BAB 3
METODOLOGI PENELITIAN

3.1 Metodologi Penelitian

Bagian ini menjelaskan tahap-tahap yang dilakukan selama menyusun dan mengerjakan penelitian dengan judul " Rancang Bangun Aplikasi Penjadwalan Prodiakon Berbasis Web Menggunakan Algoritma Genetika (Studi Kasus: Gereja Katolik Paroki Alam Sutera)". Tahap penelitian dijabarkan mulai dari studi literatur hingga penulisan laporan. Metodologi penelitian yang digunakan adalah:

1. Studi Literatur

   Tahapan ini berisikan pengumpulan literatur yang relevan dengan penelitian. Literatur yang dikumpulkan berfungsi sebagai dasar teori dalam melakukan penelitian. Literatur yang dikumpulkan mencakup informasi mengenai liturgi, prodiakon, Algoritma Genetika, dan beberapa penelitian penjadwalan pendahulu yang berkaitan dengan penelitian ini. Pencarian literatur yang dilakukan pada penelitian dilakukan dengan menjelajahi beberapa situs web resmi, seperti Mendeley, Google Scholar, portal berita resmi, dan lain-lain.

2. Pengumpulan Data

   Pada tahapan pengumpulan data, data yang didapat berasal dari data yang telah dikumpulkan oleh pihak gereja.

3. Perancangan

   Tahapan perancangan meliputi perancangan desain dari web serta pengembangan web dari sisi *frontend* dan *backend*.

4. Implementasi

   Pada tahapan implementasi, dilakukan pembuatan sistem berdasarkan perancangan yang telah dibuat sebelumnya dan menerapkan informasi yang telah didapatkan dari studi literatur. Tahapannya adalah implementasi Algoritma Genetika dalam penjadwalan prodiakon berdasarkan batasan yang telah ditetapkan.

5. Pengujian

   Pada tahapan ini, pengujian yang dilakukan berupa pencobaan penjadwalan dengan Algoritma Genetika dalam berbagai kondisi dan tingkat kepuasan

pengguna dalam menggunakan aplikasi. Tujuan pengujian ini adalah untuk mengukur kinerja dari aplikasi yang dihasilkan menggunakan EUCS dengan Skala Likert.

6. Penulisan laporan
   Penulisan laporan dilakukan ketika pengumpulan data, perancangan, implementasi, dan pengujian telah dilakukan.

3.2 Analisis Sistem

Hasil analisis kebutuhan proses dalam aplikasi penjadwalan prodiakon menggunakan Algoritma Genetika pada studi kasus Gereja Katolik Paroki Alam Sutera dibagi menjadi dua hal. Hal pertama adalah kebutuhan *input* dan *output* dari sistem. Kebutuhan *input* dari aplikasi ini meliputi data induk (*master data*) yang mencakup data gereja, dan data prodiakon, data jadwal misa gereja.

Hal kedua adalah kebutuhan dari segi fungsionalitas. Gereja Katolik Paroki Alam Sutera masih melakukan penjadwalan prodiakon secara *manual*, yaitu dengan menggunakan Ms Excel. Fungsionalitas aplikasi penjadwalan ini diharapkan dapat melakukan penjadwalan prodiakon dengan lebih cepat dan melihat jadwal bertugas dengan lebih cepat. Berdasarkan hasil analisis kebutuhan fungsionalitas, terdapat kebutuhan untuk membuat untuk dua sistem berbeda yang dibedakan dengan hak admin. Pada sisi admin, terdapat pengelolaan data induk (*master data*), serta pembuatan jadwal misa untuk memenuhi kebutuhan dalam pembuatan jadwal misa. Di sisi *non* admin, terdapat daftar jadwal bertugas untuk memenuhi kebutuhan dalam melihat jadwal bertugas.

Pembuatan aplikasi diputuskan menggunakan Algoritma Genetika sebagai algoritma utama dalam melakukan penjadwalan prodiakon. Aplikasi berbasis web dibuat dengan menggunakan *framework* ReactJS pada sisi *frontend* dan *framework* ExpressJS pada sisi *backend* dengan PostGreSQL sebagai *database*. Algoritma Genetika diterapkan pada sisi *backend* untuk melakukan penjadwalan dan hasilnya disimpan pada *database* yang kemudian dikirimkan ke sisi *frontend* melalui *Application Programming Interface* atau API.

3.3 Perancangan Aplikasi

Berdasarkan hasil analisis sistem, maka dirancang suatu aplikasi penjadwalan prodiakon berbasis web dengan beberapa fungsionalitas.

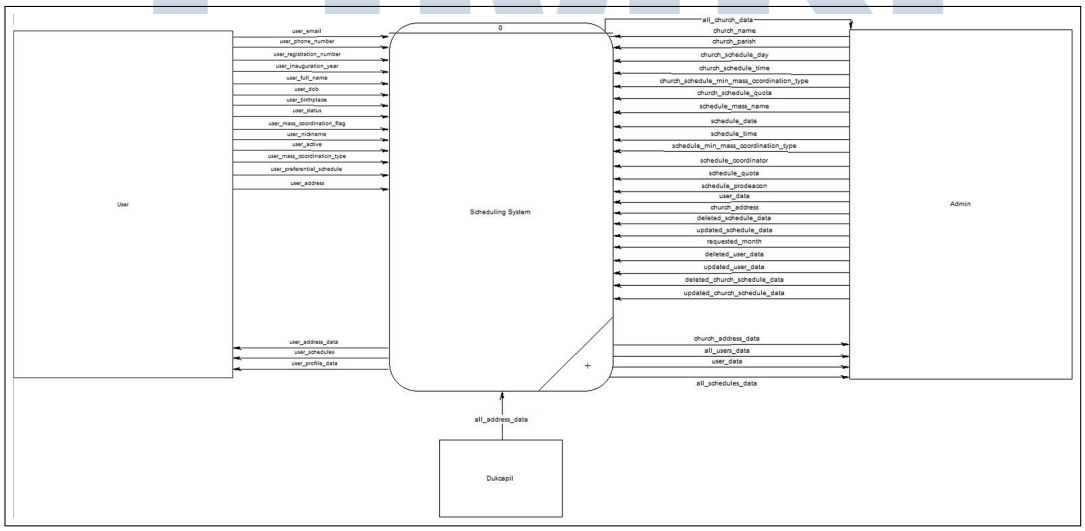
Fungsionalitas pada sistem dapat dilihat sebagai berikut:

- Proses *create, read, update, delete master data*
  Proses ini memungkinkan admin untuk mengelola data gereja, data prodiakon, dan data jadwal misa tetap suatu gereja.
- Proses *create, read, update, delete* jadwal misa
  Proses ini memungkinkan admin untuk mengelola jadwal misa suatu gereja.
- Proses pembuatan jadwal misa bulanan
  Proses ini menghasilkan jadwal misa bulanan pada suatu gereja. Jadwal yang dibuat akan berdasarkan batasan-batasan masalah yang telah ditetapkan.
- Proses melihat jadwal bertugas
  Pada proses ini, pengguna tanpa hak admin dapat melihat jadwal bertugas pada bulan tertentu.

Aplikasi yang dibangun pada penelitian ini dijelaskan melalui *Data Flow Diagram* (DFD), Diagram Alir (*Flowchart*), skema *database*, dan Desain Antar Muka (*Wireframe*).

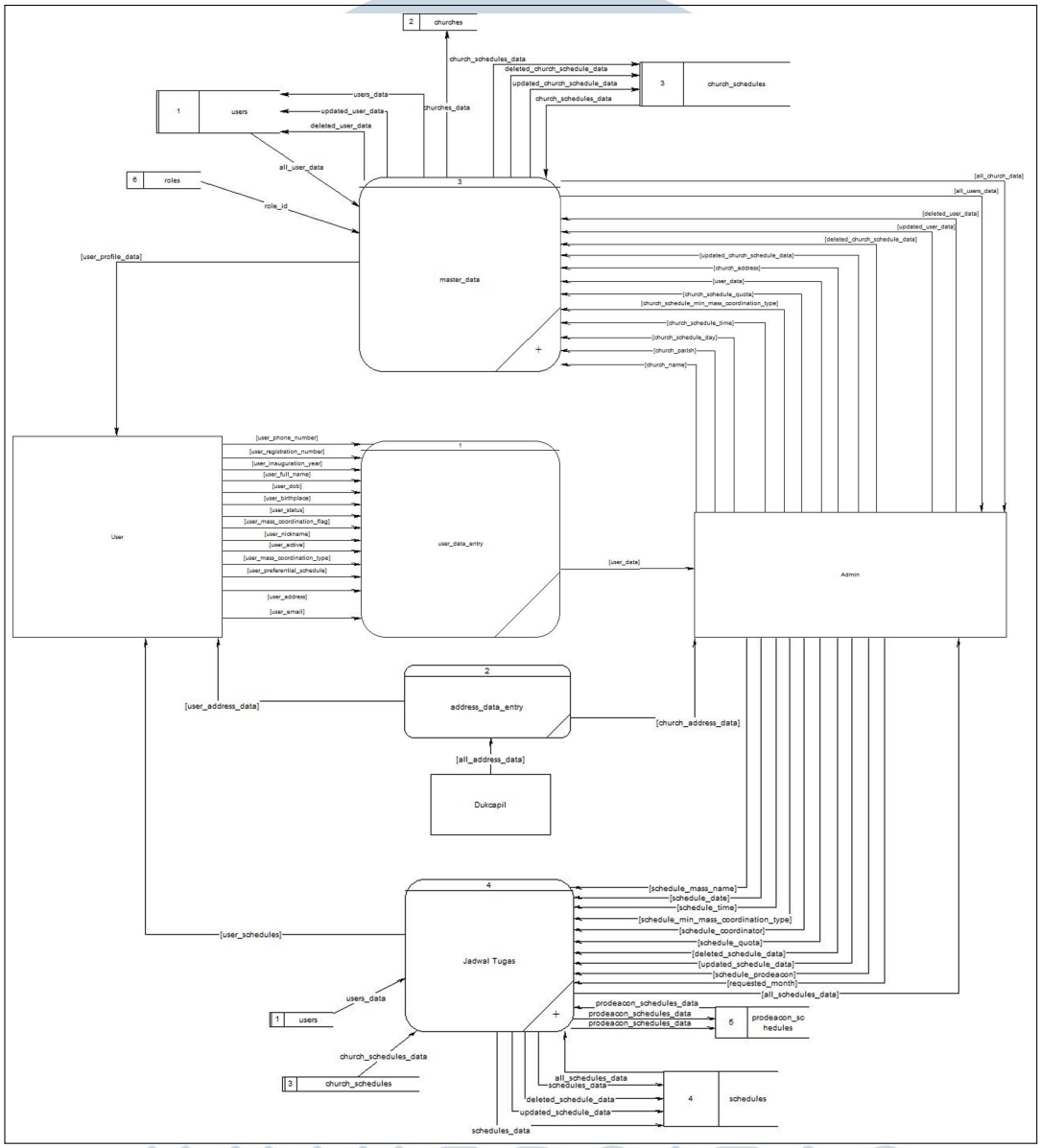
3.3.1 Data Flow Diagram

Alur data pada sistem penjadwalan dijabarkan dengan menggunakan Data Flow Diagram yang diturunkan menjadi beberapa level. *Data Flow Diagram* dimulai dari *context diagram* yang ditunjukkan oleh Gambar 3.1.



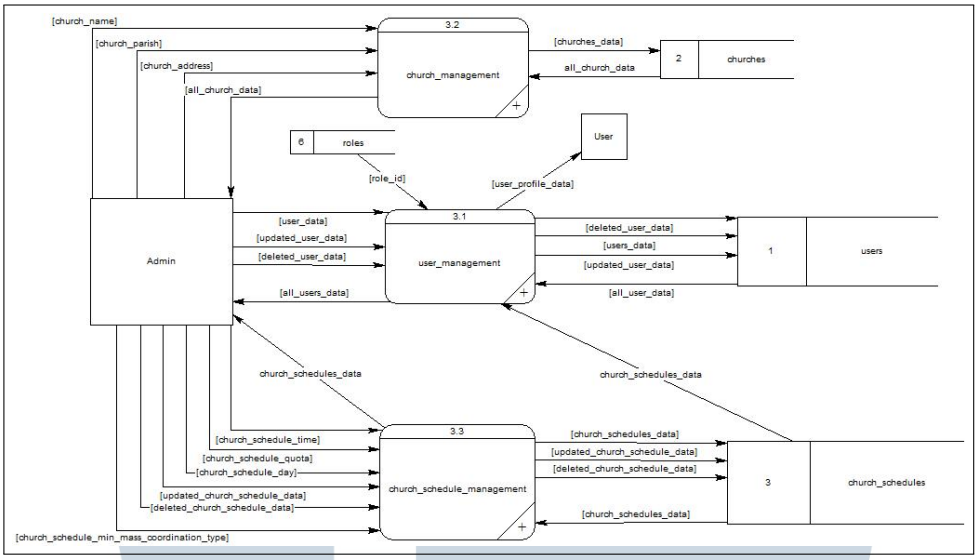
Gambar 3.1. *Context Diagram*

Gambar 3.1 menunjukkan *context diagram* aplikasi penjadwalan yang menggambarkan alur data secara keseluruhan.



