# BAB III

# METODE PENELITIAN

## *Requirements Gathering*

Menganalisa kebutuhan adalah tahap yang tidak dapat terpisahkan dari proses pengembangan perangkat lunak, begitu pula penelitian kami dalam merancang aplikasi untuk penyedia jasa kursus mengemudi, hal pertama yang kita lakukan adalah melakukan wawancara dan observasi terhadap tiga penyedia jasa kursus mengemudi di Surabaya untuk mendapatkan pengetahuan tentang proses bisnis dan kendala-kendala yang selama ini dihadapi oleh mereka. Berikut adalah hasil dari proses pengumpulan dan analisa kebutuhan yang kami lakukan.

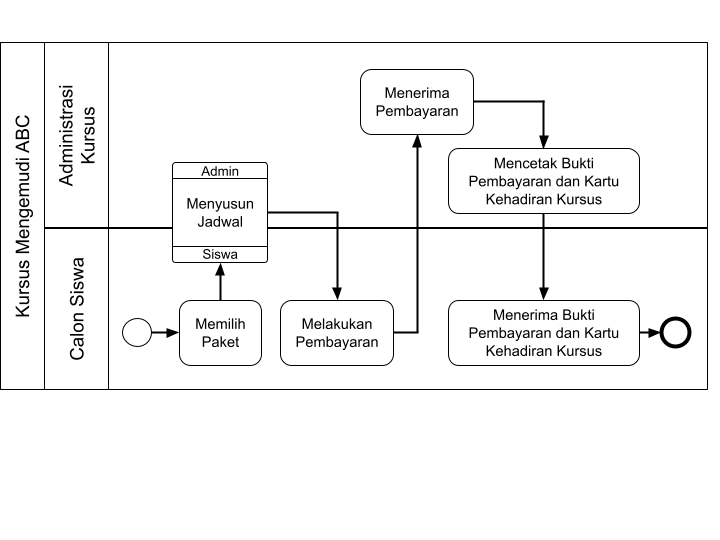
### Proses Bisnis

Seperti yang kita ketahui, berkendara dengan tertib di jalan raya merupakan tanggung jawab kita semua, namun, dalam menjaga ketertiban tersebut dibutuhkan pihak yang memiliki kewenangan untuk mengatur, mengarahkan, bahkan memberikan sanksi apabila terjadi pelanggaran di jalan raya, pihak tersebut tidak lain adalah kepolisian dan dinas perhubungan. Oleh karena itu, para penyedia jasa kursus mengemudi wajib bekerja sama dengan pihak-pihak tersebut untuk mendapatkan izin sebelum melangsungkan bisnisnya. Jika kita mengacu pada Peraturan Daerah Kota Surabaya Nomor 22 Tahun 2012 tentang Izin Penyelenggaraan Pendidikan dan Pelatihan Mengemudi Kendaraan Bermotor, para penyedia jasa kursus mengemudi, baik perorangan maupun lembaga pendidikan, yang ingin menawarkan jasanya kepada masyarakat diharapkan dapat memenuhi persyaratan yang diperlukan.

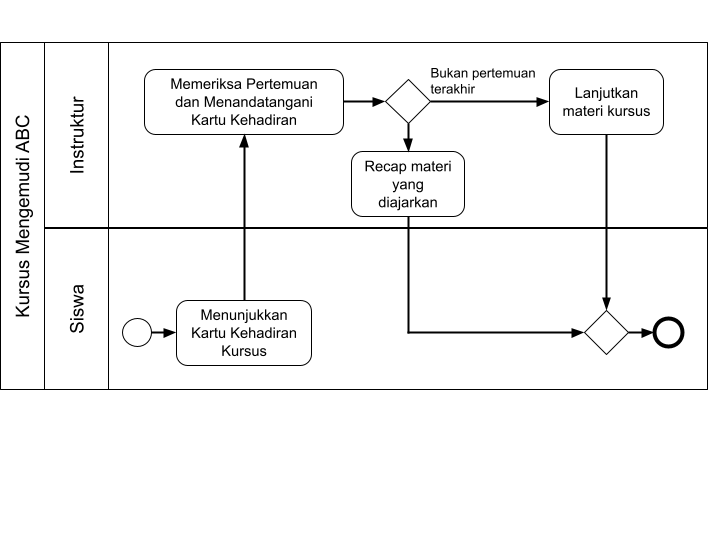
Namun, setelah kami lakukan wawancara dengan tiga penyedia jasa kursus mengemudi, ternyata pemenuhan persyaratan yang dibutuhkan hanya dilakukan sebagian saja. Berikut merupakan hasil analisa pemenuhan persyaratan kami dengan tiga penyedia jasa kursus mengemudi.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Dokumen Persyaratan** | **Kursus ABC** | **Sie Bersaudara** | **“Hafiz”** |
| 1 | Akte pendirian perusahaan (Lembaga Kursus) atau KTP (Perorangan) | ✔ | ✔ | ✔ |
| 2 | Struktur organisasi penyelenggara pendidikan mengemudi |  | ✔ |  |
| 3 | Daftar nama personil, riwayat hidup pengelola dan instruktur & Sertifikat Instruktur | ✔ | ✔ | ✔ |
| 4 | Tata tertib penyelenggaraan pendidikan dan pelatihan mengemudi kendaraan bermotor | ✔ | ✔ |  |
| 5 | Rekomendasi penyelenggaraan kursus mengemudi dari Satlantas Polrestabes Surabaya |  |  |  |
| 6 | Kurikulum pendidikan dan pelatihan mengemudi kendaraan bermotor |  |  | ✔ |

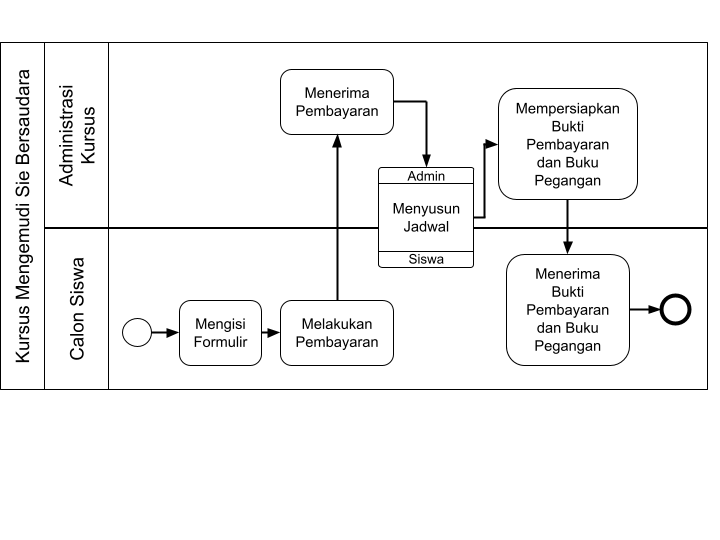
Pada saat kami melakukan penelitian ini belum ada standar atau peraturan yang mengatur atau mewajibkan para penyedia jasa kursus mengemudi untuk memiliki suatu proses tertentu. Sehingga, langkah selanjutnya adalah kami menanyakan bagaimana proses bisnis yang ada pada ketiga kursus mengemudi tersebut. Agar lebih mudah dipahami, kami bagi proses bisnis menjadi 2 alur, Pendaftaran Kursus dan Proses Kursus Mengemudi.



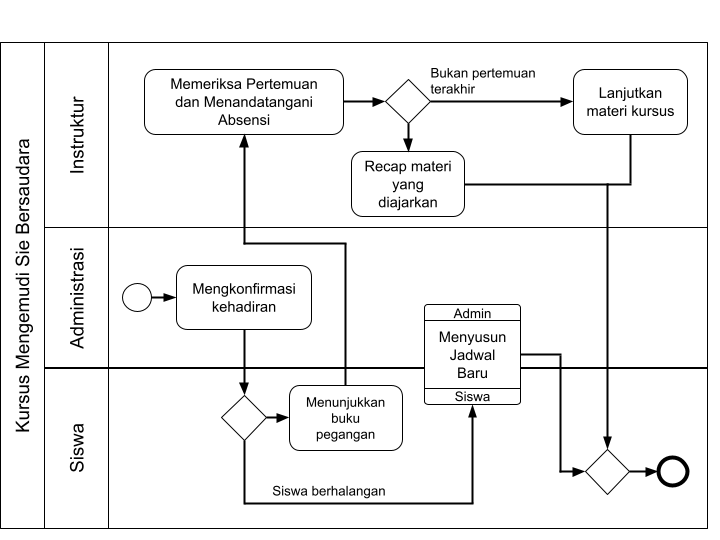
Untuk proses pendaftaran pada Kursus ABC dimulai dengan calon siswa memilih kelas kursus kemudian menyusun jadwal, jika calon siswa merasa sudah puas proses selanjutnya adalah pembayaran, dimana nominal uang yang dibayarkan harus diperiksa terlebih dahulu oleh pihak administrasi Kursus ABC, jika sudah benar, pihak administrasi Kursus ABC memberikan tanda terima pembayaran ke calon siswa beserta kartu kehadiran kursus yang nantinya wajib dibawa setiap kursus.



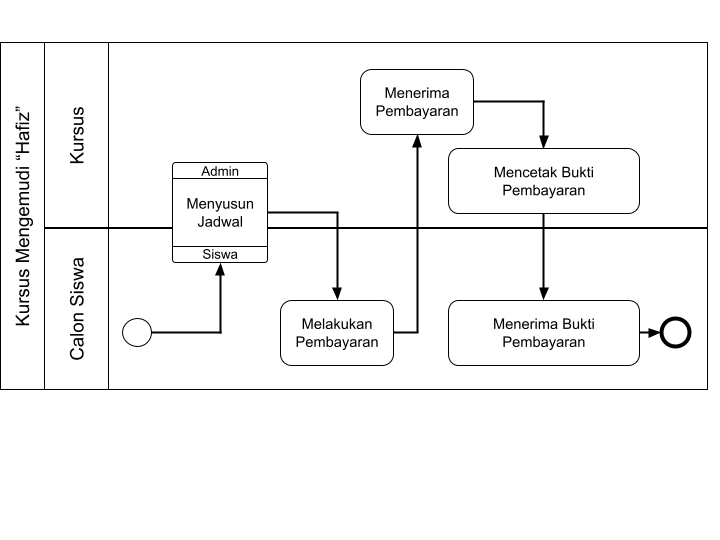
Proses kursus di kursus mengemudi ABC dimulai dengan siswa hadir ke tempat kursus atau siswa bisa memilih untuk dijemput oleh instruktur kursus di rumahnya. Sebelum memulai praktik, siswa menunjukkan kartu kehadiran kursus ke instruktur untuk ditandatangani kehadirannya bersamaan dengan instruktur memeriksa siswa saat ini berada di pertemuan ke berapa, dimana siswa selanjutnya mempelajari materi sesuai dengan pertemuan saat ini.



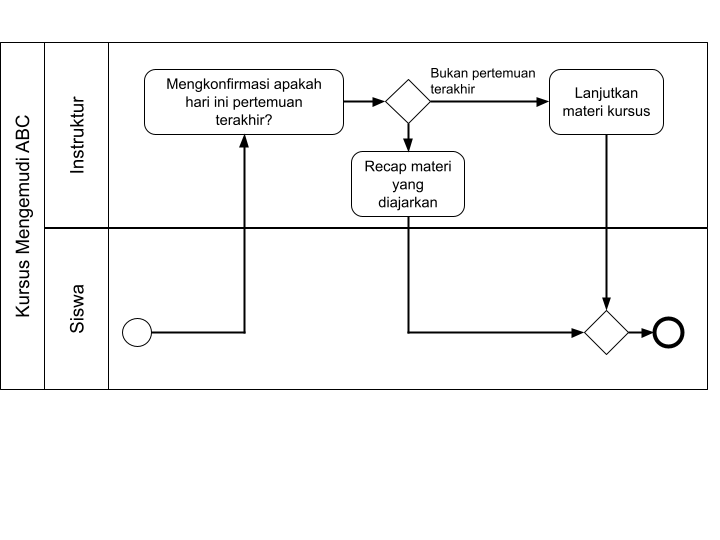
BPMN di atas menunjukkan alur proses pendaftaran di Kursus Sie Bersaudara, dimulai dengan siswa mengisi formulir, di dalam formulir tersebut terdapat opsi kelas dan deskripsi dari masing-masing kelas, setelah mengisi formulir, siswa melakukan pembayaran, sama dengan kursus ABC, pihak administrasi kursus Sie Bersaudara memeriksa apakah nominal yang dibayarkan sudah benar, jika sudah, selanjutnya siswa memilih jadwal kursus. Kemudian, pihak administrasi memberikan bukti pembayaran dan buku pegangan yang berisi materi yang nantinya akan diajarkan serta tertera lembar absensi di dalamnya.



Untuk proses kursus di kursus Sie Bersaudara dimulai dengan pihak administrasi mengkonfirmasi kehadiran siswa 1-2 jam sebelum jadwal kursus, jika siswa ternyata tidak bisa hadir, pihak administrasi akan menawarkan jadwal baru ke siswa yang tidak hadir tersebut, jika siswa hadir, selanjutnya siswa akan bertemu dengan instruktur, siswa harus menunjukkan buku pegangannya untuk ditandatangani lembar absensinya. Sama dengan kursus mengemudi ABC, instruktur memeriksa di pertemuan ke berapa siswa saat ini, apabila siswa ada di pertemuan terakhir, instruktur nantinya akan mengulang materi dari awal hingga pertemuan sebelumnya, kemudian, sebelum siswa pulang, instruktur wajib memfoto siswa didepan mobil kursus untuk selanjutnya dicetak menjadi sertifikat seandainya siswa ingin mendapatkan sertifikat kursus.



Berbeda dengan dua kursus sebelumnya, kursus mengemudi “Hafiz” tergolong kursus mengemudi perorangan. Proses pendaftaran dimulai dengan siswa langsung memilih jadwal, kemudian siswa melakukan pembayaran, dan yang terakhir pihak kursus mencetak bukti pembayaran untuk diberikan ke siswa.



Hampir sama dengan proses kursus sebelumnya, dimulai dengan siswa hadir ke tempat kursus atau dijemput oleh instruktur, selanjutnya instruktur menanyakan apakah pertemuan saat ini pertemuan terakhir, jika tidak, instruktur dan siswa bisa melanjutkan materi dari pertemuan sebelumnya, jika iya, instruktur bersama siswa akan mempraktekkan materi kursus dari awal hingga pertemuan sebelumnya.

### Analisa Kebutuhan

Dalam mengembangkan perangkat lunak, analisa kebutuhan merupakan hasil riset dengan beberapa pihak dan bidang yang berbeda, seperti dari bidang bisnis diantaranya adalah situasi pasar, pengguna akhir, calon konsumen, dan peluang-peluang teknis. Sedangkan dari perspektif desain dan *prototype*, analisa kebutuhan dapat dibagi menjadi 4 aspek: bisnis/pemasaran, fungsional/fitur, teknologi, dan kegunaan.

Kebutuhan bisnis/pemasaran memberikan pemahaman tentang kebutuhan bisnis atau kondisi *marketplace* yang ada. Kebutuhan fungsional memberikan pemahaman akan fitur-fitur yang dibutuhkan untuk mendukung kebutuhan bisnis atau pemasaran. Kebutuhan fungsional juga terkadang merupakan hasil dari hasil riset pengguna dan pengujian kemudahan penggunaan. Selanjutnya spesifikasi fitur ini nantinya akan diterapkan pada *prototype*. Kebutuhan teknologi memberikan pemahaman tentang teknologi yang dibutuhkan untuk mengimplementasikan fitur yang diperlukan. Sedangkan kebutuhan kegunaan memberikan pemahaman tentang *User Experience* dan syarat-syarat kegunaan yang dibutuhkan untuk memudahkan pengguna dalam mengadopsi perangkat lunak yang baru nantinya.

|  |  |  |
| --- | --- | --- |
| **No.** | **Kebutuhan / Persyaratan** | **Aspek** |
| Sisi Admin Kursus (Lembaga) / Pemilik (Perorangan & Lembaga) | | |
| 1 | Fungsi *Login* & *Logout* sebagai admin / pemilik | Fungsional |
| 2 | Dashboard yang berisi jadwal kursus setiap Instruktur per hari | Fungsional |
| 3 | Menambah kelas kursus beserta deskripsi, harga, jumlah pertemuan, benefit, dan kategori kelas kursus | Fungsional |
| 4 | Mengubah nama, deskripsi, harga, jumlah | Fungsional |

|  |  |  |
| --- | --- | --- |
| **No.** | **Kebutuhan / Persyaratan** | **Aspek** |
|  | pertemuan, benefit, dan kategori kelas kursus |  |
| 5 | Menghapus / menonaktifkan kelas kursus | Fungsional |
| 6 | Menampilkan detail *progress* setiap siswa | Fungsional |
| 7 | Mengajukan perubahan jadwal kursus kepada siswa dan instruktur | Kegunaan |
| 8 | Menampilkan hasil pengisian formulir pendaftaran kursus | Fungsional |
| 9 | Menghubungi siswa dan instruktur | Fungsional |
| 10 | Mengkonfirmasi status pembayaran kursus dari siswa | Bisnis/Pemasaran |
| Sisi Instruktur (Lembaga) | | |
| 1 | Fungsi *Login* & *Logout* sebagai Instruktur | Fungsional |
| 2 | Dashboard yang berisi jadwal kursus per hari | Fungsional |
| 3 | Menampilkan daftar siswa aktif | Fungsional |
| 4 | Menampilkan detail *progress* siswa | Fungsional |
| 5 | Mengajukan atau mengkonfirmasi perubahan jadwal kursus kepada siswa | Kegunaan |
| 6 | Menghubungi siswa | Fungsional |

|  |  |  |
| --- | --- | --- |
| **No.** | **Kebutuhan / Persyaratan** | **Aspek** |
| Sisi *General User* | | |
| 1 | Fungsi *Login* & *Logout* sebagai *General User* | Fungsional |
| 2 | Dashboard yang berisi rekomendasi kelas kursus dan *progress* kursus yang sedang aktif | Fungsional |
| 3 | Mendaftarkan diri ke kursus mengemudi | Fungsional |
| 4 | Menghubungi admin / pemilik kursus dan instruktur | Fungsional |
| 5 | Menampilkan detail *progress* kursus yang berlangsung | Fungsional |
| 6 | Mengajukan perubahan jadwal kursus kepada instruktur | Kegunaan |
| 7 | Menampilkan hasil pengisian formulir pendaftaran kursus | Fungsional |
| 8 | Menjadi pemilik kursus dengan mengirimkan surat izin penyelenggaraan kursus mengemudi | Fungsional |
| Kebutuhan Teknis | | |
| 1 | Sistem mampu dijalankan melalui Smartphone semua pihak | Teknologi |
| 2 | Sistem mudah diakses di web/internet | Teknologi |

## Iterasi Pertama

Selama rangkaian proses *prototyping* yang kami lakukan, terjadi dua kali iterasi dimana setelah kami melakukan pengujian awal dengan pengguna, kami harus kembali lagi ke proses desain singkat, untuk menyisipkan saran dan masukan yang kami terima pada proses pengujian.

