang telah dipilih sebelumnya. Terakhir, laman *preview invitation* berfungsi sebagai tahap finalisasi. Pada laman ini, *user* dapat memeriksa hasil kustomisasi sebelum undangan dibuat secara permanen. Jika disetujui, hasil akhirnya adalah sebuah undangan *custom* yang dipersonalisasi secara mandiri. Secara keseluruhan, *sitemap* ini menunjukkan alur kerja yang

terstruktur dan modular, memandu *user* dari pemilihan desain awal hingga menghasilkan undangan *custom* yang telah dipersonalisasi.

B *Flowchart* CMS



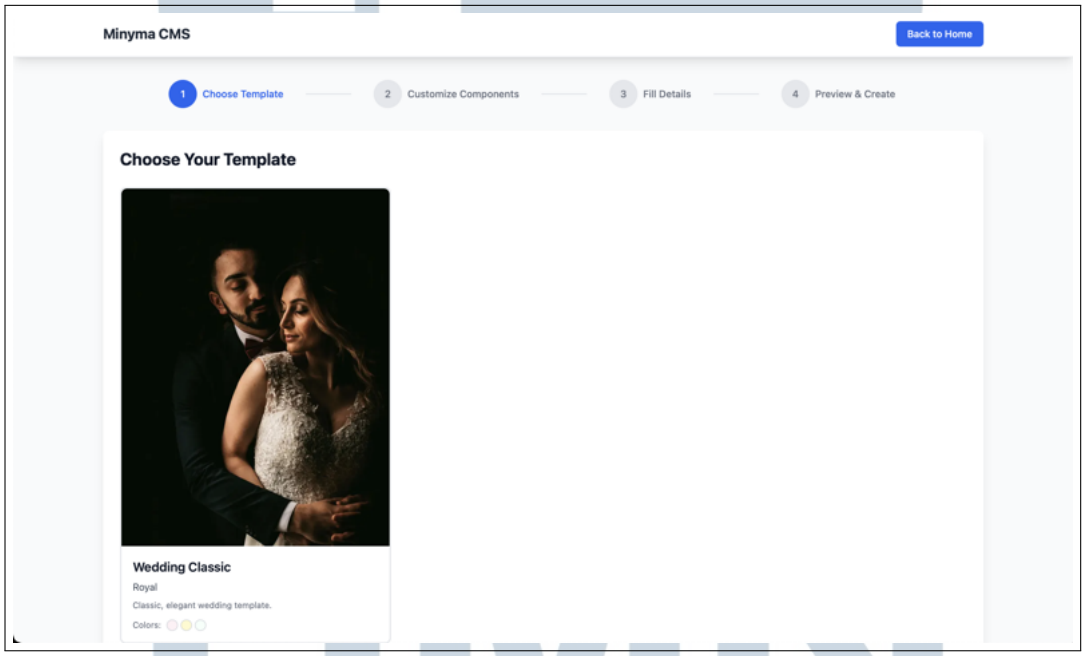
Gambar 3.32. *Flowchart* Keseluruhan Laman CMS

Alur dari laman CMS dapat dilihat pada Gambar 3.32. Proses pembuatan undangan digital dimulai ketika *user* mengakses laman *template selector*, di mana laman akan menampilkan pilihan dari template yang tersedia. Setelah itu *user* memilih salah satu dari *template*. Setelah *user* memilih *template*, *template* tersebut akan dijadikan *default* dari undangan. Lalu *user* berlanjut ke laman *component selector*. Karena setiap *template* terdiri dari beberapa komponen, maka *user* dapat memilih komponen bawaan yang tersedia dari *template*. Setelah *user* memilih komponen, komponen akan diset menjadi *default*. Lalu *User* memasuki laman *component customizer*, laman tersebut berfungsi agar *user* dapat menginput teks dan gambar sesuai dengan keinginan. *User* diwajibkan untuk mengisi beberapa detail wajib seperti nama pengantin dan tanggal acara. Hal tersebut dibutuhkan oleh *template* agar bisa berjalan. Jika *user* sudah selesai mengisi detail wajib, maka *user* akan diarahkan untuk mengisi detail opsional, jika hal tersebut tidak