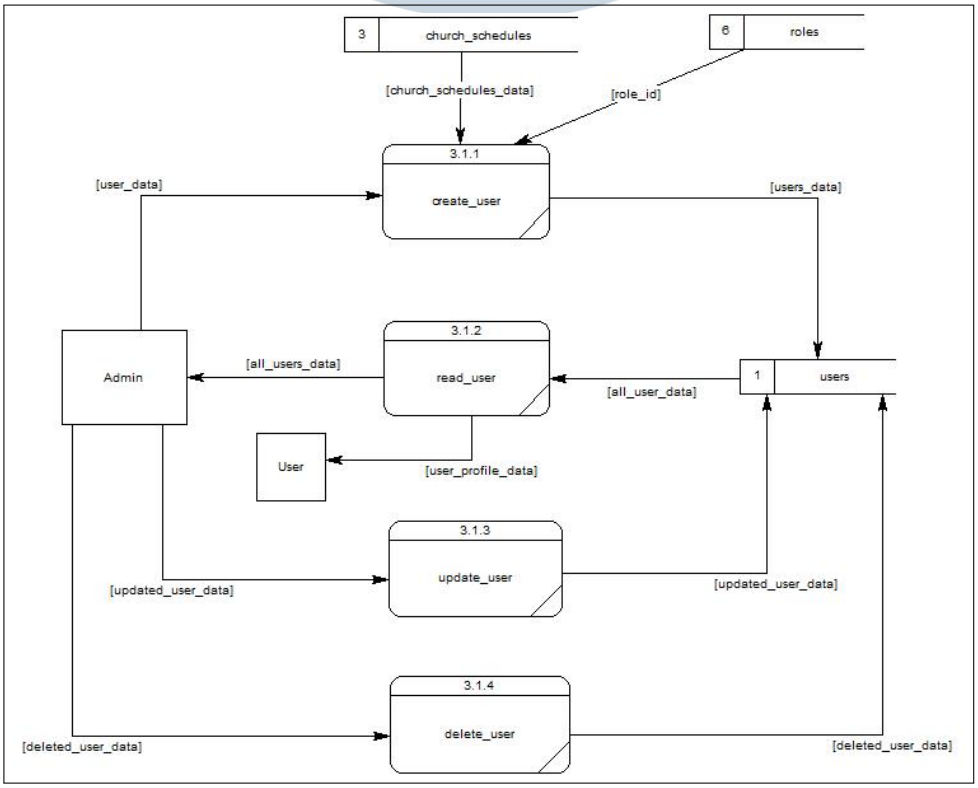
Gambar 3.2. DFD Level 1 *Scheduling System*

Gambar 3.2 menunjukkan DFD level 1 yang mencakup 4 proses utama, yaitu *master_data* (untuk mengelola data pada aplikasi), jadwal_tugas (untuk mengelola jadwal pada aplikasi), *user_data_entry* (untuk mendapatkan data prodiakon dari Google Form), dan *address_data_entry* (untuk mendapatkan semua data alamat dari dukcapil).



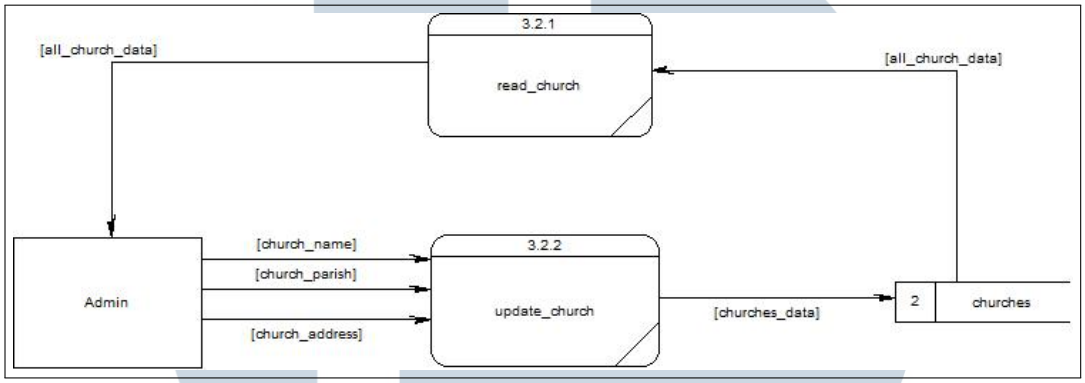
Gambar 3.3. DFD Level 2 *master_data*

Gambar 3.3 menunjukkan DFD level 2 yang menggambarkan proses dari *master_data*. Pada gambar tersebut, terdapat 3 sub proses, yaitu *user_management*, *church_management*, dan *church_schedule_management*.



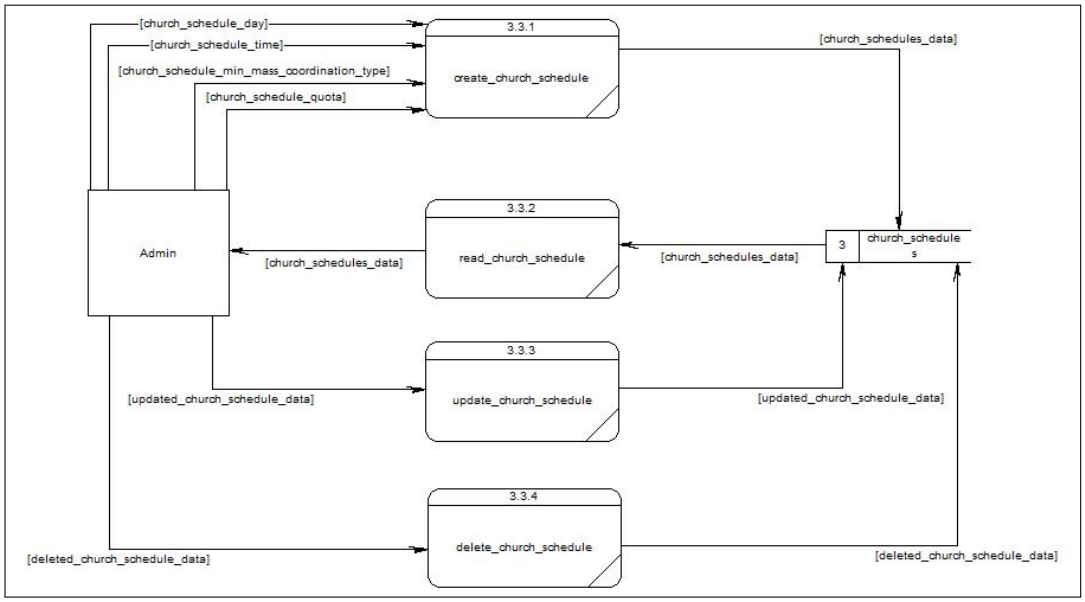
Gambar 3.4. DFD Level 3 *user_management*

Gambar 3.4 menunjukkan DFD level 3 yang menjelaskan sub proses dari *user_management*. Terdapat 4 sub proses, yaitu membuat *user* baru, membaca data *user*, memperbarui *user*, dan menghapus *user*. Pada bagian ini, diperlukan data *church_schedules* untuk mendapatkan semua jadwal pada sebuah gereja.



Gambar 3.5. DFD Level 3 *church_management*

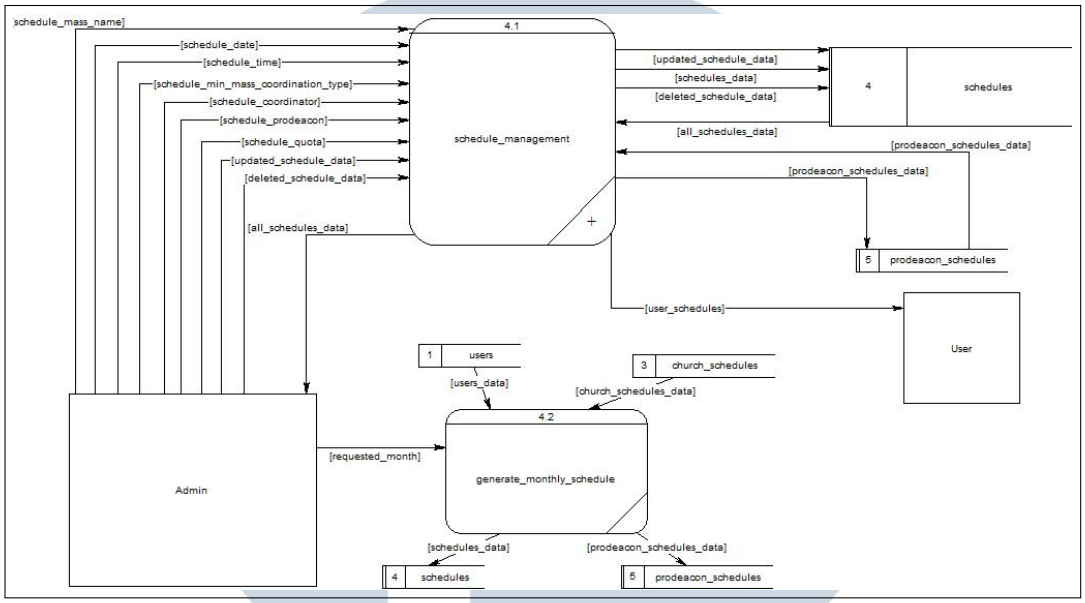
Gambar 3.5 menunjukkan DFD level 3 yang menjelaskan sub proses dari *church_management*. Terdapat 2 sub proses, yaitu membaca data gereja, dan memperbarui gereja.



Gambar 3.6. DFD Level 3 *church_schedule_management*

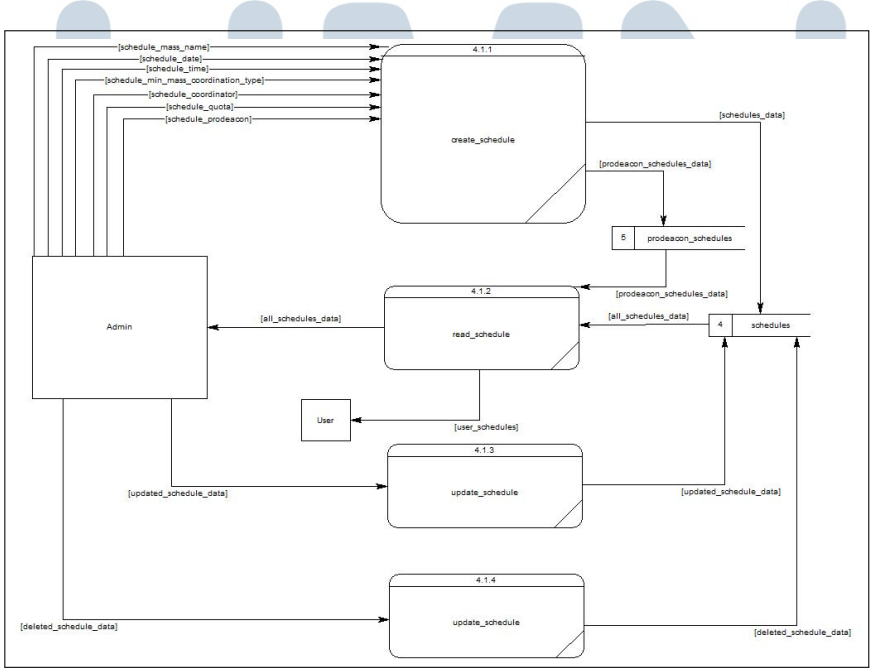
Gambar 3.6 menunjukkan DFD level 3 yang menjelaskan sub proses dari *church_schedule_management*. Terdapat 4 sub proses, yaitu membuat data

jadwal gereja baru, membaca data jadwal gereja, memperbarui jadwal gereja, dan menghapus jadwal gereja.



Gambar 3.7. DFD Level 2 jadwal_tugas

Gambar 3.7 menunjukkan DFD level 2 yang menggambarkan proses dari jadwal_tugas. Pada gambar tersebut, terdapat 2 sub proses, yaitu *schedule_management*, dan *generate_monthly_schedule*.