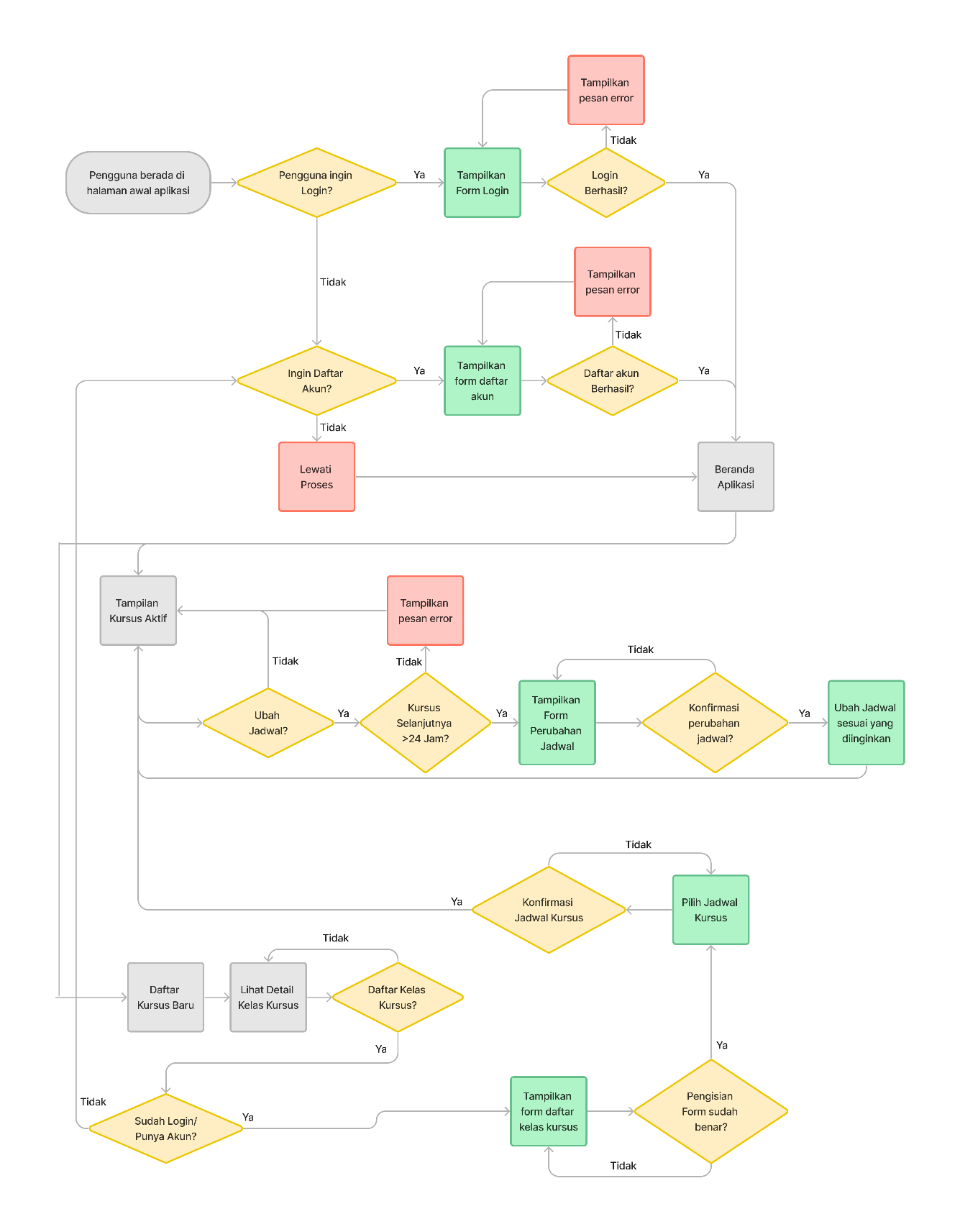
### *Quick Design*

Dari analisa kebutuhan diatas, kami mulai memahami kebutuhan dan kendala-kendala yang dihadapi oleh pihak penyedia jasa kursus mengemudi. Selanjutnya, pada tahap ini, kami mulai dengan membuat rangkaian alur pengguna atau sering dikenal dengan *user flow*, alur ini nantinya akan menjadi basis bagi kami untuk membuat sketsa awal aplikasi.

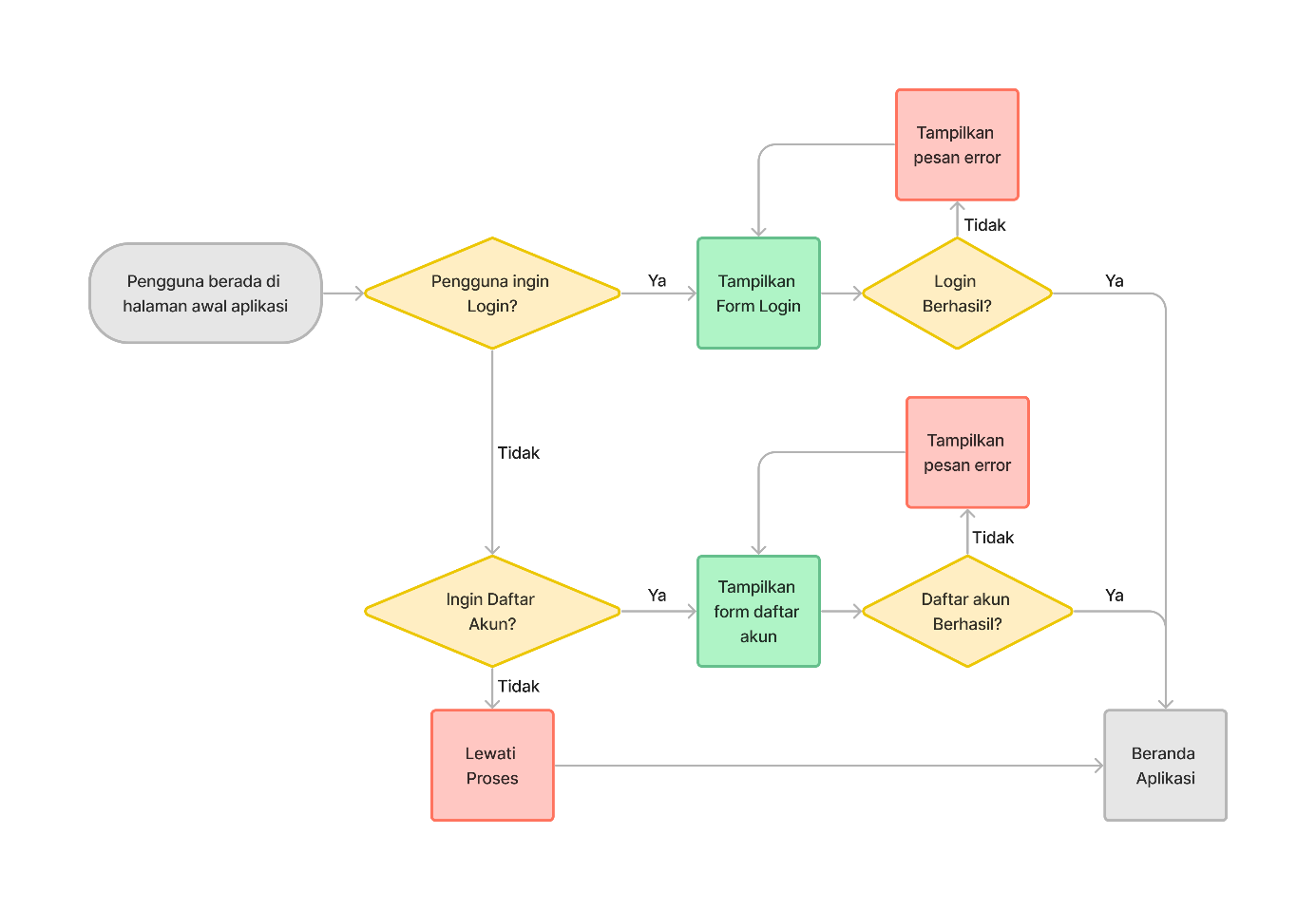
#### Alur Pengguna

Alur pengguna bertujuan tidak hanya membantu tim pengembang dalam menghasilkan *prototype*, pihak lain yang merasakan manfaat dari alur pengguna ini adalah pengguna itu sendiri, dalam hal ini adalah pihak penyedia jasa kursus mengemudi. Dengan mengetahui alur pengguna, pihak penyedia jasa kursus mengemudi dapat mengevaluasi apakah alur pengguna yang dirancang oleh tim pengembang sudah sesuai dengan proses bisnis yang saat ini sedang berjalan.

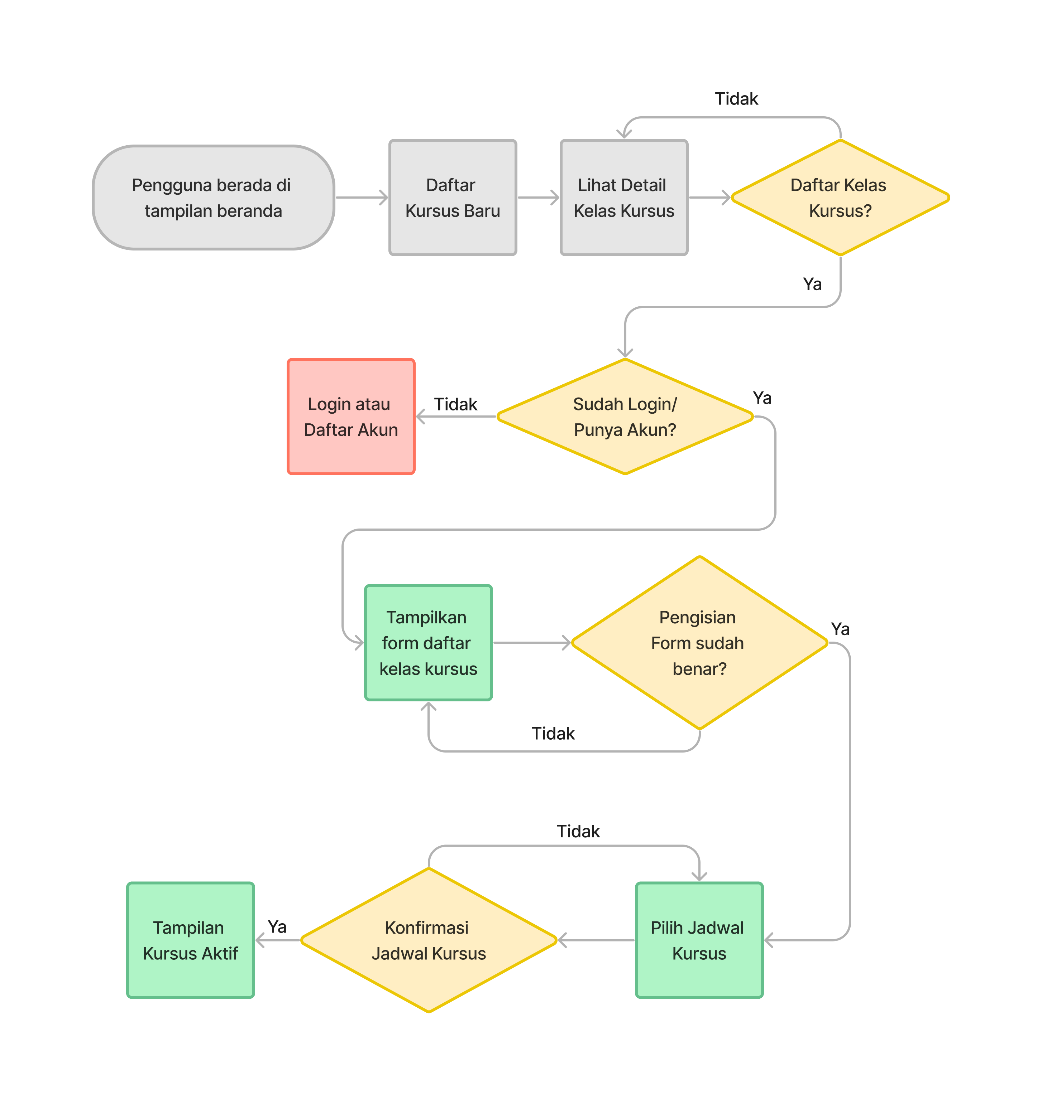
Alur yang menjadi fokus kami untuk saat ini adalah alur yang sama ketika seorang calon siswa / peserta kursus akan mendaftar untuk mengikuti kelas kursus mengemudi, yang sebelumnya sudah kami lakukan wawancara dan observasi bersama dengan pihak penyedia jasa kursus mengemudi pada sub bab sebelumnya. Berikut adalah alur pengguna untuk *prototype* iterasi pertama kami.



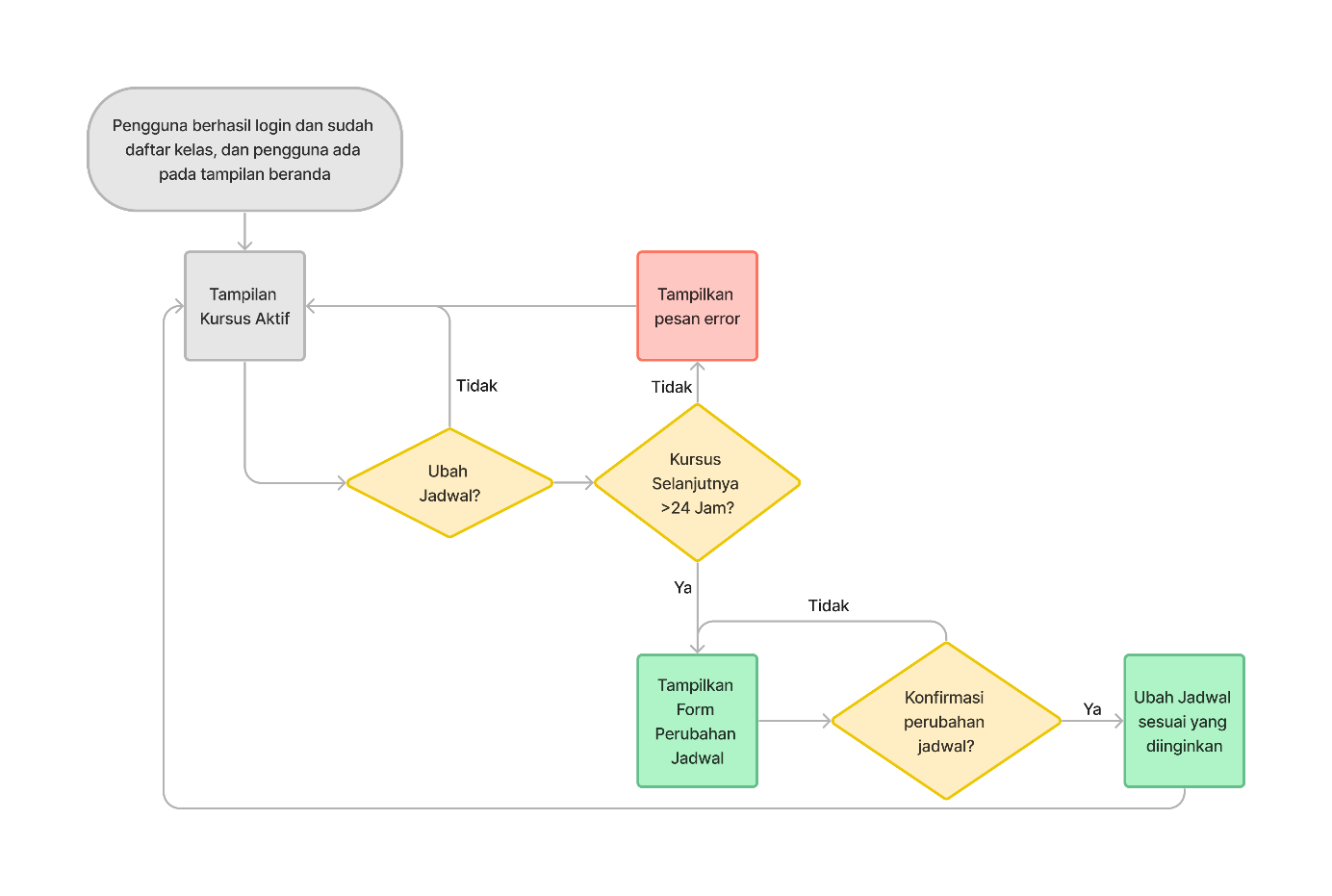
Untuk menyederhanakan alur pengguna yang kompleks diatas, kami akan membagi alur pengguna menjadi alur tugas (*task flow*), alur tugas untuk *login* atau mendaftarkan akun, alur tugas untuk melakukan pendaftaran kelas kursus, dan alur tugas untuk mengubah jadwal kursus. Dibawah ini adalah hasil penyederhanaan alur pengguna diatas.



