# BAB III

# METODE PENELITIAN

## Requirements

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.

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.



Gambar 3. Proses Pendaftaran pada Kursus Mengemudi ABC

Dapat dilihat pada gambar 3.1 diatas, 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.



Gambar 3. Proses Kursus di Kursus Mengemudi ABC

Sedangkan proses kursus di kursus mengemudi ABC sebagaimana yang dijelaskan pada gambar 3.2, 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.



Gambar 3. Proses Pendaftaran di Kursus Mengemudi Sie Bersaudara

Gambar 3.3 diatas 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.



Gambar 3. Proses Kursus di Kursus Mengemudi Sie Bersaudara

Gambar 3.4 diatas menunjukkan 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.



Gambar 3. Proses Pendaftaran di Kursus Mengemudi "Hafiz"

Berbeda dengan dua kursus sebelumnya, gambar 3.5 menjelaskan proses pendaftaran pada kursus mengemudi “Hafiz” yang 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.



Gambar 3. Proses Kursus di Kursus Mengemudi "Hafiz"

Sedangkan gambar 3.6 diatas menjelaskan proses kursus yang saat ini dilakukan pada kursus mengemudi “Hafiz”. 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. Tabel 3.1 dibawah ini adalah daftar kebutuhan aplikasi untuk penyedia jasa kursus mengemudi

Tabel 3. 1 Kebutuhan Aplikasi

|  |  |
| --- | --- |
| **No.** | **Kebutuhan / Persyaratan** |
| Sisi Admin Kursus (Lembaga) / Pemilik (Perorangan & Lembaga) | |
| 1 | Fungsi *Login* & *Logout* sebagai Admin / Pemilik |
| 2 | Dashboard yang berisi jadwal kursus setiap Instruktur |
| 3 | Mengubah data yang ada pada profil akun lembaga kursus |
| 4 | Menambah kelas kursus beserta deskripsi, harga, jumlah pertemuan, benefit, dan kategori kelas kursus |
| 5 | Mengubah nama, deskripsi, harga, jumlah pertemuan, benefit, dan kategori kelas kursus |
| 6 | Menghapus / menonaktifkan kelas kursus |
| 7 | Menambah Instruktur kursus |
| 8 | Mengundang Instruktur kursus |

Tabel 3. 1 Kebutuhan Aplikasi (Lanjutan-1)

|  |  |
| --- | --- |
| **No.** | **Kebutuhan / Persyaratan** |
| 9 | Menonaktifkan / menghapus data Instruktur kursus |
| 10 | Menampilkan daftar Siswa aktif |
| 11 | Menampilkan detail *progress* setiap Siswa |
| 12 | Mengajukan atau mengkonfirmasi perubahan jadwal kursus kepada Siswa dan Instruktur |
| 13 | Menampilkan hasil pengisian formulir pendaftaran kursus |
| 14 | Mengakses bukti pembayaran kursus dari Siswa |
| 15 | Menghubungi Siswa dan Instruktur |
| 16 | Mengkonfirmasi status pembayaran kursus dari Siswa |
| 17 | Mengunggah dokumen izin penyelenggaraan kursus mengemudi yang sah |
| Sisi Instruktur (Lembaga & Perorangan) | |
| 1 | Fungsi *Login* & *Logout* sebagai Instruktur |
| 2 | Dashboard yang berisi jadwal kursus per hari |
| 3 | Mengubah data yang ada pada profil akun Instruktur |
| 4 | Menampilkan daftar Siswa aktif |
| 5 | Menampilkan detail *progress* Siswa |
| 6 | Mengajukan atau mengkonfirmasi perubahan jadwal kursus kepada Siswa dan Admin / Pemilik |
| 7 | Menghubungi Siswa dan Admin / Pemilik |