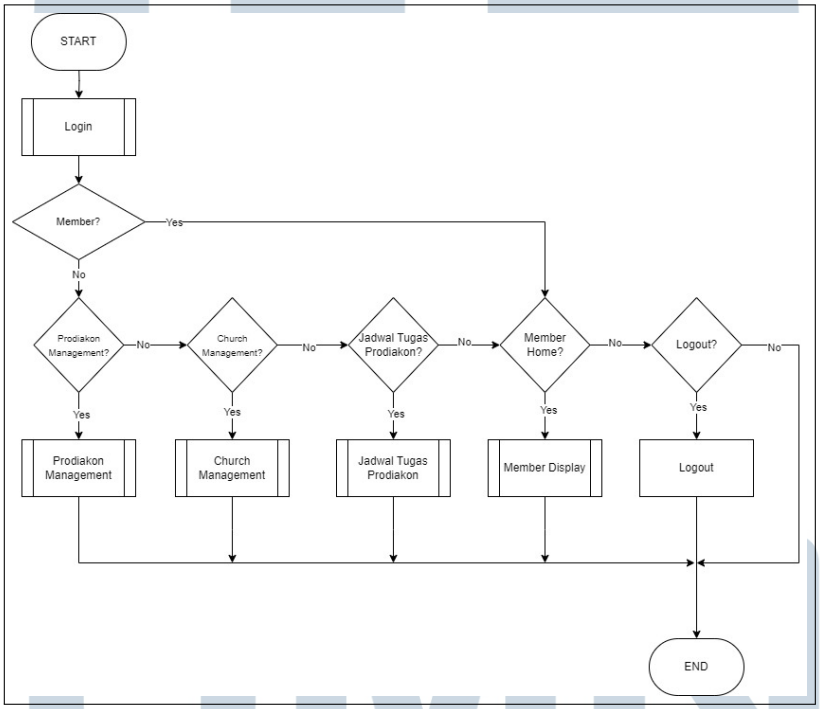


Gambar 3.8. DFD Level 3 *schedule_management*

Gambar 3.8 menunjukkan DFD level 3 yang menjelaskan sub proses dari *schedule_management*. Terdapat 4 sub proses, yaitu membuat data jadwal baru, membaca data jadwal, memperbarui jadwal, dan menghapus jadwal. Pada proses ini, data jadwal dibutuhkan dan data jadwal diambil dari tabel schedules.

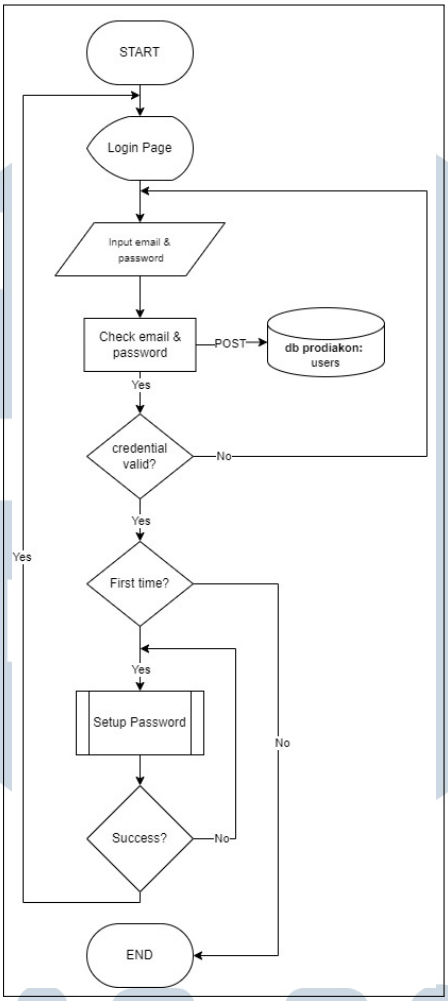
3.3.2 Flowchart

Alur pada aplikasi penjadwalan dijabarkan dalam bentuk *flowchart*. Tiap proses akan dijabarkan dari Gambar 3.9 hingga Gambar 3.27.



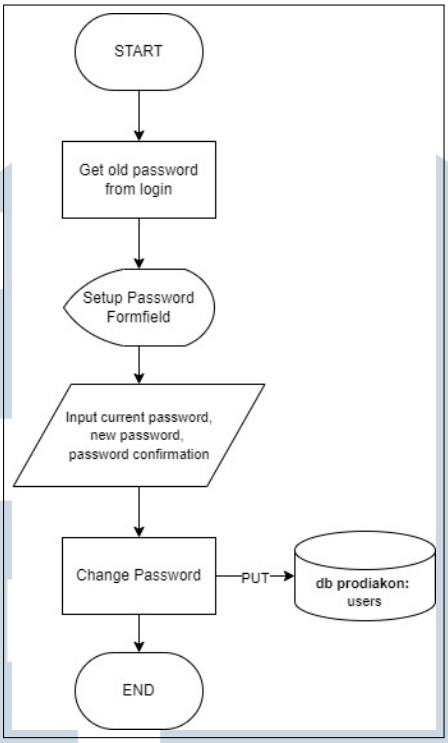
Gambar 3.9. *Flowchart general* aplikasi

Gambar 3.9 menggambarkan *flowchart general* pada aplikasi. Proses diawali dengan *login* dan pengecekan apakah pengguna merupakan *member*. Kemudian, pengguna akan diarahkan menuju halaman-halaman yang ada pada aplikasi.



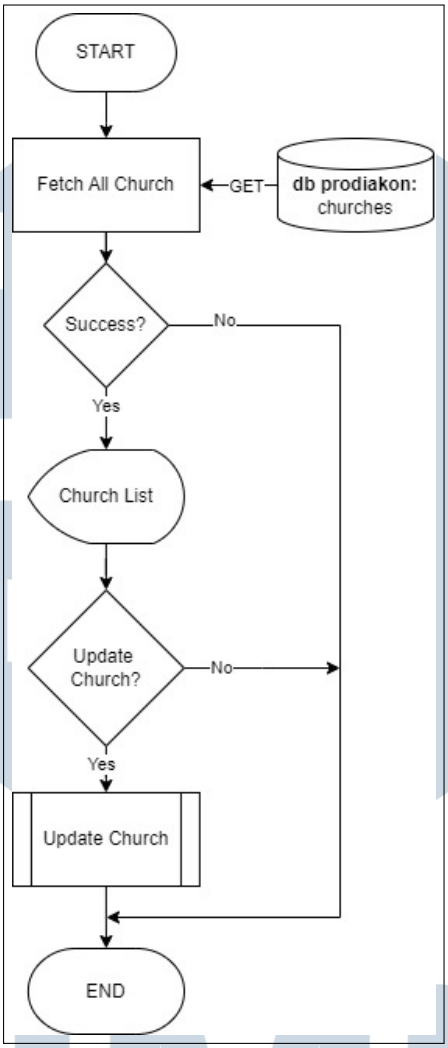
Gambar 3.10. Flowchart *Login*

Gambar 3.10 menggambarkan *flowchart login*. Proses diawali dengan menampilkan halaman *login*, dan pengguna dapat memasuki *email* dan *password*. Kemudian akan dilakukan pemrosesan *input* ke *database*, dan apabila sukses akan dilanjutkan untuk pengecekan terhadap *input* dengan data di *database*. Jika *valid*, akan dilakukan pengecekan apakah pengguna baru pertama kali melakukan *login*. Apabila iya, pengguna akan diarahkan untuk melakukan *setup password*.



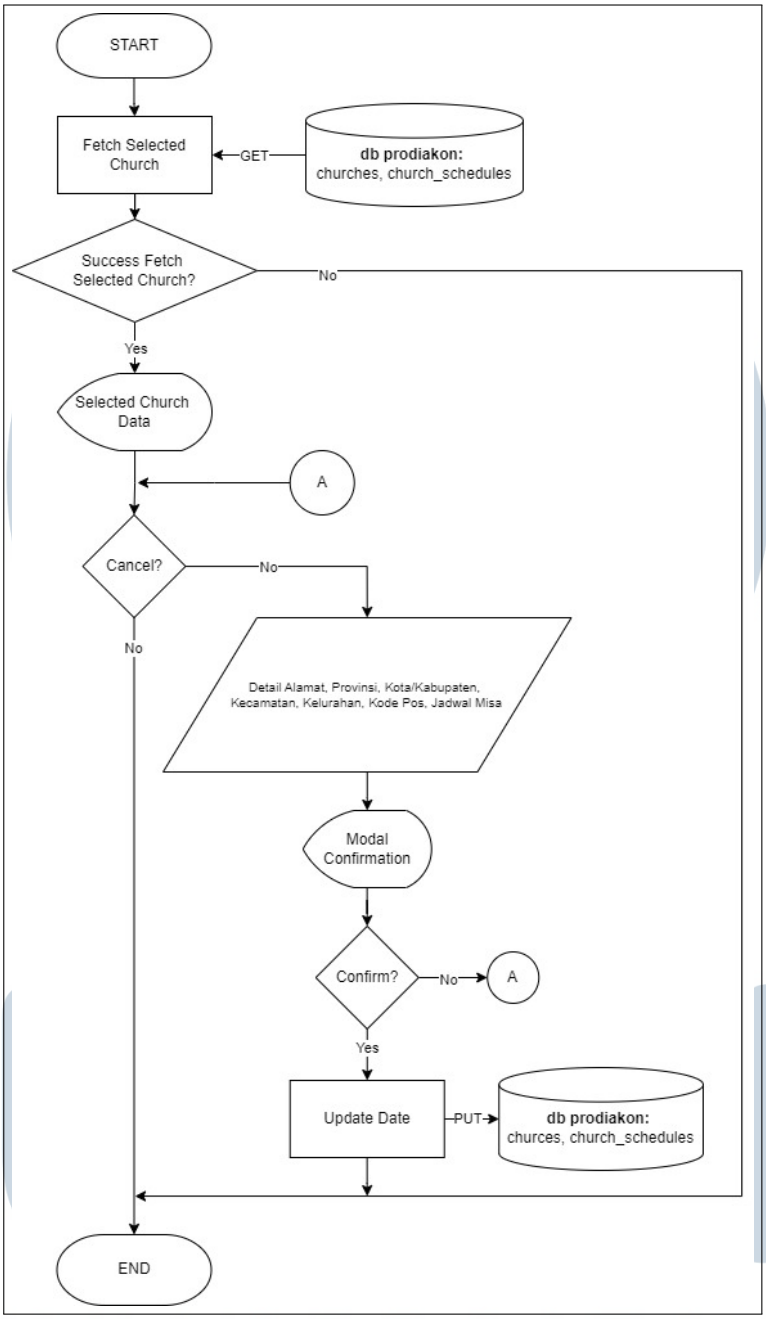
Gambar 3.11. *Flowchart Setup Password*

Gambar 3.11 menunjukkan *flowchart setup password* yang diawali dengan mengambil *password* dari proses *login*. Kemudian, pengguna diminta untuk memasukkan *password* lama, *password* baru, serta konfirmasi *password* yang baru. Lalu, dilakukan pengecekan terhadap *database*, dan apabila tidak sukses, pengguna akan kembali ke halaman *login*.



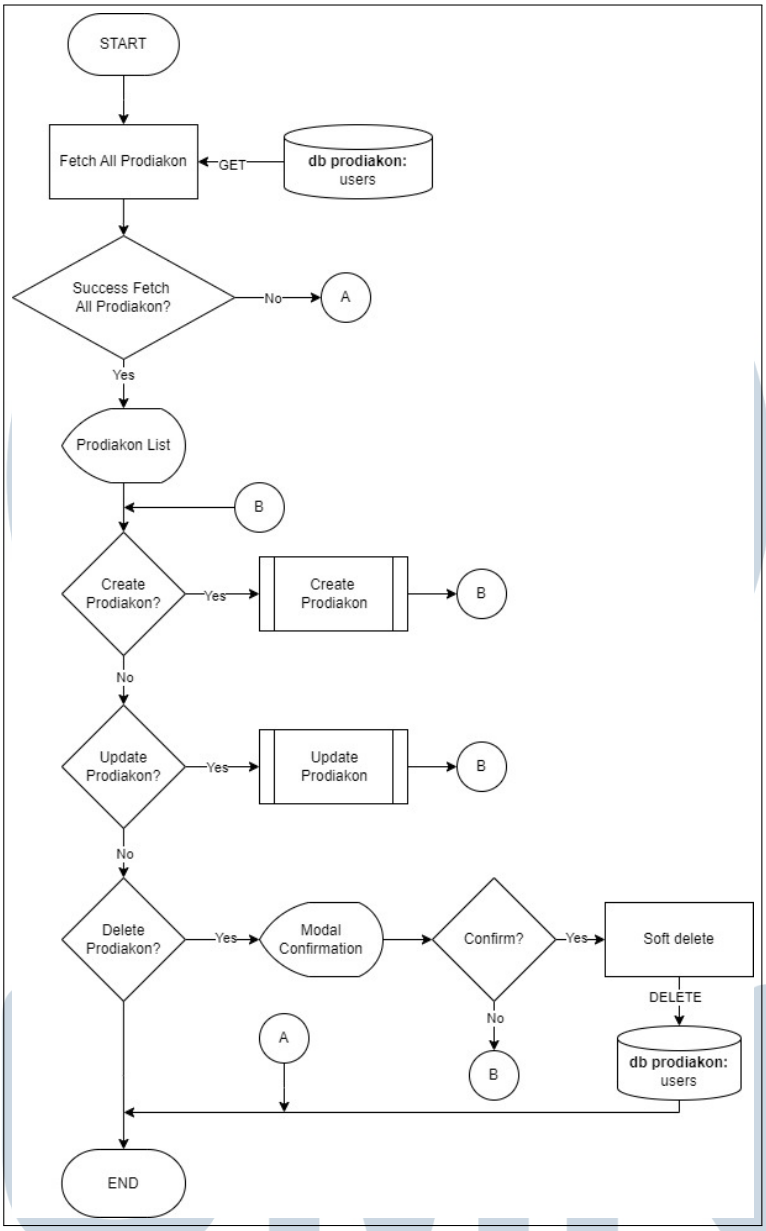
Gambar 3.12. *Flowchart Church Management*

Gambar 3.12 menggambarkan *flowchart* daftar gereja yang diawali dengan mengambil data dari *database*. Apabila sukses, semua data gereja akan ditampilkan. Pengguna dapat memilih untuk *update* gereja.