Alur tugas dimulai dengan pengguna berada di tampilan awal aplikasi. Kemudian pengguna dapat memilih ingin *login* atau daftar akun baru, atau opsi selanjutnya adalah memilih untuk melakukan proses ini nanti apabila memang dibutuhkan. Apabila pengguna memilih untuk *login*, selanjutnya tampilan yang muncul adalah *form* *login*. Jika, pengguna berhasil *login*, tampilan selanjutnya adalah beranda aplikasi, namun, jika pengguna gagal, maka akan tampil pesan *error* pada *form login*. Sedangkan jika pengguna memilih untuk mendaftarkan akun baru, tampilan yang muncul selanjutnya adalah tampilan *form* daftar akun. Serupa dengan alur *login*, apabila pengguna berhasil melakukan pendaftaran akun, pengguna akan melihat tampilan beranda, namun, jika pengguna gagal, akan timbul pesan *error* pada form pendaftaran akun.



Alur dimulai dengan pengguna berada di tampilan beranda, pengguna memilih kelas kursus yang ingin diikuti. Ketika pengguna memilih kelas kursus yang akan diikuti, tampilan selanjutnya adalah detail kelas kursus, jika pengguna memilih untuk mendaftarkan diri ke kelas tersebut, sistem akan memeriksa ulang apakah pengguna sudah melakukan *login* atau daftar akun pada alur tugas sebelumnya. Jika belum, tampilan berikutnya adalah tampilan awal aplikasi, jika pengguna sudah melakukan *login* atau daftar akun, tampilan selanjutnya adalah *form* pendaftaran kelas kursus. Jika *form* sudah diisi dengan benar, tampilan berikutnya adalah tampilan untuk memilih jadwal kursus, jika *form* tidak diisi dengan lengkap, maka akan dikembalikan lagi ke tampilan *form* pendaftaran kursus. Jika pengguna puas dengan jadwal yang dipilih, pengguna akan dialihkan ke tampilan kursus aktif, jika pengguna merasa tidak puas, pengguna dapat mengubah ulang jadwal kursus.



Alur tugas dapat dimulai apabila pengguna memilih untuk melakukan *login* atau daftar akun sebelumnya, kemudian sudah pernah mendaftar kursus dan berada di tampilan beranda. Selanjutnya, pengguna dapat beralih ke tampilan kursus aktif. Jika pengguna memilih untuk mengubah jadwal kursus, sistem akan memeriksa apakah pertemuan kursus selanjutnya lebih dari 24 jam? Jika kurang dari 24 jam, akan muncul pesan *error* pada tampilan kursus aktif. Jika lebih dari 24 jam, pengguna akan ditampilkan oleh *form* perubahan jadwal. Jika pengguna mengkonfirmasi perubahan, selanjutnya jadwal baru akan ditampilkan pada tampilan kursus aktif, jika pengguna belum yakin dengan jadwal baru, pengguna akan diarahkan kembali ke *form* perubahan jadwal.

#### Sketsa Awal Aplikasi

Berikut adalah daftar sketsa awal aplikasi yang sudah kami desain untuk iterasi pertama.

|  |  |  |
| --- | --- | --- |
| **No.** | **Sketsa Aplikasi** | **Keterangan** |
| 1 |  | Sketsa awal untuk tampilan depan aplikasi. Selanjutnya, peserta kursus dapat melakukan dua aksi, yaitu, *Login* / Daftar akun atau langsung masuk ke beranda aplikasi |

|  |  |  |
| --- | --- | --- |
| **No.** | **Sketsa Aplikasi** | **Keterangan** |
| 2 |  | Sketsa awal *form* pendaftaran akun, dimana peserta kursus dapat membuat akun baru sebelum menggunakan aplikasi |
| 3 |  | Sketsa awal beranda aplikasi yang menampilkan kursus aktif, bagian tersebut akan muncul apabila peserta sudah pernah mendaftarkan diri ke sebuah kelas kursus dan berisi rekomendasi-rekomendasi kelas kursus yang lain |

|  |  |  |
| --- | --- | --- |
| **No.** | **Sketsa Aplikasi** | **Keterangan** |
| 4 |  | Sketsa awal detail kelas kursus, pada tampilan ini, peserta kursus nantinya mampu mendapatkan informasi singkat tentang kelas kursus sebagai pertimbangan |
| 5 |  | Sketsa *form* pendaftaran kelas kursus sebagai pengganti formulir pendaftaran yang untuk saat ini menggunakan media kertas yang membutuhkan proses pengarsipan lebih lanjut |

|  |  |  |
| --- | --- | --- |
| **No.** | **Sketsa Aplikasi** | **Keterangan** |
| 6 |  | Sketsa awal pemilihan atau peubahan jadwal kursus, dimana peserta kursus dapat melakukan pemilihan jadwal untuk setiap pertemuan |
| 7 |  | Sketsa awal detail kursus aktif, dimana peserta kursus dapat memperoleh informasi-informasi terkait kursus yang diikuti |

### *Built* *Prototype*

Hasil dari desain singkat sebelumnya akan dikonversi dan ditambahkan detail-detail pendukung didalamnya. Dengan dibangunnya *prototype* pertama ini, fitur-fitur aplikasi mulai dapat dievaluasi kemudahan dan efektifitas penggunaannya. Berikut adalah hasil proses *prototyping* untuk iterasi pertama.

|  |  |  |
| --- | --- | --- |
| **No.** | **Desain *Prototype*** | **Keterangan** |
| 1 |  | Tampilan depan aplikasi kami tambahkan gambar pendukung dan membedakan warna tombol untuk memberikan kesan hierarki yang lebih jelas |

|  |  |  |
| --- | --- | --- |
| **No.** | **Sketsa Aplikasi** | **Keterangan** |
| 2 |  | Tampilan *form* pendaftaran akun, ikon mata untuk memberi tahu kan peserta kursus bahwa *password* dapat disembunyikan/diperlihatkan |
| 3 |  | Tampilan beranda aplikasi, penambahan warna dan ikon-ikon pendukung untuk memudahkan pencernaan informasi |

|  |  |  |
| --- | --- | --- |
| **No.** | **Sketsa Aplikasi** | **Keterangan** |
| 4 |  | Tampilan detail kelas kursus, sama seperti sebelumnya, penambahan gambar, ikon, dan pemberian warna tombol untuk memberikan kontras agar memudahkan peserta untuk mengidentifikasi keberadaan tombol |
| 5 |  | Tampilan *form* pendaftaran kursus sesuai dengan pemilihan warna sebelumnya |

|  |  |  |
| --- | --- | --- |
| **No.** | **Sketsa Aplikasi** | **Keterangan** |
| 6 |  | Tampilan pemilihan jadwal, pemberian warna, ikon, dan pemilihan tipografi untuk memberikan kontras dan hierarki yang lebih jelas |
| 7 |  | Tampilan detail kursus aktif, menu-menu diberikan gambar ilustratif untuk mengurangi beban kognitif peserta kursus |

### *Customer Evaluation*

Pengujian awal kami lakukan dengan menunjukkan hasil desain *prototype* sesuai dengan alur pengguna yang sudah dirancang sebelumnya. Selanjutnya, pengguna kami berikan kesempatan untuk mencoba *prototype* yang kami buat. Selain itu, kami mencatat semua komentar, masukan, dan pertanyaan yang diajukan oleh peserta pengguna *prototype* pada proses ini. Komentar, masukan, dan pertanyaan tersebut nantinya akan kami gunakan untuk menyesuaikan alur *prototype* pada iterasi selanjutnya dan meningkatkan kualitas *prototype*. Berikut adalah daftar peserta pengujian dan komentar/masukan yang diberikan oleh peserta pengujian.

|  |  |  |
| --- | --- | --- |
| **No.** | **Aspek *Prototype*** | **Masukan/Saran** |
| Pihak Kursus ABC | | |
| 1 | - | - |
| Pihak Kursus Mengemudi Sie Bersaudara | | |
| 1 | Alur Tugas Pendaftaran Kelas Kursus | Tambahkan status pembayaran (Lunas, Diproses, dll.) |
| Pihak Kursus Mengemudi “HAFIZ” | | |
| 1 | Alur Tugas Pendaftaran Kelas Kursus | Tambahkan *prototype* yang menampilkan daftar kelas kursus yang ditawarkan oleh pihak kursus |
| Pengguna diluar pihak kursus | | |
| 1 | Tampilan *form* pendaftaran akun | * Tambahkan jarak antara *subheading* dan *heading* |
| **No.** | **Aspek *Prototype*** | **Masukan/Saran** |
| Pengguna diluar pihak kursus | | |
|  |  | * Teks pada tombol dibuat lebih tebal * Teks pengisi sementara pada kolom pengisian dibuat lebih gelap * Label kolom pengisian dibuat lebih tebal |
| 2 | Tampilan beranda aplikasi | * Kurangi jarak untuk *Heading* dibagian atas beranda * Teks untuk tombol “Lihat Semua” dibuat lebih tebal lagi * Warna hitam sekunder di tulisan “- Pemula” dan “Pertemuan ke-3” dibuat lebih muda lagi * Kolom pencarian dipindah ke atas ke bagian *navbar* |
| 3 | Tampilan detail kelas kursus | * Teks deskripsi kelas dibuat abu-abu * Gambar pendukung dibuat lebih besar * Garis luar komponen |

|  |  |  |
| --- | --- | --- |
| **No.** | **Aspek *Prototype*** | **Masukan/Saran** |
| Pengguna diluar pihak kursus | | |
|  |  | bagian “Fasilitas & Keuntungan” ubah warnanya jadi lebih gelap   * Teks pengisi sementara pada kolom pengisian dibuat lebih gelap * Label kolom pengisian dibuat lebih tebal * Teks untuk harga dibuat lebih tebal |
| 4 | Tampilan *form* pendaftaran kelas kursus | * Tambah jarak antara bagian *form* dan *subheading* * Label diatas kolom pengisian dibuat lebih tebal |
| 5 | Tampilan pemilihan jadwal | * Teks pada *Tab* untuk pertemuan yang tidak terpilih diberikan warna yang lebih muda |
| 6 | Tampilan detail kursus aktif | * Tombol untuk “Jadwal Kursus” kurang tinggi * Ikon untuk kembali disamakan dengan tampilan detail kursus |

### *Review & Updation*

Tahapan ini berfokus untuk memmenyesuaikan antarmuka *prototype* sesuai dengan keinginan pengguna *prototype*, serta mengakomodasi perubahan yang disarankan pengguna *prototype* pada tahap pengujian sebelumnya, jika pengguna merasa kurang senang dengan desain yang kami tawarkan. Semua masukan atau komentar terkait warna, tipografi, ukuran tombol, navigasi, dan semua hal yang terkait dengan antarmuka pengguna akan diperbaiki pada tahapan ini. Peningkatan kualitas *prototype* inilah yang menghantarkan kita ke iterasi kedua atau dalam projek pengembangan ini merupakan iterasi terakhir.

## Iterasi Kedua

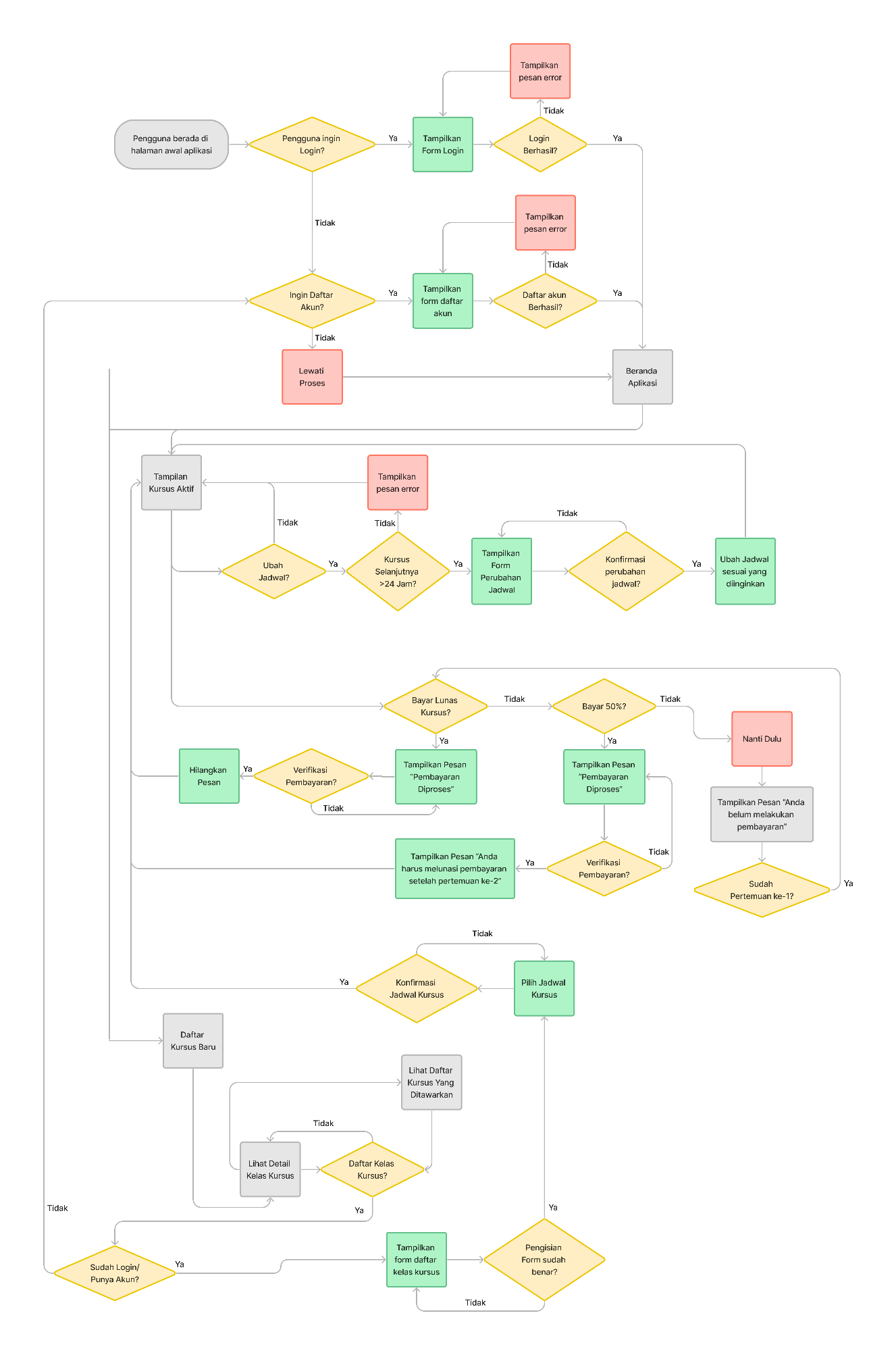
Pada iterasi kedua perbaikan dan peningkatan kualitas *prototype*, kami mengakomodasi semua masukan yang didapatkan pada iterasi sebelumnya. Berikut adalah hasil dari iterasi kedua.

### *Quick Design*

Pada iterasi sebelumnya, terdapat masukan untuk menambahkan beberapa proses tambahan yang dibutuhkan demi memberikan penjelasan yang lebih baik akan hasil dari produk akhir nantinya. Berikut adalah langkah-langkah kami mengakomodasi perubahan tersebut.