Tabel 3. 1 Kebutuhan Aplikasi (Lanjutan-2)

|  |  |
| --- | --- |
| **No.** | **Kebutuhan / Persyaratan** |
| 8 | Mengisi form data instruktur dari hasil undangan Admin / Pemilik |
| 9 | Mengunggah Sertifikat hasil Pelatihan Instruktur Kursus Mengemudi |
| Sisi *General User* | |
| 1 | Fungsi *Login* & *Logout* sebagai *General User* |
| 2 | Dashboard yang berisi rekomendasi kelas kursus, penyedia kursus dan *progress* kursus yang sedang aktif jika ada |
| 3 | Mengubah data yang ada pada profil akun *General User* |
| 4 | Melakukan pengisian formulir pendaftaran kelas kursus |
| 5 | Mengunggah bukti pembayaran kelas kursus |
| 6 | Menghubungi Admin / Pemilik dan Instruktur |
| 7 | Menampilkan detail *progress* kursus yang berlangsung |
| 8 | Mengajukan atau mengkonfirmasi perubahan jadwal kursus kepada Instruktur dan Admin / Pemilik |
| 9 | Mengakses panduan teknik berkendara dan menyelesaikan quiz |
| 10 | Menjadi pemilik kursus dengan mengunggah surat izin penyelenggaraan kursus mengemudi |
| Kebutuhan Teknis | |
| 1 | Menggunakan bahasa pemrograman berbasis web seperti HTML, CSS, Javascript dan PHP lebih spesifik menggunakan *Framework* Laravel |

Tabel 3. 1 Kebutuhan Aplikasi (Lanjutan-3)

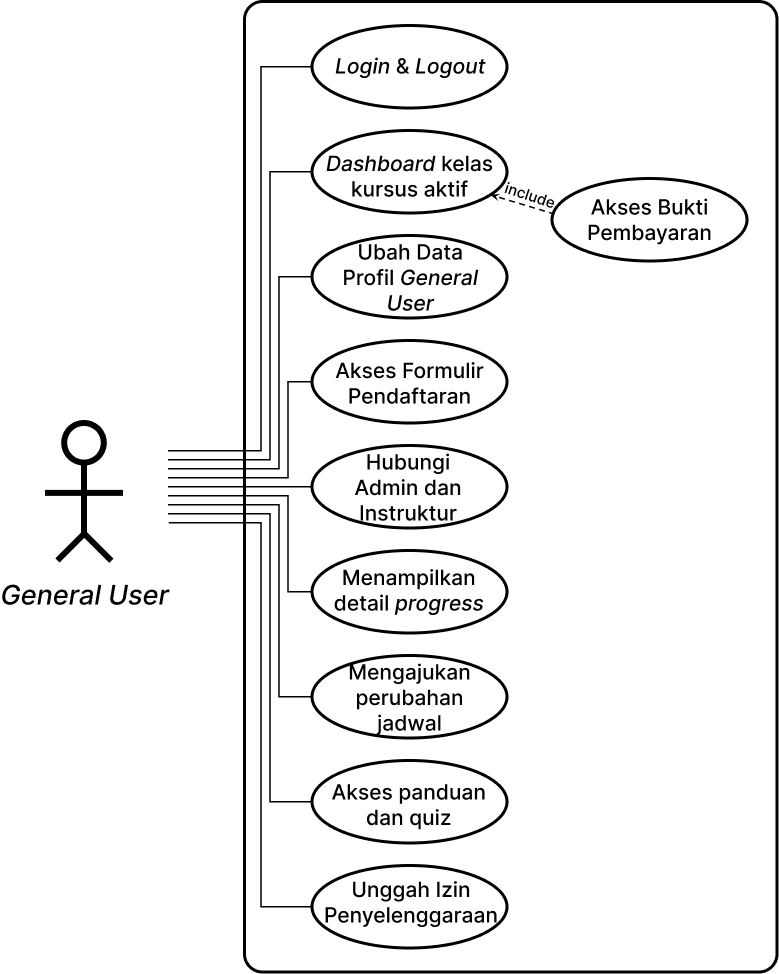
|  |  |
| --- | --- |
| **No.** | **Kebutuhan / Persyaratan** |
| 2 | Menggunakan sistem pengelolaan basis data relasional MySQL |
| 3 | Aplikasi dibangun untuk semua jenis *browser* |
| 4 | Spesifikasi perangkat keras yang disarankan adalah RAM diatas 2 GB |