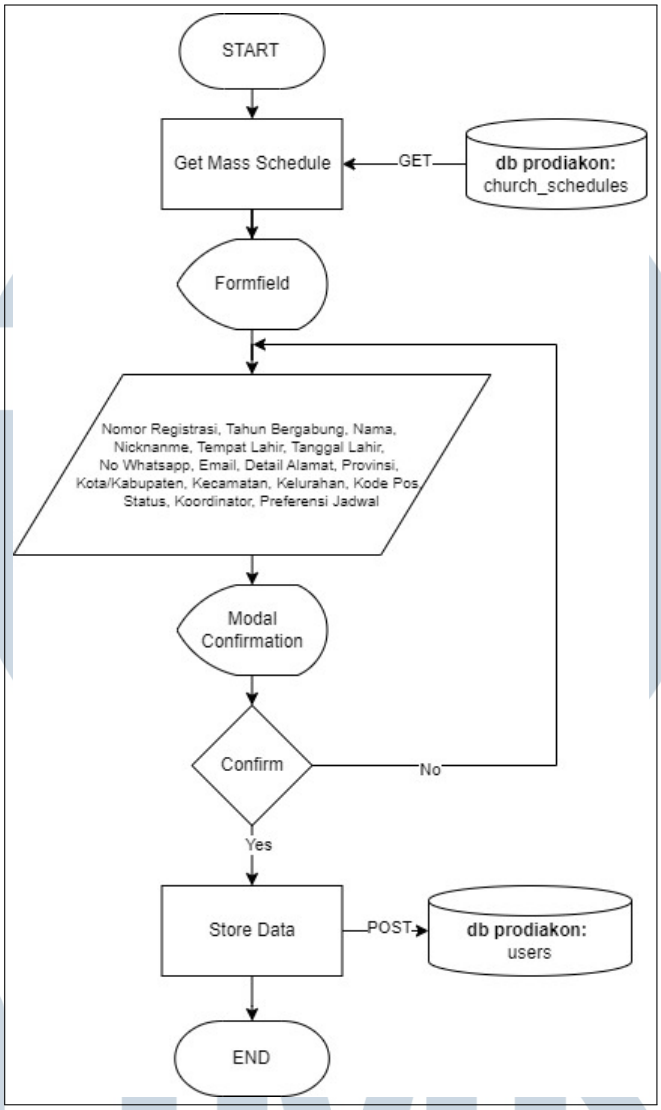
Gambar 3.13. *Flowchart Update Church*

Gambar 3.13 menggambarkan *flowchart update* gereja yang diawali dengan mengambil data dari *database*. Apabila sukses, maka akan ditampilkan semua data gereja yang dipilih. Pengguna dapat memilih untuk *edit* gereja dengan mengisi data yang diperlukan. Setelah itu akan muncul modal konfirmasi, dan akan dilakukan *update* terhadap data yang ada di *database*.



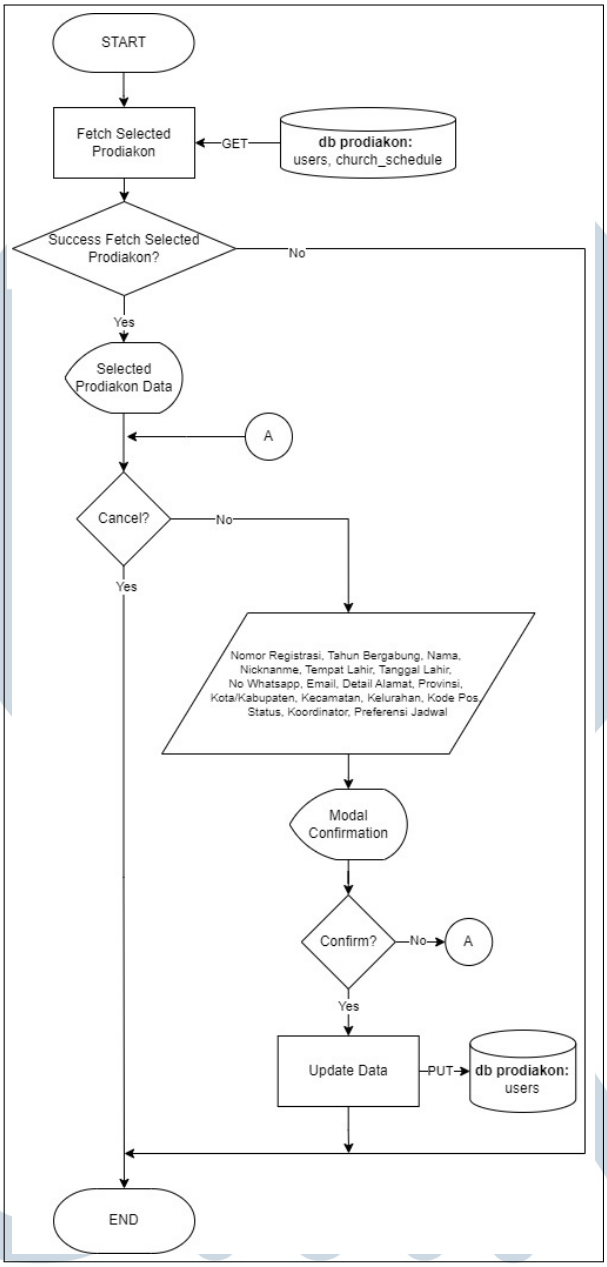
Gambar 3.14. *Flowchart Prodiakon Management*

Gambar 3.14 menggambarkan *flowchart* daftar prodiakon yang diawali dengan mengambil data dari *database*. Apabila sukses, maka akan ditampilkan semua daftar prodiakon. Pengguna dapat memilih untuk membuat prodiakon baru, *edit* prodiakon, atau menghapus prodiakon. Jika pengguna memilih untuk menghapus prodiakon, akan muncul *modal* konfirmasi untuk memastikan prodiakon terhapus dengan cara *soft delete* ke *database*.



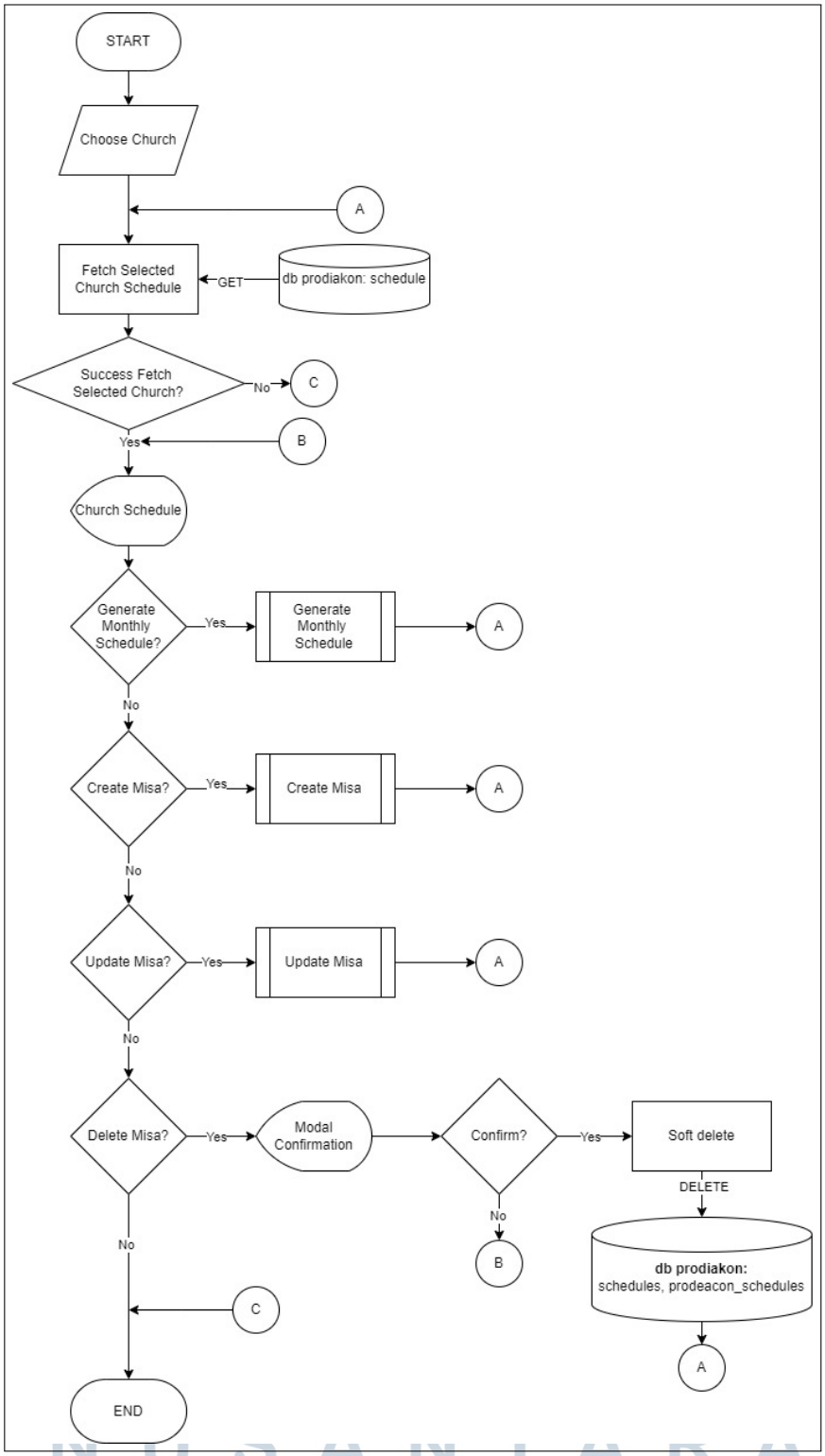
Gambar 3.15. *Flowchart Create* Prodiakon

Gambar 3.15 menggambarkan *flowchart* ketika pengguna ingin menambahkan prodiakon baru yang diawali dengan mengambil data jadwal gereja dari *database*. Apabila sukses, maka akan ditampilkan semua *input* yang perlu dimasukkan. Setelah selesai mengisi, akan dikonfirmasi melalui modal konfirmasi, yang kemudian akan diikuti dengan menyimpan data ke *database*.



Gambar 3.16. *Flowchart Update* Prodiakon

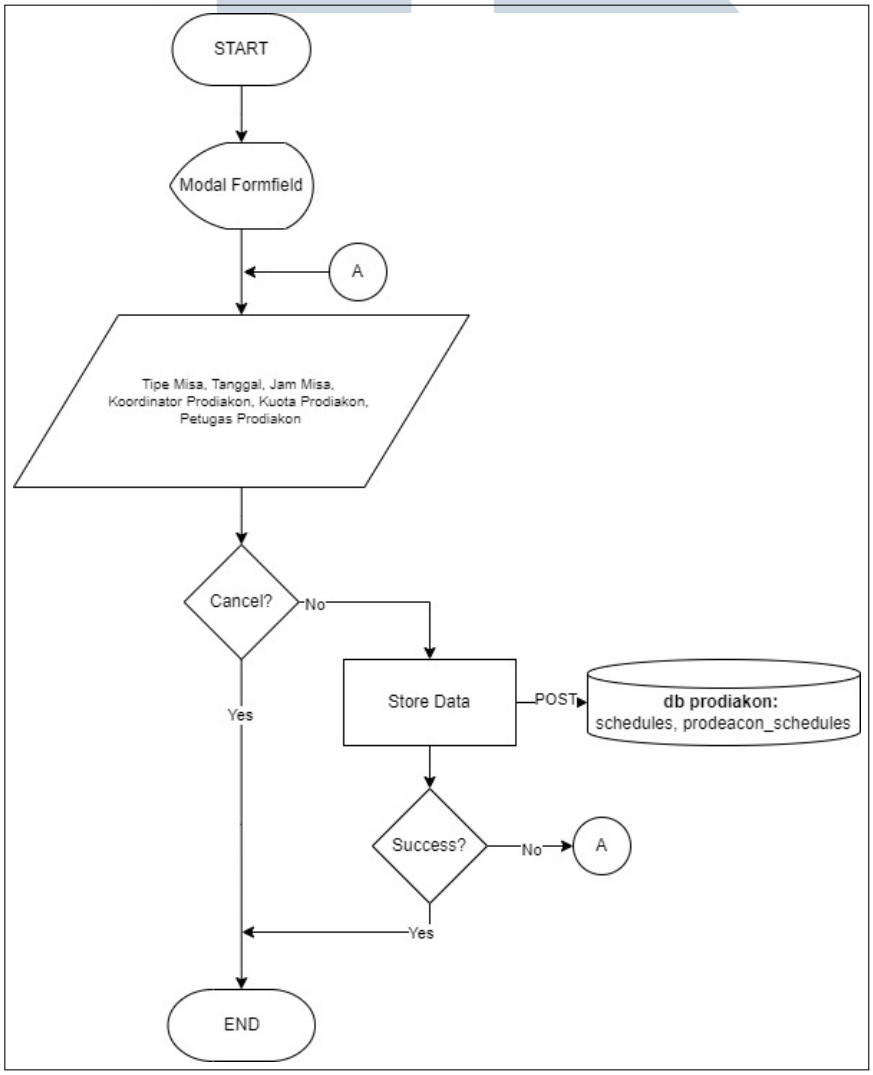
Gambar 3.16 menggambarkan *flowchart update* prodiakon yang diawali dengan mengambil data jadwal dan data prodiakon yang dipilih dari *database*. Apabila sukses, maka akan ditampilkan semua data milik prodiakon tersebut dan pengguna dapat melakukan perubahan terhadap data tersebut. Setelah selesai melakukan *edit*, akan dikonfirmasi melalui modal konfirmasi, yang kemudian akan diikuti dengan menyimpan data ke *database*.