#### Alur Pengguna

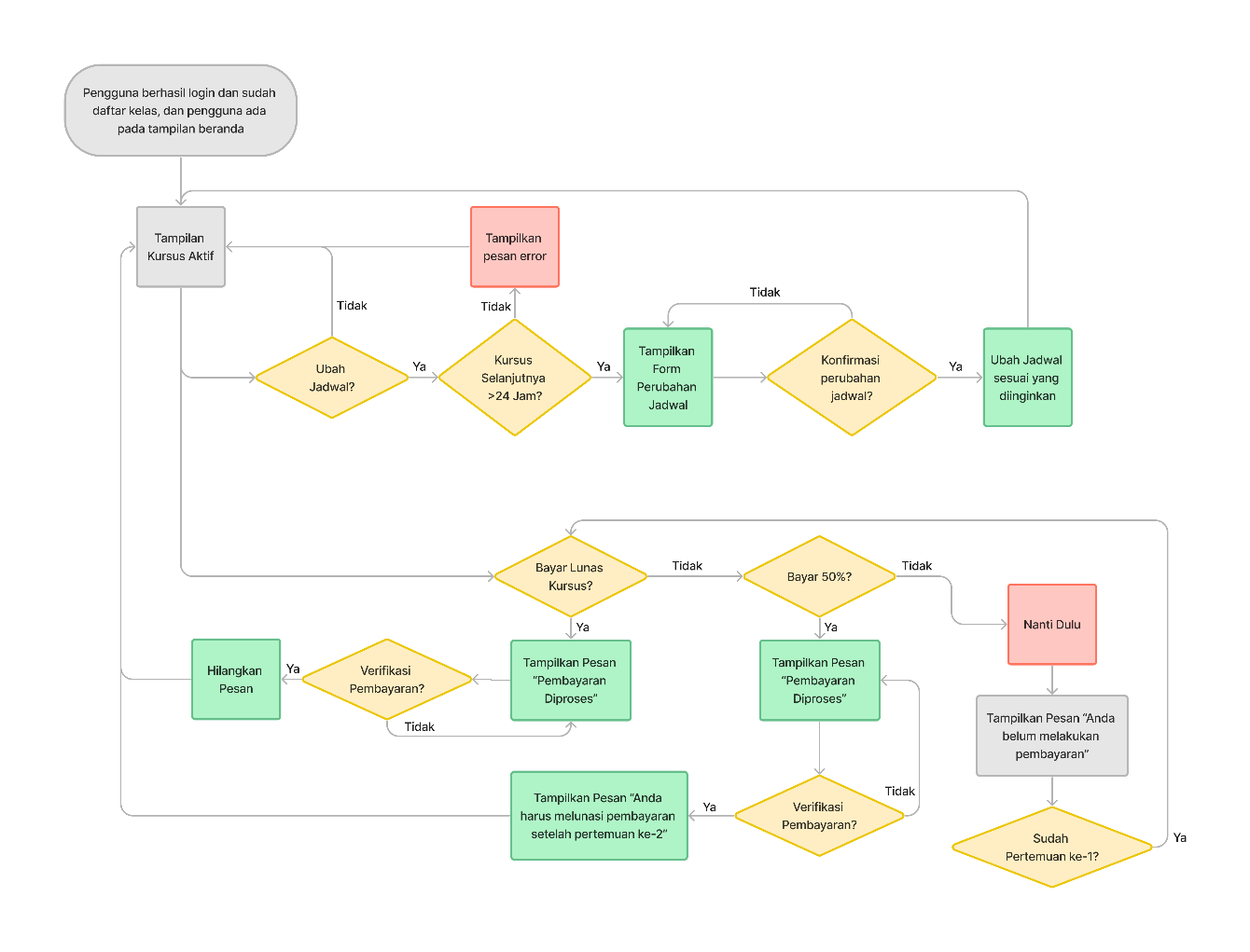
Pihak kursus berpendapat dengan menambahkan proses yang mengindikasikan status pembayaran akan memberikan kesan transparansi yang lebih baik kepada calon peserta kursus. Selain itu, aplikasi diharapkan dapat menampilkan daftar kelas kursus yang ditawarkan kepada calon peserta kursus sehingga pihak kursus berkesempatan untuk melakukan pemasaran jasa. Berikut adalah rencana perubahan yang kami lakukan untuk memenuhi dua poin masukan diatas.



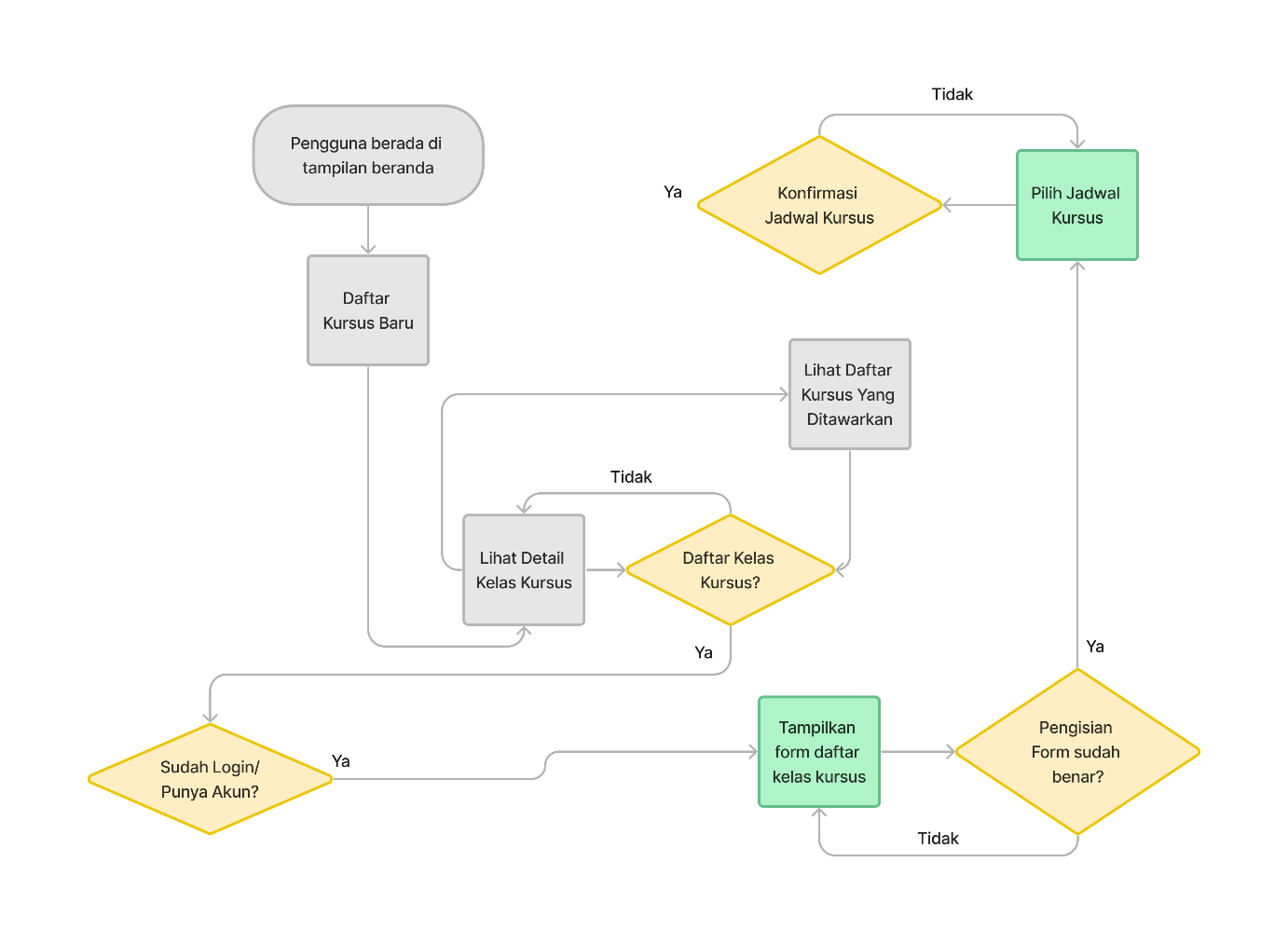
Jika dapat dilihat dari gambar diatas, dan dengan membandingkan alur pengguna pada iterasi sebelumnya, alur diatas menjadi lebih kompleks dengan penambahan status pembayaran dan daftar kelas yang ditawarkan, dibawah ini adalah penjelasan lebih rinci tentang alur tugas dari perubahan alur pengguna diatas.



Tidak ada perubahan yang disarankan dari pihak kursus mengemudi maupun pengguna diluar pihak kursus untuk alur tugas *login* atau daftar akun.



Alur tugas diatas sebelumnya hanya untuk mengubah jadwal yang sudah dipilih pada saat pendaftaran kursus. Pada iterasi kedua ini, kami menambahkan alur status pembayaran, dimana peserta kursus memiliki dua pemilihan pembayaran, pembayaran lunas dan DP 50%. Peserta kursus yang memilih DP 50% wajib melakukan pelunasan setelah pertemuan ke-2 kursus, jika tidak, kursus akan dibekukan sementara. Namun, jika setelah pertemuan ke-1, peserta kursus belum melakukan pembayaran apapun, kursus juga akan dibekukan sampai peserta membayar biaya kursus.



Dapat dilihat pada gambar diatas, setelah calon peserta kursus berada pada tampilan detail kelas kursus, pengguna dapat melihat daftar kelas kursus lain yang ditawarkan oleh instruktur atau lembaga kursus bersangkutan. Hal ini memungkinkan calon peserta untuk melakukan perbandingan antar kelas atau dengan instruktur lain sebelum melakukan pendaftaran kursus.

#### Sketsa Awal Aplikasi

Karena terdapat penambahan alur, maka, kami juga harus melakukan penambahan tampilan *prototype* untuk diujikan ulang kepada pengguna. Dibawah ini adalah daftar sketsa awal aplikasi untuk iterasi kedua.

|  |  |  |
| --- | --- | --- |
| **No.** | **Sketsa Aplikasi** | **Keterangan** |
| 1 |  | Sketsa awal untuk tampilan depan aplikasi tidak mengalami perubahan kecuali teks pada tombol yang dibuat lebih tebal |
| 2 |  | Sketsa awal untuk tampilan *form* pendaftaran akun disesuaikan dengan masukan yang diterima pada iterasi sebelumnya |

|  |  |  |
| --- | --- | --- |
| **No.** | **Sketsa Aplikasi** | **Keterangan** |
| 3 |  | Sketsa awal untuk tampilan beranda, kolom pencarian dipindah ke *navbar* dibagian atas aplikasi. Bagian kursus berlangsung diubah agar lebih mudah diproses |
| 4 |  | Sketsa awal untuk tampilan detail kelas kursus disesuaikan dengan masukan yang diterima. Ditambahkan juga fungsi untuk menampilkan semua kelas yang ditawarkan oleh instruktur atau lembaga kursus |

|  |  |  |
| --- | --- | --- |
| **No.** | **Sketsa Aplikasi** | **Keterangan** |
| 5 |  | Sketsa awal untuk tampilan daftar kelas kursus yang dimiliki lembaga kursus atau instruktur terkait |
| 6 |  | Sketsa awal untuk tampilan *form* pendaftaran kelas kursus sudah disesuaikan dengan masukan yang diterima dari iterasi sebelumnya |

|  |  |  |
| --- | --- | --- |
| **No.** | **Sketsa Aplikasi** | **Keterangan** |
| 7 |  | Sketsa awal untuk tampilan pemilihan jadwal kursus, yang sudah disesuaikan berdasarkan masukan yang diterima pada iterasi sebelumnya |
| 8 |  | Sketsa awal untuk tampilan detail kursus aktif yang menunjukkan status pembayaran |

|  |  |  |
| --- | --- | --- |
| **No.** | **Sketsa Aplikasi** | **Keterangan** |
| 9 |  | Sketsa awal untuk tampilan detail kursus aktif untuk status pembayaran yang berbeda |
| 10 |  | Sketsa awal untuk tampilan detail kursus aktif dengan asumsi peserta kursus sudah melakukan pembayaran dan sudah diverifikasi. |

### *Built Prototype*

Untuk *prototype* iterasi kedua, kami melanjutkan hasil dari proses desain singkat sebelumnya. Berikut adalah hasil pembangunan *prototype*.

|  |  |  |
| --- | --- | --- |
| **No.** | **Sketsa Aplikasi** | **Keterangan** |
| 1 |  | Tampilan depan aplikasi didasarkan pada sketsa awal pada proses sebelumnya |

|  |  |  |
| --- | --- | --- |
| **No.** | **Sketsa Aplikasi** | **Keterangan** |
| 2 |  | Tampilan *form* pendaftaran didasarkan pada sketsa awal dan masukan yang disarankan pada iterasi sebelumnya |
| 3 |  | Tampilan beranda aplikasi dimana kolom pencarian diganti menjadi ikon pencarian dibagian *navbar* |

|  |  |  |
| --- | --- | --- |
| **No.** | **Sketsa Aplikasi** | **Keterangan** |
| 4 |  | Tampilan detail kelas kursus disesuaikan dengan masukan yang diterima pada iterasi sebelumnya |
| 5 |  | Tampilan daftar kelas kursus yang dimiliki lembaga kursus sesuai masukan yang didapatkan iterasi sebelumnya |

|  |  |  |
| --- | --- | --- |
| **No.** | **Sketsa Aplikasi** | **Keterangan** |
| 6 |  | Tampilan *form* pendaftaran kelas kursus berdasarkan hasil sketsa awal pada proses sebelumnya |
| 7 |  | Tampilan untuk pemilihan jadwal kursus, yang sudah disesuaikan berdasarkan masukan yang diterima pada iterasi sebelumnya |

|  |  |  |
| --- | --- | --- |
| **No.** | **Sketsa Aplikasi** | **Keterangan** |
| 8 |  | Tampilan detail kursus aktif yang menunjukkan status pembayaran |
| 9 |  | Tampilan detail kursus aktif untuk status pembayaran yang berbeda |

|  |  |  |
| --- | --- | --- |
| **No.** | **Sketsa Aplikasi** | **Keterangan** |
| 10 |  | Tampilan detail kursus aktif akhir yang sudah disesuaikan dengan masukan yang diterima pada iterasi sebelumnya |

### *Customer Evaluation*

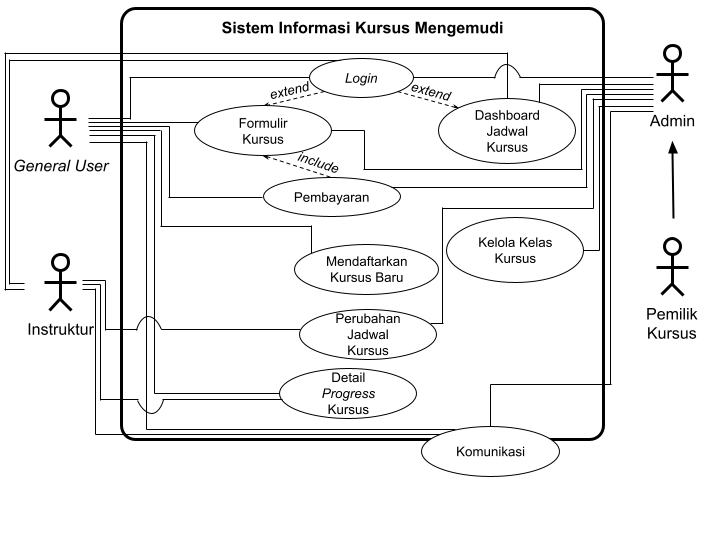
Setelah proses pembangunan *prototype* selanjutnya kami melakukan pengujian terhadap hasil desain *prototype* dan alur pengguna yang dirancang. Pengujian ini dilakukan kepada pengguna *prototype* yang sama dengan yang menguji *prototype* iterasi pertama. Tujuannya adalah untuk mendapatkan masukan dan umpan balik dari pengguna terkait dengan desain dan alur pengguna yang baru. Masukan dan umpan balik ini kemudian akan digunakan untuk menyempurnakan *prototype* sebelum memasuki tahapan pengembangan selanjutnya. Namun, setelah kami kembali melakukan pengujian, pengguna *prototype* lebih menyukai hasil desain *prototype* iterasi kedua, dan *prototype* dirasa tidak membutuhkan perbaikan.

### *Review & Updation*

Dari hasil pengujian awal sebelumnya, pengguna *prototype* tidak memberikan masukan atau umpan balik terhadap *prototype* iterasi kedua. Hal ini menunjukkan bahwa pengguna *prototype* merasa puas dengan desain dan alur pengguna yang baru. Oleh karena itu, langkah selanjutnya adalah melakukan finalisasi desain untuk aplikasi yang kami rancang, tujuannya adalah agar desain yang sudah dibuat nantinya merupakan desain yang sama yang dihasilkan pada saat proses pengembangan, atau dalam istilah pengembangan perangkat lunak sering disebut dengan *handoff*.