dilakukan, maka komponen akan menggunakan media *placeholder* bawaan dari *template*. Setelah semua data terisi, *user* diarahkan ke halaman *preview invitation* untuk meninjau hasil. Jika *user* puas dengan *preview*, sistem akan melakukan *generate* ID unik undangan dan menampilkan undangan final yang siap digunakan, menandakan selesainya keseluruhan proses pembuatan undangan digital.

C Hasil Implementasi

C.1 Laman *Template Selector*



Gambar 3.33. Tampilan Laman *Template Selector*

Pada Gambar 3.33 merupakan laman *template selector*. Laman *template selector* merupakan laman pertama yang dilihat oleh *user* saat memasuki CMS. Laman tersebut digunakan untuk memilih salah satu *template* dari pilihan yang tersedia. Saat ini, memang hanya terdapat satu *template* yang bisa digunakan. Tetapi tidak menutup kemungkinan bahwa *template* akan diperbanyak kedepannya. Walaupun manual, penambahan *template* masih tergolong cukup simpel, *template* bisa ditambahkan dengan menambahkan *template* kedalam folder komponen. Dan *template* mesti ditulis dalam JSON list *template*, setelah itu *template* akan muncul dengan sendirinya. Jika *user* sudah memilih salah satu *template*, maka *user* dapat berpindah ke laman selanjutnya.

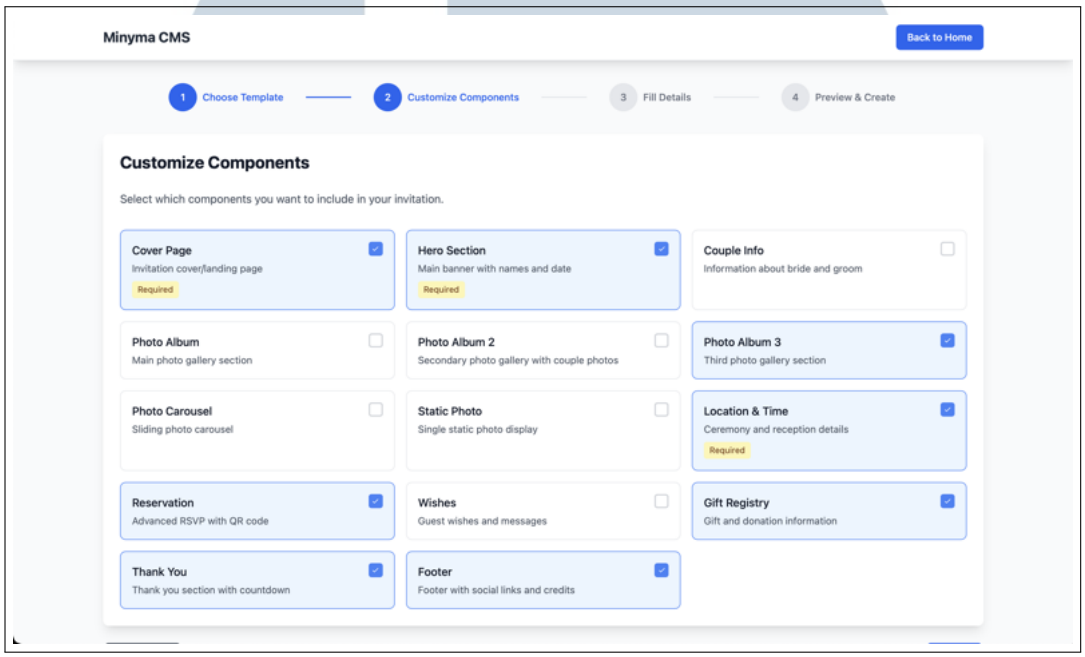
```
export default function TemplateSelector({ selectedTemplate,
    onTemplateSelect }) {
  return (
    <div>
      <h2 className="text-2xl font-bold text-gray-900 mb-6">Choose
     Your Template</h2>

      <div className="grid grid-cols-1 md:grid-cols-2 lg:grid-cols
    -3 gap-6">
        {templateData.map((template) => (
          <div
            key={template.id}
            onClick={() => onTemplateSelect(template)}
            className={`
              relative cursor-pointer rounded-lg border-2
    transition-all duration-200 hover:shadow-lg
              ${selectedTemplate?.id === template.id
                ? 'border-blue-500 ring-2 ring-blue-200'
                : 'border-gray-200 hover:border-gray-300'
              }
            `}
          >
            {/* Template Preview Image */}
            <div className="aspect-[3/4] relative overflow-hidden
    rounded-t-lg">
              <Image
                src={template.img}
                alt={template.title}
                fill
                className="object-cover"
              />
```