Gambar 3.17. *Flowchart* Jadwal Tugas Prodiakon

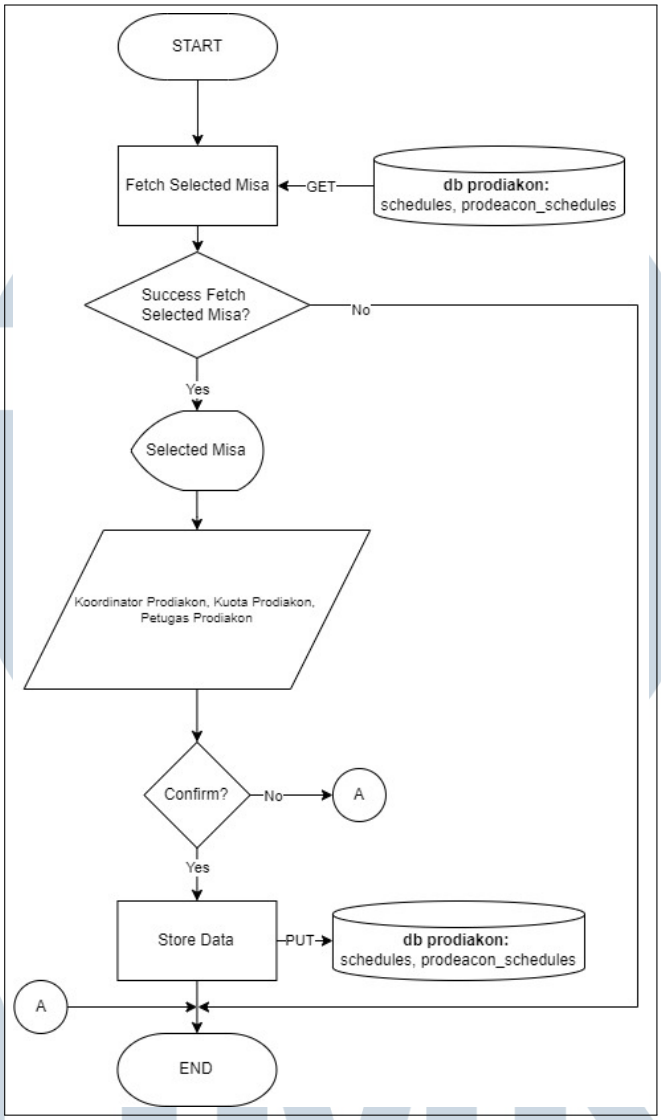
Gambar 3.17 menggambarkan *flowchart* jadwal tugas prodiakon yang diawali dengan memilih gereja dan nantinya akan mengambil data jadwal gereja

tersebut dari *database*. Apabila sukses, maka akan ditampilkan semua jadwal pada bulan yang terpilih di tanggal tersebut dan pengguna dapat melakukan perubahan terhadap jadwal tersebut, mulai dari pembuatan jadwal satu bulan, tambah jadwal misa, *update* jadwal misa, dan hapus jadwal misa.



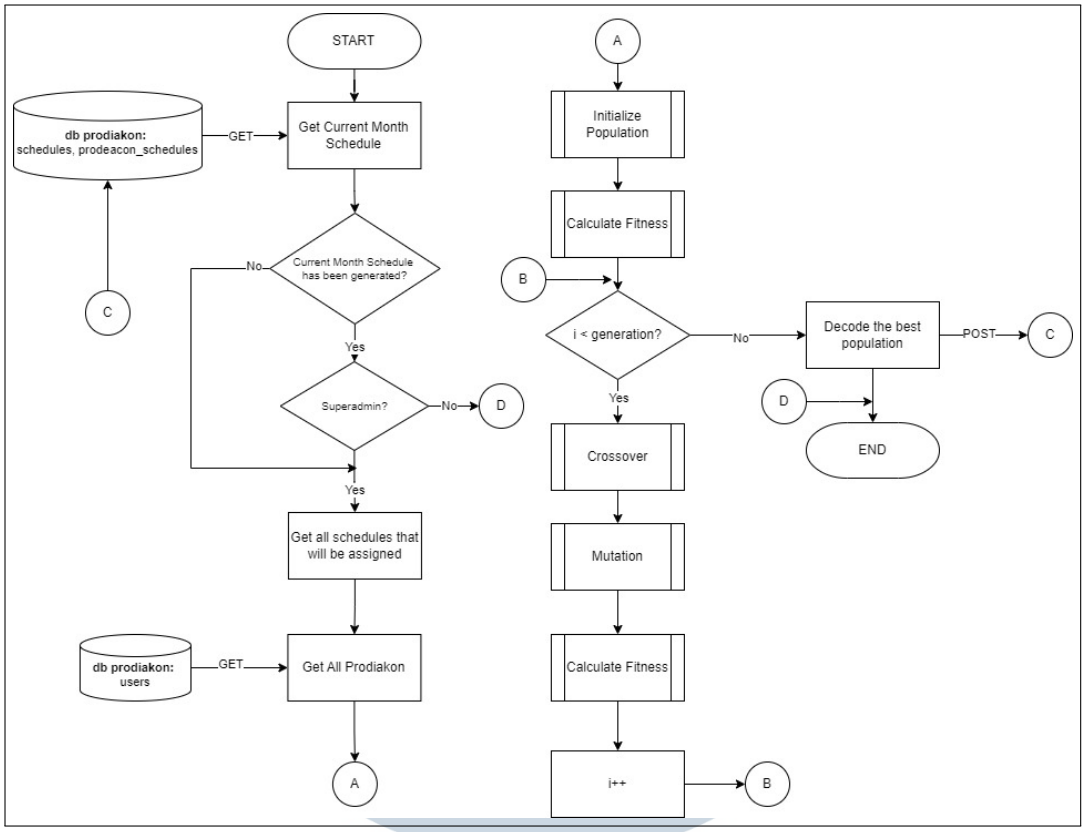
Gambar 3.18. *Flowchart Create* Misa

Gambar 3.18 menggambarkan *flowchart* membuat jadwal misa baru yang diawali dengan menampilkan *input* yang perlu diisi pengguna. Setelah diisi, data tersebut akan dimasukkan ke dalam *database*.



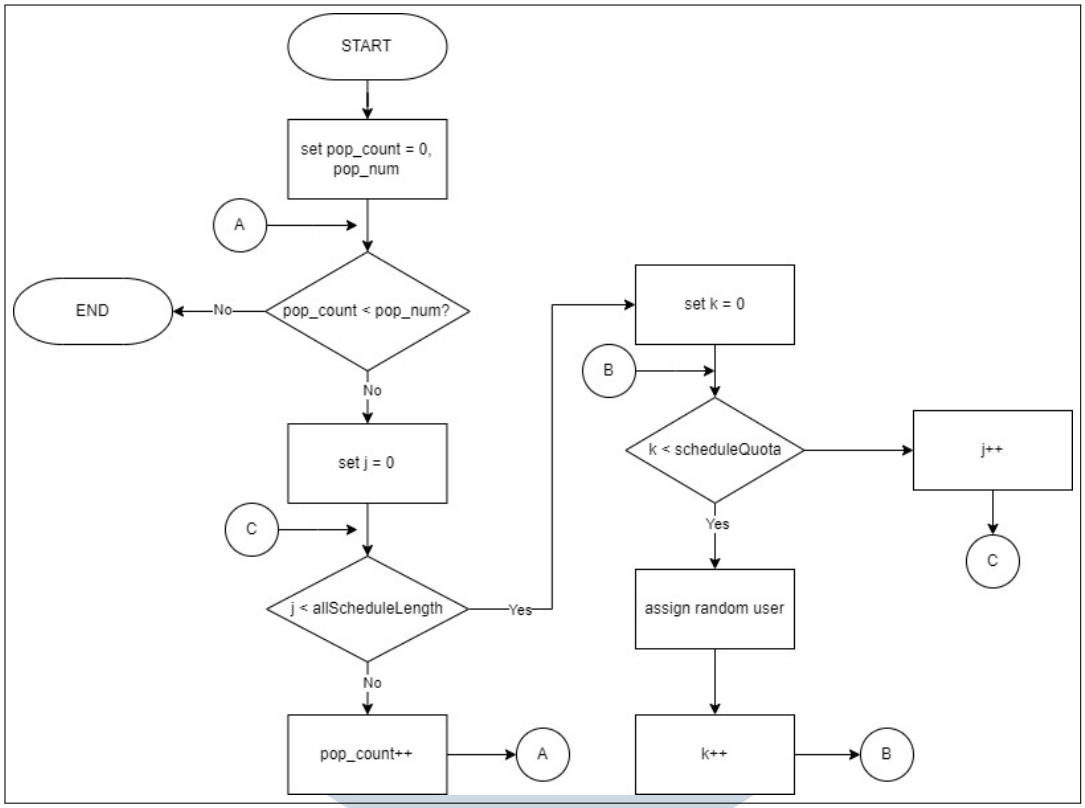
Gambar 3.19. *Flowchart Update* Misa

Gambar 3.19 menggambarkan *flowchart update* jadwal misa yang diawali dengan mengambil data jadwal yang telah dipilih dari *database* dan menampilkannya kepada pengguna. Pengguna dapat melakukan perubahan data. Setelah selesai melakukan perubahan, data tersebut akan dimasukkan ke dalam *database*.



Gambar 3.20. *Flowchart Generate Monthly Schedule*

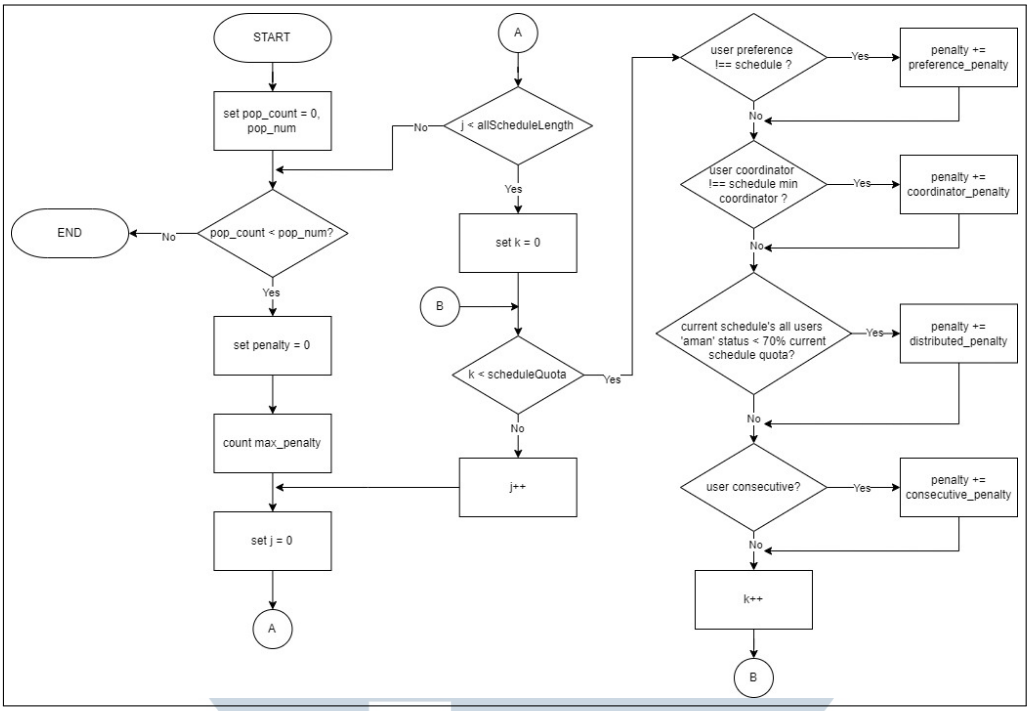
Gambar 3.20 menggambarkan *flowchart* pembuatan jadwal bulanan. Algoritma Genetika diimplementasikan pada proses ini. Proses diawali dengan mengambil data bulan tersebut dari *database* dan kemudian dilakukan pengecekan apakah jadwal pada bulan tersebut telah dibuat. Apabila jadwal telah terbuat, akan dilakukan pengecekan apakah pengguna merupakan *superadmin*. Jika pengguna adalah *superadmin*, akan dipilih jadwal pada bulan tersebut yang nantinya akan dipasangkan dengan prodiakon. Lalu, akan dilakukan inisialisasi populasi sebagai langkah awal dari tahapan Algoritma Genetika dan diikuti dengan perhitungan *fitness* pada populasi awal. Kemudian akan dilakukan iterasi sejumlah generasi yang telah ditentukan. Pada masing-masing iterasi, akan dilakukan perhitungan *fitness* pada tiap individu, yang akan dilanjutkan dengan *crossover*. Pada proses *crossover*, akan dilakukan pemilihan induk dengan metode *tournament selection*. Lalu, akan dilakukan *crossover* atau kawin silang dan diakhiri dengan mutasi. Setelah selesai melakukan semua iterasi, tiap individu pada populasi terbaik akan dipecah ke dalam bentuk yang akan disimpan ke *database*.