## *Design*

Setelah melewati proses analisa kebutuhan dan desain *prototype* singkat diatas, langkah selanjutnya adalah menerjemahkannya ke diagram-diagram UML. Hal ini ditujukan untuk memberikan alur yang lebih jelas tentang interaksi-interaksi yang dilakukan user dengan sistem, serta sekali lagi memeriksa apakah semua fungsi yang disebutkan diatas sudah terakomodir semuanya. Untuk memecah kompleksitas dan menghindari kesalahpahaman, kami akan menjelaskan setiap *use case* dengan *Activity Diagram* dan *Use Case Scenario.*

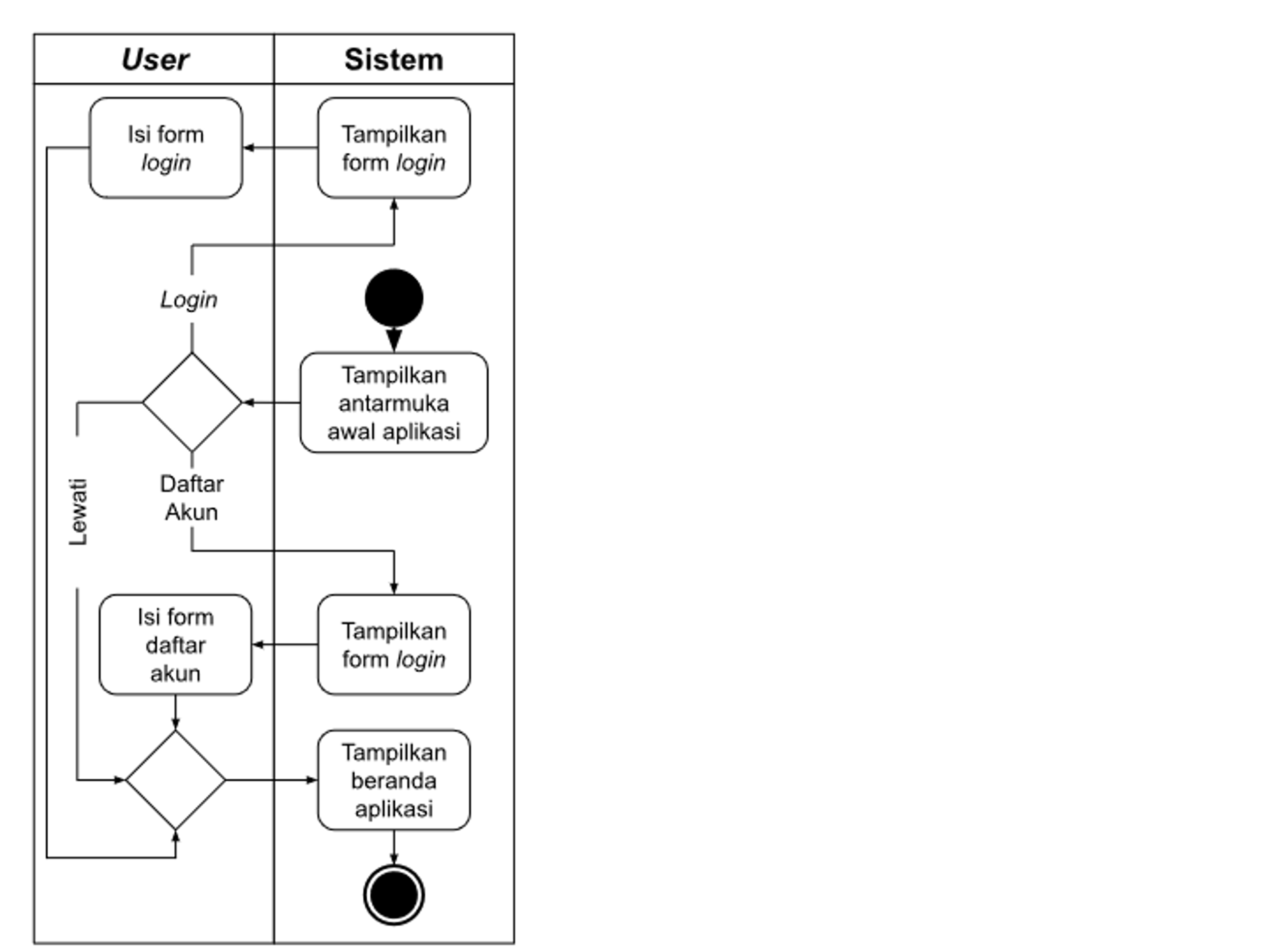


Kami mengelompokkan pengguna aplikasi kami menjadi 3 kelompok. Kelompok pertama adalah *General User*, yang mewakili masyarakat awam. Pengguna Umum dapat memanfaatkan aplikasi kami untuk mengikuti kursus mengemudi dan meningkatkan keterampilan berkendara mereka. Atau, mereka juga dapat menawarkan jasa mereka sebagai penyedia kursus mengemudi atau instruktur, membantu orang lain yang ingin belajar mengemudi. Pada gambar diatas, dapat dilihat bahwa *General User*, setidaknya dapat melakukan 7 operasi pada aplikasi kami nantinya, *Login* atau daftar sebagai pengguna aplikasi dan aktivitas pengelolaan akun yang lainnya, kemudian mengisi formulir pendaftaran kursus, melakukan pembayaran kursus, mendaftarkan diri sebagai penyedia jasa kursus baru, melakukan perubahan jadwal kursus yang diikuti, melihat detail progress kursus, dan yang terakhir melakukan komunikasi dengan pihak kursus, baik bagian Admin atau Instruktur kursus terkait.

Kelompok pengguna selanjutnya adalah Instruktur, Instruktur mengemudi adalah pihak yang melakukan pertemuan langsung dengan siswa atau *General User* pada saat proses kursus mengemudi nantinya. Untuk Instruktur, operasi-operasi yang dapat dilakukan di dalam aplikasi kami adalah proses *Login* dan aktivitas pengelolaan akun yang lain, selanjutnya menampilkan jadwal kursus per hari, mengajukan perubahan jadwal kursus, melihat detail progress kursus siswa terkait, dan melakukan komunikasi antar pihak Admin atau Siswa yang berada di bawah tanggung jawabnya.

*General User* dapat menjadi Pemilik Kursus atau Admin jasa kursus, apabila pengguna bersangkutan sudah menyelesaikan proses pendaftaran jasa kursus baru, dengan menjadi Pemilik Kursus atau Admin jasa kursus, pengguna dapat melakukan operasi-operasi seperti *Login* dan aktivitas-aktivitas pengelolaan akun lainnya, melihat formulir kursus yang diisi oleh calon siswa, mengubah status pembayaran, melihat jadwal kursus per hari, mengajukan perubahan jadwal kursus kepada Siswa dan Instruktur, melihat detail progress kursus siswa, melakukan aktivitas pengelolaan kelas kursus yang ditawarkan seperti mengubah harga, menetapkan kuota kelas, dan proses-proses lain, serta Pemilik/Admin Kursus dapat melakukan komunikasi dengan Instruktur atau Siswa kursus.

### *Login* / Daftar Akun

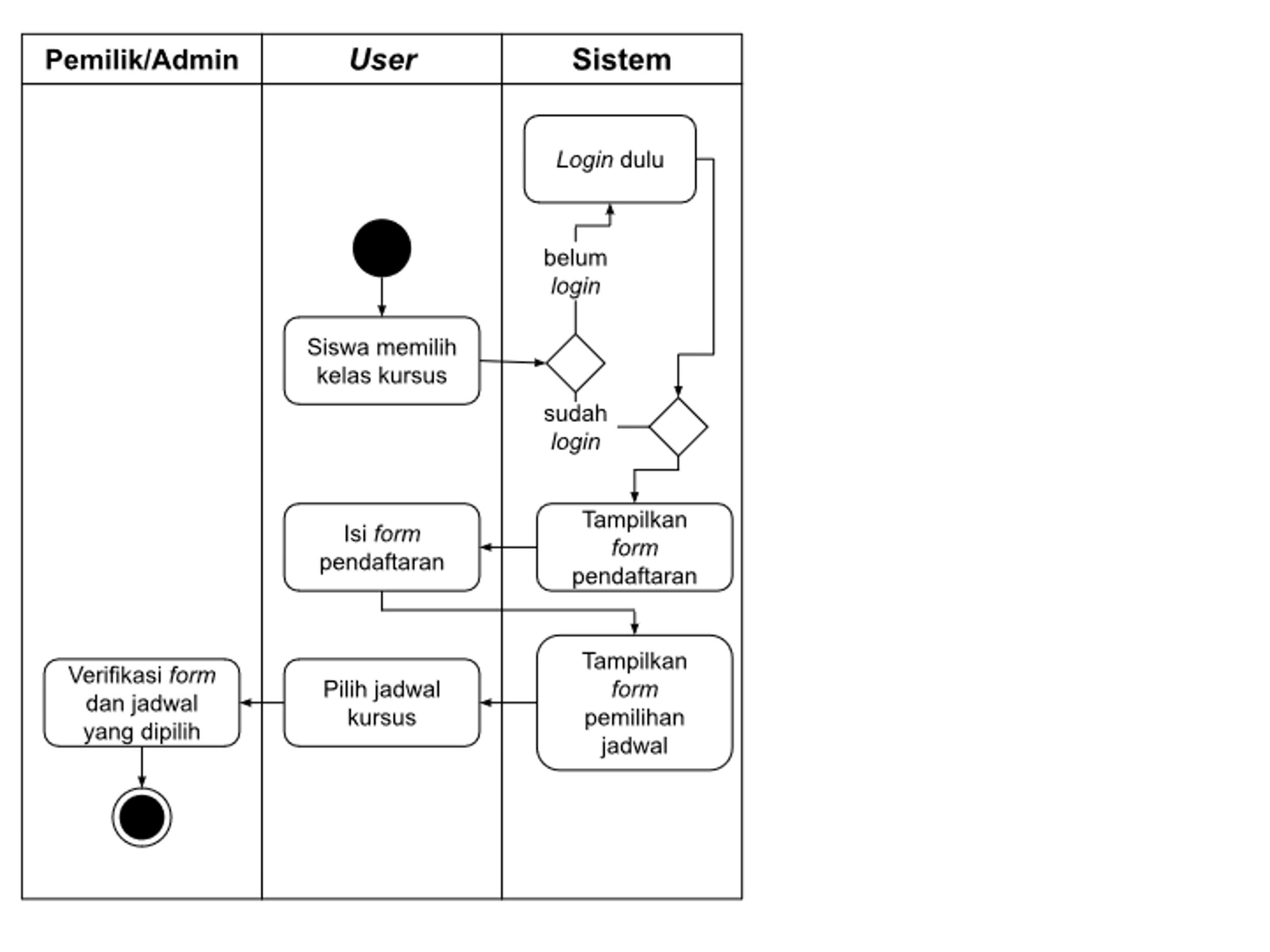


Untuk alur *Login*, sistem mulai dengan menanyakan apakah pengguna sudah punya akun? Jika ya, pengguna bisa langsung melakukan *Login*, jika tidak sistem mempunyai 2 opsi lanjutan, apakah pengguna ingin daftar akun sekarang atau pengguna bisa melakukan aksi ini nanti ketika sistem membutuhkan penyimpanan data untuk akun bersangkutan.

|  |  |  |
| --- | --- | --- |
| **Kode Use Case** | UC\_login\_untuk\_semua\_user | |
| **Nama Use Case** | *Login* | |
| **Aktor** | *General User*, Pemilik / Admin, Instruktur | |
| **Deskripsi** | Skenario untuk menyimpan data dari segala aktivitas yang dilakukan oleh Aktor | |
| **Kondisi Awal** | Sistem baru selesai memuat tampilan awal | |
| **Kondisi Akhir** | Aktor berada di halaman beranda/*dashboard* | |
| **Alur Kejadian Normal** | **Aktor** | **Sistem** |
|  | 1. Sistem menampilkan tiga tombol (*Login*, Daftar, dan Nanti) |
| 2. Aktor memilih tombol *Login* |  |
|  | 3. Sistem menampilkan *form* *Login* |
| 4. Aktor memasukkan data akun yang tersimpan di database |  |
|  | 5. Sistem memverifikasi data yang diinputkan Aktor |

|  |  |  |
| --- | --- | --- |
|  | **Aktor** | **Sistem** |
|  |  | 6. Sistem mengarahkan Aktor |
| **Alur Kejadian Alternatif** | 2a. Aktor memilih tombol “Daftar”  2b. Aktor memilih tombol “Nanti” |  |
|  | 3a. Sistem menampilkan *form* daftar akun  3b. Sistem mengubah peran Aktor dari *General User* menjadi Siswa |
| 4a. Aktor memasukkan data akun baru  4c. Aktor memasukkan data yang tidak ada di database | 4b. Sistem mengalihkan Aktor ke beranda/*dashboard* |
|  | 5a. Sistem menyimpan data baru yang diinputkan Aktor  5c. Sistem tidak menemukan data yang diinputkan Aktor |
|  | 6a. Sistem mengalihkan Aktor ke beranda/*dashboard*  6c. Sistem mengalihkan Aktor ke tampilan awal dan mengembalikan pesan *error* ke Aktor |

### Siswa Mendaftar Kursus

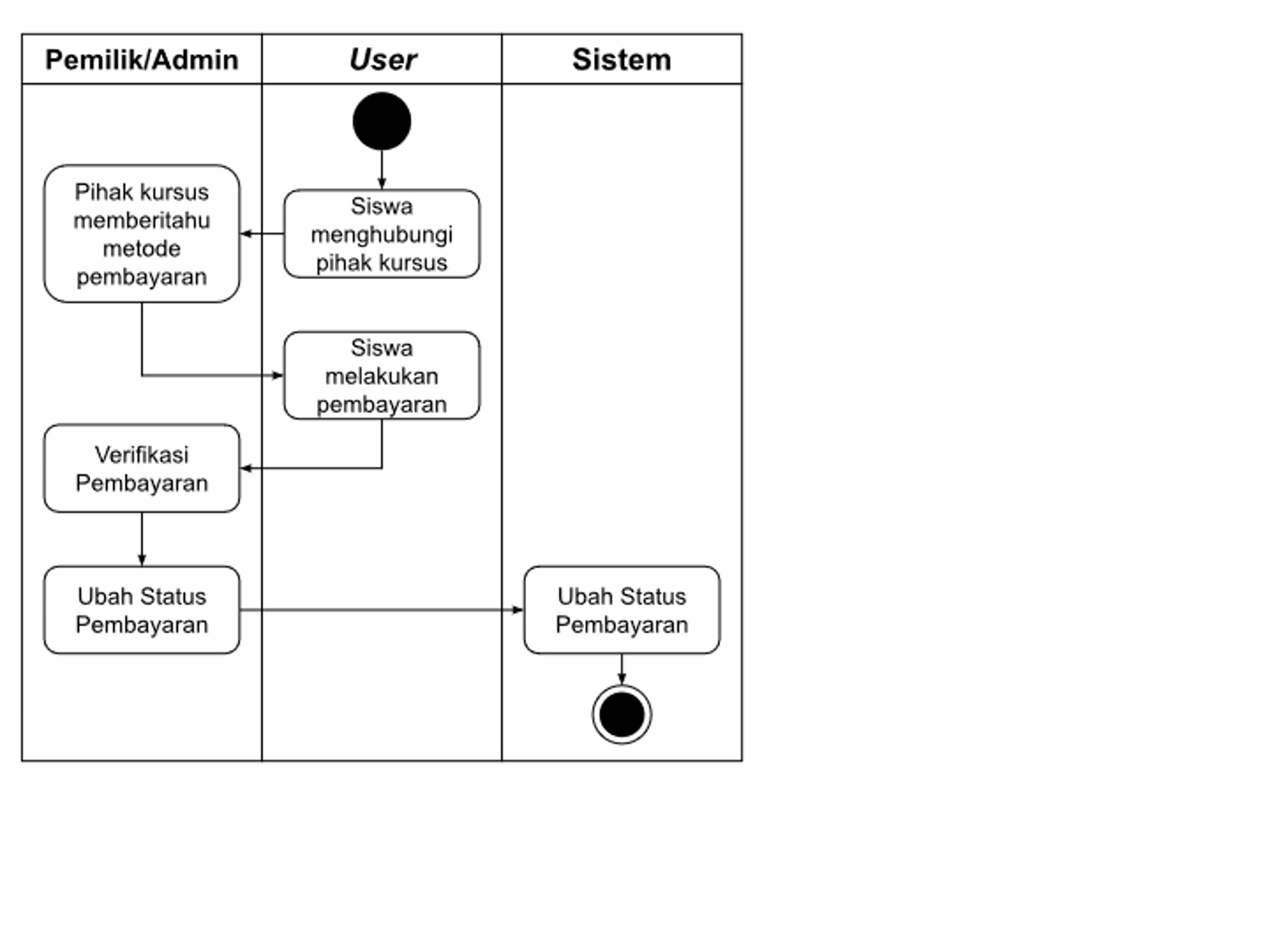


Untuk alur mendaftar kursus, Siswa memilih kelas kursus yang dimaksud terlebih dahulu, selanjutnya sistem akan melakukan pengecekkan apakah pengguna sudah melakukan proses *login* sebelumnya. Proses *login* diperlukan agar sistem mengetahui data yang akan di-*entry* oleh pengguna ini tersimpan bersama dengan data akun pengguna.

|  |  |
| --- | --- |
| **Kode Use Case** | UC\_daftar\_kelas\_kursus |
| **Nama Use Case** | Mendaftar kelas kursus |
| **Aktor** | Siswa & Pemilik/Admin |
| **Deskripsi** | Skenario untuk mendaftarkan diri mengikuti kursus mengemudi |

|  |  |  |
| --- | --- | --- |
| **Kondisi Awal** | Calon siswa sudah memilih kelas kursus yang diinginkan | |
| **Kondisi Akhir** | Calon siswa menyelesaikan pengisian formulir pendaftaran | |
| **Alur Kejadian Normal** | **Aktor** | **Sistem** |
| 1. Calon Siswa menekan tombol “Daftar Kelas” |  |
|  | 2. Sistem mengalihkan Calon Siswa ke formulir pendaftaran kursus |
| 3. Calon Siswa mengisi formulir pendaftaran dengan lengkap |  |
|  | 4. Sistem menyimpan *entry* yang dilakukan Calon Siswa |
|  | 5. Sistem mengirimkan pemberitahuan ke Pemilik/Admin |
| 6. Pemilik/Admin memverifikasi formulir yang diisi calon siswa |  |
| **Alur Kejadian Alternatif** | 3. Calon Siswa tidak jadi meneruskan pendaftaran |  |
|  | 4. Sistem mengalihkan Calon Siswa ke beranda/*dashboard* |