Kode 3.11: Kode Laman *Template Selector*

Kode 3.11 mendefinisikan komponen *TemplateSelector* yang berfungsi untuk menampilkan daftar *template* yang tersedia kepada *user*. Komponen ini menerima dua *props* utama: *selectedTemplate* untuk mengetahui *template* mana yang sedang dipilih, dan *onTemplateSelect* sebagai fungsi yang akan dieksekusi ketika *user* memilih sebuah *template*. Secara fungsional, kode ini melakukan iterasi (*mapping*) terhadap *templateData*, yang merupakan sebuah array berisi informasi setiap *template*. Untuk setiap *template*, komponen ini me-render sebuah div yang dapat diklik. Terdapat logika untuk memberikan styling kondisional, di mana *template* yang aktif akan diberikan bingkai (*border*) dan cincin (*ring*)

berwarna biru untuk membedakannya dari yang lain. Ketika sebuah *template* diklik, fungsi *onTemplateSelect* akan dipanggil, sehingga pilihan *user* dapat disimpan dan digunakan oleh komponen induk.

C.2 Laman *Component Selector*



Gambar 3.34. Laman *Component Selector*

Gambar 3.34 merupakan laman *component selector*. Pada laman tersebut, *user* disuguhkan dengan semua pilihan komponen dari *template* yang *user* pilih. Komponen dapat dipilih oleh bebas tanpa mengganggu fungsionalitas dari undangan. Saat ini, terdapat limitasi dimana *user* belum bisa mengatur urutan muncul dari komponen yang dipilih. Komponen muncul berurutan sesuai dengan urutan awal pada *template*. Jika *user* sudah selesai memilih komponen yang ingin digunakan, maka *user* akan berpindah ke laman selanjutnya.

```
1 return (
2     <div>
3       <h2 className="text-2xl font-bold text-gray-900 mb-6">
    Customize Components</h2>
4       <p className="text-gray-600 mb-8">Select which components
    you want to include in your invitation.</p>
5
6       <div className="grid grid-cols-1 md:grid-cols-2 lg:grid-cols
    -3 gap-4">
```

```
      {availableComponents.map((component) => (
        <div
          key={component.key}
          className={`
            border rounded-lg p-4 cursor-pointer transition-all duration-200
            ${selectedComponents[component.key]
              ? 'border-blue-500 bg-blue-50'
              : 'border-gray-200 hover:border-gray-300'
            }
          `}
          onClick={() => handleComponentToggle(component.key)}
        >
          <div className="flex items-start justify-between">
            <div className="flex-1">
              <h3 className="font-medium text-gray-900 mb-1">{component.name}</h3>
              <p className="text-sm text-gray-600">{component.description}</p>
            </div>
            <div className={`
              w-5 h-5 rounded border-2 flex items-center justify-center ml-3 flex-shrink-0
              ${selectedComponents[component.key]
                ? 'border-blue-500 bg-blue-500'
                : 'border-gray-300'
              }
            `}>
```