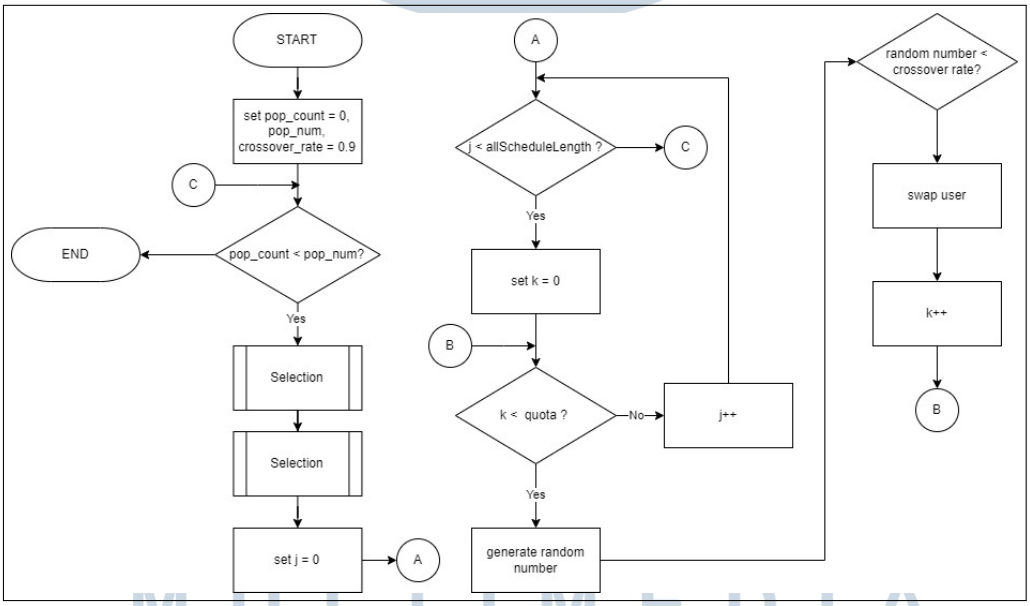
Gambar 3.21. Flowchart *Initialize Population*

Gambar 3.21 menggambarkan *flowchart initialize population* atau proses inisialisasi populasi awal. Proses tersebut diawali dengan menentukan ukuran populasi dan dilakukan iterasi sejumlah ukuran populasi yang telah ditentukan. Pada tiap iterasi, tiap jadwal akan diisi oleh prodiakon secara acak sesuai dengan jumlah kuota yang dibutuhkan masing-masing jadwal.

Gambar 3.22 menggambarkan *flowchart calculate fitness* yang berfungsi untuk menghitung nilai *fitness* pada tiap kromosom pada suatu populasi. Proses perhitungan *fitness* melakukan iterasi sejumlah ukuran populasi. Pada masing-masing prodiakon, dilakukan pengecekan terhadap sejumlah batasan yang telah ditentukan, mulai dari preferensi jadwal prodiakon, jenis minimum koordinator prodiakon, terdistribusinya status prodiakon pada suatu jadwal, dan berurutannya jadwal bertugas prodiakon. Apabila terdapat kondisi yang melanggar batasan ini, maka nilai *penalty* akan bertambah.



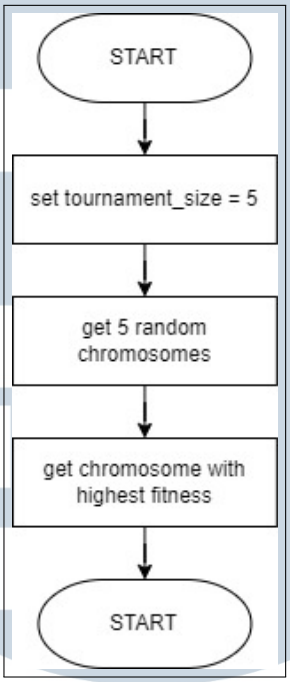
Gambar 3.22. Flowchart *Calculate Fitness*



Gambar 3.23. Flowchart *Crossover*

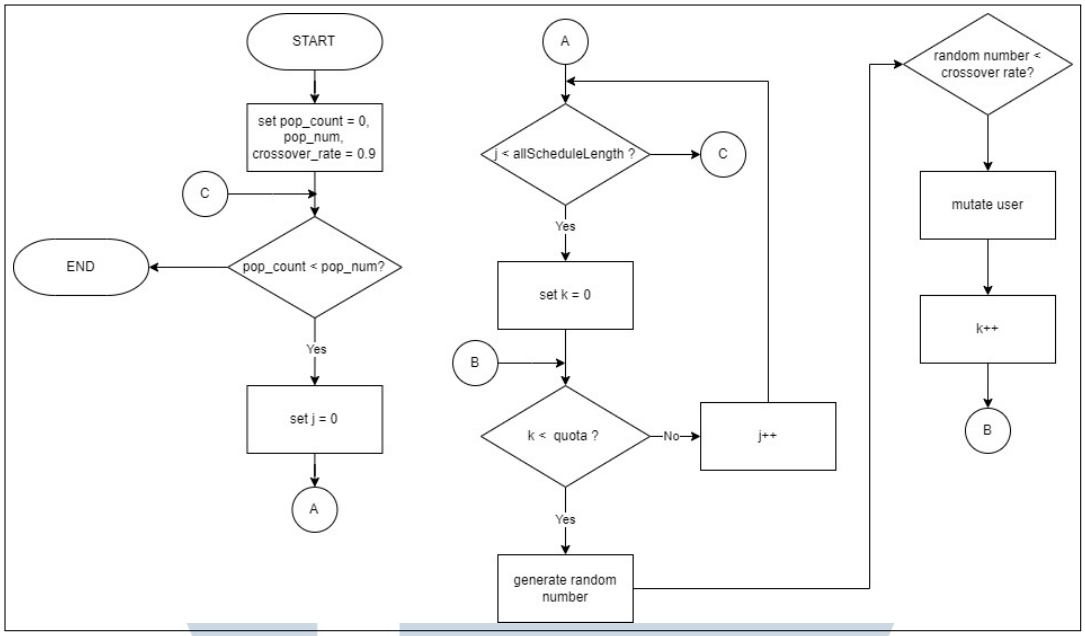
Gambar 3.23 menggambarkan *flowchart crossover* yang diawali dengan mengiterasikan setiap kromosom pada suatu populasi. Pada tiap iterasi, dilakukan seleksi untuk mendapatkan dua buah *parent* atau induk untuk *crossover*. Setelah mendapatkan dua induk, maka dilakukan pengecekan terhadap tiap prodiakon.

Sebuah angka acak akan dipilih dan apabila angka acak tersebut lebih kecil dibanding *crossover rate*, akan dilakukan *crossover* antar dua induk terhadap satu prodiakon.



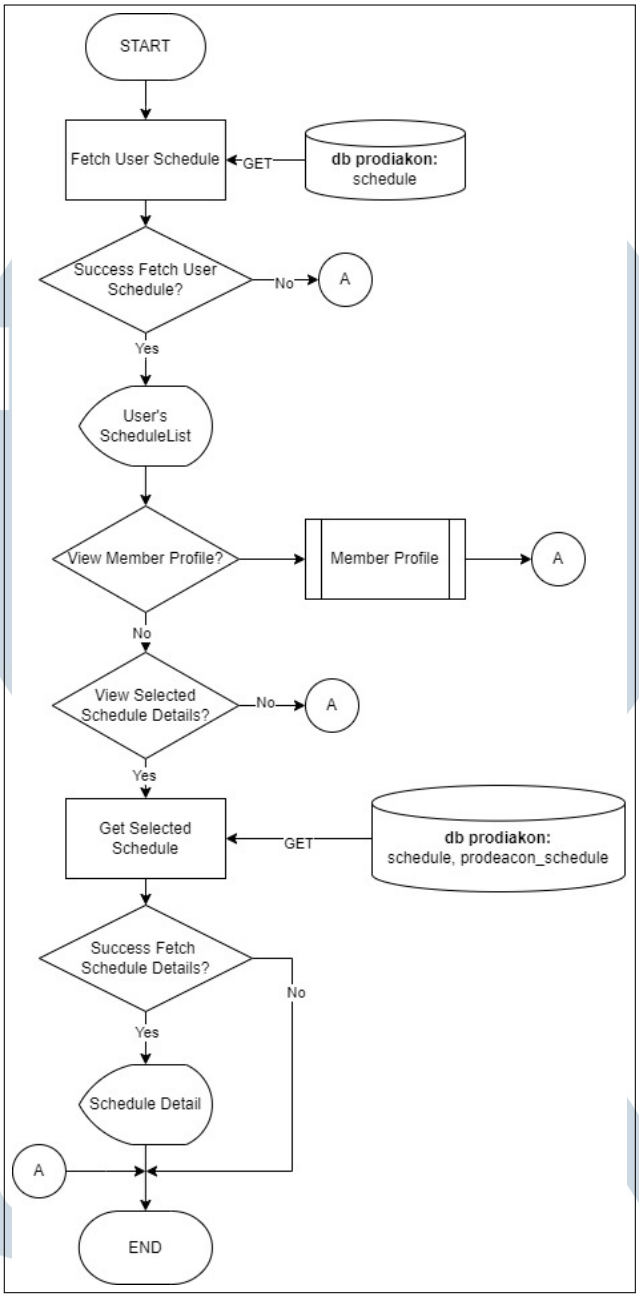
Gambar 3.24. Flowchart *Selection*

Gambar 3.24 menggambarkan *flowchart selection* yang diawali dengan mengambil 5 kromosom secara acak dari suatu populasi. Dari 5 buah kromosom tersebut, diambil salah satu kromosom dengan nilai *fitness* tertinggi untuk dijadikan induk *crossover*.



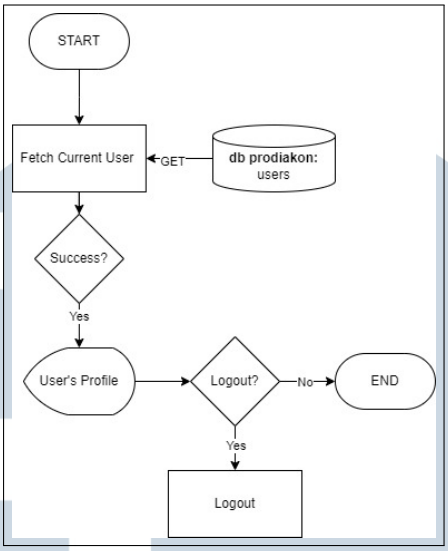
Gambar 3.25. Flowchart *Mutation*

Gambar 3.25 menggambarkan *flowchart mutation* yang diawali dengan mengiterasikan setiap kromosom pada suatu populasi. Pada tiap iterasi kromosom, dilakukan pengecekan terhadap tiap jadwal pada kromosom tersebut. Pada tiap jadwal, tiap prodiakon akan dilakukan pengecekan dengan membuat sebuah angka acak, dan apabila angka acak lebih kecil dari pada *mutation rate*, maka prodiakon tersebut akan dimutasi.



Gambar 3.26. Flowchart *Member Display*

Gambar 3.26 menggambarkan *flowchart* jadwal bulanan pada tampilan *member* yang diawali dengan mengambil data jadwal dari *database* dan menampilkannya kepada pengguna. Pengguna dapat memilih untuk melihat *profile* miliknya sendiri. Pengguna dapat memilih suatu jadwal misa untuk melihat detailnya.