### Pembayaran

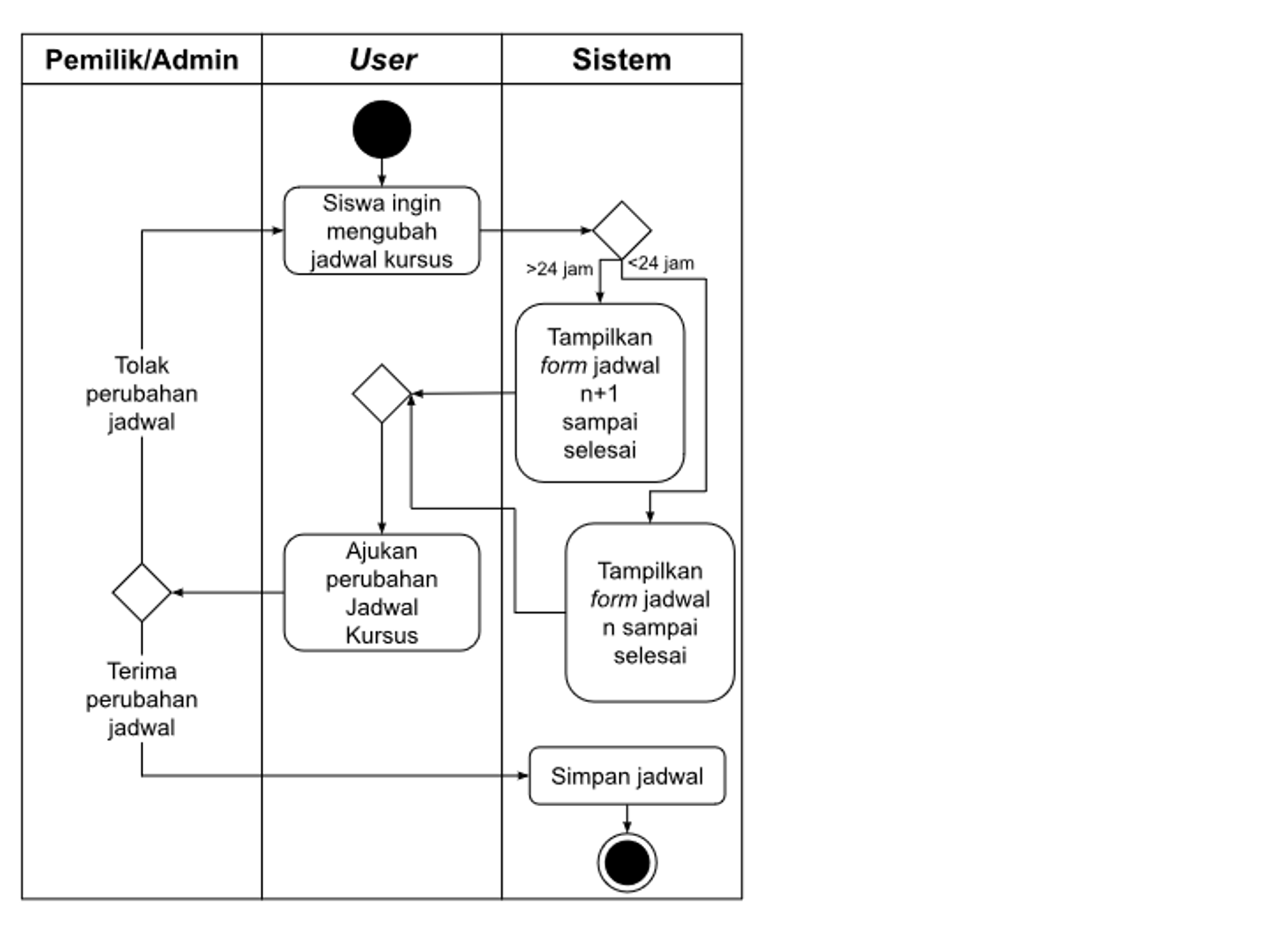


Pada sistem yang kami kembangkan proses pembayaran terjadi di luar sistem, sistem hanya memberitahukan kedua pihak status pembayaran saat ini.

|  |  |
| --- | --- |
| **Kode Use Case** | UC\_status\_pembayaran |
| **Nama Use Case** | Status Pembayaran |
| **Aktor** | Siswa & Pemilik/Admin |
| **Deskripsi** | Skenario pemberitahuan status pembayaran |
| **Kondisi Awal** | Calon siswa baru saja menyelesaikan *form* pendaftaran |
| **Kondisi Akhir** | Calon siswa mendapatkan informasi status pembayaran |

|  |  |  |
| --- | --- | --- |
| **Alur Kejadian Normal** | **Aktor** | **Sistem** |
|  | 1. Sistem mengarahkan Calon siswa ke halaman detail kursus |
| 2. Siswa menghubungi pihak kursus menanyakan tentang proses pembayaran |  |
| 3. Siswa melakukan proses pembayaran |  |
| 4. Pemilik/Admin memverifikasi pembayaran |  |
| 5. Pemilik/Admin mengubah status pembayaran |  |
|  | 6. Sistem menampilkan status pembayaran yang dipilih Admin |
| **Alur Kejadian Alternatif** | - | - |

### Mengubah Jadwal Kursus



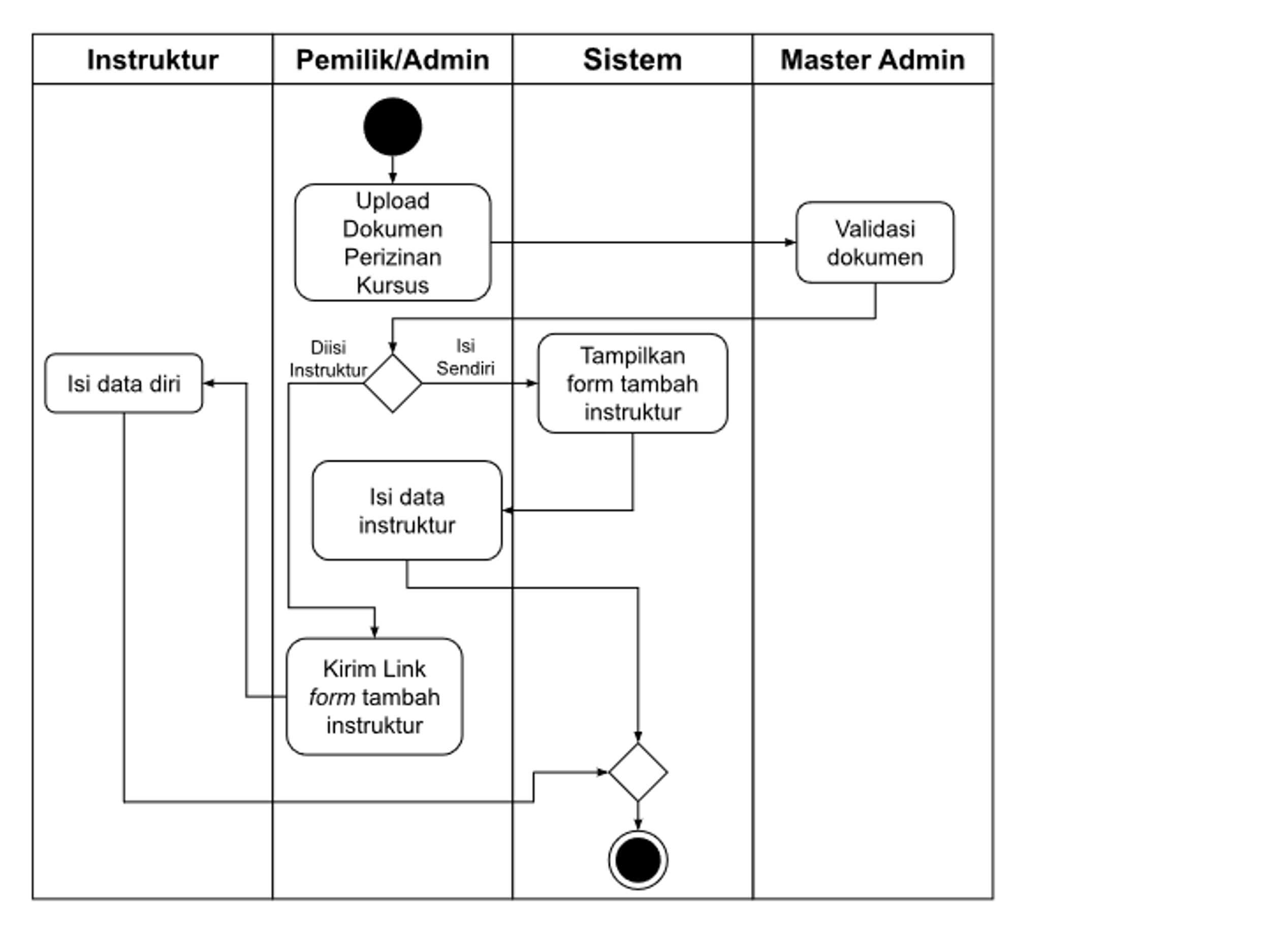
Jika siswa ingin mengubah jadwal, sistem masih bisa melanjutkan proses apabila pertemuan selanjutnya lebih dari 24 jam, jadwal yang diubah nantinya akan mengubah pertemuan selanjutnya sampai pertemuan akhir. Sebagai contoh, siswa memilih kursus dengan 5 pertemuan, pada pertemuan ke-3 siswa tidak dapat hadir sesuai dengan jadwal, jika siswa memilih untuk melakukan perubahan jadwal, maka siswa dapat mengubah jadwal kursus dari pertemuan ke-3 sampai pertemuan ke-5.

|  |  |
| --- | --- |
| **Kode Use Case** | UC\_perubahan\_jadwal |
| **Nama Use Case** | Pengajuan perubahan jadwal |
| **Aktor** | Siswa& Pemilik/Admin |
| **Deskripsi** | Skenario pengajuan perubahan jadwal |

|  |  |  |
| --- | --- | --- |
| **Kondisi Awal** | Siswa berada di halaman detail *progress* kursus | |
| **Kondisi Akhir** | Siswa berhasil mengubah rangkaian jadwal kursus | |
| **Alur Kejadian Normal** | **Aktor** | **Sistem** |
| 1. Aktor menekan tombol ‘Jadwal Kursus’ |  |
|  | 2. Sistem mengalihkan Aktor ke halaman jadwal kursus siswa terkait |
| 3. Aktor menekan tombol ‘Ubah Jadwal’ |  |
|  | 4. Sistem memeriksa apakah pertemuan selanjutnya < 24 jam |
|  | 5a. Jika pertemuan selanjutnya < 24 jam, jadwal yang dapat diubah dimulai dari pertemuan setelah pertemuan mendatang  5b. Jika pertemuan selanjutnya > 24 jam, perubahan jadwal dapat dimulai dari pertemuan mendatang |
| 6. Aktor menyesuaikan jadwal yang tersedia |  |

|  |  |  |
| --- | --- | --- |
|  | **Aktor** | **Sistem** |
| 7. Aktor menekan tombol ‘Ajukan Perubahan Jadwal’ |  |
|  | 8. Sistem mengirimkan pemberitahuan ke Aktor lain untuk dikonfirmasi pengajuan jadwal yang dikirim |
| 9. Aktor lainnya mengkonfirmasi perubahan jadwal |  |
|  | 10. Sistem menyimpan perubahan jadwal |
| **Alur Kejadian Alternatif** | 6. Aktor membatalkan perubahan jadwal |  |
|  | 7. Sistem mengarahkan Aktor ke halaman detail *progress* kursus |
| 9. Aktor lainnya menolak perubahan jadwal |  |
|  | 10. Sistem mengirimkan pemberitahuan penolakan pengajuan jadwal ke Aktor |

### Mengajukan Kursus Baru



Seperti proses-proses sebelumnya, karena proses ini memerlukan sistem untuk menyimpan data yang di*-entry* oleh pengguna, sebelum melakukan proses upload dokumen perizinan kursus, sistem akan memeriksa apakah pengguna sudah *login*? Jika sudah, proses akan berlanjut, jika belum, sistem akan mengarahkan pengguna ke halaman *login*.

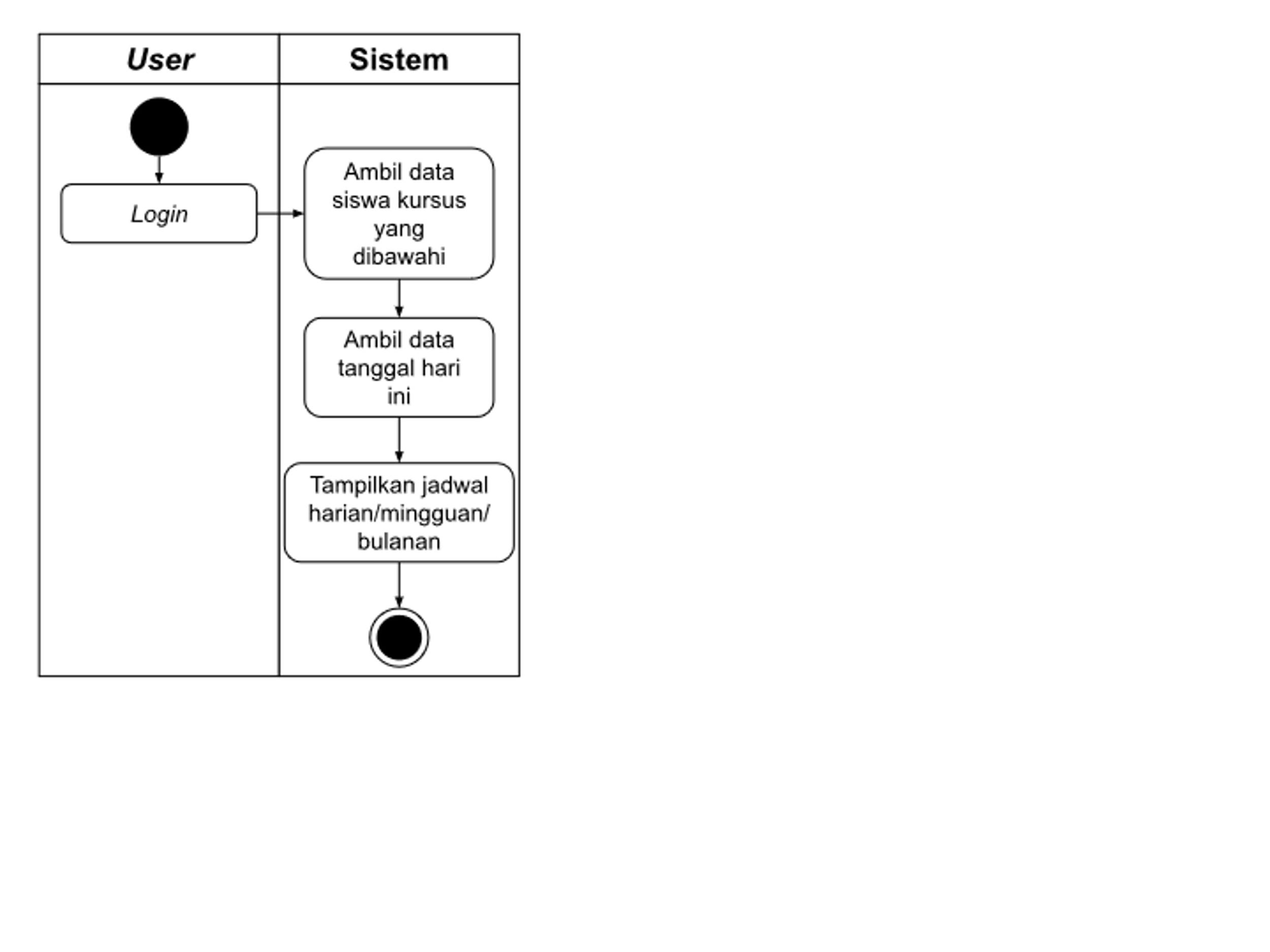
|  |  |
| --- | --- |
| **Kode Use Case** | UC\_pendaftaran\_kursus\_baru |
| **Nama Use Case** | Pendaftaran Kursus Baru |
| **Aktor** | Instruktur, *General User* (Calon Pemilik) & Master Admin |
| **Deskripsi** | Skenario untuk mendaftarkan jasa kursus mengemudi baru |
| **Kondisi Awal** | Calon Pemilik berada di halaman beranda/*dashboard* |

|  |  |  |
| --- | --- | --- |
| **Kondisi Akhir** | Pemilik berada pada halaman beranda/*dashboard* | |
| **Alur Kejadian Normal** | **Aktor** | **Sistem** |
| 1. Calon Pemilik berpindah ke halaman profil |  |
|  | 2. Sistem menampilkan data dan tombol-tombol yang nantinya ditampilkan di halaman profil |
| 3. Calon Pemilik menekan menu ‘Ingin jadi Instruktur?’ |  |
|  | 4. Sistem mengalihkan Aktor ke halaman *upload* dokumen perizinan |
| 5. Calon Pemilik mengupload dokumen perizinan penyelenggaraan kursus |  |
|  | 6. MasterAdmin memverifikasi keaslian dokumen yang diupload oleh Calon Pemilik |
|  | 7. Sistem mengirim pesan berhasil ke Calon Pemilik kemudian mengarahkan Calon Pemilik untuk mengisi data Instruktur |

|  |  |  |
| --- | --- | --- |
|  | **Aktor** | **Sistem** |
|  | 8. Sistem memberikan opsi kepada pengguna untuk mengisi data Instruktur sendiri atau diisi oleh Calon Instruktur bersangkutan |
| 9a. Calon Pemilik memilih mengisi data Instruktur sendiri  9b. Calon Pemilik memilih agar diisi oleh Calon Instruktur bersangkutan |  |
| 10a. Calon Pemilik mengisi semua data Instruktur dengan lengkap  10b. Calon Pemilik mengirimkan link *form* penambahan Instruktur ke Calon Instruktur |  |
| 11b. Calon Instruktur menerima *link* undangan mengajar | 11a. Sistem menyimpan *entry* data dari Calon Pemilik, kemudian mengarahkan Calon Pemilik ke tampilan beranda/*dashboard* |
|  | 12b. Sistem menampilkan |

|  |  |  |
| --- | --- | --- |
|  | **Aktor** | **Sistem** |
|  | halaman pendaftaran akun untuk diisi Calon Instruktur |
| 13b. Calon Instruktur mengisi *form* penambahan Instruktur |  |
|  | 14b. Sistem menyimpan *entry* yang dilakukan Instruktur |
| **Alur Kejadian Alternatif** |  | 6. *System* admin menolak keaslian dokumen atau dokumen yang di-*upload* dianggap kurang jelas |
|  | 7. Sistem mengirim pesan gagal ke Calon Pemilik dan mengarahkan untuk melakukan *upload* ulang |
| 10a. Calon Pemilik tidak mengisi *form* dengan lengkap atau membatalkan pengisian |  |
| 13b. Instruktur tidak mengisi *form* dengan lengkap atau membatalkan pengisian | 11a. Sistem mengalihkan Calon Pemilik ke halaman profil |
|  | 14b. Sistem mengalihkan Calon Instruktur ke halaman profil |