## Design

Selama rangkaian proses perancangan atau desain dengan *incremental model* yang kami lakukan kami membagi tahap perancangan menjadi dua bagian, bagian pertama untuk perancangan struktur aplikasi, bagian ini kami gunakan untuk menjelaskan akses apa saja yang dimiliki oleh masing-masing pengguna sedangkan bagian kedua menjelaskan alur kerja sistem ketika sebuah fitur dieksekusi oleh pengguna.

### *Use Case*

Dari analisa kebutuhan diatas, kami mulai memahami kebutuhan dan kendala-kendala yang dihadapi oleh pihak penyedia jasa kursus mengemudi. Selanjutnya, untuk mempermudah tim pengembang memahami bagaimana struktur aplikasi, kami akan menggunakan bantuan *use case diagram* dan *use case scenario*. Gambar 3.7 adalah *use case diagram* untuk pengguna *General User.*

**

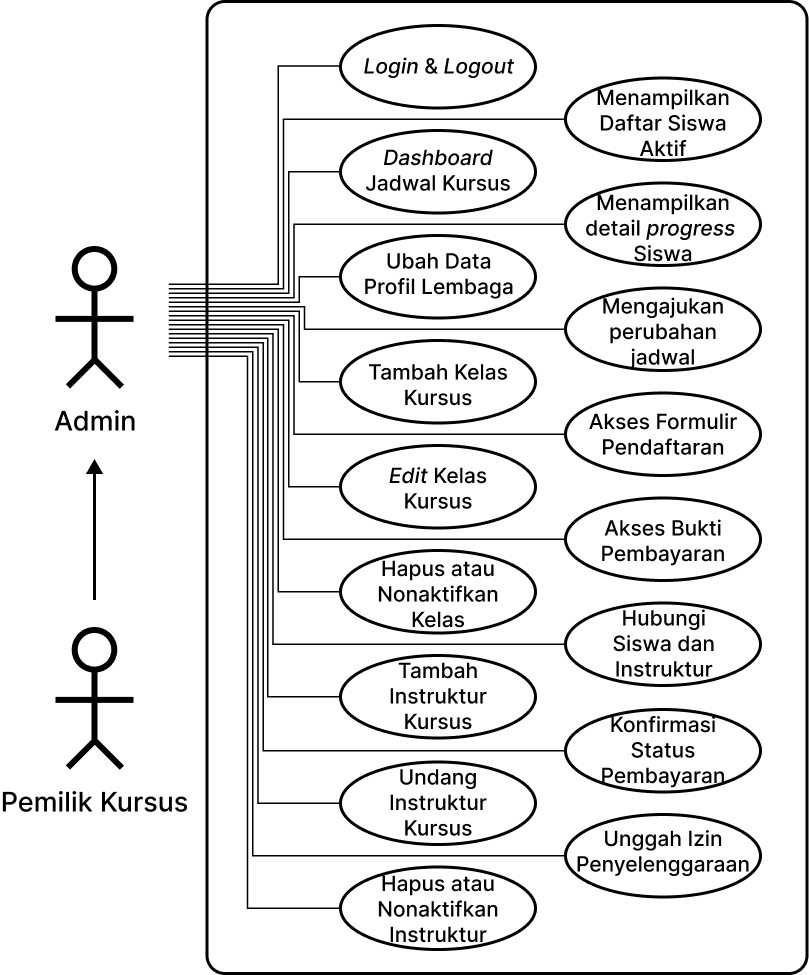
Gambar 3. *Use Case Diagram* untuk *General User*