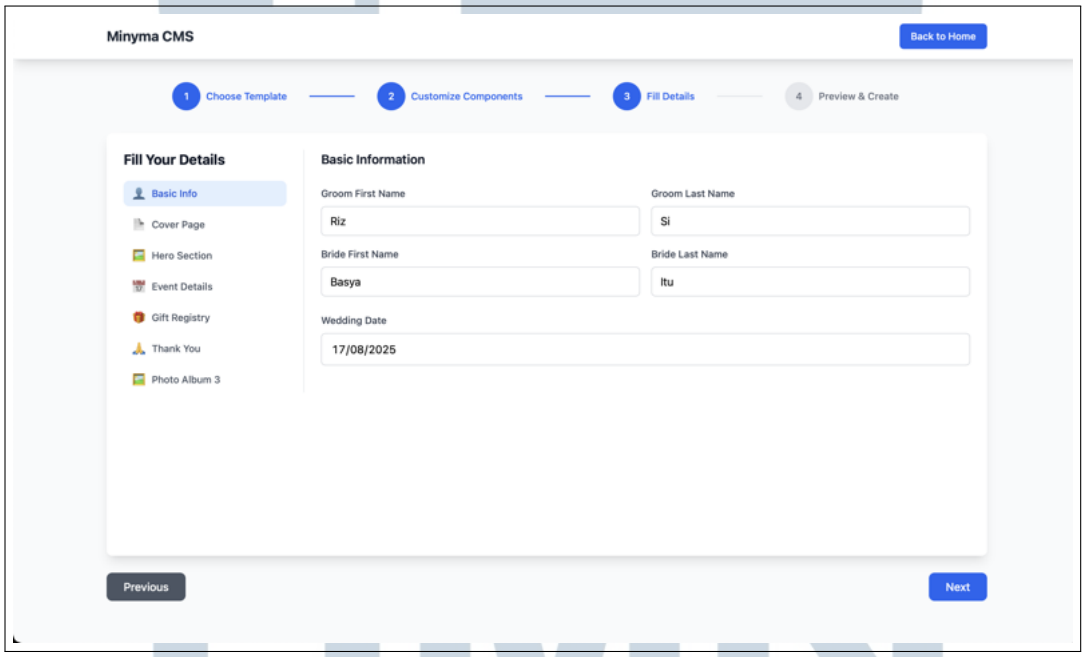
Kode 3.12: Kode Laman *Component Selector*

Kode 3.12 merupakan implementasi dari laman *component selector*. Fungsinya adalah untuk menampilkan semua komponen yang tersedia dari *template* yang telah dipilih sebelumnya dan memungkinkan *user* untuk memilih komponen mana saja yang ingin disertakan dalam undangan mereka. Komponen ini bekerja dengan melakukan iterasi pada *array availableComponents*. Setiap *item* dalam *array* tersebut dirender sebagai sebuah kartu (*card*) yang dapat dipilih, yang berisi nama dan deskripsi singkat dari komponen. Status terpilih atau tidaknya sebuah komponen dikelola melalui objek *selectedComponents*. Ketika sebuah kartu diklik, fungsi *handleComponentToggle* akan dipanggil dengan membawa kunci unik dari komponen tersebut, yang kemudian akan memperbarui status pilihan. Secara visual, kartu komponen yang terpilih akan memiliki warna latar dan bingkai yang berbeda,

serta menampilkan sebuah *checkbox* yang terisi untuk memberikan umpan balik visual yang jelas kepada *user*.

C.3 Laman *Component Customizer*

Selanjutnya, *user* akan memasuki laman *component customizer*. Setelah *user* memilih *template*, *user* dapat melakukan kustomisasi pada bagian gambar atau text tergantung dari isi konten dari masing-masing komponen yang ada. Jika *user* memilih untuk tidak melakukan kustomisasi, maka undangan akan menggunakan gambar atau teks *placeholder* yang sudah disediakan dari *template*.