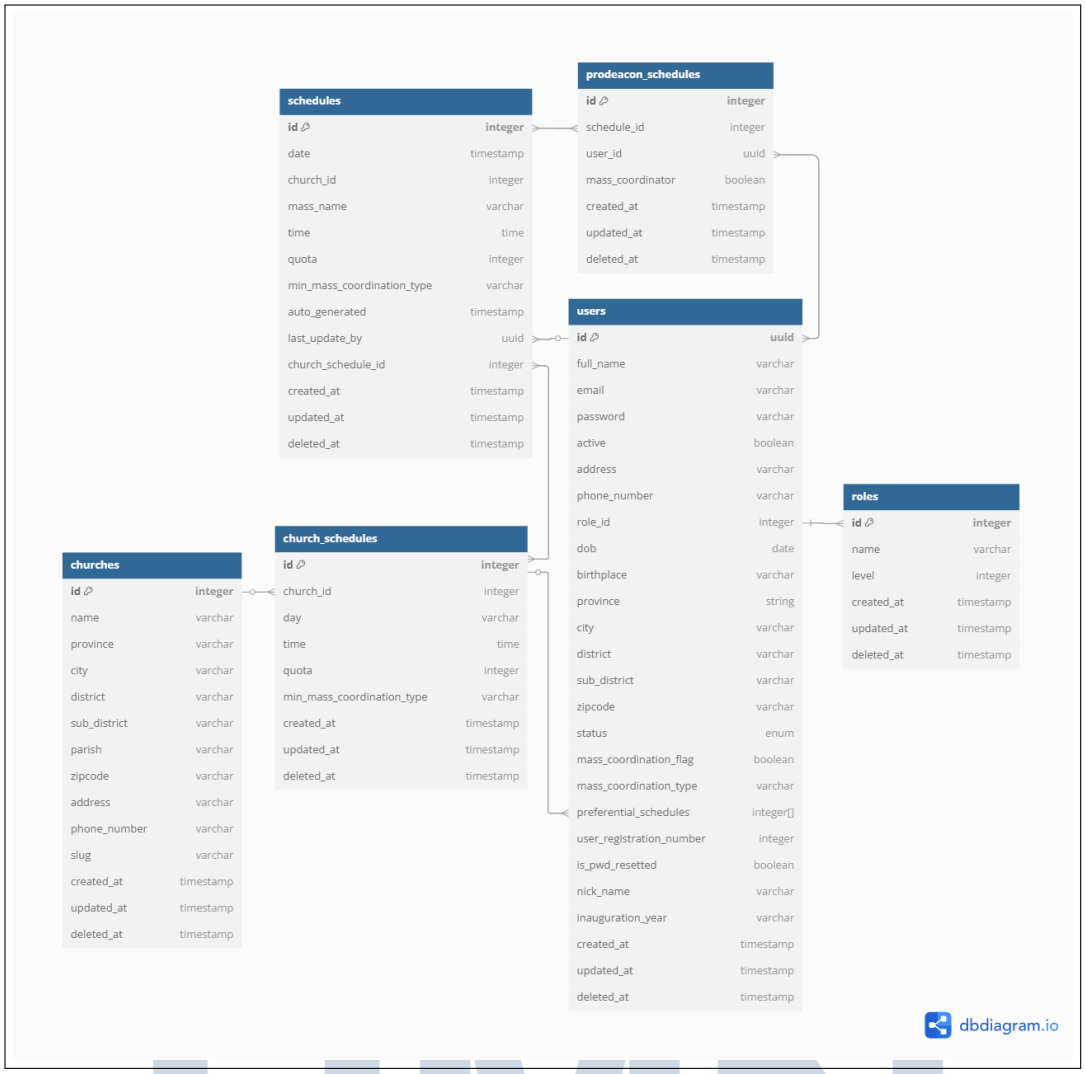


Gambar 3.27. Flowchart *Member Profile*

Gambar 3.27 menggambarkan *flowchart* profil pada tampilan *member* yang diawali dengan mengambil data jadwal dari *database* dan menampilkannya kepada pengguna. Pengguna dapat memilih untuk melakukan *logout* pada tampilan *profile*.

3.3.3 Skema Database

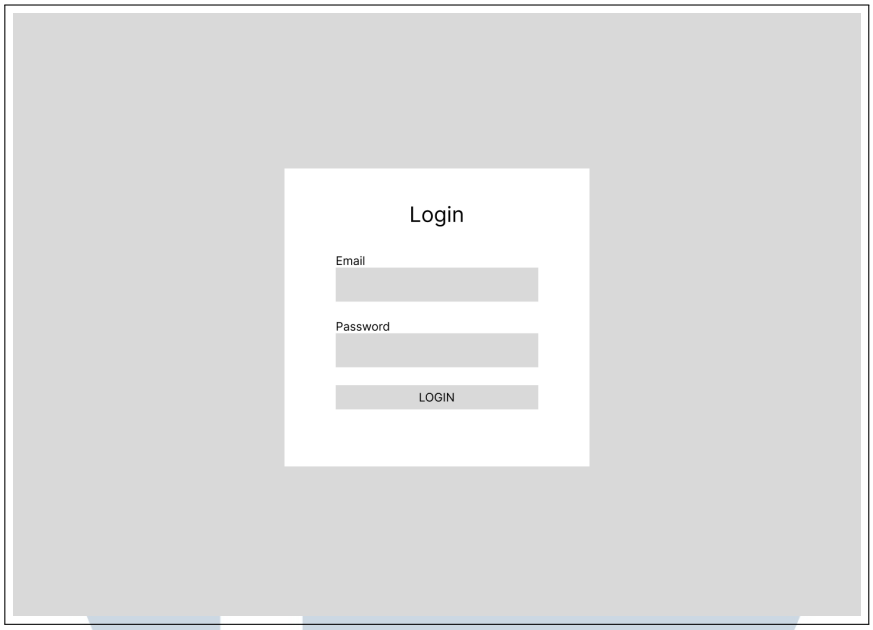
Gambar 3.28 menunjukkan skema *database* yang digunakan, terdapat sejumlah 6 tabel dalam pembuatan aplikasi, yaitu *users*, *churches*, *church_schedules*, *prodeacon_schedules*, *schedules*, dan *roles*.



Gambar 3.28. Skema Database

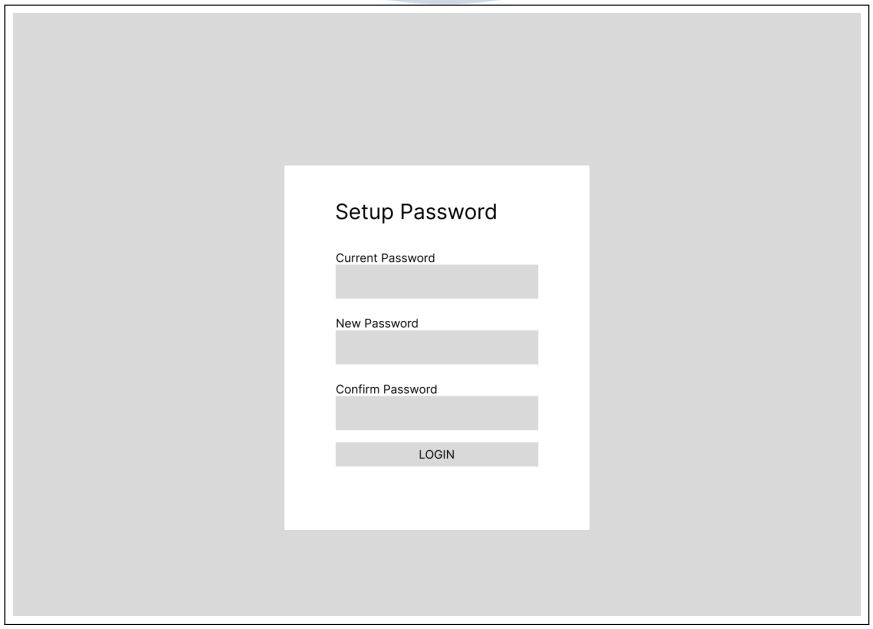
3.3.4 Wireframe

Dalam pengerjaan, terdapat *wireframe* atau desain antar muka yang berfungsi sebagai gambaran terhadap tampilan yang dikerjakan. *Wireframe* ditunjukkan oleh Gambar 3.29 hingga Gambar 3.41.



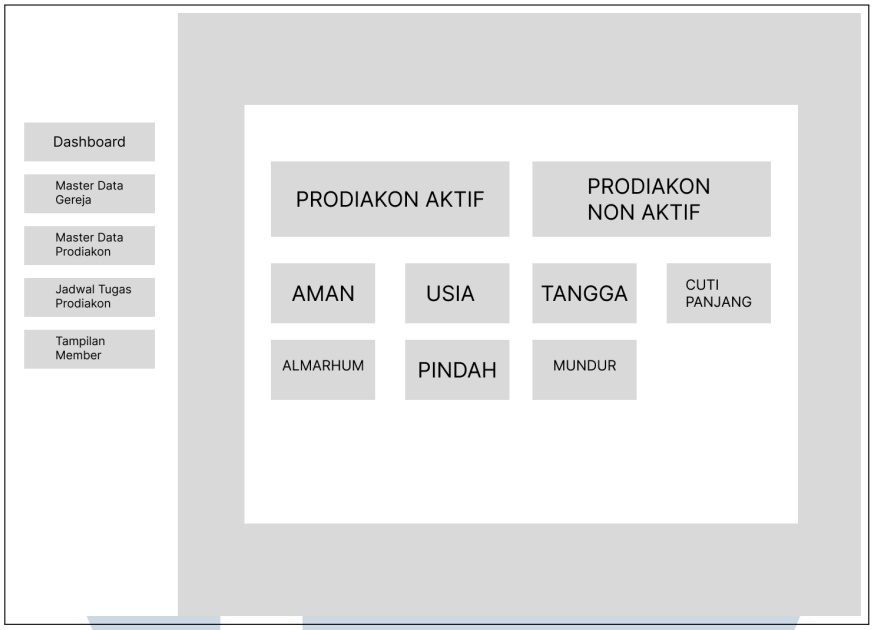
Gambar 3.29. *Wireframe* halaman *login*

Gambar 3.29 menunjukkan *wireframe* halaman *login* aplikasi. Pengguna dapat memasukkan *email* dan *password* untuk *login*.



Gambar 3.30. *Wireframe* halaman *setup password*

Gambar 3.30 menunjukkan *wireframe* halaman *setup password* pada aplikasi. Tampilan ini akan muncul ketika pengguna baru pertama kali *login* dan belum melakukan *setup password*.

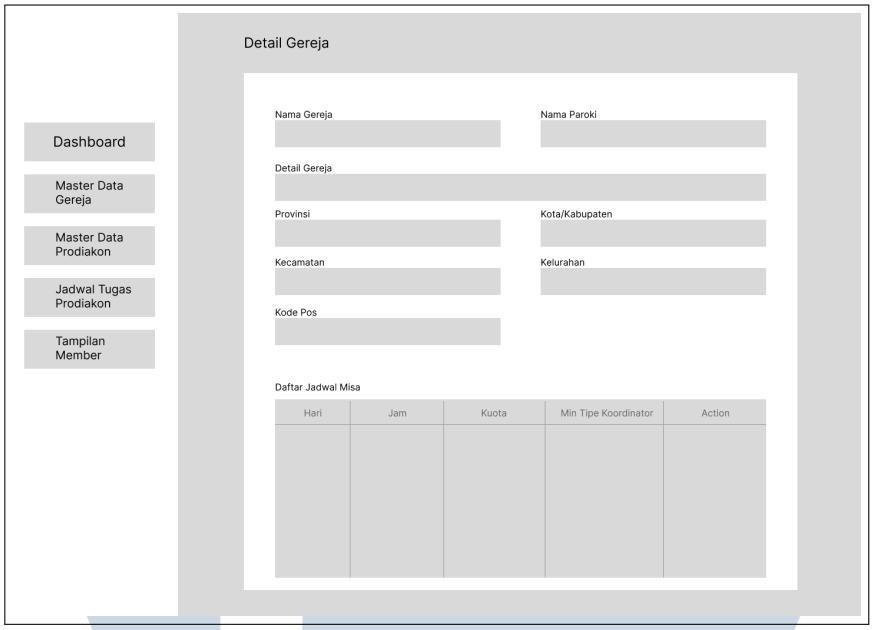


Gambar 3.31. *Wireframe* halaman *dashboard*

Gambar 3.31 menunjukkan *wireframe* halaman *dashboard* admin pada aplikasi. Tampilan ini berisikan jumlah prodiakon yang aktif dan tidak aktif beserta jumlah prodiakon tiap statusnya.