Sebagai permulaan, berikut adalah *use case diagram* untuk pengguna *General User*, penyebutan *General User* dilandasi asumsi bahwa pengguna dengan peran tersebut dianggap sebagai masyarakat umum, dimana mereka bisa mendaftar sebagai siswa kursus atau mereka bisa juga mendaftar sebagai penyedia jasa dengan mengunggah dokumen izin penyelenggaraan dan melakukan proses-proses yang dibutuhkan selanjutnya. Untuk lebih jelasnya akan kami jelaskan melalui *use case scenario* di sub-bab selanjutnya. Gambar 3.8 dibawah adalah *use case diagram* untuk Instruktur Kursus.



Gambar 3. *Use Case Diagram* untuk Instruktur

Instruktur Kursus berperan sebagai pihak penengah antara Siswa dan Admin / Pemilik Kursus, untuk beberapa kasus, Pemilik Kursus juga bisa menjadi Instruktur Kursus. *Use case diagram* selanjutnya untuk pengguna Admin / Pemilik Kursus yang akan dijelaskan menggunakan Gambar 3.9 dibawah ini.



Gambar 3. *Use Case Diagram* untuk Admin / Pemilik

Admin / Pemilik memiliki peranan yang cukup penting dalam mengelola operasional kursus. Dapat dilihat pada diagram diatas bahwa Pemilik Kursus kami generalisasi menjadi Admin karena akses fitur yang dimiliki pada aplikasi sama, sehingga, kedua belah pihak dapat saling bekerja sama mengelola lembaga kursus mereka.

#### Proses *Login*

Proses *login* dimulai dari pengguna berada pada halaman *login*. proses *login* dibutuhkan untuk mencatat semua aktivitas yang dilakukan oleh pengguna. Tabel dibawah ini menjelaskan tentang interaksi antar pengguna dengan respon sistem.

Tabel 3. *Use Case Scenario* untuk proses *Login*

|  |  |
| --- | --- |
| **Kode Use Case** | UC\_login\_untuk\_semua\_user |
| **Nama Use Case** | *Login* |
| **Aktor** | *General User*, Pemilik / Admin, Instruktur |

Tabel 3. *Use Case Scenario* untuk proses *Login* (Lanjutan-1)

|  |  |  |
| --- | --- | --- |
| **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 dua tombol (*Login* atau Daftar dan Lewati) |
| 2. Aktor memilih tombol *Login* atau Daftar |  |
|  | 3. Sistem menampilkan *form* *Login* |
| 4. Aktor memasukkan data akun yang tersimpan di *database* |  |
|  | 5. Sistem memverifikasi data yang diinputkan Aktor |
|  | 6. Sistem mengarahkan Aktor ke tampilan Beranda |
| **Alur Kejadian Alternatif** | 2a. Aktor memilih tombol “Daftar”  2b. Aktor memilih tombol “Nanti” |  |

Tabel 3. *Use Case Scenario* untuk proses *Login* (Lanjutan-2)

|  |  |  |
| --- | --- | --- |
|  | **Aktor** | **Sistem** |
|  | 3a. Sistem menampilkan *form* daftar akun  3b. Sistem memberikan peran Aktor sebagai *General User* |
| 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 |

#### Proses Siswa Mendaftar Kelas 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. Tabel dibawah ini menjelaskan tentang interaksi antar pengguna dengan respon sistem saat proses pendaftaran kelas kursus berlangsung.