### Menampilkan Dashboard Jadwal Kursus



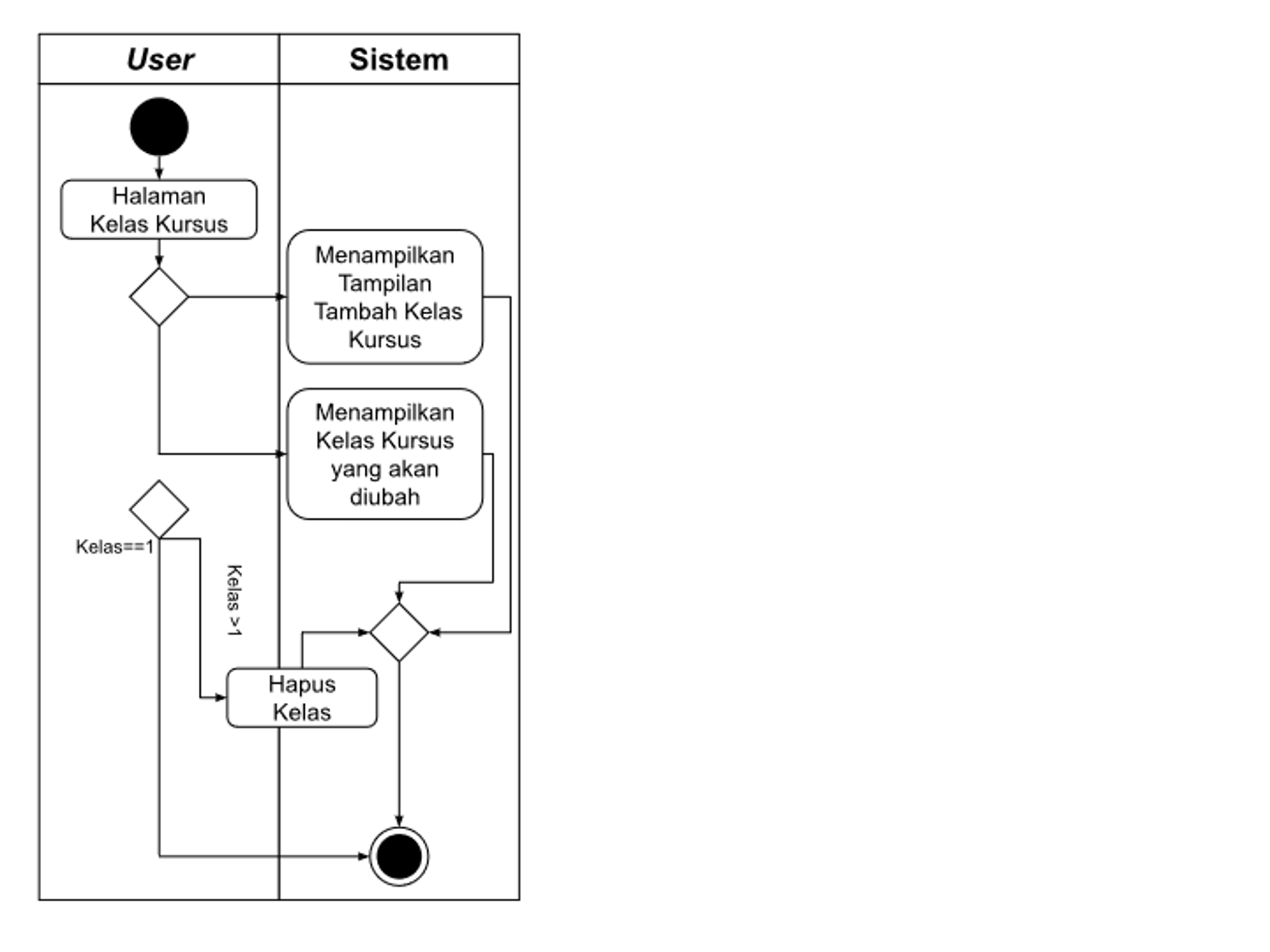
Untuk menampilkan *dashboard* pihak kursus (pemilik/admin kursus dan instruktur) harus menyelesaikan alur *Login* terlebih dulu, selanjutnya sistem akan menampilkan jadwal kursus yang dikelompokkan berdasarkan tanggal, dimulai dengan hari ini, esok hari, dan hari selanjutnya. Yang membedakan adalah jika instruktur hanya bisa melihat jadwal untuk dirinya sendiri, sedangkan pemilik/admin dapat melihat seluruh jadwal instruktur yang dibawahinya. Jumlah hari yang ditampilkan dapat diatur untuk menghindari pengguna kewalahan memproses informasi.

|  |  |
| --- | --- |
| **Kode Use Case** | UC\_tampilkan\_jadwal\_kursus |
| **Nama Use Case** | Menampilkan *dashboard* jadwal kursus |
| **Aktor** | Pemilik / Admin, Instruktur |
| **Deskripsi** | Skenario untuk menampilkan jadwal kursus per hari |

|  |  |  |
| --- | --- | --- |
| **Kondisi Awal** | Sistem selesai memverifikasi data akun yang diinputkan Aktor pada proses login | |
| **Kondisi Akhir** | Sistem menampilkan semua jadwal sesuai batas hari yang ditentukan Aktor | |
| **Alur Kejadian Normal** | **Aktor** | **Sistem** |
|  | 1. Sistem mengumpulkan data siswa yang kursus dengan Aktor |
|  | 2. Sistem secara *default*, menampilkan jadwal untuk 3 hari (hari ini, esok hari, dan hari setelahnya) |
|  | 3. Sistem memberikan opsi lain untuk menampilkan jadwal dalam 15 hari dan 30 hari |
| **Alur Kejadian Alternatif** | **Aktor** | **Sistem** |
| 4a. Aktor memilih opsi untuk tampilkan jadwal dalam 15 hari |  |
|  | 5a. Sistem menampilkan jadwal dimulai dengan hari ini sampai 14 hari kedepan |
| 4b. Aktor memilih opsi untuk tampilkan jadwal dalam 30 |  |

|  |  |  |
| --- | --- | --- |
|  | **Aktor** | **Sistem** |
| hari |  |
|  | 5b. Sistem menampilkan jadwal dimulai dengan hari ini sampai 29 hari kedepan |

### Pengelolaan Kelas Kursus



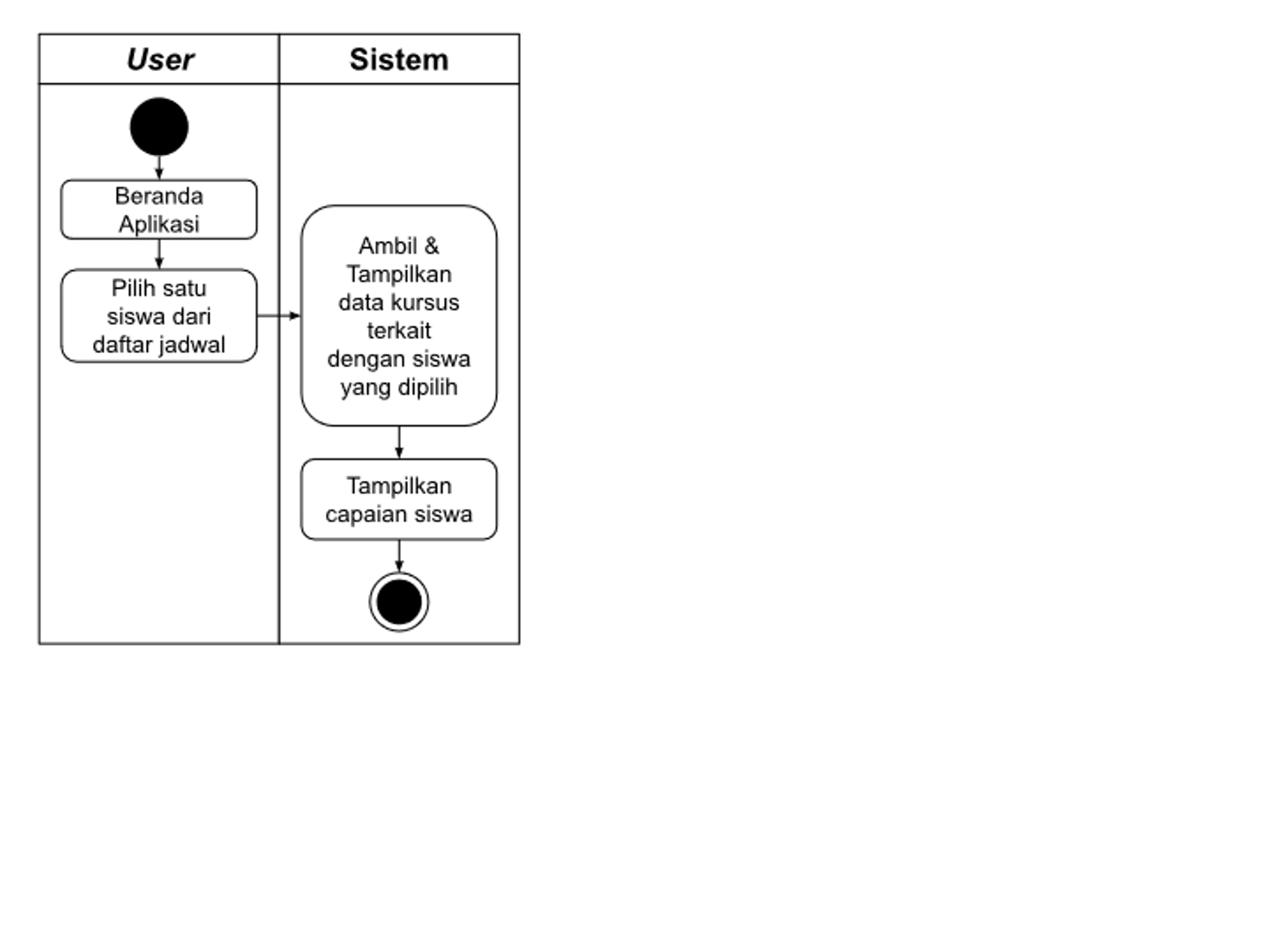
Pemilik/Admin dari suatu kursus dapat melakukan penambahan atau menonaktifkan/hapus kelas kursus yang ada di halaman kursus mereka selama masih ada 1 kelas kursus tersisa. Selain itu, pemilik/admin ini dapat mengubah informasi-informasi yang berkaitan dengan kelas kursus, seperti deskripsi kursus, lama kursus, kategori kursus, benefit, harga, dan lain-lain.

|  |  |  |
| --- | --- | --- |
| **Kode Use Case** | UC\_pengelolaan\_kelas\_kursus | |
| **Nama Use Case** | Kelola kelas Kursus | |
| **Aktor** | Pemilik / Admin, Instruktur | |
| **Deskripsi** | Skenario untuk menambah, mengubah, dan menghapus/menonaktifkan kelas kursus yang ditawarkan | |
| **Kondisi Awal** | Aktor berada di halaman kursus | |
| **Kondisi Akhir** | Aktor berhasil menambah kelas kursus baru | |
| **Alur Kejadian Normal** | **Aktor** | **Sistem** |
| 1. Aktor menekan tombol “Tambah kelas” pada halaman kursus |  |
|  | 2. Sistem mengalihkan Aktor ke *form* tambah kelas |
| 3. Aktor mengisi semua *form* dengan lengkap | 3. Sistem memverifikasi bahwa semua *form* terisi |
| 4. Aktor menekan tombol “Tawarkan kelas” |  |
|  | 5. Sistem menyimpan *entry* yang dilakukan Aktor |
|  | 6. Sistem mengarahkan Aktor ke halaman kursus dan menampilkan kelas yang baru ditambahkan pada |

|  |  |  |
| --- | --- | --- |
|  | **Aktor** | **Sistem** |
|  | halaman kursus |
| **Alur Kejadian Alternatif** |  | 1a. Sistem menampilkan tombol “Ubah kelas” di masing-masing kelas kursus  1b. Sistem menampilkan tombol “Nonaktfikan kelas” jika kondisi terpenuhi |
|  | 2a. Aktor menekan tombol “Ubah kelas” di kelas Kursus yang dipilih  2b. Aktor menekan tombol “Nonaktifkan kelas” di kelas Kursus yang dipilih |  |
|  | 3c. Aktor belum mengisi semua *form* dengan lengkap | 3a. Sistem mengalihkan Aktor ke *form* ubah kelas  3b. Sistem memeriksa apakah kelas kursus yang ditawarkan > 1  3c. Sistem membuat tombol “Tawarkan kelas” tidak bisa diklik |
|  | 4a. Aktor mengubah semua informasi yang ingin diubah  4c. Aktor menekan tombol “Simpan Draft” | 4b-0, jika kelas kursus hanya 1, sistem akan mengembalikan pesan error dan menyarankan apakah Aktor lebih memilih untuk |

|  |  |  |
| --- | --- | --- |
|  | **Aktor** | **Sistem** |
|  | menutup kursus  4b-1, jika kelas kursus > 1, sistem akan menonaktifkan kelas yang dipilih |
| 5a. Aktor menekan tombol “Ubah kelas” | 5c. Sistem menyimpan *entry* yang dilakukan Aktor |
|  | 6a. Sistem menyimpan *entry* yang dilakukan Aktor  6c. Sistem mengalihkan Aktor ke halaman kursus |
|  | 7a. Sistem mengalihkan Aktor ke halaman kursus |

### Menampilkan Detail Progress Kursus untuk Pihak Kursus

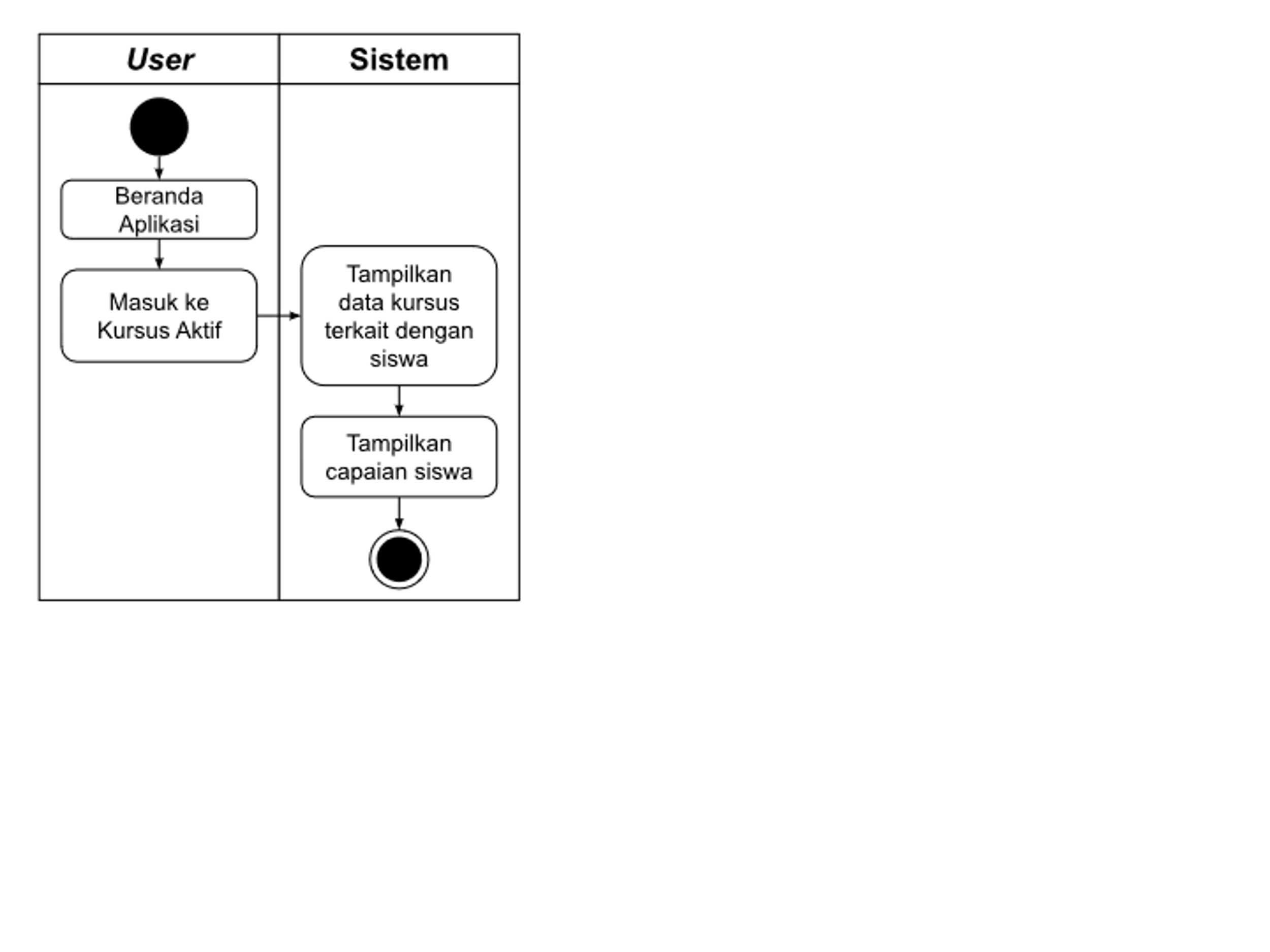


Pihak kursus dapat melihat detail dari kursus dari siswa yang dibawahi. Halaman detail *progress* ini menunjukkan instruktur di pertemuan ke berapakah siswa saat ini, serta apakah siswa sudah membaca teori yang diberikan, apakah *quiz* yang diberikan sudah dikerjakan, dan capaian-capaian lain.