Gambar 3.35. Pengisian *Basic Info*

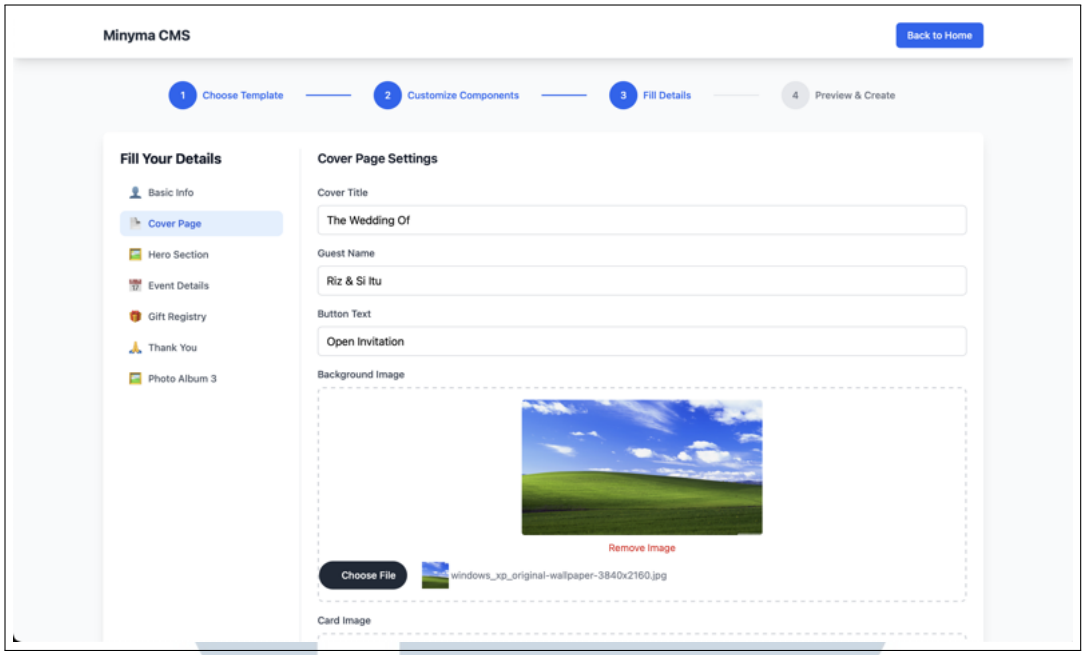
Gambar 3.35 menunjukkan pengisian *basic info*. *Basic info* merupakan informasi minimum yang diperlukan oleh undangan untuk bekerja. *Basic info* meminta input nama pengantin pria dan wanita. Dikarenakan informasi tersebut akan ditampilkan pada komponen lain seperti *Cover*, dan *Hero*. Setelah itu, *user* diharuskan untuk mengisi tanggal dari acara pernikahan. Dikarenakan tanggal pernikahan akan terkoneksi dengan komponen lain seperti *Hero*, *Location*, dan *Countdown*.

```
1 const [activeSection, setActiveSection] = useState('basic');
2
3   if (!template) return null;
4
```

```
  const updateData = (section, field, value) => {
    onDataChange({
      ...data,
      [section]: {
        ...data[section],
        [field]: value
      }
    });
  };

  const updateBasicData = (field, value) => {
    onDataChange({
      ...data,
      [field]: value
    });
  };
```