Tabel 3. *Use Case Scenario* untuk proses Siswa Mendaftar Kursus

|  |  |
| --- | --- |
| **Kode Use Case** | UC\_siswa\_daftar\_kursus |
| **Nama Use Case** | Siswa Mendaftar Kursus |
| **Aktor** | Siswa & Pemilik/Admin |

Tabel 3. *Use Case Scenario* untuk proses Siswa Mendaftar Kursus (Lanjutan-1)

|  |  |  |
| --- | --- | --- |
| **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** | 3a. Calon Siswa tidak jadi meneruskan pendaftaran |  |

Tabel 3. *Use Case Scenario* untuk proses Siswa Mendaftar Kursus (Lanjutan-2)

|  |  |  |
| --- | --- | --- |
|  | **Aktor** | **Sistem** |
| 3b. Calon Siswa belum menyelesaikan proses *login* atau daftar |  |
|  | 4a. Sistem mengalihkan Calon Siswa ke beranda/*dashboard*  4b. Sistem mengarahkan Calon Siswa ke tampilan *form login* |

#### Proses Pembayaran

Pada sistem yang kami kembangkan proses pembayaran terjadi di luar sistem, sistem hanya dapat digunakan untuk mengunggah bukti pembayaran dan memberitahukan kedua pihak status pembayaran saat ini. Tabel berikut menjelaskan interaksi antar pengguna dan respon sistem pada saat proses pembayaran terjadi.

Tabel 3. *Use Case Scenario* untuk proses Pembayaran

|  |  |  |
| --- | --- | --- |
| **Kode Use Case** | UC\_pembayaran | |
| **Nama Use Case** | 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. Calon Siswa menghubungi pihak kursus menanyakan tentang proses pembayaran |  |
| 3. Calon Siswa melakukan proses pembayaran |  |
| 4. Pemilik/Admin memverifikasi pembayaran |  |
| 5. Pemilik/Admin mengubah status pembayaran |  |

Tabel 3. *Use Case Scenario* untuk proses Pembayaran (Lanjutan-1)

|  |  |  |
| --- | --- | --- |
| **Alur Kejadian Normal** | **Aktor** | **Sistem** |
|  | 6. Sistem menampilkan status pembayaran yang dipilih Admin |
| **Alur Kejadian Alternatif** | - | - |

#### Proses 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. Tabel dibawah ini menjelaskan interaksi antar pengguna dengan respon sistem pada saat proses perubahan jadwal kursus.

Tabel 3. *Use Case Scenario* untuk proses Mengubah Jadwal Kursus

|  |  |  |
| --- | --- | --- |
| **Kode Use Case** | UC\_mengubah\_jadwal\_kursus | |
| **Nama Use Case** | Mengubah Jadwal Kursus | |
| **Aktor** | Siswa, Instruktur atau Admin/Pemilik Kursus | |
| **Deskripsi** | Skenario pengajuan perubahan jadwal | |
| **Kondisi Awal** | Aktor berada di halaman detail *progress* kursus | |
| **Kondisi Akhir** | Aktor 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 |

Tabel 3. *Use Case Scenario* untuk proses Mengubah Jadwal Kursus (Lanjutan-1)

|  |  |  |
| --- | --- | --- |
|  | **Aktor** | **Sistem** |
|  | 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 |  |
| **Alur Kejadian Alternatif** | **Aktor** | **Sistem** |
| 7. Aktor menekan tombol ‘Ajukan Perubahan Jadwal’ |  |
|  | 8. Sistem mengirimkan pemberitahuan ke Aktor lain untuk dikonfirmasi pengajuan jadwal yang dikirim |
| 9a. Aktor lainnya mengkonfirmasi perubahan jadwal |  |

Tabel 3. *Use Case Scenario* untuk proses Mengubah Jadwal Kursus (Lanjutan-2)

|  |  |  |
| --- | --- | --- |
|  | **Aktor** | **Sistem** |
| 9b. Aktor lainnya menolak perubahan jadwal |  |
|  | 10a. Sistem mengirimkan pemberitahuan pengajuan jadwal berhasil ke semua Aktor  10b. Sistem mengirimkan pemberitahuan pengajuan jadwal ditolak ke semua Aktor |

#### Proses Pengajuan Jasa Kursus Mengemudi 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*. Untuk memahami interaksi antar pengguna dan respon sistem akan kami jelaskan menggunakan tabel dibawah.