|  |  |
| --- | --- |
| **Kode Use Case** | UC\_detail\_progress\_kursus\_instruktur |
| **Nama Use Case** | Menampilkan detail *progress* kursus |
| **Aktor** | Pemilik / Admin, Instruktur |
| **Deskripsi** | Skenario untuk menampilkan detail kursus aktif |
| **Kondisi Awal** | Aktor berada di *dashboard* |
| **Kondisi Akhir** | Aktor memperoleh informasi detail kursus dari sistem |

|  |  |  |
| --- | --- | --- |
| **Alur Kejadian Normal** | **Aktor** | **Sistem** |
| 1. Aktor memilih salah satu siswa dari daftar jadwal yang ditampilkan di *dashboard* |  |
|  | 2. Sistem menampilkan data kursus terkait dengan siswa yang dipilih Aktor |
|  | 3. Sistem menampilkan daftar capaian yang sudah dilakukan oleh siswa |
| **Alur Kejadian Alternatif** | - | - |

### Menampilkan Detail Progress Kursus untuk Siswa

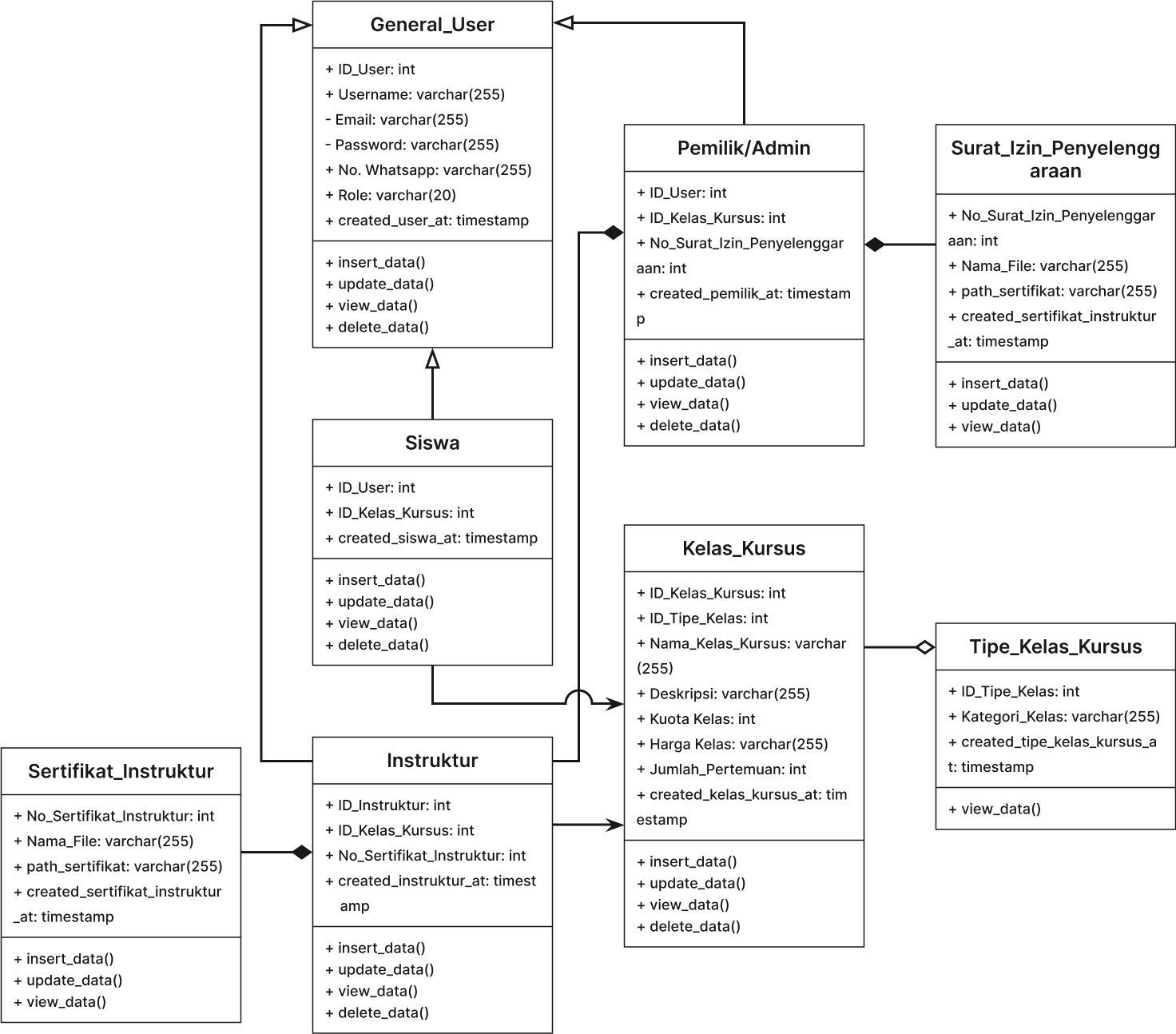


Siswa dengan kursus aktif dapat melihat detail dari kursus yang diikuti. Halaman detail *progress* inimemberikan siswa kemungkinan untuk mempelajari teori mengemudi lebih lanjut dengan memilih ‘Baca Panduan’ dan mengerjakan *quiz* yang diberikan oleh sistem.

|  |  |  |
| --- | --- | --- |
| **Kode Use Case** | UC\_detail\_progress\_kursus\_siswa | |
| **Nama Use Case** | Menampilkan detail *progress* kursus | |
| **Aktor** | *General User* (Siswa) | |
| **Deskripsi** | Skenario untuk menampilkan detail kursus aktif | |
| **Kondisi Awal** | Aktor berada di *dashboard* | |
| **Kondisi Akhir** | Aktor bisa memilih untuk perubahan jadwal, membaca panduan teori mengemudi, menyelesaikan Quiz, dan proses-proses lain | |
| **Alur Kejadian Normal** | 1. Aktor menekan ‘Progress Kursus’ pada bagian atas *dashboard* |  |
|  | 2. Sistem menampilkan informasi kursus yang dijalani siswa saat ini |
|  | 3. Sistem menampilkan capaian-capaian yang sudah dilakukan oleh siswa |
| **Alur Kejadian Alternatif** | 1. Aktor berpindah ke halaman profil |  |
| 2. Aktor menekan menu ‘Kursus Saya’ |  |

|  |  |  |
| --- | --- | --- |
| **Alur Kejadian Alternatif** |  | 3. Sistem menampilkan informasi kursus yang dijalani siswa saat ini |
|  | 4. Sistem menampilkan capaian-capaian yang sudah dilakukan oleh siswa |

Untuk rancangan desain *database* dari sistem kami menggambarkan struktur *database* dengan *class diagram*. Pada *class diagram* nantinya akan berisi atribut-atribut yang dimiliki masing-masing Aktor dan objek-objek lain, metode-metode yang dapat dieksekusi, dan relasi antar Aktor dan objek pada sistem nantinya. Berikut adalah rancangan *class diagram* untuk aplikasi penyedia jasa kursus mengemudi.



Dapat dilihat pada gambar diatas, *database* aplikasi untuk penyedia jasa kursus mengemudi nantinya akan terdapat 8 tabel, 4 tabel utama yang terdiri dari tabel Siswa, Pemilik/Admin, Instruktur, dan Kelas Kursus. 1 tabel untuk menggeneralisasi tabel-tabel pengguna, tabel yang dimaksud adalah tabel Siswa, Instruktur, dan Pemilik/Admin. 2 tabel yang bergantung pada tabel lain, tabel Sertifikat Instruktur bergantung pada tabel Instruktur, tabel Sertifikat Instruktur sendiri dibuat untuk mengakses Sertifikat Instruktur yang sudah di-*upload* ke sistem. Sama halnya dengan tabel Surat Izin Penyelenggaraan, digunakan untuk mengakses Surat Izin Penyelenggaraan Kursus, yang digantungkan dengan tabel Pemilik/Admin, untuk sebuah akun dapat menjadi Pemilik/Admin, akun tersebut harus memiliki Surat Izin Penyelenggaraan yang sah. Dan yang terakhir 1 tabel agregasi untuk mengkategorikan kelas kursus, sebagai contoh, apakah kelas yang ditawarkan untuk mobil dengan transmisi manual atau otomatis? atau apakah kelas yang ditawarkan sifatnya kursus kilat, kursus privat, atau kursus *reguler*? dan sebagainya.

## *Development*

Pada tahap selanjutnya, tim pengembang akan mengkonversi hasil desain menjadi sistem yang dapat dioperasikan di berbagai jenis perangkat, terutama smartphone semua pengguna. Pengembangan ini akan dilakukan berbasis web, sehingga tim pengembang akan menggunakan teknologi HTML, CSS, Javascript, Framework Laravel, dan MySQL untuk sistem manajemen database. Proses konversi desain menjadi sistem web akan melalui beberapa langkah berikut:

1. **Pemrograman *front-end***: Tim pengembang akan menggunakan HTML, CSS, dan *Javascript* untuk membangun tampilan dan interaksi antarmuka pengguna.
2. **Pemrograman *back-end***: Tim pengembang akan menggunakan *Framework* Laravel untuk membangun logika dan fungsionalitas sistem.
3. **Integrasi *database***: Tim pengembang akan menggunakan MySQL untuk menyimpan dan mengelola data sistem.
4. **Pengujian dan penyempurnaan**: Tim pengembang akan melakukan pengujian menyeluruh untuk memastikan sistem berjalan dengan baik dan sesuai dengan kebutuhan pengguna.

Detail lebih lanjut tentang proses pengembangan sistem web ini, termasuk teknologi yang digunakan, arsitektur sistem, dan hasil pengujian, akan dibahas pada Bab IV.

## *Test*

Salah satu metode pengujian yang kami lakukan adalah *Black Box Testing*. *Black Box Testing* adalah pengujian yang dilakukan untuk mengamati hasil eksekusi melalui data uji dan memeriksa fungsional dari perangkat lunak setelah proses pengembangan selesai (Zidan, M. et al., 2022). Fokus dari pengujian ini adalah menguji kinerja fungsi dari sistem, apakah sudah bekerja sesuai dengan yang diharapkan. Berikut adalah penggalan tabel *black box testing* dari aplikasi kami nantinya.

|  |  |  |  |
| --- | --- | --- | --- |
| **Kode** | **Skenario Tes** | **Hasil yang Diharapkan** | **Sesuai** |
| B01 | *Form* Login/Daftar Akun tidak diisi apa-apa | Sistem tidak dapat mengambil inputan yang diberikan dan menampilkan pesan “Pengisian *form* tidak sesuai” | ✔ |
| B02 | Pada *form* *login*, mengisi kolom *username* = “janedoe@gmail.com” dan *password* = “userlogin123” | Sistem mengambil inputan yang diberikan pengguna dan mengarahkan pengguna ke halaman beranda | ✔ |
| B03 | Pada *form* *login*, mengisi kolom *username* = “janedoe@gmail.com” dan *password* = “user1234” | Sistem mengambil inputan yang diberikan pengguna dan mengembalikan pesan *error* “Password yang dimasukkan tidak sesuai” | ✔ |

|  |  |  |  |
| --- | --- | --- | --- |
| **Kode** | **Skenario Tes** | **Hasil yang Diharapkan** | **Sesuai** |
| B04 | Pada *form login*, mengisi kolom *username =* “abcabc” dan *password* = “userlogin123” | Sistem mengambil inputan yang diberikan pengguna dan mengembalikan pesan *error* “Data Pengguna tidak ditemukan” | ✔ |
| **Jumlah Pengujian** | | | 4 |
| **Sesuai** | | | 2 |
| **Tidak Sesuai** | | | 2 |
| **Tingkat Keberhasilan** | | | 50% |

Selanjutnya, pada tahun 1996, John Brooke, mengembangkan sebuah skala untuk mengukur atau menilai kegunaan yang dirasakan peserta pengujian, yang dikenal dengan *System Usability Scale* (SUS). (Vlachogianni, P. et al., 2021) beranggapan mengapa SUS sangat populer bahkan sampai saat ini adalah karena SUS adalah alat pengukur psikologi yang handal dan tidak membutuhkan biaya dengan tingkat validasi tinggi dan dapat diandalkan. SUS terdiri dari 10 pernyataan yang secara bergantian menguji kesan positif dan negatif dari perasaan responden atau peserta pengujian terhadap sistem yang diujikan. Dibawah ini adalah 10 pertanyaan yang dirumuskan oleh John Brooke.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  | **Sangat Tidak Setuju** |  |  |  | **Sangat Setuju** |
| **No.** | **Pertanyaan** | **1** | **2** | **3** | **4** | **5** |
| 1 | Saya berpikir akan menggunakan sistem ini lagi |  |  |  |  |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  | **Sangat Tidak Setuju** |  |  |  | **Sangat Setuju** |
| **No.** | **Pertanyaan** | **1** | **2** | **3** | **4** | **5** |
| 2 | Saya merasa sistem ini rumit untuk digunakan |  |  |  |  |  |
| 3 | Saya merasa sistem ini mudah untuk digunakan |  |  |  |  |  |
| 4 | Saya membutuhkan bantuan dari orang lain atau teknisi dalam menggunakan sistem ini |  |  |  |  |  |
| 5 | Saya merasa fitur-fitur sistem ini berjalan dengan semestinya |  |  |  |  |  |
| 6 | Saya merasa ada banyak hal yang tidak konsisten (tidak serasi) pada sistem ini |  |  |  |  |  |
| 7 | Saya merasa orang lain akan memahami cara menggunakan sistem ini dengan cepat |  |  |  |  |  |
| 8 | Saya merasa sistem ini membingungkan |  |  |  |  |  |
| 9 | Saya merasa tidak ada hambatan dalam menggunakan sistem ini |  |  |  |  |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  | **Sangat Tidak Setuju** |  |  |  | **Sangat Setuju** |
| **No.** | **Pertanyaan** | **1** | **2** | **3** | **4** | **5** |
| 10 | Saya perlu membiasakan diri terlebih dahulu sebelum menggunakan sistem ini |  |  |  |  |  |

## *Maintenance*

Tahapan ini dapat dilakukan apabila di masa mendatang, dengan adanya kebutuhan-kebutuhan tambahan, pihak kursus merasa ada beberapa fungsi atau fitur yang perlu ditambahkan. Tahap pemeliharaan sangat penting untuk memastikan bahwa sistem tetap relevan dan memenuhi kebutuhan pengguna. Pemeliharaan yang berkelanjutan dapat membantu memperpanjang umur sistem. Serta, penting untuk memiliki rencana pemeliharaan yang jelas untuk sistem. Biaya pemeliharaan harus diperhitungkan dalam anggaran keseluruhan untuk proyek. Tim pengembang harus selalu siap untuk melakukan pemeliharaan pada sistem.

Selain penambahan fitur-fitur sesuai kebutuhan, proses-proses pemeliharaan aplikasi dapat dikelompokkan menjadi 3 kelompok:

* **Pemeliharaan Fungsional** menjamin bahwa fungsionalitas-fungsionalitas utama dari aplikasi bekerja sesuai yang diinginkan. Beberapa aktivitas-aktivitas yang dilakukan adalah : Memperbaiki *bug*, Pembaruan keamanan, Optimalisasi kinerja, Pembaruan kompatibilitas, dan Pembaruan fitur serta aktivitas-aktivitas lain.
* **Pemeliharaan Non Fungsional** mencakup aktivitas-aktivitas pemeliharaan kualitas dan pengalaman pengguna dari aplikasi secara keseluruhan diluar fungsionalitas utama. Aktivitas-aktivitas yang dilakukan diantaranya : Pembaruan konten, Pengujian kegunaan, Pengawasan kinerja, Pemeliharaan aksesibilitas, Pembaruan dokumentasi aplikasi, dan-lain-lain.
* **Pemeliharaan Infrastruktur** berfokus pada pemeliharaan-pemeliharaan infrastruktur yang mendukung aplikasi. Aktivitas-aktivitas yang tergolong pemeliharaan infrastruktur yaitu : Pemeliharaan *server*, Pemeliharaan jaringan, Pemeliharaan peluncuran, Pengelolaan skala aplikasi, dan aktivitas-aktivitas lain.