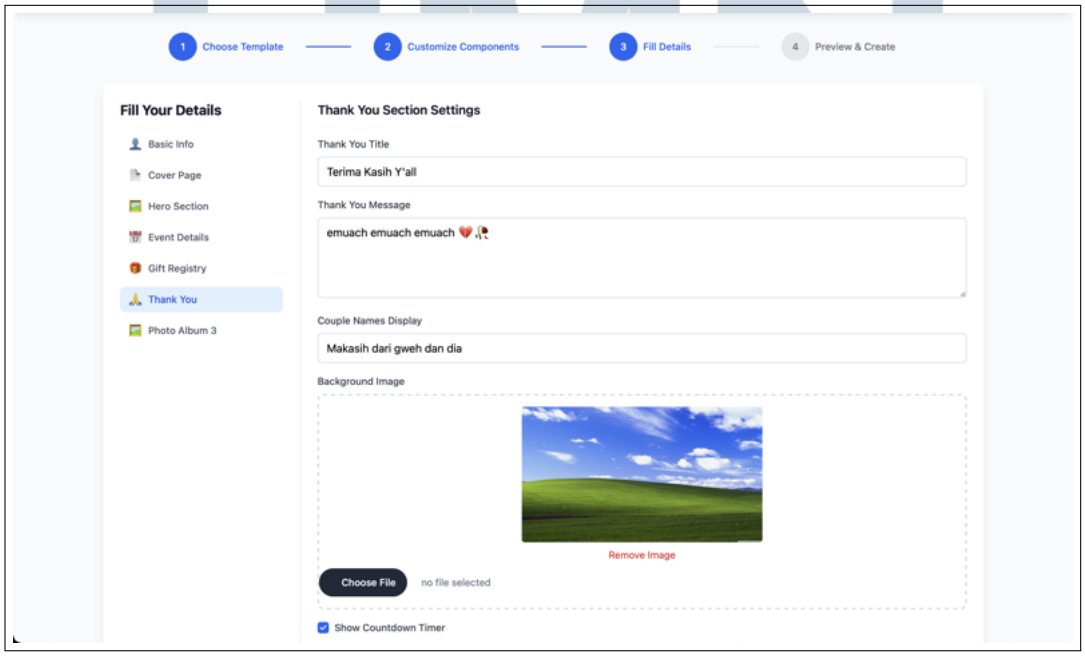
Kode 3.13: Kode *Basic Info*

Kode 3.13 menampilkan logika inti untuk pengelolaan data pada laman *component customizer*. Terdapat sebuah state bernama *activeSection* yang digunakan untuk melacak bagian kustomisasi mana yang sedang aktif. Dua fungsi utama, *updateData* dan *updateBasicData*, dirancang untuk menangani perubahan input dari *user*. Fungsi *updateData* bersifat lebih umum, mampu memperbarui data dengan menerima parameter *section*, *field*, dan *value*. Sementara itu, *updateBasicData* digunakan untuk memperbarui data seperti nama pengantin. Kedua fungsi ini memanggil *prop onDataChange* untuk mengirimkan data yang telah diperbarui kembali ke komponen induk, sebuah pola yang dikenal sebagai *lifting state up* untuk menjaga agar data tetap sinkron di seluruh aplikasi.