Tabel 3. *Use Case Scenario* untuk proses Mengajukan Kursus Baru

|  |  |  |
| --- | --- | --- |
| **Kode Use Case** | UC\_mengajukan\_kursus\_baru | |
| **Nama Use Case** | Mengajukan Kursus Baru | |
| **Aktor** | Instruktur, *General User* (Calon Pemilik) & *System 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 |

Tabel 3. *Use Case Scenario* untuk proses Mengajukan Kursus Baru (Lanjutan-1)

|  |  |  |
| --- | --- | --- |
|  | **Aktor** | **Sistem** |
| 5. Calon Pemilik mengupload dokumen perizinan penyelenggaraan kursus |  |
|  | 6. *System Admin* memverifikasi keaslian dokumen yang diupload oleh Calon Pemilik |
|  | 7. Sistem mengirim pesan berhasil ke Calon Pemilik kemudian mengarahkan Calon Pemilik untuk mengisi data Instruktur |
|  | 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 |  |

Tabel 3. *Use Case Scenario* untuk proses Mengajukan Kursus Baru (Lanjutan-2)

|  |  |  |
| --- | --- | --- |
|  | **Aktor** | **Sistem** |
| 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 halaman pendaftaran akun untuk diisi Calon Instruktur |
| 13b. Calon Instruktur mengisi *form* penambahan Instruktur |  |
|  | 14b. Sistem menyimpan *entry* yang dilakukan Instruktur |

Tabel 3. *Use Case Scenario* untuk proses Mengajukan Kursus Baru (Lanjutan-3)

|  |  |  |
| --- | --- | --- |
| **Alur Kejadian Alternatif** | **Aktor** | **Sistem** |
|  | 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 |

#### ProsesAkses *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. Tabel dibawah akan menjelaskan respon sistem akan interaksi yang dilakukan oleh pengguna.

Tabel 3. *Use Case Scenario* untuk proses menampilkan *Dashboard* Jadwal Kursus

|  |  |
| --- | --- |
| **Kode Use Case** | UC\_tampilkan\_jadwal\_kursus |
| **Nama Use Case** | Menampilkan *dashboard* jadwal kursus |

Tabel 3. *Use Case Scenario* untuk proses menampilkan *Dashboard* Jadwal Kursus (Lanjutan-1)

|  |  |  |
| --- | --- | --- |
| **Aktor** | Pemilik / Admin, Instruktur | |
| **Deskripsi** | Skenario untuk menampilkan jadwal kursus per hari | |
| **Kondisi Awal** | Aktor baru saja menyelesaikan 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 |  |

Tabel 3. *Use Case Scenario* untuk proses menampilkan *Dashboard* Jadwal Kursus (Lanjutan-2)

|  |  |  |
| --- | --- | --- |
|  | **Aktor** | **Sistem** |
|  | 5a. Sistem menampilkan jadwal dimulai dengan hari ini sampai 14 hari kedepan |
| 4b. Aktor memilih opsi untuk tampilkan jadwal dalam 30 hari |  |
|  | 5b. Sistem menampilkan jadwal dimulai dengan hari ini sampai 29 hari kedepan |

#### Proses Tambah 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.

Tabel 3. *Use Case Scenario* untuk Mengelola Kelas Kursus

|  |  |  |
| --- | --- | --- |
| **Kode Use Case** | UC\_pengelolaan\_kelas\_kursus | |
| **Nama Use Case** | Pengelolaan 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 |

Tabel 3. *Use Case Scenario* untuk Mengelola Kelas Kursus (Lanjutan-1)

|  |  |  |
| --- | --- | --- |
|  |  | 6. Sistem mengarahkan Aktor ke halaman kursus dan menampilkan kelas yang baru ditambahkan pada 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 |

Tabel 3. *Use Case Scenario* untuk Mengelola Kelas Kursus (Lanjutan-2)

|  |  |  |
| --- | --- | --- |
|  | **Aktor** | **Sistem** |
| 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 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 |

#### Proses Akses 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. Tabel dibawah akan menjelaskan interaksi antar pengguna dan sistem ketika proses dieksekusi.