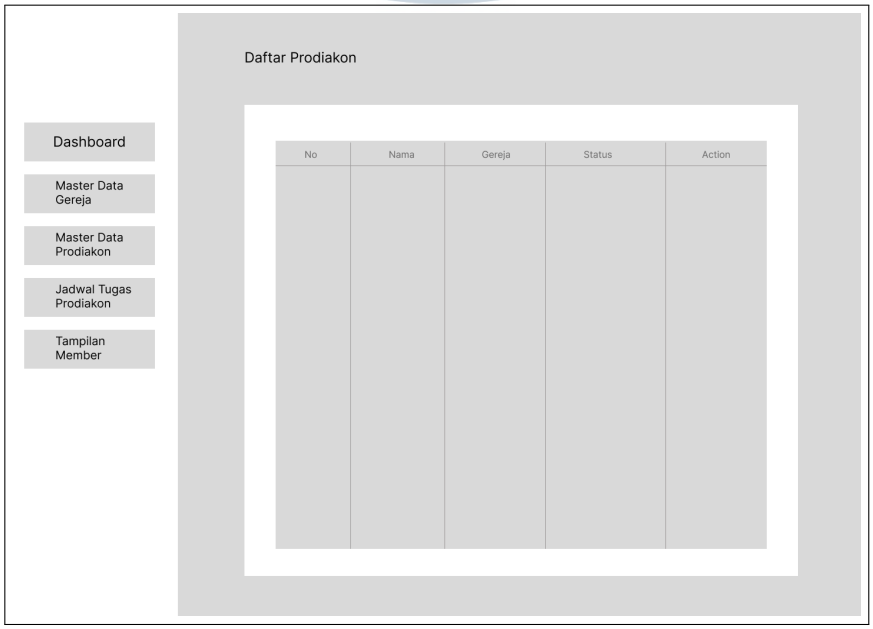
Gambar 3.32. *Wireframe* halaman *master data* gereja

Gambar 3.32 menunjukkan *wireframe* halaman daftar gereja pada aplikasi. Tampilan ini menampilkan gereja yang ada.



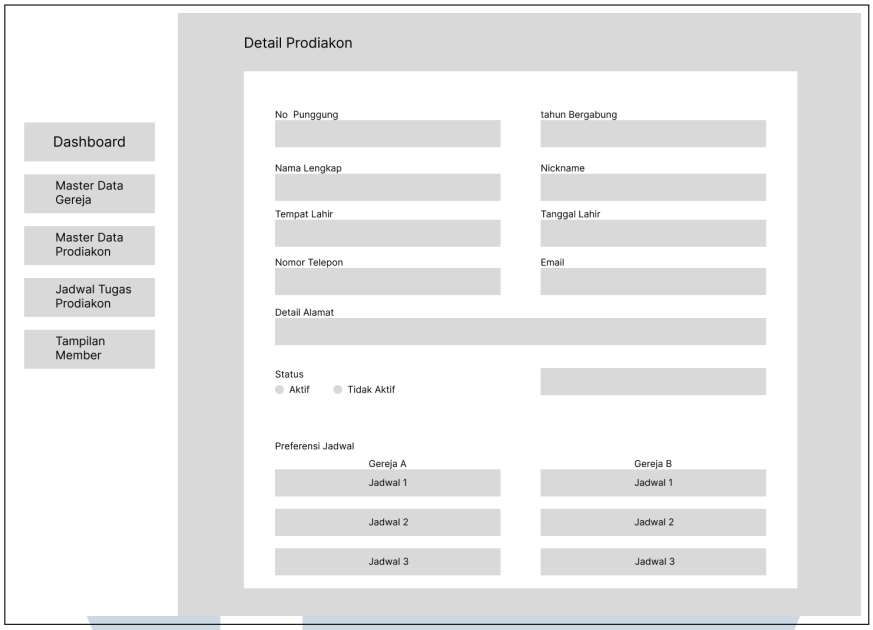
Gambar 3.33. *Wireframe* halaman detail gereja

Gambar 3.33 menunjukkan *wireframe* halaman detail gereja pada aplikasi. Admin dapat mengubah data yang ada pada gereja yang dipilih.



Gambar 3.34. *Wireframe* halaman *master data* prodiakon

Gambar 3.34 menunjukkan *wireframe* halaman daftar prodiakon pada aplikasi. Tampilan ini menampilkan semua prodiakon yang ada. Admin dapat menghapus atau melihat detail dari seorang prodiakon.

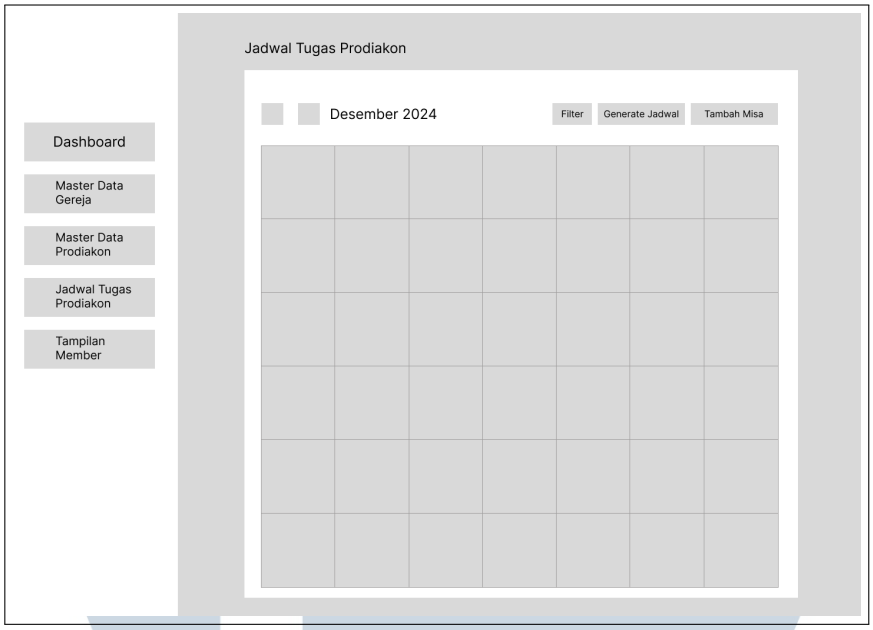


Gambar 3.35. *Wireframe* halaman detail prodiakon

Gambar 3.35 menunjukkan *wireframe* halaman detail prodiakon pada aplikasi. Admin dapat mengubah data prodiakon yang telah dipilih.

Gambar 3.36. *Wireframe* halaman daftar gereja pada pemilihan jadwal

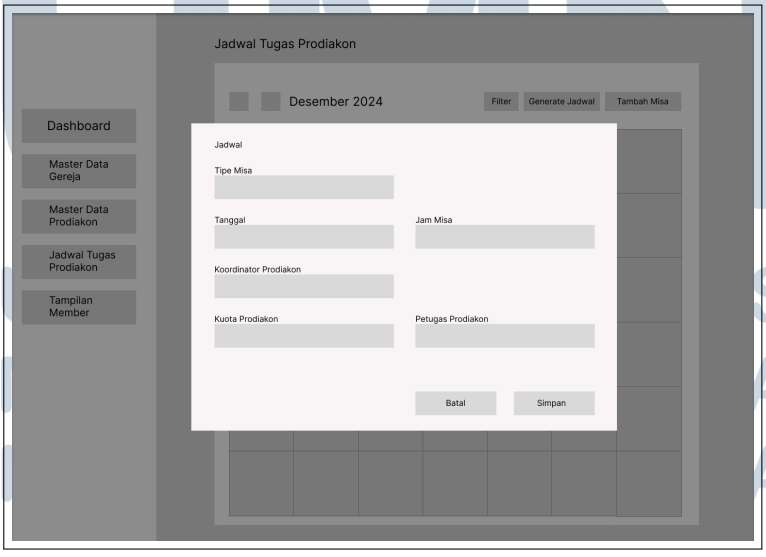
Gambar 3.36 menunjukkan *wireframe* halaman daftar gereja sebelum memasuki halaman jadwal suatu gereja. Tampilan ini menampilkan gereja yang ada dan admin dapat memilih salah satu gereja.



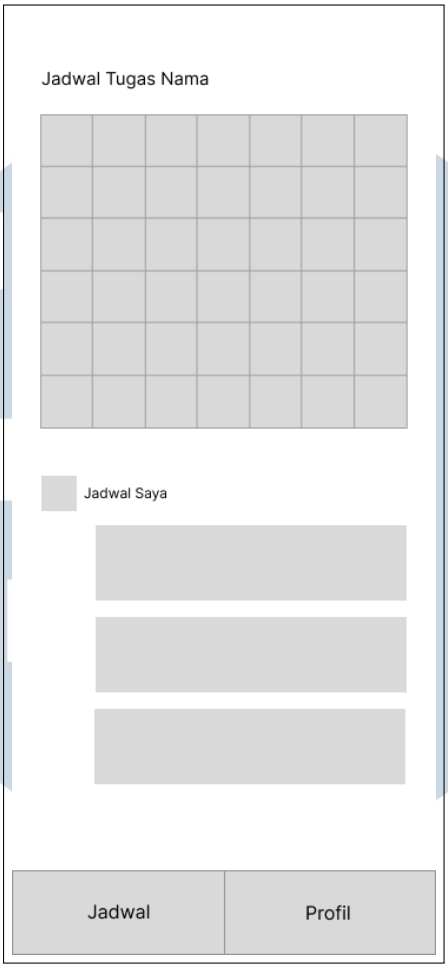
Gambar 3.37. *Wireframe* halaman jadwal gereja

Gambar 3.37 menunjukkan *wireframe* halaman jadwal pada suatu gereja. Admin dapat melihat detail dari masing-masing jadwal, menambahkan jadwal baru, membuat jadwal bulanan, dan melakukan *filtering*.

Gambar 3.38 menunjukkan *wireframe* modal tambah misa maupun *update* misa pada halaman jadwal suatu gereja. Admin dapat mengisi beberapa *form field* yang dibutuhkan.

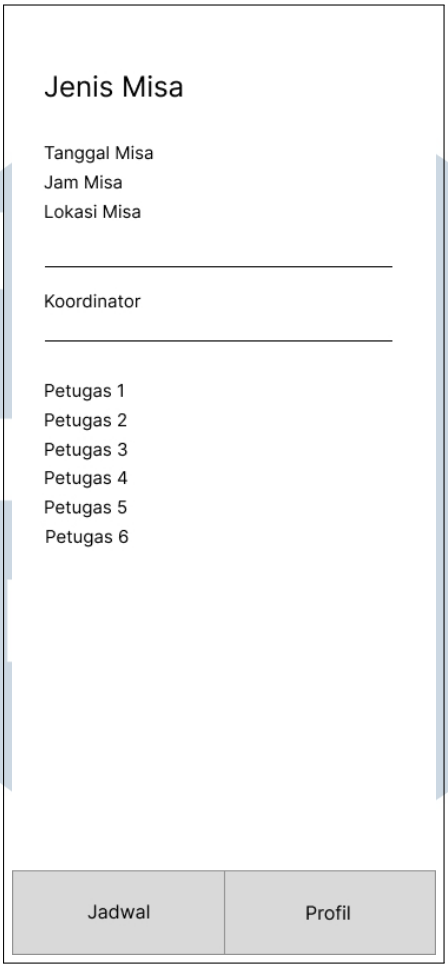


Gambar 3.38. *Wireframe* modal tambah jadwal misa



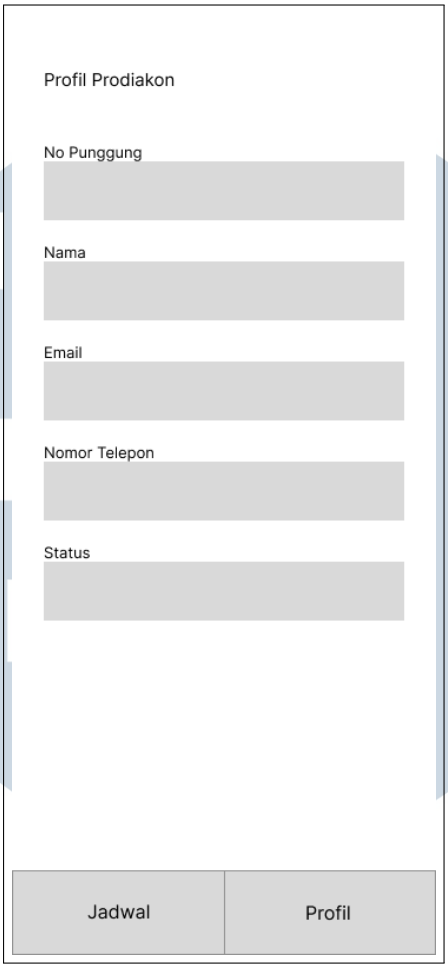
Gambar 3.39. *Wireframe* halaman jadwal (*member*)

Gambar 3.39 menunjukkan *wireframe* halaman jadwal milik seorang prodiakon. Terdapat kalender yang berisikan jadwal misa pada bulan tersebut, dan detail jadwal dalam satu hari pada bagian bawah.



Gambar 3.40. *Wireframe* halaman detail jadwal (*member)*

Gambar 3.40 menunjukkan *wireframe* halaman detail jadwal pada sisi prodiakon. Halaman ini berisikan detail misa, seperti tanggal misa, jam misa, lokasi, serta prodiakon yang bertugas.



Gambar 3.41. *Wireframe* halaman profil (*member*)

Gambar 3.41 menunjukkan *wireframe* halaman profil milik seorang prodiakon. Pada halaman ini berisikan data pribadi milik prodiakon, mulai dari nomor punggung, nama, *email*, nomor telepon, dan status.