Gambar 3.36. Kustomisasi *Cover Page*

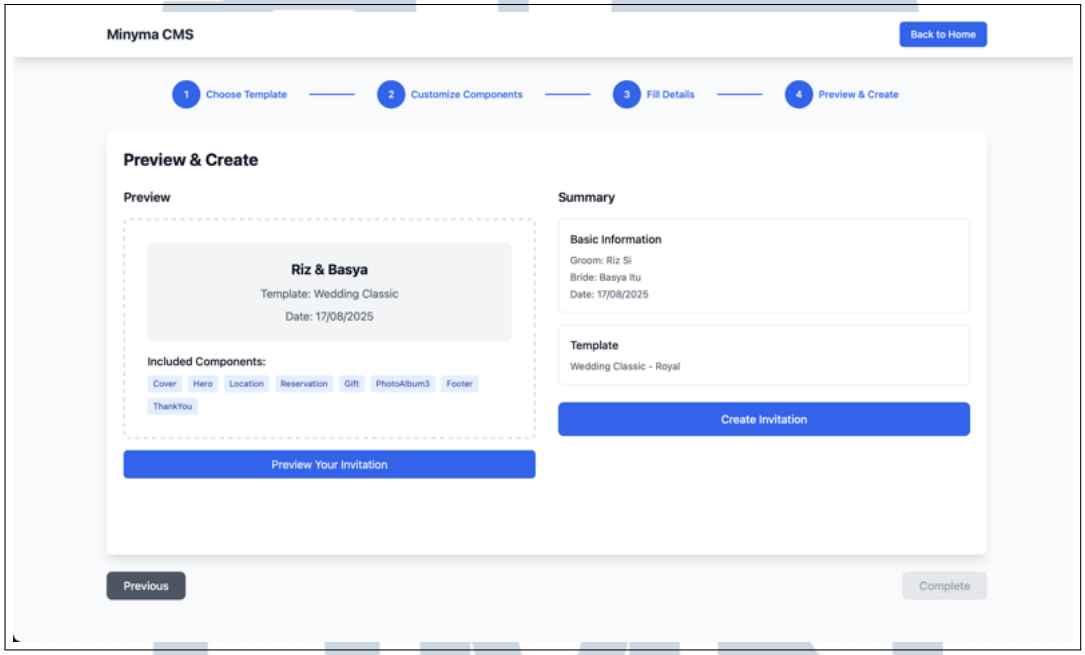
Selanjutnya, *user* dapat mencoba untuk mengisi komponen lain seperti *Cover*. Seperti yang ditunjukkan pada Gambar 3.36, konten yang dapat diubah antara lain adalah judul dari undangan tersebut. Teks dari tombol yang digunakan untuk membuka undangan. Gambar latar dari undangan, dan gambar dari *card* undangan tersebut. Demi kepentingan testing, *user* mencoba untuk mengganti titel dari undangan, serta foto latar dari undangan.



Gambar 3.37. Kustomisasi Komponen *Thank You*

Untuk komponen terakhir yaitu *thank you* yang ditunjukkan pada Gambar 3.37, *user* mencoba untuk mengkustomisasi komponen dengan mengubah titel dari komponen tersebut, lalu *user* dapat memberikan pesan untuk para tamu undangan, dan pesan penutup. *User* juga dapat mengubah latar dari komponen tersebut dengan cara meng-*upload* gambar yang diinginkan.

C.4 Laman *Preview Invitation*



Gambar 3.38. Laman *Preview Invitation*

Setelah *user* selesai melakukan kustomisasi teks dan gambar pada komponen yang dipilih, maka *user* akan masuk ke laman *preview invitation* yang ditunjukkan pada Gambar 3.38. Terdapat dua fungsi utama dalam laman tersebut. Fungsi pertama ialah *preview invitation*. Fungsi tersebut memungkinkan agar *user* dapat melihat *preview* undangan sebelum undangan dibuat secara aktual. *Preview* undangan bersifat hanya sementara, dan tidak dapat disimpan. Jika *user* merasa ingin melakukan kustomisasi ulang, maka *user* masih dapat melakukannya. Fungsi kedua ialah *create invitation*. Fungsi tersebut digunakan untuk membuat undangan secara aktual, saat undangan tersebut di-*create*, maka undangan tidak bisa diubah kembali. Setiap undangan memiliki ID unik yang diambil dari *timestamp* waktu setempat.

```
export default function PreviewInvitation({ template, components, data }) {
  const router = useRouter();
  const [isCreating, setIsCreating] = useState(false);

  if (!template) return null;

  const generateInvitation = () => {
    const invitationId = Date.now().toString();
    localStorage.setItem(`invitation_${invitationId}`, JSON.stringify(invitationData));

    const existingTemplateData = JSON.parse(localStorage.getItem('templateData') || '{}');
    existingTemplateData[invitationId] = invitationData;
    localStorage.setItem('templateData', JSON.stringify(existingTemplateData));

    // Navigate to the invitation page
    window.open(`/invitation/${invitationId}`, '_blank');
  };

  const handleCreateInvitation = async () => {
    setIsCreating(true);

    try {
      // Just call the generateInvitation function
      generateInvitation();

    } catch (error) {
      console.error('Error creating invitation:', error);
      alert('Failed to create invitation. Please try again.');
    } finally {
      setIsCreating(false);
    }
  };
```