Tabel 3. *Use Case Scenario* untuk proses Menampilkan Detail Progress Kursus untuk Pihak Kursus

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

Tabel 3. *Use Case Scenario* untuk proses Menampilkan Detail Progress Kursus untuk Pihak Kursus (Lanjutan-1)

|  |  |  |
| --- | --- | --- |
| **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** | - | - |

#### Proses Akses 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. Dibawah ini adalah tabel untuk menjelaskan interaksi antar pengguna dengan respon sistem.

Tabel 3. *Use Case Scenario* untuk proses Menampilkan Detail Progress Kursus untuk Pihak Kursus

|  |  |
| --- | --- |
| **Kode Use Case** | UC\_detail\_progress\_kursus\_siswa |
| **Nama Use Case** | Detail *progress* kursus |
| **Aktor** | *General User* (Siswa) |
| **Deskripsi** | Skenario untuk menampilkan detail kursus aktif |
| **Kondisi Awal** | Aktor berada di *dashboard* |

Tabel 3. *Use Case Scenario* untuk proses Menampilkan Detail Progress Kursus untuk Pihak Kursus (Lanjutan-1)

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

#### *Log Out*

Apabila Aktor merasa tidak ada operasi atau aktivitas lain yang akan dikerjakan, Aktor dapat memilih untuk melakukan *logout*, dimana semua proses Aktor tidak akan lagi tercatat oleh sistem, kemudian sistem mengarahkan Aktor ke tampilan awal aplikasi. Proses *log out* akan dijelaskan menggunakan tabel dibawah ini.

Tabel 3. *Use Case Scenario* untuk Proses Logout

|  |  |
| --- | --- |
| **Kode Use Case** | UC\_log\_out |
| **Nama Use Case** | *Log Out* |
| **Aktor** | *General User*, Instruktur, Pemilik/Admin |
| **Deskripsi** | Skenario untuk melakukan *log out* |
| **Kondisi Awal** | Aktor berada di *dashboard* |
| **Kondisi Akhir** | Aktor berada di tampilan awal aplikasi |

|  |  |  |
| --- | --- | --- |
| **Alur Kejadian Normal** | 1. Aktor menekan icon ‘burger’ di bagian kiri atas layar |  |
|  | 2. Sistem menampilkan navbar |
| 3. Aktor menekan menu “Keluar” |  |
|  | 4. Sistem menghentikan *session* Aktor dan mengarahkan Aktor ke tampilan awal aplikasi |
| **Alur Kejadian Normal** | - | - |

### *Activity Diagram*

### *Class Diagram*

Untuk rancangan desain *database* dari aplikasi, 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.



Gambar 3. *Class Diagram* Aplikasi untuk 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.

Tabel 3. Contoh Tabel *Black Box Testing*

|  |  |  |  |
| --- | --- | --- | --- |
| **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” | ✔ |
| 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” | ✔ |

Tabel 3. Contoh Tabel *Black Box Testing* (Lanjutan-1)

|  |  |  |  |
| --- | --- | --- | --- |
| **Kode** | **Skenario Tes** | **Hasil yang Diharapkan** | **Sesuai** |
| **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.

Tabel 3. Contoh Tabel Pengujian *System Usability Scale*

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

Tabel 3. Contoh Tabel Pengujian *System Usability Scale* (Lanjutan-1)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  | **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 |  |  |  |  |  |

Tabel 3. Contoh Tabel Pengujian *System Usability Scale* (Lanjutan-2)

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

## Implementation