Kode 3.14: Kode Laman *Preview Invitation*

Kode 3.14 mendefinisikan komponen *PreviewInvitation* yang berfungsi sebagai langkah terakhir dalam alur pembuatan undangan. Komponen ini bertanggung jawab untuk memfinalisasi dan menyimpan konfigurasi undangan yang telah dibuat oleh *user*. Di dalamnya, terdapat fungsi *generateInvitation* yang menjadi logika utama. Ketika dipanggil, fungsi ini akan membuat sebuah ID unik

untuk undangan menggunakan *Date.now()*. Selanjutnya, seluruh data undangan (termasuk *template*, komponen yang dipilih, dan kustomisasi teks/gambar) akan disimpan ke dalam *localStorage*. Data disimpan dua kali: sekali dengan ID uniknya sendiri dan sekali lagi ditambahkan ke dalam sebuah objek *global templateData*. Setelah penyimpanan berhasil, *window.open* akan membuka undangan yang baru dibuat di *tab* baru. Fungsi *handleCreateInvitation* bertindak sebagai pembungkus yang mengelola *state isCreating* untuk memberikan umpan balik visual kepada *user*, seperti menonaktifkan tombol saat proses pembuatan sedang berjalan, serta menangani potensi error.

C.5 Hasil Implementasi Undangan *Custom*

Gambar 3.39. Hasil Akhir Komponen *Cover*

Berikut adalah contoh komponen *Cover* yang sudah dikustomisasi. Seperti yang terlihat pada Gambar 3.39, kustomisasi dilakukan pada bagian nama pengantin, dan juga latar dari undangan dan *card* tersebut. Nama pengantin diambil dari input *user* pada form yang terdapat pada *basic information*. Dan *background* diambil dari foto yang diupload *user* pada laman pengisian detail.

Gambar 3.40. Hasil Akhir Komponen *Thank You*

Selanjutnya adalah komponen *Thank You*. Terdapat perubahan pada titel, pesan dan pesan penutup yang ditunjukkan pada Gambar 3.40. Selain itu, *countdown* pada laman tersebut berubah mengikuti tanggal pernikahan yang diinput oleh *user* pada form yang terdapat di *basic info*. Latar juga berubah menggunakan foto yang di-*upload user*. Hal tersebut menandakan bahwa undangan yang semula semi *hardcoded* berubah menjadi *template* dinamis yang dapat diubah sesuai dengan kemauan *user*.

3.5 Kendala dan Solusi yang Ditemukan

Dalam pelaksanaan rancang bangun web *template wedding e-invitation*, ditemukan beberapa kendala dan diterapkan solusi terhadap kendala tersebut. Berikut adalah kendala yang dihadapi dan solusi yang diterapkan pada proses rancang bangun web *template wedding e-invitation*:

3.5.1 Kendala

Komunikasi antardivisi maupun dengan *supervisor* yang minim dan tidak dilakukan secara langsung. Karena perusahaan beroperasi secara daring (WFH), komunikasi antardivisi seringkali dilakukan melalui WhatsApp dan respons yang diterima cenderung lambat. Akibatnya, komunikasi menjadi kurang lancar dan

masalah teknis sulit dijelaskan tanpa adanya diskusi langsung. Kurangnya komunikasi ini juga menyebabkan kebingungan saat implementasi fitur, karena ketidaktahuan mengenai pihak yang harus dihubungi.

3.5.2 Solusi

Solusi untuk kendala komunikasi adalah menjadwalkan meeting bersama dengan rekan kerja antardivisi, *supervisor*, ataupun pihak terkait lainnya untuk melakukan diskusi langsung. Melalui meeting tersebut, rekan kerja satu dengan yang lain dapat membagikan tampilan layar agar bisa menunjukkan langsung masalah yang setiap anggota tim hadapi dan melakukan perbincangan/diskusi secara langsung.